Comparison of Machine Learning Algorithms to Predict Football Match Outcomes

Ms Maria Joseph, Student, St. Teresa's College, Ernakulam, India, and
Ms Nithya A B, Assistant Professor, Department of Computer Applications, St. Teresa's College, Ernakulam, India

***Abstract*—**

Football, being one of the most popular sports in the world, has been open to the betting and gambling world to try and predict the outcomes of its matches. Yet football is a very complex sport and is notoriously hard to predict. In this study, we attempted to predict the football match outcomes of the Indian Super League team, Kerala Blasters, while comparing three algorithms, Random Forest, Logistic Regression and k-Nearest Neighbors (KNN) to understand which model and features work best across datasets. These algorithms were applied to different sets of features, pre-game and in-game features, to determine which led to the most accurate and precise results. We also applied artificial sampling to some models to test if a more balanced dataset found better results. Results showed that Logistic Regression without using SMOTE achieved the highest accuracy (75%) and precision (42%) compared to the other models. Thus, our findings suggest that Logistic Regression is the best model for this dataset. Future work might extend the comparison to more models and apply oversampling techniques to models that don't work as well with unbalanced datasets.

Keywords: Football, Machine Learning, Random Forest, Linear Regression, KNN

I. Introduction

Football, being one of the most widely followed sports, has had vast improvements in detailing match and performance statistics. This huge data growth has led to the advent of match and player prediction and analysis through machine learning techniques. Accurate prediction of match results has practical relevance in sports analytics, team performance evaluation, and decision-making, making it an important area of study for data science and machine learning applications [1].

Traditional methods of prediction using statistical models such as Poisson-based goal models [2] are not always effective, as they struggle to model nonlinear and complex relationships influencing football matches. Machine learning techniques are able to explore patterns in a way that traditional statistical methods may not be able to, using historical data and have shown promising results in predicting football match outcomes.

Various machine learning algorithms, including logistic regression, decision trees, random forests, support vector machines, and neural networks, have been applied to football match prediction. However, there is no single 'best algorithm' that outperforms the others across datasets. Thus, a comparison study using the same experimental set up is essential to determine the best algorithm for the given dataset.

This paper focuses on predicting the match outcomes of a team in the Indian Super League (ISL), a league that garners relatively less attention compared to European leagues. The focal team, Kerala Blasters, is a popular and consistent performer in the ISL. Eight seasons of historical match data relating to Kerala Blasters is used to train three models- Random Forest, Logistic Regression and KNN, using a common set of features and standardized evaluation metrics.

The main objectives of the paper is to compare the performance of these three machine learning algorithms (Random Forest, Logistic Regression and KNN) and determine which methods are most effective for predicting match outcomes involving the Kerala Blasters. By conducting a systematic comparative analysis, this project aims to demonstrate the practical application of machine learning techniques in football analytics and to provide insights relevant to emerging football leagues such as the ISL.

II. Literature Review

The prediction model used in [3] was generated by logistic regression using the training data taken from the video game FIFA from the 2010-2011 season until the 2015-2016 season from Barclay's Premier League. The variables of match results that were "Home Offense", "Home Defense", "Away Offense",and "Away Defense" were features that were used to learn and predict sports match results. The highest accuracy of experiments by altering seasons of training data was 69.5%.

The paper [4] compared Naive Bayes, Logistic Regression and Random Forest using match and player statistics of 11 teams in the European football league, while also comparing real world statistics from 2 sources with video game statistics. They found Logistic Regression to be the most accurate with 63% accuracy.

The paper [5] compared 10 models with varying features of match and player statistics using 4 seasons of the English Premier League taken from Sofifa and found Random Forest to be the most accurate model with 65.26% accuracy. The paper [6] applied 3 seasons of match data taken from the English Premier League to a Random Forest model and was able to more than 70% accuracy and precision.

.

III. METHODOLOGY

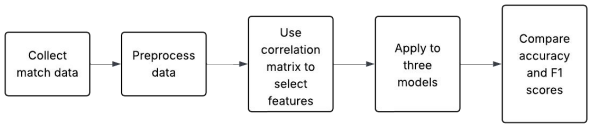

Fig. 1. Block diagram of methodology

A. Dataset

The dataset comprises of 165 observations of Kerala Blasters football match data from the Indian Super League from seasons 2017 to 2024, collected from the football statistics website, 'fbref.com' [7]. The dataset describes the match's pre-game and in-game statistics with 21 features. Pre-game features refer to statistics known to us before the match actually begins, like its date, time and venue. While in-game features refer to statistics that are made during the match's run and are summarized once the match ends, like the players in the match, penalties and shots made.

One of the major drawbacks of the dataset is that it shows a heavier class weightage for the target variable "lose" compared to "win", which was an imbalance of 112 losses and 53 wins i.e. a 2:1 class imbalance.

B. Data Preprocessing

The target variable selected was Result, having two outcomes, wins and losses (including draws). Feature engineering techniques like encoding were used for categorical data and missing data was substituted using mean imputation.

17 features were selected for a correlation matrix to understand their relation to the target variable (Fig 1). 12 features were selected from these for the models, pre-game features namely day, hour, venue, opponent, captain; and in-game features which were rolling averages from 3 to 5 previous matches namely, goals for, goals against, shots, shots on target, penalty kicks made, penalty kicks attempted. The resulting dataset was split 85% for training and 15% for testing.

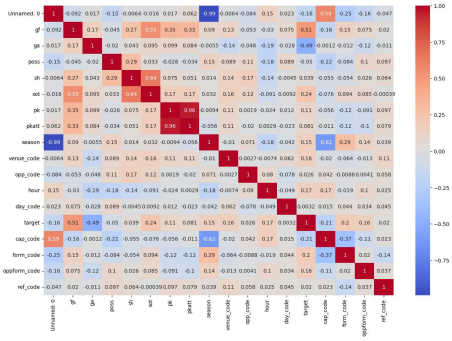

Fig. 2. Correlation matrix of features of football match

C. Model Architecture and Training

All the models were run on Google Colab with the support of libraries such as NumPy, Matplotlib, and Sciki-Learn [8] to run the models and assist with visualisations and model evaluation:

1) **Random Forest**

The Random Forest classifier is an ensemble learning method widely used for classification tasks. It operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. This architecture aims to mitigate the overfitting issues often associated with individual decision trees and improve overall accuracy and robustness.

The Random Forest classifier was implemented as an ensemble of 50 decorrelated decision trees with 1 random state. The model was trained first on just the pre-game features (3 features), and then both pre-game and in-game features (a total of 12 features). For each tree, a bootstrap sample of size n (equal to the training set size) was drawn with replacement. Nodes were recursively split until purity or the minimum sample threshold was reached. Predictions were aggregated via majority voting across trees, with OOB samples (approximately 37% per tree) used for internal error estimation:
$OOB_{error} = \frac{1}{N}\sum_{i=1}^{N} I(y_i \neq \hat{y}_{OOB,i})$, where I is the indicator function.

Two models were run with this algorithm: the first using only the pre-game features and the second with both pre-game and in-game features.

TABLE I
SPECIFICATIONS OF RANDOM FOREST ALGORITHM

Parameter	Value	Description
Number of trees	50	Ensemble size
Features per split	√p	Random feature selection
Max depth	None	Full growth, no pruning
Min samples split	10	Minimum per child node
Bootstrap samples	Yes	Bagging fraction

2) **Logistic Regression**

The developed model is designed for predictive analytics, specifically addressing a classification task within a dataset characterized by class imbalance. The architecture integrates robust data preprocessing techniques, advanced feature engineering, and a strategy for mitigating class imbalance, culminating in a supervised learning algorithm for prediction.

A Logistic Regression classifier was employed for binary classification, modeling the probability of the positive class via the sigmoid function:
$P(y=1|\mathbf{x}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}-b}}$, where $\mathbf{w} \in \mathbb{R}^p$ represents the weight vector for p features, x is the input vector, and b is the bias term. Training minimized the regularized log-loss (binary cross-entropy):
$J(\mathbf{w},b) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{p}_i) + (1-y_i)\log(1-\hat{p}_i)] + \frac{\lambda}{2}\|\mathbf{w}\|_2^2$ using stochastic gradient

descent with mini-batches of size 32 and an initial learning rate of 0.01 (reduced on plateau).

The class imbalance was handled using Synthetic Minority Over-sampling Technique (SMOTE). Given the observed imbalance between the majority and minority classes within the target variable, the Synthetic Minority Over-sampling Technique (SMOTE) was applied to the training dataset. SMOTE generates synthetic samples for the minority class, effectively balancing the class distribution and preventing the model from being biased towards the majority class. The training data was also normalized using Standard Scaler.

Four models were run on this algorithm: The first using only pre-game features, the second using both in-game and pre-game features, the third after applying SMOTE to the previous model and the fourth using SMOTE and Scaling.

TABLE II
SPECIFICATIONS OF LOGISTIC REGRESSION ALGORITHM

Parameter	Value	Description
Solver	'liblinear'	Optimization algorithm (LBFGS for large datasets)
Penalty	L2 (Ridge)	Regularization type
C (inverse reg.)	1.0	Regularization strength
Max Iterations	1000	Convergence threshold
Class Weight	Balanced using SMOTE	Handles class imbalance
Random State	16	Randomness

3) **K-Nearest Neighbors**

The K-Nearest Neighbors (KNN) classification model is employed to predict match outcomes. The primary objective is to classify whether a team will win (target=1) or lose/draw (target = 0) in a given match. The model operates by identifying the k closest data points in the training set to a new, unseen data point and assigning the majority class of these neighbors to the new point. This methodology leverages historical match data and dynamically calculated rolling averages of performance statistics to inform its predictions.

The KNN classifier was implemented as a non-parametric, instance-based learner that stores the entire training dataset during the "training" phase, deferring computation to prediction time. For a query instance x, the algorithm identifies the K training samples closest in feature space via Euclidean distance $d(\mathbf{x}, \mathbf{x}_i) = \sqrt{\sum_{j=1}^{p}(x_j - x_{i,j})^2}$ (where p is the number of features), then assigns the majority class label among them. Ties were resolved by distance-weighted voting with weights $w_i = 1/d(\mathbf{x}, \mathbf{x}_i)^2$.

TABLE III
SPECIFICATIONS OF KNN ALGORITHM

Parameter	Value	Description
K (neighbors)	12	Number of nearest neighbors
Distance Metric	Euclidean	L2 norm
Weights	Distance	Inverse distance weighting
Algorithm	'auto' (KDTree)	Ball-tree for efficient queries
Leaf Size	40	Tree construction parameter

IV. RESULTS

TABLE IV
RESULTS OF EXPERIMENTS

Model	Accuracy	F1 Score
Random Forest (with only pre-game features)	0.67	0.25
Random Forest (with in-game features)	0.64	0.23
Logistic Regression (with only pre-game features)	0.52	0.37
Logistic Regression (with in-game features)	0.75	0.42
Logistic Regression (applying SMOTE)	0.54	0.28
Logistic Regression (applying SMOTE and Scaling)	0.60	0.34
KNN (with only pre-game features)	0.67	0.00
KNN (with in-game features)	0.67	0.10

V. DISCUSSION

The Logistic Regression model, run using pre-game and rolling averages of previous 3 in-game features far out-performed all other models with accuracy 0.75 and F1 score 0.42. It was able to handle the unbalanced class weightage better than the other models, surpassing even the Logistic Regression model equipped with SMOTE. Since the football team is not consistent in its plays and victories, it was expected that an imbalance handling algorithm would be favored but the Logistic Regression model run with SMOTE did not improve the results which corroborate the findings of paper [9], which shows that oversampling techniques can be detrimental to a model if it is not calibrated well.

As we can see from Table IV, rolling averages of in-game features did not definitively increase accuracy and F1 score across the three models, with the accuracy and F1 score reducing for Random Forest but increasing for both Logistic Regression and KNN.

V. Conclusion

This study compared three ML models namely, Random Forest, Logistic Regression and KNN, and applied variations of features to it. The first conclusion we infer is that Logistic Regression is the most suitable model to use for this dataset as it is able to handle class imbalances to study whether just pre-game or including in-game features led to better predictions. We can conclude that Random Forest is not suitable for handling unbalanced classes and that perhaps a Weighted Random Forest model would be more suitable to test on this dataset [10].

References

[1] Daniel Berrar, Philippe Lopes, and Werner Dubitzky, "Incorporating domain knowledge in machine learning for soccer outcome prediction," Machine-mediated learning, vol. 108, no. 1, 2019, doi: 10.1007/S10994-018-5747-8.

[2] Mike Maher, "Modelling association football scores,", vol. 36, 1982, doi: 10.1111/J.1467-9574.1982.TB00782.X.

[3] *Darwin Prasetio, and Dra. Harlili, "Predicting football match results with logistic regression," 2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA), 2016, doi: 10.1109/ICAICTA.2016.7803111.*

[4] F. Sjöberg and S. Azimi Rashti, "Football match prediction using machine learning," thesis, Åbo Akademi University, 2023. [Online].

[5] Fátima Rodrigues, and Ângelo Pinto, "Prediction of football match results with Machine Learning," Procedia Computer Science, vol. 204, 2022, doi: 10.1016/J.PROCS.2022.08.057.

[6] Pakawan Pugsee, and Pattarachai Pattawong, "Football Match Result Prediction Using the Random Forest Classifier," ICBDT2019, 2019, doi: 10.1145/3358528.3358593.

[7] Fbref, "Football Statistics and History," FBref.com, 2024. https://fbref.com/en/

[8] scikit-learn, "scikit-learn/scikit-learn," GitHub, Apr. 15, 2019. https://github.com/scikit-learn/scikit-learn

[9] R. van den Goorbergh, M. van Smeden, D. Timmerman, and B. Van Calster, "The Harm of Class Imbalance Corrections for Risk Prediction models: Illustration and Simulation Using Logistic Regression," Journal of the American Medical Informatics Association, vol. 29, no. 9, Jun. 2022, doi: https://doi.org/10.1093/jamia/ocac093.

[10] A. More, and Dipti P Rana, "Review of random forest classification techniques to resolve data imbalance," 2017 1st International Conference on Intelligent Systems and Information Management (ICISIM), 2017.