Name: Fatima Tariq

DHC-ID: DHC-1411

AI/ML Internship Report

This report outlines the approach, challenges, model performance, and improvements for the tasks completed during the AI/ML internship.

1. Fake News Detection Model

Approach Used:

Data Preprocessing: The datasets `Fake.csv` and `True.csv` were cleaned by removing stopwords, performing tokenization, and vectorizing the text using TF-IDF. Model: Used Logistic Regression for text classification after vectorization. The trained model was saved as `fake_news_model.pkl` for easy deployment.

Challenges Faced:

Class Imbalance: The dataset had an unequal number of fake and real news articles, requiring stratified sampling to balance classes. Text Quality: Articles included symbols and special characters that needed thorough cleaning.

Model Performance & Improvements:

The Logistic Regression model achieved an accuracy of 92%, with good performance in distinguishing fake news from true news. Improvement: For better accuracy, fine-tuning transformer-based models like BERT could yield better contextual understanding. Deployment: The model is deployed using Flask in the `app.py` file. A simple web app takes user input and returns whether it's fake or real news.

2. Customer Segmentation Using Clustering

Approach Used:

Data: The dataset `Mall_Customers.csv` contains features like age, annual income, and spending score. The features were scaled and clustered using KMeans clustering. Modeling: Chose the Elbow Method to find the optimal number of clusters (K) and implemented the KMeans algorithm.

Challenges Faced:

Determining Optimal K: The optimal number of clusters was determined using the Elbow Method, but the clustering results could vary depending on feature selection.

Interpretability: Understanding and interpreting clusters based on KMeans can sometimes be ambiguous, requiring further analysis for actionable insights.

Model Performance & Improvements:

The clustering model segmented customers into meaningful groups based on their spending behavior. Improvement: Implement DBSCAN for density-based clustering and visualize clusters using PCA or t-SNE to improve interpretability. Deployment: The results are visualized using matplotlib and deployed through Flask (app.py).

3. Movie Review Sentiment Analysis

Approach Used:

Preprocessing: The `IMDB Dataset.csv` was preprocessed by removing noise, stopwords, and applying tokenization. TF-IDF vectorization was used to convert text into numerical features. Model: Logistic Regression was used for sentiment classification (positive or negative).

Challenges Faced:

Sarcasm and Context: The dataset contained sarcastic reviews, which the logistic regression model struggled to classify accurately. Data Noise: Some reviews were ambiguously labeled, making the model's predictions less reliable.

Model Performance & Improvements:

Achieved an accuracy of 89%, but the model can improve with more sophisticated techniques such as LSTM or BERT, which can handle complex language patterns better. Improvement: Implementing deep learning models like BERT for context-based sentiment analysis. Deployment: The web app allows users to input reviews and receive a sentiment classification using Flask (app.py).

Deployment Instructions:

1. Install necessary dependencies (e.g., flask, scikit-learn, nltk).
2. Run `app.py` to launch the web app.
3. Access the app via `localhost:5000` on the browser.