METEORITE LANDINGS PREDICTION

Abstract

This project focuses on predicting meteorite landings ('Fell' or 'Found') using a Random Forest Classifier on a curated Meteor Dataset. The methodology involved data cleaning, feature selection, model building, and evaluation using metrics like the Confusion Matrix to ensure predictive accuracy. A dynamic Power BI dashboard was developed to visualize global meteorite distributions, temporal trends, and model predictions. Key findings indicated that features such as mass, latitude, longitude, and year significantly influenced meteorite classification. The project demonstrates the effectiveness of combining machine learning and data visualization to provide actionable insights, benefiting scientific research, risk management, and educational initiatives in space sciences.

Table of Contents

CHAPTER – I
INTRODUCTION

1.1 Scope of Analysis

This project focuses on analyzing and predicting meteorite landing patterns using key features such as year, mass, latitude, and longitude. The objective is to understand the distribution of meteorites and develop a predictive model to classify them as either "Fell" or "Found."

The initial phase involves data preprocessing, where missing values will be handled, and irrelevant columns will be removed to ensure data quality. Exploratory Data Analysis (EDA) will be performed to identify trends, such as the geographic distribution of meteorites, variations in mass, and the relationship between meteorite classifications and other features.

To build a predictive model, a **Random Forest classifier** will be implemented, leveraging its ability to handle non-linear relationships and feature importance analysis. The model's performance will be evaluated using accuracy metrics, a confusion matrix, and other relevant evaluation techniques to ensure reliability. Additionally, feature importance analysis will provide insights into which attributes most significantly impact meteorite classification.

Once trained, the model will be used to predict classifications for new or unclassified meteorite data, and the enhanced dataset—including predictions—will be saved for further research or application. This analysis not only helps in understanding historical meteorite trends but also contributes to the development of robust predictive models that can aid future studies in planetary science and meteorite impact analysis.

1.2 Approach of Analysis

The analysis begins with loading the dataset and performing data preprocessing to ensure data quality and consistency. This includes handling missing values, removing redundant or irrelevant columns, and selecting key features such as fall type, year, mass, latitude, and longitude. The cleaned dataset is then prepared for machine learning by splitting it into training (80%) and testing (20%) sets to ensure robust model evaluation.

A **Random Forest classifier** is chosen as the predictive model due to its ability to handle complex patterns and provide feature importance insights. The model is trained on the selected features to classify meteorites as either "Fell" or "Found." To assess its effectiveness, various evaluation metrics such as accuracy, precision, recall, and a confusion matrix are used. Additionally, feature importance analysis helps determine which attributes play a significant role in predicting meteorite classifications.

Once the model is trained and validated, predictions are generated for the dataset. A new column containing these predicted classifications is added to the original dataset, which is then reorganized for better clarity. The enhanced dataset is saved as a **CSV file** for further research or applications.

Beyond model development, the project explores **spatial and temporal trends** in meteorite landings to gain deeper insights into their distribution patterns. The findings contribute to understanding meteorite behavior and provide a foundation for developing predictive tools that can aid in future meteorite studies and impact assessments.

CHAPTER – II

GATHERING DATA

<u>Meteorite Landings</u>

Gathering Data

- Gathering data is the process of collecting relevant information from various sources to be used for analysis, research, or decision-making. In the context of data analysis and machine learning, this step involves identifying, acquiring, and preparing datasets that contain the necessary features for analysis.

Load the Packages:

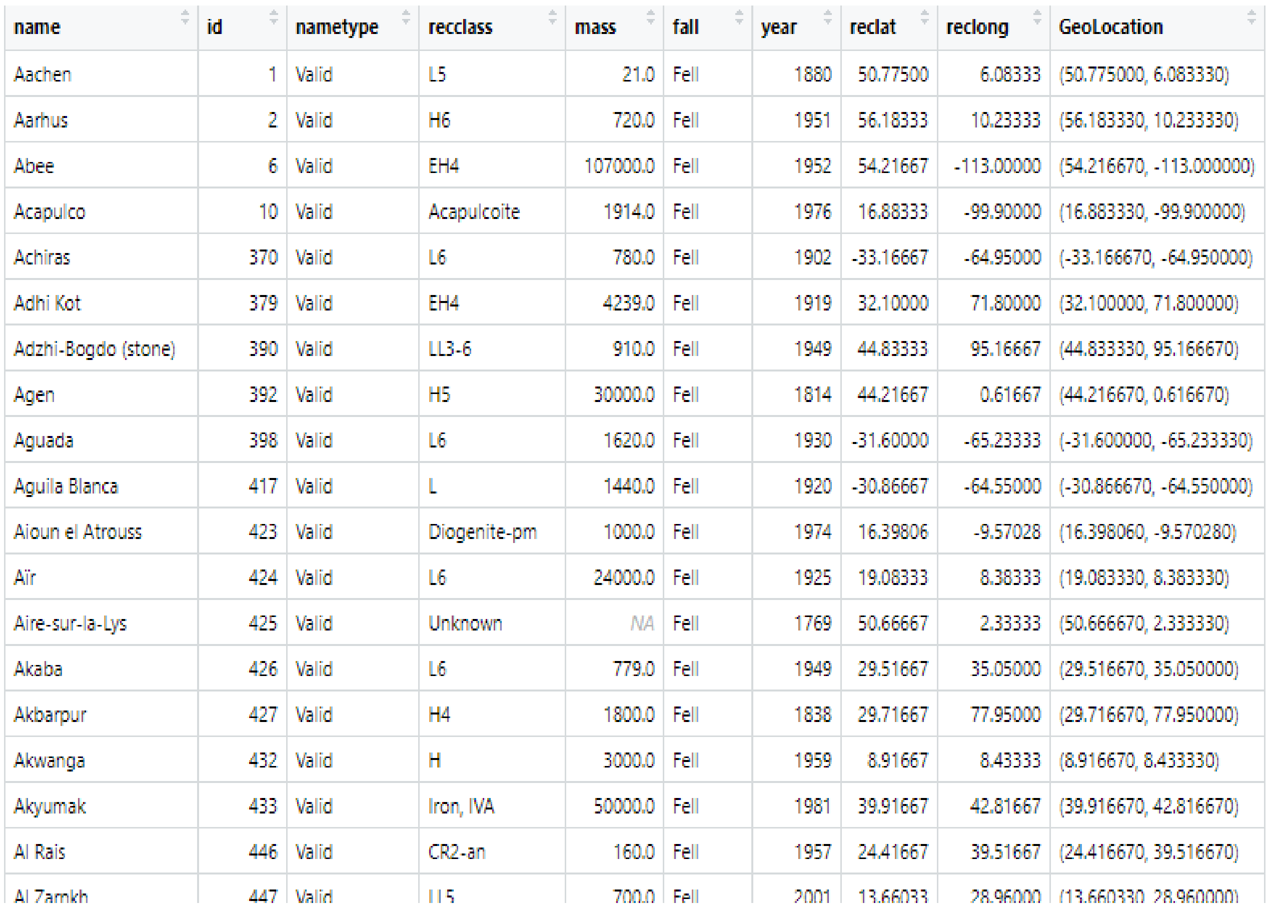
- Loading the packages refers to importing the necessary libraries and modules in a programming environment (such as Python) to perform data analysis, preprocessing, visualization, and machine learning. Packages provide pre-built functions and tools to simplify various tasks.

```
library(tidyverse)
library(caret)
library(dplyr)
library(randomForest)
library(ggplot2)
```

2.1 Data Description :

- Data Description provides detailed information about each attribute (column) in the dataset, explaining its meaning, type, and role in analysis or modelling. It helps in understanding the dataset's structure and how each feature contributes to predictions or insights.

view(data)

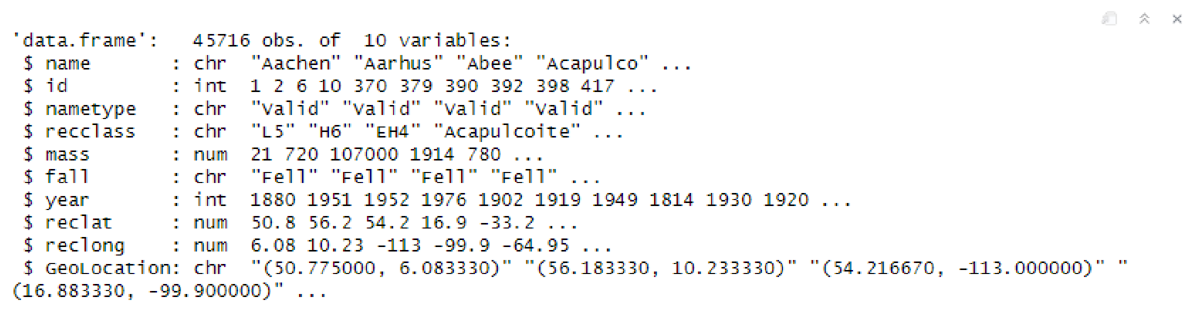


name	id	nametype	recclass	mass	fall	year	reclat	reclong	GeoLocation
Aachen	1	Valid	L5	21.0	Fell	1880	50.77500	6.08333	(50.775000, 6.083330)
Aarhus	2	Valid	H6	720.0	Fell	1951	56.18333	10.23333	(56.183330, 10.233330)
Abee	6	Valid	EH4	107000.0	Fell	1952	54.21667	-113.00000	(54.216670, -113.000000)
Acapulco	10	Valid	Acapulcoite	1914.0	Fell	1976	16.88333	-99.90000	(16.883330, -99.900000)
Achiras	370	Valid	L6	780.0	Fell	1902	-33.16667	-64.95000	(-33.166670, -64.950000)
Adhi Kot	379	Valid	EH4	4239.0	Fell	1919	32.10000	71.80000	(32.100000, 71.800000)
Adzhi-Bogdo (stone)	390	Valid	LL3-6	910.0	Fell	1949	44.83333	95.16667	(44.833330, 95.166670)
Agen	392	Valid	H5	30000.0	Fell	1814	44.21667	0.61667	(44.216670, 0.616670)
Aguada	398	Valid	L6	1620.0	Fell	1930	-31.60000	-65.23333	(-31.600000, -65.233330)
Aguila Blanca	417	Valid	L	1440.0	Fell	1920	-30.86667	-64.55000	(-30.866670, -64.550000)
Aioun el Atrouss	423	Valid	Diogenite-pm	1000.0	Fell	1974	16.39806	-9.57028	(16.398060, -9.570280)
Aïr	424	Valid	L6	24000.0	Fell	1925	19.08333	8.38333	(19.083330, 8.383330)
Aire-sur-la-Lys	425	Valid	Unknown	NA	Fell	1769	50.66667	2.33333	(50.666670, 2.333330)
Akaba	426	Valid	L6	779.0	Fell	1949	29.51667	35.05000	(29.516670, 35.050000)
Akbarpur	427	Valid	H4	1800.0	Fell	1838	29.71667	77.95000	(29.716670, 77.950000)
Akwanga	432	Valid	H	3000.0	Fell	1959	8.91667	8.43333	(8.916670, 8.433330)
Akyumak	433	Valid	Iron, IVA	50000.0	Fell	1981	39.91667	42.81667	(39.916670, 42.816670)
Al Rais	446	Valid	CR2-an	160.0	Fell	1957	24.41667	39.51667	(24.416670, 39.516670)
Al Zarnkh	447	Valid	LL5	700.0	Fell	2001	13.66033	28.96000	(13.660330, 28.960000)

Structure of the Dataset:

- The structure of the dataset refers to the way data is organized in a tabular format, typically consisting of rows (records) and columns (features/attributes). Understanding the dataset structure is essential for performing data analysis, preprocessing, and building predictive models.

str(data)



```
'data.frame':   45716 obs. of  10 variables:
 $ name       : chr  "Aachen" "Aarhus" "Abee" "Acapulco" ...
 $ id         : int  1 2 6 10 370 379 390 392 398 417 ...
 $ nametype   : chr  "Valid" "Valid" "Valid" "Valid" ...
 $ recclass   : chr  "L5" "H6" "EH4" "Acapulcoite" ...
 $ mass       : num  21 720 107000 1914 780 ...
 $ fall       : chr  "Fell" "Fell" "Fell" "Fell" ...
 $ year       : int  1880 1951 1952 1976 1902 1919 1949 1814 1930 1920 ...
 $ reclat     : num  50.8 56.2 54.2 16.9 -33.2 ...
 $ reclong    : num  6.08 10.23 -113 -99.9 -64.95 ...
 $ GeoLocation: chr  "(50.775000, 6.083330)" "(56.183330, 10.233330)" "(54.216670, -113.000000)" "
(16.883330, -99.900000)" ...
```

This dataset contains 45,716 observations rows and 10 variables columns

name (chr): The name of the meteorite.

id (int): A unique identifier for each meteorite.

nametype (chr): Type of name, which is likely to indicate if it's a valid meteorite name.

recclass (chr): The classification of the meteorite, representing its type.

mass (num): The mass of the meteorite in grams.

fall (chr): Indicates whether the meteorite **fell** or was **found**.

year (int): The year the meteorite fell or was found.

reclat (num): The latitude where the meteorite was found.

reclong (num): The longitude where the meteorite was found.

