COVID-19 report

Antonio Nappa, Celestino Santagata

July 6, 2020

Al Prof. Antonio Picariello,
un uomo straordinario ed un professionista esemplare

Contents

Introduction

The coronavirus COVID-19 pandemic is the defining global health crisis of our time and the greatest challenge we have faced since World War Two.
Countries are racing to slow the spread of the virus by testing and treating patients, carrying out contact tracing, limiting travel, quarantining citizens, and cancelling large gatherings such as sporting events, concerts, and schools. But COVID-19 is much more than a health crisis. By stressing every one of the countries it touches, it has the potential to create devastating social, economic and political crises that will leave deep scars.
We are in uncharted territory. Many of our communities are now unrecognizable. Every day, people are losing jobs and income, with no way of knowing when normality will return.
In this context we need to act. Analysis, modeling and forecasting become fundamental tools to manage this emergency; indeed, everywhere in the world methods are emerging to get to know the virus more deeply.
In Italy the Civil Protection Department provides a free access repository (CPD-github-dataset) and a dashboard that constantly monitors the situation (Italy CPD dashboard), while the World Health Organization gives a global overview (WHO dashboard).
A team of researchers from Northeastern University of Boston has created a tool. Led by Alessandro Vespignani, an Italian-American physicist, the *GLEAM Project* has developed a website that helps to predict what a possible future spread of the disease may be. *EpiRisk*, is the name of the tool created a few years ago. And obviously, given the very current coronavirus, it was used to make an estimate of any contagion that could be present. A dashboard and an application have been created, which take into account the spread of the virus on a global scale: EpiRisk_tool.
Our goal is to realize something similar, using *MicroStategy* and *Sentiment analysis* on italian tweets (with the help of *MongoDB*, to store schemaless data).

Insights

For further information:

- Twitter happiness meter: link
- Research, analysis, and news coverage of COVID-19: link
- Italian Association of Computational Linguistics repository on the COVID-19 outbreak: link
  This page groups some of the initiatives that the Computational Linguistics community is carrying out to contribute to the fight against COVID-19. It contains differents datasets (including *40wita*, the dataset we used in this project) and interesting tools.
- Measures approved by the italian Government following the international health emergency: link

Chapter 1

MicroStategy

"A set of concepts and methods to improve business decision-making by using fact-based support systems." Howard Dresner (Gartner Group), 1989

Business Intelligence is a segment of information technology which provides decision makers with valuable information and knowledge by leveraging a variety of data sources as well as structured and unstructured information. When incorporated into an information system, it enables company to utilise real-time analysis of information. Input: information and data; output: knowledge (Fig.1.1).

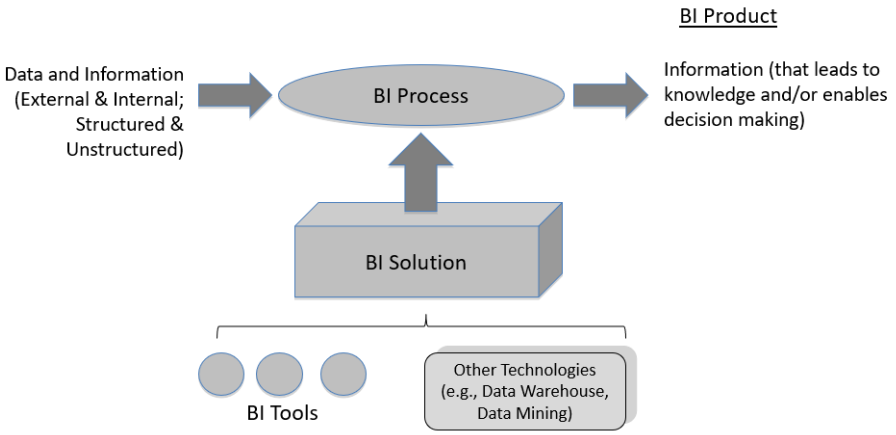

Figure 1.1: BI Product, Process, Solution, and Tools.

We used *MicroStategy*: free to download; a power-user and designer tool; no ETL tool for data integration requirements; can be used to mash, model and design engaging experiences; allows to transform and clean data; the rule is: "design once and view anywhere".

1.1 Dashboard description

A dashboard is an interactive display that can be created to show and explore business data. You can add simple visual representations of the data (called visualizations) to the dashboard to make the data easier to interpret, perform manipulations on the data to customize which information to display, organize data into multiple sheets (or "pages") to provide a logical flow to the dashboard, and so on. It is possible to quickly and easily create a polished dashboard without requiring a lot of design time using visualizations and pre-defined, presentation-quality formatting. To expand the visualization tools, we added the *VitaraCharts* package (VitaraCharts user guide).
Using the CPD dataset, we gave an overview on the national situation due to the coronavirus, focusing on the features of main interest such as total cases, numer of deaths, ecc. To better understand the geographic spread of the virus, we added a heatmap where dots size and colour are based on the number of total cases per province.
The VitaraChart package gave us the chance to introduce an animated bar chart. In particular, we used this tool to show the growing number of cases in each region (from 24/02/2020 to 21/06/2020); this specific tool represents the better way to (qualitatively) notice the fast growing rate in Lombardia.
Finally, we pointed out the daily accesses to the CPD dashboard: it is clear how the beginning of the pandemic involved major attention in the population (up to 2.6M accesses), while in the last few days it has drastically decreased (more than 5 times less).

