# Prediction of Survival in Titanic Implementing Machine Learning Algorithms

*Aishwarya Ganesh Murthy*
*Student ID: 92436*
*International Technological University*
*ganeshaish1068@students.itu.edu*

*Dipali Suryawanshi*
*Student ID: 91120*
*International Technological University*
*suryawanshdipal643@students.itu.edu*

Submitted for the Machine Learning Project
Subject: Machine Learning (CS 633 1)
Patricia Hoffman, PhD

## Declaration

This report has been prepared on the basis of our own work. Where other published and unpublished source materials have been used, these have been acknowledged.

**Team Members**: Aishwarya Ganesh Murthy, Dipali Suryawanshi

**Date of Submission**: December 10, 2017

## Acknowledgements

We would like to express our sincere thanks and gratitude to our supervisor Prof. Patricia Hoffman, who has taught and guided us through this project activity. Her subject expertise, her intuition on the work areas and her patience as my guide played a major part in making this project what it is today.

## Abstract

The sinking of the Titanic is one of the landmark disasters in history. Though it occurred over 100 years ago, it still fascinates researchers to understand why some passengers survived while others suffer death. In this project, we applied different machine learning algorithms to titanic data set to predict the survival of the passengers based on different attributes or information of those passengers like sex, age, class, fare,etc. The available data set is split into training set and test set. The training set consist of 891 records, we used this dataset to build our models using algorithms like Decision tree,Naive Bayes, Support Vector Machines, K Nearest Neighbours and Random forest to predict the survival based on some of the attributes or features present in the data. In addition to that we applied feature engineering to create new features and encoded in the model to see if we can achieve good performance. The main goal of this project is to predict the survival of the passengers based on the attributes present using different machine learning algorithms, and compare different algorithms to find the one with better accuracy. The test set consists of 418 records, these are the set of passengers whose survival is unknown to the world. This paper also predicts the survival rate of these unknown passengers in the test set by applying the models that we used in the training set.

# Table of Contents

## 1. Problem Statement

On April 15, 1912, the ship RMS Titanic struck an iceberg on its maiden voyage and sank, resulting in the deaths of most of its passengers and crew. Despite of having very less lifeboats compared to the population of passengers in the ship, there was a luck of survival for some groups of people. Our research question discussed in this paper predicts which group of people had the luck to survive, this prediction is done based on the different attributes present in the dataset like age, gender, ticket class, port of embarkation,etc by applying various machine learning algorithms.

The interesting question hidden in the data is that there seems to be a pattern among the large section of people survived through this major shipwreck accident based on different attributes like age, gender and ticket class,etc. For example, women, children or passengers in the higher ticket class had more chances of surviving than the other passengers who suffered death because of the accident. And, there is also a set of people whose survival is unknown to the world, the information of whether they survived through the accident or suffered death is missing. So, in this paper, we try to apply different machine learning algorithms to find the pattern of survival based on the different attributes and thus use this pattern to predict the unknown data. We also compare the accuracy produced by the machine learning algorithms to predict the pattern, and finally improve the efficiency in the prediction of survival in Titanic. We used Kaggle competition "Titanic: Machine Learning from Disaster" [1] to retrieve necessary data.

Recently, many studies have been conducted to compare and contrast the different machine learning techniques for this problem. Tryambak Chatterjee[2] used three different models with only gender attribute for this classification problem. He applied linear model, Multiple regression and logistic regression across the different test cases and all algorithms yields almost similar results. He had just used training data from kaggle website to create model and predict the results by dividing training data into two parts using three different ratios. He had not used any test data from site. In case of linear model accuracy is 79.820%. The maximum accuracy obtained from Multiple Linear Regression is 78.426% and for Logistic Regression, it is 80.756%. Thus in conclusion Logistic regression model performs somewhat better than other two.

## 2. Data Exploration

### 2.1 Data

The data consists of the following fields:

- PassengerId: A unique ID (‘nominal’ data) is attached to each passenger in the Titanic.
- Survived: This is the output class which we are going to predict. It contains values ‘0’ and ‘1’, ‘0’ means that the passenger did not survive and ‘1’ means that the passenger survived.
- Pclass: This is the ticket class(‘ordinal’ data). It contains values ‘1’, ‘2’ and ‘3’, ‘1’ refers the first class ticket, ‘2’ refers the second class ticket, and ‘3’ refers to the third class ticket.
- Name: This is the passenger’s name.
- Sex: This is the passenger’s sex, it contains values ‘Male’ or ‘Female’.
- Age:This is the passenger’s age.
- SibSp: This explains the number of siblings or spouse (dependants) travelling with the passengers

- Parch: This explains the number of parents or children (dependants) travelling with the passengers
- Ticket: Ticket number
- Fare: Cost of the ticket purchased for the journey
- Cabin:This is the cabin number of the passenger.
- Embarked: Port where the passenger was boarded or embarked.

## 2.2 Resolve Data Quality Issues

Titanic data set consists of nominal, ordinal and interval scale data. We found some missing values and incomplete fields in sample data. Some fields were containing hidden information so our first approach was to classify data.

Each attribute is taken separately to check for bad or missing data. The checking is done as follows:

1. The test and the training set is combined together as 'titanic.full' to check on each attribute
2. Sum of the values which are marked 'NA' is calculated
3. If the sum is '0', then there are no null values and if it is not null, then the number of null values is printed in the sum.

