# Machine Learning in Table Tennis Match Prediction

Sophie Chiang
Form: L6FKD
Rouse Essay Award
Supervisor: Dr Denes
Word Count: 3501

***Abstract*—Machine learning has shown to be useful in many different fields of classification and prediction. In this paper, these techniques are applied specifically to predicting the outcome of a table tennis match, and a similar idea has been applied to different sports for arranging effective training for players. Player and match statistics from data on historical matches were to predict the outcome, and 12 features were extracted and derived in an attempt to capture the quality of players and their opponents. A multi-layer perceptron neural network model was shown to be have the best overall performance.**

## I. Introduction

Table tennis is one of the most quickest and technical sports, requiring players to initiate an appropriate response within milliseconds of perceiving incoming ball placement and trajectory. Very subtle factors that are difficult for a human to recognise can impact the outcome of a game, thus leading to the application of machine learning techniques to sport result prediction, which has become increasingly popular recently in sport analytics.

Such methods have been frequently used in tennis [1] and football [2] by coaches to arrange for effective training to improve player performance. The unpredictable nature of sport also makes predicting results an interesting challenge to sport fans, but little has been done in table tennis prediction. The main reason being is the lack of match data that is available. Table tennis is very fast-paced in nature, with intense rallies having ball speeds of 60-70mph and a rotational speed of 9000rpm. The proximity between players is a lot lower compared to other sports such as tennis and badminton, thus demanding players to have very high reflexes. There is a variety of in game data that can be analysed, which has been achieved in previous studies e.g. [3] where data was collected manually, however due to the size and speed of the ball, it is difficult to determine how accurate this is.

Recent developments in areas of multi-class event spotting and tracking of small objects has meant that automated data collection from table tennis matches is now attainable. In this paper, I have used multiple classification algorithms from machine learning to model both men and women's professional singles matches. Player statistics collected from historical matches are predominantly used in predicting the outcome, with newly derived information calculated from combining player statistics. The main contribution of this paper aims to bridge the gap between data collection and predictive analysis in table tennis, to discuss relevant state-of-the art machine learning models and to report the model with the highest predictive performance using rigorous quantitative evaluation.

The rest of the paper is organised as follows: the next section explains the concepts of machine learning and reviews existing literature in sport prediction; I then describe the details of the dataset and how different features are selected and derived. I explain the different state-of-the art models chosen to make predictions, the experimental results, and evaluate the performance of models by comparing these results. Future work and concluding remarks are presented at the end of the paper.

## II. Related Work

### A. Machine Learning

Machine learning (ML) is a branch of artificial intelligence that has been successfully applied to many areas of industry and science, including disease diagnosis in medicine [4], pattern recognition [5], computer vision [6] and bioinformatics [7]. ML models are able to identify patterns and relationships in raw data that has been inputted, known as a feature vector, and can turn it into an output. This technology is referred to as a black box model, as it is difficult for a human to understand precisely how conclusions are produced. In classification, the aim is to predict a target variable (class) from a training dataset to predict the value of that class in testing instances. The training dataset is the data in which the model is fitted on, and in testing instances, the values of predictor features are known but the value of the class label is not [8]. This type of data processing is known as supervised learning, as instances are given with the corresponding correct outputs (labels), as opposed to unsupervised learning, where instances are unlabelled [9]. A model's performance can be evaluated by splitting the initial data into subsets and calculating different metrics to see how well a model generalizes to unseen data; more detail can be found in Section V.

### B. Sport Prediction

Knottenbelt et al. [10] proposed a common opponent model that yielded a pre-play estimate of the probability of each player winning a professional singles tennis match. This was achieved by analysing match statistics for opponents that both players have encountered in the past, which provided for a fair basis comparison. Subsequently, the model was able to compute the probability of each player winning a point on their serve, and hence the match. Barnett and Clarke [11] uses

historical data from past matches to predict the probability of a player winning a single point, and Clarke and Dyte [12] used a years worth of tournament results to predict the outcome of a tennis match using player rating points.

Extensive research has been done in tennis, however, there are a very limited number of works in table tennis prediction. Both are ideal sports to apply hierarchical probability models to; a table tennis match consists of a sequence of sets, which consists of a sequence of points, therefore due to certain similarities between the two sports, certain concepts can be applied from works that have been conducted in tennis.

