Problem Statement-1 Report

Statistics is All You Need: IPL Data Analysis and 2025 Winner Prediction – The Game Behind the Game!

Completed on

Mar 24, 2025

Team

Supreme BEings

- **Team Members**:
  1. Abhirup Saha
  2. Chandrachur Pramanik
  3. Joydip Paul
  4. Jyotirmoy Dutta
  5. Swarnava Banerjee

1. Table of Contents :

2. Introduction

“matches.csv” and **“deliveries.csv”** were given. Performed extensive **data cleaning**, **feature engineering**.
Created **new datasets** from existing datasets so that **EDA** or insights can be easily achieved.

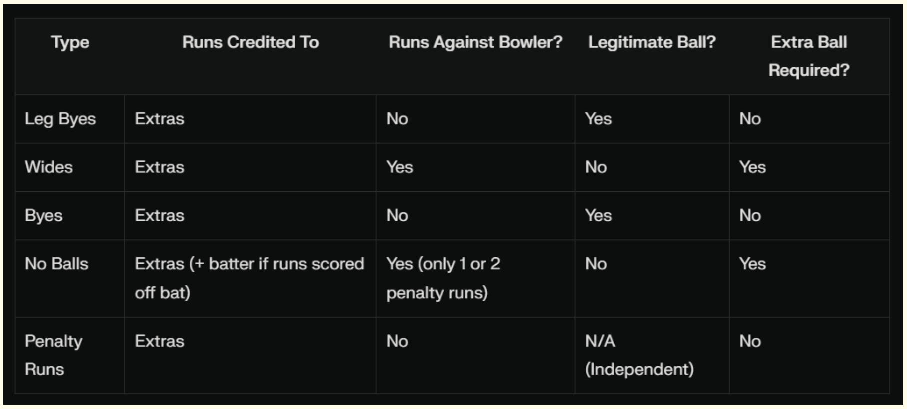
- player_stats.csv
- team_stat.csv
- team_season_stats.csv
- seasonal_stat.csv
- top_bowlers_stat.csv

With these datasets, gaining insight became helpful. Finally, created **Classical Machine Learning Ensemble Models** and **Artificial Neural Networks** to do future prediction of the winner of IPL matches.

3. Data Cleaning & Feature Engineering

- Data Preprocessing:
  - Filtered matches **unaffected by the D/L method** or **without results**.
  - Handled **zero-division errors** in calculating metrics like strike rate, bowling economy, and batting average.
- Standardized Team Names:
  - ’Delhi Daredevils’ → ’**Delhi Capitals**’
  - ’Deccan Chargers’ → ’**Sunrisers Hyderabad**’
  - ’Kings XI Punjab’ → ’**Punjab Kings**’
  - ’Rising Pune Supergiants’ → ’**Rising Pune Supergiant**’
  - ’Royal Challengers Bangalore’ → ’**Royal Challengers Bengaluru**’

- Season Identification:
  - Extracted season_id using the **first three digits** of match_id and assigned them to respective IPL seasons.
    - 335, 336 → 1
    - 392 → 2
    - 419 → 3
    - 501 → 4
    - 548 → 5
    - 597, 598 → 6
    - 729, 733, 734 → 7
    - 829 → 8
    - 980, 981 → 9
    - 108 → 10
    - 113 → 11,
    - 117, 118 → 12,
    - 121, 123 → 13
    - 125 → 14
    - 130, 131 → 15
    - 135: 16, 137 → 16
    - 142 → 17
- Wicket Adjustment:
  - Discarded **run-outs**, **retired-hurt**, **retired-out**, **obstructing-the-field** from bowler statistics.
  - Followed this guidelines



Type	Runs Credited To	Runs Against Bowler?	Legitimate Ball?	Extra Ball Required?
Leg Byes	Extras	No	Yes	No
Wides	Extras	Yes	No	Yes
Byes	Extras	No	Yes	No
No Balls	Extras (+ batter if runs scored off bat)	Yes (only 1 or 2 penalty runs)	No	Yes
Penalty Runs	Extras	No	N/A (Independent)	No

- Feature Addition:
  - Added **derived metrics** such as average powerplay score, death overs score, run rate, and economy rate.
  - **Renamed** some columns as per needed
- With this type of extended work, new datasets can easily be formed.

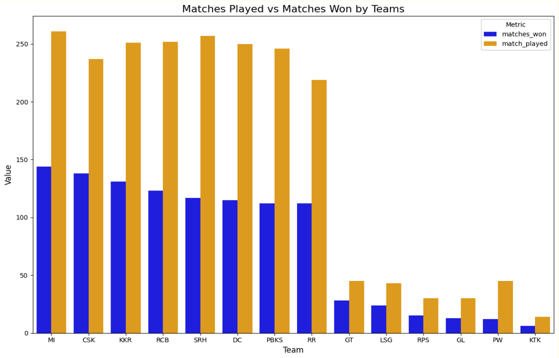
4. Structure of Newly Created CSV Files

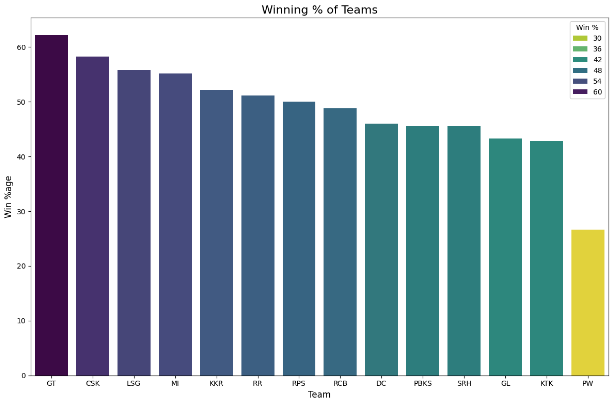
- **player_stats.csv**: Player-level metrics were stored (*e.g., player name, total runs, wickets taken, batting average, strike rate, bowling economy etc.*)
- **team_stat.csv**: Team-level metrics were stored (*e.g., match results, run rate, economy rate, powerplay and death overs statistics*)
- **seasonal_stat.csv**: Season-level metrics were stored (*e.g., total runs, innings, 200+ targets, Orange/Purple Cap winners*)
- **team_season_stats.csv**: Team performance across seasons (*e.g., total runs, average score*)
- **top_bowlers_stat.csv**: Top bowlers by season (*e.g., wickets, economy rate*)

5. Exploratory Data Analysis (EDA)

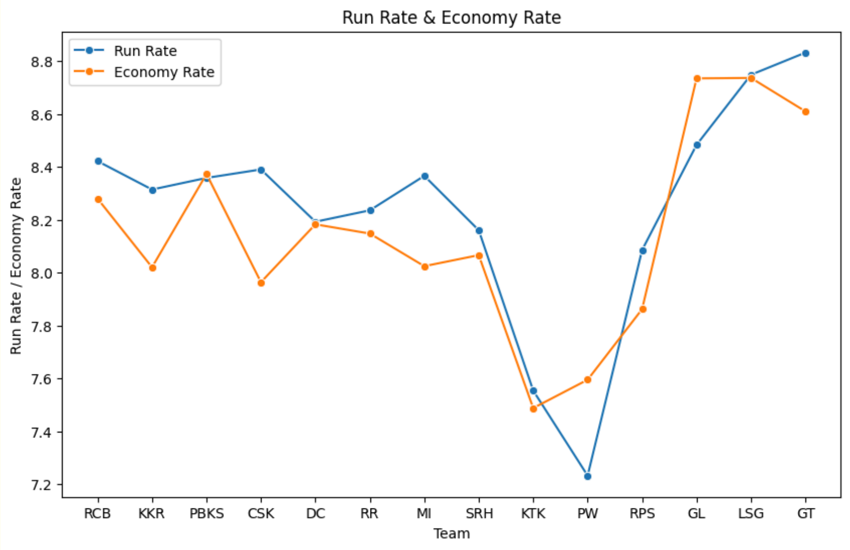
a. Team Performance Analysis:

- ***Matches played vs. win percentages***

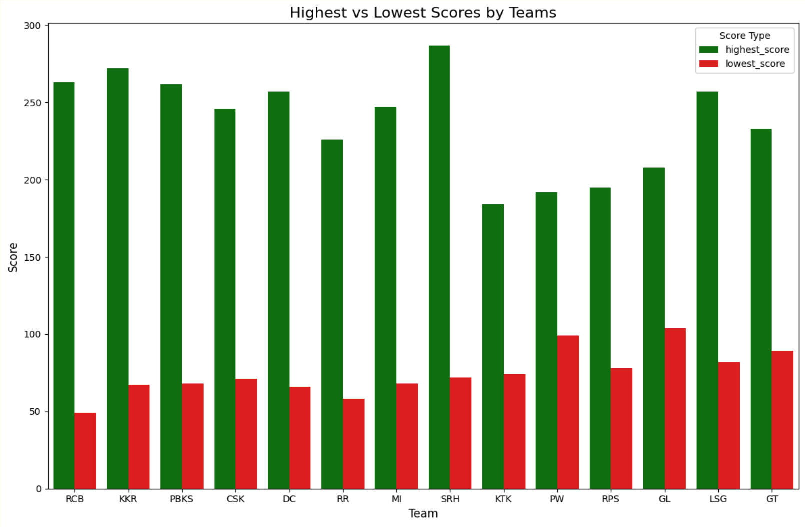




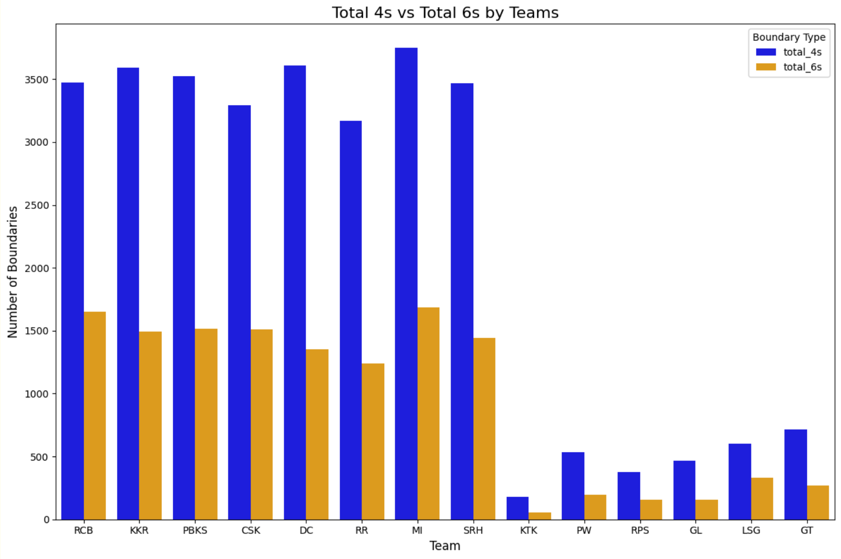
- ***Run rate and economy rate***



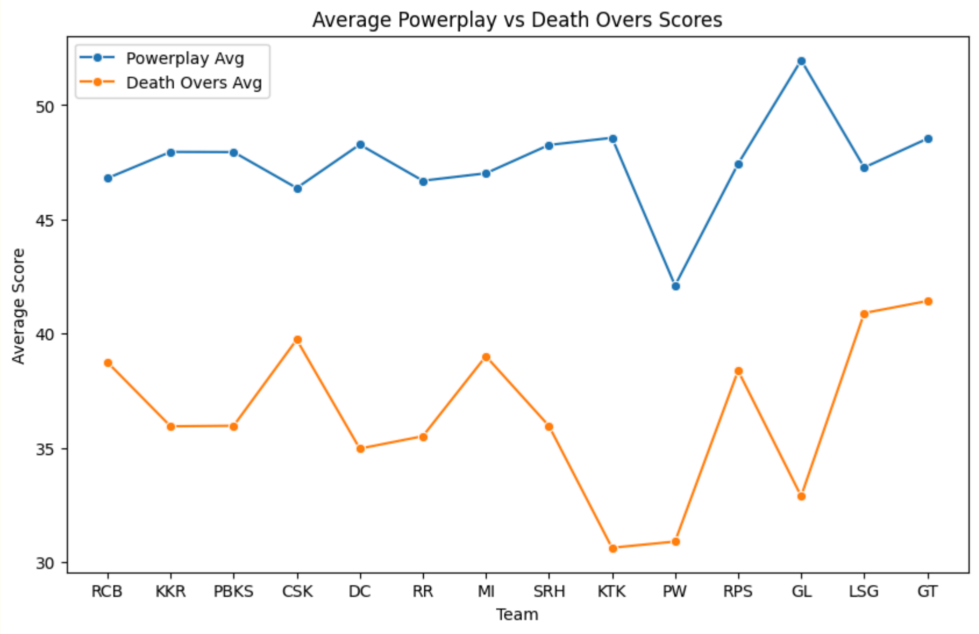
- ***Highest and lowest team scores***



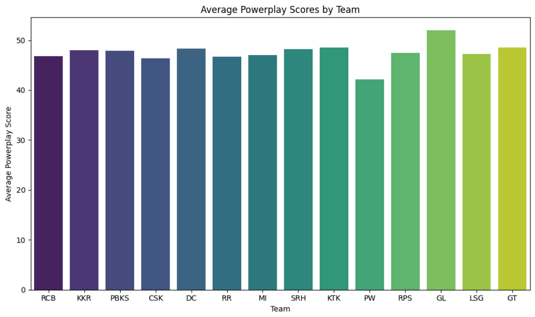
- ***Total 4s vs 6s***

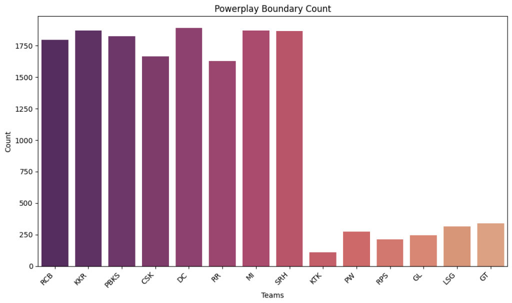


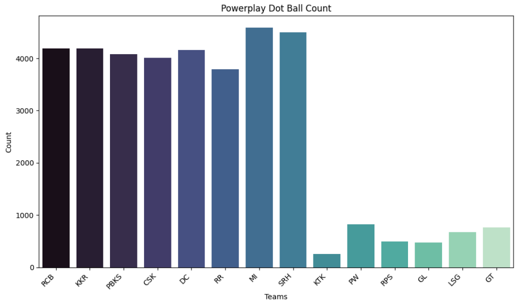
- ### *Powerplay and death over scores*

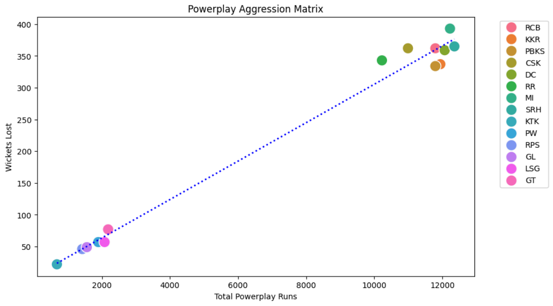


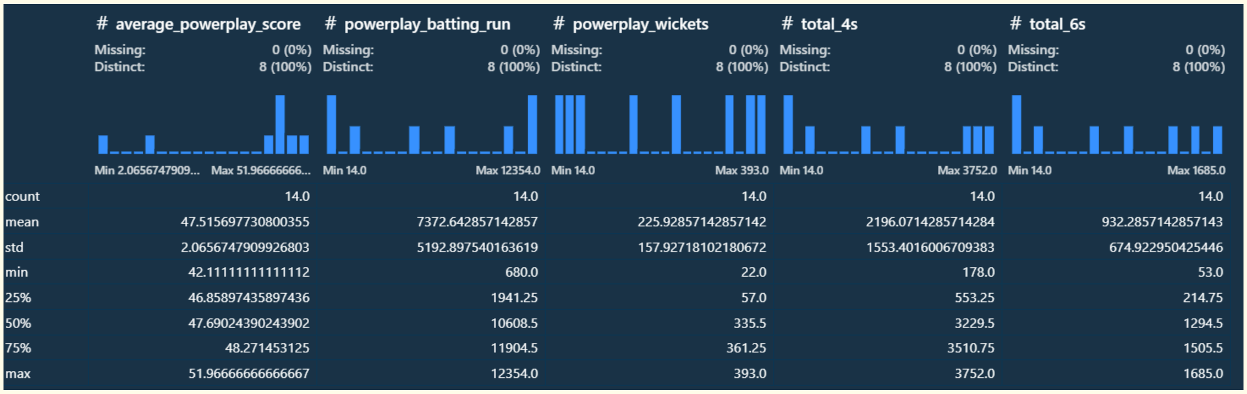
- ### *Powerplay Analysis*







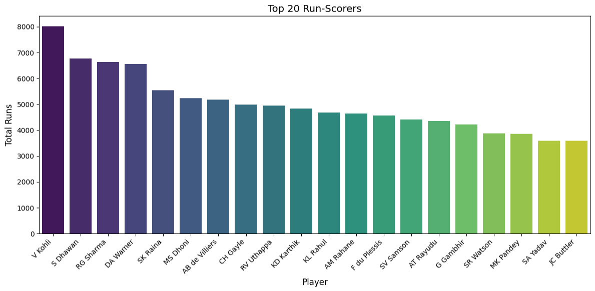


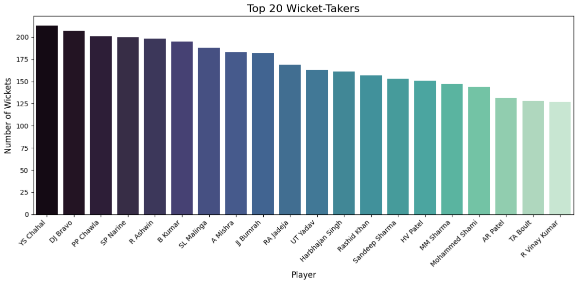


	# average_powerplay_score	# powerplay_batting_run	# powerplay_wickets	# total_4s	# total_6s
	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)	Missing: 0 (0%)
	Distinct: 8 (100%)	Distinct: 8 (100%)	Distinct: 8 (100%)	Distinct: 8 (100%)	Distinct: 8 (100%)
	Min 2.0656747909... Max 51.96666666...	Min 14.0 Max 12354.0	Min 14.0 Max 393.0	Min 14.0 Max 3752.0	Min 14.0 Max 1685.0
count	14.0	14.0	14.0	14.0	14.0
mean	47.515697730800355	7372.642857142857	225.92857142857142	2196.0714285714284	932.2857142857143
std	2.0656747909926803	5192.897540163619	157.92718102180672	1553.4016006709383	674.922950425446
min	42.11111111111112	680.0	22.0	178.0	53.0
25%	46.85897435897436	1941.25	57.0	553.25	214.75
50%	47.69024390243902	10608.5	335.5	3229.5	1294.5
75%	48.271453125	11904.5	361.25	3510.75	1505.5
max	51.96666666666667	12354.0	393.0	3752.0	1685.0

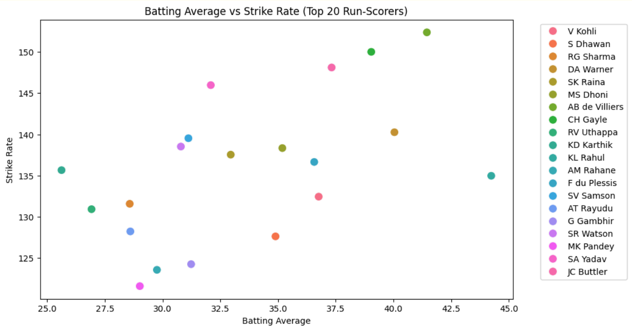
b. Player Performance Analysis:

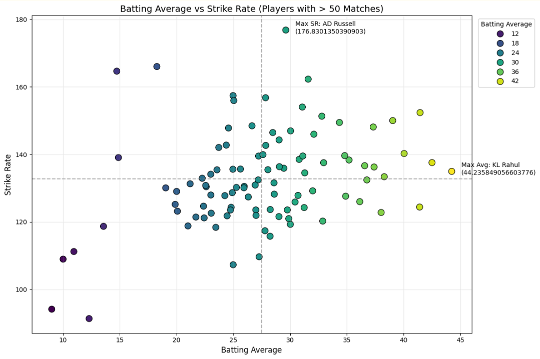
- ***Top 20 run-scorers and wicket-takers***


Top 20 Run-Scorers

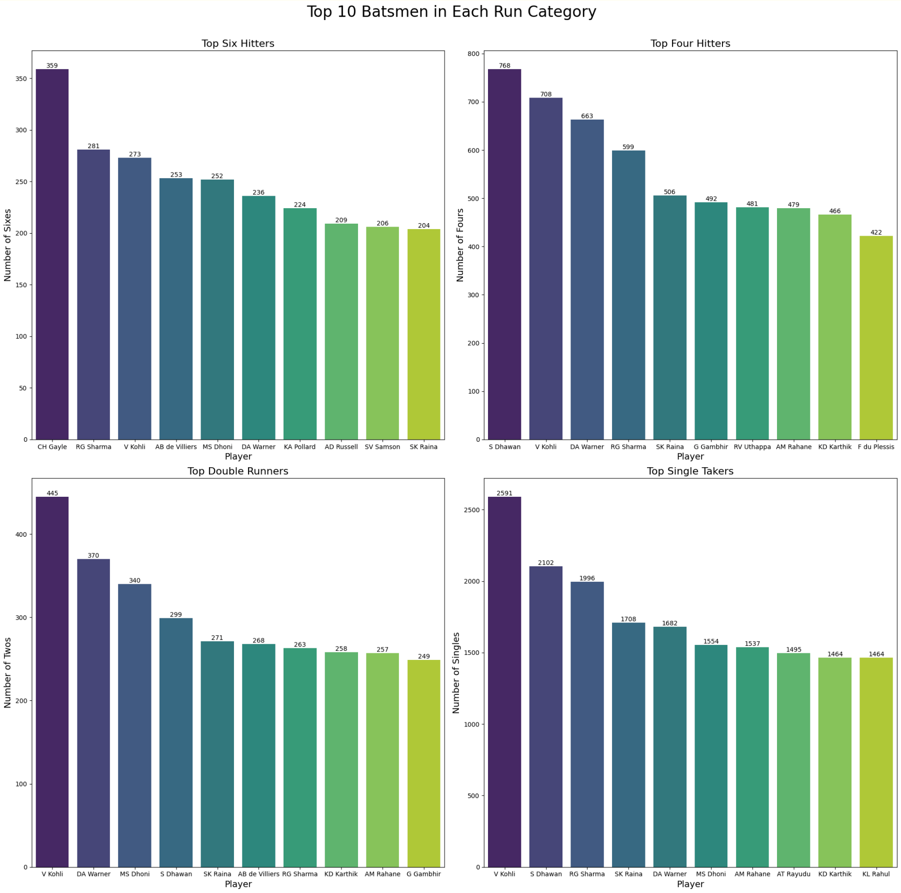

Top 20 Wicket-Takers

- ***Batting average vs. strike rate***


Batting Average vs Strike Rate (Top 20 Run-Scorers)


Batting Average vs Strike Rate (Players with > 50 Matches)

- ***Top batsmen by boundary type (6s, 4s, 2s, 1s)***


Top 10 Batsmen in Each Run Category

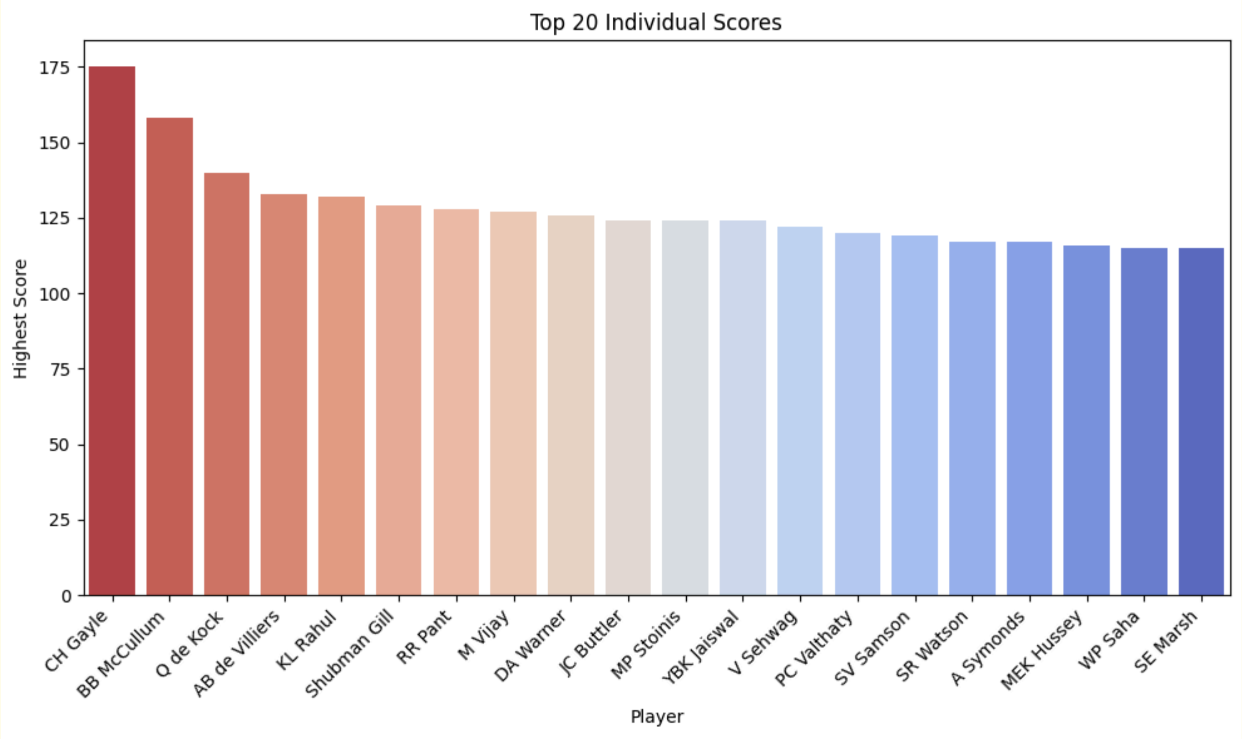
Top Six Hitters

Top Four Hitters

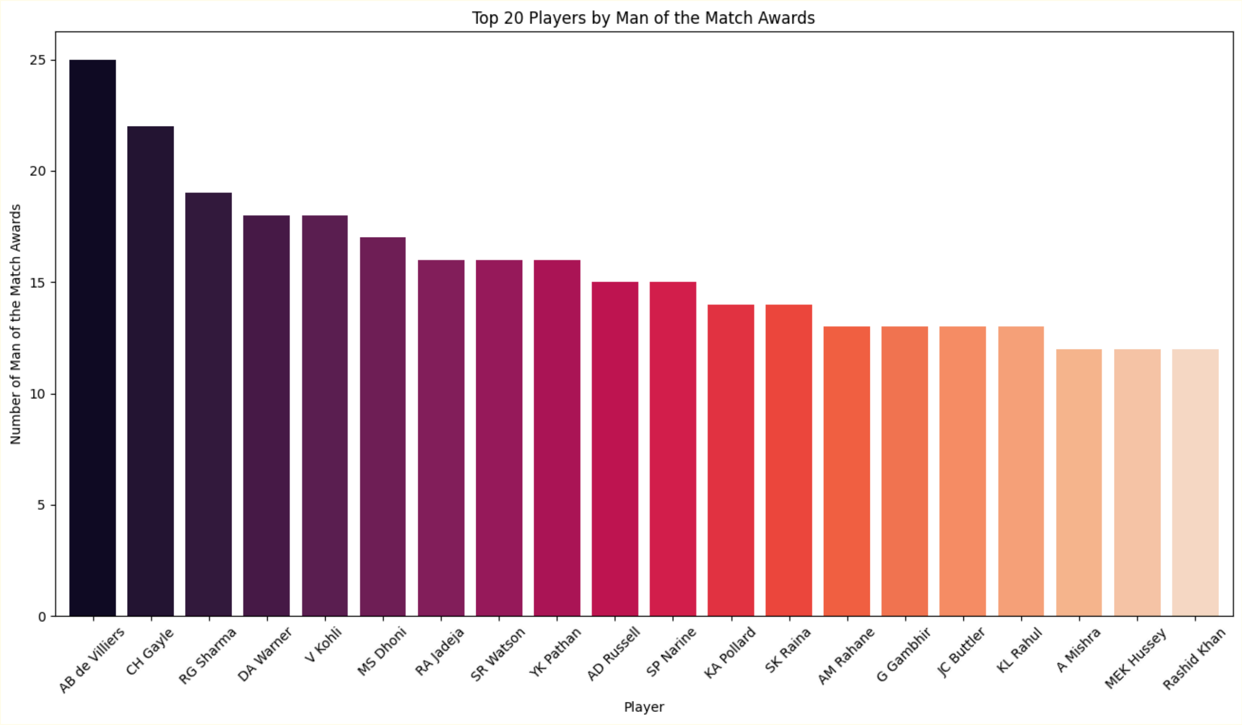
Top Double Runners

Top Single Takers

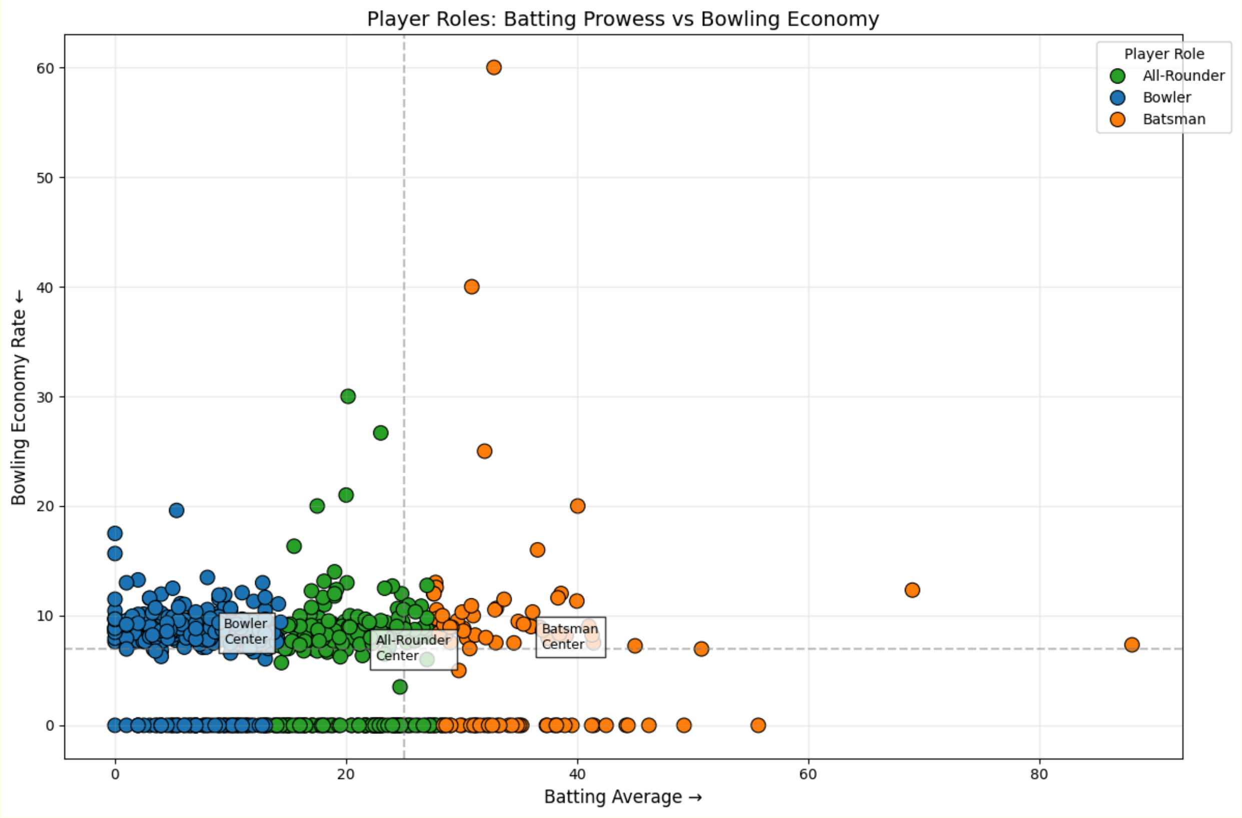
- ***Top highest individual scores***



- ***Man of the Match frequency***

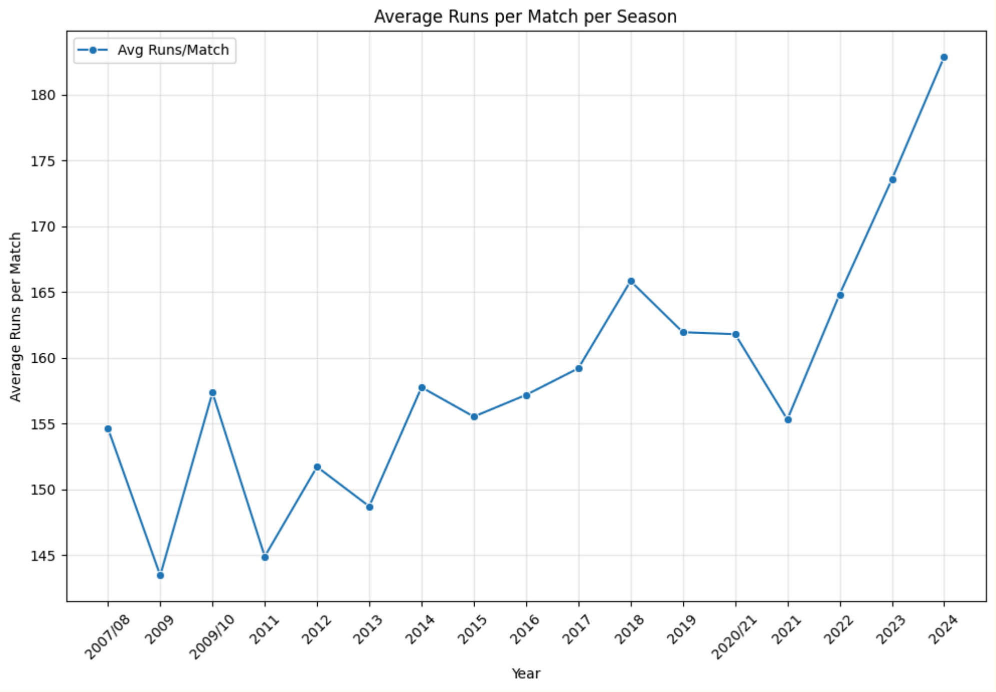


- *K-Means clustering for player classification (batsmen, bowlers, all-rounders)*

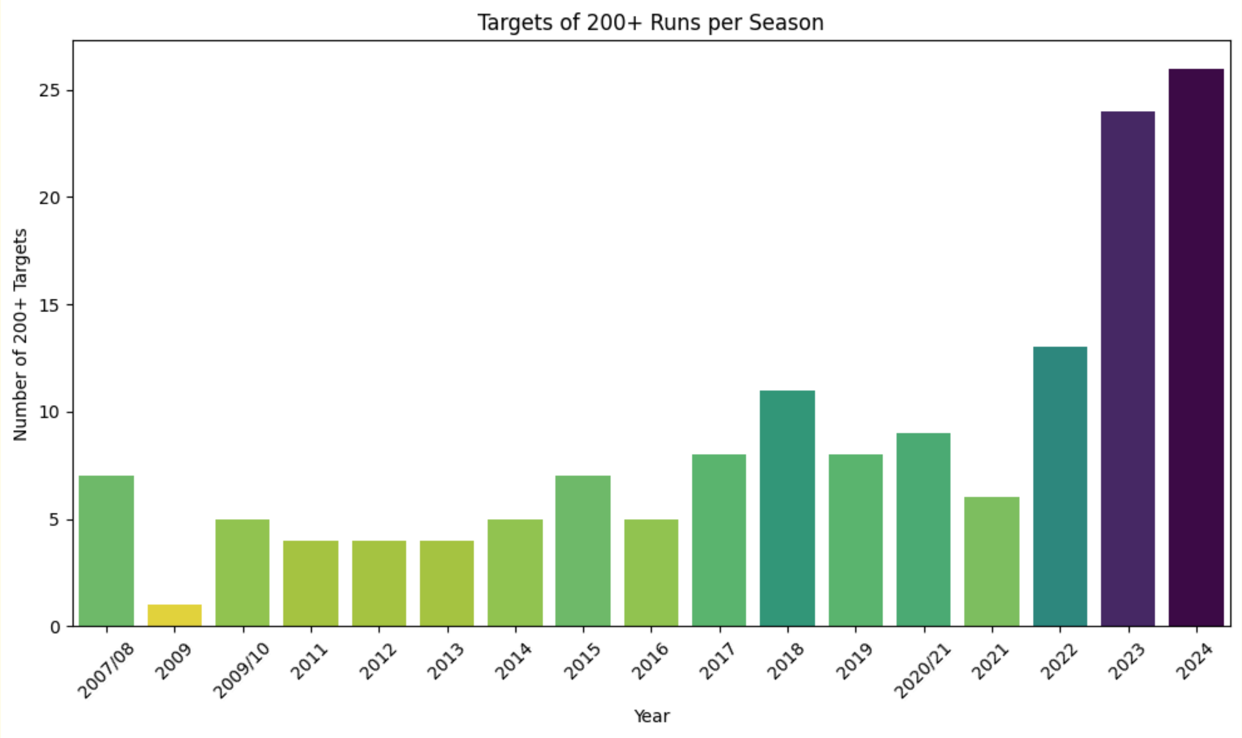


c. Seasonal Analysis:

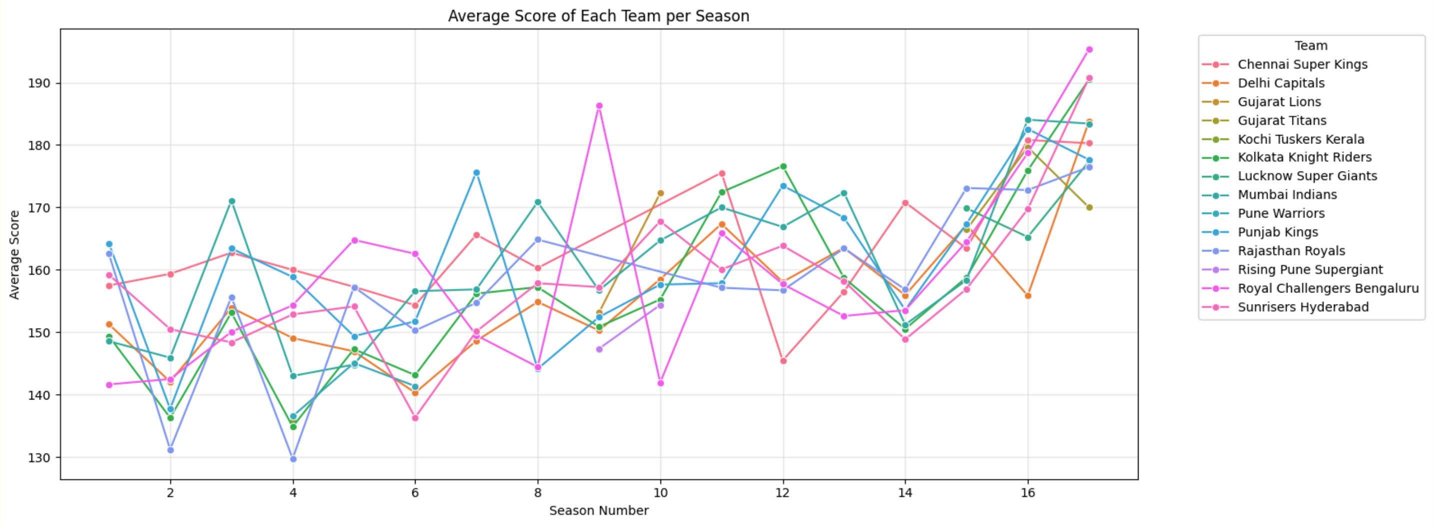
- *Average runs per match*



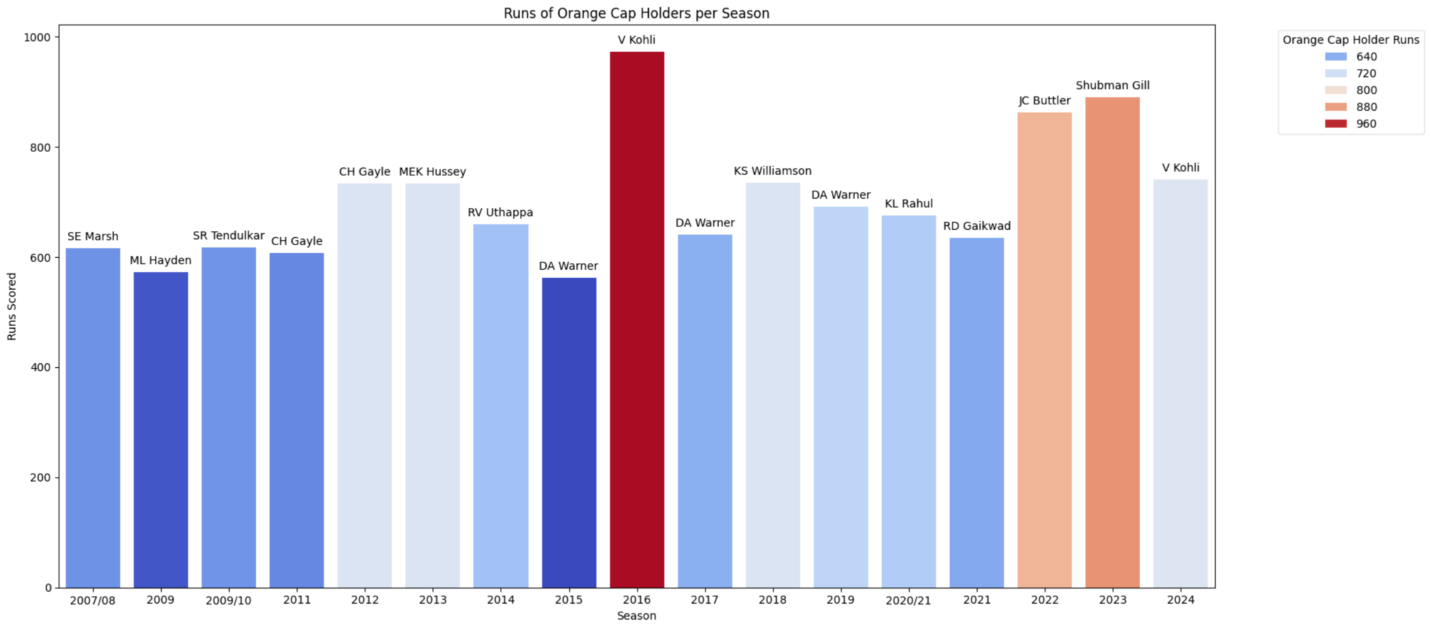
- ***Number of 200+ targets per season***

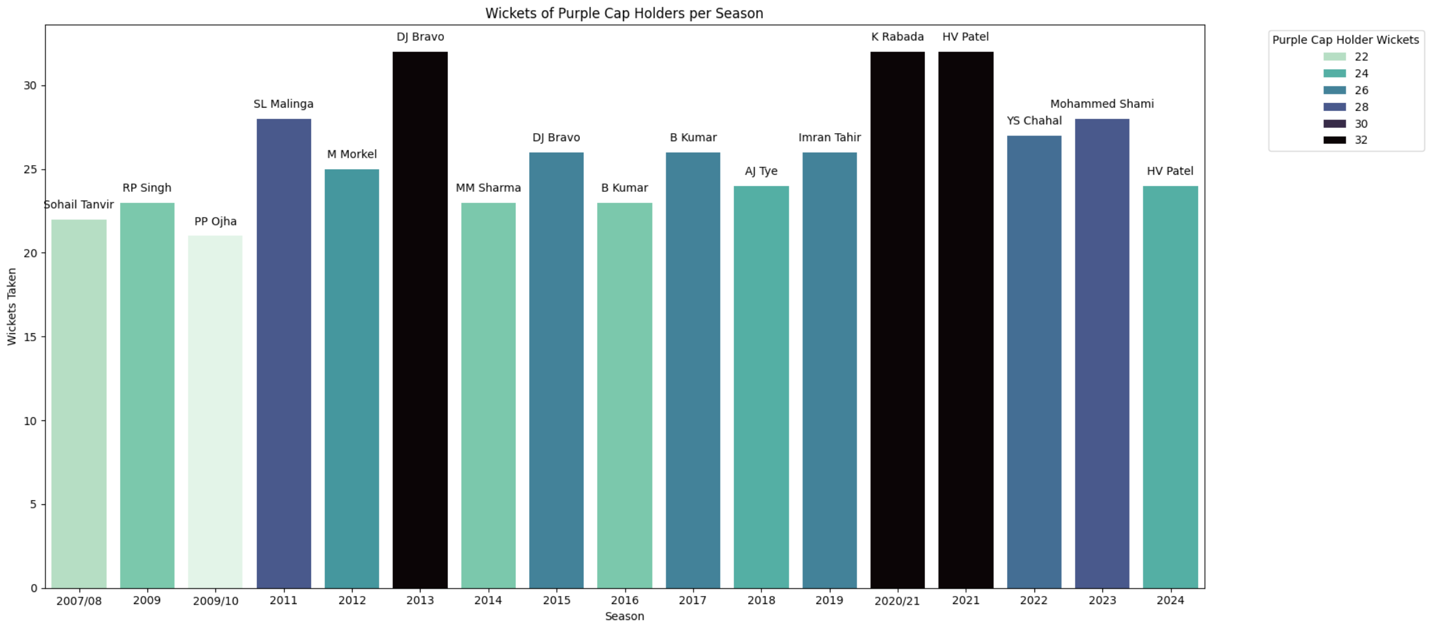


- ***Team average scores by season***

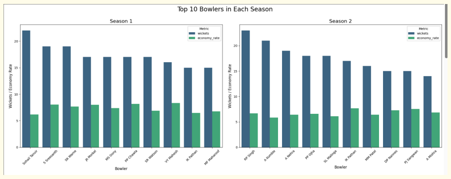


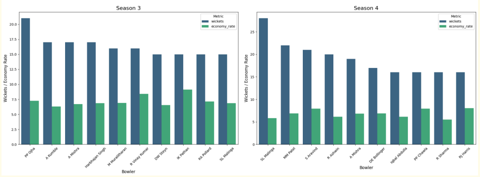
- ### *Orange/Purple Cap winners by season*

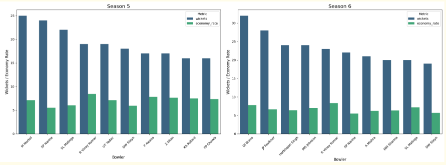


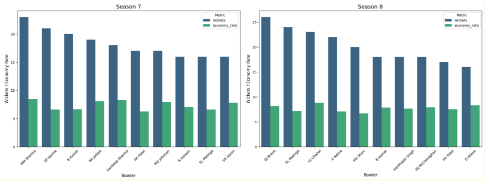


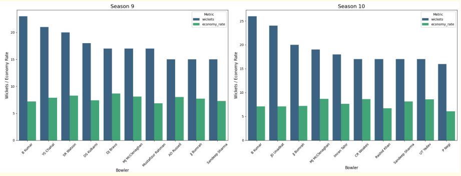
- ## *Top 10 bowlers per season*

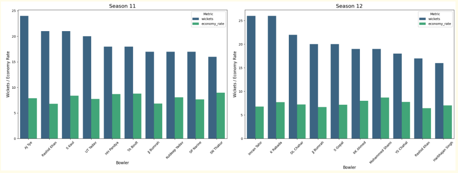


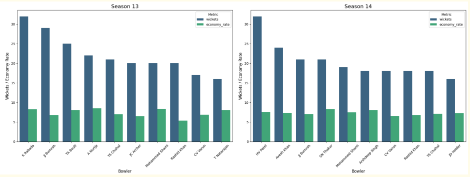


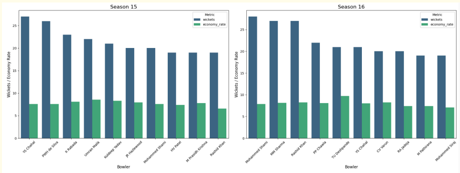


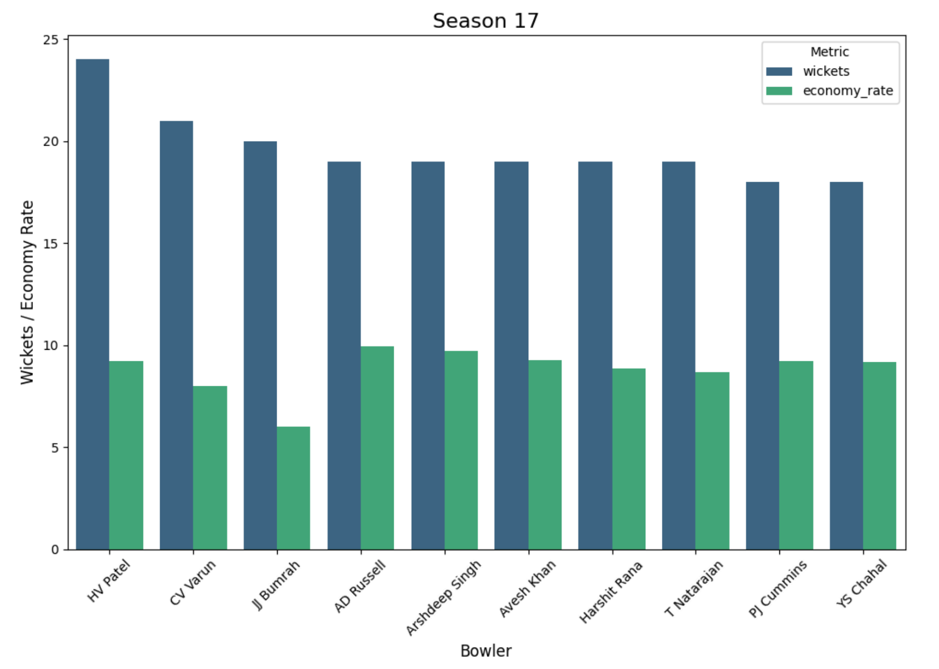








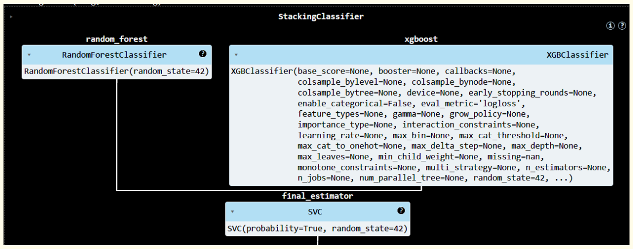




6. Model Building Methodology : IPL Winner Prediction Model

A. Classical Machine Learning Models

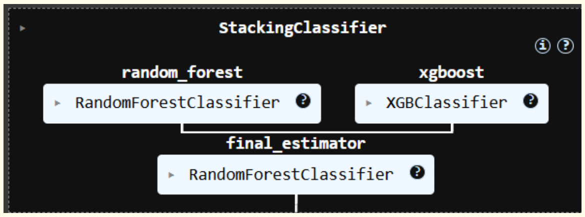
- **Data Preparation:**
  - Encoded object type features with **Label Encoder.**
  - **Missing Value** imputation with **frequent values.**
  - **Target** variable: **’winner**’.
  - **Stratified Train-Test split** done.
- **Models Used:**
  - 2 base models used :
    - **RandomForestClassifier**
    - **XGBClassifier**
  - Used **StackingClassifier** with 2 different algorithm as **final_estimator**
    - **SVC with RBF kernel**
    - **RandomForestClassifier**



```
RandomForestClassifier(random_state=42)
```

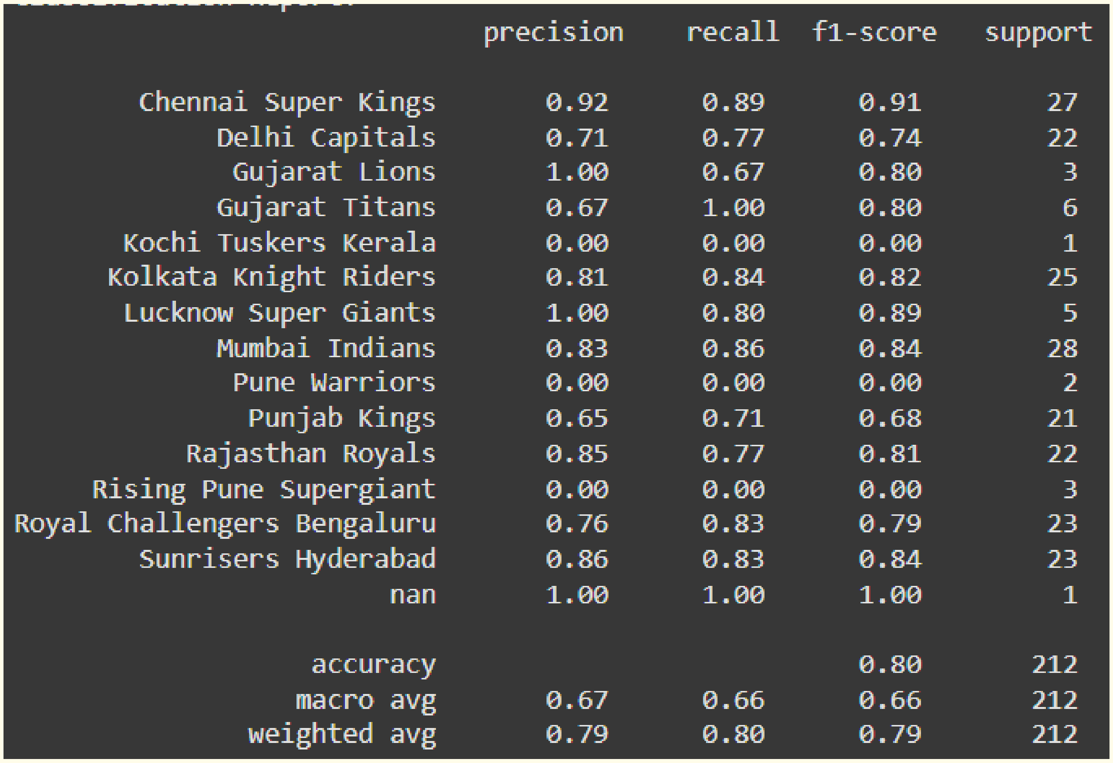
```
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric='logloss',
              feature_types=None, gamma=None, grow_policy=None,
              importance_type=None, interaction_constraints=None,
              learning_rate=None, max_bin=None, max_cat_threshold=None,
              max_cat_to_onehot=None, max_delta_step=None, max_depth=None,
              max_leaves=None, min_child_weight=None, missing=nan,
              monotone_constraints=None, multi_strategy=None, n_estimators=None,
              n_jobs=None, num_parallel_tree=None, random_state=42, ...)
```

```
SVC(probability=True, random_state=42)
```



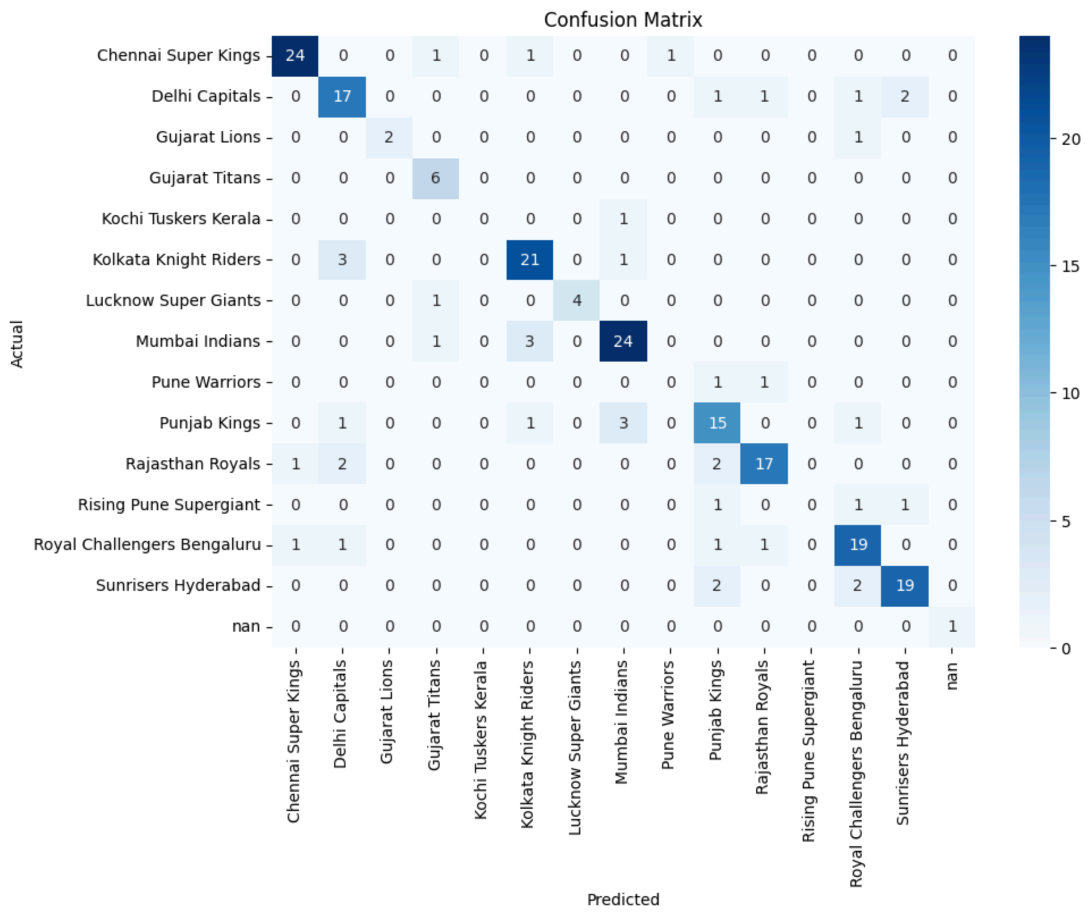
- **Model Performance:**
  - Stacking (SVM RBF): **75.94%** accuracy
  - Stacking (RandomForest): **77.01%** accuracy
  - Thus, chose StackingClassifier with RandomForest for further **Hyperparameter Tuning**
    - Used “**hyperopt**” & **stratified k-fold cross-validation**
    - Final better result : **79.71%** accuracy
- **Final Evaluation Metrics:**
  - **Accuracy:** 79.71%
  - **Precision:** 79.98%
  - **Recall:** 80%
  - **F1-Score:** 79.99%

- ### Classification Report:



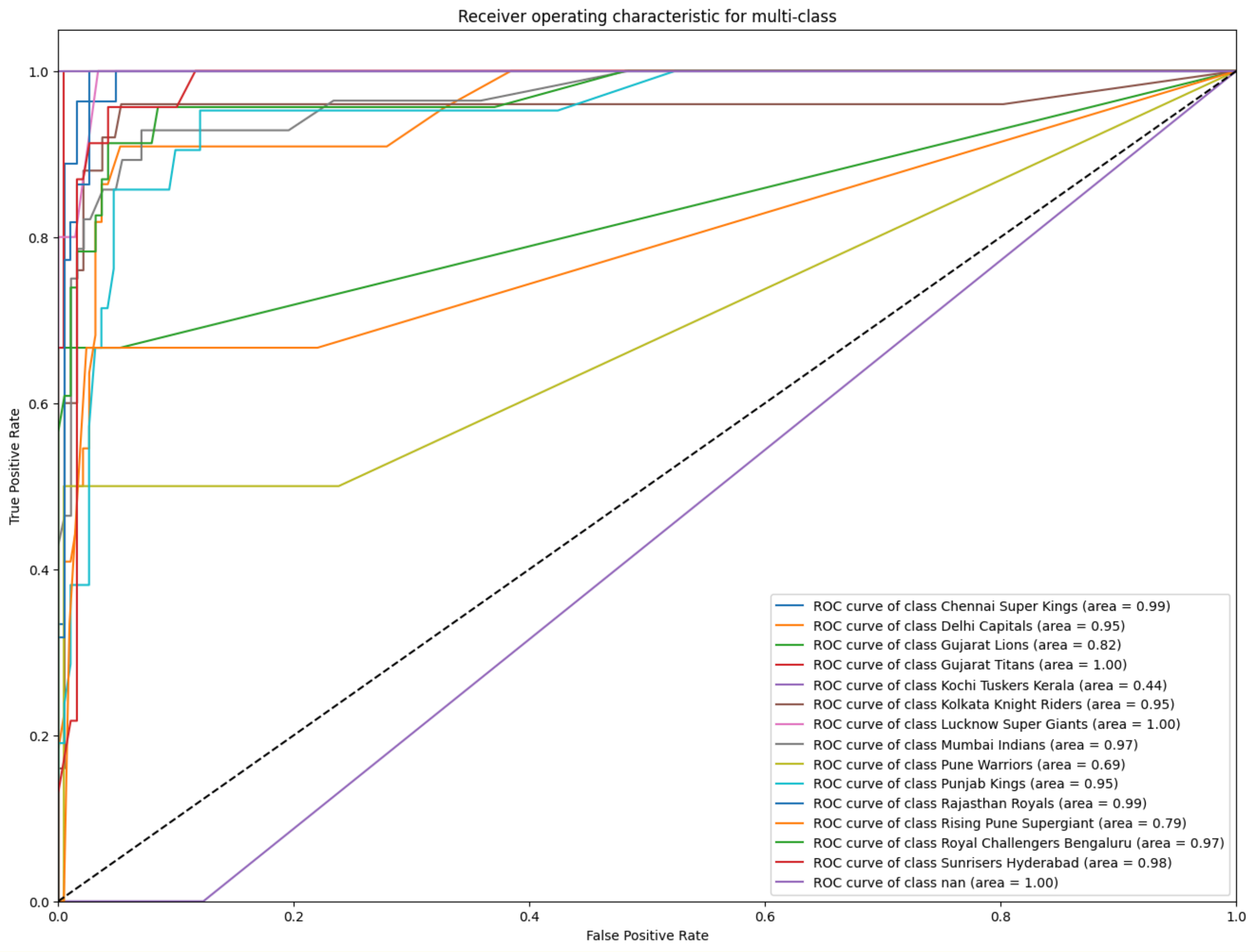
```
                              precision    recall  f1-score   support

           Chennai Super Kings       0.92      0.89      0.91        27
               Delhi Capitals       0.71      0.77      0.74        22
                Gujarat Lions       1.00      0.67      0.80         3
               Gujarat Titans       0.67      1.00      0.80         6
          Kochi Tuskers Kerala       0.00      0.00      0.00         1
         Kolkata Knight Riders       0.81      0.84      0.82        25
          Lucknow Super Giants       1.00      0.80      0.89         5
               Mumbai Indians       0.83      0.86      0.84        28
                Pune Warriors       0.00      0.00      0.00         2
                 Punjab Kings       0.65      0.71      0.68        21
             Rajasthan Royals       0.85      0.77      0.81        22
        Rising Pune Supergiant       0.00      0.00      0.00         3
   Royal Challengers Bengaluru       0.76      0.83      0.79        23
          Sunrisers Hyderabad       0.86      0.83      0.84        23
                          nan       1.00      1.00      1.00         1

                     accuracy                           0.80       212
                    macro avg       0.67      0.66      0.66       212
                 weighted avg       0.79      0.80      0.79       212
```

- ### Confusion Matrix:



Actual \ Predicted	Chennai Super Kings	Delhi Capitals	Gujarat Lions	Gujarat Titans	Kochi Tuskers Kerala	Kolkata Knight Riders	Lucknow Super Giants	Mumbai Indians	Pune Warriors	Punjab Kings	Rajasthan Royals	Rising Pune Supergiant	Royal Challengers Bengaluru	Sunrisers Hyderabad	nan
Chennai Super Kings	24	0	0	1	0	1	0	0	1	0	0	0	0	0	0
Delhi Capitals	0	17	0	0	0	0	0	0	0	1	1	0	1	2	0
Gujarat Lions	0	0	2	0	0	0	0	0	0	0	0	0	1	0	0
Gujarat Titans	0	0	0	6	0	0	0	0	0	0	0	0	0	0	0
Kochi Tuskers Kerala	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
Kolkata Knight Riders	0	3	0	0	0	21	0	1	0	0	0	0	0	0	0
Lucknow Super Giants	0	0	0	1	0	0	4	0	0	0	0	0	0	0	0
Mumbai Indians	0	0	0	1	0	3	0	24	0	0	0	0	0	0	0
Pune Warriors	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
Punjab Kings	0	1	0	0	0	1	0	3	0	15	0	0	1	0	0
Rajasthan Royals	1	2	0	0	0	0	0	0	0	2	17	0	0	0	0
Rising Pune Supergiant	0	0	0	0	0	0	0	0	0	1	0	0	1	1	0
Royal Challengers Bengaluru	1	1	0	0	0	0	0	0	0	1	1	0	19	0	0
Sunrisers Hyderabad	0	0	0	0	0	0	0	0	0	2	0	0	2	19	0
nan	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- **ROC Curve for Multi-Class Model:**



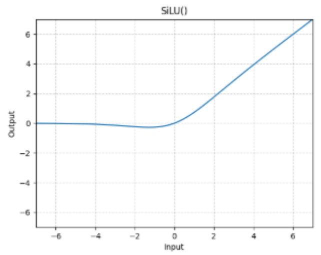
B. Artificial Neural Networks (ANN)

- **Data Preparation:**
  - Used some of the derived datasets alongside.
  - Extracted **Relevant Features** only [e.g., total runs, wickets, toss winner etc.]
  - Used **Label / One-Hot-Encoding**.
  - Data Splitted in **70 : 15 : 15 ratio.**
  - Used **PyTorch**.
  - Converted **Input Features** to **float32 tensors**.
  - Converted **Target Labels** to **long type tensors** using **np.argmax()**.

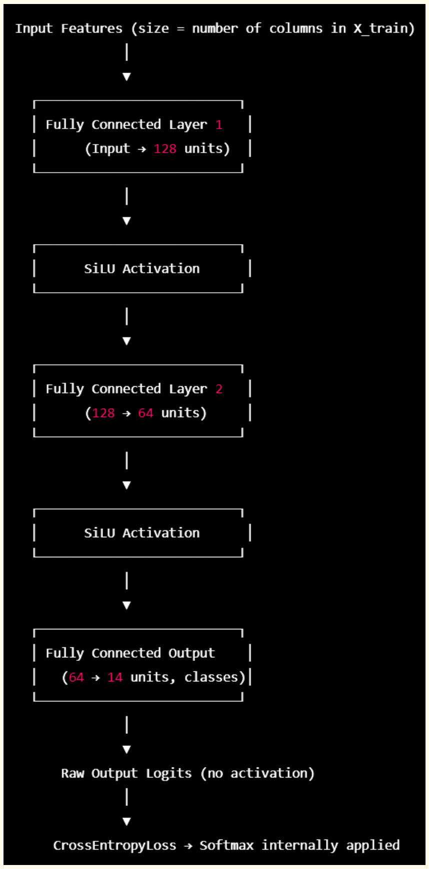
- ## Model Architecture:
  - **Input Layer:**Fully connected, size = number of features.
  - **Hidden Layer 1:** 128 neurons + SiLU activation.
  - **Hidden Layer 2:** 64 neurons + SiLU activation.
  - **Output Layer:** 14 neurons

    (*for 14 classes, no activation for CrossEntropyLoss*).

$\mathrm{silu}(x) = x * \sigma(x)$, where $\sigma(x)$ is the logistic sigmoid.

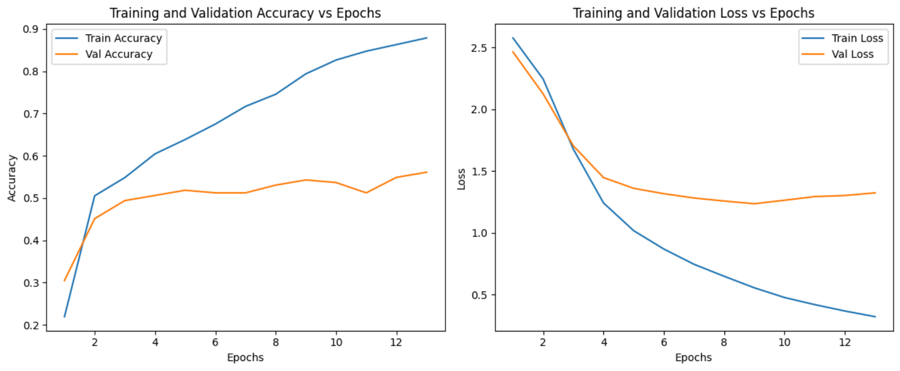


- ## Training Methodology:
  - **Loss Function**: CrossEntropyLoss
  - **Optimizer**: AdamW (***LR = 0.001***)
  - **Batch size**: 32
  - **Early stopping** with patience = **4 epochs**
  - **Total desired Epoch count**: 30

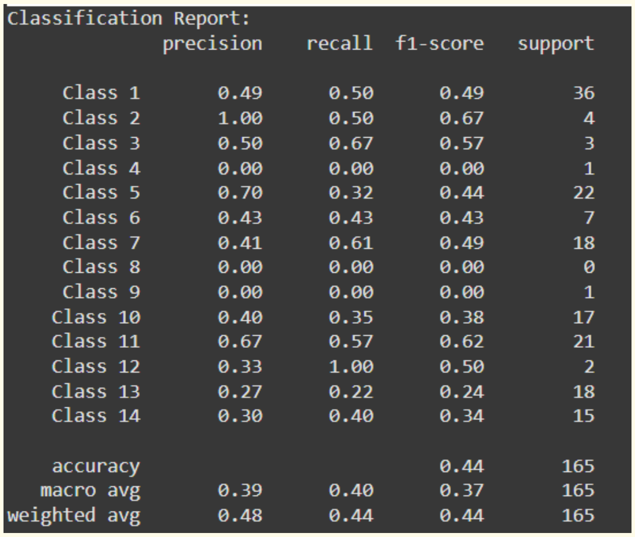


```
Input Features (size = number of columns in X_train)
                |
                ▼
    ┌──────────────────────────┐
    │ Fully Connected Layer 1  │
    │     (Input → 128 units)  │
    └──────────────────────────┘
                |
                ▼
    ┌──────────────────────────┐
    │      SiLU Activation     │
    └──────────────────────────┘
                |
                ▼
    ┌──────────────────────────┐
    │ Fully Connected Layer 2  │
    │     (128 → 64 units)     │
    └──────────────────────────┘
                |
                ▼
    ┌──────────────────────────┐
    │      SiLU Activation     │
    └──────────────────────────┘
                |
                ▼
    ┌──────────────────────────┐
    │ Fully Connected Output   │
    │  (64 → 14 units, classes)│
    └──────────────────────────┘
                |
                ▼
      Raw Output Logits (no activation)
                |
                ▼
     CrossEntropyLoss → Softmax internally applied
```

- ## Model Performance:
  - **Training Accuracy:** 87.86%
  - **Validation Accuracy:** 56.10%
  - **Training Loss:** 0.3222
  - **Validation Loss:** 1.3245
- ### Training vs Validation Plots:

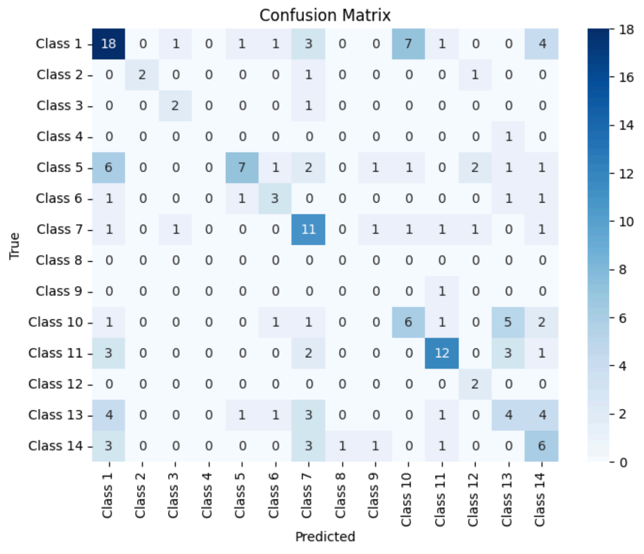


- ### Classification Report:



```
Classification Report:
              precision    recall  f1-score   support

     Class 1       0.49      0.50      0.49        36
     Class 2       1.00      0.50      0.67         4
     Class 3       0.50      0.67      0.57         3
     Class 4       0.00      0.00      0.00         1
     Class 5       0.70      0.32      0.44        22
     Class 6       0.43      0.43      0.43         7
     Class 7       0.41      0.61      0.49        18
     Class 8       0.00      0.00      0.00         0
     Class 9       0.00      0.00      0.00         1
    Class 10       0.40      0.35      0.38        17
    Class 11       0.67      0.57      0.62        21
    Class 12       0.33      1.00      0.50         2
    Class 13       0.27      0.22      0.24        18
    Class 14       0.30      0.40      0.34        15

    accuracy                           0.44       165
   macro avg       0.39      0.40      0.37       165
weighted avg       0.48      0.44      0.44       165
```

  - ### Confusion Matrix:



	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7	Class 8	Class 9	Class 10	Class 11	Class 12	Class 13	Class 14
Class 1	18	0	1	0	1	1	3	0	0	7	1	0	0	4
Class 2	0	2	0	0	0	0	1	0	0	0	0	1	0	0
Class 3	0	0	2	0	0	0	1	0	0	0	0	0	0	0
Class 4	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Class 5	6	0	0	0	7	1	2	0	1	1	0	2	1	1
Class 6	1	0	0	0	1	3	0	0	0	0	0	0	1	1
Class 7	1	0	1	0	0	0	11	0	1	1	1	1	0	1
Class 8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Class 9	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Class 10	1	0	0	0	0	1	1	0	0	6	1	0	5	2
Class 11	3	0	0	0	0	0	2	0	0	0	12	0	3	1
Class 12	0	0	0	0	0	0	0	0	0	0	0	2	0	0
Class 13	4	0	0	0	1	1	3	0	0	0	1	0	4	4
Class 14	3	0	0	0	0	0	3	1	1	0	1	0	0	6

7. Conclusion

“

1. **EDA**, **Feature Engineering**, and **Feature Extraction** :
    a. The extracted/new features were shortlisted in accordance with various parameters like,
        i. **Dropping** features which are not relevant
        ii. Making features to avoid **zero division error**
    b. The two original dataset [“matches.csv”, “deliveries.csv”] dataframes, were merged w.r.t. the match_id’s in order to **create new features**
    c. **5 new datasets** were created with new features from the available ones, providing a view of various new aspects of the tournament.
    d. These new datasets/features were,
        i. used to make **Insightful graphs** and **plots**.
        ii. further utilized in the **winner prediction model.**
2. IPL Winner Prediction Model :
    a. Classical ensemble model (RandomForest) outperformed ANN.
    b. ANN performance degraded with **insufficient data**, and overfitting due to more features.
    c. An accuracy of **79.71%** was achieved with **RandomForest** using optimized parameters.
    d. **ANN** model reached validation accuracy of **56.10%**, showing problems in working with smaller datasets.
    e. **SiLU** activation was used in the **ANN model** implementation.

”

8. References

I. Singh, P., Kaur, J. and Singh, L., 2024, March. Predicting IPL Victories: An Ensemble Modeling Approach Using Comprehensive Dataset Analysis. In 2024 2nd International Conference on Artificial Intelligence and Machine Learning Applications Theme: Healthcare and Internet of Things (AIMLA) (pp. 1-6). IEEE.

II. Kumar, Y., Sharma, H. and Pal, R., 2021, September. Popularity Measuring and Prediction Mining of IPL Team Using Machine Learning. In 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO) (pp. 1-5). IEEE.

Problem Statement-2 Report

Research Article Summarization Using Advanced NLP Techniques

Completed on **Mar 24, 2025**

Team:

Supreme BEings

- **Team Members**:
  1. Abhirup Saha
  2. Chandrachur Pramanik
  3. Joydip Paul
  4. Jyotirmoy Dutta
  5. Swarnava Banerjee

1. Table of Contents :

2. Introduction

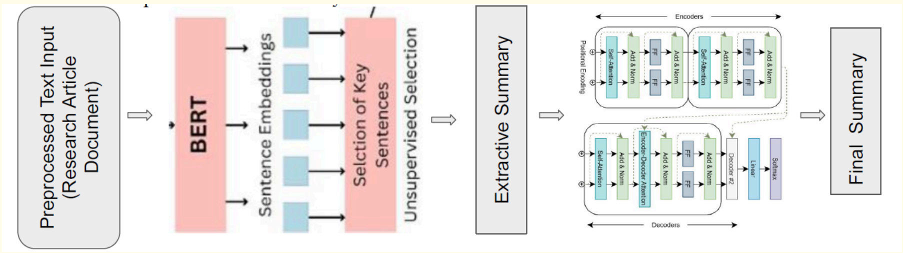
Dataset used : PubMed Dataset and **arXiv** Dataset for training
the hybrid Model and
CompScholar DataSet for evaluation.

3. Data Preprocessing

- The **Data Preprocessing Pipeline** provides support for normal text and document(s).
- First the text received is cleaned, the leading and trailing white spaces are trimmed and the new line characters are replaced with a space to maintain sentence boundaries.
- To handle large files the processed text is divided into overlapping chunks of size compatible for transformer input, the overlapping helps in maintaining the context between the chunks.
- For this reason **SciBert tokenizer** has been used to tokenize the text and then the input text is divided into chunks based on token count.

4. Model Description

- An hybrid model is build combining the strength of **Extractive** and **Abstractive** methods of summarization which can handle multiple research documents and along with the long documents maintaining efficiency.
- The model accepts the preprocessed document and generates an extractive summary by identifying the key sentences using **fine-tuned BERT.**
- The extractive summary is then fed to the **fine-tuned T5** model to generate the abstractive summary from it. It provides the **Final Summary.**



5. Conclusion

1. 66We have developed the working prototype or to be specific the basic idea & code for the model.
2. Due to having no access to a dedicated GPU server, we could not execute this file.
3. Hope, judging team will consider this scenario.
4. Our **innovative approach** was to deploy an hybrid model so that extractive & abstractive records could be handled.

6. References

I. Chaudhari, N., Vora, D., Kadam, P., Vaishali Khairnar, Patil, S. and Kotecha, K. (2024). Towards efficient knowledge extraction: Natural language processing-based summarization of research paper introductions. IAES International Journal of Artificial Intelligence, [online] 14(1), pp.680–680. doi:https://doi.org/10.11591/ijai.v14.i1.pp680-691.

II. Kolambkar, V., Shingade, B., Matha, Y., Kasar, S. and Palve, P., 2024. A Survey of Text Summarization Using NLP. A Survey of Text Summarization Using NLP (March 24, 2024).