Capstone Project 1 – ATP Tennis Match Statistics

Introduction

Sports statistics have been around for a long time. In recent years, it seems like advances in technology has increased the number of statistics that have been published in virtually every major professional sport. The problem then, isn't the availability of sports statistics, the problem is understanding what those statistics mean and whether sports participants can improve aspects of their performance, thereby increasing their chances of winning.

If you've ever watched a professional tennis match on television, you've no doubt seen various match statistics displayed on the screen and discussed by the commentators—before, during and after a match. They may comment on the differences between key statistics for each player and try to tie those statistics to why the players are winning or losing. Most often, however, there is no clear "winner" in each category, much less any consistency from one match to the next on which category(ies) matters most. In fact, it is not uncommon for a player to perform well in one category in one match and win, and then perform as well or better in that same category in the next match and lose.

After a match, players and their coaches review match statistics to determine what their next practice sessions need to focus on. Unfortunately, absence of any meaningful analysis on those match statistics, practice time could result in spending time trying to improve a particular skill that won't necessarily translate into more match wins. It is easy to say, "Your opponent had fewer unforced errors on her backhand side, so let's work on your backhand." It may be true that the opponent had fewer unforced errors, but it's difficult to say whether reducing the number of unforced backhand errors will result in a win the next time with any degree of certainty. Or a coach will say, "You missed all of your drop shots; let's work on that." Yet the player only missed three drop shots in a match with over 500 hundred total shots, so even if the player made all three drop shots, it's unlikely the match outcome would be any different.

The Problem

So the challenge is to develop a machine learning model of match statistics that will result in meaningful information that will help a player her team spend time practicing what will be most likely to improve match outcomes.

Wikipedia describes **sports analytics** as a "collection of relevant, historical, statistics that when properly applied can provide a competitive advantage to a team or individual." That's what this project is all about.

Goals

1. Identify which features (match statistics) are the most meaningful.
2. Explore relationship between match statistics and for match winners and losers.
3. Create machine learning model to predict match winners.

The Data

The data for the project came from the ATP World Tour Website as distributed on datahub.io. The data contains ATP tournaments, match scores, match stats, rankings and players overview. The data was scraped from the ATP's website using python scripts written by Kevin Lin.

The dataset has 53, unindexed CSV files broken into 5 different categories: tournaments, match scores, match stats, player rankings and player overviews. Tournament and match score files go back to 1877, while the rankings data goes back to 1973. The match stats, however, only go back to 1991. The focus of this project, therefore, will be to analyze all statistics from 1991 to 2016 since the match stats are the main focus of this analysis.

There were 95,359 match scores in 2,054 tournaments included in the original data matching the criteria above.

Data Wrangling

Summary Files. The results of the data scraping produced five separate files. For analysis purposes and faster processing, these files were merged into one comprehensive dataframe.

Performance Features for Exploratory Data Analysis. Due to the unique nature of tennis scoring, the variety of match lengths (some matches end relatively quickly with few statistics in all categories, while some can last five full sets), and the large number of statistics, a the statistics were converted to relative performance metric. This addition will allow for an easier EDA phase later. These performance metrics are as follows:

- Rank difference – the difference between opposing player rankings
- Percentage of aces
- Percentage of double faults
- Percentage of first serves in
- Percentage of first serve points won
- Percentage of second serve points won
- Percentage of break points won
- Percentage of return points won, and
- Percentage of total points won.

Duplicates, Unnecessary, Incomplete and/or Missing Values. In merging the different datasets, duplicate and/or irrelevant columns were removed. Also, several matches included in the overall dataset were from matches that either were not played or were due to retirement. These matches were removed.

Data Limitations. In today's tennis world, there is a much more robust set of match statistics that are being collected. It would be interesting to have access to this data for this project. Shot related data, such as cross court , down-the-line, drop shot, overheads, etc. broken down into winners, forced errors and unforced errors, length of rallies, serve and return placement, etc., would no doubt provide some additional insights not captured in this project.

Tournament and Match Dates

Unfortunately, the match statistics for each match did not include the specific data of the match. As such, all matches were assigned the start date of the tournament in which they were played. The dates were then broken into year, month and day for data exploration purposes, as well as providing additional options for training and testing the machine learning model.

Two Final Datasets

The data was ultimately put into two main datasets for Exploratory Data Analysis and Machine Learning: 1) A match statistics dataset that allows for analysis and comparison between match statistics in general, and 2) a "results" dataset that separates winner players' statistics from loser players' statistics and then combines them into one dataset to allow for analysis on match statistics differences between match winners and losers.

Exploratory Data Analysis (EDA)

Exploring the data will be split into two stages. The first will include an analysis of statistics within matches. The second will be an analysis of winners' and losers' statistics.

Match Statistics

After cleaning and wrangling the data, there are match statistics from 88,206 matches. Here is a summary of the DataFrame columns in terms of number and type of values (all float64), along with confirmation that there are no null values.

```
statsdf[main_eda_stats].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 88206 entries, 0 to 88205
Data columns (total 20 columns):
match_duration              88206 non-null float64
rank_dif                    88206 non-null float64
winner_ace_pct              88206 non-null float64
loser_ace_pct               88206 non-null float64
winner_df_pct               88206 non-null float64
loser_df_pct                88206 non-null float64
winner_firstsrv_in_pct      88206 non-null float64
loser_firstsrv_in_pct       88206 non-null float64
winner_firstsrv_won_pct     88206 non-null float64
loser_firstsrv_won_pct      88206 non-null float64
winner_secondsrv_won_pct    88203 non-null float64
loser_secondsrv_won_pct     88202 non-null float64
winner_srv_pts_pct          88206 non-null float64
loser_srv_pts_pct           88206 non-null float64
winner_rtn_pts_pct          88206 non-null float64
loser_rtn_pts_pct           88206 non-null float64
winner_brk_pts_pct          88206 non-null float64
loser_brk_pts_pct           88206 non-null float64
winner_points_won_pct       88206 non-null float64
loser_points_won_pct        88206 non-null float64
dtypes: float64(20)
memory usage: 13.5 MB
```

The table below shows the results of the describe() function with the summary statistics for each column:

```
statsdf[main_eda_stats].describe().T
```

	count	mean	std	min	25%	50%	75%	max
match_duration	88206.0	104.614777	33.616448	60.000000	80.000000	99.000000	121.000000	353.000000
rank_dif	88206.0	29.128869	87.778525	-298.000000	-18.000000	22.000000	70.000000	299.000000
winner_ace_pct	88206.0	8.340684	6.336654	0.000000	3.676471	6.976744	11.538462	56.756757
loser_ace_pct	88206.0	5.599402	4.661899	0.000000	2.127660	4.587156	7.865169	40.000000
winner_df_pct	88206.0	3.471327	2.647796	0.000000	1.612903	3.076923	4.938272	26.027397
loser_df_pct	88206.0	4.511130	3.174744	0.000000	2.150538	4.000000	6.250000	35.416667
winner_firstsrv_in_pct	88206.0	58.351252	14.343813	0.000000	54.054054	60.112715	66.019417	100.000000
loser_firstsrv_in_pct	88206.0	58.864649	8.704818	21.276596	53.061224	58.762887	64.583333	100.000000
winner_firstsrv_won_pct	88206.0	76.232252	8.290399	7.500000	70.588235	76.190476	81.818182	100.000000
loser_firstsrv_won_pct	88206.0	65.354815	9.674287	5.555556	59.259259	65.714286	72.000000	100.000000
winner_secondsrv_won_pct	88203.0	56.121267	10.466212	0.000000	49.253731	55.555556	62.500000	100.000000
loser_secondsrv_won_pct	88202.0	44.723647	10.149542	0.000000	38.235294	45.000000	51.515152	94.117647
winner_srv_pts_pct	88206.0	68.222500	6.719134	19.696970	63.529412	67.857143	72.580645	100.000000
loser_srv_pts_pct	88206.0	56.694481	7.437214	14.285714	52.054795	57.031250	61.682243	85.185185
winner_rtn_pts_pct	88206.0	43.319388	7.449335	14.814815	38.317757	42.982456	47.959184	85.714286
loser_rtn_pts_pct	88206.0	31.802591	6.894812	0.000000	27.419355	32.142857	36.486486	79.166667
winner_brk_pts_pct	88206.0	49.534994	19.620488	0.000000	36.363636	50.000000	60.000000	100.000000
loser_brk_pts_pct	88206.0	31.585153	28.057964	0.000000	0.000000	28.571429	50.000000	100.000000
winner_points_won_pct	88206.0	55.467523	4.115671	30.303030	52.525253	54.819277	57.731959	82.758621
loser_points_won_pct	88206.0	44.532477	4.115671	17.241379	42.268041	45.180723	47.474747	69.696970

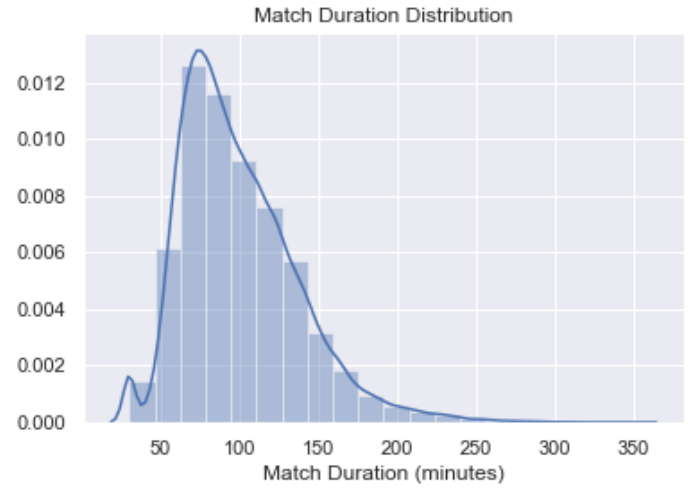
Points of interest include:

- The first six columns have mean values higher than the median value for the column.
- There is a notably large difference between the 25th%tile/75th%tile and min/max values \
- This observation suggests that there are extreme value outliers in our data set.

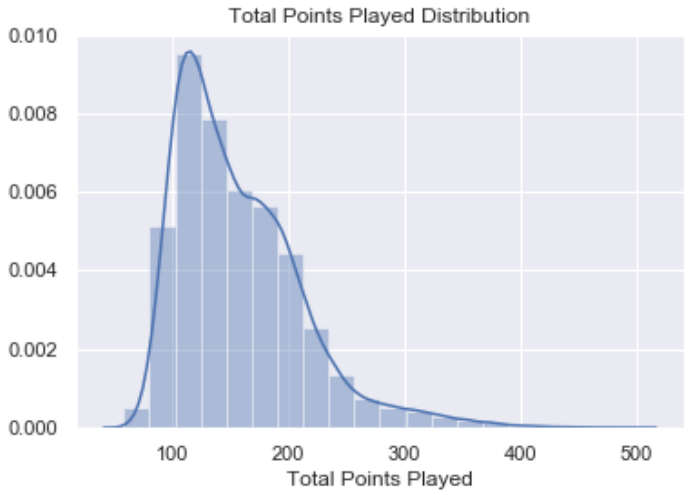
Here are some interesting findings about the match statistics:

- The average match duration is 104.6 minutes.
- On average, the matches had a total of 157.0 points played, or an average of 40 seconds per point.
- On average, matches had a total of 24.6 games played with each game lasting 4 minutes 15 seconds.
- On average, players played 2.5 sets per match, with each set lasting 41 minutes 11 seconds.

Here are plots of the distributions of match duration and total points played:

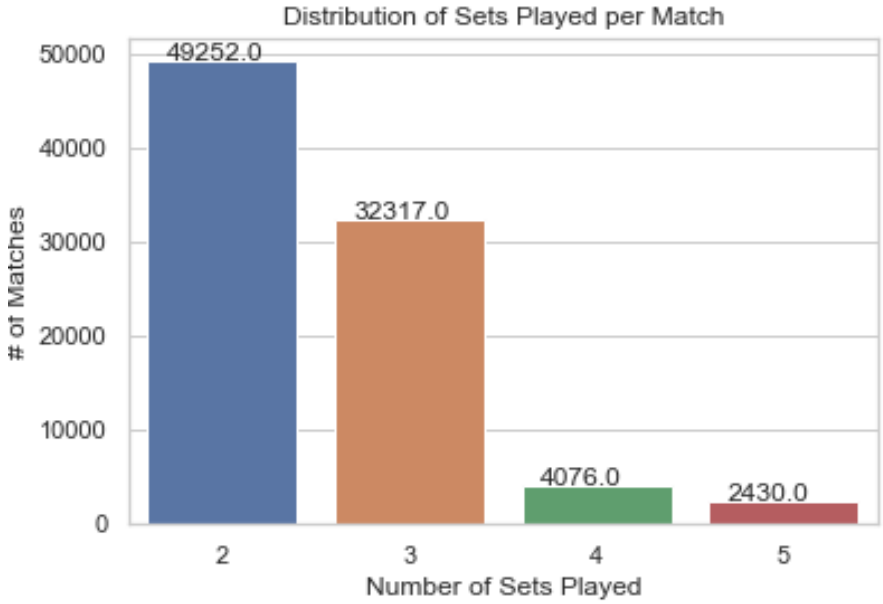


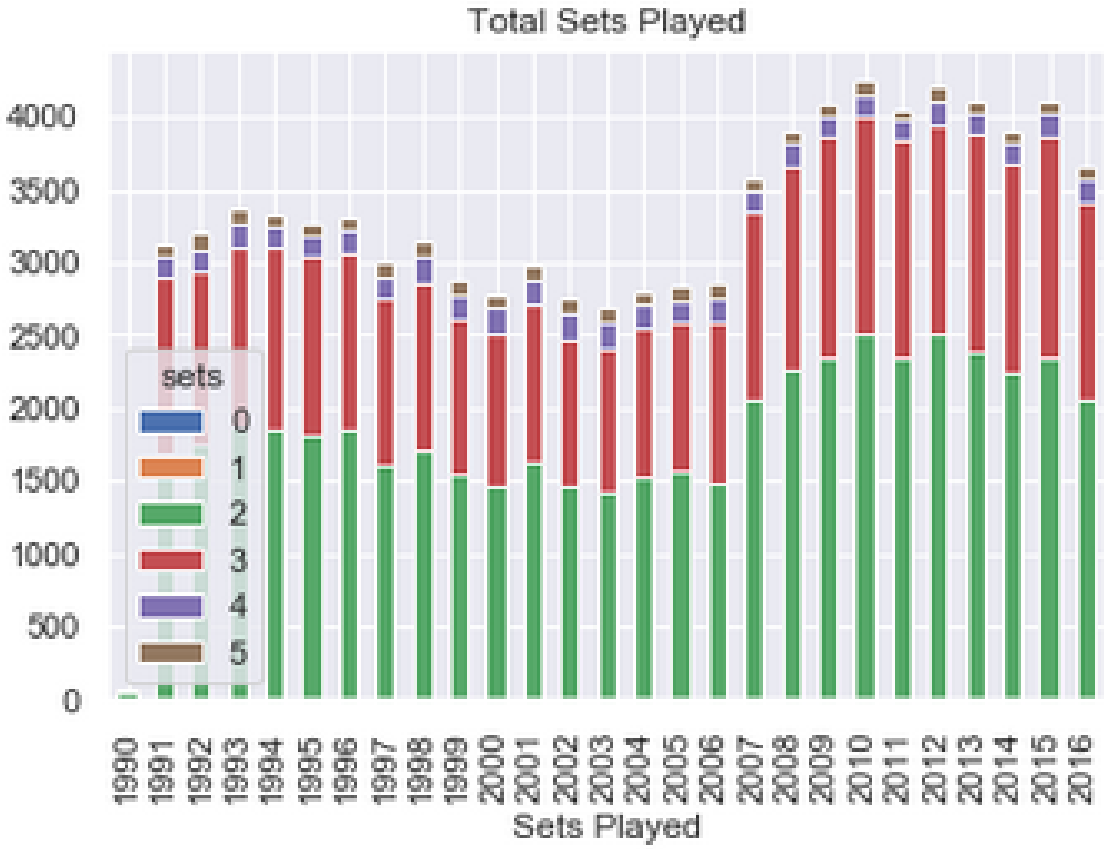
As noted earlier, the mean match duration is 104.6 minutes (1 hr 44 minutes 36 seconds), whereas the median is 99 minutes (1 hr 39 minutes). This is due to: 1) minimum sets per match is two and it takes a certain amount of time to play two sets, and 2) some matches can last a long time, since tennis matches (at least back then) did not have a time limit and had a best of 3 out of 5 sets format.



As expected, the distributions for match duration and total points played are very similar, and the reasons for this mirror the explanation provide earlier.

The average number of sets played per match is 2.54. Below are the distributions of sets played overall and sets player per match by year.


Distribution of Sets Played per Match
# of Matches
50000
40000
30000
20000
10000
0
49252.0
32317.0
4076.0
2430.0
2
3
4
5
Number of Sets Played


Total Sets Played
4000
3500
3000
2500
2000
1500
1000
500
0
sets
0
1
2
3
4
5
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
Sets Played

The plot shows that 55.8% of matches only played two sets. As shown in the second plot, the percentage of matches by sets seemed pretty consistent over the years. The plot for sets played helps to further explain the differences between mean and median values in some of the match statistics.

Performance Categories. When comparing performance category statistics, the following is worth noting:

Percent of Times the Match Winner Performed Better in By Category

Category	% Winner Performed Better
Percentage of Total Points Won	94.3
Percentage of Service Aces	63.9
Percentage of Double Faults	58.8
Percentage of First Serves In	54.1
Percentage of First Serves Won	82.9
Percentage of Second Serves Won	78.5
Percentage of Service Points Won	92.7
Percentage of Return Points Won	92.6
Percentage of Break Points Won	70.7
Winner Ranked Higher	65.5

What is so interesting about these statistics is that although the data shows that the mean of the winners' statistics for each category was better than the mean for the losers', the winner did not outperform the loser in every category in a match. It will be interesting to see which ones matter most.

