EAS 499 Final Thesis

A Thesis
Presented to
Department of Computer and Cognitive Science
University Of Pennsylvania

Ali Kozlu

Dec 2018

Approved for the
Bachelor of Science in Computer and Cognitive Science

____
Professor Marcus Mitchell

____

____

____
, DUS

Acknowledgements

Firsly, I want to thank my advisor, Professor Marcus Mitchell, for his valuable input throughout the project. I also want to thank Mark, my partner for this project, for his valuable ideas on how to utilize the our tennis match predictor model on betting exchange markets. As always, a special thanks to my family. This thesis would not be possible without them.

Table of Contents

List of Tables

List of Figures

Abstract

The rise of online sports betting websites has stimulated a dramatic increase in tennis-related financial activity, as tennis features amongst the most popular sports internationally. This has resulted in extensive research being conducted on tennis match models to predict the outcome. Initial approaches used hierarchical Markov models to compute the respective probabilities of each player winning the match. Although mathematically sound, these stochastic approaches ignore a variety of match statistics that contribute to the outcome. More recently, researchers have focused on supervised machine learning approaches that use an immense amount of diverse historical tennis match and player performance data. The present thesis first explains how these two major approaches attempt to create a tennis match predictor model. It then proposes a novel approach for tennis modeling by introducing stacked generalization, an ensemble learning technique used to combine information from multiple predictive models to generate a new model, a so-called super learner. A historical dataset of 101295 ATP tennis matches played between 2004-2014 is used to train the meta-model. The final model is then evaluated on a test set of 23918 ATP matches played in the years 2014-2018. The model is then evaluated on return on investment in high volume pre-match betting. We argue that state of the art tennis models cannot yet make a profit from the pre-match market and. We then propose a novel framework of using model class probabilities to make a profit from betting exchange platforms by predicting in-play market movements.

Introduction

In recent years, there has been a noticeable rise in sports betting and gambling promotion during sporting events. The two main drivers of growth behind this rise are the increased amount of online sports betting platforms and the introduction of "in play betting", where consumers can place a bet on a range of possible outcomes both after the game has commenced and, as they occur during a game. The rise of the online sports betting volume has stimulated a dramatic increase in sports-related financial activity. Large monetary amounts involved in betting became a primary motivator for research being conducted on prediction models and applications of statistical analysis to the domain of sport. As one of the most popular sports globally, tennis has become an ideal candidate for modeling and predicting the outcome of professional matches. In addition, professional singles tennis is an attractive sport to model algorithmically for a number of reasons. In tennis, each match is played between just two players, as opposed to the multitude of players involved in a team-based game such as football, rugby or basketball. Because of this, there is no need to analyze the offensive and defensive strengths of possible combinations of starting lineups. Also, the large amount of historical data freely available makes the sport of Tennis an ideal candidate for Machine learning (ML) algorithms, which have shown promising results in the domains of classification and prediction. The goal of the present thesis is to investigate the applicability of combining multiple of machine learning models for the prediction of professional tennis matches. In doing so, we explore appropriate means of feature engineering from data sources, model evaluation, and specific challenges of predicting tennis results.

Chapter 1

Background

1.1 Tennis Datasets

As mentioned in the introduction, there is a large amount of historical tennis data freely available. Tennis websites such as atpworldtour.com and tennis.data.co.uk provide the most comprehensive access to information about players, the outcomes of matches and statistics of certain matches. Other important sources that include historical datasets (as CSV) of ATP Tennis Rankings, Results, and Stats are provided by Jeff Sackman and Daniel Korzekwa. A more complex dataset with a longer historical timespan and advanced metrics is provided by the OnCourt System. The present thesis uses the OnCourt Dataset, which includes statistics of 17625 ATP tennis matches played between 2004-2018 and has results for over 500 thousand ATP matches played since the 1990s. The Oncourt dataset also contains further information such as tournament seeding, tournament round, player age and length of the match. OnCourt Dataset, which is in MDB format, is converted into a Sqlite3 database using Mdb Tools. A Python script then extracts different information from Sqlite3 Database (tournaments, player names, statistics) and converts the data into Pandas data frame.[1] The full list of statistics found in the OnCourt dataset is omitted, as our model uses a portion of the information provided. The relevant statistics provided for each match can be seen in the figure below.

[1]Source Code can be found at the author's Github

Table 1.1: Per Match Statistics found in OnCourt Dataset

OnCourt Match Statistics	
First Serve percentage	FS
Percentage of points won on first serve	WSP_1
Percentage of points won on second serve	WSP_2
Percentage of receiving points won	WRP
Double Faults	DF
Aces	ACES
Total Points Won	TPW
Break Points Won	BP
Total Break Points	TBP
Net Approaches	NA
Total Net Approaches	TNA

1.2 Tennis Betting

There are two main categories of tennis betting: pre-play and in-play. The former is considered as a traditional form of betting, in which a bookmaker offers odds and accepting punters bet directly against the bookmakers. The second category can be played in betting exchanges such as Bet Fair, in which customers instead offer odds and place bets against each other. A unique option brought by exchanges, called lay betting, allows punters to oppose a selection, i.e. to bet against something happening. In this case, punters can play the role of a traditional bookmaker but offering odds to sell a bet instead of the usual odds to back a bet. In both categories, it is possible to bet on a variety of factors, such as such as the winner of the match, the number total sets and the over/under of the total number of games. Traditionally, research has focused on bets placed on the overall outcome prior to the match starting as it allows a more comprehensive evaluation of the performance of the generated model against the market. However, experts claim that over 80% of the overall money wagered on tennis matches is bet in-play, i.e., during the course of the match. Thus, instead of only focusing on pre-play, the present thesis attempts to use the model for both pre-play and in-play betting to see the true value of our model Webb (2011).

1.3 Review of Previous Work

The earliest attempts to the statistical analysis of tennis matches can be traced back to early 2000s. S. Clarke & Dyte (2000) and Boulier & Stekler (1999) used a single feature, namely ATP computer tennis rankings, to predict a player's chance of winning. The goal was to see whether the position of a tennis player in world rankings was related to his/her performance in an upcoming tournament. These early attempts fit a logistic regression model to the ATP ratings, to estimate the probability of winning as a function of the difference in rating points. Del Corral & Prieto-Rodriguez (2010) attempt to assess the degree to which the difference in ranking points is good indicators of the outcome of Grand Slam matches. More recently, Spanias & Knottenbelt (2013) claimed that the ATP ranking system is inaccurate at ranking players with rankings greater than 32, showing that ATP rankings perform much more poorly in this subset of matches with a success rate as low as 55%. Another problem is that there is only one ranking for the four surfaces, thus ranking system does not capture player strengths on different surfaces. More recent state-of-the-art tennis prediction models can be seen as a hierarchical/point model using Markov chains. These models take advantage of the fact that scoring system in tennis has a hierarchical structure, with a match being composed of sets, which are composed of games, which in turn are composed of individual points. Thus one can construct a hierarchical Markov model to estimate the match-winning probabilities of both players by using *only the probabilities of the two players winning points while serving.* Using this singular statistic, one can deduce the probability of a player winning a single point, then a game, then a set, and finally the match, since the Markov chain will reflect the stochastic progression of the score in a game, as in Figure 1.1. Hertzmann & Zorin (2001) Newton & Keller (2005) James (2008) used Markov chains to generate equivalent hierarchical expressions using the assumption that points are independent and identically distributed. Klaassen & Magnus (2003) later show that winning the previous point can have a positive influence on winning the current point, but still, argue that the model is a good approximation since the deviations from i.i.d. are small. Arguing that the small deviations can be tolerated in the domain of forecasting, Barnett & Clarke (2005), Spanias & Knottenbelt (2013) and Newton & Keller (2005) focus in detail on the estimation of the respective probability of each player winning a point while serving. In order to calculate this probability, Barnett & Clarke (2005) attempt to mathematically combine the serving ability of the serving player against the returning ability of the returning player. In order to come up with these two numbers, Barnett & Clarke (2005)'s approach is to can look at historical service statistics of both players and average each player's statistics. As explained in Spanias & Knottenbelt (2013), Barnett's approach computes two statistics for each player f_i *and* g_i reflecting their serving and receiving strengths against an average opponent respectively, where f_i and g_i are computed as:[2]

$$f_i = a_i b_i + (1 - a_i)\ c_i \tag{1.1}$$

$$g_i = a_{av} d_i + (1 - a_v)\ e_i \tag{1.2}$$

In which,

$$f_i = \text{proportion of points won on serve by player i}$$

$$g_i = \text{proportion of points won on return by player i}$$

[2]The following mathematical calculations are taken from Spanias & Knottenbelt (2013). Derivation of the formulas can be found at Barnett & Clarke (2005)

$$a_i = \text{probability of successful first serve by player i}$$

$$a_{av} = \text{average probability of successful first serve (across all players)}$$

$$b_i = \text{proportion of own successful first serves won by player i}$$

$$c_i = \text{proportion of own second serves won by player i}$$

$$d_i = \text{proportion of first serves of opponent won on return by player i}$$

$$e_i = \text{proportion of second serves of opponent won on return by player i}$$

