ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA

Modelado de temas para el análisis de la similitud entre usuarios en Twitter

---

Topic Modeling for Analysing Similarity between Users in Twitter

Realizado por

D. Haritz Puerto San Román

Tutorizado por

Dr. D. Ezequiel López Rubio

Departamento

Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

MÁLAGA, junio 2017

Fecha defensa:

El Secretario del Tribunal

Topic Modeling for Analyzing Similarity Between Users in Twitter

Haritz Puerto San Román

Spanish Abstract

La minería de datos en redes sociales está ganando importancia debido a que permite realizar campañas de marketing más precisas. Por ejemplo, Google realiza un análisis de todos nuestros datos: vídeos que vemos, términos que buscamos, páginas webs a las que accedemos, aplicaciones que descargamos, etc. para conocernos mejor y mostrarnos publicidad personalizada.

LDA es un modelo estadístico generativo para modelar documentos. Existen diversos algoritmos que dado un conjunto de documentos permiten obtener un módelo LDA que podría haber generado esos documentos. Con ese modelo es posible observar los temas usados en esos documentos y las palabras más relevantes para cada tema.

En el presente trabajo se pretende realizar una primera aproximación a la minería de datos en Twitter. Para ello, usando la API de Twitter se han descargado tweets de diversos usuarios y de sus seguidores. Posteriormente se han procesado esos Tweets generando documentos y se ha aplicado la implementación de Gensim del algoritmo Online LDA para obtener los temas de los documentos. Posteriormente, se han comparado los temas de los usuarios con los de sus seguidores.

También se proporciona un análisis del estado del arte de la minería de datos en Twitter.

Palabras clave— Aprendizaje computacional, Minería de datos, Procesamiento de lenguaje natural, Modelado de temas, LDA, Ciencias Sociales computacionales, Twitter

English Abstract

Data Mining in social networks is becoming increasingly important since it allows to perform effective marketing campaigns. For instance, Google carries out an analysis of all our data: videos we watch, queries we search, web pages we access, apps we download, etc. to know us better and show us personalized ads.

LDA is a generative statistical model for modeling documents. There are several algorithms which given a set of documents allow us to obtain an LDA model which can generate those documents. It is possible to see the topics and the most relevant words of each topic using that model.

The goal of this work is to attempt a first approach to Data Mining in Twitter. In order to do that, using the Twitter's APIs tweets from several users and their followers have been downloaded. After that, those Tweets have been processed to generate documents. Then, using the implementation of Online LDA offered by Gensim, the topics of those documents have been obtained. Finally, the topics of the users and their followers have been compared.

An analysis of the current state of the state of the art of the Data Mining in Twitter is also presented.

Keywords— Machine Learning, Natural Language Processing, Data Mining, Topic Modeling, LDA, Computational Social Science, Twitter

Acknowledgements

I wish to express my sincere gratitude to my advisor, Prof. Ezequiel López Rubio, for his guidance throughout this Bachelor's Thesis.

Last but not the least, I wish to thank my family who has always respected my choices, constantly encouraged me to pursue my goals and for all what they have done for me throughout the years.

Contents

Chapter 1

Introduction

1.1 Motivation

Communication in humans is something unique. No other animal have such a sophisticated way of communication as humans. First, written systems were invented. Next, codes to communicate with fire were invented. After that, telegraph was invented. Cell phones followed and finally the Internet. Online social networks have become important in everyday life. They have evolved a lot since the creation of Facebook. Every minute a huge amount of data is uploaded to each social network. If we analyze that data, we can obtain very useful information. This is often call "Computational Social Science". We upload a lot of information about the things we like, dislike and we do. Many companies have realized that they can take advantage of it to offer better products to their clients. All big companies have accounts on Twitter, Facebook and other social networks in order to be able to listen to their users. That huge amount of data can be seen as a valuable source of information, however, processing that data is an arduous task. A single person is not able to process it, so algorithms to automatize that data mining are needed. It is possible to obtain information from the graphs formed by users. For instance, there are studies to detect anomalous adult advertisers in the who-follows-whom Twitter social network [1]. Another used technique is Topic Modeling. It can be used to obtain the topics of the texts written by the users. Those topics can be used to improve search results and advertisements.

There are different approaches to obtain topics in a document. One is Latent Semantic Analysis (LSA). It assumes that words that are close in meaning will occur in similar pieces of text. Nonetheless, it has several drawbacks. It does not scale well and it only assigns one topic per document which is not always true.

Latent Dirichlet Allocation (LDA) is another technique which assigns several topics per document. In this work, we will work with it and we will elaborate an in-depth

explanation of how it works.

1.2 Problem

In some social networks, it is common to be connected only to your friends and relatives while in others like Twitter or Instagram, it is common to be connected to strangers. One reason for this behavior is that it can give more followers. In Twitter, some people wants as many followers as possible so they follow strangers because they expect them to follow back although it does not happen always.

It is a fact that a user can follow many other users but he might not be interested in all the content of those users. For instance, user X likes sports, TV shows, rock music, theatre and politics. This user can follow user Y who publishes content about sports, cell phones, computers, gadgets, environment and cars. In this case, X may only be interested in the sport contents that user Y publishes. Those users are not very similar because the disjunction of their common topics is smaller than the symmetric difference. The goal of this work is to obtain the topics of a few users and their followers to compare them. This can be used by Twitter to improve the timeline in order to show first Tweets of the most similar users one follows.

1.3 Introduction to Machine Learning

Machine learning is a subfield of Computer Science which studies algorithms that can learn from data and make predictions from it. It is related to mathematical optimization and statistics.

Machine learning tasks can be classified according to:

- Feedback available to the system:
  - Supervised learning: The system is given input data with the outputs. The goal is to learn a rule to map new inputs to outputs.
  - Unsupervised learning: The system is given input data without the outputs. The goal is to find a structure in the data.
  - Reinforcement learning: The system interacts with a dynamic environment to achieve a goal. For instance, driving a car.
- Desired output:
  - Classification: Inputs are divided into two or more classes. The ML algorithm must generate a model that can assign one or more labels to new inputs.

- Regression: Inputs have a mapping to a set of continuous values. The ML algorithm must generate a model that can assign a value in the continuous space to new inputs.
- Clustering: It is an unsupervised task. The input must be divided into groups previously unknown.
- Density estimation: It is the estimation of a probability density function based on the observed data.
- Dimensionality reduction: Maps the inputs to a lower-dimensional space.

There are several approaches to machine learning to perform those tasks:

- Decision Trees: It uses a tree-like graph of decisions and their possible consequences as a predictive model.

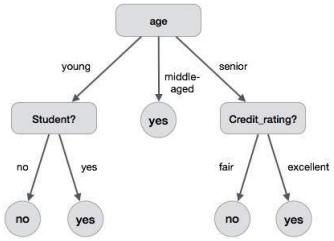

Figure 1.1: Decision tree example

- Association Rules: It is a rule-based machine learning method for discovering relations between variables. An example of rule is: (onion, potato)->beer.
- Artificial Neural Networks: They are a computation model based on a large connection of simple units called neurons. A neuron can be activated if the signals it receives are strong enough. Then, that neuron sends a signal to other neurons.

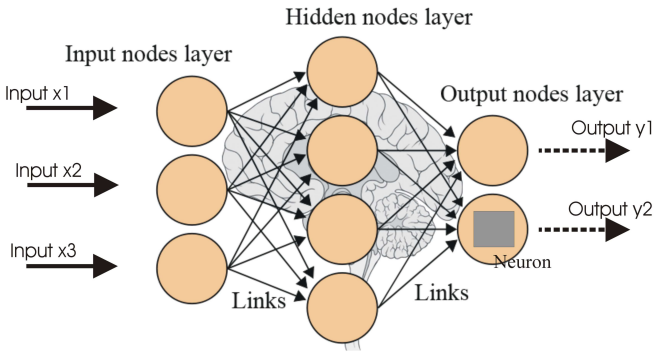

Figure 1.2: Artificial Neural Network example

- Deep Learning: It is an artificial neural network with multiple hidden layers of units between the input and output layers.
- Support Vector Machines: It constructs an hyperplane which can be used for classification, regression, or other tasks.

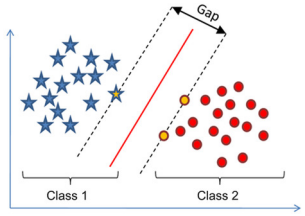

Figure 1.3: SVM example

1.4 Introduction to Natural Language Processing

It is a subfield of Computer Science and Artificial Intelligence which deals with the interaction between human language and computers. Computers use NLP to obtain the meaning of a text or a conversation in a human language. Next, a list of a few tasks of NLP is shown:

- Morphological segmentation: Given a word, get its morpheme.
- Machine translation: Translate a text from one human language into another one.
- Natural language generation: Given a database, create text in a human language using the data available in that database.
- Question answering: Given a question in a human language, give an answer in that language.
- Sentiment analysis: Given a text, find out the sentiment it reflects. For example: Is the writer happy?
- Topic segmentation: Given a text, divide it into segments where each segment represents a topic.
- Speech recognition: Given a sound clip with a speech, determine the textual representation of that speech.

In this Bachelor's thesis, we will work with topic segmentation. In order to do that, a clustering approach has been followed using Latent Dirichlet Allocation.

1.5 State of the Art

Data Mining in social networks is a hot topic nowadays. Makoto Okazaki, Takeshi Sakaki and Yutaka Matsuo propose an earthquake detector analyzing tweets in [2]. When an earthquake occurs, many people write tweets about it. Each Twitter user is considered as a sensor. Using Kalman and particle filtering it is possible to obtain an approximation of the location of the earthquake.

It has been possible to study the phenomena of multilingual societies and the role that bilinguals play in them using LDA and data from Twitter. Suin Kim et al. have found in [3]:

- Users of local languages (English in the U.S. for example) have a higher number of followers than others.
- Bilinguals act as a bridge, i.e.: English is a hub language that connects people writing in German with people writing in French
- Although we may think that bilinguals write always in English to gain a wider audience this is not the case. They usually mimic the language mix of their followers
- Bilinguals use different languages for different topics. For instance, in Switzerland, English is used for tourism and leisure-related tweets while, French or German is used for politics, news and recruitment among others.

Suin Kim, JinYeong Bak, and Alice Haeyun Oh use Sentence-LDA in [4] to discover that Twitter users are likely to express a positive emotion regardless of the emotion in the previous tweets. There are topics that can change the emotion of a conversation partner. In most conversations, the interlocutors share a common emotion, but sometimes they have strong opposite emotions like feeling upset - sympathy or complaining - making an apology.

Daniel Ramage, Susan T. Dumais, and Daniel J. Liebling use Labeled-LDA to map tweets into four topics: substance, social, status and style in order to characterize users according to the topics they write about in [5].

Twitter lists have been analyzed using LDA. Jeon Hyung Kang and Kristina Lerman show in [6] that topically similar users are more likely to be linked via a follow relationship than less similar users.

Kyungyup Daniel Lee, Kyung-Ah Han, and Sung-Hyon Myaeng use LDA to detect fake reviews in [7].

Liangjie Hong and Brian D. Davison propose in [8] to aggregate tweets to improve the quality of the models. It is stated that the Author-Topic model yields to worse results than standard LDA on aggregated tweets.

While many researchers have focused on the detection of new topics and their propagation, Gwan Jang and Sung-Hyon Myaeng have focused in the analysis of the subtopics by region and by time in [9]. A topic can have several subtopics which can change over time or location, for example, the topic "iPhone" can have the subtopic "release" when it is announced and can have the subtopic "new iOS" after a new version of its O.S.

Other researchers have studied the spreading of information in social networks. For instance, Aisylu Khairullina et al. found out in [10] that the Independent Cascade model cannot illustrate the information diffusion process in Vkontakte, a Russian social network. Nevertheless, they could show that the topic of the post is as important as the number of friends and relation strength in the information diffusion process. They also suspected that the relations play a crucial role in the information propagation so they proposed to use in the future the Linear Threshold model to examine if the diffusion in Vkontakte follows the model and users are dependent on other users opinion.

Jagan Sankaranarayanan et al. propose in [11] a news processing system called TwitterStand. Using real-time data from Twitter, they filter the Tweets about news using a naive Bayes classifier and remove all those Tweets which are not about news. Then, Tweets are clustered and finally, each cluster is geolocated in order to show it on a map.

Andres Lou, Diana Inkpen, and Chris Tanasescu explain how they automatically classify poems in [12]. They focus on the classification of poem based only on subjects. They use the categories and subcategories proposed by the Poetry Foundation. First, they extract features using tf-idf and LDA and with those features, they use an SVM algorithm to classify poems. All their results were acceptable with an AUC over 0.6.

Thomas L. Griffiths and Mark Steyvers use LDA in [13] for automatically extracting the topics of the abstracts from PNAS from 1991 to 2001. Their method allows them to express the similarity between abstracts. They also plot graphics with the popularity of research topic over time.

Chapter 2

Background

In this chapter, we will introduce some concepts which will be needed throughout the course of this work. First, some probability concepts will be defined. After that, those concepts will be used to define graphical models and topic models. Then, LDA, the topic of this work, will be introduced and last but not least, a brief introduction to Data Mining will be explained.

2.1 Probability concepts

2.1.1 Joint Probability Distribution

Given 2 random variables, X and Y, the joint probability distribution of X and Y is the probability distribution of both events happening at the same time.

2.1.2 Marginal Distribution

It is the probability distribution of a subset of the set of random variables. Given 2 random variables, X and Y, and their joint probability distribution: $F_{X,Y}(x, y)$, the marginal distribution of x is:

If X and Y are discrete random variables:

$$P(X = x) = \sum_y P(X = x, Y = y) = \sum_y P(X = x|Y = y)P(Y = y)$$

If X and Y are continuous random variables:

$$p_x(x) = \int_y p_{X,Y}(x, y)dy = \int_y p_{X|Y}(x|y)p_y dy$$

2.1.3 Prior Probability Distribution

It is the probability distribution that expresses the belief about a random variable before some evidence is given.

2.1.4 Likelihood Function

The likelihood of a parameter value, θ, given the outcome x, is equal to the probability of that outcome given the parameter which is:

$$\mathcal{L}(\theta|x) = P(x|\theta)$$

Let's see it with an example:

If f is a Gaussian density, it is characterized by two parameters: the mean, μ, and the standard deviation, σ. So we can write f as $f(x, \mu, \sigma)$. In addition, those two parameters can be written as a vector, θ. Then, f could be written as $f(x, \theta)$. When we think of f as a density function, θ is constant and the function varies in x. It is the opposite case in the likelihood. x is constant and θ is the one that varies.

If x is a discrete random variable, its definition is as follows:

$$\mathcal{L}(\theta|x) = p_\theta(x) = P_\theta(X = x)$$

If x is a continuous random variable, its definition is as follows:

$$\mathcal{L}(\theta|x) = f_\theta(x)$$

2.1.5 Posterior Probability Distribution

It is the probability distribution of a random variable given some evidence. More formally, it is the probability of the parameter θ given the evidence X: $p(\theta|X)$.

Given a prior belief $p(\theta)$ and observations x with the likelihood $p(x|\theta)$, then the posterior probability is defined as:

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)}$$

2.1.6 Latent Variables

They are variables which are inferred from other variables that are observed. Mathematical models which aim to explain observed variables in terms of latent variables are called latent variable models.

Sometimes latent variables correspond to something of the real world, but for some reason, it is not possible to measure them. In this situation, the term "hidden variables" is commonly used. Other times, latent variables correspond to abstract concepts. In that case, the term "hypothetical variables" is used.

One advantage of using latent variables is that it reduces the dimensionality of the data. A large number of observable variables can be aggregated in a model to represent an underlying concept, making it easier to understand the data.

2.1.7 Simplex

In geometry, a simplex is a generalization of a triangle or tetrahedron to arbitrary dimensions. A k-simplex is a k-dimensional polytope which is the convex hull of its k + 1 vertices. More formally, a simplex is determined by the set:

$$S = \{x \in R^n : x_i \geq 0, \sum_{i=0}^{n} x_i = 1\}$$

Given any multinomial distribution of n+1 variables, the probabilities of the outcomes can be any of the values of the following set:

$$S = \{x \in R^n : x_i \geq 0, \sum_{i=0}^{n} x_i = 1\}$$

S is by definition a simplex.

2.1.8 KL Divergence

The Kullback–Leibler divergence is a measure, not a metric, of the difference between two probability distributions P and Q. The Kullback–Leibler divergence from Q to P, denoted $D_{KL}(P||Q)$ is the amount of information lost when Q is used to approximate P. This measure is not symmetric, i.e.: in general $D_{KL}(P||Q) \neq D_{KL}(Q||P)$. P usually represents the real distribution of the data while Q the model or approximation of P. For discrete probability distributions P and Q, the KL Divergence is defined as:

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$$

For continuous probability distributions P and Q, the KL Divergence is defined as:

$$D_{KL}(P||Q) = \int_{-\infty}^{\infty} P(x) log \frac{P(x)}{Q(x)} dx$$

2.2 Dirichlet Distribution

It is a distribution over the space of multinomial distributions parameterized by a vector α of positive reals i.e., to generate data X from a Dirichlet distribution with parameter α, we need to draw a $\vec{p} \sim Dir(\vec{\alpha})$, and then draw $X \sim Multi(\vec{p})$, therefore, there is one level of indirection. Its probability distribution function, pdf, is the following:

$$f(x_1, ..., x_k; \alpha_1, ...\alpha_n) = \frac{1}{B(\alpha)} \prod_{i=1}^{k} x_i^{\alpha_i - 1}$$

It is commonly used as the prior in Bayesian Statistics. In LDA it will be its prior distribution and the topics will follow a multinomial distribution.

2.3 Graphical Models

It is a probabilistic model for which a graph shows the conditional dependence structure between random variables. The graph is composed by:

- Nodes: represent random variables.
- Edges: represent dependencies.
- Shaded nodes: represent observed variables.
- Plates: represent replicated structures.

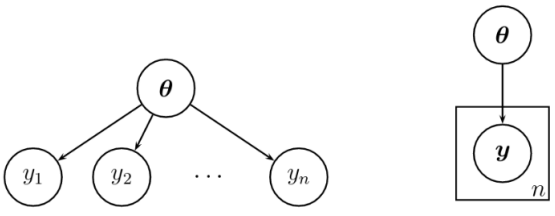

Figure 2.1: Example of a graphical model

In figure 2.1 the edge between Theta and y_1 means that y_1 depends on Theta. As there is a repetition of this structure, it can be written with the plate notation as shown at the right of the image. The letter n denotes the number of times that structure is replicated. The joint probability distribution is:

$$P(\theta, y_1, ..., y_n) = P(\theta) \prod_{i=1}^{n} P(y_i|\theta)$$

2.4 Topic Model

A topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. It is a common text-mining tool for discovering hidden semantic structures in texts. Topic models can help to organize large collections of unstructured texts. Originally, it was developed for text mining, but nowadays, it has applications in bioinformatics.

2.5 Latent Dirichlet Allocation

In 2003, D. Blei, Michelle Jordan and Andrew Ng proposed a model named Latent Dirichlet Allocation (LDA) in [14]. It is a generative statistical model for modeling documents. LDA assumes that a document is a bag of words, i.e.: the document "Hello my name is Haritz" is the same as "Haritz name Hello my is". Furthermore, LDA assumes that a document is created as follows:

1. Pick N, the number of words.
2. Choose $\theta \sim Dir(\alpha_1, ..., \alpha_k)$. θ is a topic mixture which lies in the $k-1$ dimensional simplex.
3. For each of the N words:
   (a) Choose a topic $z_n \sim Multinomial(\theta)$.
   (b) Choose a word w_n from $P(w_n|z_n)$, a multinomial probability conditioned on the topic z_n.

More formally, the joint probability distribution of a topic mixture θ, a set of N topics z and a set of N words w is given by:

$$p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta) p(w_n|z_n, \beta)$$

Integrating over θ and summing over z, we obtain the marginal distribution of a document:

$$p(\mathbf{w}|\alpha, \beta) = \int p(\theta|\alpha) \left(\prod_{n=1}^{N} \sum_{z_n} p(z_n|\theta) p(w_n|z_n, \beta) \right) d\theta$$

Finally, taking the product of the marginal probabilities of single documents, we obtain the probability of a corpus:

$$p(D|\alpha,\beta) = \prod_{d=1}^{M} \int p(\theta_d|\alpha) \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\theta_d) p(w_{dn}|z_{dn},\beta) \right) d\theta_d$$

The parameters α and β are corpus-level parameters. They are sampled once in the process of generating a corpus. The variables θ_d are document-level variables sampled once per document. Finally, the variables z_{dn} and w_{dn} are word-level variables and are sampled once for each word in each document.

