R Notebook

Code

Hide

```
library(e1071)
library(MASS)
```

First we load the data, in this case a dataset based off of Twitch game statistics, and split into train and test

Hide

```
df <- read.csv("Twitch_game_data.csv")
str(df)
```

```
'data.frame':   14400 obs. of  12 variables:
 $ Rank            : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Game            : chr  "League of Legends" "Counter-Strike: Global Offensive" "Dota 2" "Heart
hstone" ...
 $ Month           : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Year            : int  2016 2016 2016 2016 2016 2016 2016 2016 2016 2016 ...
 $ Hours_watched   : int  94377226 47832863 45185893 39936159 16153057 10231056 8771452 7894571
7688369 6988475 ...
 $ Hours_Streamed  : chr  "1362044 hours" "830105 hours" "433397 hours" "235903 hours" ...
 $ Peak_viewers    : int  530270 372654 315083 131357 71639 64432 46130 41588 84051 145728 ...
 $ Peak_channels   : int  2903 2197 1100 517 3620 1538 1180 460 148 756 ...
 $ Streamers       : int  129172 120849 44074 36170 214054 88820 33375 21396 10779 46462 ...
 $ Avg_viewers     : int  127021 64378 60815 53749 21740 13769 11805 10625 10347 9405 ...
 $ Avg_channels    : int  1833 1117 583 317 1549 659 461 276 71 274 ...
 $ Avg_viewer_ratio: num  69.3 57.6 104.3 169.3 14 ...
```

Hide

```
names(df)[names(df) == "Rank"] <- "game_rank" ## rename column to not overlap name

df <- df[,c(1,5,7,8,9,10,11)] #grabs Rank, Hours_watched, Peak_viewers, Peak_channels, Streamer
s, Avg_viewers, and Avg_channels

set.seed(1234)
i <- sample(1:nrow(df), .8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]

sapply(df, function(x) sum(is.na(x)==TRUE))
```

```
    game_rank Hours_watched  Peak_viewers Peak_channels     Streamers   Avg_viewers  Avg_channel
s
            0             0             0             0             0             0
0
```

We run some basic statistics on the data, getting a sense of the scale of the data

Hide

```
summary(train)
```

```
   game_rank     Hours_watched        Peak_viewers      Peak_channels        Streamers          Avg
_viewers
 Min.   :  1.0   Min.   :    89811   Min.   :    441   Min.   :     1.0   Min.   :      0   Min.
:   120
 1st Qu.: 51.0   1st Qu.:   366018   1st Qu.:   8335   1st Qu.:    51.0   1st Qu.:   1485   1st
Qu.:   502
 Median :101.0   Median :   818609   Median :  20374   Median :   122.0   Median :   4099   Medi
an :  1123
 Mean   :100.8   Mean   :  4806836   Mean   :  56702   Mean   :   610.4   Mean   :  17451   Mean
:  6589
 3rd Qu.:151.0   3rd Qu.:  2331542   3rd Qu.:  46694   3rd Qu.:   313.2   3rd Qu.:  10708   3rd
Qu.:  3183
 Max.   :200.0   Max.   :344551979   Max.   :3123208   Max.   :129860.0   Max.   :1013029   Max.
:479209
  Avg_channels
 Min.   :    0
 1st Qu.:   16
 Median :   43
 Mean   :  218
 3rd Qu.:  122
 Max.   :13789
```