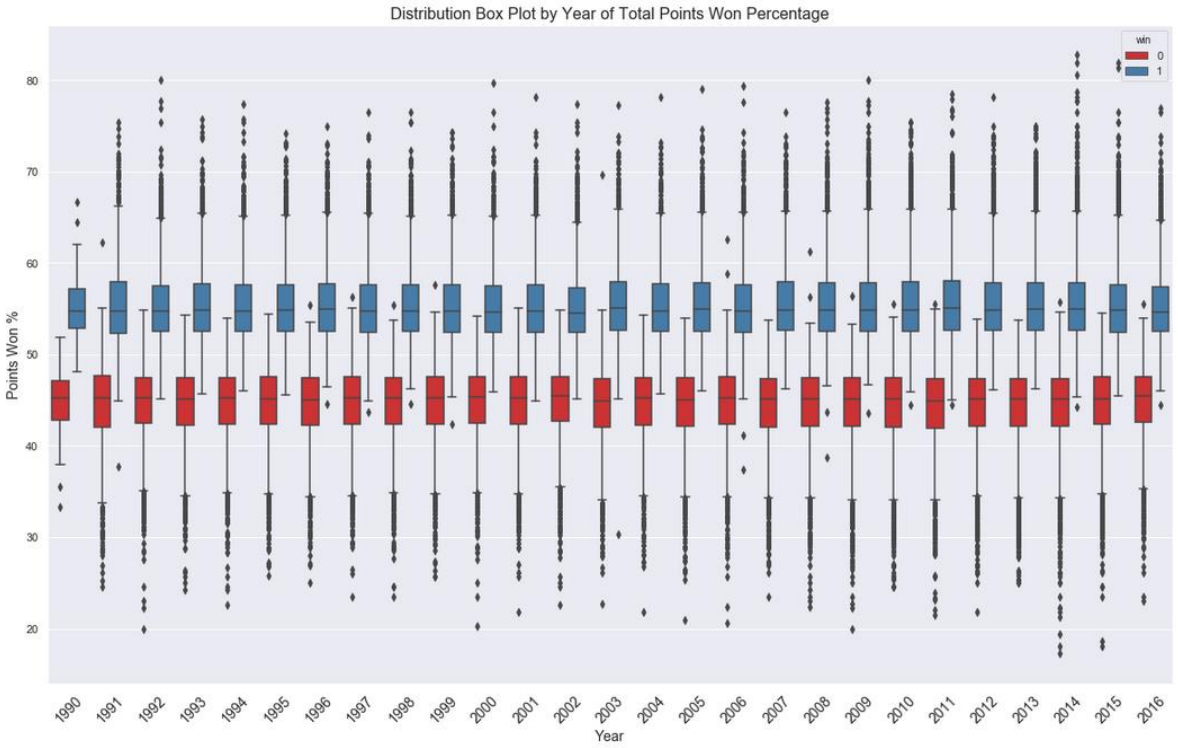
Also, the three categories in which the winner outperformed the loser are related—total points won can be broken down into winning points while serving (Percentage of Service Points Won) and winning points while returning (Percentage of Return Points Won). In fact, with one exception, Rank Difference, all of the performance categories are really just subsets of the Total Points Won category, merely indicating how a player won the point. As such, Total Points Won will not be used later for our machine learning model, as it would certainly cause overfitting.

It's worth noting that while the average margin of victory was 10.9 percent of Total Points Won, or 14.3 points per match for the winner, with an average of 24.6 games played per match, that's only 0.58 points per game. Since there is some overlap in these categories, we may need to drop some prior to beginning the machine learning.

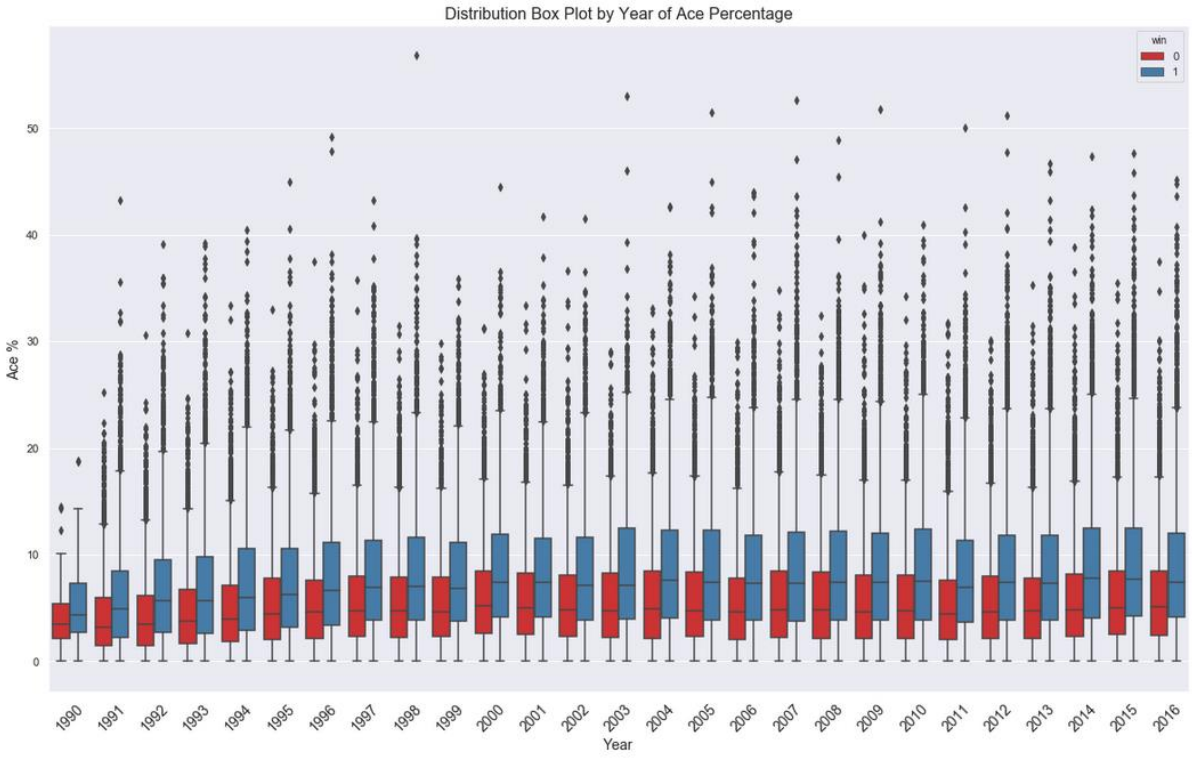
Winner and Loser Statistics

Here are the plots and the related statistics for each of the categories listed above, except for player rankings.

