A Comparison of Cancer Classification Methods Based on Microarray Data

UNIVERSITY OF
KWAZULU - NATAL

INYUVESI
YAKWAZULU-NATALI

Mohanad Mohammed

April, 2018

A Comparison of Cancer Classification Methods Based on Microarray Data

by

Mohanad Mohammed

A thesis submitted to the
University of KwaZulu-Natal
in fulfilment of the requirements for the degree
of
MASTER OF SCIENCE
in
STATISTICS

Thesis Supervisor: Henry Mwambi

Thesis Co-supervisor: Bernard Omolo

UNIVERSITY OF KWAZULU-NATAL
INYUVESI YAKWAZULU-NATALI

UNIVERSITY OF KWAZULU-NATAL

SCHOOL OF MATHEMATICS, STATISTICS AND COMPUTER SCIENCE

PIETERMARITZBURG CAMPUS, SOUTH AFRICA

Declaration - Plagiarism

I, Mohanad Mohammed, declare that

1. The research reported in this thesis, except where otherwise indicated, is my original research.
2. This thesis has not been submitted for any degree or examination at any other university.
3. This thesis does not contain other persons' data, pictures, graphs or other information, unless specifically acknowlegded as being sourced from other persons.
4. This thesis does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then
   (a) their words have been re-written but the general information attributed to them has been referenced, or
   (b) where their exact words have been used, then their writing has been placed in italics and referenced.
5. This thesis does not contain text, graphics or tables copied and pasted from the internet, unless specifically acknowledged, and the source being detailed in the thesis and in the reference sections.

[illegible]	06/08/2018
Mohanad Mohammed (Student)	Date
[illegible]	6/8/2018
Henry Mwambi (Supervisor)	Date
[illegible]	06/08/2018
Bernard Omolo (Co-supervisor)	Date

Disclaimer

Abstract

Cancer is among the leading causes of death in both developed and developing countries. Through gene expression profiling of tumors, the accuracy of cancer classification has been enhanced, leading to correct diagnoses and the application of effective therapies. Here, we discuss a comparative review of the binary class predictive ability of seven classification methods (support vector machines, with the radial basis kernel (SVM(RK)), linear kernel (SVM(LK)) and the polynomial kernel (SVM(PK)), artificial neural networks (ANN), random forests (RF), *k*-nearest neighbor (KNN), and naive Bayes (NB)), using publicly-available gene expression data from cancer research. Results indicate that NB outperformed the other methods in terms of the accuracy, sensitivity, specificity, kappa coefficient, area under the curve (AUC), and balanced error rate (BER) of the binary classifier. Thus, overall the Naive Bayes (NB) approach turned out to be the best classifier with our datasets.

Dedication

This dissertation is dedicated to
my supervisors,
my parents,
my brothers,
my sisters,
teachers,
and friends
who have always taught me how to see the vitality in myself,
conquer fears and overcome challenges.
I hope that I have made you proud.

Acknowledgements

In the name of the Almighty God, most Gracious, most Merciful. Praise be to God, the Cherisher and Sustainer of the world.

I am highly grateful to the Almighty God for His unmeasurable Mercies that saw me through my studies. It is such a great honour for me and humbling at the same time because it has been a long journey characterized by long hours of hard work and sacrifice. It is with pleasure and gratitude to acknowledge those who rendered assistance through the lengthy process of my study. I sincerely appreciate all the big and small contributions from everybody who either directly or indirectly helped make my studies less stressful and a success.

I am specifically grateful to my supervisors, **Prof. Henry Mwambi** and **Prof. Bernard Omolo** for their presence, inspiration, patience, dedication and their powerful guidance and help. I again thank them for their availability and readiness to read and correct all the many errors in my work, as well as the financial support. Much thanks for their quick responses whenever I consulted. Their help in and outside the research has been immense. I am deeply grateful for their willingness to work with me. Thank you both for sharing your enormous statistical knowledge with me.

I would also like to thank **Miss Christel Bernard** for ensuring I have a comfortable working place environment and for her patience on my many questions.

Special thanks to all my colleagues and friends inside and outside South Africa, especially I would like thank **Dr. Murtada Khalafallah** for his moral and enthusiastic support, who encouraged me when I first started working on this project.

Many thanks go to **Ms Justine Nasejje** for her help, encouragement and assisting on some methods relevant to my research.

I would like to thank my parents, **Mohammed** and **Entisar** who have taught me

the important things in life and have supported all my endeavours, for their unconditional love, encouragement, patience for being away when they needed me the most.
I am grateful and thankful to my brothers and sisters, **Modather**, **Mogtba**, **Zahrra**, and **Tanzeel** for their caring of my parents. Many thanks to the rest of my extended family for your moral support.

Last, but not least, thanks to everyone who his or her names may not appear here, but I won't forget their help. May the Almighty bless them all.

I would like to express my profound gratitude to the School of Mathematics, Statistics, and Computer Science, College of Agriculture, Engineering, and Science, University of KwaZulu-Natal, South Africa and the Ministry of Higher Education & Scientific Research represented in Faculty of Mathematical and Computer Sciences, University of Gezira, Sudan for their support and financial help to complete this Masters project successfully.

Funding
This work was supported through the DELTAS Africa Initiative. The DELTAS Africa Initiative is an independent funding scheme of the African Academy of Sciences (AAS) Alliance for Accelerating Excellence in Science in Africa (AESA) and supported by the New Partnership for Africa's Development Planning and Coordinating Agency (NEPAD Agency) with funding from the Wellcome Trust [grant 107754/Z/15 /Z- DELTAS Africa Sub-Saharan Africa Consortium for Advanced Biostatistics (SSA CAB) programme] and the UK government. The views expressed in this publication are those of the author(s) and not necessarily those of AAS, NEPAD Agency, Wellcome Trust or the UK government.

Contents

Page

List of Figures **ix**

List of Tables **xiv**

Abbreviations **xvi**

Chapter 1: Introduction **1**

1.1 Background 1

1.2 Microarray Technology 3

1.2.1 Affymetrix Gene Expression Array Technology 3

1.3 Literature Review 4

1.4 Problem Statement 8

1.5 Objectives of the Study 9

1.6 The Structure of the Dissertation 9

Chapter 2: Data description **10**

2.1 Dataset 1: GSE5851 11

2.2 Dataset 2: GSE7670 11

2.3 Dataset 3: GSE8401 11

2.4 Dataset 4: GSE10072 11

2.5 Dataset 5: GSE10245 11

2.6 Dataset 6: GSE23988 12

2.7 Dataset 7: GSE25136 12

2.8 Dataset 8: GSE32962 12

2.9 Dataset 9: GSE35896 12

2.10 Dataset 10: GSE103091 13

Chapter 3: Methodology **14**

3.1 Support Vector Machines (SVM) 14
3.1.1 Kernel Function 16
3.2 Artificial Neural Networks (ANN) 17
3.3 Naive Bayes (NB) 19
3.4 Random Forests (RF) 20
3.5 k-Nearest Neighbor (KNN) 21
3.6 Performance Measures 22

Chapter 4: Application and Results **24**
4.1 Data Source 25
4.2 Data Preparation 25
4.2.1 Filtration 25
4.2.2 Normalization and Transformation 25
4.2.3 Datasets Partitioning 25
4.2.4 Features (Genes) Selection 26
4.3 Model Training 26
4.4 Model Validation 26
4.5 Baseline Model 27
4.6 Kruskal-Wallis (KW) Test 27
4.7 Results 28
4.7.1 Training Results of the Methods on the GSE8401 28
4.7.2 GSE8401: 34
4.7.3 GSE23988: 35
4.7.4 GSE7670: 36
4.7.5 GSE10072: 38
4.7.6 GSE10245: 39
4.7.7 GSE25136: 41
4.7.8 GSE35896: 42
4.7.9 GSE103091: 44
4.7.10 GSE5851: 45
4.7.11 GSE32962: 47

Chapter 5: Discussion and Conclusion **50**

References **58**

Appendix A **59**

Appendix B **66**

Appendix C **111**

List of Figures

Figure 1.1 Work flow of the Affymetrix microarray experiment. 4

Figure 3.1 (a) Original data in the input space. (b) Mapped data in the feature space.16

Figure 3.2 Multi-layer neural networks architecture. 19

Figure 3.3 k-nearest neighbor approach. 22

Figure 4.1 Flow diagram of the data processing stages until final model comparison and performance. 24

Figure 4.2 ROC curves for all the methods applied to the GSE8401 dataset. 34

Figure 4.3 Plot of 1 - BER against kappa for the GSE8401 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 35

Figure 4.4 ROC curves for all the methods applied to the GSE23988 dataset. 36

Figure 4.5 Plot of 1 - BER against kappa for the GSE23988 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 36

Figure 4.6 ROC curves for all the methods applied to the GSE7670 dataset. 37

Figure 4.7 Plot of 1 - BER against kappa for the GSE7670 dataset. The dashed lines represent the values of the baseline model. 38

Figure 4.8 ROC curves for all the methods applied to the GSE10072 dataset. 39

Figure 4.9 Plot of 1 - BER against kappa for the GSE10072 dataset. The dashed lines represent the values of the baseline model. 39

Figure 4.10 ROC curves for all the methods applied to the GSE10245 dataset. 40

Figure 4.11 Plot of 1 - BER against kappa for the GSE10245 dataset. The dashed lines represent the values of the baseline model. 41

Figure 4.12 ROC curves for all the methods applied to the GSE25136 dataset. 42
Figure 4.13 Plot of 1 - BER against kappa for the GSE25136 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 42
Figure 4.14 ROC curves for all the methods applied to the GSE35896 dataset. 43
Figure 4.15 Plot of 1 - BER against kappa for the GSE35896 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 44
Figure 4.16 ROC curves for all the methods applied to the GSE103091 dataset. 45
Figure 4.17 Plot of 1 - BER against kappa for the GSE103091 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 45
Figure 4.18 ROC curves for all the methods applied to the GSE5851 dataset. 46
Figure 4.19 Plot of 1 - BER against kappa for the GSE5851 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 47
Figure 4.20 ROC curves for all the methods applied to the GSE32962 dataset. 48
Figure 4.21 Plot of 1 - BER against kappa for the GSE32962 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right. 48

List of Tables

Table 2.1 Summary of the gene expression datasets used in the study. Here Pos and Neg represents positive and negative, respectivlely. GSE(GEO Sample), GPL(GEO Platform). 10

Table 3.1 Kernel Functions. 16
Table 3.2 Structure of the confusion matrix 22

Table 4.1 Tuning Parameter Results of KNN for GSE8401 29
Table 4.2 Confusion Matrix of KNN on GSE8401 Validation Set 29
Table 4.3 Tuning Parameter Results of NB for GSE8401 29
Table 4.4 Confusion Matrix of NB on GSE8401 Validation Set 29
Table 4.5 Tuning Parameter Results of RF for GSE8401 30
Table 4.6 Confusion Matrix of RF on GSE8401 Validation Set 30
Table 4.7 Tuning Parameter Results of SVM(RK) for GSE8401 30
Table 4.8 Confusion Matrix of SVM(RK) on GSE8401 Validation Set 31
Table 4.9 Tuning Parameter Results of SVM(LK) for GSE8401 31
Table 4.10 Confusion Matrix of SVM(LK) on GSE8401 Validation Set 31
Table 4.11 Tuning Parameter Results of SVM(PK) for GSE8401 32
Table 4.12 Confusion Matrix of SVM(PK) on GSE8401 Validation Set 33
Table 4.13 Tuning Parameter Results of ANN for GSE8401 33
Table 4.14 Confusion Matrix of ANN on GSE8401 Validation Set 33
Table 4.15 Summary accuracy measures for all the candidate methods using the GSE8401 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 34
Table 4.16 Summary accuracy measures for all the candidate methods using the GSE23988 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 35
Table 4.17 Summary accuracy measures for all the candidate methods using the GSE7670 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 37

Table 4.18 Summary accuracy measures for all the candidate methods using the GSE10072 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 38
Table 4.19 Summary accuracy measures for all the candidate methods using the GSE10245 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 40
Table 4.20 Summary accuracy measures for all the candidate methods using the GSE25136 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 41
Table 4.21 Summary accuracy measures for all the candidate methods using the GSE35896 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 43
Table 4.22 Summary accuracy measures for all the candidate methods using the GSE103091 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 44
Table 4.23 Summary accuracy measures for all the candidate methods using the GSE5851 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 46
Table 4.24 Summary accuracy measures for all the candidate methods using the GSE32962 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets. 47
Table 4.25 Average accuracies of the seven classification methods based on the ten validation datasets. 49

Table B.1 Tuning Parameter Results of KNN for GSE23988 66
Table B.2 Confusion Matrix of KNN on GSE23988 Validation Set 67
Table B.3 Tuning Parameter Results of NB for GSE23988 67
Table B.4 Confusion Matrix of NB on GSE23988 Validation Set 67
Table B.5 Tuning Parameter Results of RF for GSE23988 67
Table B.6 Confusion Matrix of RF on GSE23988 Validation Set 68
Table B.7 Tuning Parameter Results of SVM(RK) for GSE23988 68
Table B.8 Confusion Matrix of SVM(RK) on GSE23988 Validation Set 68
Table B.9 Tuning Parameter Results of SVM(LK) for GSE23988 68
Table B.10 Confusion Matrix of SVM(LK) on GSE23988 Validation Set 68
Table B.11 Tuning Parameter Results of SVM(PK) for GSE23988 69
Table B.12 Confusion Matrix of SVM(PK) on GSE23988 Validation Set 70
Table B.13 Tuning Parameter Results of ANN for GSE23988 70
Table B.14 Confusion Matrix of ANN on GSE23988 Validation Set 70
Table B.15 Tuning Parameter Results of KNN for GSE7670 71

Table B.16 Confusion Matrix of KNN on GSE7670 Validation Set 71
Table B.17 Tuning Parameter Results of NB for GSE7670 71
Table B.18 Confusion Matrix of NB on GSE7670 Validation Set 71
Table B.19 Tuning Parameter Results of RF for GSE7670 72
Table B.20 Confusion Matrix of RF on GSE7670 Validation Set 72
Table B.21 Tuning Parameter Results of SVM(RK) for GSE7670 72
Table B.22 Confusion Matrix of SVM(RK) on GSE7670 Validation Set 72
Table B.23 Tuning Parameter Results of SVM(LK) for GSE7670 73
Table B.24 Confusion Matrix of SVM(LK) on GSE7670 Validation Set 73
Table B.25 Tuning Parameter Results of SVM(PK) for GSE7670 74
Table B.26 Confusion Matrix of SVM(PK) on GSE7670 Validation Set 75
Table B.27 Tuning Parameter Results of ANN for GSE7670 75
Table B.28 Confusion Matrix of ANN on GSE7670 Validation Set 75
Table B.29 Tuning Parameter Results of KNN for GSE10072 76
Table B.30 Confusion Matrix of KNN on GSE10072 Validation Set 76
Table B.31 Tuning Parameter Results of NB for GSE10072 76
Table B.32 Confusion Matrix of NB on GSE10072 Validation Set 76
Table B.33 Tuning Parameter Results of RF for GSE10072 77
Table B.34 Confusion Matrix of on GSE10072 Validation Set 77
Table B.35 Tuning Parameter Results of SVM(RK) for GSE10072 77
Table B.36 Confusion Matrix SVM(RK) on GSE10072 Validation Set 77
Table B.37 Tuning Parameter Results of SVM(LK) for GSE10072 78
Table B.38 Confusion Matrix of SVM(LK) on GSE10072 Validation Set 78
Table B.39 Tuning Parameter Results of SVM(PK) for GSE10072 79
Table B.40 Confusion Matrix of SVM(PK) on GSE10072 Validation Set 80
Table B.41 Tuning Parameter Results of ANN for GSE10072 80
Table B.42 Confusion Matrix of ANN on GSE10072 Validation Set 80
Table B.43 Tuning Parameter Results of KNN for GSE10245 81
Table B.44 Confusion Matrix of KNN on GSE10245 Validation Set 81
Table B.45 Tuning Parameter Results of NB for GSE10245 81
Table B.46 Confusion Matrix of NB on GSE10245 Validation Set 81
Table B.47 Tuning Parameter Results of RF for GSE10245 82
Table B.48 Confusion Matrix of RF on GSE10245 Validation Set 82
Table B.49 Tuning Parameter Results of SVM(RK) for GSE10245 82
Table B.50 Confusion Matrix of SVM(RK) on GSE10245 Validation Set 82
Table B.51 Tuning Parameter Results of SVM(LK) for GSE10245 83
Table B.52 Confusion Matrix of SVM(LK) on GSE10245 Validation Set 83
Table B.53 Tuning Parameter Results of SVM(PK) for GSE10245 84
Table B.54 Confusion Matrix of SVM(PK) on GSE10245 Validation Set 85

Table B.55 Tuning Parameter Results of ANN for GSE10245 85
Table B.56 Confusion Matrix of ANN on GSE10245 Validation Set 85
Table B.57 Tuning Parameter Results of KNN for GSE25136 86
Table B.58 Confusion Matrix of KNN on GSE25136 Validation Set 86
Table B.59 Tuning Parameter Results of NB for GSE25136 86
Table B.60 Confusion Matrix of NB on GSE25136 Validation Set 86
Table B.61 Tuning Parameter Results of RF for GSE25136 87
Table B.62 Confusion Matrix of RF on GSE25136 Validation Set 87
Table B.63 Tuning Parameter Results of SVM(RK) for GSE25136 87
Table B.64 Confusion Matrix of SVM(RK) 87
Table B.65 Tuning Parameter Results of SVM(LK) for GSE25136 88
Table B.66 Confusion Matrix of SVM(LK) on GSE25136 Validation Set 88
Table B.67 Tuning Parameter Results of SVM(PK) for GSE25136 89
Table B.68 Confusion Matrix of SVM(PK) on GSE25136 Validation Set 90
Table B.69 Tuning Parameter Results of ANN for GSE25136 90
Table B.70 Confusion Matrix of ANN on GSE25136 Validation Set 90
Table B.71 Tuning Parameter Results of KNN for GSE35896 91
Table B.72 Confusion Matrix of KNN on GSE35896 Validation Set 91
Table B.73 Tuning Parameter Results of NB for GSE35896 91
Table B.74 Confusion Matrix of NB on GSE35896 Validation Set 91
Table B.75 Tuning Parameter Results of RF for GSE35896 92
Table B.76 Confusion Matrix of RF on GSE35896 Validation Set 92
Table B.77 Tuning Parameter Results of SVM(RK) for GSE35896 92
Table B.78 Confusion Matrix of SVM(RK) on GSE35896 Validation Set 92
Table B.79 Tuning Parameter Results of SVM(LK) for GSE35896 93
Table B.80 Confusion Matrix of SVM(LK) on GSE35896 Validation Set 93
Table B.81 Tuning Parameter Results of SVM(PK) for GSE35896 94
Table B.82 Confusion Matrix of SVM(PK) on GSE35896 Validation Set 95
Table B.83 Tuning Parameter Results of ANN for GSE35896 95
Table B.84 Confusion Matrix of ANN on GSE35896 Validation Set 95
Table B.85 Tuning Parameter Results of KNN for GSE103091 96
Table B.86 Confusion Matrix of KNN on GSE103091 Validation Set 96
Table B.87 Tuning Parameter Results of NB for GSE103091 96
Table B.88 Confusion Matrix of NB on GSE103091 Validation Set 96
Table B.89 Tuning Parameter Results of RF for GSE103091 97
Table B.90 Confusion Matrix of RF on GSE103091 Validation Set 97
Table B.91 Tuning Parameter Results of SVM(RK) for GSE103091 97
Table B.92 Confusion Matrix of SVM(RK) on GSE103091 Validation Set 97
Table B.93 Tuning Parameter Results of SVM(LK) for GSE103091 98

Table B.94 Confusion Matrix of SVM(LK) on GSE103091 Validation Set 98
Table B.95 Tuning Parameter Results of SVM(PK) for GSE103091 99
Table B.96 Confusion Matrix of SVM(PK) on GSE103091 Validation Set 100
Table B.97 Tuning Parameter Results of ANN for GSE103091 100
Table B.98 Confusion Matrix of ANN on GSE103091 Validation Set 100
Table B.99 Tuning Parameter Results of KNN for GSE5851 101
Table B.100 Confusion Matrix of KNN on GSE5851 Validation Set 101
Table B.101 Tuning Parameter Results of NB for GSE5851 101
Table B.102 Confusion Matrix of NB on GSE5851 Validation Set 101
Table B.103 Tuning Parameter Results of RF for GSE5851 102
Table B.104 Confusion Matrix of RF on GSE5851 Validation Set 102
Table B.105 Tuning Parameter Results of SVM(RK) for GSE5851 102
Table B.106 Confusion Matrix of SVM(RK) on GSE5851 Validation Set 102
Table B.107 Tuning Parameter Results of SVM(LK) for GSE5851 103
Table B.108 Confusion Matrix of SVM(LK) on GSE5851 Validation Set 103
Table B.109 Tuning Parameter Results of SVM(PK) for GSE5851 104
Table B.110 Confusion Matrix of SVM(PK) on GSE5851 Validation Set 105
Table B.111 Tuning Parameter Results of ANN for GSE5851 105
Table B.112 Confusion Matrix of ANN on GSE5851 Validation Set 105
Table B.113 Tuning Parameter Results of KNN for GSE32962 106
Table B.114 Confusion Matrix of KNN on GSE32962 Validation Set 106
Table B.115 Tuning Parameter Results of NB for GSE32962 106
Table B.116 Confusion Matrix of NB on GSE32962 Validation Set 106
Table B.117 Tuning Parameter Results of RF for GSE32962 107
Table B.118 Confusion Matrix of RF on GSE32962 Validation Set 107
Table B.119 Tuning Parameter Results of SVM(RK) for GSE32962 107
Table B.120 Confusion Matrix of SVM(RK) on GSE32962 Validation Set 107
Table B.121 Tuning Parameter Results of SVM(LK) for GSE32962 108
Table B.122 Confusion Matrix of SVM(LK) on GSE32962 Validation Set 108
Table B.123 Tuning Parameter Results of SVM(PK) for GSE32962 109
Table B.124 Confusion Matrix of SVM(PK) on GSE32962 Validation Set 110
Table B.125 Tuning Parameter Results of ANN for GSE32962 110
Table B.126 Confusion Matrix of ANN on GSE32962 Validation Set 110

Abbreviations

SVM	Support Vector Machines
ANN	Artificial Neural Networks
SVM(RK)	Support Vector Machines (Radial basis Kernel)
SVM(LK)	Support Vector Machines (Linear Kernel)
SVM(PK)	Support Vector Machines (Polynomial Kernel)
RF	Random Forests
KNN	k-Nearest Neighbor
NB	Naive Bayes
AUC	Area Under the Curve
BER	Balanced Error Rate
ROC	Receiver Operating Curve
DNA	Deoxyribonucleic Acid
RNA	Ribonucleic Acid
LDA	Linear Discriminant Analysis
DT	Decision Trees
MLP	Multi-Layer Perceptrons
DLBCL	Diffuse Large B-cell Lymphoma
GCB-like	Germinal Centre B-like DLBCL
AB-like	Activated B-like DLBCL
QDA	Quadratic Discriminant Analysis
KPCA	Kernel Principal Component Analysis
LR	Logistic Regression
NC	nearest centroid
ALL	Acute Lymphoblastic Leukemia
AML	Acute Myleloid Leukemia
GEO	Gene Expression Omnibus
GPL	GEO Platform accession number
GSE	GEO Sample accession number
Prim	primary
Met	metastatic
NSCLC	Non-Small Cell Lung Cancer
AC	Adenocarcinoma
SCC	Squamous Cell Carcinoma
Rec	Recurrence
NonRec	Non-Recurence
MetFree	metastatis-free
Sens	sensitive
Resist	resistant
TP	True Positive
FP	False Positive
TN	True Negative
FN	False Negative
KW	Kruskal-Wallis
ER	Estrogen Receptor

Chapter 1

Introduction

1.1 Background

Classification plays an essential role in understanding diseases. It is a form of data analysis that uses statistical and mathematical methods or models to classify observations or samples into different distinct categories. To classify the data, two steps are followed; the learning step and the classification step. The learning step is where the classification model is constructed based on training data. The classification step is where the constructed model is used to predict classes for a given data. Predictive modeling is a statistical method used to build predictive models to separate and classify new data points. There are two types of learning approaches, namely, the supervised and unsupervised learning. Supervised learning is where the class labels of each training observation or sample is provided. In contrast, unsupervised learning (or clustering) is where the class label of each training data is unknown, and the number of classes to be learned may not be known in advance (Han et al., 2011). Disease classification is very important for early detection, treatment, containment and other etiological and intervention applications. This ultimately helps to improve health and reduce negative outcomes such as death.

Recent global public health research shows an epidemiological transition from infectious to non-communicable diseases, the latter including different types of cancers. The incidence and prevalence of cancer is on the increase worldwide, both in the developing and developed countries (Olsen, 2015; Morhason-Bello et al., 2013).

