Success Prediction

Garima Sood

March 14, 2018

```
library(ggplot2)
library(gridExtra)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:base':
##
##     date
```

```
library(Metrics)
library(rpart)
library(rpart.plot)
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
dataPath <- "C:/Users/garim/Documents/Quarter 2/Data Mining/Project/Merged data"
master_data <- read.csv(paste(dataPath,"master_data_with_imputed_budget_and_revenue.csv", sep =
"/"))
```

Data Processsing

```
master_data$release_date <- as.Date(master_data$release_date)

#To cut the impact of inflation on movie revenues & budgets, I am excluding data of movies relea
sed before Jan 1985

master_data <- master_data[master_data$release_date > as.Date("01/01/1985","%m/%d/%Y"),]
master_data <- master_data[master_data$budget > 0,]
master_data$actor_1_gender <- as.factor(ifelse(master_data$actor_1_gender==0,NA,ifelse(master_da
ta$actor_1_gender==2,1,0)))
master_data$actor_2_gender <- as.factor(ifelse(master_data$actor_2_gender==0,NA,ifelse(master_da
ta$actor_2_gender==2,1,0)))
master_data$actor_3_gender <- as.factor(ifelse(master_data$actor_3_gender==0,NA,ifelse(master_da
ta$actor_3_gender==2,1,0)))
master_data$actor_4_gender <- as.factor(ifelse(master_data$actor_4_gender==0,NA,ifelse(master_da
ta$actor_4_gender==2,1,0)))
master_data$actor_5_gender <- as.factor(ifelse(master_data$actor_5_gender==0,NA,ifelse(master_da
ta$actor_5_gender==2,1,0)))
master_data$director_gender <- as.factor(ifelse(master_data$director_gender==0,NA,ifelse(master_
data$director_gender==2,1,0)))
master_data$producer_gender <- as.factor(ifelse(master_data$producer_gender==0,NA,ifelse(master_
data$producer_gender==2,1,0)))
master_data$collection <- as.factor(ifelse(nchar(as.character(master_data$belongs_to_collectio
n))>0,1,0))

master_data$num_prod_comp <-(master_data$production_company_1!="")+(master_data$production_compa
ny_2!="")+
                            (master_data$production_company_3!="")

master_data$num_prod_ctry <-(master_data$production_country_1!="")+(master_data$production_count
ry_2!="")+
                            (master_data$production_country_3!="")

master_data$release_month <- month.abb[month(master_data$release_date)]

master_data <- master_data[ , -which(names(master_data) %in%
              c( "movie_id" ,"actor_1_name","actor_2_name","actor_3_name","actor_4_name","actor_
5_name","director_name","producer_name",
                 "casting_gender","casting_name","belongs_to_collection","genre_2","genre_3","ge
nre_4","production_company_1",
                 "production_company_2","production_company_3" ,"production_country_1", "product
ion_country_2",  "production_country_3" , "spoken_language_1","spoken_language_2", "spoken_langu
age_3" ,"homepage","imdb_id" ,"original_title","overview","poster_path", "status","title","vide
o"))]
```

Plots show that there are a lot of NA values in the different columns. Counting the NA values per column in the data

```
perc_na <- function(x){
  return(sum(is.na(x))/length(x))
}

round(apply(master_data, 2, function(x) perc_na(x)),2)
```

```
##    actor_1_gender    actor_2_gender    actor_3_gender    actor_4_gender 
##              0.27              0.30              0.32              0.35 
##    actor_5_gender   director_gender   producer_gender           genre_1 
##              0.39              0.39              0.55              0.17 
##             adult            budget original_language        popularity 
##              0.17              0.17              0.17              0.17 
##      release_date           revenue           runtime           tagline 
##              0.17              0.17              0.17              0.17 
##      vote_average        vote_count        collection     num_prod_comp 
##              0.17              0.17              0.17              0.17 
##     num_prod_ctry     release_month 
##              0.17              0.17
```

```
master_data$na_count <- apply(master_data, 1, function(x) sum(is.na(x)))
table(master_data$na_count)
```

```
## 
##    0    1    2    3    4    5    6    7    8   22 
## 2550 1895 1053  571  386  300  262  244    4 1508
```

Deleting records with missing data in more than 9 columns, and checking the poportion of missing values in the updated data set

```
data <- master_data[master_data$na_count<9,]
dim(data)
```

```
## [1] 7265   23
```

```
round(apply(data, 2, function(x) perc_na(x)),2)
```

```
##    actor_1_gender    actor_2_gender    actor_3_gender    actor_4_gender 
##              0.12              0.15              0.18              0.22 
##    actor_5_gender   director_gender   producer_gender           genre_1 
##              0.26              0.27              0.46              0.00 
##             adult            budget original_language        popularity 
##              0.00              0.00              0.00              0.00 
##      release_date           revenue           runtime           tagline 
##              0.00              0.00              0.00              0.00 
##      vote_average        vote_count        collection     num_prod_comp 
##              0.00              0.00              0.00              0.00 
##     num_prod_ctry     release_month          na_count 
##              0.00              0.00              0.00
```

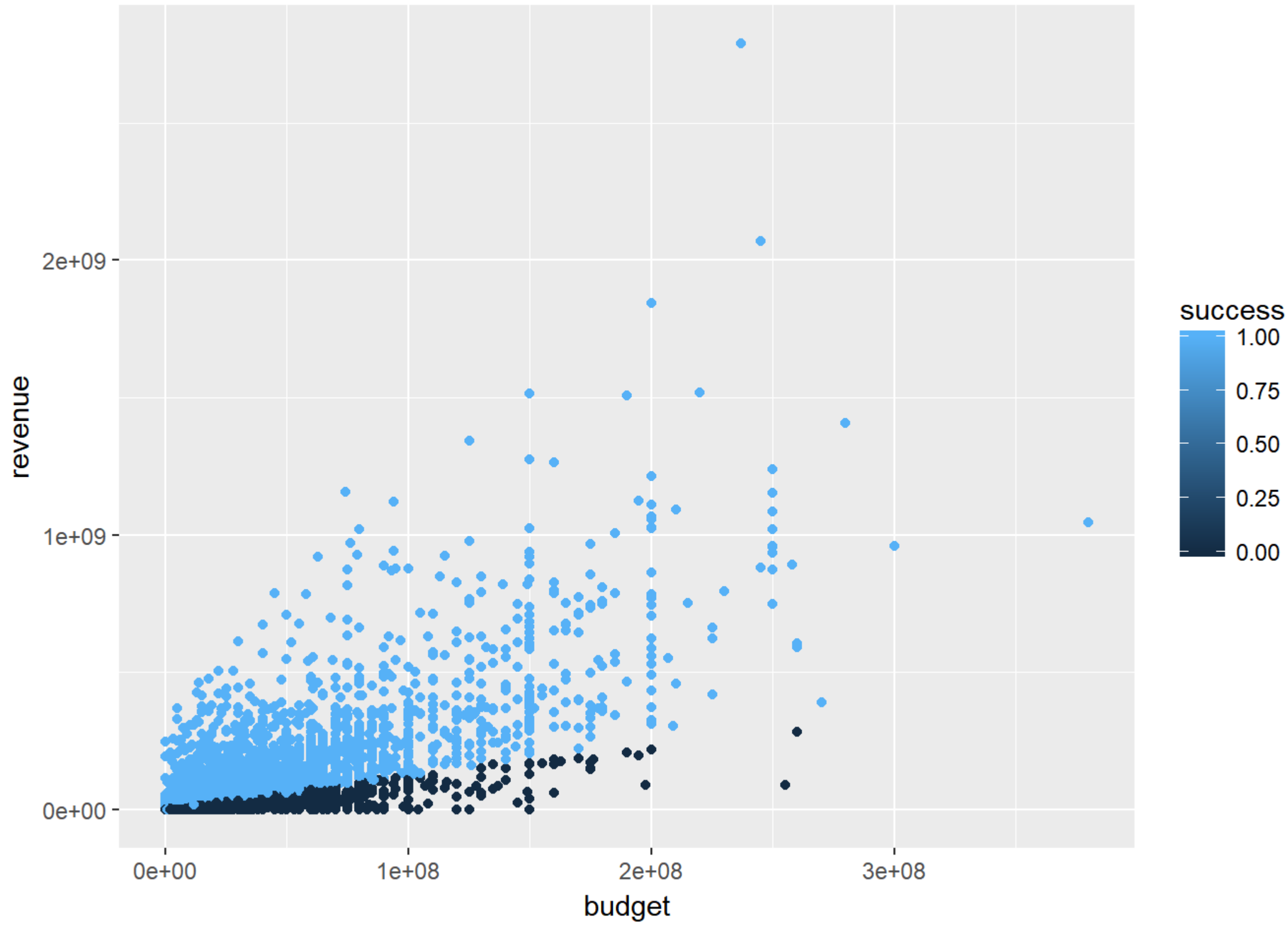
After removing thee records, we are left with about 10% missing values in the gender column of lead actors, and 24% missing values in budget. Rest of the columns look good.

Checking the distribution of our dependent variable (revenue)

```
data$success <- (ifelse(data$revenue/data$budget >1.25,1,0))
```

Plot of budget & revenue

```
ggplot(data,aes(budget,revenue,colour = success)) + geom_point()
```



I see a few outliers. But first I will impute the missing values and then remove the outliers if needed.

Missing value estimation:

```
mean_impute <- function(x){
  a<- (mean(x[!is.na(x)]))
  x <- ifelse(is.na(x), a, x)
  return(x)
}

median_impute <-  function(x){
  a<- (median(x[!is.na(x)]))
  x <- ifelse(is.na(x), a, x)
  return(x)
}

mode_impute <-  function(x){
  ux <- (unique(x))
  a<-ux[which.max(tabulate(match(x[!is.na(x)], ux)))]
  x <- ifelse(is.na(x), a, x)
  return(x)
}

data.imp <-data

#Imputing missing data in gender and runtime columns using mode and median respectively

data.imp$actor_1_gender <- as.factor(mode_impute(data$actor_1_gender))
data.imp$actor_2_gender <- as.factor(mode_impute(data$actor_2_gender))
data.imp$runtime <- median_impute(data.imp$runtime)

perc_blank <- function(x){
  return(sum(x ==""|x==" ")/length(x))
}
round(apply(data.imp, 2, function(x) perc_blank(x)),2)
```

```
##    actor_1_gender    actor_2_gender    actor_3_gender    actor_4_gender
##              0.00              0.00                NA                NA
##    actor_5_gender   director_gender   producer_gender           genre_1
##                NA                NA                NA              0.02
##             adult            budget original_language        popularity
##              0.00              0.00              0.00              0.00
##      release_date           revenue           runtime           tagline
##              0.00              0.00              0.00              0.26
##      vote_average        vote_count        collection     num_prod_comp
##              0.00              0.00              0.00              0.00
##     num_prod_ctry     release_month          na_count           success
##              0.00              0.00              0.00              0.00
```

Removing the extreme revenue records

```
sd.budget <- sqrt(var(data.imp$budget))
sd.budget*6
```

```
## [1] 220472844
```

```
length(data.imp$budget[data.imp$budget>2e+08])
```

```
## [1] 32
```

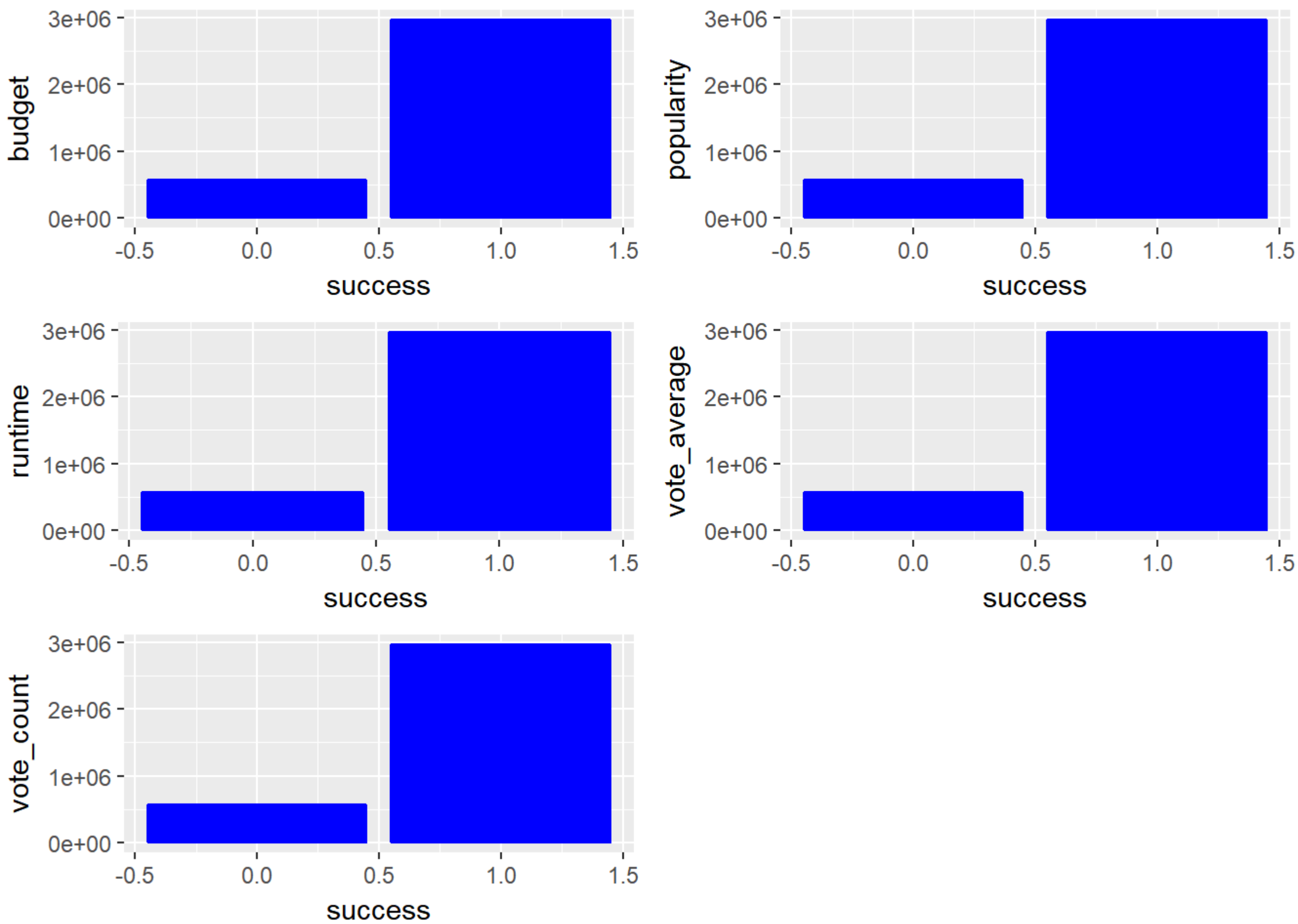
```
data.imp <- data.imp[data.imp$budget<2e+08,]
```

```
data.final <-data.imp[,-which(names(data.imp) %in% c( "actor_2_gender", "actor_3_gender", "actor_4_gender", "actor_5_gender",  "director_gender", "producer_gender", 'na_count', 'genre_1', 'adult','tagline','original_language'))]
data.final$actor_1_gender <- ifelse(data.final$actor_1_gender==1, "Female", "Male")
data.final$collection <- ifelse(data.final$collection==1, "Yes", "No")


#Splitting the data into test & train
c <- round(nrow(data.final)*0.7,0)
s <- sample(1:nrow(data.final), c)

train <- data.final[s,]
test <- data.final[-s,]
```

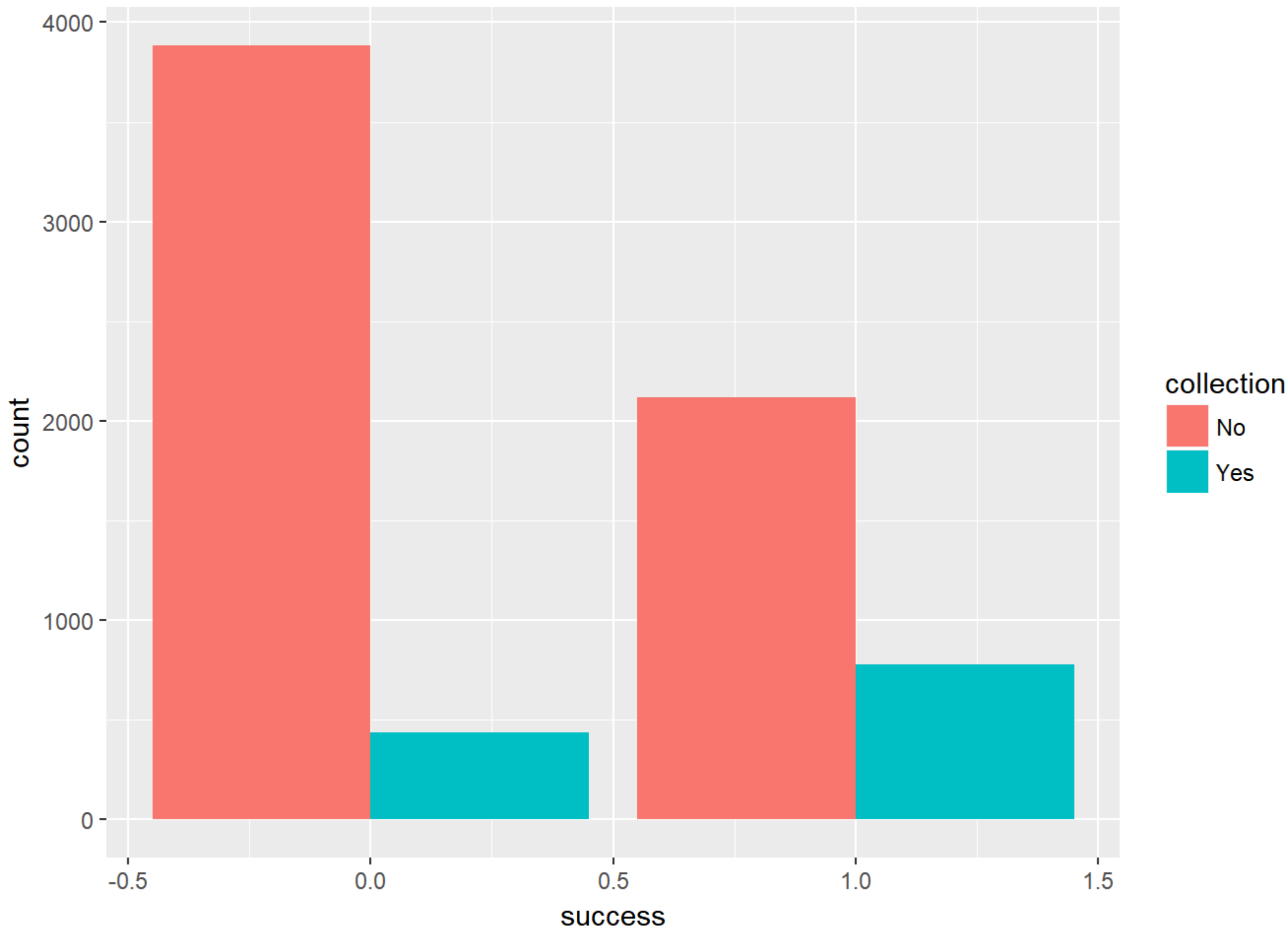
Building a model to predict success of the movie before it is released. I will not model the vote count and vote average variables as they are collected after the release of the movie.

Build a multiple linear model for revenue prediction

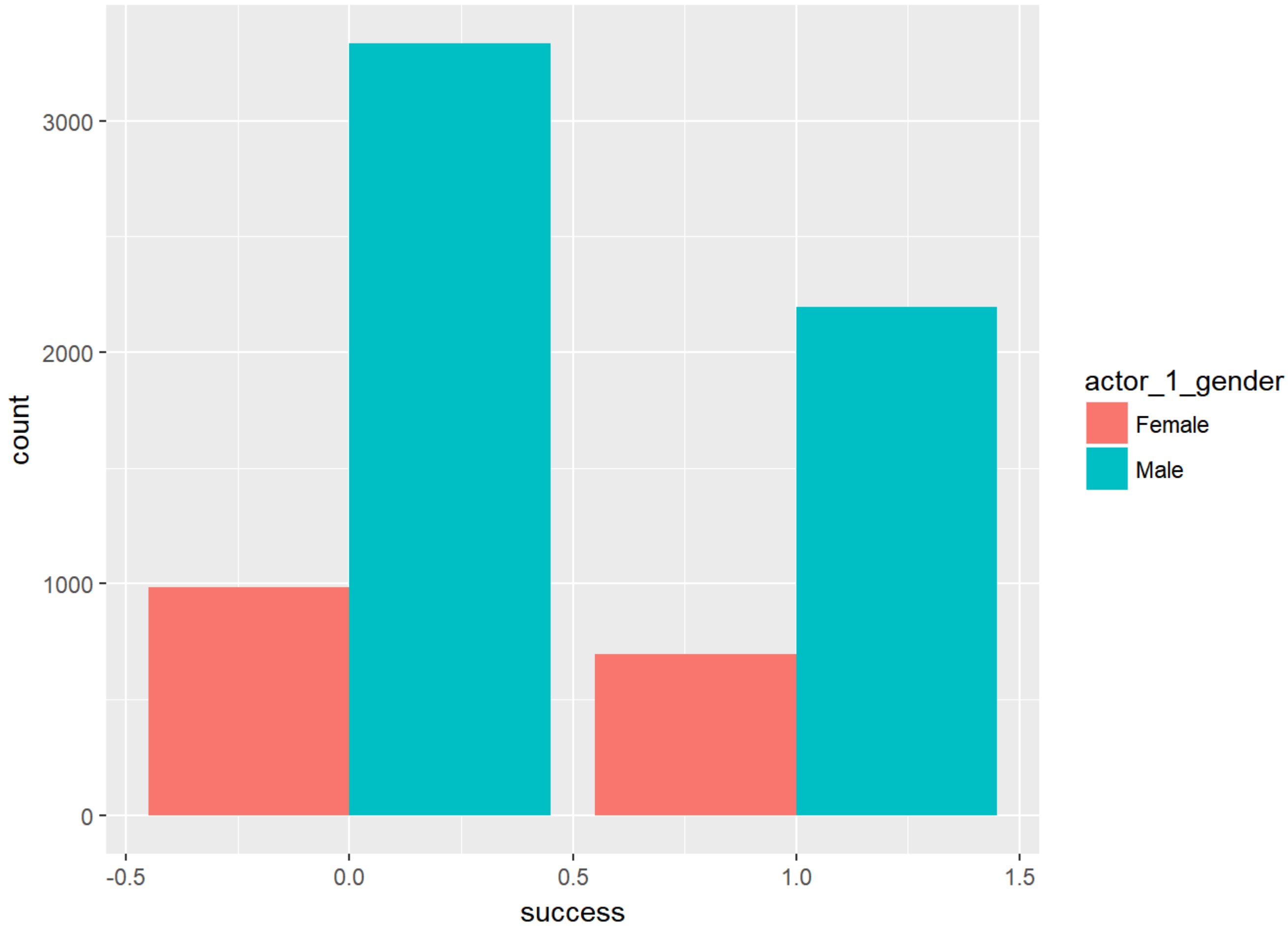
```
for (i in c("budget","popularity","runtime", "vote_average", "vote_count")){
  assign(paste0("p",i),ggplot(data.final, aes(success, eval(parse(text = i))))+labs(y = i)+geom_bar(stat = "identity", color = "blue"))
}

grid.arrange(pbudget,ppopularity,pruntime, pvote_average, pvote_count,nrow= 3,ncol = 2)
```



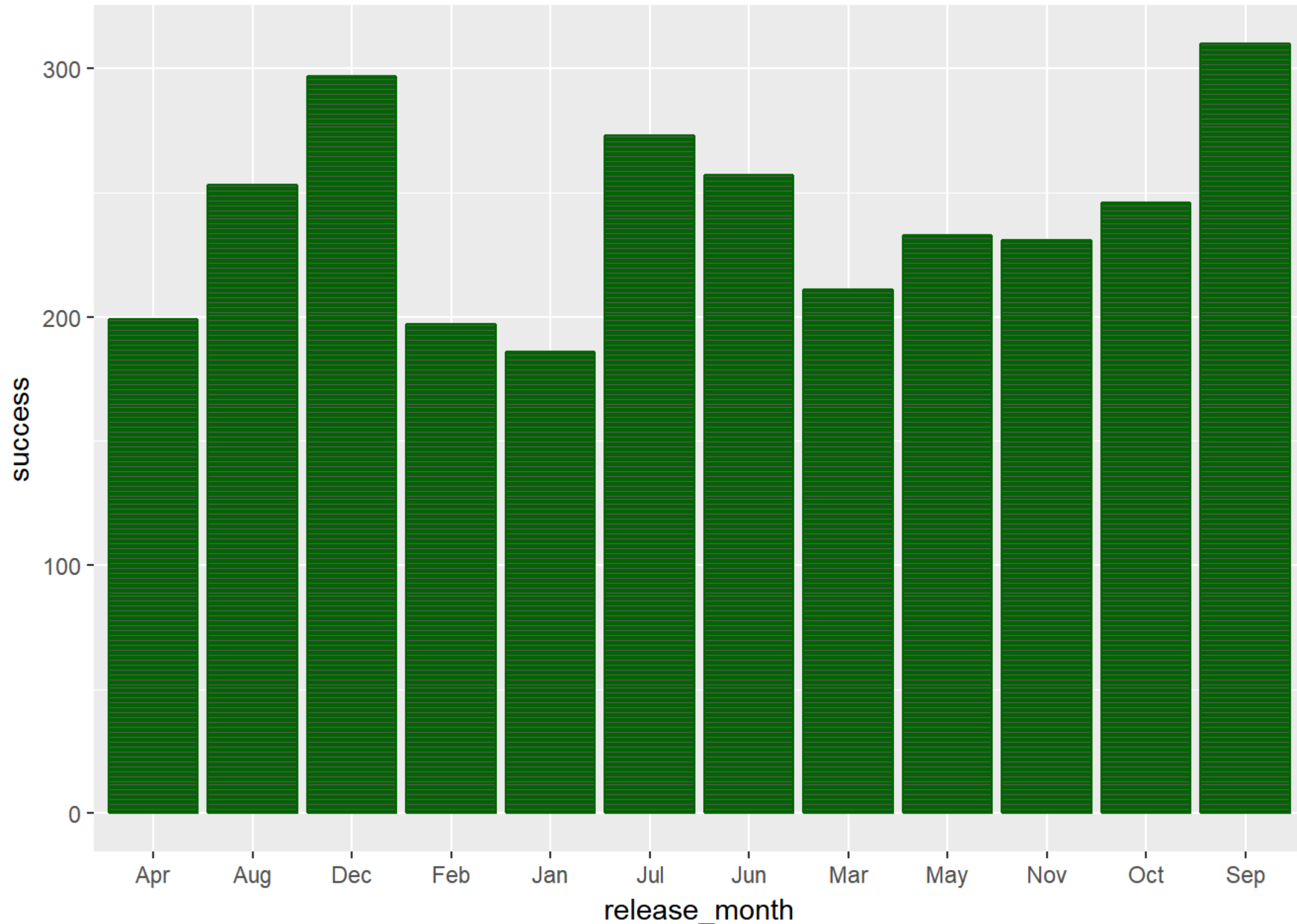

```
ggplot(data=data.final, aes(x=success, y=..count..)) + geom_bar(aes(fill = collection), position
= "dodge")
```




```
ggplot(data=data.final, aes(x=success, y=..count..)) + geom_bar(aes(fill = actor_1_gender), posi
tion = "dodge")
```




```
ggplot(data=data.final, aes(x=release_month, y=success)) + geom_bar(stat = 'identity', color =
'dark green')
```



Initial review of the graphs say that success of a movie is indicated by budget, popularity, runtime (suprisingly), movie belonging to a collectionand voting statistics

Movie success is actually dependent on the month of launch! Larger proportion of movies released in the summer or late in the year are successful

Fitting a logistic regression model

```
success_pred <- glm(success~ actor_1_gender+ popularity+runtime+collection+num_prod_comp+num_pro
d_ctry+release_month, data = train, family = binomial(link = "logit"))
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
summary(success_pred)
```

```
##
## Call:
## glm(formula = success ~ actor_1_gender + popularity + runtime +
##     collection + num_prod_comp + num_prod_ctry + release_month,
##     family = binomial(link = "logit"), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.3805  -0.8418  -0.5141   0.9594   2.3610
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -2.736833   0.201858 -13.558  < 2e-16 ***
## actor_1_genderMale -0.173050   0.077014  -2.247   0.0246 *
## popularity          0.182277   0.007693  23.693  < 2e-16 ***
## runtime             0.007903   0.001340   5.899 3.65e-09 ***
## collectionYes       0.945508   0.088375  10.699  < 2e-16 ***
## num_prod_comp       0.224087   0.034852   6.430 1.28e-10 ***
## num_prod_ctry      -0.225283   0.054774  -4.113 3.91e-05 ***
## release_monthAug    0.100999   0.163591   0.617   0.5370
## release_monthDec    0.121430   0.161788   0.751   0.4529
## release_monthFeb    0.089610   0.170007   0.527   0.5981
## release_monthJan   -0.083123   0.171190  -0.486   0.6273
## release_monthJul    0.344144   0.171790   2.003   0.0451 *
## release_monthJun    0.392938   0.175330   2.241   0.0250 *
## release_monthMar   -0.088683   0.169066  -0.525   0.5999
## release_monthMay    0.004425   0.170263   0.026   0.9793
## release_monthNov    0.114160   0.172391   0.662   0.5078
## release_monthOct   -0.301153   0.161013  -1.870   0.0614 .
## release_monthSep   -0.136519   0.153329  -0.890   0.3733
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6807.1  on 5046  degrees of freedom
## Residual deviance: 5471.6  on 5029  degrees of freedom
## AIC: 5507.6
##
## Number of Fisher Scoring iterations: 5
```

```
yTr <- ifelse(success_pred$fitted.values>0.5,1,0)
yTest <- ifelse(predict(success_pred,test, type = "response")>0.5,1,0)

#confusion matrix for train data
round(prop.table(table(actual = train$success, pred=yTr),1),2)
```

```
##       pred
## actual    0    1
##      0 0.83 0.17
##      1 0.42 0.58
```

```
accTr <- sum(train$success==yTr)/nrow(train)
#confusion matrix for test data
round(prop.table(table(actual = test$success, pred=yTest),1),2)
```

```
##       pred
## actual    0    1
##      0 0.84 0.16
##      1 0.41 0.59
```

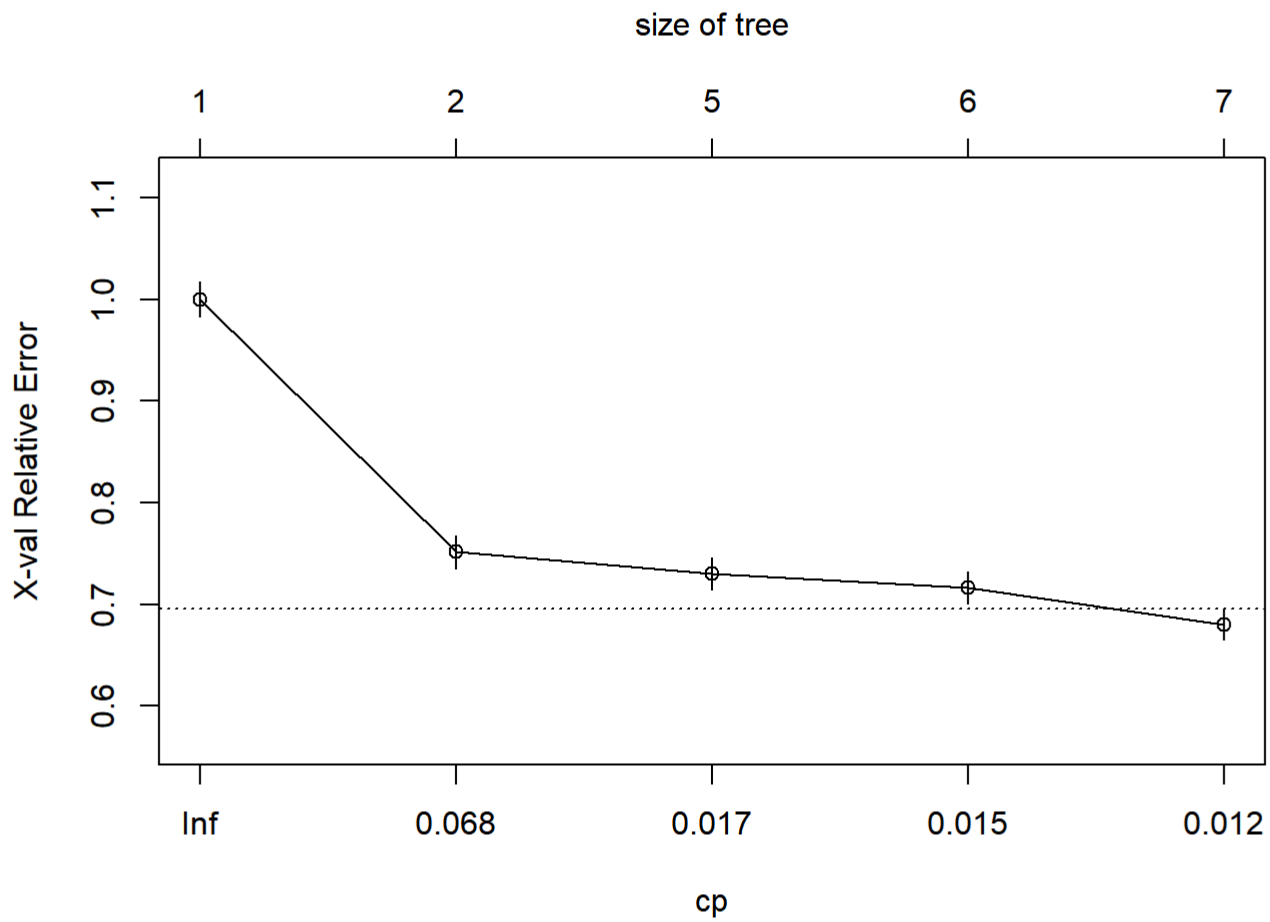
```
accTest <- sum(test$success==yTest)/nrow(test)

cbind(trainAcc=accTr, testAcc <- accTest)
```

```
##       trainAcc
## [1,] 0.7297404 0.7424873
```

