Journal of Computer and Communications, 2025, 13(11), 41-62
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Comparative Evaluation of Machine Learning Models for NBA Game Outcome Prediction

Yuqi Wang

Department of Applied and Computational Mathematical Sciences (ACMS), College of Arts and Sciences, University of Washington, Washington, USA
Email: wyq20040113@163.com

How to cite this paper: Wang, Y.Q. (2025) Comparative Evaluation of Machine Learning Models for NBA Game Outcome Prediction. *Journal of Computer and Communications*, **13**, 41-62.
https://doi.org/10.4236/jcc.2025.1311004

Received: October 15, 2025
Accepted: November 9, 2025
Published: November 12, 2025

Abstract

This paper evaluates the performance of multiple machine learning models in predicting NBA game outcomes. Both regression and classification approaches were explored, with models including Logistic Regression, Lasso, Elastic Net, Random Forest, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Deep Neural Networks (DNN), and the AutoML framework AutoGluon. The results indicate that classification models outperform regression-based models, with SVM achieving the highest accuracy (0.7749), followed closely by AutoGluon (0.7738) and DNN (0.7726). Logistic Regression (0.7743) and Random Forest (0.7685) also demonstrated competitive performance, while KNN (0.7508) lagged behind. Feature engineering, such as standardization and polynomial expansion, significantly enhanced linear models, while regularization improved stability. Ensemble learning, particularly through AutoGluon, proved critical for capturing complementary model strengths, and deep learning benefited from dropout, batch normalization, and early stopping to mitigate overfitting. Overall, models that leverage non-linear structures and ensemble strategies showed the greatest effectiveness for sports outcome prediction.

Keywords

NBA Game Prediction, Machine Learning, Feature Engineering, AutoGluon, Model Comparison

1. Introduction

The prediction of sports game outcomes has long been a central focus for analysts, bettors, coaches, and sports enthusiasts. Accurate forecasting not only informs betting markets and fantasy sports platforms but also supports decision-making

for team managers, sponsors, and media outlets [1]. With the increasing availability of rich statistical data and the continuous advancement of machine learning technologies, sports result prediction has evolved from simple statistical estimations to sophisticated algorithmic systems [2]. Among various sports, the National Basketball Association (NBA) has emerged as an ideal subject for predictive analytics due to its highly structured gameplay, abundant historical records, and global popularity.

A substantial body of research has investigated different methodologies for predicting basketball outcomes. Traditional approaches have included regression-based statistical models and Poisson regression frameworks, which provide interpretable insights but may struggle with complex, high-dimensional data [3]. Subsequent studies incorporated machine learning techniques such as decision trees, support vector machines (SVMs), and neural networks, demonstrating improved predictive accuracy in certain contexts. More recent work has explored deep learning and hybrid approaches, for example integrating neural networks with feature selection or explainable AI methods to enhance both accuracy and interpretability. Comparative evaluations have also been conducted, assessing multiple algorithms under similar conditions to identify performance trade-offs [4].

Despite these advances, several research gaps remain. Many existing studies focus on a single model or a narrow range of algorithms, limiting the generalizability of findings across different modeling paradigms. The influence of feature engineering and preprocessing techniques, including normalization, encoding, and feature selection, on prediction accuracy has not been systematically compared in the context of NBA game forecasting. Moreover, while some studies evaluate classification-based win/loss predictions, others focus on regression-based score predictions, yet few combine both perspectives in a unified comparative framework [5].

To address these gaps, this study conducts a comprehensive comparison of multiple machine learning approaches in predicting NBA game outcomes using structured tabular features. The investigated models include Logistic Regression, Support Vector Machines (SVM), Random Forests, Deep Neural Networks (DNN), and the AutoML framework AutoGluon. Both classification (win/loss) and regression (score differential) formulations are examined, allowing for a broader evaluation of model capabilities [6]. Beyond baseline model implementation, we perform systematic hyperparameter tuning for each algorithm to optimize performance, using techniques such as grid search, random search, and automated optimization within AutoGluon. Comparative performance is assessed not only through conventional metrics (accuracy, precision, recall, RMSE) but also via visualization tools, allowing a more intuitive interpretation of model strengths and weaknesses [7]. This study not only bridges the gap between classification and regression approaches under a unified framework but also demonstrates the critical role of preprocessing, systematic hyperparameter tuning, and visual interpretability [8]. By leveraging a large-scale dataset and benchmarking a diverse set of algorithms, our work provides one of the most comprehensive eval-

uations to date, offering both theoretical contributions and practical guidance for future sports analytics applications.

The dataset comprises 25,796 historical NBA games sourced directly from the official NBA website, containing variables such as game date, team IDs, final scores, shooting percentages, assists, and rebounds, with a binary label for home team victory.

The contributions of this study are threefold: (1) we provide one of the most extensive empirical comparisons of traditional, machine learning, deep learning, and AutoML methods for NBA game prediction; (2) we systematically evaluate the effect of feature engineering and preprocessing on model performance; and (3) we offer practical guidance for selecting predictive models in sports analytics contexts [9]. By identifying the most effective approaches under modern data conditions, this research contributes theoretical and practical knowledge to the growing field of sports analytics and highlights the strengths and limitations of different modeling paradigms when applied to real-world prediction tasks [10].

2. Data Description

This dataset contains detailed historical records of 25,796 NBA games spanning from the 2014 season through the 2022 season, compiled from the official NBA website and supplemented with statistics from Basketball-Reference.com. It covers both regular season and playoff games, providing a comprehensive view of nearly a decade of NBA competition.

Each row represents a single game, identified by a unique GAME_ID, along with the game date (GAME_DATE_EST), season (SEASON), and the IDs of the home and away teams. The dataset includes the final score (PTS_home, PTS_away), shooting statistics (field goal percentage FG_PCT, free throw percentage FT_PCT, three-point percentage FG3_PCT), and core team performance metrics such as assists (AST) and rebounds (REB) for both home and away teams.

A binary label column HOME_TEAM_WINS indicates whether the home team won (1) or lost (0), making the dataset suitable for classification tasks and regression tasks.

This dataset offers a robust foundation for sports analytics research. Its temporal coverage from 2014 to 2022 allows for long-term performance trend analysis, while its granularity enables machine learning experiments in areas such as game outcome prediction, feature importance evaluation, and the effect of feature engineering techniques. The dataset was divided into training and testing part with a test size of 20%. The split was performed randomly, ensuring that approximately 80% of the games were used for model training and the remaining 20% for testing. This approach allows for an unbiased evaluation of model performance.

3. Model Description/Methodology

In this study, a diverse set of machine learning and deep learning models are em-

ployed to predict NBA game outcomes, ranging from interpretable linear approaches to complex ensemble and neural architectures.

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

In this study, accuracy is consistently computed using the above formula, where TP and TN denote the number of correctly classified positive and negative samples, and FP and FN represent the misclassified positive and negative samples, respectively.

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i \right)^2}{\sum_{i=1}^{n} \left(y_i - \bar{y} \right)^2}$$

In linear models, the coefficient of determination R^2 is commonly used to evaluate model performance. Here, y_i denotes the true values, $\hat{y}_i$ the predicted values, and $\bar{y}$ the mean of the observed values. An R^2 value closer to 1 indicates a better fit of the model to the data.

