MasterClass(ifier):A performance predictor for MasterChef Australia© contestants

1st Warren Freeborough
Department of Computer Science and Applied Mathematics
University of Witwatersrand
Johannesburg, South Africa
ORCID ID: 0000-0003-3825-3401

***Abstract*—Entrance into the information era has led to machine learning being used in the entertainment industry, however there are no reports of machine learning techniques being used to predict the outcome of a TV show. MasterChef Australia© is a popular cooking show that is an ideal candidate to investigate whether or not the shows outcome can be predicted using machine learning, as it follows a predictable sequential structure. An accurate predictive algorithm would provide insights about the predictive nature of the competition and potentially predict who would win season 12. Three supervised classifiers: logistic regression, k-nearest neighbours and extreme gradient boost, were trained on 11 seasons of randomly over-sampled categorical performance data. The algorithms predicted each contestants performance at the end of the competition: winner, runner-up, top 5 and not failed to place in the top 5. All three algorithms performed adequately, with a weighted F1 score between 0.812-0.857, while a bagged ensemble of the three tests models performed better (0.853). It was noted that the most informative features for predicting contestant performance lay between weeks 8 and 12, with very little increase in performance observed after this period. Using the ensemble models, at the current leg of the competition (11 weeks), it is predicted that either Laura or Reynold win. While only 77.33% accurate at 11 weeks, time will tell if one of these contestants will win.**

***Index Terms*—logistic regression, KNN, XGboost, multi-class classifier, dimensionality reduction**

I. Introduction

Machine learning occurs when a process shows increased performance for a specific task, given time and experience with respect to the task [1]. Machine learning, facilitated by the emergence of the information era, displays growing popularity in various domains such as: health-care, finance and even the entertainment industry [2]–[4]. Traditionally in the entertainment industry, machine learning methods are popularly used as recommender systems that function in the background to keep viewers engaged. Arguably one of the biggest factors that determines viewer engagement is the level of predictability in the shows content. If the content is too predictable, the audience grows bored and seeks out other entertainment.

Machine learning offers the opportunity to determine whether or not a show is predictable, thus allowing the content-creators to make the necessary changes to address these issues and keep audiences engaged. However, very little work has been done with machine learning to analyze entertainment data to determine whether the content is predictable. Furthermore, an accurate algorithm that predicts the outcome of TV shows would mean that a viewer would not need to spend as much time watching the show, thereby freeing up time to pursue other activities, such as studying machine learning algorithms. Competitive cooking shows are good candidates for this study as they follow a periodic, predictable structure with only a few possible outcomes, namely would win the competition

Masterchef Australia© is one of the most popular competitive cooking shows in circulation, with an average of 1.42 million viewers each premier. Currently in its twelfth season, contestants are expected to compete in various cooking challenges (such as innovation, mystery box, immunity and elimination challenges) each week, whereby judges rate the quality of their cooking. Failure of these challenges may result in elimination of the contestant from the competition, which continues until only one contestant remains.

Particularly in these type of shows, viewers are constantly predicting who may win. There is value to be had in investigating whether a supervised learning strategy can accurately predict the outcome of the competition (first, runner-up, in the top 3 or top 5). In doing so, this paper aims to answer the question of how predictable the series is, and who is predicted to win the currently unresolved season 12.

The algorithm should function so that when provided with a contestants categorical performance data for various challenges, it will output the predicted outcome for the contestant at the end of the competition.

II. Related Work

To date there has been no publications that go about trying to predict the outcome of a cooking competition. This means that there is no background to draw from regarding benchmarks in this field or feature selection. However, there are numerous publications that aim to predict the outcome of various sports competitions. Despite tackling completely separate subject matter, these publication may be informative regarding what type of algorithms they used and possibly used as a benchmark for this study.

Ofogh *et al*. applied both supervised (Naive Bayes) and unsupervised (K-means) multi-class classifiers to predict the placement of cyclists competing in the Olympic Track Cycling Omnium [6]. Similar to the research design of this study, Ofogh *et al*. aimed to predict whether cyclists would place

in the top 3, between 3-10 and those who would place after 10. This prediction was used to provide narrative regarding which features are strong determiners for medal winners.

Despite having a small dataset (< 100 records), the authors managed to demonstrate a high degree of accuracy from their Naive Bayes: weighted F1 score of 0.839 for male and 0.865 for female racers. Interestingly, they found considerable overlap between the supervised and unsupervised results, showing that there is some use in clustering methods to predict performance. Using their model, the authors concluded that winning a medal is dependant on both sprint and endurance and not just one of those skills.

