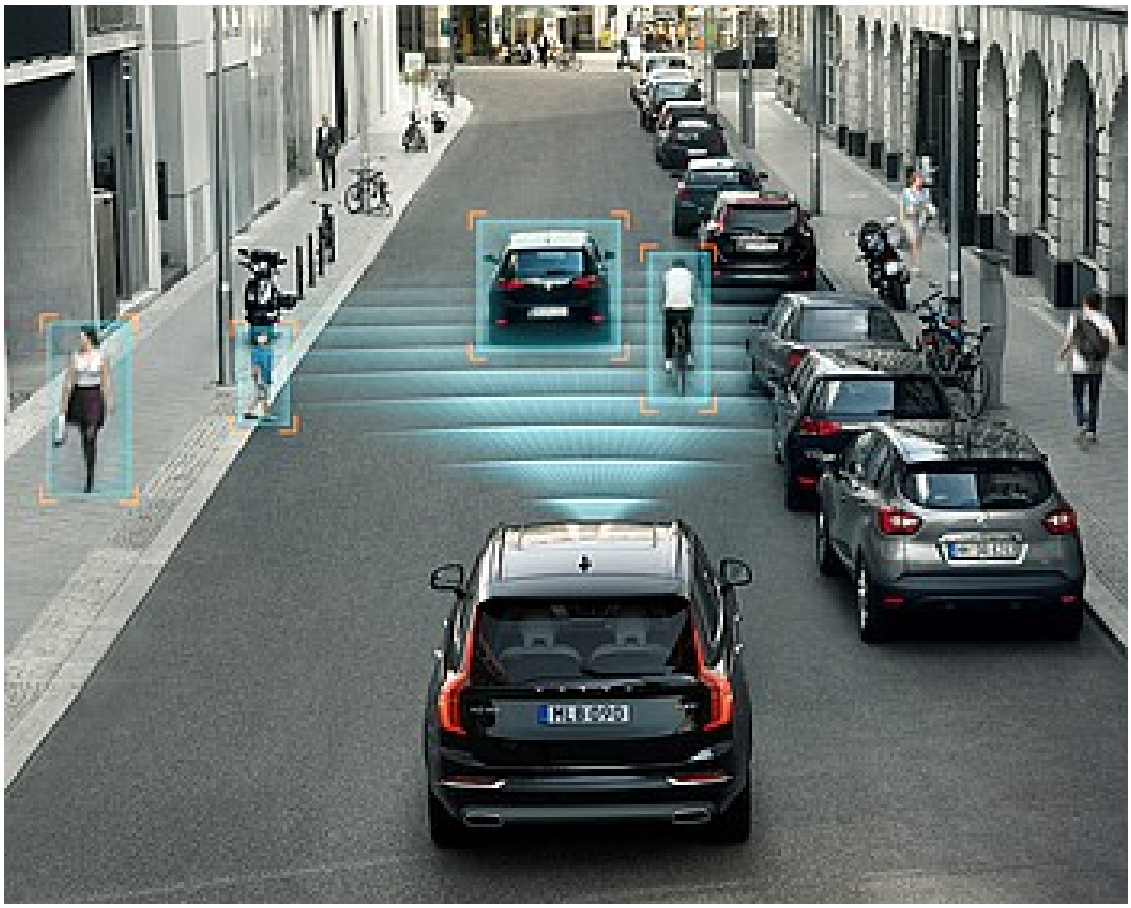
CHALMERS
UNIVERSITY OF TECHNOLOGY



Data Driven Development: Machine Learning in Improving Active Safety Functions Based on Customer Data

Master's Thesis in Systems, Control and Mechatronics

Aditya Pujar & Arshad Nowsath

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2022
www.chalmers.se

Master's thesis 2022

Data Driven Development: Machine Learning in Improving Active Safety Functions Based on Customer Data

Aditya Pujar
Arshad Nowsath

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2022

Data Driven Development: Machine Learning in improving active safety functions using customer data
Aditya Pujar
Arshad Nowsath

Supervisor:	Carina Björnsson	Technical Expert - DA and Active Safety Test Methods, Volvo Car Corporation.
Advisor:	Jinxiang Song	Department of Electrical Engineering, Chalmers University of Technology.
Examiner:	Fredrik Brännström	Head of Communication Systems Group, Department of Electrical Engineering, Chalmers University of Technology.

Master's Thesis 2022
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Represents City Safety technologies in the standard active safety package of the all new Volvo XC90 will include the IntelliSafe technologies with two new safety technologies: the run-off road protection package and auto brake at intersection capability. It will assist the driver when there is a high risk of collision with another vehicle, pedestrian or cyclist

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2022

Data Driven Development: Machine Learning in Improving Active Safety Functions Based on Customer Data
Aditya Pujar, Arshad Nowsath
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The Active Safety department of Volvo Cars Corporation (VCC) has identified false positive events in the vehicle read-outs of data from customer vehicles in the past three years. The false-positive signals play a major role in the automatic braking of the vehicle. The task is to come up with a classification algorithm based on machine learning which can find patterns in data that are classified as a true positive or false positive. In addition, the scenarios or the patterns under which the event is wrongly classified as false positive are to be found to improve AEB (Automatic Emergency Braking) performance.

Keywords: false positive, customer, automatic braking, patterns, classification, algorithm.

Acknowledgements

We would like to give our supervisor at Volvo Cars, Carina Björnsson, and manager Sofia Sköld a special thanks for helping us to acclimate to a new work environment and giving us guidance throughout the thesis, and being always supportive for smooth carrying out of the thesis work.

We would like to give our greatest thanks to our supervisor at Volvo Cars, Andreas Lökholm, for being very invested in helping us achieve our goals in this thesis and for always providing us with valuable information and feedback. This thesis would not have been the same without your help.

We also want to give a big thank you to our Advisor at Chalmers, Jinxiang Song, for giving us great feedback during our meetings and for guiding us forward through this process.

we would like to thank our Examiner at Chalmers, Fredrik Brännström for the academic guidance he provided. We are very grateful for all the academic support you provided us with.

Lastly, we want to give a great thank you to our families and friends, for supporting us throughout this thesis and the rest of our academic journey

Aditya Pujar & Arshad Nowsath, Gothenburg, June 2022

Contents

List of Figures

List of Tables

1

Introduction

1.1 What is active safety?

The purpose of an Active safety system is to give support to avoid or mitigate an accident before it happens sometimes even without human intervention. The term "Active Safety" is being used to describe a system that has an understanding of the state of the vehicle to both avoid and minimize the effects of a crash. These include braking systems, like brake assist, traction control systems, and electronic stability control systems, that interpret signals from various sensors to help the driver control the vehicle. Additionally, forward-looking, sensor-based systems such as advanced driver-assistance systems including adaptive cruise control and collision warning/avoidance/mitigation systems are also considered as active safety systems under this definition [1].

Figure 1.1: Active safety in volvo cars

The impact of Active Safety systems on human life can be better understood by the statistics. The road fatalities in European Union (EU) were 54,900 in 2001 and have been reduced to less than 25,000 in 2020, although around 60 million cars were added on road in the same period [2]. Currently, the number of fatalities per million in the European Union (EU) is only 49 whereas the global average is 174 fatalities per million, proving that the EU has the safest roads in the world. To achieve the vision of zero traffic fatalities in the future, the European Union (EU) automotive industry is heavily investing in research and development of Active Safety Systems to make the vehicles even safer.
Levels of Driving Automation is defined by the Society of Automotive Engineers (SAE)[3]. Currently, Conditional automation of vehicles level 3 is achieved where the vehicle is in full control in some situations, monitors the road and traffic, and will inform the driver when he or she must take control. According to the definition given by SAE, level 4 autonomous driving vehicles should be able to drive for the entire trip in most conditions and at level 5 the vehicle should take full control for the entire trip in all conditions. Therefore, Active Safety System is expected to play an increasing role in collision avoidance and mitigation in the future of the automobile industry to reach autonomous levels of 3, 4, or 5 to ensure the safety of the passengers.

1.2 Problem definition

The Active Safety department of Volvo Cars Corporation (VCC) has identified false positive events in the vehicle read-outs of data from customer vehicles in the past three years. The false-positive signals play a major role in the automatic braking of the vehicle. The task is to come up with a classification algorithm based on machine learning which can identify patterns in data that are classified as a true positive or false positive. By this pattern, the scenarios or the patterns under which the event is wrongly classified as true positive are to be found to improve AEB(Automatic Emergency Braking) performance.

Event: An event in our thesis work refers to the occurrence of auto-braking of the vehicle.

True Positive events -An event where the vehicle correctly predicts the possibility of a collision and decides to auto break.

False Positive events - An event where the vehicle wrongly predicts the possibility of a collision and decides to auto break.

1.3 Goal

The goal is to develop classification algorithms based on machine learning using the Active safety(AS) signals registered on the logger as data that can identify patterns

in the data that are crucial for true/false triggering and can be used on future data. It is desired to determine if the pattern recognition can be done using machine learning for the available dataset and if yes, then find those patterns.

Understanding the available data, preprocessing, applying different algorithms, and tuning the hyperparameters to obtain the best possible results are the challenges faced.

1.4 Data description

The data used for the thesis work is provided by Volvo Cars Corporation(VCC). The Active Safety department of VCC has identified false positive events in the vehicle read-outs of data from customer vehicles in the past three years, from 2017-to 2020. The provided data is based on quarters starting from the year 2017 first quarter to 2020 first quarter(13 quarters in total). During these 13 quarters, VCC has deployed 6 different versions of software. Hence the data is segregated into six sets based on the software version of the vehicle. The folder of each quarter has the subfolders containing CSV files regarding the events that happened in that particular quarter. Along with the event subfolders, there is a CSV file for each quarter, which holds the read-out ids, row numbers, classification type of the event, and many more information related to every event of that quarter. The subfolders are named based on the readout id and row number as mentioned in the CSV file for each quarter.

Each subfolder in a quarter contains data regarding one or more events, depending upon how many events were registered by the particular vehicle readout.

1.5 Limitations

Customer data is limited in sample frequency/time including the number of signals, and thereby not potentially as efficient as development logs, but contains enough information to create an adequate classification method.

The data available for training the model might not be balanced. That is, the number of events corresponding to the two classes (True-Positive, False-Positive) might not be the same. This has to be taken care of while training the particular model.

1.6 Research questions

- How can the events leading to True positive/ False positive be identified using machine learning algorithms?

- Which machine learning algorithm performs better in classifying the events?

- Is it possible to find complex or simple patterns that VCC has not yet easily found? Why/why not? What recommendation do we have for VCC to analyze data better in the future to improve the performance of the AEB system?

1.7 Hardware and software used

For hardware, we use a windows 10 64-bit operating system with an x64-based processor. Intel(R) Core(TM) i5-8365U CPU @ 1.60GHz 1.90 GHz. we used one GPU-enabled virtual machine provided by VCC as well for machine learning model training to reduce the processing time.

For software, we use Python 3.9 integrated with Jupyter Notebook(Anaconda3) for data preprocessing & machine learning models and Tableau 2020.2 for finding additional patterns in given data sets.

1.8 Report outline

The first chapter gives a brief introduction to active safety systems, problem definition, and goal of this thesis work. The second chapter explains more about the data set used for this thesis and elaborates on the importance of data cleansing before starting any analysis of the available data. Since the feature space is huge and the major information about the data is hidden in only a few features, it is important to apply dimension reduction techniques and reduce the feature space. The theory behind dimension reduction techniques are discussed in chapter 3 along with the practical implementation.
After the feature space is reduced, the data associated with the selected features are considered for further analysis. The preprocessing methods on the reduced dataset are discussed in chapter 4. After the preprocessing, various machine learning algorithms used for classification are analyzed in chapter 5. Since the aim is to find out the reason for the occurrence of false-positive events, the pattern recognition techniques are discussed in chapter 6. Finally, chapter 7 summarises the thesis work along with suggestions for future work.

2

Data Cleansing and Preprocessing

2.1 Data cleansing

Dean Abbot, the co-founder and Chief data scientist at Smarter HQ, feels that "No data is clean, but most are useful". Data cleansing is the first and most important process before beginning any further analysis to gain insights about the data. It is the process of identifying the data which is relevant from different files/folders and replacing or modifying it according to the requirement [4]. This process will not tamper with any information in the data, but fetch the relevant data from different sources and put them all together for further analysis. As a machine learning engineer, the challenge is to study and understand the data thoroughly and then decide what data is important and needed for getting useful insights about the data.

Figure 2.1: Data cleansing

2.1.1 Description of available dataset

The data available from Diagnostic Readout(DRO) which is the information in the car available for readout through standardized diagnostic communication protocols which is the data provided to us had 13 DRO files corresponding to 13 quarters and 13 folders, containing data of each quarter. Each DRO file had 77 columns containing information such as vehicle system, event type, host speed, object type, map location, and so on. Every event had an event ID mentioned in the DRO file, along with the buffer number. Event ID and buffer number were important features since the 13 folders containing the data of 13 quarters were named based on event ID. The buffer number signifies the number of events recorded in the readout having a particular event ID. Buffer number in each readout varies from 1 to 3 depending upon the number of events registered in that particular readout.

Every folder with a unique readout ID contained pre1, pre2, single, post1, post2, metadata, statistics, and vehicle global data in the form of comma-separated values (CSV) files. "pre1" and "pre2" are the CSV files containing data of all active safety signals 4 seconds before the occurrence of the event. The data is registered at the frequency of 5 Hz. So the total number of samples saved before the occurrence of the event is 20. "Single" file contains information of active safety signals at the instant of occurrence of the event. Similarly, "post1" and "post2" are the CSV files containing data of all active safety signals 4 seconds after the occurrence of the event. The data is registered at the frequency of 5 Hz. So the total number of samples saved before and after the occurrence of the event is 20. So, Whenever an event of auto braking occurs, 20 samples of active safety signals are logged before and after the event, along with the data samples at the instant of event occurrence. The requirement was to first merge "pre1" with "pre2" and post1 with post2 column-wise and then have the readout ID and classification of the respective event in the respective merged CSV files. All such pre1+pre2, post1+post2 merged files would be merged row-wise for every quarter. There are different software versions in different quarters as mentioned in the vehicle global data". The task is to analyze the data in different software versions separately and build classification algorithms using machine learning and try to find out the patterns in the data collected in different software versions.

2.1.2 Data cleansing using python

The data cleansing is done in Python, using Jupyter notebook. First, the Pandas library is imported. The path for the DRO file of the first quarter is loaded into the Jupyter notebook. Upon loading the DRO file, only the columns "Readout ID", "Buffer number" and "Classification" are extracted and saved in different variables. This information is stored in nested lists named as "list a", each list containing four elements namely, "Year", "Row number", "Buffer number", and "Classification". Next, the path to the folder containing data of the first quarter is loaded into Jupyter notebook. Using the Python package "os", the contents in the directory are listed and assigned a variable name. Since the sub-folders are named based on the year

of occurrence and row number, now the years and row numbers are extracted and saved in nested lists named as "list b". Each list contains two elements namely, "Year" and "Row number". The two nested lists "list a" and "list b" are compared for their respective "Year" and "Row number". Every time these two elements match in the "list a" and "list b", the code gets into a loop to match the "buffer number" from "list a" to fetch the exact event. Once the event identified correctly, the respective "pre1", "pre2", "post1", "post2" and "single" files are loaded into the data frame. "Readout ID", "row number" and "Classification" columns are added to "pre1" and merged with "pre2" column-wise and saved as CSV file. Similarly, ""Readout ID", "row number" and "Classification" columns are added to "post1" and merged with "post2" column-wise and saved as CSV file. "Readout ID", "row number" and "Classification" columns are added to "single" file and saved it as CSV file. The same is repeated for all quarters.

Once the above process was done on all quarters, it was observed that there were 6 different software versions in the given dataset as mentioned in the "vehicle global data" CSV file. So, the quarters having similar software versions were grouped together for further analysis.

The required information from the DRO file was "Readout ID", "Buffer number" and "Classification". The remaining columns in the DRO file were neglected for this analysis. This fetched data from the DRO file was merged with the respective "pre1" and "pre2" merged file, "post1", "post2" merged file, and "single" file.

2.2 Data preprocessing

Data preprocessing is a method of transforming the raw form of data into the machine-understandable format. The data collected in many real-world situations are either incomplete, prone to contain errors, include categorical data, different columns will be having different scales, and so on [5] .

Steps involved in Data Preprocessing are:

Figure 2.2: Steps involved in data pre-processing

2.2.1 Importing libraries

```
from sklearn import datasets
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
%matplotlib inline
from sklearn.preprocessing import scale # Data scaling
from sklearn import decomposition #PCA
import pandas as pd # pandas
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
```

Figure 2.3: Example of importing libraries

The libraries used for data preprocessing are different packages in scikit-learn namely train test split, NumPy, preprocessing, SimpleImputer, standard scalar, and one hot encoder.

