Indian Institute of Information Technology Surat

Lab Report on
Machine Learning (CS 601) Practical

Submitted by

[RAHUL KUMAR SINGH] (UI21CS44)

Course Faculty
Dr. Pradeep Kumar Roy
Dr. Rajesh K. Ahir
Mr. Vipul Kania

Department of Computer Science and Engineering
Indian Institute of Information Technology Surat
Gujarat-394190, India

Jan-2024

Table of Contents

Lab No: 1

Aim:
Data Collection from E-Commerce, Twitter and Similar Platforms

Description:
Write a Python script for:

(a) Collecting tweets that may incorporate owner, date of post, number of retweet, number of followers, no of followers , and other associated information from Twitter and store it into a .csv file. (The size of collected tweets >5000)

(b) To scrap users reviews from any E-commerce or similar portals (Ex- Amazon, Flipkart, Yelp) and store it into a csv file that may incorporate date of post, number of likes/dislikes, reviews, location, and other associated fields (The size of collected reviews>5000).

Source Code:

For Task (a):

```python
# ----------------------------------------------------------------------------
from selenium import webdriver
from selenium.webdriver.common.by import By
from fake_useragent import UserAgent
from webdriver_manager.firefox import GeckoDriverManager
import time
import json
import os
from selenium.webdriver.common.keys import Keys

MY_USERNAME_VAR = os.getenv('USERNAME')
MY_PASS_VAR = os.getenv('PASS')
def wait_for_window(self, timeout = 2):
    time.sleep(round(timeout / 1000))
    wh_now = self.driver.window_handles
    wh_then = self.vars["window_handles"]
    if len(wh_now) > len(wh_then):
        return set(wh_now).difference(set(wh_then)).pop()
keywords = ["WWE","Rock","RomanReigns"]
ulrs = []
options = webdriver.FirefoxOptions()
options.headless = False
ua = UserAgent()
userAgent = ua.random
```

1

```python
options.add_argument(f'user-agent={userAgent}')

driver =
webdriver.Firefox(executable_path=GeckoDriverManager().install(),options=options)
driver.get("https://twitter.com/i/flow/login")
driver.maximize_window()
time.sleep(10)
try:
    input_element = driver.find_element(By.CSS_SELECTOR,
'.r-30o5oe.r-1niwhzg.r-17gur6a.r-1yadl64.r-deolkf.r-homxoj.r-poiln3')
    input_element.click()
    time.sleep(5)
    password_x = driver.find_element(By.CSS_SELECTOR,
'.r-30o5oe.r-1niwhzg.r-17gur6a.r-1yadl64.r-deolkf.r-homxoj.r-poiln3.r-7cikom.r-1ny4l
3l.r-t60dpp.r-1dz5y72.r-fdjqy7.r-13qz1uu')
    password_x.click()
    password_x.send_keys(MY_PASS_VAR)
    time.sleep(5)
    with open('keyword_numbers.json', 'w') as file:
        json.dump(keyword_numbers, file)

except Exception as e:
    print(ulrs)
    print("An error occurred:", str(e))
```

For Task (b):

```python
import csv
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

def extract_reviews(product_url, num_reviews_to_scrape=10):
    driver = webdriver.Chrome()
    driver.get(product_url)
    time.sleep(8)
    reviews = []
    review_elements = driver.find_elements(By.CSS_SELECTOR, '.a-section.review')
    temp_Date = ""
    for review_element in review_elements[:num_reviews_to_scrape]:
        time.sleep(1)
        review = {}
        review['author'] = review_element.find_element(By.CSS_SELECTOR,
```

```python
'.a-profile-name').text.strip()
        temp_Date = review_element.find_element(By.CSS_SELECTOR,
'.review-date').text.strip()
        review['date'] = temp_Date[temp_Date.find('on')+3:]
        review['location'] = temp_Date[12:temp_Date.find('on')-1]
        review['text'] = review_element.find_element(By.CSS_SELECTOR,
'.review-text-content').text.strip()
        review['rating'] =
review_element.find_element_by_xpath('//i[@data-hook="review-star-rating"]').text.st
rip()
        review['title'] = review_element.find_element(By.CSS_SELECTOR,
'.review-title').text.strip()
        reviews.append(review)
        print(review)
    driver.quit()
    return reviews
product_url =
'https://www.amazon.in/ZAPCASE-Compatible-Xiaomi-Covers-Carbon/product-reviews/B07GQ
Y2RN2/ref=cm_cr_arp_d_paging_btm_next_2?ie=UTF8&reviewerType=all_reviews'
reviews_data = []
for i in range(1,4):
    reviews_data += extract_reviews(product_url+'&pageNumber='+str(i),
num_reviews_to_scrape=10)
def export_csv(reviews, csv_filename='reviews_data.csv'):
    with open(csv_filename, 'w', newline='', encoding='utf-8') as csv_file:
        fieldnames = ['date','names','location','reviewtitles','ratings','reviews']
        writer = csv.DictWriter(csv_file, fieldnames=fieldnames)

        writer.writeheader()
        for review in reviews:
            writer.writerow({'date': review['date'], 'names': review['author'],
'location': review['location'], 'reviewtitles': review['title'], 'ratings':
review['rating'], 'reviews': review['text']})

export_csv(reviews_data)
```

Output:

For Task (a):

Figure 1.1 Output for Twitter Data Collection

For Task (b):

Figure 1.2 Output for Amazon Review Scraping

Conclusion:

- Efficient and direct access to Twitter's data through the API.
- Provides real-time data retrieval, enabling instant updates.
- Offers structured data in JSON format for easy processing.
- Overcomes API limitations for certain tasks, such as scraping dynamic content using custom scraping.

Lab No: 2

Aim:
To perform exploratory data analysis on the attached dataset

Description:
Perform the Exploratory Data Analysis (EDA) by considering the following tasks. Use the attached dataset for the same.

1. Check for Duplication
2. Missing Values Calculation
3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)
4. Feature Engineering
5. Creating Features
6. Data Cleaning/Wrangling
7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)
8. Analyzing/visualizing the dataset by taking one variable at a time
9. Data Transformation

Source Code:

Import Libraries and Read Dataset

```python
import pandas as pd
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler, LabelEncoder
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime

df = pd.read_csv('cars_data.csv')
print(df.head())
```

1. Check for Duplication

```python
duplicates = df.duplicated()
print(df[duplicates])
```

```python
num_duplicates = df.duplicated().sum()
percentage_duplicates = (num_duplicates / len(df)) * 100

print(f"Number of duplicate rows: {num_duplicates}")
print(f"Percentage of duplicate rows: {percentage_duplicates:.2f}%")
```

```python
df = df.drop_duplicates()
print(df.head())
```

2. Missing Values Calculation

```python
total_missing = df.isnull().sum().sum()
print(total_missing)
```

```python
missing_by_column = df.isnull().sum()
print(missing_by_column)
```

```python
percentage_missing = (df.isnull().sum() / len(df)) * 100
print(percentage_missing)
```

3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)

```python
# Replace missing values
df['Price'].fillna(0, inplace=True)
df['New_Price'].fillna(0, inplace=True)
df.dropna(inplace=True) # Dropping few inconsequntial records
```

```python
# Drop irrelevant columns for analysis
cols_to_drop = ['Name','Location','Fuel_Type','Transmission','Owner_Type','Mileage','Engine','Power','New_Price']
dropdf = df.drop(columns=cols_to_drop)

scaler = StandardScaler()
cars_data_scaled = scaler.fit_transform(dropdf)

# Apply Principal Component Analysis (PCA) for dimensionality reduction
pca = PCA(n_components=2)
cars_pca = pca.fit_transform(cars_data_scaled)

plt.figure(figsize=(10, 6))
plt.scatter(cars_pca[:, 0], cars_pca[:, 1])
plt.title('PCA: First Two Principal Components')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

4. Feature Engineering

```python
selected_features = df[['S.No.', 'Kilometers_Driven', 'Seats', 'Price']]

scaler = StandardScaler()
scaled_features = scaler.fit_transform(selected_features)

num_components = 2
pca = PCA(n_components=num_components)
reduced_features = pca.fit_transform(scaled_features)
reduced_features_df = pd.DataFrame(data=reduced_features, columns=['PC1', 'PC2'])
final_data = pd.concat([df, reduced_features_df], axis=1)
print(final_data.head())
```

5. Creating Features

```python
cars_df = df.copy()
cars_df['Brand'] = cars_df['Name'].str.split().str[0]
cars_df['Mileage'] = cars_df['Mileage'].str.split().str[0]
cars_df['Mileage'] = pd.to_numeric(df['Mileage'], errors='coerce')
current_year = datetime.now().year
cars_df['Age'] = current_year - cars_df['Year']
cars_df['Price_per_Mile'] = cars_df['Price'] / cars_df['Mileage']

print("\nCars Dataset with New Features:")
print(cars_df)
```

```python
# Visualization
plt.figure(figsize=(12, 6))

plt.subplot(1, 2, 1)
sns.scatterplot(x='Age', y='Price', data=cars_df, hue='Brand', palette='Set1')
plt.title('Age vs Price')

plt.subplot(1, 2, 2)
sns.scatterplot(x='Mileage', y='Price_per_Mile', data=cars_df, hue='Brand', palette='Set2')
plt.title('Mileage vs Price_per_Mile')

# plt.tight_layout()
plt.show()
```

6. Data Cleaning/Wrangling

```python
clean_df = df.copy()

clean_df['Brand'] = clean_df['Name'].str.split().str[0]
clean_df['Engine'] = clean_df['Engine'].str.extract('(\d+)').astype(float)
clean_df['Mileage'] = clean_df['Mileage'].str.extract('(\d+)').astype(float)
clean_df['Power'] = clean_df['Power'].str.extract('(\d+)').astype(float)
clean_df['New_Price'] = clean_df['New_Price'].str.extract('(\d+)').astype(float)
clean_df['New_Price'].fillna(0, inplace=True)
current_year = datetime.now().year
clean_df['Mileage'][clean_df['Mileage']==0] = 1
clean_df['Age'] = current_year - clean_df['Year']
clean_df['Price_per_Mile'] = clean_df['Price'] / clean_df['Mileage']
clean_df = clean_df.drop(['Name', 'Year'], axis=1)

print("\nCleaned and Wrangled Dataset:")
print(clean_df)
```

7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)

```python
print("Dataset Information:")
print(df.info())

print("\nSummary Statistics:")
print(df.describe())
```

```python
cars_data = dropdf.copy()
summary_stats = {
    'Count': cars_data.shape[0],
    'Mean': cars_data.mean(),
    'Standard Deviation': cars_data.std(),
    'Median': cars_data.median(),
    'Mode': cars_data.mode().iloc[0],
    'Minimum Value': cars_data.min(),
    'Maximum Value': cars_data.max(),
    'Range': cars_data.max() - cars_data.min(),
}
summary_df = pd.DataFrame(summary_stats)

print("\nStatistics Summary:")
print(summary_df)
```

```python
import seaborn as sns
import matplotlib.pyplot as plt

sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
plt.show()
```

8. Analyzing/visualizing the dataset by taking one variable at a time

```python
cars_data = clean_df.copy()
def visualize_variable(variable_name):
    plt.figure(figsize=(8, 6))
    plt.hist(cars_data[variable_name], bins=20, color='skyblue', edgecolor='black')
    plt.title(f'Distribution of {variable_name}')
    plt.xlabel(variable_name)
    plt.ylabel('Frequency')
    plt.show()

numerical_variables = cars_data.select_dtypes(include='number').columns
for variable in numerical_variables:
    visualize_variable(variable)
```

9. Data Transformation

```python
cars_data = clean_df.copy()
# Encode Categorical Variables
label_encoder = LabelEncoder()
cars_data['Brand'] = label_encoder.fit_transform(cars_data['Brand'])
cars_data['Fuel_Type'] = label_encoder.fit_transform(cars_data['Fuel_Type'])
cars_data['Transmission'] = label_encoder.fit_transform(cars_data['Transmission'])

# Scale Numerical Features
numerical_features = ['Price', 'Mileage', 'Engine']
scaler = StandardScaler()
cars_data[numerical_features] = scaler.fit_transform(cars_data[numerical_features])

print("\nTransformed Dataset:")
print(cars_data.head())
```

Output:

1. Check for Duplication

1. Check for Duplication

```python
duplicates = df.duplicated()
print(df[duplicates])
```

```
Empty DataFrame
Columns: [S.No., Name, Location, Year, Kilometers_Driven, Fuel_Type, Transmission, Owner_Type, Mileage, Engine, Power, Seats, New_Price, Price]
Index: []
```

```python
num_duplicates = df.duplicated().sum()
percentage_duplicates = (num_duplicates / len(df)) * 100

print(f"Number of duplicate rows: {num_duplicates}")
print(f"Percentage of duplicate rows: {percentage_duplicates:.2f}%")
```

```
Number of duplicate rows: 0
Percentage of duplicate rows: 0.00%
```

Figure 2.1 Output for task 1

2. Missing Values Calculation

```
[5]: total_missing = df.isnull().sum().sum()
     print(total_missing)

     7636

[6]: missing_by_column = df.isnull().sum()
     print(missing_by_column)

     S.No.                0
     Name                 0
     Location             0
     Year                 1
     Kilometers_Driven    1
     Fuel_Type            2
     Transmission         1
     Owner_Type           2
     Mileage              3
     Engine              46
     Power               46
     Seats               53
     New_Price         6247
     Price             1234
     dtype: int64
```

Figure 2.2 Output for task 2

3. Data Reduction (Some columns or variables can be dropped if they do not add value to our analysis.)

```
        S.No.     Year  Kilometers_Driven  Seats  Price
0           0   2010.0            72000.0    5.0   1.75
1           1   2015.0            41000.0    5.0  12.50
2           2   2011.0            46000.0    5.0   4.50
3           3   2012.0            87000.0    7.0   6.00
4           4   2013.0            40670.0    5.0  17.74
...       ...      ...                ...    ...    ...
7248     7248   2011.0            89411.0    5.0   0.00
7249     7249   2015.0            59000.0    5.0   0.00
7250     7250   2012.0            28000.0    5.0   0.00
7251     7251   2013.0            52262.0    5.0   0.00
7252     7252   2014.0            72443.0    5.0   0.00
```

Figure 2.3 Output for task 3

4. Feature Engineering

```
    S.No.                            Name     Location     Year  \
0     0.0           Maruti Wagon R LXI CNG      Mumbai   2010.0
1     1.0    Hyundai Creta 1.6 CRDi SX Option     Pune   2015.0
2     2.0                    Honda Jazz V      Chennai   2011.0
3     3.0                Maruti Ertiga VDI    Chennai   2012.0
4     4.0    Audi A4 New 2.0 TDI Multitronic  Coimbatore 2013.0

   Kilometers_Driven Fuel_Type Transmission Owner_Type     Mileage  Engine  \
0            72000.0       CNG       Manual      First  26.6 km/kg   998 CC
1            41000.0    Diesel       Manual      First  19.67 kmpl  1582 CC
2            46000.0    Petrol       Manual      First   18.2 kmpl  1199 CC
3            87000.0    Diesel       Manual      First  20.77 kmpl  1248 CC
4            40670.0    Diesel    Automatic     Second   15.2 kmpl  1968 CC

      Power  Seats  New_Price  Price       PC1       PC2
0  58.16 bhp   5.0          0   1.75  0.746503 -0.304137
1  126.2 bhp   5.0          0  12.50  1.421956 -0.656362
2   88.7 bhp   5.0  8.61 Lakh   4.50  0.906441 -0.545824
3  88.76 bhp   7.0          0   6.00  1.453131  1.470690
4  140.8 bhp   5.0          0  17.74  1.760415 -0.703806
```

Figure 2.4 Output for task 4

5. Creating Features

```
2         2                         Honda Jazz V        Chennai
3         3                     Maruti Ertiga VDI        Chennai
4         4          Audi A4 New 2.0 TDI Multitronic  Coimbatore
...       ...                               ...            ...
7248    7248          Volkswagen Vento Diesel Trendline  Hyderabad
7249    7249                 Volkswagen Polo GT TSI       Mumbai
7250    7250                 Nissan Micra Diesel XV       Kolkata
7251    7251                 Volkswagen Polo GT TSI         Pune
7252    7252  Mercedes-Benz E-Class 2009-2013 E 220 CDI Avan...    Kochi


