PGA Tour Prediction

Lexi DeLorimiere

Table of Contents

Background

As a collegiate golfer, I have been interested in what sets successful players apart from the rest of the playing field. Being that golf is a very detailed sport, there are several factors that contribute to a good round.
From this data, I conducted exploratory data analysis to examine the distribution of players over several game-related variables, identify outliers, and learn more about how the game has changed between 2010 and 2018. To forecast a player's winnings and profits, I also used a variety of supervised machine learning models. I used multiple classification approaches to predict the player's win, including logistic regression, SVM (Support Vector Machines), and Random Forest Classification. I discovered that the Random Forest Classification approach gave me the strongest results. By using linear regression I was able to forecast a player's earnings.

Problem Scenario

In this project, I aim to accurately predict a player's earnings based off a training model. As previously stated, there are various factors that contribute to the game of golf. The problem lies within inconsistency regarding weather, different events

played, COVID-19 absences, etc. Given these inconsistencies, the results might not be as accurate without a more in-depth dataset.

Goals

- Predict if a player will win based on game-related variables
- Predict a player's earnings
- Visualize how golf has changed from 2010-2018

Dataset Dictionary

The dataset I am using is originally from Kaggle. It consists of the following variables.

Variable Title	Description
Player Name	Name of the golfer
Rounds	The number of rounds a player played
Fairway Percentage	The percentage of time a tee shot lands on the fairway
Year	The year in which the statistic was collected

Avg Distance	The average distance of the tee-shot
gir	(Green in Regulation) is met if any part of the ball is touching the putting surface while the number of strokes taken is at least two fewer than par
Average Putts	The average number of strokes taken on the green
Average Scrambling	Scrambling is when a player misses the green in regulation, but still makes par or better on a hole
Average Score	Average Score is the average of all the scores a player has played in that year
Points	The number of FedExCup points a player earned in that year. These points can be earned by competing in tournaments.
Wins	The number of competitions a player has won in that year
Top 10	The number of competitions where a player has placed in the Top 10
Average SG Putts	Strokes gained: putting measures how many strokes a player gains (or loses) on the greens.
Average SG Total	The Off-the-tee + approach-the-green + around-the-green + putting statistics combined
SG:OTT	Strokes gained: off-the-tee measures player performance off the tee on all par-4s and par-5s.
SG:APR	Strokes gained: approach-the-green measures player performance on approach shots. Approach shots include all shots that are not from the tee on par-4 and par-5 holes and are not included in strokes gained: around-the-green and strokes gained: putting. Approach shots include tee shots on par-3s.
SG:ARG	Strokes gained: around-the-green measures player performance on any shot within 30 yards of the edge of the green. This statistic does not include any shots taken on the putting green.
Money	The amount of prize money a player has earned from tournaments

Data Exploration

After viewing the data, I decided to clean the data further. The changes I made were:

- For the columns Top 10 and Wins, convert the null values to 0s.
- Change Top 10 and Wins into an integer
- Drop null values for players who do not have the full statistics
- Change the columns Rounds into integer
- Change points to integer
- Remove the dollar sign ($) and commas in the Money Column

```
# replacing null values in Top 10
data['Top 10'].fillna(0, inplace=True)
data['Top 10'] = data['Top 10'].astype(int)

# replacing null values with 0 in # of wins
data['Wins'].fillna(0, inplace=True)
data['Wins'] = data['Wins'].astype(int)

# dropping null values
data.dropna(axis = 0, inplace=True)
```

```
#changing round to int
data['Rounds'] = data['Rounds'].astype(int)

# changing points to int
data['Points'] = data['Points'].apply(lambda x: x.replace(',',''))
data['Points'] = data['Points'].astype(int)
```

```
#removing $ and , from money variable
data['Money'] = data['Money'].apply(lambda x: x.replace('$',''))
data['Money'] = data['Money'].apply(lambda x: x.replace(',',''))
data['Money'] = data['Money'].astype(float)
```

Using Python, I used various libraries to perform involved data analysis. There are a total of 1674 rows and 18 columns.

	Player Name	Rounds	Fairway Percentage	Year	Avg Distance	gir	Average Putts	Average Scrambling	Average Score	Points	Wins	Top 10	Average SG Putts	Average SG Total	SG:OTT	SG:APR	SG:ARG
0	Henrik Stenson	60	75.19	2018	291.5	73.51	29.93	60.67	69.617	868	0	5	-0.207	1.153	0.427	0.960	-0.027
1	Ryan Armour	109	73.58	2018	283.5	68.22	29.31	60.13	70.758	1006	1	3	-0.058	0.337	-0.012	0.213	0.194
2	Chez Reavie	93	72.24	2018	286.5	68.67	29.12	62.27	70.432	1020	0	3	0.192	0.674	0.183	0.437	-0.137
3	Ryan Moore	78	71.94	2018	289.2	68.80	29.17	64.16	70.015	795	0	5	-0.271	0.941	0.406	0.532	0.273
4	Brian Stuard	103	71.44	2018	278.9	67.12	29.11	59.23	71.038	421	0	3	0.164	0.062	-0.227	0.099	0.026
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1673	Phil Mickelson	76	52.66	2010	299.1	65.13	28.79	61.84	69.966	1629	1	5	-0.147	1.001	0.185	0.738	0.228
1674	John Daly	63	52.21	2010	305.7	65.66	29.78	53.53	71.697	97	0	0	-0.653	-0.989	0.336	-0.374	-0.298
1675	Jimmy Walker	82	51.29	2010	292.9	65.88	29.14	58.46	70.953	554	0	2	0.252	0.093	-0.538	0.336	0.047
1676	Daniel Chopra	74	51.27	2010	295.9	61.64	28.88	56.16	72.194	142	0	0	0.361	-1.096	-0.307	-1.070	-0.084
1677	Martin Flores	75	50.15	2010	300.7	64.79	29.41	54.00	71.882	137	0	1	-0.106	-0.883	-0.223	-0.553	-0.001

1674 rows × 18 columns

After cleaning the data, these are the datatypes and non-null count for each variable.

