Movie databse QM project – MIM1617

January 4, 2017

## PART 1: EDA

## Load the data and rename it

movie\_metadata <- read.csv("C:/Users/Luu/Desktop/MIM/QM/data/movie\_metadata.csv", header=TRUE, na.strings = c("NA",""))

movie<-movie\_metadata

## Look at the structure of the dataset and transform to data.table for easy handling, load necessary packages

str(movie)

library(data.table)

movie=as.data.table(movie)  
library(ggplot2)

library(corrplot)

library(reshape2)

## 1. Movies of each country contributed to the dataset

sort(table(movie$country))

cumsum(sort(table(movie$country)))

## It can be seen that USA is the dominant with 3807 movies compares to cumsum 1236 movies of all others countries

## 2. The change of movie number thoughout the year

plot(table(movie$title\_year), col = "red", main = "Change of movie quantity through the years", xlab="year", ylab="Quantity")

Top 5 directors with most number of movies

top5.director=head(sort(table(movie$director\_name, useNA = "no"), decreasing=TRUE), n=5)  
top5.director

## 4. Movie with highest and lowest IMDB score

movie.naomit=na.omit(movie)  
movie.naomit[which.max(movie.naomit$imdb\_score),list(director\_name,gross,genres,movie\_title,language, country,content\_rating, title\_year, budget, imdb\_score)]

movie.naomit[which.min(movie.naomit$imdb\_score),list(director\_name,gross,genres,movie\_title,language, country,content\_rating, title\_year, budget, imdb\_score)]

summary(movie$imdb\_score)

## 5.Most popular actor and their mean IMDB scores

actor\_max<- melt(table(movie$actor\_1\_name))  
colnames(actor\_max)=c("Actor name","Frequency")  
actor\_max[which.max(actor\_max$Frequency),]

## Actor and mean imdb score

meanimdb=tapply(movie$imdb\_score, movie$actor\_1\_name, FUN=mean,na.rm=TRUE)  
b=table(movie$actor\_1\_name)  
c=cbind(meanimdb,b)  
d=c[c[,2]>5,]  
e=d[order(d[,1],decreasing=TRUE),]  
e=as.data.frame(e)  
e$actor.name=rownames(e)  
e1=e[1:20,]  
e1

## Draw the chart

colours=c("#727272", "#f1595f", "#79c36a", "#599ad3", "#f9a65a", "#9e66ab", "#cd7058", "#d77fb3","#727272", "#f1595f", "#79c36a", "#599ad3", "#f9a65a", "#9e66ab", "#cd7058", "#d77fb3","#f9a65a", "#9e66ab", "#cd7058", "#d77fb3")  
ggplot(e1, aes(x=reorder(actor.name,-meanimdb),y=meanimdb, fill=actor.name))+geom\_bar(stat="identity")+coord\_flip()+scale\_fill\_manual(values=colours)+ geom\_text(aes(label=round(meanimdb,2)), vjust=0.8, hjust=1.1, colour="white") + ggtitle("Top 20 actor mean imdb score")

Information about budget

summary(movie$budget)

## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's   
## 2.180e+02 6.000e+06 2.000e+07 3.975e+07 4.500e+07 1.222e+10 492

## Find the highest budget movie

movie[which.max(movie$budget),]

## It can be seen that this is a Korean movie, and through google, we see that it is counted in KRW, not USD. Therefore, the following analysis relating to money will consider only US and UK movie which is the currency mainly in USD

USUK=subset(movie, country=="USA"| country=="UK")

## top 10 highest budget movies

head(USUK[order(USUK$budget, decreasing= T),list(director\_name,gross,genres,movie\_title,language, country,content\_rating, title\_year, budget, imdb\_score)], n = 10)

## top 10 lowest budget movie

head(USUK[order(USUK$budget, decreasing= F),list(director\_name,gross,genres,movie\_title,language, country,content\_rating, title\_year, budget, imdb\_score)], n = 10)

## => The score is not bad comparing to mean and median

## Budget change through the years

BG=tapply(USUK$budget, USUK$title\_year, FUN=mean, na.rm=TRUE)  
BG=melt(BG)   
colnames(BG)=c("Year", "Average budget")

ggplot(BG,aes(x=`Year`,y=`Average budget`))+geom\_line(aes(color="red"))+ ggtitle("Average budget change through the year")

through the year.

library(ggplot2)  
Score\_plot=ggplot(movie,aes(x=title\_year, y=imdb\_score, color=content\_rating))+geom\_point() + ggtitle("IMDB score change through the year")  
Score\_plot

## Warning: Removed 108 rows containing missing values (geom\_point).

content\_rating

bp1<-ggplot(movie,aes(x=content\_rating,y=imdb\_score))+geom\_boxplot(color="black",fill="green")+ggtitle("IMDB score by content rating")  
bp1

corrplot(correlation, method="pie")

