Estrada-Rand_Filce_Hoskins_Final_Proj_CP2

Brayden Hoskins, Noah Estrada-Rand, Charlie Filce

November 21, 2019

```
#-----------------------------#
#       Library Calls         #
#-----------------------------#
library(forcats)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.2.1       v readr   1.3.1
## v tibble  2.1.1       v purrr   0.3.2
## v tidyr   0.8.3       v dplyr   0.8.0.1
## v ggplot2 3.2.1       v stringr 1.4.0
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(summarytools)
```

```
## Registered S3 method overwritten by 'pryr':
##   method      from
##   print.bytes Rcpp
##
## Attaching package: 'summarytools'
## The following object is masked from 'package:tibble':
##
##     view
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ggplot2)
library(glmnet)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
## The following object is masked from 'package:tidyr':
##
##     expand
## Loading required package: foreach
##
## Attaching package: 'foreach'
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loaded glmnet 2.0-18
```

```
library(glmnetUtils)
```

```
##
## Attaching package: 'glmnetUtils'
```

```
## The following objects are masked from 'package:glmnet':
##
##     cv.glmnet, glmnet
```

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## 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:ggplot2':
##
##     margin
```

```
library(randomForestExplainer)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
#-----------------------------------#
#       Importing Data Set          #
#-----------------------------------#
steam <- read.csv("steam.csv")

#-------------------------------------------#
#        Exploratory Data Analysis          #
#-------------------------------------------#
nrow(steam[!complete.cases(steam),])
```

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

```
# There are no rows with missing values

#-------------------------------------------#
#       Variable Engineering/Cleaning       #
#-------------------------------------------#
```

```
steam$price <- steam$price *1.28
# Turns the units into dollars
steam<- subset(steam, select = -c(appid,english,steamspy_tags,
                                  name,release_date,
                                  platforms,publisher,developer))
# Removing variables that are not useful
steam$genres <- do.call('rbind',strsplit(as.character(steam$genres), ';', fixed=TRUE))[,1]
```

```
## Warning in rbind("Action", "Action", "Action", "Action", "Action",
## "Action", : number of columns of result is not a multiple of vector length
## (arg 23)
```

```
steam$categories <- do.call('rbind',strsplit(as.character(steam$categories), ';', fixed=TRUE))[,1]
```

```
## Warning in rbind(c("Multi-player", "Online Multi-Player", "Local Multi-
## Player", : number of columns of result is not a multiple of vector length
## (arg 1)
```

```
# Getting rid of all the ';' delimited variables and instead assigned them a single value for that colu
steam$categories <- as.factor(steam$categories)
steam$genres <- as.factor(steam$genres)
# Making categories and genres as factored variables
steam <- steam[steam$average_playtime < 100000,]
steam <- steam[steam$price <50,]
steam <- steam[steam$average_playtime < 40000,]
steam <- steam[steam$negative_ratings < 2e+05,]
steam <- steam[steam$positive_ratings < 1e+06,]
# Removing outliers
fct_count(steam$categories)
```

```
## # A tibble: 23 x 2
##    f                             n
##    <fct>                     <int>
##  1 Captions available            5
##  2 Co-op                         9
##  3 Cross-Platform Multiplayer    3
##  4 Full controller support      18
##  5 In-App Purchases              6
##  6 Includes level editor        10
##  7 Includes Source SDK           1
##  8 Local Co-op                   5
##  9 Local Multi-Player           88
## 10 MMO                          44
## # ... with 13 more rows
```

```
#steam %>%
#  mutate(categories = fct_lump(categories, n =5))%>%
#  count(categories)
# Creating Groups of factored variables

#steam$simple_categories <- fct_lump(steam$categories,n = 4)
# Creating a factored feature with only 4 options
steam$categories <- ifelse(steam$categories == "Single-player","SinglePlayer",
                           ifelse(steam$categories == "Multi-player","Multi-Player",
                             ifelse(steam$categories == "Online Multi-Player","Multi-Player",
                               ifelse(steam$categories == "Local Multi-Player","Multi-Player",
```

```
                                    ifelse(steam$categories == "MMO","MMO",
                                      ifelse(steam$categories == "Co-op","Co-op",
                                        ifelse(steam$categories == "Shared/Split Screen","Co-op",
                                          ifelse(steam$categories == "Local Co-op","Co-op",
                                            ifelse(steam$categories == "Online Co-op","Co-op",
                                              ifelse(steam$categories == "Steam Cloud","Steam",
                                                ifelse(steam$categories == "Steam Trading Cards","Stea
                                                  ifelse(steam$categories == "Steam Leaderboards","St
                                                    ifelse(steam$categories == "Steam Achievements","S
steam$categories <- as.factor(steam$categories)
unique(steam$categories)
```

```
## [1] Multi-Player SinglePlayer Other        Steam        MMO
## [6] Co-op
## Levels: Co-op MMO Multi-Player Other SinglePlayer Steam
```

```
# Another way of lumping together factors
steam$genres <- fct_lump(steam$genres,n = 5)
unique(steam$genres)
```

```
## [1] Action    Indie     Other     Casual    Adventure Violent
## Levels: Action Adventure Casual Indie Violent Other
```

```
# Reducing the number of genres

#---------------------------------------------#
#          Creating Predictor Variable        #
#---------------------------------------------#
steam$successfulGame <- ifelse(steam$owners == "10000000-20000000",1,
                               ifelse(steam$owners == "20000000-50000000",1,
                                      ifelse(steam$owners == "50000000-100000000",1,
                                             ifelse(steam$owners == "100000000-200000000",1,
                                                    ifelse(steam$owners == "5000000-10000000",1,
                                                           ifelse(steam$owners == "2000000-5000000",1,
                                                                  ifelse(steam$owners == "1000000-20000
steam$successfulGame <- as.factor(steam$successfulGame)
steam <- subset(steam, select = -c(owners))
# Creating variable successful game, 1 = Successful with over 1 million games sold, 0 for everything le
#-------------------------------------#
#       Descriptive Statistics        #
#-------------------------------------#
descr(steam)
```

```
## Non-numerical variable(s) ignored: categories, genres, successfulGame
```

```
## Descriptive Statistics
## steam
## N: 26876
##
##                     achievements   average_playtime   median_playtime   negative_ratings
## ----------------- -------------- ------------------ ------------------ ------------------
##              Mean          45.31             117.46             105.56             162.85
##           Std.Dev         353.96             827.80             769.94            1796.90
##               Min           0.00               0.00               0.00               0.00
##                Q1           0.00               0.00               0.00               2.00
##            Median           7.00               0.00               0.00               9.00
```