Integration with a Python script

Thinking it as a challenge, we tried to reproduce an attractive overview of the entire features variations using a python script, *COVID_dashboard.ipynb*: importing data with pandas and plotting with matplotlib, we created a dashboard-like data visualization (fig.1.2)

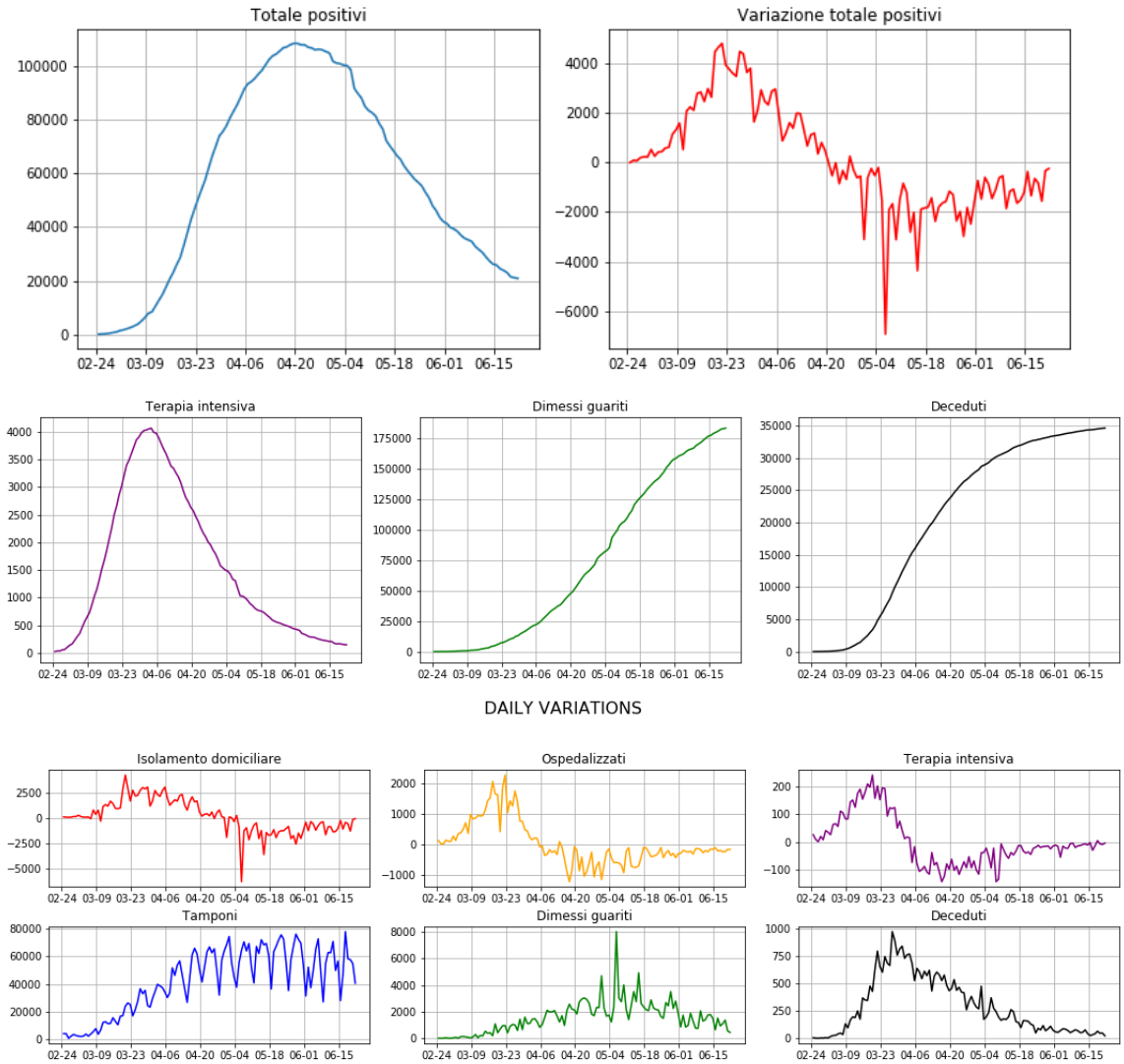

Figure 1.2: Dashboad emulation using a Python script.

