BOOK RECOMMENDATION SYSTEM

R.P.R.T.N.Samaraweera
Department of Computer Engineering
Faculty of Engineering
University of Ruhuna
Galle,SriLanka
samaraweera_rprtn_e22@engug.ruh.ac.lk

D.M.C.L.Dissanayaka
Department of Computer Engineering
Faculty of Engineering
University of Ruhuna
Galle,SriLanka
dissanayaka_dmcl_e22@engug.ruh.ac.lk

Abstract— **This project explores two machine learning models, Decision Tree and K-Nearest Neighbors (KNN), to predict book ratings. As an undergraduate engineering student, the goal is to see how well these models perform and understand which one might be better for this task.**
In the Decision Tree model, we train our model to show examples. We also do tuning to make it better at guessing book ratings. The KNN model looks at the neighbors of a book to make its prediction. We adjust how many neighbors it should consider to get the best results.
We then test both models to see how accurate they are at predicting book ratings. We use simple charts to show how well they do and explain the results easily. Additionally, we let people input information about a book to get a prediction, making it interactive and practical.
The project also looks at how the models learn. We show graphs to see if they get better at predicting as they see more examples. This helps us understand if they're too simple or too complex.
This study provides a straightforward look at how these models work for predicting book ratings. It's a useful project for understanding machine learning and how it can be applied to everyday problems.
(*Abstract*)

*Keywords—: **Machine Learning, Decision Tree, K-Nearest Neighbors, Book Ratings, Interactive Prediction, Learning Process**. (key words)*

I. INTRODUCTION

With the tremendous advancements in technology, there has been a huge increase in the amount of data released on the internet, so it has become difficult to choose which data is the most accurate and correct. Recommendation systems aim to solve this type of problem. With the assistance of them one can quickly get to material information without looking through the web physically. The main objective of this project is to facilitate people who love to read and to encourage more people to read. And also A reader cannot understand whether a book is good or not just by picking it up. So these systerm can be used to get a rough understanding of the book using only the information on the cover of the book. But in this project also the same method has been done using two types of algorithms. Its main purpose is to choose the most suitable algorithm for this task from the two algorithms. Different methods are used to test the nudity of these two algorithms

There are so many advantage and few disadvantage related this algorithms we move to KNN algorithm advantage of decision tree algorithm is Easy to understand and implement and Captures user preferences based on similarity and move decision tree algorithms it Offers a clear decision-making process. And it can Handles both numerical and categorical data. It is easy our implementation. [1] [2]

II. METHODOLOGY

A. Our data set

Our dataset serves as the main thing of our book recommendation system. We obtained the data from the 'Books_Data_Clean.csv' file in our code, and also we got this data set in https://www.kaggle.com/datasets/thedevastator/books-sales-and-ratings . [3] a carefully curated collection of information about various books. This dataset have many features, including Publishing Year, Book Name, Author, language_code, Author_Rating, Book_average_rating, Book_ratings_count, genre, gross sales, publisher revenue, sale price, sales rank, Publisher and units sold.The initial exploration of our dataset, comprising 1070 entries, provided crucial insights into the distribution of book ratings.

To enhance the interpretability of our data, we began by renaming certain columns for improved readability. For example, we changed 'sale price' to 'sale_price,' 'Book Name' to 'Book_Name,' and 'Publishing Year' to 'Publishing_Year.' Additionally, we introduced a categorical variable, 'Rating,' derived from the 'Book_average_rating,' using a mapping function. This function categorized ratings into labels such as book
'Book_average_rating among 2.5-3 it is 'very bad,'
'Book_average_rating among 3-3.5 it is 'bad,'
'Book_average_rating among 3.5-4 it is 'good,'
'Book_average_rating among 4-4.5 'very good,'
'Book_average_rating among 4.5-5 'excellent.'