```
##                Q3          23.00              0.00              0.00             40.00
##               Max        9821.00          38805.00          38805.00         142079.00
##               MAD          10.38              0.00              0.00             11.86
##               IQR          23.00              0.00              0.00             38.00
##                CV           7.81              7.05              7.29             11.03
##          Skewness          13.38             22.16             28.47             50.17
##       SE.Skewness           0.01              0.01              0.01              0.01
##          Kurtosis         189.74            676.18           1072.34           3325.75
##           N.Valid       26876.00          26876.00          26876.00          26876.00
##         Pct.Valid         100.00            100.00            100.00            100.00
##
## Table: Table continues below
##
##
##
##                     positive_ratings      price   required_age
## ----------------- ------------------ ---------- --------------
##              Mean             848.56       7.33           0.34
##           Std.Dev            9594.90       7.34           2.35
##               Min               0.00       0.00           0.00
##                Q1               6.00       2.16           0.00
##            Median              24.00       5.11           0.00
##                Q3             122.00       9.20           0.00
##               Max          863507.00      49.91          18.00
##               MAD              32.62       5.69           0.00
##               IQR             116.00       7.04           0.00
##                CV              11.31       1.00           6.93
##          Skewness              45.17       1.89           6.90
##       SE.Skewness               0.01       0.01           0.01
##          Kurtosis            3071.79       4.66          46.24
##           N.Valid           26876.00   26876.00       26876.00
##         Pct.Valid             100.00     100.00         100.00
```

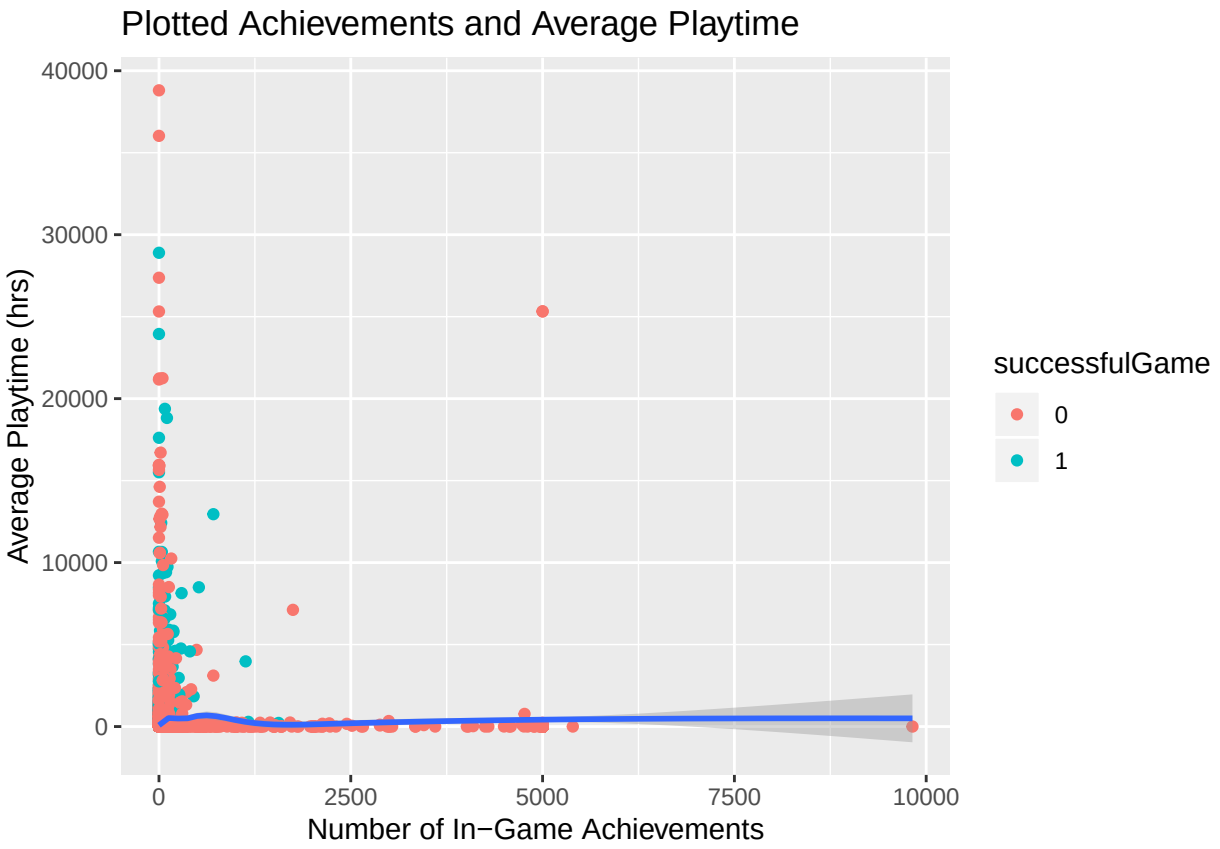
```
str(steam)
```

```
## 'data.frame':    26876 obs. of  10 variables:
##  $ required_age    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ categories      : Factor w/ 6 levels "Co-op","MMO",..: 3 3 3 3 5 3 5 5 5 5 ...
##  $ genres          : Factor w/ 6 levels "Action","Adventure",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ achievements    : int  0 0 0 0 0 0 0 0 0 33 ...
##  $ positive_ratings: int  124534 3318 3416 1273 5250 2758 27755 12120 3822 67902 ...
##  $ negative_ratings: int  3339 633 398 267 288 684 1100 1439 420 2419 ...
##  $ average_playtime: int  17612 277 187 258 624 175 1300 427 361 691 ...
##  $ median_playtime : int  317 62 34 184 415 10 83 43 205 402 ...
##  $ price           : num  9.2 5.11 5.11 5.11 5.11 ...
##  $ successfulGame  : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

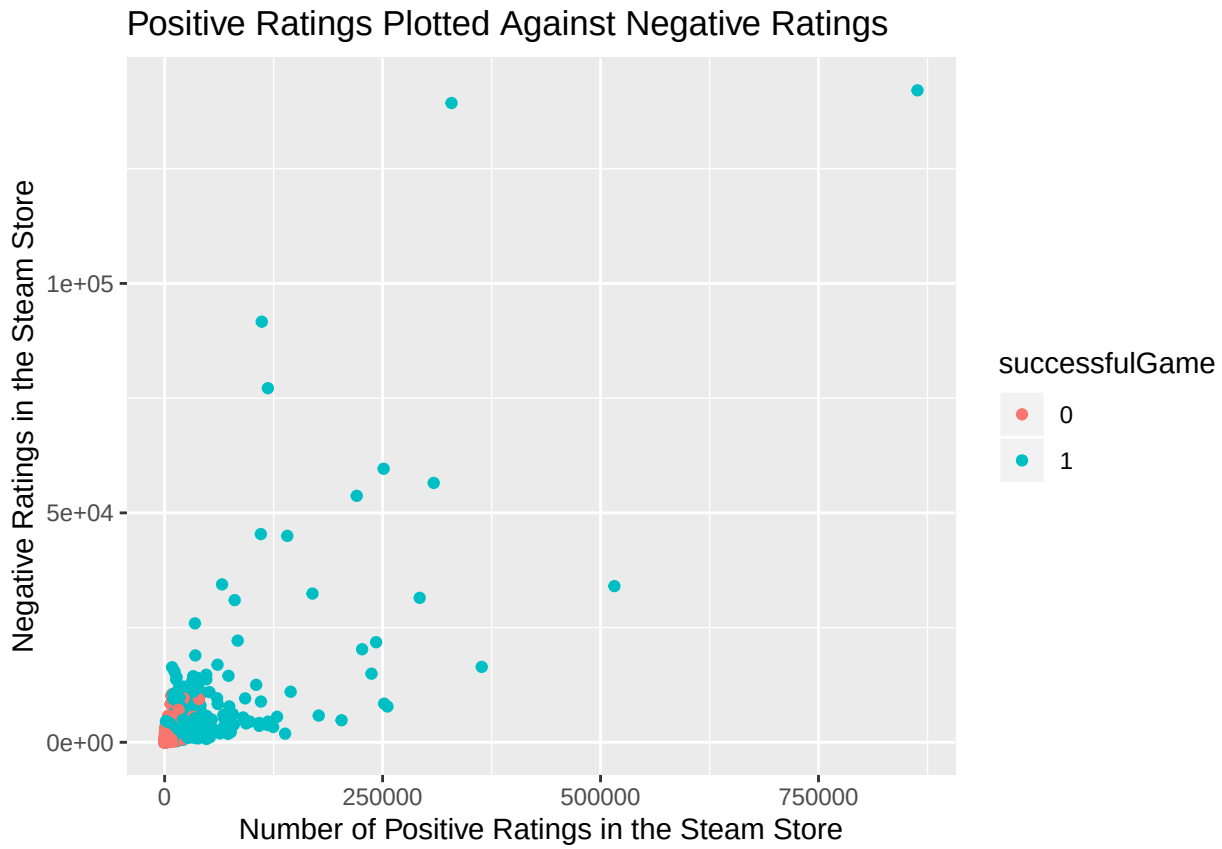
```
#---------------------------------------------#
#                Correlations                 #
#---------------------------------------------#
numeric_cols <- sapply(steam,is.numeric)
correlations <- cor(steam[,numeric_cols])
corrplot(correlations)
```


```
#---------------------------------------------#
#                 Summary Plots               #
#---------------------------------------------#
ggplot(steam,aes(x = achievements,y = average_playtime)) + geom_point(aes(color = successfulGame))+
  geom_smooth() + labs(x = "Number of In-Game Achievements",y = "Average Playtime (hrs)",
                       title = "Plotted Achievements and Average Playtime")
```