Chapter 2

MongoDB

MongoDB is an open source document-oriented database (documents are stored as *json* files), which has no Data Definition Language (DDL) and is schema-less: it supports the storing of any hash with custom key-value pairs, document identifiers (_id) will be created by default for each document, custom indexes can be built to improve query performance. It employs master-slave replication with automated failover (replica sets): if a primary fails, the secondaries automatically elect a new primary. The view is a read-only collection, where rows of relational DB correspond to BSON documents: BSON stands for Binary-encoded serialization of JSON-like documents.
Instead of joining junction tables, embedding of subdocuments in documents is required: this may lead to store the same data multiple times. Moreover, no integrity check is provided, it has to be implemented at the application level.
Considering the CAP theorem, MongoDB focuses on (eventual) Consistency and Partition tolerance.
We used MongoDB Community Edition.
Why did we choose MongoDB? It is easy to work with, it supports high frequency of reads/writes and, most of all, it is very flexible (it is not tied to the specific type of query we want to submit).
The data we used in this section refers to the work of Valerio Basile and his team of the University of Turin (Covid ITA dataset). This dataset is part of the effort by the Italian Association of Computational Linguistics to collect and track resources to alleviate the national and global crisis following the COVID-19 outbreak in 2020: Computational Linguistics and the Covid-19 Outbreak.
The dataset is collected daily from 1/2/2020, by filtering *TWITA*[1] with the fol-

[1]TWITA is a collection of tweets identified as being written in the Italian langauge. This collection of tweets has been harvested using a two-pass language identification, aiming for general Italian language: 1) using cURL to download from the Twitter Streaming API searching for a list of representative words (the list consists of the most frequent lemma in the ItWaC corpus; all

lowing words: covid, covid19, covid-19, corona virus, coronavirus, quarantena, autoisolamento, auto-isolamento, iorestoacasa, stateacasa, COVID19Italia, redditodicittadinaza, eurobond, coronabond, restiamoacasa, preghiamoinsieme, NoMes, milanononsiferma, bergamononsiferma, l'italianonsiferma, abbraccciauncinese, iononsonounvirus, iononmifermo, aperisera, covidunstria, italiazonarossa, bergamoisrunning, quarantena, chiudetetutto, apritetutto, CuraItalia, ciricordiamotutto, oggisciopero, chiudiamolefabbriche, iononrinuncioalletradizioni, andràtuttobene, INPSdown, percheQuando, cercareDi, ringraziarevoglio, 600euro, CineINPS, COVID19Pandemic (fig.2.1).

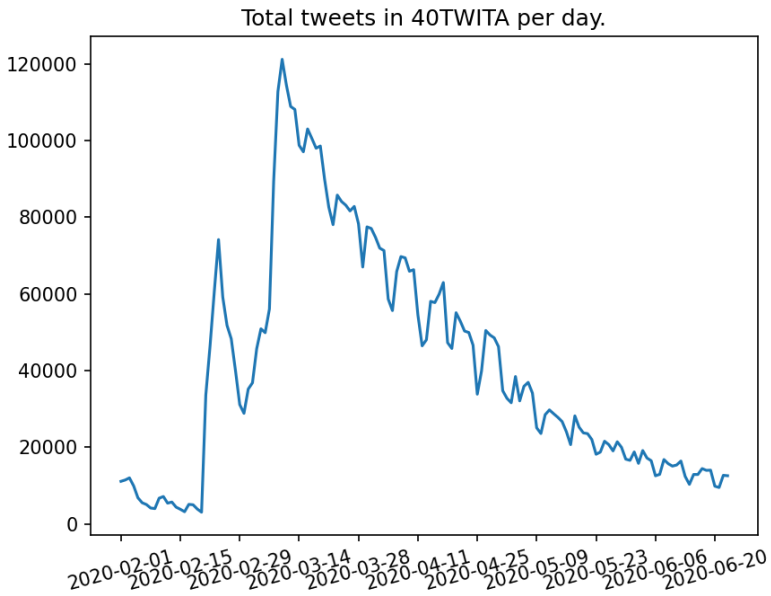

Figure 2.1: Total italian tweets per day related to COVID-19.

We considered the interval from 1/2/2020 to 23/6/2020; each day is contained in a different csv file, so the first thing we thought to do was to merge all tweets in a single file (*csvMerge.ipynb*). After that, we imported the file in MongoDB and used the query system to quickly investigate the data and to select the subsets relating to the time intervals of our interest: the intervals we chose are referred to the most relevant events and government moves linked to Coronavirus management; in order:
- 17/02 First case in Codogno

words that are frequent in other languages (English, Spanish and Portuguese) are filtered out); 2) as a second step, the tweets are input to the language identification software langid.py to detect Italian language. Link to TWITA collection.

- 09/03 National scale lockdown
- 01/04 First extension of the lockdown
- 10/04 Second extension of the lockdown
- 04/05 Beginning of the phase 2

In particular, we took the temporal window which includes two days before the event, the event itself and the day after the event. This is done in prevision of what we did next: evaluate the change in mood related to these events.
The filtering system of *MongoDB Compass GUI* (Community Edition) provides regular expression capabilities for pattern matching strings in queries, through $regex (fig.2.2). MongoDB uses Perl compatible regular expressions (i.e. "PCRE") version 8.42 with UTF-8 support.

