



January 2015

Clustering And Control Of Streaming Synchrophasor Datasets

Anupam Mukherjee

Follow this and additional works at: <https://commons.und.edu/theses>

Recommended Citation

Mukherjee, Anupam, "Clustering And Control Of Streaming Synchrophasor Datasets" (2015). *Theses and Dissertations*. 1936.
<https://commons.und.edu/theses/1936>

This Thesis is brought to you for free and open access by the Theses, Dissertations, and Senior Projects at UND Scholarly Commons. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of UND Scholarly Commons. For more information, please contact zeinebyousif@library.und.edu.

CLUSTERING AND CONTROL OF STREAMING SYNCHROPHASOR DATASETS

by

Anupam Mukherjee
Bachelor of Technology, West Bengal University of Technology

A Thesis

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements


for the degree of

Master of Science

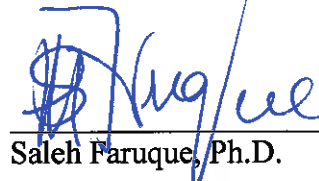
Grand Forks, North Dakota
December
2015

© 2015 Anupam Mukherjee

This thesis, submitted by Anupam Mukherjee in partial fulfillment of the requirements for the Degree of Master of Science from the University of North Dakota, has been read by the Faculty Advisory Committee under whom the work has been done and is hereby approved.


 12/09/15
Prakash Ranganathan, Ph.D., Chairperson

 12/09/15
Naima Kaabouch, Ph.D.

 12/09/15
Saleh Faruque, Ph.D.

This thesis meets the standards for appearance, conforms to the style and format requirements of the Graduate School of the University of North Dakota, and is hereby approved.


Wayne Swisher
Dean of the Graduate School


Date

PERMISSION

Title Clustering and control of streaming synchrophasor datasets
Department Electrical Engineering
Degree Master of Science

In presenting this thesis in partial fulfillment of the requirements for a graduate degree from the University of North Dakota, I agree that the library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by the professor who supervised my thesis work or, in his absence, by the chairperson of the department or the dean of the Graduate School. It is understood that any copying or publication or other use of this thesis or part thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of North Dakota in any scholarly use which may be made of any material in my thesis.

Anupam Mukherjee
December 2015

TABLE OF CONTENTS

LIST OF FIGURES	viii
LIST OF TABLES	xv
ACKNOWLEDGEMENTS	xix
ABBREVIATIONS	xx
ABSTRACT	xxiv
CHAPTERS	
I. INTRODUCTION	1
1.1 Synchrophasors/Phasor Measurement Units (PMU's)	3
1.2 Utilization of Synchrophasor Data	6
1.3 Thesis Contributions.....	11
1.4 Thesis Organization.....	12
II. LITERATURE REVIEW	14
2.1 Data mining and visualization	14
2.2 Forecasting techniques	20
III. CLUSTERING METHODS AND openPDC INTERFACE	27
3.1 openPDC set-up and Data Extraction Process.....	27
3.2 Application of k-means clustering.....	29
3.3 Application of Density-based spatial clustering of applications with noise (DBSCAN) Clustering.....	33
3.4 Need for a hybrid method.....	35
IV. FORECASTING OF ELECTRIC UTILITY DATASETS.....	45

4.1 Forecasting in electric utilities.....	45
4.2 Investigation of forecasting models and need for novel forecasting	51
4.3 LOWESS model	52
4.4 Random forest (RF) model	56
4.5 Seasonal autoregressive integrated moving average (SARIMA)	60
4.6 Developing a hybrid method	68
V. RESULTS AND DISCUSSIONS	77
5.1 Results obtained from data clustering algorithms	77
5.2 Results obtained from data forecasting algorithms	94
VI. CONCLUSION AND FUTURE WORK	175
6.1 Conclusion.....	175
6.2 Future Work.....	177
PUBLICATIONS.....	179
APPENDICES	181
APPENDIX A.....	183
CLUSTER FORMATION AND DATA VISUALIZATION	183
1. C# CODE FOR DATA ACQUISITION FROM openPDC.....	183
2. MATLAB CODE FOR CLUSTERING AND VISUALIZATION	186
2.1 MATLAB CODE FOR VOLTAGE DATA	187
2.1 MATLAB CODE FOR CURRENT DATA.....	200
APPENDIX B	214
FORECASTING OF LOAD DATA.....	214

1. MATLAB CODE FOR DATA ACQUISITION FROM '.xls' FILE	214
2. R CODE FOR FORECASTING	216
3. MATLAB CODE FOR ERROR CALCULATION AND PLOTTING DATA	217
REFERENCES	219

LIST OF FIGURES

Figure	Page
1. A fully connected Smart Grid Network	2
2. Digital Layer of a Smart Grid	3
3. Functional Block Diagram of a PMU	4
4. A Wide Area Monitoring System (WAMS)	5
5. System separation, August 2003	7
6. Redrawn phase angle with linear scale	8
7. Smart Grid Management Framework (SGDMF).....	12
8. Data Extraction and Conversion Stage	29
9. Data Mining and Visualization providing basis for decision making.....	29
10. Pseudo-code for k-means Clustering	32
11. k-means cluster formation.....	33
12. Growth of clusters based on their density.....	34
13. Pseudo-code for DBSCAN clustering	35
14. DBSCAN Cluster formation.....	36
15. Integrated software framework for PMU/AMI data	37
16. Multi-Tier k-means Cluster Categories	39
17. Pseudocode of multi-tier k-means (Phase I & II)	42
18. Cluster formation in multi-tier k-means	43
19. Load vs Temperature scatter plot for Mid-Atlantic region.....	48
20. Load vs hour of day for business day and holiday.....	49
21. LOWESS curve fitting.....	54

22. Pseudocode for LOWESS Smoothing	56
23. Ozone vs. Temperature plot	57
24. Decision tree function layout	58
25. Random Forest formation from decision trees.....	59
26. Pseudocode for Random Forest	60
27. Demand power plot for the year 2014 Mid-Atlantic region	65
28. Demand power plot between 06/25/2014 and 06/26/2014 for Mid-Atlantic region ..	66
29. Functional block diagram of RF + SARIMA process	76
30. Steady State Voltage Output – DBSCAN.....	79
31. Steady State Voltage output – Multi-tier k-means.....	80
32. Steady State Voltage output – k-means	81
33. Heavy Load Condition – DBSCAN.....	81
34. Heavy Load Condition – Multi-tier k-means.....	82
35. Heavy Load Condition - k-means.....	82
36. Light Load Condition – DBSCAN	83
37. Light Load Condition – Multi-tier k-means.....	84
38. Light Load condition – k-means	84
39. Line to Ground Fault – DBSCAN	85
40. Line to Ground Fault – Multi-Tier k-means	86
41. Line to Ground Fault – k-means	86
42. Steady state condition multi-tier k-means.....	87
43. Heavy load condition multi-tier k-means	88
44. Heavy load condition multi-tier k-means	89

45. Comparison of hourly load data for the months of Mar-May in 2012.....	95
46. Demand power for Mid-Atlantic region on Jan 3 rd , 2012.....	96
47. MAE values for May 2015 projection (MTLF 2012-2015) at 2am.....	98
48. MSE values for May 2015 projection (MTLF 2012-2015) at 2am	98
49. RMSE values for May 2015 projection (MTLF 2012-2015) at 2am.....	99
50. MAPE values for May 2015 projection (MTLF 2012-2015) at 2am.....	99
51. MAE values for May 2015 projection (MTLF 2012-2015) at 9am.....	101
52. MSE values for May 2015 projection (MTLF 2012-2015) at 9am	101
53. RMSE values for May 2015 projection (MTLF 2012-2015) at 9am.....	102
54. MAPE values for May 2015 projection (MTLF 2012-2015) at 9am.....	102
55. MAE values for May 2015 projection (MTLF 2012-2015) at 2pm.....	104
56. MSE values for May 2015 projection (MTLF 2012-2015) at 2pm	104
57. RMSE values for May 2015 projection (MTLF 2012-2015) at 2pm.....	105
58. MAPE values for May 2015 projection (MTLF 2012-2015) at 2pm	105
59. MAE values for May 2015 projection (MTLF 2012-2015) at 8pm.....	107
60. MSE values for May 2015 projection (MTLF 2012-2015) at 8pm	107
61. RMSE values for May 2015 projection (MTLF 2012-2015) at 8pm.....	108
62. MAPE values for May 2015 projection (MTLF 2012-2015) at 8pm	108
63. MAE values for Aug 2015 projection (MTLF 2012-2015) at 2am	110
64. MSE values for Aug 2015 projection (MTLF 2012-2015) at 2am.....	110
65. RMSE values for Aug 2015 projection (MTLF 2012-2015) at 2am	111
66. MAPE values for Aug 2015 projection (MTLF 2012-2015) at 2am	111
67. MAE values for Aug 2015 projection (MTLF 2012-2015) at 9am	113

68. MSE values for Aug 2015 projection (MTLF 2012-2015) at 9am.....	113
69. RMSE values for Aug 2015 projection (MTLF 2012-2015) at 9am	114
70. MAPE values for Aug 2015 projection (MTLF 2012-2015) at 9am.....	114
71. MAE values for Aug 2015 projection (MTLF 2012-2015) at 2pm.....	116
72. MSE values for Aug 2015 projection (MTLF 2012-2015) at 2pm.....	116
73. RMSE values for Aug 2015 projection (MTLF 2012-2015) at 2pm	117
74. MAPE values for Aug 2015 projection (MTLF 2012-2015) at 2pm.....	117
75. MAE values for Aug 2015 projection (MTLF 2012-2015) at 8pm.....	119
76. MSE values for Aug 2015 projection (MTLF 2012-2015) at 8pm.....	119
77. RMSE values for Aug 2015 projection (MTLF 2012-2015) at 8pm.....	120
78. MAPE values for Aug 2015 projection (MTLF 2012-2015) at 8pm.....	120
79. MAE values for Nov 2014 projection (MTLF 2012-2014) at 2am	122
80. MSE values for Nov 2014 projection (MTLF 2012-2014) at 2am.....	122
81. RMSE values for Nov 2014 projection (MTLF 2012-2014) at 2am	123
82. MAPE values for Nov 2014 projection (MTLF 2012-2014) at 2am.....	123
83. MAE values for Nov 2014 projection (MTLF 2012-2014) at 9am	125
84. MSE values for Nov 2014 projection (MTLF 2012-2014) at 9am.....	125
85. RMSE values for Nov 2014 projection (MTLF 2012-2014) at 9am	126
86. MAPE values for Nov 2014 projection (MTLF 2012-2014) at 9am.....	126
87. MAE values for Nov 2014 projection (MTLF 2012-2014) at 2pm.....	128
88. MSE values for Nov 2014 projection (MTLF 2012-2014) at 2pm.....	128
89. RMSE values for Nov 2014 projection (MTLF 2012-2014) at 2pm.....	129
90. MAPE values for Nov 2014 projection (MTLF 2012-2014) at 2pm.....	129

91. MAE values for Nov 2014 projection (MTLF 2012-2014) at 8pm	131
92. MSE values for Nov 2014 projection (MTLF 2012-2014) at 8pm.....	131
93. RMSE values for Nov 2014 projection (MTLF 2012-2014) at 8pm.....	132
94. MAPE values for Nov 2014 projection (MTLF 2012-2014) at 8pm.....	132
95. MAE values for Dec 2014 projection (MTLF 2012-2014) at 2am.....	134
96. MSE values for Dec 2014 projection (MTLF 2012-2014) at 2am	134
97. RMSE values for Dec 2014 projection (MTLF 2012-2014) at 2am.....	135
98. MAPE values for Dec 2014 projection (MTLF 2012-2014) at 2am	135
99. MAE values for Dec 2014 projection (MTLF 2012-2014) at 9am.....	137
100. MSE values for Dec 2014 projection (MTLF 2012-2014) at 9am	137
101. RMSE values for Dec 2014 projection (MTLF 2012-2014) at 9am.....	138
102. MAPE values for Dec 2014 projection (MTLF 2012-2014) at 9am	138
103. MAE values for Dec 2014 projection (MTLF 2012-2014) at 2pm	140
104. MSE values for Dec 2014 projection (MTLF 2012-2014) at 2pm.....	140
105. RMSE values for Dec 2014 projection (MTLF 2012-2014) at 2pm	141
106. MAPE values for Dec 2014 projection (MTLF 2012-2014) at 2pm	141
107. MAE values for Dec 2014 projection (MTLF 2012-2014) at 8pm	143
108. MSE values for Dec 2014 projection (MTLF 2012-2014) at 8pm.....	143
109. RMSE values for Dec 2014 projection (MTLF 2012-2014) at 8pm	144
110. MAPE values for Dec 2014 projection (MTLF 2012-2014) at 8pm	144
111. Comparison of MAE values for the projections for May 2015	145
112. Comparison of MAE values for the projections for August 2015	146
113. Comparison of MAE values for the projections for November 2014.....	146

114. Comparison of MAE values for the projections for December 2014	147
115. Comparison of MSE values for the projections for May 2015	147
116. Comparison of MSE values for the projections for August 2015.....	148
117. Comparison of MSE values for the projections for November 2014	148
118. Comparison of MSE values for the projections for December 2014.....	149
119. Comparison of RMSE values for the projections for May 2015	149
120. Comparison of RMSE values for the projections for August 2015	150
121. Comparison of RMSE values for the projections for November 2014.....	150
122. Comparison of RMSE values for the projections for December 2014	151
123. Comparison of MAPE values for the projections for May 2015	151
124. Comparison of MAPE values for the projections for August 2015	152
125. Comparison of MAPE values for the projections for November 2014	152
126. Comparison of MAPE values for the projections for December 2014.....	153
127. Comparison of MAE values for the projections at 2am	153
128. Comparison of MAE values for the projections at 9am	154
129. Comparison of MAE values for the projections at 2pm	154
130. Comparison of MAE values for the projections at 8pm	155
131. Comparison of MSE values for the projections at 2am	155
132. Comparison of MSE values for the projections at 9am	156
133. Comparison of MSE values for the projections at 2pm.....	156
134. Comparison of MSE values for the projections at 8pm.....	157
135. Comparison of RMSE values for the projections at 2am	157
136. Comparison of RMSE values for the projections at 9am	158

137. Comparison of RMSE values for the projections at 2pm	158
138. Comparison of RMSE values for the projections at 8pm	159
139. Comparison of MAPE values for the projections at 2am	159
140. Comparison of MAPE values for the projections at 9am	160
141. Comparison of MAPE values for the projections at 2pm	160
142. Comparison of MAPE values for the projections at 8pm	161
143. ACF and pACF plot for residuals from 1/1/2012 to 4/30/2015 at 2am	165
144. Standardized residuals plot for the period 1/1/2012 - 4/30/2015 at 2am	165
145. LOWRIMA predicted load vs actual load for 6/25/2015	167
146. Error plot for hourly predicted load using LOWRIMA data on 6/25/2015	167
147. RFSRIMA predicted load vs actual load for 6/25/2015	169
148. Error plot for hourly predicted load using RFSRIMA data on 6/25/2015	170
149. MAE for values projected quarterly by LOWRIMA on 6/25/2015	170
150. MSE for values projected quarterly by LOWRIMA on 6/25/2015	171
151. RMSE for values projected quarterly by LOWRIMA on 6/25/2015	171
152. MAPE for values projected quarterly by LOWRIMA on 6/25/2015	172
153. MAE for values projected quarterly by RFSRIMA on 6/25/2015	172
154. MAE for values projected quarterly by RFSRIMA on 6/25/2015	173
155. MAE for values projected quarterly by RFSRIMA on 6/25/2015	173
156. MAPE for values projected quarterly by RFSRIMA on 6/25/2015	174

LIST OF TABLES

Table	Page
1. Update cycle for forecasting	47
2. Classification of load forecasts	51
3. Sample dataset for all business days at 2am	71
4. Name of predictor variables with their types	72
5. Possible level and label names for variable Month	72
6. Possible level and label names for variable Season.....	73
7. Possible level and label names for variable Day of Week.....	73
8. Input arguments and their properties for randomForest	74
9. Calculating residuals from RF model	74
10. RFSRIMA Predicted values for load data at 2 am.....	75
11. Distribution of data points with multi-tier k-means.....	89
12. Distribution of data points with DBSCAN	90
13. Distribution of data points with k-means.....	90
14. Computation time for different data sizes.....	92
15. Visual Fidelity of clustering algorithms	92
16. Dunn's indices for voltage and current data.....	93
17. Demand data predicted by each model for May 2015 at 2am	97
18. Prediction error percentages for May 2015 at 2am.....	97
19. Standardized error for each model at 2am on May 2015.....	98
20. Demand data predicted by each model for May 2015 at 9am	100
21. Prediction error percentages for May 2015 at 9am.....	100
22. Standardized error for each model at 9am on May 2015.....	101

23. Demand data predicted by each model for May 2015 at 2pm	103
24. Prediction error percentages for May 2015 at 2pm	103
25. Standardized error for each model at 2pm on May 2015.....	104
26. Demand data predicted by each model for May 2015 at 8pm	106
27. Prediction error percentages for May 2015 at 8pm	106
28. Standardized error for each model at 8pm on May 2015.....	107
29. Demand data predicted by each model for August 2015 at 2am	109
30. Prediction error percentages for August 2015 at 2am	109
31. Standardized error for each model at 2am on August 2015.....	110
32. Demand data predicted by each model for August 2015 at 9am	112
33. Prediction error percentages for August at 2015 9am	112
34. Standardized error for each model at 9am on August 2015.....	113
35. Demand data predicted by each model for August 2015 at 2pm.....	115
36. Prediction error percentages for August at 2015 2pm	115
37. Standardized error for each model at 2pm on August 2015	116
38. Demand data predicted by each model for August 2015 at 8pm	118
39. Prediction error percentages for August at 2015 8pm	118
40. Standardized error for each model at 8pm on August 2015	119
41. Demand data predicted by each model for November 2014 at 2am.....	121
42. Prediction error percentages for November 2014 at 2am	121
43. Standardized error for each model at 2am on November 2014	122
44. Demand data predicted by each model for November 2014 at 9am.....	124
45. Prediction error percentages for November 2014 at 9am	124

46. Standardized error for each model at 9am on November 2014	125
47. Demand data predicted by each model for November 2014 at 2pm.....	127
48. Prediction error percentages for November 2014 at 2pm.....	127
49. Standardized error for each model at 2pm on November 2014	128
50. Demand data predicted by each model for November 2014 at 8pm.....	130
51. Prediction error percentages for November 2014 at 8pm.....	130
52. Standardized error for each model at 8pm on November 2014	131
53. Demand data predicted by each model for December 2014 at 2am	133
54. Prediction error percentages for December 2014 at 2am.....	133
55. Standardized error for each model at 2am on December 2014.....	134
56. Demand data predicted by each model for December 2014 at 9am	136
57. Prediction error percentages for December 2014 at 9am.....	136
58. Standardized error for each model at 9am on December 2014.....	137
59. Demand data predicted by each model for December 2014 at 2pm	139
60. Prediction error percentages for December 2014 at 2pm	139
61. Standardized error for each model at 2pm on December 2014	140
62. Demand data predicted by each model for December 2014 at 8pm	142
63. Prediction error percentages for December 2014 at 8pm	142
64. Standardized error for each model at 8pm on December 2014	143
65. SARIMA models for forecasting residuals.....	166
66. Hourly data (Actual vs Predicted) for LOWRIMA	168
67. Hourly data (Actual vs Predicted) for RFSRIMA	168
68. Statistical indices measuring Accuracy of LOWRIMA and RFSRIMA	169

69. Performance analysis of clustering algorithms 177

ACKNOWLEDGEMENTS

First of all, I want to thank my parents, Gautam and Rina Mukherjee and my sister Monomayee for supporting me throughout my life for my happiness. I am forever indebted to them.

I would like to express my sincere gratitude to my advisor, Dr. Prakash Ranganathan for his continuous guidance, support and advice which he had provided from the day one. I'm very fortunate to have him as my advisor.

I am thankful to the members of my committee, Naima Kaabouch and Saleh Faruque for their acceptance of being in my committee as well as their guidance. I would also like to acknowledge the role of the Department of Electrical Engineering at University of North Dakota (UND), Grand Forks, North Dakota for providing me the opportunity to study and conduct research.

My thanks also goes to my friends and colleagues here in Grand Forks for supporting me both academically and socially.

Above all, I am grateful to the Almighty God for giving me the capability and the chance to finish my thesis.

ABBREVIATIONS

PMU	Phasor Measurement Unit
SCADA	Supervisory Control and Data Acquisition
AMI	Advanced Metering Infrastructure
GPS	Global Positioning System
SG	Smart Grid
ISS	Integrated Software Suite
DBSCAN	Density Based Spatial Clustering of Applications with Noise
DC	District of Columbia
RF	Random Forest
LOWESS	Locally Weighted Scatterplot Smoothing
SARIMA	Seasonal Auto Regressive Integrated Moving Average
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
PCE	Percentage Error
RTU	Remote Terminal Unit
SE	State Estimation
WAMS	Wide Area Monitoring System
PDC	Phasor Data Concentrator

DFT	Discrete Fourier Transform
ADC	Analog to Digital Conversion
UTC	Coordinated Universal Time
TB	Terra Bytes
SDM	Synchrophasor Data Mining
ISO	Independent System Operator
DB	Database
GPA	Grid Protection Alliance
SGMF	Smart Grid Management Framework
ARIMA	Auto Regressive Integrated Moving Average
EM	Expectation Maximization
SVM	Support Vector Machine
LF	Load Forecasting
DT	Decision Tree
RTDMS	Real Time Dynamic Monitoring System
CART	Classification and Regression Trees
DER	Distributed Energy Resources
MA	Moving Average
WMA	Weighted Moving Average
ES	Exponential Smoothing
ARMA	Auto Regressive Moving Average
AI	Artificial Intelligence
FL	Fuzzy Logic

MLR	Multiple Linear Regression
DES	Double Exponential Smoothing
T&D	Transmission and Distribution
DSM	Demand Side Management
VSTLF	Very Short Term Load Forecasting
STLF	Short Term Load Forecasting
MTLF	Mid Term Load Forecasting
LTLF	Long Term Load Forecasting
RF	Random Forest
LSR	Least Square Regression
DT	Decision Tree
CTR	Click Through Rate
GDP	Gross Domestic Product
AR	Auto Regressive
RTO	Regional Transmission Organization
NOAA	National Oceanic and Atmospheric Administration
F	Fahrenheit
DI	Dunn's Index
ACF	Autocorrelation Function
pACF	Partial Autocorrelation Function
SR	Standardized residuals
R	Residuals
SD	Standard Deviation

SLG Single Line to Ground

MC Mis-classification

ABSTRACT

With the large scale deployment of phasor measurement units (PMU) in the United States, a resonating topic has been the question of how to extract useful “information” or “knowledge” from voluminous data generated by supervisory control and data acquisition (SCADA), PMUs and advanced metering infrastructure (AMI).

With a sampling rate of 30 to as high as 120 samples per second, the PMU provide a fine-grained monitoring of the power grid with time synchronized measurements of voltage, current, frequency and phase angle. Running the sensors continuously can produce nearly 2,592,000 samples of data every day. This large data need to be treated efficiently to extract information for better decision making in a smart grid network (SG) environment.

My research presents a flexible software framework to process the streaming data sets for smart-grid applications. The proposed Integrated Software Suite (ISS) is capable of mining the data using various clustering algorithms for better decision-making purposes. The decisions based on the proposed methods can help electric grid’s system operators to reduce blackouts, instabilities and oscillations in the smart-grid. The research work primarily focus on integrating a density-based clustering (DBSCAN) and variations of k-means clustering methods to capture specific types of anomalies or faults. A novel method namely, multi-tier k-means was developed to cluster the PMU data. Such a grouping scheme will enable system operators for better decision making. Different fault conditions, such as voltage, current, phase angle or frequency deviations, generation, and load trips, are investigated and a comparative analysis of application of three methods are studied.

A collection of forecasting techniques has also been applied to PMU datasets. The datasets considered are from the PJM Corporation that describes the energy demand for 13 states and District of Columbia (DC). The applications and suitability of forecasting techniques to PMU data using random forest (RF), locally weighted scatterplot smoothing (LOWESS) and seasonal auto regressive integrated moving average (SARIMA) has been investigated. The approaches are tested against standardized error indices like mean absolute percentage error (MAPE), mean squared error (MSE), root mean squared error (RMSE) and normal percentage error (PCE), to compare the performance. It is observed that the proposed hybrid combination of RF and SARIMA can be used with good results in day ahead forecasting of load dispatch.

I. INTRODUCTION

The term, “Smart Grid” has been around for decades and is used in connection with ways to automate the functioning of the conventional grid system. Various definitions of a smart grid including the context of technical, functional and economic benefits exist. However, the common definition that runs through all of them is the addition of a digital processing, intelligent and communications layer to the grid system. The SG concept and its benefits were originally published in [1]. The idea was to make the conventional power grid more efficient and self-aware of its environment. Although an engineering marvel, the power grid was not able to keep up with the growing demand of users [2]. Coupled with reduced power quality during peak hours, blackouts and brownouts, the necessity of a more efficient power grid was felt [2].

A power grid that would be self-aware of grid-environment and is able to take decisions and self-heal is now a necessary requirement. This obviated the introduction of automation in the existing system. The application of a digital layer with a communication infrastructure made information management and data flow central to the smart grid. A two way communication could be established between the users and a distributed set of power generation units. Figure 1 gives an overall view of the structure of a Smart Grid and Figure 2 shows the all-important digital layer controlling a smart grid. The evolution of metering devices has played a key role in modelling the smart grid. In the 1980’s, automatic meters were introduced to monitor loads from large customers. This later resulted in the

development of the Advanced Metering Infrastructure (AMI) in the 1990's that could also store the usage of electricity at different times of the day [3]. Smart Meters (SM), aided real-time monitoring of the grid and functioned as a gateway to demand response-aware devices and "smart sockets". Italy's Telegestore Project came out as a front runner in the smart grid technology when it connected a large number of users (approx. 27 million) through these smart meters, using low bandwidth power line communication (PLC) [4].

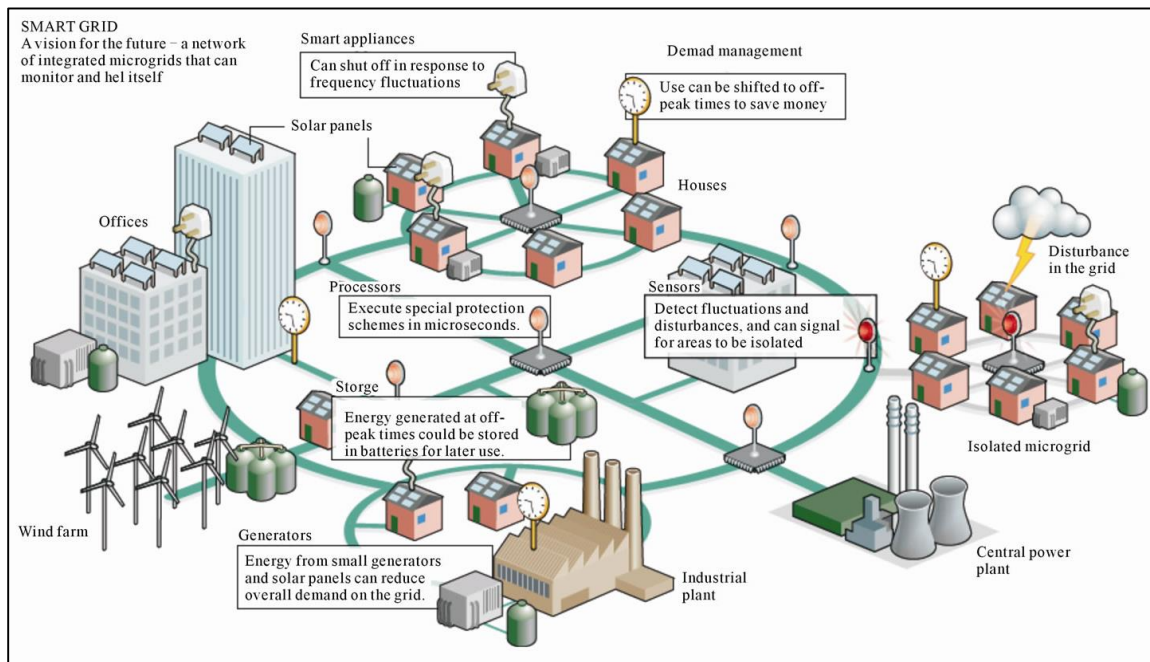


Figure 1: A fully connected Smart Grid Network [5]

A fully realized Smart Grid should perform the following functions:

1. Be able to heal itself, which basically means that it should be in control in case of any contingency scenario.
2. Be able to allow producers as well as consumers to participate actively in grid operations
3. Be able to resist any kind of attacks.
4. Be able to maintain power quality, thus eliminating financial losses due to power cuts.

5. Be able to accommodate different generation sources (conventional and renewable) of electricity.
6. Be able to improve the efficiency and reliability of the current power grid.

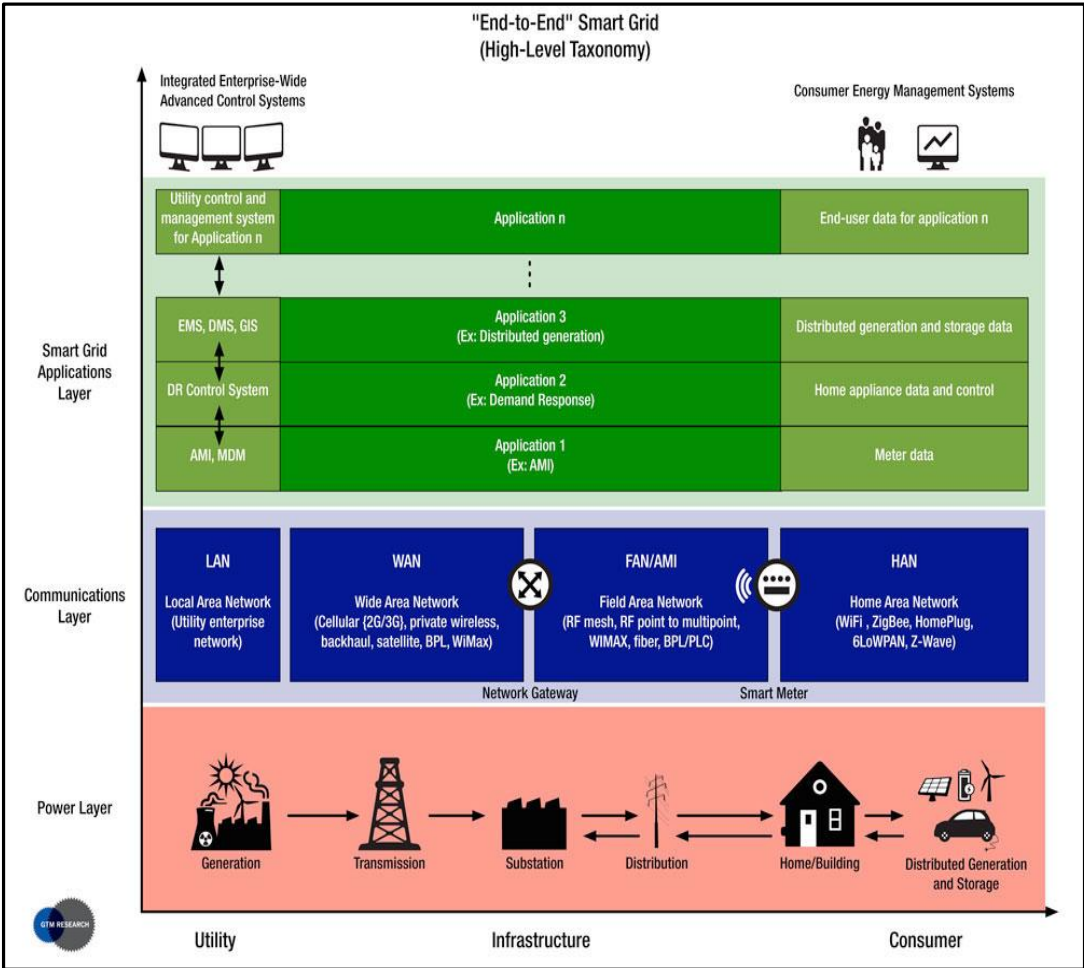


Figure 2: Digital Layer of a Smart Grid [8]

1.1 Synchrophasors/Phasor Measurement Units (PMU's)

The concept of a Phasor Measurement Unit (PMU), also known as a synchrophasor was developed as early as the late 80's, by Dr. Arun G. Phadke and Dr. James S. Thorp. Synchrophasors are the modern-day measuring devices that measure the magnitudes of voltage and current, frequency and phase angle of the electrical buses and lines. Parameters that can be used to monitor the state of a power grid [6]. Figure 3 shows the functional

block diagram of a PMU. Synchrophasors are replacing the previously installed Remote Terminal Units (RTU) to overcome the short-comings of the Supervisory Control and Data Acquisition (SCADA) system. Under this system, state estimators (SE) approximate the condition of a grid from the data collected at each RTU. The data collected has information about the basic parameters of the system topology. This data is collected asynchronously at periodic scans of the network, every 2 to 10 seconds [7]. While the estimated results would not vary from that of a synchrophasor during steady state conditions, SCADA would fail to record any contingency scenario in transient state and for a duration less than the scan period. The asynchronous nature of collected data disallowed the possibility of comparing deviations in phasor values between two nodes at a given time.

Also, the analog measurements are proportional to the time difference between the measurements [9]-[10].

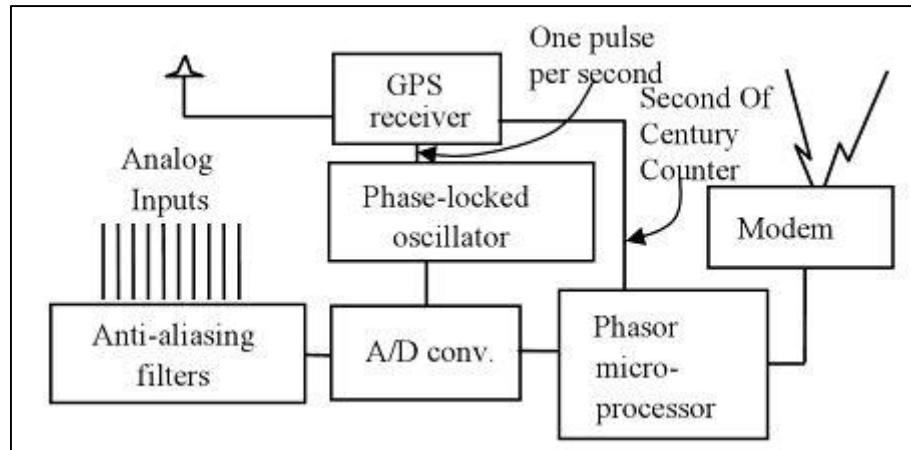


Figure 3: Functional Block Diagram of a PMU [11]

PMUs are essentially metering devices which record phasor data at a very high sampling rate (the latest PMUs have 240 samples/second rate of data acquisition). They are also very precise time keepers, being synchronized with a GPS time clock. Unlike their mechanical predecessors, PMUs are equipped with a communication system that allows a two way communication with the power generation units as well as other PMUs in the grid. These

units can not only record live data with high granularity, but also have the capability to detect and respond to the network load, changing tariff rates. In a real network, there are several PMUs to form a distributed network of sensors that allow the utilities to monitor the grid status. Such a distributed network is called a Wide Area Monitoring System (WAMS) as shown in Figure 4. The three major components that make up a WAMS are as follows:

1. Phasor Measurement Unit (PMU)
2. Phasor Data Concentrator (PDC)
3. Global Positioning System (GPS) clock

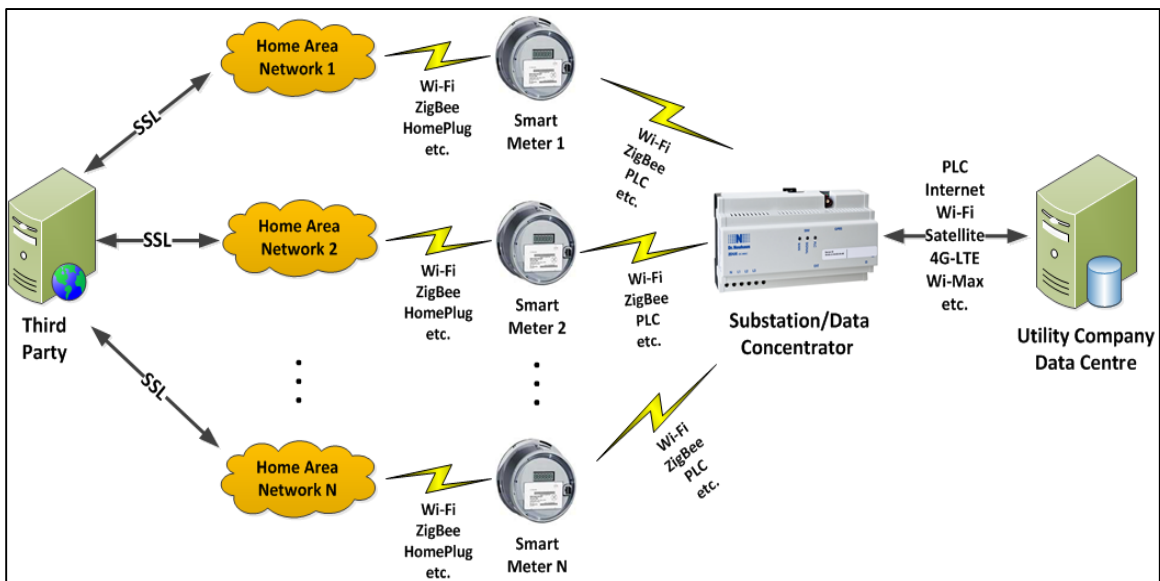


Figure 4: A Wide Area Monitoring System (WAMS) [12]

Phasors are the repetitions of the waveform of alternating current which should ideally conform to the same shape all over the network. PMUs apply a very common method of generating phasor signals that represent the line voltage/current [12]. The actual waveforms are obtained as samples through a series of network components like filters and transformers [12]. The samples are then subjected to Discrete Fourier Transform (DFT) for phasor computation [13] and then digitized through Analog to Digital Conversion (ADC).

The GPS clock accurately time stamps the resulting signal to the coordinated universal time (UTC) to assist the WAMS operation. These time synchronized phasor data are aggregated into centralized data servers called Phasor Data Concentrators (PDC) as per IEEE C37.244 standard [14]-[15].

The time stamped data gathered from all the PMUs present a consolidated view of the entire grid system and thus allows the data to be used a variety of applications. The applications include state estimation, transient analysis, load shedding schemes, inter-area oscillations monitoring and micro- grid operations [16]-[17] like islanding and anti-islanding and detecting fault locations on transmission lines [18]-[20].

1.2 Utilization of Synchrophasor Data

Modern day power transmission and delivery is affected by changing load characteristics, limited transmission paths and distributed generation. The increased stress on the power system can only be relieved by improved operator and automatic response times. For this reason, accurate data collected synchronously at a high time resolution is as important as the proper utilization of the highly granular data. GPS time stamping and high sampling rate can provide such accuracy and precision. To extract information out of the raw data, specific data scan techniques should be applied to filter the data. The concept of organizing and cleaning large amounts of raw data is called Big Data Analysis.

The importance of PMUs its basis can be reinforced by discussing the case of the North American Eastern blackout in the year 2003 [21]. Figure 5 show the differences in phase angle between two buses. It indicates how, PMUs identify that North American Eastern

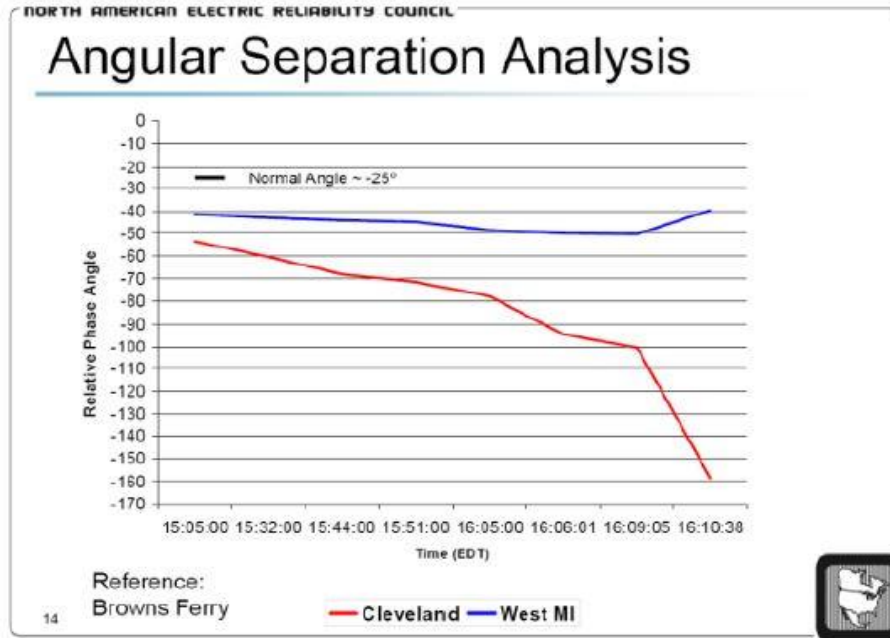


Figure 5: System separation, August 2003 [21]

Interconnection System was heading for a blackout. The phase angle values change at a high rate and in order to perform a differencing operation between two angle values at a given time, synchronized measurements are critical. This can only be done by synchrophasors. Figure 6 represents the difference in phase angles of the two buses on a linear scale. A closer inspection of Figure 6 reveals that the phase angles have been plotted on a non-linear scale. A comparison of the two figures reveal that the linear representation in Figure 6 fails to detect the change in phase angles until the last three seconds, whereas, the deviations can be clearly noted in Figure 5 from much earlier. This proves that, both time synchronized phasor data collection, and proper representation of the gathered data is equally important.

1.2.1 Challenges with PMU data

Synchrophasors capture phasor data at around 30 to 120 samples per second, with the modern day PMUs can even take 240 sample measurements per second. Thus, the system operators at the utilities are bombarded with nearly 108,000 samples/hour, which amounts

to about 2,592,000 samples/a day or 77,660,000 samples a month. These numbers grow exponentially with an addition of new PMUs and as much as 1.5 Terra Bytes (TB) of data can be accumulated in a month's span. This huge volume of data needs to be handled tactfully so that critical information can be extracted from it and appropriate alarm systems can be generated to inform the system operators.

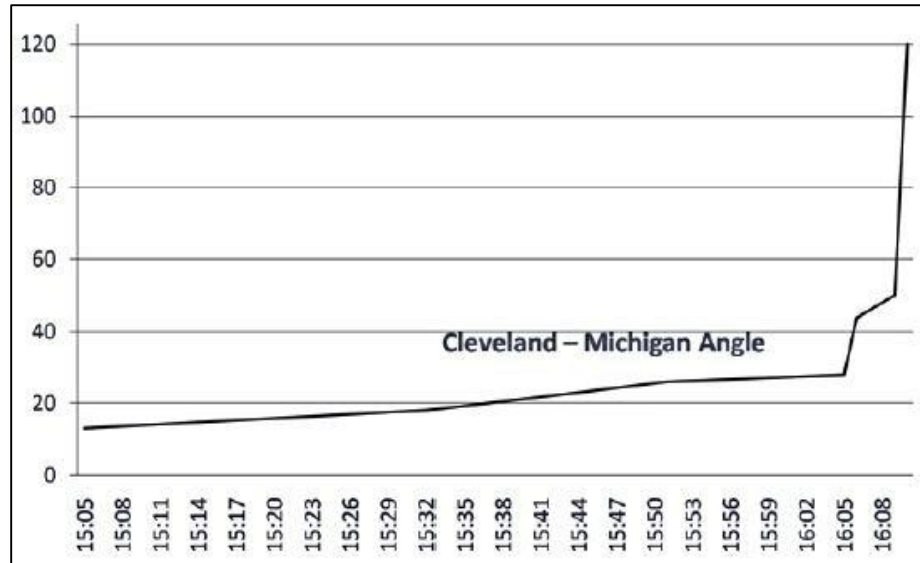


Figure 6: Redrawn phase angle with linear scale [21]

To perform any kind of analysis in protecting the grid, synchrophasor data needs to be utilized. This can be done only through Synchrophasor Data Mining (SDM). SDM is the process of analyzing synchrophasor data through clustering algorithms. As a result meaningful information can be derived from the tons of data provided by synchrophasors for use of wide area monitoring services such as fault detection, and line protection etc. This allows effective monitoring of the state of the grid, thus enabling crucial decisions to be made. A data analysis and visualization technique has been proposed in this work that will allow an independent system operator (ISO) to conveniently study the incoming phasor data and visualize it on a near real-time basis. A novel multi-tier k-means approach has been implemented and was compared against the performance of prevalent methods,

namely Density-based spatial clustering of applications with noise (DBSCAN) and k-means, under different test scenarios. The streaming data is also stored in a database (DB), which creates a repository for historical data. This history was also used to develop a load forecasting scheme that allows the ISO to do a day ahead prediction of the load on a given region or zone.

1.2.2 Application of open source phasor data concentrator (openPDC)

In a WAMS network, the PMUs connected to an optimally selected transmission and distribution lines [22] send streaming data that gets collected in a Phasor Data Concentrator (PDC), ready to be analyzed by ISO. Various non-storage, vendor based software are used to process the PDC data to derive valuable information. The openPDC is one such PDC software that can be used to process streaming time-series data in real-time [23]. The software is managed by Grid Protection Alliance (GPA) that gather time-stamped data from hundreds of utilities. The data is time-sorted and provided for user defined actions as well as to custom outputs for archival. The openPDC is not only a phasor data concentrator, it is a flexible platform for processing high speed, streaming data that can adapt with changing technology to provide a future-proof phasor data architecture [24]. The phasor data concentrator has been in production use since 2004. It is based on the SuperPDC, which was developed by the Tennessee Valley Authority [24]. It is a Microsoft.NET application that runs as a Windows service and is responsible for managing the life-cycle of adapters that create and process the streaming phasor measurements. The openPDC software framework functions by receiving data broadcasted by a PMU. It provides features like data storage, data archival, rebroadcasting, and analysis of the phasor data. openPDC suite provides different options to view the historian data by selecting a sampling

rate to display various parameters such as phasor, frequency etc. Generally, openPDC archives the time-synchronized data as it receives it. A comprehensive input protocol supported by openPDC includes IEEE 37.118, IEEE1344, SEL FastMessage, Macrodyne and Virginia Tech F-Net protocols [24]. Synchrophasor data can be mined by extracting the archived data that is stored in openPDC. For this task, a C# code using Microsoft Visual Studio has been formulated [25]. A SQL database have been set up for the openPDC environment that produces PMU data using a PDC connection tester file. The data is produced as per the Grid Protection Alliance directives. The generated data gets stored in an Archives folder in the computer as “.d” files, namely ‘ppa_archive.d’. There are separate virtual ports to access frequency, magnitude, and phase angle data. The streaming data gets stored in the “.d” file with the port number as an identifier. To record the data for a post-event analysis, or a near real-time analysis, the port numbers corresponding to the fields can be utilized. A data extraction code is required to extract the data from the “.d” files and transferring them to an “.xlsx” or a “.csv” file. This extraction is done using the Microsoft Visual Studio for C# programming. The research work exploits the parallel computing feature of C# by using parallel for-loops to extract the data from different ports simultaneously, with minimal time offset. Once the data is extracted, it gets written in a “.csv” file. This is done by using the ‘Microsoft.Office.Interop.Excel’ library function built in the Visual Studio IDE. The program for extracting the data is developed as a console application. The whole process takes an average of about 47 seconds for a five minute dataset [26]. Figure 7 represent the entire process of data extraction from openPDC environment and application of various data mining algorithms on it.

1.3 Thesis Contributions

The following are the two objectives of this research work.

Objective 1 To provide a software solution framework which will reduce the chances of a blackout conditions. This is achieved by mining the smart grid data for effective decision making.

To accomplish this objective, following four tasks were carried out:

Task 1 Conducted literature review on topics such as representing and grouping of Smart Grid data.

Task 2 Application of data mining algorithms, namely ‘Density-Based Spatial Clustering of Applications with Noise’ (DBSCAN) and k-means clustering for time synchronized synchrophasor measurements. The data source used in this was from the openPDC environment.

Task 3 Developed a novel hybrid clustering method, namely ‘multi-tier k-means’ suited to classify synchrophasor data into good, bad, low-noise and high-noise clusters.

Task 4 Evaluated the separation of clusters using ‘Dunn Index’ in DBSCAN, k-means and multi-tier k-means clustering methods.

Objective 2 Develop forecasting models for time-synchronized demand data from PJM Corporation.

To accomplish this objective, following two tasks were carried out:

Task 5 Conduct literature review of various forecasting algorithms suitable for time-series datasets.

Task 6 Test the applicability of techniques such as multiple linear regression, random forest, locally weighted scatterplot smoothing (LOWESS), curve fitting and time series analysis seasonal auto regressive integrated moving average (SARIMA) to PJM datasets. A statistical comparison of the above method are done using indices such as, mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean square error (RMSE).

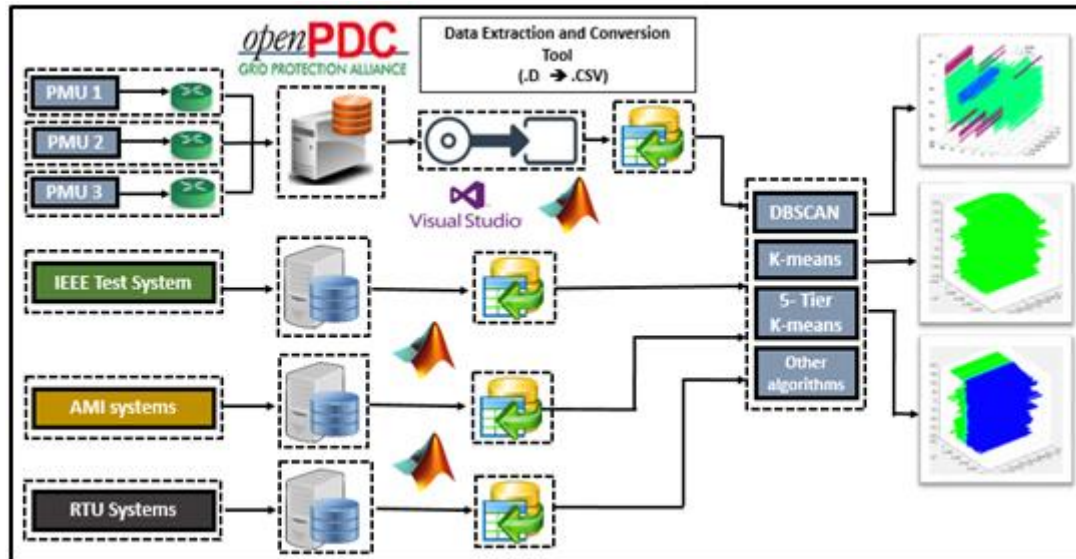


Figure 7: Smart Grid Management Framework (SGDMF)

1.4 Thesis Organization

The thesis is organized as follows. **Chapter 2** is divide into two sections. The first section represent the literature review on clustering algorithms for PMU datasets. The second section presents the literature review of existing forecasting methods. **Chapter 3** discusses application of existing clustering methods and the development of a novel method of clustering phasor data. It is a hybrid k-means method. It was developed using Matlab/Simulink with an aim to identify clusters based on threshold values and thus allowing the development of an effective alarm generation. **Chapter 4** presents the application of existing forecasting models on phasor data sets. It also discuss the development of a hybrid model that conducts short term and mid-term load forecasting.

Chapter 5 is divided into two sections. The first section present the results obtained from application of the novel clustering algorithm and compare it to other existing methods. The second section presents the accuracy of results obtained from the application of the hybrid forecasting model and compares with the other existing methods. Finally, **Chapter 6** summarize the conclusion and the directions of future work.

II. LITERATURE REVIEW

2.1 Data mining and visualization

The term data mining was first coined in the 1990's. However, the idea existed already in the form of classical statistics, artificial intelligence and machine learning [27]. There are generally two main reasons for using the concept of data mining: 1) there is a lot of data to comb through which have too little information and 2) there is a need to extract useful information from the huge amount of raw data which will serve as the basis for some decision making system. Data mining is an interdisciplinary field that merges areas like database systems, statistics, machine learning and pattern recognition [28]. It forms the pre-processing block of any control system providing services like data extraction, data cleaning, data fusion, data reduction, and feature construction. As a post-processing block, it can provide pattern and model interpretation, hypothesis confirmation and generation. Data mining is typically a highly iterative and interactive process as new data generated. It is this feature of data mining that form the backbone of any decision making and forecasting algorithms.

2.1.1 Data Clustering and Classification

Data clustering and classification are fundamental data mining processes applied on n data points in a d dimensional space. While classification is a supervised method of grouping data, clustering is largely unsupervised [29]. Supervised method means that there is already a predefined set of classes present and the incoming data is categorized based on the class it belongs to. Unsupervised learning, on the other hand groups data into separate clusters

and tries to find out the relationship between the groups. Since the research concentrates on clustering of PMU data into pre-defined ranges. The research topic aligns largely with supervised learning methods.

2.1.2 Clustering of data

Clustering of data and technique depend on the type of data and the desired cluster characteristics. The methods can be representative-based, hierarchical, density-based, graph-based and spectral clustering [28]. Representative based clustering includes k-means clustering, Expectation-Maximization (EM) and its various forms. Hierarchical clustering is an agglomerative process where each data point starts from its own cluster and successively merge pairs of clusters until the desired number of clusters has been found [28]. Density based clustering works on the density or the connectedness of clusters. In some cases, the distance between two data points in the same cluster may be more than that between two points in different clusters [28]. The density based clustering schemes (DBSCAN) are useful when applied on nonconvex data clusters. Graph based clustering methods focus on spectral density of graph data. They can be considered as an optimization problem over a k -way cut in a graph. Multiple objective functions can be formulated for spectral decomposition of graphs. They employ different graph matrices such as adjacency matrix and Laplacian matrix [28].

2.1.3 Classification of data

The task of classification encompasses predicting the label of a data object based on the number of classes present. There are different types of classification algorithms that are available, 1) decision trees, 2) probabilistic classifiers, 3) support vector machines (SVM). The powerful Bayes classification is a probabilistic approach to classify data. It uses Bayes

theorem to classify data objects that would maximize the posterior probability [28]. A decision tree classifier is known for its simplistic models [28]. It recursively partitions the data set into groups that are different from each other with very few exceptions. SVM is by far the most effective classification methods for many different problem domains [28]. The goal of SVMs is to find the optimal hyperplane that maximizes the margin between different classes. With the use of a kernel, the SVMs can be used to find non-linear boundaries, which correspond to some linear hyperplane in some high-dimensional, non-linear space [28].

2.1.4 Visualization of data

Data visualization is the use of any combination of text, audio, video, charts, images, timelines, trends, etc. to represent information. While data mining can provide effective methods to extract information from data sets, visualization methods can help represent information for clearer human perception [30]. Information extracted from a data set can be used by computers and processing units to make decisions, but it is not always effective for human understanding [30]. We restrict the term “Visualization” in this work to system operators perception in the grid.

2.1.5 Visualization and data analysis

Data visualization uses the brain-to-eye connection and brings with it an array of promises, some of which are discussed as follows:

1. It is able to affect the visual cortex directly, bypassing the language centers.
2. A good visualization scheme is able to leverage the ability of pattern recognition and visual sense making of human beings.

3. Powerful graphics chips enable processing of live data and animation of stored data possible, thus giving us a better sense of perception what the data suggest.

The current trend of data visualization in practice is not only to facilitate the monitoring of data points in a multidimensional space, but also to empower users to interact with data objects. The ability to interact with data gives an added advantage towards effective analysis of data. One of the most powerful techniques of visual analysis involve the simultaneous display of multiple graphs, which feature either different subsets of data taken from a larger data set, or different views of a shared data set [30]. For example, several instances of static graphs could be bar graphs, line graphs, scatterplots, etc. These graphs could be displayed simultaneously. Each of them represent the same data, but give a different perspective to it. To establish a good trend in visual analytics, data visualization needs to play the following roles:

1. Making sense of new information

For information which was previously unknown to man, visualization should be able to provide new insights about the data. It should provide a way of simply “throwing things on a wall” in order to examine them. This would allow users to perceive possibilities which were previously unknown and thus unfathomable. However, the data visualization should still maintain the integrity of data, i.e. it should be more about the data than visualization [30].

2. Provide an interactive environment for data exploration

In order make data analysis more interactive, visual analytics needs to provide tools which allows users a holistic exploration of data. Provisions should be made to

represent the data in multiple dimensions like time or space. In addition to that, there should be filters provided that allows in depth analysis of a data object. Finally, it should also provide for marking and sharing discoveries within the tool or over the internet [30].

2.1.6 Visualizing power grid data

In order to modernize the power grid to increase its efficiency and keep up with the growing demands led to the realization of a robust smart grid. This led to the innovation of the PMUs. These are time synchronized metering devices that relay information to utilities with a data rate of 30 samples per second. The large amount of data generated by PMUs need to be analyzed for effective control of the grid. There has been little or no work done on the data analytics platform of handling PMU data. References [31]-[51] describe various approaches to analyze and represent phasor data to solve for various power system challenges. The time-tagged data from synchrophasors can be used for applications such as state estimation (SE) [31]-[32], load forecasting (LF), fault detection and micro-grid islanding operations [33]-[36]. Using synchrophasor data, a voltage stability assessment technique has been proposed in [37]. An algorithm has been developed to detect and locate the faults on the transmission lines using the phasor data in [38]. A RTU/SCADA system provide data at a scan rate of 30 samples for a 5 minute period, while the same number of data samples are provided by synchrophasors within a second. This makes a major difference to operators to capture faults or abnormal conditions in the grid using synchrophasors. Although, synchrophasors provide power system information at a large sampling rate, they can be useful only if the operators know how to utilize the data. Recently, a method for visualizing and interpreting the synchrophasor data was developed

using Hilbert analysis. It has been discussed in [39]. This problem of visualizing real time data has been addressed by using circle representation [39]. Although the utilities have been using services of PMUs for pre- and post-event monitoring of the grid, the true potential of the data provided by the PMUs hasn't been fully realized. Due to expensive cost of procuring real hardware, a virtual synchrophasor monitoring network (PMU and PDC) using LabVIEW software has been developed in [40], and used as a teaching tool in power system courses. A synchrophasor network was developed and used to monitor the PMU data for both on-line and off-line data analysis in [41]. The authors in [33] use multiple data classification algorithms (k-means and Naïve Bayes) for classification and fault detection from synchrophasor data. Synchrophasor data quality has been addressed in [42] for data conditioning. Decision Tree (DT) methods are used to detect the loss of synchronism within PMU data [43]. Using software platforms such as Real Time Dynamic Monitoring System (RTDMS), a methodology has been proposed for assessing the static and dynamic stresses in large, interconnected power grids [44].

A classification and regression trees (CART) method was used to classify the dynamic events in the power systems using synchrophasor measurements in [45]. A more realistic model of a power system is proposed in [46] to monitor security assessment using decision trees and [47] describes a similar method to monitor voltage levels in particular. In Ref. [48], different methods such as Fourier transform, Yule-Walker- and matrix-pencil were used to analyze the power system events. Recently, a parallel de-trended fluctuation analysis method has been proposed to detect the transient events utilizing a computer cluster in [49]. In [50], researchers proposed a SPASE state estimator. It uses the synchrophasor data to perform state estimation and also for visualization application. A

prediction and frequency quality detection theorem based on Bayesian networks is developed in [51] to improve the reliability of the grid. The time synchronized phasor measurements provided by PMUs can be used for applications such as state estimation, transient analysis, capacitor bank's performance, analysis of load shedding schemes and inter-area oscillations in [52]-[54]. PMU data can also be used for generator black start, islanding and anti-islanding conditions with Distributed Energy Resources (DER), to measure transmission relay parameters. These analysis methods have been discussed in [55]-[57]. The PDC software that has been used in this research to simulate and gather phasor data for analysis is an open source one and is called openPDC [23].

2.2 Forecasting techniques

Time series based forecasting methods are applied to data points collected over successive intervals of time. Examples of such data include weather, energy demand over time and ocean tides. Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data. Time series forecasting is the use of a model to predict future values based on previously observed values. Time series are used in statistics, signal processing, pattern recognition, econometrics, mathematical finance, weather forecasting, load forecasting, and so on. Some of the types of time-series forecasting are 1) Moving average (MA), 2) Weighted moving average (WMA), 3) Exponential smoothing (ES), 4) Autoregressive moving average (ARMA), 5) Autoregressive integrated moving average (ARIMA), 6) Seasonal Autoregressive integrated moving average (SARIMA), 7) Extrapolation, 8) Linear prediction and 9) Trend estimation [58].

Explanatory models and time series models together form what is known as Quantitative

forecasting methods. While a time series model is concerned with a single dimensional data set which has been collected sequentially in time, explanatory models take into account all factors that might have caused the observation to occur. For example, the cause for energy demand can depend on environmental factors like outside temperature, time of the day, month of the year, season, humidity, etc. All these factors play a key role to induce the amount of electricity being used. So, in order to get a better idea of what the usage is going to be in the future, having such information of the future could help in better estimating the future energy demand.

In this research, both time series and explanatory models have been investigated and a hybrid model has been tested. The results obtained from the model has been presented in chapter 6.

2.2.1 Forecasting data model types

Statistical models used for forecasting can be broadly categorized in to parametric and non-parametric models. These models basically describe the test parameters used in accumulation of observational data collected from test scenarios. A brief description of both the models are presented below:

1. Parametric Model

In statistics, a parametric model or parametric family or finite-dimensional model is a family of distributions that can be described using a finite number of parameters [59]. These parameters are usually collected together to form a single k-dimensional parameter vector $\theta = (\theta_1, \theta_2, \dots, \theta_k)$. Hence, in a collection of probability distributions making up an observation model, each member can be denoted as:

$$\rho = \{P_\theta \mid \theta \in \Theta\} \quad (1)$$

Where, P_θ is a member probability distribution function and $\Theta \subseteq \mathbb{R}^k$ and R is the number of observations for each distribution.

For example, the Poisson family of distributions is parametrized by a single number $\lambda > 0$:

$$P = \left\{ \rho_{\lambda(j)} = \frac{\lambda^j}{j!} e^{-\lambda}, j = 0, 1, 2, \dots \mid \lambda > 0 \right\} \quad (2)$$

Where ρ_λ the probability mass function and the family is an exponential.

2. Non-parametric model

Non-parametric statistical models are not based on parameterized families of probability distributions [60]. They include both descriptive and inferential statistics [60]. The typical parameters are the mean, variance, etc. Unlike parametric statistics, non-parametric statistics make no assumptions about the probability distributions of the variables being assessed. The difference between parametric model and non-parametric model is that the former has a fixed number of parameters, while the latter grows the number of parameters with the amount of training data. Examples of such models in use are as follows:

- a) Anderson-Darling test
- b) Statistical bootstrap methods
- c) Friedman two-way analysis of variance

2.2.2 Steps involved in forecasting tasks

The number of steps involved in performing a forecasting task can be broadly divided into five stages [61]. These have been briefly described as follows:

- a) Problem definition: The definition of the problem is sometimes the most difficult task for a forecaster. It involves understanding how the forecasts will be used

and who requires the forecasts.

- b) Gathering information: It is important to collect historical data of the fields of interest. This data can be used to construct a model for forecasting. There are always at least two kinds of information available, statistical data based accumulated judgement and expertise of key personnel. Both of these information needs to be referred to make a good forecasting model.
- c) Preliminary (exploratory) analysis: This stage involves extracting useful information from the data. The data can be graphed for visual inspection, followed by calculating simple descriptive statistics like mean, standard deviation, etc. Scatter plots can then be used to represent datasets that contains collecting historical data and the forecasted data.
- d) Choosing and fitting models: Based on the understanding of the historical data, appropriate forecasting models can fitted. This done to figure out which model would be more appropriate for use. For example, an ARIMA model would be well suited for a time series data. However, if the data is impacted by several other factors, then certain regression analysis methods can also be used for forecasting.
- e) Evaluating forecasting models: Once a model has been selected judiciously and its parameters are estimated appropriately, the model is now ready to make forecast. As time progresses, the pros and cons of using the model can then be evaluated by the user. The performance of the model can only be evaluated properly after the forecasted data becomes available.

In most forecasting scenarios, the accuracy of the forecast decide the selection of the model. In many cases, the accuracy of the model basically means the “goodness

of fit” of the model, which basically means how well the model is being able to reproduce the data that are already known [61]. There are number of standardized statistical indices that are used for this purpose. They have been listed below:

a) Mean absolute error (MAE):

$$MAE = \frac{\sum_{t=1}^N |E_t|}{N} \quad (3)$$

b) Mean absolute percentage error (MAPE):

$$MAPE = \frac{\sum_{t=1}^N \left| \frac{E_t}{Y_t} * 100 \right|}{N} \quad (4)$$

c) Mean squared error (MSE):

$$MSE = \frac{\sum_{t=1}^N |E_t^2|}{N} \quad (5)$$

d) Root mean squared error (RMSE):

$$RMSE = \sqrt{\frac{\sum_{t=1}^N |E_t^2|}{N}} \quad (6)$$

In all of the above cases, error E_t , actual data Y_t and forecasted value F_t are related as:

$$E_t = Y_t - F_t \quad (7)$$

There has been a large number of papers published in the field of load forecasting over the past few decades. The literature review in this section concentrates on papers published in reputed journals that discuss forecasting models and its criteria for selection. The authors of [62] conceptualize STLF as a regression model, while a time series approach has been

employed to model the same in [63]. On the other hand, the authors of [64]-[66] use the application of Artificial Intelligence (AI) methods and their combinations to perform STLF. While [64]-[65] discuss about the application of ANN, [60] uses Fuzzy Logic (FL) and [67] applies SVM. The success of explanatory forecasting methods lie in the correct choice of its independent variables that govern the use of electricity by people. A lot of work has been done in choosing the most suitable variables for that purpose.

The researchers discuss the choice of temperature and relative humidity as the controlling factors in [68], while the effect of humidity and wind speed was considered through linear transformation of temperature in the improved version [69]. In general, the electric load is mainly driven by nature and human activities. The effects of nature are normally represented by weather variables like temperature, while the effects of human activities are normally reflected by the calendar variables like business hours in a day. The combined effects of both elements exist as well but are nontrivial. The authors of [70] describe a semi-parametric approach of forecasting loads, where a historical database of demand at corresponding temperatures have been used to predict the future power consumption. The errors resulted from prediction is then used as a time series dataset for a Seasonal ARIMA (SARIMA) input. The authors of [71] study the effect of application of various models on historical data collected from various sites. They discuss the criteria for choosing a particular model based on the characteristics of available data. The other popular forecasting models for time-series datasets are Multiple Linear Regression (MLR), SARMA and SARIMA, Random Forest (RF) and Double Exponential Smoothing (DES). It was observed that model selection is heavily influenced by the variability in the data. Models which do not use weather forecast information, but rely only on historical usage

data perform better on sites with highly variable loads. Finally, detailed discussion on load forecasting is very well summed up by the author of [72] which elaborates on the history of load forecasting and how it has evolved over several decades from simply counting bulbs to today's complex and computationally extensive models.

In the research, at first RF method was used to create a forecasting model with the decision variables being average daily temperature, day of the week, hour of the day, season and type of day. RF was used to take advantage of all factors available that affect the load usage. The error between actual and forecasted data was then calculated. This error value was identified as a result of some unnatural spikes in load data. Since such spike's causes cannot be accounted for in a general forecasting model, a data dependant forecasting technique, SARIMA was used. Here, the error values were used as a time series dataset. The predicted error was then adjusted with the RF forecasted value to get a final load data prediction.

III. CLUSTERING METHODS AND openPDC INTERFACE

PMUs with basic features can generate up to a least of 1.5 Terra Bytes (TB) of data in a month. For complete observability and better situational awareness of the grid, there is a need to include multiple PMUs for a large scale grid. Thus, it is evident that huge amount of data will get generated in the long run. In order to extract useful information from the huge data to control smart grid operations, a proper data handling and mining approach is required. This research focuses on the use of two common data clustering methods, namely DBSCAN and k-means to power grid data sets. In addition a novel, hybrid multi-tier clustering approach has been developed and integrated. This section describes the software setup to acquire and handle streaming datasets from PMUs through an openPDC software framework. The rest of sections are organized as: 1) openPDC setup and data extraction process. 2) Application of k-means Clustering. 3) Application of DBSCAN Clustering. 4) Development of a novel method, multi-tier k-means clustering method.

3.1 openPDC set-up and Data Extraction Process

The openPDC is an open source software framework that is maintained by the Grid Protection Alliance (GPA) [23]-[25]. It is a phasor data concentrator (PDC) software that allows utilities to connect multiple PMUs and extract time-synchronized phasor measurements. This data can then be archived and used at a later stage for any post-event analysis. The openPDC software also provide simulated PMU data that can be used for developing and testing streaming algorithms. This test dataset is actually a phasor data

retrieved from a real PMU that is run in a cyclic order when it is used for testing purpose. The data is time-stamped with coordinated universal time (UTC) and archived as a “*.d” file in a source folder of the software. The data extraction stage of the data analysis process includes collecting data from various PMUs, storing it in a PDC and access them through openPDC software. This process is essential for the later part of data analytics work. At this stage, data stored in various file formats is converted to a common “.csv” format. This data format serves as the source for the data analysis phase. For my research, streaming data from only one PMU is considered for investigating the clustering methods. It is a continuous stream of phasor data generated from the openPDC software. This data remains stored with its time stamp information in the archive folder in a computer. The format of the stored data is “.d”. These are shared object code, something like .obj files created by Visual Studio in windows [73]. The data analysis software codes are developed and executed in the MATLAB environment [74]. The openPDC is a software created in C# language and developed in the Visual Studio environment. To bring clarity between the two data formats, there is a need for a common data format. It was realized that “.csv” would be the best fit for storing large amounts of data. A C# based software code was developed to convert the “.d” files into “.csv” format. The code is present in Appendix A. The code developed will allow the user to select data based on inputted time periods (mm/dd/yyyy hh:mm:ss:ffff). Upon receiving time period, the code can be run to correct the formats. The rest of the process is automatic. Once the data gets converted, the program would trigger the Matlab code to run clustering algorithms on historical datasets. The process automatically forms clusters of good, bad (noise) and bordering data points. Figure 8 gives a diagrammatic representation of this process.

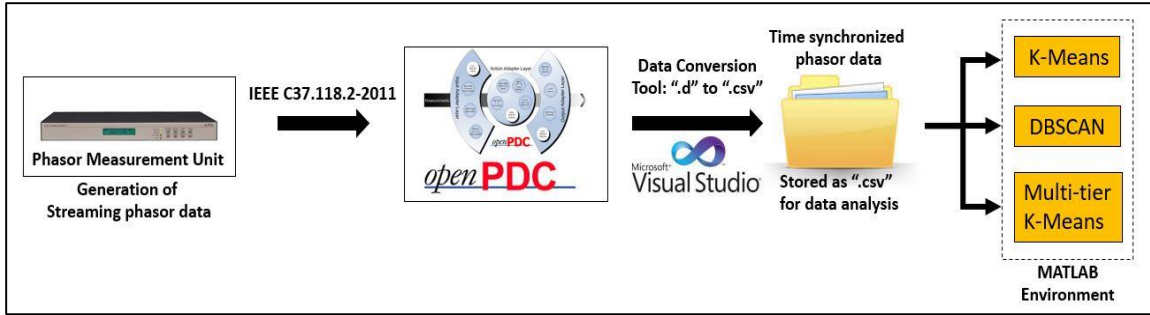


Figure 8: Data Extraction and Conversion Stage

Upon clustering the data into different groups, an average value from the cluster is represented as a sample for decision making. Depending on values of each cluster, alarms are generated to alert the grid operators to take appropriate actions. The entire process of data extraction to alarm generation has been summarized in figure 9.

