Final_Project

Haw-Jan Hwang

October 14, 2019

Packages

Import Data

There are total 20 columns and 3338 rows and most of the features are numeric data

```
df = read.csv('SkillCraft.csv')
str(df)
```

```
## 'data.frame':    3338 obs. of  20 variables:
##  $ GameID              : int  52 55 56 57 58 60 61 72 77 81 ...
##  $ LeagueIndex         : int  5 5 4 3 3 2 1 7 4 4 ...
##  $ Age                 : int  27 23 30 19 32 27 21 17 20 18 ...
##  $ HoursPerWeek        : int  10 10 10 20 10 6 8 42 14 24 ...
##  $ TotalHours          : int  3000 5000 200 400 500 70 240 10000 2708 800 ...
##  $ APM                 : num  144 129 70 108 123 ...
##  $ SelectByHotkeys     : num  0.00352 0.0033 0.0011 0.00103 0.00114 ...
##  $ AssignToHotkeys     : num  0.00022 0.000259 0.000336 0.000213 0.000327 ...
##  $ UniqueHotkeys       : int  7 4 4 1 2 2 6 6 2 8 ...
##  $ MinimapAttacks      : num  1.10e-04 2.94e-04 2.94e-04 5.33e-05 0.00 ...
##  $ MinimapRightClicks  : num  0.000392 0.000432 0.000461 0.000543 0.001329 ...
##  $ NumberOfPACs        : num  0.00485 0.00431 0.00293 0.00378 0.00237 ...
##  $ GapBetweenPACs      : num  32.7 32.9 44.6 29.2 22.7 ...
##  $ ActionLatency       : num  40.9 42.3 75.4 53.7 62.1 ...
##  $ ActionsInPAC        : num  4.75 4.84 4.04 4.92 9.37 ...
##  $ TotalMapExplored    : int  28 22 22 19 15 16 15 45 29 27 ...
##  $ WorkersMade         : num  0.001397 0.001193 0.000745 0.000426 0.001174 ...
##  $ UniqueUnitsMade     : int  6 5 6 7 4 6 5 9 7 6 ...
##  $ ComplexUnitsMade    : num  0 0 0 0 0 ...
##  $ ComplexAbilitiesUsed: num  0.00 2.08e-04 1.89e-04 3.84e-04 1.93e-05 ...
```

Attribute Information

1. GameID: Unique ID number for each game (integer)
2. LeagueIndex: Bronze, Silver, Gold, Platinum, Diamond, Master, GrandMaster, and Professional leagues coded 1-7 (Ordinal)
3. Age: Age of each player (integer)
4. HoursPerWeek: Reported hours spent playing per week (integer)
5. TotalHours: Reported total hours spent playing (integer)
6. APM: Action per minute (continuous)
7. SelectByHotkeys: Number of unit or building selections made using hotkeys per timestamp (continuous)

8. AssignToHotkeys: Number of units or buildings assigned to hotkeys per timestamp (continuous)
9. UniqueHotkeys: Number of unique hotkeys used per timestamp (continuous)
10. MinimapAttacks: Number of attack actions on minimap per timestamp (continuous)
11. MinimapRightClicks: number of right-clicks on minimap per timestamp (continuous)
12. NumberOfPACs: Number of PACs per timestamp (continuous) (A PAC is when one changes screen location and performs 1+ actions before changing screen location again to repeat.)
13. GapBetweenPACs: Mean duration in milliseconds between PACs (continuous)
14. ActionLatency: Mean latency from the onset of a PACs to their first action in milliseconds (continuous)
15. ActionsInPAC: Mean number of actions within each PAC (continuous)
16. TotalMapExplored: The number of 24x24 game coordinate grids viewed by the player per timestamp (continuous)
17. WorkersMade: Number of SCVs, drones, and probes trained per timestamp (continuous)
18. UniqueUnitsMade: Unique unites made per timestamp (continuous)
19. ComplexUnitsMade: Number of ghosts, infestors, and high templars trained per timestamp (continuous)
20. ComplexAbilitiesUsed: Abilities requiring specific targeting instructions used per timestamp (continuous)

Data Summary

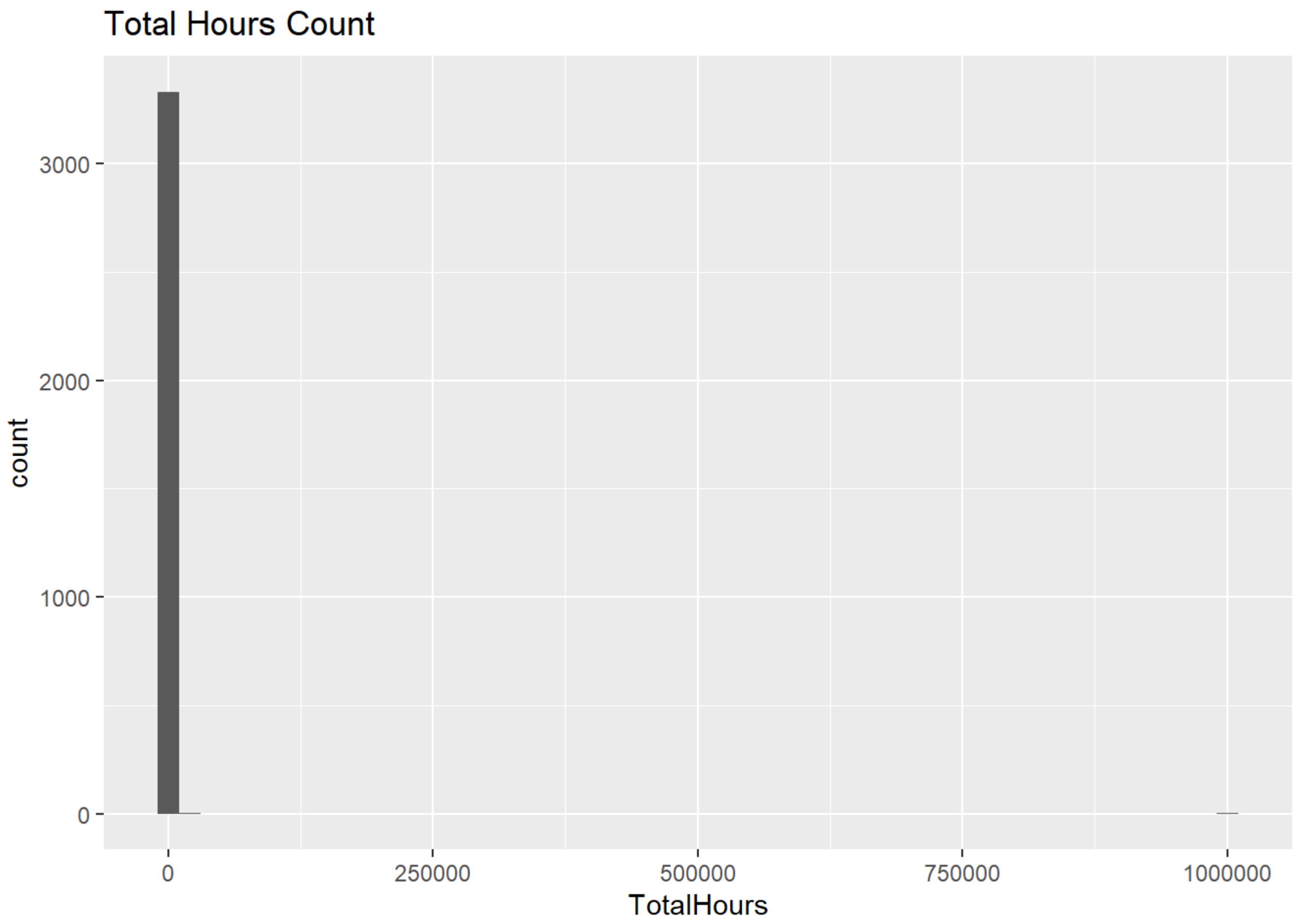
From the code below, we can see the preliminary summary of all the features.

```
summary(df)
```

```
##      GameID      LeagueIndex         Age         HoursPerWeek
##  Min.   :  52   Min.   :1.000   Min.   :16.00   Min.   :  0.00
##  1st Qu.:2423   1st Qu.:3.000   1st Qu.:19.00   1st Qu.:  8.00
##  Median :4788   Median :4.000   Median :21.00   Median : 12.00
##  Mean   :4720   Mean   :4.121   Mean   :21.65   Mean   : 15.91
##  3rd Qu.:6995   3rd Qu.:5.000   3rd Qu.:24.00   3rd Qu.: 20.00
##  Max.   :9271   Max.   :7.000   Max.   :44.00   Max.   :168.00
##    TotalHours             APM         SelectByHotkeys
##  Min.   :      3.0   Min.   : 22.06   Min.   :0.000000
##  1st Qu.:    300.0   1st Qu.: 79.23   1st Qu.:0.001245
##  Median :    500.0   Median :107.07   Median :0.002445
##  Mean   :    960.4   Mean   :114.58   Mean   :0.004023
##  3rd Qu.:    800.0   3rd Qu.:140.16   3rd Qu.:0.004945
##  Max.   :1000000.0   Max.   :389.83   Max.   :0.043088
##  AssignToHotkeys     UniqueHotkeys    MinimapAttacks
##  Min.   :0.0000000   Min.   : 0.000   Min.   :0.000e+00
##  1st Qu.:0.0002017   1st Qu.: 3.000   1st Qu.:0.000e+00
##  Median :0.0003487   Median : 4.000   Median :3.864e-05
##  Mean   :0.0003641   Mean   : 4.316   Mean   :9.378e-05
##  3rd Qu.:0.0004929   3rd Qu.: 6.000   3rd Qu.:1.134e-04
##  Max.   :0.0016483   Max.   :10.000   Max.   :3.019e-03
##  MinimapRightClicks   NumberOfPACs      GapBetweenPACs    ActionLatency
##  Min.   :0.0000000   Min.   :0.000679   Min.   :  6.667   Min.   : 24.63
##  1st Qu.:0.0001388   1st Qu.:0.002743   1st Qu.: 29.327   1st Qu.: 50.89
##  Median :0.0002784   Median :0.003376   Median : 37.059   Median : 61.30
##  Mean   :0.0003802   Mean   :0.003433   Mean   : 40.714   Mean   : 64.21
##  3rd Qu.:0.0005076   3rd Qu.:0.004003   3rd Qu.: 48.510   3rd Qu.: 74.03
##  Max.   :0.0036877   Max.   :0.007971   Max.   :237.143   Max.   :176.37
##   ActionsInPAC    TotalMapExplored  WorkersMade        UniqueUnitsMade
##  Min.   : 2.039   Min.   : 5.00    Min.   :7.698e-05   Min.   : 2.000
##  1st Qu.: 4.262   1st Qu.:17.00    1st Qu.:6.818e-04   1st Qu.: 5.000
##  Median : 5.087   Median :22.00    Median :9.042e-04   Median : 6.000
##  Mean   : 5.267   Mean   :22.12    Mean   :1.031e-03   Mean   : 6.541
##  3rd Qu.: 6.027   3rd Qu.:27.00    3rd Qu.:1.258e-03   3rd Qu.: 8.000
##  Max.   :18.558   Max.   :58.00    Max.   :5.149e-03   Max.   :13.000
##  ComplexUnitsMade    ComplexAbilitiesUsed
##  Min.   :0.000e+00   Min.   :0.000e+00
##  1st Qu.:0.000e+00   1st Qu.:0.000e+00
##  Median :0.000e+00   Median :2.043e-05
##  Mean   :5.998e-05   Mean   :1.419e-04
##  3rd Qu.:8.742e-05   3rd Qu.:1.823e-04
##  Max.   :9.023e-04   Max.   :3.084e-03
```

From the summary above and the graph, we can see there is 1 outlier (TotalHours): max is 100000 but the mean and the mediarn are much less (500, 960)

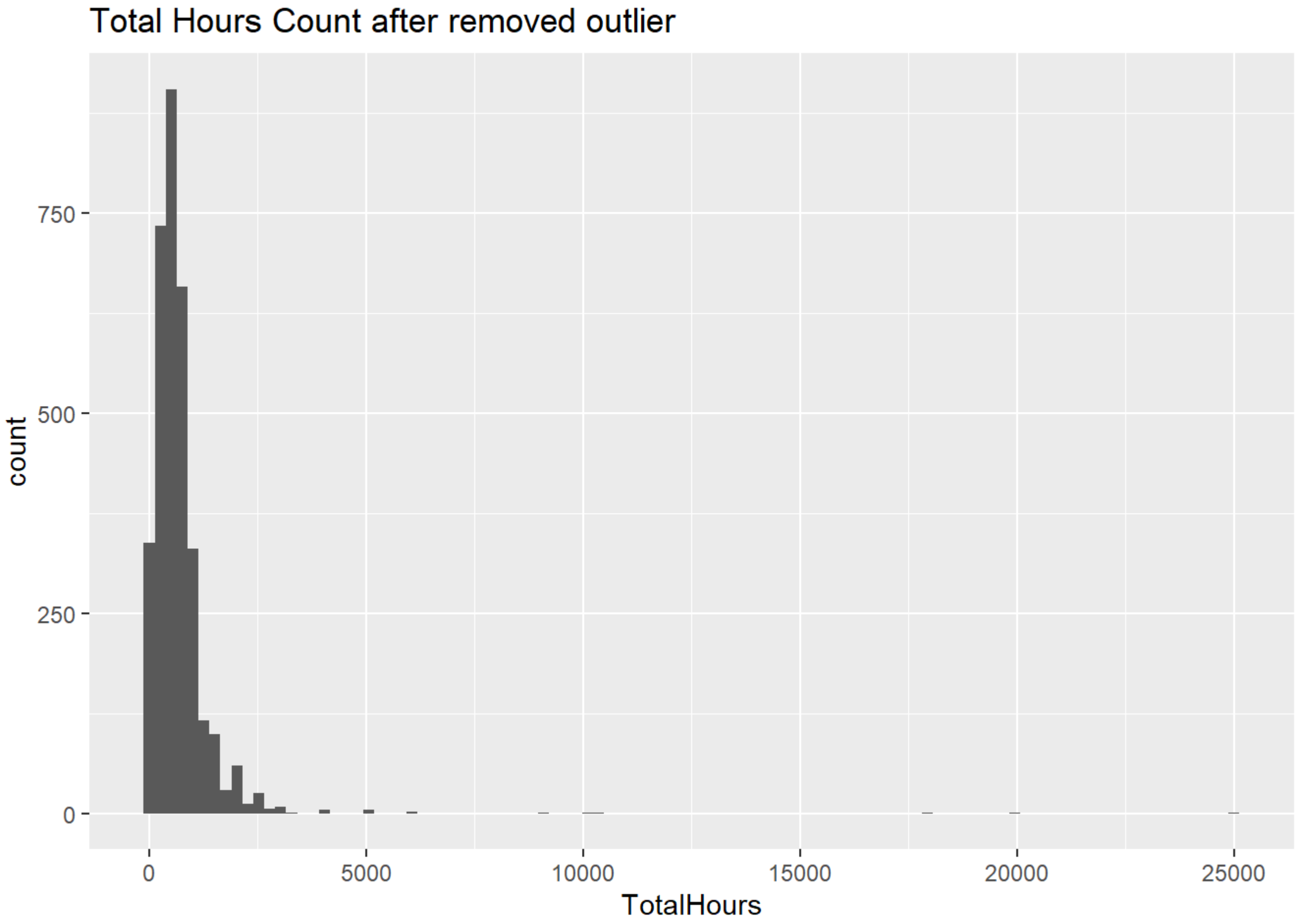
```
total_hours = ggplot(data=df,aes(x=TotalHours)) + ggtitle('Total Hours Count') +geom_histogram(b
ins=50)
print(total_hours)
```



#

Data Cleaning I remove the outlier and the GameID feature.

```
df = df[,-1]
df = df[df$TotalHours!=1000000,]
new_total_hours = ggplot(data=df,aes(x=TotalHours)) + ggtitle('Total Hours Count after removed o
utlier') + geom_histogram(bins=100)
print(new_total_hours)
```



Transform The LeagueName feature into Categorical variable

```
league = function(x){
    if (x==1) {return('Bronze')}
    else if (x==2) {return('Silver')}
    else if (x==3) {return('Gold')}
    else if (x==4) {return('Platinum')}
    else if (x==5) {return('Diamond')}
    else if (x==6) {return('Master')}
    else {return('Grand Master')}
}
df$LeagueName = sapply(df$LeagueIndex,league)
```

```
df <- na.omit(df)
```