Cancer is a disease in which cells in particular tissues in the body undergo uncontrolled division. This situation results in a malignant growth or tumor. Cancerous cells very often invade and destroy surrounding healthy tissues and organs. When there is no intervention, the body cells continue to divide and spread into adjacent tissues. The tumor can occur at any part of the human cells. Normally, the human cells grow and separate to make new cells as needed by the body to replace the old cells. When cancer occurs, this process does not go as it is supposed to be, rather the cells abnormally split without constraint and form outgrowths in the body called *tumors*. The tumors can spread or attack the surrounding tissues, which in this case, these are called *malignant tumors*. Furthermore, some growth cells can move to a distant part of the body, either through blood or lymph nodes and produce new tumors far from the original tumor location. This process is called *metastasis* in oncology research. There is another type of tumor growth called *benign* tumors which are not like the malignant tumors. They do not spread or attack the surrounding tissues. They are slow in growth and non-metastatic which when it is removed either by surgery or other treatment do not grow again. In contrast, the malignant tumors are fast in growing than metastatic ones, which sometimes recur after removal (Ganti, 2015; WHO, 2002).

There are many risk factors for cancer such as tobacco, alcohol, overweight and obesity, etc. In the United States alone, cancer has been identified as the second leading cause of death. The American Cancer Society, in a report published in 2017, indicated that more than 15.5 million Americans with a history of cancer existed by January 1, 2016 (ACS, 2017). Moreover, about 1.7 million new cases were expected, and close to 0.7 million expected to die of cancer in 2017 (Siegel, 2017).

Cancer remains the leading killer in the developed world and is emerging to be the second or third leading cause of mortality (after malaria) in developing countries, including Sub-Saharan Africa (Jemal et al., 2011; Moten et al., 2014), where HIV has also had a devastating effect. A high proportion of cancers that are relatively curable in developed countries are detected only at advanced stages in developing countries, due to late or inaccurate diagnoses (WHO, 2002). This motivates the evaluation of methods for the classification of different cancer-types and stages of the disease, in order to improve early detection and the design of targeted treatment strategies that may reduce mortality.

Microarray data have had a profound impact in disease diagnoses and prognoses, through accurate disease classification. This has helped clinicians to choose the appropriate treatment plans for patients (Abusamra, 2013). However, using gene expression data for cancer classification has been challenging because the data type is different in structure from other commonly used structures. Microarray data consists of small sample sizes, where each sample has a large number of the genes. As a way to mitigate this problem, it has been suggested to first perform filtration and gene selection through methods such as the two-sample t-test at a given stringent significance threshold such as 0.001 (Haury et al., 2011). This procedure ensures that only informative and sufficiently differentially expressed genes between the outcome classes are used in building the classifiers.

Microarrays provide a unique view into the biology of DNA. Previously, biologists worked hard to produce sufficient amounts of biological data for a given research problem (Seidel, 2008). Recently, DNA microarrays have provided a powerful tool to study thousands of genes simultaneously, leading to the production of massive amounts of data that have made research in microarrays so attractive. However it is important to note that there are many types of microarrays depending on the type of specimen placed on the microscope slides (e.g DNA, RNA, protein, and tissue). DNA microarrays are commonly utilized, and used to determine the expression levels and sequence of genes in a sample (Tuimala & Laine, 2003). Generally, microarrays experiments can be prepared from a variety of sources such as human, mouse, rat, and yeast.

Different methods for cancer classification using gene expression data have been proposed, including SVM, ANN, RF, linear discriminant analysis (LDA), and Bayesian network analysis (Dudoit et al., 2002; Chu & Wang, 2003; Hu et al., 2006; Musa, 2014; Vanitha et al., 2015; Mahmoud et al., 2014; Stephens & Diesing, 2014; Khan et al., 2001). These classification methods have been applied and their predictive performance compared in many studies (Furey et al., 2000; Hu et al., 2006; Abusamra, 2013; Vanitha et al., 2015).

1.2 Microarray Technology

There are several microarray technologies available in the market (e.g. the spotted two-color arrays, oligonucleotide (single-channel) arrays, etc.). The most commonly used technology is the single-color microarray (Affymetrix), hence its choice in this thesis. At the most general level, a microarray is a flat surface (slide) on which one molecule interacts with another. What differentiates the dual-channel from the single-channel microarray is the way probes are placed on the slide.

1.2.1 Affymetrix Gene Expression Array Technology

The Affymetrix system hybridizes only one sample per chip. This requires more slides per experiment and does not enjoy the advantage of using competitive hybridization. However, it simplifies experimental design and is based on a much more sensitive technology.
Affymetrix arrays, also commonly known as GeneChips, are microscopic slides that contain an ordered series of samples (DNA, RNA, protein, or tissue). The experiment starts with constructing the microarray, where single extracted RNA sample are fixed to a glass slide at known positions in the array. RNA is obtained from the cells, under different experimental conditions. Consequentially, these samples undergo labeling and hybridization, and the comparisons are then made computation-

ally, and then purification of the labeled products. Then thereafter the Affymetrix are hybridized with labeled sample. This is then eventually followed by scanning the sample to measures the ratio of each sample using laser scanners.
In the Affymetrix arrays each gene is represented as a probe set of 10 to 25 oligonucleotide pairs instead of one full length or partial cDNA clone. The oligonucleotide pair (probe pair) comprises of one oligonucleotide perfectly matching to the gene sequence (Perfect Match, PM) and a second oligonucleotide having one nucleotide mismatch in the middle of it (Mismatch, MM). Probes are designed within 500 base pairs of the 3′ end of each gene to hybridize uniquely in the same, predetermined hybridization conditions (Dalma-Weiszhausz et al., 2006; Brown et al., 1999; Tuimala & Laine, 2003; Robinson & Speed, 2007). See Figure 1.1.

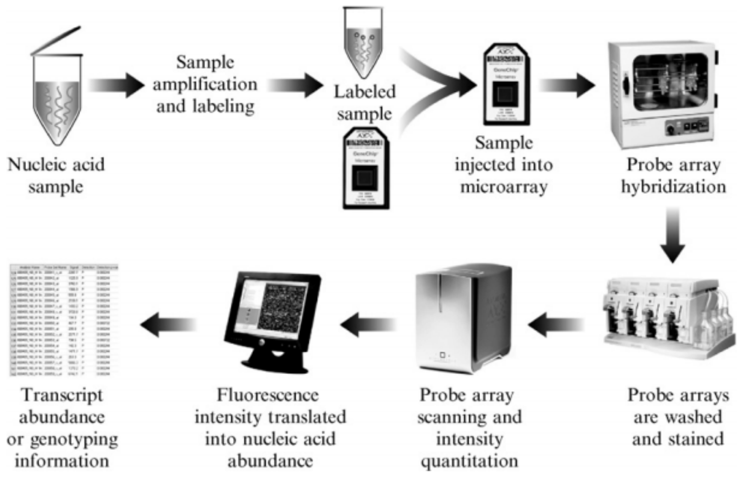
Nucleic acid sample
Sample amplification and labeling
Labeled sample
Sample injected into microarray
Probe array hybridization
Probe arrays are washed and stained
Probe array scanning and intensity quantitation
Fluorescence intensity translated into nucleic acid abundance
Transcript abundance or genotyping information

Figure 1.1 – Work flow of the Affymetrix microarray experiment.

Generally, the data is from m series of experiments (samples) and n genes (gene expression matrix). Thus the gene expression matrix is represented as follows

$$G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1m} \\ g_{21} & g_{22} & \cdots & g_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n1} & g_{n2} & \cdots & g_{nm} \end{bmatrix}$$

1.3 Literature Review

Recent literature has reported an increased spread of non-communicable diseases such as cancer (of all types). This has made cancer classification in early stages necessary to enhance disease diagnoses and prognoses. Such improvements in classification will help physicians to choose the suitable treatment (Abusamra, 2013) in order to avoid negative outcomes such as death. The last decade has seen a growing trend towards non-communicable disease classification using microarray gene expression data.

Over the past decades, biologists had to work hard to produce a small amount of data for use to explore a hypothesis with one observation at a time (Seidel, 2008). When microarray gene expression was invented, one experiment can now generate thousands of observations. Hence, this has led to decrease in the amount of time needed to generate data.

Since the discovery of microarray technology, cancer classification research has rapidly gained attention. During the past few years contribution have been made in the literature regarding the classification of various type of cancers or cancer subtypes. Also, various classification methods have been applied to cancer. These methods include support vector machines (SVM), artificial neural networks (ANN), K-nearest neighbor (KNN), Naive Bayes (NB), and decision trees (DT). In addition ensemble methods such as Bootstrap aggregating (bagging) and Boosting are also among methods of interest.

Valentini (2002) applied the SVM with linear, polynomial, and radial basis kernel functions and multi-layer perceptrons (MLP) with one hidden layer on a lymphoma, using gene expression data named Lymphochip (DNA microarray developed at stanford university school of medicine). The data consisted of 96 tissue samples from normal and malignant population of human lymphocytes, and 4026 different genes expressed in lymphoid cells with known roles in processes importance in immunology or cancer. The samples contained missing gene expression levels of about 6% of all the data whose values were replaced with zeros. They considered two problems, which are the classification of cancerous and non cancerous lymphoid tissues, and the identification of Diffuse Large B-cell Lymphoma (DLBCL). The second problem above includes the Germinal Centre B-like DLBCL (GCB-like), and Activated B-like DLBCL (AB-like). The analysis showed that the SVM with a linear kernel had the best performance. For classifying malignant and normal tissues the author estimated the generalization error using 10 fold cross validation and found that SVM with linear kernel achieved the best results with 1.04 error and 100% sensitivity. In identifying DLBCL subgroups, they performed 5 classification tasks using SVM and leave one out cross validation to estimate the generalization. For each classification task performed using different expression signatures (proliferation, T cell, lymphnode, and genes that distinguish germinal centre B-cells from other stages in B-cell ontogeny (GCB expression signature)) and all signatures together, the results showed that SVM with RBF had good results on GCB expression signatures with 4% of an estimated generalization error and 91% sensitivity. That means GCB expression signatures are specifically related to the separation of GCB-like and AB-like subgroup of lymphoma inside the DLBCL group.

Wu et al. (2003) compared the performance of SVM, RF, KNN, LDA, quadratic discriminant analysis (QDA), and bagging and boosting classification trees, using an ovarian cancer mass spectrometry data. The dataset was obtained from the National Ovarian Cancer Early Detection Program at Northwestern University Hospital. This data set consists of MS spectra that extend from 800 to 3500 Da that were obtained on serum samples from 47 patients with ovarian cancer and 44 normal patients. Preprocessing were done using approaches such as taking the log of the intensities, background subtraction, and peak identification, etc. Thereafter, the feature selection were done using two methods to select subset features. First they ranked the features based on the t-statistic, and used the RF to select the subset feature based on the a variable importance measure. Then they applied the classifiers on the data with 15 and 25 selected markers (features) and compared their performance based on the prediction error rate using 10-fold cross validation and 0.632+ bootstrap methods. The results showed that the 0.632+ rule provides more stable estimates of the error rate than 10 fold cross validation for some methods. Comparatively the results showed that RF and LDA performed well among other approaches. Using 15 markers selected using the t-statistic, SVM achieved the lowest error rate followed by the LDA method and the RF approach was among the top three. When the number of selected markers increases from 15 to 25, SVM had the lowest error rate and RF followed closely in performance. While RF outperformed all the methods when the selected variable are derived from the importance measures based on RF. In general RF had lower prediction error rate than the minimum error rate obtained using variables selected through t-statistics.

Liu et al. (2005) proposed a novel analysis procedure, which involved reducing the dimension using kernel principal component analysis (KPCA) and classification using logistic regression (LR) for discrimination. Gene (features) selection was done based on the likelihood ratio, to obtain the most informative genes based on the highest likelihood ratio score. The proposed method was applied to five gene expression datasets involving human tumor samples such as leukemia, colon, lung cancer, lymphoma, and NCI. The procedure was then compared with SVM and ANN. Thereafter, the classification performance were assessed using the leave one out cross validation for all the datasets except for the leukemia data set which was based on one training and test data only. They showed that the new procedure was able to distinguish between different classes with high accuracy.

Delen et al. (2005) compared three methods, namely, ANN, decision trees (DT), and LR prediction models using a large dataset from breast cancer. The authors used data contained in the SEER cancer Incidence Public-Use Database for the years 1973 and 2000. This dataset contained of $433,272$ records/cases and 72 variables which provided social demographic and cancer specific information. In an exploratory analysis of the data it was found 40% of the records contained missing data. In the subsequent analysis the missing data were dealt with by removing the records with missing information leading to the so called complete case analysis. Furthermore, they examined the impact of removing these records on other variables and the analysis showed there was no significant effect on the distribution of the variables. After the preliminary data cleaning and preparation for analysis step, the final analysis dataset consisted of 16 predictor variables with one dependent variable in a total of $202,932$ records. Their results indicated that the DT model was the best predictive model with 93.6% accuracy on the holdout sample, while the ANN model had an accuracy of 91.2% and the LR model an accuracy of 89.2%.

Hu et al. (2006) conducted a comparison of five classification methods (LibSVMs, C4.5, BaggingC4.5, AdaBoostingC4.5, and RF) on seven distinct microarray cancer datasets, namely breast, lung, lymphoma, ALL-AML leukemia, colon, ovarian, and prostate. Preprocessing of the microarray data was done via information gain ratio for gene selection and used (Fayyad and Irani's MDL) attribute discretization methods. Ten fold cross validation was used for performance estimation. Two statistical methods, namely, the Wilcoxon signed rank test and sign test were used to validate the average accuracies of ten-fold cross validation on all the data sets. They showed that all the ensemble methods (BaggingC4.5, AdaBoostingC4.5, and RF) performed better than single-classification methods (LibSVMs, C4.5) and that the Wilcoxon signed rank test was better than the sign test.

In order to determine the best feature selection method, Haury et al. (2011) compared 32 feature selections methods using five classification methods, namely, the nearest centroid (NC), KNN with $k = 9$, linear SVM with $C = 1$, linear discriminant analysis (LDA), and NB, to evaluate their performance. They showed that the simple t-test feature selection method provided the best results out of 32 feature selection methods among the classification methods used.

A more general comparative study on classification methods was performed by Abusamra (2013), who applied SVM, KNN, and RF, and eight different feature selection methods, namely, information gain, twoing rule, sum minority, max minority, Gini's index, sum of variances, t-statistics, and one-dimension SVM, on two publicly-available glioma datasets. Five-fold cross-validation was used to evaluate the classification performance. In terms of accuracy, the SVM approach was the best in comparison to all other models even before feature selection, due to its suitability for high dimensional data. These results suggest that by performing feature selection, the accuracy of classification can be significantly improved by using a smaller number of genes.

Dwivedi (2016) presented a classification of leukemia (the data set consisted of 46

samples in which there were 32 acute lymphoblastic leukemia (ALL) samples and 14 acute myleloid leukemia (AML) samples), using ANN but also compared the ANN approach to five other methods (SVM, NB, LR, KNN, and classification trees). This study applied the leave-one-out and ten-fold cross-validation methods to assess the performance of the methods using eight measures. ANN had a significant classification accuracy of 98% on ten-fold cross-validation and leave-one-out approach. While, SVM had the second best classification accuracy of 91%. Furthermore, ANN, KNN and NB simultaneously had the highest sensitivity. However ANN had both the highest sensitivity and specificity of 100% and 93% respectively.

Huang et al. (2018) concluded that SVM are a very powerful method in various fields, including cancer genomics, compared to the other methods. To date SVM have been applied in many application such as classification/ subtyping, biomarker/ signature discovery, drug discovery, driver gene discovery, and gene interaction. The success of SVM is partly related to the flexibility of any kernel approach and partly related to the robustness of SVM in the presence of bias in the training data.

A review of the literature suggests that SVM, KNN and RF are the three top most commonly used classification methods in microarray studies. In most of the previous comparative studies, the methods have been evaluated based on a single cancer-type data (Abusamra, 2013; Valentini, 2002; Wu et al., 2003; Dwivedi, 2016), without replication across the cancer spectrum, to ensure robustness. In studies that have employed different cancer-types, the comparisons have been between ensemble and single-classification methods (see Hu et al. (2006)). Generally, factors that influence performance of the classification methods such as microarray platform, disease under study and the gene selection method, have not been addressed (Novianti et al., 2014). Consequently, none of the methods have been unanimously agreed upon as the best method in microarray based cancer classification.

In this study, we performed a comparative review of the SVM (with the linear, polynomial and radial basis kernels), KNN, RF, NB and ANN, in an attempt to identify the best method for microarray-based cancer classification. The methods were evaluated based on classification accuracy, sensitivity, specificity, kappa coefficient, AUC, receiver operating curve (ROC), and BER (Stephens & Diesing, 2014), using ten publicly-available microarray datasets from the same platform.

1.4 Problem Statement

Cancer tumor classification based on morphological characteristics alone has been shown to have serious limitations in some studies (Golub et al., 1999). The use of gene expression data from microarrays has improved the classification. However, numerous classification algorithms have been developed in this regard but none has been unanimously accepted as the best method for microarray data. Here, we seek to review these methods with the goal of identifying a superior method, based on publicly-available microarray data.

1.5 Objectives of the Study

The general objective of the thesis is to review cancer classification techniques, and apply these methods on publicly available microarray gene expression data. The specific objectives of this study are:

- To obtain and analyze microarray data from a public repository (GEO).
- To perform gene selection, i.e. obtain a small panel of genes to employ as input variables/predictors in the classification algorithms.
- To identify a classification method that can be regarded as the best method for cancer classification, based on certain statistical measures of performance.

1.6 The Structure of the Dissertation

This study is concerned with classification methods for disease applied to cancer datasets. There are five chapters in the dissertation, which are structured as follows:

Chapter 1: This chapter introduces the study, by giving the background, a literature survey, and the aims and objectives of the study.

Chapter 2: This chapter presents the background information on the data in general, a description of the each dataset used in the study, and a summary table for the datasets.

Chapter 3: In this chapter, we discuss the methods support vector machines, artificial neural networks, naive Bayes, random forests, and k-nearest neighbor models as used in cancer classification. The write up explains the methodology and steps that will be used to compute the classification performance of each method.

Chapter 4: Here, we present results for each methods on the ten datasets. The results compares the performance of the different methods based on seven different measures of performance namely accuracy, sensitivity, specificity, kappa coefficient, ROC, area under the curve (AUC), and balanced error rate (BER).

Chapter 5: This chapter gives the discussion, conclusion and future research areas emanating from the current research.

Chapter 2

Data description

In this study, we used ten gene expression datasets on the most common cancer-types among men and women, which were downloaded from Gene Expression Omnibus (https://www.ncbi.nlm.nih.gov/geo/), with accession numbers $GSE23988$, $GSE7670$, $GSE8401$, $GSE10072$, $GSE10245$, $GSE25136$, $GSE35896$, $GSE103091$, $GSE5851$ and $GSE32962$. The description for each dataset is as given below. Clinical information would have been useful. But not all datasets in GEO contain clinical information for the samples, which is a limitation in clinical studies. Table 2.1 below provides a concise summary of the gene expression datasets used in the study.

Table 2.1: Summary of the gene expression datasets used in the study. Here Pos and Neg represents positive and negative, respectivlely. GSE(GEO Sample), GPL(GEO Platform).

Cancer type	**Accession number**	**Platform**	**Class attribute**	**Pos(Neg) class**	**Pos(Neg) class size**	**Sample type**
Colorectal	GSE5851	GPL571	Response status	No(Yes)	43 (25)	Independent
Lung	GSE7670	GPL96	Tissue type	Tumor(Normal)	27 (27)	Paired
Melanoma	GSE8401	GPL96	Tumor type	Prim(Met)	31 (52)	Independent
Lung	GSE10072	GPL96	Tissue type	Tumor(Normal)	33 (33)	Paired
Lung	GSE10245	GPL570	Tumor type	SCC(AC)	18 (40)	Independent
Breast	GSE23988	GPL96	Estrogen receptor (ER)	ERpos(ERneg)	32 (29)	Independent
Prostate	GSE25136	GPL96	Recurrence status	Rec(NonRec)	39 (40)	Independent
Leukemia	GSE32962	GPL570	Prednisolone Sensitivity	Sens(Resist)	19 (24)	Independent
Colorectal	GSE35896	GPL570	Mutation status	Yes(No)	29 (33)	Independent
Breast	GSE103091	GPL570	Metastasis status	Met(MetFree)	31 (76)	Independent

2.1 Dataset 1: GSE5851

The dataset contains 22,277 probes from 68 metastatic colorectal cancer tumors, subjected to cetuximab monotherapy. Samples were classified by response status as either a non-responder (No) with 43 or a responder (Yes) with 25 cases (Khambata-Ford et al., 2007). After filtration and normalization, 12,084 genes were retained. Fourteen (14) differentially expressed genes between the two classes were employed in model training and validation downstream.

2.2 Dataset 2: GSE7670

The dataset consists of 22,283 probes from 66 lung adenocarcinoma tissues, 27 of which were matched normal-tumor samples (Su et al., 2007). Filtration and probe reduction to one per gene resulted in 12,084 genes. Analysis of differentially expressed genes 19 paired training samples yielded 1,450 genes for classification. Here, we used the tumor status (Tumor/Normal) as classification variable for model training and validation.

2.3 Dataset 3: GSE8401

The dataset consists of 22,283 probes from 83 samples. There are 31 primary (Prim) and 52 metastatic (Met) melanoma tumors from patients undergoing surgery, collected from 1992 to 2001 as a part of the diagnostic work-up or therapeutic strategy (Xu et al., 2008). Probe filtration and reduction to one per gene yielded 12,084 genes, of which 1,483 were differentially expressed between the 59 training primary and metastatic samples. Model training and validation were based on 1,483 genes.

2.4 Dataset 4: GSE10072

There was a total of 22,283 probes from each of the 33 paired (tumor and normal) samples used in this analysis (Landi et al., 2008). Filtration reduced the initial 22,283 probes to 12,084 genes, of which 3002 were differentially expressed genes hence were selected for model training and validation.

2.5 Dataset 5: GSE10245

The dataset contains 54,675 probes from 58 non-small cell lung cancer (NSCLC) tumor samples, classified as either adenocarcinoma (AC) with 40 or squamous cell carcinoma (SCC) with 18 cases (Kuner et al., 2009). Filtration reduced the probes to

18, 978 genes, and differential expression analysis selected 819 genes from 41 training samples, for model training and validation.

2.6 Dataset 6: GSE23988

The dataset consists of 22, 283 probes taken from 61 patients with HER2-normal stage I-III breast cancer who received preoperative chemotherapy to identify gene sets associated with pathological complete response to therapy (Iwamoto et al., 2010). Estrogen receptor (ER) status (32 ERpos) and (29 ERneg) was used as the class attribute. After filtration and normalization, 12, 084 genes were retained, out of which 579 were selected for model building and validation.

2.7 Dataset 7: GSE25136

The dataset consists of 22, 283 probes from 79 prostate cancer tumors, classified as either having disease recurrence (Rec) with 39 or non-recurence (NonRec) with 40 cases. Recurrent status was used as the positive class in the model building and validation processes (Sun & Goodison, 2009). Filtration reduced the probes to 12, 084 genes, and differential expression analysis selected 52 genes from 56 training samples, for model training and validation.

2.8 Dataset 8: GSE32962

The dataset contains 54, 517 probes from blood samples from 43 infants with Acute Lymphoblastic Leukemia (ALL), subjected to prednisolone treatment. The samples were classified by prednisolone sensitivity as either sensitive (Sens) with 19 or resistant (Resist) with 24 cases (Spijkers-Hagelstein et al., 2012). Filtration reduced the probes to 18, 956 genes, of which 117 genes were differentially expressed and used for model training and validation.

2.9 Dataset 9: GSE35896

The dataset contains 54, 675 probes from 62 colorectal cancer tumors, classified by KRAS mutation status as either a KRAS-mutant (Yes) with 29 or KRAS wild-type (No) with 33 cases sub-type (Schlicker et al., 2012). Filtration reduced the probes to 18, 978 genes. Differential gene expression analysis yielded 43 genes, based on 45 training samples, for subsequent model training and validation.

2.10 Dataset 10: GSE103091

The dataset also contains 54, 675 probes from 107 primary breast tumours (Jézéquel et al., 2015). The tumors were classified by metastasis status as either metastatic (Met) with 31 or metastatis-free (MetFree) with 76 cases, with metastasis-free as the positive class attribute. Probe filtration and reduction to one per gene yielded 18, 978 genes. Differential gene expression analysis resulted in a 54 gene list, which was employed in model training and validation.

Chapter 3

Methodology

Several methods have been developed for classification using microarray gene expression data, each with its advantages and disadvantages. In this study, we present a comparative analysis of seven classification methods using ten cancer datasets described in Chapter 2. Below we present an overview of each method.

