Winning a Tennis Match: Classification Model

Brooke Baker, Christopher Holmstead, Connor Williams

Section 2 | Group 06

April 8, 2019

Project Abstract

This section summarizes the main question we researched and introduces the purposes, methods, results, and conclusions of our project.

Our group set out to predict the probability of a tennis player winning a tennis match. This question poses many possible solutions, but we chose to focus on the Women's Tennis Association (WTA), the Grand Slam tournaments, and generating a prediction after the first set had been played. We found that making a prediction after the first set had been played was over 10 times more accurate than making a generalized prediction prior to the match starting. Our outcome variable is a categorical value; however, we were also interested in exploring the probabilities of winning a match in addition to seeing the predicted category.

Because our dataset contained match statistics dating back through 1968 and for several tournaments, we needed to narrow in on a subpopulation. We chose to focus on the four Grand Slam tournaments (the Australian Open, French Open, Wimbledon, and US Open), and as was stated above, we limited our data to the WTA.

As for the purpose of our project, as athletes, our research question held personal interest to our group. We also recognize that making an accurate prediction model in this field of study is often used in gambling. While this is not our intent, we recognize the psychological edge it could give a player to know--based on their last set performance--the prediction for the match outcome. The player could then make improvements and develop an overall strategy that would be beneficial in the next set.

We began by cleaning the dataset for missing values and removing insignificant variables. We were ultimately able to reduce the 49 original variables down to the top six. We then conducted several iterations of various categorical algorithms including Logistic Regression, CART, KNN, ANN, Boosted Tree, Decision Forest, and SVM and compared model indicators to determine the best model.

We ultimately determined the best model for forecasting match outcome was an Artificial Neural Network (ANN) using 20 tours and two hidden layers; the first layer had 3 TanH nodes and 1 Linear node while the second layer had 1 TanH node and 1 Linear node. A few of the model quality indicators we evaluated are recorded in the table below. As can be seen, our model is very accurate.

Recall	Precision	Error
96.1%	95.5%	4.3%

Since ANN cannot be directly interpreted, we also explain the logistic regression model in the Input Variable Evaluation and Model Testing section. This provides greater insight into the individual variables' contributions in the overall model.

For the remainder of this paper, we will describe the conditions of our original dataset and outline the processes and procedures we followed to clean and prepare the data for model testing. After establishing this foundation, we'll show and explain our test results and the conclusions that can be drawn from our project.

Data Description and Preparation

This section details the process of finding a tennis dataset and preparing the data for modeling.

Data Source

Our group found the tennis dataset for our project on a GitHub repository.[1] This repository was created four years ago and contains a csv file of tennis statistics for the Women's Tennis Association (WTA) matches for each year since 1968. The last update was on January 1, 2019, which posted the match data for 2018.

Initial Dataset Conditions

Each dataset contains 49 variables specifying information about the tournament, the winner of the match, and the loser of the match. Each file has over 2800 records, and each row refers to one match. We decided to combine the last three years of data (2016, 2017, and 2018) and filtered the records to comprise matches played at the four Grand Slams (Australian Open, French Open, Wimbledon, and US Open). After these adjustments, we had a total of 1500 records, which ultimately would be doubled.

Variables Table

As was stated above, our dataset contained 49 variables initially. After some preliminary cleaning, we reduced that number to 36. After completing our data cleaning and performing some cross-validation, we found the top 11 statistically significant variables; we were further able to reduce the number of important variables to six, but we will discuss that in more depth in a later section. We've included the attribute names, data types, and descriptions for the 11 variables in the table below. Please note, these column names may differ from some of the column references in the sections below because we renamed some of the fields during our data cleaning; an explanation of our relabeling rationale can be found in the Features Engineered section.

Attribute Name	Data Type	Description
y_win	Categorical (binary)	The outcome of the match, from the player's perspective; won (w)/lost (l)
p_1stWon	Numeric (integer)	The number of points scored by the player off of a first serve during the match
o_1stWon	Numeric (integer)	The number of points scored by the opponent off of a first serve during the match
p_2ndWon	Numeric (integer)	The number of points scored by the player off of a second serve during the match
o_2ndWon	Numeric (integer)	The number of points scored by the opponent off of a second serve during the match
p_svpt	Numeric (integer)	The number of first serves the player had during the match
o_svpt	Numeric (integer)	The number of first serves the opponent had during the match
p_ht	Numeric (integer)	The height of the player in centimeters
o_ht	Numeric (integer)	The height of the opponent in centimeters

[1] https://github.com/JeffSackmann/tennis_wta

p_ioc	Categorical (47 values)	The country code of the player as defined by The International Olympic Committee (IOC)
o_ioc	Categorical (47 values)	The country code of the opponent as defined by The International Olympic Committee (IOC)
o_bpFaced	Numeric (integer)	The number of break points the opponent faced during the match; the number of times the player was about to win the game on the opponent's serve

Missing Data

A few columns contained a significant portion of blank records. Namely, the winner_seed and loser_seed columns were missing more than 50 percent of their values. Given this fact and the high correlation with winner_rank and loser_rank respectively, we removed these columns from the dataset. Additionally, winner_entry and loser_entry were more than 50 percent blank. We chose to eliminate these columns as well.

The players' and opponents' heights (in centimeters) were recorded in the dataset. However, out of the 264 unique players, 178 entries were missing. Since height could possibly be predictive in match outcome and tennis player information is relatively accessible, we chose to Google the missing values. This proved to be an effective method of finding and filling the missing values. We were unable to find 10 players' heights. For these records, we used the average height to fill the missing values.

A few player and opponent ranks were missing. For these missing values, we chose to use the average rank value. Additionally, one record was missing the match statistics. We chose to remove this record since that information is vital to our model.

Outliers

There were some outliers in the data. However, based on our domain knowledge and the univariate visualizations, we determined these were valid observations rather than mislabeled ones. Therefore, we kept the observations to see the complete story.

Data Excluded

Since 49 columns is too many for determining predictive significance, we utilized our domain knowledge to eliminate some of the fields. We began by removing tourney_id since we have the tourney_name, which conveys more meaning. We also eliminated tourney_date because the Majors are played at the same time every year, so having the tourney_name will also communicate the variation in weather. From there we eliminated draw_size. Draw size refers to the initial number of participants in the tournament. All the Majors all have the same draw size of 128 so this information would not provide any insightful information in our algorithm. In addition to draw size, we also removed tourney_level (since the tournament level is the same across the Grand Slams), best_of (since all women matches are the best of three sets), match_num (since the match number is not as important as tournament round and has too many unique values to hold statistical significance), and winner_rank_points and loser_rank_points (since these columns are highly correlated with the player's ranking but ranking is more commonly used).

