

Section B – Practical Questions (Code + Output)

1. CSV Data Exploration & Visualization

1.1 Load & Inspect

```
1_data_exploration_and_visualization.py X
assignments > 1_data_exploration_and_visualization.py > ...
1 # Importing necessary libraries
2 import pandas as pd
3 import matplotlib.pyplot as plt
4
5 # Loading CSV file into a dataframe and showing the first 10 rows
6 # Note: Titanic dataset used here is downloaded from kaggle dataset repository
7 df = pd.read_csv(r"C:\Users\devar\Documents\Dotkonnect\genai-workshop\week2\data\titanic.csv")
8 df.head()
9
10 # Displaying shape, column names, and data types.
11 print("Shape of the DataFrame:", df.shape)
12 print("Column Names:", df.columns.tolist())
13 print("Data Types:\n", df.dtypes)
14
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS python +

(env) C:\Users\devar\Documents\Dotkonnect\genai-workshop\week2\assignments>python 1_data_exploration_and_visualization.py
Shape of the DataFrame: (418, 12)
Column Names: ['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked']
Data Types:
PassengerId int64
Survived int64
Pclass int64
Name object
Sex object
Age float64
SibSp int64
Parch int64
Ticket object
Fare float64
Cabin object
Embarked object
dtype: object

1.2 Summary Statistics

```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...
15 # Description of the dataset
16 print("Summary Statistics:\n", df.describe())
17
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS cmd +

Summary Statistics:

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	0.363636	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.481622	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	0.000000	1.000000	0.170000	0.000000	0.000000	0.000000
25%	996.250000	0.000000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	0.000000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	1.000000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	1.000000	3.000000	76.000000	8.000000	9.000000	512.329200

Missing Values:

```
1_data_exploration_and_visualization.py X
assignments > 1_data_exploration_and_visualization.py > ...

18 # Count missing values per column.
19 print("Missing Values:\n", df.isna().sum())
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Missing Values:

PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	86
SibSp	0
Parch	0
Ticket	0
Fare	1
Cabin	327
Embarked	0

dtype: int64

```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...

22 # Fill missing numeric values with the mean.
23 for column in df.columns.tolist():
24     if df[column].isna().sum() > 0 and df[column].dtype in ['float64', 'int64']:
25         df[column].fillna(df[column].mean(), inplace=True)
26 # Count missing values after filling
27 print("Missing Values:\n", df.isna().sum())
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

df[column].fillna(df[column].mean(), inplace=True)

Missing Values:

PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	0
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	327
Embarked	0

dtype: int64

1.3 Filter & Sort

```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...

29 # Filter rows by a numeric condition (e.g., Age > 30)
30 filtered_df = df[df['Age'] > 30]
31 print("Filtered DataFrame (Age > 30):\n", filtered_df.head())
```

Filtered DataFrame (Age > 30):

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.50000	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	9.6875	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	9.6875	NaN	Q
10	902	0	3	Ilieff, Mr. Ylio	male	30.27259	0	0	349220	7.8958	NaN	S
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	9.6875	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	9.6875	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.00000	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.00000	0	0	240276	9.6875	NaN	Q
10	902	0	3	Ilieff, Mr. Ylio	male	30.27259	0	0	349220	7.8958	NaN	S
11	903	0	1	Jones, Mr. Charles Cresson	male	46.00000	0	0	694	26.0000	NaN	S

```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...

33 # Sort dataset by a column in descending order (eg. PassengerId).
34 sorted_df = df.sort_values(by = 'PassengerId', ascending = False)
35 print("Sorted DataFrame by PassengerId (Descending):\n", sorted_df.head())
36
```

Sorted DataFrame by PassengerId (Descending):

	PassengerId	Survived	Pclass	Name	Sex	...	Parch	Ticket	Fare	Cabin	Embarked
417	1309	0	3	Peter, Master. Michael J	male	...	1	2668	22.3583	NaN	C
416	1308	0	3	Ware, Mr. Frederick	male	...	0	359309	8.0500	NaN	S
	PassengerId	Survived	Pclass	Name	Sex	...	Parch	Ticket	Fare	Cabin	Embarked
417	1309	0	3	Peter, Master. Michael J	male	...	1	2668	22.3583	NaN	C
416	1308	0	3	Ware, Mr. Frederick	male	...	0	359309	8.0500	NaN	S
417	1309	0	3	Peter, Master. Michael J	male	...	1	2668	22.3583	NaN	C
416	1308	0	3	Ware, Mr. Frederick	male	...	0	359309	8.0500	NaN	S
415	1307	0	3	Saether, Mr. Simon Sivertsen	male	...	0	SOTON/O.Q. 3101262	7.2500	NaN	S
414	1306	1	1	Oliva y Ocana, Dona. Fermina	female	...	0	PC 17758	108.9000	C105	C
413	1305	0	3	Spector, Mr. Woolf	male	...	0	A.5. 3236	8.0500	NaN	S

