

**INSTITUTE FOR ADVANCED COMPUTING AND**

**SOFTWARE DEVELOPMENT, AKURDI, PUNE**

**Olympics Data Analysis & Prediction Of Medals**

PG-DBDA March 2024

Submitted By:

Group No: G1

|  |  |
| --- | --- |
| Roll No. | Name |
| 243513 | Dhavale Alisha Satish |
| 243548 | Hedau Sharvary Snehadeep |

**Mr. Abhijit Nagargoje Mr. Rohit Puranik**

Project Guide Centre Coordinator

**ABSTRACT**

The Olympic Games represent the pinnacle of athletic achievement, where nations from around the world compete for the honour of winning medals. Predicting the outcomes of these competitions, particularly the distribution of medals, is a complex task due to the multitude of factors that influence athletic performance. This project, titled "Olympics Data Analysis & Prediction of Medals" aims to leverage historical Olympic data to develop a predictive model for medal outcomes.

The analysis incorporates a comprehensive dataset that includes variables such as athletes' previous performances, country-specific attributes, socio-economic factors, and other relevant historical data. Machine learning techniques, including classification algorithms and ensemble methods, are employed to uncover patterns and trends that contribute to an athlete's or a country's likelihood of winning medals.

The project's findings offer valuable insights into the determinants of Olympic success and present a predictive framework that can be utilized for future games. This analysis not only enhances our understanding of the factors driving Olympic success but also provides a robust tool for sports analysts and enthusiasts who seek to forecast outcomes in an increasingly competitive field.

**ACKNOWLEDGEMENT**

We would like to extend our sincere gratitude to everyone who supported us throughout this project. Our heartfelt thanks go to Mr. Abhijit Nagargoje for their guidance and invaluable feedback. We also appreciate the assistance provided by Mrs. Priyanka Bhor, Mrs. Priti Take and Mr. Shantanu Pathak for their insights and suggestions.

Special thanks to the developers of the libraries and tools we used, including scikit-learn, CatBoost, Plotly, Seaborn, and Streamlit, which were crucial to the success of our project.

Additionally, we are grateful to the broader data science community and open-source contributors whose resources and tutorials facilitated our understanding and implementation of complex techniques. Their shared knowledge and tools were instrumental in navigating challenges and achieving our project goals.

Lastly, we are grateful to our peers and friends for their encouragement and support during this endeavor.

Dhavale Alisha Satish 240341225013

Hedau Sharvary Snehadeep 240341225057

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Content** | **Page No.** |
| 1. | **Introduction** | 1 |
| 1.1 | Problem Statement | 1 |
| 1.2 | Product Scope | 1 |
| 1.3 | Aims & Objectives | 1 |
| 2. | **Overall Description** | 2 |
| 2.1 | Workflow Of Project | 2 |
| 2.2 | Data Preprocessing And Cleaning | 3 |
| 2.2.1 | Data Cleaning | 3 |
| 2.2.2 | Imputation Techniques | 3 |
| 2.2.3 | Removing Duplicates | 3 |
| 2.2.4 | One Hot Encoding | 4 |
| 2.3 | Exploratory Data Analysis | 5 |
| 2.4 | Model Building | 8 |
| 2.4.1 | Train/Test Split | 9 |
| 2.4.2 | Logistic Regression | 10 |
| 2.4.3 | Naïve Bayes | 12 |
| 2.4.4 | K Nearest Neighbors | 14 |
| 2.4.5 | Linear SVC | 16 |
| 2.4.6 | Random Forest Model | 18 |
| 2.4.7 | Xgboost | 20 |
| 2.4.8 | Catboost | 22 |
| 2.4.9 | ARIMA | 24 |
| 3. | **User Interface** | 26 |
| 4. | **Requirements Specification** | 29 |
| 4.1 | Hardware Requirement | 29 |
| 4.2 | Software Requirement | 29 |
| 5. | **Conclusion** | 30 |
| 6. | **Future Scope** | 31 |
| 7. | **References** | 32 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Description** | **Page No.** |
| Figure1 | Workflow Diagram | 2 |
| Figure 2 | Line chart showing the distribution of the athletes participated over countries | 5 |
| Figure 3 | Pie chart showing the distribution of the athletes according to gender | 6 |
| Figure 4 | Bar plot of the Top 10 teams | 6 |
| Figure 5 | Top 10 Sports from different countries over the number of participants | 7 |
| Figure 6 | Histograms showing Age distribution of athletes | 7 |
| Figure 7 | Box plot of the distribution of age by sports | 8 |
| Figure 8 | Line chart shows the comparison of medal wins among the top Countries | 8 |
| Figure 9 | Example of logistic regression function | 9 |
| Figure.10 | The accuracy score of logistic regression model | 11 |
| Figure 11 | Naïve Bayes formula | 12 |
| Figure 12 | The classification report of Naïve bayes model | 13 |
| Figure 13 | Example of KNN | 13 |
| Figure 14 | The classification report of KNN model | 14 |
| Figure 15 | The data points for present in the dataset | 16 |
| Figure 16 | SVC gives us the decision boundary between the distinct data | 17 |
| Figure 17 | The classification report of Linear SVC model | 17 |
| Figure 18 | Working of the random forest model | 18 |
| Figure 19 | The classification report of random forest model | 19 |
| Figure 20 | XG boost (Extreme gradient Boosting) | 20 |
| Figure 21 | The classification report of the XGBoost model | 21 |
| Figure 22 | The classification report of the CatBoost model | 22 |
| Figure 23 | Comparison of all algorithm performed on the data set | 23 |
| Figure 24 | Result of ARIMA model | 25 |
| Figure 25 | Forecasting of Olympics Medal wins by country | 25 |
| Figure 26 | The web application where the user can do medal tally by selecting specific Year and Country | 26 |
| Figure 27 | The web application with Overall Analysis | 27 |
| Figure 28 | Country-wise analysis for a particular country | 27 |
| Figure 29 | Athlete wise analysis of height vs Weight for a particular sport | 28 |

1. **Introduction**

**1.1 Problem Statement**

The Olympic Games represent one of the most significant international sporting events, attracting athletes from all over the world. Understanding the factors that contribute to a country’s success in the Olympics, particularly in terms of medal counts, is of great interest to sports analysts, policymakers, and enthusiasts. This project aims to analyze historical Olympic data to identify trends, patterns, and factors that influence medal counts. Furthermore, the project seeks to develop a predictive model that can forecast

the number of medals a country might win in future Olympic Games.

**1.2 Product Scope**

The “Olympics Data Analysis & Prediction Of Medals” project is designed to predict medal outcomes based on historical data from the Olympic Games. This scope defines the objectives, key features, and deliverables of the project, outlining the boundaries and expectations for the final product.

**1.3 Aims & Objectives**

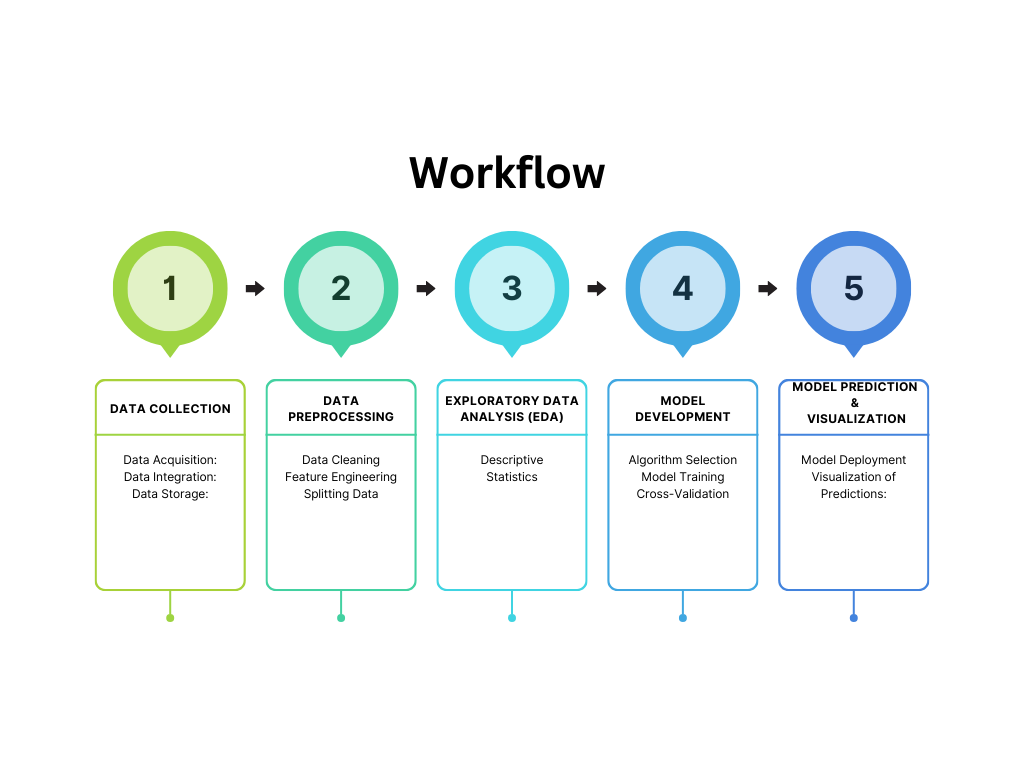
The primary aim of the "Olympics Data Analysis & Prediction Of Medals" project is to develop a predictive model that accurately forecasts medal outcomes in future Olympic Games based on historical data. The data is from Athens 1896 to Rio 2016. This project seeks to uncover the underlying patterns and factors that contribute to an athlete's or a nation's success at the Olympics, providing valuable insights into the determinants of medal achievements.

Before giving the question the training of the models will be done using a bunch of different ML models and after the training is done the ML models will be compared based on their accuracy score and f1-score and the best model will be selected which will then be used to make predictions for the number of medal given.

1. **Overall Description**

**2.1 Workflow of Project**

The diagram below shows the workflow of this project.



*Figure 1: Workflow Diagram*

**2.2 Data Preprocessing and Cleaning**

**2.2.1 Data Cleaning**

Identify Missing Values: Use techniques such as heatmaps or summary statistics to detect missing values in the dataset

**2.2.2 Imputation Techniques**

* Numerical Data: Replace missing values using mean, median, or mode imputation. Advanced methods like K-Nearest Neighbours (KNN) imputation or regression imputation may also be considered if the missing data is significant.
* Categorical Data: Use the most frequent value (mode) or introduce a new category like "Unknown" or "Not Available" to handle missing categorical data.

**2.2.3 Removing Duplicates**

* Identification of Duplicates: Use functions like drop duplicates() to identify and remove any duplicate entries in the dataset.
* Data Integrity Check: Ensure that removing duplicates does not inadvertently remove important data. Examine patterns of duplication to understand their cause.

**2.2.4 One Hot Encoding**

* Representation of Categorical Data: One-hot encoding transforms categorical variables into a series of binary columns, where each unique category is represented by a separate column. This method ensures that the machine learning model interprets categorical data correctly, without assuming any ordinal relationship between the categories.
* Identify Categorical Variables: Determine which features in the dataset are categorical and need to be one-hot encoded. Common examples in the Olympics dataset might include Country, Sport, Event, Gender, and Medal Type.
* Apply One-Hot Encoding:
  + Use libraries like pandas in Python to apply one-hot encoding. For instance, pd.get\_dummies() can be used to transform categorical columns into one-hot encoded columns.
  + Alternatively, scikit-learn’s One Hot Encoder can be used, especially when you want to apply encoding within a machine learning pipeline.
* Handling Large Numbers of Categories:

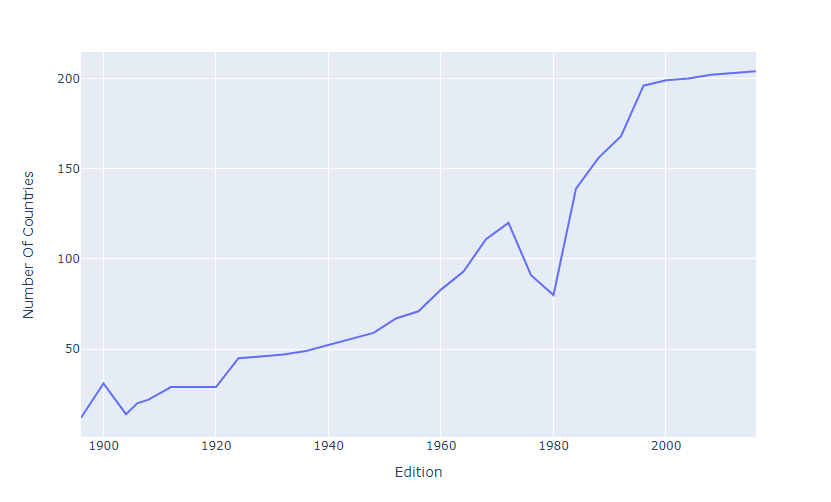
Dimensionality Consideration: If a categorical feature has a large number of unique categories (e.g., countries or events), one-hot encoding can lead to a high-dimensional dataset. This increase in dimensionality can be managed by applying techniques such as feature selection, dimensionality reduction (e.g., PCA), or by considering alternative encoding methods like target encoding.

* Column Naming Conventions: Ensure that the new columns created by one-hot encoding are named appropriately, typically combining the original column name with the category value (e.g., Country USA, Sport Swimming).

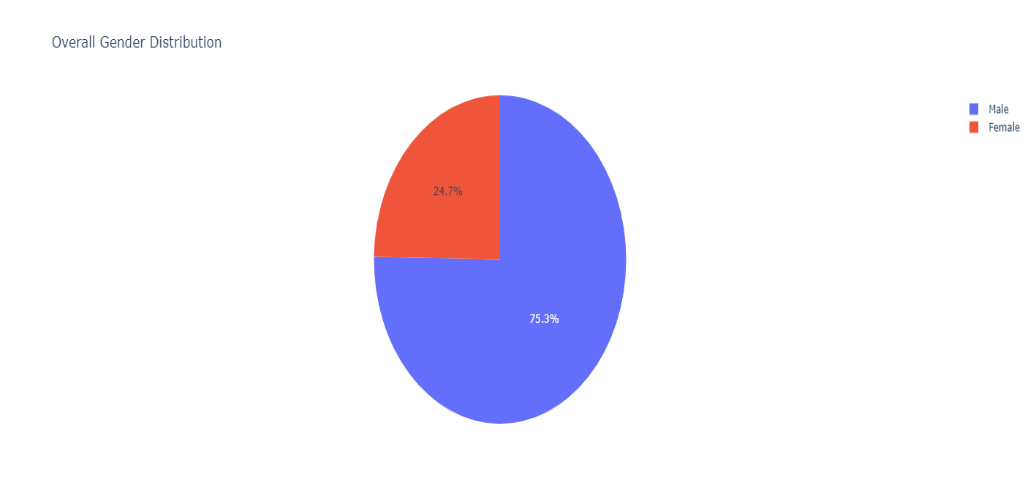
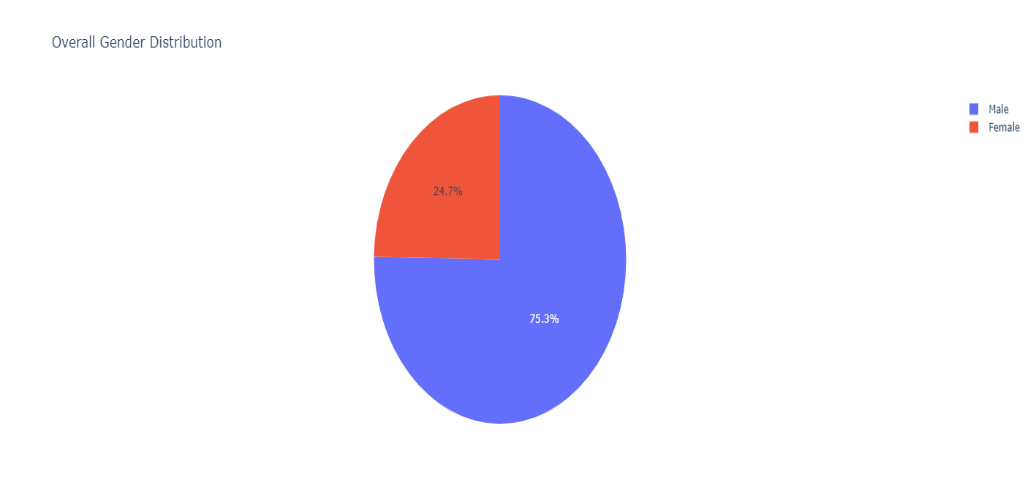
**2.3 Exploratory Data Analysis**

Exploratory Data Analysis (EDA) is a critical phase in the "Olympics Data Analysis & Prediction Of Medals" project. It involves analysing the dataset to uncover patterns, trends, and relationships among the variables. EDA helps in understanding the data's structure, identifying potential outliers, and formulating hypotheses that guide the subsequent modelling process.

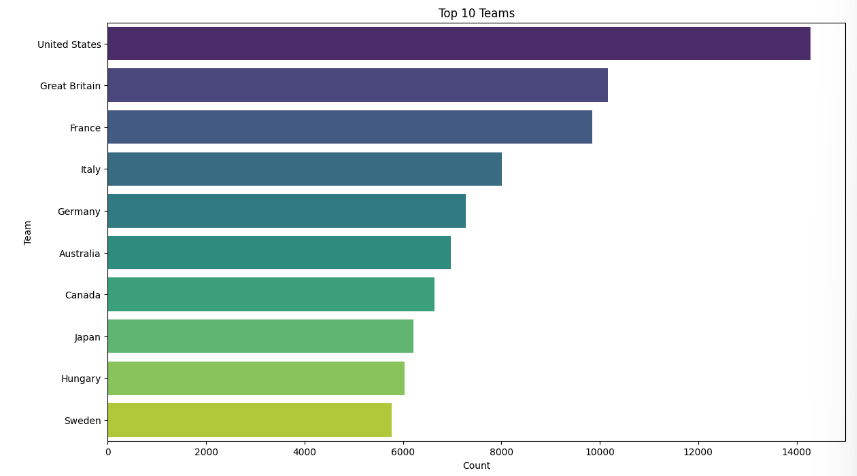
Following are some plots we used to extract some useful information:



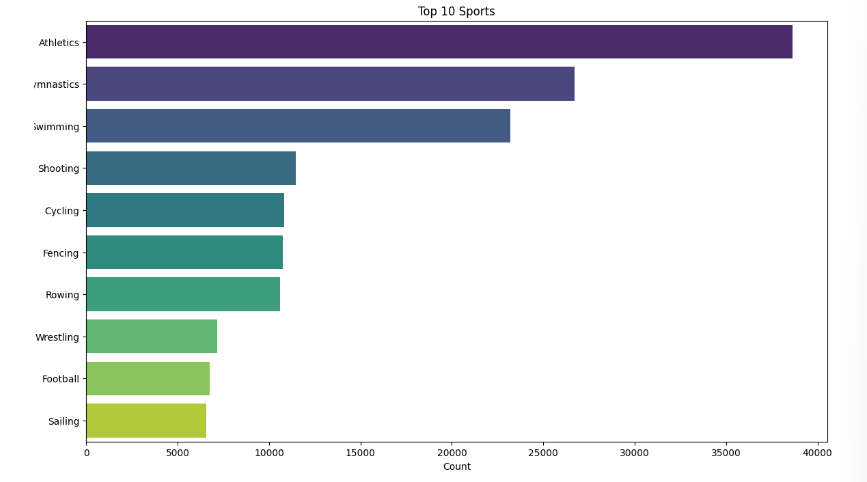
*Figure 2: Line chart showing the distribution of the athletes participated over countries*



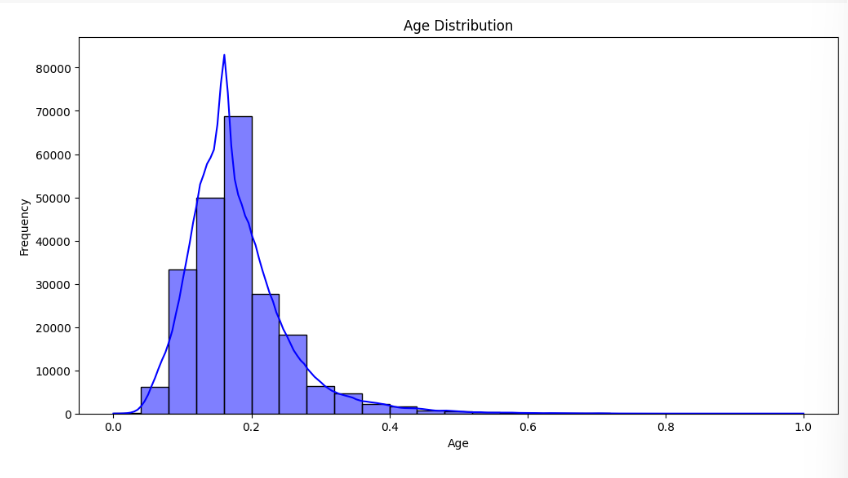
*Figure 3: Pie chart showing the distribution of the athletes according to gender*



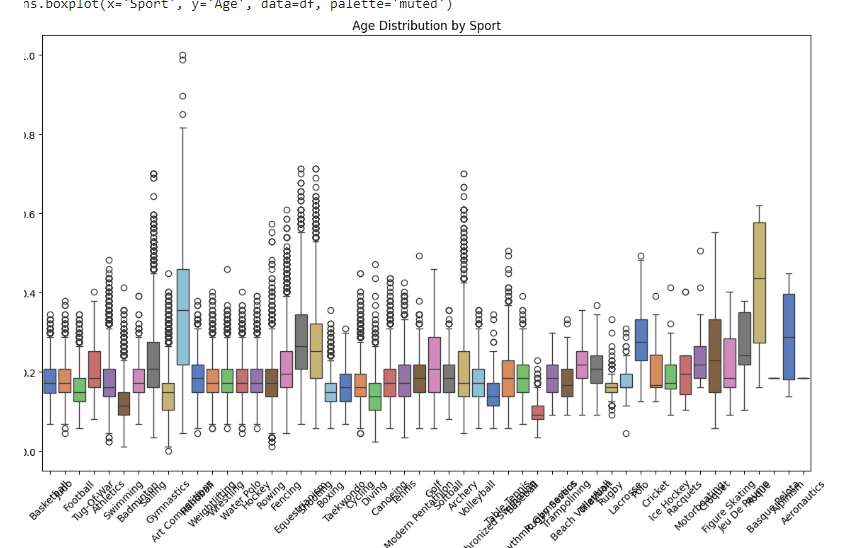
*Figure 4: Bar plot of the Top 10 teams.*

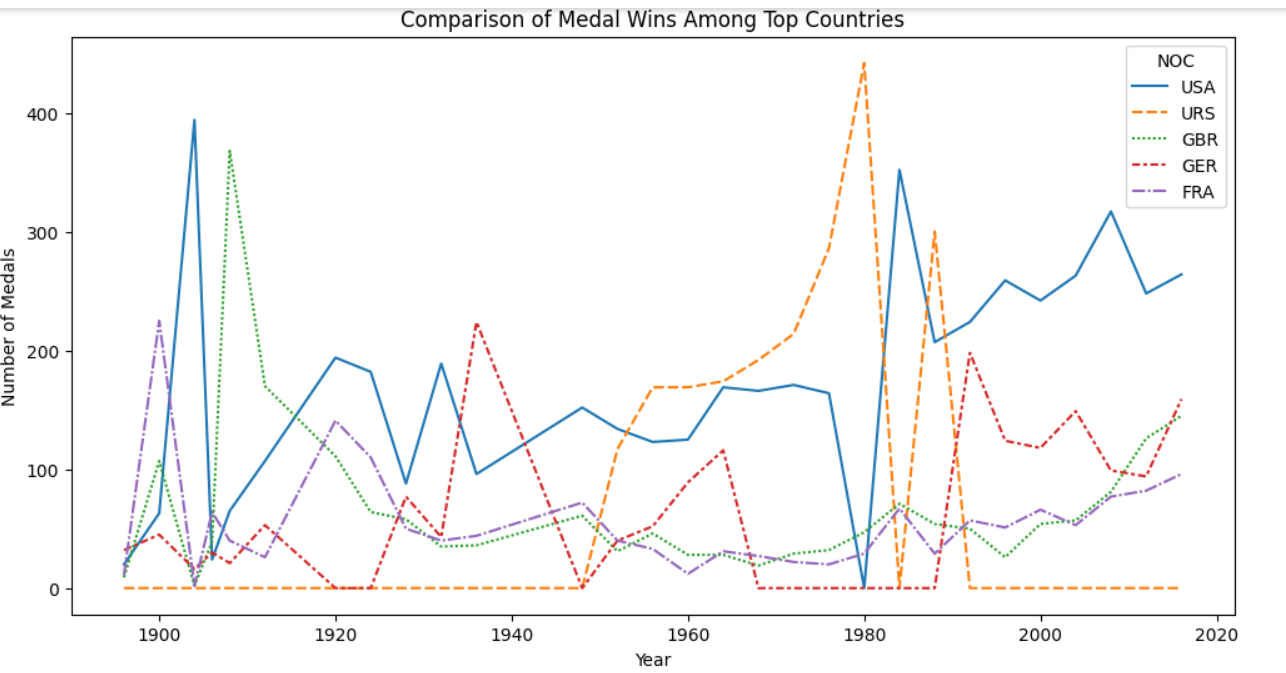


*Figure 5: Top 10 Sports from different countries over the number of participants*



*Figure 6: Histogram showing Age distribution of athletes.*



*Figure 7: Box plot of the distribution of age by sports*

*Figure 8: Line chart shows the comparison of medal wins among the top Countries*

**2.4 Model Building**

**2.4.1 Train/Test split:**

One important aspect of all machine learning models is to determine their accuracy. Now, in order to determine their accuracy, one can train the model using the given dataset and then predict the response values for the same dataset using that model and hence, find the accuracy of the model. A better option is to split our data into two parts: first one for training our machine learning model, and second one for testing our model.

* Split the dataset into two pieces: a training set and a testing set.
* Train the model on the training set.
* Test the model on the testing set, and evaluate how well our model did.

**Advantages of train/test split:**

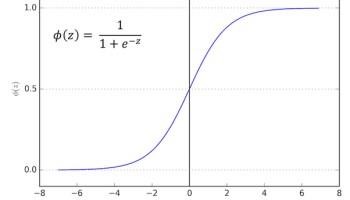
* Model can be trained and tested on different data than the one used for training.
* Response values are known for the test dataset, hence predictions can be evaluated
* Testing accuracy is a better estimate than training accuracy of out-of-sample performance.

Machine learning consists of algorithms that can automate analytical model building. Using algorithms that iteratively learn from data, machine learning models facilitate computers to find hidden insights from Big Data without being explicitly programmed where to look.

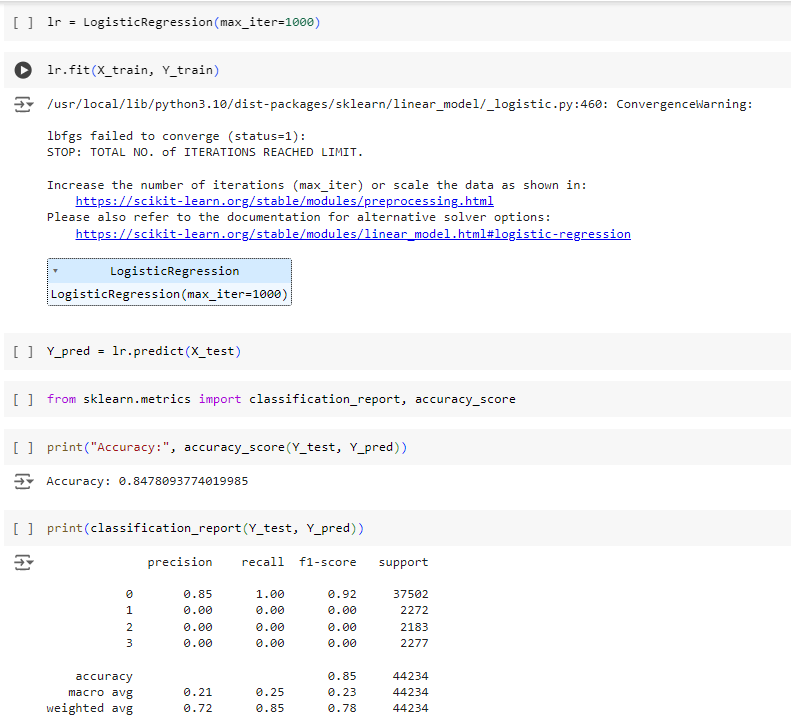
We have used the following these algorithms to build predictive model.

**2.4.2 Logistic Regression:**

Logistic regression is the appropriate regression analysis to conduct when the dependent variable is dichotomous (binary). Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.



*Figure 9: Example of logistic regression function*



*Figure 10: The classification report of logistic regression model*

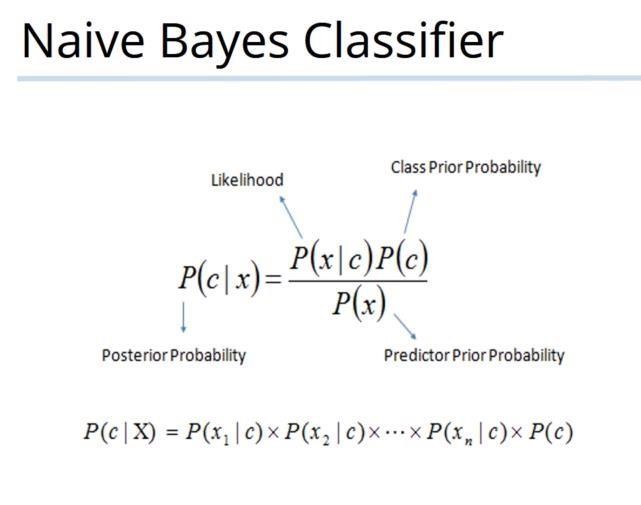
**2.4.3 Naïve Bayes:**

Naive Bayes is a kind of classifier which uses the Bayes Theorem. It predicts membership probabilities for each class such as the probability that given record or data point belongs to a particular class. The class with the highest probability is considered as the most likely class.

*Figure*

*11:*

*Naïve Bayes formula*





*Figure 12: The classification report of Naïve bayes model*

**2.4.4 K nearest neighbor:**

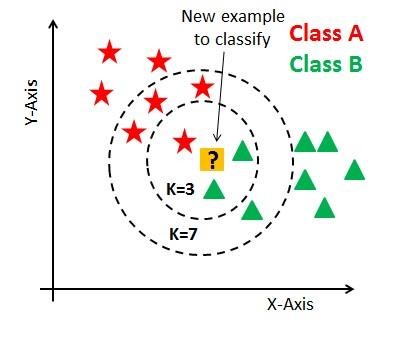
K Nearest Neighbour Algorithm. K nearest neighbour algorithm is very simple. It works based on minimum distance from the query instance to the training samples to determine the K-nearest neighbors. The data for KNN algorithm consist of several multivariate attributes name that will be used to classify.

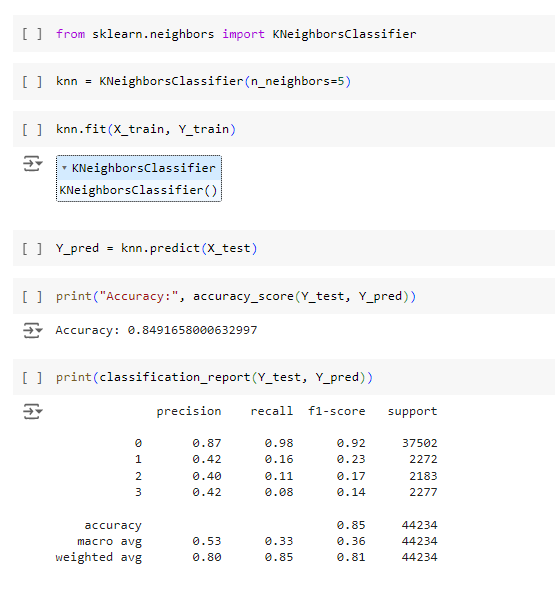
K in KNN is a parameter that refers to the number of nearest neighbors to include in the majority of the voting process.

*Figure*

*13:*

*Example of KNN.*

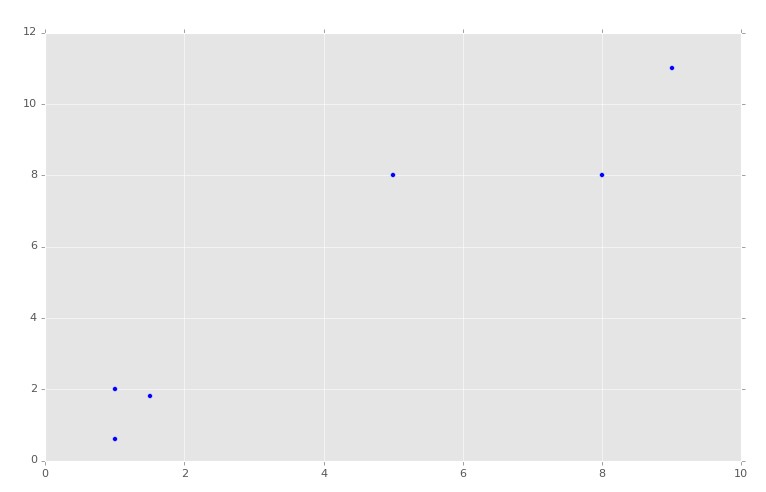


**

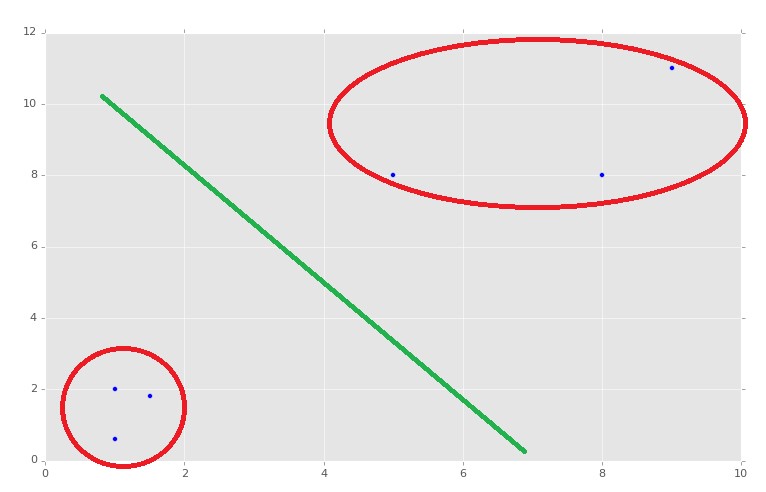
*Figure 14: The classification report of KNN model*

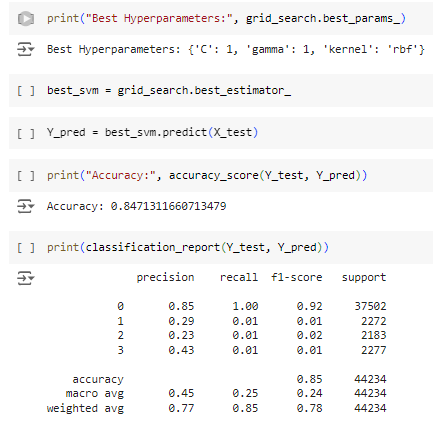
**2.4.5 Linear SVC**

The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is. This makes this specific algorithm rather suitable for our uses, though you can use this for many situations.



*Figure 15: The data points for present in the dataset*



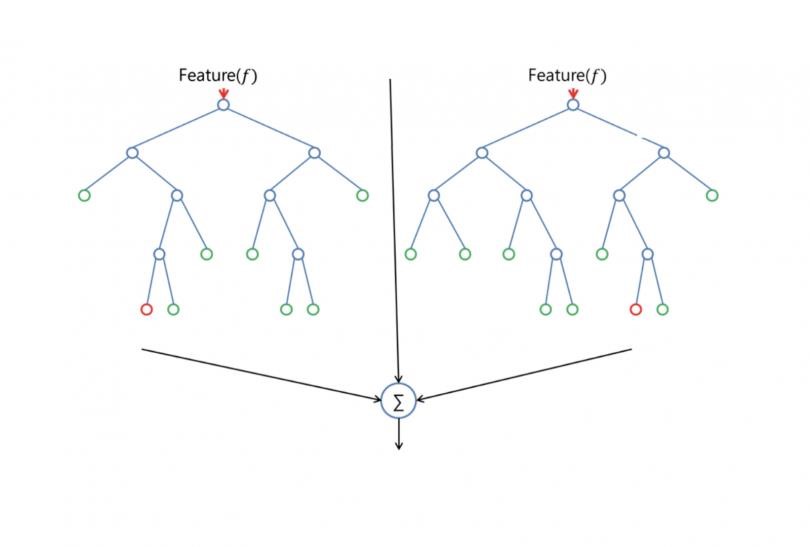
*Figure 16: SVC gives us the decision boundary between the distinct data points*

*Figure 17: The classification report of Linear SVC model*

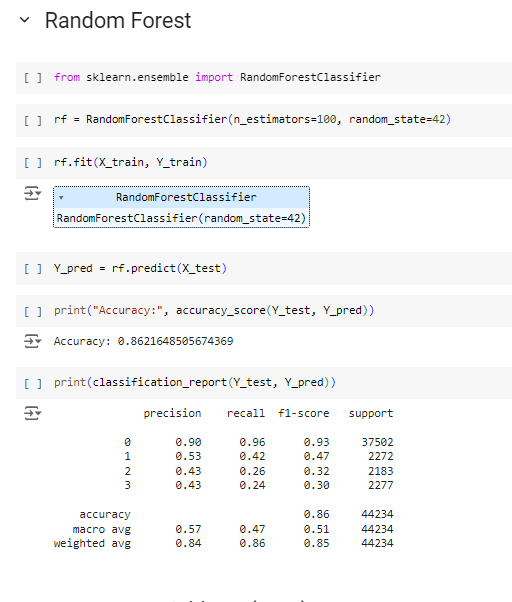
**2.4.6 Random Forest Model:**

Random forest is a [supervised learning algorithm.](https://builtin.com/data-science/supervised-learning-python) The "forest" it builds, is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.



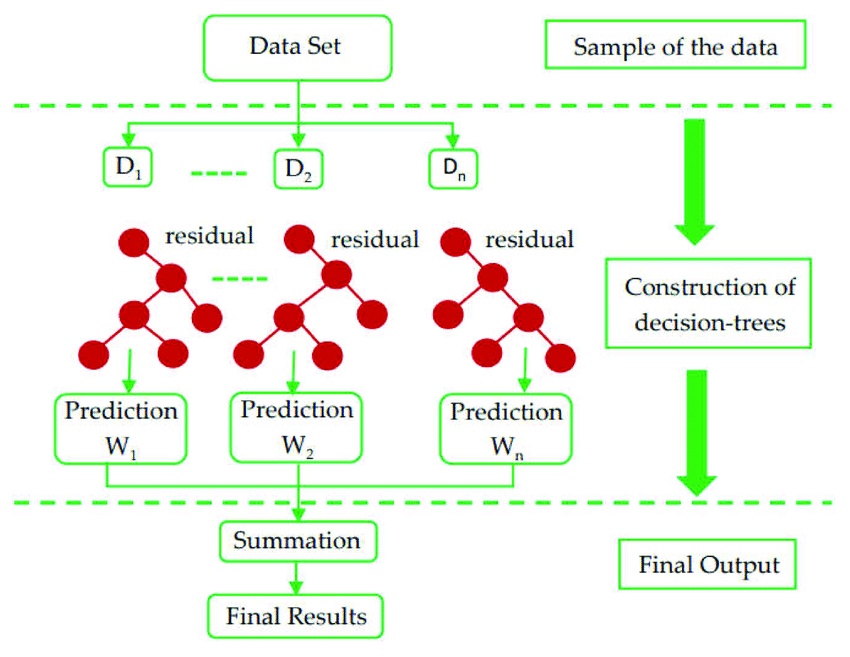
*Figure 18: Shows the working of the random forest model*



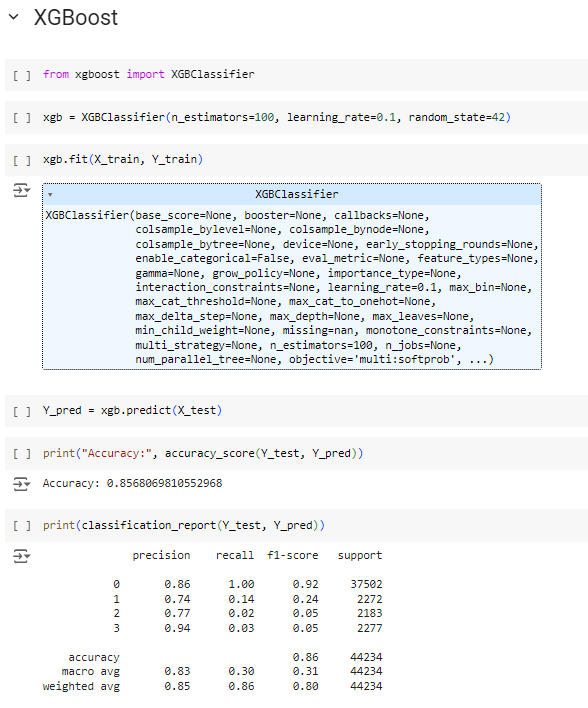
*Figure 19: The classification report of random forest model*

**2.4.7 XGBoost:**

[XGBoost](https://xgboost.ai/) is a decision-tree-based ensemble Machine Learning algorithm that uses a [gradient boosting](https://en.wikipedia.org/wiki/Gradient_boosting) framework. In prediction problems involving unstructured data (images, text, etc.) artificial neural networks tend to outperform all other algorithms or frameworks. However, when it comes to small-to-medium structured/tabular data, decision tree based algorithms are considered best-in-class right now.



*Figure 20 :XG boost (Extreme gradient Boosting)*

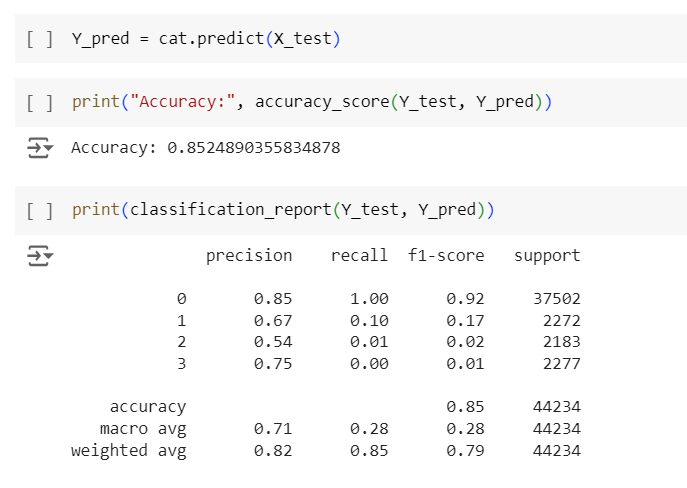


*Figure 21: The classification report of the XGBoost model*

**2.4.8 CatBoost:**

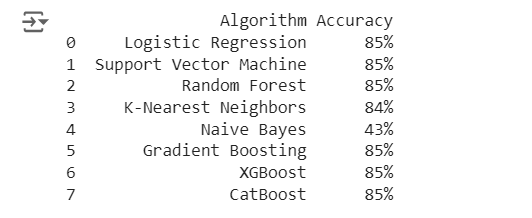
CatBoost or Categorical Boosting is an open-source boosting library developed by Yandex. It is designed for use on problems like regression and classification having a very large number of independent features.

Catboost is a variant of gradient boosting that can handle both categorical and numerical features. It does not require any feature encodings techniques like One-Hot Encoder  to convert categorical features into numerical features. It also uses an algorithm called symmetric weighted quantile sketch (SWQS) which automatically handles the missing values in the dataset to reduce overfitting and improve the overall performance of the dataset.

****

*Figure 22: Catboost classification report*

**Comparison between the algorithm:**



*Figure 23: It represent the comparison of all algorithm performed on the data set*

**CatBoost (Categorical Boosting) is giving us the highest accuracy.**

CatBoost is a machine learning algorithm well-suited for handling categorical features. It is designed to provide high performance and ease of use.

**Key Features of CatBoost:**

* Handling Categorical Data
* Robust to Overfitting
* Fast Training
* Great Out-of-the-Box Performance
* Support for Imbalanced Data
* Built-in Cross-Validation

**2.4.9 ARIMA**

**Forecasting medal prediction (Time Series Analysis):**

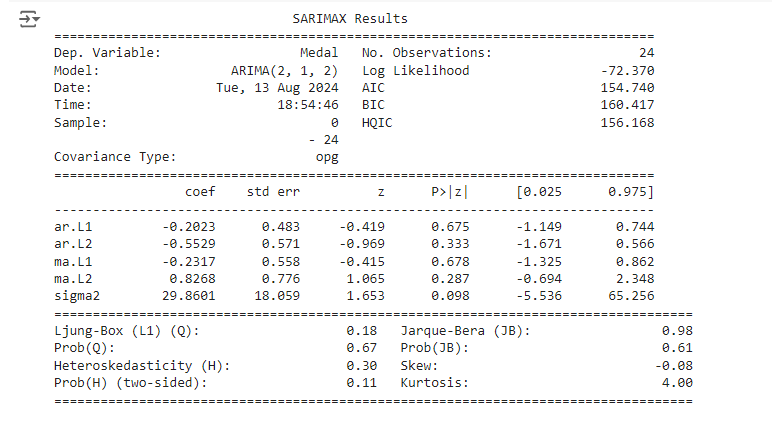
The ARIMA model, short for Auto Regressive Integrated Moving Average, is a widely used approach for time series forecasting. It combines three components:

1. Auto Regressive (AR): This part models the relationship between an observation and a certain number of its previous values (lags). The parameter p indicates the number of lag terms used.
2. Integrated (I): This step involves differencing the time series data to make it stationary, meaning it removes trends or seasonality. The parameter d indicates the number of differencing steps needed.
3. Moving Average (MA): This component models the relationship between an observation and a residual error from previous observations. The parameter q represents the number of lagged forecast errors included.

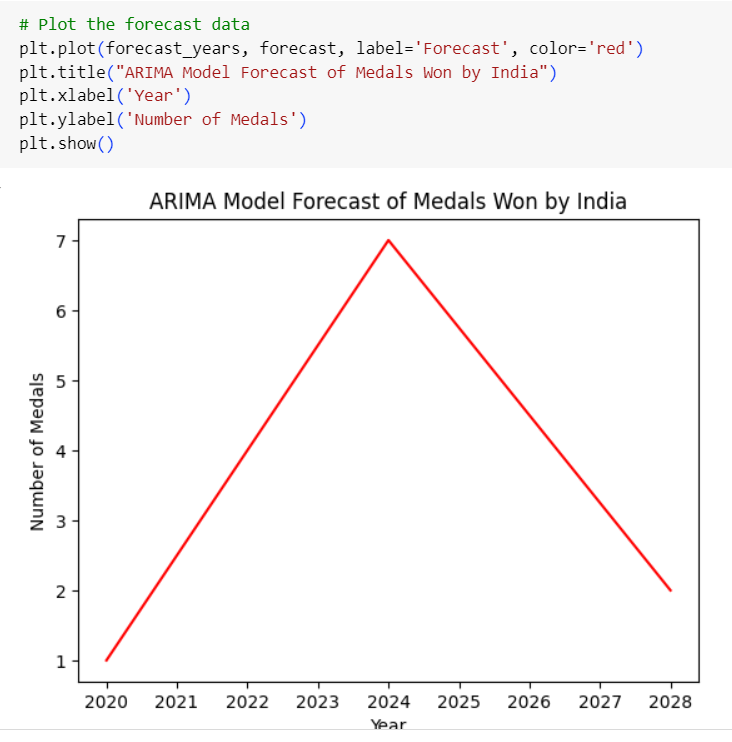
The ARIMA model is specified as ARIMA (p, d, q), where:

* p is the number of lag observations.
* d is the degree of differencing.
* q is the size of the moving average window.

ARIMA is useful for forecasting time series data by capturing patterns like trends and seasonality. It requires data preprocessing to ensure stationarity and careful selection of parameters for accurate modelling.



*Figure 24: Shows result of ARIMA model*

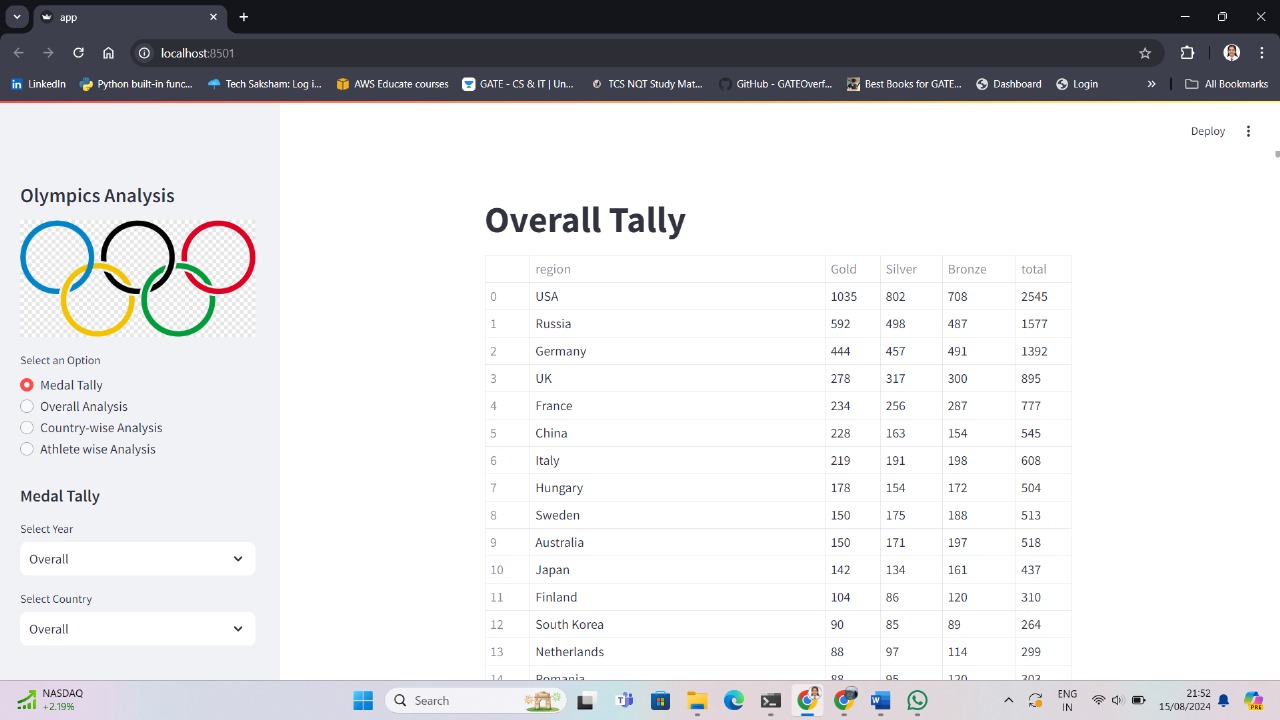


*Figure 25: Showing the Forecasting of Olympics*

**3. User Interface**

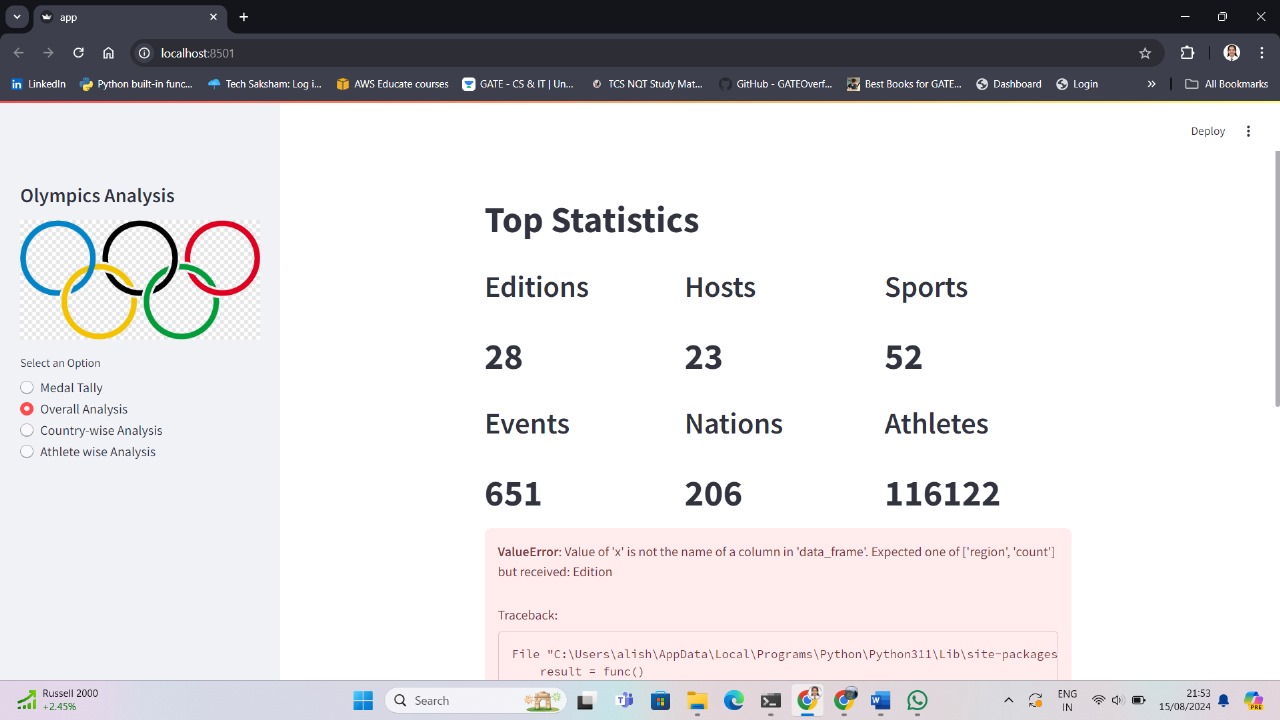
After Training the models and finding out the best model to be Streamlit we can go ahead with building the user interface for our models. This will give us a clean and simple way of accessing our models and make the predictions on our questions.

Streamlit is an open-source Python library used to create interactive web applications specifically for data science and machine learning projects. It allows you to build custom data-driven apps with minimal coding effort. Here are a few screenshots of the web app prediction models.

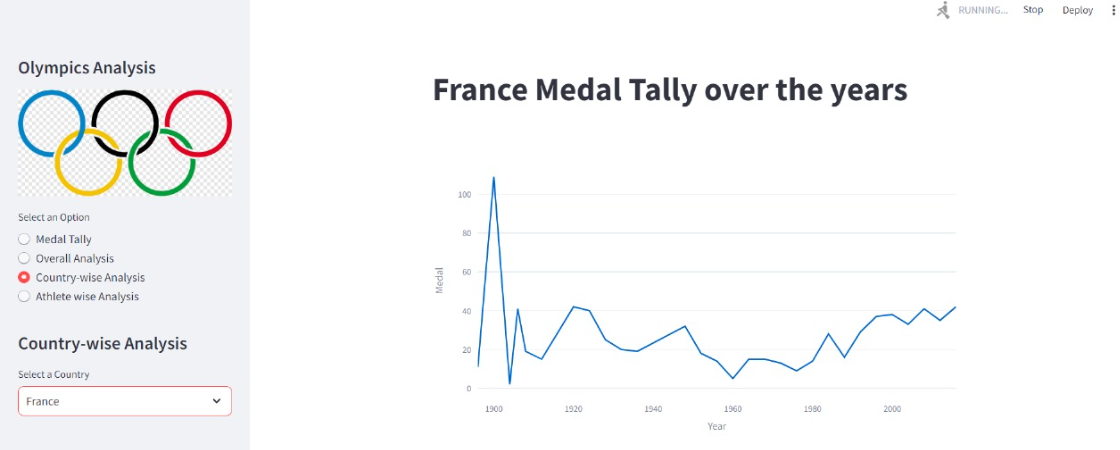


*Figure 33: The web application where the user can do medal tally*

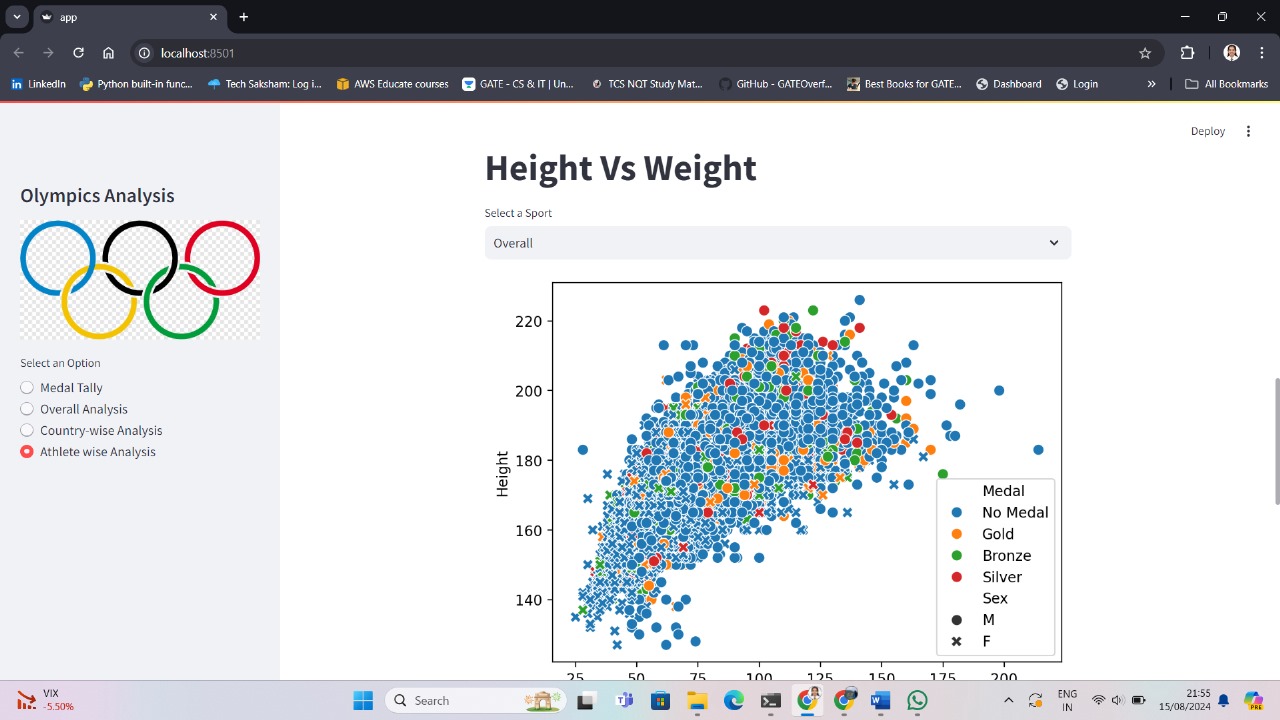
*by selecting specific Year and Country*



*Figure 34: The web application with Overall Analysis*



*Figure 35: This gives us the Country-wise analysis for a particular country.*



*Figure 36: Athlete wise analysis of height vs Weight for a particular sport*

**4. Requirements Specification**

## **4.1 Hardware Requirement:**

* 500 GB hard drive (Minimum requirement)
* 8 GB RAM (Minimum requirement)
* PC x64-bit CPU

## **4.2 Software Requirement:**

* Windows/Mac/Linux
* Google Colab
* Google Drive
* Python-3.9.1
* VS Code/Anaconda/Spyder
* Python Extension for VS Code
* Libraries:
  + Numpy 1.18.2
  + Pandas 1.2.1
  + Matplotlib 3.3.3
  + Scikit-learn 0.24.1
  + Streamlit 1.1.2
* Any Modern Web Browser like Google Chrome

**5. Conclusion**

**Historical Analysis**:

* Identified consistent dominance by countries like the USA, Russia, and China.
* Built a webapp using Streamlit to easily use the model to classify different questions.

**Predictive Modelling**:

* Developed machine learning models to predict future medal counts based on socioeconomic factors.
* Achieved reasonable accuracy with strong correlations between GDP, previous performance, and future success.

**Challenges**:

* Faced challenges in predicting due to the multifactorial nature of Olympic success and unpredictable events.
* Difficulties in quantifying sudden changes in sports infrastructure and emerging talents.

**Key Insights**:

* Economic strength and historical performance are critical indicators of future Olympic success.
* The need for more dynamic factors in prediction models to account for real-time changes.

**6. Future Scope**

**Incorporate Athlete-Level Data**:

* Integrate detailed data on individual athletes, including their performance history, training progress, and injury status, to improve the accuracy of medal predictions.

**Real-Time Data Integration**:

* Include real-time updates on countries’ investments in sports infrastructure, government policies, and emerging sports programs to enhance the predictive models.

**Advanced Machine Learning Techniques**:

* Experiment with more sophisticated algorithms, such as ensemble models, neural networks, and deep learning, to capture complex patterns and interactions in the data.

**Expand to Winter Olympics and Paralympics**:

* Extend the analysis to include the Winter Olympics and Paralympics, offering a more comprehensive view of global sports performance and medal trends.

**Incorporate External Factors**:

* Factor in the impact of external events (e.g., pandemics, political conflicts) that could influence Olympic outcomes, making the predictions more robust.

# **7. References**

* 120 years of Olympic history: athletes and results
* <https://www.kaggle.com/datasets/heesoo37/120-years-of-olympic-history-athletes-and-results>
* <https://en.wikipedia.org/wiki/Summer_Olympic_Games>
* https://numpy.org/doc/
* https://pandas.pydata.org/docs/
* https://docs.streamlit.io/
* https://scikit-learn.org/