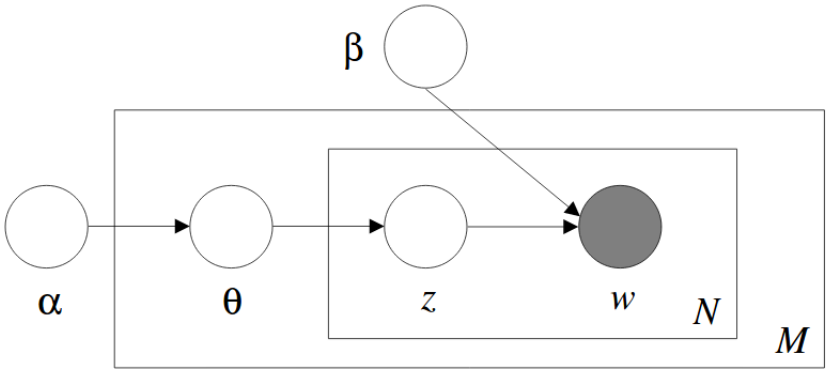

Figure 2.2: Graphical model representation of LDA

LDA does not assign names to the topics. The topics it provides are abstract so we have to assign labels to each topic in order to understand them better. Let's see how a document is created according to LDA with an example. Let's say we want a document with five words and two topics: food and animals. Each topic is composed of a map of (word, probability). The topics are the following:

- Topic 1: (broccoli, 0.3), (banana, 0.15), (breakfast, 0.1), (munching, 0.1), ... We will assign the label food.
- Topic 2: (chinchillas, 0.2), (cats, 0.2), (cute, 0.2), (hamster, 0.15), ... We will assign the label animals.

A possible document created using that data is: "Broccoli cute munching cats banana". As we can see, the document is not grammatically correct because documents are bags of words for LDA and nothing else. When we apply LDA to a document written by a person, it will assume that the document has been written as in the example, therefore, it will try to find the mixture of topics and their multinomial distributions which have generated it.

2.5.1 Inference

In order to use LDA, computing the posterior distribution of the hidden variables given a document is needed:

$$p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta) = \frac{p(\theta, \mathbf{z}, \mathbf{w}|\alpha, \beta)}{p(\mathbf{w}|\alpha, \beta)}$$

Unfortunately, this distribution is intractable. Nevertheless, a wide variety of approximate inference algorithms are available for LDA.

2.5.2 Variational Inference

A simple way to obtain a tractable family of lower bounds is to consider simple modifications of the original graphical model in which some of the edges and nodes are removed.

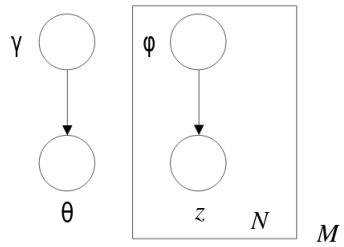

Figure 2.3: Graphical model representation of the variational distribution used to approximate the posterior in LDA

The distribution of this simplified graphical model is:

$$q(\theta, \mathbf{z}|\gamma, \varphi) = q(\theta|\gamma) \prod_{n=1}^{N} q(z_n|\varphi_n)$$

where the Dirichlet parameter γ and the multinomial parameters $\varphi_1, ...\varphi_n$ are the free variational parameters. To determine these parameters, it is needed to set up the following optimization problem:

$$(\gamma^*, \varphi^*) = \arg\min_{(\gamma,\varphi)} D(q(\theta, \mathbf{z}|\gamma, \varphi) \;||\; p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta))$$

Minimizing the Kullback-Leibler (KL) divergence between the variational distribution and the true posterior $p(\theta, \mathbf{z}|\mathbf{w}, \alpha, \beta)$ is possible to obtain the optimal values of the variational parameters. The minimization can be achieved via an iterative fixed-point method. Developing that formula, we obtain the following two formulas:

$$\varphi_{ni} \propto \beta_{iw_n} exp\{E_q[log(\theta_i)|\gamma]\}$$

$$\gamma_i = \alpha_i + \sum_{n=1}^{N} \varphi_{ni}$$

The expectation in the multinomial update can be computed as follows:

$$E_q[log(\theta_i)|\gamma] = \Psi(\gamma_i) - \Psi(\sum_{j=1}^{k} \gamma_j)$$

All the calculations can be read in [14].

This yields to the following algorithm:

Algorithm 1 Variational Inference for LDA

$\varphi_{ni}^0 \leftarrow 1/k$ ▷ for all i and n
$\gamma_i \leftarrow \alpha_i + N/k$ ▷ for all i
repeat
    for `n = 1 to N` **do**
        for `i = 1 to k` **do**
            $\varphi_{ni}^{t+1} \leftarrow \beta_{iw_n} exp(\Psi(\gamma_i^t))$
            `normalize` φ_{ni}^{t+1} `to sum 1`
        end for
        $\gamma^{t+1} \leftarrow \alpha + \sum_{n=1}^{N} \varphi_n^{t+1}$
    end for
until convergence

From the pseudo code it is easy to see that each iteration of variational inference of LDA takes $\mathcal{O}(Nk + N)$ operations.

2.6 Related models

2.6.1 Pachinko Allocation Model

Latent Dirichlet allocation (LDA) and other related topic models are very popular tools for summarization of texts. However, LDA does not capture correlations between topics. Wei Li and Andrew McCallum introduced the the Pachinko Allocation Model (PAM) in [15]. It uses a directed acyclic graph (DAG) structure to represent topic correlations. Each leaf node is associated with a word and each non-leaf node corresponds to a topic. In figure 2.4 (c) and (d), it is possible to see the structure. For instance, given the topics: "health", "cooking", "insurance" and "drugs" the first topic occurs often with the second one and with the two remaining ones. Each of these topics would have a node and all of them would be at the same level. There

would be two additional nodes at a higher level. One would be the parent of cooking and health and the other one the parent of health, insurance and drugs.

LDA can be viewed as a special case of PAM: the DAG corresponding to LDA is a three-level hierarchy consisting of one root at the top, a set of topics in the middle and words at the bottom. The root is fully connected to all the topics, and each topic is fully connected to all the words. It is shown in 2.4 (b).

In PAM each topic t_i is associated with a Dirichlet distribution $g_i(\alpha_i)$, where α_i is a vector with the same dimension as the number of children in t_i . To generate a document d, we follow the following process:

1. Sample $\theta_{t_1}^{(d)}, \theta_{t_2}^{(d)}, ..., \theta_{t_s}^{(d)}$ from $g_1(\alpha_1), g_2(\alpha_2), ..., g_s(\alpha_s)$, where $\theta_{t_i}^{(d)}$ is a multinomial distribution of topic t_i
2. For each word w in the document:
   - Sample a topic path z_w of length L_w
   - Sample word w from $\theta_{z_{w(i-1)}}^{(d)}$

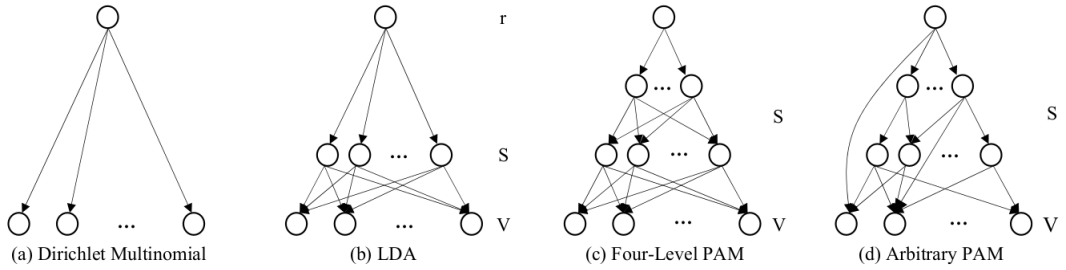

Figure 2.4: Pachinko Allocation

2.6.2 Latent Semantic Indexing

Latent Semantic Indexing is a technique that projects queries and documents into a space with "latent" semantic dimensions. This space has fewer dimensions than the original space so it is a method for dimensionality reduction like PCA. Latent semantic indexing is the application of Singular Value Decomposition or SVD to a word-by-document matrix.

SVD takes a matrix A and represents it as $\hat{A}$ in a lower dimensional space such that the "distance" between the two matrices measured by the 2-norm is minimized:

$$\Delta = |A - \hat{A}_2|$$

LSI assumes the following:

- Documents are represented as "bags of words" like LDA.

- Concepts are represented as sets of words that usually appear together in documents. For example "banana", "apple", and "strawberry" may usually appear in documents about fruits.
- Words are assumed to have only one meaning. Polysemy is a difficult problem for LSI so it uses this simplification.

2.6.3 Hierarchical LDA

This is a variant of LDA in which the goal is to obtain a hierarchy of topics given a collection of documents. General topics are at a high level while very specific topics are at lower levels. To generate a document, it samples a topic path from the hierarchy and then samples words from those topics. Therefore, a document can be about a mixture of Computer Science, Artificial Intelligence and Autonomous Robotics.

2.7 Data Mining

Data Mining is an interdisciplinary subfield of Computer Science that deals with the extraction of non-trivial, implicit, previously unknown and potentially useful patterns or knowledge from huge amount of data. It is also known as Knowledge Discovery in Databases (KDD) [16].

KDD has the following stages:

1. Data Selection
2. Data Cleaning and Preprocessing
3. Data Reduction and Transformation
4. Data Mining
5. Interpretation and Evaluation

2.7.1 Data Selection

It involves the search of datasets to use in the following steps. It is possible to use databases, plain-text documents, formatted documents, spreadsheets, etc.

2.7.2 Data Cleaning and Preprocessing

Data in the real world is dirty, i.e.: it is incomplete, noisy, inconsistent, etc. This affects to the machine learning algorithms to obtain patterns, so this step is needed to obtain a good data set. According to [16], it may take 60% of the total time used in the KDD process.

2.7.3 Data Reduction and Transformation

It includes the combination of data from different sources, resolving conflicts in units, removal of redundancy, aggregation, generalization and normalization of data.

2.7.4 Data Mining

It is the application of machine learning algorithms to the data obtained in the previous steps. The problems it can solve among others are:

- Classification
- Regression
- Clustering
- Association rules

2.7.5 Interpretation and Evaluation

It is the evaluation Interpretation and Evaluation by the domain experts of the patterns or new knowledge found. Visualization tools are usually needed.

Chapter 3

Proposal

In this chapter, the functionality of the libraries needed will be explained. First, we will show how the Twitter REST and Streaming APIs work. Then, we will explain how to preprocess the downloaded Tweets. After that, the Gensim's implementation of Online LDA will be explained and the results will be interpreted using pyLDA, a visualization tool for LDA.

3.1 How to retrieve Tweets

Twitter offers an API to read and write Twitter data, create new Tweets, read user profiles and follower data, and more. The REST API identifies Twitter applications and users using OAuth and their responses are in JSON format.

3.1.1 How to gain access to the Twitter API?

At the apps page of Twitter, https://apps.twitter.com/ it is possible to create a new application. Since we do not have a web page, we can put: "https://www.twitter.com" in the form when we are asked to enter our website. Then, we need to search the API Key and the API Secret. We will need them for accessing Twitter in our Python code.

Instead of using directly the Twitter API, Twython has been used. It is a wrapper for the Twitter API which allows us to get the responses from Twitter in Python objects instead of JSON.

Figure 3.1 shows the class diagram of the class Twython. As we can see, it is a simple class without any relationship like composition, aggregation or inheritance.

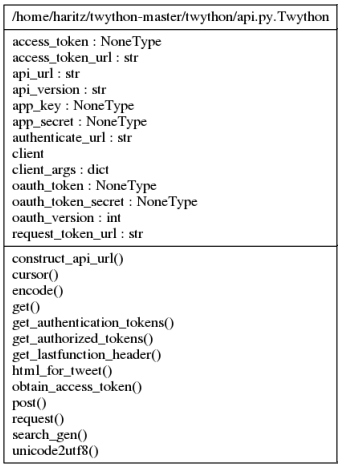

Figure 3.1: Class Diagram of the class Twython

Figure 3.2 shows the inheritance relationship in the exception classes defined by Twython. TwythonError is a general error class which inherits from exceptions.Exception. It also has three children classes for each of the possible errors that can happen using this library: TwythonAuthError, TwythonStreamError and TwythonRateLimitError. In addition, it has defined a warning, TwythonDepecrationWarning, which is a exceptions.DeprecationWarning. This last one is a exceptions.Warning which inherits from exceptions.Exception. TwythonDepecrationWarning is used to warn about using a deprecated feature.

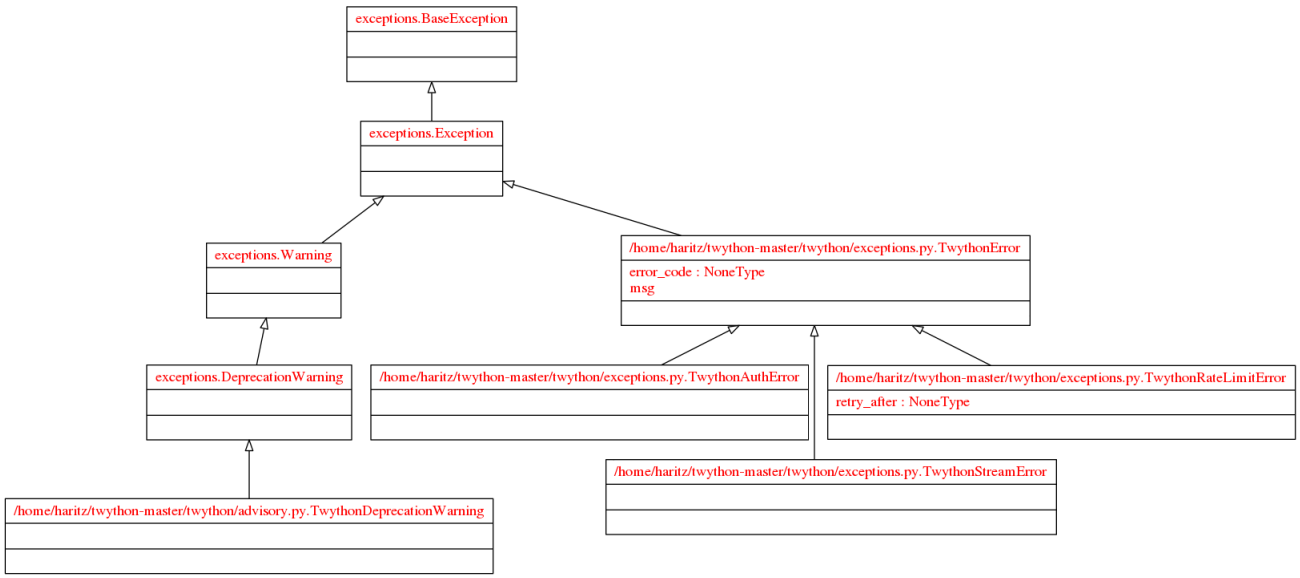

Figure 3.2: Class Diagram of the classes of exceptions of Twython

3.1.2 REST API

The REST API provides access to read and write data in Twitter. For example, it is possible to create new Tweets, read user profiles, etc.

This API has a limit of queries per time-window which is 15 minutes length. Depending on the query, the number of possible queries is different [17]. A few commands are shown next:

The function get_user_timeline [18] retrieves tweets of a given user. This function can only return up to 3,200 of a user's most recent Tweets. Its parameters, among others, are:

- screen_name: The screen name of the user for whom to return results for.
- User_id: The ID of the user for whom to return results for.
- count: Specifies the number of Tweets to try and retrieve, up to a maximum of 200 per distinct request.
- max_id: Returns results with an ID less than (that is, older than) or equal to the specified ID.
- since_id: Returns results with an ID greater than (that is, more recent than) the specified ID.

An example of this function is:

Example 3.1: get_user_timeline function

```
user_timeline=twitter.get_user_timeline(screen_name="MercedesAMGF1"
    , count=10) #get a list of dictionaries. A Tweet is inside a
    dictionary
for tweet in user_timeline:
    # prints Tweets and other information
    print tweet['text'] + " " + tweet['created_at'] + " " + str(
        tweet['retweet_count']) + " " + tweet['lang'] +  " " + tweet
        ['id_str']
```

The function search [19] returns a collection of relevant Tweets matching a specified query. In addition to the parameters which get_user_timeline has, it has two new ones:

- q: The query
- result_type: It is optional. It specifies what type of search results you would prefer to receive. It can have three possible values:
  - recent: Returns only the most recent results in the response
  - popular: Returns only the most popular results in the response.

– mixed: Includes both popular and real time results in the response.

Example 3.2: search function

```
tweets_searched = twitter.search(q='Canada', result_type='recent',
    count=10)
for tweet in tweets_searched['statuses']:
    print tweet['text'] + " " + tweet['created_at'] + " " + str(
        tweet['retweet_count']) + " " + tweet['lang'] +  " " + tweet
        ['id_str']
```

The function get_followers_list [20] returns a collection of user objects for users following the specified user. Among other parameters, it has:

- user_id: The ID of the user for whom to return results for.
- screen_name: The screen name of the user for whom to return results for.
- count: The number of users to return per page, up to a maximum of 200.

Example 3.3: get_followers_list function

```
followers = twitter.get_followers_list(screen_name = "@
    JustinTrudeau", count=10)
for follower in followers["users"]:
    print follower['screen_name'] + " " + follower['name'] + " " +
        follower['location']
```

Now, the following question arises: "How can I retrieve all followers of a user?" For example, Justin Trudeau, the Prime Minister of Canada, has more than 200 followers but the count parameter has an upper limit of 200. In order to do that, there is another parameter called cursor which lets us paginate the results. So the following code does the same as before:

Example 3.4: example of cursor parameter

```
next_cursor = -1
for i in xrange(2):
    followers = twitter.get_followers_list(screen_name = "@
        JustinTrudeau", count=5, cursor=next_cursor)
    for follower in followers["users"]:
        print follower['screen_name'] + " " + follower['name'] + "
            " + follower['location']
    next_cursor = followers['next_cursor'] #WATCH OUT HERE; it is
        followers, not follower
```

This code retrieves 2 pages of the followers of Justin Trudeau and there are 5 followers on each page. We could add more followers to each page modifying the parameter count.

Thanks to Twython, there will be a better way to do it. Twython has implemented the function cursor to manage the cursor for us, but right now has a bug and it is in the process of being fixed [21].

3.1.3 Streaming API

It allows access to Twitter's global stream of Tweet data [22]. It should be used when real-time data from Twitter is needed. This API does not have a rate limit as the REST API. In order to use this API we need to create the following class:

Example 3.5: Class MyStreamer

```
from twython import TwythonStreamer
class MyStreamer(TwythonStreamer):
    def on_success(self, data):
        if 'text' in data:
            print data['text'].encode('utf-8')

    def on_error(self, status_code, data):
        print status_code, data

        # Want to stop trying to get data because of the error?
        # Uncomment the next line!
        # self.disconnect()
```

Which can be instantiated it as follows:

Example 3.6: Get tweets in real time

```
stream = MyStreamer(APP_KEY, APP_SECRET,
                    OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
SearchTerm = 'Trudeau' # If spaces are included, they are 'OR', ie
    finds tweets with any one of the words, not the whole string.
Tweeter = '25073877' # This is Donald Trump, finds tweets from him
    or mentioning him
stream.statuses.filter(track=SearchTerm)
#Or
#stream.statuses.filter(follow=Tweeter)
```

3.2 Data Retrieval

LDA needs documents as input, so we have downloaded Tweets and grouped them in documents. First, Tweets have been stored in a database. This allows us to create different documents with the same Tweets. The database has the following schema:

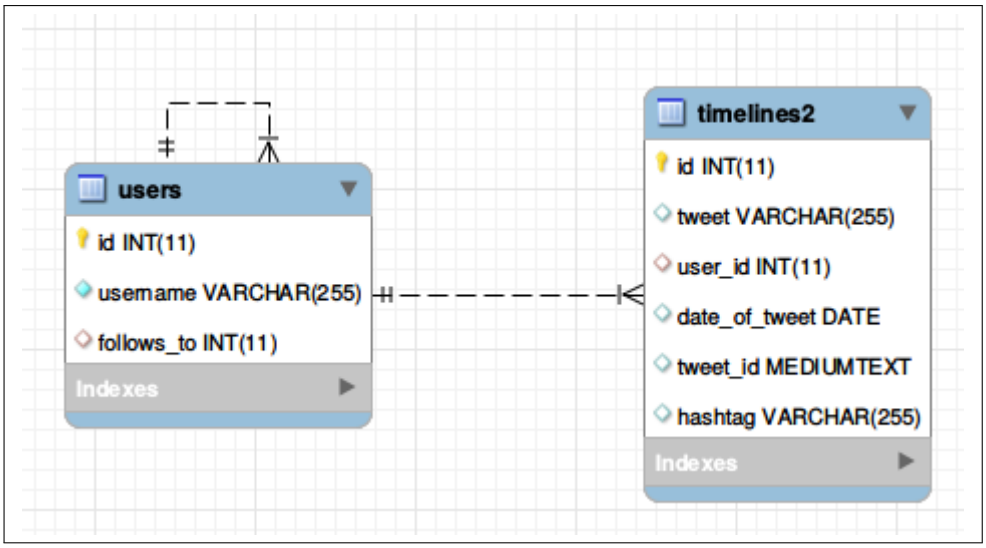

Figure 3.3: Schema of the database

The table users have a one-to-many self-relationship which models that one user has many followers. In addition, it has a one-to-many relationship with the table timelines2. Each row of this table represents a Tweet. It has an id column which is filled automatically by MySQL when it is stored and a tweet_id column in which it is stored the id that Twitter gives to that Tweet.

