Classify Book Genres

A Project Report

Submitted by

Rachit Gupta – (202401100300189)

In partial fulfillment for the award of the degree

Of

Bachelor of computer science and engineering(artificial intelligence)

KIET GROUP OF INSTITUTIONS (GHAZIABAD)

➢ Introduction

The goal of this project is to classify book genres based on metadata features. Genre classification can help readers find content tailored to their preferences and assist digital platforms in better content recommendation. The features used in this dataset are author popularity, book length, and the number of keywords associated with the book. We apply a machine learning model to predict the genre of a book based on these attributes.

➢ Methodology

We used a Random Forest Classifier to perform the classification task. The dataset was split into training and testing sets (80%-20%). The classifier was trained using the training data and evaluated on the test data. A confusion matrix was generated to visualize the prediction results. Accuracy, precision, and recall metrics were also calculated to evaluate model performance.

Steps followed:

1. Load and preprocess the dataset.
2. Split the dataset into training and testing sets.
3. Train a Random Forest classifier.
4. Predict on the test set.
5. Generate and plot the confusion matrix.
6. Calculate accuracy, precision, and recall.

➢ Code

```
import pandas as pd
from sklearn.model_selection import train_test_split
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 dataset
df = pd.read_csv("/content/book_genres.csv")  # Adjust path if needed

# Split features and target
X = df.drop("genre", axis=1)
y = df["genre"]

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

# Train classifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train, y_train)

# Predict
y_pred = clf.predict(X_test)
```

```
# Confusion Matrix
cm = confusion_matrix(y_test, y_pred, labels=clf.classes_)

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
        xticklabels=clf.classes_, yticklabels=clf.classes_)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

# Evaluation Metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted', zero_division=0)
recall = recall_score(y_test, y_pred, average='weighted', zero_division=0)

print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
```

➢ Output/Result

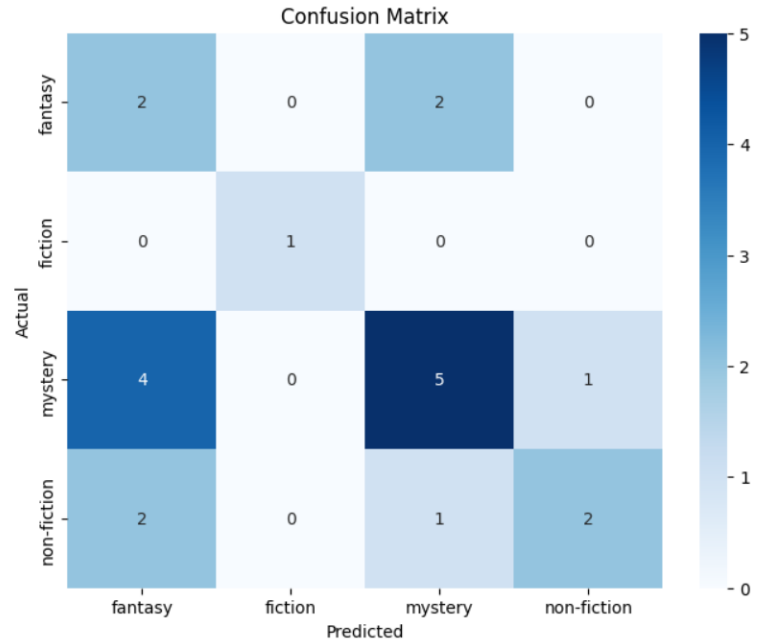
Confusion Matrix Heatmap:

Evaluation Metrics:

Accuracy: 0.50

Precision: 0.58

Recall: 0.50




```
Accuracy: 0.50
Precision: 0.58
Recall: 0.50
```

➢ References/Credits

1. Dataset provided for the exam
2. Scikit-learn documentation: https://scikit-learn.org
3. Matplotlib and Seaborn for visualization

➢ Files Uploaded to GitHub:

- Jupyter Notebook (.ipynb)
- PDF Report
- README.md