UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Hadi Mir

MACHINE LEARNING-BASED MOTION TYPE CLASSIFICATION FROM 5G DATA

Master's Thesis
Degree Programme in Computer Science and Engineering
June 2022

Mir H. (2022) Machine Learning-Based Motion Type Classification From 5G Data. University of Oulu, Degree Programme in Computer Science and Engineering. Master’s Thesis, 66p.

ABSTRACT

To improve the quality of their services and products, nowadays every industry is using artificial intelligence and machine learning. Machine learning is a powerful tool that can be applied in many applications including wireless communications. One way to improve the reliability of wireless connections is to classify motion type of the user and hook it with beamforming and beam steering. With the user equipment’s motion type classification ability, the base station can allocate proper beamforming to the given class of users. With this motivation, the studies of ML algorithms for motion classification is conducted in this thesis. In this work, the supervised learning technique is used to predict and classify motion types using the 5G data. In this work, we used the 5G data collected in 4 different scenarios or classes which are (i) Walking (ii) Standing (ii) Driving and (iv) Drone. The data is then operated on for cleaning and feature engineering and then is fed into different classification algorithms including Logistic Regression Cross Validation (LRCv), Support Vector Classifier (SVC), k-nearest neighbors (KNN), Linear Discriminant Analysis (LDA), AdaBoost, and Extra Tree Classifier. Upon analyzing the evaluation metrics for these algorithms, we found that with the accuracy of ~99% and log-loss of 0.044, Extra Tree Classifier performed better than others. With such promising results, the output of classification process can be used in another pipeline for resource optimization or hooked with hardware for beamforming and beam steering. It can also be used as an input to a digital twin of radio to change its variables dynamically which will be reflected in the physical copy of that radio.

Keywords: 5G data, artificial intelligence, wireless communication

TABLE OF CONTENTS

ABSTRACT
TABLE OF CONTENTS
FOREWORD
LIST OF ABBREVIATIONS AND SYMBOLS
1. INTRODUCTION 7
    1.1. Background and Motivation 7
    1.2. Thesis Scope and Contribution 10
    1.3. Structure of Thesis 10
2. LITERATURE REVIEW 11
    2.1. Localization of User Equipments 11
    2.2. User Equipment Tracking 11
    2.3. Motion and Activity Recognition 11
3. 5G DATA COLLECTION AND DESCRIPTION 13
    3.1. Data Collection 14
    3.2. Data Description 14
    3.3. Data Visualization 16
    3.4. Exploratory Data Analysis and Feature Engineering 27
    3.5. One-Hot Encoding 31
    3.6. EDA Conclusion 31
4. MACHINE LEARNING APPROACHES 32
    4.1. Supervised Learning 32
    4.2. Logistic Regression 33
    4.3. Support Vector Machine 34
    4.4. KNN-Classifier 35
    4.5. Adaptive Boosting 36
    4.6. Linear Discriminant Analysis 36
    4.7. Extra Tree Classifier 37
    4.8. Evaluation Metrics 38
5. IMPLEMENTATION 41
    5.1. Language and Packages 41
    5.2. Pandas 41
    5.3. Matplotlib and Seaborn 41
    5.4. Pandas Profiling 41
    5.5. Scikit-learn 42
    5.6. Ipyleaflet 42
    5.7. ML-Flow 42
    5.8. Integration Tools 42
6. RESULTS 44
    6.1. Logistic regression CV 44
    6.2. SVC 45
    6.3. KNN 46
    6.4. AdaBoost 47
    6.5. LDA 48
    6.6. Extra Tree Classifier 49
    6.7. Pipeline Overview 53
7. DISCUSSION 54

7.1. Future work 54
8. CONCLUSION 55
9. REFERENCES 56
10. APPENDICES 60

FOREWORD

This work has been carried out for Nokia Solutions Oy in Oulu Finland. I want to present my gratitude to Jami Järviö, Ilkka Kansala, and Eero Heikkinen for providing me with an opportunity to carry out this work. I also want to thank my supervisor from Nokia, D.Sc. Tachporn Sanguanpuak for guiding me all along. I also want to thank Johannes Jyrkka and Risto Martikkala for helping me with 5G data collection. I also want to thank my main supervisor from the university Prof. Juha Röning and supervisor Assist. Prof. Jaakko Suutala for their guidance. Finally, I want to thank God for providing me strength, ideas, and motivation to finishing my thesis on time.

Oulu, June 20, 2022

Hadi Mir

LIST OF ABBREVIATIONS AND SYMBOLS

2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
6G	Sixth Generation
AI	Artificial Intelligence
API	Application programming interface
BDMA	Beam Division Multiple Access
BI	Beam index
BTS	Base transceiver station
CDMA	Code-Division Multiple Access
CNN	Convolutional Neural Networks
EDA	Exploratory data analysis
GPS	Global positioning system
GSM	Global System for Mobile communication
HTML	Hyper Text Markup Language
KNN	K nearest neighbors
LDA	Linear discriminant analysis
LTE	Long Term Evolution
mmWave	Millimeter wave
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning
MMS	Multimedia Messaging Service
NR-PCI	New radio physical cell ID
NR-ARFCN	New radio absolute radio-frequency channel number
OFDM	Orthogonal frequency-division multiplexing
PCI	Physical cell ID
RSS	Received signal strength
RSPR	Reference Signal Received Power
SINR	Signal-to-Interference-plus-Noise Ratio
SMS	Short message service
SVM	Support vector machine
TDMA	Time Division Multiple Access
UE	User equipment
UI	User interface
UMTS	Universal Mobile Telecommunications System
WCDMA	Wideband Code Division Multiple Access
WiMAX	Worldwide Interoperability for Microwave Access

φ_k	Correlation coefficient
δ_ϕ	Difference between 2 latitude coordinates
ϕ_1	First latitude
ϕ_2	Second latitude
λ_ϕ	Difference between two longitude coordinates
R	Radius of earth

1. INTRODUCTION

Wireless communication has evolved rapidly over the last decade. We have seen incredible increase in reliability and data transfer speed. In this new era of communication, people are already using 5G (Fifth Generation) networks. Although the research has already started for development of 6G (Sixth Generation) communication there are still some challenges in 5G that need to be addressed to make it more robust. At the same time integration of artificial intelligence (AI) and machine learning (ML) in products and services is becoming more and more common these days as it makes the system more efficient and reliable.

In this thesis, we implement a ML pipeline along with 5G data to determine the motion type of user equipment (UE). The outcome or the predictions can be then used in beam forming and beam steering, thus improving the quality and reliability of the 5G networks.

1.1. Background and Motivation

Every major technology in the world is turning to be smart. In the last two decades, we have seen a revolution in every domain of technology and recently the boom of artificial intelligence, and machine learning has changed how we used to think about the term "smart" in technological domain [1]. Artificial intelligence witnessed two major ‘AI winters’ from 1974–1980 and 1987–1993 in which AI saw reduced funding and interest. Back then the computational power of computers was far less than the power of our smartphones which was major cause of AI winter. But in this decade AI has seen rapid growth [2]. The field of AI was not formally founded until 1956, In a conference at Dartmouth College, in Hanover, New Hampshire, the term "artificial intelligence" was coined [3]. As time passed computational power increased and computers became fast and cheap, and researchers used the opportunity to develop AI systems and algorithms. These algorithms performed surprisingly well and, in several cases, even beat humans.

Nowadays, every major industry is applying AI to their product which not only results in increasing in revenue but also it takes user experiences to next level. Machine learning, which is a sub-field of AI, is used to carry out multiple tasks and it covers the statistical part of AI where the computer solves a problem by looking at hundreds and thousands of data points. Machine learning has lot of different algorithms to do variety of tasks. The choice of algorithm depends on the end goal of our task. Machine learning can be divided into distinct categories based on the type of input and output data, and type of problem they solve. Some primary categories of machine learning algorithms are supervised, unsupervised and reinforcement learning. In supervised learning the algorithm maps the input to the output and the data always has labels. Furthermore, in supervised learning we can do regression in which we can predict the future continuous value based on the past data also in supervised learning we can do classification, in which algorithm predicts the discrete class to which class the data belongs. On the other hand, unsupervised learning can be used to find the underlying pattern in data which does not have any specified output. Thus, unsupervised algorithms can effectively be used in “clustering” data into groups. In reinforcement learning, an agent is trained in an environment with rewards and penalties and the agent tries to maximize the reward and minimize the penalties. This approach can be applied to gaming, i.e., an AI agent can learn how to play the game. These are only some basic and most widely used ML categories, however there are several other approaches for ML algorithms. [4]

The recent advances in machine learning is to apply AI technology to wireless communications. The growth of diverse services and data generated by mobile devices and internet of things (IoT) are increasing exponentially. In 2020 alone total data generation reached 64.2 zettabytes [5]. To transport and consume this data we need good and reliable connection, that is one of the reasons we continuously evolve and upgrade our cellular infrastructure. We all heard about the buzz word 5G. The term 5G stands for "fifth generation". The wireless industry adopted the standards for 5G in 2017. Since then, all the major cellphone industries have been developing the infrastructure and the products that can support 5G. 5G offers significantly more bandwidth for more devices and allows much faster upload and download speeds than 4G. In the past people used 2G, 3G and 4G networks. Each generation in some way has improved over its predecessor [6]. 5G not only offers high internet speed it also ensures better coverage and lower power consumption [7]. Although 5G is a great technology with lots of pros, it does have some drawbacks and challenges e.g., 5G has low penetration power, building and trees can block the signal. Also, the range of 5G signal is far less than the 4G signal. Table 1 shows network comparison [9], where several types of technology from 2G-5G are briefly explained.

Table 1. Comparison of different network generations

Comparison	**2G**	**3G**	**4G**	**5G**
Introduction Year	1993	2001	2009	2018
Technology	GSM	WCDMA	LTE, WiMAX	MIMO, mmWave
Access system	TDMA, CDMA	CDMA	CDMA	OFDM, BDMA
Switching type	Circuit switching for voice, packet switching for data	Packet switching except for air interface	Packet switching	Packet switching
Internet service	Narrowband	Broadband	Ultra-Broadband	Wireless World Wide Web
Bandwidth	25 MHz	25 MHz	100 MHz	30 GHz – 300 GHz
Advantage	Multimedia features SMS, MMS. Internet Access, and SIM introduced	High Security, International roaming	Speed, High speed handoffs, Global mobility	High speed, low latency
Applications	Voice calls, short messages	Video conferencing, mobile TV, GPS	High speed applications, mobile TV, devices	High resolution video streaming, remote control of vehicles, robots

Even though we have some challenges in 5G networks, we do have several techniques to counter those disadvantages, one of which is 5G beamforming. An antenna has a radiating element which radiates electro-magnetic waves in a particular direction, if we place multiple radiating elements right next to each other (still having only one antenna), we increase the energy that goes forward and less energy going sideways, or in any other direction and thus the wave travels farther. Figure 1 depicts the beam forming, where all the radiating elements are fed the same signal. But another problem that arises here is that the beam is directed straight, and it cannot cover the users who are not in front of this beam and cannot reach the user who are in a different location. The solution to that problem is beamforming. Figure 2 illustrates the beamforming technique. In beamforming we use the same number of radiating elements, but we feed them with different signals thus creating constructive interference in the direction of interest thus sending beam towards the user [8].

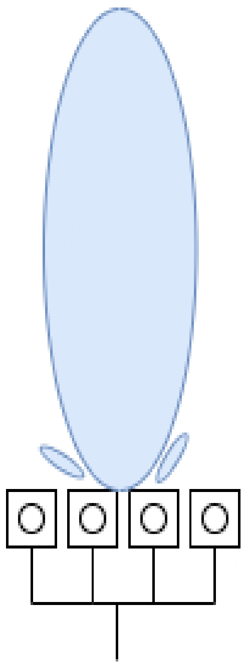


Figure 1. Straight network beam.

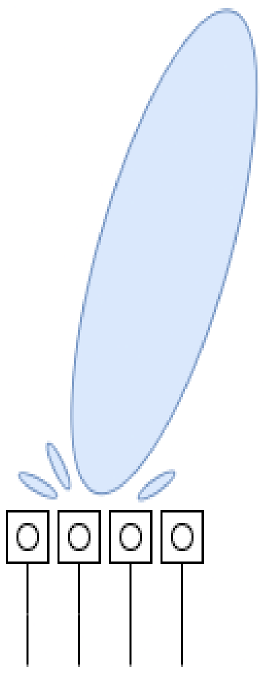


Figure 2. Shifted network beam.

Another solution is beam steering using which we can alter the beam pattern dynamically by changing the signal phase [9]. MmWave is one of the technologies that enable the 5G [10] and researchers have already been developing technology that enables beam steering in these mmWave antennas [11], [12]. Now the interesting question arises, what if we can join two technologies i.e., AI and beamforming/beam steering? How can AI and ML help in effective and uninterrupted communication? How can it help in delivering promising connectivity and reduce the hardware cost? This introduces the need for intelligent processing to categorize the motion type of the UE into distinct categories for effective optimization of the beamforming resources.

1.2. Thesis Scope and Contribution

In this thesis, we explore how we can use ML along with 5G data to classify the motion type of UEs into distinct categories, i.e., walking, standing, driving or a drone. This work builds and sets the basic way for further research in machine learning for effective and uninterrupted connectivity. The work can be used to complement the beamforming and beam-steering for delivering robust network. Our work can also be used as an input to the digital twin of a radio which in turn will change its variables and the changes will be reflected in a physical copy of the same radio [13]. This work is a part of a bigger pipeline together which will be used to improve network connectivity for the users. The research lays basic building blocks for future researchers and paves the way for further development on top of it. Moreover, the findings in this work can be further used in other research to improve 5G and 6G wireless connectivity.

1.3. Structure of Thesis

ML is an effective tool that can be particularly useful in wireless communication [14]. In this work, we use ML along with 5G data (collected in various scenarios) for motion type classification of UEs.

The thesis is divided into eight chapters where in the first chapter, we discuss background and motivation of this work and define the scope of the thesis. In the second chapter, we provide literature review and discuss the proposed methodologies with prior art. In the third chapter, we discuss more about how the data was collected and will focus on exploratory data analysis (EDA) [15] of data. In the fourth chapter, we describe machine learning approaches that were used in this work. In the next chapter, we discuss how we implemented the entire system and what challenges we faced. Then, we illustrate and evaluate the results in chapter six. In the seventh chapter, we further discuss the results and talk about future work and the last chapter provides the conclusions of this work. It is also important to highlight any challenges, limitations, and improvements in the research and throughout all the chapters it will be done whenever possible.

2. LITERATURE REVIEW

There is already some prior research done in this field. In the following sections we go through the different ways researchers have utilized to classify the motion type of UEs.

2.1. Localization of User Equipments

A lot of researchers have worked to combine machine learning with wireless communication. In [16] authors have tried to estimate the speed of UEs with low computational requirements. They have used spectral analysis method and time-based spectrum spreading method although they were successful in classifying the UE with high accuracy. However, they have not used ML in their work.

In [17], the authors developed novel framework where multi-antenna network used “channel charting” in order to learn chart of radio geometry of its neighborhood. Working in an unsupervised manner, channel chart detects the local spatial geometry so that points that are close in space are also close in the channel chart and vice versa.

In [18], the authors worked with deep convolutional neural networks for fingerprint-based positioning using measured massive multi-input multi-output (MIMO) channels. In appropriate domains, the sparse structure of massive MIMO channels can be learned by convolutional neural network (CNN) for positioning purposes. In this work, the authors concluded that if enough training data is available for CNN training, a moderately deep CNN can have fractional-wavelength positioning accuracy.

In [19] the authors have proposed a novel ML approach of using only one base station to determine the positions of mobile targets. Prior to [19], to determine the position, multi-lateration of at least three base stations was needed. Locating the position of mobile phones or UEs has become one of the most key features of the next generation mobile communication systems [20].

2.2. User Equipment Tracking

In the context of ML enable beamforming optimization, the authors in [21] proposed ML based beam-forming design. They use a deep neural network structure and input channel vector with transmit power and output the combining factors for the transmitters’ beamforming. As compared to brute force search their method had a sum rate of more than 99%. The power and application of AI can be seen in [22], where the authors introduced RF-Pose3D, a system that infers 3D human skeletons from RF signals. With the help of CNN, it tracks the person as they move. There has been lot of work in this domain and some algorithms even localize within ten of centimeter [23].

2.3. Motion and Activity Recognition

In [24] the authors proposed a new innovative approach for activity recognition which makes use of both supervised and unsupervised learning. They used the information from received signal strength (RSS) and classified user state if it is static or moving. In this work, the authors used K-means algorithm for differentiation between static and mobile states.

Moreover, different sliding window lengths were tested, and it was found out that sliding window of length of 1 second without overlapping performs best. Compared with other supervised learning approaches, the decision tree classifier performs best for both the user static and mobile cases with an accuracy of 100% and 98.0%, respectively.

Nevertheless, the researchers worked on finding and classifying user motion type using mobile networks, they also used sensors available in smartphones to classify and detect the user activity. The authors of [25] used data collected from tri-axial accelerometer and gyroscope of smartphone and after extracting statistical features of the data, then the authors used K-nearest neighbor [26] and Naive Bayes [27] algorithms and achieved accuracy of 90.1%.

In [28], the authors used the same data used in [25] and single layered feedforward neural network long with long short-term memory network. In this work, the authors were able to classify with the accuracy of 97.7%.

With all the existing prior art, it is evident that along with the traditional communication mobile networks, ML can be used in a variety of ways to improve network performance. The promising results shown by ML have encouraged increased research done in this field. With good hardware, ML can analyze data much more quickly than humans and provide solutions to complex problems. Previous researchers have worked on various aspects and applications of combining machine learning with mobile networks however, none of them have used 5G data to classify the motion type. In our work, we used ML algorithms and trained them using 5G data which was collected under different scenarios including walking, driving, standing still and data from the drone.

3. 5G DATA COLLECTION AND DESCRIPTION

Data in ML is as important as oil in a car. It is the hidden insight in the data that ML algorithm figure out and produces output based on their learning. Each year we produce more data than the previous year and this number goes up with each passing year.

The data needs to be in correct format so that algorithms can understand it. If we supply data in the wrong format, we will get wrong results. In ML the rule is garbage in, garbage out, i.e., the quality of predictions will only be as good as the data which the model is trained on, so we must make sure that the data we are using is of excellent quality as well as the data that we are using is representative otherwise the model will not be able to perform well. We want our algorithm to learn from the signal and not from the noise, that is why we need to remove the noise from the signal such as outliers and non-existing values. To establish a good ML pipeline, we need to follow some specific steps and it all starts with data collection and usually the data we get is unprocessed or raw. We show the hierarchy of AI/ML implementation in Figure 3.

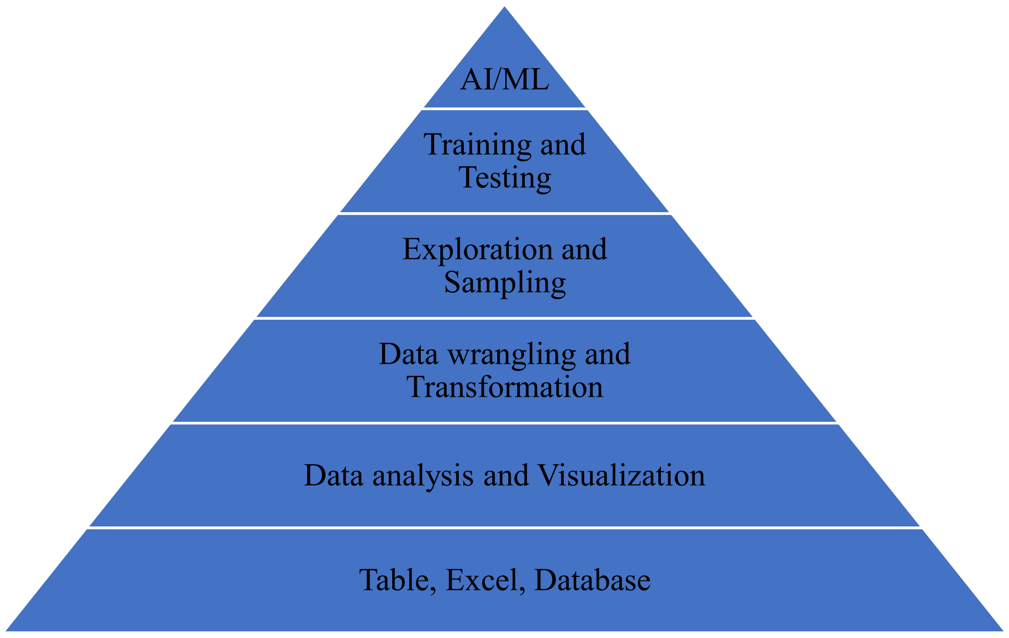


Figure 3. Hierarchy of AI/ML implementation.

We first define what we want to predict and then we collect the data which will help us to make good predictions. There are some basic steps for data processing that we followed. It included data collection, data analysis and treatment, data exploration and transformation and model training.