Before storing the Tweets, a small preprocessing has been done. The hashtags have been retrieved and stored, the words "RT", which stands for retweet, have been removed; idem with the usernames when somebody was quoted, all non-ASCII characters like "ñ" or "í" and the hashtags.

In order to create the files, i.e.: the documents, we have differentiated between followed users (famous people in our case) and followers. In the first case, Tweets have been grouped by hashtag, so a document is a set of Tweets with the same hashtag. All Tweets without hashtag have been grouped in the same document and then this document has been divided into chunks of 10 KiB. In the case of followers of a user, a document has been created per user. This last approach has also been followed in [8].

The following code shows how to create the documents from The New York Times' Tweets. It is trivial to modify it in order to create all the documents of all the users in the database.

Example 3.7: Create documents of The New York Times

```
import MySQLdb


conn = MySQLdb.connect(host= "localhost",
                       user="haritz",
```

```
                        passwd="haritz",
                        db="NLP_TFG")
cursor = conn.cursor()

concat = "SET group_concat_max_len = 18446744073709551615;"
cursor.execute(concat)


selectTweets = """SELECT GROUP_CONCAT(tweet SEPARATOR ' '), hashtag
FROM NLP_TFG.timelines2 t
WHERE t.user_id = 3122 and hashtag != ''
GROUP BY hashtag, user_id;
"""

cursor = conn.cursor()
cursor.execute(selectTweets)
# Fetch all the rows in a list of lists.
results = cursor.fetchall()
cnt = 1
for tweet in results: #for each doc
        nameOfDoc = tweet[1]
        doc = open('/home/haritz/TFG/nytimes_hashtags2/' +
            nameOfDoc+ '.txt', 'a') #appending
        doc.write(tweet[0])
        doc.write(" ")
        doc.close()


# tweets without hashtag

selectTweetsWithoutHashtag = """SELECT tweet
        FROM NLP_TFG.timelines2 t
        WHERE t.user_id = 3122 and hashtag = '';"""

noHashtagCursor = conn.cursor()
noHashtagCursor.execute(selectTweetsWithoutHashtag)
# Fetch all the rows in a list of lists.
results = noHashtagCursor.fetchall()
cnt = 1
for tweet in results: #for each doc
        nameOfDoc = str(cnt)
        cnt += 1
        doc = open('/home/haritz/TFG/nytimes_hashtags2/' +
            nameOfDoc + '.txt', 'a') #appending
        doc.write(tweet[0])
        doc.write(" ")
        doc.close()
```

3.3 Modeling users' timelines with LDA

Gensim's [23] LDA algorithm implements Online LDA [24] in Python. It needs three parameters:

- Corpus: A vector of (word id, number of occurrences)
- Number of topics
- Dictionary: a mapping between words id and words

Example 3.8: Example LDAModel

```
gensim.models.ldamodel.LdaModel(corpus, num_topics, dictionary)
```

Figure 3.4 shows the class diagram of the class LDAModel. It is worth noting the composition between LDAModel and LDAState. An LDAModel has one LDAState.

3.3.1 How to create a corpus?

First, a list of documents is needed. It is possible to obtain it with the following code:

Example 3.9: read_data

```
def read_data(path):
    docs = []
    for file_name in os.listdir(path):
        file = open(path + file_name)
        docs.append(unicode(file.read(), errors='replace'))
    return docs
```

Data Cleaning

Data cleaning is the most important step in natural language processing. In the case of Twitter, this task is even more difficult due to the length of the Tweets and the topic diversity. First of all, all words have been converted to lowercase. Then, each document has been split into words. After that, all numbers have been removed, but words with numbers have been kept. Next, words with three characters or less have been removed because short words like "to", "I", "me", etc. do not help the algorithm. Next, stopwords have been removed using the nltk stopwords list. After that, words have been lemmatized. It is a transformation of the word to convert it into the word's lemma, or dictionary form. For instance: "are" is transformed into "be" and "cats" into "cat". Furthermore, bigrams have also been taken into account. There are words like New York or San Francisco which are

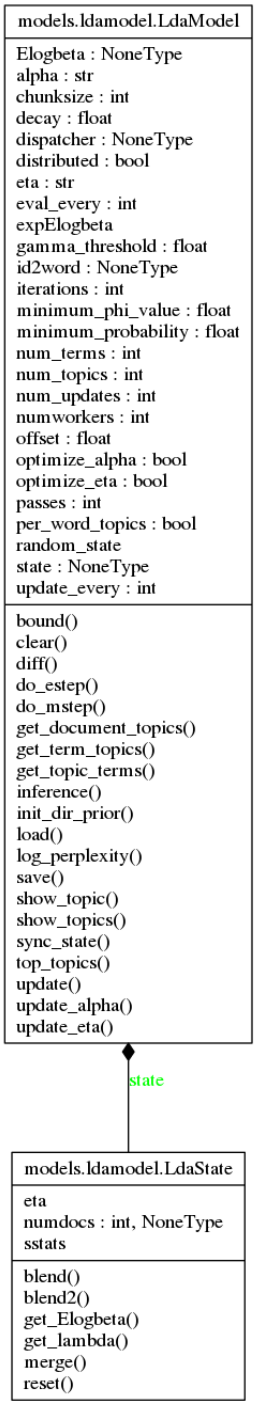

Figure 3.4: Class Diagram of LDAModel

always together. Using bigrams we can detect them and treat them as only one. In the code below, we find bigrams and then, add them to the original data, because we would like to keep the words "machine" and "learning" as well as the bigram "machine_learning". Computing n-grams of a large dataset can be computationally and memory expensive.

This is the code that used for preprocessing the data.

Example 3.10: preprocessing

```
def preprocessing(docs):
    # Split the documents into tokens.
    tokenizer = RegexpTokenizer(r'\w+')
    stops = set(stopwords.words('english'))  # nltk stopwords list
    for idx in range(len(docs)):
        docs[idx] = docs[idx].lower() # Convert to lowercase.
        docs[idx] = tokenizer.tokenize(docs[idx])  # Split into
            words.

    # Remove numbers, but not words that contain numbers.
    docs = [[token for token in doc if not token.isnumeric()] for
        doc in docs]

    # Remove words that <= three character.
    docs = [[token for token in doc if len(token) > 3] for doc in
        docs]

    docs = [[token for token in doc if token not in stops] for doc
        in docs]

    # Lemmatize all words in documents.
    lemmatizer = WordNetLemmatizer()
    docs = [[lemmatizer.lemmatize(token) for token in doc] for doc
        in docs]

    # Add bigrams to docs (only ones that appear 20 times or more).
    bigram = gensim.models.Phrases(docs, min_count=20)
    for idx in range(len(docs)):
        for token in bigram[docs[idx]]:
            if '_' in token:
                # Token is a bigram, add to document.
                docs[idx].append(token)

    return docs
```

Dictionary and corpus

Once we have the data preprocessed, creating the dictionary and the corpus is straightforward thanks to Gensim.

Example 3.11: get_dictionary

```
def get_dictionary(texts):
    # turn our tokenized documents into a id <-> term dictionary
    dictionary = corpora.Dictionary(texts)
    #dictionary.filter_extremes(no_below=20, no_above=0.5)
    dictionary.filter_n_most_frequent(5)
    return dictionary
```

Example 3.12: get_corpus

```
def get_corpus(dictionary, texts):
    # convert tokenized documents into a document-term matrix
    users_corpora = [dictionary.doc2bow(text) for text in texts]
    return users_corpora
```

Figure 3.5 shows the class Dictionary, which is a subclass of Mapping. This last class is a subclass of Container, Iterable and Sized. This is possible because Python supports multiple inheritance in contrast to Java. The module _abcoll defines abstract base classes for collections.

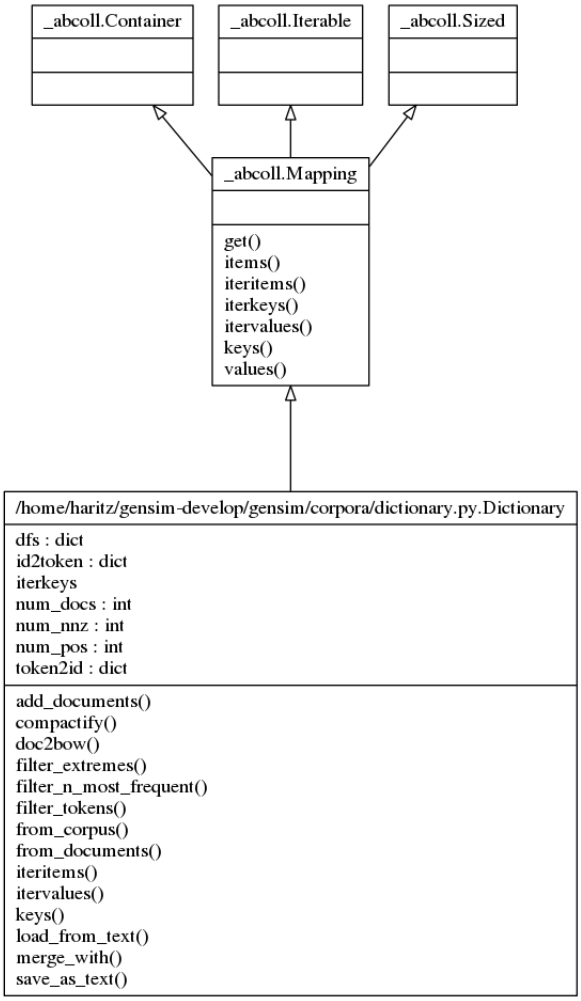

Figure 3.5: Class Diagram of Dictionary

3.3.2 LDA Model

The function LdaModel has three mandatory parameters as we stated before: the corpus, the number of topics and the dictionary. So creating an LDA model is easy

since we have already created the auxiliary functions explained before.

Example 3.13: LdaModel

```
docs = read_data(path)
texts = preprocessing(docs)
dictionary = get_dictionary(texts)
users_corpora = get_corpus(dictionary, texts)
gensim.models.ldamodel.LdaModel(users_corpora, num_topics=
    num_topics, id2word = dictionary)
```

In addition to the three main parameters mentioned before, LdaModel has several optional parameters. In this thesis, we have used the following:

- alpha: hyperparameter that affects sparsity of the document-topic distribution. The lower it is, the lower the number of topics per documents.
- eta: hyperparameter that affects sparsity of the topic-word distribution. The lower it is, the lower the number of words per topic.
- passes: puts a limit on how many times LDA will execute the E-Step for each document, meaning that some documents may not converge in time.
- iterations: allows LDA to see the corpus multiple times. It is very handy for small corpora.

3.3.3 Data Visualization

Once models are created, we need to interpret them. It is possible to obtain the distributions of the models executing the following code:

Example 3.14: Retrieving topic distributions

```
for i in xrange(number_of_topics): #for each topic
    print 'Topic ' + str(i)
    for tup in small_ldamodel.get_topic_terms(i, topn=len(
        dictionary)):
        print dictionary.get(tup[0]) + " " + str(tup[1]) #(word,
            probability)
```

Nevertheless, it is difficult to obtain a quick global idea of the topics. Data Visualization has a huge importance in the interpretation step of data mining. That is why there are so many data visualization libraries. LDAvis [25] is a web-based interactive visualization of LDA topics built using R and D3 for Python and R. The objective of LDAvis is to answer the following questions:

1. What is the meaning of each topic?

2. How prevalent is each topic?

3. How do the topics relate to each other?

LDAvis is composed of two parts:

- Left panel: Gives a global view of the topic model. It answers questions 2 and 3. The area of each circle means the topic's overall prevalence and the location of each circle is determined by computing the distance between topics.
- Right panel: Answers question 1. It is composed of horizontal bars. Each bar represents a term for the selected topic. The bars in blue represents the overall term frequency while the red ones the estimated term frequency within the selected topic. In addition, there is a λ parameter which is a weight for the relevance of a term to a topic.

LDAvis ranks the terms within a topic using its relevance.

$$relevance = r(w, k|\lambda) = \lambda log(\phi_{kw}) + (1 - \lambda) log(\frac{\phi_{kw}}{p_w})$$

where

$$log(\phi_{kw}) = P(word = w|topic = k)$$

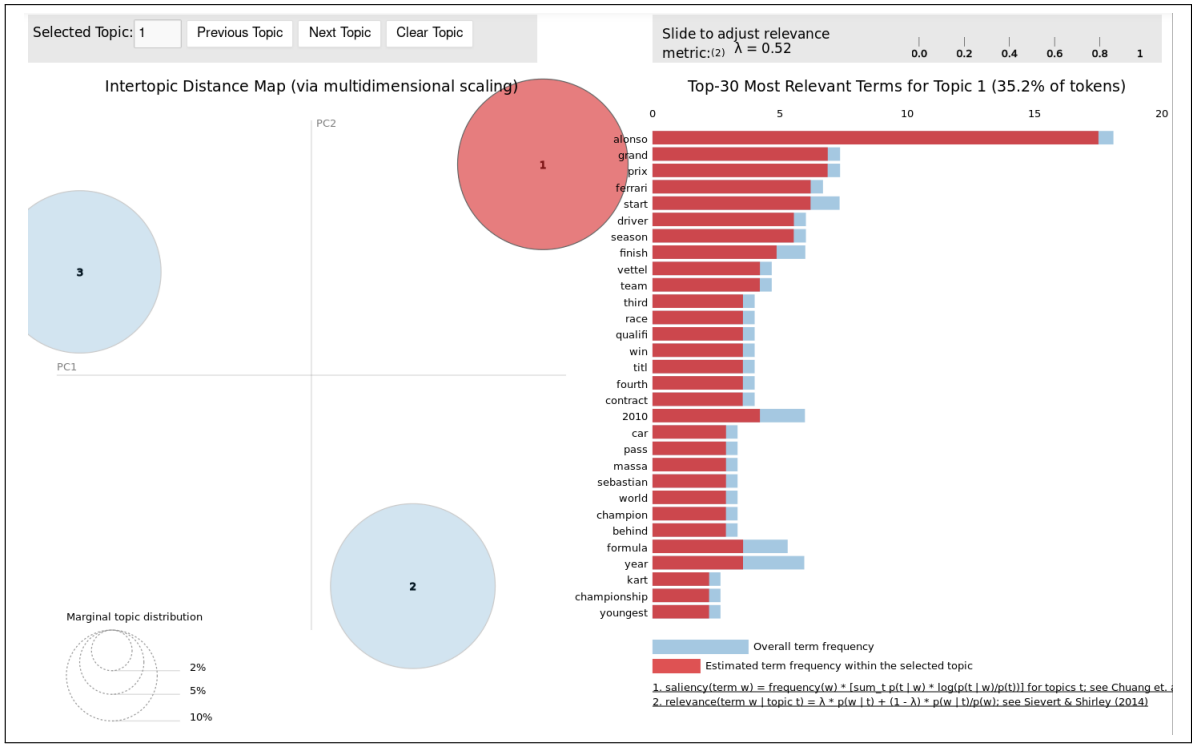

Figure 3.6: Example of a graphical model

The lambda parameter allows us to modify how we want to compute the relevance. If $\lambda = 1$, then:

$$relevance = r(w, k|\lambda = 1) = log(\phi_{kw})$$

In this case, the shown words would be relative to the overall corpus but not necessarily to the topic. On the other hand, if If $\lambda = 0$, then:

$$relevance = r(w, k|\lambda = 0) = log(\frac{\phi_{kw}}{p_w})$$

and this implies that words which appear a lot in the corpus do not have a high relevance while words that appear a few times times in the corpus would have a high relevance.

3.3.4 Use of Gensim's LDA

First, LDA has been executed on a small corpus created with articles from Wikipedia. This corpus is composed of three documents:

- One about Fernando Alonso, a Spanish F1 pilot.
- One about Barack Obama, President of the United States of America.
- One about Kurt Gödel, one of the most important logicians in the history.

After the execution of the following code:

Example 3.15: LdaModel

```
(model, corpus, texts) = get_LDA_model(path_to_test_documents1, 3)
lda_Data =  pyLDAvis.gensim.prepare(model, corpus, get_dictionary(
    texts))
pyLDAvis.display(lda_Data)
```

A model with three topics have been obtained. Using LDAvis to interpret the topics, it is possible to clearly identify three separated topics: the first one about Fernando Alonso, the second one about Barack Obama and the last one about Kurt Gödel.

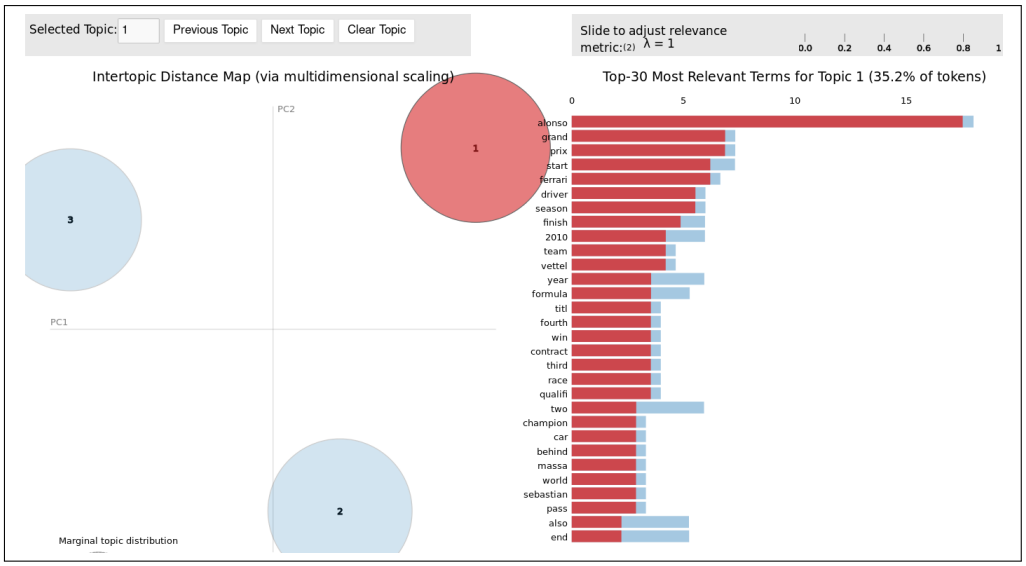

Figure 3.7: Topic 1: Alonso

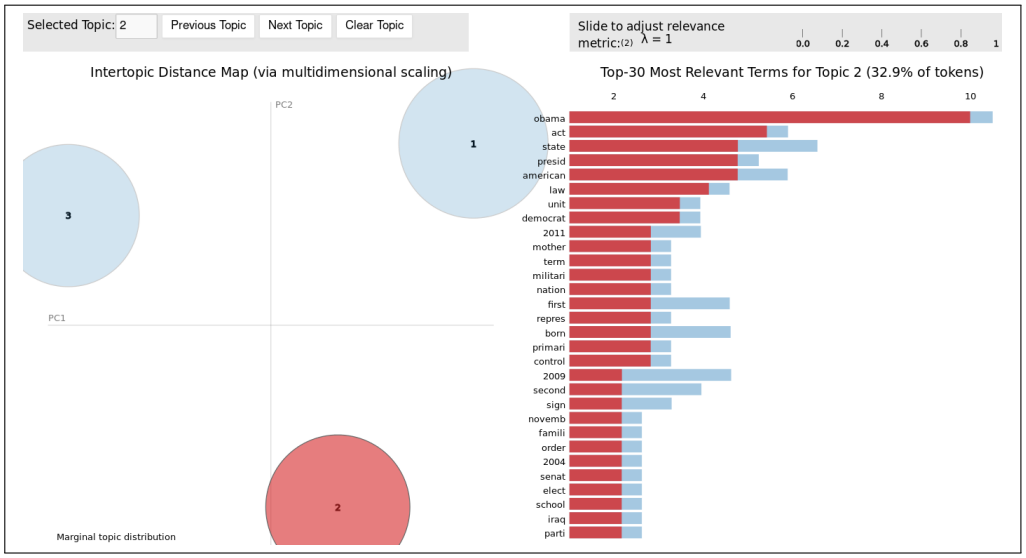

Figure 3.8: Topic 2: Obama

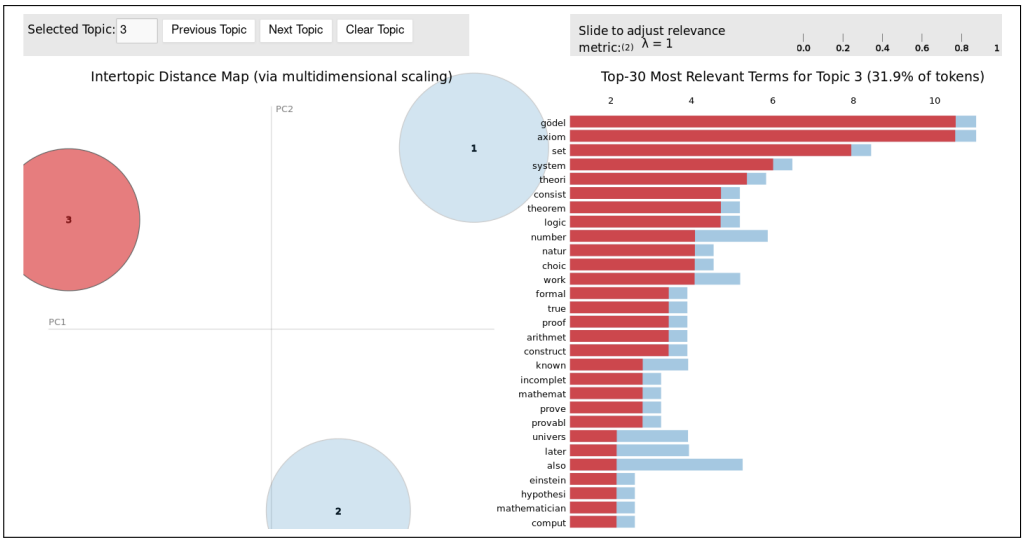

Figure 3.9: Topic 3:Kurt Gödel

Another small test has been done. In this case, the following documents have been used:

- An article in Wikipedia about Xiaomi Inc., a Chinese mobile phone manufacturer.
- An article in Wikipedia about the United States of America.
- An article about Alan Turing, the father of the Theoretical Computer Science and Artificial Intelligence.

