News Article Classification Using Metadata

Problem Statement: Classify News Articles by Category

Use article metadata and keywords to classify news into sports, tech, business, etc.

Jahnavi Nagar

Roll no.

202401100300130

AI - B

Introduction

In today's digital era, news articles are published online in massive volumes and need to be organized for better accessibility. Manually categorizing these articles into predefined groups such as sports, tech, and business is tedious and inefficient. This project focuses on automating this task by using classification techniques based on metadata attributes rather than full-text content. The objective is to accurately classify articles using features such as word count, presence of keywords, and estimated reading time.

Problem Statement

The goal is to develop a machine learning model to classify news articles into categories like sports, tech, and business based on metadata. Unlike traditional text classification that requires natural language processing (NLP) on article content, this project utilizes numerical and binary metadata attributes for classification.

Methodology

Dataset Overview:

The dataset used contains four primary columns:

- word_count: Total number of words in the article.
- has_keywords: A binary indicator showing if the article contains specified keywords.
- read_time: Estimated time to read the article.
- category: The target class (e.g., sports, tech, business).

Data Preprocessing:

- The data is loaded from a CSV file.
- No null values or major inconsistencies were found during exploration.
- Feature and target variables are separated into x and y.

Model Used:

- A Gaussian Naive Bayes (GNB) model was chosen for its efficiency and suitability for numeric data.
- The dataset is split into 80% training and 20% testing sets.

Evaluation Metrics:

- **Accuracy:** Measures the overall correctness of the model.
- **Precision (weighted):** Indicates the proportion of correct positive predictions.
- **Recall (weighted):** Indicates the proportion of actual positives correctly predicted.

Visualization:

- A confusion matrix is generated and visualized using a heatmap to better understand model performance across categories.

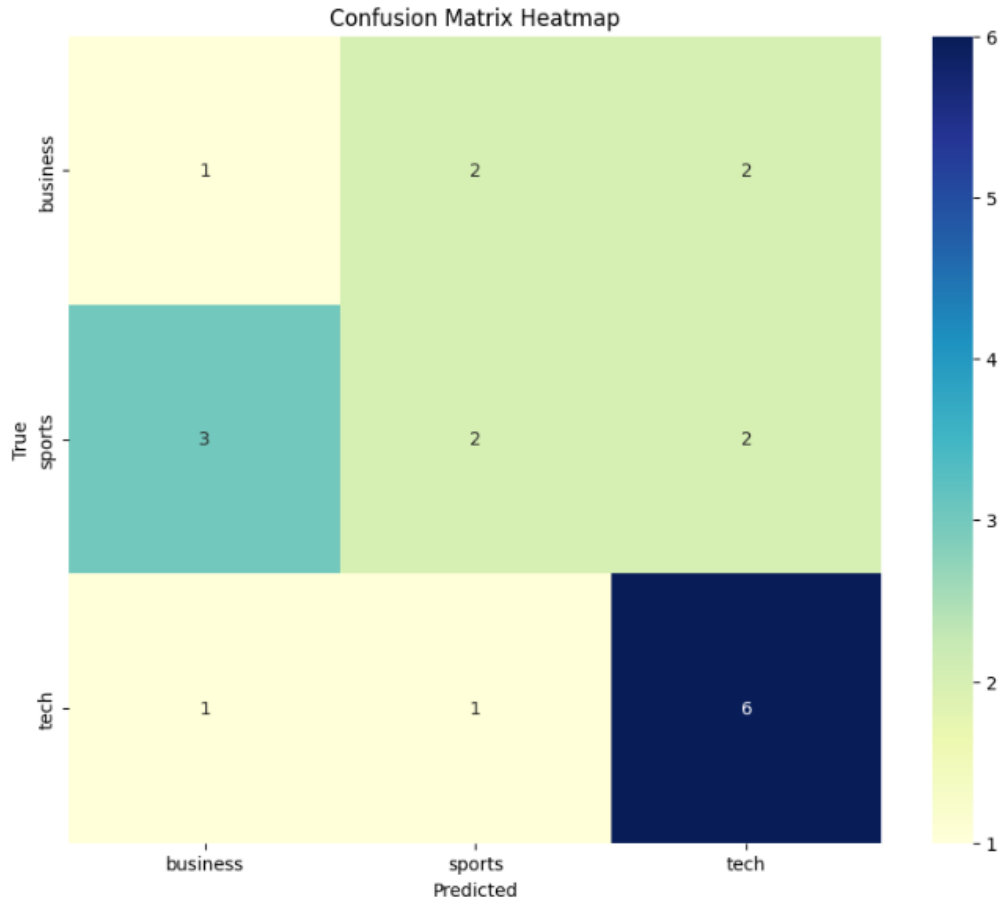
Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
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('news_articles.csv')  # Update with your actual file name

# Check data structure
print(df.head())

# Define features and target
X = df[['word_count', 'has_keywords', 'read_time']]  # input features
y = df['category']  # target labels

# Split the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train a model (Naive Bayes for numeric features)
```

```
model = GaussianNB()
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
acc = accuracy_score(y_test, y_pred)
prec = precision_score(y_test, y_pred, average='weighted', zero_division=0)
rec = recall_score(y_test, y_pred, average='weighted', zero_division=0)

print(f"Accuracy: {acc:.2f}")
print(f"Precision: {prec:.2f}")
print(f"Recall: {rec:.2f}")

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

# Heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='YlGnBu', xticklabels=labels,
yticklabels=labels)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

Output

```
   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
Accuracy: 0.45
Precision: 0.43
Recall: 0.45
```



✓ 0s completed at 2:28 PM

5. Conclusion

This project demonstrates that metadata attributes alone can provide valuable insights for classifying news articles. While the accuracy may be improved with additional features or more complex models like Random Forest or SVM, the current approach is efficient and interpretable. Future work may involve incorporating textual features to further boost performance.

6. References

- Scikit-learn documentation: https://scikit-learn.org/
- Matplotlib documentation: https://matplotlib.org/
- Seaborn documentation: https://seaborn.pydata.org/
- Dataset: news_article.csv