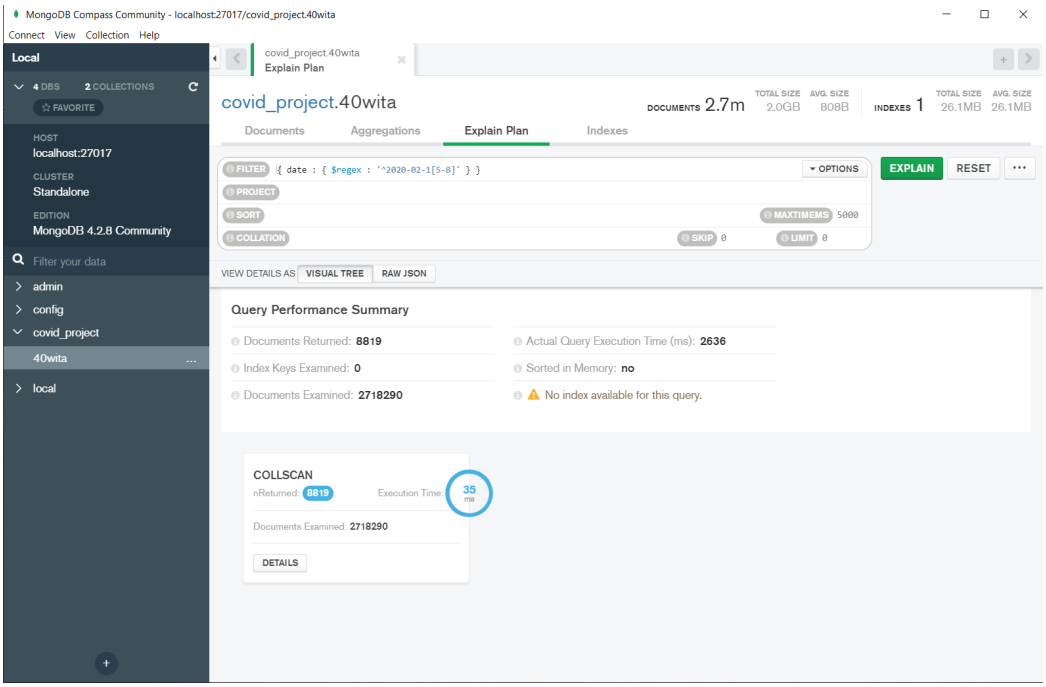

Figure 2.2: MongoDb Compass GUI.

Chapter 3

Sentiment Analysis

The extraordinary potential of the Web has made itself clear to everyone with the introduction of social networks. Web-based social applications are constantly evolving and creating increasing amount of data, this data is a treasure trove of information about user preferences, their connections, and their influences on others. Therefore, it is natural to leverage this data for analytical insights [1].
In this context, Twitter has an outstanding role. It is an online service which lets subscribers post short messages ("tweets") of up to 140 characters about anything, from good-morning messages to political stands.
Such micro-texts are a precious mine for grasping opinions of groups of people, possibly about a specific topic or product. This is even more so, since tweets are associated to several kinds of meta-data, such as geographical coordinates of where the tweet was sent from, the id of the sender, the time of the day — information that can be combined with text analysis to yield an even more accurate picture of who says what, and where, and when.

3.1 NLP and Polarity Lexicon

The last years have seen an enormous increase in research on developing opinion mining systems of various sorts applying Natural Language Processing (NLP) techniques. Systems range from simple lookups in polarity or affection resources, i.e. databases where a polarity score (usually positive, negative, or neutral) is associated to terms, to more sophisticated models built through supervised, unsupervised, and distant learning involving various sets of features.
Tweets are produced in many languages, but most work on sentiment analysis is done for English (even independently of Twitter). This is also due to the availability of tools and resources. Developing systems able to perform sentiment analysis for tweets in a new language requires at least a corpus of tweets and a polarity

lexicon, both of which, are not available for all languages [2].
So, sentiment analysis is an active research area including work on acquiring lexica of opinionated words. Most approaches to opinion mining and sentiment analysis rely on these lexicons or lists of words that are used to express sentiment. Knowing the polarity (positive, negative or neutral) of these words helps the system recognize the positive and negative sentiment in the text [3].
For this purpose, we referred to the italian lexicon **Sentix** (**Sent**iment **I**talian Le**x**icon, TWITA Sentix).