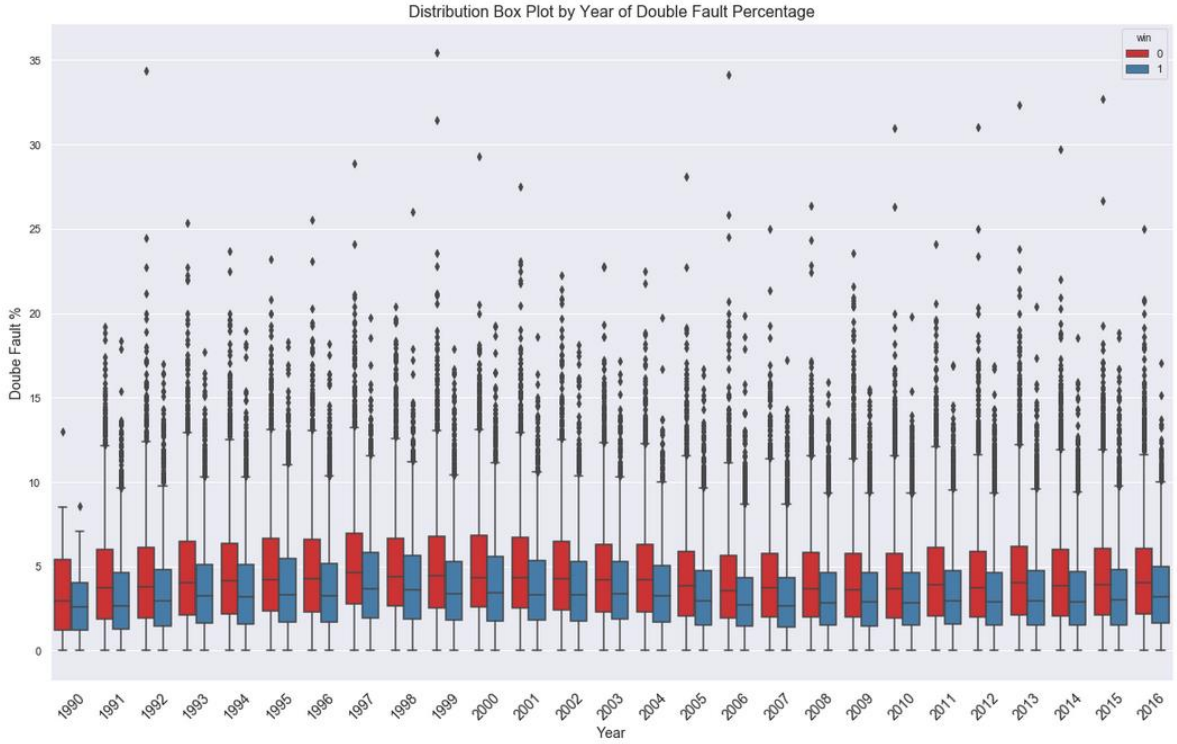
Percent Total Points Won



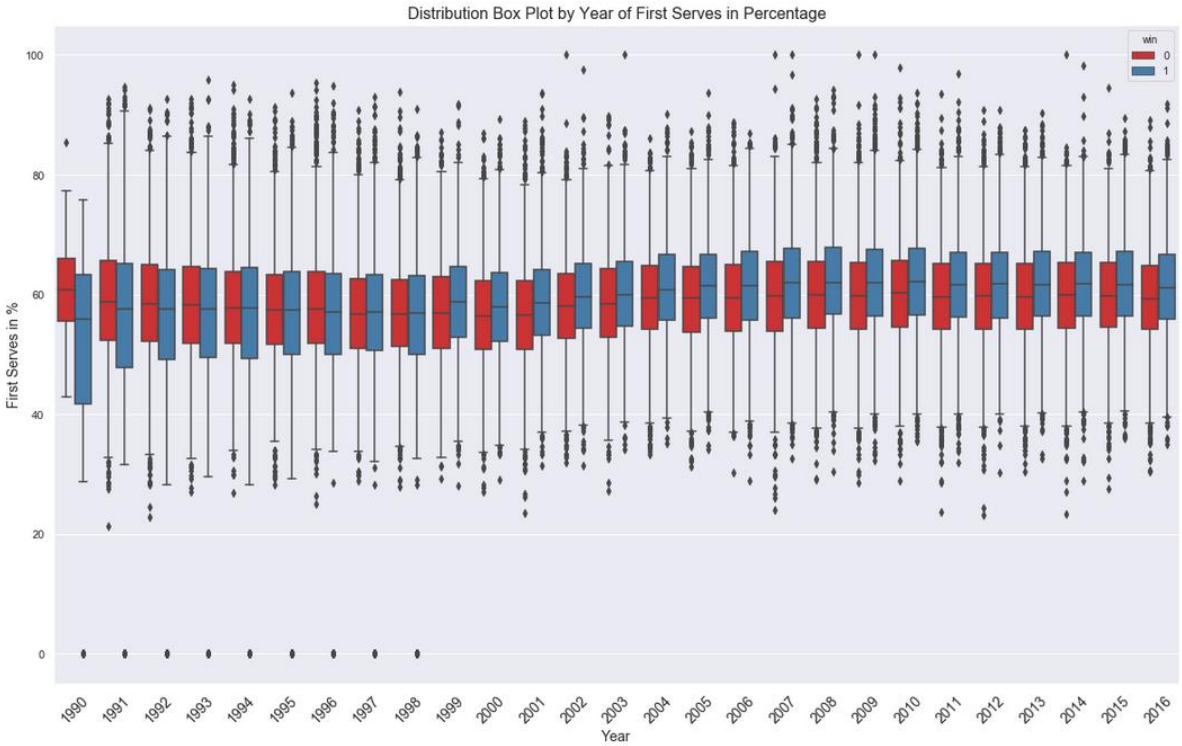
Percent Service Aces



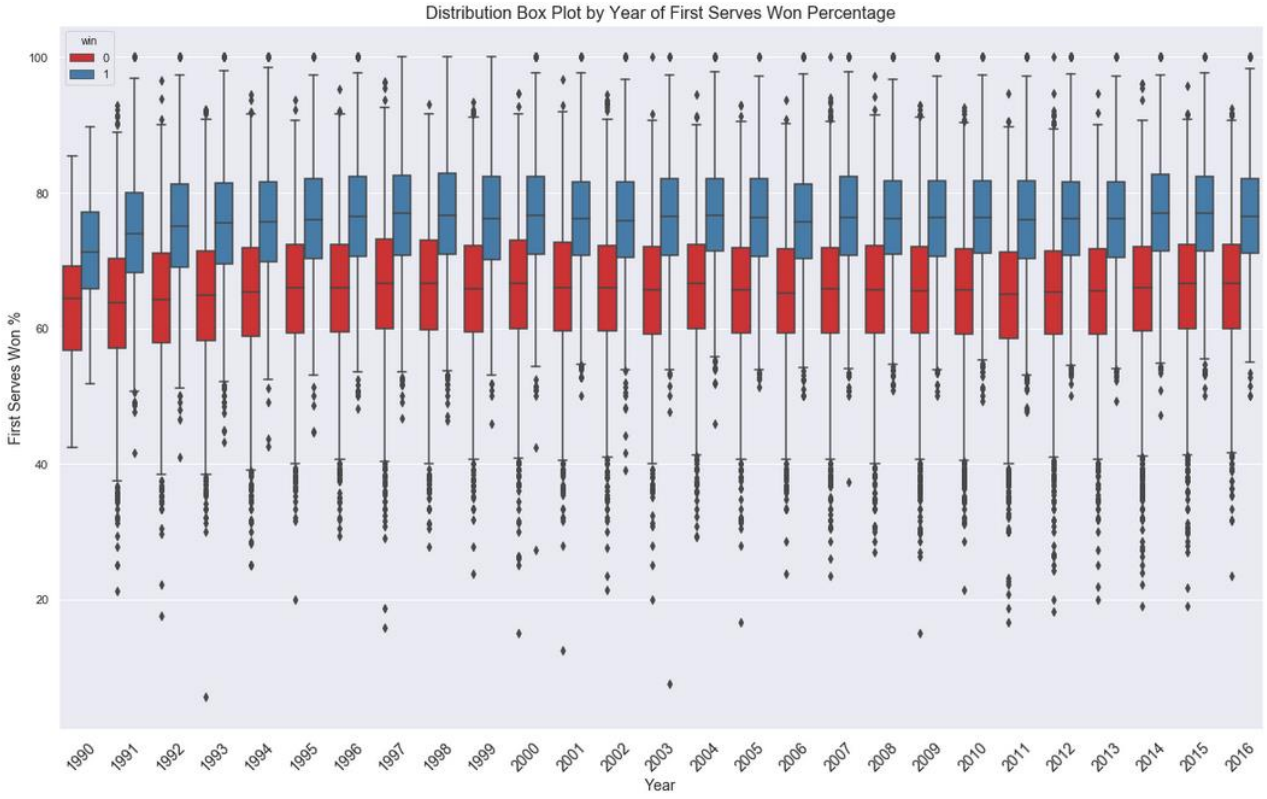
Percent Serve Double Faults



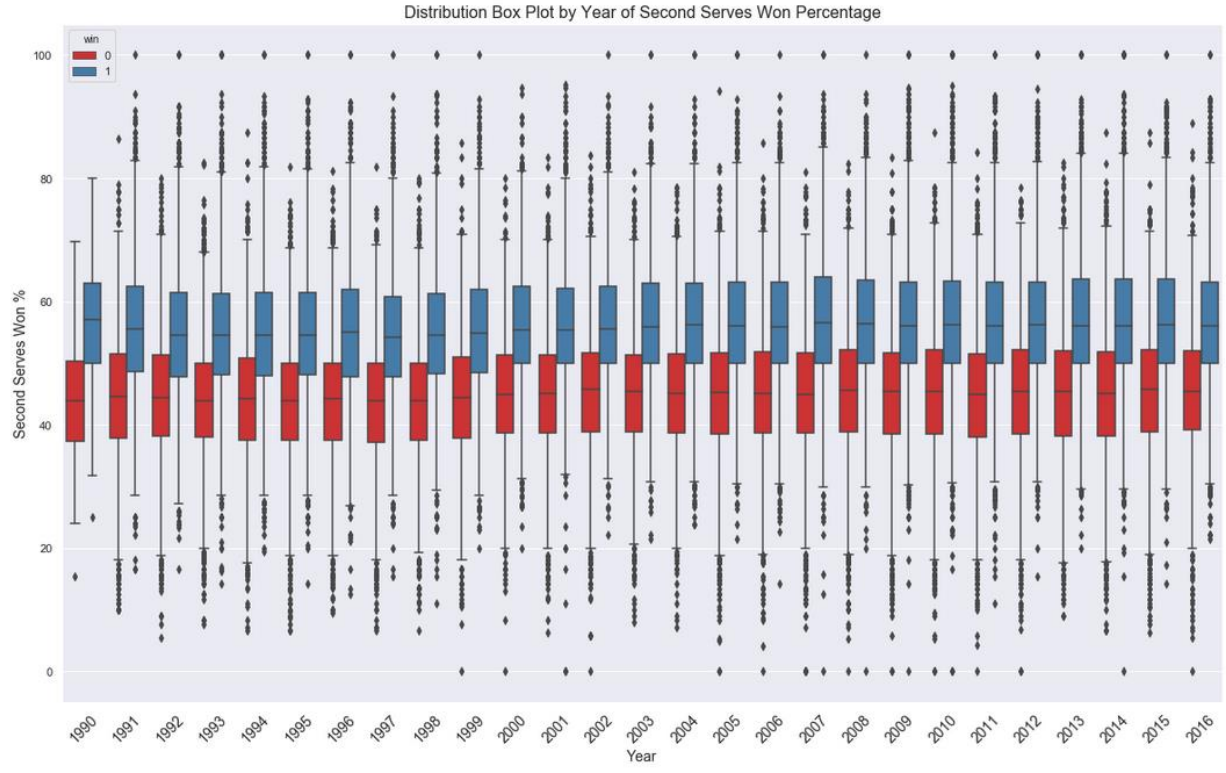
Percent First Serves In



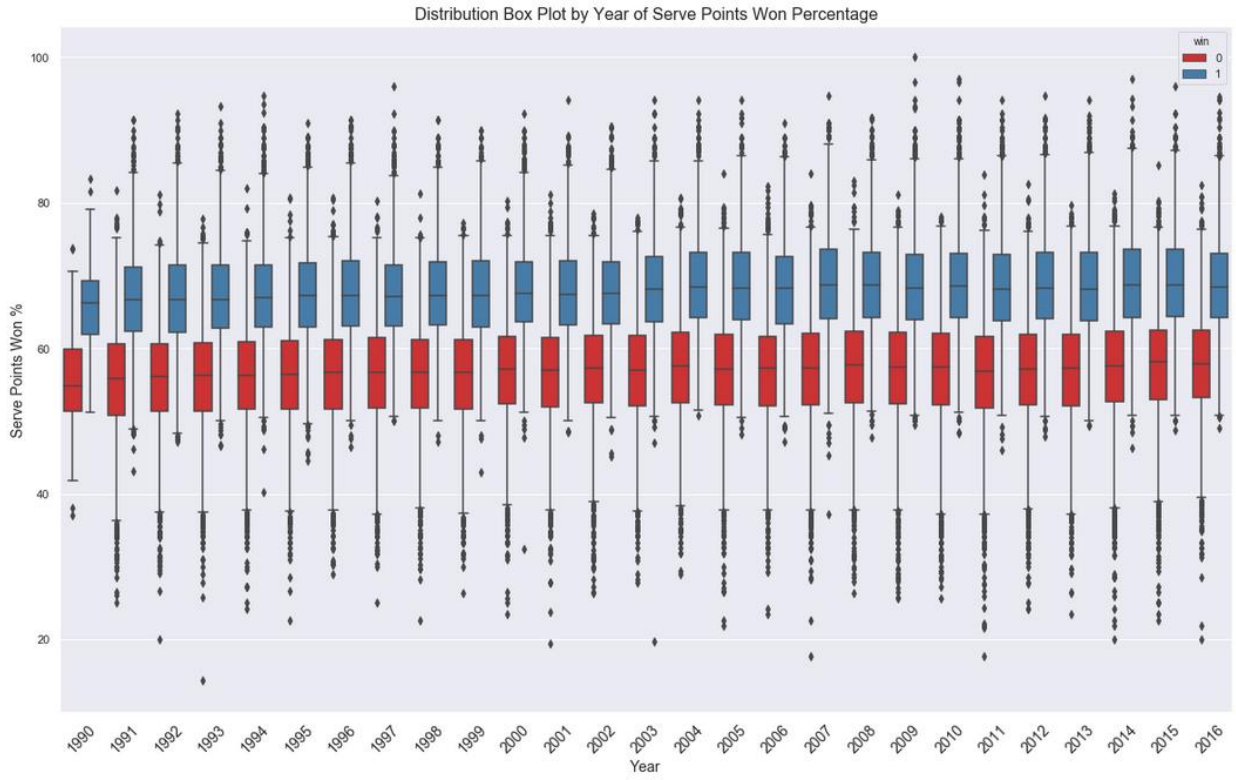
Percent First Serves Won



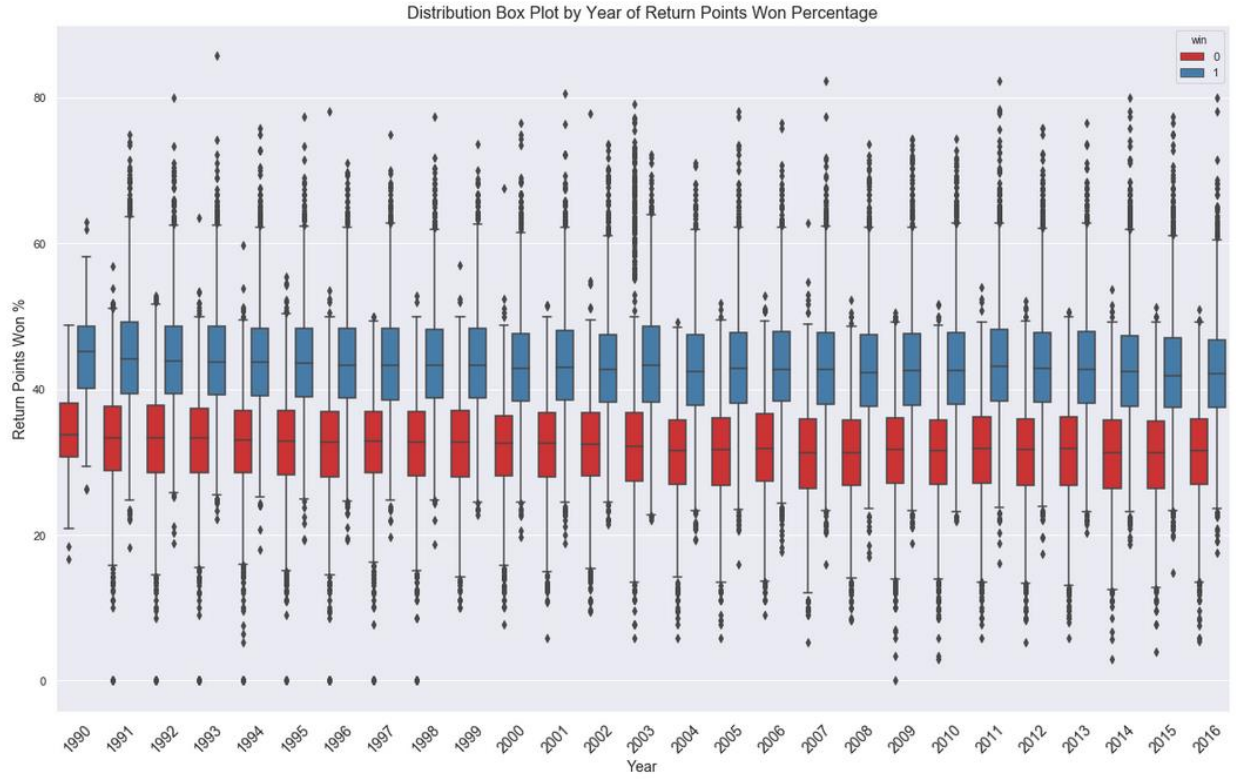
Percent Second Serves Won



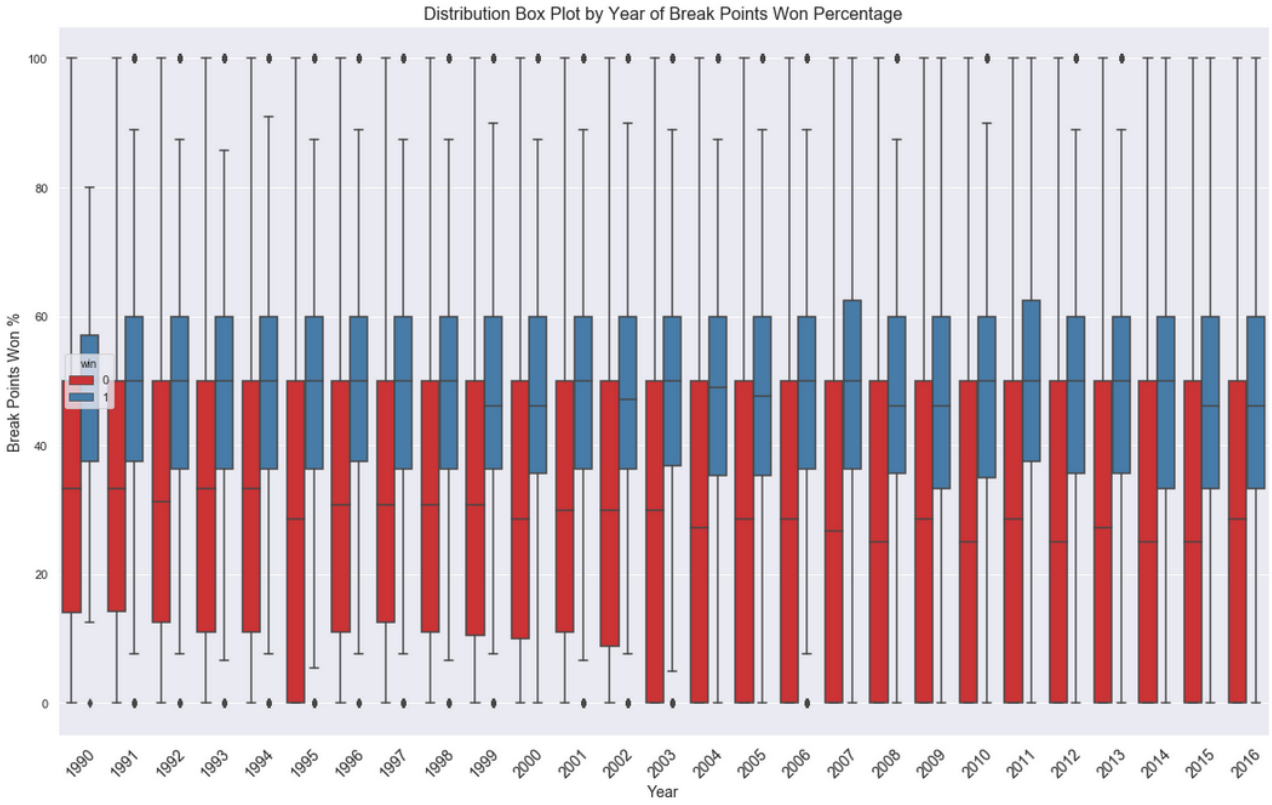
Percent of Serve Points Won



Percent of Return Points Won



Percent of Break Points Won



In examining all the distribution plots, all of them seem consistent except for the distribution plot for break points won percentage. This makes sense because break points are a unique type of point in that a break point is only "available" when the player serving is one point away from losing a game. In some matches there may be a lot of them; in others you may not have very many. Plus, you could also have several in a single game, because once the players are at break point in a game, they could alternate winning points, and have several break points in that game, with either zero break points converted because the server ends up winning the game, or one break point out of several when the returning player wins the break point.

Also, worth noting is that the mean for each category in every year was higher for the winner than the match loser with one exception—Percentage of First Serves In. From 1991 to 1998, the mean for percentage of first serves in for the match loser was slightly higher than or close to equal with the match winner. Although this makes sense in that just getting the ball may not be as important as serve speed and placement, it would still be interesting to see what caused the change after 1998.

The plots also confirm what was seen earlier, in that most of the data falls within the 25th and 75th percentiles, but there are definitely some outliers both above and below these quartiles.

Inferential Statistical Analysis

This section loos at statistical significance on observations made and thoughts about the findings during the EDA section. This is an essential step to understanding whether the differences between Match Winners and Losers are unlikely to have happened by chance alone.

The focus lies primarily on three categories: 1) Distribution between Match Winners and Match Losers, 2) Correlation with the Target Variable (i.e., Win or Non-Win), and 3) Collinearity between Features.

Challenges

Normality. A challenge for this project was the lack of normally distributed data (see Q-Q plots below). Normally distributed data are often a requirement for classical statistical tests. Fortunately, the Central Limit Theorem allows us to use the Z-test to compare sample distribution differences. (Central Limit Theorem states that the sampling distribution of the sample means approaches a normal distribution as the sample size gets larger — no matter what the shape of the population distribution. This is especially true for sample sizes over 30. In other words, as you take more samples, especially large ones, the graph of the sample means will look more like a normal distribution.)

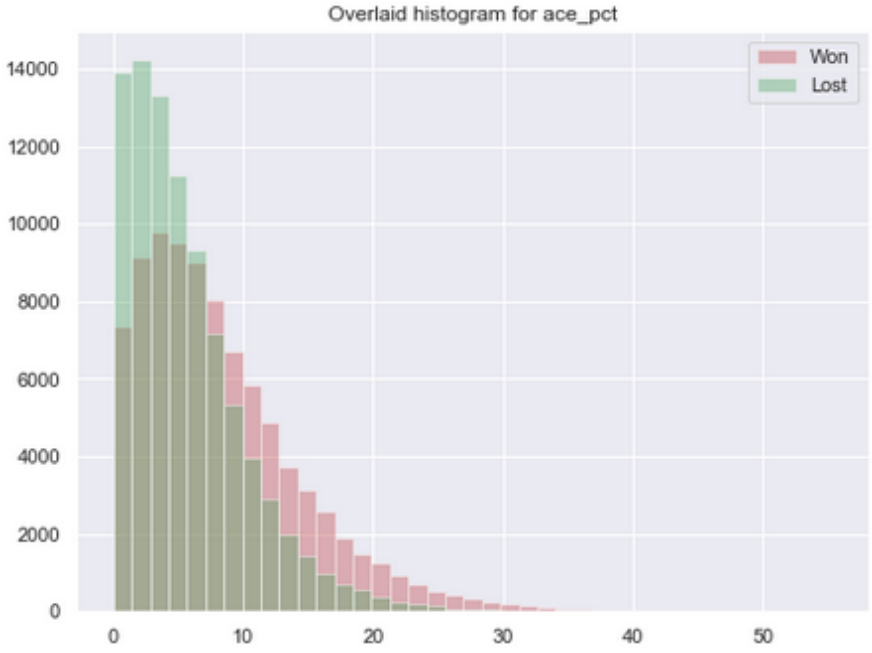
Statistical and Practical Significance. Due to the relatively large sample sizes, even small differences can be considered statistically significant, but may not allow us to use a particular feature since this may put a limitation on that feature's predicative qualities.

Continuous Features. In many ways, prediction match winners is about finding the subtle differences and similarities between Wins and Non-Wins. Continuous features tend to be much more valuable in uncovering those differences and trends across time. This machine learning model only uses continuous features.

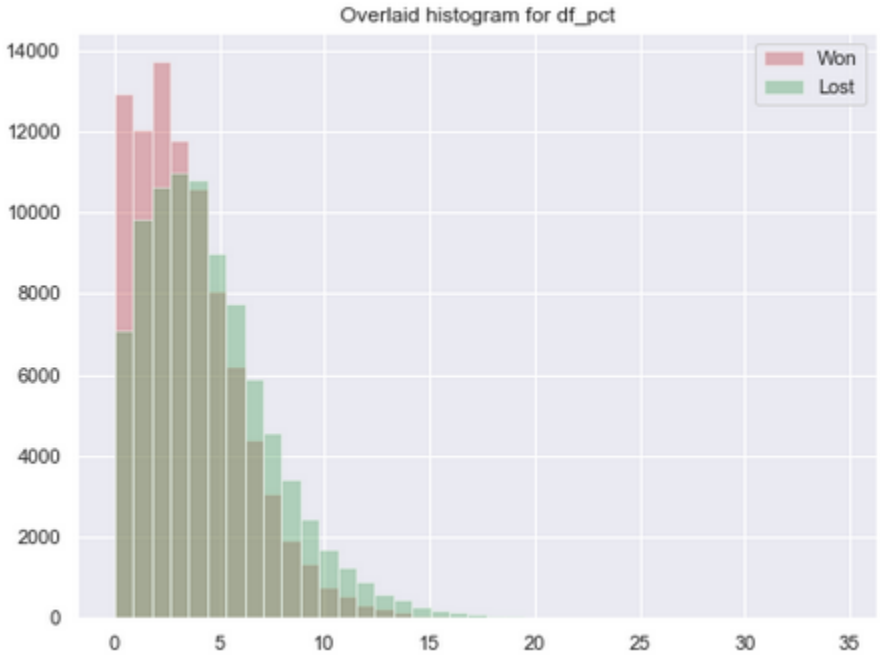
Distributions. To understand whether the differences between Wins and Non-Wins observed are significant, I conducted Z-tests for the distributions on the following: Ace Percent, Double Fault Perfect,

First Serves In Percent, First Serve Points Won Percent, Second Serve Points Won Percent, Service Points Won Percent, Return Points Won Percent, Break Points Won Percent and Total Points Won Percent.

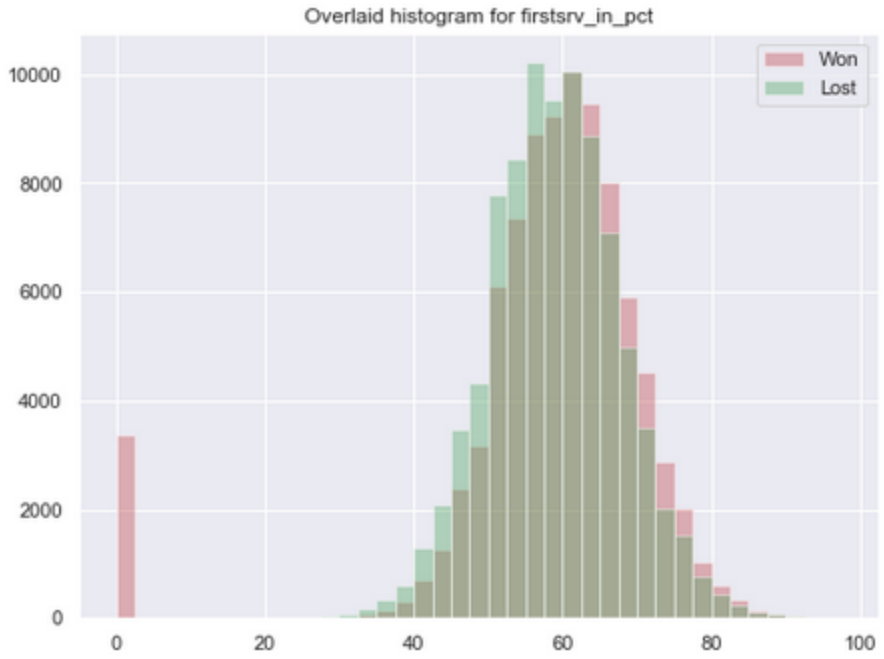
The method used was to compare mean differences across 10,000 permutations and then check whether the mean difference of the observed distributions would fall into the realm of significant possibilities. The results are shown below.




```
Wins Mean:
8.340683934072723
No-Wins Mean:
5.599402213071454
Mean Diff:
2.741281721001269
H0 Diff: 0

p: 0.0
CI: [-0.05431697
0.05277014]
ME:
0.053402597518318703
```

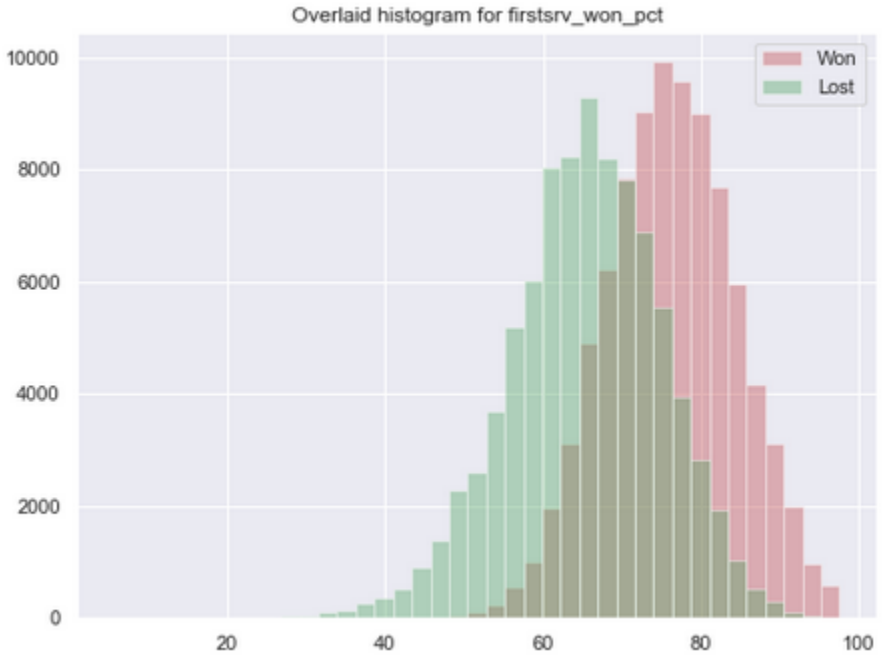



```
Wins Mean:
3.471326834959865
No-Wins Mean:
4.511129954525473
Mean Diff: -
1.0398031195656081
H0 Diff: 0

p: 0.0
CI: [-0.02795768
0.02747843]
ME: 0.0275627659420
```



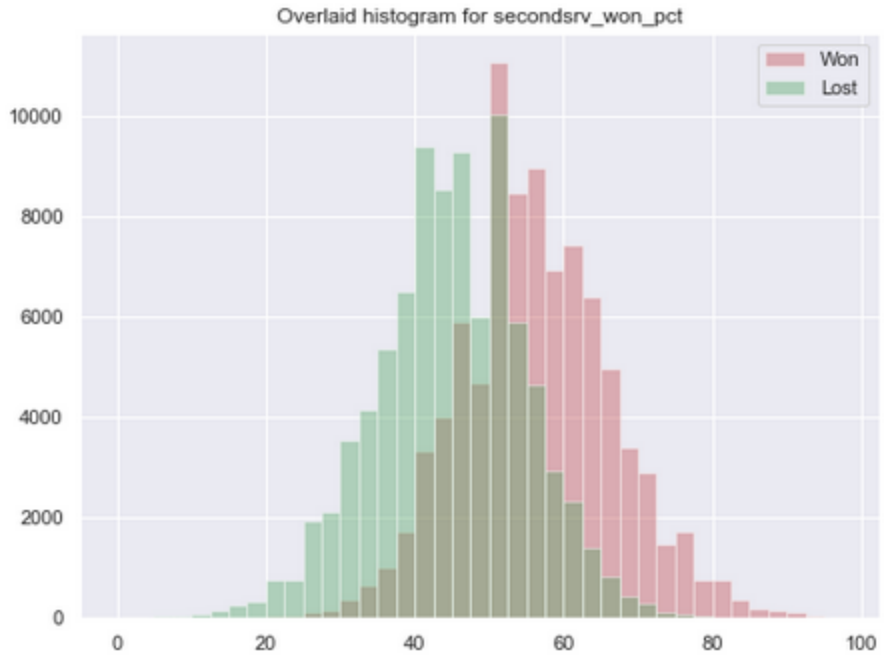
Wins Mean: 58.35125231156482
No-Wins Mean: 58.8646494896052
Mean Diff: -1.0398031195656081
H0 Diff: 0

p: 0.0
CI: [-0.11174051 0.10695299]
ME: 0.10793662133669706



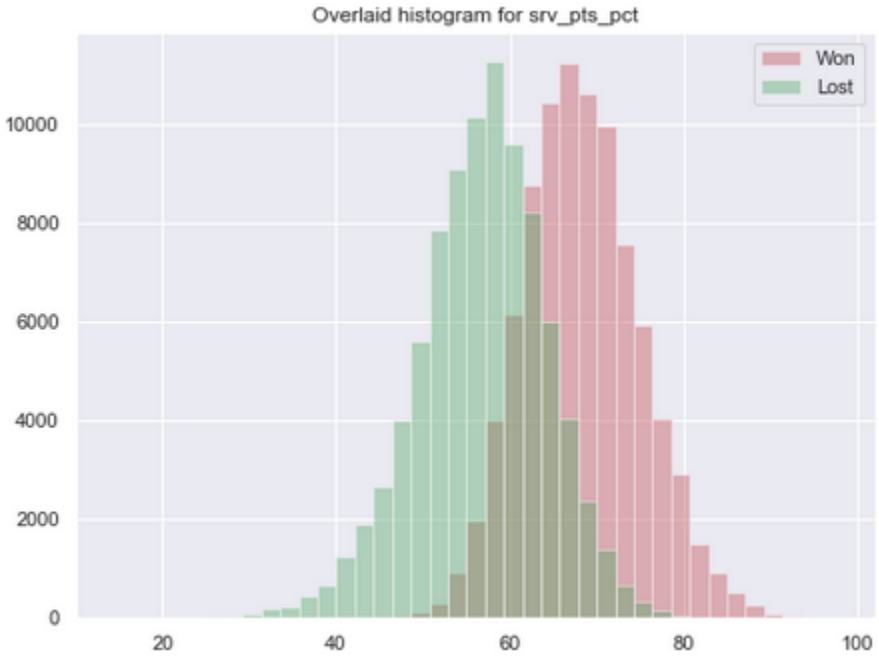
Wins Mean: 76.23225218749391
No-Wins Mean: 65.35481512219901
Mean Diff: 10.877437065294899
H0 Diff: 0

p: 0.0
CI: [-0.09521021 0.09905118]
ME: 0.09875387007053657



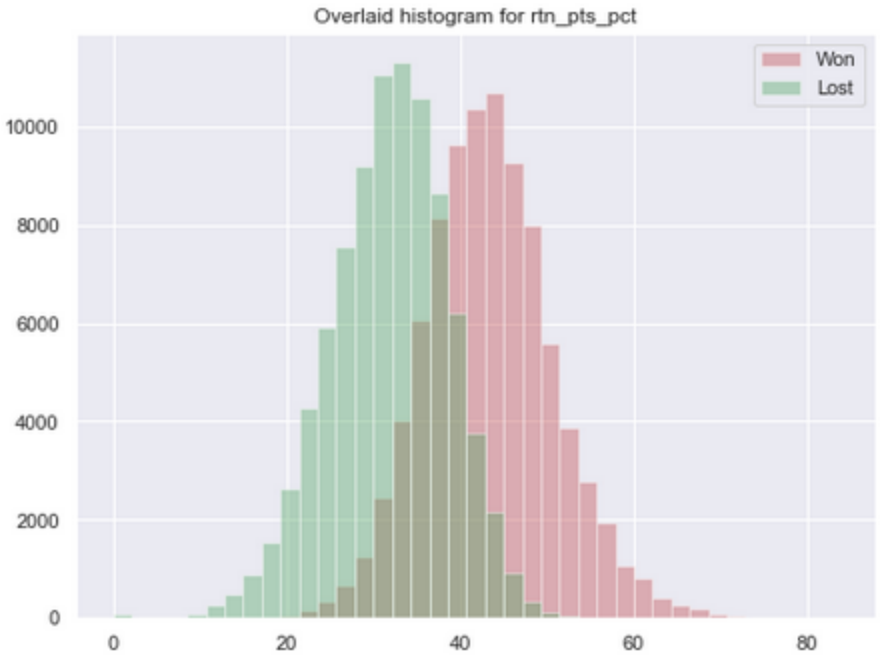
Wins Mean:
56.11935795826883
No-Wins Mean:
44.72161921208218
Mean Diff:
11.397738746186647
H0 Diff: 0

p: 0.0
CI: [-0.11144302
0.10846935]
ME:
0.10890684540854724



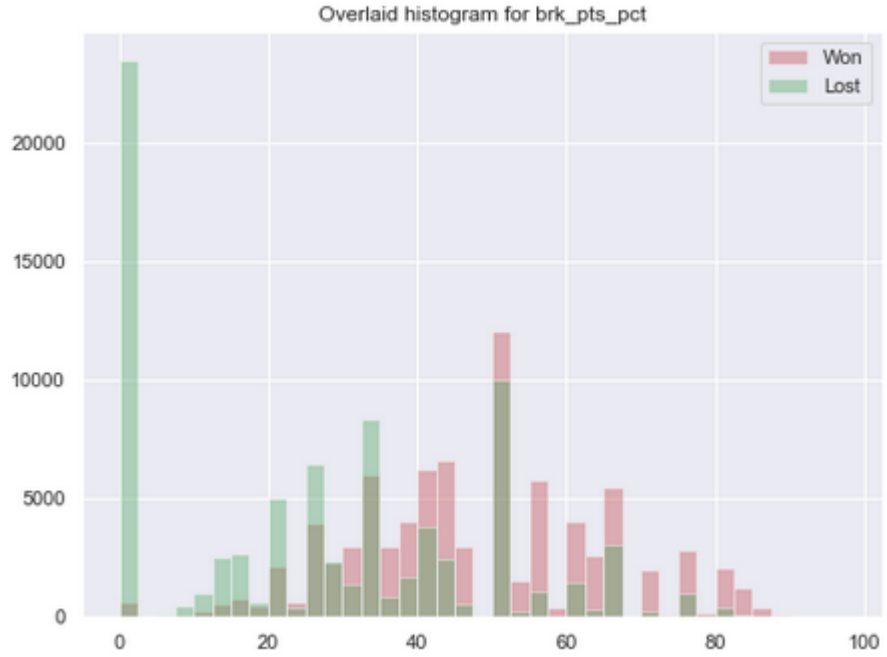
Wins Mean:
68.22250043301648
No-Wins Mean:
56.694481221203695
Mean Diff:
11.528019211812783
H0 Diff: 0

p: 0.0
CI: [-0.08612867
0.08654083]
ME:
0.0869216829786314



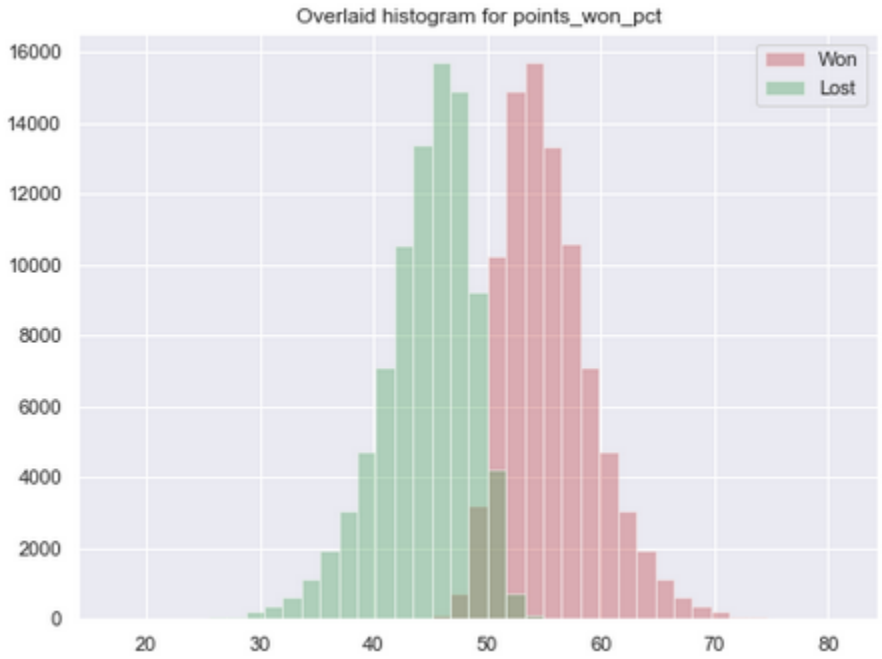
Wins Mean:
43.3193882825373
No-Wins Mean:
31.80259127897296
Mean Diff:
11.516797003564335
H0 Diff: 0

p: 0.0
CI: [-0.08571761
0.08432219]
ME:
0.08365446555536553



Wins Mean:
49.53499443350504
No-Wins Mean:
31.58515297066974
Mean Diff:
17.949841462835302
H0 Diff: 0

