USE CASE STUDY REPORT

Group No.: Group 17

Student Names: Saurav Rohidas Palekar and Devanshu Akhilkumar Chokhani

Executive Summary:

The health insurance is a must as it provides financial protection in case of serious accident and illness. Similarly, the car insurance is also a must in current times. Having a car insurance can help to provide financial protection to repair or service the damage car. We can co-relate health insurance and car insurance with each other. A health insurance company can lift its company's economical assets as well as will gain its reputation if it starts to offer car insurance as well. Building a model to predict whether a customer would be interested in Vehicle Insurance is extremely helpful for the company because it can then accordingly plan its communication strategy to reach out to those customers and optimize its business model and revenue.

The company needs the data to decide whether a person should be given car insurance as well with the health insurance. In this case study we propose a solution which uses machine learning techniques like Classification Trees, Naive Bayes, Logistic regression, and Support Vector Machines to model and classify the data. We have taken the data set from Kaggle and have performed various machine learning techniques and analysis.

I. Background and Introduction

An insurance policy is an arrangement by which a company undertakes to provide a guarantee of compensation for specified loss, damage, illness, or death in return for the payment of a specified premium. A premium is a sum of money that the customer needs to pay regularly to an insurance company for this guarantee. Just like medical insurance, there is vehicle insurance where every year customer needs to pay a premium of certain amount to insurance provider company so that in case of unfortunate accident by the vehicle, the insurance provider company will provide a compensation to the customer. An Insurance company that has provided Health Insurance to its customers, they need to up-sell the policyholders (customers) from past year will a Vehicle Insurance provided by the company.

The Problem:
Building a model to predict whether a customer would be interested in Vehicle Insurance is extremely helpful for the company because it can then accordingly plan its communication strategy to reach out to those customers and optimize its business model and revenue.

Solution:
The data set is unbalanced. So, we will be using under sampling techniques to balance the data. The data set has a few columns with text data values, so we will be converting those text values into numerical values. We will implement classification algorithms like logistic regression, decision trees, support vector machines, Random forests and Naïve Bayes. To analyze the accuracy of the model we will calculate AUROC score to find the best model.

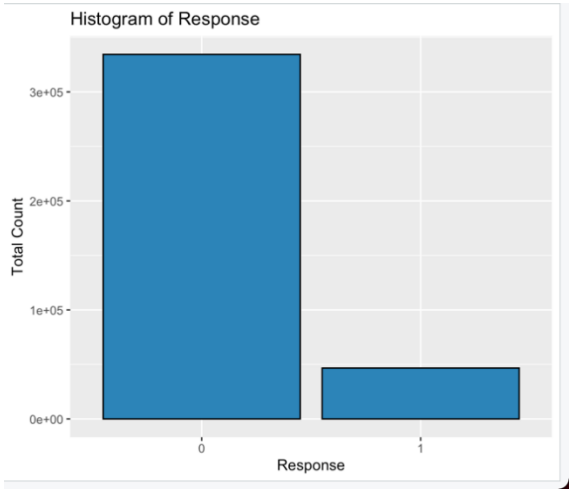
II. Data Exploration and Visualization

Data Description:
The training data has over 380k policy holders' records in 12 columns. The test data set has over 127k over which we will apply the prediction model prepared using the training data set. Data has been gathered from a data repository called "Kaggle".

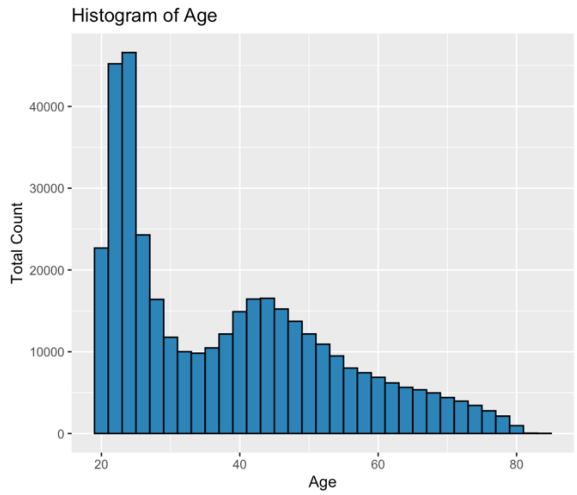
Data Visualization:
The training data has over 380k policy holders' records in 12 columns. The test data set has over 127k over which we will apply the prediction model prepared using the training data set. Data has been gathered from a data repository called "Kaggle".