1.4 Group & Aggregate

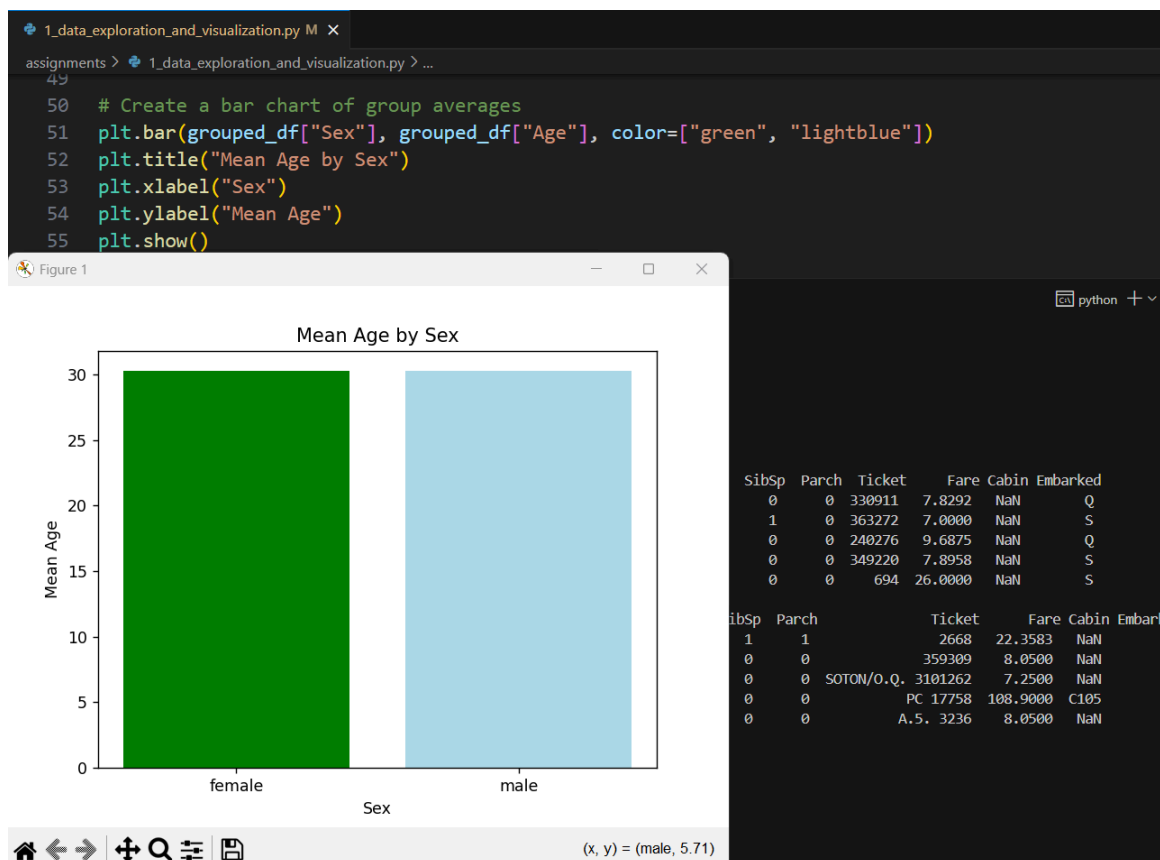
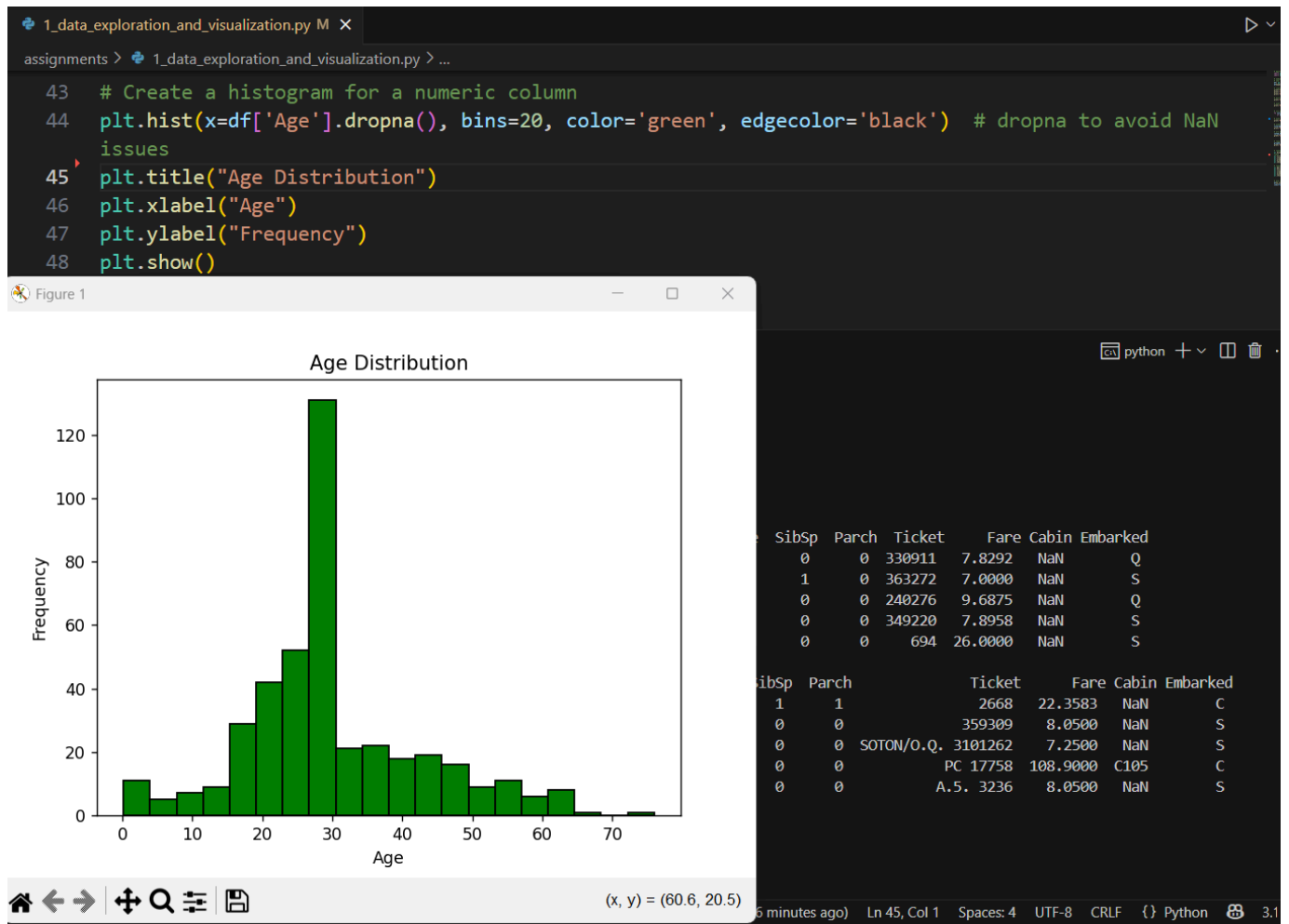
```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...

38 # Group by a categorical column, calculate mean of a numeric column ()
39 grouped_df = df.groupby("Sex")["Age"].mean().reset_index()
40 print("Grouped DataFrame (Mean Age):\n", grouped_df)
41
```

