Machine Learning Applied to PGA Tour Championship Statistics

Andrew Yanke

10 October, 2021

Contents

0.1 Abstract

The PGA Tour employs a system called ShotLink to generate statistics related to a golfer's course performance. Lasers allow the Tour to measure distances, heights, and angles of golf shots from a variety of course settings, whether they be fairways, bunkers, distances from the pin, and more. This flexibility allows ShotLink to generate hundreds of unique player statistics that are combined to summarize a player's ranking for a single round, tournament, or Tour history. The Official World Golf Ranking (OWGR) is a universally accepted rating system of Tour players based on the weighted two-year average of points that a player receives during events. This report uses random forests and k-nearest neighbors (KNN) classification algorithms to determine the top 10% of players in the OWGR based on data acquired by ShotLink. A total of 116 Tour statistics were scraped from the PGATour.com for all players in the PGA Championship between 2015-2020. Input features (i.e., Tour statistics) to the models were selected using approaches based on principal component analysis (PCA) or feature correlations to the *strokes-gained* metric, a key summary statistic correlated to OWGR. This study focused on which physical parts of the game separate the winners from the losers, so scoring statistics were excluded from the models. The random forests classifier outperformed the KNN algorithm with a true-positive rate of 73% while the same for KNN was 64%. In general, a player's driving distance, number of rounds played, and ability to hit mid-iron shots between 150-225 yards on the fairway

were the most important features for identifying the top and bottom OWGR deciles. Performances around the green (i.e., scrambling) and putting were much less descriptive of OWGR and gained importance only when scrambling from 10-20 yards to the hole away from the fringe or putting within 10 feet. In other words, top players distinguish themselves not by their ability to play tough lies or long putts, but by performances on the tee, mid-fairway, and short putts. These findings flip the old adage of "drive for the show, putt for dough" to "putt for show, hit long and close for dough."

1 Introduction

1.1 Data

PGA Tour statistics were scraped from the PGA Tour statistics web pages using a python scraper that can be cloned from GitHub. This web-scraper creates a .csv file for each Tour statistic among all players in the PGA Tour Championship for a given year. A total of 432 .csv files, or tour statistics, are created for each year. The complete data set is a merge of all these files for the years 2015-2020, representing over 2500 files. The merge and statistic selection processes were completed in Microsoft's Power BI visual interface: a coding solution is simply not feasible given the large number of files with different names and unknown column structures.

The PGA Tour can publish statistics of other statistics. These statistics are often non-intuitive and sometimes meaningless because they are removed to far from a player's physical game. For this reason, these statistics were eliminated prior to merging. The final merged data set contains 1142 rows, each representing a player and Tour year, and 116 columns, each a Tour statistic. The final Official World Golf Ranking (OWGR) for each player, per year, is the measure of success that this report tries to predict.

1.2 Description of Summary Statistics

Tour statistics are categorized according to which part of the course they were acquired. Table 1, below, displays these categories alongside their meanings and aliases used for the remainder of this report. The column *Count* is the quantity of unique Tour statistics (i.e., features) for the corresponding alias used in this report.

Table 1: Divisions of player statistics based on which aspect of the game they were acquired.

Alias	Game.Aspect	Count	Meaning
OTT or OT	Off-the-Tee	10	All shots hit from the tee box on par 4s and 5s.
APR	Approach-the-Green	56	All shots hit after the tee-shot (or teebox on par 3s) until the ball lands on or around the green (ARG) or in the hole.
ARG	Around-the-Green	15	All shots taken where the ball is within 30 yards of the edge of the green.
G or PUTT	Green	16	All putting strokes.
T2G	Tee-to-Green	1	All shots taken from the teebox to the green.
TOT	Total	1	Total shots on a hole.
SC	Golf Scores	9	Scoring statistics such average par 4 scores or birdie percentages.

1.2.1 Relative to Par (RTP)

In general, relative-to-par is the number of strokes above or below par a player scores in a round, but this definition changes when viewing specific game aspects. For instance, a player's RTP score of -0.2 on a 150 yard fairway shot means that the player scores 0.2 strokes under par, on average, on holes where he performs

this shot. This metric helps represent a player's abilities with scores rather than physical measures (e.g., the distance a ball lands from the hole).

1.2.2 Green in Regulation (GIR)

Hitting a green in regulation means that a player lands his ball on the green with at least two fewer shots than par. This statistic is often a percentage and is among the most popular in golf.

1.2.3 Scrambling

Scrambling is the percent of time a player misses the green-in-regulation but still scores par or better. Scrambling statistics from the PGA Tour relate to shots taken around the green (i.e., less than 30 yards away from the green).

1.2.4 Strength of the Field

The strength of the field is an average player's performance in an event versus regular PGA Tour events (e.g., non-majors). The units of measure are how many strokes better, or worse, the players "in the field" (i.e., in the current event) score than the average player in a regular PGA Tour event.

1.2.5 Official World Golf Ranking (OWGR)

The Official World Golf Ranking (OWGR) is refreshed every Monday following completion of the previous week's tournaments. This statistic is the average number of points earned per event in the last 104 weeks (i.e., two years). These points are awarded based upon finish position and the strength of the field. The points are initially worth double their original value and decline gradually over the two-year period. There are eight 13-week periods and points decline by .25 times their value each period.

1.2.6 Strokes Gained (SG)

Strokes gained is a popular statistic that summarizes a player's scoring performance relative to the field for individual aspects of the game. These individual aspects are shown in Table 1. Traditional golf statistics, such as greens in regulation, fail to capture these individual aspects because they often depend on previous shots. To explain by example, a player who shoots 69 on a day when the field average is 72 gains three strokes to SG_{TOT}, or the total strokes gained. The higher the SG, the better the player. The total strokes gained is a summation of similar SG elements from other parts of the game:

$$SG_{TOT} = SG_{OTT} + SG_{APR} + SG_{ARG} + SG_{PUTT}$$

Each SG element,above, is the difference between a player's score and the field average within the defined game aspect (e.g., OTT or APR) for a given hole. To further explain by example, the Tour scoring average on a 450 yard long par 4 is 4.1. Player "X" hits a tee shot that lands on the fairway 115 yards from the hole. On average, players take 2.8 strokes to hole-out from the fairway 115 yards away. The strokes gained on this player's tee-shot is then $SG_{OTT} = +0.30$ because $4.1 - 2.8 = 1.3 - 1 = +0.30$. One is subtracted from the difference between the two averages to account for the player's tee-shot.

Player "X" continues with his approach shot (i.e., from 115 yards) and lands the ball 20ft from the hole on the green. As mentioned, the baseline from 115 yards away in the fairway is 2.8 strokes. The average player takes 1.9 shots to hole-out from 20ft away on the green. The strokes gained on the approach shot is then $SG_{APR} = 2.8 - 1.9 = 0.9 - 1 = -0.1$.

This calculation continues for the remaining SG elements in order to understand a player's final ranking in relation to different parts of the game.

2 Data Processing

2.1 Replace NAs

Tour statistics can be so specific that a player might not be in a position where the statistic applies for a given year. These empty statistics are labeled with `NA` values in the Tour database. These `NA` values were replaced by the corresponding player's performance in the previous year where the statistic was available. If the statistic did not exist for previous years, then the search continued in *future* years (i.e., between 2015-2020). If no statistic is found for a player, then that statistic is replaced by the tour average. There are 287 `NA` values in the data set of 132472 values.

```
## Fill NA's with previous player's performance or tour averages ##

temp<-colnames(source)[colSums(is.na(source))>0] #columns with NA values
mns <- colMeans(source[temp],na.rm=TRUE) #columns averages (i.e., tour averages)

y<-0
## Find number of NA values in data set ##
for (x in temp) {
  y<-y+sum(is.na(source[,x]))
}

## Fill NA values with players' past/future performance
for (x in temp) {

  source<-source %>% group_by(PLAYER_NAME) %>%
    arrange(PLAYER_NAME,desc(Year)) %>% fill(x,.direction="updown") %>% ungroup()

  source[is.na(source[,x]),x]<- as.integer(round(mns[x]))

}
```

2.1.1 Transform Official World Golf Rankings

The Official World Golf Ranking is a continuous scale, so machine learning models can be regression models. This report identifies which aspects of the game describe winning players, so an absolute ranking is not of interest. The OWGR is binned into ten percentiles in order to determine which aspects of the game create the top 10% of players. By transforming the OWGR data into percentiles, the model becomes a classification model. The OWGR can be split into just the decile end-members (i.e., the top and bottom 10%), but this split will run into issues of prevalence when training the models because the bottom 90% will dominate the classification. In other words, the *sensitivity* (the proportion of true positives) of the top 10% will be very low, so the results will not be reliable. For this reason, the training models use all ten percentiles.

```
## Define Bins for OWGR data ##

#get percentiles increments of 0.1
Q<-quantile(source$OWGR,probs=seq(0,1,0.1))
tags <- as.character(c(10:1)) #create labels

#cut up rankings according to quantiles
group_tags<-cut(source$OWGR,breaks=Q,include.lowest = TRUE,right=FALSE,labels=tags)
group_tags<-factor(group_tags,levels = tags,ordered = TRUE)

source <- source %>% mutate(OWGR_pct=group_tags)

#eliminate some redundant columns
```

```
idx_gt<-colnames(source)[str_detect(colnames(source),"^(APR_gt(10|20).+_Fwy)|(ARG_.+_RTP)")]

source<-source %>% select(-all_of(idx_gt))
```

