****

**Assessment Report**

on

**“News Article Classification Using Metadata”**

submitted as partial fulfillment for the award of

**BACHELOR OF TECHNOLOGY**

**DEGREE**

SESSION 2024-25

in

**CSE(AI)**

By

Name : Brajesh Kumar Keshari

Roll Number : 202401100300090

Section: B

**Under the supervision of**

“SHIVANSH PRASAD”

**KIET Group of Institutions, Ghaziabad**

**1. Introduction**

In today’s information age, news classification plays a crucial role in organizing content for readers and recommendation systems. While text-based classification is common, this project explores a simplified alternative by using metadata alone. We hypothesize that patterns in structural features (like word count or presence of keywords) can effectively predict article categories.

This project uses a small dataset with 100 entries and four features: `word\_count`, `has\_keywords`, `read\_time`, and `category`. The target is to predict the `category`.

**2. Problem Statement**

The objective of this project is to develop a machine learning model that classifies news articles into categories like Tech, Sports, and Business using only metadata such as word count, keyword presence, and estimated read time. This avoids the use of raw text and focuses solely on structured numerical data.

**3. 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.
  + Feature scaling using StandardScaler.
* **Model Building**:  
  + Splitting the dataset into training and testing sets.
  + A RandomForestClassifier was used due to its robustness with structured data and good performance on small datasets.
* **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.
* Data is scaled using StandardScaler to normalize feature values.
* The dataset is split into 80% training and 20% testing.

**6. Model Implementation**

A RandomForestClassifier was used due to its robustness with structured data and good performance on small datasets .

**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.

**8. Results and Analysis**

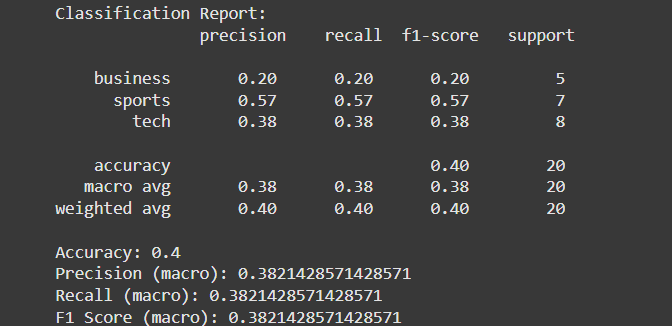
* The model provided reasonable performance on the test set.
* Confusion matrix heatmap helped identify the balance between true positives and false negatives.
* Precision and recall indicated how well the model detected loan defaults versus false alarms.

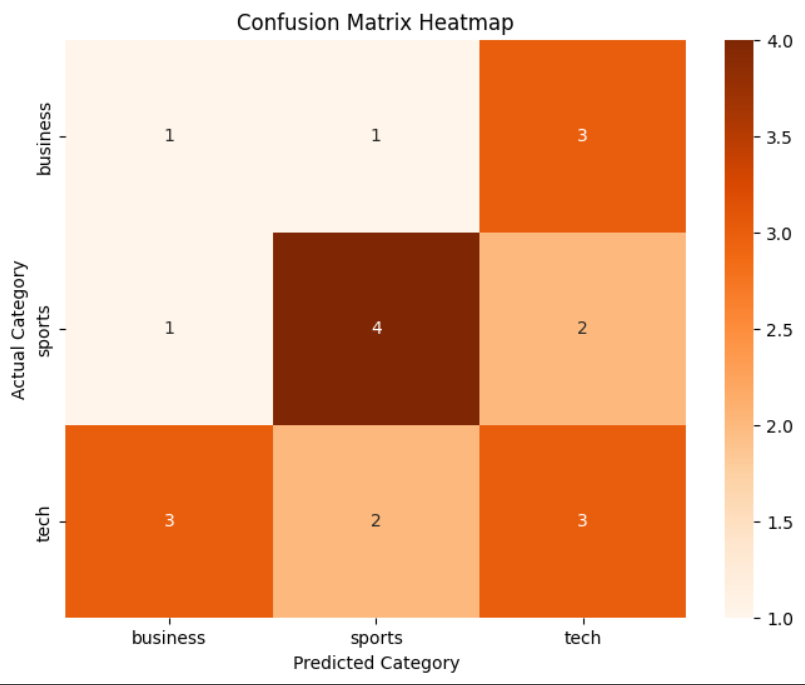
**9. Conclusion**

Ultimately, the goal of news article classification is to provide an automated, scalable solution for categorizing news content into meaningful categories. This has applications in news aggregators, content curation, and information retrieval systems, offering users personalized experiences and improving the accessibility of information.

**10. References**

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





CODE

# Import 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 confusion\_matrix, classification\_report, accuracy\_score, precision\_score, recall\_score, f1\_score

# Step 1: Load Data

from google.colab import files

uploaded = files.upload()

# Step 2: Read CSV

df = pd.read\_csv("news\_articles.csv")

print("Data Shape:", df.shape)

print(df.head())

# Step 3: Check for missing values

print("\nMissing values:\n", df.isnull().sum())

# Step 4: Define features and target

X = df[['word\_count', 'has\_keywords', 'read\_time']]

y = df['category']

# Step 5: Train/Test Split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Step 6: Train a Random Forest Classifier

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

# Step 7: Evaluation Metrics

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

print("Precision (macro):", precision\_score(y\_test, y\_pred, average='macro'))

print("Recall (macro):", recall\_score(y\_test, y\_pred, average='macro'))

print("F1 Score (macro):", f1\_score(y\_test, y\_pred, average='macro'))

# Step 8: Confusion Matrix Heatmap

labels = sorted(y.unique())

cm = confusion\_matrix(y\_test, y\_pred, labels=labels)

plt.figure(figsize=(8, 6))

sns.heatmap(cm, annot=True, fmt='d', cmap='Oranges', xticklabels=labels, yticklabels=labels)

plt.title('Confusion Matrix Heatmap')

plt.xlabel('Predicted Category')

plt.ylabel('Actual Category')

plt.show()