3.1 Support Vector Machines (SVM)

The SVM method was first presented by Boser et al. (1992) at the Computational Learning Theory (COLT92) ACM Conference in 1992. SVM are based on the idea of the plane that lies furthermost from both classes. This plane is known as the *optimal (maximum) margin hyperplane*. The hyperplane is completely determined by a subset of the samples known as the *support vectors* (Moguerza & Muñoz, 2006). SVM have the ability to handle problems where the data are not linearly separable by transforming the data using mapping kernel functions such as the radial basis function (RBF) kernel, polynomial function, and the linear function (Stephens & Diesing, 2014). Moreover, SVM have strong capability in handling high dimensional data, which is clearly an add advantage. Accordingly, this strength makes SVM widely appealing and have been successfully applied to real-life data analysis problems such as handwritten character recognition, human face recognition, radar target identification, speech identification, and gene expression data analysis (Brown et al., 1999; Chu & Wang, 2003).

Suppose we have m samples and n genes. Further, assume samples belong to two distinct outcome classes represented by $+1$ or -1 and a feature vector $\mathbf{g}_i$ such that $(\mathbf{g}_i, y_i) \in G \times Y \quad i = 1, 2, \ldots m$, where $\mathbf{g}_i = (g_{i1}, g_{i2}, \ldots, g_{in})'$ is the sample profile (vector) and $y_i \in \{+1, -1\}$ is the outcome class dichotomy. The goal is to classify the samples into the two classes by training the SVM to map the input data (using a suitable kernel function) onto a high-dimensional space (feature space)

$\{(\Phi(\mathbf{g}_i), y_i)\}_{i=1}^{m}$. This is achieved by constructing an optimal separating hyperplane that lies furthest from both classes (see Fig 3.1).

The general form of a separating hyperplane in the space of the mapped data is defined by

$$\mathbf{w}^T\Phi(\mathbf{g}) + b = 0. \tag{3.1}$$

Here, $\mathbf{w} = (w_1, w_2, \ldots, w_n)'$, is the weight vector. We can rescale the $\mathbf{w}$ and b such that the following equation determines the point in each class that is nearest to the hyperplane

$$|\mathbf{w}^T\Phi(\mathbf{g}) + b| = 1. \tag{3.2}$$

Therefore, it should follow that for each sample i, $i \in \{1, 2, \ldots, m\}$,

$$\mathbf{w}^T\Phi(\mathbf{g}_i) + b = \begin{cases} \geq 1 & \text{if} \quad y_i = +1 \\ \leq -1 & \text{if} \quad y_i = -1 \end{cases} \tag{3.3}$$

After the rescaling, the distance from the nearest point in each class to the hyperplane is $\frac{1}{\|\mathbf{w}\|}$. Thus, the distance between the two classes is $\frac{2}{\|\mathbf{w}\|}$, which is called the *margin*. To maximize the margin, the following optimization problem has to be solved

$$\min_{\mathbf{w},b} \quad \|\mathbf{w}\|^{\mathbf{2}} \tag{3.4}$$

subject to

$$y_i(\mathbf{w}^T\Phi(\mathbf{g}_i) + b) \geq 1, \quad i = 1, 2, \ldots, m. \tag{3.5}$$

The square in the norm of $\mathbf{w}$ is introduced to make the problem quadratic. Suppose $\mathbf{w}^*$ and b^* are the solutions to Eq (3.4). Then this solution determines the hyperplane in the feature space where $(\mathbf{w}^*)^T\Phi(\mathbf{g}) + b^* = 0$. The points $\Phi(\mathbf{g}_i)$ that satisfy the qualities $y_i((\mathbf{w}^*)^T\Phi(\mathbf{g}_i) + b^*) = 1$ are called *support vectors* as shown in Fig 3.1 (b) (Moguerza & Muñoz, 2006). There are many packages in *R* for SVM implementation (e.g. e1071 package), but we have chosen the *kernlab* package because it contains various kernelized learning algorithms and all the features that we need as well, Moreover, *kernlab* is customized for kernel methods in *R* (Karatzoglou et al., 2016).

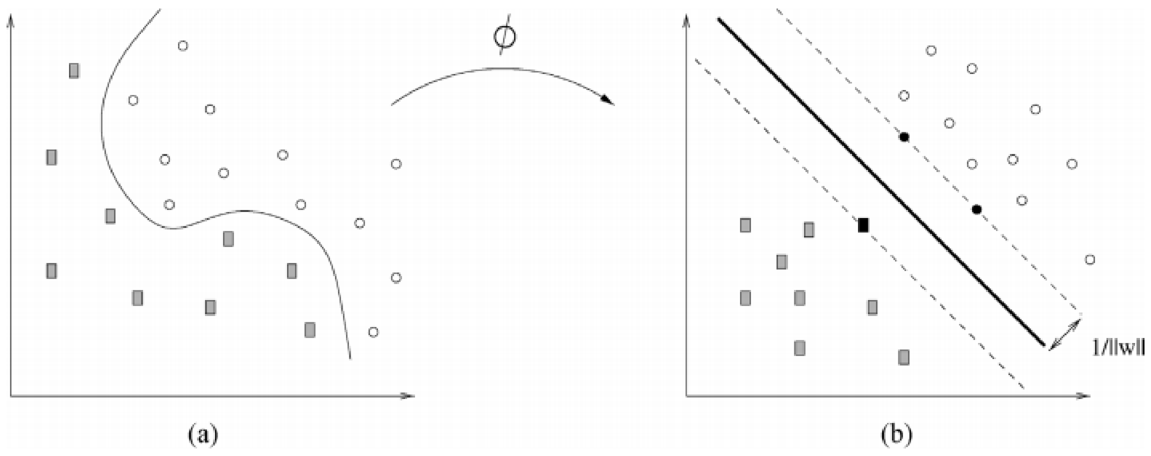

Figure 3.1 – (a) Original data in the input space. (b) Mapped data in the feature space.

3.1.1 Kernel Function

The real world problems are not usually linearly separable. Therefore, there is need for a function which maps the original space to some higher dimensional feature space where the training set is separable. Thus, we need to find a non-linear transformation function $\Phi(\mathbf{g})$, to achieve this task hence a class of functions called kernels is used.

A kernel $K(\mathbf{g}, \mathbf{y})$ is a real valued function $K : GXG \rightarrow \mathbb{R}$ for which there exist a function $\Phi : G \rightarrow Z$, where Z is a real vector space, with the property $K(\mathbf{g}, \mathbf{y}) = \Phi(\mathbf{g})^T \Phi(\mathbf{y})$.

The kernel $K(\mathbf{g}, \mathbf{y})$ acts as a dot product in the space of Z. In the literature G and Z are called input space and feature space, respectively. As well as, the $K(\mathbf{g}, \mathbf{y})$ must satisfy Mercer's condition, hence it is known as a Mercers kernel.

There are many kernel functions, and in Table 3.1 below we present the ones used in the current study.

Table 3.1: Kernel Functions.

Name	*Kernel Function*
$Linear$	$K(\mathbf{g}, \mathbf{y}) = \mathbf{g}^T \mathbf{y}$
$Polynomial$	$K(\mathbf{g}, \mathbf{y}) = (c + \mathbf{g}^T \mathbf{y})^d$
$RadialBasisFunction(RBF)$	$K(\mathbf{g}, \mathbf{y}) = \exp(-\gamma \|\|\mathbf{g} - \mathbf{y}\|\|^2)$

Where c is the constant value, d is the polynomial degree, and γ is a parameter that sets the spread of the kernel, it defines how far the influence of a single training example reaches. Intuitively, a small gamma value define a Gaussian function with a large variance.

3.2 Artificial Neural Networks (ANN)

Artificial neural networks (ANN) are multi-layered models that are constructed from three layers, each layer consisting of nodes called *neurons* (Dwivedi, 2016). The input layer contains nodes whose number is based on the input features. The output layer contains nodes equal to the number of classes, and finally the hidden layer contains nodes determined by the level of tuning required. The inputs are weighted by multiplying each input by a weight as a measure of its contribution. The layers are connected together via connection weights. These weights are determined through stages of model fitting. The hidden nodes receive the sum weighted from the input layer plus some bias. This summation is passed onto the transform function (activation function) to generate the results. These results are called *outputs* and interpreted as class probability in our case.

There are many types of architectures of ANN. The choice of the number of hidden layers is an important part of deciding the overall neural network architecture and can affect the results. Here, we used the default setting for the *nnet* package, which is the single hidden layer and is sufficient for our purposes. Neural networks are used widely in different fields such as prediction in time series models, economic modeling and medical applications among others (Stephens & Diesing, 2014). In addition, ANN can be applied to the classification problem using microarray gene expression data (Dwivedi, 2016).

Consider the simplest multi-layered network, with one hidden layer, as in Fig 3.2 below. Assume we have gene expression data where n denotes the number of genes. Then the input layer receives the n gene expression levels for a sample, each multiplied by the corresponding weight, $w_{ij}^{(1)} g_j$, as shown in Eq 3.6, below:

$$b_i = \sum_{j=0}^{n} w_{ij}^{(1)} g_j \quad i = 1, 2, \ldots, m, \tag{3.6}$$

where $\mathbf{g} = (g_0, g_1, g_2, \ldots, g_n)'$ is a vector of input features and $g_0 = 1$ is a constant input feature with weight w_{i0}. The quantities, b_i, are called *activations*, and the parameters $w_{ij}^{(1)}$ are the weights. Note that alternatively b_i can be viewed as a summary of the n genes from sample i. The superscript "(1)" indicates that this is the first layer of the network. Each of the activations is then transformed by a nonlinear activation function f, typically a sigmoid, as in Eq 3.7 below:

$$z_i = f(b_i) = \frac{1}{1 + \exp(-b_i)}. \tag{3.7}$$

The quantities z_i are interpreted as the output of hidden units, so called because they do not have values specified by the problem (as is the case for input units) or target values used in the training (as is the case for output units).

In the second layer, the outputs of the hidden units are linearly combined to give the activations of the K output units:

$$a_k = \sum_{i=0}^{m} w_{ik}^{(2)} z_i \quad k = 1, 2, \ldots, K. \tag{3.8}$$

Again, $z_0 = 1$ corresponds to the bias. The transformations in the second layer of the neural networks are parameterized by weights $w_{ik}^{(2)}$. The output units are transformed using an activation function. Again, a sigmoid function may be used as shown below:

$$y_k = f(a_k) = \frac{1}{1 + \exp(-a_k)}. \tag{3.9}$$

These equations may be combined to give the overall equation that describes the forward propagation through the network, and describes how an output vector is computed from an input vector, given the weight matrices as

$$y_k = f\left(\sum_{i=0}^{m} w_{ik}^{(2)} \left(f\left(\sum_{j=0}^{n} w_{ij}^{(1)} g_i\right)\right)\right). \tag{3.10}$$

ANN are implemented using the *R* package *nnet*, because it is the simplest one and restricted to a single layer (Ripley et al., 2016).

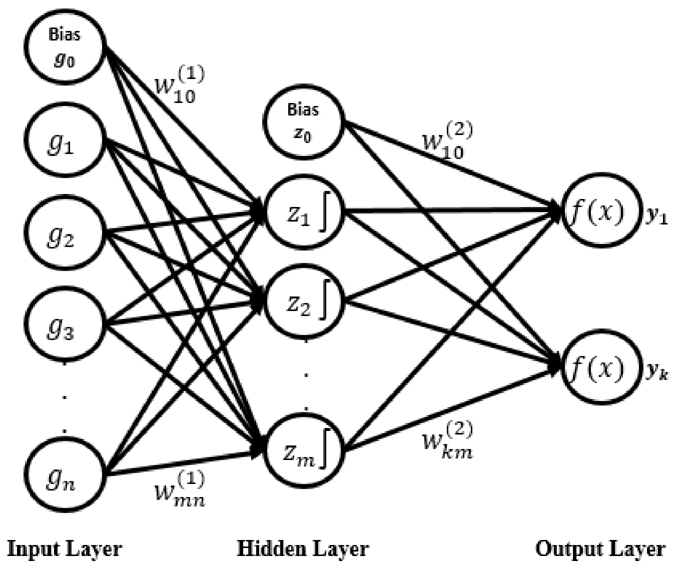

Figure 3.2 – Multi-layer neural networks architecture.

3.3 Naive Bayes (NB)

The Naive Bayes classifier uses probability theory to find the most likely of the possible classes in a classification problem. The NB classifier relies on two assumptions, namely, that each attribute is conditionally independent from the other attributes given the class and that all the attributes have influence on the class (De Campos et al., 2011). The popularity of this classifier is mainly due to its simplicity, yet exhibiting a surprisingly competitive predictive accuracy. The NB classifier has previously been applied in many fields, including microarray gene expression data (Stephens & Diesing, 2014; Dwivedi, 2016).

Consider an m by n gene expression data matrix, where m is the number of the samples and n is the number of the genes (features). Let $g_{kj}, j = 1, 2, \ldots, n$, denote the j-th gene on the k-th sample. Let C_i be the i-th class, $i = 1, 2, \ldots, L$. The Naive Bayes classifier uses the *maximum a posteriori* (MAP) classification rule to classify these samples. The probability of the k-th sample vector, $\mathbf{G}_k = (g_{k1}, g_{k2}, \cdots, g_{kn})'$, is calculated and then the sample is assigned the class with largest probability from L conditional probabilities. That is, let $P(C_1|\mathbf{G}_k), P(C_2|\mathbf{G}_k), \ldots, P(C_L|\mathbf{G}_k)$ denote the set of L conditional probabilities. The NB classification depends on the Bayes rule, which states that a posterior probability

$$P(C_i|\mathbf{G}_k) = \frac{P(\mathbf{G}_k|C_i)P(C_i)}{P(\mathbf{G}_k)} \propto P(\mathbf{G}_k|C_i)P(C_i), \ k = 1, 2, \ldots, m, \tag{3.11}$$

where $P(\mathbf{G}_k)$ is considered a common factor for all the L probabilities.

The NB classification assumes all input features are conditionally independent, that is,

$$\begin{aligned} P(g_{k1}, g_{k2}, \dots, g_{kn}|C_i) &= P(g_{k1}|g_{k2}, \dots, g_{kn}, C_i)P(g_{k2}, \dots, g_{kn}|C_i) \\ &= P(g_{k1}|C_i)P(g_{k2}, \dots, g_{kn}|C_i) \\ &= P(g_{k1}|C_i)P(g_{k2}|C_i)\dots P(g_{kn}|C_i) \end{aligned} \tag{3.12}$$

Ultimately, NB classifies a new sample, $\mathbf{G}^*$, according to the model with MAP probability given the sample, as

$$\text{Class}(\mathbf{G}^*)_{MAP} = \text{argmax}(P(C_i|\mathbf{G}^*)). \tag{3.13}$$

NB are implemented using the *R* package *naivebayes*.

3.4 Random Forests (RF)

Random forests were first introduced in 2001 (Friedman et al., 2001; Breiman, 2001). They are an extension of classification and regression trees, and also an improvement over bagged trees by the way of a random small tweak to de-correlate the trees. Growing random forests leads to an improvement in prediction accuracy compared to single or bagged trees (Qi, 2012). We build a number of forests of decision trees on bootstrapped training samples from the original data. A tree is obtained by recursively splitting the genes set of size p. At each node of the tree, a candidate gene for splitting is obtained from a random sample of size v. A typical choice for v is such that $v \approx \sqrt{p}$. We then grow the trees to maximum depth. Therefore, the two step randomization help to decorrelate the trees (Chen & Ishwaran, 2012). To determine the prediction for an unknown sample, an average over all the trees is taken for a regression problem and a majority vote for a classification problem (Friedman et al., 2001; Pappu & Pardalos, 2014; Do et al., 2009).

Random Forests Algorithm for Regression or Classification (Friedman et al., 2001)

1. For $b = 1$ to B (# random-forest trees):
   - Draw a bootstrap sample of size N from the training data.
   - Grow a random-forest tree, T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size, n_{min}, is reached.
     (a) Select v genes at random from the p genes.

   (b) Pick the best gene to split on among the v based on an impurity measure.
   (c) Using the selected gene, split the node into two daughter nodes.

2. To make a prediction for a new sample, x:

   Let $\hat{C}_b(x)$ be the class prediction of the b-th random-forest tree. Then

$$\hat{C}_{rf}^{B}(x) = \text{majority vote}\left\{\hat{C}_b(x)\right\}_{b=1}^{B}$$

Random forests comprise a number of decision trees, and each node in the decision tree is a condition on a single feature, which divides the data into two. Ginis index was used for the impurity measure in this work because it is the best for classification problems (Pal, 2005; Breiman et al., 2011). Ginis index is a measure of how often a randomly chosen element from a set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. It is computed as

$$\text{Gini(D)} = 1 - \sum_{i=1}^{m} p_i^2$$

Where D is a set of training samples and their associated class labels belongs to class i, p_i is the probability that a sample in D, the sum is computed over m classes.

RF are implemented using the *R* package *randomForest* (Breiman et al., 2011).

3.5 k-Nearest Neighbor (KNN)

The k-nearest neighbor classifiers (KNN) are known to be most useful instance-based learners. KNN is a non-parametric model (Yao & Ruzzo, 2006). If the classification is based on Euclidean distance in feature space, then k determines the number of neighbors to be used. In the testing set assign the new sample the class that is most likely among the k neighbors. The number of neighbors can be tuned to choose the optimal value of k (Stephens & Diesing, 2014; Dwivedi, 2016).

There are many types of similarity measure, such as Euclidean distance, cosine similarity, and Mahalanobis distance. The *knn* function uses the Euclidean distance as the default to find the k-th neighbors. Moreover, the Euclidean distance is simple and works well with continuous data (Wilson & Martinez, 1997; Ding & Peng, 2005). Since the gene expression data is continuous, the Euclidean distance was the preferred similarity measure.

The KNN uses the Euclidean distance measure to find the closest samples for the new sample. Suppose we have two samples, each one containing n genes. Specifically denote the two samples as $S_1 = (g_{11}, g_{12}, \ldots, g_{1n})'$ and $S_2 = (g_{21}, g_{22}, \ldots, g_{2n})'$.

Then the Euclidean distance is calculated as squared root of the sum of the squared differences in their corresponding values. Using the Euclidean distance definition, the distance between two points, dist(S_1, S_2), is given as

$$\text{dist}(S_1, S_2) = \sqrt{\sum_{j=1}^{n} (g_{1j} - g_{2j})^2}. \tag{3.14}$$

Figure 3.3 below present the idea of the k-nearest neighbor.

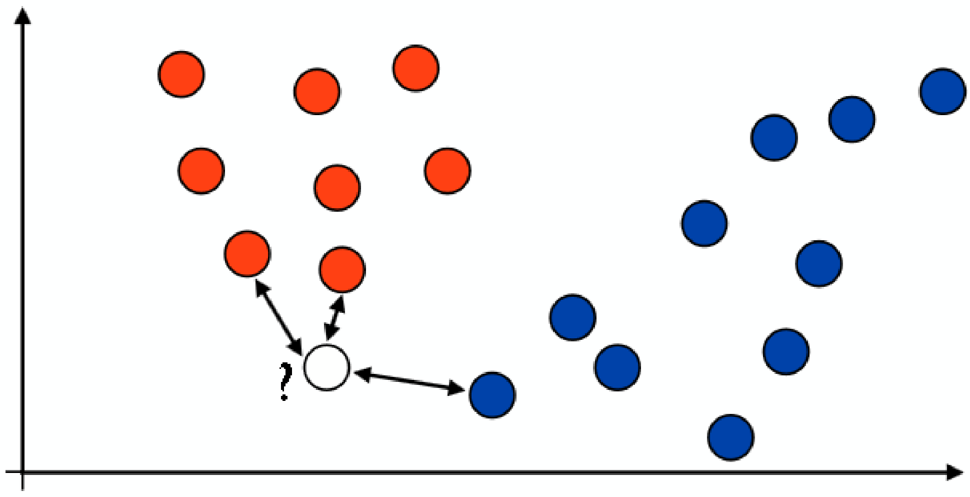

Figure 3.3 – k-nearest neighbor approach.

3.6 Performance Measures

In this paper, the comparison was based on seven performance measures, as defined below. These measures were calculated from the generic confusion matrix below.

Table 3.2: Structure of the confusion matrix

	True Condition	
Predicted Condition	Positive	Negative
Positive	True Positive (TP)	False Positive (FP)
Negative	False Negative (FN)	True Negative (TN)

1. **Accuracy** is the percentage of correctly classified samples:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}.$$

2. **Kappa** is a chance-corrected measure of agreement between the classifications and the true classes:

$$Kappa = \frac{\text{Accuracy} - \text{Random Accuracy}}{1 - \text{Random Accuracy}}.$$

$$\begin{aligned}\text{Random Accuracy} &= \frac{\text{ActNegitave} \times \text{PredNegitave} + \text{ActPositive} \times \text{PredPositive}}{\text{Total} \times \text{Total}} \\ &= \frac{(TN+FP) \times (TN+FN) + (FN+TP) \times (FP+TP)}{(TP+TN+FP+FN) \times (TP+TN+FP+FN)}.\end{aligned}$$

3. **Specificity** is the proportion of actual negatives which are predicted negative:

$$Specificity = \frac{TN}{(TN+FP)}.$$

4. **Sensitivity** is the proportion of actual positives which are predicted positive:

$$Sensitivity = \frac{TP}{(TP+FN)}.$$

5. **Balanced Error Rate (BER)** is the average of the proportion of wrong classifications in each class:

$$BER = \frac{1}{2}\left(\frac{FP}{(TN+FP)} + \frac{FN}{(FN+TP)}\right).$$

6. **Receiver Operating Characteristic Curve (ROC)** is a two-dimensional curve parameterized by one parameter of the classification algorithm, e.g. some threshold in the (true positive rate / false positive rate). That is, ROC curve is a plot of sensitivity against 1-specificity.

7. **Area Under the Curve (AUC)**:

$$AUC = \frac{1}{2}\left(\frac{TP}{(TP+FN)} + \frac{TN}{(TN+FP)}\right).$$

The AUC is between 0 and 1.

Chapter 4

Application and Results

Here, we discuss in detail the application steps that were followed in the study. Thereafter, we present the results of the application. First, we downloaded the datasets then processed them in several phases as shown in Fig 4.1. The data preparation included filtration, normalization and transformation, partitioning the dataset, and gene selection. Thereafter, the training of the candidate methods was done using the 10-fold cross validation sampling method using the training set, from which model comparison and selection of the best model(s) was achieved. Consequently, validation of the best model(s) was done using the validation data set. Finally, the performance results were used to distinguish between the different classification methods.

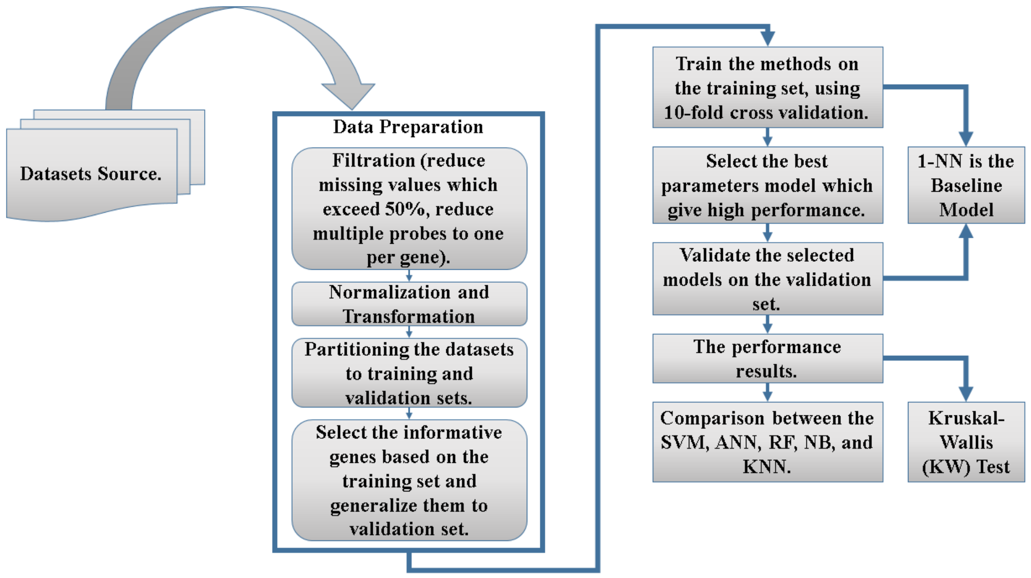
Datasets Source.
Data Preparation
Filtration (reduce missing values which exceed 50%, reduce multiple probes to one per gene).
Normalization and Transformation
Partitioning the datasets to training and validation sets.
Select the informative genes based on the training set and generalize them to validation set.
Train the methods on the training set, using 10-fold cross validation.
Select the best parameters model which give high performance.
1-NN is the Baseline Model
Validate the selected models on the validation set.
The performance results.
Comparison between the SVM, ANN, RF, NB, and KNN.
Kruskal-Wallis (KW) Test

Figure 4.1 – Flow diagram of the data processing stages until final model comparison and performance.