        Year  Kilometers_Driven Fuel_Type Transmission Owner_Type  Mileage  \
0      2010.0          72000.0       CNG      Manual     First       NaN
1      2015.0          41000.0    Diesel      Manual     First       NaN
2      2011.0          46000.0    Petrol      Manual     First       NaN
3      2012.0          87000.0    Diesel      Manual     First       NaN
4      2013.0          40670.0    Diesel   Automatic    Second       NaN
...       ...             ...      ...        ...        ...          ...
7248   2011.0          89411.0    Diesel      Manual     First       NaN
7249   2015.0          59000.0    Petrol   Automatic     First       NaN
7250   2012.0          28000.0    Diesel      Manual     First       NaN
7251   2013.0          52262.0    Petrol   Automatic     Third       NaN
7252   2014.0          72443.0    Diesel   Automatic     First       NaN


        Engine      Power  Seats  New_Price  Price          Brand   Age  \
0       998 CC   58.16 bhp    5.0          0   1.75         Maruti  14.0
1      1582 CC   126.2 bhp    5.0          0  12.50        Hyundai   9.0
2      1199 CC    88.7 bhp    5.0  8.61 Lakh   4.50          Honda  13.0
3      1248 CC   88.76 bhp    7.0          0   6.00         Maruti  12.0
4      1968 CC   140.8 bhp    5.0          0  17.74           Audi  11.0
...       ...        ...      ...        ...    ...            ...   ...
7248   1598 CC   103.6 bhp    5.0          0   0.00     Volkswagen  13.0
7249   1197 CC   103.6 bhp    5.0          0   0.00     Volkswagen   9.0
7250   1461 CC    63.1 bhp    5.0          0   0.00         Nissan  12.0
7251   1197 CC   103.6 bhp    5.0          0   0.00     Volkswagen  11.0
7252   2148 CC     170 bhp    5.0          0   0.00  Mercedes-Benz  10.0
```

Figure 2.5.1 Tabular Representation
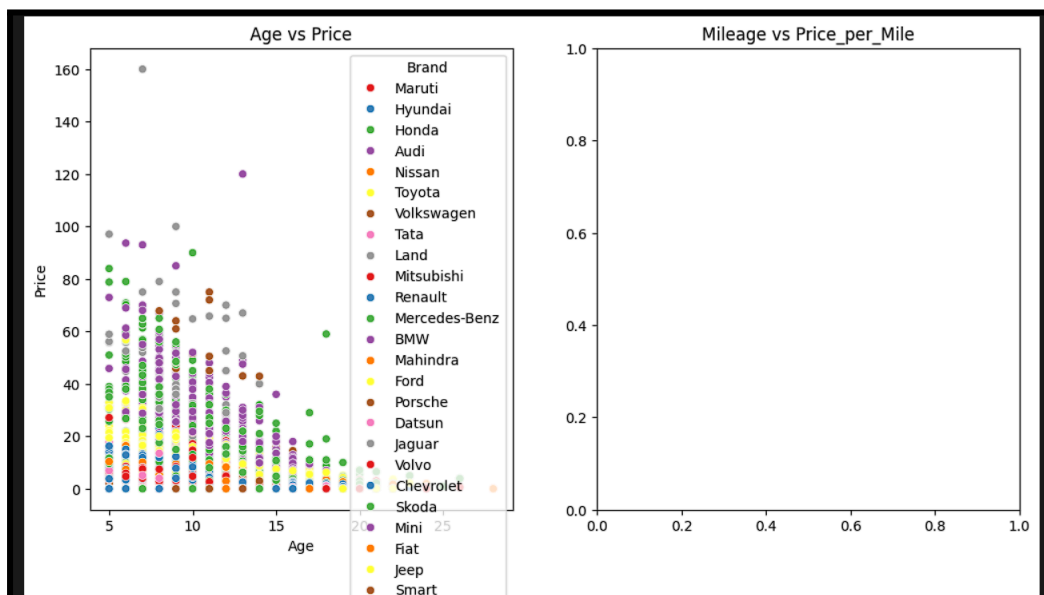
Figure 2.5.2 Graphical Representation

10

6. Data Cleaning/Wrangling

```
Cleaned and Wrangled Dataset:
      S.No.    Location  Kilometers_Driven Fuel_Type Transmission Owner_Type  \
0         0      Mumbai            72000.0       CNG       Manual      First
1         1        Pune            41000.0    Diesel       Manual      First
2         2     Chennai            46000.0    Petrol       Manual      First
3         3     Chennai            87000.0    Diesel       Manual      First
4         4  Coimbatore            40670.0    Diesel    Automatic     Second
...     ...         ...                ...       ...          ...        ...
7248   7248   Hyderabad            89411.0    Diesel       Manual      First
7249   7249      Mumbai            59000.0    Petrol    Automatic      First
7250   7250     Kolkata            28000.0    Diesel       Manual      First
7251   7251        Pune            52262.0    Petrol    Automatic      Third
7252   7252       Kochi            72443.0    Diesel    Automatic      First

      Mileage  Engine  Power  Seats  New_Price  Price          Brand   Age  \
0        26.0   998.0   58.0    5.0        0.0   1.75         Maruti  14.0
1        19.0  1582.0  126.0    5.0        0.0  12.50        Hyundai   9.0
2        18.0  1199.0   88.0    5.0        8.0   4.50          Honda  13.0
3        20.0  1248.0   88.0    7.0        0.0   6.00         Maruti  12.0
4        15.0  1968.0  140.0    5.0        0.0  17.74           Audi  11.0
...       ...     ...    ...    ...        ...    ...            ...   ...
7248     20.0  1598.0  103.0    5.0        0.0   0.00     Volkswagen  13.0
7249     17.0  1197.0  103.0    5.0        0.0   0.00     Volkswagen   9.0
7250     23.0  1461.0   63.0    5.0        0.0   0.00         Nissan  12.0
7251     17.0  1197.0  103.0    5.0        0.0   0.00     Volkswagen  11.0
7252     10.0  2148.0  170.0    5.0        0.0   0.00  Mercedes-Benz  10.0
```

Figure 2.6 Output for task 6

7. Statistics Summary (Count, Mean, Standard Deviation, median, mode, minimum value, maximum value, range, standard deviation)

```
Statistics Summary:
                   Count          Mean  Standard Deviation    Median      Mode  \
S.No.               7191   3627.190655         2094.568997    3629.0       0.0
Year                7191   2013.391322            3.235169    2014.0    2014.0
Kilometers_Driven   7191  58606.050897        84711.727076   53226.0   60000.0
Seats               7191      5.279516            0.811614       5.0       5.0
Price               7191      7.888618           10.819356       4.7       0.0

                   Minimum Value  Maximum Value        Range
S.No.                        0.0         7252.0       7252.0
Year                      1996.0         2019.0         23.0
Kilometers_Driven          171.0      6500000.0    6499829.0
Seats                        0.0           10.0         10.0
Price                        0.0          160.0        160.0
```

Figure 2.7 Output for task 7

8. Analyzing/visualizing the dataset by taking one variable at a time
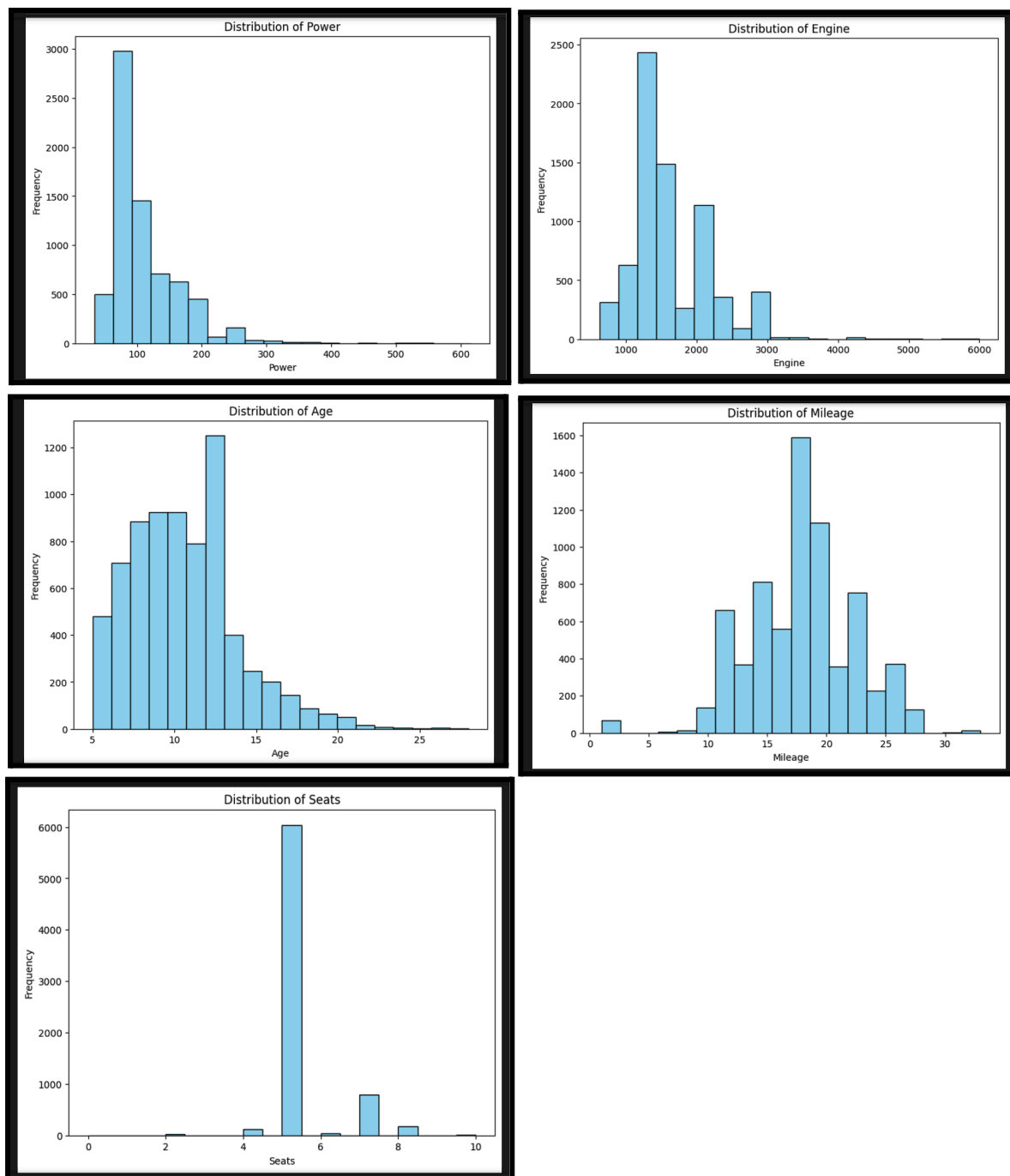
Figure 2.8 Output for task 8

9. Data Transformation

```
Transformed Dataset:
   S.No.    Location  Kilometers_Driven  Fuel_Type  Transmission Owner_Type  \
0      0      Mumbai            72000.0          0             1      First
1      1        Pune            41000.0          1             1      First
2      2     Chennai            46000.0          3             1      First
3      3     Chennai            87000.0          1             1      First
4      4  Coimbatore            40670.0          1             0     Second

    Mileage    Engine  Power  Seats  New_Price     Price  Brand   Age  \
0  1.842662 -1.039810   58.0    5.0        0.0 -0.567413     19  14.0
1  0.275923 -0.058350  126.0    5.0        0.0  0.426246     11   9.0
2  0.052103 -0.702013   88.0    5.0        8.0 -0.313221     10  13.0
3  0.499743 -0.619664   88.0    7.0        0.0 -0.174571     19  12.0
4 -0.619356  0.590354  140.0    5.0        0.0  0.910596      1  11.0

    Price_per_Mile
0         0.067308
1         0.657895
2         0.250000
3         0.300000
4         1.182667
```

Figure 2.9 Output for task 9

Conclusion:

- EDA provides a comprehensive overview of the cars dataset
- Identification and handling of missing values, outliers, and anomalies ensure data integrity and improve analysis accuracy.
- Descriptive statistics, including mean, median, and standard deviation, offer a summary of numerical attributes, aiding in understanding central tendencies and data dispersion.
- Visualization techniques, such as histograms and kernel density plots, reveal the distributions of key features, providing insights into the data's underlying patterns.
- Techniques like correlation, mutual information, or model-based feature importance assessments help prioritize variables based on their impact on the target variable.

Lab No: 3

Aim:
To perform linear regression and utilize Python libraries to plot attribute relations, design optimal line fitting, and analyze global minima for given data.

Description:
Perform the following task with using inbuilt Python Libraries.:

- Plot the input-output relation for given attributes.
- Design a mathematical function to find the best-fitted line for the given data (attached here).
- Plot Error vs. Slope graph and show the global minima for the sample data X={2, 4, 6, 8} and Y={3, 7, 5, 10} considering different learning rate values (alpha).

Source Code:

Task1: Plot the input-output relation for given attributes. ¶

```python
import pandas as pd
import matplotlib.pyplot as plt

# Load data
csv_file_path = 'Salary_Data.csv'
data = pd.read_csv(csv_file_path)

# Extracting input-output column
years_of_experience = data['YearsExperience']
salary = data['Salary']

# Plotting the input-output relationship
plt.figure(figsize=(10, 6))
plt.scatter(years_of_experience, salary, color='blue', marker='o')
plt.title('Input-Output Relationship: Years of Experience vs Salary')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.grid(True)
plt.show()
```

Task2: Design a mathematical function to find the best-fitted line for the given data (attached here).

```python
import numpy as np
def linear_regression(x, y):
    n = len(x)
    mean_x, mean_y = np.mean(x), np.mean(y)
    m = np.sum((x - mean_x) * (y - mean_y)) / np.sum((x - mean_x) ** 2)
    b = mean_y - m * mean_x
    return m, b
years_of_experience = data['YearsExperience']
salary = data['Salary']
slope, intercept = linear_regression(years_of_experience, salary)
print(f"Best-fitted line: y = {slope:.2f}x + {intercept:.2f}")
```

```
Best-fitted line: y = 9449.96x + 25792.20
```

```python
best_fit_line = slope * years_of_experience + intercept

# Plotting the input-output relationship and the best-fitted line
plt.figure(figsize=(10, 6))
plt.scatter(years_of_experience, salary, color='blue', marker='o', label='Data points')
plt.plot(years_of_experience, best_fit_line, color='red', label='Best-fitted line')
plt.title('Input-Output Relationship with Best-Fitted Line')
plt.xlabel('Years of Experience')
plt.ylabel('Salary')
plt.legend()
plt.grid(True)
plt.show()
```

Task3: Plot Error vs. Slope graph and show the global minima for the sample data X={2, 4, 6, 8} and Y={3, 7, 5, 10} considering different learning rate values (alpha).