Feature Group 1

```
group1<-df %>% group_by(LeagueIndex) %>% select(Age, HoursPerWeek, TotalHours, APM, ActionLatenc
y, GapBetweenPACs, ActionsInPAC) %>% summarise(avg_age = mean(Age), avg_Hours = mean(HoursPerWee
k), avgTotalHours = mean(TotalHours), avg_APM = mean(APM), avg_actionLatency = mean(ActionLatenc
y), avg_GapPacs= mean(GapBetweenPACs), avg_actionPac = mean(ActionsInPAC))
```

```
## Adding missing grouping variables: `LeagueIndex`
```

Feature Group 2

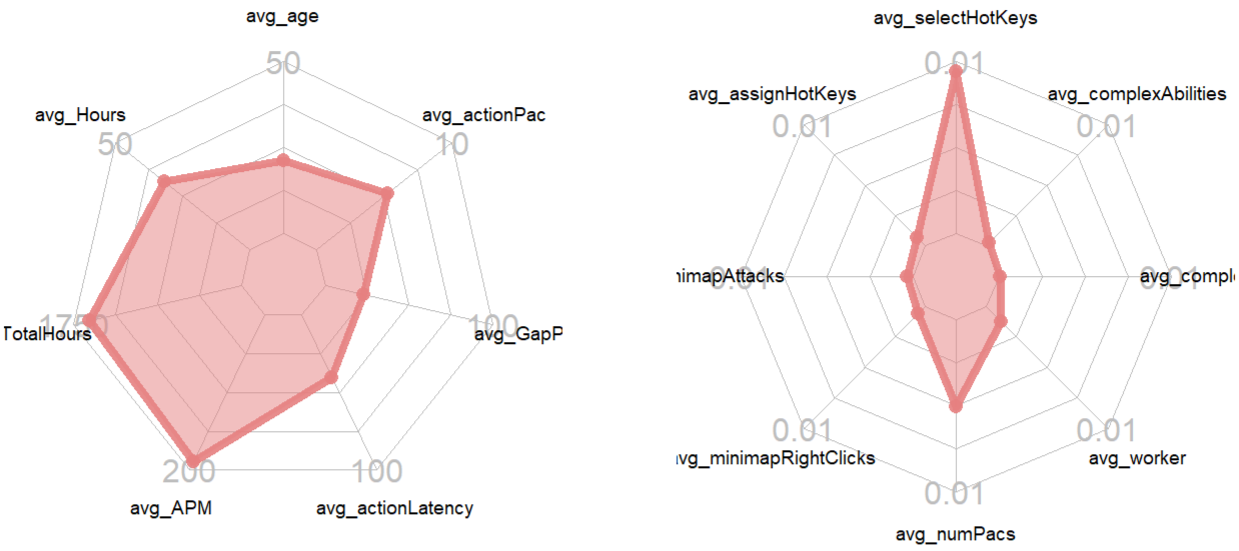
```
group2<-df %>% group_by(LeagueIndex) %>% select(SelectByHotkeys,AssignToHotkeys,UniqueHotkeys,MinimapAttacks, MinimapRightClicks,NumberOfPACs, TotalMapExplored, WorkersMade, UniqueUnitsMade, ComplexUnitsMade, ComplexAbilitiesUsed) %>% summarise(avg_selectHotKeys = mean(SelectByHotkeys), avg_assignHotKeys = mean(AssignToHotkeys), avg_minimapAttacks = mean(MinimapAttacks),avg_minimapRightClicks = mean(MinimapRightClicks), avg_numPacs = mean(NumberOfPACs), avg_worker = mean(WorkersMade), avg_complexUnit = mean(ComplexUnitsMade), avg_complexAbilities = mean(ComplexAbilitiesUsed))
```

```
## Adding missing grouping variables: `LeagueIndex`
```

```
maxGr1<-c(50,50,1750,200,100,100,10)
minGr1<-rep(0,7)
maxGr2<-c(.01,.01,.01,.01,.01,.01,.01,.01)
minGr2<-rep(0,11)
```

Grand Master

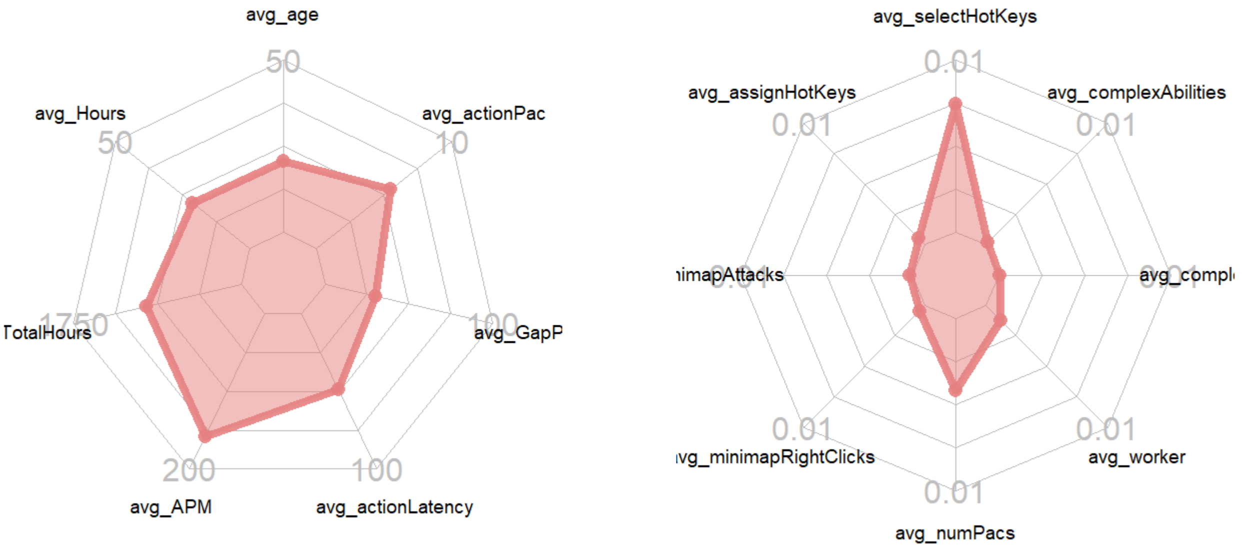
```
gr1GML<-rbind(maxGr1,minGr1,group1[7,2:8])
gr2GML<-rbind(maxGr2,minGr2,group2[7,2:9])

op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1GML  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.9,0.5,0.5,0.9) , pfcol=rgb(0.9,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2GML  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.9,0.5,0.5,0.9) , pfcol=rgb(0.9,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```



```
par(op)
```

Master

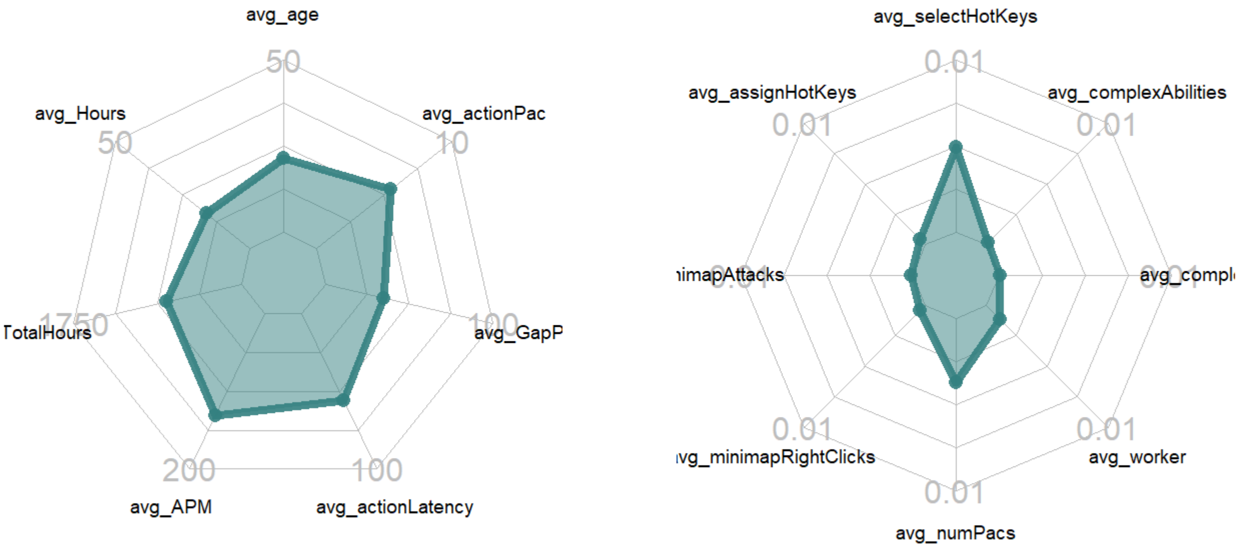
```
gr1ML<-rbind(maxGr1,minGr1,group1[6,2:8])
gr2ML<-rbind(maxGr2,minGr2,group2[6,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1ML  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.9,0.5,0.5,0.9) , pfcol=rgb(0.9,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2ML  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.9,0.5,0.5,0.9) , pfcol=rgb(0.9,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```



#

Diamond

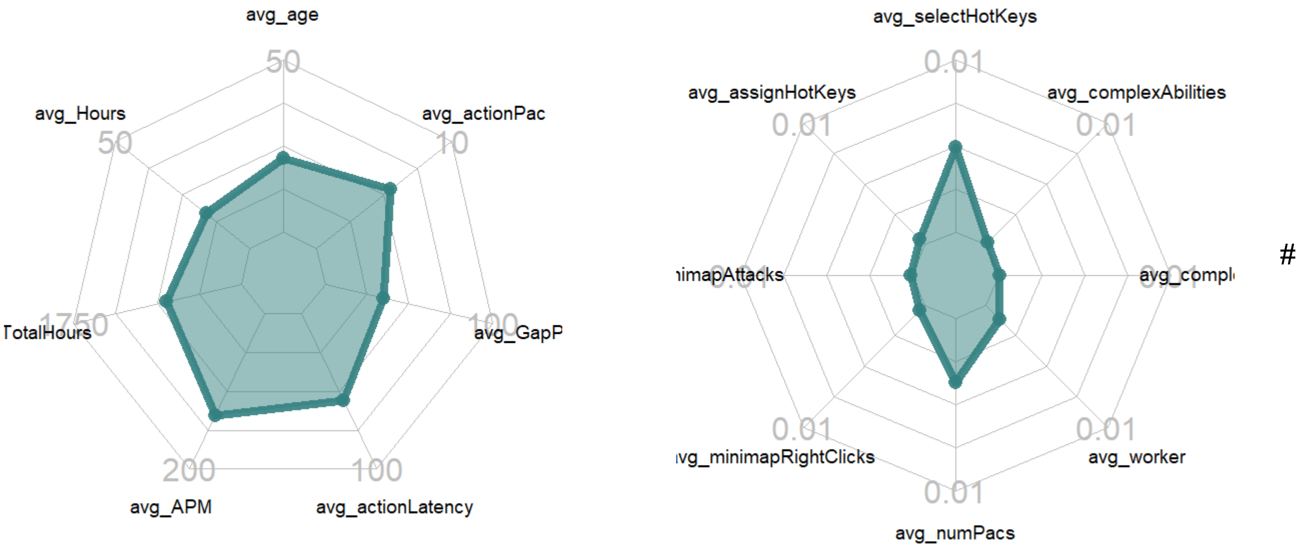
```
gr1DL<-rbind(maxGr1,minGr1,group1[5,2:8])
gr2DL<-rbind(maxGr2,minGr2,group2[5,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1DL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.5,0.9) , pfcol=rgb(0.2,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2DL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.5,0.9) , pfcol=rgb(0.2,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```



#

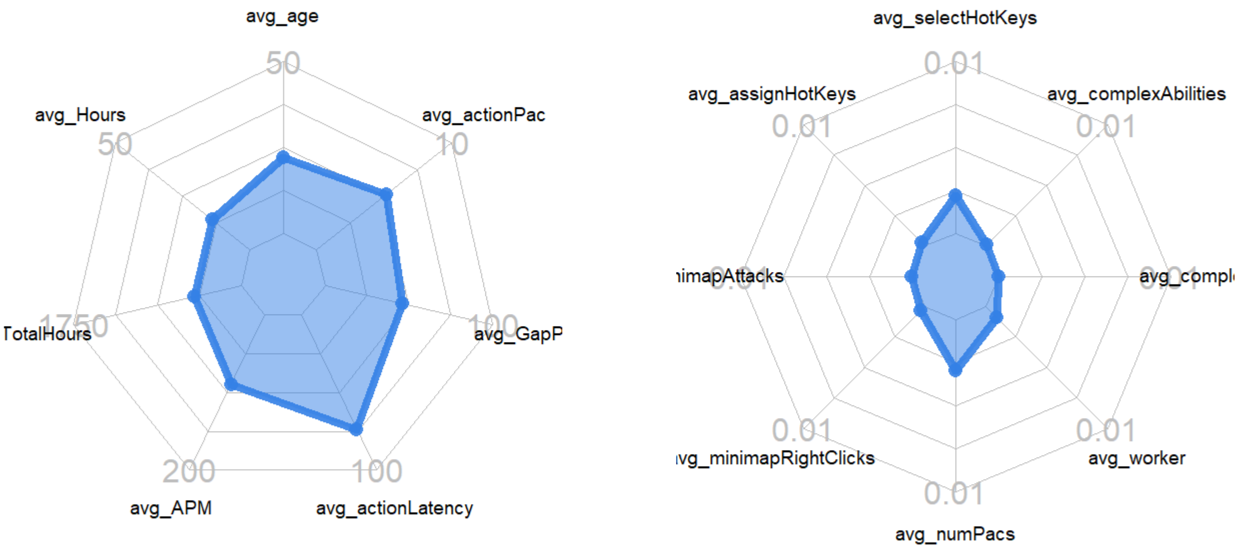
Platinum

```
gr1PL<-rbind(maxGr1,minGr1,group1[4,2:8])
gr2PL<-rbind(maxGr2,minGr2,group2[4,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1DL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.5,0.9) , pfcol=rgb(0.2,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2DL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.5,0.9) , pfcol=rgb(0.2,0.5,0.5,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```



Gold

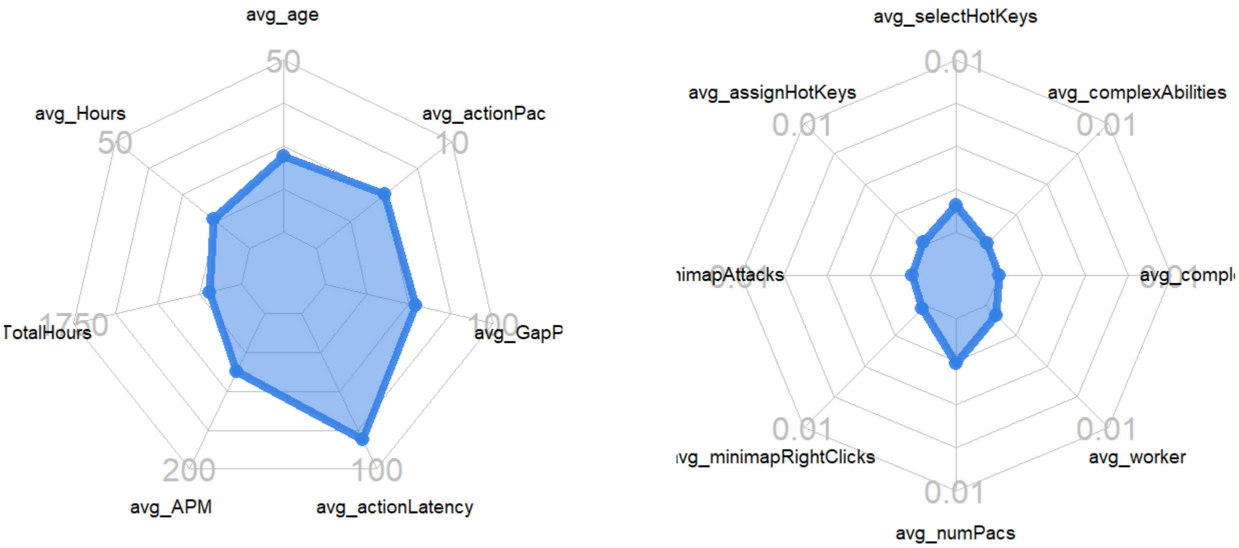
```
gr1GL<-rbind(maxGr1,minGr1,group1[3,2:8])
gr2GL<-rbind(maxGr2,minGr2,group2[3,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1GL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2GL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```



#

Silver

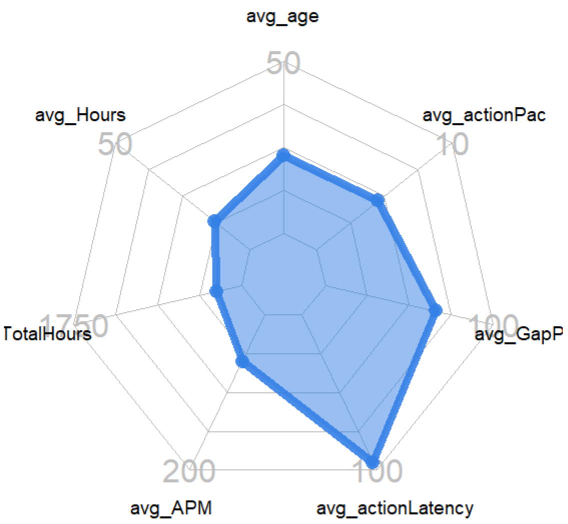
```
gr1SL<-rbind(maxGr1,minGr1,group1[2,2:8])
gr2SL<-rbind(maxGr2,minGr2,group2[2,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1SL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2SL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```