We held onto the winner_id, winner_name, loser_id, and loser_name fields initially but ultimately ended up removing those fields before testing different models. Our rationale for this decision was

supported by the fact that we wanted our algorithm to be dependent on the player's ranking and statistics but independent of the actual player. Additionally, these fields contain too many unique values to hold predictive significance for a general use-case.

At this point, we had approximately 30 independent variables remaining. We removed some additional columns after performing cross-validation and running some initial (mainly logistic regression) test models. We will discuss the decisions we made and the rationale behind them in the Data Understanding section below.

Features Engineered

While we knew who won the match based on the winner and loser variables, we did not formally have a dependent variable column. This response variable would need to contain records of both outcomes (won and lost) in order for our model to have predictive accuracy. Therefore, we needed to have a column that would label the match outcome from a *player's* perspective--not just the *winner's* and not just the *loser's*. We created this neutrality among players by relabeling all the columns that referenced the winner (contained "winner" or "w") to player ("p"). We repeated this process for the columns that referenced the loser ("loser" or "l") by changing the column names to opponent ("o") respectively. We then created the y_win column and recorded the match outcome based on the player's perspective. We duplicated the dataset's records but switched the player and opponent information, so the opposite outcome could also be recorded. We used the value of "W" for "won" and "L" for "lost". As can be seen from the dependent variable's values, our dependent variable is categorical. While we used the confusion matrix counts to compare the various models, we also recognize it's more insightful to generate a probability of winning the match rather than just returning whether the player would win and lose; therefore, we were mindful of these probabilities and our cutoff values when creating the Azure Machine Learning Web Service.

In addition to creating the dependent variable, we also cleaned the tourney_name column. Specifically, the French Open is also known as Roland Garros, so we changed the "Roland Garros" values to "French Open." There were also a few discrepancies in how the US Open was spelled; we fixed these values before proceeding.

We modified the original dataset's age variable because its values were listed with an abnormal level of significant figures—to seven or eight decimal places. We chose to round age to the nearest whole number.

We also had data on the match score. However, this column's values were not in a useful format. The match score was recorded in a single column (i.e., "6-2 7-6(4)"). Although score cannot be used in predicting the probability of a player winning a match before the match is played, it will likely prove valuable in updating the prediction after a single set has been played; therefore, we chose to leave this column in and analyze it for predictive significance later. However, having the score in this format is not beneficial to our predictions, so we split the score into three columns labeled set1, set2, and set3. Upon further discussion, we determined the information listed as a set score rather than individual player-opponent scores would not be beneficial either, so we split the columns again to be p_set_1, o_set_1, etc. As the example listed in parentheses above shows, certain sets went to tie breaker; the value in parentheses refers to the number of points the opponent scored in the game before the player won. We determined this would be too specific and too rare to be valuable, so we chose to eliminate that information when splitting the player-opponent scores. After this cleaning, we removed the original

score column as well as the set1, set2, and set3 splits. Since it's not valuable to update the probability of winning a match *after* the match was over, we also removed the player's and opponent's set_3 scores. After discussion on what we wanted our algorithm to be capable of, we chose to focus on generating a model that would predict a player's probability of winning after set 1 had been played. Therefore, we removed the set_2 scores as well.

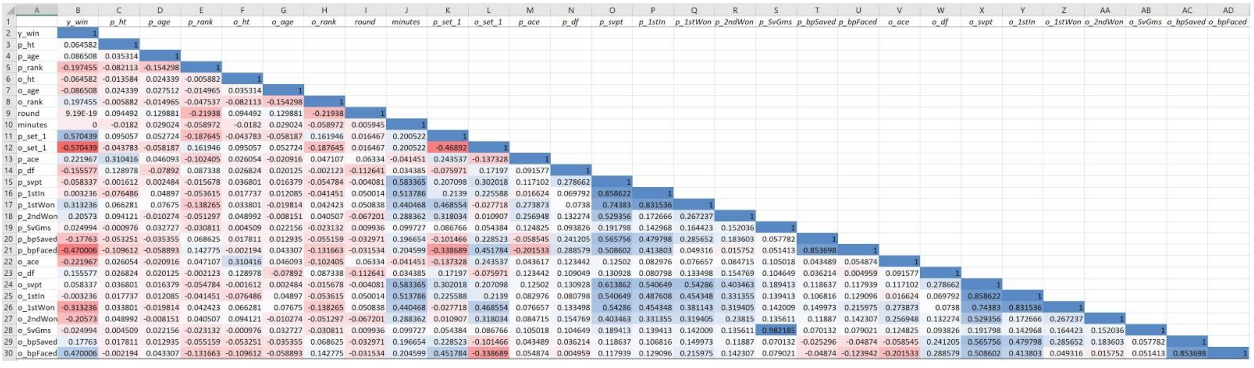
The tourney_date column listed dates in the following format: YYYYMMDD. We chose to reformat the date as MM/DD/YYYY.

The round column was recorded as a categorical variable. Since rounds are ordinal by nature, we converted this column's data type to ordinal numeric. For example, we converted "R128" (or the first round of the tournament) to "1"; we converted "F" (or the final round of the tournament) to "7". We also modified the data types of a few other columns that could/should be classified as numeric but were originally string.

Lastly, our original dataset contained composite statistics about the match (i.e., number of double faults the player had in the entire match). Since our algorithms will be focused on updating a probability after a single set, if we tried to compare a single set's statistics to a match's statistics, our model would be inflated. Therefore, we replaced the match values with the set values, based on the average values obtained by dividing the match values by the number of sets played. While this does introduce inaccuracy into the model, we determined it would be more accurate than working with the match values.