3.1. Data Collection

In our work, we want to classify the motion type of UEs using the 5G data, thus, for the first step we started with data collection for our algorithms. We used a software tool called Keysight Nemo Outdoor[1]. Nemo Outdoor is a laptop-based drive test tool for wireless network testing. The data was collected in different scenarios such as while driving in a van, while walking on street, while standing still in one place and we also had data from drone. The location of all these collections was around only one single 5G BTS and the data was collected for several days. The collected data contained a lot of variables in encrypted form. We used another tool called Keysight Nemo Analyze[2] to decrypt and select the variables we need, later we use these variables as features and input them to our algorithm. The labeling of data was done manually as there was no option in the collection software to do so.

3.2. Data Description

Once the data was decrypted, we took the help of data visualization and explanatory data analysis (EDA), to select set of variables for our classification and the rest were discarded. Since there are irrelevant variables, we needed to reduce them by reducing the number of features. Therefore, the system computational cost is also reduced. Table 2 lists the variables that were in the collected data.

Table 2. List of all collected variables

No	Collected Variables
1	First-best-reference-signal-received-power (1st-best-RSRP)
2	Second-best-reference-signal-received-power (2nd-best-RSRP)
3	Third-best-reference-signal-received-power (3rd-best-RSRP)
4	Fourth-best-reference-signal-received-power (4th-best-RSRP)
5	First-best-signal-to-interference-plus-noise-ratio (1st-best-SINR)
6	Second-best-signal-to-interference-plus-noise-ratio (2nd-best-SINR)
7	Third-best-signal-to-interference-plus-noise-ratio (3rd-best-SINR)
8	Fourth-best-signal-to-interference-plus-noise-ratio (4th-best-SINR)
9	New-radio-absolute-radio-frequency-channel-number (NR-ARFCN)
10	Physical-cell-ID (PCI)
11	New-radio-physical-cell-ID-beam-index (NR-PCI-beam-index)
12	Beam-index (BI)
13	Time
14	Longitude
15	Latitude
16	Height

[1] https://www.keysight.com/fi/en/product/NTA00002B/nemo-outdoor-5g-nr-drive-test-solution.html
[2] https://www.keysight.com/fi/en/product/NTN00000B/nemo-analyze-drive-test-post-processing-solution.html

After collecting all these features, we only select the following features for our classification task. Table 3 lists the variables or features that were used to train our ML model. Note that in Table 3 feature engineered variables are also included.

Table 3. List of selected variables

No	Selected Variables
1	First-best-reference-signal-received-power (1st-best-RSRP)
2	Best-signal-to-interference-plus-noise-ratio (1st-best-SINR)
3	New-radio-absolute-radio-frequency-channel-number (NR-ARFCN)
4	Physical-cell-ID (PCI)
5	New-radio-physical-cell-ID-beam-index (NR-PCI-beam-index)
6	Beam-index (BI)
7	Distance
8	Velocity
9	Height

Among all the variables present in the collected data we choose only these because they have a strong relationship with the target, i.e., the motion type of UEs. The final variables or features include, Reference Signal Received Power (RSRP), it is the form signal strength indicator which defines the power present in a received radio signal. Signal having RSPR value greater or equal to -80 dBm is considered good and signal with RSPR value less than -100 dBm is considered extremely poor connection or no connection at all. Keysight Nemo Outdoor tool can measure and collect multiple RSPR values, but we are only interested in the first best one. Signal-to-Interference-plus-Noise Ratio (SINR) is used to measure the quality of wireless connection. The value greater than 20 dBm is considered good and less than or equal to zero is considered disconnected. While we were collecting data the software was automatically adding the time on which sample was taken, we used this feature later in feature engineering for calculating velocity. Absolute radio-frequency channel number or NR-ARFCN is a code that specifies a pair of reference frequencies used for transmission and reception in radio system. ARFCN started from GSM and evolved with modern technologies. For UMTS/WCDMA it was known as UARFCN, for E-UTAR/LTE it was named EARFCN and now for 5G/New Radio it is called NR-ARFCN. PCI stands for Physical Cell ID, each 5G NR cell corresponds to a PCI and it is used to distinguish cells on the radio side. In 5G New Radio, there are 1008 unique PCIs compare to LTE which only has 504 PCIs. BI tells which beam index is the strongest, and beam index is the identification separating beams from each other. All the features have numeric data type except NR-PCI-beam-index which is categorical.

We collect all the data and use Keysight Nemo Analyze tool to decrypt the file generated by Keysight Nemo Outdoor tool. The data was extracted and stored in Excel file and is then manually labeled in Excel. We labeled data such as numerical 1 represented ‘Driving’, 2 represented ‘Standing still’, 3 represented ‘Walking’ and 4 represented ‘Drones’. Although we used any other number to represent them it would not have made any difference to output, we just had to make sure all the classes are represented by a unique value.

3.3. Data Visualization

After collecting, decrypting, and labeling the data, the first step is to do analysis by visualization. In data visualization, we use charts and plots to represent the information of datasets. Data visualization helps in spotting trends, outlier, and patterns in the data. Moreover, in the big data world, visualization could be challenging, however, we can visualize a noticeably big amount of data using a suitable set of tools [29]. We visualize the data to analyze each data set before doing feature engineer. In this thesis, we used Tableau[3], a data visualization tool for our basic visualizations.

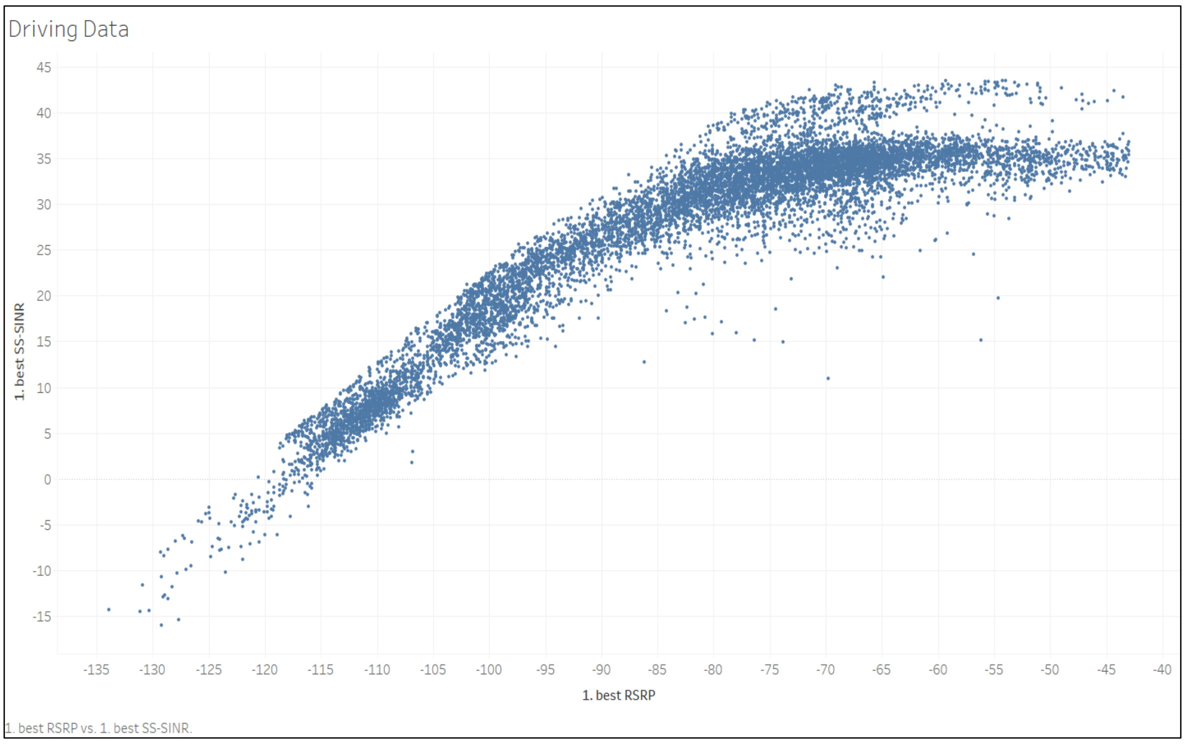


Figure 4. 1st-best-SINR versus 1st-best-RSRP (dBm) for driving data.

[3] https://www.tableau.com/

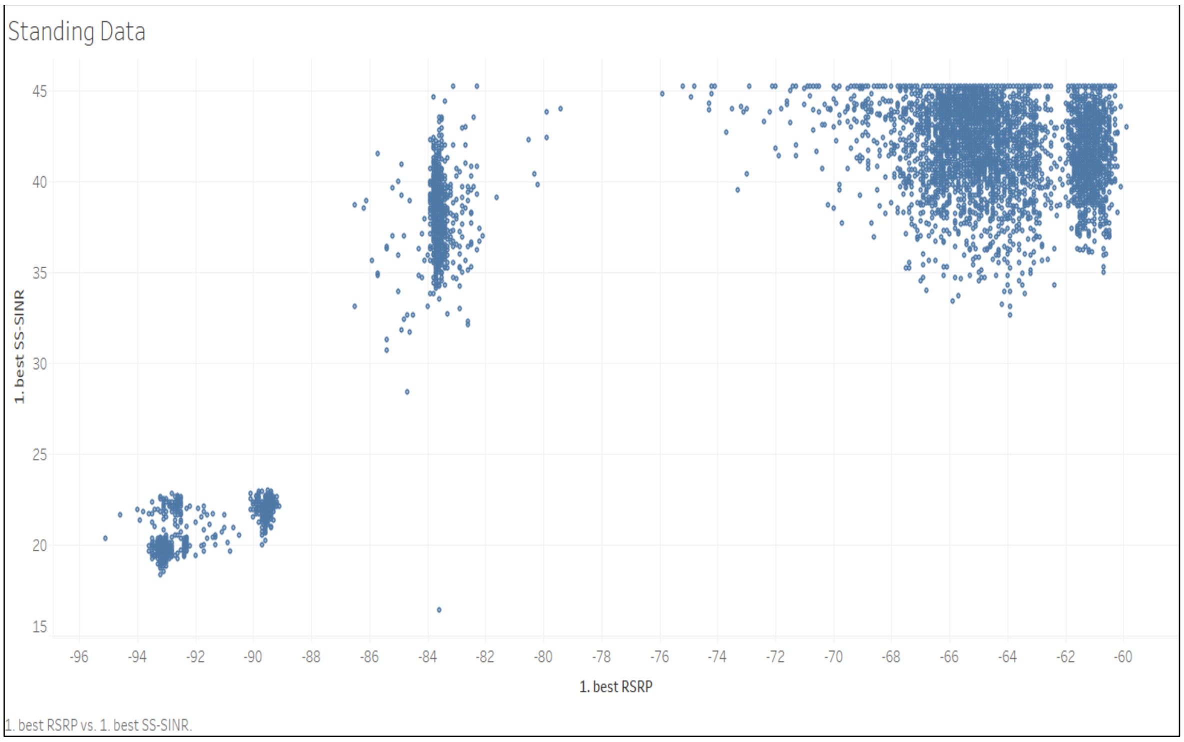


Figure 5. 1st-best-SINR versus 1st-best-RSRP (dBm) for standing data.

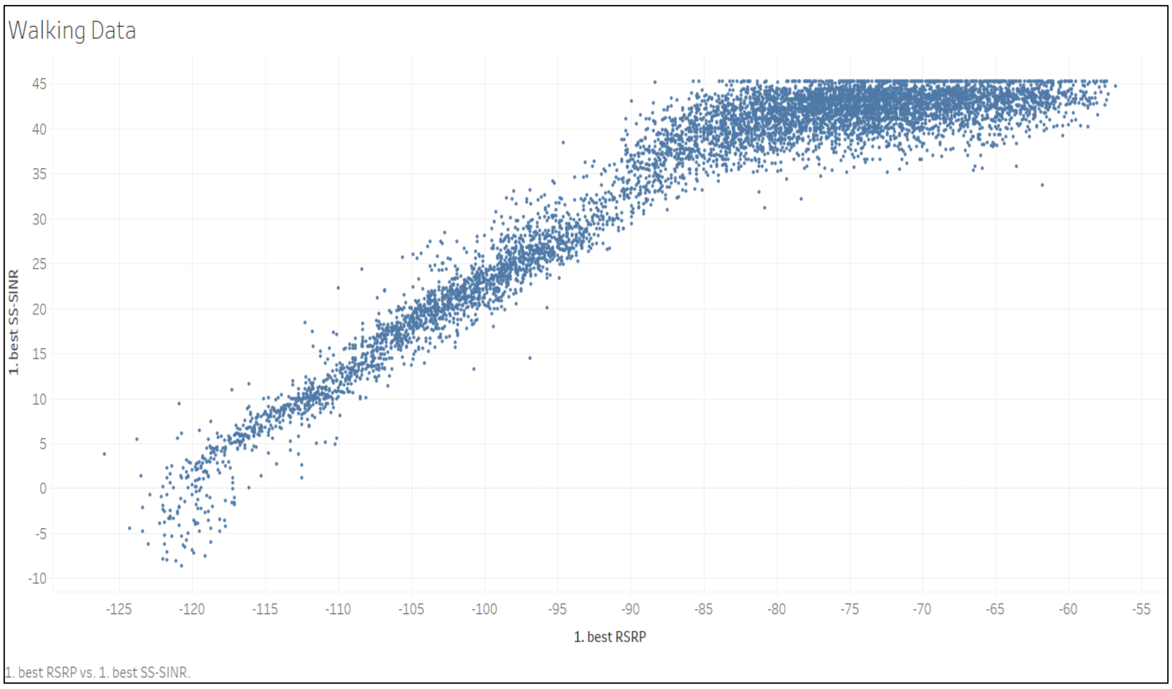


Figure 6. 1st-best-SINR versus 1st-best-RSRP (dBm) for walking data.

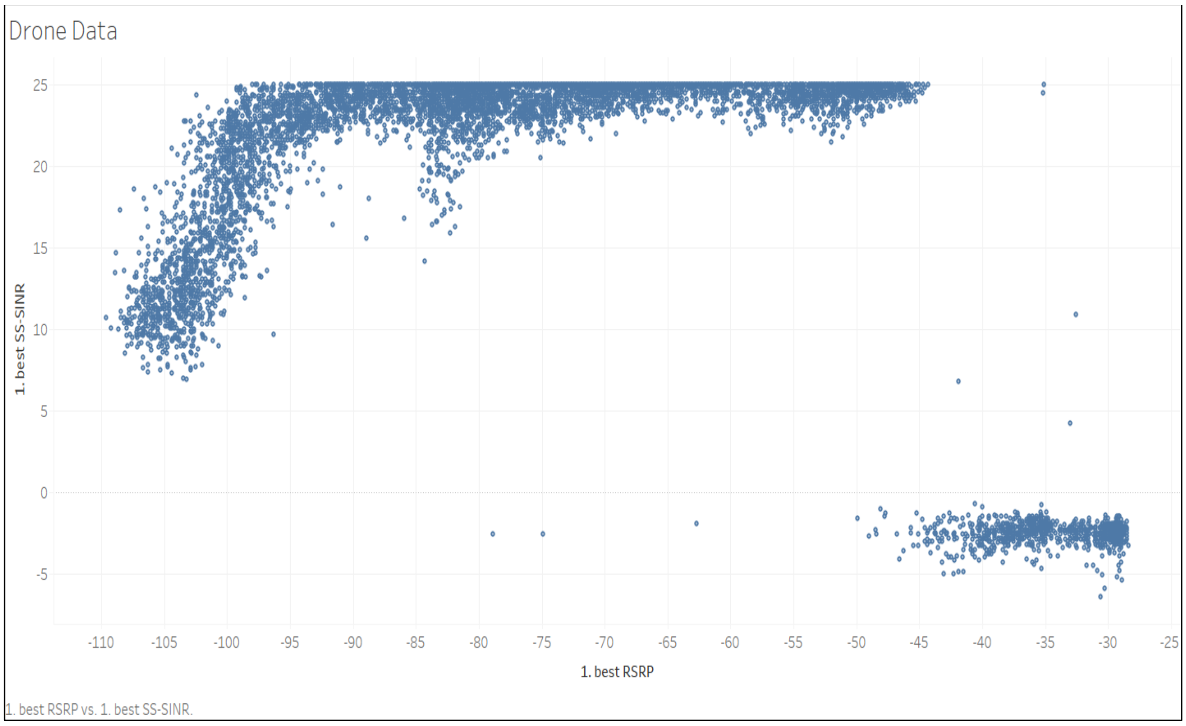


Figure 7. 1st-best-SINR versus 1st-best-RSRP (dBm) for drone data.

In Figure 4 to Figure 7 we plotted the 1st-best-SINR with 1st-best-RSRP for all the four classes. Through our visualizations we can observe that there is almost linear relationship between 1st-best-SINR and 1st-best-RSRP when we are driving and walking and in case of standing still the values remain clustered with some deviation. In our standing still graph, we have multiple clusters because the data was collected on various locations and at separate times.

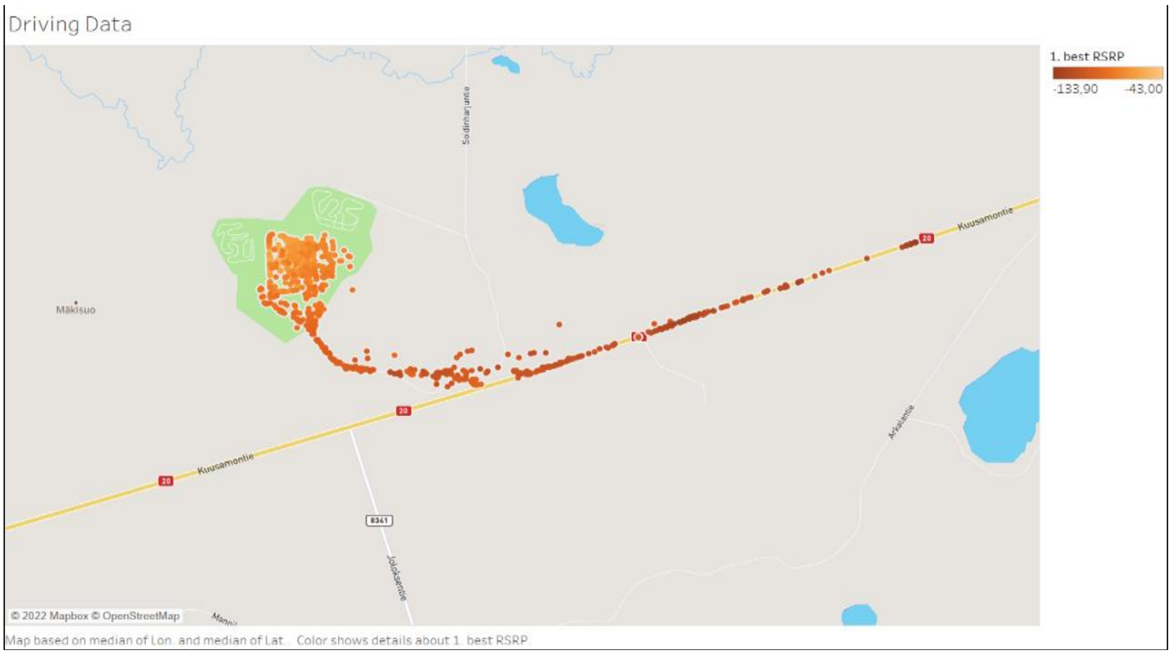

Driving Data
1. best RSRP
-133,90
-43,00
Sodinharjuntie
Kuusamontie
Mäkisuo
20
8341
Jokikantie
Arkalantie
© 2022 Mapbox © OpenStreetMap
Map based on median of Lon. and median of Lat.. Color shows details about 1. best RSRP.

Figure 8. 1st-best-RSRP location on map for driving data.

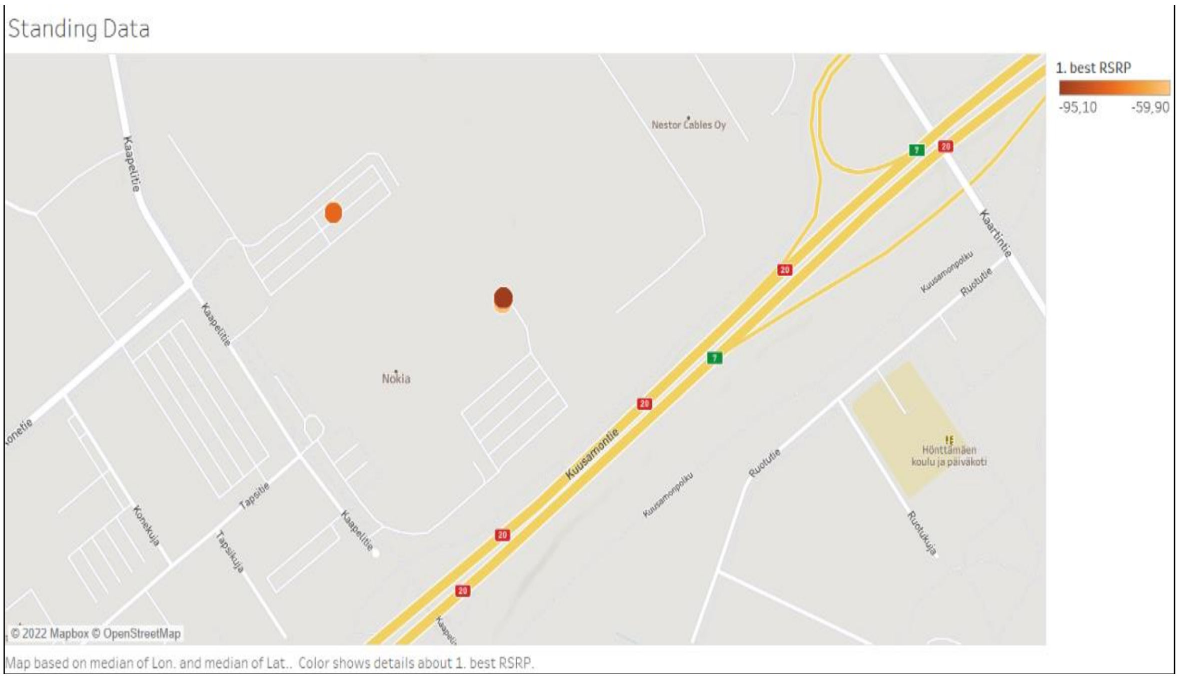


Figure 9. 1st-best-RSRP location on map for standing data.