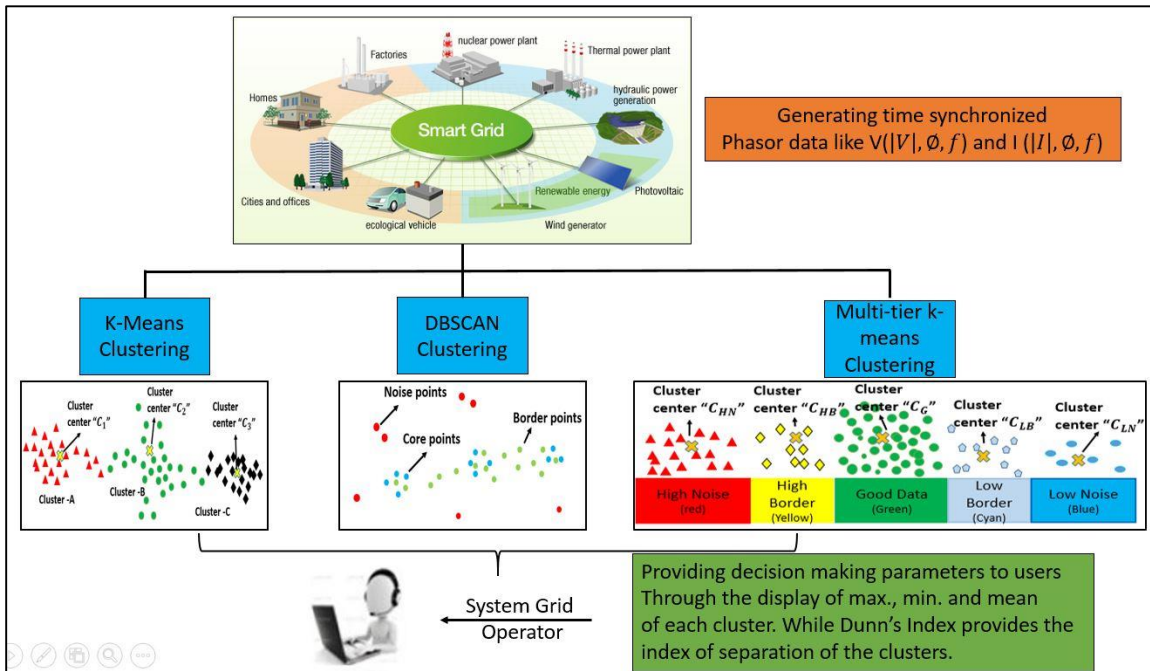


Figure 9: Data Mining and Visualization providing basis for decision making

3.2 Application of k-means clustering

k-means clustering is a kind of representative-based clustering method [28]. Given a d dimensional space having n data points, a representative-based clustering method work towards grouping the n data points into k different clusters [28]. For each cluster, there is

a point, usually the mean or average value, which represents the statistical information for the entire cluster. Thus, we can mathematically represent the clustering algorithm as follows:

If $d = \{x\}_{i=1}^n$, is the d dimensional space, having k clusters, such that each cluster is represented as $C = \{C_1, C_2, \dots, C_k\}$, then the centroid of each cluster is given by

$$\mu_i = \frac{1}{n_i} \sum_{x_j \in C_i} \{x_j\} \quad (8)$$

Where, $n_i = |C_i|$ is the number of points in the cluster C_i .

Using simple brute-force method or exhaustive algorithm, a good clustering method will be to simply generate all possible partitions of the n data points into k clusters, then calculate an optimization score for each of them and retain the one that performs the best with respect to other methods. The number of possibilities to partition n points into k non-empty and disjoint parts is given by:

$$P(n, k) = \frac{1}{k!} \sum_{t=0}^k (-1)^t \binom{k}{t} (k-t)^n \quad (9)$$

k-means is a greedy iterative approach to find clustering that minimizes the sum of squared error (SSE), where the SSE is represented mathematically as [28]:

$$SSE(C) = \sum_{i=1}^k \sum_{x_j \in C_i} \|x_j - \mu_i\|^2 \quad (10)$$

Where, x_j = actual data and μ_i is the mean value.

k-means initializes the cluster means by randomly generating k points in the data space. This is done by generating values randomly within the range for each dimension. Each iteration of k-means consists of the following two steps: (1) cluster assignment, and (2) centroid update.

Given the k cluster means, in the cluster assignment step, each point $x_j \in d$ is assigned to the closest mean, which induces a clustering with each cluster C_i comprising points that are closer to μ_i than any other cluster mean. That is, each point x_j is assigned to cluster C_{j^*} , where

$$j^* = \underset{i = 1}{\overset{k}{\operatorname{arg\,min}}} \{ \|x_j - \mu_i\|^2 \} \quad (11)$$

Given a set of clusters $C_i, i = 1, \dots, k$, in the centroid update step, new average values are computed for each cluster from the point in C_i . The cluster assignment stage and centroid update steps are carried out iteratively until we reach a fixed point or local minima is reached. Practically speaking, it can be assumed that k-means has converged, if the centroids do not change from one iteration to the next. For instance, the iterations can be stopped subject to the following condition being true.

$$\sum_{i=1}^k \| \mu_i^t - \mu_i^{t-1} \|^2 \leq \epsilon \quad (12)$$

where $\epsilon > 0$ and $t =$ the iteration number

The pseudo-code for k-means is given in Figure 10.

Input: $X = \{x_1, x_2, \dots, x_n\}$ (Set of entities to be clustered)

k (Number of clusters)

maxIters (limit of Iterations)

Output $C = \{c_1, c_2, \dots, c_k\}$ (Set of cluster centroids)

$L = \{l(e) | e = 1, 2, \dots, n\}$ (Set of cluster labels of E)

```
foreach  $c_i \in C$  do
     $c_i = x_j \in X$  (e.g. Random Selection)
end
foreach  $x_i \in X$  do
     $l(x_i) = \operatorname{argminDistance}(x_i, c_j) \text{ where } j \in \{1, 2, \dots, k\}$ 
end
If (changed = false) then
    Iter = 0;
end
repeat
    foreach  $c_i \in C$  do
        Update Cluster ( $c_i$ )
    end
    foreach  $x_i \in X$  do
         $\minDist = \operatorname{argminDistance}(x_i, c_j) \text{ where } j \in \{1, 2, \dots, k\}$ 
        If ( $\minDist \neq l(x_i)$ ) then
             $l(x_i) = \minDist$ ;
            changed = true;
        end
    end
    iter++;
until changed = true and iter  $\leq$  maxIters;
```

Figure 10: Pseudo-code for k-means Clustering

The k-means clustering algorithm was applied on the data collected from openPDC. The idea was to create a set of three clusters which will group the entire data into acceptable (good), non-acceptable (bad) and border (ok) values. The results obtained from the k-means clustering method are discussed in chapter 5. Figure 11 gives a diagrammatic representation of cluster formation in k-means algorithm. The figure shows the formation of three clusters based on user input. The centroid values act as user input. The algorithm starts with initial, random values of centroids, C_1, C_2 and C_3 . With every iteration, the centroid values get updated $C_i \rightarrow C'_i \rightarrow C''_i \rightarrow C'''_i$, where $i \in (1, 2, 3)$. The centroid values are updated based on threshold conditions. If the condition does not satisfy, then the

centroid value is no longer updated, which basically imply that the no other value from the input dataset can be fit into the cluster. In figure 11, we can see that the centroid $C_1'' = C_1'$ for the red cluster. This value remains unchanged after the first iteration as no other data point (from the yellow or the green cluster) are included in the red cluster for the rest of the k-means process.

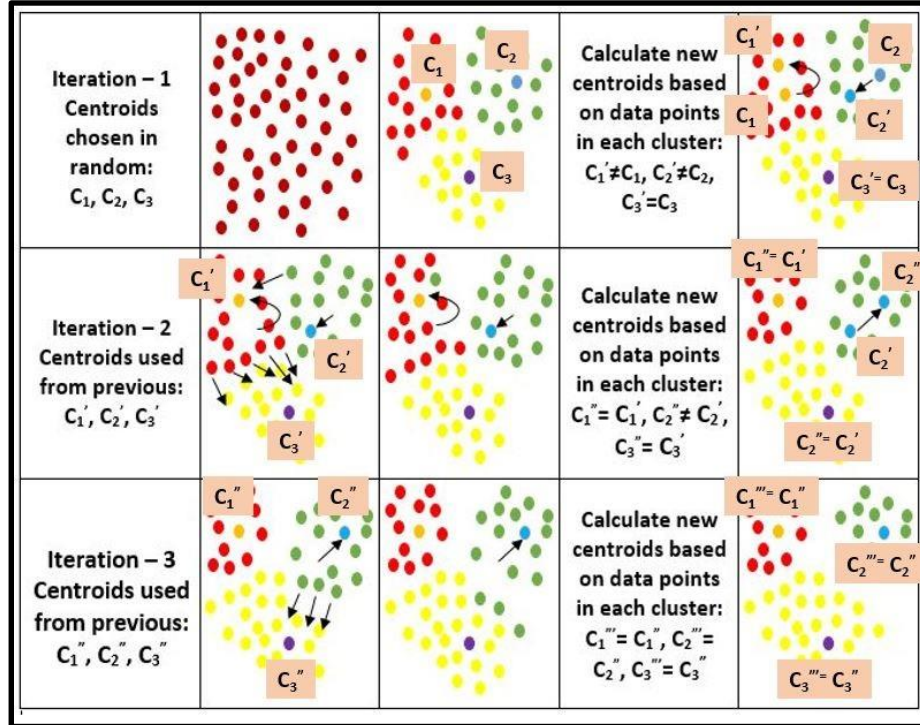


Figure 11: k-means cluster formation

3.3 Application of Density-based spatial clustering of applications with noise (DBSCAN) Clustering

Density based clustering (DBSCAN) of data points focuses on the density of points to determine the clusters instead of using only the distance between points [28]. A ball of radius ϵ is defined around a point $x \in \mathbb{R}^d$, called the ϵ neighborhood of x , as follows:

$$N_\epsilon(x) = B_d(x, \epsilon) = \{y \mid \delta(x, y) \leq \epsilon\} \quad (13)$$

Here, $\delta(x, y)$ represents the distance between points x and y , which is usually assumed to be Euclidean distance, that is, $\delta(x, y) = \|x - y\|_2$ [28]. Now, for any point $x \in d$, it is considered that x is a core point, if there are at least $minpts$ points in its ϵ -neighborhood. That means, x is a core point if $|N_\epsilon(x)| \geq minpts$, where $minpts$ is a user defined local density threshold. A border point is defined as one which does not meet the $minpts$ threshold, that is it has $|N_\epsilon(x)| < minpts$, but it belongs to the ϵ -neighborhood of some core point z , that is, $x \in N_\epsilon(z)$. Finally, if a point is neither a core nor a border point, then it is called a noise point or an outlier.

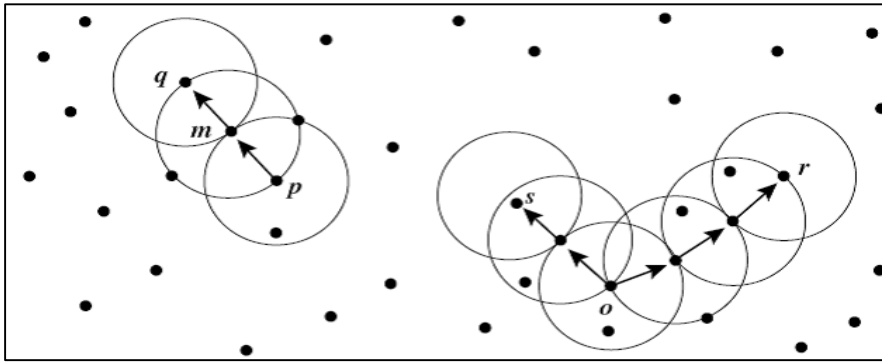


Figure 12: Growth of clusters based on their density

It is considered that a point x is density reachable from y , if there exist a chain of points, x_0, x_1, \dots, x_i , such that $x = x_0$ and $y = x_i$ and x_l is directly density reachable from x_{l-1} for all $l = 1, 2, \dots, i$. In Figure 12, point “r” is density reachable from “o”, while point “q” is directly density reachable from “m”. The pseudocode for a DBSCAN clustering algorithm is given in Figure 13.

Input: $X = \{x_1, x_2, \dots, x_n\}$ (set of entities to be clustered)

eps = Minimum distance between two points to be clustered (D).

$MinPts$ = Minimum number of points that should be in a cluster to consider it as a border group (Pt).

Output: $L = \{l(x) | x = 1, 2, \dots, n\}$ (Set of cluster labels of x)

```
DBSCAN (Input Set X, Pt, D)
  foreach  $x_i$  in the input set do
    If ( $x_i$  is not in any cluster) then
      If ( $x_i$  is a core point) then
        Generate a new clusterID.
        Label  $x_i$  with clusterID.
        expandCluster ( $x_i$ , X, Pt, D, clusterID)
      else
        Label ( $x_i$ , NOISE)
      end
    end
  end
end
end
end
expandCluster ( $x_i$ , X, Pt, D, clusterID)
  put  $x_i$  in seed queue
  While the queue is not empty
    extract  $c$  from the queue (where  $c \in X$  and  $c \neq x_i$ )
    retrieve the neighborhood (eps) of  $c$ 
    If (there are at least minPts neighbors) then
      foreach neighbor  $n$ 
        If ( $n$  labeled NOISE) then
          Label  $n$  with clusterID
        elseif ( $n$  is not labeled) then
          Label  $n$  with clusterID
          Put  $n$  in the queue
        end
      end
    end
  end
end
end
end
end
```

Figure 13: Pseudo-code for DBSCAN clustering

Figure 14 gives a diagrammatic presentation of DBSCAN clustering. From Figure 14 it is evident how, that the clusters C_1 , C_2 and C_3 (in green) which consists of core points (i.e., satisfies both conditions of minpts. and eps.) together form a core group of data points which have values close to each other. Whereas, the border group (in yellow) consists of points satisfying the border point properties (i.e., eps, not minpts.). Finally, the isolated red colored points form the noise group.

3.4 Need for a hybrid method

The purpose of introducing the concepts of data mining into power systems is to aid in

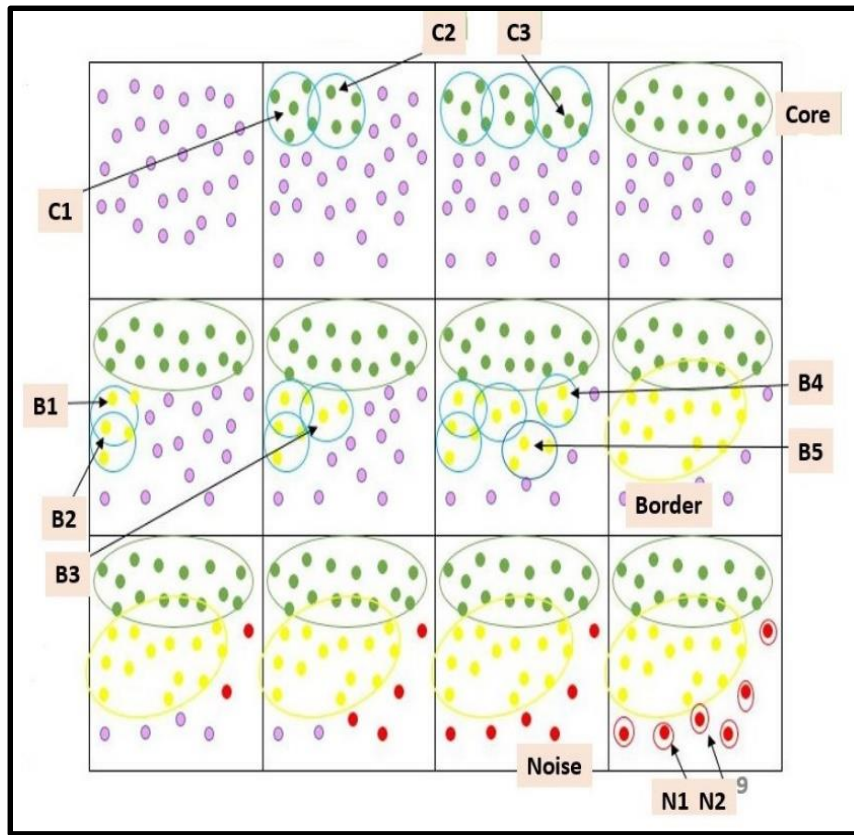


Figure 14: DBSCAN Cluster formation

taking critical decisions by exploring data. Thus, allowing the smart grid to be reliable and efficient in generating and delivering power. This research focuses on application of data mining techniques to create a robust monitoring and alarm system. Thus, serving as a software interface that can provide services like data mining, data visualization, intrusion detection and mitigation, alert services, topology capturing and state estimator and forecasting. A diagrammatic representation of the entire software framework is presented in Figure 15.

This research work deploy common clustering methods, namely k-means and DBSCAN on PMU data to study its grouping and its effects in capturing anomalies. A comparative analysis of this study has been presented in the results section, in chapter 4. The results obtained demonstrated that there is a need for such clustering methods to accurately track

the changes in load values on the transmission lines. These techniques can also detect any contingency scenarios and alert the system operators to take corrective measures to mitigate the problem. A hybrid data clustering (multi-tier k-means) method was developed to with improve performance in capturing anomalies. The method has been discussed in the following section.

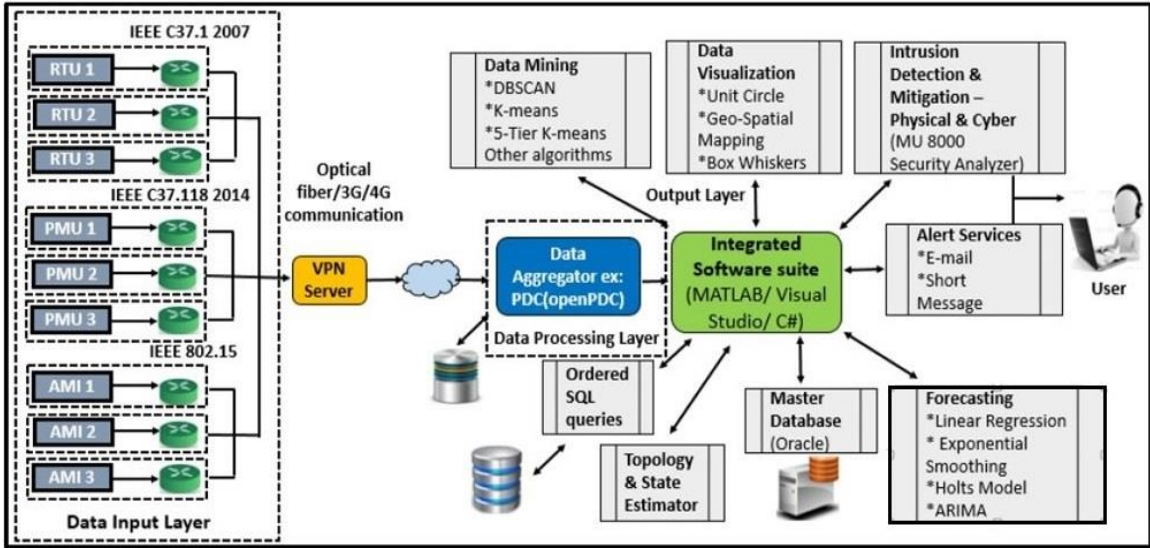


Figure 15: Integrated software framework for PMU/AMI data

3.4.1 Development of Multi-tier k-means method

There are several variations of k-means discussed in literatures, where the original k-means algorithm has been modified to suit the type of data [75]-[78]. They include k-medians, k-medoids, k-means ++, weighted k-means, fuzzy c-means, Kd- trees, and spherical k-means. k-means has been popular for its speed and accuracy of clustering data objects. Although DBSCAN and k-means clustering algorithms proved to accurately classify normal steady state conditions (without faults) they failed to capture faults. It was mainly because these algorithms could not accurately identify the border lying data points, which are critical to generate alarm. While DBSCAN could correctly track the shift in data points due to changing demand in the system, the computation time was too slow. On the other hand, k-

means, although being one of the fastest clustering algorithms, could not correctly identify the good data points from the border lying ones, as far as the phasor data was concerned. Hence, there is a need for a method that capture both steady-state (normal) and abnormal conditions of the grid. This is carried out using multi-tier k-means method.

The multi-tier k-means algorithm can be described in two stages. The first stage is similar to k-means clustering, where three clusters are formed. The algorithm begins with three initial clusters. Each of the three clusters represent normal, abnormal (faulty) data in low threshold and abnormal (faulty) data in the high threshold ranges. The normal data indicates ideal steady state scenario, i.e., the power system is not under any stress. The high and low fault data clusters indicate that a contingency scenario (a line-to-ground fault) was recorded where magnitude of voltage (V) or current (I) or frequency (F) reached very large or very small values. The algorithm allows the user with two options. Option 1 allows user to choose the centroid values of the three clusters. Option 2 chooses centroid values automatically. In case of automatic choice, the user is prompted to enter the ideal line parameters like voltage or current magnitude. The ideal value of the line parameter (current, voltage or frequency) is taken as centroid of one of the cluster and the other two cluster centroids are chosen at $\pm 1\%$ threshold of user input. Based on this input, the clusters are formed. There may or may not be three clusters, depending on the PMU data. Once the initial clusters are formed, the algorithm enters the second phase of execution. This second phase is developed with the goal of identifying the border regions (outliers) near the clusters. Based on specific threshold values (shown in equations 14-17), the data points in the groups resulting from the initial stage of clustering, are further categorized into border values. These are overlapping data between the three clusters from stage 1, which have the

potential to detect contingencies. The second stage also acts as a way of visualizing the data. Each resulting cluster are subjected to certain conditions. A color scheme has been developed which categorize data points as follows:

1. Good data – Green
2. High bordering data – Yellow
3. Low bordering data – Cyan
4. High noise (bad) data – Red
5. Low noise (bad) data – Blue

Figure 16 shows a diagrammatic representation of the color scheme.

The following variables indicate the centroid values of individual cluster.

1. CN_{good} – Centroid value of a cluster with good data points during iteration N
2. CN_{low} – Centroid value of a cluster with low range of data points during iteration N
3. CN_{high} – Centroid value of a cluster with higher range of data points during iteration N.

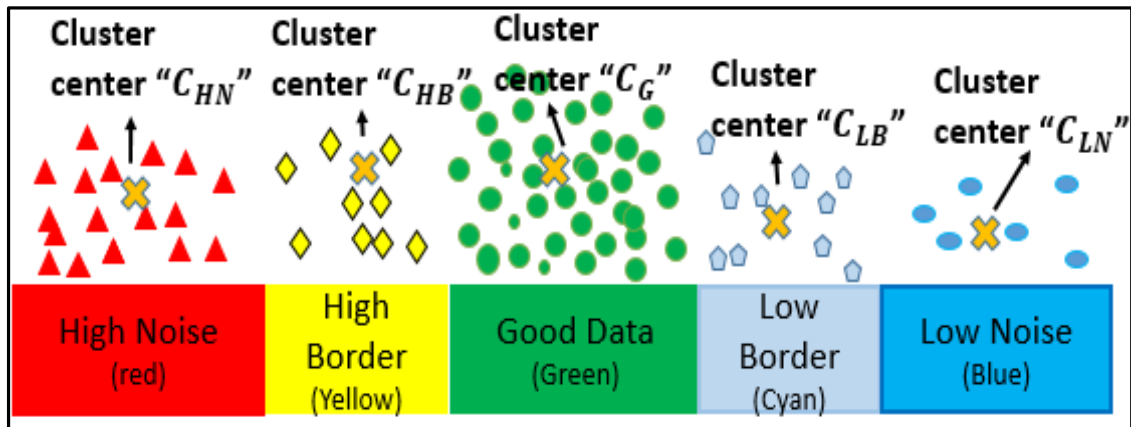


Figure 16: Multi-Tier k-means Cluster Categories

The novel hybrid algorithm was applied to voltage magnitude, current magnitude and frequency of the PMU datasets. The thresholds for each parameter, namely voltage magnitude, current magnitude and frequency to be grouped into five clusters as per the equations 14-17.

The algorithm can be mathematically represented as follows:

Stage – I Creation of 1-3 clusters based on the phasor data input:

If $d = \{x\}_{i=1}^n$, is the d dimensional space, having k clusters, such that each cluster is represented as $C = \{C_1, C_2, \dots, C_k\}$, then the centroid of each cluster is given by

$$\mu_i = \frac{1}{n_i} \sum_{x_j \in C_i} \{x_j\} \quad (14)$$

Where, $n_i = |C_i|$ is the number of points in the cluster C_i .

Since, the number of clusters are set at 3, $C = \{C_1, C_2, C_3\}$, using brute force, iterative method, the data points are clustered into any one of the three clusters. Each iteration results in new centroid values. The iterations continue until the centroids stop shifting to form a new centroid center points, that is, $|\mu_i - \mu'_i| = 0$. Here, μ'_i is the new centroid and $i = \{1, 2, 3\}$.

Stage – II Extending the clusters from stage I to 4 or 5 clusters based on any border data present:

The clusters from stage I are further analyzed to detect the ‘outliers’ – data points that are not exactly good data. These border clusters lie close enough to the upper or lower threshold values. They are to not be considered dangerous to the system. The density of these border points acts as an indicator for contingency scenarios. Each of the clusters are tested, if the data points lie within the following thresholds in the cluster:

$$\max. \{C_i\} \leq \left[1 + \frac{P}{100} * X_{ideal}\right] \&\& \min. \{C_i\} \geq \left[1 - \frac{P}{100} * X_{ideal}\right] \quad (15)$$

If the maximum and the minimum of the cluster falls within the range $\left(\left[1 - \frac{P}{100} * V_{ideal}\right], \left[1 + \frac{P}{100} * X_{ideal}\right]\right)$, where 'P' is a set percentage of points to be considered as good or bad and X_{ideal} is the ideal voltage (V) or current (I) or frequency (F) of the line that is known or considered to be normal value.

$$\max. \{C_i\} > \left[1 + \frac{P}{100} * X_{ideal}\right] \&\& \min. \{C_i\} \geq \left[1 - \frac{P}{100} * X_{ideal}\right] \quad (16)$$

Where, $X_{ideal} = \{V, I, F\}$.

If certain number of points in a cluster are greater than large threshold values, then they are classified as high noise. This indicate that the system is under low load (demand) condition. The rest of the points in the cluster belong to the core group.

$$\max. \{C_i\} \leq \left[1 + \frac{P}{100} * X_{ideal}\right] \&\& \min. \{C_i\} < \left[1 - \frac{P}{100} * X_{ideal}\right] \quad (17)$$

This condition suggests that data points that have values lower than the low threshold values are low border points. This indicate the presence of high load (demand) conditions.

$$\max. \{C_i\} > \left[1 + \frac{P}{100} * X_{ideal}\right] \&\& \min. \{C_i\} < \left[1 - \frac{P}{100} * X_{ideal}\right] \quad (18)$$

Equation 17 illustrate the presence of core, high and low border points in the cluster.

The multi-tier k-means method was developed with the knowledge that a k-means algorithm creates clusters based on their distance from a pre-defined centroid value. As the average value keeps updating, there is an inclination of the clusters to move further apart

from each other. It is this property of the k-means algorithm that is used in stage 1 to isolate the noise cluster from the rest of its data. The second stage is used to differentiate the core and the border (high and low) points. Thus, if a cluster extremities (max. and min.) do not match any of the conditions presented in equations 15-18, then it is categorized as noise cluster.

Figure 17 shows the pseudo-code for multi-tier k-means:

<pre> Multi-tier k-means: Phase I foreach c_i ∈ C do c_i = V + n * P_c * V, where, n ∈ {0, +1, -1} end foreach x_i ∈ X do l(x_i) = argminDistance(x_i, c_j), where j ∈ {1, 2, 3} end changed false; iter 0; repeat foreach c_i ∈ X do Update Cluster (c_i) end foreach x_i ∈ X do minDist = argminDistance(x_i, c_j), where j ∈ {1, 2, 3} If (minDist ≠ l(x_i)) then l(x_i) = minDist; changed = true; end end iter ++; until changed = true and iter ≤ MaxIters; </pre>	<pre> Formation of Border Clusters based on data : Phase II If unique(l(x) > 1) then foreach l(x) ∈ C do If (mx ≤ (1 + Pc) * V && mn ≥ (1 - Pc) * V) then l(x_i) = goodData; end If (mx > (1+Pc) * V) then l(x_i) = highNoise; end If (mn < (1-Pc) * V) then l(x_i) = lowNoise; end If (mx ≥ (1-Pc) * V && mx ≤ (1+Pc) * V && mn < (1-Pc) * V) then l(x_i) = goodData; ∪ {lowBorder noise (for all l(x) ∈ mn < (1-Pc) * V)} end If (mn ≥ (1-Pc) * V && mn ≤ (1+Pc) * V && mx > (1+Pc) * V) then l(x_i) = goodData; ∪ {highBorder noise (for all l(x) ∈ mx > (1+ Pc) * V)} end end else foreach l(x) ∈ C do If (mx ≤ (1+Pc) * V && mn ≥ (1-Pc) * V) then l(x_i) = goodData; end If (mx > (1+Pc) * V) then l(x_i) = highNoise; end If (mn < (1-Pc) * V) then l(x_i) = lowNoise; end end end </pre>
--	---

Figure 17: Pseudocode of multi-tier k-means (Phase I & II)

Figure 18 illustrates two case scenarios that can be captured using multi-tier k-means. For the first case, it is evident that the three clusters (red, yellow and green) with centroids C₁, C₂ and C₃ that were formed in stage I gets further divided into five separate clusters (red, yellow, green, cyan and blue) indicating that the system is under a lot of stress. For the second case however, only three clusters are formed (green, cyan and blue) indicating the system is under a high load condition, which finally resulted in a fault. The results obtained

from the application of the multi-tier k-means method are discussed in details in **chapter 5.**

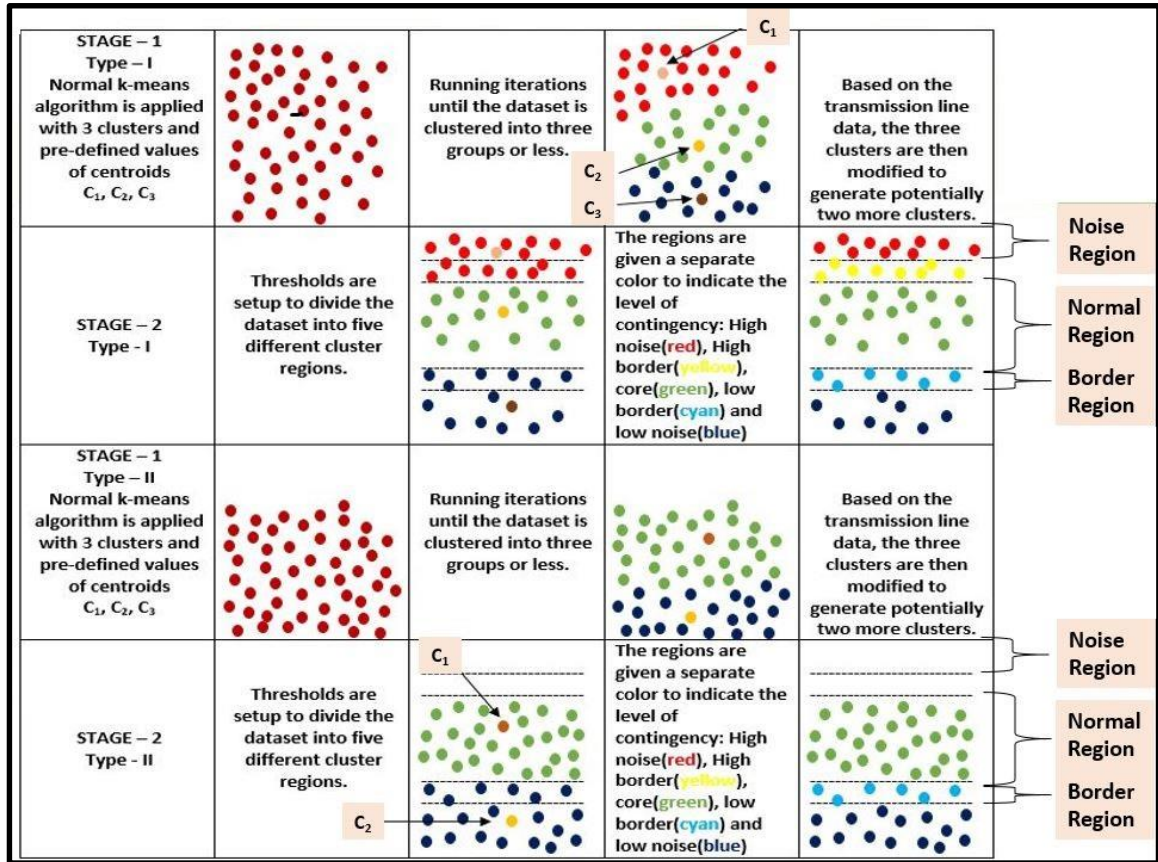


Figure 18: Cluster formation in multi-tier k-means

3.4.2 Advantages of using proposed Multi-tier k-means method

There are three major advantages of using multi-tier k-means method. They are as follows.

1. The proposed approach dynamically form varying number of clusters. This could be anywhere from 1 to 5 clusters depending on the data thresholds and fault type. The need for such varying clusters in the approach is to clearly identify specific type of faults that may not be visible otherwise. For example, the addition of two new clusters; 1) low and 2) high border clusters indicate that the values are

beginning to reach their end of thresholds. Detecting those border points will help operators to caution and arrive at pro-active remedial actions to stabilize the grid.

2. Capable of clearly distinguishing the good, bad and the noisy data regions based on thresholds.
3. Requires only 1 parameter (voltage or current) as input, initiate the algorithm.

Thus the application of multi-tier k-means method can not only track changes in load conditions, but also aids in decision making of the system that enable operators to generate alarms. This will help avoid contingency scenarios that disrupt the normal functions of the power grid.

IV. FORECASTING OF ELECTRIC UTILITY DATASETS

The electric power system network created by mankind is one of the most complex objects on this planet. It generates electricity and distributes to close to 7 billion people on this planet. Naturally, for a power generation company, forecasting of supply, demand and pricing becomes imperative for normal functioning. Load forecasting has been a conventional and important process in electric utilities since the early 20th century [79]. Due to the deregulation of the electric utility industry, utilities tend to be conservative about any infrastructure upgrade that burden the assets with more stress [79]. Consequently, the load forecasting process for planning, operations and maintenance has become more crucial than before. In addition, participation in the electricity market require the utilities to forecast their loads accurately and consistently. Nowadays, with the innovation of new smart grid technologies, load forecasting plays an active role in planning demand side management of electric vehicles, distributed and renewable energy resources.

4.1 Forecasting in electric utilities

To keep up with the growing demand and to maintain stable and efficient generation of power, utilities need to plan and schedule the resources optimally. To accomplish this task, load forecasting is an important process for planning. As a result, the forecasting task span across number of departments within a utility. While the operations and planning department decide on what a utility should do given a certain forecast of energy demand, the trading department uses the billing data to generate a price rate for the following day. The business needs for forecasting of a utility can be summarized as follows:

1. Energy trading Whether a utility purchases its own energy supplies from the market place, or outsources this function to other parties, appropriate load forecast methods are essential for purchasing energy. The utilities can perform bi-lateral purchases and asset commitment in the long term, e.g., 10 years ahead. They can also do hedging and block purchases one month to 3 years ahead, and adjust (buy or sell) the energy purchase in the day-ahead market [79]. So, energy trading is a key business need that rely on forecasting.

2. Transmission and distribution (T&D) planning [80]

The utilities continuously need to properly maintain and upgrade their sub-system to meet the growing demand for its services and to improve its reliability. Sometimes, the utilities hedge the real estate to place any further substations. The planning decisions heavily rely on spatial load forecasting, where the projections of customers and volume of load demands need to be made ahead for better planning resources.

3. Operations and maintenance

In daily operations, load patterns obtained during the load forecasting process guide the system operators to make switching and loading decisions, and schedule maintenance outages [79]. Hence, there is a need for scheduling such maintenance activities that rely on forecasting methods.

4. Demand side management (DSM)

Although a large number of DSM activities are done for daily operations, it is worthwhile to separate DSM from the operations category due to its importance in

smart-grid era. A load forecast can support key decisions in load control, smart energy reduction and in predicting end user behavior patterns.

5. Financial planning

The load forecasts can help the utilities to project medium and long-term revenues and thus aid in making decisions related to acquisitions, project budgets and plan new technologies [79]. Depending on the individual business models and utility requirements in discussed category, the minimum update cycle and maximum horizon of the forecasts are summarized in Table 1.

Table 1: Update cycle for forecasting [79]

Areas	Minimum Updating Cycles	Maximum Horizon
Energy Purchasing	1 hour	10 years and more
T&D Planning	1 day	30 years
Operations	15 mins	2 weeks
DSM	15 mins	10 years and more
Financial Planning	1 month	10 years and more

Classification of load forecasts in utilities

There is no single forecasting technique that satisfy all needs of a utility. Hence, classification of forecasting techniques in utilities is dependent not only on the business need of the organization, but also dependent on factors that affect the consumption of electricity like weather or climatic changes, human activities, holidays, or other factors. Weather encompasses information about meteorological components for a certain place, like temperature, humidity, wind, rainfall, etc. Climate includes the same components but are recorded for longer periods of time. Of the various meteorological components temperature usually plays the most important motivating factor for the use of electricity [79]. For example, users need more heating or cooling depending on whether it is colder or warmer outside. The dependence of power consumed with respect to the atmospheric

temperature can be plotted as shown in Figure 19. It shows how the power consumed in the Mid-Atlantic region from 2012 to 2015 at 10 am varies with temperature. It is observed that the scatter plot forms a deformed U shaped pattern with values of demand peaking towards high and low temperature values. This can be explained by the increased heating or cooling needs of human beings when the outside temperature is too hot or too cold. The data used here was collected from PJM Corporation website [75]. Nowadays, temperature forecast can be accurate up to a day ahead and at least somewhat reliable up to about two weeks in ahead [79].

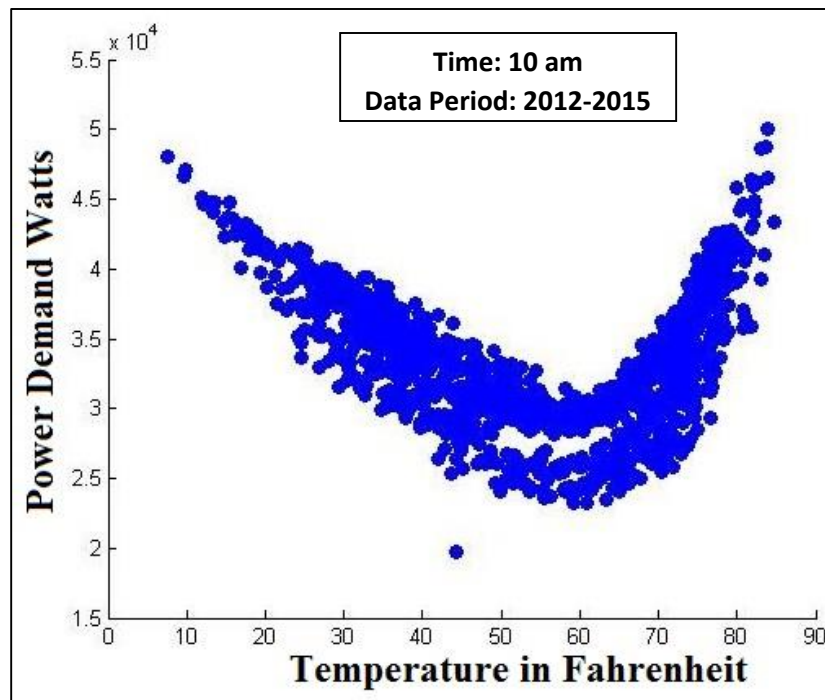


Figure 19: Load vs Temperature scatter plot for Mid-Atlantic region [92]

The impact of human behavior on energy consumption can be studied on the basis of several time related factors. On an hourly data, the impact varies over the day of the week and the month of the year. This calendar information is fairly certain over the next decade. For example, the power consumed during business days would vary relative to holidays [79]. Figure 20 shows how power consumption vary over an entire day for a holiday

compared to a business day. It also shows that power consumption vary by hour of the day. For example, power consumed at 2 am in the morning will be low and fairly remain constant over different days as opposed to that at 8 pm of the same day.

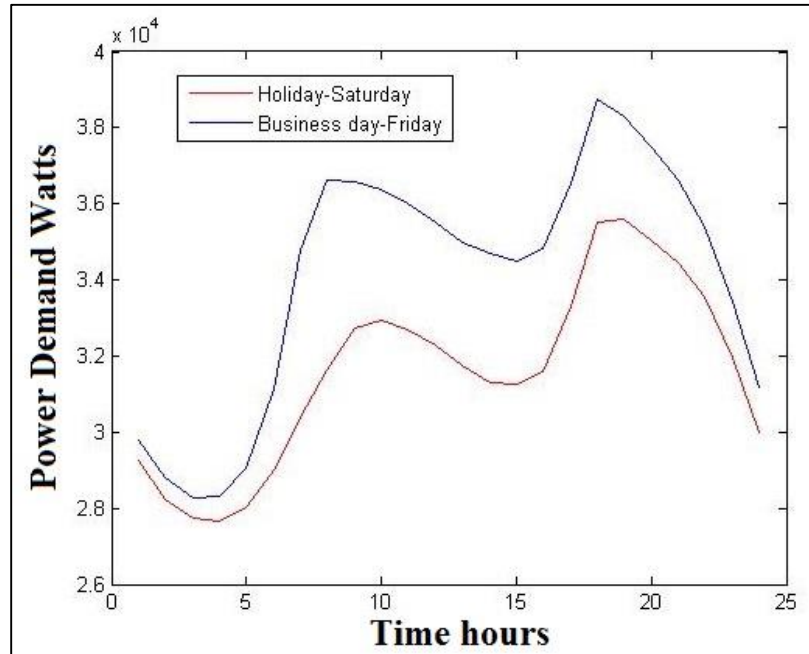


Figure 20: Load vs hour of day for business day and holiday

Again, in monthly or quarterly resolution, the usage of electricity may depend on various economic factors. For example, the power consumed during the recession of 2009, was much less than that of the earlier years [79]. This was mostly because a number of offices were shut down that resulted in conservative usage of power. With the advancement of econometric techniques, the economics information can be relatively accurate up to one year ahead, and be inaccurate but reliable up to 3 years ahead for load forecasting purpose. In the annual resolution, both climate and economics can affect the energy consumption. However, due to the unavailability of both inputs, the system level load forecast can be only obtained by simulating various scenarios [79]. Long term energy consumption on the circuit level is affected by urban development, which can be realized by land use changes. The land use information is normally accurate within one year, inaccurate but reliable up

to 5 years. Although some counties can provide a 30 years ahead urban development plan, it is still not clear what exactly would happen year by year during the next 30 years.

The availability of information regarding various factors affecting the amount of energy usage can facilitate various forecasting periods. Types of load forecasting can be broadly categorized into very short term load forecast (VSTLF), short term load forecast (STLF), medium term load forecast (MTLF) and long term load forecast (LTLF) [79].

In VSTLF, temperature, economics and land use information can all be optional, because the load in the near future can be forecasted by the load in the past. Since both economics and land use information are relatively constant in the short time span (less than 2 weeks), they can be optional in STLF. Temperature, however plays a key role in STLF. In MTLF, temperature cannot be predicted accurately for the coming 3 years. Therefore, simulated scenarios of temperature based on the local temperature history can be used into the model. While the economics is predictable and affect the mid-term load consumption, it is required in MTLF. Land use information is optional in VSTLF, STLF, and MTLF because the land use could barely change a lot during a 3-year period. However, in the long term, land use change is the major factor that drives the load. Therefore, land use information is required in LTLF. Table 2 gives an account of required fields for successful forecasting for various periods of time [79].

In this research, both parametric and non-parametric models of prediction have been studied for both midterm (MTLF) and short term (STLF) load forecasts. A combination of non-parametric and time series model as well as parametric and time series model has been applied on load data obtained from PJM website for the Mid-Atlantic region. Their performances have been compared under standardized indices.

Table 2: Classification of load forecasts [79]

Forecast type	Temperature	Economics	Land Use	Updating Cycle	Period
VSTLF	Optional	Optional	Optional	≤ 1 Hour	1 Day
STLF	Required	Optional	Optional	1 Day	2 Weeks
MTLF	Simulated	Required	Optional	1 Month	3 Years
LTLF	Simulated	Simulated	Required	1 Year	30 Years

4.2 Investigation of forecasting models and need for novel forecasting

In this research work a collection of common forecasting models have been applied on PJM data sets for future demand allocation. The performance has been measured using standardized statistical metrics like mean absolute error (MAE), mean absolute percentage error (MAPE) and root mean squared error (RMSE). The results have been gathered for short-term and mid-term load forecasting. These results obtained are discussed in chapter 6. Finally, a novel hybrid forecasting method resulting from the combination of the two existing methods is also tested and compared against the standard forecast methods. The prediction models that are proposed and investigated are as follows:

1. Locally weighted scatterplot smoothing (LOWESS)

It is a semi-parametric, data driven method of forecasting that is used to generate a fitted line through a scatter plot involving two variables [81].

2. Random forest model (RF)

It is an ensemble learning method, used for classification, regression and other tasks. It works by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees [82].

3. Seasonal autoregressive integrated moving average (SARIMA) model

For time-series data, an autoregressive integrated moving average (ARIMA) model is a generalization of an autoregressive moving average (ARMA). These models are fitted to time series data either to better understand the data or to predict future points in the series (forecasting). These are applied to where data show evidence of non-stationarity, where an initial differencing step (corresponding to the "integrated" part of the model) can be applied to reduce the non-stationarity. Often time-series possess a seasonal component that repeat every 's' observations. For monthly observations $s = 12$ (12 in 1 year), for quarterly observations $s = 4$ (4 in 1 year). In order to deal with seasonality, ARIMA models have been generalized to form SARIMA models.

4.3 LOWESS model

LOWESS is a non-parametric regression method that combine multiple regression models in a k -nearest-neighbor-based meta-model. Originally proposed by Cleveland in 1979 [81] and was later developed Cleveland and Devlin in 1988 [81]. It is based classical methods of forecasting, like least squares regression (LSR). It addresses situations in which the classical forecasting procedures do not perform well or cannot be effectively applied without undue labor. LOWESS combines much of the simplicity of linear least squares regression with the flexibility of non-linear regression [81]. It does this by fitting simple models to localized subsets of the data to build up a function that describes the deterministic part of the variation in the data, point by point. The popularity of this method lies in the fact that data analyst is not required to specify a global function of any form to fit a model to the data, only to fit segments of the data [81]. At each point in the data set a low-degree polynomial is fitted to a subset of the data, with explanatory variable values near

the point whose response is being estimated. The polynomial is fitted using weighted least squares, giving more weight to points near the point whose response is being estimated and less weight to points further away. The value of the regression function for the point is then obtained by evaluating the local polynomial using the explanatory variable values for that data point. The LOWESS fit is complete after regression function values have been computed for each of the n data points [81]. However, it is computationally intensive and practically impossible to use LOWESS at a time, when LSR was being developed. This is because computers were unable to handle long and complex computations in that period. A smooth curve through a set of data points obtained with this statistical technique is called a lowess curve, particularly when each smoothed value is given by a weighted quadratic least squares regression over the span of values of the y -axis scatter-gram criterion variable. Many of the details of this method, such as the degree of the polynomial model and the weights, are flexible [81]. The range of choices for each part of the method and typical defaults are briefly discussed in the next section.

The subsets of data used for each weighted least squares fit in LOWESS are determined by a nearest neighbors algorithm [81]. A user-specified input to the procedure called the "bandwidth" or "smoothing parameter" determines how much of the data is used to fit each local polynomial. The smoothing parameter, α , is a number between $(\lambda + 1)/n$ and 1, with λ denoting the degree of the local polynomial. The value of α is the proportion of data used in each fit. The subset of data used in each weighted least squares fit comprises the $n\alpha$ points (rounded to the next largest integer) whose explanatory variables values are closest to the point at which the response is being estimated. α is called the smoothing parameter because it controls the flexibility of the LOWESS regression function. Large

values of α produce the smoothest functions that wiggle the least in response to fluctuations in the data. The smaller α is, the closer the regression function will conform to the data. Using too small a value of the smoothing parameter is not desirable, however, since the regression function will eventually start to capture the random error in the data. Useful values of the smoothing parameter typically lie in the range 0.25 to 0.5 for most LOWESS applications. Figure 21 shows a LOWESS curve fitted through a scatter plot representing demand vs. temperature data collected over a period of 4 years at 1 am. The data was gathered from PJM website and plotted for only business days.

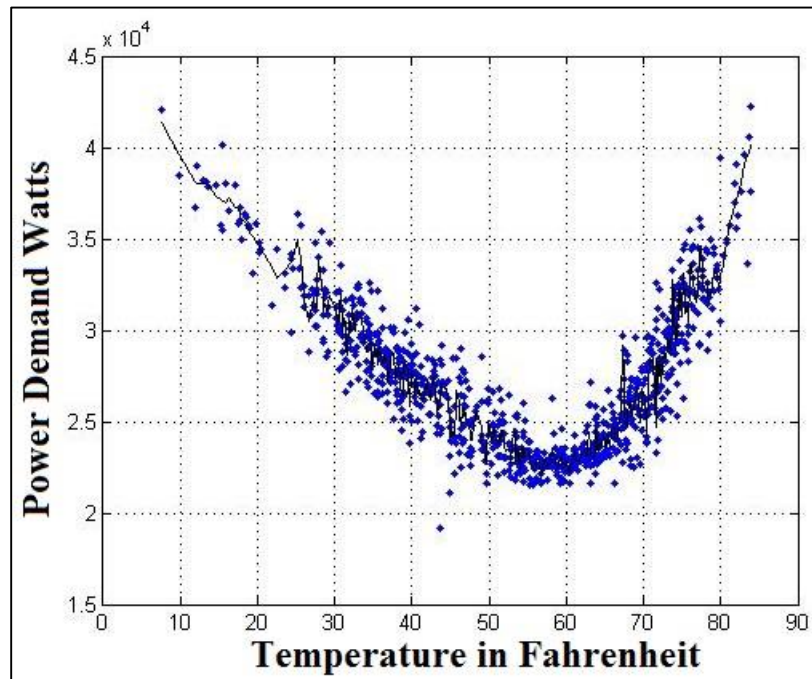


Figure 21: LOWESS curve fitting

In [70], the authors have applied this method of curve fitting through a load vs. temperature data scatter. In order to evaluate the smoothing parameter α , they have used the cross-validation algorithm as follows:

$$\alpha = \operatorname{arg}_{\alpha} \min \left\{ \frac{1}{n} \sum_{i=1}^n [Y_i - \hat{f}_{\lambda,[-i]}(X_i)]^2 \right\} \quad (19)$$

where (X_i, Y_i) is the i^{th} observation, and $\hat{f}_{\lambda,[-i]}$ is the estimated function omitting the i th observation using bandwidth α . LOWESS uses a weight function which gives maximum weight to data points that are closest to the points of estimation, whereas points have lesser weights if they are further away. The use of weights is based on the idea that points nearer to each other are more likely to be related than those further apart. The following equations give the set of weight functions properties that were set by Cleveland [81].

$$W(x) > 0 \text{ for } |x| < 1; \quad (20)$$

$$W(-x) = W(x); \quad (21)$$

$$W(x) \text{ is a non-increasing function for } x \geq 0; \quad (22)$$

$$W(x) = 0 \text{ for } |x| \geq 1. \quad (23)$$

Where ‘W’ is the weight function and x is a data point.

The weight function used for LOWESS traditionally is as follows:

$$W(x) = \begin{cases} (1 - x^2)^3 & \text{for } |x| < 1 \\ 0 & \text{for } |x| \geq 1 \end{cases} \quad (24)$$

The pseudocode for LOWESS can be represented as shown in Figure 22 below:

```

LOWESS (Input Set X(m,n),  $\alpha$ )
  foreach  $x_i$  in the input set do
     $W_k(x_i)$  for all  $x_k$ , where  $k = 1, \dots, n$ 
    If ( $W(x_k)$  is 0 for any  $x_k$  with  $x_i$  as the center)
       $x_k$  is the  $r^{\text{th}}$  nearest point to  $x_i$ 
      Initially fitted value  $\hat{y}$  at each  $x_i$  is a fitted value of  $d^{\text{th}}$  degree polynomial
      fitted to the data using weighted least squares with weights  $W_k(x_i)$ 
      referred to as locally fitted regression
    end
    If ( $(y_i - \hat{y})$  for  $x_k > (y_i - \hat{y})$  for  $x_{k-1}$ )
       $\delta_i(x_k) < \delta_i(x_{k-1})$ 
    else
       $\delta_i(x_k) > \delta_i(x_{k-1})$ 
    end
    Update fitted values for  $x_i$  with new weights  $\delta_i W_k(x_k)$ 
  end
end

```

Figure 22: Pseudocode for LOWESS Smoothing

4.4 Random forest (RF) model

Random forest is an ensemble approach that can also be thought of as a form of nearest neighbor predictor. It was developed by Breiman in the year 2001[82]. Ensembles are a divide-and-conquer approach used to improve performance of classification. The main principle behind ensemble methods is that a group of “weak learners” can come together to form a “strong learner”. Figure 23 below shows a plot representing the relationship between temperature and ozone. The data used here was gathered from the website of the University of Cologne and provided by Rousseeuw and Leroy in the year 1986. Figure 23 mediate that each classifier, individually, is a “weak learner,” but taken together they become a “strong learner”. The data to be modeled are the blue circles. It is assumed that they represent some underlying function along with noise. Each individual learner is shown as a gray curve. Each gray curve (a weak learner) is a fair approximation to the underlying data. The red curve (the ensemble “strong learner”) can be seen to be a much better approximation to the underlying data.

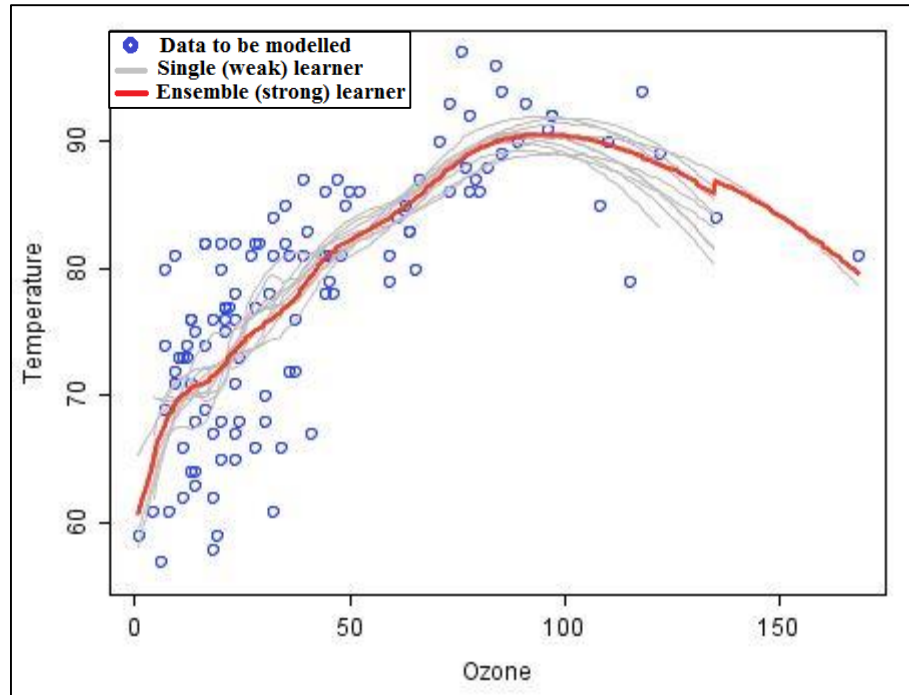


Figure 23: Ozone vs. Temperature plot [83]

The fundamental component of random forests is a “Decision tree” (DT). From an ensemble point of view, a DT is basically a weak learner. In a decision tree, the input data to be classified, is entered at the top layer. As it trickles down to the bottom, the data gets split into smaller sets based on the conditions present at each node that the data comes across. The functioning of a DT can be explained with the help of an example represented in figure 24. In the example, a decision tree is used to conclude whether or not to go out to play [83]. There are 14 data points that are dependent on weather variables and atmospheric conditions like sunny, rainy, windy, etc. The figure shows how the entire data set gets broken into smaller decision sets at every node based on the independent (decision) variables.

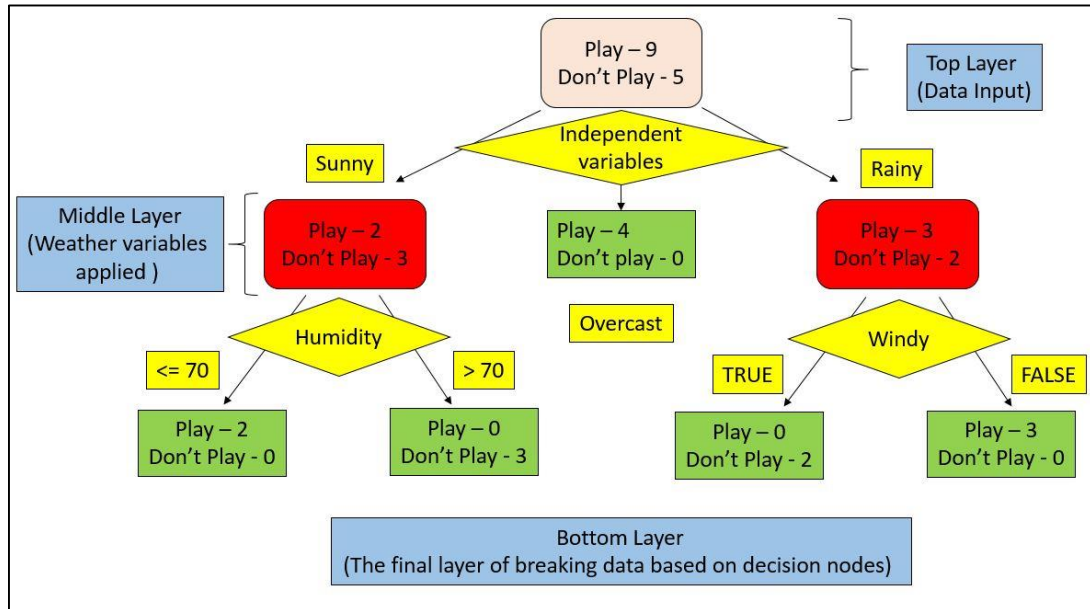


Figure 24: Decision tree function layout [83]

In a RF, several such trees are used to come to a conclusion that has a higher possibility of being accurate. Thus, defining from an ensemble viewpoint, each such DT forms a weak learner since they are incapable of considering all the possible conditions. However, if a large number of such trees are used with combination of decision nodes which are mutually exclusive to each other, then the total group becomes a strong learner and is better able to fit data. Usually, in a random forest, each decision tree block consists of a random collection of decision nodes and these combinations may or may not be repeated. Figure 25 represents a block diagram conceptually representing a random forest. Each decision node is known as a classifier. When running RF for a given data, the data points are divided into 1) a training set and 2) a test set. The classifiers are trained using data from the training set and then applied on the test set to find out their accuracy. The classifier testing is done under a four-fold validation process, namely, cross-validation, thresholding, mean

precision and precision above chance. This is done to find out how a classifier predict click through rates (CTR) for the test data set which is unfamiliar to it.

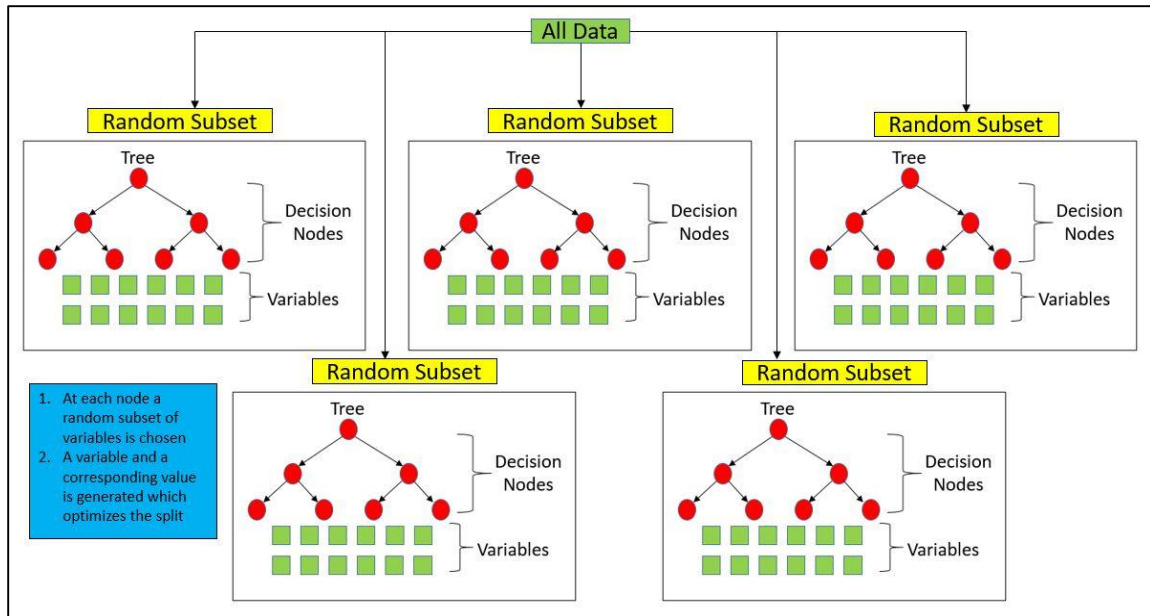


Figure 25: Random Forest formation from decision trees [83]

Success of a RF depends on the following factors [83]:

1. With a large number of predictors, the eligible predictor set will be quite different from node to node.
2. The greater the inter-tree correlation, the greater the random forest error rate, so one pressure on the model is to have the trees as un-correlated as possible.
3. As number of predictor variables reduce, the inter-tree correlation and the strength of individual trees reduce as well. So, optimal value of such variables must be set.

Random forest runtimes are quite fast, and they are able to deal with unbalanced and missing data. Random Forest weaknesses are that when used for regression they cannot predict beyond the range in the training data, and that they may over-fit data sets that are particularly noisy. A pseudocode for RF can has been presented in Figure 26.

```

Random Forest ( Input Set  $X(i,j)$ , No. of data points per tree  $n$ , Predictor Set  $s$ , No. of predictors per tree  $m$ )
  Sample  $n$  cases at random, with replacement to create subset of data  $n \in X$ 
  The subset should be about 66% of the total set
  At each node,  $m$  predictor variables are selected at random from  $s$ .
  foreach  $N_i$  group having  $m$  predictors out of  $s$  do
    If (  $m_k$ , where  $m_k \in m$  gives best split as per an objective function)
      Binary split forest at that node.
    end
  At next node choose another  $m$  variables at random from  $s$ .
end
end

```

Note: Depending on value of m , there will be three different cases:

1. Random Splitter for $m = 1$.
2. Breiman's bagger for $m =$ total number of variables.
3. Random Forest for $m \ll$ number of predictor variables.

There are three possible values of m , $1/2\sqrt{m}$, \sqrt{m} or $2\sqrt{m}$.

Figure 26: Pseudocode for Random Forest

4.5 Seasonal autoregressive integrated moving average (SARIMA)

Time series datasets require special conditioning for forecasting data. ARIMA and SARIMA are time series models. These are more generalized forms of auto regressive moving average (ARMA) models. These are used in order to include more realistic dynamics, in particular, respectively, non-stationarity in mean and seasonal behaviors of a time-series data. Hence, to conceptualize a SARIMA process, it is imperative that an ARMA process is discussed first [84].

ARMA process was originated out of two lines of thinking (For keeping things simple, univariate models have been discussed here). The first case scenario was, for a series x_t , it can be thought that the level of the current observation are dependent on that of its lagged values. For example, if there is a high value of gross domestic product (GDP) realization

in the present quarter, it would lead to be expected that the GDP in the next few quarters will good as well [84]. This notion can be modelled by an auto regressive (AR) model.

Thus, the auto regressive model of order 1 can be represented as follows:

$$x_t = \phi x_{t-1} + \epsilon_t \quad (25)$$

Where, $\epsilon_t \sim WN(0, \sigma_t^2)$ is a back-shift operator.

Extrapolating from the AR(1) process, an AR(p) process can be represented as follows:

$$x_t = \phi x_{t-1} + \phi x_{t-2} + \dots + \phi x_{t-p} + \epsilon_t \quad (26)$$

Therefore, p represents the order of the AR process. In a second manner of thinking, the observations of a random variable at time t is thought to not only be affected by the shock at time t, but also the shocks that have taken place before time t. For example, if a negative shock to the economy is observed due to, say, a catastrophic earthquake, then it can be expected that this negative effect will affect the economy not only for the time it takes place, but also for the near future. This kind of thinking can be represented by a moving average (MA) model. The MA(1) (moving average of order one) and MA(q) (moving average of order q) can be written as follows:

$$x_t = \epsilon_t + \theta_{\epsilon_{t-1}} \quad (27)$$

and

$$x_t = \epsilon_t + \theta_{1\epsilon_{t-1}} + \dots + \theta_{q\epsilon_{t-q}} \quad (28)$$

For order q.

On combination of the AR and the MA model in one representative equation, the resulting process becomes an ARMA(p,q) model. The resulting equation can be represented as follows:

$$x_t = \phi x_{t-1} + \phi x_{t-p} + \dots + \phi x_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \dots + \theta_q \epsilon_{t-q} \quad (29)$$

ARMA model provides one of the basic tools in time series modeling.

The following properties of ARMA provides useful insight to draw inference from an univariate model:

1. Lag Operators

These help in representing ARMA in a concise format. While applying a lag operator L once, the index is moved back by a single time unit, and applying it k times, moves the index back by k units [84]. The process can be represented as follows:

$$Lx_t = x_{t-1} \quad (30)$$

$$L^2x_t = x_{t-2} \quad (31)$$

.....

$$L^kx_t = x_{t-k} \quad (32)$$

The lag operator has distributive properties over addition, that is:

$$L(x_t + y_t) = x_{t-1} + y_{t-1} \quad (33)$$

Using lag operators, the ARMA model can be rewritten as follows:

$$AR(1): (1 - \phi L)x_t = \epsilon_t \quad (34)$$

$$AR(p): (1 - \phi_1 L - \phi_2 L^2 - \dots - \phi_p L^p)x_t = \epsilon_t \quad (35)$$

$$MA(1): x_t = (1 + \theta L)\epsilon_t \quad (36)$$

$$MA(q): x_t = (1 + \theta_1 L + \theta_2 L^2 + \dots + \theta_q L^q)\epsilon_t \quad (37)$$

Putting $\phi_0 = 1$ and $\theta_0 = 1$, the lag polynomials can be defined as:

$$\phi(L) = 1 - \phi_1 L - \phi_2 L^2 - \dots - \phi_p L^p \quad (38)$$

$$\theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \dots + \theta_q L^q \quad (39)$$

Finally, with lag polynomials, ARMA can be rewritten in a more compact way:

$$AR: \phi(L)x_t = \epsilon_t \quad (40)$$

$$MA: x_t = \theta(L)\epsilon_t \quad (41)$$

$$ARMA: \phi(L)x_t = \theta(L)\epsilon_t \quad (42)$$

2. Invertibility

A time series probability model can be represented in multiple ways. The representation to choose depends on the problem at hand. For example, to study impulse-response function, MA representations maybe more convenient; while to estimate an ARMA model, AR representations maybe more convenient as usually x_t is observable while ϵ_t is not. Although it is an important property, not all ARMA processes can be inverted. The following equations show how an AR(1) process can be represented by an MA(∞) process through invertibility with the use of an inversion operator $(1 - \phi L)^{-1}$, such that $(1 - \phi L)^{-1}(1 - \phi L) = 1$.

An AR(1) process, multiplied by the inversion operator can be represented as [84]:

$$x_t = (1 - \phi L)^{-1}\epsilon_t \quad (43)$$

Now, if $\theta_k = \phi^k$, for $|\phi| < 1$, then it can be shown that

$$AR(1): (1 - \phi L)x_t = \epsilon_t \quad (44)$$

And,

$$MA(\infty): x_t = \theta(L)\epsilon_t \quad (45)$$

3. Auto-covariance functions and stationarity of ARMA models

For an MA(1) process, the first two moments can be calculated as follows:

$$x_t = \epsilon_t + \theta_{\epsilon_{t-1}} \quad (46)$$

Where, $\epsilon_t \sim WN(0, \sigma_t^2)$, is a back-shift operator. The first two moments are:

$$E(x_t) = E(\epsilon_t + \theta_{\epsilon_{t-1}}) = 0 \quad (47)$$

$$E(x_t^2) = E(1 + \theta^2)\sigma_t^2 \quad (48)$$

And,

$$\gamma_x(t, t+h) = E[(\epsilon_t + \theta_{\epsilon_{t-1}})(\epsilon_{t+h} + \theta_{\epsilon_{t+h-1}})] \quad (49)$$

$$\gamma_x(t, t+h) = \begin{cases} \theta\sigma_\epsilon^2 & \text{for } h = 1 \\ 0 & \text{for } h > 1 \end{cases} \quad (50)$$

So, for an MA(1) process, the mean is fixed and the co-variance function does not depend on time. Hence, MA (1) is a stationary process [84]. This notion can be generalized for an MA(q) process and extended to an MA(∞) process as well, provided the co-efficients are absolutely summable, which basically means that the integral function of the squares of their absolute values is finite. Similarly, an AR(1) process is also stationary, since AR(1) = MA(∞) through invertibility. This can again be generalized to an AR(p) process, where the inverted co-efficients need to be absolutely summable. Combining the AR and MA processes and considering that it holds its invertible property, it can presented that [84]

$$\phi(L)x_t = \theta(L)\epsilon_t \quad (51)$$

Inverting $\phi(L)$ it can be seen that

$$x_t = \phi(L)^{-1}\theta(L)\epsilon_t = \psi(L)\epsilon_t \quad (52)$$

Where, ψ can be calculated recursively by equating the co-efficients of the lag operator L^k .

Therefore, an ARMA(p,q) process is stationary as long as $\phi(L)$ is invertible [84]. This reduces the entire ARMA process to its AR component and is independent of the moving average part, assuming all the parameters are finite. This basically proves that in order to apply ARMA to a time series function, the function first needs to be broken down to its stationary component. Only then can ARMA be applied.

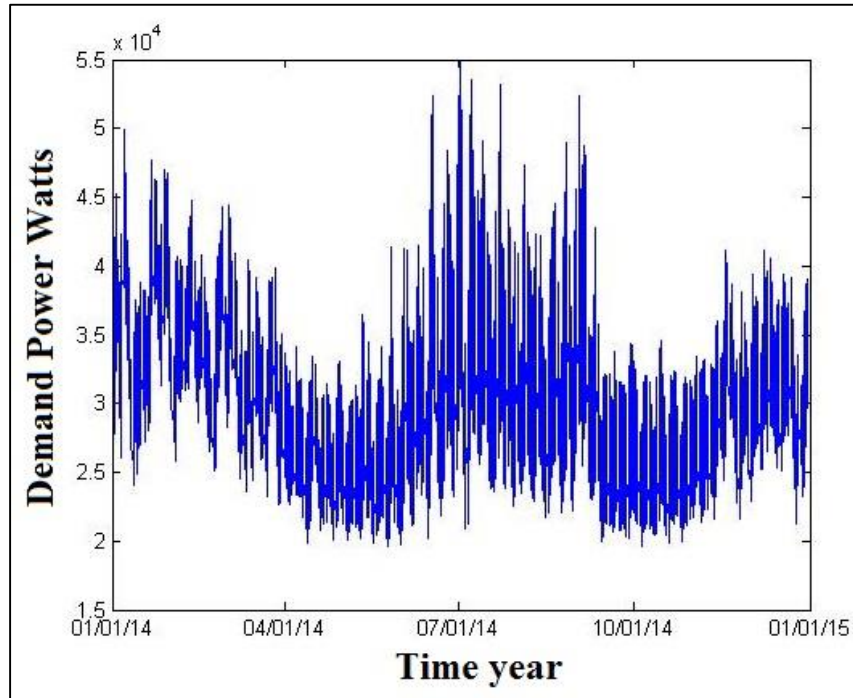


Figure 27: Demand power plot for the year 2014 Mid-Atlantic region

The power demand data obtained from PJM Corporation website for the Mid-Atlantic region, was plotted against time. Figure 27 shows the data plotted over a period of the 2014 year, whereas, Figure 28 shows the plot for a period of 48 hours between 06.25.2014 and 06.26.2014. It is evident that the data is seasonal for 12 months and do show a periodicity of 24 hours with a non-stationary mean. A visual inspection of the curves thus prove that application of ARMA process to the time series is infeasible, since it is not capable to deal with seasonality in data or non stationary data. So, in this research, application of ARIMA and SARIMA processes was necessary.

