Model Building for NFL Winners - part1

Build multiple predictive models by loading **DifferentialStats.tsv** that was generated in **DataProcessing.tsv**.

```
DifferentialStats <- read.table("./DifferentialStats.tsv",, header =T)
head(DifferentialStats)
```

The first goal is to try and just predict the winner. This will all be done with respect to the Home team, by creating a new variable called "Winner", coded with a 1 if the HomeTeam won or a 0 if the AwayTeam won. I'll create a train/dev/test split. The Train and Dev sets will be generated from 1950 through the 2016 season. The test set will be whatever games are in this dataset from the 2017 season.

```
DifferentialStats$date <- as.character(DifferentialStats$date)
DifferentialStats$date <- as.Date(DifferentialStats$date, "%Y-%m-%d")
DifferentialStats$Winner <- ifelse(DifferentialStats$Delta_Score > 0, 1,0)
TrainDev <- subset(DifferentialStats, date < "2017-03-01", select = -c(Delta_Score))
Test <- subset(DifferentialStats, date > "2017-03-01", select = -c(Delta_Score))
TrainDev <- subset(TrainDev, select = -c(date))
Test <- subset(Test, select = -c(date))
```

I'm going to try and build the following models:

1. RandomForest
2. XG Boost
3. ExtraTrees
4. SVM
5. Logistic Regression

```
#set the random seed generator
set.seed(1)
#shuffle the Train/Dev TrainingSet
TrainDevShuffled <- TrainDev[sample(nrow(TrainDev)),]
#90% train
xtrain <- head(TrainDevShuffled[1:35],nrow(TrainDevShuffled) *.9)
ytrain <- head(TrainDevShuffled[c(36)],nrow(TrainDevShuffled)*.9)
#10%Dev
xdev <- tail(TrainDevShuffled[1:35],nrow(TrainDevShuffled) *.1)
ydev <- tail(TrainDevShuffled[c(36)], nrow(TrainDevShuffled) * .1)
#Convert the dataframes into matrix
#xtrainMat <- data.matrix(xtrain)
#xdevMat <- data.matrix(xdev)
#ytrainMat <- ytrain
#ydevMat <- data.matrix(ydev)
```

```
library(randomForest)
library(parallel)
library(doParallel)
registerDoParallel(cores = 20)
RFmodel <- foreach(ntree=rep(25,20), .combine = combine, .packages = 'randomForest') %dopar% randomFores
yhatRF <- predict(RFmodel, xdev)
RFAccuracy <- sum(yhatRF == ydev)/length(ydev)*100
RFAccuracy
TruePositives <-
```

```
library(xgboost)
XGBmodel <- xgboost(data = data.matrix(xtrain),label = ytrain$Winner, nrounds = 500, verbose = F,objecti
```

```
yhatXG <- ifelse(predict(XGBmodel, xdevMat) > .5, 1,0)
XGAccuracy <- sum(yhatXG == ydev$Winner)/length(ydev$Winner)*100
XGAccuracy
```

```
library(extraTrees)
XTreesLaborModel <- extraTrees(x = xtrain, y = ytrain$Winner, ntree = 100, mtry = sqrt(ncol(xtrain)), n
yhatXT <- ifelse(predict(XTreesLaborModel, xdev) > .5, 1, 0)
XT_accuracy <- sum(yhatXT  == ydev$Winner)/length(ydev$Winner) *100
XT_accuracy
```

```
library(e1071)
SVMmodel <- svm(x = xtrain,y = ytrain$Winner, type = 'C')
yhatSVM <- predict(SVMmodel, xdev)
SVM_accuracy <- sum(yhatSVM == ydev$Winner)/length(ydev$Winner) * 100
SVM_accuracy
```

```
Train <- cbind(xtrain, ytrain)
Dev <- cbind(xdev, ydev)
LogisticReg <- glm(Winner ~., data = Train, family = binomial(link = "logit"))
yhatLogisticReg <- ifelse(predict(LogisticReg, Dev)> .5,1,0)
LogisticAccuracy <- sum(yhatLogisticReg == ydev$Winner)/length(ydev$Winner) * 100
LogisticAccuracy
```

Model	Accuracy
Random Forest	68.4%
XG boost	66.2%
Extra Trees	68.9%
SVM	59.99%
Logistic Accuracy	70.1%

These results indicate that none of these models are very good. I would have expected to do better considering that we aren't trying to predict the game against the spread. I could look at **precision** and **recall** to try and determine which models are the best, but since we aren't anywhere near a useful accuracy I'm going to keep feature engineering. The best way to do this is to determine what features are the most important ones from the Random Forest model

```
Importance <- as.data.frame(RFmodel$importance)
Importance$Features <- rownames(Importance)
Importance <- Importance[order(-Importance$MeanDecreaseGini),]
head(Importance)
```