IPL Score Prediction Using XGBoost Regressor(Nikhil Sharma,Jitesh Maurya,Kaavya Dhir)

# 1. Introduction & Problem Statements

This project aims to predict the final runs, wickets, and dot balls in an IPL match using ball-by-ball data. It employs machine learning techniques, specifically the XGBoost Regressor, to make accurate predictions based on features like overs, batting and bowling teams, and current match statistics.

This project aims to build an IPL score prediction model using a Random Forest Regressor, focusing on forecasting:

- Final score of the innings

- Total wickets lost

- Total boundaries scored

- Total dot balls bowled

The model utilizes ball-by-ball IPL match data to make accurate predictions, providing a

**data-driven approach** to understanding match scenarios.

## 2. Project Objective

• Build a predictive model to estimate final scores in an IPL match using historical data.  
• Predict key match metrics like runs, wickets, and dot balls based on current in-game parameters.  
• Implement an end-to-end ML pipeline including preprocessing, training, evaluation, and deployment using Flask.

## 3. Technology Stack

• Python: Core scripting language  
• pandas, numpy: Data manipulation  
• matplotlib, seaborn: Data visualization  
• scikit-learn: Model evaluation and data splitting  
• xgboost: Regression modeling  
• joblib: Saving trained models  
• Flask: Backend web framework for deploying predictions

## 4. Data Preprocessing

The dataset includes ball-by-ball details such as date, match ID, innings, teams, overs, and cumulative scores. Preprocessing steps include:  
• Renaming columns  
• Filtering consistent teams  
• Handling missing values  
• Merging final scores and dropping duplicates  
• One-hot encoding of categorical variables

• Train-Test Split(80% train&20% test)

## 5. Modeling with XGBoost

The XGBoost Regressor is used due to its high performance with tabular data. Hyperparameters used:  
• n\_estimators: Number of trees (200 for runs, 300 for wickets)  
• learning\_rate: Controls the step size (0.1)  
• max\_depth: Maximum depth of trees (6 for runs, 4 for wickets)  
Separate models are trained for runs and wickets. Dot balls are estimated from remaining deliveries.

## 6. One-hot Encoding

One-hot encoding is used to convert categorical data (like team names) into a format that can be provided to ML algorithms. It creates binary columns for each category, ensuring that models treat them without implicit ordering.

## 7. Evaluation Metrics

• MAE (Mean Absolute Error): Average of absolute errors between predicted and actual values  
• MSE (Mean Squared Error): Average of squared differences; penalizes large errors  
• RMSE (Root Mean Squared Error): Square root of MSE; gives error in original units (e.g., runs)  
• R² Score: Measures how well predictions approximate actual results

## 8. Prediction Output Ranges

• Final Runs: Range from 30 to 250 runs  
• Wickets: Range from 0 to 10 wickets  
• Dot Balls: Estimated based on remaining deliveries

## 9.Challenges & Future Improvements

Challenges Faced:

**- Handling massive data and ensuring proper feature transformations.**

**- Balancing model complexity and interpretability.**

**- Accounting for external factors (weather, pitch conditions) that influence IPL matches.**

## Future Enhancements:

- Implementing \*Neural Networks (LSTMs)\* for time-series cricket prediction.

- Player-specific impact analysis (e.g., batsman form, bowler economy rate).

- Win probability estimation based on real-time match updates.

## 10. Target Variables

• final\_score: Predicted total runs at the end of the innings  
• wickets: Predicted number of wickets  
• dot\_balls: Estimated number of dot balls from ball tracking

## 11. Conclusion & Learnings

This project successfully demonstrates how \*machine learning can predict IPL match outcomes\* using \*ball-by-ball data\*. The model provides real-time match insights, making it valuable for analysts and broadcasters.

Key takeaways:

**ML models improve cricket analytics** by leveraging historical data.

**Feature selection is crucial** for accurate predictions.

**Future enhancements can incorporate advanced models** for better real-time performance.

## References

- Kaggle IPL Dataset

- scikit-learn Documentation

-Supervised Machine Learning: Regression and Classification <https://www.coursera.org/learn/machine-learning>

-Videos that break down complex ML concepts into simpler explanations.

<https://www.youtube.com/user/joshstarmer>

Code and Flask guide

<https://www.youtube.com/live/4CtyDxfhoN8?si=3MXfWRfMkCHdVFqJ>