2.2 Determine Normality, Find Outliers, Scale Data

The data set contains multiple data types including distances, scores, percentages, and more. Scaling the data is a necessary step so that features do not weigh the model differently simply because of their orders of magnitude. A conventional scaler is the z-scaler transform that centers each feature, x, around its mean, μ, and scales by the standard deviation, σ:

$$z = \frac{x - \mu}{\sigma}$$

The z-scaler transform works well when the data is normally distributed with few outliers. The PGA Tour data set has more than 100 features (i.e., statistics), so plotting methods are not feasible to assess the normality of each. Instead, the summary statistics *skewness* and *kurtosis* are computed for each feature and plotted against normal values. Hair et al. (2010) and Bryne (2010) argue that data is considered to be approximately normal if skewness is between -2 to +2 and kurtosis is between -7 to +7. A perfectly normal distribution has a skewness of 0 and a kurtosis of 3, so skewness values between 1 and 2 are moderately skewed. Moreover, normally distributed data should contain less than 0.3% outliers, which corresponds to data values that fall further than three standard deviations from the mean. The skewness, kurtosis, and percentage of outliers for each aspect of the game are shown in Figure 1. The red lines delineate normal values. The dashed grey line delimits the upper kurtosis boundary of normality. Most of the data is positively skewed (i.e., weighted to the left) and contains increasing outliers with skewness. Points that fall close to or outside the bounds of normality are labeled in Figure 1.

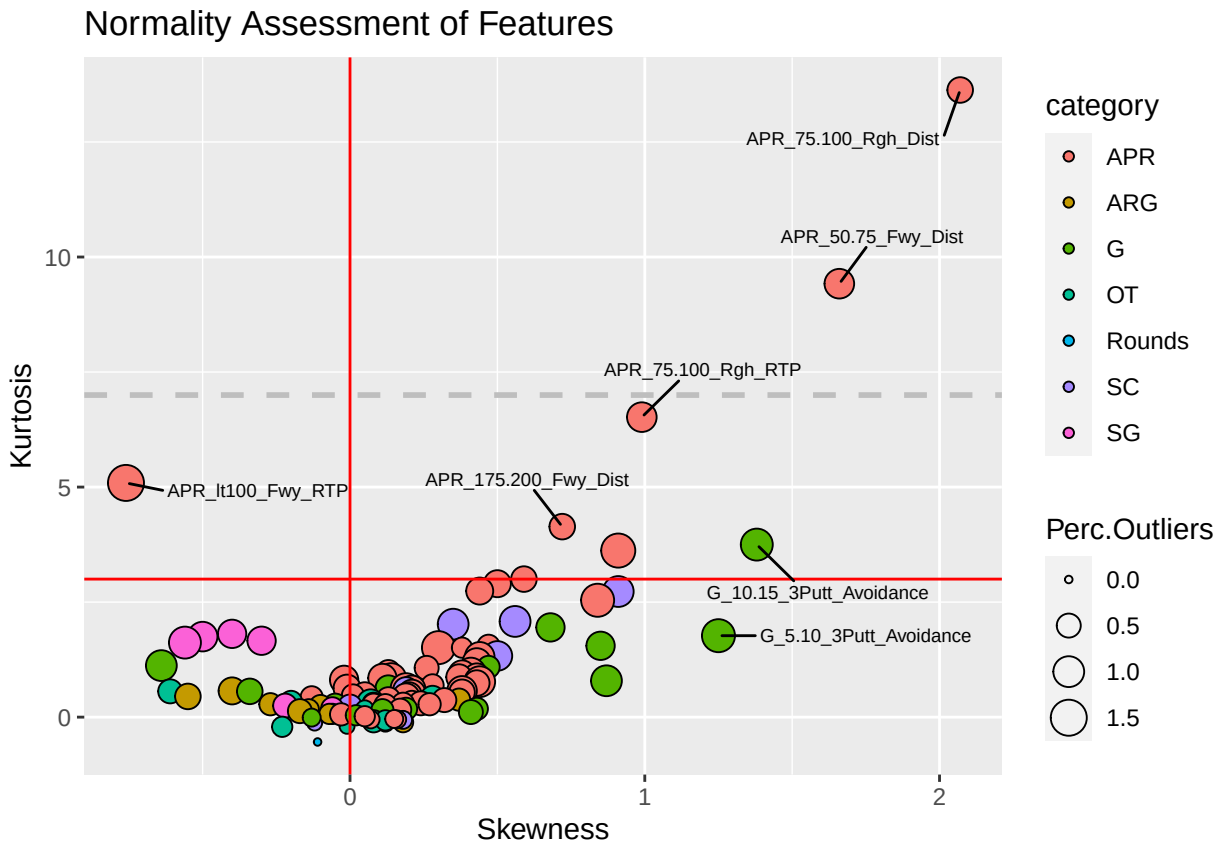

Figure 1: Skewness vs. Kurtosis for all features in PGA TOur dataset. Points are sized by the percentage of outliers identified for each feature. The red lines delimit the skewness and kurtosis for a normal distribution while the grey dashed line is an upper bound to normality. Features with significant deviations are labeled.

As seen in Figure 1, most data show some levels of skewness and kurtosis alongside non-negligible percentages of outliers that strain the assumption of normality. For this reason, the data is scaled with a robust-scaler transform instead of the z-scaler transform. The robust-scaler transform is less sensitive to outliers and non-parametric, so no assumptions of the underlying distributions are made. The robust-scaler centers the data on its median and scales according to the interquartile range.

$$t = \frac{x - median(X)}{IQR(x)}$$

```
## SCALE THE DATA USING A ROBUST SCALER APPROACH ##

scaled<-df %>% as.matrix() %>%
  scale(center=colMedians(.),scale=colIQRs(.)) %>%
  as_tibble() %>% cbind(select(source,OWGR_pct))
```

After scaling, I partitioned the data set into training and test sets with a 50/50 ratio. Fifty-percent of the data is reserved for testing in order to obtain an adequate outcome space for computing result statistics. The data set is not large to begin with, so a small test set would cause the result statistics (e.g., sensitivity and specificity) to lose some robustness.

3 Exploratory Data Analysis

A major part of exploratory data analysis is to identify which features affect the parameters of interest the most. Analysts of the PGA Tour rephrase this question to identify which aspects of the game divide the real winners from the losers. As a first pass, box plots of the the different SG categories against the success statistic, OWGR, show that tee-to-green (T2G) performances are much more descriptive of rankings than putting averages (Figure 2). The top decile of players is OWGR=1 in the following figures.

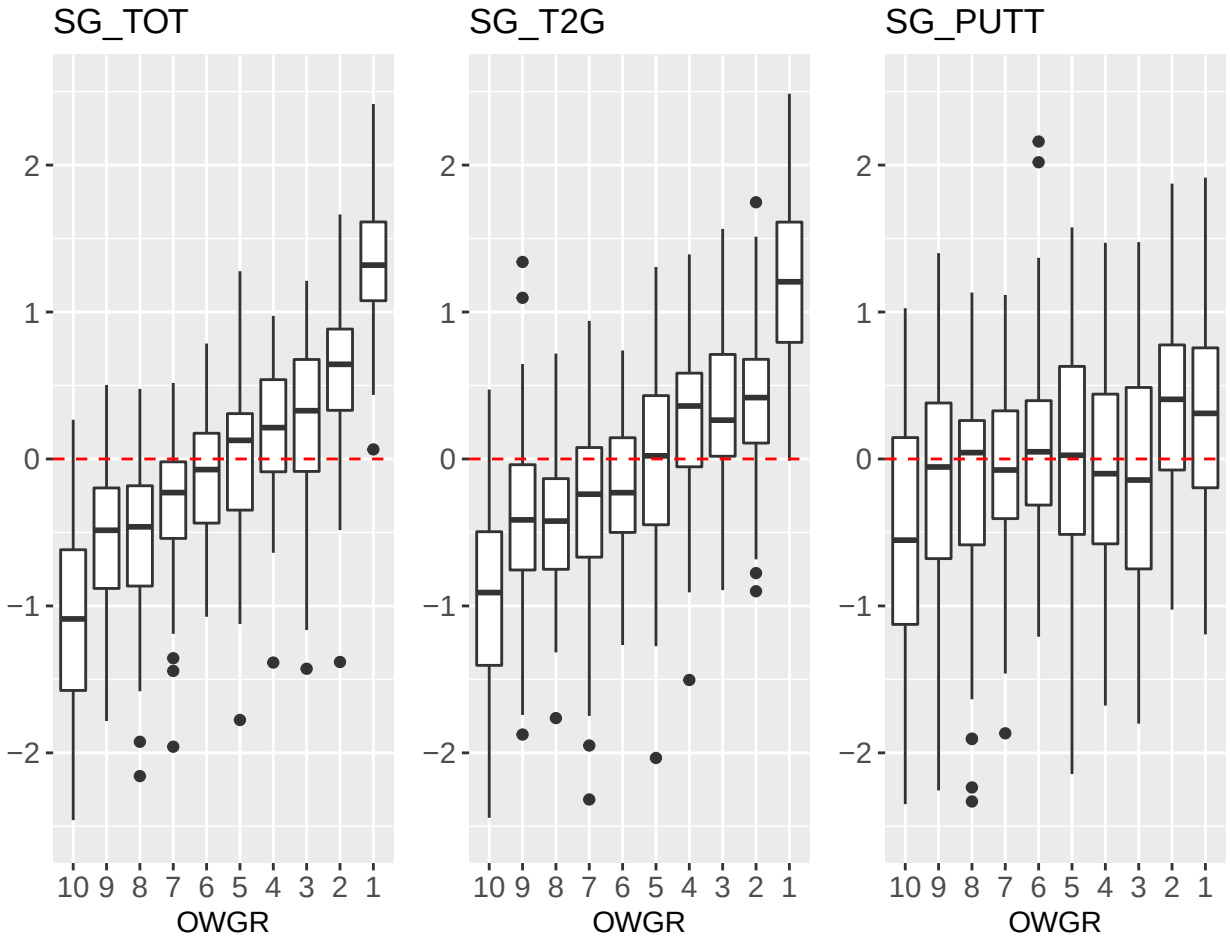

Figure 2: Boxplots of each SG_{TOT}, SG_{T2G}, and SG_{PUTT} against the decile rank within OWGR. Ten is the bottom decile while 1 is the top decile of players. The sum of SG_{T2G} and SG_{PUTT} equals SG_{TOT}, the total strokes-gained.