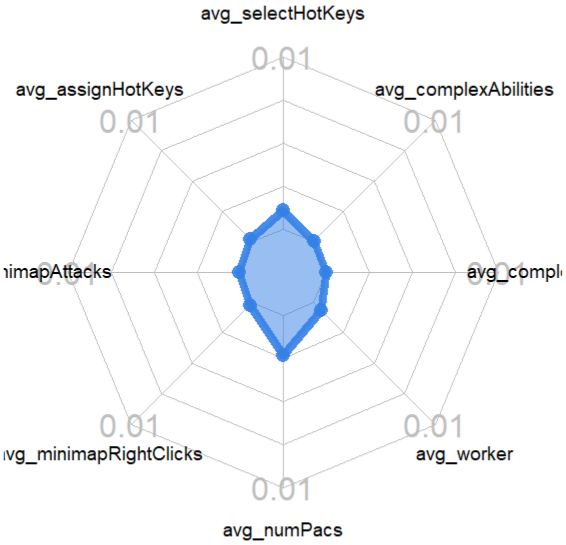


#

Bronze

```
gr1BL<-rbind(maxGr1,minGr1,group1[1,2:8])
gr2BL<-rbind(maxGr2,minGr2,group2[1,2:9])
op <- par(mar=c(1, 2, 2, 1),mfrow=c(1, 2))
radarchart( gr1BL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,2000,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
radarchart( gr2BL  , axistype=2 ,
    #custom polygon
     pcol=rgb(0.2,0.5,0.9,0.9) , pfcol=rgb(0.2,0.5,0.9,0.5) , plwd=4 ,
     #custom the grid
    cglcol="grey", cglty=1, axislabcol="grey", caxislabels=seq(0,.1,5), cglwd=0.8,
    #custom labels
     vlcex=0.6
     )
```

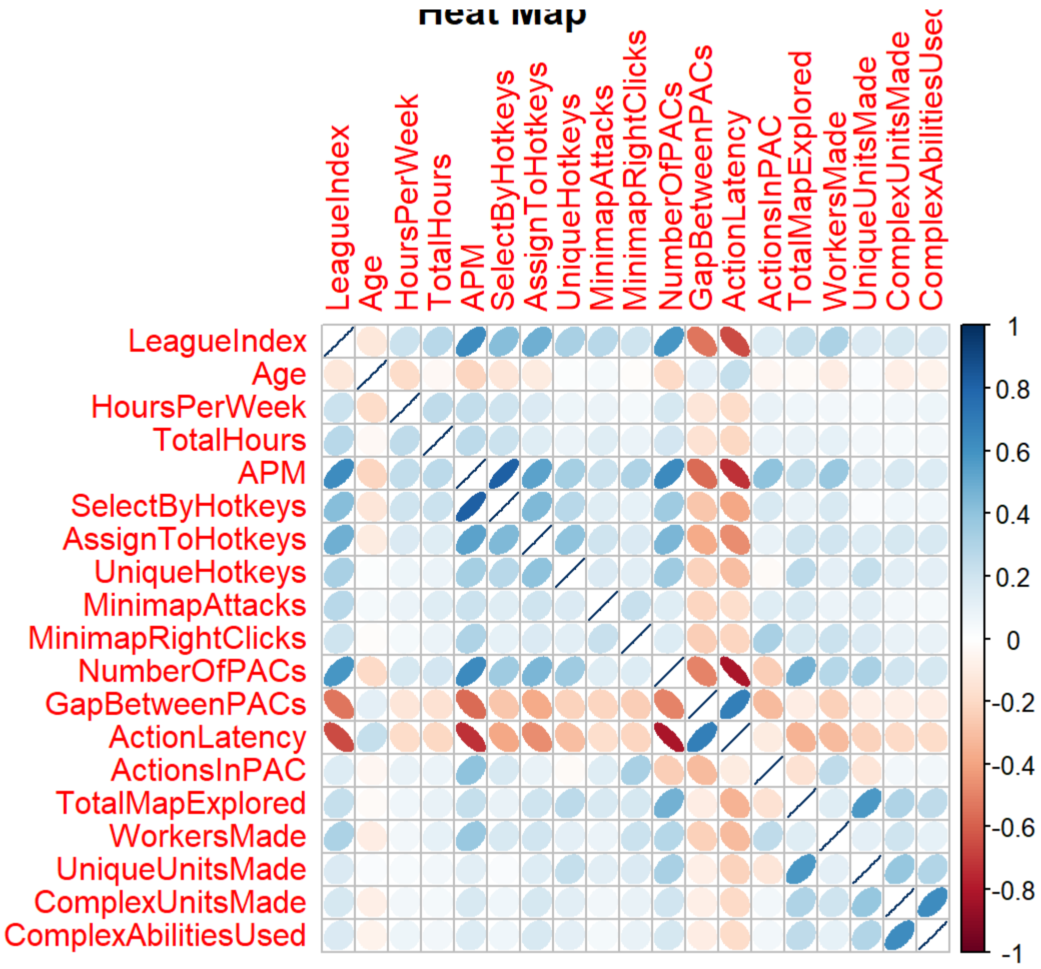




Correlation plot

Check if there are features correlated with each other. From below, we can see the ActionLatency and GapBetweenPACs have high correlation with multiple variable.

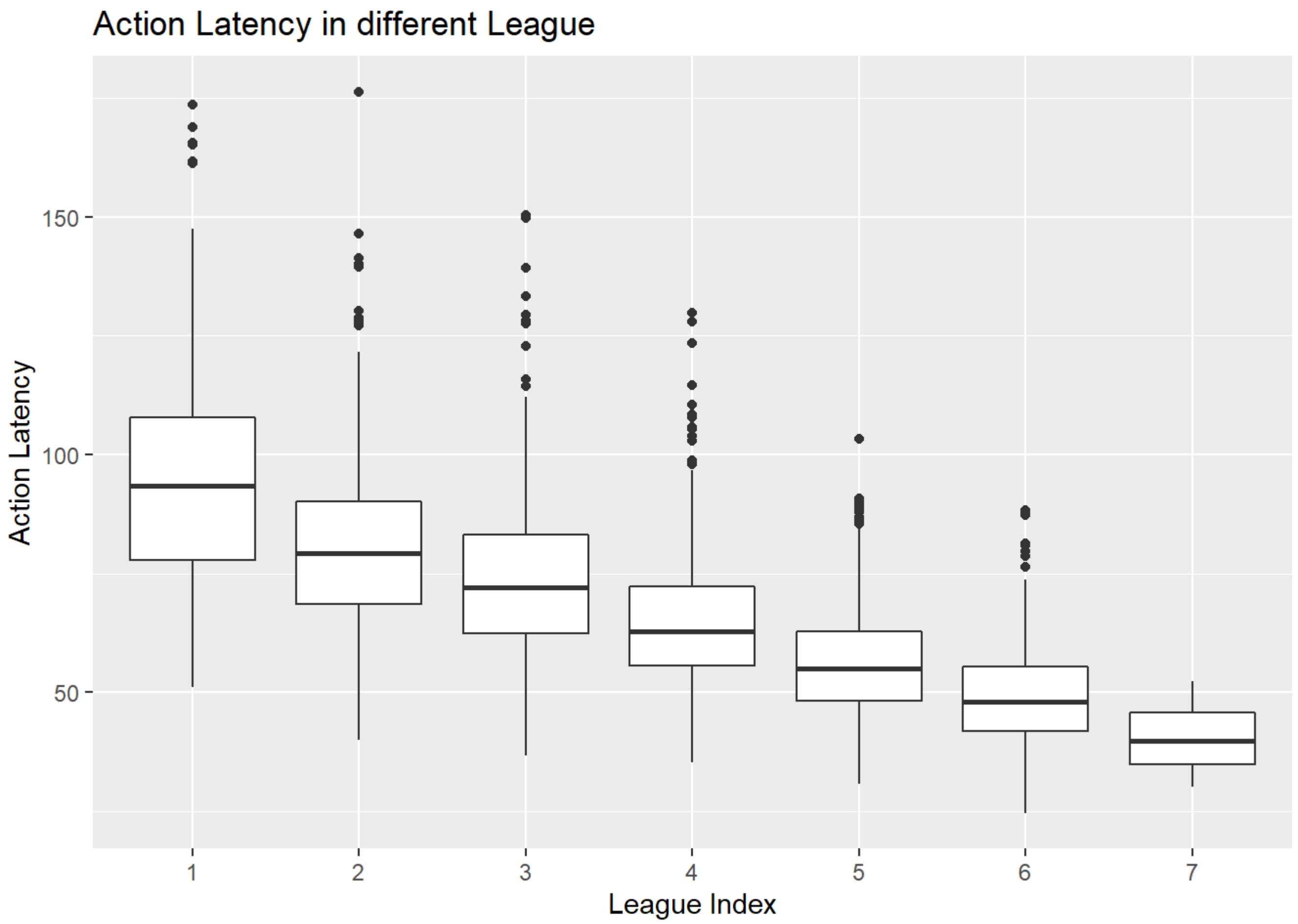
```
num.cols = sapply(df, is.numeric)
cor.data = cor(df[,num.cols])
corrPLOT = corrplot(cor.data,method='ellipse',  title="Heat Map")
```



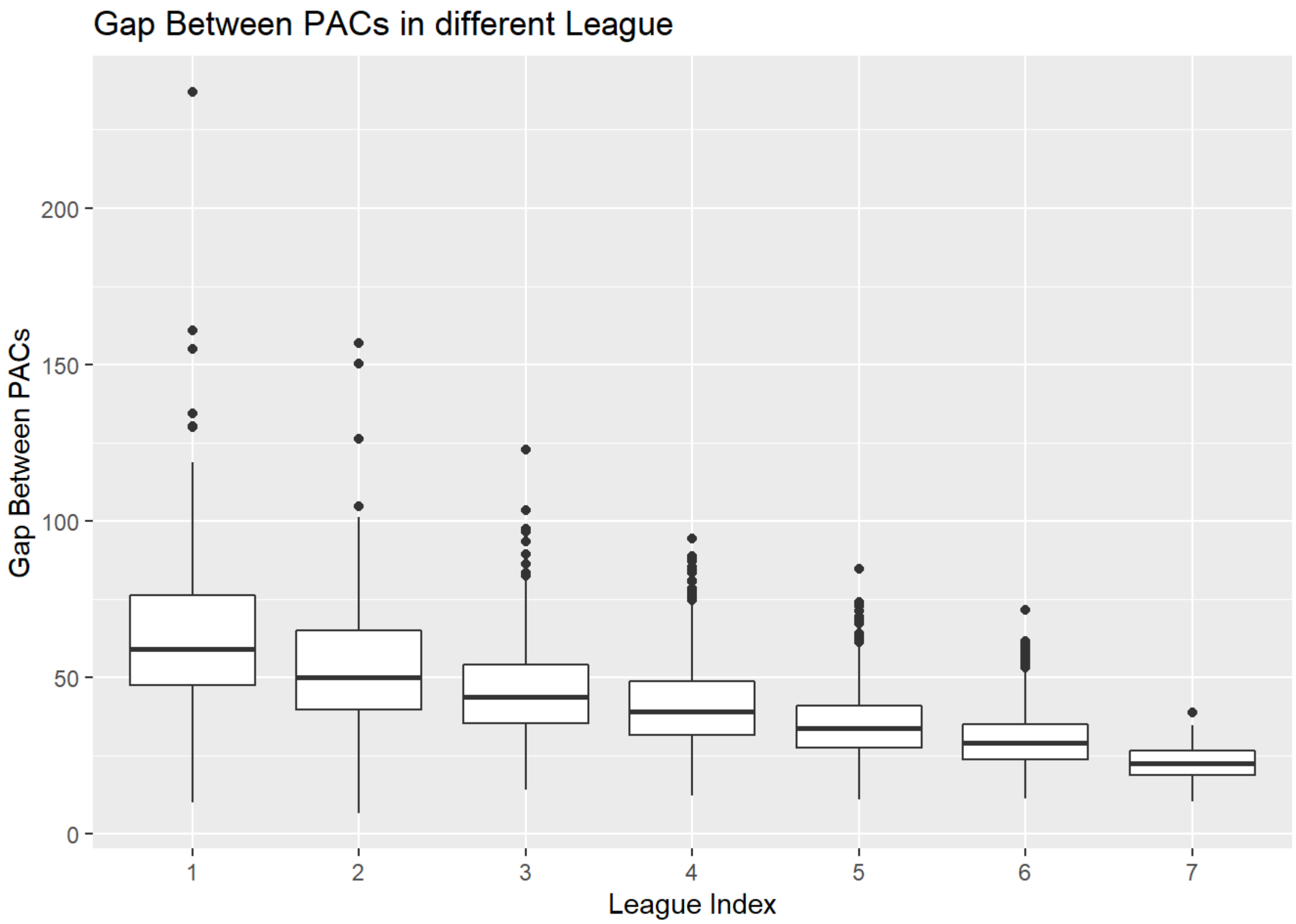
Data Exploration

Our target variable is LeagueName, see different class of LeagueName have what features, help us to classificate future data. 1. Start from variable Action Latency and Gap Between PACs, because these two have high correlation

```
# ActionLatency in different League
actionLatVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= ActionLatency)) +
geom_boxplot()  + ggtitle('Action Latency in different League')+
xlab('League Index') + ylab('Action Latency')
print(actionLatVsLeague)
```