```python
import numpy as np
import matplotlib.pyplot as plt
X = np.array([2, 4, 6, 8])
Y = np.array([3, 7, 5, 10])
def mean_squared_error(slope, X, Y):
    predictions = slope * X
    error = np.mean((predictions - Y) ** 2)
    return error
def gradient_descent(X, Y, alpha, iterations):
    slopes = []
    errors = []
    slope = 0
    for _ in range(iterations):
        slope = slope - alpha * (1/len(X)) * np.sum((slope * X - Y) * X)
        error = mean_squared_error(slope, X, Y)
        slopes.append(slope)
        errors.append(error)
    return slopes, errors
alpha_values = [0.01, 0.02, 0.03, 0.04]
plt.figure(figsize=(10, 6))
for alpha in alpha_values:
    slopes, errors = gradient_descent(X, Y, alpha, iterations=100)
    plt.plot(slopes, errors, label=f'Alpha = {alpha}')
plt.title('Error vs. Slope for Different Learning Rates')
plt.xlabel('Slope')
plt.ylabel('Mean Squared Error')
plt.legend()
plt.grid(True)
plt.show()
```

Output:

1. Plot the input-output relation for given attributes.
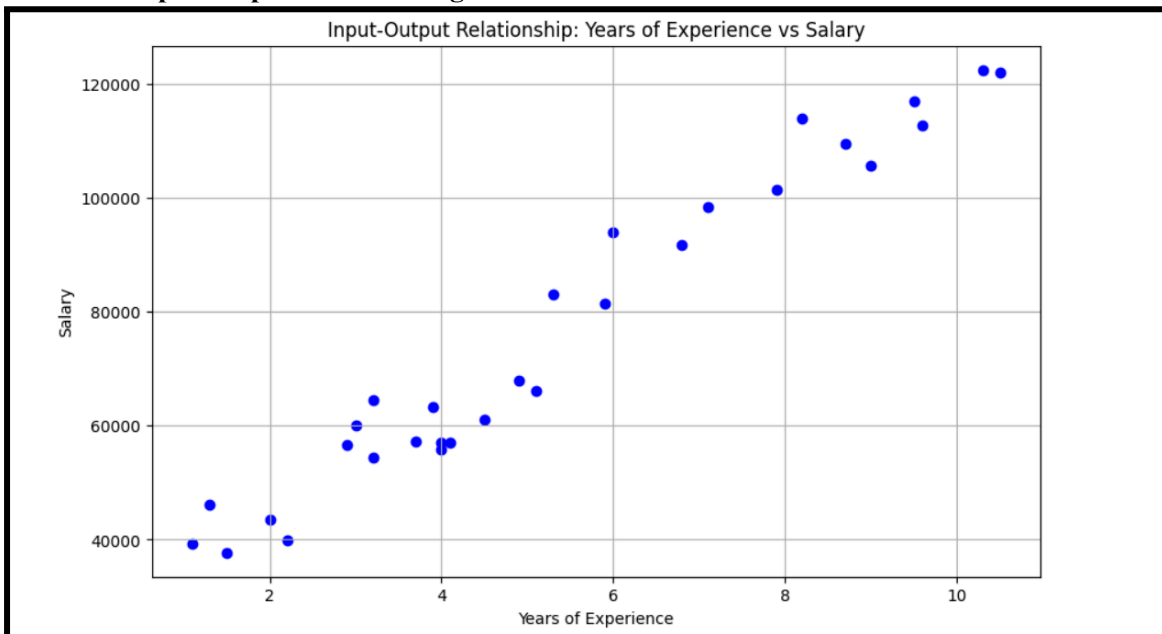
Figure 3.1 Output for Input-Output Relation

2. Design a mathematical function to find the best-fitted line for the given data

```
Best-fitted line: y = 9449.96x + 25792.20
```
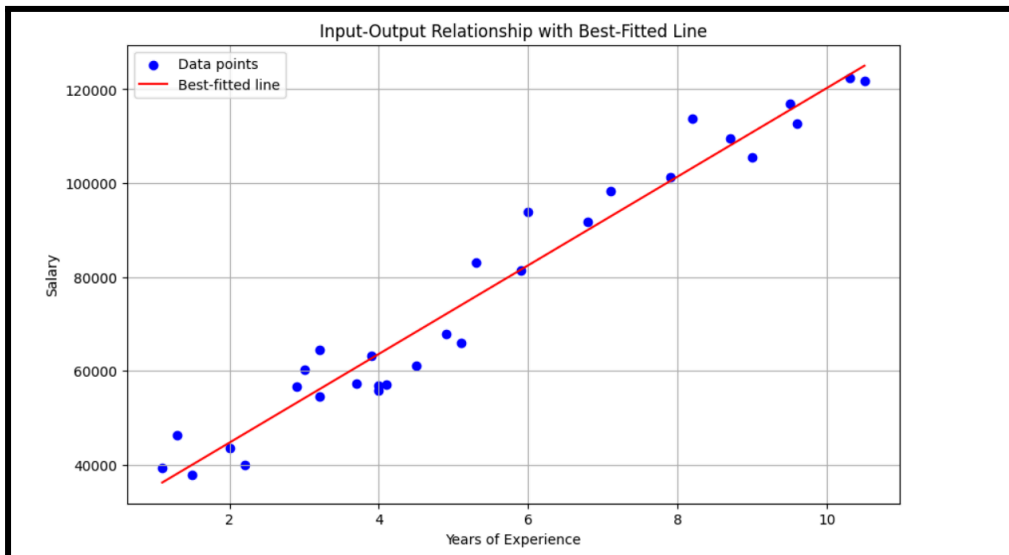

Figure 3.2 Output for Best-Fit Line

3. Plot Error vs. Slope graph and show the global minima for the sample data X={2, 4, 6, 8} and Y={3, 7, 5, 10} considering different learning rate values (alpha).
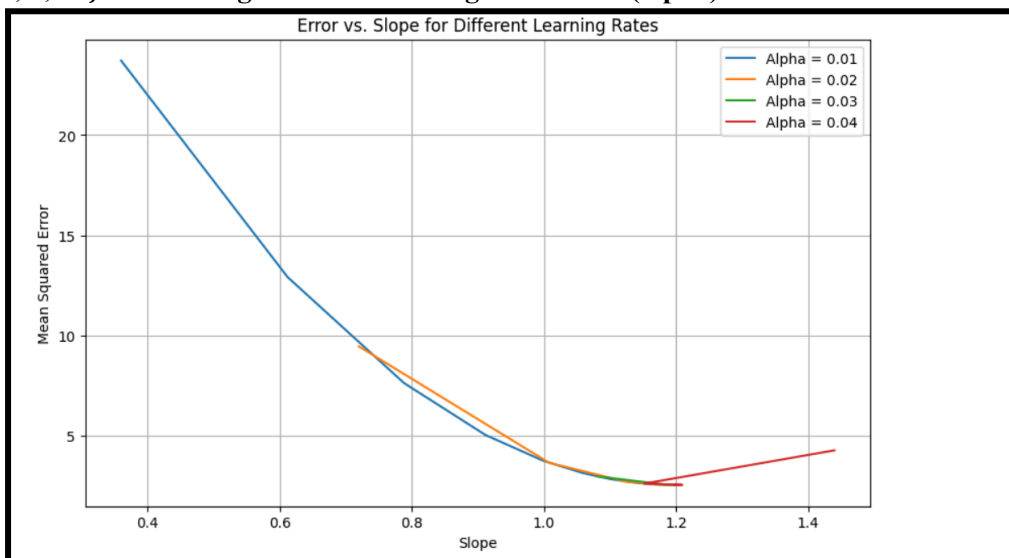
Figure 3.3 Error vs Slope Graph

Conclusion:

- A custom linear regression function was developed to find the best-fitted line for the given data.
- Utilized the gradient descent algorithm to minimize the cost function, aiming to find the optimal slope for the given linear regression problem.
- Plotted the Error vs. Slope graph for each learning rate, illustrating the convergence behavior over epochs.
- The impact of the learning rate on the convergence speed and the final error was observed through the plotted graphs.

Lab No: 4

Aim:

Study of essential text pre-processing techniques. Write python script for the essential text preprocessing techniques. Store the preprocessed data into a separate column of .CSV file. Compare the outcomes with and without using libraries for the same.

Description:

Perform the following task with using inbuilt Python Libraries:

- Lower Casing: Converts text into lower case text. It Helps ensure uniformity in text analysis and processing, as it treats uppercase and lowercase forms of words as the same.
- Tokenization: Break the text into individual words or tokens. It Facilitates analysis at the word level, making it easier to extract meaningful information and perform various natural language processing tasks.
- Punctuation Mark Removal: Eliminate punctuation marks from the text. Enhances the accuracy of text analysis by removing non-alphanumeric characters that don't contribute to the core meaning of the text.
- Stop Word Removal: Exclude common words (stop words) like "and," "the," and "is" that don't carry significant meaning. Improves the efficiency of text processing and analysis by focusing on content-bearing words.
- Stemming: Reduce words to their root or base form by removing suffixes. Aims to group variations of a word together, simplifying analysis and information retrieval. For example, "running" becomes "run."
- Lemmatization: Similar to Stemming but considers the word's context to reduce it to its base or dictionary form (lemma). Results in more accurate representation of the base form of a word, addressing potential ambiguities introduced by stemming.
- Translation: Convert text from one language to another. Facilitates cross-language communication and analysis, enabling understanding of content in different linguistic contexts.
- Emoji to Text: Translate emojis (emotion icons) into their corresponding textual representation. Helps in extracting meaning from textual data that includes emojis, making it easier for analysis and understanding sentiment.

Source Code:

```python
# Study of essential text pre-processing techniques. Write python script for the essential text preprocessing techniques. Store the preprocessed data
into a separate column of .CSV file. Compare the outcomes with and without using libraries for the same.

## Perform the following task with using inbuilt Python Libraries:

import pandas as pd
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
from deep_translator import GoogleTranslator
import emoji
import string
import re

nltk.download('stopwords')

nltk.download('punkt')

nltk.download('wordnet')

data = pd.read_csv("PTweet_WWE.csv")
data.head()

df = pd.DataFrame(data['text'])
df.head()

### 1. Lower Casing

# Task 1: Lowercasing
df['lowercased_text'] = df['text'].apply(lambda x: x.lower())
df.head()

### 2. Tokenization

# Task 2: Tokenization
# df['tokens'] = df['lowercased_text'].apply(lambda x: re.findall(r'\b\w+\b', x))
df['tokens'] = df['lowercased_text'].apply(lambda x: word_tokenize(x))
df.head()
```

```python
### 3. Punctuation Mark Removal

# Task 3: Punctuation Mark Removal
df['cleaned_text'] = df['tokens'].apply(lambda x: ''.join(char for char in x if char not in string.punctuation))
df.head()

### 4. Stop Word Removal

# Task 4: Stop Word Removal
stop_words = set(stopwords.words('english'))
df['filtered_text'] = df['tokens'].apply(lambda x: ' '.join(word for word in x if word not in stop_words))
df.head()

### 5. Stemming

# Task 5: Stemming
stemmer = PorterStemmer()
df['stemmed_Text'] = df['tokens'].apply(lambda x: ' '.join(stemmer.stem(word) for word in x))
df.head()

### 6. Lemmatization

# Task 6: Lemmatization
lemmatizer = WordNetLemmatizer()
df['lemmatized_text'] = df['tokens'].apply(lambda x: ' '.join(lemmatizer.lemmatize(word) for word in x))
df.head()
```

```
### 7. Translation

# Task 7: Translation
# translator = google_translator()
df['translated_text'] = df['lowercased_text'].apply(lambda x: GoogleTranslator(source='auto', target='es').translate(x))  # Translate to Spanish
df.head()

### 8. Emoji to text

# Task 8: Emoji to Text
df['emoji_to_text'] = df['text'].apply(lambda x: emoji.demojize(x))
df.head()
```

```
## Perform the following task without using inbuilt Python Libraries (The last two task (Translation and Emoji) are not possible without libraies):

import re
import string

# Sample text data
text_data = data.head()['text']

# Task 1: Lowercasing
lowercased_texts = [text.lower() for text in text_data]

# Task 2: Tokenization
tokenized_texts = [re.findall(r'\b\w+\b', text) for text in text_data]

# Task 3: Punctuation Mark Removal
cleaned_texts = [''.join(char for char in text if char not in string.punctuation) for text in text_data]

# Task 4: Stop Word Removal
stop_words = set(["a", "an", "the", "is", "from", "this"])
filtered_texts = [' '.join(word for word in text.split() if word.lower() not in stop_words) for text in text_data]

# Task 5: Stemming
def simple_stemming(text):
    return ' '.join(word[:4] if len(word) > 4 else word for word in text.split())

stemmed_texts = [simple_stemming(text) for text in text_data]

# Task 6: Lemmatization
def simple_lemmatization(text):
    return ' '.join(word[:-2] if word.endswith("es") else word for word in text.split())

lemmatized_texts = [simple_lemmatization(text) for text in text_data]

# Display results
for i in range(len(text_data)):
    print(f"\nOriginal Text: {text_data[i]}")
    print(f"Lowercased Text: {lowercased_texts[i]}")
    print(f"Tokenized Text: {tokenized_texts[i]}")
    print(f"Cleaned Text: {cleaned_texts[i]}")
    print(f"Filtered Text: {filtered_texts[i]}")
    print(f"Stemmed Text: {stemmed_texts[i]}")
    print(f"Lemmatized Text: {lemmatized_texts[i]}")
```

Output:

Twitter Data:

	link	text	name	username	date	is_rt	n_comment	n_rt	n_quote	n_like
0	https://twitter.com/WWE/status/174201779437427...	IF YA SMELL..... @TheRock has come back to #W...	WWE	@WWE	Jan 2, 2024 · 2:59 AM UTC	False	1088	9916	1856	49998
1	https://twitter.com/WWE/status/174573989977118...	Who had the best Instagram photo of the week?!...	WWE	@WWE	Jan 12, 2024 · 9:30 AM UTC	False	47	39	3	342
2	https://twitter.com/WWE/status/174567949971749...	These #RoyalRumble crashers were RUTHLESS! ht...	WWE	@WWE	Jan 12, 2024 · 5:30 AM UTC	False	46	72	2	624
3	https://twitter.com/WWE/status/174564199274113...	An All Mighty moment in the 2023 Men's #RoyalR...	WWE	@WWE	Jan 12, 2024 · 3:00 AM UTC	False	58	213	17	2454
4	https://twitter.com/WWE/status/174559668762676...	Outta nowhere! 😳	WWE	@WWE	Jan 12, 2024 · 12:00 AM UTC	False	70	354	10	3853

Figure 4.0 Twitter Data

Perform the following task with using inbuilt Python Libraries:

1. Lower Casing

	text	lowercased_text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...
4	Outta nowhere! 😳	outta nowhere! 😳

Figure 4.1.1 Lower Casing

2. Tokenization

	text	lowercased_text	tokens
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]

Figure 4.1.2 Tokenization

3. Punctuation Mark Removal

	text	lowercased_text	tokens	cleaned_text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]	outtanowhere 😳

Figure 4.1.3 Punctuation Mark Removal

4. Stop Word Removal

	text	lowercased_text	tokens	cleaned_text	filtered_text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw	ya smell ..... @ therock come back # wweraw !
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...	best instagram photo week ? ! https : //www.ww...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...	# royalrumble crashers ruthless ! https : //tu...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch	mighty moment 2023 men 's # royalrumble match !
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]	outtanowhere 😳	outta nowhere ! 😳

Figure 4.1.4 Stop Word Removal

5. Stemming

	text	lowercased_text	tokens	cleaned_text	filtered_text	stemmed_Text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw	ya smell ..... @ therock come back # wweraw !	if ya smell ..... @ therock ha come back to # ...
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...	best instagram photo week ? ! https : //www.ww...	who had the best instagram photo of the week ?...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...	# royalrumble crashers ruthless ! https : //tu...	these # royalrumbl crasher were ruthless ! htt...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch	mighty moment 2023 men 's # royalrumble match !	an all mighti moment in the 2023 men 's # roya...
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]	outtanowhere 😳	outta nowhere ! 😳	outta nowher ! 😳

Figure 4.1.5 Stemming

6. Lemmatization

	text	lowercased_text	tokens	cleaned_text	filtered_text	stemmed_Text	lemmatized_text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw	ya smell ..... @ therock come back # wweraw !	if ya smell ..... @ therock ha come back to # ...	if ya smell ..... @ therock ha come back to # ...
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...	best instagram photo week ? ! https : //www.ww...	who had the best instagram photo of the week ?...	who had the best instagram photo of the week ?...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...	# royalrumble crashers ruthless ! https : //tu...	these # royalrumbl crasher were ruthless ! htt...	these # royalrumble crasher were ruthless ! ht...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch	mighty moment 2023 men 's # royalrumble match !	an all mighti moment in the 2023 men 's # roya...	an all mighty moment in the 2023 men 's # roya...
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]	outtanowhere 😳	outta nowhere ! 😳	outta nowher ! 😳	outta nowhere ! 😳

Figure 4.1.6 Lemmatization

7. Translation

	text	lowercased_text	tokens	cleaned_text	filtered_text	stemmed_Text	lemmatized_text	translated_text
0	IF YA SMELL..... @TheRock has come back to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw	ya smell ..... @ therock come back # wweraw !	if ya smell ..... @ therock ha come back to # ...	if ya smell ..... @ therock ha come back to # ...	si hueles..... ¡@therock ha regresado a #wweraw!
1	Who had the best Instagram photo of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...	best instagram photo week ? ! https : //www.ww...	who had the best instagram photo of the week ?...	who had the best instagram photo of the week ?...	¿Quién tuvo la mejor foto de Instagram de la s...
2	These #RoyalRumble crashers were RUTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...	# royalrumble crashers ruthless ! https : //tu...	these # royalrumbl crasher were ruthless ! htt...	these # royalrumble crasher were ruthless ! ht...	¡Estos intrusos del #royalrumble fueron despia...
3	An All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch	mighty moment 2023 men 's # royalrumble match !	an all mighti moment in the 2023 men 's # roya...	an all mighty moment in the 2023 men 's # roya...	¡Un momento poderoso en el combate #royalrumbl...
4	Outta nowhere! 😳	outta nowhere! 😳	[outta, nowhere, !, 😳]	outtanowhere 😳	outta nowhere ! 😳	outta nowher ! 😳	outta nowhere ! 😳	¡de la nada! 😳

Figure 4.1.7 Translation

8. Emoji to text

text	lowercased_text	tokens	cleaned_text	filtered_text	stemmed_Text	lemmatized_text	translated_text	emoji_to_text
IF YA SMELL..... @TheRock has come ck to #W...	if ya smell..... @therock has come back to #w...	[if, ya, smell, ..., @, therock, has, come, ...	ifyasmell.....therockhascomebacktowweraw	ya smell ..... @ therock come back # wweraw !	if ya smell ..... @ therock ha come back # ...	if ya smell ..... @ therock ha come back to # ...	si hueles..... ¡@therock ha regresado a #wweraw!	IF YA SMELL..... @TheRock has come back to #W...
o had the best Instagram oto of the week?!...	who had the best instagram photo of the week?!...	[who, had, the, best, instagram, photo, of, th...	whohadthebestinstagramphotooftheweekhttps//www...	best instagram photo week ? ! https : //www.ww...	who had the best instagram photo of the week ?...	who had the best instagram photo of the week ?...	¿Quién tuvo la mejor foto de Instagram de la s...	Who had the best Instagram photo of the week?!...
These alRumble hers were UTHLESS! ht...	these #royalrumble crashers were ruthless! ht...	[these, #, royalrumble, crashers, were, ruthle...	theseroyalrumblecrasherswereruthlesshttps//tub...	# royalrumble crashers ruthless ! https : //tu...	these # royalrumbl crasher were ruthless ! htt...	these # royalrumble crasher were ruthless ! ht...	¡Estos intrusos del #royalrumble fueron despia...	These #RoyalRumble crashers were RUTHLESS! ht...
All Mighty moment in the 2023 Men's #RoyalR...	an all mighty moment in the 2023 men's #royalr...	[an, all, mighty, moment, in, the, 2023, men, ...	anallmightymomentinthe2023men'sroyalrumblematch	mighty moment 2023 men 's # royalrumble match !	an all mighti moment in the 2023 men 's # roya...	an all mighty moment in the 2023 men 's # roya...	¡Un momento poderoso en el combate #royalrumbl...	An All Mighty moment in the 2023 Men's #RoyalR...
Outta vhere! 🙂	outta nowhere! 🙂	[outta, nowhere, !, 🙂]	outtanowhere 🙂	outta nowhere ! 🙂	outta nowher ! 🙂	outta nowhere ! 🙂	¡de la nada! 🙂	Outta nowhere! :astonished_face:

Figure 4.1.8 Emoji To Text

Perform the following task without using inbuilt Python Libraries (Wont't work for last two tasks):

