TÉCNICO LISBOA

Automating the Response Processes in TAP PORTUGAL's Social Networks

Tiago Miguel Rodrigues Simões

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. João Paulo Baptista de Carvalho
Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur

Examination Committee

Chairperson: Prof. Daniel Jorge Viegas Gonçalves
Supervisor: Prof. João Paulo Baptista de Carvalho
Members of the Committee: Prof. Bruno Emanuel da Graça Martins

May 2017

Acknowledgments

I would like to thank my supervisors, Professors João Carvalho and Luísa Coheur, and TAP PORTUGAL for giving me the opportunity to work with them.

I would also like to thank my family and friends for all the support throughout all these years studying at Instituto Superior Técnico. A special thanks to Clara for all the support and help.

Abstract

The rapid dissemination of information and ideas, in combination with the rise of the development of instant communication, lead to an accelerated growth of short texts. Since short texts might enclose valuable intelligence, mining these sources has become of increased interest for corporations. It was in this context that TAP PORTUGAL showed interest in the elaboration of a study, based on machine learning algorithms, to identify the comments that its clients make on Facebook and Twitter in 4 typologies: Praise, Complaints, Questions and Suggestions.

Analyzing the new created corpus, it was possible to characterize it as: sparse and short; multilingual; unlabeled; noisy; imbalanced; and non-standard. From the annotated corpus and acquired knowledge of manually labeling it, we retrieved and analyzed predominant complaints, praises, questions and suggestions.

The conducted experiments using machine learning algorithms pinpointed k-Nearest Neighbors and Support Vector Machine as the best classifiers among the ones studied, achieving very high scores in certain conditions. Despite the good scores achieved by both classifiers, Support Vector Machine was clearly the most robust model presenting very good results when reducing the training data as well.

Keywords

TAP PORTUGAL; Text Categorization; Social Networks

Resumo

A rápida disseminação de informações e de ideias, conjuntamente com o aumento do desenvolvimento de comunicações instantâneas, potenciou o rápido crescimento do aparecimento de textos curtos. Estes podem conter/esconder informação valiosa em âmbito empresarial, razão pela qual tornou-se interessante, para as empresas extrair informação, se possível de forma automática, a partir destes. Foi neste contexto que a TAP PORTUGAL mostrou interesse na elaboração de um estudo, com base em algoritmos de aprendizagem automática, que classificassem os comentários que os seus clientes efetuam no Facebook e no Twitter em 4 tipologias: Elogios, Reclamações, Questões e Sugestões.

Analisando o novo corpus criado, foi possível caracterizá-lo em: escasso e curto; multilíngue; não categorizado; ruidoso, não balanceado e não estruturado. Com base no conhecimento adquirido e da categorização manual, foi criado um corpus anotado onde identificamos e analisamos queixas, elogios, perguntas e sugestões, mais predominantes.

As várias experiências realizadas usando algoritmos de aprendizagem automática classificaram o k-vizinhos mais próximos e a máquina de vetores de suporte como os melhores classificadores, entre os estudados, obtendo resultados bastante elevados em certas condições. Apesar dos bons resultados alcançados por ambos os classificadores, a máquina de vetores de suporte demonstrou ser o classificador mais robusto, apresentando resultados muito bons mesmo quando se reduz os dados de treino.

Palavras Chave

TAP PORTUGAL; Categorização de Texto; Redes Sociais

Contents

List of Figures

List of Tables

Acronyms

ARC	Air Race Championship
BoW	Bag of Words
FAQ	Frequently Asked Questions
FFP	Fuzzy Fingerprints
FN	False Negative
FP	False Positive
HTML	HyperText Markup Language
icf	Inverse Class Frequency
idf	Inverse Document Frequency
IG	Information Gain
IR	Information Retrieval
IST	Instituto Superior Técnico
kNN	k-Nearest Neighbors
ML	Machine Learning
MNB	Multinomial Naïve Bayes
NB	Naïve Bayes
NLP	Natural Language Processing
NLTK	Natural Language Toolkit
RBF	Radial Basis Function

SMO Sequential Minimal Optimization

SMS Short Message Service

SVM Support Vector Machine

TAP TAP PORTUGAL

TFIDF Term Frequency-Inverse Document Frequency

TFITF Term Frequency-Inverse Topic Frequency

TN True Negative

TP True Positive

T2S2 Tweet-Topic Similarity Score

UEFA Union of European Football Associations

URL Uniform Resource Locator

1

Introduction

Contents

"Feedback is the cheapest, most powerful, yet, most under used management tool that we have at our disposal." [1]. Feedback provides valuable knowledge in different areas, ranging from politics to fashion, and from leisure activities to scientific methods. Feedback comes in different formats including surveys, comments on social networks, word of mouth (referral) and it can be both positive and negative. The presence of feedback can be found in many different contexts such as: (a) guiding a company creating products or services that customers want to buy; (b) understanding if a product or a service requires enhancement; (c) a quality measure.

Due to the copious amount of text data available, understanding what is relevant has become humanly unfeasible [2]. Important insights given by customers' feedback are lost everyday because retrieving, sorting and classifying information from text is demanding, time consuming and expensive if executed manually. Therefore, the need to develop automated tools to carry out these tedious chores is rapidly gaining importance.

Automated text categorization is the process that consists of the automatic classification of documents into a fixed number of predefined categories, and it has become one of the core techniques for handling and organizing text data. Falling at the crossroads of Information Retrieval (IR) and Machine Learning (ML), automated text categorization allows to identify relevant information, having a significant impact in applications such as content management, contextual search, opinion mining, product review analysis and spam filtering [3].

1.1 Motivation

Social networks are platforms that bring people together to communicate, share ideas, opinions and interests. Although considered a relatively recent phenomenon, social networks are becoming an increasingly important part of any business's marketing and client based development platform. From the 2012 Social Media Marketing Industry Report [1], we can quote that: "94% of all businesses with a marketing department used social media as part of their marketing platform". Allying this fact with the statistics in the study State of the Media: The Social Media [4], that states that people spend more time on social networks than any other place on the Internet, we can conclude that social media is part of everyone's daily life. Hence the number of social networks is increasing every day and they are being used more frequently.

Social platforms, such as Facebook and Twitter, create opportunities to establish interactions among people, leading to mutual learning and sharing of valuable knowledge. Web-based applications contain a large volume of information about diverse demographics all over the world, offering insights about humans' patterns behavior that are impossible to observe in a more artificial setting, such as a lab or

[1] http://www.socialmediaexaminer.com/social-media-marketing-industry-report-2012/

a focus group [5]. This kind of network allows customers to create and discuss their opinions about anything in a matter of seconds, making social networks a powerful tool of communication. Users from all over the world can share their impressions about a product or a service with everyone as soon as they experience it. As consequence, the number of customers' reviews[2] is growing rapidly. For a popular service as a commercial airline, the magnitude of reviews can be in hundreds or even thousands per day.

TAP has an active participation on Customer Care service on its social networks, and it has an application to manage the response task of comments, messages, posts and tweets from its social media services. However, it is desired to have an application to automatically categorize all the feedback, that otherwise would have to be manually addressed and sorted by the Contact Center operators.

Structured and categorized feedback allows a better understanding of clients' needs. One of the main reasons for applying data mining methods to text documents collections is to structure them [6].

1.2 Objective

The goals of the work consists in:

1. Retrieve comments, messages, posts and tweets from TAP's clients on TAP's social media channels;

2. Create a corpus from the gathered text;

3. Create an annotated dataset with the following categories: complaints, praises, questions and suggestions, that can be used to train supervised ML algorithms;

4. Study and evaluate ML algorithms able to classify automatically the corpus in the given categories;

Creating a corpus containing TAP's customers feedback and setup a machine to automatically classify feedback into complaints, praises, questions and suggestions, allows the company to learn more about its clients and garner answers to the following questions: (a) What are the main flaws and what can be improved; (b) What are the main doubts or what information is elusive to access to; (c) What are the clients enjoying the most. Developing a mechanism that gleans all this information enables the company to design a platform that provides faster answers to its clients.

[2] In this context, review can be, among other things, a question, a complaint, a suggestions, etc.

1.3 Contributions

The work described in this dissertation consists of multiple experiments using supervised ML algorithms to address the problem of short text categorization on a new created corpus. This contains approximately 8400 short texts from TAP costumers gathered from TAP social media platforms (Facebook and Twitter). The results from experiments show that Support Vector Machine (SVM) and k-Nearest Neighbors (kNN) were able to correctly classify most of the instances despite the unbalanced dataset.

The main contributions are:

- A Portuguese annotated corpus containing acknowledgments, insults, complaints, praises, questions and suggestions from an airline's perspective;
- An analysis to the comments left at TAP social media accounts. Acquired knowledge from manually labeling the corpus and the list of the most common words, allow us to capture most frequent complaints and questions.
- A study using supervised ML algorithms, such as SVM, kNN, Multinomial Naïve Bayes (MNB) and an adaptation of Fuzzy Fingerprints (FFP) to short texts, to automatically classify complaints, praises, questions and suggestions.

The work results were presented to TAP and it was concluded that the goals were successfully accomplished.

1.4 Structure of the Document

The present thesis is organized as follows. Chapter 2: Fundamental Concepts and Related Work - It introduces essential concepts to understand the problem and the solution proposed and an analysis of similar studies about text categorization realized for other researchers. Chapter 3: Corpus - It explains the process of creating the corpus, along with a detailed description of the it. Chapter 4: Experiments and Results - It explains the setup and the results of the experiments using ML classifiers and the knowledge retrieved from the built corpus. Chapter 5: Conclusion and Future Work - This chapter closes the document summarizing its core aspects, and proposing directions for future work.

2

Fundamental Concepts and Related Work

Contents

The present chapter features essential concepts required to understand the topics throughout the manuscript (Section 2.1) and some studies that relate the state of the art for short text categorization (Section 2.2).

2.1 Fundamental Concepts

This section details the main concepts necessary to understand the developed work. Section 2.1.1 describes traditional methods for text representation. Section 2.1.2 introduces some algorithms usually used to address (short) text categorization and short text characteristics. Section 2.1.3 lists several approaches to preprocessing the text data retrieved from social media platforms. Section 2.1.4 presents adequate evaluation metrics for text categorization problems.

2.1.1 Text Representation

In the context of Natural Language Processing (NLP) and IR, the most common approach is to represent text is through unigrams and its respective counts, also known as, Bag of Words (BoW) [7–10]. In this representation all the information about the order of the words in a document is lost and the text is transformed into a bag of words together with word frequency counts. Longer n-grams have been considered in previous text classification works. Although, they failed to provide consistent results, depending deeply on the problem and dataset [7, 8].

Document 1	"Get your new car here. We have the best car for you."
Document 2	"Come here please. We need to talk to you."
Document 3	"Here we do the maintenance of your motorcycle at affordable prices."

Table 2.1: Example of documents.

Term	Frequency
Come	1
here	1
please	1
.	2
...	...

Table 2.2: Part of the Bag of Words of Document 2.

In the context of text categorization, the BoW representation is usually coupled with a weighting scheme, such as Term Frequency-Inverse Document Frequency (TFIDF) and binary weighting (presence/absence or 1/0) [11].

Term	car	here	.
Occurrences of this term in all documents	2	3	6
Number of documents that contain this term	1	3	3

Table 2.3: Number of occurrences of terms - "car", "here" and ".".

Taking into account the documents in Fig. 2.1 as example, words that appear exclusively in a document are more relevant to identify the subject of a document (e.g. "car" in Document 1), and less relevant if they appear in many documents (e.g. "here" that is present in all documents). Defining a set of documents as $D =< d_1, \ldots, d_n >$, TFIDF determines the most important terms in a document d.

It is widely recognized that TFIDF is one of the most popular term weighting schemes in text categorization [8, 12–14]. Its formula can be given by:

$$\mathrm{tfidf}(w) = \mathrm{tf}(w) \times \mathrm{idf}(w) \tag{2.1}$$

$$\mathrm{idf}(w) = \log\left(\frac{N}{N_i}\right) \tag{2.2}$$

Where w is the term, tf is the term frequency in a given document, N is the total number of documents the dataset contains, N_i is the number of documents in the dataset that contains the term w. The tfidf value is proportional to the number of times a term w appears in the document, but is offset by the frequency of the term w in the document, which accounts for the fact that some terms appear more frequently [9, 15].

2.1.2 Text Categorization

As outlined in the introduction, text categorization is the task of finding the correct category (or categories) for each document, given a restricted set of categories and a set of texts documents [16]. This area has received much attention over the years, being supervised learning one of the most common approaches to text categorization problems.

Supervised learning is divided in two phases (see Fig. 2.1): (i) learning phase – given a dataset with correctly classified instances, the algorithm builds a model around it; (ii) prediction phase – given a new instance, the model predicts its label considering what learned was on the previous phase, i.e., the machine uses knowledge provided by the training model to label the new documents.

Labeled documents can be gathered in several ways and from different sources, such as, social networks — through hash-tags (E.g. "I love shrimps #food") and dataset repositories — manually annotated datasets by experts (E.g. Reuters-21578[1]).

Taking into consideration that our aim is to build a predictive model to deal certain data, it makes sense to use this type of learning.

[1] A well-known dataset and possibly the most widely used test collection for text categorization research.

