University College Cork

Data Mining - CS6405

Titanic dataset

Explore, Clean, Visualise

Ashwini Iral Barboza & Dhanya Sringeri Jayachandra

supervised by
Dr. Marc Van Dongen

April 15, 2018

Contents

List of Tables

List of Figures

1 Introduction

RMS Titanic was a British Passenger Liner which sank in the Atlantic ocean during its maiden voyage from the UK to New York City after colliding with an iceberg[wikipedia 2010]. The analysis attempts to predict the probability for survival of the 891 Titanic passengers and analyse main factors that could lead to the survival of a specific person .In order to do this, we used the different features available about the passengers, used a subset of the data to train an algorithm and then run the algorithm on the rest of the data set to get a prediction.

1.1 Variables

Titanic Dataset consits of 12 variables as shown in Table 1.

Variable Name	**Description**
PassengerId	Passenger Identifier
Survived	Survived (1) or died (0)
Pclass	Passenger's class
Name	Passenger's name
Sex	Passenger's sex
Age	Passenger's age
SibSp	Number of siblings/spouses aboard
Parch	Number of parents/children aboard
Ticket	Ticket number
Fare	Fare
Cabin	Cabin
Embarked	Port of embarkation

Table 1: Titanic Variables

Data Frame

The titanic Dataset has 12 Variables with 891 passengers(Observations). The data is in .csv file.

```
df

##    VariableName ClassType Size    Context
## 1   PassengerId   integer  891 predictors
## 2      Survived   integer    2     output
## 3        Pclass   integer    3 predictors
## 4          Name character  891 predictors
## 5           Sex character    2 predictors
## 6           Age   numeric   89 predictors
## 7         SibSp   integer    7 predictors
```

```
## 8         Parch   integer    7 predictors
## 9        Ticket character  681 predictors
## 10         Fare   numeric  248 predictors
## 11        Cabin character  148 predictors
## 12     Embarked character    4 predictors
```

2 Exploratory Data Analysis

We first explore the relationship of the variables with the response `Survived`. The added benefit of these plots is to see quickly if the number of passengers for each variables are more or less consistent.

Pclass

This can be considered as the social status of the passenger i.e. 1st class passenger are probably wealthier than the passengers from 3rd class. We can see that the many passengers in `Pclass`1 survived, so did about half of the people from `Pclass`2. Majority of people were from `Pclass`3 and it is evident from Figure 1 that they had lower survival rate.

```
##       class passengers
## 1 1st class        216
## 2 2nd class        184
## 3 3rd class        491
```

Passenger Name

Additional information can be fetched from the `Passenger Name`. The passengers have titles in their Names, which can be used to predict the survived passengers.

```
##
##          Capt Col Don  Dr Jonkheer Lady Major Master Miss Mlle Mme  Mr Mrs
##   female    0   0   0   1        0    1     0      0  182    2   1   0 125
##   male      1   2   1   6        1    0     2     40    0    0   0 517   0
##
##           Ms Rev Sir the Countess
##   female   1   0   0            1
##   male     0   6   1            0
```

It can be seen that there are 17 titles among the passengers.

Sex

From Figure 2, it is evident that women have higher chances of survival than male.

Age

Figure 3 indicates that `Age` is an important factor in predicting the survival of an passenger. There are few missing values which needs to be filled.

Number of Siblings/Spouses Aboard

This variable suggests if the passenger has a sibling onboard. Figure 4 indicates that the survival rate of passenger is low if there are siblings.

Number of Parents/Children Aboard

This variable indicates the relationship of the passenger with other Passengers.Figure 5 indicates that the survival rate of passenger is low if there are relationshipship.

Fare

Clearly, from the Figure 6, a big portion of passengers from lower `Fare` died.

Cabin

This variable indicates if the passenger on board had booked a cabin, he/she has higher survival rate.

Embarked

There are 3 ports of Embarkation C = Cherbourg, Q = Queenstowna and S = Southampton as shown in Figure 8. Many passengers `Embarked` in Southampton and had low chances of survival.

The code for the graphs is in Page 22 to Page 23 .