*The command used to check the null values in each attribute gives the value:*

```
> passengerId_null = sum(is.na(titanic.full$PassengerId))
> sprintf("No of null values for passennger Id: %d",passengerId_null)
[1] "No of null values for passennger Id: 0"
```

*Similarly, for other attributes:*

```
> survived_null = sum(is.na(titanic.full$Survived))
> sprintf("No of null values for survied : %d",survived_null)
[1] "No of null values for survied : 418"

> age_null = sum(is.na(titanic.full$Age))
> sprintf("No of null values for age : %d",age_null)
[1] "No of null values for age : 263"

> fare_null = sum(is.na(titanic.full$Fare))
> sprintf("No of null values for fare : %d",fare_null)
[1] "No of null values for fare : 1"
```

- Null values are present only in survived, age and fare. Null values of survived is the values that we added in the test set since 'survived' is not present in it.
- Age has null values, so those null values are replaced by the median age of the passengers. This is done using the following command:

```
> age.median<-median(titanic.full$Age, na.rm = TRUE)
> titanic.full[is.na(titanic.full$Age),"Age"] <- age.median
```

- Fare has one missing value, we have to find where that missing value is present in fare. This is replaced by the median of fare of other people of the same class and embarkment port

```
> titanic.full[(which(is.na(titanic.full$Fare))) , 1]
[1] 1044
>titanic.full$Fare[1044] <- median(titanic.full[titanic.full$Pclass == '3' & titanic.full$Embarked == 'S', ]$Fare,
na.rm = TRUE)
```

- Cabin has missing values like "", this is replaced by NA. Then, the null values are calculated for it.

```
> titanic.full$Cabin[titanic.full$Cabin == ""] <- NA
> cabin_null = sum(is.na(titanic.full$Cabin))
> sprintf("No of null values for cabin : %d",cabin_null)
[1] "No of null values for cabin : 1014
```

A lot of data is missing in the cabin attribute, so we cannot derive it using the little amount of cabin information that we have. So, we ignore the cabin information in our prediction.

- Embarked has missing values like "", this is replaced by NA. Then, the number of null values are calculated.

```
> titanic.full$Embarked[titanic.full$Embarked == ""] <- NA
> embarked_null = sum(is.na(titanic.full$Embarked))
> sprintf("No of null values for Embarked : %d",embarked_null)
[1] "No of null values for Embarked : 2
```

Two values are missing in Embarked, we have to find where that missing value is present in embarked.

```
> titanic.full[(which(is.na(titanic.full$Embarked))), 1]
[1] 62 830
> titanic.full$Embarked[c(62,830)] <- median(titanic.full$Embarked)
```

## 2.3Feature Creation and Selection

Feature engineering is fundamental to the application of machine learning that makes machine learing algorithms work better.Since titanic data has some fields with hidden information, we approached for feature engineering procees. We created new attributes to see if we can improve performance of model.

**Extracting title from name:**

Names in data set are like "Braund, Mr. Owen Harris". Name is unique attribute for each passenger so it won't help much in prediction but we can break down name into additional meaningful varibles that can be used in the creation of additional attributes.

We extracted a passenger's title from name. We found below list of titles in data set.

'Capt', 'Col', 'Don','Dona', 'Dr', 'Jonkheer', 'Lady', 'Major', 'Rev', 'Sir', 'the Countess','Mlle','Ms','Mme'

We noticed that title indicates passenger's sex , profession ('Dr'). We created titles from name and below is table showing title count against gender

| | Master | Miss | Mr | Mrs | Unusual Title |
|---|---|---|---|---|---|
| Female | 0 | 264 | 0 | 198 | 4 |
| Male | 61 | 0 | 757 | 0 | 25 |

**Creating new attribute child:**

Below is code to create new attribute. It uses attribute age to separete child from adult.

```
>titanic.full$child[titanic.full$Age<18]<-"Child"
>titanic.full$child[titanic.full$Age>=18]<-"Adult"
```

**Creating new attribute Mother:**

Below is code to create new attribute Mother. It uses already present attributes like sex,age, parch and title attribute which we created above.

```
>titanic.full$isMother <- "Not Mother"
>titanic.full$isMother[titanic.full$Sex=="female" & titanic.full$Parch >0 &titanic.full$Age >18 & titanic.full$title
!= "Miss"]<-"Mother"
```

**Creating new attribute Family Size:**

We used formula Family = Sibsp +parch +1 (for him/herself) to calculate family size. Below is code to extract family size

```
>titanic.full$familysize  <- titanic.full$SibSp  + titanic.full$Parch  +1
>family_null  = sum(is.na(titanic.full$familysize))
```

We further divided family size into categories like family of single, small and large family.Below are conditions used to categories family

```
>titanic.full$fsizeD[titanic.full$familysize==1]<-'single'
>titanic.full$fsizeD[titanic.full$familysize<5  & titanic.full$familysize>1]<-'small'
>titanic.full$fsizeD[titanic.full$familysize>4]<-'large'
```

## 2.4 Statistics of Data Subsets

Before finding the pattern to predict the survival, we have to find the correlation between each attribute with the survival. When a feature correlates well with the output, it means that some section of that feature influences the output more than any other feature present in the dataset. This will help us filter the necessary attributes that are required to build the model. We do this step using the following command to find the correlation of each attribute with the Survival:

```
> print("Correlation between Pclass and Survived")
[1] "Correlation between Pclass and Survived"
> aggregate(titanic.train["Survived"], by = list(Pclass=titanic.train$Pclass),FUN=mean,na.rm=TRUE)
```

| | Pclass | Survived |
|---|---|---|
| 1 | 1 | 0.6296296 |
| 2 | 2 | 0.4728261 |
| 3 | 3 | 0.2423625 |

| | Sex | Survived |
|---|---|---|
| 1 | female | 0.742038 |
| 2 | male | 0.1889081 |

| | Sibsp | Survived |
|---|---|---|
| 1 | 0 | 0.3453947 |
| 2 | 1 | 0.5358852 |
| 3 | 2 | 0.4642857 |
| 4 | 3 | 0.2500000 |
| 5 | 4 | 0.1666667 |
| 6 | 5 | 0.0000000 |
| 7 | 8 | 0.0000000 |

| | Parch | Survived |
|---|---|---|
| 1 | 0 | 0.3436578 |
| 2 | 1 | 0.5508475 |
| 3 | 2 | 0.5000000 |
| 4 | 3 | 0.6000000 |
| 5 | 4 | 0.0000000 |
| 6 | 5 | 0.2000000 |
| 7 | 6 | 0.0000000 |

| | Child | Survived |
|---|---|---|
| 1 | Adult | 0.3611825 |
| 2 | Child | 0.5398230 |

| | isMother | Survived |
|---|---|---|
| 1 | Mother | 0.7090909 |
| 2 | NotMother | 0.3624402 |

| | fsizeD | Survived |
|---|---|---|
| 1 | large | 0.1612903 |
| 2 | Single | 0.3035382 |
| 3 | Small | 0.5787671 |

| | Embarked | Survived |
|---|---|---|
| 1 | 1 | 0.5535714 |
| 2 | 2 | 0.3896104 |
| 3 | 3 | 0.3369565 |

