**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(AI)

By

Name: Avinash Kumar

Roll Number: 202401100300079

Section: B

**Under the supervision of**

"Shivansh Prasad"

## KIET Group of Institutions, Ghaziabad

**May, 2025**

## Introduction

The goal of this project is to classify news articles into predefined categories, such as sports, technology, business, etc., using article metadata and keywords. This task involves training a machine learning model on a dataset containing various article features, including word count, presence of specific keywords, and estimated read time. By leveraging these features, the model will predict the category of each article.

## Model Training

To classify news articles into categories, we used the **Random Forest Classifier**, a robust ensemble learning method that combines multiple decision trees to improve accuracy and avoid overfitting. Here's a breakdown of the model training process:

1. **Dataset Split**: The dataset is split into training (80%) and testing (20%) sets to ensure the model can be trained on a majority of the data while being evaluated on a separate portion.
2. **Features Used**: The features chosen for the classification model are:
    - word_count: Total number of words in the article.
    - has_keywords: A binary feature indicating whether the article contains specific keywords.
    - read_time: Estimated time required to read the article.

3. **Model Training**: The Random Forest Classifier is trained on the training set using the selected features. The model learns to map the feature values to their respective categories.

## Methodology

The model utilizes the following steps for classification:

1. **Data Preprocessing**:
    - The dataset is first loaded, and relevant columns (features) are selected.
    - The data is then split into training and testing sets using an 80/20 ratio.
2. **Model Training**:
    - A Random Forest Classifier is trained using the training dataset. This model uses multiple decision trees to make predictions based on majority voting, thus improving accuracy.
3. **Model Evaluation**:
    - After training the model, predictions are made on the test dataset.
    - The results are evaluated using a **confusion matrix**, which helps visualize the performance of the classifier.
    - Additional metrics such as **accuracy**, **precision**, **recall**, and **F1 score** are also computed using the classification report.
4. **Visualization**:

- A confusion matrix heatmap is plotted to show the distribution of predicted versus actual categories.

## 3. CODE

```
# Import necessary libraries

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import confusion_matrix, classification_report, accuracy_score

import seaborn as sns

import matplotlib.pyplot as plt


# Load the dataset (ensure you upload the CSV file to your Google Colab session)

from google.colab import files

uploaded = files.upload()


# Read the uploaded file

file_name = list(uploaded.keys())[0]

news_data = pd.read_csv(file_name)


# Prepare data for classification

features = ['word_count', 'has_keywords', 'read_time']

X = news_data[features]

y = news_data['category']


# Split data into training and testing sets
```

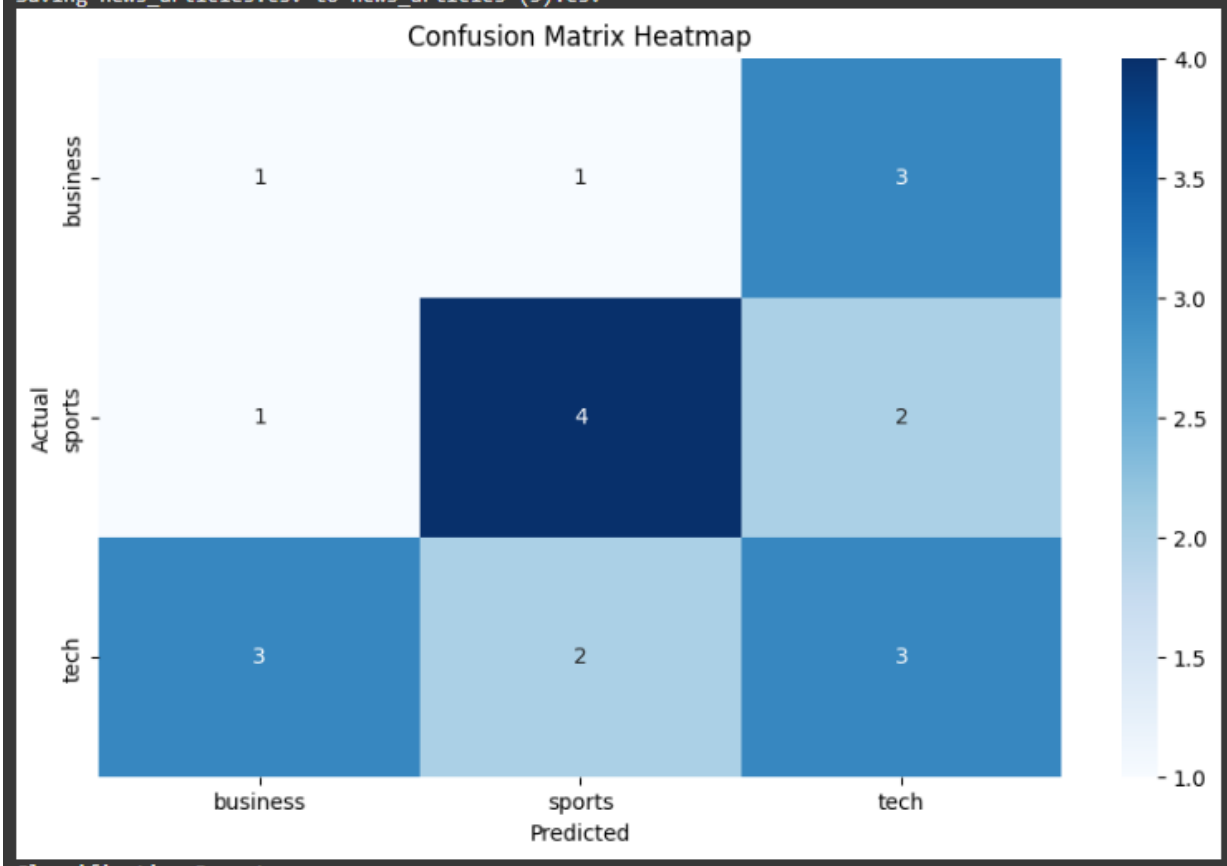
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train a Random Forest Classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Predict on test data
y_pred = clf.predict(X_test)

# Generate confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred, labels=clf.classes_)

# Plot the confusion matrix heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', xticklabels=clf.classes_, yticklabels=clf.classes_,
cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap')
plt.show()

# Print classification report and accuracy
print("Classification Report:")
print(classification_report(y_test, y_pred, target_names=clf.classes_))
print("Accuracy:", accuracy_score(y_test, y_pred))
```

## 4. Output

Choose Files news_articles.csv

- **news_articles.csv**(text/csv) - 1611 bytes, last modified: 4/22/2025 - 100% done

```
Saving news_articles.csv to news_articles (3).csv
```




```
Classification Report:
              precision    recall  f1-score   support

    business       0.20      0.20      0.20         5
      sports       0.57      0.57      0.57         7
        tech       0.38      0.38      0.38         8

    accuracy                           0.40        20
   macro avg       0.38      0.38      0.38        20
weighted avg       0.40      0.40      0.40        20

Accuracy: 0.4
```

# References


1. **Random Forest Classifier**:
    - Breiman, L. (2001). Random Forests. *Machine Learning, 45*(1), 5–32. https://doi.org/10.1023/A:1010933404324
    - Scikit-learn documentation on Random Forests: https://scikit-learn.org/stable/modules/ensemble.html#random-forest
2. **Data Science and Machine Learning Resources**:
    - "Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow" by Aurélien Géron
    - "Python Machine Learning" by Sebastian Raschka
3. **Confusion Matrix and Classification Metrics**:
    - Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies, 2*(1), 37-63.