```
Original Text: IF YA SMELL.....  @TheRock has come back to #WWERaw!
Lowercased Text: if ya smell.....  @therock has come back to #wweraw!
Tokenized Text: ['IF', 'YA', 'SMELL', 'TheRock', 'has', 'come', 'back', 'to', 'WWERaw']
Cleaned Text: IF YA SMELL  TheRock has come back to WWERaw
Filtered Text: IF YA SMELL..... @TheRock has come back to #WWERaw!
Stemmed Text: IF YA SMEL @The has come back to #WWE
Lemmatized Text: IF YA SMELL..... @TheRock has come back to #WWERaw!

Original Text: Who had the best Instagram photo of the week?!  https://www.wwe.com/gallery/the-25-best-instagram-photos-of-the-week-january-7-2024#fid-40
650941
Lowercased Text: who had the best instagram photo of the week?!  https://www.wwe.com/gallery/the-25-best-instagram-photos-of-the-week-january-7-2024#fid-
40650941
Tokenized Text: ['Who', 'had', 'the', 'best', 'Instagram', 'photo', 'of', 'the', 'week', 'https', 'www', 'wwe', 'com', 'gallery', 'the', '25', 'best', 'i
nstagram', 'photos', 'of', 'the', 'week', 'january', '7', '2024', 'fid', '40650941']
Cleaned Text: Who had the best Instagram photo of the week  httpswwwwwecomgallerythe25bestinstagramphotosoftheweekjanuary72024fid40650941
Filtered Text: Who had best Instagram photo of week?! https://www.wwe.com/gallery/the-25-best-instagram-photos-of-the-week-january-7-2024#fid-40650941
Stemmed Text: Who had the best Inst phot of the week http
Lemmatized Text: Who had the best Instagram photo of the week?! https://www.wwe.com/gallery/the-25-best-instagram-photos-of-the-week-january-7-2024#fid-4
0650941

Original Text: These #RoyalRumble crashers were RUTHLESS!  https://tube.mint.lgbt/VV5fxHfxCE4?si=naZCLWedRVreRISE
Lowercased Text: these #royalrumble crashers were ruthless!  https://tube.mint.lgbt/vv5fxhfxce4?si=nazclwedrvrerise
Tokenized Text: ['These', 'RoyalRumble', 'crashers', 'were', 'RUTHLESS', 'https', 'tube', 'mint', 'lgbt', 'VV5fxHfxCE4', 'si', 'naZCLWedRVreRISE']
Cleaned Text: These RoyalRumble crashers were RUTHLESS  httpstubemintlgbtVV5fxHfxCE4sinaZCLWedRVreRISE
Filtered Text: These #RoyalRumble crashers were RUTHLESS! https://tube.mint.lgbt/VV5fxHfxCE4?si=naZCLWedRVreRISE
Stemmed Text: Thes #Roy cras were RUTH http
Lemmatized Text: These #RoyalRumble crashers were RUTHLESS! https://tube.mint.lgbt/VV5fxHfxCE4?si=naZCLWedRVreRISE

Original Text: An All Mighty moment in the 2023 Men's #RoyalRumble Match!
Lowercased Text: an all mighty moment in the 2023 men's #royalrumble match!
Tokenized Text: ['An', 'All', 'Mighty', 'moment', 'in', 'the', '2023', 'Men', 's', 'RoyalRumble', 'Match']
Cleaned Text: An All Mighty moment in the 2023 Mens RoyalRumble Match
Filtered Text: All Mighty moment in 2023 Men's #RoyalRumble Match!
Stemmed Text: An All Migh mome in the 2023 Men' #Roy Matc
Lemmatized Text: An All Mighty moment in the 2023 Men's #RoyalRumble Match!

Original Text: Outta nowhere! 🙂
Lowercased Text: outta nowhere! 🙂
Tokenized Text: ['Outta', 'nowhere']
Cleaned Text: Outta nowhere 🙂
Filtered Text: Outta nowhere! 🙂
Stemmed Text: Outt nowh 🙂
Lemmatized Text: Outta nowhere! 🙂
```

Figure 4.2 Without Library

Conclusion:

- Lowercasing ensures uniformity, treating uppercase and lowercase forms equally, preventing discrepancies in analysis.
- Tokenization breaks down text into meaningful units, enabling granular analysis at the word level and facilitating various natural language processing tasks.
- Punctuation mark removal eliminates non-alphanumeric characters, reducing noise and focusing on the core meaning of the text.
- Stop word removal improves efficiency by excluding common words, allowing a focus on content-bearing words and enhancing the relevance of analysis.
- Stemming and lemmatization contribute to word form normalization, reducing words to their base form for better consistency and information retrieval.
- Translation enables the understanding of text in different languages, fostering cross-language communication and analysis.
- Emoji-to-text conversion aids in extracting emotional context from textual data, contributing to sentiment analysis and understanding user expressions.

Lab No: 5

Aim:

The aim is to employ Logistic Regression and Term Frequency-Inverse Document Frequency (TF-IDF) for spam classification, comparing accuracies across datasets to identify the most effective preprocessing technique.

Description:

Perform the following task with using inbuilt Python Libraries:

- Feature Extraction: Utilize TF-IDF vectorization to convert text data into numerical features.
- Data Splitting: Divide the datasets into 80% training and 20% testing subsets.
- Model Training: Train Logistic Regression models on the training data for each dataset.
- Prediction: Evaluate model performance by predicting labels on the testing sets.
- Accuracy Assessment: Calculate and compare accuracies to identify the most effective preprocessing technique among datasets.
- End Result: Determine which dataset, whether raw or preprocessed, yields the highest accuracy with Logistic Regression and TF-IDF.

Source Code:

```
# The aim is to employ Logistic Regression and Term Frequency-Inverse Document Frequency (TF-IDF) for spam classification, comparing accuracies across datasets to identify the most effective preprocessing technique.

## Perform the following task with using inbuilt Python Libraries: acy.

#### - Perform Classification of data sets (Data 1 (Raw Data), Data 2 (Data with Lowercase), ,..... Data n ) using Logistic Regression.
#### - Use Tf-Idf vectorizor for Feature Extraction.
#### - Use 80% of data for training and 20% of data for testing.
#### - Check the accuracy of the model for each dataset.
#### - Write conclusion for with data  (Data 1 (Raw Data), Data 2 (Data with Lowercase), ,..... Data n ), the Logistic Regression provides best accuracy.

import pandas as pd
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
import string
import re
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
```

```
data = pd.read_csv("spam.csv", encoding="ISO-8859-1")
data.head()

data = data.drop(columns=["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"])
data.head()

df = pd.DataFrame(data)
df.head()
```

```python
data_columns = ['v2', 'lowercased_v2', 'tokens', 'cleaned_v2', 'filtered_v2', 'stemmed_v2', 'lemmatized_v2']
accuracies = []
precisions = []
recalls = []
f1s = []
for column in data_columns:
    X = df[column].astype(str)
    y = df['v1']
    # Training
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    tfidf_vectorizer = TfidfVectorizer()
    X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
    X_test_tfidf = tfidf_vectorizer.transform(X_test)
    model = LogisticRegression()
    model.fit(X_train_tfidf, y_train)
    # Prediction
    y_pred = model.predict(X_test_tfidf)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, pos_label='spam')
    recall = recall_score(y_test, y_pred, pos_label='spam')
    f1 = f1_score(y_test, y_pred, pos_label='spam')
    conf_matrix = confusion_matrix(y_test, y_pred)

    print(f"\nResults for {column}:")
    print(f"Accuracy: {accuracy:.4f}")
    print(f"Precision: {precision:.4f}")
    print(f"Recall: {recall:.4f}")
    print(f"F1 Score: {f1:.4f}")
    print(f"Confusion Matrix:\n{conf_matrix}")
    accuracies.append(accuracy)
    precisions.append(precision)
    recalls.append(recall)
```

```python
    f1s.append(f1);
# Conclusion
best_index = accuracies.index(max(accuracies))
best_dataset = data_columns[best_index]
print(f'\nLogistic Regression provides the best accuracy with {best_dataset} having accuracy of {accuracies[best_index]}.')
best_index = precisions.index(max(precisions))
best_dataset = data_columns[best_index]
print(f'Logistic Regression provides the best precision with {best_dataset} having precision of {precisions[best_index]}.')
best_index = recalls.index(max(recalls))
best_dataset = data_columns[best_index]
print(f'Logistic Regression provides the best recall with {best_dataset} having recall of {recalls[best_index]}.')
best_index = f1s.index(max(f1s))
best_dataset = data_columns[best_index]
print(f'Logistic Regression provides the best F1 score with {best_dataset} having F1 score of {f1s[best_index]}.')
```

Output:

Spam Data:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

Figure 5.0.1 Spam Data

After Dropping Unnecessary Columns:

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

Figure 5.0.2 After Dropping Columns

After Applying Preprocessing Techniques:

	v1	v2	lowercased_v2	tokens	cleaned_v2	filtered_v2	stemmed_v2	lemmatized_v2
0	ham	Go until jurong point, crazy.. Available only ...	go until jurong point, crazy.. available only ...	[go, until, jurong, point, ,, crazy, .., avail...	gountiljurongpointcrazy..availableonlyinbugisn...	go jurong point , crazy .. available bugis n g...	go until jurong point , crazi .. avail onli in...	go until jurong point , crazy .. available onl...
1	ham	Ok lar... Joking wif u oni...	ok lar... joking wif u oni...	[ok, lar, ..., joking, wif, u, oni, ...]	oklar...jokingwifuoni...	ok lar ... joking wif u oni ...	ok lar ... joke wif u oni ...	ok lar ... joking wif u oni ...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	free entry in 2 a wkly comp to win fa cup fina...	[free, entry, in, 2, a, wkly, comp, to, win, f...	freeentryin2awklycomptowinfacupfinaltkts21stma...	free entry 2 wkly comp win fa cup final tkts 2...	free entri in 2 a wkli comp to win fa cup fina...	free entry in 2 a wkly comp to win fa cup fina...
3	ham	U dun say so early hor... U c already then say...	u dun say so early hor... u c already then say...	[u, dun, say, so, early, hor, ..., u, c, alrea...	udunsaysoearlyhor...ucalreadythensay...	u dun say early hor ... u c already say ...	u dun say so earli hor ... u c alreadi then sa...	u dun say so early hor ... u c already then sa...
4	ham	Nah I don't think he goes to usf, he lives aro...	nah i don't think he goes to usf, he lives aro...	[nah, i, do, n't, think, he, goes, to, usf, ,,...	nahidon'tthinkhegoestousfhelivesaroundherethough	nah n't think goes usf , lives around though	nah i do n't think he goe to usf , he live aro...	nah i do n't think he go to usf , he life arou...

Figure 5.0.3 After Applying Preprocessing Techniques

Output:

```
Results for v2:                    Results for cleaned_v2:
Accuracy: 0.9659                   Accuracy: 0.8834
Precision: 0.9912                  Precision: 1.0000
Recall: 0.7533                     Recall: 0.1333
F1 Score: 0.8561                   F1 Score: 0.2353
Confusion Matrix:                  Confusion Matrix:
[[964   1]                         [[965   0]
 [ 37 113]]                         [130  20]]

Results for lowercased_v2:         Results for filtered_v2:
Accuracy: 0.9659                   Accuracy: 0.9578
Precision: 0.9912                  Precision: 0.9558
Recall: 0.7533                     Recall: 0.7200
F1 Score: 0.8561                   F1 Score: 0.8213
Confusion Matrix:                  Confusion Matrix:
[[964   1]                         [[960   5]
 [ 37 113]]                         [ 42 108]]

Results for tokens:                Results for stemmed_v2:
Accuracy: 0.9686                   Accuracy: 0.9668
Precision: 0.9915                  Precision: 0.9829
Recall: 0.7733                     Recall: 0.7667
F1 Score: 0.8689                   F1 Score: 0.8614
Confusion Matrix:                  Confusion Matrix:
[[964   1]                         [[963   2]
 [ 34 116]]                         [ 35 115]]
```

```
Results for lemmatized_v2:
Accuracy: 0.9677
Precision: 0.9914
Recall: 0.7667
F1 Score: 0.8647
Confusion Matrix:
[[964    1]
 [ 35 115]]
```

```
Logistic Regression provides the best accuracy with tokens having accuracy of 0.968609865470852.
Logistic Regression provides the best precision with cleaned_v2 having precision of 1.0.
Logistic Regression provides the best recall with tokens having recall of 0.7733333333333333.
Logistic Regression provides the best F1 score with tokens having F1 score of 0.8689138576779025.
```

Figure 5.1 Output

Conclusion:

- Applied preprocessing steps including lowercasing, tokenization, cleaning, filtering, stemming, and lemmatization to enhance text data quality.
- Utilized Tf-Idf vectorization for feature extraction, capturing term importance in each dataset.
- Trained Logistic Regression models on each preprocessed dataset using 80% of data for training and 20% for testing.
- Evaluated model accuracy for each dataset, measuring performance on spam classification.
- Identified the dataset with the highest accuracy, indicating that Logistic Regression performs best on token generation approach.

Lab No: 6

<u>**Aim:**</u>

Classification of Handwritten Digits using Artificial Neural Networks

<u>**Description:**</u>

Perform the following task with using inbuilt Python Libraries:
1. Search relevant datasets to perform classification (MNIST dataset)
2. Classify handwritten digits using a simple neural network that has only input and output layers.
3. Add a hidden layer and see how the performance of the model improves.
4. Apply various activation functions to hidden and output layers to assess the model performance.
5. Apply various cost functions to measure the error of the model.

Activation Functions:
1. Softmax: Output layer for multi-class classification probabilities.
2. Sigmoid: Binary classification activation for output layer probabilities.
3. Tanh: Symmetric activation for hidden layers, mapping values to [-1, 1].
4. ReLU: Rectified Linear Unit, popular for hidden layer activations.

Cost/Loss Functions:
1. Mean Squared Error: Measures squared difference between predicted and actual values.
2. Categorical Crossentropy: Ideal for multi-class classification tasks, penalizing class probability deviations.
3. Binary Crossentropy: Suited for binary classification problems, optimizing log-likelihood of true labels.
4. Poisson: Used for count data; models distribution of rare events.

<u>**Source Code:**</u>

Classification of Handwritten Digits using Artificial Neural Networks

Perform the following steps for above mentioned problem statement:

1. Search relevant datasets to perform classification

```python
from keras.datasets import mnist
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()

print(test_images)

print(test_labels)
```

2. Classify handwritten digits using a simple neural network that has only input and output layers.

