Predicting Professional Tennis Player Success using Binary Classification Methods

Karan Vombatkere

(Dated: 15 December 2017)

CSC 440: Data Mining Project

A quantitative model was constructed to characterize and predict the success of professional male tennis players, given their performance in their first 40 matches. The (probabilistic) prediction of player success was transformed into a binary classification problem that was solved using 2 Machine Learning methods: Logistic Regression and a Multi-Layer Feed-Forward Neural Network. Based on a detailed analysis of the ATP ranking system and tournament performance, a top 30 ATP rank was used as a metric of player success. The classification of tennis player success was to essentially predict the likelihood of a given player attaining a top 30 rank. The machine learning models were trained using processed data consisting of input vectors of extracted features, such as cumulative/average match statistics from a player's initial 40 matches. Python scripts were executed to extract the required features from a data set containing detailed individual tennis match data from 2000 - 2017. Feature selection and hyperparameter tuning were performed on the models to optimize performance. The Logistic Regression model (accuracy=0.82) marginally outperformed the Neural Network (accuracy=0.78) on a test data set. In terms of stratified k-fold cross validation performance, both models performed similarly yielding a mean accuracy of about 76%, mean precision of 77% and mean recall of 75%.

I. INTRODUCTION

Men's professional tennis has (almost) always had an elite set of players that have dominated the rankings and bigger tournaments. Based on preliminary observations, it is evident that top-ranked tennis players in the world consistently outperform lower-ranked players. Understanding the trajectory of a tennis player's long-term performance is a complex problem, and consequently player success prediction is a good candidate problem to be approached using machine learning classification.

Predicting tennis match outcomes has been studied and both probabilistic Markov chain models as well as machine learning models have been used to predict the outcome of a match [2]. The current state-of-the-art probabilistic models use hierarchical stochastic expressions based on Markov chains [6].

The goal of this project is to investigate the use of (supervised) machine learning classification methods to predict the future success of a tennis player. We begin by first constructing a predictive model and extracting features from a dataset containing tennis match data. The underlying assumptions for constructing the model are explained and a metric to measure the 'success' of a player is chosen - a top 30 ATP rank - and explained based on the observations and results of [7] and [8].

After careful assessment of several classification models and approaches in [1], [2] and [6], Logistic Regression and Feed-Forward Neural Network methods were selected. Both these model have good performance characteristics for classification and are suitable given the type and size of dataset in this problem. This paper discusses the training process of the two models. We then analyze and compare the predictive classification performance of these two models on test data based on several metrics such as accuracy, precision, recall, F1 score, ROC curve and cross validation performance.

II. MODEL CONSTRUCTION

A. Data Source and Acquisition

The primary source of raw data for this project is a data set (csv file obtained from data.world website) containing detailed tennis match data of professional tennis matches played on the men's ATP tour from 2000 to 2017. This is a large data set with about 52,000 tuples and 48 attributes of tennis match data, containing information about the players and match statistics. Each row in the dataset represents a single match. The following Figure 1. shows the attribute information of the data set - tournament information, player details and match scores and data (such as number of serves, aces, etc).

FIG. 1: Attributes in ATP Tennis Match Data set

```
In [8]:  1 ATPData.columns

Out[8]: Index(['tourney_id', 'tourney_name', 'surface', 'draw_size', 'tourney_level',
               'tourney_date', 'match_num', 'winner_id', 'winner_seed', 'winner_entry',
               'winner_name', 'winner_hand', 'winner_ht', 'winner_ioc', 'winner_age',
               'winner_rank', 'winner_rank_points', 'loser_id', 'loser_seed',
               'loser_entry', 'loser_name', 'loser_hand', 'loser_ht', 'loser_ioc',
               'loser_age', 'loser_rank', 'loser_rank_points', 'score', 'best_of',
               'round', 'minutes', 'w_ace', 'w_df', 'w_svpt', 'w_1stIn', 'w_1stWon',
               'w_2ndWon', 'w_SvGms', 'w_bpSaved', 'w_bpFaced', 'l_ace', 'l_df',
               'l_svpt', 'l_1stIn', 'l_1stWon', 'l_2ndWon', 'l_SvGms', 'l_bpSaved',
               'l_bpFaced'],
              dtype='object')
```

The match data gives information about the exact

rank, age, points of both the winning and losing players. This dataset was mined first to extract a feature set for the machine learning model, by extracting average and cumulative match statistics for each player in the data set, and also generating a binary value for the target class label, explained subsequently.

