# Sentiment Analysis (using Transformers and PyTorch)

This notebook demonstrates how to use `transformers` to perform sentiment analysis. These transformers use the PyTorch library to perform the actual computation.

## Imports

Here we import the required packages.

In [ ]:
```
import os
import warnings
import pandas
import torch
from transformers import pipeline
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
```

## Functions

Here we define the functions that we will use in this notebook. We will use these functions to perform sentiment analysis. Which are:

- **get_pipeline**: This function creates a pipeline that will be used to perform sentiment analysis.
- **get_dir**: This function returns the directory where the data is stored.
- **get_reviews**: This function returns the reviews.
- **get_sentiments**: This function returns the sentiments of the reviews.
- **export**: This function exports the sentiment analysis.
- **analyse_directory**: This function analyses the sentiment analysis of the reviews in a directory.

### Function: get_pipeline

This function creates a pipeline that will be used to perform sentiment analysis.

In [ ]:
```
def get_pipeline():
    model = AutoModelForSequenceClassification.from_pretrained(
        "gchhablani/bert-base-cased-finetuned-sst2"
    )
    tokenizer = AutoTokenizer.from_pretrained(
        "gchhablani/bert-base-cased-finetuned-sst2", do_lower_case=False
    )
    senti_pipeline = pipeline(
        "sentiment-analysis", model=model, tokenizer=tokenizer, truncation=True
    )
    return senti_pipeline
```

### Function: get_dir

This function returns the directory where the data is stored.

In [ ]:
```
def get_dir(path):
    files = []
    for file in os.listdir(path):
        if file.endswith(".csv") and not file.startswith("list"):
            files.append(file)
    return files
```

### Function: get_reviews

This function returns the reviews.

In [ ]:
```
def get_reviews(df):
    reviews = []
    for i in range(0, len(df)):
        reviews.append(str(df["Avaliações"][i]))
    return reviews
```

### Function: get_sentiments

This function returns the sentiments of the reviews.

In [ ]:
```python
def get_sentiments(reviews, senti_pipeline):
    sentiments = []
    for review in reviews:
        sentiments.append(str(senti_pipeline(review)[0].get("label")))
    return sentiments
```

## Function: export

This function exports the sentiment analysis.

In [ ]:
```python
def export(sentiments, reviews, path, name):
    df = pandas.DataFrame({"sentiment": sentiments, "review": reviews})
    path = path.replace("/../scrapes/", "/")
    df.to_csv(str(path) + str(name), index=False)
```

## Function: analyse_directory

This function analyses the sentiment analysis of the reviews in a directory.

In [ ]:
```python
def analyse_directory(path, senti_pipeline):
    files = get_dir(path)
    for file in files:
        df = pandas.read_csv(path + file, encoding="utf-8")
        reviews = get_reviews(df)
        sentiments = get_sentiments(reviews, senti_pipeline)
        export(sentiments, reviews, path, file)
```

## Execution

Here we execute the notebook. First we create the pipeline.

In [ ]:
```python
senti_pipeline = get_pipeline()
```

Then we get the directory where the data is stored. Then we get the reviews and the sentiments. Finally we export the sentiment analysis.

In [ ]:
```python
current_dir = os.getcwd()
path = current_dir + "/../scrapes/"
analyse_directory(path + "booking/hotels/", senti_pipeline)
analyse_directory(path + "zomato/restaurants/", senti_pipeline)
analyse_directory(path + "tripadvisor/hotels/", senti_pipeline)
analyse_directory(path + "tripadvisor/activities/", senti_pipeline)
analyse_directory(path + "tripadvisor/restaurants/", senti_pipeline)
```