PERFORMANCE ANALYSIS OF MOVIE RATING

SUMMER PROJECT REPORT

Submitted in partial fulfilment for the award of the degree of

MS

in

Software Engineering

by

POTUREDDI GOWTHAM RAJEEV SWAROOP (14MSE0070)

ABBAVARAM PAVANI (14MSE0085)

Under the guidance of

Prof. LAWANYA SHRI M

SITE

School of Information Technology and Engineering

July 2018

VIT UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
VELLORE ■ CHENNAI
www.vit.ac.in

School Of Information Technology and Engineering

DECLARATION BY THE CANDIDATE

We hereby declare that the summer project report entitled **"PERFORMANCE ANALYSIS OF MOVIE RATING"** submitted by us to VIT University ,Vellore, in partial fulfillment of the requirement for the award of the degree of **MS (Software Engineering)** is a record of bonafide project work carried out by me under the supervision of **Prof. LAWANYA SHRI M , Assistant Professor(Senior)** .We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree in this institute or any other institute or university.

Place: Vellore

Date: **Signature of the Candidate(s)**

School of Information Technology and Engineering

BONAFIDE CERTIFICATE

This is to certify that the project work entitled " **PERFORMANCE ANALYSIS OF MOVIE RATING"** by **POTUREDDI GOWTHAM (14MSE0070),ABBAVARAM PAVANI (14MSE0085)** to VIT University, Vellore, in partial fulfillment of the requirement for the award of the degree of **MS(Software Engineering)** is a project bonafide work carried out by them under my supervision. The project fulfills the requirements as per the regulations of this Institute and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this Institute or any other Institute or University.

Prof. LAWANYA SHRI M
Supervisor
Assistant Professor(Senior)

Internal Examiner(1) **InternalExaminer(2)**

ACKNOWLEDGEMENT

I wish to express our heartfelt gratitude to **Dr.G. Viswanathan**, Chancellor,VIT University, Vellore for providing facilities for the summer semester project.I am highly grateful to our Vice Presidents. **Shri. Shankar Viswanathan, Shri.Sekar Viswanathan, Shri G.V.Selvam**, and **Dr. Anand A. Samuel,** Vice chancellor, **Dr.S.Narayanan, Pro-Vice Chancellor** for providing necessary resources.

My sincere gratitude to **Dr. Aswani Kumar Cherukuri**,, Dean, School of Information Technology and Engineering, for giving us the opportunity to undertake the project.

I wish to express my sincere gratitude to **Dr. S.Sree Dharinya**, Head of the Department, **Prof. MAGESH G** and **Prof. JAYA LAKSHMI P**, Summer Project Coordinators, MS(Software Engineering),Department of Software and Systems Engineering, School of information Technology and Engineering for providing me an opportunity to do my project work in the ***VIT University.***

I would like to express my special gratitude and thanks to my internal guide **Prof. LAWANYA SHRI.M, Assistant Professor(Senior).** School of information Technology and Engineering whose esteemed guidance and immense support encouraged to complete the project successfully.

I thank the Management of VIT University for permitting me to use the library resources. I also thank all the faculty members of VIT University for giving me the courage and strength I needed to complete my goals. This acknowledgement would be incomplete without expressing my whole hearted thanks to my family and friends who motivated me during the course of the work.

Place: Vellore

Date:

POTUREDDI GOWTHAM RAJEEV SWAROOP

ABBAVRAM PAVANI

TABLE OF CONTENTS

ABSTRACT

Data mining has been popularly used in many areas such as marketing, customer relationships, banking, finance, statistics, inventory forecasting, bioinformatics, healthcare etc. The techniques have also been used for many recommender systems in many aspects like classification, clustering, predictions, and performance analysis. It can be combined with many existing areas in the computer and information technology fields such as embedded system design etc. The technique of data mining requires lots of data to be investigated. The data attributes must be studied in details to create an accurate model. The measurement of the accuracy of models can be F-measure, Precision, Recall, True positive/True negative/False positive/False negative etc. In this work, we are interested in the movie rating classification approaches.

Movie rating is one of the most unpredictable thing because different viewers have different opinions. In the old system, very less accurate ratings were predicted which affected the opinions of other users who viewed it. Different reviewers rates the movie based on the genre ,actors ,directors which leads to positive or negative impact on the movie.There are different classification approaches available for the prediction,but there is no true evidence which algorithm suits best for our system. So we apply the performance analysis to get the right accuracies.

We in our system demonstrate the analysis of Internet Movie DataBase(IMDB). In this modern world where the ratings of movie are being dependent on various factors namely content_rating, title,genre, language etc. It becomes difficult for an user to know on which factor it is being mainly dependent. Even though there are many classification algorithms that are being used to know the ratings, the user couldn't know which one is the best to apply in real world. To avoid this problem , the performance analysis of those algorithms is done. It also contains the information about the techniques used and their usefulness. Between those all algorithms the performance is analyzed and compared.

The algorithms are implemented in the 'R-studio' environment using the 'R – Programing' language.We have implemented our system in the OS X environment having 2.5 GHZ Intel Core I5 using an Apple Extended Keyboard.

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACRONYM	EXPANSION
IMDB	Internet Movie DataBase
INEQ	Inequality Measurement
PARTY	Recursive partioning
TREE	Classification and regression tree
GINI	Gini coefficient
ENTROPY	Theils entropy coefficient
LC	Lorenz curve
SVM	Support vector machine

1. INTRODUCTION

1.1 PROBLEM STATEMENT:

Movie rating is one of the most unpredictable thing because different viewers have different opinions. In the old system, very less accurate ratings were predicted which affected the opinions of other users who viewed it. The ratings of movie are being dependent on various factors namely content_rating, title, genre, language etc.. (It becomes difficult for an user to know on which factor it is being mainly dependent. Even though there are many classification algorithms that are being used to know the ratings, the user couldn't know which one is the best to apply in real world. To avoid this problem , the performance analysis of those algorithms is done. It also contains the information about the techniques used and their usefulness. Between those all algorithms the performance is analyzed and compared.

MOTIVATION:

To predict the movie ratings with high accuracy using very efficient algorithms like decision tree ,regression , support vector machine,CART etc..

1.1.1 Existing System

In the existing system the prediction of movie rating is done using different algorithms. For the same system different algorithms are applied and this leads to the confusion in the user to predict the ratings. The user cannot predict which algorithm will give the best accurate ratings. This is the major disadvantage in the existing system. It is difficult for the user to predict on which factor the outcome depends.

1.1.2 Disadvantages of Existing system

- Leads to confusion for the user to predict the outcome.
- Increase in complexity.

1.2 OBJECTIVE:

1.2.1 PROPOSED SYSTEM:

We in our system demonstrate the analysis of Internet Movie DataBase(IMDB). In this modern world where the ratings of movie are being dependent on various factors namely content_rating, title,genre, language etc. It becomes difficult for an user to know on which factor it is being mainly dependent. Even though there are many classification algorithms that are being used to know the ratings, the user couldn't know which one is the best to apply in real world. To avoid this problem , the performance analysis of those algorithms is done. It also contains the information about the techniques used and their usefulness. Between those all algorithms the performance is analyzed and compared.

1.2.2 ADVANTAGES OF PROPOSED SYSTEM

- Increase in accuracy
- Increase in scalability
- Decrease in complexity

2. LITERATURE SURVEY

2.1 Literature

SNO	PAPER TITLE	JOURNAL NAME AND YEAR	METHEDOLOGY USED
1.	Sentiment analyis of Indian movie review with various feature selection techniques	2016 IEEE International Conference on Advances in Computer Applications (ICACA), Coimbatore, 2016, pp. 181-185	1. A proper analysis and study is done on different approaches that are affecting the sentiment score of a movie review which is having impact on the rating. 2. Classification algorithms are used for the evaluation of performance and accuracy of the approach. Doing this ,they performed random forest technique also which gave highest accurate result. They also found some features that determine the polarity of reviews.
2.	Predicting Movie Success Based on IMDB Data	International Journal of Data Mining Techniques and Applications ,June 2014.	1. In the past when they used other techniques none of them has succeeded in advising about the model that can be used to give high polarity level on movie rating. Here they used the IMDB data set to predict it. They performed logistic regression and SVM regression on this IMDB data to predict the rating 2. They had found that by using linear regression it is 51% accurate. Also they got the error tolerance or SVM and Linear techniques is 20%.

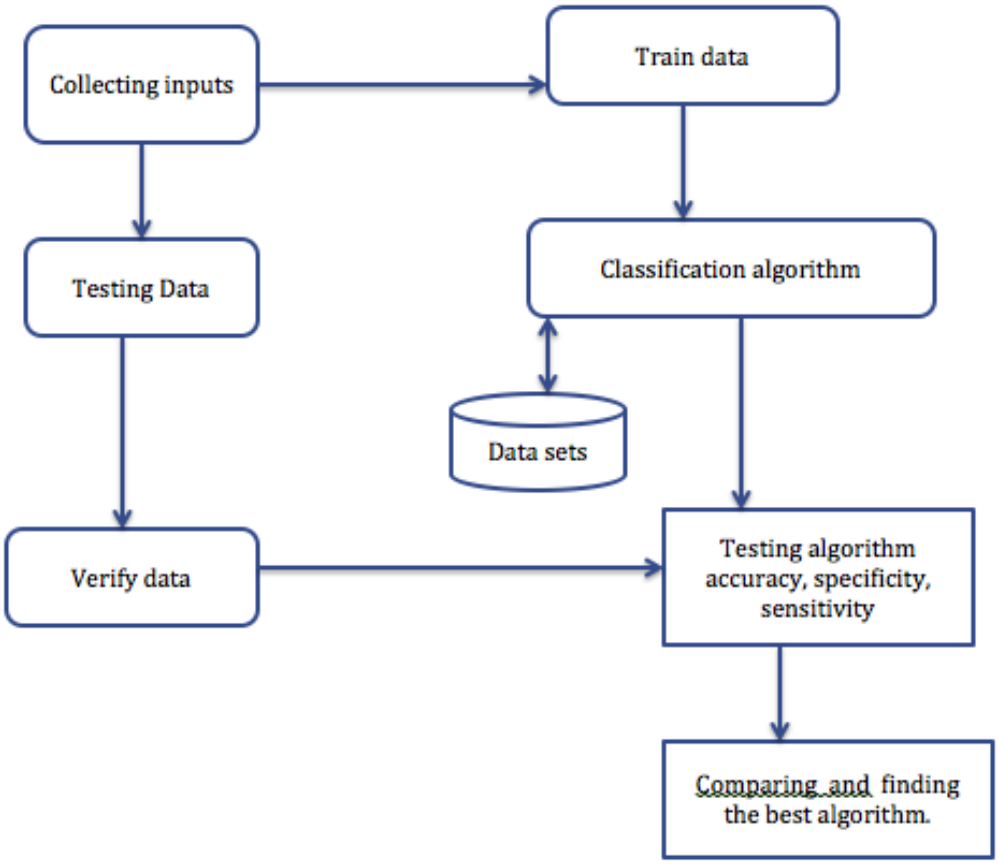
3	Prediction of Movies popularity Using Machine Learning Techniques	International Journal of Computer Science and Network Security, VOL.16 No.8, August 2016	1. IMDB is used for performing data analysis and required machine learning approaches are taken here to build the model which will result in predicting the movies popularity very accurately. 2. They found that result achieved through the simple logistic and logistic regression is around 84%, which is highly accurate. But they failed to show more accurate results with the linear regression technique
4	Sentiment analysis of movie review data using Senti-lexicon algorithm	2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Bangalore, 2016, pp. 592-597.	1. They used algorithm known as Senti-Lexical is used here to find the polarity of reviews as positive, negative or neutral. They also used a method to handle words which have negation effect on the reviews and the role of their emotions. They obtained a histogram depicting the overall result. 2. This resulted around 70% accuracy which is very good enough to find the simplistic approach of the algorithm taken. But there were many challenges that are identified here such as complex sentences ,spam detection etc. which need to be addressed.

2.2 Findings:

Prediction is a challenging task and that too for movie ratings is even more complex, dynamic and mind- boggling. Movie rating prediction poses a big herculean task, because it depends on various parameters to predict. Movie ratings primarily depends on Imdb score, the number of Facebook likes, content rating, movie Facebook likes, rating of actor, duration of the movie and so on. The present research is focused on improving the accuracy of the ratings with help of different algorithms such as Decision tree, Regression, Random Forest, SupportVectorMachine, and Logistic Regression.

3. SYSTEM DESIGN

3.1 System Architecture:

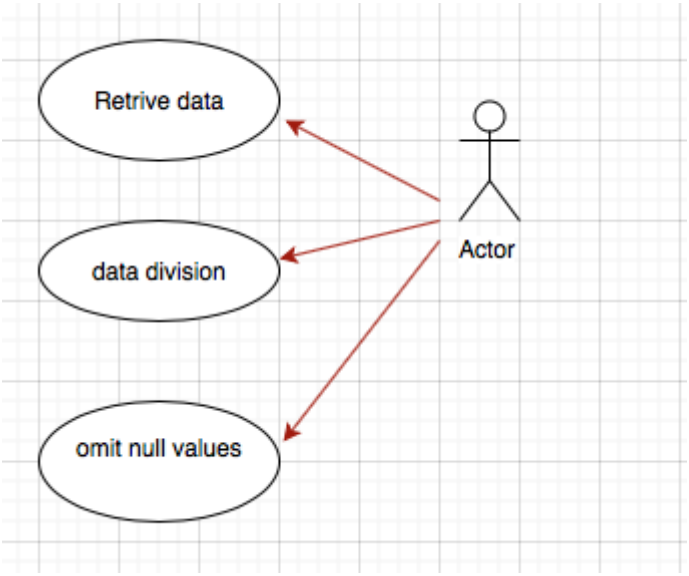


3.2 Module Description:

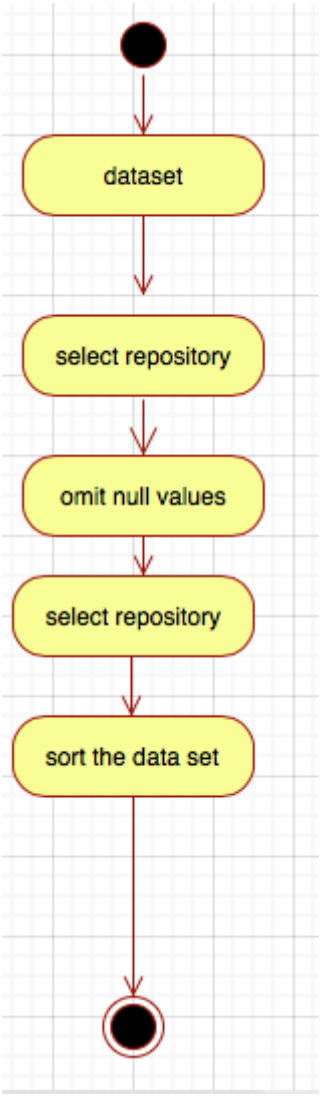
Module 1: Data Pre-Processing:

Data pre-processing is a data mining technique that involves transforming raw data into an understandable format. Preparing of dataset before algorithm is applied on it. Here we take different raw data sets and missing values are filtered using the data cleaning process

Use case Diagram:



Activity Diagram:



Module 2: Feature Selection:

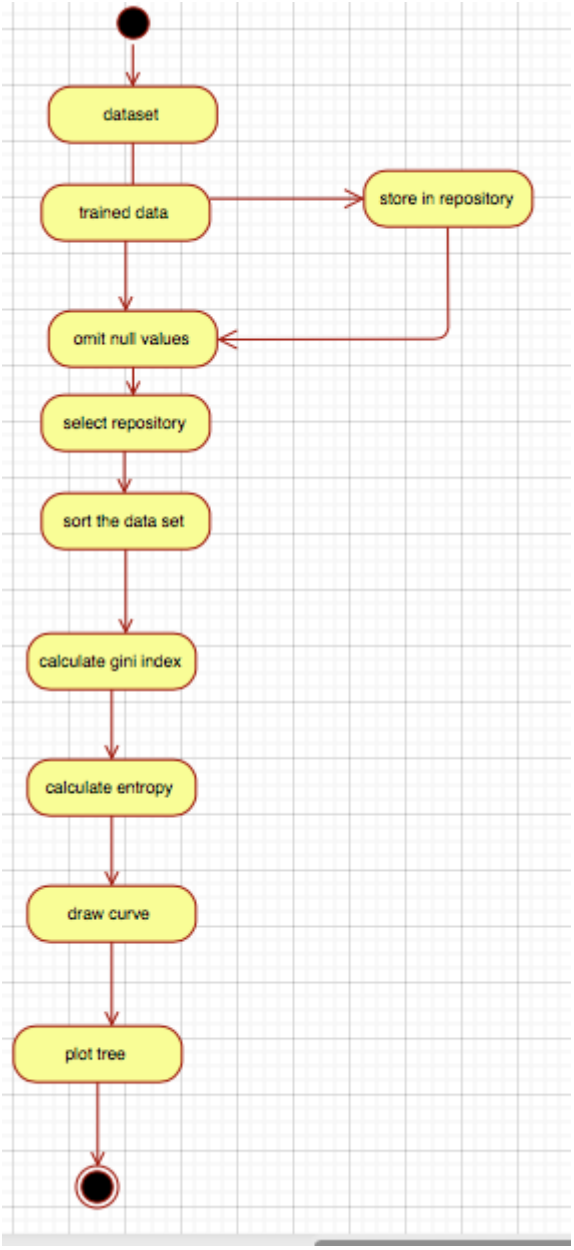
Once initial state is done we analyse the pre-processed data available. It is done to know the common patterns that are affecting the rating like genre, director name, actor name, gross. Information gain of analysed data is done by using the gini index and algorithm for deciding the class label is applied. We plot histogram for clear classification of each attribute based on its frequency and density to this processed dataset. Then, Gini index and entropy are applied to each and every attribute. Next, we plot Lc for each of these Gini and Entropy values. All the values of gini are stored in one variable and entropy is stored in another variable then graph is plotted for these two variables thereby showing the accuracy. When the data is sorted along with its corresponding class labels. Evaluate the interval range

[Interval range = (max – min)/group size]

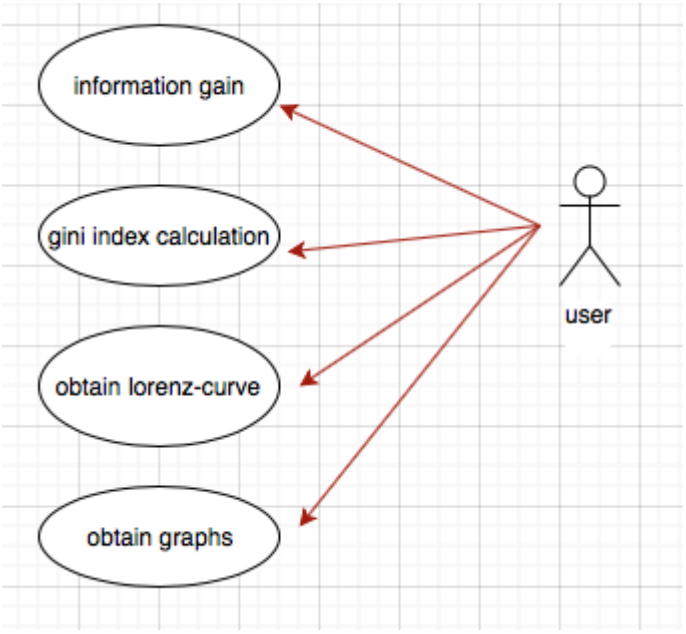
Based on this interval range, evaluate the split points whenever there is a change in the class label.

[split point= midpoint(changed class labels)].

Activity Diagram:



Use case diagram:



Module 3: Classification Algorithms:

1) Logistic Regression: It is one of the highly used regression technique which is useful to predict value of dependent variable. Dependent variable might depend on many independent variables. This prediction can be done using logit function.
2) Decision Tree: It is of two types. One is classical trees and conditional Inference trees. This is a classification technique that helps in predicting the outcome of a variable. The root node is the one with highest information gain or gini index value. After performing the decision tree with training data set ,it is validated with testing set. The prediction is done and an elegant decision tree is formed which gives a crystal clear view of different association rules that can be formed with help of it. These rules help the users or producers to depict the outcome before on hand. In conditional inference tree the features and splits are taken on basis of tests that are performed on the dataset. These tests are known as permutation tests.
3) RandomForest : This contains a group of decision trees. All these trees are grouped into one to make the cases into the groups. It is supervised learning approach. We use randomForest function here to perform it. It is available in the randomForest package. Pruning of tree is done here. It also provides a natural measure to know the importance of feature. Determines the importance of each attribute in the dataset. According to the weight, chooses the best tree. According to the project perspective, the attribute that has higher importance is taken and the bar graph is plotted. According to the plotted graph, the value of highest frequency is taken as reference

for accurate prediction. The dataset is divided into training and testing samples where we apply package (random forest) and test this system using the testing sample.

4) Support Vector Machines: It is also called as SVMs. Now these became popular because of improved prediction in showing up accuracy. In this we use the field of mathematics, a mathematical approach that is underlying in this algorithm gives the highly accurate data and focus on binary classification. We can use this SVMs by using function ksvm(). This is the respective function that is used in R programming. And also usage of svm() function available in the e1071 package is done.

Module 4: Performance Analysis: Rating is classified as A1,A2 using supervised machine-learning techniques. To know which approach is accurate we calculated the accuracy of each classification classifier. But , accuracy alone is not enough to tell that particulary algorithm is the best.So, we calculated the sensitivity,specificity,positive predictive value, negative predective value and accuracy of each classification technique. This one we performed by including all these in performance function. Once the performance function is executed and the one classifier with highest measures will be treated as the best algorithm in terms of accuracy.

3.3 SYSTEM SPECIFICATION

3.3.1 Software Requirements:

Operating System: OS X
Software used: R-Studio, programing using R.

3.3.2 Hardware Requirements:

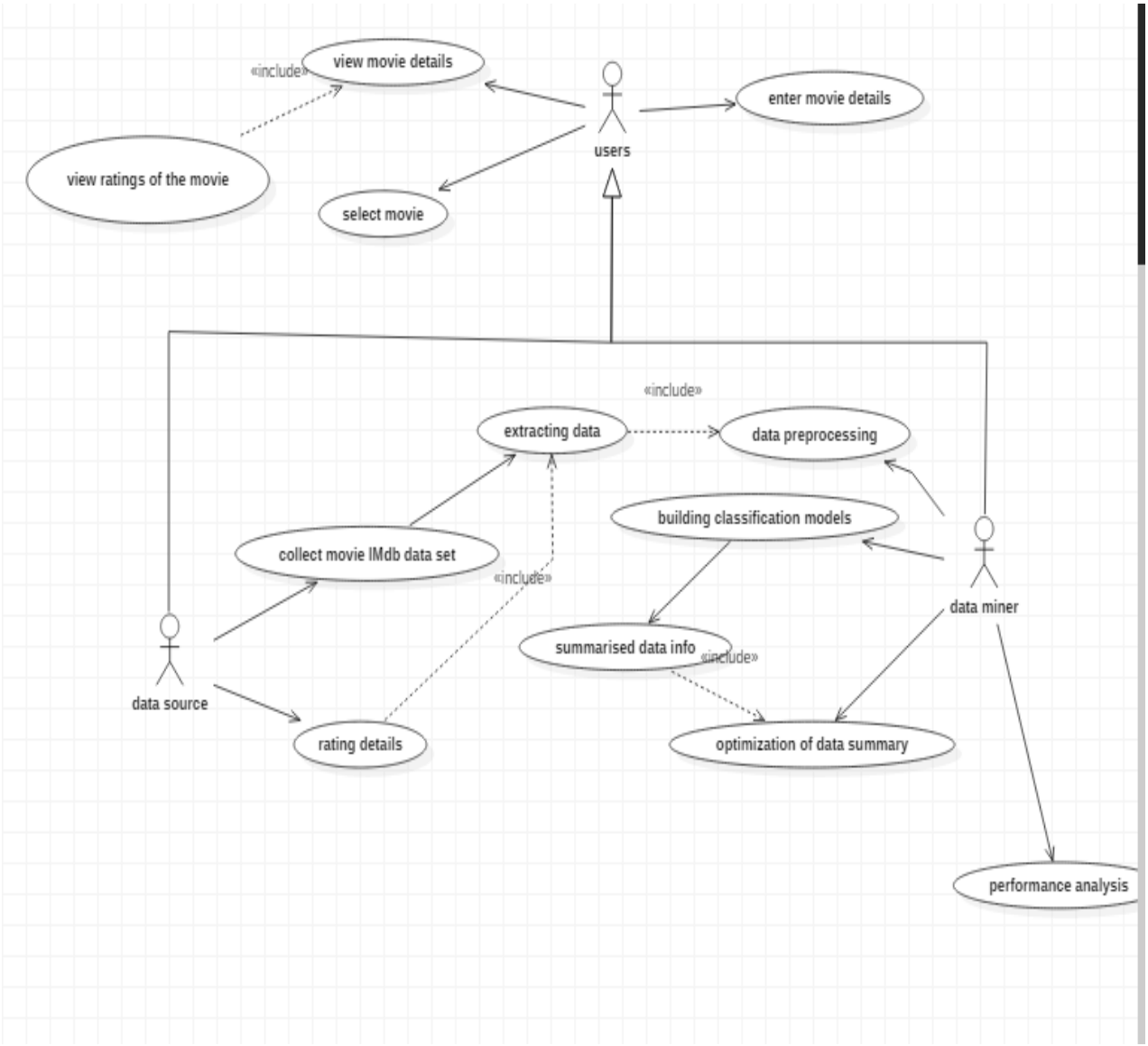
Processor: 2.5 GHZ INTEL CORE I5.

Memory: 4GB 1600 MHz DDR3.

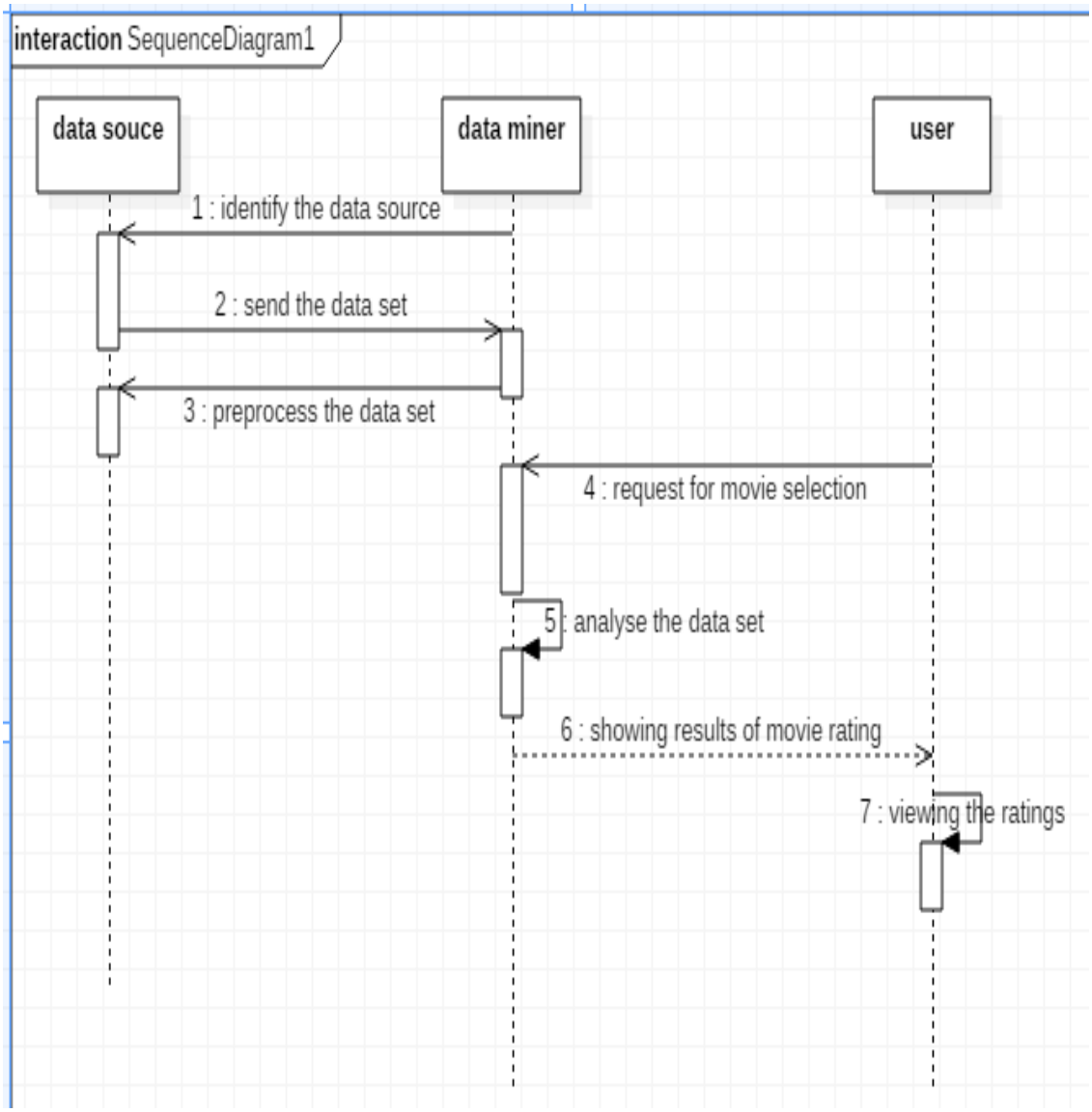
Key Board: Apple Extended Keyboard.