B. Pre-processing

1) Handling Missing Values:

We meticulously addressed missing values in our dataset. Initial inspection revealed null values in 'Publishing_year' and 'Book_Name' columns.there are 23 null values in 'Book_Name' Colum and there are 53 null values in 'Publishing_year' colum and one null value in Publishing_Year cplum To maintain data integrity, we decided to omit rows with missing values in these particular columns. Moreover, we fill the missing values in the 'Language_Code' column with the category label 'Unknown'.

a) Feature Encoding:

As part of preparing our data for machine learning models, we employed one-hot encoding for the categorical variable 'Author_Rating.' And 'sale_price' This process created dummy variables that could be used in our subsequent algorithms.

b) Dataset Splitting:

To evaluate our models effectively, we split the dataset into training and testing sets. This division, achieved through the 'train_test_split' function from the 'sklearn.model_selection' module, allocated 70% of the data for training and the remaining 30% for testing.

2) Decision Tree Implementation

Our first algorithm, the Decision Tree classifier, was implemented using the 'DecisionTreeClassifier' from the 'sklearn.tree' module. Following training on the designated training data, the model made predictions on the test data. The accuracy of the Decision Tree model was assessed using the 'accuracy_score' function, indicating how well it performed in predicting book ratings.

3) Data Tuning for decision tree

To increse the Decision Tree model's performance, we used hyperparameter tuning through GridSearchCV. The grid search explored various combinations of hyperparameters, such as 'criterion,' 'max_depth,' 'min_samples_split,' and 'min_samples_leaf.' The best hyperparameters, identified through this process, this is we used hyperparameters 'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split': 2, were then utilized to create the best Decision Tree model. Subsequent evaluations demonstrated the enhanced accuracy of this tuned model.

4) K-Nearest Neighbors Implementation

Our second algorithm, the K-Nearest Neighbors (KNN) classifier, after a similar process. Beginning with the creation of a KNN model using 'KNeighborsClassifier' from 'sklearn.neighbors,' we further scaled the training data using 'StandardScaler' to normalize the features. This normalization ensures that no particular feature dominates the model due to differences in scale.

5) Data Tuning for KNN

Similar to the Decision Tree, the KNN model hyperparameter tuning using GridSearchCV. We systematically explored different values for 'n_neighbors,' representing the number of neighbors considered during classification. The optimal 'n_neighbors' value, discovered through the grid search, resulted in an improved KNN model.

6) User Input and Predictions

To demonstrate the practicality of our recommendation system, we add a user interaction component. Users can input the author's rating and book price, and the models predict the likely rating category ('very bad,' 'bad,' 'good,' 'very good,' or 'excellent') for the user's book.

7) Visualizations and Comparisons

To compare the models' performance, we used visualizations. Confusion matrices provided insights into the accuracy of predictions, while ROC curves and Precision-Recall curves showcased the models' discriminatory power. These visualizations help to get comparison what are the effective ,specific strengths and accurate algorithm for our recommendation system.

8) Classification Reports and Learning Curves

Classification reports offer detailed metrics such as precision, recall, and F1-score for both the Decision Tree and KNN models. Learning curves visualize how model accuracy evolves with increasing training data.

C. About algorithms

In the process of selecting the algorithm for our book recommendation system, first we understanding of our problem and aiming to find the most effective solution. We start by acquiring a suitable dataset, then we ensure that all data points were labeled. Our focus then shifted to supervised learning algorithms, which are well-suited for scenarios where we have labeled data and find to make predictions.

By considering various aspects of supervised learning algorithms, we selected in on two specific techniques: K-Nearest Neighbors (KNN) and Decision Tree algorithms. These algorithms are particularly relevant for our task, as they are capable of handling labeled data and are good for classification problems.

Moving forward with the solution, we specified that our goal was to achieve binary classification from the trained data model. This means we aimed to categorize books into five classes, likely based on their ratings. As a result, we made the decision to utilize supervised classification machine learning algorithms, specifically KNN and Decision Tree algorithms, for training our dataset.

There is a important thing that throughout this entire process, we focused on accuracy and effectiveness. and also Ability to enter data from the Book cover and make a decision as soon as the reader picks up the book The choice of algorithms was made based on their compatibility with labeled data.

Both algorithms were used in their standard forms without modification in this implementation . The implementation of both algorithms in this project used the scikit-learn library in Python. The code integrates the Decision Tree and KNN classifiers.

1) Decision tree

Using decision tree equation we use some important equation for find best path for tree. it call Information Gain it has another name Entropy. The decision tree splits data by choosing features that maximize information gain Entropy, denoted as **H(S),** measures the impurity or disorder in a set **S** The formula for entropy is given by:

$\mathbf{H(S)=-\sum pi\cdot log2(pi)}$ [3]

pi is the proportion of instances

2) K-Nearest Neighbors (KNN)

For classification, KNN counts the occurrences of each class among the k nearest neighbors of a data point and assigns the class with the highest count.

KNN relies on a distance metric, commonly Euclidean distance

$\mathbf{d(p,q)=\sum^{n} (pi-qi)^{2}}$

to determine the proximity of data points [4]

RESULT*(Heading 4)*

The first step was to calculate how many missing and null values in our data set were. Then we found some null and missing values. Then we handle that using this method. Rows with missing values in the 'Publishing_Year' and 'Book_Name' columns were dropped. Missing values in the 'language_code' column were filled with 'unknown the calculating Accuracy we got these values before data tuning. The initial Decision Tree model achieved an accuracy of approximately 82.9% on the test data. The KNN model achieved an initial accuracy of approximately 83.8% on the test data. Using Hyperparameter Tuning we found some parameters corresponding to two algorithms. GridSearchCV was employed to find the best hyperparameters for the Decision Tree model. Best Hyperparameters for Decision Tree Model: {'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split'2}. Hyperparameter Tuning: GridSearchCV was used to find the optimal number of neighbors (k). Best Hyperparameter: {'n_neighbors': 16}.After Data Tuning the accuracy was increased. Improved Model Accuracy: The best-tuned Decision Tree model achieved an improved accuracy of approximately 83.8% on the test data. Improved Model Accuracy: The best-tuned KNN model achieved an improved accuracy of approximately 84.8% on the test data. To assess the performance of these models, we employed Confusion Matrices, a valuable tool for understanding the accuracy and challenges of predictionsText heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic. If there are two or more sub-topics, the next level head (uppercase Roman numerals) should be used and, conversely, if there are not at least two sub-topics, then no subheads should be introduced. Styles named “Heading 1”, “Heading 2”, “Heading 3”, and “Heading 4” are prescribed.

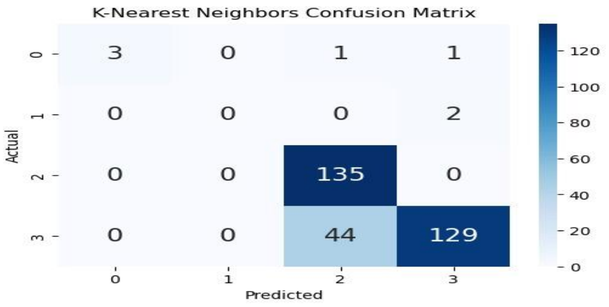


Figure 1:K-Nearest Neighbors Confusion Matrix

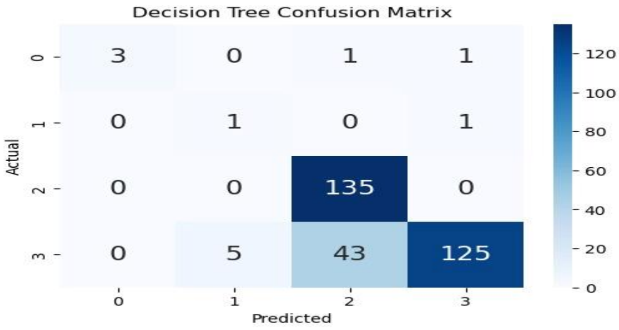


Figure 2:Decison Tree Confusion Matrix

In the K-Nearest Neighbors Confusion Matrix, we can obtain these things.

- True Positives (TP): The model correctly predicted 135 instances as "very good."
- True Negatives (TN): No instances were correctly predicted for "bad" or "excellent."
- False Positives (FP): There was a mistake in predicting "bad" once and "very good" once when it was not.
- False Negatives (FN): The model missed predicting "excellent" in 44 instances.

In the Decision Tree Confusion Matrix we can obtain these things

- True Positives (TP): The model correctly predicted 135 instances as "very good."
- True Negatives (TN): It correctly predicted "good" once and "very good" 125 times.
- False Positives (FP): There was a mistake in predicting "bad" once and "good" five times when it was not.
- False Negatives (FN): The model missed predicting "bad" once and "excellent" once.

KNN excelled in correctly predicting "very good" instances, with zero false positives in this category. However, it struggled with predicting "excellent," with 44 false negatives. The Decision Tree also performed well in predicting "very good."It had fewer false negatives for "excellent" compared to KNN but more false positives in other categories. However, both models had difficulty predicting instances of "bad" and "excellent."

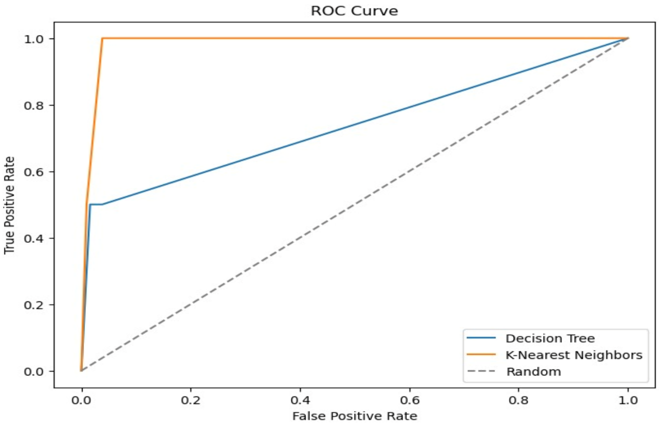
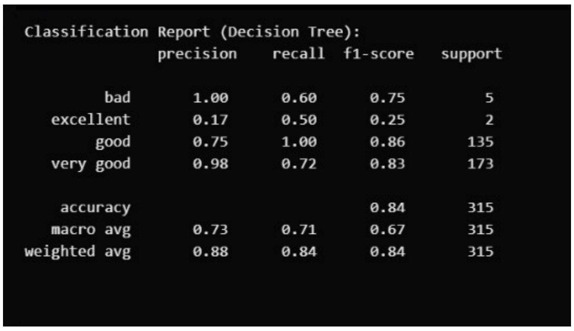


Figure 3:ROC Curve

According to this graph, the Decision Tree classifier has a high true positive rate with a low false positive rate, including a good performance. the K-Nearest Neighbors classifier also has a high true positive rate, but its false positive rate is slightly higher than that of the decision tree. but both Decision and KNN classifiers are better than a random guess.

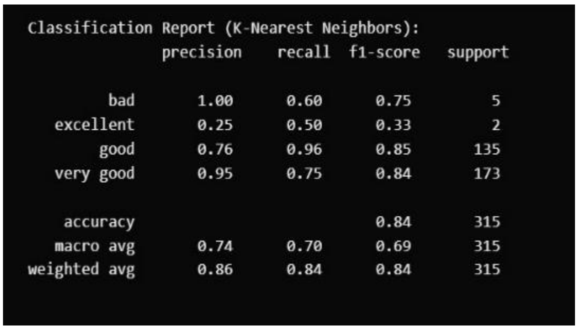


```
Classification Report (Decision Tree):
              precision    recall  f1-score   support

         bad       1.00      0.60      0.75         5
   excellent       0.17      0.50      0.25         2
        good       0.75      1.00      0.86       135
   very good       0.98      0.72      0.83       173

    accuracy                           0.84       315
   macro avg       0.73      0.71      0.67       315
weighted avg       0.88      0.84      0.84       315
```