2.2.2 Importing data

The data was imported in Jupyter Notebook using Pandas library. The command "pd.readcsv" was used for importing dataset. It is important to note that, there were situations where the file was separated by a semicolon instead of a comma. Also, sometimes the header was made equal to "none" and the first row was neglected in order to read the file in a required form. The data files contain "signals" and "name" as the first row, which needs to be neglected while analyzing the file.

2.2.3 Checking for missing values

The data collected from real-world applications often tend to have missing values. This might be because of loss of information, data corruption, or failure to record data. As many machine learning algorithms fail to analyze the data containing missing values, it is important that the missing values are treated before any further analysis. In our case, less than 2% dataset had missing values and we totally neglect that event for further analysis. There are various ways of handling missing data issues [6].

Classification	signal 1	signal 2	signal 3	signal4	signal 5
TRUE	1.5	100	0.3	12.5	0.5
TRUE	2	68	0.5	13.6	1.1
FALSE	NaN	87	0.2	11.8	1.4
TRUE	2.3	70	0.6	12.8	NaN
FALSE	3.5	96	1.3	13.3	0.65
FALSE	1.4	NaN	0.6	14.2	1.21
FALSE	2.6	84	0.9	12.8	1.3
TRUE	3.2	79	0.3	13.5	0.7

Figure 2.4: Example for missing values in dataset

2.2.3.1 Deleting rows with missing values

Deleting the entire row containing missing values in one or more columns is one way of handling the missing value problem. It works effectively only if the percentage of missing values is less, which may be around 5 percent. The machine learning model trained on data treated with removal of the row having missing values will be robust and gives better results. Hence this method of treating the missing values was employed in this thesis work.

2.2.3.2 Replacing missing values with mean or median

If it is not desired to delete the entire row as the percentage of missing values is considerably high, then the missing values can be replaced by the mean or median of that particular column. This method works well on the smaller dataset, is easy to implement, and prevents the loss of data. On the other hand, this method works well only on a numerical continuous variable. Hence this method was not used in this thesis work.

2.2.3.3 Replacing for categorical columns

If the value is missing from categorical columns, then it can be replaced with the most frequently occurring category. If the percentage of missing values in the categorical column is considerably high, then a new category can be added. Since no such issue was faced in this thesis work, it was not used.

The library "Sci-Kit learn" contains a package called "SimpleImputer". According to the documentation of the Sci-Kit learn library, the SimpleImputer class provides basic strategies for imputing missing values. Missing values can be imputed with a provided constant value, or using the statistics (mean, median, or most frequent) of each column in which the missing values are located. This class also allows for different missing values encoding. It was chosen to neglect the rows with missing values during this thesis work since the dataset was very large and the number of missing data was considerably less.

2.2.4 Checking for categorical data

Since the machine learning model expects only the numeric data for training, testing and validation, there is a need to convert the categorical data into numeric data. "One hot encoding" is one famous method of converting categorical data into the new categorical column and allot a binary value of 1 or 0 to those columns. In this thesis work, "Classification" was the categorical data with labels "True" and "False". This data was one hot encoded suitably to fit the data for applying machine learning algorithms [7].

Classification	One hot encoding	
TRUE	1	0
TRUE	1	0
FALSE	0	1
TRUE	1	0
FALSE	0	1
FALSE	0	1
FALSE	0	1
TRUE	1	0

Figure 2.5: Example for one hot encoding

For the machine learning classification models, as explained in the data cleansing section, only the "Classification" column selected from the DRO file was the categorical data. The one-hot encoding technique was used to convert the True category into 10 and the False category into 01. This was achieved by importing OneHotEncoder from sklearn.preprocessing library.

2.2.5 Feature scaling

Feature Scaling is a technique to standardize the independent variables or features present in the data in a fixed range. feature scaling not only brings all variables in the same scale but also increases the speed of calculations in the algorithm [8]. The most commonly used methods for feature scaling are,

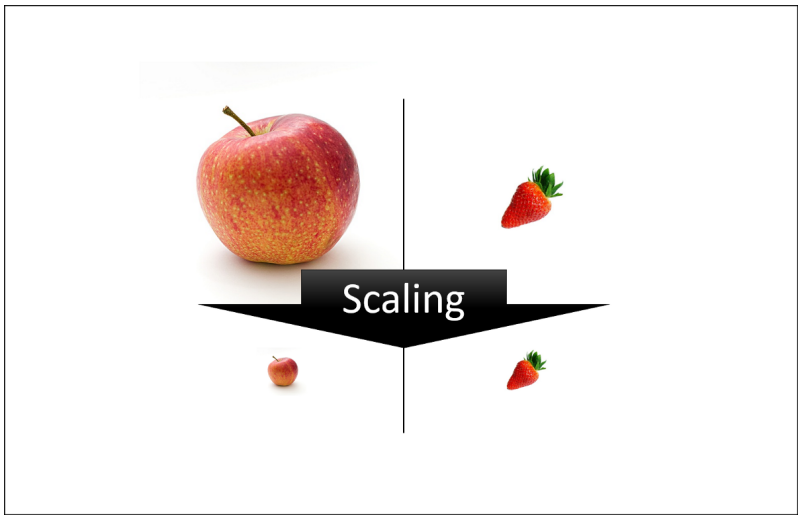


Figure 2.6: Example for feature scaling

2.2.5.1 Normalization

It is a technique where the range of features is re-scaled to the range [0,1]. The formula for normalization is given by,

$$x^{'} = \frac{x - min(x)}{max(x) - min(x)} \tag{2.1}$$

max(x) is the maximum value and min(x) is the minimum value of the feature. The range can also be other than [0,1] depending on the application. Normalization is a preferred way of feature scaling when the data does not follow a Gaussian distribution. This technique is also preferred for data normalization in image processing. In our thesis work, the Normalization technique is used in the low variance filter method, which is one of the dimension reduction techniques as explained in the previous chapter under section 3.1.0.2.

2.2.5.2 Standardization

Standardization is the feature scaling technique used when it is desired to transform the data to have zero mean and unit variance. The formula used to calculate the value of each data point in a feature after calculating standard deviation and mean is,

$$x^{'} = \frac{x - \bar{x}}{\sigma} \tag{2.2}$$

Where $\bar{x}$ is the mean of the feature and σ is the standard deviation of the feature. Standardization is usually preferred when the data follows a Gaussian distribution, which need not be necessary all the time. Even if there are outliers in the data, they won't be affected by standardizing the data. Also, Standardization is the best-suited feature scaling technique when using Principal Component Analysis, since the focus is on the components that maximize the variance. In the previous chapter

where dimension reduction techniques are discussed, Principal Component Analysis is discussed in detail.

out of all the available feature scaling techniques, Standardization is used for this thesis work, as explained above. According to the documentation of the scikit-learn library, Standardization of data sets is a common requirement for many machine learning estimators implemented in scikit-learn; they might behave badly if the individual features do not more or less look like standard normally distributed data: Gaussian with zero mean and unit variance. The preprocessing module provides the StandardScaler utility class, which is a quick and easy way to perform the Standardization of every feature in the data sets.

2.2.6 Splitting the data set

Separating data into training, validation, and testing sets is an important part of evaluating machine learning models. The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. The train-test split procedure is appropriate when the dataset is very large and requires a good estimate of model performance quickly [9].

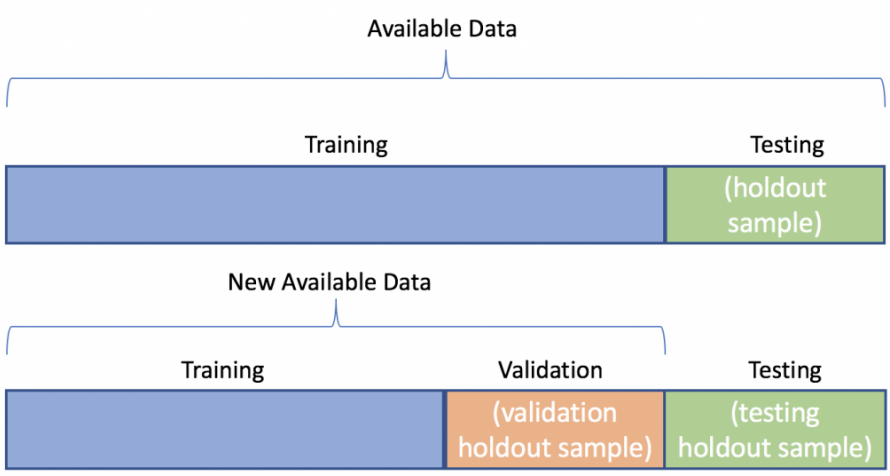


Figure 2.7: Example of splitting the data

From sklearn.model selection library, train test split is imported. The data is split into training, validation, and test data randomly in the ratio of 0.6, 0.2, 0.2 respectively. The training data is only used to train the machine learning models and the validation set is used to validate the models. Once the training and validation are done satisfactorily, the model can be tested on the test data and the performance matrices can be used to analyze the performance of the model. There are various performance models depending upon the application and whether the dataset is balanced or not.

2.3 Conclusion

At the end of data preprocessing, the data was free from missing values, all categorical data were one-hot encoded, all the features in the dataset were standardized to have zero mean and unit variance. The dataset was split into training, validation, and test sets to feed them to the machine learning algorithms.

3

Dimension reduction techniques

Criag Mundai, senior advisor to the CEO at Microsoft has rightly mentioned that " Data are becoming the new raw material to the business ". Every industry is moving towards automation involving more and more sensors, increasing the amount of data collected every single day. Not all the data collected can be useful for analyzing the trend in the business or finding a pattern in the way the system behaves. In order to get meaningful insights from such data sets, their dimensional should be significantly reduced in a way that most of the information in the data is still preserved. Many dimension reduction techniques have been developed for this purpose, most widely used techniques are Principle Component Analysis, Low variance filter method, forward feature selection method, and backward feature elimination method.

3.1 Principle component analysis(PCA)

Principal component analysis (PCA) is one of the oldest and most widely used dimension reduction techniques where the number of features of a dataset is reduced while preserving as much 'variability' (i.e. statistical information) as possible [10].

3.1.1 Terminology in PCA

Dimensions/ Feature space: It is the number of features in dataset.

Correlation: It is the parameter that indicates how strongly two variables are related to each other. The value of co-relation varies between -1 to +1. Positive correlation implies that the increment in one feature results in an increase in the other as well. While the negative correlation indicates that if one feature decreases, the other one increases.

Orthogonal: The features are uncorrelated to each other indicating that the correlation between any pair of variables is 0.

Eigen vectors: If v is a non-zero vector and A is a square matrix, then the vector v is regarded as an eigen vector of A, if Av is a scalar multiple of v.

$$Av = \lambda v \tag{3.1}$$

3.1.2 How does PCA work?

The idea behind principal component analysis (PCA) is to select only those features in the dataset which are correlated with each other while retaining the variation present in the dataset up to the maximum extent to make sure the essence of the data is still preserved. This is achieved by computing principal components which new set of variables obtained by transforming the original features. Hence the variance of data in the direction of the first principal component is maximum, retaining the maximum variation that was present in the original components. The principal components are nothing but the eigenvectors of a covariance matrix and are orthogonal to each other.

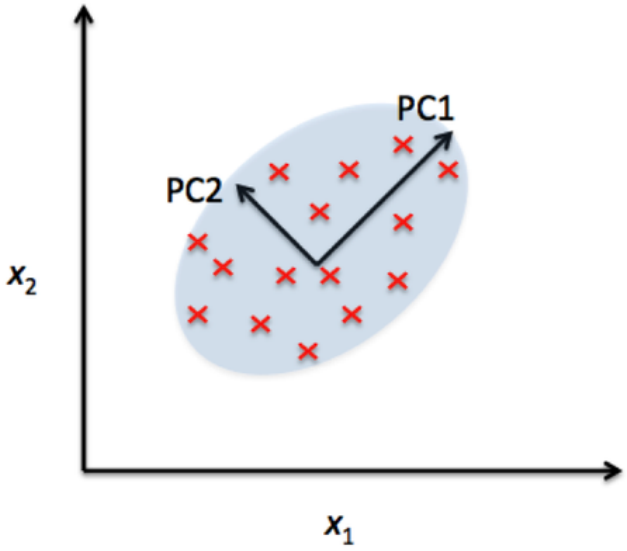


Figure 3.1: Principle component analysis

3.1.3 PCA using python

The principle component analysis was implemented in Python using Jupyter notebook. The libraries used as Sci-kit learn, Pandas and matplotlib. The PCA is performed separately on pre-data and post-data. For the analysis of pre-data, the path to the pre-file is loaded into the notebook. Using Pandas, the file is loaded into the data frame. The columns "Readout ID", "Buffer number", "Row number" are dropped from the data frame since they are not needed for the analysis. The "Classification" column is also dropped since PCA is an up-supervised dimension reduction technique. The remaining data is now applied with standardization technique, to convert all the features into data having zero mean and unit variance.
The Sci-kit learn library has an inbuilt package for PCA under "sklearn.decomposition". The PCA will produce the loadings for all principle components. The number of principle components is equal to the number of features available in the data. In our case, the total number of signals in the pre-file was 266, hence there were 266 principle components observed. When the explained variance for the pre-data was

observed, 80 percent of the data was covered by the first 40 principle components, as shown in the image.

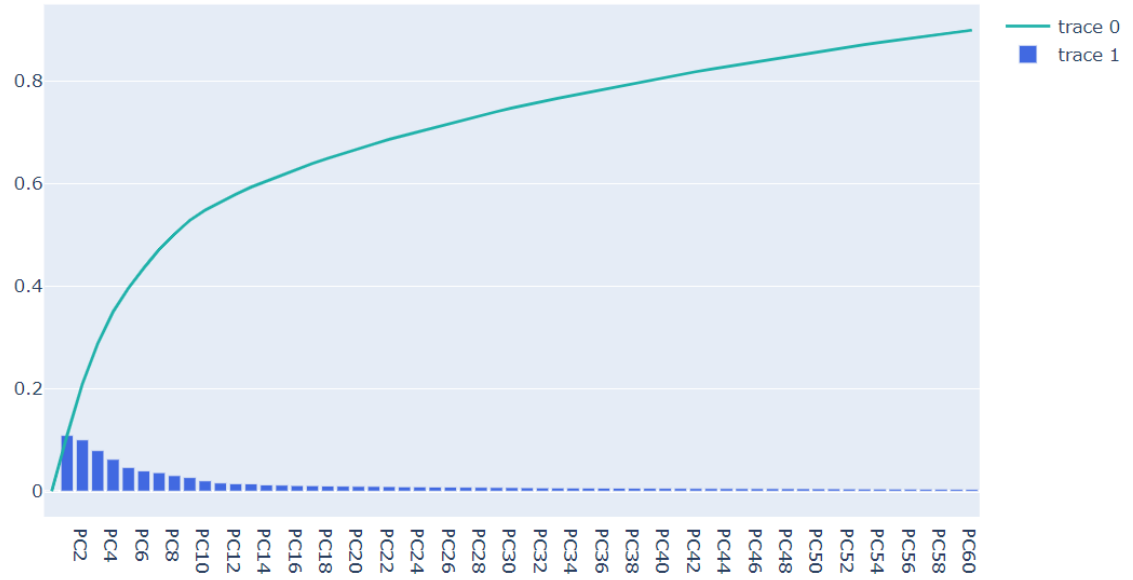


Figure 3.2: explained variance Vs principle components

Thus, the loading's for the first 40 principle components were analyzed and the signals having the largest loading value in each principle component were selected.

3.2 Low variance filter method

A low variance filter is another dimension reduction technique that depends on the variance of data available in each column. The major difference between PCA and low variance filter method is, the data is subjected to normalization before calculating the variance unlike the standardization method in PCA. This is due to the fact that the low variance filter method wants to preserve the variance of the original data but standardization will force the data to have zero mean and unit variance [11]. In case a particular feature contains a constant value, the variance of such feature would be zero and that feature would be of no help in finding a pattern in the data to classify it into different groups. As a result, the Low Variance Filter method computes the variance of each feature and removes those features with a variance value below a selected threshold.

The variance of a feature can only be computed for numerical columns. Hence, this method of dimension reduction can be applied only to features containing numerical values. Hence, preprocessing plays an important role if there are categorical variables. Also, the variance of a feature depends on the range of numerical data. Therefore the range of data needs to be normalized before computing their variance. By doing so, variance values will be made independent from the column domain range.

3.2.1 Low variance using python

The Low variance filter method was implemented in Python using Jupyter notebook. The libraries used as Sci-kit learn, Pandas and matplotlib. The Low variance filter method is performed separately on pre-data and post-data. For the analysis of pre-data, the path to the pre-file is loaded into the notebook. Using Pandas, the file is loaded into the data frame. The columns "Readout ID", "Buffer number", "Row number" are dropped from the data frame since they are not needed for the analysis. The "Classification" column is also dropped since the method is purely based on the variance of each feature and does not depend on the classification label. The remaining data is now applied with the normalization technique instead of standardization since the intention is not to tamper with the original variance value of each feature.