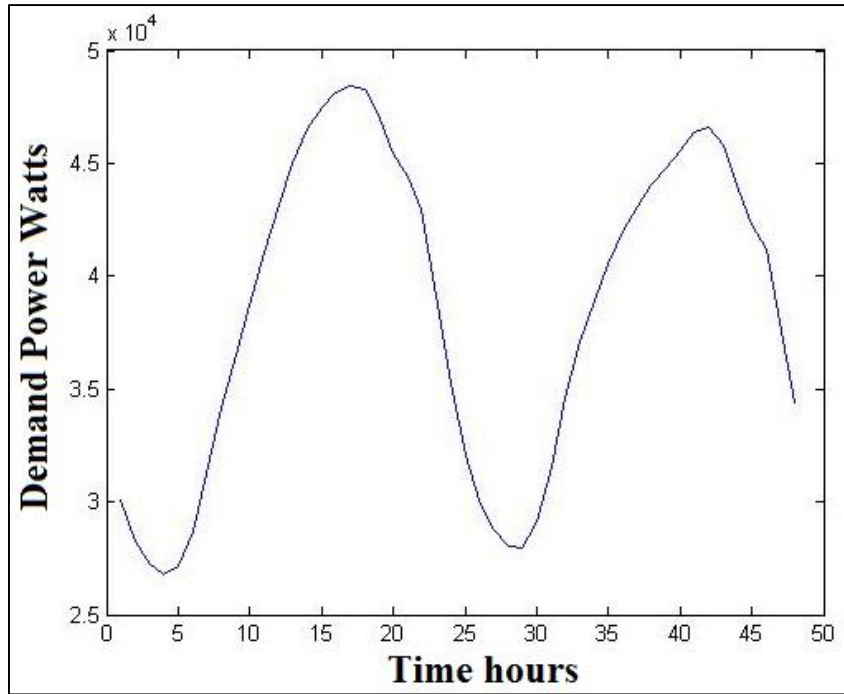


Figure 28: Demand power plot between 06/25/2014 and 06/26/2014 for Mid-Atlantic region

ARIMA and SARIMA models are extensions of ARMA class in order to include more realistic dynamics, in particular, respectively, non-stationarity in mean and seasonal behavior [85]. In practice, many load time series are nonstationary in mean as evident from Figure 27 and Figure 28. These can be modelled as an ARMA model only by removing the nonstationary source of variation. This is done by differencing the series. For example, if X_t is a time-series data which is non-stationary in mean, then an ARMA model can be built on a series, w_t , where

$$w_t = \Delta^d X_t \quad (53)$$

Where, Δ^d is the differencing function. So, an ARIMA can be represented as ARMA process defined on the d^{th} difference of the original process [85]. It can be shown in the following equation:

$$\phi(L)\Delta^d X_t = \theta(L)\epsilon_t \quad (54)$$

Where, $\phi(L)$ is the AR component and $\Delta^d X_t$ is a series made stationary through differentiation and can be modelled through ARMA. Characteristics of an ARIMA process are as follows [85]:

1. When $d = 0$, it is a stationary process. For $d = 1$, it is a nonstationary process. The level changes in time, but the increase is constant, that is, the level is nonstationary, but its increments are. For $d = 2$, it is a nonstationary process, that is, both level and increments are non-stationary.
2. When X_t is nonstationary its theoretical ACF is not defined (only the empirical ACF is). However, by observing the behavior of processes that are nearly stationary the following points can be assumed.
 - a) The auto-correlation function (ACF) decreases extremely slowly to zero, the decrease is not exponential by linear.
 - b) The partial ACF (PACF) takes value 1 for lag $k = 1$ and zero elsewhere. These characteristics of ACF and PACF are motivated by the dominance of the trend on the other dynamics in the series. Unless the trend is removed, nothing else (e.g. other MA or AR components) can be recognized from ACF and PACF.

SARIMA is a generalized version of ARIMA, applied to a series which has seasonal characteristics along with being non-stationary in mean. Seasonality refers to the property of a time-series data, where similar patterns are repeated after a known period of time. For example, seasonality of 12 months means that similar patterns will be found in the time-series data set after every 12 months. A SARIMA process can be modelled as [85]:

$$\phi(L)\Delta^d X_t = \theta(L)\epsilon_t \quad (55)$$

Where, ϵ_t is such that,

$$\phi_s(L^s)\Delta_s^D \epsilon_t = \theta_s(L^s)e_t \quad (56)$$

So, the ARIMA equation can be re-written as:

$$\phi(L)_s\phi(L^s)\Delta_s^D \Delta^d X_t = \theta(L)_s\theta(L^s)\epsilon_t \quad (57)$$

The above equation can be represented as $X_t \sim \text{ARIMA}(p,d,q) \times (\text{P,D,Q})_s$. What it means is that SARIMA is basically ARIMA(p,d,q) models whose residuals ϵ_t are again ARIMA(P,D,Q). In the equation, the operators defined by L^s and its successive powers represent ARIMA(P,D,Q) [85].

4.6 Developing a hybrid method

From an initial literature study on the topic of load forecasting, it was evident that a major portion of the total variation in electricity load time series can be attributed to the strong periodic behavior in the data [70]. Meteorological factors, such as temperature, humidity, and wind speed, are important sources of variation in electricity load. Among these meteorological variables, temperature was found to be the most important in many studies [86]. It is also observed that the temperature-load relationship is highly non-linear [87]. The research work investigated this relationship and found that it can be approximated by an asymmetric V-shaped function with a minimum at around 65°F. This minimum represents the transition point between the needs for heating and cooling. However, it was later assumed that a better representation could be made if the relationship was approximated to be U-shaped function because there is generally a range of temperature when neither heating nor cooling is needed [88]. Consequently, this approach requires two

transition points to be identified. One common method of practice is to transform temperature into degree-days or degree-hours based on the transition points and use piecewise regression to model the temperature effect [86] and [88]-[89]. It is also found that the hourly temperature–load relationship may be affected by factors such as time of day, day of the week, season, location, income, price, holidays, and so on. Some researchers model the temperature–load relationship individually for different intraday period and day type [90]-[91].

To capture the influences of as many factors influencing the temperature-load relationship as possible, this research has used a combination of random forest and SARIMA forecasting methods. The forecasting method implemented for STLF and MTLF is a two-step process where the available power demand data has been represented as a dependent variable governed by controlling factors like type of day (holiday or business day), average temperature for the day, hour of the day, day of the week, month of the year and season. The hourly power demand data for Mid-Atlantic region was obtained from PJM website [92]. The daily temperature data for the entire region was obtained from the National Centers for Environmental Information website [93]. This temperature was later averaged over all the states consisting of the Mid-Atlantic region (New York, Pennsylvania, New Jersey, Delaware, Maryland, Virginia and West Virginia) [94]. The entire process of forecasting is divided into 3 stages. They are described as follows:

1. Data collection from utilities

PJM Interconnection LLC is a regional transmission organization (RTO) in the United States, with the headquarters being at Valley Forge, Pennsylvania. It is part of the Eastern Interconnection grid operating an electric transmission system,

serving all or parts of Delaware, Illinois, Indiana, Kentucky, Maryland, Michigan, New Jersey, North Carolina, Ohio, Pennsylvania, Tennessee, Virginia, West Virginia and District of Columbia. PJM coordinates the movement of wholesale electricity in all or parts of 13 states and the District of Columbia [75]. Hourly Load data is available from the PJM website in an ‘.xlsx’ format. In order to create a historical database to run the forecasting algorithms, the files were downloaded from the website and stored. The data archive in PJM has metered load data dating back to as early as ten years old. However, since the research focusses on STLF and MTLF, only the data from the past four years was used. This was done to capture the trend of power usage in the recent history, in a bid to create a more accurate model. Each ‘.xlsx’ file consists of load data from 28 regions. However, for this research, data from the Mid-Atlantic region was considered only. The downloaded data was then integrated into a single ‘.csv’ file for all years starting from the year 2012 till the most recent available date of year 2015. This process was executed in a matlab environment.

The temperature data for the corresponding period was collected from the National Centers for Environmental Information (NOAA) database. The National Oceanic and Atmospheric Administration is an American scientific agency within the United States Department of Commerce focused on the conditions of the oceans and the atmosphere. NOAA warns of dangerous weather, charts seas and skies, guides the use and protection of ocean and coastal resources, and conducts research to improve understanding and stewardship of the environment [93]. The temperature data was acquired in a ‘.txt’ format from all weather stations

present in the seven states comprising the Mid-Atlantic region. The temperature readings for the day was then averaged over all the stations to get a single temperature value representing the entire region for each day. This was done using the matlab environment. The resulting, approximated temperatures were then integrated for the entire period from year 2012 to the latest date and stored in a ‘.csv’ file.

2. Data preprocessing

A user interface was developed in this stage in the matlab environment, where, the user has a flexibility to choose what part of the data (both temperature and power demand) should be allocated for a training dataset. This dataset serves as a historical database required for building the RF forecasting model. In this stage, the data was also divided on the basis of the type of day (business day or holiday). The interface developed allows the user to choose the hour of the day for which the forecasting model is to be built. Based on this time, the rest of the forecasting process will be carried out for future days’ load forecasting. New variables were created which indicates properties of the data which have been presented in the Table 3.

Table 3: Sample dataset for all business days at 2am

Date	DemandP	Season	Month	Day_of_Week	Temperature
1/3/2012	28128.519	3	1	3	26.676
1/4/2012	33965.268	3	1	4	20.329
1/5/2012	30738.251	3	1	5	33.214
1/6/2012	29455.599	3	1	6	41.045
1/9/2012	27201.48	3	1	2	34.048
1/10/2012	27602.549	3	1	3	38.279

Table 3 shows demand data (in Watts) represented in a chronological format along with corresponding temperatures in Fahrenheit (F) scale. Columns season, month and day of week are indicator variables, describing properties of observation like

the season of the year, the month of the year and the day of the week, respectively. This type of representation of a variable is called a categorical variable, where, the values of the variable are not treated as factors, but labels that indicate certain information about the data which otherwise cannot be quantified. For example, the variable ‘Day of Week’ can have 7 levels, each level representing each day of a week. Table 5, Table 6 and Table 7 represent the variables with their possible levels. Hence, the dataset in Table 3 has three categorical variables and one quantitative variable.

Table 4 shows the name of the variable and its type. Together the categorical variables and the quantitative variable form the set of predictor variables, which are used in modelling the random forest process (the dependent variable).

Table 4: Name of predictor variables with their types

Variable Name	Variable Type
Season	Categorical
Month	Categorical
Day of Week	Categorical
Temperature	Quantitative

Table 5: Possible level and label names for variable Month

Month	
Label Name	Level Number
January	1
February	2
March	3
April	4
May	5
June	6
July	7
August	8
September	9
October	10
November	11
December	12

Table 6: Possible level and label names for variable Season

Season	
Label Name	Level Number
Summer (Jun, Jul, Aug)	1
Fall (Sept, Oct, Nov)	2
Winter (Dec, Jan, Feb)	3
Spring (Mar, Apr, May)	4

Table 7: Possible level and label names for variable Day of Week

Day of Week	
Label Name	Level Number
Sunday	1
Monday	2
Tuesday	3
Wednesday	4
Thursday	5
Friday	6
Saturday	7

With the database getting created, the forecasting moves to the third and final stage of processing.

3. Building forecasting model to predict future load values

This is a two-step process, which were executed in R programming environment. R is a programming specifically used to execute statistical experiments, hence the choice of the environment. A) In the first step, the dataset generated in the data preprocessing stage is used to build a RF model. For this, a pre-built software package called ‘randomForest’ was used. The function used to execute the modelling is “fit <- randomForest(DemandP ~ season + month + dow + Temperature, data = dat[1:r1,],replace = TRUE, importance=TRUE, ntree=2000,mtry = 2)” [95]. A random forest object, ‘fit’ is created which consists of the software model of RF pertaining to the input arguments. Here, ‘DemandP’ is the dependent variable, having the load information. ‘Season’, ‘month’, ‘dow’ and

‘Temperature’ are the predictor variables. The randomForest function has built in provisions to work with categorical variables. Table 8 gives a complete idea of the input arguments and their properties in the randomForest function.

Table 8: Input arguments and their properties for randomForest

randomForest	
Input Argument	Property
DemandP	Dependent Variable
Season	Independent Variable (Categorical)
Month	Independent Variable (Categorical)
Dow	Independent Variable (Categorical)
Temperature	Independent Variable (Quantitative)
Data	Dataset to be used to train the model
Replace	Set replacement of case samples from dataset
Importance	Set importance of predictors assessment
Ntrees	Number of trees in the RF model
Mtry	Number of predictors sampled for splitting at each node.

The ‘fit’ model is then used to predict the power demand for all the days in the train dataset. It should be noted here that the prediction is done for the same dataset that was used to train the RF model. The predicted values are then differenced from the actual values to create a residual dataset. Table 9 illustrates how the residuals are created from the actual datasets.

Table 9: Calculating residuals from RF model

Residuals from Random Forest Model			
Date	Actual Data (PJM) at 2am	Predicted Data (RF) at 2am	Residuals (PJM-RF)
1/3/2012	43241.29	43064.02	222.95
1/4/2012	42545.21	43575.10	-1048.93
1/5/2012	39572.38	39606.04	-91.84

B) In the second step, the residuals are used as a time-series input to a SARIMA model. SARIMA is available as an R package, which can be used by calling the function "mod_fit <- sarima(dat_ts,p,d,q,P,D,Q,S)". Here, the ‘dat_ts’ corresponds to the residual values and ‘p,d,q,P,D,Q’ values are the parameters to indicate which

model we are using in SARIMA and S is the seasonality in the dataset. For example, `sarima(dat_ts, 1,0,1,1,1,1,24)` means An ARMA process is being used inside an ARIMA model with a seasonality of 24 hours. The `sarima` function creates the ‘`mod_fit`’ object, which consists of the software model of SARIMA. This model is then used to predict residual values for the rest of the days in the month (business days or holidays, depending on user input). The `randomForest` model created in the first step is used here to get load forecasts for the rest of the month as well. Finally, the forecasted loads from RF is added to the predicted residuals from `sarima` to get a final, adjusted load forecast. This data is compared against actual PJM data set for the corresponding dates to calculate errors. This concludes the MTLF. The results have been discussed in chapter 4.

The same setup can be used for STLF, where the first predicted value will be the predicted load for the next day. Table 10 shows the final predicted results.

Table 10: RFSRIMA Predicted values for load data at 2 am

Random Forest + SARIMA Forecasting					
Date	RF Prediction	SARIMA Prediction	Adjusted Result (RFSRIMA)	Actual Data (PJM)	% Error (PJM-RFSRIMA)
1/2/2015	27379.52	-181.308	27198.21	27422.008	0.816
1/3/2015	27338.02	-237.799	27100.23	24633.998	-10.011
1/4/2015	31405.59	-99.406	31306.19	31695.589	1.23

Table 10 shows the results obtained from hybrid RF and SARIMA forecasts. Here, the dataset used to train the RF model is expected to be present till the day before 1/1/2015 for best results. If we consider a scenario that the latest recorded data from a power system is available till the 31st of December, 2014. Then, the predicted result for 2nd January, 2015 will be an hourly forecasted value, thus satisfying the STLF properties. This process can be repeated for all the 24 hours of day, thus

resulting in a day-ahead load forecasting scheme. Figure 29 shows a block diagram for the entire process of forecasting. All results obtained from the STLF and MTLF have been presented and discussed in chapter 4. All codes for the RF+SARIMA hybrid process is presented in Appendix B.

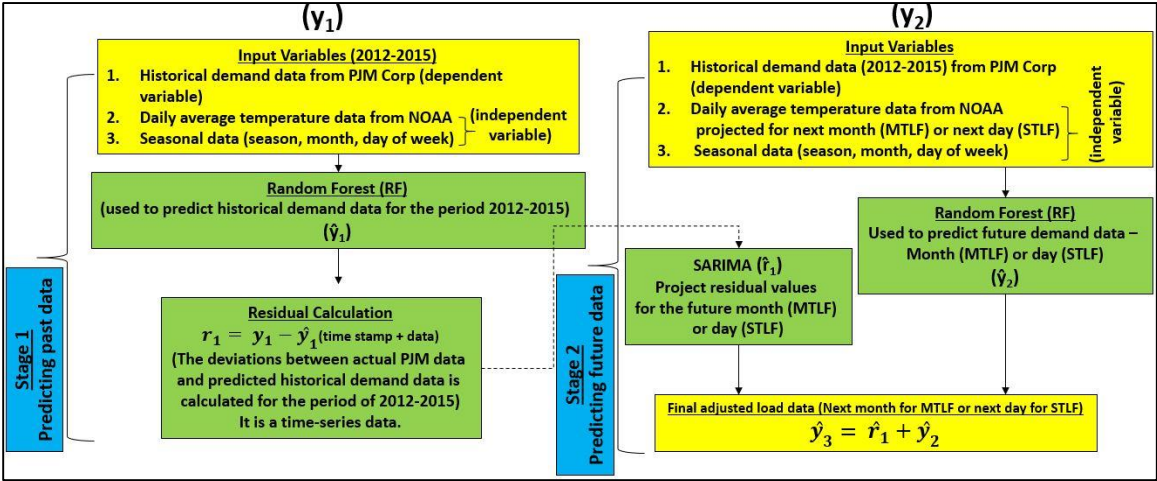


Figure 29: Functional block diagram of RF + SARIMA (RFSRIMA) process

V. RESULTS AND DISCUSSIONS

A comparative analysis and investigation on various clustering and forecast models has been discussed in this chapter. This chapter is organized into two sections. The first section presents the results obtained from clustering algorithms (DBSCAN, k-means and multi-tier k-means). The second section presents a discussion on the results obtained from (random forest, SARIMA, hybrid method including LOWESS and SARIMA model) and a novel hybrid model (which includes random forest and SARIMA models).

5.1 Results obtained from data clustering algorithms

Four different test cases have been formulated to evaluate the performance of existing clustering methods against multi-tier k-means. The cases under consideration are the following: 1) Ideal or Normal Load condition, 2) Heavy Load condition, 3) Light Load condition and 4) Single Line-to-ground Fault condition. The methods evaluated are, 1) DBSCAN, 2) k-means and 3) multi-tier k-means. The performance of the clustering algorithms have been evaluated using dunn's index (DI). This is a standard metric to evaluate cluster separation [96].

The parameters studied for clustering methods are voltage and current (magnitude and phase angle) and the frequency values from the openPDC software suite. It showcases the level of accuracy of the three algorithms during fault conditions. The four test cases are conducted for time-intervals of 5, 10, 15 and 20 minutes. The data include voltage, current

and frequency data from the synchrophasor sensors collected at a rate of 30 samples per second [98].

5.1.1 Test case 1: System under steady state (normal) conditions

Under normal conditions, the power system conditions are generally stable and its parameters lie within acceptable threshold values. Figure 30 shows clustering of voltage data from openPDC for a 15 minute time period using DBSCAN. Figure 31 indicates the clustered voltage data from the proposed multi-tier k-means algorithms and Figure 32 shows the clusters from k-means algorithm which is initialized to with a total of 3 clusters. These clusters are formed based on the value of the data points and they do not conform to any power system thresholds. Hence, we will see later in the chapter that although they have a desirable DI value, the clustering scheme fails to correctly classify the phasor data, in a manner that would be helpful for system operators to monitor the system. DBSCAN cluster the data into three clusters as core, border and noise as shown in Figure 30. The core points are data points that are equal or nearer to the threshold value of ideal transmission line voltage (300 kv). They are encoded in green color. Values that are slightly away from nominal value of core points are defined as border points. Any further isolated points are classified as noise points. From power systems perspective, this is a miss classification, since the voltage data captured is still within the allowable range of $\pm 5\%$ of ideal voltage value for the range of voltage considered [97]. Figure 31 represents the voltage data clustered by multi-tier k-means algorithm. The only input that is needed to be provided by the user is to initialize the value for ideal transmission line parameter, i.e. voltage value of 300 kv. It is evident from Figure 31 that the voltage data gets correctly clustered into core points (green) as the system is in steady state condition. Based on its

initial centroid values, the k-means algorithm continues to cluster all data points, where the number of clusters were set to 3. This is done with the idea that the power data can have three possible clusters, the core group representing normal data, the border group consisting of voltage values just outside the threshold values and finally the noise group consisting of voltage data having values further away from threshold values of border and core. Figure 32 shows the clustering output from k-means algorithm.

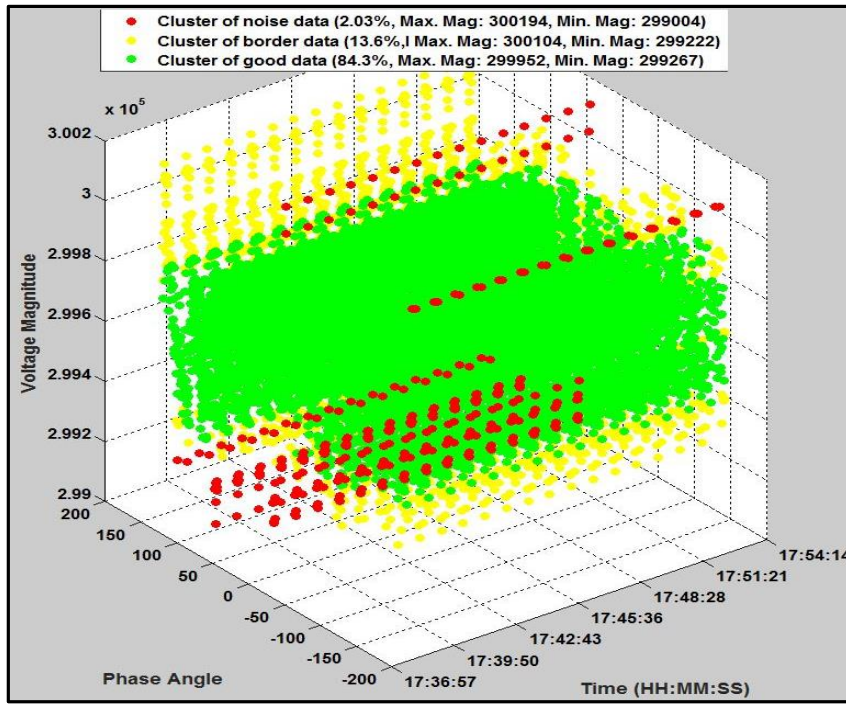


Figure 30: Steady State Voltage Output – DBSCAN

5.1.2 Test case 2: System under heavy load conditions

During the heavy loaded condition, the demand increases and thus voltage level starts to drop. Figures 33, 34 and 35 represent the clustering outputs from DBSCAN, multi-tier k-means and k-means algorithms. In Figure 33, operators can easily identify the densely populated areas (green) and the sparsely populated data points (yellow). In comparison with the Figure 30, the shift in the density of core points from 299800 volts to 299500 volts can be easily identified. Figure 34 represents the multi-tier k-means clustering of heavy

load condition. As per the enforced conditions, the data points that are within tolerable limits are clustered as good cluster (green) and other data points form a low noise cluster (blue) and the data that do not fall into either of these two clusters are grouped as an outlier region (cyan). The shift in the data points due to heavy load conditions. This means that there is a reduction in the percentage of core group points and an increase in percentage of low outlier cluster points. Finally, Figure 35 represents the output from k-means algorithm to form 3 clusters. The k-means clustering method again forms three clusters, but fails to clearly indicate the density shift and thus cannot accurately capture the heavy load conditions.

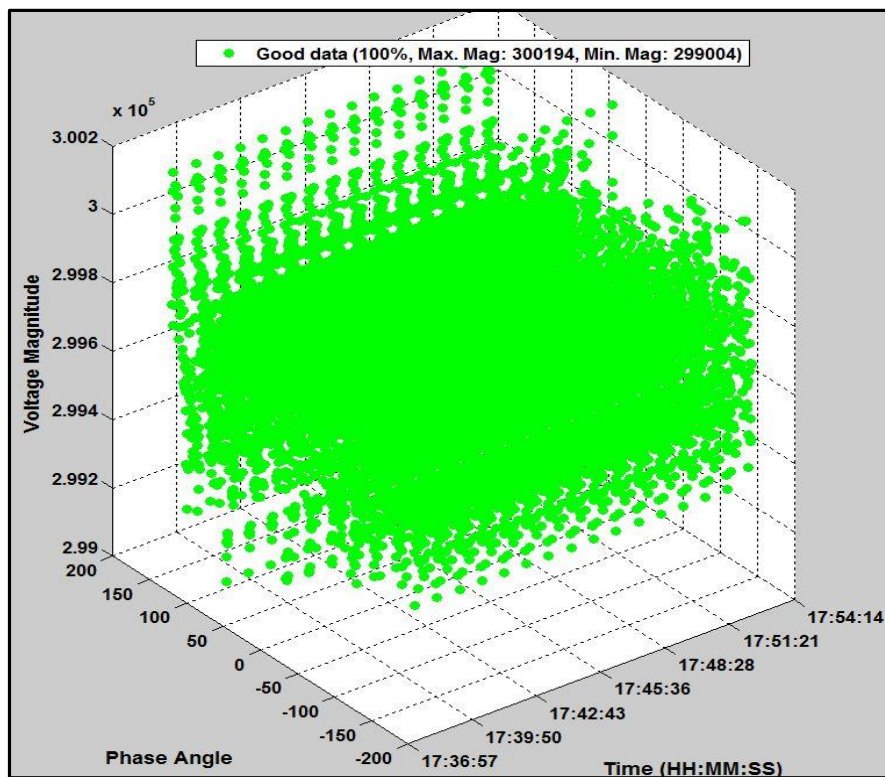


Figure 31: Steady State Voltage output – Multi-tier k-means

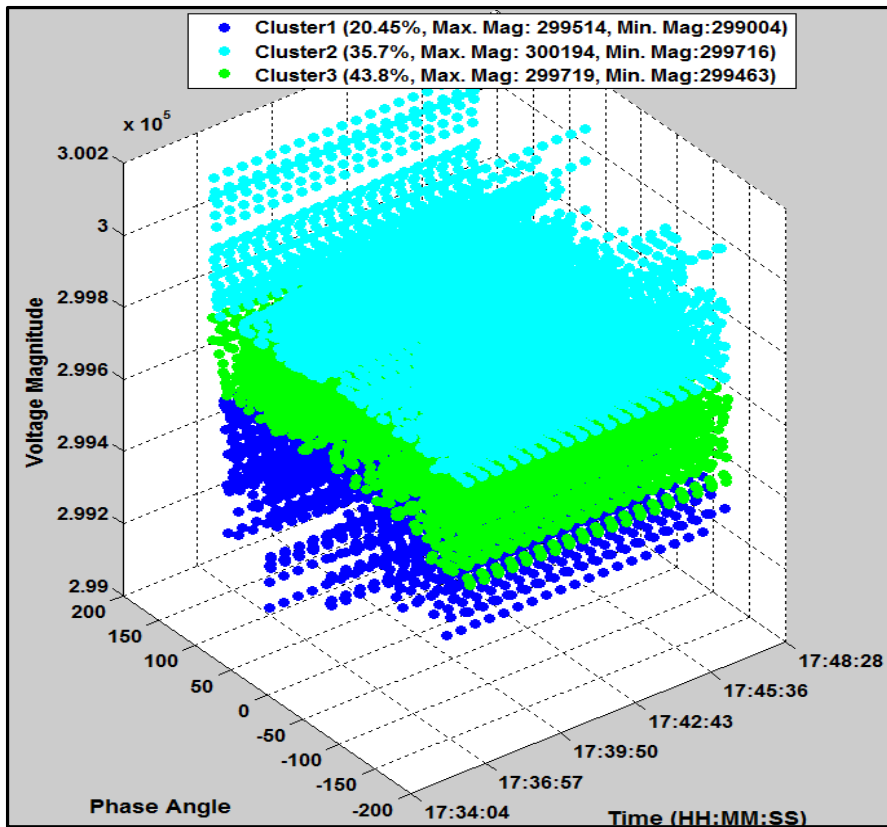


Figure 32: Steady State Voltage output – k-means

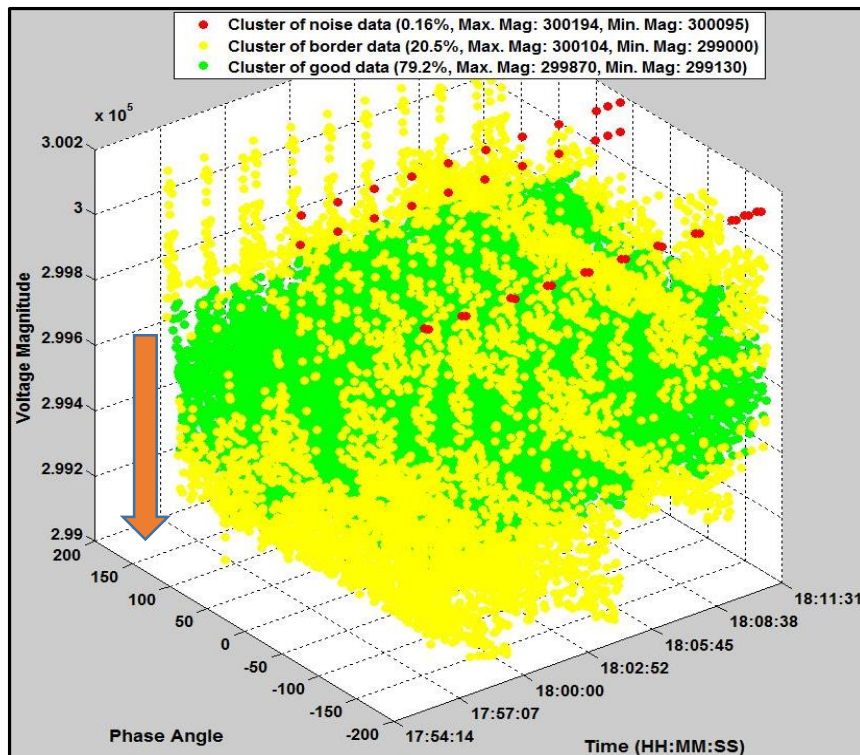


Figure 33: Heavy Load Condition – DBSCAN

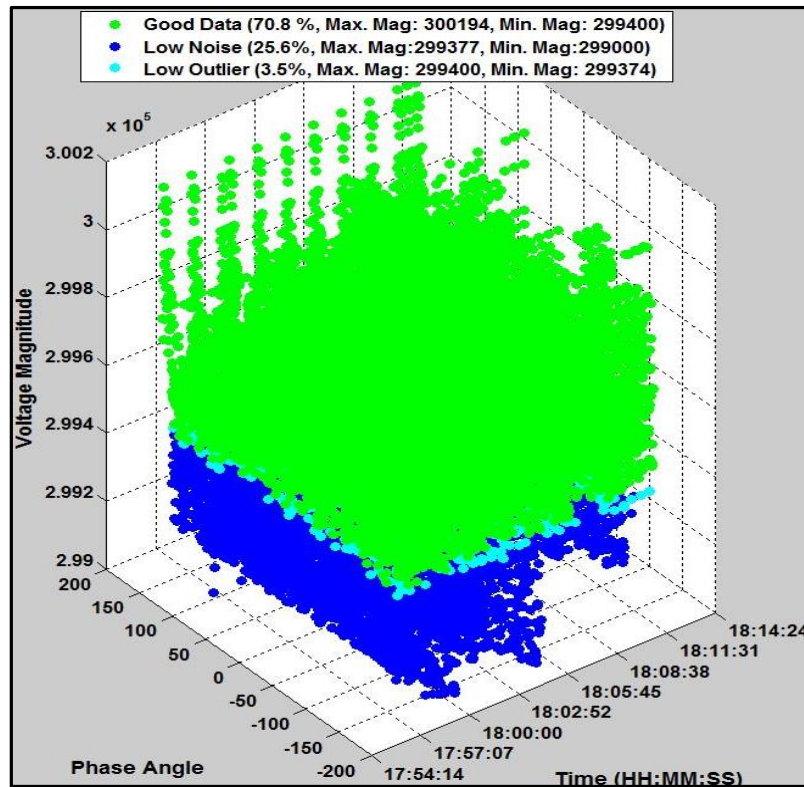


Figure 34: Heavy Load Condition – Multi-tier k-means

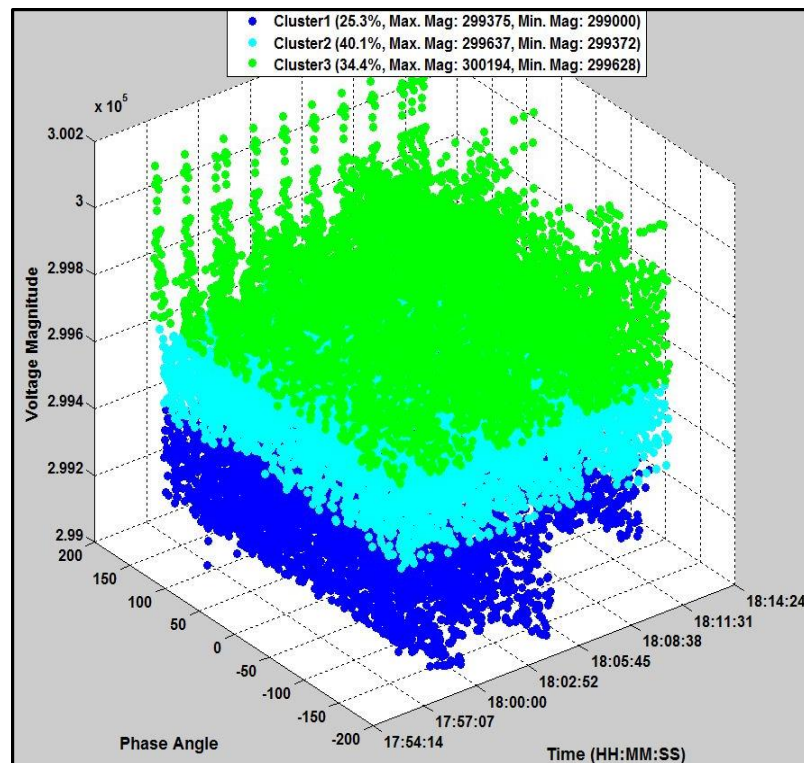


Figure 35: Heavy Load Condition - k-means

5.1.3 Test case 3: System under light load conditions

Under light load conditions, there will be a slight increase in the voltage values due to low demand. Figure 36 shows the clustered output from DBSCAN. In light load conditions, the voltage value increases to above normal. Figures 30 and 33 indicate the shift in the density of core points from 299800 volts to 300600 volts. Figure 37 shows the data clustered by multi-tier k-means under light load conditions. As per the pre-defined centroid settings, the data is clustered into four (4) clusters. They are the good data (green), the high value outliers (yellow), the low value outliers (cyan) and the high value noise data (red) clusters. The change in load conditions can again be captured by tracking the increase in percentage of high outlier cluster points in Figure 37, with respect to that of Figure 31. Figure 38 is the output from the normal k-means algorithm to form 3 clusters for the light loaded condition. It is again evident from the figure that the k-means algorithm fails to capture the light load conditions.

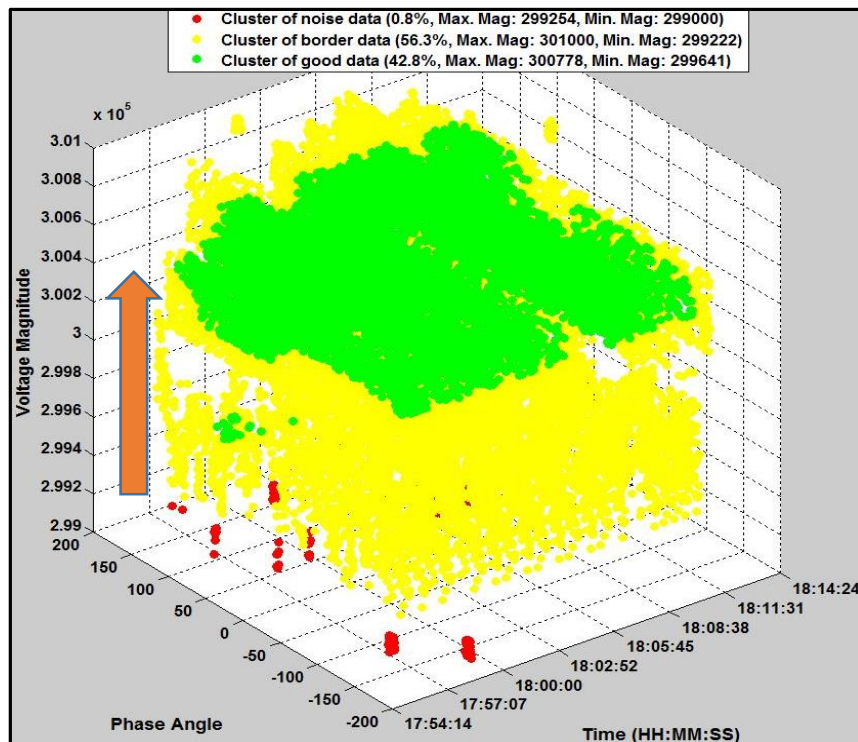


Figure 36: Light Load Condition – DBSCAN

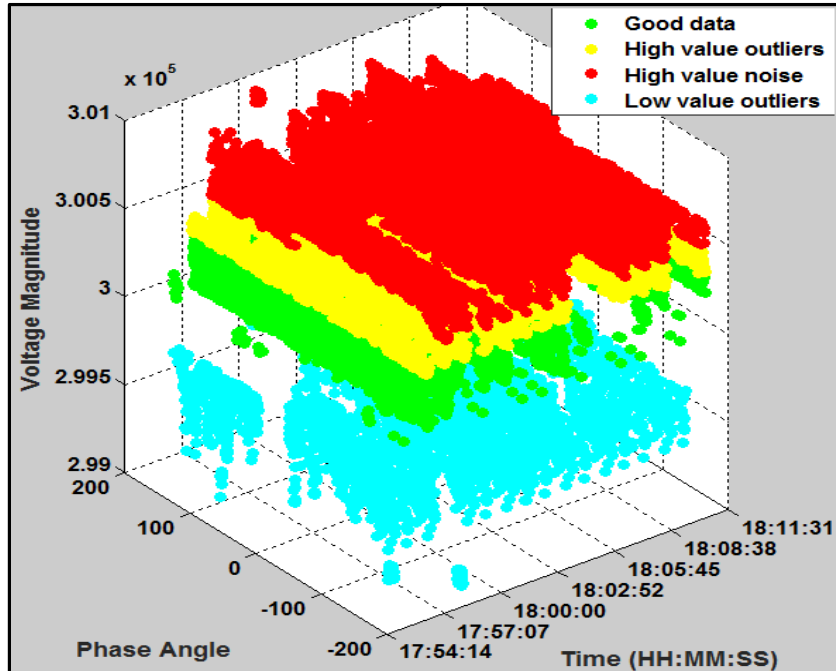


Figure 37: Light Load Condition – Multi-tier k-means

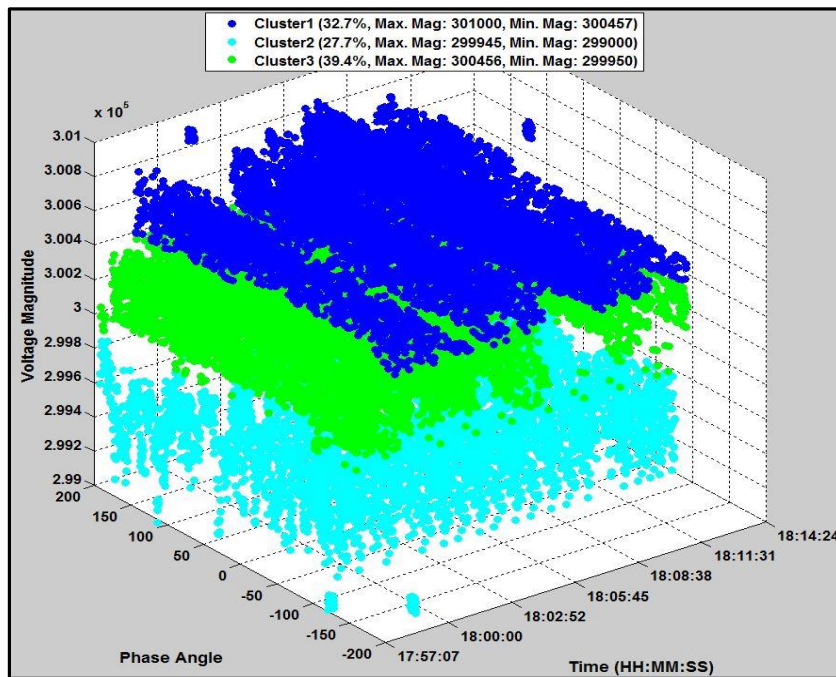


Figure 38: Light Load condition – k-means

5.1.4 Test case 4: System under single line to ground fault condition

The final test case is a single line to ground (SLG) fault condition. This fault is captured using the proposed clustering methods. Figure 39 shows the DBSCAN clustered output for

the fault condition for voltage data. The DBSCAN clusters the fault conditions into noise and clearly separates the good and faulty points. However, it fails to clearly identify the beginning of the fault, i.e., the capture the time when a contingency occurred, resulting in the drop in voltage below normal threshold. This makes it unsuitable to generate alarms. Figure 40 shows the same fault case clustered correctly using multi-tier k-means method. It does a good job in clustering the fault data into low outliers (cyan) and low noise (blue) clusters. From Figure 40 the fault condition is clearly visible when the voltage level goes below a critical value (i.e., changes over from cyan to blue). Thus, using this method, an alarm generated will alert the system operator through a pre-warning message that there is a contingency scenario (the line to ground fault). Figure 41 shows the output from k-means algorithm. It clusters the given fault condition into 3 clusters irrespective of any threshold values.

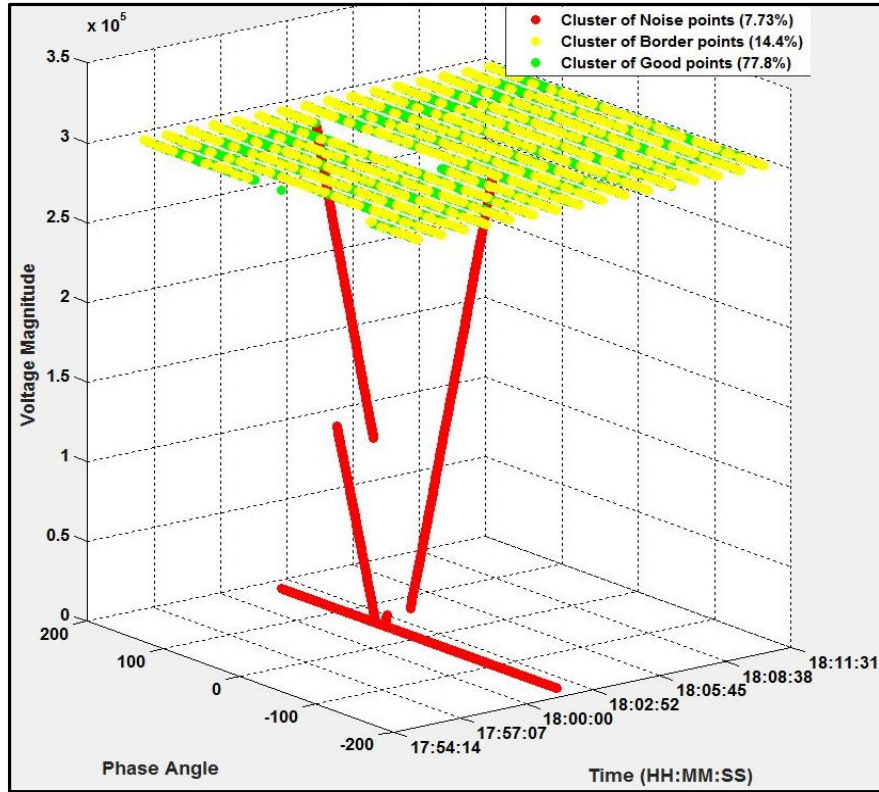


Figure 39: Line to Ground Fault – DBSCAN

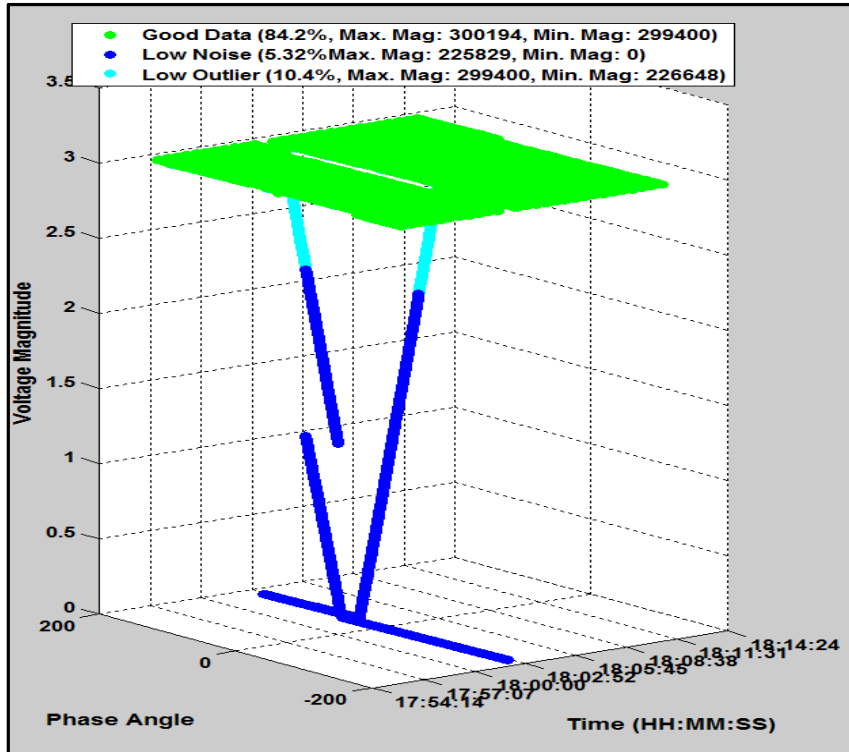


Figure 40: Line to Ground Fault – Multi-Tier k-means

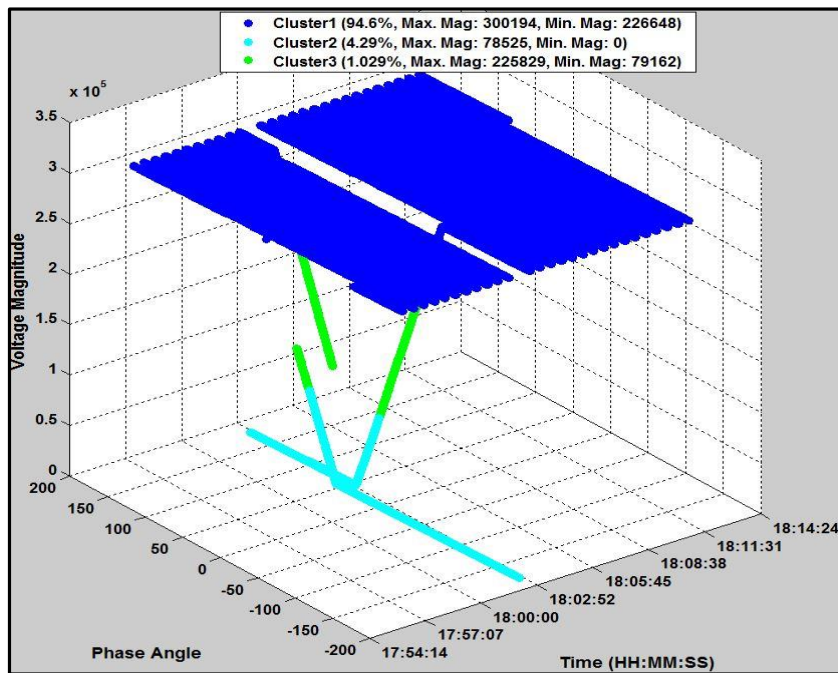


Figure 41: Line to Ground Fault – k-means

5.1.5 Test results with current (I) data

The clustering algorithms (DBSCAN, multi-tier k-means and k-means) were applied to current (magnitude and phase angle) and frequency data as well. The results obtained were analogous to the voltage dataset. Figures 42, 43 and 44 illustrate the results of these clustering methods on current parameter under normal, heavy load and light load conditions. The ideal line current value is 500 amps. The data was gathered over a period of 15 minutes.

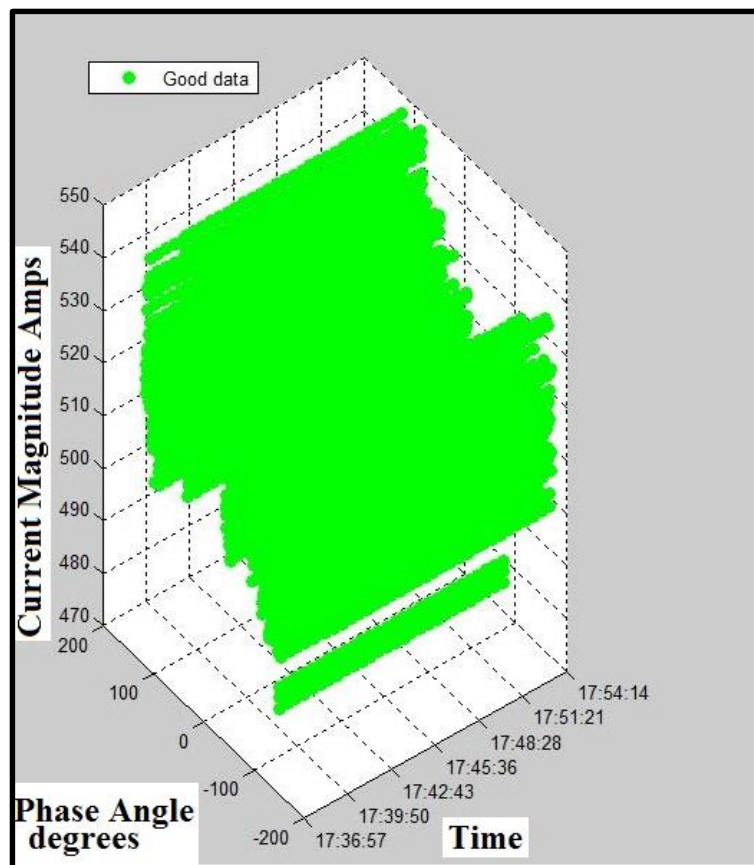


Figure 42: Steady state condition multi-tier k-means

From Figure 42, the multi-tier k-means captures accurately the normal state of the system and, thus generates only the core cluster. The percentage of good data points in the cluster is 100%.

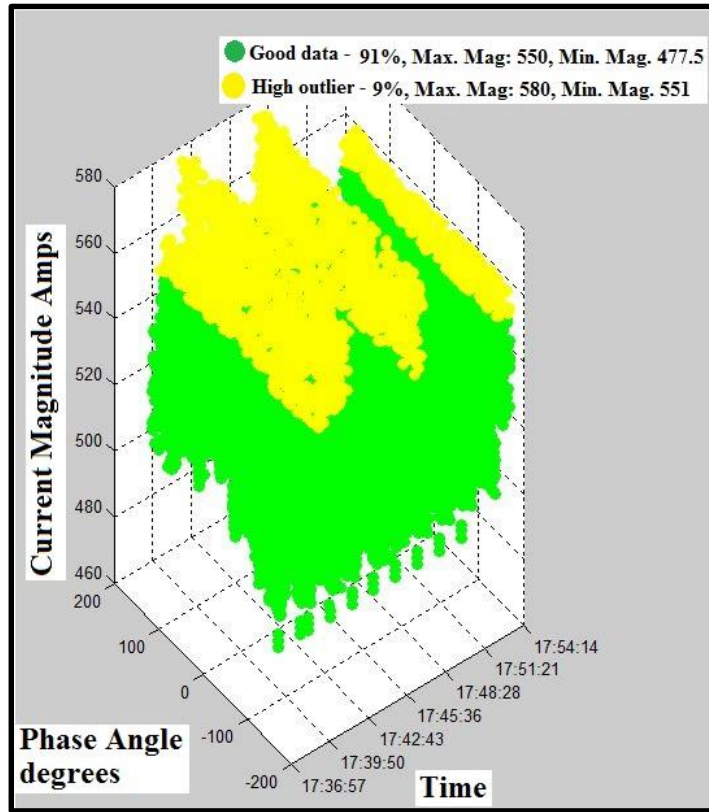


Figure 43: Heavy load condition multi-tier k-means

Figure 43 illustrates how multi-tier k-means is able to capture the heavy load condition (when the current drawn from the system starts to increase). The percentage of the core data points drops from 100% to 91% and the high outliers have 9% of the data.

Finally, Figure 44 captures the effect of light load conditions on the current data. The percentage of core points reduces and the value shifts toward the lower outliers indicating that there is less demand than the ideal conditions.

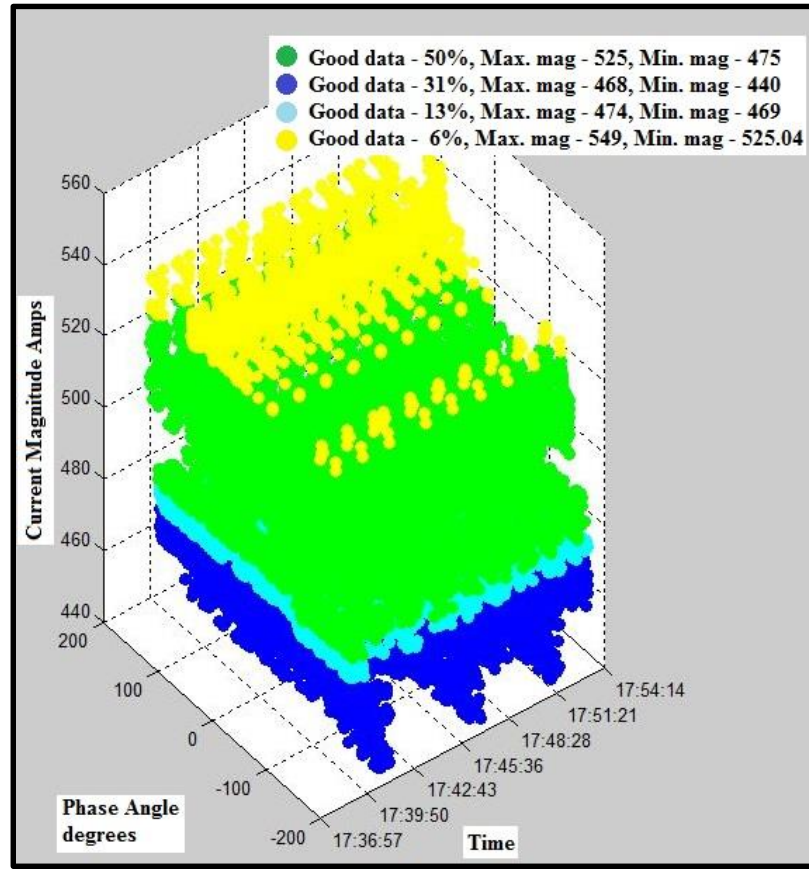


Figure 44: Heavy load condition multi-tier k-means

5.1.6 Quantification of the visual data representation

This section discusses the distribution of voltage data captured in the figures 30-44.

Under normal conditions, with different data sizes, Tables 11, 12 and 13 provide the percentage distribution of cluster points in multi-tier k-means, DBSCAN and k-means algorithms.

Table 11: Distribution of data points with multi-tier k-means

Load condition	Low Noise (Blue)	Low Outliers (Cyan)	Good Points (Green)	High Outliers (Yellow)	High Noise (Red)
Normal	0	10.53	89.47	0	0
Heavy	24.7	5.2	70.04	0	0
Light	0	3.3	79.76	16.94	0
Fault	5.32	10.4	84.2	0	0

As the conditions change from normal values, the percentage of good points decrease from 89%. In the similar way, light load condition can be observed with high border points (16%) and the heavy load condition with low noise (24%).

From the Table 12, the distribution of points is DBSCAN can be observed.

Table 12: Distribution of data points with DBSCAN

Load condition	Noise points (Red)	Border points (Yellow)	Core points (Green)
Normal	0.5	6.3	93.2
Heavy	0.078	8.96	90.5
Light	0.8	56.3	42.8
Fault	7.73	14.4	77.8

Table 13 shows the distribution of data points under different load conditions.

Table 13: Distribution of data points with k-means

Load condition	Cluster 1 (Blue)	Cluster 2 (Cyan)	Cluster 3 (Green)
Normal (100%)	27.1	36	36.7
Heavy (100%)	25.3	40.1	34.4
Light (100%)	32.7	27.7	39.4
Fault (100%)	94.6	4.29	1.02

5.1.7 Computational time

The computational time is calculated for various phases in the data mining proces. A pre-processing time is required to interface with MATLAB environment, and extract data from openPDC by C# application. Once the data is made available, the algorithms perform their operations of clustering. This run-time varies between the algorithms and with different data sizes. After the clustering process is completed, users visualize the clusters and decision samples in MATLAB environment. The time it takes to form completed clusters is referred as post-processing time. The processing times for data sizes of various time-intervals 10, 15, 20 and 30 minutes are tabulated in table 14. It is clear from table 14 that k-means is fastest when compared to multi-tier and DBSCAN methods. The proposed

multi-tier k-means does take extra seconds to cluster the data. The clustered information is very easy to visualize and accurate in multi-tier k-means than DBSCAN and k-means. However, DBSCAN does take a longer time to process the data ranging from 10's of seconds to 100's of seconds depending on the time-interval. We believe that this is due to the fact that the number of iterations in DBSCAN is higher than other algorithms.

All test cases have been conducted using the following PC configurations:

PC Configuration Tested:

1. Server Configuration: AMD FX-9590 4.7GHz 8-Core, 32 GB DDR3 RAM, 500 GB SSD
2. Software: Windows operating system, Microsoft Visual Studio 2012, openPDC 2.1, MySQL, SQLite, R, MATLAB/Simulink

From the case studies tested, each of the three algorithms has their advantages, and disadvantages. Based on the feedback from utility operator's requirement, any of these algorithms can be selected at various stages for power system planning and operations. DBSCAN can be used to provide valuable understanding on the system during heavy load conditions. The multi-tier k-means can provide insight on the grid situation as well as cluster the data to show the load intensities. Although the original k-means known to be the fastest algorithm in the literatures, it does not provide any useful information for power system data sets. Table 14 presents a break-up of the time taken to perform the experiments under various periods of collected data.

Table 14: Computation time for different data sizes

Algorithm	Time Duration (Minutes)	Pre-processing (seconds)	Processing (seconds)	Post- processing (seconds)
DBSCAN	10	68.4	8.35	13.4
	15	126	26.04	30.6
	20	227.966	56.3988	66.7
	30	478.17	132.3	146.7
Multi-Tier k-means	10	62.1	5.13	4.49
	15	123.8	8.2	6.71
	20	204.1	9.6	8.86
	30	429.9	15.2	13.1
k-means	10	55.4	0.02	2.62
	15	113.7	0.03	3.86
	20	186.7	0.09	6.19
	30	414.8	0.14	7.49

5.1.8 Visual Fidelity

A survey of 10 students were conducted to assess the visual appeal of the three clustering methods as well as their tendencies of misclassification (MC) and identify which method is effective for visualizing faults or anomalies under varying conditions. MC over here indicates the failure of a data clustering method to correctly place data points in the appropriate cluster. Table 15 indicates that proposed multi-tier k-means visually appeals better under heavy load conditions, and DBSCAN appeals good for light load conditions. A work is in progress to seek inputs on the visual fidelity of proposed clustering method from system operators.

Table 15: Visual Fidelity of clustering algorithms

Visual Fidelity under	k-means	DBSCAN	multi-tier k-means
Heavy Load condition	+ (MC ~ 50%)	-- (MC > 80%)	++ (MC < 20%)
Light Load condition	++ (MC < 20%)	-- (MC > 80%)	+ (MC ~ 50%)
Normal condition	++ (MC < 20%)	-- (MC > 80%)	+ (MC ~ 50%)
Fault condition	-- (MC > 80%)	-- (MC > 80%)	++ (MC < 20%)

5.1.9 Dunn's Index (DI)

Dunn's index is used as a metric to evaluate clustering algorithms. The aim is to identify sets of clusters that are compact, with smaller variance values between members of the cluster, and are well separated. Here, the average values of different clusters are sufficiently far apart compared to values of individual cluster variance. Dunn's index can be mathematically represented:

$$DI_m = \frac{\min_{1 \leq i \leq j \leq m} \delta(C_i, C_j)}{\max_{1 \leq k \leq m} \Delta_k} \quad (58)$$

Where, C_i and C_j are the i^{th} and j^{th} centroids of m clusters formed from a clustering algorithm and Δ_k is the distance between any two points within the k^{th} cluster. Therefore, DI calculates the ratio of the minimum inter-cluster distance to the maximum intra-cluster distance [96].

Table 16: Dunn's indices for voltage and current data

Dunn's Index	Time (mins)	DBSCAN		mutli-tier k-means		k-means	
		V	C	V	C	V	C
Heavy Load	10	0.1894	0.0462	0.2069	0.0506	0.4350	101.3
	15	0.1922	0.0365	0.3714	0.0641	0.4297	178.9
	20	0.3136	0.0529	0.2004	0.0586	0.4241	56.8
	30	0.2167	0.0523	0.1901	0.0516	0.4543	59.2
Steady State	10	0.2236	0.326	0.3016	0.816	0.4203	75.9
	15	0.323	0.0425	0.3016	0.0818	0.2968	118.2
	20	0.1936	0.2443	0.3016	0.0818	0.2836	74.6
	30	0.1536	0.0904	0.3016	0.0818	0.4203	79.7
Light Load	10	1.162	0.36	0.4035	0.04	0.4098	117.5
	15	0.1978	0.695	0.3714	0.0381	0.369	57.3
	20	1.158	0.852	0.3538	0.0372	0.405	59.9
	30	0.7375	0.09	1.1695	0.0516	0.4202	153.2

The DI calculated in each experiment was used both as a performance metric to check the compactness of the clusters and also to track load conditions. In table 16, the "time in min" column presents the period of data used for the calculations and columns "V" and "C"

indicate data calculated for voltage and current parameter respectively. It was observed that under steady state conditions, the DI for multi-tier k-means held the same value, regardless of the tested time period. This was further re-enforced in figure 31, which showed that only core points were present in the overall data. Thus, any changes in the DI value (increase or decrease) would mean the formation of new clusters, which in turn indicated that there was presence of border or noise clusters. Table 16 suggests that the DI tends to increase under light load conditions and tends to decrease under heavy load conditions, for voltage data. Using this observation as a reference, the operators can capture whether a system is in steady state or transient condition.

5.2 Results obtained from data forecasting algorithms

The load data obtained from PJM Corporation website was used for both MTLF and STLF. Four models, namely random forest, SARIMA, a hybrid model utilizing LOWESS and SARIMA (LOWRIMA) models and finally a novel, hybrid model using RF and SARIMA (RFSRIMA) models were used to forecast the demand values. Besides the historical load data, other factors, such as average daily temperature, day of the week, month of the year, type of day, etc. are also considered to build prediction models. The following section discuss the application of these models for mid-term load forecasting and short-term load forecasting. For each model, their accuracy of prediction is evaluated using statistical indices like MSE, MAE, RMSE and MAPE. A calendar year was divided into four seasons (spring, summer, fall and winter). A month was chosen at random from each season as a representative of the whole season. The rationale for this assumption was the notion that temperatures do not vary drastically during a season. This is shown in Figure 45, where hourly load data for working days has been plotted for the months of the spring season

(March to May). It can be seen that of the total of 528 hours in all the working days for the season, there was only one hour when a high peak of load values was detected. Hence, the power demand would not have any significant variance for the three month season being tested. Hence, a four month seasonal representation was considered appropriate for testing the annual load demand for MTLF to allow capturing variations in the data.

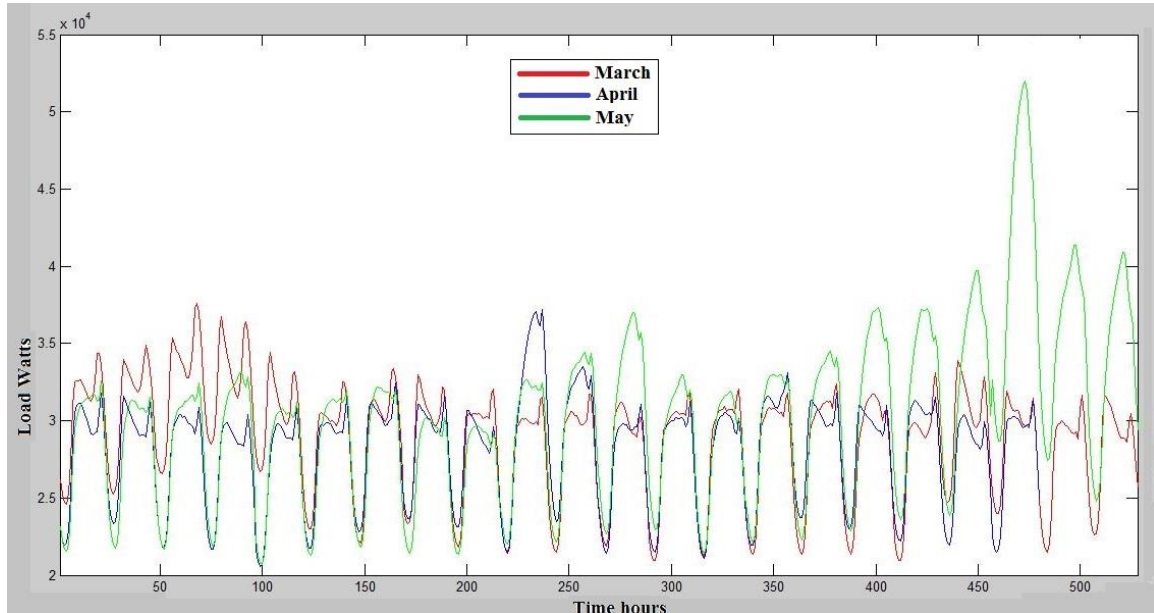


Figure 45: Comparison of hourly load data for the months of Mar-May in 2012

The models were applied to four different hours of a working day (2 am, 9 am, 2 pm and 8 pm). From a study of the hourly data for a day, it was clear that although the load demand varies over a whole day, there are certain periods of the day, when the load characteristics behave similarly (increases, decreases or hold constant). Hence, a day was divided into 4 parts (early morning, mid morning, afternoon and evening), as a reasonable approximate for the 24 hours. Figure 46 illustrates how the demand during certain periods of a day are similar.

5.2.1 Mid-term load forecasting (MTLF)

In this section, the forecasted values of load have been presented and the forecasting accuracies have been compared using statistical indices like MAE, MSE, RMSE, and MAPE. Along with the data tables, this section also uses charts to compare the error indices for each method. In each case, a whole month's load values were forecasted using four different methods and the results were compared to the actual data. For the test cases studied, the RF model and the RFSRIMA hybrid models yielded similar accuracies. When separately run, the LOWESS and the SARIMA models, were not quite accurate for the tested cases. However, when used together, the hybrid model LOWRIMA performed much better.