#	Column	Non-Null Count	DType
0	Player Name	1674 non-null	Object
1	Rounds	1674 non-null	Int64

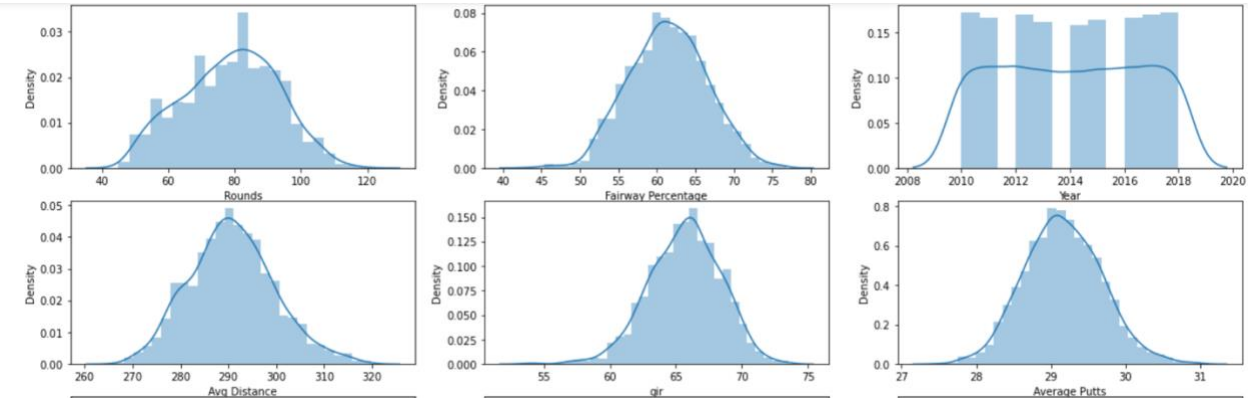
2	Fairway Percentage	1674 non-null	Float64
3	Year	1674 non-null	Int64
4	Avg Distance	1674 non-null	Float64
5	gir	1674 non-null	Float64
6	Average Putts	1674 non-null	Float64
7	Average Scrambling	1674 non-null	Float64
8	Average Score	1674 non-null	Float64
9	Points	1674 non-null	Int64
10	Wins	1674 non-null	Int64
11	Top 10	1674 non-null	Int64
12	Average SG Putts	1674 non-null	Float64
13	Average SG Total	1674 non-null	Float64
14	SG:OTT	1674 non-null	Float64
15	SG:APR	1674 non-null	Float64
16	SG:ARG	1674 non-null	Float64
17	Money	1674 non-null	Object

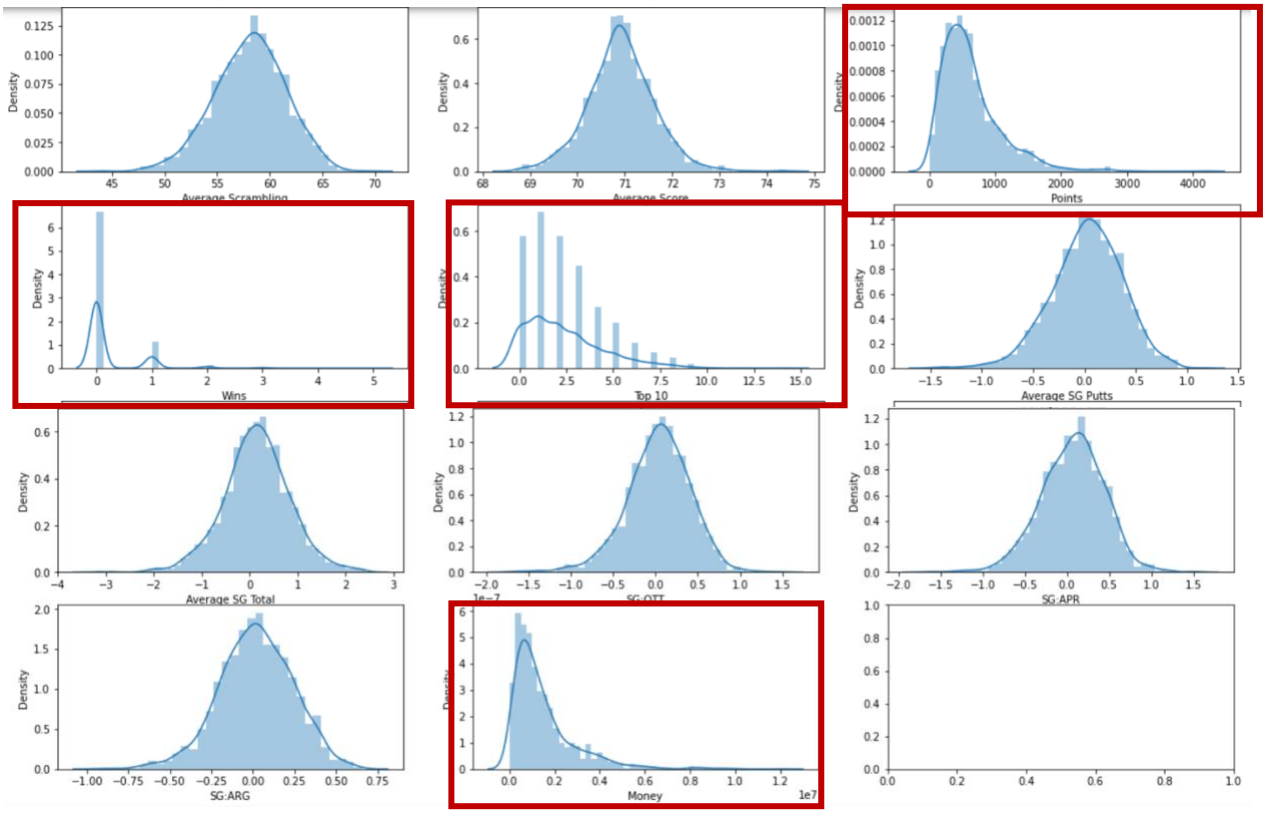
Statistical measures of the data:

	Rounds	Fairway Percentage	Year	Avg Distance	gir	Average Putts	Average Scrambling	Average Score	Points	Wins	Top 10	A
count	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1674.000000	1
mean	78.769415	61.448614	2014.002987	290.786081	65.667103	29.163542	58.120687	70.922877	631.125448	0.206691	2.337515	
std	14.241512	5.057758	2.609352	8.908379	2.743211	0.518966	3.386783	0.698738	452.741472	0.516601	2.060691	
min	45.000000	43.020000	2010.000000	266.400000	53.540000	27.510000	44.010000	68.698000	3.000000	0.000000	0.000000	
25%	69.000000	57.955000	2012.000000	284.900000	63.832500	28.802500	55.902500	70.494250	322.000000	0.000000	1.000000	
50%	80.000000	61.435000	2014.000000	290.500000	65.790000	29.140000	58.290000	70.904500	530.000000	0.000000	2.000000	
75%	89.000000	64.910000	2016.000000	296.375000	67.587500	29.520000	60.420000	71.343750	813.750000	0.000000	3.000000	
max	120.000000	76.880000	2018.000000	319.700000	73.520000	31.000000	69.330000	74.400000	4169.000000	5.000000	14.000000	

Data Visualization

I looked at the distribution for each variable. Based on the graphs, it appears that most are normally distributed. However, we can see that Points, Wins, Top 10, and Money are all right skewed. While the average player will have no wins and just a few Top 10 placings, the best players will have several Top 10 finishes and wins that allow them to earn more from tournaments.



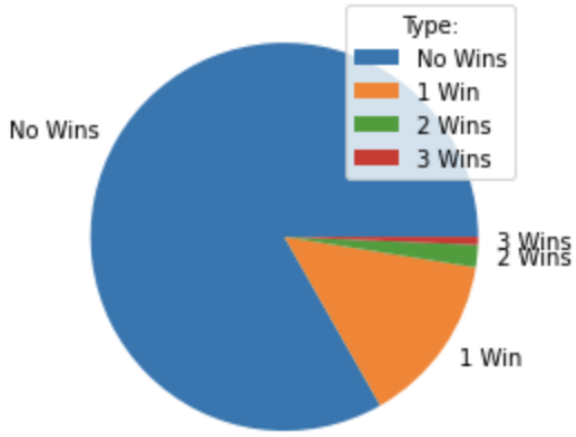


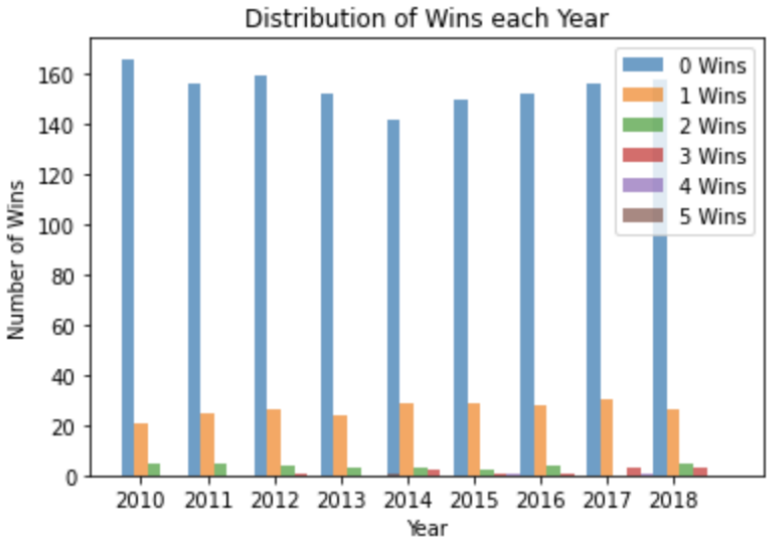
This table shows the number of players who won each year.