Figure 2 shows that the strokes-gained on putts have little bearing on OWGR, but the top and bottom deciles display some separation where top players demonstrate better putting averages than bottom players (on a median basis). The magnitude of this separation pales in comparison to that of SG_{T2G}, so statistics from tee-to-green should dominate the model. More importantly, maximum separation occurs between the top and bottom deciles with plenty of overlap in the middle. For this reason, the model should classify the top and bottom deciles of players better than the remainder.

A deeper look into the constituents of SG_{T2G} shows that SG_{APR} and SG_{OTT} discriminate OWGRs the best even though significant overlap exists between the top and bottom deciles (Figure 3). Strokes-gained around the green, SG_{ARG}, appear more correlated with OWGR than SG_{PUTT} by a small margin, but the long-game (i.e., SG_{OTT} and SG_{APR}) still dominates.

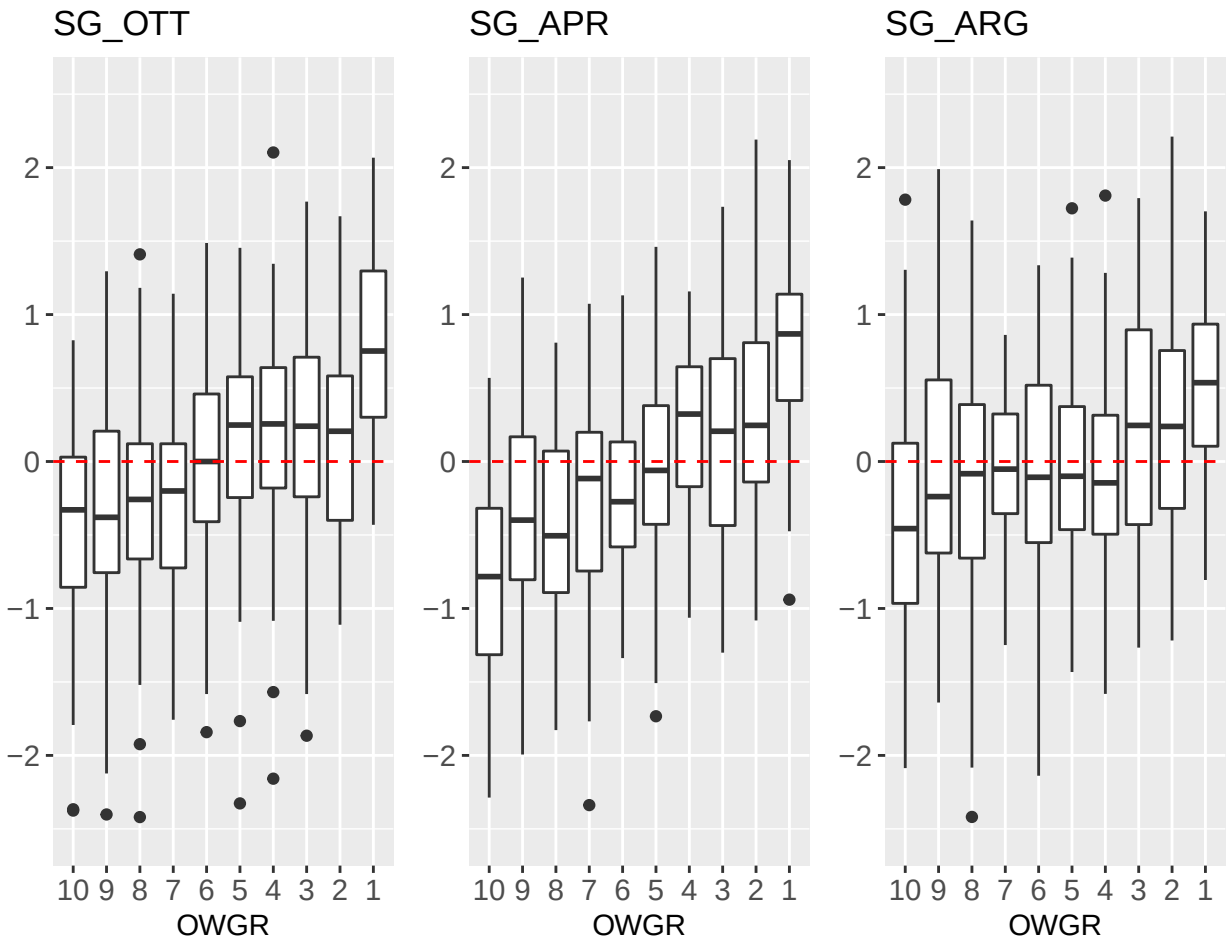

Figure 3: Boxplots of the subcategories to SG_{T2G} against the decile rank within OWGR. Ten is the bottom decile while 1 is the top decile of players.

Another high-level view of the data is the strokes-gained versus the number of rounds played by a golfer from 2015-2020. Figure 4 shows lines of best fit for each of the SG categories against the number of rounds played by a golfer. The relationships are weak, but positive.

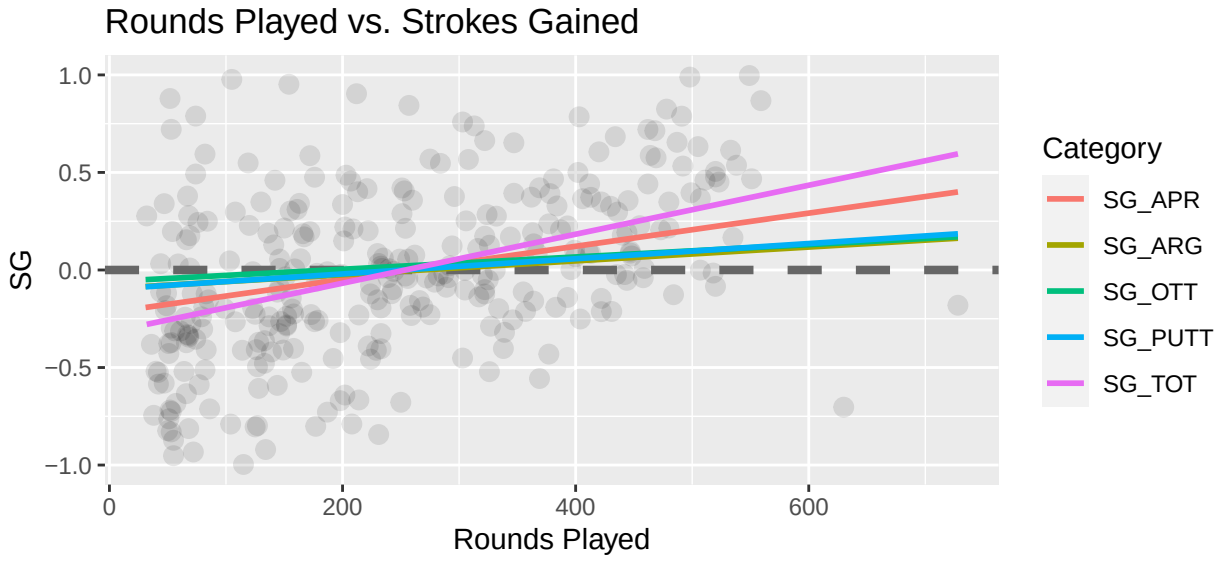

Figure 4: Scatter plot of the SG_{TOT} versus the number of rounds played by a player from 2015-2020. Overlain are lines of best fit for each SG category. The grey dashed-line is the field average (i.e., SG=0).

Strokes-gained on the approach (i.e., SG_{APR}) has the strongest correlation with rounds played, and 250 rounds appears to be a golden-cross where players begin to score better than the field average (i.e., SG=0). Interpretation of Figure 4 can be ambiguous, though, because a player can either improve by playing more rounds, or good players get more opportunities to play. In either case, this relationship can help predict the best players.