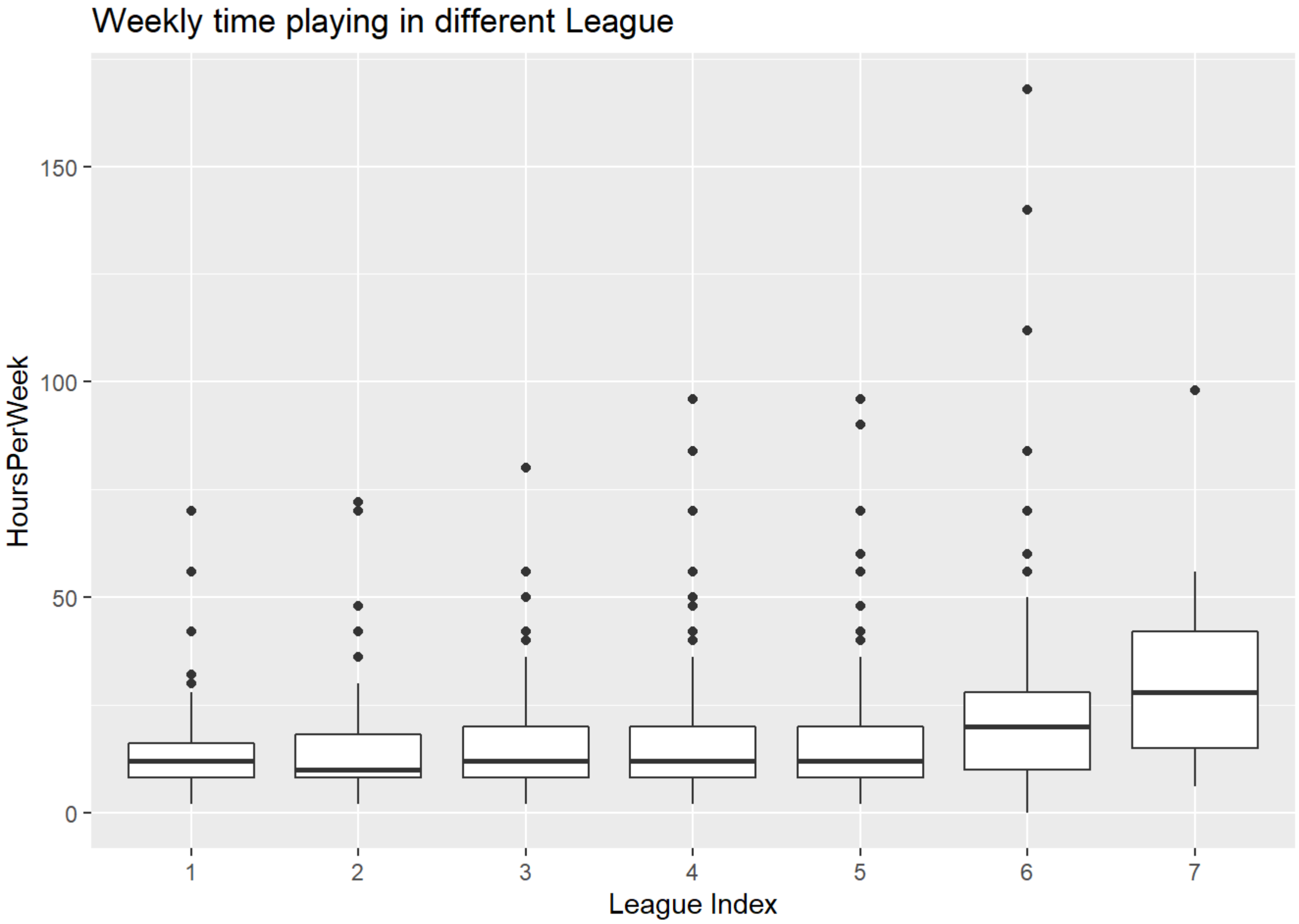


```
# GapBetweenPACs in different League
gapPacVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= GapBetweenPACs)) +
geom_boxplot() + ggtitle('Gap Between PACs in different League')+
xlab('League Index') + ylab('Gap Between PACs')

print(gapPacVsLeague)
```

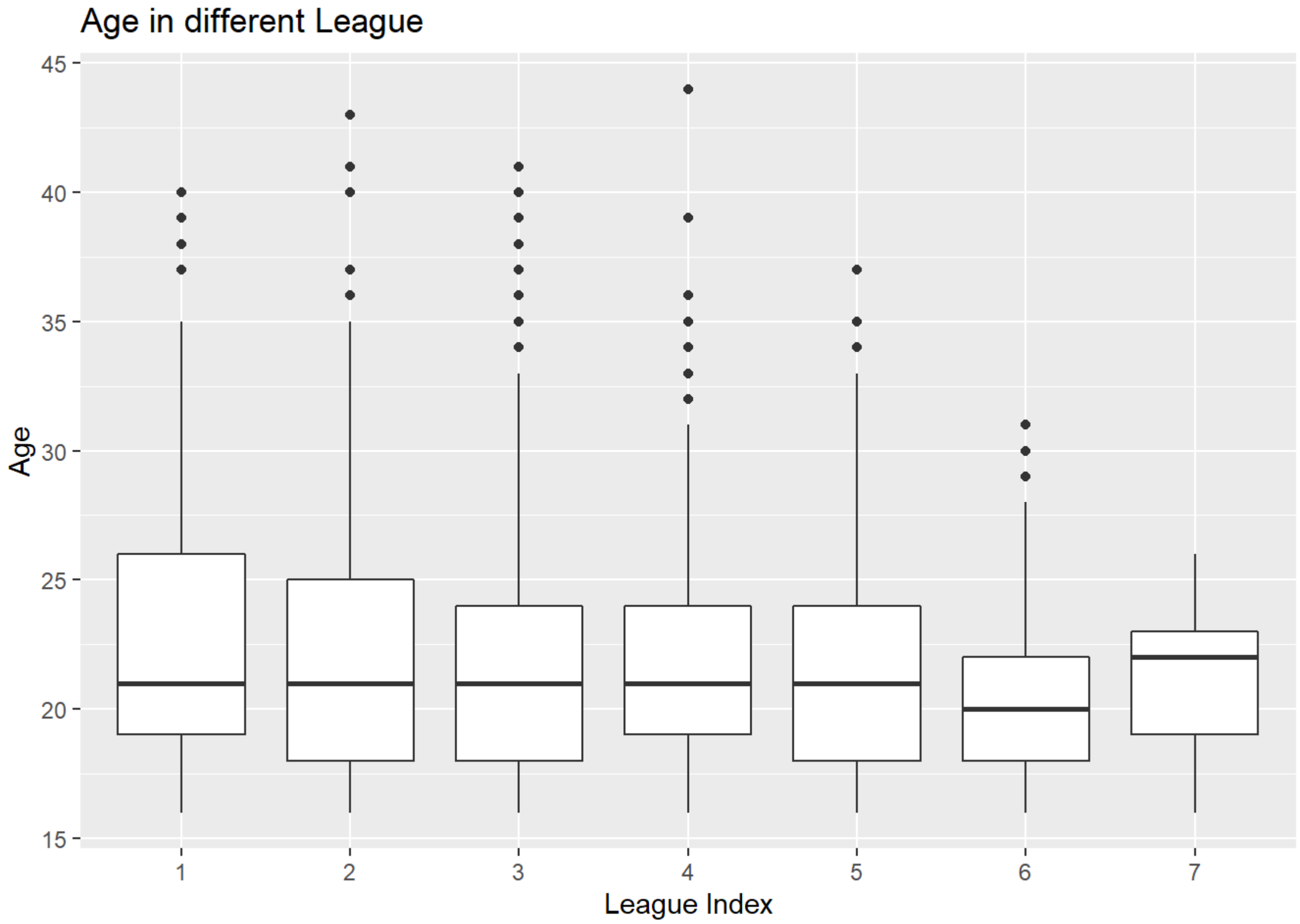


From the two graphs above we can see that these 2 variables have very simillar feature, consider to use feature selection in the future.

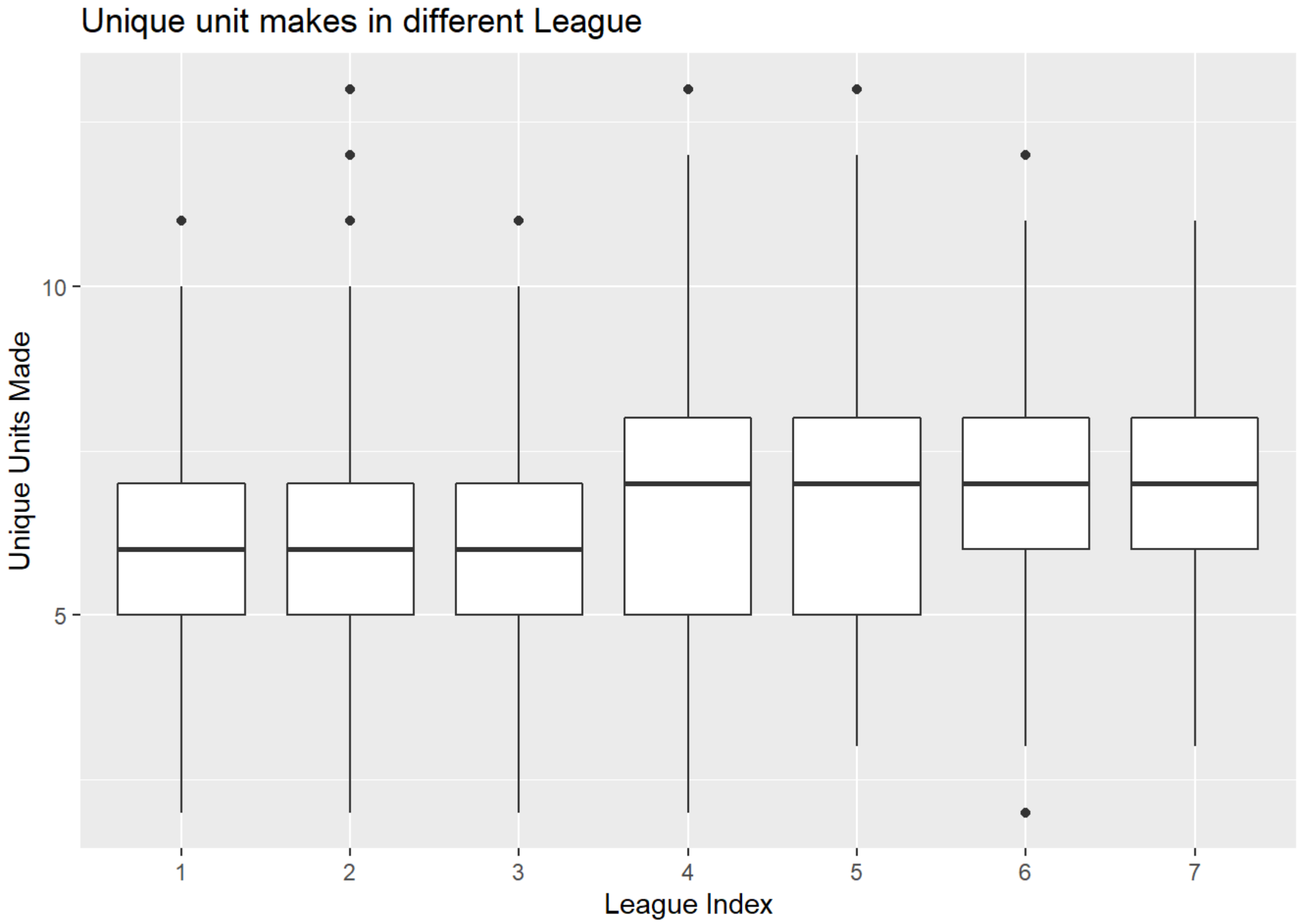
```
# Weekly time playing in different League
hourWeekVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= HoursPerWeek)) +
geom_boxplot()  + ggtitle('Weekly time playing in different League')+
xlab('League Index') + ylab('HoursPerWeek')
print(hourWeekVsLeague)
```



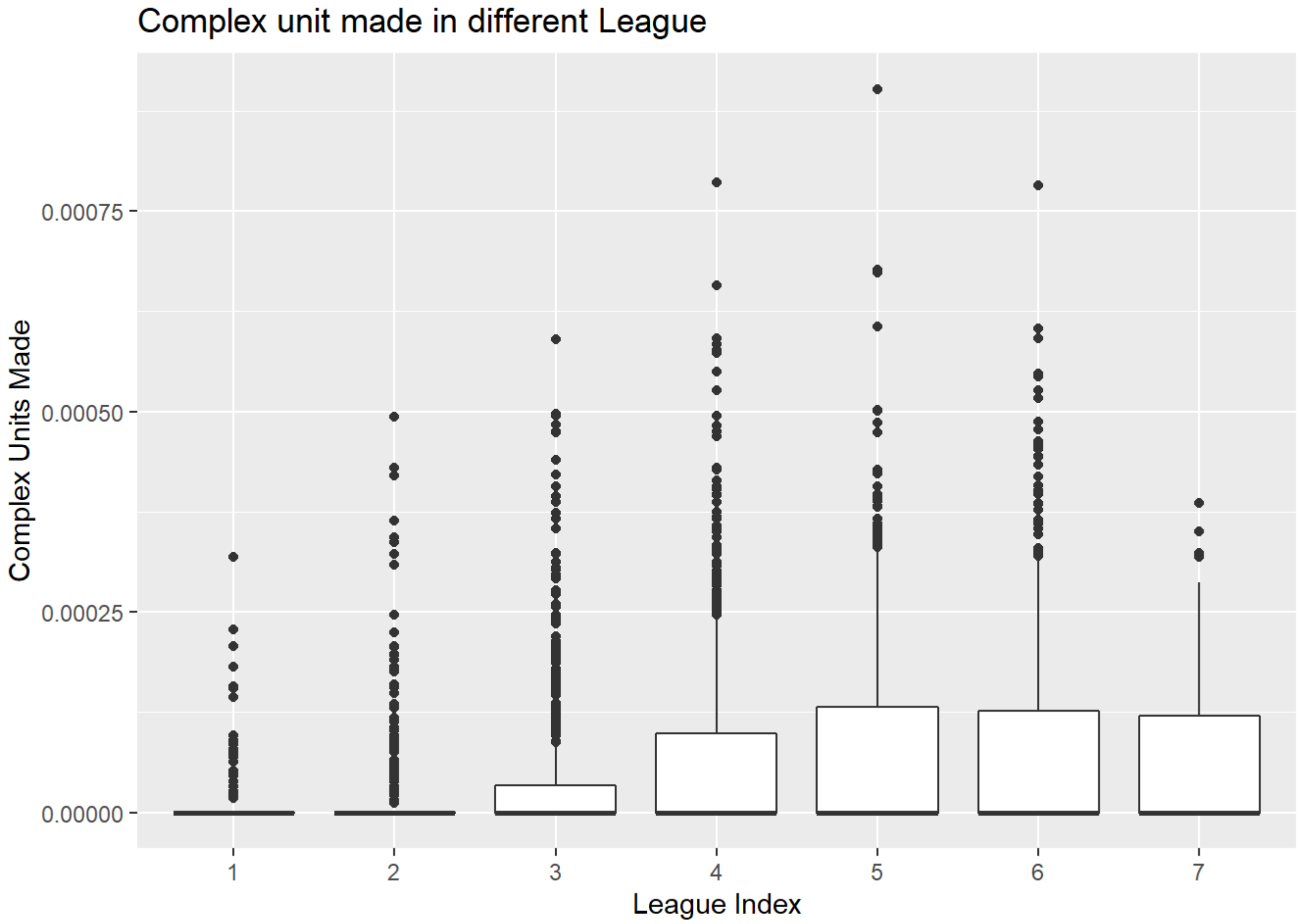
```
ageVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= Age)) +
geom_boxplot()+ ggtitle('Age in different League')+
xlab('League Index') + ylab('Age')
print(ageVsLeague)
```



```
uniqueUnitsMadeVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= UniqueUnitsMade)) +
geom_boxplot() + ggtitle('Unique unit makes in different League')+
xlab('League Index') + ylab('Unique Units Made')
print(uniqueUnitsMadeVsLeague)
```



```
complexUnitsMadeVsLeague<-ggplot(data=df,aes(x=factor(LeagueIndex),y= ComplexUnitsMade)) +
geom_boxplot()  + ggtitle('Complex unit made in different League')+
xlab('League Index') + ylab('Complex Units Made')
print(complexUnitsMadeVsLeague)
```



#

Hypotheses after Data Exploration It looks like the age is not very sensitive to different class; GapBetweenPACs and ActionLatency are very similar, maybe I should try feature selection model(Lasso Regression); Weekly time playing might be a very strong variable.