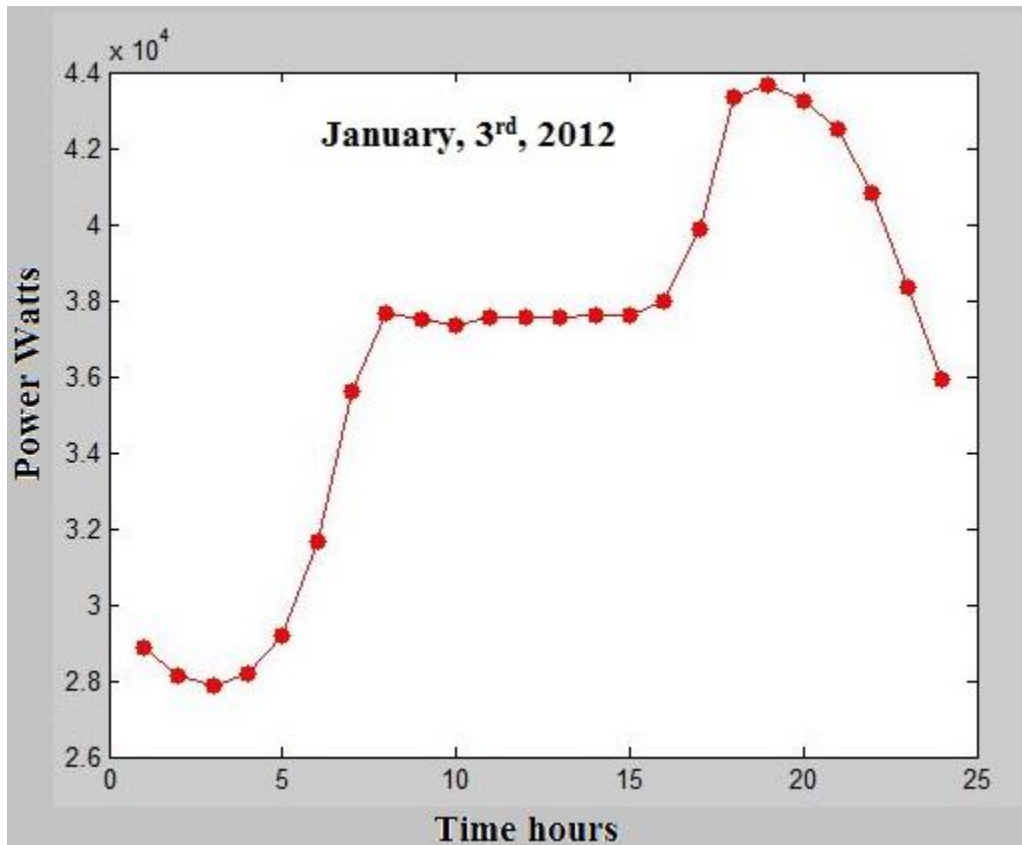


Figure 46: Demand power for Mid-Atlantic region on Jan 3rd, 2012

Table 17: Demand data predicted by each model for May 2015 at 2am

May 2015, 2am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
05/01/2015	23662.86	25260.20	26986.60	23327.75	22542.86	22639.09
05/04/2015	23166.08	24586.59	23973.48	22539.55	22370.67	23622.81
05/05/2015	22923.27	23012.76	23316.22	21081.97	21996.21	22059.73
05/06/2015	23395.35	24586.59	25535.02	22099.97	22398.85	22660.33
05/07/2015	23575.58	26497.27	25513.92	24625.90	22774.48	24959.52
05/08/2015	25377.92	28322.32	25421.23	26606.94	24814.34	25938.52
05/11/2015	22374.61	22534.18	22391.30	20521.01	21462.95	24270.48
05/12/2015	22117.32	21732.99	22176.03	20091.01	21451.97	21682.22
05/13/2015	22391.86	22104.53	23188.15	20231.58	21514.87	21587.79
05/14/2015	23276.53	25818.91	22947.41	23885.18	22397.63	25413.72
05/15/2015	23516.46	24791.13	22161.29	22831.73	22680.99	24345.59
05/18/2015	22381.42	22534.18	22403.86	20831.34	21391.39	25082.14
05/19/2015	22312.44	22328.21	23238.71	20144.35	21262.48	21872.17
05/20/2015	22620.85	22026.71	22973.26	19963.99	21642.45	21733.52
05/21/2015	25629.92	28322.32	23495.29	26843.84	24795.26	24009.07
05/22/2015	26643.06	29007.44	25035.37	27070.61	25670.86	27509.50
05/26/2015	26714.60	29007.44	23283.37	27079.75	25799.56	27027.23

Table 18: Prediction error percentages for May 2015 at 2am

Percentage Error May 2015, 2am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-4.52	-11.58	-19.20	-3.04	0.43
1.93	-4.08	-1.48	4.59	5.30
-3.91	-4.32	-5.70	4.43	0.29
-3.24	-8.50	-12.69	2.47	1.15
5.54	-6.16	-2.22	1.34	8.75
2.16	-9.19	1.99	-2.58	4.33
7.81	7.15	7.74	15.45	11.57
-2.01	-0.23	-2.28	7.34	1.06
-3.72	-2.39	-7.41	6.28	0.34
8.41	-1.59	9.70	6.01	11.87
3.41	-1.83	8.97	6.22	6.84
10.77	10.16	10.68	16.95	14.71
-2.01	-2.09	-6.25	7.90	2.79
-4.08	-1.35	-5.70	8.14	0.42
-6.75	-17.97	2.14	-11.81	-3.27
3.15	-5.45	8.99	1.60	6.68
1.16	-7.33	13.85	-0.19	4.54

Table 19: Standardized error for each model at 2am on May 2015

May 2015, 2am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1056.07	1448.75	1796.55	1477.98	1236.35
MSE	1541656.34	3274538.16	4565340.41	3467703.90	2764910.35
RMSE	1241.63	1809.57	2136.67	1862.18	1662.80
MAPE	4.39	5.96	7.47	6.25	4.96

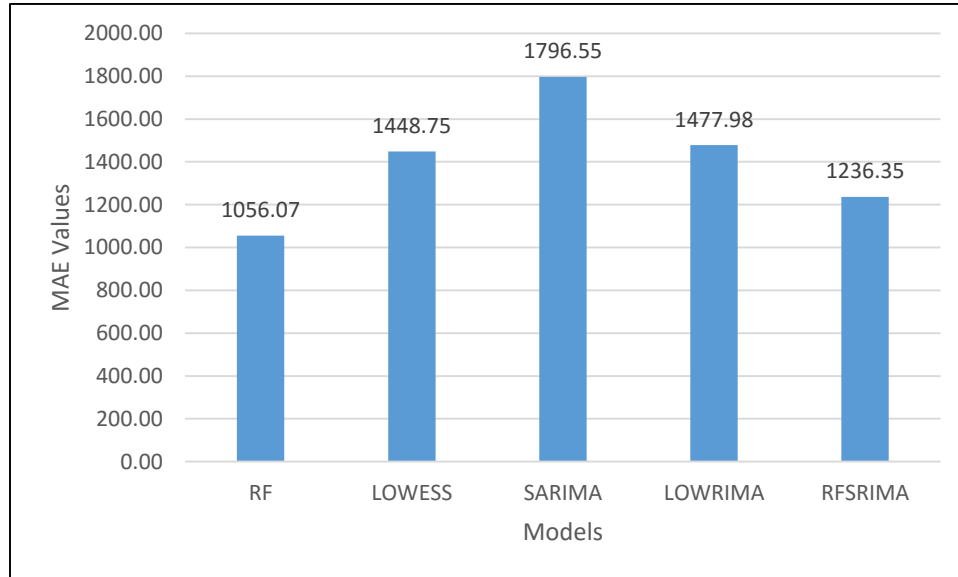


Figure 47: MAE values for May 2015 projection (MTLF 2012-2015) at 2am

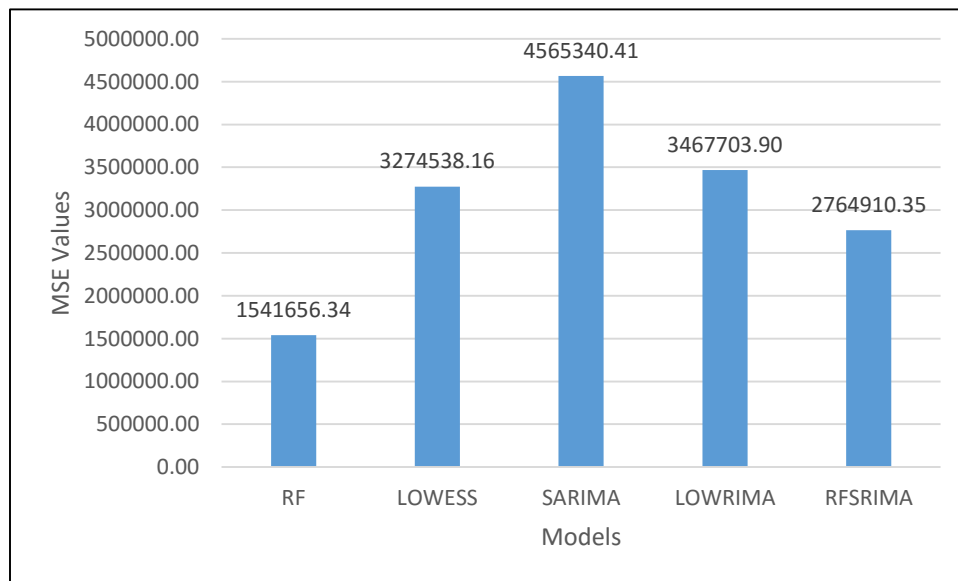


Figure 48: MSE values for May 2015 projection (MTLF 2012-2015) at 2am

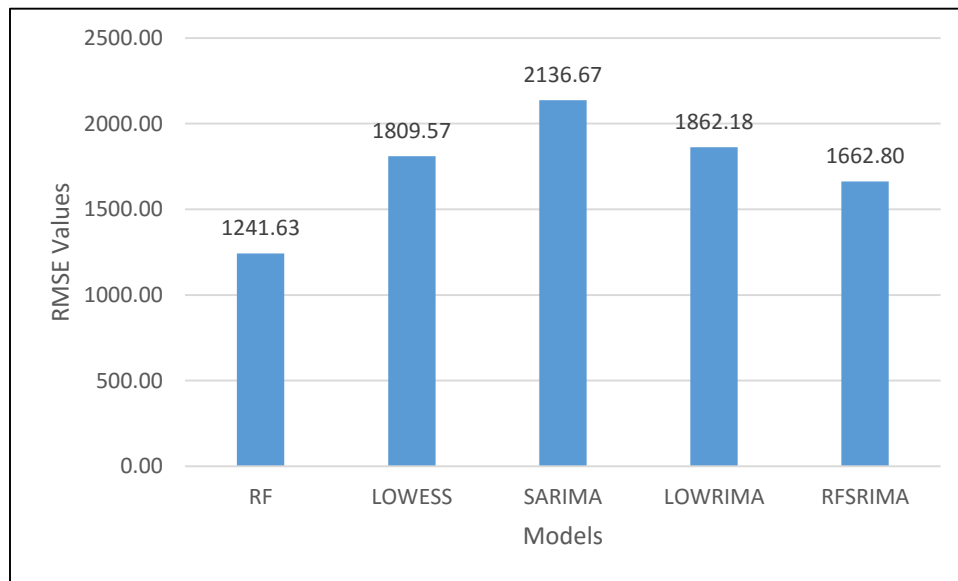


Figure 49: RMSE values for May 2015 projection (MTLF 2012-2015) at 2am

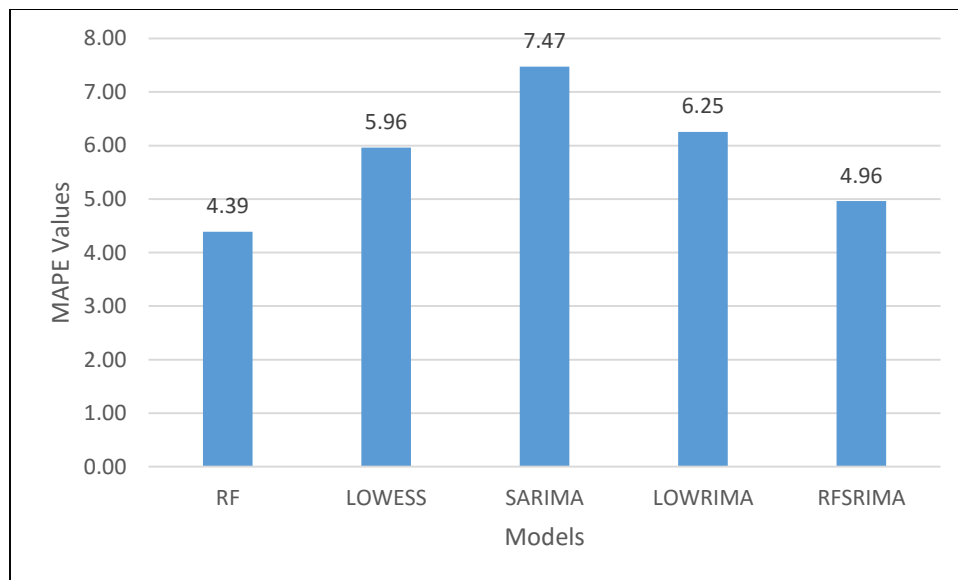


Figure 50: MAPE values for May 2015 projection (MTLF 2012-2015) at 2am

Table 20: Demand data predicted by each model for May 2015 at 9am

May 2015, 9am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
05/01/2015	31376.32	32297.53	34249.16	30408.34	31040.83	29545.54
05/04/2015	30769.37	30800.22	31682.15	29058.99	30317.88	30201.54
05/05/2015	30203.37	29985.43	30867.18	28074.43	28511.36	28823.61
05/06/2015	30359.82	30800.22	33791.61	28768.57	29446.17	29434.92
05/07/2015	32066.67	33214.05	33509.48	31578.28	31739.28	32905.66
05/08/2015	33177.26	34647.85	32985.06	33087.38	32632.03	33473.44
05/11/2015	29671.05	29770.65	30393.98	27921.72	28681.90	29458.07
05/12/2015	29234.22	28920.37	30194.50	27229.72	27838.78	28191.83
05/13/2015	29323.98	29245.41	30732.39	27387.68	28129.28	28383.38
05/14/2015	31475.06	32982.57	30644.35	31387.32	30305.75	33699.37
05/15/2015	31318.28	31598.18	29764.96	29680.16	29791.78	31347.59
05/18/2015	29677.58	29770.65	31231.90	28155.47	28500.15	29683.36
05/19/2015	29565.72	30181.59	30365.44	28456.84	27996.43	28720.01
05/20/2015	29436.58	29152.51	30467.09	27463.52	27695.07	28011.67
05/21/2015	33309.09	34647.85	31430.29	33371.69	31503.45	32076.01

Table 21: Prediction error percentages for May 2015 at 9am

Percentage Error May 2015, 9am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-6.20	-9.31	-15.92	-2.92	-5.06
-1.88	-1.98	-4.90	3.78	-0.39
-4.79	-4.03	-7.09	2.60	1.08
-3.14	-4.64	-14.80	2.26	-0.04
2.55	-0.94	-1.84	4.03	3.54
0.88	-3.51	1.46	1.15	2.51
-0.72	-1.06	-3.18	5.22	2.63
-3.70	-2.58	-7.10	3.41	1.25
-3.31	-3.04	-8.28	3.51	0.90
6.60	2.13	9.07	6.86	10.07
0.09	-0.80	5.05	5.32	4.96
0.02	-0.29	-5.22	5.15	3.99
-2.94	-5.09	-5.73	0.92	2.52
-5.09	-4.07	-8.77	1.96	1.13
-3.84	-8.02	2.01	-4.04	1.78

Table 22: Standardized error for each model at 9am on May 2015

May 2015, 9am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1056.07	1448.75	1796.55	1477.98	1236.35
MSE	1541656.34	3274538.16	4565340.41	3467703.90	2764910.35
RMSE	1241.63	1809.57	2136.67	1862.18	1662.80
MAPE	4.39	5.96	7.47	6.25	4.96

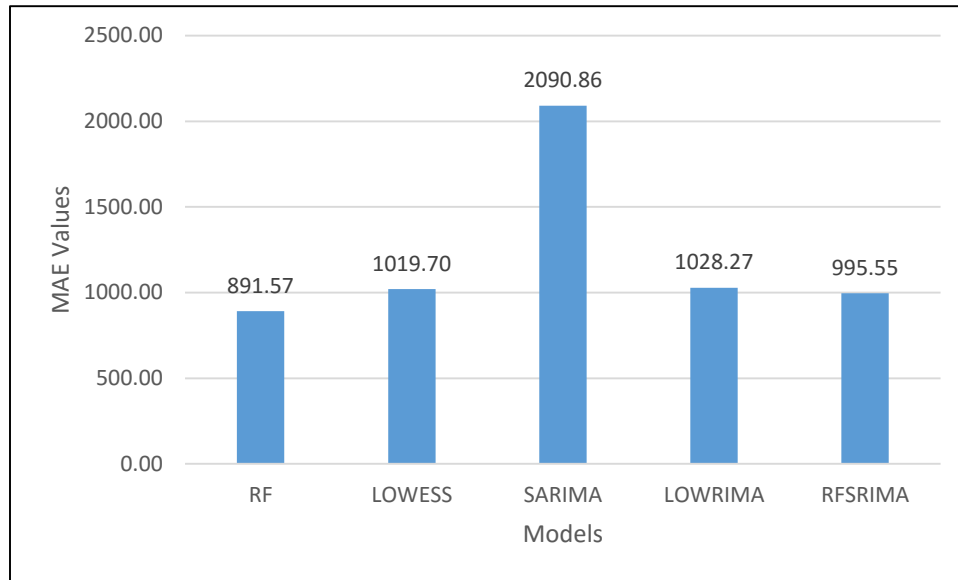


Figure 51: MAE values for May 2015 projection (MTLF 2012-2015) at 9am

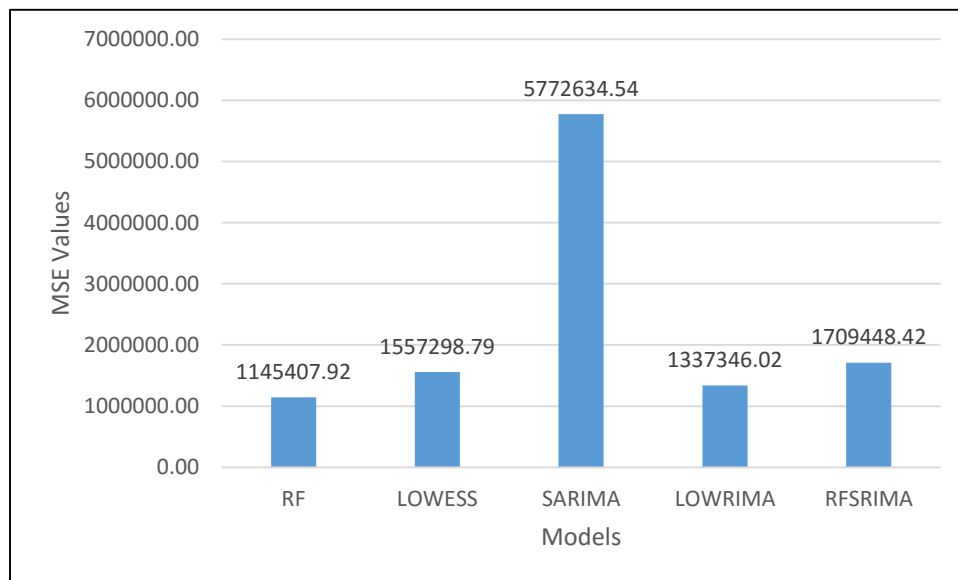


Figure 52: MSE values for May 2015 projection (MTLF 2012-2015) at 9am

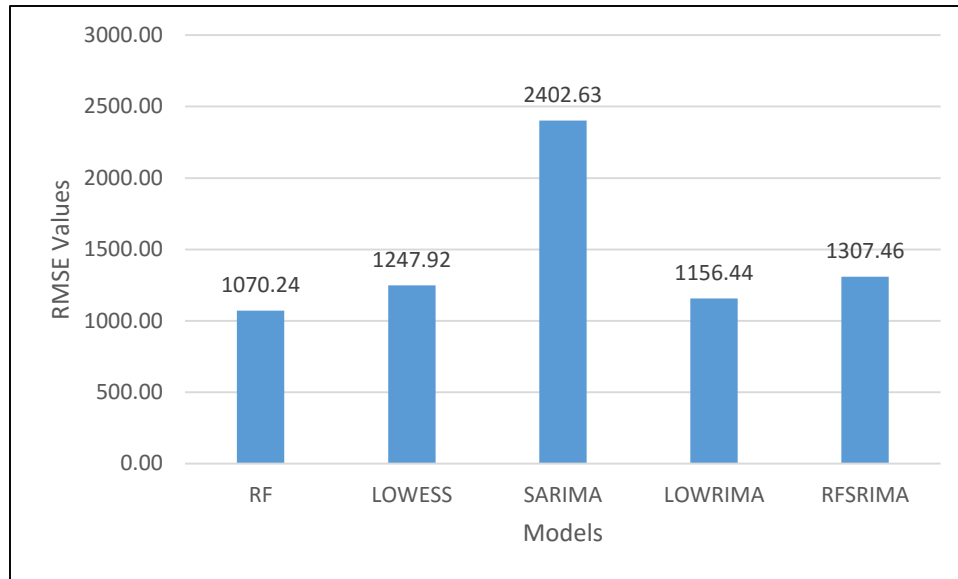


Figure 53: RMSE values for May 2015 projection (MTLF 2012-2015) at 9am

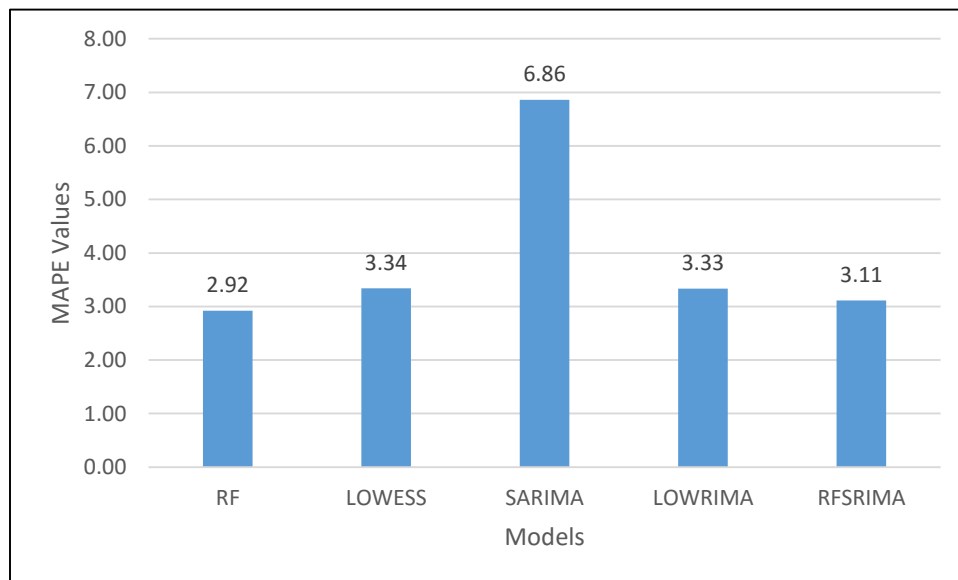


Figure 54: MAPE values for May 2015 projection (MTLF 2012-2015) at 9am

Table 23: Demand data predicted by each model for May 2015 at 2pm

May 2015, 2pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
05/01/2015	34611.69	35109.61	31390.93	33346.53	34421.97	32234.76
05/04/2015	32357.07	32643.43	30240.04	29702.52	33229.88	32169.47
05/05/2015	33958.65	35109.61	29520.11	32673.31	34277.59	34387.97
05/06/2015	38594.53	40301.68	30986.62	39114.35	38704.38	39040.84
05/07/2015	40107.06	41523.07	32231.47	41326.25	40118.97	39502.57
05/08/2015	30981.26	31726.35	33850.68	32943.70	29620.17	30153.30
05/11/2015	29857.83	29603.92	32381.14	28344.79	28685.25	29424.67
05/12/2015	30890.27	31107.84	28864.18	28392.81	29196.72	29997.65
05/13/2015	36257.84	38856.42	29717.77	36825.35	34955.63	38874.41
05/14/2015	34690.51	37163.70	28539.48	34760.04	33583.49	35792.23
05/15/2015	30979.09	31726.35	28546.15	29857.10	30243.64	30499.74
05/18/2015	29903.50	29917.32	29535.77	27270.65	27967.92	29591.96
05/19/2015	29855.28	29686.84	30303.23	27793.20	30033.61	29141.76
05/20/2015	40334.01	41523.07	28394.54	39239.81	38977.47	40662.59

Table 24: Prediction error percentages for May 2015 at 2pm

Percentage Error May 2015, 2pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-4.29	-13.05	4.65	-6.41	-3.99
-7.37	-8.92	2.62	-3.45	-6.79
-0.58	-1.47	6.00	7.67	-3.30
1.25	-2.10	14.16	4.99	0.32
1.14	-3.23	20.63	-0.19	0.86
-1.53	-5.11	18.41	-4.62	-1.56
-2.75	-5.22	-12.26	-9.25	1.77
-1.47	-0.61	-10.05	3.67	2.51
-2.98	-3.70	3.78	5.35	2.67
6.73	0.05	23.55	5.27	10.08
3.08	-3.83	20.26	2.88	6.17
-1.57	-4.02	6.41	2.11	0.84
-1.05	-1.10	0.19	7.84	5.49
-2.45	-1.87	-3.99	4.63	-3.06
0.81	-2.12	30.17	3.50	4.14

Table 25: Standardized error for each model at 2pm on May 2015

May 2015, 2pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1010.08	1182.06	5309.11	1638.06	1381.61
MSE	1604667.75	2519077.58	47333705.15	3217901.07	2924795.25
RMSE	1266.75	1587.16	6879.95	1793.85	1710.20
MAPE	2.84	3.48	14.01	4.80	3.86

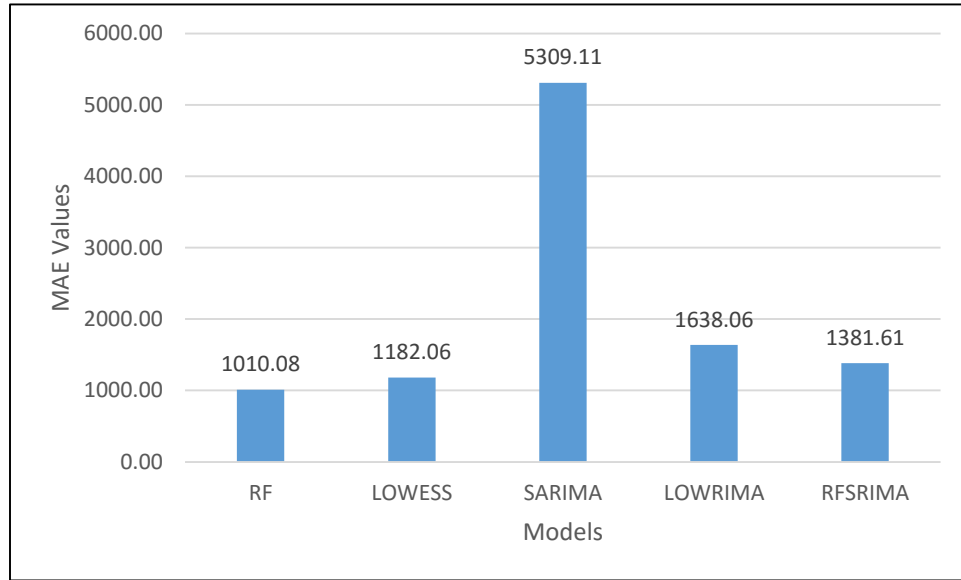


Figure 55: MAE values for May 2015 projection (MTLF 2012-2015) at 2pm

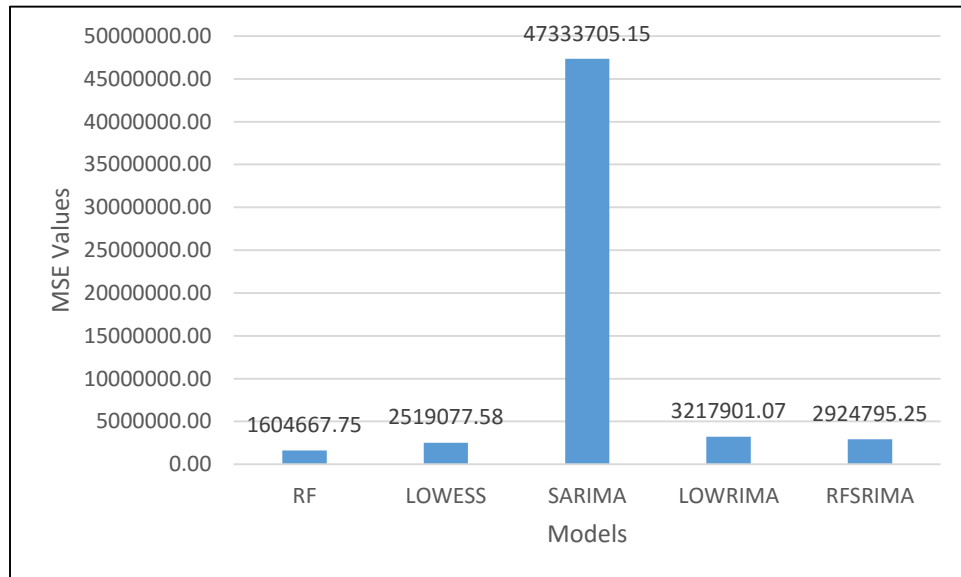


Figure 56: MSE values for May 2015 projection (MTLF 2012-2015) at 2pm

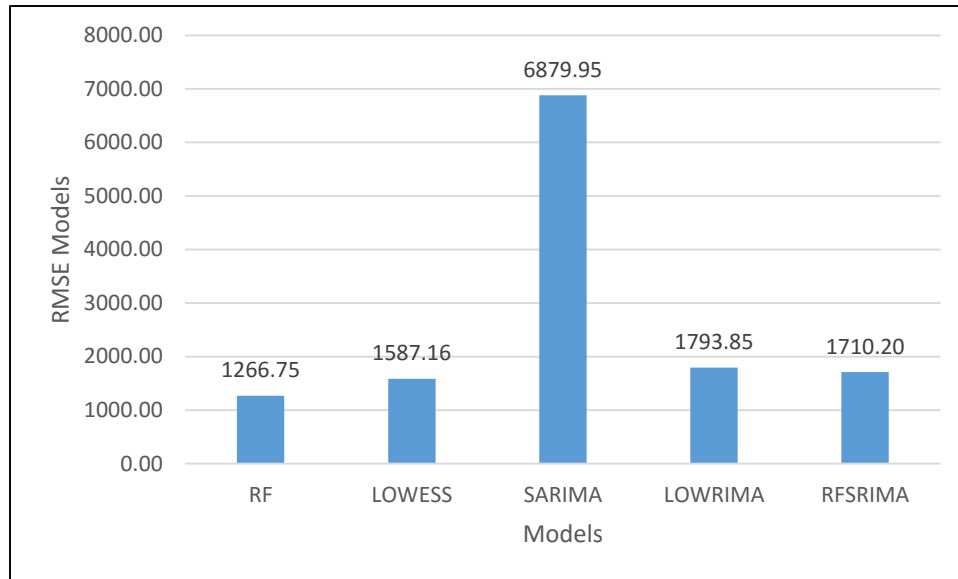


Figure 57: RMSE values for May 2015 projection (MTLF 2012-2015) at 2pm

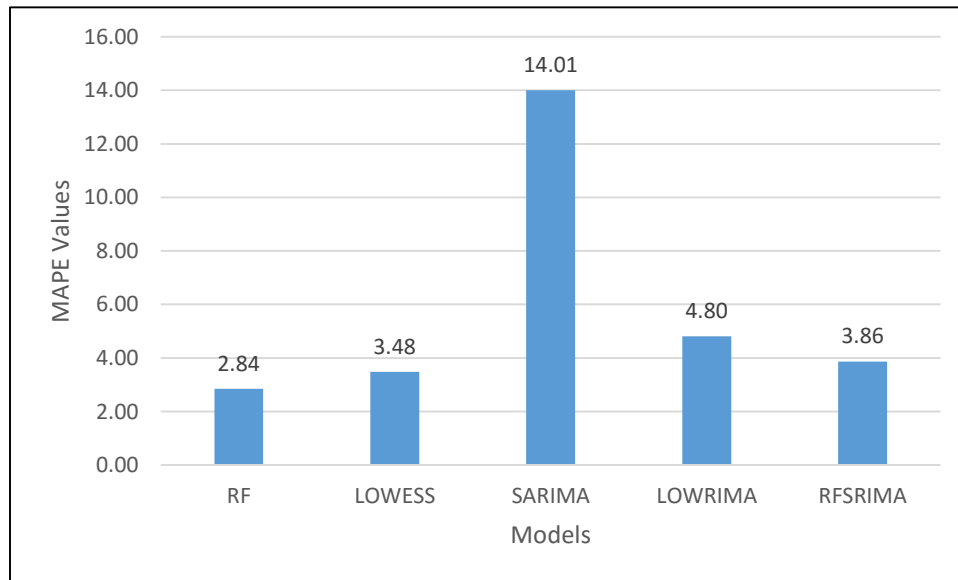


Figure 58: MAPE values for May 2015 projection (MTLF 2012-2015) at 2pm

Table 26: Demand data predicted by each model for May 2015 at 8pm

May 2015, 8pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
05/01/2015	33381.46	34532.58	29657.49	31067.79	33369.33	31205.08
05/04/2015	31681.20	32089.34	28949.80	29080.70	31830.52	32515.24
05/05/2015	32527.60	34532.58	29667.47	32315.53	32708.51	33388.00
05/06/2015	38533.35	39170.55	29785.20	38062.35	38613.33	38108.71
05/07/2015	39809.98	41066.91	29459.28	41840.52	39764.94	38193.93
05/08/2015	30856.61	31800.84	29581.63	32012.69	30690.39	29483.68
05/11/2015	29511.02	30538.34	28609.84	27802.29	29841.55	29377.19
05/12/2015	29507.46	31704.20	28783.56	28932.52	29353.90	29010.41
05/13/2015	36080.58	38033.18	28896.10	36096.50	36137.29	35789.12
05/14/2015	34274.81	36943.88	27915.44	34430.46	34383.15	38687.26
05/15/2015	30875.58	31800.84	28191.90	29846.85	30993.52	29722.10
05/18/2015	29660.64	31263.48	28465.57	28120.26	29490.51	29013.88
05/19/2015	28853.99	30137.04	29056.99	27880.07	28814.56	27973.59
05/20/2015	39850.35	41066.91	28315.03	38669.36	39933.99	41732.70

Table 27: Prediction error percentages for May 2015 at 8pm

Percentage Error May 2015, 8pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-2.60	-10.72	12.67	-3.75	-1.99
-6.97	-10.66	4.96	0.44	-6.94
2.57	1.31	10.97	10.56	2.11
2.58	-3.43	11.14	3.21	2.04
-1.11	-2.79	21.84	0.12	-1.32
-4.23	-7.52	22.87	-9.55	-4.11
-4.66	-7.86	-0.33	-8.58	-4.09
-0.46	-3.95	2.61	5.36	-1.58
-1.71	-9.29	0.78	0.27	-1.18
-0.81	-6.27	19.26	-0.86	-0.97
11.41	4.51	27.84	11.00	11.13
-3.88	-6.99	5.15	-0.42	-4.28
-2.23	-7.75	1.89	3.08	-1.64
-3.15	-7.73	-3.87	0.33	-3.01
4.51	1.60	32.15	7.34	4.31

Table 28: Standardized error for each model at 8pm on May 2015

May 2015, 8pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1170.42	1955.35	5305.15	1517.26	1115.76
MSE	2337308.99	4666966.08	49841783.89	4410728.65	2183832.48
RMSE	1528.83	2160.32	7059.87	2100.17	1477.78
MAPE	3.37	5.94	14.05	4.28	3.21

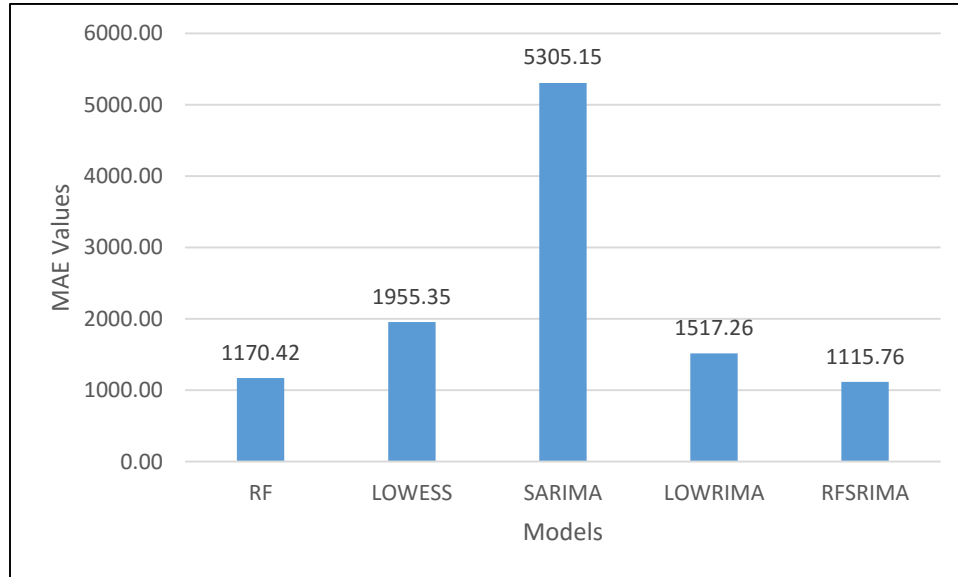


Figure 59: MAE values for May 2015 projection (MTLF 2012-2015) at 8pm

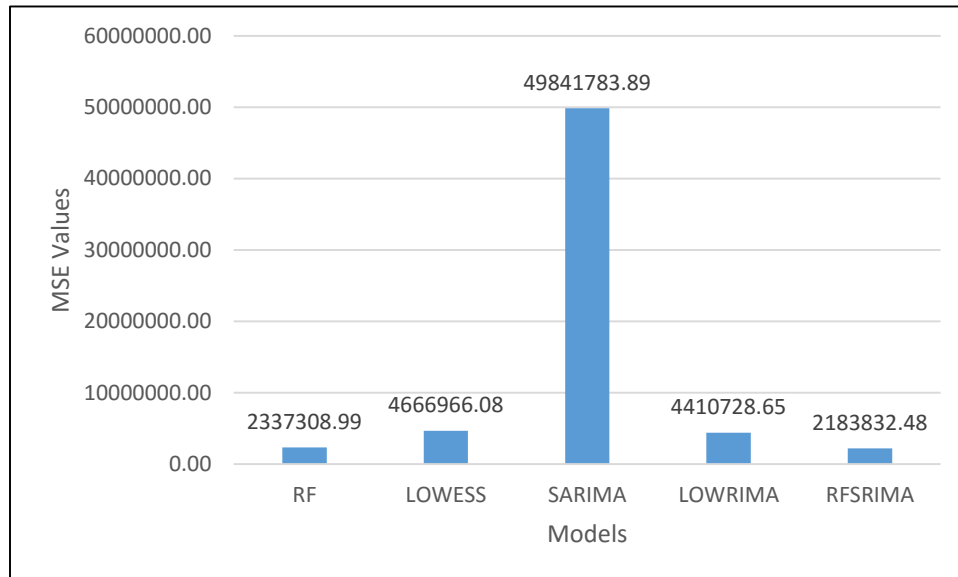


Figure 60: MSE values for May 2015 projection (MTLF 2012-2015) at 8pm

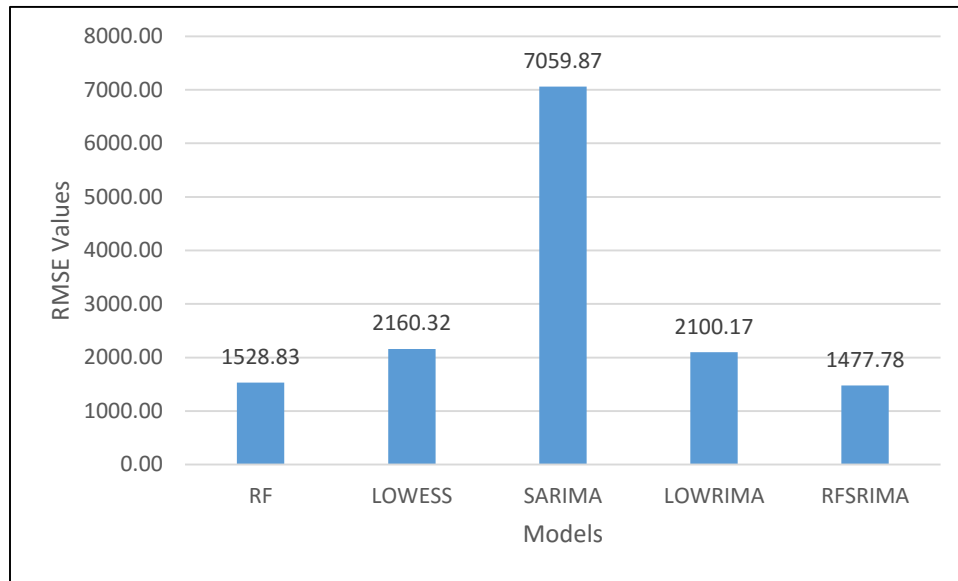


Figure 61: RMSE values for May 2015 projection (MTLF 2012-2015) at 8pm

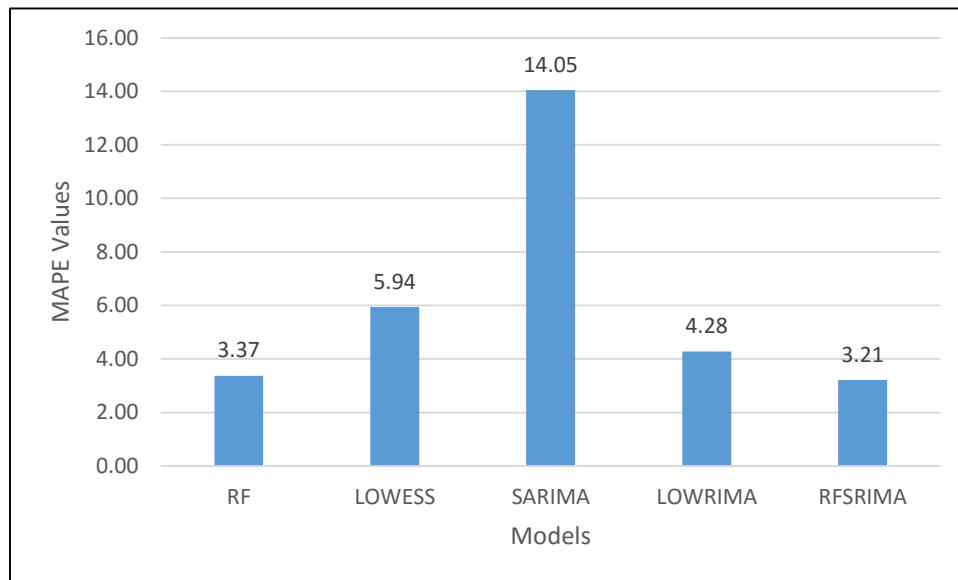


Figure 62: MAPE values for May 2015 projection (MTLF 2012-2015) at 8pm

Table 29: Demand data predicted by each model for August 2015 at 2am

August 2015, 2am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
08/03/2015	30467.03	30326.73	30460.54	31447.46	30986.47	31975.05
08/04/2015	29771.76	30499.75	32863.75	28537.91	29827.04	29841.56
08/05/2015	27034.35	26133.55	32281.11	24827.18	27742.78	28465.52
08/06/2015	27247.21	26133.55	32018.15	27094.51	27298.79	27774.64
08/07/2015	26003.64	26516.21	34976.47	26799.25	25918.33	27190.08
08/10/2015	28806.63	28093.17	37004.98	27609.05	29109.41	27819.74
08/11/2015	27608.01	26516.21	35823.62	26696.28	27188.09	27779.77
08/12/2015	26420.03	25318.31	35743.57	25843.77	26153.57	26495.53
08/13/2015	26867.62	26133.55	33736.61	28397.69	26052.35	25870.82
08/14/2015	30410.10	30940.49	34365.22	33572.16	30635.62	30245.43
08/17/2015	30326.92	30940.49	35863.95	31006.59	30831.27	32200.88
08/18/2015	30630.06	30940.49	31418.35	32382.76	29749.91	31788.02
08/19/2015	30721.14	30940.49	30988.72	32586.92	30630.48	32081.41
08/20/2015	29170.20	28723.90	41402.42	30107.11	30847.71	31061.29
08/21/2015	27876.44	28093.17	40585.13	29239.09	30091.41	26655.93
08/24/2015	28565.46	28723.90	35172.93	30189.58	29251.90	29824.07
08/25/2015	26200.83	24852.18	32340.70	25926.26	26433.46	26961.88
08/26/2015	25552.33	24349.30	31825.54	25706.31	24355.16	25327.51

Table 30: Prediction error percentages for August 2015 at 2am

Percentage Error August 2015, 2am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
4.72	5.16	4.74	1.65	3.09
0.23	-2.21	-10.13	4.37	0.05
5.03	8.19	-13.40	12.78	2.54
1.90	5.91	-15.28	2.45	1.71
4.36	2.48	-28.64	1.44	4.68
-3.55	-0.98	-33.02	0.76	-4.64
0.62	4.55	-28.96	3.90	2.13
0.28	4.44	-34.90	2.46	1.29
-3.85	-1.02	-30.40	-9.77	-0.70
-0.54	-2.30	-13.62	-11.00	-1.29
5.82	3.91	-11.38	3.71	4.25
3.64	2.67	1.16	-1.87	6.41
4.24	3.56	3.41	-1.58	4.52
6.09	7.53	-33.29	3.07	0.69
-4.58	-5.39	-52.26	-9.69	-12.89
4.22	3.69	-17.93	-1.23	1.92
2.82	7.82	-19.95	3.84	1.96
-0.89	3.86	-25.66	-1.50	3.84

Table 31: Standardized error for each model at 2am on August 2015

August 2015, 2am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	882.05	1183.81	6136.31	1255.88	957.58
MSE	1118252.37	1754304.85	49601630.43	2590727.64	1493942.10
RMSE	1057.47	1324.50	7042.84	1609.57	1222.27
MAPE	3.01	4.11	22.17	4.42	3.35

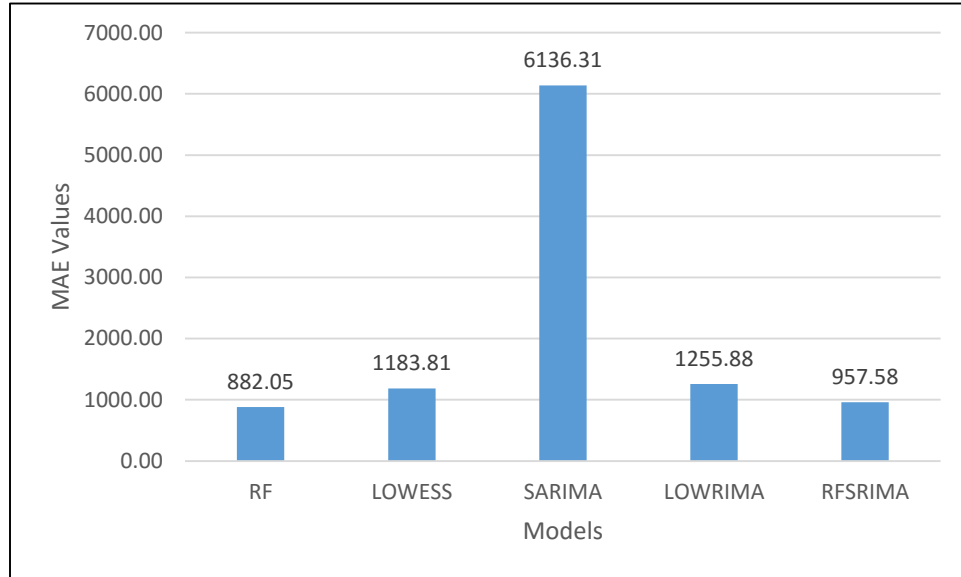


Figure 63: MAE values for Aug 2015 projection (MTLF 2012-2015) at 2am

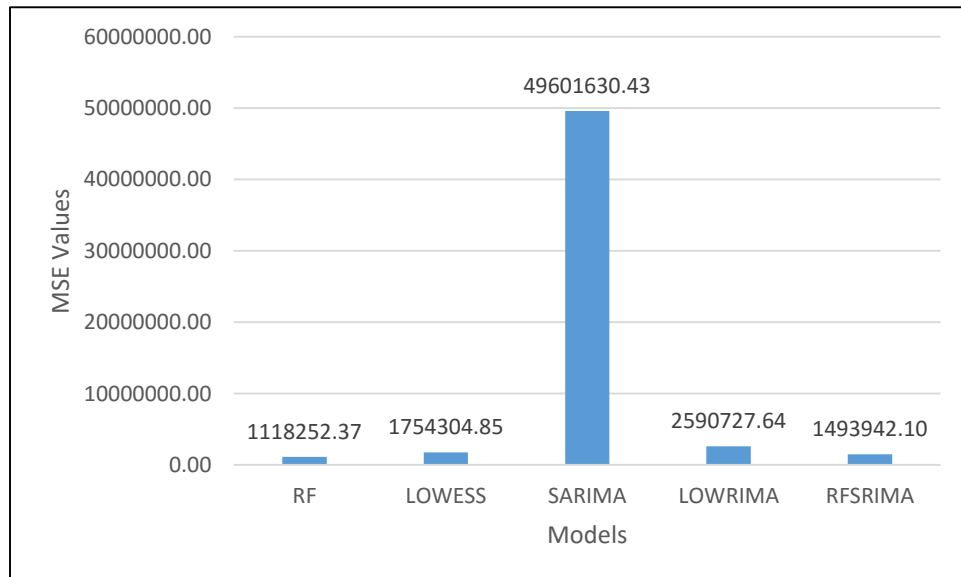


Figure 64: MSE values for Aug 2015 projection (MTLF 2012-2015) at 2am

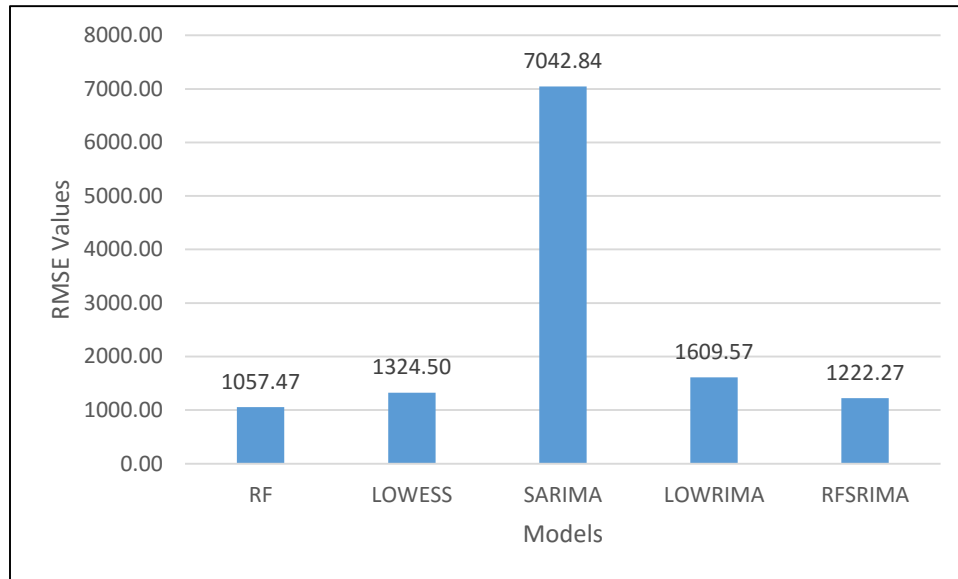


Figure 65: RMSE values for Aug 2015 projection (MTLF 2012-2015) at 2am

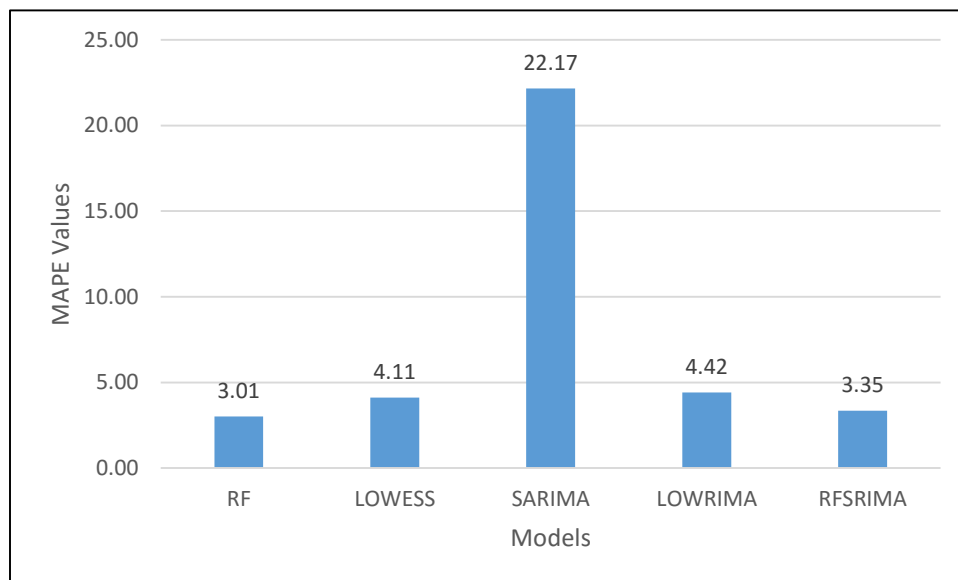


Figure 66: MAPE values for Aug 2015 projection (MTLF 2012-2015) at 2am

Table 32: Demand data predicted by each model for August 2015 at 9am

August 2015, 9am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
08/03/2015	37911.93	38051.03	36393.31	36984.88	37279.96	36690.66
08/04/2015	36641.97	36701.39	37487.40	35875.82	35259.18	34757.05
08/05/2015	33165.94	32902.29	36265.60	32013.30	32249.56	33159.69
08/06/2015	33577.47	32902.29	38576.39	32196.79	32984.18	32761.59
08/07/2015	33034.42	33158.31	41434.17	32297.11	31665.28	33629.12
08/10/2015	35346.91	34424.02	42406.66	33269.17	34969.91	34303.97
08/11/2015	33826.19	33158.31	40663.51	32013.85	33188.32	33554.57
08/12/2015	32673.24	32026.47	39208.81	31145.87	31598.73	32011.02
08/13/2015	33249.54	32902.29	39456.55	32007.75	33568.67	31523.59
08/14/2015	38271.38	37942.51	40109.61	37237.51	37723.48	36996.86
08/17/2015	37719.85	37942.51	40154.37	37167.31	37284.36	37775.96
08/18/2015	37980.61	37942.51	35327.92	37064.81	39057.01	38834.80
08/19/2015	38100.45	37942.51	36309.58	37268.80	38713.05	39259.32
08/20/2015	35703.19	35403.31	46659.80	34451.01	34712.60	35607.93
08/21/2015	35283.44	34424.02	44395.81	33537.80	34678.34	33508.06
08/24/2015	35206.62	35403.31	38470.34	34343.84	34655.42	35576.48
08/25/2015	31859.27	31604.59	36911.00	30724.20	31037.59	32491.97

Table 33: Prediction error percentages for August at 2015 9am

Percentage Error August 2015, 9am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-3.33	-3.71	0.81	-0.80	-1.61
-5.42	-5.59	-7.86	-3.22	-1.44
-0.02	0.78	-9.37	3.46	2.74
-2.49	-0.43	-17.75	1.72	-0.68
1.77	1.40	-23.21	3.96	5.84
-3.04	-0.35	-23.62	3.02	-1.94
-0.81	1.18	-21.19	4.59	1.09
-2.07	-0.05	-22.49	2.70	1.29
-5.48	-4.37	-25.17	-1.54	-6.49
-3.44	-2.56	-8.41	-0.65	-1.96
0.15	-0.44	-6.30	1.61	1.30
2.20	2.30	9.03	4.56	-0.57
2.95	3.35	7.51	5.07	1.39
-0.27	0.57	-31.04	3.25	2.51
-5.30	-2.73	-32.49	-0.09	-3.49
1.04	0.49	-8.13	3.46	2.59
1.95	2.73	-13.60	5.44	4.48
1.68	1.14	-16.29	3.93	1.25

Table 34: Standardized error for each model at 9am on August 2015

August 2015, 9am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	817.86	649.65	5550.36	969.81	787.97
MSE	971206.78	708373.18	39410798.64	1253979.95	875688.65
RMSE	985.50	841.65	6277.80	1119.81	935.78
MAPE	2.37	1.85	16.42	2.81	2.32

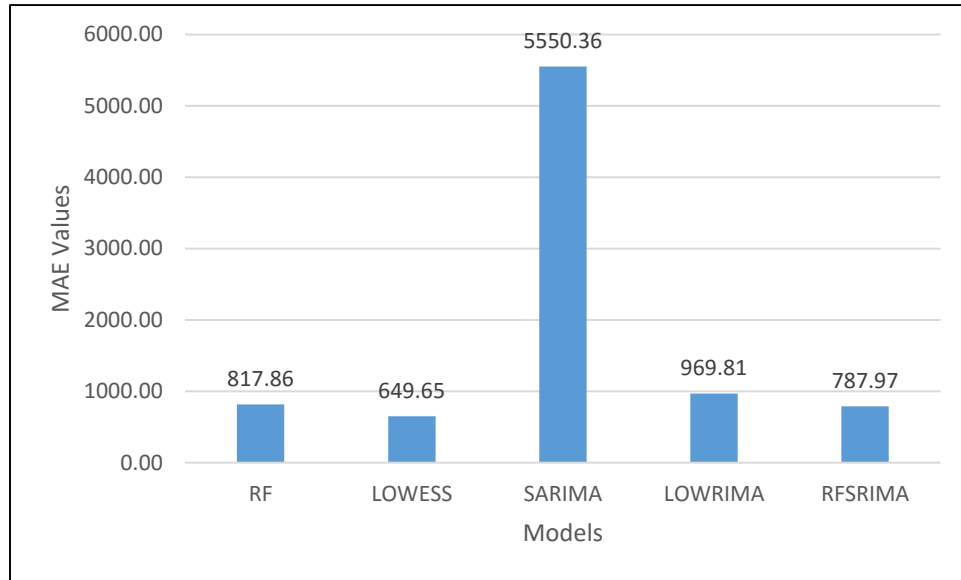


Figure 67: MAE values for Aug 2015 projection (MTLF 2012-2015) at 9am

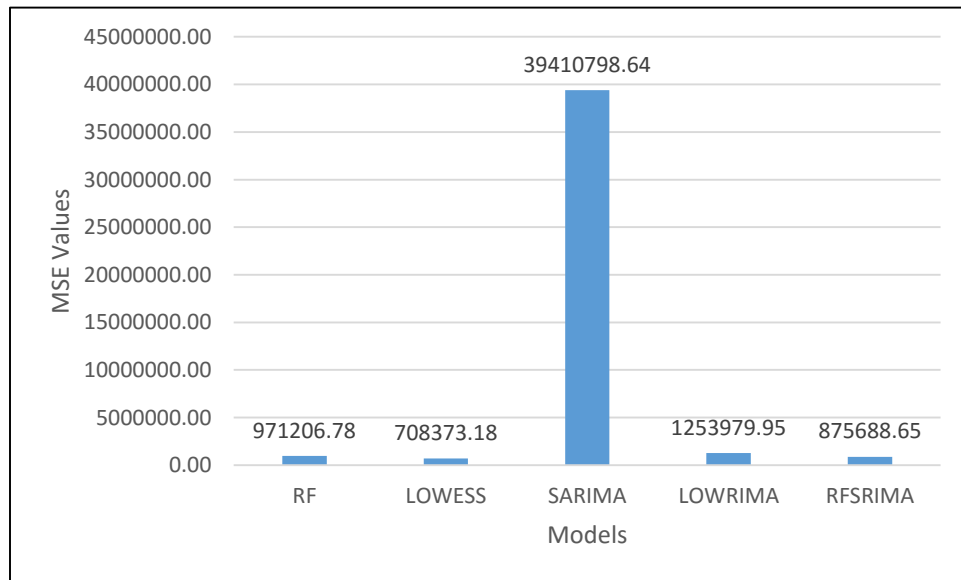


Figure 68: MSE values for Aug 2015 projection (MTLF 2012-2015) at 9am

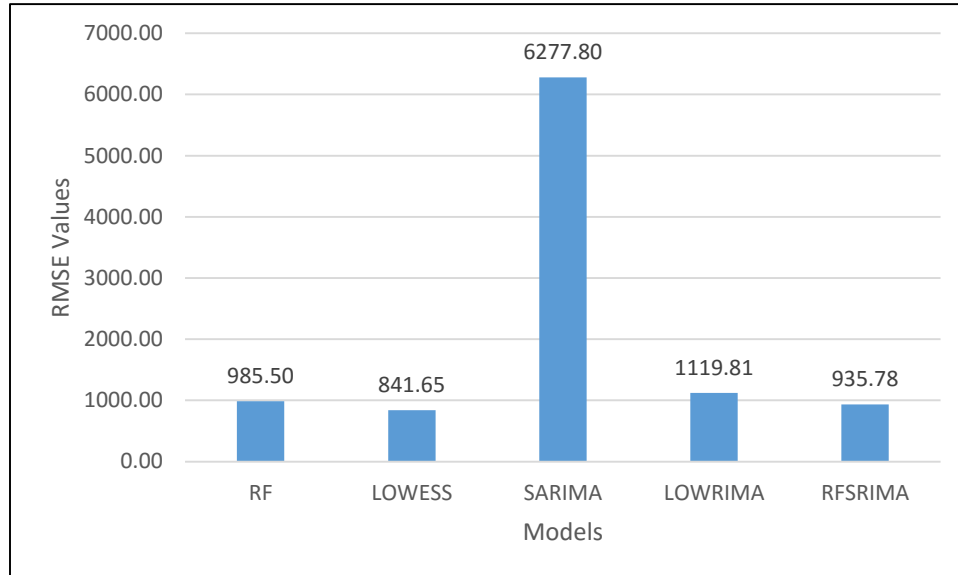


Figure 69: RMSE values for Aug 2015 projection (MTLF 2012-2015) at 9am

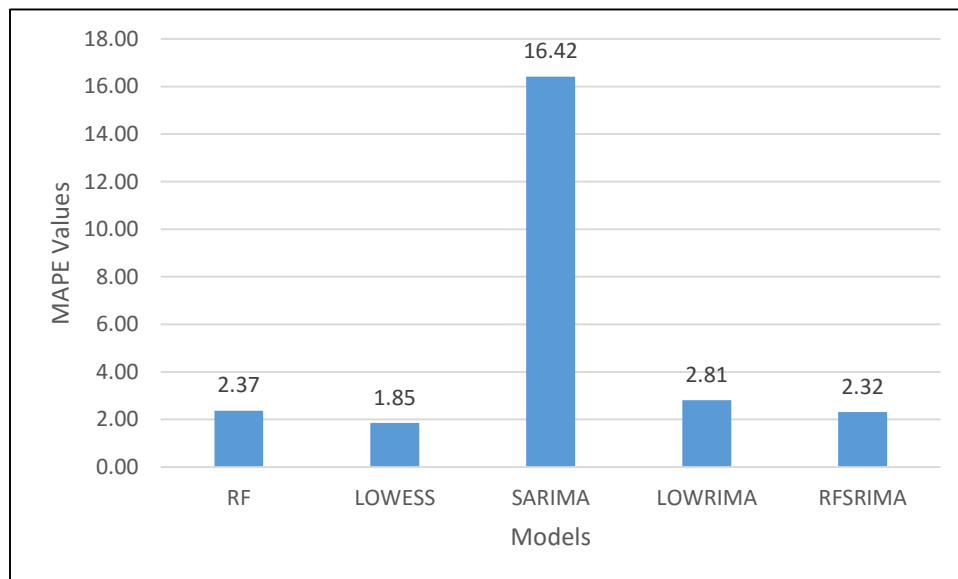


Figure 70: MAPE values for Aug 2015 projection (MTLF 2012-2015) at 9am

Table 35: Demand data predicted by each model for August 2015 at 2pm

August 2015, 2pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
08/03/2015	48651.31	48853.57	44904.43	48950.52	46651.97	47783.47
08/04/2015	45205.51	45263.99	45655.03	44891.33	43510.13	44295.73
08/05/2015	40723.87	38889.37	45827.32	36763.23	39531.74	40549.58
08/06/2015	40071.68	38889.37	45383.41	39985.70	38954.91	39286.18
08/07/2015	41106.58	39860.05	44857.68	42793.30	39864.72	38415.81
08/10/2015	42601.50	41573.34	44931.19	44517.44	41086.59	40522.01
08/11/2015	41055.16	39860.05	45059.69	40492.91	41723.91	39946.63
08/12/2015	38859.00	37269.24	45531.40	40782.97	38382.96	39233.87
08/13/2015	39016.66	38889.37	45973.14	43716.92	38085.43	40626.07
08/14/2015	48050.60	47057.09	45397.99	49039.05	47183.71	51055.51
08/17/2015	47001.13	47057.09	45950.71	49273.08	46768.39	49566.65
08/18/2015	47626.37	47057.09	45772.60	49231.34	48493.67	48792.07
08/19/2015	48035.55	47057.09	45666.69	47643.01	46876.27	47294.03
08/20/2015	43666.80	43031.83	45549.30	45512.71	42946.85	42890.69
08/21/2015	43407.82	41573.34	46090.29	42549.29	42249.81	44588.13
08/24/2015	43066.10	43031.83	45818.32	44933.35	43154.74	44085.93
08/25/2015	38924.90	37126.89	45686.23	38960.89	37990.53	38796.38

Table 36: Prediction error percentages for August at 2015 2pm

Percentage Error August 2015, 2pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-1.82	-2.24	6.03	-2.44	2.37
-2.05	-2.19	-3.07	-1.34	1.77
-0.43	4.09	-13.02	9.34	2.51
-2.00	1.01	-15.52	-1.78	0.84
-7.00	-3.76	-16.77	-11.40	-3.77
-5.13	-2.59	-10.88	-9.86	-1.39
-2.78	0.22	-12.80	-1.37	-4.45
0.96	5.01	-16.05	-3.95	2.17
3.96	4.27	-13.16	-7.61	6.25
5.89	7.83	11.08	3.95	7.58
5.18	5.06	7.30	0.59	5.65
2.39	3.56	6.19	-0.90	0.61
-1.57	0.50	3.44	-0.74	0.88
-1.81	-0.33	-6.20	-6.11	-0.13
2.65	6.76	-3.37	4.57	5.24
2.31	2.39	-3.93	-1.92	2.11
-0.33	4.30	-17.76	-0.42	2.08
3.45	5.15	-23.01	-0.72	0.50

Table 37: Standardized error for each model at 2pm on August 2015

August 2015, 2pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1219.80	1434.43	4403.20	1603.70	1175.32
MSE	2108508.55	2999316.36	25155541.62	4276045.21	2346721.45
RMSE	1452.07	1731.85	5015.53	2067.86	1531.90
MAPE	2.83	3.33	10.75	3.85	2.69

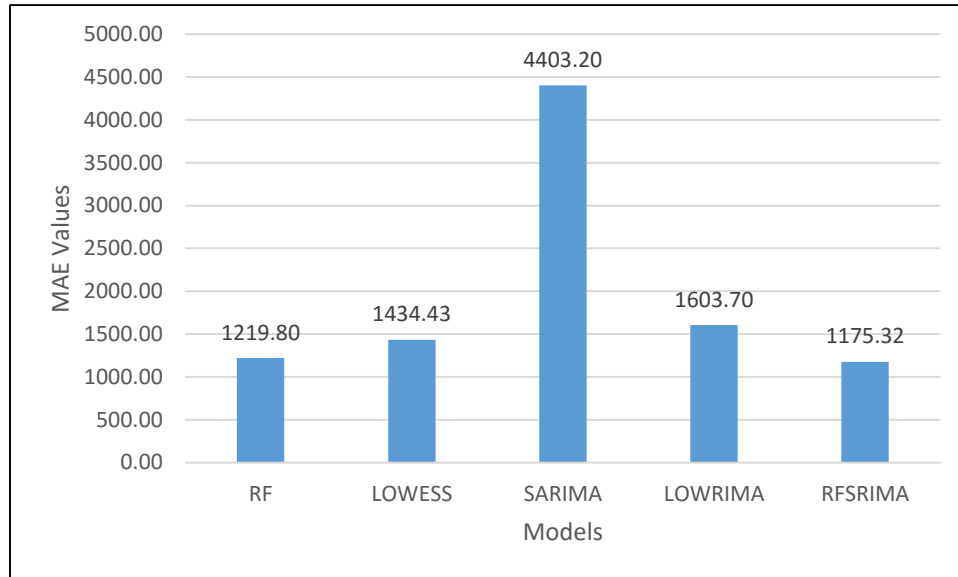


Figure 71: MAE values for Aug 2015 projection (MTLF 2012-2015) at 2pm

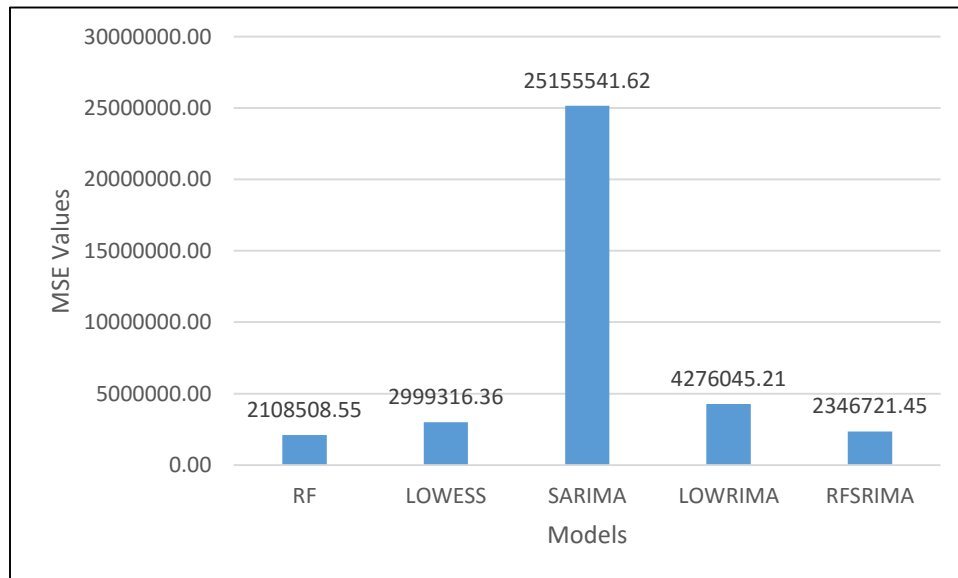


Figure 72: MSE values for Aug 2015 projection (MTLF 2012-2015) at 2pm

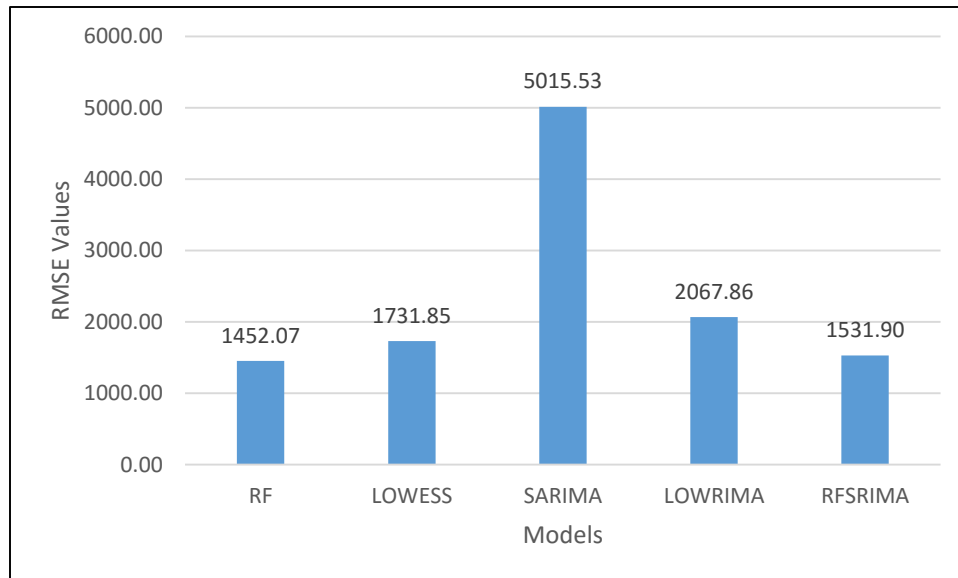


Figure 73: RMSE values for Aug 2015 projection (MTLF 2012-2015) at 2pm

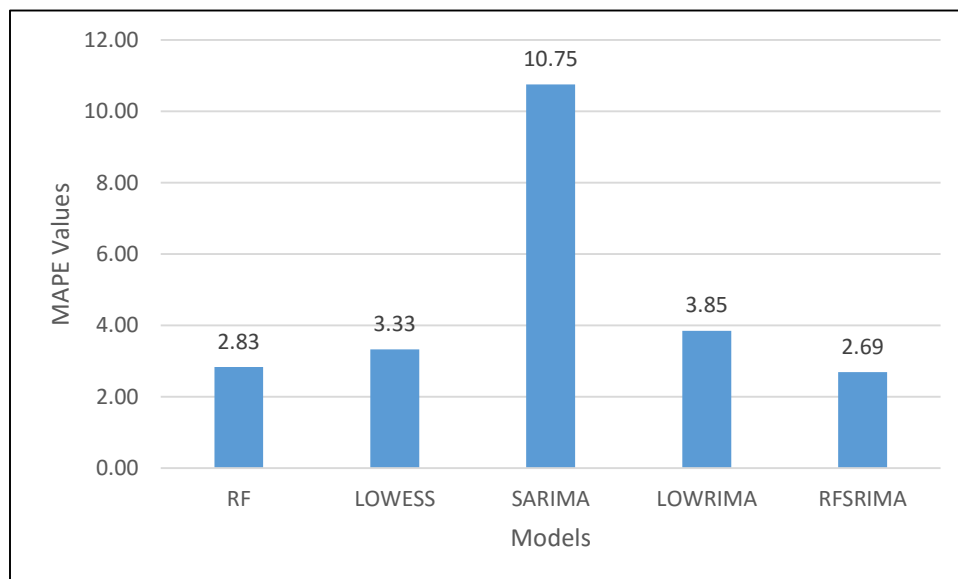


Figure 74: MAPE values for Aug 2015 projection (MTLF 2012-2015) at 2pm

Table 38: Demand data predicted by each model for August 2015 at 8pm

August 2015, 8pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
08/03/2015	46542.30	47580.53	44288.97	47138.37	46564.90	45983.03
08/04/2015	44545.54	44732.41	44742.14	43716.20	44721.47	43246.47
08/05/2015	38861.99	37899.46	44404.21	36611.63	37194.75	38687.54
08/06/2015	36588.87	37899.46	43787.99	37883.07	36292.72	38333.63
08/07/2015	39528.78	38942.08	43492.43	40670.27	39246.28	36701.91
08/10/2015	41973.73	40949.68	43527.57	39815.49	40543.09	40205.56
08/11/2015	40058.37	38942.08	44227.55	37604.89	38706.88	39695.85
08/12/2015	38529.41	36837.97	44388.49	35003.27	36773.74	38611.60
08/13/2015	36696.30	37899.46	44879.00	38202.74	37017.27	41036.78
08/14/2015	45837.03	45404.06	44189.64	46451.21	46051.68	49012.29
08/17/2015	45255.86	45404.06	44965.96	42854.63	45200.40	46474.00
08/18/2015	45605.26	45404.06	44191.52	47654.63	46048.52	45816.68
08/19/2015	45991.42	45404.06	44208.67	45279.65	46138.74	43740.81
08/20/2015	42621.43	42193.17	44544.90	41216.85	42793.21	40943.41
08/21/2015	42572.90	40949.68	45106.40	40244.68	41700.85	43917.65
08/24/2015	42616.81	42193.17	44523.91	41884.23	41758.59	42079.56
08/25/2015	38164.84	37125.84	44470.70	38398.86	38550.83	37615.75