Grouped DataFrame (Mean Age):

	Sex	Age
0	female	30.272400
1	male	30.272699

1.5 Visualize



1.6 Extra Challenge: Save the cleaned dataset as processed_data.csv.

```
1_data_exploration_and_visualization.py M X
assignments > 1_data_exploration_and_visualization.py > ...
56
57
58 # Save the cleaned dataset as processed_data.csv
59 df.to_csv(r"C:\Users\devar\Documents\Dotkonnekt\genai-workshop\week2\data\processed_titanic.csv",
60          index=False)
61
```

2. Decision Tree Classifier

```
2_decision_tree_classifier.py M X
assignments > 2_decision_tree_classifier.py > ...
1 # Import necessary libraries
2 from sklearn.datasets import load_iris
3 from sklearn.tree import DecisionTreeClassifier, plot_tree
4 from sklearn.model_selection import train_test_split
5 from sklearn.metrics import accuracy_score
6 import matplotlib.pyplot as plt
7
8 # Load the Iris dataset
9 iris = load_iris()
10 X = iris.data
11 y = iris.target
12
13 # Split into training and testing sets (80% train, 20% test)
14 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
15
16 # Create and train the Decision Tree Classifier
17 clf = DecisionTreeClassifier(random_state=42)
18 clf.fit(X_train, y_train)
19
20 # Predict on the test set
21 y_pred = clf.predict(X_test)
22
23 # Calculate accuracy
24 accuracy = accuracy_score(y_test, y_pred)
25 print(f"Accuracy: {accuracy:.2f}")
26
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

python +

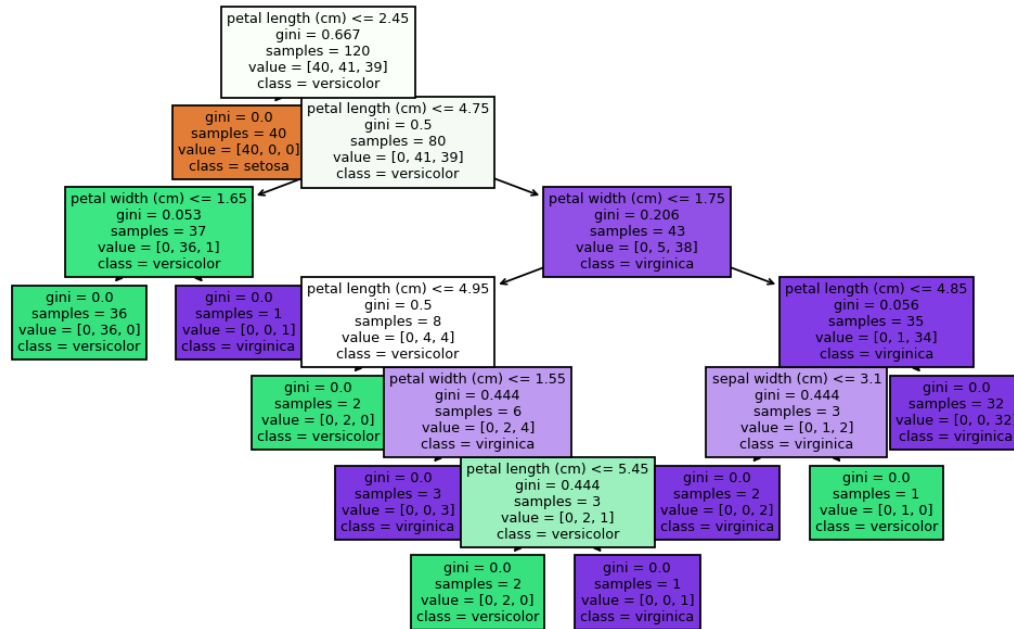
Accuracy: 1.00

2_decision_tree_classifier.py M X

assignments > 2_decision_tree_classifier.py > ...

```
26
27 # Plot the Decision Tree
28 plt.figure(figsize=(12, 8))
29 plot_tree(clf, filled=True, feature_names=iris.feature_names, class_names=iris.target_names)
30 plt.show()
```

Figure 1



3. Text Processing with NLTK

3_text_processing_with_nltk.py M X

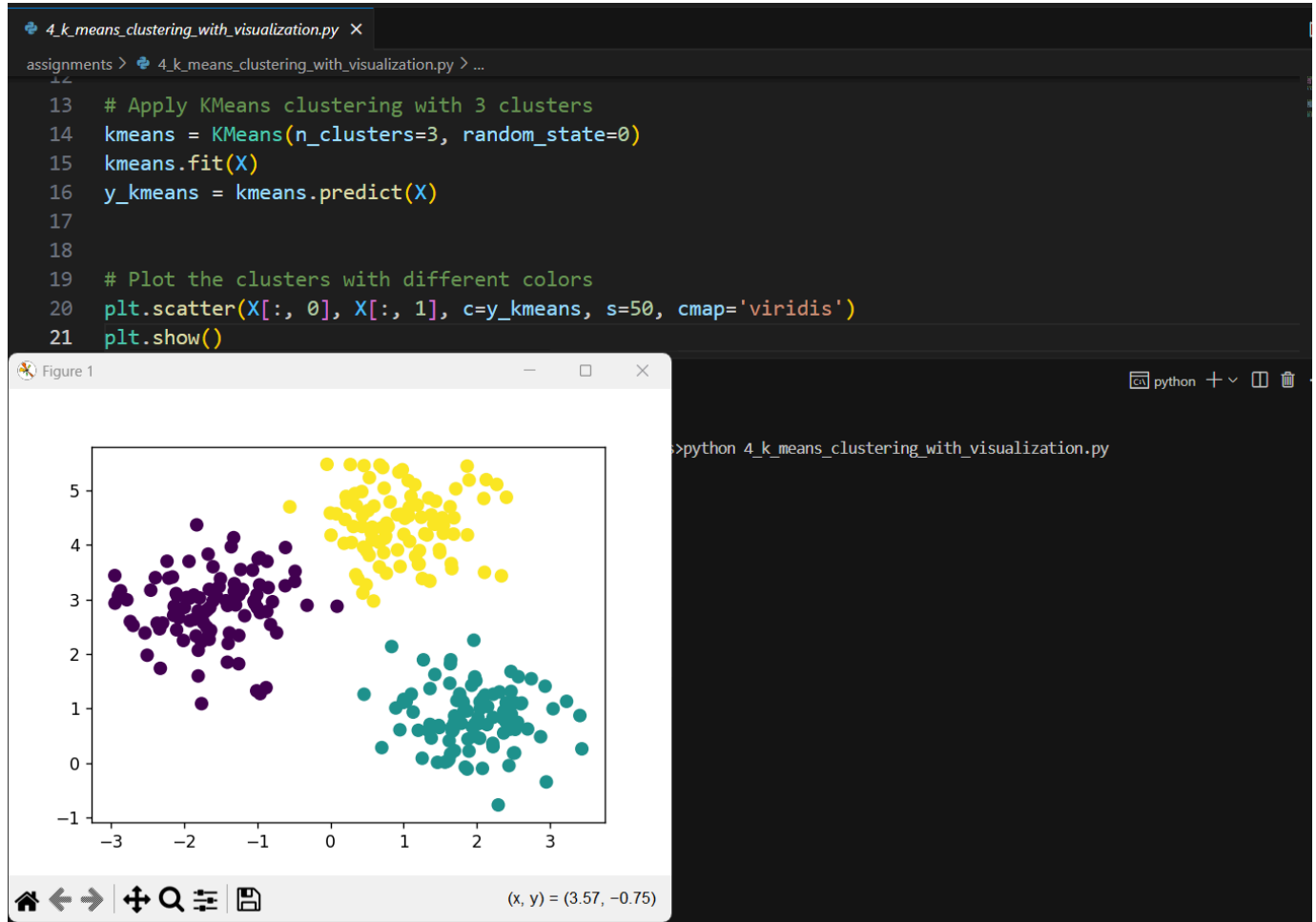
assignments > 3_text_processing_with_nltk.py > ...

