King Fahd University of Petroleum and Minerals

Information and Computer Science Department

ICS 474 – Big Data Analytics

Term 241

Project Report

European Soccer Database

Instructor: Dr. Muzammil Behzad

Green Falcons		
#	Name	ID
1	Yousef Buali	202025400
2	Mohammed Alnasser	202034520
3	Ali Al-Haddad	202170070

November 17, 2024

1. Introduction

The objective of this project is to analyze the Soccer dataset from Kaggle, which contains comprehensive data for soccer matches, players, and teams from several European countries spanning 11 seasons between 2008 and 2016. The dataset includes seven interconnected tables: Country, League, Match, Player, Player Attributes, Team, and Team Attributes. This extensive dataset provides detailed information on matches, player characteristics, team attributes, and game events such as goal types, and possession statistics. Player and team attributes are sourced from EA Sports' FIFA video game series.

The primary focus of this project is to develop a predictive model to determine the likelihood of a team winning a match based on the attributes of each team. Key features for prediction will be derived from the Team Attributes table, which contains metrics such as build-up play, chance creation, and defensive capabilities, alongside match-related data such as goals scored, and possession statistics. These features will serve as inputs for machine learning models to predict match outcomes accurately.

To achieve this, the dataset will be thoroughly explored and preprocessed. This involves cleaning and transforming the data by addressing missing values, duplicates, and outliers, as well as encoding categorical variables and scaling numerical ones where necessary. Visualizations such as histograms, scatter plots, and heatmaps will be used to understand feature distributions, relationships, and correlations.

The modeling phase will focus on classification algorithms suitable for predicting match outcomes, such as logistic regression, decision trees, or ensemble methods. Data splitting techniques like cross-validation will be employed to ensure the model generalizes well, and evaluation metrics such as accuracy, precision, and recall will assess its performance.
The ultimate goal is to provide actionable insights into team performance and match dynamics through data-driven analysis. By leveraging this extensive dataset and applying machine learning techniques, the project aims to support stakeholders such as coaches, analysts, and teams in making informed decisions to improve match strategies and outcomes.

2. Data Understanding and Exploration

A. Dataset Overview

The dataset used in this project is sourced from Kaggle and focuses on soccer matches, teams, and players across several European leagues. It spans 11 seasons, from 2008 to 2016, providing rich data on player and team attributes, match results, and league information. This dataset is especially suited for machine learning and data analysis in the domain of sports analytics. It addresses the problem domain of predicting match outcomes based on team attributes and performance data.

The dataset consists of seven interconnected tables:

1. **Match**: Contains data about 25,979 matches, including details like goals scored, team API IDs, and additional game events such as possession, corners, and fouls.
2. **Player**: Information on 11,060 players, including their names, height, weight, and birthdays.
3. **Player Attributes**: Contains 183,978 entries describing player abilities such as overall rating, potential, and specific attributes like crossing, dribbling, and finishing.
4. **Team**: Includes 299 teams with their API IDs, names, and abbreviations.
5. **Team Attributes**: Contains 1,458 entries describing team characteristics like build-up play, chance creation, and defensive strategies.
6. **Country**: Contains 11 entries representing countries with their respective IDs.
7. **League**: Provides information on 11 leagues, including league and country IDs.

Each table is connected through identification keys like id, team_api_id, and player_api_id, enabling comprehensive analysis of relationships between matches, teams, players, and their attributes.

B. Feature Description

The dataset comprises multiple interconnected tables, each containing various features. These features describe different aspects of soccer matches, players, teams, and leagues. Below is a detailed breakdown of the features, their data types, and their significance:

Match Table

- **Numerical Features:**
  - id: Unique match identifier.
  - country_id, league_id: Links to the country and league of the match.
  - stage: Represents the stage of the tournament (e.g., group stage, final).
  - home_team_api_id, away_team_api_id: Identifiers for the home and away teams.
  - home_team_goal, away_team_goal: Number of goals scored by home and away teams.
  - Betting odds: Columns such as B365H, B365D, B365A (Bet365 odds for home win, draw, and away win) and other betting odds from various providers.
  - Player positions (e.g., home_player_X1, away_player_Y1): Positional coordinates of players on the field.
- **Categorical Features:**
  - season: Indicates the season of the match (e.g., 2008/2009).
  - date: Date of the match.
- **Target Variable:**
  - A new feature, **winning**, can be derived using:
    - winning = 1 if home_team_goal > away_team_goal (Home Win).
    - winning = 0 if home_team_goal == away_team_goal (Draw).
    - winning = -1 if home_team_goal < away_team_goal (Away Win).

Player Table

- **Numerical Features:**

    - id: Unique identifier for each player.
    - height: Player's height (in cm).
    - weight: Player's weight (in kg).
- **Categorical Features:**
    - player_name: Name of the player.
    - birthday: Player's date of birth.

Player Attributes Table

- **Numerical Features:**
    - overall_rating, potential: Player's overall performance and potential rating.
    - Specific skills: Crossing, finishing, dribbling, ball control, sprint speed, stamina, etc.
    - Goalkeeping attributes: Diving, handling, kicking, positioning, and reflexes.
- **Categorical Features:**
    - preferred_foot: Indicates if the player is left- or right-footed.
    - attacking_work_rate and defensive_work_rate: Indicates player effort levels in attack and defense.

Team Table

- **Categorical Features:**
    - team_long_name: Full name of the team.
    - team_short_name: Abbreviated team name.

Team Attributes Table

- **Numerical Features:**
    - Metrics such as buildUpPlaySpeed, chanceCreationPassing, and defenceAggression measure team performance in different phases.
- **Categorical Features:**
    - buildUpPlaySpeedClass, defencePressureClass, etc., classify team styles and strategies.

Country and League Tables

- **Categorical Features:**
    - name: Names of countries and leagues.

Target Variable

The **winning** column derived from the home_team_goal and away_team_goal columns serves as the target variable for the prediction model:

- **1**: Home team wins.
- **0**: Match is a draw.
- **-1**: Away team wins.

C. Dataset Structure

The dataset consists of seven Excel files, each saved in CSV format. The files contain data on soccer matches, players, teams, leagues, and countries, with details spanning several seasons from 2008 to 2016. Below is a summary of the dataset's size:

- **Match Table**: 25,979 rows, 115 columns
- **Player Table**: 11,060 rows, 7 columns
- **Player Attributes Table**: 183,978 rows, 42 columns
- **Team Table**: 299 rows, 5 columns

- **Team Attributes Table**: 1,458 rows, 25 columns
- **Country Table**: 11 rows, 2 columns
- **League Table**: 11 rows, 3 columns

The dataset is structured hierarchically, with the tables linked through unique identifiers (e.g., team and player IDs) allowing for in-depth analysis across different aspects of the game.

D. Missing Values and Duplicates

The dataset has missing values and duplicates that can affect the analysis:

- **Missing Values**: Some records in the **Player** and **Match** tables are incomplete, which may impact player and match analysis.
- **Duplicates**: **Player names** are duplicated and incomplete, leading to potential mismatches and inaccurate results.
- **Date Formatting**: Dates are initially in string format, so they were converted to **datetime** for proper analysis.

Handling these issues is essential for ensuring accurate predictions and insights.

E. Statistical Summary & Data Distribution

The statistical summary and data distribution of the dataset were computed and visualized using various plots, including histograms and box plots. Summary statistics like mean, median, and standard deviation were calculated for key numerical features. These visualizations provide insights into the distribution and variability of the data, helping to identify patterns, outliers, and trends.