3.1 Correlations of PGA Statistics with OWGR and SG

Correlation matrices of Tour statistics with strokes-gained and the OWGR can hint at the most critical game aspects that divide winners from losers. Because OWGR and strokes-gained are ordinal statistics (i.e., they represent rankings), correlation coefficients were calculated using the Spearman's Method instead of the more common Pearson's Method. Spearman's method is also less sensitive to outliers than Pearson's and does not make assumptions of bivariate normality. Both of these characteristics are favorable for the PGA data set.

The following sets of figures show correlation matrices of Tour statistics to SG and OWGR for every game-aspect (i.e., OTT, APR, etc.). The correlations are sorted by their absolute values in order to reveal which Tour statistics are the most indicative of game-aspect performance. Red X's denote correlations that have p-values greater than 0.05, representing NULL significance. Even though some correlations are weak, they can be helpful when training the model.

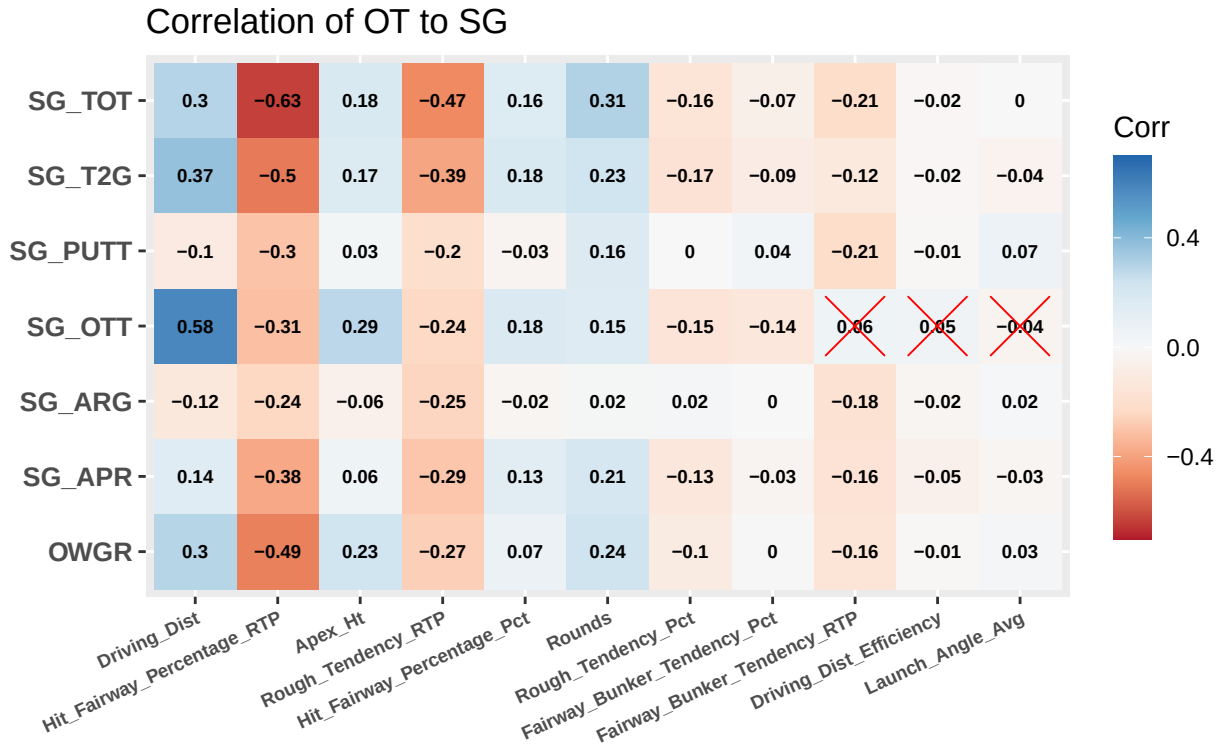

Figure 5: Spearman correlation plot of OTT statistics with SG and OWGC. Correlations are sorted in descending order by the absolute value of the correlation to SG_{OTT}.

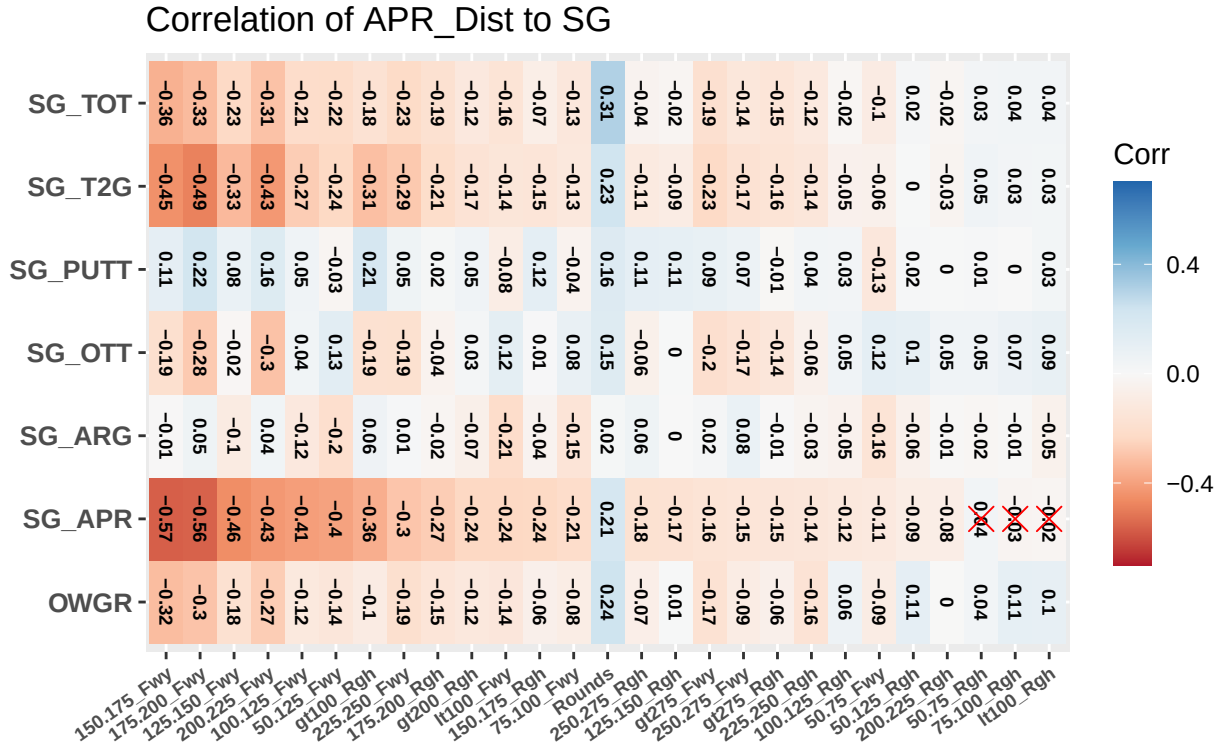

Figure 6: Spearman correlation plot of APR statistics with SG and OWGC. Correlations are sorted in descending order by the absolute value of the correlation to SG_{APR}.

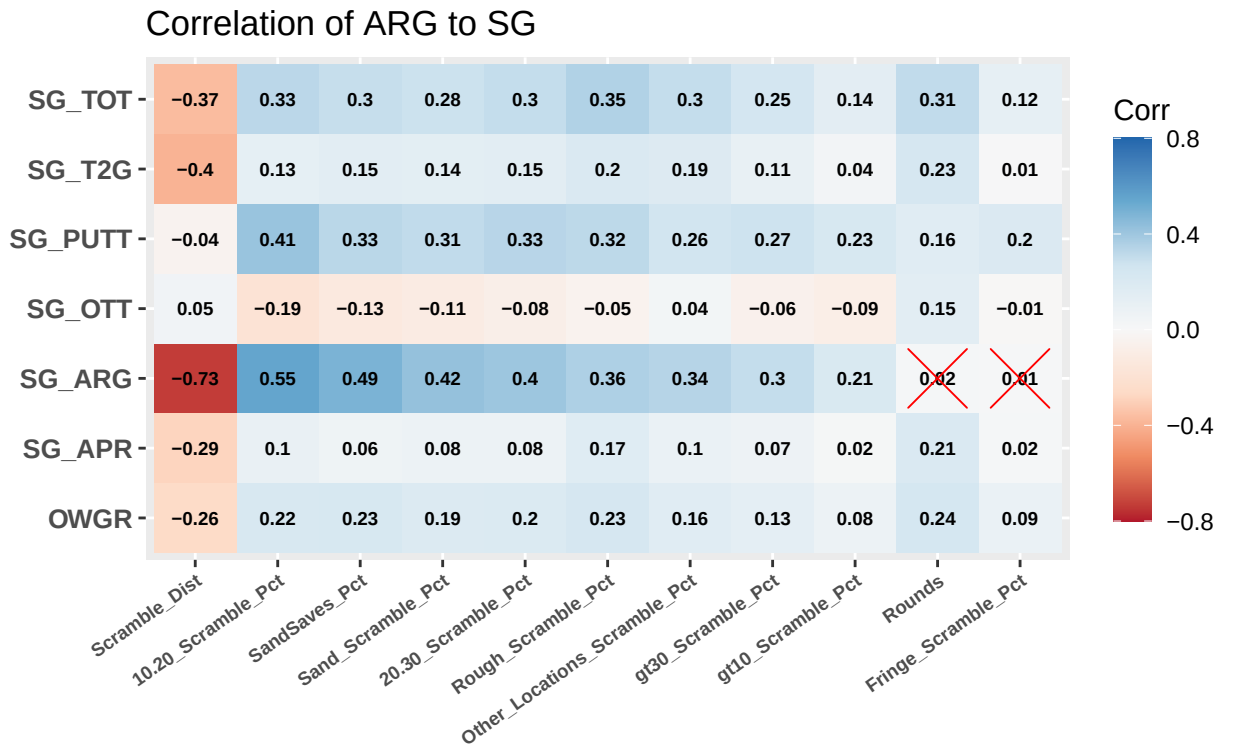

Figure 7: Spearman correlation plot of ARG statistics with SG and OWGC. Correlations are sorted in descending order by the absolute value of the correlation to SG_{ARG}.

