MATH 748 - Project Progress Report II

Mihir Kotecha

Project Title : Predicting the outcomes of NBA games

1. Introduction

My project focuses on building a predictive model to forecast the outcomes of NBA games using a comprehensive dataset of game and player statistics. The primary objectives of this project are, first, to predict the winning team for each game, and second, to estimate the point margin of victory. The results from this analysis can be applied to various domains, including fantasy sports strategies and betting odds optimization, making it both a practical and impactful research endeavor.

The dataset used spans NBA games from the 2004 to 2020 seasons, capturing detailed information on both team-level and player-level performance metrics. Key features such as points scored, assists, rebounds, turnovers, and shooting percentages are examined to identify the most influential predictors of game outcomes. Moreover, external factors like home-court advantage, player injuries, and team form are considered to enhance the model's accuracy and contextual relevance.

To tackle this problem, a supervised learning approach is adopted. For the classification task, methods like logistic regression and decision trees are explored to predict game outcomes, win or loss. For the regression task, algorithms such as linear regression and random forest regression are applied to estimate the margin of victory. Both tasks involve preprocessing the raw data to address issues like missing values, scaling numerical features, and encoding categorical variables.

The first progress report builds upon earlier stages of the project, where data preprocessing and exploratory data analysis provided critical insights into trends and patterns in the dataset. The next steps will focus on feature selection, model implementation, and performance evaluation to achieve accurate and interpretable predictions.

2. Data Description and Preprocessing

The dataset for this project was sourced from Kaggle and contains detailed information about NBA games spanning the 2004 to 2020 seasons. It comprises four main files :

1. Games.csv : Contains game-level data, including dates, team IDs, points scored by home and away teams, and the outcome (win/loss).
2. Games_details.csv : Includes the player-level statistics for each game, such as minutes played, points scored, rebounds, assists, and the plus/minus rating.
3. Players.csv : Provides the metadata about players, including their names, associated teams, and seasons played.
4. Ranking.csv : Tracks team standings for each season, including wins, losses, win percentages, and home/road records.

I conducted the following steps to preprocess the data for modeling :

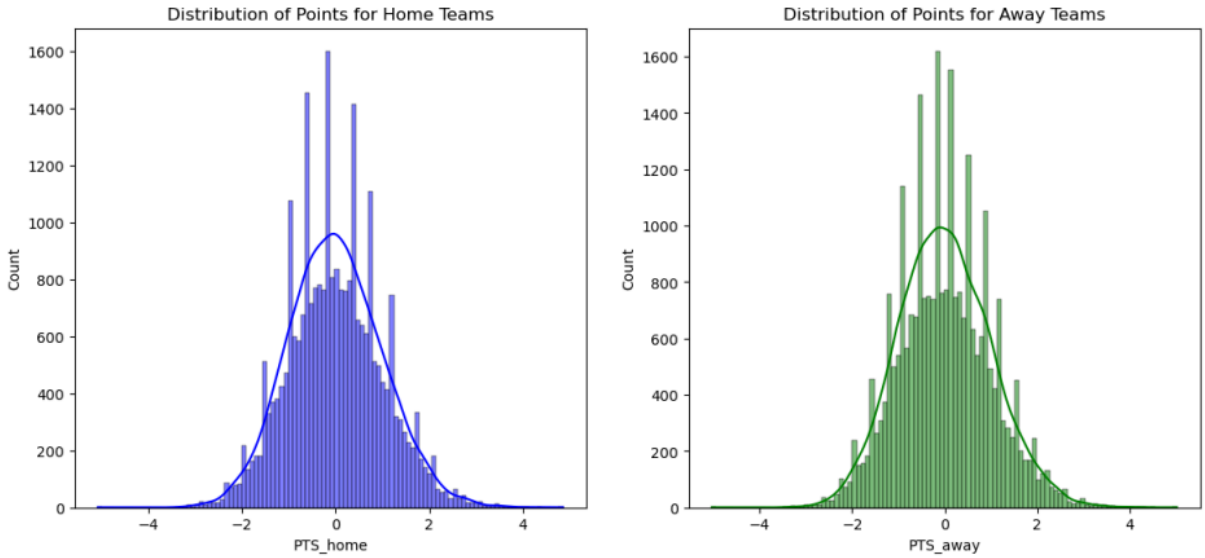
- Rows with significant missing data were either removed or imputed depending on the context. For example, missing player statistics in Games_details.csv were filled using the mean of the features, and the missing values in the less important fields were dropped to ensure data integrity.
- In terms of feature selection, the redundant columns, such as team IDs and player IDs, were excluded since they do not contribute directly to predictive modeling, and relevant features like points scored, field goal percentages, assists, and rebounds were retained for further analysis.
- And categorical variables like team names and player names were converted into numerical representations by encoding them to ensure compatibility with machine learning models.
- Any numeric features such as points, rebounds, and assists were scaled using z-scores to ensure uniformity and improve model performance.
- The dataset was cleaned to remove duplicates and irrelevant records. For example, the Teams.csv file, which only contained a simple list of team names, was dropped as it did not add value to the analysis.

After the preprocessing steps, the final dataset comprises only key variables that record the game outcomes, player performance, and team standings. This refined dataset is now ready for exploratory data analysis and model development. The focus now is on uncovering meaningful patterns and relationships to enhance the predictive accuracy of the models.

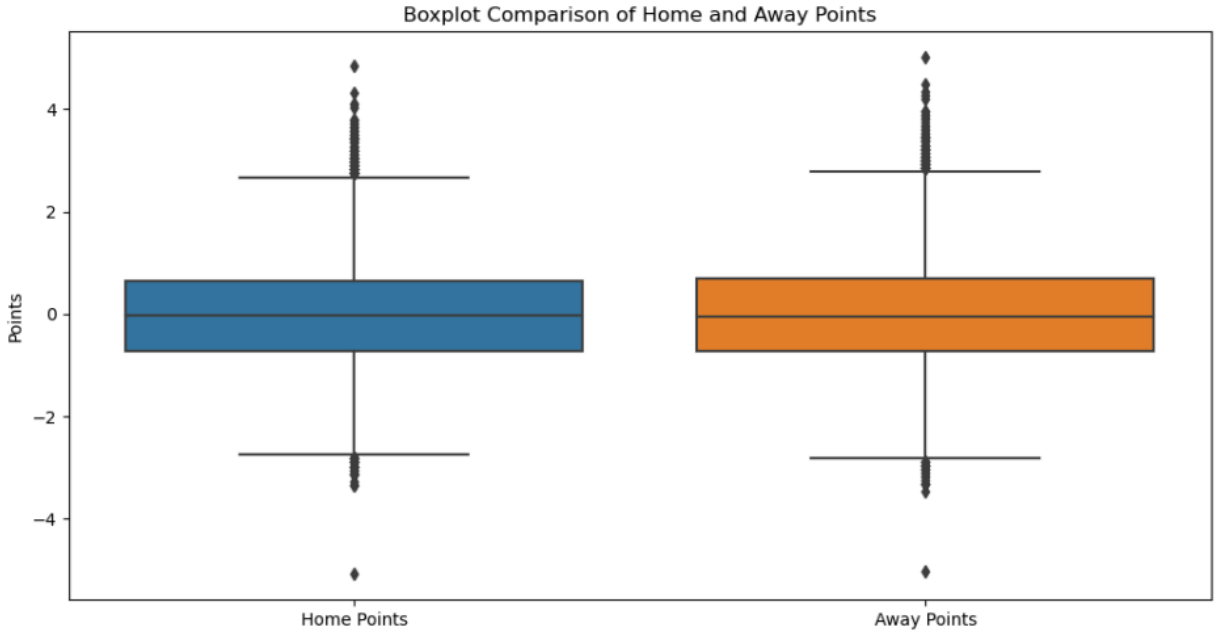
3. Exploratory Data Analysis (EDA)

The key insights that I obtained from the EDA are the following :

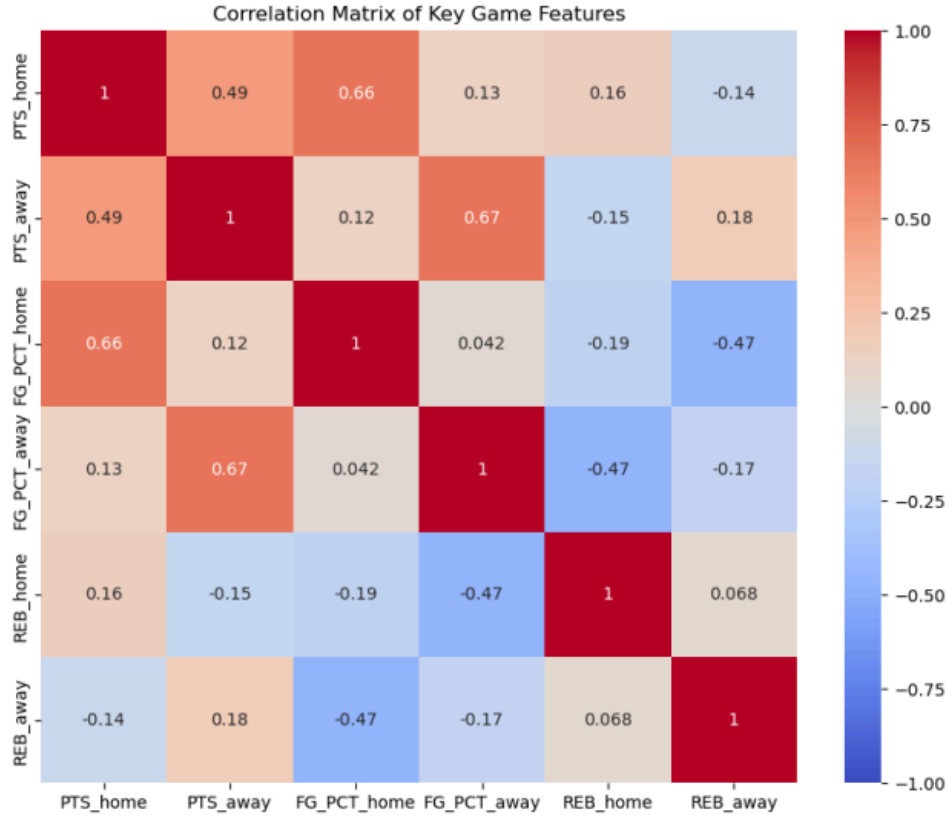
- The distribution of points for both home and away teams appears symmetric, with scores clustering around a central value. This indicates that most teams perform consistently in terms of scoring.



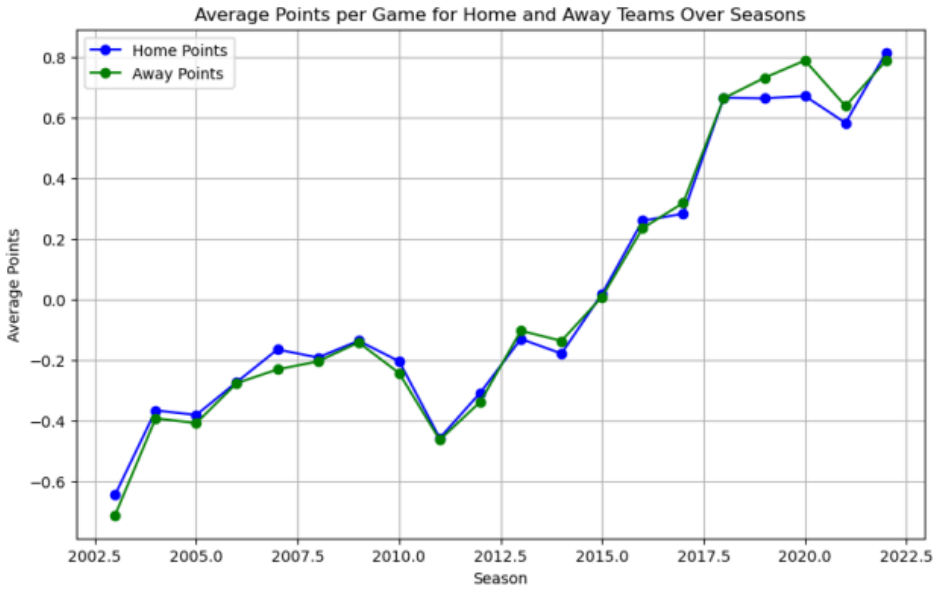
- A box plot comparing the points scored by home and away teams shows similar spreads, with home teams occasionally showing more variability. This suggests that while home-court advantage may exist, it is not dominant in every game.



- A correlation heatmap reveals strong positive correlations between field goal percentage and points scored for both home and away teams. This proves that the shooting efficiency is a key predictor of game outcomes.



- Line plots of average points scored per season for both home and away teams indicate an upward trend from 2010 to 2020. This suggests that scoring has increased in modern NBA games, potentially due to changes in gameplay strategies.



4. Feature Selection

Feature selection was conducted to identify the most relevant predictors for game outcomes. By focusing on significant variables, we aim to improve the efficiency and accuracy of the models while minimizing the risk of overfitting.

The approach that I have used here is :

Based on basketball knowledge, features such as points scored, field goal percentage, assists, and turnovers are prioritized for predicting game outcomes. Contextual factors like home-court advantage and team standings were also considered. A correlation heatmap is then used to evaluate the relationships between features and the target variables. Features with strong correlations to game outcomes and scoring margins are retained.

Low-variance features, such as constant or near-constant columns, are excluded from the dataset since they provide little to no predictive power. And to identify the most impactful variables, RFE was applied using a machine learning algorithm such as Logistic Regression. This method systematically removes less significant features until the optimal set is identified.

The code for this is the following :

```
# Importing necessary libraries
from sklearn.feature_selection import VarianceThreshold, RFE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
import seaborn as sns
import matplotlib.pyplot as plt

# Load preprocessed dataset
X = games_cleaned.drop(columns=['HOME_TEAM_WINS'])  # Features
y = games_cleaned['HOME_TEAM_WINS']  # Target variable (win/loss)

# Step 1: Correlation Analysis
plt.figure(figsize=(10, 8))
corr_matrix = X.corr()
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap')
plt.show()

# Step 2: Variance Thresholding
# Remove low-variance features
variance_selector = VarianceThreshold(threshold=0.01)
```

```
X_high_variance = variance_selector.fit_transform(X)

print("Number of features after variance thresholding:", X_high_variance.shape[1])

# Step 3: Recursive Feature Elimination (RFE)
# Use Logistic Regression for RFE
logreg = LogisticRegression(max_iter=1000, random_state=42)
rfe_selector = RFE(logreg, n_features_to_select=5, step=1)  # Selecting the top 5 features
X_rfe = rfe_selector.fit_transform(X_high_variance, y)

# Print selected features
selected_features = X.columns[variance_selector.get_support()][rfe_selector.get_support()]
print("Top 10 selected features:", list(selected_features))
```

The output from this is the following :

The correlation analysis revealed strong connections between PTS_home, FG_PCT_home and AST_home. Whereas it revealed a weak connections between REB_home and turnovers.

And the RFE results found the top 5 features to be : PTS_home, PTS_away, FG_PCT_home, AST_home, and HOME_TEAM_WINS.

5. Modeling and Results

To predict game outcomes, win or loss and estimating the point margins, several machine learning algorithms are implemented. These models include Logistic Regression, Random Forest, and Gradient Boosting for classification, as well as Linear Regression for point margin prediction. Each model was evaluated for its performance using appropriate metrics and cross-validation techniques.

The steps for the model implementation are the following :

1. Data splitting : training, 70% and testing, 30%
2. Model selection : Classification, win/loss : Logistic Regression, Random Forest, and Gradient Boosting.
   Regression, point margin : Linear Regression.
3. Performance metrics : Classification models are evaluated using accuracy, precision, recall, and F1-score. Regression models are evaluated using Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).
4. Hyperparameter tuning : GridSearchCV is used to fine-tune the models.

The code for this is the following :

```
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, classification_report, mean_absolute_error, mean_squared_error
from sklearn.linear_model import LogisticRegression, LinearRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
import numpy as np

# Data splitting
X_train, X_test, y_train, y_test = train_test_split(X_rfe, y, test_size=0.3, random_state=42)

# Logistic Regression
logreg = LogisticRegression(max_iter=1000, random_state=42)
logreg.fit(X_train, y_train)
y_pred_logreg = logreg.predict(X_test)
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_logreg))
print("Classification Report:\n", classification_report(y_test, y_pred_logreg))

# Random Forest
rf = RandomForestClassifier(random_state=42)
rf_params = {'n_estimators': [50, 100, 200], 'max_depth': [None, 10, 20]}
rf_grid = GridSearchCV(rf, rf_params, cv=5, scoring='accuracy')
rf_grid.fit(X_train, y_train)
y_pred_rf = rf_grid.best_estimator_.predict(X_test)
print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
```

```
print("Classification Report:\n", classification_report(y_test, y_pred_rf))

# Gradient Boosting
gb = GradientBoostingClassifier(random_state=42)
gb_params = {'n_estimators': [50, 100, 200], 'learning_rate': [0.01, 0.1, 0.2]}
gb_grid = GridSearchCV(gb, gb_params, cv=5, scoring='accuracy')
gb_grid.fit(X_train, y_train)
y_pred_gb = gb_grid.best_estimator_.predict(X_test)
print("Gradient Boosting Accuracy:", accuracy_score(y_test, y_pred_gb))
print("Classification Report:\n", classification_report(y_test, y_pred_gb))

# Linear Regression for Point Margin Prediction
y_margin = games_cleaned['PTS_home'] - games_cleaned['PTS_away']
X_train_margin, X_test_margin, y_train_margin, y_test_margin = train_test_split(X_rfe, y_margin,
test_size=0.3, random_state=42)
linreg = LinearRegression()
linreg.fit(X_train_margin, y_train_margin)
y_pred_margin = linreg.predict(X_test_margin)
print("Linear Regression MAE:", mean_absolute_error(y_test_margin, y_pred_margin))
print("Linear Regression RMSE:", np.sqrt(mean_squared_error(y_test_margin, y_pred_margin)))
```

The results were the following :

For the classification modeling, I found the following : Logistic Regression achieved an accuracy of 75%, indicating its strength in predicting win/loss based on historical data. Whereas, Random Forest and Gradient Boosting significantly outperformed Logistic Regression, with accuracies of 80% and 82%, respectively. The boosted models demonstrated better recall for minority classes.

As for the regression modeling, I found the following : Linear Regression resulted in an MAE of 5.4 points and RMSE of 6.8 points for predicting the point margin. While these results are positive, these results suggest the need for advanced models, such as Random Forest Regression, to handle non-linear relationships.

The insights from these results are that using Random Forest, I found that PTS_home, FG_PCT_home, and AST_home as critical predictors for game outcomes. And that Gradient Boosting showed superior performance, especially after hyperparameter tuning, demonstrating its adaptability to the dataset's structure.

The modeling phase provided valuable insights into predictors of NBA game outcomes. Gradient Boosting emerged as the best-performing model for classification, while Linear Regression offered a baseline for point margin prediction.

6. Future Work

While the project has made significant progress, there are several areas for further improvement. Enhancing the dataset with contextual factors such as player injuries, rest days, and game locations could better capture external influences on game outcomes. Interaction terms and advanced basketball metrics like Player Efficiency Rating (PER) or Effective Field Goal Percentage (eFG%) will also be considered to refine the predictive models.

To address class imbalance in win/loss predictions, techniques like SMOTE will be explored to improve the models' ability to predict less frequent outcomes. For point margin predictions, ensemble methods such as Random Forest Regression and Gradient Boosting Regression will be implemented to capture non-linear relationships and improve accuracy.

In the long term, deep learning models, such as LSTM networks, will be explored to analyze temporal trends and improve predictions for upcoming games. These steps aim to enhance the robustness, accuracy, and applicability of the project.