In table tennis, a number of computer vision based approaches have been applied. Voeikov et al. [13] proposed a neural network that allowed for real-time processing of high-resolution table tennis videos. This was able to extract temporal and spatial data such as ball detection and in game events, and is potentially capable of substituting manual data collection by sport scouts, in addition to assisting with referee decision making. Zhang et al. [14] is able to compute the 3D coordinates of a table tennis ball by its image coordinates. The flying trajectory of the ball can be predicted, and hence the balls landing and striking point can be calculated, however, this study does not extend to predicting the match outcome.

Current works in table tennis mainly focus on ball detection or calculating ball speed, and predicting the result of a match has not yet been addressed, thus leading to the development of this project.

## III. Dataset

All table tennis match data was retrieved from OSAI [15]. This includes match and player statistics from Tischtennis-Bundesliga, the top professional German table tennis league, as well as men and women's singles matches from the Tokyo 2020 Olympics. Many potential features such as player age, rank, match duration as well as in-match statistics such as percentage of points won on serve and receive, stroke types and error types were available. Interactive maps that demonstrated the ball position of each shot on the table, as well as the stroke type were also accessible. For an example, see Figure 1. Samples with missing data entries were removed from the dataset.

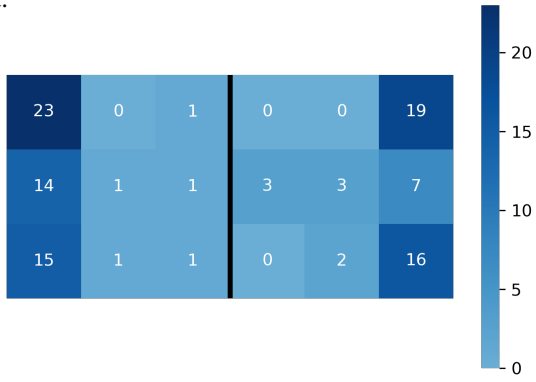


Fig. 1. Distribution of ball placement of a winning shot for both players by region if each side of the table were to be split into nine equal parts. The heatmap is plotted as if the table were being viewed from above and each value indicates the number of balls landing in it's respective region [15].

## IV. Feature Engineering & Selection

Rich data is valuable, however irrelevant or redundant variables can increase the computation time, as well as decrease a model's performance, therefore choosing appropriate features to input into a classification model is very important.

### A. Match Representation

In supervised machine learning, a set of labelled data is required for the model to train on. In the context of table tennis prediction, each match corresponded to two instances of data, one from the perspective of each player respectively, where every sample is composed of two elements:

- A vector of input features ($\underline{x}$) consisting of player and match statistics
- The target variable ($y$), indicating the result of the match that corresponds to its respective sample

The outcome of the match for player $i$ is defined as follows:

$$y = \begin{cases} 1, & \text{if } player_i \text{ wins} \\ -1, & \text{if } player_i \text{ loses} \end{cases}$$

As any incomplete matches were removed from the dataset, combined with the inability to draw in table tennis, there is no other possible outcome.

### B. Feature Engineering

In addition to the features extracted from the dataset in Section III, I combined players' statistics to form new features [11]. Using pre-existing knowledge on the sport, adding combinations of player statistics as features may improve the predictive model. These features were calculated as differences between different player statistics as this considers the characteristics of *both* players participating in a match. This was inspired by Sipko and Knottenbelt [16] and Cornman et al. [1], who both use features calculated as differences to predict the outcome of a tennis match.

In table tennis, there are only two possible outcomes of a match; a win or a loss. Unlike team sports, a combination of players with each of varying individual ability and skill level

TABLE I
Feature Summary

| Feature | Explanation |
|---|---|
| SP | percentage of total points won on serve |
| RP | percentage of total points won on receive |
| LRP | percentage of total points won on a long rally |
| SRP | percentage of total points won on a short rally |
| FHP | percentage of total points won on a forehand |
| BHP | percentage of total points won on a backhand |
| RANK | player ranking |
| RANKDIFF* | difference in rank between opponents |
| SA* | player serve advantage |
| SRA* | player short rally advantage |
| FHA* | player forehand advantage |
| BALANCE* | measure of how well rounded a player is |

do not need to be considered in the predictive outcome. Due to this, the likelihood of player substitutions nor offensive and defensive combinations do not need to be analysed.