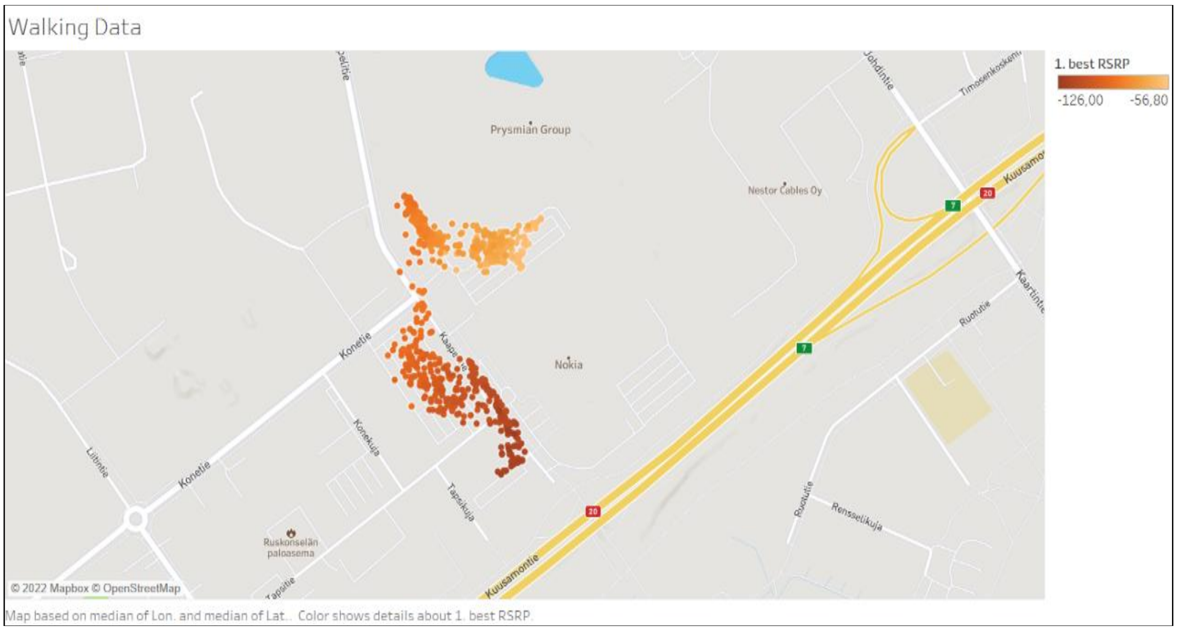


Figure 10. 1st-best-RSRP location on map for walking data.

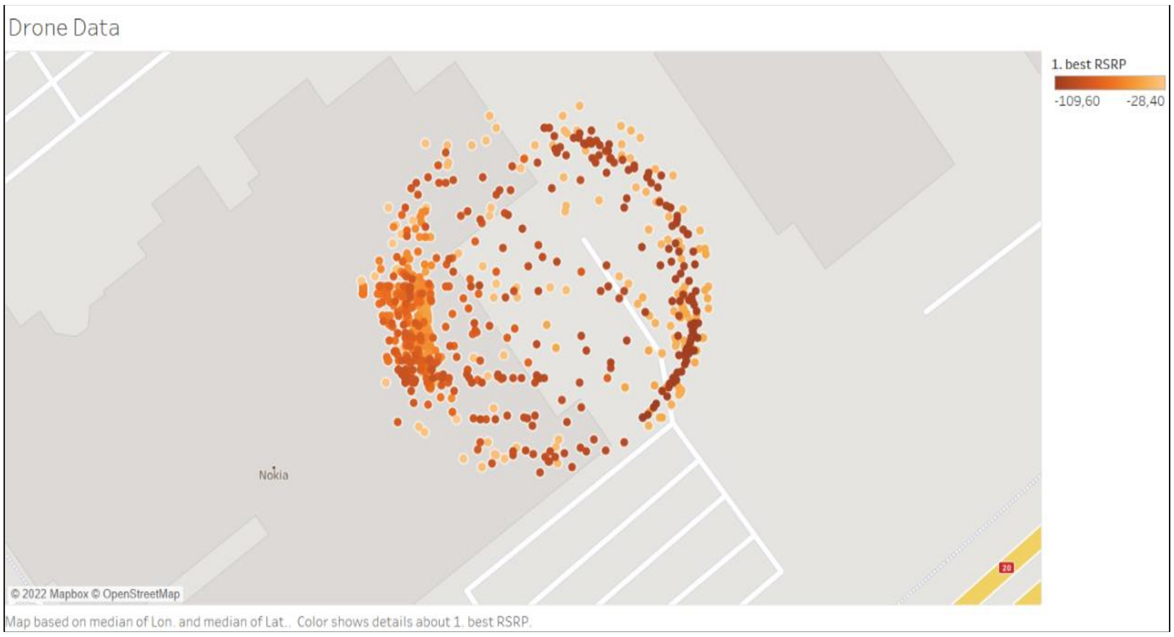

Drone Data
1. best RSRP
-109,60
-28,40
Nokia
© 2022 Mapbox © OpenStreetMap
Map based on median of Lon. and median of Lat.. Color shows details about 1. best RSRP.

Figure 11. 1st-best-RSRP location on map for drone data.

In Figure 8 to Figure 11 we plotted the 1st-best-RSRP values on the map, and it helped us to visualize how the value is changing along the path. It also displays the path we took while collecting the data. In the case of drones, we can observe that the drone revolved around the 5G tower.

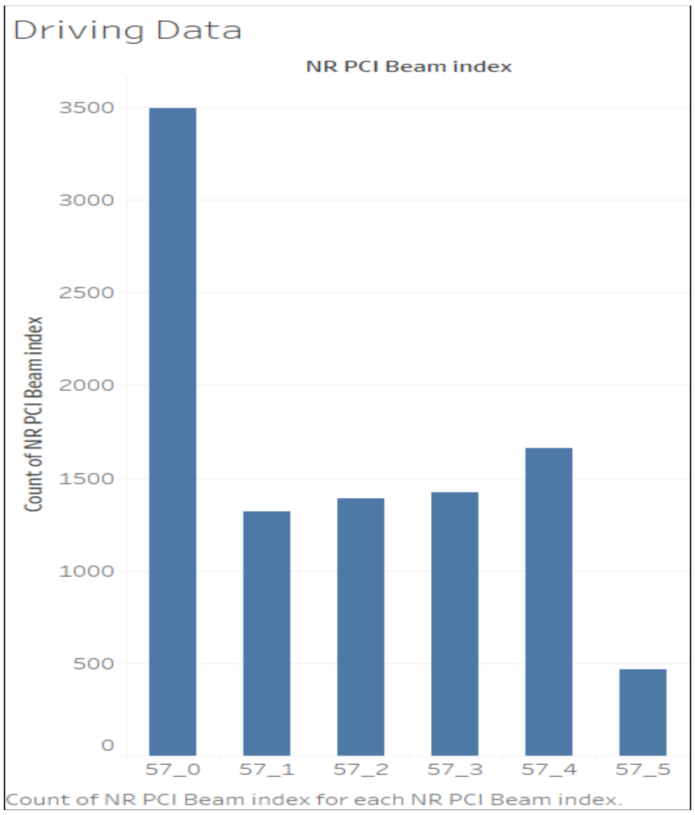


Figure 12. NR-PCI-beam index count for driving data.

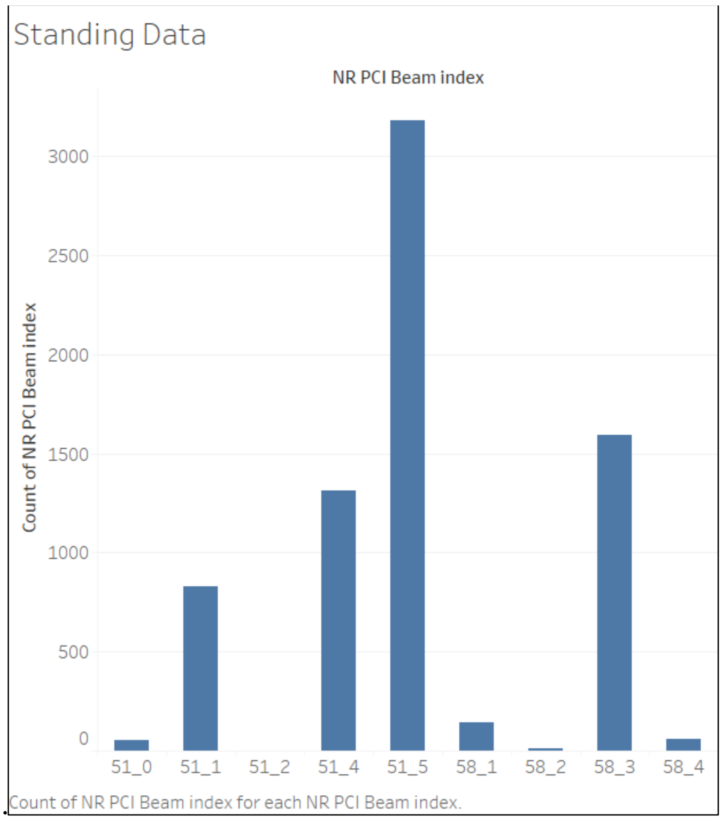


Figure 13. NR-PCI-beam index count for standing data.

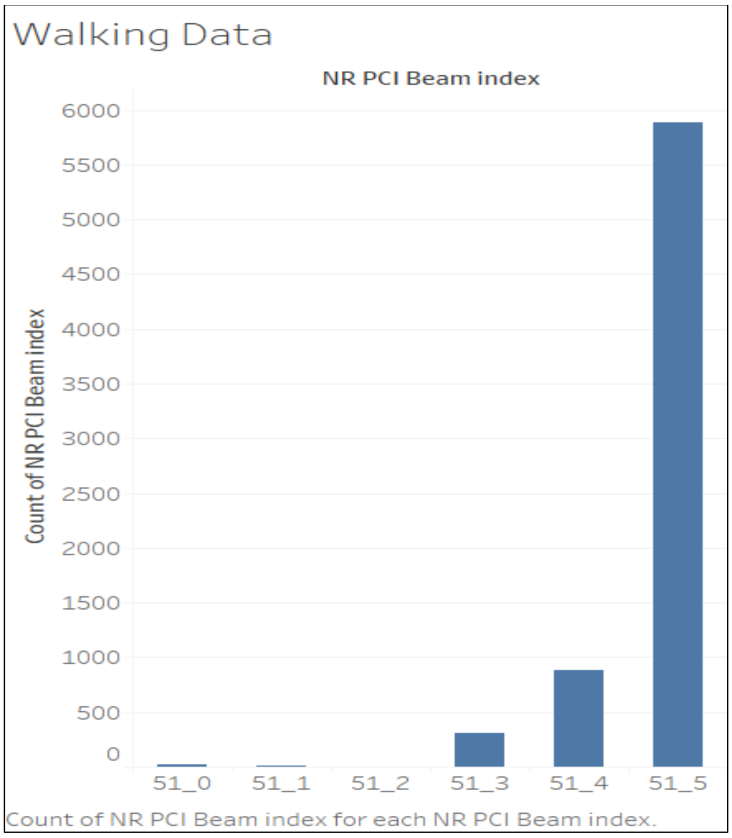


Figure 14. NR-PCI-beam index count for walking data.

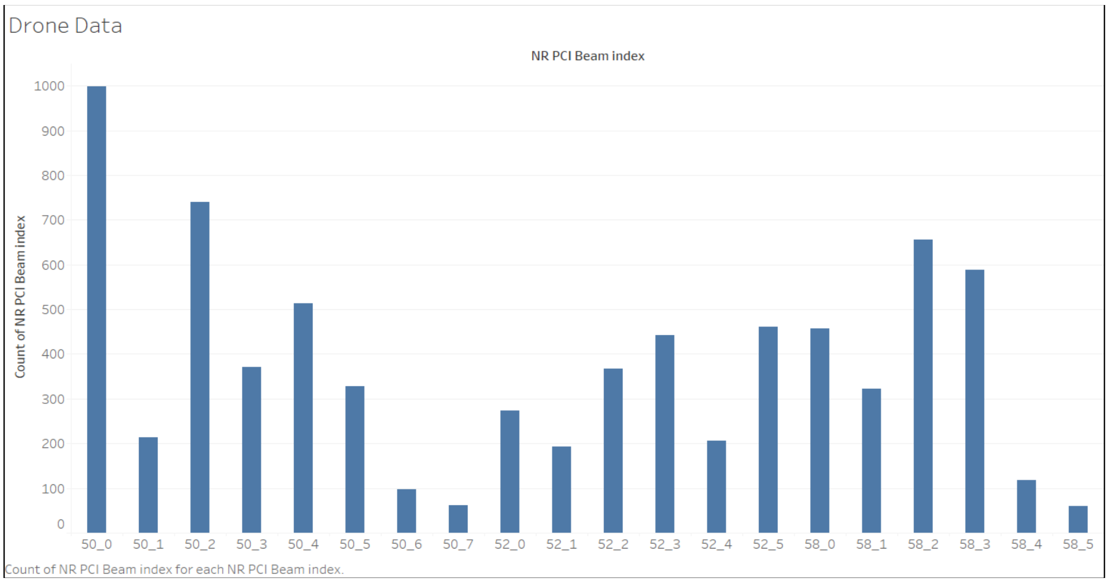


Figure 15. NR-PCI-beam index count for drone data.

From Figure 12 to Figure 15, we tried to visualize NR-PCI-Beam-Index in our data NR-PCI-Beam-Index has a categorical value. We plotted it and checked how the categories are distributed and, in our data, we can observe that walking data has the lowest beam index categories and drone has the maximum.

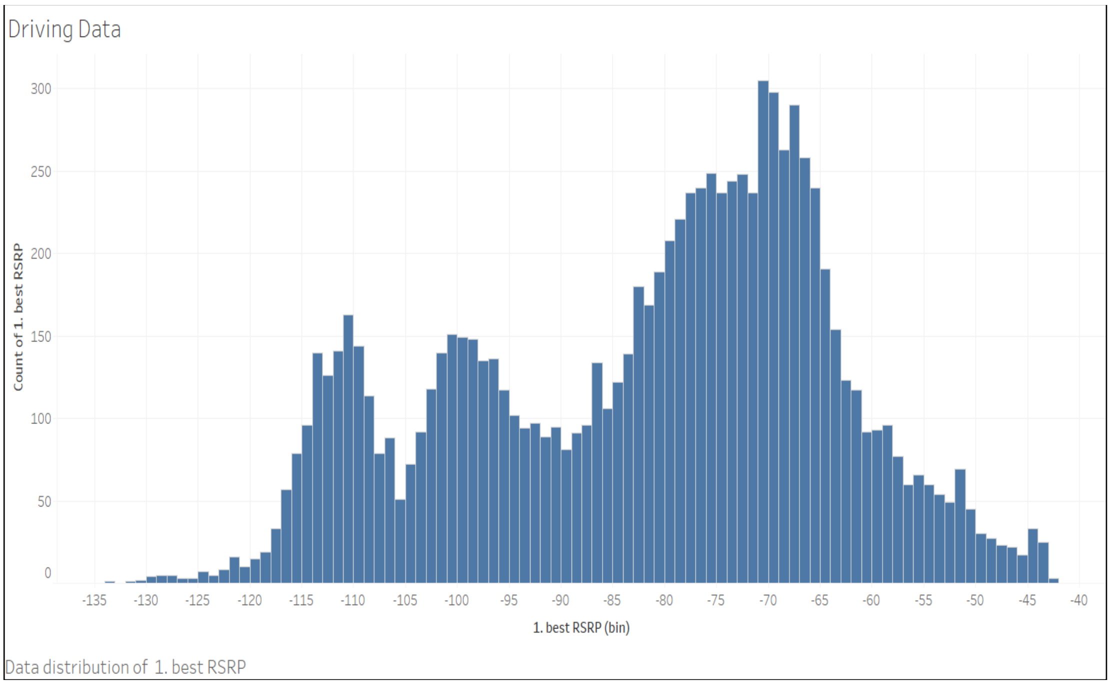


Figure 16. 1st-best-RSRP data distribution for driving data.

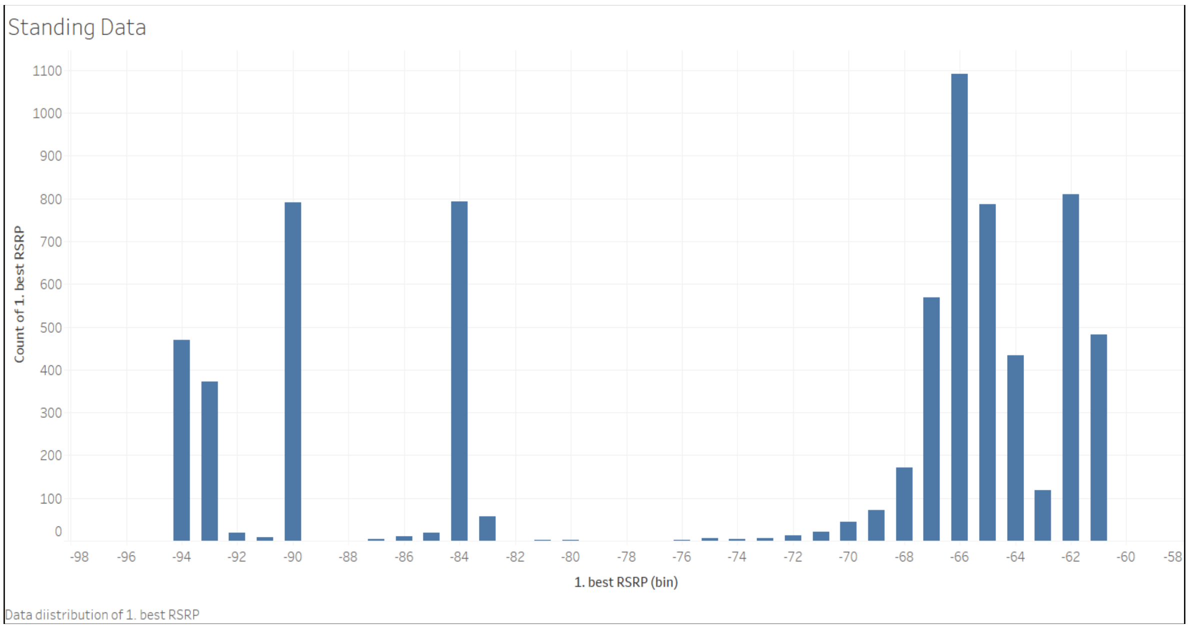


Figure 17. 1st-best-RSRP data distribution for standing data.

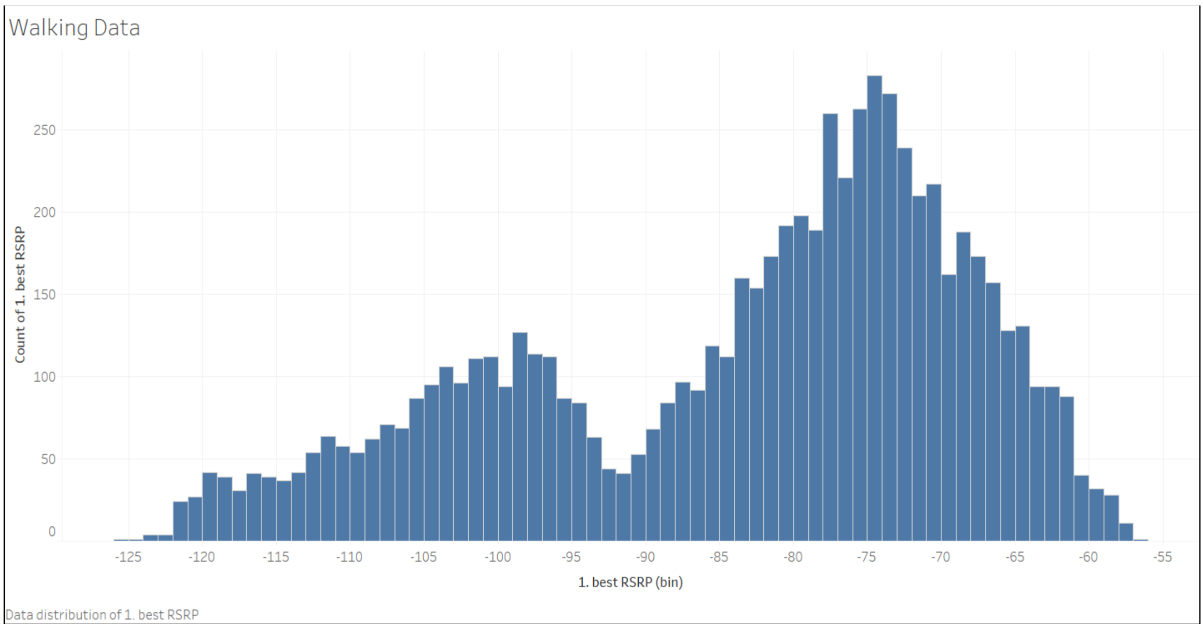


Figure 18. 1st-best-RSRP data distribution for walking data.

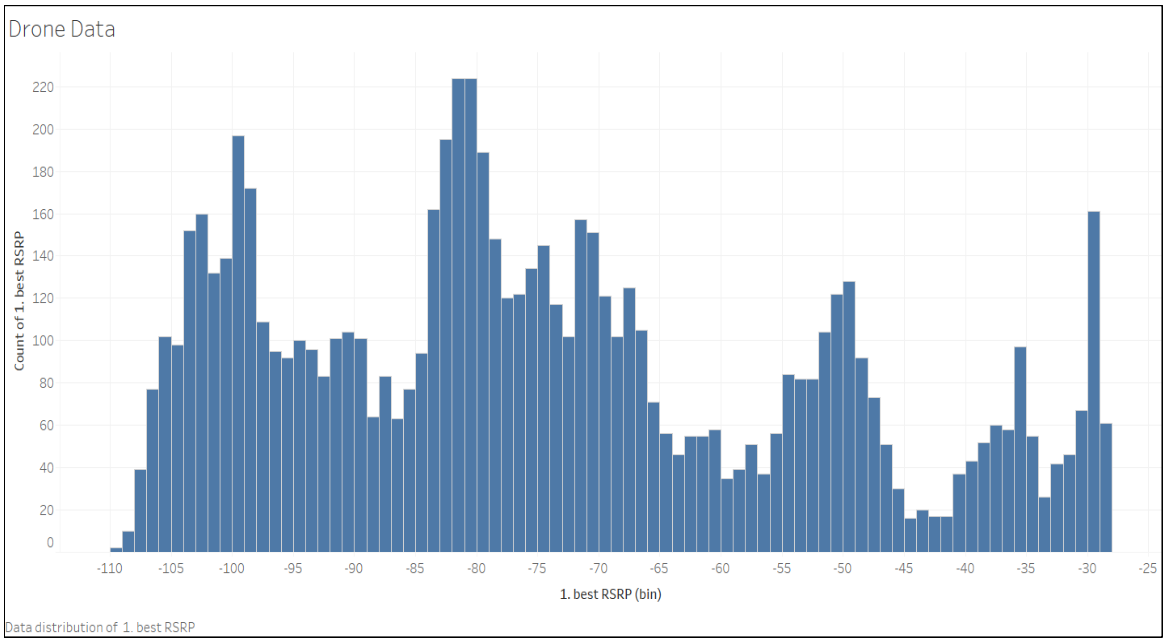


Figure 19. 1st-best-RSRP data distribution for drone data.

From Figure 16 to Figure 19 we plot the data distribution of 1st-best-RSRP. Through these plots we can check how the data varies. Through this observation we can check the peak value and skewness of the data.

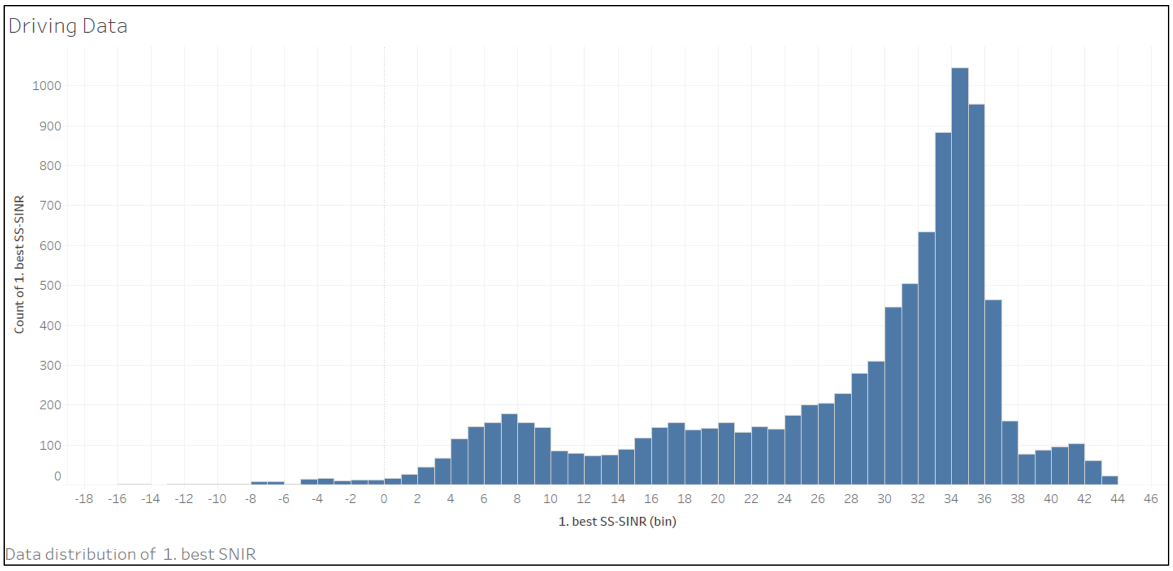


Figure 20. 1st-best-SINR data distribution for driving data.

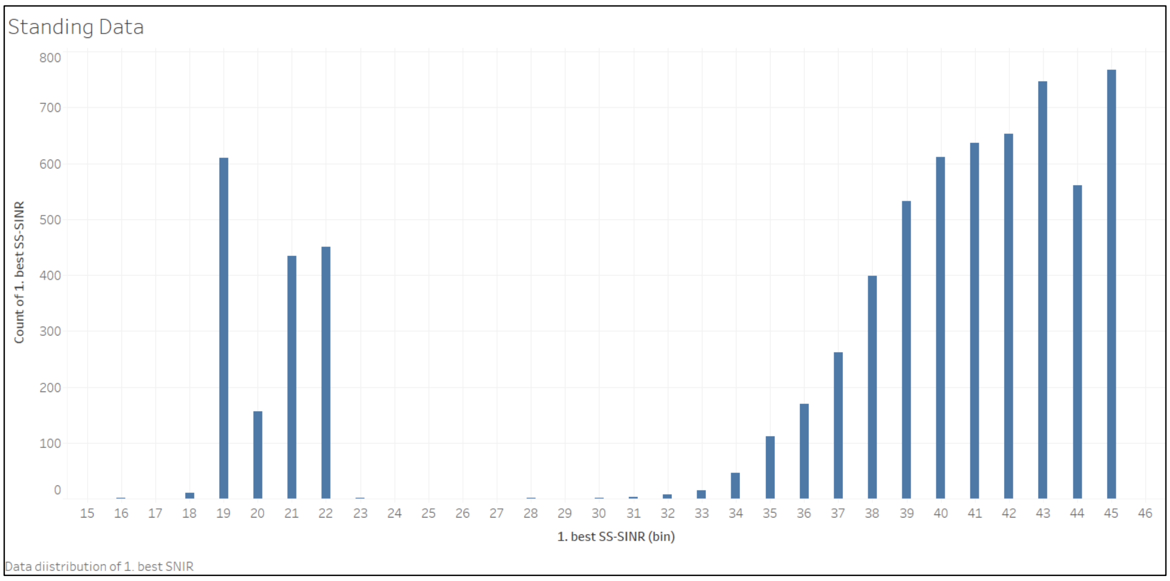


Figure 21.1st-best-SINR data distribution for standing data.

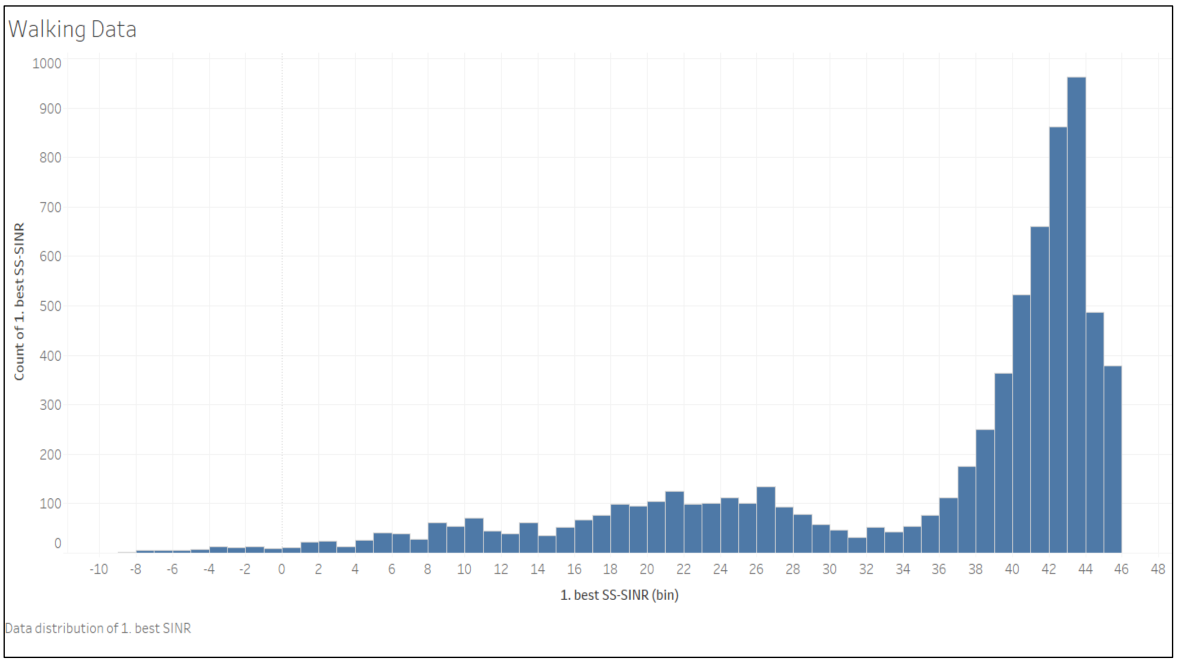


Figure 22. 1st-best-SINR data distribution for walking data.

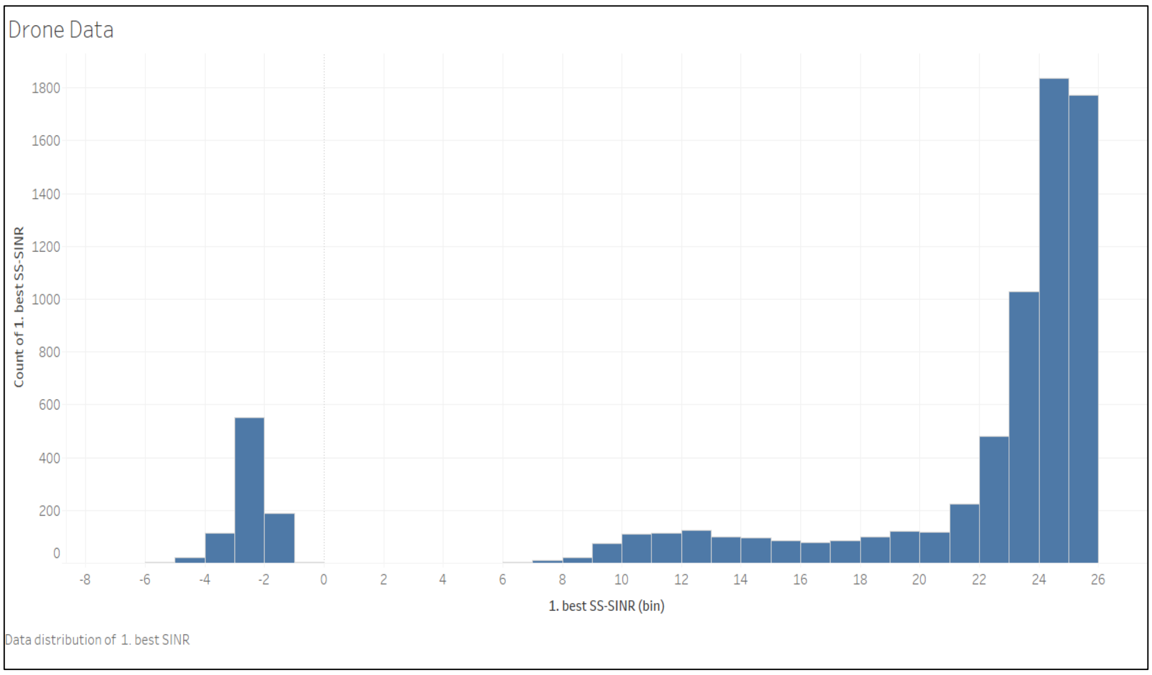


Figure 23. 1st-best-SINR data distribution for drone data.

From Figure 20 to Figure 23 we plot the data distribution of 1st-best-SINR. As mentioned in previous plots these graphs help us to check how the data varies and observe the peak value and skewness of the data.

The main use case of data visualization is that we can translate the information in our datasets into visual context. In our work data visualization was used to make it easier to identify patterns, outliers and trends in our dataset [30] moreover it also allows information to be conveyed more quickly. For example, without looking into numbers in our dataset we can see through these visualizations the location where we collected the data (Figure 8 – Figure 11) also the color of the point signifies the strength and value of RSPR value at that point. Since NR-PCI-beam index has categorical value through these visualizations we can see how its values are distributed. Through these visualizations we can observe that different classes have different categories of NR-PCI-beam index (Figure 12 – Figure 15). It can be observed that drone data have maximum number of categories, and the walking data has maximum amount of data in a single category. Without the data visualization it would be hard to figure out these patterns. Data visualization is essential for exploratory data analysis as it helps us to check the data quality and become familiar with the structure and features of the data.

In conclusion the data visualization helped us to verify the relationships between the data and through data visualizations we made sure our data does not have any outliers. Nevertheless, as described in above paragraph, it also helped us to understand data visually. Also, through data visualization we can see in our dataset we have different types. Different types of data convey different information and often must be analyzed differently.

3.4. Exploratory Data Analysis and Feature Engineering

After data visualization in chapter 3.3, we focus on exploratory data analysis (EDA) and feature engineering [31].

The EDA is used to summarize the main statistical characteristics of our data, it takes the data analysis to another level and helps us understand the data better. In our project, we used Pandas profiling[4] to perform the EDA, it is a python package which saves a lot of time. We just pass the data, and it generates the report in HTML. This is a powerful package, not only does it give an overview of our data but also calculates the 'interactions' where we can see how one feature is related to another. Moreover, it can also calculate correlations and missing values in dataset. The time taken to generate the report depends upon the data size. For the correlations, we have φ_k, it is a correlation coefficient that works consistently between categorical, ordinal and interval variables, captures non-linear dependency and reverts to the Pearson correlation coefficient in case of a bivariate normal input distribution.

Table 4 summarizes the statistics of the four different datasets that we have. We can check the total number of variables and observation along with missing cells and missing data percentage. It also gives us information about duplicate rows and the size of the dataset.

Table 4. Data Statistics

Data	Total variables	Total observations	Missing cells	Missing cells (%)	Duplicate rows	Duplicate rows (%)	Total size MB
Driving	11	9758	0	0.0%	0	0.0%	1.3
Walking	11	7120	0	0.0%	0	0.0%	0.9
Standing	11	7187	0	0.0%	0	0.0%	0.9
Drone	11	7468	1	<0.1%	0	0.0%	1.0

[4] https://pandas-profiling.ydata.ai/docs/master/rtd/

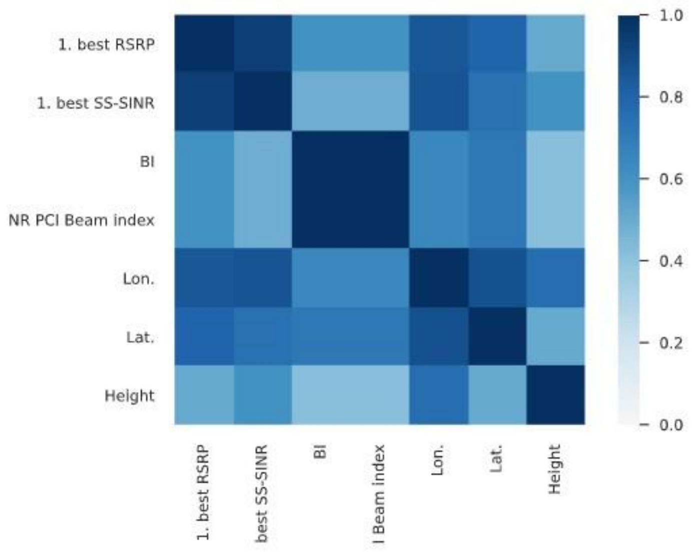


Figure 24. Features correlation for driving data.

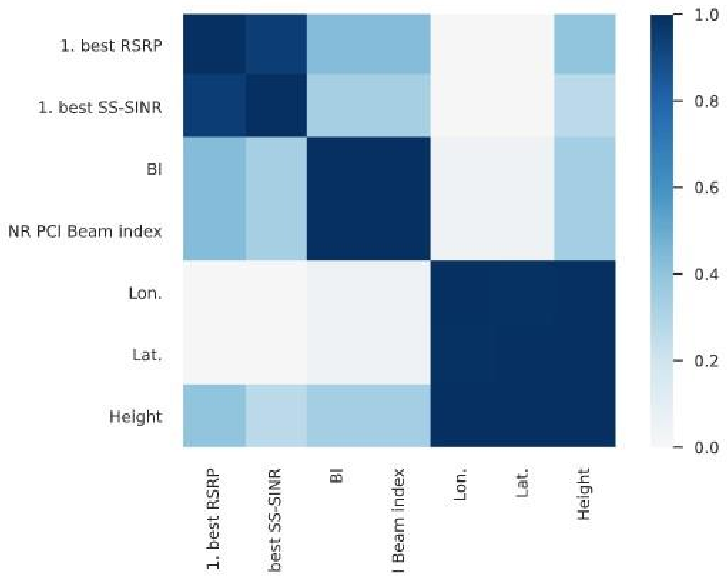


Figure 25. Features correlation for walking data.

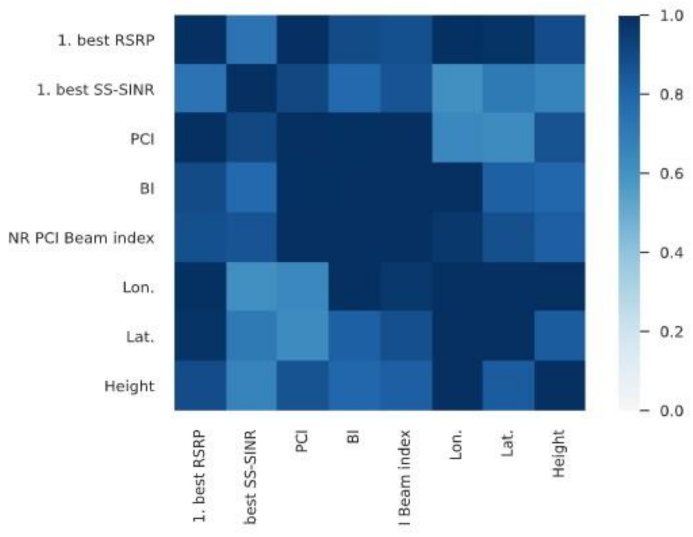


Figure 26. Features correlation for standing data.

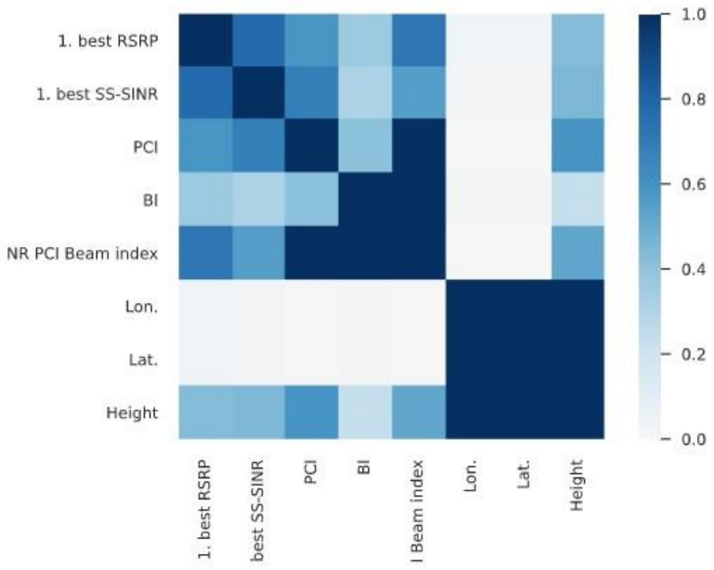


Figure 27. Features correlation for drone data.

EDA is a data exploration technique which is primarily used to understand the various aspects of the data. In many cases, EDA is used to find out what data may reveal outside of formal modelling and to understand how variables in a data collection interrelate. It could likewise assist us with sorting out whether or not the factual systems we are thinking about for information examination are fitting. Before modelling the data, it gives insight into all the data and the numerous interactions between the data elements [32].

In Figure 24 – 27 we have plotted the φ_k, correlation between the features in each dataset. Correlations help us find the degree of relationship between features in dataset which can be especially useful to find the underlying patterns and predicting the values [33],[34]. These correlations play significant role while model training.

The essential objective of EDA is to make information 'clean' suggesting that it ought to be without any trace of redundancies. It aids in identifying incorrect data points so that they may be readily removed, and the data cleaned. Besides, it helps us in fathoming the connection between the factors, furnishing us with a more extensive perspective on the information and permitting us to develop it by utilizing the connection between the factors. It also aids in the evaluation of the dataset's statistical measurements [32]. In Table 4, we can see the statistical measurements of our dataset. We can use these measurements to check if we have any missing and duplicate values so that we can remove it from our dataset. As evident from the Table 4 the datasets have similar statistics, that is because they were collected using same tool and in similar environment.

After performing EDA and feature engineering, we clean our data by dropping any missing values, in our datasets the only dataset that had missing values was drone data. However, this step is necessary for pipeline as in future data that we might have some missing values. After that, we balance the data set values by equalizing the number of instances or rows in all datasets including walking, driving, standing and drone. Imbalanced datasets create skewness in data which results in overrepresentation of one class and underrepresentation of another. When feeding to ML algorithm, the unbalanced data set creates bias towards overrepresented class. To avoid this problem, we created the balanced dataset [35]. To equalize the dataset, we choose whatever minimum value of rows, thus save in all four datasets, and keep that many rows in every dataset. After that, we sort the datasets according to the time and we can calculate the time difference between two consecutive samples or rows.

When it comes to feature engineering [31], we calculate and add two new features one of which is the distance covered between consecutive data points and another is the velocity between them. The distance is calculated from latitude and longitude coordinates using Haversine formula. The Haversine formula determines the great circle, (which is a circle on a sphere with the same centers as the sphere) distance between two points on a sphere given their longitudes and latitudes [36].

$$a = sin^2\left(\frac{\delta_\phi}{2}\right) + cos\phi 1 + cos\phi 2 * sin^2\left(\frac{\lambda_\phi}{2}\right) \quad (1)$$

$$c = 2 * atan2\left(\sqrt{a}, \sqrt{1-a}\right) \ldots \quad (2)$$

$$d = R * c \quad (3)$$

In Equation (1), $\delta_\phi$is the difference between 2 latitude coordinates. ϕ_1 is the first latitude. $\phi_2$is the second latitude. λ_ϕ is the difference between two longitude coordinates. Using the value that we got in Equation (1) in Equation (2), we calculate the value of 'c' which we finally use in (3) by multiplying it with the radius of earth represented by 'R' and whose value is 6371 kilometers. It should be noted that all the coordinate values are in radians. Since the coordinates are close to each other, we obtain a very small distance between the given two points. Also, while we are standing in one place, in ideal condition, the coordinates should be same all the time but as per our collected data suggest the coordinates differ slightly, due to which we have very small distance even in standing still data. We can fix this by just replacing them with zero.

The Haversine formula is frequently used for calculating distances between two points, it gives us a way to relate latitudes and longitudes to great circle distances. Great circle distance is the shortest distance between two points on the surface of a sphere, measured along the surface of the sphere (as opposed to a straight line through the sphere's interior).

Next, we calculate velocity between two consecutive points which is calculated using the previously calculated time delta and distance. Thus, we add these two features to our existing dataset. It should be noted that this process does create some outliers especially in distance feature. This is handled by taking only values which are handled by keeping only the values which were within +2 to -2 standard deviations of the data [37].

For our final data processing step, we split each class of data into the train and test set, and we window these train and test set separately using rolling mean. We take the window size 3 and then take the mean of the three values to create one new value. The process is then repeated by shifting the window one step ahead. After we finish windowing, we combine all the data into one single train dataset and test dataset. The data is then shuffled so that we can our models remain general and also overfit less.

3.5. One-Hot Encoding

In the collected data, along with the numerical values we have categorical values as well. The categorical values are also called nominal values. When we have this kind of feature in our dataset the possible values are limited to a fixed set. While some algorithms can understand and work with categorical value directly others cannot, and it might affect their performance. Many algorithms cannot understand categorical data, they work on numerical values only, that is where one-hot encoding is used. This is one of the simple yet important techniques in machine learning where we convert categorical values into numerical ones. There are other methods to convert categorical to numerical as well, but one-hot encoding is most widely used. One-hot encoding transforms a single variable with ‘k’ observations and ‘j’ distinct values, to ‘j’ binary variables with ‘k’ observations each. Once we apply one-hot encoding we create a binary vector in which each observation indicates the has binary 1 or 0 where 1 represents presence and 0 represents absence [38]. In this thesis, we had one categorical value namely NR-PCI-beam-index. We used One-hot encoding to convert it from categorical to numerical.

3.6. EDA Conclusion

In conclusion, EDA made it easy for us to see the relations between the data features as well as we got insight about the quality of data we are working with. Through EDA we found out in drone data we had some missing values which we dropped during data cleaning. Moreover, through EDA it was clear that the datasets were unequal, so to have equal representation of each class and to avoid our model being biased towards certain class we made sure all our datasets have equal number of observations.

4. MACHINE LEARNING APPROACHES

Now that the data for our models is prepared, we now discuss the machine learning approaches and algorithms that we use in our work. Although ML algorithms have a lot of categories, but our work is focused on supervised learning and thus we use the algorithms under the same category.

4.1. Supervised Learning

In supervised learning, the algorithm expects labeled data. When a dataset is labeled, it means we know what that data represents, or we can say we know what class our data belongs to, hence these labels ‘supervise’ the algorithm until it minimizes the loss or error and give us good predictions. Since we have the data as well as the labels, the algorithm learns the mapping between two and uses that to predict unseen data [39]. There are two major sub-categories or classes of algorithms in supervised learning which are categorized based on the output they generate; one is regression, and another is classification. In regression, the algorithm predicts the real value, e.g., integer or float values. In regression, the algorithm estimates the relationship between dependent variables and independent variables. Depending on number of independent variables, i.e., if the independent variable is one or more, the regression can be univariate or multivariate but regardless there will be only one output in both cases. The regression works by minimizing the error or cost while it tries to fit a line through data. This cost is minimized by adjusting the weights so that the line fits the data in such a way that the cost is minimum. When it comes to linear regression, the cost function we usually use is the square error cost [40].

$$J = \frac{1}{n}\sum_{1}^{n}(pred_i - y_i) \tag{4}$$

In Equation (4) ’n’ is the number of data points $pred_i$ is the predicted value and y_i is the actual value.

In classification, the algorithm learns the pattern in underlying data and tries to classify the data into desired classes. We can have as few as two classes and there is no upper limit. When we have only two classes, it is called binary classification and with more than two, it is called multiclass classification. An example of classification can be classifying emails as spam or not spam. Different ML algorithms for classification have different ways of classifying the data. The cost function used in classification is different than the one used in regression. In classifications, the most common cost function used is called cross-entropy loss. The algorithm calculates the probability of each class and the class having higher probability is considered the winner for the classification. The cross-entropy loss helps us to calculate the deviation of the predicted probability distribution from the actual one. Cross-entropy loss is also called log-loss. We go through log-loss in more detail in section 4.9 where we discuss the evaluation metrics. In this work, we also need to classify the UE motion type hence in the later part we will focus on the algorithms related to classification and try to explain how they work. In Figure 28, we can observe the case of classification where an algorithm tries to separate two classes.

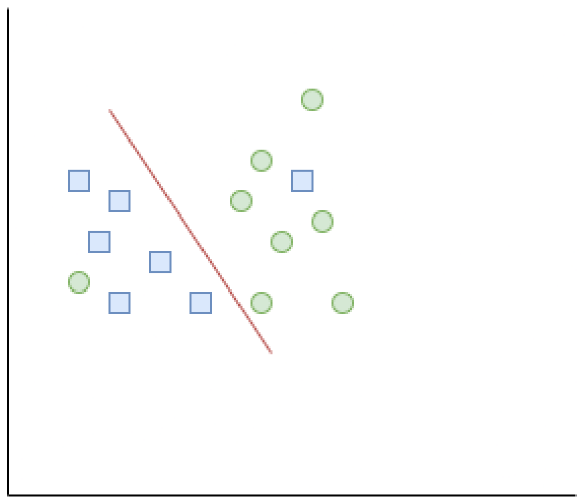


Figure 28. Illustration of classification algorithm in case of binary classification.

There are many classification algorithms available under supervised learning category. In the following sections we will discuss about the classification algorithms that we used in our thesis.

4.2. Logistic Regression

Logistic regression does have regression in its name, but it is a classification algorithm. It derives its name from sigmoid function which is also known as logistic function [41].

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{5}$$

Equation (5) represents the logistic function which is an S shaped curve whose value is bound between 0 and 1 and hence we get the probability of input belonging to defined classes. In Figure 29 we can see how the graph of sigmoid function looks.

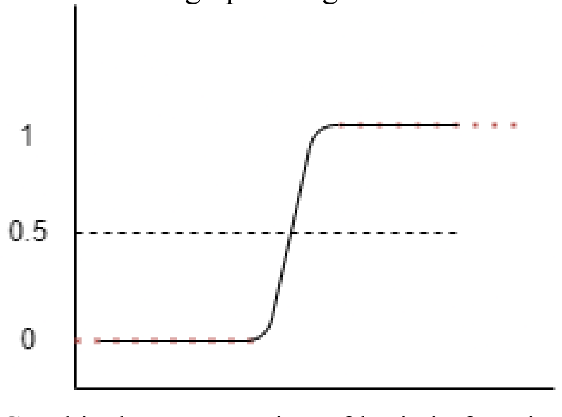


Figure 29. Graphical representation of logistic function.

4.3. Support Vector Machine

The support vector machine, also known as SVM for short, is a powerful ML algorithm. It can be used in both regression and classification tasks, but it is mostly used in classification. The way SVM works is that it tries to find a hyperplane in an N-dimensional space where N is the number of data features. There can be multiple orientations of the hyperplane that can separate the one class from another, but the SVM chooses one which will have maximum margin or maximum distance between data of all classes [42]. Figure 30 tries to describe visually how SVM works.

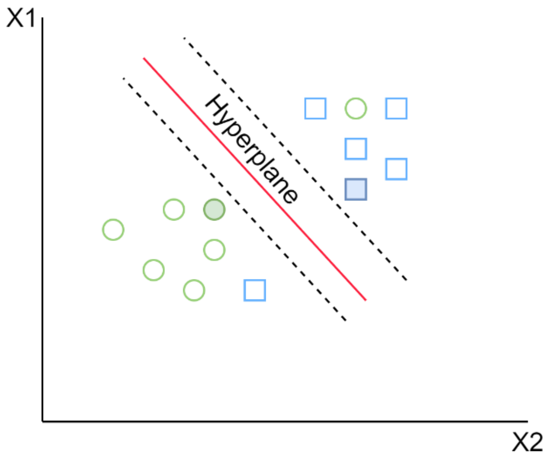


Figure 30. SVM classification in 2D.

Not only in 2D can we have SVM in any dimension. In Figure 31 we can see how SVM works in 3D. As the dimension of feature increase so does the dimension of the hyperplane between them. Hyperplanes are the decision boundaries that help to classify the data. The datapoints that are close to the hyperplane are called supporting vectors. They influence the orientation and the position of the hyperplane.

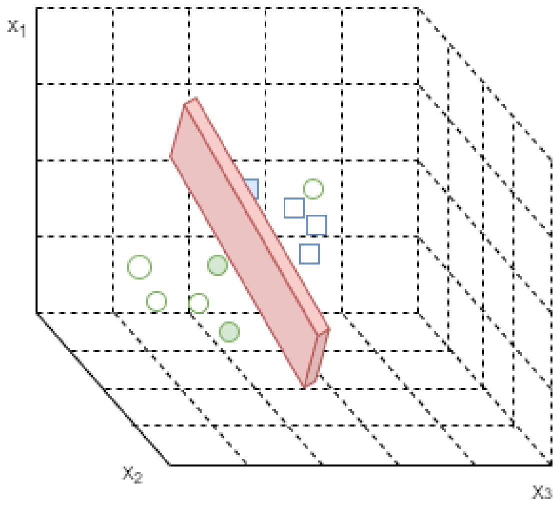


Figure 31. SVM classification in 3D.

4.4. KNN-Classifier

KNN is acronym for k-nearest neighbors. Like other supervised learning algorithms KNN works on data that has been labeled, it learns the underlying relationship between input and output and can classify the unseen data according to the mapping it made during learning phase. The KNN algorithm works on the principle that similar things exist close to each other. The 'k' in KNN is the number of neighbors the algorithm considers classifying the data [43]. There are several ways to choose k, and this may result in difference in the output. A smaller value of k can be noisy and can have a higher influence on the result. A large value of k can cause lower variance and increase in bias. One of the ways to select this parameter correctly is using cross validation. For classification of an unseen data the KNN relies on majority vote between k similar neighbors, similarity is defined according to a distance metric like Euclidean distance, represented by Equation (6), Manhattan distance, represented by Equation (7), Minkowski distance, represented by Equation (8), etc.

$$Euclidean = \sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{6}$$

$$Manhattan = \sum_{i=1}^{k} x_i - y_i \vee \tag{7}$$

$$Minkowski = \left(\sum_{i=1}^{n} \left|x^i - y^i\right|^p\right)^{\frac{1}{p}} \tag{8}$$

One of the disadvantages of KNN is that on large datasets, it requires a lot of computational power. Hence, more computational resources are needed for big data computation. Figure 32 describes visually the working of KNN.

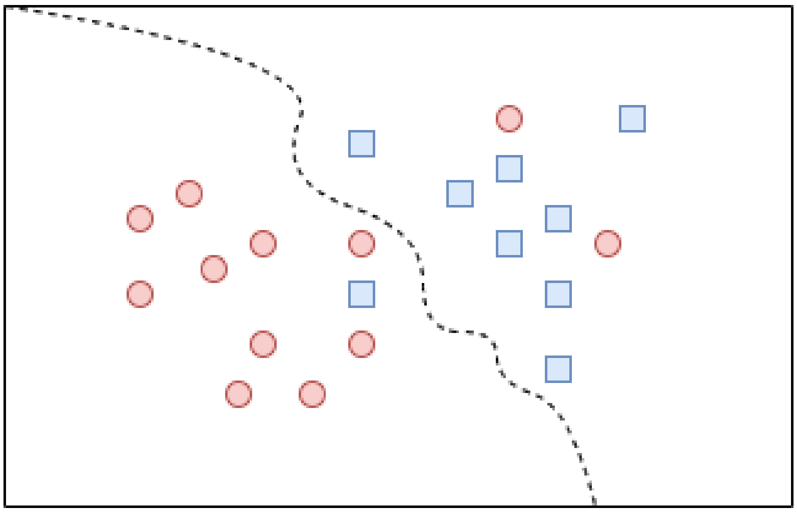


Figure 32. Illustration of KNN algorithm classifying the data.

4.5. Adaptive Boosting

Adaptive boosting, also known as Ada-boost for short, is an ensemble learning method. AdaBoost works on the mistakes of weak classifiers and uses an iterative approach to turn those weak classifiers into strong ones. For example, instead of using one decision tree, the ensemble method uses multiple decision trees and aggregates them together to form a strong predictor. Ensemble methods can be divided into two groups which are sequential learners and parallel learners. The models of sequential learners are generated sequentially, and the mistakes and errors of previous model are learned by their successor. Note that, Ada-boost belongs to sequential learners' category. In parallel learners, as contrary to sequential generation the base models are generated in parallel note that random forest algorithm belongs to this category. When it comes to the use case of Ensemble methods they can be used for Bagging [43] (which is used to decrease variances). Boosting [43], [44] (used to decrease bias), or Stacking [45] (used to improve predictions).

AdaBoost is a boosting algorithm [46] and is immensely powerful in combining several weak classifiers into a strong one. A single classifier which does not have a good accuracy is grouped together and each individual algorithm learns from the mistakes of its predecessor and progressively increases the overall accuracy of the model.

There are some drawbacks of using AdaBoost which include the fact that it is sensitive to outliers and noisy data. Hence the data should be cleaned and handled properly before using AdaBoost. Figure 33 represents the ensemble models visually.

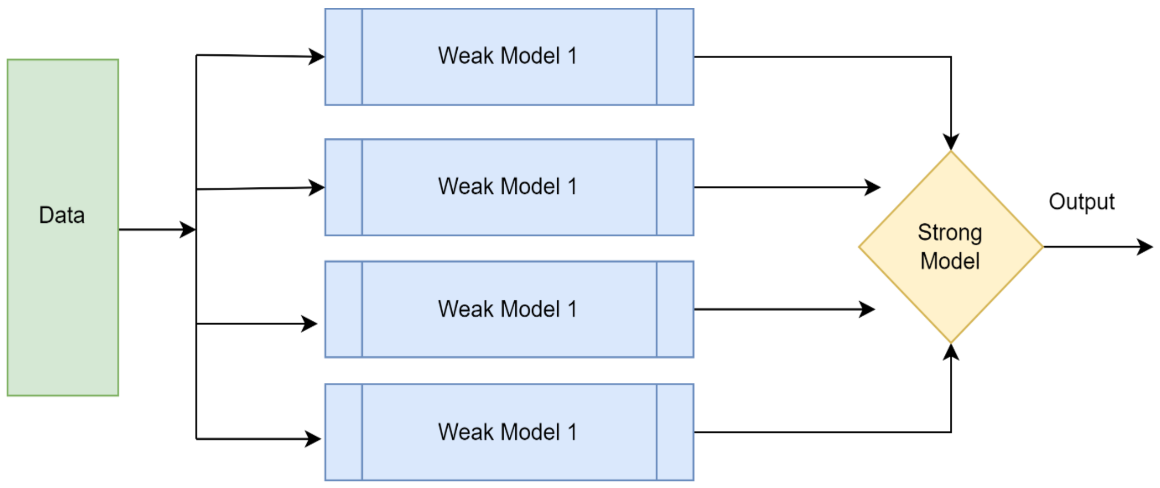


Figure 33. Illustration of how ensemble model combines weak models into strong one.

4.6. Linear Discriminant Analysis

LDA or Linear Discriminant Analysis is often used in dimensionality reduction, this algorithm can be used in classification as well [47]. In LDA, the algorithm projects the data in such a way that it maximizes class separability. In LDA the algorithm assumes that the classes are linearly separable so the more the number of classes the more hyperplanes will be created to distinguish the classes [48]. Figure 34 shows the visual representation of LDA algorithm.

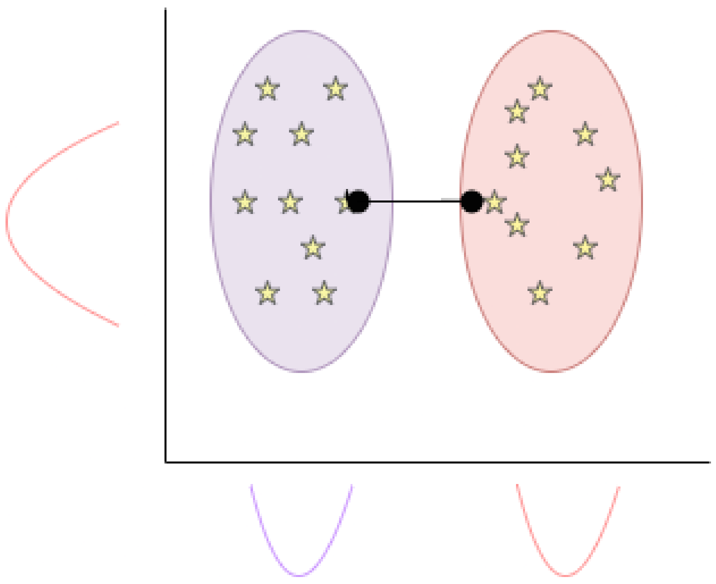


Figure 34. Illustration of LDA algorithm.

The limitations in LDA are such that if the data distribution is not gaussian, the LDA projections may not preserve complex structure in the data needed for classification.

4.7. Extra Tree Classifier

Extra tree classifier is also one of the ensemble algorithms which combines several decision trees together to make predictions. It is also known as Extremely Randomized Trees [49]. This algorithm is also called the Extremely Randomized Trees algorithm. The Extra Trees algorithm works by creating many unpruned decision trees from the training dataset. Predictions are made by averaging the prediction of the decision trees in the case of regression or using majority voting in the case of classification. In Extra Tree Classifier, the splits and features are selected randomly which makes it computationally less expensive than a Random Forest.

Figure 35 tried to illustrate graphically how Extra Tree Classifier splits the data randomly and then build multiple trees and finally combining them to make a strong classifier.

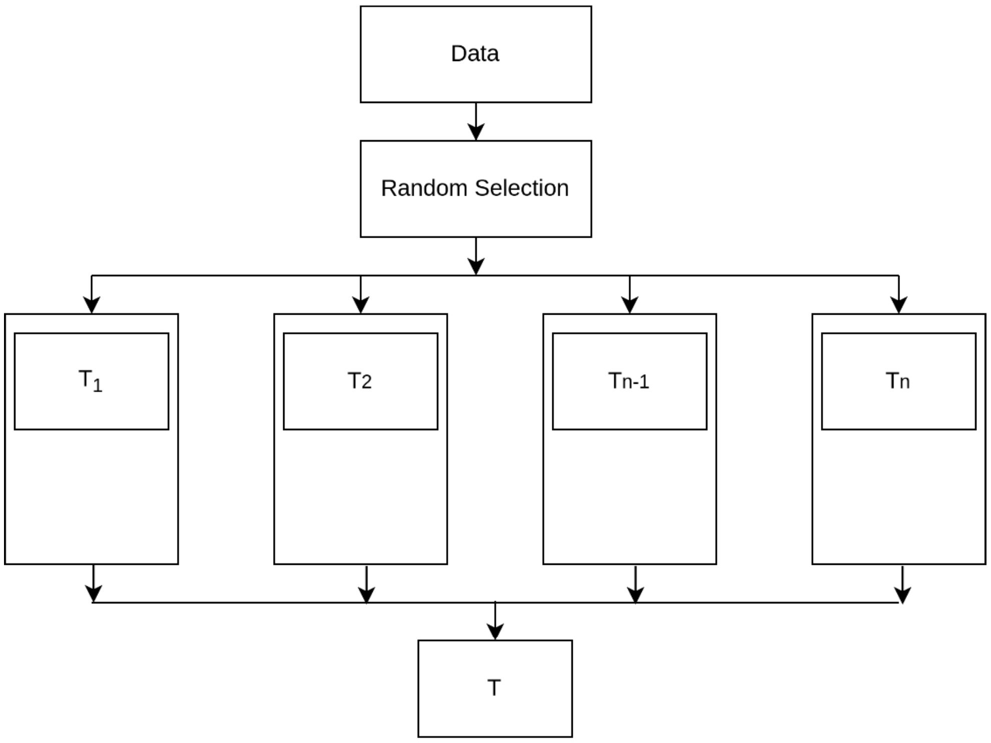


Figure 35. Graphical illustration of Extra Tree Classifier.

4.8. Evaluation Metrics

In machine learning, we evaluate model performance by tracking its metrics. In classification, we use a separate set of metrics than that of regression that is due to the nature of output that model produces. It should be noted that a metric is different from loss or cost function. A loss function is used to train the given model. A metric is used to evaluate the given model and the loss function is used during model training while a metric is used after the model has been trained [50]. We use metric on the test data, which is the data that our algorithm has not yet seen. The idea to evaluate model on test set is to make sure our model is not over or underfitting rather than it is generalizing. Since we work on classification, the metrics we will discuss belong to classification tasks.

Before explaining the metrics, we need to understand some terminology such as True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). True Positive is an outcome where the model has correctly predicted the class of the input data. True Negative is an outcome where the model correctly predicts the negative class, i.e., when the data does not belong to a certain category. False Positive is an outcome when the model incorrectly predicts the data and places it in true category, while the real values are false. False Negative is an outcome when model incorrectly predicts the negative class and places data in positive category. We used five metrics to track the performance of our model on the test set. The metrics we used included 'Accuracy' [50], 'F-Measure' [50], 'Precision' [50], 'Recall' [51], 'Log-loss' [51] and 'confusion matrix' [52].

- Accuracy:

  Represented in Equation (9), it is one of the simplest metrics for classification. It is equal to the number of correct predictions divided by total number of predictions, multiplied by 100

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100\,. \tag{9}$$

- Precision:

  If in any case your data is imbalanced the accuracy metric will not be a good one to choose. Precision calculates the percentage of correct predictions of positive class. Maximizing precision will minimize the false-positive errors. Precision is calculated by dividing true positive by the sum of true positive and false positive. Mathematically it is represented by Equation (10)

$$Precision = \frac{TP}{TP + FP}. \tag{10}$$

- Recall:

  Another important metric is recall, represented by Equation (11). It is calculated by dividing true positive by the sum of true positive and false negative. Maximizing recall will minimize the false-negative errors

$$recall = \frac{TP}{TP + FN}\,. \tag{11}$$

- F- Measure:

  Mathematically represented by Equation (12), the F-Measure or F-score is the harmonic mean of precision and recall. F-score is the special case of F-beta score in which the value of beta is equal to 1

$$F - score = \frac{2 * precision * recall}{precision + recall}. \tag{12}$$

- Log-loss:

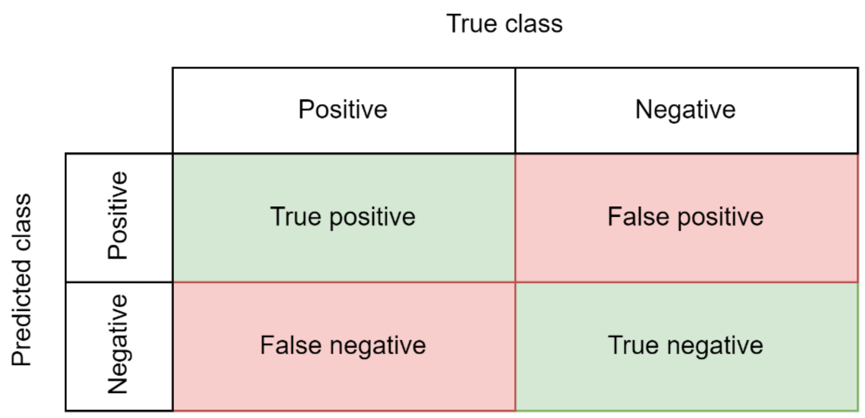
  Log-loss in another metric to measure the performance of the classification model. Log-loss tells us how close the prediction probability is to the actual value. The more the predicted probability diverges from the actual value, the higher is the log-loss value. The mathematical equation of log-loss is described in Equation (13)

$$Logloss = \frac{-1}{N}\sum_{i=1}^{N}[y_i ln p_i + (1-y_i)ln(1-p_i)]\,. \qquad (13)$$

  where "i" is the given observation, "y" is the actual value, "p" is the prediction probability, and ln refers to the natural logarithm (logarithmic value using base of e) of a number and N is the total number of observations.

- Confusion matrix:

  Another good way to see how our model is doing is using confusion matrix. It is a way we summarize the performance of an algorithm. It counts the number of data classified for each class and lists them in a matrix. This gives us an idea as to which class our algorithm is confused for. In Figure 36 we can see how confusion matrix looks like for binary classification.



		True class	
		Positive	Negative
Predicted class	Positive	True positive	False positive
	Negative	False negative	True negative

Figure 36. Graphical representation of confusion matrix for binary class.

5. IMPLEMENTATION

In the previous sections we discussed what problem we are trying to solve, the process of data collection, what operations we did on it, and how we did feature engineering to calculated new features. Then, we discussed more details of ML and described the implemented algorithms in this thesis as well as metrics that we used. this section, we will discuss about how we combined all the tools and what packages we used.

5.1. Language and Packages

For the machine learning part, we used python [53] as our programming language. Due to its versatility and flexibility, there are lot of contributions to this open source [54] programming language which made our work implementation easier. The rest of the packages are discussed in the next subsections.

5.2. Pandas

When it comes to data analysis and data manipulation pandas [55] is the best python library. It is fast, flexible, and open source. It converts data into data frames, which is a table consists of rows and columns and we can then run all sorts of operations on those data frames. Pandas can read the input file in multiple formats including ‘CSV’, ‘XLS’, ‘SQL’, ‘JSON’, etc. and can store the data back to the files with these formats. Moreover, using pandas makes it easier to plot the graphs. In our work we used pandas to read the files, clean the data, manipulate the data, and one-hot encoding.

5.3. Matplotlib and Seaborn

The first step in every machine learning or data science project is to visualize the data. Data visualization helps in spotting trends, outlier, and patterns in the data. It also helps us to see how our data is distributed. Not only Matplotlib [56] and Seaborn [57] help us to visualize the data trends it also helps us to plot neat graphs and charts of our results. Both Matplotlib and Seaborn are python based and Seaborn is developed on top of Matplotlib. Using these libraries, we can create static, animated, and interactive visualizations. In our work we used Matplotlib along with seaborn to plot the graphs and charts.

5.4. Pandas Profiling

Pandas profiling [58] is a great tool to generate reports for our datasets. With one line of code, it gives us information about datasets including datatypes, missing values, quantile statistics, descriptive statistics, histogram, and correlations between the various features. We used this package to generate the reports for our data.

5.5. Scikit-learn

Without any doubt one of the most famous and useful python packages for machine learning is Scikit-learn [59]. Build on top of python, it is an open-source package that packs up most of the ML algorithms. It has algorithms for classification, regression, clustering, and for data pre-processing. It also comes but inbuild datasets to try the algorithms.

In this thesis, we used Scikit-learn to split the data into training and testing sets. Moreover, this library also provides us with ways and functions to track the performance of our model.

5.6. Ipyleaflet

Ipyleaflet is an interactive widget library. We used it to display interactive maps in our notebook so that we could visualize predictions over various locations. An especially useful feature of this library is that we can dynamically update the attributes of a widget [60] that means we can dynamically change the map parameters.

5.7. ML-Flow

Since we trained six different machine learning models, we needed an efficient way to keep track of the performance and metrics. We also required a smart method to compare the models, so for that purpose we used ML-flow. An open-source platform for ML lifecycle [61]. It offers four different components MLflow Tracking, MLflow projects, MLflow models, and MLflow Registry. In this thesis, we were only interested in MLflow tracking as we needed to track different models. Using MLflow tracking not only can we compare different models, but we can compare same models with changed parameters.

5.8. Integration Tools

By using all the tools mentioned, we are now ready to implement the entire system. We first begin with the process of data collection then we follow the steps already described above, i.e., we visualize the data. Then the data is cleaned, and feature engineering is done along with the one-hot encoding. Data is then windowed and driving data, walking data, standing still data and drone data is then combined into one single dataset. This dataset is then divided into training and test datasets using scikit-learns *train_test_split* function. 30% of data is kept for testing and 70% is kept for training. Splitting the data into training and testing set is necessary as it helps us to check how well our model is doing on unseen data. Our model does not see the test data until we finish training it and we always evaluate our model on test data. Moreover, the training data is shuffled, shuffling data serves the purpose of reducing variance and making sure that models remain general and overfit less. Also, we make sure that all the classes are equally represented in the training and test datasets hence our datasets are not skewed.

At last, our data is ready, and we train previously discussed models on training data. The implemented algorithms include Logistic regression cv, SVC, KNN, AdaBoost, LDA and Extra tree classifier. All of these are provided by Scikit-learn. We train the algorithms on default hyper-parameters and check which among those perform best. We check the performance of each model on test data, then we monitor the performance metrics which are accuracy, precision, recall, F-measure, log-loss. All these metrics are provided by scikit-learn library and are discussed above. With each training and testing we used MLflow tracking to log the parameters and metrics. MLflow provides UI to compare and view models and metrics. Figure 37 shows a combined view of how different models performed. The lines in the plot are color coded, red represent the metric value is close to 1 and blue represent the values are closer to 0. Middle values i.e., green, yellow and orange represent the value between 0 and 1.

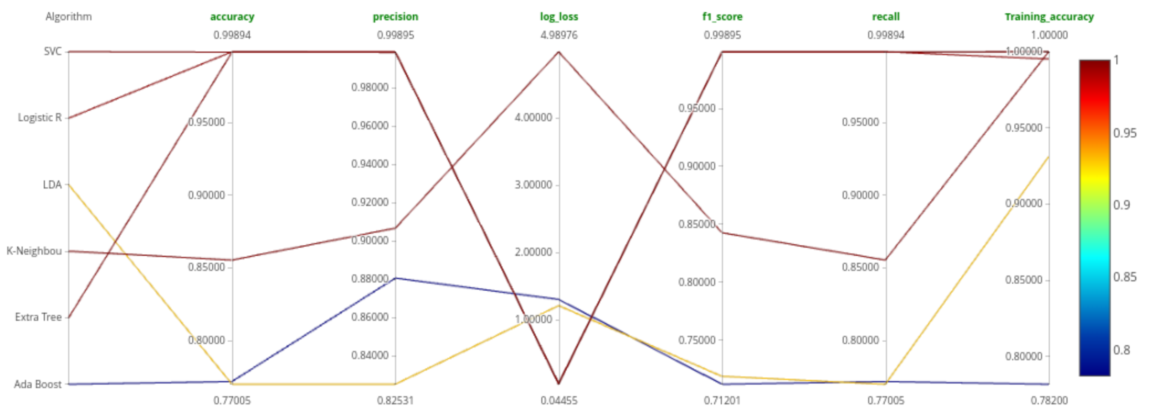


Figure 37. Parallel coordinates plot showing how metrics vary for different algorithms.

6. RESULTS

This chapter presents the results from all the models that were used in this thesis work. The models are evaluated on test data which is 30% of the original data. It is necessary to evaluate the data that the model has not seen yet so that we can check if the model is able to generalize well or not. The evaluation metrics we calculate for our algorithms include accuracy, recall, precision, F-measure, and log-loss. All these evaluation metrics have been discussed in earlier chapter in detail.

6.1. Logistic regression CV

Table 5 summarizes the results of the logistic regression cv. The table includes the values of test accuracy, precision, recall, F-measure, log-loss and training accuracy. As we can observe from the evaluation metric data, logistic regression cv performed well on the test dataset and had a test accuracy of 0.99. It can also be observed that log-loss in logistic regression cv is 0.057. The parameters that were used to train this model can be found in Appendix 1.

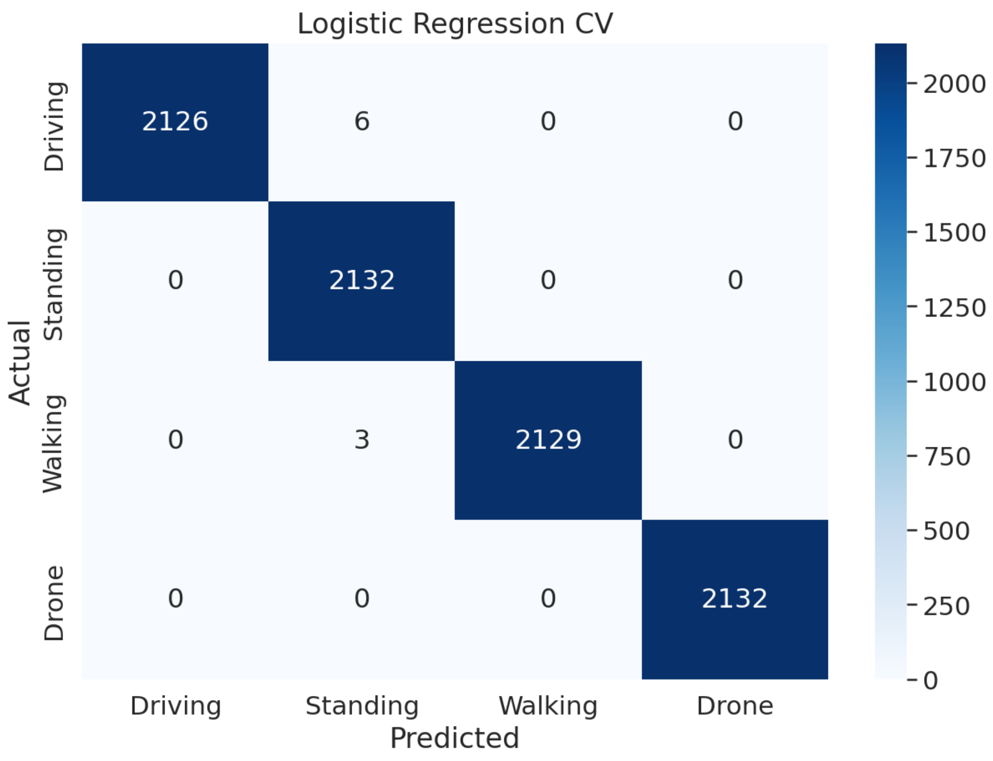


Figure 38. Confusion matrix of logistic regression CV.

As evident from the confusion matrix in Figure 38, the model performed well on drone data and standing data. It misclassified 6 driving data as standing data and 3 walking data into standing data. Standing data and drone data are classified correctly.

6.2. SVC

The results of the SVC are summarized in Table 5. According to the data we can see that training accuracy of SVC is equal to 0.99 and the log-loss is 0.060 which signifies that SVC performed well on the data it has not seen before. The parameters used to train our SVC model can be found in Appendix 2.

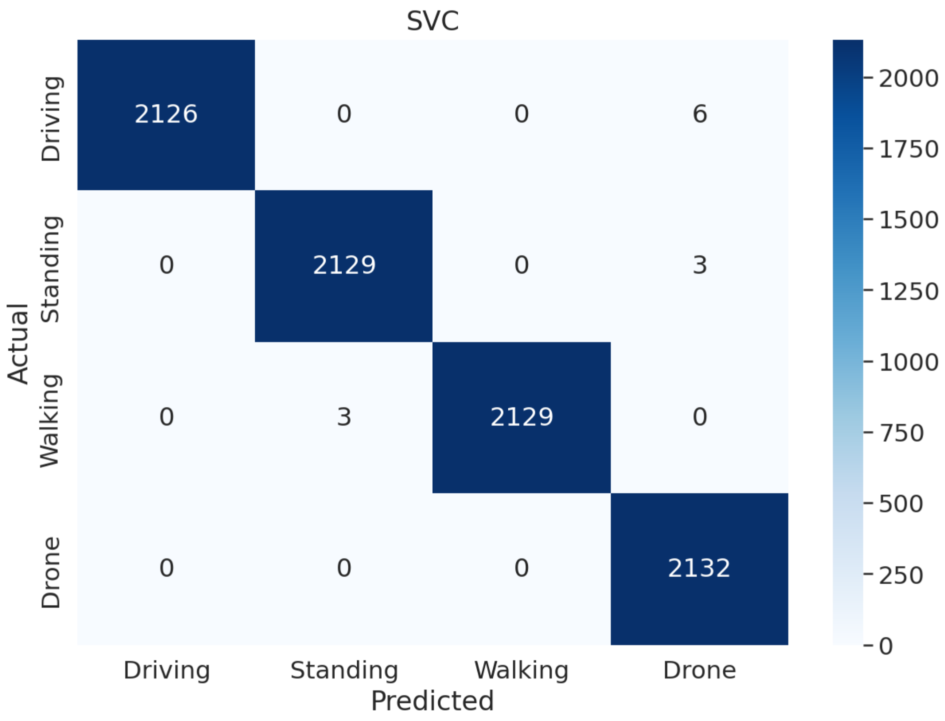


Figure 39. Confusion matrix of SVM.

Upon analyzing confusion matrix in Figure 39, we can observe that SVC classified all datapoints in drone correctly. 6 data from driving are misclassified as drone. 3 from walking are classified as standing and 3 from standing are classified as drone. overall SVC performed well on the test set.

6.3. KNN

For KNN the evaluation metrics are summarized in Table 5. As observed the test accuracy for KNN is 0.85 while the log-loss, with the value of 4.98, is higher than the previous algorithms. Generally, the log-loss should be low for a good classification algorithm. Appendix 3 lists out the parameters that were used to train the KNN model.

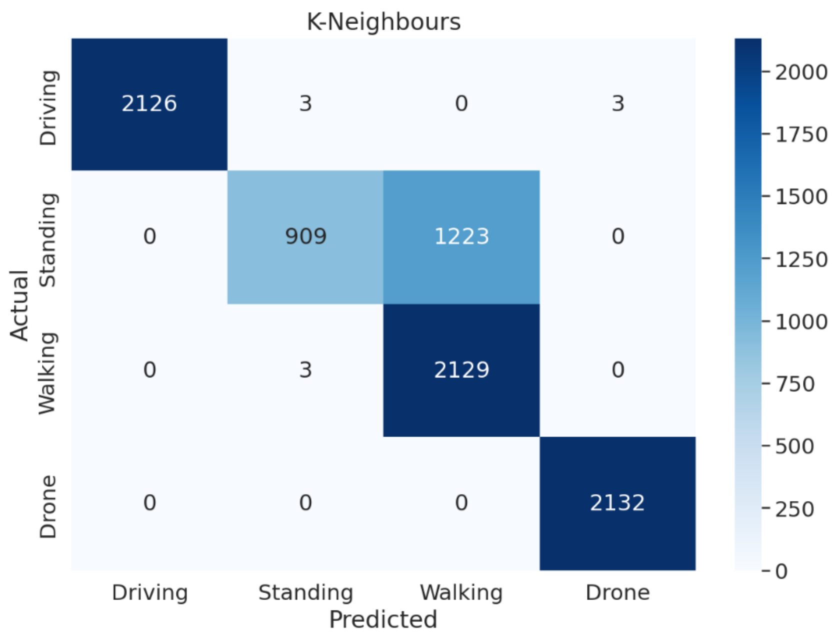


Figure 40. Confusion matrix of KNN.

The confusion matrix for KNN, Figure 40, has the most misclassified cases for standing data. It misclassified 1223 data points as walking while they were standing and 3 of driving class are classified as standing and 3 as drone. 3 from walking were also misclassified as standing. This is the reason the log-loss of KNN is higher than other algorithms.

6.4. AdaBoost

Table 5 summarizes the results of the Ada Boost model. As evident from the data we can see that the test accuracy is only 0.77 much less than the other algorithms used. Although the log-loss is 1.30 which is less than KNN, but it is still higher than the other algorithms. In Ada Boost the training accuracy is also less than other with the value of 0.78. In Appendix 4 the list of parameters for our Ada Boost model can be found.

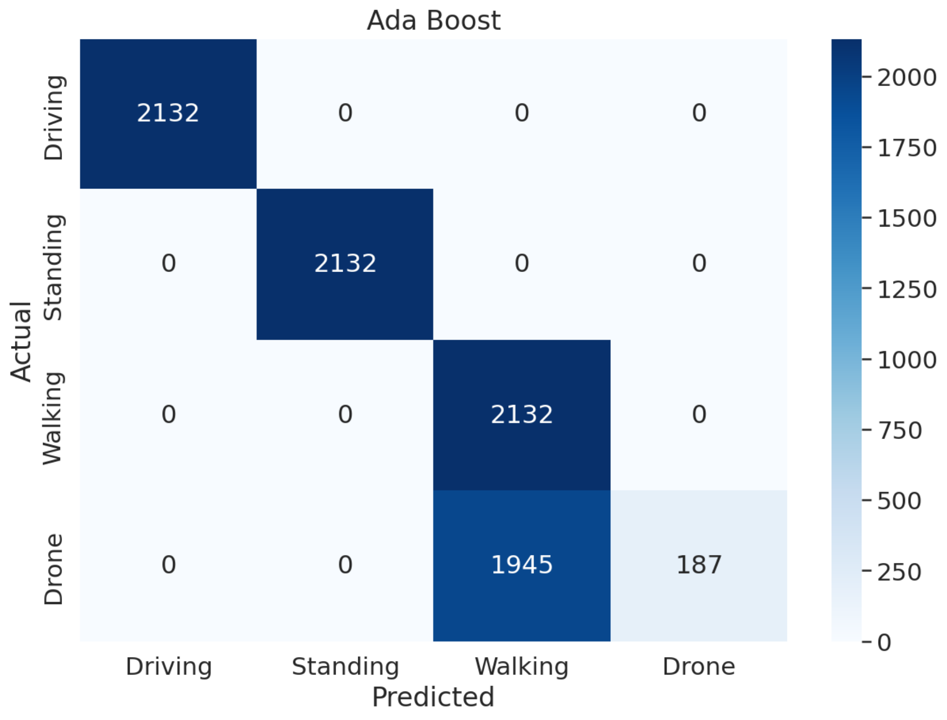


Figure 41. Confusion matrix of Ada Boost.

As evident from Figure 41, the confusion matrix for AdaBoost, this algorithm did not perform well on drone data. It misclassified 1945 data points from drone into walking data. Although it performed well in other classes, it performs poorly when it comes to drone data.

6.5. LDA

Table 5 summarizes the evaluation metrics of LDA model. We can observe that the test accuracy for LDA is 0.77 and the log-loss is 1.21, the results of LDA model are similar to the Ada Boost model. The difference can be seen in confusion matrix. Also, the parameters used to train the LDA can be found in Appendix 5.

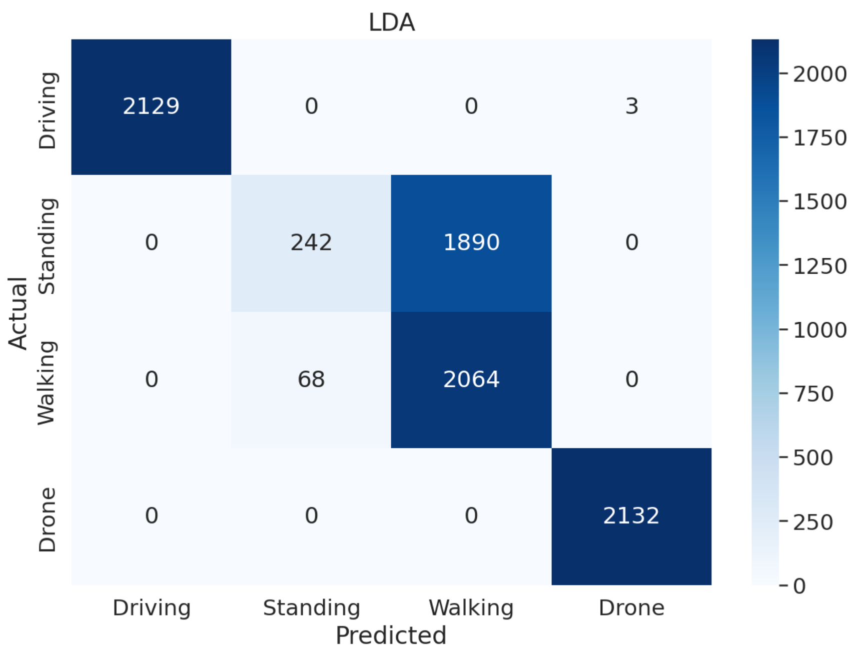


Figure 42. Confusion matrix of LDA.

The confusion matrix of LDA is displayed in Figure 42. Although the metrics of LDA is similar to Ada boost, it is through confusion matrix that we can spot the difference. LDA worked well on drone classes, but it performed poorly on standing data, it misclassified 1890 standing class as walking. It also misclassified 3 driving data as drone data. Apart from that 68 of walking classes are also misclassified as standing data.

6.6. Extra Tree Classifier

Table 5 summarizes the evaluation metrics of the Extra Tree Classifier. We can observe that test accuracy of Extra tree classifier model is 0.99 and the log-loss is also low with the value of 0.044. Apart from that the other metrics are also with high value of 0.99. The parameters for this model can be found in Appendix 6.

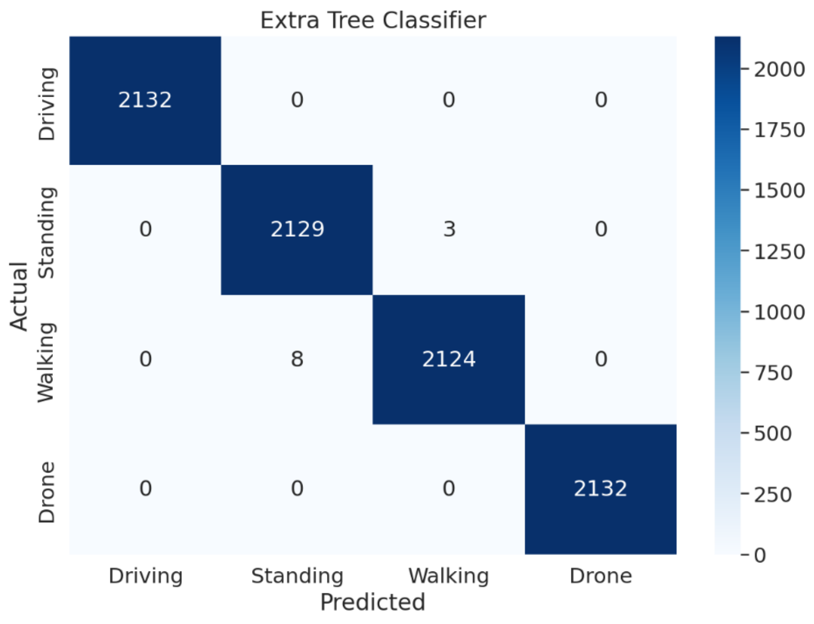


Figure 43. Confusion matrix of extra tree classifier.

As evident from confusion matrix in Figure 43, extra tree classifier performed extremely well on every class. It only misclassified 3 standing data into walking classes and 8 of walking class is misclassified as standing data.

Table 5. Evaluation Metrics

Algorithm	Accuracy	F1-Score	Log-loss	Precision	Recall	Train acc
Extra Tree Classifier	0.99871	0.99871	0.04455	0.9987115	0.9987101	1.00
Logistic Regression CV	0.99894	0.99894	0.05763	0.9989490	0.9989446	0.99
Support Vector Classifier	0.99859	0.99859	0.06043	0.9985973	0.9985928	1.0
K-nearest Neighbor	0.85553	0.84274	4.98975	0.9067951	0.8555347	1.0
Adaptive Boosting	0.77192	0.71200	1.30565	0.8807333	0.7719277	0.7819
Linear Discriminant Analysis	0.77005	0.71876	1.21334	0.8253107	0.7700515	0.9311

The evaluation metrics for every model have been collected in Table 5 for easy comparison and in the table, we can observe the value of log-loss for each algorithm. Log-loss is the most important classification metric based on probabilities. For any given problem, a lower log loss value means better predictions and for our data and parameters we can see that the log-loss of Extra tree classifier having the value of 0.044 and KNN has the highest log-loss with the value of 4.9.

From Figure 44 to Figure 49 each metric for every model have been plotted together to gives us better understanding of how each metric varied for different models.

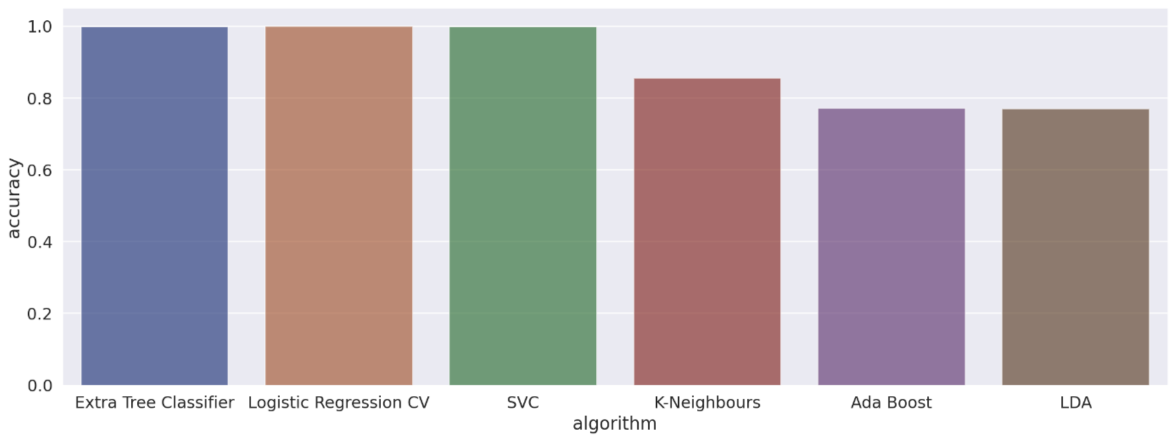


Figure 44. Comparison of accuracy metric of all the algorithms used.

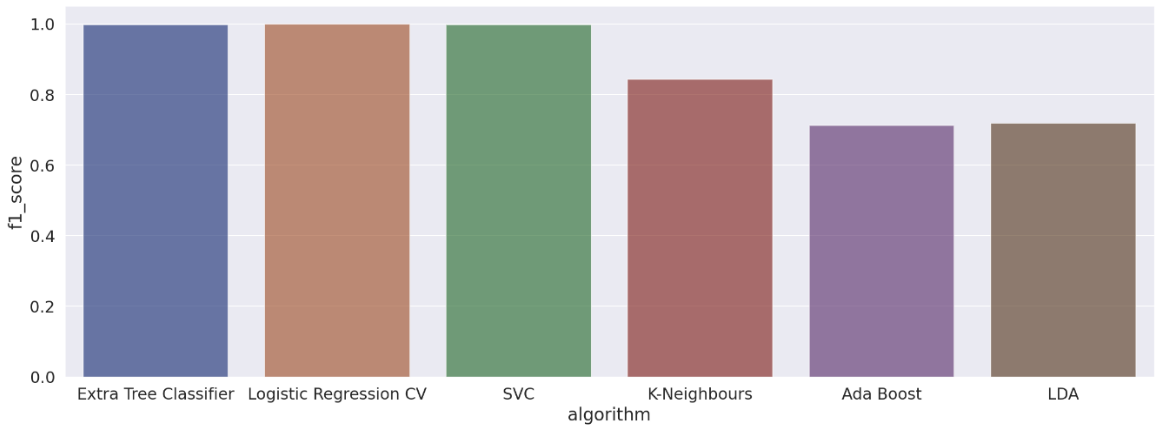


Figure 45. Comparison of F1-score metric of all the algorithms used.

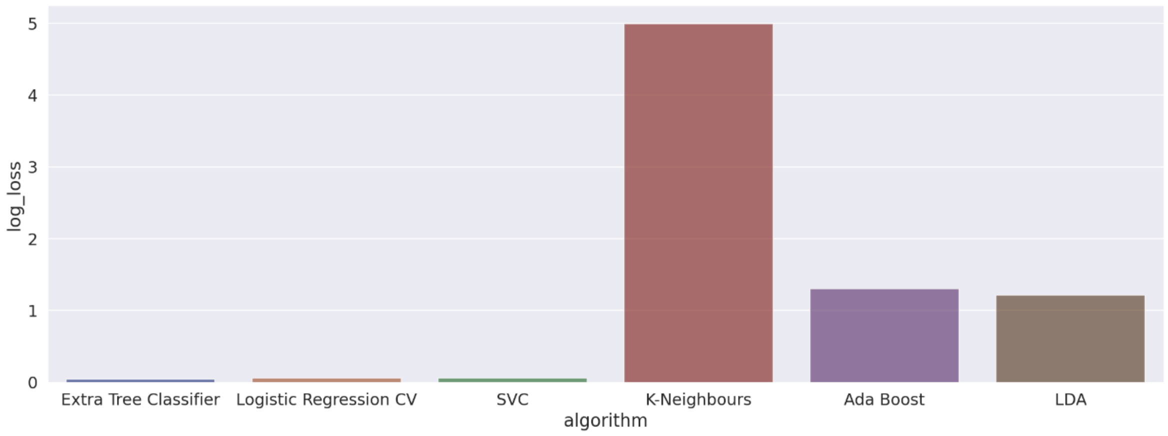


Figure 46. Comparison of log-loss metric of all the algorithms used.

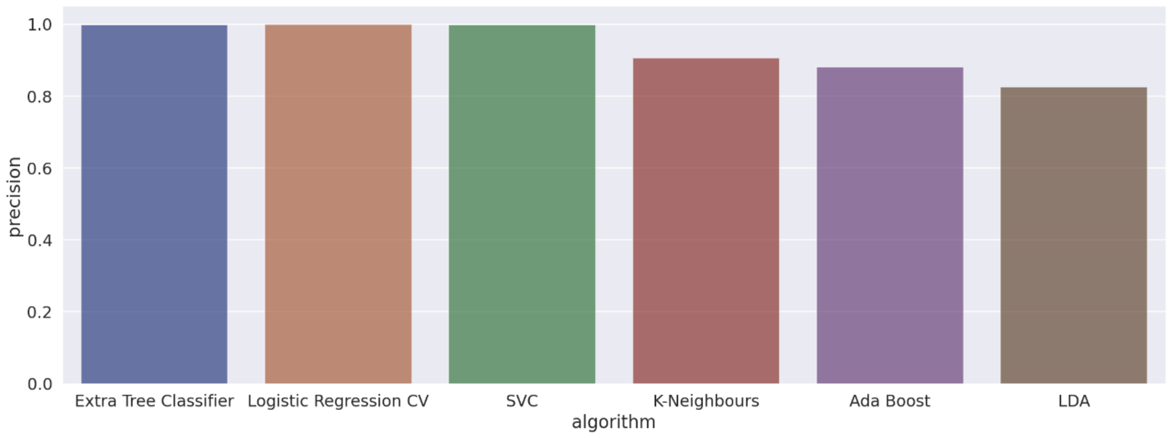


Figure 47. Comparison of precision metric of all the algorithms used.

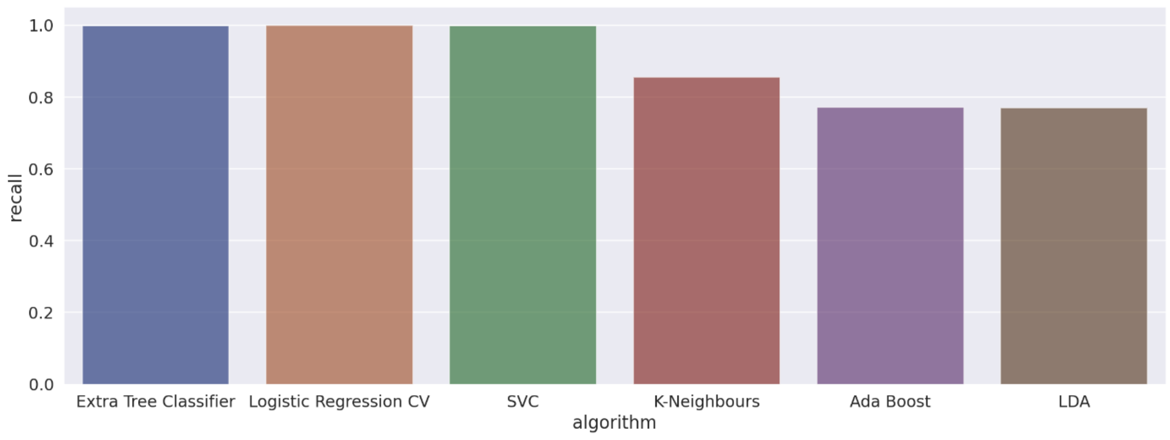


Figure 48. Comparison of recall metric of all the algorithms used.

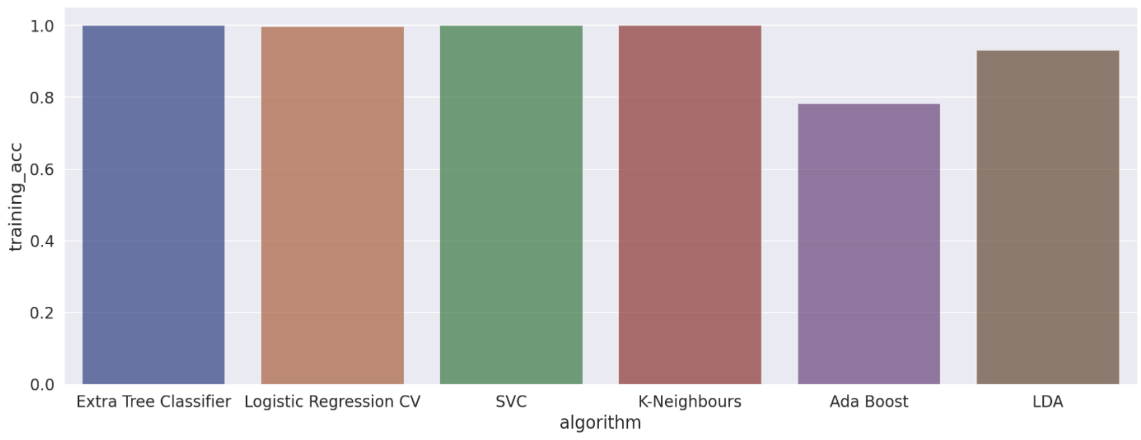


Figure 49. Comparison of training accuracy of all the algorithms used.

From all the above graphs we can observe and compare the performances of all the algorithms and the visual representation of these metrics also suggest that the Extra tree classifier performed better than other algorithms in every metric. Although it is evident that logistic regression CV and SVC algorithms also performed very well, but, as far as the log-loss is concerned Extra tree classifier had the minimum of all.

Although the goal of each model was to predict and classify the motion type of the user, not every model had good accuracy, as evident from the metric data we can see that Ada Boost, KNN, and LDA did not perform well on certain classes of data. This can be because of the variance. The variance is a measure of how sensitive the algorithm is to the specific data used during training. A more sensitive algorithm has a larger variance, which will bring about more contrast in the model, and thusly, the forecasts made and assessment of the model. Conversely, a less sensitive algorithm has a smaller variance which will bring about less contrast in the subsequent model with different training data, and thus, less distinction in the subsequent expectations and model assessment. Algorithms with a high variance often require more training data than those algorithms with less variance. The variance can be lowered with increasing the size of the training dataset. Another way to reduce the variance is by changing the hyperparameters of an algorithm.

6.7. Pipeline Overview

In Figure 50, we can see the whole pipeline from data cleaning to making predictions and evaluation of our models. This helps us to visualize quickly through what steps our data passes.