Table 39: Prediction error percentages for August at 2015 8pm

Percentage Error August 2015, 8pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-1.22	-3.47	3.68	-2.51	-1.27
-3.00	-3.44	-3.46	-1.09	-3.41
-0.45	2.04	-14.78	5.37	3.86
4.55	1.13	-14.23	1.18	5.32
-7.70	-6.10	-18.50	-10.81	-6.93
-4.40	-1.85	-8.26	0.97	-0.84
-0.91	1.90	-11.42	5.27	2.49
0.21	4.59	-14.96	9.35	4.76
10.58	7.65	-9.36	6.91	9.79
6.48	7.36	9.84	5.23	6.04
2.62	2.30	3.24	7.79	2.74
0.46	0.90	3.55	-4.01	-0.51
-5.15	-3.80	-1.07	-3.52	-5.48
-4.10	-3.05	-8.80	-0.67	-4.52
3.06	6.76	-2.71	8.36	5.05
-1.28	-0.27	-5.81	0.46	0.76
-1.46	1.30	-18.22	-2.08	-2.49
2.06	3.39	-23.93	1.35	0.31

Table 40: Standardized error for each model at 8pm on August 2015

August 2015, 8pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1410.42	1443.51	3993.96	1732.84	1567.80
MSE	3170052.65	2940676.63	21609370.40	4588930.63	3426679.79
RMSE	1780.46	1714.84	4648.59	2142.18	1851.13
MAPE	3.40	3.45	10.12	4.19	3.79

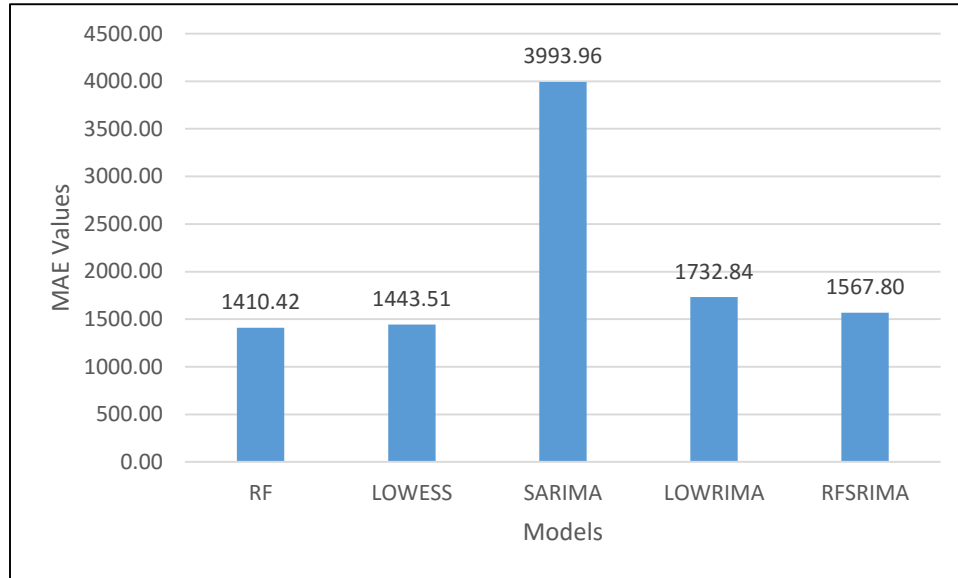


Figure 75: MAE values for Aug 2015 projection (MTLF 2012-2015) at 8pm

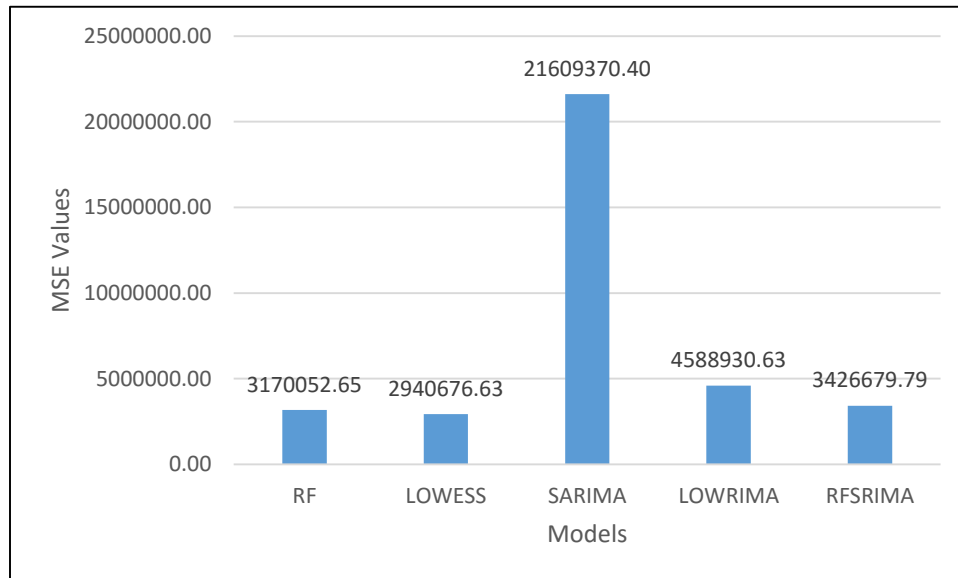


Figure 76: MSE values for Aug 2015 projection (MTLF 2012-2015) at 8pm

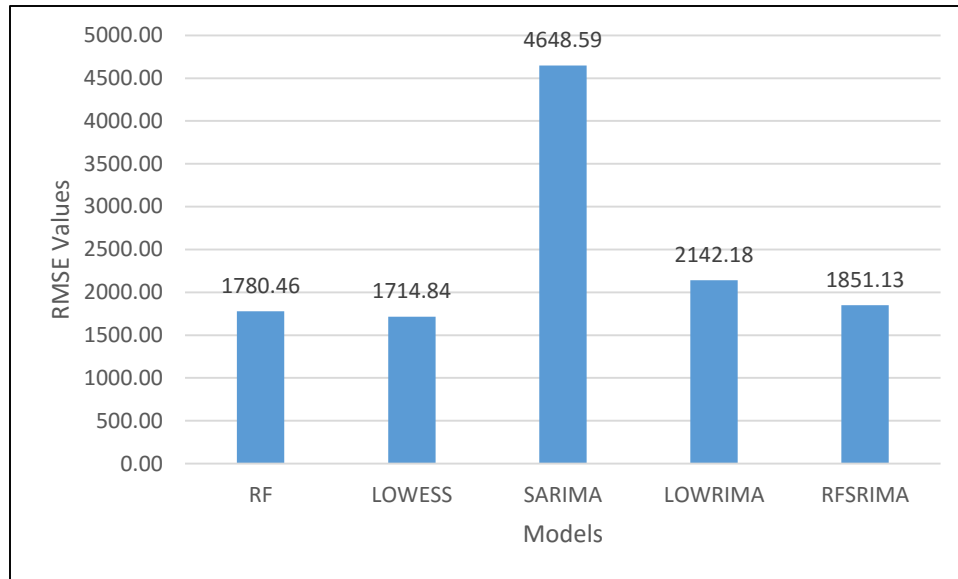


Figure 77: RMSE values for Aug 2015 projection (MTLF 2012-2015) at 8pm

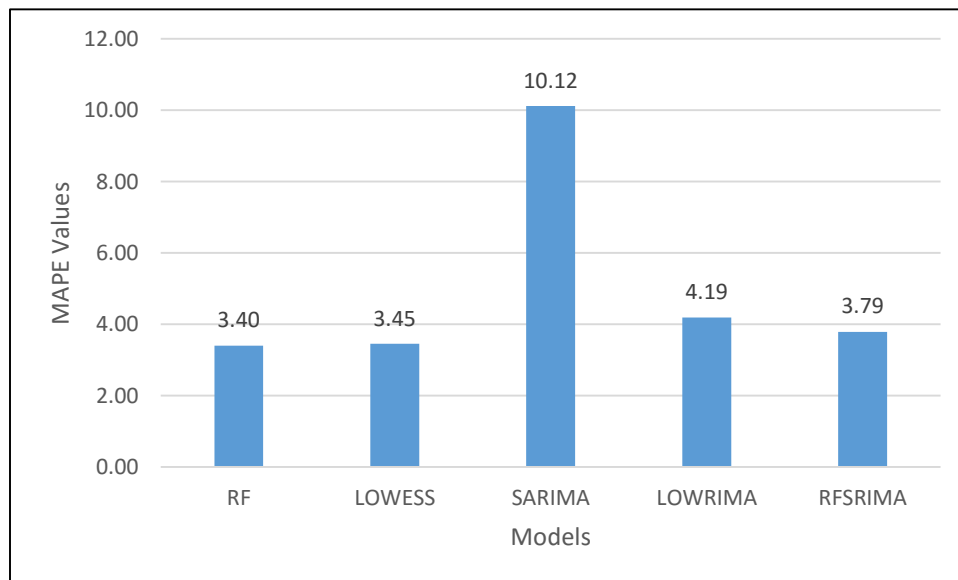


Figure 78: MAPE values for Aug 2015 projection (MTLF 2012-2015) at 8pm

Table 41: Demand data predicted by each model for November 2014 at 2am

November 2014, 2am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
11/02/2015	22867.29	21663.17	22827.06	22095.75	23209.75	23091.81
11/03/2015	23082.11	21704.85	22811.26	21633.44	22716.24	21960.04
11/04/2015	22897.73	22340.39	22472.42	22352.27	22678.36	22258.38
11/05/2015	23141.95	23880.11	23029.28	23884.74	23466.59	22717.78
11/06/2015	23169.09	24647.64	22931.33	24702.38	22907.01	23593.44
11/09/2015	23212.68	23010.74	23191.98	23093.36	22825.62	23507.23
11/10/2015	23140.25	21663.17	22812.19	21720.01	22758.36	22071.89
11/11/2015	25751.43	25937.76	22659.84	25924.84	26043.23	23476.53
11/12/2015	27567.85	27686.27	22829.55	27550.70	27588.92	25734.70
11/13/2015	24695.12	26119.62	22662.66	26153.78	25068.42	24726.81
11/16/2015	27760.76	28879.45	21944.05	29033.27	27691.36	26197.97
11/17/2015	28561.69	31153.58	22382.98	31246.90	28483.62	31645.70
11/18/2015	27199.73	27099.07	22926.14	27183.79	27259.35	29235.86
11/19/2015	27674.84	28879.45	23467.47	29072.03	27876.77	29147.20
11/20/2015	22370.88	22011.35	22958.26	22014.51	22355.42	23370.82
11/23/2015	22873.18	21663.17	22646.12	21641.98	22801.26	22332.79
11/24/2015	25972.57	27922.49	22143.44	27879.71	25788.60	24272.04
11/25/2015	27716.96	29844.54	22003.47	30001.91	27373.01	26525.79

Table 42: Prediction error percentages for November 2014 at 2am

Percentage Error November 2014, 2am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
0.97	6.19	1.15	4.31	-0.51
-5.11	1.16	-3.88	1.49	-3.44
-2.87	-0.37	-0.96	-0.42	-1.89
-1.87	-5.12	-1.37	-5.14	-3.30
1.80	-4.47	2.81	-4.70	2.91
1.25	2.11	1.34	1.76	2.90
-4.84	1.85	-3.35	1.59	-3.11
-9.69	-10.48	3.48	-10.43	-10.93
-7.12	-7.58	11.29	-7.06	-7.21
0.13	-5.63	8.35	-5.77	-1.38
-5.97	-10.24	16.24	-10.82	-5.70
9.75	1.56	29.27	1.26	9.99
6.96	7.31	21.58	7.02	6.76
5.05	0.92	19.49	0.26	4.36
4.28	5.82	1.77	5.80	4.34
-2.42	3.00	-1.40	3.09	-2.10
-7.01	-15.04	8.77	-14.86	-6.25
-4.49	-12.51	17.05	-13.10	-3.19

Table 43: Standardized error for each model at 2am on November 2014

November 2014, 2am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1162.47	1403.87	2334.89	1368.98	1145.00
MSE	1984041.00	3083418.55	12033747.96	3064036.97	1938359.34
RMSE	1408.56	1755.97	3468.97	1750.44	1392.25
MAPE	4.53	5.63	8.53	5.49	4.46

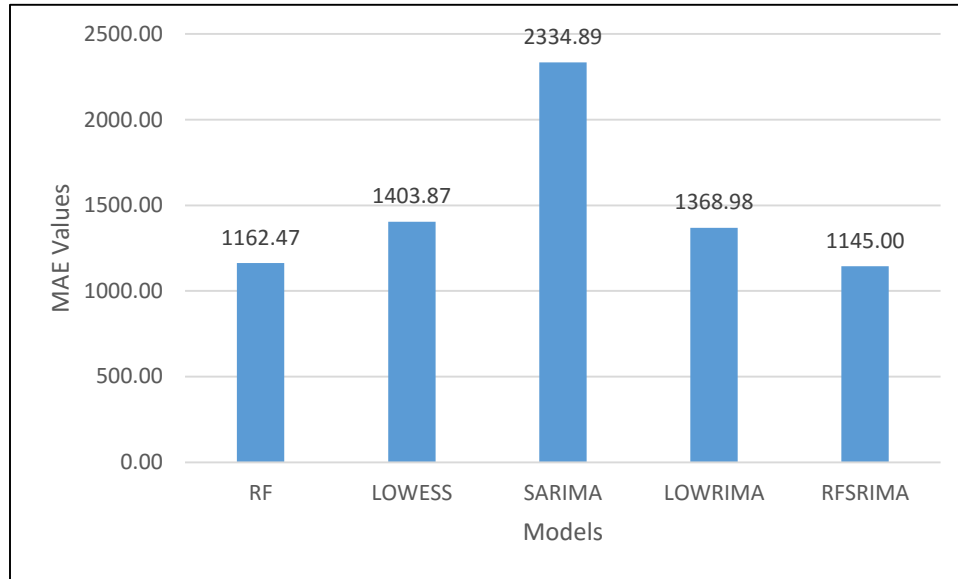


Figure 79: MAE values for Nov 2014 projection (MTLF 2012-2014) at 2am

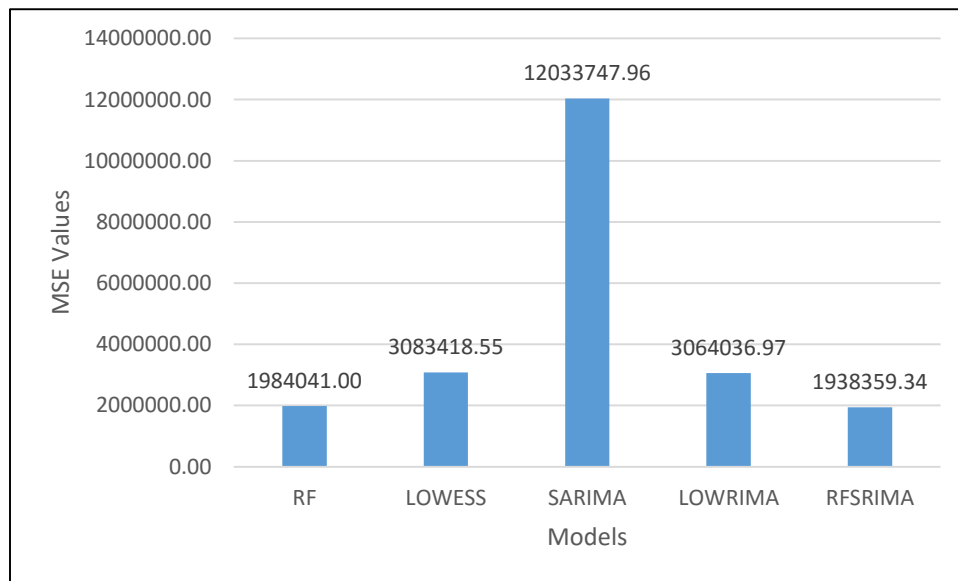


Figure 80: MSE values for Nov 2014 projection (MTLF 2012-2014) at 2am

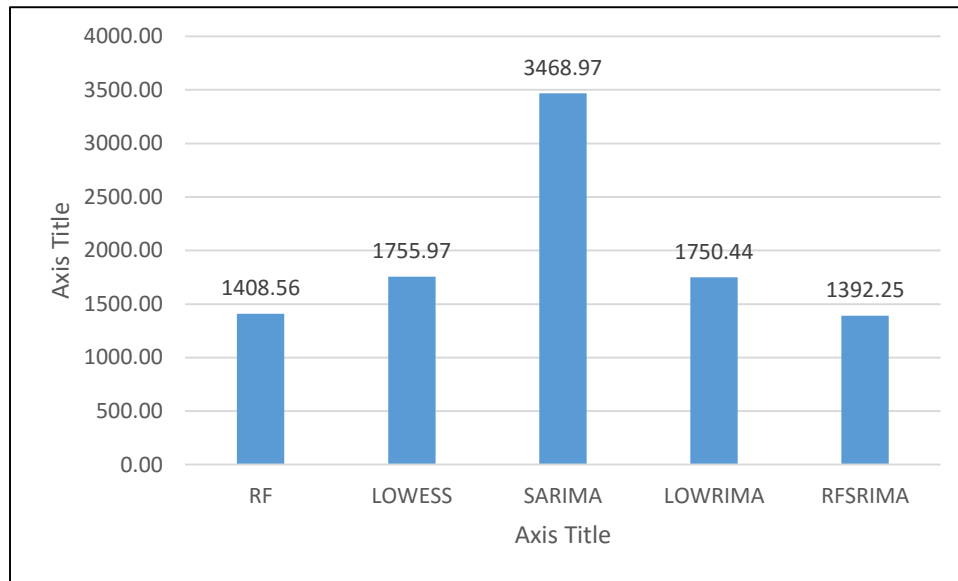


Figure 81: RMSE values for Nov 2014 projection (MTLF 2012-2014) at 2am

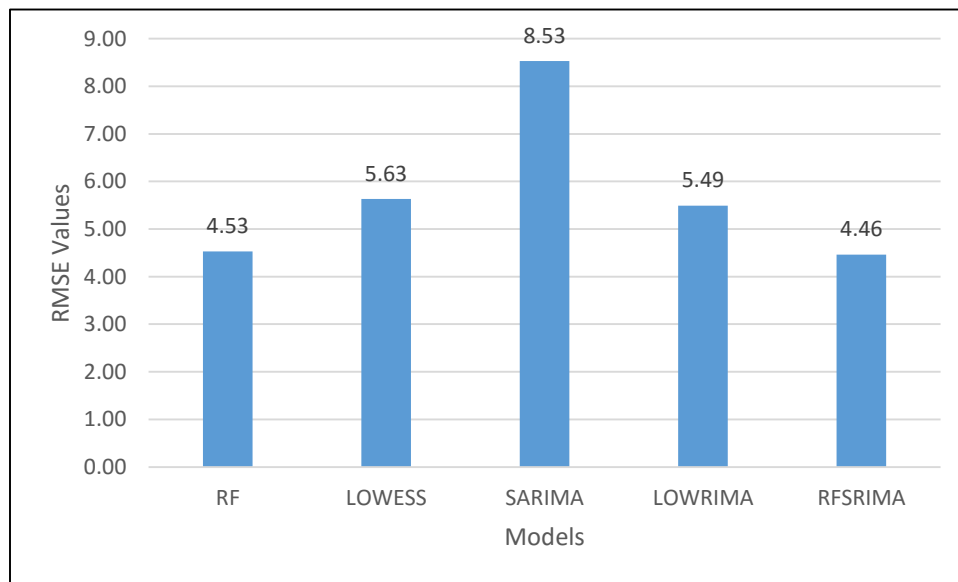


Figure 82: MAPE values for Nov 2014 projection (MTLF 2012-2014) at 2am

Table 44: Demand data predicted by each model for November 2014 at 9am

November 2014, 9am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
11/02/2015	30758.67	29798.67	30901.73	30383.55	30420.15	29543.60
11/03/2015	30618.93	29798.67	30963.28	29775.84	30074.01	29483.85
11/04/2015	30245.07	30606.33	30651.97	30591.10	29985.84	30356.34
11/05/2015	30648.33	29781.29	31385.52	29755.69	30823.35	29854.22
11/06/2015	31077.28	29842.69	31057.90	29815.58	30536.50	31405.07
11/09/2015	31351.37	28917.43	31363.46	28878.52	30965.03	30538.63
11/10/2015	30686.24	29308.63	30824.70	29227.43	30313.92	29564.36
11/11/2015	33570.29	29046.65	30971.62	29014.98	33008.18	31767.20
11/12/2015	34855.75	31568.83	31084.82	31572.37	34804.10	33352.17
11/13/2015	32944.17	31306.91	30683.11	31319.93	33243.43	33376.92
11/16/2015	34845.17	29781.29	30145.90	29775.47	34346.89	36324.65
11/17/2015	36657.76	29361.27	30625.38	29373.72	36637.46	38518.08
11/18/2015	34489.96	32146.11	31287.59	32130.38	34263.22	36119.24
11/19/2015	34408.03	29335.64	31430.83	29301.90	34264.29	37002.62
11/20/2015	30228.94	30175.73	31108.44	30125.43	30026.29	30457.96
11/23/2015	30770.81	30175.73	30450.65	30184.32	30625.86	29923.21
11/24/2015	33695.56	29335.64	30115.04	29318.37	33731.02	33236.54
11/25/2015	34668.08	30563.85	30194.92	30560.32	34267.59	31317.02

Table 45: Prediction error percentages for November 2014 at 9am

Percentage Error November 2014, 9am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
0.97	6.19	1.15	4.31	-0.51
-5.11	1.16	-3.88	1.49	-3.44
-2.87	-0.37	-0.96	-0.42	-1.89
-1.87	-5.12	-1.37	-5.14	-3.30
1.80	-4.47	2.81	-4.70	2.91
1.25	2.11	1.34	1.76	2.90
-4.84	1.85	-3.35	1.59	-3.11
-9.69	-10.48	3.48	-10.43	-10.93
-7.12	-7.58	11.29	-7.06	-7.21
0.13	-5.63	8.35	-5.77	-1.38
-5.97	-10.24	16.24	-10.82	-5.70
9.75	1.56	29.27	1.26	9.99
6.96	7.31	21.58	7.02	6.76
5.05	0.92	19.49	0.26	4.36
4.28	5.82	1.77	5.80	4.34
-2.42	3.00	-1.40	3.09	-2.10
-7.01	-15.04	8.77	-14.86	-6.25
-4.49	-12.51	17.05	-13.10	-3.19

Table 46: Standardized error for each model at 9am on November 2014

November 2014, 9am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1205.98	2413.06	2375.00	2460.69	1150.52
MSE	2132266.78	13188454.05	10469489.64	13280654.77	1965505.14
RMSE	1460.23	3631.59	3235.66	3644.26	1401.96
MAPE	3.67	6.87	6.89	7.03	3.46

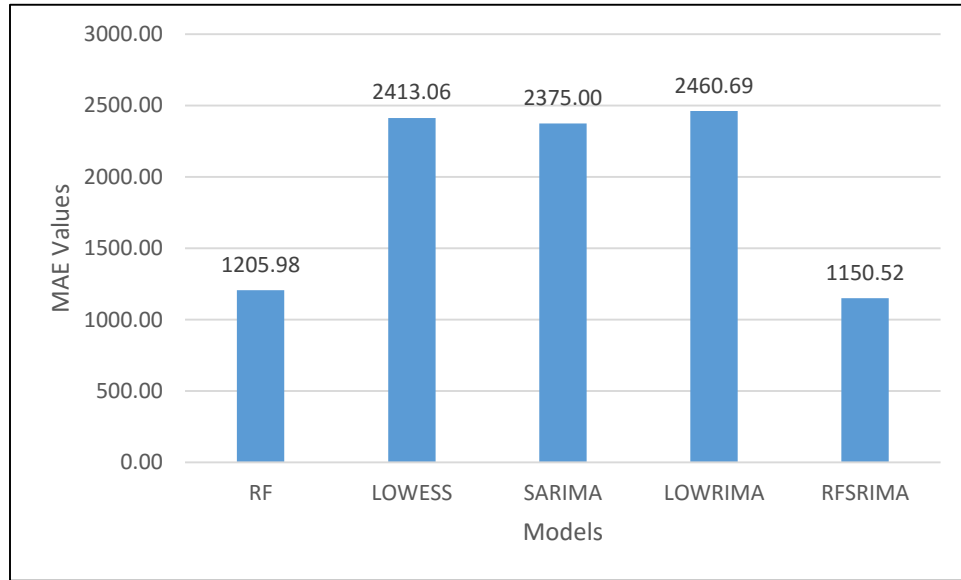


Figure 83: MAE values for Nov 2014 projection (MTLF 2012-2014) at 9am

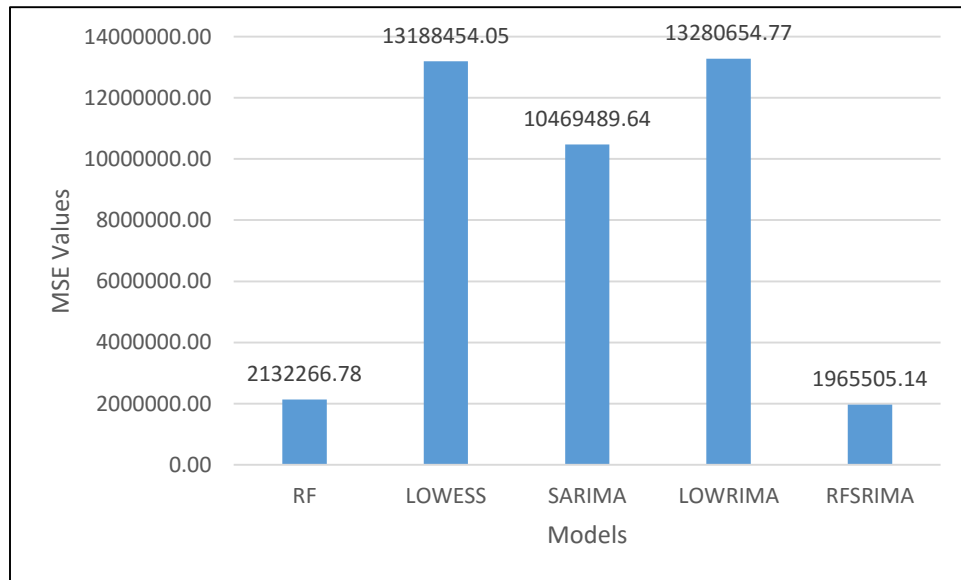


Figure 84: MSE values for Nov 2014 projection (MTLF 2012-2014) at 9am

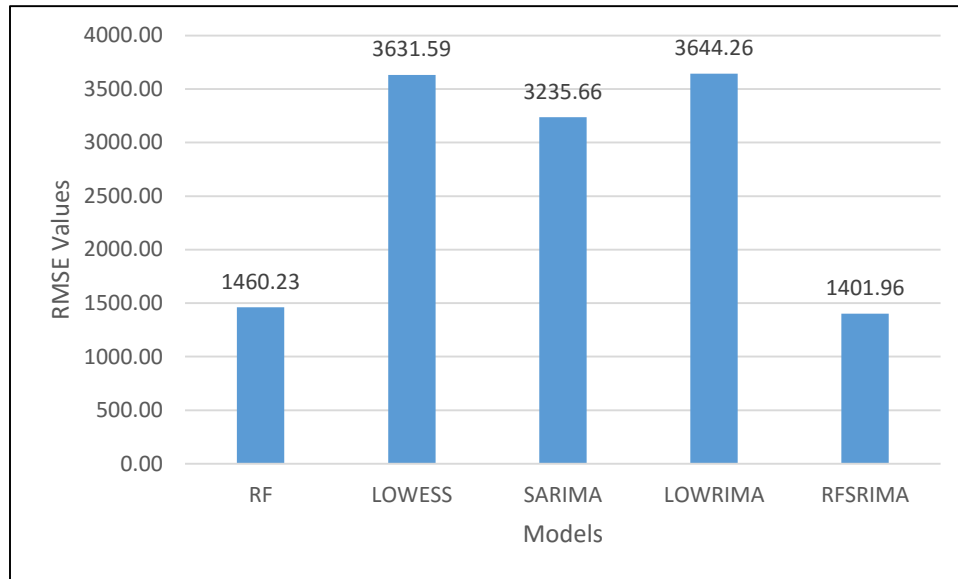


Figure 85: RMSE values for Nov 2014 projection (MTLF 2012-2014) at 9am

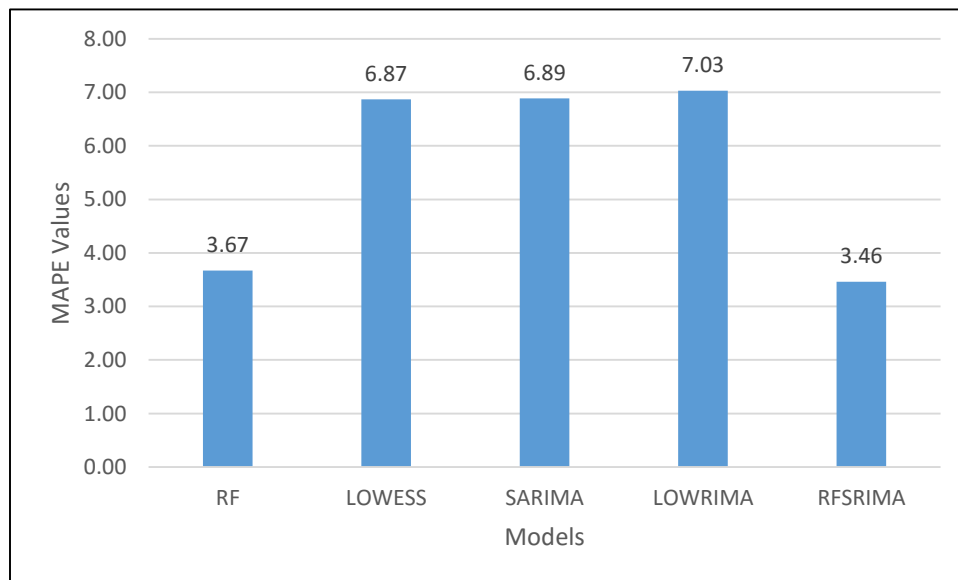


Figure 86: MAPE values for Nov 2014 projection (MTLF 2012-2014) at 9am

Table 47: Demand data predicted by each model for November 2014 at 2pm

November 2014, 2pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
11/02/2015	30542.97	29959.48	29597.63	30421.00	30076.00	29183.30
11/03/2015	30315.15	29622.86	29826.37	29507.91	30101.98	29580.40
11/04/2015	29949.16	29626.46	29960.49	29926.21	29676.19	30725.85
11/05/2015	29724.26	31146.11	30277.78	31214.41	29516.83	29784.36
11/06/2015	30166.43	31318.96	29781.48	31154.86	29754.31	29277.36
11/09/2015	30702.09	30386.62	29574.37	29959.80	30304.80	29390.65
11/10/2015	30305.50	29959.48	29672.71	29610.76	29961.28	29427.32
11/11/2015	32104.30	32854.08	29604.00	32402.56	31630.35	31839.55
11/12/2015	32559.08	33611.13	29801.80	33347.91	32260.55	31971.16
11/13/2015	30603.14	32132.71	28917.51	32198.93	30172.99	33890.08
11/16/2015	33665.63	35936.01	28482.23	35443.95	33289.21	34913.69
11/17/2015	34017.49	36965.93	28999.90	36750.87	33591.88	35140.88
11/18/2015	31791.96	33182.82	29994.86	32778.88	31342.75	32448.37
11/19/2015	32205.95	35936.01	30061.20	35627.98	31895.28	34217.79
11/20/2015	30418.85	31164.19	29566.32	30792.88	30266.13	29951.37
11/23/2015	30544.69	29959.48	28863.73	29601.64	30301.68	30366.90
11/24/2015	32060.59	33371.15	28479.77	33172.02	31698.21	35808.39
11/25/2015	32282.78	34490.12	28659.33	34223.32	31801.50	31089.97

Table 48: Prediction error percentages for November 2014 at 2pm

Percentage Error November 2014, 2pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-4.66	-2.66	-1.42	-4.24	-3.06
-2.48	-0.14	-0.83	0.25	-1.76
2.53	3.58	2.49	2.60	3.42
0.20	-4.57	-1.66	-4.80	0.90
-3.04	-6.97	-1.72	-6.41	-1.63
-4.46	-3.39	-0.63	-1.94	-3.11
-2.98	-1.81	-0.83	-0.62	-1.81
-0.83	-3.19	7.02	-1.77	0.66
-1.84	-5.13	6.79	-4.31	-0.91
9.70	5.19	14.67	4.99	10.97
3.57	-2.93	18.42	-1.52	4.65
3.20	-5.19	17.48	-4.58	4.41
2.02	-2.26	7.56	-1.02	3.41
5.88	-5.02	12.15	-4.12	6.79
-1.56	-4.05	1.29	-2.81	-1.05
-0.59	1.34	4.95	2.52	0.21
10.47	6.81	20.47	7.36	11.48
-3.84	-10.94	7.82	-10.08	-2.29

Table 49: Standardized error for each model at 2pm on November 2014

November 2014, 2pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1154.17	1334.39	2392.21	1169.91	1148.64
MSE	2250759.47	2384634.14	11124252.87	2011654.00	2594052.16
RMSE	1500.25	1544.23	3335.30	1418.33	1610.61
MAPE	3.55	4.18	7.12	3.66	3.47

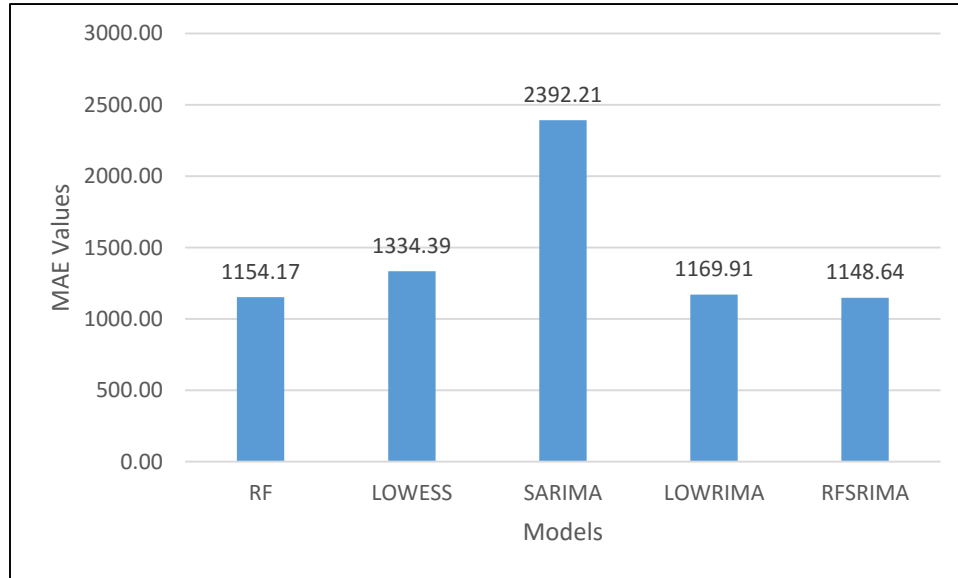


Figure 87: MAE values for Nov 2014 projection (MTLF 2012-2014) at 2pm

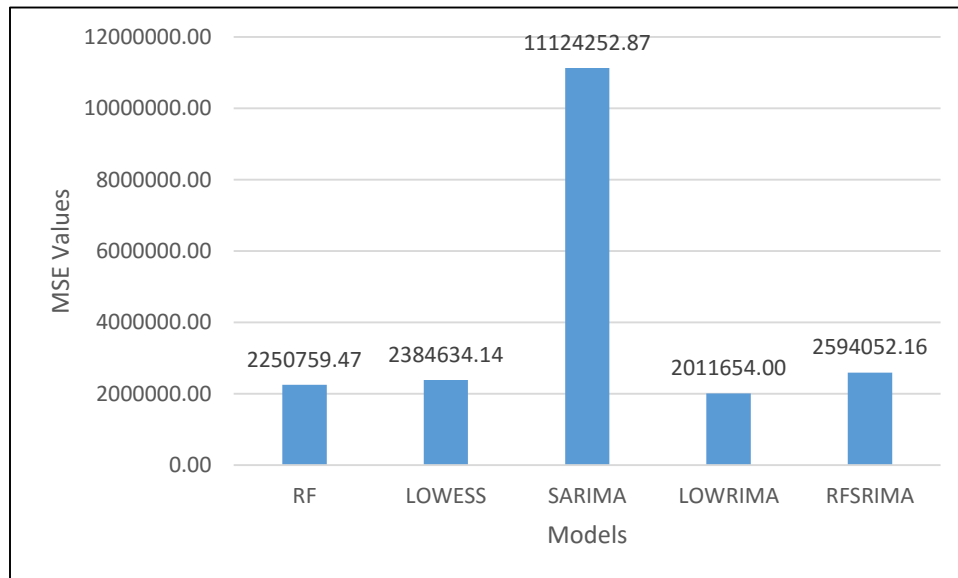


Figure 88: MSE values for Nov 2014 projection (MTLF 2012-2014) at 2pm

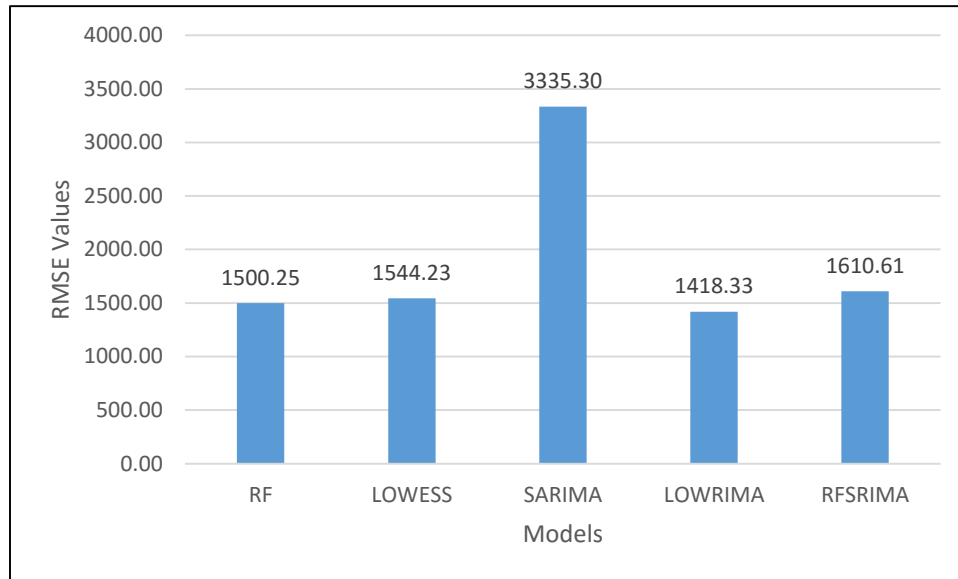


Figure 89: RMSE values for Nov 2014 projection (MTLF 2012-2014) at 2pm

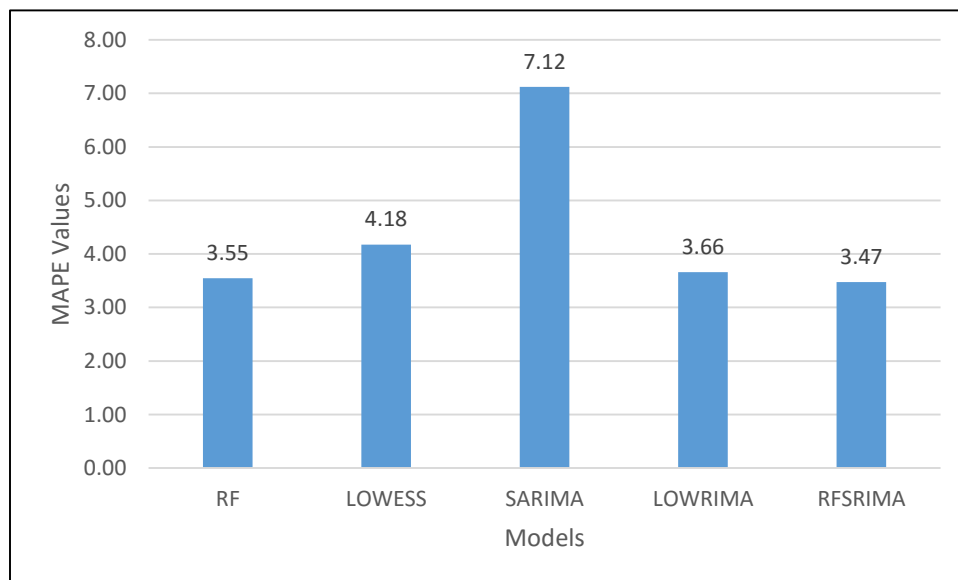


Figure 90: MAPE values for Nov 2014 projection (MTLF 2012-2014) at 2pm

Table 50: Demand data predicted by each model for November 2014 at 8pm

November 2014, 8pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
11/02/2015	33278.38	30943.53	30465.96	31341.87	32559.72	31567.32
11/03/2015	32661.88	30552.51	30594.55	30497.44	32130.58	31667.38
11/04/2015	32333.21	31695.88	30909.47	31696.90	31830.19	32197.91
11/05/2015	31375.39	33534.15	30987.43	33335.21	30943.01	32507.77
11/06/2015	33357.59	34037.44	30477.00	33906.85	33018.54	32281.97
11/09/2015	33509.96	32163.52	30470.75	32246.12	33281.65	31751.39
11/10/2015	32676.79	30943.53	30246.89	30845.14	32418.43	31879.96
11/11/2015	35080.74	36504.11	30234.74	36001.60	34999.51	35030.22
11/12/2015	35566.65	36718.83	30343.98	36247.93	35515.67	35032.28
11/13/2015	33923.18	35434.80	29406.09	35499.44	33548.17	34898.78
11/16/2015	37040.10	38841.56	29042.45	38704.87	36530.24	40804.53
11/17/2015	39203.41	42185.11	30019.06	42079.64	38683.34	39652.96
11/18/2015	35785.02	36755.52	30906.46	36502.43	35558.45	37060.68
11/19/2015	35685.43	38841.56	30685.23	38184.93	35409.72	37948.20
11/20/2015	32806.28	31637.98	29921.00	31190.66	32315.32	32312.28
11/23/2015	33277.92	30943.53	29268.57	30871.24	32851.37	33206.18
11/24/2015	35571.29	36002.27	28853.98	35946.47	35163.38	36482.67
11/25/2015	35614.37	37608.50	29762.63	37528.93	35185.76	35349.37

Table 51: Prediction error percentages for November 2014 at 8pm

Percentage Error November 2014, 8pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-4.66	-2.66	-1.42	-4.24	-3.06
-2.48	-0.14	-0.83	0.25	-1.76
2.53	3.58	2.49	2.60	3.42
0.20	-4.57	-1.66	-4.80	0.90
-3.04	-6.97	-1.72	-6.41	-1.63
-4.46	-3.39	-0.63	-1.94	-3.11
-2.98	-1.81	-0.83	-0.62	-1.81
-0.83	-3.19	7.02	-1.77	0.66
-1.84	-5.13	6.79	-4.31	-0.91
9.70	5.19	14.67	4.99	10.97
3.57	-2.93	18.42	-1.52	4.65
3.20	-5.19	17.48	-4.58	4.41
2.02	-2.26	7.56	-1.02	3.41
5.88	-5.02	12.15	-4.12	6.79
-1.56	-4.05	1.29	-2.81	-1.05
-0.59	1.34	4.95	2.52	0.21
10.47	6.81	20.47	7.36	11.48
-3.84	-10.94	7.82	-10.08	-2.29

Table 52: Standardized error for each model at 8pm on November 2014

November 2014, 8pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1036.63	1190.98	4390.87	1119.98	1065.75
MSE	1867343.43	1913055.89	28970117.15	1750345.12	2156637.27
RMSE	1366.51	1383.13	5382.39	1323.01	1468.55
MAPE	2.95	3.41	12.11	3.21	2.97

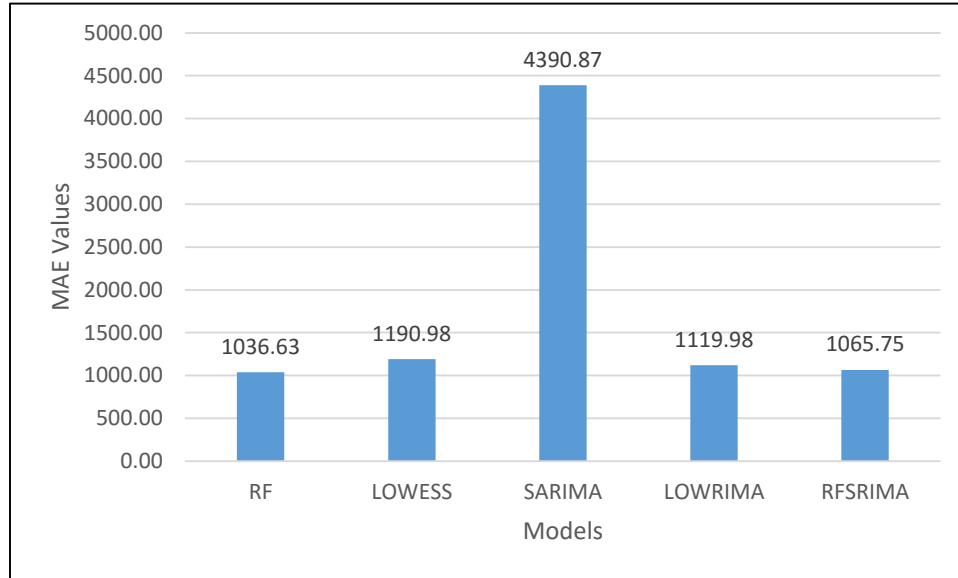


Figure 91: MAE values for Nov 2014 projection (MTLF 2012-2014) at 8pm

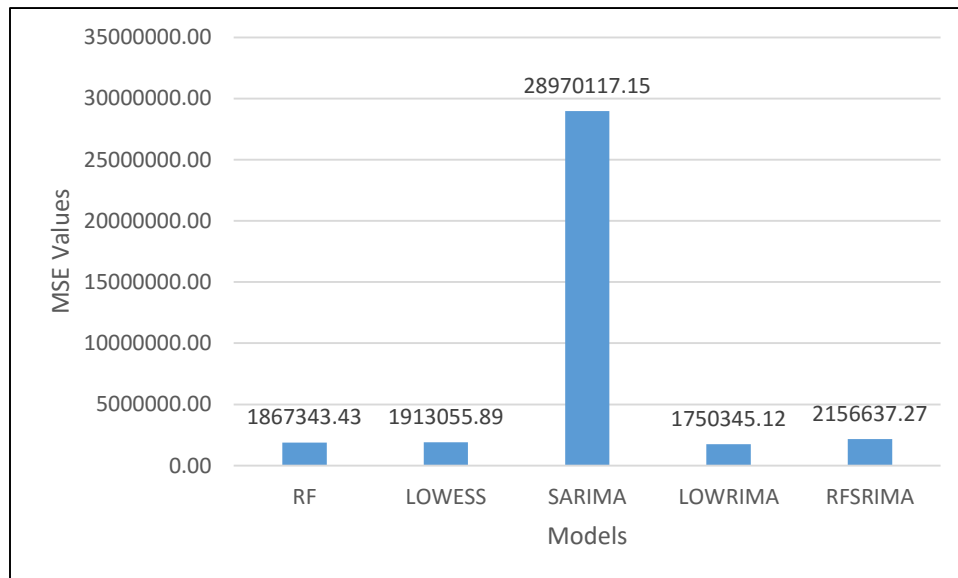


Figure 92: MSE values for Nov 2014 projection (MTLF 2012-2014) at 8pm

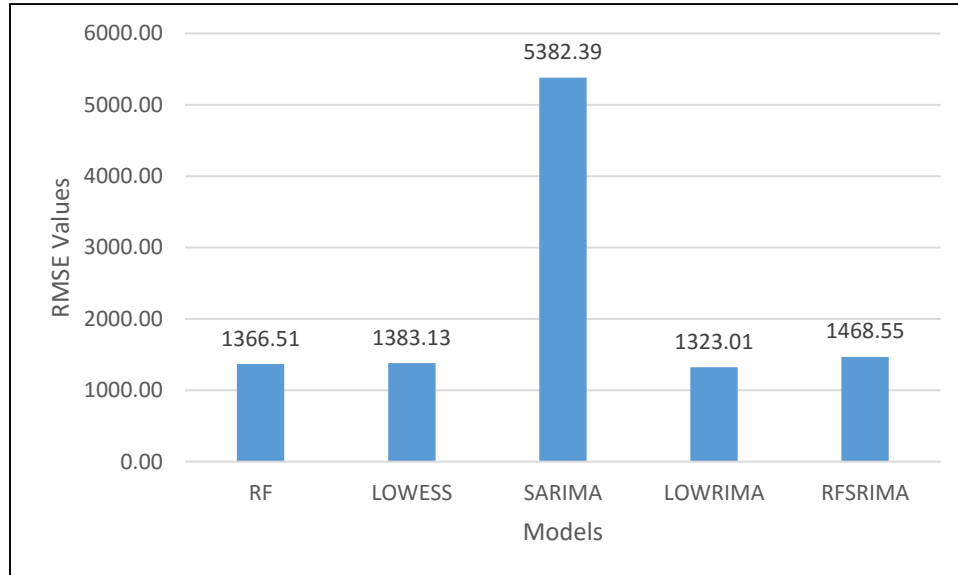


Figure 93: RMSE values for Nov 2014 projection (MTLF 2012-2014) at 8pm

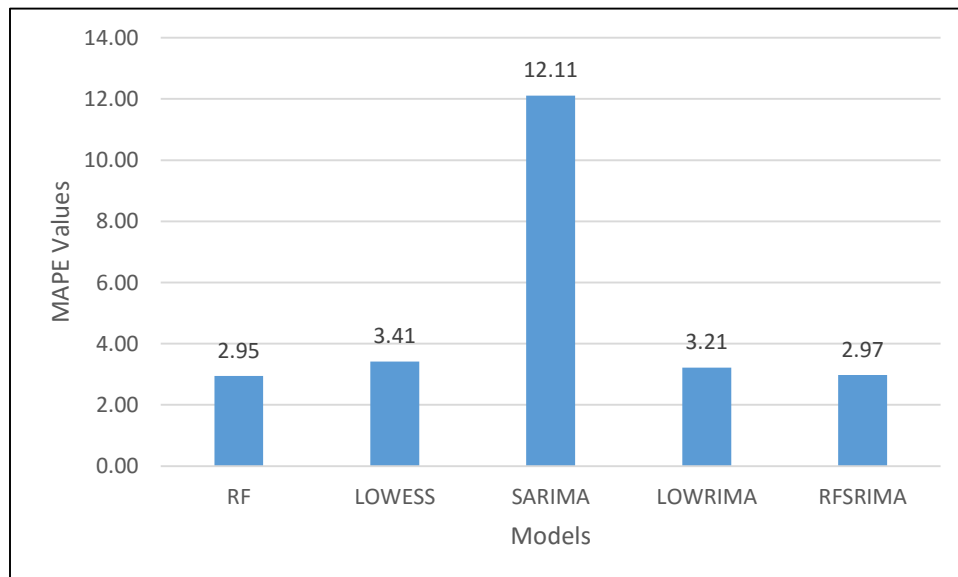


Figure 94: MAPE values for Nov 2014 projection (MTLF 2012-2014) at 8pm

Table 53: Demand data predicted by each model for December 2014 at 2am

December 2014, 2am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
12/01/2015	27245.21	26826.23	27310.67	26827.52	27138.76	27403.36
12/02/2015	28931.03	29798.84	26200.41	29767.93	29220.76	28217.01
12/03/2015	26528.26	27346.78	26648.14	26811.84	26826.67	28859.91
12/04/2015	26535.76	27580.79	27082.22	27290.53	26106.82	26914.96
12/07/2015	28651.29	30283.52	27335.84	30088.06	28277.93	28756.94
12/08/2015	28270.23	27612.92	27393.23	27926.76	27910.40	28792.23
12/09/2015	25787.18	25928.63	26294.02	25752.38	25734.83	25623.00
12/10/2015	26102.39	26357.82	27520.55	26247.60	26104.93	27201.90
12/11/2015	26109.11	24934.80	29426.56	24750.55	25525.83	25496.38
12/14/2015	27551.42	27267.49	28904.89	27012.65	26657.02	27514.49
12/15/2015	28551.78	27885.39	28899.11	27742.24	27818.38	28117.96
12/16/2015	27670.45	27885.39	31652.03	27633.12	27375.73	29136.42
12/17/2015	25832.62	25217.93	30526.55	24681.47	25388.94	26831.72
12/18/2015	24380.54	22958.56	30366.74	22758.39	25946.19	24718.15
12/21/2015	27035.94	27580.79	27678.96	27476.38	27855.14	24329.31
12/22/2015	25851.31	26357.82	27109.45	26388.80	26615.16	24212.34
12/23/2015	28976.78	30283.52	27480.41	30396.98	29380.54	27213.18
12/24/2015	31100.53	30153.72	29017.82	29981.25	30644.46	29996.15

Table 54: Prediction error percentages for December 2014 at 2am

Percentage Error December 2014, 2am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
0.58	2.11	0.34	2.10	0.97
-2.53	-5.61	7.15	-5.50	-3.56
8.08	5.24	7.66	7.10	7.05
1.41	-2.47	-0.62	-1.40	3.00
0.37	-5.31	4.94	-4.63	1.67
1.81	4.10	4.86	3.01	3.06
-0.64	-1.19	-2.62	-0.50	-0.44
4.04	3.10	-1.17	3.51	4.03
-2.40	2.20	-15.41	2.93	-0.12
-0.13	0.90	-5.05	1.82	3.12
-1.54	0.83	-2.78	1.34	1.07
5.03	4.29	-8.63	5.16	6.04
3.72	6.01	-13.77	8.01	5.38
1.37	7.12	-22.85	7.93	-4.97
-11.12	-13.36	-13.77	-12.94	-14.49
-6.77	-8.86	-11.97	-8.99	-9.92
-6.48	-11.28	-0.98	-11.70	-7.96
-3.68	-0.53	3.26	0.05	-2.16

Table 55: Standardized error for each model at 2am on December 2014

December 2014, 2am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	893.18	1209.41	1813.25	1261.18	1124.14
MSE	1345900.27	2227070.42	5314607.20	2434353.19	1966240.44
RMSE	1160.13	1492.34	2305.34	1560.24	1402.23
MAPE	3.35	4.57	6.89	4.76	4.23

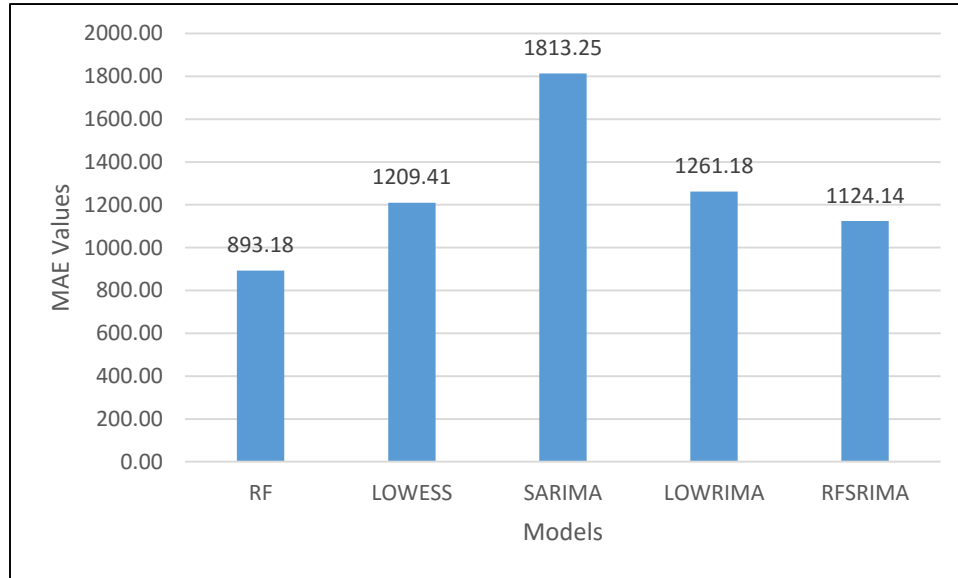


Figure 95: MAE values for Dec 2014 projection (MTLF 2012-2014) at 2am

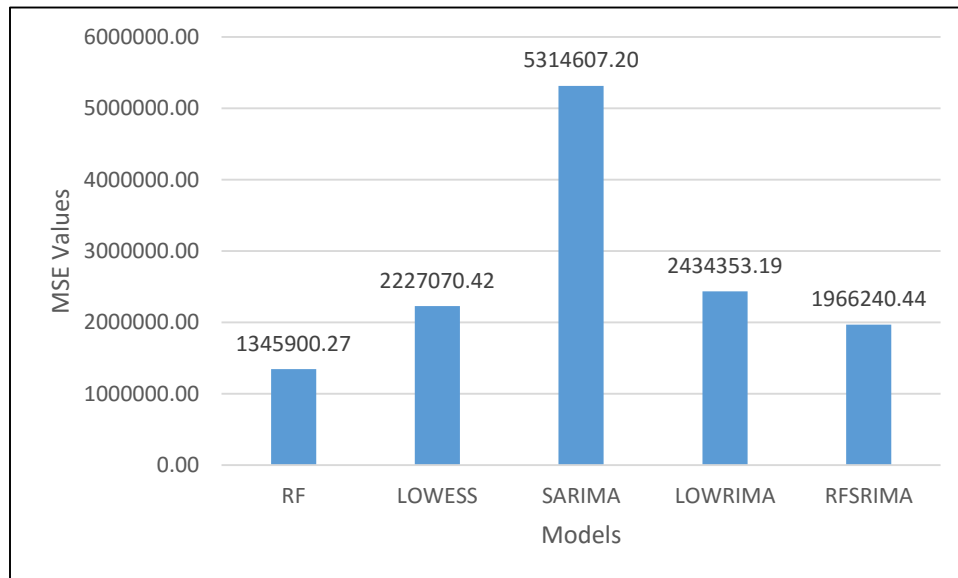


Figure 96: MSE values for Dec 2014 projection (MTLF 2012-2014) at 2am

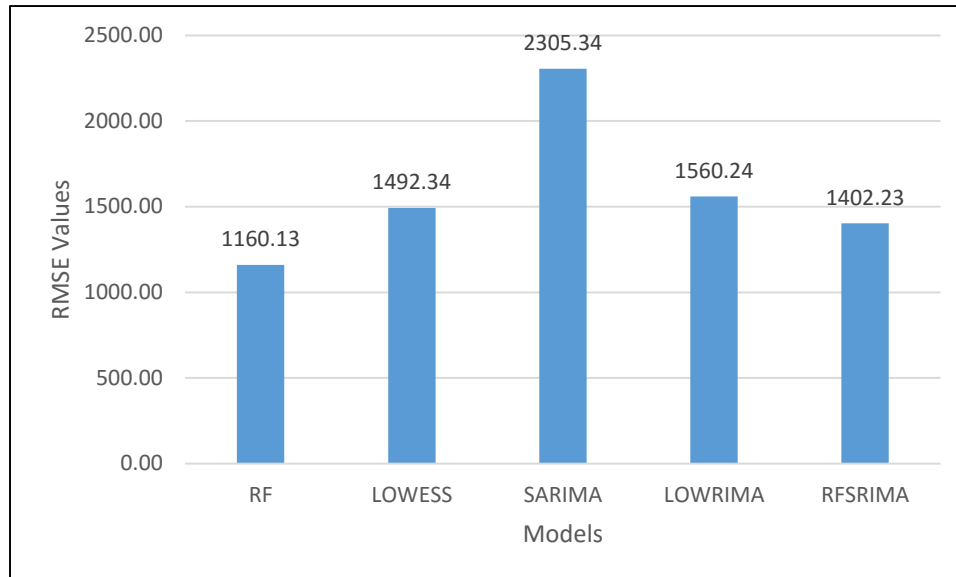


Figure 97: RMSE values for Dec 2014 projection (MTLF 2012-2014) at 2am

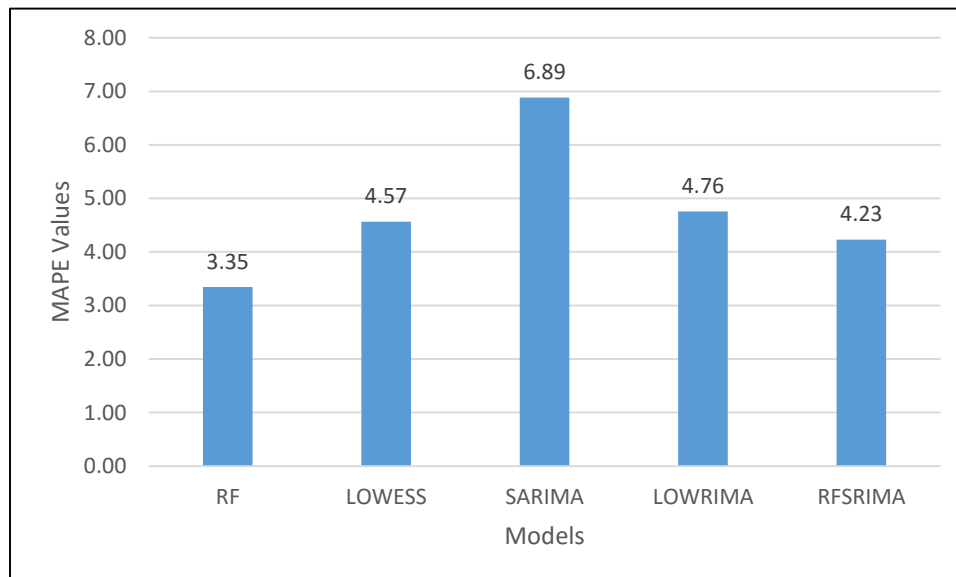


Figure 98: MAPE values for Dec 2014 projection (MTLF 2012-2014) at 2am

Table 56: Demand data predicted by each model for December 2014 at 9am

December 2014, 9am						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
12/01/2015	35091.42	29781.29	32387.11	29755.69	35283.99	35273.24
12/02/2015	36204.86	29842.69	32783.71	29815.58	36417.97	37981.81
12/03/2015	34626.15	28917.43	31476.33	28878.52	34386.57	37002.87
12/04/2015	34615.60	29308.63	31203.09	29227.43	34141.39	35114.92
12/07/2015	36270.12	29046.65	31521.23	29014.98	36104.13	36926.94
12/08/2015	35904.64	31568.83	31125.01	31572.37	35340.93	36570.08
12/09/2015	33846.59	31306.91	31898.98	31319.93	33657.15	34151.59
12/10/2015	34029.38	29781.29	31590.29	29775.47	33589.54	35354.07
12/11/2015	34115.47	29361.27	30807.64	29373.72	33341.50	33327.45
12/14/2015	35867.70	32146.11	32192.00	32130.38	35039.18	36046.44
12/15/2015	36053.87	29335.64	32918.69	29301.90	35509.38	36447.65
12/16/2015	35466.28	30175.73	33585.57	30125.43	35227.98	37479.88
12/17/2015	33797.59	30175.73	35982.74	30184.32	34511.99	34030.44
12/18/2015	32371.69	29335.64	36660.09	29318.37	33405.82	30713.83
12/21/2015	34653.54	30563.85	34576.06	30560.32	35054.93	30597.25
12/22/2015	33846.32	29361.27	35930.28	29357.59	35177.03	32310.69
12/23/2015	36321.57	29361.27	31688.18	29345.39	36380.29	34727.64
12/24/2015	39079.32	30942.99	31568.24	30892.32	38848.84	36254.55

Table 57: Prediction error percentages for December 2014 at 9am

Percentage Error December 2014, 9am				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
0.52	15.57	8.18	15.64	-0.03
4.68	21.43	13.69	21.50	4.12
6.42	21.85	14.94	21.96	7.07
1.42	16.54	11.14	16.77	2.77
1.78	21.34	14.64	21.43	2.23
1.82	13.68	14.89	13.67	3.36
0.89	8.33	6.60	8.29	1.45
3.75	15.76	10.65	15.78	4.99
-2.36	11.90	7.56	11.86	-0.04
0.50	10.82	10.69	10.86	2.79
1.08	19.51	9.68	19.61	2.57
5.37	19.49	10.39	19.62	6.01
0.68	11.33	-5.74	11.30	-1.42
-5.40	4.49	-19.36	4.54	-8.76
-13.26	0.11	-13.00	0.12	-14.57
-4.75	9.13	-11.20	9.14	-8.87
-4.59	15.45	8.75	15.50	-4.76
-7.79	14.65	12.93	14.79	-7.16

Table 58: Standardized error for each model at 9am on December 2014

December 2014, 9am					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1117.66	4953.93	3954.64	4945.20	1462.08
MSE	2327139.71	28798606.93	16809136.29	28789836.06	3310208.48
RMSE	1525.50	5366.43	4099.89	5365.62	1819.40
MAPE	3.25	13.89	11.31	13.87	4.26

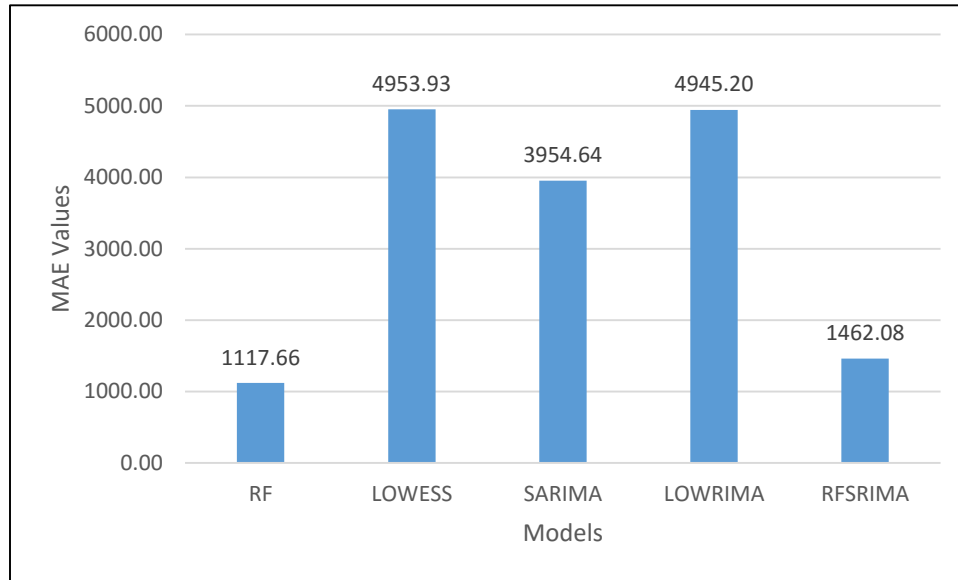


Figure 99: MAE values for Dec 2014 projection (MTLF 2012-2014) at 9am

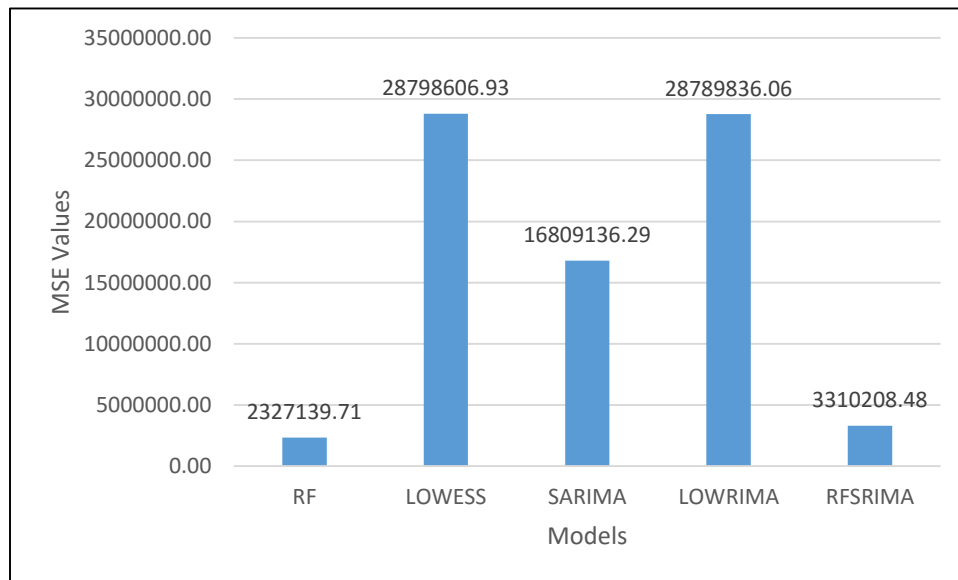


Figure 100: MSE values for Dec 2014 projection (MTLF 2012-2014) at 9am

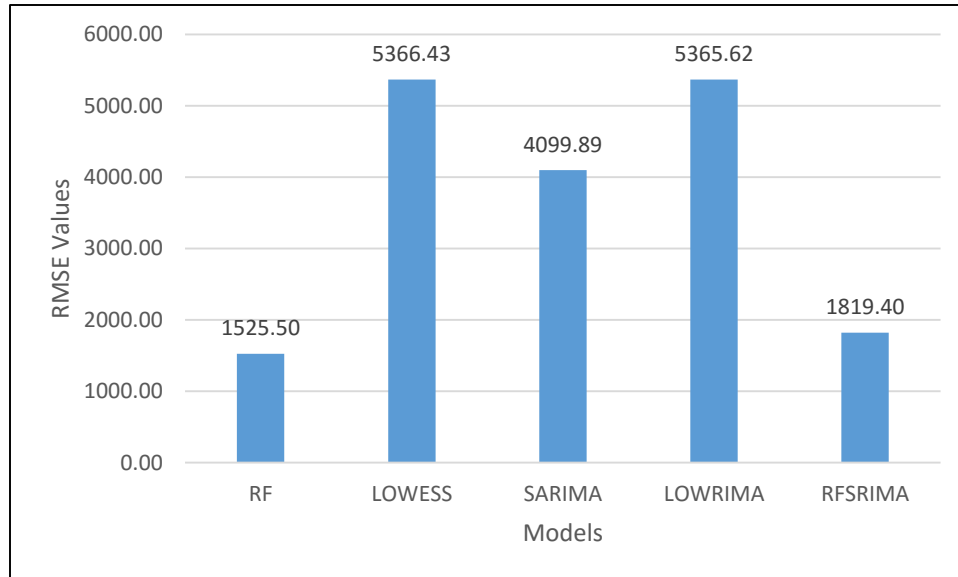


Figure 101: RMSE values for Dec 2014 projection (MTLF 2012-2014) at 9am

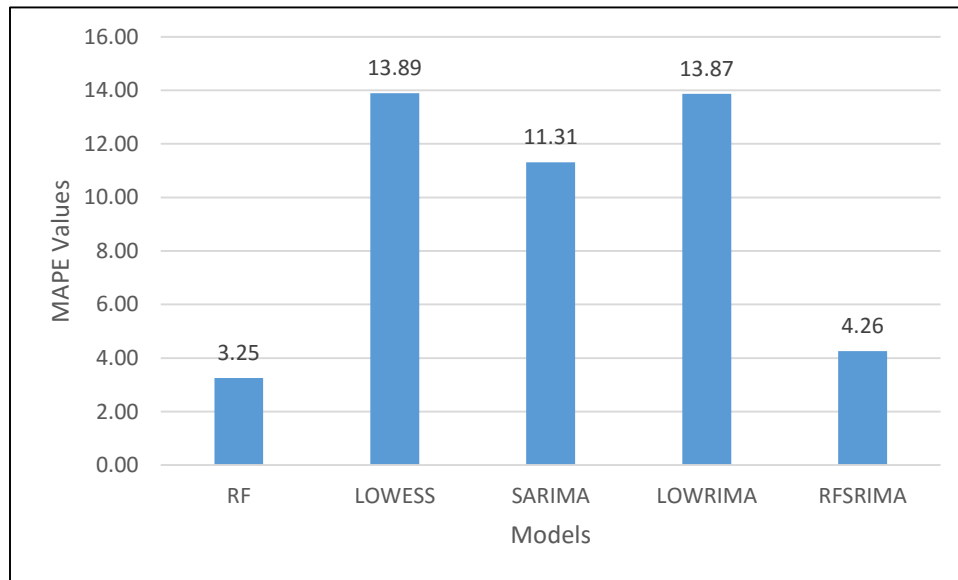


Figure 102: MAPE values for Dec 2014 projection (MTLF 2012-2014) at 9am

Table 59: Demand data predicted by each model for December 2014 at 2pm

December 2014, 2pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
12/01/2015	33360.03	32576.83	32309.14	32845.97	33290.14	34523.76
12/02/2015	35083.00	34392.21	32512.68	34815.22	34967.83	37051.94
12/03/2015	33071.50	33096.48	31450.72	33512.59	32264.66	35728.86
12/04/2015	32863.85	33225.20	31346.37	33638.41	32489.51	34597.13
12/07/2015	34309.98	35246.08	31632.31	35421.62	34911.64	36438.50
12/08/2015	33390.28	33234.04	30539.22	33200.86	33374.61	34707.38
12/09/2015	32882.69	32696.90	30652.30	32813.12	32294.38	31713.79
12/10/2015	32120.25	32055.95	30879.87	32350.75	31391.18	33729.23
12/11/2015	32098.11	30978.38	30421.35	31424.01	31597.42	31930.01
12/14/2015	33432.69	34065.51	31580.92	34077.26	33397.90	34381.85
12/15/2015	34064.54	34289.81	32052.74	34474.57	33813.24	34410.83
12/16/2015	35044.73	34289.81	34090.73	34475.04	36943.82	35482.02
12/17/2015	31845.76	31209.16	35395.38	31346.60	32511.63	32991.19
12/18/2015	31681.40	30002.40	34157.00	29980.03	32603.28	31478.16
12/21/2015	32919.47	33225.20	32335.36	33245.84	33354.65	28488.58
12/22/2015	32655.80	32055.95	32822.92	32129.44	34214.15	31728.97
12/23/2015	35847.78	35246.08	30751.51	35140.98	35692.39	33213.19
12/24/2015	36193.67	36787.03	31454.24	36859.57	36455.51	33946.66