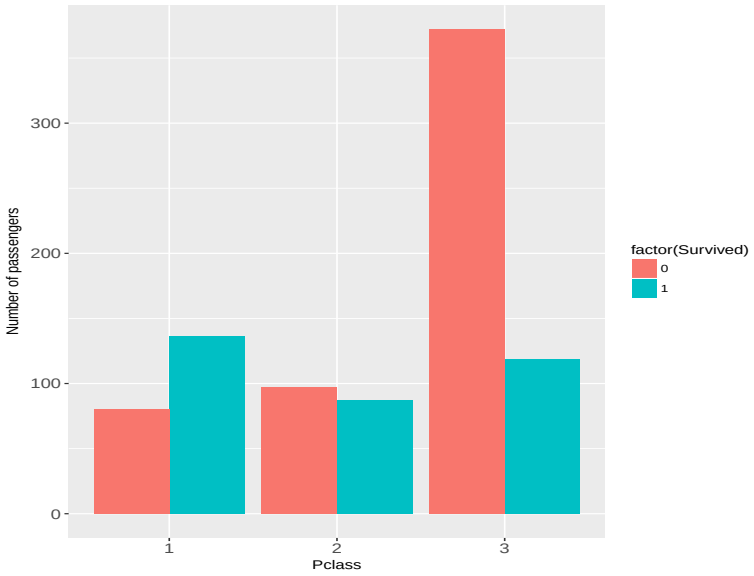

Figure 1: Relationship between Pclass vs Survived

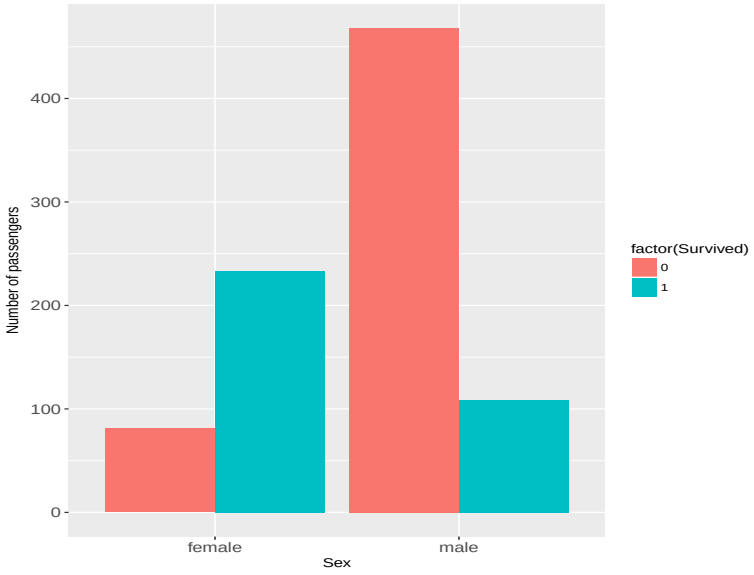

Figure 2: Relationship between Sex vs Survived

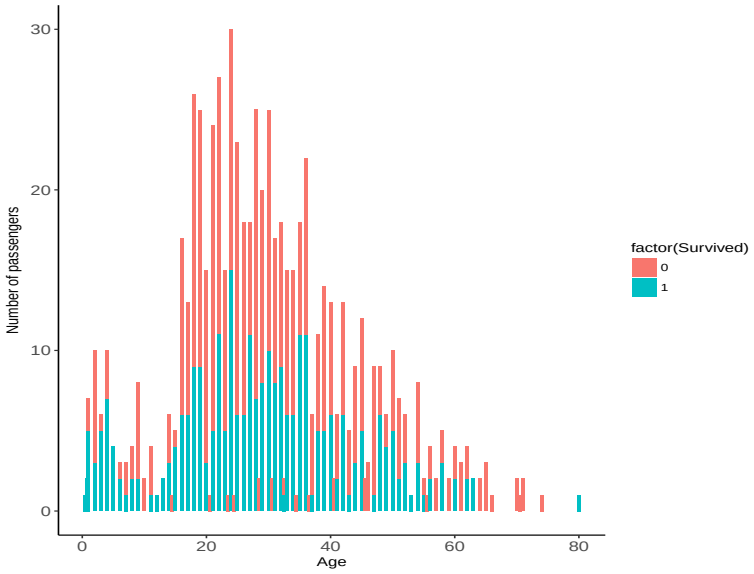

Figure 3: Relationship between Age vs Survived

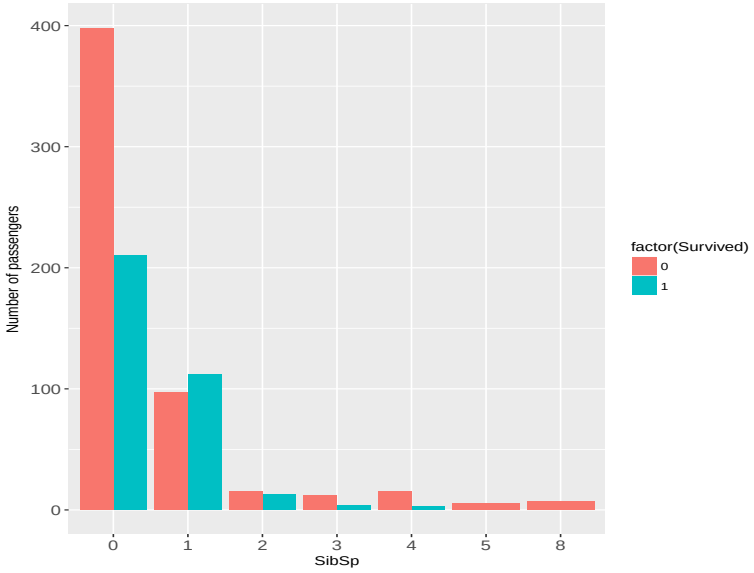

Figure 4: Relationship between SibSp vs Survived

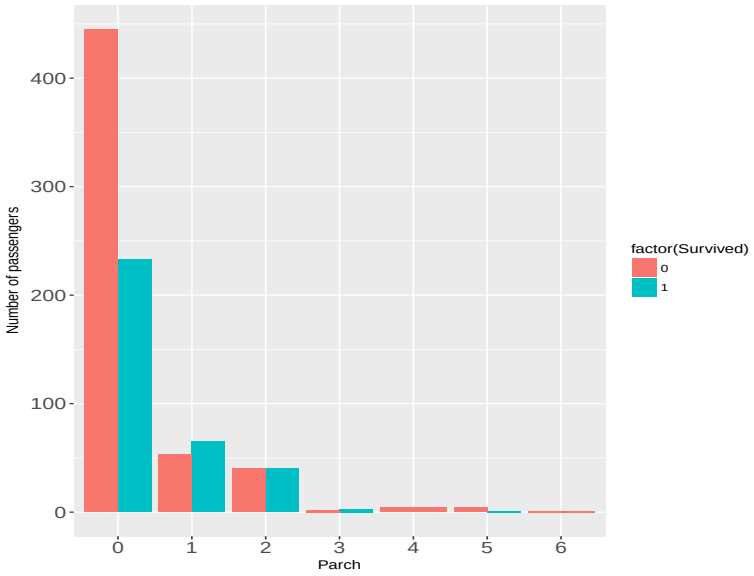

Figure 5: Relationship between Parch vs Survived

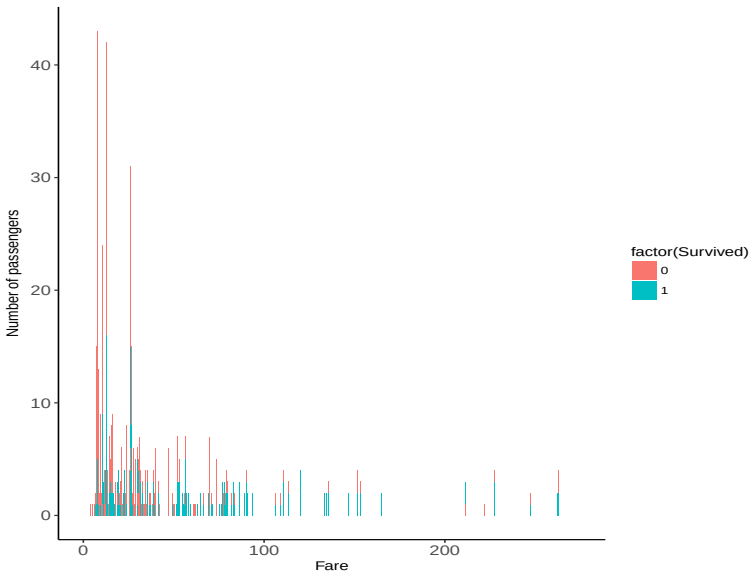

Figure 6: Relationship between Fare vs Survived

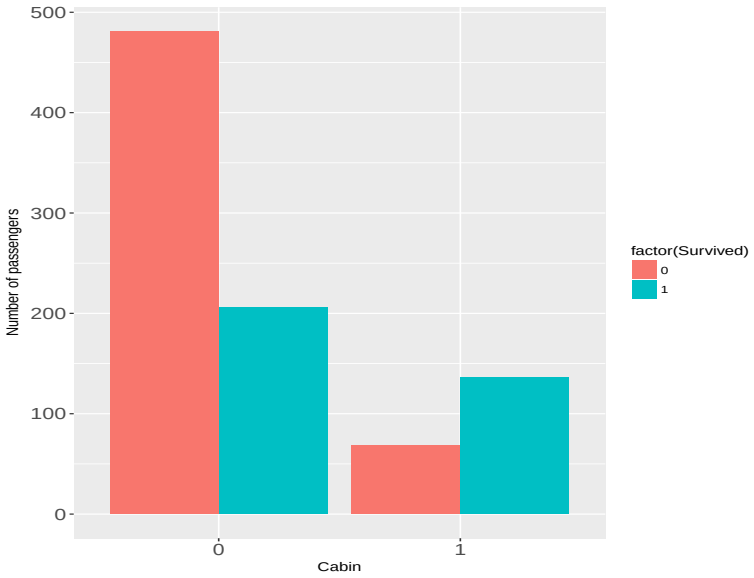

Figure 7: Relationship between Cabin vs Survived

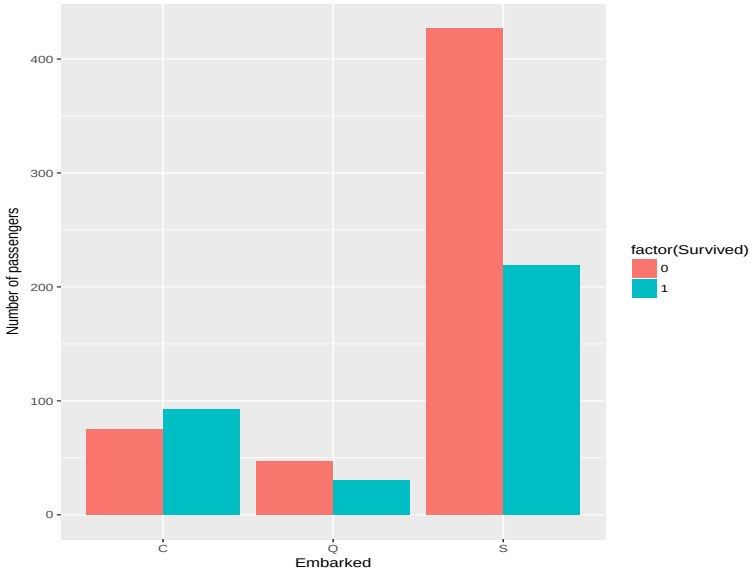

Figure 8: Relationship between Embarked vs Survived

3 Data Cleaning

3.1 Missing Values

The scatterplot matrix shown in Figure 9 gives us information about the missing values in the dataset in proportions. We can see that there are 177 passengers whose `Age` is not specified. It is also evident that `Cabin` of 687 passengers is not giving, indicating that they didnot book a cabin. Also 2 values of `Embarked` are missing.

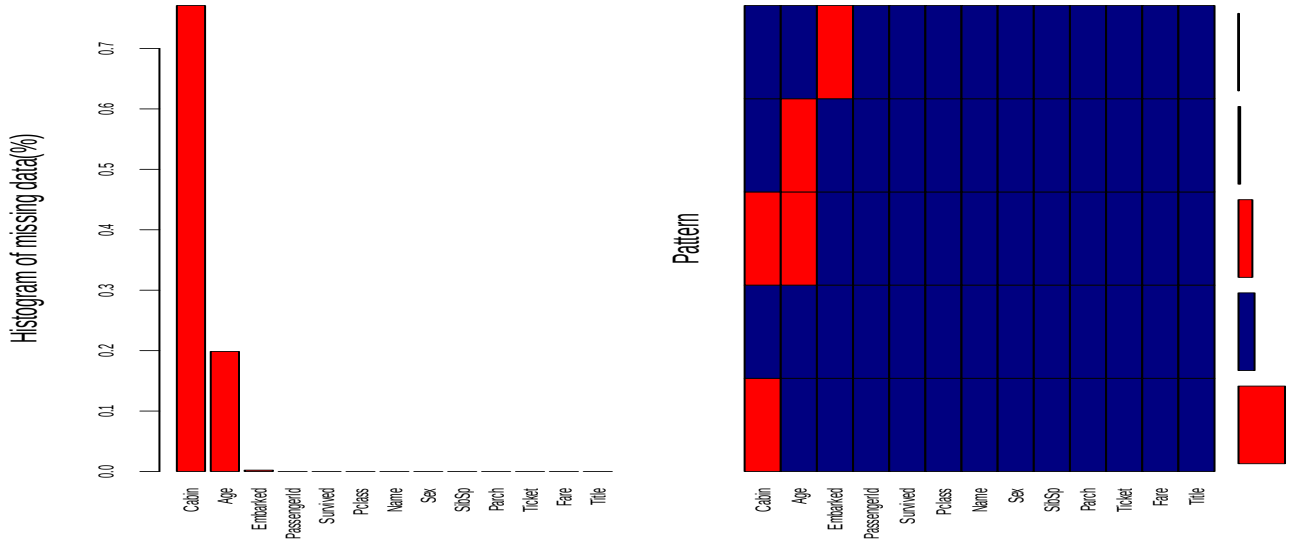

Figure 9: Graph of missing data

```
##
##  Missings in variables:
##  Variable Count
##       Age   177
##     Cabin   687
##  Embarked     2
```

Page 24 has the code.

3.2 Impution of Values

Since `Age` is a continuous variable we used prediction to find the missing values.We made a prediction of an individual passenger's `Age` using the other variables with ANOVA.

The mode of `Embarked` Column was Port S indicating that it was most common among the passengers. Therefore we imputed the missing value with Port S ,assuming these two passengers `Embarked` at S

We assumed that the empty values in `Cabin` were for passengers who did not book a cabin. Hence we converted the `Cabin` variable into a factor as person with cabin = 1 and Person without cabin = 0

After imputation of missing data, we have the following results.

```
summary(titanic$Age)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.42   22.00   27.84   29.57   37.00   80.00

levels(titanic$Embarked)

## [1] "C" "Q" "S"

levels(titanic$Cabin)

## [1] "0" "1"
```

Page 24 has the code.

