FDS Project 2022/2023
Sentimental Analysis

Federico Barreca - 1736423, Laura Mignella - 1920520,
Antonio Ratti - 1700089, Maria Vittoria Vestini - 1795724,
Cosmin Zaharia - 1805207

27/12/2022

Abstract

This report presents the analysis of two classification problems about human emotions in texts.
In the first part there is an introduction to the problem, then it continues with the descriptions of the methods used and the processing of the dataset.
In the end, instead, the focus is going to be on the results that we managed to obtain.

Author roles: The work here described was divided, as:

- Jupyter notebook: Everyone worked on it, but mainly done by Cosmin with the help of Laura and Maria Vittoria.
- Demo: Federico with some help of Cosmin for the front-end.
- Report: Laura and Maria Vittoria.
- Presentation: Antonio.

1 Introduction

Sentiment analysis is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer's attitude towards a particular topic, product, etc. is positive or negative.
The importance of sentiment analysis can be seen especially in commercial settings, where the manufacturers of a product could be interested in analyzing the consumer's response to their product, so they can understand what has to be improved.

2 Methods

Here are the methods used to classify the texts:

Pre-implemented methods

- **Logistic Regression**, algorithm used to categorize the inputs, that uses the *sigmoid function* to map predictions to probabilities that are going to be used to assign future labels.
- **K-Nearest Neighbors**, given a fixed number k, assign the label to a new sample by checking the ones of the k nearest elements.
- **Naive Bayes**, classification method that uses probabilities, computed by applying the Bayes'Theorem, to assign the labels to the elements.

From Scratch

- **Softmax Regression**, a generalization of logistic regression that replace the *sigmoid function* by the so-called *softmax function*.

3 Dataset and pre-processing

Dataset. For our project we chose a dataset from *KAGGLE* [1]. The dataset is composed of, more or less, 200k *Reddit* posts and their linked emotions, out of a list of 27 different ones.

Cleaning and Choosing the data. Before working with our data, we had to perform some operations to make it viable. In particular, we focused on cleaning the texts, to do so we used different methods coming from the *nltk* and *emoji* modules.
During the data analysis, the first thing we realized was that some entries in the dataset weren't linked to any emotion, while others had more than one. To deal with this problem, we got rid of all the texts that were not assigned to one and only one emotion.

Splitting the data. The last thing, before actually applying any of the methods named above, was for us to decide how to split our data. To do that we tried some different numbers of percentages, but, in the end, we settled on splitting the data, with the help of the *sklearn* library, as 80% train and the remaining 20% test.

4 Original Idea

Emotion Recognition is the process of identifying human emotion, with help coming from technology.

The initial intention for the project was to build an Emotion Recognition classifier, to do so we used the *wheel of emotions*[2] to map all the 27 emotions represented in the dataset to 6 core ones: *joy, love, anger, surprise, fear* and *sadness*.
Sadly we had some issues with the data that stopped us from obtaining the results we hoped for. The main problems were:

1. On the original dataset there was a problem of balance between the emotions. For example, the emotion *grief* had only ≈ 350 entries, while there were some a lot more present, like *approval* with over $15k$ samples.
2. *Common words*, we later noticed that there were quite a few words shared in different classes.

Together those two things caused the classifier not to work as well as we hoped for.
Nevertheless, the following section will be focused on the results that we obtained for both the Binary Classifier, Sentiment Analysis, and the Multi-class one, Emotion Recognition.

5 Results

In this section, we will report the results obtained on both classifiers.
As said before the analysis is made on a test set, composed of 20% of our clean dataset.

Emotion Recognition

In the following images are reported the results obtained with the *Softmax Classifier* applied on the *Emotion Recognition* problem.

Summary	Values
Accuracy	0.53
Precision	0.52
Recall	0.53
F1-score	0.52

Table 1: Accuracy and weighted averages of precision, recall and F1-score.

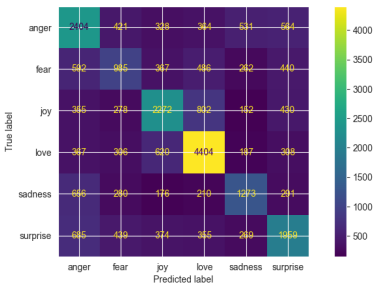

Figure 1: Confusion Matrix of the *Emotion Recognition* classifier

Sentiment Analysis

In the following images are reported the results obtained with the *Softmax Classifier* applied on the *Sentiment Analysis* problem.
More results, like the accuracy and confusion matrices of the pre-implemented methods, can be seen in the *jupyter notebook*.

Summary	Values
Accuracy	0.80
Precision	0.80
Recall	0.80
F1-score	0.80

Table 2: Accuracy and weighted averages of precision, recall and F1-score.

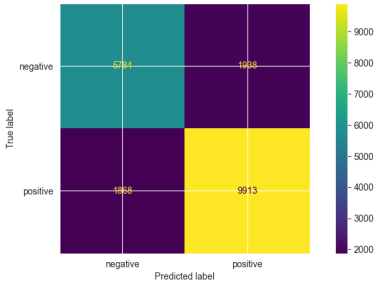

Figure 2: Confusion Matrix of the *Sentiment Analysis* classifier

6 Additional Work

Demo. We worked on implementing a small demo to better show the results we obtained with the Sentiment Analysis classifier.
The demo consists of a web application featuring *HTML5, CSS,* and *JavaScript* for the front-end and *Python* for the back-end.
After the back-end execution, the front-end shows a graphic interface where the user can input and submit a sentence. Then the submitted sentence is processed in order to extract machine learning model input features and produce the binary result, "positive" or "negative".
The decision to choose this technology is due to easier implementation and more portable application. Plus, most of the computational cost is handled by the server. The interaction between front-end and back-end is handled by the *Flask* library which allows to adopt an MVC-like pattern for the development.
Regarding the prediction of the emotion, the *joblib* library allows to export in advance of both vectorizer, for feature extraction, and trained model, for result prediction as .joblib files.

7 Conclusions

As can be seen in the *Results* section, the binary Sentiment Analysis classifier performs better than the multiclass Emotion Recognition one. This is something we expected and already talked about in the *Original Idea* section.

We probably could have obtained a better Emotion Recognition classifier by merging our two ideas. A way to do that could have been by applying both methods: first the binary classifier and then the multiclass one.

References

[1] KAGGLE dataset, *Go Emotions: Google Emotions Dataset*, https://www.kaggle.com/datasets/shivamb/go-emotions-google-emotions-dataset

[2] Wheel of emotions, *A Visual Guide to Human Emotion*, https://www.visualcapitalist.com/a-visual-guide-to-human-emotion/

[3] Notebook used as reference, *NLP ML - Emotion*, https://www.kaggle.com/code/yitianlai/nlp-ml-emotion-classification