This lexicon is part of the TWITA project investigation and is the result of the alignment of several resources: *WordNet* - a large lexical database of English, *MultiWordNet* - a multilingual lexical database, *BabelNet* - a very large multilingual ontology, *SentiWordNet* - a lexical resource for opinion mining[1]. We used Sentix to classify our tweets in three different levels: positive, neutral, negative. The classification is made by averaging the scores of lemmas present in the tweet and to do this it is necessary to make a previous step: lemmatize and recognise the *POS* (Part Of Speech) of each word.
A tool used for annotating text with part-of-speech and lemma information, and available in many languages, is *TreeTagger* (documentation). TreeTagger supports the most common languages and is adaptable to other languages if a lexicon and a manually tagged training corpus are available [2].

lemma	POS	Wordnet synset ID	positive score	negative score	polarity	intensity
naturale	A	00074346	0,75	0,125	0,789	0,76

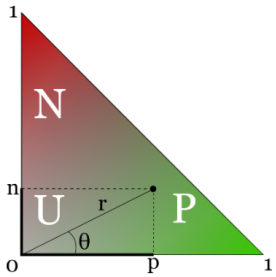

Figure 3.1: Geometrically, a synset (set of cognitive synonyms) in SentiWordNet is a point in the cartesian space where its y coordinate is the positive score and the y coordinate is the negative score. Since x+y≤1, the sentiment plane is restricted to a triangle.

[1] A valid alternative could have been the work of Maks et al., carried out within the framework of the FP7 OpeNER project (opener-project.eu). The main goal of the OpeNER project is to make available a set of open and ready to use tools to perform NLP tasks in multiple languages. As part of this project they developed the polarity lexicons in five different languages. The Italian Sentiment lexicon was developed in a semi-automated way from ItalWordNet, starting from a list of 1000 manually checked seeds. It contains 24293 lexical entries, annotated with positive/negative/neutral polarity.

[2] A new part-of-speech tagger implemented in Python using the Deep Learning library PyTorch is *RNNTagger* (link). Compared to TreeTagger, the pros of RNNTagger are: higher tagging accuracy, lemmatizes all tokens; the cons are: slower, requires PyTorch, requires a GPU for improved speed, larger parameter files, lemmas of unknown tokens are guessed and are therefore not guaranteed to be always correct.

It was developed by Helmut Schmid in the TC project at the Institute for Computational Linguistics of the University of Stuttgart and has proved to be a good tool for the italian language (Italian parameter file).
Our script, *PolProcessing.ipynb*, summarizes all these steps and gives in output a new dataframe, stored as *LabelledTweets.csv*, whose records contain the lemmas of a tweet and its polarity.

3.2 Classification model

We used the previously cited lexicon Sintex to assign polarity to tweets, in particular we assigned polarity to the subset of tweets which does not contain the time intervals of our interest (simply generated by filtering the dataset in MongoDB and exporting the resulting subset); in this way, we could train the classification model on an independent set of data.
This subset represents the basis for building our classification model.
We first built a sentiment classifier using simple Naïve Bayes. Naïve Bayes classifiers are probabilistic classifiers with strong independence assumptions between features. It is a simple technique for constructing classifiers. To improve more on accuracy, we used Multinomial Naïve Bayes.
Multinomial Naïve Bayes (MNB) is a specialized version of Naïve Bayes that is designed for text. Simple Naïve Bayes would model a document as the presence and absence of particular words [4], whereas Multinomial Naïve Bayes explicitly models the word counts and adjusts the underlying calculations to deal with. It implements Naïve Bayes for data distributed multinomially and also uses one of its version for text classification in which word counts are used to represent data. With MultinomialNB classifier, we could achieve higher accuracy.
Rennie et al. [5] discuss problems with the multinomial assumption in the context of document classification and possible ways to alleviate those problems, including the use of **tf–idf** weights[3] instead of raw term frequencies and document length normalization, to produce a Naïve Bayes classifier that is really competitive with other machine learning techniques.

The script used in this section is *ClassificationModel.ipynb*.

[3] In information retrieval, *tf–idf* or *TFIDF*, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.[1] It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf–idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general. tf–idf is one of the most popular term-weighting schemes today.

After a phase of tweets cleaning and preparation of the dataset for the model, we set up a simple Naïve Bayes classifier using the validation set approach (subdivision of 70/30 for training/testing data); we reached an accuracy of 0.59.
Subsequently, we proceeded in the analysis with a Multinomial Naïve Bayes classifier:

```
def classification_model(train_data, test_data, n_features):
     pipeline = Pipeline([('chi2', SelectKBest(chi2, k=n_features)),
                ('nb', MultinomialNB())])

     classif = SklearnClassifier(pipeline)
     classif.train(train_data)
     accuracy = classify.accuracy(classif,test_data)
     return accuracy


k_fold = 10                    #k-fold value
l = int(len(dataset)/k_fold) #subset_length
values = [10,100,1000,5000,10000,20000,50000]
accuracy = []

dataset = positive_dataset + negative_dataset + neutral_dataset
random.shuffle(dataset)

for n_features in values:
     counter = 0
     for i in range(k_fold):
          test_data = dataset[(l*i):(l*(i+1))]
          train_data = dataset[0:(l*i)] + dataset[(l*(i+1)):]
          counter += (classification_model(train_data,test_data,n_features))
     counter = counter/k_fold
     accuracy.append((n_features,counter))
```

The classifier is created using a pipeline: here we introduced the parameter *k* which let us choose the best features number based on a *chi square* statistic. The section below implements a routine for the *k-Fold Cross Validation* with k_fold=10. In fig.3.2 it is clear that the best value for *n_features* is (approximately) 1500, the corresponding accuracy is 0.70 and represents a significant improvement compared to the case of the simple Naïve Bayes classifier.

