**Distributed Data Processing and Sentiment Analysis of a Twitter Dataset**

Author: Jefferson William Teixeira

e-mail: [sba22201@student.cct.ie](mailto:sba22201@student.cct.ie)

Student ID: sba22201

Contents

[1 Introduction 3](#_Toc135767564)

[2 Materials and Methods 3](#_Toc135767565)

[3 Data Preparation and Visualization 3](#_Toc135767566)

[4 Distributed Data Processing and NoSQL Storage 3](#_Toc135767567)

[5 Sentiment Analysis and Time Series Forecast 3](#_Toc135767568)

[5.1 Experiment Setup 3](#_Toc135767569)

[5.2 Results 4](#_Toc135767570)

[6 Conclusions 4](#_Toc135767571)

[7 References 4](#_Toc135767572)

# Introduction

In this project, distributed data processing and machine learning (ML) were applied to a large Twitter dataset that contains comments about the war in Ukraine. The dataset was collected from a Kaggle [repository](https://www.kaggle.com/code/ssaisuryateja/eda-and-sentiment-analysis/input) and contains about 364,875 tweets on this topic. For this project, the dataset was stored using distributed environment and NoSQL databases and processed for sentiment analysis. The tweets were analysed using time-series forecasting to predict the overall Twitter user’s sentiment in different time intervals as per the experiments presented in this report. Moreover, a benchmarking tool called Yahoo Cloud Serving Benchmark (YCSB) was used to evaluate and compare the performance of MySQL and MongoDB databases.

This report is organized as follows: Section 2 describes the tools used to complete this project. In Section 3, the dataset, the data preparation steps, and the interactive dashboard for the sentiment analysis and time-series forecasting are described. In Section 4, the distributed environment and benchmarking results are discussed, while Section 5 covers the sentiment analysis and time series forecasting experiments. Finally, in Section 6, the conclusions are summarized.

# Materials and Methods

The sentiment analysis was implemented in Python in a Jupyter notebook running in Anaconda environment. The source code and files used in this project were hosted on GitHub under a public organization called [CCT-MastersDA](https://github.com/CCT-MastersDA). The project’s repository is called [cct-sem2-ca2](https://github.com/CCT-MastersDA/cct-ca2), which can be accessed with the following command in any terminal: *git clone* [*https://github.com/CCT-MastersDA/cct-sem2-ca2.git*](https://github.com/CCT-MastersDA/cct-sem2-ca2.git).

The source code is organized under the *jupyter* folder in the root directory. It contains one Jupyter file called ca2-jupyter.ipynb and the following folders: datasets, which contains the csv files used in this project, the images folder, where all the generated images are stored, and the modules folder, which basically contains auxiliary classes for the text processing and data manipulation. The reason to separate these modules from the Jupyter notebooks was to keep the code organised and to follow the best programming practices with regards to reuse and code modularization.

The distributed environment used to store the datasets was Hadoop and the technology implemented to access and manipulate the data in a programmatic way was PySpark, a Python API that enables large-scale data processing in Python. The dataset was also loaded in MongoDB, a NoSQL database. Regarding the benchmarking approach, MongoDB, and MySQL databases were compared using YCSB, an open-source specification and software package for benchmarking NoSQL database management solutions' relative performance.

Regarding the setup, for the distributed environment and data storage experiments, a Linux VM was prepared with the following main required software: MongoDB, MySQL, YCSB, Hadoop, PySpark and Anaconda. The sentiment analysis and time-series forecasting were implemented in Windows environment.

# Data Preparation and Visualization

Finally, a dashboard called **DataPrepVisDashboard** was also created using the Voila tool containing the static and interactive graphs created in the EDA steps. The previously extracted HTML version of the dashboard can be found in the *jupyter/dashboard* folder.

Four versions of the tweets were generated from the raw text: (1) cleaned with stop words, (2) cleaned without stop words, (3) *lemmatized,* and (4) *stemmerized* tweets. Stop words do not add much information to the text, so their frequency could bias the models, therefore, they are usually removed from the dataset. *Lemmatization* and *Porter Stemmer* are common ways to extract the core meaning from the words, this way these techniques were applied to the datasets to evaluate their impact in the performance of the classifiers.

# Distributed Data Processing and NoSQL Storage

Started Hadoop

# Sentiment Analysis and Time Series Forecast

The analysis described in this section was implemented and discussed in more detail in the accompanying Jupyter notebook called **ML**.

In section 5.1, it is discussed the sentiment analysis performed on the tweet’s dataset about agriculture. In section 5.2, the ML models used to predict the import and export of agriculture products in Ireland and other countries are presented.

Using the dataset 4, the sentiment analysis described in this section focuses on the classification of user’s comments about agriculture, extracted from the Twitter platform, into negative, neutral, or positive categories. This analysis was implemented in the Section 3 of the **ML** Jupyter notebook.

## Experiment Setup

## Results

# Conclusions

In this project, it was discussed several ML and statistical methods to acquire insights from the datasets. The main challenges faced in this project was the number of datasets needed to attend the brief. Each dataset had different characteristics and needed specific EDA discipline. Multiple datasets also complicated the organization of the Jupyter notebooks that had to be split to avoid rework and code duplication, while keeping good programming practices.

It was also difficult think of all different questions in order to apply the required techniques. In the statistical part, seeing the aggregated data graphs was very helpful to guide the discussions on the results of the inferential tests applied. For the ML, several datasets had to be extracted from the import/export dataset, which made the analysis complex and hard to explain. This problem could be simplified by focusing on a single aspect of the dataset (e.g., Import quantity in Ireland) not trying to model the whole dataset.

It was also very laborious to keep several Jupyter files and the report synchronized so the references to the code from this report are correct. Knowing which content to keep in each document was another challenge.

# References

Chapman, P. *et al.* (2000) *CRISP-DM 1.0 Step-by-step data mining guide*. Available at: http://www.crisp-dm.org/CRISPWP-0800.pdf.

Eurostat (2022) *Facts and figures on life in the European Union*, *Eurostat*. Available at: https://european-union.europa.eu/principles-countries-history/key-facts-and-figures/life-eu\_en (Accessed: 24 December 2022).

Scikit-learn (2022) *Scikit-learn: Machine Learning in Python*, *https://scikit-learn.org*.