## Correlation between Profit and budget

USUK$profit = USUK$gross - USUK$budget  
 ggplot(USUK, aes(x=profit, y=budget, color=title\_year))+geom\_point()+geom\_smooth()+scale\_colour\_gradient(low = "green",high="red")+ggtitle("Investment vs profit")

## `geom\_smooth()` using method = 'gam'

**##II PCA AND CLUSTER ANALYSIS**

## PCA We try to find a lower dimensional represenation of quantitative variable in the dataset Choose the quantitative variable in the dataset. We omit some variable relate to single cast facebook like and choose 2 variables represent for social popularity of the movie, which is cast total facebook like and movie facebook like. Finally we have 7 variables to perform PCA. Another notation is that the quantitative variables relate to money, so we subset from USUK data only to avoid difference in currency. Scale option is True because data of different nature

movie.number=na.omit(USUK[,list(movie\_title,num\_critic\_for\_reviews,gross, num\_voted\_users, cast\_total\_facebook\_likes,budget,imdb\_score, movie\_facebook\_likes)])

pr.out=prcomp(movie.number[,2:8], scale=TRUE)  
pr.out

## We try to define the Variance explained by these 7 PC, by computing the PVE and chart them

variance=pr.out$sdev^2  
pve=variance/sum(variance)  
cumsum(pve)

## [1] 0.4859353 0.6382496 0.7629171 0.8709871 0.9280035 0.9673253 1.0000000

plot(cumsum(pve), type="b", xlab="Number of PC", ylab="Cummulative Variance explained", main="Variance explain")

## The dimension doesnt reduce drammatically , at least 5 first PC explain more than 90% of the whole data = > All variable are quite important and bring different information ## Try the biplot of the PC1 and 2

biplot(pr.out, xlabs=rep(".", nrow(movie.number[,2:8])), main = "Biplot for first 2 PC")

## Customize the biplot. Look at the variables imdb\_score and Budget

biplot(pr.out$x,pr.out$rotation[c(5,6),])

Call out some items that have distinct Score

movie.number[c(1650,272, 10,27,8,24,25,1623,90),list(movie\_title, budget, imdb\_score, gross)]

## Analyse: The direction of arrow, and soome clear item like 1650.272.10....Try to plot the first 2 PC on the 2d plot to see if any cluster can be seen

Pc= as.data.frame(pr.out$x)  
ggplot(Pc,aes(x=PC1,y=PC2))+geom\_point(aes(color=PC3),alpha=0.3)+ggtitle("Plot of first 2 PC")

## No distinct group can be seen from this. Try to give some information from the loading factors and scores.

## K MEANS CLUSTERING

## Identify how many cluster we should use:

## Scale the data

movie.number.scale=scale(movie.number[,2:8])

## Check the totwithinss rate

ratiowss=vector()  
for (i in 2:10){km=kmeans(movie.number.scale,i,iter.max=30,nstart=50)  
ratiowss[i]=km$tot.withinss/km$totss}

## Plot the result

dt<-data.table("K"=2:10, "ratiowss"=ratiowss[2:10])  
plot(dt, type="b", main="How many clusters is good?")

## The result doesnt indicate good result. Try with h-clust to see how many cluster should be used?

hclust.s=hclust(dist(movie.number.scale), method="single")  
plot(hclust.s)

hclust.c=hclust(dist(movie.number.scale), method="complete")  
plot(hclust.c)

hclust.a=hclust(dist(movie.number.scale), method="average")

## Plot the dendrogram

plot(hclust.a)

## We will go deeper into average method which produce a quite nice tree ## Convert hclust object into dendrogram object

hclust.a1=as.dendrogram(hclust.a)

## Cut the tree at a specific height for easy interpretation

plot(cut(hclust.a1, h = 12)$upper, main = "Upper tree of cut at h=12")

## From above result, 5 clusters will be a compromising choice to see ## Lets use the 5 clusters from h clust

h.clust.5=cutree(hclust.a,k=5)  
table(h.clust.5)

## h.clust.5  
## 1 2 3 4 5   
## 13 3381 1 2 1

## Compute k=5 by K-means

km=kmeans(movie.number.scale,5,iter.max=30,nstart=50)  
km

table(km$cluster,h.clust.5)

## h.clust.5  
## 1 2 3 4 5  
## 1 0 1500 0 0 0  
## 2 13 139 1 0 1  
## 3 0 888 0 0 0  
## 4 0 433 0 2 0  
## 5 0 421 0 0 0

The result of h-clust and k-clust are very different. K cluster: From the mean of variables can intepret something? H-cluster: There is 1 very big group and other small group. There may be no clear grouping available using this method However, the observation 1623 and 90 is something different Call out the item 1623 and 90 for analysis

movie.number[c(1623,90),]

Comput the summary of movie.number to compare if any thing can be seen

summary(movie.number)