Data Understanding

This section supports the Data Description and Preparation section above. This section primarily showcases data visualizations that identify data insights. These will be helpful in generating our models.



	y_win	p_ht	p_age	p_rank	o_ht	o_age	o_rank	round	minutes	p_set_1	o_set_1	p_ace	p_df	p_svpt	p_1stIn	p_1stWon	p_2ndWon	p_SvGms	p_bpSaved	p_bpFaced	o_ace	o_df	o_svpt	o_1stIn	o_1stWon	o_2ndWon	o_SvGms	o_bpSaved	o_bpFaced
y_win	1																												
p_ht	0.064582	1																											
p_age	0.086508	0.035314	1																										
p_rank	-0.197455	-0.082113	-0.154298	1																									
o_ht	-0.064582	-0.013584	0.024339	-0.005882	1																								
o_age	-0.086508	0.024339	0.027512	-0.014965	0.035314	1																							
o_rank	0.197455	-0.005882	-0.014965	-0.047537	-0.082113	-0.154298	1																						
round	9.19E-19	0.094492	0.129881	-0.21938	0.094492	0.129881	-0.21938	1																					
minutes	0	-0.0182	0.029024	-0.058972	-0.0182	0.029024	-0.058972	0.005945																					
p_set_1	0.570439	0.095057	0.052724	-0.187645	-0.043783	-0.058187	0.161946	0.016467	0.200522																				
o_set_1	-0.570439	-0.043783	-0.058187	0.161946	0.095057	0.052724	-0.187645	0.016467	0.200522	-0.46892	1																		
p_ace	0.221967	0.310416	0.046093	-0.102405	0.026054	-0.020916	0.047107	0.06334	-0.041451	0.243537	-0.137328	1																	
p_df	-0.155577	0.128978	-0.07892	0.087338	0.026824	0.020125	-0.002123	-0.112641	0.034385	-0.075971	0.17197	0.091577	1																
p_svpt	-0.058337	-0.001612	0.002484	-0.015678	0.036801	0.016379	-0.054784	-0.004081	0.583365	0.207098	0.302018	0.117102	0.278662	1															
p_1stIn	0.003236	-0.076486	0.04897	-0.053615	0.017737	0.012085	-0.041451	0.050014	0.513786	0.2139	0.225588	0.016624	0.069792	0.858622	1														
p_1stWon	0.313236	0.066281	0.07675	-0.138265	0.033801	-0.019814	0.042423	0.050838	0.440468	0.468554	-0.027718	0.273873	0.0738	0.74383	0.831536	1													
p_2ndWon	0.20573	0.094121	-0.010274	-0.051297	0.048992	-0.008151	0.040507	-0.067201	0.288362	0.318034	0.010907	0.256948	0.132274	0.529356	0.172666	0.267237	1												
p_SvGms	0.024994	-0.000976	0.032727	-0.030811	0.004509	0.022156	-0.023132	0.009936	0.099727	0.086766	0.054384	0.124825	0.093826	0.191798	0.142968	0.164423	0.152036	1											
p_bpSaved	-0.17763	-0.053251	-0.035355	0.068625	0.017811	0.012935	-0.055159	-0.032971	0.196654	-0.101466	0.228523	-0.058545	0.241205	0.565756	0.479798	0.285652	0.183603	0.057782	1										
p_bpFaced	-0.470006	-0.109612	-0.058893	0.142775	-0.002194	0.043307	-0.131663	-0.031534	0.204599	-0.338689	0.451784	-0.201533	0.288579	0.508602	0.413803	0.049316	0.015752	0.051413	0.853698	1									
o_ace	-0.221967	0.026054	-0.020916	0.047107	0.310416	0.046093	-0.102405	0.06334	0.041451	-0.137328	0.243537	0.043617	0.123442	0.12502	0.082976	0.076657	0.084715	0.105018	0.043489	0.054874	1								
o_df	0.155577	0.028824	0.020125	-0.002123	0.128978	-0.07892	0.087338	-0.112641	0.034385	0.17197	-0.075971	0.123442	0.109049	0.130928	0.080798	0.133498	0.154769	0.104649	0.036214	0.004959	0.091577	1							
o_svpt	0.058337	0.036801	0.016379	-0.054784	-0.001612	0.002484	-0.015678	-0.004081	0.583365	0.302018	0.207098	0.12502	0.130928	0.613862	0.540649	0.54286	0.403463	0.189413	0.118637	0.117939	0.117102	0.278662	1						
o_1stIn	-0.003236	0.017737	0.012085	-0.041451	-0.076486	0.04897	-0.053615	0.050014	0.513786	0.225588	0.2139	0.082976	0.080798	0.540649	0.487608	0.454348	0.331355	0.139413	0.106816	0.129096	0.016624	0.069792	0.858622	1					
o_1stWon	-0.313236	0.033801	-0.019814	0.042423	0.066281	0.07675	-0.138265	0.050838	0.440468	-0.027718	0.468554	0.076657	0.133498	0.54286	0.454348	0.381143	0.319405	0.142009	0.149973	0.215975	0.273873	0.0738	0.74383	0.831536	1				
o_2ndWon	-0.20573	0.048992	-0.008151	0.040507	0.094121	-0.010274	-0.051297	-0.067201	0.288362	0.010907	0.318034	0.084715	0.154769	0.403463	0.331355	0.319405	0.23815	0.135611	0.11887	0.142307	0.256948	0.132274	0.529356	0.172666	0.267237	1			
o_SvGms	-0.024994	0.004509	0.022156	-0.023132	-0.000976	0.032727	-0.030811	0.009936	0.099727	0.054384	0.086766	0.105018	0.104649	0.189413	0.139413	0.142009	0.135611	0.982186	0.070132	0.079021	0.124825	0.093826	0.191798	0.142968	0.164423	0.152036	1		
o_bpSaved	0.17763	0.017811	0.012935	-0.055159	-0.053251	0.035355	0.068625	-0.032971	0.196654	0.228523	-0.101466	0.043489	0.036214	0.118637	0.106816	0.149973	0.11887	0.070132	-0.025296	-0.04874	-0.058545	0.241205	0.565756	0.479798	0.285652	0.183603	0.057782	1	
o_bpFaced	0.470006	-0.002194	0.043307	-0.131663	-0.109612	-0.058893	0.142775	-0.031534	0.204599	0.451784	-0.338689	0.054874	0.004959	0.117939	0.129096	0.215975	0.142307	0.079021	-0.04874	-0.123942	-0.201533	0.288579	0.508602	0.413803	0.049316	0.015752	0.051413	0.853698	1

By creating a correlation table of all numeric variables (screenshot above), we identified five independent variables that should be removed because of collinearity. We used a 0.75 cutoff in our determinations and kept the variable most correlated with the dependent variable. The variables with collinearity can be seen in the table below.

