Health Insurance Cross-Sell Prediction

A Project Report submitted in partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

Sai Chand Sunkara	**121810314030**
Chandanika Abburi	**121810314035**
Kattoju Bala Kiran	**121810314015**
K M S Sita Rama Raju	**121810314006**

Under the esteemed guidance of

Dr. Naina Narang

Assistant Professor, GIT

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM

(Deemed to be University)

VISAKHAPATNAM

March 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY

GITAM

(Deemed to be University)

DECLARATION

We, hereby declare that the Project review entitled "**Health Insurance Cross-Sell Prediction**" is an original work done in the Department of Computer Science and Engineering, GITAM Institute of Technology, GITAM (Deemed to be University) submitted in partial fulfilment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: ,March 2022.

Registration No:	**Name:**	**Signature:**
121810314030	**Sai Chand Sunkara**	**Sai Chand**
121810314035	**Chandanika Abburi**	**Chandanika**
121810314015	**Kattoju Bala Kiran**	**Bala Kiran**
121810314006	**K M S Sita Rama Raju**	**Rama Raju**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GITAM INSTITUTE OF TECHNOLOGY

GITAM (Deemed to be University)

<u>BONAFIDE CERTIFICATE</u>

This is to certify that the project report entitled **"Health Insurance Cross-Sell Prediction"** is a bonafide record of work carried out by **Sai Chand Sunkara(121810314030), Chandanika Abburi(121810314035), Kattoju Bala Kiran(121810314015), KMS Sita Rama Raju(121810314006)** submitted in partial fulfilment of requirement for the award of degree of Bachelors of Technology in Computer Science and Engineering.

PROJECT GUIDE	**HEAD OF THE DEPARTMENT**
Dr. Naina Narang	**Dr.R. Sirresha**
(Assistant Professor)	**(PROFESSOR)**
CSE, GIT	**CSE, GIT**
GITAM	**GITAM**

ACKNOWLEDGEMENT

We would like to thank our project guide **Dr.Naina Narang**, Assistant Professor, Department of CSE for his stimulating guidance and profuse assistance. We shall always cherish our association with him for his guidance, encouragement and valuable suggestions throughout the progress of this work. We consider it a great privilege to work under his guidance and constant support

We also express our thanks to the projects reviewers **G. Sivanarayana** and **P. Sanyasi Naidu**, Department of CSE, GITAM (Deemed to be University) for their valuable suggestions and guidance for doing our project.

We consider it is a privilege to express our deepest gratitude to **Dr.R. Sireesha**, Head of the Department, Computer Science Engineering for her valuable suggestions and constant motivation that greatly helped us to successfully complete this project.

Our sincere thanks to **Dr. M. Venkateswara Rao**, Principal, GITAM Institute of Technology, GITAM (Deemed to be University) for inspiring us to learn new technologies and tools.

Finally, we deem it a great pleasure to thank one and all that helped us directly and indirectly throughout this project.

Sai Chand Sunkara	121810314030
Chandanika Abburi	121810314035
Kattoju Bala Kiran	121810314015
K M S Sita Rama Raju	121810314006

1. ABSTRACT

Greater competition on the financial market, as well as the drive to earn massive returns, implies that banks are increasingly seeking for strategies that will enable them to enhance sales. One of these turned out to be sales strategies such as cross selling and up selling, which surely improve the bank's profit. In furthermore, cross-selling helps to establishing a wider product portfolio of the client, which links it more tightly to the bank and decreases the chance of the customer leaving to a rival business. This research is undertaken to illustrate the techniques and outcomes of cross-selling tactics by using multiple approaches to create future data-oriented decision making in the banking industry and is one of the key financial businesses. We did data analysis, data transformation, scaling, and oversampling methods for the result column is done separately before using machine learning algorithms to categorize potential clients for cross-selling. To oversample the unbalanced data set, we employed the SMOTE approach to artificially construct the dataset and make it balanced. Precision, Recall, and F1 scores are utilized as the accuracy measures of the machine learning algorithms.

2. INTRODUCTION

As an example of cross-selling methods, this initiative seeks to forecast healthcare insurance customers who are likely to be interested in purchasing automobile insurance. We utilized a dataset including over 381,000 samples, as well as 12 different variables such as demographics and vehicle details. The Age and Annual Premium variables are scaled using the min-max scaling approach, while the Gender, Vehicle Age, and Vehicle Damage data are label encoded. To oversample and balance the dataset, the SOMTE approach described below is utilized.

We trained and performed hyper tuning of the Decision tree, Random Forest, Logistic regression, XGBoost, KNN, Linear SVC and ANN algorithms to predict the target variable. We have generated the classification report after training the models to get the model's accuracy both with training and testing data sets.

This research is arranged as follows, basic information offers complete literature analysis about cross-selling and briefly examines relevant studies released in the recent 5 years. Material and methods section provides the approaches and procedures for cross-selling forecasts. In result and discussion, prediction results of algorithms were shown related two categorization metrics and lastly, discussion and conclusion portion summarizes the research and offers other possibilities to expand the study.

3. LITERATURE SURVEY

Cross-selling is very well method that tries to maximize the institution’s profit by recommending additional products in a single transaction. The main goal of cross-selling is the concept that in order to give additional products while purchasing Methodologies such as cross-selling as well as market-basket analysis are used to decrease client turnover rate by guaranteeing more targeted offers, service and rewards.

Existing research implementing the solution for this use case using ROSE oversampling in R language along with outlier handling and feature selection techniques and missing value treatment just by removing the columns has given accuracies around 85% with Random Forest algorithm without hyper-tuning. To improve the classification accuracy, this use case can be implemented using other oversampling techniques and also by hyper-tuning the models for more better accuracies on test data which is implemented and presented in this research. Similar to the previously made researches for the accuracy measurement, ROC curve is taken as the primary measurement method predicting on the test data with each and every model to be trained on best parameters.

A well-planned and managed data collection process is crucial for having effective cross-selling marketing approach for their present clients as described in the research for telecoms customers. People like to buy products together in our everyday buying patterns while these two or more items are connected to each other.

Machine learning algorithms got so prominent to gain valuable insights after data analysis and give benefits for growing the sales via numerous marketing tactics and minimize the rate of consumers who want to discontinue company's services. The machine learning algorithms that employ ensemble-based or boosting algorithms have attained the greatest overall accuracy, over 92 percent, when forecasting the clients the organization should seek for cross-selling.

Producing consistent suggestions to boost profit by expanding the number of items to be sold via cross-selling tactics is vital for any business and numerous state-of-art methodologies are utilized to make product recommendations.

Numerous different ways have been presented to improve effective cross-selling tactics for higher sales figures and customer commitment, such as categorizing consumers with similar purchasing habits, market basket research, or showing customer characteristics with regard to standard RFM (recency, frequency, monetary) models to provide suggestions.

The ability to disclose the customer's purchasing behavioral pattern from saved interactions may be highly advantageous for maintaining cross-selling and up-selling strategies in any type of sale platforms, whether online or offline shops, and there are numerous different data mining procedures like as association rule that can give suggestions for marketing strategies.

4. PROBLEM IDENTIFICATION AND OBJECTIVES

4.1 PROBLEM IDENTIFICATION

An insurance plan is an agreement in which a corporation agrees to offer a promise of reimbursement for defined loss, damage, disease, or death in return for money of a certain premium. A premium is an amount of money that the client must pay to an insurance firm in exchange for this assurance. Vehicle insurance, the same as medical insurance, requires the users to pay the premium of a certain amount to the insurance company each year and so in the event of an unfortunate turn of events caused by vehicle, the insurance company would provide a reimbursement to the customer.

4.2 OBJECTIVES

Developing a model to anticipate if a client would be engaged in Insurance is incredibly useful for the firm as it can then properly organize its marketing strategies to reach out to all those consumers and maximize its business strategy and profits. Therefore, to anticipate if the consumer would really be engaged in Motor insurance, information on user demography (gender, age, area code type), Vehicles demography (Vehicle Age, Damage) and Policy details (Premium, sourcing channels), etc. is given as the input. Moreover, we employ algorithms like Decision Tree, Random Forest, Logistic Regression, KNN, Linear SVC, XGBoost classifier to perform the complete workflow.

4.3 HARDWARE AND SOFTWARE REQUIREMENTS

- LANGUAGE: Python 3.6
- TOOLS: Jupyter Notebook, Pycharm, Colab, Kaggle (PREFERRED)
- LIBRARIES: Numpy, Pandas, imblearn, sklearn, matplotlib
- OPERATING SYSTEM: Any 64-bit windows 7 and above, MAC OS, LINUX
- PROCESSOR: 1.5GHz and ABOVE
- RAM: 4GB and above
- HARDDISK: 8GB and above

4.4 LIBRARIES

4.4.1 Numpy

This is a package that contains multidimensional array elements as well as a set of procedures for managing such arrays. NumPy can conduct arithmetic and scientific calculations on arrays.

Installation: pip install numpy

4.4.2 Pandas

Pandas is utilized in a variety of scientific and corporate disciplines such as banking, marketing, statistical analysis, and so on. This is a quick and effective data frame with indexing as well. Even used to import data from various file types into data frame objects.

Installation: pip install pandas

4.4.3 imblearn

This package is often used to counterbalance the unbalanced dataset it includes many components used to either to undersample or to oversample the existing data. Now in this research scenario we utilized SMOTE component to oversample the existing dataset and make it balanced.

Installation: pip install imblearn

4.4.4 sklearn

A scikit is a tailored framework used for specialized tasks like machine learning or data analysis. Scikit-learn is a specialized package that are utilized for these objectives. This package offers collections of helpful algorithms needed to manage all mechanisms involved in machine learning model training.

Installation: pip install -U scikit-learn

4.4.5 matplotlib

Matplotlib is indeed a Python 2D charting package that generates high-quality charts in a range of printed and dynamic formats among platforms. Matplotlib is compatible with Python scripts, Jupyter notebooks, web app servers and GUI toolkits.

Installation: pip install -U matplotlib

5 ALGORITHMS METHODOLOGY:

- DECISION TREE CLASSIFIER
- RANDOM FOREST CLASSIFIER
- LOGISTIC REGRESSION
- KNN CLASSIFIER
- XGBOOST CLASSIFIER
- LINEAR SUPPORT VECTOR CLASSIFIER
- ANN

5.1.1 DECISION TREE CLASSIFIER

Decision tree algorithm may be utilized for classification and also regression issues. The term itself indicates that it employs a flow resembling a tree structure to depict the projections. It begins with a root of the tree and finishes with a choice made by the leaves of the tree.

5.1.1.1 ENTROPY:

- Entropy is the level of uncertainty in the dataset or a metric of chaos.

$$E(S) = -p_{(+)}\log p_{(+)} - p_{(-)}\log p_{(-)}$$

5.1.1.2 INFORMATION GAIN:

- Information gain quantifies the decrease of ambiguity associated with a particular attribute and serves as a decisive factor in determining which attribute must be chosen as a root node of the tree.

$$Information\ Gain = E(Y) - E(Y|X)$$

5.1.2 RANDOM FOREST CLASSIFIER

- Random Forest algorithm is a brand name for a collection of decision trees, hence the name "Forest". For classifying a new instance based on properties, each decision tree provides a classification, which we call a "vote" for that specific class. The categorization with the highest votes is chosen by the forest.

5.1.2.1 STEPS:

• Step-1: In Random Forest, n random entries are selected at random from the data collection of k records.
• Step-2: For each sample, individual decision trees are built.
• Step-3: Every decision tree algorithm would provide a classification.
• Step-4: For Classification and Regression, the end result is based on Majority Rule or Averaging of the values.

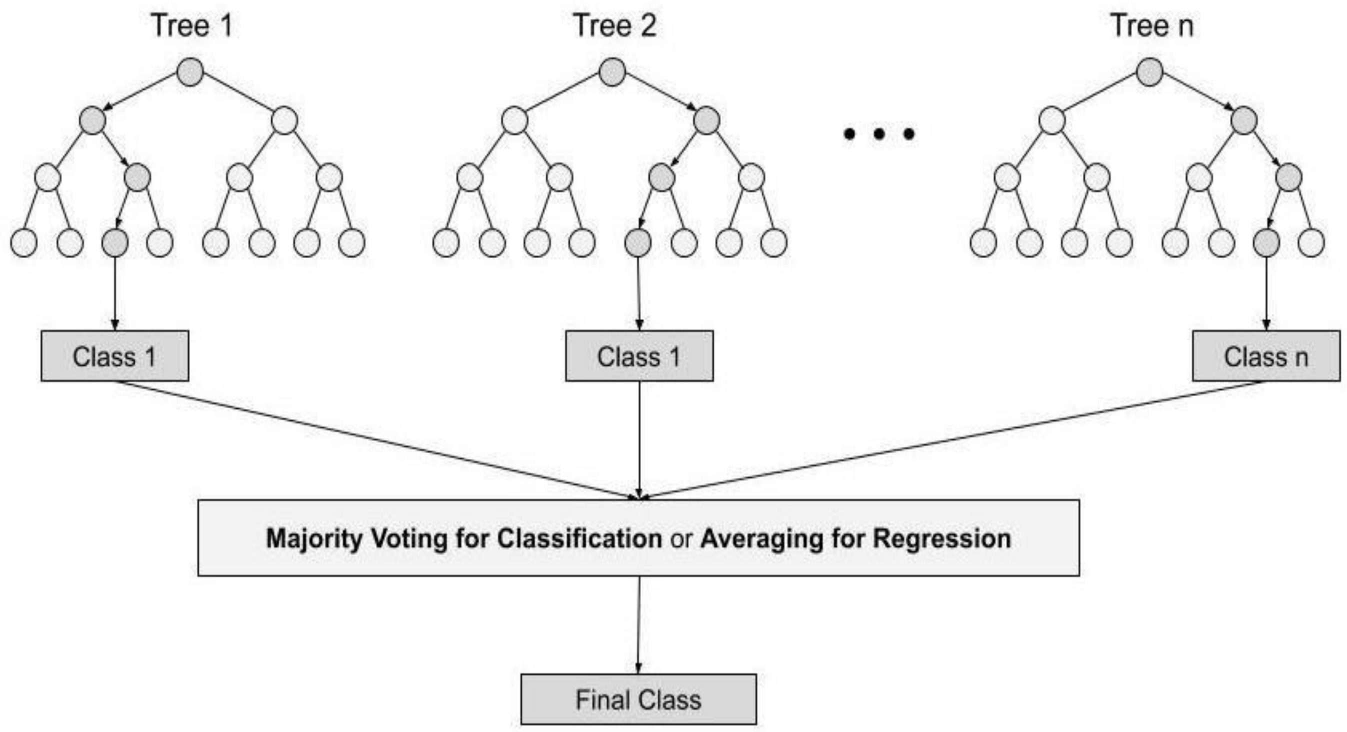

Tree 1
Tree 2
Tree n
Class 1
Class 1
Class n
Majority Voting for Classification or Averaging for Regression
Final Class

Figure 1: Random Forest Structure

5.1.3 LOGISTIC REGRESSION

This is a classification technique but not a regression method. It can be used to predict distinct values (binary values such as 0 or 1 and "yes" or "no" and true or false) based on a set of independent variables (s). Its result ranges from 0 to 1.

The linear equation may be expressed formally as follows.

$$z = \beta 0 + \beta 1x1 + \beta 2x2 + \ldots\ldots..+ \beta nxn$$

5.1.3.1 SIGMOID FUNCTION:

This projected response value, which is represented by "z", is then transformed into a probability value ranging from 0 to 1. In order to transfer expected values to probability values, we employ the sigmoid function. After that, the sigmoid function converts any actual number to a probability ranging value between 0 and 1.

$$\phi(z) = \frac{1}{1 + e^{-z}}$$

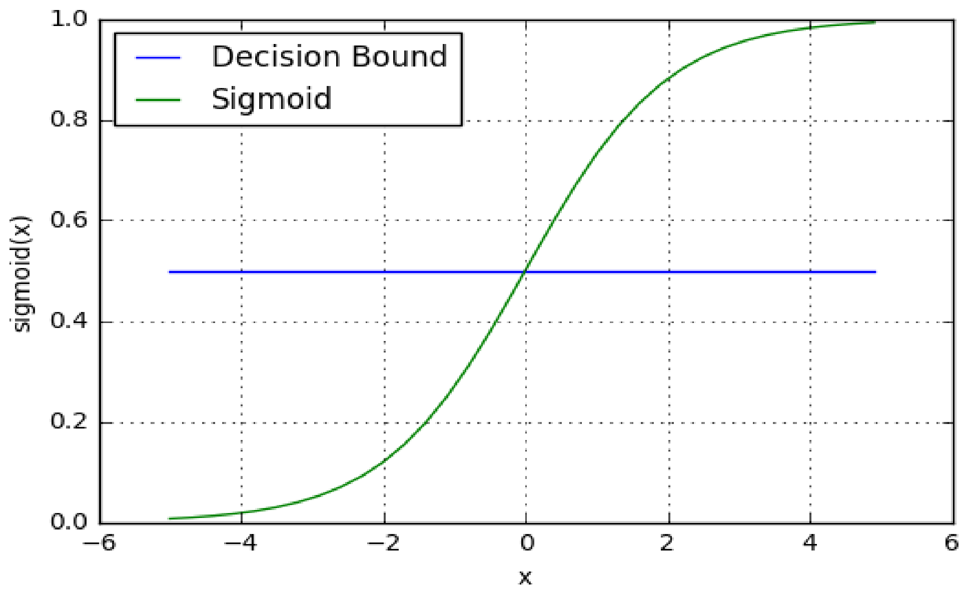


Figure 2: Sigmoid Function graph

5.1.4 KNN CLASSIFIER

In the industry, it is more often employed in categorization difficulties. The K closest neighbours method is a simple algorithm that maintains all existing examples and classifies new cases based on a majority vote of its k neighbors. The case allocated to the group is the most frequent among its K closest neighbors as determined by a distance measure.

The distance equations may be Euclidean, Manhattan, Minkowski or Hamming. The first three processes can be used for continuous data, while the fourth (Hamming) would be used for categorical attributes. When doing KNN modeling, selecting K might be difficult at times.

Euclidean = sqrt((x2-x1)^2 – (y2-y1)^2)
Manhattan = |(x2-x1)|+|(y2-y1)|

5.1.4.1 STEPS:

o Step 1: Determine the K neighbors.

o Step 2: Determine the Euclidean distance between K neighbors.

o Step 3: Select the K closest neighbors based on the determined Euclidean distance.

o Step 4: Estimate the number of observations in each category among these k neighbors.

o Step 5: Allocate the data points to the group with the greatest number of neighbors.

5.1.5 XGBOOST CLASSIFIER (Proposed algorithm)

The word 'Boosting' refers to the set of methods that transforms poor learners into strong learners.

• The basic learner accepts all of the distributions and gives each observation equal weight or attention.

• If the initial base learning method causes a prediction mistake, we give greater attention to the observations with the prediction error. The following basic learning method is then used.

• Repeat Step 2 until the limit of the basic learning algorithm is met or a better level of accuracy is obtained.

XGBoost is a technique of ensemble methods. XGBoost is a gradient-boosted decision tree implementation optimized for speed and reliability.

STEPS:

- calculate the base probability and then calculate residual values for all rows.
- Now construct the decision tree by splitting the data in the way that the calculated gain must be high as possible where gain = left similarity score + right ss – root ss.
- Now check pruning, if gain is less than cover value (pr(1-pr)) then cut the branch.
- When new data comes, find the output of base model first by the formula log(odds)=log(p/1-p).
- And then for the new data go through the splits and check similarity scores of final split and apply sigmoid activation function i.e, sigmoid (base output + learning rate (similarity score)).
  And sigmoid = 1/1-(e^-value). For every sigmoid value new probability will be produced for each record.

5.1.6 LINEAR SVC

SVMs are sophisticated machine learning methods used for classifying problems, regression, and outliers identification. An SVM classifier constructs a model that assigns fresh pieces of data to one of the specified categories.

The Linear Support Vector Classifier (SVC) approach conducts classification using a linear kernel function and works effectively with a large amount of observations. When compared to the SVC model.

- **Hyperplane**

  The SVM classifier uses a hyperplane with the greatest margin to distinguish the data points. The hyperplane is referred to as the decision boundary hyperplane, and the linear classifier it creates is referred to as the maximum margin classifier..

- **Support Vectors**

  The sample data points nearest to the hyperplane are called support vectors. By computing margins, these measured values will help describe the separation line or hyperplane.

- **Margin**

  A margin is the distance between two lines on the closest points. It is determined as the perpendicular distance between the line and the nearest support vectors. In SVMs, we aim to maximize the separation distance in order to increase the margin.

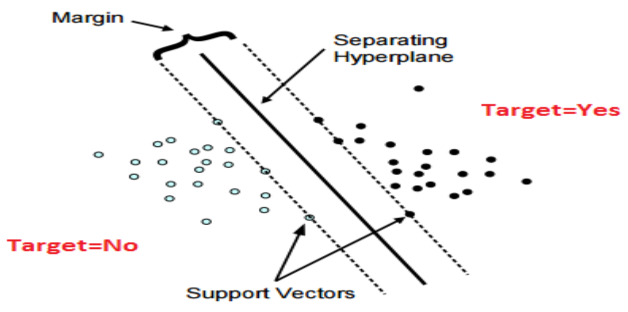
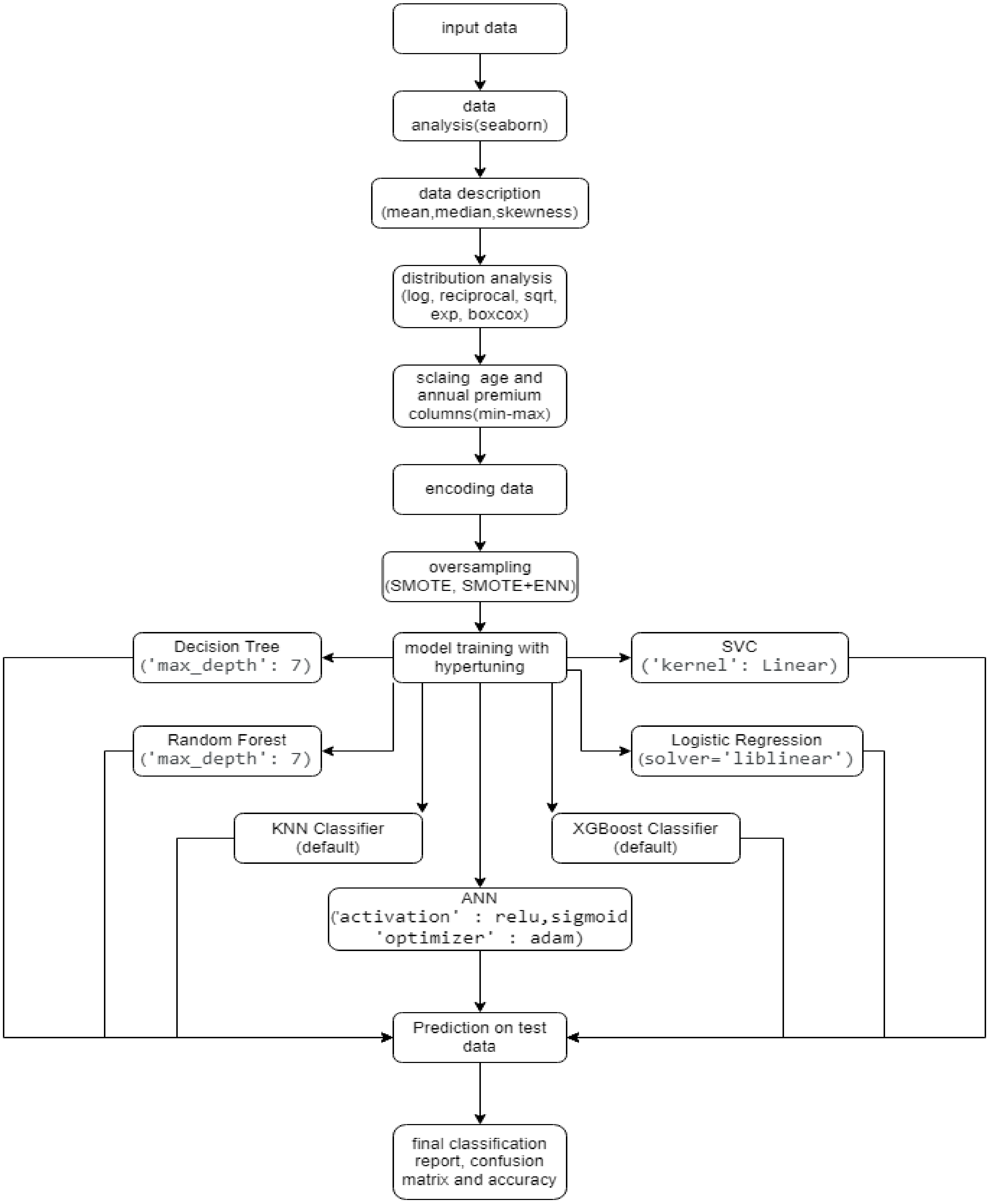


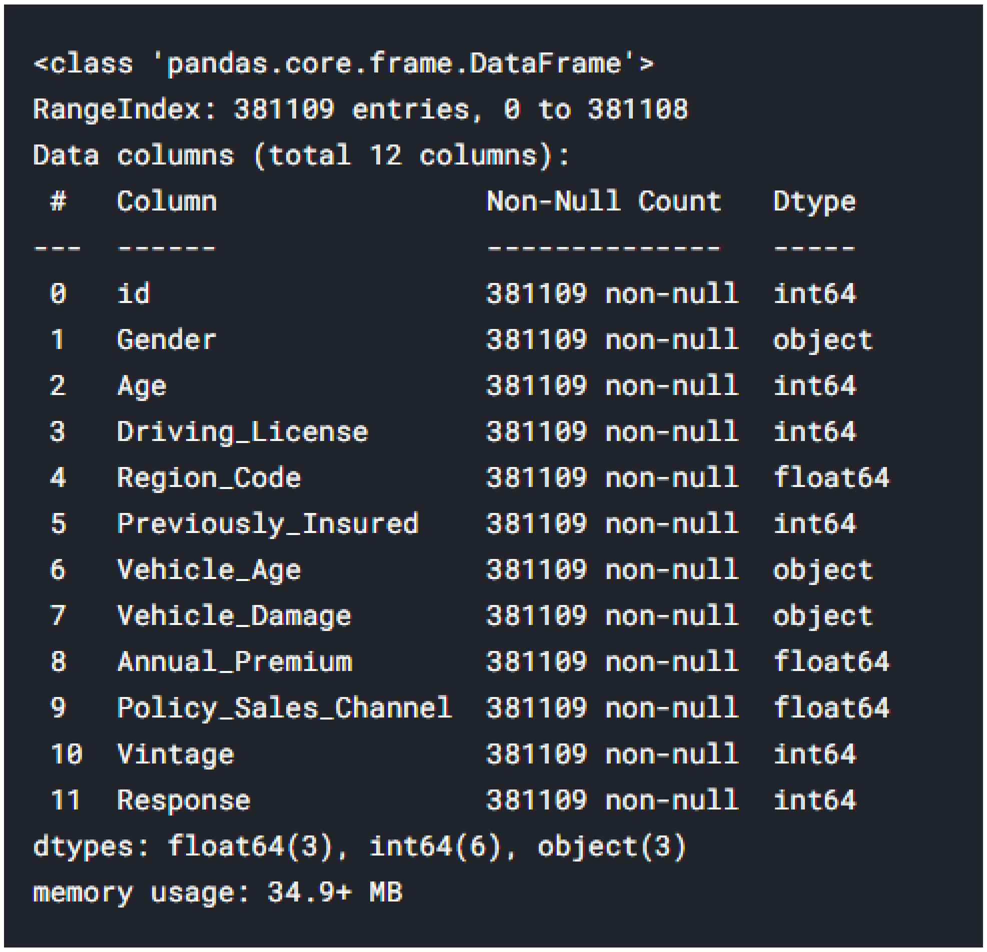
Figure 3: Support Vector Machine

5.1.7 IMPLEMENTATION FLOW CHART


input data
data analysis(seaborn)
data description (mean,median,skewness)
distribution analysis (log, reciprocal, sqrt, exp, boxcox)
sclaing age and annual premium columns(min-max)
encoding data
oversampling (SMOTE, SMOTE+ENN)
model training with hypertuning
Decision Tree ('max_depth': 7)
SVC ('kernel': Linear)
Random Forest ('max_depth': 7)
Logistic Regression (solver='liblinear')
KNN Classifier (default)
XGBoost Classifier (default)
ANN ('activation' : relu,sigmoid 'optimizer' : adam)
Prediction on test data
final classification report, confusion matrix and accuracy

5.2 EXPLORATORY DATA ANALYSIS

Below are the attributes of the dataset taken

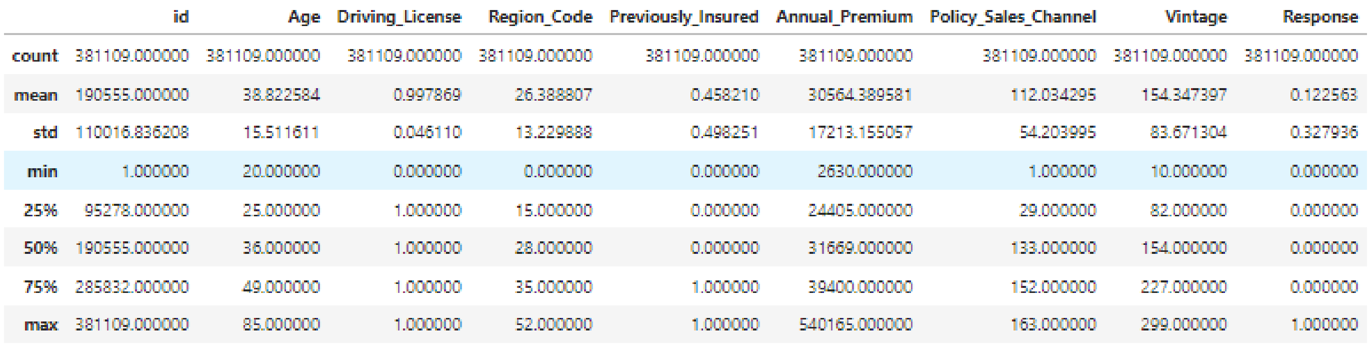


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 381109 entries, 0 to 381108
Data columns (total 12 columns):
 #   Column                Non-Null Count   Dtype  
---  ------                --------------   -----  
 0   id                    381109 non-null  int64  
 1   Gender                381109 non-null  object 
 2   Age                   381109 non-null  int64  
 3   Driving_License       381109 non-null  int64  
 4   Region_Code           381109 non-null  float64
 5   Previously_Insured    381109 non-null  int64  
 6   Vehicle_Age           381109 non-null  object 
 7   Vehicle_Damage        381109 non-null  object 
 8   Annual_Premium        381109 non-null  float64
 9   Policy_Sales_Channel  381109 non-null  float64
 10  Vintage               381109 non-null  int64  
 11  Response              381109 non-null  int64  
dtypes: float64(3), int64(6), object(3)
memory usage: 34.9+ MB
```

- TOTAL COLUMNS: 12
- CATEGORICAL COLUMNS: GENDER, VEHICLE_AGE, VEHICLE_DAMAGE
- NUMERICAL COLUMNS: ID, AGE, DRIVING_LICENSE, REGION_CODE, PREVIOUSLY_ISURED, ANNUAL_PREMIUM, POLICY_SALES_CHANNEL, VINTAGE, RESPONSE(TARGET).
- NO NULL VALUES
- VALUE COUNTS: RESPONSE 0 = 334399, RESPONSE 1 = 46710 (IMBALANCED)

<u>5.2.1 CONTINOUS VARIABLE ANALYSIS</u>

In the picture below, a histogram graph of the age attribute reveals the distribution, so there is a right-skewed distribution indicating that the occurrence of young policy holders (20-30 years old) observations is much higher than that of older policy holders in the dataset. The Annual Premium characteristic similarly has a slightly right-skewed distribution considering the outliers to be removed or can be ignored as they are few.



	id	Age	Driving_License	Region_Code	Previously_Insured	Annual_Premium	Policy_Sales_Channel	Vintage	Response
count	381109.000000	381109.000000	381109.000000	381109.000000	381109.000000	381109.000000	381109.000000	381109.000000	381109.000000
mean	190555.000000	38.822584	0.997869	26.388807	0.458210	30564.389581	112.034295	154.347397	0.122563
std	110016.836208	15.511611	0.046110	13.229888	0.498251	17213.155057	54.203995	83.671304	0.327936
min	1.000000	20.000000	0.000000	0.000000	0.000000	2630.000000	1.000000	10.000000	0.000000
25%	95278.000000	25.000000	1.000000	15.000000	0.000000	24405.000000	29.000000	82.000000	0.000000
50%	190555.000000	36.000000	1.000000	28.000000	0.000000	31669.000000	133.000000	154.000000	0.000000
75%	285832.000000	49.000000	1.000000	35.000000	1.000000	39400.000000	152.000000	227.000000	0.000000
max	381109.000000	85.000000	1.000000	52.000000	1.000000	540165.000000	163.000000	299.000000	1.000000

Figure 4: Table Description

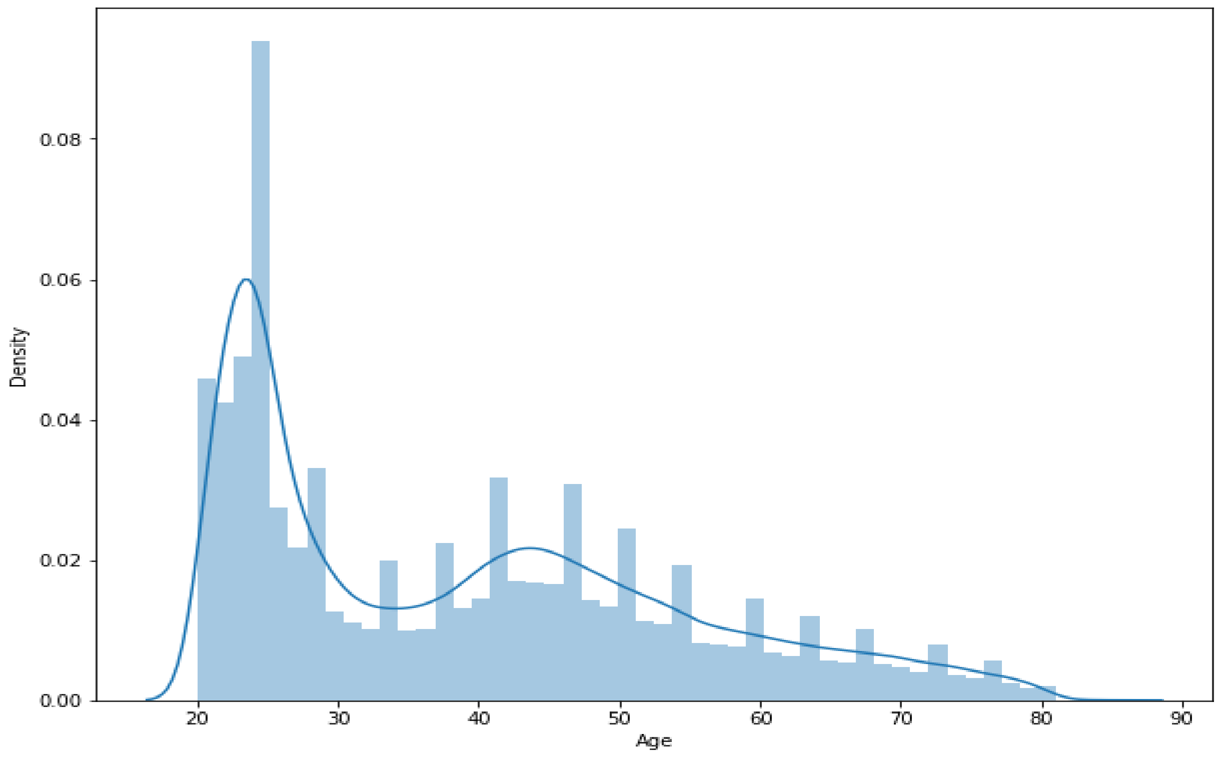


Figure 5: Age column distribution

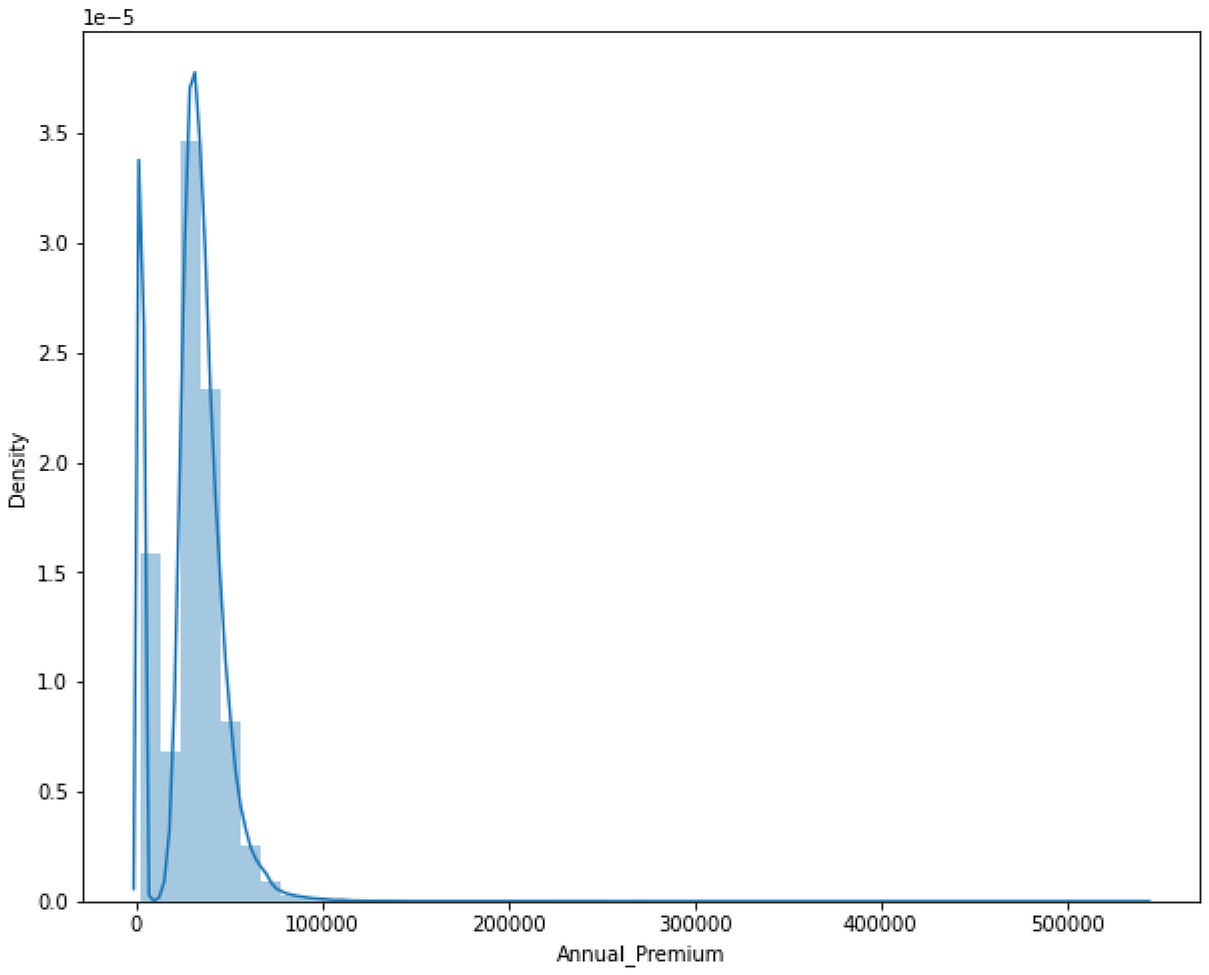


Figure 6: Annual Premium column distribution

5.2.2 MIN MAX SCALING

In min-max scaling, you deduct the dataset's minimal value from all of the values and divide the result by the dataset's ranges (maximum-to-minimum). In this scenario, your dataset will always be within 0 and 1, when it was before within -1 and +1. This method, once again, is vulnerable to outliers. Min-Max scaling is critical for avoiding bias induced by variables described as distinct scales and having varying contributions in the models.

```
max_age=max(data.Age)
min_age=min(data.Age)
data.Age=data.Age.apply(lambda x: (x-min_age)/(max_age-min_age))
```

```
max_premium=max(data.Annual_Premium)
min_premium=min(data.Annual_Premium)
data.Annual_Premium=data.Annual_Premium.apply(lambda x: (x-min_premium)/(max_premiu
m-min_premium))
```

5.2.3 CATEGORICAL VARIABLE ANALYSIS

This dataset includes 7 distinct parameters such as Gender, Vehicle Age, Driving License, Previously Insured, Region Code, and Policy Sales Channel, and the analysis of these columns is studied with respect to the Response feature which is the target value.

Gender vs Response:

The below figure shows the bar graph between Gender and Response. It is clear that more male candidates are likely to make response either 1 or 0 than female candidates.

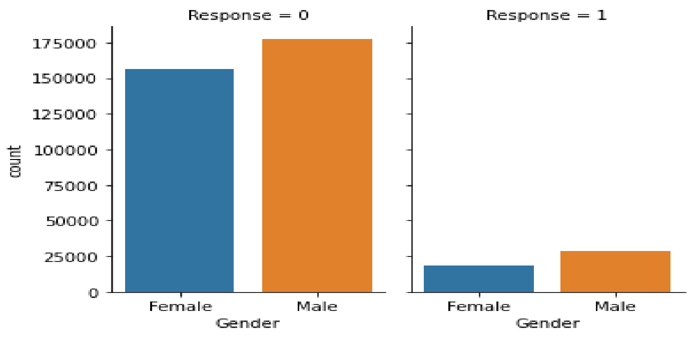


Figure 7: Gender vs Response Bar chart

Response and Vehicle age:

From the below bar chart, it is clear that vehicle age with 1-2years is likely to take insurance.

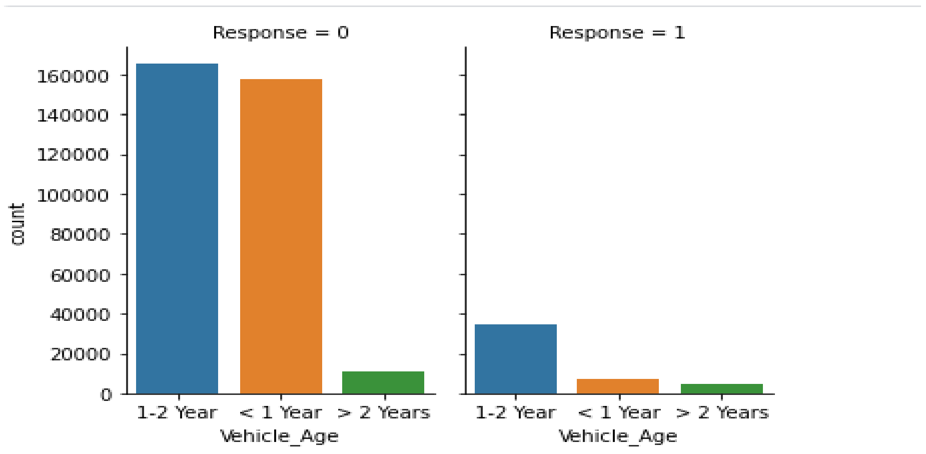


Figure 8: Response vs Vehicle Age Bar chart

5.2.4 CORRELATION FOR FEATURE SELECTION

Correlation is a statistical tool for determining the correlation value of linear relationship between the connection between two parameters and calculate their correlation. A strong correlation indicates a strong association between the two attributes, whereas a low correlation suggests that the variables are weakly associated. There is indeed a positive correlation between two attributes when a rise in one variable leads to the increase in another. But at the other side, a negative correlation indicates that even when one variable grows, the other attribute falls and vice-versa.

Below is the correlation heat map of all the attributes of the dataset. From this it is clear that that all the attributes except vintage column having either positive or negative correlation on the target variable (Response).

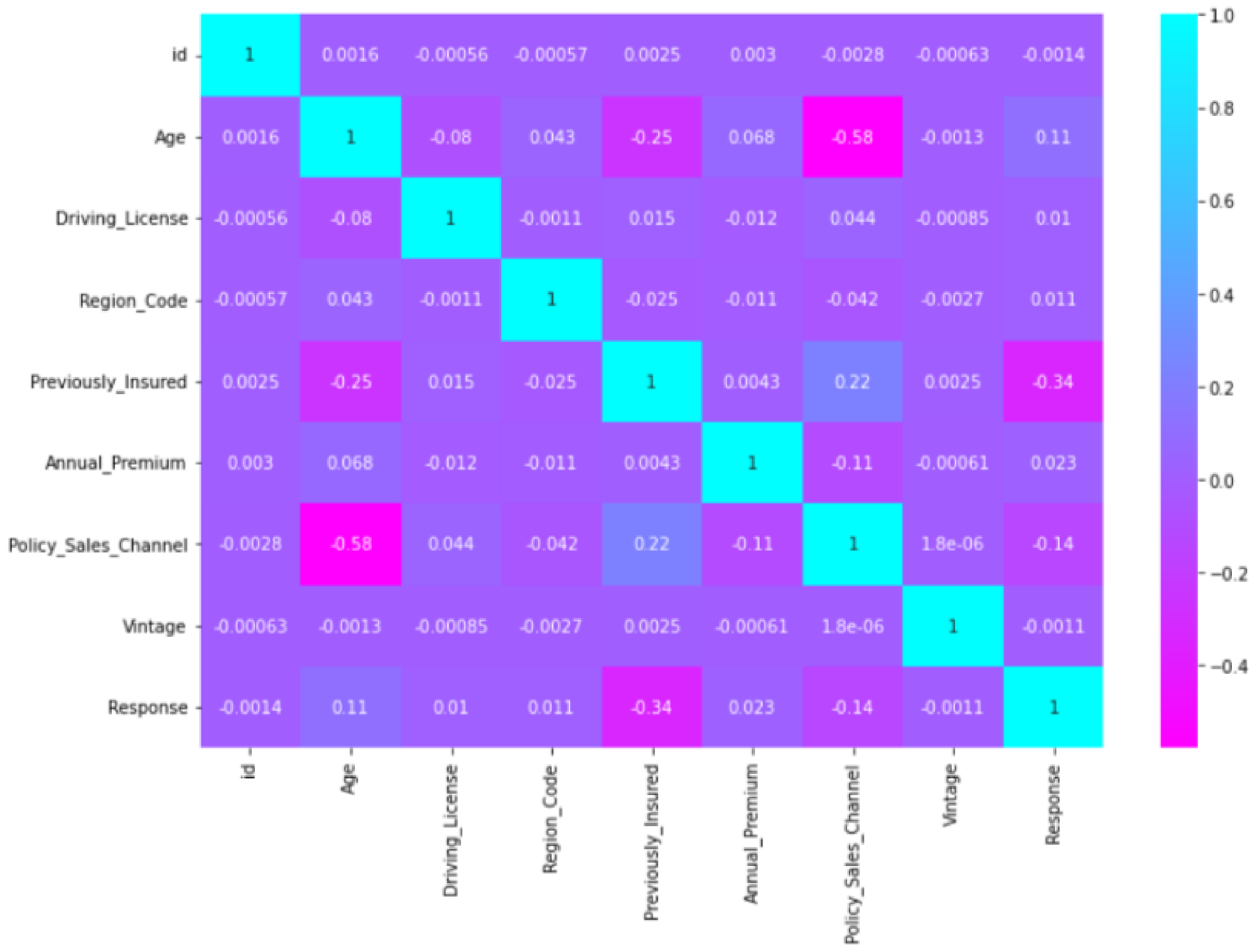


Figure 9: Correlation Heatmap

5.2.5 LABEL ENCODING

In Python, label encoding means the category value is replaced by a numerical range between 0 and the total number of classes - 1. Label encoding using pandas for the categorical variables Gender, Vehicle age, and Vehicle damage is shown below.

```
data.Gender=pd.Categorical(data.Gender,categories=['Male','Female'],ordered=True).c
odes
data.Vehicle_Age=pd.Categorical(data.Vehicle_Age,categories=['1-2 Year','< 1 Yea
r','> 2 Years'],ordered=True).codes
data.Vehicle_Damage=pd.Categorical(data.Vehicle_Damage,categories=['Yes','No'],orde
red=True).codes
```

6 MODELLING AND TRAINING

6.1 SMOTE OVERSAMPLING THE DATA

In this strategy, we raise the sample size of the minority class in order to make the sample sizes of the minor and major classes equal.
One possibility is to reproduce the minority class's samples, while another is to construct some synthetic points using SMOTE.

SMOTE (Synthetic Minority Over-sampling Technique) starts by choosing a minority class data point at randomly and determining its k nearest minority class neighbours. The synthetic sample is then produced by picking one of it's k closest neighbours "b" randomly and joining "a" and "b" in the feature forming a line segment. The synthetic variants are constructed by convexly joining the two chosen instances "a" and "b".

6.2 HYPERTUNING

Hyper - parameters are parameters that can be commonly provided while building the Machine Learning model. As a result, hyper - parameters are supplied before values, or we may say that hyper - parameters are used to evaluate optimum parameters of the machine learning model. The most appealing aspect of hyper - parameters is that its parameters are specified by the person creating the model.

Grid Search CV assesses the performance for every permutation of all the specified hyper - parameters and their values and returns the optimal value for the hyper - parameters. Because of the number of hyper - parameters concerned, the procedure is time-taking and expensive.

Grid Search CV is a model evaluation stage that comes after the Data Processing jobs. It is usually beneficial to test the efficiency of Optimized and Untuned ML Models. It will take time and resources, but it will undoubtedly provide the finest results.

6.3 FEATURE TRANSFORMATION

Amongst the most crucial stages in constructing a Machine learning model is feature preprocessing. If your model has too few attributes, it won't have anything to learn from. We may be offering the model excessive information if we have too many attributes. Not just that, but still the values for each of the areas must also be examined.

It is known that there are several specified standards for handling categorical data, such as encoding them in different ways. Meanwhile, dealing with continuous data is a large element of the method. There are numerous techniques of dealing with continuous variables. Transforming them to a gaussian distribution, for example, is one of them.

The MinMax scaling technique is one of the simplest scalers to comprehend. It simply adjusts all ranging between zero and 1. The equation for determining the scaled values is as follows:

x_scaled = (x – x_min)/(x_max – x_min)

6.4 TENSOFLOW

Keras is a Python-based high-level neural network API designed to allow for rapid experimentation. I Keras provides a uniform and straightforward API, lowering the number of user actions required for normal use cases and delivering clear and actionable information in the case of user error. Keras can operate on top of a variety of deep learning backends, including TensorFlow, CNTK, and Theano. This feature enables the Keras model to be portable across all of these backends.

It is a simple and straightforward model. A tf.keras.Model is composed of a linear stack of methods that group a linear stack of layers. The major duty, as implied by its name, is to organise the layers of the Keras in a sequential order.

The data in this paradigm flows from one layer to the next. The data flow is maintained until the data reaches the last layer. The Sequential API Model is used by the majority of ANNs.

Dense layer is the usual strongly linked NN layer. It is most popular and commonly utilised layer. Dense layer executes the below action on the inputs and produce the result.

6.4.1 RELU ACTIVATION

The rectified linear activation function, abbreviated ReLU, is a piece - wise linear function that produces the input directly if it is affirmative; else, it produces zero. It's the standard activation function for several kinds of neural network models as it is quicker to train and often results in superior performance.

In order to train DNNs using stochastic gradient with backpropagation of mistakes, an activation function that looks and behaves like a linear model but is actually a nonlinear function that permits intricate relationship between the data to be learned is needed.

A ReLU is a node or unit that executes this activation function. Rectified network is a network that utilise the rectifier function for their hidden units.

Deployment of ReLU is commonly viewed as one of the few watershed points in the deep learning era, alongside technologies that currently allow for the daily development of extraordinarily DNNs.

6.4.2 SIGMOID ACTIVATION

The sigmoid function is exploited as an activation function. An activation function is applied to a weighted combination of inputs, and the output is utilized as an input to the subsequent layer.

Whenever a neuron's activation function is a sigmoid, the outcome of this unit is always within 0 and 1. Furthermore, since the sigmoid is indeed non-linear the outcome of this component is also a non-linear function of the weighted combination of inputs. It utilizes a sigmoid activation function as primary function.

6.4.3 ADAM OPTIMIZER

Adam may be regarded of as a mix of RMSprop and the SGD with momentum. It adjusts the learning rate by squaring gradients, similar to RMSprop, and it makes advantage of momentum by employing the moving average of the gradient instead of the gradient directly, comparable to SGD with momentum.

Adam is an adaptive learning rate approach, which implies that it calculates individual learning rates for various parameters. Its name is derived from adaptable moment estimation, and Adam employs estimates of the first and second moments of gradient to change the learning rate for each weight of the neural network. The anticipated value of a random variable to the power of n is defined as its N-th moment.
It is worth noting that the gradient of the cost function of a neural network can be regarded a random variable because it is often assessed on a tiny random batch of data. The first instant represents mean, while the second moment is uncentered variance (we don't remove the mean when calculating variance). We shall see how we use these values later; for now, we must decide how to obtain them. Adam employs exponentially moving averages to estimate the moments, which are computed on the gradient assessed on a current mini-batch:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

6.5 CLASSIFICATION REPORT

This is one of the measurements used to assess the effectiveness of a classification of the machine learning model. It represents accuracy, recall, F1 score, precision and support of your model. It helps us to get a better understanding of our trained model's general performance.

These were the four approaches to judge whether the predictions are true or erroneous

1. True Negative - an instance was negative and projected negative.
2. True Positive - an instance was positive and projected positive.
3. False Negative - an instance was positive and anticipated negative.
4. False Positive - an instance was negative and anticipated positive.

Precision is a classifier's way to ignore identifying a negative event as positives. It is defined for each category as the fraction of true positives to the total of true positives as well as the false positives.

Precision = TP/(TP + FP)

Recall is the capacity of a classifiers to discover all the positive occurrences. It is defined as the fraction of true positives to the total of true positives as well as the false negatives.

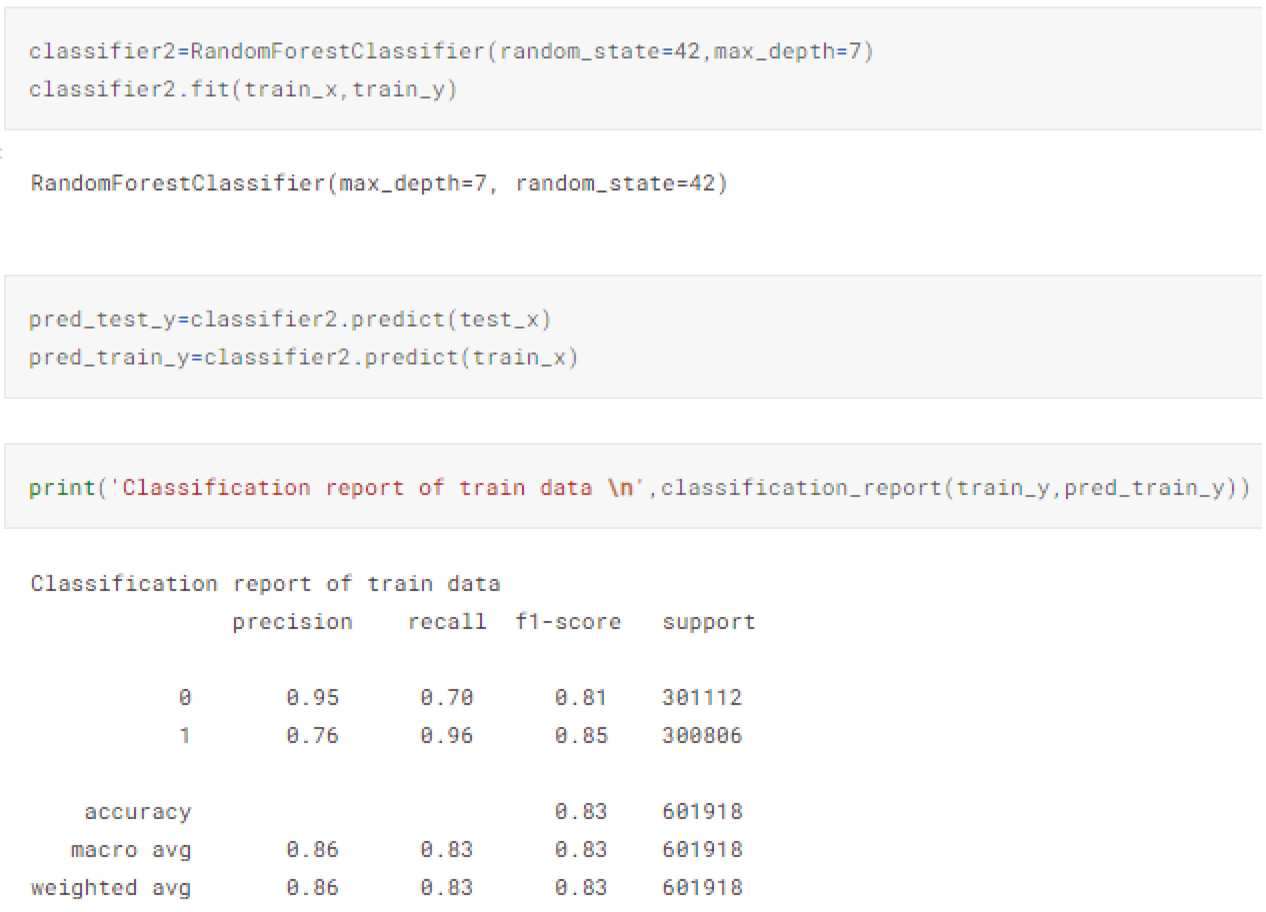
Recall = TP/(TP+FN)

The F1 score is the weighted harmonic mean of accuracy and recall, with 1 being the greatest and 0 being the shortest. F1 scores are frequently lower than accuracy assessments because they integrate precision and recall into their calculation.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

7. RESULTS AND DISCUSSION

7.1 Random Forest Accuracy



```
classifier2=RandomForestClassifier(random_state=42,max_depth=7)
classifier2.fit(train_x,train_y)
```

```
RandomForestClassifier(max_depth=7, random_state=42)
```

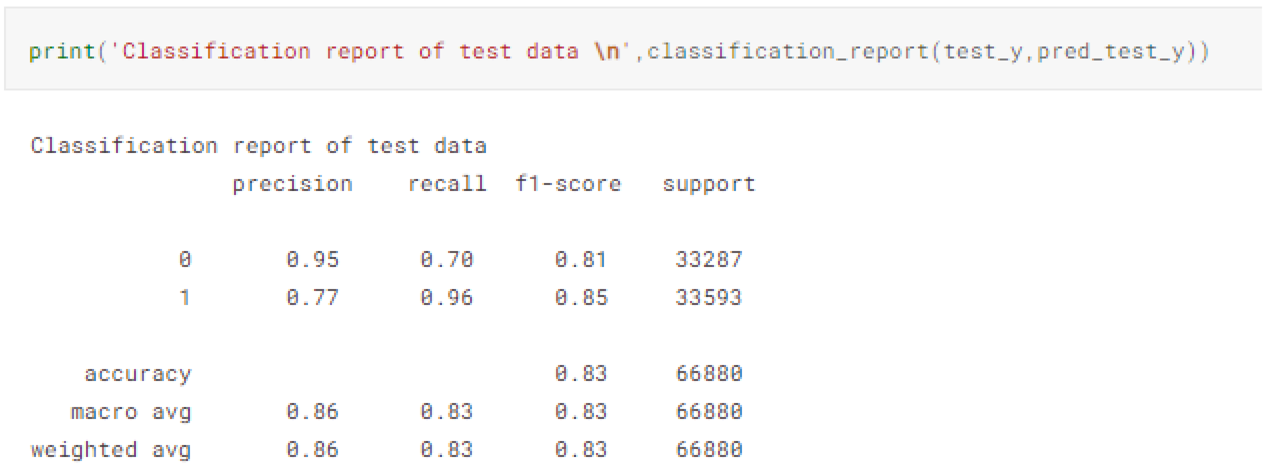
```
pred_test_y=classifier2.predict(test_x)
pred_train_y=classifier2.predict(train_x)
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.95      0.70      0.81    301112
           1       0.76      0.96      0.85    300806

    accuracy                           0.83    601918
   macro avg       0.86      0.83      0.83    601918
weighted avg       0.86      0.83      0.83    601918
```



```
print('Classification report of test data \n',classification_report(test_y,pred_test_y))
```

```
Classification report of test data
               precision    recall  f1-score   support

           0       0.95      0.70      0.81     33287
           1       0.77      0.96      0.85     33593

    accuracy                           0.83     66880
   macro avg       0.86      0.83      0.83     66880
weighted avg       0.86      0.83      0.83     66880
```

7.2 XGBoost Accuracy

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.300000012, max_delta_step=0, max_depth=6,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=100, n_jobs=2, num_parallel_tree=1, random_state=0,
              reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=1,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

```
pred_test_y=classifier5.predict(test_x)
pred_train_y=classifier5.predict(train_x)
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.88      0.96      0.92    301112
           1       0.96      0.87      0.91    300806

    accuracy                           0.92    601918
   macro avg       0.92      0.92      0.92    601918
weighted avg       0.92      0.92      0.92    601918
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.88      0.96      0.92    301112
           1       0.96      0.87      0.91    300806

    accuracy                           0.92    601918
   macro avg       0.92      0.92      0.92    601918
weighted avg       0.92      0.92      0.92    601918
```

7.3 Linear SVC Accuracy

```
classifier8=LinearSVC()
classifier8.fit(train_x,train_y)
```

```
LinearSVC()
```

```
pred_test_y=classifier8.predict(test_x)
pred_train_y=classifier8.predict(train_x)
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.99      0.59      0.74    301112
           1       0.71      1.00      0.83    300806

    accuracy                           0.79    601918
   macro avg       0.85      0.79      0.78    601918
weighted avg       0.85      0.79      0.78    601918
```

```
print('Classification report of test data \n',classification_report(test_y,pred_test_y))
```

```
Classification report of test data
               precision    recall  f1-score   support

           0       0.99      0.59      0.74     33287
           1       0.71      1.00      0.83     33593

    accuracy                           0.80     66880
   macro avg       0.85      0.79      0.79     66880
weighted avg       0.85      0.80      0.79     66880
```

7.4 KNN Accuracy

```
classifier4=KNeighborsClassifier()
classifier4.fit(train_x,train_y)
```

```
KNeighborsClassifier()
```

```
classifier4=KNeighborsClassifier()
classifier4.fit(train_x,train_y)
```

```
KNeighborsClassifier()
```

```
pred_test_y=classifier4.predict(test_x)
pred_train_y=classifier4.predict(train_x)
```

```
print('Classification report of test data \n',classification_report(test_y,pred_test_y))
```

```
Classification report of test data
               precision    recall  f1-score   support

           0       0.94      0.74      0.83     33287
           1       0.79      0.95      0.86     33593

    accuracy                           0.85     66880
   macro avg       0.86      0.85      0.85     66880
weighted avg       0.86      0.85      0.85     66880
```

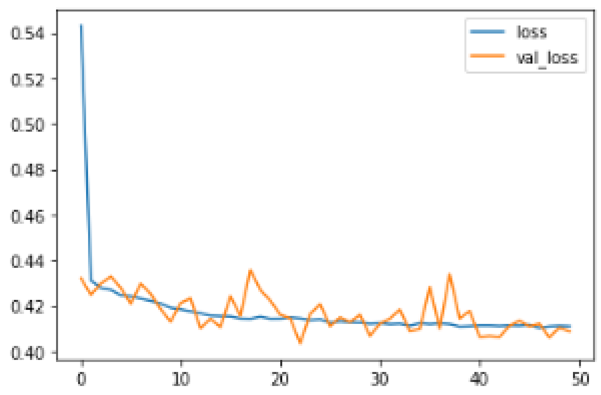
```
print(confusion_matrix(pred_test_y, test_y))
```

```
[[24797  1594]
 [ 8490 31999]]
```

7.5 ANN Accuracy

```
model_loss = pd.DataFrame(model.history.history)
model_loss.plot()
```

```
<AxesSubplot:>
```



```
pred = model.predict(test_x)
pred = np.round(pred)
print(confusion_matrix(pred, test_y))
```

```
[[20629   323]
 [12658 33270]]
```

```
print(classification_report(test_y, pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.62      0.76     33287
           1       0.72      0.99      0.84     33593

    accuracy                           0.81     66880
   macro avg       0.85      0.81      0.80     66880
weighted avg       0.85      0.81      0.80     66880
```

7.6 Decision Tree Accuracy

```
classifier1=DecisionTreeClassifier(random_state=42,max_depth=7)
classifier1.fit(train_x,train_y)
```

```
DecisionTreeClassifier(max_depth=7, random_state=42)
```

```
pred_test_y=classifier1.predict(test_x)
pred_train_y=classifier1.predict(train_x)
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.92      0.73      0.81    301112
           1       0.77      0.94      0.85    300806

    accuracy                           0.83    601918
   macro avg       0.85      0.83      0.83    601918
weighted avg       0.85      0.83      0.83    601918
```

```
print('Classification report of test data \n',classification_report(test_y,pred_test_y))
```

```
Classification report of test data
               precision    recall  f1-score   support

           0       0.92      0.73      0.81     33287
           1       0.78      0.94      0.85     33593

    accuracy                           0.83     66880
   macro avg       0.85      0.83      0.83     66880
weighted avg       0.85      0.83      0.83     66880
```

7.7 Logistic regression Accuracy

```
classifier3=LogisticRegression(solver='liblinear')
classifier3.fit(train_x,train_y)
```

```
LogisticRegression(solver='liblinear')
```

```
pred_test_y=classifier3.predict(test_x)
pred_train_y=classifier3.predict(train_x)
```

```
print('Classification report of train data \n',classification_report(train_y,pred_train_y))
```

```
Classification report of train data
               precision    recall  f1-score   support

           0       0.97      0.61      0.75    301112
           1       0.71      0.98      0.83    300806

    accuracy                           0.79    601918
   macro avg       0.84      0.79      0.79    601918
weighted avg       0.84      0.79      0.79    601918
```

```
print('Classification report of test data \n',classification_report(test_y,pred_test_y))
```

```
Classification report of test data
               precision    recall  f1-score   support

           0       0.96      0.61      0.75     33287
           1       0.72      0.98      0.83     33593

    accuracy                           0.80     66880
   macro avg       0.84      0.79      0.79     66880
weighted avg       0.84      0.80      0.79     66880
```

From all the above accuracies it can be concluded that the random forest classifier and XGBoost classifier gives almost equal accuracies with ~99% while even considering the confusion matrix of all the models. Below is the bar graph showing the accuracies achieved by all the trained models,

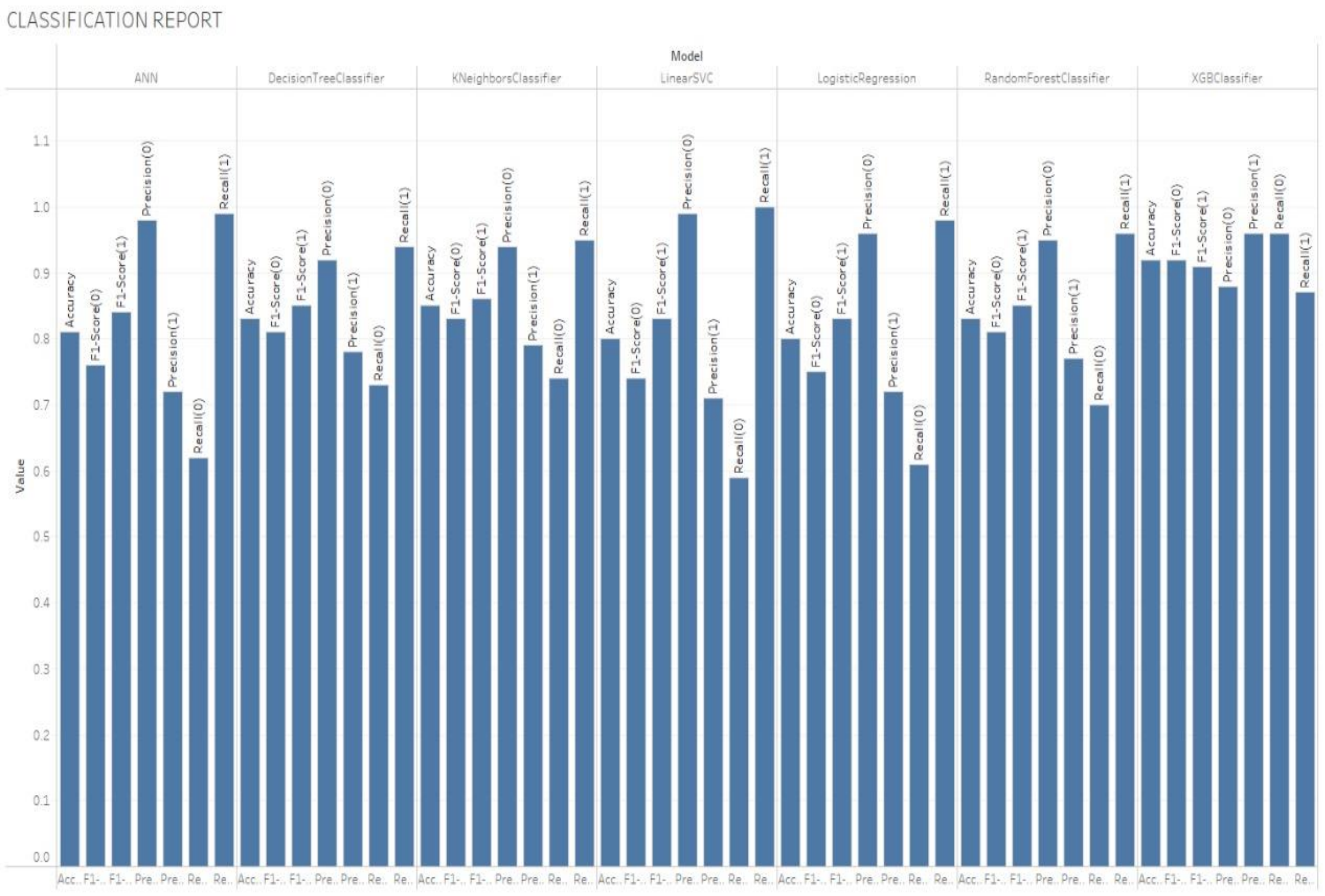
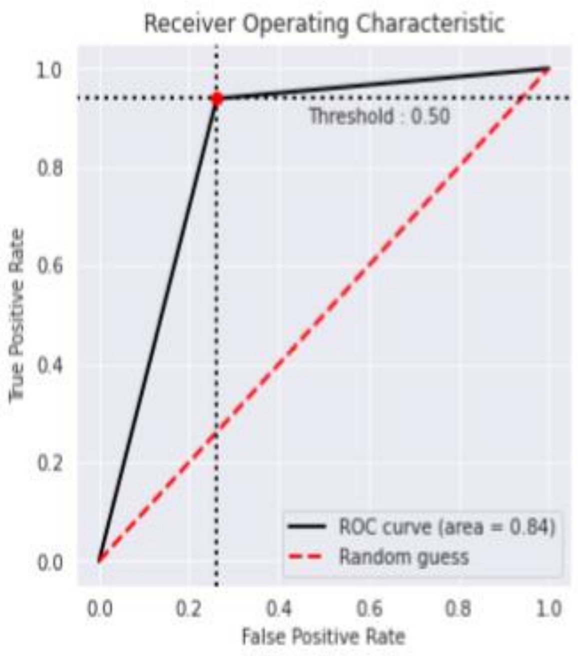
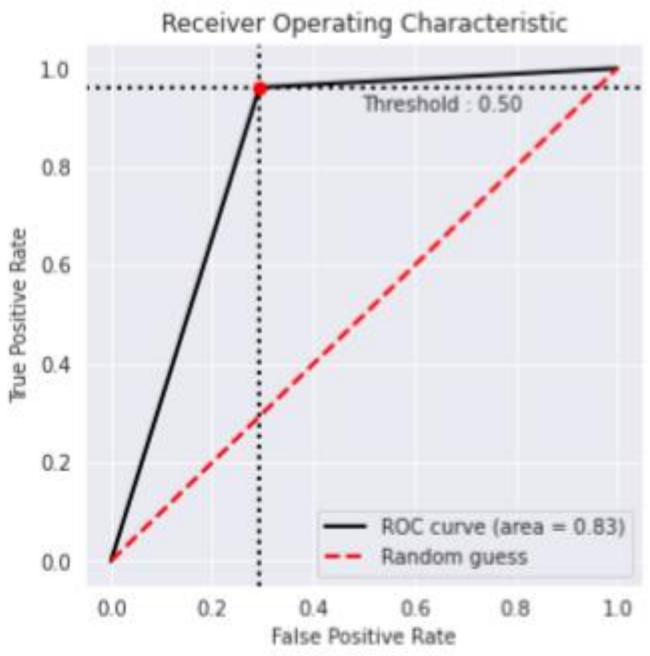


Figure 10: Classification Report bar chart for all trained models

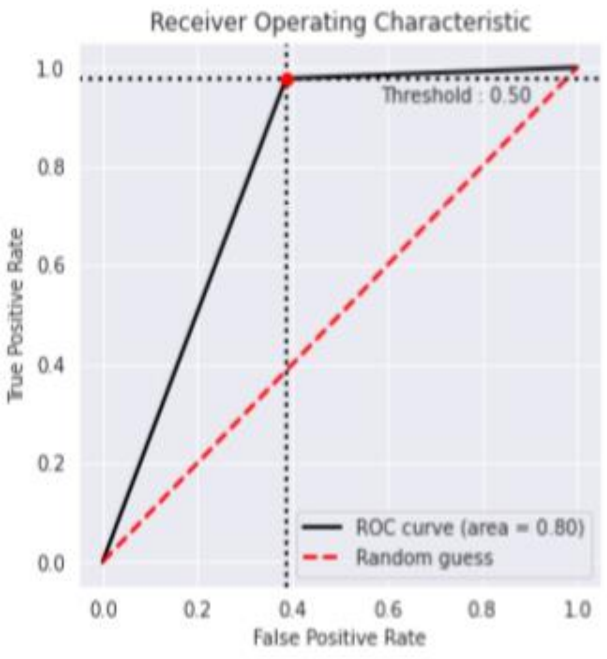
ROC curves of all the trained models are shown below,



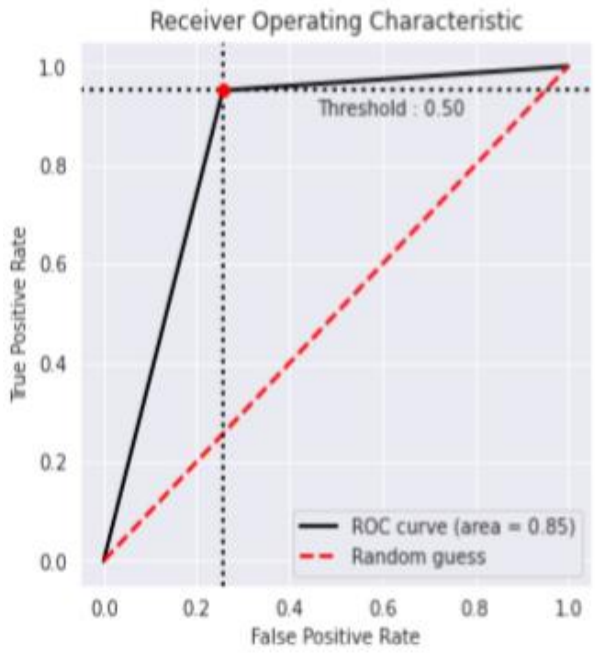
Decision Tree



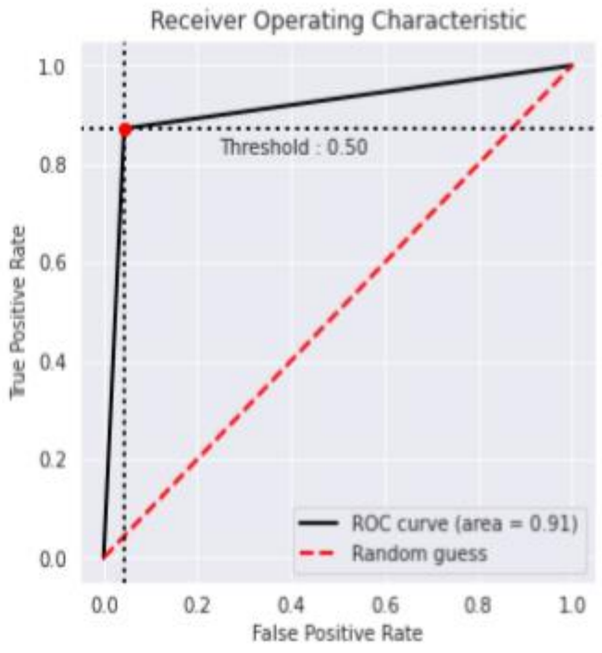
Random Forest



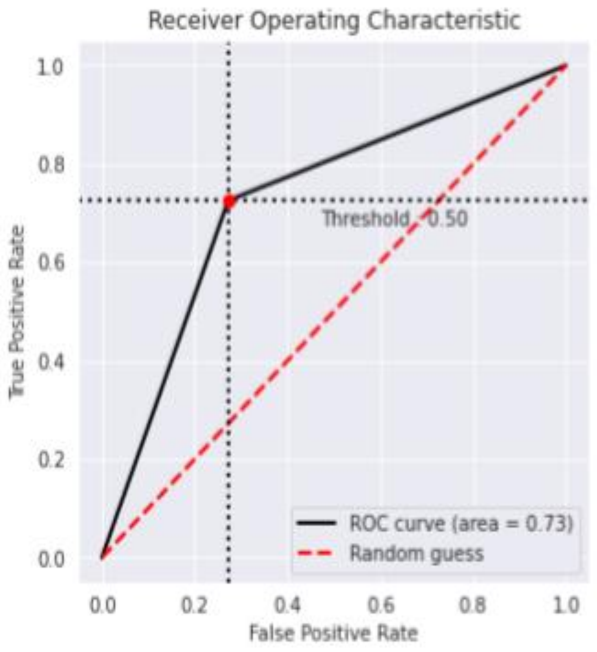
Logistic Regression



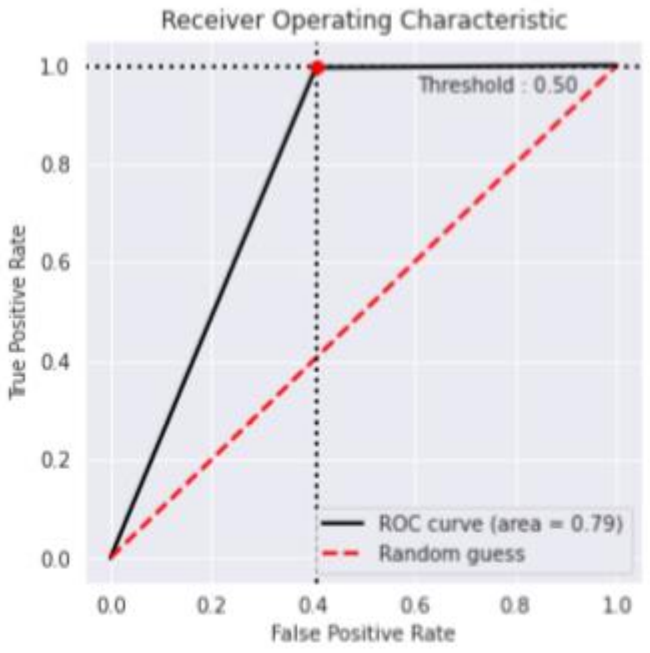
KNN



XGBoost



Support Vector Machine



ANN

8.CONCLUSION

This research is meant to offer comparative findings of machine learning algorithms in order to categorize the samples in terms of eligibility for cross-selling marketing campaigns. There are different pre-processing methods have been applied before constructing the models such as label-encoding, min-max scaling,feature transformation and selection and SMOTE,SMOTE+ENN methods for oversampling the data for dealing with imbalanced feature which is anticipated by the models and hyper-parameter optimization is performed using GridsearchCV for best results. XGBoost agorithms predicts with accuracy above 90% on test data. This research could be furthur extended by implementing other oversampling techniques like SMOTE+Tomek links and ADASYN and also by ensembling different trained models using methods such as stacking and blending which might possibly increase the accuracy of the ensembled model.

9. REFERENCES

1. Impacts of Feature Selection Techniques in Machine Learning Algorithms for Cross Selling: A Comprehensive Study for Insurance Industry (DOI:10.13140/RG.2.2.24254.41284)
2. Prediction of Online Vehicle Insurance System using Decision Tree Classifier and Baye&apos;s Classifier – A comparative Analysis (https://www.ijcaonline.org/proceedings/micro/number1/18308-1804)
3. Machine Learning Approaches for Auto Insurance Big Data (https://doi.org/10.3390/risks9020042)
4. https://www.kaggle.com/anmolkumar/health-insurance-cross-sell-prediction
5. Research on the Features of Car Insurance Data Based on Machine Learning (10.1016/j.procs.2020.02.016)

END