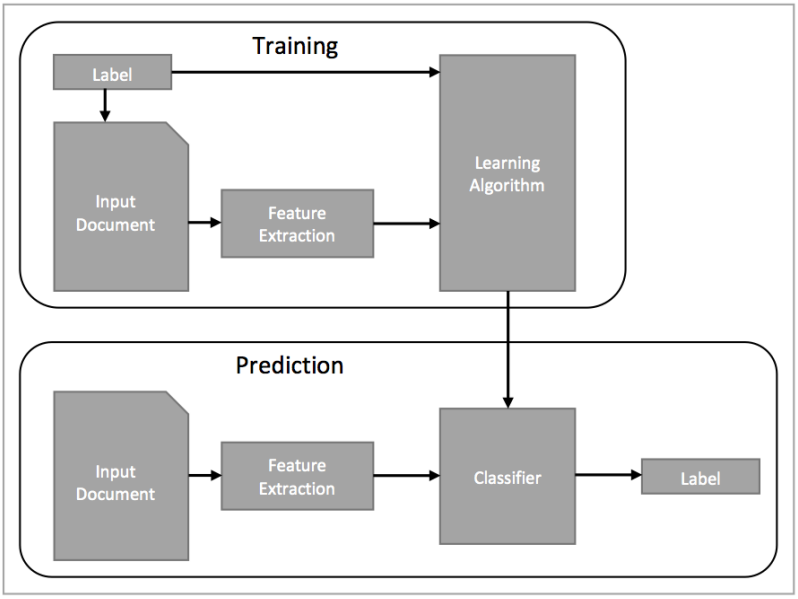

Figure 2.1: Supervised Text Classification. Adapted from Python Course.

Short Text Categorization

Extensive research has been made on the subject of text categorization, although few of these studies have focused on short text categorization [17]. Due to the sudden growth of e-commerce and online communications, short texts have become the most common type of text found on the web [17]. These texts are growing rapidly and they can be encountered in many application areas, such as online chats, Short Message Service (SMS), tweets, blog comments and short news. Unlike conventional documents (e.g. reports, magazines), these are usually: (a) noisier — abundance of irrelevant content; (b) less structured — misspells and non-standard terms are often found; (c) noticeable shorter — no longer than a few sentences; (d) large scale — they can be found in every application, process, messing system, among others; (e) unlabeled — majority of texts are not identified as belonging to a category.

Researches have demonstrated that supervised ML algorithms have been applied to an appreciable number of cases with different benchmark collections and have achieved satisfactory results in both short text classification and text classification [2, 3, 8, 9, 12, 17–19].

k-Nearest Neighbors kNN is a well-documented example-based classifier, that determines the class of an object according to its closest k neighbors [20]. It is known to have a fast training phase and be one of the simplest and best performing text classifiers [11].

In the training phase, kNN stores the representations of all training documents and the respective class labels, instead of building explicitly "declarative representations of categories" [21]. On the classification phase, kNN computes the distance between the new instance x and every training example x_i,

assigning to x the class of its k nearest neighbors. As illustrated by Fig. 2.2, if $k = 3$ (dashed line circle), c is assigned to class a, because there are two instances of class a just one of b. However, if $k = 9$ (solid line circle), c is assigned to class b because there are three instances of class a and six instances of class b. In order to minimize the impact of isolated instances, it is common to use $k > 1$ and odd values of k to avoid draws between classes. One interesting characteristic of kNN is that kNN does not attribute weights to features. They are equally relevant (irrelevant features have the same influence on the distance as important features).

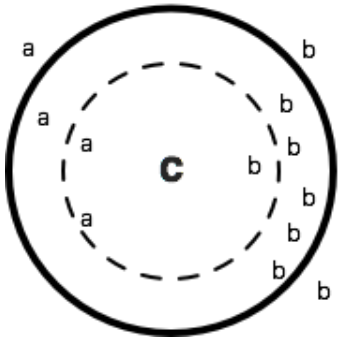

Figure 2.2: Example of kNN classification.

Taking in consideration that most of real word datasets are unbalanced (see Section 2.1.3), establishing the number of neighbors k is of utmost importance [11]. When $k = 1$, the new document is simply attributed to its nearest neighbors class, while high k values might lead kNN to attribute the new document to the majority classes.

Another characteristic of kNN is that it tends to incorrectly classify a document holding features never seen before (i.e., in the training phase), because it fails to calculate the distance value between the new document and the existing documents (undefined distance).

The main drawbacks of kNN are the "high computation cost of classification, because for each test document it must compute its similarity to all of the training documents" and the required space to store all training examples [11, 21].

Multinomial Naïve Bayes MNB is a popular probabilistic model designed for text classification based on Naïve Bayes (NB) theorem, known to be computational efficiency and a relatively good predictive performance. [8, 22, 23]. This classifier calculates the maximum probability of a document belonging to a category [22]. The probability of a document d belonging to a class c is given by Eq. 2.3.

$$P(c|d) \propto P(c) \prod_{1 \leq k \leq n_d} P(t_k|c) \tag{2.3}$$

Where $P(t_k|c)$ is the conditional probability of a term t_k occurring in a document of class c; n_d is the total number of terms in document d and $P(c)$ is the prior probability of a document occurring in class c [24].

MNB assumes that a document is defined by the BoW representation, meaning that what is relevant for this algorithm is the number of occurrences of each term, and not the order of appearance in a document; and that the probability of different features/terms are independent giving the class.

Similarly to kNN, this classifier tends to misclassify documents holding new features never seen before (i.e., in the training phase), because it fails to calculate the probability of the new features in the existing documents (zero probability).

Support Vector Machine SVM is a binary ML algorithm commonly used for text classification problems that aims to find the optimal hyperplane that separates two classes [8, 9, 12, 18, 25].

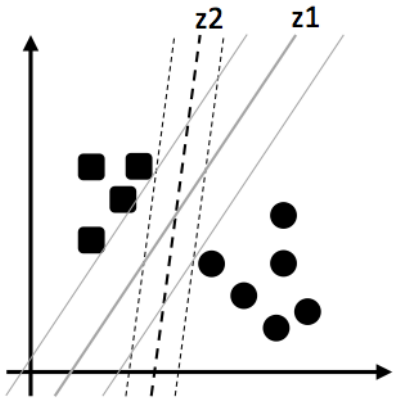

Figure 2.3: Example of a linear boundary. Adapted from Wikipedia: Support Vector Machine.

Figure 2.3 illustrates two classes: circles and squares, and two hyperplanes of separation: $z1$ and $z2$. A good separation is achieved by the hyperplane that has the largest width of a band around a decision boundary without any training samples, also known as, functional margin. In this case, $z1$ is the optimal hyperplane, because it separates correctly both classes and it has the maximum margin. A larger margin means a lower classifier generalization error [11].

Unlike other classifiers, SVM might not use all documents provided in the training phase, i.e., the margins are built only by documents near to the classification border; it can build an irregular border to separate positive and negative training documents; "not all the features (unique words) from the training documents are necessary for classification" [11, 26].

SVMs can efficiently perform linear and non-linear classifications using what is called the *kernel trick*, implicitly mapping their inputs into high-dimensional feature spaces, i.e., it converts not separable problem into a separable problem [11].

Figure 2.4 is an illustration of the mapping $\Phi(x)$ — on the left a set of samples in the input space, on the right the same samples in the feature space where a kernel Φ is applied.

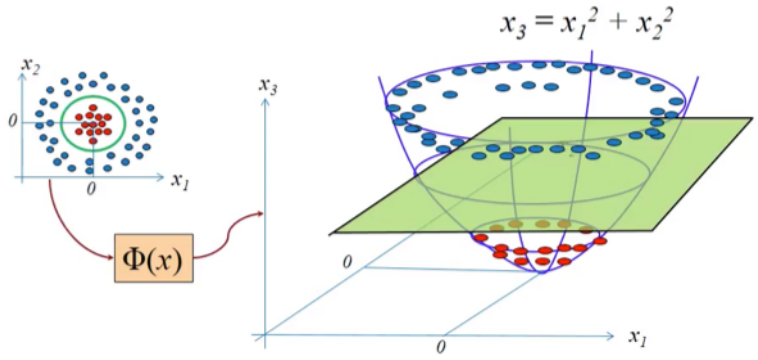

Figure 2.4: Example of a non-linear boundary. Adapted from YouTube channel: VisualPatterns, video: SVM Tutorial Part2.

RBF is widely used and it is assumed to be the most common used kernel in SVM. However, polynomial kernel is the most popular in NLP [27, 28]. This allows "feature expansion without loss of generality or an increase in computational costs" [27]. From literature we see that is also common to use degrees not higher than 3 in NLP, because this calculation "becomes not trivial, since the size of feature space exponentially increases." [27].

Despite SVM being a two class classier, it can be used for multi-class problems. The two most common approaches that problem are pairwise (1-vs-1) and 1-vs-all [29]. The former converts a multi-class problem into a series of two-class problems. One problem for each pair of classes, i.e., we have to train a separate classifier for each different pair of problems. Suppose that we have three classes: A, B and C. We train A against B, A against C, B against C, creating $\frac{N(N-1)}{2}$ classifiers. This approach is less sensitive to the problems of imbalanced datasets. In 1-vs-all, a class is trained against all the others at the same time. Taking in consideration the same three classes: A is trained against $\neg A$, B against $\neg B$ and C against $\neg C$.

This classifier is characterized by being fast in the testing phase and highly accurate even in multi class problems. The major drawbacks are the long training phase (this classifier is highly dependent on the number of training examples and number of features); and the difficulty of picking an adequate kernel.

Fuzzy Fingerprints FFP is a machine learning algorithm originally used to text authorship that has an analogy with human fingerprints [12,30]. Each human has its own fingerprints, so does a set of texts from the same author. As the author state: "The main concept behind this algorithm is that authors have a stable enough behavior that allows a set of features to be extracted, fuzzified and then compared." [30], i.e., the most relevant features, such as words and punctuation, of each author are gathered from a set of texts, and through a fuzzifing function, a unique fingerprint is generated. In order to detect the authorship of a text, a fingerprint is created based on text's features and then compared to authors'

fingerprints employing a similarity measure.

Despite being originally used for text authorship, the approach followed by this classifier proved to be effective and efficient in topic detection as well. Two adaptations for topic detection based in short texts have been made in [11] and [12], outperforming the most common text classification classifiers, such as kNN, SVM, MNB. Given our corpus, the most suitable and interesting approach is the latter.

Rosa et al. [12] proposed an adaptation of the original classification method that performs better in short texts, like tweets, when working with a high number of classes.

Due to the constrain of 140 characters in tweets, it was unreasonable to count the number of individual word occurrences. Therefore, "its features should be as unique as possible in order to make the fingerprints distinguishable amongst the various topics". To do that the author applied an adaptation of the popular Inverse Document Frequency (idf): Inverse Class Frequency (icf), "where topics are used instead of documents to distinguish the occurrence of common words" amongst topics [16].

$$\mathrm{icf_v} = \log\left(\frac{J}{j_v}\right) \tag{2.4}$$

In Eq. 2.4, J is the total number of topics and j_v stands for the number of #topics where the word v is present.

Due to the small size of a single tweet, a new scoring metric had to be developed, Tweet-Topic Similarity Score (T2S2).

$$\mathrm{T2S2}(\Phi, T) = \frac{\sum_v \mu_\Phi(v) : v \in (\Phi \cap T)}{\sum_{i=0}^{j} \mu_\Phi(w_i)} \tag{2.5}$$

T2S2 approach "tests how much a tweet fits to a given topic", taking into account the size of the tweet. It returns values between 0 and 1, where the lowest score indicate that the tweet in questions had no similarity to the topic fingerprint. As the authors enunciates "T2S2 divides the sum of the membership values $\mu_\Phi(v)$ of every word v that is common between the tweet and the #topic fingerprint, by the sum of the top j membership values in $\mu_\Phi(w_i)$, $w \in \Phi$.".

The area of text categorization is vast and it is impossible to cover all the different algorithms in detail [31]. Therefore we covered the most relevant for this work.

It is important to note that Neural Networks have proved to be an effective classifier in this area [31]. However these approaches usually require enormous quantity of data and time to train. Given the available data we discarded neural networks approaches.

2.1.3 Preprocessing Social Media Text Data

Text Normalization

The majority of available text data in social media is highly unstructured and noisy, due to informal language used for communicating through these platforms, and due to the occurrence of encoding errors, such as HyperText Markup Language (HTML) tags, scripts and advertisements [32]. Mining inherently unstructured text may lead to poor text analysis that affects the accuracy of an output [33], i.e., data needs to be prepared so that the information enclosed within is easily accessed by the mining tools.

The following techniques should be considered for cleaning and normalizing text retrieved from social media [12, 32, 34, 35]:

Escaping HTML Characters Text data obtained from web usually contains a large quota of HTML entities like **<** and **>** that gets embedded in the original data. It is generally advantageous to convert these entities into understandable symbols.

Example: **<** is converted to $<$ and **>** to $>$.

Removal Or Substitution of URLs, Emails and Twitter Usernames Tagging someone, sharing an URL or an email address is a recurring event in a social network, but usually these pieces of text do not provide any insight. Therefore, they are commonly removed or transformed into a dummy symbol.

Example:

"@username send the pictures to email@email.xyz"

1. "_twitter_name_ send the pictures to _email_address_" – Both Twitter's username and email are converted;

2. "send the pictures to" – Both Twitter's username and email are removed.

Decoding Data Text is often encoded in different formats. For an improved analysis, it is recommended to keep all the data in a standard encoding format.

Example: UFT-8 and Latin-1 are possible encodings for text data. To maintain text data uniform across all project, a default encoding should be chosen.

Removal of Stop-words Stop-words are the most frequent words[2] in a language and often do not carry much meaning. As we remove these words, it is easier to focus on the important words instead.

[2] E.g. Some stop-words in English: "the", "and".

Example: “I would like to buy two cokes.” once the stop-words are removed, the outcome is “would like buy two cokes.”.

Identify Emoji Emoji can provide useful information to a text. They underscore tone, introduce humor and display emotions. Despite most websites and instant messengers convert simple emoji made with punctuation (e.g. “:D”) into a yellow emoji face, some are not able to do it. It is essential to identify and translate these emoji into something that is easily identifiable.

Example: “:)” is transformed into a dummy symbol like “_happy_emoji_” and “:(” is converted into “_sad_emoji_”.

Removal of Punctuation Punctuation gives sense to the overall sentence construction and readability. Marks, such as full stop, comma, and brackets are used in writing to separate sentences and their elements, and to clarify meaning are generally missing, misplaced or used improperly in social media. Thus, discarding punctuation might have a positive impact in a study like this.

Example:

1. “Whose luggage is this!” – An exclamation mark is used instead of a question mark;
2. “Its yours” – It is missing an apostrophe and an full stop.

However, it should be done carefully because some punctuation marks might provide us some important insights, i.e, punctuation marks such as “?” might ease the identification of questions.

Split Attached Words When hashtags (e.g. #RainyDay) are used, occasionally there are two or more words without a blank space between them. Splitting connected words into several meaningful words improves the comprehension of the text.

Example: #RainyDay becomes Rainy Day.

Standardize Words Words can be written in a way to reinforce an idea or a sentiment (e.g. “I looooove you”). Although, they transmit emotions, the difference of using five or six straight “o” is insignificant. In the problem we are approaching, these instances should be standardized to have no more than two letters repeated, because there are no words in with 3 repeated letters in Portuguese.

Example:

1. “I looooooove you.” becomes “I loove you.”;
2. “I saw a reindeer.” remains the same.

Removal or Substitution of Numbers Different numbers frequently do not enrich the majority of researches but if transformed into dummy symbols they can enhance results [36].

If converting numbers into dummy symbols fail at improving results, numbers are removed from text. Example: “I paid 41€.” and “I want my 7$ back.” become “I paid _money_symbol_.” and “I want my _money_symbol_ back”.

Normalize Case Sometimes it is advantageous to convert all text to lower case to prevent counting words using different capitalization.

Example:

1. “I like to play football.”
2. “I Like To Play Football.”

Both sentences have the same words and the same meaning. However, a sensitive-case word counter counts the words: “play”: one time; and “Play”: one time, instead of “play”: two times.

Stemming This method is generally used to reduce the terms to their stems or root variant. It aims to group words with a similar basic meaning together.

Example:

1. “The doll is pink.”
2. “The dolls are pink.”

These terms are reduced to – “The doll is/are pink.”

Minimum Length of a Word A word must have at least N characters to be considered. If the word has less than the pre-set N value, it is removed.

Minimum Frequency of a Feature Each feature (e.g. word, symbol) must appear at least N times in a text/document.

Word Segmentation Separate a portion of continuous text into separate words. This means to fragment a text into a way that can be counted.
Example: “I love it!” will become “I”, “love”, “it” and “!”.

Feature Selection

Feature space in text documents consists of the unique terms that occur in documents, which can be in the order of hundreds or thousands for a simple corpus. Due to high dimensionality of feature space data retrieved from social media, it is difficult to identify the most relevant features for use in the model [37].

Feature selection is the process of selecting a subset of relevant features. Many of these features are either redundant or irrelevant. Thus, if removed, the prediction performance of the classifier boosts, becoming faster and more cost-effective, without incurring much loss of information.

Numerous experiments in text categorization have established that Information Gain (IG) and Chi-Square Static are considered the most effective feature selection methods, without losing categorization accuracy [38]. However, the former performs better in a dataset composed of short texts containing misspells, mistakes and a high feature space [39, 40]. This measures the number of bits required for category prediction by knowing the presence or the absence of a term in a document, i.e., IG tell us how much information about Y is contained in X [38].

Dataset Balance

A dataset is said to be unbalanced, when it contains much more samples from one class (majority class) than from the rest of the classes (minority classes).

Investigations about unbalanced datasets state that unbalance data can be are commonly found in real world application, and they can cause negative effects on classifications performance of ML algorithms [41–46]. Usually, ML algorithms tend to: (i) minimize the overall error to which minority classes contribute little, i.e., ML algorithms are accuracy driven; (ii) assume that errors from classes with little examples have the same cost as classes with significant amount of samples [47, 48]. Hence, the performance of prevailing classifiers favors the majority class.

Several approaches can be taken in consideration to overcome this problem: (i) select an adequate evaluation metric[3]. E.g. using F1 instead of Accuracy results in a more reliable evaluation metric; (ii) re-sample the dataset. E.g. over-sampling – adding copies of instances from the under-represented class, and under-sampling – deleting instances from the over-represented class; (iii) generate synthetic samples; (iv) use different algorithms. E.g. penalized classification - by imposing an additional cost on the model for making classifications mistakes on the minority class during training, the model is able to pay more attention to the minority class.

Ganganwar [47] studied the influence of unbalanced datasets on classification algorithms, such as kNN, MNB and SVM, reviewing briefly advisable manners to deal with the class-imbalance problem. From the experiments carried out the author stated that oversampling categories is the most effective

[3] Evaluation metrics are presented in detail in Section 2.1.4.

method to enhance the performance of local classifiers. On the contrary, when global learning classifiers are employed some under-sampling strategies surpass over-sampling. Thus, there is not a definite solution to this problem, depending greatly on the dataset.

2.1.4 Evaluation Metrics

To assess the classification of the ML algorithms we chose the well-know metrics recall, precision, accuracy and F1-score [17, 39].

Before explaining the formulas of the evaluation metrics, it is essential to explain the statistics concepts belonging to them [17, 49]:

True Positive (TP) – when an instance is correctly classified as true;

False Positive (FP) – when an instance is incorrectly classified as true;

True Negative (TN) – when an instance is correctly classified as false;

False Negative (FN) – when an instance is incorrectly classified as false.

Precision

$$P = \frac{TP}{TP + FP} \tag{2.6}$$

It presents the proportion of correct positive classifications from cases that are predicted as positives. This means, P is the percentage of selected items that are correctly classified (Eq. 2.6).

Recall

$$R = \frac{TP}{TP + FN} \tag{2.7}$$

It returns the proportion of correct positive classifications from cases that are positive. This means, R is the percentage of selected correct items (Eq. 2.7).

F1-Score

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{2.8}$$

F1-Score or F-Measure is a harmonic mean that combines precision and recall (Eq. 2.8).

Accuracy

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{2.9}$$

Accuracy is a statistical measure of how well a classification method correctly. It yields the proportion of correct classifications from overall number of cases (Eq. 2.9).

For real-life contexts, where datasets are commonly unbalanced, metrics like: Precision, Recall, F1-Score and Accuracy should be weighted according to the number of samples a class owns, i.e., a class with little examples should contribute less to the weighted average metric than a class with numerous samples. Weighting by class frequency presents a better estimate of overall performance, since the class frequencies can be very different, and a naive average calculation regards each class as equally important.

2.2 Related Work

An interesting survey [17] covering numerous short text classification approaches was released recently. This survey overviews numerous methods for short text classification, tackling traditional ML classifiers like kNN, NB, Maximum Entropy and SVM. The authors argue that these classic methods tend to underperform in comparison with: (i) Short Text Classification Based on Semantic Analysis; (ii) Semi-Supervised Short Text Classification; (iii) Ensemble Short Text Classification; and (iv) Real-time Classification of Large Scale Short Text. Despite a considerable number of studies suggesting that some of these classifiers perform very well in some datasets, many of these technologies are in the initial stage and some problems were found in the area of “dynamic short text stream, multi-label short text classification, comment emotional classification spam filtering, and topic tracking control.”.
Since, this survey covers well most of the state of the art techniques for short text classification, it might seem reasonable to referencing it and redirect the reader to this study.

3

Corpus

Contents

The following chapter describes the corpus used for the development of the project. Here, it is explained how data was retrieved, the challenges faced in the process and what was necessary to prepare the corpus for further use.

3.1 Data Retrieval

The starting point for any data preparation project is to successfully find and collect the data. The attempt to access data is not always an easy task, as complications arising from legal matter can often be encountered [50]. As a matter of fact, the legal aspect in the beginning of the project was a major challenge. The text data used to create the necessary models to classify text was collected from both public and private messages between the client and the company. Therefore, it was crucial to obtain a formal authorization from the company to access their data.

Due to particular events, promotions and seasons, the main focus of the messages arriving to TAP's social media platforms is constantly changin. For a better understanding of the TAP's social media reality, the company provided datasets ranging from late February 2016 until late November 2016, featuring the variation entailed by differences in high and low seasons in concomitance with big events occurring in the same time. E.g. (a) On the 2nd and 3th of July 2017, the Air Race Championship (ARC) took place in Lisbon, Portugal and several TAP's planes performed stunts. A direct correlation was: an increase of cheerful messages towards the company; (b) During June and July 2017, the Portuguese national team played at the 2016 Union of European Football Associations (UEFA) European Champion in France. When the competition finished, the Portuguese national team was supposed to fly back to Portugal in a TAP airplane. Instead, a Greek company transported the team. Hence, Portuguese people asked more questions to TAP than in a regular day.

Several sets of data were retrieved from TAP's Facebook and Twitter accounts, each of them combining comments, posts and messages from both TAP's customers and TAP's Contact Center operators. Using NEXT Analytics Dashboard Refresh Tool [1] and SONAR [2], it was possible to extract samples of public and private data flowing through these two social media platforms. SONAR was used at the beginning of the project to retrieve data, but soon the company stopped [3] using this tool to manage its social platforms and switched to a new. By the time, retrieved text data was not enough to train a classifier, thereby, it was necessary to find an auxiliary program to retrieve more examples - NEXT Analytics Dashboard Refresh Tool. In total, it was gathered more than 15 000 uncategorized and unstructured

[1] Developed by Next Analytics.
[2] Developed by Whale.
[3] TAP's decision.

comments, messages, posts and tweets from social platforms and then merged to create an appreciable corpus.

3.2 Data Sampling and Analysis

In order to demonstrate the faced challenges, Table 3.1 and 3.2 contain a few examples of the available raw data.

Date	origin	object_type	description	name
29/02/2016	FACEBOOK	COMMENT	weeeee!! estou lá	user1
29/02/2016	FACEBOOK	COMMENT	Se eu fosse rica, passavã a vida a viajar na TAP Portugal adoro viajar.amigos	user2
29/02/2016	FACEBOOK	MESSAGE	Guilherme, [redacted]	TAP PORTUGAL
29/02/2016	FACEBOOK	MESSAGE	gostaria de [redacted]	user3
29/02/2016	FACEBOOK	COMMENT	sempre a pensar em ti ?	user4
29/02/2016	FACEBOOK	COMMENT	Temos de falar com o Pedro Gordinho nos levar	user5
29/02/2016	FACEBOOK	MESSAGE	Muss ich [redacted]	user6

Table 3.1: A sample of data retrieved from TAP PORTUGAL's Facebook and Twitter account using Sonar.

ID	comment_ comment_id	created_ time	from/name	from/id	message	like_count	...
X1	Y1	11/11/2016 03:03	user1	Z1	CS-TOR 'Bartolomeu Dias' em uma tarde de sol em Recife.	9	...
X2	Y2	03/11/2016 15:11	user2	Z2	C'est pas assez 15 cm ?!? ;)	1	...
X3	Y3	25/10/2016 05:24	user3	Z3	Obrigada pelo atendimento	0	...
X4	Y3	25/10/2016 07:48	user4	Z4	Worst online booking and in person costumer service of any airline. Never again!	1	...

Table 3.2: A sample of data from TAP PORTUGAL's Facebook account using NEXT Analytics Dashboard Refresh Tool.

By analyzing Table 3.1 and 3.2, it becomes evident that these unstructured sets of texts are not categorized and they have text data in multiple languages (multilingual corpus). These tables have several fields, however, it is obvious that the relevant fields for the work are (i) *description* and *message*, and (ii) *name* and *from/name* — since a message is directly linked to a username, the fields *name* and *from/-name* allow us to identify irrelevant messages from TAP Contact Center operators.

Language Distribution It may be reasonable to suppose that Portuguese is the most common language in TAP social media accounts. To validate this assumption, we developed a program that links the text to a language. The program identifies the most spoken idioms on board of TAP flights, namely, English, French, German, Italian, Spanish and Portuguese. It uses stop-words and other relevant lexicon [4] of each language to recognize the one(s) present(s) in the corpus. In order to prevent improper identification of Portuguese texts, common words between Portuguese and the other languages were discarded and not used as key-factor, e.g., "no" and "a" are common in more than one language. Despite following a naive approach, it detected most of the cases correctly.

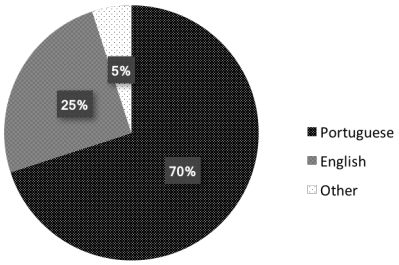

Figure 3.1: Language distribution across the dataset.

Figure 3.1 puts in evidence Portuguese as the most common language in the corpus, followed by English. These two languages take over 95% of the dataset. The remaining 5% is mostly French, Italian and Spanish. Considering that the goal is to gather as much information as possible from Portuguese texts, all the non-Portuguese texts are unheeded in the analysis.

Corpus Characteristics As mentioned before (Section 2.1.3), text found in social networks' conversations and discussions tends to be different from text in books and articles for several reasons. Analyzing the corpus, it was possible to characterize it as: (a) sparse and short — users tend to avoid type long messages, as the average text in the corpus has approximately 24 terms [5]; (b) non-standard — users do not to pose as much care into an accurate choice of words or in the structures of their sentences as they,

[4] List containing these words available in A.1.
[5] A term comprehend anything from a number to punctuation or a word.

ID		Original Sentence	Misspellings Found	Mistype Found
a	PT	Portunhol...essa língua k sem saber como tão bem domino ;-). Cobtem comigo, kd kerem partir?	Portunhol; k; kd; kerem	Cobtem
	EN	Portunhol...that language dat without knowing how I am so good at ;-). Coubt me in, wen do you want to leave?	Portunhol; dat; wen	Coubt
b	PT	É assim que voces tratam os voços clientes?	voços	
	EN	This is how you treat yor customers?	yor	
c	PT	nao estou disposto a pagar essa tarifa.	nao	
	EN	I am not willing to pay this rate.		

Table 3.3: Examples of misspellings and mistypings mistakes found in the corpus. (When counting the term frequency of the words of this dataset, the missing tilde in the word "não" is the most common orthographic mistake.)

probably, otherwise would, in other contexts (See Table 3.3). That may lead to different types of ambiguities, such as lexical, syntactic, and semantic [32]. Therefore, analyzing and extracting information patterns from such data sets is a complex task; (c) noisy and imbalanced — as it can be seen later on Section 3.6, most of the texts retrieved belong to a single class, which contains the irrelevant messages, i.e., the noisy messages; (d) multilingual corpus — 30% of the corpus was in English, French, Italian, German, Spanish, among others; (e) unlabeled;

Another particularity in social media texts are emoji. They add meaning and manifest emotions, enriching the text (as detailed in Table 3.4).

ID		Original Sentence	Emoji
a	PT	Gosto muito! :')	:')
	EN	I really like it! :')	
b	PT	Tenho tantas saudades!!!!!!♡	♡
	EN	I miss it so much!!!!!!♡	

Table 3.4: Examples of sentences with emoji found in the corpus. Emoji can be presented by a sequence of punctuation or by an icon.

3.3 Corpus Preprocessing

The built corpus contains texts considered irrelevant to achieve the goal of this work, such as, comments from TAP Contact Center and messages not in Portuguese. Table 3.1 and 3.2 show that a message is directly linked to a user, allowing to identify which messages are from TAP Contact Center and which messages are from users.

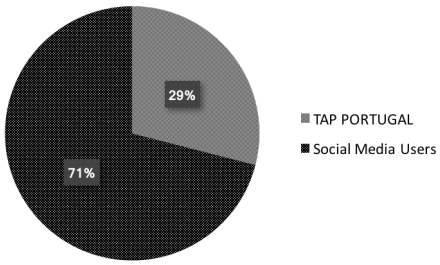

Figure 3.2: The chart demonstrates the comments, messages, posts and tweets percentages from TAP and its clients in the corpus.

Fig. 3.2 shows that considerable amount of the original corpus (29%) is written by the Contact Center operators. These comments, messages, posts and tweets are extraneous for the work because the model we wanted to build must contain only complaints, praises, questions and suggestions made by the costumers. Hence, the starting point of cleaning the corpus is to remove texts from TAP Contact Center.

The following step is to remove non-Portuguese text. As mentioned in Section 1.2 the model has to automatically classify complaints, praises, questions and suggestions in Portuguese. Therefore, texts in a language other than Portuguese are removed. Applying the same naive approach briefly explained in Section 3.2, a text is identified as "written in Portuguese" if it excludes words provided by NLTK [51] as stop-words of English, French, Italian, German or Spanish or term contained in the list A.1. Once again, as precaution, common words between Portuguese and other languages were excluded.

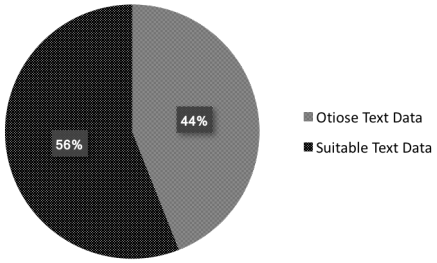

Figure 3.3: Distribution of usefulness of the original corpus.

As we can see from Fig. 3.3, a rather small corpus (approximately 8400 texts) remains to train the classifiers.

The following step is to normalize the remaining data. The baseline techniques [6] adopted to normalize the corpus were: removing Uniform Resource Locator (URL)s, email addresses, Twitter's usernames and capitalization; parsing HTML entities and standardizing emoji and words; tokenization and removal of diacritics - due to the complex orthography of Portuguese, which makes use of the Latin alphabet

[6] Techniques explained in Section 2.1.3.

and the acute accent, the circumflex accent, the grave accent, the tilde, and the cedilla, orthographic mistakes are common in social platforms. Removing diacritics reduces the percentage of errors found since none of the words will feature any of these special characters.

ID		Original Sentence	Preprocessed Sentence
a	PT	Para mais informações contate-nos em www.flytap.pt/faleconosco .	Para mais informacoes contate - nos em www . flytap . pt / faleconosco .
	EN	For more informations contact us at www.flytap.pt/faleconosco .	For more informations contact us at www . flytap . pt / faleconosco .
b	PT	Para mais informações contate-nos em www.flytap.pt/faleconosco .	Para mais informacoes contate - nos em .
	EN	For more informations contact us at www.flytap.pt/faleconosco .	For more informations contact us at .

Table 3.5: Result of applying baseline preprocessing techniques in different orders.

The order of preprocessing techniques execution is important. They must follow a specific order to guarantee the desired outcome. In this particular work, the order to follow must be:

1. Removing URLs
2. Removing email address
3. Removing Twitter's usernames
4. Converting the text to lowercase
5. Parsing HTML entities
6. Transforming emoji into dummy symbols
7. Standardizing words
8. Splitting entities by spaces or tokenization
9. Removing diacritics

Table 3.5 demonstrates that the same sentence leads to two different outcomes depending on the order of the techniques. In (a), the sequence detailed previously is not followed, resulting in an unsought outcome — the URL is not removed. This happens because tokenization is executed before the other operations. As consequence, it is impossible to correctly identify the URL and eventually to remove it. Contrasting with (b) that entails the desired result.

It should, however, be noted that due to the limitation of 140 characters in tweets, acronyms and abbreviations are commonly used in social networks [9]. E.g. "b4" is short for "before". In some studies it is used a customized stop-words list that identify and transforms acronyms in real words. Although we are dealing with social networks text and this is a common practice in this context, its presence is estimated to be under 4% in all corpus. Hence, it was decided to ignore it.

3.4 Categories

Once preprocessed, the next step is to manually categorize the comments, messages, posts and tweets into the appropriate class. The purpose of this process is to make every text correspond to a specific class, according to a so called single-label classification.

TAP defined as main categories: Complaints, Praises, Questions and Suggestions. However, whilst the work proceeded it was possible to perceive that some text suited better other categories. As result, we added the categories: Acknowledgments, Insults and Other.

Complaint When a client states that a service they experienced was unsatisfactory or unacceptable. E.g. “It is unacceptable that the cabin crew does not speak English.”.

Praise This label is applied when a customer writes a positive remark that emphasizes a service or the company quality. E.g. “It is a pleasure to fly with TAP. It is my 30th flight with you.”.

Question It is given a “question” label to a text that demands information related to any service provided by the company. E.g. “Can I use my miles to buy a flight from Lisbon, Portugal to Oporto, Portugal?”.

Suggestion Sometimes people want to active participate in a company’s future. An idea that can improve a service provided by the company or the creation of a new service it is labeled as “suggestion”. E.g. “TAP should consider providing XL seats.”.

Acknowledgement A quick look to the dataset allowed us to see that there were many messages acknowledging the information given by the Contact Center. As consequence, this category was created. E.g. “Thank you for the information.”.

Insult When someone express hatred to the company without further explanation. E.g. “Worst airline ever!”.

Other Every text that is in a language other than Portuguese or is unsuitable to any of the categories listed above, it is classified with this label. Since TAP’s social platforms are public, everyone can potentially publish a text of any nature or purpose (i.e. noise). E.g. “Last time I flew on a TAP’s plane was in 1999.”.

Multi-label Challenge

Occasionally, as seen in Table 3.6 - ID 16, it is possible that a comment fits more than just one category.

ID	Text in Portuguese	Text in English	Category
1	Adoro viajar com a TAP.	I love to travel with TAP.	Praises
2	Excelente serviço!	Excellent service!	Praises
3	Adoro!	I love it!	Other
4	Lindo.	Beautiful	Other
5	Não gosto da comida servida a bordo.	I do not like the food served on board.	Complaint
6	Os voos chegam sempre atrasados.	The flights are always late.	Complaint
7	Não gosto!	I do not like!	Other
8	Outra vez?!	Again?!	Other
9	A que horas parte o voo TP77 no dia 1 de Janeiro?	What is the departure time of the flight TP77 on the 1st of January?	Question
10	Como está o tempo em Lisboa?	How is the weather in Lisbon?	Other
11	Deviam voltar a meter o voo das 9h para Genébra.	You should put the 9h flight to Geneva again.	Suggestion
12	Ok. Obrigado.	Ok. Thank you.	Acknowledgment
13	Ok.	Ok.	Acknowledgment
14	Precisa de dinheiro? Ligue já para 9xxxxxxxx!	Do you need money? Call 9xxxxxxxx now!	Other
15	Gosto tanto de Lisboa.	I like Lisbon so much.	Other
16	A comida servida a bordo é boa. No entanto, os voos estão constantemente atrasados.	The food served on board is good. However, the flights are constantly late.	Complaint
17	A TAP devia ter vergonha.	TAP should be ashamed.	Insult

Table 3.6: Sample containing categorized texts.

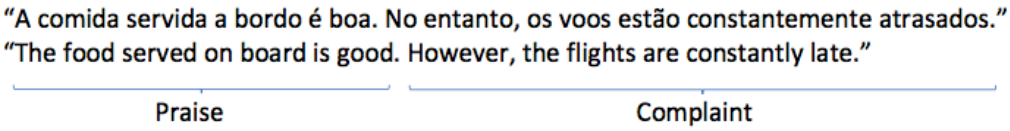

Figure 3.4: Example of an ambiguous classification text (Table 3.6, ID 16).

In Fig. 3.4 we present an example of a text where it is clear the presence of a praise and a complaint. This means that we are facing a multi-label classification problem.

25% of the total corpus was sampled and analyzed to gain an insight about the impact and frequency of this circumstance. The evidence points to the probability that its occurrence is under 3% over all the dataset, which is not substantial for further analysis.

There are certain rules that must be followed when a text fits more than one label. As the goal is to improve the satisfaction of the customers, it is essential to highlight and reply to any complaints and questions. Making these categories a priority to identify.

1. If a text contains several sentences and one expresses a complaint - it is classified as complaint.

2. If a text contains several sentences where none of them is a complaint, but it features at least one question - it is classified as question.

3. If a text contains several sentences where there is an acknowledgment and some other category x – it is classified as x.

3.5 Validation of the Dataset

The categorization of the built corpus was conducted by a single annotator. Therefore it was necessary to validate its quality. In order to assess the accuracy of the instructions to categorize a text [7] and to measure the reliability of the categorization process, it was asked to two students of Instituto Superior Técnico (IST) to manually classify several instances of the dataset collected.

Prior to categorization, it was given sufficient time for the students to carefully read section 3.4 and comprehend the concepts of each category. It is important to notice that before this experiment the students had no previous contact with the project nor they had experience classifying reviews.

The dataset granted consisted of 104 non-preprocessed Portuguese texts from TAP's clients randomly selected. The students carried out the task independently and at different times.

Krippendorff's α	**N. annotators**	**N. cases**
0.664	3	104

Table 3.7: Annotation score — Krippendorff's α.

Krippendorff's α has proved to be a reliable measure of content analysis, outperforming other statistics [8] independently of the number of annotators and the size of the dataset [52]. Therefore, Krippendorff's α was the measure of content analysis selected to assess the quality of the annotated data. Values close to 1 indicate the corpus is identically classified amongst annotators, while values close to 0 point out the opposite. In this case, it was obtained a Krippendorff's α of 0.664, i.e., there were some inconsistencies, but overall a considerable amount of the classification was identical.

Inspecting carefully the classification of the annotators, the prominent challenge was to classify ironic comments, i.e., complaints masked as praises, and suggestions that can be classified as questions and vice versa.

3.6 Distribution of the Dataset

A dataset is said to be unbalanced when contains significantly more samples from one class than from the rest of the classes [50]. As mentioned in Section 2.1.3 and summarized in Figure 3.5, real world datasets are often unbalanced.

[7] Guidelines described in the previous section (Section 3.4).
[8] E.g., Fleiss' kappa; Spearman's rank correlation coefficient, among others.

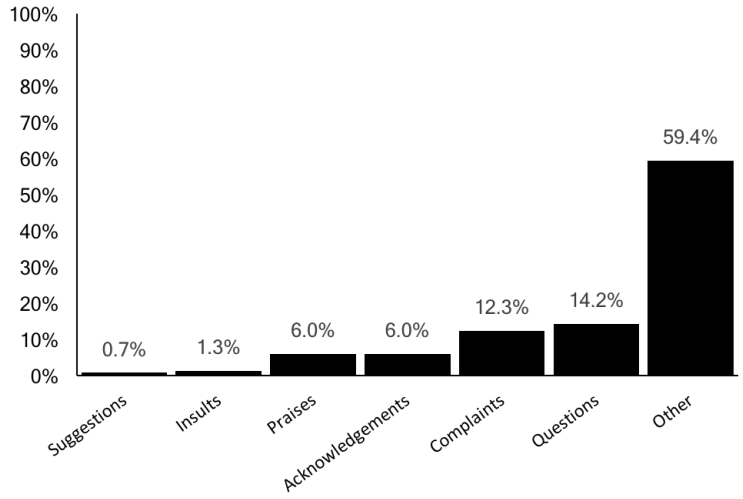

Figure 3.5: Distribution of clients' texts.