For the final feature set with abbreviated feature names and explanations, see Table I. Newly derived features are indicated with * and more detailed explanations can be found in subsections IV-B1, IV-B2 and IV-B3. For the purposes of this paper, a long rally is defined as a rally of over five shots, and a short rally is any rally of length under five.

*1) Rank Difference:* The feature *RANKDIFF* was constructed by calculating the difference between rankings of two opponents.

$$RANKDIFF = \begin{cases} RANK_i - RANK_j & \text{for player } i \\ RANK_j - RANK_i & \text{for player } j \end{cases}$$

where $RANK_i$ and $RANK_j$ are the rankings of players $i$ and $j$ respectively at the time of the match. Therefore if a player's rank is higher than their opponent's rank, $RANKDIFF$ will be a negative value, and vice versa. However, for some match instances where the ranking of both players are above 100, the feature $RANKDIFF$ is considered to be 0. This is due to the fact that the lower the rank of a player, the more likely it is that there will be other players of a similar standard where rank doesn't accurately represent the standard of a player.

For example, players of rank 2 and rank 7 is much more likely to have an accurate depiction of their standard in comparison to two players of rank 150 and 155, despite the rank difference being the same. Therefore, if both players have a rank of below 100, the expected difference in skill level is considered to have no benefit.

*2) Serve Advantage:* The serve advantage of a player is calculated as the difference between their serve and receive winning percentage. This depicts the contrast as to how likely a player is to win a point if they are serving, compared to if they are on receive. Subsequently, the advantage a respective player has in a short rally over a long rally, as well as the advantage a respective player has in a forehand stroke over a backhand stroke, can be calculated therefrom.

*3) Balance:* An attempt to measure the *completeness* of a player can be calculated by taking the average of serve, short rally and forehand advantage:

$$BALANCE = \frac{|SA| + |SRA| + |FHA|}{3}$$

Players of a higher skill level tend to have fewer weaknesses and are stronger in more aspects of the game, therefore the feature $BALANCE$ indicates the overall well-roundness of a player's ability.

### C. Feature Scaling

Different features tend to have a varying range of values, therefore it is normal to scale features as part of data pre-processing prior to learning. *Standardization* is a scaling technique to centre values around the mean with a unit standard deviation [17], and was performed across features.

## V. Models

A number of different machine learning algorithms were used, and the *accuracy* for each model was calculated as an estimate for it's overall performance. Accuracy is defined to be the proportion of correctly predicted instances to the total number of predictions:

$$accuracy = \frac{tp + tn}{tp + tn + fp + fn}$$

where $tp$ and $tn$ are true positives and true negatives, and $fp$ and $fn$ are false positives and false negatives respectively [18]. These parameters are the different types of outcomes that can occur in a classification problem, and can be visualised in a confusion matrix, as shown in Figure 2.

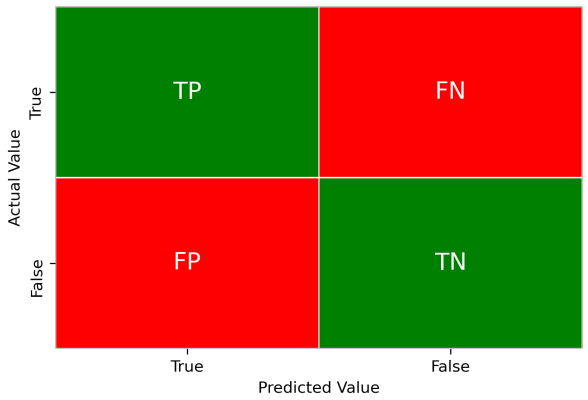


Fig. 2. Confusion matrix comparing actual to predicted values

Additionally, *precision* and *recall* were calculated for each model. Precision is defined to be the proportion of true positives to the total number of predictions predicted positive, and is a percentage of returned results which are relevant. Recall is defined to be the proportion of true positives to the total number of actual positives, and is a percentage of relevant data which have been correctly classified [19].

$$precision = \frac{tp}{tp + fp} \quad recall = \frac{tp}{tp + fn}$$

Both metrics are important to take into consideration, and ultimately a model's *F1 score* is calculated, which can be interpreted as the harmonic mean between precision and recall. This value is best at 1, and worst at 0.

$$F1\ score = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

An issue associated with training models is the possibility for the model to *overfit*. Overfitting occurs when a model captures unwanted bias and noise in the data that it negatively impacts the performance of a model. The model corresponds to it's initial training data too well, and fails to predict unseen data reliably. In order to limit overfitting, a popular re-sampling technique called $k$-fold cross validation is used.