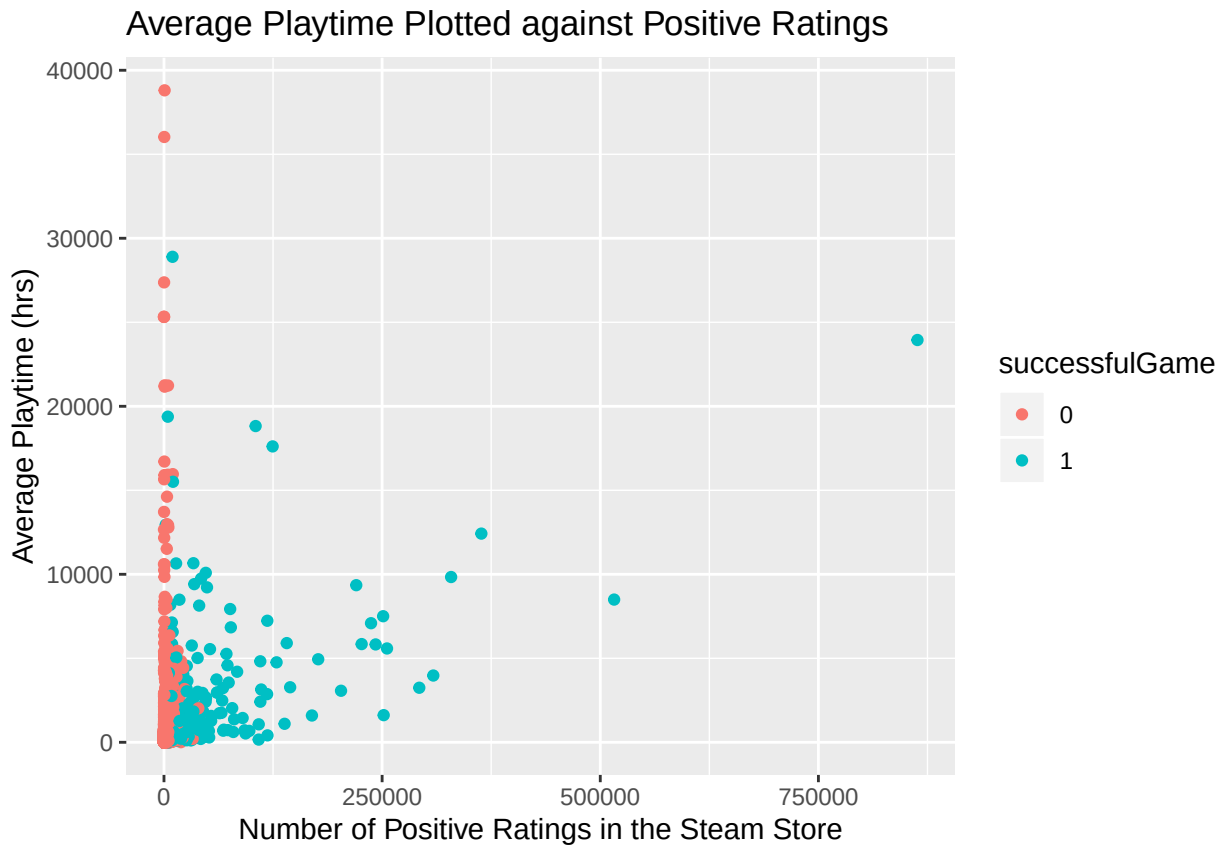
```
## `geom_smooth()` using method = 'gam' and formula 'y ~ s(x, bs = "cs")'
```



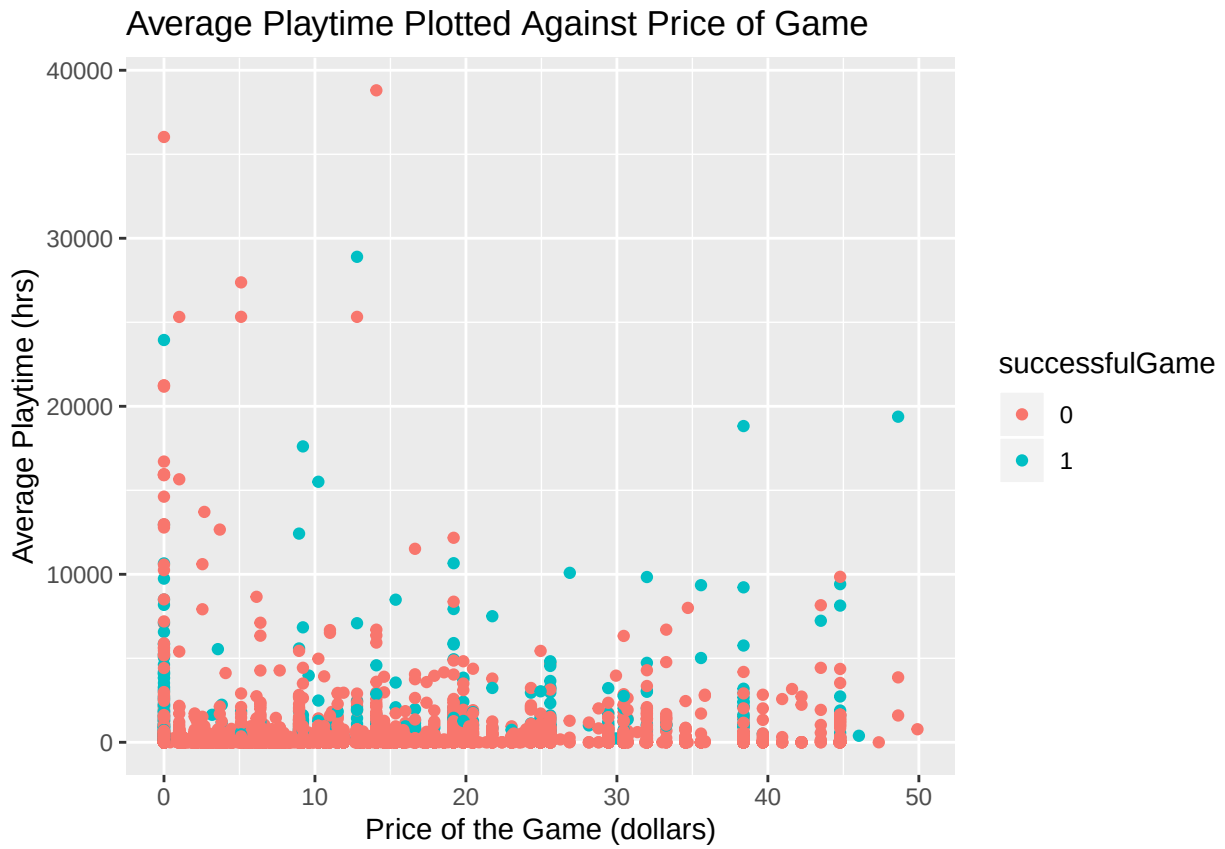

```
ggplot(steam,aes(x = positive_ratings,y = negative_ratings)) +
  geom_point(aes(color = successfulGame))+
  labs(x = "Number of Positive Ratings in the Steam Store",y = "Negative Ratings in the Steam Store",
       title = "Positive Ratings Plotted Against Negative Ratings")
```



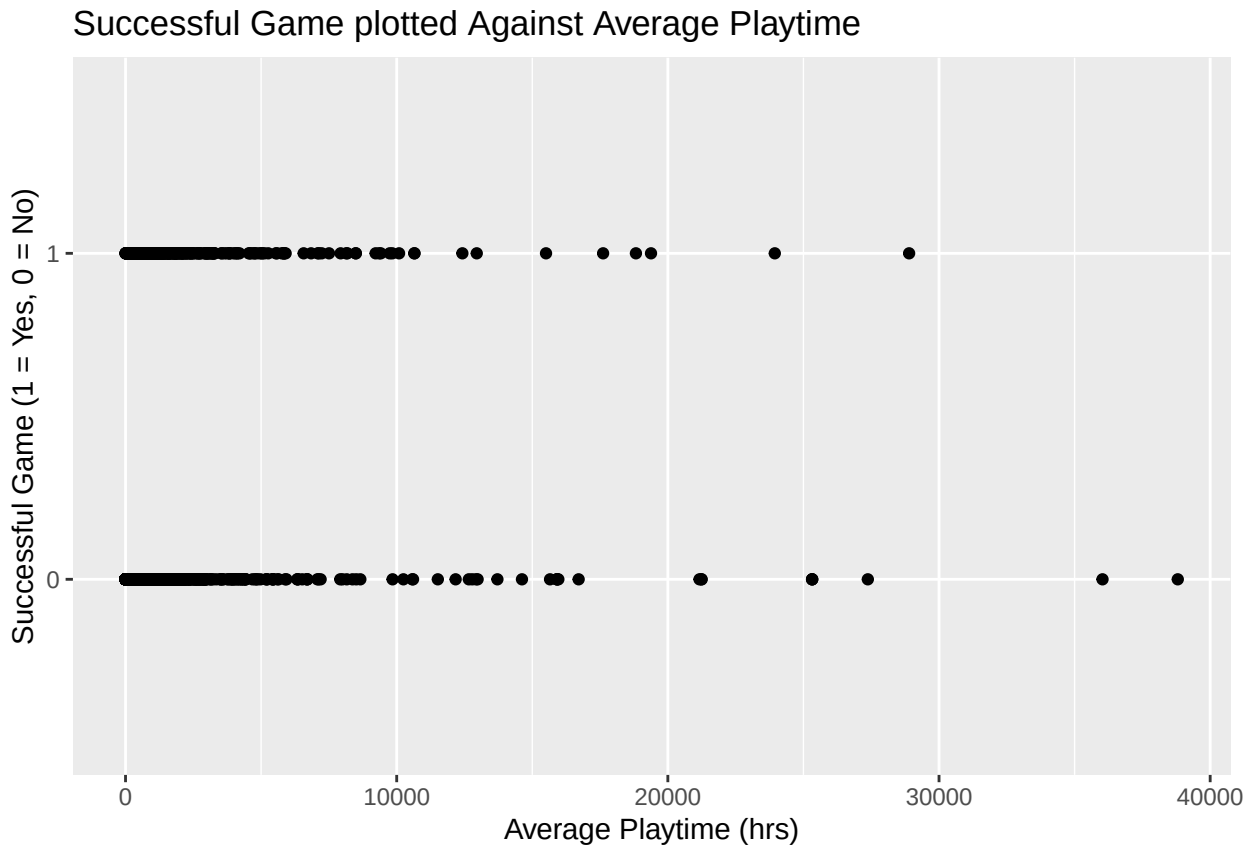

```
ggplot(steam,aes(x = positive_ratings,y = average_playtime)) +geom_point(aes(color = successfulGame))+
  labs(x = "Number of Positive Ratings in the Steam Store",y = "Average Playtime (hrs)",
       title = "Average Playtime Plotted against Positive Ratings")
```



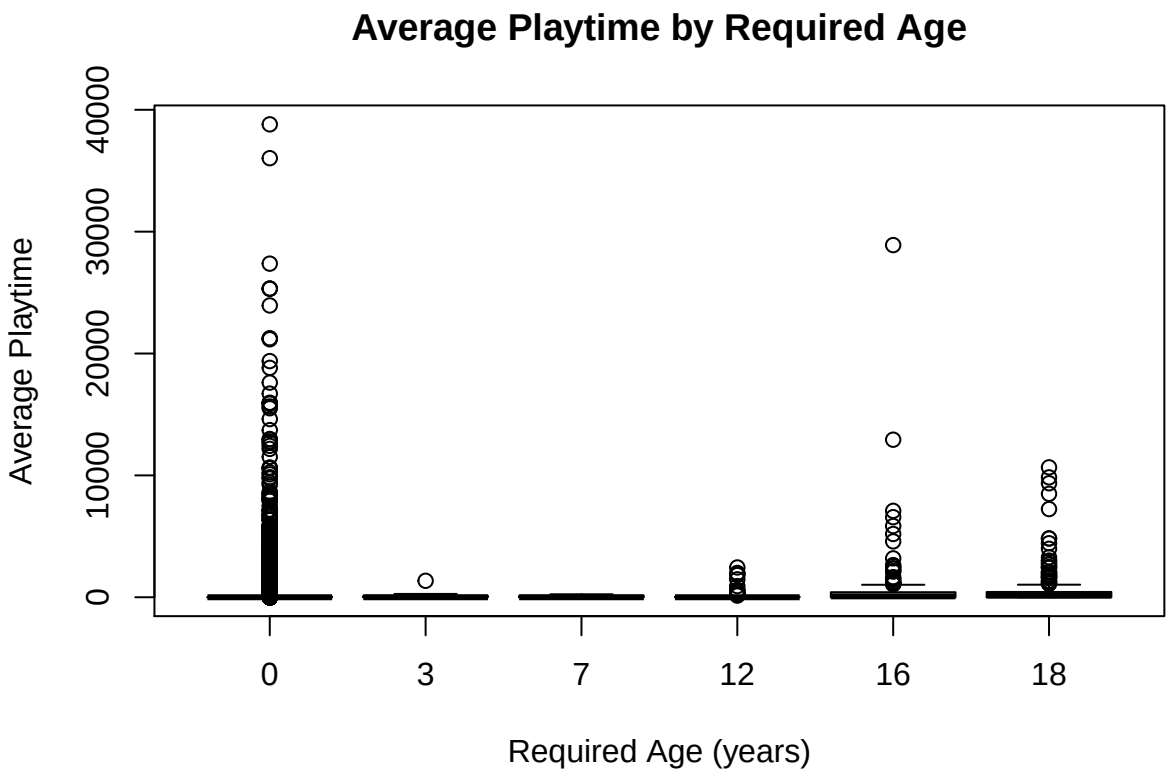

```
ggplot(steam,aes(x = price,y = average_playtime)) +
  geom_point(aes(color = successfulGame)) +
  labs(x = "Price of the Game (dollars)",y ="Average Playtime (hrs)",
       title = "Average Playtime Plotted Against Price of Game")
```