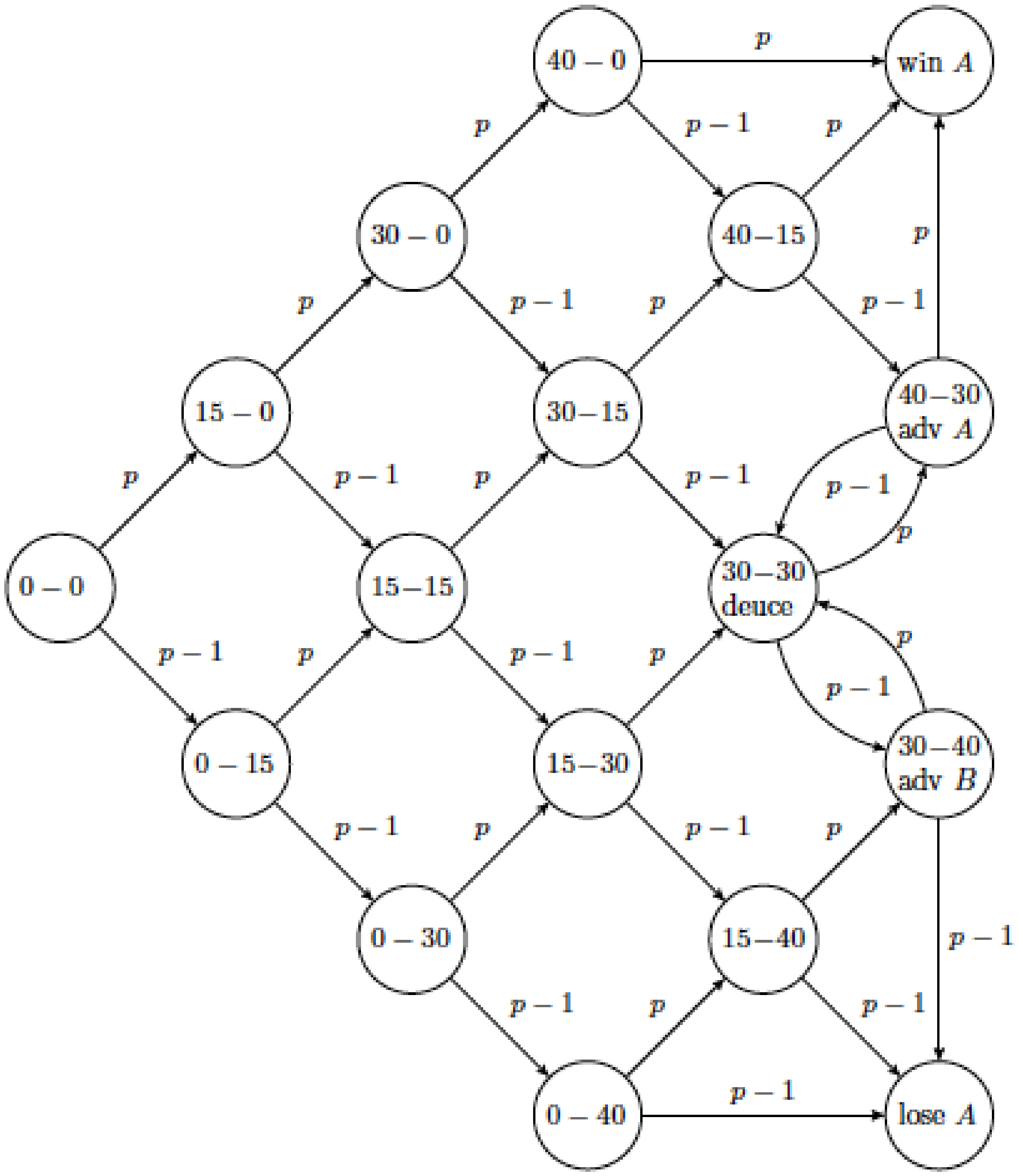


Figure 1.1: Markov Chain model of a Tennis Game. Taken from Sipko and Knottenbelt

Although mathematically sound, one can argue that this approach is far from perfect empirically. Even if is a good approximation; Barnett's algorithm suffers from bias because players face different opponents of varying skill levels. Knottenbelt's Common Opponent Model attempts to solve this flaw with Barnett's algorithm. He adopts the way the serve-winning probabilities of players are calculated before being supplied to the Barnett formulas by only looking at the players' performance against *common opponents*. The crucial point is that this version provides a fair basis of comparison between players by analyzing match statistics for opponents that both players have encountered in the past. Knottenbelt claims that his common opponent hierarchical Markov model yields 3.8% ROI against the best odds offered by bookmakers for a data set of 2178 diverse tennis matches, but admits that the sample size is big enough to represent the average performance of the model.

Perhaps more importantly, the hierarchical Markov models do not take other important match statistics such as the number of aces, double faults, total points won and second serve winning percentage, into account. Hence it is unrealistic to expect that these models can successfully capture important factors such as playing styles of players, fatigue and individual player strengths. Considering the availability of an immense amount of historical tennis data, recently several master theses have proposed machine learning models as an alternative to stochastic models described above. Somboonphokkaphan, Phimoltares, & Lursinsap (2009) train neural networks with basic features, including previous head-to-head match outcomes and first serve percentage etc. The authors claim that the model has 75% accuracy for predicting matches in the 2007 and 2008 Grand Slam tournaments. Chen, Tian, & Zhong (2017) try support vector classification models (SVC) with linear, RBF and polynomial kernels, but the results of the thesis are inconclusive. A more comprehensive paper by Sipko (2015), whose thesis advisor was William Knottenbelt, uses the OnCourt dataset to train a Logistic Regression with interactive features and a Feed-Forward Neural Network Model. Sipko claims the neural network model generated a return on investment of 4.4% when betting on 6315 ATP matches in 2013-2014..[3] Knottenbelt and Sipko use the above mentioned common opponent model to derive not only the probability of winning a point but all the possible variables representing the qualities of two players, as shown in Table 1.1. Sipko's paper is significant not only because of the models it introduces but also because the novel feature engineering methods it uses to extract tennis match features from raw historical data, characterizing the qualities of two players in the most representative way. The following thesis closely follows the novel feature engineering done by Sipko, which is explained in the next chapter.

[3]The C# source code for implementing Sipko's paper can be found at Github

Chapter 2

Feature Engineering

2.1 Feature Extraction

Any tennis prediction model must be able to represent the skill difference between both players. Since our dataset includes two x observations (of the same variable) uniformly sampled in the real number range [0,1], taking the difference or ratio for two values of the same variable becomes natural choices for feature engineering. For example, if we wanted to compute the ACES variable for an upcoming match, we simply do: $aces_{final} = aces_1 - aces_2$. This approach also cuts the dimension of our feature space in half, which is important in machine learning because the amount of data required providing a reliable analysis grows exponentially with the dimensionality of features. The so-called *large p, small n* problem, where p is the number of features and n is the number of samples, tend to be prone to over-fitting. An overfitted model can mistake small fluctuations for an important variance, which can lead to classification errors. One can argue that using two observations of the same statistic can preserve more information and give our model a better chance to find certain patterns, but in practice the difference in statistics of two player's has enough predictive performance.[1] So we will be using the differences in variables as our match features. Any effective tennis prediction model must also consider the most representative set of features to capture all possible qualities of both players. Constructing or selecting features from tennis statistics requires a deep understanding of the sport and the true meaning behind the numbers. Thus, a significant challenge for any tennis prediction model is the transformation match statistics into the most representative match features. In the present work, this transformation is done in two major phases, namely the Feature Selection and Feature Construction phases.

[1]In practice, hierarchical models of Barnett show that the difference between the serve-winning probabilities of the two players is enough to predict outcome

Table 2.1: Difference Transformation for Generating Final Features

Avg. Return % for P1	Avg. Return % for P2	Final Feature
0.58961	0.14544	0.44417
0.66203	0.98726	-0.32523
0.89746	0.58317	0.31429
0.06489	0.11745	-0.05256
0.44753	0.34509	0.10244

2.2 Feature Selection

2.2.1 Computing Advanced Statistical Metrics

The most innovative aspect about Sipko and Knottenbelt's work is how they apply their knowledge of the tennis domain to generate advanced metrics using player's performance statistics found in OnCourt Dataset. Three metrics that Sipko and Knottenbelt introduce that are part of our feature set are:[2]

Overal Winning on Serve Percentage

The idea behind this metric is to combine the winning percentage on first and second serve found in OnCourt dataset to come up with a single metric that quantifies a player's *strength when serving*. WSP is calculated in the following way for player i:

$$WSP_i = W1SP_i \cdot \ FS_i + W2SP_i \cdot \ (1 - FS_i) \tag{2.1}$$

$W1SP$ = *Winning % on first Serve*
$W2SP$ = *Winning % on second Serve*
FS = *First Serve Percentage.*

Completeness

Similar to WSP, Completeness combines a player's serve and return percentages in order to combine their offensive and defensive game. For player i:

$$Complete_i = WSP_i \cdot \ WRP_i \tag{2.2}$$

WSP = *Overal Winning on Serve Percentage*
WRP = *Overal Winning on Receiving Points percentage*

[2]Taken from Sipko & Knottenbelt 3.1

Advantage on Serve

This is the only metric in our dataset that combines information from both players. The idea of Knottenbelt and Sipko is to grasp the difference between the serving player's serve strength and the returning player's return strength, thereby quantifying how much advantage serving player has over the returning player. The authors name this feature SERVEADV (player's advantage on serve). The feature is computed in the following way:

$$SERVEADV_1 = WSP_1 - WRP_2 \tag{2.3}$$

$$SERVEADV_2 = WSP_2 - WRP_1 \tag{2.4}$$

2.2.2 Additional Features

Head to Head

This metric takes the outcomes of matches between two players into account. For any two players in a given match, we look at the past matches between two players and calculate the number of matches both player has won. Afterwards Head-to-Head feature is calculated as follows for players i and j:

$$HijH = \text{Player i wins/Total Matches between i and j} \tag{2.5}$$

$$HjiH = \text{Player j wins/Total Matches between i and j} \tag{2.6}$$

Aces and Percentage of Total Points Won

These metrics quantify the total number of Aces and Percentage of Total Points Won by a player in a match. Since these values are dependent on the number of games in a match, these features will be normalized later.

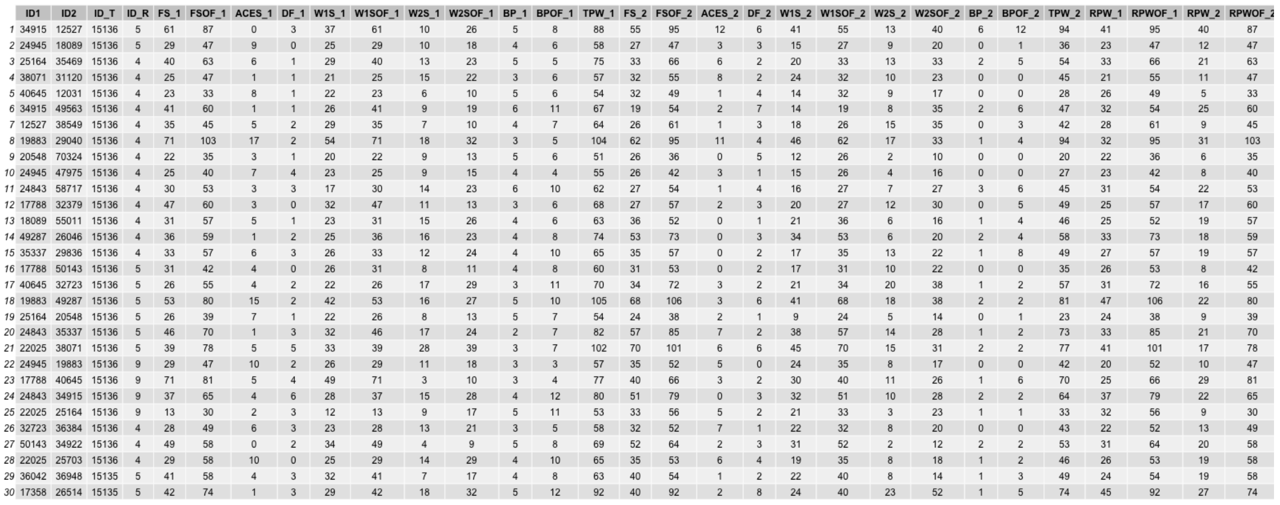
$$TPW_i = \text{Total points won by player i/Total points in a match} \tag{2.7}$$

Percentage of Breaking Points

This feature quantifies the percentage of break points won in that match for both players. If there were no breaking points in the match, that match is dropped from the dataset. Also, breaking point percentages that are not in the range [0,1] are dropped to reduce noise in the data. Adding this feature is a controversial decision because they are few observations (breaking points) in a match, which inevitably result in extreme probabilities. However, this issue is partly alleviated by how we construct features, which is explained in the next section.