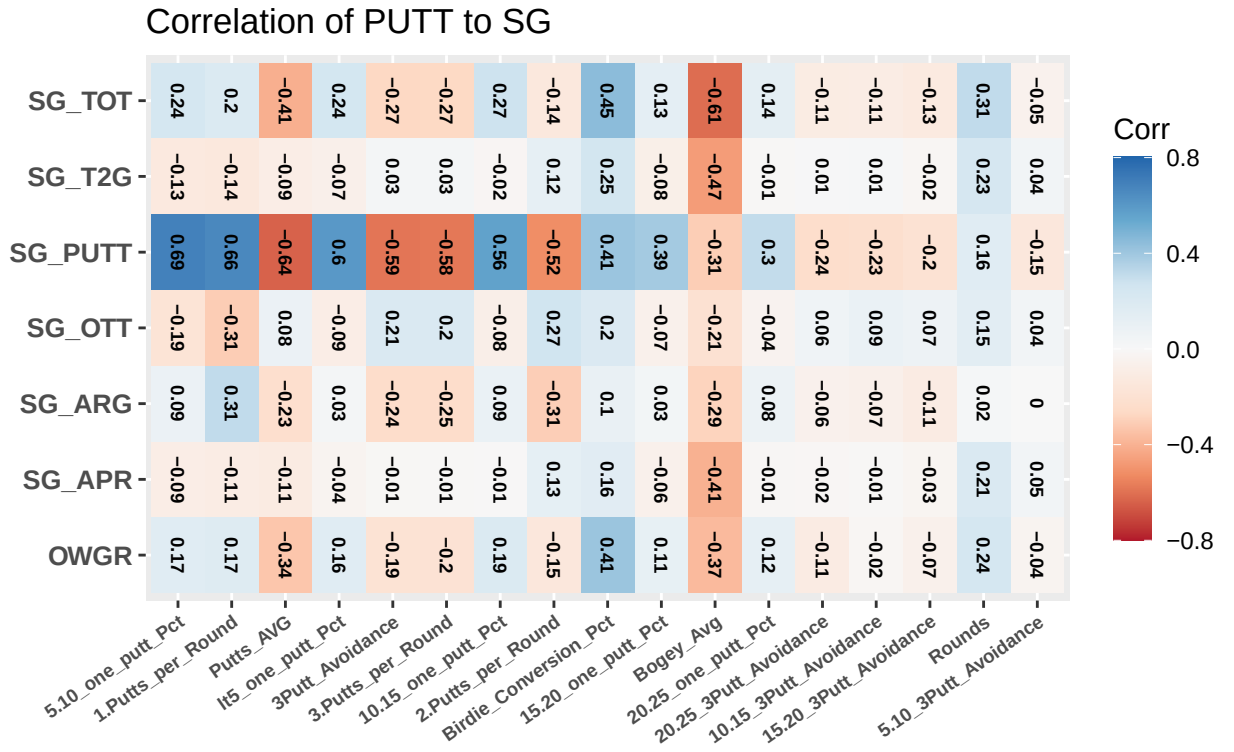

Figure 8: Spearman correlation plot of PUTT statistics with SG and OWGC. Correlations are sorted in descending order by the absolute value of the correlation to SG_{PUTT}.

3.2 Feature Selection for Model Training

A function was created to select features (i.e., PGA statistics) based on the correlation plots, shown previously, and their p-values. The function selects features by first filtering to those with p-values less than 0.05 (if p-values are supplied). Two cut-off values shrink the list of available features further based on user input. The first cut-off value is a lower correlation limit between a statistic and the specified strokes-gained category. The second cut-off is an upper correlation limit among the features, themselves, in order to eliminate redundant features that will only add noise to the model. In this report, the lower cut-off is between 0.1-0.15 and the upper cut-off is 0.85.

```
######## CHOOSE FEATURES BASED ON CORRELATIONS AND/OR P-VALUES #########

choose_feats <- function(corr,cor_lim=0.15,feat_lim=0.85,SG,p_vals) {
  ## corr: correlation matrix
  ## cor_lim: cut off correlation beneath which features will be eliminated
  ## feat_lim: Maximum correlation among features
  ## SG: The strokes gained metric (as a string) that you want to compare
        ##feature correlations to
  ## p_vals: tibble of p_values (two-sided) where two columns are named "Feature"
  # and "p_val" containing the corresponding data
  ## WARNING: p_value cutoff is set to 0.05.

  # If no p-values are supplied, then use cut-offs only
  if (missing(p_vals)) {
    temp<-as.data.frame(corr) %>% filter(abs(.[SG])>cor_lim) %>%
      select(SG) %>% mutate(Feature=rownames(.)) %>%
      filter(str_detect(Feature,"^(SG|SC_Score|G_[BP]|OWGR).*|(_RTP)$")==FALSE)
  }
  #If p-values are supplied, then use them
  #to identify statistically significant features in addition to the cutoffs.
  else {

    p_vals <- p_vals %>% filter(p_val<0.05)

    temp<-as.data.frame(corr) %>% filter(abs(.[SG])>cor_lim) %>%
      select(SG) %>% mutate(Feature=rownames(.)) %>% semi_join(p_vals,by="Feature") %>%
      filter(str_detect(Feature,"^(SG|SC_Score|G_[BP]|OWGR).*|(_RTP)$")==FALSE)
  }

  Features<-temp$Feature

  #create correlation matrix among features, themselves
  corr_n<- pga %>% select(Features) %>% cor(method="pearson")

  ## Eliminate features that have greater than 0.85 correlation among themselves ##
  nonFeatures <- as.data.frame(corr_n) %>%
    gather(key="Feature",value="Correlation") %>%
    filter(Correlation<1 & abs(Correlation)>feat_lim) %>%
    left_join(temp, by="Feature") %>%
    mutate(SG=abs(.[SG])) %>% group_by(Correlation)

  feats<-Features

  # Failsafe in case all features are relevant in the data
  if (length(nonFeatures$Feature)>0) {
```

```
    nonFeatures<-nonFeatures %>% filter(SG==min(SG)) %>% pull(Feature)
    feats <- Features[!Features%in%nonFeatures]
  }
  feats
}

########## APPLY FUNCTION ############

feats<-c(choose_feats(corr_OTT,0.1,0.85,"SG_OTT",p_vals_OTT),
         choose_feats(corr_APR_Dist,0.1,0.85,"SG_APR",p_vals_APR),
         choose_feats(corr_ARG,0.15,0.85,"SG_ARG",p_vals_ARG),
         choose_feats(corr_PUTT,0.15,0.85,"SG_PUTT",p_vals_PUTT)) %>% unique()

## Eliminate features with little variance
temp<-nearZeroVar(pga,uniqueCut = 20,names=TRUE)
feats <- feats[!feats %in% temp]
numfeats<-length(feats)
```

Feature selection is based on strokes-gained instead of the OWGR (the result statistic) because the correlation of individual statistics is stronger with SG than OWGR, especially for less sensitive game-aspects like putting. The correlation plot in Figure 9 shows that the total strokes-gained, SG_{TOT}, relates to OWGR, directly. For this reason, feature selection based on the constituents of SG_{TOT} in order to ultimately predict OWGR is appropriate.

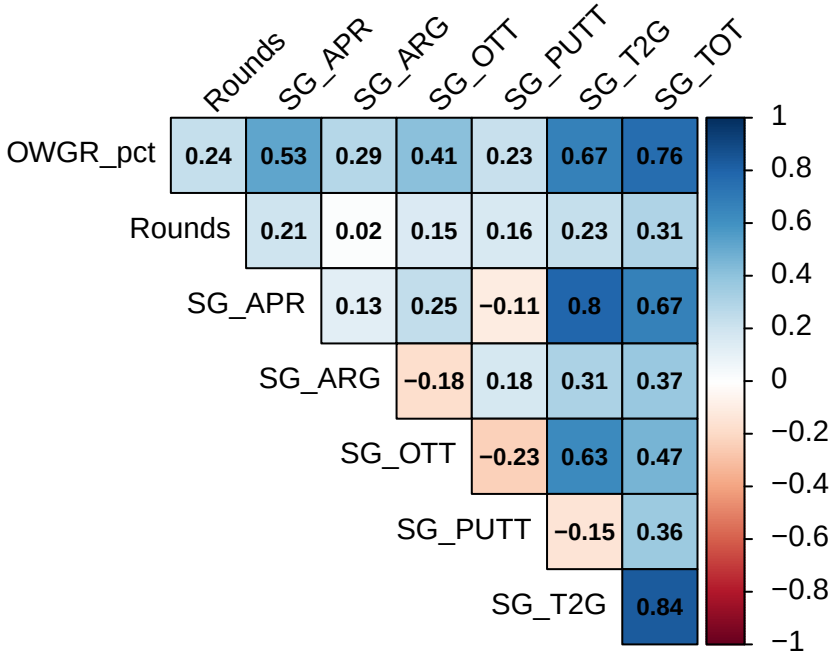

Figure 9: Upper correlation matrix of strokes gained and the OWGR. Note the strong correlation between SG_{TOT} and OWGR.