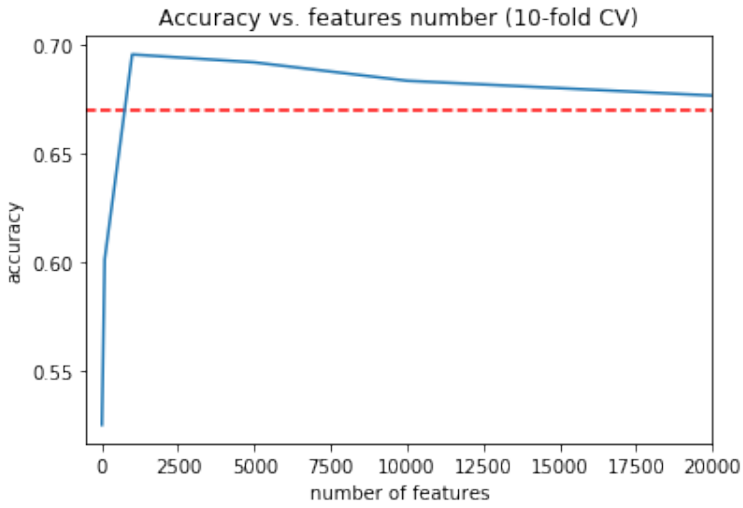

Figure 3.2: MNB 10-fold CV: selection of the number of features.

Following the work of Rennie et al., we added in the pipeline the TF-IDF weighting:

```
def classification_model(train_data, test_data, n_features):
    pipeline = Pipeline([('tfidf', TfidfTransformer()),
                ('chi2', SelectKBest(chi2, k=n_features)),
                ('nb', MultinomialNB())])

    classif = SklearnClassifier(pipeline)
    classif.train(train_data)
    accuracy = classify.accuracy(classif,test_data)
    return accuracy
```

In fig.3.3 the best value for *n_features* is (approximately) 5000, the corresponding accuracy is 0.58. The accuracy has decreased with the addition of TF-IDF and reaches the peak at 5000 features, much more than in the previous case.
- Why this changing?
Maybe the reason is due to the fact that sentiment accuracy depends on many factors: the type of data we are dealing with, the people who created the lexicon library or the pos-tagging tool, the complexity of the language and the list goes on. According to Saif et al.[6], human-agreement for Twitter sentiment analysis reaches a 0.655 value of Krippendorff's Alpha (link to the wiki), which is a good deal of agreement but still far from great. Custom machine learning models for sentiment analysis suffer of a very variable range of accuracy, but with proper training, and with a domain-specific dictionary, accuracy levels over 70%-80% can be reached. So, we can be satisfacted of our Multinomial Naïve Bayes model, even though it suffers of a lack of accuracy in presence of tf-idf weigthing.

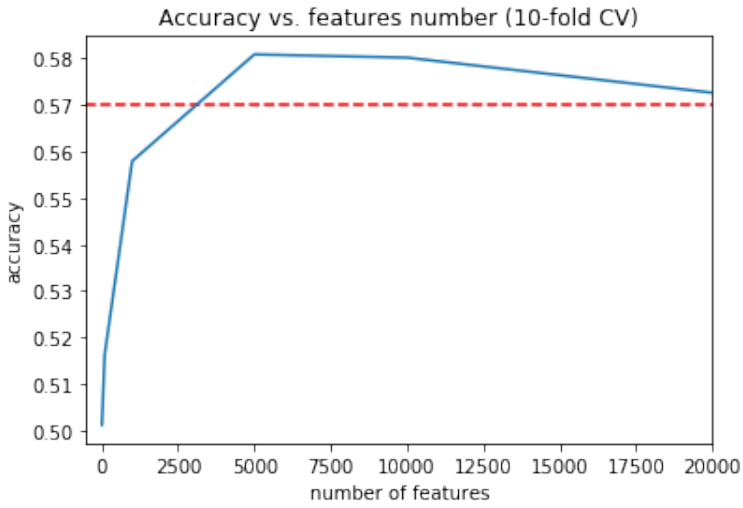

Figure 3.3: MNB 10-fold CV with TF-IDF weighting: selection of the number of features.

3.3 MNB model in action

Finally, we put our MNB classifier in action studying the five time intervals of interest:
- 17/02 First case in Codogno (from 15/02 to 18/02)
- 09/03 National scale lockdown (from 07/03 to 10/03)
- 01/04 First extension of the lockdown (from 30/03 to 02/04)
- 10/04 Second extension of the lockdown (from 08/04 to 11/04)
- 04/05 Beginning of the phase 2 (from 02/05 to 05/05)

Once classified the tweets in each subset, to show the results we decided to plot them in a bar chart using matplotlib, fig.3.4-3.8. In every subplot are reported the total number of tweets analyzed for that day and the ratio between positive and negative tweets.

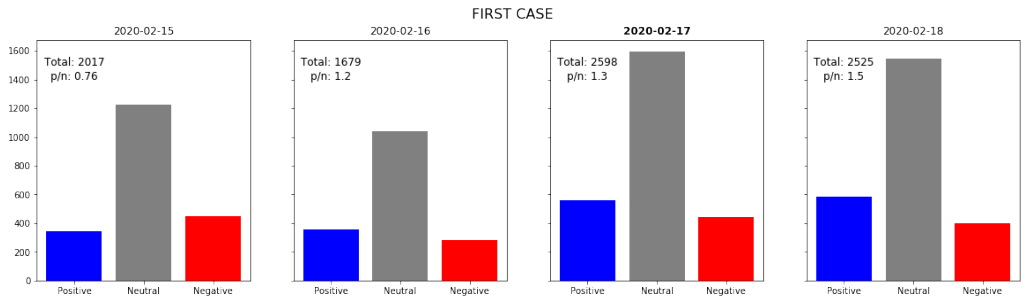

Figure 3.4: Tweets classes (positive/neutral/negative) per day referred to the First Case time interval. In bold the specific day of the event.

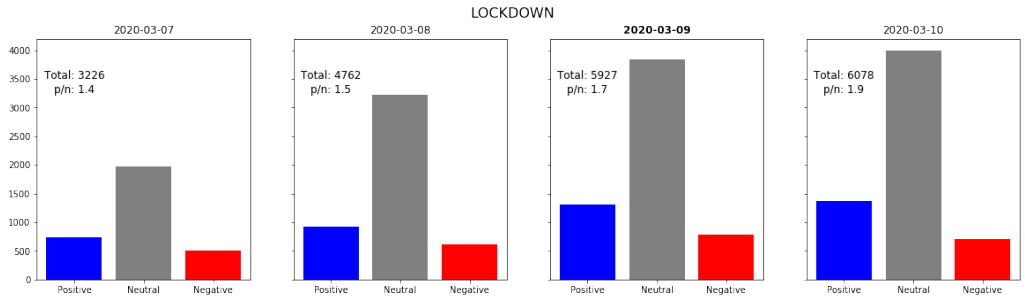

Figure 3.5: Tweets classes (positive/neutral/negative) per day referred to the Lockdown time interval. In bold the specific day of the event.

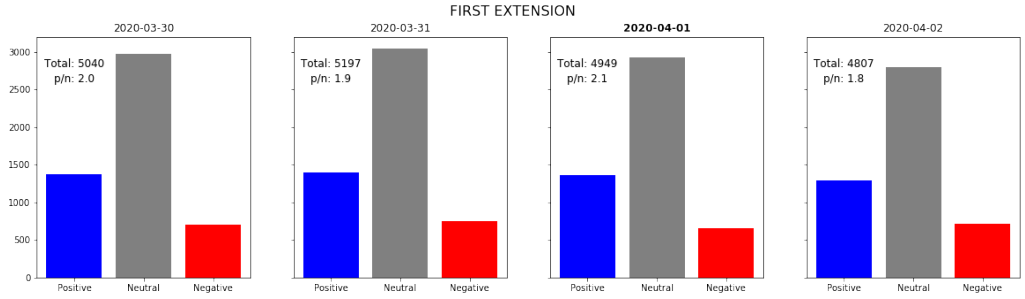

Figure 3.6: Tweets classes (positive/neutral/negative) per day referred to the First Extension time interval. In bold the specific day of the event.

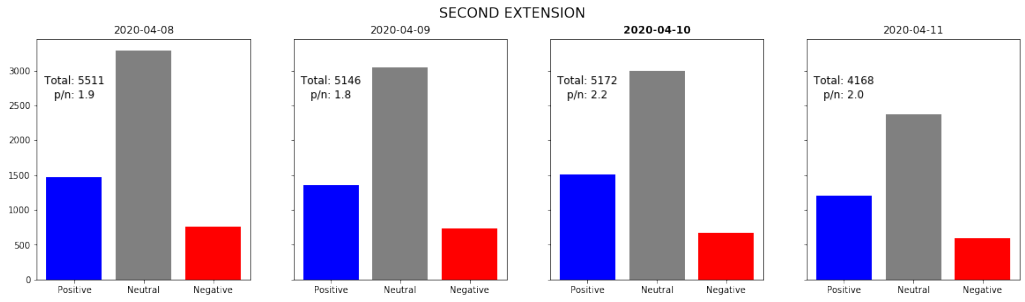

Figure 3.7: Tweets classes (positive/neutral/negative) per day referred to the Second Extension time interval. In bold the specific day of the event.

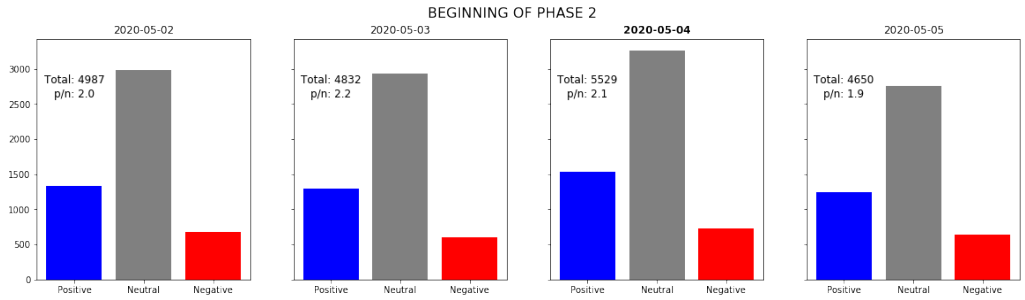

Figure 3.8: Tweets classes (positive/neutral/negative) per day referred to the Beginning of Phase 2 time interval. In bold the specific day of the event.

- What can be understood from these bar charts?
During First_case and Lockdown intervals the ratio of positive/negative tweets increases, while in the other periods it is generally stationary or slightly decreasing. We could attribute this difference to the fact that: while at the beginning the solidarity and optimism of the population saw a great diffusion, in the following phases a certain indifference or monotony took over which did not lead to considerable variations. Indeed, in the samples taken from the individual subsets (20000 tweets each) it can be seen that during First Case and Lockdown intervals the total number of tweets is greater on the day of the announcement and the following one, while in the other periods the number remains (approximately) unchanged.

3.4 Other resources

Our path is not unique, many other possibilities exist and we decided to report the most interesting we encountered:

- *Sentita*, a sentiment analysis tool for Italian (link)
  The machine learning model applied for sentiment classification is a deep learning model, a Bidirectional LSTM-CNN, that operates at word level.

- *TextBlob* (link)
  TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

- *RDRPOSTagger* (link)
  RDRPOSTagger is a robust and easy-to-use toolkit for POS and morphological tagging. It employs an error-driven methodology to automatically

construct tagging rules in the form of a binary tree [7]. Pre-trained POS and morphological tagging models are available.

- Italian affective lexicons (link)
  Collection of affective lexicons for Italian and related resources. Some of these are properly lexicons, others are resources of various types and methodologies for Italian or in any case linked to the Italian NLP community. Sentix is taken into account.

- *Open Multilingual Wordnet* (link)
  This page provides access to open wordnets in a variety of languages, all linked to the Princeton Wordnet of English (PWN). The goal is to make it easy to use wordnets in multiple languages. The individual wordnets have been made by many different projects and vary greatly in size and accuracy. You can access the wordnets through the (python) Natural Language Tool-Kit wordnet interface (NLTK).

Conclusions

With this project we aimed to collect a wide documentation related to the Coronavirus, the NLP and, in particular, its application in the world of Twitter.
NLP main field of work is Sentiment Analysis, a research area in continuous expansion, which observes the reactions aroused in the human soul following certain events and keeps track of their constant evolution.
Using an Italian Lexicon (*Sentix*) and a POS-tagging tool (*TreeTagger*), we classified the tweets of our dataset and trained a Multinomial Naïve Bayes model, reaching an accuracy level of 0.70. This model let us study the sentiment changing during specific time intervals of the pandemic spread.
Better handling of emojis, domain-specific dictionary, improved Italian pos-tagging tool and optimal boundaries/cutoff for positive-neutral-negative subdivision can definetly improve the results, but our model can be considered quite accurate since the typical human-agreement for Twitter sentiment analysis reaches a 0.655 value of Krippendorff's Alpha.

Bibliography

[1] Charu C Aggarwal. *Data mining: the textbook*. Springer, 2015.

[2] Valerio Basile and Malvina Nissim. Sentiment analysis on italian tweets. In *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, 2013.

[3] Isa Maks, Ruben Izquierdo, Francesca Frontini, Rodrigo Agerri, Andoni Azpeitia, and Piek Vossen. Generating polarity lexicons with wordnet propagation in five languages. *Proceedings of LREC2014, Reykjavik*, 2014.

[4] Andrew McCallum, Kamal Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48. Citeseer, 1998.

[5] Jason D Rennie, Lawrence Shih, Jaime Teevan, and David R Karger. Tackling the poor assumptions of naive bayes text classifiers. In *Proceedings of the 20th international conference on machine learning (ICML-03)*, pages 616–623, 2003.

[6] Hassan Saif, Miriam Fernandez, Yulan He, and Harith Alani. Evaluation datasets for twitter sentiment analysis. *Emotion and Sentiment in Social and Expressive Media*, page 9, 2013.

[7] Dat Quoc Nguyen, Dai Quoc Nguyen, Dang Duc Pham, and Son Bao Pham. RDRPOSTagger: A Ripple Down Rules-based Part-Of-Speech Tagger. In *Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 17–20, Gothenburg, Sweden, April 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/E14-2005.