3.4 Detailed Design:

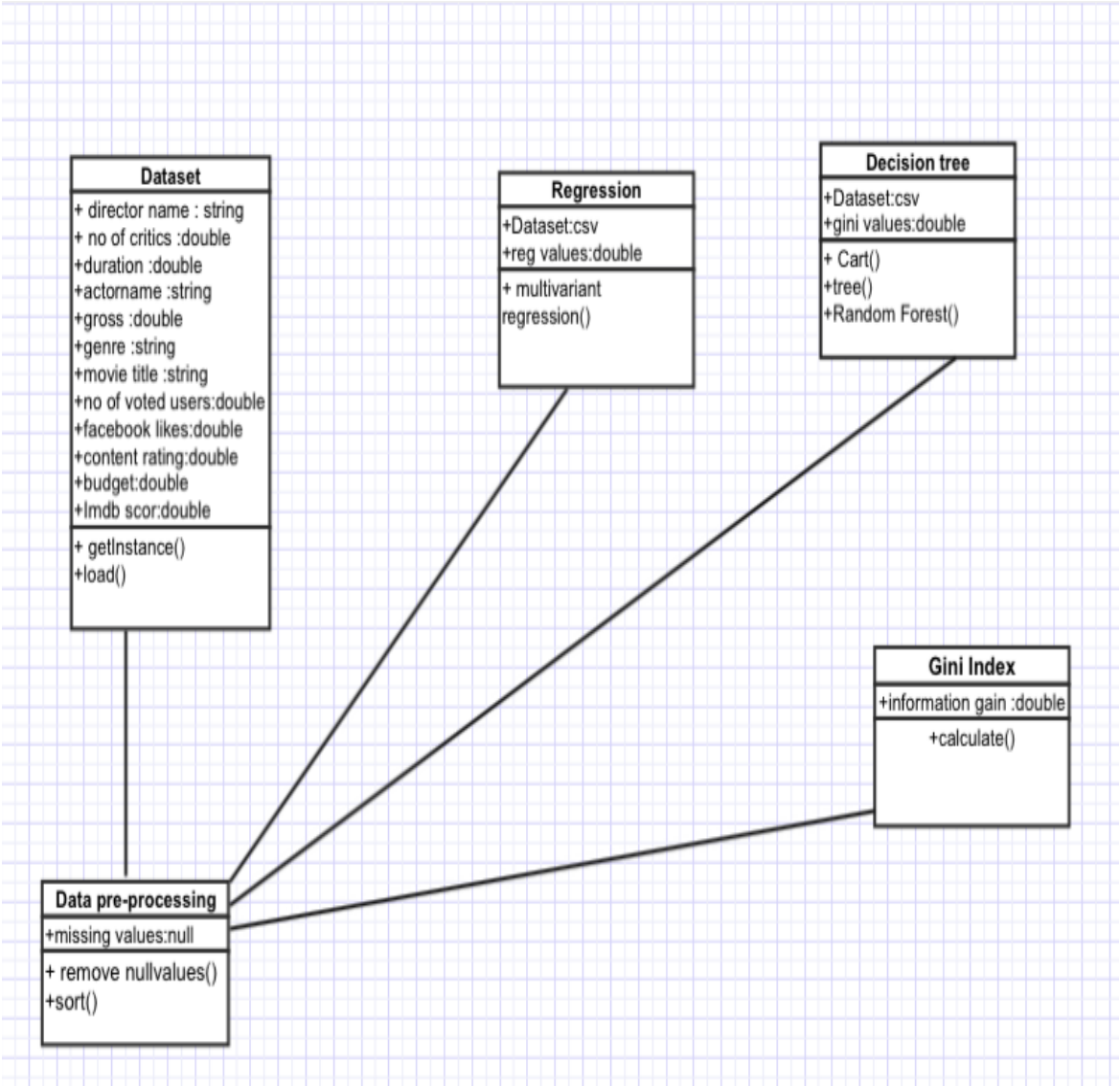
3.4.1 Use-Case Diagram:



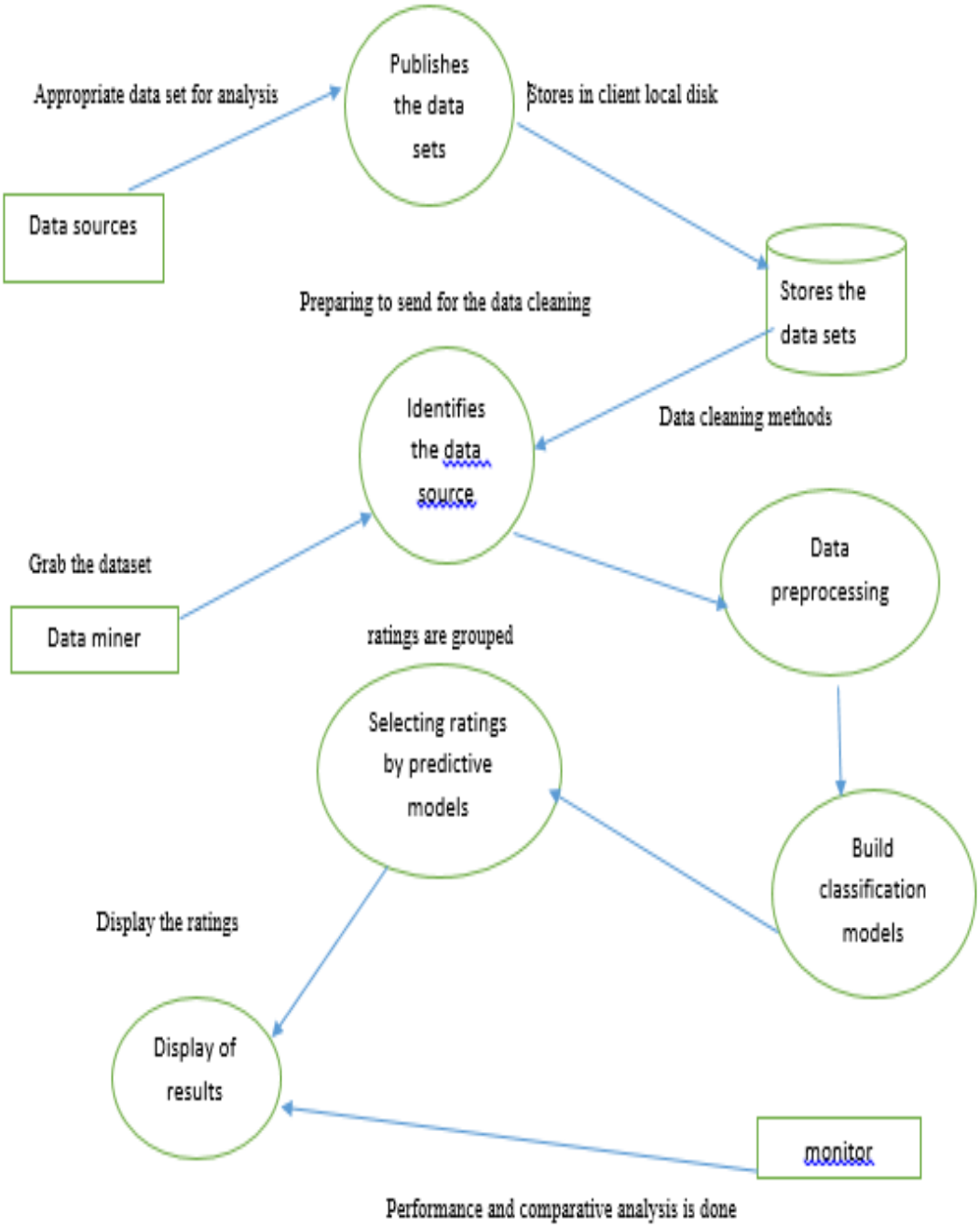
3.4.2 Sequence Diagram:



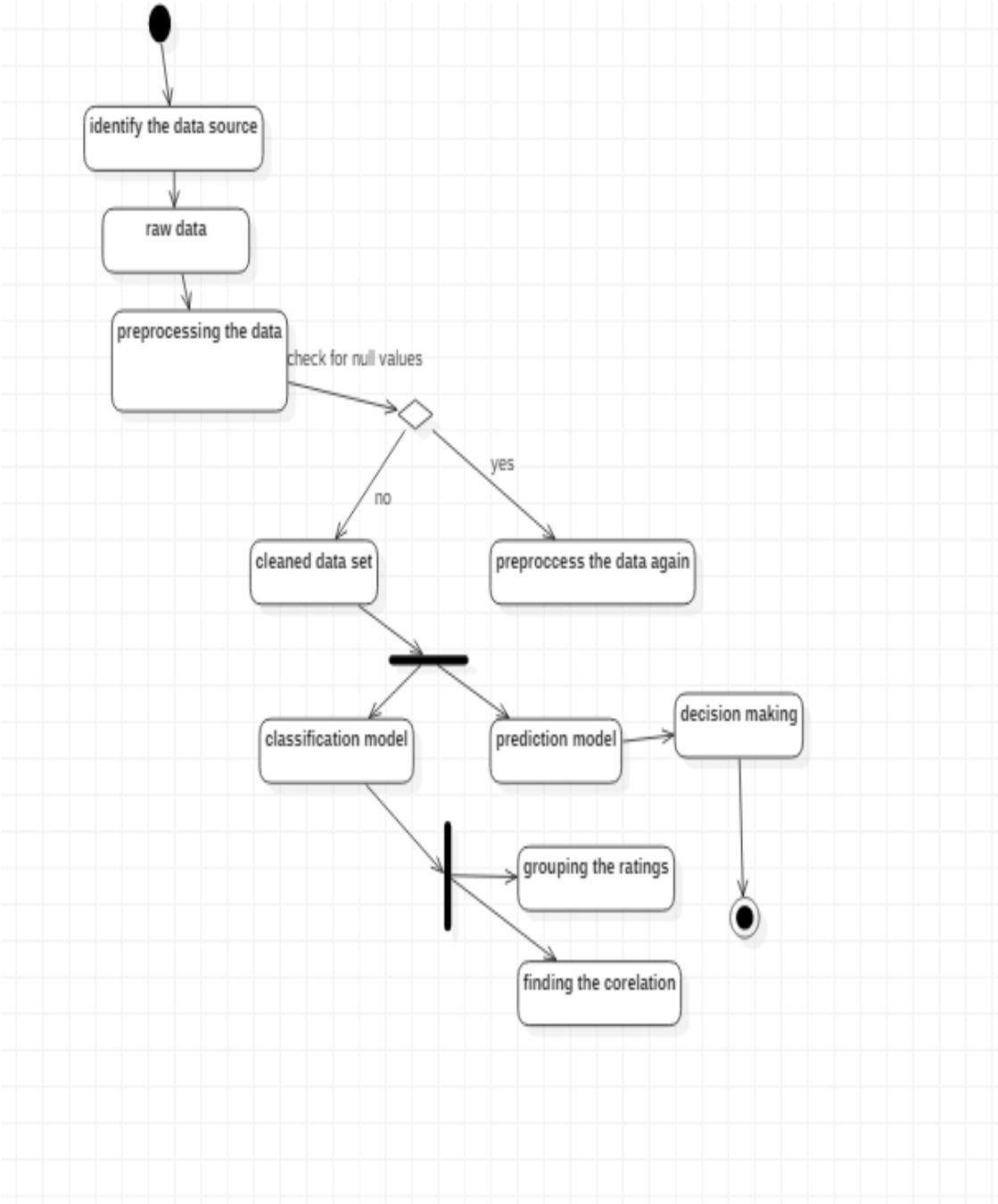
3.4.3 Class Diagram:



3.4.4 Data Flow Diagram:



3.4.5 Activity Diagram:



4. IMPLEMENTATION:

4.1 Implementation Details:

CODE:

```
library("party")
library("rpart")
library("rpart.plot")
library("e1071")
library("randomForest")
library("ineq")
#add file
setwd("C:/Users/Admin/Desktop/summer project/")
play_decision<-play_decision<-read.csv("C:/Users/Admin/Desktop/summer
project/Book2.csv" ,header=TRUE, sep=",")
#show table
View(play_decision)
#null all the values
play1<-play_decision[complete.cases(play_decision),]
View(play1)

play1$rating<-
cut(play1$rating,breaks=c(0,125,290),labels=paste("A",1:2,sep=""))
play2<-play1[complete.cases(play1),]
set.seed(1234)

play_decision <- play_decision[complete.cases(play_decision),]
summary(play_decision)

#divide values to train=70% and validate=30%
dec<-sample(2, nrow(play2), replace=TRUE, prob=c(0.7,0.3))
train<-play2[dec==1,]
validate<-play2[dec==2,]
```

```
table(train$rating)

#glm is used to fit generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution.
fit.logit<-
glm(rating~director_facebook_likes+actor_3_facebook_likes+actor_1_facebook_likes+num_voted_users+facenumber_in_poster+num_user_for_reviews+country+content_rating+actor_2_facebook_likes+imdb_score+aspect_ratio+movie_facebook_likes, data=train , family=binomial())
summary(fit.logit)


prob<- predict(fit.logit ,validate , type="response")
logit.pred<-factor(prob> .5, levels=c(FALSE,TRUE), labels=c("A1","A2"))
logit.perf<- table(validate$rating , logit.pred, dnn=c("Actual","Predicted"))
logit.perf


performance<- function(table,n=2) {
  if(!all(dim(table) == c(2,2)))
    stop("Must be a 2 X 2 table")
  tn = table[1,1]
  fp = table[1,2]
  fn = table[2,1]
  tp = table[2,2]
  sensitivity = tp/(tp+fn)
  specificity = tn/(tn+fp)
  ppp = tp/(tp+fp)
  npp = tn/(tn+fn)
  hitrate = (tp+tn)/(tp+tn+fp+fn)
  result <- paste("sensitivity = ",round(sensitivity, n),
                  "\nSpecificity = ", round(specificity, n),
                  "\nPositive Predictive Value = ", round(ppp, n),
                  "\nNegative Predictive Value = ", round(npp, n),
```

```
            "\nAccuracy = ", round(hitrate, n), "\n", sep="")
  cat(result)
}
#performance analysis of glm
performance(logit.perf)

#Recursive Partitioning And Regression Trees
fit<-
rpart(rating~director_facebook_likes+actor_3_facebook_likes+imdb_score+a
spect_ratio,data=train, method="class",parms=list(split="information"))
summary(fit)
fit$cptable

plotcp(fit)

#decision tree
dtree.pruned<-prune(fit, cp=.0125)
fit.pruned<-prune(fit, cp=.0125)
prp(fit.pruned, type=2, extra=104, fallen.leaves = TRUE, main="Decision
Tree")
dtree.pred<-predict(fit.pruned, validate, type="class")
dtree.perf<-table(validate$rating, dtree.pred, dnn=c("Actual","Predicted"))
dtree.perf

performance(dtree.perf)

#Conditional Inference Trees
#Recursive partitioning for continuous, censored, ordered, nominal and
multivariate response variables in a conditional inference framework.
fit.ctree<-ctree(rating~.,data=train)
plot(fit.ctree, main="Conditional Inference Tree")
ctree.pred<-predict(fit.ctree, validate, type="response")
ctree.perf<-table(validate$rating, ctree.pred, dnn=c("Actual","Predicted"))
ctree.perf
```

```
performance(ctree.perf)

#histogram
hist(play2$movie_facebook_likes, main="histogram of movie likes", xlab="movie_facebook_likes",border="blue",col="green",xlim=c(36,98),las=0,breaks=10)
hist(play2$aspect_ratio, main="histogram of aspect_ratio", xlab="aspect_ratio",border="red",col="orange",xlim=c(1,2.4),las=0,breaks=10)
hist(play2$imdb_score, main="histogram of imdb_score", xlab="imdb_score",border="blue",col="pink",xlim=c(2,8),las=0,breaks=10)
hist(play2$movie_facebook_likes, main="histogram of movie likes", xlab="movie_facebook_likes",border="blue",col="green",xlim=c(36,98),las=0,breaks=10)

#Random forest
fit.forest<-
randomForest(rating~content_rating+aspect_ratio+imdb_score+language+country+director_facebook_likes, data=train, na.action=na.roughfix, importance=TRUE)
 fit.forest
plot(fit.forest)
forest.pred<-predict(fit.forest,validate)
forest.perf<-table(validate$rating,forest.pred,dnn=c("Actual","Predicted"))
forest.perf
performance(forest.perf)

#Simple vector machine
fit.svm<-svm(rating~.,data=train)
fit.svm
svm.pred<-predict(fit.svm, na.omit(validate))
svm.perf<-
table(na.omit(validate)$rating,svm.pred,dnn=c("Actual","Predicted"))
```

```
svm.perf

performance(svm.perf)

# Gini index
g<-ineq(play2$director_facebook_likes, type="Gini")
g
plot(Lc(play2$director_facebook_likes), col="purple", lwd=2)
savehistory("~/C:/Users/Admin/Desktop/summer project/gini.Rhistory")

#green--content_rating
h<-ineq(play2$content_rating, type="Gini")
h
plot(Lc(play2$content_rating), col="green", lwd=2)

#blue--imdb_score
h<-ineq(play2$imdb_score, type="Gini")
h
plot(Lc(play2$imdb_score), col="blue", lwd=2)

#pink--country
i<-ineq(play2$country, type="Gini")
i
plot(Lc(play2$country), col="pink", lwd=2)

#orange--language
j<-ineq(play2$language, type="Gini")
j
plot(Lc(play2$language), col="orange", lwd=2)

#blue--imdb_score
k<-ineq(play2$imdb_score, type="Gini")
k
plot(Lc(play2$imdb_score), col="blue", lwd=2)
```

```
#blue--director_facebook_likes
a<-ineq(play2$director_facebook_likes, type="Gini")
a
plot(Lc(play2$director_facebook_likes), col="purple", lwd=2)

#Entropy
#CONTENT RATING
b<-ineq(play2$content_rating, type="entropy")
b
plot(Lc(play2$content_rating), col="green", lwd=2)

#director facebook
a<-ineq(play2$director_facebook_likes, type="entropy")
a
plot(Lc(play2$content_rating), col="purple", lwd=2)

#country
c<-ineq(play2$country, type="entropy")
c
plot(Lc(play2$country), col="pink", lwd=2)

#language
d<-ineq(play2$language, type="entropy")
d
plot(Lc(play2$language, col="orange", lwd=2))

x1<-c(g,k)
x2<-c(b,c,d)

plot(x1, type="o", col="blue",ylim=c(0.0,0.4))
par(new=TRUE)
lines(x2,type="o",col="red")
model<-naiveBayes(rating~.,data=train)
```

```
newdata<-
data.frame(director_facebook_likes="0",actor_3_facebook_likes="530",actor
_1_facebook_likes="895",num_voted_users="700",facenumber_in_poster="
1",num_user_for_reviews="600",language="English",country="USA",conten
t_rating="PG",actor_2_facebook_likes="89",imdb_score="7.5",aspect_ratio=
"2.35",movie_facebook_likes="8900",movie_title="Bahubali")
newdata
model<-naiveBayes(rating~.,data=train)
predict(model,newdata,type="class")
```

