Assignment 9: Amazon Gourmet Foods Sentiment Analysis with Apache Spark

Introduction

This assignment aims to perform sentiment analysis on the Amazon Gourmet Foods reviews dataset. The objective is to use **Apache Spark** for distributed data processing and the **Hugging Face Transformers** library to classify each review as **POSITIVE** or **NEGATIVE**. The assignment is divided into two tasks:

1. **Task 1:** Perform sentiment classification using Hugging Face's pre-trained DistilBERT model.
2. **Task 2:** Evaluate the performance of the sentiment classifier by calculating Precision, Recall, and displaying the confusion matrix.

Task 1: Sentiment Classification

Objective:

The goal of Task 1 is to classify Amazon Gourmet Foods reviews as either **POSITIVE** or **NEGATIVE**.

System Adjustment:

In *task1.py,* adjust these things according to the system requirements

```
# Limit thread usage (important for HuggingFace on CPU)
os.environ["OMP_NUM_THREADS"] = "10" # Should be adjusted based on CPU cores
os.environ["PYSPARK_PYTHON"] = "/home/amar/Desktop/2nd_sem/MLOps/bin/python" # Path to your Python environment
```

```
# Step 2: Create Spark Session with limited parallelism
spark = SparkSession.builder \
    .appName("SentimentAnalysisGourmet") \
    .config("spark.sql.shuffle.partitions", "10") \
    .config("spark.default.parallelism", "10") \
    .getOrCreate()
```

Approach:

1. **Data Parsing:**
    - The reviews are stored in Gourmet_Foods.txt. The file is structured as key-value pairs where the key is the review score and the value is the review text.
    - The reviews are parsed and the text is passed to a **DistilBERT** model using Hugging Face's sentiment analysis pipeline.
2. **Sentiment Analysis:**
    - A **sentiment analysis pipeline** is used for sentiment classification using Hugging Face's pre-trained DistilBERT model.
    - Reviews are classified as either **POSITIVE** or **NEGATIVE** based on the model's prediction.
3. **Spark UDF:**
    - The sentiment classification logic is applied to the dataset using a **Spark UDF** (User-Defined Function). The UDF processes each review and appends the predicted sentiment as a new column to the DataFrame.
4. **Output:**
    - The results are saved as CSV files (partitioned by Spark) inside the sentiment_output/ directory.

```
# Step 3: Sentiment classifier

sentiment_model = pipeline("sentiment-analysis")
```

- This loads the default model, **DistilBERT fine-tuned on SST-2** from HuggingFace's `transformers` library.
- This model is then used in a Spark UDF to classify each review's sentiment and save the output as `sentiment_output`.

Task 2: Sentiment Classifier Evaluation

Objective:

The objective of Task 2 is to evaluate the performance of the sentiment classifier model. Reviews with a rating of **3.0 or greater** will be considered as **POSITIVE**, and reviews with a rating of **less than 3.0** will be considered as **NEGATIVE**. This evaluation will include:

1. **Precision**: The proportion of positive predictions that were correct.
2. **Recall**: The proportion of actual positives that were correctly identified.
3. **Confusion Matrix**: A matrix to visualise the performance of the classifier.

Approach:

1. **Ground Truth Labels:**
   The ratings in the dataset are used to define the ground-truth labels:

   - Reviews with a rating ≥ 3.0 are labelled as **POSITIVE**.
   - Reviews with a rating < 3.0 are labelled as **NEGATIVE**.

2. **Confusion Matrix:**
   A confusion matrix is calculated using the following components:

   - **True Positives (TP)**: Predicted **POSITIVE**, and actual **POSITIVE**.
   - **False Positives (FP)**: Predicted **POSITIVE**, and actual **NEGATIVE**.
   - **True Negatives (TN)**: Predicted **NEGATIVE**, and actual **NEGATIVE**.
   - **False Negatives (FN)**: Predicted **NEGATIVE**, and actual **POSITIVE**.
3. **Precision and Recall Calculation:**
   - **Precision** is calculated as:
     $$Precision = \frac{TP}{TP+FP}$$
   - **Recall** is calculated as:
     $$Recall = \frac{TP}{TP+FN}$$
4. **Visualization:**
   - A **heatmap** is generated to display the confusion matrix.
   - A **bar chart** compares the **Precision** and **Recall** scores.

Key Results and Visualisations

1. **Confusion Matrix**: The confusion matrix visualises the true vs. predicted sentiments, helping to understand how well the model distinguishes between **POSITIVE** and **NEGATIVE** reviews.
2. **Precision and Recall**: The bar chart compares **Precision** and **Recall** scores, which are critical metrics to assess the performance of the sentiment classifier.

Conclusion

This assignment demonstrates the use of **Apache Spark** for large-scale data processing and **Hugging Face Transformers** for sentiment analysis. The evaluation metrics, such as **Precision**, **Recall**, and the **Confusion Matrix,** provide insights into the effectiveness of the sentiment classification model. The project can be scaled further by incorporating more complex models or experimenting with larger datasets.