```python
from keras.models import Sequential
from keras.layers import Flatten, Dense
from keras.utils import to_categorical

train_images = train_images.reshape((60000, 28 * 28)).astype('float32') / 255
test_images = test_images.reshape((10000, 28 * 28)).astype('float32') / 255
train_labels = to_categorical(train_labels)
test_labels = to_categorical(test_labels)

model = Sequential()
model.add(Dense(10, activation='softmax', input_shape=(28 * 28,)))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5, batch_size=25, validation_split=0.2)
evaluation = model.evaluate(test_images, test_labels)
print(f"Test Accuracy: {evaluation[1]*100:.2f}%")
print(f"Test Loss: {evaluation[0]}")
```

3. Add a hidden layer and see how the performance of the model improves.

```python
model = Sequential()
model.add(Dense(100, activation='relu', input_shape=(28 * 28,)))
model.add(Dense(10, activation='softmax'))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(train_images, train_labels, epochs=5, batch_size=25, validation_split=0.2)
evaluation = model.evaluate(test_images, test_labels)
print(f"Test Accuracy: {evaluation[1]*100:.2f}%")
print(f"Test Loss: {evaluation[0]}")
```

4. Apply various activation functions to hidden and output layers to assess the model performance.

```python
activation_functions = ['sigmoid', 'tanh', 'relu']
for activation_function in activation_functions:
    model = Sequential()
    model.add(Dense(100, activation='relu', input_shape=(28 * 28,)))
    model.add(Dense(10, activation= activation_function))
    model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
    model.fit(train_images, train_labels, epochs=5, batch_size=25, validation_split=0.2)
    evaluation = model.evaluate(test_images, test_labels)
    print(f"Test Accuracy with {loss_function}: {evaluation[1]*100:.2f}%")
    print(f"Test Loss with {loss_function}: {evaluation[0]}")
```

5. Apply various cost functions to measure the error of the model.

```python
loss_functions = ['mean_squared_error', 'categorical_crossentropy', 'binary_crossentropy', 'poisson']
for loss_function in loss_functions:
    print(f"\nUsing {loss_function} as loss function:")
    model.compile(optimizer='adam', loss=loss_function, metrics=['accuracy'])
    model.fit(train_images, train_labels, epochs=5, batch_size=25, validation_split=0.2)
    evaluation = model.evaluate(test_images, test_labels)
    print(f"Test Accuracy: {evaluation[1]*100:.2f}%")
    print(f"Test Loss: {evaluation[0]}")
```
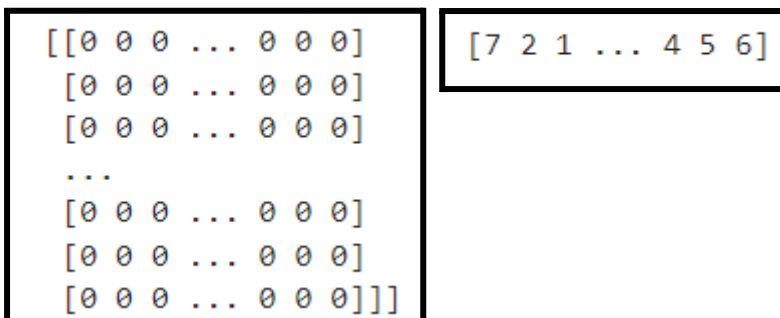
Output:

Task 1:

Figure 5.1 Test Images and Test Label

Task 2:

```
Test Accuracy: 92.47%
Test Loss: 0.27226367592811584
```

Figure 5.2 Without Hidden Layer

Task 3:

```
Test Accuracy: 97.57%
Test Loss: 0.08260300010442734
```

Figure 5.3 With Hidden Layer

Task 4:

```
Test Accuracy with sigmoid: 97.66%
Test Loss with sigmoid: 0.07542455196380615
```

```
Test Accuracy with tanh: 9.80%
Test Loss with tanh: nan
```

```
Test Accuracy with relu: 9.80%
Test Loss with relu: nan
```

Figure 5.4 Activation Functions

Task 5:

```
Test Accuracy with mean_squared_error: 97.20%
Test Loss with mean_squared_error: 0.004385668784379959
```

```
Test Accuracy with categorical_crossentropy: 97.77%
Test Loss with categorical_crossentropy: 0.07678196579217911
```

```
Test Accuracy with binary_crossentropy: 98.04%
Test Loss with binary_crossentropy: 0.017176324501633644
```