Variable 1	Variable 2	Correlation	Variable Kept
p_SvGms	o_SvGms	0.98	p_SvGms

p_svpt	p_1stIn	0.86	p_svpt
o_svpt	o_1stIn	0.86	o_svpt
p_bpSaved	p_bpFaced	0.85	p_bpFaced
o_bpSaved	o_bpFaced	0.85	o_bpFaced

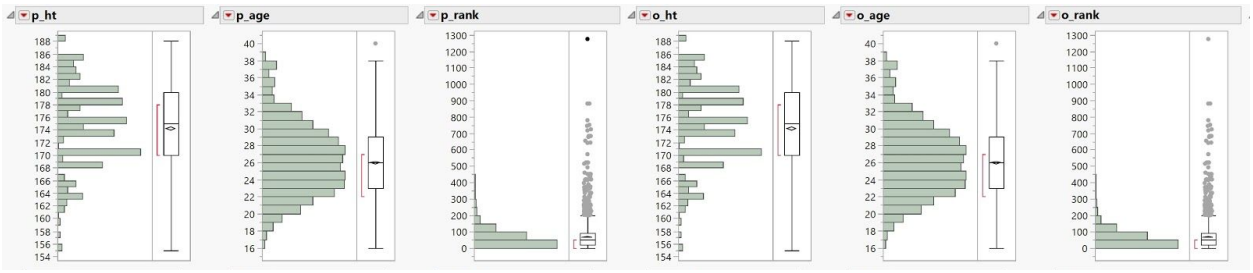
Some of the collinear relationships were a little surprising to us at first. For example, we thought break points saved (pbSaved) would be a better indication of whether the player would win/lose a match than the total number of break points faced. However, as we reflected on the data and what the relationships meant, we determined that actually made sense because a player was more likely to win the match if she faced *less* break points regardless of how many she won.

As can be seen from the screenshot below, the match statistics (for both the player and opponent) have the largest correlation with the match outcome. Our team originally thought player rank, player height, and surface could be strong predictors of match outcome, but as can be seen from the table below, these variables have a relatively small effect on our outcome variable. Instead, the player's and opponent's score in the first round has the highest correlation with the match outcome followed by the number of breakpoints faced (by both the player and opponent). Other significant variables include 1stWon, which refers to the number of points won off a first serve, and aces. All of these variables translate into actual points in the match, so it makes sense that these variables would be highly predictive in match outcome.

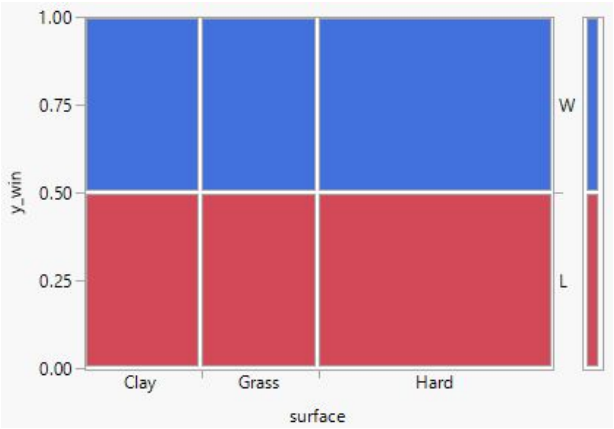
Variable	*Correlation to y_win*
y_win	1.000000
p_set_1	0.570439
o_bpFaced	0.470006
p_1stWon	0.313236
p_ace	0.221967
p_2ndWon	0.205730
o_rank	0.197455
o_df	0.155577
p_age	0.086508
p_ht	0.064582
o_svpt	0.058337
p_SvGms	0.024994
round	0.000000
minutes	0.000000
p_svpt	-0.058337
o_ht	-0.064582
o_age	-0.086508
p_df	-0.155577
p_rank	-0.197455
o_2ndWon	-0.205730
o_ace	-0.221967
o_1stWon	-0.313236
p_bpFaced	-0.470006
o_set_1	-0.570439

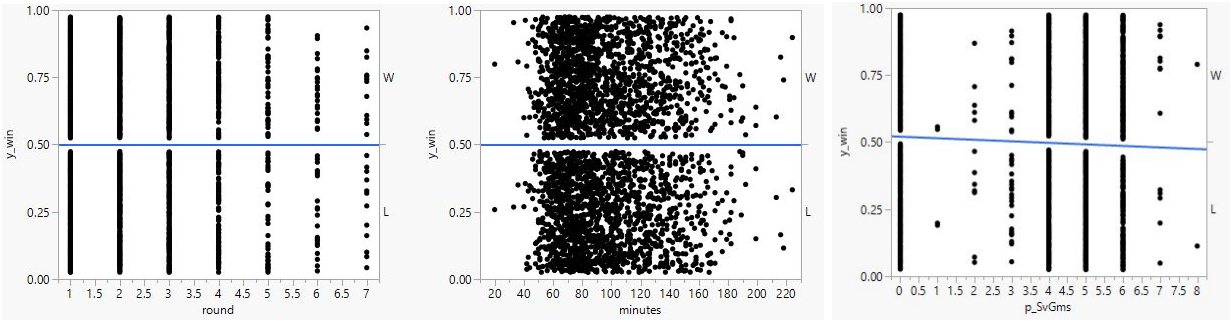
Once we loaded our data in JMP, we performed various univariate and multivariate statistics to evaluate the independent variables individually as well as assess their relationship with the dependent variable.

We evaluated histograms and box plots of the independent variables to get a better understanding of the data values. We've included some of of the visualizations below. As can be seen, most of the independent variables resembled a normal distribution; however, both player and opponent rank are exceptions to that. It makes sense for these variables to be negatively skewed since most of the tennis players at the Majors are the best in the world. Because we would be creating a classification model, we chose not to standardize those variables.