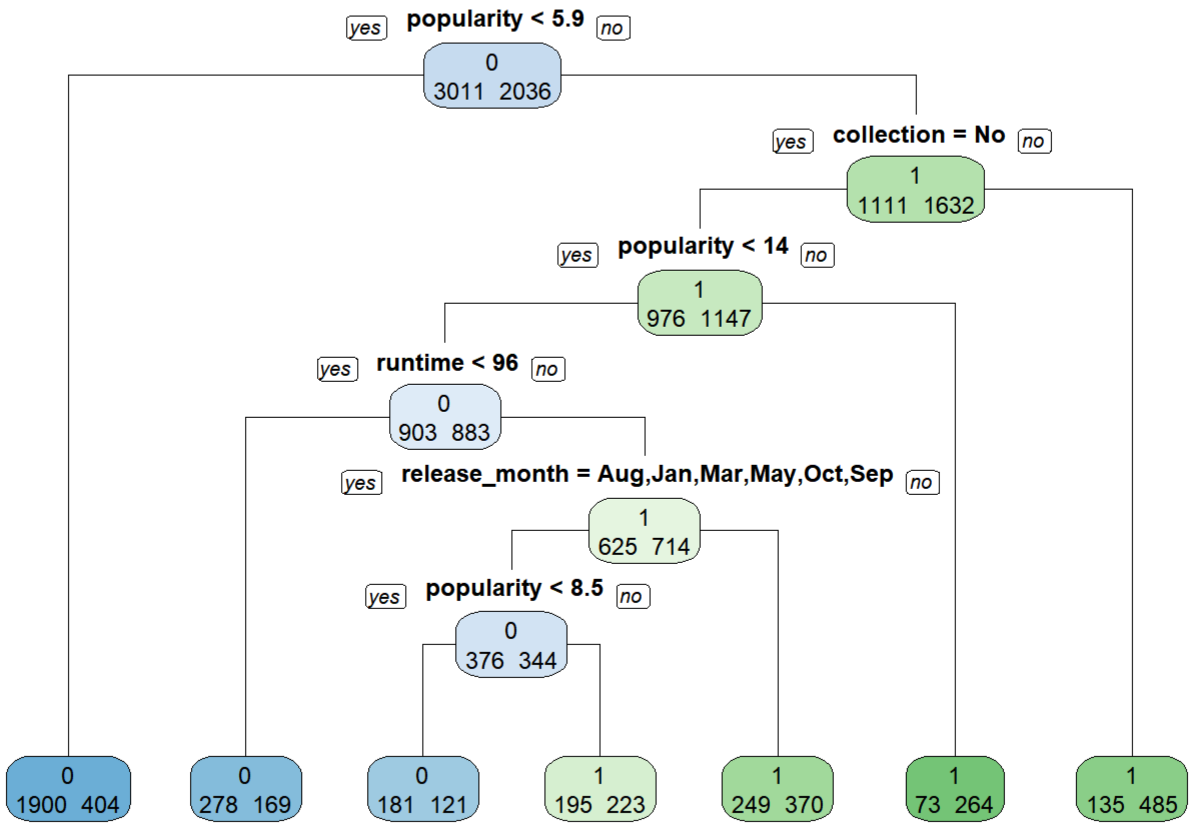
This model shows a stable fit based on the confusion matrix of text and train data and their respective accuracies, but sensitivity of our model is low. I can try doing tree classification to account for interaction between different predictors

Fitting a classification tree model

```
success_tree <- rpart(success~ actor_1_gender+ popularity+runtime+collection+num_prod_comp+num_p
rod_ctry+release_month, data = train, method = "class")

plotcp(success_tree)
```




```
# Retreive optimal cp value based on cross-validated error
opt_index <- which.min(success_tree$cptable[, "xerror"])
cp_opt <- success_tree$cptable[opt_index, "CP"]

success_tree_opt <- prune(tree = success_tree,
                          cp = cp_opt)

# Display the pruned tree results
rpart.plot(x = success_tree_opt, yesno = 2, type = 1, extra = 1)
```



Looking at the accuracy and confusion matrix from tree model (test vs. train)

```
yTr <- predict(success_tree_opt,train, type = "class")
yTest <- predict(success_tree_opt,test, type = "class")

#confusion matrix for train data
round(prop.table(table(actual = train$success, pred=yTr),1),2)
```

```
##        pred
## actual    0    1
##       0 0.78 0.22
##       1 0.34 0.66
```

```
accTr <- sum(train$success==yTr)/nrow(train)
#confusion matrix for test data
round(prop.table(table(actual = test$success, pred=yTest),1),2)
```

```
##        pred
## actual    0    1
##       0 0.79 0.21
##       1 0.33 0.67
```

```
accTest <- sum(test$success==yTest)/nrow(test)

cbind(trainAcc=accTr, testAcc <- accTest)
```

```
##       trainAcc
## [1,] 0.7333069 0.7387887
```

The fit of regression tree is similar to the logistic regression model. I will fit random forests to see if the accuracy can be improved further by bootstrapping:

```
train$success <- as.factor(train$success)
test$success <- as.factor(test$success)
train$release_month <- as.factor(train$release_month)
test$release_month <- as.factor(test$release_month)
train$collection <- as.factor(train$collection)
test$collection <- as.factor(test$collection)
train$actor_1_gender <- as.factor(train$actor_1_gender)
test$actor_1_gender <- as.factor(test$actor_1_gender)
success_rfor <- randomForest(success~ actor_1_gender+ popularity+runtime+collection+num_prod_com
p+num_prod_ctry+release_month, data= train, nodesize = 10)
```

```
success_rfor
```

```
##
## Call:
##  randomForest(formula = success ~ actor_1_gender + popularity +      runtime + collection + n
um_prod_comp + num_prod_ctry + release_month,      data = train, nodesize = 10)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 26.89%
## Confusion matrix:
##      0    1 class.error
## 0 2412  599   0.1989372
## 1  758 1278   0.3722986
```

```
yTest <- predict(success_rfor,test, type = "class")


#confusion matrix for test data
round(prop.table(table(actual = test$success, pred=yTest),1),2)
```

```
##       pred
## actual    0    1
##      0 0.80 0.20
##      1 0.34 0.66
```

```
accTest <- sum(test$success==yTest)/nrow(test); accTest
```

```
## [1] 0.7457235
```

I have defined movie success as earning 1.25 times the budget. Based on the prediction results above, random forest give the highest sensitivity although marginarrly higher than classification tree model.