The threshold for the low variance filter method was set by trial and error, to make sure that the signals with top 50 variance values are selected for further analysis.

3.3 Conclusion

The result of dimension reduction is that the features containing the maximum information in the dataset are detected and the data associated with only the selected signals are considered for further analysis. By making use of dimension reduction techniques, the feature space is reduced from 266 to less than 50. This prevents overfitting of machine learning models and helps in better training of model resulting in getting more accurate classification results.

There were 38 signals from the pre-file, 48 signals from the post-file suggested by VCC as important signals during the thesis work. We found that, out of 70 post signals selected by the PCA method, 28 were matching with the post signals suggested by VCC. 24 out of 70 pre signals from the PCA method were matching with pre signals suggested by VCC.

We also found that, out of 75 post signals selected by the low variance filter method, 25 were matching with the post signals suggested by VCC. 29 out of 70 pre signals from the low variance method were matching with pre signals suggested by VCC.

These two dimension reduction techniques are purely based on "how widely the data is spread in each feature". In other words, if the range of spread or the difference between the minimum and maximum value of a feature is high, it has a high variance and that feature will be selected. While applying dimension reduction techniques, the data is analyzed by the two methods and the output of both the methods were the names of the features which are statistically significant. The data related to Only those features were fetched by the code and this subset of the original data was used for further analysis. Hence the dimension reduction techniques did alter

the values of features from the original data.

There are possibilities of selecting wrong signals and neglecting important signals since it purely depends upon their variance. Thus these dimension reduction techniques can not be completely trusted and it is better to have the domain-specific knowledge or discuss the results with the domain expert and then remove the signals with a high variance that do not make sense at all and include the signals suggested by the domain expert.

If the signals are selected purely based on these two methods, possibilities are few less significant signals might be selected and a few important signals might be neglected. We do not have any measure to quantify the effect of selecting less important signals and/or neglecting significant signals on pattern recognition. Careful scrutiny of signals suggested by these methods by a domain expert is suggested.

4

Convolutional neural networks in classification

4.1 Convolutional neural networks (CNN)

4.1.1 Terminologies used in CNN

Input layer: The layer through which the data is fed into the network.

Convolutional layer: The layers where kernels are applied to the original image and features are extracted.

Pooling layer: The layers which reduce the dimensions of the feature maps.

Fully connected layer: The output from the final Pooling layer will be the input to the fully connected layer. This is used to flatten the data and then feed it into the deep neural network.

Output layer: The layer through which the network comes up with the decision regarding the input.

A Convolutional neural network (CNN) has one or more convolutional layers, pooling layers, fully connected layers followed by a deep neural network. CNN's are used in image recognition and processing that are specifically designed to process pixel data. The application of CNN's is mainly in image processing, classification, and segmentation [12].

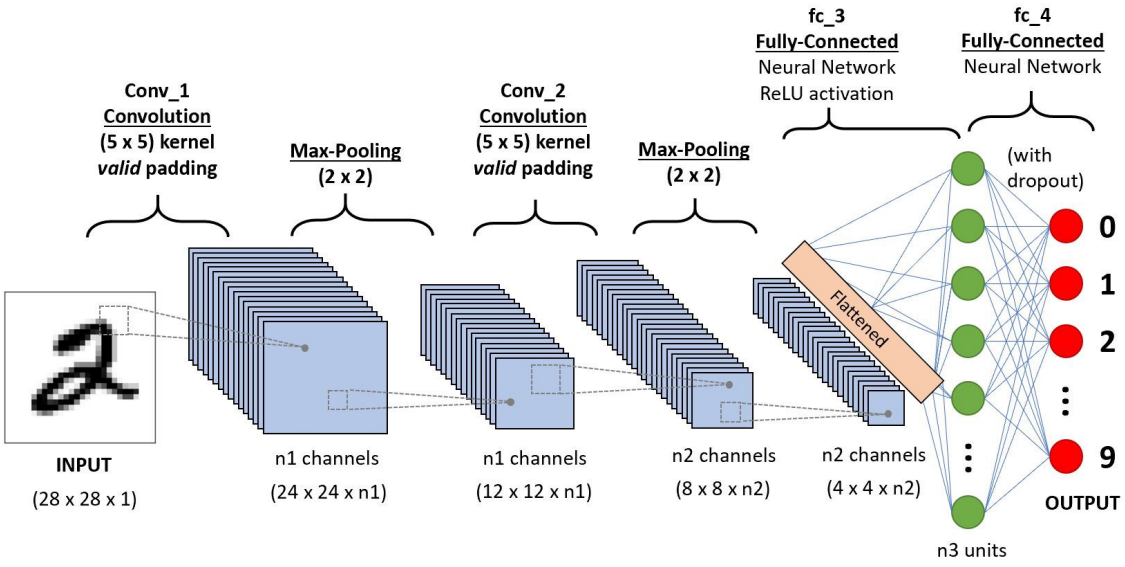


Figure 4.1: Example of a convolutional neural network

4.2 CNN for classification using python

Since the Convolutional Neural Networks perform better at recognizing patterns and classifying the images into different groups after training the model with the training dataset, the idea was to use CNN's to build a model to classify events into truly positive and false positive. The task was to convert the CSV file of each event into an image with respective True-positive and False-positive labels, build the CNN model by choosing the appropriate hyperparameters, train and validate the model and test it on the new data which was not used for training. The following steps were involved:
Checking for missing values: Since the data was already preprocessed in the previous step, there were no missing values found.

Normalizing the data: The numeric data of each event had to be normalized between 0 to 255. It was desired to convert each of the event CSV files into an 8-bit color image where the maximum number of colors that can be displayed at any one time is 256. Using the MinmaxScaler package, the data was normalized between 0 to 255.

Figure 4.2: The image obtained by converting CSV file into 8-bit image

Building and training the CNN model: Using the ImageGenerator package, the hyperparameters "target size" and "classes" were fed into the model. The batch

size was selected to be 30.

4.3 Results of classification using CNN

The model was trained and validated on the images obtained after converting every event data from the CSV file into an 8-bit image. The accuracy of the model was around 50 percent. The performance of CNN was poor and the trained model completely failed to classify the events into True-positive and False-positive.

This poor performance of CNN can be explained as follows.

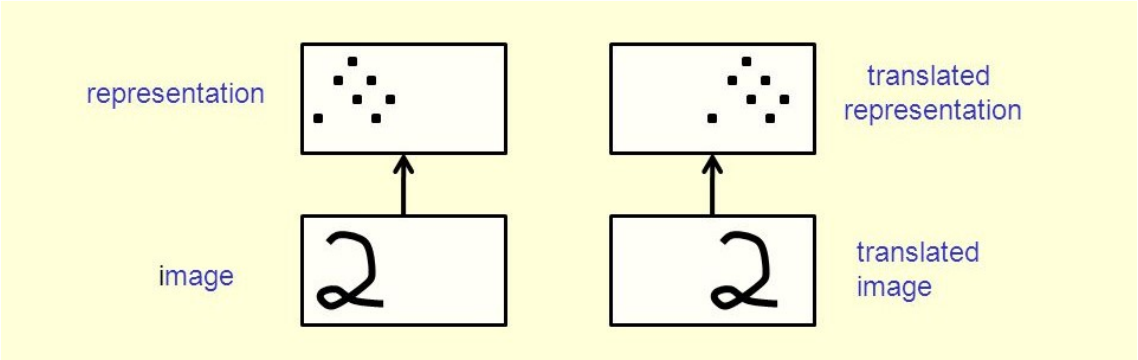


Figure 4.3: Example of translation invariance in CNN

Convolutional Neural Networks are designed to be translation invariant. Meaning, CNN's are not sensitive to the position of an object in the image. Even if the input image is translated, the CNN can still detect the class to which the input belongs to [13].

When the event data was converted from CSV files to images, the images from different events were behaving as if they were translated images of a single image. There was no proper decision boundary for CNN to learn the True-positive and False-positive events and understand the difference. Hence, using CNN to train the model to classify the events into True-positive and False-positive was not successful.

The rule of thumb is to use images of at least 256x256 pixels for CNN's for better results. The images used in our work were 20x266, without dimension reduction. If the dimension reduction was applied to original data and then the data with only selected signals were converted into images, it would be of an even smaller size. Hence, this technique of converting data into images might give better results with more pixels, but not recommended for any future works with similar data. CNN's are used for object detection, image segmentation, and pattern recognition on images with high resolution.

5

Machine learning in classification

A common job of machine learning algorithms is to recognize objects and being able to separate them into categories. This process is called classification, and it helps us segregate vast quantities of data into discrete values, i.e.distinct, like 0/1, True/False, or a pre-defined output label class. Machine learning is divided into three different types of algorithms as follow:

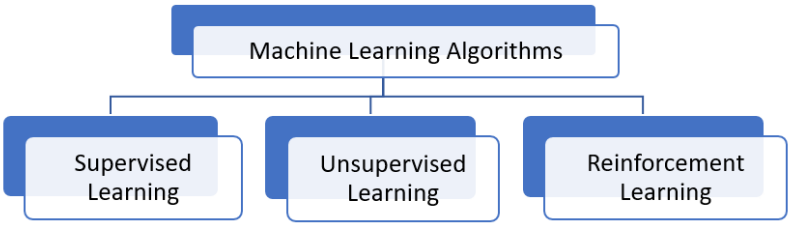


Figure 5.1: Types of machine learning algorithms

5.1 What is classification in machine learning

Before we dive into Classification, let's take a look at what Supervised Learning is. In a supervised technique where a computer program learns from the given dataset and gives us some new observations or classifications[15]. Suppose you are trying to learn a new concept in maths and after solving a problem, you may refer to the solutions to see if you were right or not. Once you are confident in your ability to solve a particular type of problem, you will stop referring to the answers and solve the questions put before you by yourself.

This is also how Supervised Learning works with machine learning models. In Supervised Learning, the model learns by example. Along with our input variable, we also give our model the corresponding correct labels. While training, the model gets to look at which label corresponds to our data and hence can find patterns between our data and those labels.

Some examples of Supervised Learning include:

- It classifies spam Detection by teaching a model of what mail is spam and not spam.

- Speech recognition where you teach a machine to recognize your voice.
- Object recognition by showing a machine what an object looks like and having it pick that object from among other objects.

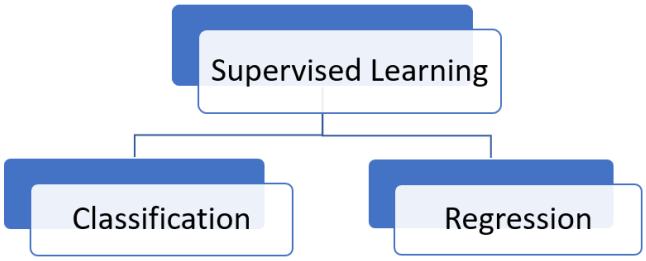


Figure 5.2: Types of supervised learning

Classification is a supervised technique to categorize our training dataset into a desired and distinct number of classes where we can assign a label to each class. It can be either performed on both structured and unstructured data. The process starts with predicting, recognizing, understanding, and grouping the class of a given dataset into preset classes. The classes are often referred to as target, label, or categories[16].

The classification predictive modeling is the task of approximating the mapping function from input variables to discrete output variables. The main goal is to identify which class/category the new data will fall into. In other words, classification can also be used for pattern recognition to find some patterns (True positive or False positive events) in training data sets. Also, we will explore more classification algorithms in detail and see how they perform based on some classification metrics.

5.2 Classification terminologies in machine learning

To avoid some confusion due to ambiguity, here are some of the terms that will be used in upcoming sections:

Classifier	It is an algorithm that is used to map the input data to a specific category.
Classification model	The model predicts or draws a conclusion to the input data given for training, it will predict the class or category for the data.
Feature	A feature is an individual measurable property of the phenomenon being observed.
Binary classification	It is a type of classification with two outcomes, for eg – either true or false.
Multi-class classification	The classification with more than two classes, in multi-class classification each sample is assigned to one and only one label or target.
Multi-label classification	This is a type of classification where each sample is assigned to a set of labels or targets.
Initialize	It is to assign the classifier to be used for the model to be trained
Train the classifier	Each classifier in sci-kit learn uses the fit(X, y) method to fit the model for training the train X and train label y.
Predict the target	For an unlabeled observation X, the predict(X) method returns predicted label y.
Evaluate	This basically means the evaluation of the model i.e classification report, accuracy score, etc.

Table 5.1: Classification Terminologies in Machine Learning

5.3 Classification algorithms

From the literature review, we conclude that the following five algorithms are good at classification problems hence the selection.

5.3.1 Logistic regression

Logistic Regression, the word itself termed as regression but it performs classification based on regression and classifies dependent variables into either of the class. It is a model which makes predictions using a logistic function to find dependency between output and input variables[17]. That's why this model is called a significant machine learning algorithm since it can give the probabilities and able to classify new data by using the discrete and continuous dataset, this is in contrast to linear regression but it doesn't require a linear regression between dependent and independent variables. The name "Logistic Regression" is derived from the concept of a logistic function and is used for solving classification problems. The value of logistic function varies from 0 to 1 as an "S" shaped logistic function[18]. This function is also known as the sigmoid function. the curve in the logistic function from Figure 5.3 represents the likelihood of whether the event is True positive or False positive

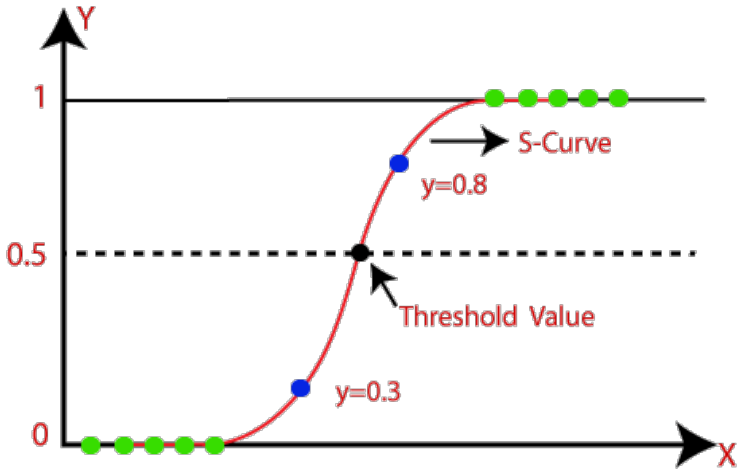


Figure 5.3: Sigmoid logistic function

Logistic regression equation: This can be obtained from the Linear regression equation. The steps are as follow:

- the equation of straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + ... + b_nx_n \tag{5.1}$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y} \tag{5.2}$$

- where 0 for y = 0 and ∞ for y = 1, but we need $-\infty$ to $+\infty$, so we can take log of this equation:

$$log[\frac{y}{1-y}] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + ... + b_nx_n \tag{5.3}$$

This equation 5.3 can be considered as a Logistic regression equation and can do further analysis. There are three types of logistic regression based on the categories:

- **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, True or False, etc.
- **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable.
- **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables

Some assumptions for logistic regression are:

- The dependent variable must be categorical.
- The independent variable should not have multi-co-linearity.

5.3.2 k-nearest neighbors (K-NN)

A popular supervised machine learning algorithm is used for solving a classification problem. k-NN stands for "k-Nearest Neighbors" and it is one of the simplest classification algorithms which is used to identify the data points that are separated into several classes to predict the classification of a new sample point. k-NN is also called the non-parametric, lazy learning algorithm. To understand k-NN better here is an example Fig 5.4, In supervised learning where all the data points were labeled let's say 'category A' and 'category B', and then to find a label to a new data point, it looks into the label points closest to that new point also called nearest neighbors. Based on the neighbor's vote whatever the label neighbor has, will be the label for the new data point. The number K refers to the number of neighboring objects/-points in the n-dimensional space and that will be compared with the object/points that are classified.

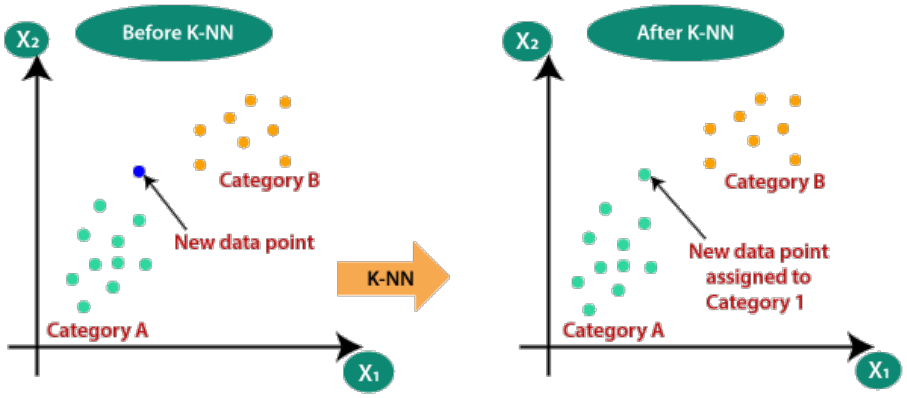


Figure 5.4: How k-NN algorithm works

Honestly, k-NN is not a learning algorithm, it is just classified based on similarity of the features which is input data[19]. Classification is computed from a simple majority vote of the k nearest neighbors of each point.