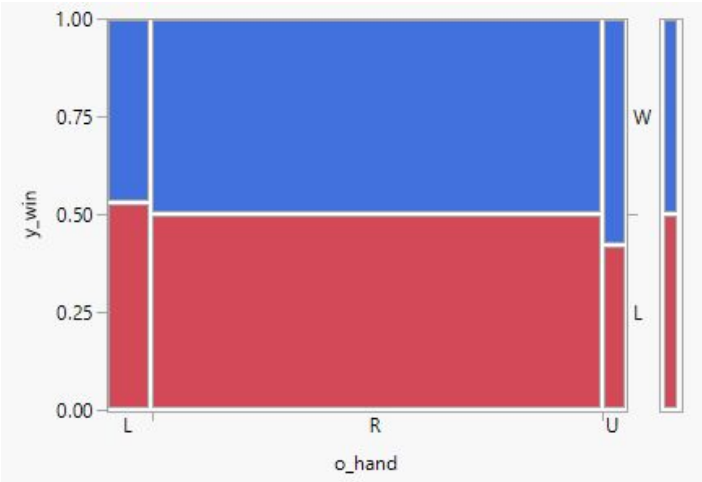


By performing a "Fit Y by X" test, we were able to evaluate the independent variables' relationship with the dependent variable. We saw that tourney_name, surface, round, minutes, and p_SvGms would not be significant predictors of the match outcome, so we chose to remove these variables. The screenshots below support these conclusions.

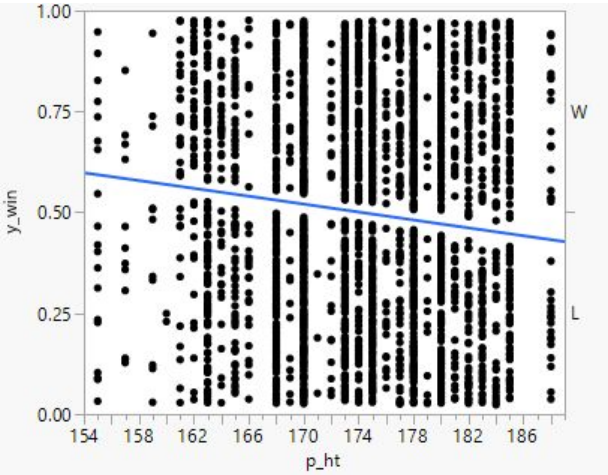


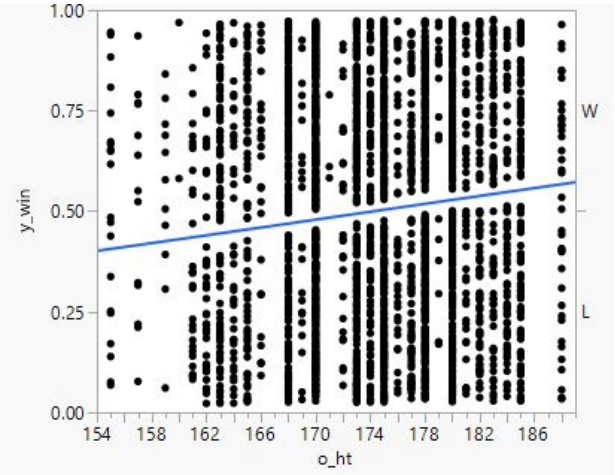


We also saw some interesting trends. For example, while opponent hand is not a very significant factor in determining match outcome, we can see in the screenshot below that the player was more likely to lose when playing against an opponent who was left handed.

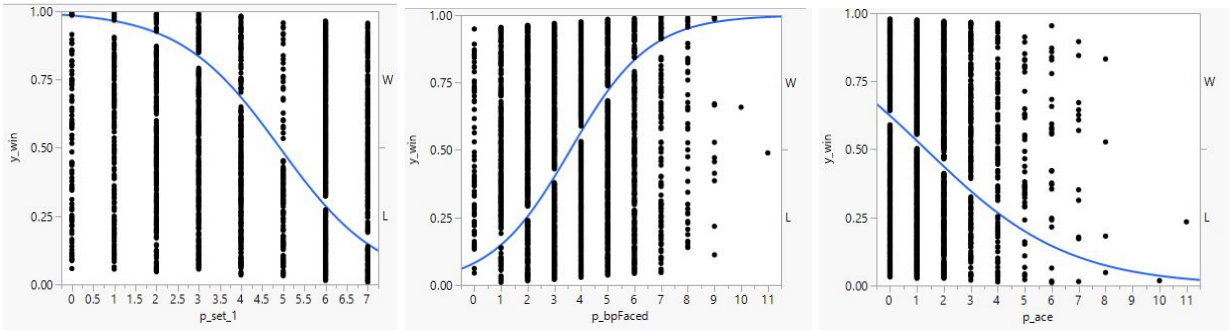


We can also see that taller players were more likely to win. On the flip side, the opponent was more likely to win if she was tall.





In support of the correlation matrix, the set statistics reveal strong relationships with match outcome. As we can see in the screenshots below, the more games won by the player in the first round along with fewer break points faced and more aces translates into a higher likelihood of the player winning the match. The opposite is true for the opponent variables.



Since our model will generate a categorical variable, we were not as concerned about non-linear relationships.

Overall, by visualizing the data, we understand the story it's trying to tell better. We were also able to reduce the number of variables to 24. This will be effective in understanding and generating more accurate models, which we'll dive into next.

Input Variable Evaluation and Model Testing

This section overviews the measures of model quality we evaluated as well as the results of our model testing.

Since we're predicting a categorical outcome of whether the player will win or lose a match, model quality will be assessed through weighing the tradeoffs in recall, precision, f-measure, specificity, accuracy, and error. Based on our domain knowledge, we've ranked the outcomes as follows:

TP	TN	FP	FN
Best	Good	Worst	Bad

Correctly classifying a winning outcome (TP) is better, more insightful, and often more lucrative than correctly classifying a losing outcome (TN). This also means that incorrectly predicting a player would win the match when she loses (TP) is worse than predicting a player would lose the match when she actually wins (FN). Because of our rankings, precision will be one of the most important indicators of model quality as well as error in general.

Using a 60-40 split and a seed of 1 for the validation column, we used the Logistic Regression algorithm to run an initial test on variable significance. The output of that test can be seen in the screenshot below. According to the results, 11 variables fall below the 0.05 p-value cutoff; in other words, 11 variables are statistically significant. As we calculated from the counts in the validation data's confusion matrix, the error was 6.5 percent.