Also observation from K-cluster that first group is quite different, all the mean are far higher. Let's try to analyse the first group

movie.number$group=km$cluster  
k.group1=movie.number[group==1]  
summary(k.group1)

**## III TREE**

Building a classification tree for imdb score Because the variable budget and gross will also be used in this prediction, the data used here will be USUK only with the same currency difference reason Before compute the tree, we do some data cleaning. Variable relate to actor name, movie title, director name, plot keyword, website link will be omitted, either because they doesnt give input for prediction or their levels are too many which can not be executed in R. Variable language are transformed into LangEng holds 2 value TRUE or FALSE Because Random forest doesnt handle the NA value, so we remove all NA from now to run all the tree, bagging and random forest, because later on, if the number of observation between trees and random forest are not equal, then it wont be equal to compare the misclassification rate.

USUK$langEng=USUK$language=="English"  
USUK$country <- factor(USUK$country)  
USUK$qualityrating=cut(USUK$imdb\_score, breaks=c(0,5,6.5,8,10), include.lowest = TRUE, labels=c("Bad","Average","Good","Great"))  
USUK.datatree=na.omit(USUK[, list(color, num\_critic\_for\_reviews, duration, director\_facebook\_likes, actor\_3\_facebook\_likes, actor\_1\_facebook\_likes,gross, num\_voted\_users, cast\_total\_facebook\_likes,facenumber\_in\_poster, num\_user\_for\_reviews, langEng, country, content\_rating,budget, title\_year, actor\_2\_facebook\_likes, qualityrating, aspect\_ratio, movie\_facebook\_likes)])

Divide the data into 1000 for test set and balance for training set

library(tree)

## Warning: package 'tree' was built under R version 3.3.2

set.seed(1)  
test=sample(nrow(USUK.datatree), 1000)  
movie.train=USUK.datatree[-test,]  
movie.test=USUK.datatree[test,]

Build first regression tree

r1.tree=tree(qualityrating~.,movie.train)  
r1.tree

summary(r1.tree)

Plot the tree

plot(r1.tree,lwd=3)  
text(r1.tree,pretty=0,cex=1,col="blue")

Check with the test set

r1.tree.predict=predict(r1.tree, movie.test,type="class")  
  
qualityrating.test=USUK.datatree$qualityrating[test]  
  
table(r1.tree.predict, qualityrating.test)

(84+194+2+8+64+18+3)/nrow(movie.test)

## [1] 0.373

[1] 0.373 Grow a bigger tree

setup1=tree.control(nrow(USUK.datatree),mincut =5,minsize =10,mindev =0.001)  
r2.tree=tree(qualityrating~.,data=movie.train,control=setup1)  
summary(r2.tree)

Plot the r2.tree

plot(r2.tree,lwd=3)  
text(r2.tree,pretty=0,cex=1.5,col="blue")

Predict on test set the big tree

r2.tree.predict=predict(r2.tree, movie.test,type="class")  
table(r2.tree.predict, qualityrating.test)

(55+13+41+117+20+114+15+5)/nrow(movie.test)

## [1] 0.38

[1] 0.38 WOrse Cross validation for the tree to choose the best number of nod

set.seed (2)  
cv.movie =cv.tree(r2.tree ,FUN=prune.misclass )  
cv.movie

plot(cv.movie$size, cv.movie$dev, type="b")

From the CV result, we see that tree with 11 nods is as good as tree with 132 nods. so we gonna prune the tree with 11 nods

prune.r2.tree=prune.misclass(r2.tree,best =11)

Plot it

plot(prune.r2.tree,type="uniform",lwd=2)  
text(prune.r2.tree ,pretty =0,cex=0.9,col="blue")

Again predict this tree on test set

prune.r2.tree.predict=predict(prune.r2.tree, movie.test,type="class")  
table(prune.r2.tree.predict, qualityrating.test)

(83+156+1+9+85+19+3)/nrow(movie.test)

## [1] 0.356

[1] 0.356 Improvement!!! ## Bagging

library (randomForest)

set.seed (3)  
bag.movie=randomForest(qualityrating~.,movie.train,mtry=19,importance =TRUE,ntree=500)  
bag.movie

Compare with the test set

bag.pred=predict(bag.movie, movie.test,type="class")  
table(bag.pred, qualityrating.test)

(10+61+112+10+92+13+3)/nrow(movie.test)

## [1] 0.301

[1] 0.301 even better RAndom forest

random.forest=randomForest(qualityrating~.,movie.train,mtry=4,importance =TRUE,ntree=500)  
random.forest

Predict with test set

rf.pred=predict(random.forest, movie.test,type="class")  
table(rf.pred, qualityrating.test)

(4+71+101+14+98+18+4)/nrow(movie.test)

## [1] 0.31

REsult [1] 0.31 Not as good as bagging Try to see the importance of variables in splitting.

varImpPlot(random.forest, col="blue", lwd=5)