4.1 Data Source

The ten datasets were downloaded from the gene expression omnibus (GEO) web site (https://www.ncbi.nlm.nih.gov/geo/).

4.2 Data Preparation

The data preparation (data preprocessing), included many tasks for manipulation of the data into a suitable form for further analysis. These tasks included avoidance of inconsistency, poor quality data, missing data, removing poor genes which are not able to differentiate the classes, etc. The sequence of the specific tasks such as filtration, normalization, transformation, partitioning of the dataset to training and validation, and genes selection are illustrated in Fig 4.1. The details of each task are shown below.

4.2.1 Filtration

In most cases the microarray data contains probes for many genes that are either not expressed, expressed in only a few samples or expressed at a relatively constant level. So, filtering microarray data is a process of selecting a subset of available probes for exclusion or inclusion in the analysis. Filtration was performed to eliminate insufficiently expressed probes across the samples and those with excessive missing expression levels (Simon et al., 2007; Chaba et al., 2016). It allows for the exclusion of uninformative probes, hence a reduction in the number of genes leading to an increase in power of the results (Hackstadt & Hess, 2009). BRB-ArrayTools (Biometric Research Branch) software (https://brb.nci.nih.gov/BRB-ArrayTools/) was used to implement the filtration.

4.2.2 Normalization and Transformation

Normalization is a process of reducing the variation in gene expression. In other words, the reduction or removal of experimental variation. Each dataset was thus quantile-normalized and log2-transformed (Bolstad et al., 2003) under this sub-process. BRB-ArrayTools (Biometric Research Program) software (https://brb.nci.nih.gov/BRB-ArrayTools/) was used to implement normalization and log2 transformation of the datasets.

4.2.3 Datasets Partitioning

Each dataset was partitioned into two parts (the training and validation sets) in the ratio of 7 to 3, while keeping the distribution of the class attributes in the training

and validation sets the same as as in the original dataset. The partitioning was done using *createDataPartition* function from the *R* package called *caret*.
Since we did not have two separate datasets containing the same genes for each cancer-type, we used the split-validation method, while keeping the distribution of the class in the training and the validation sets the same as in the original data. We applied the 10-fold cross-validation method on the training set (which divides the training into 10 parts (folds), 9 folds for training and 1 fold for testing the models). The selected model was then validated using the validation set. We used the split-validation method due to its simplicity and wide application in microarray data analysis (see Valentini (2002); Wu et al. (2003); Hu et al. (2006); Dwivedi (2016)).

4.2.4 Features (Genes) Selection

Each component of a sample in the training data is called a feature (attribute). Feature selection is an important part of modelling because selecting the optimal data and representing it in the right order can affect the end results greatly. We used the two-sample and paired *t*-tests, with the 0.001 significance level threshold, for feature (gene) selection from the training set for building classification models. In this process the *t*-test is used to rank the genes based on the p-value comparing the mean expression values for each gene across the two classification groups. A gene which attains or exceeds the threshold p-value of 0.001 is considered informative (or differentially expressed) between the two groups.

4.3 Model Training

The ten-fold cross-validation method was used as the model training and testing method on each dataset. In a ten-fold cross-validation, a training dataset is equally divided into 10 partitions (folds), then each 9-fold data is used for "training" and the remaining one-fold for "testing". This implies that the training-testing step is iterated 10 times. The overall accuracy of a classification algorithm (method) would be calculated as the average of the ten accuracy measures from the 10 iterations. Consequantly, the best parameters model are obtained. The training was implemented using *train* function from the *R* package called *caret*.

4.4 Model Validation

After the training and testing stage, the trained model was applied to the validation set to yield the method accuracy and other performance measures for comparison with the other classification methods. The classification measures are then used to compare the performance of the classification methods.

4.5 Baseline Model

The baseline model is a model that gives some sense of what we are trying to achieve, thus conceptually the baseline model represents a very simple model. KNN with k=1 was selected as the baseline reference model to compare the performance of the other models against (Stephens & Diesing, 2014; Ganti, 2015).

4.6 Kruskal-Wallis (KW) Test

In the (Konig et al., 2008), they used exact McNemars test for the statistical comparison. Also, in the (Kruppa et al., 2014), they used the bootstrap test for comparing the Brier scores (BS), which is used for evaluating the performance of a probability estimator, unlike the other mentioned studies. In this thesis, we used the Kruskal-Wallis (KW) test for comparing the methods.

The Kruskal-Wallis (KW) ANOVA is the non-parametric equivalent of a one-way ANOVA. It is used when the assumptions of normality in the one-way ANOVA analysis may not hold. For this reason the KW test, tests the null hypothesis of no difference between three or more group medians against the alternative hypothesis that a significant difference exists between at least two medians. The KW test assume the samples in each group drawn from the population are random and have the same shape of the distribution (Ostertagová et al., 2014). The procedure for implementing the KW test are as follows:

1. Arrange the data from all samples in a single series in ascending order.
2. Rank the observations in the combined data in ascending order. In the case of repeated values assign the averaged rank position to all the repeated observations.
3. Sum the ranks for each of the different groups.
4. Calculate the value of H

$$H = \frac{12}{N(N+1)} \sum_{i=1}^{g} \frac{R_i^2}{n_i} - 3(N+1) \quad ,$$

where H = Kruskal-Wallis Test statistic, N is the total number of observations in all samples, R_i is the sum of the ranks assigned to group i, n_i is the number of observations in group i, g is the number of groups.

If the calculated value of KW is less than the critical value from the chi square distribution, then we cannot reject the null hypothesis and conclude that there is no significant difference between the group medians. However, when the KW statistic is significant, a multiple comparison approach becomes necessary for further analysis to determine exactly which groups are different.

4.7 Results

The results of microarray gene expression data were obtained using the R package *caret* (version 3.4.2). The optimal values of the parameters were determined automatically by training (tuning) each method on the training set using the *train* function, in order to achieve valid results. Thereafter, the model was validated using the validation set. The performance of the methods were based on seven measures (accuracy, sensitivity, specificity, kappa, ROC, AUC, and BER) as shown in Table 4.16 - 4.24. In addition, the respective ROC curves and the BER plots are given in Figure 4.4 - 4.21 for each data set. The ROC and the BER provide more insight into the performance of the methods used.

4.7.1 Training Results of the Methods on the GSE8401

Here, we present training results from the GSE8401 data set for all the methods in order to show the steps of how the best model was selected and how the selected model parameters were tuned. For the remaining nine data sets only validation results are presented. However, full details as those for the GSE8401 data set are provided in Appendix B.

Summary for the data set from the software for all models;
Number of samples: 59
Number of predictors: 1483
Re-sampling method: Cross-Validation (10 fold)

For all the models, accuracy was used to select the optimal model using the largest value.

k-Nearest Neighbors (KNN)

Table 4.1: Tuning Parameter Results of KNN for GSE8401

k	Accuracy	Kappa
5	0.9157143	0.8335385
$\boxed{7}$	$\boxed{0.9157143}$	$\boxed{0.8335385}$
9	0.8990476	0.7918718

The parameter k is the number of neighbors ″voting″ on the test samples class. The optimal model is that with k that maximizes the accuracy and kappa coefficient.

Table 4.2: Confusion Matrix of KNN on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	14	1
Prim	1	8

Naive Bayes (NB)

Table 4.3: Tuning Parameter Results of NB for GSE8401

$usekernel$	Accuracy	Kappa
FALSE	0.932381	0.8668718
$\boxed{\text{TRUE}}$	$\boxed{0.952381}$	$\boxed{0.9053333}$

The final tuning parameter values used for the model were fL = 0, $usekernel$ = TRUE and $adjust$ = 1 where fL is the factor for Laplace correction, default factor is 0, i.e. no correction, $usekernel$ is logical; if TRUE, density is used to estimate the densities of metric predictors, and $adjust$ is the bandwidth adjustment. The $adjust$ parameter is ignored when $usekernel$ = FALSE.

Table 4.4: Confusion Matrix of NB on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	14	1
Prim	1	8

Random Forests (RF)

Table 4.5: Tuning Parameter Results of RF for GSE8401

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.9323810}$	$\boxed{0.8515455}$
54	0.9323810	0.8503550
1483	0.8990476	0.7848788

The parameter $mtry$ is the number of variables randomly sampled as candidates at each split. Note that the default values are different for classification (square root(p) where p is number of variables in the feature (x)) and regression ($p/3$). Accuracy was used to select the optimal model using the largest value.
The final value used for the model was $mtry$ = 2.

Table 4.6: Confusion Matrix of RF on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	15	0
Prim	2	7

Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table 4.7: Tuning Parameter Results of SVM(RK) for GSE8401

C	Accuracy	Kappa
0.25	0.9523810	0.9053333
$\boxed{0.50}$	$\boxed{0.9690476}$	$\boxed{0.9386667}$
1.00	0.9690476	0.9386667

The parameter $sigma$ is the inverse kernel width used by the Gaussian, the Laplacian, the Bessel and the ANOVA kernel, while C is the trade off parameter for misclassification of training examples against simplicity of the decision surface. The final values used for the model were $sigma$ = 0.0004198846 and C = 0.5.

Table 4.8: Confusion Matrix of SVM(RK) on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	15	0
Prim	2	7

Support Vector Machines with Linear Kernel (SVM(LK))

Table 4.9: Tuning Parameter Results of SVM(LK) for GSE8401

C	Accuracy	Kappa
1	0.9690476	0.9386667

The tuning parameter C was held constant at a value of 1, where C it is defined as above in SVM(RK).

Table 4.10: Confusion Matrix of SVM(LK) on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	15	0
Prim	1	8

Support Vector Machines with Polynomial Kernel (SVM(PK))

Table 4.11: Tuning Parameter Results of SVM(PK) for GSE8401

$degree$	$scale$	C	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.001}$	$\boxed{0.25}$	$\boxed{0.9690476}$	$\boxed{0.9386667}$
1	0.001	0.50	0.9690476	0.9386667
1	0.001	1.00	0.9690476	0.9386667
1	0.010	0.25	0.9690476	0.9386667
1	0.010	0.50	0.9690476	0.9386667
1	0.010	1.00	0.9690476	0.9386667
1	0.100	0.25	0.9690476	0.9386667
1	0.100	0.50	0.9690476	0.9386667
1	0.100	1.00	0.9690476	0.9386667
2	0.001	0.25	0.9690476	0.9386667
2	0.001	0.50	0.9690476	0.9386667
2	0.001	1.00	0.9690476	0.9386667
2	0.010	0.25	0.9490476	0.8932121
2	0.010	0.50	0.9490476	0.8932121
2	0.010	1.00	0.9490476	0.8932121
2	0.100	0.25	0.8957143	0.7525195
2	0.100	0.50	0.8623810	0.6775195
2	0.100	1.00	0.8957143	0.7525195
3	0.001	0.25	0.9490476	0.8932121
3	0.001	0.50	0.9490476	0.8932121
3	0.001	1.00	0.9490476	0.8932121
3	0.010	0.25	0.9123810	0.8049004
3	0.010	0.50	0.9123810	0.8049004
3	0.010	1.00	0.9123810	0.8049004
3	0.100	0.25	0.8957143	0.7632338
3	0.100	0.50	0.9123810	0.8049004
3	0.100	1.00	0.9123810	0.8049004

The additional tuning parameters are the $degree$ of the polynomial and the $scale$ parameter which is c is the scaling parameter of the polynomial and tangent kernel is a convenient way of normalizing patterns without the need to modify the data itself. The final values used for the model were $degree$ = 1, $scale$ = 0.001 and C = 0.25.

Table 4.12: Confusion Matrix of SVM(PK) on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	15	0
Prim	2	7

Neural Networks (ANN)

Table 4.13: Tuning Parameter Results of ANN for GSE8401

size	*decay*	Accuracy	Kappa
1	0e+00	0.6438095	0.06666667
1	1e-04	0.7990476	0.53353846
$\boxed{1}$	$\boxed{\text{1e-01}}$	$\boxed{0.9690476}$	$\boxed{0.93866667}$
3	0e+00	0.7890476	0.43866667
3	1e-04	0.8504762	0.69840160
3	1e-01	0.9690476	0.93866667
5	0e+00	0.7004762	0.21666667
5	1e-04	0.9523810	0.90533333
5	1e-01	0.9690476	0.93866667

The parameter *size* is the number of units in the hidden which can be zero if there are skip-layer units while *decay* is the parameter for weight decay with the default is 0. The final values used for the model were *size* = 1 and *decay* = 0.1.

Table 4.14: Confusion Matrix of ANN on GSE8401 Validation Set

	Reference	
Prediction	Met	Prim
Met	15	0
Prim	1	8

After the training step for all the models on each data set, validation was done on the 30% of the data with the results as given in Table 4.15 for GSE8401. Similar summary tables for the remaining data sets as shown in Table 4.16 - 4.24 without training detail steps which are all indicated in Appendix B of the thesis.

4.7.2 GSE8401:

For this dataset, SVM (with linear kernel) and ANN scored the highest accuracy, kappa, AUC, and 1-BER values, while KNN and NB had the lowest values in terms of 1-BER (Table 4.15). The results were validated by the ROC curve (Fig 4.2) and the plot of kappa against 1-BER (Fig 4.3). Sensitivity was perfect for all the methods except for KNN and NB which both gave a value of 0.889.

Table 4.15: Summary accuracy measures for all the candidate methods using the GSE8401 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.917 (0.916)	0.822 (0.834)	0.889	0.933	0.911	0.089
RF	0.917 (0.932)	0.814 (0.852)	1	0.882	0.889	0.059
NB	0.917 (0.952)	0.822 (0.905)	0.889	0.933	0.911	0.089
ANN	0.958 (0.969)	0.909 (0.939)	1	0.938	0.944	0.031
$SVM(RK)$	0.917 (0.969)	0.814 (0.939)	1	0.882	0.889	0.059
$SVM(LK)$	0.958 (0.969)	0.909 (0.939)	1	0.938	0.944	0.031
$SVM(PK)$	0.917 (0.969)	0.814 (0.939)	1	0.882	0.889	0.059

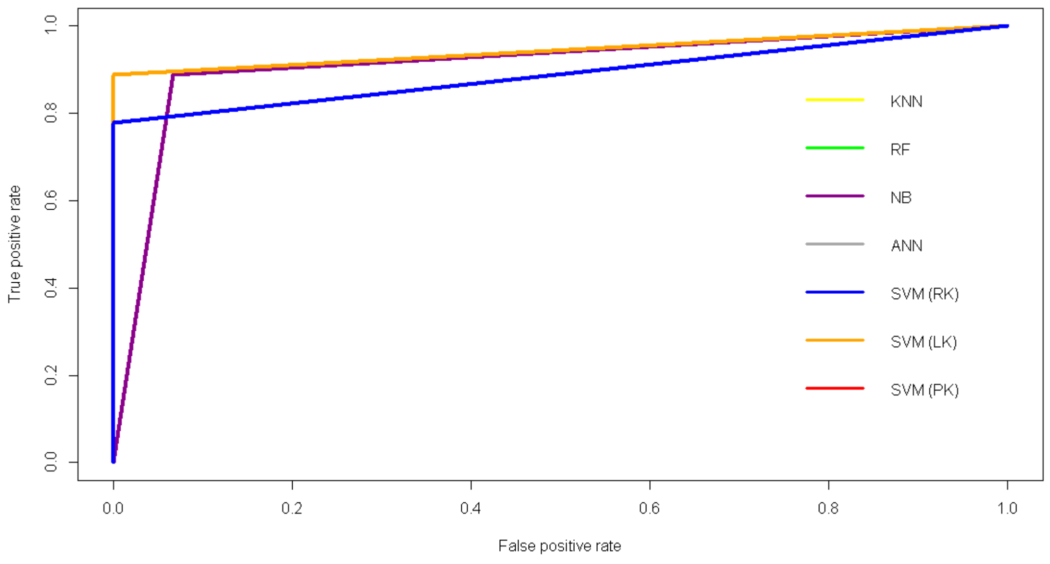

Figure 4.2 – ROC curves for all the methods applied to the GSE8401 dataset.

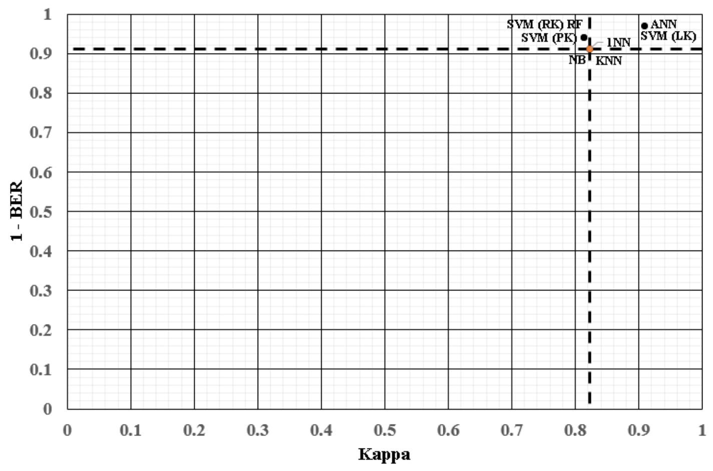

Figure 4.3 – Plot of 1 - BER against kappa for the GSE8401 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.3 GSE23988:

For this dataset, SVM with linear kernel, ANN, NB, and RF achieved the highest accuracy, kappa, AUC and 1-BER values, while SVM (radial and polynomial kernels) and KNN had a good results (Table 4.16). These results were further validated by the ROC curve (Fig 4.4) and the plot of kappa against 1-BER (Fig 4.5).

Table 4.16: Summary accuracy measures for all the candidate methods using the GSE23988 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.824 (0.93)	0.648 (0.855)	0.875	0.778	0.826	0.174
RF	0.882 (0.93)	0.767 (0.855)	1	0.8	0.889	0.1
NB	0.882 (0.955)	0.767 (0.905)	1	0.8	0.889	0.1
ANN	0.882 (0.955)	0.767 (0.905)	1	0.8	0.889	0.1
$SVM(RK)$	0.824 (0.955)	0.648 (0.905)	0.875	0.778	0.826	0.174
$SVM(LK)$	0.882 (0.955)	0.767 (0.905)	1	0.8	0.889	0.1
$SVM(PK)$	0.824 (0.955)	0.648 (0.905)	0.875	0.778	0.826	0.174

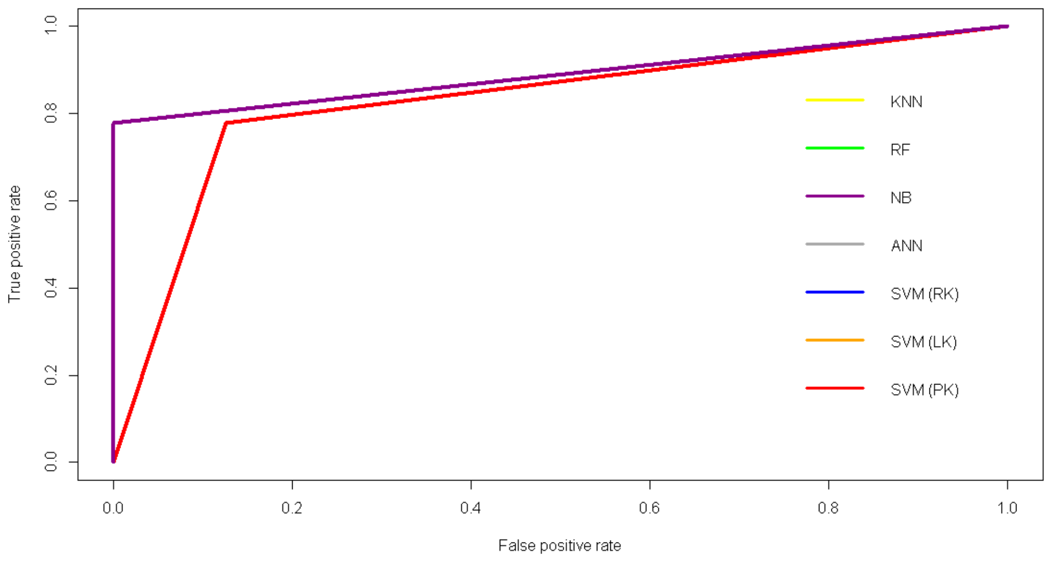

Figure 4.4 – ROC curves for all the methods applied to the GSE23988 dataset.

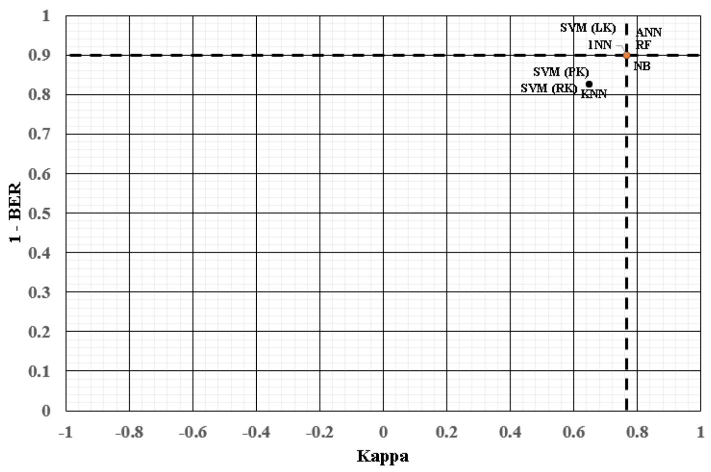

Figure 4.5 – Plot of 1 - BER against kappa for the GSE23988 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.4 GSE7670:

For this dataset, all the methods had perfect accuracy, kappa, AUC, and 1-BER values (Table 4.17). These results were further confirmed by the ROC curve (Fig 4.6) and the plot of kappa against 1-BER plot in (Fig 4.7). Logically it means under this dataset we cannot differentiate between the different methods.

Table 4.17: Summary accuracy measures for all the candidate methods using the GSE7670 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	1 (0.975)	1 (0.950)	1	1	1	0
RF	1 (0.975)	1 (0.950)	1	1	1	0
NB	1 (0.950)	1 (0.900)	1	1	1	0
ANN	1 (0.975)	1 (0.950)	1	1	1	0
$SVM(RK)$	1 (0.950)	1 (0.900)	1	1	1	0
$SVM(LK)$	1 (0.975)	1 (0.950)	1	1	1	0
$SVM(PK)$	1 (0.975)	1 (0.950)	1	1	1	0

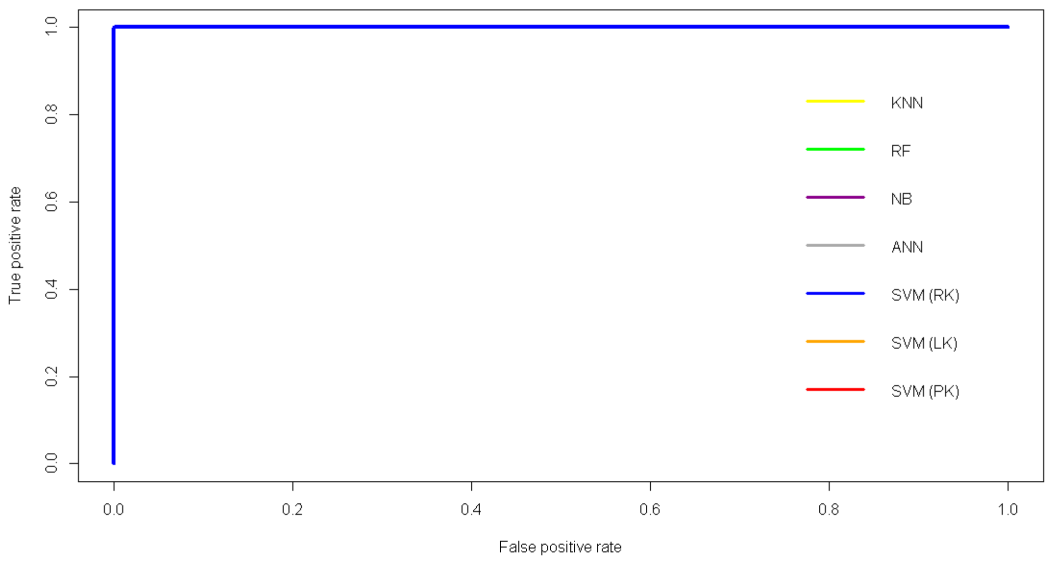

Figure 4.6 – ROC curves for all the methods applied to the GSE7670 dataset.

