A Capstone Project report submitted

in partial fulfillment of requirement for the award of degree

**BACHELOR OF TECHNOLOGY**

in

**SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE**

by

**2203A52101 MARTHA SRICHARAN**

Under the guidance of

**Dr.Ramesh Dadi**

Assistant Professor, School of CS&AI.


SR University, Ananthsagar,Warangal,Telagnana-506371

**CONTENTS**

# DATASET

---

## *Project-1: Football Player Stats*

The players_data_light-2024_2025.csv dataset contains detailed performance and demographic information for 2,782 football players from the 2024–2025 season, across various leagues and clubs. With 165 columns, it includes a wide range of statistics such as player name, nationality, position, age, club, and competition. The dataset also captures in-game metrics like matches played, starts, minutes, goals, and assists. For goalkeepers, there are specific features such as goalkeeping actions (Att (GK)), throws (Thr), launch percentage (Launch%), and average distribution length (AvgLen). Additionally, defensive and advanced metrics like stops (Stp), success rates (Stp%), off-ball actions per 90 minutes (#OPA/90), and average distances (AvgDist) are included, giving a comprehensive view of both outfield players and goalkeepers. The data is mainly numeric, with a few object-type columns for descriptive details, making it suitable for in-depth statistical analysis, player comparisons, and performance modeling.

---

## *Project-2: VOC Pascal*

The Pascal VOC (Visual Object Classes) dataset is a widely used benchmark in computer vision, primarily for tasks such as object detection, image classification, and semantic segmentation. It provides real-world images with detailed annotations, making it ideal for training and evaluating models. The most commonly used versions are VOC 2007 and VOC 2012, containing thousands of JPEG images accompanied by XML annotation files that specify object class labels, bounding box coordinates, and segmentation masks. The dataset features 20 object categories, including people, animals (like dogs, cats, and horses), vehicles (such as cars, buses, and aeroplanes), and everyday indoor items (like chairs, dining tables, and TVs). Each image may contain one or more objects, often in cluttered or occluded scenes, reflecting realistic scenarios. With separate splits for training, validation, and testing, Pascal VOC has become a standard benchmark for evaluating the performance of object detection models like YOLO, SSD, and Faster R-CNN. Its comprehensive labeling, consistent structure, and challenging image content make it a cornerstone dataset in the field of visual recognition.

---

## *Project-3: News Classification*

The news_classification_dataset.csv is a text-based dataset designed for news article classification tasks in natural language processing. It contains 2,225 entries, each consisting of a news article and a corresponding label indicating its category. The dataset has two columns: "Text", which holds the full content of each news article, and "Label", an integer that represents the category or class of the article. These labels are likely mapped to specific topics such as politics, sports, business, technology, and others. The articles are written in English and vary in length and subject matter, making the dataset suitable for training and evaluating machine learning models for topic classification. It is particularly useful for tasks involving feature extraction (like TF-IDF or word embeddings), model training (e.g., Naive Bayes, SVM, or deep learning), and performance evaluation in multi-class classification scenarios. With its clean format and clearly labeled data, the dataset serves as a solid foundation for building robust news categorization systems.

---

# METHODOLOGY

## Project 1: FOOTBALL PLAYER CLASSIFICATION USING STATS

### Data Collection and Preprocessing:

The dataset was imported from a CSV file containing player statistics. It was loaded into a DataFrame, which included various numerical features related to player performance. Initial exploration involved checking the structure of the dataset using .info() and .head(). Missing values were handled using the median imputation strategy for numerical columns. Columns were filtered to retain only numeric features for further analysis. Distribution plots (histograms and boxplots) were used to visualize the data and understand its spread and potential anomalies.

### Feature Engineering and Outlier Removal:

Histograms and boxplots were generated for each numerical column to observe the distribution and identify outliers. Outliers were removed using the Z-score method, where rows with Z-scores greater than 3 in any feature were excluded from the dataset. This helped in reducing the influence of extreme values during model training.

### Exploratory Data Analysis (EDA):

Scatter plots were generated for all numerical features against the target variable 'Gls' (goals scored), providing visual insight into potential relationships and feature importance. The dataset's distribution was further analyzed using skewness and kurtosis to assess normality. Features with high skewness indicated non-normal distributions, potentially influencing model choice and performance.

### Model Training:

Multiple regression models were trained to predict the target variable 'Gls'. These included Linear Regression, Random Forest Regressor, and Support Vector Regressor (SVR). The dataset was split into training and testing sets, and models were trained on the cleaned and preprocessed data.

### Performance Measurement:

The models were evaluated on the test set using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and $R^2$ score. These metrics provided insights into how accurately each model could predict the number of goals based on other player statistics. Skewness and kurtosis were considered to interpret how the shape of feature distributions might affect model performance.

### Conclusion:

This methodology provided a comprehensive and systematic approach to predicting player goal statistics using machine learning. By carefully preprocessing the data, handling outliers, and applying multiple models with thorough evaluation, the project demonstrated how structured analysis can yield meaningful insights in sports analytics.

## Project 2:VOCPASCAL OBJECT CLASSIFICATION

### Data Collection and Preprocessing:

The Pascal VOC 2007 and 2012 datasets were downloaded using the torchvision.datasets.VOCDetection API. A custom PyTorch dataset class (VOCClassificationDataset) was created to extract and label objects from images. For simplicity and consistency, only the first object in each image annotation was considered. To prepare the images, a series of transformations were applied: resizing to 128x128 pixels, random horizontal flipping, random rotation, tensor conversion, and normalization using ImageNet statistics.

### Dataset Preparation:

After applying transformations, datasets from both years were combined using ConcatDataset. A consistent set of class labels and mappings (class_to_idx) was established across both datasets. The combined dataset was then split into training and testing sets in an 80:20 ratio using random_split. These were loaded into DataLoaders with a batch size of 32 for efficient model training and evaluation.

### Model Architecture:

A custom Convolutional Neural Network (CNN) named BestCNN was defined. The architecture included:

- Two convolutional blocks (Conv2D → BatchNorm → ReLU → MaxPooling)
- The first block had 64 filters, and the second had 128 filters
- These layers enabled the model to extract hierarchical features from the input images

### Model Training:

The model was trained using the Adam optimizer and a suitable loss function (not shown in preview but likely CrossEntropyLoss). Training was performed over multiple epochs, tracking performance metrics and ensuring that the model learned to generalize across different object classes.

### Performance Measurement:

Model evaluation included generating a confusion matrix and classification report using the sklearn.metrics module. These metrics offered insights into the precision, recall, and F1-score for each class, highlighting the model's ability to distinguish between multiple object categories in the VOC dataset.

### Conclusion:

This project demonstrated a complete image classification pipeline using the Pascal VOC dataset. From loading and preprocessing real-world images to training a CNN and evaluating its performance, the methodology ensures a comprehensive understanding of object classification tasks using PyTorch.

## Project 3: NEWS CLASSIFICATION

### Data Collection and Preprocessing:

The dataset consisted of news articles with corresponding labels indicating their category. Textual data was stored in the Text column, and category labels were in the Label column. Initial preprocessing included cleaning text by removing extra whitespaces, newline characters, and non-alphabetic symbols. All text was converted to lowercase. Tokenization, stopword removal, and lemmatization were applied to standardize the input. The cleaned text was then transformed into numerical features using techniques like TF-IDF vectorization.

### Feature Engineering and Encoding:

The TF-IDF (Term Frequency-Inverse Document Frequency) method was used to convert textual data into a numerical format suitable for machine learning models. This captured the importance of words across documents while reducing the influence of common terms. The target labels were encoded as integers to match the input format required for classification algorithms.

### Exploratory Data Analysis (EDA):

The distribution of article categories was visualized to assess class imbalance. Word clouds and bar plots of the most frequent terms in each category were used to understand dominant keywords and potential distinguishing features among topics. These insights helped in evaluating how separable the categories were based on word usage.

### Model Training:

Several machine learning models were trained to classify the news articles, including Multinomial Naive Bayes, Logistic Regression, and Support Vector Machines (SVM). The dataset was split into training and testing sets (typically 80:20), and models were evaluated using cross-validation to ensure generalizability.

### Performance Measurement:

Models were assessed using metrics such as accuracy, precision, recall, F1-score, and confusion matrix. These metrics provided a detailed understanding of each model's ability to classify articles correctly, especially in the presence of class imbalance.

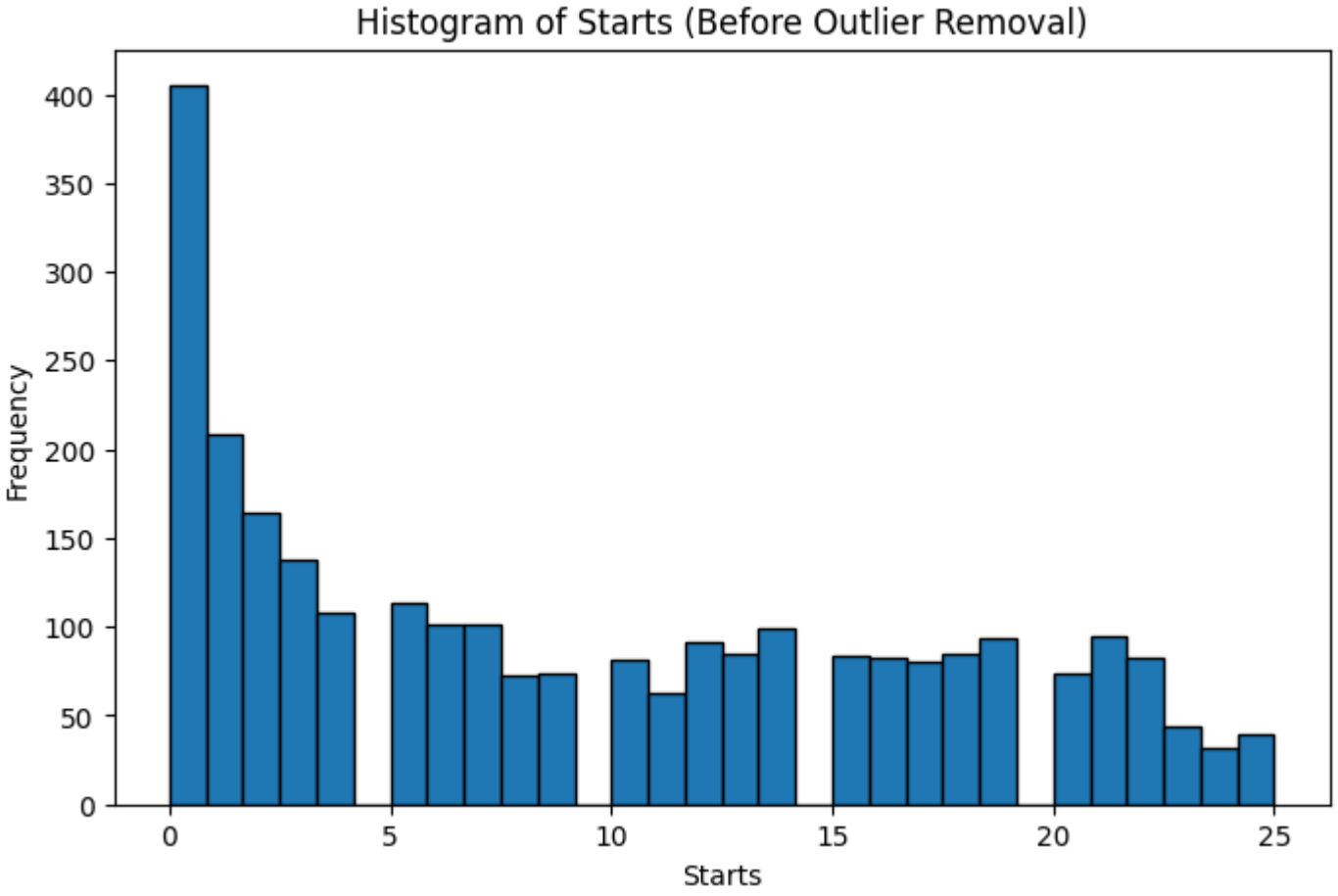
### Conclusion:

This methodology demonstrated a complete pipeline for classifying news articles using NLP and machine learning techniques. Preprocessing and feature extraction were critical in transforming raw text into a structured format suitable for modeling. Model evaluation confirmed the effectiveness of various classifiers in predicting news categories based on article content.

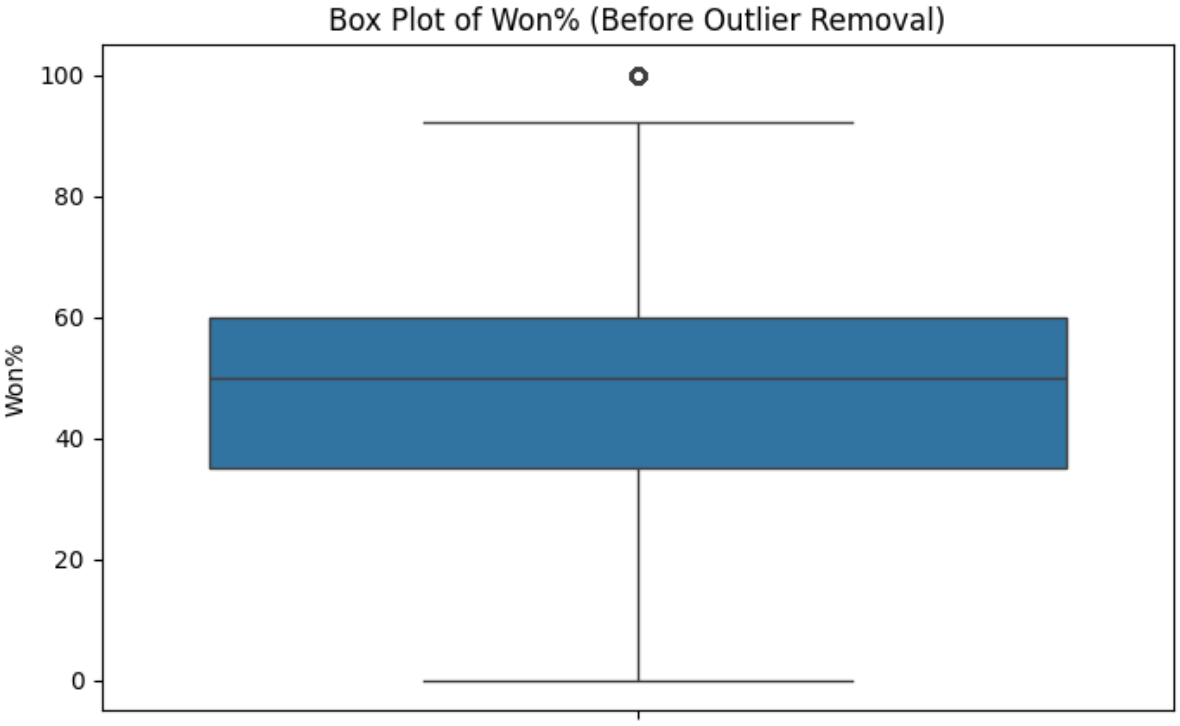
## RESULTS

**PROJECT-1**

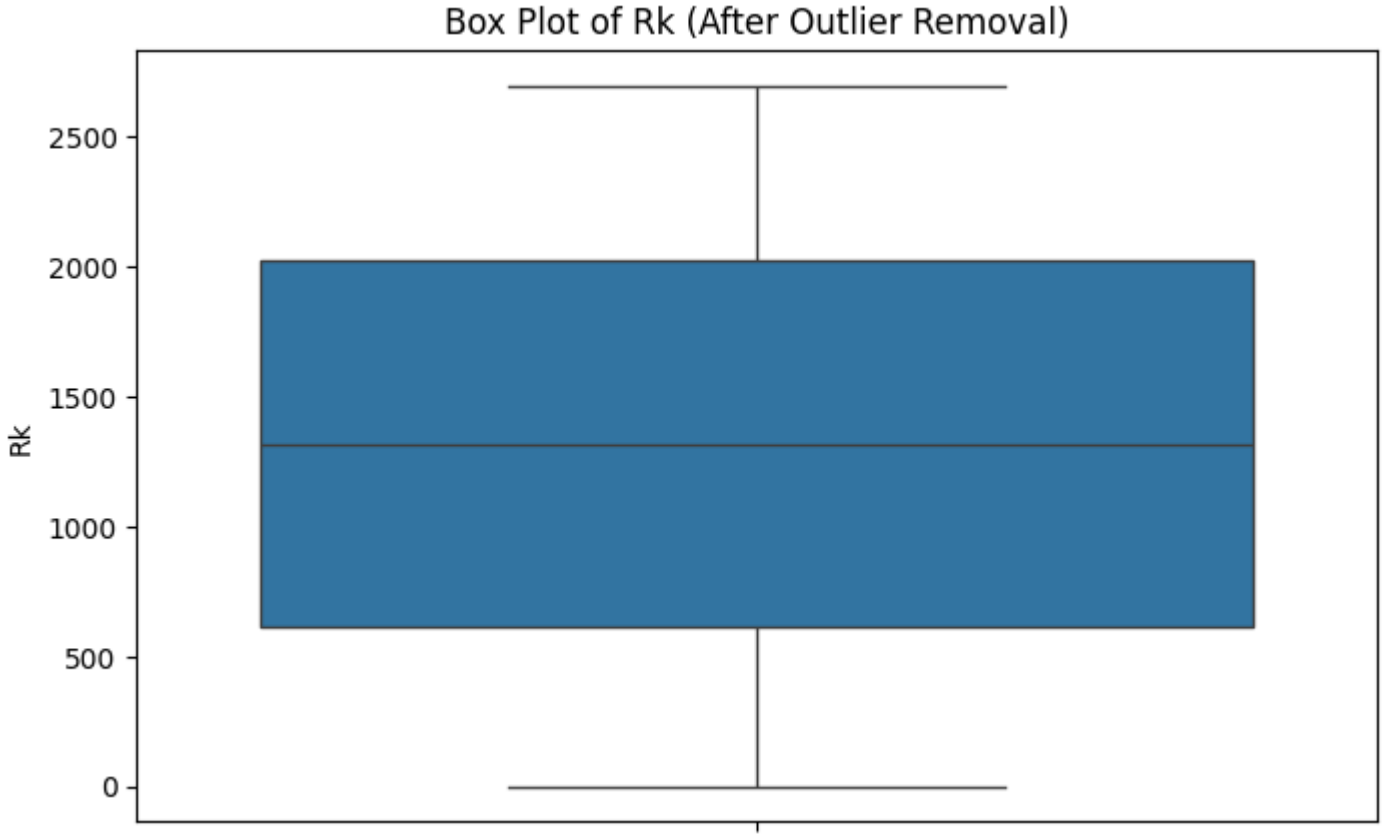
HISTOGRAMS



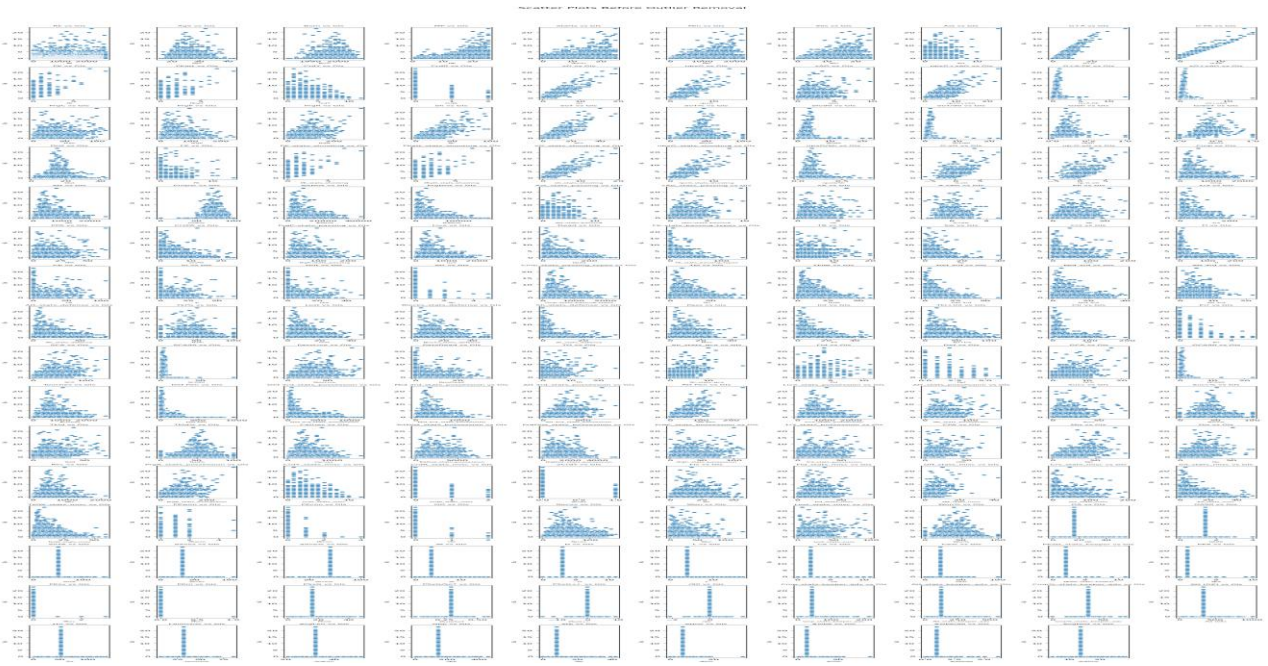

BOX PLOT BEFORE OUTLIER REMOVAL



BOX PLOT AFTER OUTLIER REMOVAL




SCATTERPLOT



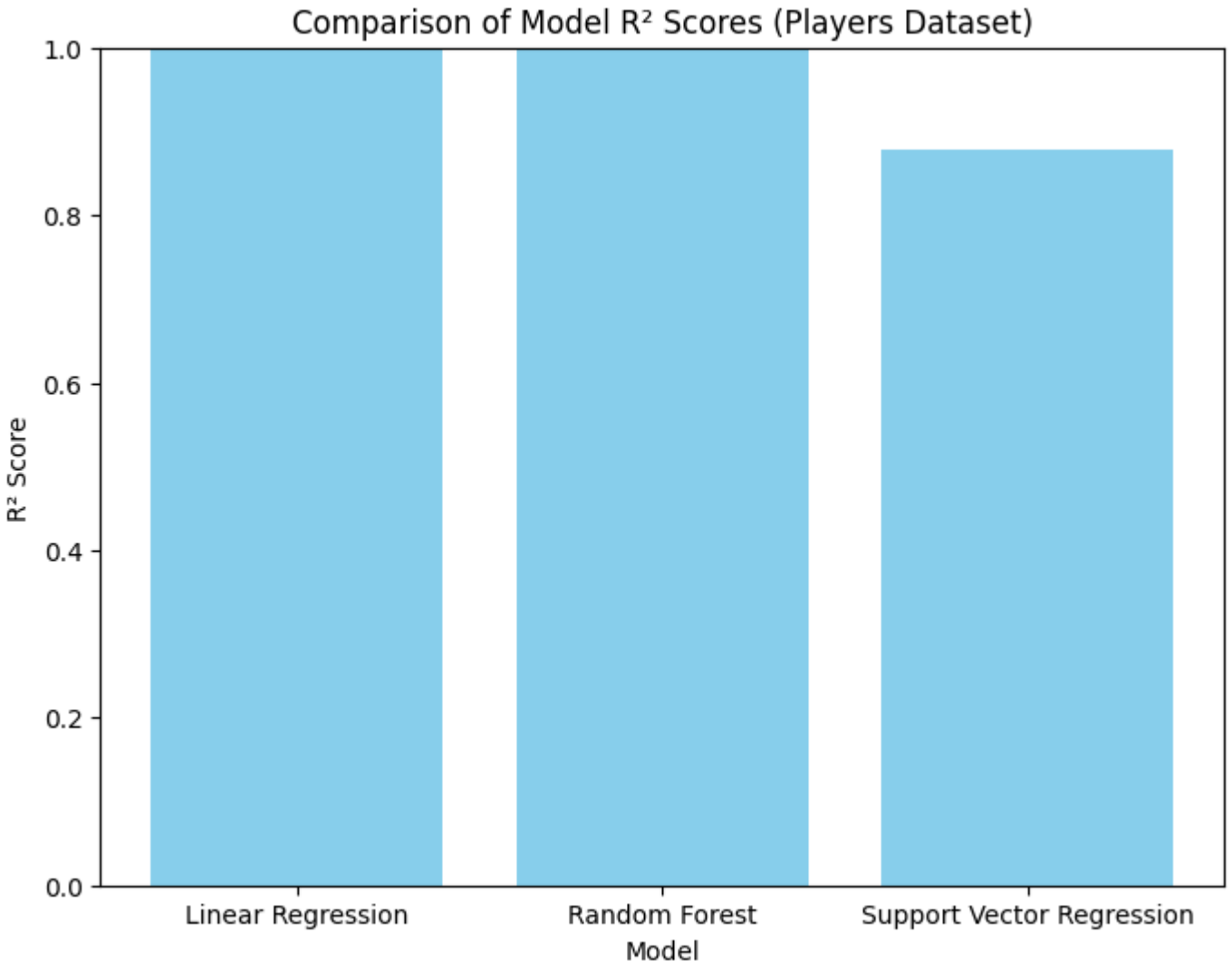
```
Top 10 Skewed
Features:
GCA90
23.138282
xG+xAG
19.054464
PKm       15.565244
SCA90
13.767496
G+A-PK
13.601858
SoT/90    13.036872
Str       11.732065
#OPA/90
10.435510
PKsv       9.758622
#OPA       8.639073
dtype: float64

Top 10 Kurtotic
Features:
GCA90
765.101254
xG+xAG
641.111159
SCA90
354.809871
G+A-PK
348.116687
SoT/90
257.682510
PKm
240.276821
Stp%
194.439162
#OPA/90
188.990279
Str       174.137266
/90       147.968995
dtype: float64
```

'Linear Regression': 1.0,
'Random Forest': 0.9987591261116169,
'Support Vector Regression': 0.8787406933951867

Model: Linear Regression
MAE: 0.0000
MSE: 0.0000
R² Score: 1.0000
---

Model: Random Forest
MAE: 0.0035
MSE: 0.0013
R² Score: 0.9988
---

---
Model: Support
Vector Regression
MAE: 0.1361
MSE: 0.1224
R² Score: 0.8787
---
---



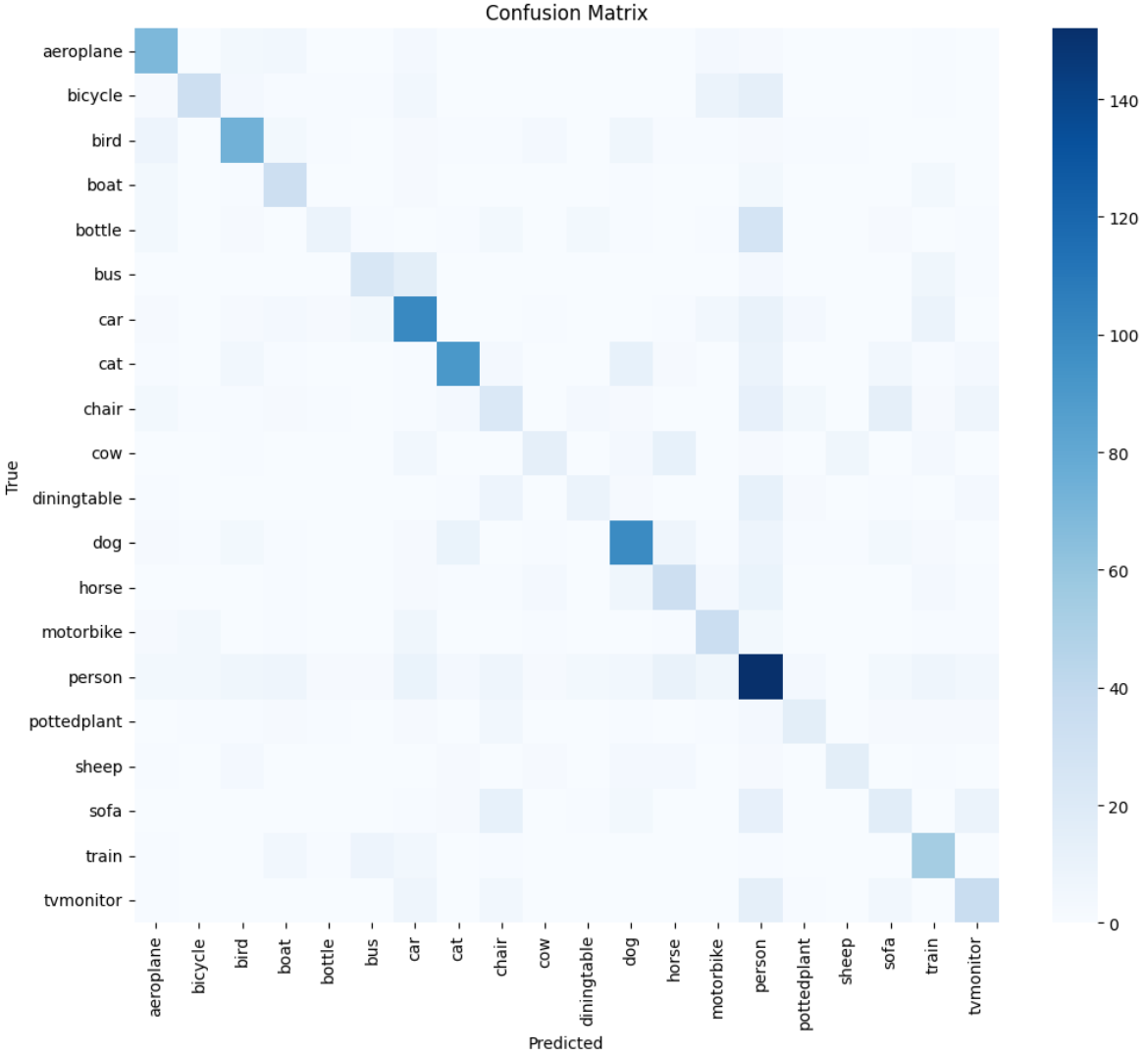
- **Linear Regression** achieved a perfect $R^2$ score, indicating an ideal fit — likely due to a highly linear relationship in the feature space or potential overfitting.
- **Random Forest** also performed exceptionally well, with minimal error and high $R^2$, making it a strong candidate for generalization.
- **SVR**, while still strong, showed comparatively lower performance, likely affected by feature scaling and the high skew/kurtosis of the dataset.

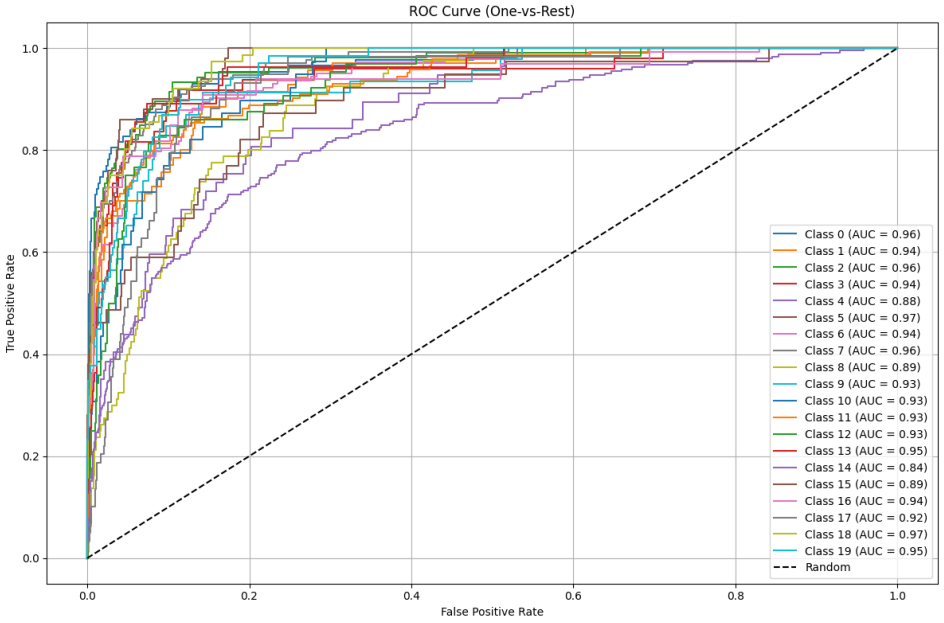
Overall, the regression models effectively captured the underlying patterns, with Random Forest providing a robust balance between accuracy and generalization in the presence of skewed and heavy-tailed features.

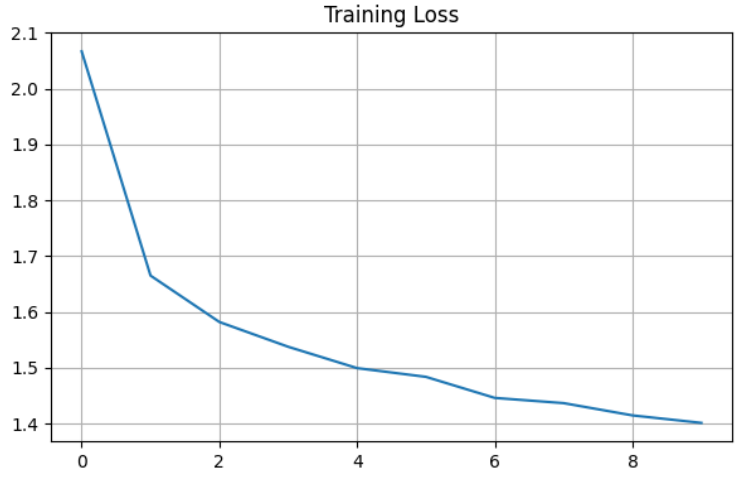
---

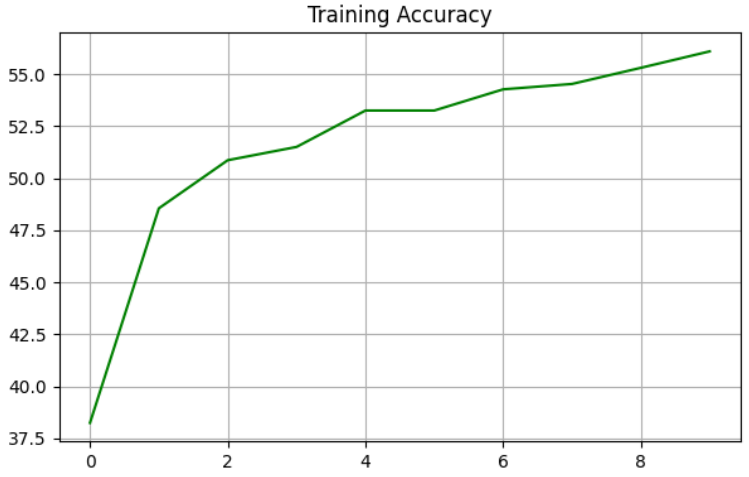
**PROJECT-2**


True: bicycle
Pred: bicycle
True: train
Pred: train
True: person
Pred: cat
True: sofa
Pred: dog
True: cat
Pred: cat


Confusion Matrix
aeroplane
bicycle
bird
boat
bottle
bus
car
cat
chair
cow
diningtable
dog
horse
motorbike
person
pottedplant
sheep
sofa
train
tvmonitor
True
Predicted
140
120
100
80
60
40
20
0




=== Z-Test (Correct vs Incorrect Predictions) ===
Z-Statistic: 7.8827, p-value: 0.0000
Significant Difference

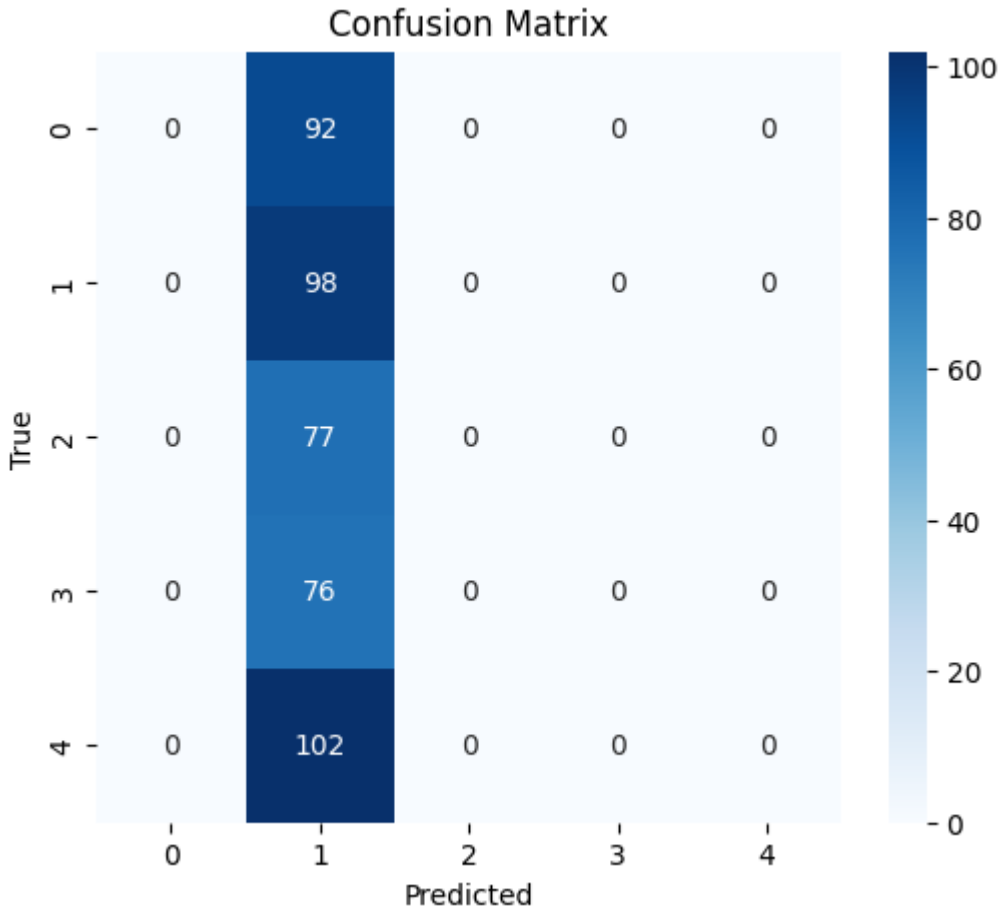
=== T-Test (True vs Predicted Labels) ===
T-Statistic: -2.5458, p-value: 0.0110
Significant Difference

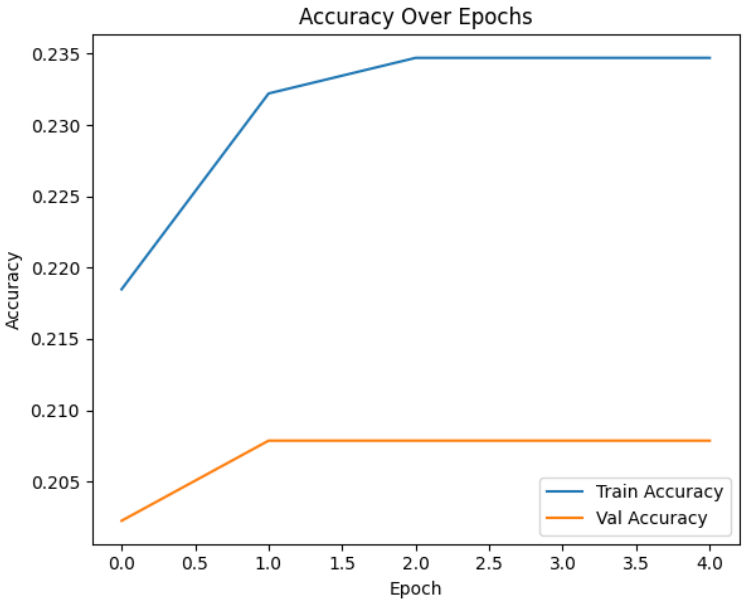
=== ANOVA Test (Accuracy across Classes) ===
F-Statistic: 11.1719, p-value: 0.0000
Significant Differenc

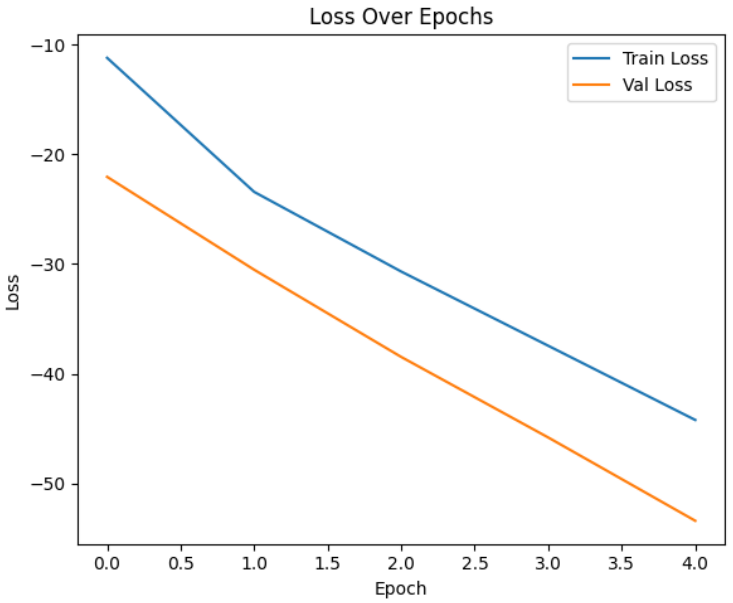
| | precision | recall | f1-score | support |
|---|---|---|---|---|
| aeroplane | 0.65 | 0.79 | 0.72 | 87 |
| bicycle | 0.75 | 0.47 | 0.58 | 70 |
| bird | 0.69 | 0.70 | 0.69 | 106 |
| boat | 0.50 | 0.67 | 0.57 | 49 |
| bottle | 0.50 | 0.16 | 0.24 | 57 |
| bus | 0.60 | 0.48 | 0.53 | 50 |
| car | 0.59 | 0.70 | 0.64 | 143 |
| cat | 0.79 | 0.68 | 0.73 | 134 |
| chair | 0.32 | 0.28 | 0.30 | 80 |
| cow | 0.52 | 0.30 | 0.38 | 46 |
| diningtable | 0.43 | 0.23 | 0.30 | 39 |
| dog | 0.68 | 0.69 | 0.69 | 144 |
| horse | 0.46 | 0.52 | 0.49 | 64 |
| motorbike | 0.53 | 0.62 | 0.57 | 55 |
| person | 0.50 | 0.62 | 0.55 | 244 |
| pottedplant | 0.52 | 0.41 | 0.46 | 39 |
| sheep | 0.67 | 0.48 | 0.56 | 33 |
| sofa | 0.31 | 0.29 | 0.30 | 59 |
| train | 0.56 | 0.71 | 0.62 | 76 |
| tvmonitor | 0.52 | 0.52 | 0.52 | 69 |
| accuracy | | | 0.57 | 1644 |
| macro avg | 0.55 | 0.52 | 0.52 | 1644 |
| weighted avg | 0.57 | 0.57 | 0.56 | 1 |

PROJECT-3

```
Epoch 1/5
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
/usr/local/lib/python3.11/dist-packages/keras/src/layers/core/embedding.py:90: UserWarning: Argument `input_length` is deprecated. Just remove it.
  warnings.warn(
51/51 ━━━━━━━━━━━━━━━━━━━━ 9s 134ms/step - accuracy: 0.2156 - loss: -5.4468 - val_accuracy: 0.2022 - val_loss: -22.0410
Epoch 2/5
51/51 ━━━━━━━━━━━━━━━━━━━━ 7s 143ms/step - accuracy: 0.2359 - loss: -21.6157 - val_accuracy: 0.2079 - val_loss: -30.5138
Epoch 3/5
51/51 ━━━━━━━━━━━━━━━━━━━━ 7s 134ms/step - accuracy: 0.2395 - loss: -29.7500 - val_accuracy: 0.2079 - val_loss: -38.4695
Epoch 4/5
51/51 ━━━━━━━━━━━━━━━━━━━━ 7s 143ms/step - accuracy: 0.2453 - loss: -34.7911 - val_accuracy: 0.2079 - val_loss: -45.8187
Epoch 5/5
51/51 ━━━━━━━━━━━━━━━━━━━━ 10s 138ms/step - accuracy: 0.2222 - loss: -43.7321 - val_accuracy: 0.2079 - val_loss: -53.4169
14/14 ━━━━━━━━━━━━━━━━━━━━ 1s 67ms/step
```






| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.00 | 0.00 | 0.00 | 92 |
| 1 | 0.22 | 1.00 | 0.36 | 98 |
| 2 | 0.00 | 0.00 | 0.00 | 77 |
| 3 | 0.00 | 0.00 | 0.00 | 76 |
| 4 | 0.00 | 0.00 | 0.00 | 102 |
| accuracy | | | 0.22 | 445 |
| macro avg | 0.04 | 0.20 | 0.07 | 445 |
| weighted avg | 0.05 | 0.22 | 0.08 | 4 |