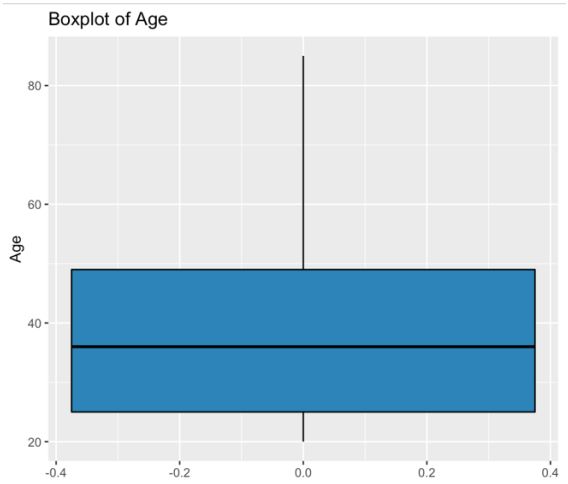
```
'data.frame':   381109 obs. of  12 variables:
 $ id                  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender              : Factor w/ 2 levels "Female","Male": 2 2 2 2 1 1 2 1 1 1 ...
 $ Age                 : int  44 76 47 21 29 24 23 56 24 32 ...
 $ Driving_License     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Region_Code         : num  28 3 28 11 41 33 11 28 3 6 ...
 $ Previously_Insured  : int  0 0 0 1 1 0 0 0 1 1 ...
 $ Vehicle_Age         : Factor w/ 3 levels "< 1 Year","> 2 Years",..: 2 3 2 1 1 1 1 3
 1 1 ...
 $ Vehicle_Damage      : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 2 2 2 1 1 ...
 $ Annual_Premium      : num  40454 33536 38294 28619 27496 ...
 $ Policy_Sales_Channel: num  26 26 26 152 152 160 152 26 152 152 ...
 $ Vintage             : int  217 183 27 203 39 176 249 72 28 80 ...
 $ Response            : int  1 0 1 0 0 0 0 1 0 0 ...
```

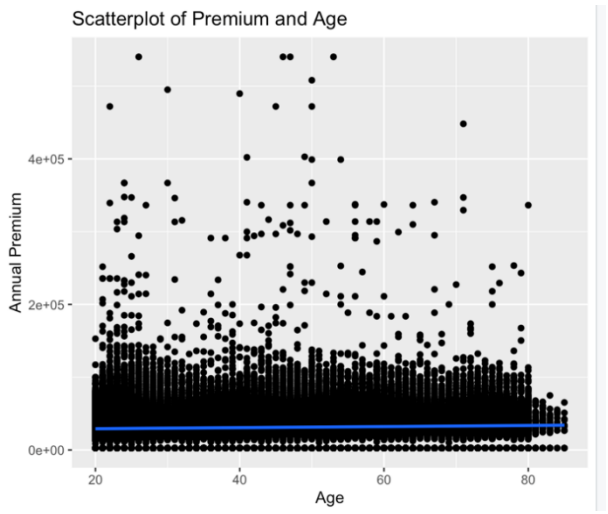
Distribution of Responses



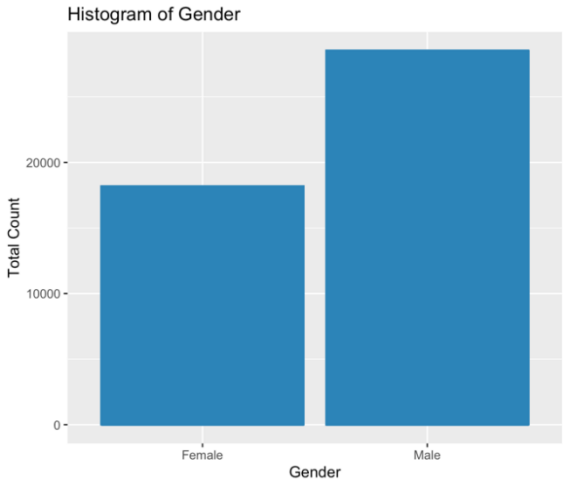
Distribution of Age

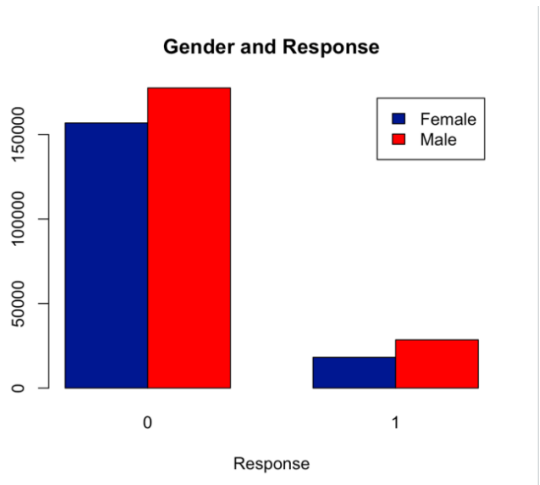


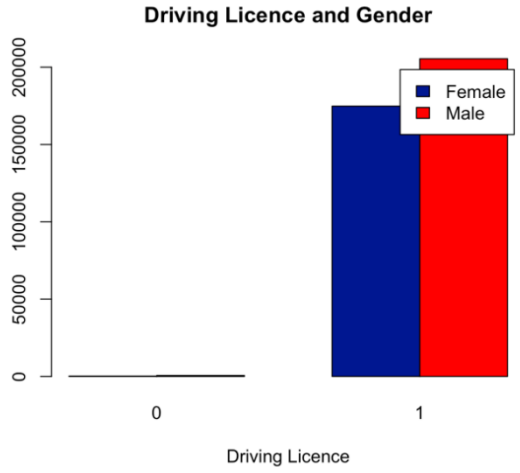




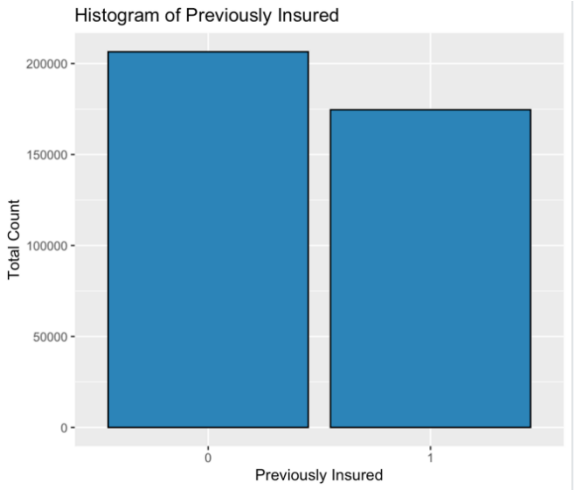
Distribution of Gender



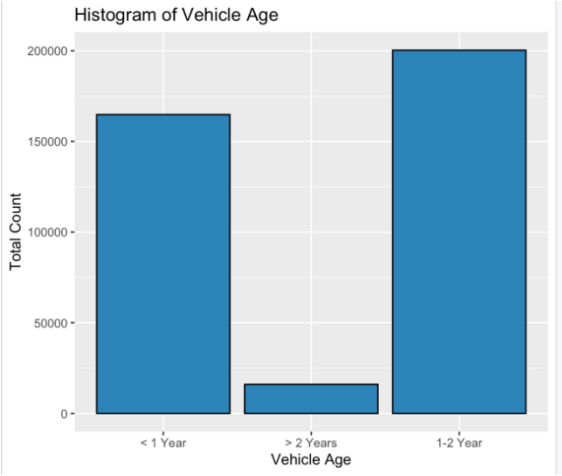


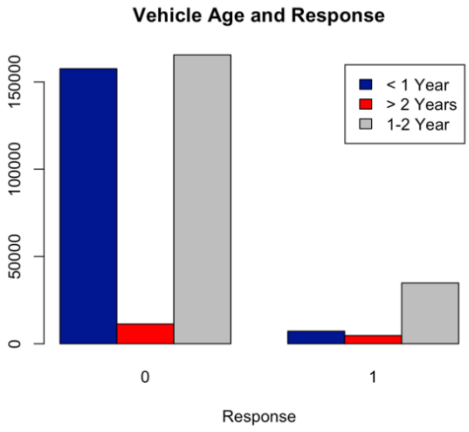


Distribution of Previous Insured

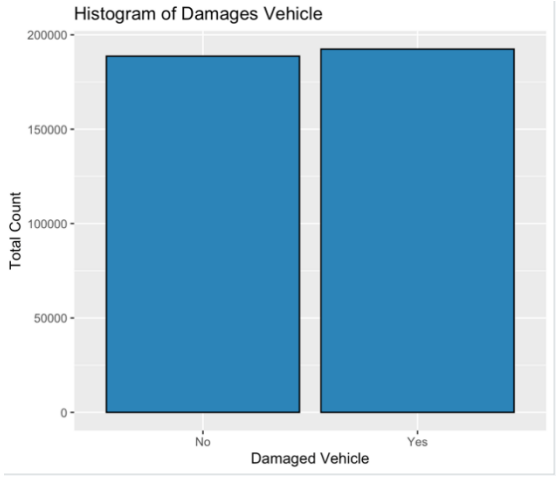


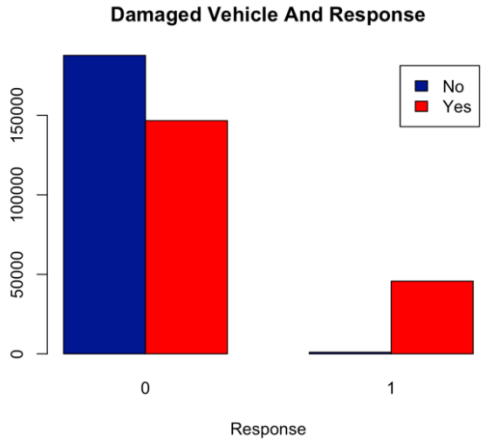
Distribution of Vehicle Age



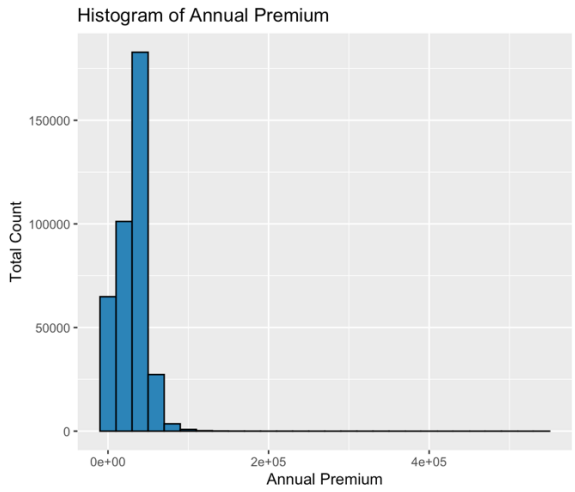


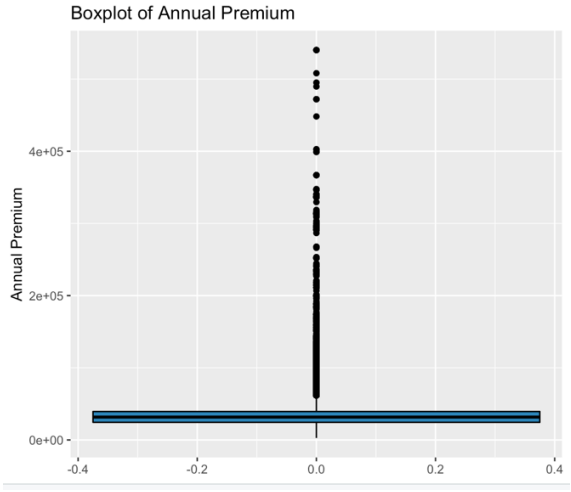
Distribution of Vehicle Damage



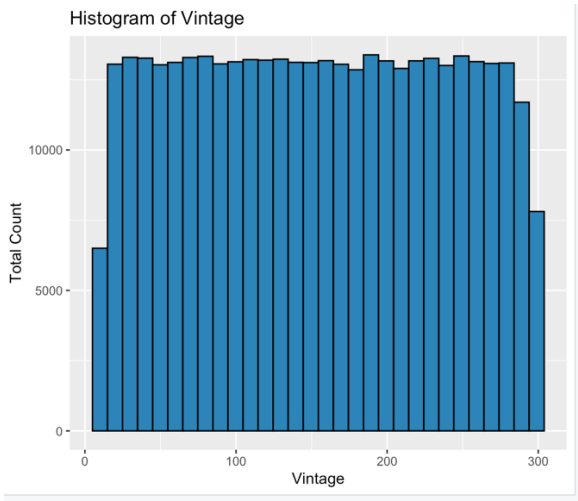


Distribution of Annual Premium





Distribution of Vintage



III. Data Preparation and Preprocessing

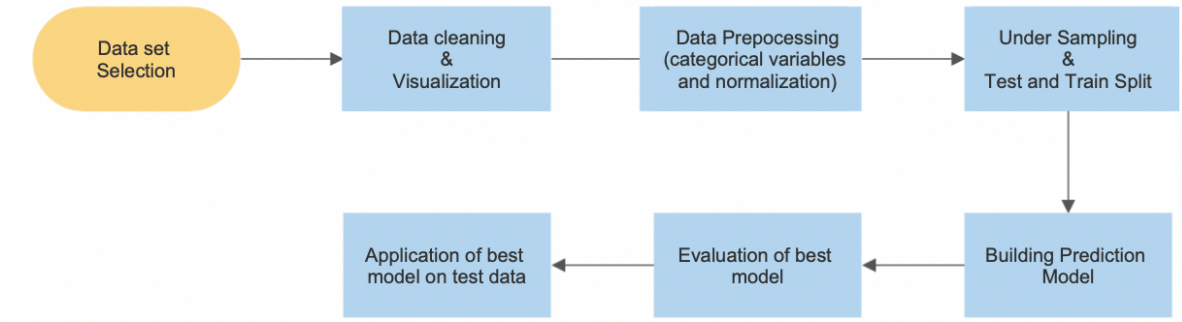
We have 12 variables with 381k observations for training data. The number of observations >>> variables so we will not be performing PCA. There is an imbalance in the data. The binary response has 88% of data as '0' and remaining 12% of data as '1'. So, we will be performing under sampling where all the responses of '1' will be taken and equal number of '0' will be taken for training models. The 3 categorical variables will be converted to dummy columns with factor values as '0' and '1'. The 'id' column will be dropped from the data set. The remaining data will be normalized before applying various prediction models.

IV. Data Mining Techniques and Implementation

We will be using the following algorithms:

- Logistic Regression
- Naive Bayes Classifier
- Classification Tree
- Random Forests
- Support Vector Machine

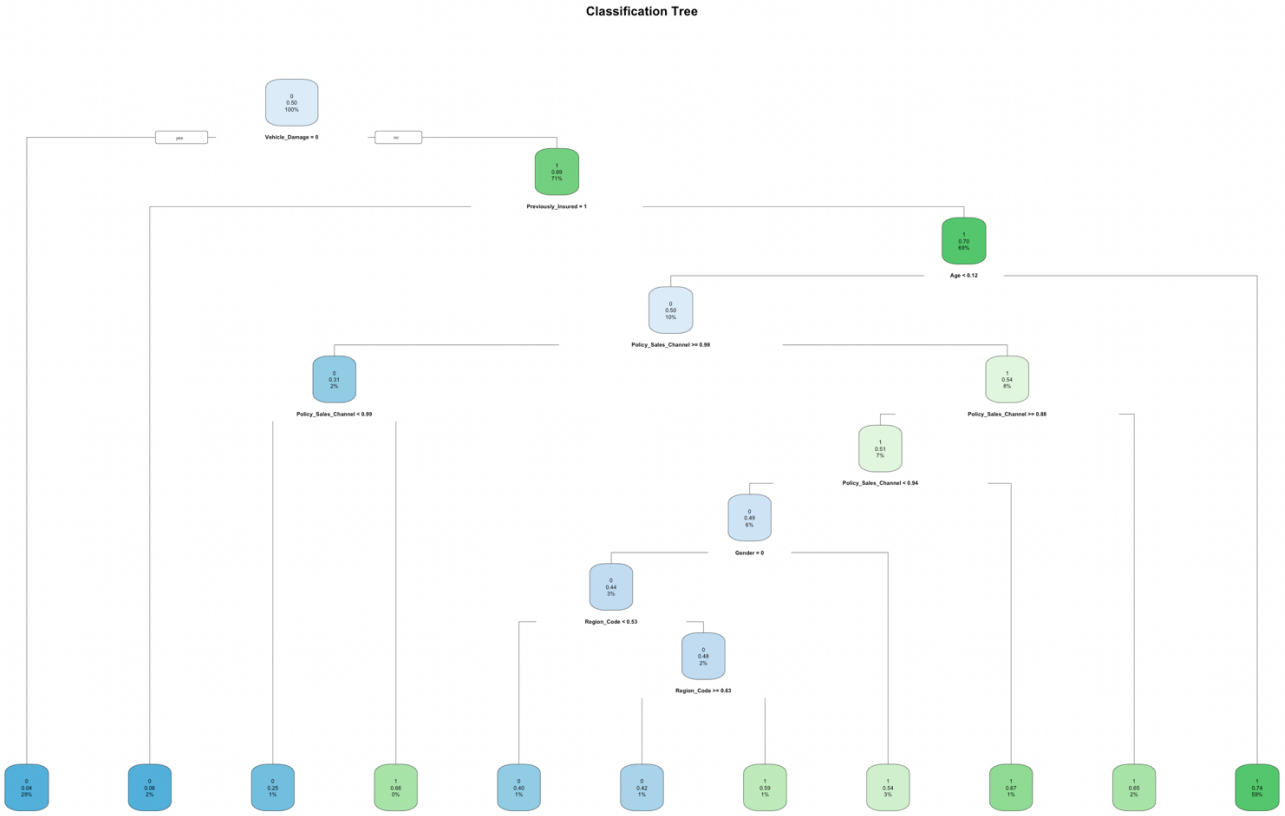
Flowchart:



V. Performance Evaluation

We applied the above-mentioned algorithms on the dataset and evaluated them using their Area Under Curve Values and Accuracy on the validation set.

The following classification architectures performed on the data:



	Model	AUC	Accuracy
1	Logistic Regression	83.85	78.58
2	Naive Bayes	78.55	78.64
3	Decision Tree	79.68	79.74
4	Random Forest	79.73	79.79
5	Support Vector Machine	79.45	79.51

VI. Discussion and Recommendation

Although we found that Logistic Regression model performs the best on this data, a strong case can be made that a Random Forest or a Decision tree can perform fairly well. However, feature selection requires domain knowledge on data mining and machine learning. In conclusion, the more the data, the more the accurate results we will get.

VII. Summary

In conclusion, in this case study we test many different machine learning models to perform this binary classification task, we trained the model on a normalized dataset. We conclude that a logistic regression models performs the best in this study, with AUC = 83.85% and Accuracy = 78.58%. The other models also performed well in this study with notable mention, Random Forest with AUC = 79.73% and 79.79% and also Decision Tree with AUC = 79.39% and Accuracy = 79.38%.

Appendix: R Code for use case study

```
library("tidyverse")
library("psych")
library("caret")
library("FNN")
library("ISLR")
library("tree")
library("randomForest")
library("neuralnet")
library("ROCR")
library("e1071")
library("gains")
library("ggplot2")
library("reshape2")
library("rpart")
library("rpart.plot")
library("corrplot")
library("mlr")
library("ROSE")
library("class")
library("InformationValue")
library("neuralnet")
library("e1071")
library("rpart")

train <- read.csv("train.csv", stringsAsFactors = TRUE)
test <- read.csv("test.csv", stringsAsFactors = TRUE)

head(train)
head(test)

dim(train)
dim(test)

sum(is.null(train))

numerical_columns <- c("Age", "Region_Code","Annual_Premium","Vintage")
```

```
categorical_columns <-
c("Gender","Driving_License","Previously_Insured","Vehicle_Age","Vehicle_Damage",
"Response")


summary(train[numerical_columns])
describe(train[numerical_columns])

##Response Histogram
ggplot(train, aes(x=as.factor(Response))) + geom_bar(color = "black", fill = "steelblue")
+ ggtitle("Histogram of Response") + labs(x="Response", y="Total Count")
table(train$Response)

##Age Histogram
ggplot(train, aes(Age)) +geom_histogram(binwidth = 2, color = "black", fill =
"steelblue") + ggtitle("Histogram of Age") + labs(x="Age", y="Total Count")

##Age Boxplot
ggplot(train, aes(y=Age)) + geom_boxplot(color = "black", fill = "steelblue") +
ggtitle("Boxplot of Age")

##Age vs Premium
ggplot(train, aes(x=Age, y=Annual_Premium)) + geom_point() +
geom_smooth(method=lm, se=FALSE) + ggtitle("Scatterplot of Premium and Age") +
labs(x="Age", y="Annual Premium")

##Gender Bar
ggplot(train, aes(x = Gender, y = Response)) + geom_bar(stat = "identity", color =
"steelblue") + ggtitle("Histogram of Gender") + labs(x="Gender", y="Total Count")

##Gender and Response Bar
barplot(table(train$Gender,train$Response), main="Gender and Response",
xlab="Response", beside=TRUE, legend =
colnames(table(train$Response,train$Gender)), col=c("darkblue","red"))

##Driving License and Gender
table(train$Gender,train$Driving_License)
barplot(table(train$Gender,train$Driving_License), main="Driving Licence and Gender",
xlab="Driving Licence", beside=TRUE, legend =
rownames(table(train$Gender,train$Driving_License)), col=c("darkblue","red"))

##Already Insured Bar
ggplot(train, aes(x=as.factor(Previously_Insured))) + geom_bar(color = "black", fill =
"steelblue") + ggtitle("Histogram of Previously Insured") + labs(x="Previously Insured",
y="Total Count")
```

```
##Vehicle Age Bar
ggplot(train, aes(x=Vehicle_Age)) + geom_bar(color = "black", fill = "steelblue") +
ggtitle("Histogram of Vehicle Age") + labs(x="Vehicle Age", y="Total Count")

##Response And Vehicle Age
table(train$Vehicle_Age,train$Response)
barplot(table(train$Vehicle_Age,train$Response), main="Vehicle Age and Response",
xlab="Response", beside=TRUE, legend =
rownames(table(train$Vehicle_Age,train$Response)), col=c("darkblue","red","Grey"))

##Damaged Vehicle Bar
ggplot(train, aes(x=Vehicle_Damage)) + geom_bar(color = "black", fill = "steelblue") +
ggtitle("Histogram of Damages Vehicle") + labs(x="Damaged Vehicle", y="Total
Count")

##Damaged Vehicle And Response
table(train$Vehicle_Damage,train$Response)
barplot(table(train$Vehicle_Damage,train$Response), main="Damaged Vehicle And
Response", xlab="Response", beside=TRUE, legend =
rownames(table(train$Vehicle_Damage,train$Response)), col=c("darkblue","red"))

##Premium Dist
ggplot(train, aes(Annual_Premium)) +geom_histogram(binwidth = 20000, color =
"black", fill = "steelblue")+ ggtitle("Histogram of Annual Premium") + labs(x="Annual
Premium", y="Total Count")

##Premium Boxplot
ggplot(train, aes(y=Annual_Premium)) + geom_boxplot(color = "black", fill =
"steelblue") + ggtitle("Boxplot of Annual Premium") + labs(y="Annual Premium")

##Vintage Dist
ggplot(train, aes(Vintage)) +geom_histogram(color = "black", fill = "steelblue") +
ggtitle("Histogram of Vintage") + labs(x="Vintage", y="Total Count")

#Converting Categorical variables in dummy variables
train["Gender"] <- as.numeric(ifelse(train$Gender == "Male", 1, 0))
train["Vehicle_Damage"] <- as.numeric(ifelse(train$Vehicle_Damage == "Yes", 1, 0))
train_dummies <- createDummyFeatures(train, cols = c("Vehicle_Age"))

colnames(train_dummies)[12] <- c("Vehicle_Age_lt_1_Year")
colnames(train_dummies)[13] <- c("Vehicle_Age_gt_2_Year")
colnames(train_dummies)[14] <- c("Vehicle_Age_1_2_Year")

num_feat = c('Age','Vintage')
```

```
cat_feat = c('Gender', 'Driving_License', 'Previously_Insured',
'Vehicle_Age_lt_1_Year','Vehicle_Age_1_2_Year','Vehicle_Age_gt_2_Years','Vehicle_
Damage_Yes','Region_Code','Policy_Sales_Channel')

#Normalize the dataframe
normalize <- function(x) {
  return ((x - min(x)) / (max(x) - min(x))) }

df.norm <-as.data.frame(cbind(as.data.frame(lapply(train_dummies[c(2:10,12:14)],
normalize)),train_dummies$Response)) %>% rename(Response =
"train_dummies$Response")

#Sampling

train_over <- ovun.sample(Response ~ ., data = df.norm, method = "under")$data
table(train_over$Response)
train.index <- sample(rownames(train_over), floor(nrow(train_over)*0.5))
train_data <- train_over[train.index,]
validation_data <- train_over[!rownames(train_over) %in% train.index,]

levels(train_data$Response) <-make.names(levels(factor(train_data$Response)))
levels(validation_data$Response) <-
make.names(levels(factor(validation_data$Response)))

train.labels <- train_data[,13]
test.labels <- validation_data[,13]

performance_list <-data.frame("Model" = character(),"AUC" = numeric(),"Accuracy" =
numeric())

#test set modification
test["Gender"] <- as.numeric(ifelse(test$Gender == "Male", 1, 0))
test["Vehicle_Damage"] <- as.numeric(ifelse(test$Vehicle_Damage == "Yes", 1, 0))
test_dummies <- createDummyFeatures(test, cols = c("Vehicle_Age"))

colnames(test_dummies)[11] <- c("Vehicle_Age_lt_1_Year")
colnames(test_dummies)[12] <- c("Vehicle_Age_gt_2_Year")
colnames(test_dummies)[13] <- c("Vehicle_Age_1_2_Year")

test_dummies <- test_dummies[,-1]
test_dummies <- as.data.frame(lapply(test_dummies, normalize))

#log reg
glm.fit <- glm(Response~., data = train_data, family = binomial)
glm.probs <-predict(glm.fit, newdata = validation_data, type = "response")
optCutOff <- optimalCutoff(validation_data$Response, glm.probs)
```

```
a <- confusionMatrix(validation_data$Response, glm.probs, threshold = optCutOff)
temp <-data.frame("Model" = "Logistic Regression","AUC" =
round(auc(validation_data$Response, glm.probs)*100,2),"Accuracy" =
round((a[1,1]+a[2,2])/nrow(validation_data)*100,2))
performance_list <- rbind(performance_list,temp)

#Naive Bayes
NaiveB <- naiveBayes(Response~., data = train_data)
pred.nb <- predict(NaiveB, validation_data)
a <- confusionMatrix(as.numeric(pred.nb), validation_data$Response, threshold =
optCutOff)
temp <-data.frame("Model" = "Naive Bayes","AUC" =
round(auc(validation_data$Response, as.numeric(pred.nb))*100,2),"Accuracy" =
round((a[1,1]+a[2,2])/nrow(validation_data)*100,2))
performance_list <- rbind(performance_list,temp)

#Decision Tree
Dtree <-rpart(Response ~ ., data = train_data, method='class', cp=0.001)
rpart.plot(Dtree, main = "Classification Tree")
pred.dt <-predict(Dtree, validation_data, type = 'class')
a <- table(validation_data$Response, pred.dt)
temp <-data.frame("Model" = "Decision Tree","AUC" =
round(auc(validation_data$Response, as.numeric(pred.dt))*100,2),"Accuracy" =
round((a[1,1]+a[2,2])/nrow(validation_data)*100,2))
performance_list <- rbind(performance_list,temp)

#Random Forest
rf <- randomForest(as.factor(Response) ~ ., data = train_data, ntree=500)
rf.pred <- predict(rf, validation_data)
a <- confusionMatrix(rf.pred, validation_data$Response)
temp <-data.frame("Model" = "Random Forest","AUC" =
round(auc(validation_data$Response, as.numeric(rf.pred))*100,2),"Accuracy" =
round((a[1,1]+a[2,2])/nrow(validation_data)*100,2))
performance_list <- rbind(performance_list,temp)

#SVM
svm <-svm(Response ~ ., train_data, type = 'C-classification', kernel = 'radial')
y_pred_svr = predict(svm, newdata = validation_data, type="prob")
a <- confusionMatrix(y_pred_svr, validation_data$Response)
temp <-data.frame("Model" = "Support Vector Machine","AUC" =
round(auc(validation_data$Response, as.numeric(y_pred_svr))*100,2),"Accuracy" =
round((a[1,1]+a[2,2])/nrow(validation_data)*100,2))
performance_list <- rbind(performance_list,temp)
```