The working of k-NN can be explained in the following steps:

- **To determine the right value of 'k'**: The process of selecting the right k value is called parameter tuning, it is one of the important steps to achieve higher accuracy. There is no particular way or rule to find the value of k, it all depends on what type of problem we are going to solve. The recommendation is always to go with an odd value especially five why because if we choose one or two that can be noisy and lead to outliers in the model, which results in over-fitting and odd value of k is to avoid confusion between two classes of data we have.
- **To calculate the Euclidean distance of K number of neighbors:** From the Figure 5.5 we can calculate the euclidean distance between two data points A and B as follow

$$EuclideanDistance = \sqrt{(X_2^2 - X_1^2)^2 + (Y_2^2 - Y_1^2)^2} \tag{5.4}$$

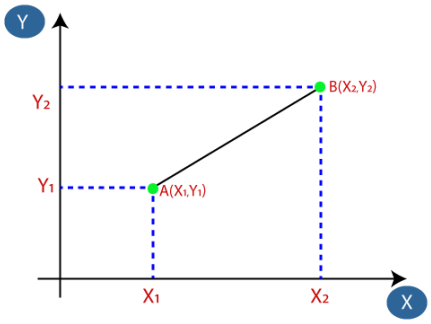


Figure 5.5: Distance between two points A and B

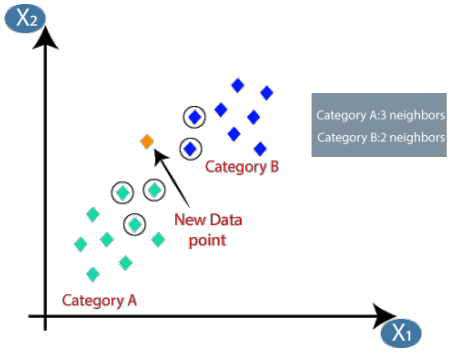


Figure 5.6: Choosing the nearest neighbors from category A and category B

- **To count the number of the data points in each category:** From the Figure 5.6 we can see that 3 nearest neighbors from category A and 2 nearest neighbors from category B.
- **labeling the new data point:** Based on the maximum number of neighbors from both the category, we can see that category A has 3 nearest neighbors so the new data point belongs to category A.

This algorithm is quite simple in its implementation and is robust to noisy training data. Even if the training data is large, it is quite efficient[14]. The only disadvantage with the k-NN algorithm is that there is no need to determine the value of K and computation cost is pretty high compared to other algorithms. Initially, the k value was assumed to be 0 to 30 then compared with the error rate. After k = 20 the error rate was stable and low, so we decided to go with 20.

5.3.3 Decision trees

A decision tree algorithm is a supervised machine learning algorithm which is a tree-structured classifier where each internal nodes represent a feature (predictor variable), the link or branches between the nodes represent a decision rule, and the leaf nodes represent the outcome (response variable). The decision or root node is used to make any suggestions and will have multiple branches. The leaf or terminating node represents the output and will not contain any further branches. From the figure, we understand that decision trees are the graphical representation for finding the best possible solutions to a problem given with the conditions. The CART algorithm (Classification and Regression Algorithm) is used to build the trees and this decision tree is used for both categorical (Yes/No) and numerical data. This algorithm breaks down a larger dataset into smaller subsets and an associated decision tree is incrementally developed simultaneously[20]. The decision tree is similar to the human brain as it looks for all possible outcomes for a problem. There are two types of decision trees:

- Categorical variable decision tree
- Continuous variable decision tree

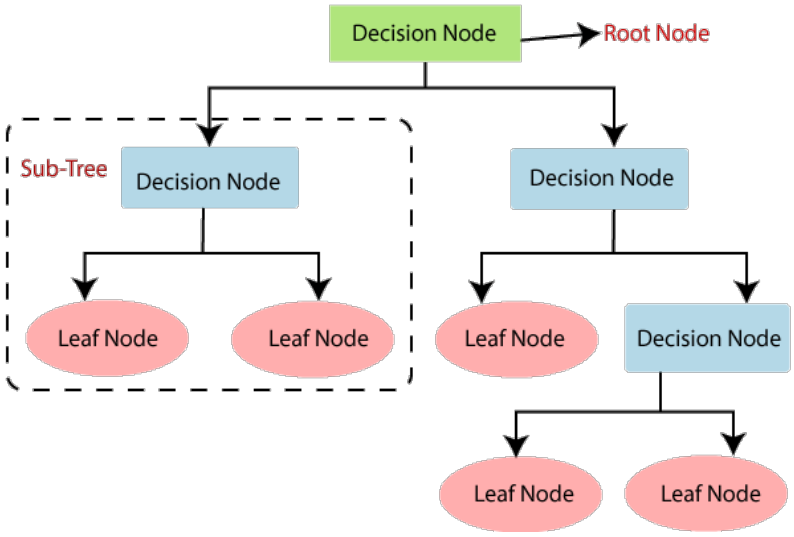


Figure 5.7: Structure of decision tree

The working of decision tree is as follows:

Step 1: The tree begins with the root node and has the entire dataset.

Step 2: The best attribute which effectively separates the data into different classes or the feature (predictor variable) that best splits the dataset is selected using ASM.

Step 3: Divide the root node into subsets based on the decision rule we have on each node and generate the decision tree node.

Step 4: Subsequently make new decision nodes using the subsets until the stage where we cannot further classify the nodes and the final node is called a leaf node.

Attribute selection measures (ASM): The main goal of decision trees is to find the best attribute for the root node and the decision nodes. This can be done with the Attribute selection measures (ASM) technique. The two main measures that select the best attribute are:

- Information gain
- Gini Index

Information gain: The information gained indicates how much information does the feature in the dataset provides about the class. Based on this value, we split the node and build a decision tree. It is the measurement of changes in the entropy after splitting the dataset based on attributes. The variable with the highest information gain will separate the data at the root node.

$$InformationGain = Entropy(S) - [(WeightedAvg) * Entropy(eachfeature)] \tag{5.5}$$

Where entropy is the measure of randomness or impurity in data. It can be calculated as:

$$Entropy(s) = -P(yes)log2P(yes) - P(no)log2P(no) \tag{5.6}$$

where,
S = Total number of samples
P(yes) = probability of yes
P(no) = probability of no

Gini Index: It is the measurement of impurity when creating a decision tree. Only the attribute with a low Gini index Is preferred when compared to a high Gini index and it is used for binary splits.

$$GiniIndex = 1 - \sum_{j} P_j^2 \tag{5.7}$$

The disadvantage of the decision tree algorithm is overfitting thereby reducing the test accuracy. It can be resolved by the technique called pruning which decreases the size of the decision tree without losing accuracy. Thereby we get an optimal decision tree by deleting the unnecessary nodes. The two main types of pruning are:

- Cost complexity pruning
- Reduced error pruning

Another common issue that arises is that it becomes unstable when the small variations in the data can result in a completely different tree being generated.

5.3.4 Support vector machine

Support vector machine (SVM) is a supervised machine learning algorithm used for classification and regression problems. The main objective of SVM is to create the best line or decision boundary called the hyperplane that can segregate n-dimensional space into classes[21]. To create this hyperplane, SVM chooses the extreme points/vectors and these points are called support vectors.

There are two types of SVM:

- Linear SVM: It is used when a dataset can be classified into two separate classes by a single straight line for linearly separable data and this classifier is known as Linear SVM classifier.
- Non-linear SVM: It is used when a dataset cannot be classified by a single straight line for non-linear data and this classifier is known as a Non-Linear SVM classifier.

Hyper plane: The dimensions of the hyperplane depend on the dataset features i.e. when we have 2 features in the dataset, the hyperplane will be a straight line. And when we have 3 features, the hyperplane will become a 2-dimensional plane. The maximum distance between the data points means the maximum margin hyperplane will always be created.

Case 1: For example, consider the following steps:

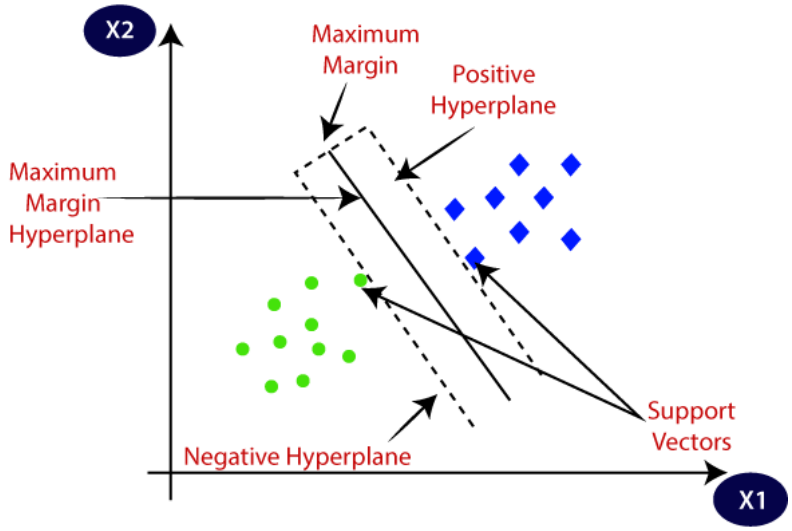


Figure 5.8: Structure of SVM

- If we have a dataset containing two tags (green and blue) and features (x1 and x2).
- We now want a classifier (best line) to classify this pair (x1, x2) in either green or blue.

As it is a 2-d space, a single straight line can easily separate two classes. If we have multiple lines that can separate these classes, their SVM algorithm is used to find the best decision boundary or hyperplane. It can be done by finding the closest point of the lines from the classes. These points are known as support vectors and the SVM must maximize the distance between the vectors and the hyperplane. The hyperplane with the maximum margin is called the optimal hyperplane.

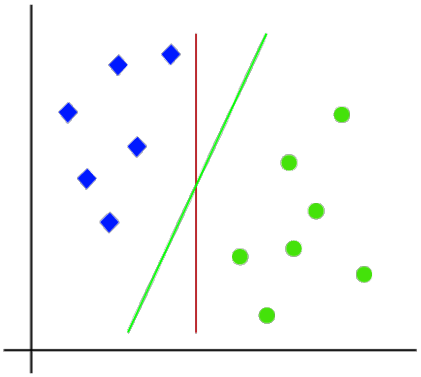


Figure 5.9: Multiple line that separate two classes

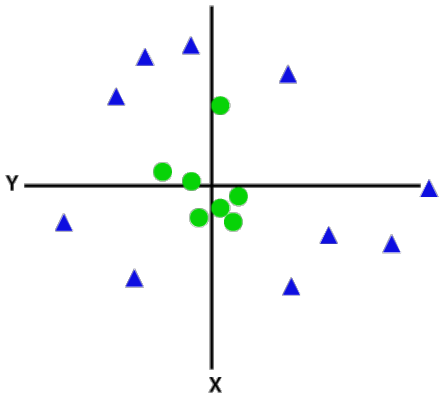


Figure 5.10: Non-linear data distributed in 2D space

Case 2: For linear data, we have two dimensions x and y so we can separate by using a straight line. For non-linear data as shown in Figure **??**, we cannot do like that so one more dimension z is added.

$$Z = x^2 + y^2 \tag{5.8}$$

So the third dimension was added and the samples will look like this in Figure **??**

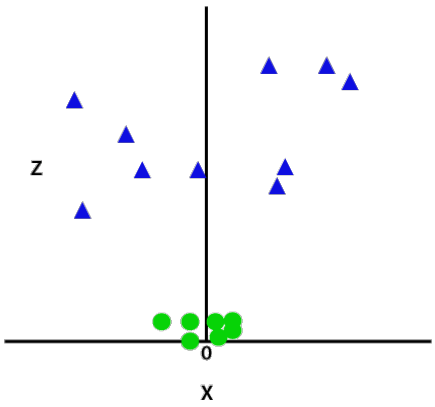


Figure 5.11: Non-linear data after adding 3^{rd} dimension Z

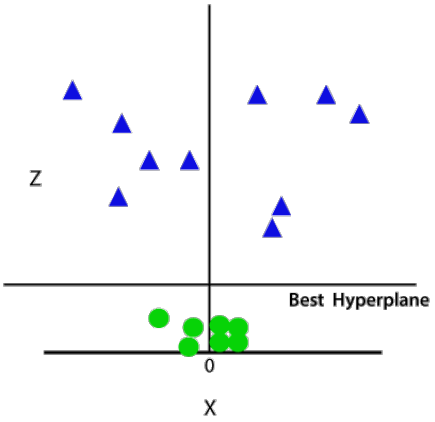


Figure 5.12: Hyper-plane that divide the data sets into classes

Now SVM divide the data sets into classes as follows in Figure 5.17
Since we are in 3D Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as:

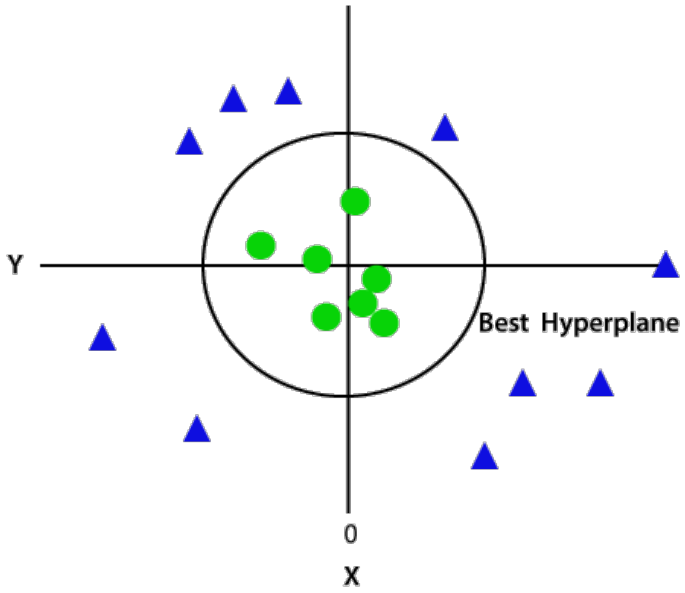


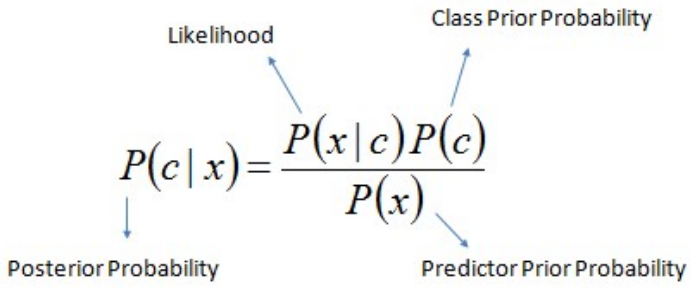
Figure 5.13: Hyper-plane in 2D that divide the data sets into classes

The SVM algorithm has a kernel tool which is a function used to map the non-linear dataset into a higher dimensional space where we can find an optimal hyper-plane that can separate the classes. Based on the labels and outputs we defined, it does extremely complex transformations and makes it easier to classify the data thus it could be linearly divided by a plane[22].

5.3.5 Naive bayes

The Naive Bayes algorithm is based on the Bayes theorem which we can apply to classify data based on historical results. Bayes theorem is used to determine the probability of an event based on prior knowledge of conditions related to that event. The Naive bias classifier works with the assumption of independence between predictors which means it assumes the presence of a feature in a class even it is unrelated to any other features and all these properties individually contribute to the probability. It can be used for both binary and multi-class classifications[23]. It is extremely fast and easy compared to other ML algorithms and requires only a small amount of training data for the necessary parameters to be estimated and to give a valid prediction.

Bayes' Theorem: It is also known as Bayes' Rule or Bayes' law which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability. The formula for Bayes' theorem is given below:



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Figure 5.14: Naive bayes theorem

where,
P(C|X) is posterior probability = probability of C given that X has occurred.
P(X|C) is likelihood probability = Probability of evidence if hypothesis probability is true.
P(C) is Prior probability = Probability of hypothesis before observing the evidence.
P(X) is Marginal probability = Probability of evidence.

Types of Naive Bayes Algorithm:

- **Gaussian Naive Bayes classification:** If the features possess continuous values instead of discrete, the model assumes that this is a normal distribution and the values are sampled from the Gaussian distribution.
- **Multinomial Naive Bayes classification:** It is used when the data is distributed multinomially. Thus, the multinomial Naive Bayes classifier uses the

frequency of words for the predictors, and it is primarily used for document classification problems.

- **Bernoulli Naive Bayes classification:** It is similar to the multinomial classifier except that the fact its predictor variables are the independent Boolean variables. It identifies if a particular word is present or not in a document.

5.4 Classification evaluation

One of the important parts after performing any type of classifier is to evaluate and check its accuracy and efficiency. There exists a lot of other methods where we can evaluate a classifier. Some of the methods are given below:

5.4.1 Holdout method

This is the most common method to evaluate a classifier. In this method, the given data set is divided into two parts such as test and train set 20% and 80% respectively.

The train set is used to train the data and the unseen test set is used to test its predictive power.

5.4.2 Cross-validation

Over-fitting is the most common problem prevalent in most machine learning models. K-fold cross-validation can be conducted to verify if the model is over-fitted at all[24].

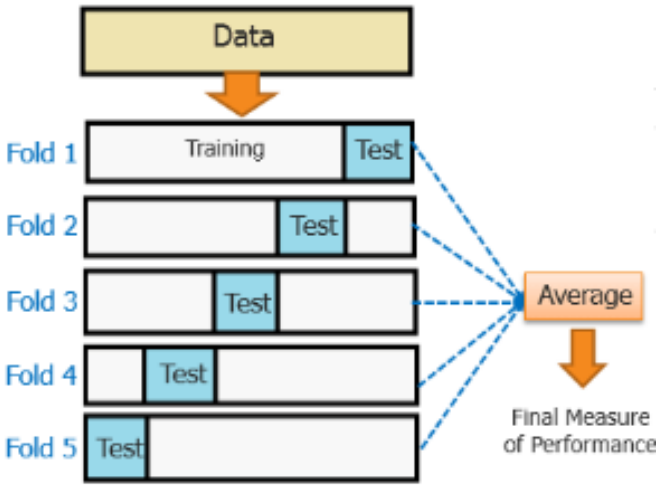


Figure 5.15: Working of K fold cross validation

In this method, the data set is randomly partitioned into k mutually exclusive subsets, each of which is of the same size. Out of these, one is kept for testing and others are used to train the model. Since the split for the training dataset is 60% we consider k to be 6. The same process takes place for all k folds.

5.4.3 Metrics

A metric is defined as a measure of performance and in our context, it is used for estimating the classification performance[25]. These measurements can help to evaluate different characteristics of a model after the training is done. Another common use of metrics is for model selection. When comparing the performances of the trained models in this thesis, we use threshold types of discriminator metrics to evaluate the classification of the test data. An imbalance of the number of observations in each class can cause poor performance when classifying members of the minority classes. This is the reason why we need to evaluate our models with a metric that takes this issue into account and better reflects the model performance than just accuracy[26]. In the list below, some of the common metrics are defined. For a simpler interpretation, we make use of the confusion matrix[27].

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positives (TP)	False Negatives (FN)
	Negative	False Positives (FP)	True Negatives (TN)

Figure 5.16: Confusion matrix

The rows in the table represent the actual class labels while the columns correspond to the classification results. True positive (TP) and true negative (TN) predictions in a binary classification are observations that were correctly positively and negatively classified, respectively, while false positive (FP) and false-negative (FN) predictions represent the elements identified as positive and negative, respectively, when the real labels are the opposite.

Below are the four important metrics used for evaluation

- **Accuracy** - It is a ratio of correctly predicted observation to the total observations. Correct classes are treated equally, so this type of metric can be used to measure performance for data sets with a balanced number of classes. Numerically, it can be calculated as

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

- **Precision** - Probability of the patterns correctly predicted as positive when they were actually positive. This metric can be calculated as

$$Precision = \frac{TP}{TP + FP}$$

- **Recall** - It is the Fraction of positive patterns correctly identified as positive. Recall is defined as follows

$$Recall = \frac{TP}{TP + FN}$$

- **F1-score** - It is the weighted average of precision and recall. It is used as a trade-off for measuring the performance of classification where the imbalance problem is present. When using the F1-score false positive and false negative results are equally penalized. Mathematically, it can be described as

$$F1 - score = \frac{2 \times recall \times precision}{recall + precision}$$

5.4.4 Receiver Operating Characteristics (ROC) curve

The Receiver Operating Characteristics (ROC) curve is used for visual comparison of classification models, which shows the relationship between the true positive rate and the false positive rate. The area under the ROC curve is the measure of the accuracy of the model.

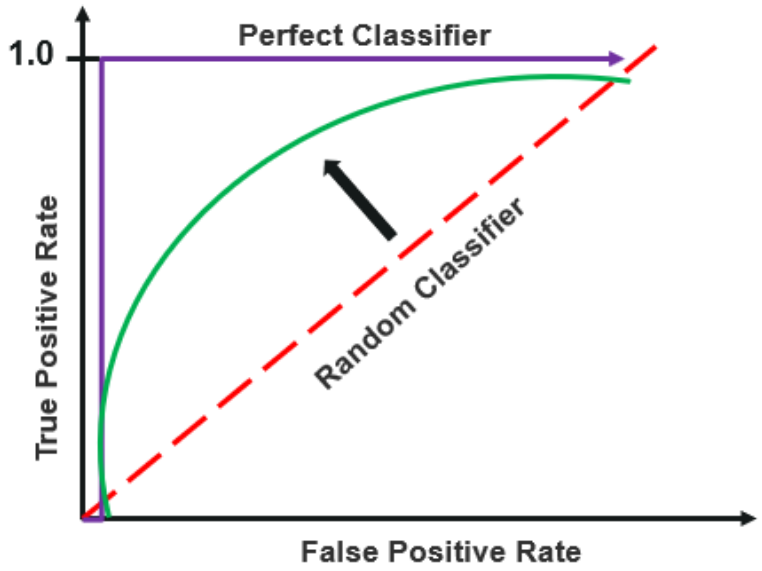


Figure 5.17: ROC curve

5.5 Comparing models and metrics

There are certain criteria that we can use to compare the efficiency of the models.

- The output of the model let say how good the model is predicting the class.
- The accuracy of each classification method. This can be seen in the confusion matrix where the number of correctly predicted classes should be as high as possible.
- Each model should be interpretable as possible.

This section is to evaluate which set of software versions and classification algorithm gives us the best performance while building a classification model for classifying True positive(TP) events and False positive(FP) events. For each software version, the classification model with high accuracy is bolded. To complement the accuracy metric, metrics such as macro-averages of Precision, Recall, and F1-Score are provided. Furthermore, to understand the number of correctly predicted events and wrongly predicted events the confusion matrix has been provided.

Classifier	**Accuracy**	**precision**	**Recall**	**F1-score**
Logistic regression	**84.60%**	**0.84**	**0.86**	**0.6337**
k-NN	83.56%	0.80	0.81	0.8924
Decision trees	84.23%	0.83	0.83	0.8308
SVM	84.09%	0.82	0.83	0.8145
Naive bayes	80.11%	0.81	0.82	0.8005

Table 5.2: Results for the models trained for Prepost combined events from software version 1

Classifier	**Accuracy**	**precision**	**Recall**	**F1-score**
Logistic regression	83.66%	0.83	0.82	0.8447
k-NN	82.12%	0.81	0.79	0.8134
Decision trees	83.46%	0.80	0.81	0.8364
SVM	**83.99%**	**0.84**	**0.85**	**0.8745**
Naive bayes	80.29%	0.82	0.83	0.8157

Table 5.3: Results for the models trained for pre post combined events from software version 2

Tables 5.2 and 5.3 show the performance of classification models trained with prepost combined events from software version 1 & 2. For software version 1 the maximum accuracy of **84.60%** is achieved by using **Logistic Regression**. For software version 2 the maximum accuracy of **83.99%** is achieved by using **SVM**.

Classifier	**Accuracy**	**precision**	**Recall**	**F1-score**
Logistic regression	87.1%	0.85	0.83	0.8542
k-NN	85.3%	0.86	0.87	0.8756
Decision trees	86.8%	0.81	0.84	0.8732
SVM	**88.5%**	**0.85**	**0.84**	**0.8467**
Naive bayes	85.9%	0.83	0.85	0.8615

Table 5.4: Results for the models trained for pre post combined events from software version 3

Classifier	Accuracy	precision	Recall	F1-score
Logistic regression	86.16%	0.84	0.86	0.8615
k-NN	85.48%	0.87	0.84	0.8527
Decision trees	**87.56%**	**0.86**	**0.87**	**0.8728**
SVM	84.63%	0.88	0.85	0.8746
Naive bayes	85.39%	0.86	0.84	0.8596

Table 5.5: Results for the models trained for pre post combined events from software version 4

Tables 5.4 and 5.5 show the performance of classification models trained with prepost combined events from software version 3 & 4. For software version 3 the maximum accuracy of **88.5%** is achieved by using **SVM**. For software version 4 the maximum accuracy of **87.56%** is achieved by using **Decision Trees**.

Classifier	Accuracy	precision	Recall	F1-score
Logistic regression	84.21%	0.83	0.85	0.8447
k-NN	85.14%	0.86	0.87	0.8699
Decision trees	84.66%	0.87	0.85	0.8129
SVM	**85.39%**	**0.83**	**0.83**	**0.8756**
Naive bayes	83.21%	0.81	0.80	0.8723

Table 5.6: Results for the models trained for pre post combined events from software version 5

Classifier	Accuracy	precision	Recall	F1-score
Logistic regression	88.52%	0.88	0.89	0.8754
k-NN	86.12%	0.86	0.83	0.8649
Decision trees	**88.52%**	**0.87**	**0.85**	**0.8760**
SVM	85.72%	0.82	0.87	0.8536
Naive bayes	86.93%	0.83	0.85	0.8638

Table 5.7: Results for the models trained for pre post combined events from software version 6

Table 5.6 and 5.7 shows the performance of classification models trained with prepost combined events from software version 5 & 6. For software version 5 the maximum accuracy of **85.39%** is achieved by using **SVM**. For software version 6 the maximum accuracy of **88.52%** is achieved by using **Decision Tress**.

Performance of each dataset based software version in terms of Confusion matrix:

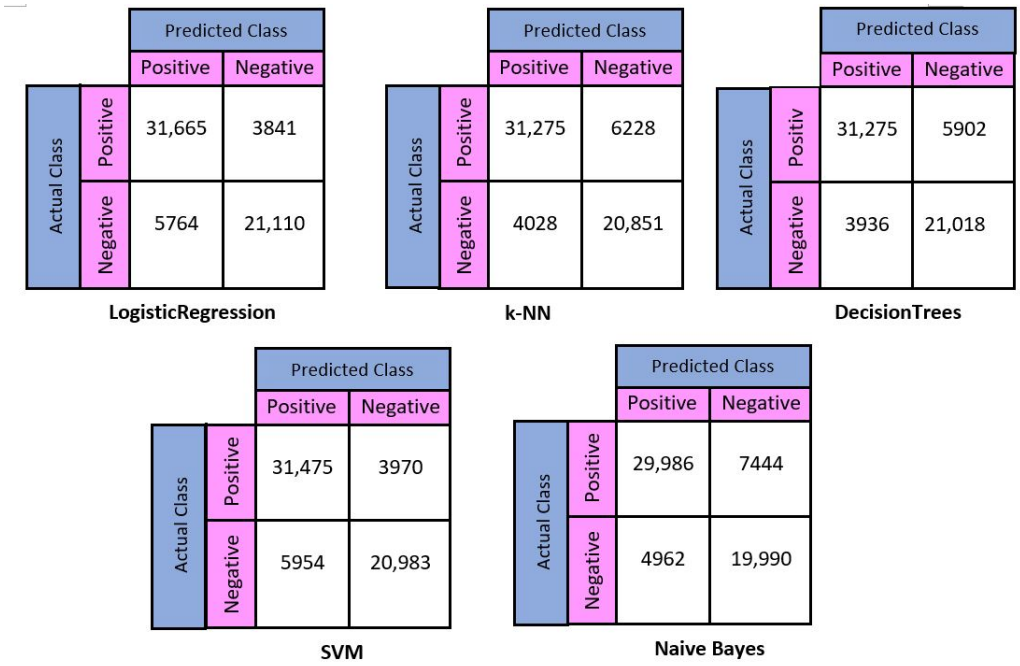


Figure 5.18: Confusion matrix for all different models based on software version 1

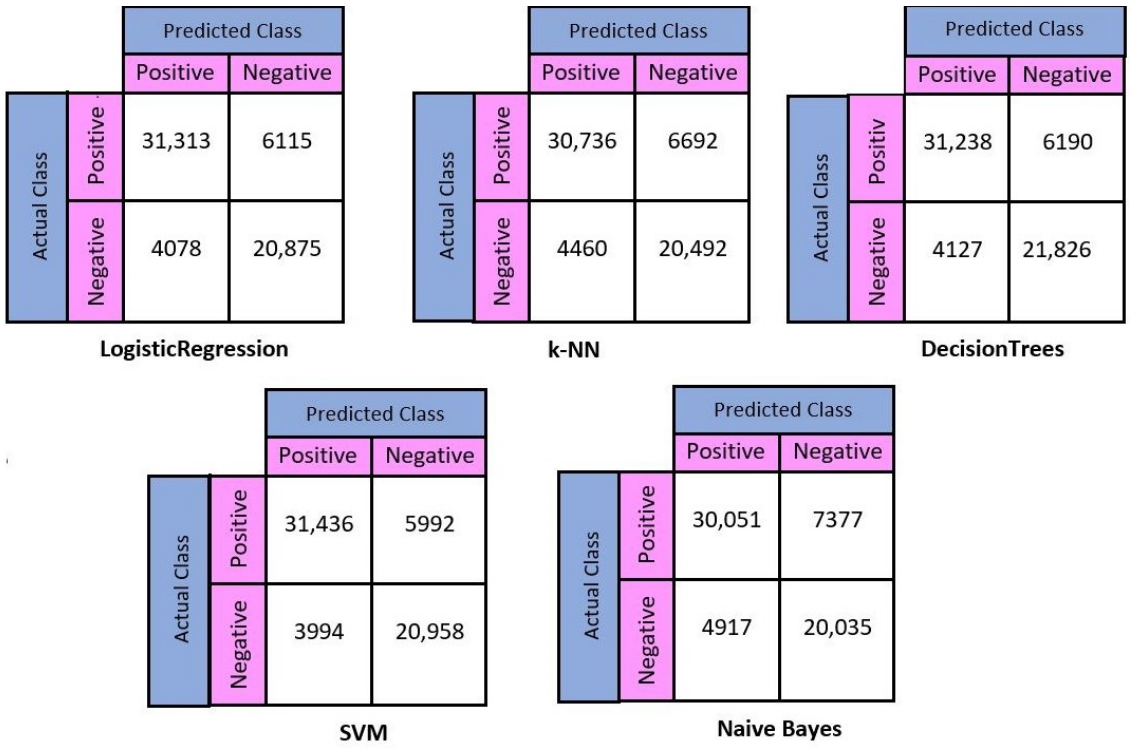


Figure 5.19: Confusion matrix for all different models based on software version 2

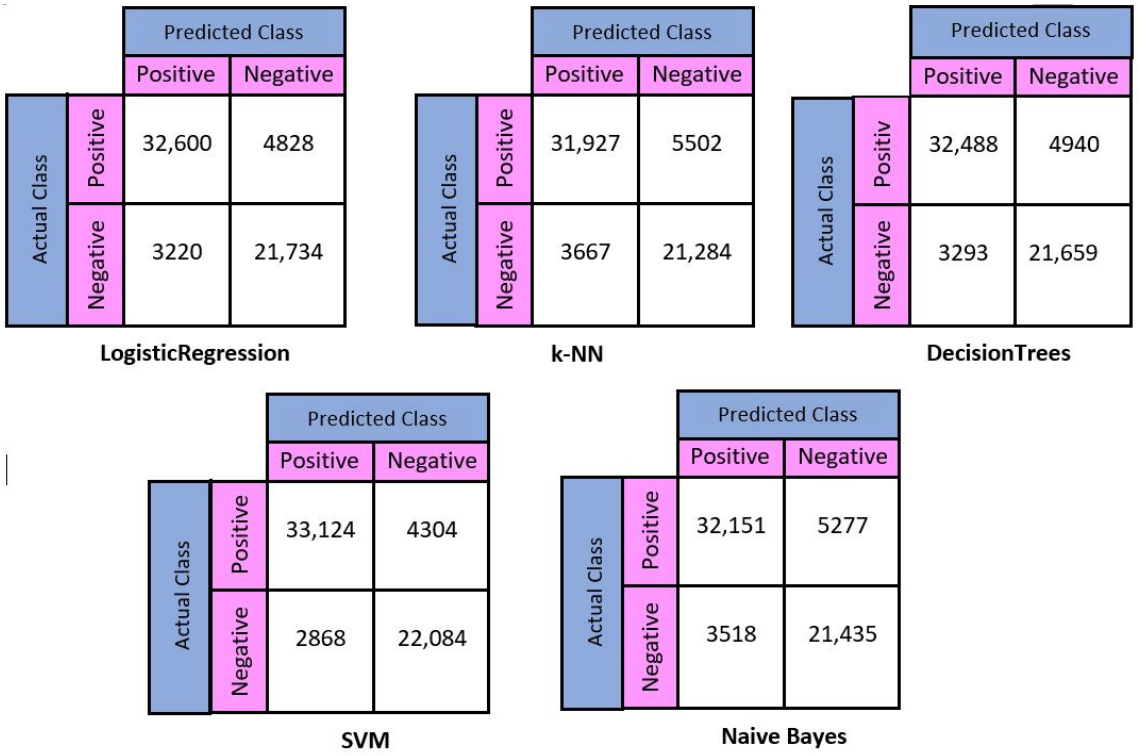


Figure 5.20: Confusion matrix for all different models based on software version 3

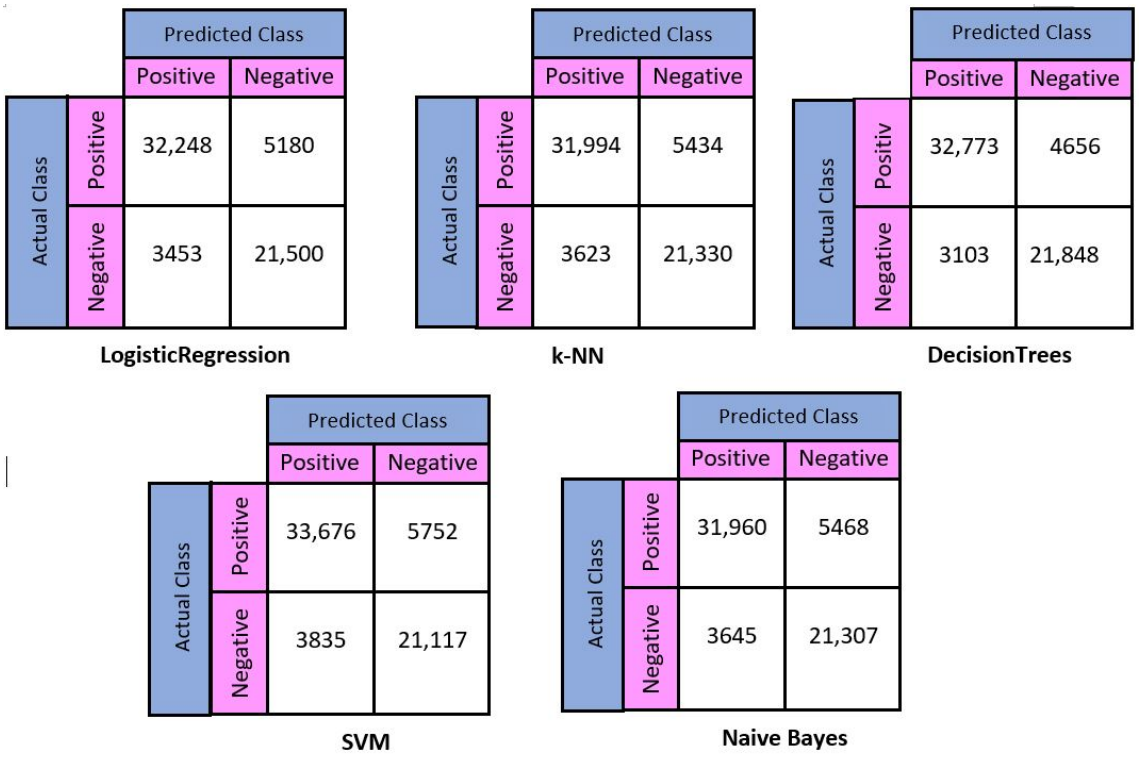


Figure 5.21: Confusion matrix for all different models based on software version 4

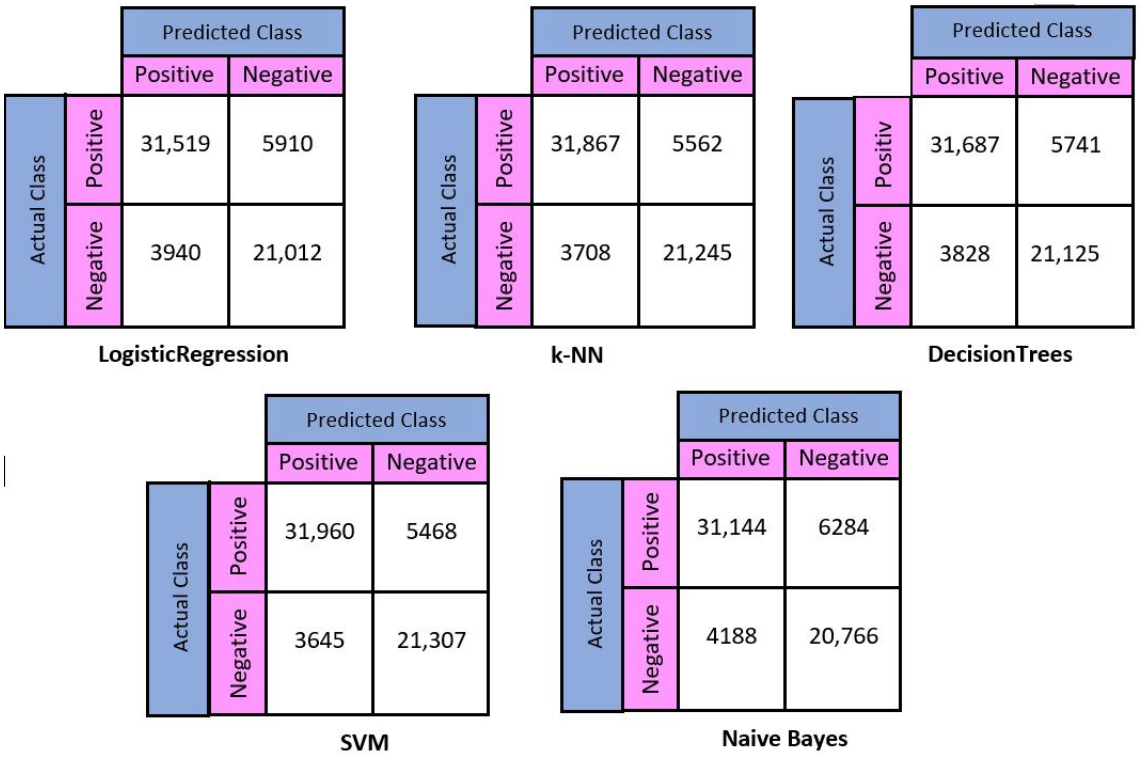


Figure 5.22: Confusion matrix for all different models based on software version 5

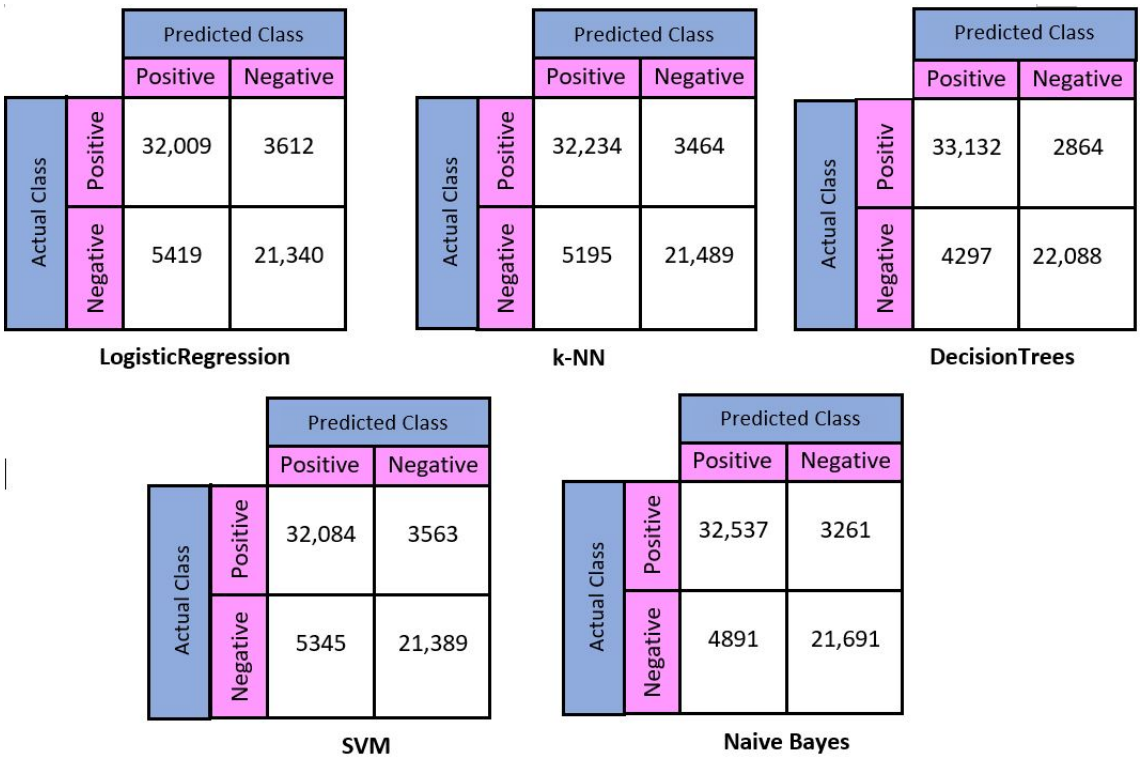


Figure 5.23: Confusion matrix for all different models based on software version 6

6

Pattern recognition techniques

In Machine Learning, pattern recognition is a technique that has some relation from the information of the previous dataset with the new incoming data. The main difference between Machine learning and pattern recognition is ML learns from the available dataset based on the type of algorithm used where pattern recognition is a type of machine learning[28]. Some of the basic components of a pattern recognition system are described in a flow chart:

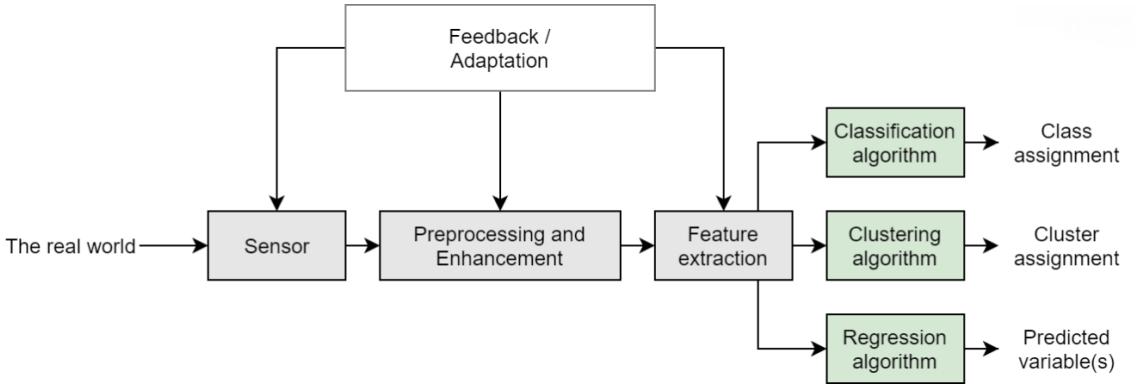


Figure 6.1: Basic components of pattern recognition system

From Figure 6.1, it is seen that the output of pattern recognition is either classification Clustering or Regression. So there is no point in talking about the difference of pattern recognition with classification even-though if we're using it for a different purpose. In this process, it tries to look at the available data and tries to see whether there are any regularities within the dataset or not. There are two parts to how pattern recognition work:

- Descriptive part: Here the algorithms try to find the patterns and start to categorize them.
- Exploratory part: Refers to the critical process of performing initial investigations on data to discover patterns, to spot anomalies, to test hypotheses, and to check assumptions with the help of summary statistics and graphical representations.

6.1 Logistic regression in pattern recognition

As we already worked and explore more on classification algorithms, let's shift to regression methods. Logistic Regression, many think that it is a machine learning

model but it's true that it has a strong background in statistics for a long time. Statistics is very important in machine learning and essential for data analysis or data mining to check how one feature is related to another feature of the dataset. There is a brief description of how logistic regression works in section 5.3.1. If we analyze the dataset using logistic regression we will get a lot of other parameters for each feature such as Coefficient, Standard Error, t, p-values. Here we're only focusing on P-values which tell you whether the relationships are statistically significant or not.

This **p-value** was used in testing the null hypothesis that the coefficient is 0. In this method, we will compare each p-value with the pre-selected value of alpha(threshold). For example, if we use alpha to be 0.05, then the coefficients having p-values of 0.05 or less than the alpha value are statistically significant. On the other hand, a p-value that is greater than the alpha level indicates that there is insufficient evidence in the sample to conclude that a non-zero correlation exists. It is a standard practice to use the p-values of the coefficient to decide whether to include the features or not in the final model.

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.7856	0.001	818.019	0.000	0.784	0.788
x1	-0.0130	0.023	-0.574	0.566	-0.058	0.031
x2	0.0014	0.003	0.473	0.636	-0.004	0.007
x3	-0.0035	0.001	-2.694	0.007	-0.006	-0.001
x4	0.0016	0.002	1.035	0.301	-0.001	0.005
x5	-66.2797	71.807	-0.923	0.356	-207.020	74.461
x6	-0.0029	0.001	-2.995	0.003	-0.005	-0.001
x7	-0.0515	0.043	-1.210	0.226	-0.135	0.032
x8	-0.0018	0.002	-0.847	0.397	-0.006	0.002
x9	0.0003	0.003	0.104	0.917	-0.006	0.007
x10	-47.9504	51.955	-0.923	0.356	-149.782	53.881

Figure 6.2: Regression results for each features in a dataset

We run a logistic regression analysis for all the available datasets and filtered only a certain set of features for further analysis. Figure 6.2 shows sample analysis of one particular dataset where x1,x2, ...x10 are the name of the features. Here in this dataset, there are 266 features reduced to 50 features based on p-values. The smaller the p-value, the stronger the evidence that the null hypothesis should be rejected and the alternate hypothesis should be accepted. The 50 features have a p-value less than 0.05 and are statistically significant, being the main reason for false-positive events. This claim can be verified by further analysis.

6.2 Finding patterns in sorted features

A new dataset is created with the sorted features from logistic regression analysis. For further analysis of each feature, we decide to explore more on visual analytics software called Tableau[29] that helps in simplifying data in a very easily understandable format. Here are some of the samples of visualization of each feature into true positive/false positive events at the same time instance.

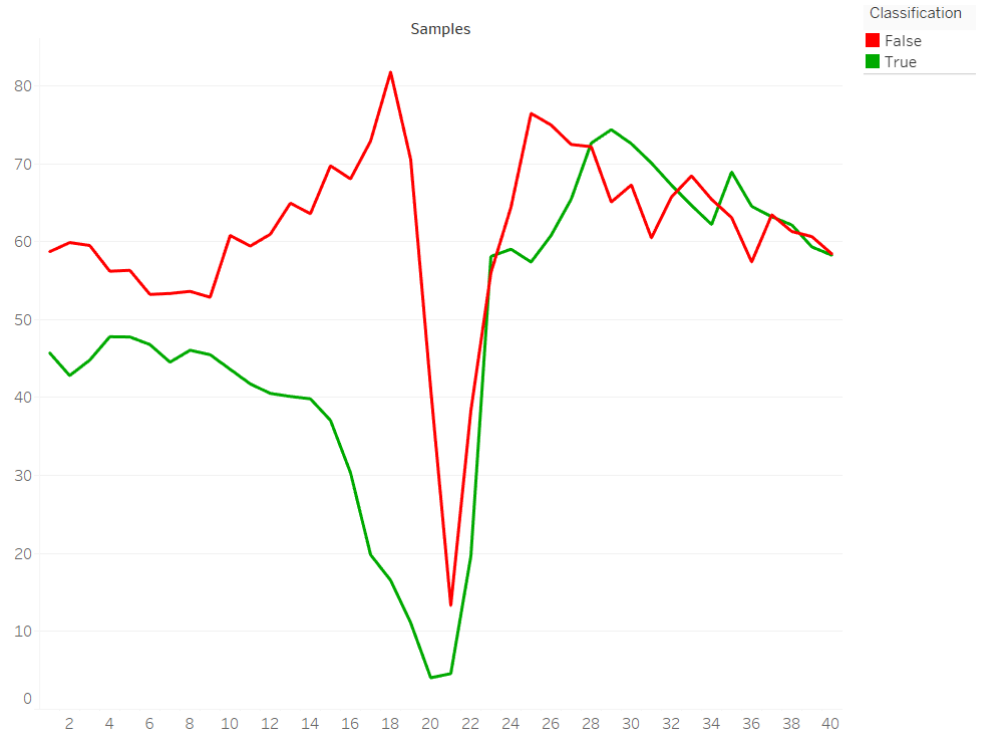


Figure 6.3: Patterns found in feature 1

In Figure 6.3 the x-axis represents samples, Here at sample 20, the number looks quite strange. When we look at samples 9-17 they are diverging a lot, in True positives events the object comes closer and closer but in False positives one, they're increasing and suddenly drops to a lower one.

Figure 6.4: Patterns found in feature 2

In this Figure 6.4 looks like both the events are behaving the opposite way. This feature is related to vehicle steering. Here, in this case, True Positives turn to the left and False Positives turn to the right or something like this but both happened to avoid the accidents.

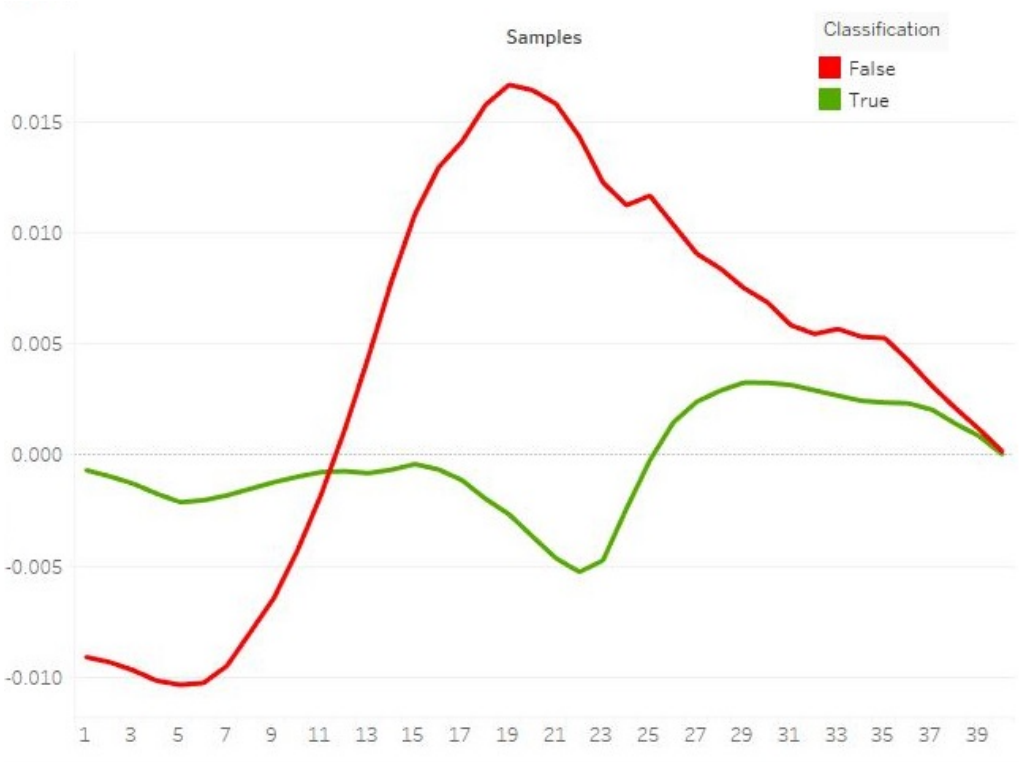


Figure 6.5: Patterns found in feature 3

In Figure 6.5 the feature is based on yaw rate where it clearly shows that there are larger yaw rates for False positives

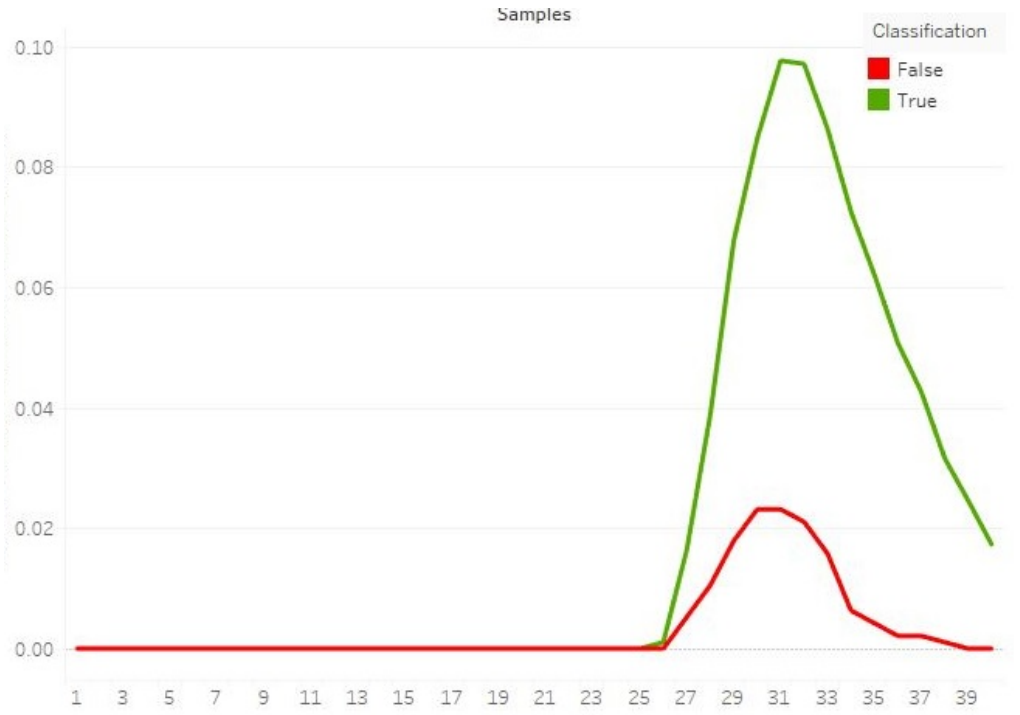


Figure 6.6: Patterns found in feature 4

Here this Figure 6.9 shows something interesting where this feature is requested more frequently for True positives.

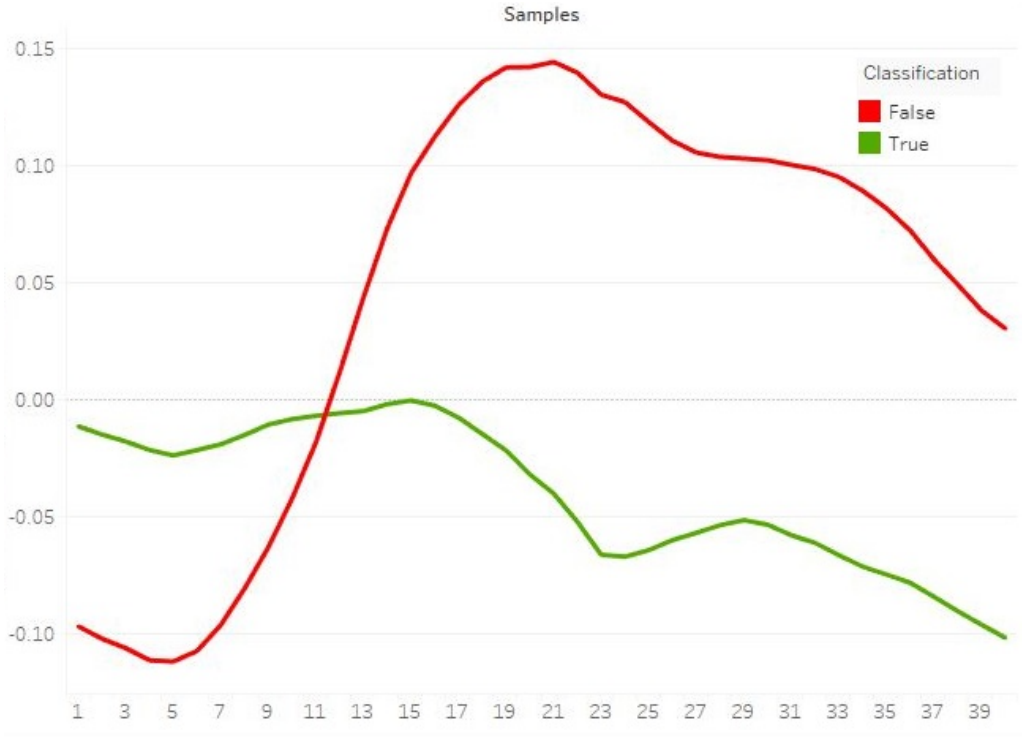


Figure 6.7: Patterns found in feature 5

In Figure 6.7 the false positives after sample 5 have more offset and change a lot than the true ones.

6.3 Patterns from decision tree

6.3.1 Patterns in pre-post files

Once the decision tree model was trained, it can be visualized using the Graphviz library in Python. Such an attempt was made to understand how the model is learning the data and finding the boundaries at which the classification is changed from one class to another. The conditions under which the decision tree is built are,

1) The analysis is performed on the software version 32134732AA which was used in the quarters 201827-201839 and 201927-201939. The software version was chosen randomly. A similar analysis can be done on other software versions as well.

2) only 62 signals were selected and considered for the decision tree analysis based on p values from logistic regression.

3) For the selected 62 signals, only the samples from 18 to 22 (5 samples) of each event were considered since those were the samples that had maximum information about the event. It was observed in Tableau that, both TP and FP showed similar behavior for samples 1-17 and 23-40. However, no mathematical analysis was done to support this claim. It was purely based on observations in Tableau.

4) The accuracy of a Decision tree is close to 82 percent. So, the numbers given by the tree might have slight errors (ex: if the tree says the threshold as 0.5, it might be + or - 0.05). But we suspect that the complex correlation suggested by the tree should be correct.

5) Pruning of the decision tree was done to make sure every pattern mentioned below is observed in at least 100 events out of 6645 events under consideration.

The visualization of decision tree and interpretation of the model after training has resulted in some of the possible patterns in the pre-post data files which VCC is not aware of, are as mentioned below:

1) signal-13 <= 0.34 —> signal-24 <= -0.65 —> signal-11<= -0.85 —> False Positive.

2) signal-46 <= 0.34 —>signal-52 > -0.65 —> signal-24 <= 0.81 —> signal-51 <= 0.77 —> signal-30 <= -0.92 ———>signal-34 <= -1.34 —> signal-43 > 0.40 —> False Positive.

3) signal-2 <= 0.34 —>signal-9 > -0.65 —> signal-15 <= 0.81 —> signal-37 <= 0.77 —> signal-28 <= -0.92 —>signal-41 > -1.34 —> signal-38 <= 0.16 —> signal-13 <= 0.12 —> False Positive.

4) signal-35 <= 0.34 —>signal-26 > -0.65 —> signal-28 <= 0.81 —> signal-17 <= 0.77 —> signal-5 > -0.92 —> signal-12 <= 0.45 —> signal-3 <= 1.22 —>signal-3 <= -0.90 —> signal-3> -0.74 —> False Positive.

5) signal-56 <= 0.34 —>signal-49 > -0.65 —> signal-34 <= 0.81 —> signal-21 <= 0.77 —> signal-40 > -0.92 —> signal-16 <= 0.45 —> signal-19 <= 1.22 —> signal-54 > -0.90 —> signal-30 <= -0.05 —> False Positive.

6) signal-12 <= 0.34 —>signal-39 > -0.65 —> signal-22 <= 0.81 —> signal-43 <= 0.77 —> signal-33 > -0.92 —> signal-56 <= 0.45 —> signal-21 > 1.22 —> False Positive.

7) signal-20 <= 0.34 —> signal-6 > -0.65 —> signal-3 <= 0.81 —> signal-35 > 0.77—> signal-53 > 0.08 —> False Positive.

8) signal-16 <= 0.34 —> signal-37 > -0.65 —> signal-13 > 0.81 —> signal-47 <= 0.21 —> signal-8 <= -0.71 —> False Positive.

9) signal-25 <= 0.34 —> signal-38 > -0.65 —> signal-42 > 0.81 —> signal-53 > 0.21 —> signal-9 <= -0.36 —> signal-13 <= -0.60 —> False Positive.

10) signal-43 <= 0.34 —> signal-58 > -0.65 —> signal-31 > 0.81 —> signal-23 > 0.21 —> signal-4 > -0.36 —> signal-13 <= -0.74 —> signal-47 <= -0.14 —> False Positive.

11) signal-10 <= 0.34 —> signal-42 > -0.65 —> signal-34 > 0.81 —> signal-57 > 0.21 —> signal-23 > -0.36 —> signal-31 > -0.74 —> signal-44 <= 1.29 —> False Positive.

12) signal-19 <= 0.34 —> signal-50 > -0.65 —> signal-9 > 0.81 —> signal-14 > 0.21 —> signal-37 > -0.36 —> signal-29 > -0.74 —> signal-32 > 1.29 —> signal-45 > -0.25 —> False Positive.

13) signal-4 > 0.34 —> signal-16 <= -0.14 —> signal-23 <= -0.61 —> signal-36 <= 0.33 —> signal-53 <= 0.49 —> False Positive.

14) signal-5 > 0.34 —> signal-45 <= -0.14 —> signal-12 <= -0.61 —> signal-38 <= 0.33 —> signal-15 > 0.49 —> signal-37 > 0.27 —> False Positive.

15) signal-46 > 0.34 —> signal-11 <= -0.14 —> signal-46 <= -0.61—> signal-29 > 0.33 —> signal-30 <= -0.55 —> False Positive.

16) signal-14 > 0.34 —> signal-55 <= -0.14 —> signal-33 <= -0.61—> signal-58 > 0.33 —> signal-14 > -0.55 —> signal-31 > 0.77 —> False Positive.

17) signal-32 > 0.34 —> signal-23 <= -0.14 ——> signal-36 > -0.61—>signal-15 > 2.20 —> signal-10 > -0.99 —> False Positive.

18) signal-41 > 0.34 —> signal-37 > -0.14 —> signal-51 <= 0.31 —> signal-26 <= -0.59 —> signal-13 <= -0.49 —> False Positive.

6.3.2 Patterns in single files

Similar to the last section, the decision tree algorithms were applied on single files. The conditions under which the decision tree is built are,

1) The analysis is performed on the software version 32134613AC which was used in the quarters 201714-201726, 201727-201739, 201814-201826, 201840-201852, and 201901-201913. The software version was chosen randomly since it had enough data for training. A similar analysis can be done on other software versions as well.

2) only 40 signals were selected and considered for the decision tree analysis based

on p values from logistic regression.

3) The accuracy of a Decision tree is close to 80 percent. So, the numbers given by the tree might have slight errors (ex: if the tree says the threshold as 0.5, it might be + or - 0.05). But we suspect that the complex correlation suggested by the tree should be correct.

4) Pruning of the decision tree was done to make sure every pattern mentioned below is observed in at least 100 events out of 5948 events under consideration.

The visualization of decision tree and interpretation of the model after training has resulted in some of the possible patterns in the single data files which VCC is not aware of, are as mentioned below:

1) signal-40 <= 0.91 —> signal-24 <= -0.15 —> signal-12 <= -0.34 —> signal-15 > 0.56 —> signal-22 <= -0.42 —> signal-30 > 0.94 —> False Positive.

2) signal-32 <= 0.91 —> signal-14 <= -0.15 —>
signal-16 > -0.34 —> False Positive.

3) signal-38 <= 0.91 —> signal-23 > -0.15 —> signal-17 <= 0.09 —> signal-39 > -0.37 —> False Positive.

4) signal-22 <= 0.91 —> signal-32 > -0.15 —> signal-7 > 0.09 —> False Positive.

5) signal-3 > 0.91 —> signal-32 <= -0.04 —> signal-23 <= 0.50 —> signal-38 <= 0.56 —> signal-27 > -0.66 —> False Positive.

6) signal-12 > 0.91 —> signal-33 <= -0.04 —> signal-31 <= 0.50 —> signal-27 > 0.56 —> False Positive.

7) signal-7 > 0.91 —> signal-35 <= -0.04 —> signal-28 > 0.50 —> False Positive.

8) signal-32 > 0.91 —> signal-13 > -0.04 —> False Positive.

6.4 Discussions

From the analysis of each feature using Tableau, it is evident that they play a role in classifying the events as false-positive events. But the complex correlation between the signals can be realized better by analyzing the decision tree model. The decision tree model was visualized in image and text form after the model was trained. Based on the decision tree model that has learned the data, the patterns leading to

false-positive events were noted and presented in this chapter.

The decision tree images for pre+post and single data files are attached for reference. The same patterns are presented in the above sections in a statement form. The images are too big to fit on one page and are readable. The same images will be attached to the report while submitting it.

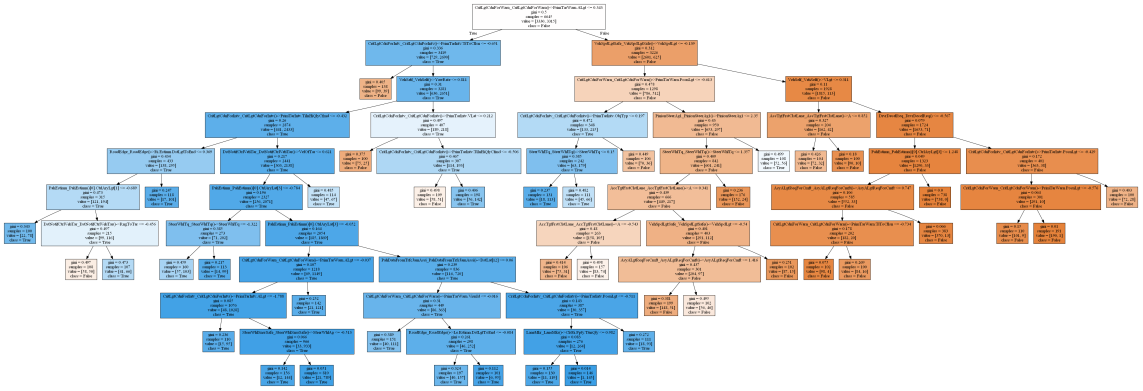


Figure 6.8: Decision tree images for pre+post data

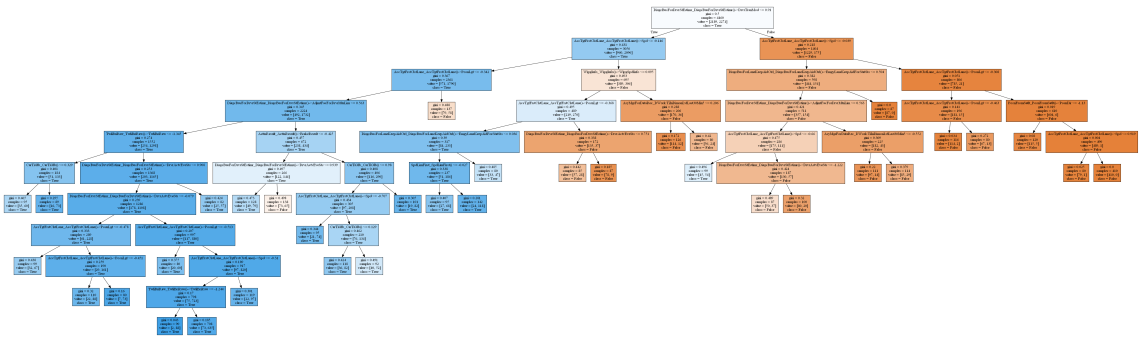


Figure 6.9: Decision tree images for single data

7

Conclusions and future work

7.1 Summary of the thesis work

The goal of the thesis was divided into two parts:

1) Classification algorithm using machine learning for classifying events into true-positives and false-positives.

The data provided by VCC was collected over 13 quarters and six different software versions were involved during the 13 quarters. The thesis work started with an analysis of the dataset provided by VCC. Data cleansing and preprocessing is the first step in data analysis. After treating the missing values, categorical data, and feature scaling, the data was ready for further analysis.

Too much information fed into the machine learning model while training will make the trained model perform well on training data and perform poorly on the test data. Hence, the two well-known dimension reduction techniques, i.e., Principle component analysis, and low variance filter method were used to reduce the feature space from 266 to less than 60. There was no loss or change in the data during the process of dimension reduction. While applying dimension reduction techniques, the data is analyzed by the two methods and the output of both the methods were the names of the features which are statistically significant. The data related to Only those features were fetched by the code and this subset of the original data was used for further analysis. It was manually verified for one feature for both methods that the original data was unchanged.

The convolutional neural networks (CNN) are popularly used for object detection and segmentation in images. The idea was to convert the data corresponding to every event into eight-bit color image representation and train the CNN to classify the events into true-positive and false-positive. Since CNN's are translation invariant which means that CNNs are not sensitive to the position of an object in the image, this approach failed to classify the events. Hence, CNNs can not be used for such data for classification.

Logistic regression, K-nearest neighbors, decision trees, support vector machines, and naive Bayes are the popularly known Machine learning algorithms that were in-

vestigated to classify the events into true-positives and false-positives in this work. Since the data was segregated into six sets, based on the software version of the quarter, all the machine learning algorithms were applied to six sets of data. It was observed that support vector machine algorithm, decision tree algorithm, and logistic regression algorithm performed well on different datasets.

2) Investigating the possibility of finding complex or simple patterns in the data that VCC is not aware of.

The second goal of the thesis was to investigate if it is possible to find complex or simple patterns in the data that VCC is not aware of. We proceeded with finding patterns in data from two approaches. The first approach is to use a logistic regression model, get the model coefficients, find their p-values. P-values lesser than 0.05 indicate that the null hypothesis is rejected and the signal has a greater impact on deciding the class of the event. The signals having p-values lesser than 0.05 were selected and visualized using the tool called Tableau. This enabled us to visualize individual signals values belonging to true-positive and false-positive classes. The results were presented in chapter 6.

The second approach is by visualizing the decision tree model. Once the decision tree model is trained using the data, it can be visualized using an open-source package called Graphviz in an image format and text form as well. This visualization explains how the tree model is constructed and the complex correlation between the signals. 18 patterns in the pre+post data file and 8 patterns in a single data file were found from the decision tree model.

The DRO file also contains many features, which were not considered for this thesis work. Only pre, post, and single files were analyzed. We feel that there might be more insights in the DRO file which can be the future work.

7.2 Summary of research question

- How can the True positive/ False positive events be identified using machine learning algorithms?

  The starting point for the thesis work was the annotated data provided by VCC. This annotated data can be used to train the machine learning models after suitable data cleansing and preprocessing. The only challenge was to rearrange the data according to the requirement of the machine learning model. Thus, the annotation of data plays a major role in the correctness of the patterns found by the machine learning models.
  This thesis work was carried out considering the annotations to be accurate and supervised machine learning techniques were explored in fining the patterns in the false-positive events. As future work, unsupervised machine learning

algorithms can be explored where annotation of the data has no role to play. The decision boundary learned by the unsupervised models can be studied to find the patterns in the events. Care has to be taken while applying dimension reduction techniques and signals to be discussed with the domain specialist as discussed in section 3.3.

- Which machine learning algorithm performs better in classifying the events?

  It was observed that the support vector machine (SVM) algorithm, decision tree algorithm, and logistic regression algorithm performed well on different sets of data, segregated based on software versions. There is no proper theoretical explanation for why different machine learning models are performing better at data from different software versions. The results were observed practically.

- Is it possible to find complex or simple patterns that VCC has not yet easily found? Why/why not? What recommendation do we have for VCC to analyze data better in the future to improve performance of AEB system?

  The pattern recognition task was a challenging part of the thesis. We tried approaching the problem in different ways. The effort was to find patterns concerning individual signals as well as a bunch of signals correlated to result in a false-positive event. We could find patterns in individual signals using logistic regression at first to select important signals and then visualize them using Tableau. The results are presented in section 6.2

  There were four patterns discovered by VCC under which the false positives occurred.

  a) Object type = Large Animal.

  This is found in the DRO file and was confirmed by visualizing in Tableau. This signal is present in DRO files. During the initial analysis of DRO files, this pattern was noticed. This thesis work did not explore the DRO files as the focus was on the pre, post, and single data files. There may be more patterns in the DRO files, we could not explore more as our focus got shifted to pre, post, and single data files.

  b) Object lives for a very brief time.

  The Sudden position change of an object in the vicinity of the car will be reflected in a rapid change in the value of the signal "signal-23". Pattern 12 in section 6.3.2 includes this signal at the top of the tree. This signal had relatively higher variance compared to other signals, thus got selected by the dimension reduction techniques. This is one typical scenario where trusting

the dimension reduction techniques without domain-specific knowledge can lead to misinterpretation of the signal. The difference in the value of any two adjacent samples of this signal decides whether there is a sudden change in the position of an object or not. Hence, it is the relative change of value from one sample to the next sample which should be considered for analysis and not the absolute value of the sample itself. Thus, the inclusion of this signal in the pattern given by the decision tree doesn't make complete sense since the relative value of this signal needs to be considered not the absolute value as in pattern 12.

c) Object moves very fast.

The signals indicating the movement of an object in the vicinity of the car are, "signal-24", "signal-34", and "signal-18". These signals did not appear in any of the patterns, as the signal values had relatively lower variance compared to other signals and were dropped off during the dimension reduction step. We failed to realize that there were signals where the difference in value between the adjacent samples is considered to infer something, then the absolute values of samples themselves. We did not have any special care taken to include/ exclude such signals during dimension reduction techniques. This is again a scenario where we lacked domain-specific knowledge and blindly continued with the results of dimension reduction techniques. In the future, care has to be taken to treat such signals differently.

d) signal-19 > 0 from a single file.

When the decision tree model was trained on single data files, the branch "signal-19 > 0.09" was observed at the lowest level of the tree (leaf node) in the fourth pattern mentioned in section 6.3.2. However, there was one more pattern having the signal "signal-19 < 0.09" (pattern 3, section 6.3.2).

7.3 Future work and recommendations

The collision avoidance team of VCC has access to a huge amount of data from customer cars on the cloud. This could play an important role in data-driven decisions. When it comes to detecting false positive events and patterns, the following points can be taken into consideration.

1) Data Mining: The data has to be extracted and saved in a suitable form. For instance, collecting all pre, post, and single data files along with their readout ids, classification, etc. Data mining can be done using SQL or Python.

2) Data preprocessing: Once the data mining is done, it has to be preprocessed to remove missing values if any, address the categorical data, scale the values of all signals.

3) Dimension reduction: This is a bit tricky part. The methods used in this thesis work are principle component analysis and the low variance filter method. Both these methods focus on eliminating the features which have low variance. In simple words, if the difference between the maximum value and minimum value of the feature is less, the spread of the values will be limited to a small range. This will result in low variance and the feature will be eliminated.
This might be straightforward from a statistics point of view, but without the domain knowledge, it would be difficult to decide which feature is important and which is not. It is a tough call while selecting the signals for further analysis. This issue has to be addressed in future work if the work is continued on upcoming data.

4) machine learning models to be considered for classification: Converting the data into an image file and using a Convolutional Neural Network was analyzed and found to be not useful for this kind of data. various Machine learning algorithms were used to analyze the pre-post data files combined and single files separately. In pre-post data files, the maximum information was available in samples 18-22 from visual analysis. But, all the samples of the pre-data file along with a single data file can be considered as "sequential data" and Recurrent Neural Networks can be used to train the model with selected signals for the entire 20 samples (or the duration of the pre-data file).
The same can be extended to pre single post combined data file for better analysis.

5) For pattern recognition in upcoming data, unsupervised learning techniques can be explored for 2 class classification. The decision boundary given by the unsupervised learning algorithms can be visualized or represented in a text form, indicating the way the algorithm has learned to classify the events. This decision boundary analysis will give us a deeper understanding of individual signals affecting the TP-FP event occurrence.

It was realized at the later stages of the thesis that the machine learning models can not only be used just for classification of the events but the models can also be visualized at every node in case of decision tree to understand how the model has learned the data. This could give a better understanding of knee points where the decision of the model changes from one class to another. A similar approach can be taken in the DRO data file as well to find the complex patterns for false-positive events.

Bibliography

[1] Active Safety:https://en.wikipedia.org/wiki/Active-safety

[2] Impact of Active Safety: https://roadsafetyfacts.eu/road-safety-what-progress-has-been-made/

[3] The 6 Levels of Vehicle Autonomy Explained: https://www.synopsys.com/automotive/autonomous-driving-levels.html

[4] Z. Guan, T. Ji, X. Qian, Y. Ma and X. Hong, "A Survey on Big Data Preprocessing," 2017 5th Intl Conf on Applied Computing and Information Technology/4th Intl Conf on Computational Science/Intelligence and Applied Informatics/2nd Intl Conf on Big Data, Cloud Computing, Data Science (ACIT-CSII-BCD), 2017, pp. 241-247, doi: 10.1109/ACIT-CSII-BCD.2017.49.

[5] J. Abasova, J. Janosik, V. Simoncicova and P. Tanuska, "Proposal of Effective Preprocessing Techniques of Financial Data," 2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES), 2018, pp. 000293-000298, doi: 10.1109/INES.2018.8523922.

[6] 7 Ways to Handle Missing Values in Machine Learning https://towardsdatascience.com/7-ways-to-handle-missing-values-in-machine-learning-1a6326adf79e

[7] Ordinal and One-Hot Encodings for Categorical Data: https://machinelearningmastery.com/one-hot-encoding-for-categorical-data/

[8] Feature Scaling for Machine Learning: Understanding the Difference Between Normalization vs. Standardization: https://www.analyticsvidhya.com/blog/2020/04/feature-scaling-machine-learning-normalization-standardization/

[9] 3 Things You Need To Know Before You Train-Test Split: https://towardsdatascience.com/3-things-you-need-to-know-before-you-train-test-split-869dfabb7e50

[10] E. Slanjankic, H. Balta, A. Joldic, A. Cvitkovic, D. Heric and E. Veledar, "Data mining techniques and SAS as a tool for graphical presentation of principal components analysis and disjoint cluster analysis results," 2009 XXII International Symposium on Information, Communication and Automation Technologies, 2009, pp. 1-5, doi: 10.1109/ICAT.2009.5348419.

[11] Dimensionality Reduction: Analysis on Low Variance Filter method: http://dataminingreporting.weebly.com/blog/2-dimensionality-reduction-low-variance-filter

[12] R. Xin, J. Zhang and Y. Shao, "Complex network classification with convolutional neural network," in Tsinghua Science and Technology, vol. 25, no. 4, pp. 447-457, Aug. 2020, doi: 10.26599/TST.2019.9010055.

[13] R. Jiang and S. Mei, "Polar Coordinate Convolutional Neural Network: From Rotation-Invariance to Translation-Invariance," 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 355-359, doi: 10.1109/ICIP.2019.8802940.

[14] Sahu, S.K., Kumar, P. Singh, A.P. Modified K-NN algorithm for classification problems with improved accuracy. Int. j. inf. tecnol. 10, 65–70 (2018). https://doi.org/10.1007/s41870-017-0058-z.

[15] Kotsiantis, S.B., Zaharakis, I.D. Pintelas, P.E. Machine learning: a review of classification and combining techniques. Artif Intell Rev 26, 159–190 (2006). https://doi.org/10.1007/s10462-007-9052-3.

[16] Ikonomakis, M., Kotsiantis, S. and Tampakas, V., 2005. Text classification using machine learning techniques. WSEAS transactions on computers, 4(8), pp.966-974.

[17] P. Rao and J. Manikandan, "Design and evaluation of logistic regression model for pattern recognition systems," 2016 IEEE Annual India Conference (INDICON), 2016, pp. 1-6, doi: 10.1109/INDICON.2016.7839010.

[18] Jordan, M.I., 1995. Why the logistic function? A tutorial discussion on probabilities and neural networks.

[19] P. Soucy and G. W. Mineau, "A simple KNN algorithm for text categorization," Proceedings 2001 IEEE International Conference on Data Mining, 2001, pp. 647-648, doi: 10.1109/ICDM.2001.989592.

[20] J. R. Quinlan, "Decision trees and decision-making," in IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 2, pp. 339-346, March-April 1990, doi: 10.1109/21.52545.

[21] Noble, W. What is a support vector machine?. Nat Biotechnol 24, 1565–1567 (2006). https://doi.org/10.1038/nbt1206-1565

[22] Bhavsar, H. and Panchal, M.H., 2012. A review on support vector machine for data classification. International Journal of Advanced Research in Computer Engineering Technology (IJARCET), 1(10), pp.185-189.

[23] Rish, I., 2001, August. An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46).

[24] J. D. Rodriguez, A. Perez and J. A. Lozano, "Sensitivity Analysis of k-Fold Cross Validation in Prediction Error Estimation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 3, pp. 569-575, March 2010, doi: 10.1109/TPAMI.2009.187.

[25] Naser, M.Z. and Alavi, A., 2020. Insights into performance fitness and error metrics for machine learning. arXiv preprint arXiv:2006.00887.

[26] Handelman, G.S., Kok, H.K., Chandra, R.V., Razavi, A.H., Huang, S., Brooks, M., Lee, M.J. and Asadi, H., 2019. Peering into the black box of artificial intelligence: evaluation metrics of machine learning methods. American Journal of Roentgenology, 212(1), pp.38-43.

[27] Visa, S., Ramsay, B., Ralescu, A.L. and Van Der Knaap, E., 2011. Confusion matrix-based feature selection. MAICS, 710, pp.120-127.

[28] S. Alhusain, S. Coupland, R. John and M. Kavanagh, "Towards machine learning based design pattern recognition," 2013 13th UK Work-

shop on Computational Intelligence (UKCI), 2013, pp. 244-251, doi: 10.1109/UKCI.2013.6651312.

[29] Wesley, R., Eldridge, M. and Terlecki, P.T., 2011, June. An analytic data engine for visualization in tableau. In Proceedings of the 2011 ACM SIGMOD International Conference on Management of data (pp. 1185-1194).

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY