# F20DL - Data Mining and Machine Learning Coursework (2024/2025)

**Group 18**

## Introduction

Sports analytics has emerged as a transformative field, leveraging data and machine learning techniques to enhance decision-making, performance analysis, and fan engagement. In this project, we focus on two key applications within the sports domain: ‘Predictive modelling for match outcomes in cricket’ and ‘Image classification of sports balls’. The **first task** aims to predict the outcomes of Indian Premier League (IPL) matches using match-level data (MLP). The **second task** involves utilizing convolutional neural networks (CNNs) to classify images of sports balls into distinct categories.

## Dataset Description and Analysis

This project utilizes the **Sports Ball Images** and **IPL Match Predictor** datasets. All values given here are after the preprocessing of these datasets.

1. The IPL dataset includes 980 matches, featuring 21 categorical attributes such as venue, teams, toss decisions, and winners. Missing data in columns like "city" and "method" was cleaned by removing rows lacking winner values.
2. The Sports Ball dataset comprises 2,698 images of 10 ball types (e.g., football, rugby), sourced from Kaggle. The preprocessing was to clean damaged images and augment data with rotations and zooms for better generalization.

## Experimental setup

The experiments were conducted using two platforms: **Google Colab (with GPU)** and **Jupyter Notebook**, ensuring compatibility and computational efficiency.

**D1**: Evaluated using **Decision Trees, Random Forest, k-NN, and MLP**. This focused on traditional machine learning models and foundational neural network architectures to establish baseline performance.

**D2**: Extended to include **CNNs** alongside Decision Trees, Random Forest, k-NN, and MLP. This setup leveraged the GPU in Google Colab to handle the computational demands of CNNs, allowing deeper exploration of spatial patterns and class distinctions in the image dataset.

**Detailed descriptions and examples of each requirement, along with the complete requirements analysis can be found in the notebooks and appendix for further reference.**

## R2- Data Analysis and Exploration

**D1:** Data preprocessing was performed to clean and standardize the dataset, ensuring consistency and relevance. Stadium names in the `"venue"` column were standardized by removing redundant details (e.g., "M Chinnaswamy Stadium, Bengaluru" to "M.Chinnaswamy Stadium"), and team names in columns such as `"team1"`, `"team2"`, `"toss_winner"`, and `"winner"` were updated to reflect rebranding (e.g., "Delhi Daredevils" to "Delhi Capitals"). Historical anomalies like "Deccan Chargers" were aligned with their modern equivalents, and rows containing defunct teams such as "Rising Pune Supergiants" and "Gujarat Lions" were removed to focus solely on active teams. These preprocessing steps prepared the dataset for accurate analysis and reliable insights.

EDA was conducted to uncover patterns in the dataset using various visualizations. **Histograms** with KDE overlays analyzed the distributions of numerical features, such as result margins and target runs. A **bar chart** highlighted the frequency of matches played in different cities, reflecting venue popularity. **Pie charts** revealed team dominance and the likelihood of toss winners also winning the match, showing the percentage of matches each team won and how often toss winners converted their advantage into victories. A **scatter plot** illustrated the relationship between target runs and result margins, showing how much a team won relative to the target, with distinct colours highlighting wins by wickets versus runs for better clarity. **Separate bar charts** explored the impact of toss decisions ("bat" or "field") on match outcomes, showing batting-first and bowling-first wins for each team and highlighting their performance trends based on toss choices. These visualizations provided a detailed understanding of venue trends, team strategies, and how toss decisions influenced match results. Visuals for EDA are provided in the appendix.

---

**D2:** The dataset underwent extensive preprocessing and exploratory data analysis (EDA) to optimize it for model training and address inherent challenges. Preprocessing steps included **resizing images, normalizing pixel values,** and **managing class imbalances** by limiting each class to 400 images, prioritizing those with aspect ratios close to 1.23.
EDA provided critical insights through aspect ratio analysis to **remove outliers**, class distribution visualization to address imbalances, and the creation of **mean** and **standard deviation images** to explore class-specific features and variability. Pairwise difference analysis highlighted overlapping attributes between **similar classes** while **bounding box evaluations** ensured uniform object placement. Additional steps included **feature importance analysis** to identify **key pixel regions**, **clustering** to uncover hidden patterns, and inter-class variability assessment to measure separability. These efforts collectively streamlined the dataset for effective model training and highlighted its **unique characteristics and complexities**. Visuals for EDA are provided in the appendix.

## R3- Clustering

**D1:** Clustering was performed on the dataset to uncover hidden patterns and segment the data into meaningful groups based on venue characteristics, team performance, and toss dynamics. The clusters were as follows:

**Venue-Based Clustering** included Moderately Utilized, Balanced Venues - High-Usage, Well-Balanced Venues - Low-Scoring, Rarely Used Venues - Low-Margin, Occasionally Used Venues - High-Scoring, Specialized Venues - High-Result-Margin Venues.

**Team Performance Clustering** grouped teams into Balanced Teams with Moderate Margins - Consistent Teams with Strategic Victories - Dominant Teams with High Margins.

**Toss-Based Clustering** identified Teams with Moderate Toss Performance - High Toss Efficiency with Fewer Matches Played - Consistent Teams with High Toss Success.

The venue-based clustering, with six clusters, was determined to be optimal based on the Silhouette Score, which measures the quality of clustering by evaluating how well-separated clusters are. This revealed insights into venue dynamics, aiding in selecting features like venue-specific statistics for model training. The team performance clustering identified three distinct groups based on average victory margins, highlighting the diversity in team strategies and outcomes. Finally, toss-based clustering offered insights into how teams handle toss outcomes, with some excelling in converting toss advantages into match wins, while others showed inconsistencies. These findings provide a deeper understanding of the dataset and aid in feature engineering, enabling context-aware modeling by incorporating venue conditions, team dynamics, and toss strategies into the predictive framework. Visuals for Clustering are provided in the appendix.

---

**D2:** Clustering experiments with **K-Means** and **Hierarchical Clustering** explored class separability, with **key parameters** and **configurations** tested for **optimization**. In K-Means, the number of clusters (k) was varied from 10 to 120 in increments of 10, with 10 initializations for each k to ensure stability. These experiments revealed **improved accuracy with increasing cluster** numbers, reaching 29% for k=120, where distinct classes like Tennis and Golf were grouped effectively. However, overlapping patterns in similar classes like Football and Rugby persisted, highlighting the limitations of feature similarity-based clustering.

For **Hierarchical Clustering,** the Ward linkage method was used to **minimize variance** within clusters, and experiments were conducted with cluster numbers ranging from 10 to 50. The method achieved 25.9% accuracy with 40 clusters, effectively **grouping some class-specific features** but struggling with the dataset's overlapping RGB features. These clustering experiments emphasized the trade-off between granularity and accuracy, providing insights into class distributions but reinforcing the necessity of spatially aware methods like CNNs for better performance. Visuals for Clustering are provided in the appendix.

## R4- Baseline Training and Evaluation Experiments

**D1:** This section evaluates the performance of three classification models—K-Nearest Neighbors (KNN), Decision Tree, and Random Forest—on the IPL dataset. By comparing their accuracy, precision, recall, and F1 scores, we identify key strengths and limitations of each approach, providing insights into their suitability for the task.

| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| K-Nearest Neighbors | 52.5% | 46% | 46% | 46% |
| Decision Tree | 69.4% | 74% | 68% | 69% |
| Random Forest | 73.5% | 77% | 72% | 73% |

Table 1: Train-Test Split (80:20)

The evaluation of three classification models—**K-Nearest Neighbors (KNN), Decision Tree, and Random Forest**—reveals clear distinctions in their performance. KNN demonstrates the weakest results, with an **accuracy of 52.5%** and an **F1 score of 46%**. This underperformance likely stems from its sensitivity to feature scaling and the possible presence of noise in the data, which hampers its ability to generalize effectively. In contrast, the Decision Tree performs notably better, achieving **a 69.4% accuracy** and an **F1 score of 69%**. This improvement highlights the model's capacity to capture non-linear relationships within the data, although it remains prone to overfitting, as suggested by the moderate difference between **precision (74%) and recall (68%).** The Random Forest model stands out as the best performer, with an **accuracy of 73.5%** and **an F1 score of 73%**. Moreover, model performance varies with train-test splits, as larger training sets (90:10) enable better learning, evident in improved metrics for **Decision Tree (70.4% accuracy)** and **Random Forest (73.5% F1 score).** However, smaller test sets may limit robust evaluation compared to 70:30, which better reflects generalization, with **Random Forest achieving 75.5% accuracy**. KNN remains consistently poor across splits, highlighting its limited ability to generalize effectively. This can be seen in Table 3 and Table 4 in the appendix.

---

**D2:** Decision Trees achieved **high training accuracy (~90%)** but struggled with generalization, as test accuracy dropped significantly (e.g., ~17% with an 80-20 split) due to high-dimensional and overlapping features. **Varying train-test splits (e.g., 70-30, 60-40)** and constraints yielded slight improvements but failed to address fundamental limitations. **k-NN performed well** for simpler classes like Tennis and Golf but faced scalability issues in high-dimensional data due to its reliance on distance metrics, which were inadequate for capturing pixel-level correlations in RGB datasets. **Random Fores**t achieved better generalization than Decision Trees, with test accuracy reaching 35%, leveraging ensemble learning to improve stability and reduce overfitting, though it lacked spatial feature extraction capabilities. **k-Means clustering experiments (varying k from 10 to 120**) provided moderate alignment with class labels but struggled with overlapping features. These results underscore the limitations of traditional models in handling **complex image datasets**, highlighting the need for **advanced methods** with spatial feature extraction capabilities. A visual representation of the Random forest is in the Appendix.

## R5- Neural Networks

**D1:** Neural networks, specifically Multi-Layer Perceptrons (MLPs), are widely recognized for their ability to model complex, non-linear relationships in data. In this analysis, an MLP classifier was implemented to evaluate its predictive performance on the dataset. The model was configured with two hidden layers containing 100 and 50 neurons, using the ReLU activation function and the Adam optimizer. Below are the results of the MLP classifier:

| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| Multi-Layer Perceptrons | 86.7% | 79% | 75% | 74% |
| Multi-Layer Perceptrons (Hyperparameter Tuning) | 95.4% | 95% | 96% | 95% |

Table 2: Model Performance (80:20 Train-Test Split) Before and After Hyperparameter Tuning

The MLP classifier achieved an impressive accuracy of 86.7%, with a precision of 79% and an F1 score of 74%, showcasing its strong ability to identify patterns across the dataset. Its performance in frequently occurring classes is particularly notable, as reflected in the clear diagonal dominance of the confusion matrix documented in the notebook. While there are slight misclassifications in neighboring classes, these provide opportunities to further enhance the model by addressing class imbalances in the training data. After hyperparameter tuning—with hidden layers set to (50, 30, 30), logistic activation, an alpha value of 0.0001, a learning rate of 0.005, and a maximum of 1000 iterations—the MLP achieved a significant boost in performance, with accuracy and F1 score reaching 95%. This highlights the importance of parameter optimization in maximizing predictive accuracy. However, the hyperparameter accuracy of 96% is exceptionally high and may indicate overfitting, suggesting the model is overly tailored to the training data. Overall, the MLP demonstrates highly competitive performance, positioning it as a reliable and robust choice for our predictive analysis.

---

**D2: CNN (Convolutional Neural Network):**
CNNs utilize convolutional layers to retain spatial information, making them better suited for image data. Two CNN models were tested, differing in architecture and design. The first model comprised two convolutional layers with **64 filters each**, **using a kernel size of (3x3)**, followed by **MaxPooling layers** and a **dense layer with 128 neurons**. This architecture focused on extracting key features while maintaining simplicity. The **second model** introduced an additional convolutional layer and increased the number of filters in the **second layer to 128**, enabling more comprehensive feature extraction and improved handling of complex patterns. Both

models utilized **ReLU activation** and **softmax for classification**, demonstrating the importance of deeper architectures in capturing intricate image details.

**Performance Overview:** Two CNN models were tested, showcasing significant improvements over MLPs in both training and validation metrics. The second CNN model achieved a **training accuracy of 98.58%**, a **validation accuracy of 83.28%**, and **a test accuracy of 85.86%**, with a **train loss of 0.0473** and a **validation loss of 0.6425**, indicating strong convergence and minimal overfitting. These metrics highlight CNN's robustness in learning complex patterns and generalizing across unseen data.

**Graph Analysis:** The training and validation accuracy graphs show a steady upward trend in accuracy for both datasets, with validation accuracy closely following the training curve, indicating effective generalization. Loss values consistently decrease during training, reflecting efficient learning and convergence. Graph attached in the Appendix
MLP models with **2, 3, and 4 hidden layers (300 or 500 neurons)** were tested on flattened image data. While increasing layers and neurons improved training accuracy, validation accuracy remained **stable between 33-35.5%**, and test accuracy hovered **around 38%**. This highlights the limitations of MLPs in capturing spatial patterns and color relationships critical for RGB datasets. The multi-class nature of the dataset further complicates classification due to subtle feature overlaps, underscoring why MLPs are less effective than convolutional architectures for image-based tasks. This limitation, combined with a reliance on feature selection and susceptibility to overfitting, restricts their performance on high-dimensional RGB datasets. These results highlight that MLPs are not well-suited for image classification tasks compared to models like CNNs, which can inherently leverage spatial hierarchies.

## Conclusion

D1: The IPL dataset analysis demonstrated the effectiveness of both traditional models and neural networks for match outcome prediction. Random Forest outperformed Decision Trees and k-NN, leveraging ensemble techniques to handle categorical features and complex relationships. The MLP achieved an impressive accuracy of 83.2%, with high precision and F1 scores, showcasing its ability to model non-linear patterns effectively. Clustering provided valuable insights into team performance, venue dynamics, and toss outcomes, enriching the feature engineering process. These findings highlight the importance of combining robust preprocessing, feature engineering, and advanced models to extract meaningful insights from structured sports data.

D2: Setup evaluated traditional models (Decision Trees, Random Forest, k-NN, MLP) alongside CNNs. While traditional models and clustering provided useful baselines and insights, they struggled with the complexities of RGB datasets, particularly spatial patterns and overlapping features. CNNs significantly outperformed them, leveraging spatial hierarchies and feature extraction for robust multi-class classification, reaffirming their necessity for high-dimensional image datasets.

## Appendix

| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| K-Nearest Neighbors | 52% | 56% | 47% | 48% |
| Decision Tree | 63.2% | 67% | 62% | 64% |
| Random Forest | 75.5% | 78% | 73% | 73% |

Table 3: Train-Test Split(70:30)

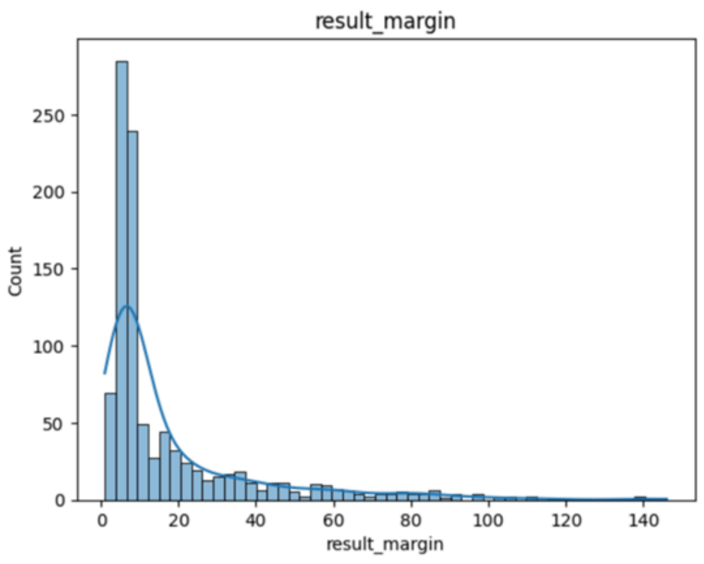
| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| K-Nearest Neighbors | 56.1% | 53% | 48% | 49% |
| Decision Tree | 70.4% | 71% | 69% | 69% |
| Random Forest | 73.5% | 76% | 74% | 75% |

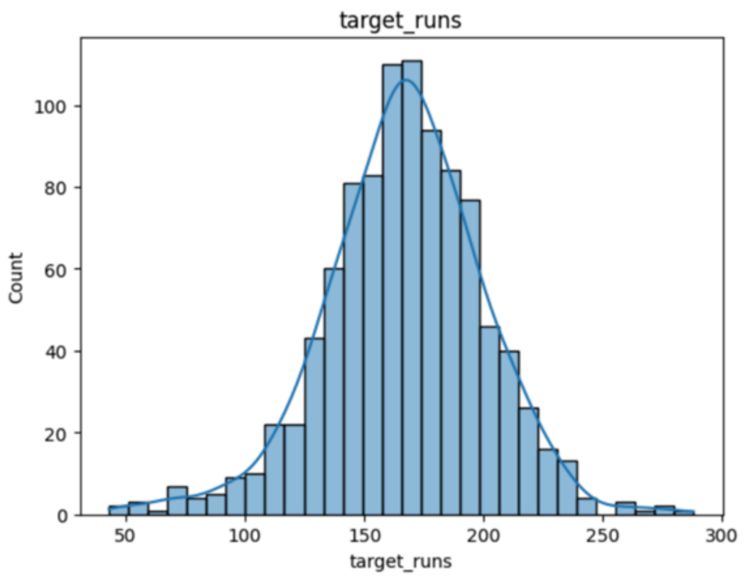
Table 4: Train-Test Split(90:10)

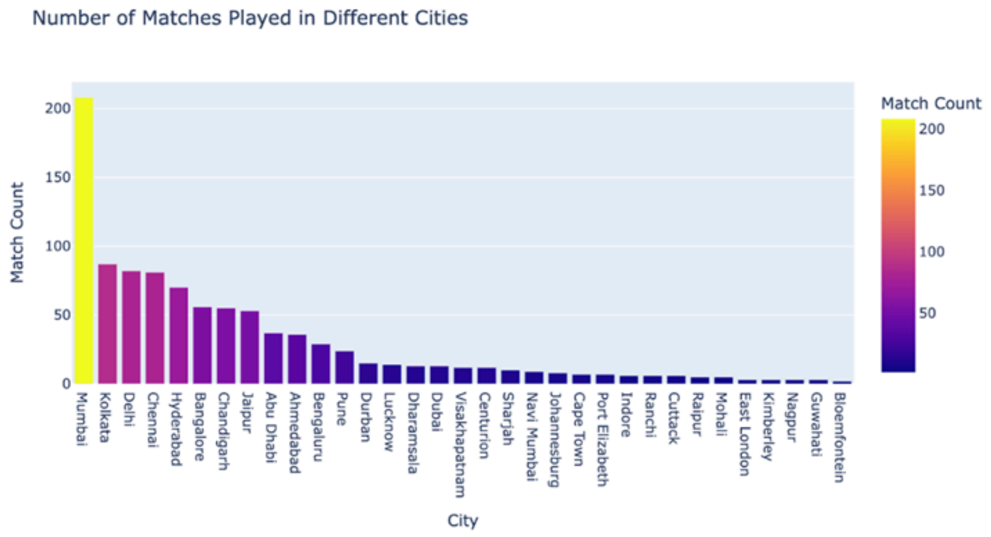
| Model | Accuracy | Precision | Recall | F1 Score |
|---|---|---|---|---|
| Multi-Layer Perceptrons(70:30) | 80.6% | 78% | 74% | 75% |
| Multi-Layer Perceptrons(90:10) | 85.7% | 87% | 80% | 81% |

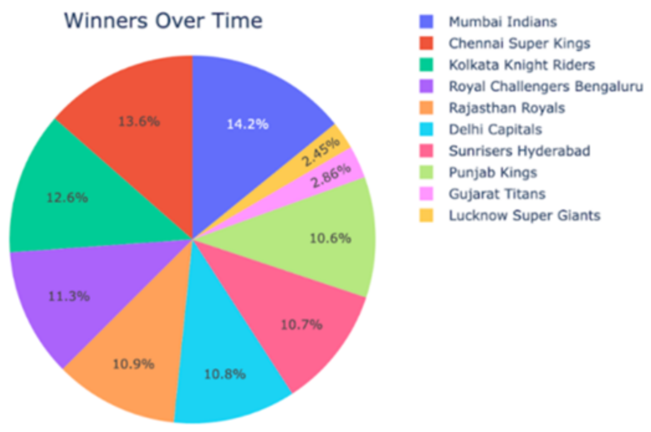
Table 5: Different Train-Test Splits for MLP

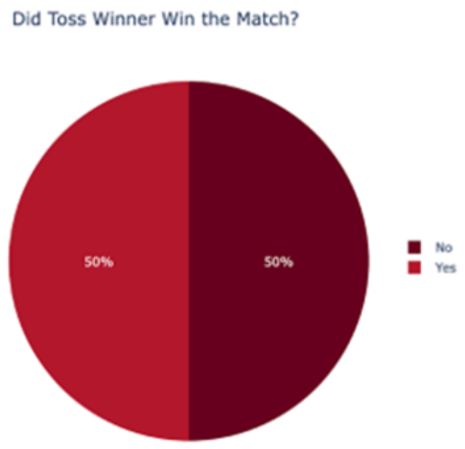
## EDA Images for D1:

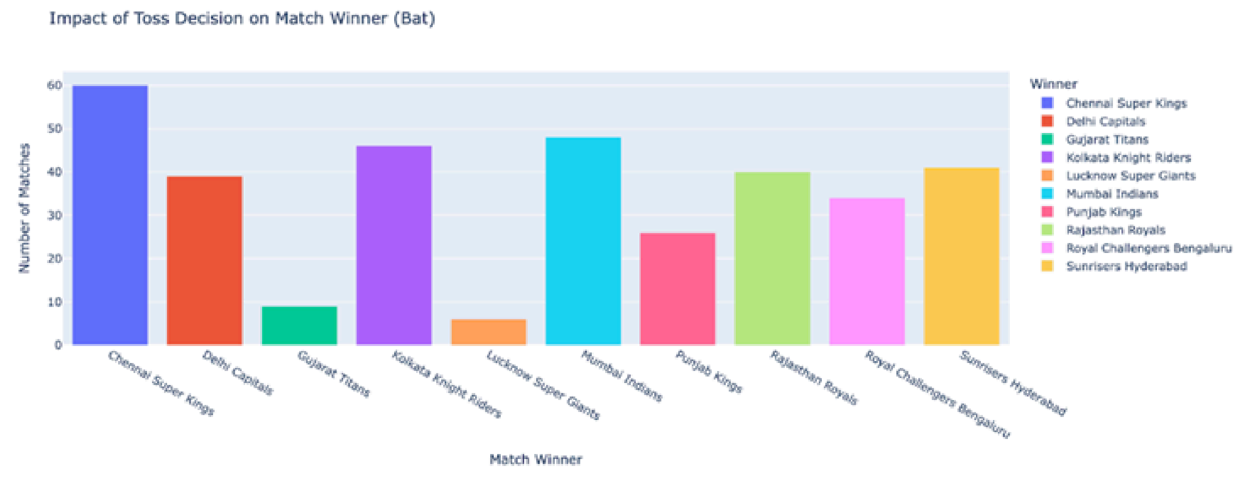


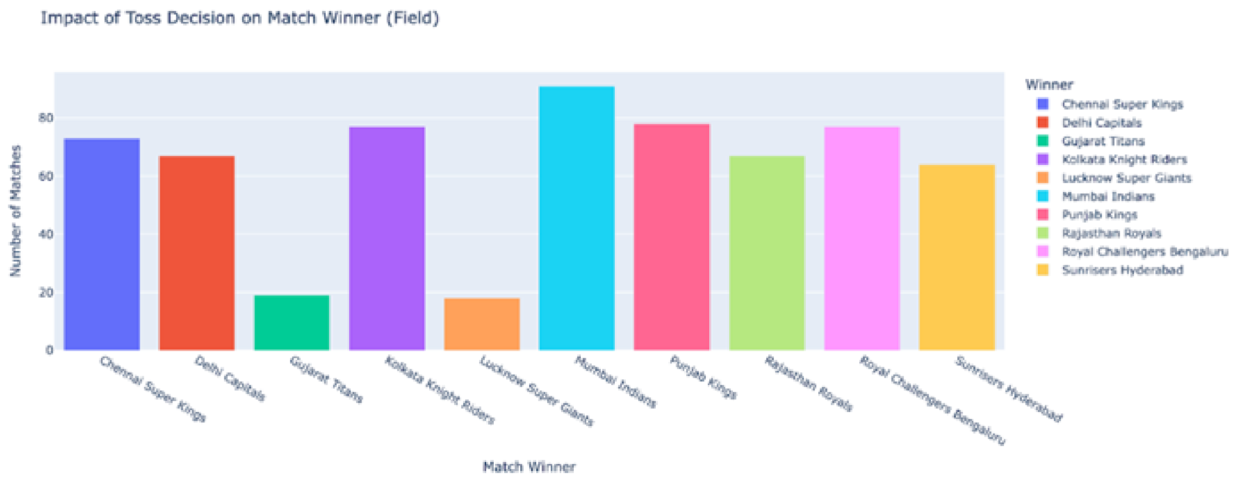


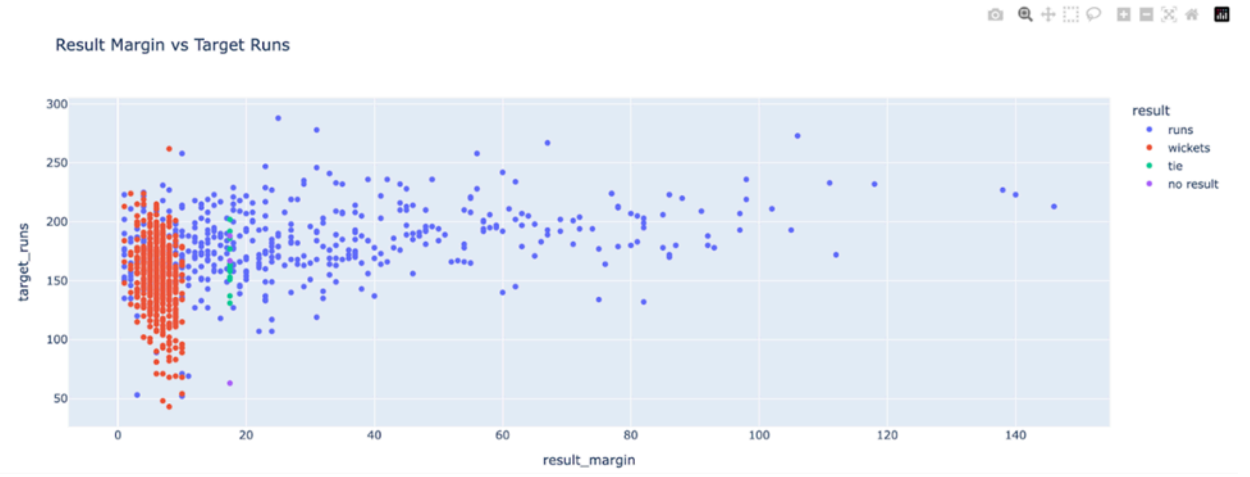




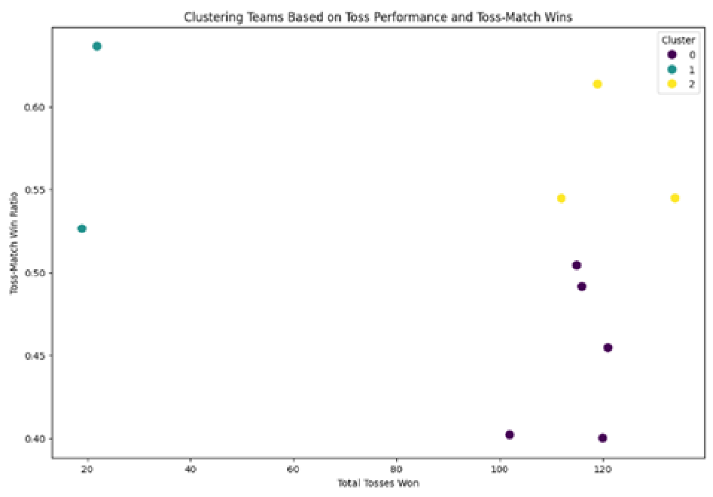


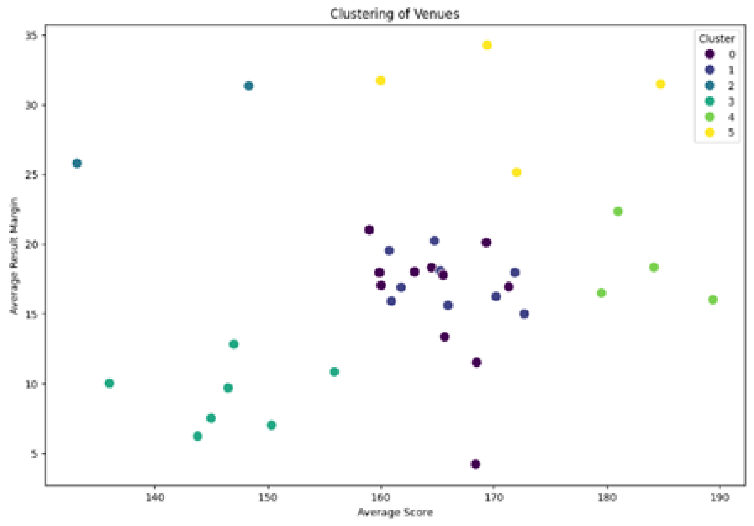

Impact of Toss Decision on Match Winner (Bat)
Number of Matches
Match Winner
Winner
Chennai Super Kings
Delhi Capitals
Gujarat Titans
Kolkata Knight Riders
Lucknow Super Giants
Mumbai Indians
Punjab Kings
Rajasthan Royals
Royal Challengers Bengaluru
Sunrisers Hyderabad

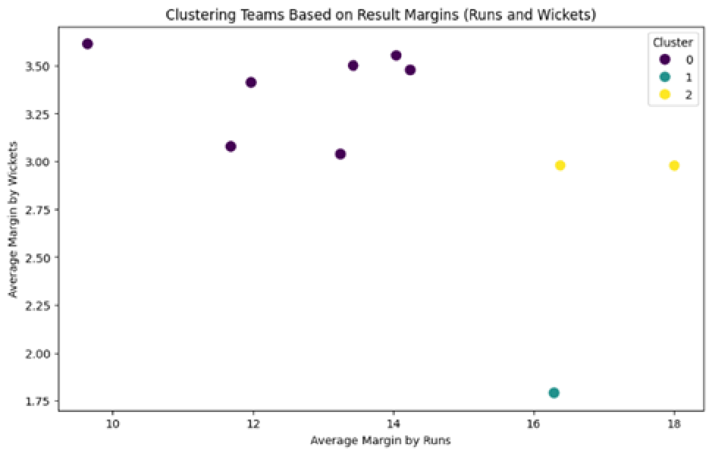

Impact of Toss Decision on Match Winner (Field)
Number of Matches
Match Winner
Winner
Chennai Super Kings
Delhi Capitals
Gujarat Titans
Kolkata Knight Riders
Lucknow Super Giants
Mumbai Indians
Punjab Kings
Rajasthan Royals
Royal Challengers Bengaluru
Sunrisers Hyderabad

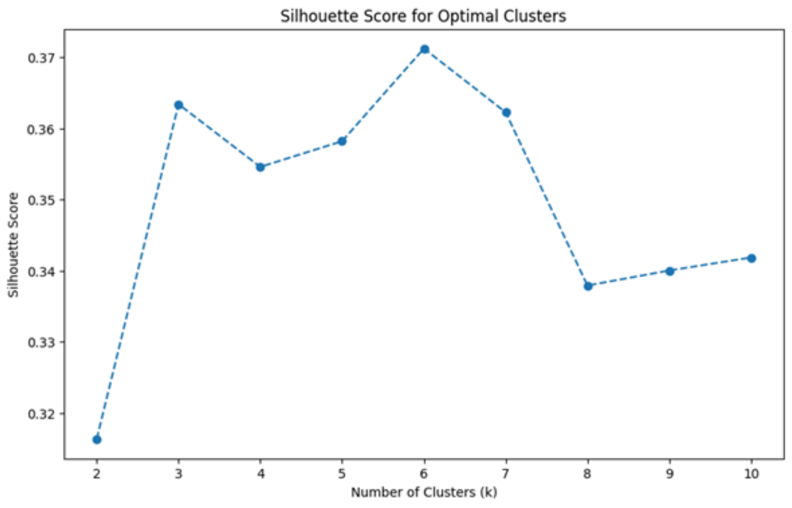

Result Margin vs Target Runs
target_runs
result_margin
result
runs
wickets
tie
no result

## Clustering for D1:

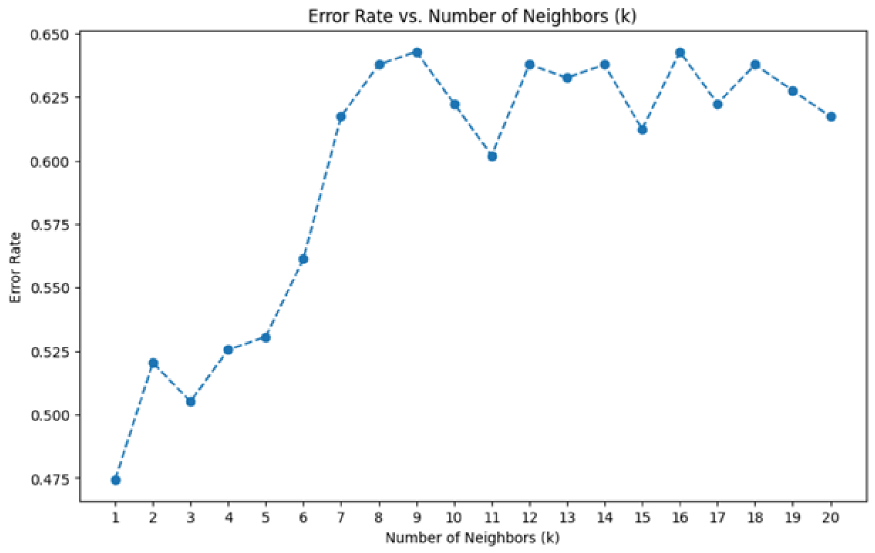




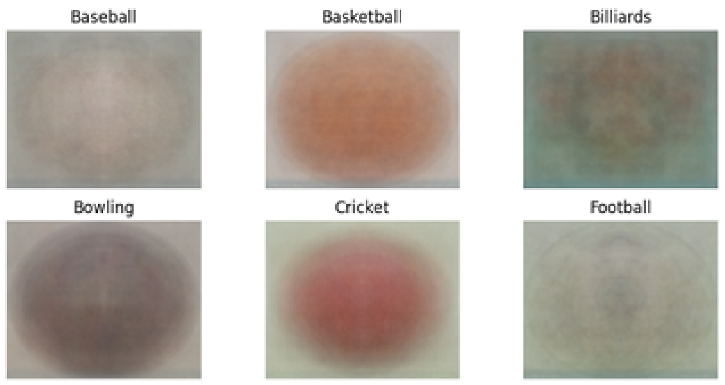


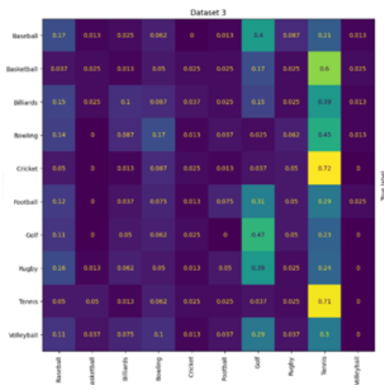


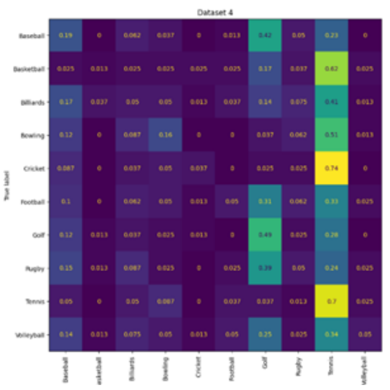
## Finding the optimal number of K for the K-NN Classifier:

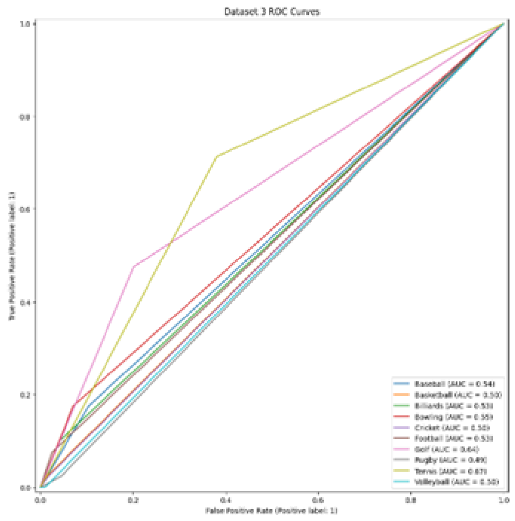


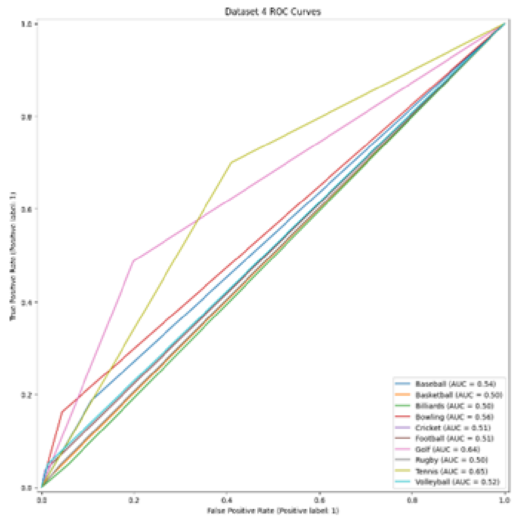
EDA Images for D2:



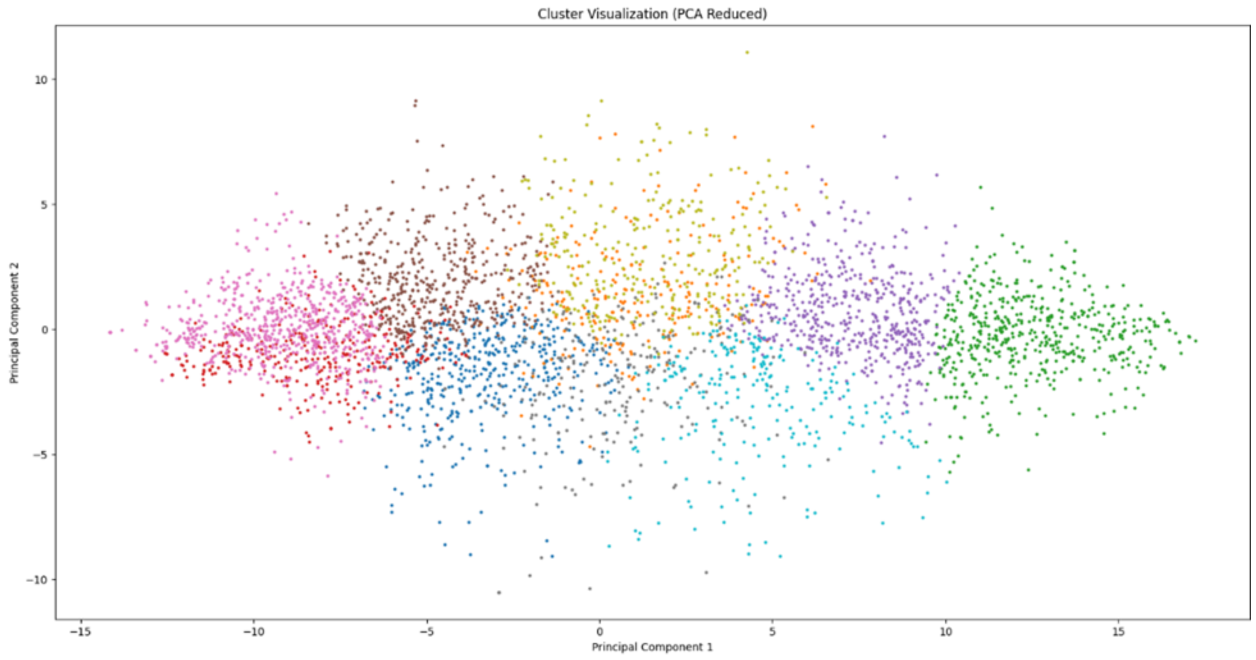


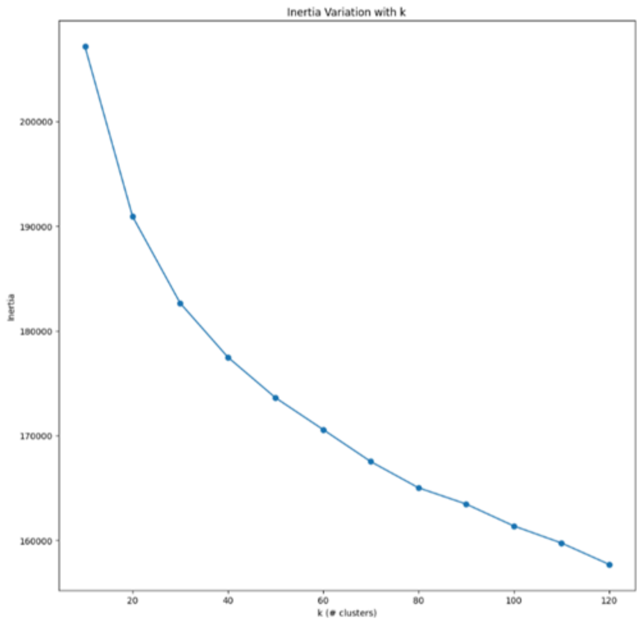


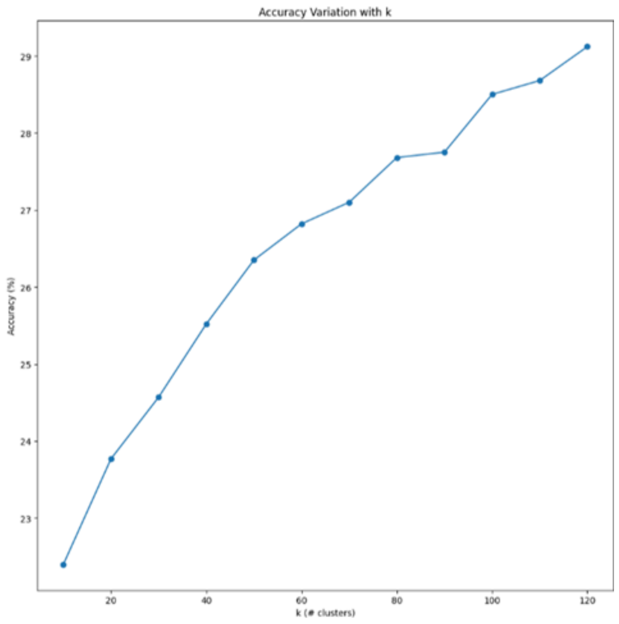




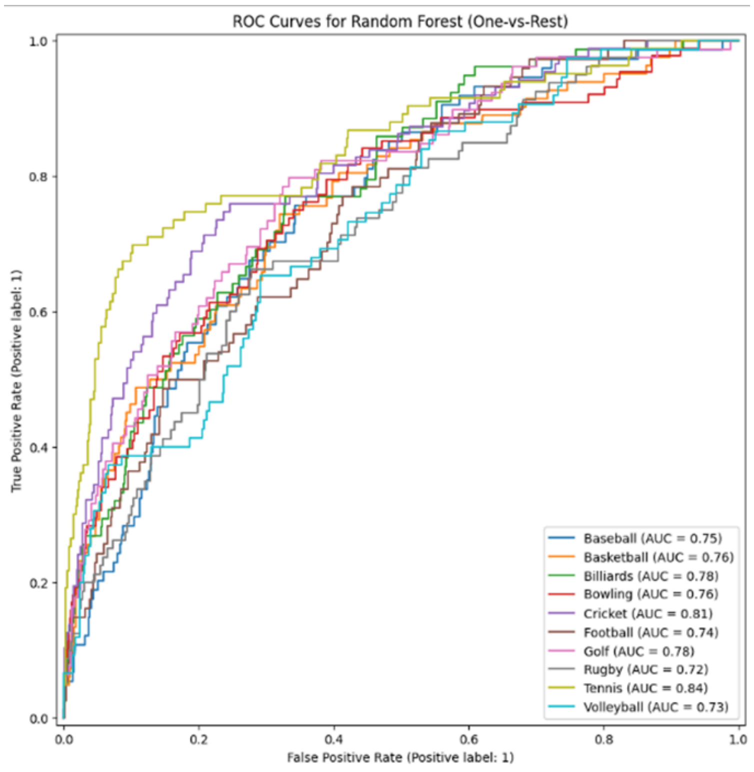
Clustering for D2:



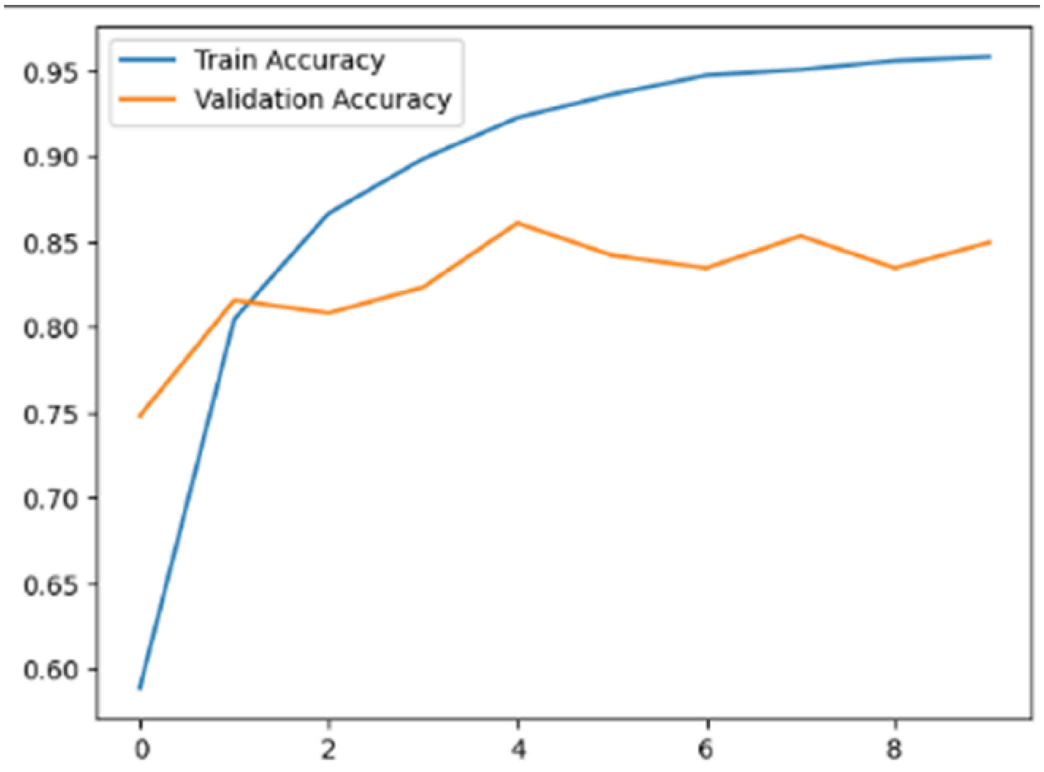




Random Forest Graph for D2:



CNN model Train and Validation graph for D2:


Train Accuracy
Validation Accuracy