Machine Learning Assignment

PROJECT REPORT

TEAM ID : 19

Project ID : 19

Sentiment Analysis for Book Character Labelling (good or evil)

Name	SRN
B TEJA DEEP SAI KRISHNA	PES2UG23CS135
BOBBA KOUSHIK	PES2UG23CS133

Problem Statement

This project aims to solve the text classification problem of determining the moral alignment of fictional characters based on their textual descriptions. Given a short descriptive text about a book character, the goal is to classify the character as either "Good" or "Bad".

Objective / Aim

The primary objective is to build and evaluate a machine learning model capable of accurately performing sentiment analysis on character descriptions to predict their moral verdict. The project compares five different classification algorithms to identify the most effective model for this task.

Dataset Details

 Source: This is a custom-collected dataset. Character descriptions were manually gathered from online literary resources such as LitCharts, SparkNotes, GradeSaver, and CliffsNotes for a variety of books.

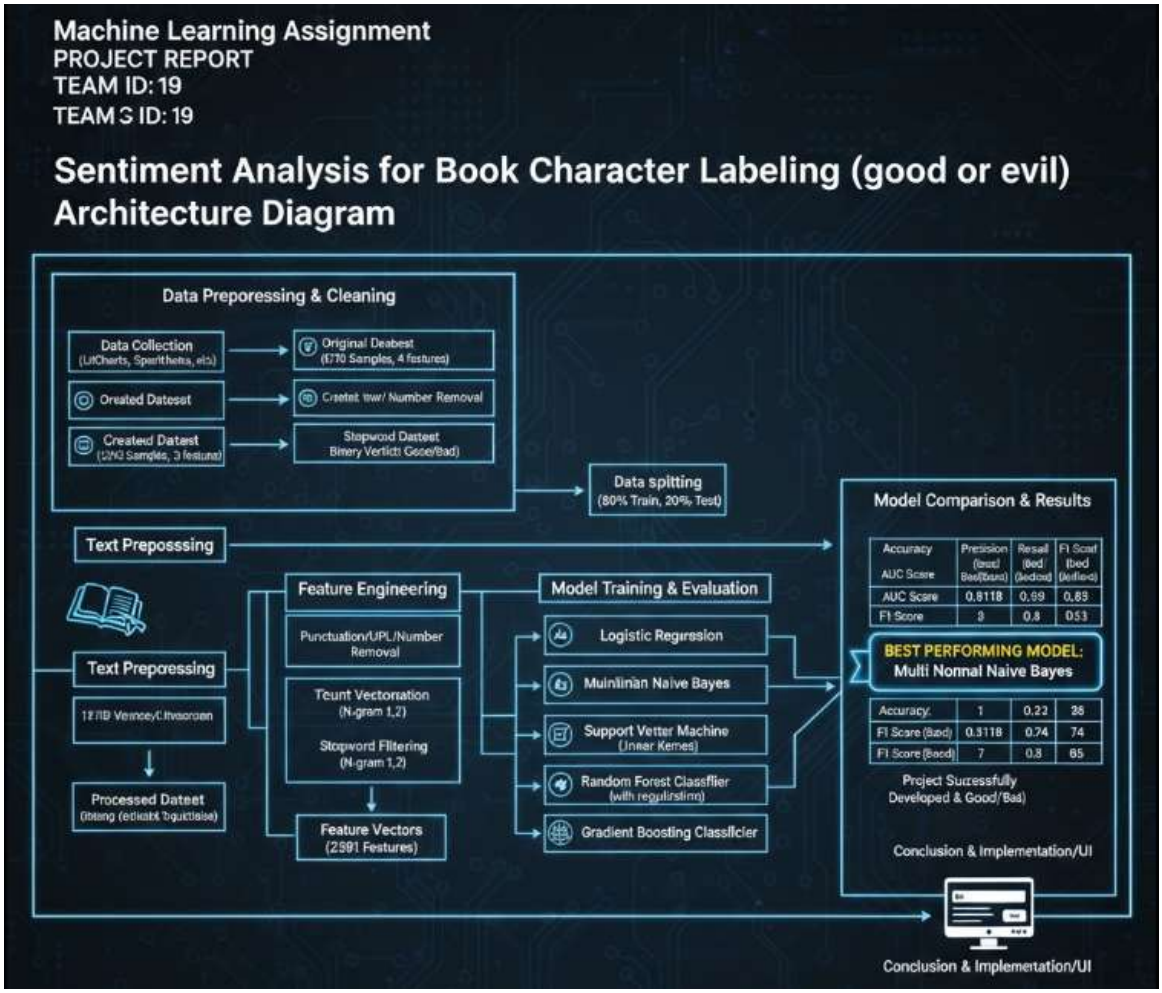
 Size:

- **Original Dataset: 670 Samples, 4 features.**
- **Created Dataset: 1442 samples, 3 features.(from the info of books from the original dataset created a dataset manually which contains character, description, Character Verdict)**
- **Processed Dataset (for binary classification): 1274 samples.**

 Key Features: The primary feature used for prediction is the Description of the character.