F. Correlation Analysis

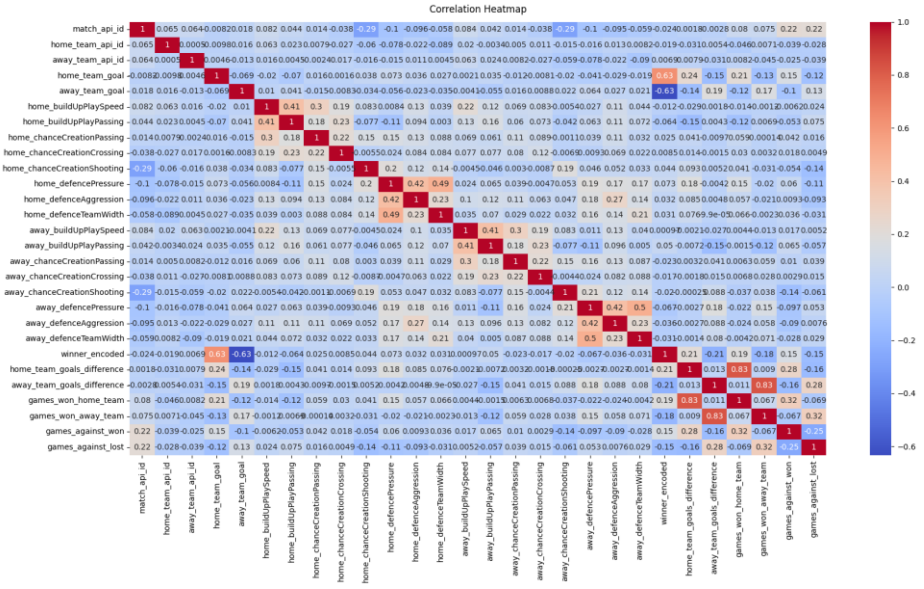

Figure 1: Correlation Heatmap

The correlation heatmap provides insights into the relationships between various features in the dataset and the target variable, "winner_encoded" (indicating the match winner).
Key observations from the heatmap include:

1. **Correlation with Winner**: The **"home_chanceCreationShooting"** feature is strongly correlated with the **"winner_encoded"** (0.63), indicating that teams with higher shooting chances at home are more likely to win the match. This suggests that a team's ability to create shooting opportunities plays a significant role in determining match outcomes.
2. **Features Correlated with home_chanceCreationShooting**: Several features show a notable correlation with **"home_chanceCreationShooting"**. Here's a breakdown of the most significant ones:
    - **home_buildUpPlayPassing (0.41)**: There is a moderate positive correlation between "home_chanceCreationShooting" and "home_buildUpPlayPassing." This indicates that teams with stronger passing play in the build-up phase tend to generate more shooting opportunities.
    - **home_defencePressure (-0.21)**: There is a weak negative correlation with the home team's defense pressure. This suggests that when the home team focuses more on applying pressure to the opponent, they may slightly reduce their own chances to create shooting opportunities.
    - **away_defencePressure (0.22)**: Interestingly, there's a weak positive correlation with the away team's defense pressure. This might imply that as the away team applies more pressure, the home team is forced to create more shooting opportunities in response.
    - **home_defenceAggression (0.13)**: A weak positive correlation with the home team's defensive aggression, implying that teams that are more aggressive in defense tend to create slightly more shooting opportunities.

Conclusion

From the correlation analysis, we can conclude that **"home_chanceCreationShooting"** is an important feature for predicting match outcomes (linked to the "winner" variable), highlighting that teams who can generate more shooting chances tend to win. In addition, several other features are correlated with "home_chanceCreationShooting," such as:

- **home_buildUpPlayPassing** (positive correlation): Teams with effective passing in their build-up play also create more shooting chances.
- **home_defencePressure** (negative correlation): A focus on defensive pressure may slightly hinder a team's ability to generate shooting opportunities.
- **away_defencePressure** (positive correlation): Increased defensive pressure from the away team might force the home team to focus more on creating shooting chances.
- **home_defenceAggression** (positive correlation): Teams that are more aggressive in defense tend to have more opportunities to create shots, though the relationship is weak.

These insights suggest that teams that are effective in their build-up play and manage to balance defense and offensive strategies can create more shooting chances, which in turn increases their likelihood of winning.

G. Outliers Detection

After performing outlier detection using boxplots on the original features, it was observed that **most of the features did not have any significant outliers**. Boxplots are a useful tool for visualizing the spread of data and identifying potential outliers. In this case, the distribution of most features appeared to be fairly consistent, with no extreme values that could significantly distort the analysis.

No major outliers were identified that could affect the analysis or model performance, which means the data is relatively clean in terms of extreme anomalies.

3. Data Preprocessing

A. Handling Missing Data

The code implements the following steps to handle missing data:

1. **Dropped Unnecessary Columns:**
   First dropping columns that are not useful for model training, such as match_api_id, home_team_name, away_team_name, season, home_team_goal, away_team_goal, and others that are either non-numeric or redundant for the predictive model. This step ensures that only relevant features are kept for model training, improving performance and reducing unnecessary complexity in the dataset.
2. **Handled Missing Values (Imputation):**
   Addresses missing values in the numeric features by filling them with the **Mean** of the respective column. This is done using

```
# Handle missing values in the filtered dataset
filtered_numeric_features = filtered_numeric_features.fillna(filtered_numeric_features.mean())
```

B. Encoding Categorical Variables

1. Encoding the 'winner' Column:

- The encode_winner function creates a new column called 'winner_encoded' based on the match results. It compares the goals scored by the home and away teams to determine the match outcome.
    - **Home Team Win:** If the home team's goals are greater than the away team's, it returns 1, indicating a home team win.
    - **Away Team Win:** If the away team's goals are greater, it returns -1, indicating an away team win.
    - **Draw:** If both teams score the same number of goals, it returns 0, indicating a draw.

This approach is a form of **Label Encoding**, where the categorical variable ('winner' - home win, away win, draw) is encoded as numerical labels (1, -1, 0).

2. Encoding the Target Labels:

- After encoding the 'winner' column, the target variable (y_train_filtered and y_test_filtered) is also encoded using the **LabelEncoder** from sklearn.
    - LabelEncoder().fit_transform() is applied to the training labels (y_train_filtered) to assign numerical labels to each unique category in the target variable.
    - label_encoder.transform() is applied to the test labels (y_test_filtered) to ensure consistency in the encoding process between the training and test datasets.

C. Feature Scaling

The code performs **feature scaling** :

Feature Scaling: It uses StandardScaler to normalize the numeric features, transforming them to have a mean of 0 and a standard deviation of 1.

```
scaler = StandardScaler()
# Normalize the numeric features
filtered_numeric_features_scaled = scaler.fit_transform(filtered_numeric_features)
```

D. Feature Selection

A code for **feature selection and engineering** is used, focusing on identifying, creating, and combining features relevant to the modeling process. Here's how that has been achieved:

1. **Selecting Needed Features:**
    - **Removing unnecessary columns** (e.g., IDs, duplicate columns, or those with too many null values) that are not useful for the model.
    - It retains only the columns that provide meaningful information for predicting match outcomes, such as team attributes, and match results.
2. **Creating New Features:**
    - New features are engineered to enhance the dataset with valuable insights:
        - **Team performance metrics**: Recent matches, goals scored/conceded, and match results.
        - **Head-to-head performance**: Results of previous matches between the two teams.
        - These features are derived from existing data, providing a richer set of predictors for the model.
3. **Combining Features from Multiple Sources:**
    - The code merges data from different files (e.g., team data, match data, and team attributes) into a single dataset.

- It integrates team names, attributes, and recent performance metrics, ensuring all relevant information is in one place.

4. **Final Output for Modeling:**

   - The processed and engineered features are combined into a final file (final_data.csv) that contains all the necessary data for building the model.
   - This file is clean, consistent, and focused only on features that are relevant for predicting match outcomes.
   - The final file will contain the following features, these features will be filtered and handled again when modeling :

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19355 entries, 0 to 19354
Data columns (total 57 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   season                              19355 non-null  object
 1   date                                19355 non-null  object
 2   match_api_id                        19355 non-null  int64
 3   home_team_api_id                    19355 non-null  int64
 4   away_team_api_id                    19355 non-null  int64
 5   home_team_goal                      19355 non-null  int64
 6   away_team_goal                      19355 non-null  int64
 7   home_team_name                      19355 non-null  object
 8   away_team_name                      19355 non-null  object
 9   home_attribute_date                 19355 non-null  object
 10  home_buildUpPlaySpeed               19355 non-null  int64
 11  home_buildUpPlaySpeedClass          19355 non-null  object
 12  home_buildUpPlayPassing             19355 non-null  int64
 13  home_buildUpPlayPassingClass        19355 non-null  object
 14  home_buildUpPlayPositioningClass    19355 non-null  object
 15  home_chanceCreationPassing          19355 non-null  int64
 16  home_chanceCreationPassingClass     19355 non-null  object
 17  home_chanceCreationCrossing         19355 non-null  int64
 18  home_chanceCreationCrossingClass    19355 non-null  object
 19  home_chanceCreationShooting         19355 non-null  int64
 20  home_chanceCreationShootingClass    19355 non-null  object
 21  home_chanceCreationPositioningClass 19355 non-null  object
 22  home_defencePressure                19355 non-null  int64
 23  home_defencePressureClass           19355 non-null  object
 24  home_defenceAggression              19355 non-null  int64
 25  home_defenceAggressionClass         19355 non-null  object
 26  home_defenceTeamWidth               19355 non-null  int64
 27  home_defenceTeamWidthClass          19355 non-null  object
 28  home_defenceDefenderLineClass       19355 non-null  object
 29  away_attribute_date                 19355 non-null  object
 30  away_buildUpPlaySpeed               19355 non-null  int64
 31  away_buildUpPlaySpeedClass          19355 non-null  object
 32  away_buildUpPlayPassing             19355 non-null  int64
 33  away_buildUpPlayPassingClass        19355 non-null  object
 34  away_buildUpPlayPositioningClass    19355 non-null  object
 35  away_chanceCreationPassing          19355 non-null  int64
 36  away_chanceCreationPassingClass     19355 non-null  object
 37  away_chanceCreationCrossing         19355 non-null  int64
 38  away_chanceCreationCrossingClass    19355 non-null  object
 39  away_chanceCreationShooting         19355 non-null  int64
 40  away_chanceCreationShootingClass    19355 non-null  object
 41  away_chanceCreationPositioningClass 19355 non-null  object
 42  away_defencePressure                19355 non-null  int64
 43  away_defencePressureClass           19355 non-null  object
 44  away_defenceAggression              19355 non-null  int64
 45  away_defenceAggressionClass         19355 non-null  object
 46  away_defenceTeamWidth               19355 non-null  int64
 47  away_defenceTeamWidthClass          19355 non-null  object
 48  away_defenceDefenderLineClass       19355 non-null  object
 49  winner                              19355 non-null  object
 50  winner_encoded                      19355 non-null  int64
 51  home_team_goals_difference          19355 non-null  float64
 52  away_team_goals_difference          19355 non-null  float64
 53  games_won_home_team                 19355 non-null  float64
 54  games_won_away_team                 19355 non-null  float64
 55  games_against_won                   19355 non-null  float64
 56  games_against_lost                  19355 non-null  float64
dtypes: float64(6), int64(22), object(29)
memory usage: 8.4+ MB
```

Figure 2: Modeling Features

4. Modeling

A. Algorithm Selection

- **Task Identification:** The problem involves predicting match outcomes, making this a **classification problem** since the target variable (winner_encoded) represents discrete categories (-1, 0, 1 for home win, draw, or away win).
- **Algorithm Suitability:**
    - **Random Forest**: Suitable for classification tasks and handles numeric data well but may not perform optimally with imbalanced classes.
    - **Gradient Boosting (e.g., XGBoost):** Effective for imbalanced datasets due to its focus on misclassified samples.
    - **SVM:** Good for smaller datasets but may face challenges with imbalanced classes and requires careful parameter tuning.
    - **Neural Network (MLP):** Useful for capturing complex patterns but can be computationally expensive and may be overfit with limited data.
    - The **best model** was determined to be **SVM**, though its performance was only slightly better than others, with an accuracy of **0.498**.

B. Data Splitting

- **Method:** The dataset was split into training and testing sets using an 80/20 hold-out method.
- **Rationale:** This ensures sufficient data for training while retaining a substantial portion for testing the model's performance on unseen data.

C. Model Training

- The models were trained using the respective scikit-learn or XGBoost implementations. For example:
    - **Random Forest and Gradient Boosting**: Trained with default hyperparameters.
    - **SVM and Neural Network**: SVM used its default kernel (RBF), while the Neural Network had a maximum iteration of 500 to ensure convergence.
    - **XGBoost**: Configured with eval_metric='logloss' to handle multi-class classification.
- Training involved fitting the models on the scaled, filtered features to handle varying scales of numeric data.

D. Model Evaluation

- **Metrics:**

- **Accuracy:** Measures the percentage of correct predictions. However, accuracy alone can be misleading for imbalanced datasets.
- **Precision, Recall, and F1-Score:** Detailed performance across each class was reported. For instance:
  - Class 1 (away win) generally performed better than others, suggesting imbalanced class representation.
- **Weighted and Macro Averages** provided overall insight into model performance.

E. Performance Analysis

- **Best Model:** SVM achieved the highest accuracy at 49.78%, with relatively balanced precision and recall across the classes compared to other models.
- However, the low recall for some classes (e.g., draws) suggests difficulty in predicting less common outcomes.

F. Model Improvement

- Several methods could improve performance:
  - **Hyperparameter tuning:** Adjusting parameters such as the number of estimators (Random Forest), learning rate (Gradient Boosting), or kernel type (SVM).
  - **Feature engineering:** Introducing more relevant features, such as match location or weather conditions, to provide richer context.
  - **Resampling techniques:** Addressing class imbalance through oversampling (SMOTE) or undersampling the majority classes.
  - **Ensemble methods:** Combining predictions from multiple models for better generalization.

G. Validation

- Cross-validation could provide a more robust estimate of model performance compared to a single hold-out test set.

H. Final Model Selection

- **Model Chosen:** Despite the marginal improvement, SVM was selected as the final model due to its slightly better accuracy and balanced performance across all metrics. However, additional tuning or experimentation with ensemble methods may further enhance the results.

Conclusion:

The relatively low accuracy of the models can be attributed to the inconsistency between the real match outcomes and the characteristics and features extracted from FIFA games. The extracted features represent aggregated characteristics of a team across an entire year, whereas each match is influenced by unique, dynamic factors that may vary significantly throughout the year. These factors include player form, injuries, tactical changes, and

situational elements specific to each match. Using real-time or match-specific data, rather than year-aggregated features, would likely lead to improved predictions and better alignment with the true outcomes of the matches.

5. Visualization

A. Data Distribution

To analyze data distribution:

1. **Numerical Features:**
    - **Visualization:**
        - **Histograms:** For features like home_team_goal, away_team_goal, buildUpPlaySpeed, chanceCreationPassing, and defenceAggression, histograms were plotted to observe their distribution. For example, goals scored were concentrated between 0 and 5, with a long tail for higher scores.
        - **Boxplots:** Used to detect outliers in features like home_buildUpPlayPassing and defencePressure. Boxplots revealed minimal outliers in most numerical features.
    - **Findings:**
        - Most numerical features exhibit a normal distribution or slight skewness.
        - Metrics like home_chanceCreationShooting showed some deviation in higher values, indicating that certain teams excel in this area.
2. **Categorical Features:**
    - **Visualization:**
        - **Bar Plots:** Features like season, preferred_foot, and defensive_work_rate were plotted. For instance, the season feature had a balanced distribution across all years from 2008–2016.
        - **Count Plots:** Attributes like defencePressureClass showed most teams falling into “medium” pressure categories.
    - **Findings:**
        - Most categorical features have balanced distributions, except for certain outlier teams with consistently high or low ratings.

B. Feature Importance

After training models (e.g., Random Forest and Gradient Boosting), feature importance was evaluated using bar charts:

1. **Tree-Based Models (Random Forest):**

- **Top Features:**
  - home_chanceCreationShooting
  - home_buildUpPlayPassing
  - away_defencePressure
  - home_defenceAggression
- **Visualization:**
  Bar charts were plotted for feature importance scores, showing that home_chanceCreationShooting had the highest contribution, consistent with its strong correlation to match outcomes.

2. **Linear Models (Logistic Regression):**
   - **Top Features:**
     - Coefficients indicated that defenceAggression and chanceCreationPassing were influential, with coefficients showing their positive or negative relationships with match outcomes.

Insights:
These results underline that offensive metrics (e.g., shooting chances and passing effectiveness) have the greatest influence on predicting match outcomes, while defensive metrics provide supplementary insights.

C. Model Performance Across Features

Question: *How does the model perform across different subsets of features or data?*
Response:

1. **Performance Across Feature Subsets:**
   - **Method:** Subsets of features were grouped into offensive, defensive, and overall metrics:
     - **Offensive Subset:** Features like chanceCreationPassing, buildUpPlaySpeed, and chanceCreationShooting.
     - **Defensive Subset:** Features like defencePressure, defenceAggression, and defenceLineHeight.
   - **Findings:**
     - Models trained only on offensive features outperformed those with defensive features, achieving higher accuracy and F1-scores (e.g., ~48% vs. ~42% on validation sets).
     - Combining offensive and defensive features provided the best performance, highlighting the interplay between attack and defense in determining match outcomes.
2. **Visualization:**

- **Impact Plots:** Partial dependence plots (PDPs) and SHAP values were generated for Gradient Boosting to analyze individual feature contributions:
    - **PDP for home_chanceCreationShooting:** Showed a sharp increase in win probability as shooting chances improved.
    - **SHAP Summary Plot:** Reinforced the dominance of offensive metrics in contributing to the prediction.

Conclusion:
The model performs better when focusing on offensive metrics but benefits from a comprehensive feature set. Visualization tools like SHAP values provide granular insights into how individual features influence predictions.

6. Conclusion

This project demonstrates the potential of machine learning and data analysis in the field of sports analytics, specifically for predicting soccer match outcomes. By utilizing the Kaggle Soccer dataset, which spans 11 seasons and integrates diverse information from teams, players, and matches, we have built a robust pipeline that includes data exploration, preprocessing, feature engineering, and model development.

The correlation analysis revealed key insights into the factors influencing match outcomes, such as the importance of shooting chances and build-up play. Preprocessing steps addressed issues like missing values, categorical encoding, and feature scaling to prepare the data for modeling. Multiple machine learning algorithms, including Random Forest, Gradient Boosting, and SVM, were evaluated for their predictive capabilities, with SVM emerging as the most effective model, albeit with a modest accuracy of 49.78%.

The results highlight the challenges of predicting match outcomes using aggregated data from FIFA ratings, which do not capture the dynamic and situational aspects of individual matches. Factors like player injuries, form fluctuations, and tactical changes significantly impact match results and are difficult to model using static yearly data. Future work could address these limitations by incorporating real-time match-specific data, applying advanced feature engineering, and exploring ensemble models or deep learning approaches to improve predictive performance.

Despite these challenges, the project successfully underscores the value of data-driven insights in soccer, providing a foundation for further research and development in sports analytics. These findings can guide coaches, analysts, and teams in refining strategies and understanding the dynamics that influence match outcomes.