In cross validation, the dataset is split into $k$ random subsets, known as folds, and one is selected as a test set for the model

to test on, while the others are used as a training set for the model to train on. This is repeated $k$ times where a different subset of data is used as the test set each time, and the overall performance of the model is calculated as the average of accuracy scores for each iteration [20]. This can be visualised in Figure 3.

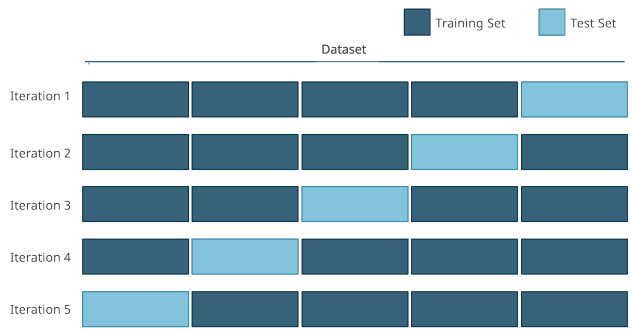


Fig. 3. How data is split for $k$-fold cross validation where $k = 5$

This paper uses Scikit-Learn's implementation to apply different models to the dataset [21].

### A. Logistic Regression

The logistic function $\sigma(t)$ is defined as follows:

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

The logistic function maps any input value $x$ where $x = \{x \in \mathbb{R}\}$ to a value between 0 and 1, allowing the output to be interpreted as a probability. If this probability is considered to be over 0.5, it can be classified as true, where the player is predicted to win the match, otherwise the result is classified as false. The model gives the best reproduction of match outcomes for the training set by minimising the *logistic loss* function:

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

$$n = \text{number of matches}$$

$$p_i = \text{predicted probability of a player winning match } i$$

$$y_i = \text{outcome of match } i$$

where a loss function measures the disparity between observations and their estimated fits [22].

### B. Random Forest

A random forest is a classifier consisting of a collection of simpler tree-structured classifiers $\{h(\mathbf{x}, \theta_k),\ k = 1, ...\}$ where the $\{\theta_k\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input $\mathbf{x}$. For the $k$th tree, a random vector $\theta_k$ is generated and a tree is grown using $\theta_k$ and the training set, to produce a classifier $h(\mathbf{x}, \theta_k)$ [23]. After a large number of trees are produced, the output of a random forest model is the class that receives the most votes. Decision trees tend to be simple to interpret and "quick" to learn, making it a popular ML technique.

### C. Support Vector Machines (SVM)

SVMs were used in predicting tennis match outcome [1]. The idea is that SVMs map an input vector in a feature space of $n$ dimensions, where $n$ is the number of features. The optimal hyperplane is identified which separates data points into two classes. This is known as the decision boundary, and the marginal distance between this boundary and the instances closest to the boundary is maximized. The existence of a decision boundary can allow for any detection of miss-classification. SVM algorithms use a set of mathematical functions that are defined as *kernels*, and different SVM algorithms use different type of kernel functions. Different kernels including linear, polynomial, sigmoid and radial basis function (RBF) were used for the purposes of this study.

### D. Multilayer Perceptron Neural Networks (MLP)

An MLP neural network is a mathematical model inspired by human neurons that consists of one or more hidden layers in between it's input and output layers, and is designed to imitate the behaviour of biological neurons in the human brain. The network consists of mutually connected artificial neurons, where neurons are organised in layers, and connections are directed from lower layers to upper layers. Neurons from the same layer are not interconnected with each other [24].

Each connection between two neurons has an associated weight, and in the process of learning and training an MLP model, these weights are adjusted such that there is a minimal difference between the model output and the desired output.