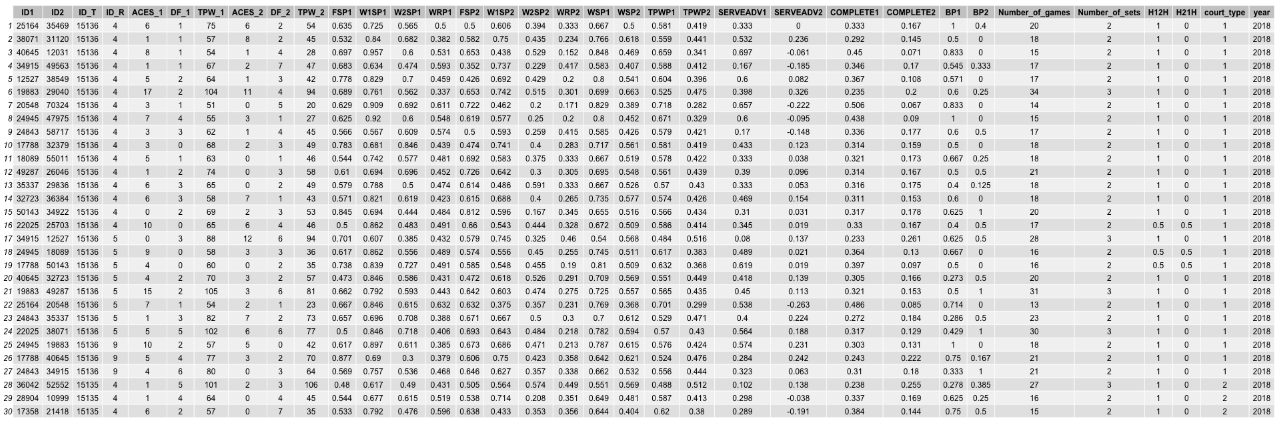
Additional Information

For every match, the total number of games, the total number of sets, the year and the court type of the match are also added to our data frame, as these data are needed for construction our final features. Computing number of games and sets also give us flexibility for further work, as we are also interested in predicting the number of games and sets. A Python script is responsible for extracting advanced metrics, head to head statistics and additional information from the initial OnCourt per match statistics. The resulting data frame is converted into a Sqlite3 Database for further use. The first thirty rows of our initial dataset and the resulting database can be seen below.[3]



	ID1	ID2	ID_T	ID_R	FS_1	FSOF_1	ACES_1	DF_1	W1S_1	W1SOF_1	W2S_1	W2SOF_1	BP_1	BPOF_1	TPW_1	FS_2	FSOF_2	ACES_2	DF_2	W1S_2	W1SOF_2	W2S_2	W2SOF_2	BP_2	BPOF_2	TPW_2	RPW_1	RPWOF_1	RPW_2	RPWOF_2
1	34915	12527	15136	5	61	87	0	3	37	61	10	26	5	8	88	55	95	12	6	41	55	13	40	6	12	94	41	95	40	87
2	24945	18089	15136	5	29	47	9	0	25	29	10	18	4	6	58	27	47	3	3	15	27	9	20	0	1	36	23	47	12	47
3	25164	35469	15136	4	40	63	6	1	29	40	13	23	5	5	75	33	66	6	2	20	33	13	33	2	5	54	33	66	21	63
4	38071	31120	15136	4	25	47	1	1	21	25	15	22	3	6	57	32	55	8	2	24	32	10	23	0	0	45	21	55	11	47
5	40645	12031	15136	4	23	33	8	1	22	23	6	10	5	6	54	32	49	1	4	14	32	9	17	0	0	28	26	49	5	33
6	34915	49563	15136	4	41	60	1	1	26	41	9	19	6	11	67	19	54	2	7	14	19	8	35	2	6	47	32	54	25	60
7	12527	38549	15136	4	35	45	5	2	29	35	7	10	4	7	64	26	61	1	3	18	26	15	35	0	3	42	28	61	9	45
8	19883	29040	15136	4	71	103	17	2	54	71	18	32	3	5	104	62	95	11	4	46	62	17	33	1	4	94	32	95	31	103
9	20548	70324	15136	4	22	35	3	1	20	22	9	13	5	6	51	26	36	0	5	12	26	2	10	0	0	20	22	36	6	35
10	24945	47975	15136	4	25	40	7	4	23	25	9	15	4	4	55	26	42	3	1	15	26	4	16	0	0	27	23	42	8	40
11	24843	58717	15136	4	30	53	3	3	17	30	14	23	6	10	62	27	54	1	4	16	27	7	27	3	6	45	31	54	22	53
12	17788	32379	15136	4	47	60	3	0	32	47	11	13	3	6	68	27	57	2	3	20	27	12	30	0	5	49	25	57	17	60
13	18089	55011	15136	4	31	57	5	1	23	31	15	26	4	6	63	36	52	0	1	21	36	6	16	1	4	46	25	52	19	57
14	49287	26046	15136	4	36	59	1	2	25	36	16	23	4	8	74	53	73	0	3	34	53	6	20	2	4	58	33	73	18	59
15	35337	29836	15136	4	33	57	6	3	26	33	12	24	4	10	65	35	57	0	2	17	35	13	22	1	8	49	27	57	19	57
16	17788	50143	15136	5	31	42	4	0	26	31	8	11	4	8	60	31	53	0	2	17	31	10	22	0	0	35	26	53	8	42
17	40645	32723	15136	5	26	55	4	2	22	26	17	29	3	11	70	34	72	3	2	21	34	20	38	1	2	57	31	72	16	55
18	19883	49287	15136	5	53	80	15	2	42	53	16	27	5	10	105	68	106	3	6	41	68	18	38	2	2	81	47	106	22	80
19	25164	20548	15136	5	26	39	7	1	22	26	8	13	5	7	54	24	38	2	1	9	24	5	14	0	1	23	24	38	9	39
20	24843	35337	15136	5	46	70	1	3	32	46	17	24	2	7	82	57	85	7	2	38	57	14	28	1	2	73	33	85	21	70
21	22025	38071	15136	5	39	78	5	5	33	39	28	39	3	7	102	70	101	6	6	45	70	15	31	2	2	77	41	101	17	78
22	24945	19883	15136	9	29	47	10	2	26	29	11	18	3	3	57	35	52	5	0	24	35	8	17	0	0	42	20	52	10	47
23	17788	40645	15136	9	71	81	5	4	49	71	3	10	3	4	77	40	66	3	2	30	40	11	26	1	6	70	25	66	29	81
24	24843	34915	15136	9	37	65	4	6	28	37	15	28	4	12	80	51	79	0	3	32	51	10	28	2	2	64	37	79	22	65
25	22025	25164	15136	9	13	30	2	3	12	13	9	17	5	11	53	33	56	5	2	21	33	3	23	1	1	33	32	56	9	30
26	32723	36384	15136	4	28	49	6	3	23	28	13	21	3	5	58	32	52	7	1	22	32	8	20	0	0	43	22	52	13	49
27	50143	34922	15136	4	49	58	0	2	34	49	4	9	5	8	69	52	64	2	3	31	52	2	12	2	2	53	31	64	20	58
28	22025	25703	15136	4	29	58	10	0	25	29	14	29	4	10	65	35	53	6	4	19	35	8	18	1	2	46	26	53	19	58
29	36042	36948	15135	5	41	58	4	3	32	41	7	17	4	8	63	40	54	1	2	22	40	8	14	1	3	49	24	54	19	58
30	17358	26514	15135	5	42	74	1	3	29	42	18	32	5	12	92	40	92	2	8	24	40	23	52	1	5	74	45	92	27	74

Figure 2.1: On Court Per Match Statistics of Last 30 Matches



	ID1	ID2	ID_T	ID_R	ACES_1	DF_1	TPW_1	ACES_2	DF_2	TPW_2	FSP1	W1SP1	W2SP1	WRP1	FSP2	W1SP2	W2SP2	WRP2
1	25164	35469	15136	4	6	1	75	6	2	54	0.635	0.725	0.565	0.5	0.5	0.606	0.394	0.333
2	38071	31120	15136	4	1	1	57	8	2	45	0.532	0.84	0.682	0.382	0.582	0.75	0.435	0.234
3	40645	12031	15136	4	8	1	54	1	4	28	0.697	0.957	0.6	0.531	0.653	0.438	0.529	0.152
4	34915	49563	15136	4	1	1	67	2	7	47	0.683	0.634	0.474	0.593	0.352	0.737	0.229	0.417
5	12527	38549	15136	4	5	2	64	1	3	42	0.778	0.829	0.7	0.459	0.426	0.692	0.429	0.2
6	19883	29040	15136	4	17	2	104	11	4	94	0.689	0.761	0.562	0.337	0.653	0.742	0.515	0.301
7	20548	70324	15136	4	3	1	51	0	5	20	0.629	0.909	0.692	0.611	0.722	0.462	0.2	0.171
8	24945	47975	15136	4	7	4	55	3	1	27	0.625	0.92	0.6	0.548	0.619	0.577	0.25	0.2
9	24843	58717	15136	4	3	3	62	1	4	45	0.566	0.567	0.609	0.574	0.5	0.593	0.259	0.415
10	17788	32379	15136	4	3	0	68	2	3	49	0.783	0.681	0.846	0.439	0.474	0.741	0.4	0.283
11	18089	55011	15136	4	5	1	63	0	1	46	0.544	0.742	0.577	0.481	0.692	0.583	0.375	0.333
12	49287	26046	15136	4	1	2	74	0	3	58	0.61	0.694	0.696	0.452	0.726	0.642	0.3	0.305
13	35337	29836	15136	4	6	3	65	0	2	49	0.579	0.788	0.5	0.474	0.614	0.486	0.591	0.333
14	32723	36384	15136	4	6	3	58	7	1	43	0.571	0.821	0.619	0.423	0.615	0.688	0.4	0.265
15	50143	34922	15136	4	0	2	69	2	3	53	0.845	0.694	0.444	0.484	0.812	0.596	0.167	0.345
16	22025	25703	15136	4	10	0	65	6	4	46	0.5	0.862	0.483	0.491	0.66	0.543	0.444	0.328
17	34915	12527	15136	5	0	3	88	12	6	94	0.701	0.607	0.385	0.432	0.579	0.745	0.325	0.46
18	24945	18089	15136	5	9	0	58	3	3	36	0.617	0.862	0.556	0.489	0.574	0.556	0.45	0.255
19	17788	50143	15136	5	4	0	60	0	2	35	0.738	0.839	0.727	0.491	0.585	0.548	0.455	0.19
20	40645	32723	15136	5	4	2	70	3	2	57	0.473	0.846	0.586	0.431	0.472	0.618	0.526	0.291
21	19883	49287	15136	5	15	2	105	3	6	81	0.662	0.792	0.593	0.443	0.642	0.603	0.474	0.275
22	25164	20548	15136	5	7	1	54	2	1	23	0.667	0.846	0.615	0.632	0.632	0.375	0.357	0.231
23	24843	35337	15136	5	1	3	82	7	2	73	0.657	0.696	0.708	0.388	0.671	0.667	0.5	0.3
24	22025	38071	15136	5	5	5	102	6	6	77	0.5	0.846	0.718	0.406	0.693	0.643	0.484	0.218
25	24945	19883	15136	9	10	2	57	5	0	42	0.617	0.897	0.611	0.385	0.673	0.686	0.471	0.213
26	17788	40645	15136	9	5	4	77	3	2	70	0.877	0.69	0.3	0.379	0.606	0.75	0.423	0.358
27	24843	34915	15136	9	4	6	80	0	3	64	0.569	0.757	0.536	0.468	0.646	0.627	0.357	0.338
28	36042	52552	15135	4	1	5	101	2	3	106	0.48	0.617	0.49	0.431	0.505	0.564	0.574	0.449
29	28904	10999	15135	4	1	4	64	0	4	45	0.544	0.677	0.615	0.519	0.538	0.714	0.208	0.351
30	17358	21418	15135	4	6	2	57	0	7	35	0.533	0.792	0.476	0.596	0.638	0.433	0.353	0.356

	WSP1	WSP2	TPWP1	TPWP2	SERVEADV1	SERVEADV2	COMPLETE1	COMPLETE2	BP1	BP2	Number_of_games	Number_of_sets	H12H	H21H	court_type	year
1	0.667	0.5	0.581	0.419	0.333	0	0.333	0.167	1	0.4	20	2	1	0	1	2018
2	0.766	0.618	0.559	0.441	0.532	0.236	0.292	0.145	0.5	0	18	2	1	0	1	2018
3	0.848	0.469	0.659	0.341	0.697	-0.061	0.45	0.071	0.833	0	15	2	1	0	1	2018
4	0.583	0.407	0.588	0.412	0.167	-0.185	0.346	0.17	0.545	0.333	17	2	1	0	1	2018
5	0.8	0.541	0.604	0.396	0.6	0.082	0.367	0.108	0.571	0	17	2	1	0	1	2018
6	0.699	0.663	0.525	0.475	0.398	0.326	0.235	0.2	0.6	0.25	34	3	1	0	1	2018
7	0.829	0.389	0.718	0.282	0.657	-0.222	0.506	0.067	0.833	0	14	2	1	0	1	2018
8	0.8	0.452	0.671	0.329	0.6	-0.095	0.438	0.09	1	0	15	2	1	0	1	2018
9	0.585	0.426	0.579	0.421	0.17	-0.148	0.336	0.177	0.6	0.5	17	2	1	0	1	2018
10	0.717	0.561	0.581	0.419	0.433	0.123	0.314	0.159	0.5	0	18	2	1	0	1	2018
11	0.667	0.519	0.578	0.422	0.333	0.038	0.321	0.173	0.667	0.25	18	2	1	0	1	2018
12	0.695	0.548	0.561	0.439	0.39	0.096	0.314	0.167	0.5	0.5	21	2	1	0	1	2018
13	0.667	0.526	0.57	0.43	0.333	0.053	0.316	0.175	0.4	0.125	18	2	1	0	1	2018
14	0.735	0.577	0.574	0.426	0.469	0.154	0.311	0.153	0.6	0	18	2	1	0	1	2018
15	0.655	0.516	0.566	0.434	0.31	0.031	0.317	0.178	0.625	1	20	2	1	0	1	2018
16	0.672	0.509	0.586	0.414	0.345	0.019	0.33	0.167	0.4	0.5	17	2	0.5	0.5	1	2018
17	0.54	0.568	0.484	0.516	0.08	0.137	0.233	0.261	0.625	0.5	28	3	1	0	1	2018
18	0.745	0.511	0.617	0.383	0.489	0.021	0.364	0.13	0.667	0	16	2	0.5	0.5	1	2018
19	0.81	0.509	0.632	0.368	0.619	0.019	0.397	0.097	0.5	0	16	2	0.5	0.5	1	2018
20	0.709	0.569	0.551	0.449	0.418	0.139	0.305	0.166	0.273	0.5	20	2	1	0	1	2018
21	0.725	0.557	0.565	0.435	0.45	0.113	0.321	0.153	0.5	1	31	3	1	0	1	2018
22	0.769	0.368	0.701	0.299	0.538	-0.263	0.486	0.085	0.714	0	13	2	1	0	1	2018
23	0.7	0.612	0.529	0.471	0.4	0.224	0.272	0.184	0.286	0.5	23	2	1	0	1	2018
24	0.782	0.594	0.57	0.43	0.564	0.188	0.317	0.129	0.429	1	30	3	1	0	1	2018
25	0.787	0.615	0.576	0.424	0.574	0.231	0.303	0.131	1	0	18	2	1	0	1	2018
26	0.642	0.621	0.524	0.476	0.284	0.242	0.243	0.222	0.75	0.167	21	2	1	0	1	2018
27	0.662	0.532	0.556	0.444	0.323	0.063	0.31	0.18	0.333	1	21	2	1	0	1	2018
28	0.551	0.569	0.488	0.512	0.102	0.138	0.238	0.255	0.278	0.385	27	3	1	0	2	2018
29	0.649	0.481	0.587	0.413	0.298	-0.038	0.337	0.169	0.625	0.25	16	2	1	0	2	2018
30	0.644	0.404	0.62	0.38	0.289	-0.191	0.384	0.144	0.75	0.5	15	2	1	0	2	2018

Figure 2.2: Calculated Per Match Statistics of Last 30 Matches

In Figure 2.2 the number of decimals is limited to three for readability.

[3]You can find the script for extracting metrics and information here.

2.3 Feature Construction

After developing metrics representing the qualities of the players in the previous section, the next step is to generate the final feature set to train our model. As we are trying to predict the outcome *pre-game*, we cannot directly use the metrics we calculated for a match as our final features, as none of them are computable pre-game. It follows logically for each match, the final feature set must be estimated based on the past performance of the players, which is what Knottenbelt does for estimating serve and return winning percentages. With this approach, our features become the *statistics we are predicting the match will have*, as we are estimating the statistics of an upcoming match. Generalizing Knottenbelt's method, we can use the subset of past matches of both players where they faced the same opponents and find the average value of our final features for both players. Taking the difference of the same feature between two players, as described in Table 2.1, will give us the final feature. Knottenbelt and Sipko use the same model, which is shown in Figure 2.3.

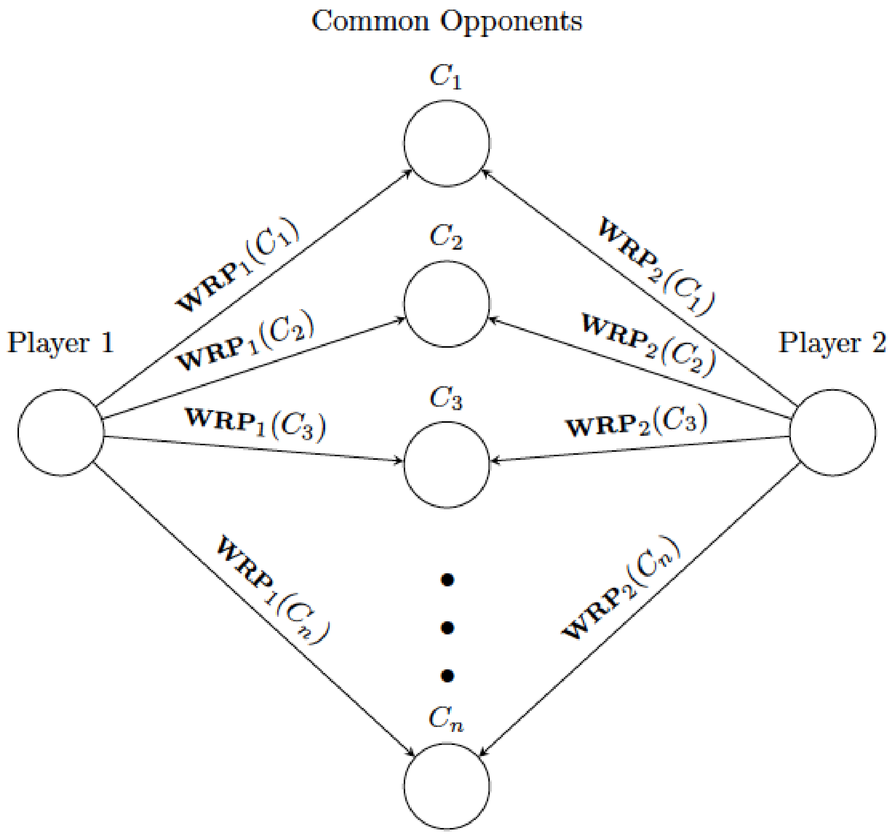
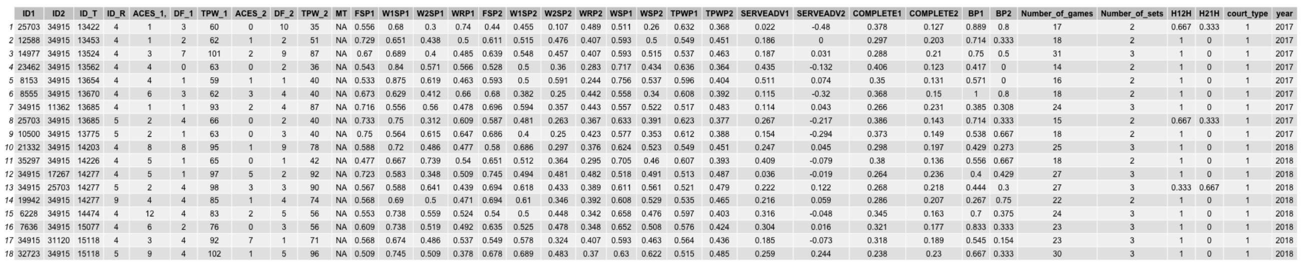


Figure 2.3: Common Opponent Model [Taken from Sipko]

To see the workflow of deriving features, et us look at the match between Pruchya

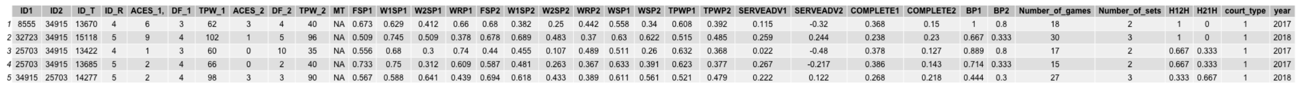
Isarow and Linh Giang Trinh played in Montevideo Challenger 2018[4] as an example. First we extract all the past matches of both of these players. Isarow has 58 past matches and Giang Trinh has 21. Figure 2.4 shows Giang Trinh's past matches in our dataset.



	ID1	ID2	ID_T	ID_R	ACES_1,	DF_1	TPW_1	ACES_2	DF_2	TPW_2	MT	FSP1	W1SP1	W2SP1	WRP1	FSP2	W1SP2	W2SP2	WRP2	WSP1	WSP2	TPWP1	TPWP2	SERVEADV1	SERVEADV2	COMPLETE1	COMPLETE2	BP1	BP2	Number_of_games	Number_of_sets	H12H	H21H	court_type	year
1	25703	34915	13422	4	1	3	60	0	10	35	NA	0.556	0.68	0.3	0.74	0.44	0.455	0.107	0.489	0.511	0.26	0.632	0.368	0.022	-0.48	0.378	0.127	0.889	0.8	17	2	0.667	0.333	1	2017
2	12588	34915	13453	4	1	2	62	1	2	51	NA	0.729	0.651	0.438	0.5	0.611	0.515	0.476	0.407	0.593	0.5	0.549	0.451	0.186	0	0.297	0.203	0.714	0.333	18	2	1	0	1	2017
3	14977	34915	13524	4	3	7	101	2	9	87	NA	0.67	0.689	0.4	0.485	0.639	0.548	0.457	0.407	0.593	0.515	0.537	0.463	0.187	0.031	0.288	0.21	0.75	0.5	31	3	1	0	1	2017
4	23462	34915	13562	4	4	0	63	0	2	36	NA	0.543	0.84	0.571	0.566	0.528	0.5	0.36	0.283	0.717	0.434	0.636	0.364	0.435	-0.132	0.406	0.123	0.417	0	14	2	1	0	1	2017
5	8153	34915	13654	4	4	1	59	1	1	40	NA	0.533	0.875	0.619	0.463	0.593	0.5	0.591	0.244	0.756	0.537	0.596	0.404	0.511	0.074	0.35	0.131	0.571	0	16	2	1	0	1	2017
6	8555	34915	13670	4	6	3	62	3	4	40	NA	0.673	0.629	0.412	0.66	0.68	0.382	0.25	0.442	0.558	0.34	0.608	0.392	0.115	-0.32	0.368	0.15	1	0.8	18	2	1	0	1	2017
7	34915	11362	13685	4	1	1	93	2	4	87	NA	0.716	0.556	0.56	0.478	0.696	0.594	0.357	0.443	0.557	0.522	0.517	0.483	0.114	0.043	0.266	0.231	0.385	0.308	24	3	1	0	1	2017
8	25703	34915	13685	5	2	4	66	0	2	40	NA	0.733	0.75	0.312	0.609	0.587	0.481	0.263	0.367	0.633	0.391	0.623	0.377	0.267	-0.217	0.386	0.143	0.714	0.333	15	2	0.667	0.333	1	2017
9	10500	34915	13775	5	2	1	63	0	3	40	NA	0.75	0.564	0.615	0.647	0.686	0.4	0.25	0.423	0.577	0.353	0.612	0.388	0.154	-0.294	0.373	0.149	0.538	0.667	18	2	1	0	1	2017
10	21332	34915	14203	4	8	8	95	1	9	78	NA	0.588	0.72	0.486	0.477	0.58	0.686	0.297	0.376	0.624	0.523	0.549	0.451	0.247	0.045	0.298	0.197	0.429	0.273	25	3	1	0	1	2018
11	35297	34915	14226	4	5	1	65	0	1	42	NA	0.477	0.667	0.739	0.54	0.651	0.512	0.364	0.295	0.705	0.46	0.607	0.393	0.409	-0.079	0.38	0.136	0.556	0.667	18	2	1	0	1	2018
12	34915	17267	14277	4	5	1	97	5	2	92	NA	0.723	0.583	0.348	0.509	0.745	0.494	0.481	0.482	0.518	0.491	0.513	0.487	0.036	-0.019	0.264	0.236	0.4	0.429	27	3	1	0	1	2018
13	34915	25703	14277	5	2	4	98	3	3	90	NA	0.567	0.588	0.641	0.439	0.694	0.618	0.433	0.389	0.611	0.561	0.521	0.479	0.222	0.122	0.268	0.218	0.444	0.3	27	3	0.333	0.667	1	2018
14	19942	34915	14277	9	4	4	85	1	4	74	NA	0.568	0.69	0.5	0.471	0.694	0.61	0.346	0.392	0.608	0.529	0.535	0.465	0.216	0.059	0.286	0.207	0.267	0.75	22	2	1	0	1	2018
15	6228	34915	14474	4	12	4	83	2	5	56	NA	0.553	0.738	0.559	0.524	0.54	0.5	0.448	0.342	0.658	0.476	0.597	0.403	0.316	-0.048	0.345	0.163	0.7	0.375	24	3	1	0	1	2018
16	7636	34915	15077	4	6	2	76	0	3	56	NA	0.609	0.738	0.519	0.492	0.635	0.525	0.478	0.348	0.652	0.508	0.576	0.424	0.304	0.016	0.321	0.177	0.833	0.333	23	3	1	0	1	2018
17	34915	31120	15118	4	3	4	92	7	1	71	NA	0.568	0.674	0.486	0.537	0.549	0.578	0.324	0.407	0.593	0.463	0.564	0.436	0.185	-0.073	0.318	0.189	0.545	0.154	23	3	1	0	1	2018
18	32723	34915	15118	5	9	4	102	1	5	96	NA	0.509	0.745	0.509	0.378	0.678	0.689	0.483	0.37	0.63	0.622	0.515	0.485	0.259	0.244	0.238	0.23	0.667	0.333	30	3	1	0	1	2018

Figure 2.4: Past Matches of Linh Giang Trinh

Next we find the players which both players have played against. These players had three common opponents in the past, which is definitely a below average sample size. The number of common opponents is positively correlated with player's performance estimates and consequently the predictive strength of our model. When we start evaluating our model, the number of past common opponents will be useful metric to evaluate the confidence of our model in the prediction it makes.



	ID1	ID2	ID_T	ID_R	ACES_1,	DF_1	TPW_1	ACES_2	DF_2	TPW_2	MT	FSP1	W1SP1	W2SP1	WRP1	FSP2	W1SP2	W2SP2	WRP2	WSP1	WSP2	TPWP1	TPWP2	SERVEADV1	SERVEADV2	COMPLETE1	COMPLETE2	BP1	BP2	Number_of_games	Number_of_sets	H12H	H21H	court_type	year
1	8555	34915	13670	4	6	3	62	3	4	40	NA	0.673	0.629	0.412	0.66	0.68	0.382	0.25	0.442	0.558	0.34	0.608	0.392	0.115	-0.32	0.368	0.15	1	0.8	18	2	1	0	1	2017
2	32723	34915	15118	5	9	4	102	1	5	96	NA	0.509	0.745	0.509	0.378	0.678	0.689	0.483	0.37	0.63	0.622	0.515	0.485	0.259	0.244	0.238	0.23	0.667	0.333	30	3	1	0	1	2018
3	25703	34915	13422	4	1	3	60	0	10	35	NA	0.556	0.68	0.3	0.74	0.44	0.455	0.107	0.489	0.511	0.26	0.632	0.368	0.022	-0.48	0.378	0.127	0.889	0.8	17	2	0.667	0.333	1	2017
4	25703	34915	13685	5	2	4	66	0	2	40	NA	0.733	0.75	0.312	0.609	0.587	0.481	0.263	0.367	0.633	0.391	0.623	0.377	0.267	-0.217	0.386	0.143	0.714	0.333	15	2	0.667	0.333	1	2017
5	34915	25703	14277	5	2	4	98	3	3	90	NA	0.567	0.588	0.641	0.439	0.694	0.618	0.433	0.389	0.611	0.561	0.521	0.479	0.222	0.122	0.268	0.218	0.444	0.3	27	3	0.333	0.667	1	2018

Figure 2.5: Matches of Linh Giang Trinh played with common opponents of Pruchya Isarow

In order to construct the serve advantage feature for this match (SERVEADV), we average the values from matches both players played against their common opponents and take their difference, as shown in the table below. The final SERVEADV feature for this match will be 0.313.

Table 2.2: Calculating the SERVEADV feature for Isarow - Trinh

Matches	ServeAdv_Isarow	ServeAdv_Trinh
Match 1	0.387	-0.32
Match 2	0.316	0.244
Match 3	-0.094	-0.48
Match 4	NA	-0.217
Match 5	NA	0.222
Average Serve Advantage	0.203	-0.11

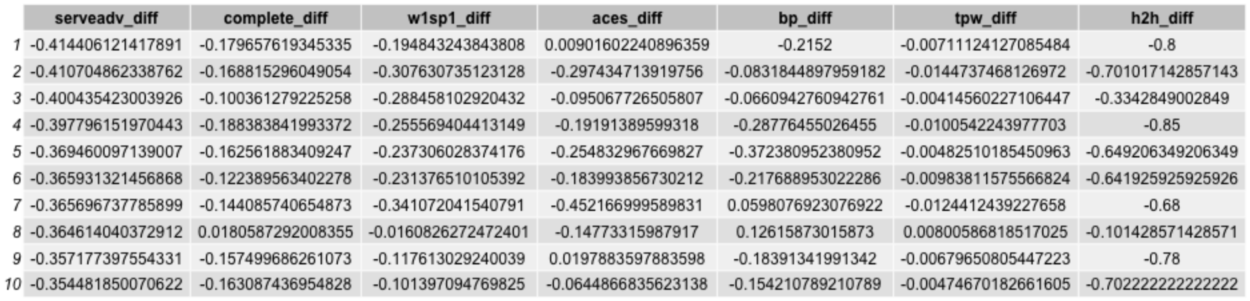
[4]`http://www.tennislive.net/atp/match/pruchya-isarow-VS-linh-giang-trinh/montevideo-challenger-2018/`

2.4 Feature Weighting

The simple averaging of player performance across all common opponents does not take the surface and the time the match played into account. This is a naive approach, as we would definitely expect the player's more recent matches to be more important estimates of his future performance than old matches. Therefore, we use time discounting and surface matrix weighting, as suggested by Knottenbelt and Sipko. Time discounting gives higher weights to more recent matches according to an exponential function, whereas surface weighting aims to increase the weight of matches played in the same surface more than other surfaces, as both player's past matches on the same surface tend to be more informative.

2.5 Final Learning Set

After constructing the weighted feature set according to our common-opponent model, we end up with our final learning set, which includes ~110.000 ATP professional tennis matches played in between 2009-2018. A snapshot of our final feature set can be seen in 2.6. The correlation matrix is also shown. Positive correlations are displayed in blue and negative correlations in red color. Color intensity and the size of the circle are proportional to the correlation coefficients. In the right side of the correlogram, the legend color shows the correlation coefficients and the corresponding colors. The second chart shows for each feature, the correlation coefficient (r), the significance of the relationship (p-value) with codes, the histogram with kernel density estimation and the scatter plot with fitted line.



	serveadv_diff	complete_diff	w1sp1_diff	aces_diff	bp_diff	tpw_diff	h2h_diff
1	-0.414406121417891	-0.179657619345335	-0.194843243843808	0.00901602240896359	-0.2152	-0.00711124127085484	-0.8
2	-0.410704862338762	-0.168815296049054	-0.307630735123128	-0.297434713919756	-0.0831844897959182	-0.0144737468126972	-0.701017142857143
3	-0.400435423003926	-0.100361279225258	-0.288458102920432	-0.095067726505807	-0.0660942760942761	-0.00414560227106447	-0.3342849002849
4	-0.397796151970443	-0.188383841993372	-0.255569404413149	-0.19191389599318	-0.28776455026455	-0.0100542243977703	-0.85
5	-0.369460097139007	-0.162561883409247	-0.237306028374176	-0.254832967669827	-0.372380952380952	-0.00482510185450963	-0.649206349206349
6	-0.365931321456868	-0.122389563402278	-0.231376510105392	-0.183993856730212	-0.217688953022286	-0.00983811575566824	-0.641925925925926
7	-0.365696737785899	-0.144085740654873	-0.341072041540791	-0.452166999589831	0.0598076923076922	-0.0124412439227658	-0.68
8	-0.364614040372912	0.0180587292008355	-0.0160826272472401	-0.14773315987917	0.12615873015873	0.00800586818517025	-0.101428571428571
9	-0.357177397554331	-0.157499686261073	-0.117613029240039	0.0197883597883598	-0.18391341991342	-0.00679650805447223	-0.78
10	-0.354481850070622	-0.163087436954828	-0.101397094769825	-0.0644866835623138	-0.154210789210789	-0.00474670182661605	-0.702222222222222

Figure 2.6: First 10 rows of our Learning Set

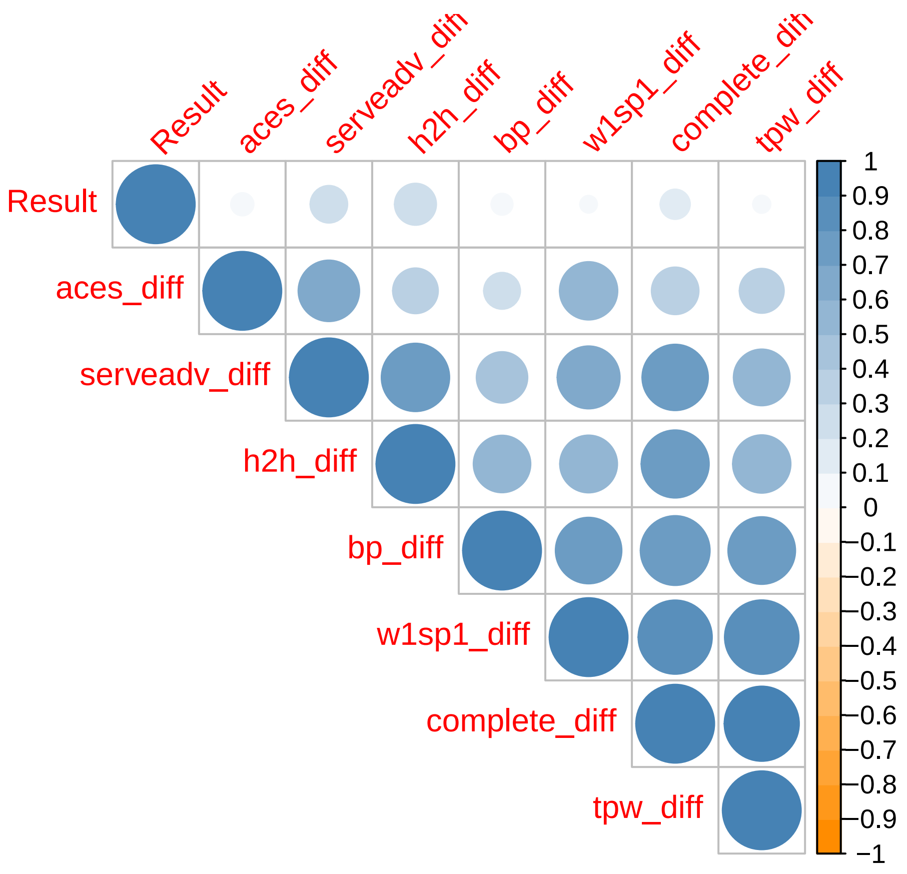


Figure 2.7: Correlation Matrix for our Final Features. SERVEADV AND H2H have the highest correlations.