Train Test Split

1. Split data into train, test

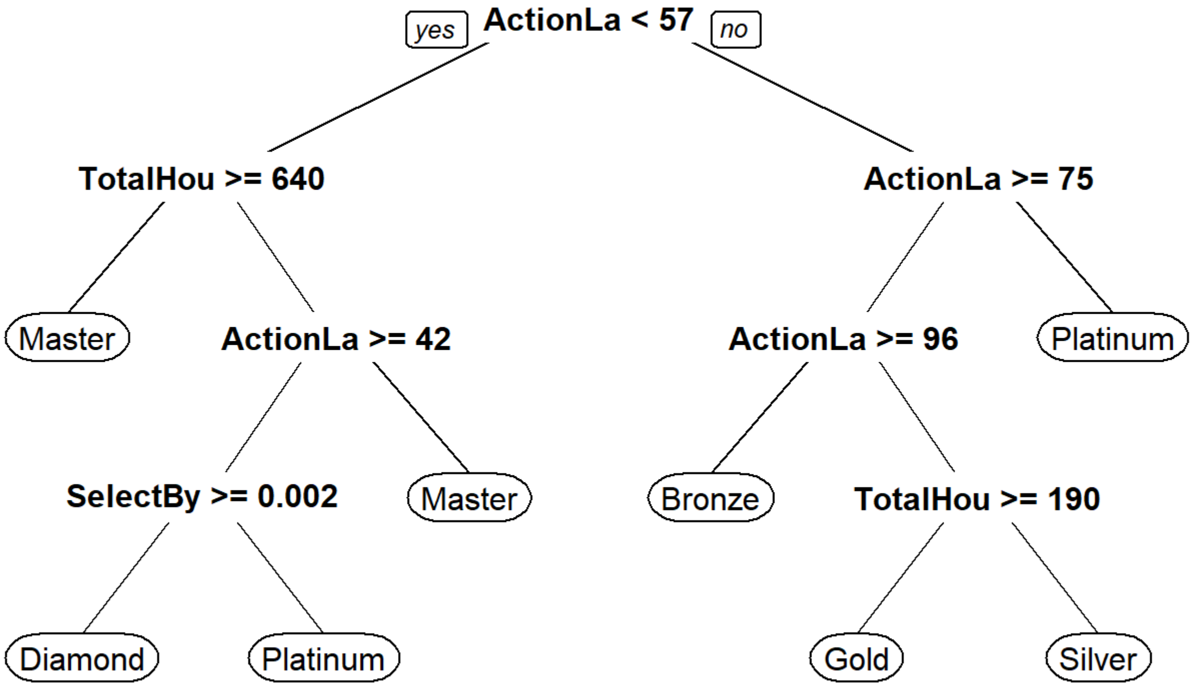
```
set.seed(123)
df2<-subset( df, select = -c(LeagueIndex) )
split<-sample.split(df2$LeagueName,SplitRatio=0.7)
train<-subset(df2,split==T)
test<-subset(df2,split==F)
```

Decision Tree

```
tree <- rpart(LeagueName ~ ., method='class',data = train)
print(tree$variable.importance)
```

```
##      ActionLatency       NumberOfPACs                APM 
##       189.63872942       102.51255230        97.30029932 
##    SelectByHotkeys     GapBetweenPACs         TotalHours 
##        55.86567432        53.85143443        33.34644111 
##    AssignToHotkeys        WorkersMade       HoursPerWeek 
##        31.14934494         7.25672545         6.15298127 
##       ActionsInPAC     MinimapAttacks MinimapRightClicks 
##         0.22939825         0.07815776         0.07815776
```

```
prp(tree)
```



LDA

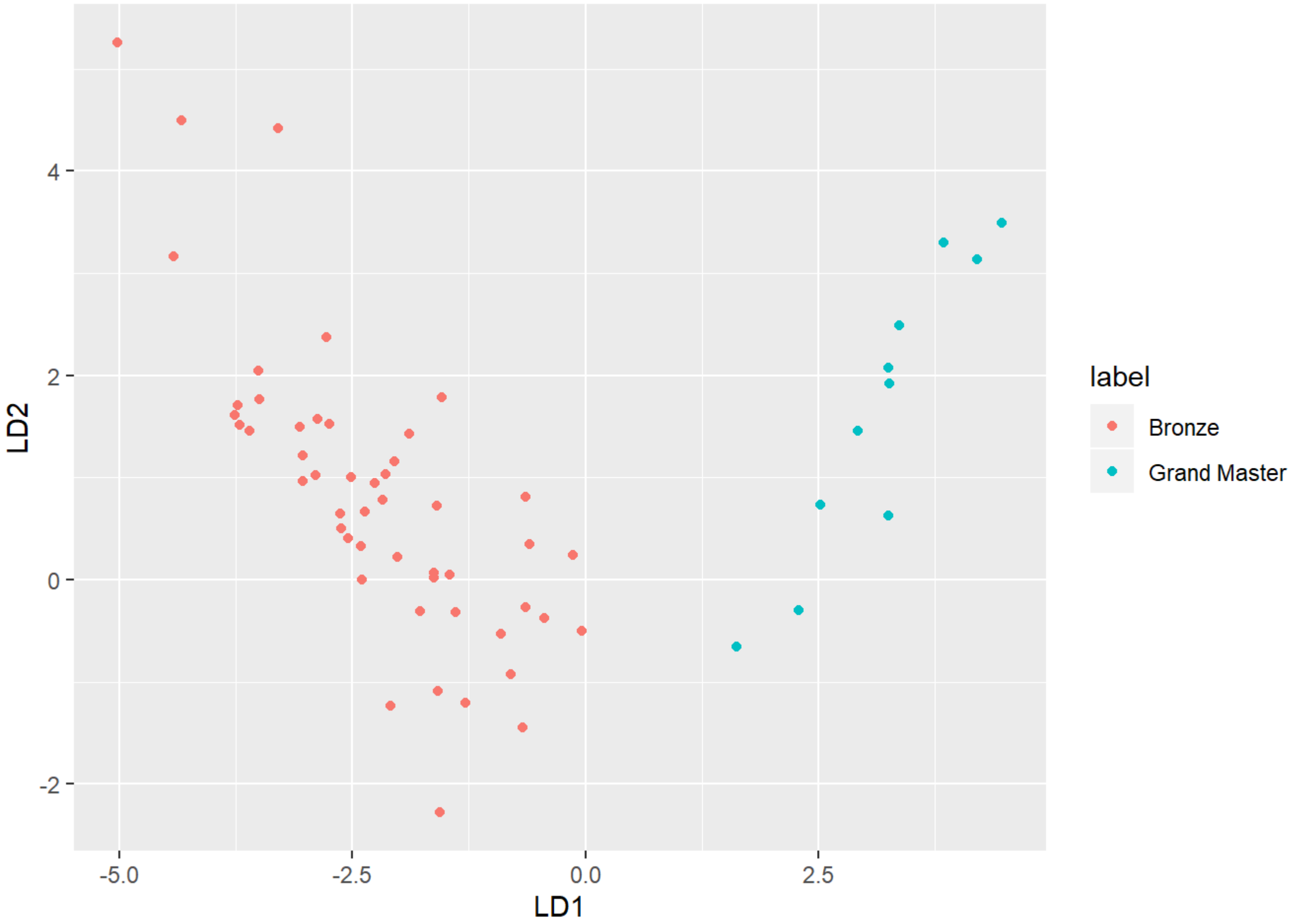
```
lda.pred=lda(LeagueName~.,data=train)
lda.pred
```

```
## Call:
## lda(LeagueName ~ ., data = train)
##
## Prior probabilities of groups:
##       Bronze      Diamond         Gold Grand Master       Master
##   0.05008562   0.24058219   0.16566781   0.01027397   0.18621575
##     Platinum       Silver
##   0.24315068   0.10402397
##
## Group means:
##                   Age HoursPerWeek TotalHours       APM SelectByHotkeys
## Bronze       22.49573     14.00000   257.6581  58.83999     0.001062932
## Diamond      21.42883     16.15658   783.0516 130.63458     0.004868788
## Gold         22.06460     13.81912   513.4806  88.91779     0.002145077
## Grand Master 21.33333     34.00000  1772.5000 182.32355     0.008742126
## Master       20.75632     20.87816   958.8276 160.03235     0.007688514
## Platinum     22.03345     13.72887   561.6496 105.73900     0.003114548
## Silver       22.03292     13.21811   318.8025  75.73658     0.001586608
##              AssignToHotkeys UniqueHotkeys MinimapAttacks
## Bronze          0.0001862951      2.923077   2.978232e-05
## Diamond         0.0004052962      4.706406   1.158963e-04
## Gold            0.0002816129      3.909561   5.488151e-05
## Grand Master    0.0006766374      6.791667   3.458070e-04
## Master          0.0005161578      5.581609   1.504730e-04
## Platinum        0.0003399526      4.008803   7.939951e-05
## Silver          0.0002227272      3.296296   4.308762e-05
##              MinimapRightClicks NumberOfPACs GapBetweenPACs ActionLatency
## Bronze             0.0002076772  0.002381306       65.79290      95.84893
## Diamond            0.0004171074  0.003746964       35.24713      56.36364
## Gold               0.0003214296  0.002965733       46.15452      74.22697
## Grand Master       0.0005333497  0.005035265       24.37271      41.54375
## Master             0.0004724815  0.004247200       30.25161      48.92357
## Platinum           0.0003806799  0.003292261       40.43913      64.89123
## Silver             0.0002725534  0.002659261       53.58335      80.81740
##              ActionsInPAC TotalMapExplored  WorkersMade UniqueUnitsMade
## Bronze           4.489144         19.38462 0.0006206763        5.871795
## Diamond          5.434637         23.22598 0.0011693208        6.775801
## Gold             5.121120         20.27907 0.0009198385        6.330749
## Grand Master     5.158579         27.83333 0.0012346825        7.166667
## Master           5.441317         24.37241 0.0011722084        6.926437
## Platinum         5.263245         21.95951 0.0010014881        6.623239
## Silver           5.034291         19.98354 0.0007923544        5.958848
##              ComplexUnitsMade ComplexAbilitiesUsed
## Bronze           1.275678e-05         3.631013e-05
## Diamond          7.504868e-05         1.684595e-04
## Gold             4.313020e-05         1.050045e-04
## Grand Master     9.860102e-05         2.585798e-04
## Master           8.438838e-05         1.922371e-04
## Platinum         6.564012e-05         1.323914e-04
## Silver           2.094619e-05         7.355346e-05
##
## Coefficients of linear discriminants:
##                                LD1           LD2           LD3
## Age                   1.456265e-02 -1.102927e-02 -4.955421e-02
## HoursPerWeek          8.701369e-03  3.672260e-02 -2.004778e-02
## TotalHours            2.900210e-04  3.494916e-05 -6.458428e-04
## APM                  -5.861201e-03  2.256855e-02  2.238597e-02
## SelectByHotkeys       7.543948e+01 -7.756451e+01  3.533394e+01
## AssignToHotkeys       1.154205e+03  6.594133e+02 -4.744327e+02
## UniqueHotkeys         5.833801e-02  4.131316e-02 -1.879799e-02
## MinimapAttacks        1.265328e+03  1.273398e+03 -2.553464e+03
## MinimapRightClicks    3.925708e+01 -1.953506e+02  1.440650e+02
## NumberOfPACs          5.165904e+02  8.811524e+02 -4.761345e+02
## GapBetweenPACs       -1.128214e-02  1.557792e-02  1.903605e-02
## ActionLatency        -2.984980e-02  8.101008e-02 -9.827876e-03
## ActionsInPAC          1.257898e-01 -1.052150e-01 -1.653077e-01
## TotalMapExplored     -1.165464e-02  2.081683e-03  8.277697e-04
## WorkersMade           2.747279e+02 -5.167426e+02 -3.665187e+02
## UniqueUnitsMade      -2.586298e-02 -8.636367e-02  4.100022e-02
## ComplexUnitsMade      5.744370e+02 -1.024000e+03 -3.303081e+02
## ComplexAbilitiesUsed  8.945493e+01  2.287344e+02 -1.216603e+02
##                                LD4           LD5           LD6
## Age                  -4.661649e-02  6.371564e-02  1.061305e-03
## HoursPerWeek          1.025083e-02 -1.664092e-03 -3.537873e-02
## TotalHours           -4.896269e-05 -2.379360e-04  3.674535e-04
## APM                  -4.092559e-02  1.917234e-02  5.287391e-02
## SelectByHotkeys       2.303130e+02 -1.139476e+02 -3.024654e+02
## AssignToHotkeys      -2.144105e+03  1.133723e+03 -1.215325e+03
## UniqueHotkeys         7.121539e-02 -2.870497e-01 -3.116055e-03
## MinimapAttacks        1.723757e+03  2.287222e+03 -1.535564e+02
## MinimapRightClicks   -5.912364e+02  2.685603e+00 -7.149599e+02
## NumberOfPACs          8.161903e+02 -6.377036e+02 -4.993337e+02
## GapBetweenPACs        3.028946e-02  3.500089e-02  1.689146e-02
## ActionLatency        -3.805419e-02 -3.558865e-02  3.503786e-02
## ActionsInPAC          7.545162e-01 -2.558650e-01 -5.304410e-01
## TotalMapExplored      3.744598e-02  7.099127e-02 -1.286991e-02
## WorkersMade           6.375793e+02 -3.582827e+02  1.136143e+03
## UniqueUnitsMade      -2.185240e-01 -1.162822e-01  9.039948e-02
## ComplexUnitsMade     -5.640276e+03  2.060536e+03  2.870001e+02
## ComplexAbilitiesUsed  1.384406e+03 -6.311793e+02  3.728856e+02
## 
## Proportion of trace:
##    LD1    LD2    LD3    LD4    LD5    LD6 
## 0.8696 0.0993 0.0139 0.0062 0.0060 0.0049
```

```
ldatest=predict(lda.pred,test)
#ldatest$x
table(ldatest$class,test$LeagueName)
```

```
##
##                Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           21       0   14            0      0        7     19
##   Diamond           0     110   13            0     70       54      4
##   Gold             11      11   37            0      2       21     35
##   Grand Master      0       0    0            5     11        2      0
##   Master            0      56    3            6     83       21      1
##   Platinum          9      63   87            0     20      129     34
##   Silver            9       1   12            0      0        9     11
```

```
z <- data.frame(ldatest$x, label=test$LeagueName)

ggplot(z[z$label=='Grand Master' | z$label=="Bronze",], aes(LD1,LD2))+geom_point(aes(col=label))
```



```
mean(ldatest$class==test$LeagueName)
```

```
## [1] 0.3956044
```

QDA

```
qda.pred=qda(LeagueName~.,data=train)
qda.pred
```

```
## Call:
## qda(LeagueName ~ ., data = train)
## 
## Prior probabilities of groups:
##       Bronze      Diamond         Gold Grand Master       Master 
##   0.05008562   0.24058219   0.16566781   0.01027397   0.18621575 
##     Platinum       Silver 
##   0.24315068   0.10402397 
## 
## Group means:
##                   Age HoursPerWeek TotalHours       APM SelectByHotkeys
## Bronze       22.49573     14.00000   257.6581  58.83999     0.001062932
## Diamond      21.42883     16.15658   783.0516 130.63458     0.004868788
## Gold         22.06460     13.81912   513.4806  88.91779     0.002145077
## Grand Master 21.33333     34.00000  1772.5000 182.32355     0.008742126
## Master       20.75632     20.87816   958.8276 160.03235     0.007688514
## Platinum     22.03345     13.72887   561.6496 105.73900     0.003114548
## Silver       22.03292     13.21811   318.8025  75.73658     0.001586608
##              AssignToHotkeys UniqueHotkeys MinimapAttacks
## Bronze          0.0001862951      2.923077   2.978232e-05
## Diamond         0.0004052962      4.706406   1.158963e-04
## Gold            0.0002816129      3.909561   5.488151e-05
## Grand Master    0.0006766374      6.791667   3.458070e-04
## Master          0.0005161578      5.581609   1.504730e-04
## Platinum        0.0003399526      4.008803   7.939951e-05
## Silver          0.0002227272      3.296296   4.308762e-05
##              MinimapRightClicks NumberOfPACs GapBetweenPACs ActionLatency
## Bronze             0.0002076772  0.002381306       65.79290      95.84893
## Diamond            0.0004171074  0.003746964       35.24713      56.36364
## Gold               0.0003214296  0.002965733       46.15452      74.22697
## Grand Master       0.0005333497  0.005035265       24.37271      41.54375
## Master             0.0004724815  0.004247200       30.25161      48.92357
## Platinum           0.0003806799  0.003292261       40.43913      64.89123
## Silver             0.0002725534  0.002659261       53.58335      80.81740
##              ActionsInPAC TotalMapExplored  WorkersMade UniqueUnitsMade
## Bronze           4.489144         19.38462 0.0006206763        5.871795
## Diamond          5.434637         23.22598 0.0011693208        6.775801
## Gold             5.121120         20.27907 0.0009198385        6.330749
## Grand Master     5.158579         27.83333 0.0012346825        7.166667
## Master           5.441317         24.37241 0.0011722084        6.926437
## Platinum         5.263245         21.95951 0.0010014881        6.623239
## Silver           5.034291         19.98354 0.0007923544        5.958848
##              ComplexUnitsMade ComplexAbilitiesUsed
## Bronze           1.275678e-05         3.631013e-05
## Diamond          7.504868e-05         1.684595e-04
## Gold             4.313020e-05         1.050045e-04
## Grand Master     9.860102e-05         2.585798e-04
## Master           8.438838e-05         1.922371e-04
## Platinum         6.564012e-05         1.323914e-04
## Silver           2.094619e-05         7.355346e-05
```

```
qdatest=predict(qda.pred,test)

table(qdatest$class,test$LeagueName)
```

```
## 
##                 Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           29       5   31            0      0       25     32
##   Diamond           0      48   15            1     39       28      3
##   Gold              3      14   23            0      6       27     13
##   Grand Master      0       2    1            3      4        0      0
##   Master            0      66    3            7     87       25      2
##   Platinum          3      90   62            0     46      104     20
##   Silver           15      16   31            0      4       34     34
```

```
mean(qdatest$class==test$LeagueName)
```

```
## [1] 0.3276723
```

Random Forest

```
train$LeagueName = factor(train$LeagueName)
rf.model<-randomForest(LeagueName ~ . , data = train,importance = TRUE)
print(rf.model)
```

```
## 
## Call:
##  randomForest(formula = LeagueName ~ ., data = train, importance = TRUE) 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
## 
##         OOB estimate of  error rate: 57.92%
## Confusion matrix:
##              Bronze Diamond Gold Grand Master Master Platinum Silver
## Bronze           35       0   22            0      1       10     49
## Diamond           0     240   29            0    123      166      4
## Gold             14      48  120            0      6      148     51
## Grand Master      0       2    0            0     22        0      0
## Master            0     142    1            1    248       43      0
## Platinum          5     145   98            0     33      265     22
## Silver           20       9   84            0      0       55     75
##              class.error
## Bronze         0.7008547
## Diamond        0.5729537
## Gold           0.6899225
## Grand Master   1.0000000
## Master         0.4298851
## Platinum       0.5334507
## Silver         0.6913580
```

```
predictionRF<-as.data.frame(predict(rf.model,test))
colnames(predictionRF)<-c('res')
test$LeagueName <- as.factor(test$LeagueName)
confusionMatrix(predictionRF$res,test$LeagueName)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           17       0    3            0      0        3     12
##   Diamond           0     111   17            0     63       63      3
##   Gold              9      13   45            0      3       48     36
##   Grand Master      0       0    0            0      0        0      0
##   Master            0      60    2           11    103       19      1
##   Platinum          6      56   80            0     17      102     25
##   Silver           18       1   19            0      0        8     27
## 
## Overall Statistics
## 
##                Accuracy : 0.4046          
##                  95% CI : (0.374, 0.4357)
##     No Information Rate : 0.2428          
##     P-Value [Acc > NIR] : < 2.2e-16       
## 
##                   Kappa : 0.2535          
## 
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity                0.34000         0.4606     0.27108
## Specificity                0.98107         0.8079     0.86946
## Pos Pred Value             0.48571         0.4319     0.29221
## Neg Pred Value             0.96584         0.8253     0.85714
## Prevalence                 0.04995         0.2408     0.16583
## Detection Rate             0.01698         0.1109     0.04496
## Detection Prevalence       0.03497         0.2567     0.15385
## Balanced Accuracy          0.66054         0.6342     0.57027
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                      0.00000        0.5538          0.4198
## Specificity                      1.00000        0.8859          0.7573
## Pos Pred Value                       NaN        0.5255          0.3566
## Neg Pred Value                   0.98901        0.8969          0.8028
## Prevalence                       0.01099        0.1858          0.2428
## Detection Rate                   0.00000        0.1029          0.1019
## Detection Prevalence             0.00000        0.1958          0.2857
## Balanced Accuracy                0.50000        0.7198          0.5885
##                      Class: Silver
## Sensitivity                0.25962
## Specificity                0.94872
## Pos Pred Value             0.36986
## Neg Pred Value             0.91703
## Prevalence                 0.10390
## Detection Rate             0.02697
## Detection Prevalence       0.07293
## Balanced Accuracy          0.60417
```

SVM linear B4 tune

```
svm_linear<-svm(LeagueName~., data=train, kernel='linear', cost=0.01)
summary(svm_linear)
```

```
## 
## Call:
## svm(formula = LeagueName ~ ., data = train, kernel = "linear", 
##     cost = 0.01)
## 
## 
## Parameters:
##    SVM-Type:  C-classification 
##  SVM-Kernel:  linear 
##        cost:  0.01 
## 
## Number of Support Vectors:  2296
## 
##  ( 562 568 387 24 395 117 243 )
## 
## 
## Number of Classes:  7 
## 
## Levels: 
##  Bronze Diamond Gold Grand Master Master Platinum Silver
```

```
# Prediction
pred_train_linear <- svm_linear$fitted
pred_test_linear <- predict(svm_linear,test)

# Error
conf_mtrx_train <- confusionMatrix(train$LeagueName,pred_train_linear)
cat("Linear train error rate(B4 tuned):",1-conf_mtrx_train$overall[1],"\n\n")
```

```
## Linear train error rate(B4 tuned): 0.5856164
```

```
conf_mtrx_test <- confusionMatrix(test$LeagueName,pred_test_linear)
cat("Linear test error rate(B4 tuned):",1-conf_mtrx_test$overall[1],"\n\n")
```

```
## Linear test error rate(B4 tuned): 0.6093906
```

```
print(conf_mtrx_train)
```

```
## Confusion Matrix and Statistics
## 
##                Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze            5       0   65            0      1       27     19
##   Diamond           0     265   10            0    102      185      0
##   Gold              0      48  105            0      5      223      6
##   Grand Master      0       2    0            0     22        0      0
##   Master            0     166    0            0    225       44      0
##   Platinum          0     146   51            0     22      348      1
##   Silver            0       8  101            0      0      114     20
## 
## Overall Statistics
## 
##                Accuracy : 0.4144
##                  95% CI : (0.3943, 0.4347)
##     No Information Rate : 0.4028
##     P-Value [Acc > NIR] : 0.1319
## 
##                   Kappa : 0.2501
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity                1.00000         0.4173     0.31627
## Specificity                0.95195         0.8254     0.85928
## Pos Pred Value             0.04274         0.4715     0.27132
## Neg Pred Value             1.00000         0.7914     0.88353
## Prevalence                 0.00214         0.2718     0.14212
## Detection Rate             0.00214         0.1134     0.04495
## Detection Prevalence       0.05009         0.2406     0.16567
## Balanced Accuracy          0.97598         0.6214     0.58777
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA       0.59682          0.3698
## Specificity                      0.98973       0.89280          0.8423
## Pos Pred Value                        NA       0.51724          0.6127
## Neg Pred Value                        NA       0.92004          0.6646
## Prevalence                       0.00000       0.16139          0.4028
## Detection Rate                   0.00000       0.09632          0.1490
## Detection Prevalence             0.01027       0.18622          0.2432
## Balanced Accuracy                     NA       0.74481          0.6061
##                      Class: Silver
## Sensitivity               0.434783
## Specificity               0.902620
## Pos Pred Value            0.082305
## Neg Pred Value            0.987578
## Prevalence                0.019692
## Detection Rate            0.008562
## Detection Prevalence      0.104024
## Balanced Accuracy         0.668701
```

```
print(conf_mtrx_test)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze            0       0   28            0      0       12     10
##   Diamond           0     117    3            0     50       71      0
##   Gold              0      14   35            0      2      113      2
##   Grand Master      0       0    0            0     11        0      0
##   Master            0      77    0            0     88       21      0
##   Platinum          0      61   20            0     14      145      3
##   Silver            1       4   43            0      1       49      6
## 
## Overall Statistics
## 
##                Accuracy : 0.3906          
##                  95% CI : (0.3602, 0.4216)
##     No Information Rate : 0.4106          
##     P-Value [Acc > NIR] : 0.9064          
## 
##                   Kappa : 0.219           
## 
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity               0.000000         0.4286     0.27132
## Specificity               0.950000         0.8297     0.84977
## Pos Pred Value            0.000000         0.4855     0.21084
## Neg Pred Value            0.998948         0.7947     0.88743
## Prevalence                0.000999         0.2727     0.12887
## Detection Rate            0.000000         0.1169     0.03497
## Detection Prevalence      0.049950         0.2408     0.16583
## Balanced Accuracy         0.475000         0.6291     0.56054
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA       0.53012          0.3528
## Specificity                      0.98901       0.88263          0.8339
## Pos Pred Value                        NA       0.47312          0.5967
## Neg Pred Value                        NA       0.90429          0.6491
## Prevalence                       0.00000       0.16583          0.4106
## Detection Rate                   0.00000       0.08791          0.1449
## Detection Prevalence             0.01099       0.18581          0.2428
## Balanced Accuracy                     NA       0.70638          0.5933
##                      Class: Silver
## Sensitivity               0.285714
## Specificity               0.900000
## Pos Pred Value            0.057692
## Neg Pred Value            0.983278
## Prevalence                0.020979
## Detection Rate            0.005994
## Detection Prevalence      0.103896
## Balanced Accuracy         0.592857
```

SVM radial B4 tune

```
svm_radial<-svm(LeagueName~., data=train, kernel='radial', cost=0.01)
summary(svm_radial)
```

```
##
## Call:
## svm(formula = LeagueName ~ ., data = train, kernel = "radial",
##     cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  0.01
##
## Number of Support Vectors:  2336
##
##  ( 562 568 387 24 435 117 243 )
##
##
## Number of Classes:  7
##
## Levels:
##  Bronze Diamond Gold Grand Master Master Platinum Silver
```

```
# Prediction
pred_train_radial <- svm_radial$fitted
pred_test_radial <- predict(svm_radial,test)

# Error
conf_mtrx_train <- confusionMatrix(train$LeagueName,pred_train_radial)
cat("Radial train error rate(B4 tuned):",1-conf_mtrx_train$overall[1],"\n\n")
```

```
## Radial train error rate(B4 tuned): 0.7568493
```

```
conf_mtrx_test <- confusionMatrix(test$LeagueName,pred_test_radial)
cat("Radial test error rate(B4 tuned):",1-conf_mtrx_test$overall[1],"\n\n")
```

```
## Radial test error rate(B4 tuned): 0.7572428
```

```
print(conf_mtrx_train)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze            0       0    0            0      0      117      0
##   Diamond           0       0    0            0      0      562      0
##   Gold              0       0    0            0      0      387      0
##   Grand Master      0       0    0            0      0       24      0
##   Master            0       2    0            0      0      433      0
##   Platinum          0       0    0            0      0      568      0
##   Silver            0       0    0            0      0      243      0
## 
## Overall Statistics
## 
##                Accuracy : 0.2432
##                  95% CI : (0.2259, 0.2611)
##     No Information Rate : 0.9991
##     P-Value [Acc > NIR] : 1
## 
##                   Kappa : 0
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity                   NA      0.0000000          NA
## Specificity                0.94991      0.7592117      0.8343
## Pos Pred Value                NA      0.0000000          NA
## Neg Pred Value                NA      0.9988726          NA
## Prevalence                 0.00000      0.0008562      0.0000
## Detection Rate             0.00000      0.0000000      0.0000
## Detection Prevalence       0.05009      0.2405822      0.1657
## Balanced Accuracy             NA      0.3796058          NA
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                         NA            NA        0.243359
## Specificity                    0.98973        0.8138        1.000000
## Pos Pred Value                      NA            NA        1.000000
## Neg Pred Value                      NA            NA        0.001131
## Prevalence                     0.00000        0.0000        0.999144
## Detection Rate                 0.00000        0.0000        0.243151
## Detection Prevalence           0.01027        0.1862        0.243151
## Balanced Accuracy                   NA            NA        0.621680
##                      Class: Silver
## Sensitivity                   NA
## Specificity                  0.896
## Pos Pred Value                NA
## Neg Pred Value                NA
## Prevalence                   0.000
## Detection Rate               0.000
## Detection Prevalence         0.104
## Balanced Accuracy             NA
```

```
print(conf_mtrx_test)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze            0       0    0            0      0       50      0
##   Diamond           0       0    0            0      0      241      0
##   Gold              0       0    0            0      0      166      0
##   Grand Master      0       0    0            0      0       11      0
##   Master            0       0    0            0      0      186      0
##   Platinum          0       0    0            0      0      243      0
##   Silver            0       0    0            0      0      104      0
## 
## Overall Statistics
## 
##                Accuracy : 0.2428          
##                  95% CI : (0.2165, 0.2706)
##     No Information Rate : 1               
##     P-Value [Acc > NIR] : 1               
## 
##                   Kappa : 0               
## 
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity                     NA             NA          NA
## Specificity                0.95005         0.7592      0.8342
## Pos Pred Value                  NA             NA          NA
## Neg Pred Value                  NA             NA          NA
## Prevalence                 0.00000         0.0000      0.0000
## Detection Rate             0.00000         0.0000      0.0000
## Detection Prevalence       0.04995         0.2408      0.1658
## Balanced Accuracy               NA             NA          NA
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA            NA          0.2428
## Specificity                      0.98901        0.8142              NA
## Pos Pred Value                        NA            NA              NA
## Neg Pred Value                        NA            NA              NA
## Prevalence                       0.00000        0.0000          1.0000
## Detection Rate                   0.00000        0.0000          0.2428
## Detection Prevalence             0.01099        0.1858          0.2428
## Balanced Accuracy                     NA            NA              NA
##                      Class: Silver
## Sensitivity                     NA
## Specificity                 0.8961
## Pos Pred Value                  NA
## Neg Pred Value                  NA
## Prevalence                  0.0000
## Detection Rate              0.0000
## Detection Prevalence        0.1039
## Balanced Accuracy               NA
```

SVM linear after tune

```
# Tuned model
tune_linear <- tune(svm, LeagueName~., data=train, kernel='linear', range = list(cost=seq(0.01,
2.5,0.5)))

# Prediction
pred_train_linear_tuned <- tune_linear$best.model$fitted
pred_test_linear_tuned <- predict(tune_linear$best.model,test)

# Error
conf_mtrx_train_tuned <- confusionMatrix(train$LeagueName,pred_train_linear_tuned)
cat("Radial tuned train error rate(after tuned):",1-conf_mtrx_train_tuned$overall[1],"\n\n")
```

```
## Radial tuned train error rate(after tuned): 0.5552226
```

```
conf_mtrx_test_tuned <- confusionMatrix(test$LeagueName,pred_test_linear_tuned)
cat("Radial tuned test error rate(after tuned):",1-conf_mtrx_test_tuned$overall[1])
```

```
## Radial tuned test error rate(after tuned): 0.5874126
```

```
print(conf_mtrx_train_tuned)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           22       0   22            0      1       14     58
##   Diamond           0     247   21            0    118      170      6
##   Gold              4      46   98            0      5      182     52
##   Grand Master      0       2    0            0     22        0      0
##   Master            0     142    0            0    254       39      0
##   Platinum          0     138   54            0     28      321     27
##   Silver           10       6   53            0      0       77     97
## 
## Overall Statistics
## 
##                Accuracy : 0.4448          
##                  95% CI : (0.4245, 0.4652)
##     No Information Rate : 0.3438          
##     P-Value [Acc > NIR] : < 2.2e-16       
## 
##                   Kappa : 0.3002          
## 
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity               0.611111         0.4251     0.39516
## Specificity               0.958696         0.8205     0.86159
## Pos Pred Value            0.188034         0.4395     0.25323
## Neg Pred Value            0.993691         0.8117     0.92304
## Prevalence                0.015411         0.2487     0.10616
## Detection Rate            0.009418         0.1057     0.04195
## Detection Prevalence      0.050086         0.2406     0.16567
## Balanced Accuracy         0.784903         0.6228     0.62838
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA        0.5935          0.3998
## Specificity                      0.98973        0.9051          0.8389
## Pos Pred Value                        NA        0.5839          0.5651
## Neg Pred Value                        NA        0.9085          0.7274
## Prevalence                       0.00000        0.1832          0.3438
## Detection Rate                   0.00000        0.1087          0.1374
## Detection Prevalence             0.01027        0.1862          0.2432
## Balanced Accuracy                     NA        0.7493          0.6193
##                      Class: Silver
## Sensitivity                0.40417
## Specificity                0.93034
## Pos Pred Value             0.39918
## Neg Pred Value             0.93168
## Prevalence                 0.10274
## Detection Rate             0.04152
## Detection Prevalence       0.10402
## Balanced Accuracy          0.66726
```

```
print(conf_mtrx_test_tuned)
```

```
## Confusion Matrix and Statistics
## 
##                Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze            7       0   12            0      0        7     24
##   Diamond           0     114    8            0     58       59      2
##   Gold              3      10   33            0      3       94     23
##   Grand Master      0       0    0            0     11        0      0
##   Master            0      65    1            0    102       18      0
##   Platinum          2      61   23            0     16      125     16
##   Silver            7       4   26            0      1       34     32
## 
## Overall Statistics
## 
##                Accuracy : 0.4126          
##                  95% CI : (0.3819, 0.4438)
##     No Information Rate : 0.3367          
##     P-Value [Acc > NIR] : 3.284e-07       
##                                           
##                   Kappa : 0.2599          
##                                           
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity               0.368421         0.4488     0.32039
## Specificity               0.956212         0.8300     0.85189
## Pos Pred Value            0.140000         0.4730     0.19880
## Neg Pred Value            0.987382         0.8158     0.91617
## Prevalence                0.018981         0.2537     0.10290
## Detection Rate            0.006993         0.1139     0.03297
## Detection Prevalence      0.049950         0.2408     0.16583
## Balanced Accuracy         0.662316         0.6394     0.58614
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA        0.5340          0.3709
## Specificity                      0.98901        0.8963          0.8223
## Pos Pred Value                        NA        0.5484          0.5144
## Neg Pred Value                        NA        0.8908          0.7203
## Prevalence                       0.00000        0.1908          0.3367
## Detection Rate                   0.00000        0.1019          0.1249
## Detection Prevalence             0.01099        0.1858          0.2428
## Balanced Accuracy                     NA        0.7152          0.5966
##                      Class: Silver
## Sensitivity                0.32990
## Specificity                0.92035
## Pos Pred Value             0.30769
## Neg Pred Value             0.92754
## Prevalence                 0.09690
## Detection Rate             0.03197
## Detection Prevalence       0.10390
## Balanced Accuracy          0.62513
```

