CrickXpert

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

TEAM MEMBERS

Manan Chugh

2021335

manan21335@iiitd.ac.in

Abhijeet Anand

2021509

abhijeet21509@iiitd.ac.in

Kartikey Dhaka

2021534

kartikey21534@iiitd.ac.in

Abhijay Singh

2021226

abhijay21226@iiitd.ac.in

MOTIVATION

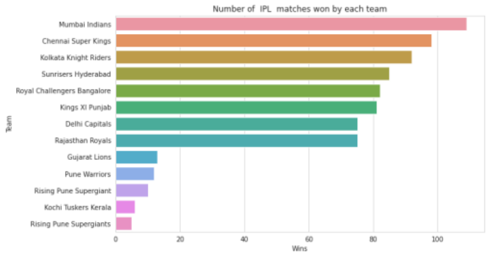
Merging our love for sports and technology, predicting cricket scores with machine learning captivates us. Unveiling data trends for match outcomes enriches our cricket passion. This project sharpens skills, pioneers innovation, and adds thrill to the game. Our shared ambition drives this journey.

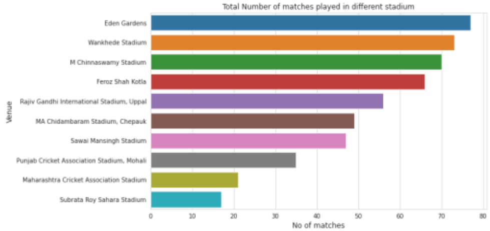
Hence, in this project, we aim to build models that can predict key metrics in a IPL cricket match.

LITERATURE REVIEW

The paper on "Machine Learning Applications in Baseball" discusses the utilization of machine learning in baseball analytics, particularly focusing on Regression, Binary, and Multiclass Classification problems. It highlights the dominance of Support Vector Machines and k-nearest neighbors in the literature and suggests a potential increase in the use of neural networks for future analyses.

https://www.tandfonline.com/doi/epdf/10.1080/08839514.2018.1442991?needAccess=true





The paper investigating "Sport analytics for cricket game results using machine learning" explores the use of machine learning techniques to predict IPL cricket match outcomes. It details the process of feature selection and the application of models like Naïve Bayes and Random Forest, noting the latter's superior performance in accuracy and recall, especially for features based on the home team advantage.

https://www.emerald.com/insight/content/doi/10.1016/j.aci.2019.11.006/full/pdf?title=sport-analytics-for-cricket-game-results-using-machine-learning-an-experimental-study

DATASET DESCRIPTION (Dataset Link)

Model 1:

The dataset consists of the data of IPL matches from 2007 to 2017. The dataset is of 76015x15 size. It consists of the following columns

- **Mid:** Match identifier number.
- **Date:** Date of the match.
- **Venue:** Stadium where the match was played.
- **Bat Team:** Batting team's name.
- **Bowl Team:** Bowling team's name.
- **Batsman:** Name of the batsman playing.
- **Bowler:** Name of the bowler bowling.
- **Runs:** Runs scored by the batsman.
- **Wickets:** Number of wickets taken by the bowler.
- **Overs:** Number of overs bowled.
- **Runs_Last_5:** Runs scored in the last 5 overs.
- **Wickets_Last_5:** Wickets taken in the last 5 overs.
- **Striker:** Runs scored by the batsman currently facing.
- **Non-Striker:** Runs scored by the batsman not facing.
- **Total:** Total runs scored by the batting team in the match.

We scraped the internet for IPL matches data and we were able to collect the data for all the IPL matches from 2008 - 2017. After extracting this data we were left with 75000+ rows to work upon.

For preprocessing: For optimal results we deleted all the rows which had any missing column in them. Then we thoroughly analyzed the data and came upon a conlusion that only 11 features were important for our model development("mid", "date", "bat_team", "bowl_team", "runs", "wickets", "overs", "runs_last_5", "wickets_last_5", "total", "runs_in_each_ball"). Out of the 14, We have singled out 8 consistent team namely, Kolkata Knight Riders, Chennai Super Kings, Rajasthan Royals, Mumbai Indians, Kings XI Punjab, Royal Challengers Bangalore, Delhi Daredevils, Sunrisers Hyderabad to train our model on hence reducing our dataset size from 76014x11 to 53811x11 improving our accuracy. We took more than or equal to 5 overs because the first 5 overs constitute the powerplay in the IPL. Consequently, performance during this period is inconsistent, with the potential for either a substantial number of runs being scored, a very low score, or the loss of a significant number of wickets. For more accurate results we computed the feature "runs_in_each_ball" and added it to our dataset. For better comparison of the dates we converted the "date" feature from String datatype to Date_Time datatype. We used One Hot Encoding to encode the categorical data ("bat_team", "bowl_team")

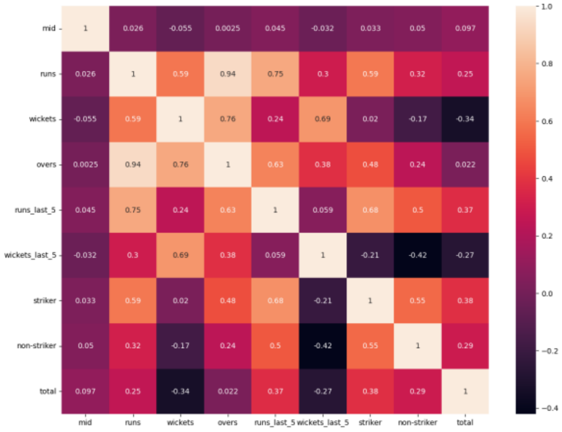
DATASET DESCRIPTION (Dataset Link)

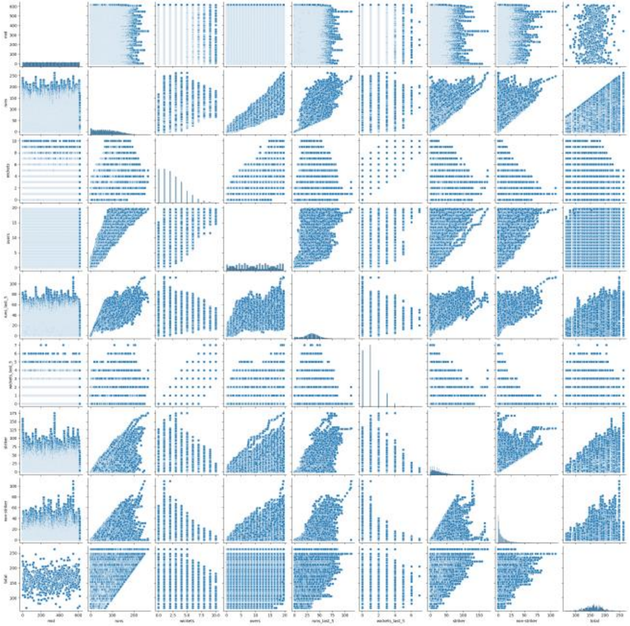
Model 2:

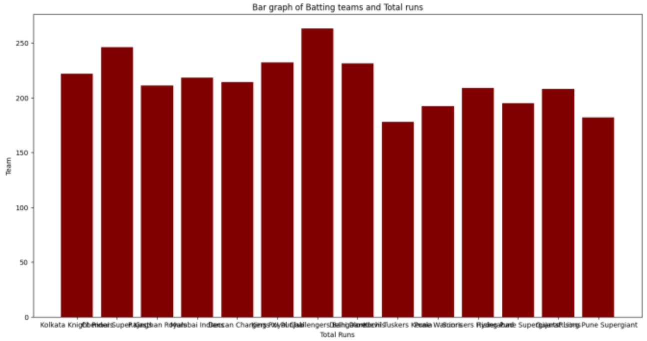
Preprocessing:

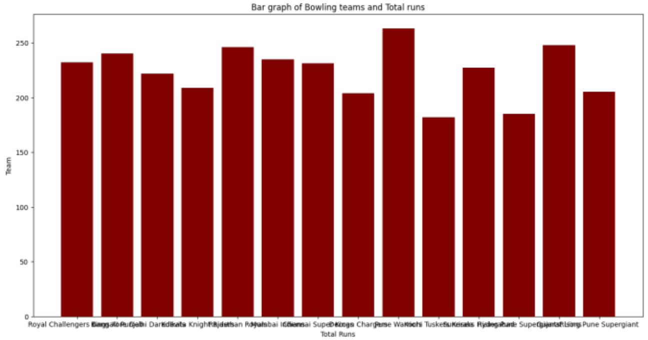
- **Computing Runs in Each Ball:** A new column, 'runs_in_each_ball', is initialized to 0 in the dataset to store the runs scored on each ball. If there is a change in the 'mid' column, the 'previous_runs' is reset to 0.The difference between the 'runs' for the current ball and 'previous_runs' is calculated and stored in 'runs_in_each_ball'.The 'previous_runs' variable is updated with the 'runs' of the current ball for the next iteration.
- **Cleaning Data:** Rows with NULL values are removed to ensure the dataset is clean and suitable for modeling.
- **Selecting Relevant Features:** A subset of columns is selected for further analysis. These columns include 'bat_team', 'bowl_team', 'runs', 'wickets', 'overs', 'runs_last_5', and 'runs_in_each_ball', indicating that these are the features considered most relevant for the predictive model.
- **Handling Categorical Data:** Categorical variables ('bat_team' and 'bowl_team') are transformed using one-hot encoding.
- **Feature Scaling:** The dataset is scaled using StandardScaler. Scaling features to a range is a common practice to normalize the range of independent variables or features of data.
- **Training Classifiers**: The preprocessed dataset is then used to train three different classifiers: Logistic Regression, Random Forest, and Multilayer Perceptron (MLP). These models are chosen to compare their performances and predict the outcomes.

DATASET DESCRIPTION





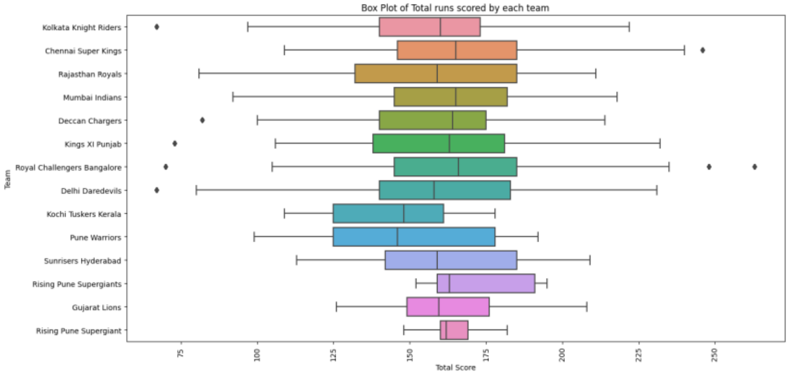


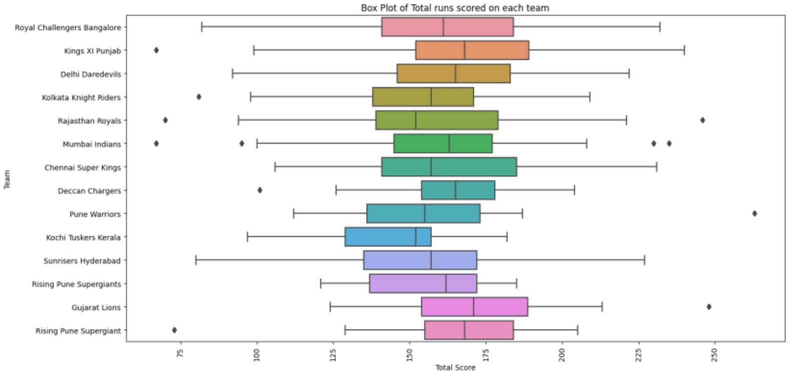


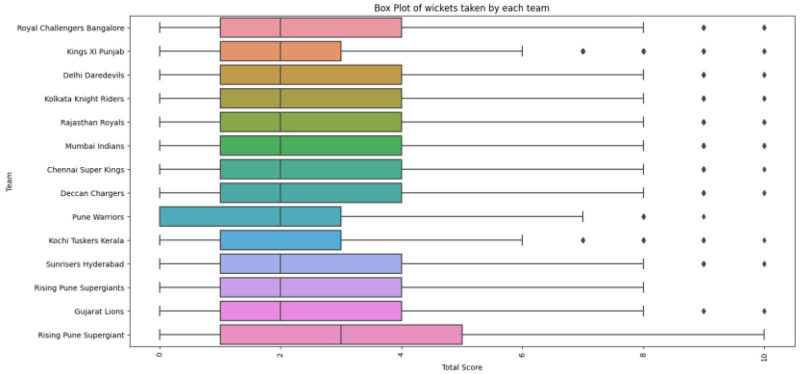
- Runs and overs show very high positive correlation.
- Wickets and overs also show high positive correlation.
- Wickets and total runs are negatively correlated.

- Runs and overs show linear dependency to some extent.

DATASET DESCRIPTION

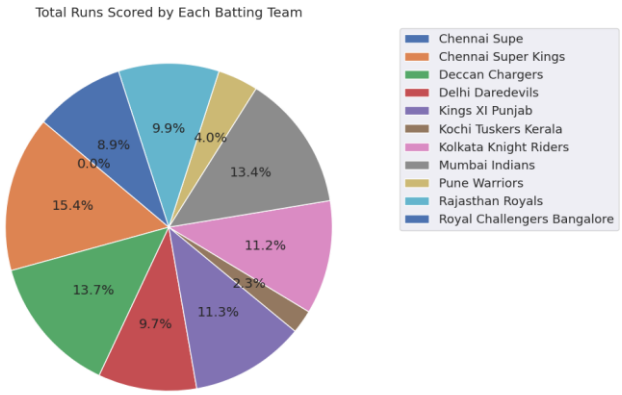


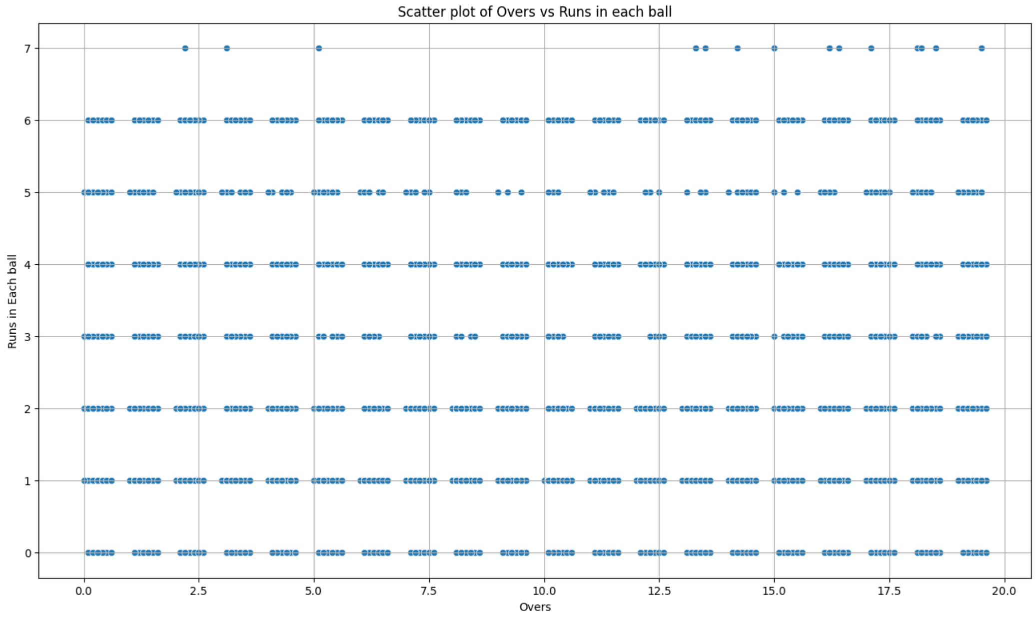




- The median runs scored by each team is in the range of 150-175.
- Matches in which all 10 wickets fall are computed to be outliers.
- The median wickets taken in each match are 2-3.
- There is only one team (Pune Warriors) which had matches with 0 wickets taken by them.

DATASET DESCRIPTION





- Mumbai Indians and Chennai Super Kings scored the highest number of runs per match on average.
- Rising Pune Super Giants and Kochi Tuskers Kerala were the least-scoring teams per match on average.

- The maximum number of runs that can be achieved on one ball is 7 assuming a wide ball and 6 runs.

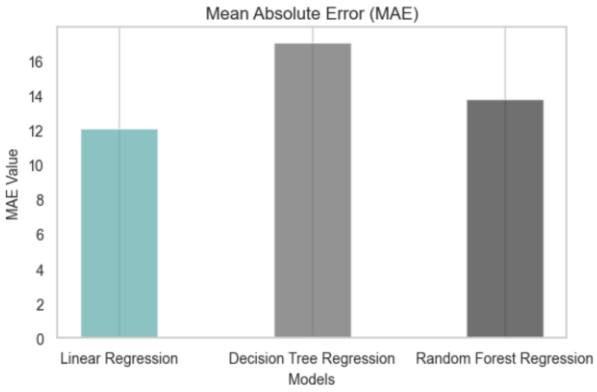
METHODOLOGY

We have trained, tested and evaluated two models. The main objective of our first model is to predict the detailed insights of a IPL Matches. Firstly, we will split the encoded dataset according to date, the datapoints before the year 2016 will be used in training the model and the datapoints after 2016 will be used in testing. We are dividing it along time to maintain a timeline continuity in the data. We experimented with **Regression techniques such as Linear Regression, Decision Tree Regression, Random Forest Regression and AdaBoosting** to increase and evaluate the performance of our model. Adaboost is a powerful technique that converts the weak learners(multiple models) to strong learners(final combined model). At the end, we have performed **K-Cross Validation** by dividing the test and train dataset according to different years.

In the 2nd model we compute the runs scored on each ball. **We then removed rows with NULL values** and selects a subset of columns for further analysis. **Categorical data is handled by one-hot encoding**. The dataset is **scaled using StandardScaler**, and used to train three different classifiers: **Logistic Regression, Random Forest, and Multilayer Perceptron (MLP)**

RESULT & ANALYSIS

Model 1:

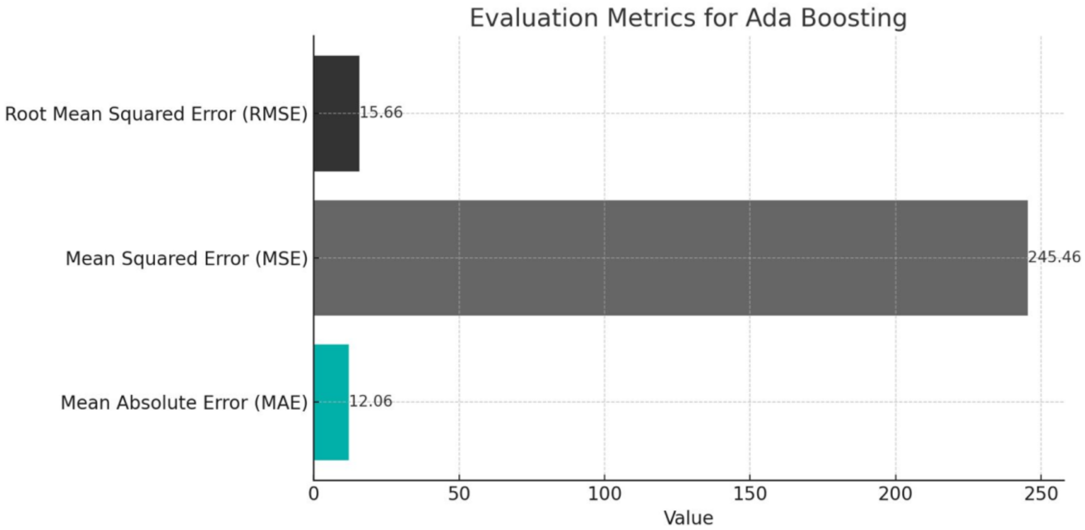


Mean Absolute Error (MAE)
Linear - 12.11
Decision Tree – 17.08
Random Forest – 13.76

Mean Squared Error (MSE)
Linear – 251.01
Decision Tree – 531.05
Random Forest – 330.21

Root Mean Squared Error (RMSE)
Linear – 15.84
Decision Tree – 23.04
Random Forest – 18.17

RESULT & ANALYSIS



After testing all the different models, the Ada Boosting Model gives the best accuracy. As loss is the least for the Ada Boosting Model among all the different models we tested the data for.

RESULT & ANALYSIS

Model 2:

Logistic Regression:
Accuracy: 0.477

	Precision	Recall	f1-Score	Support
Accuracy			0.48	15203
Macro Avg	0.16	0.15	0.13	15203
Weighted Avg	0.4	0.48	0.41	15203

Multi-Level Perceptron:
Accuracy: 0.476

	Precision	Recall	f1-Score	Support
Accuracy			0.48	15203
Macro Avg	0.17	0.16	0.14	15203
Weighted Avg	0.41	0.48	0.42	15203

Random Forrest Classifier:
Accuracy: 0.390

	Precision	Recall	f1-Score	Support
Accuracy			0.39	15203
Macro Avg	0.14	0.14	0.14	15203
Weighted Avg	0.35	0.39	0.37	15203

Considering the provided metrics, the Logistic Regression and MLP models have similar performances, both outperforming the Random Forest classifier in terms of accuracy. The MLP has a slightly better macro average precision and f1-score compared to Logistic Regression, which might make it the preferable model among the three, especially if the balance between classes is a concern. Despite the low absolute accuracy figures, the context of having eight possible outcomes for each ball (12.5% chance if guessing randomly) puts these results in a more positive light. Achieving accuracy figures close to 48% indicates that the models are learning and making predictions that are significantly better than random guessing. In predictive tasks with a high number of classes, such as this one, even small improvements over random chance can be considered a good performance. The models are providing valuable insights and may be further optimized with more data or feature engineering.

TIMELINE

We have successfully adhered to the outlined project timeline. We curated the necessary datasets, developed, and tested two models for predicting key metrics in a cricket match, achieving satisfactory results. Looking ahead, our plans for future enhancements include:

Looking ahead, our plans for future enhancements include:

- **Deep Learning:** Explore deep learning architectures, such as recurrent neural networks (RNNs) or Long Short-Term Memory networks (LSTMs), to better model the time-series nature of the data.
- **User Interface:** Create a user-friendly interface for the model's deployment that could be used by coaches, analysts, or fans to make real-time decisions or analyses.
- **Hyperparameter Optimization:** Conduct a more extensive hyperparameter search using techniques like grid search or randomized search to fine-tune the models.
- **Feature Engineering:** Investigate more sophisticated feature engineering techniques. For instance, include features that capture the sequential nature of balls within an over or the match situation like current run rate, required run rate, wickets in hand, etc.

INDIVIDUAL EFFORTS

- **Manan Chugh:** Literature review, Data Extraction and Collection, Extraction, EDA, feature selection and visualization of dataset, Analysis and inference of the data and results.

- **Abhijeet Anand:** Literature review, Data Extraction and Collection, Extraction, EDA, feature selection and visualization of dataset, Analysis and inference of the data and results.

- **Kartikey Dhaka:** Literature review, Data Extraction and Collection, Extraction, EDA, feature selection and visualization of dataset, Analysis and inference of the data and results.

- **Abhijay Singh:** Literature review, Data Extraction and Collection, Extraction, EDA, feature selection and visualization of dataset, Analysis and inference of the data and results.

Drive link containing all related material