Logistic Regression was selected as a baseline model for the binary classification of home team victories. Logistic regression estimates the probability of a binary outcome using the logistic (sigmoid) function:

$$P\left(y = 1 \mid X \right) = \frac{1}{1 + \mathrm{e}^{-\left(\beta_0 + \beta^{\mathrm{T}} X \right)}}$$

Here, β represents the coefficients, and the output is the probability of the positive class. As a generalized linear model, it estimates outcome probabilities using the logistic function, providing easily interpretable coefficients that indicate the direction and magnitude of feature influence. This simplicity and transparency make Logistic Regression a widely used tool in areas such as credit scoring, medical diagnosis, and sports analytics. In our case, it offered a strong and interpretable starting point, achieving around 77% accuracy with balanced precision and recall.

Linear regression variants, including Lasso and Elastic Net, were applied to the task of predicting continuous score differentials (PTS_home, PTS_away) which were then derive to win/loss outcomes. Positive values indicate a home team win (1), while zero indicate a loss (0). This threshold ensures consistency with the binary labels used in classification models. These regularized linear models were applied to manage multicollinearity and enhance feature selection. Their ability to shrink irrelevant coefficients makes them suitable for identifying the most influential performance metrics, though they may struggle with complex nonlinear patterns in NBA data.

$$\hat{\beta} = \arg\min_{\beta} \left(\sum_{i=1}^{n} \left(y_i - \beta^{\mathrm{T}} x_i \right)^2 + \lambda_1 \left| \beta \right|_1 + \lambda_2 \left| \beta \right|_2^2 \right)$$

This combines L1 and L2 penalties, balancing sparsity and stability when predictors are correlated. While standard linear regression assumes all features contribute equally in a linear fashion, Lasso introduces L1 regularization to encourage

sparsity, effectively performing feature selection, while Elastic Net combines L1 and L2 penalties to stabilize predictions when features are highly correlated. These models are commonly used in forecasting, econometrics, and scenarios where interpretability is important. In this study, we progressively improved performance through feature standardization, polynomial feature expansion, and hyperparameter tuning, with a combined Lasso + Elastic Net approach reaching an R^2 of 0.6346.

Random Forest, an ensemble learning method based on aggregating multiple decision trees, was introduced to capture nonlinear patterns in the data. Random Forest ensemble structure reduces overfitting and highlights feature importance, making it especially valuable in identifying decisive factors like shooting efficiency and rebounds. They are widely used in classification and regression tasks across finance, healthcare, and sports analytics, particularly when feature interactions are complex. In our experiments, Random Forest outperformed single decision trees and provided feature importance measures that highlighted shooting percentage and rebounding as key predictors of game outcomes.

Support Vector Machines (SVM) were also explored for classification.

$$\min_{\boldsymbol{w},b} \frac{1}{2}|\boldsymbol{w}|^2 \quad \text{s.t. } y_i\left(\boldsymbol{w}^{\mathrm{T}} x_i + b\right) \geq 1, \ \forall i$$

SVMs seek an optimal hyperplane that maximizes the margin between classes, and through kernel functions such as the radial basis function (RBF) or polynomial kernels, they can handle nonlinear separations. Known for their effectiveness in high-dimensional spaces and robustness with clear margins of separation, SVMs are used in fields ranging from image classification to bioinformatics. In our case, the best configuration with standardized features achieved an accuracy of 77.49%.

K-Nearest Neighbors (KNN) was employed as a non-parametric, instance-based learning method. KNN offers a simple, instance-based approach that adapts well to local decision boundaries. For NBA game outcomes, it can exploit similarity between games, though its sensitivity to feature scaling and noise makes it less robust than other models.

$$\hat{y} = \arg\max_{c} \sum_{i=1}^{k} \mathbf{1}\left(y_i = c\right)$$

It classifies each game based on the majority label among its nearest neighbors in feature space, making no strong assumptions about data distribution. While its simplicity and adaptability to complex decision boundaries are appealing, KNN is sensitive to feature scaling. In our experiments, standardizing features led to a notable improvement, with accuracy increasing from roughly 69% to about 75% [11].

Deep Neural Networks (DNN), implemented as multilayer perceptrons, were incorporated to capture complex nonlinear relationships in the data.

$$h^{(l)} = \sigma\left(W^{(l)} h^{(l-1)} + b^{(l)}\right)$$

Here, σ is an activation function and weights are learned via backpropagation. By stacking multiple fully connected layers and training via backpropagation, DNNs can learn hierarchical feature representations. They are a standard choice for structured and unstructured data alike, powering applications in computer vision, natural language processing, and predictive analytics. In this study, optimizer selection and early stopping were used to improve convergence and prevent overfitting, resulting in an accuracy of 77.26% [12].

We also leveraged AutoGluon, an AutoML framework designed to automate model selection, hyperparameter tuning, and ensembling.

$$\hat{y} = \sum_{m=1}^{M} w_m \cdot h_m(x)$$

where w_m are the learned weights of model h_m, summing to 1.

AutoGluon evaluates a variety of base learners—such as XGBoost, LightGBM, and neural networks—and combines them into a weighted ensemble that captures complementary strengths. This approach is valuable in scenarios where rapid prototyping and robust performance are required [13]. In our case, the AutoGluon ensemble achieved 77.38% accuracy, surpassing all individual models [14].

Finally, a Convolutional Neural Network (CNN) was adapted for structured tabular data to explore whether spatial feature relationships could improve prediction accuracy [15]. CNNs, traditionally used in image and spatial data processing, apply convolutional layers to detect local patterns and reduce the number of parameters compared to fully connected networks. They have been widely adopted in computer vision, speech recognition, and bioinformatics. In our experiments, overfitting was mitigated through dropout, batch normalization, and early stopping, resulting in a validation accuracy exceeding 77% and strong generalization performance [16].

4. Experimental Setup

The experiments were conducted using a dataset comprising 25,796 NBA games played between the 2014 and 2022 seasons, covering both regular season and playoff matchups. Game data was sourced from the official NBA website and supplemented with statistics from *Basketball-Reference.com*.

The prediction task was formulated as a binary classification problem, where the goal was to predict whether the home team won (HOME_TEAM_WINS, where 1 indicates a win and 0 indicates a loss).

Each record in the dataset includes game metadata (date, team ID, season) and home team performance metrics only, including points scored (PTS_home), field goal percentage (FG_PCT_home), free throw percentage (FT_PCT_home), three-point percentage (FG3_PCT_home), assists (AST_home), and rebounds (REB_home).

In the early stage of model experimentation, both home and away team statistics were included as input features. However, as the score differences directly determines the outcome, and the model could easily identify which side had the

higher points—a variable that perfectly determines the game outcome. As a result, the model achieved an accuracy of around 98%, which clearly indicated an unrealistic prediction scenario. To avoid this problem and ensure that the model truly learned predictive patterns rather than deterministic labels, we intentionally excluded all away team statistics from the final modeling pipeline. This restriction forces the model to infer a home team's winning probability solely from its own performance indicators, creating a more authentic and generalizable predictive task.scor

5. Preprocessing

- Feature Standardization for distance-based and kernel models (KNN, SVM, DNN) to ensure consistent feature scales.
- Categorical Encoding for the home team identifier.
- Polynomial Feature Expansion was applied up to the second and third orders for linear models (Lasso, Elastic Net), increasing the original six features to 27 and 83 respectively, allowing the models to capture higher-order nonlinear relationships.

Model Parameter Settings and Loss Functions

- Logistic Regression—max_iter = 1000, penalty = 'l2'.
- Lasso Regression/Elastic Net—Regularization parameter α and l1_ratio selected via cross-validation.
- Random Forest—random state = 42.
- Support Vector Machine (SVM)—kernel = 'linear', 'rbf' and 'poly', C = 1.0, gamma = 'scale'.
- K-Nearest Neighbors (KNN)—n_neighbors = 5 by default, tuned up to 9; distance metric: Euclidean distance.
- Deep Neural Network-Hidden layer sizes (64, 32), max_iter = 500, optimizer: Adam or SGD with learning_rate_init = 0.01; loss function: binary cross-entropy.
- AutoGluon—Default parameter search across multiple learners (XGBoost, LightGBM, neural networks).
- Convolutional Neural Network (CNN)—Two convolutional layers followed by fully connected layers; dropout rate 0.5; loss function: binary cross-entropy.

6. Training and Evaluation

The dataset was split into training and test sets using a fixed random seed to ensure reproducibility. Cross-validation was applied for hyperparameter tuning. Model performance was evaluated using accuracy, precision, recall, and F1 score.

Hyperparameters were tuned using different strategies depending on the model type. For tree-based models such as Random Forest, a grid search was conducted with n_estimators ∈ [50, 100, 200], max_depth ∈ [None, 10, 20], min_samples_split ∈ [2, 5], min_samples_leaf ∈ [1, 2], and max_features ∈ ['sqrt', 'log2']. Regularized linear models, including Lasso and Elastic Net, were optimized via

cross-validation to select the best α and l1_ratio. For Support Vector Machines (SVM), the kernel type (linear, rbf) and regularization parameter C ∈ [0.1, 1, 10] were tested. In neural network models such as DNN and CNN, learning rate, batch size, and optimizer were tuned manually (e.g., learning_rate ∈ [0.001, 0.01], batch_size ∈ [32, 64]). Visualization tools such as ROC curves, precision-recall curves, and feature importance charts were used to interpret and compare model performance.

7. Models

7.1. Regression

In this study, we compared the predictive performance of Logistic Regression and Linear Regression models on the same dataset.

Logistic Regression was applied to handle the classification task. In the experiment, we set the maximum number of iterations to 1000, and the model achieved an accuracy of 0.7743 on the test set.

From the confusion matrix and classification report, it can be observed that the model achieved relatively balanced precision and recall across the two classes (F1 score of 0.71 for class 1 and 0.82 for class 0). The overall macro and weighted average F1 scores were both 0.77.

These results indicate that the Logistic Regression model demonstrates good predictive capability for this binary classification task. In this classification task, Table 1 represents home-team wins and label 0 represents losses.

Table 1. Logistic regression test report.

	precision	recall	f1-score	support
0	0.73	0.69	0.71	2040
1	0.8	0.83	0.82	3100
accuracy			0.77	5140
weighted avg	0.77	0.77	0.77	5140

In contrast, the Linear Regression model was applied to the same dataset to predict continuous labels, using the coefficient of determination (R^2) as the evaluation metric.

The R^2 score on the test set was 0.3212, indicating that the model explained approximately 32% of the variance in the target variable, which reflects a relatively limited fitting capability.

This further confirms that the task is better suited to a classification approach.

7.2. Lasso Regression

In this study, we employed Lasso (Least Absolute Shrinkage and Selection Operator) regression to address potential multicollinearity issues and enhance model interpretability. The Lasso model introduces an L1-norm penalty term to the or-

dinary least squares objective function, which automatically performs feature selection by shrinking coefficients of irrelevant variables to zero.

The initial model, without any preprocessing, yielded an R^2 score of 0.196, indicating a weak ability to explain the variance in the target variable.

Subsequently, feature standardization was applied, and cross-validation was used to automatically select the optimal regularization parameter α, resulting in an improved R^2 score of 0.321, which reflected an enhanced model fit.

Building on this, second-order polynomial feature expansion was introduced to capture nonlinear relationships among variables. This led to a further increase in R^2 to 0.329.

Finally, with the inclusion of third-order polynomial features, the model achieved its highest R^2 score of 0.345 during this stage.

This progression, as illustrated in Figure 1, clearly demonstrates how the model's performance was incrementally improved through standardization, parameter tuning, and feature expansion, showcasing the effectiveness of these techniques in enhancing the explanatory capacity of the Lasso regression model.

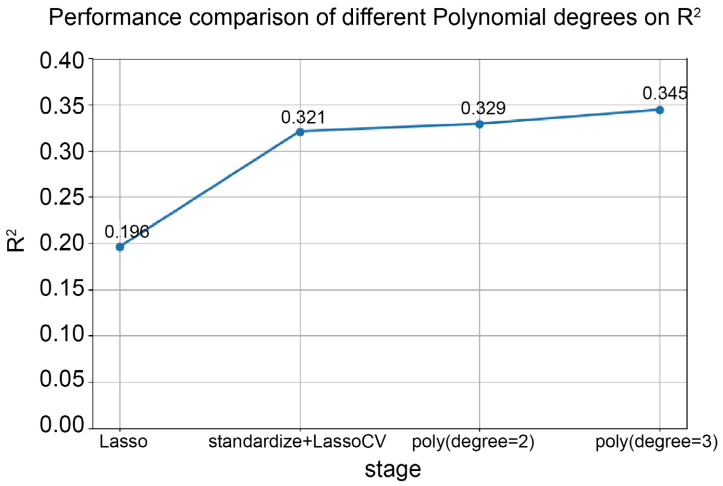

Figure 1. R^2 scores for various stages of model transformations.

7.3. Elastic Net Regression

Building upon the optimization of the Lasso model, we further experimented with the Elastic Net model.The initial Elastic Net model, without any preprocessing, achieved an R^2 score of 0.273 on the test set, outperforming the original Lasso model.

Subsequently, features were standardized, and third-order polynomial feature expansion was introduced to capture higher-order nonlinear relationships.

Additionally, cross-validation using ElasticNetCV was performed to jointly tune the regularization parameter α and the L1/L2 mixing ratio (l1_ratio).

The optimized Elastic Net model achieved an R^2 score of 0.345 on the test set, comparable to the best result obtained by the Lasso model under the same conditions, but with improved stability. By combining Lasso and Elastic Net, the model

achieved a significantly improved performance, with the R^2 score increasing to 0.6346, which is a substantial gain compared to using either model individually. This indicates that integrating the strengths of both regularization techniques can effectively enhance the model's explanatory power.

However, despite these optimization efforts, the R^2 score remained relatively low, indicating limited explanatory power regarding the target variable, which is shown in Table 2. This suggests that the underlying relationship between features and the target variable in the current dataset may involve complex nonlinearities or high-order interactions that are not well captured by linear modeling approaches.

Therefore, we infer that this data structure might not be well suited for linear regression methods.

Table 2. Model performance comparison.

	Lasso	Elastic net	Lasso + elastic net
R^2	0.34458	0.34466	0.6346
MSE	0.15689	0.15687	0.0875

7.4. Random Forest

To evaluate the performance of different models in the task of predicting competition outcomes, this study compared a single Decision Tree with the ensemble learning method Random Forest.

The experimental results showed that the Random Forest model significantly outperformed the single Decision Tree in terms of prediction accuracy on the test set.

Specifically, the Decision Tree model achieved an accuracy of 66%, while the Random Forest model reached 76% under the same feature inputs and data split conditions, representing an improvement of approximately 10 percentage points in Figure 2.

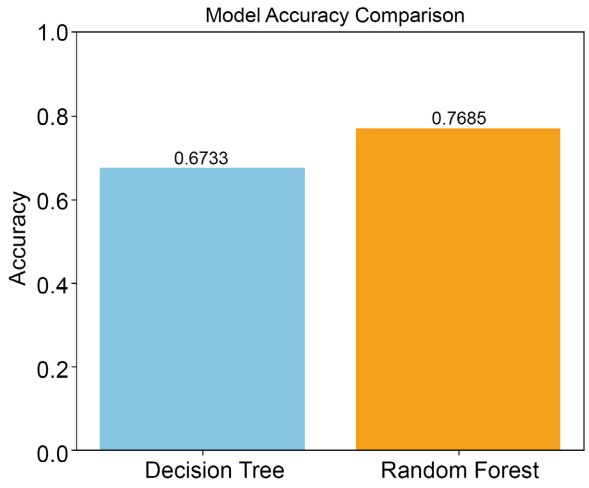

Figure 2. Model accuracy between decision tree and random forest.

In the process of further optimizing model performance, two hyperparameter tuning strategies—Grid Search (GridSearchCV) and Randomized Search (RandomizedSearchCV)—were applied to the Random Forest model.

The experimental results showed that the two methods produced no significant difference in the final model accuracy, with both achieving a test set accuracy of 77% in their best-performing models.

Hyperparameter optimization for the Random Forest model was conducted using both GridSearchCV and RandomizedSearchCV. GridSearchCV evaluated all parameter combinations within the defined grid, while RandomizedSearchCV randomly sampled 20 combinations from the same search space: n_estimators ∈ [50, 200], max_depth ∈ [None, 5, 10, 20], min_samples_split ∈ [2, 5], min_samples_leaf ∈ [1, 5], and max_features ∈ ['sqrt', 'log2']. Both methods identified similar optimal configurations and achieved identical accuracy scores of 0.7648, indicating comparable effectiveness in improving model performance in Figure 3.

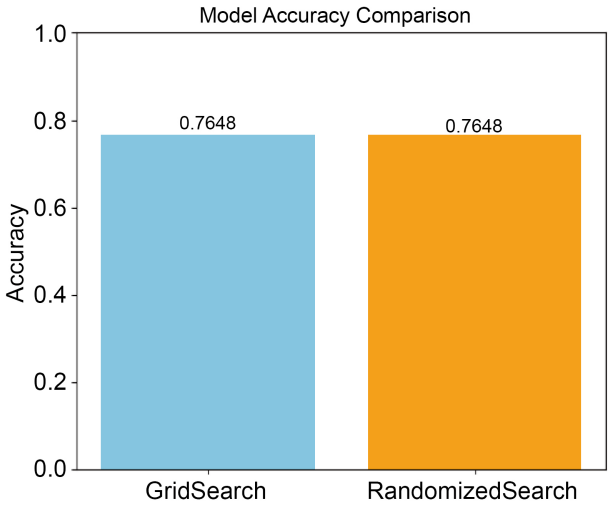

Figure 3. Model accuracy comparison between GridSearch and RandomizedSearch.

Furthermore, based on the feature importance output of the Random Forest model, this study examined the contribution of various game-related features to the model's decision-making process.

As shown in the figure, FG_PCT_home (home team field goal percentage) had the highest importance weight among all features, reaching approximately 0.32. It was followed by REB_home (home team rebounds) and PTS_home (home team points), contributing around 0.22 and 0.18, respectively.

These results suggest that shooting efficiency and rebounding ability are decisive factors in predicting game outcomes, demonstrating stronger discriminative power.

In contrast, features such as AST_home (assists) and FT_PCT_home (free throw percentage) showed relatively low importance in Figure 4, indicating their limited predictive value within the current modeling framework. Based on this

analysis, it can be inferred that shooting percentage and rebounding performance not only directly reflect a team's offensive efficiency, but may also indirectly capture aspects of tactical execution and on-court control.

Therefore, in future modeling or variable engineering efforts, it would be beneficial to prioritize the development of derived features related to shooting efficiency and rebound control, in order to further enhance the model's predictive performance.

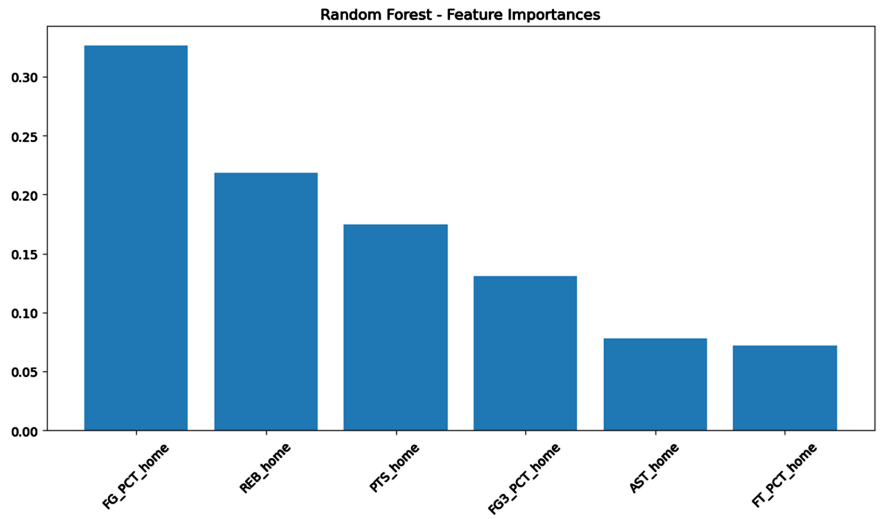

Figure 4. Random forest feature importances.

7.5. Support Vector Machine (SVM)

A Support Vector Machine model was initially constructed using a linear kernel (kernel = 'linear') to predict game outcomes. Without any feature preprocessing, the model achieved an accuracy of 0.7741 on the test set. After applying standardization to the features, the model was retrained and achieved a slightly lower accuracy of 0.7737, indicating that standardization had a limited impact on performance improvement.

To investigate the effect of feature weighting on model performance, this study attempted to increase the weight of rebound-related features during training while slightly reducing the emphasis on scoring features. Based on this adjusted feature scheme, SVM models were trained using three different kernel functions: linear, radial basis function (RBF), and polynomial. The results showed that regardless of the kernel function used, the accuracy consistently hovered around 0.712, which was significantly lower than the performance of the model trained on the original feature set.

In summary, the original feature set already captured most of the key information required for predicting game outcomes. The SVM's inherent ability to maximize the decision margin makes it particularly robust for data structures where the boundary between winning and losing outcomes is approximately linearly separable in high-dimensional space. Moreover, variables such as points

scored and shooting percentage may have already implicitly represented the influence of rebounding, so emphasizing a single feature dimension did not yield additional performance gains. Ultimately, after applying feature standardization and setting the kernel parameter gamma = 'scale', the model achieved its highest prediction accuracy of 0.7749. Accuracy rates under different models is reported in Figure 5.

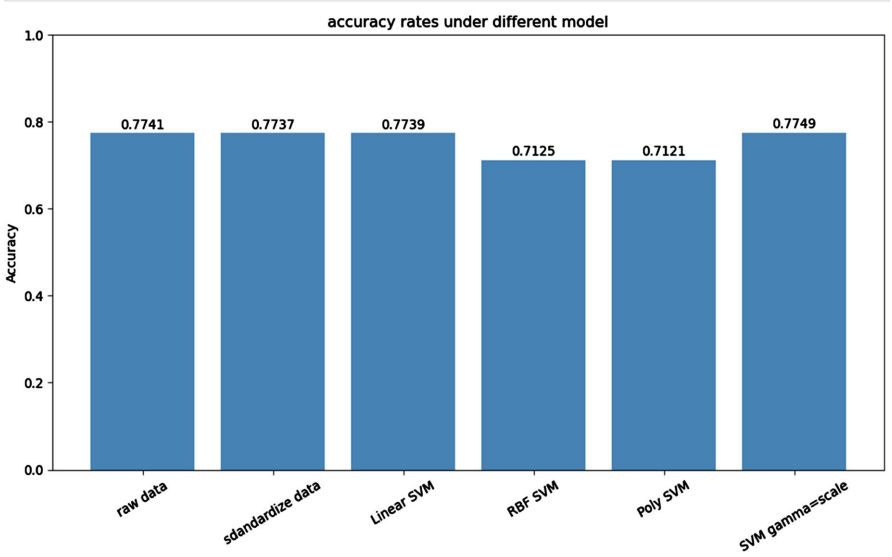

Figure 5. Accuracy comparison of various models.

7.6. K-Nearest Neighbors (KNN)

To evaluate the performance of the KNN model under different feature preprocessing strategies, this study conducted modeling experiments on both the raw (unscaled) data and the standardized data.When using the original unscaled features, the KNN model (with n_neighbors = 5) achieved an accuracy of 68.73% on the test set, indicating that the model struggles to handle datasets with significant feature scale differences [17].

To mitigate this issue, all input features were standardized, and the classifier was retrained using the same model parameters.

The experimental results showed a notable improvement in performance, with the accuracy increasing to 73.97%, representing a gain of 5.24 percentage points.

Furthermore, by tuning the n_neighbors parameter, it was observed that the model accuracy improved slightly with larger neighborhood sizes but tended to stabilize after n_neighbors = 9.

Under this setting, the KNN model achieved test set accuracies of 0.7029 on the raw data and 0.7508 on the standardized data, further confirming the critical role of feature standardization in enhancing the performance of KNN classifiers. The experiment results are shown in Figure 6.

However, despite these improvements, KNN still underperformed compared to other models. This may be because game outcomes are not simply determined by

the most similar historical matches but rather by more complex, global patterns that involve multiple interacting variables. As an instance-based learning method, KNN primarily relies on local similarity and therefore struggles to capture these broader dependencies within the data.

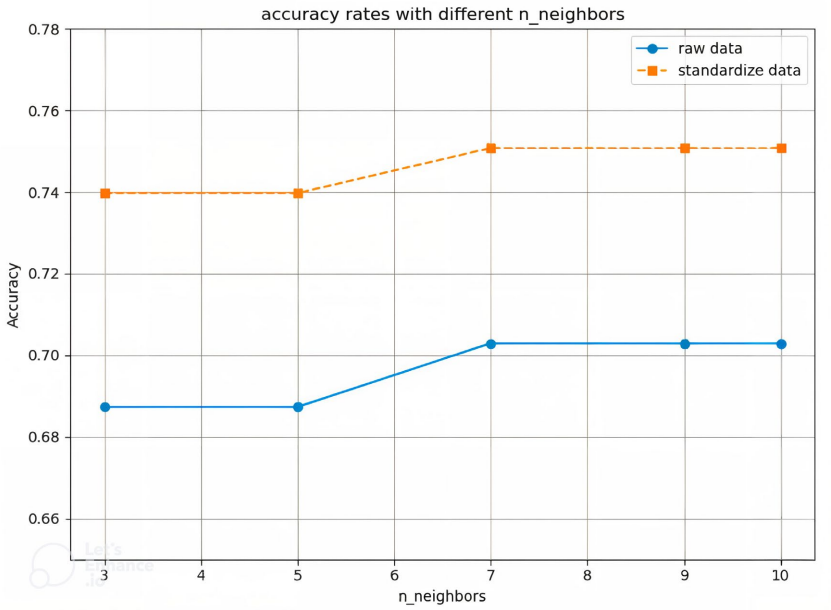

Figure 6. Accuracy rates with different numbers of neighbors for raw and standardized data.

7.8. Deep Neural Network (DNN)

In this experiment, a multilayer perceptron (MLPClassifier) model was used to train and evaluate the dataset.

The baseline model employed the default optimizer Adam with a maximum of 500 iterations, achieving an accuracy of 0.7607 on the test set.

Subsequently, we introduced the Stochastic Gradient Descent (SGD) optimizer, set the initial learning rate to 0.01, and adopted the invscaling learning rate schedule. With these modifications, the model's accuracy improved to 0.7726.

As shown in the figure, the improved deep neural network exhibited a clear loss reduction trend during training. The loss value rapidly decreased from approximately 0.526 to around 0.485 within the first few epochs, indicating fast convergence.

After that, the loss curve began to stabilize, and training was stopped early due to the validation error improvement falling below the set threshold (tol = 0.0001) for several consecutive iterations.

This trend indicates that the model achieved a satisfactory level of fitting in a short period, demonstrating good convergence. The rapid and stable decline in the loss curve in Figure 7 highlights the effectiveness of the chosen optimizer and learning rate adjustment strategy, helping the model to learn faster and more robustly, while reducing the risk of overfitting.

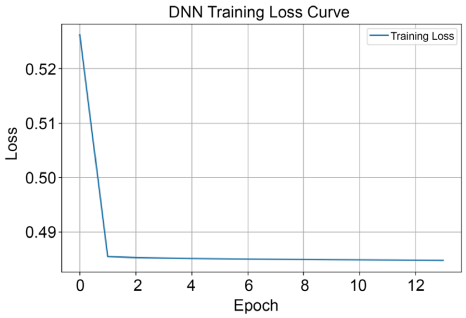

Figure 7. DNN training loss curve.

Overall, while the DNN achieved strong performance with an accuracy of 0.7726, its improvement over simpler models such as SVM or Random Forest remained modest. This may be because the dataset's features are primarily structured and low-dimensional, limiting the advantage of deep architectures that typically excel at capturing complex nonlinear representations. Moreover, without spatial or sequential dependencies in the data, the depth of the network does not translate into substantial predictive gains.

7.9. Autogluon

To evaluate the performance of various machine learning models on the prediction task, we employed AutoGluon's TabularPredictor to automatically train and validate a diverse set of base learners (Level 1 models) using default hyperparameters. Among the 13 L1 models evaluated sequentially, XGBoost achieved the highest individual validation accuracy of 0.7675, closely followed by LightGBM (0.7670), NeuralNetTorch (0.7704), and NeuralNetFastAI (0.7714). The final ensemble model (WeightedEnsemble_L2) combined multiple base models, assigning the highest weight to NeuralNetFastAI (0.75), and achieved a validation accuracy of 0.7738, surpassing all individual models. Different models' accuracy are shown in **Figure 8**. This result highlights the advantage of ensemble learning in capturing complementary strengths across different model architectures.