SVM radial after tune

```
# Tuned model
tune_radial <- tune(svm, LeagueName~., data=train, kernel='radial', range = list(cost=seq(0.01,10,0.1)))

# Prediction
pred_train_radial_tuned <- tune_radial$best.model$fitted
pred_test_radial_tuned <- predict(tune_radial$best.model,test)

# Error
conf_mtrx_train_tuned <- confusionMatrix(train$LeagueName,pred_train_radial_tuned)
cat("Radial tuned train error rate(after tuned):",1-conf_mtrx_train_tuned$overall[1],"\n\n")
```

```
## Radial tuned train error rate(after tuned): 0.3827055
```

```
conf_mtrx_test_tuned <- confusionMatrix(test$LeagueName,pred_test_radial_tuned)
cat("Radial tuned test error rate(after tuned):",1-conf_mtrx_test_tuned$overall[1])
```

```
## Radial tuned test error rate(after tuned): 0.6043956
```

```
print(conf_mtrx_train_tuned)
```

```
## Confusion Matrix and Statistics
## 
##                Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           46       2   28            0      0       12     29
##   Diamond           0     350   14            0     63      132      3
##   Gold              4      41  198            0      3      115     26
##   Grand Master      0       3    0            9     12        0      0
##   Master            0      81    0            0    320       34      0
##   Platinum          0      87   49            0     20      392     20
##   Silver            6       5   47            0      0       58    127
## 
## Overall Statistics
## 
##                 Accuracy : 0.6173
##                   95% CI : (0.5972, 0.6371)
##      No Information Rate : 0.3181
##      P-Value [Acc > NIR] : < 2.2e-16
## 
##                    Kappa : 0.5195
## 
##   Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                       Class: Bronze Class: Diamond Class: Gold
## Sensitivity                 0.82143         0.6151     0.58929
## Specificity                 0.96886         0.8800     0.90550
## Pos Pred Value              0.39316         0.6228     0.51163
## Neg Pred Value              0.99549         0.8766     0.92919
## Prevalence                  0.02397         0.2436     0.14384
## Detection Rate              0.01969         0.1498     0.08476
## Detection Prevalence        0.05009         0.2406     0.16567
## Balanced Accuracy           0.89514         0.7476     0.74739
##                       Class: Grand Master Class: Master Class: Platinum
## Sensitivity                      1.000000        0.7656          0.5276
## Specificity                      0.993554        0.9400          0.8895
## Pos Pred Value                   0.375000        0.7356          0.6901
## Neg Pred Value                   1.000000        0.9484          0.8015
## Prevalence                       0.003853        0.1789          0.3181
## Detection Rate                   0.003853        0.1370          0.1678
## Detection Prevalence             0.010274        0.1862          0.2432
## Balanced Accuracy                0.996777        0.8528          0.7086
##                       Class: Silver
## Sensitivity                 0.61951
## Specificity                 0.94557
## Pos Pred Value              0.52263
## Neg Pred Value              0.96273
## Prevalence                  0.08776
## Detection Rate              0.05437
## Detection Prevalence        0.10402
## Balanced Accuracy           0.78254
```

```
print(conf_mtrx_test_tuned)
```

```
## Confusion Matrix and Statistics
## 
##               Reference
## Prediction     Bronze Diamond Gold Grand Master Master Platinum Silver
##   Bronze           11       0   14            0      0        9     16
##   Diamond           0     110   12            0     63       55      1
##   Gold              5      13   46            0      2       80     20
##   Grand Master      0       1    0            0     10        0      0
##   Master            0      69    2            0     96       19      0
##   Platinum          4      62   38            0     19      109     11
##   Silver           10       3   33            0      3       31     24
## 
## Overall Statistics
## 
##                Accuracy : 0.3956
##                  95% CI : (0.3652, 0.4267)
##     No Information Rate : 0.3027
##     P-Value [Acc > NIR] : 2.569e-10
## 
##                   Kappa : 0.2404
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: Bronze Class: Diamond Class: Gold
## Sensitivity                0.36667         0.4264     0.31724
## Specificity                0.95984         0.8237     0.85981
## Pos Pred Value             0.22000         0.4564     0.27711
## Neg Pred Value             0.98002         0.8053     0.88144
## Prevalence                 0.02997         0.2577     0.14486
## Detection Rate             0.01099         0.1099     0.04595
## Detection Prevalence       0.04995         0.2408     0.16583
## Balanced Accuracy          0.66325         0.6250     0.58853
##                      Class: Grand Master Class: Master Class: Platinum
## Sensitivity                           NA        0.4974          0.3597
## Specificity                      0.98901        0.8886          0.8080
## Pos Pred Value                        NA        0.5161          0.4486
## Neg Pred Value                        NA        0.8810          0.7441
## Prevalence                       0.00000        0.1928          0.3027
## Detection Rate                   0.00000        0.0959          0.1089
## Detection Prevalence             0.01099        0.1858          0.2428
## Balanced Accuracy                     NA        0.6930          0.5839
##                      Class: Silver
## Sensitivity                0.33333
## Specificity                0.91389
## Pos Pred Value             0.23077
## Neg Pred Value             0.94649
## Prevalence                 0.07193
## Detection Rate             0.02398
## Detection Prevalence       0.10390
## Balanced Accuracy          0.62361
```

Three Class

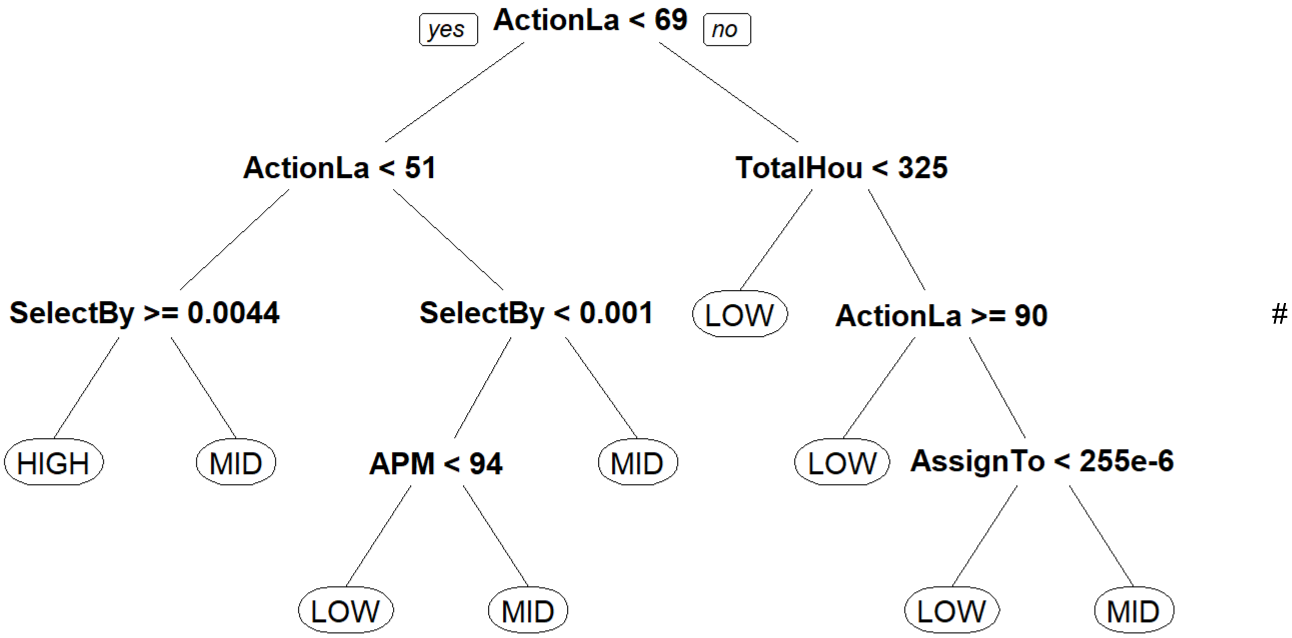
```
makeLeague<-function(x){
  if(x>=1 & x<=3) {return('LOW')}
  else if(x>3 & x<6) {return('MID')}
  else if(x>=6) {return('HIGH')}
}
df$League3<-sapply(df$LeagueIndex,makeLeague)
```

```
set.seed(123)
df3<-subset( df, select = -c(LeagueIndex,LeagueName) )
split<-sample.split(df3$League3,SplitRatio=.7)
train<-subset(df3,split==T)
test<-subset(df3,split==F)
```

```
tree <- rpart(League3 ~ ., method='class',data = train)
print(tree$variable.importance)
```

```
##        ActionLatency                  APM         NumberOfPACs 
##           290.469869           163.254224           149.874068 
##      SelectByHotkeys       GapBetweenPACs           TotalHours 
##           100.882202            88.829068            29.808489 
##      AssignToHotkeys          WorkersMade         ActionsInPAC 
##            29.788745            28.296950             6.905421 
##         HoursPerWeek        UniqueHotkeys   MinimapRightClicks 
##             4.256168             3.796401             1.427551 
## ComplexAbilitiesUsed 
##             1.287757
```

```
prp(tree)
```



LDA with 3 class

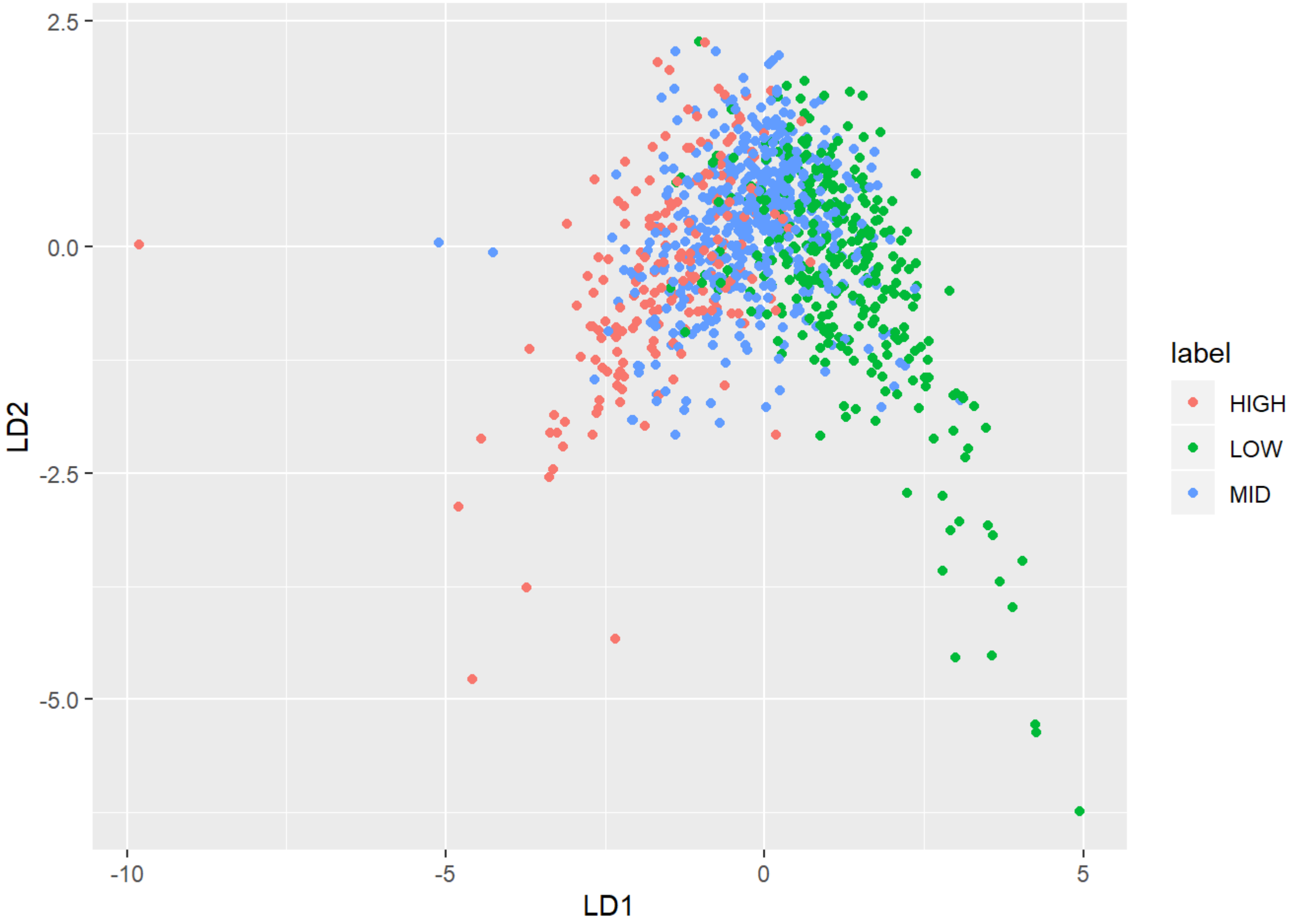
```
lda.pred=lda(League3~.,data=train)
lda.pred
```

```
## Call:
## lda(League3 ~ ., data = train)
##
## Prior probabilities of groups:
##      HIGH       LOW       MID
## 0.1964897 0.3197774 0.4837329
##
## Group means:
##           Age HoursPerWeek TotalHours       APM SelectByHotkeys
## HIGH 20.62527     21.53813  1001.3725 161.21118     0.007768743
## LOW  22.31325     13.66265   396.4833  80.97109     0.001798621
## MID  21.81239     15.07257   669.2867 119.09114     0.004099626
##      AssignToHotkeys UniqueHotkeys MinimapAttacks MinimapRightClicks
## HIGH    0.0005254050      5.453159   1.696700e-04       0.0004718415
## LOW     0.0002463032      3.518072   4.878552e-05       0.0003055715
## MID     0.0003769588      4.335398   9.859250e-05       0.0003970688
##      NumberOfPACs GapBetweenPACs ActionLatency ActionsInPAC
## HIGH  0.004230874       30.40525      48.87134     5.514926
## LOW   0.002768619       51.21784      79.33933     5.069330
## MID   0.003515721       37.58463      60.44756     5.393560
##      TotalMapExplored  WorkersMade UniqueUnitsMade ComplexUnitsMade
## HIGH         24.54248 0.0011947894        6.808279     7.674661e-05
## LOW          20.01205 0.0008398422        6.160643     3.157717e-05
## MID          22.54690 0.0010805373        6.620354     7.066367e-05
##      ComplexAbilitiesUsed
## HIGH         0.0001861491
## LOW          0.0000864652
## MID          0.0001566727
##
## Coefficients of linear discriminants:
##                                LD1           LD2
## Age                  -9.717403e-03  4.897444e-02
## HoursPerWeek         -6.229100e-03 -3.235613e-02
## TotalHours           -3.033711e-04  2.572641e-05
## APM                   2.052005e-03 -2.100185e-02
## SelectByHotkeys      -6.158618e+01  4.716885e+01
## AssignToHotkeys      -1.244125e+03 -3.219542e+02
## UniqueHotkeys        -3.409502e-02 -5.418959e-02
## MinimapAttacks       -1.323885e+03 -1.016802e+03
## MinimapRightClicks    1.152001e+01 -1.180814e+00
## NumberOfPACs         -4.610065e+02 -6.836806e+02
## GapBetweenPACs        6.142940e-03 -1.422594e-02
## ActionLatency         2.448082e-02 -7.387326e-02
## ActionsInPAC         -8.735300e-02  9.027744e-02
## TotalMapExplored      7.618260e-03  5.169887e-03
## WorkersMade          -1.461792e+02  4.789116e+02
## UniqueUnitsMade       3.483241e-02  4.588062e-02
## ComplexUnitsMade     -6.781899e+02  2.562432e+03
## ComplexAbilitiesUsed  1.287289e+02 -2.458063e+02
##
## Proportion of trace:
##    LD1    LD2
## 0.9284 0.0716
```

```
ldatest=predict(lda.pred,test)

table(ldatest$class,test$League3)
```

```
## 
##        HIGH LOW MID
##   HIGH   91   2  52
##   LOW     2 198  65
##   MID   104 120 367
```

```
z <- data.frame(ldatest$x, label=test$League3)

ggplot(z[z$label=='HIGH' | z$label=="MID" | z$label=="LOW",], aes(LD1,LD2))+geom_point(aes(col=l
abel))
```



```
mean(ldatest$class==test$League3)
```

```
## [1] 0.6553447
```