B. Predictive Model and Assumptions

The first step in model construction involved the defining of a metric that would provide us with an accurate measure of a player's success. After an analysis of the men's ATP ranking system provided in [9] it was concluded that a top 30 ATP rank would serve as a good measure of a player being successful. The top 30 ranked players represent an elite group in men's tennis who consistently perform at a high level, and we accept this as 'success' for a tennis player. A top 10 ranking would be too stringent, since a player ranked in the top 30 also often has tournament wins and comparable performance data. On the other hand a top 50 ranking doesn't necessarily indicate player success since there are several top 50 ranked players who have never won a tournament.

We then identify a set of features and define the number of matches or time period that defines a player's initial match performance (initial match data). In order to be consistent across players, we consider the first 40 matches of the player, in chronological order in the dataset. We impose another constraint of age such that the 40 matches played should be before the player turned 26. This ensures that we have initial or early match data for all players that have played at least 40 matches on the tour.

There are several underlying assumptions in the predictive model, which are detailed below:

- We assume that there must be some correlation between a player's initial performance and his future success. If the two were not correlated then there would be a random distribution of success based on players' initial match data.
- It is assumed that 40 matches provides a fair representation of a player's initial performance. This would correspond to several tournament's of tennis matches and should (in theory) account for random deviations and minor performance outliers.
- It is assumed that the discussion and results of [7] are accurate and that tennis player's peak during or after their mid-late twenties. This allows us to use the age 26 as a threshold to filter out older players in the dataset
- We assume it is possible to extract a set of features, X from the raw dataset that represent the performance of a tennis player accurately and help model his future career trajectory, and that a top 30 rank is an adequate measure of a player's success

III. FEATURE EXTRACTION

A. Data Cleaning and Processing

Data cleaning and processing was an important element of this project, since we work a large (and imperfect) dataset. The *pandas* Python library was used to process and manipulate data. As a first measure, rows with incomplete match statistic data were removed and/or ignored in computation involving row iterations. Additionally, certain player heights were missing and were filled in manually.

Once features were extracted, all columns were normalized using the following min-max normalization procedure, to ensure an attributes data was between 0 and 1. Note here that v_i is a value under attribute A and is mapped to v_i' which lies in the range [0,1].

$$v_i' = \frac{v_i - min_A}{(max_A - min_A)}$$

B. Feature Vector Generation

A supervised machine learning algorithm requires a set of labelled examples for training. In the context of our model for predicting player success, the following two elements were generated from the raw dataset:

1. An input feature vector $\vec{X} = (x_1, x_2, ..., x_n)$, representing the various feature values for each row of the training data. In our model, these features are calculated cumulative and average match statistics as well as physical features such as height.
2. A target class label value y, representing the binary classification output value for a row. In our model, this is a True or False value depending on whether or not a given player broke into the top 30 ranked players in the world.

Python scripts were written to process the raw data set to extract the relevant features for this model. These features were selected based on a similar approach as used in [2] and represent the best estimated (non-random) predictors of a tennis player's success - match statistics and physical features. The features are described subsequently as well as the extraction method used for each feature.

In order to process the data in a organized manner, first a Python script was written to extract unique player information from the data set. Figure 2 shows this unique player information. The data yielded 1981 different players and then features relevant to each of these players were extracted, while keeping the constraint of 40 matches played under the age of 26.

