Assessment Report

on

“Classify Book Genres”

submitted as partial fulfillment for the award of

BACHELOR OF TECHNOLOGY DEGREE

SESSION 2024-25

in

CSE(AI)

By

Name : Pawan Kumar Agrahari

Roll Number : 202401100300169

Section: C

Under the supervision of

“Mayank Lakhotia”

KIET Group of Institutions, Ghaziabad

Intoduction

In this project, we explore a machine learning approach to classify books into different genres based on three attributes: **author popularity**, **book length**, and **number of keywords**. The dataset provided contains these features, and the goal is to predict the genre of the book. The genre is a categorical variable that will be predicted using a classification model.

Data Overview:

- **author_popularity**: A measure of how popular the author is.
- **book_length**: The number of pages in the book.
- **num_keywords**: The number of keywords assigned to the book.
- **genre**: The genre of the book (the target variable).

The Random Forest Classifier, a supervised learning algorithm, is chosen for this task due to its robustness and ability to handle both classification and regression tasks efficiently.

Methodology Overview: The steps taken to solve the problem include:

1. Data Preprocessing
2. Feature Extraction
3. Model Training (Random Forest Classifier)

Methodology

Data Preprocessing:

- The dataset is loaded and cleaned. Non-numeric columns (such as the genre) are encoded into numerical labels using **LabelEncoder** to make them compatible with machine learning models.
- The dataset is then split into a training set and a testing set (80% for training and 20% for testing).

Model:

- The **Random Forest Classifier** is used as the model to classify the books into genres. The classifier is initialized with 100 estimators (trees) and trained on the training set.

Model Evaluation:

- After training, the model is evaluated using the confusion matrix, which visualizes the classification results. In addition, evaluation metrics such as **accuracy**, **precision**, **recall**, and **F1-score** are computed.

The Random Forest Classifier is chosen because of its high performance and ability to handle overfitting, especially when the number of features is large.

Steps Involved:

1. **Preprocessing**: Clean and prepare the data for model training.
2. **Training**: Train the Random Forest Classifier on the training set.
3. **Testing**: Test the trained model on unseen data.
4. **Evaluation**: Use evaluation metrics to assess the model's performance.

Code

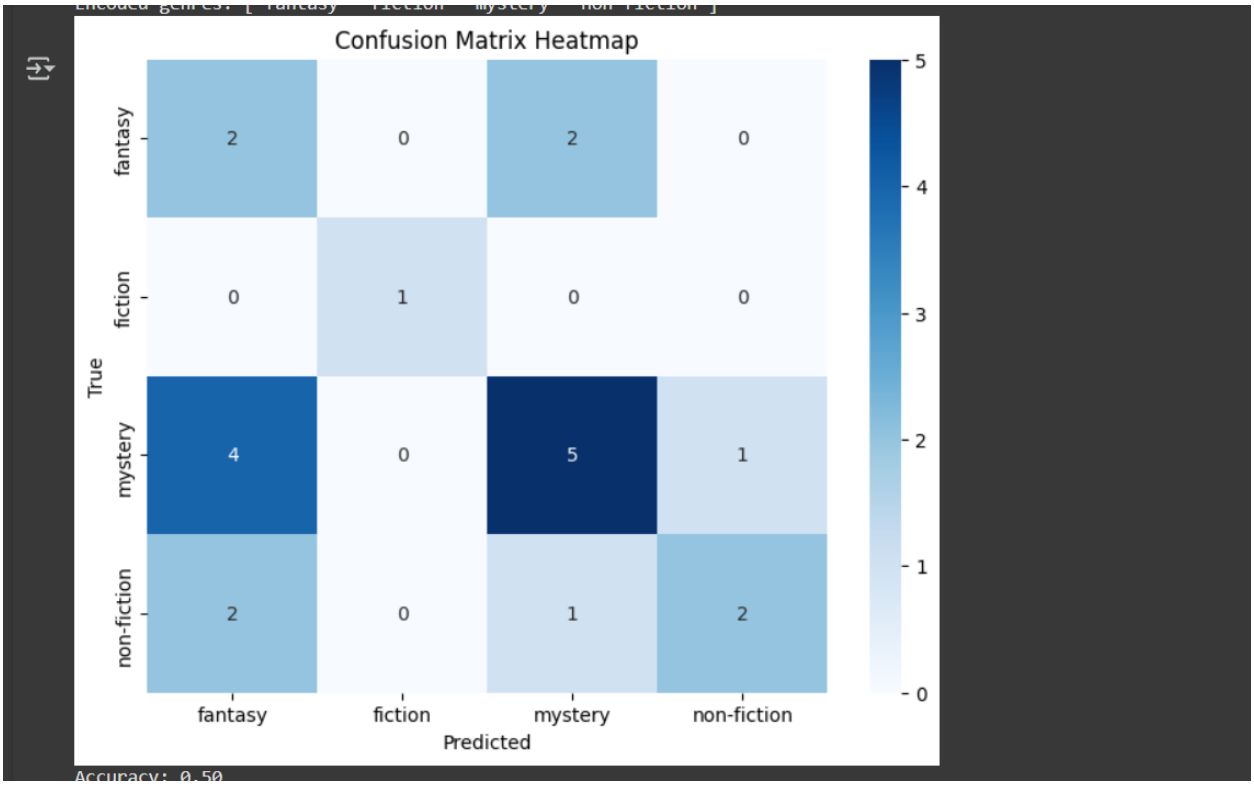
```
# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset (Assuming the data is in a CSV file called
'book_data.csv')
data = pd.read_csv('/content/book_genres.csv')

# Show the first few rows of the dataset
print(data.head())

# Extract features and target variable
X = data[['author_popularity', 'book_length', 'num_keywords']]  #
Features
y = data['genre']  # Target label (genre)

# Encode the categorical target variable (genre) using LabelEncoder
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)
```

```
# Show the encoded labels for genres
print("Encoded genres:", label_encoder.classes_)

# Split data into training and test sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y_encoded,
test_size=0.2, random_state=42)

# Initialize the Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier on the training data
clf.fit(X_train, y_train)

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

# Compute confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Generate a heatmap of the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='g', cmap='Blues',
xticklabels=label_encoder.classes_, yticklabels=label_encoder.classes_)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
# Calculate Accuracy
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Calculate Precision (weighted average for multi-class)
precision = precision_score(y_test, y_pred, average='weighted')
print(f'Precision: {precision:.2f}')

# Calculate Recall (weighted average for multi-class)
recall = recall_score(y_test, y_pred, average='weighted')
print(f'Recall: {recall:.2f}')

# Calculate F1-Score (weighted average for multi-class)
f1 = f1_score(y_test, y_pred, average='weighted')
print(f'F1 Score: {f1:.2f}')
```

Output/Result

```
   author_popularity  book_length  num_keywords    genre
0          41.052297          776             5  mystery
1          48.950098          674             5  mystery
2           2.323401          633            19  fantasy
3          41.564184          169            12  mystery
4          65.129649          992            18  fantasy
Encoded genres: ['fantasy' 'fiction' 'mystery' 'non-fiction']
```




```
Accuracy: 0.50
Precision: 0.58
Recall: 0.50
F1 Score: 0.52
```

References / Credits

1. Dataset:

- **Book Genres Dataset**: Ensure to credit the source of the book_genres.csv dataset (e.g., if from Kaggle or another repository).

2. Libraries/Tools:

- **Pandas**: McKinney, W. (2010) for data manipulation.
- **Scikit-learn**: Pedregosa, F., et al. (2011) for machine learning.
- **Seaborn**: Waskom, M. L. (2021) for statistical visualization.
- **Matplotlib**: Hunter, J. D. (2007) for plotting.

3. Model:

- **Random Forest**: Breiman, L. (2001) for Random Forest classifier.

4. Evaluation Metrics:

- Standard metrics (Accuracy, Precision, Recall, F1) from **Scikit-learn**.

5. Visualizations:

- **Confusion Matrix Heatmap**: Generated with **Seaborn** and **Matplotlib** libraries.