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
p_ioc	46	45	70.5847898	0.0088*
p_hand	2	2	1.80753282	0.4050
o_ioc	46	44	73.5582321	0.0034*
o_hand	2	2	0.4972931	0.7799
p_ht	1	1	6.45531175	0.0111*
p_age	1	1	1.21451809	0.2704
p_rank	1	1	0.62491645	0.4292
o_ht	1	1	9.97446267	0.0016*
o_age	1	1	0.97815874	0.3227
o_rank	1	1	1.72492874	0.1891
p_set_1	1	1	1.20598741	0.2721
o_set_1	1	1	1.76497125	0.1840
p_ace	1	1	1.99293582	0.1580
p_df	1	1	1.27602177	0.2586
p_svpt	1	1	180.34518	<.0001*
p_1stWon	1	1	441.198857	<.0001*
p_2ndWon	1	1	367.635586	<.0001*
p_bpFaced	1	1	1.39946545	0.2368
o_ace	1	1	1.8439328	0.1745
o_df	1	1	1.37284436	0.2413
o_svpt	1	1	160.273752	<.0001*
o_1stWon	1	1	455.651794	<.0001*
o_2ndWon	1	1	337.173427	<.0001*
o_bpFaced	1	1	13.5801602	0.0002*

Confusion Matrix

Training

Actual y_win	Predicted Count W	L
W	890	16
L	13	909

Validation

Actual y_win	Predicted Count W	L
W	574	43
L	36	565

We proceeded by taking out the significant variables with the lowest contribution on y_win (measured by LogWorth) to see if we could produce a better dataset. Our test results can be seen in the spreadsheet below.

		Best	Best	actual = W			actual = L									
Algorithm	Variables Kept	Split	K	TP	FN	Total	TN	FP	Total	Total	Recall	Specifity	Precision	F-meas	%Error	AUC
LogReg	All variables	N/A	N/A	574	43	617	565	36	601	1218	93.0%	94.0%	94.1%	93.6%	6.5%	0.979
LogReg	All significant variables (based on first run)... p_ic	N/A	N/A	576	41	617	565	36	601	1218	93.4%	94.0%	94.1%	93.7%	6.3%	0.981
LogReg	All significant minus o_ioc	N/A	N/A	580	37	617	568	33	601	1218	94.0%	94.5%	94.6%	94.3%	5.7%	0.987
LogReg	All significant minus o_ioc and p_ioc	N/A	N/A	587	30	617	573	28	601	1218	95.1%	95.3%	95.4%	95.3%	4.8%	0.992
LogReg	All significant minus iocs and o_bpFaced	N/A	N/A	586	31	617	574	27	601	1218	95.0%	95.5%	95.6%	95.3%	4.8%	0.992
LogReg	All significant minus iocs, o_bpFaced, and p_ht	N/A	N/A	589	28	617	573	28	601	1218	95.5%	95.3%	95.5%	95.5%	4.6%	0.992
LogReg	All significant minus iocs, o_bpFaced, p_ht, o_ht	N/A	N/A	588	29	617	574	27	601	1218	95.3%	95.5%	95.6%	95.5%	4.6%	0.993
LogReg	Last model plus p_set_1 and o_set_1	N/A	N/A	589	28	617	574	27	601	1218	95.5%	95.5%	95.6%	95.5%	4.5%	0.993
LogReg	Last model minus p_svpt and o_svpt	N/A	N/A	518	99	617	510	91	601	1218	84.0%	84.9%	85.1%	84.5%	15.6%	0.928
LogReg	Last model plus p_bpFaced and o_bpFaced	N/A	N/A	572	45	617	551	50	601	1218	92.7%	91.7%	92.0%	92.3%	7.8%	0.982
LogReg	Last model minus p_set_1 and o_set_1	N/A	N/A	569	48	617	549	52	601	1218	92.2%	91.3%	91.6%	91.9%	8.2%	0.982
CART	All variables	9	N/A	477	140	617	542	59	601	1218	77.3%	90.2%	89.0%	82.7%	16.3%	0.909
CART	All significant variables from LogReg model (row	19	N/A	498	119	617	459	142	601	1218	80.7%	76.4%	77.8%	79.2%	21.4%	0.854
KNN	All variables	N/A	9	523	94	617	509	92	601	1218	84.8%	84.7%	85.0%	84.9%	15.3%	--
KNN	All variables from best LogReg model (row 11)	N/A	8	545	72	617	541	60	601	1218	88.3%	90.0%	90.1%	89.2%	10.8%	--

Overall, we found a combination of six variables (p_svpt, p_1stWon, p_2ndWon, o_svpt, o_1stWon, and o_2ndWon) that produced a more accurate model than the original test and reduced error to 4.6 percent. See the screenshots below for more details. As can be seen in the spreadsheet above, two other models produced comparable results but had more variables so we chose to go with the simpler model.

Source	LogWorth	PValue
p_1stWon	208.742	0.00000
o_1stWon	208.694	0.00000
p_svpt	125.658	0.00000
p_2ndWon	123.141	0.00000
o_svpt	118.993	0.00000
o_2ndWon	115.510	0.00000

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	0.6090535	0.7383679	0.68	0.4094
p_svpt	-1.0700779	0.0883724	146.62	<.0001*
p_1stWon	2.25948491	0.1744054	167.84	<.0001*
p_2ndWon	2.18866445	0.1811769	145.93	<.0001*
o_svpt	1.03573959	0.0857506	145.89	<.0001*
o_1stWon	-2.2595539	0.1746457	167.39	<.0001*
o_2ndWon	-2.0955818	0.1745241	144.18	<.0001*

For log odds of W/L

Covariance of Estimates

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
p_svpt	1	1	571.87404	<.0001*
p_1stWon	1	1	953.976089	<.0001*
p_2ndWon	1	1	560.299608	<.0001*
o_svpt	1	1	541.236155	<.0001*
o_1stWon	1	1	953.755244	<.0001*
o_2ndWon	1	1	525.225023	<.0001*

Confusion Matrix

Training

Actual	Predicted Count	
y_win	W	L
W	874	32
L	31	891

Validation

Actual	Predicted Count	
y_win	W	L
W	588	29
L	27	574

We used this dataset as well as the 11 significant variables and the 24 cleaned variables datasets when testing the various algorithms.

Since we tested several Logistic Regression models when determining what variables to use in our dataset, we set out to test the other categorical prediction algorithms to see if we could get better results. We ran several iterations of CART, KNN, ANN, Boosted Tree, Decision Forest, and SVM. A portion of our results can be seen in the spreadsheet below; for a more complete view, please reference the link in the footnote.[2] In total we ran 69 models.

[2] https://docs.google.com/spreadsheets/d/1-H0YsC7x0baA3caZar7nK_1d4C-0i_Q-0En7l0JcytA/edit?usp=sharing

				ANN Measures																
				Layer 1		Layer 2		Actual = W			Actual = N									
	Model#	Description	Best K	TanH	Linear	TanH	Linear	TP	FN	Tot	FP	TN	Tot	Total	Precision	Recall	F-meas	Specificity	%Error	%Accur	AUC
	1	All variables	N/A					477	140	617	59	542	601	1218	89.0%	77.3%	82.7%	90.2%	16.3%	83.7%	0.9085
	2	Significant variables	N/A					498	119	617	142	459	601	1218	77.8%	80.7%	79.2%	76.4%	21.4%	78.6%	0.8535
CART	3	Top-6 variables	N/A					489	128	617	136	465	601	1218	78.2%	79.3%	78.7%	77.4%	21.7%	78.3%	0.8607
	4	All variables	10					522	95	617	87	514	601	1218	85.7%	84.6%	85.2%	85.5%	14.9%	85.1%	
	5	Significant variables	9					524	93	617	80	521	601	1218	86.8%	84.9%	85.8%	86.7%	14.2%	85.8%	N/A
KNN	6	Top-6 variables	9					545	72	617	62	539	601	1218	89.8%	88.3%	89.1%	89.7%	11.0%	89.0%	
	7	All variables; # of tours: 20	N/A	3	1			581	36	617	30	571	601	1218	95.1%	94.2%	94.6%	95.0%	5.4%	94.6%	0.9891
	8	All variables; # of tours: 50	N/A	3	1			585	32	617	32	569	601	1218	94.8%	94.8%	94.8%	94.7%	5.3%	94.7%	0.9896
	9	Significant variables; # of tours: 20	N/A	3	1			588	29	617	31	570	601	1218	95.0%	95.3%	95.1%	94.8%	4.9%	95.1%	0.9896
	10	Significant variables; # of tours: 50	N/A	3	1			586	31	617	25	576	601	1218	95.9%	95.0%	95.4%	95.8%	4.6%	95.4%	0.9906
	11	Top-6 variables; # of tours: 20	N/A	3	1			591	26	617	30	571	601	1218	95.2%	95.8%	95.5%	95.0%	4.6%	95.4%	0.9926
ANN-1	12	Top-6 variables; # of tours: 50	N/A	3	1			591	26	617	29	572	601	1218	95.3%	95.8%	95.6%	95.2%	4.5%	95.5%	0.9927
	13	All variables; # of tours: 20	N/A	3	2			581	36	617	30	571	601	1218	95.1%	94.2%	94.6%	95.0%	5.4%	94.6%	0.9886
	14	All variables; # of tours: 50	N/A	3	2			582	35	617	31	570	601	1218	94.9%	94.3%	94.6%	94.8%	5.4%	94.6%	0.9889
	15	Significant variables; # of tours: 20	N/A	3	2			589	28	617	31	570	601	1218	95.0%	95.5%	95.2%	94.8%	4.8%	95.2%	0.9909
	16	Significant variables; # of tours: 50	N/A	3	2			589	28	617	32	569	601	1218	94.8%	95.5%	95.2%	94.7%	4.9%	95.1%	0.9905
	17	Top-6 variables; # of tours: 20	N/A	3	2			590	27	617	26	575	601	1218	95.8%	95.6%	95.7%	95.7%	4.4%	95.6%	0.9928
ANN-2	18	Top-6 variables; # of tours: 50	N/A	3	2			592	25	617	28	573	601	1218	95.5%	95.9%	95.7%	95.3%	4.4%	95.6%	0.9928
	19	All variables; # of tours: 20	N/A	3	3			584	33	617	34	567	601	1218	94.5%	94.7%	94.6%	94.3%	5.5%	94.5%	0.9892
	20	All variables; # of tours: 50	N/A	3	3			584	33	617	30	571	601	1218	95.1%	94.7%	94.9%	95.0%	5.2%	94.8%	0.9891
	21	Significant variables; # of tours: 20	N/A	3	3			588	29	617	28	573	601	1218	95.5%	95.3%	95.4%	95.3%	4.7%	95.3%	0.9905
	22	Significant variables; # of tours: 50	N/A	3	3			587	30	617	24	577	601	1218	96.1%	95.1%	95.6%	96.0%	4.4%	95.6%	0.9904
	23	Top-6 variables; # of tours: 20	N/A	3	3			592	25	617	29	572	601	1218	95.3%	95.9%	95.6%	95.2%	4.4%	95.6%	0.9927
ANN-3	24	Top-6 variables; # of tours: 50	N/A	3	3			593	24	617	28	573	601	1218	95.5%	96.1%	95.8%	95.3%	4.3%	95.7%	0.9928
	25	All variables; # of tours: 20	N/A	3	1	1	1	583	34	617	31	570	601	1218	95.0%	94.5%	94.7%	94.8%	5.3%	94.7%	0.9890
	26	All variables; # of tours: 50	N/A	3	1	1	1	584	33	617	33	568	601	1218	94.7%	94.7%	94.7%	94.5%	5.4%	94.6%	0.9892

One of the decision forest models we ran had the best recall; however, the tradeoff in precision was not worth the benefit. Ultimately, the best models in relation to precision and error were from the ANN iterations. Using the 11 significant variables, 50 tours, and a single layer comprised of 3 TanH nodes and 3 Linear nodes, we generated a model with 96.1 percent precision and 4.4 percent error. We had two other models that were comparable: one with the top-6 variables, 50 tours, and a single layer comprised of 3 TanH nodes and 3 Linear nodes, which generated 95.5 percent precision and 4.3 percent error and another with the top-6 variables, 20 tours, and two layers comprised of 3 TanH and 1 Linear and 1 TanH and 1 Linear in the respective layers, which returned the same results. Because the results were quite similar, we evaluated simplicity. Using the 6-variable dataset and running only 20 tours were both ideal from an execution standpoint, so we selected that model. The screenshot below highlights our selection; however, it can be seen in greater resolution from the link in Footnote 2.

29	Top-6 variables; # of tours: 20	N/A	3	1	1	1	593	24	617	28	573	601	1218	95.5%	96.1%	95.8%	95.3%	4.3%	95.7%	0.9926
*	Top-6 variables; # of tours: 20; AML; 0.5 cutoff	N/A					535	84	619	4	595	599	1218	99.3%	86.4%	92.4%	99.3%	7.2%	92.8%	0.9940

Since the ANN algorithm typically overfits the validation data, we created a new validation column with a test set (50-30-20 split with a seed of 1) and ran the model again. The model did not perform quite as well, but this can also be attributed to the variation in samples rather than just overfitting. By checking five additional test sets (all with different seeds), we learned there can be about 3 percent variability in error rate due to random sampling. Because of this variety, and the fact that ANN models are quite accurate in learning over time, we are confident in our model selection.

