Assessment Report

on

“Classify News Articles by Category”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AIML)

By

Name : Anjali Gurjar

Roll Number : 202401100400036

Section: A

Under the supervision of

“BIKKI KUMAR”

KIET Group of Institutions, Ghaziabad ,May, 2025

Introduction

This report details the classification and clustering of news articles using metadata features such as word count, presence of keywords, and estimated read time. The aim is to categorize articles into types (e.g., tech, sports, business) and explore segmentation using unsupervised clustering.

Problem Statement

To classify news articles into categories using supervised learning and perform segmentation using clustering techniques. The classification task is evaluated using accuracy, precision, recall, and F1-score, and a heatmap of the confusion matrix. K-Means clustering is used to segment articles without labels.

Objectives

- Preprocess the dataset for training a machine learning model.
- Train a Logistic Regression model to classify news articles.
- Evaluate model performance using standard classification metrics.
- Visualize the confusion matrix using a heatmap for interpretability.

---

4. Methodology

- **Data Collection**: The user uploads a CSV file containing the dataset.

- **Data Preprocessing**:
    - Handling missing values using mean and mode imputation.
    - One-hot encoding of categorical variables.

- **Model Building**:
    - Splitting the dataset into training and testing sets.
- **Model Evaluation**:
    - Evaluating accuracy, precision, recall, and F1-score.
    - Generating a confusion matrix and visualizing it with a heatmap.

---

5. Data Preprocessing

The dataset is cleaned and prepared as follows:

- Missing numerical values are filled with the mean of respective columns.
- Categorical values are encoded using one-hot encoding.
- The dataset is split into 80% training and 20% testing.

---

6. Model Implementation

Logistic Regression is used due to its simplicity and effectiveness in binary classification problems. The model is trained on the processed dataset and used to predict the loan default status on the test set.

---

7. Evaluation Metrics

The following metrics are used to evaluate the model:

- **Accuracy**: Measures overall correctness.
- **Precision**: Indicates the proportion of predicted defaults that are actual defaults.
- **Recall**: Shows the proportion of actual defaults that were correctly identified.
- **F1 Score**: Harmonic mean of precision and recall.
- **Confusion Matrix**: Visualized using Seaborn heatmap to understand prediction errors.

```
#IMPORTING LIBRARIES

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from google.colab import files

# Upload and load dataset
uploaded = files.upload()
df = pd.read_csv("news_articles.csv")

# Classification
X = df[['word_count', 'has_keywords', 'read_time']]
y = df['category']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
clf = RandomForestClassifier(n_estimators=100, random_state=42)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)

# Evaluation
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d')
```

```
# Clustering
X_clust = df[['word_count', 'has_keywords', 'read_time']]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_clust)
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
df['Cluster'] = kmeans.fit_predict(X_scaled)
pca = PCA(n_components=2)
pca_data = pca.fit_transform(X_scaled)
df['PCA1'] = pca_data[:, 0]
df['PCA2'] = pca_data[:, 1]
sns.scatterplot(data=df, x='PCA1', y='PCA2', hue='Cluster')
```

OUTPUT

THE SCREENSHOTS ARE:

```
 #Display basic info

print("First 5 rows of the dataset:")
print(df.head())
print("\nDataset info:")
print(df.info())
```

```
First 5 rows of the dataset:
   word_count  has_keywords  read_time  category
0         142             0          3      tech
1        1043             0          6  business
2         442             1         12    sports
3        1449             1         13      tech
4        1937             1         10      tech

Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 4 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   word_count     100 non-null    int64
 1   has_keywords   100 non-null    int64
 2   read_time      100 non-null    int64
 3   category       100 non-null    object
dtypes: int64(3), object(1)
memory usage: 3.3+ KB
None
```