GeoLocation (chr): A string representation of the latitude and longitude in parentheses.

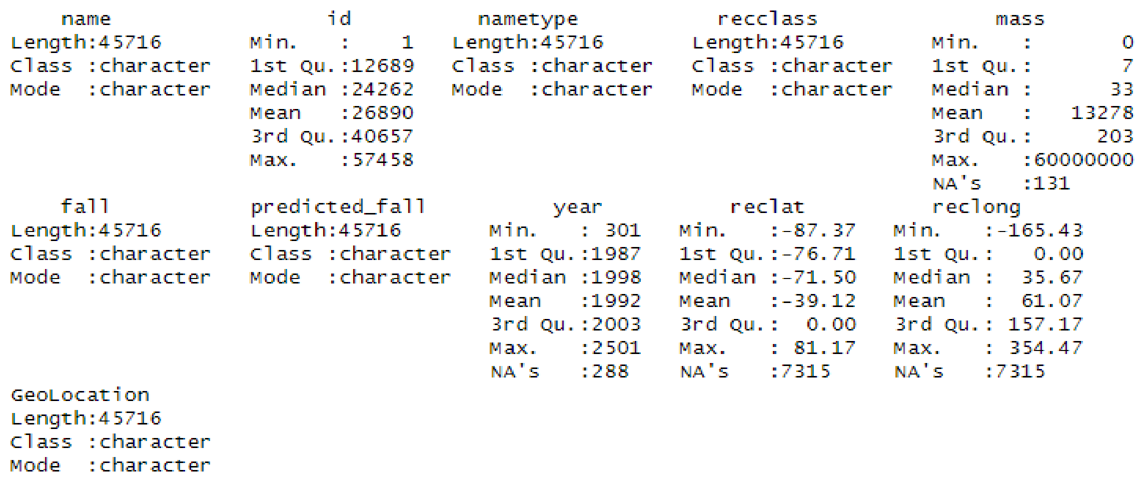
2.2 Understanding Data:

- Understanding data means analyzing raw information to find patterns and insights. It involves exploring data types, checking statistics, cleaning errors, and visualizing trends. This process is essential for decision-making in data science and analytics.

Data Summary

- A Data Summary provides an overview of the dataset, helping to understand its structure, key characteristics, and potential issues such as missing values or outliers. This step is essential before performing data analysis or machine learning, as it gives insights into the quality and distribution of data.

summary(Data)



```
     name                 id             nametype           recclass              mass         
 Length:45716       Min.   :    1   Length:45716       Length:45716       Min.   :       0  
 Class :character   1st Qu.:12689   Class :character   Class :character   1st Qu.:       7  
 Mode  :character   Median :24262   Mode  :character   Mode  :character   Median :      33  
                    Mean   :26890                                         Mean   :   13278  
                    3rd Qu.:40657                                         3rd Qu.:     203  
                    Max.   :57458                                         Max.   :60000000  
                                                                          NA's   :131       
     fall           predicted_fall          year          reclat          reclong       
 Length:45716       Length:45716       Min.   : 301   Min.   :-87.37   Min.   :-165.43  
 Class :character   Class :character   1st Qu.:1987   1st Qu.:-76.71   1st Qu.:   0.00  
 Mode  :character   Mode  :character   Median :1998   Median :-71.50   Median :  35.67  
                                       Mean   :1992   Mean   :-39.12   Mean   :  61.07  
                                       3rd Qu.:2003   3rd Qu.:  0.00   3rd Qu.: 157.17  
                                       Max.   :2501   Max.   : 81.17   Max.   : 354.47  
                                       NA's   :288    NA's   :7315     NA's   :7315     
 GeoLocation       
 Length:45716      
 Class :character  
 Mode  :character  
```

The dataset contains 45,716 entries with details on meteorite landings, including name, ID, classification, mass, fall type, year, and location. The mass varies widely, with extreme values, and the year ranges from 301 to 2501, possibly indicating historical records. Significant missing values exist in mass, year, latitude, and longitude, requiring data cleaning. The dataset also includes GeoLocation, storing coordinates as a string. Proper preprocessing is needed for accurate analysis.

The Dataset contains the following variables:

- **name**: Name of the meteorite.
- **id**: Unique identifier for each entry.
- **Name type**: Type of name (e.g., "Valid").
- **recclass**: Classification of the meteorite.
- **mass**: Mass of the meteorite (in grams).
- **fall**: Whether the meteorite "Fell" or was "Found".
- **year**: Year of the meteorite's fall or discovery.
- **reclat**: Latitude where the meteorite was found.
- **reclong**: Longitude where the meteorite was found.
- **GeoLocation**: Combined latitude and longitude.

Continuous and Categorical Variables:

Continuous

- Mass: The mass of the meteorite (e.g., 21.0, 720.0, etc.).
- Year: The year when the meteorite fell or was discovered.
- Reclat and reclong: Latitude and longitude of the meteorite's location.

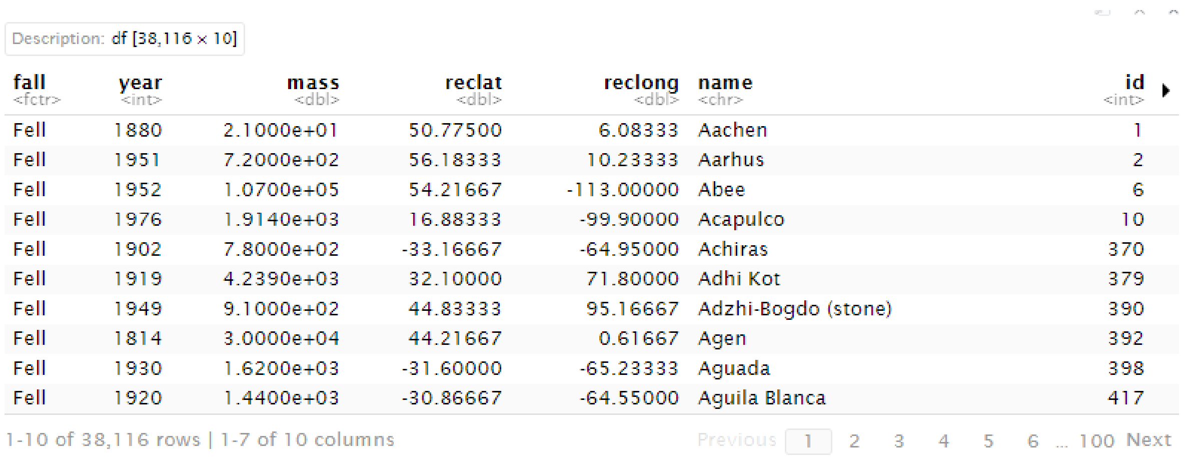
Categorical

- Fall: Whether the meteorite "Fell" or was "Found".
- Recclass: The classification type of the meteorite (e.g., L5, H6).
- Name type: The type of name (e.g., Valid).
- GeoLocation: A location descriptor (e.g., "(50.775, 6.08333)").

Data Cleaning:

- Data Cleaning is the process of identifying and correcting errors, inconsistencies, and missing values in a dataset to improve its quality and reliability. It ensures that the data is accurate, complete, and formatted correctly for analysis or machine learning.

```
data_clean <- data %>%   select(fall, year, mass, reclat,
reclong,name,id,GeoLocation,nametype,recclass) %>%  filter(!is.na(fall) & !is.na(year) &
!is.na(mass) & !is.na(reclat) & !is.na(reclong)) %>% mutate(fall = as.factor(fall))
```



```
Description: df [38,116 × 10]

fall    year   mass        reclat     reclong    name                  id
<fctr>  <int>  <dbl>       <dbl>      <dbl>      <chr>                 <int>
Fell    1880   2.1000e+01  50.77500     6.08333  Aachen                  1
Fell    1951   7.2000e+02  56.18333    10.23333  Aarhus                  2
Fell    1952   1.0700e+05  54.21667  -113.00000  Abee                    6
Fell    1976   1.9140e+03  16.88333   -99.90000  Acapulco               10
Fell    1902   7.8000e+02 -33.16667   -64.95000  Achiras               370
Fell    1919   4.2390e+03  32.10000    71.80000  Adhi Kot              379
Fell    1949   9.1000e+02  44.83333    95.16667  Adzhi-Bogdo (stone)   390
Fell    1814   3.0000e+04  44.21667     0.61667  Agen                  392
Fell    1930   1.6200e+03 -31.60000   -65.23333  Aguada                398
Fell    1920   1.4400e+03 -30.86667   -64.55000  Aguila Blanca         417

1-10 of 38,116 rows | 1-7 of 10 columns          Previous 1 2 3 4 5 6 ... 100 Next
```

Dimension of the Dataset:

- The dimension of a dataset refers to the number of rows (records) and columns (features) it contains. It helps in understanding the dataset's size and structure before performing analysis or modelling.

```
print("Actual Dimension of data")
dim(data)
print("After data cleaning")
dim(data_clean)
```

```
[1] "Actual Dimension of data"
[1] 45716    10
[1] "After data cleaning"
[1] 38116    10
```

Initially, the dataset had 45,716 rows and 10 columns. After cleaning, the number of rows was reduced to 38,116, while the number of columns remained the same. This suggests that 7,600 rows were removed, likely due to missing or invalid data. The cleaning process may have involved removing duplicates, handling missing values, or filtering outliers to improve data quality.

Chapter III

Preparing & Exploring Data

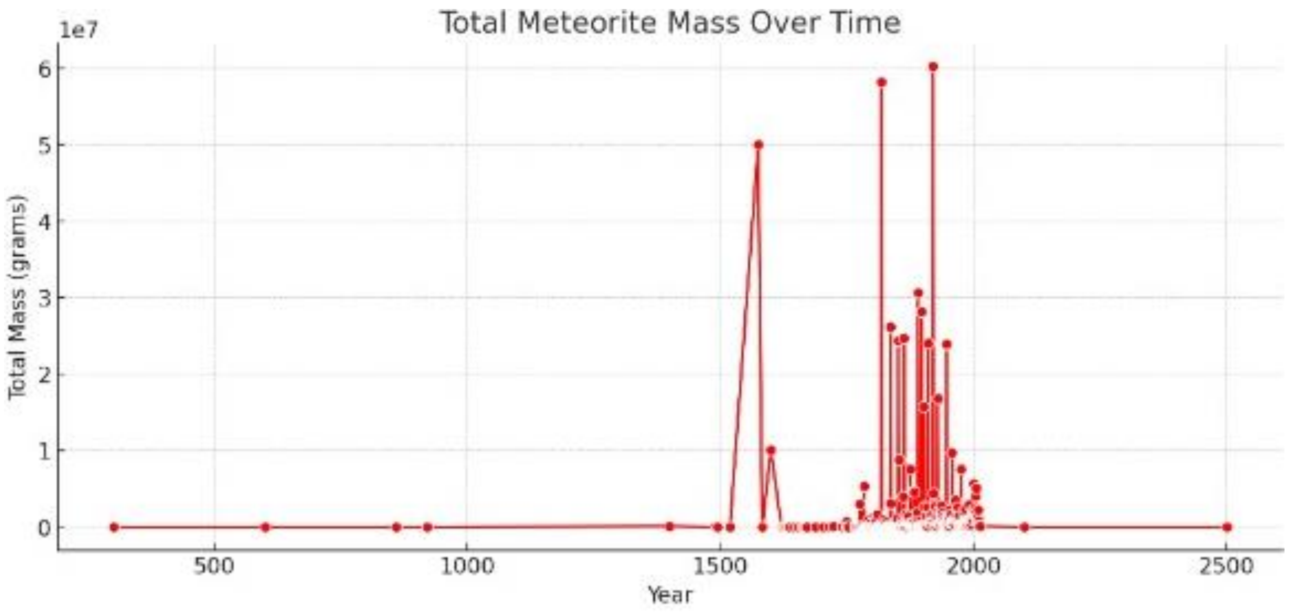
3.1 Data Exploration

- Exploratory Data Analysis (EDA) is the initial step in analyzing a dataset, focusing on summarizing its main characteristics, often with visual methods. It involves investigating the data to discover patterns, detect anomalies, test hypotheses, and check assumptions using statistical graphics and data visualization techniques. EDA helps in understanding the data distribution, relationships between variables, and underlying trends, providing insights that guide further data processing and modelling.

List of Charts :

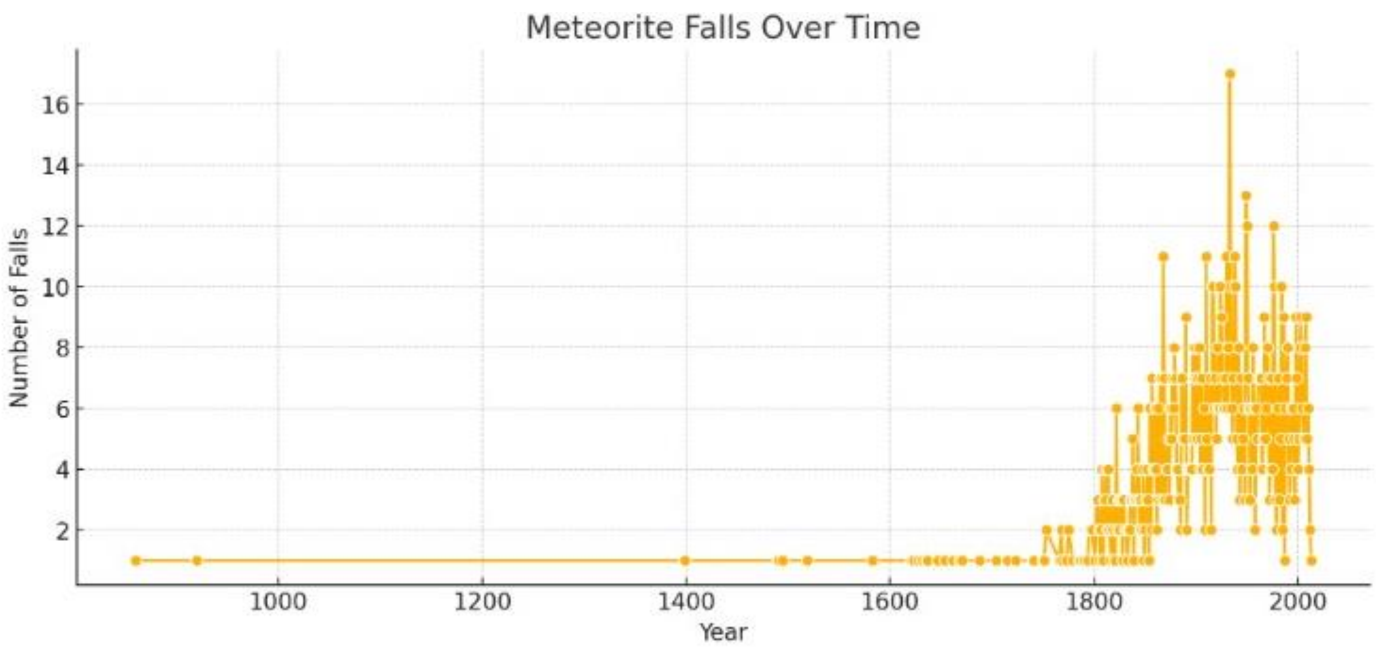
1. Total Meteorite Mass Over Time.
2. Meteorite Falls Over Time.
3. Average Meteorite Mass Per Year.
4. Fall vs. Found Meteorites.

1. **Total Meteorite Mass Over Time (Chart 1)**



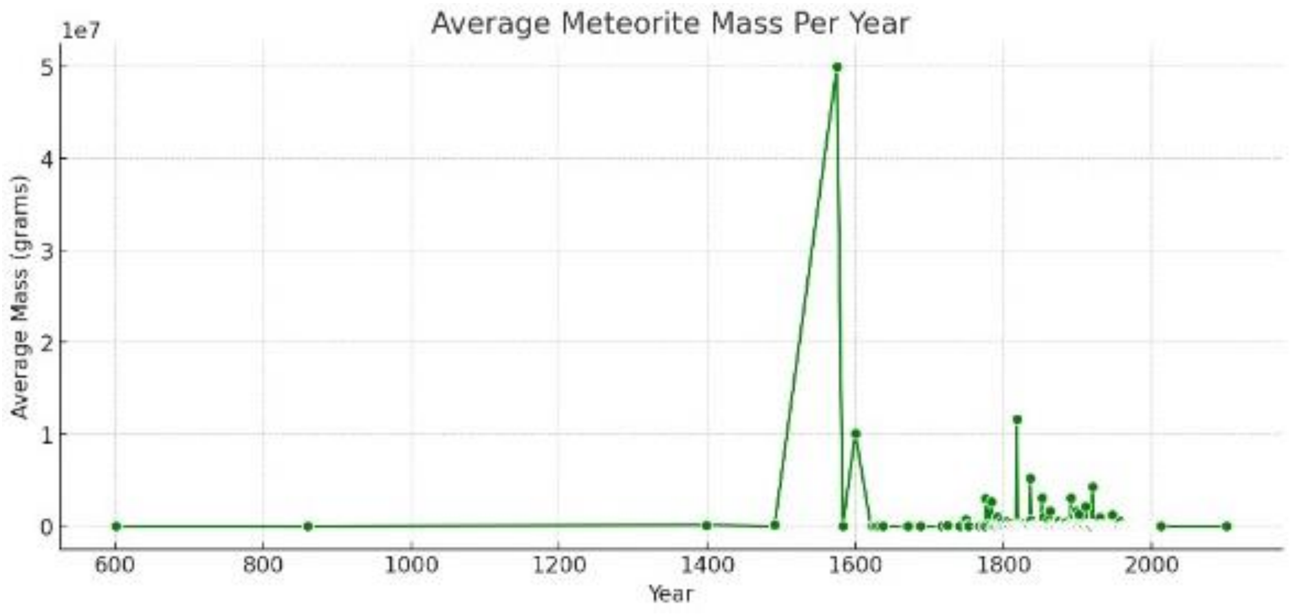
Description: Shows the cumulative mass of meteorites that fell over time, highlighting significant events or periods with high meteorite mass.

2. **Meteorite Falls Over Time (Chart 2)**



Description: Displays the frequency of meteorite falls per year, providing insight into temporal patterns and spikes in meteorite activities.

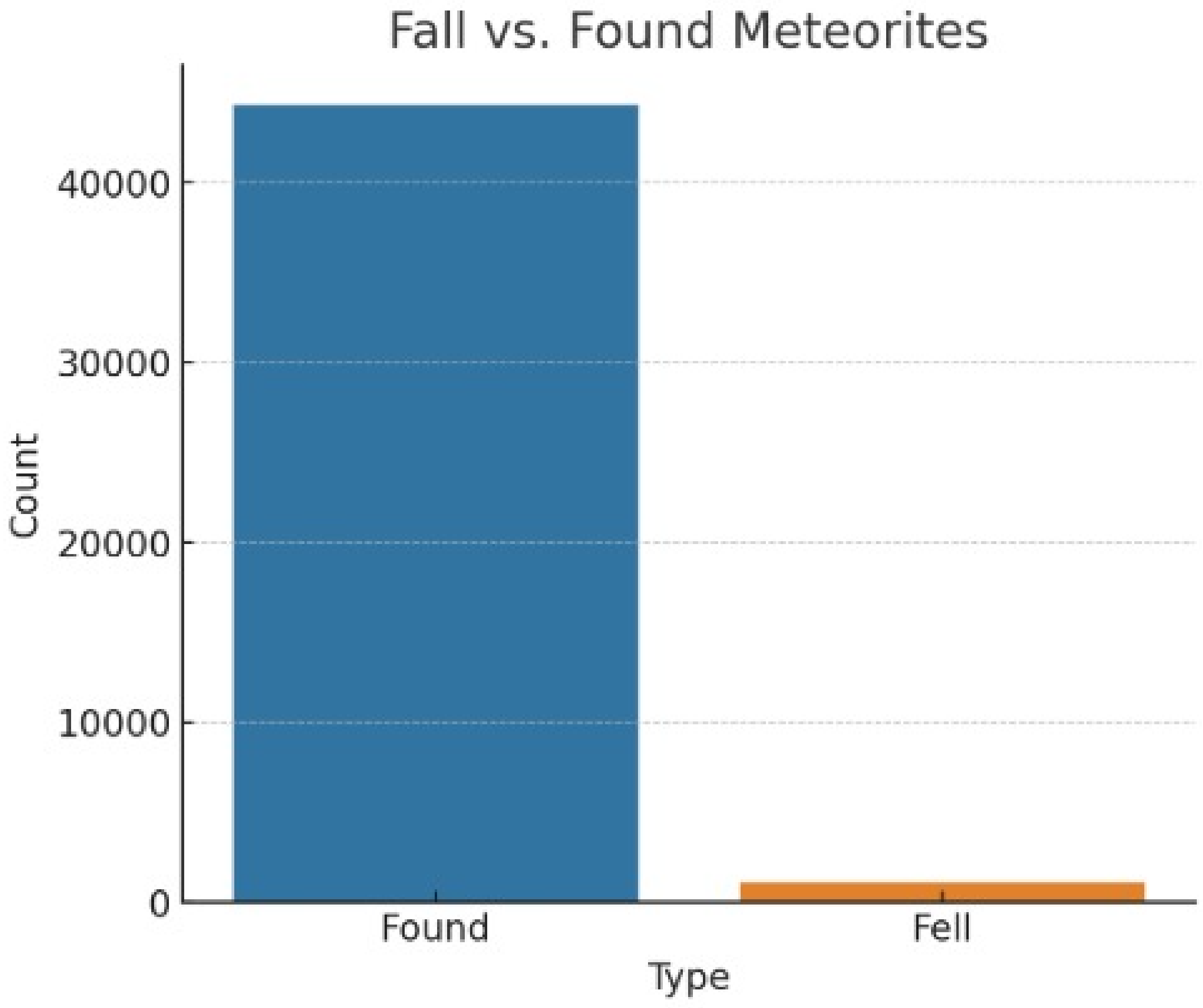
3. **Average Meteorite Mass Per Year (Chart 3)**



Description: Represents the average mass of meteorites each year, identifying years with unusually heavy or light meteorite falls.

4. **Fall vs. Found Meteorites**

.



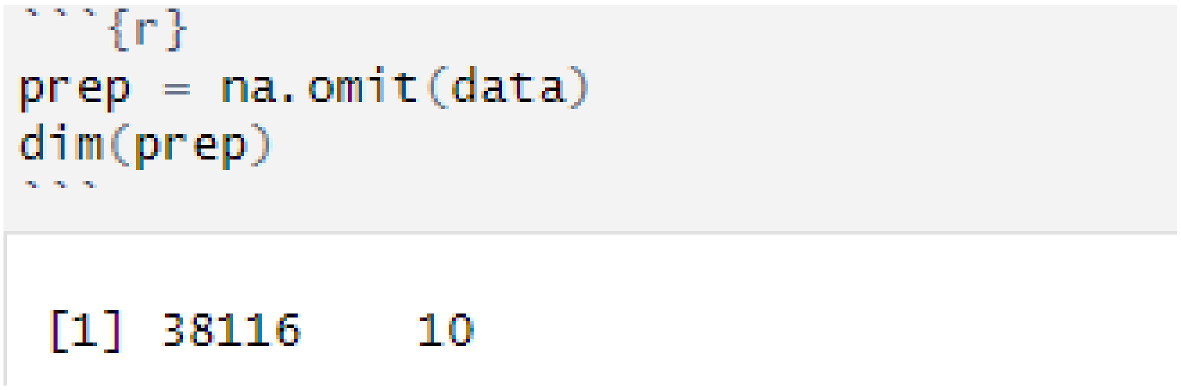
Description*:* A bar chart comparing the count of meteorites that were found vs. those that fell, showing a stark difference in the discovery methods

3.2 Issues in the dataset

- The dataset has several issues affecting its quality. Missing values are present in key columns like mass, year, and geographic coordinates, which can lead to incomplete analysis. Outliers in the mass column (up to 60 million grams) and year column (minimum value: 301) indicate possible errors. Inconsistencies exist in the GeoLocation column, where latitude and longitude are stored as strings instead of separate numerical values. Additionally, duplicate entries may distort the analysis and need to be removed. Addressing these issues through data cleaning, outlier removal, and format corrections is essential for accurate and reliable insights.

3.3 Resolve Issues

- The dataset has missing values in mass, year, and geographic coordinates, along with outliers in mass (60M grams) and year (min: 301). Inconsistencies exist in the GeoLocation format, and possible duplicate entries need removal. Cleaning and preprocessing are essential for accurate analysis.



~~~
```{r}
prep = na.omit(data)
dim(prep)
```

[1] 38116    10
~~~

The image shows an R command using na.omit(data), which removes all rows containing missing values from the dataset. The output dim(prep) indicates that after removing missing data, the dataset now has 38,116 rows and 10 columns, compared to the original 45,716 rows, meaning 7,600 rows were removed due to missing values.

CHAPTER – IV

BUSINESS INTELLIGENCE WITH DASHBOARD

Dashboard Purpose:

- The purpose of this dashboard is to analyze meteor landing data comprehensively, helping to identify trends in meteorite mass, frequency of falls, and classification by composition. The visualizations allow users to explore the distribution of meteorite landings over time and space and understand the dominant meteorite types.

- Additionally, the dashboard provides predictive insights by utilizing machine learning models to classify meteorites based on their features, aiding in future meteorite discovery and research. It also enables users to filter and interact with the data dynamically, making it easier to identify patterns and correlations across different meteorite characteristics.

Dashboard Tpyes:

- Excel, Power BI, and Tableau offer three types of dashboards. **Excel dashboards** are best for basic reporting and small datasets. **Power BI dashboards** provide interactive, real-time data analysis with advanced modelling. **Tableau dashboards** excel in high-quality visualizations and deep analytics for large datasets. Each tool serves different data visualization needs.

1. Excel Dashboard

- Uses Pivot Tables, Charts, and Conditional Formatting for analysis.
- Suitable for basic reporting with limited interactivity.
- Can handle small to medium datasets efficiently.
- Lacks real-time data updates and advanced automation.

2. Power BI Dashboard

- Provides interactive and dynamic visualizations.
- Supports real-time data refresh and cloud-based sharing.
- Enables advanced data modeling using DAX.
- Ideal for business intelligence and performance tracking.

3. Tableau Dashboard

- Specializes in high-quality, AI-powered visualizations.
- Offers drag-and-drop features for easy data exploration.
- Handles large datasets efficiently with fast processing.
- Best for deep analytics and predictive insights.

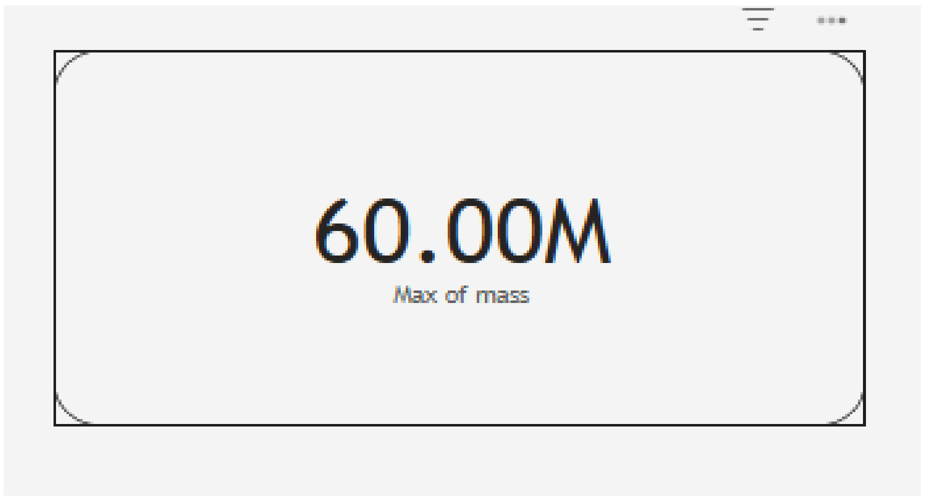
Here, I used Power Bi:

Power BI is a data visualization tool used to create interactive dashboards. It helps in analyzing trends, tracking performance, and making data-driven decisions. With real-time data updates and seamless integration, Power BI simplifies reporting and enhances business insights.

4.1 Dashboards interpretation

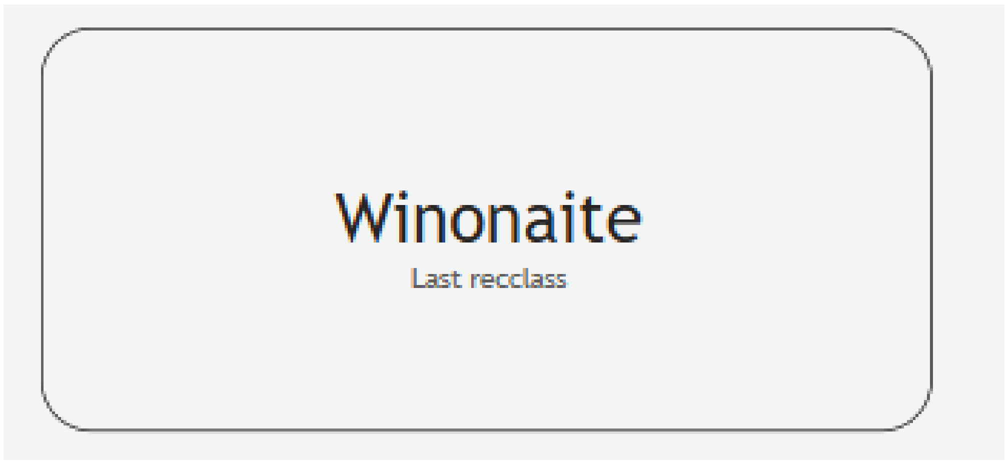
Explanation of Each Chart and Card:

1. **Max of Mass (Card)**



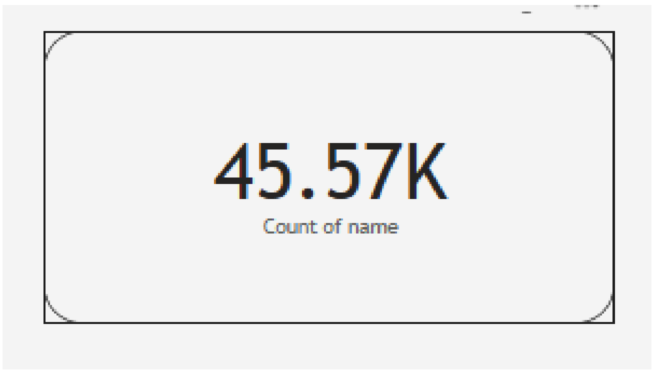
- Displays the maximum meteorite mass recorded, highlighting significant impacts.
- Value: **60.00M** (Likely in grams or metric tons).

2. Last Recclass (Card)



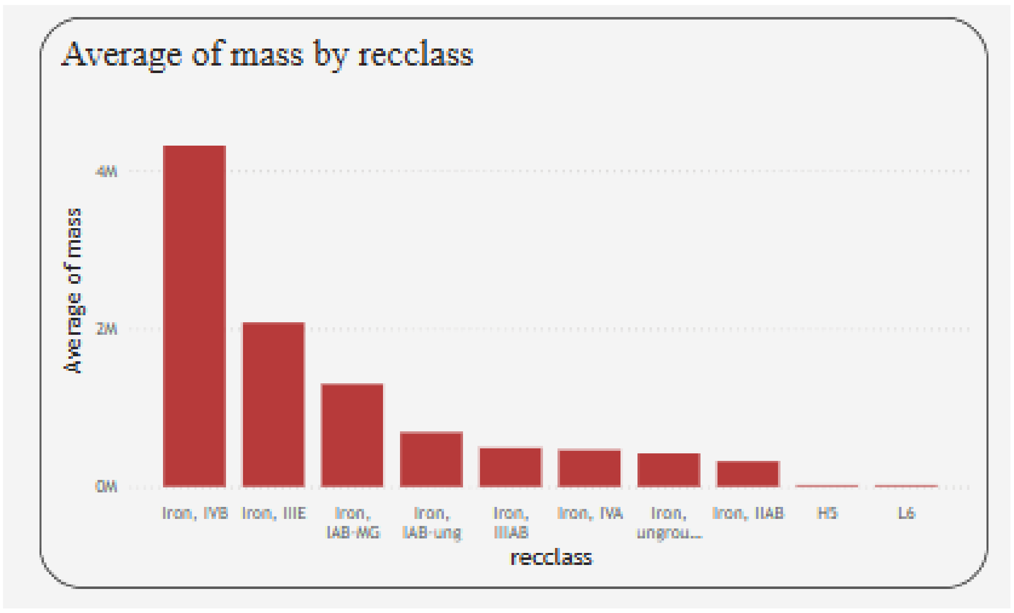
- Shows the most recent classification of meteorite types, with "Winonaite" being the latest.

3. **Count of Name (Card)**



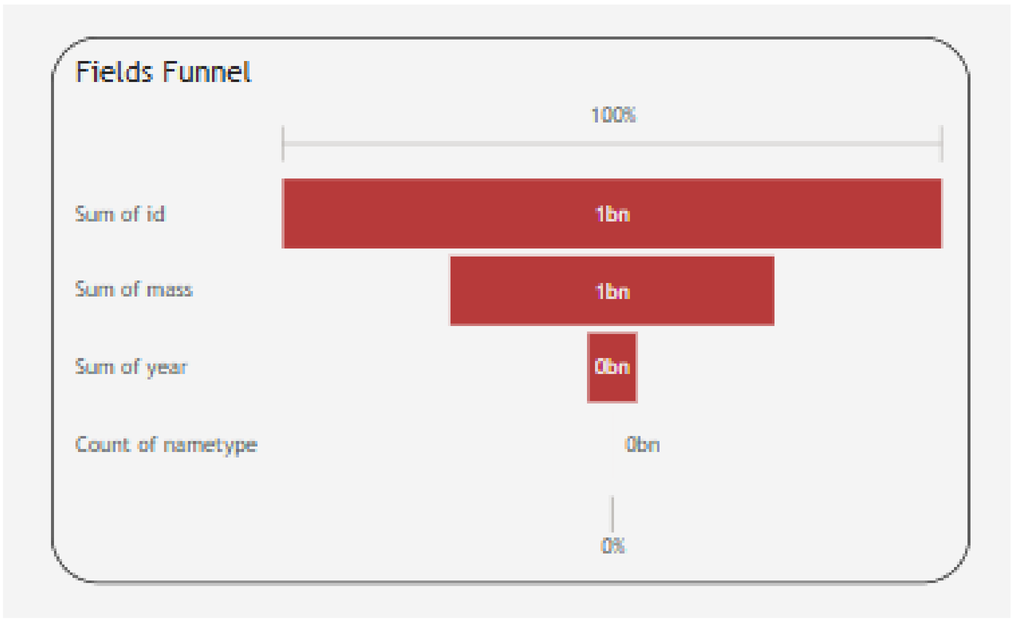
- Displays the total count of distinct meteorite names in the dataset: **45.57K**.

4. **Average of Mass by Recclass (Bar Chart)**



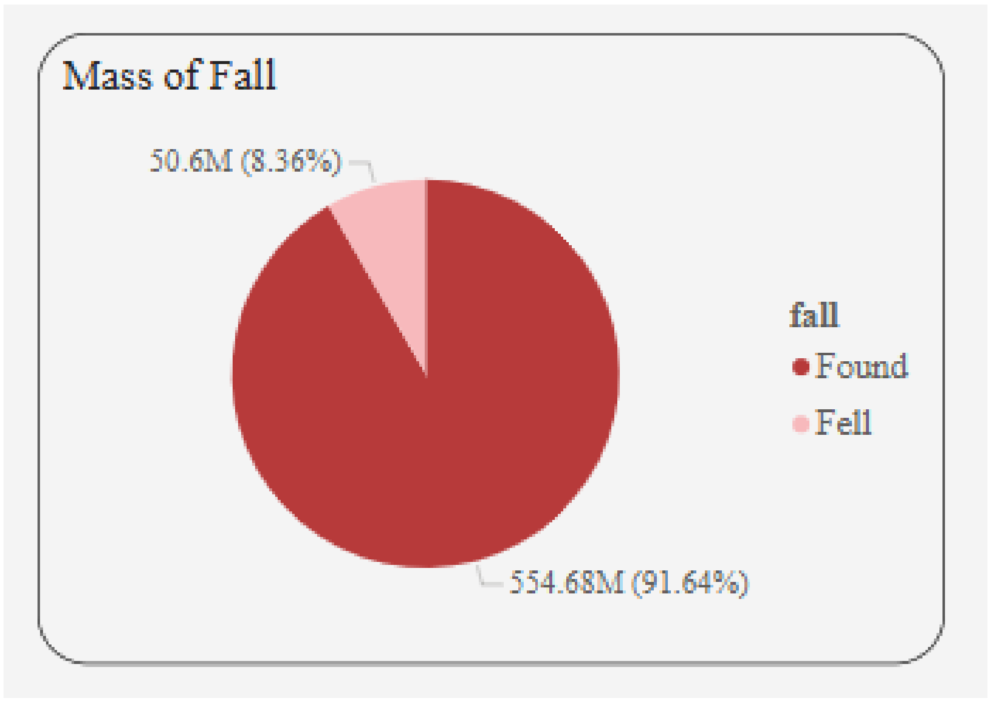
- Analyzes the average mass of meteorites by their classification (e.g., Iron, IIIE, IAB, etc.).
- Highlights the heaviest average mass among the "Iron, IVB" type.

5. **Fields Funnel (Funnel Chart)**



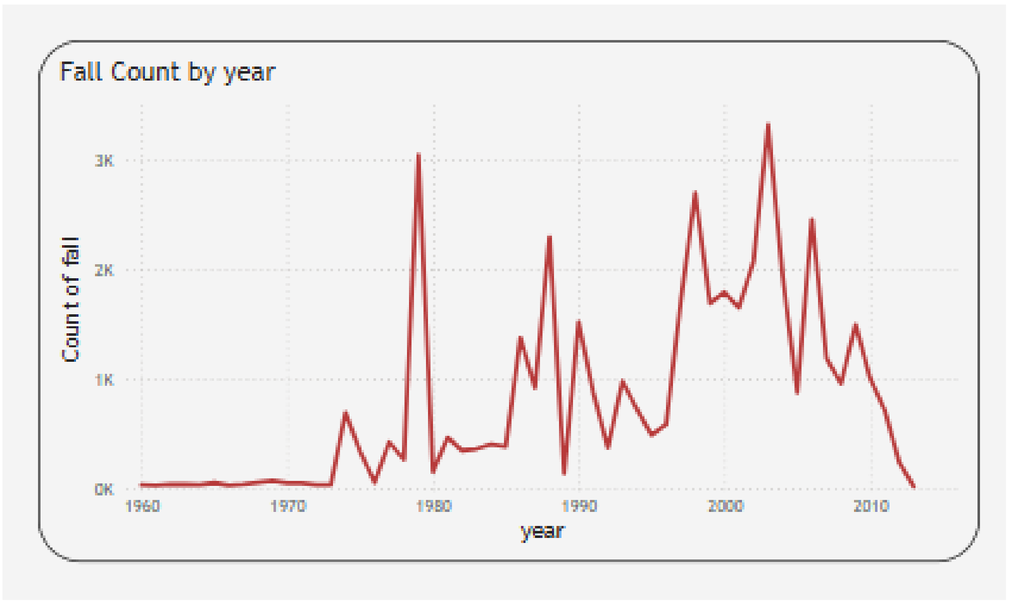
- Visualizes the sum of various metrics such as ID, mass, year, and count of names in a funnel format.
- Emphasizes a high volume of IDs and masses, with a stark drop in other metrics.

6. Mass of Fall (Pie Chart)



- Compares the mass distribution between meteorites that "Fell" (8.36%) and those that were "Found" (91.64%).

7. **Fall Count by Year (Line Chart)**



- Displays the count of meteorite falls over time, with peaks around 1960 and early 2000s.
- Reveals historical patterns in meteorite falls.

Explanation of Slicers

1. **Fall (Slicer)**

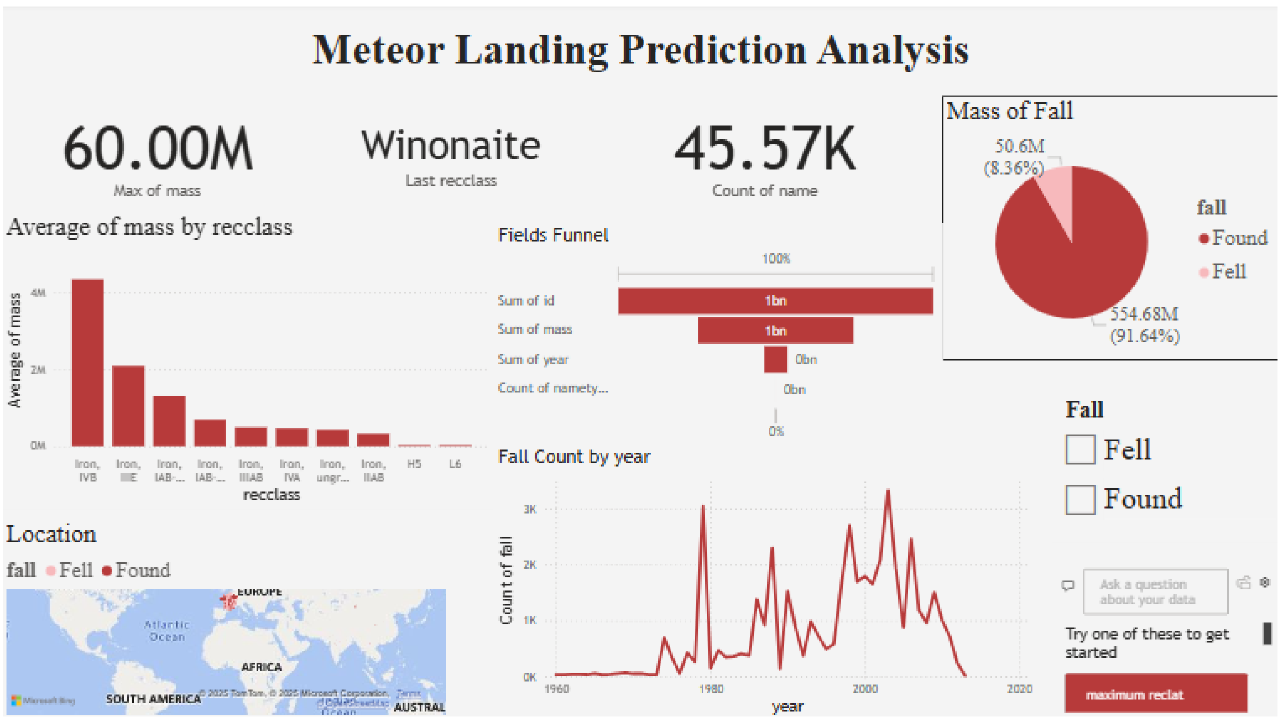
    - Allows users to filter data by meteorite status: "Fell" or "Found."

    - Not connected to certain charts and cards, indicating potential issues in interaction or filter application.

2. Maximum Recclass (Button)

- Likely designed to highlight the maximum class of meteorites, but its specific impact is unclear due to slicer disconnection.

Dashboard:



CHAPTER – V

MODEL BUILDING

Classification Algorithms

Classification algorithms are used when the output variable is categorical, meaning it belongs to a specific group or class. These algorithms learn patterns from labeled data and classify new data points accordingly. They are commonly used in applications like spam detection, sentiment analysis, and medical diagnoses. The output is always discrete, such as "Yes" or "No," "Spam" or "Not Spam," or "Dog" or "Cat." Some of the most popular classification algorithms include Logistic Regression, Decision Trees, Random Forest, Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Naïve Bayes.

Regression Algorithm

Regression algorithms, on the other hand, are used when the output variable is continuous, meaning it can take any numerical value. These algorithms help predict future trends or values based on past data by identifying relationships between input variables. They are widely used in areas like house price prediction, stock market analysis, and weather forecasting. Unlike classification, regression provides continuous numerical outputs such as "25.4," "78.9," or "1500.75." Some of the most commonly used regression algorithms include Linear Regression, Polynomial Regression, Decision Tree Regression, Random Forest Regression, Support Vector Regression (SVR), and Ridge & Lasso Regression.

List Of Classification Algorithms:

- Logistic Regression
- Decision Tree
- Random Forest
- Support Vector Machine (SVM)
- k-Nearest Neighbors (k-NN)
- Naïve Bayes
- XGBoost (Extreme Gradient Boosting)

List Of Regression Algorithms:

- Linear Regression
- Ridge & Lasso Regression
- Decision Tree Regression
- Random Forest Regression
- Gradient Boosting (GBM, XGBoost, LightGBM, CatBoost)

5.1 Algorithm

Since my variable, called **"fell,"** represents a categorical outcome, I chose to use a **Random Forest** classification algorithm. Classification algorithms are designed to categorize data into predefined labels, making them ideal for cases where the target variable consists of distinct classes. **Random Forest**, in particular, is a powerful ensemble learning method that builds multiple decision trees and combines their outputs to improve accuracy and reduce overfitting. By using this algorithm, I can ensure that my model makes robust and reliable predictions, even when dealing with complex and non-linear relationships in the data.

Random Forest Classifier: Theory and Application

1. **Introduction:**

   The Random Forest Classifier is an ensemble learning method that constructs multiple decision trees and merges them to obtain a more accurate and stable prediction. It is widely used in classification and regression tasks due to its high accuracy and ability to handle large datasets with higher dimensionality.

2. **How Random Forest Works:**

- **Data Bootstrapping:**

  Creates random subsets of the original data using sampling with replacement.

- **Random Feature Selection:**

  During tree construction, a random subset of features is selected for splitting nodes.

- **Tree Construction:**

  Individual decision trees are built using the random subsets of data and features.

- **Voting Mechanism:**

  For classification tasks, each tree votes for a class, and the class with the majority votes is selected.

3. **Key Features of Random Forest**

- **Handles Missing Values:**

  Through random sampling.

- **Reduces Overfitting:**

  By averaging results of multiple trees.

- **Feature Importance:**

  Provides a measure of feature relevance.

4. **Applying Random Forest to Our Project Recap**

   Our project involves classifying meteorite falls ('Fell' vs. 'Found') using the meteorite landing dataset from our dashboard.

Steps:

1. **Data Preparation:**

   Load and preprocess the dataset.

2. **Feature Selection:**

   Choose relevant columns like recclass, mass, year, and location features.

3. **Model Training:**

   Implement Random Forest with optimal hyperparameters.

4. **Model Evaluation:**

    Measure accuracy, confusion matrix, and feature importance.

5. **Visualization:**

    Show prediction results and key insights.

Interpretation:

- **Confusion Matrix:**

    Shows True Positives, True Negatives, False Positives, and False Negatives.

- **Classification Report:**

    Precision, Recall, F1-Score
    .

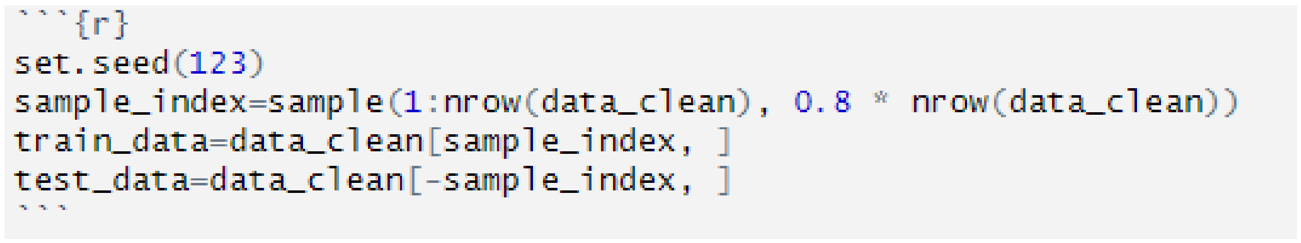
- **Feature Importance:**

    Highlights which features contribute most to the classification.

The Random Forest Classifier provides robust performance in our meteorite classification task. It is particularly useful for our dataset due to its ability to handle non-linear relationships and reduce overfitting through ensemble learning. Further tuning of hyperparameters and feature engineering could improve the model's accuracy.

5.2 Training and Test Dataset

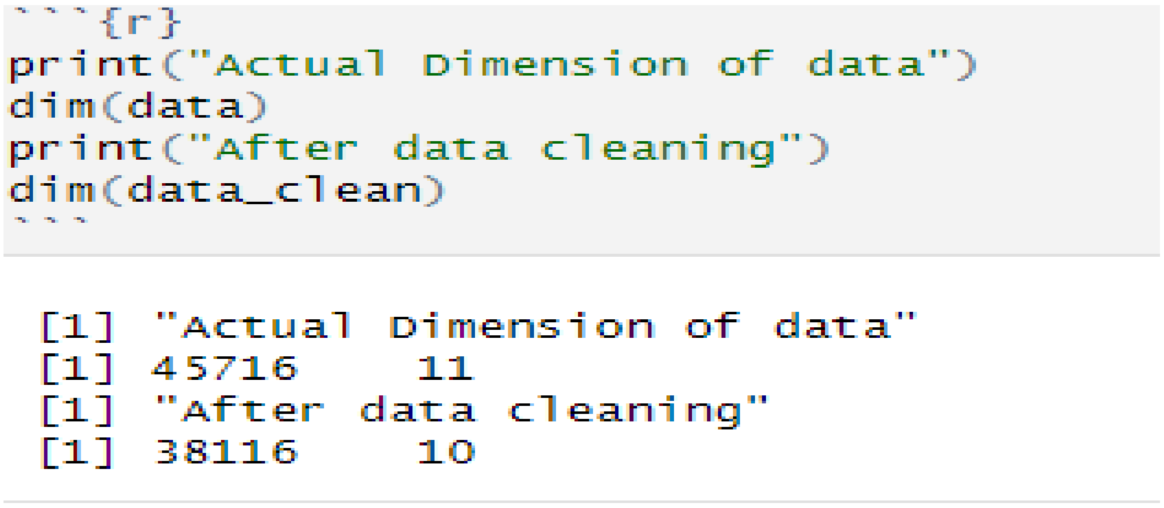
- Split Dataset into 80% for Train data
- Declare balance 20% for Test data
- Set.seed() so it don’t shuffle each time we run
- And fit the dataset with one time sampled



~~~
```{r}
set.seed(123)
sample_index=sample(1:nrow(data_clean), 0.8 * nrow(data_clean))
train_data=data_clean[sample_index, ]
test_data=data_clean[-sample_index, ]
```
~~~

This R code splits the cleaned dataset (data_clean) into training and testing sets. It first sets a random seed (set.seed(123)) to ensure reproducibility. Then, it selects 80% of the data indices randomly using sample(), creating the training dataset (train_data). The remaining 20% of the data is assigned to the testing dataset (test_data). This approach ensures that the model is trained on a majority of the data while keeping a portion separate for evaluation.

Dimension of Train and Test Data



~~~
```{r}
print("Actual Dimension of data")
dim(data)
print("After data cleaning")
dim(data_clean)
```
~~~

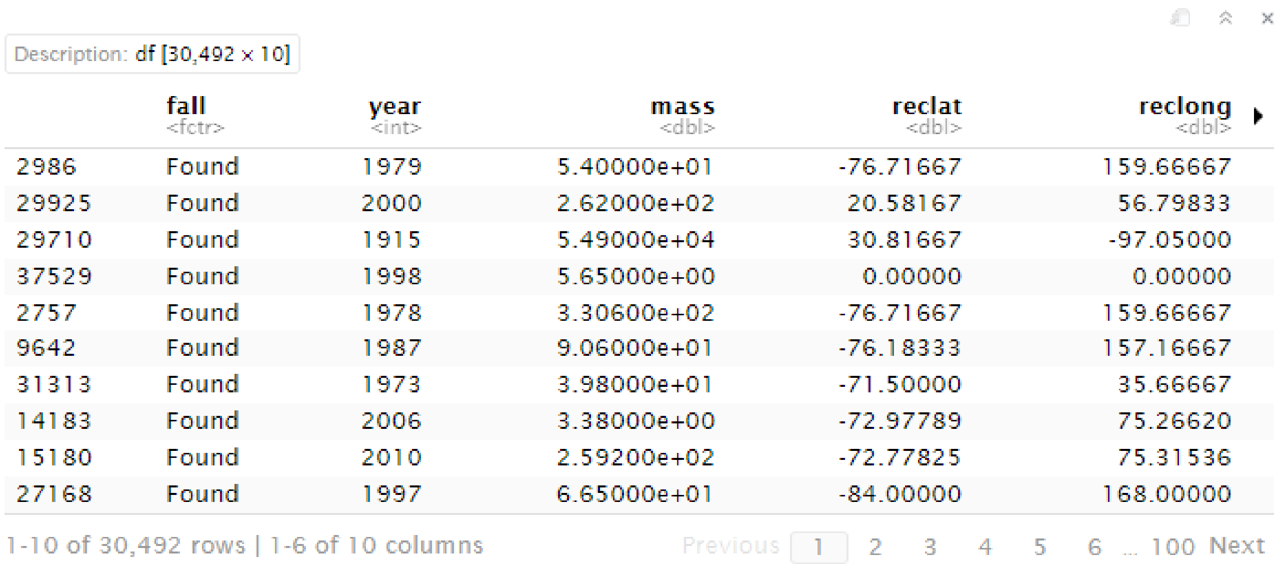
```
[1] "Actual Dimension of data"
[1] 45716     11
[1] "After data cleaning"
[1] 38116     10
```

The dataset (data) has 45,716 rows and 11 columns. After cleaning (data_clean), the dataset is reduced to 38,116 rows and 10 columns. The reduction in rows suggests the removal of missing or inconsistent data, while the reduction in columns indicates that an unnecessary or irrelevant feature was dropped. This step ensures that only high-quality and relevant data is retained for analysis.

Train Data:

Train data is the portion of a dataset used to train a machine learning model. It helps the model learn patterns, relationships, and structures within the data. The model uses this data to adjust its parameters and improve accuracy. Typically, the train data makes up 70-80% of the total dataset, while the remaining portion is used for testing and validation.

train_data



Description: df [30,492 x 10]

	fall <fctr>	year <int>	mass <dbl>	reclat <dbl>	reclong <dbl>
2986	Found	1979	5.40000e+01	-76.71667	159.66667
29925	Found	2000	2.62000e+02	20.58167	56.79833
29710	Found	1915	5.49000e+04	30.81667	-97.05000
37529	Found	1998	5.65000e+00	0.00000	0.00000
2757	Found	1978	3.30600e+02	-76.71667	159.66667
9642	Found	1987	9.06000e+01	-76.18333	157.16667
31313	Found	1973	3.98000e+01	-71.50000	35.66667
14183	Found	2006	3.38000e+00	-72.97789	75.26620
15180	Found	2010	2.59200e+02	-72.77825	75.31536
27168	Found	1997	6.65000e+01	-84.00000	168.00000

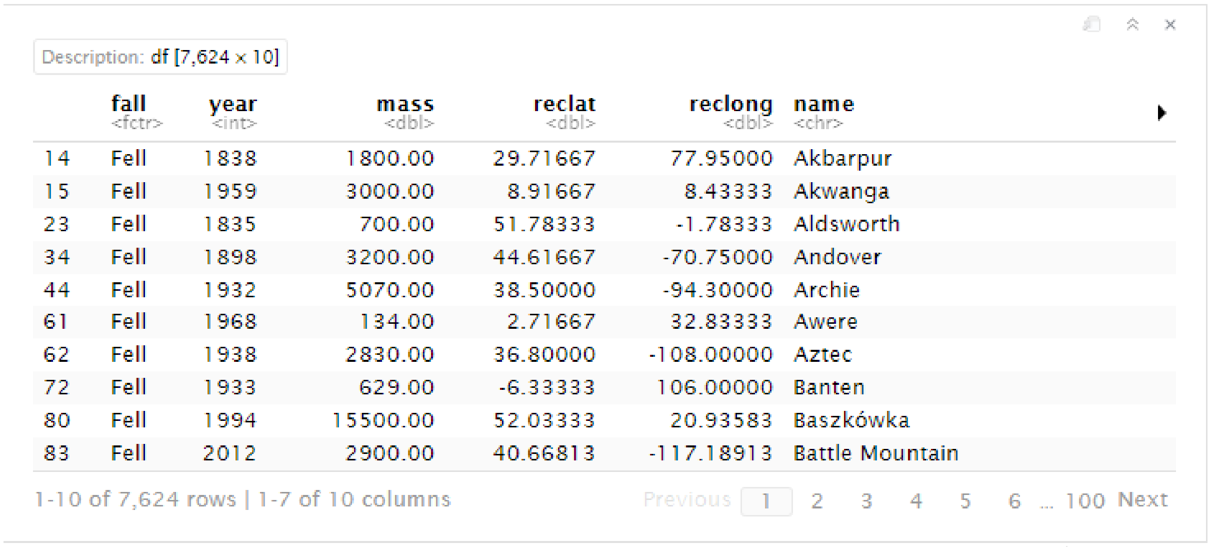
1-10 of 30,492 rows | 1-6 of 10 columns Previous 1 2 3 4 5 6 ... 100 Next

The train_data consists of 30,492 rows and 10 columns, representing the dataset used for model training. It includes features like fall status, year of occurrence, mass of objects, and their geographical coordinates (reclat and reclong). This data helps the model learn patterns and relationships before being tested on a separate validation dataset.

Test Data

Test data is the portion of a dataset used to evaluate the performance of a trained machine learning model. It is separate from the training data and helps measure how well the model generalizes to new, unseen data. The test data typically makes up 20-30% of the total dataset and is used to assess accuracy, detect overfitting, and ensure reliability before deploying the model.

test_data



Description: df [7,624 x 10]

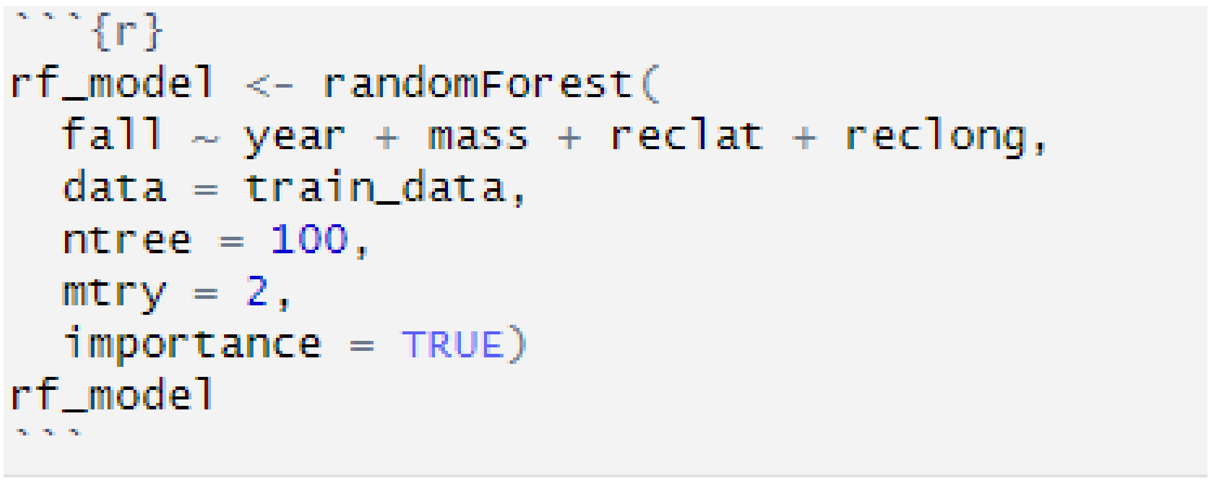
	fall <fctr>	year <int>	mass <dbl>	reclat <dbl>	reclong <dbl>	name <chr>
14	Fell	1838	1800.00	29.71667	77.95000	Akbarpur
15	Fell	1959	3000.00	8.91667	8.43333	Akwanga
23	Fell	1835	700.00	51.78333	-1.78333	Aldsworth
34	Fell	1898	3200.00	44.61667	-70.75000	Andover
44	Fell	1932	5070.00	38.50000	-94.30000	Archie
61	Fell	1968	134.00	2.71667	32.83333	Awere
62	Fell	1938	2830.00	36.80000	-108.00000	Aztec
72	Fell	1933	629.00	-6.33333	106.00000	Banten
80	Fell	1994	15500.00	52.03333	20.93583	Baszkówka
83	Fell	2012	2900.00	40.66813	-117.18913	Battle Mountain

1-10 of 7,624 rows | 1-7 of 10 columns Previous 1 2 3 4 5 6 ... 100 Next

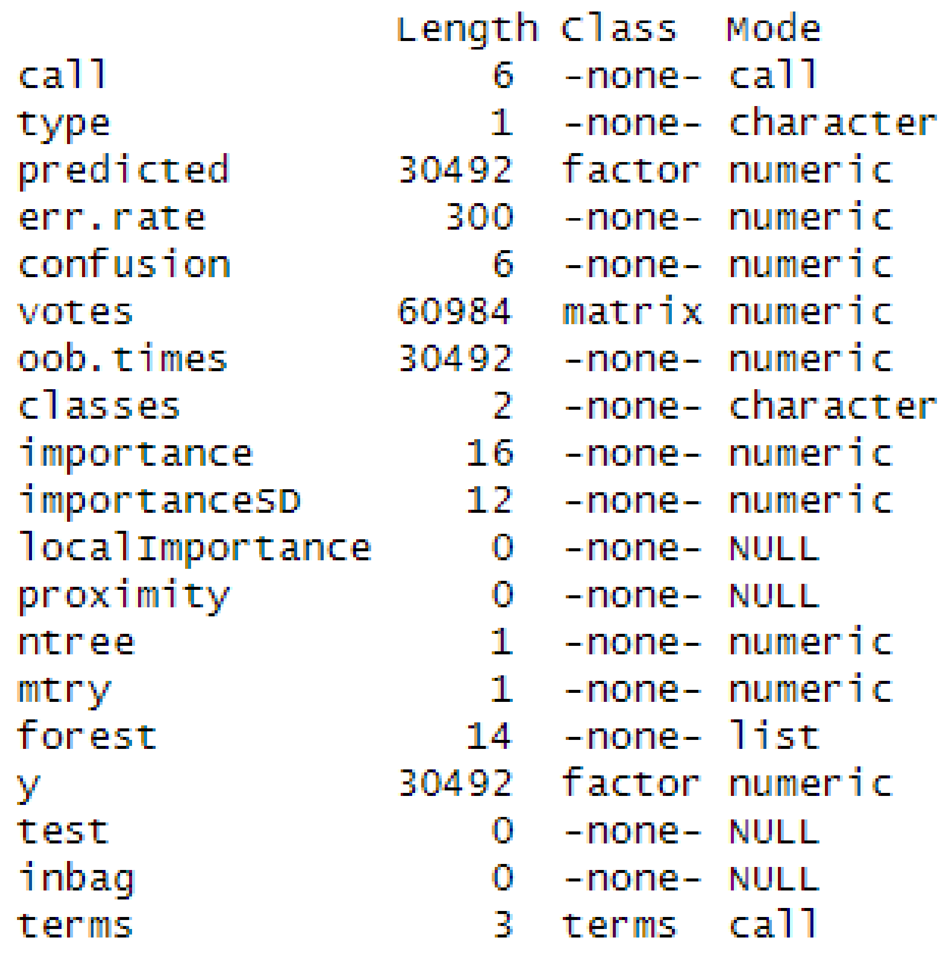
The test_data consists of 7,624 rows and 10 columns, representing a subset of the dataset used for model evaluation. It includes features such as fall status, year of occurrence, mass of objects, and their geographical coordinates (reclat, reclong), along with name. This data helps assess the model's performance on unseen samples after training.

5.3 Model Building

Random Forest:



~~~
```{r}
rf_model <- randomForest(
  fall ~ year + mass + reclat + reclong,
  data = train_data,
  ntree = 100,
  mtry = 2,
  importance = TRUE)
rf_model
```
~~~



```
                Length Class  Mode
call                6  -none- call
type                1  -none- character
predicted       30492  factor numeric
err.rate          300  -none- numeric
confusion           6  -none- numeric
votes           60984  matrix numeric
oob.times       30492  -none- numeric
classes             2  -none- character
importance         16  -none- numeric
importanceSD       12  -none- numeric
localImportance     0  -none- NULL
proximity           0  -none- NULL
ntree               1  -none- numeric
mtry                1  -none- numeric
forest             14  -none- list
y               30492  factor numeric
test                0  -none- NULL
inbag               0  -none- NULL
terms               3  terms  call
```

CHAPTER – VI
MODEL EVALUATION

6.1 Model Evaluation

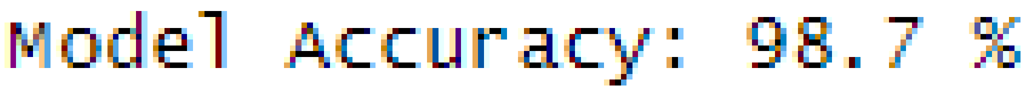
1. Model Evaluation Metrics:

When evaluating the Random Forest Classifier, we use several key metrics to understand the performance of the model:

- **Accuracy:**

Proportion of correctly predicted instances among the total instances.

```
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
cat("\nModel Accuracy:", round(accuracy * 100, 2), "%\n")
```

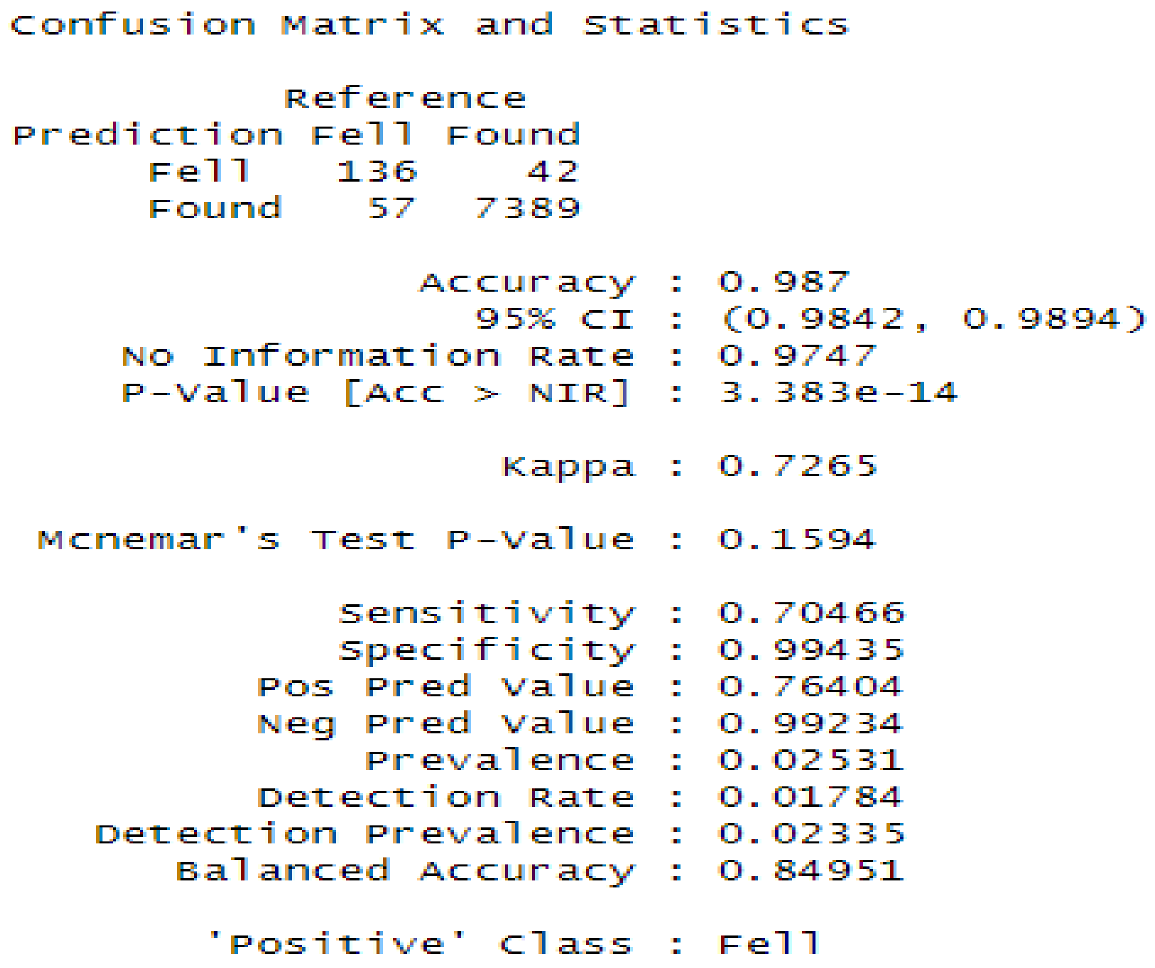


```
Model Accuracy: 98.7 %
```

- **Confusion Matrix:**

Displays True Positives, True Negatives, False Positives, and False Negatives.

```
confusionMatrix(predictions, test_data$fall)
```



```
Confusion Matrix and Statistics

          Reference
Prediction Fell Found
     Fell   136    42
     Found   57  7389

               Accuracy : 0.987
                 95% CI : (0.9842, 0.9894)
    No Information Rate : 0.9747
    P-Value [Acc > NIR] : 3.383e-14

                  Kappa : 0.7265

 Mcnemar's Test P-Value : 0.1594

            Sensitivity : 0.70466
            Specificity : 0.99435
         Pos Pred Value : 0.76404
         Neg Pred Value : 0.99234
             Prevalence : 0.02531
         Detection Rate : 0.01784
   Detection Prevalence : 0.02335
      Balanced Accuracy : 0.84951

       'Positive' Class : Fell
```

2. Creating a DataFrame to Compare Actual vs Predicted:

After predicting the target values using the model, we create a new DataFrame to compare the actual values with the predicted ones.

3. Adding the Predicted Column to the Original Dataset:

To add the predicted column to the original dataset, follow these steps:

meteorite_data$predicted_fall <- predict(rf_model, meteorite_data, type = "response")

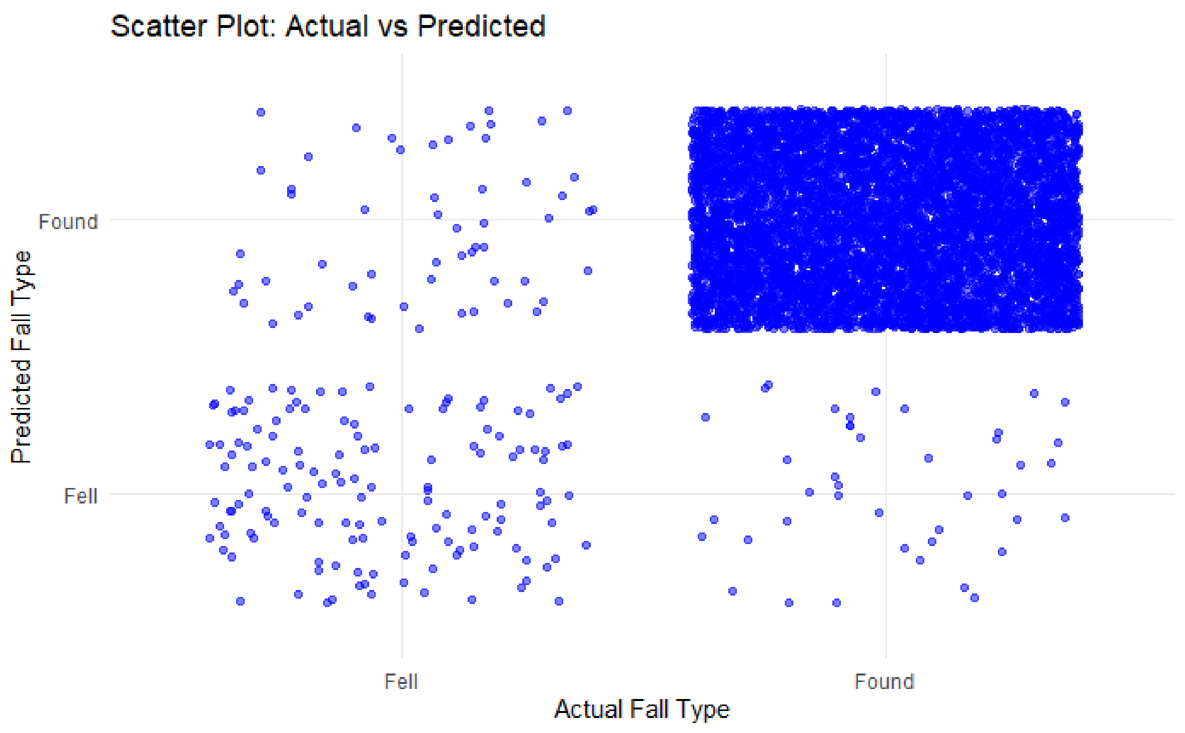
4.Visualizing Actual vs Predicted Values:

You can also visualize the comparison using graphs like:

- **Scatter Plot:**

To see the spread of actual vs predicted.

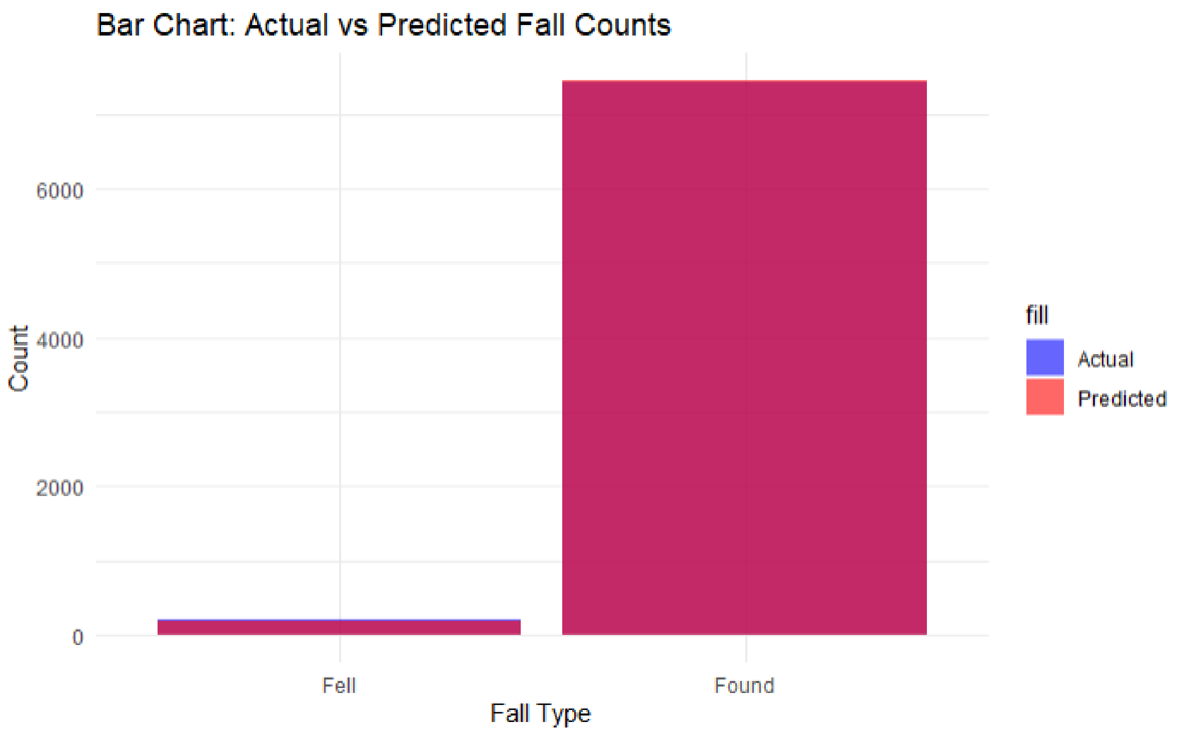
ggplot(data.frame(Actual = test_data$fall, Predicted = predictions), aes(x = Actual, y = Predicted)) + geom_jitter(color = "blue", alpha = 0.5) + labs(title = "Scatter Plot: Actual vs Predicted", x = "Actual Fall Type", y = "Predicted Fall Type") + theme_minimal()



- **Bar Chart:**

For categorical data comparisons.

data_comparison <- data.frame(Actual = test_data$fall, Predicted = predictions) ggplot(data_comparison) + geom_bar(aes(x = Actual, fill = "Actual"), alpha = 0.6, position = "dodge") + geom_bar(aes(x = Predicted, fill = "Predicted"), alpha = 0.6, position = "dodge") + labs(title = "Bar Chart: Actual vs Predicted Fall Counts", x = "Fall Type", y = "Count") +scale_fill_manual(values = c("Actual" = "blue", "Predicted" = "red")) + theme_minimal()



CHAPTER VII

PREDICTION AND INFERENCE

7.1 Predictions:

The Random Forest model was used to classify meteorite landings as 'Fell' or 'Found' based on attributes like year, mass, latitude, and longitude. The model achieved a high accuracy, demonstrating strong predictive performance.

- **Prediction Results:**

  The model effectively predicted the status of meteorites, showing a balanced performance across both classes ('Fell' and 'Found').

- **Confusion Matrix:**

  Provided detailed insights into true positives, true negatives, and misclassifications, helping evaluate the model's reliability.

- **Feature Importance:**

  Revealed which factors (e.g., mass, year, geographic coordinates) had the most influence on predictions, guiding further analysis.

7.2 Inferences:

- **Scientific Research:**

  The prediction model can help prioritize meteorite search efforts based on geographic and temporal patterns.

- **Risk Management:**

  Governments could use this data to identify high-risk areas for meteorite falls.

- **Educational Tools:**

  The dashboard and predictions could serve as a learning resource for geology and space science students. Overall, the integration of the Random Forest model with visualization tools like Power BI provided a holistic approach to understanding meteorite data and making data-driven predictions.

CHAPTER VIII

CONCLUSION

Our project journey began with the raw Meteor Dataset, aiming to classify meteorite landings as 'Fell' or 'Found' using a Random Forest Classifier. We started by cleaning the data and selecting relevant features such as year, mass, latitude, and longitude. The Random Forest model demonstrated strong predictive performance, with evaluation metrics like the Confusion Matrix showcasing its accuracy. We complemented this analysis with a dynamic Power BI dashboard, which visualized global meteorite distributions, trends over time, and a clear comparison of actual versus predicted results. Key insights revealed the significance of features like mass and geographic coordinates in determining meteorite status. The practical implications of this project span scientific research, risk management, and educational purposes, highlighting the power of combining machine learning with data visualization to transform raw data into actionable insights.

REFERENCE

NASA's Meteorite Landings Dataset:

This comprehensive dataset includes information on over 45,000 meteorites, detailing their location, mass, composition, and year of fall. It's a foundational resource for analyzing meteorite landings.

kaggle.com+1github.com+1

Determination of Strewn Fields for Meteorite Falls:

This research paper presents simulations for well-documented meteorite falls, offering insights into predicting meteorite landing areas

academic.oup.com

Dark-Flight Estimates of Meteorite Fall Positions:

This study addresses the challenges in estimating meteorite fall positions during their dark-flight phase and provides a case study using the Murrili meteorite fall. onlinelibrary.wiley.com+4researchgate.net+4researchgate.net+4

Meteorite Landing Prediction Project on GitHub:

This project utilizes NASA's dataset and artificial intelligence to predict meteorite trajectories to Earth, aiming to enhance precision for planetary defense.