```
Test Accuracy with poisson: 97.85%
Test Loss with poisson: 0.11014503240585327
```

Figure 5.5 Cost/Loss Function

Conclusion:

- The initial model achieves a baseline performance (92.47%) for handwritten digit classification.
- The inclusion of a hidden layer improves the model's ability having accuracy of 97.57%.
- The model's performance with "Sigmoid" activation function is the highest (97.66%).
- "Categorical cross entropy" is commonly used for classification tasks, but the experimentation with other loss functions provides a better loss function ("Binary cross entropy") with accuracy of 98.04%.

GovUnityXplorer

Trust, Privacy, and Empowerment

Overview:

GovUnityXplorer aims to develop a robust digital governance platform leveraging Web3 technologies. This platform facilitates decentralized communication channels between citizens and government entities, fostering transparent policy discussions and public decision-making processes. AI-driven sentiment analysis assists policymakers in understanding community needs, while machine learning models enhance conflict resolution and identity verification capabilities, all accessible through intuitive front-end interfaces and Web3 features like no-cost voting and create-to-earn models.

Problem Statement:

- Need for a comprehensive digital governance platform leveraging Web3 technologies.
- Aims to bridge the communication gap between citizens and government institutions.
- Seeks transparency, accessibility, and inclusivity in public decision-making processes.
- Empowers individuals to actively participate in shaping policies affecting their lives.
- Integrates AI-driven sentiment analysis for informed policymaking reflecting community sentiments.
- Challenges in resolving community conflicts and legal disputes addressed.
- Development of machine learning models for conflict resolution and identity verification.
- Incorporates features like Policy Thread and Issue Thread for collaborative governance.
- Utilizes Web3 technologies for social media integration, cost-free voting, and create-to-earn model.
- Ensures accessibility and inclusivity irrespective of socio-economic barriers.

Challenge Overcoming:

- Addresses need for inclusive, transparent, and efficient governance system.
- Traditional models suffer from centralized communication and limited participation opportunities.
- Lack of effective mechanisms for conflict resolution and sentiment analysis.
- Results in disenchantment among citizens and unresolved disputes.
- Project leverages Web3 technologies for decentralized digital governance.

- Decentralizes communication, integrates AI-driven analysis, and implements innovative features.
- Aims to empower citizens for active participation and collaborative decision-making.
- Promotes transparency, inclusivity, and trust in the governance process.
- Ensures policies are informed by genuine community needs and sentiments.

UN Goals and Targets Aligned with the Project:

- **Goal 16: Peace, Justice, and Strong Institutions**
  - Target 16.6: Develop effective, accountable, and transparent institutions at all levels.
- **Goal 9: Industry, Innovation, and Infrastructure**
  - Target 9.5: Enhance scientific research, upgrade technological capabilities, and promote innovation.
- **Goal 10: Reduced Inequalities**
  - Target 10.2: Empower and promote the social, economic, and political inclusion of all, irrespective of age, sex, disability, race, ethnicity, origin, religion, or economic or other status.

Flow Chart (Containing only 3 components of the total 17 components)
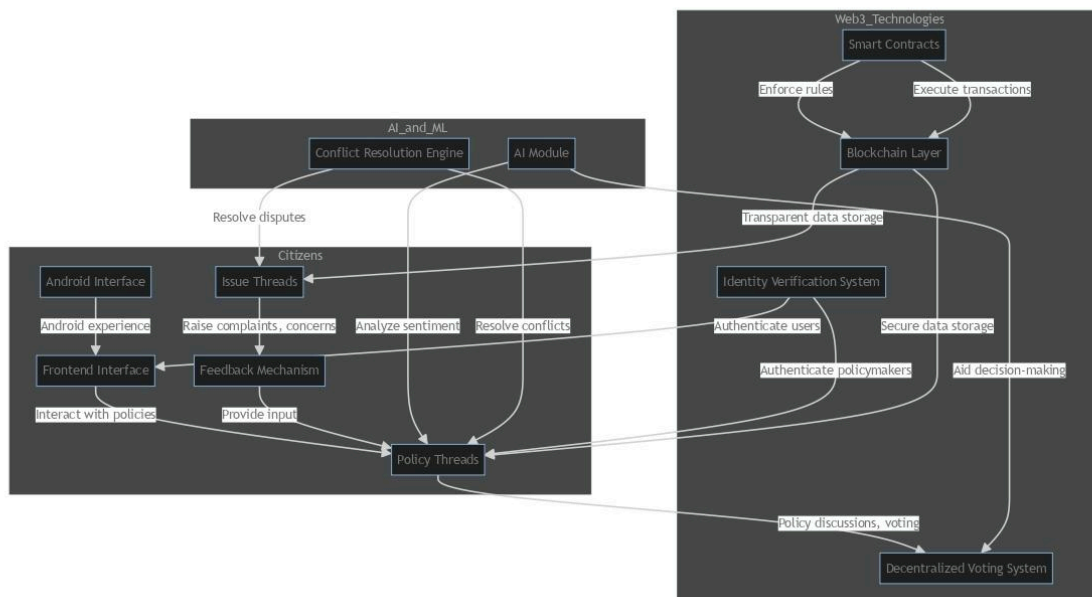
Task List

Frontend Development:

- ❌ Develop user interfaces for citizens and policymakers.
  - ✅ Implement UI for Forum.

- ✅ Implement UI for Accounts.
- ❌ Implement UI for Transaction.

Web3 Integration:

- ❌ Integrate blockchain for decentralized communication and transactions.
  - ✅ Research Ethereum integration for smart contracts.
  - ✅ Implement Ethereum blockchain integration.
  - ❌ Test and ensure seamless interaction with the platform.
- ❌ WEB3 (DID, Social Media, no cost Voting, create-to-earn model)
  - ✅ Integrate DID (Decentralized Identifiers) for user authentication.
  - ✅ Implement social media integration for user engagement.
  - ❌ Integrate no-cost voting mechanism.
  - ❌ Implement create-to-earn model for incentivizing participation.

Mobile Application Development:

- ❌ Develop a mobile application using Flutter for Android devices.
  - ✅ Implement Mobile UI for Forum.
  - ✅ Implement Mobile UI for Accounts.
  - ❌ Implement Mobile UI for Transaction.
  - ❌ Compatibility Test across various Android devices.

AI/ML Module Implementation:

- ❌ Develop AI module for sentiment analysis and decision support.
  - ✅ Research TensorFlow for machine learning capabilities.
  - ✅ Implement sentiment analysis algorithms.
  - ❌ Integrate AI module with the platform's backend.
- ❌ Develop ML models for conflict resolution.
  - ✅ Collect and preprocess conflict data.
  - ✅ Train ML model.
  - ❌ Improve accuracy.
  - ❌ Write scripts for integration.
- ❌ Develop ML models for identity verification.
  - ✅ Create identity datasets.
  - ✅ Train identity verification model.
  - ❌ Script integration with Dapp.

Platform Features Development:

- ❌ Implement Policy Threads for policy discussions, voting, and feedback.
  - ✅ Develop backend logic for policy threads.
  - ❌ Implement frontend components for policy threads.
- ❌ Implement Issue Threads for citizen complaints and concerns.
  - ❌ Develop backend logic for issue threads.

- ❌ Implement frontend components for issue threads.
- ❌ Implement feedback mechanism for citizen input on policies and services.
  - ❌ Develop backend logic for feedback submissions.
  - ❌ Implement frontend components for feedback mechanism.

Blockchain Layer Development:
- ❌ Deploy a Dapp on Testnet or local blockchain.
  - ✅ Set up accounts on Testnet.
  - ❌ Configure Truffle for local blockchain testing.
  - ❌ Ensure smart contracts deploy successfully.
- ❌ Implement immutable ledger for transparent and secure data storage.
  - ✅ Research Hyperledger Fabric for blockchain architecture.
  - ✅ Develop smart contracts using Solidity for Ethereum.
  - ❌ Test and deploy smart contracts on the blockchain.

Identity Verification System:
- ❌ Develop system for secure and decentralized identity verification.
  - ✅ Research Civic for identity verification solutions.
  - ❌ Integrate identity verification system with user registration process.
  - ❌ Test and ensure reliability and security of the verification process.

Analytics Dashboard Development:
- ❌ Develop analytics dashboard for visualizing data and insights.
  - ✅ Research Tableau for data visualization capabilities.
  - ✅ Design dashboard layout and components.
  - ❌ Implement backend logic for data retrieval and visualization.

Database Layer Implementation:
- ❌ Design and implement database schema for storing user data, policies, complaints, and feedback.
  - ✅ Research PostgreSQL for relational data storage.
  - ✅ Develop backend logic for database interactions.
  - ❌ Test database operations and ensure data integrity and security.

Smart Contracts Deployment:
- ❌ Deploy smart contracts on the blockchain network.
  - ❌ Ensure proper configuration and deployment of smart contracts.
  - ❌ Test smart contract functionality and interactions with the platform.
  - ❌ Monitor smart contract performance and address any issues.

Integration Testing:
- ❌ Conduct integration testing to ensure seamless interaction between platform components.

- ❌ Test communication between frontend and backend modules.
- ❌ Verify data flow and consistency across the platform.
- ❌ Address any integration issues and ensure platform stability.

User Acceptance Testing:
- ❌ Invite users to participate in user acceptance testing.
  - ❌ Gather feedback and suggestions from users.
  - ❌ Identify and address any usability issues or bugs.
  - ❌ Ensure that the platform meets user expectations and requirements.

Documentation and Training:
- ❌ Include in README (libraries to install) and (steps to run).
  - ❌ Document required libraries and dependencies.
  - ❌ Provide detailed steps to deploy and run the Dapp.
- ❌ Prepare documentation for platform usage and administration.
  - ❌ Document platform features, functionalities, and workflows.
  - ❌ Provide user guides and tutorials for citizens and policymakers.
- ❌ Conduct training sessions for platform users and administrators.
  - ❌ Train users on how to navigate and use the platform effectively.
  - ❌ Provide training materials and resources for ongoing support.
- ❌ Post Announcement with social links
  - ✅ Draft announcement message.
  - ✅ Share announcement on social media platforms.
- ❌ Create banner using project screenshots
  - ✅ Capture project screenshots.
  - ❌ Design banner using screenshots.

Deployment:
- ❌ Prepare for platform deployment to production environment.
  - ❌ Configure servers or cloud instances for hosting the platform.
  - ❌ Set up necessary infrastructure and dependencies.
- ❌ Deploy the platform to the production environment.
  - ❌ Ensure smooth deployment process with minimal downtime.
  - ❌ Conduct final testing and validation in the production environment.

Post-Deployment Monitoring and Support:
- ❌ Monitor platform performance and user activity post-deployment.
  - ❌ Set up monitoring tools to track platform usage and performance metrics.
  - ❌ Address any issues or performance bottlenecks as they arise.
- ❌ Provide ongoing support and maintenance for the platform.
  - ❌ Respond to user inquiries and troubleshoot any technical issues.

- – ❌ Implement updates and enhancements based on user feedback and evolving requirements.

Inspirations:

In India, some welfare initiatives meant to benefit the needy are often plagued by corruption, where money meant for the poor ends up in the wrong hands. Our project aims to tackle such injustices by creating a transparent digital governance platform where citizens can directly engage with policymakers, ensuring that resources reach those who truly need them, thus promoting fairness and accountability in our society.

Architecture for the Project:
1. Frontend Interface: User interaction layer for citizens and policymakers.
2. Web3 Integration: Incorporating blockchain for decentralized communication and transactions.
3. Android Interface: Mobile application developed using Flutter for seamless user experience on Android devices.
4. AI Module: Analyzing public sentiment and assisting policymakers in decision-making.
5. Policy Threads: Platform section for policy discussions, voting, and feedback.
6. Issue Threads: Platform section for citizens to raise complaints and concerns.
7. Blockchain Layer: Immutable ledger for transparent and secure data storage.
8. Identity Verification System: Ensuring authenticity of users and policymakers.
9. Decentralized Voting System: Allowing citizens to participate in decision-making processes.
10. Conflict Resolution Engine: ML-powered mechanism for resolving disputes and conflicts.
11. Feedback Mechanism: System for citizens to provide input on policies and services.
12. Analytics Dashboard: Visualizing data and insights for policymakers and administrators.
13. Database Layer: Storing user data, policies, complaints, and feedback securely.
14. Smart Contracts: Executing transactions and enforcing rules on the blockchain.

Implementation Products:
1. Frontend Interface: React.js for flexibility and rich user interfaces.
2. Web3 Integration: Ethereum for its widespread adoption and developer support.
3. Android Interface: Flutter for cross-platform development and fast performance.
4. AI Module: TensorFlow for robust machine learning capabilities.
5. Policy Threads: Django for rapid development and security.
6. Issue Threads: Node.js for real-time communication and scalability.
7. Blockchain Layer: Hyperledger Fabric for permissioned blockchain architecture.
8. Identity Verification System: Civic for secure and decentralized identity verification.
9. Decentralized Voting System: Tezos for on-chain governance and smart contracts.

10. Conflict Resolution Engine: PyTorch for deep learning flexibility and performance.
11. Feedback Mechanism: MongoDB for flexible document storage and scalability.
12. Analytics Dashboard: Tableau for powerful data visualization capabilities.
13. Database Layer: PostgreSQL for relational data storage and ACID compliance.
14. Smart Contracts: Solidity for Ethereum smart contract development.

Feedback Points:
1. Gas fee for providing feedback or issuing a complaint is high.
2. Concerns about the possibility of purchasing votes.
3. Static voting price regardless of the significance of the issue.

Implemented Improvements:
1. Provided 5 voting rights per individual from the government's pocket to address high gas fees, promoting more participation.
2. Temporarily banned account creation to prevent vote purchasing and maintain the integrity of the voting process.
3. Introduced dynamic voting prices based on the significance of the issue, implementing upvote and downvote options with varying and opposite prices to ensure fair and proportional engagement.

One Challenge:

One challenge I encountered was the integration issue when addressing the high gas fees for transactions involving feedback submission or issuing complaints. Initially, the Ethereum network's gas fees posed a significant barrier for users, especially in regions where transaction costs could be prohibitive. To tackle this, I optimized smart contracts to minimize gas usage, employing efficient coding practices and reducing unnecessary computation. Secondly, I leveraged dynamic gas fee estimation algorithms to recommend optimal gas prices based on current network conditions, ensuring that users could submit feedback or complaints at reasonable costs.

Impact:

GovUnityXplorer significantly addresses inefficiency and transparency issues in governance, fostering communication between citizens and government, facilitating feedback, discussions, and decision-making. AI-driven sentiment analysis and ML models empower policymakers to align decisions with community needs. Quantifiable metrics like engagement levels and reduced transaction costs demonstrate impact. Analytics dashboards, surveys, and feedback mechanisms provide insights. Overall, the project enhances transparency, inclusivity, and decision-making processes, evidenced by increased engagement and reduced costs.

Future Steps:
• Explore interoperability with other blockchain networks.
• Conduct awareness campaigns and partnerships.

- Optimize infrastructure for scalability.

Technical Architecture Support for Scaling:
- Replicate and deploy components across multiple servers or cloud instances.
- Utilize message queues for asynchronous communication between components.

Tools and Technologies:

Machine Learning for Conflict Resolution (Component 1):
- Languages:
  - Python (for machine learning model development)
- Tools:
  - Scikit-learn/TensorFlow for machine learning algorithms
  - Jupyter Notebooks for experimentation and model development

Digital Governance Ecosystem (Component 2):
- Languages:
  - JavaScript (for frontend development)
  - Node.js (for backend development)
  - Solidity/Func (for smart contract development on the blockchain)
- Tools:
  - React.js or Angular for frontend UI
  - Express.js for backend API
  - Web3.js for interacting with the blockchain
  - Hardhat for local blockchain development

Decentralized Identity Verification System (Component 3):
- Languages:
  - Solidity/Func (for smart contract development on the blockchain)
  - Python (for machine learning integration)
- Tools:
  - OpenZeppelin for secure smart contract development
  - TensorFlow or PyTorch for machine learning integration
  - Hardhat for local blockchain development

Common Tools for Integration:
- Databases:
  - MongoDB or PostgreSQL for storing application and user data
- Communication and Integration:
  - RESTful APIs for communication between different components
  - GraphQL for flexible data queries
- Security:
  - SSL/TLS for secure communication

– Ethlint for Ethereum smart contract code linting

Development Workflow Tools:
- Version Control:
    – Git for version control
    – GitHub for collaborative development
- Project Management:
    – Github Projects for project tracking and management
- Containerization:
    – Docker for containerization of application components

Testing and Quality Assurance:
- Testing Frameworks:
    – Jest for JavaScript/Node.js testing
    – Truffle for Ethereum smart contract testing
- Continuous Integration/Continuous Deployment (CI/CD):
    – GitHub Actions for automated testing and deployment

Reference:

- ML Conflict:
  https://www.sciencedirect.com/science/article/abs/pii/016792369390034Z

- Decentralized Identity: https://www.dock.io/post/decentralized-identity

- Governance Singularity:
  https://blog.oceanprotocol.com/making-ocean-protocols-smart-contracts-and-it-s-governance-unstoppable-45cf99dc1b65

- Conflict Resolution Strategies In Artificial Intelligence.
  https://conflictresolved.com/conflict-resolution-strategies-in-artificial-intelligence/
  .

- Artificial Intelligence Techniques for Conflict Resolution.
  https://link.springer.com/article/10.1007/s10726-021-09738-x.

- Machine Learning and Conflict Prediction: A Use Case.
  https://stabilityjournal.org/articles/10.5334/sta.cr.

- Using Artificial Intelligence to provide Intelligent Dispute Resolution ….
  https://link.springer.com/article/10.1007/s10726-021-09734-1.

- Machine Learning-based SON function conflict resolution | IEEE ….
  https://ieeexplore.ieee.org/document/8969675/.

- Decentralized identity | ethereum.org. https://ethereum.org/decentralized-identity.

- Machine Learning in Digital Identity Verification - emudhra.com. https://emudhra.com/blog/the-role-of-machine-learning-in-digital-identity-verification.

- Responsible use of machine learning to verify identities at scale. https://venturebeat.com/ai/responsible-use-of-machine-learning-to-verify-identities-at-scale/.

- How Machine Learning is Reinventing Identity Verification. https://www.socure.com/blog/how-machine-learning-is-reinventing-identity-verification.

- How Artificial Intelligence is taking ID verification to the next level?. https://shuftipro.com/blog/how-artificial-intelligence-is-taking-id-verification-to-the-next-level/.

- Identity Verification | Machine Learning | Amazon Web Services. https://aws.amazon.com/machine-learning/ml-use-cases/identity-verification/.

- What Is Decentralized Identity? A Comprehensive Guide. https://www.identity.com/decentralized-identity/.

- Blockchain for Digital Identity Verification | Reintech media. https://reintech.io/blog/blockchain-digital-identity-verification.

- What is Web3 technology (and why is it important)? | McKinsey. https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-web3.

- Web3 - Wikipedia. https://en.wikipedia.org/wiki/Web3.

- Top 15 sentiment analysis tools to consider in 2024. https://sproutsocial.com/insights/sentiment-analysis-tools/.

- Top 10 Web3 Applications You Must Know - 101 Blockchains. https://101blockchains.com/top-web3-applications/.

- What Is Web3? - Harvard Business Review. https://hbr.org/2022/05/what-is-web3.

- Ultimate Guide: What is Web3 Technology? - Moralis. https://moralis.io/ultimate-guide-what-is-web3-technology/.

- The Impact of AI Sentiment Analysis: Benefits and Use Cases. https://appinventiv.com/blog/ai-sentiment-analysis-in-business/.

- The 14 Best AI Sentiment Analysis Tools [2024] - Brand24 Blog. https://brand24.com/blog/best-sentiment-analysis-tools/.

- The Power of Sentiment Analysis in AI | by Humans.ai - Medium. https://medium.com/humansdotai/the-power-of-sentiment-analysis-in-ai-43a7c3cfe1ba.

- Top 9 Best AI-Driven Sentiment Analysis Tools in 2024 | Infraon. https://infraon.io/blog/ai-driven-sentiment-analysis-tools/.

- Designing the Future of Platform Governance with Web3 - Best. https://blog.trustkeys.network/the-future-of-platform-governance-with-web3/.

- Decentralized Governance in Web3: Beyond DAOs - Medium. https://blog.blockmagnates.com/decentralized-governance-in-web3-beyond-daos-b68b9e859ec1.

- What is Web3 and how could it change the internet? - The World Economic …. https://www.weforum.org/agenda/2023/03/what-is-web3-and-how-could-it-change-the-internet/.

I Introduction

1 Project Overview

In today's legal landscape, the ability to predict Supreme Court decisions accurately holds immense significance. We introduce lawGov, a groundbreaking Natural Language Processing (NLP) application designed to predict the ideological direction of Supreme Court case decisions. Our goal is to provide valuable insights to parties involved in upcoming cases, enabling them to assess their prospects for success and plan their litigation strategies accordingly.

2 The Purpose of the Project

2a The Importance of Solving the Identified Problem

The ability to predict legal judgments with precision is crucial for several reasons. First and foremost, it enables legal professionals to make informed decisions and devise effective strategies for their clients. By having insights into potential outcomes, lawyers can better advise their clients, manage expectations, and optimize resources.

Furthermore, accurate legal predictions can enhance access to justice. Many individuals and businesses are deterred from pursuing legal action due to uncertainty regarding the outcome and the associated costs. lawGov can help mitigate these concerns by providing clarity on the likely outcome of a case, thereby promoting fairness and transparency within the legal system.

Moreover, the efficiency gains offered by lawGov are significant. Legal research and analysis are time-consuming tasks that often involve sifting through mountains of documents. By automating this process and providing accurate predictions, lawGov enables legal professionals to focus their time and energy on more strategic aspects of their work, ultimately improving productivity and reducing costs.

2b Goals of the Project (UN Goals)

Goal 16: Peace, Justice, and Strong Institutions

Target 16.6: Develop effective, accountable, and transparent institutions at all levels.

Goal 9: Industry, Innovation, and Infrastructure

Target 9.5: Enhance scientific research, upgrade technological capabilities, and promote innovation.

Goal 10: Reduced Inequalities

Target 10.2: Empower and promote the social, economic, and political inclusion of all, irrespective of age, sex, disability, race, ethnicity, origin, religion, or economic or other status.

3 The Domain of the Project

The legal domain is complex, multifaceted, and often fraught with uncertainty. Legal professionals face the daunting task of analyzing vast amounts of information, including past cases, legal precedents, statutes, and regulations, to make informed decisions. However, even the most seasoned legal experts can struggle with predicting the outcome of a case accurately. This is where lawGov steps in, utilizing advanced NLP algorithms to analyze and interpret legal data swiftly and accurately.

3a Methodology

Modeling the Court as a Whole:

This approach involves predicting the single outcome of a case based on the generally nine justices' votes.

Ensemble Method:

Here, we separately model the decisions of individual justices in probability terms and then aggregate these probabilities to predict the outcome of the case.

4 Existing Solutions

Existing techniques and solutions for legal prediction encompass various approaches, including machine learning (ML) and artificial intelligence (AI) methodologies. These techniques leverage advanced algorithms to improve legal decision-making, with ML achieving up to 90% prediction accuracy (theoretical and case study), and AI-based systems providing automated mediation and early warning systems for conflicts.

Machine Learning-based SON Function Conflict Resolution

Sarker et al. used reinforcement learning and clustering algorithms to dynamically optimize SON functions, enhancing network performance and resource allocation.

Conflict Resolution Strategies in AI

Game theory and multi-agent systems are used to model and analyze conflicts, aiding in strategic decision-making and resource optimization.

Artificial Intelligence Techniques for Conflict Resolution

Noorian et al. employ neural networks and fuzzy logic, enabling adaptive decision-making and efficiency gains.

Using Artificial Intelligence to Provide Intelligent Dispute Resolution

Lin et al. apply NLP and case-based reasoning, resulting in automated mediation and consistent, fair outcomes.

5 Proposed Solution

We conducted 8 key experiments on a dataset. Data preprocessing included removing stopwords, lowercasing, stemming, and cleaning non-alphabet characters. Anonymization replaced party names with _PARTY_, and class imbalance was addressed. With an 80:20 train-test split, we used 4-fold cross-validation. After training, we achieved an average testing accuracy of 85%.

5a Model Evaluation Summary

TF-IDF:

Among combinations 3 and 4, the fourth model of the third combination achieved the highest testing accuracy of 0.972 and a testing loss of 0.141.

GloVe:

Combination 2 yielded the best results, with the first model achieving a testing accuracy of 0.916 and a testing loss of 0.384.

Doc2Vec:

The second model of combination 5 outperformed others, achieving a testing accuracy of 0.945 and a testing loss of 0.282.

CNN:

For combinations 2 and 5, the second model of the second combination had the highest testing accuracy of 0.933 and a testing loss of 0.325.

Overall, TF-IDF with combination 3's fourth model demonstrated the best performance, achieving the highest testing accuracy of 0.972, while GloVe's best model in combination 2 came close with an accuracy of 0.916. Doc2Vec's top model and CNN's top model also showed competitive performance, with accuracies of 0.945 and 0.933, respectively.

II Existing Techniques and Solutions for Legal Prediction

1 Machine Learning-based SON Function Conflict Resolution (Reference: [1])

The study by Sarker et al. focuses on using machine learning techniques for resolving conflicts in Self-Organizing Network (SON) functions, with implications for telecommunications and beyond.

1a Techniques Used

Reinforcement Learning

Sarker et al. employ reinforcement learning algorithms to optimize SON function allocation and resolve conflicts between network elements.

Clustering Algorithms

Clustering techniques are used to group network elements based on their characteristics, facilitating conflict resolution strategies.

1b Key Findings

Dynamic Optimization

ML-based conflict resolution enables dynamic optimization of SON functions, adapting to changing network conditions in real-time.

Resource Allocation

By efficiently allocating resources and mitigating conflicts, ML models improve network performance and reliability.

2 Conflict Resolution Strategies in AI (Reference: [2])

The article on Conflict Resolution Strategies in Artificial Intelligence explores various AI-based approaches to resolving conflicts, including those encountered in legal contexts.

2a Techniques Used

Game Theory

Game theoretic approaches are employed to model and analyze conflicts, considering the strategic interactions between parties involved.

Multi-Agent Systems

AI techniques such as multi-agent systems are utilized to simulate and study complex interactions between legal entities, facilitating conflict resolution.

2b Key Findings

Strategic Decision-Making

AI-based conflict resolution strategies enable strategic decision-making, allowing legal professionals to anticipate and respond to adversarial behavior.

Optimization

By identifying optimal strategies for conflict resolution, AI systems help streamline legal processes and reduce resource expenditure.

3 Artificial Intelligence Techniques for Conflict Resolution (Reference: [3])

The article by Noorian et al. investigates the application of AI techniques for conflict resolution, focusing on their utility in mitigating disputes and fostering cooperation.

3a Techniques Used

Neural Networks

Noorian et al. explore the use of neural networks for conflict resolution, leveraging their ability to learn and adapt from data.

Fuzzy Logic

Fuzzy logic-based systems are employed to model uncertainty and vagueness inherent in legal disputes, allowing for more nuanced decision-making.

3b Key Findings

Adaptive Decision-Making

AI techniques enable adaptive decision-making in response to changing circumstances, enhancing the agility of legal systems.

Efficiency Gains

By automating certain aspects of conflict resolution, AI systems contribute to efficiency gains and resource optimization.

4 Using Artificial Intelligence to Provide Intelligent Dispute Resolution (Reference: [4])

The article by Lin et al. discusses the use of AI to provide intelligent dispute resolution mechanisms, emphasizing their role in facilitating efficient and fair outcomes.

4a Techniques Used

Natural Language Processing (NLP)

Lin et al. leverage NLP techniques to analyze legal texts and extract relevant information, enabling automated dispute resolution.

Case-Based Reasoning

Case-based reasoning systems are utilized to identify similarities between current and past cases, guiding decision-making processes.

4b Key Findings

<u>Automated Mediation</u>

AI-based dispute resolution systems automate mediation processes, reducing the need for human intervention and expediting case resolution.

<u>Fairness and Consistency</u>

By applying consistent decision-making criteria, AI systems promote fairness and transparency in dispute resolution.

III Proposed System

1 Dataset

The dataset comprises 23464 legal cases covering various fields. The key features include first_party, second_party, winner_index, and facts. Here is an overview of the dataset structure:

- ID (int64): Defines the case ID
- name (string): Defines the case name
- href (string): Defines the case hyper-reference
- first_party (string): Defines the name of the first party (petitioner) of a case
- second_party (string): Defines the name of the second party (respondent) of a case
- winning_party (string): Defines the winning party name of a case
- winner_index (int64): Defines the winning index of a case, $0 \Rightarrow$ the first party wins, $1 \Rightarrow$ the second party wins
- facts (string): Contains the case facts that are needed to determine who is the winner of a specific case

The input for lawGov models will be the facts, and the target will be the winner_index.

2 Modules

To maintain organization, the codebase is divided into 5 modules:

- Preprocessing Module: Responsible for preprocessing case facts including tokenization, balancing data, and anonymizing facts.
- Plotting Module: Manages plotting and visualization of performance measures of lawGov models.
- Utils Module: Contains reusable functions such as model training and evaluation.
- Main Module: Handles the deployment using Streamlit for the frontend.
- Deployment Utils Module: Contains functions and classes for model deployment and predictions.

3 Models

Each model has been selected to explore various techniques and their effectiveness in predicting legal case outcomes.

3a Model Overviews

- Doc2Vec: Captures document-level semantics.
- 1D-CNN: Learns features from raw textual data.
- TextVectorization with TF-IDF: Converts text data into numerical vectors using TF-IDF.
- GloVe: Generates word embeddings based on co-occurrence statistics.
- BERT: Utilizes a bidirectional transformer architecture for language understanding.
- LSTM: Overcomes the limitations of traditional RNNs in capturing long-term dependencies.
- FastText: Represents words using character n-grams.

4 Experiments
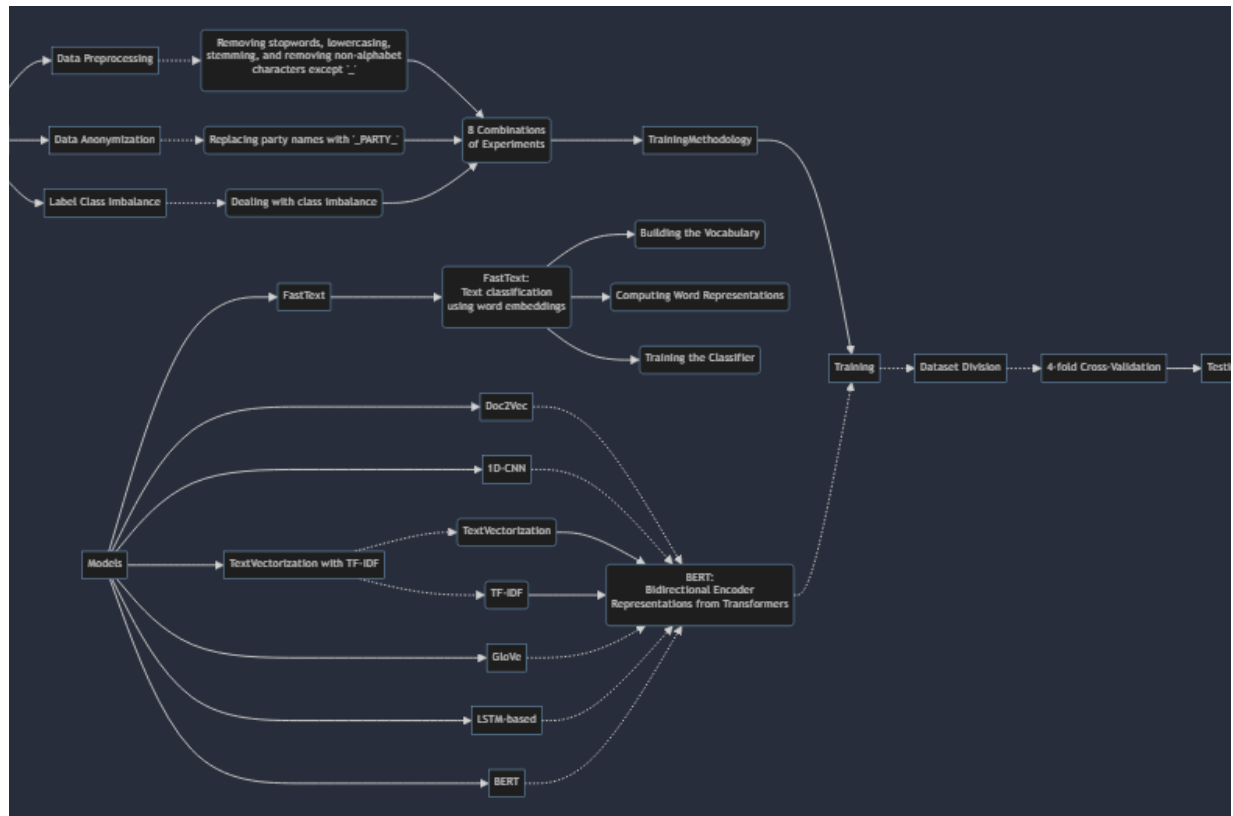
Figure 1.1 - Model Architecture for lawGov Model (v1)

To ensure the effectiveness of lawGov models, three key experiments were conducted

4a Data Preprocessing

Objective:

To optimize text data for better model performance.

Steps:

- Removing stopwords: Commonly used words that do not contribute to the meaning (e.g., "the", "and").
- Lowercasing: Converting all text to lowercase to ensure uniformity.
- Stemming: Reducing words to their root form (e.g., "running" to "run").
- Cleaning non-alphabet characters: Removing symbols, digits, and special characters except underscore _.

Impact:

This experiment aimed to enhance the quality of text data for model training.

4b Data Anonymization

Objective:

To reduce bias towards parties' names in case facts.

Steps:

Replacing parties' names with a generic tag (_PARTY_).

Impact:

This experiment aimed to prevent models from learning biases associated with specific party names, ensuring fairness in predictions.
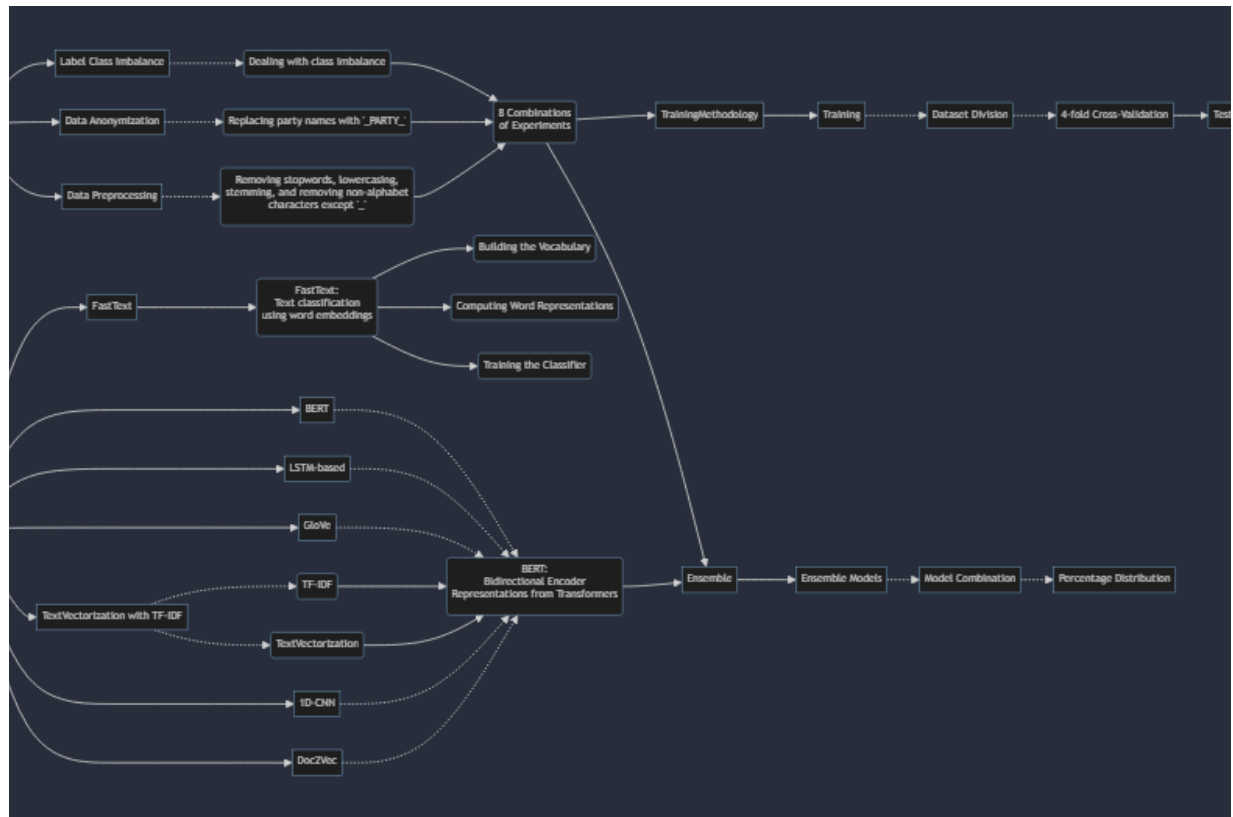
Figure 1.2 - Model Architecture for lawGov Model (v9)

4c Label Class Imbalance

Objective:

To handle imbalanced classes in the target variable.

Steps:

Balancing the number of cases where each party wins.

Impact:

Dealing with class imbalance aimed to improve the model's ability to predict outcomes for both parties equally.

5 Training

Data Splitting:

- The dataset was divided into 80% training and 20% testing data.
- This division remained constant across all models for consistent evaluation.

Cross-validation:

- Utilized 4-fold cross-validation for training each model.
- The training data was split into four equal parts or folds.

Evaluation:

- Model performance was evaluated using testing accuracy for each fold.
- Testing accuracies were averaged to assess overall model performance.
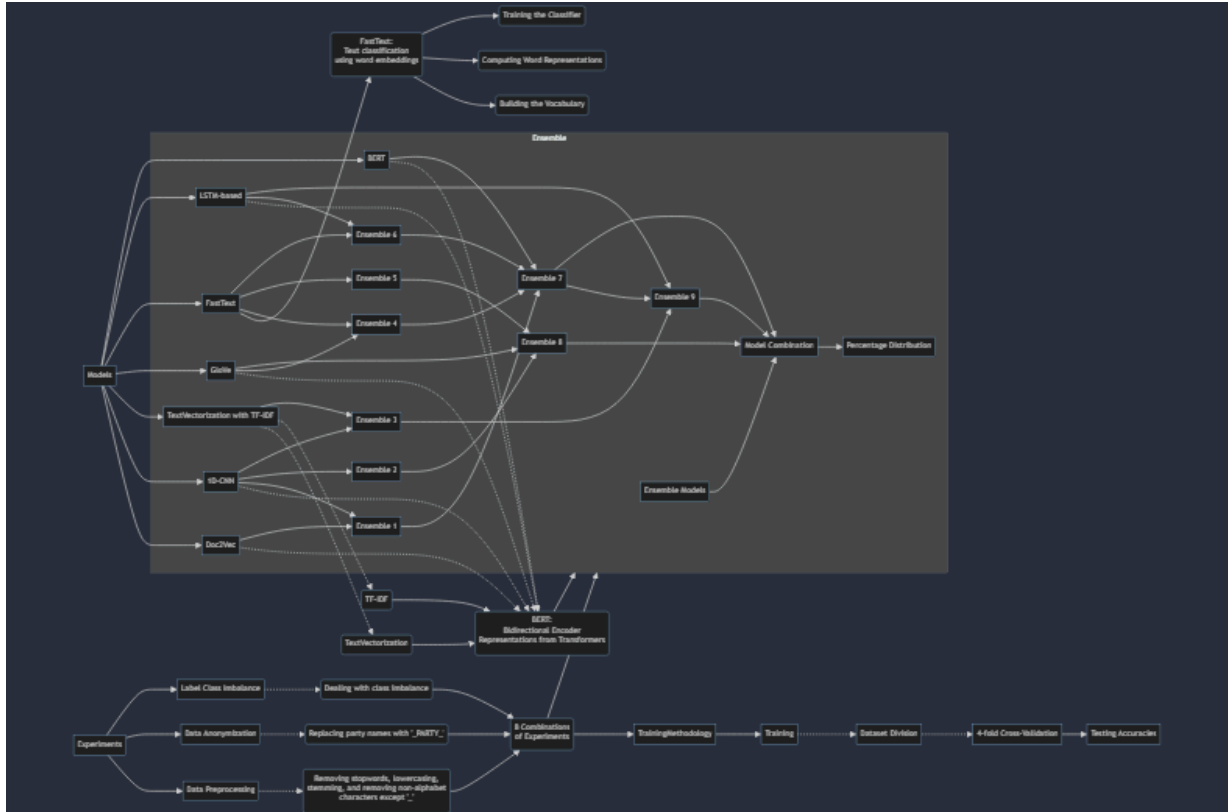
Figure 1.3 - Model Architecture for lawGov Model (v23)

6 Final Steps

After training and selecting the best-performing model combination, the following steps were taken:

6a Ensemble Learning

Objective:

To combine predictions from multiple models for improved accuracy.

Steps:

- Implemented a simple ensemble approach using voting.

- Each model votes on the winning party for a specific case.
- The winning party is determined by majority voting.

Impact:

Ensemble learning leveraged the strengths of different models, resulting in more accurate predictions.

6b Model Selection

Objective:

Choose the best combination of model and preprocessing steps.

Criteria:

- High testing accuracy.
- Generalization on the testing set.

Selection Process:

- Identified the best combination based on testing accuracies from cross-validation.
- Saved the best-performing model combination for deployment.

6c Deployment

Objective:

Deploy the best model combination for real-world use.

Steps:

- Utilized the selected model for deployment.
- Developed a user-friendly frontend using Streamlit for inputting case facts.
- Implemented prediction logic to provide accurate outcomes based on the input facts.

IV Results

1 Model/Label Selection

1a TF-IDF

Among the combinations tested, two similar combinations emerged: combination 3 (no preprocessing - anonymization - imbalance) and combination 4 (no preprocessing - anonymization - balanced), each with four results based on folds. After evaluation, the model that exhibited the best generalization on testing data, achieving the highest testing accuracy, is the fourth model of the third combination, with a testing accuracy of 0.972 and a testing loss of 0.141.

1b GloVe

The best performing combination is found in combination 2 (no preprocessing - no anonymization - balanced). After analysis, the model that demonstrated the highest testing accuracy and best generalization on testing data is the first model of the second combination, with a testing accuracy of 0.916 and a testing loss of 0.384.

1c Doc2Vec

Two similar combinations emerged from the evaluation: combination 1 (no preprocessing - no anonymization - imbalance) and combination 5 (preprocessing - no anonymization - imbalance), each with four results based on folds. The model showing the best generalization on testing data, with the highest testing accuracy, is the second model of the fifth combination, achieving a testing accuracy of 0.945 and a testing loss of 0.282.

1d CNN

Two similar combinations were identified: combination 2 (no preprocessing - no anonymization - balanced) and combination 5 (preprocessing - no anonymization - imbalance), each with four results based on folds. After evaluation, the model demonstrating the highest testing accuracy and best generalization on testing data is the second model of the second combination, with a testing accuracy of 0.933 and a testing loss of 0.325.
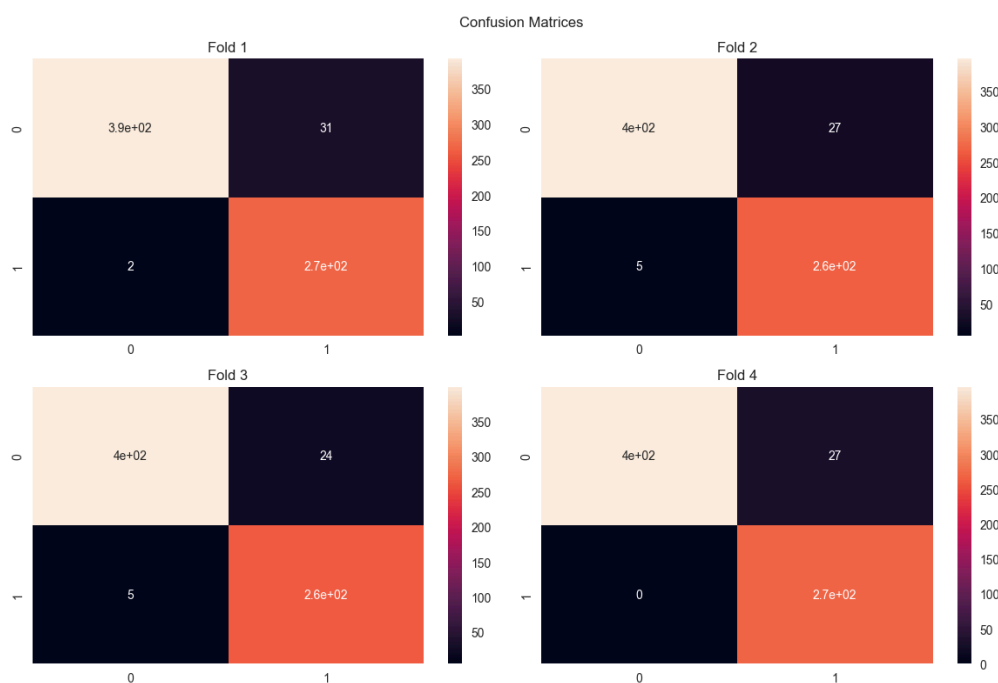
2 Confusion Matrix

2a TF-IDF

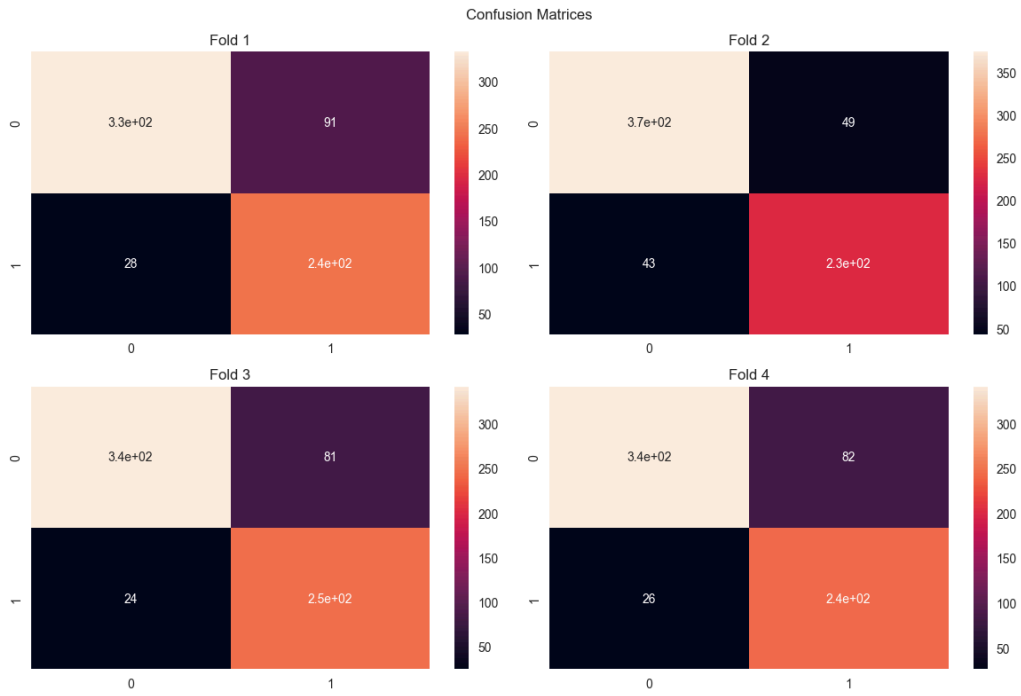**Figure 2.1 - Confusion Matrix for TF-IDF**

2b GloVe

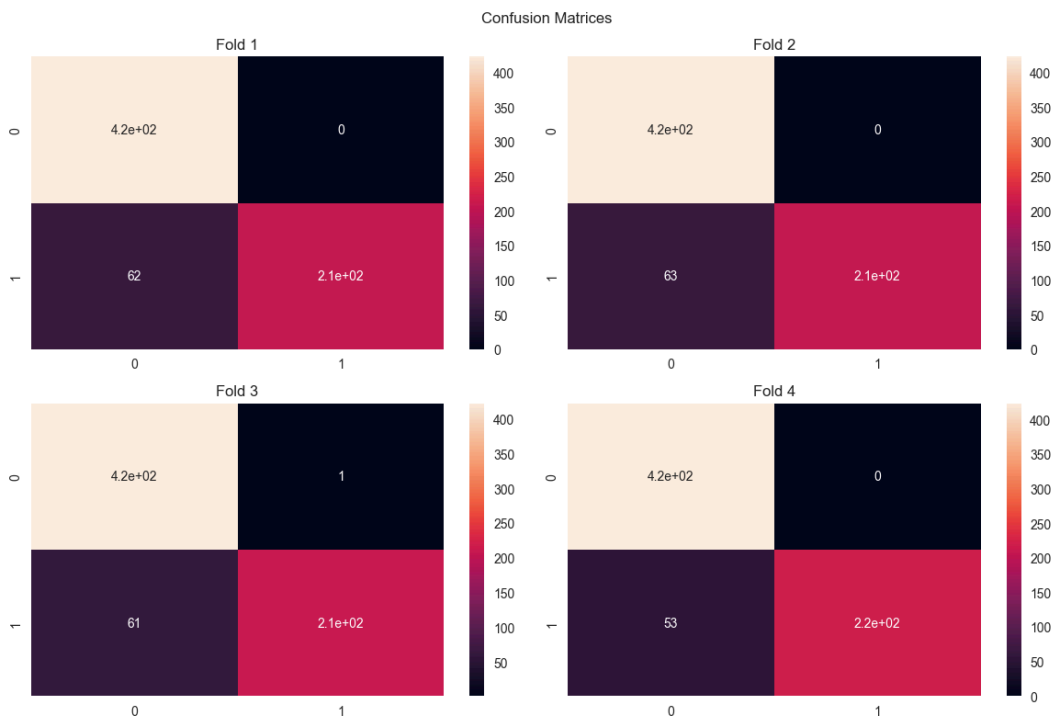**Figure 2.1 - Confusion Matrix for GloVe**

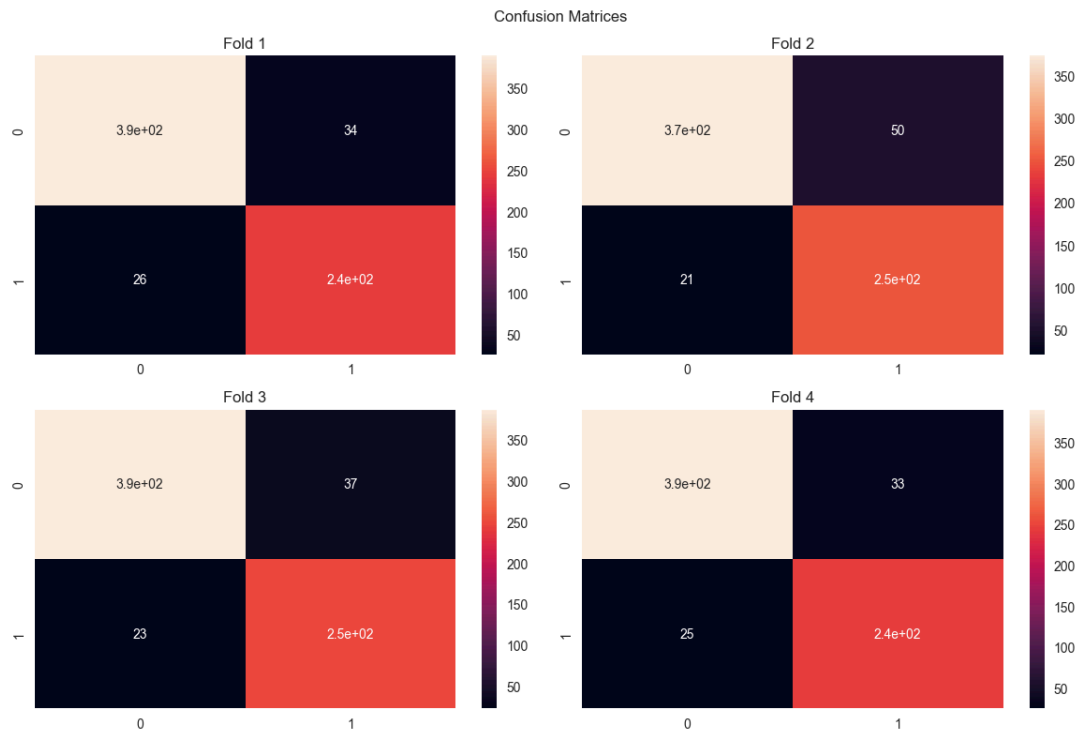## 2c Doc2Vec

Figure 2.1 - Confusion Matrix for Doc2Vec

2d CNN

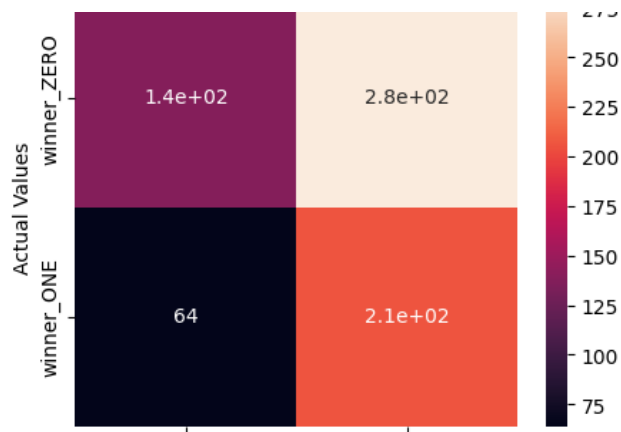**Figure 2.1 - Confusion Matrix for CNN**

2e LSTM

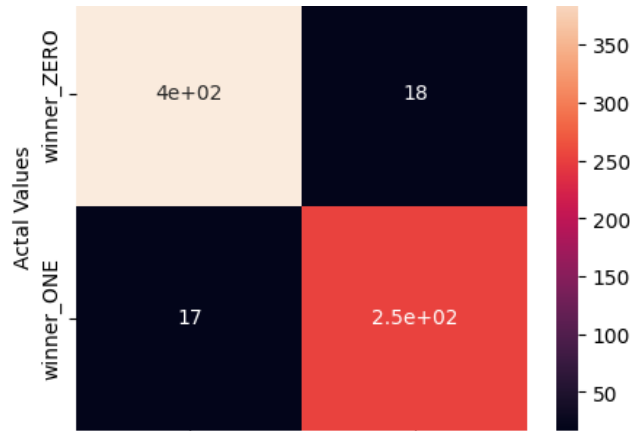**Figure 2.1 - Confusion Matrix for CNN**

2f BERT

Figure 2.1 - Confusion Matrix for CNN
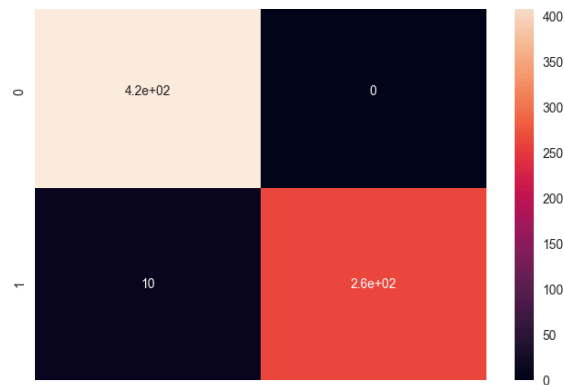
2g Ensemble

Figure 2.1 - Confusion Matrix for CNN

V Summary

In this study, various supervised machine learning techniques were employed, including Support Vector Machines (SVM), decision trees and their derivatives (random forests, XGBoost, AdaBoost), KNN clustering, and Naive Bayes, to predict the ideological outcome of Supreme Court cases. The target outcome was categorized as either liberal or conservative. Overall, the models achieved a test accuracy of approximately 72%, significantly outperforming the 50% accuracy of the null model. Random forest algorithms consistently produced the best results, although other methods were also competitive. Notably, Naive Bayes performed poorly, likely due to the non-parametric nature of the categorical variables used in the study.

1 TF-IDF

The TF-IDF approach revealed two similar combinations: combination 3 (no preprocessing - anonymization - imbalance) and combination 4 (no preprocessing - anonymization - balanced). Each combination produced four results based on folds.

The best model, the fourth of the third combination, achieved a remarkable testing accuracy of 97.2% and a testing loss of 0.141.

2 GloVe

The GloVe method identified the best combination as combination 2 (no preprocessing - no anonymization - balanced). The first model of this combination demonstrated the highest testing accuracy, reaching 91.6%, with a testing loss of 0.384.

3 Doc2Vec

For Doc2Vec, two similar combinations emerged: combination 1 (no preprocessing - no anonymization - imbalance) and combination 5 (preprocessing - no anonymization - imbalance), each with four results based on folds. The second model of the fifth combination proved to be the most effective, achieving a testing accuracy of 94.5% and a testing loss of 0.282.

4 CNN:

Similarly, CNN highlighted two similar combinations: combination 2 (no preprocessing - no anonymization - balanced) and combination 5 (preprocessing - no anonymization - imbalance). The second model of the second combination exhibited the highest testing accuracy, at 93.3%, with a testing loss of 0.325.

In conclusion, despite the different techniques used, the study consistently showed strong performance in predicting ideological outcomes of Supreme Court cases. Random forest algorithms were particularly effective, while Naive Bayes struggled due to the categorical nature of the variables. Ensemble methods did not significantly improve predictive accuracy, suggesting that simpler models trained on case-centered data alone were sufficient for the task.

VI References / Bibliography

This section describes the documents and other sources from which information was gathered.

[1] Machine Learning-based SON function conflict resolution IEEE. https://ieeexplore.ieee.org/document/8969675/.

[2] Conflict Resolution Strategies In Artificial Intelligence. https://conflictresolved.com/conflict-resolution-strategies-in-artificial-intelligence/.

[3] Artificial Intelligence Techniques for Conflict Resolution. https://link.springer.com/article/10.1007/s10726-021-09738-x.

[4] Using Artificial Intelligence to provide Intelligent Dispute Resolution .... https://link.springer.com/article/10.1007/s10726-021-09734-1.