Tab 1

**Business Use Case:** Sentiment Analysis of Google Comments for Disneyland Paris Using BERT

**Students’ names:** Marie-ange Ayangma and Emmanuelle Desablin

**Tables of Contents**

[**1. Business Use Case (BUC) 1**](#_tsvzz1ivns3q)

[Constraints and Challenges: 1](#_w7hgqnqzoogr)

[**2. Dataset: 2**](#_oz6dxzburmlx)

[**3. Machine Learning Model and Workflow 2**](#_3c42aeurujwi)

[A) Exploratory Data Analysis (EDA) 2](#_i253vd8jqwt0)

[C) Saving the Best Parameters 3](#_38qhrsw251tu)

[D) Testing the Model 4](#_l1pl1e96w9ni)

[**4. Baseline Model 4**](#_9ej3wwfnryw9)

[**5. Loop of Iteration 4**](#_e6vy8u4iohks)

## 1. Business Use Case (BUC)

**Context**:  
The goal of this project is to enhance customer sentiment analysis for Disneyland Paris by fine-tuning a pre-trained BERT (Bidirectional Encoder Representations from Transformers) model on Google comments specific to Disneyland Paris. The model is referred to as "pre-trained" because it has been trained on general-purpose data. As a result, it may not fully capture the nuances of user-generated comments in this specific context, such as varying languages, tones, and sarcasm.

**Stakeholders**:

* Disneyland Paris Client Experience team including Business Analysts and Data Scientists
* Various Operations teams (show, hotels, F&B etc.)

**Benefits and takeaways of the Machine Learning modelling:**

* Exploration of new relevant data sources on clients satisfaction.
* Provide an automatized tool to analyse the new data and since provide time saving in data analysis

**Constraints and Challenges:**

* The dataset consists of Google comments from a limited time period (April 1st to April 15th, 2024).
* The diversity of tones, and potential use of sarcasm introduces complexity to the analysis.
* Automatic translation of the comments via Google Translate may result in the loss of nuances such as tone or sarcasm, further complicating the interpretation of sentiment.

## 2. Dataset:

* The dataset consists of Google comments related to Disneyland Paris, collected between April 1st and April 15th, 2024.
* These comments reflect real customer feedback about their experiences at the park.
* The dataset includes both comments originally in English and translations of comments from various other languages, performed automatically using Google Translate. However, this translation process may obscure important nuances, such as tone or sarcasm, which could affect sentiment interpretation.

To ensure high-quality input for the model, the following preprocessing steps were applied:

* The dataset was filtered to include only English-language comments.
* Each comment was split into sentences for easier processing.
* Manual annotation was performed to assign one of four sentiment labels to each comment: 'positif,' 'négatif,' 'neutre,' or 'mitigé.'

## 3. Machine Learning Model and Workflow

### A) Exploratory Data Analysis (EDA)

The eda.ipynb file contains the following key analyses:

* **Data Overview**: Provided an initial look at the data, including distribution of rating and sentiment labels ('positif,' 'négatif,' 'neutre,' or 'mitigé.') in the dataset.
* **Data Cleaning and preprocessing**: Displayed steps for cleaning the data and structuring the data for training.
  + Remove of empty rows (no comments) and duplicates
  + Transform the data type for specific columns
  + Reorder of column (from the most recent to the oldest)
* **Sentiment Distribution**:
  + Visualisation of the distribution of sentiments ('positif,' 'négatif,' 'neutre,' or 'mitigé.') in the dataset
  + Visualisation of the length of comments
* Lenght of tokens: we decide to keep comments with a length of 100 after tokenization because we make the assumption that the beginning of the comment is sufficient to grasp the sentiment of the whole comment, being common to introduce one’s feeling at the outset.

### B) Training the Model

The script.ipynb file contains the following key analyses:

* **Model**: BERT, a pre-trained Natural Language Processing (NLP) model, is fine-tuned using the dataset.
* **Metrics**: During training, the two key indicator monitored:
  + **Train Loss**: Evaluates how far the model's predictions deviate from the true sentiment labels, with the goal of minimizing this loss.

Calculation methodology:

Each time the model makes a prediction, it compares its prediction with the true label. The difference between the prediction and the correct label is called the ‘loss’.

If the model’s prediction is far from the correct label, the loss will be large. If the prediction is close, the loss will be small. During training, the model continually adjusts its internal parameters to minimize this loss. The goal is to reduce the train loss as much as possible.

>>Example:

If the model predicts a comment is ‘negative’ when it’s actually ‘positive,’ the loss will be high. If it predicts ‘positive’ but with some uncertainty (e.g., a score slightly below 100%), the loss will be smaller.

* **Class-wise error rate**: a performance metric that shows how well the model performs for each individual class. It helps identify which classes the model is struggling with. Lower class-wise error rates indicate that the model is performing well for that particular sentiment class.Higher error rates could point to issues with class imbalance, model bias, or complexity in that specific sentiment category (e.g., "neutre" may be harder to classify).

Calculation Methodology: Error Rate (Class i)=Number of incorrect predictions for Class i / Total number of samples in Class i Where:

* + "Class i" refers to one of the sentiment labels, e.g., 'positif,' 'négatif,' 'neutre,' or 'mitigé.'
  + The numerator is the count of incorrect predictions for that class.
  + The denominator is the total number of samples that truly belong to that class.

### C) Saving the Best Parameters

* The model's internal weights are continuously adjusted during training to minimize the train loss and class-wise error rate.
* Application of cross-validation to find the best hyper parameters which are saved for future use. We look at the following hyper parameters:
  + Max Lenght
  + Batch size
  + Learning rate
  + Number of epochs

We find the optimal number of epochs by monitoring the training and validation loss curves over epochs. These curves plot loss values against the number of epochs, providing insights into the model's learning behavior.

### D) Testing the Model

* **Test Data**: After training, the model is tested with the best hyper parameters on a separate dataset to evaluate its ability to generalize and accurately predict sentiments on unseen data.
  + We could not reach the end of the model testing given recurrent kernel crash. We picked up hyper parameters that looks relevant. We test the model with them. We can therefore compare metrics (the ones of the baseline vs. the ones of the test)
* **Overview of the results**

## 4. Baseline Model

**Baseline Setup**:

* **Preprocessing:** Tokenization and sentence splitting using BERT’s tokenizer
* **Model**: The pre-trained BERT model before fine-tuning.

**Metrics:**

* **Final Loss: 1.1240148629461015**
* **Class-wise error rate: [positif : 1.0, négatif:1.0, neutre: 0.0, mitigé: 1.0]**

## 5. Loop of Iteration

Our loop is a series of tests of hyper parameters combination. Thus the first iteration is the result of the first loop. For each iteration we get metrics.

We use the cross-validation to test different hype parameters according the recommended range for the BERT model:

* + **Max Lenght**
  + **Batch Size Suggestion:** 10, 20, 30, or 50
  + **Learning Rate Suggestion:** chosen from a continuous range between 1e-5 and 5e-5, on a logarithmic scale
  + **Number of Epochs selected according the loss curve over the number of epoch:** 1, 3, or 5

We use Optuna to display the metrics and see which combination of hyper parameters is the best.

## 6. Score Analysis

| **Max Length** | **Batch Size** | **Learning Rate** | **Epoch** | **Loss** | **Loss Difference (vs Baseline)** | **Error Rates**  **[Pos, Neg, Neutre, Mitigé]** | **Error Rate Difference**  **(vs Baseline)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **350** | **15** | **1e-5** | **1** | **/** | **-** | **/** | **-** |
| **100** | **15** | **1e-5** | **1** | **1.13** | **Baseline** | **[1.0, 1.0, 0.0, 1.0]** | **Baseline** |
| **100** | **15** | **1e-5** | **3** | **0.75** | **-0.38** | **[0.33, 1.0, 0.04, 0.55]** | **[-0.67, 0, 0.04, -0.45]** |
| **100** | **15** | **1e-5** | **4** | **0.70** | **-0.43** | **[0.33, 1.0, 0.04, 0.67]** | **[-0.67, 0, 0.04, -0.33]** |
| **100** | **15** | **1e-5** | **5** | **0.81** | **-0.32** | **[0.44, 1.0, 0.0, 0.76]** | **[-0.56, 0, 0, -0.24]** |
| **100** | **15** | **2e-5** | **3** | **0.73** | **-** | **[0.61, 1.0, 0.07, 0.52]** | **-** |