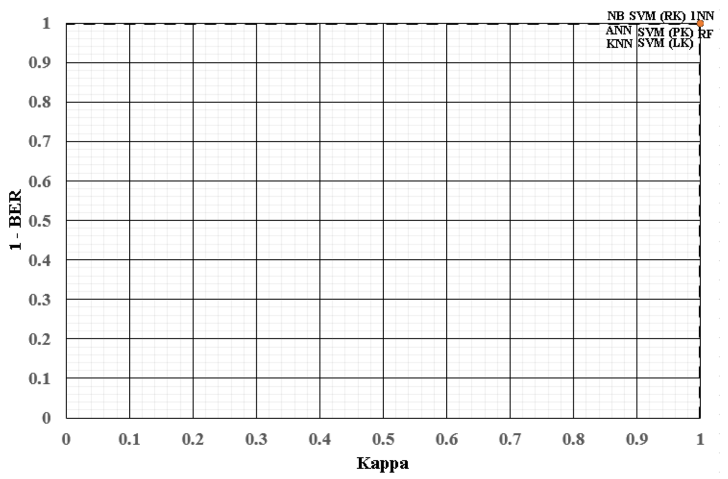

Figure 4.7 – Plot of 1 - BER against kappa for the GSE7670 dataset. The dashed lines represent the values of the baseline model.

4.7.5 GSE10072:

For this dataset, all the methods showed perfect accuracy, kappa, AUC, and 1 - BER values (Table 4.18). These results were further validated by the ROC curve (Fig 4.8) and the plot of kappa against 1-BER (Fig 4.9). However under this dataset one cannot place any method better than any other.

Table 4.18: Summary accuracy measures for all the candidate methods using the GSE10072 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	1 (1)	1 (1)	1	1	1	0
RF	1 (0.98)	1 (0.962)	1	1	1	0
NB	1 (0.98)	1 (0.962)	1	1	1	0
ANN	1 (1)	1 (1)	1	1	1	0
$SVM(RK)$	1 (1)	1 (1)	1	1	1	0
$SVM(LK)$	1 (1)	1 (1)	1	1	1	0
$SVM(PK)$	1 (1)	1 (1)	1	1	1	0

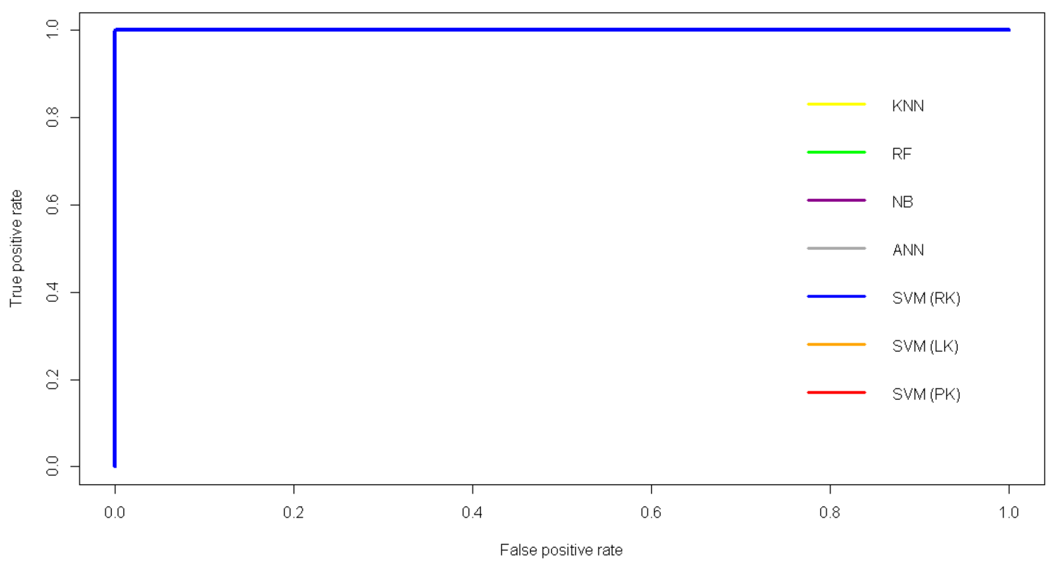

Figure 4.8 – ROC curves for all the methods applied to the GSE10072 dataset.

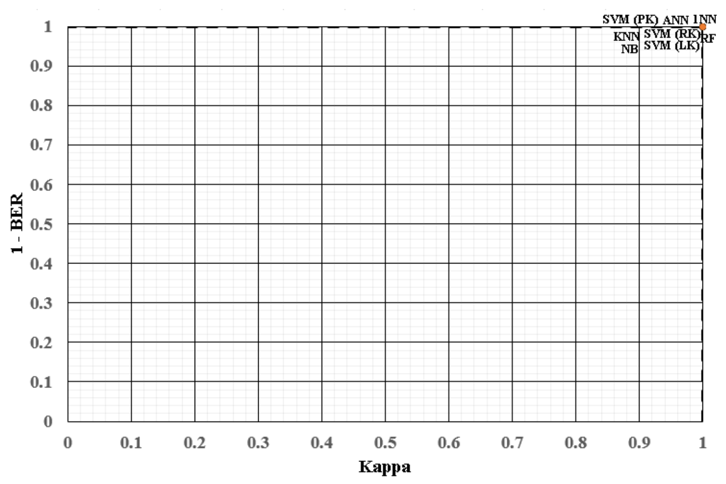

Figure 4.9 – Plot of 1 - BER against kappa for the GSE10072 dataset. The dashed lines represent the values of the baseline model.

4.7.6 GSE10245:

For this dataset, SVM (with radial, polynomial, and linear kernels), RF, and NB achieved perfect accuracy, kappa, AUC, and 1-BER values, while ANN and KNN had high values (Table 4.19). These results were further supported by the ROC curve (Fig 4.10) and the plot of kappa against 1-BER (Fig 4.11).

Table 4.19: Summary accuracy measures for all the candidate methods using the GSE10245 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.941 (0.930)	0.850 (0.805)	1	0.923	0.9	0.038
RF	1 (0.980)	1 (0.955)	1	1	1	0
NB	1(0.955)	1 (0.905)	1	1	1	0
ANN	0.941 (0.980)	0.850 (0.955)	1	0.923	0.9	0.038
$SVM(RK)$	1 (0.980)	1 (0.955)	1	1	1	0
$SVM(LK)$	1 (0.980)	1 (0.955)	1	1	1	0
$SVM(PK)$	1 (0.980)	1 (0.955)	1	1	1	0

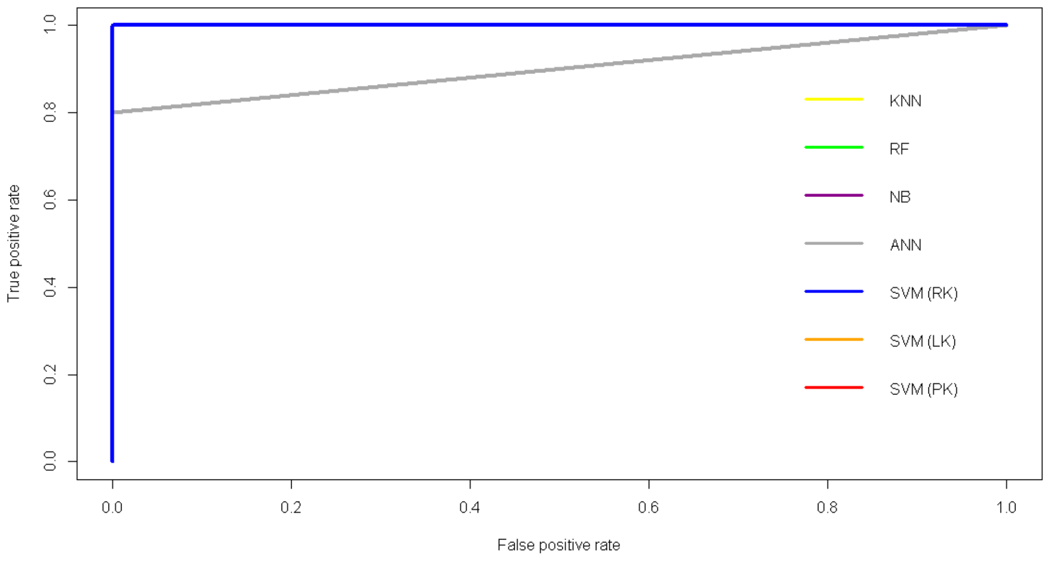

Figure 4.10 – ROC curves for all the methods applied to the GSE10245 dataset.

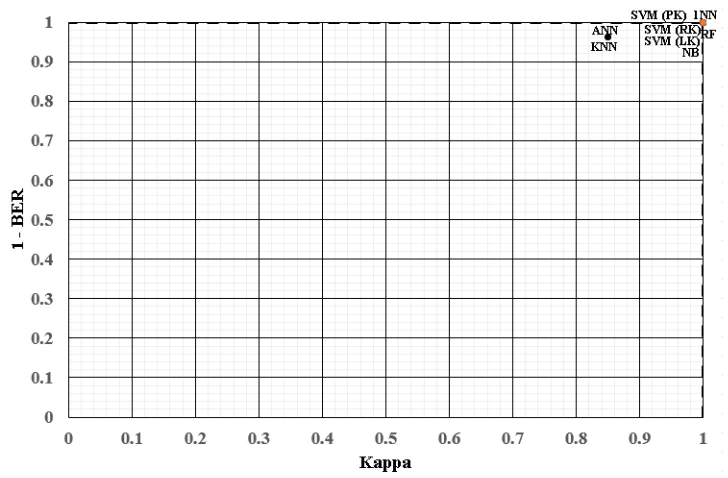

Figure 4.11 – Plot of 1 - BER against kappa for the GSE10245 dataset. The dashed lines represent the values of the baseline model.

4.7.7 GSE25136:

For this dataset, ANN and KNN methods achieved the highest accuracy, kappa, AUC, and 1-BER values, while NB had the lowest values (Table 4.20). The results were supported by the ROC curve (see Fig 4.12) and the plot of kappa against 1-BER (Fig 4.13).

Table 4.20: Summary accuracy measures for all the candidate methods using the GSE25136 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.696 (0.960)	0.388 (0.916)	0.700	0.692	0.693	0.304
RF	0.652 (0.930)	0.298 (0.862)	0.667	0.643	0.648	0.345
NB	0.565 (0.947)	0.129 (0.895)	0.546	0.583	0.564	0.436
ANN	0.696 (0.910)	0.388 (0.816)	0.700	0.692	0.693	0.304
$SVM(RK)$	0.609 (0.927)	0.213 (0.850)	0.600	0.615	0.606	0.392
$SVM(LK)$	0.609 (0.927)	0.219 (0.842)	0.583	0.636	0.610	0.390
$SVM(PK)$	0.652 (0.927)	0.303 (0.850)	0.636	0.667	0.652	0.348

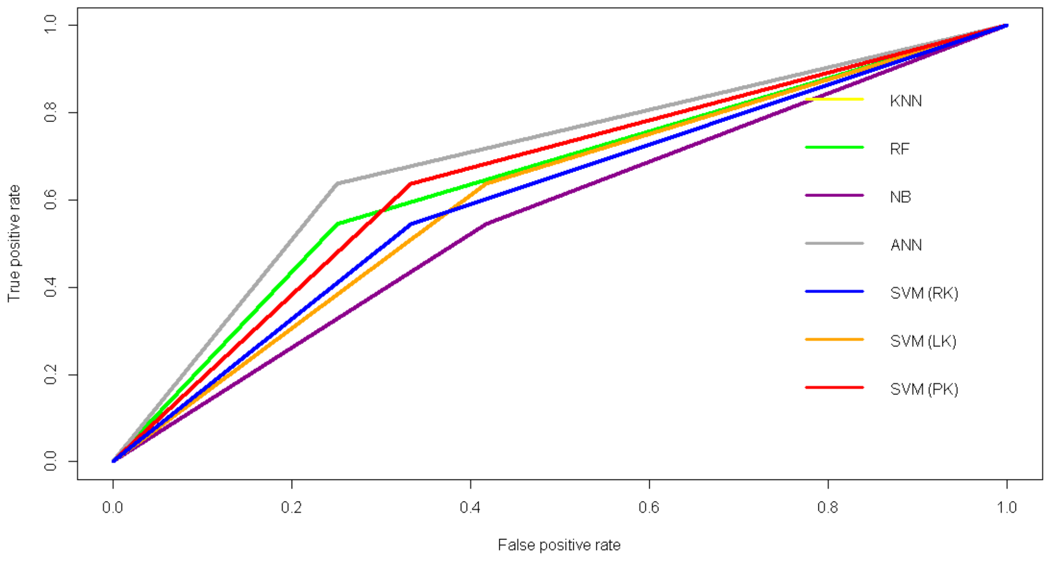

Figure 4.12 – ROC curves for all the methods applied to the GSE25136 dataset.

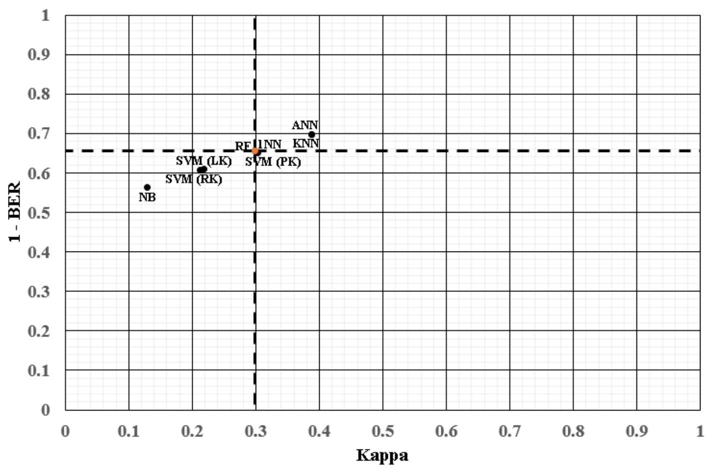

Figure 4.13 – Plot of 1 - BER against kappa for the GSE25136 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.8 GSE35896:

For this dataset, the NB achieved the highest accuracy, kappa, AUC, and 1-BER values, while SVM (with linear kernel) had the lowest values (Table 4.21). These were validated by the ROC curve (Fig 4.14) and the plot of kappa against 1-BER (Fig 4.15).

Table 4.21: Summary accuracy measures for all the candidate methods using the GSE35896 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.706 (0.935)	0.414 (0.867)	0.667	0.750	0.708	0.292
RF	0.765 (0.960)	0.514 (0.929)	1	0.692	0.750	0.154
NB	0.882 (0.980)	0.764 (0.962)	0.875	0.889	0.882	0.118
ANN	0.706 (0.960)	0.414 (0.917)	0.667	0.750	0.708	0.292
$SVM(RK)$	0.824 (0.980)	0.643 (0.962)	0.857	0.800	0.819	0.171
$SVM(LK)$	0.647 (0.955)	0.292 (0.912)	0.625	0.667	0.646	0.354
$SVM(PK)$	0.824 (1)	0.643 (1)	0.857	0.800	0.819	0.171

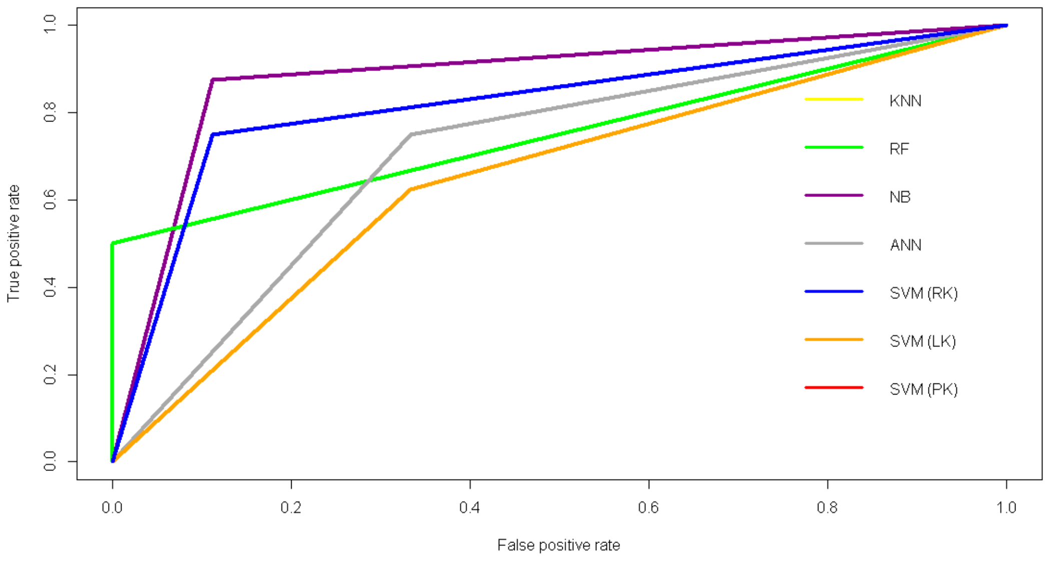

Figure 4.14 – ROC curves for all the methods applied to the GSE35896 dataset.

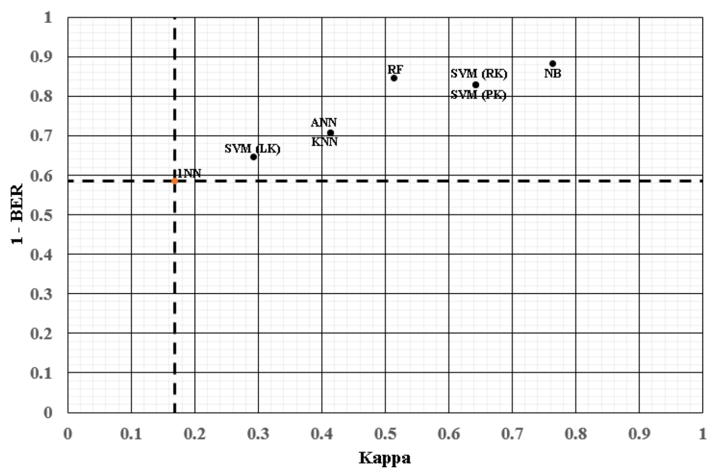

Figure 4.15 – Plot of 1 - BER against kappa for the GSE35896 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.9 GSE103091:

For this dataset, the RF achieved the highest accuracy, and 1-BER values. Moreover, NB had the highest kappa, AUC, and 1-BER, while KNN had lowest values in terms of kappa, AUC, and 1-BER values. Besides, SVM (with linear and polynomial kernels) performed nearly the same as KNN (Table 4.22). These results were validated by the ROC curve (Fig 4.16) and the plot of kappa against 1-BER (Fig 4.17).

Table 4.22: Summary accuracy measures for all the candidate methods using the GSE103091 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.677 (0.903)	0.025 (0.701)	0.333	0.714	0.510	0.476
RF	0.710 (0.891)	0.157 (0.690)	0.500	0.741	0.566	0.380
NB	0.677 (0.867)	0.162 (0.660)	0.429	0.750	0.576	0.411
ANN	0.677 (0.889)	0.162 (0.749)	0.429	0.750	0.576	0.536
$SVM(RK)$	0.677 (0.892)	0.099 (0.706)	0.400	0.731	0.543	0.435
$SVM(LK)$	0.645 (0.892)	0.045 (0.714)	0.333	0.720	0.520	0.473
$SVM(PK)$	0.645 (0.917)	0.045 (0.771)	0.333	0.720	0.520	0.473

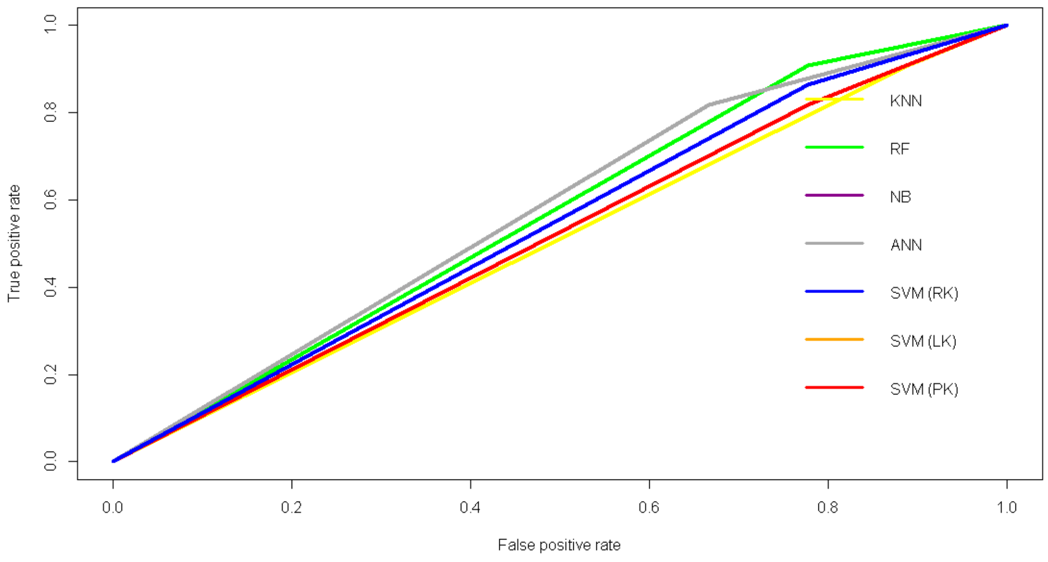

Figure 4.16 – ROC curves for all the methods applied to the GSE103091 dataset.

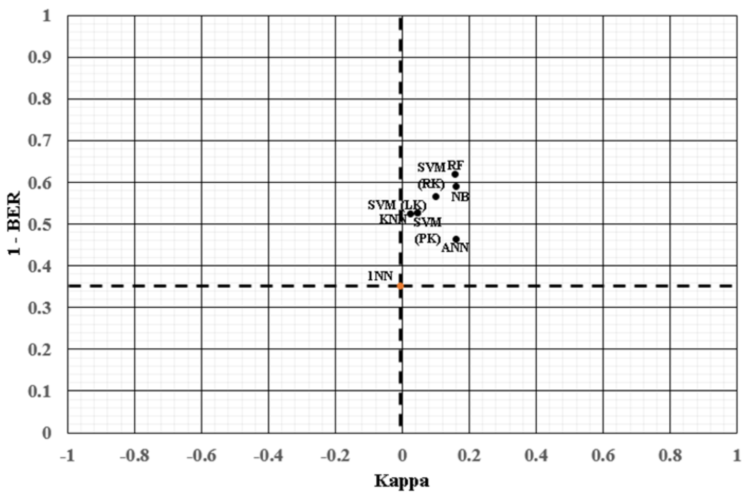

Figure 4.17 – Plot of 1 - BER against kappa for the GSE103091 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.10 GSE5851:

For this dataset, KNN and NB had the highest accuracy, kappa, AUC, and 1-BER values, followed by SVM (with radial kernel) and RF, which performed nearly the same. SVM (with linear kernel) had the lowest values (Table 4.23). The results were further validated by the ROC curve (Fig 4.18) and the plot of kappa against 1-BER (Fig 4.19).

Table 4.23: Summary accuracy measures for all the candidate methods using the GSE5851 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.700 (0.900)	0.341 (0.787)	0.769	0.571	0.670	0.330
RF	0.700 (0.870)	0.241 (0.632)	0.706	0.667	0.604	0.314
NB	0.700 (0.895)	0.341 (0.732)	0.769	0.571	0.670	0.330
ANN	0.600 (0.915)	0.059 (0.821)	0.667	0.400	0.527	0.467
$SVM(RK)$	0.700 (0.895)	0.294 (0.732)	0.733	0.600	0.637	0.333
$SVM(LK)$	0.550 (0.870)	-0.023 (0.699)	0.643	0.333	0.489	0.512
$SVM(PK)$	0.650 (0.895)	0.205 (0.732)	0.714	0.500	0.599	0.393

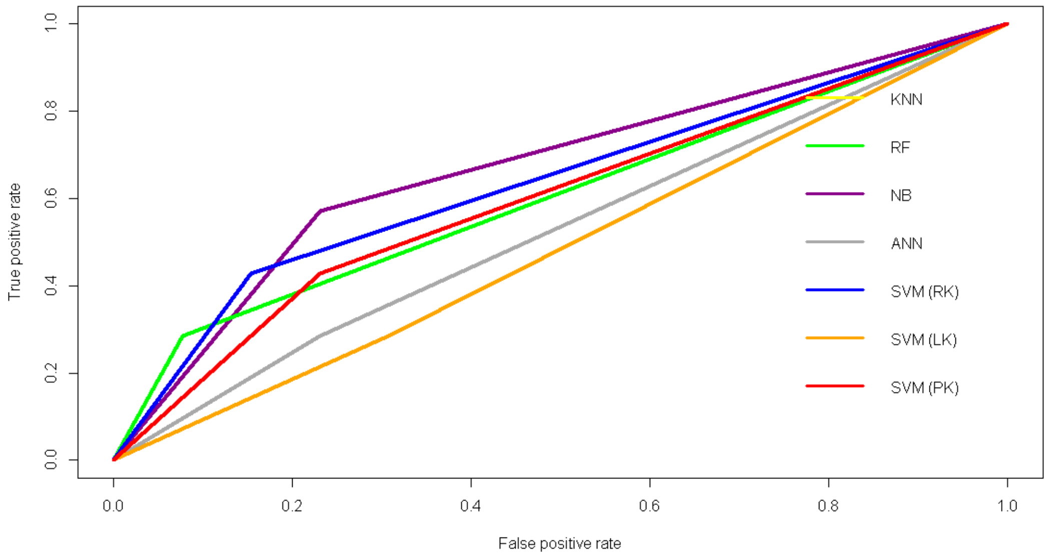

Figure 4.18 – ROC curves for all the methods applied to the GSE5851 dataset.

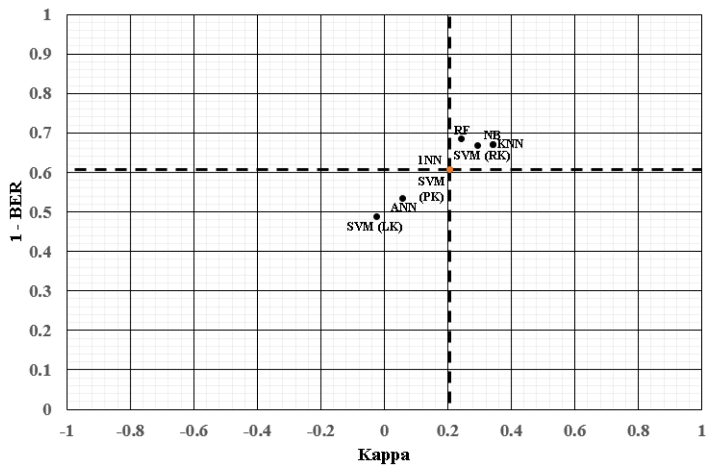

Figure 4.19 – Plot of 1 - BER against kappa for the GSE5851 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

4.7.11 GSE32962:

For this dataset, SVM (with linear kernel) and NB achieved the highest accuracy, kappa, AUC, and 1-BER values, while RF had the lowest values (Table 4.24). Besides, SVM (with polynomial kernel), ANN, and KNN had the same performance, and SVM (with radial kernel) had a similar performance to RF. These results were further supported by the ROC curve (Fig 4.20) and the plot of kappa against 1-BER (Fig 4.21).

Table 4.24: Summary accuracy measures for all the candidate methods using the GSE32962 data set. Accuracy and Kappa values were obtained for both the validation and training (in bracket) sets.

Method	*Accuracy*	*Kappa*	*Sensitivity*	*Specificity*	*AUC*	*BER*
KNN	0.667 (0.950)	0.314 (0.900)	0.600	0.714	0.657	0.343
RF	0.583 (1)	0.167 (1)	0.500	0.667	0.586	0.417
NB	0.667 (1)	0.351 (1)	0.571	0.800	0.686	0.314
ANN	0.667 (0.950)	0.314 (0.900)	0.600	0.714	0.657	0.343
$SVM(RK)$	0.583 (1)	0.211 (1)	0.500	0.750	0.614	0.375
$SVM(LK)$	0.667 (1)	0.351 (1)	0.571	0.800	0.686	0.314
$SVM(PK)$	0.667 (1)	0.314 (1)	0.600	0.714	0.657	0.343

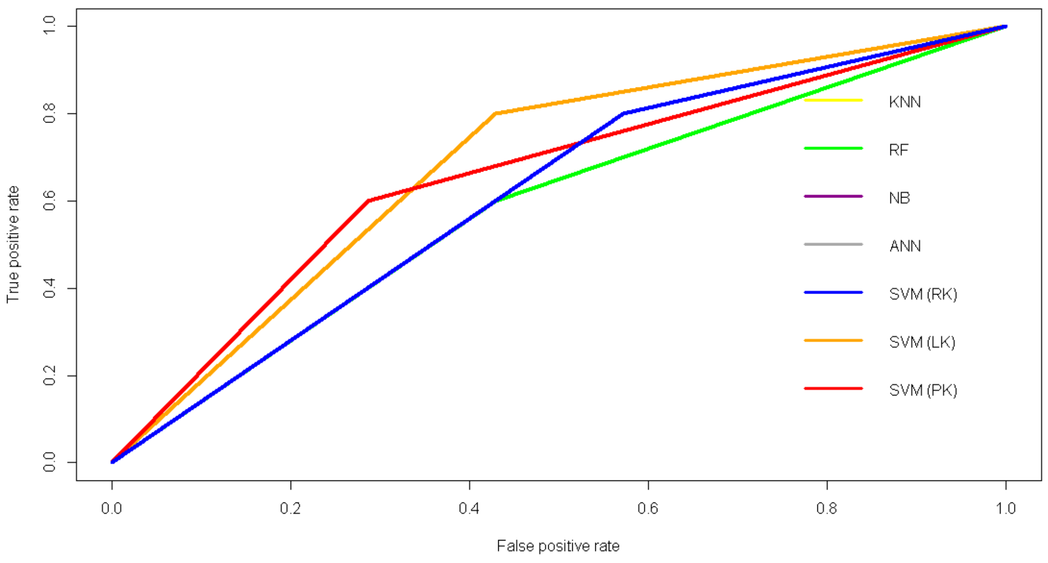

Figure 4.20 – ROC curves for all the methods applied to the GSE32962 dataset.

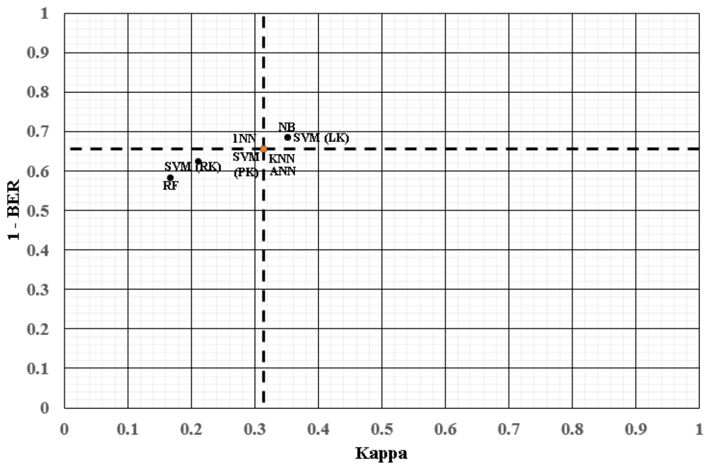

Figure 4.21 – Plot of 1 - BER against kappa for the GSE32962 dataset. The dashed lines represent the values of the baseline model. The best performing models are to the top-right.

Table 4.25: Average accuracies of the seven classification methods based on the ten validation datasets.

Dataset	*SVM(RK)*	*SVM(LK)*	*SVM(PK)*	*ANN*	*RF*	*NB*	*KNN*
GSE23988	0.824	0.882	0.824	0.882	0.882	0.882	0.824
GSE7670	1	1	1	1	1	1	1
GSE8401	0.917	0.958	0.917	0.958	0.917	0.917	0.917
GSE10072	1	1	1	1	1	1	1
GSE10245	1	1	1	0.941	1	1	0.941
GSE25136	0.609	0.609	0.652	0.696	0.652	0.565	0.696
GSE35896	0.824	0.647	0.824	0.706	0.765	0.882	0.706
GSE103091	0.677	0.645	0.645	0.677	0.710	0.677	0.677
GSE5851	0.700	0.550	0.650	0.600	0.700	0.700	0.700
GSE32962	0.583	0.667	0.667	0.667	0.583	0.667	0.667
Average	0.813	0.796	0.818	0.813	0.821	0.829	0.813

Chapter 5

Discussion and Conclusion

Different types of cancer are increasingly becoming prevalent and continue to affects millions of people worldwide. Due to the alarming increase of cancer incidence, both in developed and developing countries, detection and classification in the early stages of the disease are very important for the diagnosis, treatment, and containment of the disease. Microarray technology allows the study of thousand of genes simultaneously, unlike traditional molecular biology tools, which only allow the study of single genes at a time. The use of the high-dimensional microarray gene expression data has necessitated the development of powerful statistical classification methods such as support vector machines (SVM), artificial neural networks (ANN), random forests (RF), and linear discriminant analysis (LDA), naive bayes (NB), *k*-nearest neighbor (KNN) among others.

In this work, we compared the performance of seven of such methods (SVM (linear kernel), SVM(polynomial kernel), SVM(radial basis kernel), ANN, RF, NB, and KNN), using seven performance measures (accuracy, kappa, sensitivity, specificity, area under the curve (AUC), receiver operating curve (ROC), balanced error rate (BER)). We controlled for platform effect by using datasets from a single platform. Novianti et al. (2014) showed that applying different methods to datasets from different platforms may yield misleading results, due to confounding. We used KNN with $k = 1$ as the baseline model in the comparisons (Stephens & Diesing, 2014). We also used ten datasets from the top five common cancers in men and women as reported in Torre et al. (2015), as opposed to previous studies that compared the methods based on one dataset from a particular cancer type as in Delen et al. (2005); Dwivedi (2016); Valentini (2002); Haury et al. (2011), and Abusamra (2013).

In seven of the ten datasets, NB performed the best, followed by SVM(LK), ANN, and RF (see Table 4.16, 4.17, 4.18, 4.19, 4.21, 4.23, and 4.24), while in the remaining datasets, NB had nearly the same or equal performance to SVM, ANN, RF and KNN. For one dataset, RF had the best performance in terms of accuracy, sensitivity, and

BER (Table 4.22).

To further validate the statistical results, we plotted 1-BER against kappa for each of the datasets, as in Stephens & Diesing (2014). The best performing methods appeared in the top right quadrant (top right corner). Methods in the bottom right corner of the plot were good in terms of the kappa coefficient but poor in terms of 1-BER. Methods to the top left corner were good with respect to 1-BER but poor in terms of the kappa coefficients. Methods in the bottom left corner are poor both in terms of the kappa coefficient and 1-BER. All methods performed the same for the $GSE7670$ and $GSE10072$ datasets, which contained paired samples. On average, the top three methods were NB, RF, and SVM(PK), as shown in Table 4.25. However, there were no statistically significant differences in accuracies among the methods (Kruskal-Wallis test statistic $= 0.60229, p = 0.9964$), despite the visual and absolute value differences in performance.

Thus, our approach further validates the superiority of NB in cancer classification and could help in the identification of robust methods in similar high dimensional settings. The success of NB as a classifier is attributed to its simplicity to implement and the fact that it requires fewer training samples to achieve a good accuracy. NB assumes that the attributes are conditionally independent given the class (see Kelemen et al. (2003); Ahmed et al. (2017)). Future research should extend our approaches to multiple classification problems, using other "omics" datasets. It would also be interesting to compare the performance of the nearest shrunken centroids method discussed in Tibshirani et al. (2003) to NB, which is currently missing in the literature. Moreover, our ongoing research efforts are geared toward the ensemble methods, in which a set of the classifiers are combined to produce improved results.

Our study was limited by the type of class attribute arising from the type of medical question of interest in the dataset (e.g. tumor type, prognostic or treatment response/status) and class size imbalance. Our class attributes consisted of prognostic response (metastasis status and recurrence status), estrogen receptor, tumor and tissue type. Moreover, the handling of the missing data plays role in the methods performance. With the exception of the datasets with paired samples, the positive and negative class size were unbalanced (Table 2.1). Some of these limitations have been encountered in previous studies as well (see Novianti et al. (2014)). The current analysis focused on data where the outcome variable is dichotomous but in reality data where the outcome variable is polychotomous are ubiquitously encountered. Thus, extension of the research to methods that can deal with such more informative high dimensional type of data sets is necessary.

References

Abusamra, H. (2013). A comparative study of feature selection and classification methods for gene expression data of glioma. *Procedia Computer Science*, *23*, 5–14.

ACS (2017). Cancer facts & figures 2017.

Ahmed, M., Shahjaman, M., Rana, M., Mollah, M., & Haque, N. (2017). Robustification of naïve bayes classifier and its application for microarray gene expression data analysis. *BioMed Research International*, *2017*.

Bolstad, B. M., Irizarry, R. A., Åstrand, M., & Speed, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, *19*(2), 185–193.

Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, (pp. 144–152). ACM.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32.

Breiman, L., Cutler, A., Liaw, A., & Wiener, M. (2011). Packagerandomforest. *software available at URL: http://stat-www. berkeley. edu/users/breiman/RandomForests*.

Brown, M. P., Grundy, W. N., Lin, D., Cristianini, N., Sugnet, C., Ares, M., & Haussler, D. (1999). Support vector machine classification of microarray gene expression data. *University of California, Santa Cruz, Technical Report UCSC-CRL-99-09*.

Chaba, L., Odhiambo, J., & Omolo, B. (2016). Evaluation of methods for gene selection in melanoma cell lines. *International Journal of Statistics in Medical Research*, *6*(1), 1–9.

Chen, X., & Ishwaran, H. (2012). Random forests for genomic data analysis. *Genomics*, *99*(6), 323–329.

Chu, F., & Wang, L. (2003). Gene expression data analysis using support vector machines. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 3, (pp. 2268–2271). IEEE.

Dalma-Weiszhausz, D. D., Warrington, J., Tanimoto, E. Y., & Miyada, C. G. (2006). [1] the affymetrix genechip® platform: An overview. *Methods in enzymology*, *410*, 3–28.

De Campos, L. M., Cano, A., Castellano, J. G., & Moral, S. (2011). Bayesian networks classifiers for gene-expression data. In *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*, (pp. 1200–1206). IEEE.

Delen, D., Walker, G., & Kadam, A. (2005). Predicting breast cancer survivability: a comparison of three data mining methods. *Artificial Intelligence in Medicine*, *34*(2), 113–127.

Ding, C., & Peng, H. (2005). Minimum redundancy feature selection from microarray gene expression data. *Journal of bioinformatics and computational biology*, *3*(02), 185–205.

Do, T.-N., Lenca, P., Lallich, S., & Pham, N.-K. (2009). Classifying very-high-dimensional data with random forests of oblique decision trees. In *EGC (best of volume)*, (pp. 39–55). Springer.

Dudoit, S., Fridlyand, J., & Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association*, *97*(457), 77–87.

Dwivedi, A. K. (2016). Artificial neural network model for effective cancer classification using microarray gene expression data. *Neural Computing and Applications*, (pp. 1–10).

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning*, vol. 1. Springer series in statistics New York.

Furey, T. S., Cristianini, N., Duffy, N., Bednarski, D. W., Schummer, M., & Haussler, D. (2000). Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, *16*(10), 906–914.

Ganti, D. (2015). *Lung Cancer Subtype Classification from Whole Slide Histopathological Images*. Ph.D. thesis.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., & Caligiuri, E. S. L., C. D. Bloomfield (1999). Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *science*, *286*(5439), 531–537.

Hackstadt, A. J., & Hess, A. M. (2009). Filtering for increased power for microarray data analysis. *BMC bioinformatics*, *10*(1), 11.

Han, J., Pei, J., & Kamber, M. (2011). *Data mining: concepts and techniques*. Elsevier.

Haury, A.-C., Gestraud, P., & Vert, J.-P. (2011). The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. *PLOS One*, *6*(12), e28210.

Hu, H., Li, J., Plank, A., Wang, H., & Daggard, G. (2006). A comparative study of classification methods for microarray data analysis. In *Proceedings of the fifth Australasian conference on Data mining and analystics-Volume 61*, (pp. 33–37). Australian Computer Society, Inc.

Huang, S., CAI, N., PACHECO, P. P., NARANDES, S., WANG, Y., & XU, W. (2018). Applications of support vector machine (svm) learning in cancer genomics. *Cancer Genomics-Proteomics*, *15*(1), 41–51.

Iwamoto, T., Bianchini, G., Booser, D., Qi, Y., Coutant, C., Ya-Hui Shiang, C., Santarpia, L., Matsuoka, J., Hortobagyi, G. N., Symmans, J., William Fraser, OShaughnessy, B., Hellerstedt, J., Pippen, F., Andre, R., Simon, L., & Pusztai (2010). Gene pathways associated with prognosis and chemotherapy sensitivity in molecular subtypes of breast cancer. *Journal of the National Cancer Institute*, *103*(3), 264–272.

Jemal, A., Bray, F., Center, M. M., Ferlay, J., Ward, E., & Forman, D. (2011). Global cancer statistics. *CA: a cancer journal for clinicians*, *61*(2), 69–90.

Jézéquel, P., Loussouarn, D., Guérin-Charbonnel, C., Campion, L., Vanier, A., Gouraud, W., Lasla, H., Guette, C., Valo, I., Verrièle, M., Véronique, & Campone (2015). Gene-expression molecular subtyping of triple-negative breast cancer tumours: importance of immune response. *Breast Cancer Research*, *17*(1), 43.

Karatzoglou, A., Smola, A., Hornik, K., & Karatzoglou, M. A. (2016). Package kernlab.

Kelemen, A., Zhou, H., Lawhead, P., & Liang, Y. (2003). Naive bayesian classifier for microarray data. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, vol. 3, (pp. 1769–1773). IEEE.

Khambata-Ford, S., Garrett, C. R., Meropol, N. J., Basik, M., Harbison, C. T., Wu, S., Wong, T. W., Huang, X., Takimoto, C. H., Godwin, A. K., , B. R., Tan, S. S., Krishnamurthi, H. A., Burris III, E. A., Poplin, M., Hidalgo, J., Baselga, E. A., Clark, D. J., & Mauro (2007). Expression of epiregulin and amphiregulin and k-ras mutation status predict disease control in metastatic colorectal cancer patients treated with cetuximab. *Journal of Clinical Oncology*, *25*(22), 3230–3237.

Khan, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C. R., Peterson, C., Peterson, P. S., & Meltzer (2001). Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, *7*(6), 673–679.

Konig, I., Malley, J. D., Pajevic, S., Weimar, C., Diener, H.-C., & Ziegler, A. (2008). Patient-centered yes/no prognosis using learning machines. *International journal of data mining and bioinformatics*, *2*(4), 289–341.

Kruppa, J., Liu, Y., Diener, H.-C., Holste, T., Weimar, C., König, I. R., & Ziegler, A. (2014). Probability estimation with machine learning methods for dichotomous and multicategory outcome: Applications. *Biometrical Journal*, *56*(4), 564–583.

Kuner, R., Muley, T., Meister, M., Ruschhaupt, M., Buness, A., Xu, E. C., Schnabel, P., Warth, A., Poustka, A., Sültmann, H., & Hoffmann, H. (2009). Global gene expression analysis reveals specific patterns of cell junctions in non-small cell lung cancer subtypes. *Lung Cancer*, *63*(1), 32–38.

Landi, M. T., Dracheva, T., Rotunno, M., Figueroa, J. D., Liu, H., Dasgupta, A., Mann, F. E., Fukuoka, J., Hames, M., Bergen, A. W., Sharon E. Murphy, S., Yang, P., C. Pesatori, A., Consonni, D., Alberto Bertazzi, P., Wacholder, S., H. Shih, J., E. Caporaso, N., & Jen, J. (2008). Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival. *PLOS One*, *3*(2), e1651.

Liu, Z., Chen, D., & Bensmail, H. (2005). Gene expression data classification with kernel principal component analysis. *BioMed Research International*, *2005*(2), 155–159.

Mahmoud, O., Harrison, A., Perperoglou, A., Gul, A., Khan, Z., Metodiev, M. V., & Lausen, B. (2014). A feature selection method for classification within functional genomics experiments based on the proportional overlapping score. *BMC Bioinformatics*, *15*(1), 274.

Moguerza, J. M., & Muñoz, A. (2006). Support vector machines with applications. *Statistical Science*, (pp. 322–336).

Morhason-Bello, I. O., Odedina, F., Rebbeck, T. R., Harford, J., Dangou, J.-M., Denny, L., & Adewole, I. F. (2013). Challenges and opportunities in cancer control in africa: a perspective from the african organisation for research and training in cancer. *The Lancet Oncology*, *14*(4), e142–e151.

Moten, A., Schafer, D., & Ferrari, M. (2014). Redefining global health priorities: Improving cancer care in developing settings. *Journal of global health*, *4*(1).

Musa, A. B. (2014). Gene expression data classification with kernel independent component analysis. *Research Journal of Mathematical and Statistical Sciences, ISSN 23206047*, *2(5)*, 1–7.

Novianti, P. W., Roes, K. C., & Eijkemans, M. J. (2014). Evaluation of gene expression classification studies: factors associated with classification performance. *PLOS One*, *9*(4), e96063.

Olsen, M. (2015). Cancer in sub-saharan africa: The need for new paradigms in global health.

Ostertagová, E., Ostertag, O., & Kováč, J. (2014). Methodology and application of the kruskal-wallis test. In *Applied Mechanics and Materials*, vol. 611, (pp. 115–120). Trans Tech Publ.

Pal, M. (2005). Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, *26*(1), 217–222.

Pappu, V., & Pardalos, P. M. (2014). High-dimensional data classification. In *Clusters, Orders, and Trees: Methods and Applications*, (pp. 119–150). Springer.

Qi, Y. (2012). Random forest for bioinformatics. In *Ensemble machine learning*, (pp. 307–323). Springer.

Ripley, B., Venables, W., & Ripley, M. B. (2016). Package nnet. *R package version*, (pp. 7–3).

Robinson, M. D., & Speed, T. P. (2007). A comparison of affymetrix gene expression arrays. *BMC bioinformatics*, *8*(1), 449.

Schlicker, A., Beran, G., Chresta, C. M., McWalter, G., Pritchard, A., Weston, S., Runswick, S., Davenport, S., Heathcote, K., Castro, D., Orphanides, G., French, T., & Wessels, L. F. (2012). Subtypes of primary colorectal tumors correlate with response to targeted treatment in colorectal cell lines. *BMC Medical Genomics*, *5*(1), 66.

Seidel, C. (2008). Introduction to dna microarrays. *Analysis of microarray data: a network-based approach*, *1*, 1.

Siegel, A. J., Kimberly D. Miller (2017). Cancer statistics, 2017. *CA: A Cancer Journal for Clinicians*, *67*(1), 7–30.

Simon, R., Lam, A., Li, M.-C., Ngan, M., Menenzes, S., & Zhao, Y. (2007). Analysis of gene expression data using brb-array tools. *Cancer Informatics*, *3*, 11.

Spijkers-Hagelstein, J. A., Schneider, P., Hulleman, E., de Boer, J., Williams, O., Pieters, R., & Stam, R. W. (2012). Elevated s100a8/s100a9 expression causes glucocorticoid resistance in mll-rearranged infant acute lymphoblastic leukemia. *Leukemia*, *26*(6), 1255–1265.

Stephens, D., & Diesing, M. (2014). A comparison of supervised classification methods for the prediction of substrate type using multibeam acoustic and legacy grain-size data. *PLOS One*, *9*(4), e93950.

Su, L.-J., Chang, C.-W., Wu, Y.-C., Chen, K.-C., Lin, C.-J., Liang, S.-C., Lin, C.-H., Whang-Peng, J., Hsu, S.-L., Chen, C.-H., & Huang, C.-Y. F. (2007). Selection of ddx5 as a novel internal control for q-rt-pcr from microarray data using a block bootstrap re-sampling scheme. *BMC Genomics*, *8*(1), 140.

Sun, Y., & Goodison, S. (2009). Optimizing molecular signatures for predicting prostate cancer recurrence. *The Prostate*, *69*(10), 1119–1127.

Tibshirani, R., Hastie, T., Narasimhan, B., & Chu, G. (2003). Class prediction by nearest shrunken centroids, with applications to dna microarrays. *Statistical Science*, (pp. 104–117).

Torre, L. A., Bray, F., Siegel, R. L., Ferlay, J., Lortet-Tieulent, J., & Jemal, A. (2015). Global cancer statistics, 2012. *CA: A Cancer Journal for Clinicians*, *65*(2), 87–108.

Tuimala, J., & Laine, M. M. (2003). Dna microarray data analysis. *Espoo, Finland: Finnish IT Center for Science*.

Valentini, G. (2002). Supervised gene expression data analysis using support vector machines and multi-layer perceptrons. In *Proc. of KES2002, the Sixth International Conference on Knowledge-Based Intelligent Information & Engineering Systems, special session Machine Learning in Bioinformatics*.

Vanitha, C. D. A., Devaraj, D., & Venkatesulu, M. (2015). Gene expression data classification using support vector machine and mutual information-based gene selection. *Procedia Computer Science*, *47*, 13–21.

WHO (2002). *National cancer control programmes: policies and managerial guidelines*. World Health Organization.

Wilson, D. R., & Martinez, T. R. (1997). Improved heterogeneous distance functions. *Journal of artificial intelligence research*, *6*, 1–34.

Wu, B., Abbott, T., Fishman, D., McMurray, W., Mor, G., Stone, K., Ward, D., Williams, K., & Zhao, H. (2003). Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data. *Bioinformatics*, *19*(13), 1636–1643.

Xu, L., Shen, S. S., Hoshida, Y., Subramanian, A., Ross, K., Brunet, J.-P., Wagner, S. N., Ramaswamy, S., Mesirov, J. P., & Hynes, R. O. (2008). Gene expression changes in an animal melanoma model correlate with aggressiveness of human melanoma metastases. *Molecular Cancer Research*, *6*(5), 760–769.