Wins	0	1	2	3	4	5
Year						
2010	166	21	5	0	0	0
2011	156	25	5	0	0	0
2012	159	26	4	1	0	0

2013	152	24	3	0	0	1
2014	142	29	3	2	0	0
2015	150	29	2	1	1	0
2016	152	28	4	1	0	0
2017	156	30	0	3	1	0
2018	158	26	5	3	0	0

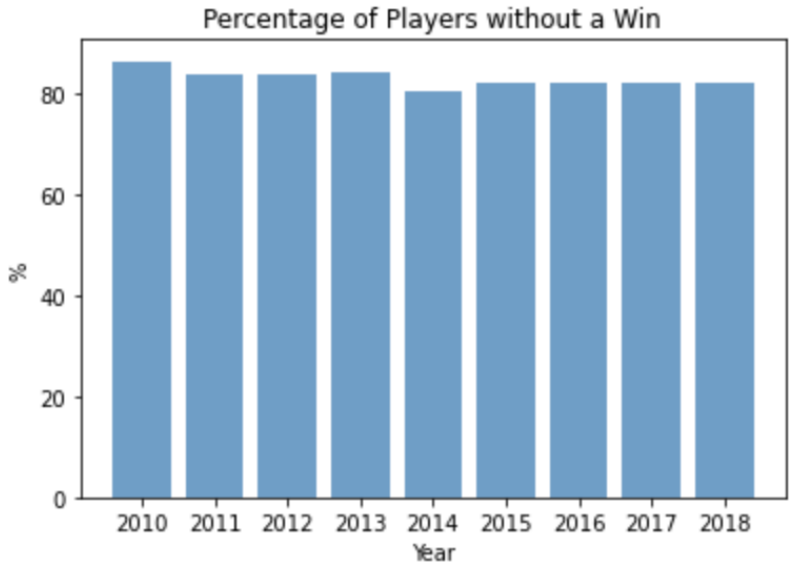
From a pie and bar chart, we can see the representation between number of wins.



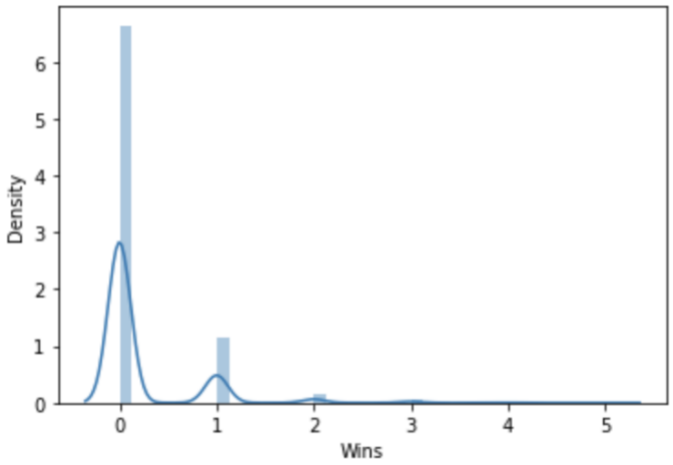


Based on the results, we can see that it is rare to have a player win more than once. Majority of players do not win, and very few win more than once.

In the bar chart below, we can see that the percentages of players without a win is about 80% with little to no variation.



The distplot refers to the distribution plot. It takes an input as an array and plots a curve corresponding to the distribution of points in the array. This plot displays two different plots for the ‘Wins’ variable.



This table shows the percentages of players who did not place in the Top 10 for that year.

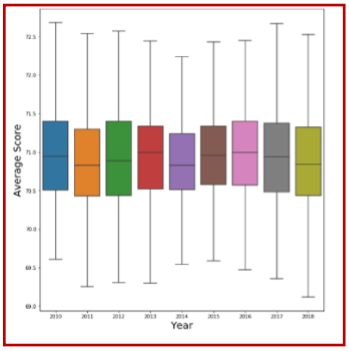
Year	Percentage
2010	17.19%
2011	26.27%
2012	23.16%
2013	18.99%
2014	16.48%
2015	18.58%
2016	20.00%
2017	15.79%
2018	17.19%

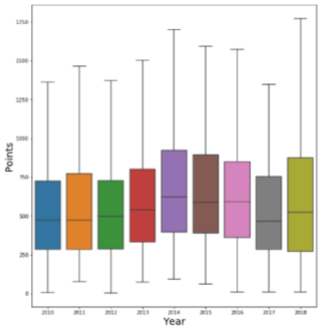
Based on the results, we can see that only around 20% of players did not place in the Top 10. The variation for these players is on 9.47%, meaning that this statistic does not change much per year.

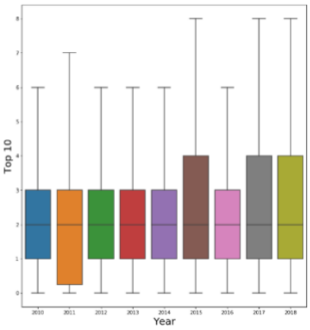
The table below shows which player made the most money each year. Some of the most recognizable were Jordan Speith's earning of 12 million dollars and Justin Thomas earning the most money in both 2017 and 2018.

