Guillermo Ortiz Macías
216787558
guillermo.ortiz7875@alumnos.udg.mx
Maestría en Ciencia de los Datos
Programación 2
Challenge 2: Analysis of comments on Glassdoor

This challenge consists of using a natural language processing model to categorize job reviews from a dataset into good reviews or bad reviews.

The dataset

The dataset used for this challenge can be found in kaggle in the following link: https://www.kaggle.com/datasets/davidgauthier/glassdoor-job-reviews. This dataset has more than 600 thousand different job reviews posted on glassdoor. The more important columns in the dataset for the purpose of this challenge are:

- Recommend: This column will be our objective variable, it tells us whether a person recommends a job or not. Its possible values are: “v” for positive, “r” for mild, “x” for negative and “o” for no opinion.
- Headline: This column is the first part of the text review that a user wrote down.
- Pros: This column contains the positive comments of a user about a job.
- Cons: This column contains the negative comments of a user about a job.

Model construction

To solve this challenge I used a pre-trained NLP model called **BERT.** Specifically the **bert-base-multilingual-uncased-sentiment** model from *HuggingFace.* I used this model because it is specialized for sentiment analysis and can be used in 6 different languages, including English and Spanish. More about this model here:

https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment

In the python script I use this model using the *transformers* and *pytorch* libraries.

The model can be used using two steps:

1. **Tokenize** the input text. First the input text (in this case the job review) needs to be tokenized in different blocks of text that the model understands. I did the tokenization using the **AutoTokenizer** class that comes with the *transformers* library.
2. **Run the model** using the tokenized input string. The model receives a tensor of tokens and it returns a tensor of *logits* of length 5. This returning tensor tells me the probabilities for the text to be in one of five categories for a review.

   For example the text “Este ha sido el peor trabajo en el que he estado” returns the tensor: [3.9755, 1.8709, -0.1890, -2.6711, -2.2595]. This tells me that:
   - 3.97 chances that the text is a category 1 review (very bad review)
   - 1.87 chances that the text is a category 2 review (bad review)
   - -0.18 chances that the text is a category 3 review (neutral)
   - -2.67 chances that the text is a category 4 review (good review)

- -2.26 chances that the text is a category 5 review (very good review)

For this challenge I made the decision to take the category with the biggest chance. In this case the text “Este ha sido el peor trabajo en el que he estado” is categorized as a **very bad review.**

In order to get only “good” or “bad” reviews I made the choice that any results in categories 1,2 or 3 are categorized as “bad” and results in categories 4 and 5 are categorized as “good”.

Since the dataset contains more than 600 thousand reviews, and it would take hours to classify all the reviews, I decided to make a random sample of 200 reviews from the dataset and test the **BERT** model with it. These are the results:

Confusion matrix:

	Actually Positive	Actually Negative
Predicted Positive	46	14
Predicted Negative	32	108

Precision: 0.88

Accuracy: 0.77

As you can see, the model does a decent job of categorizing job reviews into good or bad reviews. Though it can be better if, maybe, I balance the dataset to have a similar amount of positive and negative reviews.

ML Ops

I integrated a **mlflow** run in the script. This script is going to log the metrics for the different confusion matrix results, as well as the result for the precision and accuracy. These logs can be seen in the *mlruns* folder that is created after running the script.

How to run the script?

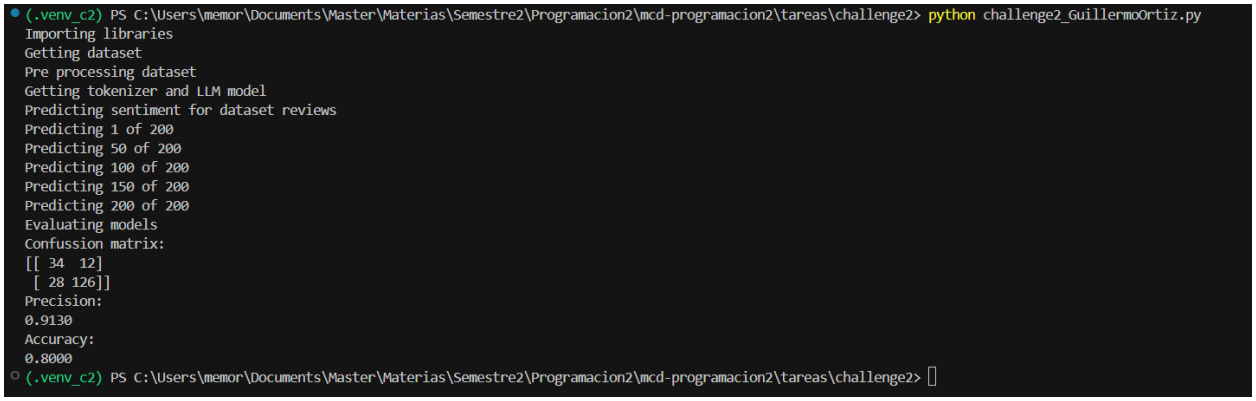
1) Download the git repository from my github account:
https://github.com/guillermortiz21/mcd-programacion2 if for some reason you don’t have permissions to see the repository or to download it, send me an email to *guillermo.ortiz7875@alumnos.udg.mx*

2) With the repository downloaded you can navigate to the folder *tareas/challenge2* here you will find a python script called *challenge2_GuillermoOrtiz.py* in this script contains all the logic about how to download, instantiate and run the BERT NLP model, and the implementation to use it with the glassdoor reviews dataset.

3) Download the dataset *glassdoor_reviews.csv* from glassdoor: https://www.kaggle.com/datasets/davidgauthier/glassdoor-job-reviews and add the csv in the same folder as the *challenge2_GuillermoOrtiz.py* file.

4) Download dependencies for the script to run it: Ideally in a new python environment only to run this script, install the dependencies that are listed in the *requirements.txt* file. You can do this by running the command *pip install -r requirements.txt*

5) With all dependencies downloaded you can simply run the script with the command *python challenge2_GuillermoOrtiz.py* you will see the process of the script in the logs:



Research references:

To solve this challenge I have to make a lot of investigation about the theory behind Natural Language Models and how to implement them. This challenge was possible thanks to the following 3 references:

- "*Grandes Modelos de Lenguaje*" book by the author *John Atkinson-Abutridy*
- *"Using a BERT Model for Sentiment Analysis"* blog post by *Mirza Yusuf.* You can find it here: https://medium.com/@Mirza_Yusuf/using-a-bert-model-for-sentiment-analysis-6c6fcc106843
- *"Sentiment Analysis with BERT Neural Network and Python"* youtube video from *Nicholas Renotte.* You can find it here: https://www.youtube.com/watch?v=szczpgOEdXs&ab_channel=NicholasRenotte