OUTPUT:

```
> library("party")
Loading required package: grid
Loading required package: mvtnorm
Loading required package: modeltools
Loading required package: stats4
Loading required package: strucchange
Loading required package: zoo

Attaching package: ‘zoo’

The following objects are masked from ‘package:base’:

    as.Date, as.Date.numeric

Loading required package: sandwich
> library("rpart")
> library("rpart.plot")
> library("e1071")
Warning message:
package ‘e1071’ was built under R version 3.5.1
> library("randomForest")
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.
> library("ineq")
> #add file
> setwd("C:/Users/Admin/Desktop/summer project/")
> play_decision<-play_decision<-read.csv("C:/Users/Admin/Desktop/summer project/Book2.csv" ,
header=TRUE, sep=",")
```

```
> #show table
> View(play_decision)
> #null all the values
> play1<-play_decision[complete.cases(play_decision),]
> View(play1)
>
> play1$rating<-cut(play1$rating,breaks=c(0,125,290),labels=paste("A",1:2,sep=""))
> play2<-play1[complete.cases(play1),]
> set.seed(1234)
>
> play_decision <- play_decision[complete.cases(play_decision),]
> summary(play_decision)
    rating      director_facebook_likes actor_3_facebook_likes actor_1_facebook_likes
num_voted_users
Min.   : 73.0   Min.   :    0.00       Min.   :    0.0       Min.   :    2         Min.   :     57
1st Qu.:101.0   1st Qu.:   13.75       1st Qu.:  329.8       1st Qu.: 1000         1st Qu.:  78808
Median :116.0   Median :  124.00       Median :  595.0       Median :11000         Median : 166742
Mean   :119.9   Mean   : 1327.14       Mean   : 1640.4       Mean   :12106         Mean   : 228894
3rd Qu.:134.0   3rd Qu.:  401.25       3rd Qu.:  940.0       3rd Qu.:18000         3rd Qu.: 314088
Max.   :240.0   Max.   :22000.00       Max.   :23000.0       Max.   :87000         Max.   :1676169

 facenumber_in_poster num_user_for_reviews       language       country    content_rating
actor_2_facebook_likes
Min.   : 0.000       Min.   :  1.0              : 0   USA     :453   PG-13 :277    Min.   :    0.0
1st Qu.: 0.000       1st Qu.: 248.8       Aboriginal:  1   UK      : 28   PG    :132    1st Qu.:  592.0
Median : 1.000       Median : 476.5         English   :508   Germany : 10   R     : 79    Median :  946.5
Mean   : 1.229       Mean   : 667.6         French    :  3   France  :  7   G     : 24    Mean   : 4179.7
3rd Qu.: 2.000       3rd Qu.: 799.8         Japanese  :  2   Australia:  6         :  4    3rd Qu.: 8000.0
Max.   :15.000       Max.   :5060.0         Mandarin  :  1   Canada  :  3   TV-14 :  0    Max.   :27000.0
                                            Spanish   :  1   (Other) :  9   (Other):  0
  imdb_score     aspect_ratio   movie_facebook_likes
Min.   :2.200   Min.   :1.500   Min.   :     0
1st Qu.:6.000   1st Qu.:1.850   1st Qu.:     0
Median :6.700   Median :2.350   Median :  2000
Mean   :6.606   Mean   :2.206   Mean   : 21941
3rd Qu.:7.300   3rd Qu.:2.350   3rd Qu.: 29000
Max.   :9.000   Max.   :2.390   Max.   :349000

> #divide values to train=70% and validate=30%
> dec<-sample(2, nrow(play2), replace=TRUE, prob=c(0.7,0.3))
> train<-play2[dec==1,]
> validate<-play2[dec==2,]
> table(train$rating)

 A1  A2
236 123
>
> #glm is used to fit generalized linear models, specified by giving a symbolic description
of the linear predictor and a description of the error distribution.
> fit.logit<-glm(rating~director_facebook_likes+actor_3_facebook_likes+actor_1_facebook_likes
```

```
+num_voted_users+facenumber_in_poster+num_user_for_reviews+country+content_rating
+actor_2_facebook_likes+imdb_score+aspect_ratio+movie_facebook_likes, data=train ,
 family=binomial())
> summary(fit.logit)

Call:
glm(formula = rating ~ director_facebook_likes + actor_3_facebook_likes +
    actor_1_facebook_likes + num_voted_users + facenumber_in_poster +
    num_user_for_reviews + country + content_rating + actor_2_facebook_likes +
    imdb_score + aspect_ratio + movie_facebook_likes, family = binomial(),
    data = train)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.0558  -0.7212  -0.2904   0.7110   2.7713

Coefficients:
                        Estimate Std. Error z value Pr(>|z|)
(Intercept)           -2.836e+01  3.956e+03  -0.007    0.994
director_facebook_likes -4.125e-06  3.565e-05  -0.116    0.908
actor_3_facebook_likes   7.133e-05  5.854e-05   1.218    0.223
actor_1_facebook_likes   1.972e-05  1.503e-05   1.312    0.189
num_voted_users         -2.197e-06  1.494e-06  -1.470    0.142
facenumber_in_poster     9.526e-02  7.420e-02   1.284    0.199
num_user_for_reviews     1.380e-03  3.474e-04   3.973 7.09e-05 ***
countryCanada           -1.140e+00  2.192e+00  -0.520    0.603
countryChina             1.764e+01  2.679e+03   0.007    0.995
countryFrance           -5.879e-01  2.135e+00  -0.275    0.783
countryGermany          -5.326e-01  1.913e+00  -0.278    0.781
countryJapan            -3.092e+00  4.845e+03  -0.001    0.999
countryNew Line         -1.290e+01  3.956e+03  -0.003    0.997
countryNew Zealand       1.599e+01  1.650e+03   0.010    0.992
countryUK                4.820e-01  1.728e+00   0.279    0.780
countryUSA               4.385e-01  1.651e+00   0.266    0.791
content_ratingG         -2.001e+00  4.081e+03   0.000    1.000
content_ratingPG         1.368e+01  3.956e+03   0.003    0.997
content_ratingPG-13      1.527e+01  3.956e+03   0.004    0.997
content_ratingR          1.547e+01  3.956e+03   0.004    0.997
actor_2_facebook_likes  -4.084e-05  3.922e-05  -1.041    0.298
imdb_score               1.378e+00  2.809e-01   4.905 9.33e-07 ***
aspect_ratio             1.091e+00  7.373e-01   1.480    0.139
movie_facebook_likes    -5.260e-06  4.213e-06  -1.248    0.212
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 461.50  on 358  degrees of freedom
Residual deviance: 313.49  on 335  degrees of freedom
AIC: 361.49
```

```
Number of Fisher Scoring iterations: 16

> prob<- predict(fit.logit ,validate , type="response")
> logit.pred<-factor(prob> .5, levels=c(FALSE,TRUE), labels=c("A1","A2"))
> logit.perf<- table(validate$rating , logit.pred, dnn=c("Actual","Predicted"))
> logit.perf
      Predicted
Actual A1 A2
   A1 86 11
   A2 30 30
>
>
> performance<- function(table,n=2) {
+   if(!all(dim(table) == c(2,2)))
+     stop("Must be a 2 X 2 table")
+   tn = table[1,1]
+   fp = table[1,2]
+   fn = table[2,1]
+   tp = table[2,2]
+   sensitivity = tp/(tp+fn)
+   specificity = tn/(tn+fp)
+   ppp = tp/(tp+fp)
+   npp = tn/(tn+fn)
+   hitrate = (tp+tn)/(tp+tn+fp+fn)
+   result <- paste("sensitivity = ",round(sensitivity, n),
+                   "\nSpecificity = ", round(specificity, n),
+                   "\nPositive Predictive Value = ", round(ppp, n),
+                   "\nNegative Predictive Value = ", round(npp, n),
+                   "\nAccuracy = ", round(hitrate, n), "\n", sep="")
+   cat(result)
+ }
> #performance analysis of glm
> performance(logit.perf)
sensitivity = 0.5
Specificity = 0.89
Positive Predictive Value = 0.73
Negative Predictive Value = 0.74
Accuracy = 0.74
> #Recursive Partitioning And Regression Trees
> fit<- rpart(rating~director_facebook_likes+actor_3_facebook_likes+imdb_score
+aspect_ratio,data=train, method="class",parms=list(split="information"))
> summary(fit)
Call:
rpart(formula = rating ~ director_facebook_likes + actor_3_facebook_likes +
    imdb_score + aspect_ratio, data = train, method = "class",
    parms = list(split = "information"))
  n= 359

         CP nsplit rel error    xerror       xstd
```

```
1 0.20325203      0 1.0000000 1.0000000 0.07310654
2 0.03658537      1 0.7967480 0.9268293 0.07171056
3 0.02439024      5 0.6422764 0.8211382 0.06926570
4 0.01626016      7 0.5934959 0.7967480 0.06862478
5 0.01000000      8 0.5772358 0.8211382 0.06926570

Variable importance
director_facebook_likes           imdb_score  actor_3_facebook_likes          aspect_ratio
                     44                   41                        9                     6

Node number 1: 359 observations,    complexity param=0.203252
  predicted class=A1  expected loss=0.3426184  P(node) =1
    class counts:   236   123
   probabilities: 0.657 0.343
  left son=2 (252 obs) right son=3 (107 obs)
  Primary splits:
      imdb_score            < 7.15  to the left,  improve=24.803440, (0 missing)
      director_facebook_likes < 2.5   to the right, improve=13.936990, (0 missing)
      aspect_ratio          < 2.025 to the left,  improve= 9.586516, (0 missing)
      actor_3_facebook_likes  < 13500 to the left,  improve= 3.667399, (0 missing)
  Surrogate splits:
      director_facebook_likes < 15000 to the left,  agree=0.733, adj=0.103, (0 split)
      actor_3_facebook_likes  < 17500 to the left,  agree=0.716, adj=0.047, (0 split)

Node number 2: 252 observations,    complexity param=0.02439024
  predicted class=A1  expected loss=0.2261905  P(node) =0.7019499
    class counts:   195    57
   probabilities: 0.774 0.226
  left son=4 (211 obs) right son=5 (41 obs)
  Primary splits:
      director_facebook_likes < 3     to the right, improve=9.922617, (0 missing)
      imdb_score            < 5.95  to the left,  improve=6.898891, (0 missing)
      aspect_ratio          < 2.025 to the left,  improve=3.762866, (0 missing)
      actor_3_facebook_likes  < 59.5  to the left,  improve=2.356167, (0 missing)

Node number 3: 107 observations,    complexity param=0.03658537
  predicted class=A2  expected loss=0.3831776  P(node) =0.2980501
    class counts:    41    66
   probabilities: 0.383 0.617
  left son=6 (96 obs) right son=7 (11 obs)
  Primary splits:
      director_facebook_likes < 15000 to the left,  improve=5.701371, (0 missing)
      aspect_ratio          < 2.1   to the left,  improve=5.156497, (0 missing)
      actor_3_facebook_likes  < 983.5 to the left,  improve=3.632754, (0 missing)
      imdb_score            < 8.4   to the left,  improve=1.813589, (0 missing)
  Surrogate splits:
      actor_3_facebook_likes < 21500 to the left,  agree=0.916, adj=0.182, (0 split)
      imdb_score            < 8.4   to the left,  agree=0.907, adj=0.091, (0 split)

Node number 4: 211 observations
```

```
  predicted class=A1  expected loss=0.1706161  P(node) =0.5877437
    class counts:   175    36
   probabilities: 0.829 0.171

Node number 5: 41 observations,    complexity param=0.02439024
  predicted class=A2  expected loss=0.4878049  P(node) =0.1142061
    class counts:    20    21
   probabilities: 0.488 0.512
  left son=10 (13 obs) right son=11 (28 obs)
  Primary splits:
      actor_3_facebook_likes < 535   to the left,  improve=1.6224440, (0 missing)
      imdb_score             < 6     to the left,  improve=1.2156250, (0 missing)
      aspect_ratio           < 2.1   to the right, improve=0.0594772, (0 missing)
  Surrogate splits:
      imdb_score < 5.3   to the left,  agree=0.756, adj=0.231, (0 split)

Node number 6: 96 observations,    complexity param=0.03658537
  predicted class=A2  expected loss=0.4270833  P(node) =0.2674095
    class counts:    41    55
   probabilities: 0.427 0.573
  left son=12 (23 obs) right son=13 (73 obs)
  Primary splits:
      aspect_ratio           < 2.1   to the left,  improve=4.4700750, (0 missing)
      director_facebook_likes < 2.5   to the right, improve=2.5208750, (0 missing)
      actor_3_facebook_likes  < 8000  to the left,  improve=1.8706720, (0 missing)
      imdb_score             < 8.05  to the right, improve=0.7395902, (0 missing)
  Surrogate splits:
      imdb_score < 8.25  to the right, agree=0.771, adj=0.043, (0 split)

Node number 7: 11 observations
  predicted class=A2  expected loss=0  P(node) =0.03064067
    class counts:     0    11
   probabilities: 0.000 1.000

Node number 10: 13 observations
  predicted class=A1  expected loss=0.3076923  P(node) =0.0362117
    class counts:     9     4
   probabilities: 0.692 0.308

Node number 11: 28 observations,    complexity param=0.01626016
  predicted class=A2  expected loss=0.3928571  P(node) =0.07799443
    class counts:    11    17
   probabilities: 0.393 0.607
  left son=22 (8 obs) right son=23 (20 obs)
  Primary splits:
      imdb_score             < 6.95  to the right, improve=1.2504590, (0 missing)
      actor_3_facebook_likes < 853   to the right, improve=0.5606993, (0 missing)
  Surrogate splits:
      actor_3_facebook_likes < 588   to the left,  agree=0.75, adj=0.125, (0 split)
```

```
Node number 12: 23 observations
  predicted class=A1  expected loss=0.3043478  P(node) =0.06406685
    class counts:    16     7
   probabilities: 0.696 0.304

Node number 13: 73 observations,    complexity param=0.03658537
  predicted class=A2  expected loss=0.3424658  P(node) =0.2033426
    class counts:    25    48
   probabilities: 0.342 0.658
  left son=26 (48 obs) right son=27 (25 obs)
  Primary splits:
      director_facebook_likes < 2.5   to the right, improve=1.8026580, (0 missing)
      actor_3_facebook_likes  < 983.5 to the left,  improve=1.4715050, (0 missing)
      imdb_score              < 7.75  to the right, improve=0.5757009, (0 missing)
  Surrogate splits:
      actor_3_facebook_likes < 437.5 to the right, agree=0.699, adj=0.12, (0 split)
      imdb_score             < 8.35  to the left,  agree=0.699, adj=0.12, (0 split)

Node number 22: 8 observations
  predicted class=A1  expected loss=0.375  P(node) =0.02228412
    class counts:     5     3
   probabilities: 0.625 0.375

Node number 23: 20 observations
  predicted class=A2  expected loss=0.3  P(node) =0.05571031
    class counts:     6    14
   probabilities: 0.300 0.700

Node number 26: 48 observations,    complexity param=0.03658537
  predicted class=A2  expected loss=0.4166667  P(node) =0.1337047
    class counts:    20    28
   probabilities: 0.417 0.583
  left son=52 (10 obs) right son=53 (38 obs)
  Primary splits:
      director_facebook_likes < 87.5  to the left,  improve=10.700580, (0 missing)
      imdb_score              < 8.05  to the right, improve= 1.488198, (0 missing)
      actor_3_facebook_likes  < 738   to the right, improve= 1.159026, (0 missing)
  Surrogate splits:
      actor_3_facebook_likes < 20    to the left,  agree=0.833, adj=0.2, (0 split)
      imdb_score             < 8.15  to the right, agree=0.812, adj=0.1, (0 split)

Node number 27: 25 observations
  predicted class=A2  expected loss=0.2  P(node) =0.06963788
    class counts:     5    20
   probabilities: 0.200 0.800

Node number 52: 10 observations
  predicted class=A1  expected loss=0  P(node) =0.02785515
    class counts:    10     0
   probabilities: 1.000 0.000
```

```
Node number 53: 38 observations
  predicted class=A2  expected loss=0.2631579  P(node) =0.1058496
    class counts:    10    28
   probabilities: 0.263 0.737

> fit$cptable
         CP nsplit rel error    xerror       xstd
1 0.20325203      0 1.0000000 1.0000000 0.07310654
2 0.03658537      1 0.7967480 0.9268293 0.07171056
3 0.02439024      5 0.6422764 0.8211382 0.06926570
4 0.01626016      7 0.5934959 0.7967480 0.06862478
5 0.01000000      8 0.5772358 0.8211382 0.06926570
>
> plotcp(fit)
>
> #decision tree
> dtree.pruned<-prune(fit, cp=.0125)
> fit.pruned<-prune(fit, cp=.0125)
> prp(fit.pruned, type=2, extra=104, fallen.leaves = TRUE, main="Decision Tree")
> dtree.pred<-predict(fit.pruned, validate, type="class")
> dtree.perf<-table(validate$rating, dtree.pred, dnn=c("Actual","Predicted"))
> dtree.perf
      Predicted
Actual A1 A2
    A1 86 11
    A2 28 32
>
> performance(dtree.perf)
sensitivity = 0.53
Specificity = 0.89
Positive Predictive Value = 0.74
Negative Predictive Value = 0.75
Accuracy = 0.75
> #Conditional Inference Trees
> #Recursive partitioning for continuous, censored, ordered, nominal and
 multivariate response variables in a conditional inference framework.
> fit.ctree<-ctree(rating~.,data=train)
> plot(fit.ctree, main="Conditional Inference Tree")
> ctree.pred<-predict(fit.ctree, validate, type="response")
> ctree.perf<-table(validate$rating, ctree.pred, dnn=c("Actual","Predicted"))
> ctree.perf
      Predicted
Actual A1 A2
    A1 76 21
    A2 26 34
>
> performance(ctree.perf)
sensitivity = 0.57
Specificity = 0.78
```

```
Positive Predictive Value = 0.62
Negative Predictive Value = 0.75
Accuracy = 0.7
> #histogram
> hist(play2$movie_facebook_likes, main="histogram of movie likes",
xlab="movie_facebook_likes",border="blue",col="green",xlim=c(36,98),las=0,breaks=10)
> hist(play2$aspect_ratio, main="histogram of aspect_ratio",
xlab="aspect_ratio",border="red",col="orange",xlim=c(1,2.4),las=0,breaks=10)
> hist(play2$imdb_score, main="histogram of imdb_score",
 xlab="imdb_score",border="blue",col="pink",xlim=c(2,8),las=0,breaks=10)
> hist(play2$movie_facebook_likes, main="histogram of movie likes",
 xlab="movie_facebook_likes",border="blue",col="green",xlim=c(36,98),las=0,breaks=10)
>
> #Random forest
> fit.forest<-randomForest(rating~content_rating+aspect_ratio
+imdb_score+language+country+director_facebook_likes,
data=train, na.action=na.roughfix, importance=TRUE)
> ¬        fit.forest
Error: unexpected input in "\"
> plot(fit.forest)
> forest.pred<-predict(fit.forest,validate)
> forest.perf<-table(validate$rating,forest.pred,dnn=c("Actual","Predicted"))
> forest.perf
      Predicted
Actual A1 A2
    A1 90  7
    A2 27 33
> performance(forest.perf)
sensitivity = 0.55
Specificity = 0.93
Positive Predictive Value = 0.82
Negative Predictive Value = 0.77
Accuracy = 0.78
> #Simple vector machine
> fit.svm<-svm(rating~.,data=train)
> fit.svm

Call:
svm(formula = rating ~ ., data = train)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.02702703

Number of Support Vectors:  224

> svm.pred<-predict(fit.svm, na.omit(validate))
```

```
> svm.perf<-table(na.omit(validate)$rating,svm.pred,dnn=c("Actual","Predicted"))
> svm.perf
      Predicted
Actual A1 A2
    A1 93  4
    A2 39 21
>
> performance(svm.perf)
sensitivity = 0.35
Specificity = 0.96
Positive Predictive Value = 0.84
Negative Predictive Value = 0.7
Accuracy = 0.73
> # Gini index
> g<-ineq(play2$director_facebook_likes, type="Gini")
> g
[1] 0.8726979
> plot(Lc(play2$director_facebook_likes), col="purple", lwd=2)
> savehistory("~/C:/Users/Admin/Desktop/summer project/gini.Rhistory")
>
> #green--content_rating
> h<-ineq(play2$content_rating, type="Gini")
> h
[1] 0.1064799
> plot(Lc(play2$content_rating), col="green", lwd=2)
> #blue--imdb_score
> h<-ineq(play2$imdb_score, type="Gini")
> h
[1] 0.08417039
> plot(Lc(play2$imdb_score), col="blue", lwd=2)
>
> #pink--country
> i<-ineq(play2$country, type="Gini")
> i
[1] 0.03907215
> plot(Lc(play2$country), col="pink", lwd=2)
>
> #orange--language
> j<-ineq(play2$language, type="Gini")
> j
[1] 0.009521078
> plot(Lc(play2$language), col="orange", lwd=2)
>
> #blue--imdb_score
> k<-ineq(play2$imdb_score, type="Gini")
> k
[1] 0.08417039
> plot(Lc(play2$imdb_score), col="blue", lwd=2)
> #blue--director_facebook_likes
> a<-ineq(play2$director_facebook_likes, type="Gini")
```

```
> a
[1] 0.8726979
> plot(Lc(play2$director_facebook_likes), col="purple", lwd=2)
>
> #Entropy
> #CONTENT RATING
> b<-ineq(play2$content_rating, type="entropy")
> b
[1] 0.02444277
> plot(Lc(play2$content_rating), col="green", lwd=2)
>
> #director facebook
> a<-ineq(play2$director_facebook_likes, type="entropy")
> a
[1] 1.859445
> plot(Lc(play2$content_rating), col="purple", lwd=2)
>
> #country
> c<-ineq(play2$country, type="entropy")
> c
[1] 0.01792849
> plot(Lc(play2$country), col="pink", lwd=2)
> #language
> d<-ineq(play2$language, type="entropy")
> d
[1] 0.002762154
> plot(Lc(play2$language, col="orange", lwd=2)
> x1<-c(g,k)
> x2<-c(b,c,d)
> plot(x1, type="o", col="blue",ylim=c(0.0,0.4))
> par(new=TRUE)
> lines(x2,type="o",col="red")
> model<-naiveBayes(rating~.,data=train)
> newdata<- data.frame(director_facebook_likes="0",actor_3_facebook_likes="530",actor_1_facebook
in_poster="1",num_user_for_reviews="600",language="English",country="USA",content_rating="PG"
pect_ratio="2.35",movie_facebook_likes="8900",movie_title="Bahubali")
> newdata
  director_facebook_likes actor_3_facebook_likes actor_1_facebook_likes
num_voted_users facenumber_in_poster
1                       0                    530                    895          700                    1
  num_user_for_reviews language country content_rating actor_2_facebook_likes
imdb_score aspect_ratio
1                  600  English     USA            PG                     89       7.5        2.35
  movie_facebook_likes movie_title
1                 8900    Bahubali
> model<-naiveBayes(rating~.,data=train)
> predict(model,newdata,type="class")
[1] A2
Levels: A1 A2
```

```
q<- play2[order(play2$aspect_ratio),c(2,5)]
> q
    director_facebook_likes num_voted_users
316                      63          71527
299                      10         128285
363                       7         171792
444                     255         117212
510                      28         644348
467                      30         112167
1                         0         886204
111                     150         106446
238                       0         464310
263                      67          66593
264                       6          13581
298                      21         102933
481                      11          27543
8                        15         294810
18                        0         995415
20                      188         268154
34                    13000         306320
36                       37         235025
44                      125         544884
53                        0         381148
68                        0         665575
71                    17000         245333
72                      188         129601
77                       58         144337
79                        0         345198
80                     4000         106072
84                      198         317542
107                      33          21352
108                      50         211971
119                   13000         320284
122                      35         146019
132                      59         146766
134                     473         152826
135                   13000         199039
137                     394          89442
139                      58         105902
145                      42          85086
147                      50          47900
149                      35         119213
153                     188         270207
157                      52         123553
162                       0         544665
168                       0         212106
171                       6          72259
183                      35          60230
186                     235         313866
189                      12          70121
```