## 3. Model Building & Deployment

Model is built to predict the survival of the passengers of Titanic. The training data is used to build the model to predict the survival of test data set. The steps followed to do the above is the following:

1. The training set has the actual survival of each passenger
2. Different algorithms are used to predict the survival based on other attributes in the training data
3. The predicted survival is compared with the actual survival to check the accuracy of the prediction algorithm
4. The test set is also predicted using the same algorithm, thus we will be able to predict the survival of the unknown set of passengers

After predicting the survival rate of the passengers using different algorithms, we compare the accuracy of prediction based on the error obtained by comparing the actual "Survival" data with the predicted "Survival" data on the training dataset.
The different set of algorithms used to predict the survival of the passengers are the following:

## 3.1 Decision Tree

Decision tree is flow-chart like structure in which decision or a particular condition is laid out on attribute in every node of the tree. The decision is used to separate out the data into different sections or branches, which forms the tree, the path from root to leaf represent classification rules. Thus, here when we try to predict the passenger survived class(tree leaves), we use all the features or attributes(tree branches).

*We have implemented decision tree as follows:*

```
> library(rpart)
> fit <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked, data=titanic.train,method="class")
```

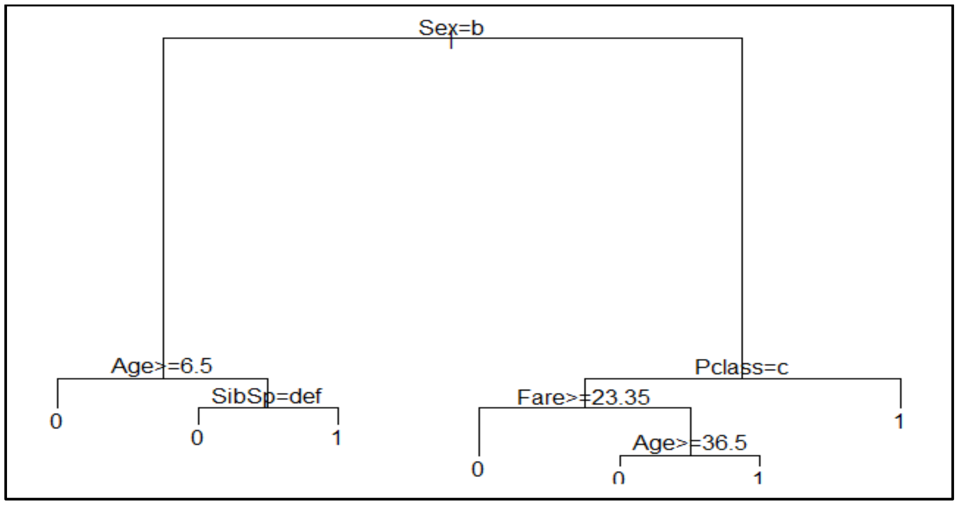

Fig 1: A tree showing applied conditions on each attribute at nodes

```
> yhat_train<- predict(fit, titanic.train, type = "class")
> submit <- data.frame(Survived = yhat_train)
> dt_error<- 1-sum(submit$Survived==y_train)/length(y_train)#train_error= 0.167604
```

*The True Positive and the False positive prediction of the Decision tree algorithm is:*

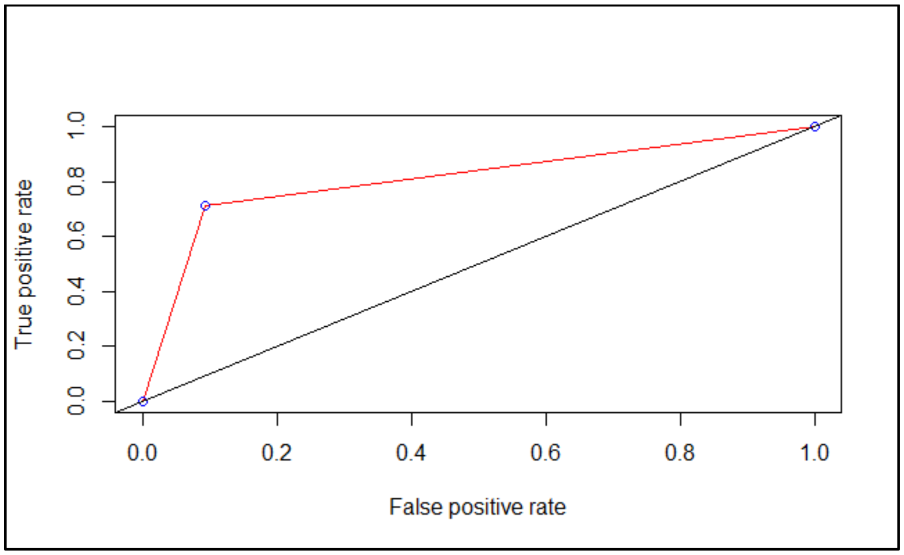

Fig 2: TPR and FPR of Decision Tree Algorithm

*And the test data was predicted:*

```
> yhat_test<-predict(fit, titanic.test, type = "class")
```

*Decision Tree algorithm is also used for prediction by adding new features other than the attributes present in the data.*

```
> #----------------------decision tree with new features added ----------
> fitnew <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked+fsizeD + isMother + child,data=titanic.train,method="class")
```

