**IPL Match Winner Prediction-ML**

**Objectives:**

The objective of this project is to build a machine learning model that accurately predicts the winner of an Indian Premier League (IPL) match based on historical match data. This involves collecting and preprocessing the data, exploring key patterns, training multiple classification algorithms, tuning hyperparameters, and selecting the best-performing model for reliable prediction.

**Problem Definition:**

The goal is to develop a machine learning model that predicts the winner of an IPL match using historical data. The model will consider factors like the teams playing, toss winner, toss decision, venue, and other match-related features to classify the winning team before the match begins.

**Dataset Overview:**

The IPL dataset contains 756 rows and 18 columns, representing historical match records from various seasons. It includes key information and features by combining both categorical and numerical features.

* Season: The year of the IPL season
* City: The city where the match was played.
* Date: The date of the match.
* Team1 and Team2: The two teams playing in the match.
* Toss Winner and Toss Decision: The team that won the toss and the decision they made (whether to bat or bowl).
* Result: The Resutl of the match (win or loss).
* DL applied: Indicates if the Duckworth-Lewis method was applied due to interruptions.
* Winner: The team that won the match.
* Win by runs and Win by wickets: The margin of victory by runs or wickets.
* Player of Match: The player awarded for exceptional performance.
* Venue: The stadium where the match was held.
* Umpires: The officials overseeing the match.

**Tools & Technologies Used:**

* To implement the IPL match winner prediction model, a range of tools and libraries were used for data analysis, visualization, preprocessing, model building, and evaluation. The entire workflow was developed using Python in an interactive environment.

 Language: Python

 Libraries: pandas, numpy, seaborn, matplotlib, scikit-learn, xgboost

 IDE: Jupyter Notebook

**Data Preprocessing:**

* The dataset underwent several preprocessing steps to prepare it for model training.
* Irrelevant columns like id, player\_of\_match, and umpire3 were dropped to reduce noise.
* Missing values in columns like winner and city were handled appropriately.
* Categorical features such as team1, team2, toss\_winner, and venue were encoded using one-hot encoding, while binary choices like toss\_decision were encoded using label encoding.
* The final dataset included only meaningful features relevant for predicting match outcomes.

**Exploratory Data Analysis:**

* Exploratory Data Analysis was performed to uncover patterns and insights from the IPL dataset.
* It was observed that Mumbai Indians had the highest number of wins across seasons.
* Teams that won the toss also won the match in approximately 52% of the games.
* Bowling first was the more frequent toss decision (around 61%).
* The most common victory margin was by 6 wickets, and most matches were held in Mumbai.
* Visualizations such as bar charts, count plots, and pie charts were used to represent team performance, toss impact, and venue statistics.  
  These insights helped validate the importance of specific features for model training.

**Insights from EDA:**

* Mumbai Indians emerged as the most successful team with 110 wins, closely followed by Chennai Super Kings with 100 wins.
* Around 61% of teams opted to bowl after winning the toss, indicating a strategic preference for chasing targets.
* Teams that won the toss had a 52% probability of winning the match, suggesting a moderate impact of the toss on the outcome.
* Mumbai City hosted the highest number of matches (100), followed by Kolkata and Delhi.
* There was a positive correlation between toss winner and match winner, implying that winning the toss may influence the final result.

**Encoding Techniques:**

* Since the dataset contained multiple categorical features, encoding was necessary to convert them into a numerical format suitable for machine learning models. Two encoding techniques were used:
* Label Encoding was applied to binary categorical columns such as toss decision, where the values (e.g., "bat", "field") were converted into integers (0, 1).
* One-Hot Encoding was used for multi-class categorical features like team1, team2, venue, toss winner, and winner. This created separate binary columns for each category, ensuring the model doesn't assume any ordinal relationship between them.

**Model Building and Evaluation:**

**S**everal supervised machine learning models were trained and evaluated to predict the match winner.

The following classification models were implemented:

* Logistic Regression
* Support Vector Machine (SVM)
* K-Nearest Neighbors (KNN)
* Decision Tree
* Random Forest
* XGBoost

Each model was trained using the preprocessed dataset and evaluated on the test set using key metrics such as:

* Accuracy
* Precision
* Recall
* F1 Score

Among all models, **XGBoost** delivered the best performance across all evaluation metrics, making it the most reliable choice for the prediction task.

**Hyperparameter Tuning:**

To enhance model performance, GridSearchCV was used to tune the hyperparameters of the top-performing model — XGBoost.

A parameter grid was defined with combinations of:

* Learning rate: [0.01, 0.1, 0.2]
* Max depth: [3, 5, 7]
* N estimators: [50, 100, 150]

The tuning process applied 3-fold cross-validation to find the best parameter set based on accuracy.

**Best Parameters Selected**:

* learning\_rate = 0.01
* max\_depth = 5
* n\_estimators = 150

After tuning, the XGBoost model achieved a significant performance improvement, reaching an accuracy of 98%, with strong precision, recall, and F1 scores as well.

**Final Model Summary:**

* After evaluating multiple classification models and tuning their hyperparameters, XGBoost was selected as the final model for IPL match winner prediction.
* It outperformed other models in terms of accuracy and generalization due to its ability to handle categorical features, built-in regularization, and gradient boosting framework.

Final Model Performance (Tuned XGBoost):

* Accuracy: 0.980
* Precision: 0.977
* Recall: 0.980
* F1 Score: 0.977

A performance comparison chart and confusion matrix were used to visualize and confirm the superiority of XGBoost over other models.

**Future Improvements:**

* While the current model performs with high accuracy, there are several areas where the project can be enhanced further:
* Include More Features: Incorporating player-specific statistics (e.g., top batsman, bowler performance) and match conditions (e.g., weather, pitch report) could improve prediction accuracy.
* Use More Recent Seasons: Training the model on the latest IPL seasons would make predictions more relevant and adaptive to recent trends.
* Handle Class Imbalance: If certain teams are underrepresented in the dataset, applying techniques like SMOTE or class weights could balance the learning process.

**Conclusion:**

* This project successfully demonstrates how machine learning can be applied to predict the outcomes of IPL matches using historical match data.
* By preprocessing the data, performing exploratory analysis, testing multiple classification models, and tuning hyperparameters, the final XGBoost model achieved an impressive accuracy of 98%.
* The model’s performance was validated using various metrics such as precision, recall, and F1 score, confirming its reliability.
* With further enhancements like additional feature integration, real-time data, and model deployment, this project can evolve into a practical tool for sports analysts and enthusiasts.