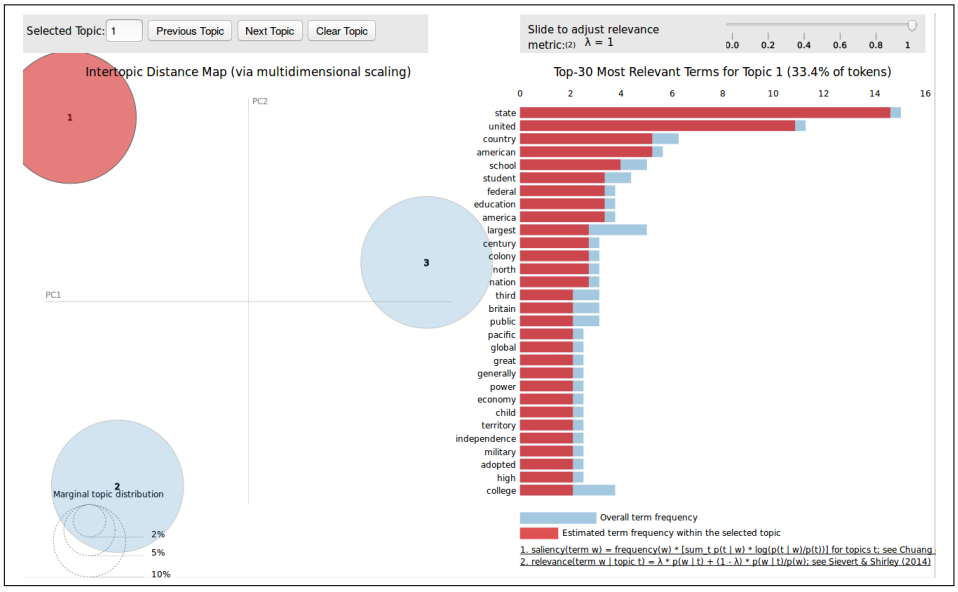

Figure 3.10: Topic 1: USA

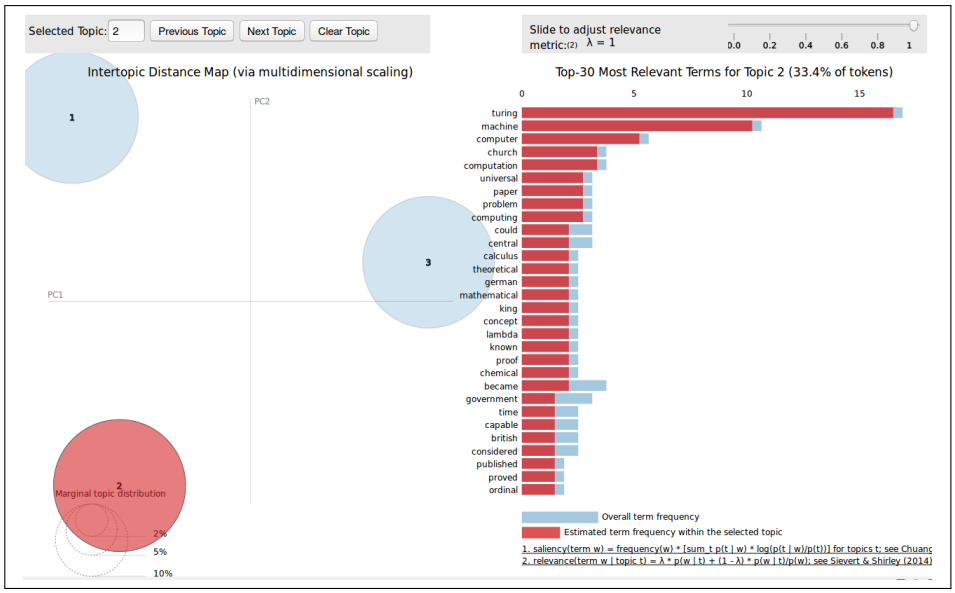

Figure 3.11: Topic 2: Turing

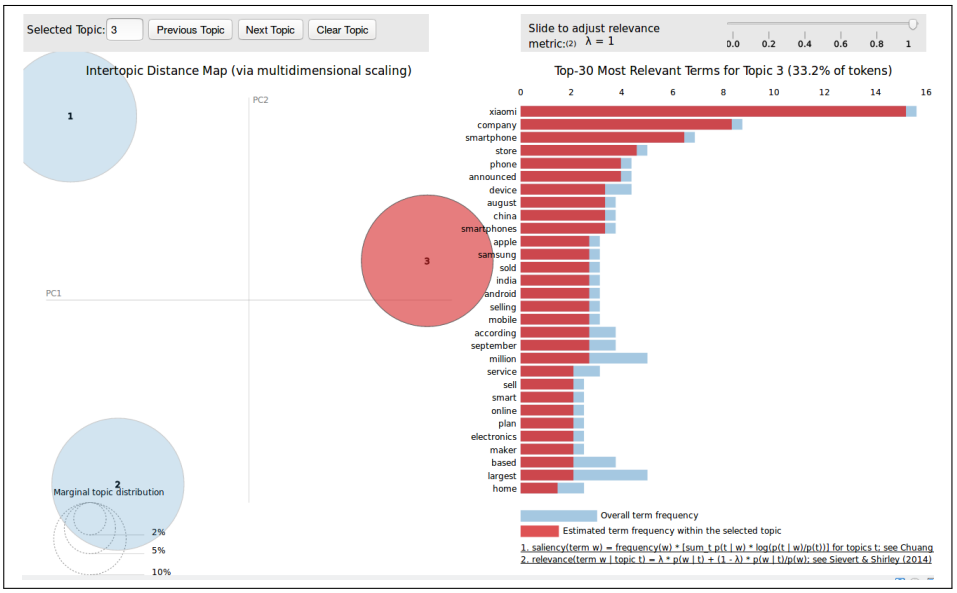

Figure 3.12: Topic 3: Xiaomi

The creation of a model is a random process, so in each execution of the algorithm, it gives different results. Figure 3.13, 3.14 and 3.15 represent a bad model. The dataset is exactly the same one as before: one document about Xiaomi, another one about Turing and the last one about the USA.

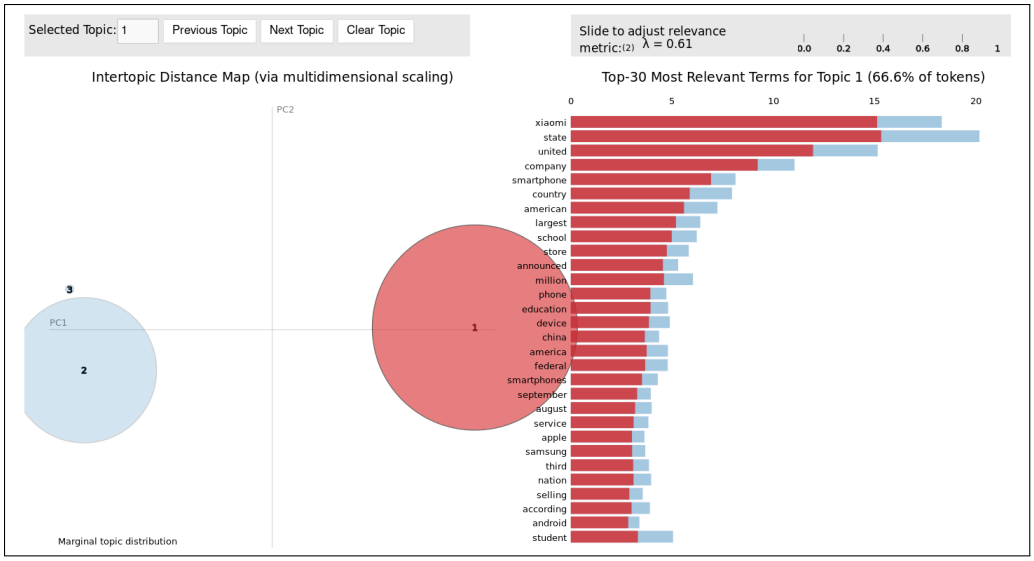

Figure 3.13: Bad model - Topic 1. It is a mixture of Xiaomi and United States.

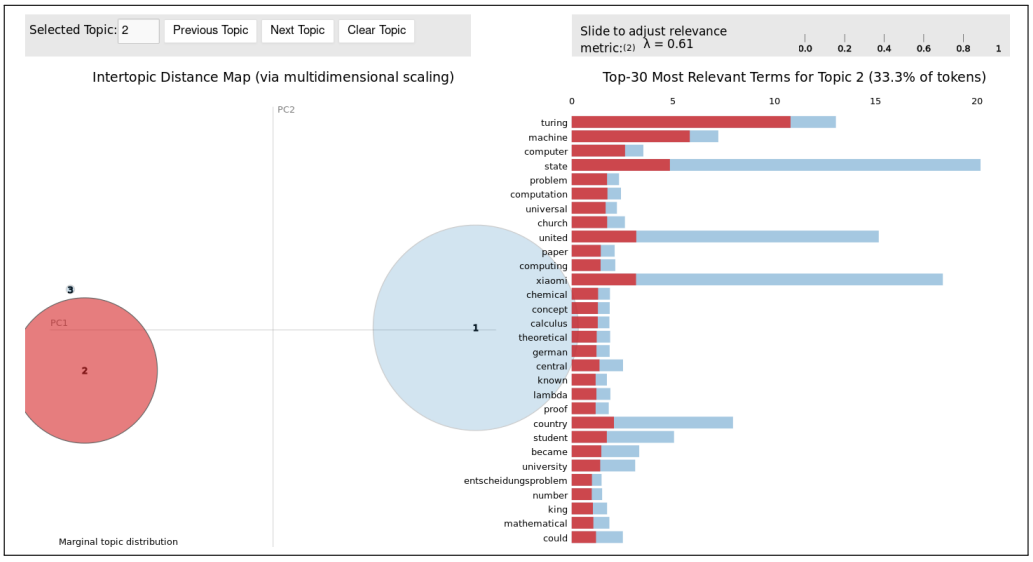

Figure 3.14: Bad model - Topic 2. It is about Turing but it also has the word Xiaomi.

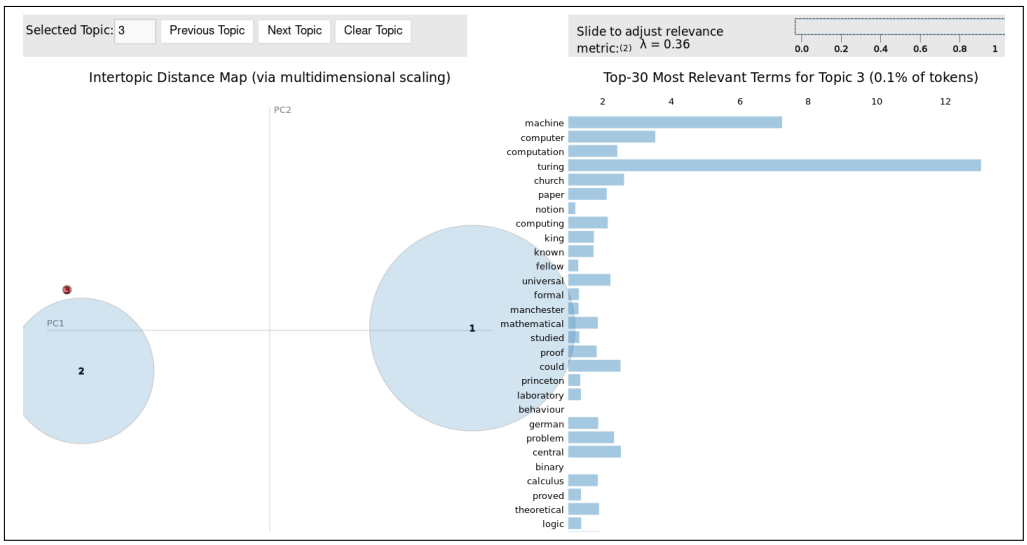

Figure 3.15: Bad model - Topic 3. It is a really small topic which does not make any sense.

Chapter 4

Results

In this chapter, the focus will be in the generation of LDA models, its analysis and the comparison of the topics obtained from a user and their followers. An analysis of the results including if the results given are valid and accurate will be provided. All the data has been retrieved between April 28, 2017 and May 3, 2017 taking into account that Twitter only allows to download up to 3,200 of a user's most recent Tweets.

4.1 Barack Obama's Topics

LDA has been able to recognize several topics among the Tweets of Barack Obama. The second topic, shown in figure 4.1, is clearly about climate change. It contains words like "climate", "climate_change", "clean", "energy", "fight", "carbon", "pollution", etc. We can see that most of the words of this topic are only relevant in it because the red bar is almost as big as the blue one.

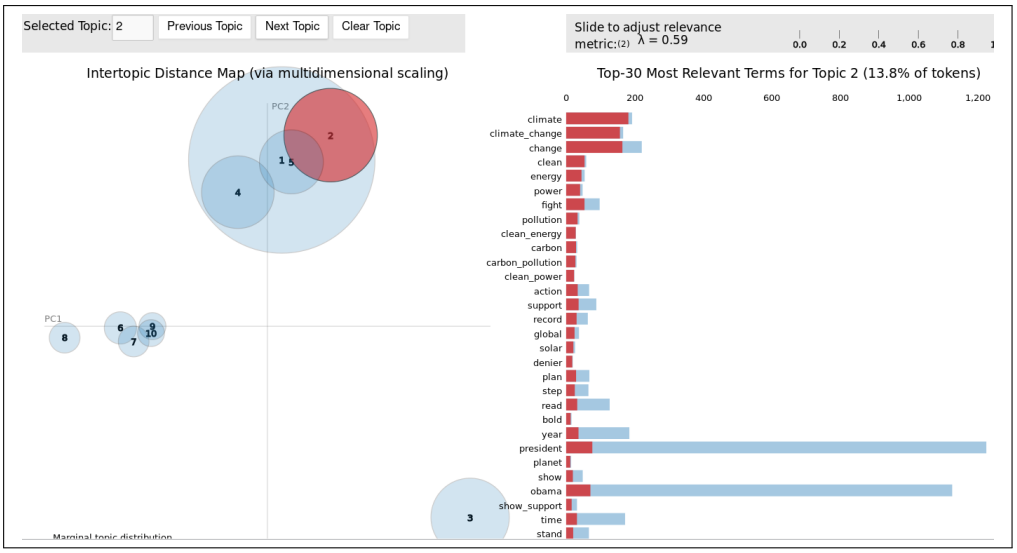

Figure 4.1: Topic 2: Climate Change

The third topic, shown in figure 4.2, is about the Supreme Court nomination Merrick Garland. Barack Obama nominated him for the Supreme Court but the Republican senate was against it so the judge Merrick Garland could not have a sit at the Supreme Court. The words that allow us to label this topic this way are: "senate", "judge", "garland", "supreme_court", "nominee", "obstruction", etc. We can see again that most of the words of this topic are only relevant in it for the same reason as before.

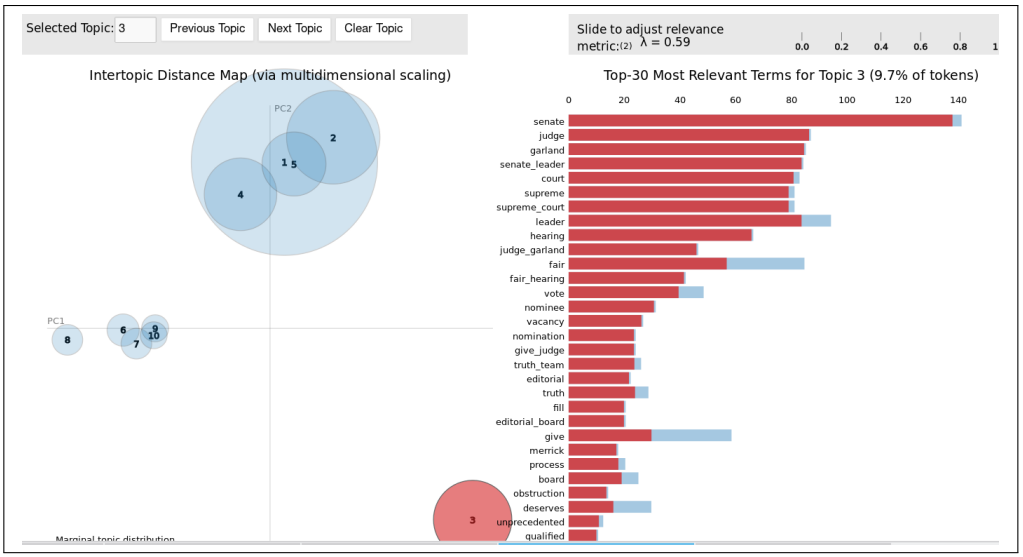

Figure 4.2: Topic 3: Merrick Garland Supreme Court nomination

The fourth topic, shown in figure 4.3, seems to be about the Patient Protection and Affordable Care Act commonly known as Obamacare. The words of this topic are about health and insurance for example: "healh_care", "uninsured", "sick", "coverage", "affordable", "million_american", etc. In this topic, we can see that

there are many words which are relevant in other topics because the red bars are not as big as the blue ones.