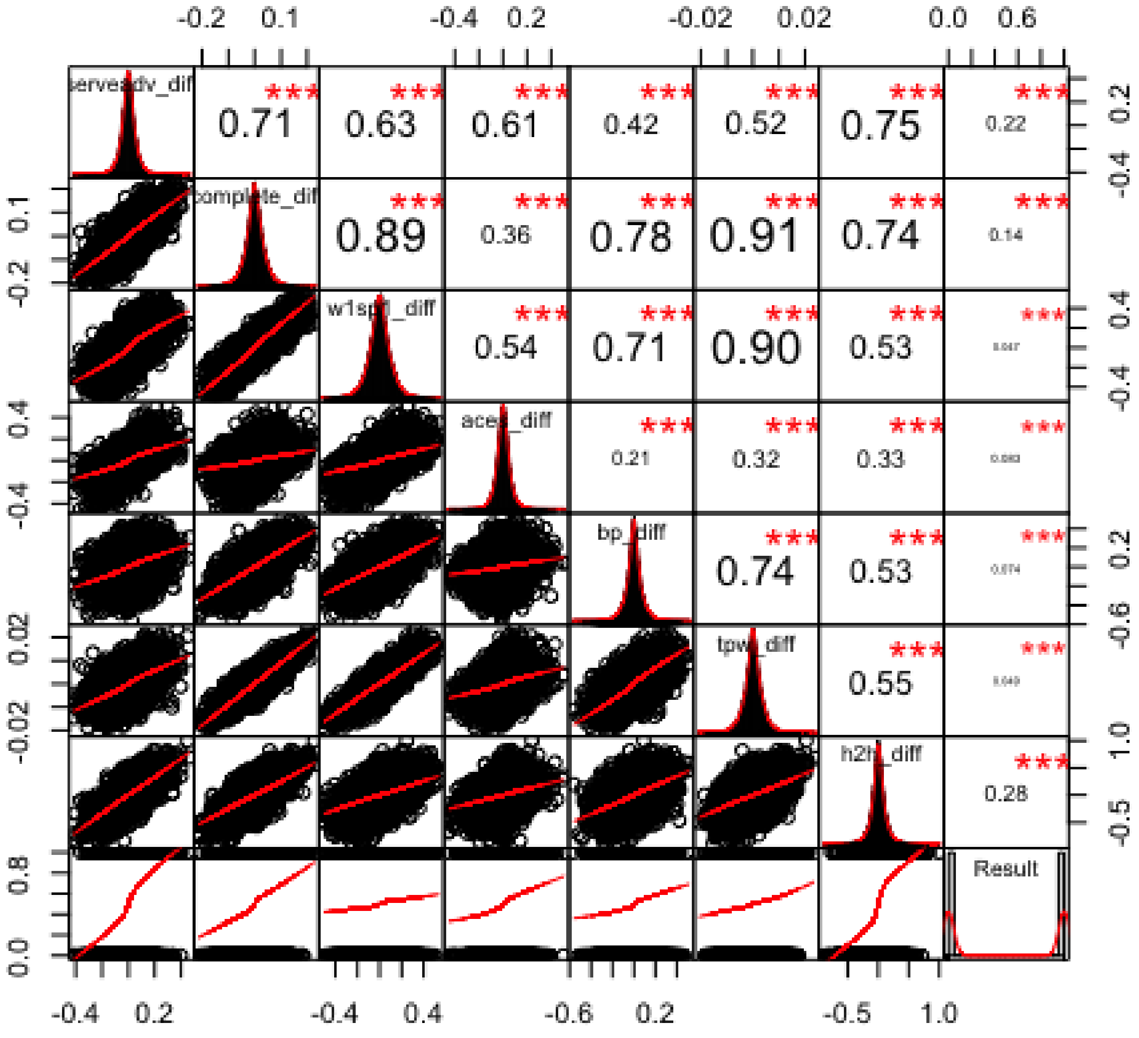


Figure 2.8: The Performance Analytics Plot

Chapter 3

Machine Learning Models

3.1 Stacked Generalization - An Ensemble Model

Dietterich (1997) and Laan, Polleyy, & Hubbardz (2007) define Ensemble Models as Supervised Learning techniques that are used to boost predictive accuracy by combining the predictions of multiple machine learning models. The idea is to generate multiple models $L1, ..., L_N$ on a learning set, S,which consists of examples $s_i = (x_i, y_i)$. This creates so-called base-level classifiers $C_1, C_2, ..., C_N$, where where $C_i = L_i(S)$. Then, a second phase combines their outputs on their hypothesis H_x to generate a second level model. The second level model, a so-called *Meta Learner*, uses the meta information provided by base classifiers, which balances the bias-variance tradeoff. Here, meta-learning literally means learning about learning. The purpose of Meta Learner is to *"optimally integrate what each of the base level generalizers says about the learning set"*. A common approach to this type of learning is Stacked Generalization, as introduced by David (1992). Stacked Generalization, also called stacking, is a type of ensemble method in which the predictions that are generated by different *base level* learning models are used as inputs in a second-level learning model, the *Meta Leaner*. The idea behind stacking is to *stack* the predictions $f_1, ..., f_m$ by some combination with weights $a_i \in 1, ..., m$:

$$f_{stacking}(x) = \sum_{n=1}^{m} a_n \ f_n \ x \tag{3.1}$$

The pseudocode for stacking can be seen in the following Table 3.1.

Table 3.1: Stacking Algorithm Pseudocode

```
1: Input: training data $D = (x_i, y_i)$ for all i in m.
2: Output: ensemble classifier H
3: Step 1: *learn base level classifiers*
4: for t = 1 to T do
5: learn $h_t$ based on D
6: end for
7: Step 2: *construct new data set of predictions*
8: for t = 1 to m do
9: $D_h = (x_i', y_i)$ where $x_i' = (h_1(x_i), ..., h_T(x_i))$
10: end for
11: Step 3: *learn a meta-level classifier*
12: learn H based on $D_h$
13: return H
```

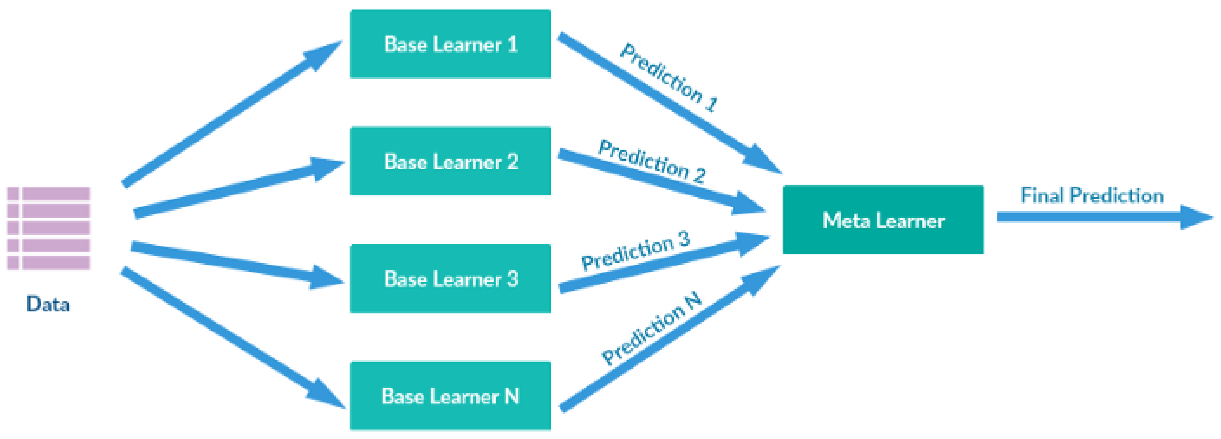


Figure 3.1: Visualization of Stacking

Some researchers describe this method as a more *sophisticated cross-validation.* Cross-validation technique follows a winner-takes-all strategy, whereas stacked generalization's strategy is to combine all generalizers rather than choosing one amongst them. Another way to view stacked generalization is as a collection of base classifiers that estimate their own generalizing biases with respect to the training set. One can, of course, argue that the combination of learners might not be the optimal solution. Given a task S and a set of learning algorithms L, there is a learning algorithm L_k in L that performs better than all of the others on S. This is inevitable, and in general holds for any generalizing method. That being said, the actual performance of L_k may still be poor and a suboptimal solution. By replacing this single model selection with a combination of models, the scheme estimates the error of individual classifiers and aims to correct those errors. Thereby mitigating the risk of a suboptimal solution. This point highlights the power of the stacking, which is the ability to both consolidate accurate predictions and correct errors across many diverse base classifiers, thereby reducing generalization error. The key word here is diversity, as Dietterich (2000) explains "*It is a necessary and sufficient condition for an ensemble model to be more accurate than any of its individual members is if the classifiers are accurate and diverse. If there is a complete consensus the ensemble cannot outperform the best base classifier*". Diversity is usually achieved by creating randomized partitions of the learning set, thereby reducing the biases of one or more generalizers with respect to the learning set. The more each base level model has to say (i.e. the more unique they are), the more we learn from them. This also highlights another important point, that the main purpose of stacked generalization technique is not to achieve the best learning accuracy, but to achieve the best generalization accuracy. This fits the goal of the present thesis, as our model's performance will be determined by whether it can achieve a certain profit from betting on future tennis matches. In general, stacking has been immensely successful in producing accurate predictions for many complex classification tasks and its performance is proven in data science competitions such as Kaggle Analytics or Vidhya. Based on my research, no one has used an ensembling technique for tennis match prediction modeling. The present thesis will be the first to investigate whether the performance of stacking can be replicated in the domain of tennis match modeling.

3.2 Ensemble Selection:

Ensemble selection is the process of choosing an appropriate learner for the combinations of predictions of base-level classifiers. The appropriateness of a learner for a given domain is assessed according to predictive accuracy. Our first ensemble model will include 50 decision stumps of depth 8 as their base classifiers. Our meta level classifier, which will take in a hundred decision tree predictions as features, will be a Logistic Regression Model, as recommended by Ting (1999) for meta-level learning. Although decision trees have a drawback of poor generalization on the test set in some domains, we will use this ensemble model to have a baseline value of performance of ensemble models. Another reason is to be able to compare the performance of three different versions of stacking. The first version will stack fifty *class predictions* made by base models as our meta-level data. The second version will stack the class *probability distributions* of base level classifiers, rather than single class values. The meta-level attributes are thus the probabilities of each of the class values by each base level model. Many researchers, including Todorovski (2000) and Ting (1999) argue that this approach is more expressive, as it allows the meta-learner to learn from the confidence of the base level classifiers.[1] A third version of stacking we will test is one that does not replace initial features with the predictions, but one that appends the predictions to original features, resulting in a hybrid meta-example. All three versions will be trained on a dataset of 86088 matches and tested on 21523 matches. At each iteration (i.e. each stacking), we randomly subsample %10 of our training data to ensure diversity and independence amongst base-level classifiers. We use five-fold Stratified Cross Validation when stacking on the training set, which leads to statistically more reliable predictions. The best performing model predictions are used to stack the predictions on the unused half of our training set. It should be noted that the number of folds and percentage of the training set that is randomly subsampled are hyperparameters that need to be optimized. Results are displayed in Table **??**.

Table 3.2: Training and Testing Accuracies for Stacked Models

Stacking Method	Shape	Training Accuracy	Testing Accuracy
Probabilities	86088, 200	0.670	0.655
Probabilities + Features	86088, 207	0.673	0.627
Predictions	86088, 100	0.661	0.642
Predictions + Feature	86088, 107	0.664	0.665

3.3 Discussion of Model Evaluation

We can observe that stacking with probabilities does perform slightly better than stacking with predictions. However, the hybrid option of original features and probability distribution clearly overfits. This is probably not due to our stacking method, but due to the similarity and high correlation between our base level classifiers. As mentioned before, diverse base-level classifiers (with weakly correlated predictions)

[1]A Bayesian meta-level model to pool base level confidence levels has been suggested for tennis match modeling. In this case, independence assumption of Naive Bayes may result in performance issues, as probabilities given by different base level models are not independent.

are necessary for good generalization. As can be seen, all models test accuracy is in range 60-65%. However, this range of accuracy is not enough for our tennis model to have a positive return on investment. We evaluated the model against the best odds offered by bwin, an international bookmaker. The historical odds for the matches in our dataset are scraped from oddsportal. Our main observation is that any tennis model with %65 predictive accuracy **can not** be profitable in high volume tennis betting. This is due partly due to nature of tennis odds, in which there is a clear favorite with very low odds in a high pertencage of matches. Also, our learning set is not designed to perform high-volume betting. Clearly, our model performs best when there is enough common opponents between two players and its performance degrades as the number of common opponents for both players decreases. If two players had two or less common opponents in the past, our model cannot represent the qualities of both players because lack of data. Thus, we can choose to train and test our model on a subset of our learning set, where we only take in matches over a certain confidence value.[2] Subsampling %30 matches that we are most confident about from our initial dataset, we get somehow increased training accuricies with slightly improving test accuricies, a clearer sign of overfitting by our stacked generalization models.

To test this, we have created another ensemble model that uses class probability predictions from three *different* machine learning classifiers, a classifier implementing the k-nearest neighbors vote, a Gaussian Naive Bayes Classifier and an Extra Trees Classifier that fits a number of randomized decision trees (a.k.a. extra-trees) on various sub-samples of the dataset. We then appended these six new features from three classifiers to our original features, creating another hybrid feature set. This ensemble model, with only thirteen dimensions, was able to outperform our original decision stump based models, showing that the problem with decision trees as base-level classifiers is that you do not end up getting enough diversity amongst base level classifiers (if the dataset is not over >250K entries). The second hybrid model performed better on unseen data than the four models mentioned above. It should be mentioned that the last model has a significantly higher area under the curve, shortly AUC, value. Many researchers argue that AUC is a better predictor of more discriminative models on unseen data.