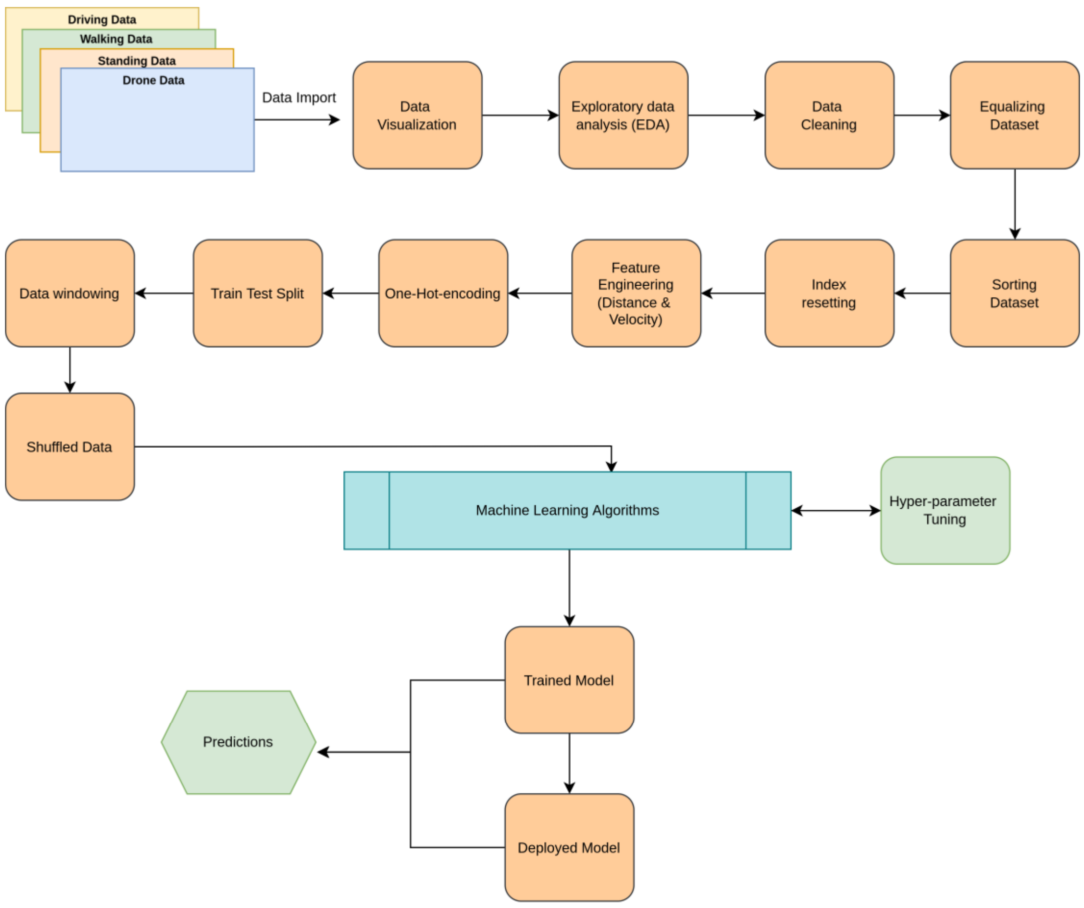


Figure 50. Flow diagram of pipeline.

7. DISCUSSION

Integrating machine learning to industry solutions have become a norm. Using ML in products have elevated the user experience. ML can be tremendously helpful in communication industry and can bring promising results. As the metrics data suggested, in this work, extra tree classifier gave promising results in classifying the motion type of the user.

The trained algorithms were able to classify the data correctly, but one important question that arises here is what if we fly a drone at a comparable height of a human and with the speed which is that of human walking. what will our algorithm predict? The answer to that question is that you do not have to treat these classes as they are by that we mean if an algorithm classifies it as walking it indicates that system should provide it the bandwidth and beams that are required for a walking person even if it is a drone because given the state it is in it will require specific bandwidth and beam. The same argument is valid if a driving car stops and stays in one place even though human and UE is in car it should get bandwidth to that of standing still user.

Our work is a part of bigger pipeline. In our work, we only do prediction as to what is the motion type of the UEs. The output can be fed to the next pipeline which determines and changes beamforming and allocates bandwidth to the user. Or can be used in digital twin of the radio.

7.1. Future work

This work paves the way for further implementation of machine learning in the field of communication where our work can be integrated and used with pipeline that controls the beamforming and beam steering. The future work can also be done by integrating the current hardware with the algorithms.

As far as the improvements to this implementation is concerned, the models can be trained with larger dataset and with more set of classes. Large dataset will also address the high variance problem of the algorithms that did not perform well. Also, with a good hardware all the algorithms can be tuned using hyper-parameter tuning which will result in more robust and accurate predictions.

The results of this work are promising although there are some limitations as well. First, the data has been collected only around single base transceiver station (BTS). Due to this limitation our data has similar characteristics and statistics. To address this limitation in future work, the data must be collected around different BTS so that our data also represent the network handovers.

Another limitation is that currently this work also requires the localization features, although they are not directly used in model training, they are however required to calculate the feature engineered variables. In future work the model can be made completely independent of localizations.

As far as limitations of the current implementation is concerned the current method does not utilize the hyper-parameter tuning rather the algorithm parameters are trained on default parameters.

8. CONCLUSION

Machine learning has revolutionized every technological industry. Since the last decade ML and AI have seen tremendous growth both academically as well as in industries. ML algorithms can find hidden insight quickly and easily as compared to humans and often outperform humans in various tasks. This is the main reason we used ML for our work.

In this work, we were trying to predict the UEs motion type using the 5G data. We were interested in this problem as it could later be used in combination of a bigger pipeline of beam steering and beamforming and in Digital Twins. This work will also be useful in a system which will provide different bandwidth to different UEs based on their need and activity. In the beginning of the thesis, we started with studying state-of-the-art and prior work done in this field which we summarized in introduction in Chapter 2. In our work, we followed the general procedure of ML pipeline which started by collecting all the relevant 5G data, and we finally implemented and verified the UE classification performance using ML algorithms. The collection and analysis that we did on the data is summarized in Chapter 3. We used open-source tools in our work and tried different ML algorithms including Logistic regression cv, SVC, LDA, KNN, AdaBoost and Extra Tree Classifier as discussed in Chapter 4. We used different metrics to track the performance of the algorithms, the metric included accuracy, precision, recall, F-measure, and log-loss. In Chapter 5, we discussed the libraries used and how we integrated everything together.

Upon analyzing the metrics that we got in Chapter 6, for all the algorithms, the data suggested that the extra-trees classifier method performed better than other algorithms followed by SVC. All the metrics of extra-trees classifier were ~99% and log-loss was also the lowest than other algorithms and had a value of 0.044.

9. REFERENCES

[1] S. Das, A. Dey, A. Pal, and N. Roy, “Applications of Artificial Intelligence in Machine Learning: Review and Prospect.”

[2] D. Pitchford, “A Decade Of Advancements As We Enter A New Age Of AI,” *Forbes*. https://www.forbes.com/sites/danielpitchford/2020/12/31/a-decade-of-advancements-as-we-enter-a-new-age-of-ai/ (accessed Mar. 07, 2022).

[3] “The History of Artificial Intelligence - Science in the News.” https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/ (accessed Mar. 03, 2022).

[4] S. Sah, *Machine Learning: A Review of Learning Types*. 2020. doi: 10.20944/preprints202007.0230.v1.

[5] “Total data volume worldwide 2010-2025,” *Statista*. https://www.statista.com/statistics/871513/worldwide-data-created/ (accessed Mar. 03, 2022).

[6] R. Zeqiri, F. Idrizi, and H. Halimi, “Comparison of Algorithms and Technologies 2G, 3G, 4G and 5G,” in *2019 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, Oct. 2019, pp. 1–4. doi: 10.1109/ISMSIT.2019.8932896.

[7] Meenal G. Kachhavay and Ajay P.Thakare, “5G Technology-Evolution and Revolution,” *Int. J. Comput. Sci. Mob. Comput.*.

[8] E. Ali, M. Ismail, R. Nordin, and N. F. Abdulah, “Beamforming techniques for massive MIMO systems in 5G: overview, classification, and trends for future research,” *Front. Inf. Technol. Electron. Eng.*, vol. 18, no. 6, pp. 753–772, Jun. 2017, doi: 10.1631/FITEE.1601817.

[9] M. Mantash and T. A. Denidni, “Millimeter-wave beam-steering antenna array for 5G applications,” in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Oct. 2017, pp. 1–3. doi: 10.1109/PIMRC.2017.8292713.

[10] T. S. Rappaport *et al.*, “Millimeter Wave Mobile Communications for 5G Cellular: It Will Work!,” *IEEE Access*, vol. 1, pp. 335–349, 2013, doi: 10.1109/ACCESS.2013.2260813.

[11] Z. Cao *et al.*, “Advanced Integration Techniques on Broadband Millimeter-Wave Beam Steering for 5G Wireless Networks and Beyond,” *IEEE J. Quantum Electron.*, vol. 52, no. 1, Art. no. 1, Jan. 2016, doi: 10.1109/JQE.2015.2509256.

[12] G. Mumcu, M. Kacar, and J. Mendoza, “Mm-Wave Beam Steering Antenna With Reduced Hardware Complexity Using Lens Antenna Subarrays,” *IEEE Antennas Wirel. Propag. Lett.*, vol. 17, no. 9, Art. no. 9, Sep. 2018, doi: 10.1109/LAWP.2018.2857441.

[13] A. Rasheed, O. San, and T. Kvamsdal, “Digital Twin: Values, Challenges and Enablers From a Modeling Perspective,” *IEEE Access*, 2020, doi: 10.1109/ACCESS.2020.2970143.

[14] M. G. Kibria, K. Nguyen, G. P. Villardi, O. Zhao, K. Ishizu, and F. Kojima, “Big Data Analytics, Machine Learning and Artificial Intelligence in Next-Generation Wireless Networks,” *ArXiv171110089 Cs Math*, Feb. 2018, Accessed: Mar. 07, 2022. [Online]. Available: http://arxiv.org/abs/1711.10089

[15] C. H. Yu, “Exploratory data analysis in the context of data mining and resampling,” *Int. J. Psychol. Res.*, vol. 3, Jun. 2010, doi: 10.21500/20112084.819.

[16] M. Haddad *et al.*, “Mobility state estimation in LTE,” in *2016 IEEE Wireless Communications and Networking Conference*, Doha, Apr. 2016, pp. 1–6. doi: 10.1109/WCNC.2016.7564917.

[17] C. Studer, S. Medjkouh, E. Gönültaş, T. Goldstein, and O. Tirkkonen, “Channel Charting: Locating Users within the Radio Environment using Channel State Information,” *ArXiv180705247 Cs Eess Math Stat*, Aug. 2018, Accessed: Mar. 08, 2022. [Online]. Available: http://arxiv.org/abs/1807.05247

[18] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, “Deep Convolutional Neural Networks for Massive MIMO Fingerprint-Based Positioning,” *ArXiv170806235 Cs Math Stat*, Aug. 2017, Accessed: Mar. 08, 2022. [Online]. Available: http://arxiv.org/abs/1708.06235

[19] J. Li, L. Wang, J.-J. Brault, and J. Conan, “Mobile Location in MIMO Communication Systems by Using Learning Machine,” in *2007 Canadian Conference on Electrical and Computer Engineering*, Apr. 2007, pp. 1066–1069. doi: 10.1109/CCECE.2007.272.

[20] J. H. Reed, K. J. Krizman, B. D. Woerner, and T. S. Rappaport, “An overview of the challenges and progress in meeting the E-911 requirement for location service,” *IEEE Commun. Mag.*, vol. 36, no. 4, pp. 30–37, Apr. 1998, doi: 10.1109/35.667410.

[21] H. Kwon, J. Lee, and W. Choi, “Machine Learning-Based Beamforming in K-User MISO Interference Channels,” *IEEE Access*, vol. PP, pp. 1–1, Feb. 2021, doi: 10.1109/ACCESS.2021.3058759.

[22] M. Zhao *et al.*, “RF-based 3D skeletons,” in *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*, Budapest Hungary, Aug. 2018, pp. 267–281. doi: 10.1145/3230543.3230579.

[23] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, “SpotFi: Decimeter Level Localization Using WiFi,” in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, London United Kingdom, Aug. 2015, pp. 269–282. doi: 10.1145/2785956.2787487.

[24] L. Zhang, Y. Hua, S. L. Cotton, S. K. Yoo, C. R. C. M. Da Silva, and W. G. Scanlon, “An RSS-Based Classification of User Equipment Usage in Indoor Millimeter Wave Wireless Networks Using Machine Learning,” *IEEE Access*, vol. 8, pp. 14928–14943, 2020, doi: 10.1109/ACCESS.2020.2966123.

[25] A. Wang, G. Chen, J. Yang, S. Zhao, and C.-Y. Chang, “A Comparative Study on Human Activity Recognition Using Inertial Sensors in a Smartphone,” *IEEE Sens. J.*, vol. 16, pp. 1–1, Jun. 2016, doi: 10.1109/JSEN.2016.2545708.

[26] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967, doi: 10.1109/TIT.1967.1053964.

[27] A. McCallum and K. Nigam, “A Comparison of Event Models for Naive Bayes Text Classification,” p. 8.

[28] Z. Chen, Le Zhang, Z. Cao, and J. Guo, “Distilling the Knowledge From Handcrafted Features for Human Activity Recognition,” *IEEE Trans. Ind. Inform.*, vol. 14, no. 10, pp. 4334–4342, Oct. 2018, doi: 10.1109/TII.2018.2789925.

[29] S. M. Ali, N. Gupta, G. K. Nayak, and R. K. Lenka, “Big data visualization: Tools and challenges,” in *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, Dec. 2016, pp. 656–660. doi: 10.1109/IC3I.2016.7918044.

[30] M. T. Rodríguez, S. Nunes, and T. Devezas, “Telling Stories with Data Visualization,” in *Proceedings of the 2015 Workshop on Narrative & Hypertext - NHT ’15*, Guzelyurt, Northern Cyprus, 2015, pp. 7–11. doi: 10.1145/2804565.2804567.

[31] J. Heaton, “An Empirical Analysis of Feature Engineering for Predictive Modeling,” *SoutheastCon 2016*, pp. 1–6, Mar. 2016, doi: 10.1109/SECON.2016.7506650.

[32] “Importance Of Exploratory Data Analysis Before ML Modelling,” *Eduonix Blog*, Sep. 25, 2021. https://blog.eduonix.com/bigdata-and-hadoop/importance-exploratory-data-analysis-ml-modelling/ (accessed Jun. 21, 2022).

[33] X. Gejingting, J. Ruiqiong, W. Wei, J. Libao, and Y. Zhenjun, “Correlation analysis and causal analysis in the era of big data,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 563, no. 4, p. 042032, Jul. 2019, doi: 10.1088/1757-899X/563/4/042032.

[34] R. J. Janse *et al.*, “Conducting correlation analysis: important limitations and pitfalls,” *Clin. Kidney J.*, vol. 14, no. 11, pp. 2332–2337, Nov. 2021, doi: 10.1093/ckj/sfab085.

[35] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, “Handling imbalanced datasets: A review,” *GESTS Int. Trans. Comput. Sci. Eng.*, vol. 30, pp. 25–36, Nov. 2005.

[36] M. Nichat, “Landmark based shortest path detection by using A* Algorithm and Haversine Formula,” Apr. 2013.

[37] E. Acuna and C. Rodriguez, “On Detection Of Outliers And Their Effect In Supervised Classification,” Nov. 2004.

[38] K. Potdar, T. Pardawala, and C. Pai, “A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers,” *Int. J. Comput. Appl.*, vol. 175, pp. 7–9, Oct. 2017, doi: 10.5120/ijca2017915495.

[39] Q. Liu and Y. Wu, “Supervised Learning,” Jan. 2012, doi: 10.1007/978-1-4419-1428-6_451.

[40] C. Hettiarachchi, *Machine Learning: Model and Cost Function*. 2021. doi: 10.13140/RG.2.2.16338.27844.

[41] J. S. Cramer, “The Origins of Logistic Regression,” Social Science Research Network, Rochester, NY, SSRN Scholarly Paper ID 360300, Dec. 2002. doi: 10.2139/ssrn.360300.

[42] D. Lee and J. Lee, “Domain described support vector classifier for multi-classification problems,” *Pattern Recognit.*, vol. 40, no. 1, pp. 41–51, Jan. 2007, doi: 10.1016/j.patcog.2006.06.008.

[43] P. Bühlmann, “Bagging, Boosting and Ensemble Methods,” *Handb. Comput. Stat.*, Jan. 2012, doi: 10.1007/978-3-642-21551-3_33.

[44] R. E. Schapire, “The Boosting Approach to Machine Learning: An Overview,” in *Nonlinear Estimation and Classification*, vol. 171, D. D. Denison, M. H. Hansen, C. C. Holmes, B. Mallick, and B. Yu, Eds. New York, NY: Springer New York, 2003, pp. 149–171. doi: 10.1007/978-0-387-21579-2_9.

[45] S. Džeroski and B. Ženko, “Is Combining Classifiers with Stacking Better than Selecting the Best One?,” *Mach. Learn.*, vol. 54, pp. 255–273, Mar. 2004, doi: 10.1023/B:MACH.0000015881.36452.6e.

[46] Y. Freund and R. Schapire, “Experiments with a New Boosting Algorithm,” *undefined*, 1996, Accessed: Mar. 12, 2022. [Online]. Available: https://www.semanticscholar.org/paper/Experiments-with-a-New-Boosting-Algorithm-Freund-Schapire/68c1bfe375dde46777fe1ac8f3636fb651e3f0f8

[47] N. A. Shashoa, N. Ahmed, I. Jleta, and O. Abusaeeda, “Classification depend on linear discriminant analysis using desired outputs,” Dec. 2016. doi: 10.1109/STA.2016.7952041.

[48] “Linear Discriminant Analysis - an overview | ScienceDirect Topics.” https://www.sciencedirect.com/topics/computer-science/linear-discriminant-analysis (accessed Mar. 12, 2022).

[49] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, Apr. 2006, doi: 10.1007/s10994-006-6226-1.

[50] M. Hossin and S. M.N, “A Review on Evaluation Metrics for Data Classification Evaluations,” *Int. J. Data Min. Knowl. Manag. Process*, vol. 5, pp. 01–11, Mar. 2015, doi: 10.5121/ijdkp.2015.5201.

[51] M. Grandini, E. Bagli, and G. Visani, “Metrics for Multi-Class Classification: an Overview,” *ArXiv200805756 Cs Stat*, Aug. 2020, Accessed: Mar. 14, 2022. [Online]. Available: http://arxiv.org/abs/2008.05756

[52] Ž. Đ. Vujovic, "Classification Model Evaluation Metrics," *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 6, 2021, doi: 10.14569/IJACSA.2021.0120670.

[53] "Welcome to Python.org," *Python.org*. https://www.python.org/ (accessed Mar. 14, 2022).

[54] J. S. M. 2018 417up 1 comment, "What is open source programming?," *Opensource.com*. https://opensource.com/article/18/3/what-open-source-programming (accessed Mar. 14, 2022).

[55] "pandas - Python Data Analysis Library." https://pandas.pydata.org/ (accessed Mar. 14, 2022).

[56] "Matplotlib — Visualization with Python." https://matplotlib.org/ (accessed Mar. 14, 2022).

[57] M. Waskom, "seaborn: statistical data visualization," *J. Open Source Softw.*, vol. 6, no. 60, p. 3021, Apr. 2021, doi: 10.21105/joss.03021.

[58] *Pandas Profiling*. YData, 2022. Accessed: Mar. 14, 2022. [Online]. Available: https://github.com/ydataai/pandas-profiling

[59] "scikit-learn: machine learning in Python — scikit-learn 1.0.2 documentation." https://scikit-learn.org/stable/ (accessed Mar. 14, 2022).

[60] "ipyleaflet: Interactive maps in the Jupyter notebook — ipyleaflet documentation." https://ipyleaflet.readthedocs.io/en/latest/ (accessed Mar. 14, 2022).

[61] "MLflow - A platform for the machine learning lifecycle," *MLflow*. https://mlflow.org/ (accessed Mar. 26, 2022).

10. APPENDICES

Appendix 1.	List of parameters for Logistic regression CV.
Appendix 2.	List of parameters for SVC.
Appendix 3.	List of parameters for KNN.
Appendix 4.	List of parameters for AdaBoost.
Appendix 5.	List of parameters for LDA.
Appendix 6.	List of parameters for Extra Tree Classifier.

Appendix 1. List of parameters for Logistic regression

- Cs=10
- fit_intercept=True
- cv=5
- dual=False
- penalty='l2'
- scoring=’accuracy’
- solver='lbfgs'
- tol=0.0001
- max_iter=100
- class_weight=’balanced’
- n_jobs=1
- verbose=0
- refit=True
- intercept_scaling=1.0
- multi_class='auto’
- random_state=0
- l1_ratios=0

Appendix 2. List of parameters for SVC

- C=10.0
- kernel='rbf'
- degree=3
- gamma=1e-5
- coef0=0.0
- shrinking=True
- probability=True
- tol=0.001
- cache_size=200
- class_weight=’balanced’
- verbose=False
- max_iter=-1
- decision_function_shape='ovo',
- break_ties=False
- random_state=0

Appendix 3. List of parameters for KNN

- n_neighbors=4
- weights='uniform'
- algorithm='auto'
- leaf_size=30
- p=2
- metric='minkowski'
- n_jobs=1

Appendix 4. List of parameters for AdaBoost

- base_estimator= DecisionTreeClassifier
- n_estimators=50
- learning_rate=1.0
- algorithm='SAMME'
- random_state=0

Appendix 5. List of parameters for LDA.

- solver='svd'
- shrinkage=’auto’
- n_components= 3
- store_covariance=False
- tol=0.0001

Appendix 6. List of parameters for Extra Tree Classifier

- n_estimators=100
- criterion='gini'
- max_depth=None
- min_samples_split=2
- min_samples_leaf=1
- min_weight_fraction_leaf=0.0
- max_features='auto'
- max_leaf_nodes=None
- min_impurity_decrease=0.0
- bootstrap=False
- oob_score=False
- n_jobs=None
- random_state=0
- verbose=0
- warm_start=False
- class_weight=None
- ccp_alpha=0.0
- max_samples=None