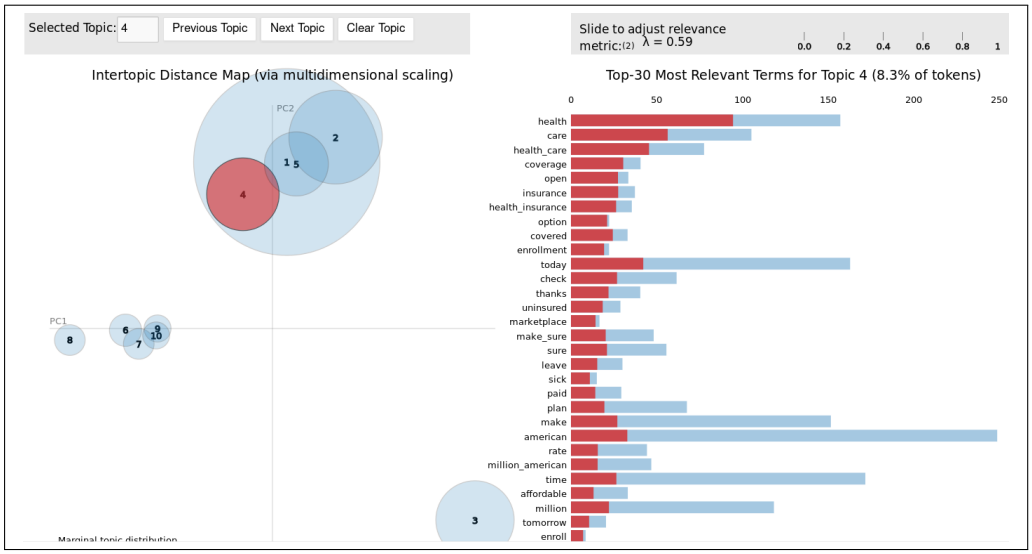

Figure 4.3: Topic 4: Health Care

With the fifth topic, shown in figure 4.4, labeling starts to be more difficult. Words like "immigration", "system", "immigrant", "congress" and "violence" may indicate that this topic is about immigration. It is worthwhile noting that the word violence appears but LDA does not tell us its meaning. It could have been used to indicate that immigrants are not violent or the contrary. It is also important to note that the relevance of this topic in the corpus is really small. We can see this because the bars are usually very small.

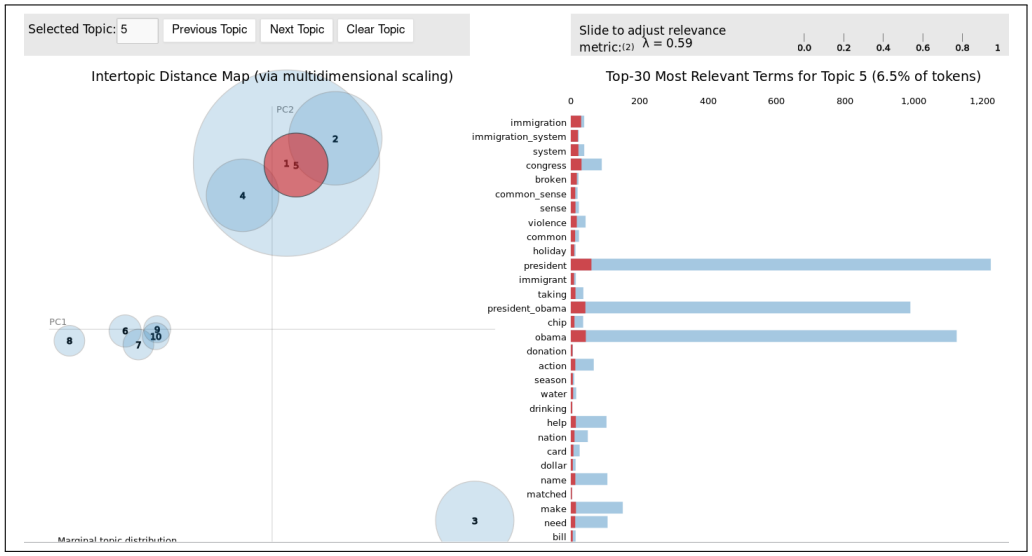

Figure 4.4: Topic 5: Immigration

The last topic which is possible to label, shown in figure 4.5, is the number six.

It contains words like "minimum", "wage", "raising", "earning", "poverty" and "private_sector" so it can be labeled as "Salaries". This topic is not too relevant in the corpus since the size of the cluster is small in comparison with other clusters.

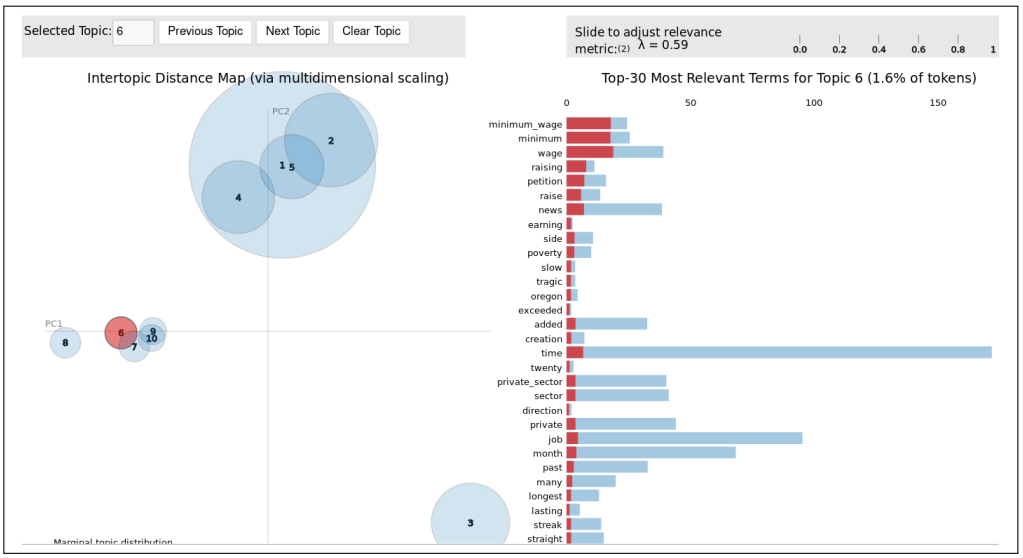

Figure 4.5: Topic 6: Salaries

Once the topics are analyzed it is possible to state that Barack Obama writes on Twitter about his politics.

4.2 Barack Obama's Followers' Topics

Unfortunately, Standard LDA has not been able to obtain any topics with some sense for humans. All of them are a mixture of unrelated words or without a useful meaning. For example, in topic 7, shown in figure 4.6, there are words like "good", "time", "people", "life", "look", "think", "need", "year", "right", "know", etc. Those words are not useful to label this topic. Most of the topics are similar to this one.

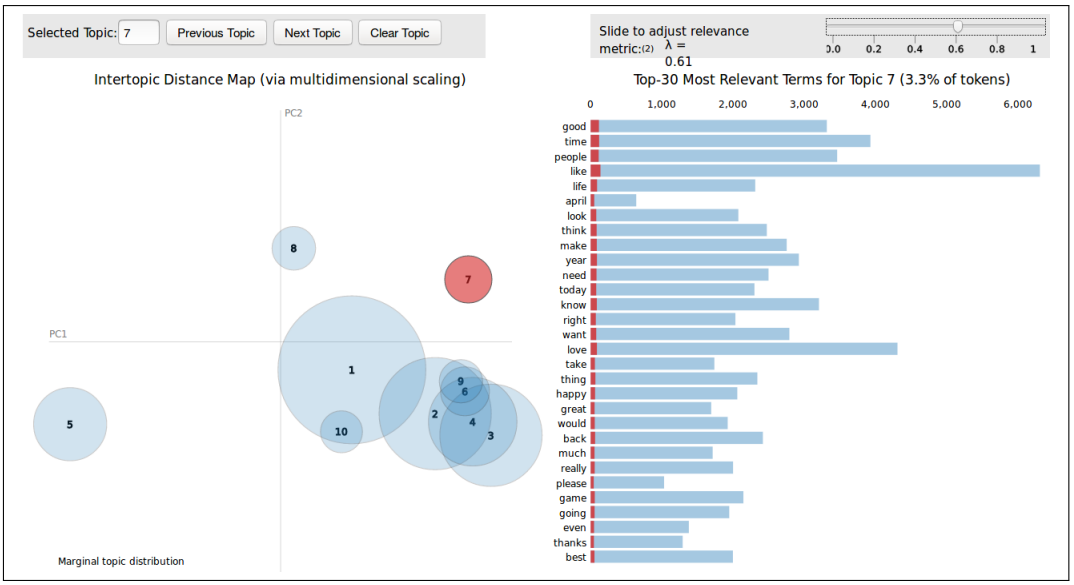

Figure 4.6: Topic 7: It's not possible to label it.

4.3 NASA's Topics

All the LDA topics generated share words like mission, space, etc. Nevertheless, LDA has been able to generate some topics which can be interpreted by humans. The first topic, shown in figure 4.7, contains words like "space", "launch", "spacecraft", "crew", "mission", "moon", "cargo" and "cygnus" so it is possible to conclude that it is about the Cygnus Spacecraft launch. We can see that the blue bars are bigger than the red ones. This means that the words of this topic are relevant in other topics. As stated before, words like space and earth appears in many topics.

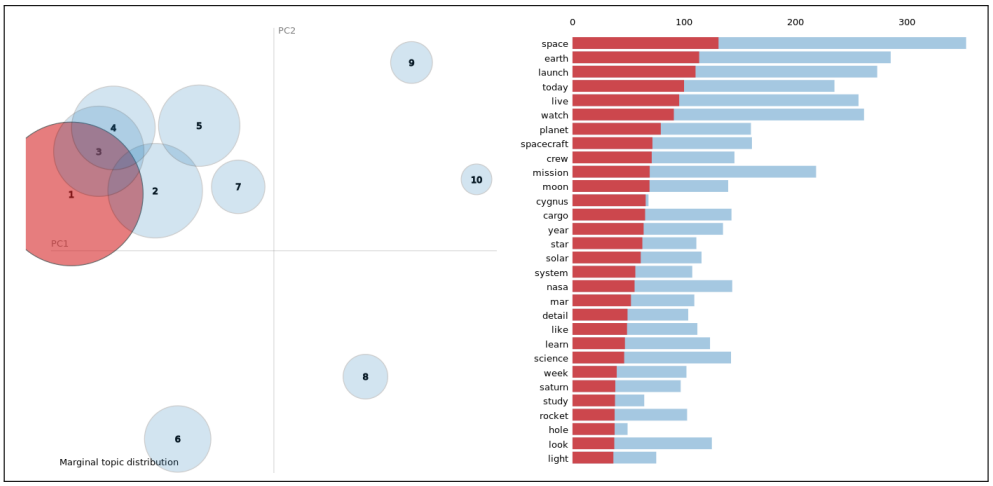

Figure 4.7: Topic 1: Cygnus Spacecraft launch

The fifth topic, shown in figure 4.8 contains words like "spacewalk", "astronaut", "earth", "look", "view" and "outside", so we can label it as "Spacewalk". In this

case, the word "spacewalk" is only relevant in this topic because the red bar is almost as big as the blue one. "outside" is not a word that appears too much in the corpus but it is quite relevant in this topic.

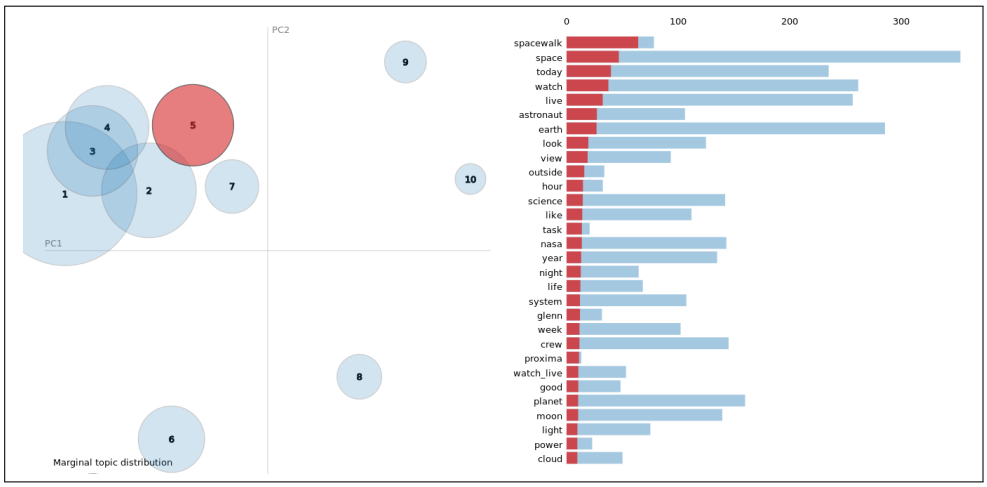

Figure 4.8: Topic 5: Spacewalk

The sixth topic, shown in figure 4.9, contains words like "launch", "mission", "cygnss", "hurricane", "satellite", "weather" and "forecasting" so it is easy to label it "CYGNSS". It is about a system sponsored by NASA to forecast hurricanes. As expected, the word "cygnss" only appears in this topic since the red bar is as big as the blue one. The other words like "launch", "mission" and "satellite" describes this topic. This means that LDA has been able to recognize this topic correctly.

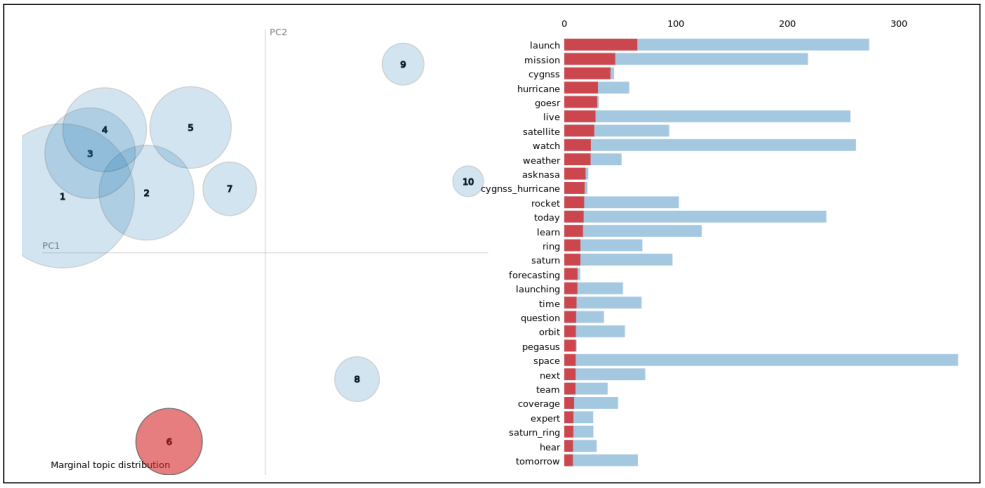

Figure 4.9: Topic 6: CYGNSS hurricane forecasting

The last topic that has been labeled is the eight one, shown in figure 4.10. It contains words like "cargo", "dragon", "vehicle", "spacecraft", "supply", "carrying" and "station" so it has been labeled as "Dragon Cargo Vehicle". In this case, LDA is able to identify correctly again the topic because words like "dragon_cargo" and

"dragon" are only relevant in this topic. We can see this because their red bars are almost as big as their blue ones. Words like "vehicle", "supply" and "spacecraft" helps to understand this topic but also other topics, that is why they are also relevant in others.

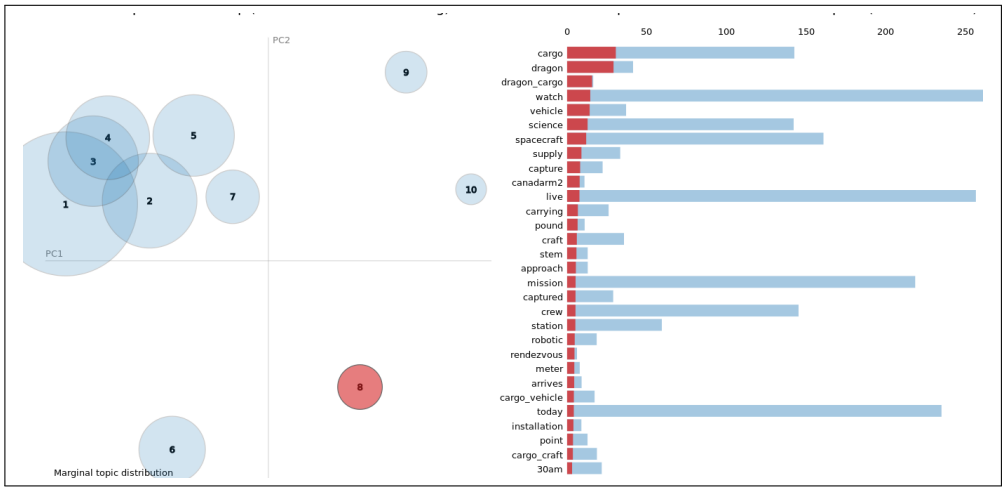

Figure 4.10: Topic 8: Dragon cargo vehicle

After the analysis of all its topics, it is easy to see that as expected, NASA writes on Twitter about the space and its missions.

4.4 NASA's Followers' Topics

After executing LDA in the corpus, four well-defined topics have been found. The first topic, shown in figure 4.11, has been labeled: "Politics" because it includes words like "Trump", "Obama", "President", "democrat", "liberal" and "President_Trump". All these words are only relevant in this topic. This means that LDA has been able to define it well.

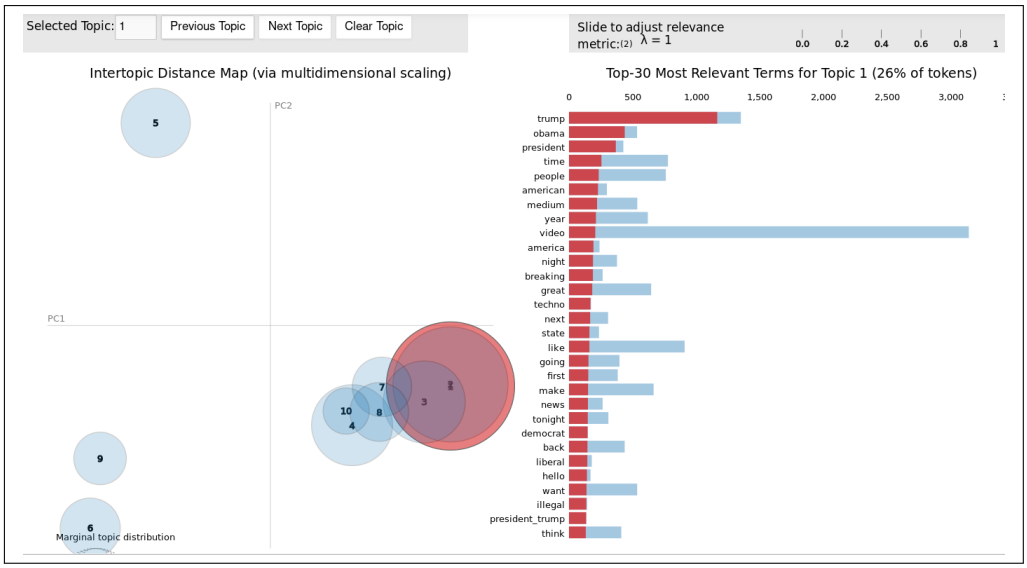

Figure 4.11: Topic 1: Politics

The third topic, shown in 4.12, is clearly about students since it contains words like "student", "learning", "teacher", "school", "kid", "classroom", "educator", etc. All those words are only relevant in this topic which means that in this case, LDA has been able again to define the topic well.

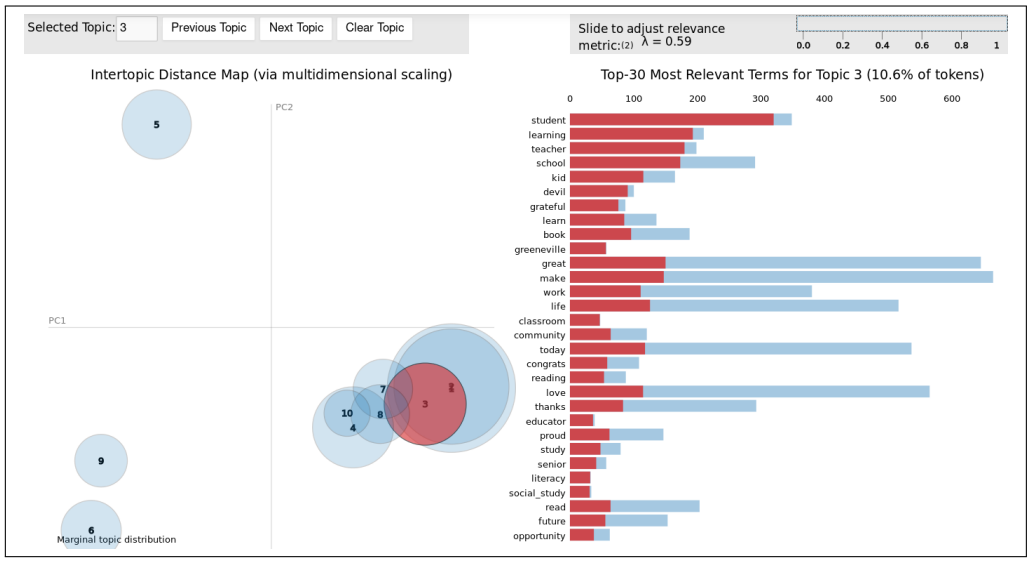

Figure 4.12: Topic 3: Students

The following well-defined topic is the eighth one, shown in figure 4.13. This topic is also easy to label since it contains words like "social", "medium", "business", "marketing", "facebook", "strategy", "business_social", "strategy" and "brand" so it has been named: "Marketing in Social Media". Most of those words are mostly relevant only in this topics. Other words like "Facebook" are relevant in other topics too as expected since it is a word used in many contexts, not only in marketing.