Other features eliminated from the model include some scoring statistics, such as the average par 4 score or birdie percentage by a player. These statistics are tied to SG and OWGR, directly, but the focus of this study is on specific game aspects (e.g., scrambling or tee-shot distances) that describe top ranked players. Narrowing the list of features in this manner leaves 43 features for training the models–quite the reduction from the original 116.

4 Machine Learning Models and Results

This report compares two classification algorithms to predict OWGR from player statistics: random forests and K nearest neighbors (KNN). KNN is run twice using the features defined earlier and then the features defined from principal component analysis (PCA). The main accuracy metric of the final model is the *sensitivity* per decile of players because this study focuses on the proportion of top and bottom players correctly identified as such. For comparison, *specificity* describes how well the algorithm classifies players as *not* belonging to a certain rank. The arithmetic average of *sensitivity* and *specificity* is the *balanced accuracy*. All of these metrics are referenced in the results tables, later on.

4.1 Random Forests

The random forests algorithm makes predictions by aggregating the results of many decision trees together either by taking the average (for regression) or a popular vote (for classification). These aggregations mitigate some of the weaknesses of decision trees on their own, such as the risk of over-training and instability to changes in training data. An added benefit of random forests is the output of *variable importance*, which ranks features according to how often they split nodes in trees. Random forests involve several tuning parameters that control the structure of each individual tree. The parameters optimized in this report are *nodesize* (the minimum number of data points in each terminal node) and *mtry* (the number of player statistics considered as candidate splitting variables) while the number of trees was set to 1500 to allow for convergence. The parameter *mtry* was tuned with 5-fold cross-validation, and *nodesize* was optimized by looping over trial values to maximize the sensitivity and balanced accuracy of the top decile of players. Increasing the *nodesize* yields a smoother result. Moreover, optimization of *nodesize* requires the test data set to be partitioned into an additional validation data set because the test data is input to the optimization procedure. This means that the final results are computed on the validation data set. Splitting the test set changes the original partitioning to 50% training data, 40% testing data, and 10% validation data.

Figure 10 shows how the sensitivity and balanced accuracy of the top players changes with *nodesize*. Some variation exists throughout the range of node sizes, but the sensitivity generally increases with *nodesize* while the balanced accuracy decreases. I assign *nodesize*=111 where both sensitivity and balanced accuracy are high in Figure 10. *Nodesize* has little effect on the results in comparison to *mtry* (Probst et al., 2019), so uncertainty in an "optimal" *nodesize* is assumed forgiven with random forests.

```
## Set up 5-fold cross validation ##
control<-trainControl(method="cv",number=5,p=0.2)

## With Feature Selection and more parameter tuning ##

pga_feats <- pga %>% select(feats,OWGR_pct)

set.seed(seed,sample.kind="Rounding")

## Partition test data set into a validation set ##
test_index <- createDataPartition(y = test$OWGR_pct, times = 1, p = 0.8, list = FALSE)
validation <- test[-test_index,]
test_2 <- test[test_index,]

set.seed(seed,sample.kind="Rounding")

## Using cross validation, optimize the model for mtry ##
temp_rf <- train(OWGR_pct~.,method="rf",
                 trControl=control,
                 tuneGrid=data.frame(mtry=seq(1,21,2)),
                 nTree=1500,
                 data=pga_feats)
```

```r
## Now optimize the model for node sizes ##
nodesizes <- seq(1, 161, 10)

stats <- sapply(nodesizes, function(ns){
  set.seed(seed,sample.kind="Rounding")
  train_rf<-train(OWGR_pct~., method = "rf", data = pga_feats,
        tuneGrid = temp_rf$bestTune,nTree=1500,
        nodesize = ns)

  #make predictions from model
  y_hat_rf <- predict(train_rf,test_2[,feats])

  #get summary statistics of the results
  cm_feats <- confusionMatrix(y_hat_rf,test_2$OWGR_pct)

  #print summary statistics
  cm_feats$byClass["Class: 1",c("Sensitivity","Balanced Accuracy")]
})
```

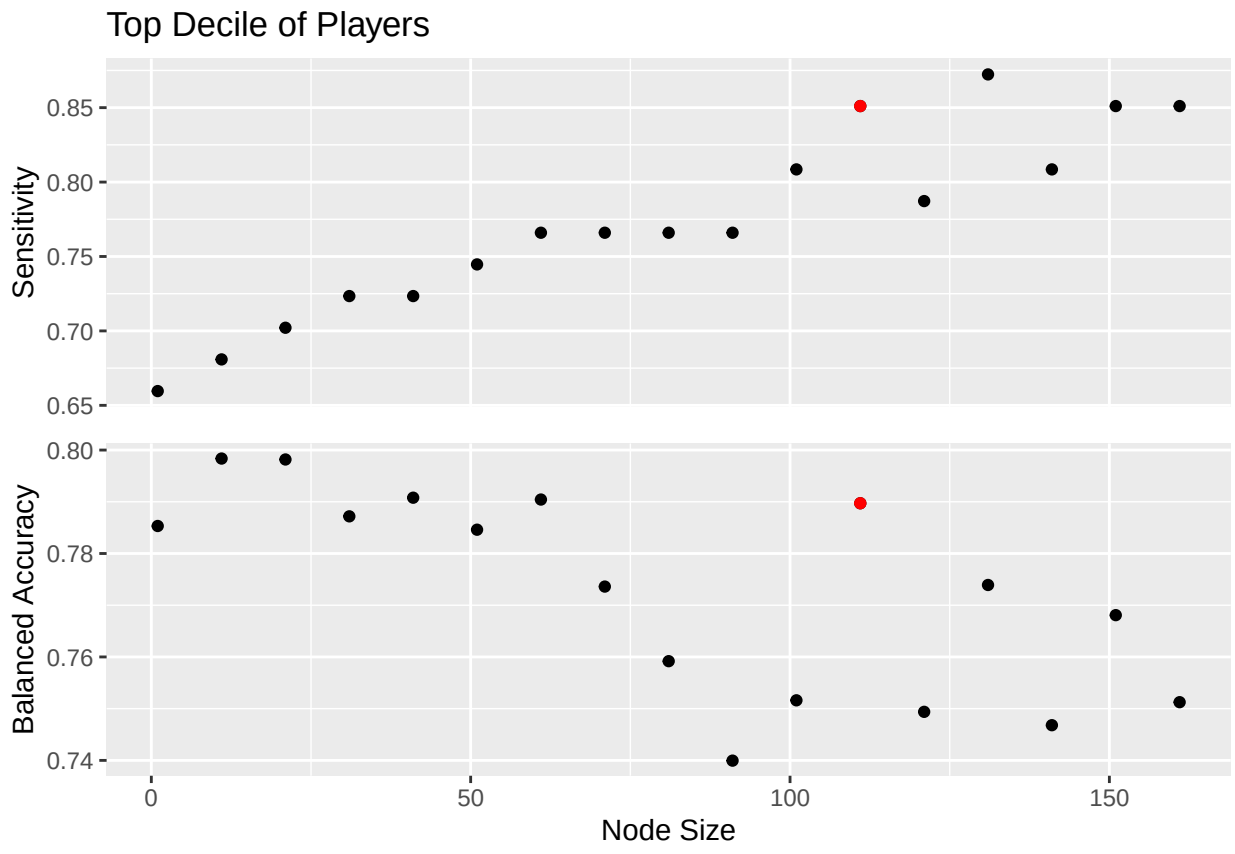

Figure 10: Sensitivity and balanced accuracy of the top decile (10%) of Tour players versus the node size used in random forests.

Using cross validation, the optimized value for *mtry* is 5. The tuned parameters are input to the final model, and summary statistics for each player decile, or "Class," are printed in Table 2. The *F1* score is a weighted combination of *sensitivity* and *specificity*.

Table 2: Sensitivity, specificity, F1-Score, and Balanced Accuracy metrics for each predicted decile of Tour players. The top and bottom deciles have reasonably high sensitivities while middle rankings appear to be guesses.

