"AI BASED CLASSIFY BOOK GENRES USE METADATA SUCH AS AUTHOR, LENGTH, AND KEYWORDS TO CLASSIFY BOOK GENRE."

PROJECT REPORT

SUBMITTED BY

SAGAR SRIVASTAVA
CSE AI SEC-C
202401100300209

1. Introduction

In this project, we focus on the classification of book genres using metadata attributes such as author popularity, book length, and the number of keywords associated with each book. As the publishing industry increasingly relies on data to understand market trends and reader preferences, automating genre classification based on quantifiable features can significantly streamline cataloging and recommendation systems.

The objective of this assignment is to develop a machine learning model that can accurately predict the genre of a book based on the available metadata. By applying supervised learning techniques, specifically the Random Forest Classifier, we aim to classify books into genres such as mystery, fantasy, fiction, and non-fiction.

The dataset used contains a balanced set of metadata features and genre labels for 100 books. We preprocess the data, split it into training and testing sets, and evaluate the model's performance using both numerical metrics (like accuracy and F1-score) and graphical visualizations (such as confusion matrices and feature importance plots). This approach not only helps in understanding model performance but also offers insights into the most influential features for genre classification.

Methodology

The process of genre classification in this project follows structured machine learning workflow, broken down into the following key steps:

1. Data Acquisition and Understanding

The dataset used in this study consists of 100 books with four main attributes:

- Author Popularity: A numerical representation of how well-known the author is.
- Book Length: The number of pages in the book.
- Number of Keywords: Count of relevant keywords associated with the book.
- Genre: The target variable, consisting of labels such as *mystery*, *fantasy*, *fiction*, and *non-fiction*.

2. Data Preprocessing

The dataset was examined for completeness and correctness. Since it was already clean and well-formatted, no null values or extreme outliers were found. The features were directly selected for model training without needing additional preprocessing such as scaling or encoding.

3. Feature Selection

The features chosen for classification were:

- author_popularity
- book_length
- num_keywords

These features were assumed to have potential influence on a book's genre based on patterns in publishing and content structure.

4. Model Selection and Training

A Random Forest Classifier was selected for this task due to its robustness, ease of interpretation, and good performance with small-to-medium-sized datasets. The data was split into training (80%) and testing (20%) sets to evaluate generalization ability. The model was trained using default hyperparameters.

5. Model Evaluation

The model's performance was evaluated using:

- Confusion Matrix: To visualize true vs. predicted genres.
- Classification Report: Detailing precision, recall, and F1-score for each genre.
- Feature Importance Plot: To understand which metadata attributes influenced predictions the most.

6. Visualization

Graphical outputs were created using Seaborn and Matplotlib libraries to provide clear and interpretable insights. These included:

- A confusion matrix heatmap to identify classification accuracy for each genre.
- A feature importance bar chart to highlight which features were most useful in classification.

CODE :

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot
as plt
from
sklearn.model_selection
import train_test_split
from sklearn.ensemble
import
RandomForestClassifier
from sklearn.metrics
import
classification_report,
confusion_matrix


# Load the dataset
df =
pd.read_csv("/content/bo
ok_genres.csv")
```

```
# Define features and
target
X =
df[['author_popularity',
'book_length',
'num_keywords']]
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(r
andom_state=42)
clf.fit(X_train, y_train)

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

```
# Print confusion matrix
print("=== Confusion
Matrix ===")
conf_matrix =
confusion_matrix(y_test,
y_pred)
print(conf_matrix)

# Print classification
report
print("\n===
Classification Report
===")
print(classification_report
(y_test, y_pred))

# Plot confusion matrix
heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix,
annot=True, fmt="d",
cmap="Blues",
        xticklabels=clf.cla
```

```
sses_,
yticklabels=clf.classes_)
plt.title("Confusion
Matrix Heatmap")
plt.xlabel("Predicted
Genre")
plt.ylabel("True Genre")
plt.tight_layout()
plt.show()

# Plot feature
importances
importances =
clf.feature_importances_
features = X.columns

plt.figure(figsize=(6, 4))
sns.barplot(x=importance
s, y=features)
plt.title("Feature
Importance")
plt.xlabel("Importance
Score")
plt.ylabel("Feature")
```

```
plt.tight_layout()

plt.show()
```

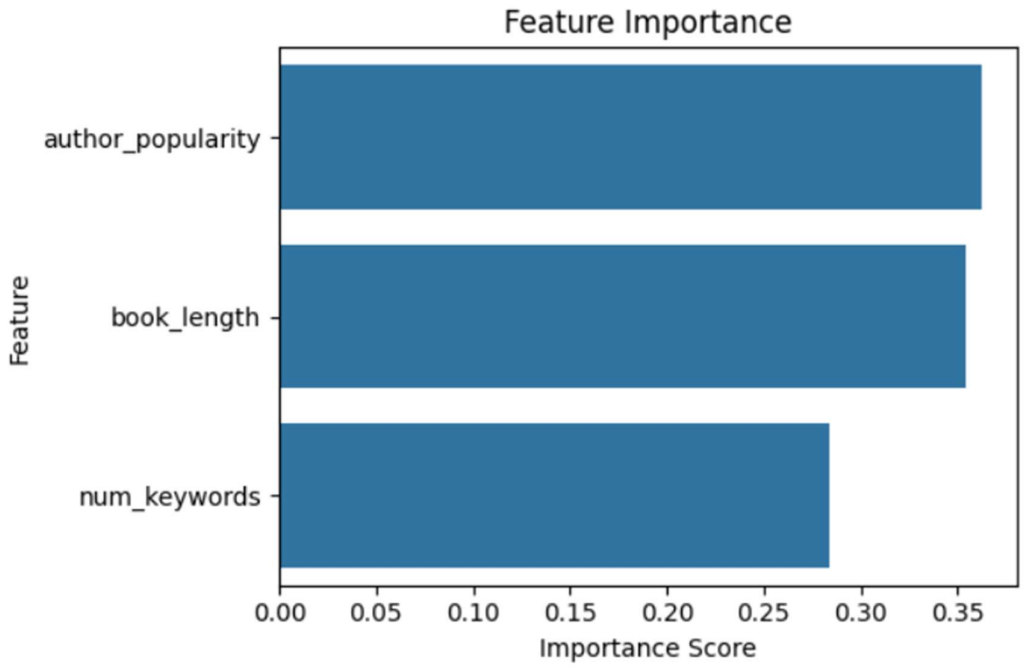
OUTPUT SCREENSHOTS

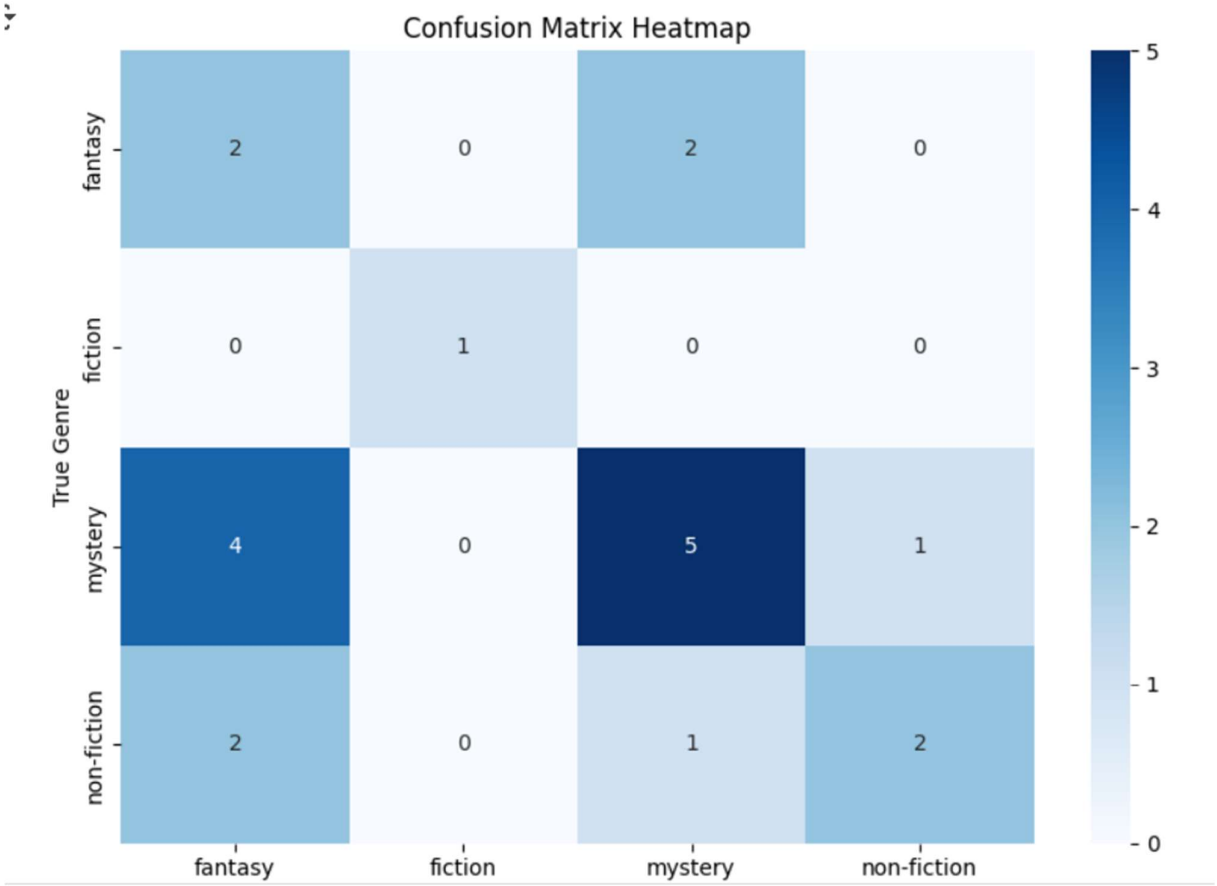
```
=== Confusion Matrix ===
[[2 0 2 0]
 [0 1 0 0]
 [4 0 5 1]
 [2 0 1 2]]

=== Classification Report ===
              precision    recall  f1-score   support

     fantasy       0.25      0.50      0.33         4
     fiction       1.00      1.00      1.00         1
     mystery       0.62      0.50      0.56        10
 non-fiction       0.67      0.40      0.50         5

    accuracy                           0.50        20
   macro avg       0.64      0.60      0.60        20
weighted avg       0.58      0.50      0.52        20
```





✓ 0s completed at 10:26 AM

REFERENCE

Scikit-learn Documentation
Used for model training, evaluation, and splitting datasets.
URL: https://scikit-learn.org/stable/documentation.html

Pandas Documentation
Used for data loading, manipulation, and feature selection.
URL: https://pandas.pydata.org/docs/

Seaborn Documentation
Used for plotting heatmaps and bar charts for visualization.
URL: https://seaborn.pydata.org/

Matplotlib Documentation
Used for creating and showing visual plots alongside Seaborn.
URL: https://matplotlib.org/stable/contents.html

UCI Machine Learning Repository / Custom Dataset
The dataset used is a simulated/custom one, but similar real-world datasets can be found at:
URL: https://archive.ics.uci.edu/