3.3 Variables Transformation

The conversion of features into factors has a great advantage in predicting the response `Survived`.
We divided the `Age` into 5 categories.

```
## [1] "1" "2" "3" "4" "5"
```

We broke down the `Name` to extract the `Title` information of the passengers. There are 17 titles gives to the passengers. We converted into factor with 5 levels. Titles with very low cell counts were combined to "rare" level.

```
## [1] "Master" "Miss"   "Mr"     "Mrs"    "Rare"
```

It is evident there there is similar pattern in the survival for `Parch` and We created a new variable `FamilySize` with `SibSp` and `Parch`

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.9046  1.0000 10.0000
```

We converted Fare into a Factor with 4 levels

```
## [1] "1" "2" "3" "4"
```

Page 24 has the code.

Data for modelling

For this analysis, we have split our cleaned and transformed data into 70% for training, and 30% for testing. The code is present in Page 25. The data is split randomly using a shuffler. Each classifier was trained on the training data. We are also using a 3-fold validation to get an estimate on how our classifier would perform on the unseen data. Finally, we use our test data which is unseen by the classifier and determine the accuracy obtained on this test data. The goal is to predict as accurately as possible.
We referred An Introduction to Statistical Learning with applications in R[James et al. 2014] and The Not so Short Introduction to Latex[Oetiker et al. 2007].
After cleaning we have the following

```
dff

##   VariableName ClassType Levels
## 1     Survived    factor      2
## 2       Pclass    factor      3
## 3          Sex    factor      2
## 4        Cabin    factor      2
## 5     Embarked    factor      3
## 6        Title    factor      5
## 7     Agegroup    factor      5
## 8   FamilySize   integer
## 9     FareBand    factor      4
```

4 Analysis

4.1 Logistic Regression

In this analysis, our goal is to classify if a person survived given other variables. Since response variable is categorical,this is a classification problem. We are modelling number of success vs failures[1,0]. This data given follows a binomial distribution. Logistic regression classifier models the log odds(logit) of Survival as a linear combination of predictors. This is achieved by specifying link as logit in the model.

Model Selection

Logistic regression performs best when there is no multicollinearity. Adding terms that are explaining the same variablity that is already described by a predictor results in greater test and training errors. To identify, which predictors describe the model best, we use forward selection. Model with all the predictors. Maximum number of predictors we can have.

The stepwise selection function step() in R, adds the variables one by one, till adding any new parameter decreases the performance. We have identified this factor(Survived) Pclass + Title + FamilySize + Pclass:Title as the best model with 501.5797 as residual deviance on 520 degrees of freedom. We can confirm this is a good model based on chi-square test. Page 25 has the code.

```
best.model$deviance

## [1] 501.5797

qchisq(0.95, 520)

## [1] 574.1574
```

574.1573968 >501.579712 This model is a good fit.

Prediction

```
##           actual
## predicted   0   1
##         0 164  27
##         1  15  61
```

We are getting 84.2696629 % accuracy on test data. The prediction accuracy is not bad, but could be better.Let's consider more classifiers.
Page 25 has the code. Warnings suggesting multi-collinearity and perfect group seperation was given by glm model. Hence, we are procceding to linear discriminant analysis which does not have any issue with perfect group seperation.

4.2 Linear Discriminant Analysis

Linear discriminant classifer is more stable than logistic regression, if the groups are well seperated. In Linear discriminant analysis, we find a classifier that given prior probabilites of the observations estimates the posterior probability of a observation belonging to a particular group.
We are testing if our classifier is able to classify the training data into Survived and Non-Survived groups appropriately using cross-validation to avoid overfitting.

We are analyzing the performance of our LDA classifier using 3-fold validation. We have chosen 3 as the value,as our ratio of training to test dataset size is 2.3370787. These number of folds will approximate our test data well. 82.6923077% is the estimated accuracy of our LDA classifier on unseen data.

Prediction

Linear discrimamant analysis is able to find a linear classifer that best seperates the groups as function of the linear combination of the predictors.

```
confusion_matrix_lda

##           actual
## predicted   0   1
##         0 161  27
##         1  18  61

lda_prediction_rate

## [1] 0.8314607
```

We are able to classify 83.1460674% of the unseen data accurately.
Page 26 has the code.

4.3 Support Vector Machines

Support Vector Classifier

A support vector classifier is a technique which allows us to find a hyperplane, that seperates the observations into 2 groups, but also allows us to control the number of violations. SVC approach is very robust to variance, the hyperplane only depends on the support vectors and does not depend on all the observations that lie far from the margin unlike LDA and Logistic regression.

We are selecting a tuning parameter for our SVC model with cross-validation approach.

We initally check SVC against a range of values 0.01, 1, 5,10, 20. From the Figure 10 it is clear that the training error is minimal when cost = 10.

We are analyzing the performance of our SVC classifier using 3-fold validation. 81.0801941% is the estimated accuracy of our SVC model on unseen data.

Figure 10: Tuning parameter vs Training Error

```
confusion_matrix_svc

##           actual
## predicted   0   1
##         0 158  27
##         1  21  61

svc_prediction_rate

## [1] 0.8202247
```

We are able to classify 82.0224719% of the unseen data accurately.

Support Vector Machine

If the underlying data is not seperable by a linear hyperplane, we can fit a non-linear support vector classifier.

The gamma value corresponds to the flexibility that is desired in the model. A cross-validation approach to find best gamma value for support vector machine suggests that the best gamma value to be used as 0.05 close to 0 in Figure 11 ,indicates that the underlying data is indeed best seperated by a linear hyperplane and adding more flexibility increases the error rate.
Using the gamma value obtained by best radial model

We are analyzing the performance of our SVM classifier using 3-fold validation. 80.9294872% is the estimated accuracy of our SVM model on unseen data.

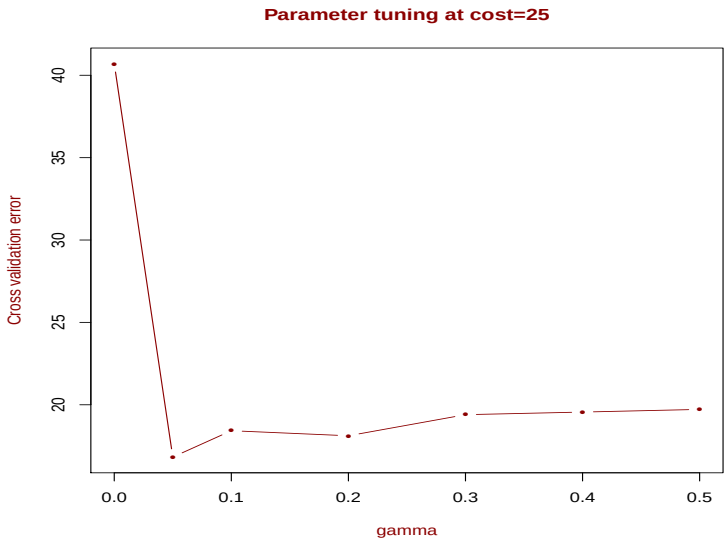

Figure 11: Kernel Parameter vs Training error

```
confusion_matrix_svm

##            actual
## prediction   0   1
##          0 161  27
##          1  18  61

svm_prediction_rate

## [1] 0.8314607

best_cost

## [1] 25

best_gamma

## [1] 0.05
```

We are able to classify 83.1460674% of the unseen data accurately.
The classification rate achieved from SVM and SVC methods is approximately the same.
Page 27 has the code.

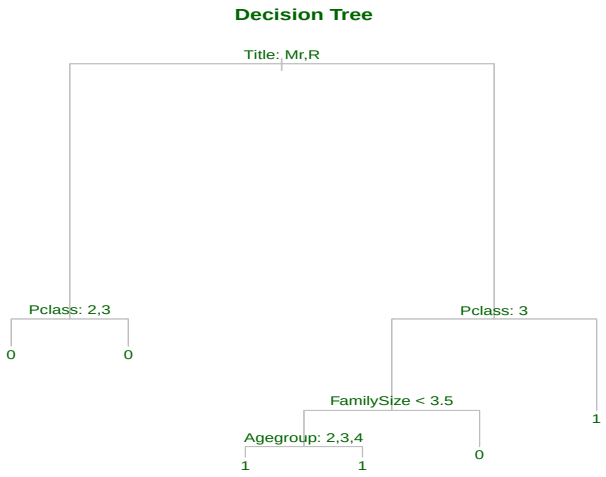

Figure 12: Decision Tree

4.4 Decision Tree

Decision trees can be used for both regression and classification problems. The tree we grow here is a classification tree. Any observation falling in a particular region of the tree is assigned to the most frequently occuring class in that region.

We are analyzing the performance of our Decision tree classifier using 3-fold validation. 75.1602564% is the estimated accuracy of our decision tree model on unseen data.

```
confuion_tree_decision_tree

##           actual
## predicted   0   1
##         0 162  27
##         1  17  61

decision_tree_rate

## [1] 0.835206
```

Though, from the Figure 12 we can see that the first split is on title. All observations with title Mr, Rare were on the left side of the tree and rest of the observations were assigned to right side.Similary, 5 other splits are identified. In the left bottom split for Pclass, we can see that on either side of its branches we have '0', and bottom right Agegroup has 1,1 split. These split was unecessary if we consider classfication rate. However, this split increases the gini index by improving node purity, and hence the split is present. We are able to classify 83.5205993% of the unseen data accurately.
Page 27 has the code.

Cross-Validation and Pruning

A smaller tree with fewer nodes improves interpretability of the model and avoids overfitting by reducing variance. Using Cross-validation we can find the optimal size of the tree.

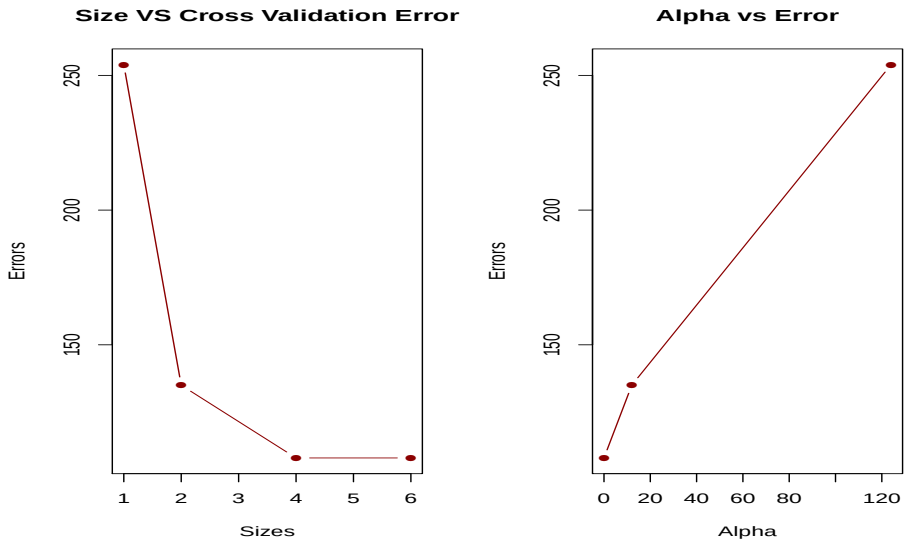

Figure 13: Optimal tree size

From the Figure 13 we can see that the error is minimal when tree size is 4. 4 is the optimal size for our tree.
Page 28 has the code.

```
confusion_matrix_prune_tree

##           actual
## predicted   0   1
##         0 162  27
##         1  17  61

prune_tree_rate

## [1] 0.835206
```

After pruning, our classification rate remians the same but, the real gain here is the low cost complexity,the tree is much smaller than the original, has lower variance and is able to give better results with fewer splits.
We are able to classify 83.5205993% of the unseen data accurately.
Page 28 has the code.

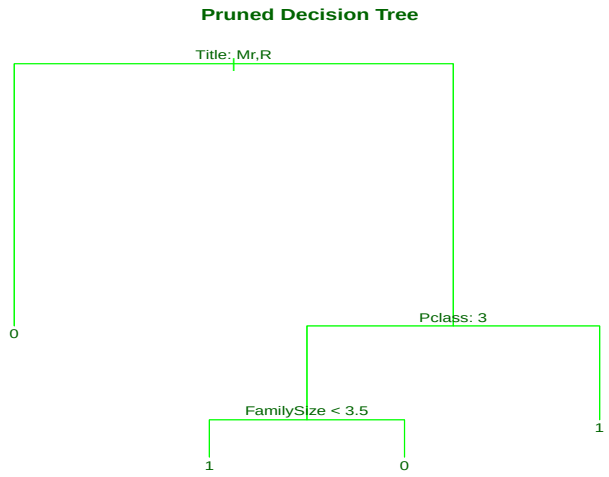

Figure 14: Pruned Decision Tree

4.5 Random Forest

The single tree models are high in variance. Many methods like bagging, random forests overcome this disadvantage by growing many trees, obtaining prediction from each tree and considering majority vote as the prediction of that observation.
Random forests have an advantage over bagging, because it decorrelates the trees, by applying restriction on the predictors that can be chosen for splitting while creating the tree and hence we choose random forests for our prediction. This ensures there is no overfitting.

Figure 15 shows the model error rate.The black line shows the overall error rate which is around 20%. The red and green lines show the error rate for 'died' and 'survived' respectively. We can see that right now we're much more successful predicting death than we are survival which is also visible from the confusion matrix

Out of bag error estimate is observed as 81.6721704% for our random forest classifier. This serves as a measure of performance metric of our classifier on the unseen data.

```
confusion_matrix_random_forest

##           actual
## predicted   0   1
##         0 164  25
##         1  15  63

random_forest_rate

## [1] 0.8501873
```

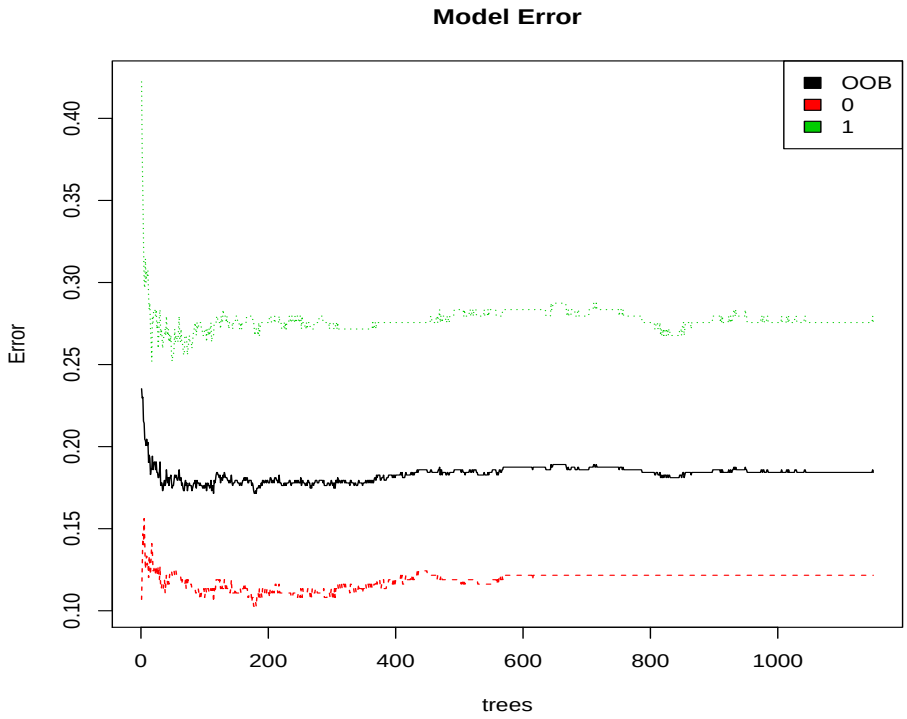

Figure 15: Model Error

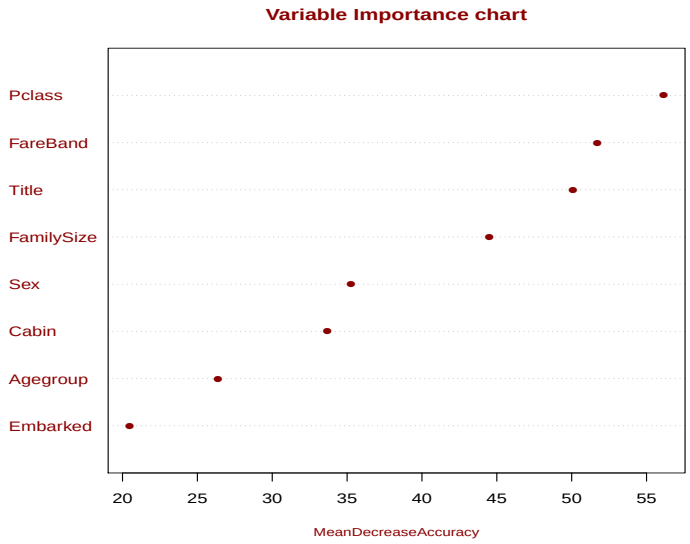

Figure 16: Variable Importance chart

Figure 16 explains the important variables.By default, the randomforest function assumes sqrt(p) variables when building random forests for classification problems. Hence here number of predictors considered for each tree = sqrt(8)=2.828 = 3. The prediction accuracy obtained by this method is the best so far.
We are able to classify 85.0187266% of the unseen data accurately.
Page 28 has the code.

5 Results

Logistic regression gives an accuracy of 84.2696629% on our data using least squares approach. However, due to perfect group seperation warnings, this prediction rate may not be accurate.
Linear discriminant analysis is another popular classification technique and works by maximizing the difference between 2 groups. When we fit this classifier to our data, we observe 83.1460674% in performance on the test data.
Support vector classifier gives the flexibility to control the number of violations, using cross-validation we found best parameters for cost function. The SVC model is just a special case of SVM when gamma = 25, using cross validation approach we find the best combination of gamma and cost parameter to improve the classification rate.
We can observe that the decision tree and pruned decision tree are giving the same prediction rate, this is a good result as our pruned tree is able to generalize just as well as the full length tree.
Random forests are taking voting from 1150 trees constructed using 3 predictors, and as expected has slightly improved the performance. We chose this model to be the best model.The final prediction from random forest is saved in /data/processed.
We see a very unusual pattern, where our cross-validation error is always lower than test error rate. Possible reasons:
Our test dataset has not captured the properties of entire dataset and much easier to predict, whereas our training dataset has all the hard to predict observations.
Our classifiers are very good at generalizing.

Method	**Cross Validation Accuracy**	**Test Accuracy**
Logistic Regression	81.8910256	84.2696629
Linear Discriminant Analysis	82.6923077	83.1460674
Support Vector Classifier	81.0801941	82.0224719
Support Vector Machines	80.9294872	83.1460674
Decision Tree	75.1602564	83.5205993
PrunedDecision Tree	83.0128205	83.5205993
Random Forest	81.6721704	85.0187266

Table 2: Results

References

James, Gareth et al. [2014]. *An Introduction to Statistical Learning* with Applications in R. Springer. ISBN: 978-1-4614-7137-0.

Oetiker, Tobias et al. [2007]. *The Not so Short Introduction to LaTeX 2ε*. URL: http://tobi.oetiker.ch/lshort/.

wikipedia [2010]. *RMS Titanic*. URL: https://en.wikipedia.org/wiki/RMS_Titanic.

A Libraries

```
# Required Libraries
library(knitr)
library(mice)
library(VIM)
library(randomForest)
library(dplyr)
library(ggvis)
library(ggplot2)
library(ggthemes)
library(rpart)
library(rpart.plot)
library(MASS)
library(e1071)
library(tidyverse)
library(tree)
library(ISLR)
```

```
# Loading Files
data_dir <- file.path(".", "data")
outpath <- file.path(data_dir, "processed", "Titanic.csv")
csvpath <- file.path(data_dir, "external", "train.csv")
titanic <- read.csv(csvpath, header = TRUE, stringsAsFactors = FALSE)
```

B Exploratory Data Analysis

```
# Plot indicating relationship of Pclass with Survived
ggplot(data = titanic[1:891, ], aes(x = factor(Pclass), fill = factor(Survived))) +
    geom_bar(stat = "count", position = "dodge") + theme(axis.text = element_text(size = 12)) +
    labs(title = "Survivors by Passenger Class", x = "Pclass", y = "Number of passengers")
```

```r
# Extracting Titles from Names
titanic$Title <- gsub("(.*, )|(\\..*)", "", titanic$Name)
tab <- table(titanic$Sex, titanic$Title)
titleCol <- ncol(tab)
```

```r
# Plot indicating relationship of Sex with Survived
ggplot(data = titanic[1:891, ], aes(x = factor(Sex), fill = factor(Survived))) +
    geom_bar(stat = "count", position = "dodge") + theme(axis.text = element_text(size = 12))
```

```r
# Plot indicating relationship of Age with Survived
df <- titanic
d <- df %>% dplyr::select(Age, Survived) %>% filter(!is.na(Age))
ggplot(d, aes(Age, fill = factor(Survived))) + stat_count(width = 0.5) + theme_classic() +
    theme(axis.text = element_text(size = 12))
```

```r
# Plot indicating relationship of SibSp with Survived
ggplot(data = titanic[1:891, ], aes(x = factor(SibSp), fill = factor(Survived))) +
    geom_bar(stat = "count", position = "dodge") + theme(axis.text = element_text(size = 12))
```

```r
# Plot indicating relationship of Parch with Survived
ggplot(data = titanic[1:891, ], aes(x = factor(Parch), fill = factor(Survived))) +
    geom_bar(stat = "count", position = "dodge") + theme(axis.text = element_text(size = 12))
```

```r
# Plot indicating relationship of Fare with Survived
ggplot(titanic[1:891, ], aes(Fare, fill = factor(Survived))) + stat_count(width = 0.5) +
    theme_classic() + xlim(0, 275) + theme(axis.text = element_text(size = 12))
```

```r
# Plot indicating relationship of Cabin with Survived
dd <- titanic %>% mutate(hascabin = Cabin != "") %>% group_by(hascabin, Survived) %>%
    summarise(n = n()) %>% mutate(survivalrate = n/sum(n)) %>% dplyr::select(hascabin,
    Survived, survivalrate) %>% spread(Survived, survivalrate)
colnames(dd) <- c("has cabin", "died", "survived")
```

```r
# Plot indicating relationship of Embarked with Survived
ggplot(data = titanic[1:891, ], aes(x = factor(Embarked), fill = factor(Survived))) +
    geom_bar(stat = "count", position = "dodge")
```

C Data Cleaning

```
# Plot indicating missing values
df <- titanic
df$Survived <- factor(df$Survived)
df$Sex <- factor(df$Sex)
df$Pclass <- factor(df$Pclass)
df$Embarked <- factor(df$Embarked)
is.na(df$Embarked[df$Embarked == ""]) <- TRUE
is.na(df$Cabin[df$Cabin == ""]) <- TRUE
NAPlot <- aggr(df, col = c("navyblue", "red"), sortVars = TRUE, labels = names(data),
    cex.axis = 0.7, gap = 3, ylab = c("Histogram of missing data(%)", "Pattern"),
    ylim = 1)
```

```
# Age Imputation
predicted_age <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked +
    Survived, data = titanic[!is.na(titanic$Age), ], method = "anova")
titanic$Age[is.na(titanic$Age)] <- predict(predicted_age, titanic[is.na(titanic$Age),
    ])
```

```
# Embark Imputation
titanic[c(62, 830), "Embarked"] <- "S"
titanic$Embarked <- as.factor(as.character(titanic$Embarked))
```

```
# Cabin Imputation
CabinBi <- ifelse(titanic$Cabin == "", 0, 1)
titanic <- titanic %>% mutate(Cabin = CabinBi)
titanic$Cabin <- factor(titanic$Cabin)
```

```
# Age transformation
titanic$Agegroup[titanic$Age < 18] <- 1
titanic$Agegroup[titanic$Age >= 18 & titanic$Age < 25] <- 2
titanic$Agegroup[titanic$Age >= 25 & titanic$Age < 40] <- 3
titanic$Agegroup[titanic$Age >= 40 & titanic$Age < 60] <- 4
titanic$Agegroup[titanic$Age >= 60] <- 5
titanic$Agegroup <- factor(titanic$Agegroup)
```

```
# Title transformation
titanic$Title <- gsub("(.*, )|(\\..*)", "", titanic$Name)
table(titanic$Sex, titanic$Title)
rare <- c("Capt", "Col", "Don", "Dr", "Jonkheer", "Lady", "Major", "Rev", "Sir",
    "the Countess")
titanic$Title[titanic$Title %in% rare] <- "Rare"
titanic$Title[titanic$Title == "Mlle"] <- "Miss"
titanic$Title[titanic$Title == "Ms"] <- "Miss"
titanic$Title[titanic$Title == "Mme"] <- "Mrs"
titanic$Title <- factor(titanic$Title)
```

```
# Fare transformation
titanic$FareBand[titanic$Fare < 10] <- 1
titanic$FareBand[titanic$Fare < 50 & titanic$Fare >= 10] <- 2
titanic$FareBand[titanic$Fare <= 100 & titanic$Fare >= 50] <- 3
titanic$FareBand[titanic$Fare > 100] <- 4
titanic$FareBand <- factor(titanic$FareBand)
```

```
Titanic = subset(titanic, select = -c(PassengerId, Age, SibSp, Parch, Ticket,
    Fare))
set.seed(185)
n <- nrow(Titanic)
shuffled_Titanic <- Titanic[sample(n), ]
train_indices <- 1:round(0.7 * n)
train <- shuffled_Titanic[train_indices, ]
test_indices <- (round(0.7 * n) + 1):n
test <- shuffled_Titanic[test_indices, ]
```

D Analysis

```
# Selecting Best Model
saturated_model = glm(Survived ~ Pclass + Sex + Embarked + Agegroup + Title +
    FamilySize + FareBand + Pclass:FareBand + Embarked:Pclass + Pclass:Sex +
    Pclass:Agegroup + Pclass:Title, family = binomial(link = "logit"), data = train)
reg.model = glm(factor(Survived) ~ Pclass, family = binomial(link = "logit"),
    data = train)
step(reg.model, direction = "forward", scope = formula(saturated_model))
```