Hide

```
mean(train$Hours_watched)
```

```
[1] 4806836
```

Hide

```
mean(train$Peak_viewers)
```

```
[1] 56701.75
```

Hide

```
mean(train$Peak_channels)
```

```
[1] 610.3609
```

Hide

```
mean(train$Streamers)
```

```
[1] 17450.99
```

Hide

```
mean(train$Avg_viewers)
```

```
[1] 6588.555
```

Hide

```
mean(train$Avg_channels)
```

```
[1] 218.028
```

Hide

```
median(train$Hours_watched)
```

```
[1] 818609
```

Hide

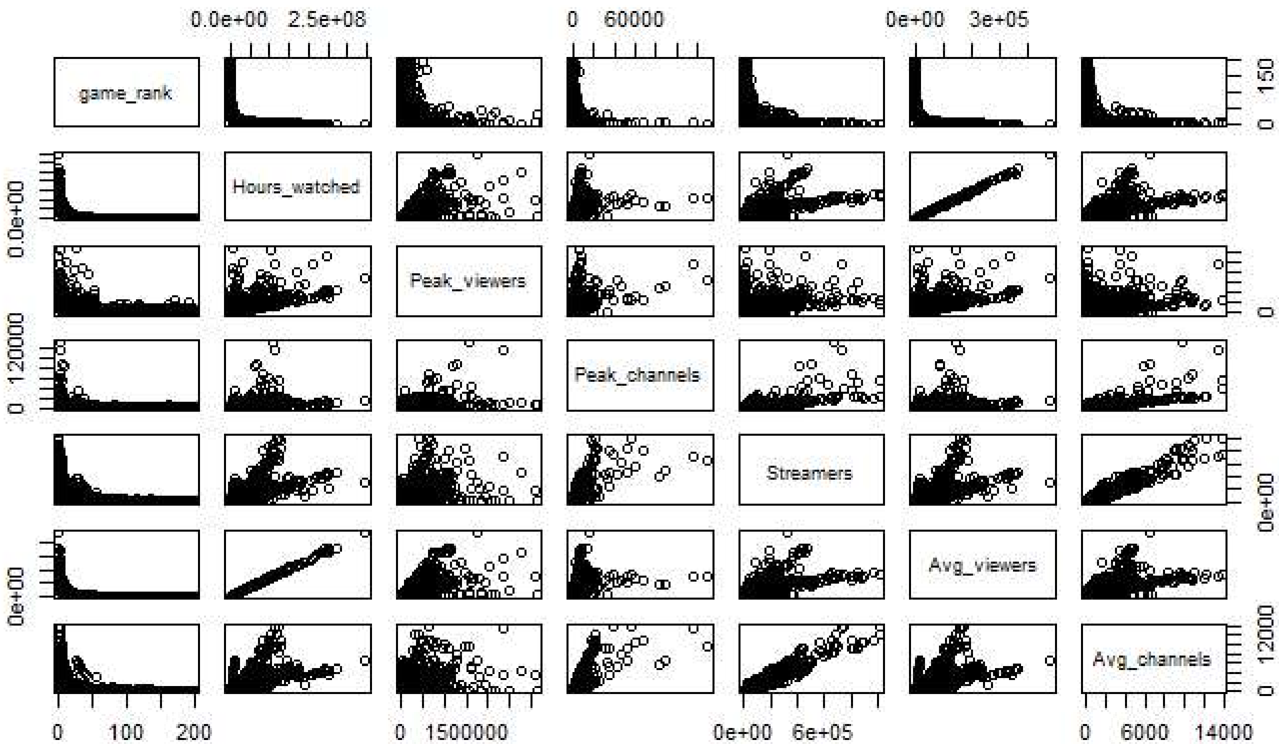
```
median(train$Avg_viewers)
```

```
[1] 1123
```

We graph the pairs for the data

Hide

```
pairs(train)
```



What we're looking to do is to see if we can judge the Rank of a game on Twitch based off of all the numerical metrics

linear kernel

Hide

```
svm1 <- svm(game_rank~., data=train, kernel="linear",cost=10,scale=TRUE)
summary(svm1)
```

```


Call:
svm(formula = game_rank ~ ., data = train, kernel = "linear", cost = 10, scale = TRUE)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  linear
       cost:  10
      gamma:  0.1666667
    epsilon:  0.1


Number of Support Vectors:  10748
```

Hide

```
pred <- predict(svm1, newdata=test)
cor_svm1 <- cor(pred, test$game_rank)
mse_svm1 <- mean((pred - test$game_rank)^2)
```

polynomial

Hide

```
svm2 <- svm(game_rank~., data=train, kernel="polynomial", cost=10, scale=TRUE)
```

```
WARNING: reaching max number of iterations
```

Hide

```
summary(svm2)
```

```
Call:
svm(formula = game_rank ~ ., data = train, kernel = "polynomial", cost = 10, scale = TRUE)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  polynomial
       cost:  10
     degree:  3
      gamma:  0.1666667
     coef.0:  0
    epsilon:  0.1


Number of Support Vectors:  10814
```

Hide

```
pred <- predict(svm2, newdata=test)
cor_svm2 <- cor(pred, test$game_rank)
mse_svm2 <- mean((pred - test$game_rank)^2)
```

radial

Hide

```
svm3 <- svm(game_rank~., data=train, kernel="radial", cost=10, gamma=1, scale=TRUE)
summary(svm3)
```

```
Call:
svm(formula = game_rank ~ ., data = train, kernel = "radial", cost = 10, gamma = 1, scale = TRU
E)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  10
      gamma:  1
    epsilon:  0.1


Number of Support Vectors:  9636
```

Hide

```
pred <- predict(svm3, newdata=test)
cor_svm3 <- cor(pred, test$game_rank)
mse_svm3 <- mean((pred-test$game_rank)^2)
```

cor and mse

Hide

```
print(paste("cor1=",cor_svm1))
```

```
[1] "cor1= 0.388576275050731"
```

Hide

```
print(paste("mse1=",mse_svm1))
```

```
[1] "mse1= 2949.66554932435"
```

Hide

```
print(paste("cor2=",cor_svm2))
```

```
[1] "cor2= 0.0664891059569638"
```

Hide

```
print(paste("mse2=",mse_svm2))
```

```
[1] "mse2= 11430.3241648377"
```

Hide

```
print(paste("cor3=",cor_svm3))
```

```
[1] "cor3= 0.842063777089085"
```

Hide

```
print(paste("mse3=",mse_svm3))
```

```
[1] "mse3= 955.599743917619"
```

Analysis

So it seems like the radial SVM regression was the most successful in finding a correlation, as compared to linear and polynomial, it is far closer to the threshold of a correlation of 1. This is likely due to radial kernal SVM including an additional hyperparameter. Polynomial SVM is really inaccurate most likely due to the dataset including a fair bit of variance.