**Part A: IMDb Movie Review Sentiment Analysis**

**Project Report**

**Video Explanation Link:-**

[**https://drive.google.com/file/d/1ilhYrqvg6ScBaCbgdJKi6\_L0JmDMdqYw/view?usp=sharing**](https://drive.google.com/file/d/1ilhYrqvg6ScBaCbgdJKi6_L0JmDMdqYw/view?usp=sharing)

**Overview**

* The goal of sentiment analysis, a task in Natural Language Processing (NLP), is to ascertain the text's emotional tone. It is possible to categorize this emotional tone as either positive or negative.To put it simply, sentiment analysis determines whether a text conveys a neutral, negative, or positive sentiment.
* Analyzing movie reviews from the IMDb dataset is the aim of this project. Our goal is to use the review text to predict if a given review will be positive or negative.
* Sentiment analysis aids in understanding audience reactions and guiding strategy in sectors such as marketing, e-commerce, and entertainment**.**

**Statement of the Problem :-**

* Building a classification model that can determine whether an IMDb movie review is positive or negative is the main goal.
* Thousands of user reviews with sentiment labels are included in the dataset.
* The project entails:
* Preprocessing and text cleaning
* TF-IDF feature extraction
* Developing a model with multiple classifiers
* Metrics like accuracy, F1-score, confusion matrix, etc. are used for evaluation**.**

**Source of Dataset Information: IMDb movie reviews dataset**

**Columns:**

* review: The user's actual assessment of the film
* sentiment: The intended classification (positive or negative**)**

**Preprocessing and Data Exploration**

* Exploratory Analysis: Verified the class balance, missing values, and dataset shape.

Distribution of review lengths plotted confirmed that the sentiment classes are roughly 50/50.

**Preprocessing Procedures: Reducing text**

* Eliminating HTML tags, numbers, and punctuation
* Eliminating stop words
* Lemmatization and Tokenization with NLTK/spaCy
* used the following to convert the cleaned text into numerical format:
* Vectorizer TF-IDF

**Feature Engineering**

* TF-IDF Features: Captured term importance relative to document frequency
* Textual Features:
  + Word count
  + Character count
  + Average word length

**Model Development**

Developed and compared the following models:

1. Logistic Regression
2. Multinomial Naive Bayes
3. Support Vector Machine (SVM)
4. Random Forest Classifier

All models were trained using a pipeline including:

* TF-IDF Vectorizer
* Stratified Train-Test Split (80-20)
* Cross-validation (5-fold)
* Hyperparameter tuning via GridSearchCV where applicable

**Model Evaluation**

**Metrics Used:**

* Accuracy
* Precision, Recall, F1-Score
* Confusion Matrix
* ROC-AUC (optional)

Confusion matrix and classification report were generated for all models. Logistic Regression and SVM performed the best, achieving 85–88% accuracy.

**Visualizations included:**

* Confusion matrices
* Bar plots of feature importances
* Word clouds for positive and negative terms