Pattern Classification Project report

29 Apr 2015

Group members:

SAKSHI GANERIWAL	12BCE1054
TARUSHREE GANDHI	12BCE1075

Topic:

1. KAGGLE Problem- Titanic: Machine Learning from Disaster

Predict survival on the Titanic (using Excel, Python, R, and Random Forests)

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.

In this challenge, we need to complete the analysis of what sorts of people were likely to survive.

Variable Descriptions:

survival	Survival (0 = No; 1 = Yes)
pclass	Passenger Class (1 = 1st; 2 = 2nd; 3 = 3rd)
name	Name
sex	Sex
age	Age
sibsp	Number of Siblings/Spouses Aboard
parch	Number of Parents/Children Aboard
ticket	Ticket Number
fare	Passenger Fare
cabin	Cabin
embarked	Port of Embarkation (C = Cherbourg; Q = Queenstown; S = Southampton)

Pclass is a proxy for socio-economic status (SES)

1st ~ Upper; 2nd ~ Middle; 3rd ~ Lower

Age is in Years; Fractional if Age less than One (1)
If the Age is Estimated, it is in the form xx.5

With respect to the family relation variables (i.e. sibsp and parch)
some relations were ignored. The following are the definitions used
for sibsp and parch.

Sibling:	Brother, Sister, Stepbrother, or Stepsister of Passenger Aboard Titanic
Spouse:	Husband or Wife of Passenger Aboard Titanic (Mistresses and Fiances Ignored)
Parent:	Mother or Father of Passenger Aboard Titanic
Child:	Son, Daughter, Stepson, or Stepdaughter of Passenger Aboard Titanic

Other family relatives excluded from this study include cousins,
nephews/nieces, aunts/uncles, and in-laws. Some children travelled
only with a nanny, therefore parch=0 for them. As well, some
travelled with very close friends or neighbors in a village, however,
the definitions do not support such relations.

<u>IMPLEMENTATION:</u>

<u>1. GENDER – CLASS MODEL</u>

We analysed sex attribute where we found that majority of female were asved compared to male.
Hence we set the test tuples with sex value='female' to survive and male to not survive.

```
#summary of Sex
summary(train$Sex)
#expand the proportion table command
prop.table(table(train$Sex,train$Survived))
# row-wise proportion
#proportions in the 1st dimension which stands for the rows (using ‘2’ instead would give you column
proportions
prop.table(table(train$Sex,train$Survived),1)
#change submission
test$Survived <- 0
test$Survived[test$Sex == 'female'] <- 1
#submit 2
submit <- data.frame(PassengerId = test$PassengerId, Survived = test$Survived)
```

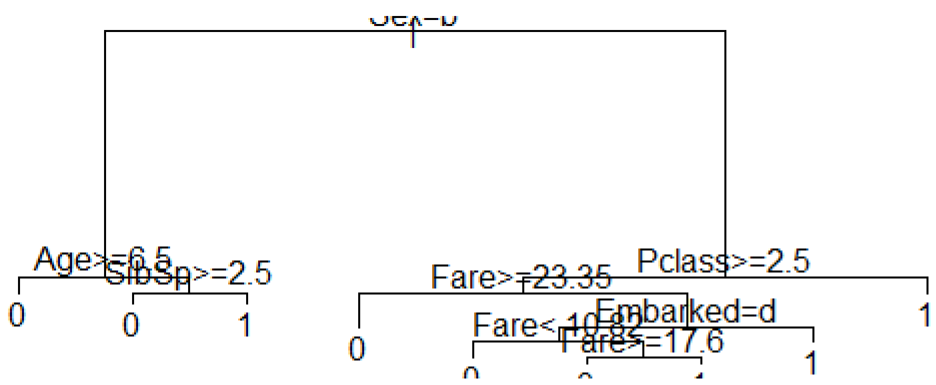
We applied binning for fair attribute in the data set where we divided it into equi-frequency bins and
analysed it.

```
summary(train$Age)
#on the basis of age divide as below 18 or above
```

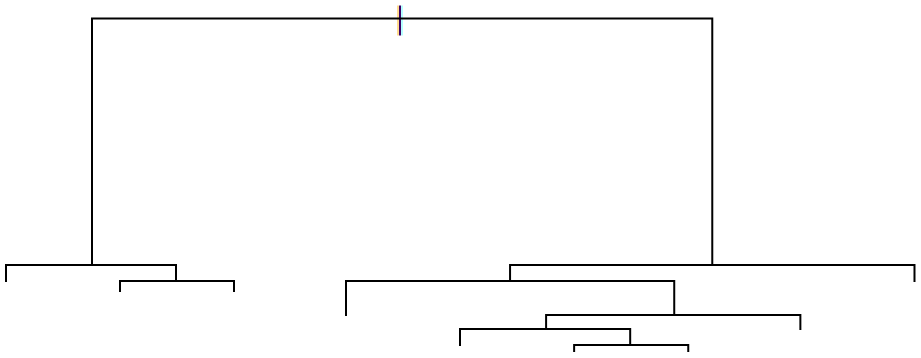
```
train$Child <- 0

train$Child[train$Age < 18] <-1
#table with both gender and age to see the survival proportions for different subsets
#number of survivors for the different subsets
aggregate(Survived ~ Child + Sex,data=train, FUN=sum)
#total no. of ppl
aggregate(Survived ~ Child + Sex, data=train, FUN=length)

aggregate(Survived ~ Child + Sex, data=train, FUN=function(x){sum(x)/length(x)})
#class and fare taken into consideration

train$Fare2 <- '30+'
train$Fare2[train$Fare <30 & train$Fare >=20] <- '20-30'
train$Fare2[train$Fare <20 & train$Fare >= 10] <- '10-20'
train$Fare2[train$Fare < 10] <- '<10'
aggregate(Survived ~ Fare2 + Pclass + Sex,data=train, FUN=function(x){sum(x)/length(x)} )
test$Survived <- 0
test$Survived[test$Sex == 'female'] <- 1
test$Survived[test$Sex == 'female' &test$Pclass == 3 & test$Fare >= 20] <- 0
#submit3
submit <- data.frame(PassengerId = test$PassengerId, Survived = test$Survived)
```

2. DESCISION TREE

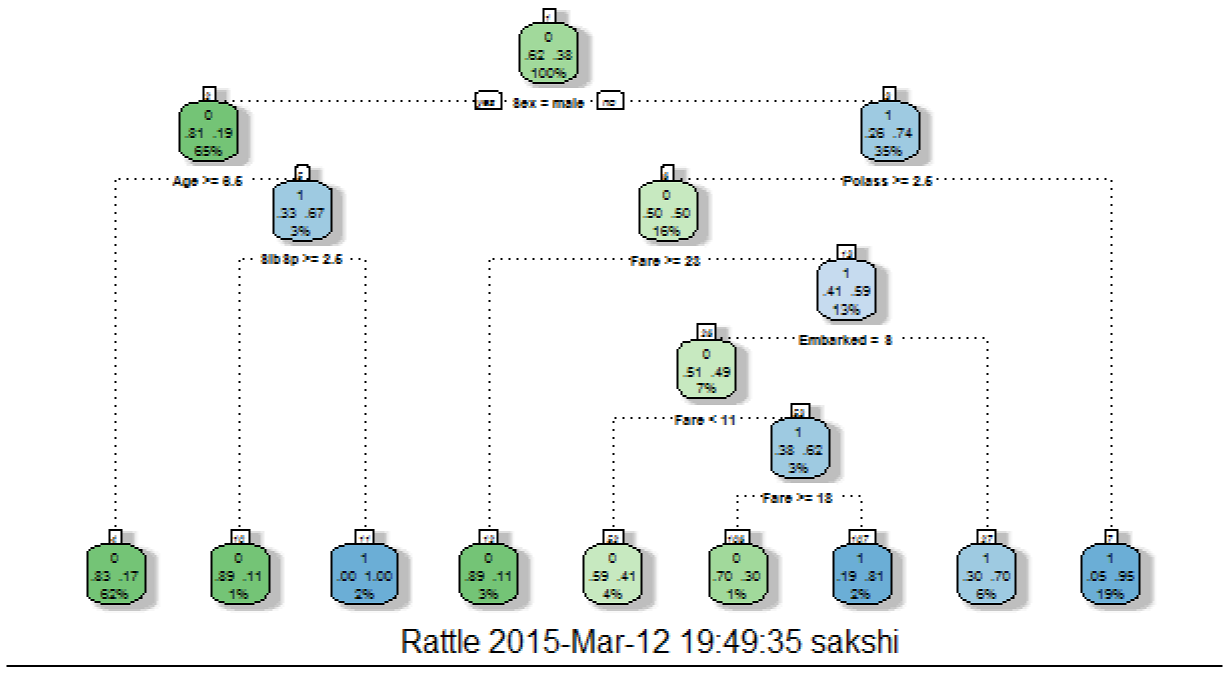
```
#machine learning to build decision trees(glass-box model) to do the heavy lifting for us.
#rpart for 'Recursive Partitioning and Regression Trees'
#import rpart
library(rpart)
#rpart same as aggregate ,equation, headed up by the variable of interest and followed by the variables
used for prediction.
# to predict a continuous variable, such as age, you may use method="anova"
#since here a one or a zero, so method="class" is appropriate
fit <- rpart(Survived ~ Pclass + Sex + Age +SibSp + Parch + Fare + Embarked,
data=train,method="class")
plot(fit)
text(fit)
install.packages('rattle')
install.packages('rpart.plot')
install.packages('RColorBrewer')
library(rattle)
library(rpart.plot)
library(RColorBrewer)
fancyRpartPlot(fit)
Prediction <- predict(fit, test, type = "class")
submit <- data.frame(PassengerId = test$PassengerId, Survived = Prediction)
```



Normal Plotted Decision tree



dendrogram

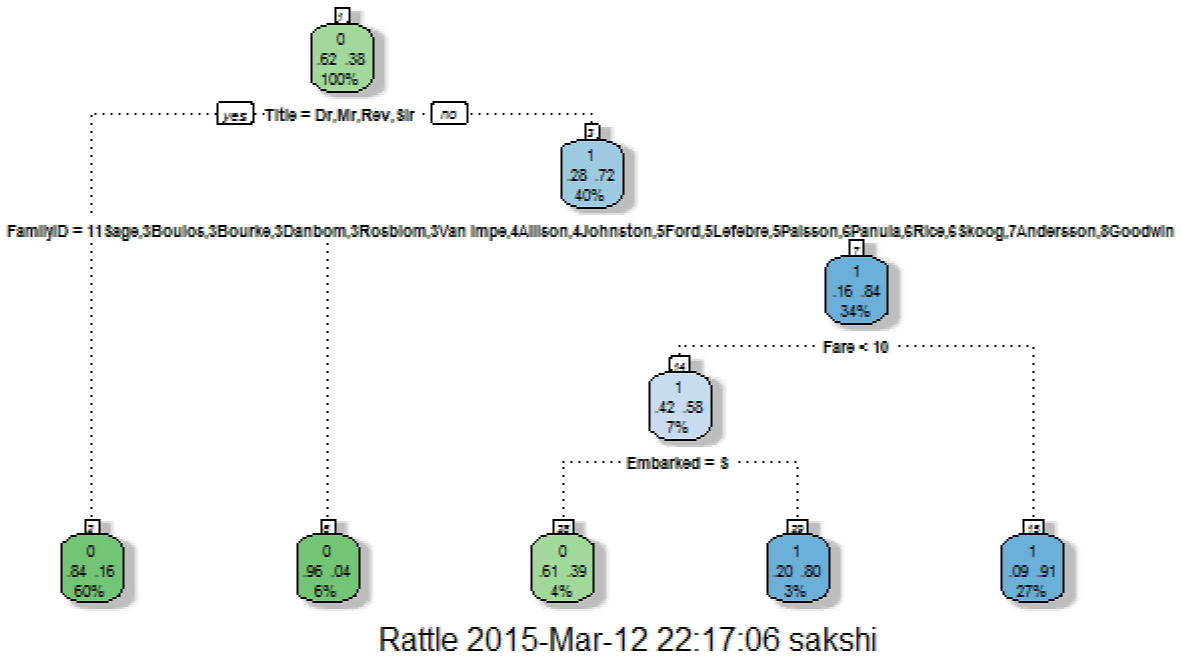


Decision Tree plotted using Rfancyplot

3. FEATURE EXTRACTION

```
#Feature Engineering
train$Name[1]
test$Survived <- NA
combi<- rbind(train,test)
combi$Name <- as.character(combi$Name)
combi$Name[1]
strsplit(combi$Name[1],split='[,.]')[[1]]
strsplit(combi$Name[1], split='[,.]')[[1]][2]
# apply this transformation to every row of the combined train/test dataframe
combi$Title <- sapply(combi$Name, FUN=function(x){strsplit(x,split='[,.]')[[1]][2]})
combi$Title <- sub(' ', '',combi$Title)
table(combi$Title)
# %in% operator checks to see if a value is part of the vector we're comparing it to
combi$Title[combi$Title %in% c('Mme','Mlle')] <- 'Mlle'
combi$Title[combi$Title %in% c('Capt', 'Don', 'Major', 'Sir')] <- 'Sir'
combi$Title[combi$Title %in% c('Dona', 'Lady', 'the Countess', 'Jonkheer')] <- 'Lady'
#decision tree biased to favour factors with many levels.
combi$Title <- factor(combi$Title)
combi$FamilySize <- combi$SibSp + combi$Parch + 1
combi$Surname <- sapply(combi$Name, FUN=function(x) {strsplit(x, split='[,.]')[[1]][1]})
combi$FamilyID <- paste(as.character(combi$FamilySize), combi$Surname, sep="")
combi$FamilyID[combi$FamilySize <= 2] <- 'Small'
table(combi$FamilyID)
```

```
famIDs <- data.frame(table(combi$FamilyID))
famIDs <- famIDs[famIDs$Freq <= 2,]
combi$FamilyID[combi$FamilyID %in% famIDs$Var1] <- 'Small'
combi$FamilyID <- factor(combi$FamilyID)
train <- combi[1:891,]
test <- combi[892:1309,]
fit <- rpart(Survived ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Title + FamilySize +
FamilyID,data=train, method="class")
fancyRpartPlot(fit)
#submit6
Prediction <- predict(fit, test, type = "class")
submit <- data.frame(PassengerId = test$PassengerId, Survived = Prediction)
```


0
.62 .38
100%
yes
Title = Dr,Mr,Rev,Sir
no
1
.28 .72
40%
FamilyID = 11Sage,3Boulos,3Bourke,3Danbom,3Rosblom,3Van Impe,4Allison,4Johnston,5Ford,5Lefebre,5Palsson,6Panula,6Rice,6Skoog,7Andersson,8Goodwin
1
.16 .84
34%
Fare < 10
1
.42 .58
7%
Embarked = S
0
.84 .16
60%
0
.96 .04
6%
0
.61 .39
4%
1
.20 .80
3%
1
.09 .91
27%
Rattle 2015-Mar-12 22:17:06 sakshi

Decision tree after implementing feature extraction

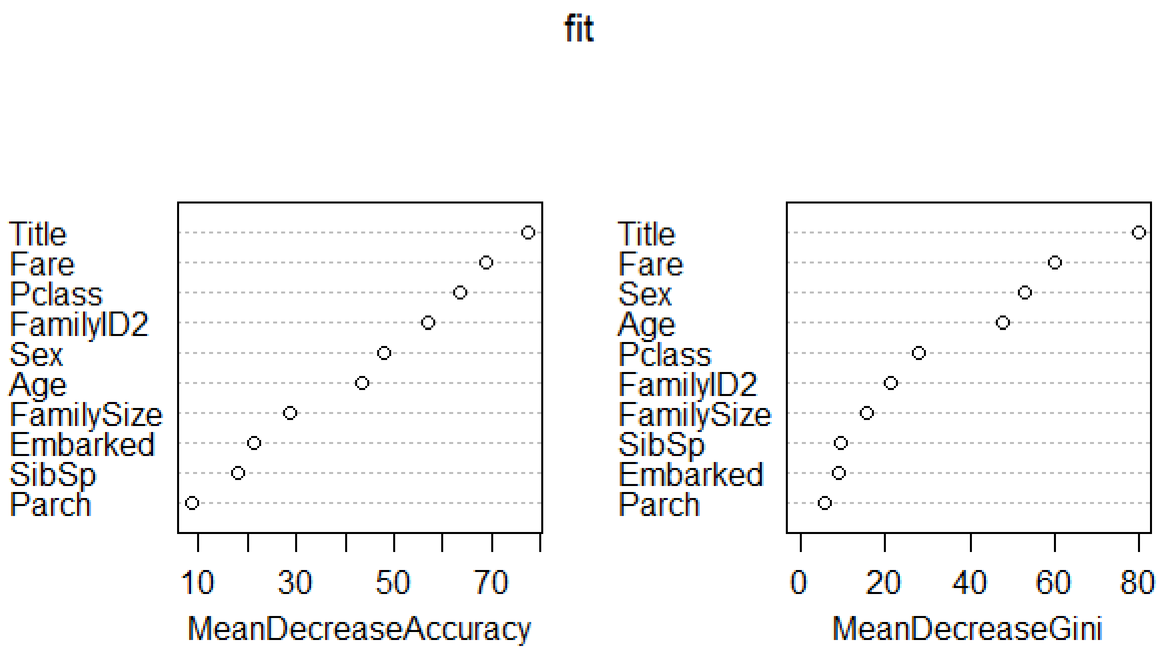
4. ENSEMBLE

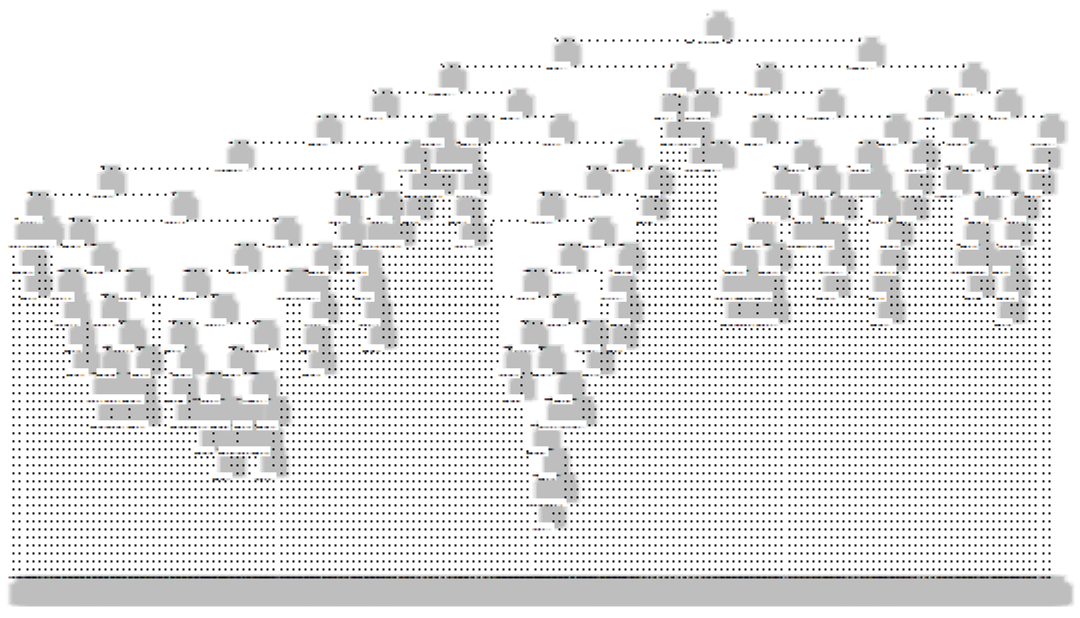
```
# ensemble models work, they grow a lot of different models, and let their outcomes be averaged or
voted across the group
#Bagging takes a randomized sample of the rows in your training set, with replacement
sample(1:10, replace = TRUE)
summary(combi$Age)
# method="anova" version of our decision tree, as we are not trying to predict a category any more, but
a continuous variable
Agefit <- rpart(Age ~ Pclass + Sex + SibSp + Parch + Fare + Embarked + Title + FamilySize,
                data=combi[!is.na(combi$Age),], method="anova")
combi$Age[is.na(combi$Age)] <- predict(Agefit, combi[is.na(combi$Age),])
summary(combi)
```

```
# reduce the number of trees, at least for initial exploration, or restrict the complexity of each tree using
nodesize as well as reduce the number of rows sampled with sampsize. You can also override the
default number of variables to choose from with mtry, but the default is the square root of the total
number available and that should work just fine.
which(combi$Embarked == '')
combi$Embarked[c(62,830)] = "S"
combi$Embarked <- factor(combi$Embarked)
summary(combi$Fare)
which(is.na(combi$Fare))
combi$Fare[1044] <- median(combi$Fare, na.rm=TRUE)
combi$FamilyID2 <- combi$FamilyID
combi$FamilyID2 <- as.character(combi$FamilyID2)
combi$FamilyID2[combi$FamilySize <= 3] <- 'Small'
combi$FamilyID2 <- factor(combi$FamilyID2)
install.packages('randomForest')
library(randomForest)
set.seed(415)
fit <- randomForest(as.factor(Survived) ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked +
Title + FamilySize +
                    FamilyID2, data=train, importance=TRUE, ntree=2000)
varImpPlot(fit)
Prediction <- predict(fit, test)
submit <- data.frame(PassengerId = test$PassengerId, Survived = Prediction)
```

5. FOREST GENERATION

```
install.packages('party')
library(party)
set.seed(415)
fit <- cforest(as.factor(Survived) ~ Pclass + Sex + Age + SibSp + Parch + Fare + Embarked + Title +
FamilySize + FamilyID,
              data = train, controls=cforest_unbiased(ntree=2000, mtry=3))
Prediction <- predict(fit, test, OOB=TRUE, type = "response")
submit <- data.frame(PassengerId = test$PassengerId, Survived = Prediction)
write.csv(submit, file = "secondforest.csv", row.names = FALSE)
```





Random Forest