Because ANN models cannot be directly interpreted, we will explain the logistic regression model to gain greater understanding about the contribution of the top six variables.

Parameter Estimates

Term	Estimate	Std Error	ChiSquare	Prob>ChiSq
Intercept	0.6090535	0.7383679	0.68	0.4094
p_svpt	-1.0700779	0.0883724	146.62	<.0001*
p_1stWon	2.25948491	0.1744054	167.84	<.0001*
p_2ndWon	2.18866445	0.1811769	145.93	<.0001*
o_svpt	1.03573959	0.0857506	145.89	<.0001*
o_1stWon	-2.2595539	0.1746457	167.39	<.0001*
o_2ndWon	-2.0955818	0.1745241	144.18	<.0001*

For log odds of W/L

Covariance of Estimates

Effect Likelihood Ratio Tests

Source	Nparm	DF	L-R ChiSquare	Prob>ChiSq
p_svpt	1	1	571.87404	<.0001*
p_1stWon	1	1	953.976089	<.0001*
p_2ndWon	1	1	560.299608	<.0001*
o_svpt	1	1	541.236155	<.0001*
o_1stWon	1	1	953.755244	<.0001*
o_2ndWon	1	1	525.225023	<.0001*

Confusion Matrix

Using the logistic regression output above, we can create the following prediction formula:

```
LogOdds(y) = 0.61 - 1.07*p_svpt + 2.26*p_1stWon +
2.19*p_2ndWon + 1.04*o_svpt - 2.26*o_1stWon - 2.10*o_2ndWon
```

The output is then a LogOdds, which can be converted to probabilities and ultimately a category based on a cutoff value. The model coefficients give insight into the variable contributions. Namely, as the player's 1stWons increase by one unit, the LogOdds of winning increase by 2.26; in other words, the player is 9 times more likely to win the match. The opponent's 1stWons will decrease the probability of winning by the same amount. The same conclusions can be drawn from the other variables' coefficients.

The coefficients resemble the values in the LogWorth column below; however, coefficients are rarely completely accurate in defining a variable's contribution on the dependent variable. The LogWorths were also valuable in narrowing our dataset down to the six most important variables because it showed that 5 of the 11 significant variables--while significant--did not have a huge effect on match outcome overall. For example, the player and opponent country (ioc) variables had a LogWorth of 1-2 while the LogWorth values of p_1stWon and the other five variables were up in the hundreds. Ultimately, when we pulled out those small-contributing variables, the model did better overall; and when we tried taking out one of the main six variables, model performance dropped substantially.

Source	LogWorth	PValue
p_1stWon	208.742	0.00000
o_1stWon	208.694	0.00000
p_svpt	125.658	0.00000
p_2ndWon	123.141	0.00000
o_svpt	118.993	0.00000
o_2ndWon	115.510	0.00000

Because we had already filtered the data on female tennis players and the Grand slam tournaments, we did not expect to find any subpopulations. This, along with our high model performance, supported investing time in other areas besides cluster analysis.

Web-Service-Based Excel Estimator

This section compares the results of our overall project model with the model we setup in Azure Machine Learning Studio to create a Web Service.

Because Azure handles most of the inner node workings of the ANN model, we could not completely replicate our results. However, we still chose to go with the ANN model. We ran a few iterations of the model to get our results as close as possible to our original model selection. By adjusting the cutoff from 0.50 to 0.23, we were able to get very similar results. Our Azure test results can be seen in the same spreadsheet as Footnote 2.

Future Research

This section highlights improvements that can be made to our project in the future.

In the future, we'd like to improve our model by investigating more detailed information about individual tennis sets and even games. This information would give additional insight into the player and opponent interaction including how long the rally lasted, the speed of the serve, net interaction, number of forced errors, etc. This information would most likely enable us to generate a match prediction prior to the end of set 1. It would be interesting to compare the accuracy of our existing model with the point-by-point data since the best players in the world only win about 55 percent of match points.

In conclusion, because of the processes we followed in cleaning, understanding, and evaluating our dataset, we are confident in our prediction model.

Appendix

1. Description of all original (relabeled) variables

Field	Description
y_win	If the player won the match (1 = won, 0 = lost)
tourney_name	Tournament name (Australian Open, French Open, Wimbledon, US Open)
surface	The surface that the match is played on
tourney_date	The date of the Tournament
player_id	Unique identifier of the player
player_name	The name of the player
player_hand	The player's dominant playing hand
player_ht	The height of the player in centimeters
player_ioc	The country code of the player as defined by the International Olympic Committee (IOC)
player_age	The age of the player
player_rank	The ranking of the player as it pertains to their overall world ranking
opponent_id	Unique identifier of the opponent
opponent_name	The name of the opponent
opponent_hand	The opponent's dominant playing hand
opponent_ht	The height of the opponent in centimeters
opponent_ioc	The country code of the opponent as defined by the International Olympic Committee (IOC)
opponent_age	The age of the opponent
opponent_rank	The ranking of the opponent as it pertains to their overall world ranking
round	The round of the tournament
minutes	The length of the match in minutes

p_set_1	The player's score for the first set
p_ace	The number of aces the player had
p_df	The number of double faults the player had
p_svpt	The number of first serves the player had during the match
p_1stIn	The number of first serves the player had that went into the service box
p_1stWon	The number of points scored by the player off of a first serve during the match
p_2ndWon	The number of points scored by the player off of a second serve during the match
p_SvGms	The number of games served by the player during the match
p_bpSaved	The number of breakpoints that were saved by the player during the match
p_bpFaced	The number of breakpoints the player faced during the match
o_set_1	The opponent's score for the first set
o_ace	The number of aces the opponent had
o_df	The number of double faults the opponent had
o_svpt	The number of first serves the opponent had during the match
o_1stIn	The number of first serves the opponent had that went into the service box
o_1stWon	The number of points scored by the opponent off of a first serve during the match
o_2ndWon	The number of points scored by the opponent off of a second serve during the match
o_SvGms	The number of games served by the opponent during the match
o_bpSaved	The number of breakpoints that were saved by the opponent during the match
o_bpFaced	The number of breakpoints the opponent faced during the match