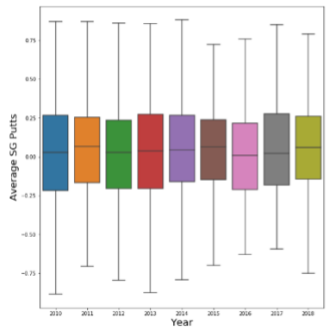
Year	Money	Player Name
2010	$4,910,477	Matt Kuchar
2011	$6,683,214	Luke Donald
2012	$8,047,952	Rory McIlroy
2013	$8,553,439	Tiger Woods
2014	$8,280,096	Rory McIlroy
2015	$12,030,465	Jordan Spieth
2016	$9,365,185	Dustin Johnson
2017	$9,921,560	Justin Thomas
2018	$8,694,821	Justin Thomas

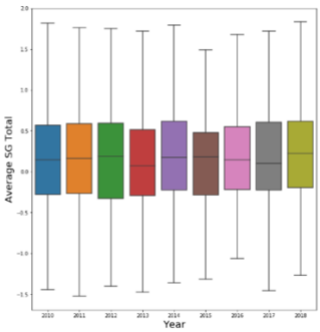
I created box plots to analyze the change of each statistic over time. Most variables had no change, while money, average score, and rounds had some change from 2010-2018.