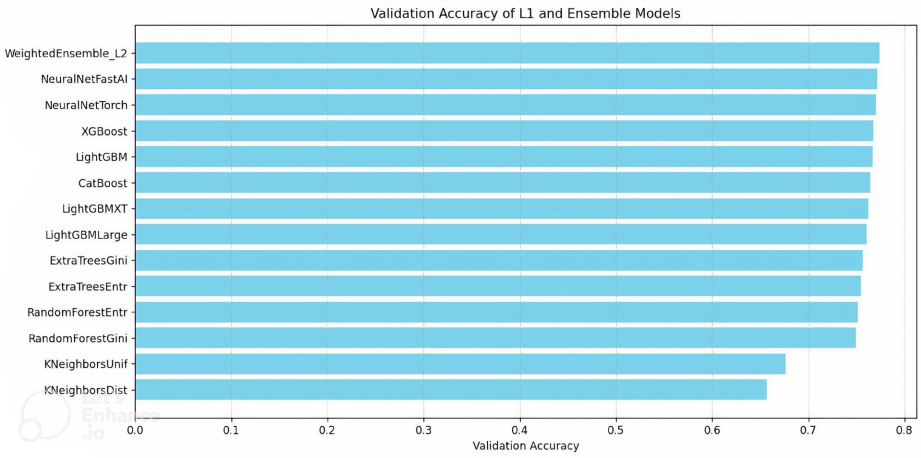

Figure 8. Accuracy comparison of various L1 and ensemble models.

The bar chart of validation accuracy clearly illustrates the relative performance of each model. Neural network-based and gradient boosting methods consistently outperformed other algorithms, while tree-based ensembles such as CatBoost, ExtraTrees, and RandomForest provided stable but slightly lower results. This comparison highlights the diversity of strengths across different model families.

AutoGluon's final ensemble model, WeightedEnsemble_L2, assigned the highest weight to NeuralNetFastAI (0.75), with smaller contributions from CatBoost, NeuralNetTorch, KNeighbors, and LightGBM. By combining these complementary learners, the ensemble achieved the best validation accuracy of 0.7738, surpassing all individual models. This demonstrates the advantage of ensemble learning, where heterogeneous models capture different aspects of the data and reduce individual weaknesses.

In this experiment, AutoGluon demonstrated several notable advantages. First, it automates the modeling process by systematically exploring multiple models and ensembles, thereby eliminating the need for extensive manual tuning. Second, it enhances interpretability by reporting the weight distribution of base learners, allowing researchers to clearly identify which models contribute most to the final predictions rather than treating the ensemble as a complete "black box." Finally, its efficiency is particularly impressive: the entire training process completed in only about 40 seconds, highlighting its effectiveness for large-scale datasets.

Overall, the results confirm that AutoGluon is a powerful tool for rapid prototyping and model selection. Its ability to combine diverse learners into an ensemble not only improves predictive performance but also offers interpretability and deployment efficiency, making it well-suited for real-world predictive tasks.

7.10. CNN

Although Convolutional Neural Networks were originally designed for spatial data such as images, in this study we adapted them for structured tabular data. The features of each game were reshaped into a two-dimensional matrix, allowing convolutional filters to slide across the feature space and capture local relationships among variables such as shooting percentage, rebounds, and assists. In this way, CNN effectively learns localized feature dependencies while retaining its advantages in feature extraction and generalization.

To evaluate the performance of deep learning methods for predicting home team victories, we trained a Convolutional Neural Network model. The model was trained for 50 epochs, with accuracy and loss trends recorded for both the training and validation sets. As shown in the graphs, the training accuracy steadily increased and converged, while the validation accuracy fluctuated between 0.765 and 0.775, ultimately reaching around 0.76. This indicates good generalization ability. At the same time, the training loss continued to decrease, and the validation loss slightly increased in the later stages in **Figure 9**, suggesting some overfitting.

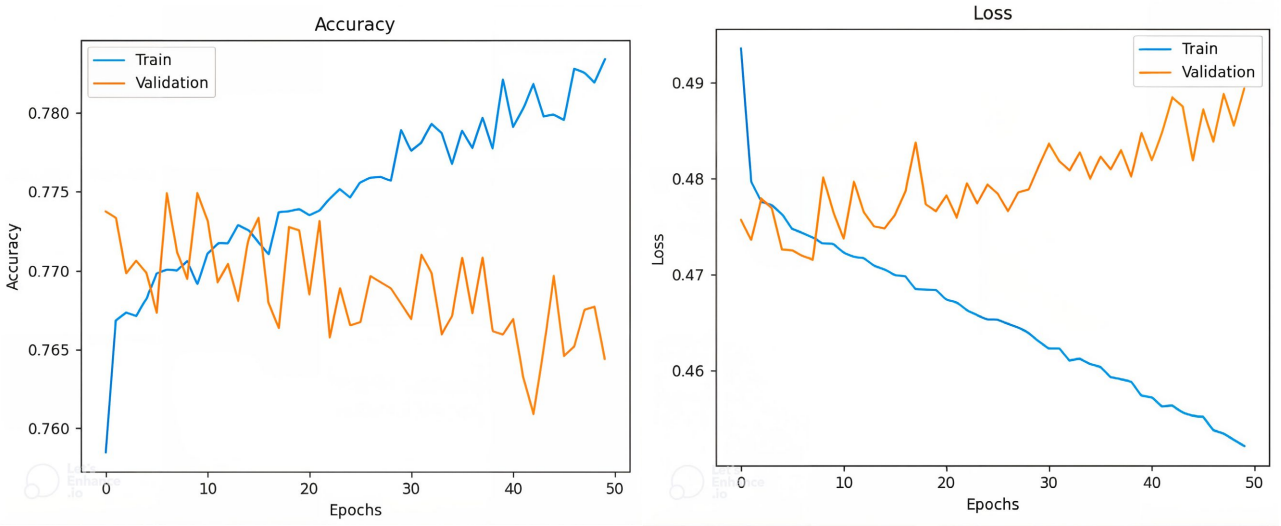

Figure 9. Training and validation accuracy and loss curves.

In the initial implementation of the convolutional neural network (CNN), the model exhibited clear signs of overfitting. While the training accuracy steadily increased, the validation accuracy fluctuated and eventually declined. Similarly, validation loss began to rise after a few epochs, indicating that the model was failing to generalize beyond the training data.

To address this issue, several modifications were introduced. First, dropout layers were added after both the convolutional and dense layers to reduce overfitting by preventing co-adaptation of neurons. Second, batch normalization layers were incorporated to stabilize and accelerate the training process. Third, an early stopping mechanism was applied to halt training once the validation performance stopped improving, thereby avoiding unnecessary epochs that could lead to overfitting.

These enhancements led to a notable improvement in model performance. The updated training curves in **Figure 10** show that both training and validation accuracies are now more aligned, and validation accuracy consistently exceeds 0.77. More importantly, the validation loss steadily decreases and remains stable, in contrast to the previous upward trend. This suggests that the revised model has better generalization ability and is no longer overfitting the training data. The improvements confirm that the model architecture and training strategy are more suitable for the given dataset.

The model's classification report shows an overall accuracy of 0.76, highlighting its strong performance in this binary classification task. This accuracy is notably higher than that achieved by previous models, such as traditional machine learning classifiers, demonstrating the effectiveness of CNN in capturing spatial patterns from structured tabular data. However, the improvement remains moderate because the dataset lacks explicit spatial or image-like structures for convolution to exploit. In this context, CNN's strength in feature locality and translation invariance is underutilized.

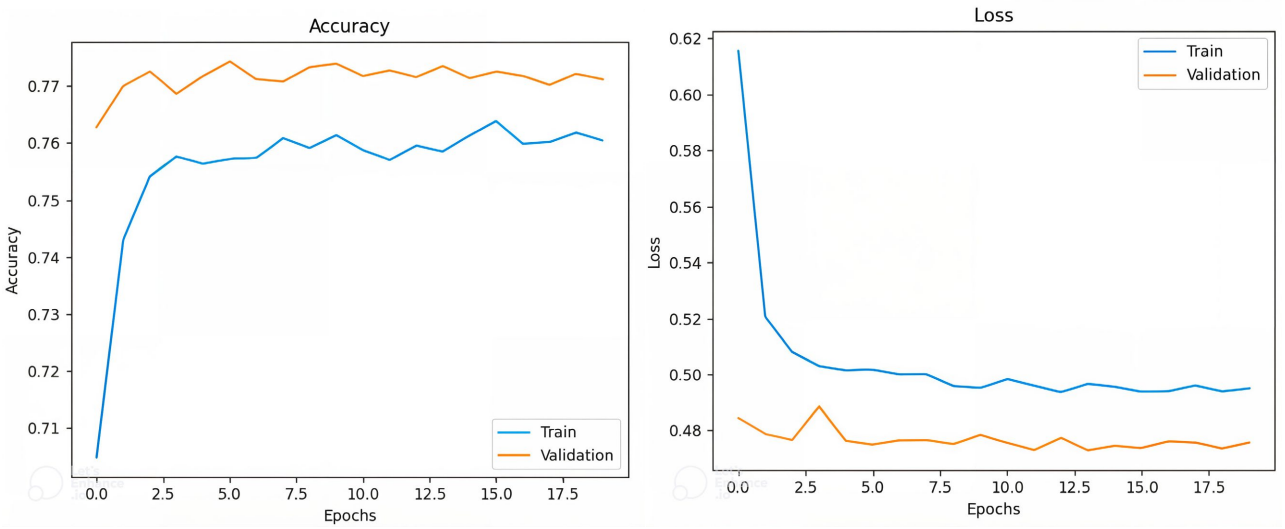

Figure 10. Improved training and validation accuracy and loss curves.

Feature Engineering Details are shown in **Table 3**.

Table 3. Different models with key hyperparameters and results.

Model	Key Hyperparameters	Results
Logistic Regression	max_iter, penalty, C	max_iter = 1000, penalty = L2, C = 1.0
Lasso + Elastic Net	l1_ratio, cv	l1_ratio = 0.9, cv = 5
Random forest	max_depth, max_features, min_samples_leaf, min_saple_split	max_depth = 10, max_features = 'log2', min_samples_leaf = 3, min_saple_split = 5
SVM	kernal, gamma	kernal = 'linear', gamma = 'scale'
KNN	n_neighbors	k = 9
DNN	hidden_layer_sizes, solver, learning_rate	hidden_layer_sizes = 64, solver = sgd, learning_rate =0.01
AutoGluon	hyperparameters of Different Models	WeightedEnsemble_L2
CNN	optimizer, epochs, batch_size	optimizer = 'adam', epochs = 50, batch_size = 32

8. Discussion

While most models achieved competitive results, several factors may explain why certain models produced lower predictive accuracy. K-Nearest Neighbors (KNN) performed the worst (0.7508), as it is highly sensitive to noise and feature scaling, and its reliance on local similarity does not generalize well to large, high-variance datasets. Linear regression variants such as Lasso and Elastic Net also struggled because they assume linear relationships. Though improvements from feature standardization and polynomial expansion, they could not capture the nonlinear interactions that drive basketball outcomes.

Random Forest showed stable but slightly lower performance (0.7685). While it is effective at capturing nonlinearities and providing feature importance, it can overemphasize dominant variables and underweight contextual features. Its en-

semble structure also increases variance when the feature space is limited.

Support Vector Machines (SVMs) achieved the highest score (0.7749), benefiting from the high-dimensional representation of standardized features, but they can be computationally expensive and scale poorly with much larger datasets. Deep Neural Networks (DNNs) performed strongly but required careful tuning of optimizers and learning rates to avoid overfitting.

Despite the strong overall performance, this study has several limitations. The models rely only on team-level aggregate data and treat each game as an independent event, which limits their ability to capture player-level dynamics, injuries, and temporal effects such as momentum or fatigue. Contextual factors like coaching strategies or team chemistry are also not represented [18].

Future Research

This study primarily focuses on team-level aggregate data. Future research could expand in several directions.

First, future work can move beyond team-level aggregates by incorporating player-level features. These include injury reports, player availability, and real-time performance indicators such as shooting efficiency, usage rate, or plus-minus statistics. For example, integrating injury data could allow the model to capture sudden performance drops when a key player is absent. Similarly, adding rolling averages of player statistics would help reflect short-term form or fatigue, which static season-level statistics cannot capture.

Second, more advanced modeling approaches can be explored. Graph Neural Networks (GNNs) could be applied to represent the relationships between players and teams as graph structures, enabling the model to account for interactions such as lineup combinations or defensive matchups. Time-series models such as LSTMs could track season dynamics, capturing momentum effects, losing streaks, or gradual improvements in team chemistry. Combining these methods with domain-specific knowledge (e.g., historical matchups or rivalry effects) would enhance both interpretability and robustness. Together, these approaches could significantly improve predictive performance in realistic, dynamic NBA contexts.

9. Conclusion

In **Figure 11**, it shows that most models perform closely, with Support Vector Machine (SVM) achieving the highest accuracy of 0.7749, followed by AutoGluon (0.7738) and Deep Neural Networks (DNN, 0.7726). These models demonstrate strong generalization and are well suited for complex, high-dimensional data. Traditional models like Logistic Regression (0.77) and Random Forest (0.7685) also remain competitive, while K-Nearest Neighbors (KNN, 0.7508) lags behind due to its sensitivity to data size and noise. Convolutional Neural Networks (CNN, 0.76) showed moderate performance, likely limited by the lack of spatial structure in tabular features. Overall, SVM, AutoGluon, and DNN stand out as the most reliable models for NBA game outcome prediction.

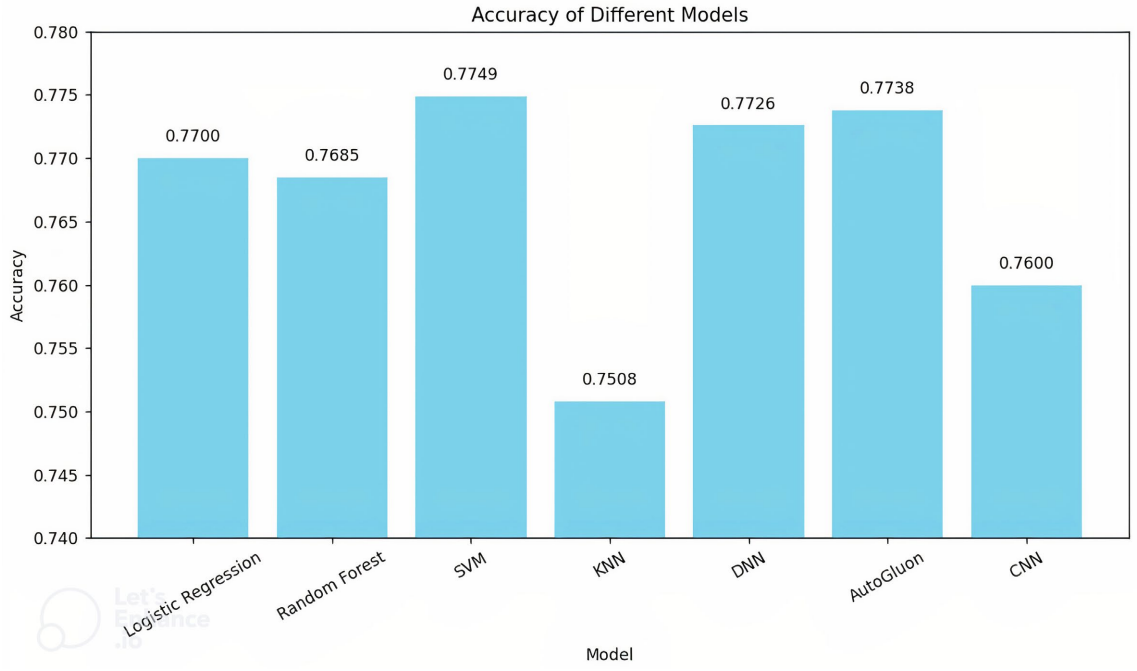

Figure 11. Accuracy of different models

In this study, we conducted a comprehensive evaluation of multiple machine learning models—including regression, classification, ensemble, and deep learning approaches—for predicting NBA game outcomes. The results demonstrate that classification models consistently outperform regression-based models, with Support Vector Machine (SVM), AutoGluon, and Deep Neural Networks (DNN) achieving the highest accuracy, all exceeding 0.77. Feature standardization, polynomial expansion, and regularization significantly improved model performance in linear models like Lasso and Elastic Net, though their explanatory power remained limited. Ensemble methods such as Random Forest and AutoGluon showcased strong generalization abilities, while deep learning models benefited from dropout, batch normalization, and early stopping to prevent overfitting. Overall, models that capture non-linear relationships and leverage ensemble or deep architectures prove more effective for sports outcome prediction using structured tabular data.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Spann, M. and Skiera, B. (2009) Sports Forecasting: A Comparison of the Forecast Accuracy of Prediction Markets, Betting Odds and Tipsters. *Journal of Forecasting*, **28**, 55-72. https://doi.org/10.1002/for.1091

[2] Galekwa, R.M., Tshimula, J.M., Tajeuna, E.G. and Kyandoghere, K. (2024) A Systematic Review of Machine Learning in Sports Betting: Techniques, Challenges, and Future Directions. arXiv:2410.21484.

[3] Koopman, S.J. and Lit, R. (2015) A Dynamic Bivariate Poisson Model for Analysing and Forecasting Match Results in the English Premier League. *Journal of the Royal Statistical Society Series A*: *Statistics in Society*, **178**, 167-186. https://doi.org/10.1111/rssa.12042

[4] Thabtah, F., Zhang, L. and Abdelhamid, N. (2019) NBA Game Result Prediction Using Feature Analysis and Machine Learning. *Annals of Data Science*, **6**, 103-116. https://doi.org/10.1007/s40745-018-00189-x

[5] Cao, C. (2012) Sports Data Mining Technology Used in Basketball Outcome Prediction. https://arrow.tudublin.ie/scschcomdis/39/

[6] Manner, H. (2016) Modeling and Forecasting the Outcomes of NBA Basketball Games. *Journal of Quantitative Analysis in Sports*, **12**, 31-41. https://doi.org/10.1515/jqas-2015-0088

[7] Chen, W., Jhou, M., Lee, T. and Lu, C. (2021) Hybrid Basketball Game Outcome Prediction Model by Integrating Data Mining Methods for the National Basketball Association. *Entropy*, **23**, Article 477. https://doi.org/10.3390/e23040477

[8] Miljkovic, D., Gajic, L., Kovacevic, A. and Konjovic, Z. (2010) The Use of Data Mining for Basketball Matches Outcomes Prediction. *IEEE 8th International Symposium on Intelligent Systems and Informatics*, Subotica, 10-11 September 2010, 309-312. https://doi.org/10.1109/sisy.2010.5647440

[9] Nguyen, N.H., Nguyen, D.T.A., Ma, B. and Hu, J. (2022) The Application of Machine Learning and Deep Learning in Sport: Predicting NBA Players' Performance and Popularity. *Journal of Information and Telecommunication*, **6**, 217-235. https://doi.org/10.1080/24751839.2021.1977066

[10] Wang, Y., Liu, W. and Liu, X. (2022) Explainable AI Techniques with Application to NBA Gameplay Prediction. *Neurocomputing*, **483**, 59-71. https://doi.org/10.1016/j.neucom.2022.01.098

[11] Guo, G., Wang, H., Bell, D., Bi, Y. and Greer, K. (2003) KNN Model-Based Approach in Classification. In: Meersman, R., Tari, Z. and Schmidt, D.C., Eds., *Lecture Notes in Computer Science*, Springer, 986-996. https://doi.org/10.1007/978-3-540-39964-3_62

[12] Loeffelholz, B., Bednar, E. and Bauer, K.W. (2009) Predicting NBA Games Using Neural Networks. *Journal of Quantitative Analysis in Sports*, **5**, 1-5. https://doi.org/10.2202/1559-0410.1156

[13] Ouyang, Y., Li, X., Zhou, W., Hong, W., Zheng, W., Qi, F., *et al.* (2024) Integration of Machine Learning Xgboost and SHAP Models for NBA Game Outcome Prediction and Quantitative Analysis Methodology. *PLOS ONE*, **19**, e0307478. https://doi.org/10.1371/journal.pone.0307478

[14] Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M. and Smola, A. (2020) Autogluon-Tabular: Robust and Accurate Automl for Structured Data. arXiv:2003.06505.

[15] Chua, L.O. (1998) CNN: A Paradigm for Complexity. Vol. 31, World Scientific. https://doi.org/10.1142/9789812798589

[16] Emmert-Streib, F., Yang, Z., Feng, H., Tripathi, S. and Dehmer, M. (2020) An Introductory Review of Deep Learning for Prediction Models with Big Data. *Frontiers in Artificial Intelligence*, **3**, Article ID: 4. https://doi.org/10.3389/frai.2020.00004

[17] Zhang, S., Li, X., Zong, M., Zhu, X. and Cheng, D. (2017) Learning *k* for KNN Classification. *ACM Transactions on Intelligent Systems and Technology*, **8**, 1-19. https://doi.org/10.1145/2990508

[18] Papageorgiou, G., Sarlis, V. and Tjortjis, C. (2024) Evaluating the Effectiveness of Machine Learning Models for Performance Forecasting in Basketball: A Comparative Study. *Knowledge and Information Systems*, **66**, 4333-4375. https://doi.org/10.1007/s10115-024-02092-9