Class	Sensitivity	Specificity	F1	Balanced Accuracy
Class: 1	0.8182	0.7677	0.5566	0.7929
Class: 2	0.0000	0.9798	0.0000	0.4899
Class: 3	0.0000	1.0000	NA	0.5000
Class: 4	0.2727	0.9394	0.2855	0.6061
Class: 5	0.0000	1.0000	NA	0.5000
Class: 6	0.0000	1.0000	NA	0.5000
Class: 7	0.4545	0.8485	0.3783	0.6515
Class: 8	0.0833	0.9592	0.0973	0.5213
Class: 9	0.0000	1.0000	NA	0.5000
Class: 10	1.0000	0.6869	0.5904	0.8434

Recall that the total strokes-gained, SG_{TOT}, demonstrates notable separation between the top and bottom deciles of players while the middle classes exhibit much overlap (Figure 2). This pattern is evident in Table 2 because the sensitivity is maximized for the top and bottom deciles while the middle is almost nil. The Balanced Accuracy is quite good for top and bottom classifications (Class: 1 & Class: 10), but the results appear as a 50/50 guess for the remaining classes. To get a better picture of the overall performance, the middle classes are grouped together so that the classification model has just three classes (Table 3). Table 3 also shows the sensitivities for a random forest model without feature selection (labeled "Sensitivity w/o Feat-Ex") as comparison to the tuned model.

```
## Recoding factors so that the middle classes are aggregated together
levels(y_hat_rf) <- c("10",rep("2-9",8),"1")
val_agg<-validation$OWGR_pct
levels(val_agg)<-c("10",rep("2-9",8),"1")
cm_feats <- confusionMatrix(y_hat_rf,val_agg)

Scores_rf<-get_scores(cm_feats,1.75)

Scores<-left_join(Scores_rf,Scores_rf_nofeats,by="Class") %>%
  select(Class,Sensitivity.y,Sensitivity.x,Specificity.x,F1.x,"Balanced Accuracy.x") %>%
  set_names("Class","Sensitivity w/o Feat-Ex",
            "Sensitivity","Specificity","F1","Balanced Accuracy")

top_bottom<-round(mean(Scores$`Balanced Accuracy`[c(1,3)])*100,1)
overall<-round(mean(Scores$`Balanced Accuracy`)*100,1)
```

Table 3: Sensitivity, specificity, F1-Score, and Balanced Accuracy metrics for each predicted decile of Tour players. The top and bottom deciles have reasonably high sensitivities while middle rankings appear to be slightly better than guesses.

Class	Sensitivity w/o Feat-Ex	Sensitivity	Specificity	F1	Balanced Accuracy
Class: 1	0.6207	0.8182	0.7677	0.5566	0.7929
Class: 2-9	0.7052	0.3977	0.9545	0.4654	0.6761
Class: 10	0.5263	1.0000	0.6869	0.5904	0.8434

The overall balanced accuracy for identifying the three classes of players is 77.1% while the accuracy of

identifying just the top and bottom classes is 81.8%. The sensitivities also show that the model with feature selection based on correlations improved predictions over a model without feature selection (i.e., w/o Feat-Ex) (Table 3).

4.2 K-Nearest Neighbors (KNN)

K-Nearest Neighbors is a household name in machine learning. As the name suggests, the algorithm classifies input data by taking the average of the K-nearest points, or neighbors, in the n-dimensional feature space. Higher values of K yield smoother results while values that are too small risk over-training the model. As such, tuning the K parameter is a core focus of the KNN algorithm. Similar to the random forests implementation, the K is tuned by maximizing the sensitivity of the top OWGR decile over a series of trial K values. This study runs KNN twice: once using principal component analysis (PCA) and the second using the same selection of features defined for random forests. Comparison of the two runs is summarized in a table, later.

Figure 11 displays the cumulative proportion of the variance explained by successive principal components of the data. The principal components that explain 95% of the variance are used to train the KNN model. Beyond 95%, the principal components contain negligible information (i.e., variance) about the data. This 95% threshold corresponds to 39 components, although 50% of the variance can be explained by the first 10, only.

```
####### Principal Component Analysis #######

pga_sub<-subset(pga_nofeats,select=-OWGR_pct)
pca <- prcomp(pga_sub)

#create dataframe to contain the cumulative variance
##explained by adding principal components
var_explained <- cumsum(pca$sdev^2/sum(pca$sdev^2)) %>% as_tibble() %>%
  rownames_to_column() %>% set_names("PCA","Variance_Explained") %>%
  mutate(PCA=as.numeric(PCA))

#specify a variance threshold that defines which
##principal components we eliminate in the training model.
threshold <- 0.95
PCA_thresh<- var_explained %>% filter(Variance_Explained<threshold) %>%
  summarize(m=max(PCA)) %>% pull(m)

#plot of the variance explained by successive principal components
var_explained %>% ggplot(aes(PCA,Variance_Explained)) + geom_point() +
  geom_text_repel(data=subset(var_explained,PCA==PCA_thresh),
                  aes(PCA,Variance_Explained,
                      label=sprintf("PCA %d @%.2f%%",PCA,Variance_Explained*100),size=3),
                  box.padding = 4,direction="y",color="red",show.legend = FALSE) +
  ggtitle("Cumalitive Variance Explained by PCs") +
  xlab("PC")
```

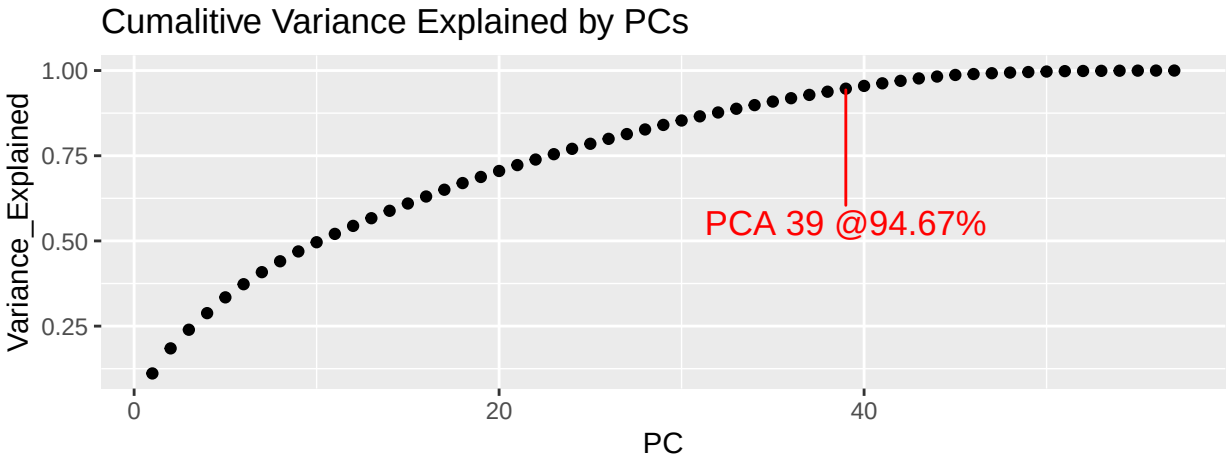

Figure 11: Cumulative variance explained by adding principal components to the data. The threshold for including principal components into the KNN model is labeled.

Much like the optimization of *nodesize* in Figure 10, the optimal K value is highlighted in Figure 12. The K value changes slightly if it's taken from the balanced accuracy, but the balanced accuracy is still near a maximum if the K value is taken from the sensitivity plot in Figure 12. A K value of 225 is input to the final KNN model and evaluated over the same validation data set as the random forests model.

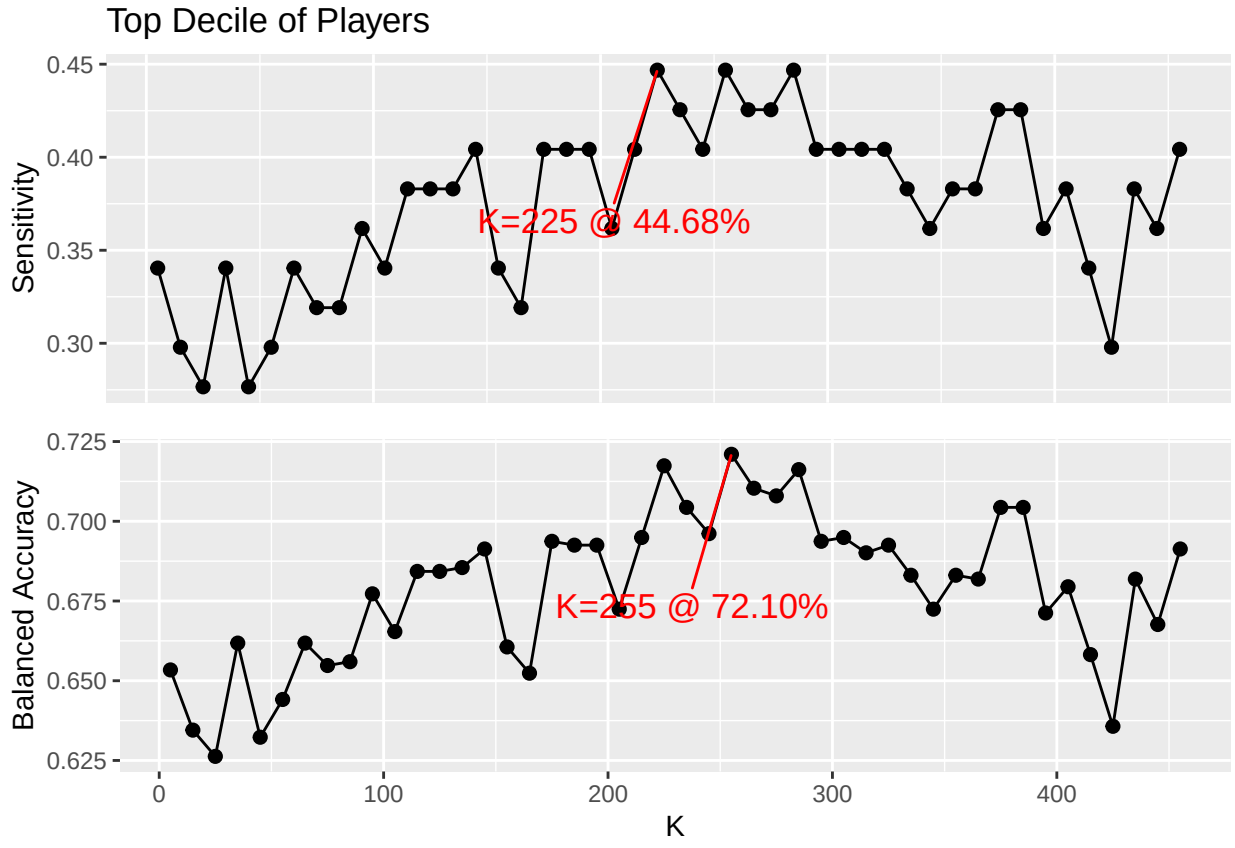

Figure 12: The value of K versus the output sensitivity on the test set. The optimal value of K is labeled.