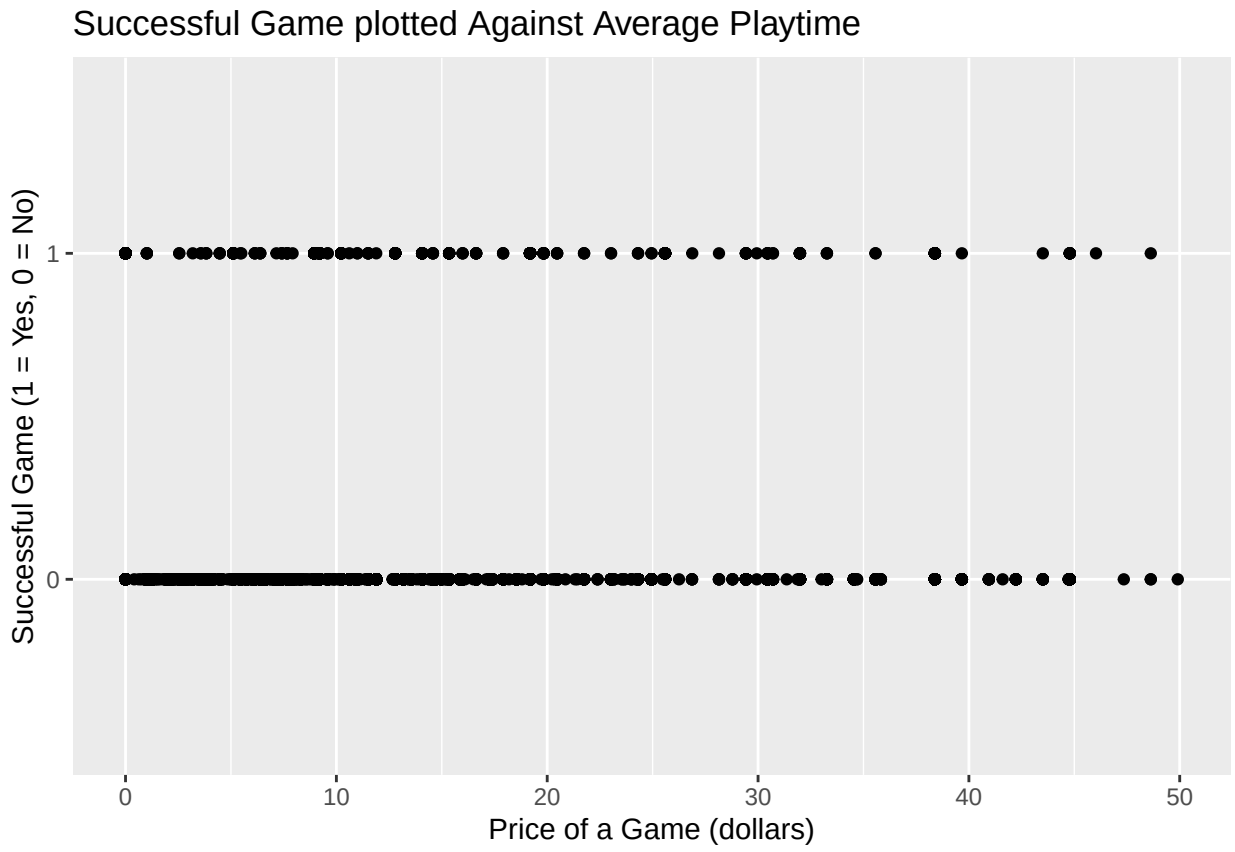

```
ggplot(steam,aes(x = average_playtime,y = successfulGame)) +
  geom_point()+
  labs(title = "Successful Game plotted Against Average Playtime",
       x = "Average Playtime (hrs)",y = "Successful Game (1 = Yes, 0 = No)")
```



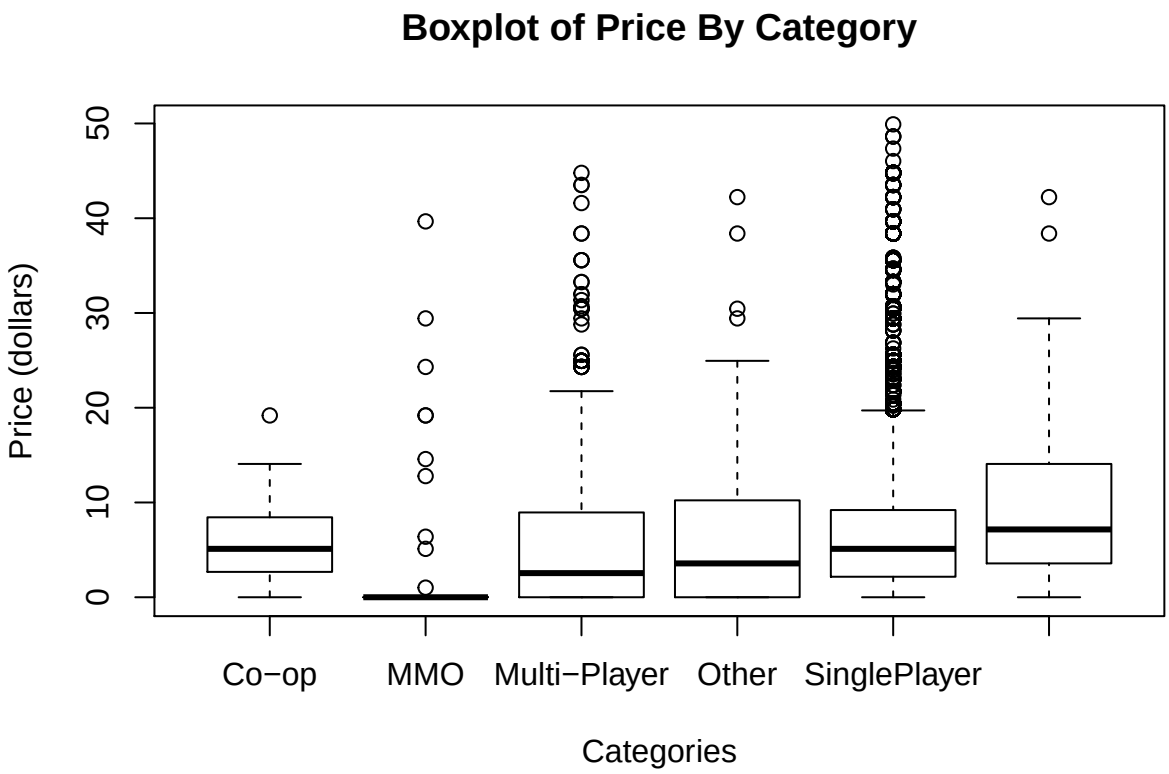

```
boxplot(steam$average_playtime~steam$required_age,
        main = "Average Playtime by Required Age",xlab = "Required Age (years)",
        ylab = "Average Playtime")
```