Table 3.3: Training and Testing Accuracies for Stacked Model based on KNN, Gaussian Naive-Bayes and Extra-Trees Classifier Predictions

Stacking Method	Shape	Training Accuracy	Testing Accuracy
Probabilities + Original Features	86088, 13	0.673	0.685

[2]For calculating of these confidence values, reader can refer to Knottenbelt and Sipko; Section 3.3

3.4 Feed Forward Neural Network

A Feed Forward Neural Network with two hidden layers was also used as a meta-learner. The neural network was trained using Randomized Relu as activation functions and with cyclical learning rates to improve generalization as suggested recently by Smith (2015). Different learning rate optimizers were also experimented, such as Adam, Adagrad and SGD from PyTorch library. The implementation of the neural net can be found at this Github Page. Neural Net, as a meta-learner, had the highest predictive and AUC accuracy on unseen data. We also were able to observe the softmax class probabilities that the network produced for our test set, which allowed us to make a second interpretation of our model evaluation in tennis match predictor domain. This is explained in a second discussion below.

3.5 A Second Discussion of Model Evaluation

Let us discuss the Wimbledon Quarter Final match between Roger Federer and Kevin Anderson. Roger Federer was a huge favorite with -1250 (1.07 in decimal odds). Anderson's odd was +861, which is approximately 8.50 in decimal odds. Our model agreed with the market, as it predicted a Federer win. As everybody knows, Federer lost the match 2:3 in what was the biggest upset of the tournament and a lot of people, including us, lost money on Federer. What we realized after the tournament was that a closer look to our model showed us another important point, the fact that we gave Federer a %56 probability to win the match. The simple classification told us that Federer was going to win the match. But if we've paid attention to the class probability of Federer, we see that our model expects the match to be much closer than the bookmakers think. So if we could derive bookmakers estimated probabilities from their odds, we could compare it with the odds of our own model and see if a player is undervalued in a match, meaning that our model expects the match to be a tight affair. This information can be than used in betting exchange markets, instead of betting on the result as we would in a traditional bookmaker. This is because betting exchanges allow their users the ability to both 'back' and 'lay' an outcome - so bettors can act as a 'bookmaker' by setting odds for an event. This is an important point because we can use convert our model probalities into odds and act much like a broker to find value and make a betting profit without even seeing the final score. Let us know go back to Federer - Anderson example to see how we can profit from market movement that we predicted. First thing is to convert bookmaker odds into probabilities, so that we can compare it with the probabilities that our model produces. The simple equation to convert a decimal odd into a probability would be

$$d_e = \frac{1}{p_e}$$

, where d_e is the decimal odds for event E, p_e is the bookmakers estimated probability of event E. However, bookmakers have a betting margin, which is defined as the difference between the odds (an implied probability) the customer is offered to bet at, and the true probability of the outcome. As an example let's again use the decimal odds of Wimbledon Quarter Final match between Federer and Anderson. If we calculate the probabilities using the formula above, we get:

$$\frac{1}{1.08} * 100 + \frac{1}{9.6} * 100 = 105.22$$

The margin for this market is therefore 5.22%. So in order to uncover the true probabilities, we have to take into account the betting margin, as follows:

$$d_e = \frac{1}{p_e + o_e}$$

, where o_e is the margin which the bookmaker adds to the decimal odds for event E. If we make the assumption that the margin is the same for both outcomes (Fededer win and Anderson win), we can calculate the true probabilities as:

$$p_e = \frac{1}{d_e - (o_e/2)}$$

Using these calculations we calculate Fededer's true win probability as

$$\frac{1}{1.07 - (0.0522/2)} = 0.91$$

and Anderson's win probability as

$$\frac{1}{8.5 - (0.0522/2)} = 0.09$$

Note that these two probabilites add up to 1. Using this information, we show the information we have on Federer - Anderson match on table below.

Table 3.4: The Match Decimal odds and Probabilities calculated by the market and our model.

Odds and Probabilities	Federer	Anderson
Bookmaker Decimal Odds*	1.07	8.5
Bookmaker Probabilities	0.91	9
Model Probabilities	0.56	44
Model Decimal Odds	1.8	2.26

The interpretation of the above table is straightforward. Our model thinks that Federer has a higher probability of winning the match but expects the match to be much closer than expected. Thus our model predicts a movement that will occur in the live betting market. So if we can lay (against) bet Federer at 1.07, since we calculate the true value of his odd as 1.80. Following the same logic, we can also back Anderson at 8.50, since we expect these odds to decrease once the match starts. To make a profit, we do not have to wait for the outcome of the match. We simply need the odds to come close to what our model predicts. So far, no one has used probabilistic outputs of a tennis model to make a profit from market movements in the exchange market. Since it is clear that no tennis model below %70 accuracy can have a positive return of investment from a large betting volume, we expect researchers to look for ways they can use their models in the betting exchange markets.

3.6 Limitations

The best stacked generalization models mentioned in this thesis are unable to break the %70 predictive accuracy on unseen data, making them unfit for making a positive return on investment on large volume pre-match bets. We need to have a betting strategy, similar to Kelly-criterion, to select certain matches to bet on in order to make profits on the pre-game market. Even if we show that there is a value when using model class probabilities in exchange markets, we have not yet come up with a certain framework to automatize our bets on the exchange markets. Certain important questions remain, such as what to do when our model agrees with the market odds or what is the exact market movement we should look for when our model thinks a player is undervalued. The final goal must always be a fully-automatic bet-placing system based on our ensemble model. In terms of match features, it is remarkable that we can achieve a %65 accuracy by just using 7 features. However, more features are needed to represent qualities of players, as the features representativeness is a bottleneck for. As more tennis related data will be made publicly available, our model will also be updated to take these new features into account.

Conclusion

The present work hopes to add to extensive research that has been conducted into the prediction of tennis matches. We have looked at how ensemble models can be used for tennis match predictor domain. Using 110008 tennis matches played between 2004-2017, we have devised a learning set that consists of seven features based on research by Sipko and Knottenbelt, using their proposed methods of weighing historical matches. We then introduced stacked generalization as a novel tennis prediction model. Based on our knowledge and research, no one has used an ensembling technique to combine information from multiple predictive models to generate a new tennis match model. We investigated whether the performance of multiple ensemble models can be replicated in the domain of tennis match prediction. Although all of our proposed ensemble models perform with similar accuracy compared to state of the art models proposed by other researchers, we argue that tennis models cannot yet achieve a positive return on investment on traditional pre-game betting markets against bookmakers. We then turn our attention to betting exchange markets and propose a new approach that uses model class probabilities to make a profit by predicting future tennis market movements. An example from Wimbledon 2018 is given to show why we think using class probabilities can give us new avenues of profit. Although we have only tested this new framework fully during the ATP Finals in November, we will continue our work to create a framework for betting on exchange markets before the first Grand Slam Tournament of 2019 season.

References

Barnett, T., & Clarke, S. R. (2005). Combining player statistics to predict outcomes of tennis matches. *IMA Journal of Management Mathematics*, *16*, 113–120.

Boulier, B. L., & Stekler, H. O. (1999). Are sports seedings good predictors? An evaluation. *International Journal of Forecasting*, *15*, 83–91.

Chen, Y., Tian, Y., & Zhong, Y. (2017). *Real time tennis match prediction using machine learning* (Master's thesis). University of Stanford.

Clarke, S., & Dyte, D. (2000). Using official ratings to simulate major tennis tournaments. *International Transactions in Operational Research*, *7*(6), 585–594.

David, W. H. (1992). Stacked generalization. *Complex Systems Group, Theoretical Division, and Center for Non-Linear Studies, MS B213*, 665.

Dietterich, T. G. (1997). Machine-learning research: Four current directions. *AI Magazine*, *18*(4), 97–139.

Dietterich, T. G. (2000). Ensemble methods in machine learning. *In Proceedings of theFirst International Workshop on Multiple Classifier Systems*, 1–15.

Hertzmann, A., & Zorin, D. (2001). Random walks in tennis. *Missiori Journal of Mathematical Sciences*.

James, O. A. (2008). Probability formulas and statistical analysis in tennis. *Journal of Quantitative Analysis in Sports*, *4*(2), 1–23.

Klaassen, F. J., & Magnus, J. R. (2003). Forecasting the winner of a tennis match. *European Journal of Operational Research*, (148), 257–267.

Laan, M. J. van der, Polleyy, E. C., & Hubbardz, A. E. (2007). Super learner. *U.C. Berkeley Division of Biostatistics Working Paper Series*, (222), 1–39.

Newton, P. K., & Keller, J. B. (2005). Probability of winning at tennis i. theory and data. *Studies in Applied Mathematics*, (114), 241–269.

Peters, J. (2017). *Predicting the outcomes of professional tennis matches* (Master's thesis). University of Edinburgh.

Praet, R. (2015). *Predicting sport results by using recommendation techniques* (Master's thesis). Ghent University.

Rokach, L. (2009). Ensemble-based classifiers. *Artificial Intelligence Review*, *33*(2), 1–39.

Sill, J., Takacs, G., Mackey, L., & Lin, D. (n.d.). Feature-weighted linear stacking.

Sipko, M. (2015). *Machine learning for the prediction of professional tennis matches* (Master's thesis). Imperial College London.

Smith, L. N. (2015). No more pesky learning rate guessing games. *CoRR*, *abs/1506.01186*. Retrieved from `http://arxiv.org/abs/1506.01186`

Somboonphokkaphan, A., Phimoltares, S., & Lursinsap, C. (2009). Tennis winner prediction based on time-series history with neural modeling. *IMECS*, *1*, 500–509.

Spanias, A. D., & Knottenbelt, B. W. (2013). Tennis player ranking using quantitative models. *Department of Computing, Imperial College London.*

Ting, &. W., K. M. (1999). Issues in stacked generalization. *Journal of Artificial Intelligence Research*, *10*, 271–289.

Todorovski, &. D., L. (2000). Combining multiple models with meta decision trees. *Fourth European Conference on Principles of Data Mining and Knowledge Discovery*, 54–64.

Webb, P. (2011). Why tennis is big business for bookmakers.

Whalen, S., & Pandey, G. (2013). A comparative analysis of ensemble classifiers: Case studies in genomics. *Department of Genetics and Genomic Sciences Icahn Institute for Genomics and Multiscale Biology.*

William J. Knottenbelt, D. S., & Madurska, A. M. (2012). A common-opponent stochastic model for predicting the outcome of professional tennis matches. *Computers and Mathematics with Applications*, *64*(17), 3820–3827.