```
35 # Output results
36 print("Original Tokens:", tokens)
37 print("Filtered Tokens (No Stopwords):", filtered_tokens)
38 print("POS Tags:", pos_tags)
39 print(f"Number of Nouns: {num_nouns}")
40 print(f"Number of Verbs: {num_verbs}")
41 print(f"Number of Adjectives: {num_adjectives}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
Filtered Tokens (No Stopwords): ['enthraling', 'draw', 'trophy', 'England', 'India', 'provided', 'dramatic', 'start', 'new', 'World', 'Test', 'Champion', 'ship', 'cycle', 'epic', 'contest', 'five', 'Tests', 'going', 'final', 'day', 'four', 'fact', 'final', 'session', 'providing', 'best', 'individual', 'c', 'ollective', 'performances', 'format', 'seen', 'recent', 'years']
POS Tags: [('enthraling', 'VBG'), ('draw', 'JJ'), ('trophy', 'NN'), ('England', 'NNP'), ('India', 'NNP'), ('provided', 'VBD'), ('dramatic', 'JJ'), ('s', 'tart', 'RB'), ('new', 'JJ'), ('World', 'NNP'), ('Test', 'NNP'), ('Championship', 'NNP'), ('cycle', 'NN'), ('epic', 'NN'), ('contest', 'NN'), ('five', 'CD'), ('Tests', 'NNS'), ('going', 'VBG'), ('final', 'JJ'), ('day', 'NN'), ('four', 'CD'), ('fact', 'NN'), ('final', 'JJ'), ('session', 'NN'), ('providi', 'ng', 'VBG'), ('best', 'JJS'), ('individual', 'JJ'), ('collective', 'NN'), ('performances', 'NNS'), ('format', 'VBP'), ('seen', 'VBN'), ('recent', 'JJ'), ('years', 'NNS')]
Number of Nouns: 16
Number of Verbs: 6
Number of Adjectives: 8
```

4. K-Means Clustering with Visualization



5. Confusion Matrix Plot

```
5_confusion_matrix_plot.py X
assignments > 5_confusion_matrix_plot.py > ...
19 # Predict
20 y_pred = clf.predict(X_test)
21
22 # Compute confusion matrix
23 cm = confusion_matrix(y_test, y_pred, labels=clf.classes_)
24
25 # Plot confusion matrix
26 disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=iris.target_names)
27 disp.plot(cmap=plt.cm.Blues)
28 plt.title("Confusion Matrix - Decision Tree on Iris")
29 plt.show()
```