Table 60: Prediction error percentages for December 2014 at 2pm

Percentage Error December 2014, 2pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
3.37	5.64	6.41	4.86	3.57
5.31	7.18	12.25	6.04	5.62
7.44	7.37	11.97	6.20	9.70
5.01	3.97	9.40	2.77	6.09
5.84	3.27	13.19	2.79	4.19
3.79	4.25	12.01	4.34	3.84
-3.69	-3.10	3.35	-3.47	-1.83
4.77	4.96	8.45	4.09	6.93
-0.53	2.98	4.72	1.58	1.04
2.76	0.92	8.15	0.89	2.86
1.01	0.35	6.85	-0.19	1.74
1.23	3.36	3.92	2.84	-4.12
3.47	5.40	-7.29	4.98	1.45
-0.65	4.69	-8.51	4.76	-3.57
-15.55	-16.63	-13.50	-16.70	-17.08
-2.92	-1.03	-3.45	-1.26	-7.83
-7.93	-6.12	7.41	-5.80	-7.46
-6.62	-8.37	7.34	-8.58	-7.39

Table 61: Standardized error for each model at 2pm on December 2014

December 2014, 2pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1545.15	1643.34	2834.53	1464.00	1777.12
MSE	3437193.63	3916358.95	9356318.83	3221928.58	4515399.55
RMSE	1853.97	1978.98	3058.81	1794.97	2124.95
MAPE	4.62	4.92	8.34	4.41	5.31

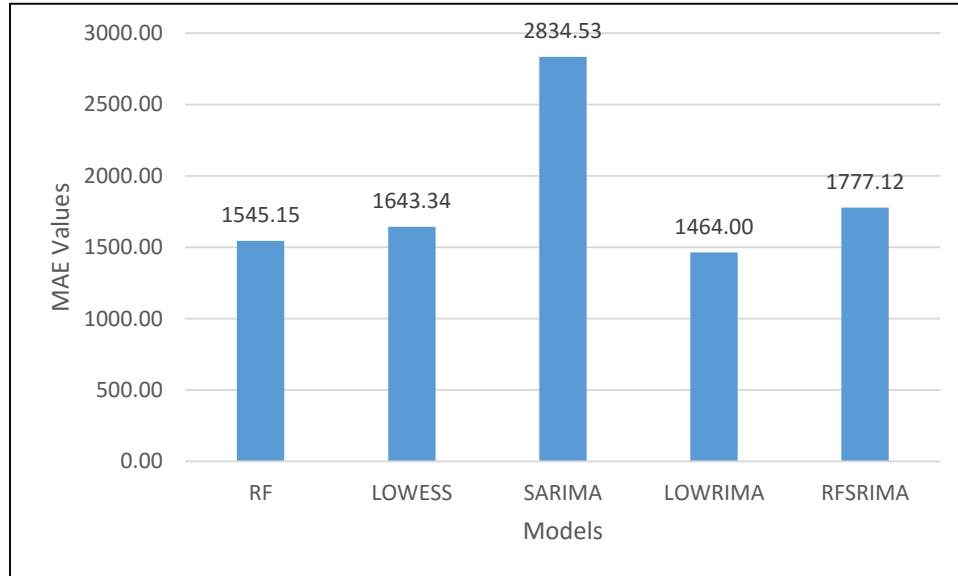


Figure 103: MAE values for Dec 2014 projection (MTLF 2012-2014) at 2pm

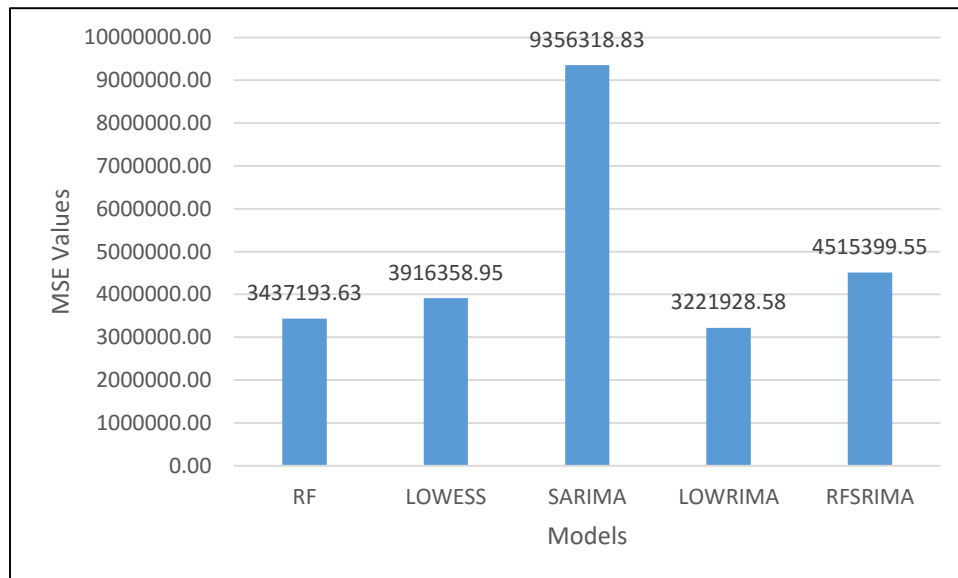


Figure 104: MSE values for Dec 2014 projection (MTLF 2012-2014) at 2pm

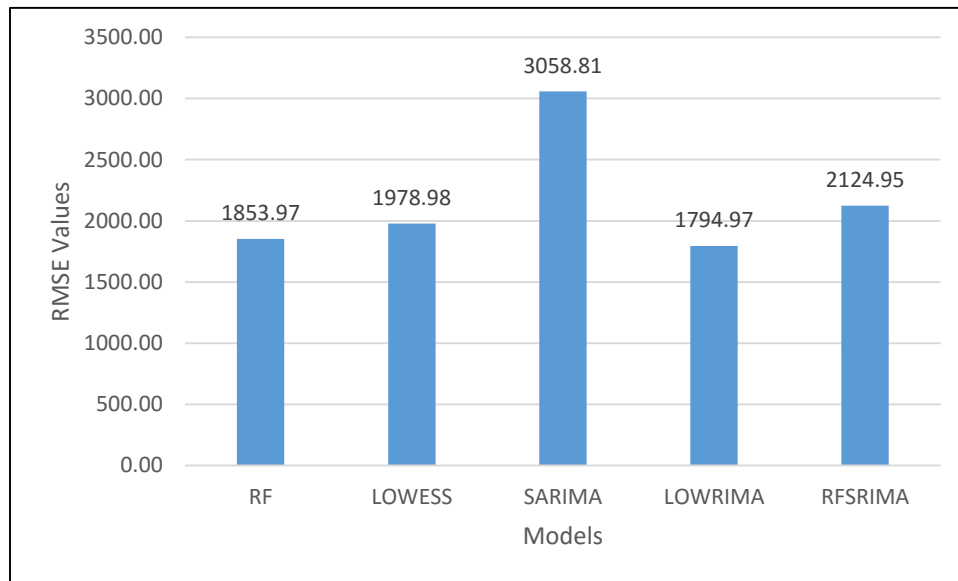


Figure 105: RMSE values for Dec 2014 projection (MTLF 2012-2014) at 2pm

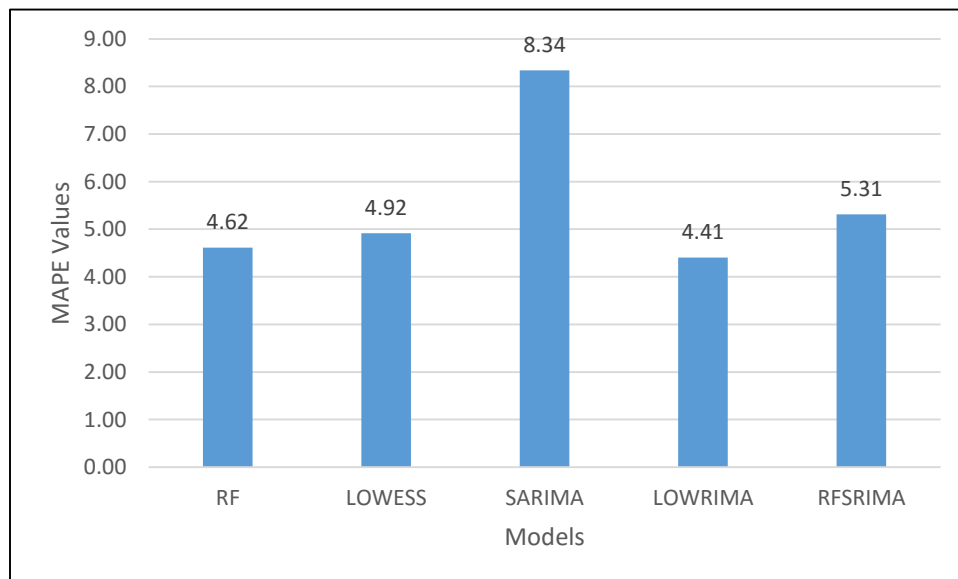


Figure 106: MAPE values for Dec 2014 projection (MTLF 2012-2014) at 2pm

Table 62: Demand data predicted by each model for December 2014 at 8pm

December 2014, 8pm						
Date	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA	Actual
12/01/2015	36022.59	35708.89	36613.55	36110.92	35944.61	35889.45
12/02/2015	38354.10	37451.04	38098.67	37702.35	38387.40	40348.95
12/03/2015	37008.87	36745.26	36897.25	37027.64	36826.45	38125.92
12/04/2015	36875.47	35997.43	36702.55	36292.65	36857.00	39067.82
12/07/2015	37898.34	38176.95	36350.04	38828.08	37986.52	39639.85
12/08/2015	36592.36	36618.21	36796.87	36768.61	37042.05	37497.10
12/09/2015	36779.20	36398.80	36829.03	36530.96	36524.57	36992.45
12/10/2015	36316.89	35005.80	36539.58	35449.40	35614.76	36549.31
12/11/2015	36088.18	33695.45	36407.78	34253.64	35920.56	36780.06
12/14/2015	38078.52	37623.42	37492.77	37810.86	37997.68	38764.22
12/15/2015	36814.76	37400.12	38349.69	37716.43	36778.18	38092.74
12/16/2015	37825.75	37400.12	39217.87	38028.88	38125.90	38320.65
12/17/2015	36105.97	34132.18	42758.51	34543.17	37122.92	35032.96
12/18/2015	35555.90	31020.37	41062.26	31182.61	36020.56	30308.43
12/21/2015	35785.48	35997.43	38886.12	36224.49	36305.71	33083.28
12/22/2015	36571.88	35005.80	40023.69	35230.61	37457.68	36206.49
12/23/2015	39239.29	38176.95	36725.68	38308.79	39184.24	38277.09
12/24/2015	41501.51	41951.80	37495.93	42347.81	42044.84	37774.51

Table 63: Prediction error percentages for December 2014 at 8pm

Percentage Error December 2014, 8pm				
RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
-0.37	0.50	-2.02	-0.62	-0.15
4.94	7.18	5.58	6.56	4.86
2.93	3.62	3.22	2.88	3.41
5.61	7.86	6.05	7.10	5.66
4.39	3.69	8.30	2.05	4.17
2.41	2.34	1.87	1.94	1.21
0.58	1.60	0.44	1.25	1.26
0.64	4.22	0.03	3.01	2.56
1.88	8.39	1.01	6.87	2.34
1.77	2.94	3.28	2.46	1.98
3.35	1.82	-0.67	0.99	3.45
1.29	2.40	-2.34	0.76	0.51
-3.06	2.57	-22.05	1.40	-5.97
-17.31	-2.35	-35.48	-2.88	-18.85
-8.17	-8.81	-17.54	-9.49	-9.74
-1.01	3.32	-10.54	2.70	-3.46
-2.51	0.26	4.05	-0.08	-2.37
-9.87	-11.06	0.74	-12.11	-11.30

Table 64: Standardized error for each model at 8pm on December 2014

December 2014, 8pm					
Err	RF	LOWESS	SARIMA	LOWRIMA	RFSRIMA
MAE	1327.66	1571.14	2219.68	1303.67	1510.73
MSE	3352170.43	3615797.23	12243459.41	2995347.34	4214191.16
RMSE	1830.89	1901.52	3499.07	1730.71	2052.85
MAPE	3.70	4.22	6.36	3.52	4.23

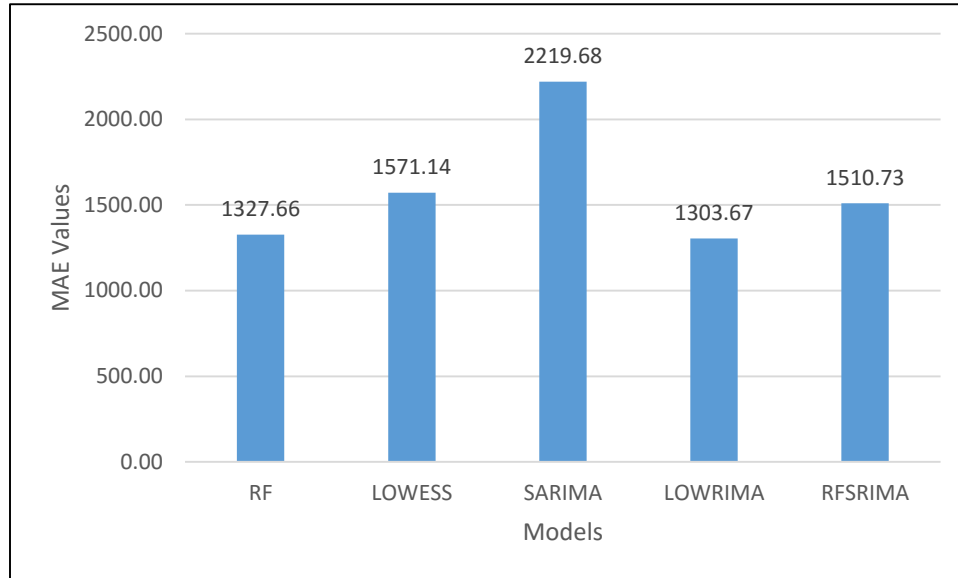


Figure 107: MAE values for Dec 2014 projection (MTLF 2012-2014) at 8pm

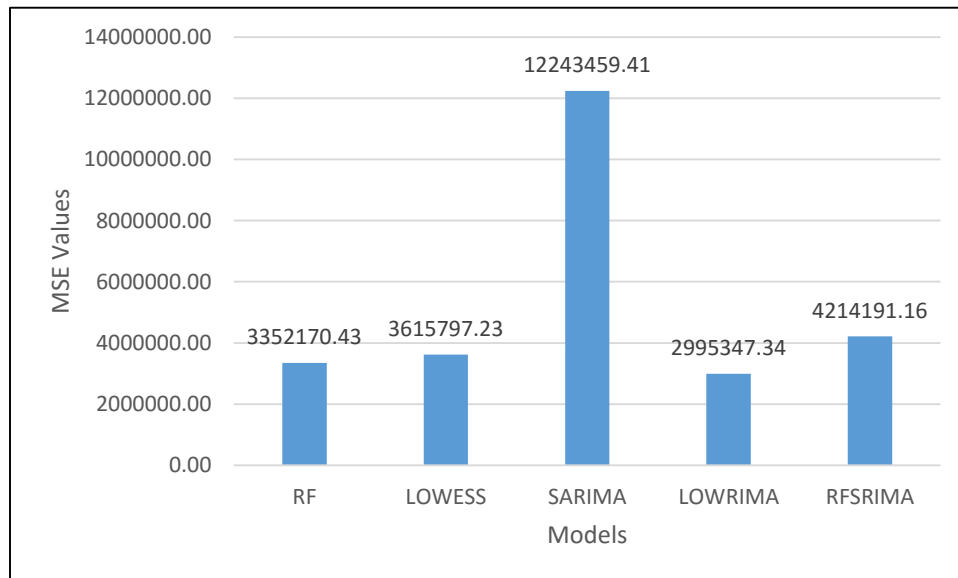


Figure 108: MSE values for Dec 2014 projection (MTLF 2012-2014) at 8pm

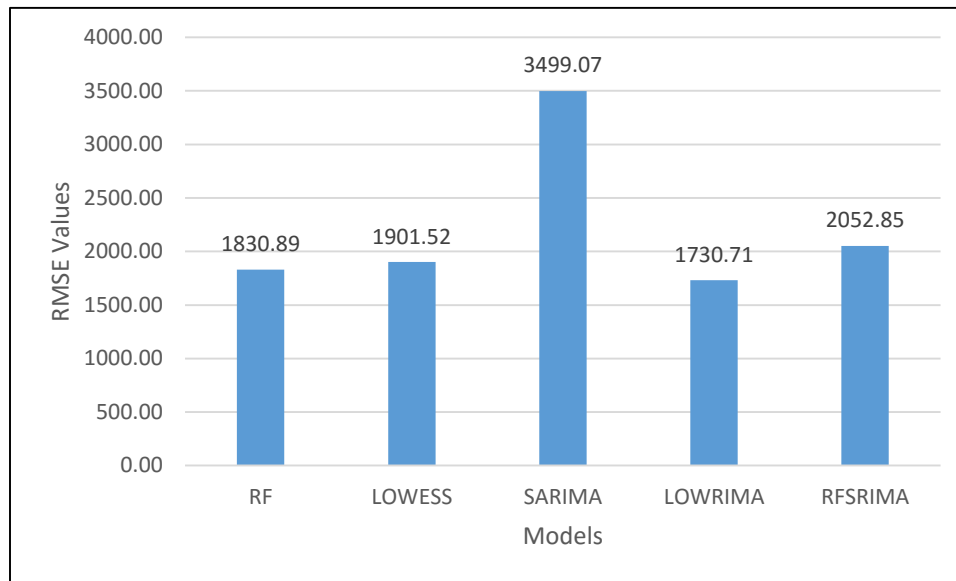


Figure 109: RMSE values for Dec 2014 projection (MTLF 2012-2014) at 8pm

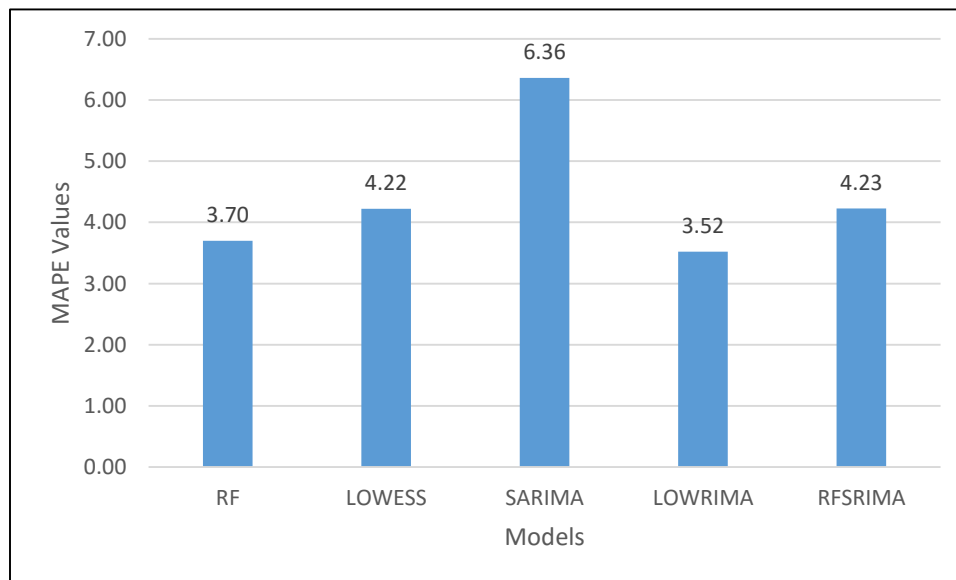


Figure 110: MAPE values for Dec 2014 projection (MTLF 2012-2014) at 8pm

Figure 111 to 126 represent a comparative analysis of the statistical indices (MAE, MSE, RMSE and MAPE) calculated for each of the forecasting methods (RF, LOWESS, SARIMA, LOWRIMA and RFSRIMA) used to project demand data for the months of May, August, November and December. Figures 111, 112, 113 and 114 represent a comparison of MAE values, figures 115, 116, 117 and 118 present a comparison of MSE values, figures 119, 120, 121 and 122 represent a comparison of the RMSE values and figures 123, 124, 125 and 126 represent a comparison of MAPE values for the forecasting methods for four separate months of a year. A study of the figures reveals that in all of the cases, SARIMA method of forecasting performs the worst, while the rest of the methods like RF, LOWESS, LOWRIMA and RFSRIMA perform similarly.

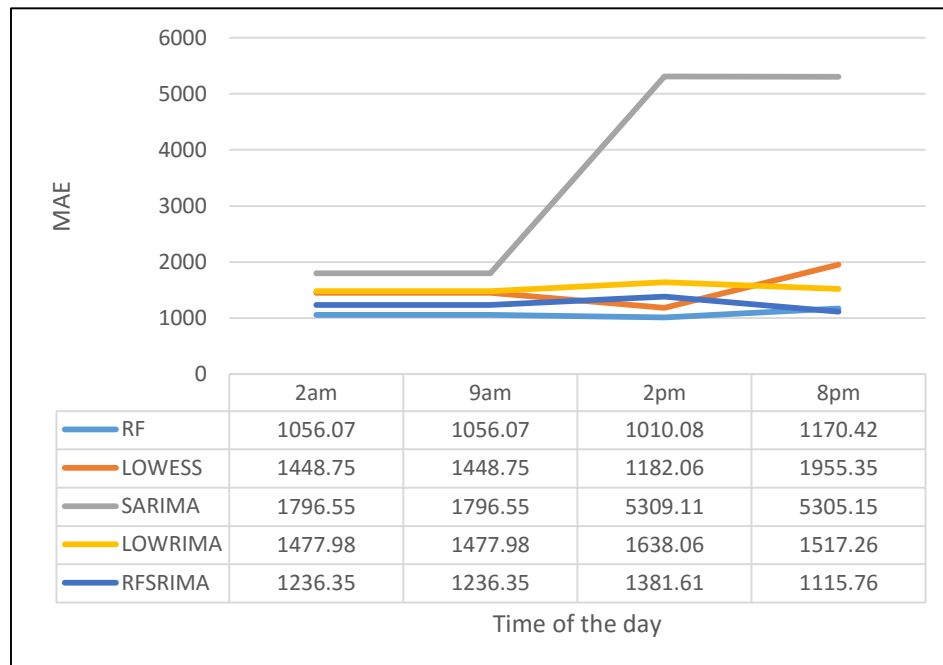


Figure 111: Comparison of MAE values for the projections for May 2015

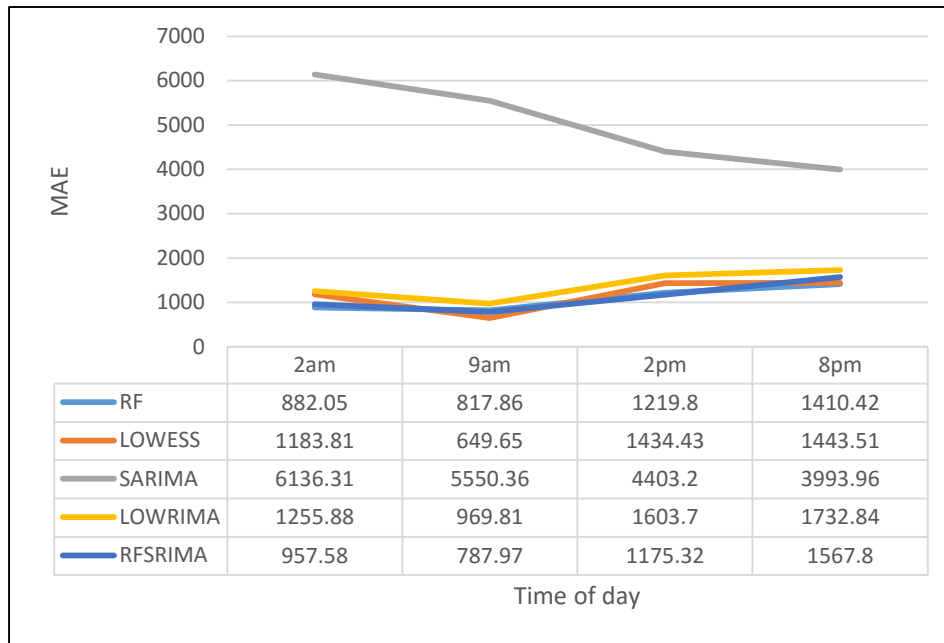


Figure 112: : Comparison of MAE values for the projections for August 2015

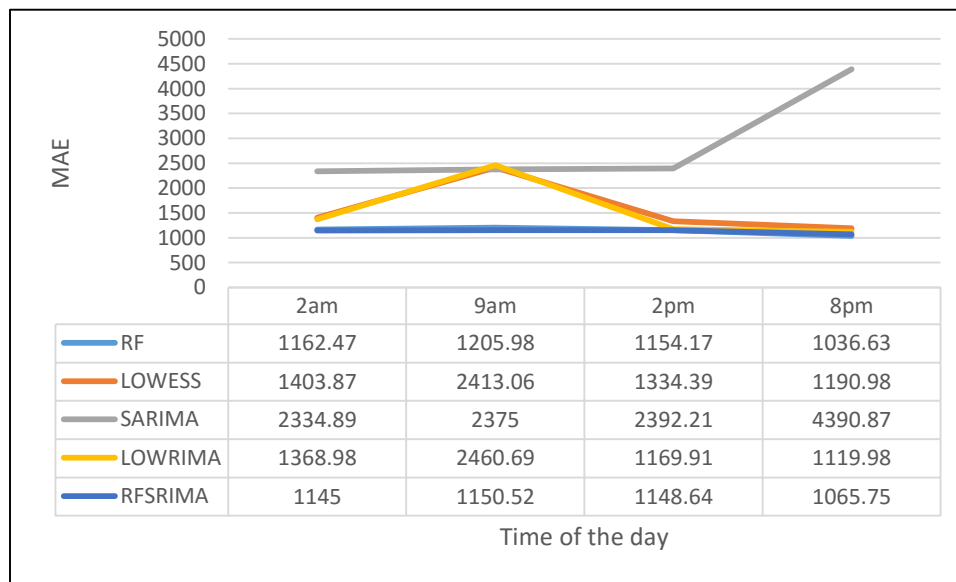


Figure 113: Comparison of MAE values for the projections for November 2014

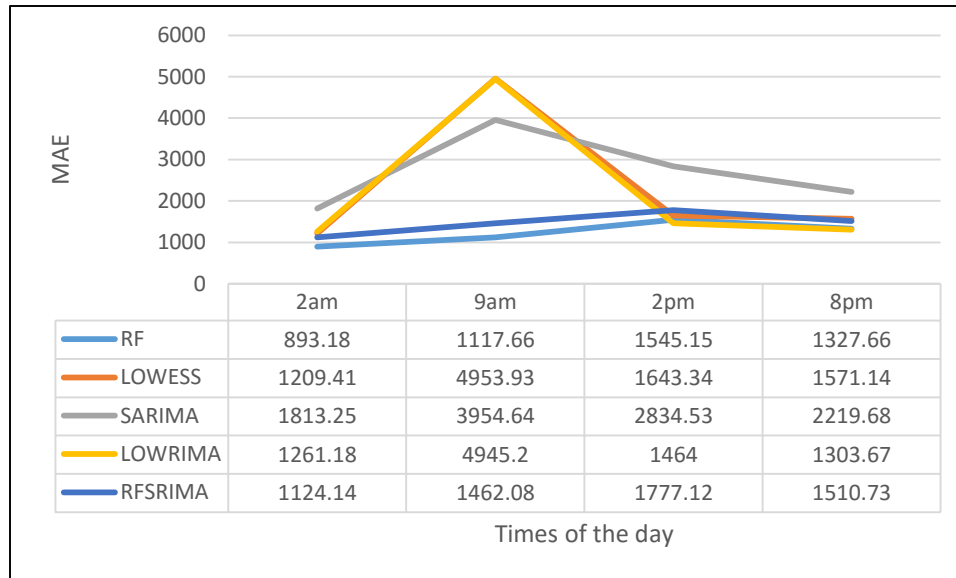


Figure 114: Comparison of MAE values for the projections for December 2014

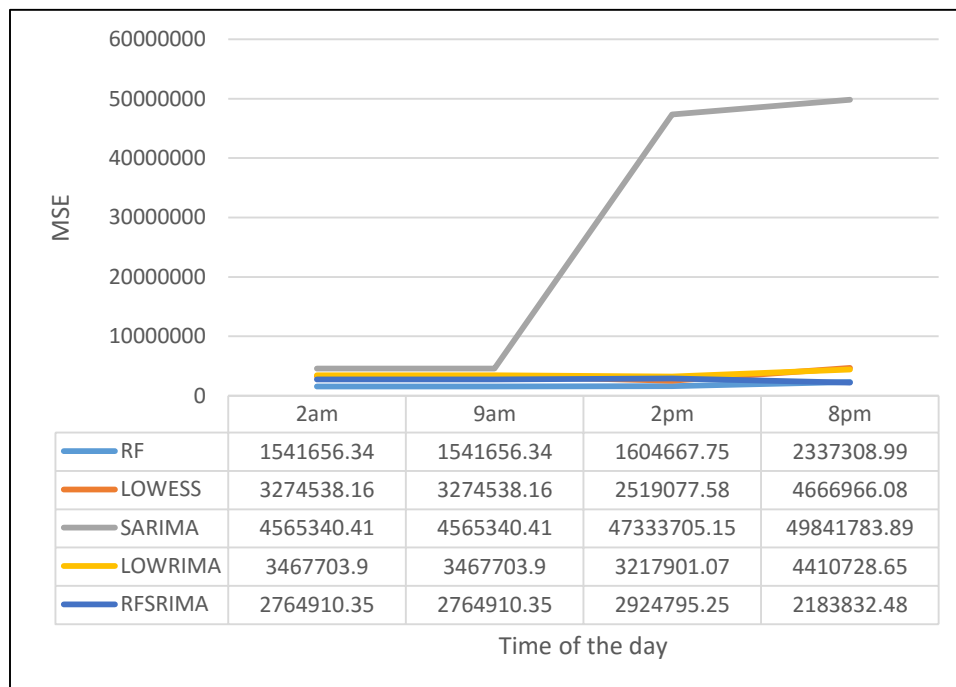


Figure 115: Comparison of MSE values for the projections for May 2015

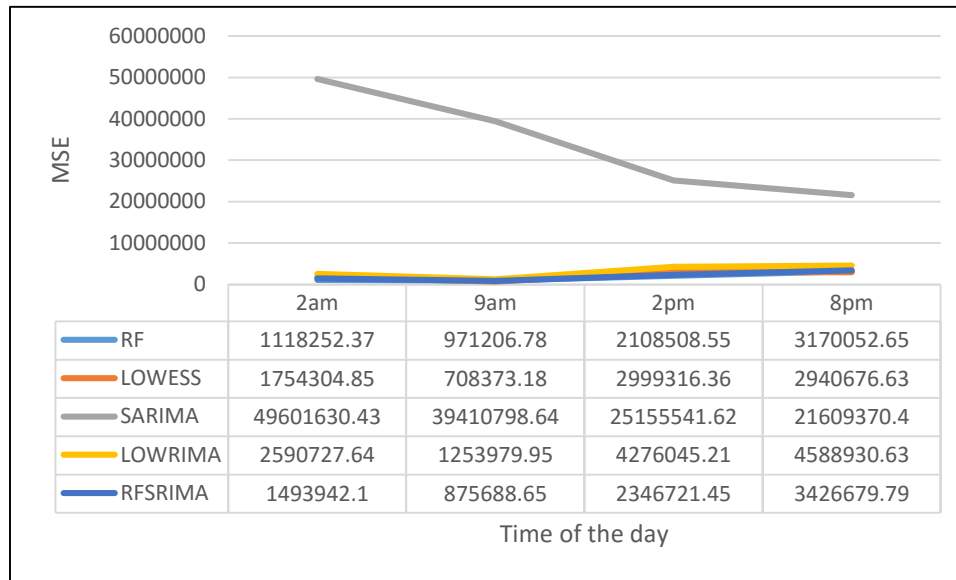


Figure 116: Comparison of MSE values for the projections for August 2015

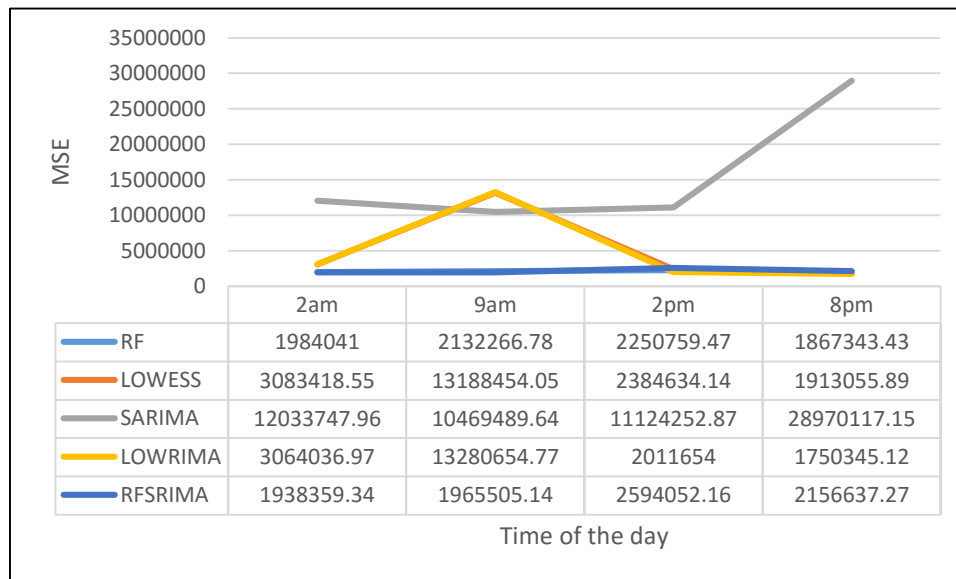


Figure 117: Comparison of MSE values for the projections for November 2014

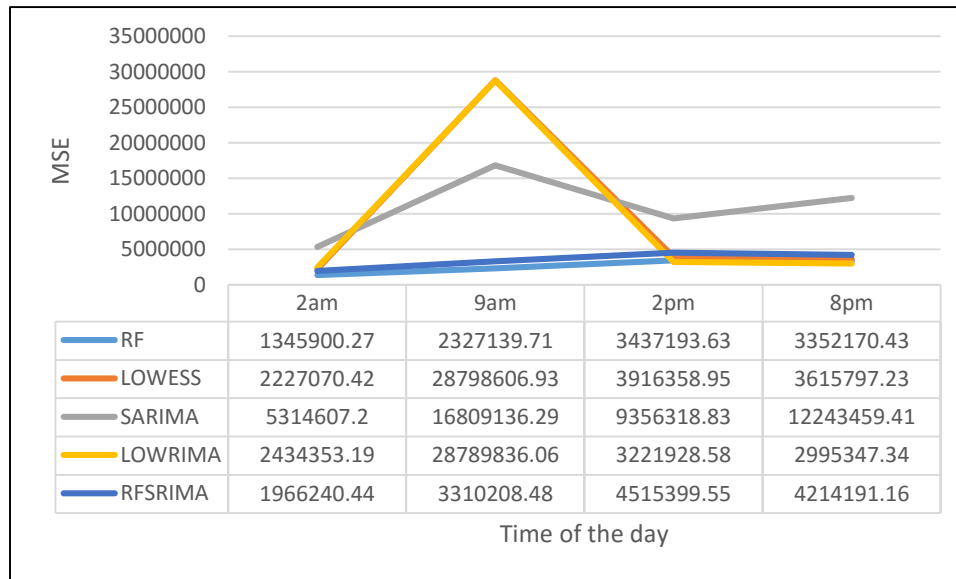


Figure 118: Comparison of MSE values for the projections for December 2014

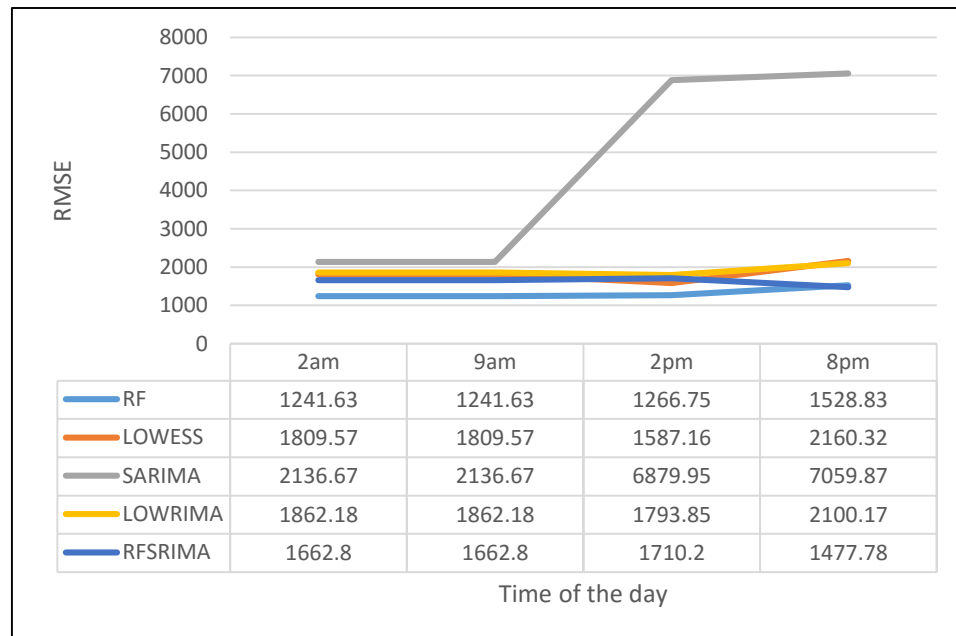


Figure 119: Comparison of RMSE values for the projections for May 2015

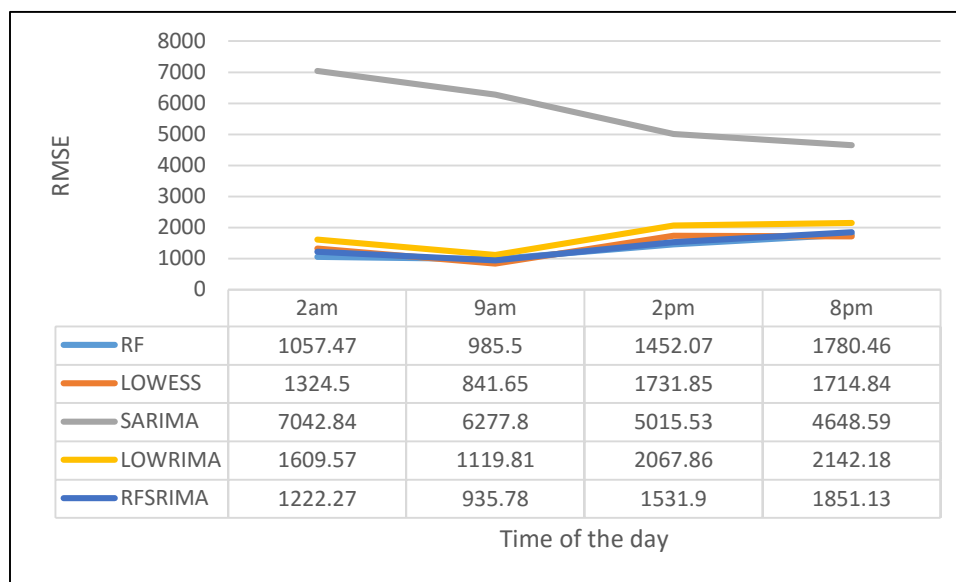


Figure 120: Comparison of RMSE values for the projections for August 2015

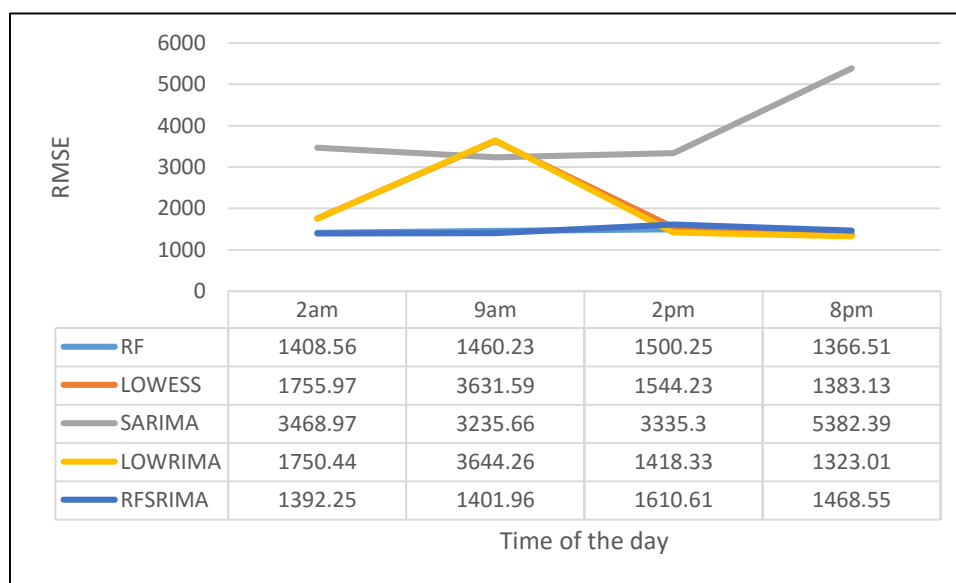


Figure 121: Comparison of RMSE values for the projections for November 2014

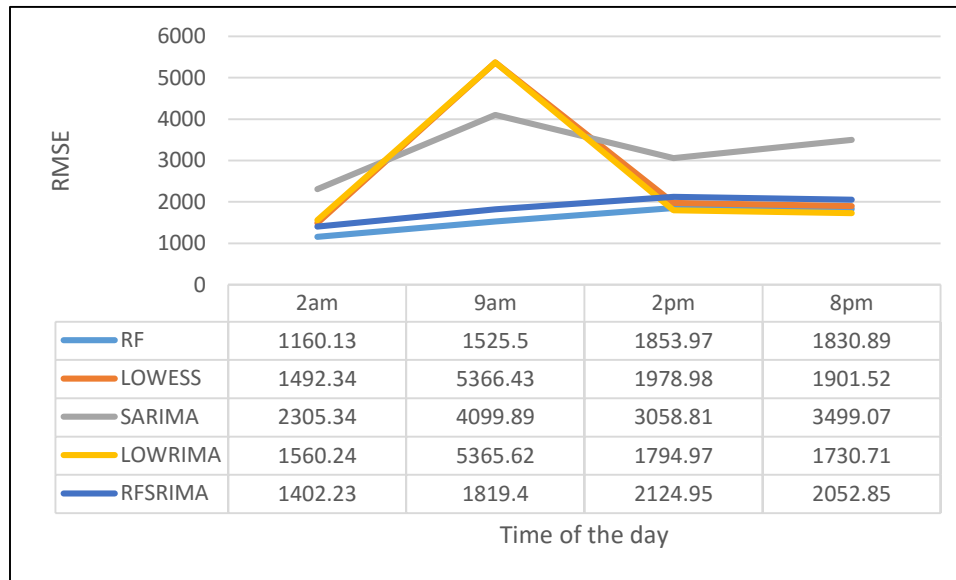


Figure 122: Comparison of RMSE values for the projections for December 2014

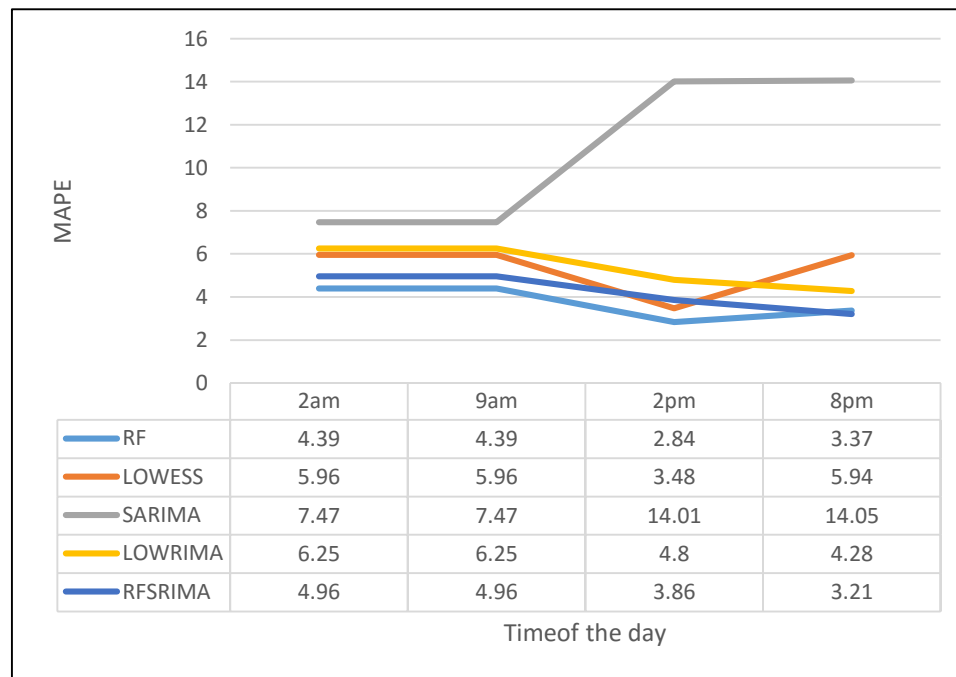


Figure 123: Comparison of MAPE values for the projections for May 2015

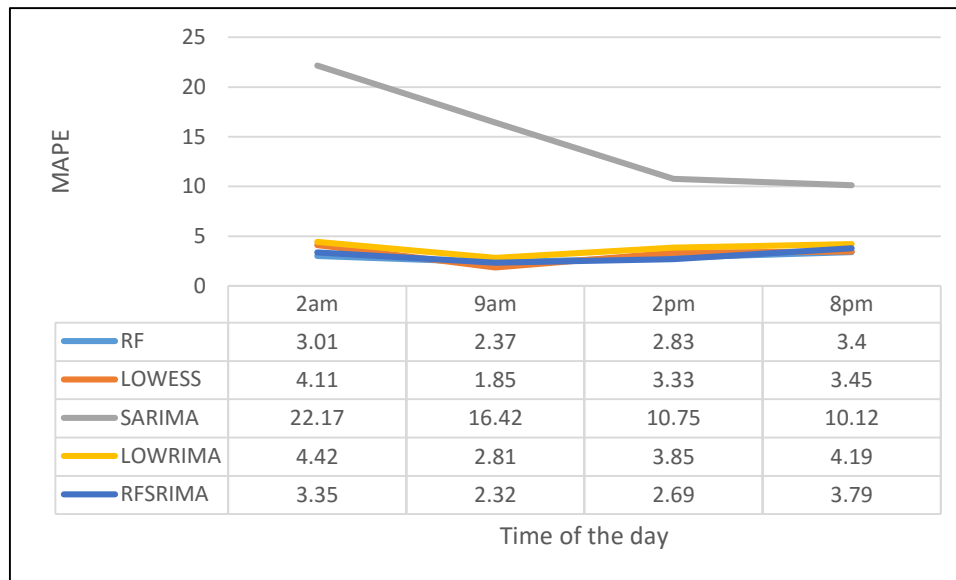


Figure 124: Comparison of MAPE values for the projections for August 2015

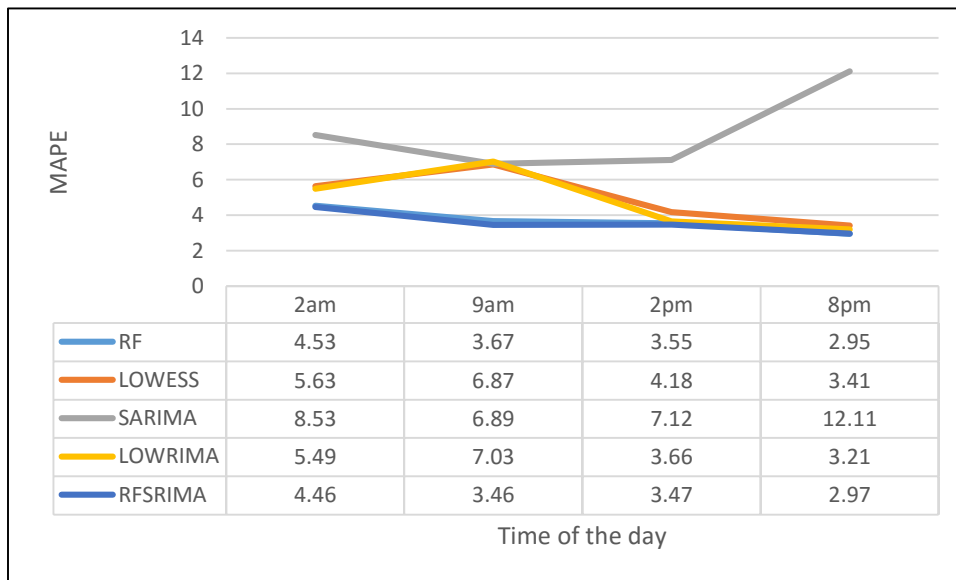


Figure 125: Comparison of MAPE values for the projections for November 2014

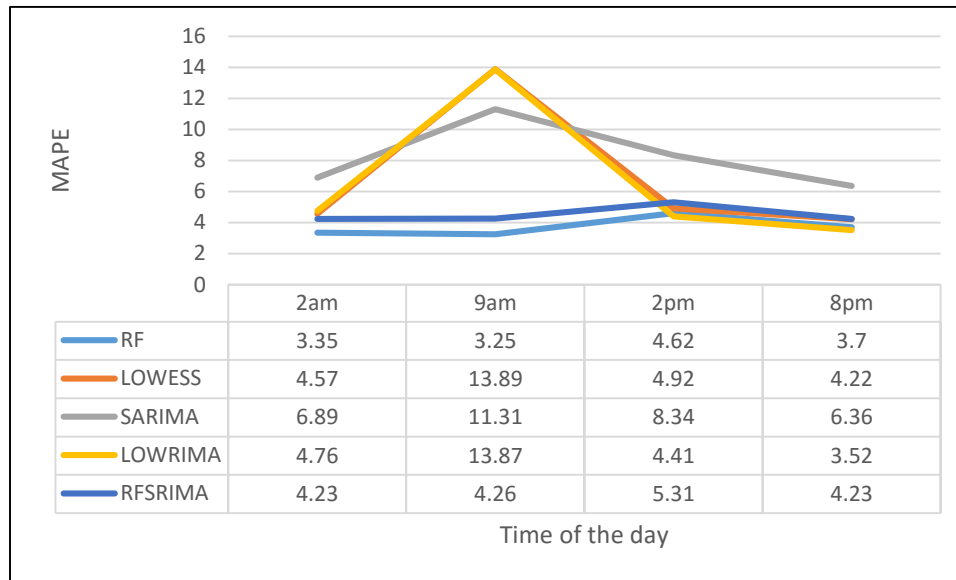


Figure 126: Comparison of MAPE values for the projections for December 2014

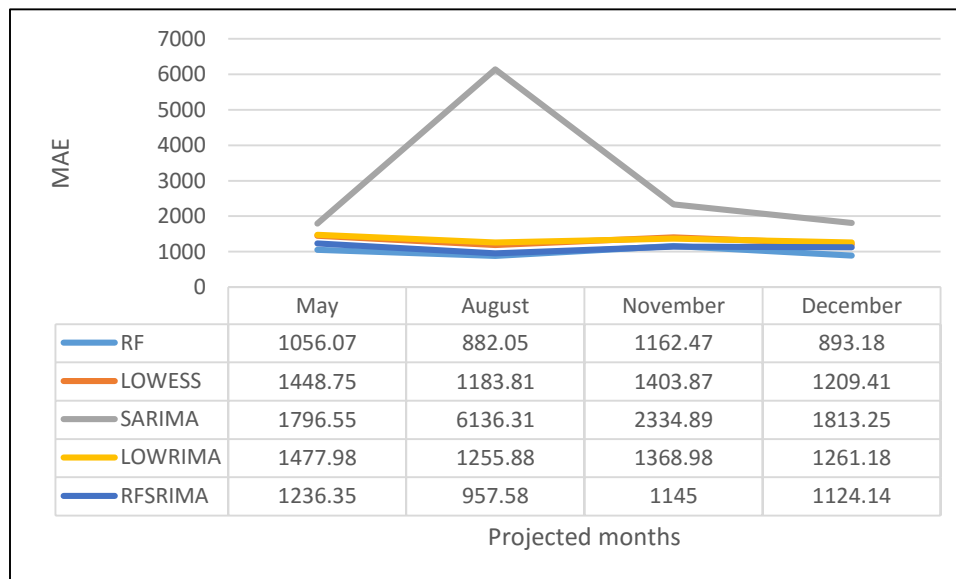


Figure 127: Comparison of MAE values for the projections at 2am

Figures 127 to 142 presents a comparative plot of statistical error indices (MAE, MSE, RMSE, MAPE) for values predicted by different forecasting methods (RF, LOWESS, SARIMA, LOWRIMA and RFSRIMA) at certain hours of a day (2am, 9am, 2pm and 8pm) for the months of May, August, November and December. Figures 127 to 130 represent

the MAE values for every month, figures 131 to 134 represent MSE values, figures 135 to 138 represent the RMSE values and finally figures 139 to 142 present the MAPE values for four separate hours of a day.

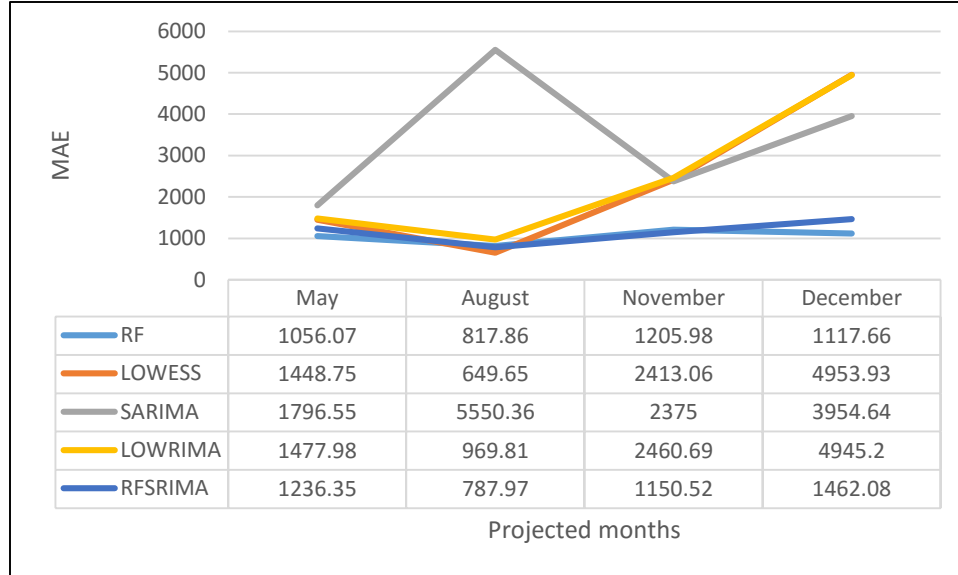


Figure 128: Comparison of MAE values for the projections at 9am

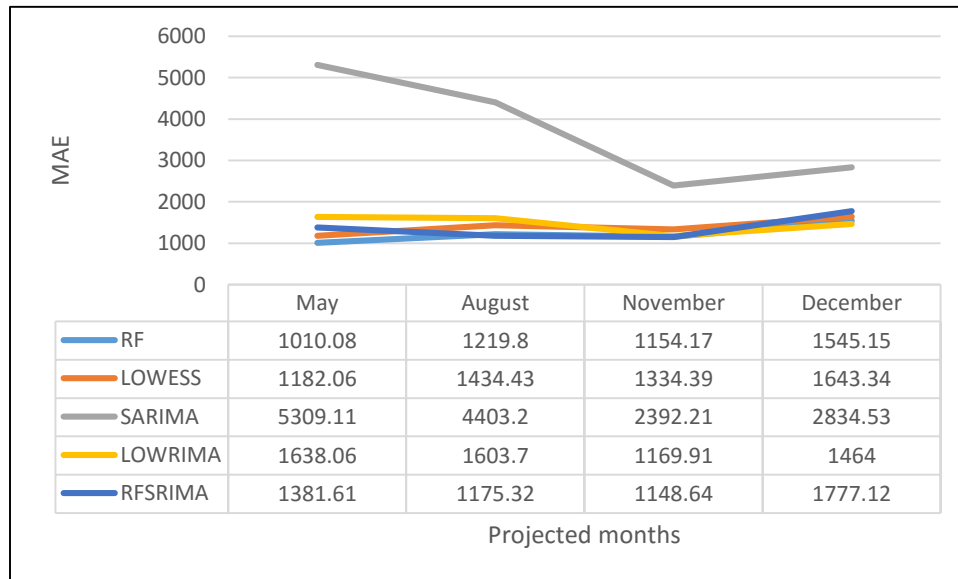


Figure 129: Comparison of MAE values for the projections at 2pm

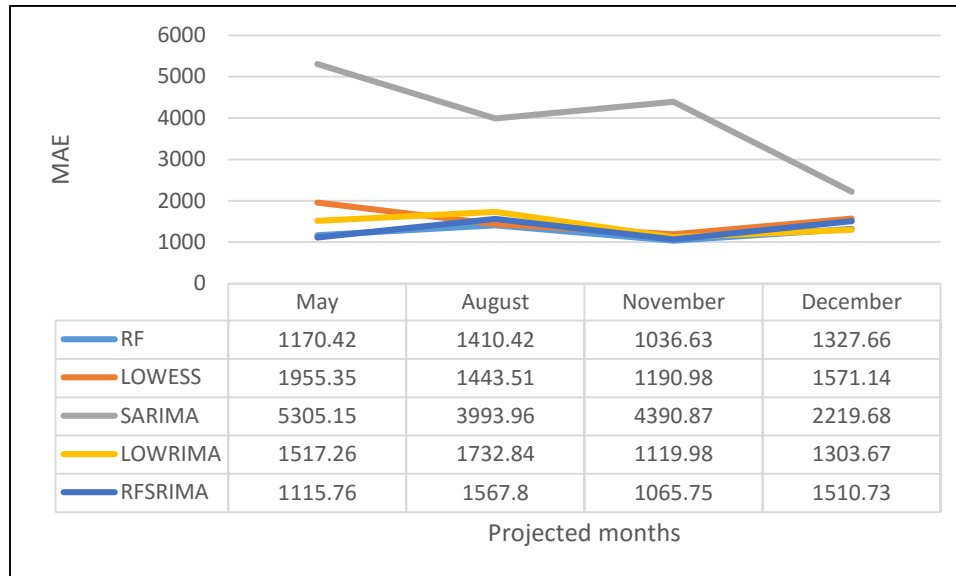


Figure 130: Comparison of MAE values for the projections at 8pm

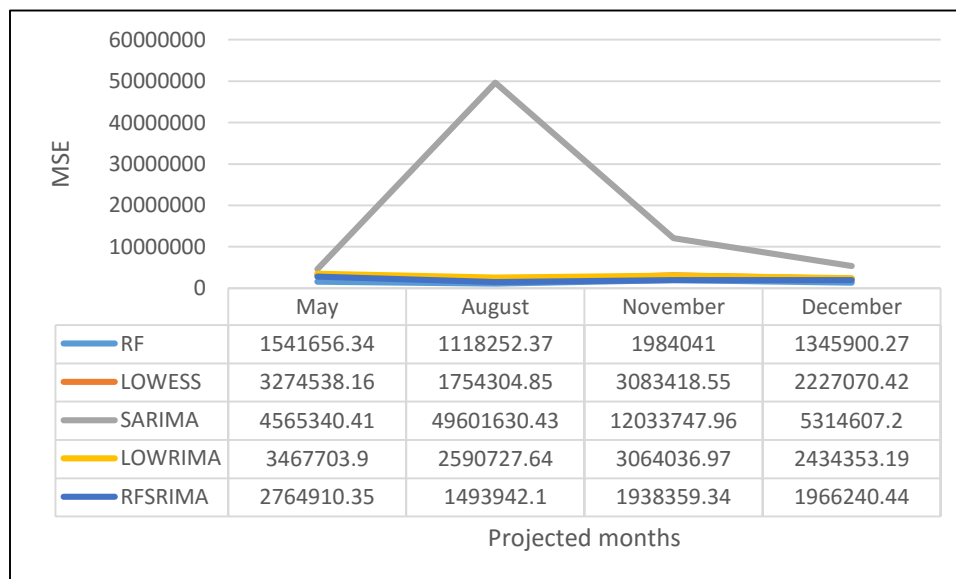


Figure 131: Comparison of MSE values for the projections at 2am

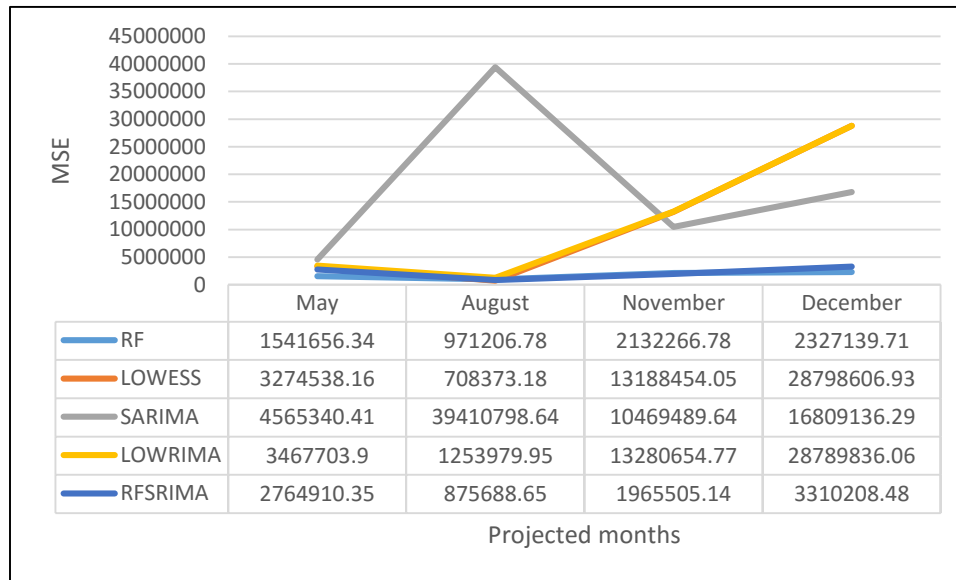


Figure 132: Comparison of MSE values for the projections at 9am

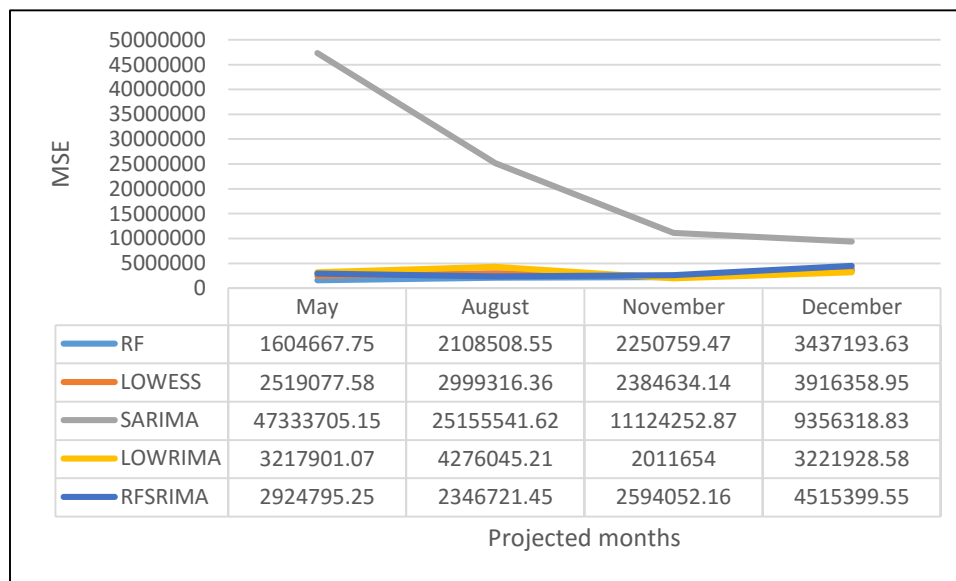


Figure 133: Comparison of MSE values for the projections at 2pm

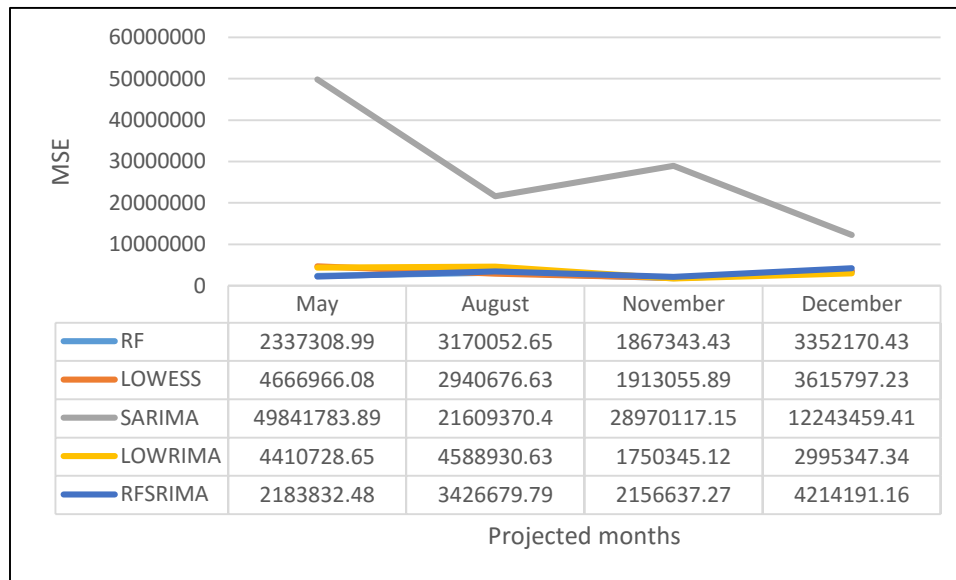


Figure 134: Comparison of MSE values for the projections at 8pm

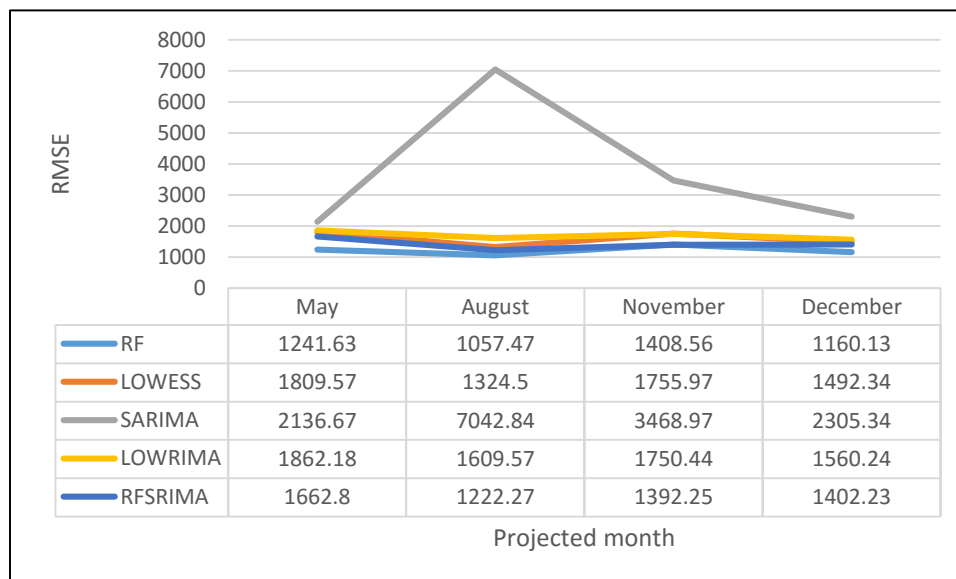


Figure 135: Comparison of RMSE values for the projections at 2am

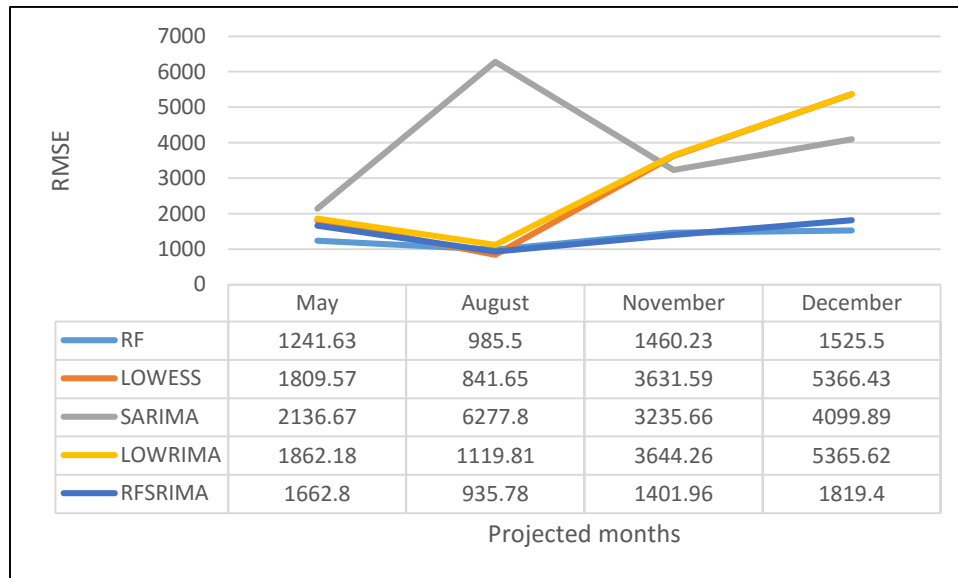


Figure 136: Comparison of RMSE values for the projections at 9am

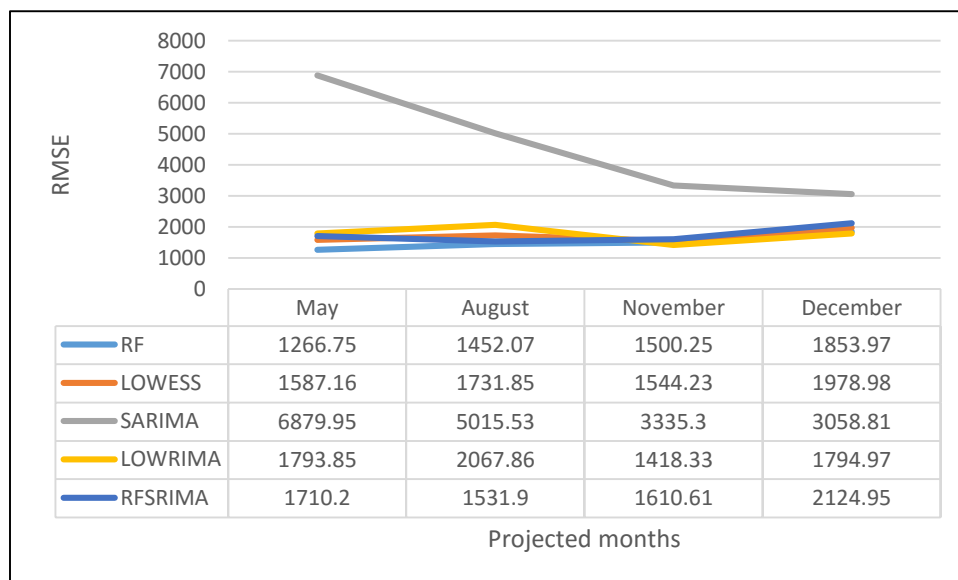


Figure 137: Comparison of RMSE values for the projections at 2pm

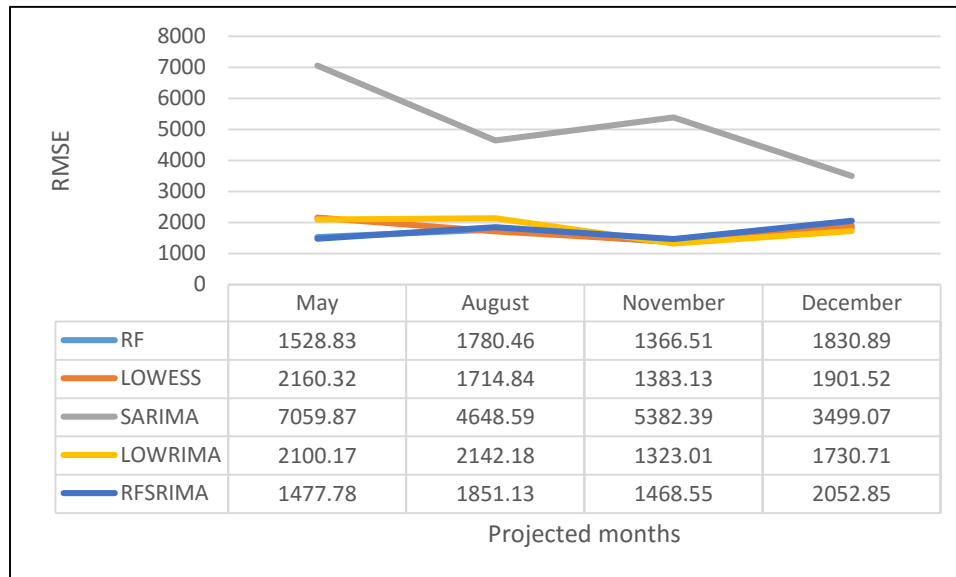


Figure 138: Comparison of RMSE values for the projections at 8pm

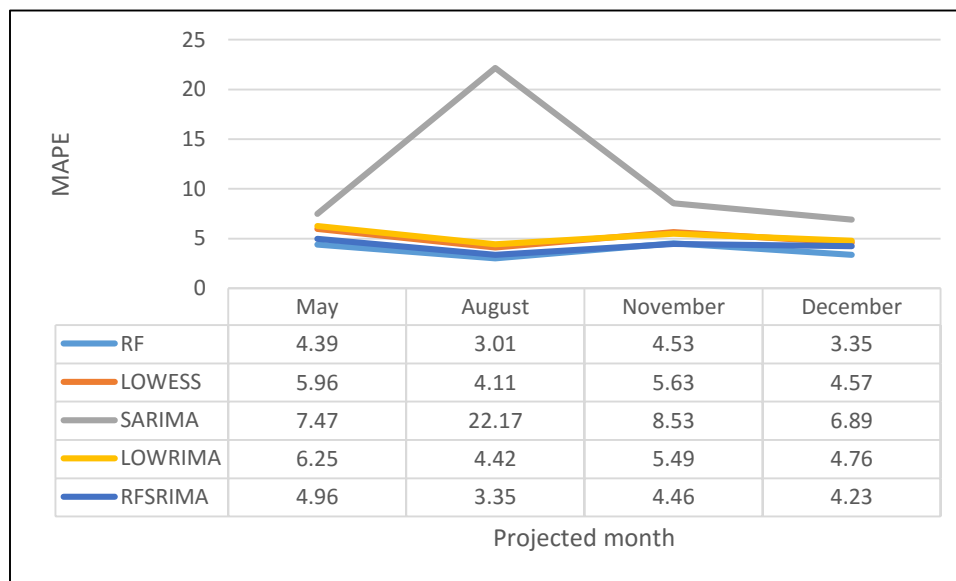


Figure 139: Comparison of MAPE values for the projections at 2am

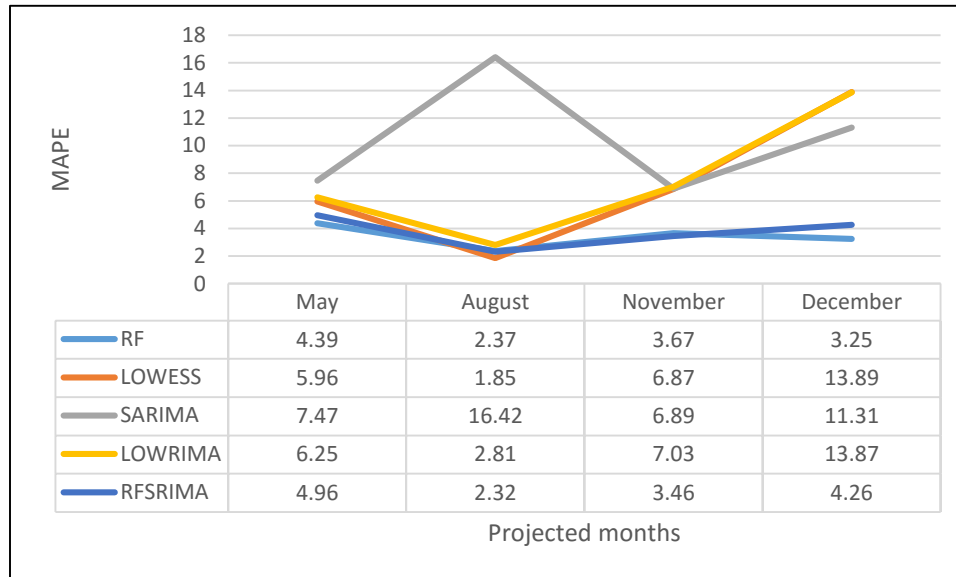


Figure 140: Comparison of MAPE values for the projections at 9am

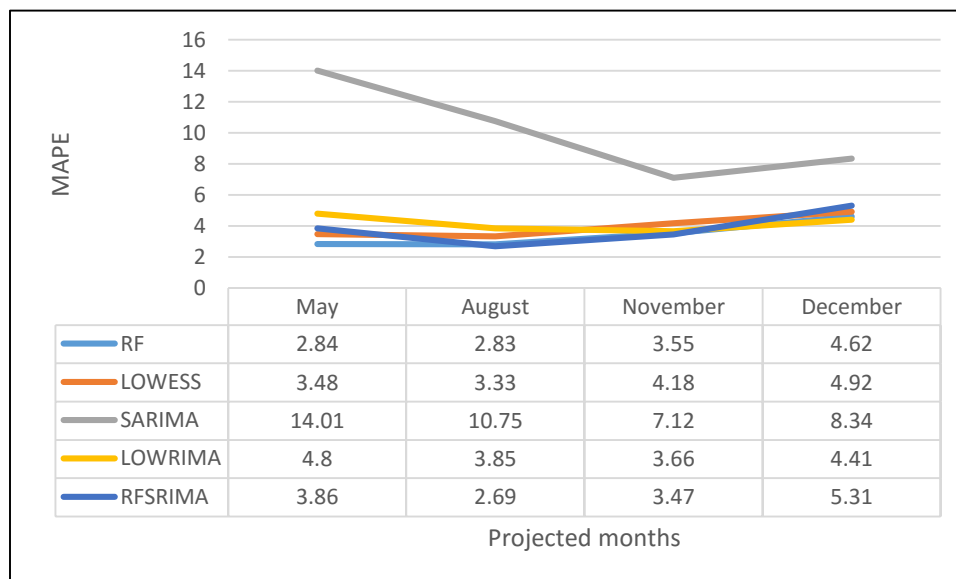


Figure 141: Comparison of MAPE values for the projections at 2pm

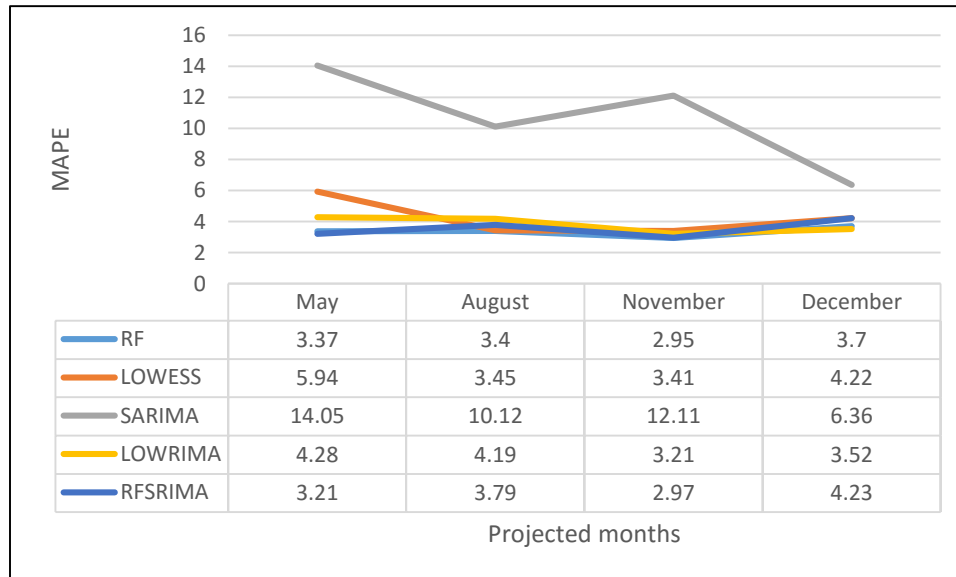


Figure 142: Comparison of MAPE values for the projections at 8pm

In order to improve the accuracy of load data forecasting, it is important that the error in load forecast is calculated. This error can be calculated on a daily basis, where the forecasted data is subtracted from the actual value for a given day. While that gives a clear idea about the deviation that exists between the actual data and the forecasted value, it is also worth noting what the overall error is for the forecast period. This is useful specially in case of longer forecasting periods like MTLF and LTLF as this error gives a wider picture to help estimate the distant future of load demand and schedule maintenance or increase in generation. Some of these methods are defined as follows.

a) Mean Absolute Error (MAE)

This index is also measured in the same units as the data, and is usually similar in magnitude to, but slightly smaller than, the root mean squared error. It is less sensitive to the occasional very large error because it does not square the errors in the calculation. This index makes it easier to interpret the data than the RMSE

index. MAE and MAPE are not a part of standard regression output. They are more commonly found in the output of time series forecasting procedures [99].

Mathematically this can be represented as

$$MAE = \frac{\sum_{i=1}^n |e_i|}{n} \quad (59)$$

Where, $e_i = y_i - \hat{y}_i$

b) Mean Squared Error (MSE)

The mean error (ME) and mean percentage error (MPE) that are reported in some statistical procedures are signed measures of error which indicate whether the forecasts are biased--i.e., whether they tend to be disproportionately positive or negative. Bias is normally considered a bad thing, but it is not the bottom line. Bias is one component of the mean squared error. In fact mean squared error equals the variance of the errors plus the square of the mean error. That is: $MSE = VAR(E) + (ME)^2$. Hence, if mean squared error is minimized, the bias and the variance of the errors get minimized as well [99]. Mathematically it can be presented as

$$MSE = \frac{\sum_{i=1}^n (e_i^2)}{n} \quad (60)$$

Where, $e_i^2 = (y_i - \hat{y}_i)^2$

c) Root Mean Squared Error (RMSE)

This index is more sensitive than other measures to the occasional large error. The squaring process gives disproportionate weight to very large errors. If an occasional large error is not a problem in inferring results, then the MAE or MAPE may be a more relevant criterion [99]. Mathematically it is represented as

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (e_i^2)}{n}} \quad (61)$$

Where, $e_i^2 = (y_i - \hat{y}_i)^2$

d) Mean Absolute Percentage Error

This index is often useful for purposes of reporting, because it is expressed in generic percentage terms which makes it easy to comprehend any big errors in the prediction. The MAPE can only be computed with respect to data that are guaranteed to be strictly positive, so if this statistic is missing the output, it is possible that it has been suppressed due to negative data values [99]. Mathematical representation of the index is given as

$$MAPE = \frac{\sum_{i=1}^n |p_i|}{n} \quad (62)$$

Where, the percentage error p_i is given by

$$p_i = 100 * \frac{e_i}{y_i} \quad (63)$$

And $e_i = y_i - \hat{y}_i$.

In each of the equations 59-63, y_i is the actual load demand data for a given working day at a certain hour of the day and \hat{y}_i is the corresponding predicted value.

Application of RFSRIMA is a two step process. In stage one, a preliminary prediction for demand power is made using RF method. The demand power data downloaded from PJM Corp. database was divided into two groups, training set and testing set. The hybrid model was used to forecast load data for the month of May and August in the year 2015 and November and December in the year 2014. The training dataset consisted of historical load data starting from the date 1/1/2012 and continuing till end of the month previous to the

forecast months i.e., while predicting load data for the month of May, 2015, the training set included data from January, 2012 till February, 2015. The error between the actual and the predicted data was then calculated for that month. This difference set was then used as a time series data for SARIMA modelling in the second stage. In order to select the best fitting SARIMA model, i.e. the model that is likely to predict with the most accuracy, for the time series data, a trial and error method was adopted. In order to select the appropriate p, d, q, P, D and Q values to generate the corresponding SARIMA model, autocorrelation (ACF) and a partial autocorrelation (pACF) plot had to be calculated [100]. A seasonality of 24 was chosen since the load data showed similar patterns after every 24 hours. From the pACF plot, the lags that are 24 or its multiples were studied for their significance. Based on a positive or negative spike as well as gradual or sharp cut-off of the plots, appropriate values were chosen [101]. Figure 143 shows a sample of the ACF and pACF plots and figure 112 shows a SR plot. In order to select an appropriate model that would give the maximum accuracy, the SR value was kept within ± 3 [102]. The standardized residual (SR) is the ratio of residual (difference between actual and predicted values) to the standard deviation of the residuals values. It can be mathematically represented as:

$$SR = \frac{\text{Residual } (R)}{\text{Standard Deviation } (SD) \text{ of } R} \quad (64)$$

Where, R can be represented as:

$$R = y - \hat{y}$$

Where, y = actual values and \hat{y} = the predicted values.

Standard Deviation (SD) of a dataset can be represented mathematically as follows.

$$(SD)\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2} \quad (65)$$

Where, N = number of data points in the dataset, x_i is the i^{th} data point and μ is the mean of the dataset.

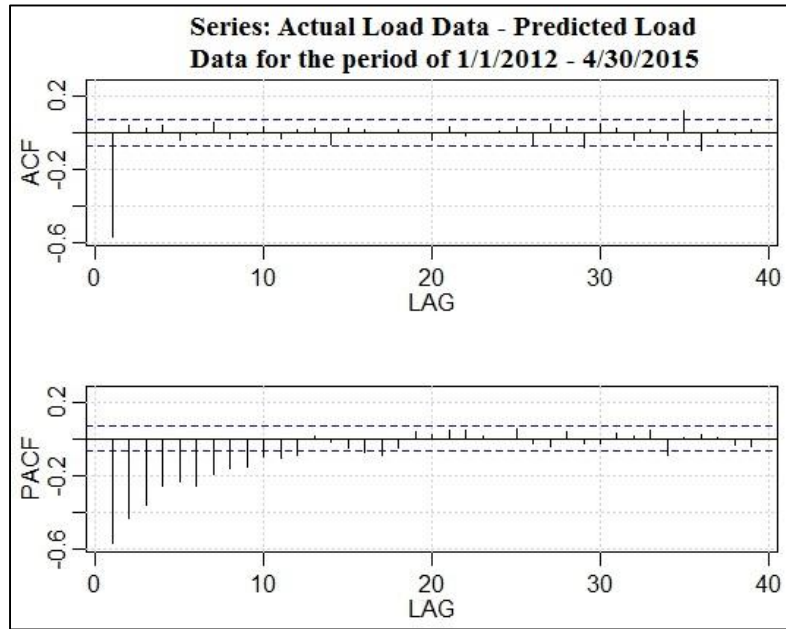


Figure 143: ACF and pACF plot for residuals from 1/1/2012 to 4/30/2015 at 2am

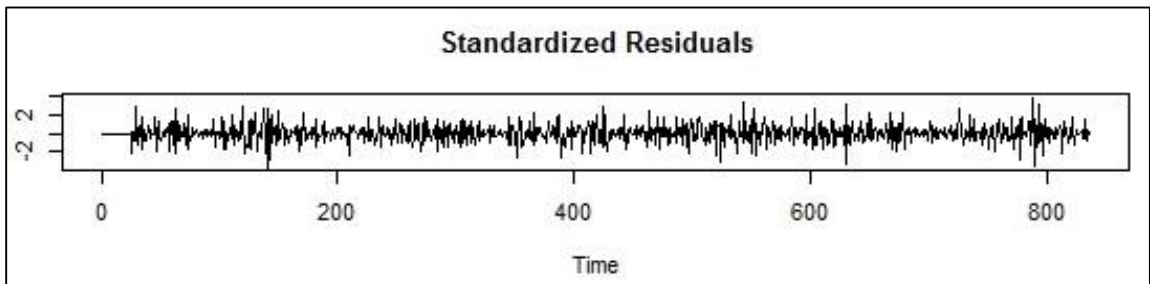


Figure 144: Standardized residuals plot for the period 1/1/2012 - 4/30/2015 at 2am

The figure 144 shows the SR plot for the residuals dataset encompassing 1/1/2012 - 4/30/2015 at 2am. The SARIMA models used for each of the four hours (2am, 9am, 2pm

and 8pm) of months May, August, November and December for the RFSRIMA hybrid method are represented in table 65.

Table 65: SARIMA models for forecasting residuals

May 2015 SARIMA(p,d,q,P,D,Q)s			
2 am	9 am	2 pm	8 pm
(1,1,0,1,1,1), s = 24	(1,1,0,1,1,1), s = 24	(1,1,0,1,1,0), s = 24	(1,1,0,1,1,0), s = 24
August 2015 SARIMA(p,d,q,P,D,Q)s			
(0,1,1,0,1,0), s = 22	(1,1,0,0,1,1), s = 22	(0,0,1,0,1,0), s = 24	(0,0,1,0,1,0), s = 24
November 2014 SARIMA(p,d,q,P,D,Q)s			
(0,0,1,1,0,1), s = 22	(0,0,1,1,0,1), s = 22	(1,0,0,0,1,1), s = 24	(0,0,1,0,1,1), s = 24
December 2014 SARIMA(p,d,q,P,D,Q)s			
(1,0,1,1,1,0), s = 24	(1,0,1,1,1,0), s = 24	(1,1,1,0,1,0), s = 24	(1,0,1,0,1,1), s = 22

5.2.2 Short term load forecasting (STLF)

In the previous section, data models were developed to best fit the load data available and reduce the error. In this section, previously fitted models have been used to predict the day ahead load data on an hourly basis. Hybrid model LOWRIMA was used for STLF and its accuracy was compared with that of the novel hybrid model RFSRIMA. The various error metrics are evaluated for the model using MAE, MSE, RMSE and MAPE. The experiment was conducted in two steps. First, a prediction model was built by using historical load data from PJM Corporation for the period 1/1/2012 to 6/24/2015. A specific SARIMA model was developed for every hour of the day. Then the model was used to forecast the load data for the following day, i.e. 6/25/2015. Figure 145 presents the hourly forecasted data for the day 6/25/2015 against the actual load data. Figure 146 shows the prediction error for each hour. Table 66 presents the actual load data and the predicted data in an hourly format.

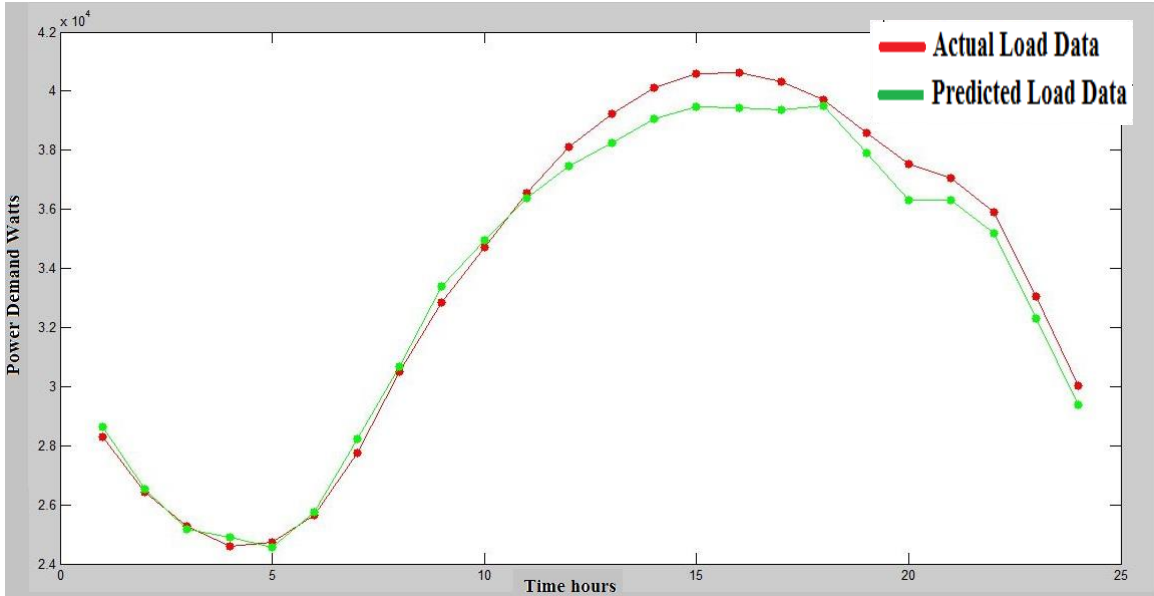


Figure 145: LOWRIMA predicted load vs actual load for 6/25/2015

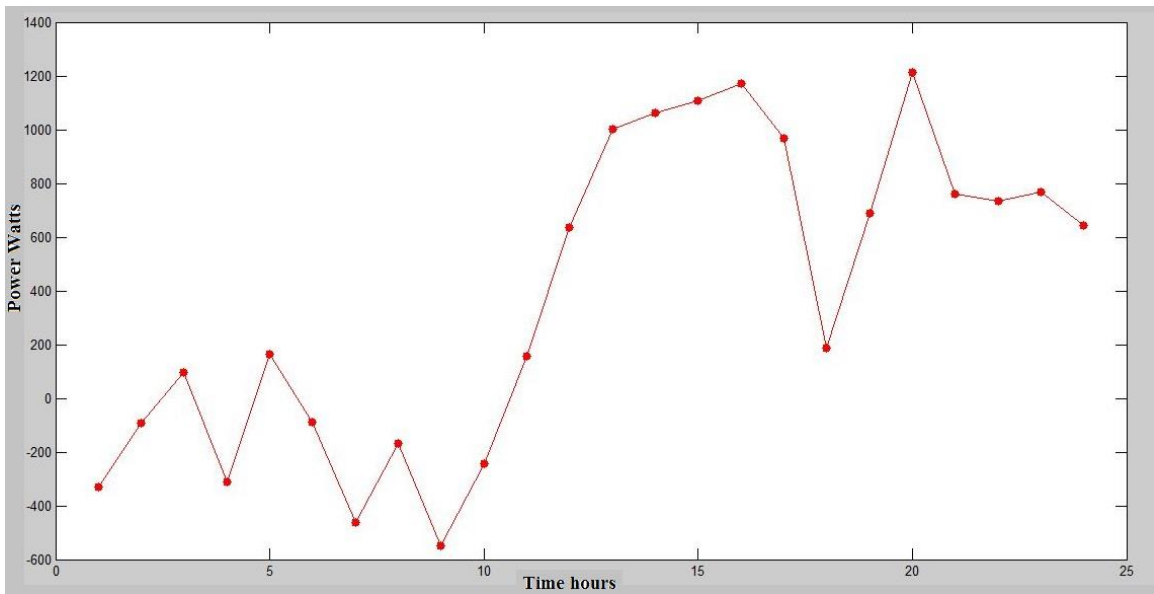


Figure 146: Error plot for hourly predicted load using LOWRIMA data on 6/25/2015

Table 66: Hourly data (Actual vs Predicted) for LOWRIMA

Actual and Predicted Hourly Load Data						
Time	1 am	2 am	3 am	4 am	5 am	6 am
Actual	28293.21	26443.05	25280.4	24588.2	24723.35	25644.63
Predicted	28623.32	26533.65	25182.3	24899.96	24557.4	25734.97
Time	7 am	8 am	9 am	10 am	11 am	12 am
Actual	27753.11	30513.82	32841.11	34718.86	36537.43	38107.71
Predicted	28217.05	30682.78	33391.7	34961.54	36382.07	37473.47
Time	1 pm	2 pm	3 pm	4 pm	5 pm	6 pm
Actual	39248.88	40128.29	40583.28	40610.92	40320.25	39702.01
Predicted	38245.98	39064.68	39475.83	39438.17	39353.94	39514.99
Time	7 pm	8 pm	9 pm	10 pm	11 pm	12 pm
Actual	38586.06	37536.37	37066.83	35914.27	33055.43	30035.9
Predicted	37897.54	36322.41	36307.89	35180.96	32288.68	29391.27

Similarly, the hourly load data for 6/25/2015 was forecasted by using the novel RFSRIMA model. Data used to train the model was the metered load data obtained from PJM Corporation website for the period 1/1/2012 – 6/24/2015 [92]. Figure 147 presents the hourly forecasted data for the day 6/25/2015 against the actual load data. Figure 148 shows the error in prediction for each hour. Table 67 presents the actual load data and the predicted data in an hourly format and Table 68 gives a comparison of accuracy of the two hybrid methods against statistical error indices.

Table 67: Hourly data (Actual vs Predicted) for RFSRIMA

Actual and Predicted Hourly Load Data						
Time	1 am	2 am	3 am	4 am	5 am	6 am
Actual	28293.214	26443.05	25280.4	24588.2	24723.35	25644.63
Predicted	27905.992	26442.47	25136.74	24581.38	24654.46	25765.24
Time	7 am	8 am	9 am	10 am	11 am	12 am
Actual	27753.11	30513.82	32841.11	34718.86	36537.43	38107.71
Predicted	28290	30832.23	32397.11	33893.72	35893.07	37008.25
Time	1 pm	2 pm	3 pm	4 pm	5 pm	6 pm
Actual	39248.88	40128.29	40583.28	40610.92	40320.25	39702.01
Predicted	38199.94	38798.63	40171.67	40981.67	40305.47	40186.88
Time	7 pm	8 pm	9 pm	10 pm	11 pm	12 pm
Actual	38586.06	37536.37	37066.83	35914.27	33055.431	30035.9
Predicted	38873.46	37956.46	37288.09	34888.01	33874.236	30090.3

Table 68: Statistical indices measuring Accuracy of LOWRIMA and RFSRIMA

Error Indices		
Error Index	LOWRIMA	RFSRIMA
MAE	567.03	461.87
MSE	461016.91	355276.9
RMSE	678.98	596.05
MAPE	1.59	1.3008

From table 68, it is observed that RFSRIMA performs better than LOWRIMA method for data based on an entire day. In comparison to MTLF, both the hybrid methods perform much better and are thus more reliable for STLF. Since, both weather parameters and electricity usage keeps changing, the model needs to continuously updated to maintain reliability.

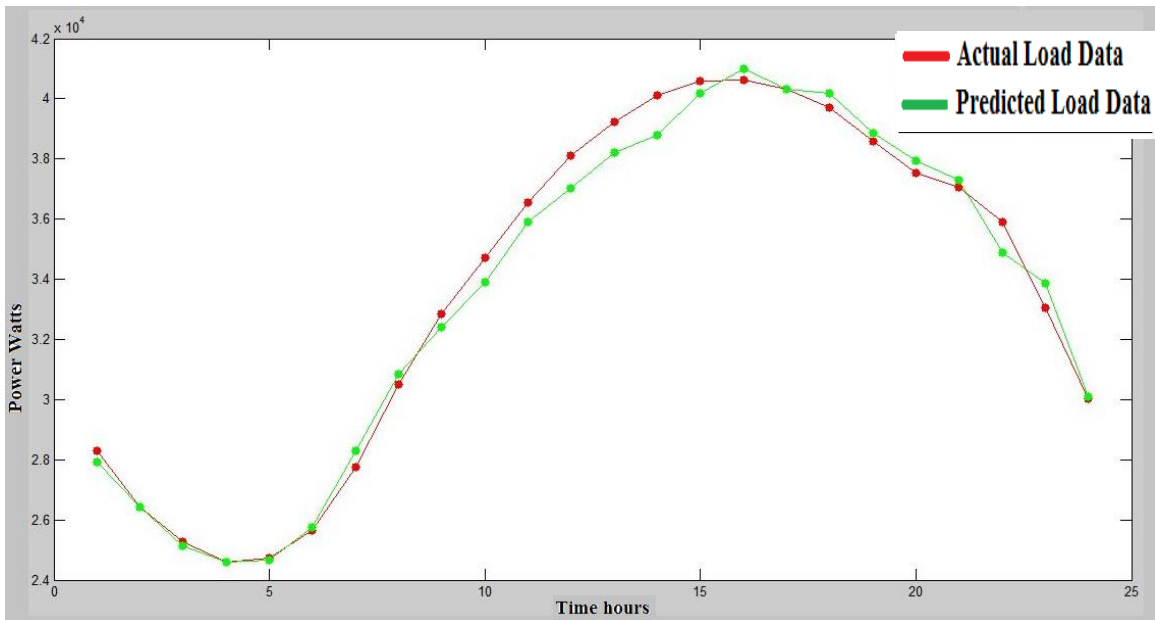


Figure 147: RFSRIMA predicted load vs actual load for 6/25/2015

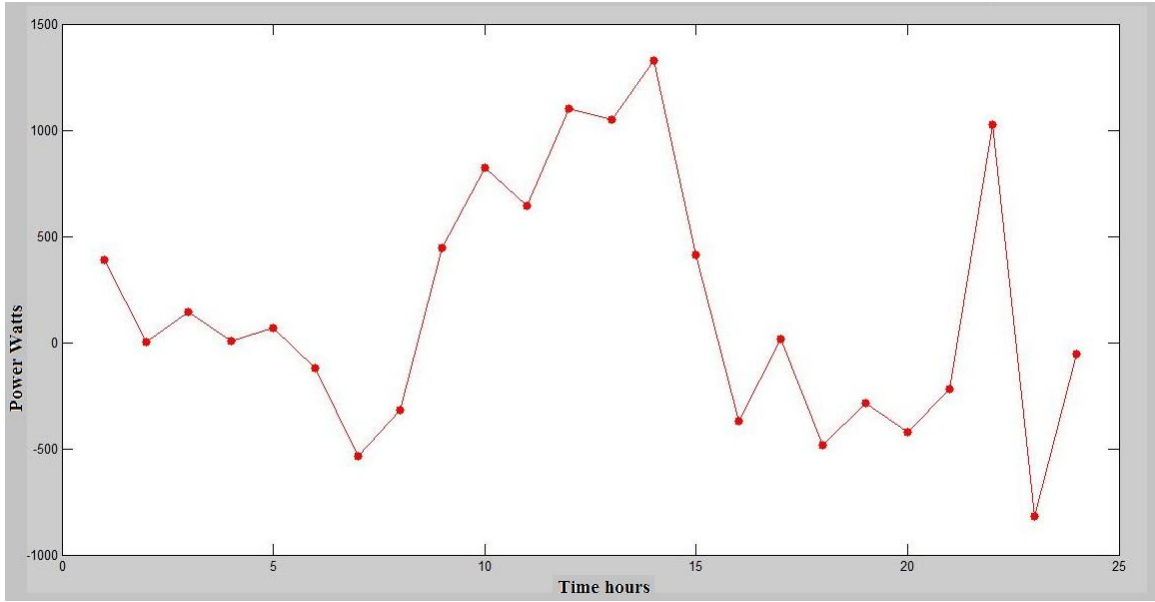


Figure 148: Error plot for hourly predicted load using RFSRIMA data on 6/25/2015

The 24 hours in a day was divided into four quarters of 6 hours each. Statistical indices for each of these 6 hour quartiles were calculated and plotted in an attempt to do a further analysis into the errors in model selection. These errors cause deviation of predicted values from the actual load data acquired from PJM Corporation. Figures 149 to 152 present the error indices (MAE, MSE, RMSE and MAPE) for LOWRIMA hybrid forecasting method calculated for every six hour periods of a day, namely, early morning, mid-day, afternoon and evening.

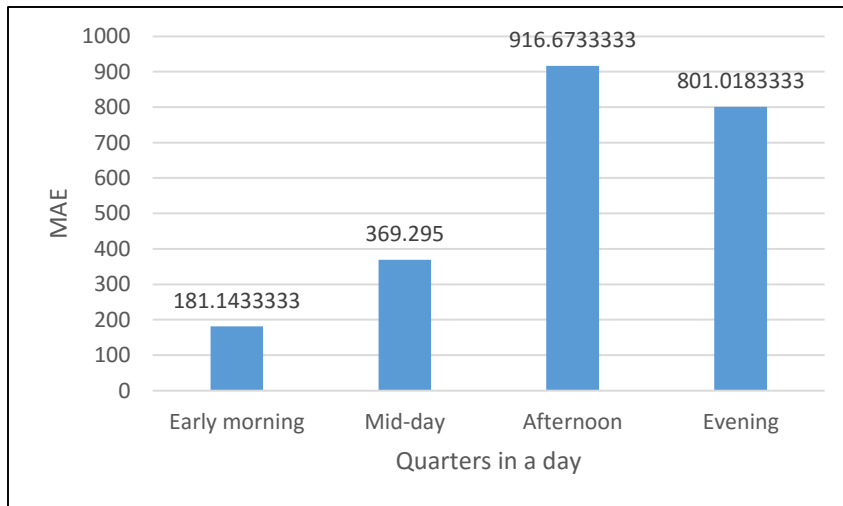


Figure 149: MAE for values projected quarterly by LOWRIMA on 6/25/2015

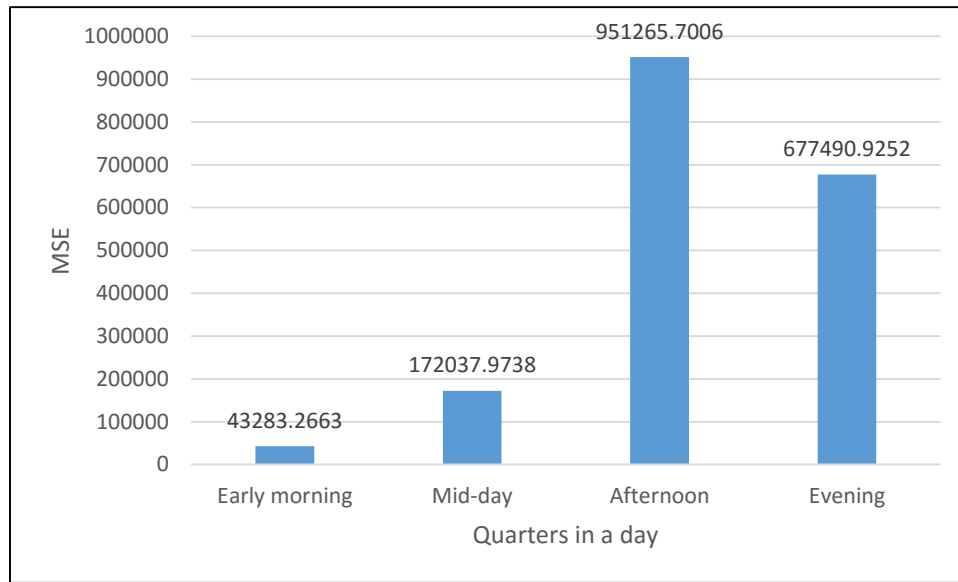


Figure 150: MSE for values projected quarterly by LOWRIMA on 6/25/2015

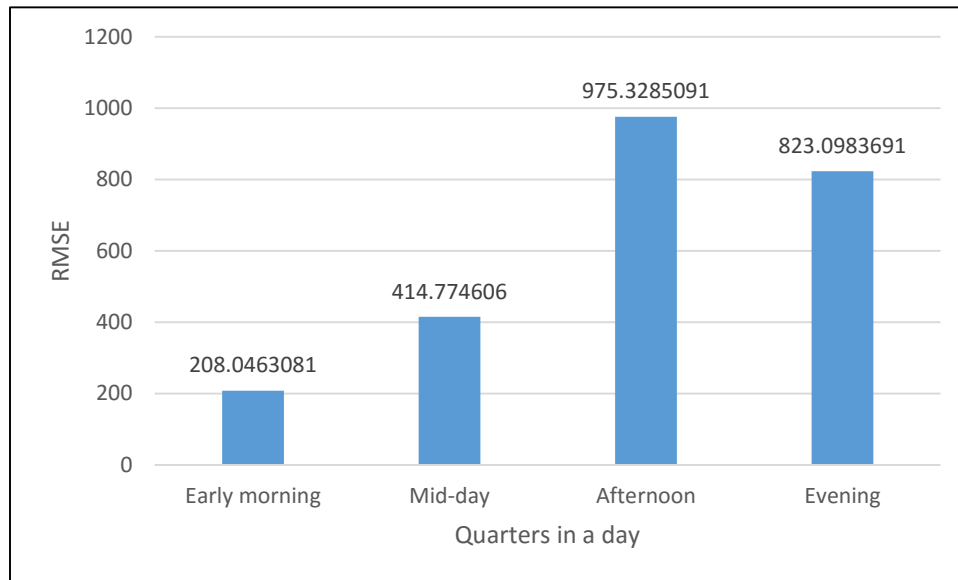


Figure 151: RMSE for values projected quarterly by LOWRIMA on 6/25/2015

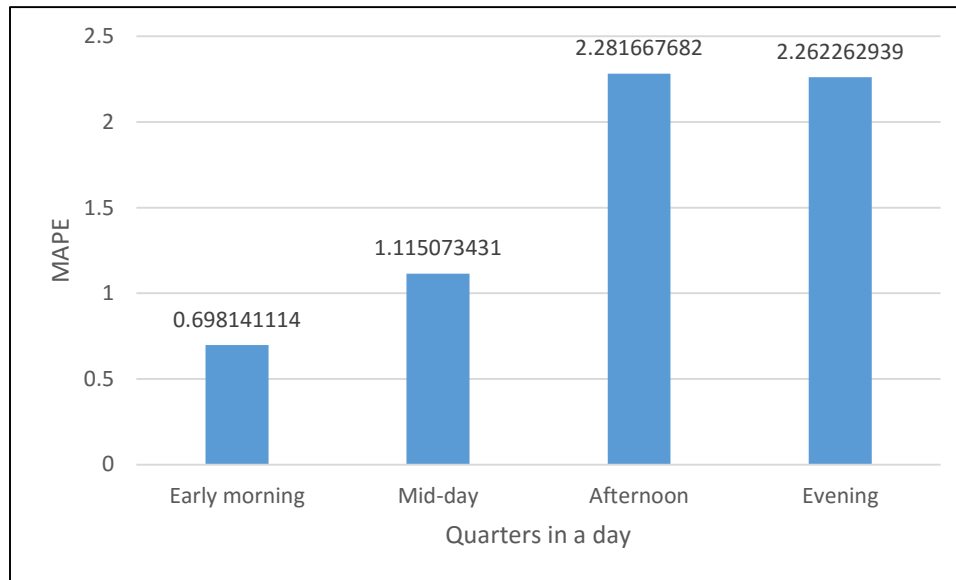


Figure 152: MAPE for values projected quarterly by LOWRIMA on 6/25/2015

Figures 153 to 156 present the error indices (MAE, MSE, RMSE and MAPE) for RFSRIMA hybrid forecasting method calculated for every six hour periods of a day, namely, early morning, mid-day, afternoon and evening.

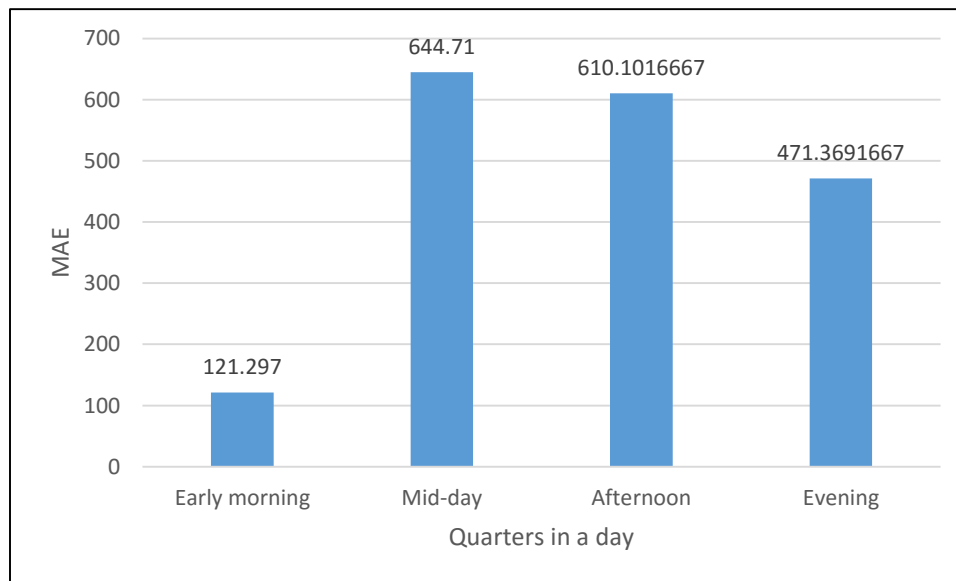


Figure 153: MAE for values projected quarterly by RFSRIMA on 6/25/2015

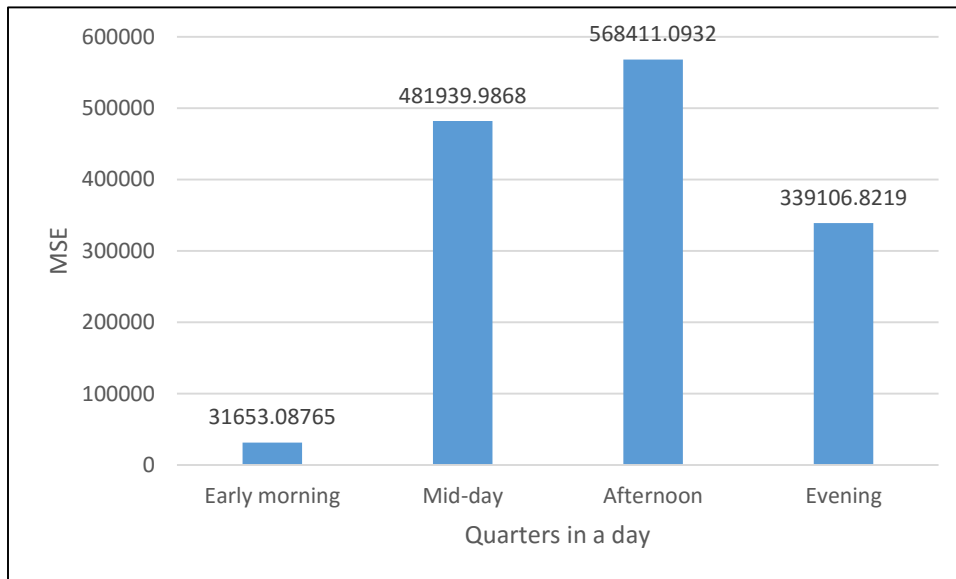


Figure 154: MAE for values projected quarterly by RFSRIMA on 6/25/2015

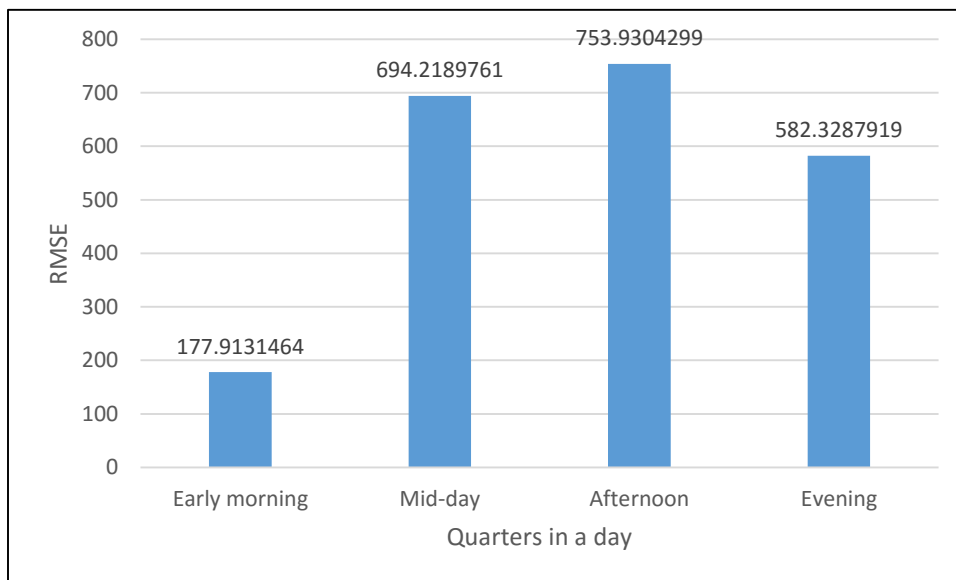


Figure 155: MAE for values projected quarterly by RFSRIMA on 6/25/2015

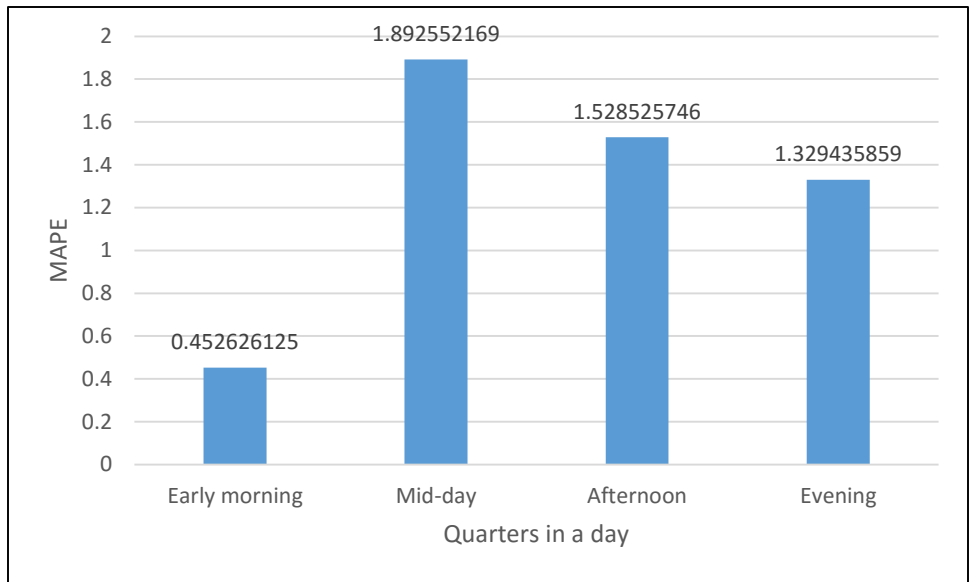


Figure 156: MAPE for values projected quarterly by RFSRIMA on 6/25/2015

VI. CONCLUSION AND FUTURE WORK

6.1 Conclusion

The conventional power grid has undergone a series of developments and evolved to the modern day smart grid for reliable and clean delivery of electricity. The vision for this development has been to integrate a data processing layer, which will allow the smart grid to have round the clock monitoring of the grid. This will not only provide solutions to get a stable power grid, but will also make existing smart grids from any probable threats [103]. With this as a background, this research was conducted. The research developed an integrated software suite (ISS) that enable system operators to monitor the grid and generate alarms in case of some contingencies. In the research, preliminary work was done on topics such as data clustering and visualization to the software suite. A novel hybrid method, namely multi-tier k-means clustering method was developed to correctly group power data. In the second part of the research, forecasting algorithms were tested and a novel hybrid forecasting technique, namely RFSRIMA, was developed that could also be integrated into the ISS [104]. The key findings of this research are as follows:

1. Developed an integrated software suite (ISS) for monitoring streaming phasor data that include a collection of software applications for state estimation, data mining, visualization, forecasting, topology management and alarm generation. The first part of this research focused on the conceptualization and development of the ISS and adding the features of data clustering and visualization to the software suite.

2. The applications were set up in such a way that the user can choose the period of phasor data that needs analysis. Once the application is initiated, it is capable of retrieving data from a local file space and perform data analysis on it, finally projecting the results in the form of a visualization scheme that would simplify the understanding of the connected network and its conditions.
3. Developed a hybrid multi-tier k-means clustering method. While integrating the data analysis software into the ISS, some data clustering algorithms (DBSCAN, k-means) were applied on the phasor data to test the clustering accuracy. Since both of these methods mis-classified the phasor data under various circumstances, the need for a novel method of clustering, suited for phasor data, was felt. The multi-tier k-means clustering method was developed and tested under four conditions (steady state, heavy load, light load and single line to ground). In each of these, it out performed the other two methods from the point of view of correctly classifying phasor data.
4. Developed a hybrid forecasting method that improved accuracies in prediction of demand data for PJM datasets. A novel, hybrid method of forecasting was developed which utilized the advantages of both RF and SARIMA methods. Such hybrid method of forecasting showed significant improvements in the accuracy of predicting demand data. This RFSRIMA method was compared with other forecasting methods like RF, SARIMA, LOWESS and another hybrid method with features of LOWESS and SARIMA (LOWRIMA) for MTLF and STLF schemes. In each of them, RFSRIMA performed better than the other methods.

5. The second part of this research investigated the performance of various forecasting algorithms applied for mid-term load forecasting (MTLF) and short-term load forecasting (STLF). Every algorithm that was tested, performed much better for STLF than MTLF. It was inferred from the overall results that in order to improve accuracy of MTLF method, more decision variables should be included along with weather correction factors for specific hours of the day. For STLF, the accuracy was high (greater than 95%) in all the cases, however, in competitive trading markets it is always advisable to strive for higher accuracies. Hence, it was inferred that further investigation needs to be conducted to improve on the accuracies obtained from RFSRIMA method.

The three clustering algorithms, DBSCAN, k-means and multi-tier k-means were tested for their rate of mis-classification (MC) under various load conditions. Here MC refers to the event of including datapoints in the wrong cluster. Their performance could be tabulated as in table 69.

Table 69: Performance analysis of clustering algorithms

Load Condition	DBSCAN	k-means	Multi-tier k-means
Normal Load	Okay (MC ~ 50%)	Fail (MC > 80%)	Best (MC < 20%)
Heavy load	Best (MC < 20%)	Fail (MC > 80%)	Okay (MC ~ 50%)
Light load	Best (MC < 20%)	Fail (MC > 80%)	Okay (MC ~ 50%)
Fault (SLG)	Fail (MC > 80%)	Fail (MC > 80%)	Best (MC < 20%)

6.2 Future Work

This research work is only a preliminary investigation on phasor data analysis. With the application of the novel clustering methods to PMU datasets, the work has shown real-time control of the grid is possible. The proposed multi-tier k-means algorithm needs to be re-organized in a way to run continuously at regular intervals. Thus, generating data charts for system operators to guide them accurately about the condition of the grid. The work

still needs to be implemented in real grid conditions to test its effectiveness. For load forecasting, the entire RFSRIMA experiment is done manually starting from data collection to model selection, it would be very useful if the whole process could be automated at periodic intervals. The objective of the research was to lay ground work for the development of an ISS which would be an open source, versatile software platform that can work as the interface of various software programs meant to implement grid system analysis applications. Presently, software platforms like MATLAB, OMNet++, R and Visual Studio have been successfully integrated to the system and are working efficiently to analyse phasor data, create bus topologies and generate alarms [105]. Further work is required in this area to facilitate a seamless integration and automation of the ISS.

PUBLICATIONS

Transactions/Journals:

1. R. Vallakati, **A. Mukherjee**, and P. Ranganathan (2015) “**Situational Awareness using DBSCAN in Smart-Grid**”, *Smart Grid and Renewable Energy, Vol.6 No.5 2015*

Web link: <http://dx.doi.org/10.4236/sgre.2015.65011>

Conference Publications:

1. R. Vallakati, **A. Mukherjee** and P. Ranganathan, “**Preserving Observability in Smart Grid using Phasor Measurement Unit by applying Optimal Redundancy Criteria (ORC)**” in *IEEE First ICDCM Conference, Atlanta, June 7-10, 2015*.
2. **A. Mukherjee**, R. Vallakati and P. Ranganathan, “**Using Phasor data for Visualization and Data Mining Purposes in Power Systems**”, in *IEEE First ICDCM Conference, Atlanta, June 7-10, 2015*.
3. Vallakati, Ranganath; **Mukherjee, Anupam**; Ranganathan, Prakash, “**A density based clustering scheme for situational awareness in a smart-grid**”, *IEEE International Conference on Electro/Information Technology (EIT)*, 2015.
4. R. Mahmud, R. Vallakati, **A. Mukherjee**, P. Ranganathan, and A. Nejadpak, “**A Survey on Attacks on Smart Grid Metering Infra-Structures: Threats and Solutions**”, in *IEEE EIT Conference, Illinois, May 23- 25 2015*.

5. N. Gellerman, P. Ranganathan, R. Vallakati, and A. Mukherjee, “**User interface for situational awareness of openPDC,**” in 2014 *North American Power Symposium (NAPS)*, pp. 1–6, 2014.

Web link: <http://dx.doi.org/10.1109/NAPS.2014.6965418>

APPENDICES

APPENDIX A

CLUSTER FORMATION AND DATA VISUALIZATION

1. C# CODE FOR DATA ACQUISITION FROM openPDC

```
using System;
using System.IO;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;
using GSF;//includes for data transport
using GSF.TimeSeries;
using GSF.TimeSeries.Transport;
using GSF.Historian.Files;
using GSF.Historian;
using Excel = Microsoft.Office.Interop.Excel;
using System.Reflection;
using System.IO.Pipes;

namespace DataReadTemplate
{
    class Program
    {
        static void Main(string[] args)
        {
            Stopwatch stopwatch = Stopwatch.StartNew();
            string filePath1 = "C:\\Users\\Anupam\\Desktop\\Research\\EPEC\\Bus
            Animation\\" + "data_feed_Vm.csv";
            string filePath2 = "C:\\Users\\Anupam\\Desktop\\Research\\EPEC\\Bus
            Animation\\" + "data_feed_Va.csv";
            string filePath3 = "C:\\Users\\Anupam\\Desktop\\Research\\EPEC\\Bus
            Animation\\" + "data_feed_Im.csv";
            string filePath4 = "C:\\Users\\Anupam\\Desktop\\Research\\EPEC\\Bus
            Animation\\" + "data_feed_Ia.csv";
```



```

string filePath5 = "C:\\Users\\Anupam\\Desktop\\Research\\EPEC\\Bus
Animation\\" + "data_feed_F.csv";
if (!File.Exists(filePath1))
{
    File.Create(filePath1).Close();
}
if (!File.Exists(filePath2))
{
    File.Create(filePath2).Close();
}
if (!File.Exists(filePath3))
{
    File.Create(filePath3).Close();
}
if (!File.Exists(filePath4))
{
    File.Create(filePath4).Close();
}
if (!File.Exists(filePath5))
{
    File.Create(filePath5).Close();
}
System.IO.File.WriteAllText(@filePath1, string.Empty);
System.IO.File.WriteAllText(@filePath2, string.Empty);
System.IO.File.WriteAllText(@filePath3, string.Empty);
System.IO.File.WriteAllText(@filePath4, string.Empty);
System.IO.File.WriteAllText(@filePath5, string.Empty);
List<int> measIDs = new List<int> { 98, 101, 102, 105, 106};
List<string> output1 = new List<string>();
List<string> output2 = new List<string>();
List<string> output3 = new List<string>();
List<string> output4 = new List<string>();
List<string> output5 = new List<string>();
ArchiveReader archive = new ArchiveReader();
archive.Open("C:\\Program Files\\openPDC\\Archive\\ppa_archive.d");
DateTime DT = new DateTime();
DT = DateTime.UtcNow;
Parallel.ForEach(measIDs, id =>
{
    foreach (IDataPoint datum in archive.ReadData(id, "*-1m", "*"))
    {
        if (id == 101)
        {
            output1.Add(datum.Value.ToString());
            output1.Add(datum.Time.ToString());
            string delimiter = ",";

```

```

string[][] output = new string[][]{
new string[]{datum.Value.ToString(),datum.Time.ToString()}};
int length1 = output.GetLength(0);
StringBuilder sb = new StringBuilder();
for (int index = 0; index < length1; index++)
{
    sb.AppendLine(string.Join(delimiter, output[index]));
}

File.AppendAllText(filePath1, sb.ToString());
}
else if (id == 102)
{
    output2.Add(datum.Value.ToString());
    output2.Add(datum.Time.ToString());
    string delimiter = ",";
    string[][] output = new string[][]{
new string[]{datum.Value.ToString(),datum.Time.ToString()}};
int length1 = output.GetLength(0);
StringBuilder sb = new StringBuilder();
for (int index = 0; index < length1; index++)
{
    sb.AppendLine(string.Join(delimiter, output[index]));
}
File.AppendAllText(filePath2, sb.ToString());
}
else if (id == 105)
{
    output3.Add(datum.Value.ToString());
    output3.Add(datum.Time.ToString());
    string delimiter = ",";
    string[][] output = new string[][]{
new string[]{datum.Value.ToString(),datum.Time.ToString()}};
int length1 = output.GetLength(0);
StringBuilder sb = new StringBuilder();
for (int index = 0; index < length1; index++)
{
    sb.AppendLine(string.Join(delimiter, output[index]));
}
File.AppendAllText(filePath3, sb.ToString());
}
else if (id == 106)
{
    output4.Add(datum.Value.ToString());
    output4.Add(datum.Time.ToString());

```

```

string delimiter = ",";
string[][] output = new string[][]{
new string[] { datum.Value.ToString(), datum.Time.ToString() } };
int length1 = output.GetLength(0);
StringBuilder sb = new StringBuilder();
for (int index = 0; index < length1; index++)
{
    sb.AppendLine(string.Join(delimiter, output[index]));
}
File.AppendAllText(filePath4, sb.ToString());
}
else if (id == 98)
{
    output5.Add(datum.Value.ToString());
    output5.Add(datum.Time.ToString());
    string delimiter = ",";
    string[][] output = new string[][]{
new string[] { datum.Value.ToString(), datum.Time.ToString() } };
int length1 = output.GetLength(0);
StringBuilder sb = new StringBuilder();
for (int index = 0; index < length1; index++)
{
    sb.AppendLine(string.Join(delimiter, output[index]));
}
File.AppendAllText(filePath5, sb.ToString());
}
}
});
stopwatch.Stop();
Console.WriteLine(stopwatch.ElapsedMilliseconds);
Console.ReadKey();
}
}
}

```

2. MATLAB CODE FOR CLUSTERING AND VISUALIZATION

The data acquisition process is common for both voltage and current. After the data is collected, the matlab code for clustering and visualization with multi-tier k-means is implemented.

Input = ‘.csv’ file from a file location. The file can contain current or voltage data.

2.1 MATLAB CODE FOR VOLTAGE DATA

```
% Data input from .CSV magnitude and angle
delimiter = ',';
formatSpec = '%f%s%[\n\r]';

filename1 =
'C:\Users\ranganath.vallakati\Documents\Classification\Test_Folder\data_feed_Vm.csv';
fileID = fopen(filename1,'r');
dataArray1 = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'EmptyValue', NaN,
'ReturnOnError', false);
d1 = dataArray1{:,1};
t1 = char(dataArray1{:,2});
t_1(1:length(t1),1) = datenum(t1(:,:));
filename2 =
'C:\Users\ranganath.vallakati\Documents\Classification\Test_Folder\data_feed_Va.csv';
fileID = fopen(filename2,'r');
dataArray2 = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'EmptyValue', NaN,
'ReturnOnError', false);
d2 = dataArray2{:,1};
t2 = char(dataArray2{:,2});
t_2(1:length(t2),1) = datenum(t2(:,:));
%% Matching both magnitude and angle and rounding off data to integers
if length(d1) >= length(d2)
    pdata = zeros(length(d1),4);
end
if length(d1) < length(d2)
    pdata = zeros(length(d2),4);
end
pdata(1:length(d1),1) = d1(:,1);
pdata(:,2) = 0;
pdata(1:length(d2),3) = d2(1:length(d2),1);
pdata(:,4) = 0;
pdata(1:length(d1),2) = t_1(:,1);
pdata(1:length(d2),4) = t_2(:,1);
for i = 1:length(pdata)
    if pdata(i,2) == 0
        pdata(i,2) = NaN;
        pdata(i,1) = NaN;
    end
    if pdata(i,4) == 0
        pdata(i,4) = NaN;
        pdata(i,3) = NaN;
    end
end
end
data1 = NaN(length(pdata),4);
```

```

for i = 1:length(pdata)
    for j = 1:length(pdata)
        if ~isnan(pdata(i,2)) && ~isnan(pdata(i,1))
            if pdata(i,2) == pdata(j,4) && ~isnan(pdata(j,3))
                data1(i,1) = pdata(i,1);
                data1(i,2) = pdata(i,2);
                data1(i,3) = pdata(j,3);
                data1(i,4) = pdata(j,4);
                t_data(i,:) = t1(i,:);
            end
        end
    end
end
data1(isnan(data1(:,1)),:) = [];
j = 1;
for i = 1:length(data1)
    if data1(i,2) ~= 0
        data(j,:) = data1(i,:);
        t_d_str(j,:) = t_data(i,:);
        j = j+1;
    end
end
x(:,1) = data(:,1);
x(:,2) = data(:,3);
end
t_d_str = datestr(datenum(t_d_str),'dd mmm yyyy HH:MM:SS.FFF');
disp('Data Acquisition and pre-processing time: ')
toc
Centroid_val = 'Manually enter centroid values? Y, N \n';
CV = input(Centroid_val,'s');
centr = zeros(3,2);
Ideal_Val = 'Ideal Value of Tx line voltage you expect:\n';
V = input(Ideal_Val);
Set_Percent = 'Set the percentage separation of initial centroids from Transmission Line Voltage \n';
SP = input(Set_Percent);
Dist = 'What kind of distance metric do you want to use? \n 1. Euclidean \n 2. Manhattan \n 3. Cosine Similarity \n 4. Vector Sum \n 5. L2 Norm \n';
D = input(Dist);
tic
%% Initializing centroids based on number of centroids chosen and data coming in.
if CV == 'Y'
    disp('Please Enter 3 values of centroids');
    for i = 1:3
        disp(i)
        centroid_vals = ' Enter Magnitude ';
    end
end

```

```

        centr(i,1) = input(centroid_vals);
        centroid_vals = ' Enter Phase ';
        centr(i,2) = input(centroid_vals);
    end
elseif CV == 'N'
    centr(1,1) = V;
    centr(1,2) = mean(x(:,2));
    centr(2,1) = (1+SP/100)*V;
    centr(2,2) = (1+SP/100)*mean(x(:,2));
    centr(3,1) = (1-SP/100)*V;
    centr(3,2) = (1-SP/100)*mean(x(:,2));
else
    disp('Enter Correct choice!')
end
%% Creating centroids and updating Cluster
test = 0;
while test ~= 3
    sep = NaN;
    count = 0;
    pdist = inf;
    sum_n_mean = zeros(1,2);
    test = 0;
    for i = 1:length(x)
        for j = 1:3
            X = (centr(j,1)-x(i,1))^2+(centr(j,2)-x(i,2))^2;
            switch D
                case 1
                    sep = sqrt(X);
                case 2
                    sep = abs((centr(j,1)-x(i,1)))+abs((centr(j,2)-x(i,2)));
                case 3
                    sep = abs((centr(j,1)-
x(i,1)))*(((centr(j,2)*x(i,2)))/(abs(centr(j,2))*abs(x(i,2))));
                case 4
                    sep = (centr(j,1))^2+(x(i,1))^2+2*centr(j,1)*x(i,1)*cos((centr(j,2)-
x(i,2))*pi/180);
                otherwise
                    sep = X;
            end
            if ~isnan(sep) && sep < pdist
                pdist = sep;
                x(i,3) = j;
            end
        end
    end
    pdist = inf;
end
end

```

```

% Shifting Centroids
for i = 1:3
    for j = 1:length(x)
        if x(j,3) == i
            sum_n_mean(1,1) = sum_n_mean(1,1)+x(j,1);
            sum_n_mean(1,2) = sum_n_mean(1,2)+x(j,2);
            count = count+1;
        end
    end
    sum_n_mean(1,1) = sum_n_mean(1,1)/count;
    sum_n_mean(1,2) = sum_n_mean(1,2)/count;
    centr1(i,1) = sum_n_mean(1,1);
    centr1(i,2) = sum_n_mean(1,2);
    count = 0;
    sum_n_mean = zeros(1,2);
end
for i = 1:3
    if abs(centr1(i,1)-centr(i,1)) > 0 || abs(centr1(i,2)-centr(i,2)) > 0
        centr(i,1) = centr1(i,1);
        centr(i,2) = centr1(i,2);
    elseif (abs(centr1(i,1)-centr(i,1)) <= 0 && abs(centr1(i,2)-centr(i,2)) <= 0) ||
isnan(centr1(i,1))
        test = test+1;
    end
end
end
end
disp('K-Means computation time:')
toc
tic
%% Grouping Data for Visualization
sz = get(0,'ScreenSize');
strng = strcat('Cluster of voltages');
fig = figure('Position',[1 1 sz(3) sz(4)],'name',strng,'NumberTitle','off');
fill = [0 0 1;0 1 1;0 1 0;1 1 0;1 0 0;1 0 1;1 1 0;1 1 1];
brdr = [0 0 1;0 1 1;0 1 0;1 1 0;1 0 0;1 1 0;1 0 1;0 0 0];
subplot(1,2,1)
temp = 1;
plot_data1 = nan(length(x),3);
plot_data2 = nan(length(x),3);
plot_data3 = nan(length(x),3);
plot_data4 = nan(length(x),3);
plot_data5 = nan(length(x),3);
data_table = zeros(8,5);
data_table(4,:) = V+V;
data_table(6,:) = 1000;
chk_gd = 0; chk_lbd = 0; chk_hbd = 0; chk_lod = 0; chk_hod = 0;

```

```

unq = unique(x(:,3));
if length(unique(x(:,3))) > 1
for i = 1:3
    mx = max(x(x(:,3)==i,1));
    mn = min(x(x(:,3)==i,1));
    if ~isempty(mx) || ~isempty(mn)
        if mx <= (1+SP/100)*V && mn >= (1-SP/100)*V
            data_table(1,3) = data_table(1,3)+length(x(x(:,3)==i,1));
            data_table(3,3) = max(mx,data_table(3,3));
            data_table(4,3) = min(mn,data_table(4,3));
            data_table(5,3) = max(max(x(x(:,1))==max(x(x(:,3)==i,1)),2),data_table(5,3));
            data_table(6,3) = min(min(x(x(:,1))==min(x(x(:,3)==i,1)),2),data_table(6,3));
            chk_gd =chk_gd+1;
            if chk_gd > 1
                data_table(7,3) = (data_table(7,3)+centr1(i,1))/chk_gd;
                data_table(8,3) = (data_table(8,3)+centr1(i,2))/chk_gd;
                plot_data1 = vertcat(plot_data1,[x(x(:,3)==i,1) x(x(:,3)==i,2)
ones(length(x(x(:,3)==i,1)),1)]);
                plot_time1 = vertcat(plot_time1,datenum(t_d_str(x(:,3)==i,:)));
            else
                data_table(7,3) = centr1(i,1);
                data_table(8,3) = centr1(i,2);
                plot_data1 = [x(x(:,3)==i,1) x(x(:,3)==i,2) ones(length(x(x(:,3)==i,1)),1)];
                plot_time1 = datenum(t_d_str(x(:,3)==i,:));
            end
            clear mx_array mn_array
        end
        if mn > (1+SP/100)*V
            data_table(1,5) = data_table(1,5)+length(x(x(:,3)==i,1));
            data_table(3,5) = max(mx,data_table(3,5));
            data_table(4,5) = min(mn,data_table(4,5));
            data_table(5,5) = max(max(x(x(:,1))==max(x(x(:,3)==i,1)),2),data_table(5,5));
            data_table(6,5) = min(min(x(x(:,1))==min(x(x(:,3)==i,1)),2),data_table(6,5));
            chk_hbd = chk_hbd+1;
            if chk_hbd > 1
                data_table(7,5) = (data_table(7,5)+centr1(i,1))/chk_hbd;
                data_table(8,5) = (data_table(8,5)+centr1(i,2))/chk_hbd;
                plot_data2 = vertcat(plot_data2,[x(x(:,3)==i,1) x(x(:,3)==i,2)
2*ones(length(x(x(:,3)==i,2)),1)]);
                plot_time2 = vertcat(plot_time2,datenum(t_d_str(x(:,3)==i,13:24)));
            else
                data_table(7,5) = centr1(i,1);
                data_table(8,5) = centr1(i,2);
                plot_data2 = [x(x(:,3)==i,1) x(x(:,3)==i,2) 2*ones(length(x(x(:,3)==i,2)),1)];
                plot_time2 = datenum(t_d_str(x(:,3)==i,:));
            end
        end
    end
end