Yao, Z., & Ruzzo, W. L. (2006). A regression-based k nearest neighbor algorithm for gene function prediction from heterogeneous data. *BMC Bioinformatics*, *7*(1), S11.

Appendix A

In appendix A we provide the R code for all the methods.

A.1 Support Vector Machines (SVM) R Code

```
# To speedup the performance and utilize all the computer processor.
library(doParallel)
nocores = detectCores() - 1
cl = makeCluster(nocores)
registerDoParallel(cl)

#Training SVM Models
library(caret)
library(dplyr)    #(Used by caret)
library(kernlab)    #(support vector machines)
library(pROC)    #(plot the ROC curves)

set.seed(1)
#Setup for cross validation
ctrl = trainControl( method="CV",
                     number = 10,
                     classProbs=TRUE,
                     savePredictions = TRUE,
                     allowParallel = TRUE,
                     )
#Train and Tune the SVM Radial
svmRadial.tune = train( x= trainX,
                        y= trainData$CLASS,
                        method = "svmRadial",    # Radial kernel
                        #tuneLength = 5,    # 5 values of the cost function
                        metric="Accuracy",
                              trControl=ctrl
                              )
svmRadial.tune
set.seed(1)
svmRadial.pred = predict(svmRadial.tune, testData[,-which(names(testData) %in%
c("CLASS"))])
svmRadial.tab = table(pred = svmRadial.pred, true = testData[,c("CLASS")])
svmRadial.Conf = confusionMatrix(testData[,c("CLASS")],svmRadial.pred,positive
= levels(testData[,c("CLASS")])[2])
svmRadial.Conf

#Linear Kernel
set.seed(1)
#Train and Tune the SVM Linear
svmLinear.tune = train( x= trainX,
                              y= trainData$CLASS,
                              method = "svmLinear",
                              #tuneLength = 5,
                              metric="Accuracy",
                              trControl=ctrl
                              )
svmLinear.tune
svmLinear.pred = predict(svmLinear.tune, testData[,-which(names(testData) %in%
c("CLASS"))])
svmLinear.tab = table(pred = svmLinear.pred, true = testData[,c("CLASS")])
svmLinear.Conf = confusionMatrix(testData[,c("CLASS")],svmLinear.pred,positive
= levels(testData[,c("CLASS")])[2])
svmLinear.Conf

#Polynomial Kernel
set.seed(1)
#Train and Tune the SVM Polynomial
svmPoly.tune = train( x= trainX,
                           y= trainData$CLASS,
                           method = "svmPoly",
                           #tuneLength = 5,
                           metric="Accuracy",
                           trControl=ctrl
                           )
```

```
svmPoly.tune
svmPoly.pred = predict(svmPoly.tune, testData[,-which(names(testData) %in% c("CLASS"))])
svmPoly.tab = table(pred = svmPoly.pred, true = testData[,c("CLASS")])
svmPoly.Conf = confusionMatrix(testData[,c("CLASS")],svmPoly.pred,positive = lev-
els(testData[,c("CLASS")])[2])
svmPoly.Conf
```

A.2 Artificial Neural Networks (ANN) R Code

```
library(doParallel)
nocores = detectCores() - 1
cl = makeCluster(nocores)
registerDoParallel(cl)

library(caret)
library(dplyr)   # Used by caret
library(nnet)   # ANN
library(pROC)   # plot the ROC curves

set.seed(1)

# Setup for cross validation
ctrl = trainControl( method= "CV",
                     number = 10,
                     classProbs=TRUE,
                     allowParallel = TRUE,
                     savePredictions = TRUE
                     )

set.seed(1)

ANNModel.tune = train( x= trainX,
                       y= trainData$CLASS,
                       method = "nnet",   # Artificial neural networks
                       #tuneLength = 5,
                       trace = F,
                       trControl=ctrl,
                       metric="Accuracy",
                       MaxNWts =13 * (13 *(ncol(trainX) + 1) + 13 + 1)
```

```
                              )

ANNModel.tune
plot(ANNModel.tune)
set.seed(1)
ANNModel.pred = predict(ANNModel.tune, testData[,-which(names(testData) AN-
NModel.tab = table(pred = ANNModel.pred, true = testData[,c(”CLASS”)])
ANNModel.Conf = confusionMatrix(testData[,c(”CLASS”)],ANNModel.pred,positive
= levels(testData[,c(”CLASS”)])[2])
ANNModel.Conf
```

A.3 Naive Bayes (NB) R Code

```
library(doParallel)
nocores = detectCores() - 1
cl = makeCluster(nocores)
registerDoParallel(cl)

library(caret)
library(dplyr)    # Used by caret
library(pROC)    # plot the ROC curves
library(naivebayes)    # naive Bayes method

set.seed(1)

# Setup for cross validation
ctrl = trainControl( method= ”CV”,    # cross validation
                     number= 10,
                     savePredictions = TRUE,
                     classProbs=TRUE,
                     allowParallel = TRUE
                     )

set.seed(1)

NBModel.tune = train( x= trainX,
                      y= trainData$CLASS,
                      method = ”naive_bayes”,    # Naive Bayes
                      #tuneLength = 5,
                        metric="Accuracy",
                        trControl=ctrl
                        )

NBModel.tune
print(NBModel.tune)
set.seed(1)
NBModel.pred = predict(NBModel.tune, testData[,-which(names(testData) %in% c("CLASS"))])
NBModel.tab = table(pred = NBModel.pred, true = testData[,c("CLASS")])
NBModel.Conf = confusionMatrix(testData[,c("CLASS")],NBModel.pred,positive =
levels(testData[,c("CLASS")])[2])

NBModel.Conf
```

A.4 Random Forests (RF) R Code

```
library(doParallel)
nocores = detectCores() - 1
cl = makeCluster(nocores)
registerDoParallel(cl)

library(caret)
library(caTools)
library(dplyr)    # Used by caret
library(pROC)     # plot the ROC curves
library(randomForest)
library(mlbench)

set.seed(1)
# Setup for cross validation

ctrl = trainControl( method= "CV",
                     number= 10,
                     savePredictions = TRUE,
                     classProbs=TRUE,
                     allowParallel = TRUE
                     )
```

```
RFModel.tune = train( x= trainX,
                      y= trainData$CLASS,
                      method = "rf", # Random Forests
                      #tuneLength = 5,
                      metric="Accuracy",
                      trControl=ctrl
                      )

RFModel.tune
plot(RFModel.tune)
set.seed(1)
RFModel.pred = predict(RFModel.tune, testData[,-which(names(testData) %in% c("CLASS"))])
RFModel.tab = table(pred = RFModel.pred, true = testData[,c("CLASS")])
RFModel.Conf = confusionMatrix(testData[,c("CLASS")],RFModel.pred,positive =
levels(testData[,c("CLASS")])[2])

RFModel.Conf
```

A.5 *k*-nearest neighbor (KNN) R Code

```
library(doParallel)
nocores = detectCores() - 1
cl = makeCluster(nocores)
registerDoParallel(cl)

library(caret)
library(pROC)

set.seed(1)
ctrl = trainControl( method= "CV",
                     number= 10,
                     savePredictions = TRUE,
                     classProbs=TRUE,
                     allowParallel = TRUE,
                     )

KNNModel.tune = train( x= trainX,
                       y= trainData$CLASS,
                       method = "knn", # k-Nearest Neighbor
```

```
                    #tuneLength = 5,
                    metric="Accuracy",
                    trControl=ctrl
                    )

KNNModel.tune
plot(KNNModel.tune)
set.seed(1)
KNNModel.pred = predict(KNNModel.tune, testData[,-which(names(testData) %in%
c("CLASS"))])
KNNModel.tab = table(pred = KNNModel.pred, true = testData[,c("CLASS")])
KNNModel.Conf = confusionMatrix(testData[,c("CLASS")],KNNModel.pred,positive
= levels(testData[,c("CLASS")])[2])

KNNModel.Conf
```

A.6 The ROC Curves R Code

```
library(ROCR)
ROCpred = prediction( as.numeric(ANNModel.pred) , as.numeric(testData[,c("CLASS")]))
ROCperf = performance(ROCpred,"tpr","fpr")
AUC = performance(ROCpred,measure = "auc")
AUCval = AUC@y.values[[1]]
AUCval
plot(ROCperf,col="DarkGray", lwd=4, add = TRUE)
legend("right", legend = c("KNN", "RF", "NB", "ANN", "SVM (RK)", "SVM (LK)",
"SVM (PK)"), col = c("Yellow", "Green", "DarkMagenta", "DarkGray", "Blue", "Or-
ange", "Red"), lty = 1,lwd = 3,bty = "n")
```

Appendix B

Here we present additional information about the tuning process and the results of the training phase to select the optimal parameters model. Moreover, we provided the confusion matrix on the validation set of each dataset see tables (B.2 - B.126).

B.1 Training (Tuning) Results of the methods on the GSE23988

44 samples
579 predictors
2 classes: 'ERneg', 'ERpos'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 39, 39, 40, 39, 40, 40, ...
Resampling results across tuning parameters:

B.1.1 *k*-Nearest Neighbors (KNN)

Table B.1: Tuning Parameter Results of KNN for GSE23988

k	Accuracy	Kappa
5	0.93	0.8545455
7	0.93	0.8545455
$\boxed{9}$	$\boxed{0.93}$	$\boxed{0.8545455}$

The final value used for the model was k = 9.

Table B.2: Confusion Matrix of KNN on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	7	1
ERpos	2	7

B.1.2 Naive Bayes (NB)

Table B.3: Tuning Parameter Results of NB for GSE23988

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.955}$	$\boxed{0.9045455}$
TRUE	0.955	0.9045455

The final values used for the model were $laplace$ = 0, $usekernel$ = FALSE and $adjust$ = 1.

Table B.4: Confusion Matrix of NB on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	8	0
ERpos	2	7

B.1.3 Random Forests (RF)

Table B.5: Tuning Parameter Results of RF for GSE23988

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.93}$	$\boxed{0.8545455}$
34	0.93	0.8545455
578	0.93	0.8545455

The final value used for the model was $mtry$ = 2.

Table B.6: Confusion Matrix of RF on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	8	0
ERpos	2	7

B.1.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.7: Tuning Parameter Results of SVM(RK) for GSE23988

C	Accuracy	Kappa
$\boxed{0.25}$	$\boxed{0.955}$	$\boxed{0.9045455}$
0.50	0.955	0.9045455
1.00	0.955	0.9045455

The final values used for the model were $sigma$ = 0.001065516 and C = 0.25.

Table B.8: Confusion Matrix of SVM(RK) on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	7	1
ERpos	2	7

B.1.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.9: Tuning Parameter Results of SVM(LK) for GSE23988

C	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.955}$	$\boxed{0.9045455}$

Tuning parameter C was held constant at a value of 1

Table B.10: Confusion Matrix of SVM(LK) on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	8	0
ERpos	2	7

B.1.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.11: Tuning Parameter Results of SVM(PK) for GSE23988

degree	*scale*	*C*	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.001}$	$\boxed{0.25}$	$\boxed{0.955}$	$\boxed{0.9045455}$
1	0.001	0.50	0.955	0.9045455
1	0.001	1.00	0.955	0.9045455
1	0.010	0.25	0.955	0.9045455
1	0.010	0.50	0.955	0.9045455
1	0.010	1.00	0.955	0.9045455
1	0.100	0.25	0.955	0.9045455
1	0.100	0.50	0.955	0.9045455
1	0.100	1.00	0.955	0.9045455
2	0.001	0.25	0.955	0.9045455
2	0.001	0.50	0.955	0.9045455
2	0.001	1.00	0.955	0.9045455
2	0.010	0.25	0.955	0.9045455
2	0.010	0.50	0.955	0.9045455
2	0.010	1.00	0.955	0.9045455
2	0.100	0.25	0.905	0.8045455
2	0.100	0.50	0.930	0.8545455
2	0.100	1.00	0.925	0.8500000
3	0.001	0.25	0.955	0.9045455
3	0.001	0.50	0.955	0.9045455
3	0.001	1.00	0.955	0.9045455
3	0.010	0.25	0.930	0.8545455
3	0.010	0.50	0.930	0.8545455
3	0.010	1.00	0.955	0.9045455
3	0.100	0.25	0.925	0.8500000
3	0.100	0.50	0.925	0.8500000
3	0.100	1.00	0.925	0.8500000

The final values used for the model were $degree$ = 1, $scale$ = 0.001 and C = 0.25.

Table B.12: Confusion Matrix of SVM(PK) on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
REneg	7	1
ERpos	2	7

B.1.7 Neural Networks (ANN)

Table B.13: Tuning Parameter Results of ANN for GSE23988

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.560	0.1000000
1	1e-04	0.865	0.7500000
$\boxed{1}$	$\boxed{\text{1e-01}}$	$\boxed{0.955}$	$\boxed{0.9045455}$
3	0e+00	0.840	0.7000000
3	1e-04	0.880	0.7545455
3	1e-01	0.955	0.9045455
5	0e+00	0.685	0.3500000
5	1e-04	0.955	0.9045455
5	1e-01	0.955	0.9045455

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.14: Confusion Matrix of ANN on GSE23988 Validation Set

	Reference	
Prediction	ERneg	ERpos
ERneg	8	0
ERpos	2	7

B.2 Training (Tuning) Results of the methods on the GSE7670

38 samples
1450 predictors
2 classes: 'Normal', 'Tumor'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 35, 34, 34, 34, 34, 34, ...

Resampling results across tuning parameters:

B.2.1 *k*-Nearest Neighbors (KNN)

Table B.15: Tuning Parameter Results of KNN for GSE7670

k	Accuracy	Kappa
5	0.975	0.95
7	0.975	0.95
$\boxed{9}$	$\boxed{0.975}$	$\boxed{0.95}$

The final value used for the model was k = 9.

Table B.16: Confusion Matrix of KNN on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.2 Naive Bayes (NB)

Table B.17: Tuning Parameter Results of NB for GSE7670

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.95}$	$\boxed{0.9}$
TRUE	0.95	0.9

The final values used for the model were fL = 0, $usekernel$ = FALSE and $adjust$ = 1.

Table B.18: Confusion Matrix of NB on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.3 Random Forests (RF)

Table B.19: Tuning Parameter Results of RF for GSE7670

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.975}$	$\boxed{0.95}$
53	0.950	0.90
1449	0.950	0.90

The final value used for the model was $mtry$ = 2.

Table B.20: Confusion Matrix of RF on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.21: Tuning Parameter Results of SVM(RK) for GSE7670

C	Accuracy	Kappa
$\boxed{0.25}$	$\boxed{0.95}$	$\boxed{0.9}$
0.50	0.95	0.9
1.00	0.95	0.9

The final values used for the model were $sigma$ = 0.0005699463 and C = 0.25.

Table B.22: Confusion Matrix of SVM(RK) on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.23: Tuning Parameter Results of SVM(LK) for GSE7670

C	Accuracy	Kappa
1	0.975	0.95

Tuning parameter C was held constant at a value of 1

Table B.24: Confusion Matrix of SVM(LK) on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.25: Tuning Parameter Results of SVM(PK) for GSE7670

$degree$	$scale$	C	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.001}$	$\boxed{0.25}$	$\boxed{0.9750000}$	$\boxed{0.95}$
1	0.001	0.50	0.9750000	0.95
1	0.001	1.00	0.9750000	0.95
1	0.010	0.25	0.9750000	0.95
1	0.010	0.50	0.9750000	0.95
1	0.010	1.00	0.9750000	0.95
1	0.100	0.25	0.9750000	0.95
1	0.100	0.50	0.9750000	0.95
1	0.100	1.00	0.9750000	0.95
2	0.001	0.25	0.9750000	0.95
2	0.001	0.50	0.9750000	0.95
2	0.001	1.00	0.9750000	0.95
2	0.010	0.25	0.9500000	0.90
2	0.010	0.50	0.9750000	0.95
2	0.010	1.00	0.9500000	0.90
2	0.100	0.25	0.1833333	-0.63
2	0.100	0.50	0.1833333	-0.63
2	0.100	1.00	0.2166667	-0.58
3	0.001	0.25	0.9750000	0.95
3	0.001	0.50	0.9750000	0.95
3	0.001	1.00	0.9750000	0.95
3	0.010	0.25	0.9750000	0.95
3	0.010	0.50	0.9500000	0.90
3	0.010	1.00	0.9750000	0.95
3	0.100	0.25	0.9500000	0.90
3	0.100	0.50	0.9750000	0.95
3	0.100	1.00	0.9500000	0.90

The final values used for the model were $degree$ = 1, $scale$ = 0.001 and C = 0.25.

Table B.26: Confusion Matrix of SVM(PK) on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.2.7 Neural Networks (ANN)

Table B.27: Tuning Parameter Results of ANN for GSE7670

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.6083333	0.25
1	1e-04	0.6333333	0.30
$\boxed{1}$	$\boxed{\text{1e-01}}$	$\boxed{0.9750000}$	$\boxed{0.95}$
3	0e+00	0.5166667	0.10
3	1e-04	0.9500000	0.90
3	1e-01	0.9750000	0.95
5	0e+00	0.5666667	0.20
5	1e-04	0.9000000	0.80
5	1e-01	0.9750000	0.95

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.28: Confusion Matrix of ANN on GSE7670 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	8	0
Tumor	0	8

B.3 Training (Tuning) Results of the methods on the GSE10072

46 samples
3002 predictors
2 classes: 'Normal', 'Tumor'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 41, 42, 41, 40, 41, 42, ...

Resampling results across tuning parameters:

B.3.1 *k*-Nearest Neighbors (KNN)

Table B.29: Tuning Parameter Results of KNN for GSE10072

k	Accuracy	Kappa
5	1	1
7	1	1
$\boxed{9}$	$\boxed{1}$	$\boxed{1}$

The final value used for the model was $k = 9$.

Table B.30: Confusion Matrix of KNN on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.2 Naive Bayes (NB)

Table B.31: Tuning Parameter Results of NB for GSE10072

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.98}$	$\boxed{0.9615385}$
TRUE	0.98	0.9615385

The final values used for the model were $fL = 0$, $usekernel$ = FALSE and $adjust = 1$.

Table B.32: Confusion Matrix of NB on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.3 Random Forests (RF)

Table B.33: Tuning Parameter Results of RF for GSE10072

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.98}$	$\boxed{0.9615385}$
77	0.98	0.9615385
3002	0.98	0.9615385

The final value used for the model was $mtry$ = 2.

Table B.34: Confusion Matrix of on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.35: Tuning Parameter Results of SVM(RK) for GSE10072

C	Accuracy	Kappa
0.25	0.98	0.9615385
0.50	0.98	0.9615385
$\boxed{1.00}$	$\boxed{1.00}$	$\boxed{1.0000000}$

The final values used for the model were $sigma$ = 0.0002970603 and C = 1.

Table B.36: Confusion Matrix SVM(RK) on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.37: Tuning Parameter Results of SVM(LK) for GSE10072

C	Accuracy	Kappa
1	1	1

Tuning parameter C was held constant at a value of 1

Table B.38: Confusion Matrix of SVM(LK) on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.39: Tuning Parameter Results of SVM(PK) for GSE10072

$degree$	$scale$	C	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.001}$	$\boxed{0.25}$	$\boxed{1.0000000}$	$\boxed{1.0000000}$
1	0.001	0.50	1.0000000	1.0000000
1	0.001	1.00	1.0000000	1.0000000
1	0.010	0.25	1.0000000	1.0000000
1	0.010	0.50	1.0000000	1.0000000
1	0.010	1.00	1.0000000	1.0000000
1	0.100	0.25	1.0000000	1.0000000
1	0.100	0.50	1.0000000	1.0000000
1	0.100	1.00	1.0000000	1.0000000
2	0.001	0.25	1.0000000	1.0000000
2	0.001	0.50	1.0000000	1.0000000
2	0.001	1.00	1.0000000	1.0000000
2	0.010	0.25	0.9800000	0.9615385
2	0.010	0.50	0.9800000	0.9615385
2	0.010	1.00	0.9800000	0.9615385
2	0.100	0.25	0.1866667	-0.6279387
2	0.100	0.50	0.1500000	-0.7125541
2	0.100	1.00	0.1666667	-0.6792208
3	0.001	0.25	0.9800000	0.9615385
3	0.001	0.50	0.9800000	0.9615385
3	0.001	1.00	0.9800000	0.9615385
3	0.010	0.25	0.9800000	0.9615385
3	0.010	0.50	0.9800000	0.9615385
3	0.010	1.00	0.9800000	0.9615385
3	0.100	0.25	0.9800000	0.9615385
3	0.100	0.50	0.9800000	0.9615385
3	0.100	1.00	0.9800000	0.9615385

The final values used for the model were $degree$ = 1, $scale$ = 0.001 and C = 0.25.

Table B.40: Confusion Matrix of SVM(PK) on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.3.7 Neural Networks (ANN)

Table B.41: Tuning Parameter Results of ANN for GSE10072

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.46	0.0000000
1	1e-04	0.78	0.6000000
$\boxed{1}$	$\boxed{1e-01}$	$\boxed{1.00}$	$\boxed{1.0000000}$
3	0e+00	0.46	0.0000000
3	1e-04	0.93	0.8615385
3	1e-01	1.00	1.0000000
5	0e+00	0.67	0.4000000
5	1e-04	0.96	0.9230769
5	1e-01	1.00	1.0000000

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.42: Confusion Matrix of ANN on GSE10072 Validation Set

	Reference	
Prediction	Normal	Tumor
Normal	10	0
Tumor	0	10

B.4 Training (Tuning) Results of the methods on the GSE10245

41 samples
819 predictors
2 classes: 'AC', 'SCC'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 37, 37, 37, 37, 37, 36, ...

Resampling results across tuning parameters:

B.4.1 *k*-Nearest Neighbors (KNN)

Table B.43: Tuning Parameter Results of KNN for GSE10245

k	Accuracy	Kappa
5	0.93	0.8045455
7	0.93	0.8045455
$\boxed{9}$	$\boxed{0.93}$	$\boxed{0.8045455}$

The final value used for the model was $k = 9$.

Table B.44: Confusion Matrix of KNN on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	1	4

B.4.2 Naive Bayes (NB)

Table B.45: Tuning Parameter Results of NB for GSE10245

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.955}$	$\boxed{0.9045455}$
TRUE	0.955	0.9045455

The final values used for the model were $fL = 0$, $usekernel$ = FALSE and $adjust = 1$.

Table B.46: Confusion Matrix of NB on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	0	5

B.4.3 Random Forests (RF)

Table B.47: Tuning Parameter Results of RF for GSE10245

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.980}$	$\boxed{0.9545455}$
40	0.955	0.8545455
818	0.930	0.8045455

The final value used for the model was $mtry$ = 2.

Table B.48: Confusion Matrix of RF on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	0	5

B.4.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.49: Tuning Parameter Results of SVM(RK) for GSE10245

C	Accuracy	Kappa
0.25	0.955	0.9045455
$\boxed{0.50}$	$\boxed{0.980}$	$\boxed{0.9545455}$
1.00	0.980	0.9545455

The final values used for the model were $sigma$ = 0.0008614187 and C = 0.5.

Table B.50: Confusion Matrix of SVM(RK) on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	0	5

B.4.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.51: Tuning Parameter Results of SVM(LK) for GSE10245

C	Accuracy	Kappa
1	0.98	0.9545455

Tuning parameter C was held constant at a value of 1

Table B.52: Confusion Matrix of SVM(LK) on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	0	5

B.4.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.53: Tuning Parameter Results of SVM(PK) for GSE10245

$degree$	$scale$	C	Accuracy	Kappa
$\boxed{1}$	$\boxed{0.001}$	$\boxed{0.25}$	$\boxed{0.980}$	$\boxed{0.9545455}$
1	0.001	0.50	0.980	0.9545455
1	0.001	1.00	0.980	0.9545455
1	0.010	0.25	0.980	0.9545455
1	0.010	0.50	0.980	0.9545455
1	0.010	1.00	0.980	0.9545455
1	0.100	0.25	0.980	0.9545455
1	0.100	0.50	0.980	0.9545455
1	0.100	1.00	0.980	0.9545455
2	0.001	0.25	0.980	0.9545455
2	0.001	0.50	0.980	0.9545455
2	0.001	1.00	0.980	0.9545455
2	0.010	0.25	0.980	0.9545455
2	0.010	0.50	0.980	0.9545455
2	0.010	1.00	0.980	0.9545455
2	0.100	0.25	0.885	0.7000000
2	0.100	0.50	0.910	0.7500000
2	0.100	1.00	0.910	0.7500000
3	0.001	0.25	0.980	0.9545455
3	0.001	0.50	0.980	0.9545455
3	0.001	1.00	0.980	0.9545455
3	0.010	0.25	0.955	0.9045455
3	0.010	0.50	0.980	0.9545455
3	0.010	1.00	0.980	0.9545455
3	0.100	0.25	0.935	0.8500000
3	0.100	0.50	0.935	0.8500000
3	0.100	1.00	0.935	0.8500000

The final values used for the model were $degree$ = 1, $scale$ = 0.001 and C = 0.25.