## VI. Experimental Results

The whole dataset was split in a 70:20:10 ratio such that there was a training set, validation set and test set. The model trains on the training set, and the validation set is used for optimising the hyperparameters of the model. The best combination of hyperparameters for a model is determined by whichever has the highest accuracy on the validation set using 5-fold cross validation. The test set is completely left out from the training process of the model, and is only used during evaluation. This is to see how well models generalise to unseen data which replicate new match data, as the test set is never used before evaluation. During evaluation, all other data is used for training the model (training and validation). Both accuracy and F1 score is reported for the validation and test

TABLE II
Model Performance Comparing Validation and Test Sets

| Model | Validation set | | Test set | |
|---|---|---|---|---|
| | Acc | F1 | Acc | F1 |
| Logistic Regression | 0.699±0.053 | 0.705±0.052 | 0.722 | 0.706 |
| Random Forest | 0.677±0.071 | 0.688±0.073 | 0.667 | 0.684 |
| Support Vector Machine | | | | |
| → Linear | 0.696±0.064 | 0.690±0.078 | 0.639 | 0.629 |
| → RBF | 0.700±0.056 | 0.677±0.077 | 0.667 | 0.600 |
| → Polynomial | 0.705±0.046 | 0.685±0.046 | 0.611 | 0.563 |
| → Sigmoid | 0.705±0.039 | 0.690±0.041 | 0.694 | 0.621 |
| MLP Neural Network | 0.696±0.043 | 0.708±0.045 | 0.694 | 0.703 |

sets. The standard deviation for each score for the validation set is reported as a basis of defining uncertainty; the results are shown in Table II.

## VII. Discussion

For each model, the *hyperparameters*, parameters that are not optimised by the training algorithm e.g. the number of trees in a random forest classifier, were tuned manually by using a grid search to test different values. A grid search uses brute force to search the entire space for different hyperparameter configurations.

For logistic regression, the type of solver, penalty function and $C$ value were adjusted. *Regularisation* prevents overfitting of training data by penalising large weights when training a logistic regression predictor, and the parameter $C$ mentioned is used to control the effect of this [25]. The lower the value of $C$, the stronger the effect of regularisation. For the logistic regression model used in this paper, a $C$ value of 1.0, and a 'liblinear' solver was chosen to give the best average accuracy score after hyperparameter tuning.

$l1$ regularisation penalises the sum of absolute values of weights, whereas $l2$ regularisation penalises the sum of squares of the weights [25]. $l2$ regularisation was chosen and the learning curves of the logistic regression model, where score refers to accuracy score, can be seen in Figure 4.

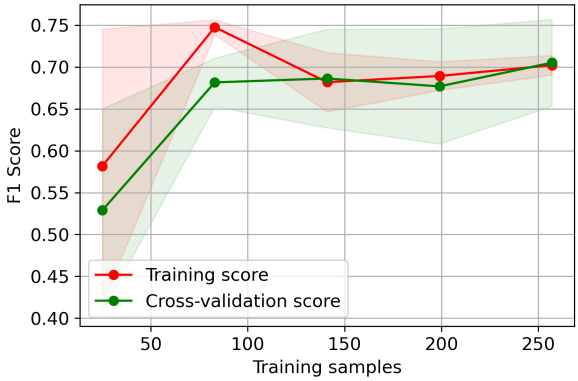


Fig. 4. Logistic Regression Learning Curve

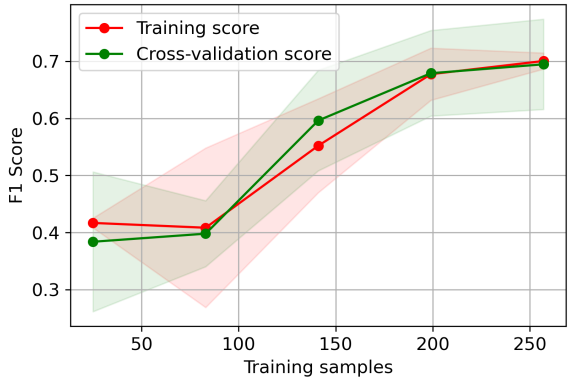


Fig. 5. Linear SVM Learning Curve

For SVMs, the two main hyperparameters that were adjusted were the kernel type and penalty value $C$. Using a linear kernel was demonstrated to have the highest F1 score on the test set compared to other kernels, and $C$ was chosen to be 0.20. The learning curve for an SVM model using a linear kernel is shown in Figure 5.

The model that achieved the highest F1 score on the validation set were MLP neural networks. The difference in accuracy and F1 score between the validation and test set was shown to be smaller in comparison to SVMs and random forest. The activation function 'relu', known as the rectified linear unit function, was chosen. This function returns $f(x) = \max(0, x)$ and defines the output of a node in a hidden layer given a set of inputs. The hidden layer size was set to 2 and the maximum number of iterations the solver iterates was chosen to be 200. The solver for weight optimization is set to 'lbfgs', an optimizer that belongs in the family of Quasi-Newton methods. The learning rate for scheduling weight updates is set to constant; more detail on different hyperparameters can be found in Scikit Learns documentation [21]. However, the generic layered structure of a neural network has proven to be time consuming. Additionally, this technique is considered a "black box" technology, and finding out why a neural network has poor performance, or how it performs the classification process, is extremely difficult [24].