FIG. 2: Extracted Unique Player Information

```
In [6]:
1 #View the Player Data
2 PlayerData.head(100)
```

Out[6]:

	PlayerID	PName	Age	Height	MaxRank	Hand	Country
102179	102179	Antony Dupuis	34.368241	185.0	57.0	R	FRA
102776	102776	Andrew Ilie	26.737851	180.0	38.0	R	AUS
103602	103602	Fernando Gonzalez	31.561944	183.0	5.0	R	CHI
102821	102821	Cecil Mamiit	35.208761	173.0	95.0	R	PHI
103387	103387	Paradorn Srichaphan	27.761807	185.0	9.0	R	THA
102205	102205	Sebastien Lareau	27.931554	183.0	94.0	R	CAN
101733	101733	Jan Siemerink	32.251882	183.0	82.0	L	NED
102925	102925	Justin Gimelstob	30.198494	196.0	73.0	R	USA
101727	101727	Jason Stoltenberg	31.225188	185.0	63.0	R	AUS
101826	101826	Alex Lopez Moron	29.270363	175.0	111.0	R	ESP
103181	103181	Jiri Vanek	31.085558	185.0	78.0	R	CZE

1. **Player Height:** The player's height in cm was recorded, *Height(cm)*

2. **Win Rates:** Each player's average win rate information against differently ranked opponents was computed over all of his initial matches. *Overall Win %* is the player's win rate against all opponents, *Top 100 Win %* is the player's win rate against opponents ranked in the top 100 and *Top 30 Win %* is the player's win rate against opponents ranked in the top 30.

3. **First Serve Statistics:** Each player's mean first serve statistics were computed over all of his initial matches. *First Serve %* and *First Serve Win %* represent the mean percentage of first serves a player made and the mean percentage of points he then won.

4. **Break Point Performance:** Each player's mean break point statistics were computed over all of his initial matches. *BPSave %* and *BPConv %* represent the mean percentage of break points a player saved and the mean percentage of break point chances he converted.

5. **Aces and Double Faults:** The number of aces and double faults for each player was used to calculate their mean number of aces per match, *Aces/Match* and their mean number of double faults per match, *DF/Match*

6. **Top 30 Target Class:** The entire data set was scanned for each player to check if the player ever attained a rank in the top 30, *Future Top 30*, and could be referred to as successful (True) or not (False), in terms of our classification problem.

These 9 distinct features (italicized above) were calculated for each player the raw data set yielded 275 tuples of processed data that could be then used to train and test the machine learning models. This is discussed in the subsequent section.

C. Preliminary Feature Analysis

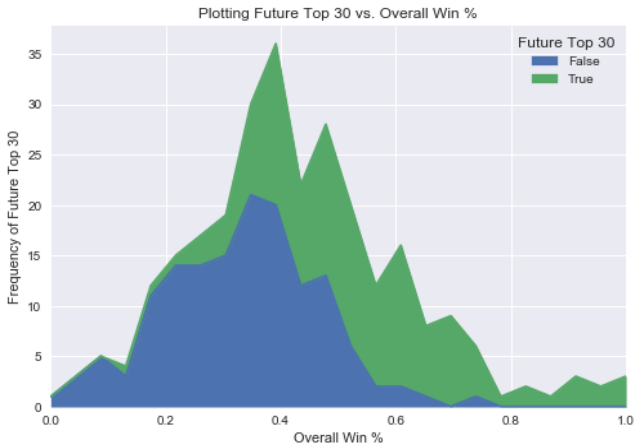
Figure 3 below shows the set of the different features that were extracted from the raw data. Some preliminary analysis was conducted on this data to get a better understanding of the relationship between attributes and the dependence of the target class label (*Future Top 30*) on them.

FIG. 3: Extracted Player Features from ATP Match Data

```
['Height(cm)',
 'Overall Win%',
 'Top 100 Win %',
 'Top 30 Win %',
 'First Serve %',
 'First Serve Win %',
 'BPSave %',
 'BPConv %',
 'Aces/DF Ratio']
```

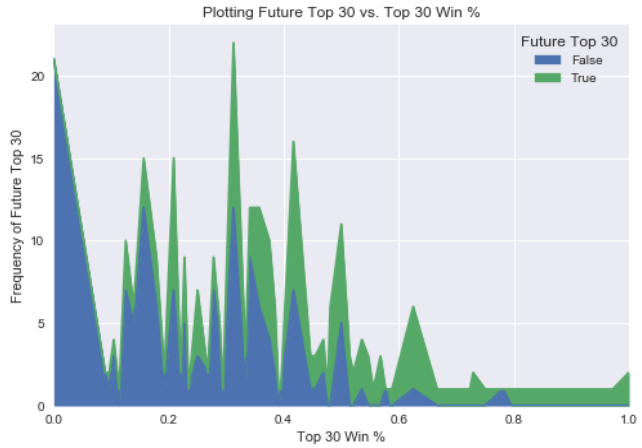
The following Figure 4. shows the distribution of Overall Win % as a function of the frequency of Future Top 30, and it is easy to see that there is a correlation between a higher win rate and success (as expected).

FIG. 4: Influence of Overall Win % on Future Top 30 Success



Similarly Figure 5. indicates a similar correlation between the Top 30 win rate and future success. This is evident from the two graphs (green and blue) being slightly shifted, indicating that players with higher win rates against higher ranked opponents are more likely to be successful in the future.

FIG. 5: Influence of Top 30 Win % on Future Top 30 Success



Additionally, it was observed that there were significant differences between the statistical values (such as mean, median and quartile ranges) of the attribute values when grouped by whether or not they influenced future top 30 success.

IV. MACHINE LEARNING MODELS

In this section we outline the machine learning models used and their relevance to the classification of player success. Also outlined are the optimization techniques used to deliver the best classification performance for the dataset using Python's machine learning library [9].

A. Logistic Regression

The Logistic Regression model uses the *logistic* or *sigmoid* function (Eq.1) to map real-valued input values to values between 0 and 1.

$$\sigma(t) = \frac{1}{1+e^{-t}} \tag{1}$$

For our prediction, the model consists of a vector of n match features, $\vec{X} = (x_1, x_2, ..., x_n)$ and a vector of $n+1$ real-valued model parameters, $\vec{\beta} = (\beta_0, \beta_1, \beta_2, ..., \beta_n)$. The n-dimensional feature vector is first projected to a real number, z (Eq.2). Thus, $\sigma(z) = \frac{1}{1+e^{-z}}$ represents the probability of a player being successful and for classification purposes we can assign a threshold (such as 0.5) to enable a binary prediction.

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \tag{2}$$

The objective of this model is to efficiently optimize $\vec{\beta}$ such that z can be mapped to the $(0, 1)$ output space, while minimizing the Logistic Loss function $L(p)$ (Eq.3). In the equation below, for each of the N tuples in the training set, p_i corresponds to the predicted value, y_i is the true value of the target class label.

$$L(p) = \frac{-1}{N} \sum_{i=1}^{N} p_i \log(y_i) + (1 - p_i) \log(1 - y_i) \tag{3}$$

The Logistic Loss function can be minimized using several methods such as Stochastic Gradient Descent, *liblinear* Algorithm, and the *newton-cg* method available in the *scikit learn* python machine learning library. The optimal solver based on exhaustive grid searching was chosen to be the *liblinear* method.

B. Multi-Layer Feed-Forward Neural Network

A Neural Network performs nonlinear regression (approximating any function) by iteratively learning a set of weights and thresholds for prediction. The network predicts an output $\vec{z} = f(\vec{X}, \vec{w}, \vec{T})$, where $\vec{X}$ is the input feature vector and $\vec{w}$ and $\vec{T}$ are the learned weights and thresholds. One of the learning methods, known as *Back-propagation*, enables the network to learn by an iterative process of optimizing a performance (or error) function P (Eq.4), where $\vec{d}$ is the desired target value.

$$P(\vec{d}, \vec{z}) = -||\vec{d} - \vec{z}||^2 \tag{4}$$

The neural network is typically structured to have one or more hidden layers to learn nonlinear associations from training data in the dataset and uses a non-linear activation function, K (such as a *sigmoid* or *tanh* function) to predict values $f(x)$, as shown in (Eq.5), where w_i is the weight of input x_i. Note that in our implementation a single hidden layer with 64 Neurons was used.

$$f(x) = K\left(\sum_{i=1}^{N} w_i x_i\right) \tag{5}$$

There are many different training algorithms, which aim to optimize the networks weights to generate the best outputs for a set of training examples. For example, the back-propagation algorithm uses gradient descent to reduce the mean-square error between the target values and the network outputs. The optimal solver based on exhaustive grid searching was chosen to be the Stochastic Gradient Descent algorithm.

C. Model Optimization and Hyper-parameter Tuning

Both models were optimized using the best-performance functions as returned by the exhaustive grid searching method. The following other parameters were tuned using the *GridSearchCV* exhaustive grid searching method. Given below is a summary of the parameters and the optimal value (in parenthesis) returned by the grid searching algorithm.

1. **Logistic Regression:** *C* (1), *solver* ('liblinear')

2. **Neural Network:** *activation*('relu'), *learning_rate_init*(0.01), *solver*('sgd')

Note C is the regularization strength, and 'sgd' refers to stochastic gradient descent, 'relu' is the rectified linear unit function, which returns $f(x) = max(0, x)$. Hyperparameter tuning was done using a training and test data set split in a 3:2 ratio, using 5-fold cross validation.

D. Feature Selection

For the logistic regression model, a Recursive Feature Elimination (RFE) algorithm was implemented, in an attempt to isolate the features having the maximum effect on the model's performance. This was performed in addition to the preliminary feature analysis in order to eliminate features that might have little or insignificant effect on our prediction accuracy. Figure 6. shows the set of 7 features that were returned by the RFE Algorithm. After trying different numbers of features, it was concluded that this set of 7 features provided the best classification performance.

FIG. 6: Features Selected using Recursive Feature Elimination

```
['Overall Win%',
 'Top 100 Win %',
 'Top 30 Win %',
 'First Serve %',
 'First Serve Win %',
 'BPSave %',
 'Aces/DF Ratio']
```

For the neural network algorithm, the features selected for the logistic regression algorithm were used as a starting point and an exhaustive grid search optimization procedure was used to isolate the features yielding the best classification performance, shown in Figure 7.

FIG. 7: Features Selected for Neural Network

```
['Overall Win%',
 'Top 100 Win %',
 'Top 30 Win %',
 'First Serve Win %',
 'BPSave %',
 'BPConv %',
 'Aces/DF Ratio']
```

Both models weight similar features heavily and the only difference in feature selection was the logistic regression model chose *First Serve %* over the Neural Network's choice of *BPConv %*.

V. MODEL COMPARISON AND EVALUATION

Python code (using the *scikit learn* library) was written to construct both the Logistic Regression and Neural Network classification models. These models were then optimized and tuned as described in the previous section and training and test sets were generated by projecting data from the selected set of 7 features. Note that all detailed code and output is available in the attached IPython notebooks.

A. Training and Test Data Performance

A 70-30 ratio of Training to Test Data was used on the models. Thus, the models were trained using 70% of the samples and the remaining 30% were used to test the models' predictive performance. Several iterations were performed and the classification result output for the Logistic regression model is summarized below in Figures 8.

FIG. 8: Logistic Regression Classification Performance on Test Set

```
LOGISTIC REGRESSION CLASSIFIER
Number of Training Samples = 192
Number of Test Samples = 83

Training model with 70.0 % of data and testing with 30.0 % of the data
Prediction Accuracy on Test Set = 0.819277108434

Full Classification Report:
             precision    recall  f1-score   support

      False       0.81      0.83      0.82        42
       True       0.82      0.80      0.81        41

avg / total       0.82      0.82      0.82        83
```

Similarly, the classification result output for the Neural Network model is summarized below in Figures 9.

FIG. 9: Neural Network Classification Performance on Test Set

```
NEURAL NETWORK CLASSIFIER
Number of Training Samples = 192
Number of Test Samples = 83

Training model with 70.0 % of data and testing with 30.0 % of the data
Prediction Accuracy on Test Set = 0.78313253012

Full Classification Report:
             precision    recall  f1-score   support

      False       0.80      0.80      0.80        44
       True       0.77      0.77      0.77        39

avg / total       0.78      0.78      0.78        83
```

It can be observed that the Logistic Regression model slightly outperforms the Neural Network with an average precision and recall of 0.82 over the 0.78 of the Neural Network. The incremental performance of both models is also summarized using Receiver Operating Characteristic (ROC) curves shown below in Figures 10 and 11.

FIG. 10: Logistic Regression ROC for Test Set

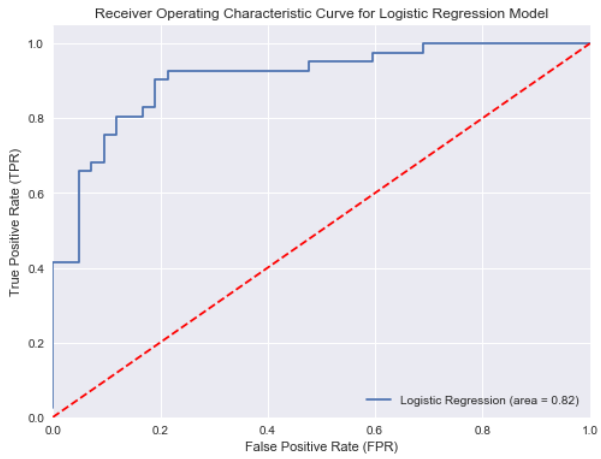
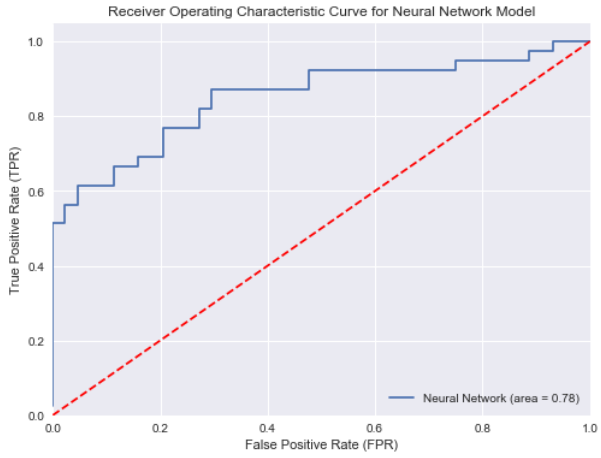


FIG. 11: Neural Network ROC for Test Set



While the Neural Network outperforms the Logistic Regression model during initial stages of classification with a higher True Positive Rate (TPR), the Logistic Regression model was again observed to have a marginally better performance as it progresses through the Test Data.

B. Cross Validation Performance

5-fold stratified cross validation was used to validate the performance of the models and check against overfitting on a specific test data set. The results of the cross-validation are summarized below, indicating both the average case performance and the best case results within the cross validation folds.

TABLE I: Comparison of Logistic Regression and Neural Network Model on Test Data

	Metric	Logistic Regression	Neural Network
1	Accuracy (best, worst)	0.757 (0.815, 0.69)	0.764 (0.833, 0.723)
2	Precision (best, worst)	0.773 (0.944, 0.69)	0.769 (0.941, 0.67)
3	Recall (best, worst)	0.733 (0.885, 0.67)	0.765 (0.885, 0.52)
4	Run Time	0.33 s	2.01 s

It is evident that the two models have very similar prediction performance, based on all three metrics. The Neural Network marginally outperforms the Logistic Regression model on accuracy and recall, and marginally under-performs on precision. In terms of the ranges of the metrics of prediction, both models had very good best case results (about 90% for precision and recall, 82% for accuracy), and the Neural Network had the smallest accuracy range with a worst case accuracy of 0.723. It is important to note that the Neural Network has an outlier worst case recall performance at 0.52, which is significantly lower than the other worst case results (which lie in the region of 0.69). In terms of run time, the Neural Network is significantly slower, as it takes about 6 times longer to run than the Logistic Regression model.