Table B.54: Confusion Matrix of SVM(PK) on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	0	5

B.4.7 Neural Networks (ANN)

Table B.55: Tuning Parameter Results of ANN for GSE10245

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.7366667	0.2000000
1	1e-04	0.8016667	0.3500000
1	1e-01	0.9550000	0.9045455
3	0e+00	0.8300000	0.4545455
3	1e-04	0.9300000	0.8045455
3	1e-01	0.9550000	0.9045455
5	0e+00	0.8300000	0.4545455
$\boxed{5}$	$\boxed{\text{1e-04}}$	$\boxed{0.9800000}$	$\boxed{0.9545455}$
5	1e-01	0.9550000	0.9045455

The final values used for the model were $size$ = 5 and $decay$ = 1e-04.

Table B.56: Confusion Matrix of ANN on GSE10245 Validation Set

	Reference	
Prediction	AC	SCC
AC	12	0
SCC	1	4

B.5 Training (Tuning) Results of the methods on the GSE25136

56 samples
52 predictors
2 classes: 'NonRec', 'Rec'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 51, 50, 50, 50, 50, 50, ...

Resampling results across tuning parameters:

B.5.1 *k*-Nearest Neighbors (KNN)

Table B.57: Tuning Parameter Results of KNN for GSE25136

k	Accuracy	Kappa
5	0.9466667	0.8878788
7	0.9433333	0.8827506
$\boxed{9}$	$\boxed{0.9600000}$	$\boxed{0.9160839}$

The final value used for the model was k = 9.

Table B.58: Confusion Matrix of KNN on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	9	3
Rec	4	7

B.5.2 Naive Bayes (NB)

Table B.59: Tuning Parameter Results of NB for GSE25136

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.9466667}$	$\boxed{0.8948718}$
TRUE	0.9466667	0.8948718

The final values used for the model were fL = 0, $usekernel$ = FALSE and $adjust$ = 1.

Table B.60: Confusion Matrix of NB on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	7	5
Rec	5	6

B.5.3 Random Forests (RF)

Table B.61: Tuning Parameter Results of RF for GSE25136

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.9300000}$	$\boxed{0.8615385}$
27	0.8766667	0.7494172
52	0.8566667	0.7039627

The final value used for the model was $mtry$ = 2.

Table B.62: Confusion Matrix of RF on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	9	3
Rec	5	6

B.5.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.63: Tuning Parameter Results of SVM(RK) for GSE25136

C	Accuracy	Kappa
$\boxed{0.25}$	$\boxed{0.9266667}$	$\boxed{0.85}$
0.50	0.9266667	0.85
1.00	0.9266667	0.85

The final values used for the model were $sigma$ = 0.01168756 and C = 0.25.

Table B.64: Confusion Matrix of SVM(RK)

	Reference	
Prediction	NonRec	Rec
NonRec	8	4
Rec	5	6

B.5.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.65: Tuning Parameter Results of SVM(LK) for GSE25136

C	Accuracy	Kappa
1	0.9266667	0.8424242

Tuning parameter C was held constant at a value of 1

Table B.66: Confusion Matrix of SVM(LK) on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	7	5
Rec	4	7

B.5.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.67: Tuning Parameter Results of SVM(PK) for GSE25136

$degree$	$scale$	C	Accuracy	Kappa
1	0.001	0.25	0.7266667	0.5333333
1	0.001	0.50	0.8666667	0.7465201
1	0.001	1.00	0.9066667	0.8045455
1	0.010	0.25	0.9266667	0.8500000
1	0.010	0.50	0.9266667	0.8500000
1	0.010	1.00	0.9466667	0.8948718
$\boxed{1}$	$\boxed{0.100}$	$\boxed{0.25}$	$\boxed{0.9466667}$	$\boxed{0.8878788}$
1	0.100	0.50	0.9266667	0.8424242
1	0.100	1.00	0.9066667	0.8039627
2	0.001	0.25	0.8666667	0.7465201
2	0.001	0.50	0.9066667	0.8045455
2	0.001	1.00	0.9266667	0.8500000
2	0.010	0.25	0.9266667	0.8500000
2	0.010	0.50	0.9100000	0.8166667
2	0.010	1.00	0.9066667	0.8045455
2	0.100	0.25	0.9100000	0.8166667
2	0.100	0.50	0.8933333	0.7833333
2	0.100	1.00	0.8933333	0.7833333
3	0.001	0.25	0.9266667	0.8500000
3	0.001	0.50	0.9266667	0.8494172
3	0.001	1.00	0.9266667	0.8500000
3	0.010	0.25	0.9100000	0.8166667
3	0.010	0.50	0.9100000	0.8166667
3	0.010	1.00	0.8900000	0.7712121
3	0.100	0.25	0.9300000	0.8615385
3	0.100	0.50	0.9100000	0.8166667
3	0.100	1.00	0.9100000	0.8166667

The final values used for the model were $degree$ = 1, $scale$ = 0.1 and C = 0.25.

Table B.68: Confusion Matrix of SVM(PK) on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	8	4
Rec	4	7

B.5.7 Neural Networks (ANN)

Table B.69: Tuning Parameter Results of ANN for GSE25136

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.4933333	0.06666667
1	1e-04	0.6300000	0.32857143
$\boxed{1}$	$\boxed{\text{1e-01}}$	$\boxed{0.9100000}$	$\boxed{0.81608392}$
3	0e+00	0.6100000	0.30000000
3	1e-04	0.8866667	0.77798868
3	1e-01	0.9100000	0.81608392
5	0e+00	0.7100000	0.42857143
5	1e-04	0.9066667	0.81095571
5	1e-01	0.9100000	0.81608392

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.70: Confusion Matrix of ANN on GSE25136 Validation Set

	Reference	
Prediction	NonRec	Rec
NonRec	9	3
Rec	4	7

B.6 Training (Tuning) Results of the methods on the GSE35896

45 samples
43 predictors
2 classes: 'No', 'Yes'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 41, 41, 40, 40, 40, 41, ...

Resampling results across tuning parameters:

B.6.1 *k*-Nearest Neighbors (KNN)

Table B.71: Tuning Parameter Results of KNN for GSE35896

k	Accuracy	Kappa
5	0.915	0.8282051
$\boxed{7}$	$\boxed{0.935}$	$\boxed{0.8666667}$
9	0.915	0.8282051

The final value used for the model was k = 7.

Table B.72: Confusion Matrix of KNN on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	6	3
Yes	2	6

B.6.2 Naive Bayes (NB)

Table B.73: Tuning Parameter Results of NB for GSE35896

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.980}$	$\boxed{0.9615385}$
TRUE	0.955	0.9115385

The final values used for the model were fL = 0, $usekernel$ = FALSE and $adjust$ = 1.

Table B.74: Confusion Matrix of NB on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	8	1
Yes	1	7

B.6.3 Random Forests (RF)

Table B.75: Tuning Parameter Results of RF for GSE35896

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.960}$	$\boxed{0.9285714}$
22	0.885	0.7785714
43	0.910	0.8285714

The final value used for the model was $mtry$ = 2.

Table B.76: Confusion Matrix of RF on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	9	0
Yes	4	4

B.6.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.77: Tuning Parameter Results of SVM(RK) for GSE35896

C	Accuracy	Kappa
$\boxed{0.25}$	$\boxed{0.98}$	$\boxed{0.9615385}$
0.50	0.98	0.9615385
1.00	0.98	0.9615385

The final values used for the model were $sigma$ = 0.01592371 and C = 0.25.

Table B.78: Confusion Matrix of SVM(RK) on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	8	1
Yes	2	6

B.6.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.79: Tuning Parameter Results of SVM(LK) for GSE35896

C	Accuracy	Kappa
1	0.955	0.9115385

Tuning parameter C was held constant at a value of 1

Table B.80: Confusion Matrix of SVM(LK) on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	6	3
Yes	3	5

B.6.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.81: Tuning Parameter Results of SVM(PK) for GSE35896

degree	*scale*	*C*	Accuracy	Kappa
1	0.001	0.25	0.955	0.9115385
1	0.001	0.50	0.955	0.9115385
1	0.001	1.00	0.935	0.8785714
1	0.010	0.25	0.980	0.9615385
1	0.010	0.50	0.980	0.9615385
1	0.010	1.00	0.980	0.9615385
1	0.100	0.25	0.980	0.9615385
1	0.100	0.50	0.955	0.9115385
1	0.100	1.00	0.955	0.9115385
2	0.001	0.25	0.935	0.8785714
2	0.001	0.50	0.955	0.9115385
$\boxed{2}$	$\boxed{0.001}$	$\boxed{1.00}$	$\boxed{1.000}$	$\boxed{1.0000000}$
2	0.010	0.25	0.980	0.9615385
2	0.010	0.50	0.980	0.9615385
2	0.010	1.00	0.980	0.9615385
2	0.100	0.25	0.980	0.9615385
2	0.100	0.50	0.980	0.9615385
2	0.100	1.00	0.980	0.9615385
3	0.001	0.25	0.935	0.8785714
3	0.001	0.50	0.955	0.9115385
3	0.001	1.00	0.980	0.9615385
3	0.010	0.25	0.980	0.9615385
3	0.010	0.50	0.980	0.9615385
3	0.010	1.00	0.980	0.9615385
3	0.100	0.25	0.980	0.9615385
3	0.100	0.50	0.980	0.9615385
3	0.100	1.00	0.980	0.9615385

The final values used for the model were $degree$ = 2, $scale$ = 0.001 and C = 1.

Table B.82: Confusion Matrix of SVM(PK) on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	8	1
Yes	2	6

B.6.7 Neural Networks (ANN)

Table B.83: Tuning Parameter Results of ANN for GSE35896

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.810	0.6000000
1	1e-04	0.905	0.8115385
$\boxed{1}$	$\boxed{1e-01}$	$\boxed{0.960}$	$\boxed{0.9166667}$
3	0e+00	0.910	0.8166667
3	1e-04	0.910	0.8166667
3	1e-01	0.960	0.9166667
5	0e+00	0.960	0.9166667
5	1e-04	0.910	0.8166667
5	1e-01	0.960	0.9166667

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.84: Confusion Matrix of ANN on GSE35896 Validation Set

	Reference	
Prediction	No	Yes
No	6	3
Yes	2	6

B.7 Training (Tuning) Results of the methods on the GSE103091

76 samples
54 predictors
2 classes: 'Met', 'MetFree'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 68, 69, 67, 67, 69, 69, ...

Resampling results across tuning parameters:

B.7.1 *k*-Nearest Neighbors (KNN)

Table B.85: Tuning Parameter Results of KNN for GSE103091

k	Accuracy	Kappa
5	0.9031746	0.7187631
7	0.9031746	0.6903743
$\boxed{9}$	$\boxed{0.9031746}$	$\boxed{0.7011160}$

The final value used for the model was k = 9.

Table B.86: Confusion Matrix of KNN on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	1	8
MetFree	2	20

B.7.2 Naive Bayes (NB)

Table B.87: Tuning Parameter Results of NB for GSE103091

$usekernel$	Accuracy	Kappa
FALSE	0.8380952	0.6063250
$\boxed{\text{TRUE}}$	$\boxed{0.8666667}$	$\boxed{0.6602892}$

The final values used for the model were fL = 0, $usekernel$ = TRUE and $adjust$ = 1.

Table B.88: Confusion Matrix of NB on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	3	6
MetFree	4	18

B.7.3 Random Forests (RF)

Table B.89: Tuning Parameter Results of RF for GSE103091

mtry	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.8906746}$	$\boxed{0.6895048}$
28	0.8416667	0.5306812
54	0.8255952	0.5006812

The final value used for the model was $mtry$ = 2.

Table B.90: Confusion Matrix of RF on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	2	7
MetFree	2	20

B.7.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.91: Tuning Parameter Results of SVM(RK) for GSE103091

C	Accuracy	Kappa
0.25	0.8492063	0.6290523
0.50	0.8634921	0.6525817
$\boxed{1.00}$	$\boxed{0.8920635}$	$\boxed{0.7064278}$

The final values used for the model were $sigma$ = 0.01131729 and C = 1.

Table B.92: Confusion Matrix of SVM(RK) on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	2	7
MetFree	3	19

B.7.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.93: Tuning Parameter Results of SVM(LK) for GSE103091

C	Accuracy	Kappa
1	0.8920635	0.7144246

Tuning parameter C was held constant at a value of 1

Table B.94: Confusion Matrix of SVM(LK) on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	2	7
MetFree	4	18

B.7.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.95: Tuning Parameter Results of SVM(PK) for GSE103091

$degree$	$scale$	C	Accuracy	Kappa
1	0.001	0.25	0.8492063	0.6290523
1	0.001	0.50	0.8492063	0.6290523
1	0.001	1.00	0.8492063	0.6290523
1	0.010	0.25	0.8492063	0.6290523
1	0.010	0.50	0.8920635	0.7064278
1	0.010	1.00	0.9031746	0.7295048
1	0.100	0.25	0.8666667	0.6148770
1	0.100	0.50	0.8666667	0.6148770
1	0.100	1.00	0.9063492	0.7379540
2	0.001	0.25	0.8492063	0.6290523
2	0.001	0.50	0.8492063	0.6290523
2	0.001	1.00	0.8492063	0.6290523
2	0.010	0.25	0.9031746	0.7295048
2	0.010	0.50	0.9031746	0.7295048
2	0.010	1.00	0.8920635	0.6967775
2	0.100	0.25	0.9031746	0.7295048
2	0.100	0.50	0.9031746	0.7295048
2	0.100	1.00	0.9031746	0.7295048
3	0.001	0.25	0.8492063	0.6290523
3	0.001	0.50	0.8492063	0.6290523
3	0.001	1.00	0.8492063	0.6290523
3	0.010	0.25	0.9031746	0.7295048
3	0.010	0.50	0.9031746	0.7295048
$\boxed{3}$	$\boxed{0.010}$	$\boxed{1.00}$	$\boxed{0.9174603}$	$\boxed{0.7706812}$
3	0.100	0.25	0.9031746	0.7295048
3	0.100	0.50	0.9031746	0.7295048
3	0.100	1.00	0.9063492	0.7272123

The final values used for the model were $degree$ = 3, $scale$ = 0.01 and C = 1.

Table B.96: Confusion Matrix of SVM(PK) on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	2	7
MetFree	4	18

B.7.7 Neural Networks (ANN)

Table B.97: Tuning Parameter Results of ANN for GSE103091

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.7817460	0.2315508
1	1e-04	0.7626984	0.2053118
$\boxed{1}$	$\boxed{1e-01}$	$\boxed{0.8888889}$	$\boxed{0.7489421}$
3	0e+00	0.7706349	0.2088235
3	1e-04	0.8634921	0.6459079
3	1e-01	0.8888889	0.7489421
5	0e+00	0.8081349	0.4674035
5	1e-04	0.8085317	0.5547890
5	1e-01	0.8888889	0.7489421

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.98: Confusion Matrix of ANN on GSE103091 Validation Set

	Reference	
Prediction	Met	MetFree
Met	3	6
MetFree	4	18

B.8 Training (Tuning) Results of the methods on the GSE5851

48 samples
14 predictors
2 classes: 'No', 'Yes'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 43, 43, 44, 43, 43, 43, ...

Resampling results across tuning parameters:

B.8.1 *k*-Nearest Neighbors (KNN)

Table B.99: Tuning Parameter Results of KNN for GSE5851

k	Accuracy	Kappa
5	0.90	0.7878788
$\boxed{7}$	$\boxed{0.90}$	$\boxed{0.7872960}$
9	0.84	0.6655012

The final value used for the model was k = 7.

Table B.100: Confusion Matrix of KNN on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	10	3
Yes	3	4

B.8.2 Naive Bayes (NB)

Table B.101: Tuning Parameter Results of NB for GSE5851

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{0.895}$	$\boxed{0.7321678}$
TRUE	0.875	0.6937063

The final values used for the model were fL = 0, $usekernel$ = FALSE and $adjust$ = 1.

Table B.102: Confusion Matrix of NB on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	10	3
Yes	3	4

B.8.3 Random Forests (RF)

Table B.103: Tuning Parameter Results of RF for GSE5851

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{0.87}$	$\boxed{0.6321678}$
8	0.87	0.6321678
14	0.81	0.5027972

The final value used for the model was $mtry$ = 2.

Table B.104: Confusion Matrix of RF on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	12	1
Yes	5	2

B.8.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.105: Tuning Parameter Results of SVM(RK) for GSE5851

C	Accuracy	Kappa
0.25	0.875	0.6937063
$\boxed{0.50}$	$\boxed{0.895}$	$\boxed{0.7321678}$
1.00	0.895	0.7321678

The final values used for the model were $sigma$ = 0.05046808 and C = 0.5.

Table B.106: Confusion Matrix of SVM(RK) on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	11	2
Yes	4	3

B.8.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.107: Tuning Parameter Results of SVM(LK) for GSE5851

C	Accuracy	Kappa
1	0.87	0.6994172

Tuning parameter C was held constant at a value of 1

Table B.108: Confusion Matrix of SVM(LK) on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	9	4
Yes	5	2

B.8.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.109: Tuning Parameter Results of SVM(PK) for GSE5851

$degree$	$scale$	C	Accuracy	Kappa
1	0.001	0.25	0.850	0.6603730
1	0.001	0.50	0.850	0.6603730
1	0.001	1.00	0.875	0.6937063
1	0.010	0.25	0.850	0.6603730
1	0.010	0.50	0.850	0.6603730
1	0.010	1.00	0.875	0.6937063
1	0.100	0.25	0.875	0.6937063
$\boxed{1}$	$\boxed{0.100}$	$\boxed{0.50}$	$\boxed{0.895}$	$\boxed{0.7321678}$
1	0.100	1.00	0.895	0.7321678
2	0.001	0.25	0.850	0.6603730
2	0.001	0.50	0.850	0.6603730
2	0.001	1.00	0.875	0.6937063
2	0.010	0.25	0.875	0.6937063
2	0.010	0.50	0.875	0.6937063
2	0.010	1.00	0.875	0.6937063
2	0.100	0.25	0.895	0.7321678
2	0.100	0.50	0.895	0.7321678
2	0.100	1.00	0.855	0.6488345
3	0.001	0.25	0.875	0.6937063
3	0.001	0.50	0.875	0.6937063
3	0.001	1.00	0.850	0.6603730
3	0.010	0.25	0.875	0.6937063
3	0.010	0.50	0.875	0.6937063
3	0.010	1.00	0.875	0.6937063
3	0.100	0.25	0.875	0.6867133
3	0.100	0.50	0.835	0.6167832
3	0.100	1.00	0.835	0.6167832

The final values used for the model were $degree$ = 1, $scale$ = 0.1 and C = 0.5.

Table B.110: Confusion Matrix of SVM(PK) on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	10	3
Yes	4	3

B.8.7 Neural Networks (ANN)

Table B.111: Tuning Parameter Results of ANN for GSE5851

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.670	0.1000000
1	1e-04	0.715	0.2712121
1	1e-01	0.890	0.7372960
3	0e+00	0.810	0.5206294
3	1e-04	0.875	0.7372960
3	1e-01	0.890	0.7372960
5	0e+00	0.750	0.3757576
$\boxed{5}$	$\boxed{\text{1e-04}}$	$\boxed{0.915}$	$\boxed{0.8206294}$
5	1e-01	0.890	0.7372960

The final values used for the model were $size$ = 5 and $decay$ = 1e-04.

Table B.112: Confusion Matrix of ANN on GSE5851 Validation Set

	Reference	
Prediction	No	Yes
No	10	3
Yes	5	2

B.9 Training (Tuning) Results of the methods on the GSE32962

31 samples
117 predictors
2 classes: 'Resist', 'Sens'

Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 28, 28, 27, 27, 28, 29, ...

Resampling results across tuning parameters:

B.9.1 *k*-Nearest Neighbors (KNN)

Table B.113: Tuning Parameter Results of KNN for GSE32962

k	Accuracy	Kappa
5	0.9166667	0.84
7	0.9166667	0.84
$\boxed{9}$	$\boxed{0.9500000}$	$\boxed{0.90}$

The final value used for the model was $k = 9$.

Table B.114: Confusion Matrix of KNN on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	5	2
Sens	2	3

B.9.2 Naive Bayes (NB)

Table B.115: Tuning Parameter Results of NB for GSE32962

$usekernel$	Accuracy	Kappa
$\boxed{\text{FALSE}}$	$\boxed{1}$	$\boxed{1}$
TRUE	1	1

The final values used for the model were $fL = 0$, $usekernel$ = FALSE and $adjust = 1$.

Table B.116: Confusion Matrix of NB on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	4	3
Sens	1	4

B.9.3 Random Forests (RF)

Table B.117: Tuning Parameter Results of RF for GSE32962

$mtry$	Accuracy	Kappa
$\boxed{2}$	$\boxed{1.0000000}$	$\boxed{1.00}$
59	0.9416667	0.89
117	0.9083333	0.83

The final value used for the model was $mtry$ = 2.

Table B.118: Confusion Matrix of RF on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	4	3
Sens	2	3

B.9.4 Support Vector Machines with Radial Basis Function Kernel (SVM(RK))

Table B.119: Tuning Parameter Results of SVM(RK) for GSE32962

C	Accuracy	Kappa
$\boxed{0.25}$	$\boxed{1}$	$\boxed{1}$
0.50	1	1
1.00	1	1

The final values used for the model were $sigma$ = 0.005723221 and C = 0.25.

Table B.120: Confusion Matrix of SVM(RK) on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	3	4
Sens	1	4

B.9.5 Support Vector Machines with Linear Kernel (SVM(LK))

Table B.121: Tuning Parameter Results of SVM(LK) for GSE32962

C	Accuracy	Kappa
1	1	1

Tuning parameter C was held constant at a value of 1.

Table B.122: Confusion Matrix of SVM(LK) on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	4	3
Sens	1	4

B.9.6 Support Vector Machines with Polynomial Kernel (SVM(PK))

Table B.123: Tuning Parameter Results of SVM(PK) for GSE32962

$degree$	$scale$	C	Accuracy	Kappa
1	0.001	0.25	0.9083333	0.83
1	0.001	0.50	0.8750000	0.77
1	0.001	1.00	1.0000000	1.00
$\boxed{1}$	$\boxed{0.010}$	$\boxed{0.25}$	$\boxed{1.0000000}$	$\boxed{1.00}$
1	0.010	0.50	1.0000000	1.00
1	0.010	1.00	1.0000000	1.00
1	0.100	0.25	1.0000000	1.00
1	0.100	0.50	1.0000000	1.00
1	0.100	1.00	1.0000000	1.00
2	0.001	0.25	0.9083333	0.83
2	0.001	0.50	1.0000000	1.00
2	0.001	1.00	1.0000000	1.00
2	0.010	0.25	1.0000000	1.00
2	0.010	0.50	1.0000000	1.00
2	0.010	1.00	1.0000000	1.00
2	0.100	0.25	1.0000000	1.00
2	0.100	0.50	1.0000000	1.00
2	0.100	1.00	1.0000000	1.00
3	0.001	0.25	0.9750000	0.95
3	0.001	0.50	1.0000000	1.00
3	0.001	1.00	1.0000000	1.00
3	0.010	0.25	1.0000000	1.00
3	0.010	0.50	1.0000000	1.00
3	0.010	1.00	1.0000000	1.00
3	0.100	0.25	0.9166667	0.84
3	0.100	0.50	0.9166667	0.84
3	0.100	1.00	0.9166667	0.84

The final values used for the model were $degree$ = 1, $scale$ = 0.01 and C = 0.25.

Table B.124: Confusion Matrix of SVM(PK) on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	5	2
Sens	2	3

B.9.7 Neural Networks (ANN)

Table B.125: Tuning Parameter Results of ANN for GSE32962

$size$	$decay$	Accuracy	Kappa
1	0e+00	0.8166667	0.60
1	1e-04	0.8500000	0.60
$\boxed{1}$	$\boxed{\text{1e-01}}$	$\boxed{0.9500000}$	$\boxed{0.90}$
3	0e+00	0.9500000	0.90
3	1e-04	0.9166667	0.84
3	1e-01	0.9500000	0.90
5	0e+00	0.9500000	0.90
5	1e-04	0.9500000	0.90
5	1e-01	0.9500000	0.90

The final values used for the model were $size$ = 1 and $decay$ = 0.1.

Table B.126: Confusion Matrix of ANN on GSE32962 Validation Set

	Reference	
Prediction	Resist	Sens
Resist	5	2
Sens	2	3

Appendix C

Here we provide the links of the datasets used in this study.

https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE23988
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE7670
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE8401
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE10072
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE10245
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE25136
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE35896
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE103091
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE5851
https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE32962