One of the main advantages of using random forest is that it is much faster to train. This made tuning hyperparameters easier as training the model several times was less computationally expensive. The maximum number of levels in each decision tree was set to 80, the maximum number of features considered for splitting a node was set to 4, the minimum number of data points allowed in a leaf node was set to 4 and the number of trees that were in the classifier was set to 200.

The importance of features from a random forest classifier can also be evaluated, as shown in Figure 6. This is achieved by calculating the mean decrease in impurity for features across all trees, where *impurity* refers to the Gini impurity index. The impurity of a node is the probability of a specific feature being classified incorrectly assuming that it is selected randomly [26].

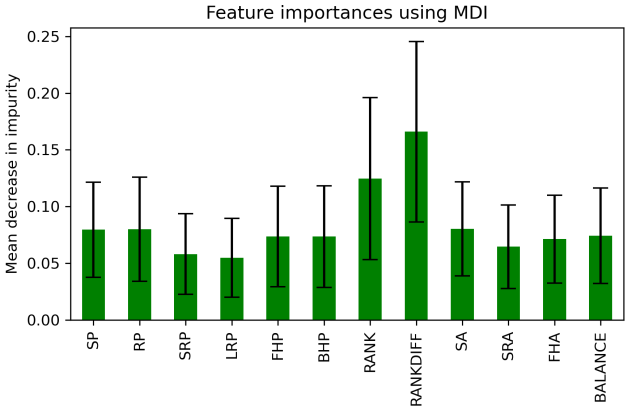


Fig. 6. Importance of features from random forest classifier

We can also compare accuracy and F1 score for each model that do not include newly derived features that were defined in Section IV-B3. All scores are lower for models that don't use newly derived features, and accuracy score is significantly lower in SVMs compared to other models. Results of the ablation study are shown in Table III.

TABLE III
MODEL PERFORMANCE WITH AND WITHOUT NEWLY DERIVED FEATURES

| Model | With | | Without | |
|---|---|---|---|---|
| | Acc | F1 | Acc | F1 |
| Logistic Regression | 0.699±0.053 | 0.705±0.052 | 0.631 | 0.668 |
| Random Forest | 0.677±0.071 | 0.688±0.073 | 0.661 | 0.673 |
| Support Vector Machine | | | | |
| → Linear | 0.696±0.064 | 0.690±0.078 | 0.556 | 0.619 |
| → RBF | 0.700±0.056 | 0.677±0.077 | 0.500 | 0.591 |
| → Polynomial | 0.705±0.046 | 0.685±0.046 | 0.500 | 0.640 |
| → Sigmoid | 0.705±0.039 | 0.690±0.041 | 0.472 | 0.642 |
| MLP Neural Network | 0.696±0.043 | 0.708±0.045 | 0.639 | 0.683 |

## VIII. CONCLUSION

In this paper, the concepts of machine learning were discussed and how the use of supervised ML methods and classification algorithms could be used in table tennis match prediction. This paper follows a state-of-the art approach in training and evaluating different machine learning models. The theory of each model and how it's performance was evaluated was explained. The original dataset retrieved from OSAI [15], in addition to deriving new features were used, and the difference in performance of models with and without newly derived features were compared. The model which achieved the highest evaluated score when presented with unseen data in the test set was by using logistic regression. From Figure 6, the feature which is shown to be the most important is $RANKDIFF$, a measure of the difference between rank in opponents in an attempt to quantify both players' skill difference. Future works could focus on a selection of the most important features established from the random forest model.

## ACKNOWLEDGEMENTS

I would like to express my gratitude towards Dr Denes for his continued support and feedback throughout this project. I would also like to thank the OSAI team for granting permission to use their data for this project.

## REFERENCES

[1] A. Cornman, G. Spellman, and D. Wright, "Machine learning for professional tennis match prediction and betting," *Stanford Unverisity*, 2017. 1, 2, 4

[2] J. Hucaljuk and A. Rakipović, "Predicting football scores using machine learning techniques," in *2011 Proceedings of the 34th International Convention MIPRO*, pp. 1623–1627, IEEE, 2011. 1

[3] J. Wang, K. Zhao, D. Deng, A. Cao, X. Xie, Z. Zhou, H. Zhang, and Y. Wu, "Tac-simur: Tactic-based simulative visual analytics of table tennis," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 1, pp. 407–417, 2019. 1

[4] K. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karamouzis, and D. I. Fotiadis, "Machine learning applications in cancer prognosis and prediction," *Computational and structural biotechnology journal*, vol. 13, pp. 8–17, 2015. 1

[5] S. M. Weiss and I. Kapouleas, "An empirical comparison of pattern recognition, neural nets, and machine learning classification methods.," in *IJCAI*, vol. 89, pp. 781–787, Citeseer, 1989. 1

[6] A. I. Khan and S. Al-Habsi, "Machine learning in computer vision," *Procedia Computer Science*, vol. 167, pp. 1444–1451, 2020. 1

[7] P. Larranaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J. A. Lozano, R. Armananzas, G. Santafé, A. Pérez, *et al.*, "Machine learning in bioinformatics," *Briefings in bioinformatics*, vol. 7, no. 1, pp. 86–112, 2006. 1

[8] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Machine learning: a review of classification and combining techniques," *Artificial Intelligence Review*, vol. 26, no. 3, pp. 159–190, 2006. 1

[9] S. B. Kotsiantis, I. Zaharakis, P. Pintelas, *et al.*, "Supervised machine learning: A review of classification techniques," *Emerging artificial intelligence applications in computer engineering*, vol. 160, no. 1, pp. 3–24, 2007. 1

[10] W. J. Knottenbelt, D. Spanias, and A. M. Madurska, "A common-opponent stochastic model for predicting the outcome of professional tennis matches," *Computers & Mathematics with Applications*, vol. 64, no. 12, pp. 3820–3827, 2012. 1

[11] T. Barnett and S. R. Clarke, "Combining player statistics to predict outcomes of tennis matches," *IMA Journal of Management Mathematics*, vol. 16, no. 2, pp. 113–120, 2005. 1, 2

[12] S. R. Clarke and D. Dyte, "Using official ratings to simulate major tennis tournaments," *International transactions in operational research*, vol. 7, no. 6, pp. 585–594, 2000. 2

[13] R. Voeikov, N. Falaleev, and R. Baikulov, "Ttnet: Real-time temporal and spatial video analysis of table tennis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 884–885, 2020. 2

[14] Z. Zhang, D. Xu, and M. Tan, "Visual measurement and prediction of ball trajectory for table tennis robot," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, no. 12, pp. 3195–3205, 2010. 2

[15] "Osai." https://osai.ai/. Accessed: 2022-01-18. 2, 6

[16] M. Sipko and W. Knottenbelt, "Machine learning for the prediction of professional tennis matches," *MEng computing-final year project, Imperial College London*, 2015. 2

[17] D. Bollegala, "Dynamic feature scaling for online learning of binary classifiers," *Knowledge-Based Systems*, vol. 129, pp. 97–105, 2017. 3

[18] G. Vanwinckelen and H. Blockeel, "On estimating model accuracy with repeated cross-validation," in *BeneLearn 2012: Proceedings of the 21st Belgian-Dutch conference on machine learning*, pp. 39–44, 2012. 3

[19] M. Buckland and F. Gey, "The relationship between recall and precision," *Journal of the American society for information science*, vol. 45, no. 1, pp. 12–19, 1994. 3

[20] D. Berrar, "Cross-validation.," 2019. 4

[21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011. 4, 5

[22] E. Hazan, T. Koren, and K. Y. Levy, "Logistic regression: Tight bounds for stochastic and online optimization," in *Conference on Learning Theory*, pp. 197–209, PMLR, 2014. 4

[23] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001. 4

[24] L. Noriega, "Multilayer perceptron tutorial," *School of Computing. Staffordshire University*, 2005. 4, 5

[25] H. Ahmadian, J. Mottershead, and M. Friswell, "Regularisation methods for finite element model updating," *Mechanical Systems and Signal Processing*, vol. 12, no. 1, pp. 47–64, 1998. 5

[26] A. P. Cassidy and F. A. Deviney, "Calculating feature importance in data streams with concept drift using online random forest," in *2014 IEEE International Conference on Big Data (Big Data)*, pp. 23–28, IEEE, 2014. 5