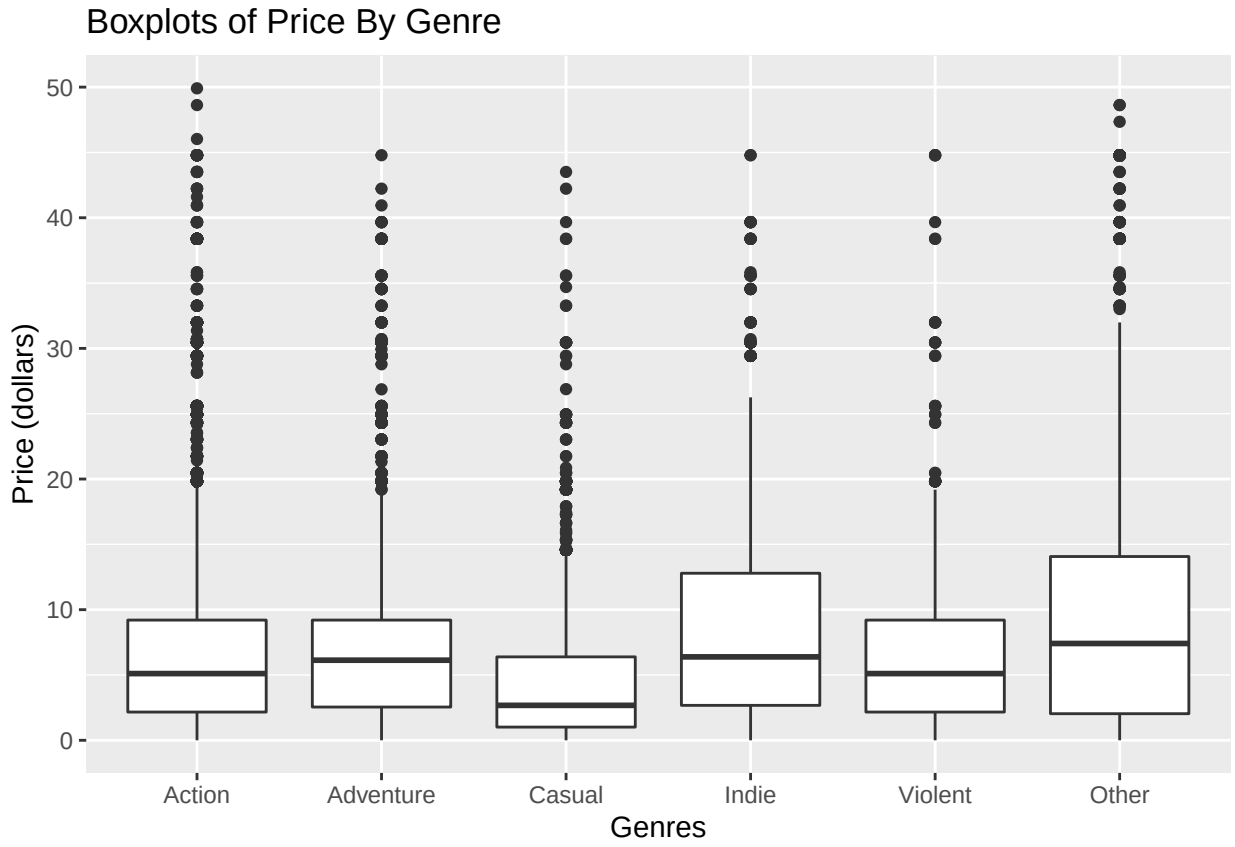

```
ggplot(steam,aes(x = price,y = successfulGame)) +
  geom_point()+
  labs(title = "Successful Game plotted Against Average Playtime",
       x = "Price of a Game (dollars)",y = "Successful Game (1 = Yes, 0 = No)")
```




```
boxplot(steam$price~steam$categories,
        main = "Boxplot of Price By Category",
        xlab = "Categories",
        ylab = "Price (dollars)")
```




```
ggplot(steam,aes(x = genres,y = price)) + geom_boxplot() +
  labs(title = "Boxplots of Price By Genre",x = "Genres",y = "Price (dollars)")
```

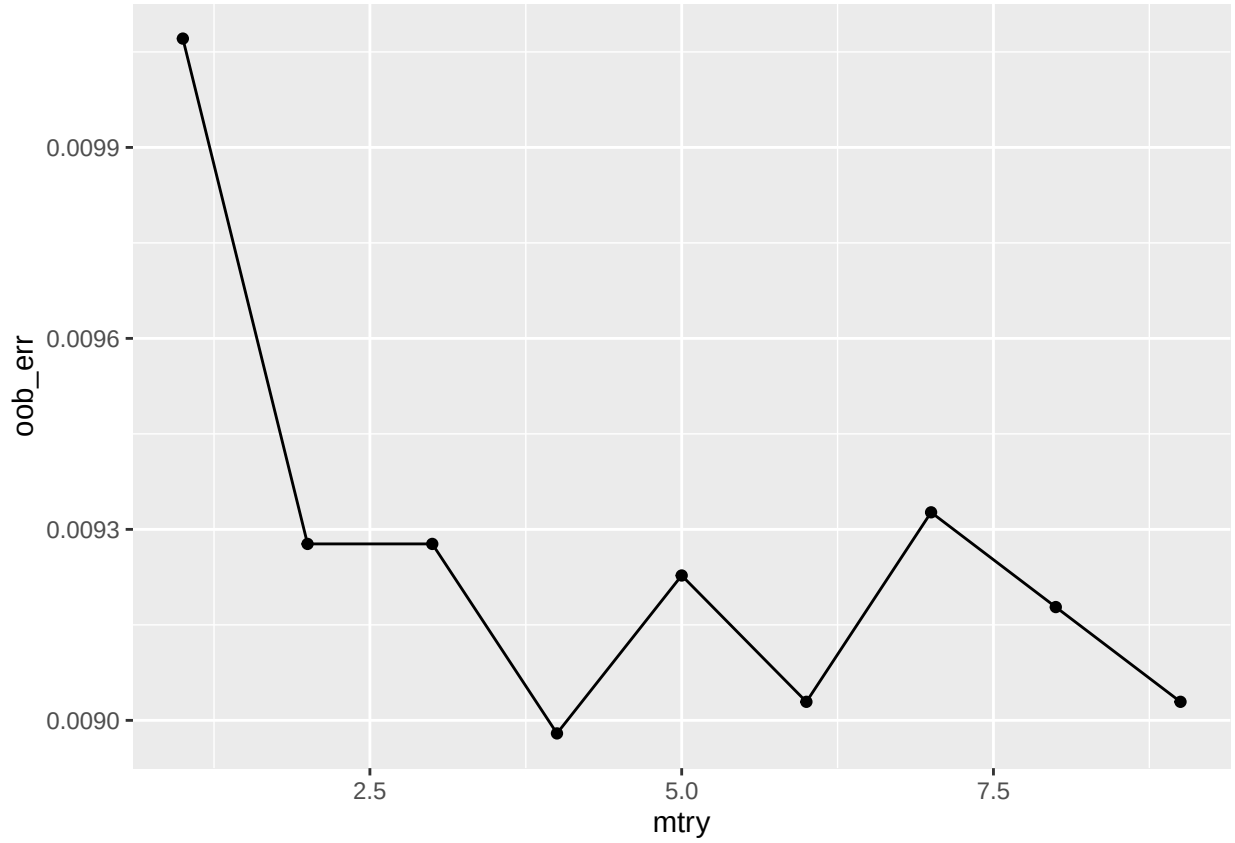



```
summary(steam)
```

```
##   required_age           categories          genres
##  Min.   : 0.0000   Co-op       :   31   Action   :11129
##  1st Qu.: 0.0000   MMO         :   44   Adventure: 5245
##  Median : 0.0000   Multi-Player: 1128   Casual   : 4368
##  Mean   : 0.3395   Other       :  102   Indie    : 2619
##  3rd Qu.: 0.0000   SinglePlayer:25505   Violent  :  706
##  Max.   :18.0000   Steam       :   66   Other    : 2809
##   achievements     positive_ratings   negative_ratings   average_playtime
##  Min.   :   0.00   Min.   :     0.0   Min.   :     0.0   Min.   :    0.0
##  1st Qu.:   0.00   1st Qu.:     6.0   1st Qu.:     2.0   1st Qu.:    0.0
##  Median :   7.00   Median :    24.0   Median :     9.0   Median :    0.0
##  Mean   :  45.31   Mean   :   848.6   Mean   :   162.8   Mean   :  117.5
##  3rd Qu.:  23.00   3rd Qu.:   122.0   3rd Qu.:    40.0   3rd Qu.:    0.0
##  Max.   :9821.00   Max.   :863507.0   Max.   :142079.0   Max.   :38805.0
##  median_playtime       price        successfulGame
##  Min.   :    0.0   Min.   : 0.000   0:26344
##  1st Qu.:    0.0   1st Qu.: 2.163   1:  532
##  Median :    0.0   Median : 5.107
##  Mean   :  105.6   Mean   : 7.332
##  3rd Qu.:    0.0   3rd Qu.: 9.203
##  Max.   :38805.0   Max.   :49.907
```

```
#-----------------------------------#
#         Random Forest Model       #
#-----------------------------------#
```

```r
set.seed(2019)
train_idx <- sample(1:nrow(steam), size = floor(.75 * nrow(steam)))
steam_train <- steam[train_idx,]
steam_test <- steam[-train_idx,]
# Creating Train and Test sets
rf_mods <- list()
oob_err <- NULL
# Used to store out of bag error
test_err <- NULL
for(mtry in 1:9){
  rf_fit <- randomForest(successfulGame~.,
                         data = steam_train,
                         mtry = mtry,
                         ntrees = 500,
                         type = classification)
  oob_err[mtry] <- rf_fit$err.rate[500]
}
results_df <- data.frame(mtry = 1:9,
                         oob_err)
ggplot(results_df,aes(x = mtry,y = oob_err)) + geom_point() +geom_line()
```

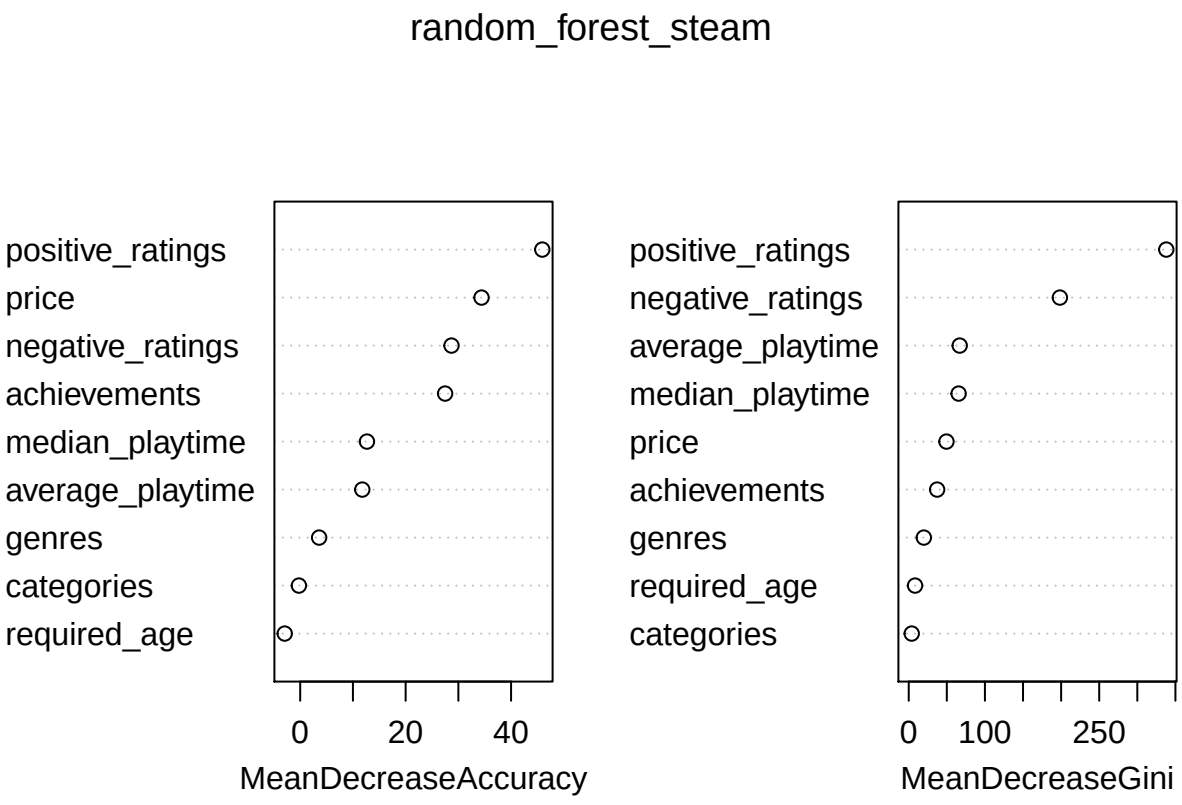



```r
# This helps us to find the best mtry for our Random Forest model
random_forest_steam <- randomForest(successfulGame~.,
                                    data = steam_train,
                                    mtry = 4,
                                    ntrees = 500,
```

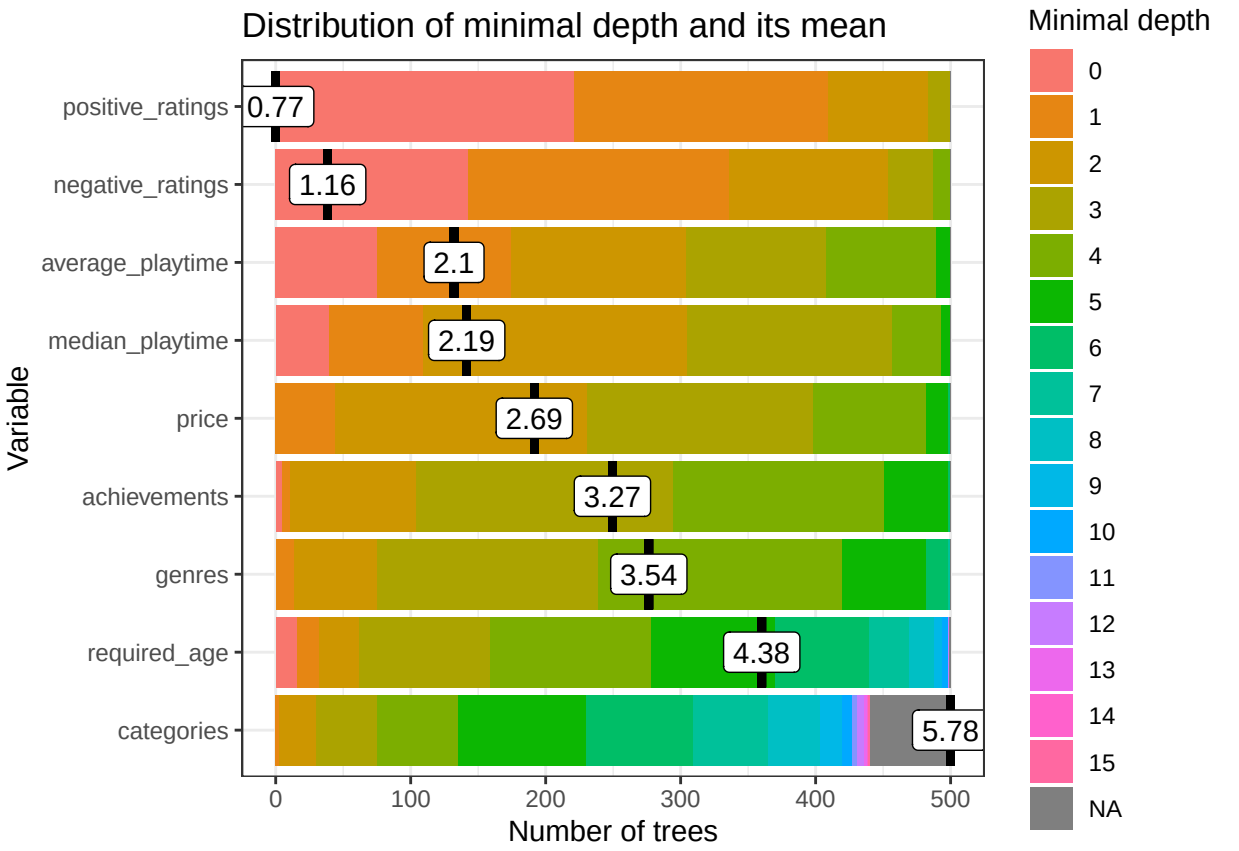
```
                                    type = classification,
                                    importance = TRUE)
# Creating a model with the best m = 5
rf_preds <- predict(random_forest_steam, newdata = steam_test)
importance(random_forest_steam)
```

```
##                          0         1 MeanDecreaseAccuracy MeanDecreaseGini
## required_age     -5.019845  1.405620           -2.9342421         8.311057
## categories       -2.780317  2.403641           -0.2448353         3.922285
## genres           -5.699167 12.956953            3.5897419        19.819459
## achievements     24.574499 12.354963           27.4878011        37.264411
## positive_ratings 15.577401 93.949662           45.9223634       338.029895
## negative_ratings  4.405298 32.201864           28.7000353       198.382708
## average_playtime 10.383053 25.520331           11.7737448        66.909095
## median_playtime  12.703329 -4.897593           12.6614801        65.502754
## price            25.258557 16.588505           34.3978205        49.542447
```