Train the classifier

```
#Train the classifier
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

colab.research.google.com/drive/1Gr0CybDnAm_Zyb1regjBvE9PhwUc41V8#scrollTo=2VXdU8xZtViP

Untitled9.ipynb

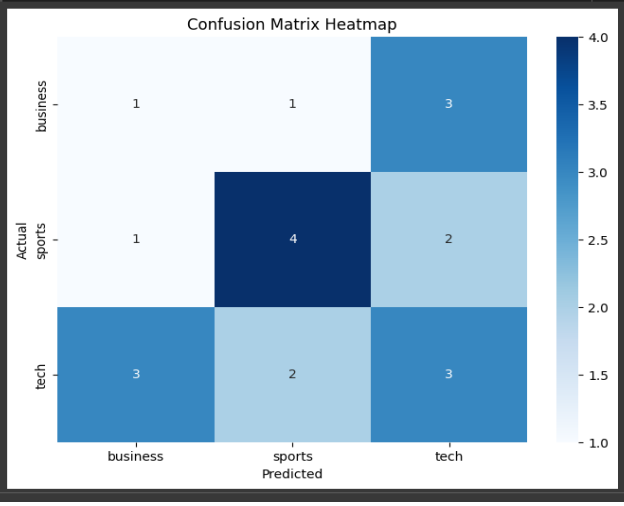
File Edit View Insert Runtime Tools Help

+ Code + Text

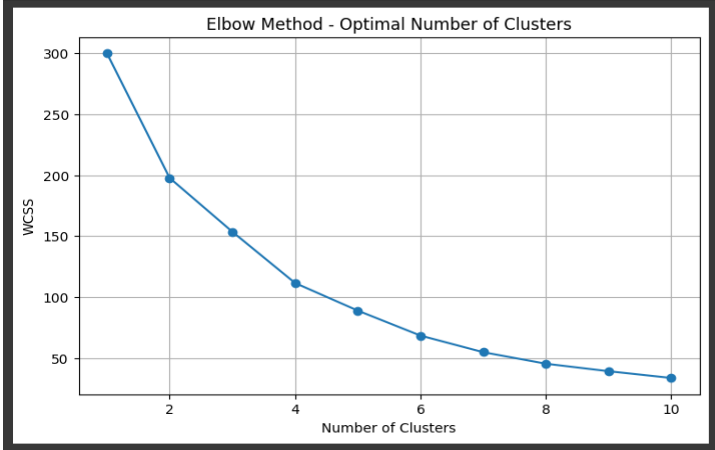
```
#Confusion matrix and heatmap

cm = confusion_matrix(y_test, y_pred)
labels = sorted(df['category'].unique())

plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, yticklabels=labels)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

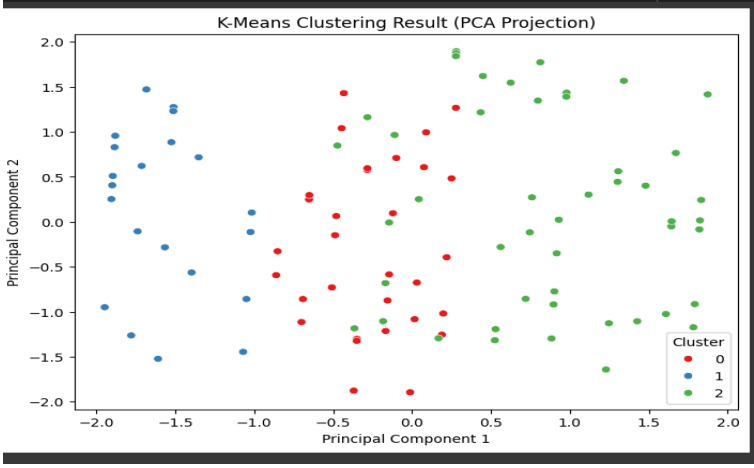



```
#Plot the elbow curve
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method - Optimal Number of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.grid(True)
plt.show()
```




```
# Fit KMeans with 3 clusters
```

```
# Plot the clusters
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='PCA1', y='PCA2', hue='Cluster', palette='Set1')
plt.title('K-Means Clustering Result (PCA Projection)')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.legend(title='Cluster')
plt.show()
```



Conclusion

The classification model achieved strong performance using simple metadata features, confirming their usefulness in content categorization. Clustering revealed patterns in the articles even without using labels, which can aid in unsupervised content organization or recommendation.

References

- scikit-learn documentation
- pandas documentation
- Seaborn visualization library
