# Random Forest

#setting up the directory

setwd("C:/Users/admin/Downloads/Kinjal R/Session 9")

#getting the data files

library(data.table)

options(scipen=999)

datat<-fread("train.csv", sep=",", header=TRUE)

datatest <- fread("test.csv", sep = ",", header = TRUE)

head(datat)

head(datatest)

#adding survived col to test dataset

datatest$Survived <- NA

tail(datatest)

#combining the dataset

tit <- rbind(datat, datatest)

head(tit)

tail(tit)

#Survived 0 = NO, 1=Yes

#checking the class of the dataset

sapply(tit,class)

str(tit)

#changing the variable types

tit$Pclass <- as.factor(tit$Pclass)

tit$Sex <- as.factor(tit$Sex)

tit$Embarked <- as.factor(tit$Embarked)

# checking values of sibsp and parch

unique(tit$SibSp)

unique(tit$Parch)

# feature engineering

head(tit$Name)

tit$title <- gsub('(.\*,) |(\\..\*)','', tit$Name)

head(tit)

#showing title by counts on sex

table(tit$Sex, tit$title)

# combing titles with low counts

rare\_title <- c('Capt', 'Col', 'Don', 'Dona', 'Dr', 'Jonkheer', 'Lady', 'Major', 'Rev',

'Sir', 'the Countess')

# Also reassign mlle, ms, and mme accordingly

tit$title[tit$title == "Mlle"] <- "Miss"

tit$title[tit$title == "Ms"] <- 'Miss'

tit$title[tit$title == 'Mme'] <- 'Mrs'

tit$title[tit$title %in% rare\_title] <- 'Rare tile'

table(tit$Sex, tit$title)

#taking surnames from names

tit$Surname <- sapply(tit$Name, function(x) strsplit(x, split = '[,.]')[[1]][1])

head(tit)

library(dplyr)

nlevels(factor(tit$Surname))

# Create a family size variable including the passenger themselves

tit$size <- tit$SibSp + tit$Parch + 1

# Create a family variable

tit$family <- paste(tit$Surname, tit$size, sep = '\_')

head(tit)

library(ggplot2)

install.packages("ggthemes")

library(ggthemes)

#understanding the size of the family over the survival

max(tit$size)

ggplot(tit[1:891,], aes(x= size, fill=factor(Survived)))+

geom\_bar(stat = 'count', position = 'dodge')+

scale\_x\_continuous(breaks =c(1:11))+

labs(x = 'Family Size') +

theme\_few()

#handling missing values

colSums(is.na(tit))

levels(tit$Embarked)

levels(tit$Pclass)

levels(tit$Sex)

# to find out which row has blank values in embraked

sum(tit$Embarked =="")

tit$Embarked[tit$Embarked==""] <- NA

tit[is.na(tit$Embarked),]

# Passengers 62 and 830 are missing Embarkment

tit[c(62,830), 'Embarked']

embark\_fare <- tit %>%

filter(PassengerId !=62 & PassengerId !=830)

# Use ggplot2 to visualize embarkment, passenger class, & median fare

ggplot(embark\_fare, aes(x = Embarked, y = Fare, fill = factor(Pclass))) +

geom\_boxplot()+

geom\_hline(aes(yintercept=80),

color = 'red', linetype = 'dashed', lwd =2)+

theme\_few()

# Since their fare was $80 for 1st class, they most likely embarked from 'C'

tit$Embarked[c(62, 830)] <- 'C'

# checking NA values in passsenger list

colSums(is.na(tit))

tit[is.na(tit$Fare),]

# Replace missing fare value with median fare for class/embarkment

tit$Fare[1044] <- median(tit[tit$Pclass =='3' & tit$Embarked =='S' ]$Fare, na.rm = TRUE)

# Show number of missing Age values

sum(is.na(tit$Age))

install.packages("mice")

library(mice)

init <- mice(tit, matix = 0)

predM <- init$predictorMatrix

predM[, c("PassengerId", "Name","Ticket","Cabin")]=0

imp<-mice(tit, m=5, predictorMatrix = predM)

# Get the final data-frame with imputed values filled in 'Age'

tit <- complete(imp)

View(tit)

sum(is.na(tit$Age))

# Feature engineering on age to get the mother and child details

# First we'll look at the relationship between age & survival

ggplot(tit[1:891,], aes(Age, fill = factor(Survived))) +

geom\_histogram()+

facet\_grid(.~Sex)+

theme\_few()

# Create the column child, and indicate whether child or adult

tit$child[tit$Age < 18] <- 'Child'

tit$child[tit$Age >= 18 ] <- 'Adult'

# Show counts

table(tit$child, tit$Survived)

# adding mother varibale

tit$mother <- 'Not mother'

head(tit$mother)

tit$mother[tit$Sex == 'female' & tit$Parch >0 & tit$Age >18 & tit$title != 'Miss'] <- 'Mother'

# Show counts

table(tit$mother, tit$Survived)

tit$child <- factor(tit$child)

tit$mother <- factor(tit$mother)

#making sure no mroe missing values

md.pattern(tit)

sum(is.na(tit))

colSums(is.na(tit))

head(tit$child)

#tit[is.na(tit$Embarked),]

tit[is.na(tit$child),]

# Prediction

# Spliting the dataset into train and test

train <- tit[1:891,]

test <- tit[892:1309,]

# Building the model using random forest

set.seed(754)

# Build the model (note: not all possible variables are used)

library(randomForest)

train$title <- as.factor(train$title)

test$title <- as.factor(test$title)

train$mother <- as.factor(train$mother)

test$mother <- as.factor(test$mother)

rf\_model <- randomForest(factor(Survived) ~ Pclass + Sex + Age + SibSp + Parch +

Fare + Embarked + title + size + child + mother,

data = train)

str(train)

# Show model error

plot(rf\_model, ylim = c(0,0.36))

legend('topright', colnames(rf\_model$err.rate), col = 1:3, fill = 1:3)

#The black line shows the overall error rate which falls below 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. What does that say about me, I wonder?

# looking for varibale importance

importance <- importance(rf\_model)

varImportance <- data.frame(Variables = row.names(importance),

Importance = round(importance[,'MeanDecreaseGini'],2))

#create a rank variable based on improtance

rankImportance <- varImportance %>%

mutate(Rank = paste0('#', dense\_rank(desc(Importance))))

# Using ggplot to visualise the importance of variables

ggplot(rankImportance, aes(x = reorder(Variables, Importance),

y = Importance, fill = Importance))+

geom\_bar(stat = 'identity') +

geom\_text(aes(x = Variables, y = 0.5, label = Rank),

hjust=0, vjust=0.55, size = 4, colour = 'red') +

labs(x = 'Variables') +

coord\_flip() +

theme\_few()

# final prediction using random forest

prediction <- predict(rf\_model, test)

# Save the solution to a dataframe with two columns: PassengerId and Survived (prediction)

solution <- data.frame(PassengerID = test$PassengerId, Survived = prediction)

# Write the solution to file

write.csv(solution, file = 'rf\_mod\_Solution.csv', row.names = F)

# checking the accuracy

conf <- rf\_model$confusion

conf

rf\_model