QDA with 3 class

```
qda.pred=qda(League3~.,data=train)
qda.pred
```

```
## Call:
## qda(League3 ~ ., data = train)
##
## Prior probabilities of groups:
##      HIGH       LOW       MID
## 0.1964897 0.3197774 0.4837329
##
## Group means:
##           Age HoursPerWeek TotalHours       APM SelectByHotkeys
## HIGH 20.62527     21.53813  1001.3725 161.21118     0.007768743
## LOW  22.31325     13.66265   396.4833  80.97109     0.001798621
## MID  21.81239     15.07257   669.2867 119.09114     0.004099626
##      AssignToHotkeys UniqueHotkeys MinimapAttacks MinimapRightClicks
## HIGH    0.0005254050      5.453159   1.696700e-04       0.0004718415
## LOW     0.0002463032      3.518072   4.878552e-05       0.0003055715
## MID     0.0003769588      4.335398   9.859250e-05       0.0003970688
##      NumberOfPACs GapBetweenPACs ActionLatency ActionsInPAC
## HIGH  0.004230874       30.40525      48.87134     5.514926
## LOW   0.002768619       51.21784      79.33933     5.069330
## MID   0.003515721       37.58463      60.44756     5.393560
##      TotalMapExplored  WorkersMade UniqueUnitsMade ComplexUnitsMade
## HIGH         24.54248 0.0011947894        6.808279     7.674661e-05
## LOW          20.01205 0.0008398422        6.160643     3.157717e-05
## MID          22.54690 0.0010805373        6.620354     7.066367e-05
##      ComplexAbilitiesUsed
## HIGH         0.0001861491
## LOW          0.0000864652
## MID          0.0001566727
```

```
qdatest=predict(qda.pred,test)

table(qdatest$class,test$League3)
```

```
##
##        HIGH LOW MID
##   HIGH  101   3  67
##   LOW    14 257 172
##   MID    82  60 245
```

```
mean(qdatest$class==test$League3)
```

```
## [1] 0.6023976
```

Random Forest with 3 class

```
train$League3 = factor(train$League3)
rf.model<-randomForest(League3 ~ . , data = train,importance = TRUE)
print(rf.model)
```

```
## 
## Call:
##  randomForest(formula = League3 ~ ., data = train, importance = TRUE) 
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
## 
##         OOB estimate of  error rate: 33.95%
## Confusion matrix:
##      HIGH LOW MID class.error
## HIGH  219   6 234   0.5228758
## LOW     3 486 258   0.3493976
## MID   115 177 838   0.2584071
```

```
predictionRF<-as.data.frame(predict(rf.model,test))
colnames(predictionRF)<-c('res')
test$League3 <- as.factor(test$League3)
confusionMatrix(predictionRF$res,test$League3)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH  101   1  42
##       LOW     0 220  72
##       MID    96  99 370
## 
## Overall Statistics
##                                           
##                Accuracy : 0.6903          
##                  95% CI : (0.6606, 0.7189)
##     No Information Rate : 0.4835          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.4886          
##                                           
##  Mcnemar's Test P-Value : 7.889e-06       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity               0.5127     0.6875     0.7645
## Specificity               0.9465     0.8943     0.6228
## Pos Pred Value            0.7014     0.7534     0.6549
## Neg Pred Value            0.8880     0.8590     0.7385
## Prevalence                0.1968     0.3197     0.4835
## Detection Rate            0.1009     0.2198     0.3696
## Detection Prevalence      0.1439     0.2917     0.5644
## Balanced Accuracy         0.7296     0.7909     0.6936
```

SVM linear B4 tuned (3 class)

```
svm_linear<-svm(League3~., data=train, kernel='linear', cost=0.01)
summary(svm_linear)
```

```
##
## Call:
## svm(formula = League3 ~ ., data = train, kernel = "linear", cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.01
##
## Number of Support Vectors:  1902
##
##  ( 944 540 418 )
##
##
## Number of Classes:  3
##
## Levels:
##  HIGH LOW MID
```

```
# Prediction
pred_train_linear <- svm_linear$fitted
pred_test_linear <- predict(svm_linear,test)

# Error
conf_mtrx_train <- confusionMatrix(train$League3,pred_train_linear)
cat("Linear train error rate(B4 tuned):",1-conf_mtrx_train$overall[1],"\n\n")
```

```
## Linear train error rate(B4 tuned): 0.328339
```

```
conf_mtrx_test <- confusionMatrix(test$League3,pred_test_linear)
cat("Linear test error rate(B4 tuned):",1-conf_mtrx_test$overall[1],"\n\n")
```

```
## Linear test error rate(B4 tuned): 0.3336663
```

```
print(conf_mtrx_train)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH  175   6 278
##       LOW     2 489 256
##       MID    65 160 905
## 
## Overall Statistics
## 
##                Accuracy : 0.6717          
##                  95% CI : (0.6522, 0.6907)
##     No Information Rate : 0.616           
##     P-Value [Acc > NIR] : 1.313e-08       
## 
##                   Kappa : 0.4454          
## 
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity              0.72314     0.7466     0.6289
## Specificity              0.86437     0.8465     0.7492
## Pos Pred Value           0.38126     0.6546     0.8009
## Neg Pred Value           0.96430     0.8955     0.5572
## Prevalence               0.10360     0.2804     0.6160
## Detection Rate           0.07491     0.2093     0.3874
## Detection Prevalence     0.19649     0.3198     0.4837
## Balanced Accuracy        0.79376     0.7965     0.6890
```

```
print(conf_mtrx_test)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH   79   0 118
##       LOW     1 206 113
##       MID    32  70 382
## 
## Overall Statistics
##                                           
##                Accuracy : 0.6663          
##                  95% CI : (0.6362, 0.6955)
##     No Information Rate : 0.6124          
##     P-Value [Acc > NIR] : 0.0002312       
##                                           
##                   Kappa : 0.438           
##                                           
##  Mcnemar's Test P-Value : 4.803e-13       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity              0.70536     0.7464     0.6232
## Specificity              0.86727     0.8428     0.7371
## Pos Pred Value           0.40102     0.6437     0.7893
## Neg Pred Value           0.95896     0.8972     0.5532
## Prevalence               0.11189     0.2757     0.6124
## Detection Rate           0.07892     0.2058     0.3816
## Detection Prevalence     0.19680     0.3197     0.4835
## Balanced Accuracy        0.78631     0.7946     0.6801
```

SVM linear after tuned (3 class)

```
# Tuned model
tune_linear <- tune(svm, League3~., data=train, kernel='linear', range = list(cost=seq(0.01,2.5,
0.5)))

# Prediction
pred_train_linear_tuned <- tune_linear$best.model$fitted
pred_test_linear_tuned <- predict(tune_linear$best.model,test)

# Error
conf_mtrx_train_tuned <- confusionMatrix(train$League3,pred_train_linear_tuned)
cat("Radial tuned train error rate(after tuned):",1-conf_mtrx_train_tuned$overall[1],"\n\n")
```

```
## Radial tuned train error rate(after tuned): 0.322774
```

```
conf_mtrx_test_tuned <- confusionMatrix(test$League3,pred_test_linear_tuned)
cat("Radial tuned test error rate(after tuned):",1-conf_mtrx_test_tuned$overall[1])
```

```
## Radial tuned test error rate(after tuned): 0.3246753
```

```
# SVM radial B4 tune (3 class)
```

```
print(conf_mtrx_train_tuned)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH  198   3 258
##       LOW     2 500 245
##       MID    81 165 884
## 
## Overall Statistics
## 
##                Accuracy : 0.6772
##                  95% CI : (0.6578, 0.6962)
##     No Information Rate : 0.5938
##     P-Value [Acc > NIR] : < 2.2e-16
## 
##                   Kappa : 0.46
## 
##  Mcnemar's Test P-Value : < 2.2e-16
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity              0.70463     0.7485     0.6373
## Specificity              0.87299     0.8519     0.7408
## Pos Pred Value           0.43137     0.6693     0.7823
## Neg Pred Value           0.95578     0.8943     0.5829
## Prevalence               0.12029     0.2860     0.5938
## Detection Rate           0.08476     0.2140     0.3784
## Detection Prevalence     0.19649     0.3198     0.4837
## Balanced Accuracy        0.78881     0.8002     0.6891
```

```
print(conf_mtrx_test_tuned)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH   88   0 109
##       LOW     3 216 101
##       MID    39  73 372
## 
## Overall Statistics
##                                           
##                Accuracy : 0.6753          
##                  95% CI : (0.6453, 0.7043)
##     No Information Rate : 0.5814          
##     P-Value [Acc > NIR] : 6.330e-10       
##                                           
##                   Kappa : 0.4598          
##                                           
##  Mcnemar's Test P-Value : 7.896e-09       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity              0.67692     0.7474     0.6392
## Specificity              0.87486     0.8539     0.7327
## Pos Pred Value           0.44670     0.6750     0.7686
## Neg Pred Value           0.94776     0.8928     0.5938
## Prevalence               0.12987     0.2887     0.5814
## Detection Rate           0.08791     0.2158     0.3716
## Detection Prevalence     0.19680     0.3197     0.4835
## Balanced Accuracy        0.77589     0.8007     0.6859
```

SVM radial b4 tuned (3 class)

```
svm_radial<-svm(League3~., data=train, kernel='radial', cost=0.01)
summary(svm_radial)
```

```
##
## Call:
## svm(formula = League3 ~ ., data = train, kernel = "radial", cost = 0.01)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  0.01
##
## Number of Support Vectors:  2218
##
##  ( 1012 747 459 )
##
##
## Number of Classes:  3
##
## Levels:
##  HIGH LOW MID
```

```
# Prediction
pred_train_radial <- svm_radial$fitted
pred_test_radial <- predict(svm_radial,test)

# Error
conf_mtrx_train <- confusionMatrix(train$League3,pred_train_radial)
cat("Radial train error rate(B4 tuned):",1-conf_mtrx_train$overall[1],"\n\n")
```

```
## Radial train error rate(B4 tuned): 0.4888699
```

```
conf_mtrx_test <- confusionMatrix(test$League3,pred_test_radial)
cat("Radial test error rate(B4 tuned):",1-conf_mtrx_test$overall[1],"\n\n")
```

```
## Radial test error rate(B4 tuned): 0.4985015
```

```
print(conf_mtrx_train)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH  LOW  MID
##       HIGH    0    0  459
##       LOW     0   66  681
##       MID     0    2 1128
## 
## Overall Statistics
## 
##                Accuracy : 0.5111          
##                  95% CI : (0.4906, 0.5316)
##     No Information Rate : 0.9709          
##     P-Value [Acc > NIR] : 1               
## 
##                   Kappa : 0.0617          
## 
##  Mcnemar's Test P-Value : NA              
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity                   NA    0.97059    0.49735
## Specificity               0.8035    0.69974    0.97059
## Pos Pred Value                NA    0.08835    0.99823
## Neg Pred Value                NA    0.99874    0.05473
## Prevalence                0.0000    0.02911    0.97089
## Detection Rate            0.0000    0.02825    0.48288
## Detection Prevalence      0.1965    0.31978    0.48373
## Balanced Accuracy             NA    0.83516    0.73397
```

```
print(conf_mtrx_test)
```

```
## Confusion Matrix and Statistics
## 
##             Reference
## Prediction HIGH LOW MID
##       HIGH    0   0 197
##       LOW     0  21 299
##       MID     0   3 481
## 
## Overall Statistics
## 
##                Accuracy : 0.5015
##                  95% CI : (0.4701, 0.5329)
##     No Information Rate : 0.976
##     P-Value [Acc > NIR] : 1
## 
##                   Kappa : 0.0421
## 
##  Mcnemar's Test P-Value : NA
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity                   NA    0.87500    0.49232
## Specificity               0.8032    0.69396    0.87500
## Pos Pred Value                NA    0.06563    0.99380
## Neg Pred Value                NA    0.99559    0.04062
## Prevalence                0.0000    0.02398    0.97602
## Detection Rate            0.0000    0.02098    0.48052
## Detection Prevalence      0.1968    0.31968    0.48352
## Balanced Accuracy             NA    0.78448    0.68366
```

SVM radial after tuned (3 class)

```
# Tuned model
tune_radial <- tune(svm, League3~., data=train, kernel='radial', range = list(cost=seq(0.01,10,
0.1)))

# Prediction
pred_train_radial_tuned <- tune_radial$best.model$fitted
pred_test_radial_tuned <- predict(tune_radial$best.model,test)

# Error
conf_mtrx_train_tuned <- confusionMatrix(train$League3,pred_train_radial_tuned)
cat("Radial tuned train error rate(after tuned):",1-conf_mtrx_train_tuned$overall[1],"\n\n")
```

```
## Radial tuned train error rate(after tuned): 0.2478596
```

```
conf_mtrx_test_tuned <- confusionMatrix(test$League3,pred_test_radial_tuned)
cat("Radial tuned test error rate(after tuned):",1-conf_mtrx_test_tuned$overall[1])
```

```
## Radial tuned test error rate(after tuned): 0.3226773
```

```
print(conf_mtrx_train_tuned)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH  264   3 192
##       LOW     5 535 207
##       MID    43 129 958
## 
## Overall Statistics
##                                           
##                Accuracy : 0.7521          
##                  95% CI : (0.7341, 0.7695)
##     No Information Rate : 0.5809          
##     P-Value [Acc > NIR] : < 2.2e-16       
##                                           
##                   Kappa : 0.5879          
##                                           
##  Mcnemar's Test P-Value : < 2.2e-16       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity               0.8462     0.8021     0.7060
## Specificity               0.9037     0.8730     0.8243
## Pos Pred Value            0.5752     0.7162     0.8478
## Neg Pred Value            0.9744     0.9169     0.6692
## Prevalence                0.1336     0.2855     0.5809
## Detection Rate            0.1130     0.2290     0.4101
## Detection Prevalence      0.1965     0.3198     0.4837
## Balanced Accuracy         0.8749     0.8375     0.7651
```

```
print(conf_mtrx_test_tuned)
```

```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction HIGH LOW MID
##       HIGH   95   1 101
##       LOW     1 219 100
##       MID    51  69 364
## 
## Overall Statistics
## 
##                Accuracy : 0.6773          
##                  95% CI : (0.6474, 0.7062)
##     No Information Rate : 0.5644          
##     P-Value [Acc > NIR] : 1.733e-13       
## 
##                   Kappa : 0.4674          
## 
##  Mcnemar's Test P-Value : 6.118e-05       
## 
## Statistics by Class:
## 
##                      Class: HIGH Class: LOW Class: MID
## Sensitivity              0.64626     0.7578     0.6442
## Specificity              0.88056     0.8581     0.7248
## Pos Pred Value           0.48223     0.6844     0.7521
## Neg Pred Value           0.93532     0.8972     0.6112
## Prevalence               0.14685     0.2887     0.5644
## Detection Rate           0.09491     0.2188     0.3636
## Detection Prevalence     0.19680     0.3197     0.4835
## Balanced Accuracy        0.76341     0.8080     0.6845
```