```



```

clear mx_array mn_array
end
if mx < (1-SP/100)*V
data_table(1,1) = data_table(1,1)+length(x(x(:,3)==i,1));
data_table(3,1) = max(mx,data_table(3,1));
data_table(4,1) = min(mn,data_table(4,1));
data_table(5,1) = max(max(x(x(:,1)==max(x(x(:,3)==i,1)),2)),data_table(5,1));
data_table(6,1) = min(min(x(x(:,1)==min(x(x(:,3)==i,1)),2)),data_table(6,1));
chk_lbd = chk_lbd+1;
if chk_lbd > 1
data_table(7,1) = (data_table(7,1)+centr1(i,1))/chk_lbd;
data_table(8,1) = (data_table(8,1)+centr1(i,2))/chk_lbd;
plot_data3 = vertcat(plot_data3,[x(x(:,3)==i,1) x(x(:,3)==i,2)
3*ones(length(x(x(:,3)==i,2)),1)]);
plot_time3 = vertcat(plot_time3,datenum(t_d_str(x(:,3)==i,:)));
else
data_table(7,1) = centr1(i,1);
data_table(8,1) = centr1(i,2);
plot_data3 = [x(x(:,3)==i,1) x(x(:,3)==i,2) 3*ones(length(x(x(:,3)==i,2)),1)];
plot_time3 = datenum(t_d_str(x(:,3)==i,:));
end
clear mx_array mn_array
end
if (mx >= (1-SP/100)*V && mx <= (1+SP/100)*V) && mn < (1-SP/100)*V
oll = x(x(:,3)==i,1:2);
t_oll = t_d_str(x(:,3)==i,:);
for j = 1:2
if j == 1
data_table(1,3) = data_table(1,3)+length(oll(oll(:,1)>=(1-SP/100)*V,1));
data_table(3,3) = max(max(oll(oll(:,1)>=(1-
SP/100)*V,1)),data_table(3,3));
data_table(4,3) = min(min(oll(oll(:,1)>=(1-
SP/100)*V,1)),data_table(4,3));
data_table(5,3) = max(max(oll(oll(:,1)==max(oll(oll(:,1)>=(1-
SP/100)*V,1)),2)),data_table(5,3));
data_table(6,3) = min(min(oll(oll(:,1)==min(oll(oll(:,1)>=(1-
SP/100)*V,1)),2)),data_table(6,3));
chk_gd = chk_gd+1;
if chk_gd > 1
data_table(7,3) = (data_table(7,3)+mean(oll(oll(:,1)>=(1-
SP/100)*V,1)))/chk_gd;
data_table(8,3) = (data_table(8,3)+mean(oll(oll(:,1)>=(1-
SP/100)*V,2)))/chk_gd;
plot_data1 = vertcat(plot_data1,[oll(oll(:,1)>=(1-SP/100)*V,1)
oll(oll(:,1)>=(1-SP/100)*V,2) ones(length(oll(oll(:,1)>=(1-SP/100)*V,1)),1)]);

```

```

        plot_time1 = vertcat(plot_time1,datenum(t_oll(oll(:,1))>=(1-
SP/100)*V,:));
    else
        data_table(7,3) = mean(oll(oll(:,1))>=(1-SP/100)*V,1));
        data_table(8,3) = mean(oll(oll(:,1))>=(1-SP/100)*V,2));
        plot_data1 = [oll(oll(:,1))>=(1-SP/100)*V,1) oll(oll(:,1))>=(1-
SP/100)*V,2) ones(length(oll(oll(:,1))>=(1-SP/100)*V,1),1)];
        plot_time1 = datenum(t_oll(oll(:,1))>=(1-SP/100)*V,:));
    end
    clear mx_array mn_array
else
    data_table(1,2) = data_table(1,2)+length(oll(oll(:,1))<(1-SP/100)*V,1));
    data_table(3,2) = max(max(oll(oll(:,1))<(1-SP/100)*V,1),data_table(3,2));
    data_table(4,2) = min(min(oll(oll(:,1))<(1-SP/100)*V,1),data_table(4,2));
    data_table(5,2) = max(max(oll(oll(:,1))>=max(oll(oll(:,1))<(1-
SP/100)*V,1),2)),data_table(5,2));
    data_table(6,2) = min(min(oll(oll(:,1))>=min(oll(oll(:,1))<(1-
SP/100)*V,1),2)),data_table(6,2));
    chk_lod = chk_lod+1;
    if chk_lod > 1
        data_table(7,2) = (data_table(7,2)+mean(oll(oll(:,1))<(1-
SP/100)*V,1)))/chk_lod;
        data_table(8,2) = (data_table(8,2)+mean(oll(oll(:,1))<(1-
SP/100)*V,2)))/chk_lod;
        plot_data4 = vertcat(plot_data4,[oll(oll(:,1))<(1-SP/100)*V,1)
oll(oll(:,1))<(1-SP/100)*V,2) 4*ones(length(oll(oll(:,1))<(1-SP/100)*V,1),1)];
        plot_time4 = vertcat(plot_time4,datenum(t_oll(oll(:,1))<(1-
SP/100)*V,:));
    else
        data_table(7,2) = mean(oll(oll(:,1))<(1-SP/100)*V,1));
        data_table(8,2) = mean(oll(oll(:,1))<(1-SP/100)*V,2));
        plot_data4 = [oll(oll(:,1))<(1-SP/100)*V,1) oll(oll(:,1))<(1-SP/100)*V,2)
4*ones(length(oll(oll(:,1))<(1-SP/100)*V,1),1)];
        plot_time4 = datenum(t_oll(oll(:,1))<(1-SP/100)*V,:));
    end
end
end
clear mx_array mn_array
end
if mx > (1+SP/100)*V && (mn >= (1-SP/100)*V && mn <= (1+SP/100)*V)
    olh = x(x(:,3)==i,1:2);
    t_olh = t_d_str(x(:,3)==i,:);
    for j = 1:2
        if j == 1
            data_table(1,3) = data_table(1,3)+length(olh(olh(:,1))<=(1+SP/100)*V,1));

```

```

        data_table(3,3) =
max(max(olh(olh(:,1)<=(1+SP/100)*V,1)),data_table(3,3));
        data_table(4,3) =
min(min(olh(olh(:,1)<=(1+SP/100)*V,1)),data_table(4,3));
        data_table(5,3) =
max(max(olh(olh(:,1)==max(olh(olh(:,1)<=(1+SP/100)*V,1)),2)),data_table(5,3));
        data_table(6,3) =
min(min(olh(olh(:,1)==min(olh(olh(:,1)<=(1+SP/100)*V,1)),2)),data_table(6,3));
        if chk_gd > 1
            chk_gd = chk_gd+1;
            data_table(7,3) =
(data_table(7,3)+mean(olh(olh(:,1)<=(1+SP/100)*V,1)))/chk_gd;
            data_table(8,3) =
(data_table(8,3)+mean(olh(olh(:,1)<=(1+SP/100)*V,2)))/chk_gd;
            plot_data1 = vertcat(plot_data1,[olh(olh(:,1)<=(1+SP/100)*V,1)
olh(olh(:,1)<=(1+SP/100)*V,2) ones(length(olh(olh(:,1)<=(1+SP/100)*V,1)),1)]);
            plot_time1 =
vertcat(plot_time1,datenum(t_olh(olh(:,1)<=(1+SP/100)*V,:)));
        else
            chk_gd = chk_gd+1;
            data_table(7,3) = mean(olh(olh(:,1)<=(1+SP/100)*V,1));
            data_table(8,3) = mean(olh(olh(:,1)<=(1+SP/100)*V,2));
            plot_data1 = [olh(olh(:,1)<=(1+SP/100)*V,1)
olh(olh(:,1)<=(1+SP/100)*V,2) ones(length(olh(olh(:,1)<=(1+SP/100)*V,1)),1)];
            plot_time1 = datenum(t_olh(olh(:,1)<=(1+SP/100)*V,:));
        end
    else
        data_table(1,4) = data_table(1,4)+length(olh(olh(:,1)>(1+SP/100)*V,1));
        data_table(3,4) =
max(max(olh(olh(:,1)>(1+SP/100)*V,1)),data_table(3,4));
        data_table(4,4) =
min(min(olh(olh(:,1)>(1+SP/100)*V,1)),data_table(4,4));
        data_table(5,4) =
max(max(olh(olh(:,1)==max(olh(olh(:,1)>(1+SP/100)*V,1)),2)),data_table(5,4));
        data_table(6,4) =
min(min(olh(olh(:,1)==min(olh(olh(:,1)>(1+SP/100)*V,1)),2)),data_table(6,4));
        if chk_hod > 1
            chk_hod = chk_hod+1;
            data_table(7,4) =
(data_table(7,4)+mean(olh(olh(:,1)>(1+SP/100)*V,1)))/chk_hod;
            data_table(8,4) =
(data_table(8,4)+mean(olh(olh(:,1)>(1+SP/100)*V,2)))/chk_hod;
            plot_data5 = vertcat(plot_data5,[olh(olh(:,1)>(1+SP/100)*V,1)
olh(olh(:,1)>(1+SP/100)*V,2) ones(length(olh(olh(:,1)>(1+SP/100)*V,1)),1)]);
            plot_time5 =
vertcat(plot_time,datenum(t_olh(olh(:,1)>(1+SP/100)*V,:)));

```

```

else
    chk_hod = chk_hod+1;
    data_table(7,4) = mean(olh(olh(:,1)>(1+SP/100)*V,1));
    data_table(8,4) = mean(olh(olh(:,1)>(1+SP/100)*V,2));
    plot_data5 = [olh(olh(:,1)>(1+SP/100)*V,1)
olh(olh(:,1)>(1+SP/100)*V,2) ones(length(olh(olh(:,1)>(1+SP/100)*V,1)),1)];
    plot_time5 = datenum(t_olh(olh(:,1)>(1+SP/100)*V,:));
end
end
end
clear mx_array mn_array
end
end
else
mx = max(x(:,1));
mn = min(x(:,1));
if mx <= (1+SP/100)*V && mn >= (1-SP/100)*V
    h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
    dateFormat = 13;
    datetick('x',dateFormat,'kepticks');
    set(h,'MarkerFaceColor',fll(3,:), 'MarkerEdgeColor',brdr(3,:));
    xlabel('Time');
    ylabel('Phase Angle');
    zlabel('Voltage Magnitude');
    labl{1,temp} = strcat('Good data');
    temp = temp+1;
    hold on
    data_table(1,3) = length(x(:,1));
    data_table(3,3) = mx;
    data_table(4,3) = mn;
    data_table(5,3) = max(x(x(:,1)==max(x(x(:,3))==1,1),2));
    data_table(6,3) = min(x(x(:,1)==min(x(x(:,3))==1,1),2));
    data_table(7,3) = centr1(1,1);
    data_table(8,3) = centr1(1,2);
end
if mn > (1+SP/100)*V
    h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
    dateFormat = 13;
    datetick('x',dateFormat,'kepticks');
    set(h,'MarkerFaceColor',fll(5,:), 'MarkerEdgeColor',brdr(5,:));
    xlabel('Time');
    ylabel('Phase Angle');
    zlabel('Voltage Magnitude');
    labl{1,temp} = strcat('High value noise');
    disp(h.Marker)

```

```

temp = temp+1;
hold on
data_table(1,5) = length(x(:,1));
data_table(3,5) = mx;
data_table(4,5) = mn;
data_table(5,5) = max(x(x(:,1)==max(x(x(:,3))==1,1)),2));
data_table(6,5) = min(x(x(:,1)==min(x(x(:,3))==1,1)),2));
data_table(7,5) = centr1(1,1);
data_table(8,5) = centr1(1,2);
end
if mx < (1-SP/100)*V
h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
dateFormat = 13;
datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fll(1,:),'MarkerEdgeColor',brdr(1,:));
xlabel('Time');
ylabel('Phase Angle');
zlabel('Voltage Magnitude');
labl{1,temp} = strcat('Low value noise');
disp(h.Marker)
temp = temp+1;
hold on
data_table(1,1) = length(x(:,1));
data_table(3,1) = mx;
data_table(4,1) = mn;
data_table(5,1) = max(x(x(:,1)==max(x(x(:,3))==1,1)),2));
data_table(6,1) = min(x(x(:,1)==min(x(x(:,3))==1,1)),2));
data_table(7,1) = centr1(1,1);
data_table(8,1) = centr1(1,2);
end
end
% Plotting Data
for i = 1:5
switch(i)
case 1
if ~isnan(plot_data1(1,1))
h = scatter3(plot_time1,plot_data1(:,2),plot_data1(:,1));
dateFormat = 13;
datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fll(3,:),'MarkerEdgeColor',brdr(3,:));
set(h,'DefaultFigureVisible','off')
hold on
xlabel('Time');
ylabel('Phase Angle');
zlabel('Voltage Magnitude');
labl{1,temp} = strcat('Good Data');

```

```

        temp = temp+1;
        hold on
    end
case 2
    if ~isnan(plot_data2(1,1))
        h = scatter3(plot_time2,plot_data2(:,2),plot_data2(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fll(5,:),'MarkerEdgeColor',brdr(5,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('High Noise');
        temp = temp+1;
        hold on
    end
case 3
    if ~isnan(plot_data3(1,1))
        h = scatter3(plot_time3,plot_data3(:,2),plot_data3(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fll(1,:),'MarkerEdgeColor',brdr(1,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('Low Noise');
        temp = temp+1;
        hold on
    end
case 4
    if ~isnan(plot_data4(1,1))
        h = scatter3(plot_time4,plot_data4(:,2),plot_data4(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fll(2,:),'MarkerEdgeColor',brdr(2,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('Low Outlier');
        temp = temp+1;
    end
end

```

```

        hold on
    end
case 5
    if ~isnan(plot_data5(1,1))
        h = scatter3(plot_time4,plot_data4(:,2),plot_data4(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'kepticks');
        set(h,'MarkerFaceColor','Yellow','MarkerEdgeColor','Yellow');
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('High Outlier');
        temp = temp+1;
        hold on
    end
end
end
legend(labl);
dcm_obj = datacursormode(fig);
set(dcm_obj,'DisplayStyle','datatip','SnapToDataVertex','on','Enable','on','UpdateFcn',@m
yupdatefcn);
for i = 1:5
    data_table(2,i) = data_table(1,i)/length(x)*100;
end
data_table(4,data_table(4,:) == V+V) = 0;
data_table(6,data_table(6,:) == 1000) = 0;
old = digits(6);
data_table = double(vpa(data_table));
tic
%% Calculate Dunn's Index
f = 1;
r = 1;
length(data_table(1,:)~=0)
dist_centroids = zeros(combntns(length(data_table(1,data_table(1,:)~=0)),2),1);
for i = f:length(data_table(1,:))
    if data_table(1,i) ~= 0
        for j = (f+1):length(data_table(1,:))
            if data_table(1,j) ~= 0
                dist_centroids(r,1) = sqrt((data_table(7,i)-data_table(7,j))^2+(data_table(8,i)-
data_table(8,j))^2);
                r = r+1;
                disp(r)
            end
        end
    end
end
end

```

```

    end
    f = f+1;
end
dist_points = zeros(length(data_table(1,data_table(1,:)~=0)),1);
for i = 1:length(data_table(1,:))
    if data_table(1,i) ~= 0
        dist_points(i,1) = sqrt((data_table(3,i)-data_table(4,i))^2+(data_table(5,i)-
data_table(6,i))^2);
    end
end
dunn_idx = zeros(length(dist_points)*length(dist_centroids),1);
r = 1;
for i = 1:length(dist_points)
    for j = 1:length(dist_centroids)
        dunn_idx(r,1) = dist_centroids(j,1)/dist_points(i,1);
        r = r+1;
    end
end
disp('Dunn Index calculation time:')
toc
data_table(10,1) = min(dunn_idx);

table_figure(data_table);
hold off
disp('Data plotting time:')
toc

%% Function table_figure

function [] = table_figure(data_table)
rowName = {'NUMBER OF POINTS','PERCENTAGE OF TOTAL','MAX
MAGNITUDE','MIN MAGNITUDE','MAX ANGLE','MIN ANGLE','CENTROID
MAGNITUDE','CENTROID ANGLE','RATE OF CHANGE OF VOLTAGE
MAG.','DUNN IDX'};
colName = {'BLUE','CYAN','GREEN','YELLOW','RED'};
subplot(1,2,1)
t=uitable('ColumnName',colName,'RowName',rowName,'Data',data_table);

tableextent = get(t,'Extent');
oldposition = get(t,'Position');
newposition = [oldposition(4)+700 oldposition(4)+200 tableextent(3) tableextent(4)];
set(t, 'Position', newposition);
end

```



```
%% Function myupdatefcn
```

```
function txt = myupdatefcn(emtp,event_obj)
```

```
% Customizes text of data tips
```

```
pos = get(event_obj,'Position');
```

```
txt = {'Time: ',datestr(pos(1),14),['Magnitude: ',num2str(pos(3))],['Phase: ',num2str(pos(2))]};
```

```
end
```

2.1 MATLAB CODE FOR CURRENT DATA

```
% Data input from .CSV magnitude and angle
```

```
delimiter = ',';
```

```
formatSpec = '%f%s%[\n\r]';
```

```
filename1 =
```

```
'C:\Users\ranganath.vallakati\Documents\Classification\Test_Folder\data_feed_Vm.csv';
```

```
fileID = fopen(filename1,'r');
```

```
dataArray1 = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'EmptyValue', NaN, 'ReturnOnError', false);
```

```
d1 = dataArray1{:,1};
```

```
t1 = char(dataArray1{:,2});
```

```
t_1(1:length(t1),1) = datenum(t1(:,:));
```

```
filename2 =
```

```
'C:\Users\ranganath.vallakati\Documents\Classification\Test_Folder\data_feed_Va.csv';
```

```
fileID = fopen(filename2,'r');
```

```
dataArray2 = textscan(fileID, formatSpec, 'Delimiter', delimiter, 'EmptyValue', NaN, 'ReturnOnError', false);
```

```
d2 = dataArray2{:,1};
```

```
t2 = char(dataArray2{:,2});
```

```
t_2(1:length(t2),1) = datenum(t2(:,:));
```

```
%% Matching both magnitude and angle and rounding off data to integers
```

```
if length(d1) >= length(d2)
```

```
    pdata = zeros(length(d1),4);
```

```
end
```

```
if length(d1) < length(d2)
```

```
    pdata = zeros(length(d2),4);
```

```
end
```

```
pdata(1:length(d1),1) = d1(:,1);
```

```
pdata(:,2) = 0;
```

```
pdata(1:length(d2),3) = d2(1:length(d2),1);
```

```
pdata(:,4) = 0;
```

```
pdata(1:length(d1),2) = t_1(:,1);
```

```
pdata(1:length(d2),4) = t_2(:,1);
```

```
for i = 1:length(pdata)
```

```
    if pdata(i,2) == 0
```

```

        pdata(i,2) = NaN;
        pdata(i,1) = NaN;
    end
    if pdata(i,4) == 0
        pdata(i,4) = NaN;
        pdata(i,3) = NaN;
    end
end
data1 = NaN(length(pdata),4);
for i = 1:length(pdata)
    for j = 1:length(pdata)
        if ~isnan(pdata(i,2)) && ~isnan(pdata(i,1))
            if pdata(i,2) == pdata(j,4) && ~isnan(pdata(j,3))
                data1(i,1) = pdata(i,1);
                data1(i,2) = pdata(i,2);
                data1(i,3) = pdata(j,3);
                data1(i,4) = pdata(j,4);
                t_data(i,:) = t1(i,:);
            end
        end
    end
end
data1(isnan(data1(:,1)),:) = [];
j = 1;
for i = 1:length(data1)
    if data1(i,2) ~= 0
        data(j,:) = data1(i,:);
        t_d_str(j,:) = t_data(i,:);
        j = j+1;
    end
end
x(:,1) = data(:,1);
x(:,2) = data(:,3);
end
t_d_str = datestr(datenum(t_d_str),'dd mmm yyyy HH:MM:SS.FFF');
disp('Data Acquisition and pre-processing time: ')
toc
Centroid_val = 'Manually enter centroid values? Y, N \n';
CV = input(Centroid_val,'s');
centr = zeros(3,2);
Ideal_Val = 'Ideal Value of Tx line voltage you expect:\n';
V = input(Ideal_Val);
Set_Percent = 'Set the percentage separation of initial centroids from Transmission Line Voltage \n';
SP = input(Set_Percent);

```

```

Dist = 'What kind of distance metric do you want to use? \n 1. Euclidean \n 2. Manhattan
\n 3. Cosine Similarity \n 4. Vector Sum \n 5. L2 Norm \n';
D = input(Dist);
tic
%% Initializing centroids based on number of centroids chosen and data coming in.
if CV == 'Y'
    disp('Please Enter 3 values of centroids');
    for i = 1:3
        disp(i)
        centroid_vals = ' Enter Magnitude ';
        centr(i,1) = input(centroid_vals);
        centroid_vals = ' Enter Phase ';
        centr(i,2) = input(centroid_vals);
    end
elseif CV == 'N'
    centr(1,1) = V;
    centr(1,2) = mean(x(:,2));
    centr(2,1) = (1+SP/100)*V;
    centr(2,2) = (1+SP/100)*mean(x(:,2));
    centr(3,1) = (1-SP/100)*V;
    centr(3,2) = (1-SP/100)*mean(x(:,2));
else
    disp('Enter Correct choice!')
end
%% Creating centroids and updating Cluster
test = 0;
while test ~= 3
    sep = NaN;
    count = 0;
    pdist = inf;
    sum_n_mean = zeros(1,2);
    test = 0;
    for i = 1:length(x)
        for j = 1:3
            X = (centr(j,1)-x(i,1))^2+(centr(j,2)-x(i,2))^2;
            switch D
                case 1
                    sep = sqrt(X);
                case 2
                    sep = abs((centr(j,1)-x(i,1)))+abs((centr(j,2)-x(i,2)));
                case 3
                    sep = abs((centr(j,1)-
x(i,1)))*(((centr(j,2)*x(i,2)))/(abs(centr(j,2))*abs(x(i,2))));
                case 4
                    sep = (centr(j,1))^2+(x(i,1))^2+2*centr(j,1)*x(i,1)*cos((centr(j,2)-
x(i,2))*pi/180);
            end
        end
    end
end

```

```

        otherwise
            sep = X;
        end
        if ~isnan(sep) && sep < pdist
            pdist = sep;
            x(i,3) = j;
        end
    end
    pdist = inf;
end
% Shifting Centroids
for i = 1:3
    for j = 1:length(x)
        if x(j,3) == i
            sum_n_mean(1,1) = sum_n_mean(1,1)+x(j,1);
            sum_n_mean(1,2) = sum_n_mean(1,2)+x(j,2);
            count = count+1;
        end
    end
    sum_n_mean(1,1) = sum_n_mean(1,1)/count;
    sum_n_mean(1,2) = sum_n_mean(1,2)/count;
    centr1(i,1) = sum_n_mean(1,1);
    centr1(i,2) = sum_n_mean(1,2);
    count = 0;
    sum_n_mean = zeros(1,2);
end
for i = 1:3
    if abs(centr1(i,1)-centr(i,1)) > 0 || abs(centr1(i,2)-centr(i,2)) > 0
        centr(i,1) = centr1(i,1);
        centr(i,2) = centr1(i,2);
    elseif (abs(centr1(i,1)-centr(i,1)) <= 0 && abs(centr1(i,2)-centr(i,2)) <= 0) ||
isnan(centr1(i,1))
        test = test+1;
    end
end
end
disp('K-Means computation time:')
toc
tic
%% Grouping Data for Visualization
sz = get(0,'ScreenSize');
strng = strcat('Cluster of voltages');% from't_d_str(1,1:11),' to');
% strng = strcat(strng,t_d_str(length(t_d_str),1:11));
fig = figure('Position',[1 1 sz(3) sz(4)],'name',strng,'NumberTitle','off');
fll = [0 0 1;0 1 1;0 1 0;1 1 0;1 0 0;1 0 1;1 1 0;1 1 1];
brdr = [0 0 1;0 1 1;0 1 0;1 1 0;1 0 0;1 1 0;1 0 1;0 0 0];

```

```

subplot(1,2,1)
temp = 1;
plot_data1 = nan(length(x),3);
plot_data2 = nan(length(x),3);
plot_data3 = nan(length(x),3);
plot_data4 = nan(length(x),3);
plot_data5 = nan(length(x),3);
data_table = zeros(8,5);
data_table(4,:) = V+V;
data_table(6,:) = 1000;
chk_gd = 0; chk_lbd = 0; chk_hbd = 0; chk_lod = 0; chk_hod = 0;
unq = unique(x(:,3));
if length(unique(x(:,3))) > 1
    for i = 1:3
        mx = max(x(x(:,3)==i,1));
        mn = min(x(x(:,3)==i,1));
        if ~isempty(mx) || ~isempty(mn)
            if mx <= (1+SP/100)*V && mn >= (1-SP/100)*V
                data_table(1,3) = data_table(1,3)+length(x(x(:,3)==i,1));
                data_table(3,3) = max(mx,data_table(3,3));
                data_table(4,3) = min(mn,data_table(4,3));
                data_table(5,3) = max(max(x(x(:,1)==max(x(x(:,3)==i,1),2)),data_table(5,3)));
                data_table(6,3) = min(min(x(x(:,1)==min(x(x(:,3)==i,1),2)),data_table(6,3)));
                chk_gd =chk_gd+1;
                if chk_gd > 1
                    data_table(7,3) = (data_table(7,3)+centr1(i,1))/chk_gd;
                    data_table(8,3) = (data_table(8,3)+centr1(i,2))/chk_gd;
                    plot_data1 = vertcat(plot_data1,[x(x(:,3)==i,1) x(x(:,3)==i,2)
ones(length(x(x(:,3)==i,1),1))]);
                    plot_time1 = vertcat(plot_time1,datenum(t_d_str(x(:,3)==i,:)));
                else
                    data_table(7,3) = centr1(i,1);
                    data_table(8,3) = centr1(i,2);
                    plot_data1 = [x(x(:,3)==i,1) x(x(:,3)==i,2) ones(length(x(x(:,3)==i,1),1))];
                    plot_time1 = datenum(t_d_str(x(:,3)==i,:));
                end
                clear mx_array mn_array
            end
            if mn > (1+SP/100)*V
                data_table(1,5) = data_table(1,5)+length(x(x(:,3)==i,1));
                data_table(3,5) = max(mx,data_table(3,5));
                data_table(4,5) = min(mn,data_table(4,5));
                data_table(5,5) = max(max(x(x(:,1)==max(x(x(:,3)==i,1),2)),data_table(5,5)));
                data_table(6,5) = min(min(x(x(:,1)==min(x(x(:,3)==i,1),2)),data_table(6,5)));
                chk_hbd = chk_hbd+1;
                if chk_hbd > 1

```

```

        data_table(7,5) = (data_table(7,5)+centr1(i,1))/chk_hbd;
        data_table(8,5) = (data_table(8,5)+centr1(i,2))/chk_hbd;
        plot_data2 = vertcat(plot_data2,[x(x(:,3))==i,1) x(x(:,3))==i,2)
2*ones(length(x(x(:,3))==i,2),1)];
        plot_time2 = vertcat(plot_time2,datenum(t_d_str(x(:,3))==i,13:24)));
    else
        data_table(7,5) = centr1(i,1);
        data_table(8,5) = centr1(i,2);
        plot_data2 = [x(x(:,3))==i,1) x(x(:,3))==i,2) 2*ones(length(x(x(:,3))==i,2),1)];
        plot_time2 = datenum(t_d_str(x(:,3))==i,:));
    end
    clear mx_array mn_array
end
if mx < (1-SP/100)*V
    data_table(1,1) = data_table(1,1)+length(x(x(:,3))==i,1));
    data_table(3,1) = max(mx,data_table(3,1));
    data_table(4,1) = min(mn,data_table(4,1));
    data_table(5,1) = max(max(x(x(:,1))==max(x(x(:,3))==i,1)),2),data_table(5,1));
    data_table(6,1) = min(min(x(x(:,1))==min(x(x(:,3))==i,1)),2),data_table(6,1));
    chk_lbd = chk_lbd+1;
    if chk_lbd > 1
        data_table(7,1) = (data_table(7,1)+centr1(i,1))/chk_lbd;
        data_table(8,1) = (data_table(8,1)+centr1(i,2))/chk_lbd;
        plot_data3 = vertcat(plot_data3,[x(x(:,3))==i,1) x(x(:,3))==i,2)
3*ones(length(x(x(:,3))==i,2),1)];
        plot_time3 = vertcat(plot_time3,datenum(t_d_str(x(:,3))==i,:));
    else
        data_table(7,1) = centr1(i,1);
        data_table(8,1) = centr1(i,2);
        plot_data3 = [x(x(:,3))==i,1) x(x(:,3))==i,2) 3*ones(length(x(x(:,3))==i,2),1)];
        plot_time3 = datenum(t_d_str(x(:,3))==i,:));
    end
    clear mx_array mn_array
end
if (mx >= (1-SP/100)*V && mx <= (1+SP/100)*V) && mn < (1-SP/100)*V
    oll = x(x(:,3))==i,1:2);
    t_oll = t_d_str(x(:,3))==i,:);
    for j = 1:2
        if j == 1
            data_table(1,3) = data_table(1,3)+length(oll(oll(:,1))>=(1-SP/100)*V,1));
            data_table(3,3) = max(max(oll(oll(:,1))>=(1-
SP/100)*V,1)),data_table(3,3));
            data_table(4,3) = min(min(oll(oll(:,1))>=(1-
SP/100)*V,1)),data_table(4,3));
            data_table(5,3) = max(max(oll(oll(:,1))==max(oll(oll(:,1))>=(1-
SP/100)*V,1)),2),data_table(5,3));

```

```

        data_table(6,3) = min(min(oll(oll(:,1))>=min(oll(oll(:,1))>=(1-
SP/100)*V,1)),2)),data_table(6,3));
        chk_gd = chk_gd+1;
        if chk_gd > 1
            data_table(7,3) = (data_table(7,3)+mean(oll(oll(:,1))>=(1-
SP/100)*V,1)))/chk_gd;
            data_table(8,3) = (data_table(8,3)+mean(oll(oll(:,1))>=(1-
SP/100)*V,2)))/chk_gd;
            plot_data1 = vertcat(plot_data1,[oll(oll(:,1))>=(1-SP/100)*V,1)
oll(oll(:,1))>=(1-SP/100)*V,2) ones(length(oll(oll(:,1))>=(1-SP/100)*V,1),1)];
            plot_time1 = vertcat(plot_time1,datenum(t_oll(oll(:,1))>=(1-
SP/100)*V,:));
        else
            data_table(7,3) = mean(oll(oll(:,1))>=(1-SP/100)*V,1));
            data_table(8,3) = mean(oll(oll(:,1))>=(1-SP/100)*V,2));
            plot_data1 = [oll(oll(:,1))>=(1-SP/100)*V,1) oll(oll(:,1))>=(1-
SP/100)*V,2) ones(length(oll(oll(:,1))>=(1-SP/100)*V,1),1)];
            plot_time1 = datenum(t_oll(oll(:,1))>=(1-SP/100)*V,:));
        end
        clear mx_array mn_array
    else
        data_table(1,2) = data_table(1,2)+length(oll(oll(:,1))<(1-SP/100)*V,1));
        data_table(3,2) = max(max(oll(oll(:,1))<(1-SP/100)*V,1),data_table(3,2));
        data_table(4,2) = min(min(oll(oll(:,1))<(1-SP/100)*V,1),data_table(4,2));
        data_table(5,2) = max(max(oll(oll(:,1))>=max(oll(oll(:,1))<(1-
SP/100)*V,1)),2)),data_table(5,2));
        data_table(6,2) = min(min(oll(oll(:,1))>=min(oll(oll(:,1))<(1-
SP/100)*V,1)),2)),data_table(6,2));
        chk_lod = chk_lod+1;
        if chk_lod > 1
            data_table(7,2) = (data_table(7,2)+mean(oll(oll(:,1))<(1-
SP/100)*V,1)))/chk_lod;
            data_table(8,2) = (data_table(8,2)+mean(oll(oll(:,1))<(1-
SP/100)*V,2)))/chk_lod;
            plot_data4 = vertcat(plot_data4,[oll(oll(:,1))<(1-SP/100)*V,1)
oll(oll(:,1))<(1-SP/100)*V,2) 4*ones(length(oll(oll(:,1))<(1-SP/100)*V,1),1)];
            plot_time4 = vertcat(plot_time4,datenum(t_oll(oll(:,1))<(1-
SP/100)*V,:));
        else
            data_table(7,2) = mean(oll(oll(:,1))<(1-SP/100)*V,1));
            data_table(8,2) = mean(oll(oll(:,1))<(1-SP/100)*V,2));
            plot_data4 = [oll(oll(:,1))<(1-SP/100)*V,1) oll(oll(:,1))<(1-SP/100)*V,2)
4*ones(length(oll(oll(:,1))<(1-SP/100)*V,1),1)];
            plot_time4 = datenum(t_oll(oll(:,1))<(1-SP/100)*V,:));
        end
    end
end

```

```

end
clear mx_array mn_array
end
if mx > (1+SP/100)*V && (mn >= (1-SP/100)*V && mn <= (1+SP/100)*V)
    olh = x(x(:,3)==i,1:2);
    t_olh = t_d_str(x(:,3)==i,:);
    for j = 1:2
        if j == 1
            data_table(1,3) = data_table(1,3)+length(olh(olh(:,1)<=(1+SP/100)*V,1));
            data_table(3,3) =
max(max(olh(olh(:,1)<=(1+SP/100)*V,1)),data_table(3,3));
            data_table(4,3) =
min(min(olh(olh(:,1)<=(1+SP/100)*V,1)),data_table(4,3));
            data_table(5,3) =
max(max(olh(olh(:,1))==max(olh(olh(:,1)<=(1+SP/100)*V,1)),2)),data_table(5,3));
            data_table(6,3) =
min(min(olh(olh(:,1))==min(olh(olh(:,1)<=(1+SP/100)*V,1)),2)),data_table(6,3));
            if chk_gd > 1
                chk_gd = chk_gd+1;
                data_table(7,3) =
(data_table(7,3)+mean(olh(olh(:,1)<=(1+SP/100)*V,1)))/chk_gd;
                data_table(8,3) =
(data_table(8,3)+mean(olh(olh(:,1)<=(1+SP/100)*V,2)))/chk_gd;
                plot_data1 = vertcat(plot_data1,[olh(olh(:,1)<=(1+SP/100)*V,1)
olh(olh(:,1)<=(1+SP/100)*V,2) ones(length(olh(olh(:,1)<=(1+SP/100)*V,1)),1)]);
                plot_time1 =
vertcat(plot_time1,datenum(t_olh(olh(:,1)<=(1+SP/100)*V,:)));
            else
                chk_gd = chk_gd+1;
                data_table(7,3) = mean(olh(olh(:,1)<=(1+SP/100)*V,1));
                data_table(8,3) = mean(olh(olh(:,1)<=(1+SP/100)*V,2));
                plot_data1 = [olh(olh(:,1)<=(1+SP/100)*V,1)
olh(olh(:,1)<=(1+SP/100)*V,2) ones(length(olh(olh(:,1)<=(1+SP/100)*V,1)),1)];
                plot_time1 = datenum(t_olh(olh(:,1)<=(1+SP/100)*V,:));
            end
        else
            data_table(1,4) = data_table(1,4)+length(olh(olh(:,1)>(1+SP/100)*V,1));
            data_table(3,4) =
max(max(olh(olh(:,1)>(1+SP/100)*V,1)),data_table(3,4));
            data_table(4,4) =
min(min(olh(olh(:,1)>(1+SP/100)*V,1)),data_table(4,4));
            data_table(5,4) =
max(max(olh(olh(:,1))==max(olh(olh(:,1)>(1+SP/100)*V,1)),2)),data_table(5,4));
            data_table(6,4) =
min(min(olh(olh(:,1))==min(olh(olh(:,1)>(1+SP/100)*V,1)),2)),data_table(6,4));
            if chk_hod > 1

```



```

        chk_hod = chk_hod+1;
        data_table(7,4) =
(data_table(7,4)+mean(olh(olh(:,1)>(1+SP/100)*V,1)))/chk_hod;
        data_table(8,4) =
(data_table(8,4)+mean(olh(olh(:,1)>(1+SP/100)*V,2)))/chk_hod;
        plot_data5 = vertcat(plot_data5,[olh(olh(:,1)>(1+SP/100)*V,1)
olh(olh(:,1)>(1+SP/100)*V,2) ones(length(olh(olh(:,1)>(1+SP/100)*V,1)),1)]);
        plot_time5 =
vertcat(plot_time,datenum(t_olh(olh(:,1)>(1+SP/100)*V,:)));
    else
        chk_hod = chk_hod+1;
        data_table(7,4) = mean(olh(olh(:,1)>(1+SP/100)*V,1));
        data_table(8,4) = mean(olh(olh(:,1)>(1+SP/100)*V,2));
        plot_data5 = [olh(olh(:,1)>(1+SP/100)*V,1)
olh(olh(:,1)>(1+SP/100)*V,2) ones(length(olh(olh(:,1)>(1+SP/100)*V,1)),1)];
        plot_time5 = datenum(t_olh(olh(:,1)>(1+SP/100)*V,:));
    end
end
end
clear mx_array mn_array
end
end
else
mx = max(x(:,1));
mn = min(x(:,1));
if mx <= (1+SP/100)*V && mn >= (1-SP/100)*V
    h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
    dateFormat = 13;
    datetick('x',dateFormat,'kepticks');
    set(h,'MarkerFaceColor','fill(3,:),'MarkerEdgeColor','brdr(3,:)');
    xlabel('Time');
    ylabel('Phase Angle');
    zlabel('Voltage Magnitude');
    labl{1,temp} = strcat('Good data');
    temp = temp+1;
    hold on
    data_table(1,3) = length(x(:,1));
    data_table(3,3) = mx;
    data_table(4,3) = mn;
    data_table(5,3) = max(x(x(:,1)==max(x(x(:,3))==1,1)),2));
    data_table(6,3) = min(x(x(:,1)==min(x(x(:,3))==1,1)),2));
    data_table(7,3) = centr1(1,1);
    data_table(8,3) = centr1(1,2);
end
if mn > (1+SP/100)*V

```

```

h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
dateFormat = 13;
datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor','fl(5,:),'MarkerEdgeColor','brdr(5,:)');
xlabel('Time');
ylabel('Phase Angle');
zlabel('Voltage Magnitude');
labl{1,temp} = strcat('High value noise');
disp(h.Marker)
temp = temp+1;
hold on
data_table(1,5) = length(x(:,1));
data_table(3,5) = mx;
data_table(4,5) = mn;
data_table(5,5) = max(x(x(:,1)==max(x(x(:,3))==1,1)),2));
data_table(6,5) = min(x(x(:,1)==min(x(x(:,3))==1,1)),2));
data_table(7,5) = centr1(1,1);
data_table(8,5) = centr1(1,2);
end
if mx < (1-SP/100)*V
h = scatter3(datenum(t_d_str(:,13:24)),x(:,2),x(:,1));
dateFormat = 13;
datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor','fl(1,:),'MarkerEdgeColor','brdr(1,:)');
xlabel('Time');
ylabel('Phase Angle');
zlabel('Voltage Magnitude');
labl{1,temp} = strcat('Low value noise');
disp(h.Marker)
temp = temp+1;
hold on
data_table(1,1) = length(x(:,1));
data_table(3,1) = mx;
data_table(4,1) = mn;
data_table(5,1) = max(x(x(:,1)==max(x(x(:,3))==1,1)),2));
data_table(6,1) = min(x(x(:,1)==min(x(x(:,3))==1,1)),2));
data_table(7,1) = centr1(1,1);
data_table(8,1) = centr1(1,2);
end
end
% Plotting Data
for i = 1:5
switch(i)
case 1
if ~isnan(plot_data1(1,1))
h = scatter3(plot_time1,plot_data1(:,2),plot_data1(:,1));

```

```

        dateFormat = 13;
        datetick('x',dateFormat,'keepsicks');
set(h,'MarkerFaceColor',fll(3,:), 'MarkerEdgeColor',brdr(3,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('Good Data');
        temp = temp+1;
        hold on
    end
case 2
    if ~isnan(plot_data2(1,1))
        h = scatter3(plot_time2,plot_data2(:,2),plot_data2(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'keepsicks');
set(h,'MarkerFaceColor',fll(5,:), 'MarkerEdgeColor',brdr(5,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('High Noise');
        temp = temp+1;
        hold on
    end
case 3
    if ~isnan(plot_data3(1,1))
        h = scatter3(plot_time3,plot_data3(:,2),plot_data3(:,1));
        dateFormat = 13;
        datetick('x',dateFormat,'keepsicks');
set(h,'MarkerFaceColor',fll(1,:), 'MarkerEdgeColor',brdr(1,:));
        set(h,'DefaultFigureVisible','off')
        hold on
        xlabel('Time');
        ylabel('Phase Angle');
        zlabel('Voltage Magnitude');
        labl{1,temp} = strcat('Low Noise');
        temp = temp+1;
        hold on
    end
case 4
    if ~isnan(plot_data4(1,1))
        h = scatter3(plot_time4,plot_data4(:,2),plot_data4(:,1));
        dateFormat = 13;

```

```

        datetick('x',dateFormat,'kepticks');
set(h,'MarkerFaceColor',fl(2,:),'MarkerEdgeColor',brdr(2,:));
    set(h,'DefaultFigureVisible','off')
    hold on
    xlabel('Time');
    ylabel('Phase Angle');
    zlabel('Voltage Magnitude');
    labl{1,temp} = strcat('Low Outlier');
    temp = temp+1;
    hold on
end
case 5
if ~isnan(plot_data5(1,1))
    h = scatter3(plot_time4,plot_data4(:,2),plot_data4(:,1));
    dateFormat = 13;
    datetick('x',dateFormat,'kepticks');
    set(h,'MarkerFaceColor','Yellow','MarkerEdgeColor','Yellow');
    set(h,'DefaultFigureVisible','off')
    hold on
    xlabel('Time');
    ylabel('Phase Angle');
    zlabel('Voltage Magnitude');
    labl{1,temp} = strcat('High Outlier');
    temp = temp+1;
    hold on
end
end
end
legend(labl);
dcm_obj = datacursormode(fig);
set(dcm_obj,'DisplayStyle','datatip','SnapToDataVertex','on','Enable','on','UpdateFcn',@m
yupdatefcn);
for i = 1:5
    data_table(2,i) = data_table(1,i)/length(x)*100;
end
data_table(4,data_table(4,:) == V+V) = 0;
data_table(6,data_table(6,:) == 1000) = 0;
old = digits(6);
data_table = double(vpa(data_table));
tic
%% Calculate Dunn's Index
f = 1;
r = 1;
length(data_table(1,:)~=0)
dist_centroids = zeros(combntns(length(data_table(1,data_table(1,:)~=0)),2),1);
for i = f:length(data_table(1,:))

```

```

if data_table(1,i) ~= 0
    for j = (f+1):length(data_table(1,:))
        if data_table(1,j) ~= 0
            dist_centroids(r,1) = sqrt((data_table(7,i)-data_table(7,j))^2+(data_table(8,i)-
data_table(8,j))^2);
            r = r+1;
            disp(r)
        end
    end
end
end
f = f+1;
end
dist_points = zeros(length(data_table(1,data_table(1,:)~=0)),1);
for i = 1:length(data_table(1,:))
    if data_table(1,i) ~= 0
        dist_points(i,1) = sqrt((data_table(3,i)-data_table(4,i))^2+(data_table(5,i)-
data_table(6,i))^2);
    end
end
dunn_idx = zeros(length(dist_points)*length(dist_centroids),1);
r = 1;
for i = 1:length(dist_points)
    for j = 1:length(dist_centroids)
        dunn_idx(r,1) = dist_centroids(j,1)/dist_points(i,1);
        r = r+1;
    end
end
disp('Dunn Index calculation time:')
toc
data_table(10,1) = min(dunn_idx);

table_figure(data_table);
hold off
disp('Data plotting time:')
toc

%% Function table_figure

function [] = table_figure(data_table)
rowName = {'NUMBER OF POINTS','PERCENTAGE OF TOTAL','MAX
MAGNITUDE','MIN MAGNITUDE','MAX ANGLE','MIN ANGLE','CENTROID
MAGNITUDE','CENTROID ANGLE','RATE OF CHANGE OF VOLTAGE
MAG.','DUNN IDX'};
colName = {'BLUE','CYAN','GREEN','YELLOW','RED'};
subplot(1,2,1)
t=uitable('ColumnName',colName,'RowName',rowName,'Data',data_table);

```

```

tableextent = get(t,'Extent');
oldposition = get(t,'Position');
newposition = [oldposition(4)+700 oldposition(4)+200 tableextent(3) tableextent(4)];
set(t, 'Position', newposition);
end

%% Function myupdatefcn

function txt = myupdatefcn(emtp,event_obj)
% Customizes text of data tips

pos = get(event_obj,'Position');
txt = {'Time: ',datestr(pos(1),14),['Magnitude: ',num2str(pos(3))],['Phase: ',num2str(pos(2))]};
end

```

APPENDIX B

FORECASTING OF LOAD DATA

1. MATLAB CODE FOR DATA ACQUISITION FROM '.xls' FILE

Inputs: 1. Date of first forecast. 2. Hour of the day to forecast load data. 3. Train database start date. 4. Train database end date.

```
%% Creating database from available xls files
prompt = {'Enter forecast start date:', 'Enter the hour:', 'Enter database end date:', 'Enter
database start date', 'Enter forecast end date'};
dlg_title = 'Input';
num_lines = 1;
def = {'m/d/yyyy', '24 hrs format', 'm/d/yyyy', 'm/d/yyyy'};
answer = inputdlg(prompt, dlg_title, num_lines, def);
%% Creating Training database
[database] = trainDatabase(answer);
%% Gathering Temperature data
[temp_data] = gatherTemperature();
vNames = {'Date', 'Temperature'};
T1 = cell2table(temp_data, 'VariableNames', vNames);
temp_file = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Temperature_Data.csv';
writetable(T1, temp_file);
%% Converting database to linear data
[linearData] = database2Linear(database, answer, temp_data);
%% Converting cell to table
varNames =
{'Date', 'Datenum', 'DemandP', 'Year_from_Start', 'Season', 'Month', 'Day_of_Week', 'Daytype',
'Hour', 'Temperature'};
T = cell2table(linearData, 'VariableNames', varNames);
filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Power_Data.csv';
filename1 = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Power_Data.xlsx';
writetable(T, filename);
writetable(T, filename1);
%% Creating file for inputted hour and business days
```

```

filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Gathering Data\Power_Data.xlsx';
[~,~,alldata] = xlsread(filename);
count = 1;
for i = 2:length(alldata(:,1))
    if cell2mat(alldata(i,9)) == str2double(cell2mat(answer(2,1))) &&
cell2mat(alldata(i,8)) == 1 && ~isnan(cell2mat(alldata(i,3)))
        linData_20(count,1) = alldata(i,1);
        linData_20(count,2) = alldata(i,3);
        linData_20(count,3:5) = alldata(i,5:7);
        linData_20(count,6) = alldata(i,10);
        linData_20(count,7) = alldata(i,8);
        count = count+1;
    end
end
varNames = {'Date','DemandP','Season','Month','Day_of_Week','Temperature'};
T = cell2table(linData_20(:,1:6),'VariableNames',varNames);
filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Gathering Data\';
filename = strcat(filename,cell2mat(answer(2,1)),'_Power_Data.csv');
writetable(T,filename);
%% Getting PJM Forecast
filename_PJM = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM
Data\LOWESS_ARIMA\Code for forecasting\Final Data\PJM_Forecast.xlsx';
[~,~,PJM_data] = xlsread(filename_PJM);
r = 1; t = 200;
for i = 2:length(PJM_data(:,1))
    if (month(cell2mat(PJM_data(i,1))) >= month(cell2mat(answer(5,1))) &&
day(cell2mat(PJM_data(i,1))) > day(cell2mat(answer(5,1)))) ||
(month(cell2mat(PJM_data(i,1))) > month(cell2mat(answer(5,1))) &&
day(cell2mat(PJM_data(i,1))) <= day(cell2mat(answer(5,1))))
        if isbusday(cell2mat(PJM_data(i,1))) == 1 && cell2mat(PJM_data(i,2)) == t
            pjm_pred(r,1) = PJM_data(i,1);
            pjm_pred(r,2) = PJM_data(i,3);
            r = r+1;
        end
    end
end
% Only for January 2015
r = 1; t = 2000;
for i = 2:length(PJM_data(:,1))
    if month(cell2mat(PJM_data(i,1))) == 1
        if isbusday(cell2mat(PJM_data(i,1))) == 1 && cell2mat(PJM_data(i,2)) == t
            pjm_pred(r,1) = PJM_data(i,1);
            pjm_pred(r,2) = PJM_data(i,3);
            r = r+1;
        end
    end
end

```



```

        end
    end
end
r = 1;s = 20;
for i = s:-1:1
    pjm_plot(r,1) = cell2mat(pjm_pred(i,2));
    r = r+1;
end
filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Gathering Data\January\20_PJM_Pred.xlsx';
xlswrite(filename,pjm_plot)

```

2. R CODE FOR FORECASTING

Input: '.csv' file with load and temperature data as well as indicator variables from a specific file location.

```

install.packages("randomForest")
library("randomForest")
install.packages("astsa")
library("astsa")
setwd("C:/Users/Anupam/Desktop/Research/EPEC/Prediction/PJM Data/Using Random
Forest/Matlab_R/Gathering Data")
#str(dat)
dat <- read.csv("24_Power_Data.csv",header = TRUE)
# Setting Variables
r1 <- 873; r2 <- r1+1; r3 <- r2+19; r4 <- 25
#dat_ts <- ts(dat$DemandP[1:r1])
# Applying Random Forest on the historical data
dat$season <- as.factor(dat$Season)
dat$month <- as.factor(dat$Month)
dat$dow <- as.factor(dat$Day_of_Week)
fit <- randomForest(DemandP ~ season + month + dow + Temperature, data =
dat[1:r1,],replace = TRUE, importance=TRUE, ntree=2000,mtry = 2)
pred <- predict(fit, dat[1:r1,-c(1:5)], type="response")
dat_pred <- as.data.frame(pred)
residuals_RF <- dat$DemandP[1:r1]-dat_pred
write.table(residuals_RF, file =
"C:/Users/Anupam/Desktop/Research/EPEC/Prediction/PJM Data/Using Random
Forest/Matlab_R/Gathering Data/RandomForest_Residuals.csv",row.names =
FALSE,col.names = FALSE)
dat_ts <- ts(residuals_RF)
acf2(dat_ts)
dat_ts_diff <- diff(dat_ts)
acf2(dat_ts_diff)

```

```

dat_ts_diff2 <- diff(dat_ts_diff)
acf2(dat_ts_diff2)
dat_ts_diff3 <- diff(dat_ts_diff2)
acf2(dat_ts_diff3)
# Random Forest Prediction for the month
rf_pred <- as.data.frame(predict(fit, dat[r2:r3,-c(1:5)], type="response"))
rownames(rf_pred) <- 1:nrow(rf_pred)
write.table(rf_pred, file = "C:/Users/Anupam/Desktop/Research/EPEC/Prediction/PJM
Data/Using Random Forest/Matlab_R/Gathering
Data/RandomForest_pred.csv",row.names = FALSE,col.names = FALSE)
# Saving Actual Demand Data
write.table(dat$DemandP[r2:r3], file =
"C:/Users/Anupam/Desktop/Research/EPEC/Prediction/PJM Data/Using Random
Forest/Matlab_R/Gathering Data/Actual_Demand.csv",row.names = FALSE,col.names =
FALSE)
# SARIMA Prediction for the month
mod_fit <- sarima(dat_ts,1,1,1,1,1,22)
mod_pred <- as.data.frame(sarima.for(dat_ts,n.ahead = r4,1,1,1,1,1,22))
SARIMA_pred <- as.data.frame(mod_pred$pred)
write.table(SARIMA_pred, file =
"C:/Users/Anupam/Desktop/Research/EPEC/Prediction/PJM Data/Using Random
Forest/Matlab_R/Gathering Data/SARIMA_pred.csv",row.names = FALSE,col.names =
FALSE)

```

3. MATLAB CODE FOR ERROR CALCULATION AND PLOTTING DATA

```

%% Plotting Actual, Estimated, PJM Model and Error
clc
clear
filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Gathering Data\20_Power_Data.xlsx';
[~,~,plot_data] = xlsread(filename);
st = 756;ed = 775;r = 1;
filename = 'C:\Users\Anupam\Desktop\Research\EPEC\Prediction\PJM Data\Using
Random Forest\Matlab_R\Gathering Data\Calculations.xlsx';
[~,~,read_data] = xlsread(filename);
l = 21;
for i = st:ed
    datePlot(r,1) = datenum(cell2mat(plot_data(i,1)));
    r = r+1;
end
% Plot Actual, Estimated and PJM Model
figure()
plot(datePlot(:,1),cell2mat(read_data(2:l,4)), 'red') % Actual
hold on
plot(datePlot(:,1),cell2mat(read_data(2:l,3)), 'black') % Model

```

```

hold on
plot(datePlot(:,1),cell2mat(read_data(2:1,8)), 'yellow') % PJM
legend('Actual Data', 'Model Prediction', 'PJM Prediction')
dateFormat = 2;
datetick('x',dateFormat)
% Plot Error percentages
figure()
plot(datePlot(:,1),cell2mat(read_data(2:1,6)), 'red') % Model
hold on
plot(datePlot(:,1),cell2mat(read_data(2:1,10)), 'yellow') % PJM
legend('Model Error', 'PJM Error')
dateFormat = 2;
datetick('x',dateFormat)
%% Calculating standardized error
T = cell2mat(read_data(2:1,4));
P = cell2mat(read_data(2:1,3));
V2(1,1) = errperf(T,P, 'mae');
V2(2,1) = errperf(T,P, 'mse');
V2(3,1) = errperf(T,P, 'rmse');
V2(3,1) = errperf(T,P, 'rmse');
V2(4,1) = errperf(T,P, 'mape');
P = cell2mat(read_data(2:1,8));
V2(1,2) = errperf(T,P, 'mae');
V2(2,2) = errperf(T,P, 'mse');
V2(3,2) = errperf(T,P, 'rmse');
V2(4,2) = errperf(T,P, 'mape');

```

REFERENCES

- [1] Toward a Smart Grid: Power Delivery for the 21st Century. Massoud Amin and Bruce F. Wollenberg in *IEEE Power and Energy Magazine*, Vol. 3, No. 5, pages 34–41; September/October 2005.
- [2] <https://pediapress.com/books/show/smart-grid-fernan-f-pusung/>
- [3] Federal Energy Regulatory Commission Staff Report (2006-2008) (pdf). *Assessment of Demand Response and Advanced Metering* (Docket AD06-2-000) <http://www.ferc.gov/legal/staff-reports/demand-response.pdf>. United States Department of Energy. p. 20. Retrieved 2008-11-27.
- [4] National Energy Technology Laboratory (2007-2008) (pdf). *NETL Modern Grid Initiative – Powering Our 21st -Century Economy* http://www.netl.doc.gov/smartgrid/referenceshelf/whitepapers/Modern%20Grid%20Benefits_Final_v1_0.pdf. United States Department of Energy Office of Electricity Delivery and Energy Reliability. p. 17. Retrieved 2008-12-06.
- [5] “How Smart Grid Works.” [Online]. Available: <http://www.nature.com/news/2008/080730/images/454570a-6.jpg>. [Accessed: 01-Jan-2015].
- [6] A. G. Phadke, “Synchronized phasor measurements in power systems,” *IEEE Comput. Appl. Power*, vol. 6, no. 2, pp. 10–15, Apr. 1993.
- [7] Mangapathirao V. Mynam, Ala Harikrishna, and Vivek Singh, Schweitzer Engineering Laboratories, Inc., “Synchrophasors Redefining SCADA Systems,” 13th Annual Western Power Delivery Automation Conference, March 2011.
- [8] <https://www.greentechmedia.com/research/report/smart-grid-in-2010>.
- [9] E. O. Schweitzer and D. E. Whitehead, “Real-time power system control using synchrophasors,” in *2008 61st Annual Conference for Protective Relay Engineers*, pp. 78–88, 2008.
- [10] M. V Mynam, A. Harikrishna, and V. Singh, “Synchrophasors Redefining SCADA Systems,” 2011.
- [11] Gopakumar P., Chandra S. G., Reddy M. J. B., “Optimal placement of PMUs for the smart grid implementation in Indian power grid—A case study,” *Front Energ*, vol. 7, no. 3, pp. 358-372, 2013.

- [12] <http://timreview.ca/article/702>
- [13] J. D. La Ree, S. Member, V. Centeno, J. S. Thorp, L. Fellow, and A. G. Phadke, “Synchronized Phasor Measurement Applications in Power Systems,” vol. 1, no. 1, pp. 20–27, 2010.
- [14] K. Martin and G. Brunello, “An overview of the IEEE Standard C37.118.2 — Synchrophasor Data Transfer for Power Systems,” in *2014 IEEE PES General Meeting | Conference & Exposition*, pp. 1–1, 2014.
- [15] “IEEE Guide for Phasor Data Concentrator Requirements for Power System Protection, Control, and Monitoring,” 2013.
- [16] B. Mohammadi, A. Rabiei, and S. Mobayen, “Online inter-area oscillation monitoring in power systems using PMU data and Prony analysis,” vol. 6, no. 22, pp. 5267–5272, 2011.
- [17] K. S. Shim, S. T. Kim, J. H. Lee, E. J. Choi, and J. H. Choi, “Detection of low-frequency oscillation using synchrophasor in wide-area rolling blackouts,” *Int. J. Electr. Power Energy Syst.*, vol. 63, pp. 1015–1022, Dec. 2014.
- [18] K. P. Lien, C. W. Liu, C. S. Yu, and J. A. Jiang, “Transmission network fault location observability with minimal PMU placement,” *IEEE Trans. Power Deliv.*, vol. 21, no. 3, pp. 1128–1136, 2006.
- [19] A. Reddy, “PMU based Real-Time Short Term Voltage Stability Monitoring – Analysis and Implementation on a Real- Time Test Bed,” 2007.
- [20] L. Huang, Y. Sun, J. Xu, W. Gao, J. Zhang, and Z. Wu, “Optimal PMU Placement Considering Controlled Islanding of Power System,” *IEEE Trans. Power Syst.*, vol. 29, no. 2, pp. 742–755, Mar. 2014.
- [21] R. Moxley, G. Zweigle, B. Flerchinger et al., “Turning Synchrophasor Data Into Actionable Information,” Schweitzer Engineering Laboratories, Inc. Website:<https://www.selinc.com/workarea/downloadasset.aspx?id=99372>.
- [22] R. Vallakati, A. Mukherjee, and P. Ranganathan, “Preserving observability in synchrophasors using Optimal Redundancy Criteria (ORC),” in *2015 IEEE First International Conference on DC Microgrids (ICDCM)*, pp. 69–74, 2015.
- [23] “Grid Protection Alliance.” [Online]. Available: <http://openpdc.codeplex.com/>.
- [24] “openPDC.” [Online]. Available: http://gridprotectionalliance.org/pdf/openPDC_Overview.pdf.
- [25] “Microsoft Developer Network.” [Online]. Available: <http://msdn.microsoft.com/en-us/library/>.

- [26] A. Mukherjee, R. Vallakati, V. Lachenaud, and P. Ranganathan, "Using phasor data for visualization and data mining in smart-grid applications," in *2015 IEEE First International Conference on DC Microgrids (ICDCM)*, pp. 13–18, 2015.
- [27] http://www.kdnuggets.com/data_mining_course/x1-intro-to-data-mining-notes.html
- [28] <http://www.cs.rpi.edu/~zaki/PaperDir/DMABOOK.pdf>
- [29] http://www.cs.cornell.edu/courses/cs578/2005fa/cs578_clustering_lecture.4up.pdf
- [30] http://web.stanford.edu/group/toolingup/cgi-bin/toolkit/wp-content/uploads/2011/03/GMcGhee_toolingup_DataVis_110506.pdf
- [31] Abur, A. Galvan, F. (2012) Synchro-Phasor Assisted State Estimation (SPASE). *IEEE PES Innovative Smart Grid Technologies (ISGT)*, Washington DC, 16-20 1-2, January 2012.
- [32] Zhao, L. and Abur, A., "Multi Area State Estimation Using Synchronized Phasor Measurements," *IEEE Transactions on Power Systems*, vol. 20, 611-617, 2005.
- [33] Al Karim, M., Chenine, M., Zhu, K., Nordstrom, L. and Nordström, L. (2012) Synchrophasor-Based Data Mining for Power System Fault Analysis. *3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, Berlin, 1-8, 14-17 October 2012.
- [34] Al-Mohammed, A.H. and Abido, M.A. (2014) An Adaptive Fault Location Algorithm for Power System Networks Based on Synchrophasor Measurements. *Electric Power Systems Research*, **108**, 153-163. <http://dx.doi.org/10.1016/i.epr.2013.10.013>
- [35] Burnett, R.O., Butts, M.M. and Sterlina, P.S., "Power System Applications for Phasor Measurement Units," *IEEE Computer Applications in Power*, **7**, 8-13, 1994.
- [36] Liu, X.A., Lavery, D. and Best, R. (2014) Islanding Detection Based on Probabilistic PCA with Missing Values in PMU Data, *IEEE PES General Meeting | Conference & Exposition*, National Harbor, 1-6, 27-31 July 2014.
- [37] Dasgupta, S., Paramasivam, M., Vaidya, U. and Ajjarapu, V., "Real-Time Monitoring of Short-Term Voltage Stability Using PMU Data," *IEEE PES General Meeting | Conference & Exposition*, **1**, 2014.
- [38] Yu, C.S., Liu, C.W., Yu, S.L. and Jiang, J.A., "A New PMU-Based Fault Location Algorithm for Series Compensated Lines," *IEEE Transactions on Power Delivery*, **17**, 33-46, 2002.

- [39] Messina, A.R., Member, S., Vittal, V. and Ruiz-vega, D., “Interpretation and Visualization of Wide-Area PMU Measurements Using Hilbert Analysis,” *IEEE Transactions on Power Systems*, **21**, 1763-1771, 2006.
- [40] P. Bilik, P. Repka, and M. Malohlava, “Virtual synchrophasor monitoring network,” in *2012 IEEE International Conference on Relay Engineers*, pp. 23-29, 2014.
- [41] A. J. Allen, S. –W. Sohn, S. Santoso, and W. M. Grady, “Algorithm for screening PMU data for power system events,” in *2012 3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, pp. 1-6, 2012.
- [42] K. D. Jones, A. Pal, and J. S. Thorp, “Methodology for Performing Synchrophasor Data Conditioning and Validation,” *IEEE Trans. Power Syst.*, vol. PP, no. 99, pp.1 1-10, 2014.
- [43] Q. Gao and S. M. Rovnyak, “Decision Trees Using Synchronized Phasor Measurements for Wide-Area Response Based Control,” *IEEE Trans. Power Syst.*, vol. 26, no. 2, pp. 855-861, May 2011.
- [44] A. Pal, I. Singh, and B. Bhargava, “Stress Assessment in power systems and its visualization using synchrophasor based metrics,” in *North American Power Symposium (NAPS)*, 2014, pp. 1-6, 2014.
- [45] A. Pal, J. S. Thorp, T. Khan, and S. S. Young, “Classification Trees for Complex Synchrophasor Data,” *Electr. Power Components Syst.*, vol. 41, no. 14, pp. 1381-1396, Oct. 2013.
- [46] Z. Li and W. Wu, “Phasor Measurements-Aided Decision Trees for Power System Security Assessment,” *Second International Conference on Information and Computing Science*, vol. 1, pp. 358–361, 2009.
- [47] V. Vittal, R. J. O’Keefe, M. R. Richardson, N. Bhatt, D. Stradford, and S. K. Sarawgi, “Decision Tree-Based Online Voltage Security Assessment Using PMU Measurements,” *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 832–839, May 2009.
- [48] A. J. Allen, S.-W. Sohn, S. Santoso, and W. M. Grady, “Algorithm for screening PMU data for power system events,” *3rd IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe)*, 2012, pp. 1–6, 2012.
- [49] M. Khan, M. Li, P. Ashton, G. Taylor, and J. Liu, “Big Data Analytics on PMU Measurements,” 2014.
- [50] A. Abur and F. Galvan, “Synchro-Phasor Assisted State Estimation (SPASE),” in *IEEE PES Innovative Smart Grid Technologies (ISGT)*, 2012, pp. 1–2, 2012.
- [51] S. Hossain and H. Kirkham, “A prediction method for power frequency quality based on Bayesian theorem and uncertainty classification,” in *2014 Power and Energy Conference at Illinois (PECI)*, pp. 1–6, 2014.

- [52] "Use of Synchrophasor Measurements in Protective Relaying Applications," 2013.
- [53] B. K. Greene, "Novel Applications for Phasor Measurement Units and Synchrophasor Data," 2013.
- [54] M. Rihan, M. Ahmad, and M. S. Beg, "Vulnerability Analysis of Wide Area Measurement System in the Smart Grid," vol. 2013, no. September, pp. 1-7, 2013.
- [55] K. Koellner, "Generator black start validation using synchronized phasor measurement." *Prot. Relay Eng.....*, pp. 1-7, 2007.
- [56] M. Mills-Price and M. Scharf, "Solar generation control with time-synchronized phasors," ... , 2011 *64th Annu.*, pp. 1-8, 2011.
- [57] B. K. Johnson, S. Jadid, and S. E. Laboratories, "Validation of Transmission Line Relay Parameters Using Synchrophasors," pp. 1-8.
- [58] http://uk.sagepub.com/sites/default/files/upm-binaries/4913_Mentzer_Chapter_3_Time_Series_Forecasting_Techniques.pdf
- [59] <http://pages.cs.wisc.edu/~jerryzhu/cs731/stat.pdf>
- [60] <http://www.mit.edu/~6.s085/notes/lecture5.pdf>
- [61] <http://www.scribd.com/doc/232111635/Makridakis-Wheelwright-Hyndman-Forecasting-Methods-and-Applications-3rd-Ed#scribd>
- [62] A. D. Papalexopoulos and T. C. Hesterberg, "A regression-based approach to short-term system load forecasting," *IEEE Transactions on Power Systems*, vol. 5, pp. 1535-1547, 1990.
- [63] M. T. Hagan and S. M. Behr, "The Time Series Approach to Short Term Load Forecasting," *IEEE Transactions on Power Systems*, vol. 2, pp. 785-791, 1987.
- [64] H. S. Hippert, C. E. Pedreira, and R. C. Souza, "Neural networks for shortterm load forecasting: a review and evaluation," *IEEE Transactions on Power Systems*, vol. 16, pp. 44-55, 2001.
- [65] Sheikh, S.K., Unde M.G., "Short-Term Load Forecasting using ANN Technique," *International Journal of Engineering Sciences & Emerging Technologies (IJESET)*, vol. 1, issue 2, pp. 97-107, 2012.
- [66] D. K. Ranaweera, N. F. Hubele, and G. G. Karady, "Fuzzy logic for short term load forecasting," *International Journal of Electrical Power & Energy Systems*, vol. 18, pp. 215-222, 1996.

- [67] B.-J. Chen, M.-W. Chang, and C.-J. Lin, "Load forecasting using support vector Machines: a study on EUNITE competition 2001," *IEEE Transactions on Power Systems*, vol. 19, pp. 1821-1830, 2004.
- [68] A. Khotanzad, R. Afkhami-Rohani, L. Tsun-Liang, A. Abaye, M. Davis, and D. J. Maratukulam, "ANNSTLF-a neural-network-based electric load forecasting system," *IEEE Transactions on Neural Networks*, vol. 8, pp. 835-846, 1997.
- [69] A. Khotanzad, R. Afkhami-Rohani, and D. Maratukulam, "ANNSTLFArtificial Neural Network Short-Term Load Forecaster generation three," *IEEE Transactions on Power Systems*, vol. 13, pp. 1413-1422, 1998.
- [70] Jun M. Liu; R Chen; LM Liu; JL Harris, "A semi-parametric time series approach in modeling hourly electricity loads," *Journal of Forecasting*, vol. 25, issue 8, pp. 537-559, 2006.
- [71] Subbayya, S., Jetcheva, J. G., "Model selection criteria for short-term microgrid-scale electricity load forecasts," *IEEE PES Innovative Smart Grid Technologies Conference (ISGT)*, pp. 1-6, 2013.
- [72] Tao Hong, "Energy Forecasting: Past, Present and Future", *Foresight: The International Journal of Applied Forecasting*, issue 32, pp. 43-48, winter 2014.
- [73] <http://fileinfo.com/extension/d>
- [74] <http://www.mathworks.com/products/matlab/>
- [75] R. Cordeiro de Amorim and B. Mirkin, "Minkowski metric, feature weighting and anomalous cluster initializing in K-Means clustering," *Pattern Recognit.*, vol. 45, no. 3, pp. 1061–1075, Mar. 2012.
- [76] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," *Mach. Learn.*, vol. 42, no. 1–2, pp. 143–175, 2001.
- [77] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2002.
- [78] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy c-means clustering algorithm," *Comput. Geosci.*, vol. 10, no. 2–3, pp. 191–203, Jan. 1984.
- [79] Tao Hong, "Short Term Electric Load Forecasting". PhD dissertation, North Carolina State University, Sep 10th, 2010
- [80] H. L. Willis, *Power Distribution Planning Reference Book, Second Edition, Revised and Expanded*. New York: Marcel Dekker, 2002.

- [81] Cleveland, W.S. (1979) “Robust Locally Weighted Regression and Smoothing Scatterplots,” *Journal of the American Statistical Association*, vol. 74, pp. 829-836, 1979.
- [82] Breiman, Leo (2001), “Random Forests,” *Machine Learning*, vol. 1, pp. 5-32, 2001.
- [83] <https://citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics/>
- [84] <http://www.econ.ohio-state.edu/dejong/note2.pdf>
- [85] <http://www.dst.unive.it/~margherita/TSlectureNotes6.pdf>
- [86] Al-Zayer J, Al-Ibrahim AA., “Modeling the impact of temperature on electricity consumption in the Eastern Province of Saudi Arabia,” *Journal of Forecasting*, vol.15, pp. 97–106, 1996.
- [87] Engle RF, Granger CWJ, Rice J, Weiss A., “Semiparametric estimates of the relation between weather and electricity sales,” *Journal of the American Statistical Association* vol. 81, pp. 310–320, 1986.
- [88] Mendenhall W, Sincich T., “A Second Course in Statistics–Regression Analysis,” *Prentice-Hall: Englewood Cliffs, NJ*, 1996.
- [89] Gupta PC., “Adaptive short-term forecasting of hourly loads using weather information. In Comparative Models for Electrical Load Forecasting,” *Bunn DW, Farmer ED (eds)*. Wiley: New York; pp. 43–56, 1985.
- [90] Peirson J, Henley A., “Electricity load and temperature,” *Energy Economics*, vol. 16, pp. 235-243, 1994.
- [91] Ramanathan R, Engle R, Granger CWJ, Vahid-Araghi F, Brace C., “Short-run forecasts of electricity loads and peaks,” *International Journal of Forecasting*, vol. 13, pp. 161–174, 1997.
- [92] <http://www.pjm.com/markets-and-operations/ops-analysis/historical-load-data.aspx>
- [93] <https://www.ncdc.noaa.gov/data-access/land-based-station-data>
- [94] http://www.sheppardsoftware.com/usa_game/mid_atlantic.htm
- [95] <https://cran.r-project.org/web/packages/randomForest/randomForest.pdf>
- [96] http://www.ece.mtu.edu/~thavens/papers/ICPR_2008_Havens.pdf
- [97] <http://www.naruc.org/grants/Documents/Silverstein%20NCEP%20T-101%200420111.pdf>

- [98] Vallakati, Ranganath; Mukherjee, Anupam; Ranganathan, Prakash, “A density based clustering scheme for situational awareness in a smart-grid”, *IEEE International Conference on Electro/Information Technology (EIT)*, 2015.
- [99] <http://people.duke.edu/~rnau/compare.htm>
- [100] http://people.duke.edu/~rnau/Notes_on_nonseasonal_ARIMA_models--Robert_Nau.pdf
- [101] <https://onlinecourses.science.psu.edu/stat510/node/68>
- [102] <http://www.statisticshowto.com/what-is-a-standardized-residuals/>
- [103] R. Mahmud, R. Vallakati, A. Mukherjee, P. Ranganathan, and A. Nejadpak, “A Survey on Attacks on Smart Grid Metering Infra-Structures: Threats and Solutions”, in *IEEE EIT Conference, Illinois, May 23- 25 2015*.
- [104] R. Vallakati, A. Mukherjee, and P. Ranganathan (2015) “Situational Awareness using DBSCAN in Smart-Grid”, *Smart Grid and Renewable Energy, Vol.6 No.5 2015*
- [105] N. Gellerman, P. Ranganathan, R. Vallakati, and A. Mukherjee, “User interface for situational awareness of openPDC,” in *2014 North American Power Symposium (NAPS)*, pp. 1–6, 2014.