Following a similar research aim, Lin *et al*. investigated whether supervised classification models could be more accurate in predicting the winners of basketball games compared to industry experts [7]. The data they used for training spanned over seven seasons between 1991-1997, and the features pertained to game statistics such as number of points, steals, and blocks. In total, five different techniques were investigated: logistic regression, Adaboost, random forest, Gaussian Naive Bayes and a support vector machine

Although there was not much difference between the accuracy of the different methods (0.633-0.651), the authors we not successful in beating the industry experts predictions (0.71). Furthermore, the authors also showed that the accuracy of the models tend to increase with the number of games played over the course of the season, with exception to random forest. Lastly, it was concluded that the running tally of wins for a team was a significant factor in the likelihood for the team to win.

III. Dataset and features

The data is freely available on wikipedia (https://en.wikipedia.org/wiki/MasterChef_Australia) where a table provides a categorical outcome for each challenge per contestant per week over the course of 11 seasons. However, there were two problems with this raw data that required attention. Firstly, the categorical outcomes were at times too specific, which led to redundant entries in the data. For example: the outcomes "Top 2" and "Top 4", despite being different, refer to the same fact that the contestant performed well the challenge. The second problem, relates to the fact that the competition structure was not uniform between season. To address these issues, the data underwent pre-processing so that all season had the same number of features and the categorical outcomes were changed so that only broad, widely applicable outcomes remain (Table I).

TABLE I
Table of the categorical outcomes found in the pre-processed data and their various meanings

Challenge outcome	**Interpretation**
Btm	Contestant placed bottom in group
Elim	Contestant eliminated from competition
IN	Contestant competed in challenge
Immune	Immune to competing in challenge
Lose	Contestant lost the challenge
None	Contestant did not compete in the challenge
Team 2nd	Contestants team came second
Team 3rd	Contestants team came third
Team Lose	Contestants team lost
Team Win	Contestants team won
Top	Contestant was one of the top in the challenge
Win	Contestants won the challenge

A. Pre-processing

Once pre-processing was performed, a dataset consisting of 257 entries with 68 features. The features of this dataset represents one of the four challenges at each week of the competition, spanning 17 weeks total. When a specific challenge was absent during that stage of the competition, or if a participant did not compete for reasons not relating to elimination from the competition, the "None" outcome was used. The label for each contestant was how well they performed in the competition. The possible outcomes were "Winner","Runner-up","Top 5" and "Elim" which denotes that the individual was not apart of the final five contestants. These four labels were represented as 0,1,2 and 3 respectively in future confusion matrices.

The last step required was to change the format of the data so that it can be subjected to supervised learning methodology. This is done by implementing one-hot-encoding on the features and labels, whereby it converts the categorical data to equally weighted numerical data. Each entry becomes a unique series of zeros and ones, leading to an increase in the number of features based on how many possible outcomes exist in the data. Once implemented, the number of features in the data grew to 816 and was now ready for splitting into the various datasets.

B. Splitting data, oversampling and dimensionality reduction

There are two distinct parts to methods: benchmarking the classification models and prediction of the season 12 contestants. Regarding the benchmarking of the algorithms, the data was randomly divided into three parts; training (50%), cross-validation(25%) and testing(25%) sets. Whereas for predicting the outcome for season 12, only the first 11 weeks from all 11 seasons were used to train the model. The reason for this choice is to determine if the current unfinished season 12 results can be predicted using knowledge of all previous 11 seasons.

There are two issues that prevent using the training datasets as they are, namely: the training dataset is too small (129 data points) and there exist a large degree of class imbalance in the data. For example: the First and Runner-up classes accounts for less than 1% of the total data. To address both these issues, random oversampling was performed using the Synthetic Minority Over-sampling Technique (SMOTE) algorithm [5]. The algorithm works by finds the k-nearest neighbours between data points of the same class label,before sampling randomly along this linear combination between two

points until all classes are equal in number. Following over-sampling, the training sets contained 2000 datapoints for each class.

Lastly, principle component analysis (PCA) was performed to reduce the dimensionality of the data following one-hot-encoding. A low-rank approximation was found to account for 95% of variation using the 72 largest eigenvalues. This reduced dataset was used to in both the logistic regression and the k-nearest neighbors (KNN) algorithms. For the extreme gradient boost (XGBoost) the non-reduced one-hot-encoded dataset was used for training.