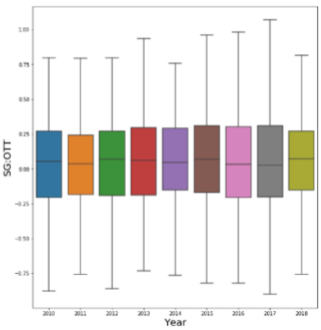

Average Score
Year

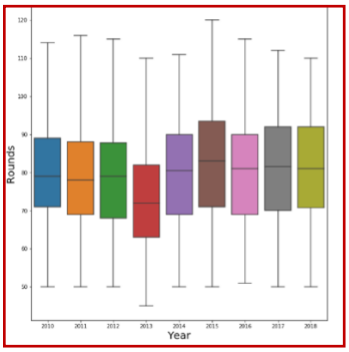

Points
Year

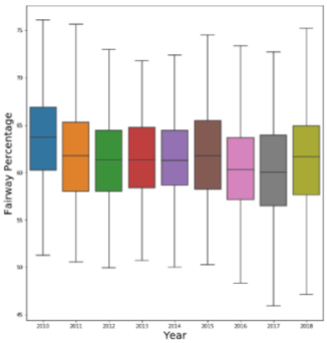

Top 10
Year

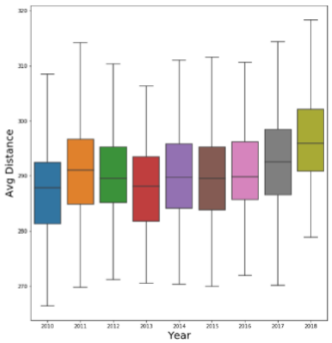

Average SG Putts
Year

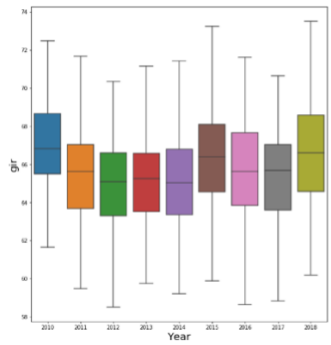

Average SG Total
Year

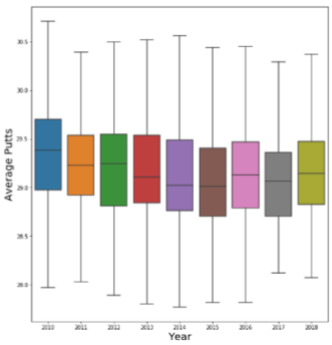

SG:OTT
Year

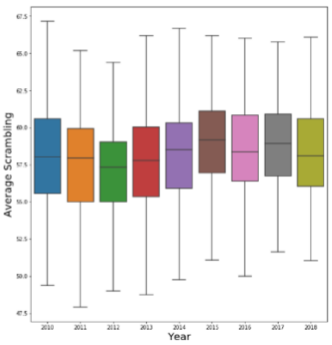

Rounds
Year

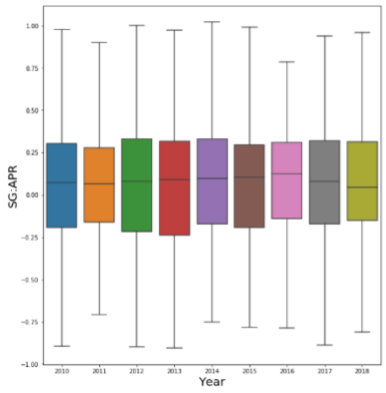

Fairway Percentage
Year

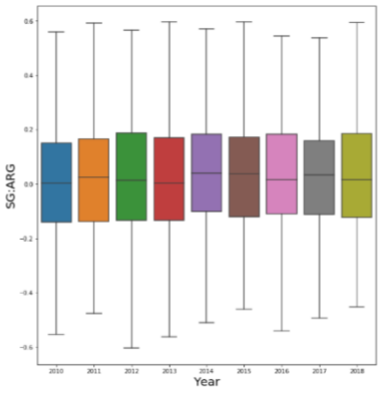

Avg Distance
Year

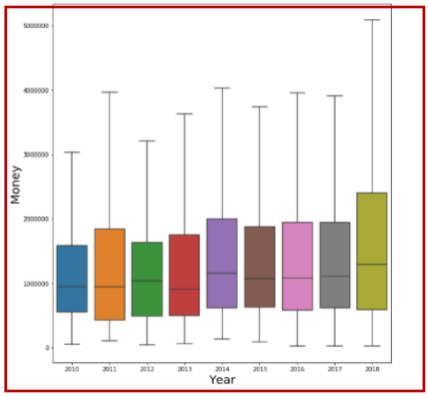

gir
Year


Average Putts
Year

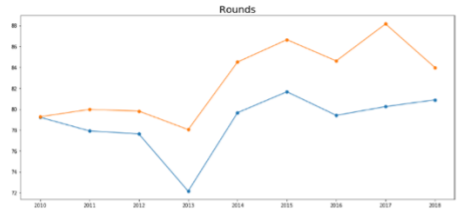

Average Scrambling
Year

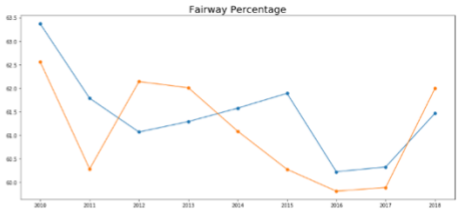


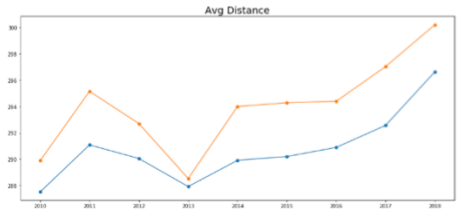


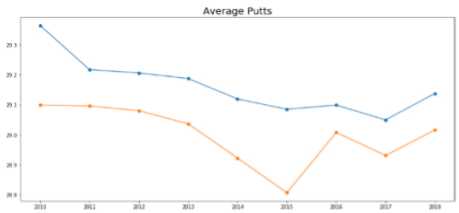


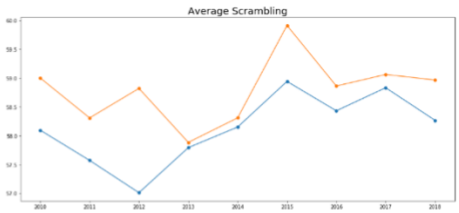
From the graphs below, we can see the average scores of the best players (players with a win) versus the PGA Tour average. This can give us an indication to which statistics help players win. Fairway percentage and greens in regulation does not seem to contribute as much to a player's win. However, we can see that all the strokes gained statistics have a large impact on the wins of these players. We can also see that the average score and average putts are lower for players with a win.