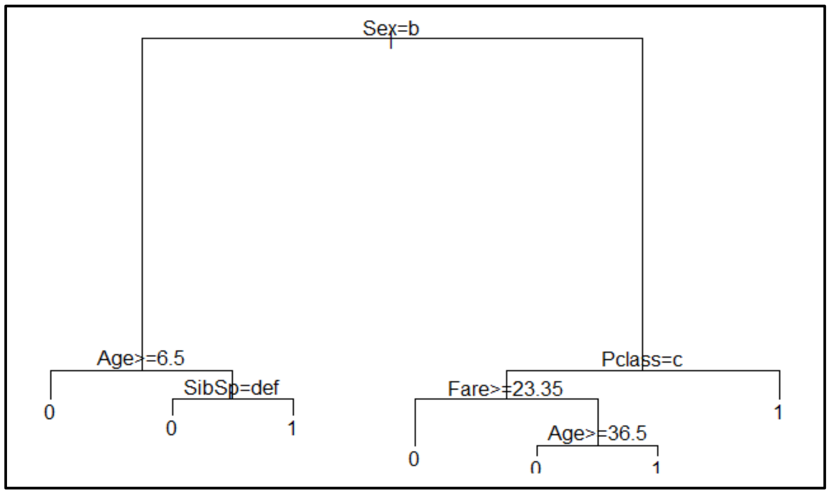

Fig 3:A tree showing applied conditions on each attribute at nodes

```
> yhat_trainnew<- predict(fitnew, titanic.train, type = "class")
> submitnew <- data.frame(Survived = yhat_trainnew)
> dt_errornew<- 1-sum(submit$Survived==y_trainnew)/length(y_trainnew) # train_error= 0.1672278
```

*The True Positive and the False positive prediction of the Decision tree algorithm with Feature Engineering is:*

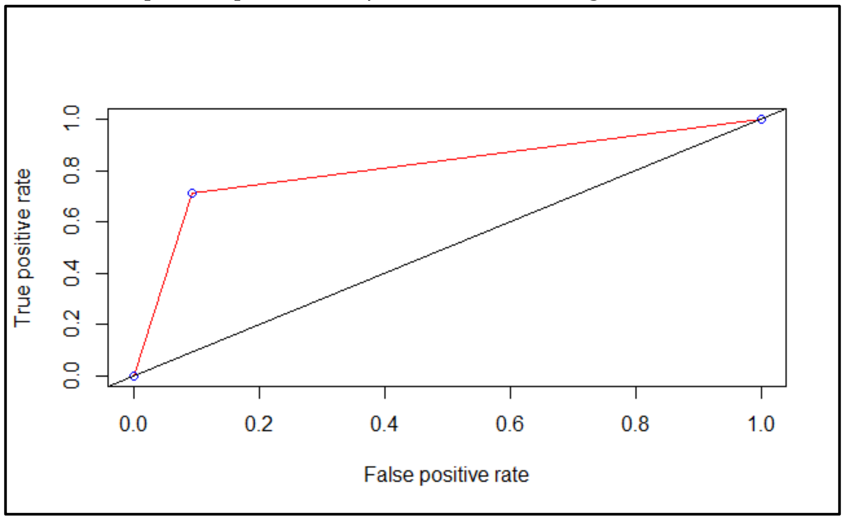

Fig 4: TPR and FPR of Decision Tree Algorithm after Feature Enjgineering

*The test set is predicted with these new features in addition to the attributes present:*

```
> yhat_testnew<-predict(fitnew, titanic.test, type = "class")
> df_resultnew<-data.frame(PassengerId = titanic.test$PassengerId, Survived = yhat_testnew)
```

## 3.2 Random Forest

Random Forest is an ensemble technique, where we create multiple decision trees on the training data, and outputs the mode of the classification labels. In our case, the labels can be classified in two classes: 0-

Not Survived, 1-Survived. In every iteration, a decision tree is built on the training data, and the output is the mode class (class that has maximum occurrence) of that data. This prevents the overfitting problem, that is high variance and low bias, faced by decision trees. Random forest decreases the variance by considering the mean of variance of each decision tree obtained at each iteration, finally obtaining low variance and low bias for a perfect prediction.

*We have implemented random forest as follows:*

```
> library(randomForest)
> titanic.model <-
randomForest(formula=Survived.formula,data=titanic.train,ntree=500,mtry=3,nodesize=0.01*nrow(titanic.train))
> rf_error<- 1-sum(submit$Survived==y_train)/length(y_train) #0.09561305
```

*The test data is also predicted using random forest:*

```
> Survived<-predict(titanic.model, newdata =titanic.test)
> rf_result<-data.frame(PassengerId = titanic.test$PassengerId, Survived = Survived)
```

*The True Positive and the False positive prediction of Random Forest Algorithm is:*

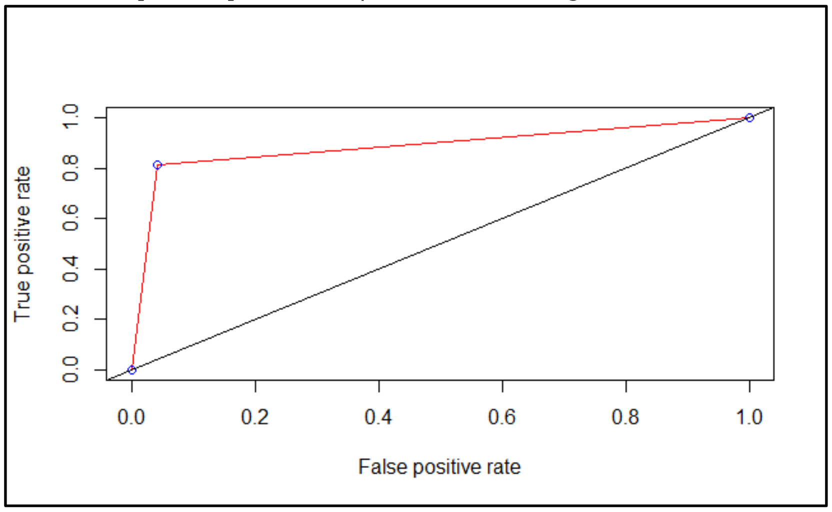

Fig 5: TPR and FPR of Random Forest Algorithm

*Random forest algorithm is also used for prediction by adding new features other than the attributes present in the data.*

```
> Survived.equtionnew<-"Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + fsizeD + isMother
+ child"
> Survived.formulanew <- as.formula(Survived.equtionnew)
> y_trainnew <- titanic.train$Survived
> titanic.modelnew <- randomForest(formula=Survived.formulanew,data=titanic.train,ntree
=500,mtry=3,nodesize=0.01*nrow(titanic.train))
> features.equtionnew<-"Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + fsizeD + isMother + child"
> yhatnew<-predict(titanic.model, newdata =titanic.train)
> submit <- data.frame(Survived = yhatnew)
> rf_errornew<- 1-sum(submit$Survived==y_trainnew)/length(y_trainnew)#0.094276
```

*The test data is predicted using the new features along with the other attributes.*

```
> Survived<-predict(titanic.modelnew, newdata =titanic.test)
> rf_resultnew<-data.frame(PassengerId = titanic.test$PassengerId, Survived = Survived)
```

*The True Positive and the False positive prediction of Random Forest algorithm after Feature Engineering is:*

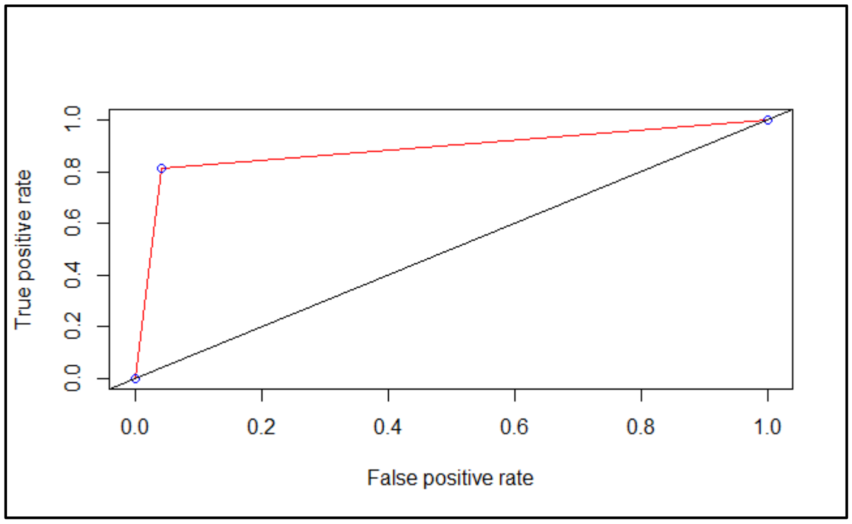

Fig 6: TPR and FPR of Random Forest Algorithm after Feature Engineering

## 3.3 Naive Bayes

Bayes' theorem defines the probability of an event, based on the conditions related to the event are known. The bayes' theorem is stated by the following formula:

Given P(A) != 0, then P(B|A) = P(A|B) P(B) / P(A)

In our case, consider event of each attribute base on a given output class: Not Survived- 0 and Survived - 1. If the events are independent for a given class (whether Survived or Not Survived), then the attribute conditions that can be obtained for the case survived and not survived are obtained.

P(A1,A2,A3,A4……An/Cj) = P(A1/Cj). P(A2/Cj). P(A3/Cj)…….P(An/Cj)

"Naive Bayes Classifier are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong independence assumptions between the features.Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle: all naive Bayes classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable."[3]

*We have implemented naive bayes in the following manner:*

```
> library(klaR)
> library(caret)
> nbModel= NaiveBayes(Survived~Pclass + Age+ Sex + SibSp + Fare+ Parch + Embarked, data=titanic.train,
usekernel = FALSE, fL = 0)
```

```
> yhat_train = predict(nbModel, titanic.train)
> submit <- data.frame(Survived = yhat_train$class)
> nb_error<- 1-sum(submit$Survived==y_train)/length(y_train)  #train_error = 0.2401796
```

*And predict the test set as follows:*

```
> yhat_test = predict(nbModel, titanic.test)
> nb_result<-data.frame(PassengerId = titanic.test$PassengerId, Survived = yhat_test)
```

*The True Positive and the False positive prediction of Naive Bayes' algorithm is:*

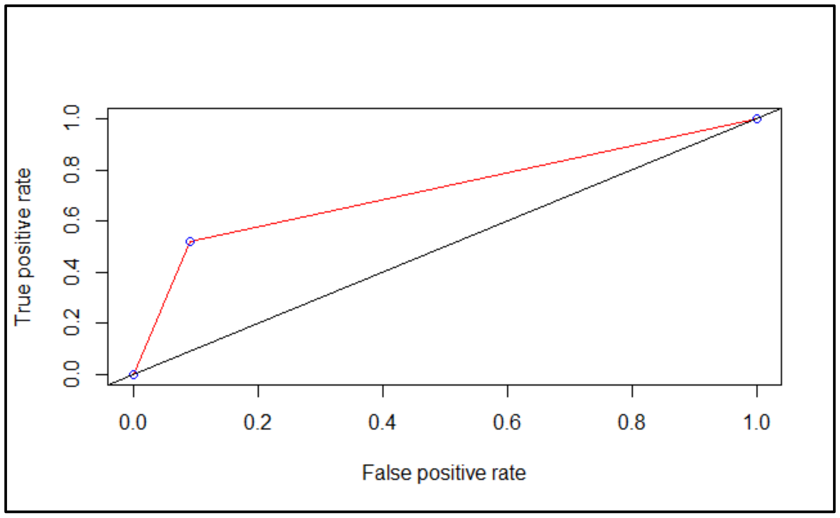

Fig 7:TPR and FPR of Naive Bayes' Algorithm

*Then, new features are added for prediction using naive bayes through feature engineering:*

```
> nbModelnew = NaiveBayes(Survived~Pclass + Age+ Sex + SibSp + Fare+ Parch + Embarked+fsizeD  + isMother
+ child
+ , data=titanic.train, usekernel = FALSE,  fL = 0)
> yhat_trainnew = predict(nbModelnew, titanic.train)
> submitnew <- data.frame(Survived  = yhat_trainnew$class)
> nb_errornew<- 1-sum(submit$Survived==y_trainnew)/length(y_trainnew)  #train_error = 0.2401796
```

*The True Positive and the False positive prediction of Naive Bayes' algorithm after Feature Engineering is:*

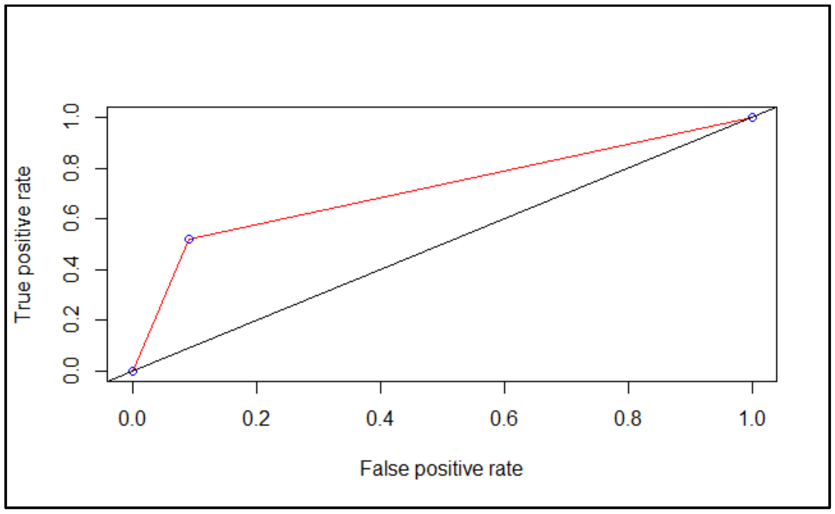

Fig 8:TPR and FPR of Naive Bayes Algorithm after Feature Engineering

*The test data is predicted with these new features along with the attributes present:*

```
> yhat_testnew = predict(nbModelnew , titanic.test)
> nb_resultnew<-data.frame(PassengerId = titanic.test$PassengerId, Survived = yhat_testnew)
```

## 3.4 Support Vector Machines

SVM is supervised learning model with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. The goal of a SVM is to find a hyperplane that separates the positive data points from the negative data points. In our case, given a set of training data, SVM algorithm builds a model that assigns test data into one of the two categories namely survived and not survived. As SVM classifier requires numerical data as input, we have applied categorical featuring to attributes.

*We implemented Support Vector machines as follows:*

```
> library(e1071)
> Svm <- svm(Survived ~ Pclass + Age+ Sex + SibSp + Fare+ Parch + Embarked, data = titanic.train)#cost: 1 , gamma: 0.04761905
> yhat_train<-predict(Svm,x)
> submit <- data.frame(Survived = yhat_train)
> svm_error<- 1-sum(submit$Survived==y)/length(y) #0.2107843
```

*The test data is predicted using SVM:*

```
> x_test <- subset(titanic.test, select = -Survived)
> yhat_test<-predict(Svm,x_test)
> svm_result<-data.frame(Survived = yhat_test)
 Survived
 0:37
 1:54
```

*The True Positive and the False positive prediction of SVM algorithm is:*

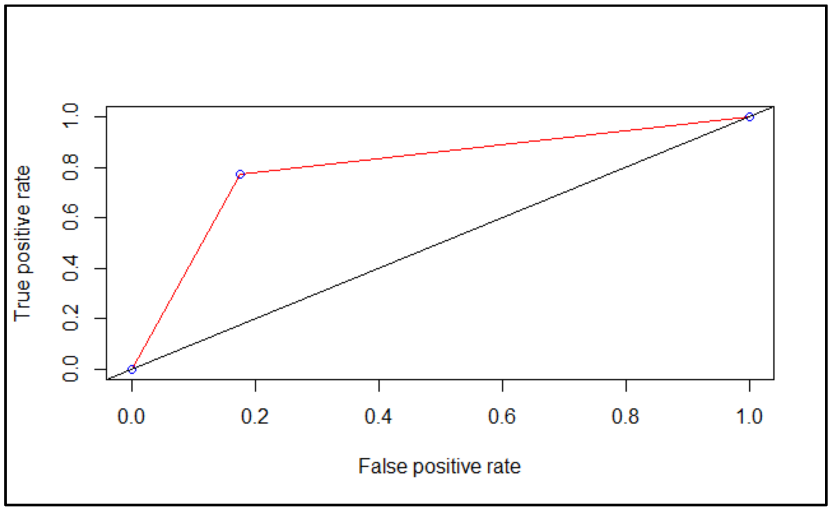

Fig 9: TPR and FPR of SVM Algorithm

*SVM is predicted using new features along with the attributes present,*

```
> x<-subset(titanic.train, select = -Survived)
> x_test <- subset(titanic.test, select = -Survived)
> svmModelnew = svm(Survived~Pclass + Age+ Sex + SibSp + Fare+ Parch + Embarked+fsizeD + isMother + child, data=titanic.train, usekernel = FALSE, fL = 0)
> yhat_trainnew = predict(svmModelnew, x)
> submit <- data.frame(Survived = yhat_trainnew)
> svm_errornew<- 1-sum(submit$Survived==y)/length(y)#train_error = 0.2205882
```

*The test data is predicted using SVM with the new features added:*

```
> yhat_testnew = predict(svmModelnew, x_test)
> svm_resultnew<-data.frame(Survived = yhat_testnew)
Survived
 0:34
 1:57
```

*The True Positive and the False positive prediction of SVM algorithm after Feature Engineering is:*

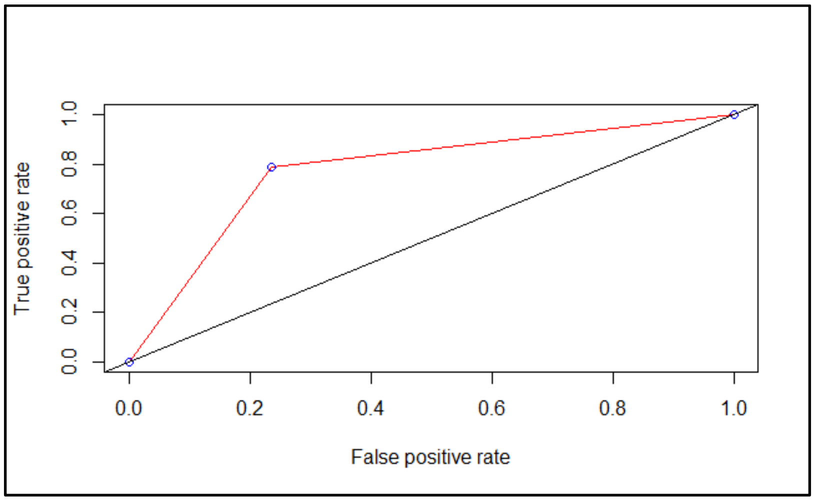

Fig 10:TPR and FPR of SVM Algorithm after Feature Engineering

## 3.5 K Nearest Neighbours

K-Nearest Neighbors is one of the simplest machine learning algorithms,but can be used to solve diffcult problems with high results. KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems. In knn for each row of the test set, k-nearest training set vectors are found using Euclidean distance, and the the classification is decided by majority vote, with ties broken at random. If there are ties for the kth nearest vector, all candidates are included in the vote.

*The K Nearest Neighbours is implemented as follows, the best 'k' value is found using cross validation using the function knn.cv in R.*

```
> library(MASS)
> library(class)
> error <- rep(0, 20)
> x<-data.frame(Pclass=as.numeric(titanic.train$Pclass),Sex= as.numeric(titanic.train$Sex), Age =
as.numeric(titanic.train$Age), SibSp = as.numeric(titanic.train$SibSp), Fare =
as.numeric(titanic.train$Fare),Embarked = as.numeric(titanic.train$Embarked))
> label <- as.numeric(titanic.train$Survived)
> for (kk in seq(from = 1, to = 20)) {
+   out <- knn.cv(x, label, k = kk)
+   Error <- 1 - sum(abs(label == out))/length(out)
+   error[kk] <- Error
+ }
> best = which.min(error) #16
> plot(error)
```

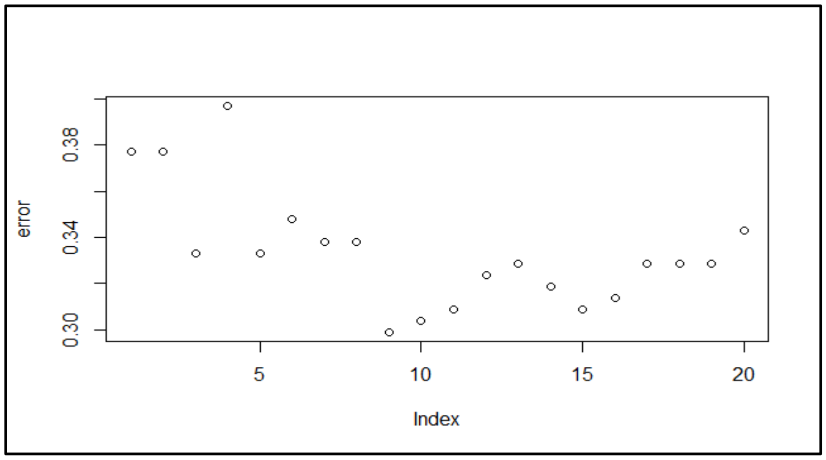

Fig 11: Error Graph in KNN with different K

```
> knn_error = error[best] # #0.306931
> yhat_train <- knn.cv(x, label, k = best)
```

*The True Positive and the False positive prediction of KNN algorithm is:*

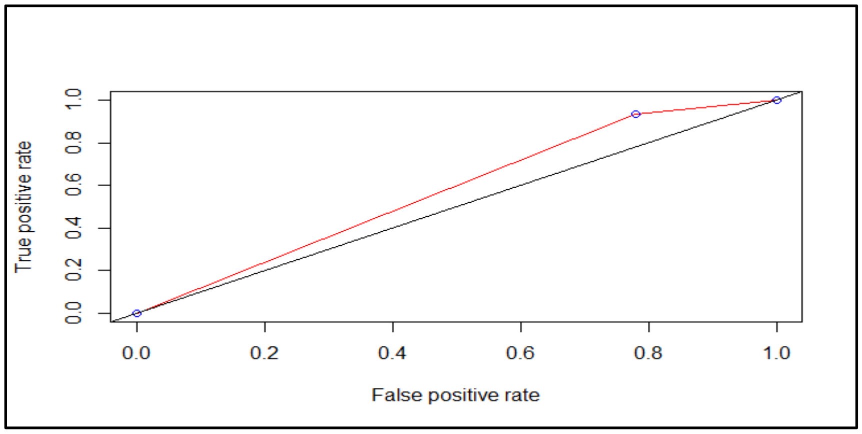

Fig 12:TPR and FPR of KNN Algorithm

*The test data is predicted using the best 'K' value found in the train data set.*

```
> # Predict the test data for k = best
> x_test<- data.frame(Pclass=as.numeric(titanic.test$Pclass), Sex = as.numeric(titanic.test$Sex),
+Age = as.numeric(titanic.test$Age), SibSp = as.numeric(titanic.test$SibSp),
+Fare = as.numeric(titanic.test$Fare),Embarked = as.numeric(titanic.test$Embarked))
> titanic.test$Survived<- as.numeric(0)
> label <- titanic.test$Survived
> yhat <- knn.cv(x_test, label, k = best)
```

*The KNN is also predicted by adding new features along with the attributes present.*

```
> error <- rep(0, 20)
> x<- data.frame(Pclass=as.numeric(titanic.train$Pclass), Sex = as.numeric(titanic.train$Sex), Age =
as.numeric(titanic.train$Age), SibSp = as.numeric(titanic.train$SibSp), Fare =
as.numeric(titanic.train$Fare),Embarked = as.numeric(titanic.train$Embarked),fsizeD =
as.numeric(titanic.train$fsizeD), isMother = as.numeric(titanic.train$isMother), child =
as.numeric(titanic.train$child))
> label <- as.numeric(titanic.train$Survived)
> for (kk in seq(from = 1, to = 20)) {
+   out <- knn.cv(x, label, k = kk)
+   Error <- 1 - sum(abs(label == out))/length(out)
+   error[kk] <- Error}
> best = which.min(error) #16
> plot(error)
```

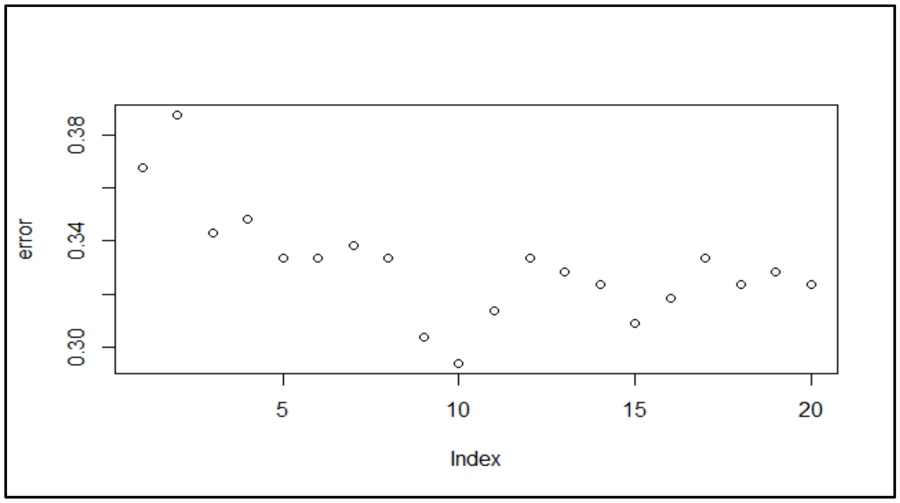

Fig 13: Error graph for different K values

```
> knn_errornew = error[best] #0.299020
> yhat_trainnew <- knn.cv(x, label, k = best)
```

*The True Positive and the False positive prediction of KNN algorithm after Feature Engineering is:*

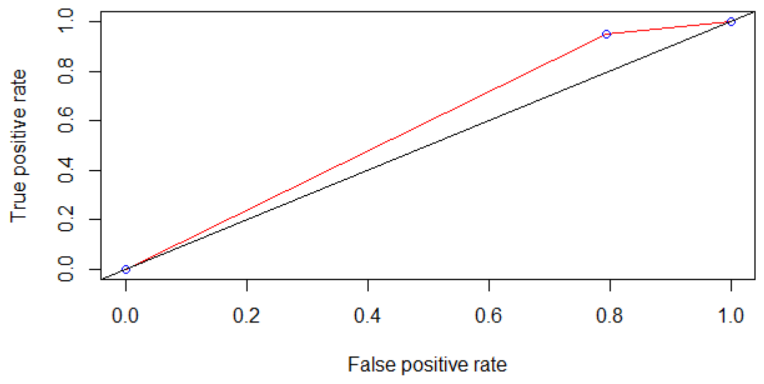

Fig 13:TPR and FPR of KNN Algorithm after Feature Engineering

*The test data is predicted using the KNN with new features along with the attributes.*

```
> x_test<- data.frame(Pclass=as.numeric(titanic.test$Pclass), Sex = as.numeric(titanic.test$Sex),Age =
as.numeric(titanic.test$Age), SibSp = as.numeric(titanic.test$SibSp), Fare = as.numeric(titanic.test$Fare),Embarked
= as.numeric(titanic.test$Embarked))
> label <- titanic.test$Survived
> yhatnew <- knn.cv(x_test, label, k = best)
```

## 4. Comparison of Algorithms

The survival of the passengers was predicted using 5 different algorithms. To check the accuracy of the prediction, we used the training data. The actual survived value is compared with the predicted survived value, this comparison is taken as the train error of each algorithm. This train error of each algorithm denotes the accuracy and efficiency of the prediction done by the algorithm.

*The train error of each algorithm is as follows:*

```
> Algorithm = c("Random forest","Decision tree","Naive Bayes","SVM", "KNN")
> TrainError = c(rf_error, dt_error, nb_error,svm_error, knn_error)
> #trainError(with new features) =c(rf_error, dt_error, nb_error,svm_error)
> df = data.frame(Algorithm, TrainError)
```

| | Algorithm | TrainError |
|---|---|---|
| 1 | Random forest | 0.09427609 |
| 2 | Decision tree | 0.16722783 |
| 3 | Naive Bayes | 0.24017957 |
| 4 | SVM | 0.21078431 |
| 5 | KNN | 0.30392157 |

*The train error of each algorithm after implementing feature engineering:*

```
> Algorithm = c("Random forest","Decision tree","Naive Bayes","SVM", "KNN")
> TrainErrorNew = c(rf_errornew, dt_errornew, nb_errornew,svm_errornew, knn_errornew)
> #trainError(with new features) =c(rf_error, dt_error, nb_error,svm_error)
> df = data.frame(Algorithm, TrainErrorNew)
```

| | Algorithm | TrainError |
|---|---|---|
| 1 | Random forest | 0.09427609 |
| 2 | Decision tree | 0.16722783 |
| 3 | Naive Bayes | 0.24017957 |
| 4 | SVM | 0.22058824 |
| 5 | KNN | 0.29901961 |

In both the cases, we observe that Random Forest produces the minimum train error. Hence, as a result to our experiments, we can conclude that random forest is the best algorithm to be implemented for the prediction of Survival of passengers in Titanic. The other algorithms have also minimal error, and so can be used for predictions.

## 5. Conclusion

After implementing all five methods, we observed that there is no significant difference in the error rates. However, so far random forest does best job by giving minimum error. We applied some featured engineering techniques to create new attributes. Even using every combination of new features, we were still not able to reduce error rate.
It seems that these features were contributing the same as already present features to predict survival rate. It would be interesting to continue this analysis by filtering more on the important features to be used for prediction.

## 6. Split of Work in Team

Team Members - Aishwarya Ganesh Murthy, Dipali Suryawanshi

Work done by team members:

- The work of report was equally split among us, we combined and did the sections by discussing the findings of our Research Work.
- **Aishwarya Ganesh Murthy -**
    - Predicting the survival of passengers using the algorithms Decision Tree, Naive Bayes, Support Vector Machines and K-Nearest Neighbours
    - Predicted the survival of passengers after performing Feature Engineering to the above algorithms
    - Calculated the training error of all the algorithms involved in prediction of survival to compare their accuracy.
    - Plotted the true positive and false positive graphs for all the algorithms involved in the prediction of survival.

- **Dipali Suryawanshi -**
    - Data exploration , prepare and clean data for model building
    - Feature Engineering to decide which features are important for model building
    - Predicting the survival of passengers using the algorithm random forest
    - Created dataframe to compare the accuracy of the algorithms using training error

## 7. Reference

[1] https://www.kaggle.com/c/titanic
[2] Tryambak Chatterjee, Prediction of Survivors in Titanic Dataset: A Comparative Study using Machine Learning Algorithms, International Journal of Emerging Research in Management &Technology ISSN: 2278-9359
[3]https://www.revolvy.com/main/index.php?s=Naive%20Bayes%20classifier&uid=1575