```r
# Logistic Regression
best.model = glm(factor(Survived) ~ Pclass + Title + FamilySize + Pclass:Title,
    family = binomial(link = "logit"), data = train)
modelpred = best.model$deviance
summary(best.model)
probabilities <- predict(best.model, newdata = test, type = "response")
predicted = ifelse(probabilities > 0.5, 1, 0)
```

```r
# Linear Discriminate Analysis
lda_output = lda(Survived ~ Pclass + Sex + Cabin + Embarked + Agegroup + Title +
    FamilySize + FareBand, data = train)
predicted = predict(lda_output, test)$class
train_control_lda = trainControl(method = "cv", number = 3)
model_lda = train(Survived ~ Pclass + Sex + Cabin + Embarked + Agegroup + Title +
    FamilySize + FareBand, data = train, trControl = train_control_lda, method = "lda")
acc_lda = sum(model_lda$results[, 2])/length(model_lda$results[, 2])
confusion_matrix_lda = table(test$Survived, predicted)
lda_prediction_rate = ClassificationRate(confusion_matrix_lda)
```

```r
# Tuning SVM to get best model.
Ranges = list(cost = c(0.01, 1, 5, 10, 20))
DEFAULT_KERNEL = "linear"
set.seed(415)
sub = subset(train, select = -c(Name))
tune.out = tune(svm, Survived ~ ., data = sub, kernel = "linear", ranges = Ranges)
plot(tune.out, main = "")
title(main = "Tuning parameter vs Training Error", col.main = "darkblue")
bestModel = tune.out$best.model
summary(bestModel)
predictor = function(data, cost = cost, kernel = kernel) {
    return(svm(Survived ~ Pclass + Sex + Embarked + Agegroup + Title + FamilySize +
        FareBand, data = data, kernel = kernel, cost = cost))
}
# cross validation
train_control_svm = trainControl(method = "cv", number = 3)
grid <- expand.grid(C = c(5))
model_svm = train(Survived ~ Pclass + Sex + Cabin + Embarked + Agegroup + Title +
    FamilySize + FareBand, data = train, trControl = train_control_svm, method = "svmLinear",
    tuneGrid = grid)
acc_svc = sum(model_svm$results[, 2])/length(model_svm$results[, 2])
# Prediction
```

```
svcFit <- predictor(data = train, cost = 5, kernel = "linear")
prediction <- predict(svcFit, test)
confusion_matrix_svc = table(test$Survived, prediction)
svc_prediction_rate = ClassificationRate(confusion_matrix_svc)
```

```
# Support Vector Machine
predictor = function(data, cost = cost, gamma = gamma, kernel = kernel) {
    return(svm(Survived ~ Pclass + Sex + Embarked + Agegroup + Title + FamilySize +
        FareBand, data = data, kernel = kernel, cost = cost, gamma = gamma))
}
Ranges = list(gamma = c(0.01, 0.05, 0.1, 0.5, 0.75, 1))
tune.out = tune(svm, Survived ~ . - Name, data = train, kernel = "radial", cost = 5,
    ranges = Ranges)
plot(tune.out, main = "")
# cross valiadtion
train_control_svm_r = trainControl(method = "cv", number = 3)
grid <- expand.grid(C = c(5), sigma = c(0.05))
model_svm_r = train(Survived ~ Pclass + Sex + Cabin + Embarked + Agegroup +
    Title + FamilySize + FareBand, data = train, trControl = train_control_svm_r,
    method = "svmRadial", tuneGrid = grid)
acc_svm = sum(model_svm$results[, 2])/length(model_svm$results[, 2])
title("Kernel Parameter vs Training error", col.main = "darkblue")
svcFit <- predictor(data = train, cost = 5, gamma = 0.05, kernel = "radial")
prediction <- predict(svcFit, test)
confusion_matrix_svm = table(test$Survived, prediction)
svm_prediction_rate = ClassificationRate(confusion_matrix_svm)
```

```
# Decision Tree
descision_tree = tree( Survived ~ . -Name, data = train)
summary(descision_tree)
plot(descision_tree, col = "grey", main = "Descision Tree")
text(descision_tree, pretty = 1, col = "darkgreen" , cex = 0.9)
title(main ="Decision Tree", col.main = "darkgreen")
predicted = predict(descision_tree, test, type='class')
confuion_tree_decision_tree=table(test$Survived,predicted)
decision_tree_rate = ClassificationRate(confuion_tree_decision_tree)
# cross validation
descision_tree = tree( Survived ~ . -Name, data = train)
summary(descision_tree)
train_control_dtree = trainControl(method="cv", number=3)
model_dtree<-train(Survived~.-Name,data=train,method="rpart",+
```

```
                        trControl=train_control_dtree)
acc_dtree = sum(model_dtree$results[,2])/length(model_dtree$results[,2])
```

```
# Error Plots in Pruning
par(mfrow = c(1, 2))
plot(cv.sizes, cv.errors, type = "b", pch = 16, col = "darkred", main = "Size VS CV Error",
    xlab = "Sizes", ylab = "Errors")
plot(cv.alphas, cv.errors, type = "b", pch = 16, col = "darkred", main = "Alpha vs Error",
    xlab = "Alpha", ylab = "Errors")
```

```
# Pruning
par(mfrow = c(1, 1))
cv.optimal.index = which.min(cv.errors)
cv.optimal.size <- cv.sizes[cv.optimal.index]
prune.train = prune.misclass(descision_tree, best = cv.optimal.size)
pred <- predict(prune.train, test, type = "class")
confusion_matrix_prune_tree = table(test$Survived, pred)
prune_tree_rate = ClassificationRate(confusion_matrix_prune_tree)
plot(prune.train, col = "green")
text(prune.train, pretty = 1, col = "darkgreen", cex = 0.9)
title(main = " Pruned Decision Tree", col.main = "darkgreen")
```

```
# Random FOrest
glm3 <- randomForest(as.factor(Survived) ~ Pclass + Sex + Cabin + Embarked +
    Agegroup + Title + FamilySize + FareBand, data = train, importance = TRUE,
    ntree = 1000)
survivedPred <- predict(glm3, test)
confusion_matrix_random_forest <- table(survivedPred, test$Survived)
random_forest_rate = ClassificationRate(confusion_matrix_random_forest)
par(mfrow = c(1, 1))
varImpPlot(glm3, pch = 16, lty = 1, col = "darkred", main = "Variable Importance chart",
    col.main = "darkred", col.lab = "darkred", cex.lab = 0.8)
out_of_bag_error = 1 - sum(glm3$err.rate[, 1])/length(glm3$err.rate[, 1])
```

E Final Prediction

```r
submit <- data.frame(PassengerId = test$Name, Survived = predicted)
write.csv(submit, file = outpath, row.names = FALSE)
```