p: 0.0
CI: [-0.2415653
0.24239284]
ME:
0.24322333891412118



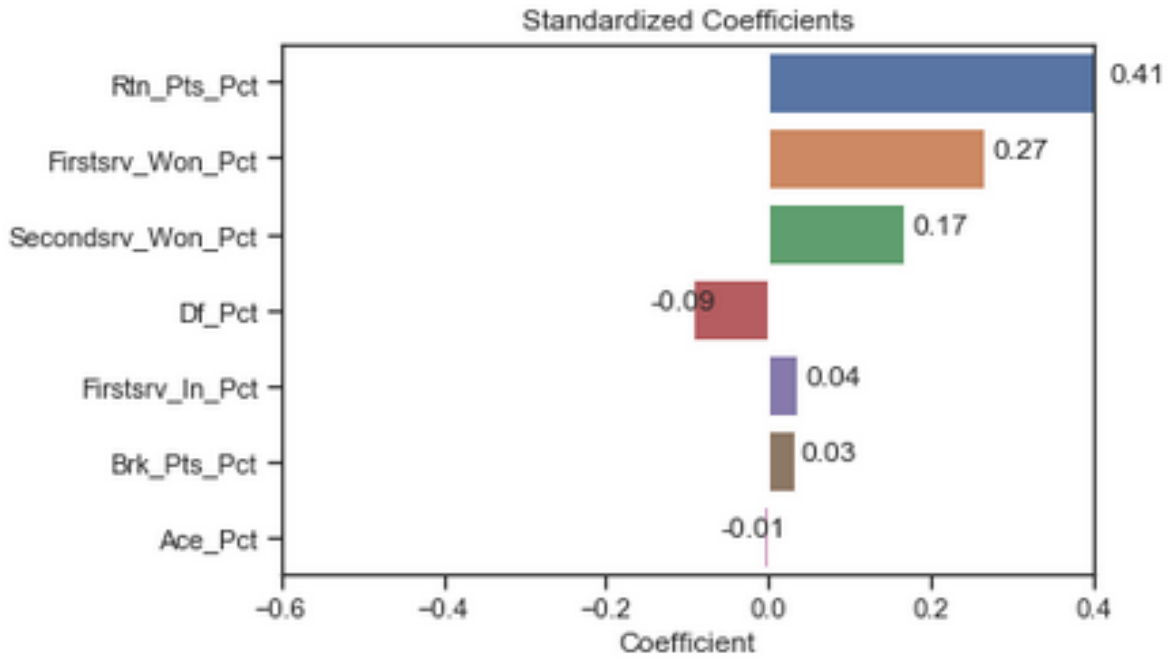

```
Wins Mean:
55.467522564258786
No-Wins Mean:
44.53247743574097
Mean Diff:
10.935045128517814
H0 Diff: 0

p: 0.0
CI: [-0.06302145
0.06394967]
ME:
0.06362357961431575
```

For all features above, the probability (p-value) was <0.001, allowing me to reject the null hypothesis (H0) that Wins and Non-Wins were equally distributed for the respective feature. In the next step I'm going to look at the correlation of these features with each other and with the target variable.

Please note that conducting a multitude of Z-tests increases the likelihood of a Type 1 Error. Unfortunately, use of ANOVA or Chi-squared is not applicable when using continuous data to predict a binary target. ANOVA would be helpful if we had a multitude of categorical data and a continuous target variable while Chi-squared allows to compare categorical data.

Correlation. As pointed out above, exploring relationships between continuous variables with binary outcomes comes with a few challenges especially when we're attempting to use popular statistical tools (Pearson's R, ANOVA etc.). Due to this issue for this project we've used logistic regression instead.

As mentioned earlier, because Total Points Won can be broken into Serve Points and Return Points Won, I removed it from the correlation table. Likewise, Serve Points Won can be broken into First and Second Serve Points and was removed for the same reason.



Using Logistic Regression's Beta based on standardized values allowed us to evaluate the relative importance of the features used. We can see at the top are three features to detect Wins: Return Points Won Percent, First Serve Points Won Percent and Second Serve Points Won Percent. The bottom three features, First Serve In Percent, Break Points Converted Percent and Aces Percent seem to have very little influence on whether a player wins the match. Hence, they will be dropped from the feature list.

Predicting Match Winners Model

In this section, I'm going to engineer the features, create a model and then test it for performance. Given the objective of creating a model that predicts match winners, this is a classification project. Using Scikit Learn, I used two different classification models: Logarithmic Regression and Random Forest.

Data & Feature Preparation. The data was prepared by creating two new initial datasets, one with only the features and one with the predictor, 'win', included in them. Then, to minimize the potential for overfitting, I divided the data into a training, validation and test dataset in a ratio of 60-20-20. This also allows for tuning the dataset using Hyperparameter Optimization.

Hyperparameter Optimization. Once the data was divided, I used the training set to optimize the hyperparameters. I fit and evaluated the basic models for each algorithm using 5-fold cross validation. This process identified the settings that could be used to optimize the performance of the models. For logistic regression, I used the standardized correlation coefficients to reduce the number of features. For random forest, I used GridSearchCV to identify the best performing parameters. For both models, the scores for accuracy, precision and recall helped identify the final model to use on the test data set.

Results. Using the features, metrics and parameters from above, Random Forest won due to its ability to predict without overfitting (as much as logistic regression did). There still seems to be some overfitting in the model, but this is most likely due to the limited number of features available rather than performance of the model. After dropping some features to reduce overfitting, the results of the Random Forest model GridSearchCV were as follows:

```
def print_results(results):
    print('BEST PARAMS: {}\n'.format(results.best_params_))

    means = results.cv_results_['mean_test_score']
    stds = results.cv_results_['std_test_score']
    for mean, std, params in zip(means, stds, results.cv_results_['params']):
        print('{} (+/-{}) for {}'.format(round(mean, 3), round(std * 2, 3), params))

rf = RandomForestClassifier()
parameters = {
    'n_estimators': [100, 500, 1000],
    'max_depth': [20, 50, None]
}

cv = GridSearchCV(rf, parameters, cv=5)
cv.fit(tr_features_new, tr_labels.values.ravel())

print_results(cv)
```

```
BEST PARAMS: {'max_depth': 20, 'n_estimators': 1000}

0.914 (+/-0.003) for {'max_depth': 20, 'n_estimators': 100}
0.914 (+/-0.002) for {'max_depth': 20, 'n_estimators': 500}
0.914 (+/-0.003) for {'max_depth': 20, 'n_estimators': 1000}
0.913 (+/-0.003) for {'max_depth': 50, 'n_estimators': 100}
0.914 (+/-0.002) for {'max_depth': 50, 'n_estimators': 500}
0.914 (+/-0.002) for {'max_depth': 50, 'n_estimators': 1000}
0.913 (+/-0.002) for {'max_depth': None, 'n_estimators': 100}
0.914 (+/-0.003) for {'max_depth': None, 'n_estimators': 500}
0.914 (+/-0.002) for {'max_depth': None, 'n_estimators': 1000}
```

Re-running the top three on the validation dataset gave the following results:

```
rf1 = RandomForestClassifier(n_estimators=500, max_depth=50)
rf1.fit(tr_features_new, tr_labels.values.ravel())

rf2 = RandomForestClassifier(n_estimators=1000, max_depth=None)
rf2.fit(tr_features_new, tr_labels.values.ravel())

rf3 = RandomForestClassifier(n_estimators=500, max_depth=None)
rf3.fit(tr_features_new, tr_labels.values.ravel())
```

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
                       max_depth=20, max_features='auto', max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=10000,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

```
for mdl in [rf1, rf2, rf3]:
    y_pred = mdl.predict(val_features_new)
    accuracy = round(accuracy_score(val_labels, y_pred), 3)
    precision = round(precision_score(val_labels, y_pred), 3)
    recall = round(recall_score(val_labels, y_pred), 3)
    print('MAX DEPTH: {} / # OF EST: {} -- A: {} / P: {} / R: {}'.format(mdl.max_depth,
                                                                         mdl.n_estimators,
                                                                         accuracy,
                                                                         precision,
                                                                         recall))
```

```
MAX DEPTH: 50 / # OF EST: 500 -- A: 0.914 / P: 0.915 / R: 0.913
MAX DEPTH: None / # OF EST: 1000 -- A: 0.914 / P: 0.915 / R: 0.913
MAX DEPTH: None / # OF EST: 500 -- A: 0.914 / P: 0.915 / R: 0.913
```

I chose the 2nd Random Forest Classifier model since it is better to use more estimators if the results are similar. So running the parameters (1000 estimators, no maximum depth) on the test dataset resulted in the following:

```
y_pred = rf2.predict(te_features_new)
accuracy = round(accuracy_score(te_labels, y_pred), 3)
precision = round(precision_score(te_labels, y_pred), 3)
recall = round(recall_score(te_labels, y_pred), 3)
print('MAX DEPTH: {} / # OF EST: {} -- A: {} / P: {} / R: {}'.format(rf2.max_depth,
                                                                     rf2.n_estimators,
                                                                     accuracy,
                                                                     precision,
                                                                     recall))
```

```
MAX DEPTH: None / # OF EST: 1000 -- A: 0.915 / P: 0.917 / R: 0.913
```

So with an accuracy, precision and recall scores of 0.915, 0.917 and 0.913 respectively, the model is able to predict match winners correctly 91.5% of the time. Overall, this is a pretty reliable model.

Conclusion

This project looked at the history of professional men's tennis match statistics from 1991-2016. We explored the ATP statistics that were available and examined the averages as well as the extremes in the data.

We were able to identify which match statistics impacted match outcomes the most, at least from those that were available. The major issue, however, is that there weren't any great revelations in terms of what specifically can be done to influence performance improvement in those statistics that mattered. Although players don't win matches simply by "winning points", as in other major professional sports, the player who won the most points won almost 95% of the time.

What does this mean? Well, it means first, that it is important to win points on both ends of the court—whether as the server or the returner. Second, it shows that a player doesn't have to win all of the statistical categories that measure specific performance in a category. That, however, is the goal for next time once more specific data is available.