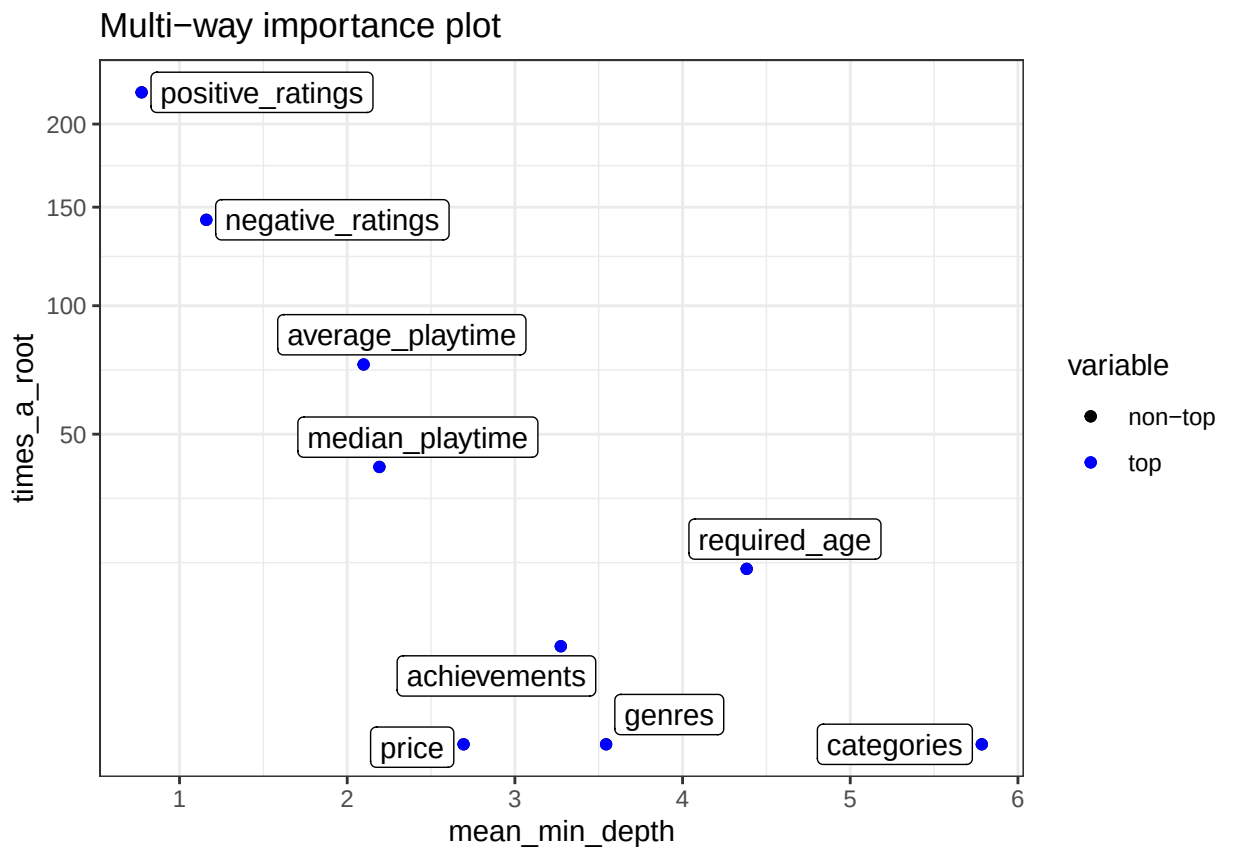
```
varImpPlot(random_forest_steam)
```




```
plot_min_depth_distribution(random_forest_steam)
```




```
plot_multi_way_importance(random_forest_steam)
```




```
# Plotting the results of the random forest model
#explain_forest(random_forest_steam)
# Explaining the random forest model

#----------------------------------------#
#       Dimensionality Reduction         #
#----------------------------------------#
steam_lasso <- cv.glmnet(successfulGame~.,
                         data = steam,
                         alpha = 1,
                         family = "binomial")
```

```
## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
```

```
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned

## Warning: from glmnet Fortran code (error code -2); Convergence for 2th
## lambda value not reached after maxit=100000 iterations; solutions for
## larger lambdas returned
```

```
# The lasso model would take very long before we reduced the dimensions of the categories and genres va
print(steam_lasso)
```

```
## Call:
## cv.glmnet.formula(formula = successfulGame ~ ., data = steam,
##     alpha = 1, family = "binomial")
##
## Model fitting options:
##     Sparse model matrix: FALSE
##     Use model.frame: FALSE
##     Number of crossvalidation folds: 10
##     Alpha: 1
##     Deviance-minimizing lambda: 9.9e+35  (+1 SE): 9.9e+35
```

```
#----------------------------------------#
#       K-fold Cross Validation          #
#----------------------------------------#
# creating folds (using createFolds from caret package)
steam$folds <- createFolds(steam$successfulGame, k = 10, list = FALSE)
steam$row_num <- 1:nrow(steam)


### K-Fold Cross Validation
nfolds <- 10
preds_KFold_CV_DF <- data.frame(folds = steam$folds,rownum = steam$row_num,preds_KFold_CV = rep(NA,nrow
```

```r
# for loop for K-Fold CV
for(i in 1:nfolds){
  mod <- lm(successfulGame ~ .,
            data = steam[steam$folds != i,])
  preds <- predict(mod,
                   newdata = steam[steam$folds == i,])
  preds_KFold_CV_DF[preds_KFold_CV_DF$folds == i,"preds_KFold_CV"]  <- preds
}
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors

## Warning in model.response(mf, "numeric"): using type = "numeric" with a
## factor response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```r
preds_DF <- data.frame(preds_KFold_CV = preds_KFold_CV_DF$preds_KFold_CV,true = steam$successfulGame)

RMSE <- function(t, p) {
  sqrt(sum(((t - p)^2)) * (1/length(t)))
```

```
}
RMSE(preds_DF$preds_KFold_CV,as.numeric(preds_DF$true))
```

```
## [1] 0.1257668
```