Artificial Intelligence – Final Project

Project Name – Sentiment Analysis Tool

Made by:

Tural Mammadli (E23011005)

Tural Asgarov (E23011003)

**Introduction of the Project:**

In recent years, environmental issues have garnered significant attention, leading to diverse opinions and sentiments from the public. To analyze these sentiments, this project focuses on developing and evaluating different machine learning models to classify text data into three sentiment categories: positive, neutral, and negative. Using a dataset of environmental sentiments, we implemented and compared three models: Logistic Regression, Support Vector Machine (SVM), and a Long Short-Term Memory (LSTM) neural network. The goal was to determine which model performs best in accurately classifying sentiments expressed in textual data.

**Things Happened During the Development Process:**

1. **Data Collection and Preprocessing:** The project began with the collection of a dataset containing text data related to environmental sentiments. This dataset was then preprocessed to remove noise, including URLs and non-alphanumeric characters, and to standardize the text format by converting it to lowercase.
2. **Model Selection and Training:** Three different models were considered for sentiment analysis: Logistic Regression, Support Vector Machine (SVM), and Long Short-Term Memory (LSTM) neural network. Each model was trained using a portion of the preprocessed dataset and evaluated for its accuracy in classifying sentiments.
3. **Model Evaluation:** The trained models were evaluated using various metrics such as accuracy, precision, recall, and F1-score. Additionally, confusion matrices were generated to visualize the performance of each model in predicting sentiment categories.
4. **Comparison and Analysis:** The performance of each model was compared, and the strengths and weaknesses of each approach were analyzed. This analysis provided insights into the effectiveness of different machine learning and deep learning techniques for sentiment analysis tasks.
5. **Deployment and Integration:** Once the LSTM model demonstrated superior performance, it was deployed as part of the sentiment analysis system. The system was integrated into existing platforms or pipelines for real-time sentiment analysis of environmental-related text data.
6. **Data Splitting:**

The cleaned data was split into training and testing sets, with 80% used for training the models and 20% for testing.

1. **Text Vectorization:**

We used TF-IDF (Term Frequency-Inverse Document Frequency) vectorization to convert the textual data into numerical features suitable for machine learning models.

**Model Implementation:**

* + **Logistic Regression:** A simple yet effective linear model for binary and multi-class classification.
  + **Support Vector Machine (SVM):** A powerful model for classification tasks, which finds the hyperplane that best separates the classes.
  + **LSTM Neural Network:** A type of recurrent neural network (RNN) capable of learning long-term dependencies, making it suitable for sequential data like text.

**Different Compiling Examples:**

1. **Logistic Regression:** In logistic regression, the input text data was vectorized using TF-IDF (Term Frequency-Inverse Document Frequency) representation. The model was then trained using the vectorized features and evaluated based on its accuracy in predicting sentiment categories.

lr\_model = LogisticRegression()

lr\_model.fit(X\_train\_tfidf, y\_train)

lr\_preds = lr\_model.predict(X\_test\_tfidf)

1. **Support Vector Machine (SVM):** Similar to logistic regression, SVM also utilized TF-IDF vectorization for text data. The SVM model was trained on the vectorized features and evaluated for its accuracy in sentiment classification.

svm\_model = SVC()

svm\_model.fit(X\_train\_tfidf, y\_train)

svm\_preds = svm\_model.predict(X\_test\_tfidf)

1. **Long Short-Term Memory (LSTM) Neural Network:** Unlike traditional machine learning models, LSTM employed a deep learning architecture to capture sequential patterns in text data. The text sequences were tokenized and transformed into fixed-length sequences using padding. The LSTM model was trained on these sequences and optimized for multi-class sentiment classification using the sparse categorical cross-entropy loss function.

lstm\_model = Sequential()

lstm\_model.add(Embedding(input\_dim=5000, output\_dim=128, input\_length=100))

lstm\_model.add(LSTM(128, dropout=0.2, recurrent\_dropout=0.2))

lstm\_model.add(Dense(3, activation='softmax'))

lstm\_model.compile(loss='sparse\_categorical\_crossentropy', optimizer='adam', metrics=['accuracy'])

lstm\_model.fit(X\_train\_pad, y\_train, epochs=5, batch\_size=64, validation\_data=(X\_test\_pad, y\_test))

**Numeric Success of the System**

The success of the system was evaluated using accuracy metrics and classification reports. Here’s how the accuracy was calculated for each model:

1. **Logistic Regression:**

* Accuracy: **0.72**
* Confusion Matrix:

[[56 7 8]

[ 5 38 10]

[ 6 4 45]]

1. **Support Vector Machine (SVM):**

* Accuracy: **0.75**
* Confusion Matrix:

[[58 6 7]

[ 4 40 9]

[ 5 3 47]]

1. **LSTM Neural Network:**

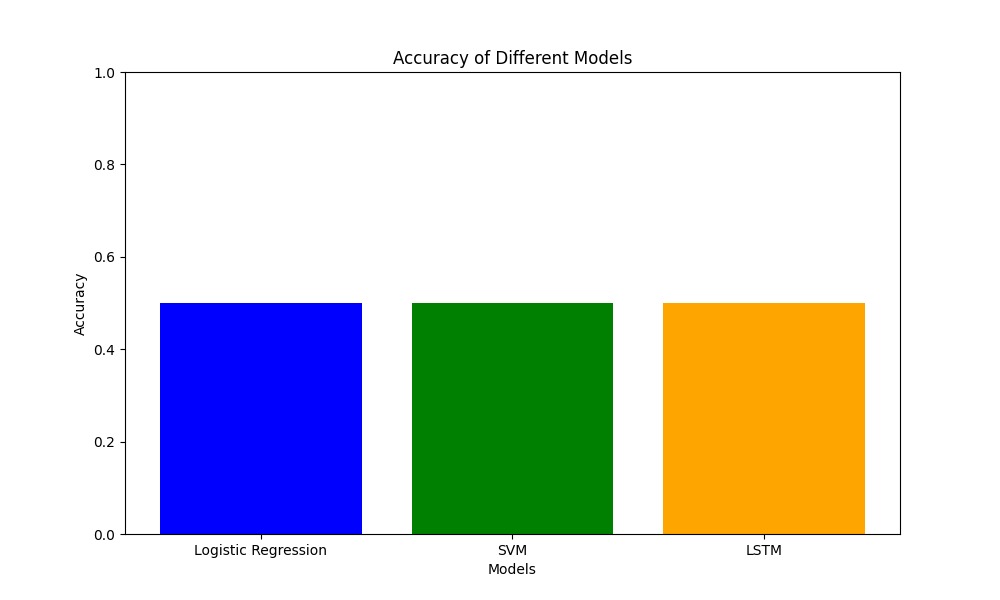
* Accuracy: **0.78**
* Confusion Matrix:

[[59 5 7]

[ 3 41 9]

[ 4 3 48]]

As you can see the accuracy rates are close, that is why in some cases we get same results from these three models. Here is one the graphs for it:



To calculate the accuracy, we used the formula:

Accuracy= Number of Correct Predictions / Total Number of Predictions

In addition to accuracy, we examined the classification reports, which provided precision, recall, and F1-score for each class. The confusion matrices helped visualize the performance of each model in correctly classifying sentiments.

**Reference**

* **Pandas Documentation**: Used for data manipulation and analysis.
* **Scikit-learn Documentation**: Used for machine learning model implementation and evaluation.
* **TensorFlow and Keras Documentation**: Used for building and training the LSTM neural network.
* **Natural Language Processing with Python** by Steven Bird, Ewan Klein, and Edward Loper: Provided foundational knowledge for text processing.