Figure 4:Classification Report (Decision Tree)



```
Classification Report (K-Nearest Neighbors):
              precision    recall  f1-score   support

         bad       1.00      0.60      0.75         5
   excellent       0.25      0.50      0.33         2
        good       0.76      0.96      0.85       135
   very good       0.95      0.75      0.84       173

    accuracy                           0.84       315
   macro avg       0.74      0.70      0.69       315
weighted avg       0.86      0.84      0.84       315
```

Figure 5:Classification Report (KNN)

according to this table the Decision Tree and K-Nearest Neighbors (KNN) classifiers, both exhibit an accuracy of approximately 84%. The Decision Tree excels in predicting "good" and "very good" books, achieving accuracies of 86% and 83%, respectively. However, it encounters challenges in accurately classifying "bad" and "excellent" books. On the other hand, KNN displays a balanced performance, particularly in predicting "bad" books, but faces difficulties with "excellent" ones. Overall, both models demonstrate competence, with the Decision Tree having specialized strengths, while KNN provides a more balanced approach to book rating predictions.

Discussion and Conclusion *(Heading 5)*

Discussion:

Looking at how well the Decision Tree and K-Nearest Neighbors (KNN) predict book ratings, both did pretty well, scoring around 84%. The Decision Tree is like an expert in certain types of books, being good at predicting "good" and "very good" ones. On the other hand, KNN is more balanced, especially in predicting "bad" books. [5]

When it comes to understanding how these models work, the Decision Tree is like an open book – easy to follow. But KNN, not so much. making it harder to know why it predicts certain things. Choosing between them depends on what you value more clear explanations or balanced predictions.

Ethical Aspects:

When it comes to ethics in using Machine Learning models for book ratings, we've got to watch out for biases in the models. We need to make sure they treat everyone fairly. Besides that, we've got to be careful with people's privacy – their info should be kept safe. To do this right, we need to get permission from users and be clear about how we're using their data. It's like being open and honest with them. Security is a big deal too – we need strong measures to stop unauthorized access and protect against any data leaks.

Making sure the models work for everyone is key. They should be easy for people with different likes and backgrounds to use. We also need to check how the models might affect things in the long run. Keeping an eye on the big picture helps us fix any problems that might pop up later. Using a fair mix of examples in training the models helps avoid unfair results. We also need to be ready to fix any unexpected issues and talk to the people using the models to get their thoughts. Following the rules and laws about data protection is a must – it keeps everything ethical and legal. When we take all these things into account, using Machine Learning models for book ratings can be done in a way that's responsible, trustworthy, and good for everyone.

Conclusion:

In wrapping up our book recommendation project, it's been a fascinating exploration into the realms of machine learning with the Decision Tree and K-Nearest Neighbors models. It's like having two book experts – one good with specific types, and the other giving a balanced view.

But, it's not just about tech stuff; we've also thought a lot about doing it the right way. Being fair, open, and keeping your info safe has been super important to us. We've worked hard to be fair in how we pick books and make sure everything is legal and easy for you to use.

This project isn't just about computer code; it's about using tech in a good way. We want our book suggestions to be reliable, fair, and something everyone can enjoy. By intricately weaving together technology and ethical considerations, our book recommendation project strives to be a beacon of reliability, inclusivity, and positivity for all book enthusiasts embarking on their literary journeys. [6]

Acknowledgment *(Heading 5)*

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

D. *References*

[1] [Online]. Available: https://aspiringyouths.com/advantages-disadvantages/knn-algorithm/.

[2] [Online]. Available: https://learn.g2.com/k-nearest-neighbor.

[3] [Online]. Available: https://www.kaggle.com/datasets/thedevastator/books-sales-and-ratings.

[4] [Online]. Available: https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c.

[5] [Online]. Available: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.

[6] [Online]. Available: https://www.researchgate.net/publication/221608315_Trust_in_recommender_systems.

[7] [Online]. Available: https://careerfoundry.com/en/blog/data-analytics/ethical-considerations-in-ai/.