TEXT MINING AND BIG DATA ANALYTICS

ARTIFICIAL INTELLIGENCE

Big Data Analytics: Final Project

Students :
Álvaro ESTEBAN MUÑOZ

Teacher :
Stefano Lodi

January 8, 2024

Contents

1 Introduction

This project aims to solve a machine learning task, namely a classification one, for a massive dataset. The task was solved using the power of Hadoop's distributed file system (also called HDFS) and Spark's engine, which relies in its own version of Hadoop's MapReduce algorithm for distributed computation.

In this report we will briefly describe the specifications of the used machine, as well as the dataset and the different ML algorithms applied to solve the task.

2 System Description

We set up a Hadoop system on a Debian 10 machine. We are working with just one machine and therefore, the HDFS was built in a single cluster. Even though this setup configuration does not allow for distributed file storing, we still decided to set it up for educational purposes. Moreover, this project could be replicated on a real distributed environment by just adding more clusters (hence more machines), which makes it scalable.

For the computation we installed Apache PySpark on our debian machine and configured it to work with jupyter notebooks, making it easier to experiment with the different ML algorithms.

3 Dataset

The chosen dataset for the project was WESAD (Wearable Stress and Affect Detection). [1]

> *WESAD contains data of 15 subjects during a stress-affect lab study, while wearing physiological and motion sensors.* [1]

The data is distributed in 15 folders, each one from a different subject. Inside each folder there is a lot of useful information, but we will focus on the **synchronised** files, which are **pickle files** containing all the data from the subject synchronised with the labels.

The features on the dataset are signals of two types:

- Taken with a chest device. All the data is sampled at 700Hz.
- Taken with a wrist device. Depending on the signal, the sampling frequency may vary (but never at more than 700Hz).

It is important to notice that the labels given are also sampled at 700Hz meaning that in case of using the signals from the wrist device we would need to **oversample** the data. However, for our task we decided to keep it simple and remove the signals from the wrist device, taking only the 8 features coming from the chest, which are already perfectly aligned with the labels.

Samples are labeled in two different ways; with the protocol label, which goes in the range from 0 to 7 and with self-report questionnaires complimented by the subject. For our experiment, we chose to use the labels assigned by the experts, not only because they are synchronised with the data sample frequency, but also because they are easier to understand for the model, **one sample has one label**.

Labels 5, 6 and 7 are said to not be used for the dataset and therefore had to be removed. In addition, we wanted to constraint our model to differ just between three states, *baseline*, *stress* and *amusement*, so that we removed 0 and 4 which represents *not defined* and *meditation* states. Even though, we ended up with more than **23 million samples**.

4 Approach

The task was solved in two notebooks contained on the repository. The first notebook *data_to_parquet.ipynb* takes the pickle data and converts it to a more PySpark adapted format, parquet. The notebook also explores the different features inside the datafiles. After executing this notebook, the data is ready to be moved to the HDFS.

The second notebook *WESAD.ipynb* does to main important things:

Data Pipeline It builds the data pipeline and applies it. This pipeline consists simply on the following:

- Remove desired labels (0, 4, 5, 6 and 7)
- Changes label range from [1, 2, 3] to [0, 1, 2].
- Assemble features in one unique vector for each sample
- Applies Min-Max scaling for min=0 and max=1.

Train and evaluate ML models The chosen algorithms are the following:

- Decision Tree (DT)
- Random Forest (RF)
- Naive Bayes (NB) - smoothing=0.5
- Logistic Regression with One VS Rest approach (LR) - maxIter=10, tol=$1e^{-6}$
- Multilayer Perceptron (MLP) - layers=$[8, 16, 32, 3]$, maxIter=20

PS: Except the specified hyperparamenters, the rest are PySpark default ones.

5 Results

All the obtained results are shown on table [1].

Classifier	Accuracy
Decision Tree	0.762597
Random Forest	0.801740
Naive Bayes	0.531165
Logistic Regression	0.553291
Multilayer Perceptron	0.553286

Table 1: Accuracy of each classifier on the test set

6 Discussion

Given the previous results the first thing we notice is the difference in accuracy between **parametric approaches** (NB, LR, MLP) and **non-parametric** ones (DT and RF). This could be due to the nature of both approaches.

Parametric models are fast to be learnt and they require less amount of data than non-parametric ones. However, they perform strong **assumptions** on the data distribution. On the other hand, non-parametric methods require much more data, nevertheless they tent to perform better and are much more **flexible** since they do not assume anything about the population.

To sum up, when dealing with Big Data Analytics, the amount of data is obviously not a problem, and performing feature engineering could cost so many efforts. Consequently, non-parametric approaches seem to be a safe starting point.

References

[1] P. Schmidt, A. Reiss, R. Duerichen, C. Marberger, and K. V. Laerhoven. Introducing wesad, a multimodal dataset for wearable stress and affect detection. ICMI 2018. 2018.