IV. Methods

To determine an optimal strategy for predicting the competition outcome, three different supervised classification algorithms were investigated, namely logistic regression, KNN and XGboost.

A. Logistic regression

Logistic regression makes a prediction by incorporating the dot-product between the parameter (Θ) vector and the feature vector (X) in a sigmoidal function (Eq. 1) [8]. This condenses the output space between [0,1], where traditionally a prediction is made based on the output surpassing a specified threshold. In the case for multi-class logistic regression, the prediction is made based on which index in the one-hot encoded label produces the largest output.

$$h(x) = \frac{1}{1 + exp(\Theta^T X)} \tag{1}$$

To learn the optimal parameters for the logistic regression, the gradient decent algorithm is applied to minimise the least-squares error with regularisation:

$$\frac{1}{2N}\sum_{i=1}^{N}(h(x^{(i)}) - y^{(i)})^2 + \frac{\lambda}{2}\sum_{k=1}^{d}\theta_k^2 \tag{2}$$

where λ is the regularisation parameter, and d is the dimension. Gradient decent was performed until convergence in θ was reached, where: $max|\Theta^n - \Theta^{n-1}| < 5 \times 10^{-4}$. Hyper-parameter tuning was performed on the learning rate, α, using the cross-validation dataset. The optimal alpha was chosen on the basis of the highest weighted F1 score. Although, regularisation was implemented in the learning algorithm, it was found that scaling the regularisation parameter, λ, did not effect accuracy in the cross-validation data.

B. KNN

The KNN algorithms function by first calculating the Euclidean distance between all training data and the unknown datum (Table II). The datum is assigned label $\hat{y}$, where $\hat{y}$ is mode of the labels for the kth closest data-points to the unknown datum [10].

In doing so, the KNN aims to learn parameters k to that it is to minimise the loss function,

$$\frac{1}{N}\sum_{n=1}^{N}[h(x) \neq y] \tag{3}$$

.

TABLE II
Pseudocode for KNN algorithm

Input (X,Y,z), where X=Training data, Y=Class labels for X ,z = unknown
Begin:
For i=1 to N do:
d(X_i, z):=$\sqrt{\sum_{i=1}^{N}(X_i - z_i)^2}$
Find set I containing indices of k smallest distances d(X,z)
$\hat{z} :=$ mode(Y$_i$ where $i \in I$)
return $\hat{z}$
End
Output: Predicted label for z

C. XGboost

The XGBoost algorithm is a decision tree-based approached coupled with a boosting strategy to improve accuracy of predictions [11]. The strength of this method is using many weak decision tree learners to build an accurate prediction through increasing the weighting of incorrect predictions when building subsequent tress. Predictions are made through

$$\hat{y}_i = \sum_{k=1}^{K} f_k(X_i) \tag{4}$$

where $f(x) = w_{q(x)}$. In this function q represents the tree structure that maps an example to the correct leaf node. While each f_k refers to a tree with structure q and w weights. This weighting is done so examples that are predicted wrong are scaled to give more weight in subsequent trees. XGBoost learns by minimising the differentiable loss function (l):

$$\begin{gathered} L(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \\ \text{where } \Omega(f) = \gamma T + \frac{1}{2}||w||^2 \end{gathered} \tag{5}$$

The learning function is regularised using the regularisation constant Ω, which adds a penalty based on the models (decision trees) complexity. The T refers to the number of leaf nodes in the tree. Implementing the regularisation, leads to a bias in selecting for simpler predictive tree models.

V. Experimental Results and Discussion

A. Performance metrics

Since the class imbalances existed in the starting data, this would mean that the imbalance would be preserved in all other datasets that were randomly sampled from it. For this reason, using the standard performance metrics would over-estimate the accuracy of the models being tested. To address this, weighted performance metrics, namely: precision (Eq. 6), recall (Eq. 7) and F1 (Eq. 8) were adopted to account for the class imbalances.

$$\text{Weighted Precision} = \frac{\sum x_i^{(n)}}{\sum x_i} \times \left(\frac{True\oplus}{True \oplus + False\oplus}\right) \tag{6}$$

$$\text{Weighted Recall} = \frac{\sum x_i^{(n)}}{\sum x_i} \times \left(\frac{True\oplus}{True \oplus + False\ominus} \right) \quad (7)$$

$$\text{Weighted F1} = \sum_{n=1}^{4} \frac{\sum x_i^{(n)}}{\sum x_i} \times \left(\frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \right) \quad (8)$$

These metrics are similar to their unweighted counter-parts, however the metric for each class is scaled by the proportion of the class in the tested dataset. This performance metric was chosen over unweighted measures as it provides a better representation of accuracy for imbalanced datasets.

B. Logistic regression

For logistic regression, the optimal learning rate was determined by iteratively running the model on the training data prior to testing the weighted (F1) accuracy on the cross-validation set (Fig.1). The α that produced the highest weighted F1 score was 0.122. This alpha was used in all subsequent training of logistic regression models. In addition, the regularisation constant underwent hyper-parameter tuning, however it was found that varying parameter did not change the accuracy in the cross-validation data(not shown). It is believed that the dimensionality reduction from the PCA removed uninformative features from the dataset which prevent the need for regularisation. Consequently, all logistic models were trained without regularisation.

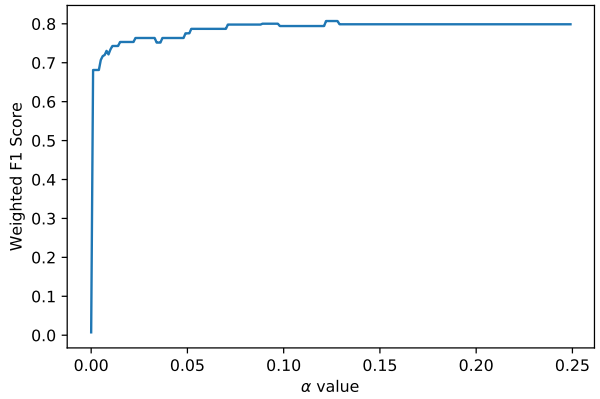

Fig. 1. Weighted F1 accuracy score for result prediction in the cross-validation set with respect to a changing learning rate (α)

The logistic regression classifier performed the best with a weighted F1 score of 0.857 (Table III). Although it performed well in predicting which contestants would be eliminated early in the competition (3), the model had particular trouble in correctly predicting whom would place second (1). This could be due to the fact that there is high degree of overlap between data classed as Runner-up and Top 5 (2).

TABLE III
CONFUSION MATRIX OF THE PREDICTED TEST DATA GENERATED FROM THE LOGISTIC REGRESSION MODEL

	Predicted			
Actual	**0**	**1**	**2**	**3**
0	2	2	0	0
1	1	1	1	0
2	0	2	9	0
3	0	0	4	42
Precision	0.884			
Recall	0.843			
F1 Score	0.857			

Although the model was only able to accurately predict half the contestants who came first (0), it is note-worthy that the wrong predictions for this class were solely predicted as coming second. This means that the model has trouble distinguishing between the first and runner-up classes, potentially indicating problems in sensitivity. This notion is further supported when considering that the model performed better in distinguishing between top 5 and the eliminated classes.

C. KNN

The K parameter was the only parameter that required hyper-tuning for the KNN algorithm. The optimal k is selected based on

$$\arg\min_k |\text{Training error} - \text{Cross-validation error}|. \quad (9)$$

where

$$error = \frac{1}{N} \sum_{n=1}^{N} [h(x^{(n)}) \neq y^{(n)}] \quad (10)$$

. It was determined that the optimal K that produced the smallest difference between the two datasets was 108. Using this K, the label for the testing data was predicted against the surrounding training data in subsequent modeling.

Upon initial inspection it appears that the KNN model performed in a similar manner to that of the logistic regression(Table 2). However, the logistic regression model performed better overall, beating the KNNs weighted F1 score of 0.812. There decrease in performance KNN can be contributed to the model reduced ability to classify the contestants apart of the Top 5 class. The likely reason for this, is that the Top 5 class sits in between both the Runner-up and the Elim class. Meaning there is likely to be a few datapoints that sit on the boundary which has more neighbours in the neighbouring class than the correct class, leading to incorrect predictions.

It appears that, much like the logistic regression, the KNN model also struggles with making predictions in the Winner and Runner-up category. This means that the KNN also suffers from sensitivity issues in making predictions among those contestants.

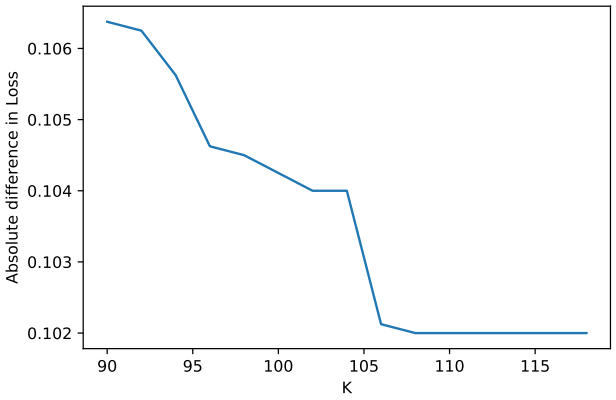

Fig. 2. Absolute difference between the training and cross-validation error with respect to changing K

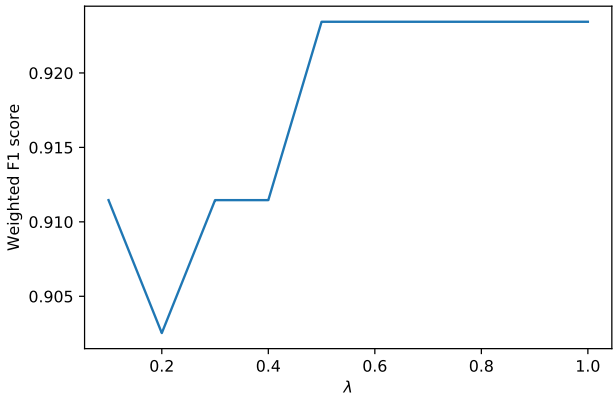

Fig. 3. Weighted F1 score from the prediction made using the cross-validated data with respect to the change in the regularisation constant

TABLE IV
CONFUSION MATRIX OF THE PREDICTED TEST DATA GENERATED FROM THE KNN MODEL

	Predicted			
Actual	**0**	**1**	**2**	**3**
0	2	2	0	0
1	2	1	0	0
2	0	2	6	3
3	0	0	3	43
Precision	0.827			
Recall	0.813			
F1 Score	0.812			

TABLE V
CONFUSION MATRIX OF THE PREDICTED CROSS-VALIDATION DATA GENERATED FROM THE XGBOOST MODEL

	Predicted			
Actual	**0**	**1**	**2**	**3**
0	0	0	1	0
1	0	1	3	0
2	0	0	6	0
3	0	0	0	53
Precision	0.947			
Recall	0.938			
F1 Score	0.923			

D. XGBoost

Much like the logistic regression, the XGBoost model required tuning of the hyper-parameter since both utilizes regularisation. In the case of XGBoost, the model was iteratively trained using the training data before prediction was performed on the cross-validation data. The optimal (0.5) was chosen based on the highest weighted F1 score and was used in all subsequent training of the model(Fig.3). Interestingly, the XGBoost model performed far better compared to any other model on the cross-validation dataset (0.923).

However, when examining the resulting confusion matrix from the cross-validated data, it is clear as to why this is the case (Table V). During the random splitting of the data, it appears that the cross-validation dataset received more data from the Elim class, and less so from the Winner class. This fact, coupled with the high recall for the Top 5 class lead to an inflation in the weighted accuracy in the cross-validation dataset.

Overall, the XGBoost model performed better than the KNN, with a weighted F1 score of 0.815 (Table VI). Compared to the previous models, the XGBoost confusion matrix is most similar when compared to the matrix of the logistic regression. However, it appears that incorrect predictions, made by the XGBoost model, do not tend to localise around the correct prediction class. This is apparent for the Winner and Top 5 classes, whereby the model makes more incorrectly predicts a class that does not neighbour the actual class. A possible reason for this is that the model was trained on data that has not undergone dimensionality reduction. This would mean that the inclusion of many uninformative features, brings with it increased variance due inclusion of additional noise. The decision to opt for the non-reduced data relates to the fact that using the reduced data lead to a reduction in the overall accuracy of the model (not shown).

TABLE VI
CONFUSION MATRIX OF THE PREDICTED TEST DATA GENERATED FROM THE XGBOOST MODEL

	Predicted			
Actual	**0**	**1**	**2**	**3**
0	2	1	1	0
1	1	0	2	0
2	1	1	8	1
3	0	0	4	42
Precision	0.825			
Recall	0.813			
F1 Score	0.815			

E. Bagging results

Bagging is a useful strategy known for increasing the accuracy of predictive models [12]. When implemented with the three

classification models, the weighted F1 score was comparable to the logistic regression (0.853) (Table VII). Although the results are similar, the bagging results are preferential as the bagged model now correctly predict the winner in most instances. Furthermore, it appears that the increased variance from the XGBoost model is not maintained in bagged confusion matrix. However, it appears that the model still has sensitivity issues in being able to accurately discriminate between contestants who won the competition and those who came second. Due to the preference in being able to accurately identify contestants whom placed first, the bagged model was used to predict the season 12 winner.

TABLE VII
CONFUSION MATRIX OF THE PREDICTED TEST DATA USING A BAGGING STRATEGY FROM THE PREVIOUS THREE CLASSIFICATION MODELS

	Predicted			
Actual	**0**	**1**	**2**	**3**
0	3	1	0	0
1	2	1	0	0
2	0	2	8	1
3	0	0	4	42
Precision	0.866			
Recall	0.844			
F1 Score	0.853			

F. Season 12 predictions

In a real world setting, when making a prediction in this sort of competition, one would not have access to the entirety of outcomes over the course of the competition. Realistically, prediction would be made before the competitions conclusion, whereby the predictive accuracy would increase as more contestants get eliminated. When training the model with partitioned data corresponding to the end of each week of the competition, this appears to be the case (Fig. 4).

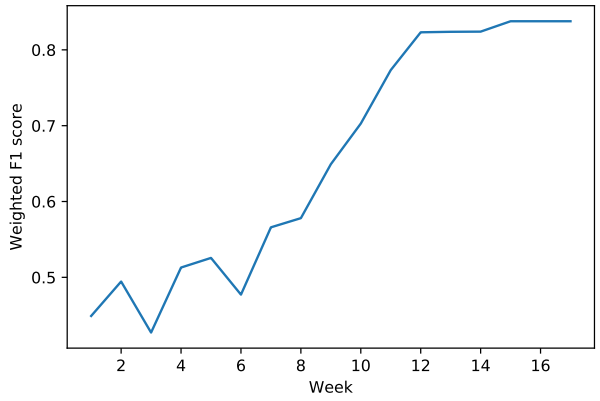

Fig. 4. The weighted F1 score of the bagged classification model using partial data to simulate prediction at each week of the competition

At the beginning of the competition, accuracy is low and but tends to rise and fall in the first few weeks. This is likely due to the competition favoring team challenges in the beginning, which is less informative than a single contestants winning a challenge. Interestingly, from the seventh week of the competition, the accuracy of prediction begins to dramatically increase.

After 12 weeks of the competition, the accuracy begins to plateau, suggesting that most of the informative features lie between 8 and 12 weeks. This would suggest that having the competition go on for longer than 12 weeks offers no predictive benefit. It seems that the content creators are aware of this, as most MasterChef Australia© seasons run for 12 weeks. Based on this, the recommendation be made that future seasons should not exceed 12 weeks or risk wasting money in production, while providing little information required by audiences to predict who may win.

As of June 2020, following 11 weeks of competition, the twelfth season of Masterchef Australia© has yet to end. This provides an opportunity to test the predictive power of the bagged models in a real world setting. After 11 weeks, the accuracy for the bagged models was found to be 0.773. Although, it would be preferential to make a prediction after the twelfth week of the competition, this level of accuracy should provide a good estimate (Table VIII).

On initial inspection, the results for season 12 look promising overall. All those who were eliminated from the competition up to this point (except for Tessa) were correctly predicted to do so. Since the algorithm predicted two contestants (Laura and Reynold) to be first, it is likely that if one were to win, the other would be the runner up.

However, there are some problems with this prediction. Notably, for first and runner-up positions, there were two contestants predicted for each category which can not be the case. Additionally, they predicted three contestants would place in the top five, however for one of the contestants, this would be an incorrect prediction. Lastly, the contestant Tessa was predicted to be placed second despite being the most recently eliminated individual. From these results, it would be safer to predict that Laura, Reynold and Emelia would be last three remaining contestants with either Laura or Reynold, winning the competition.

TABLE VIII
PREDICTED PLACEMENT OF SEASON 12 CONTESTANTS BASED ON BAGGED RESULTS FROM THE THREE CLASSIFICATION MODELS

Placement	**Contestant**
First place	Laura Reynold
Runner up	Tessa Emelia
Top 5	Callum Poh Reece

G. Comparison to related literature

As previously noted, there are no other studies that focus on competitive cooking winner prediction, making comparison to

surrounding literature difficult. However, there is still some benefit to be had in comparing these bagged results to related literature.

In terms of methodology, the study done by Lin *et al*., shares the most parallels to this study [7]. They test several models including: logistic regression and Adaboost, which shares many similarities to XGboost. Overall, this study achieved greater accuracy in test predictions (0.812-0.857) compared to Lin *et al.,*(0.56-0.69). In concurrence to this study, the authors found that performance for Adaboost and logistic regression increased as the season continued. Notably, even with a full season of data (80 games), their models still had a lower predictive accuracy compared to this bagged model at 10 weeks. It is due to this higher accuracy at an earlier stage in the competition, that leads to the conclusion that the presented bagged model performs better than those presented by Lin *et al*.

In contrast, the results published by Ofoghi *et al*. (0.839-0.865) are closer in terms of weighted accuracy to those presented here [6]. However it should be noted that the categories used by Ofoghi *et al*. are considerably broader than those used in this study. In fact, it suspected that if the class labels in both studies were the same, it is likely that the bagged model presented here would outperform the model presented by Ofoghi *et al*. Given the fact that the accuracy of this bagged model is comparable to those presented by Ofoghi *et al*, despite provided higher resolution in prediction (first and second vs top 3), it can be argued that this model is the better performing between the two.

VI. Conclusion and Future Work

In this study, three different supervised learning strategies were investigated with the aim to see if the outcome for Masterchef Australia© could be prematurely predicted. This was done as a proof of concept as well as to provide narrative into the predictability of the show. It was found that all three strategies provided sufficiently accurate predictive power ($>$0.8), while the results from ensemble model proved most desirable in predicting a winner. Furthermore, it was observed that most of the informative features occur in the 8-12 week range. This prompts the recommendation that the competition should not last longer than 12 weeks. This recommendation is leveraged on the fact that the cost of production is yielding very little return in the audiences ability to make a prediction who the winner may be, which may discourage viewership after this point.

Subsequently, the bagged models were used to predict the winner of the current (June 2020) unfinished season 12. The bagged models have an weighted accuracy of 0.772 with 11 weeks of competitive data. The results predicted that Laura or Reynold would place first in the competition followed by Emelia. Although, confirmation of these prediction would have to occur over the course of the competition.

Moving forward, additional supervised learning algorithms will be included to the ensemble to further increase accuracy and hopefully break tied predictions. Particularly, recurrent neural networks may offer a better inclusion as this algorithm functions well with sequential data like that used in this study.

References

[1] T. Mitchell, Machine learning, McGraw-hill New York ,1997, pp. 2–3.

[2] I. Kononenko, "Machine learning for medical diagnosis: history, state of the art and perspective", Artif. Intell. Med. vol. 23, 2001, pp 89–109.

[3] JB. Heaton, NG. Polson, and JH. Witte, "Deep learning in finance",arXiv preprint,arXiv:1602.06561, 2016.

[4] J. Bennett, and S. Lanning, "The netflix prize",Proceedings of KDD cup and workshop, 2007, pp 35.

[5] NV. Chwala, KW. Bowyer, LO. Hall and PW. Philip, "SMOTE: synthetic minority over-sampling technique",J. Artif. Intell. Res., 2002, pp 321–357.

[6] B. Ofoghi, J. Zeleznikow,C. MacMahon, and D. Dwyer, "TA machine learning approach to predicting winning patterns in track cycling omnium",IFIP International Conference on Artificial Intelligence in Theory and Practice, 2010, pp 67–76.

[7] J. Lin, L. Short and V. Sundaresan, "Predicting National Basketball Association Winners",CS 229 FINAL PROJECT, 2014.

[8] S. Menard "Applied logistic regression analysis",Sage, Vol. 106, 2002.

[9] E. Fix, "Discriminatory analysis: nonparametric discrimination, consistency properties",USAF School of Aviation Medicine, 1951.

[10] B. Tay, JK. Hyun, and S. Oh "A machine learning approach for specification of spinal cord injuries using fractional anisotropy values obtained from diffusion tensor images",Comput. Math. Method. M, 2014.

[11] T. Chen, C. Guestrin, "Xgboost: A scalable tree boosting system",Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016.pp785–794

[12] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants",Mach. Learn., Vol.36, 1999.pp105–139