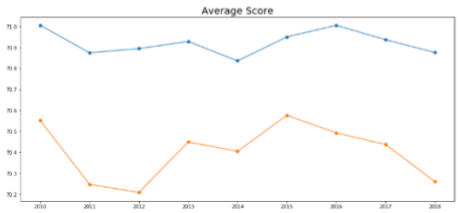


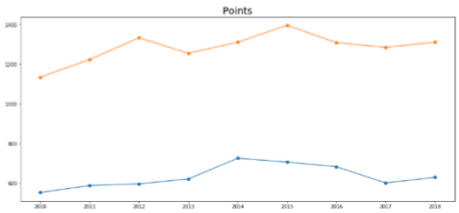


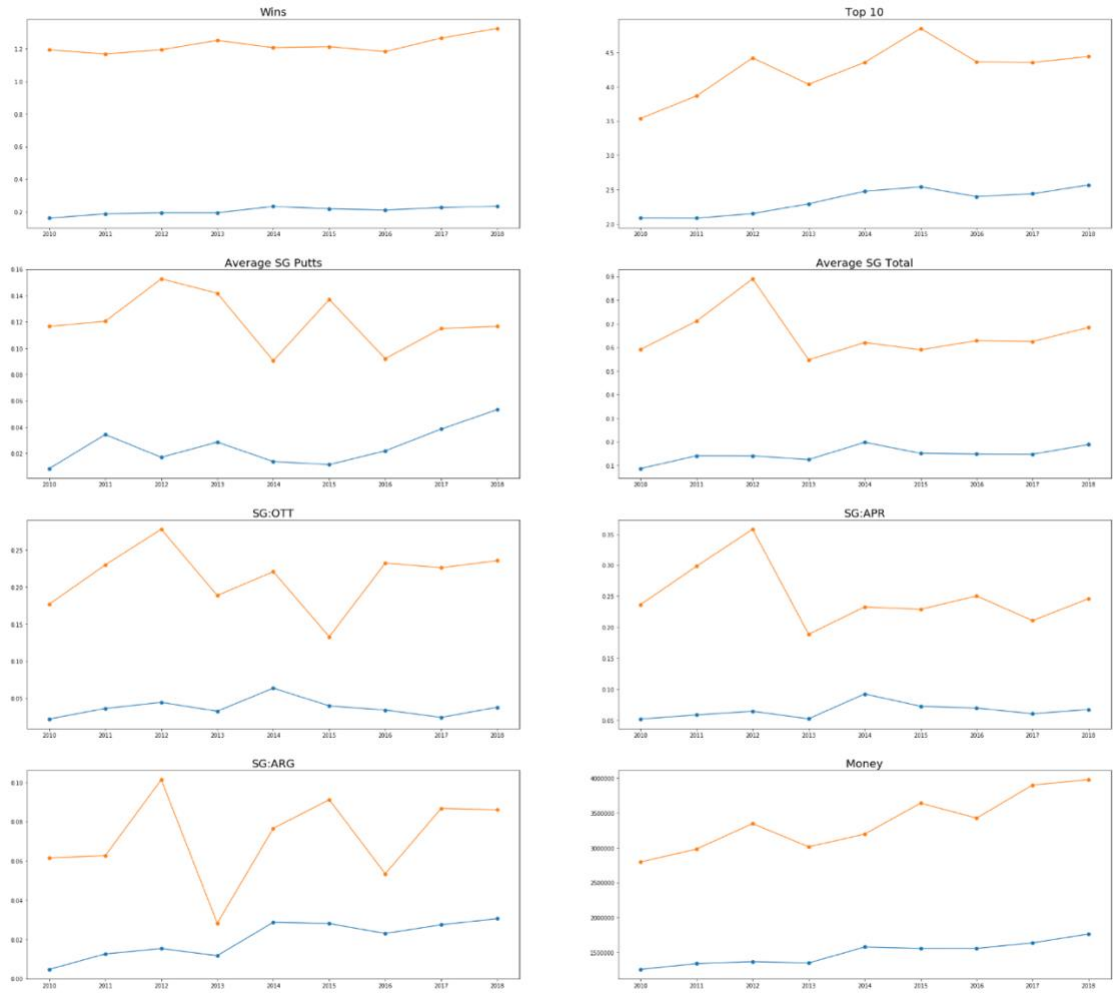




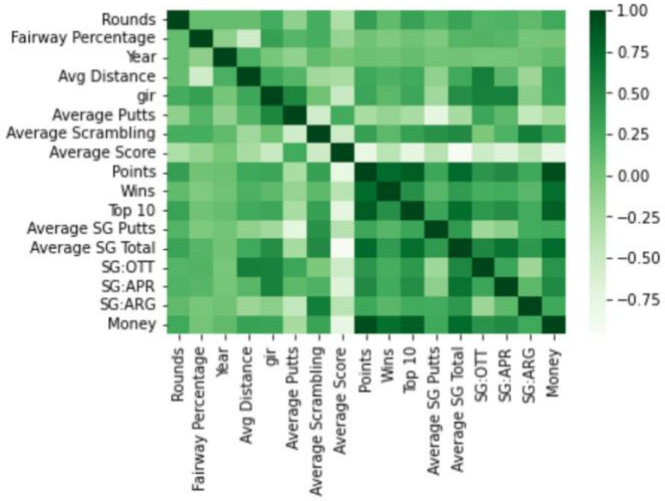








I created a heatmap to visualize the correlation matrix. We can see that money is strongly correlated with FedExCup points. We can also observe that fairway percentage, year, and rounds are not correlated to wins.



Methodology/Model Building

In order to predict winners and earnings, I used various machine learning models to explore which models could accurately classify if a player is going to win in that year. To measure the models, I used Receiver Operating Characterisitc Area Under the Curve. (ROC AUC) The ROC AUC tells us how capable the model is at distinguishing players with a win. In addition, as the data is skewed with 83% of players having no wins in that year, ROC AUC is a better measure than the accuracy of the model.

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.metrics import confusion_matrix
from sklearn.feature_selection import RFE
from sklearn.metrics import classification_report
from sklearn.preprocessing import PolynomialFeatures
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import MinMaxScaler
```

```
#adding winner column to determine if a player won or not
data['Winner'] = data['Wins'].apply(lambda x: 1 if x>0 else 0)

#new dataframe with winner column
data2 = data.copy()

# y-value for machine learning is winner column
target = data['Winner']

#dropping the columns Player Name, Wins, and Winner from the dataframe
data2.drop(['Player Name','Wins','Winner'], axis=1, inplace=True)
print(data2.head())
```

```
   Rounds  Fairway Percentage  Year  Avg Distance    gir  Average Putts  \
0      60               75.19  2018         291.5  73.51          29.93
1     109               73.58  2018         283.5  68.22          29.31
2      93               72.24  2018         286.5  68.67          29.12
3      78               71.94  2018         289.2  68.80          29.17
4     103               71.44  2018         278.9  67.12          29.11

   Average Scrambling  Average Score  Points  Top 10  Average SG Putts  \
0               60.67         69.617     868       5            -0.207
1               60.13         70.758    1006       3            -0.058
2               62.27         70.432    1020       3             0.192
3               64.16         70.015     795       5            -0.271
4               59.23         71.038     421       3             0.164

   Average SG Total  SG:OTT  SG:APR  SG:ARG      Money
0             1.153   0.427   0.960  -0.027  2680487.0
1             0.337  -0.012   0.213   0.194  2485203.0
2             0.674   0.183   0.437  -0.137  2700018.0
3             0.941   0.406   0.532   0.273  1986608.0
4             0.062  -0.227   0.099   0.026  1089763.0
```

Based on the logistic regression, I got an accuracy of 0.9 on the training set and an accuracy of 0.91 on the test set.

```
Accuracy of Logistic regression classifier on training set: 0.90
Accuracy of Logistic regression classifier on test set: 0.91
     0   1
0  345   8
1   28  38
              precision    recall  f1-score   support

           0       0.92      0.98      0.95       353
           1       0.83      0.58      0.68        66

    accuracy                           0.91       419
   macro avg       0.88      0.78      0.81       419
weighted avg       0.91      0.91      0.91       419
```

Model Selection

I received the strongest results from the SVM and Random Forest Model.

Support Vector Machine (SVM) is a Supervised Learning algorithm, which is used for Classification as well as Regression problems. It is primarily used for Classification problems in Machine Learning.

```
def svc_class(X,y):
    X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                        random_state = 10)
    scaler = MinMaxScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)


    svclassifier = SVC(kernel='rbf', C=10000)
    svclassifier.fit(X_train_scaled, y_train)
    y_pred = svclassifier.predict(X_test_scaled)
    print('Accuracy of SVM on training set: {:.2f}'
         .format(svclassifier.score(X_train_scaled, y_train)))
    print('Accuracy of SVM classifier on test set: {:.2f}'
         .format(svclassifier.score(X_test_scaled, y_test)))


    print('ROC AUC Score: {:.2f}'.format(roc_auc_score(y_test, y_pred)))
```

```
svc_class(data2, target)
```

```
Accuracy of SVM on training set: 1.00
Accuracy of SVM classifier on test set: 0.91
ROC AUC Score: 0.84
```

The Random Forest Model was scored highly on ROC AUC Score, obtaining a value of 0.89. I observed that the Random Forest Model and the SVM could accurately classify players with and without a win.

```
Accuracy of Random Forest classifier on training set: 1.00
Accuracy of Random Forest classifier on test set: 0.94
     0   1
0  343  10
1   16  50
              precision    recall  f1-score   support

           0       0.96      0.97      0.96       353
           1       0.83      0.76      0.79        66

    accuracy                           0.94       419
   macro avg       0.89      0.86      0.88       419
weighted avg       0.94      0.94      0.94       419
```

Can I predict a player's earnings by only looking at statistics?
Below you can see that I prepared the data and used various machine learning modules.

```
earning_data = data.copy()

#y-value for machine learning is the Money column
target = earning_data['Money']

#dropping the columns Player Name, Wins, Winner, Points, Top 10, and Money from the dataframe
earning_data.drop(['Player Name','Wins','Winner','Points','Top 10','Money'], axis=1, inplace=True)

print(earning_data.head())
```

```
   Rounds  Fairway Percentage  Year  Avg Distance    gir  Average Putts  \
0      60               75.19  2018         291.5  73.51          29.93
1     109               73.58  2018         283.5  68.22          29.31
2      93               72.24  2018         286.5  68.67          29.12
3      78               71.94  2018         289.2  68.80          29.17
4     103               71.44  2018         278.9  67.12          29.11

   Average Scrambling  Average Score  Average SG Putts  Average SG Total  \
0               60.67         69.617            -0.207             1.153
1               60.13         70.758            -0.058             0.337
2               62.27         70.432             0.192             0.674
3               64.16         70.015            -0.271             0.941
4               59.23         71.038             0.164             0.062

   SG:OTT  SG:APR  SG:ARG
0   0.427   0.960  -0.027
1  -0.012   0.213   0.194
2   0.183   0.437  -0.137
3   0.406   0.532   0.273
4  -0.227   0.099   0.026
```

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, roc_auc_score
from sklearn.metrics import confusion_matrix
from sklearn.feature_selection import RFE
from sklearn.metrics import classification_report
from sklearn.preprocessing import PolynomialFeatures

from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge
from sklearn.preprocessing import PolynomialFeatures
```

```
def linear_reg(X, y):
    X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 10)
    clf = LinearRegression().fit(X_train, y_train)
    y_pred = clf.predict(X_test)

    print('R-Squared on training set: {:.3f}'
          .format(clf.score(X_train, y_train)))
    print('R-Squared on test set {:.3f}'
          .format(clf.score(X_test, y_test)))

    print('linear model coeff (w):\n{}'
         .format(clf.coef_))
    print('linear model intercept (b): {:.3f}'
         .format(clf.intercept_))
```

```
linear_reg(earning_data, target)
```

```
R-Squared on training set: 0.601
R-Squared on test set 0.640
linear model coeff (w):
[    5596.7147119      3888.01470514    29923.09760851    17645.92304676
   -19456.82389588  -512051.79339988   -53537.61424529  -708935.63288728
  1780737.55094366 -1367614.7970086   2210424.45790616  2307608.08295936
  2207460.78052006]
linear model intercept (b): 4887933.950
```

Conclusions

After analyzing and testing models from this data, I learned that several factors of golf differentiate between a player who wins and an average tour player. For example, fairway percentage and greens in regulations do not seem to contribute much to a player's win. Interestingly, all of the strokes gained statistics contribute

significantly to wins for these tour players. As a golfer myself, I was able to apply these results to my own game to see which areas need improvement. It was also interesting to see which areas of the game these professionals focus on the most.

Works Cited

jmpark746. “PGA Tour Machine Learning Project.” *Kaggle*, Apache 2.0, 30 Apr. 2019, https://www.kaggle.com/code/jmpark746/pga-tour-machine-learning-project/notebook.