VI. ANALYSIS OF RESULTS AND SCOPE FOR IMPROVEMENT

Using the performance evaluation in the previous section as a baseline, we observe that we have created two predictive classification models that predict the future success of a tennis player with a mean accuracy of about 76% based on cross validation results. While this isn't state-of-the-art performance, it is important to recognize that the number of samples in our data set was $N = 275$, which is a limitation in terms of improving predictor performance. Additionally, it is important to recognize that the evolution of a tennis player is also influenced by random events that are impossible to model mathematically.

The models created enable an individual to learn new information about the probabilistic success of a given tennis player, which is very relevant for tennis player, coaches and fans within the community. There is work that can be done to improve this project, and listed below are some of these potential improvements:

- A larger match data set, spanning a longer time period could be used to improve the models by adding more data points. Additionally, different metrics of success in combination with several constraints (Player Age and Number of Matches) could be tested in an attempt to optimize the problem of defining/measuring player success.
- Classification accuracy improvement techniques such as ensemble methods and bagging could be

used in an attempt to generate better results. There is scope to further fine-tune and test the machine learning models.

- A more detailed feature set consisting of more than 9 features of statistics and player information could be extracted. This would enable feature selection algorithms a wider range of parameters to consider.

- This project only analyzes men's tennis match data, but it could be expanded to also include women to see if there are major similarities and differences.

- There is a lot of scope for real-world use and predictions in order to check the validity of our model, outside of a testing environment.

VII. CONCLUSION

A quantitative model was constructed to characterize and predict the success of professional male tennis players, given their performance in their first 40 matches. The (probabilistic) prediction of player success was transformed into a classification problem that was solved using machine learning methods.

The data acquisition and feature extraction processes were described in detail, highlighting the important . We outline the machine learning models used - Logistic Regression and a Multi-Layer Feed-Forward Neural Network - and their relevance to the classification of player success.

Also outlined are the optimization techniques used to deliver the best classification performance for the dataset using Python's machine learning library. The Logistic Regression model (accuracy=0.82) marginally outperformed the Neural Network (accuracy=0.78) on a test data set. In terms of stratified k-fold cross validation performance, both models performed similarly yielding a mean accuracy of about 76%, mean precision of 77% and mean recall of 75%. Note that the *pandas* and *scikit learn* Python libraries were used extensively to perform data processing and the machine learning classification. All commented code and output is available in the attached IPython notebooks.

VIII. REFERENCES

1. Kramer, Tamara, et al. Prediction of Tennis Performance in Junior Elite Tennis Players. Journal of Sports Science and Medicine, 1 Mar. 2017, pp. 1421.

2. Sipko, Michal. Machine Learning for the Prediction of Professional Tennis Matches. 15 June 2015.

3. Grove, Wayne A, and Michael Jetter. The Superstar Quest: Does Youth Talent Predict Professional Success for Female and Male Tennis Players? 1 Apr. 2014.

4. Guyon, Isabelle, and Andre Elisseeff. An Introduction to Variable and Feature Selection. Journal of Machine Learning Research 3, Edited by Leslie Pack Kaelbling, Mar. 2003, pp. 11571182.

5. Dingle, Nicholas, et al. On the (Page)Ranking of Professional Tennis Players. 2012.

6. W. J. Knottenbelt, D. Spanias, and A. M. Madurska. A common-opponent stochastic model for predicting the outcome of professional tennis matches. Computers and Mathematics with Applications, 64:38203827, 2012

7. Mlakar, Miha, and Tea Tusar. Analyzing and Predicting Peak Performance Age of Professional Tennis Players. Tea/Publications, 31 Jan. 2017, dis.ijs.si/tea/Publications/Mlakar15Analyzing.pdf.

8. Reid, Machar, et al. Rankings in Professional Men's Tennis: a Rich but Underutilized Source of Information. Taylor & Francis, Journal of Sports Sciences, 7 Feb. 2014, http://www.tandfonline.com/doi/full/10.1080/02640414.2013.876086?src=recsys&

9. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

10. Han, Jiawei, et al. Data Mining: Concepts and Techniques. Elsevier/Morgan Kaufmann, 2012.