190	14000	334345
194	0	200022
199	0	38438
201	189	67223
215	719	240241
218	541	189855
219	2000	141414
224	0	440084
229	50	36471
234	274	35066
239	0	585659
251	189	234480
267	67	27257
274	34	36033
277	50	94172
278	63	89351
282	23	4102
291	0	324671
294	80	85531
310	541	190786
311	719	221521
317	52	52029
320	6	40751
321	2000	71782
322	31	20295
324	20	27918
325	342	18697
329	9	54077
330	0	49739
332	116	106790
334	127	59581
335	10	58300
339	475	692482
345	394	219501
348	212	165618
353	188	403014
354	487	385871
359	107	166791
367	0	477300
370	12	9418
371	0	183247
372	221	156348
374	14000	238747
377	719	101834
386	0	87785
387	81	58227
397	0	394317
403	266	164148
407	36	26413
409	278	182757

411	278	69860
413	28	65270
420	6	35446
421	102	169023
422	99	61417
423	189	19228
424	763	70838
425	1000	273921
427	88	39391
428	67	71424
430	153	246492
431	0	29932
432	62	10233
434	0	208422
437	293	296904
442	13000	215255
443	12	127345
445	56	27838
449	31	22838
450	221	107817
451	88	168172
452	278	91092
454	91	48500
455	77	103022
459	221	172878
465	10	63625
468	189	65499
473	40	16385
477	438	86422
478	69	666
479	266	56501
483	31	22955
487	65	80639
488	0	98472
493	79	45231
494	357	53132
501	750	24868
509	2000	610568
511	62	67296
515	221	60370
518	101	65037
519	5	16611
520	11	24407
522	275	286877
524	14000	278362
525	35	266636
526	0	361767
30	365	418214
89	663	128306
126	69	421658

2	563	471220
3	0	275868
4	22000	1144337
6	475	212204
7	0	383056
9	0	462669
10	282	321795
11	0	371639
12	0	240396
13	395	330784
14	563	522040
15	563	181792
16	0	548573
17	80	149922
19	252	370704
21	0	354228
22	464	451803
23	0	211765
24	0	483540
25	129	149019
26	0	316018
27	0	793059
28	94	272670
29	532	202382
31	0	522030
32	0	411164
33	1000	557489
35	420	383427
37	0	323207
38	0	242420
39	0	175409
40	464	321227
41	364	264183
42	487	101178
43	258	223393
45	368	286095
46	0	278232
47	395	465019
48	0	514125
49	14000	395573
50	0	106416
51	1000	362912
52	179	222403
54	0	326180
55	14000	333847
56	113	62836
57	56	273556
58	681	53607
59	475	718837
60	420	121084

61	776	283418
62	0	72809
63	0	139593
64	282	42372
65	80	286506
66	0	148379
67	22000	1676169
69	11	114553
70	4000	696338
73	357	117927
74	452	118992
75	293	115099
76	218	431620
78	208	174578
81	4000	522371
82	274	228554
83	171	252257
86	47	116994
87	94	496749
88	31	138661
91	66	272534
92	0	120798
93	776	58137
94	255	485430
95	84	305340
96	571	682155
97	22000	928227
98	22000	1468200
99	28	374
101	357	272223
102	21000	459346
103	905	518537
104	508	166137
105	226	124185
106	249	82380
109	0	111609
110	230	188457
112	0	254111
113	0	513158
114	0	138863
115	282	355137
116	179	385670
117	532	343648
118	508	530870
120	663	473887
121	22000	980946
123	189	130272
124	151	361924
125	0	364948
127	0	421818

128	230	414070
129	750	552503
130	2000	207839
131	0	536314
133	12	33042
136	188	232187
138	282	42372
140	90	182718
141	0	118951
142	14000	256695
143	776	164238
144	25	17590
146	456	39956
148	249	381672
150	93	169914
151	176	69757
152	0	322395
154	0	142440
155	5	64322
156	663	365104
158	23	110788
159	380	317166
160	0	128682
161	14000	504419
163	255	221128
164	295	51892
165	357	45455
166	0	392474
167	503	127497
169	209	146352
170	42	77673
172	394	508818
173	150	157519
174	608	168207
175	386	185394
176	750	32399
177	255	326286
179	14000	12572
180	0	406020
181	13	62424
182	563	183208
184	521	491077
185	54	307029
187	508	498397
188	386	185394
191	0	178126
192	50	114287
193	176	245621
195	14000	177383
196	0	382255

197	1000	102338
198	0	158720
202	96	172754
203	0	444683
204	124	91640
205	28	374
206	563	809474
208	508	305008
209	2000	314253
210	107	58498
211	0	405973
212	681	284792
213	0	338635
214	255	229679
216	323	101411
217	209	229823
220	776	333248
221	610	242188
222	249	133076
223	167	213275
225	160	182661
226	521	40123
227	368	100821
228	0	456260
230	662	338087
231	123	183909
232	294	182899
233	0	57873
235	446	148280
236	364	387436
237	0	520104
240	446	328067
241	0	534658
242	16	150618
243	0	20567
244	610	55994
245	19	37750
246	473	167085
247	0	582917
248	79	62271
250	128	110486
252	62	147497
253	55	149680
254	776	207613
255	0	284852
256	218	348861
257	124	154621
258	17000	264318
259	11	53160
260	1000	136019

262	263	225273
265	124	44296
266	189	254841
268	101	60573
269	151	184561
270	235	336235
271	0	1238746
272	153	68720
273	0	79186
275	295	387632
276	0	217373
279	0	472488
281	0	263329
283	57	218341
284	0	982637
285	14000	399651
286	0	387616
287	258	470483
288	13000	177725
289	0	744891
290	0	230931
292	0	190439
293	12000	149998
295	285	189806
296	55	84424
297	16000	955174
300	165	246803
301	226	255447
302	14	108076
303	0	87677
304	456	39956
305	77	85833
306	207	176598
307	0	89509
308	380	400292
309	17000	780588
312	670	284825
313	0	65785
314	26	87451
315	420	115687
318	385	127258
319	19	33953
323	521	110364
326	208	248045
327	17000	314033
328	611	38690
331	0	292022
333	0	343274
336	20	145321
337	0	103737

338	0	125109
340	0	1215718
341	0	1100446
342	44	54501
343	165	157016
344	0	172707
346	81	403836
347	70	68935
349	102	16832
350	663	479166
351	212	21102
352	0	141179
355	12000	149947
356	420	160440
357	0	98403
358	97	152601
360	0	236421
361	368	145350
362	17000	873649
364	0	307539
365	21000	330152
366	323	299258
368	335	57
369	21	72591
373	0	89770
375	50	32049
376	0	200556
378	521	86152
379	87	86627
380	101	102747
381	209	40862
382	176	52136
383	12000	121259
384	0	60467
385	107	165333
388	79	45602
389	0	49311
390	128	110486
391	468	113065
392	2000	148238
393	96	2508
394	378	57661
395	101	144053
396	357	272223
398	750	132501
399	521	348232
400	249	146134
401	0	402645
402	545	88542
404	36	142496

406	79	166693
408	420	110073
410	12000	188116
412	0	256928
414	168	341058
415	218	188887
416	532	82731
417	323	106528
418	252	119286
419	681	179500
426	36	77029
429	38	203458
433	218	243053
435	21000	301279
436	55	100001
438	13000	270226
439	0	300542
440	378	701607
441	480	375879
446	75	135601
447	23	125036
448	88	130070
453	17000	786092
456	610	283967
457	163	72326
458	0	200359
461	165	225282
462	154	150764
463	333	118483
464	117	227072
466	101	58184
469	301	139423
470	503	76099
471	452	303185
472	425	60910
474	21	44143
475	153	44662
476	81	58402
482	25	103787
484	57	155532
485	258	71574
486	25	83560
489	13000	172217
490	92	979
491	93	55913
492	84	151424
495	43	8913
496	64	65464
497	287	24183
498	252	37446

```
499                  503          53057
500                    0          56403
502                   42          58752
 [ reached getOption("max.print") -- omitted 16 rows ]
```

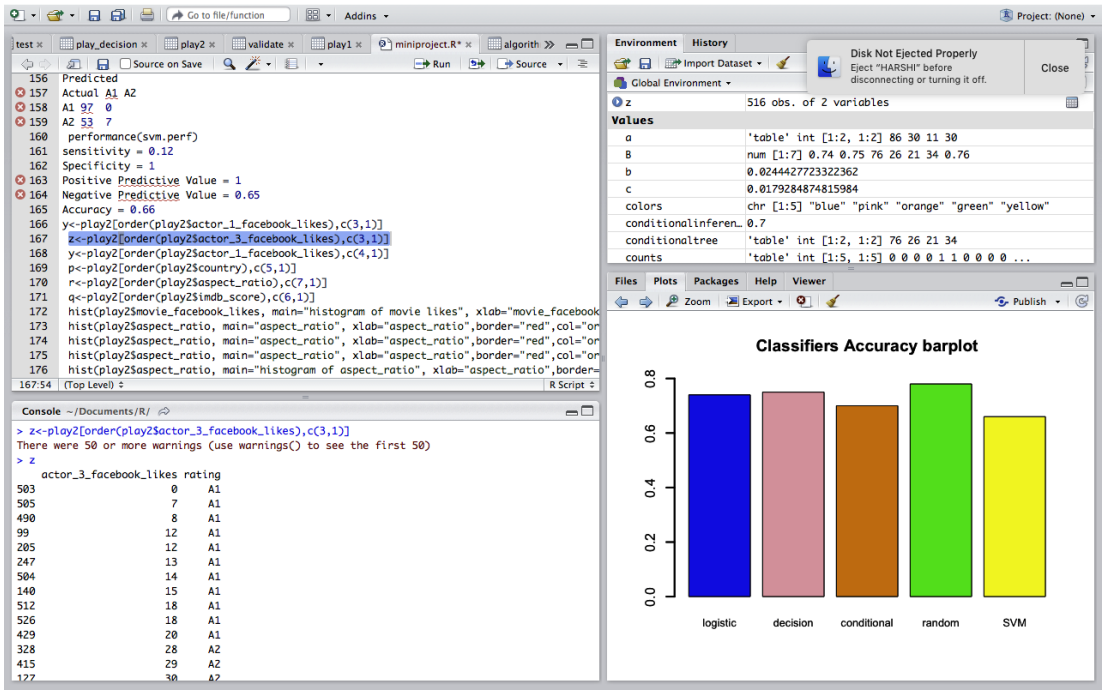
4.1 Unit Testing

- The main aim of unit testing is to break down each component of the program and test these individual components for its proper function.
- If, for any condition when set of inputs are given then it should return proper outputs. When any input is given if the system fails it should be handled gracefully.
- Unit testing is basically done before integration testing as shown below.

Code:

```
> q<- play2[order(play2$aspect_ratio),c(2,5)]
> q
```

Result



5. TEST RESULTS:

5.1 TEST CASESES:

5.1.1 Test Case -1:

Use Case ID:	1
Use Case Name:	Validate package creation
Description:	The packages to be used must be installed without any malfunction
Trigger	For the execution of code
Preconditions:	Type package name in text box and click install.
Post conditions:	1) Successfully installed 2)The details of package are displayed.
Normal Flow :	1)The System displays the list of packages. 2) Displays message “successfully installed”
Alternative Flows:	Unsuccessful installation

5.1.2 Test Case Generation :2

Use Case ID:	2
Use Case Name:	Giving appropriate parameter to plot graph
Description:	The parameter act as an input so that we can get graphs as output
Preconditions:	Packages should be installed already
Post conditions:	1)A graph is displayed

Normal Flow	1)Give limit values for x and y axis 2) Plot the graph based on the given parameters. 3) Result is a graph .
Alternative Flows:	1) Error in given parameters. 2) No graph

5.1.3 Test Case Generation 3:

Use Case ID:	3
Use Case Name:	Categorical and numerical datasets
Description:	Given datasets should be either numerical or categorical no special characters are allowed.
Trigger:	To predict the ratings of movies using Imdb data set.
Preconditions:	Obtaining the datasets from CIS repository
Post conditions:	not applicable
Normal Flow:	Taking datasets from 2011 - 2016
Alternative Flows:	Error in getting datasets.

6. RESULTS AND DISCUSSIONS:

Data mining techniques are applied on the taken dataset IMDB. Performance analysis of classical decision tree, conditional Inference tree , random Forest , linear regression and support vector machine is calculated. Out of these all randomForest gets the highest accuracy. As we know the classifier with more accuracy will be within realm of usage. In this by feature selection important features that highly effect rating are analyzed which can be used for reference.

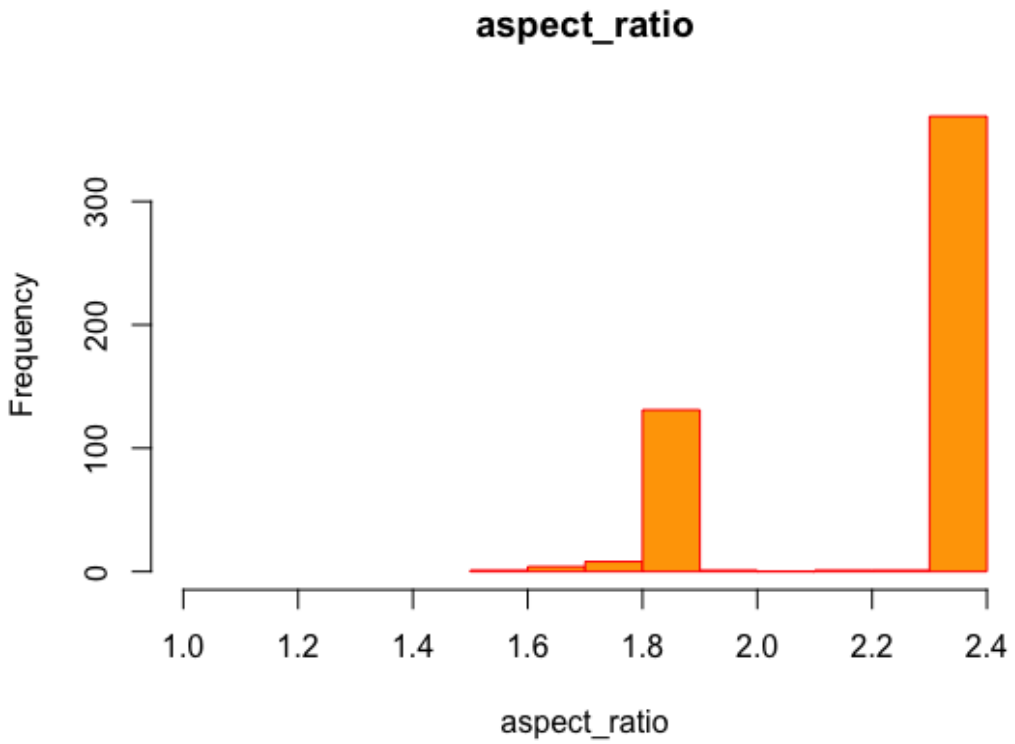
6.1 Outputs/Results

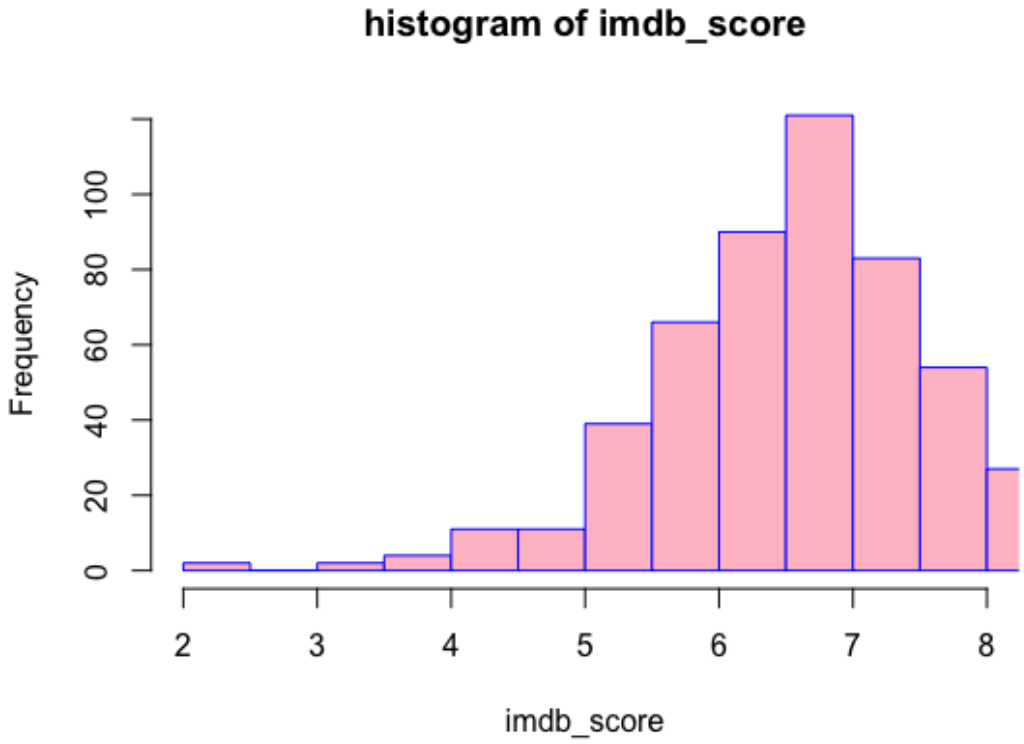
6.1.1 Dataset

	rating	director_facebook_likes	actor_3_facebook_likes	actor_1_facebook_likes	num_voted_users	facenumb
6	A2	475	530	640	212204	
15	A2	563	1000	40000	181792	
17	A2	80	201	22000	149922	
27	A2	0	794	29000	793059	
29	A2	532	627	14000	202382	
30	A1	365	1000	3000	418214	
37	A2	0	464	894	323207	
40	A2	464	825	15000	321227	
41	A1	364	1000	12000	264183	
51	A2	1000	77	29000	362912	
54	A2	0	581	894	326180	
59	A1	475	522	1000	718837	
61	A2	776	310	1000	283418	
62	A1	0	10000	18000	72809	

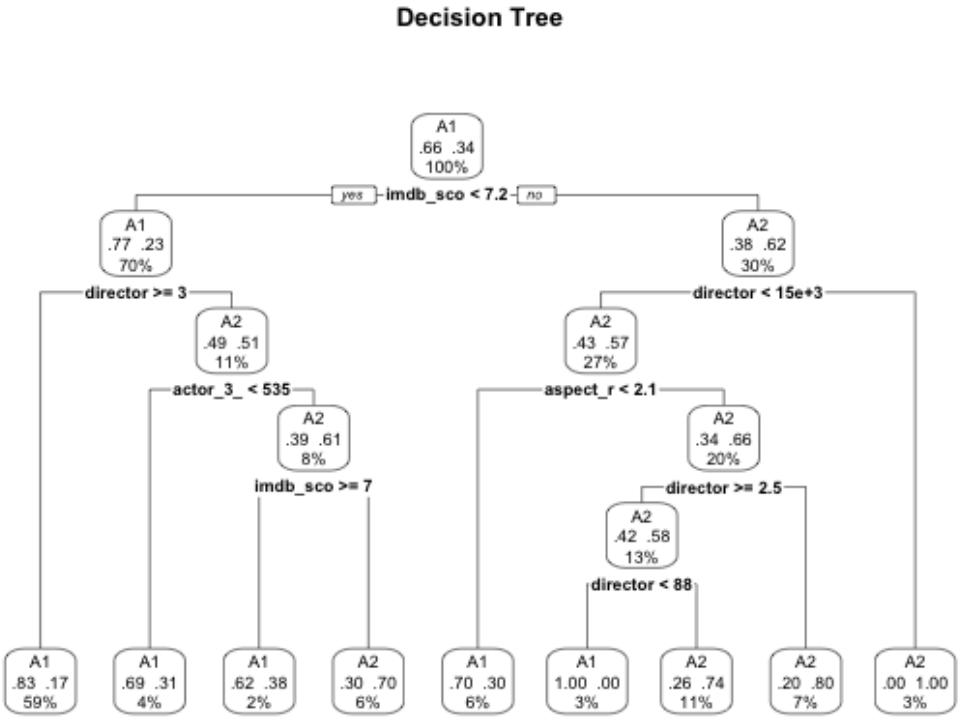
Showing 1 to 14 of 157 entries

6.1.2 Histograms

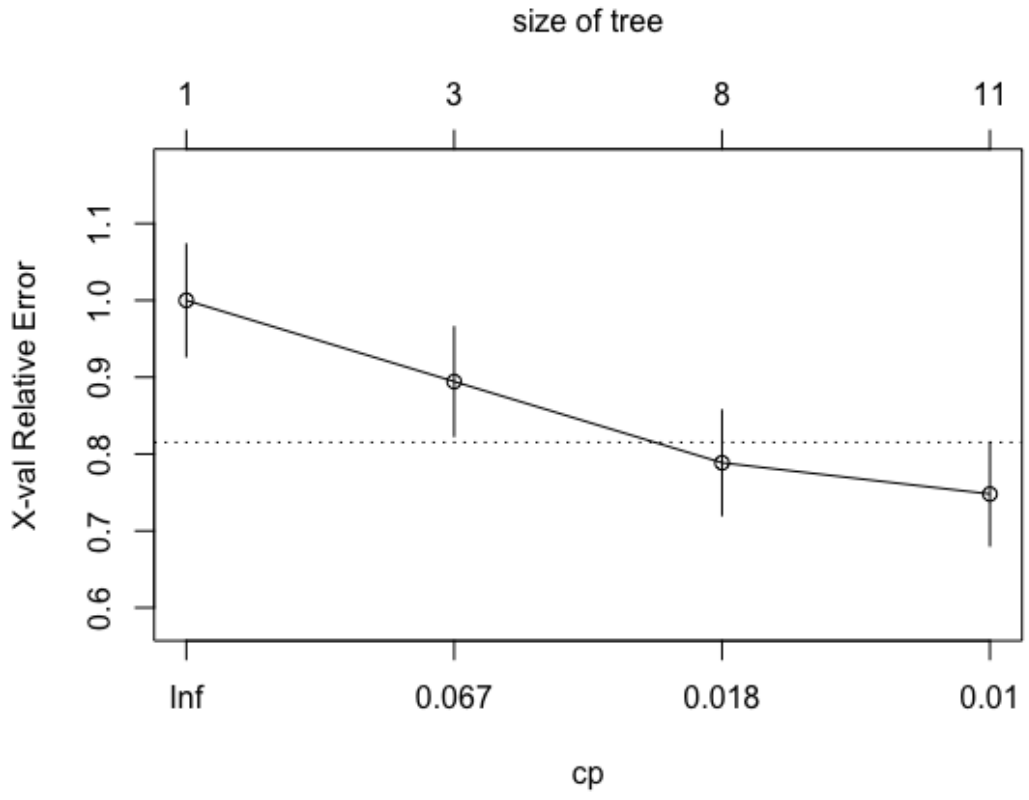




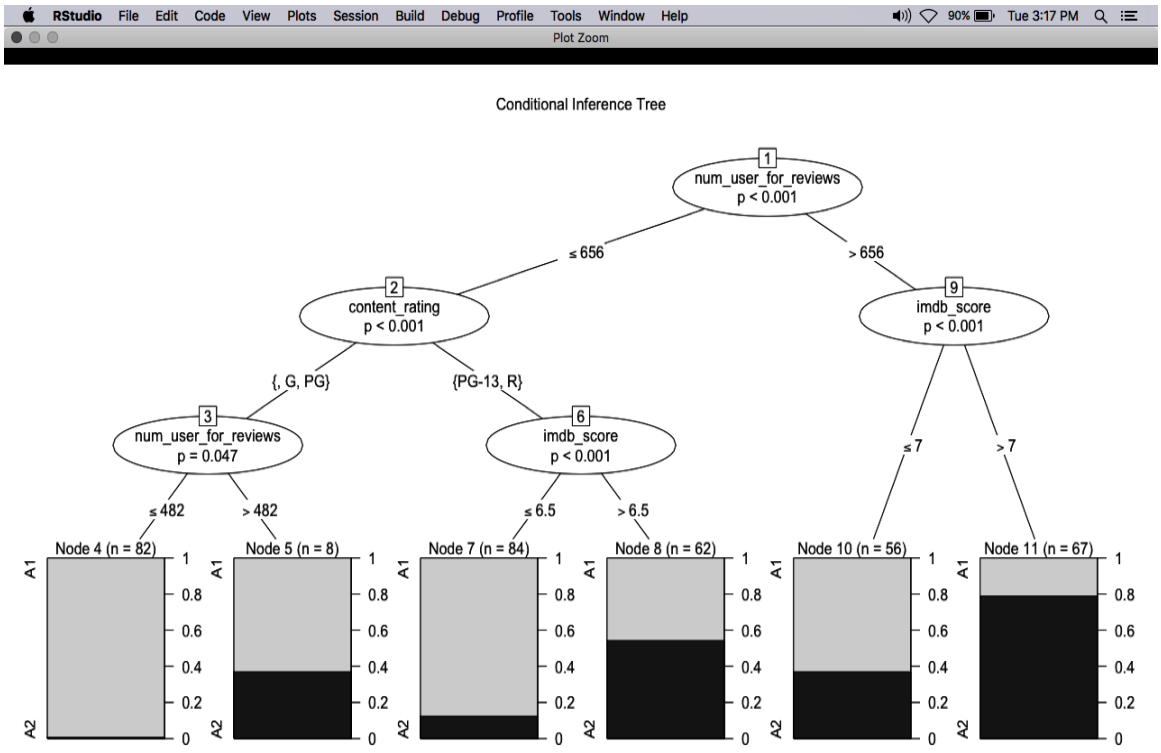
6.1.3 Decision Tree:



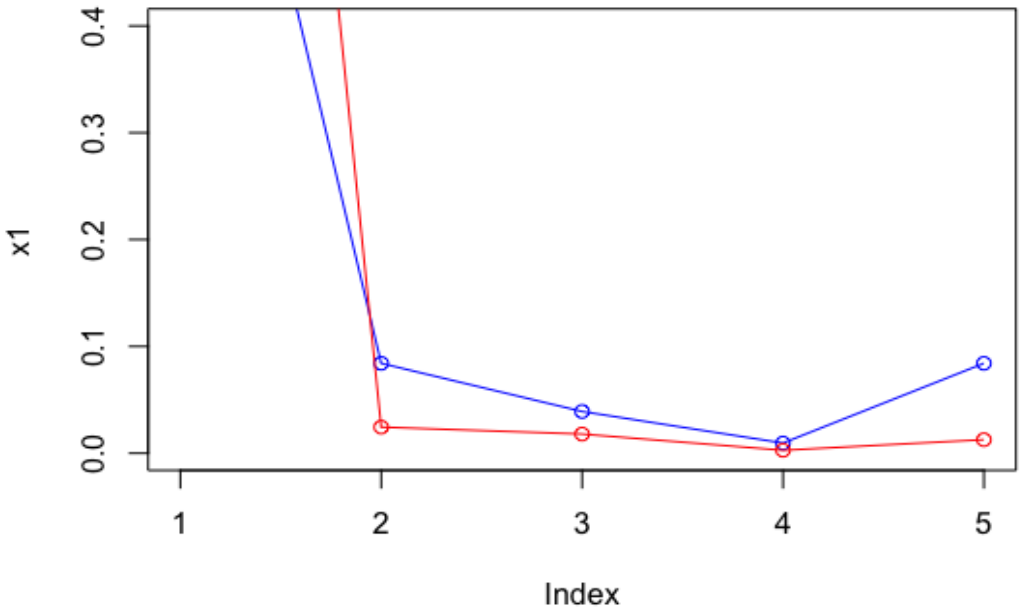
6.1.4 Dtree Graph



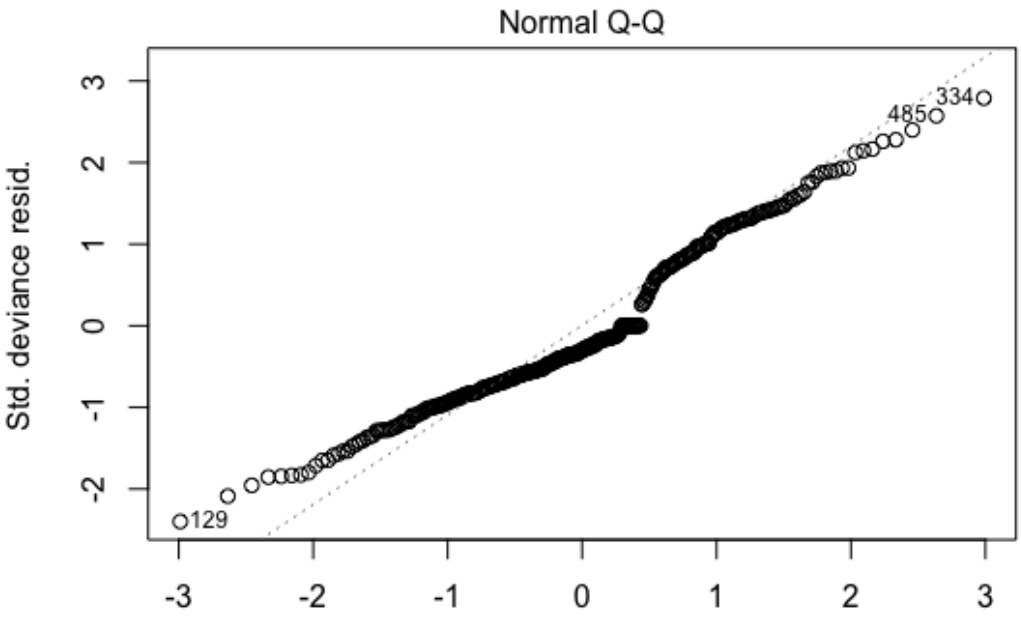
6.1.5 Conditional Inference Tree

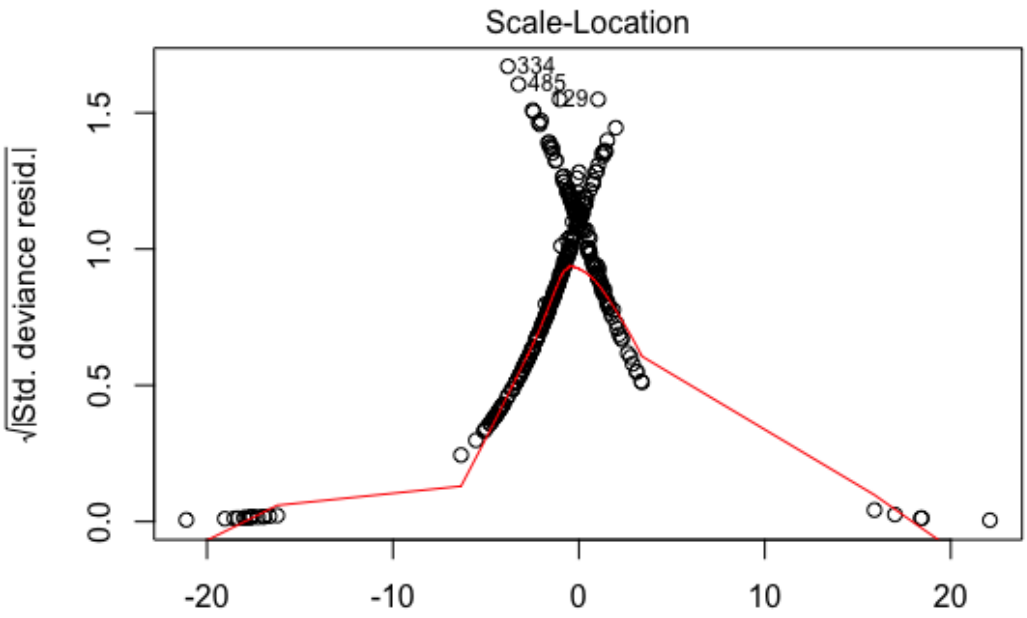


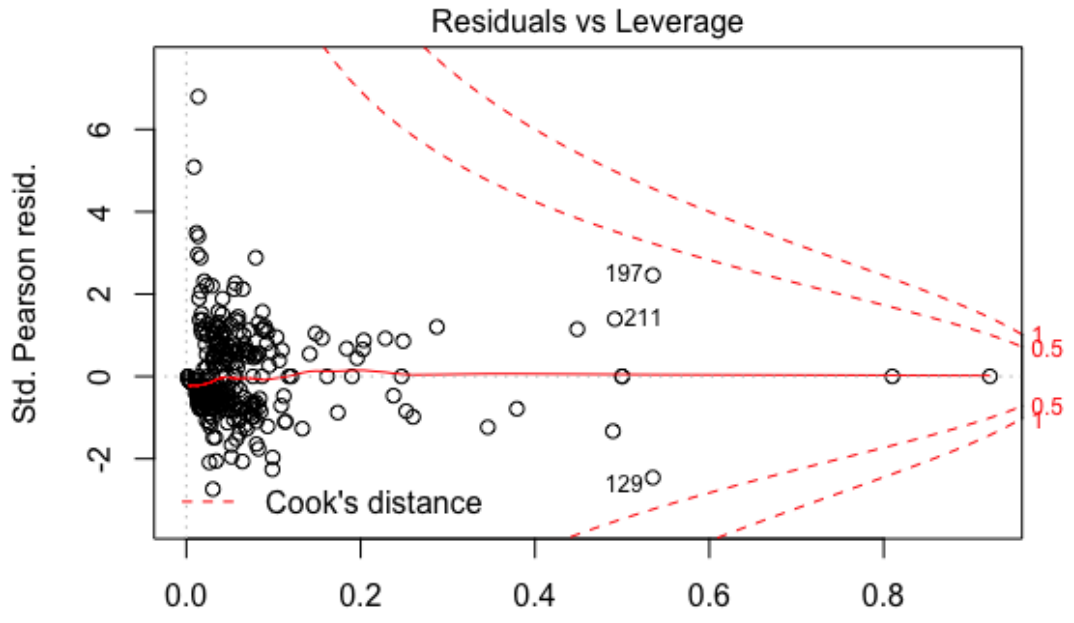
6.1.6 Lorenz curve

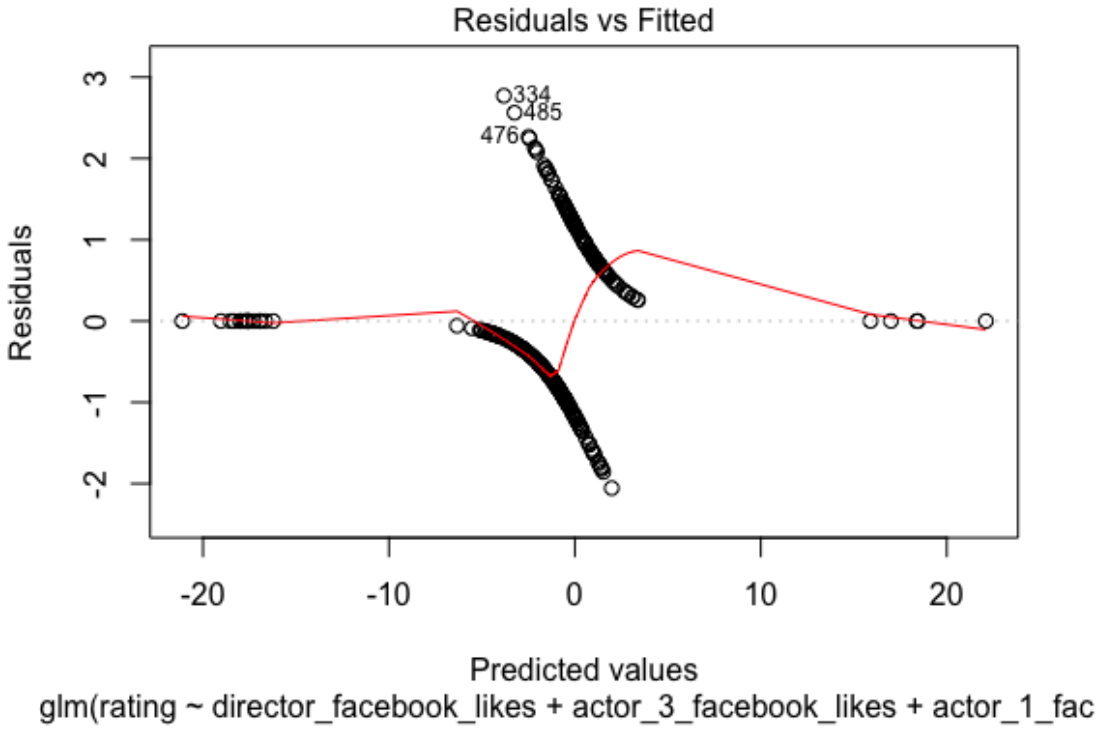


6.1.6 Logistic Regression:

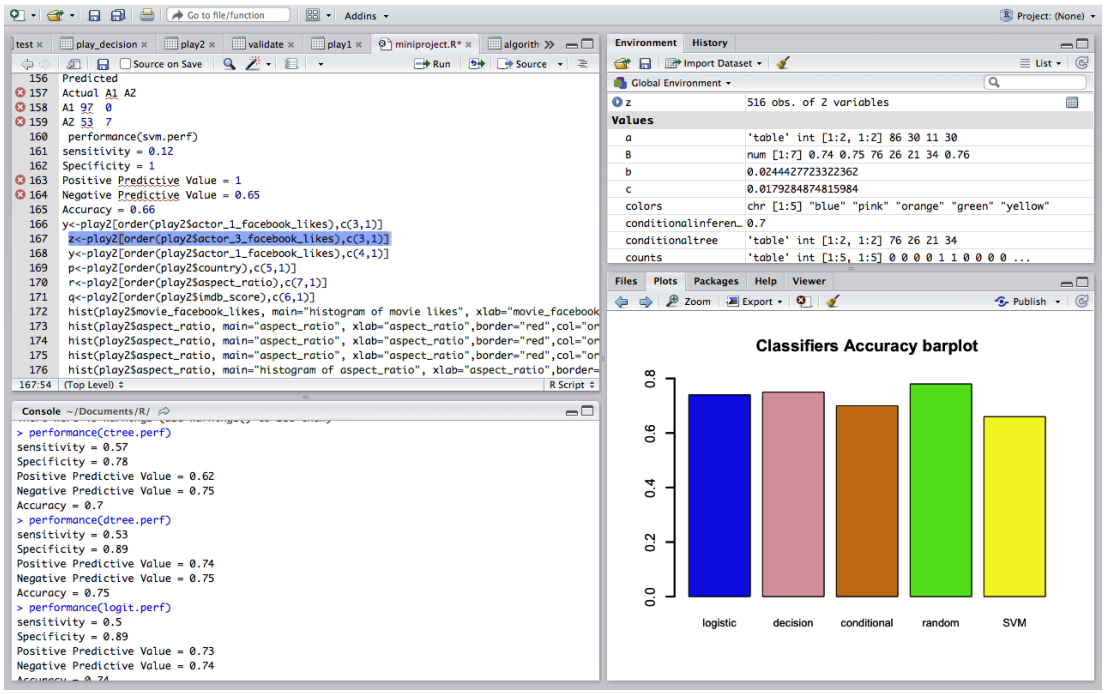


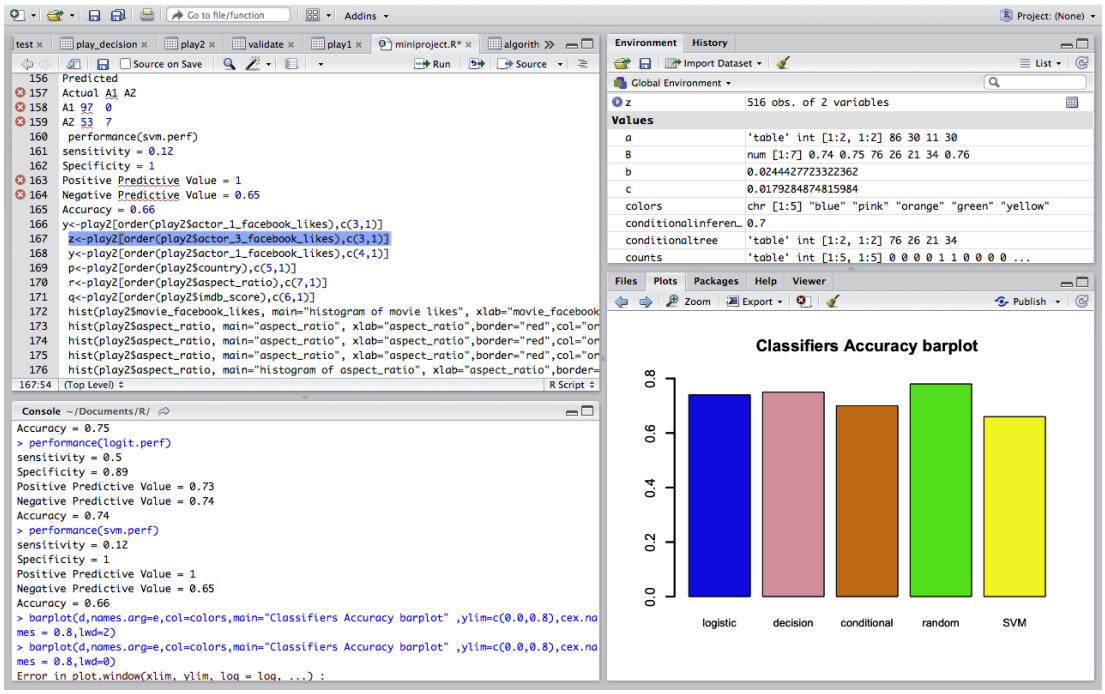

Scale-Location
334
485
129
√|Std. deviance resid.|
0.0
0.5
1.0
1.5
-20
-10
0
10
20
Predicted values
glm(rating ~ director_facebook_likes + actor_3_facebook_likes + actor_1_fac


Residuals vs Leverage
Std. Pearson resid.
-2
0
2
4
6
197
211
129
Cook's distance
1
0.5
0.5
1
0.0
0.2
0.4
0.6
0.8
Leverage
glm(rating ~ director_facebook_likes + actor_3_facebook_likes + actor_1_fac

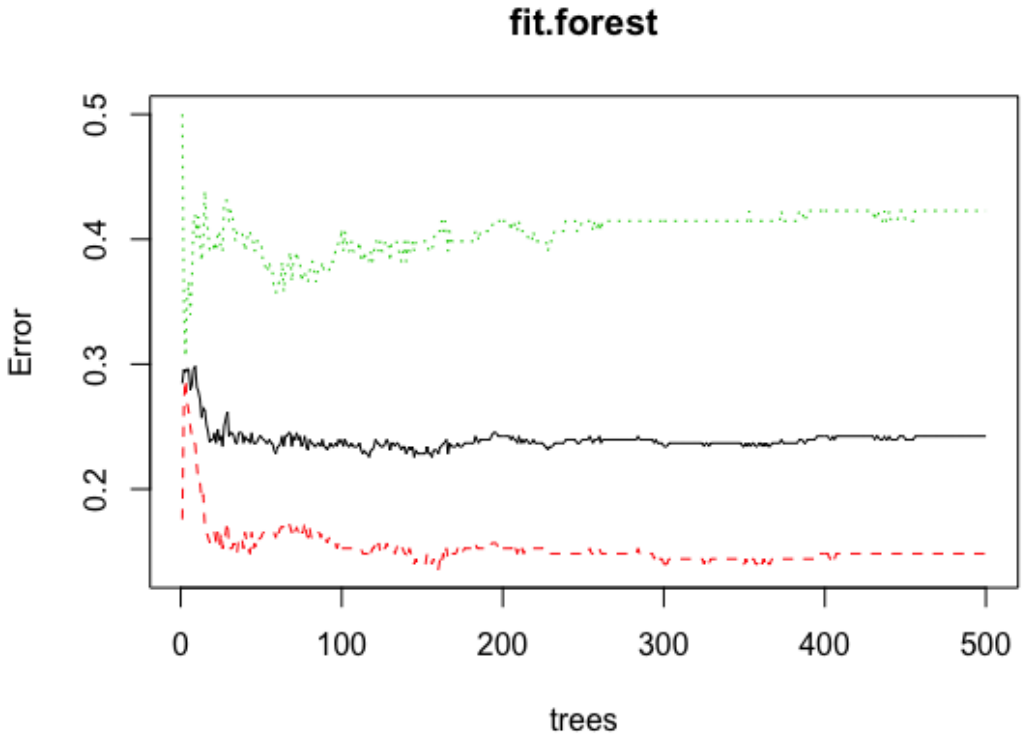


Performance Analysis:

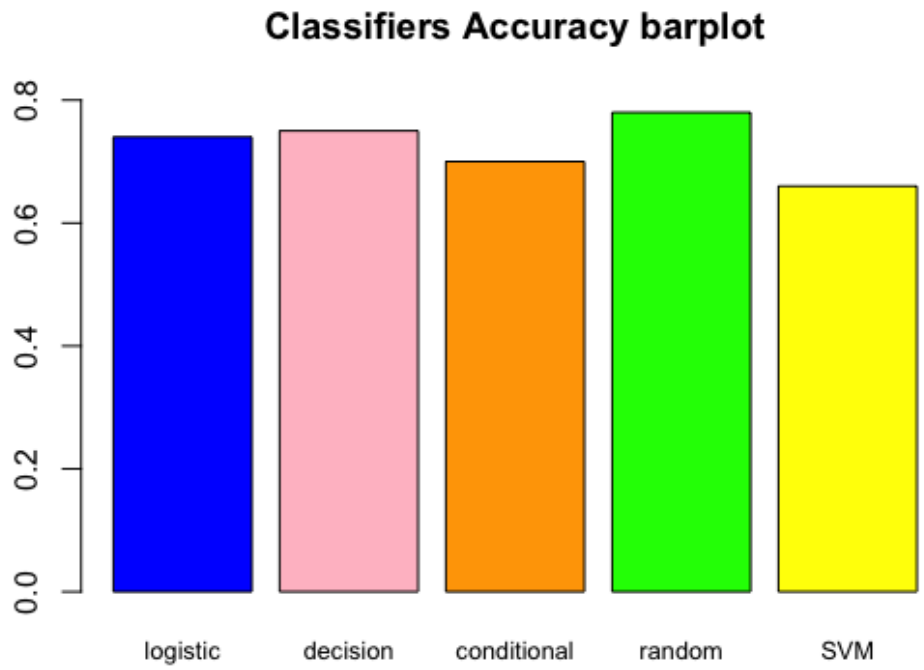




6.1.7 Random Forest



6.1.8 Bar graph



7. CONCLUSION AND FUTURE WORK:

7.1 Conclusion

we took the features that have a highly impact on the movie ratings and performed data preprocessing. This is done only after required feature selection from the IMDB is done. Derived results can be used by user to predict rating of a movie. Data mining techniques are applied on the taken dataset IMDB. Performance analysis of classical decision tree, conditional Inference tree , random Forest , linear regression and support vector machine is calculated. Out of these all randomForest gets the highest accuracy. As we know the classifier with more accuracy will be within realm of usage. In this by feature selection important features that highly effect rating are analyzed which can be used for reference. So the one with the highest accuracy can be used for reference by the user in future.

7.2 Future Work

Although the proposed techniques are very effective based on the studies, further improvements can still be made. Creating a GUI and integrating them with the new packages can bring changes to the way that the implementation of the system works.

8 REFERENCES:

[1] A. Tripathi and S. K. Trivedi, "Sentiment analyis of Indian movie review with various feature selection techniques," *2016 IEEE International Conference on Advances in Computer Applications (ICACA)*, Coimbatore, 2016, pp. 181-185.

[2] Nithin VR, Pranav M, Sarath Babu PB, Lijiya A "Predicting Movie Success Based on IMDB Data"- *International Journal of Data Mining Techniques and Applications* ,June 2014.

[3] Muhammad Hassan Latif, Hammad Afzal "Prediction of Movies popularity Using Machine Learning Techniques" *International Journal of Computer Science and Network Security*, VOL.16 No.8, August 2016

[4] D. Mumtaz and B. Ahuja, "Sentiment analysis of movie review data using Senti-lexicon algorithm," *2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, Bangalore, 2016, pp. 592-597.