 Target Variable: Verdict (Categorical, processed into binary: 'Good' or 'Bad').

Architecture Diagram



Methodology

The project followed a systematic workflow:

- **Data Collection & Initial Analysis:** The project began by loading the Character.csv dataset. An initial analysis revealed 1,442 records and no missing values. The target variable, Verdict, was found to have seven unique classes.
- **Data Preprocessing & Cleaning:** The problem was simplified to a binary classification task. Verdicts like 'Neutral' and 'Bad (Situation)' were mapped to 'Bad', while 'Tragic' was mapped to 'Good'. The 'Ambiguous' class was removed, resulting in a dataset of 1,274 records with two classes: 'Good' (786) and 'Bad' (488).
- **Text Preprocessing:** A function was implemented to clean the character Description text. This involved converting text to lowercase, removing URLs, punctuation, and numbers, and filtering out standard English stopwords using the NLTK library.
- **Feature Engineering:** The cleaned text data was converted into a numerical format suitable for machine learning.

- **TF-IDF and Count Vectorization:** Used to create numerical feature vectors with an n-gram range of (1, 2). This resulted in 2,391 features.

- **Model Training and Evaluation:** The vectorized data was split into training (80%) and testing (20%) sets. Six classification models were trained and evaluated:

    1. **Logistic Regression: This is a fundamental linear model used for binary classification. It works by calculating the probability of a sample belonging to a particular class. In this project, it predicts the likelihood of a character's verdict being "Good" based on the TF-IDF features derived from their description. The model was implemented with a regularization parameter C=1.0 and a maximum of 1000 iterations to ensure convergence.**

    2. **Multinomial Naive Bayes: A probabilistic classifier based on Bayes' theorem, Multinomial Naive Bayes is particularly effective for text classification tasks involving word counts. It operates on the "naive" assumption that features (in this case, words or n-grams) are independent of each other. For this project, it was trained using CountVectorizer features and a smoothing parameter alpha=1.0 to handle words that might not appear in the training data. This model ultimately provided the best performance on the test set.**

    3. **Support Vector Machine (Linear Kernel): A Support Vector Machine (SVM) is a powerful model that finds an optimal hyperplane that best separates the data points of different classes. By using a linear kernel, the model finds a straight-line decision boundary in the high-dimensional feature space to distinguish between "Good" and "Bad" characters. It was trained on the TF-IDF features with a regularization parameter C=1.0, and the probability=True setting was enabled to allow for AUC score calculation.**

    4. **Random Forest Classifier (with regularization): Random Forest is an ensemble learning method that constructs a collection of decision trees and outputs the class that is the mode of the classes' output by individual trees. This approach helps to control for overfitting. In this project, the model was regularized by limiting the number of estimators to 100, setting a max_depth of 10, and requiring a minimum of 10 samples to split a node and 5 samples per leaf node. It also used the class_weight='balanced' parameter to account for the imbalance between "Good" and "Bad" labels in the dataset.**

    5. **Gradient Boosting Classifier:** This is another powerful ensemble technique that builds models in a sequential, stage-wise fashion. Each new model (tree) is trained to correct the errors made by the previous ones, creating a strong final classifier. The implementation used 100 estimators

with a learning_rate of 0.1 and a max_depth of 5 for each tree. It was trained on the TF-IDF features to predict the character verdicts.

Model Comparison: The performance of five model was systematically compared using key metrics, including accuracy on the test set, mean accuracy from 5-fold cross-validation, and the Area Under the ROC Curve (AUC).

Results & Evaluation

The performance of the six models was evaluated using Accuracy, Precision, Recall, F1-Score, and AUC. The Artificial Neural Networks

model emerged as the top performer across all key metrics.

Model Performance Comparison:

Model	Accuracy*	AUC Score**	Precision (Bad)	Recall (Bad)	F1-Score (Bad)	Precision (Good)	Recall (Good)	F1-Score (Good)
Logistic Regression	0.7176	0.855	0.82	0.34	0.48	0.7	0.96	0.81
Naive Bayes	0.8118	0.8601	0.78	0.71	0.74	0.83	0.87	0.85
SVM	0.8	0.8321	0.78	0.66	0.72	0.81	0.89	0.84
Random Forest	0.7333	0.7982	0.61	0.85	0.71	0.87	0.66	0.75
Gradient Boosting	0.7373	0.7715	0.75	0.48	0.58	0.73	0.9	0.81

Best Performing Model: Multi Nomial Naïve Bayes

- Accuracy: 0.8118
- F1 Score: 0.74
- F1 score: 0.85

The **:** Multi Nomial Naïve Bayes model demonstrated a strong ability to correctly identify both "Good" and "Bad" characters, achieving a balanced performance in precision and recall for both classes.

Conclusion

This project successfully developed a machine learning pipeline to classify book characters as "Good" or "Bad" from their descriptions. After comparing five different

models, the Multi Nomial Naïve Bayes model was identified as the most effective, achieving a test accuracy of 0**.8118%**.

Implementation and UI: