Classify News Articles by Category

Name: Archy Mittal
Roll No: 2024011004000046
Subject: Introduction to AI
To: Bikki Gupta Sir

Introduction:

In this project, we focus on classifying news articles into different categories such as Sports, Tech, Business, and others based on their metadata. Classification of news articles helps in organizing large amounts of data and enables users to filter and consume relevant information efficiently.

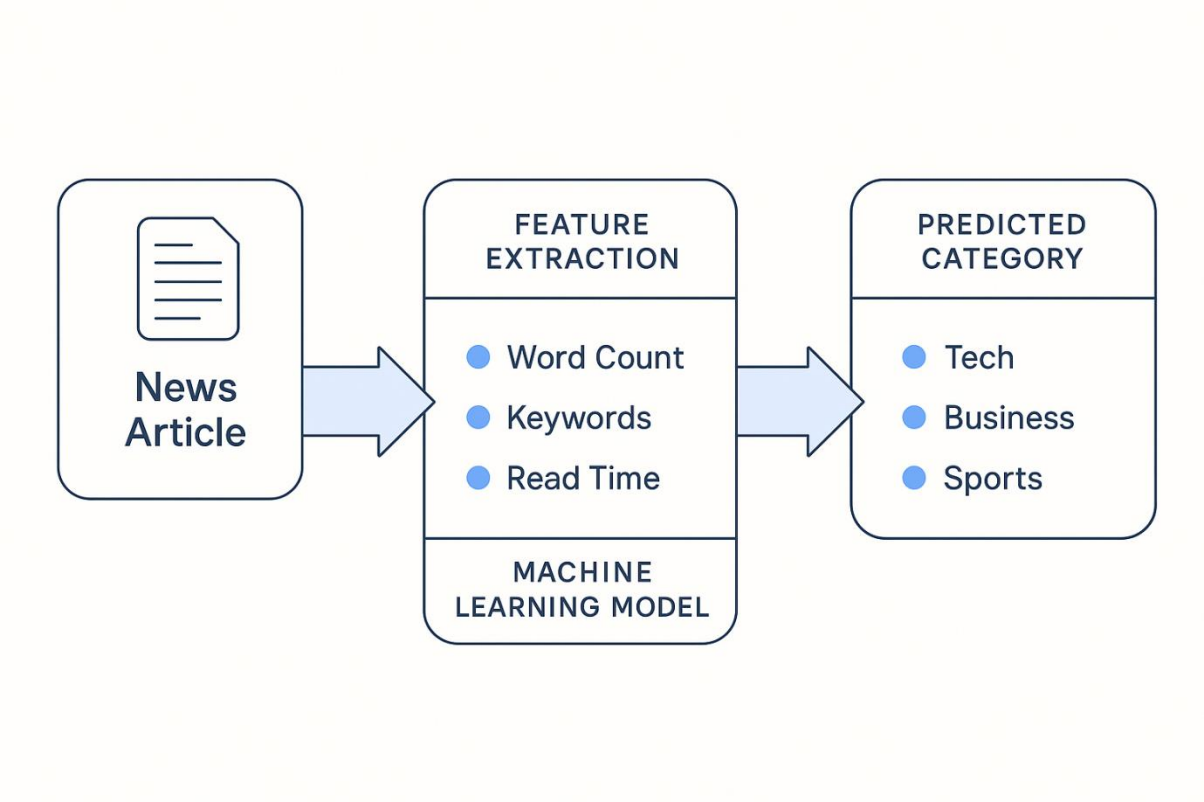
The dataset provided contains basic features like:

- **word_count — the total number of words in the article.**
- **has_keywords — a binary indicator showing whether the article has keywords or not.**
- **read_time — the estimated time to read the article.**
- **category — the target label representing the class of the article.**

The objective is to use this structured data to predict the correct category of a news article using a Machine Learning Classification Model.

For this purpose, we have implemented a Random Forest Classifier, an ensemble learning method that builds multiple decision trees and merges their outputs to improve accuracy and control overfitting. To evaluate the model's performance, we have computed the Confusion Matrix and key metrics such as Accuracy, Precision, and Recall.

This project demonstrates a simple yet effective approach to automated news categorization and highlights the importance of feature engineering and model evaluation in machine learning workflows.



Methodology:

The classification of news articles into specific categories was achieved using a structured machine learning pipeline. The methodology followed a step-by-step approach to ensure clarity, efficiency, and accuracy.

1. Data Collection

The dataset used in this project contains metadata about news articles, including:

- **Word Count: Total number of words in the article.**
- **Has Keywords: A binary indicator of whether the article includes keywords.**
- **Read Time: Estimated reading time for the article.**
- **Category: The label indicating the type of article (Sports, Tech, Business, etc.).**

---

2. Data Preprocessing

- **Label Encoding: Since the `category` column contains categorical text labels, they were converted into numerical values using Label Encoding to make the data compatible with machine learning algorithms.**
- **Feature Selection: The features selected for training were:**
  - **word_count**
  - **has_keywords**
  - **read_time**

---

3. Data Splitting

The dataset was split into two subsets:

- **Training Set: 80% of the data was used to train the model.**
- **Testing Set: 20% of the data was reserved for evaluating the model's performance.**

---

4. Model Selection

A Random Forest Classifier was chosen for this classification task due to its:

- **Robust performance on structured data.**
- **Ability to handle both categorical and numerical variables.**
- **Built-in ensemble approach which helps reduce overfitting.**

---

5. Model Training

The Random Forest Classifier was trained using the training dataset. The model learned to associate the input features with the corresponding news categories through multiple decision trees.

---

6. Model Evaluation

To assess the model's performance, the following evaluation metrics were calculated:

- **Confusion Matrix: Visual representation of prediction results.**

- **Accuracy: The percentage of correctly classified articles.**
- **Precision: The proportion of true positive predictions among all positive predictions.**
- **Recall: The proportion of true positive predictions among all actual positives.**

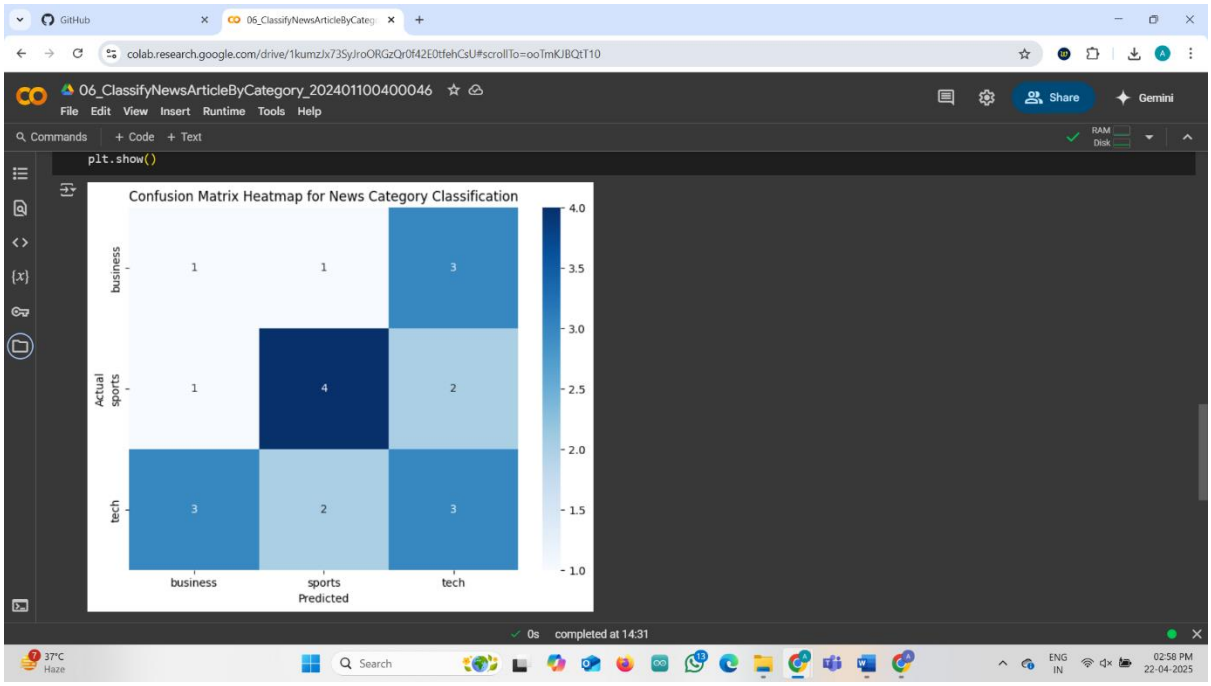
A heatmap of the confusion matrix was also generated for clear visualization of correct and incorrect predictions.

CODE

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv('/content/news_articles.csv')

print(df.head())

print(df.shape)

print(df.describe())

print(df.dtypes)

# Encode target labels
label_encoder = LabelEncoder()
df['category_encoded'] = label_encoder.fit_transform(df['category'])

# Features and target
X = df[['word_count', 'has_keywords', 'read_time']]
y = df['category_encoded']

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

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

# Generate Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
```

```
# Plot Confusion Matrix Heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d',
                    xticklabels=label_encoder.classes_,
                    yticklabels=label_encoder.classes_,
                    cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix Heatmap for News Category Classification')
plt.show()

# Calculate Evaluation Metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')

# Print Evaluation Results
print(f"Accuracy: {accuracy * 100:.2f}%")
print(f"Precision: {precision * 100:.2f}%")
print(f"Recall: {recall * 100:.2f}%")
```

OUTPUT


06_ClassifyNewsArticleByCategory_202401100400046
File Edit View Insert Runtime Tools Help
Share
Gemini
plt.show()
Confusion Matrix Heatmap for News Category Classification
business
sports
tech
Actual
Predicted
1
1
3
1
4
2
3
2
3
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0s completed at 14:31

CREDIT

Special Thanks to
Bikki Gupta Sir