Table 4 is a summary table of the KNN results using the optimal K-value. Immediately apparent, the

sensitivities are substantially lower than those for random forests and render KNN useless for ranking golfers here.

Table 4: Summary statistics for the output of KNN using principal component analysis.

Class	Sensitivity	Specificity	F1	Balanced Accuracy
Class: 1	0.1818	0.9798	0.2156	0.5808
Class: 2	0.3636	0.9596	0.3898	0.6616
Class: 3	0.0000	0.9697	0.0000	0.4848
Class: 4	0.0909	0.8788	0.0870	0.4848
Class: 5	0.0000	0.9293	0.0000	0.4646
Class: 6	0.1818	0.6970	0.1237	0.4394
Class: 7	0.2727	0.8081	0.2189	0.5404
Class: 8	0.1667	0.8469	0.1512	0.5068
Class: 9	0.0000	1.0000	NA	0.5000
Class: 10	0.0909	0.9697	0.1078	0.5303

The top decile, "Class: 1", does not have the greatest accuracy even though the model is optimized to do so. The bottom two deciles, among others, are complete guesses because both demonstrate zero or near-zero sensitivity. Smaller K-values can mitigate this problem, but classification of the top decile will worsen. The large K value infers that a significant overlap exists among the player deciles and Tour statistics. This overlap is extreme enough that nearly 25% (i.e., `K/length(data)`) of the data needs to be averaged to classify a single point. Much like Table 3, Table 5 aggregates the summary statistics into bottom, middle, and top deciles while adding a separate sensitivity column for a KNN model that uses the same features as random forests, earlier.

Table 5: Aggregrated summary statistics for the output of KNN using pincipal component analysis versus feature selection based on correlations (i.e., the same as the random forests case, earlier).

Class	Sensitivity w/o PCA	Sensitivity	Specificity	F1	Balanced Accuracy
Class: 1	0.0909	0.1818	0.9798	0.2156	0.5808
Class: 2-9	0.9545	0.9432	0.1364	0.9076	0.5398
Class: 10	0.0000	0.0909	0.9697	0.1078	0.5303

Feature extraction based on correlations to SG (Section 3.2) yields better results than extractions based on PCA (Table 5). The sensitivity of the top decile is 0.091 for the model based in feature correlations and 0.182 for the PCA-based model. Despite the improvement when using feature correlations, the KNN model gives unreliable results that are risky to bet on. The overall balanced accuracy for identifying the three classes of players is 55% while the accuracy of identifying just the top and bottom classes is 55.6%.

5 Discussion

The summary tables clearly show that the random forests models outperform KNN models by a wide margin when predicting the top decile of Tour pros. To review, the sensitivity of the top OWGR decile is 81.8% from the random forests model and 18.2% from the best KNN model. The balanced accuracy of the top OWGR decile is 79.3% from the random forests model and 58.1% from the best KNN model. Even though the KNN results improved after changing the feature extraction methodology from PCA- to correlation-based, the sensitivities are too low for the model to be useful in identifying top players. Middle-ranked players demonstrate excellent sensitivity due to their high prevalence, but the balanced accuracy is still less than random forests. A clustering approach such as KNN is not recommended for PGA statistics because of

their wide variations among players. A structured, decision based model such as random forests is a better alternative.

The top 30 Tour statistics utilized by random forests is shown in Table 6 alongside the percentage importance of each.

```
varimp<-varimp_feats$importance %>%
  mutate(Overall=round(Overall,2)) %>%
  arrange(desc(Overall)) %>% head(30)
```

Table 6: The most important Tour statistics, ordered by descending importance, for predicting the OWGR decile of each player.

	Overall
Rounds	100.00
OT_Driving_Dist	98.17
APR_175.200_Fwy_Dist	53.88
APR_150.175_Fwy_Dist	53.42
APR_200.225_Fwy_Dist	53.29
ARG_Scramble_Dist	44.48
OT_Apex_Ht	39.88
G_3Putt_Avoidance	32.36
ARG_SandSaves_Pct	31.24
ARG_10.20_Scramble_Pct	29.80
G_10.15_one_putt_Pct	27.99
G_lt5_one_putt_Pct	25.07
APR_225.250_Fwy_Dist	18.03
ARG_Other_Locations_Scramble_Pct	17.68
ARG_Rough_Scramble_Pct	16.82
ARG_gt10_Scramble_Pct	15.96
APR_250.275_Fwy_Dist	15.56
G_15.20_one_putt_Pct	15.49
ARG_gt30_Scramble_Pct	15.41
APR_250.275_Rgh_Dist	14.60
APR_125.150_Fwy_Dist	14.56
G_5.10_one_putt_Pct	14.44
APR_gt275_Fwy_Dist	13.77
APR_gt200_Rgh_Dist	13.05
APR_100.125_Fwy_Dist	12.04
G_1.Putts_per_Round	11.85
ARG_20.30_Scramble_Pct	10.94
APR_225.250_Rgh_Dist	10.13
APR_175.200_Rgh_Dist	10.01
APR_50.125_Fwy_Dist	9.40

As anticipated from Figures 2-3, Tour statistics from off-the-tee (OT) and on the approach (APR) are deemed most important when splitting decision trees. More specifically, Tour rankings are best predicted with the average driving distance, number of rounds played, and a player's ability to hit mid-iron shots from distances between 150-225 yards on the fairway. Also important is the player's around-green-performance by scoring par-or-better after missing the green-in-regulation, better known as scrambling. The top decile of players has an excellent short-game away from the fringe (e.g., sand) and distances specifically between 10-20 yards

from the hole. Putting performances influence the OWGR according to a player's ability to sink 1-putts less than 5ft or 10-15ft from the hole while distances greater than 15ft are not as important. Many of the less relevant features for predicting OWGR revolve around approach (APR) shots taken from the rough, no matter the distance-to-pin. This observation leads to an interesting idea that the best players do not necessarily distinguish themselves by their performances in non-ideal situations (i.e., non-fairway). Along the same theme, a player's ability to avoid three-putts from long distances does not separate him from the pack; sinking putts close to the pin matters the most, especially after a good approach.

6 Conclusion & Future Improvements

This study classifies the top 10% of PGA Tour players based on Tour statistics containing percentages, counts, and distances of shots for each player and different game aspects (i.e., green vs. tee-box). Scoring statistics are removed from the model because this study focuses on the physical game aspects that separate winners from losers. Random forests and k-nearest neighbors (KNN) are both optimized and tested for this task. The random forests classifier outperformed KNN for both the top and bottom deciles of players with an average balanced accuracy of 79.3%. KNN nearly gives a complete guess for most of the player rankings. The variable importance output by random forests broadly demonstrates that players distinguish themselves with their tee-shots and mid-iron approaches on the fairway. Performances around the green gain importance at locations other than the fringe and distances less than 5 yards or 10-20 yards from the hole. Putting gains importance when the 1-putts are made within 15ft of the hole and 3-putts are avoided. Altogether, a player's official world golf ranking (OWGR) depends more on his performance in ideal situations such as mid-fairway lies and makeable putts than tough situations such as rough lies and long putts (e.g., >20ft).

Because KNN struggles to classify top-ranked players, other classification algorithms can be explored. The outputs of multiple classification algorithms can be combined to give an ensemble estimate of the OWGR for each player. An ensemble model can be more robust than classification models run on their own, so the results might improve for all the OWGR deciles–not just the top and bottom. Random forests can be further optimized by testing the model for different data partitions and sample sizes, although the parameter *mtry* is the dominant variable. Given the large number of predictors in the PGA data set, the model might improve if the predictors are in units of relative-to-par (RTP) instead of physical distances and percentages. RTP units are closely tied to the final scores, so RTP predictors would improve the success statistics considerably. Despite the improvement, interpretation of RTP statistics is non-intuitive, so the observations discussed in Section 5 would lose credibility. For this reason, future editions of the model can include additional player statistics rooted in physical units rather than scoring units.

7 References

Byrne, B. M. (2010). Structural equation modeling with AMOS: Basic concepts, applications, and programming. New York: Routledge.

Hair, J., Black, W. C., Babin, B. J. & Anderson, R. E. (2010) Multivariate data analysis (7th ed.). Upper Saddle River, New Jersey: Pearson Educational International.

Probst, P., Wright, M., & Boulesteix, A. L. (2019) Hyperparameters and Tuning Strategies for Random Forest. WIREs Data Mining and Knowledge Discovery.