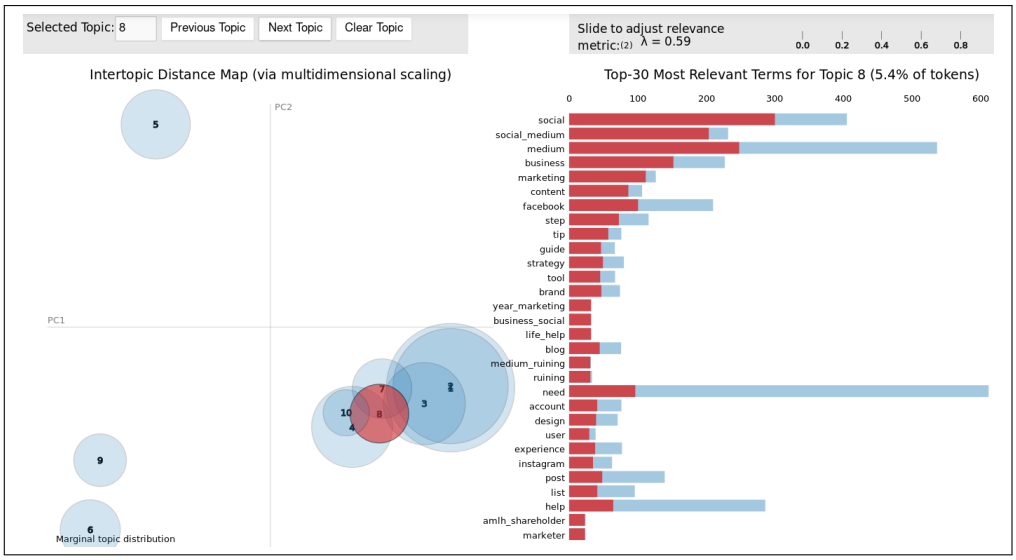

Figure 4.13: Topic 8: Marketing in Social Media

The last labeled topic is the ninth one, shown in figure 4.14. It has been called "Music" since it has words like "bossa", "Ernesto_Nazareth", "Alatamiro_Carrilho", "bossa_nova", "Leny_Andrage" and "lista_reprodu". As we can see, most of the words only belongs to this topic. This means that this topic so specific that any of its words cannot be in used in any other topic.

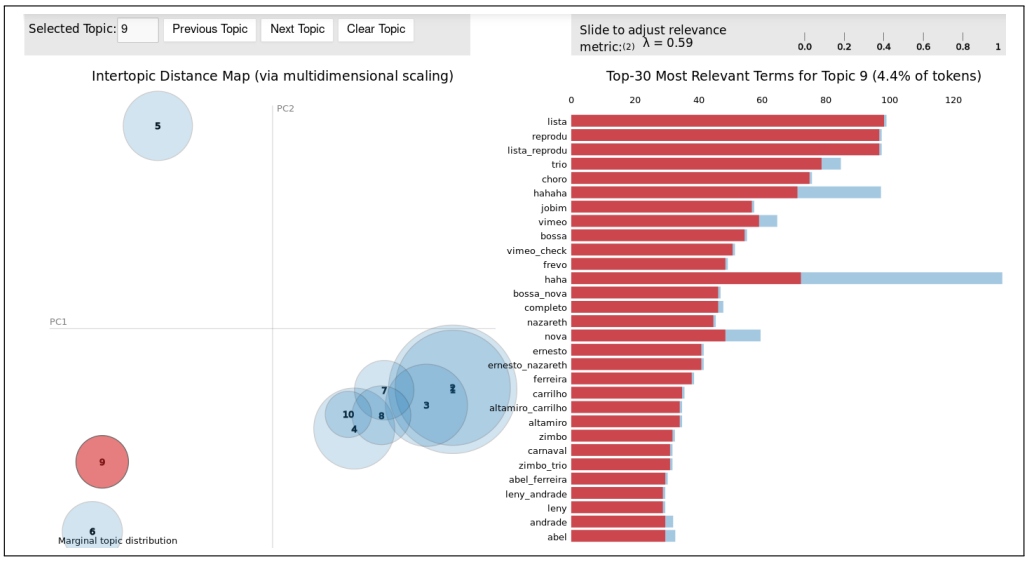

Figure 4.14: Topic 9: Music

After this analysis, it is easy to see that NASA's topics are not related to its followers since any of them wrote about the space or any space mission.

4.5 Lewis Hamilton's Topics

Among the topics LDA has generated, a few topics can be interpreted by humans. The first one, figure 4.15, is about feeling grateful. It contains words like: "thanks", "great", "love", "followed", "friend", "well" and "happy birthday". Probably he writes about this topic when he wins races or the F1 championship. As we can see, most of the words are not only relevant in this topic which means that this topic is not too specific, i.e.: its words can be used in other contexts or topics.

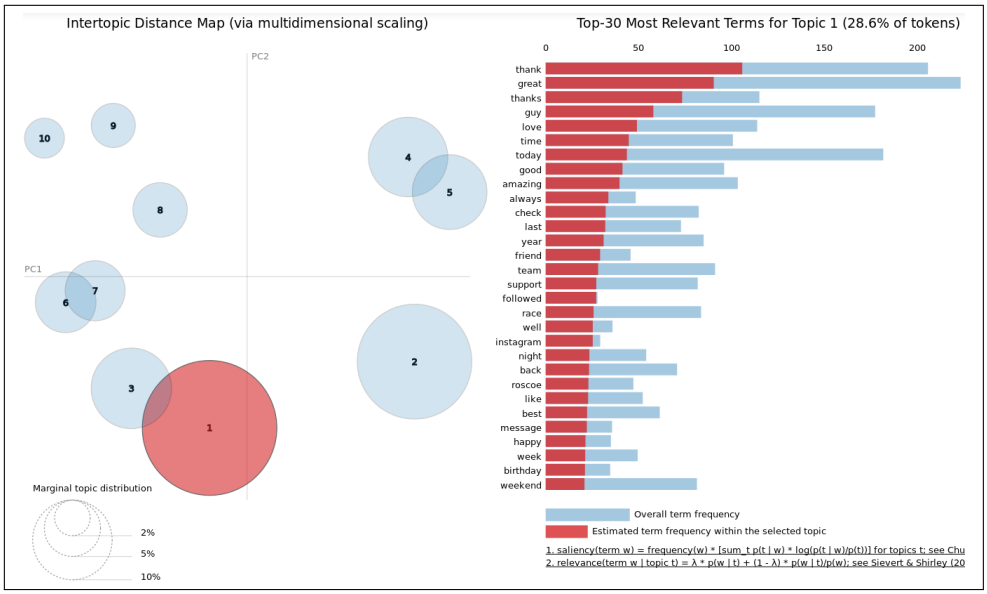

Figure 4.15: Topic 1: Feeling grateful.

The remaining topics are very noisy although it is possible to identify that the fourth, fifth and eight topics are about Grand Prix (GP). The fourth topic contains words like: "Austrian GP", "Russian GP", "Abu Dhabi GP", "Brazilian GP", "Spanish GP", "Sochi" (the Russian city where the Russian GP takes place), "Jerez (a Spanish city where sometimes there are F1 tests) and "Yas Marina" (the name of the circuit of Abu Dhabi).

The other topics are difficult to label because although most of the words are related to Formula 1, they are a mixture of words which are not related to each other.

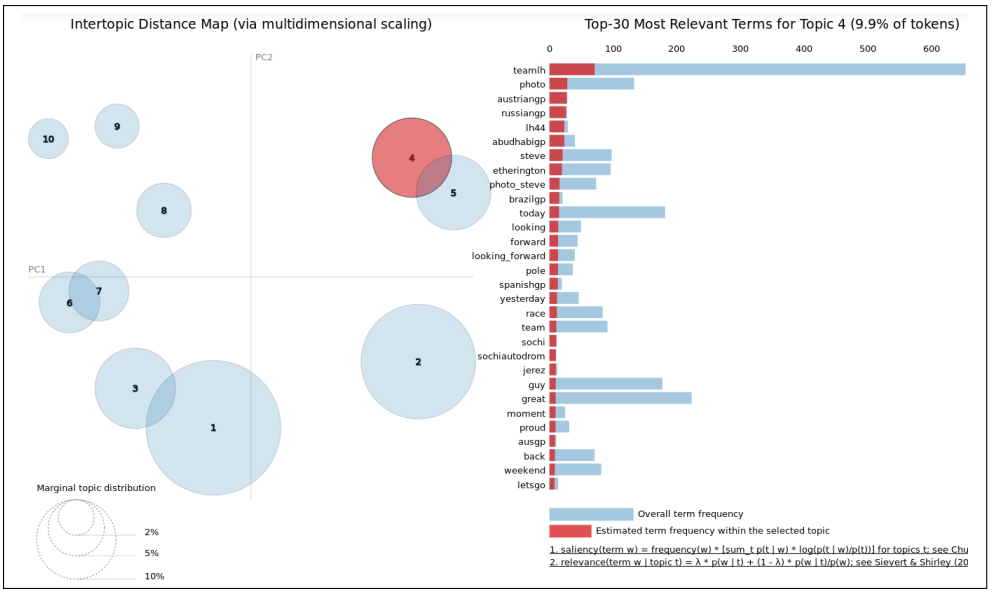

Figure 4.16: Topic 4: Grand Prix. Common words are: Austrian GP, Russian GP, Abu Dhabi GP, Brazilian GP, Spanish GP, Sochi (the Russian city where the Russian GP takes place), Jerez (the Spanish city where the Spanish GP takes place) and Yas Marina (the name of the circuit of Abu Dhabi).

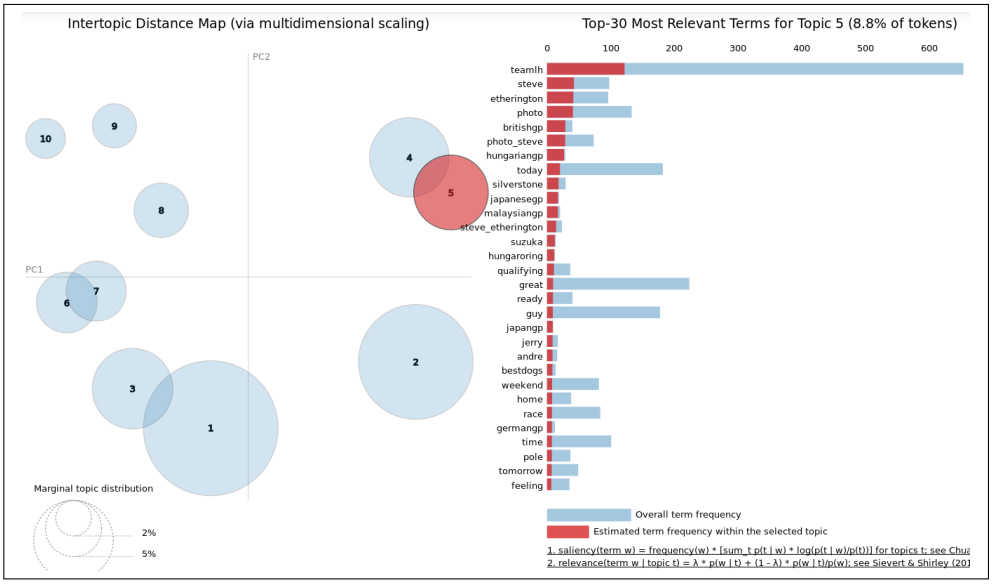

Figure 4.17: Topic 5: Grand Prix. Common words are: Hungarian GP, British GP, Japanese GP, Malaysian GP, European GP, Chinese GP, Silverstone (the circuit of the UK), Suzuka (the Japanese circuit), Hungaroring (the Hungarian circiut) and Hammertime (a word which Hamilton uses a lot to refer his piloting style).

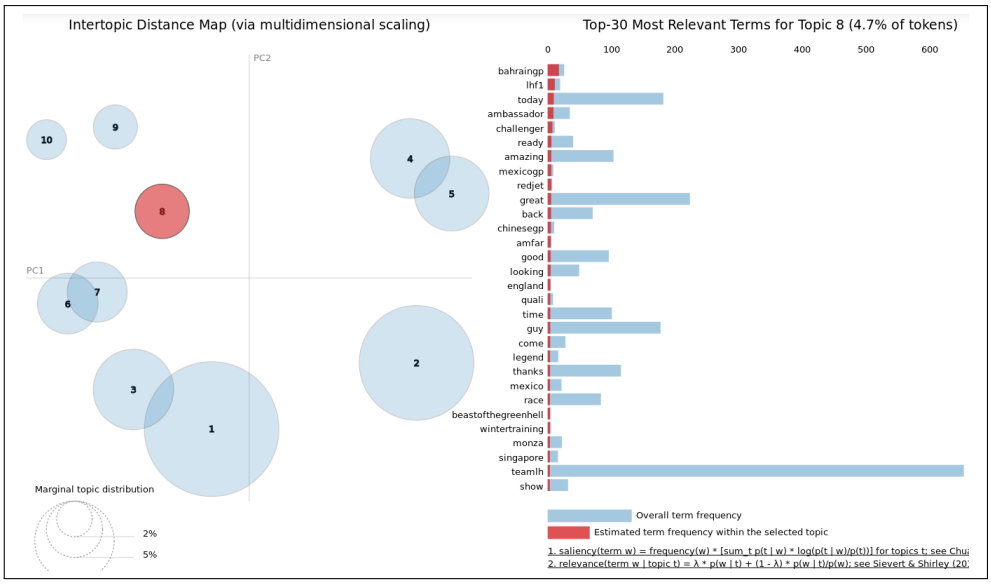

Figure 4.18: Topic 8: Grand Prix. Common words are: Barhain GP, Mexico GP, winter training, Singapore (there is a GP there) and F1isback.

In conclusion, Hamilton writes about F1 as expected.

4.6 Hamilton's Followers' Topics

A total of five topics has been identified. The first topic, figure 4.19, is about flights. It has words like: "airport", "service", "international_airport", "airline", "flight" and cities or countries like "Bali" (a very touristic place), "San Francisco", "Korea", "Melbourne", "York" and "Australia". This topic can be considered well defined and specific because most of its words are only relevant in this topic.

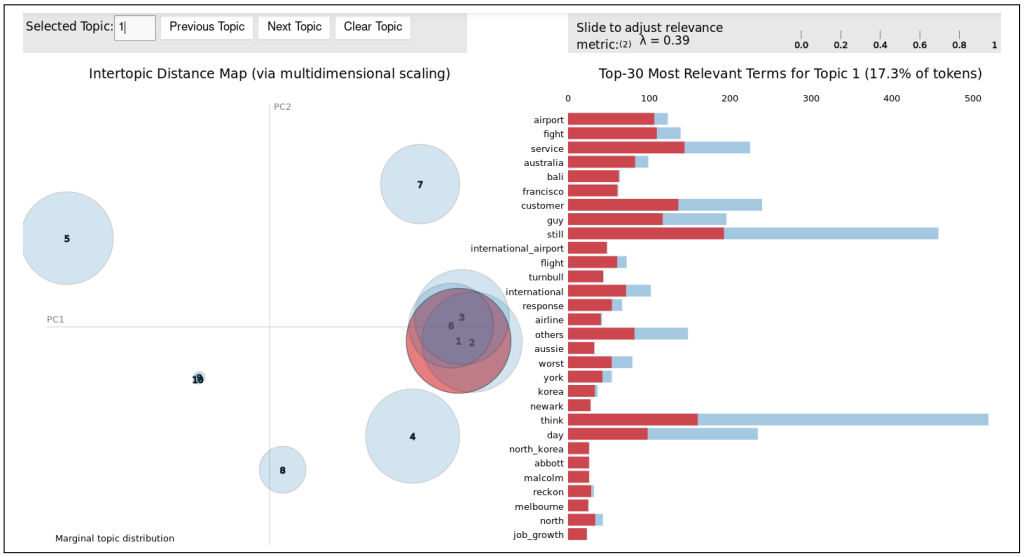

Figure 4.19: Topic 1: Flights

The following identified topic is the fourth one, figure 4.20. This one is about the Manchester United as it contains words like "Manchester United", "Mourinho", "Gaal", "Zlatan", "Schweinteiger", "Pogba", etc. In this case, LDA is able again to capture the structure of this topic successfully. Most of the words are only relevant in this topic and since all the words are related to each other we can state that the topic has been well identified.

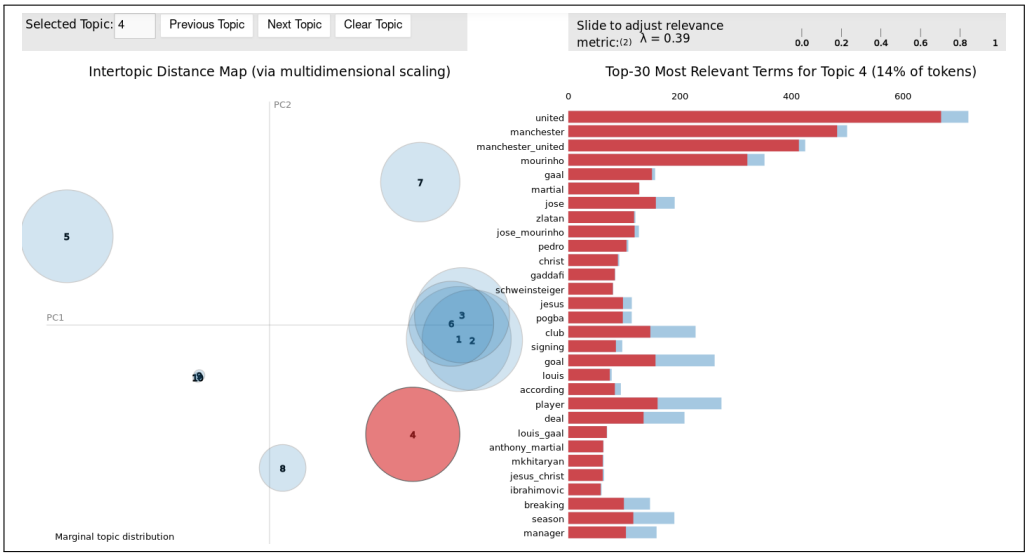

Figure 4.20: Topic 4: Manchester United

The fifth topic (figure 4.21) seems to be about Kodak. It is composed of words like: "Kodak", "Print", "nexpress" (it is a Kodak's platform), "digital", "inkjet", "printer" and more Kodak's platforms. In this case, LDA is able again to capture the structure of this topic successfully. Most of the words are only relevant in this topic and all of them are related to each other so we can declare that this topic has been well defined.

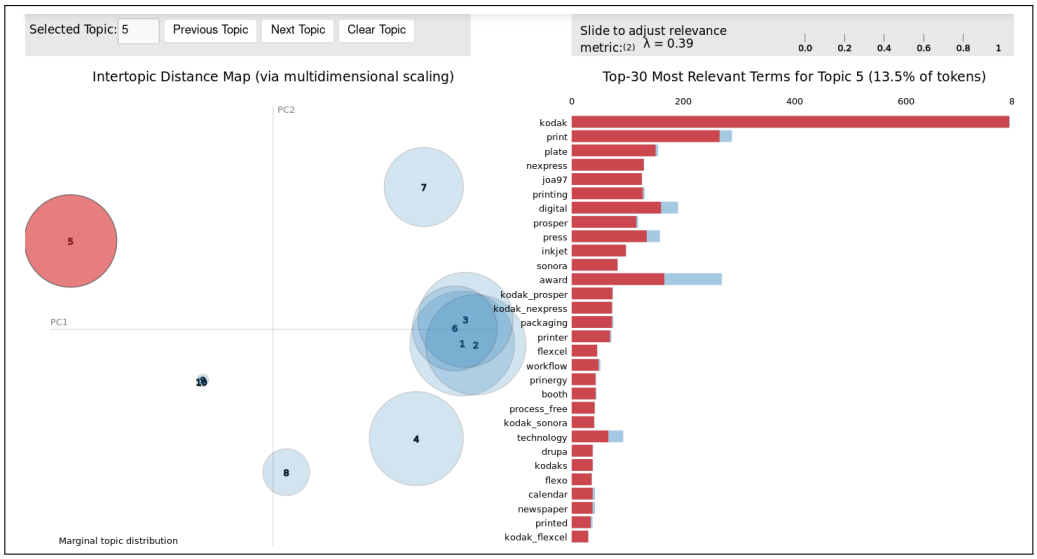

Figure 4.21: Topic 5: Kodak

The next topic (figure 4.22) is trivial to associate it to Twitter since it is made of words like "retweeted", "follower", "unfollowers", "gained", "unfollowed" and "tweet stats". The word "follower" is relevant in this topic and others since its blue bar is much bigger than the red one. The remaining words are only relevant in this topic.

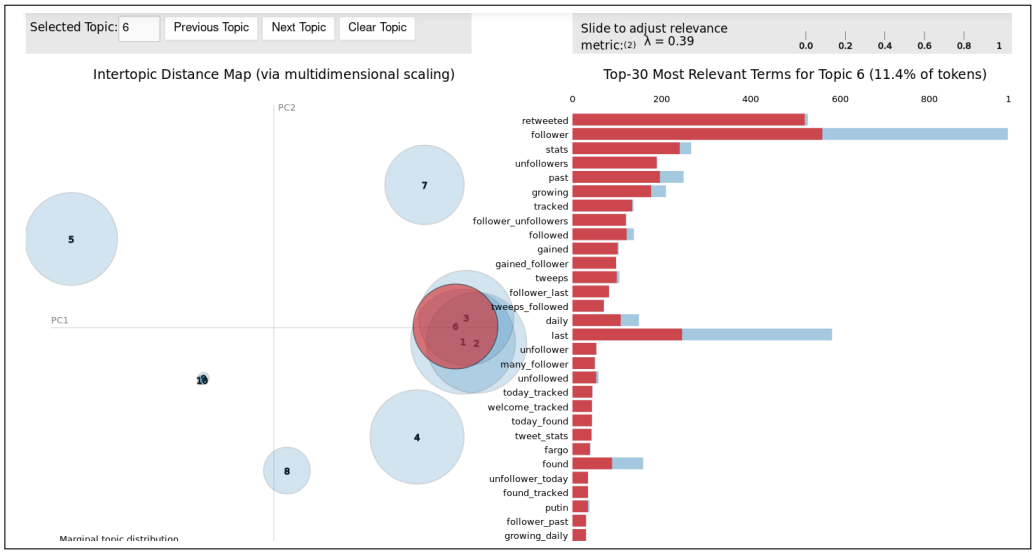

Figure 4.22: Topic 6: Twitter

Finally, the last well-defined topic is the eighth one (figure 4.23) which is about an Indian movie called "Shivaay". Almost all the compound words contains the word "Shivaay", for example: "sunday_ shivaay", "teamshivvay", "shivaay_shoot", etc. It also contains words like "bollywood" and "premiere". In this topic, only the words "action", "sunday" and "shoot" are relevant in other topics.

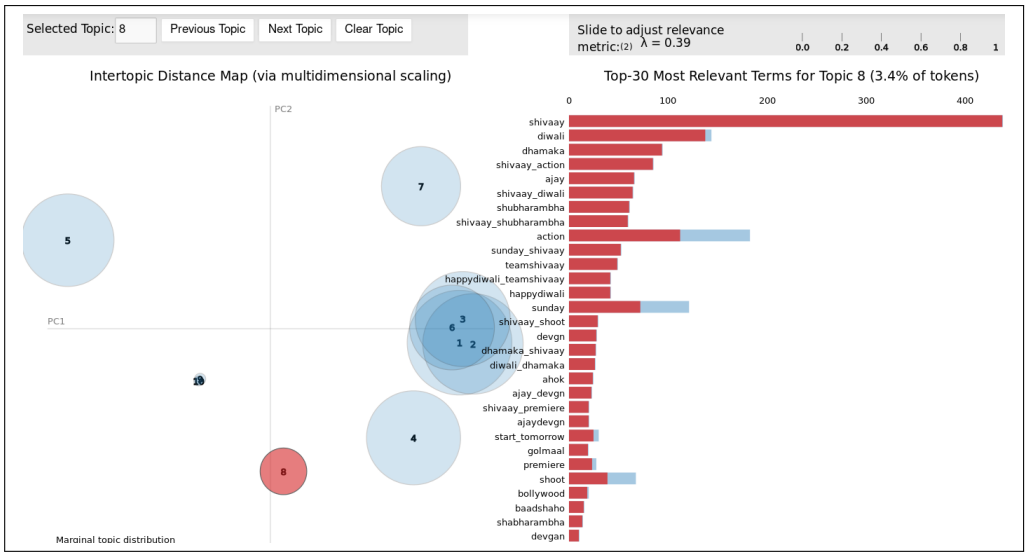

Figure 4.23: Topic 8: Shivay, an Indian movie

Again, the topics of the followers are not similar to the topics of the followed person.

4.7 New York Times's Topics

In this case, LDA has not been able to obtain good topics. There is only one topic which can be given a label, war, but it is still not well defined. This topic is the number eight (figure 4.24). It contains words like "Syria", "strike" and "army".

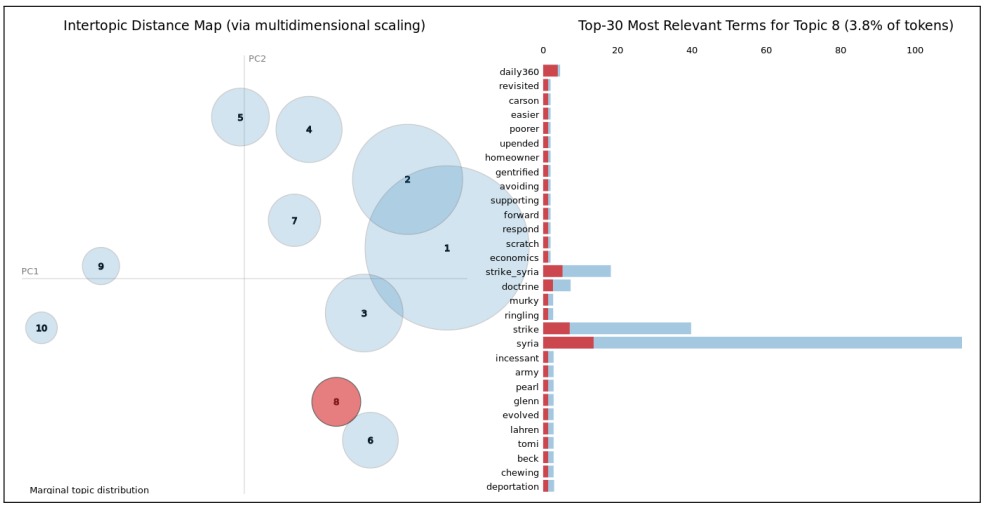

Figure 4.24: Topic 8: War

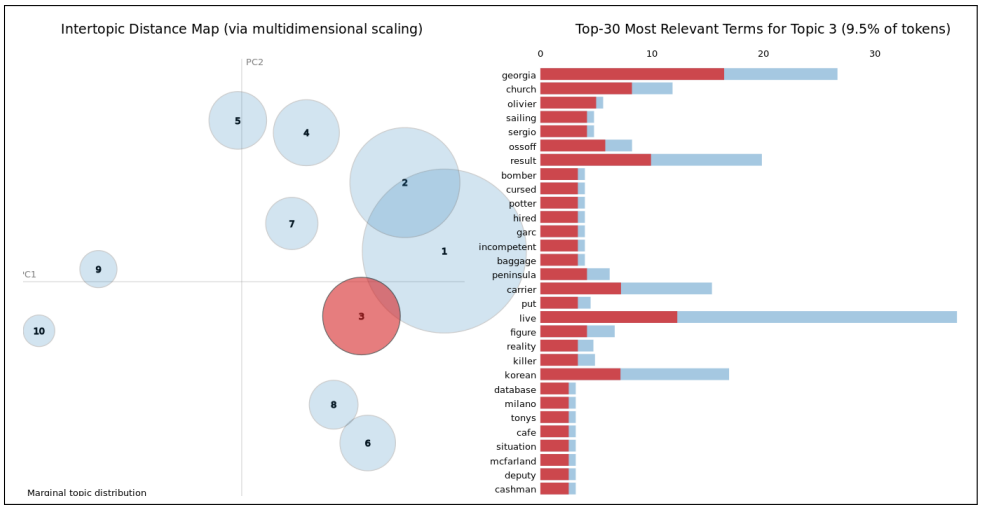

Figure 4.25: Topic 3: Contains words like "Korea", "Jon Ossoff", "Church", "Milano" and "cursed". The topic is not well defined because its word are not related to each other so we cannot assign it a label.

4.8 New York Times' Followers' Topics

Among the topics LDA has generated, only two have been well defined. The first one (figure 4.26) has been labeled as "Politics" since it contains words like "governor", "Clinton", "vote", "Obama", "Trump", "State" and "conservative".

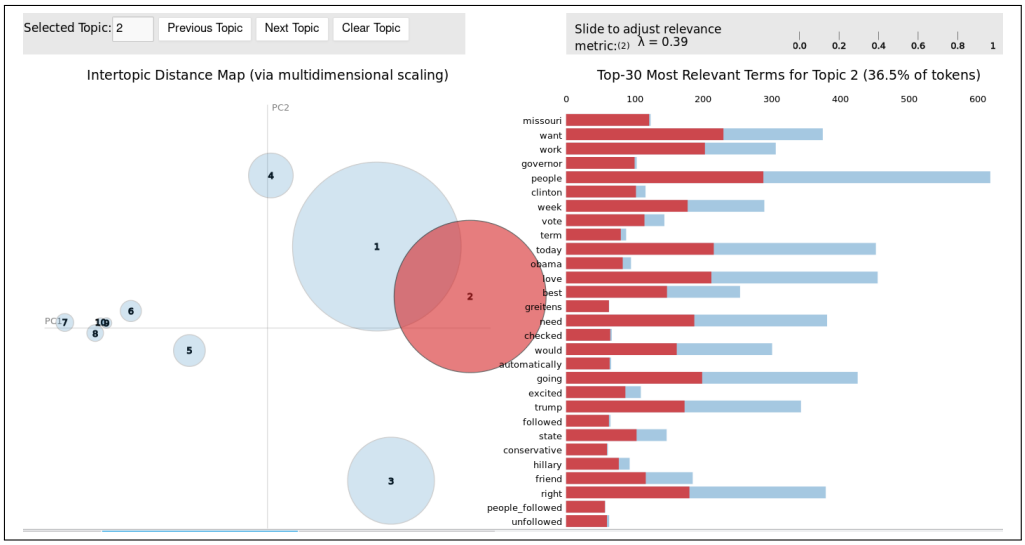

Figure 4.26: Topic 2: Politics.

The following topic (figure 4.27) has been labeled as "immigrants" since it is composed of words like "immigrant", "undocumented", "Turkish", "Kurdish", "protection", "border" and "Trump". Except this last word, the other words are only relevant in this topic, so we can affirm that this topic is well defined.

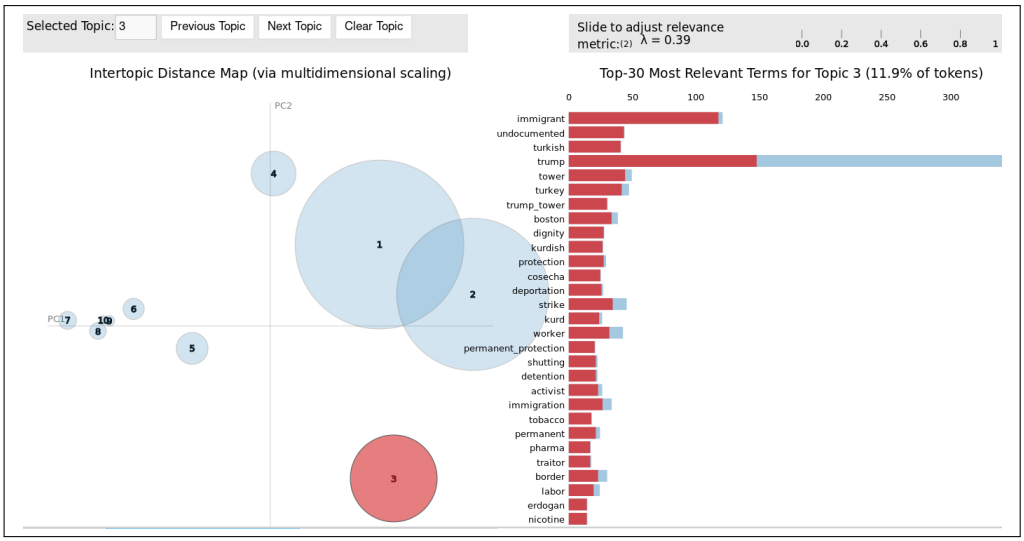

Figure 4.27: Topic 3: Inmigrants

Although LDA has not been able to generate a good model for the New York Times, we know it writes about news. This two well-defined topics for the followers of the New York Times may have been used by the New York Times too since they are a hot topic in the news, but the followers of the New York Times must write about many other topics which have not been found using LDA.

4.9 Leonardo DiCaprio's Topics

LDA has been able to obtain successfully several well-defined topics. All the topics detected are related to the environment so we can conclude that he is an environment activist as Wikipedia confirms it.

The first topic (figure 4.28) contains the following words: "climate change", "world", "action", "support", "protect", "Paris Agreement", "energy", "planet", etc. So, it is possible to call it "Climate Change". Most of the words are relevant in other topics which means that there are similar topics.

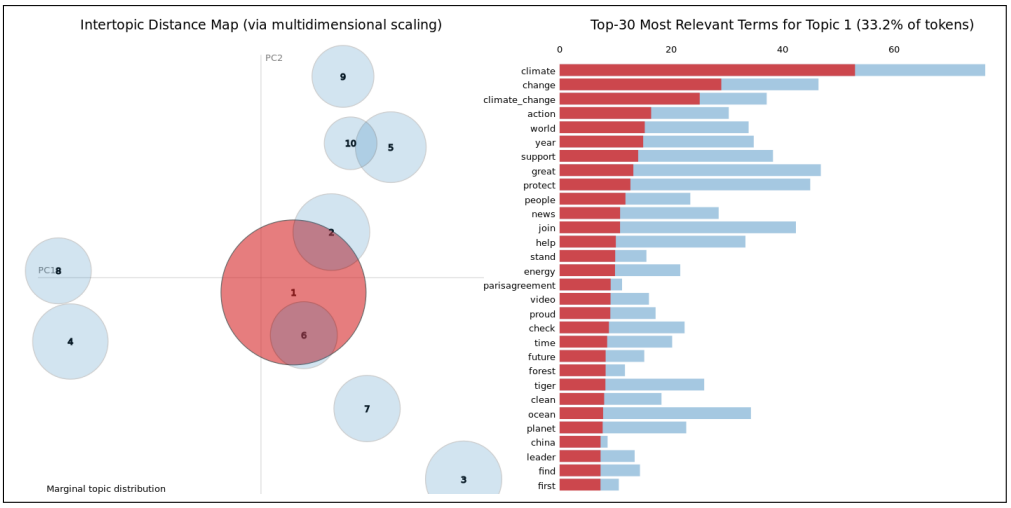

Figure 4.28: Topic 1: Climate Change

The fourth topic (figure 4.29) is clearly about elephants and ivory since it is made of words like: "elephant", "ivory", "protect", "help", "killed", "wild" and "elephantsneedus".

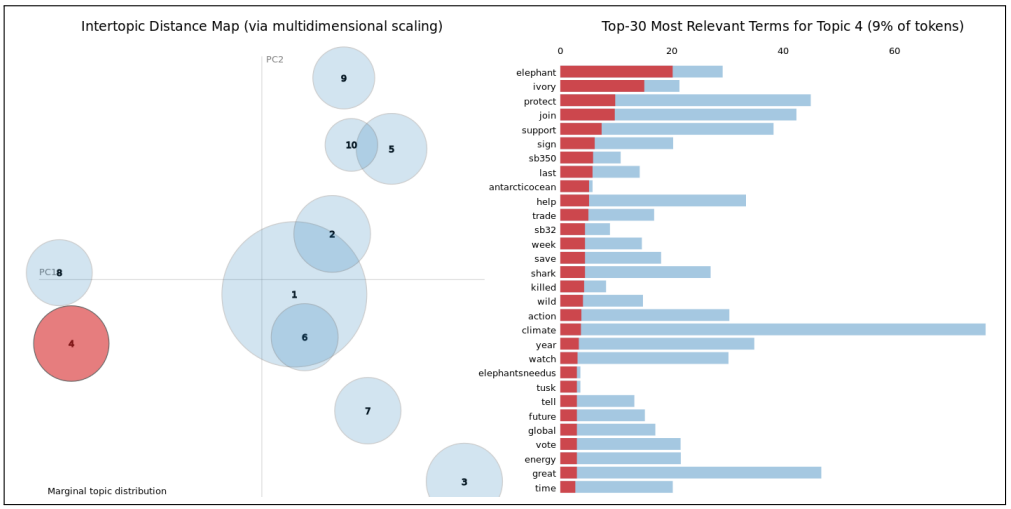

Figure 4.29: Topic 4: Elephants and ivory

The fifth topic (figure 4.30) has words like "Earth day", "planet", "climate change", "demandclimateaction" among others similar words so it has been labeled: "Earth Day".

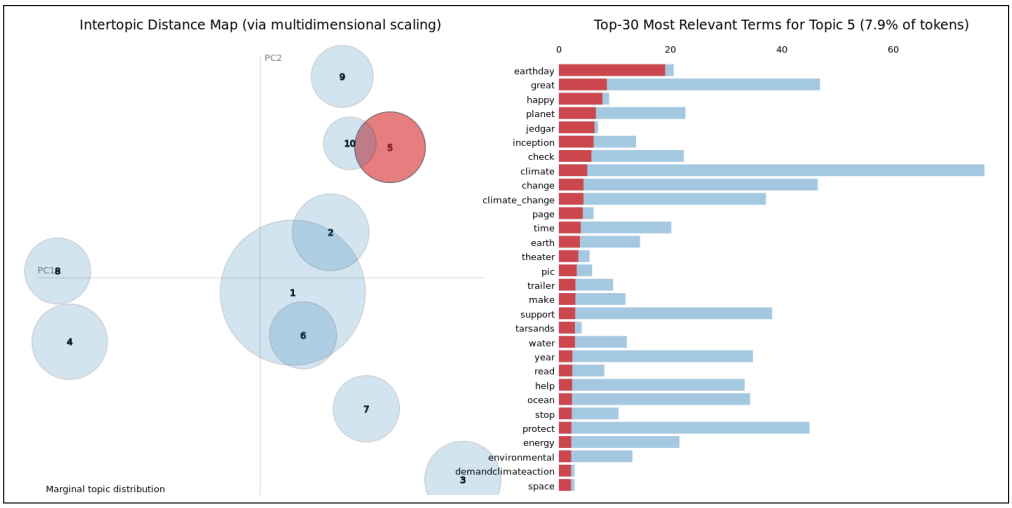

Figure 4.30: Topic 5: Earth Day

The last well-defined topic is the eighth one (figure 4.31). Since it contains words like "shark", "fin", "stopsharkfinning" it has been called "Sharks and stop shark finning".

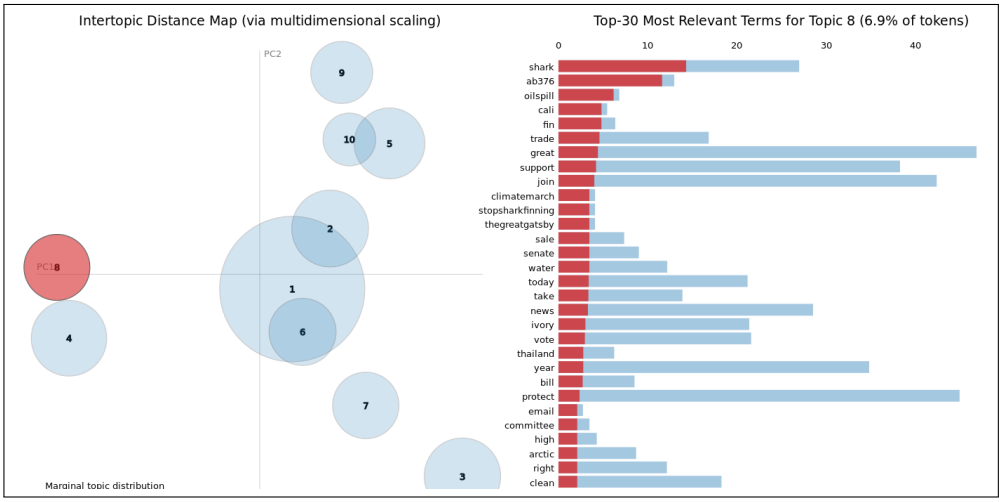

Figure 4.31: Topic 8: Shark and stop shark finning

4.10 Leonardo DiCaprio's Followers' Topics

In this case only one topic is clearly interpretable. It is the fifth one (figure 4.32) which is about mental health. It contains words like: "mental health", "wellness", "mental illness", "suicide", "counselling", "stigma" and "meditation.

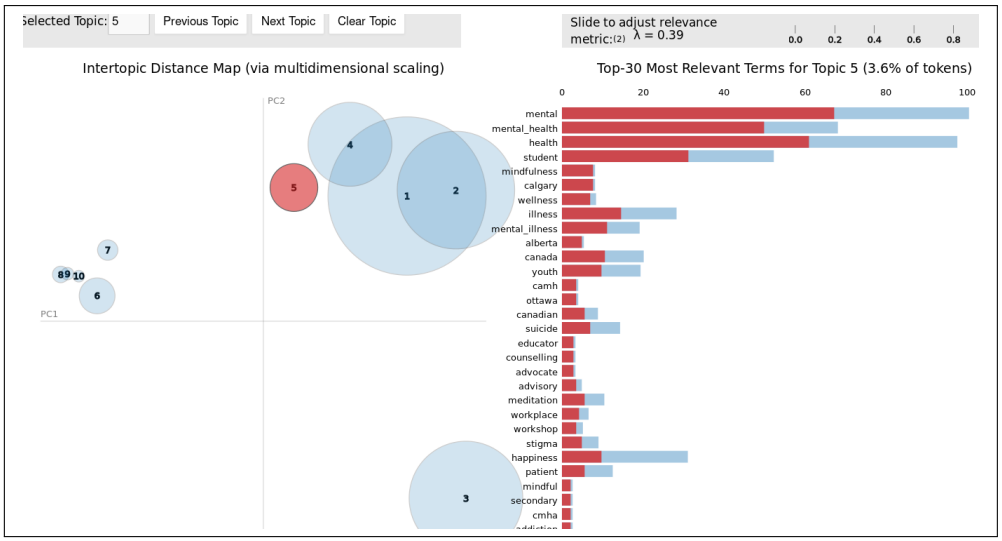

Figure 4.32: Topic 5: Mental Health

There is another topic (figure 4.33) which can be labeled but it is really small. This means that it is not a prevalent topic in the corpus. It is about politics since it contains words like "Bernie Sanders", "Hillary Clinton", "senator" and "Wikileaks".

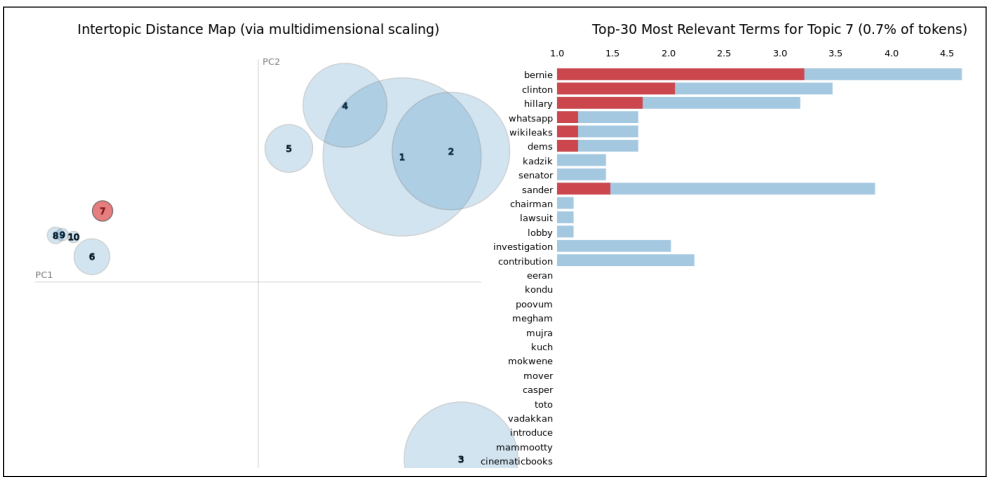

Figure 4.33: Topic 7: Politics

In this case, it also happens that a user does not write about the same topics as his/her followers. Leonardo DiCaprio writes about the protection of the environment but his followers do not.

Chapter 5

Conclusions

At the beginning of this thesis, a brief introduction to the current state of the art of Data Mining in social networks like Twitter has been shown.

After that, it is explained step by step how data mining has been used on Twitter. Furthermore, it is not only explained from a theoretical point of view, it has also been shown how to use the needed libraries.

The analysis done on the topics generated by LDA has shown that Twitter is not the best use-case for LDA. Although some topics have been clearly identified, LDA has not been able to capture the full set of topics of the users and their followers. This problem has been magnified analyzing the set of followers of a user. This is due to the nature of Twitter. A Tweet is a message of 140 characters and each Tweet is usually about a different topic. This means that the dataset has a lot of noise and it is difficult to identify topics.

Nevertheless, we have been able to see that Barack Obama, NASA, Lewis Hamilton and Leonardo DiCaprio do not talk about the same topics as their followers. This result seems correct because they usually talk about a small number of topics which are usually related to their profession while their followers, who are not famous, talk about a wider range of topics. Nonetheless, in order to obtain good results, different approaches should be followed. As we have seen, Standard LDA has not been able to obtain good models and that affects to the accuracy of the conclusions.

5.1 Different Approaches and Extensions

Although the main goal for this work has been met, there is still room for improvement and extensions. First, different approaches should be followed in order to analyze data from Twitter to obtain better results. For instance, Wayne Xin Zhao et al. propose in [26] an LDA variant specially designed for Twitter. This variant

takes into account that a single tweet is usually about a single topic. After proving that Standard LDA, the same that has been used in this work, and Author-Topic model do not provide good results, they show that Twitter-LDA clearly outperforms those two models.

Dat Quoc Nguyen et al. present in [27] two models especially designed for corpora with short documents. They propose to extend LDA and Dirichlet Multinomial Mixture (DMM) incorporating latent feature vector representations of words trained on very large corpora to improve the word-topic mapping learned on a smaller corpus. This approach is effective in short documents. The drawbacks of this approach is that DMM assumes that each document has only one topic which implies that a big preprocessing step is needed to make each document have just one topic. In addition, it has been tested only on Tweets about 4 topics: Microsoft, Google, Apple and Twitter. In this thesis, we can find a huge number of topics so we do not know how well this approach will behave in our case.

Jianhua Yin and Jianyong Wang propose in [28] Gibbs Sampling algorithm for the Dirichlet Multinomial Mixture model for short text clustering. Their proposal can infer the number of clusters and works well with short documents. In addition, each cluster also provides its representative words like LDA.

Rishabh Mehrotra et al. propose in [29] grouping tweets by hashtag as we have done in this work and they also took into account that most tweets do not have a hashtag, so they also propose an algorithm for assign hashtags automatically. This approach led them to better results than just using original hashtags. The problem of this approach is that they use Standard-LDA, a model designed for big documents without noise like news corpus. Therefore, it is not designed for Twitter.

As the approach followed in this work has not given excellent results, we should try all these approaches to find out which one is better for this case. It could be also a good idea to combine the last approach, automatically assign hashtags to Tweets without hashtags, with other approaches like Twitter-LDA.

After that, the next step would be to analyze thousands of users and their followers in order to obtain more robust results. In this thesis, only few users and their followers have been analyzed. In order to analyze thousands of users and their followers, it would be needed to use a statistical measure like Bhattacharyya distance to compute the differences between models. In addition, there are measurements to compute the quality of a model without visual inspection. In this thesis, a simple inspection of the models has been used. Lev Konstantinovskiy in [30] propose a measurement called: "topic coherence" to compute the quality of a model. This kind of measurement is needed to analyze thousands of users.

A possible extension would be to analyze the political orientation of users in Twitter. This has already been studied by Raviv Cohen and Derek Ruths in [31] and they

showed that it is difficult to obtain good results using a dataset of tweets of users who do not declare their political views although they can be inferred by manual inspection.

Another possible extension would be to perform a sentiment analysis of tweets. Alexander Pak and Patrick Paroubek do that in [32]. They build a sentiment classifier obtaining good results using three datasets: one of positive sentiments, one of negative sentiments and one about facts.

Chapter 6

Conclusiones

Al comienzo de este trabajo se ha realizado una breve introducción al actual estado de la minería de datos en redes sociales como Twitter.

A continuación, se explica paso a paso cómo se ha aplicado la minería de datos en Twitter. Además, no solo se explica desde un punto de vista teórico, sino que se muestra el uso de las bibliotecas necesarias.

El análisis de los temas generados por LDA ha mostrado que Twitter no es el mejor caso de uso para usar LDA. Aunque algunos temas han sido claramente identificados, no ha sido posible obtener el conjunto de todos los temas de los usuarios y sus seguidores. Este problema se ha agrandado al analizar el conjunto de seguidores de un usuario. Esto es debido a la naturaleza de Twitter. Un Tweet es un mensaje de 140 caracteres y cada Tweet suele ser sobre diferentes temas. Esto significa que el dataset tiene mucho ruido y por tanto es muy difícil identificar temas.

Sin embargo, ha sido posible ver que Barack Obama, NASA, Lewis Hamilton y Leonardo DiCaprio no hablan de los mismos temas que sus seguidores. Este resultado parece correcto debido a que ellos solo hablan de un número reducido de temas, generalmente relacionados con su profesion, sin embargo, sus seguidores, que no son famosos, hablan sobre un abanico de temas mucho más grande. Para obtener buenos resultados habría que seguir otros planteamientos. Como hemos visto, *Standard LDA* no ha sido capaz de obtener buenos modelos y eso afecta a la precisión de las conclusiones.

6.1 Planteamientos diferentes y extensiones

A pesar de que el objetivo principal de este trabajo se ha cumplido, hay mucho que se puede mejorar y ampliar. En primer lugar, para obtener mejores resultados usando Twitter como conjunto de datos habría que seguir otros enfoques. Por ejemplo,

Wayne Xin Zhao et al. en [26] proponen una variante de LDA especialmente diseñada para Twitter. Esta tiene en cuenta que cada Tweet suele ser sobre un único tema. Después de demostrar que *Standard LDA* y *Author-Topic Model* no consiguen dar buenos resultados, muestran que Twitter-LDA claramente los mejora.

Dat Quoc Nguyen et al. en [27] proponen dos modelos especialmente diseñados para corpus con documentos pequeños. Se presenta una extensión de LDA y *Dirichlet Multinomial Mixture* (DMM) que incorpora representaciones de vectores con características latentes de palabras entrenado en un corpus muy grande para mejorar el aprendizaje de palabras y temas en pequeños corpus. Este planteamiento es efectivo en documentos pequeños. Las desventajas son que DMM asume que cada documento tiene solo un tema, lo cual implica que haría falta una etapa de preprocesamiento muy grande para hacer que cada documento tenga un único tema. Además, solo se ha probado en Tweets sobre cuatro temas: Microsoft, Google, Apple y Twitter. En este trabajo fin de grado es posible encontrar un gran número de temas, por lo que no sabemos cómo de bueno es este planteamiento en nuestro caso.

Jianhua Yin y Jianyong Wang en [28] proponen el algoritmo *Gibbs Sampling* para el modelo *Dirichlet Multinomial Mixture* para *clustering* de textos pequeños. En su propuesta pueden inferir el número de *clusters* y además, funciona bien con documentos pequeños. Cabe citar también que consiguen mostrar las palabras más características de cada *cluster*.

Rishabh Mehrotra et al. en [29] proponen agrupar Tweets por *hashtag* como se ha realizado en el presente trabajo y además tienen en cuenta que la mayoría de los Tweets no tienen *hashtag* por lo que proponen un algoritmo para asignar *hashtags* automáticamente. Este planteamiento les permite obtener mejores resultados que usando solamente los *hashtags* originales. El problema es que usan *Standard LDA*, un modelo diseñado para grades documentos y sin ruido como los artículos periodísticos, es decir, no está diseñado para Twitter.

Como el planteamiento seguido en el presente trabajo no ha dado resultados excelentes, deberíamos probar todas las ideas comentadas anteriormente para descubrir cuál es mejor en este caso. Sería también una buena idea combinar en último planteamiento, automatizar la creación de *hashtags* de Tweets sin *hashtags*, con otras ideas como Twitter-LDA.

El siguiente paso sería analizar miles de usuarios y sus seguidores para obtener resultados más robustos. En este trabajo fin de grado solo unos pocos usuarios y sus seguidores han sido analizados. Para analizar miles de usuarios y sus seguidores, sería necesario el uso de una medida estadística como la distancia de *Bhattacharyya*. Además, hay medidas para calcular la calidad de un modelo sin necesidad de una inspección visual. En el presente trabajo, una simple inspección visual de los modelos se ha usado, sin embargo, Lev Konstantinovskiy en [30] propone una medida llamada: "*topic coherence*". Este tipo de medida es necesaria para analizar miles

de usuarios.

Una posible extensión sería analizar la orientación política de los usuarios en Twitter. Esto ya ha sido estudiado por Raviv Cohen y Derek Ruths en [31]. Mostraron que es difícil obtener buenos resultados usando un conjunto de datos formado por Tweets de usuarios que no declaran sus ideas políticas aunque puedan ser inferidas mediante inspección manual.

Otra posible extensión sería realizar un análisis de sentimientos de Tweets. Alexander Pak y Patrick Paroubek en [32] construyeron un clasificador de sentimientos con buenos resultados usando tres conjuntos de datos: uno de sentimientos positivos, otro de negativos y otro sobre hechos.

References

[1] U Kang and Christos Faloutsos. "Big Graph Mining: Algorithms and Discoveries". In: *ACM SIGKDD Explorations Newsletter* 14 (2012), pp. 29–36.

[2] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. "Earthquake shakes Twitter users: real-time event detection by social sensors". In: *Proceedings of the 19th international conference on World wide web*. ACM. 2010, pp. 851–860.

[3] Suin Kim et al. "Sociolinguistic analysis of twitter in multilingual societies". In: *Proceedings of the 25th ACM conference on Hypertext and social media*. ACM. 2014, pp. 243–248.

[4] Suin Kim, JinYeong Bak, and Alice Haeyun Oh. "Do You Feel What I Feel? Social Aspects of Emotions in Twitter Conversations." In: *ICWSM*. 2012.

[5] Daniel Ramage, Susan T Dumais, and Daniel J Liebling. "Characterizing microblogs with topic models." In: *ICWSM* 10 (2010).

[6] Jeon Hyung Kang and Kristina Lerman. "Using lists to measure homophily on twitter". In: *AAAI Workshops*. 2012.

[7] Kyungyup Daniel Lee, Kyung-Ah Han, and Sung-Hyon Myaeng. "Capturing Word Choice Patterns with LDA for Fake Review Detection in Sentiment Analysis." In: *WIMS*. 2016.

[8] Liangjie Hong and Brian D Davison. "Empirical study of topic modeling in twitter". In: *Proceedings of the first workshop on social media analytics*. ACM. 2010, pp. 80–88.

[9] Gwan Jang and Sung-Hyon Myaeng. "Analysis of spatially oriented topic versatility over time on social media". In: *Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on*. IEEE. 2015, pp. 573–578.

[10] Aisylu Khairullina et al. "Observing Behaviors of Information Diffusion Models for Diverse Topics of Posts on VK". In: *Data Mining Workshop (ICDMW), 2015 IEEE International Conference on*. IEEE. 2015, pp. 1098–1102.

[11] Jagan Sankaranarayanan et al. “Twitterstand: news in tweets”. In: *Proceedings of the 17th acm sigspatial international conference on advances in geographic information systems*. ACM. 2009, pp. 42–51.

[12] Andres Lou, Diana Inkpen, and Chris Tanasescu. “Multilabel Subject-Based Classification of Poetry”. In: *Nature* 2218 (2015), pp. 30–7.

[13] Thomas L Griffiths and Mark Steyvers. “Finding scientific topics”. In: *Proceedings of the National academy of Sciences* 101.suppl 1 (2004), pp. 5228–5235.

[14] David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent dirichlet allocation”. In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[15] Wei Li and Andrew McCallum. “Pachinko allocation: DAG-structured mixture models of topic correlations”. In: *Proceedings of the 23rd international conference on Machine learning*. ACM. 2006, pp. 577–584.

[16] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[17] Twitter Inc. *Twitter Developer Documentation, Rate Limits: Chart*. `https://dev.twitter.com/rest/public/rate-limits`. [Online; accessed 15-April-2017]. 2017.

[18] Twitter Inc. *Twitter Developer Documentation, GET statuses/usertimeline*. `https://dev.twitter.com/rest/reference/get/statuses/user_timeline`. [Online; accessed 15-April-2017]. 2017.

[19] Twitter Inc. *Twitter Developer Documentation, GET search/tweets*. `https://dev.twitter.com/rest/reference/get/search/tweets`. [Online; accessed 15-April-2017]. 2017.

[20] Twitter Inc. *Twitter Developer Documentation, GET followers/list*. `https://dev.twitter.com/rest/reference/get/followers/list`. [Online; accessed 15-April-2017]. 2017.

[21] Ryan McGrath and eoso. *Twython Pull Request Cursor fix 386*. `https://github.com/ryanmcgrath/twython/pull/386`. [Online; accessed 15-April-2017]. 2017.

[22] Twitter Inc. *Twitter Developer Documentation, Streaming APIs*. `https://dev.twitter.com/streaming/overview`. [Online; accessed 15-April-2017]. 2017.

[23] Radim Řehůřek and Petr Sojka. “Software Framework for Topic Modelling with Large Corpora”. English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. `http://is.muni.cz/publication/884893/en`. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[24] Matthew Hoffman, Francis R Bach, and David M Blei. "Online learning for latent dirichlet allocation". In: *advances in neural information processing systems*. 2010, pp. 856–864.

[25] Carson Sievert and Kenneth E Shirley. "LDAvis: A method for visualizing and interpreting topics". In: *Proceedings of the workshop on interactive language learning, visualization, and interfaces*. 2014, pp. 63–70.

[26] Wayne Xin Zhao et al. "Comparing twitter and traditional media using topic models". In: *European Conference on Information Retrieval*. Springer. 2011, pp. 338–349.

[27] Dat Quoc Nguyen et al. "Improving topic models with latent feature word representations". In: *Transactions of the Association for Computational Linguistics* 3 (2015), pp. 299–313.

[28] Jianhua Yin and Jianyong Wang. "A dirichlet multinomial mixture model-based approach for short text clustering". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 233–242.

[29] Rishabh Mehrotra et al. "Improving lda topic models for microblogs via tweet pooling and automatic labeling". In: *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM. 2013, pp. 889–892.

[30] Lev Konstantinovskiy. *America's Next Topic Model*. `http://www.kdnuggets.com/2016/07/americas-next-topic-model.html`. [Online; accessed 7-May-2017]. 2016.

[31] Raviv Cohen and Derek Ruths. "Classifying political orientation on Twitter: It's not easy!" In: *ICWSM*. 2013.

[32] Alexander Pak and Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining." In: *LREc*. Vol. 10. 2010. 2010.

Appendix A

Environment setup

This appendix provides the documentation needed to install all the necessary libraries to execute the code used in this thesis.

Example A.1: Needed libraries

```
sudo apt-get update
sudo apt-get install mysql-server
sudo mysql_secure_installation
sudo mysql_install_db

sudo apt-get install python-mysqldb
sudo apt-get install python-dev libmysqlclient-dev
sudo apt-get install python-pip
pip install MySQL-python

sudo apt-get install git
wget https://repo.continuum.io/miniconda/Miniconda2-latest-Linux-
    x86_64.sh
chmod u+x Miniconda2-latest-Linux-x86_64.sh
./Miniconda2-latest-Linux-x86_64.sh

#Close terminal and open it again
conda create --name nlp
source activate nlp
sudo apt-get install python-numpy python-scipy python-matplotlib
    ipython ipython-notebook python-pandas python-sympy python-nose
conda install -c anaconda gensim

wget https://github.com/bmabey/pyLDAvis/archive/master.zip
unzip master.zip
cd pyLDAvis-master/
python setup.py install
pip install ipython[all]
pip install nltk
pip install matplotlib
```

```
pip install twython
```