The chart above (Fig. 3.5) plots the distribution of the built corpus over the 7 classes. These divergent results suggest that we are facing an unbalanced dataset, where the category that owns gibberish text is clearly the majority class.

It is also important to notice that only 33.3% of the suitable text fits into categories established by the company. Although, if we consider the entire corpus, only 22.7% is relevant to the company — this proportion comes closer to the Pareto principle (80/20 rule).

4

Experiments and Results

Contents

This chapter summarizes results obtained and respective explanations. Section 4.1 describes the experimental setup, methodologies applied and tools. Section 4.2 presents the results obtained from ML experiments. Section 4.3 addresses knowledge retrieved from the corpus.

4.1 Experimental Setup

Studies about text mining, remark short text categorization as a vexed area mainly due to the challenges of: (i) gathering structured and annotated data, and (ii) choosing a robust classifier.

Gather good quality data is a true challenge, not only from technical or legal subtleties, but also from categorization itself. As seen in Section 3.5, even 3 annotators [1], with the same guidelines, labeled identical samples differently. It is important to reiterate that the built dataset contains approximately 8400 categorized short texts gathered between February and November 2016 in Portuguese [2]. Due to the limited amount of suitable data available, it seemed reasonable randomly split the dataset, considering 90% of the instances for training and the rest for testing.

Python [3] was the selected programming language to develop preprocessing scripts and FFP adapted to short texts. WEKA [4] was the main tool used to evaluate kNN, MNB and SVM performance. Both offer simple and efficient tools for data mining (including text classification), supported and used by academic communities worldwide.

4.1.1 Preprocessing

An effective text cleaning and normalization has a direct correlation with good results [50]. As mentioned earlier (Section 3.3), several preprocessing methods were applied to the corpus: removing URLs, Twitter's usernames, emails and diacritics, converting text to lowercase, parsing HTML entities, transforming emoji into dummy symbols, standardizing words and splitting entities by spaces [5].

In order to maximize the classification results of the prediction models and to complement the normalization techniques already detailed, more techniques were considered whilst developing this work.

Removal of Stop-words In an attempt to maximize classifiers' performance, we took two approaches on removing stop-words: (a) we removed the Portuguese stop-words provided by NLTK [51], and (b) we removed the Portuguese stop-words provided by NLTK [51] plus two words that frequently appear in the corpus: "tap" and "portugal".

[1] Two IST students and me.
[2] In detail in Section 3.
[3] Python 2.7
[4] WEKA 3.6.0
[5] From here on, for easier comprehension, the application of these methods is dubbed baseline preprocessing.

Removal of Punctuation Despite some studies supporting the removal of punctuation, it is not well established that this practice constantly enhances classifiers' behavior in text mining problems [11, 12, 53, 54]. We endeavor three approaches: (i) maintaining the original punctuation; (ii) partial removal, i.e., all the punctuation is removed, except question and exclamation marks, since it might provide a good lead to identify questions and praises; and (iii) removal of all punctuation.

Substitution or Removal of Numbers As reported previously in [36, 55, 56], replacing numbers for tags might improve the performance of classifiers. This technique might be relevant due to the appreciable amount of numbers in the corpus, namely flight numbers, reservation codes, departure and arrival times and hours of delay. Based on that reality, we tested the effectiveness of removing and replacing numbers for tags.

Type	Format	Replaced for
Date	11/3/2015 1/03/2011	TAGDATENUMBER
Delays	40min 1h	TAGDELAYNUMBER
Flight Number	TP77 TP 2013	TAGFLIGHTNUMBER
Departure / Arrival Hours	10:30 16h00m	TAGHOURNUMBER
Currency	203$ 2eur	TAGMONEYNUMBER

Table 4.1: Some examples of numbers transformation that are recognized and its respective transformations.

Remove Features with a Minimum j Frequency Studying the number of times a word occurs in the corpus might be of interest. A word that does not appear many times might be an orthographic mistake, misspell or a word that provides limited information to identify a class. By doing this, we are able to eliminate the problem of data sparseness. The downside it that, we might eliminate distinguishable features.

Remove Minimum k Sized Words Small words usually contain little meaning to the text. Considering a minimum word length might enhance the weight of lengthy and richer/meaningful words.

Remove Text Shorter than n Texts containing a small amount of features usually carry little information. Texts with one or two words might not contain enough content to identify its class.

Text Representation

It is known from the literature that BoW is the most used text representation and TFIDF the most used weighting scheme in text classification, making the most unique features of each document stand out. Therefore, this is the default format of text data representation selected for this work, except for FFP. This uses a variation of TFIDF — Term Frequency-Inverse Topic Frequency (TFITF).

Feature Selection

IG and Chi-Square are the most effective feature selection methods in text classification [39]. However, the former shows slightly better results in short texts categorization problems [40]. Hence, it was the only feature selection method considered in the experiments.

In order to select the IG threshold, we conducted several experiments with different values. We empirically selected 0.001 as the minimum value for a feature to be selected.

	Default Options
Baseline Preprocessing	Removing web content and diacritics; Lowering case; Replacing emoji for tags, Standardizing words; Tokenization
Information Gain	Threshold value = 0.001

Table 4.2: Default options for experiments.

Dataset Balance

As discussed previously, we were facing an unbalanced dataset, where the majority class held more than 50% of the instances. Taking in consideration the approaches detailed in Section 2.1.3, we undersampled the class "Other" removing randomly 70% of its instances. Figure 4.1 plots the new distribution.

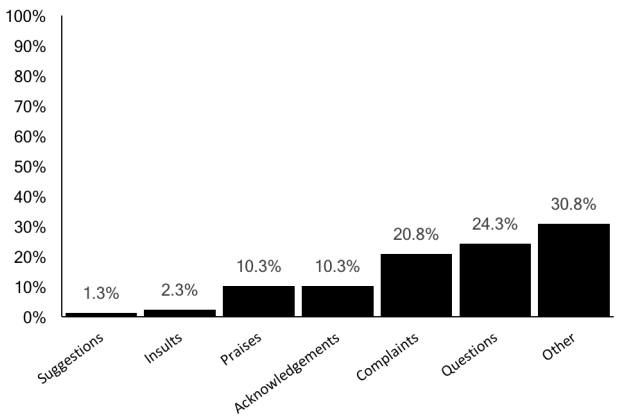

Figure 4.1: Distribution of clients' texts. Undersampled dataset.

Experiments revealed that balancing the dataset decreased the performance of the classifiers. Therefore, we focused our experiments on the original dataset.

4.1.2 Evaluation

The experiments using MNB, SVM and kNN classifiers were executed in WEKA. As mentioned previously, we split the dataset in: 90% for train and validation (for validation purposes we used stratified 10 fold cross validation); 10% for testing. On the other hand, FFP adapted to short texts used 90% of the data for training the model and 10% to test. No cross validation was performed. Both training and testing files were shared amongst experiments, making them comparable.

To assess the classification of the ML algorithms, we used the metrics Accuracy, F-Measure, Precision and Recall.

4.2 Experimental Results and Analysis

4.2.1 k Nearest Neighbors

kNN algorithm provided by WEKA was based in [57]. Due to the limitations of WEKA, the distance measure considered was the Euclidean distance as opposed to the more common measures like cosine similarity or inner product [58].

To begin with, it was important to understand the magnitude of neighbors to use. Since we were dealing with an unbalanced dataset, using a high k might lead the classifier to assign the label "Other" to all instances. Table 4.3 displays the obtained results of kNN, considering baseline preprocessing techniques and different values k.

The results displayed in Table 4.3 are in exceptionally good agreement with previous studies, that supported kNN as a great predictor. Assigning a text to its nearest neighbor ($k = 1$) achieved very good results but probably due to overfitting.

Higher values of k failed to achieve good results due to the unbalance nature of the dataset. Increasing the number of neighbors led kNN to classify more instances as "Other", since this was the majority class and it held more terms than any other class (See Table 4.6). Therefore, considering $k > 1$, kNN tended to find more similarities with this class than the others, being harder to correctly classify instances belonging to the other classes.

Further conclusions might be taken from Table 4.3 about kNN's behavior in this dataset. It can be reasoned out that employing IG and/or gathering more features per class, barely had impact on the classifier's performance. Because kNN considered only the most relevant features to identify a class rather than using them all, both practices had almost no effect.

kNN	Words To Keep	Feature Selection	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
1NN	1000	—	**88.70%**	**89.90%**	**88.90%**	**88.68%**
		InfoGain	**88.70%**	**89.90%**	**88.90%**	**88.68%**
	3000	—	88.60%	89.80%	**88.90%**	88.57%
		InfoGain	**88.70%**	**89.90%**	**88.90%**	**88.68%**
	5000	—	88.50%	89.70%	88.70%	88.46%
		InfoGain	88.60%	89.80%	88.90%	88.57%
	10000	—	88.40%	89.60%	88.60%	88.35%
		InfoGain	**88.70%**	**89.90%**	**88.90%**	**88.68%**
3NN	1000	—	77.50%	77.40%	73.20%	77.49%
		InfoGain	76.10%	75.20%	71.60%	76.14%
	3000	—	76.50%	76.70%	73.10%	76.48%
		InfoGain	77.00%	77.40%	72.70%	77.04%
	5000	—	76.10%	75.10%	71.90%	76.14%
		InfoGain	77.00%	77.30%	72.70%	77.04%
	10000	—	76.10%	75.20%	71.80%	76.14%
		InfoGain	77.70%	77.30%	72.70%	77.04%
5NN	1000	—	75.80%	77.10%	70.40%	75.81%
		InfoGain	75.90%	76.00%	70.80%	75.92%
	3000	—	75.70%	78.60%	70.30%	75.69%
		InfoGain	76.90%	78.50%	72.00%	76.93%
	5000	—	75.40%	78.40%	69.70%	75.36%
		InfoGain	76.70%	78.20%	71.70%	76.70%
	10000	—	75.40%	77.40%	69.80%	75.36%
		InfoGain	76.70%	78.20%	71.70%	76.70%
7NN	1000	—	73.90%	75.40%	66.80%	73.90%
		InfoGain	73.70%	73.40%	66.30%	73.68%
	3000	—	73.20%	77.30%	65.50%	73.23%
		InfoGain	73.30%	76.20%	65.60%	73.34%
	5000	—	73.30%	77.50%	65.90%	73.34%
		InfoGain	73.30%	76.20%	65.60%	73.34%
	10000	—	73.30%	77.70%	66.20%	73.34%
		InfoGain	73.30%	76.20%	65.60%	73.34%

Table 4.3: kNN performance employing baseline preprocessing.

Despite other values of k and $WordsToKeep$ presenting good results, they proved to be worse than $k = 1$ and $WordsToKeep = 1000$. Due to the good results summarized in Table 4.3 with these parameters, the experiments realized focused on $k = 1$ and $WordsToKeep = 1000$. With these settings we kept our model small and efficient. Larger values would only make the model larger and slower. IG was also excluded because it proved to be irrelevant to kNN in this dataset.

Considering our corpus, we assumed that removing punctuation marks, such as "?" and "!" would influence negatively the accuracy of ML algorithms correctly identifying questions and complaints. However, from the confusion matrices (See Table B.5 and Table B.3), we concluded that both punctuation marks had no influence on kNN accuracy on these two categories.

Despite Table B.1, B.2 and B.3 showing that neither removing stop-words, punctuation or numbers did not enhanced the classifiers' performance, when these 3 methods were combined they improved kNN prediction capabilities. Removing all NLTK [51] Portuguese stop-words, numbers and all the punctuation (except "!" and "?"), kNN accomplished scores of 89.10% for F-Measure, 88.90% for Recall, 90.10% for Precision and an Accuracy of 88.85%. Inspecting carefully the confusion matrices (See Table 4.5), we conclude these approaches slightly enhanced results for Praises, Insults and Questions. Additional results can be found at Appendix B.1.

kNN with $k = 1$ in general implies over-fitting. To verify that we conducted the same experiments in 67%/33% split and compared the results achieved. As we can see from the confusion matrices (Table 4.4), as we trained the classifier with more data, it overfitted to all categories enhancing the quality of the prediction model.

A	E	I	O	P	S	R	<– classified as
145	2	0	15	3	0	0	A (acknowledgments)
6	87	3	69	4	0	4	E (praises)
0	6	14	16	1	0	1	I (insults)
22	27	13	1495	22	0	18	O (others)
18	5	4	188	156	2	12	P (questions)
1	4	1	8	4	6	0	S (suggestions)
7	18	3	152	40	2	139	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	567	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table 4.4: Confusion matrices for 1NN considering the 1000 most relevant words employing baseline preprocessing. On the left, split 67% / 33%. On the right, split 90% / 10%. Accuracy and F-1 displayed in Fig 4.2.

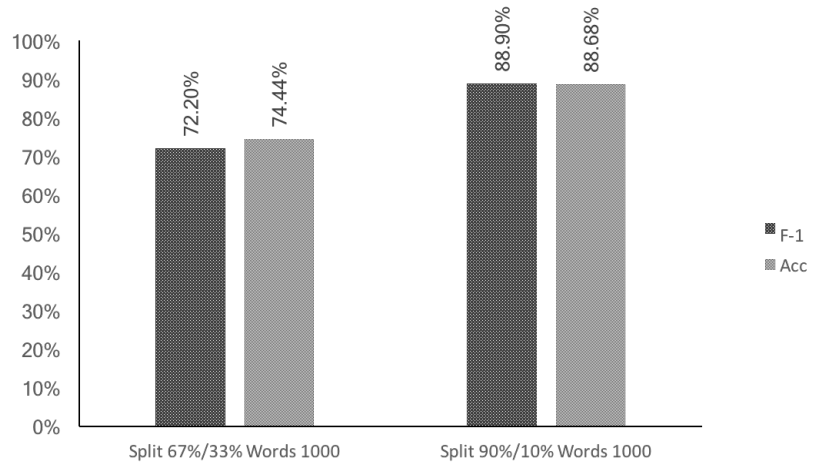

Figure 4.2: F1 and Accuracy for 1NN considering the 1000 most relevant words employing baseline preprocessing.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	93	0	10	0	0	10	E (praises)
0	0	3	1	0	0	1	I (insults)
1	12	0	565	3	2	24	O (others)
0	1	0	7	38	4	7	P (questions)
0	4	0	2	0	8	1	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	567	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table 4.5: Confusion matrices for 1NN considering the 1000 most relevant words. On the left we removed Portuguese stop-words, numbers and all the punctuation (except "!" and "?"); on the right we applied baseline preprocessing.

A	E	I	O	P	S	R	<– classified as
10	0	0	3	0	0	0	A (acknowledgments)
4	46	1	62	0	0	0	E (praises)
0	1	3	1	0	0	0	I (insults)
5	6	1	599	1	0	0	O (others)
3	2	0	40	11	1	0	P (questions)
0	1	0	14	0	0	0	S (suggestions)
3	3	2	60	2	0	8	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	567	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table 4.6: Confusion matrices for kNN considering the 1000 most relevant words employing baseline preprocessing. On the left, $k = 5$. On the right, $k = 1$.

4.2.2 Multinomial Naïve Bayes

It has long been known that MNB is the one of the most effective NB variants for text classification. In this work, we employed the WEKA implementation of this algorithm. In contrast to other classifiers tested, this lacked tuning parameters.

Normalization	Words To Keep	Feature Selection	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
False	1000	—	76.50%	80.70%	78.00%	76.48%
		InfoGain	76.60%	82.40%	78.50%	76.59%
	3000	—	80.00%	84.90%	81.20%	79.95%
		InfoGain	79.50%	82.40%	80.50%	79.50%
	5000	—	81.20%	85.20%	82.20%	81.18%
		InfoGain	80.00%	82.50%	80.80%	79.95%
	10000	—	**83.00%**	86.30%	**83.90%**	**82.97%**
		InfoGain	80.00%	82.50%	80.80%	79.95%
True	1000	—	75.00%	82.20%	77.20%	75.02%
		InfoGain	77.00%	82.10%	78.70%	77.04%
	3000	—	79.20%	85.40%	80.70%	79.17%
		InfoGain	79.20%	82.50%	80.30%	79.17%
	5000	—	82.60%	**87.20%**	83.70%	82.64%
		InfoGain	78.90%	82.20%	80.00%	78.94%
	10000	—	82.60%	86.50%	83.60%	82.64%
		InfoGain	79.70%	82.60%	80.60%	79.73%

Table 4.7: MNB performance applying baseline preprocessing techniques.

From Table 4.7 it can be seen that: (i) as expected, MNB increased performance as the number of features rose. Due to its nature, this classifier tends to perform better when it is trained with large amounts of data rich in features [22, 24, 59]; (ii) as the number of features increased we could have spared the usage of the feature selection tool. This was cause by the IG configuration used (only selects features with scores higher than 0.001), that led to a loss of features, which were crucial for MNB correctly classify instances; and finally (iii) normalizing all data to the average length of training instances appeared to have a positive impact in the performance of this classifier when $WordsToKeep > 3000$ and IG was discarded. Because MNB assumes that features are not intertwined, long texts might negatively effect parameter estimates due to their higher terms frequencies [24]. Normalizing the length of a text eliminates all information on the length of the original text, avoiding this problem.

The results summarized in Table 4.7 suggest that the most interesting cases to study MNB in this dataset was keeping the most 5000 and 10000 relevant features. Hence, the following results presented in this section took in consideration these values.

Words To Keep	Nor.	Removal Punctuation	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
5000	False	False	81.20%	85.20%	82.20%	81.18%
		True. Except "!" and "?"	81.50%	85.40%	82.50%	81.50%
		True	81.30%	85.20%	82.30%	81.26%
	True	False	82.60%	**87.20%**	83.70%	82.84%
		True. Except "!" and "?"	82.50%	87.10%	83.60%	82.51%
		True	82.30%	86.90%	83.40%	82.27%
10000	False	False	83.00%	86.30%	83.90%	82.97%
		True. Except "!" and "?"	81.10%	86.40%	84.00%	83.07%
		True	83.00%	86.30%	83.90%	82.95%
	True	False	82.60%	86.50%	83.60%	82.64%
		True. Except "!" and "?"	**83.70%**	**87.20%**	**84.60%**	**83.74%**
		True	82.70%	86.50%	83.70%	82.73%

Table 4.8: MNB performance when punctuation was removed.

Words To Keep	Normalization	Removal Stop-words	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
5000	False	False	81.20%	85.20%	82.20%	81.17%
		True. NLTK PT.	80.70%	85.20%	81.90%	80.73%
		True. Customized list.	80.90%	85.20%	82.00%	80.85%
	True	False	82.60%	**87.20%**	83.70%	82.64%
		True. NLTK PT.	82.10%	86.80%	83.20%	82.08%
		True. Customized list.	82.20%	86.80%	83.30%	82.19%
10000	False	False	83.00%	86.30%	83.90%	82.97%
		True. NLTK PT.	83.30%	86.40%	84.10%	83.31%
		True. Customized list.	**83.40%**	86.40%	**84.20%**	**83.42%**
	True	False	82.60%	86.50%	83.60%	82.64%
		True. NLTK PT.	82.10%	86.40%	83.20%	82.08%
		True. Customized list.	82.00%	86.40%	83.10%	81.97%

Table 4.9: MNB performance when stop-words were removed.

In spite of MNB performing better with large amounts of features, Table 4.8 and 4.9 suggest that some features, such as punctuation marks and stop-words might be, as expected, disadvantageous for this classifier.

The best result using MNB was obtained considering 10000 words when data was normalized to the average of the length of training corpus and in addition of the baseline preprocessing, the punctuation except "!" and "?" was removed ($R = 83.70\%, P = 87.20\%, F-1 = 84.60\%, Acc = 83.74\%$). The increase of 1% in F-1 and 1.10% in Accuracy derived mainly from the correct classification of "Others".

A	E	I	O	P	S	R	<– classified as
12	0	0	0	0	0	1	A (acknowledgments)
1	101	0	6	0	0	5	E (praises)
0	0	4	0	0	0	1	I (insults)
8	28	4	512	14	9	36	O (others)
0	1	0	4	40	3	9	P (questions)
0	5	0	2	0	6	2	S (suggestions)
0	0	0	3	2	1	72	R (complaints)

A	E	I	O	P	S	R	<– classified as
12	0	0	0	0	0	1	A (acknowledgments)
1	101	0	6	0	0	5	E (praises)
0	0	4	0	0	0	1	I (insults)
8	30	4	503	23	9	35	O (others)
0	1	0	4	40	3	9	P (questions)
0	5	0	2	0	6	2	S (suggestions)
0	0	0	3	2	1	72	R (complaints)

Table 4.10: Confusion matrices for MNB considering the 10000 most relevant words and normalized values. On the left removing all the punctuation except "?" and "!". On the right applying baseline preprocessing.

4.2.3 Support Vector Machine Performance

Previous studies refer that RBF kernel is one of the most used kernels when using SVM. However, in the majority of NLP, Polynomial kernel with low degrees achieve the best results [27, 28, 60]. These studies also state that lower degrees work better in NLP due to problems of overfitting, prolonged train times and excessive memory consumption. Therefore, for assessing SVM's performance categorizing texts in "Acknowledgments", "Complaints", "Insults", "Praises", "Suggestions", "Questions" or "Other", it was considered the degrees, $d = \{1, 2\}$, and Polynomial and RBF kernels. These were available on a WEKA implementation based on Sequential Minimal Optimization (SMO) [61–63]. By default, SMO normalized all attributes and it solved mutli-class problems using 1 vs 1 classification, i.e., it is created one SVM for each different pair of labels. This approach is less sensitive to the problems of imbalanced datasets.

Kernel	d	Words To Keep	Feature Selection	Wei. Avg. Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
RBF	1.0	1000	—	76.10%	77.10%	70.40%	76.14%
			IG	76.00%	77.10%	70.30%	76.03%
		3000	—	77.20%	78.20%	72.00%	77.15%
			IG	76.70%	77.60%	71.30%	77.70%
		5000	—	77.40%	78.50%	72.30%	77.37%
			IG	76.50%	77.50%	70.90%	76.48%
		10000	—	77.80%	78.80%	72.80%	77.82%
			IG	76.50%	77.50%	70.90%	76.48%
Poly	1.0	1000	—	87.60%	87.60%	87.20%	87.57%
			IG	86.30%	86.20%	85.70%	86.33%
		3000	—	88.80%	89.70%	88.80%	88.80%
			IG	87.20%	87.20%	86.80%	88.80%
		5000	—	**88.90%**	**89.90%**	**89.00%**	**88.91%**
			IG	87.00%	86.90%	86.50%	87.01%
		10000	—	87.60%	87.60%	87.20%	87.57%
			IG	87.00%	86.90%	86.50%	87.01%
	2.0	1000	—	87.70%	88.50%	87.60%	87.68%
			IG	87.60%	88.50%	87.40%	87.57%
		3000	—	88.50%	89.50%	88.60%	88.46%
			IG	88.20%	89.20%	88.20%	88.24%
		5000	—	88.10%	89.20%	88.30%	88.12%
			IG	88.10%	89.10%	88.00%	88.12%
		10000	—	88.20%	89.40%	88.40%	88.24%
			IG	88.10%	89.10%	88.00%	88.12%

Table 4.11: SVM performance employing baseline preprocessing methods.

Table 4.11 presents the performance of Polynomial and RBF kernels applying the baseline methods described previously. From the results displayed, several conclusions can be extracted: (i) it was evident that Polynomial kernel outperformed RBF, scoring almost 20% higher in F-1. Due to the nature of RBF kernel, this tends to overfit to the train dataset, making hard for the classifier to correctly classify text slightly different from what it was trained with (additional results presented in Appendix B.3); (ii) it was possible to compare the influence of applying IG on both kernels. Carefully inspecting the table, it was obvious that the best performances were achieved when discarding IG. As Hotho et al. stated in [2], SVMs rarely required feature selection as they "inherently select data points (the support vectors) required for a good classification´´; (iii) Table 4.11 provided compelling evidence that the degrees $d =$

Kernel	d	Words To Keep	Removal Stop-words	Wei. Avg. Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
Poly	1.0	3000	False	88.80%	89.70%	88.80%	88.80%
			True. NLTK PT.	88.90%	89.50%	88.90%	88.91%
			True. Customized list.	88.00%	88.60%	87.90%	88.01%
		5000	False	88.80%	89.70%	88.80%	88.80%
			True. NLTK PT.	89.00%	**90.00%**	**89.10%**	**89.02%**
			True. Customized list.	87.90%	88.60%	87.80%	87.90%
	2.0	3000	False	88.20%	89.20%	88.20%	88.24%
			True. NLTK PT.	87.10%	87.90%	86.80%	87.12%
			True. Customized list.	86.60%	87.30%	86.20%	86.56%
		5000	False	88.10%	89.20%	88.30%	88.12%
			True. NLTK PT.	**89.10%**	88.00%	88.10%	88.01%
			True. Customized list.	87.60%	88.60%	87.60%	87.57%

Table 4.12: SVM performance removing stop-words.

$\{1, 2\}$, of the Polynomial kernel show similarly satisfactory results in NLP.

Due to low performance of RBF kernel and the difference in results manifested by IG, these were discarded earlier in the development of the project. An appreciable number of experiments, combining different preprocessing techniques were realized in order to determine the most effective preprocessing methods and fine tuning of SVM for this problem. However, the best obtained result was achieved using Polynomial kernel ($d = 1$), by just removing NLTK Portuguese stop-words in addition of applying baseline preprocessing techniques — F-measure of 89.10% and 89.02% of Accuracy.

Generally, removing stop-words helps the classifier sticking to the richest words allowing for better results. Table 4.12 summarizes the obtained results when employing baseline preprocessing techniques followed by the removal of stop-words. Analyzing this table, we can see that removing stop-words did not have a significant impact in most situations. However we can notice that: (i) using a low degree, such as $d = 1$, independently of the number of words considered, the classifier accomplished slightly better results when removing NLTK stop-words; (ii) regarding a higher polynomial degree, the results suggested the opposite, i.e., when the text persisted with stop-words, the quality of the classifier increased. When removing stop-words the model contains the most exclusive words of each class to define each class boundary. As d increases boundaries become stricter, therefore less prompt to admit texts never seen before — overfitted model.

Kernel	Degree	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
Polynomial	1.0	**89.00%**	**90.00%**	**89.10%**	**89.02%**
RBF	1.0	77.80%	78.80%	72.80%	77.82%

Table 4.13: SVM best performance for each kernel.

Table 4.13 presents the best results obtained for each kernel. Kernels behaved at their best when different preprocessing techniques were applied to the corpus. RBF performed better when employing baseline preprocessing techniques (10000 words), and Polynomial kernel removing stop-words provided by NLTK [51] in addition to addressing baseline preprocessing techniques (5000 words).

To demonstrate that SVMs were the most robust model, we tested this classifier splitting the data in

67%/33% (Table 4.15).

A	E	I	O	P	S	R	<– classified as
11	0	0	2	0	0	0	A (acknowledgments)
4	16	0	85	0	0	8	E (praises)
0	0	0	5	0	0	0	I (insults)
3	0	0	602	0	0	7	O (others)
1	0	0	27	24	0	5	P (questions)
0	0	0	13	0	0	2	S (suggestions)
0	0	0	34	2	0	42	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	90	0	14	0	0	8	E (praises)
0	0	4	0	0	0	1	I (insults)
2	8	0	576	2	2	22	O (others)
0	0	0	9	38	3	7	P (questions)
0	4	0	2	0	6	3	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

Table 4.14: Confusion matrix of SVM best performances. RBF on the left and Polynomial on the right.

A	E	I	O	P	S	R	<– classified as
151	1	0	5	7	0	1	A (acknowledgments)
7	126	0	30	2	0	8	E (praises)
1	3	13	17	1	0	3	I (insults)
14	29	4	1490	36	0	24	O (others)
6	1	0	81	270	1	26	P (questions)
0	4	0	7	4	6	3	S (suggestions)
3	11	1	72	38	1	235	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	90	0	13	0	0	9	E (praises)
0	0	4	0	0	0	1	I (insults)
2	9	0	575	2	2	22	O (others)
0	0	0	9	38	3	7	P (questions)
0	4	0	2	0	6	3	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

Table 4.15: Confusion matrices for SVM, Polynomial kernel $d = 1$, considering the 5000 most relevant words employing baseline preprocessing. On the left, split 67% / 33%. On the right, split 90% / 10%. Accuracy and F1 displayed in Fig 4.3.

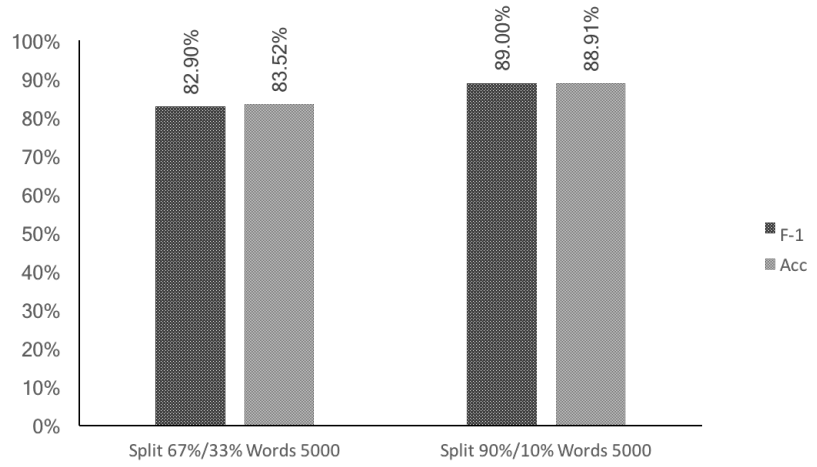

Figure 4.3: F1 and Accuracy for SVM, Polynomial kernel $d = 1$, considering the 5000 most relevant words employing baseline preprocessing.

4.2.4 Fuzzy Fingerprints Adapted to Short Texts

As described in Section 3, the dataset is mostly composed of short texts. For that reason, it was adopted the same approach as Rosa et. al. in [12] (detailed in Section 2.1.2). However, instead of detecting the topic of a tweet, among dozens of possible topics, we are doing topic classification.

Following the guidelines provided in that study, we replicated the code in Python, using auxiliary library such as NLTK [51], pandas [64], sklearn [65] and numpy [66]. For configuration purposes, the default values of $a = 0.2k$ and $b = 0.2$ were considered. These values proved to be the most effective in this classifier [30].

As it can be seen from Eq. 2.5, this algorithm's classification depends deeply from $T2S2$, which is the lowest threshold value for acceptance of a text belonging to a category. Through extensive testing it was found that 0.1 might be the best value for $T2S2$ for this dataset. Hence, it was assumed this value as the lowest threshold value for acceptance of a text belonging to a category.

To evaluate FFP adapted to short texts performance in this dataset, two approaches were followed:

1. we created a fuzzy fingerprint for every category;
2. we created a fuzzy fingerprint for every category, except for "Other". Since irrelevant texts were classified with this label, "Other" might not had a distinct fingerprint for itself. Therefore, all texts that failed to score a $T2S2$ above the defined threshold, were automatically classified as "Other", without being compared to others classes' fuzzy fingerprint. However, this approach demonstrated poorer results than the former. Despite "Other" being a sparse category, it was important to have a fuzzy fingerprint for itself.

k	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
5	53.97%	69.84%	59.97%	53.97%
10	53.19%	69.66%	59.96%	53.19%
20	61.59%	69.05%	64.68%	61.59%
50	67.07%	75.37%	70.41%	67.07%
75	71.66%	77.79%	74.92%	71.66%
100	73.79%	78.41%	75.33%	73.79%
125	75.47%	79.50%	76.65%	75.47%
150	74.02%	78.53%	75.53%	74.20%
175	75.81%	**80.78%**	**77.49%**	75.81%
200	76.14%	79.99%	77.45%	76.14%
250	75.92%	79.47%	77.17%	75.92%
300	76.25%	78.70%	77.01%	76.25%
500	**76.37%**	76.70%	76.37%	**76.37%**
1000	70.43%	69.30%	68.37%	70.43%

Table 4.16: FFP adapted to short texts' performance employing baseline preprocessing methods.

Another parameter that has a great impact in the classifier's behavior is the size of the fuzzy fingerprint, k. In order to determine the most adequate k value, the classifier was tested with several values k, $k = \{5, 10, 20, \ldots, 1000\}$, employing the baseline preprocessing techniques to the corpus. As it can be seen from Table 4.16, values ranging from 175 to 300 outperfomed other values of k.

Given the characteristics of short sized text, generating a fingerprint with a high k value resulted in large and sparse fuzzy fingerprints. Contrarily, a low k value generated rigid fuzzy fingerprints for each class. Therefore, the following results were based in the harmonious k value, 175, as this achieved the highest F-Measure in experiments.

A	E	I	O	P	S	R	<– classified as
8	0	1	2	0	2	0	A (acknowledgments)
8	41	20	19	2	10	13	E (praises)
0	1	1	2	0	1	0	I (insults)
10	32	24	471	20	27	28	O (others)
0	3	3	28	9	7	7	P (questions)
0	1	1	4	1	5	3	S (suggestions)
2	4	6	24	6	21	15	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
3	88	2	16	0	2	2	E (praises)
0	0	3	1	0	0	1	I (insults)
23	25	10	509	8	16	21	O (others)
6	2	0	14	24	4	7	P (questions)
0	3	1	7	0	3	1	S (suggestions)
5	7	4	15	2	2	37	R (complaints)

Table 4.17: Confusion matrices for FFP adapted to short texts employing baseline preprocessing. On the left we considered $k = 20$ and $k = 175$ on the right.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
10	74	12	15	0	0	2	E (praises)
0	0	4	0	0	0	1	I (insults)
74	29	40	413	27	5	24	O (others)
9	1	1	9	29	5	3	P (questions)
1	3	4	2	1	4	0	S (suggestions)
16	2	21	10	3	2	24	R (complaints)

Table 4.18: Confusion matrix for FFP adapted to short texts employing baseline preprocessing, $k = 1000$.

k	Removal Punctuation	Weighted Recall	Weighted Precision	Weighted F-Measure	Accuracy
175	False	**75.81%**	**80.78%**	**77.49%**	**75.81%**
	True. Except "!" and "?"	75.67%	80.09%	77.16%	75.67%
	True	74.15%	79.30%	75.94%	74.53%

Table 4.19: FFP adapted to short texts' performance removing punctuation.

From the confusion matrices (Tables 4.18 and 4.17) we see that the categories "Acknowledgments", "Complaints", "Praises", "Questions" and "Suggestions" were gripping the instances labeled as "Other". Which was the opposite behavior verified in the previous experiments (with kNN, SVM and MNB). In those experiments, "Other" attempted to absorb the instances of other categories.

As reported in [11, 30], maintaining punctuation might improve the classification system when using FFP. The results displayed in Table 4.19 were in conformity with these studies, as the highest values of Precision, Recall, F-measure and Accuracy were achieved when the punctuation remained in the text. Analyzing carefully this table, it can be inferred that "?" and "!" were the most relevant punctuation helping the algorithm correctly classify texts. Maintaining "!" and "?" just lowered F-1 by 0.33%, against the 1.55% of removing all the punctuation.

After employing different preprocessing techniques, excluding just stop-words from corpus was the most effective preprocessing method — $R = 79.28\%, P = 79.90\%, F - 1 = 79.17\%, Acc = 79.28\%$. To support this statement, it was evidenced in bold the highest scores in Table 4.20.

k	Removal Stop-words	Weighted Recall	Weighted Precision	Weighted F-Measure	Accuracy
175	False	75.81%	80.78%	77.49%	75.81%
	True. NLTK	78.72%	79.11%	78.46%	78.72%
	True. Customized	**79.28%**	**79.90%**	**79.17%**	**79.28%**

Table 4.20: FFP adapted to short texts' performance upon stop-words removal.

As mentioned before, removing stop-words bonds the classifier to the richest terms. Since it was defined a short value k to generate each class's fuzzy fingerprint, it was recommended to use wisely this limited space with the most unique terms. Excluding evasive words, like stop-words, "tap" and "portugal", it was granted to the classifier the most valuable terms, allowing FFP adapted to short texts to perform better. It is important to notice that feature selections methods were not applied, because FFP already selects the most important features.

Some reasons that might justify the lower performance are: (i) the limited number of training instances for certain categories; (ii) the short number of categories. Previous studies used large amounts of data to train the algorithm and a larger number of categories; (iii) our problem is different. We were facing topic classification while they were doing topic detection; (iv) the very short texts. We had a considerable amount of texts holding less than half a dozen features, which were misclassify (they were not classified as "Other"); and (v) we adapted the algorithm to the on-going problem.

4.2.5 Classifiers Performance Overview

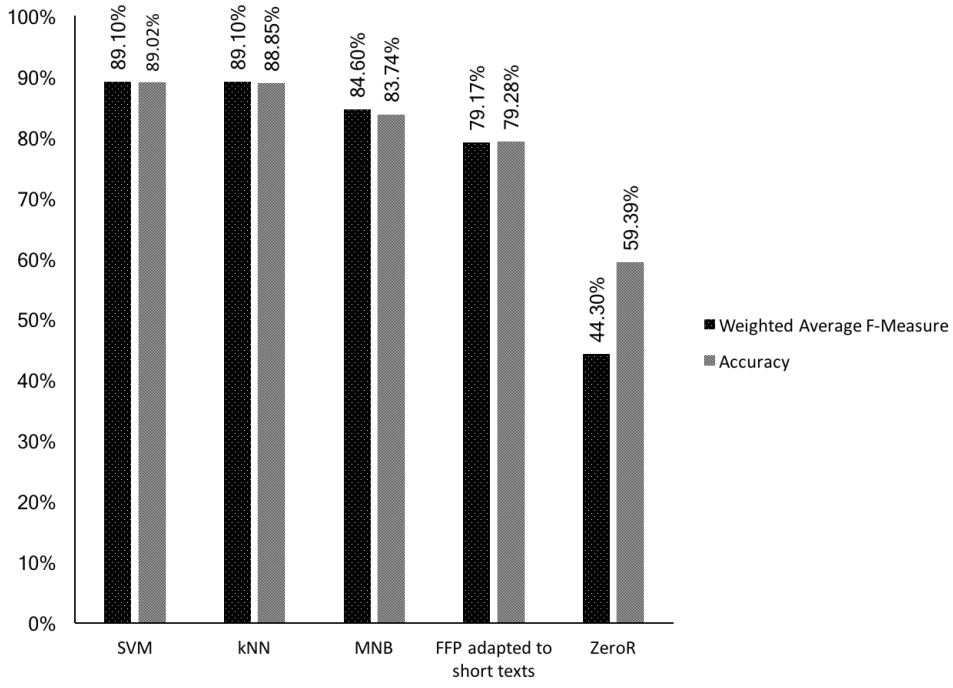

Figure 4.4: Overall best performances. Split 90% / 10 %.

One of the goals of the thesis was to study the behavior of some ML classifiers, presenting a viable solution for TAP. Figure 4.4 plots ZeroR[6] as a baseline and kNN, SVM, MNB, FFP adapted to short texts best Accuracy and F-Measure values. Despite being an unbalanced dataset, all the classifiers yielded

[6]ZeroR classifier is a commonly used baseline in NLP. This simply classify the instances to the majority class.

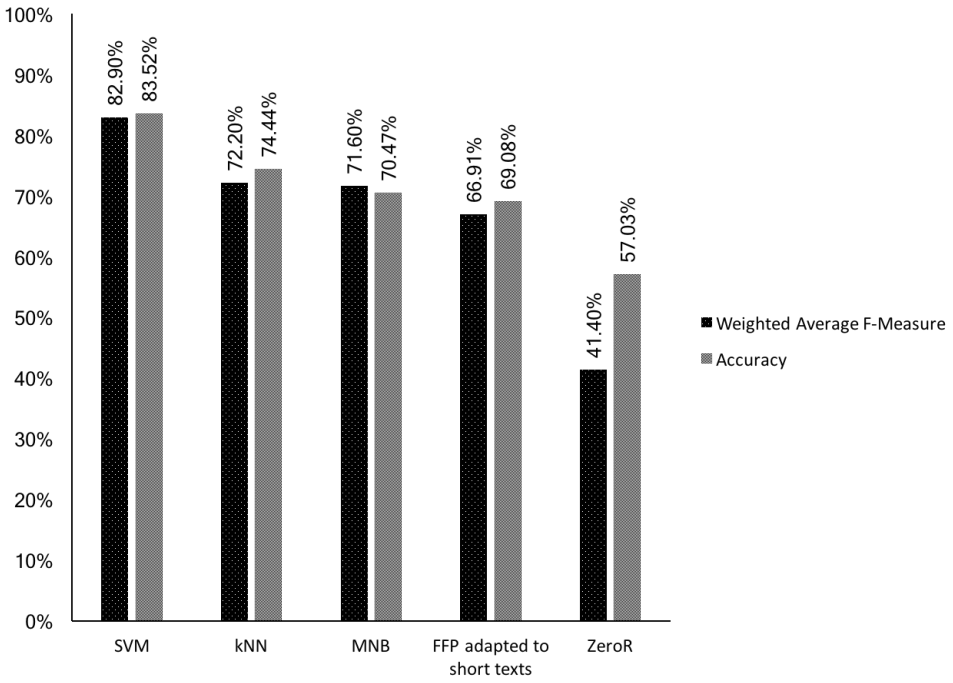

Figure 4.5: Overall performances in a 67% / 33 % split. We considered the settings applied in (Fig. 4.4).

good results. However, kNN and SVM stood out with very good performances, making these the best choices for the given dataset.

It has been long known that preprocessing and feature selection play a major role in classifiers behavior, and that became clear in Section 4.2. Despite IG yielding quite good results in ML algorithms such as kNN, SVM, MNB in previous studies [67, 68], the same did not happen in our study. Given the results, it is believed that TFIDF was sufficient to select the most relevant features.

Preprocessing is a task that should not be carried blindly. It depends greatly on the dataset and on the classier employed. The previous section summarized that results tend to change according to preprocessing methods. Also, some techniques proved to be more efficient than other in our study.

Rosa et. al [12] proposed an adaptation of Fuzzy Fingerprints for topic detection for on-the-fly processing of Big Data streams. In their study, FFP achieved outstanding results, outperforming kNN and SVM. However, this algorithm accomplished the lowest scores, which can be explained by the nature of our dataset; the difference in the problem; and the adaptation we made to deal with “Other” category. Despite obtaining worse results when comparing to kNN, MNB and SVM, this classifier training and testing time performance was several times better than any other.

MNB underperformance might be explained because: (i) kNN and SVM can be finer tuned; (ii) MNB might need more training instances.

As mentioned before, the outcome provided by both kNN and SVM were very good. However looking

carefully to the confusion matrices results (in Section 4.2.3 and B.3), these support that SVMs were the most robust classifier. Given different splits in training and test, SVM achieved the best scores. It is also important to notice that this classifier demonstrated to have the highest precision classifying "Complaints" — which is a priority to the company.

Charts displayed in Fig.4.5 and Fig.4.4 allow us to understand the impact of overfit in all classifiers. Clearly, SVM was the classifier that suffered less with this difference in the training set, and kNN the most. Supporting the hypothesis that kNN with $k = 1$ is proved to overfitting.

4.3 Dataset Retrieved Knowledge

In Section 3.4, we stated that the most interesting categories for TAP were: (a) complaints; (b) praises, (c) questions, and (d) suggestions.

A simplified manner to present and comprehend text data is through word clouds — a visual representation of text that stresses the most frequent words. For that reason, a word cloud was created for each of these categories (Figs.4.7, 4.7, 4.8, and 4.10).

Complaints

Regarding the class "Complaints", a great deal of intelligence can be glean analyzing its word cloud (Fig. 4.6). If we carefully inspect Figure 4.6, we see that most complaints are derived from: (i) long waiting times to get in touch with the contact center (given words such as "resposta", "telefone'); (ii) flight delays ("tagdelaynumber", "espera", "voo"); (iii) problems with luggage ("mala", "bagagem").

Figure 4.6: Word cloud for category complaints.

Whilst labeling data, it was possible to understand some patterns and acquire knowledge that can be advantageous to the analysis process. For example, the words "lisboa" and "porto" might not suggest anything to the reader, but for us it is clear that these words refer to the clients' complaints about the poor quality of the connection flights between Lisbon and Oporto.

Praises

In line with Praises' word cloud (Fig, 4.7), most praises derived from sympathy of the cabin crew and the feeling of security when flying with TAP. Contrasting with data retrieved from complaints, clients tended to appreciate the all service provided by the company, avoiding specifying any kind of service.

Figure 4.7: Word cloud for category praises.

Questions

Figure 4.8: Word cloud for category questions.

"Questions" is the richest category this corpus has. From the knowledge acquired while labeling the dataset and from this word cloud (Fig. 4.8), we can state that most questions are related to: (a) miles ("milhas", "victoria"); (b) booking ("reserva", "passagem", "passagens"); (c) passports ("passaporte");

(d) connection flight between Lisbon and Oporto (“lisboa”, “porto”); (e) luggage (“bagagem”, “mala”, “porao”); (f) difficulties using the website (“site”) and (g) delays (“tarde”, “taghournumber”). Even though, most of the questions might be answered in TAP website —www.flytap.pt, most clients go to social networks.

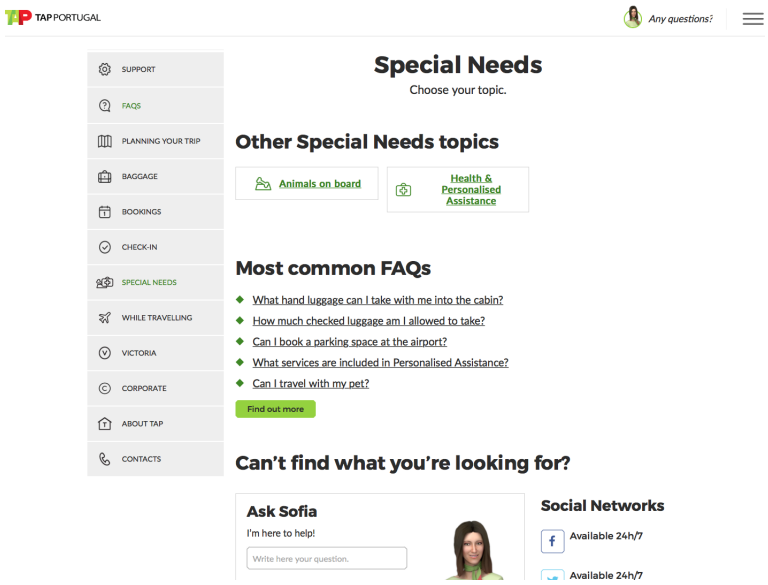

Figure 4.9: TAP Most Common FAQs. Last accessed March 2017.

As shown in Fig. 4.9, TAP website has a page dedicated to Frequently Asked Questions (FAQ)s, containing the answers to the majority of questions a costumer might have. However, FAQs webpage’s content was not in conformity with the most frequently asked questions in TAP’s Facebook and Twitter accounts. For example, no answer related to the keyword “passport” was found. For example a frequent question containing this keyword was: “I am Brazilian and I want to buy a plane ticket from Brazil to Portugal for next summer. However, I can not do it because the website is asking for a passport number which I do not have yet. Let me know how should I proceed.”

Suggestions

Due to the limited number of instances collected, it was evident that “Suggestions” was the trickiest category to retrieve valuable knowledge from. In Figure 4.10, words like: “lisboa”, “porto“and “taghour-number” stand out from the rest, supporting again the theory that this connection between these two cities has problems, and there is room for improvement.

Figure 4.10: Word cloud for category suggestions.

5

Conclusion and Future Work

Contents

This chapter summarizes key findings of the research and outlines possible future work.

5.1 Conclusions, Accomplishments and Future Work

The rapid dissemination of information and ideas, in combination with the rise of the development of instant communication, lead to an accelerated growth of short texts. Based on the fact that short texts might enclose valuable intelligence, mining these sources has become of increased interest for corporations. As previously outlined in the introduction (Section 1), costumers' feedback is of great importance and it allows corporations to meet the costumers' needs. Thus, it is essential to fully understand and take the costumers' perspectives in consideration. Since satisfied clients have higher chances to return and they tend to influence the ones around them to do the same purchase, if pleasant, as they once made.

It was in the interest of TAP to study ML algorithms that automatically identify complaints, praises, questions and suggestions in its social media platforms. So the company can understand the most frequent complaints, praises, questions and suggestions made by its clients.

From TAP's Facebook and Twitter accounts were retrieved several sets of data, each of them combining comments, posts and messages from both TAP 's customers and TAP's Contact Center operators. In total, it was gathered more than 15 000 uncategorized and unstructured comments, messages, posts and tweets from social platforms and then merged to create a corpus. Analyzing the gathered corpus, it was possible to characterize it as: (a) sparse and short; (b) non-standard; (c) noisy and imbalanced; (d) multilingual corpus; (e) unlabeled. Once the corpus was preprocessed, it was labeled in Acknowledgments, Complaints, Insults, Praises, Questions, Suggestions, and Other (gibberish texts). TAP defined as main categories: Complaints, Praises, Questions, Suggestions. However, whilst the work proceeded it was possible to understand that some text suited better other categories. As result, three categories were added: Acknowledgments, Insults and Other.

The conducted experiments pinpointed kNN and SVM as the best classifiers among the ones studied, achieving F1 scores higher than 88% in certain conditions. Despite the good scores achieved by both classifiers, SVM is clearly the most robust model — presenting very good results when reducing the training data as well. It was also possible to confirm that some preprocessing methods and supervised ML algorithms applied in text categorization might not be so effective in short text classification.

From the annotated corpus and acquired knowledge of labeling it, we retrieved and analyzed predominant complaints, praises, questions and suggestions.

The work results were presented to TAP on late March 2017. It was concluded that we successfully accomplished the objective, and due to the results, a classification model will be in production soon.

The next steps are defined as follows: (i) asking a professional to label more data — in order to build a more reliable model; (ii) developing two more classification models: one for identifying the types of complaints and another to denote the most common suggestions; and (iii) studying sentiment analysis.

Bibliography

[1] F. Lang, "The Importance of Feedback — Why Is Feedback Important? – Full Circle Feedback," 2012, Accessed: 2017-03-21. [Online]. Available: http://www.fullcirclefeedback.com.au/360-power-of/

[2] A. Hotho, A. Nürnberger, and G. Paaß, "A Brief Survey of Text Mining," *LDV Forum - Gesellschaft für Linguistische Datenverarbeitung Journal for Computational Linguistics and Language Technology*, vol. 20, 2005.

[3] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text Classification Using Machine Learning Techniques," *World Scientific and Engineering Academy Society Transactions on Computers*, vol. 4, 2005.

[4] N. Holdings, "State of the Media: The Social Media," 2012.

[5] S. S. Khan and M. G. Madden, "One-Class Classification: Taxonomy of Study and Review of Techniques," *The Knowledge Engineering Review*, vol. 29, 2014.

[6] G. Miner, J. Elder, T. Hill, R. Nisbet, D. Delen, and A. Fast, *Practical Text Mining and Statistical Analysis for Non-structured Text Data Applications*. Academic Press, 2012.

[7] F. Batista and R. Ribeiro, "Sentiment analysis and topic classification based on binary maximum entropy classifiers," *Procesamiento del Lenguaje Natural*, 2013.

[8] S. Wang and C. D. Manning, "Baselines and Bigrams: Simple, Good Sentiment and Topic Classification," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, vol. 2, 2012.

[9] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary, "Twitter Trending Topic Classification," in *Proceedings of the Institute of Electrical and Electronics Engineers International Conference on Data Mining Workshops*, 2011.

[10] T. Dunning, "Statistical Identification of Language," Computing Research Laboratory New Mexico State University, Tech. Rep., 1994.

[11] F. Batista and J. P. Carvalho, “Text based Classification of Companies in CrunchBase,” in *Proceedings of the Institute of Electrical and Electronics Engineers International Conference on Fuzzy Systems*, 2015.

[12] H. Rosa, F. Batista, and J. P. Carvalho, “Twitter Topic Fuzzy Fingerprints,” in *Proceedings of the Institute of Electrical and Electronics Engineers International Conference on Fuzzy Systems*, 2014.

[13] G. Salton and C. Buckley, “Term Weighting Approaches in Automatic Text Retrieval,” Cornell University, Tech. Rep., 1987.

[14] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, C. Potts *et al.*, “Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank,” in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2013.

[15] J. Beel, S. Langer, and B. Gipp, “TF-IDuF: A Novel Term-Weighting Sheme for User Modeling based on Users’ Personal Document Collections,” in *Proceedings of the iConference*, 2017.

[16] J. P. Carvalho, H. Rosa, and F. Batista, “Detecting relevant tweets in very large tweet collections: the London Riots case study,” in *Proceedings of the Institute of Electrical and Electronics Engineers International Conference on Fuzzy Systems*, 2017.

[17] G. Song, Y. Ye, X. Du, X. Huang, and S. Bie, “Short Text Classification: A Survey,” *Journal of Multimedia*, vol. 9, 2014.

[18] A. Cardoso-Cachopo and A. L. Oliveira, “An Empirical Comparison of Text Categorization Methods,” in *Proceedings of the International Symposium on String Processing and Information Retrieval*, 2003.

[19] T. Joachims, “Text Categorization with Support Vector Machines: Learning with Many Relevant Features,” *Machine learning: The European Conference on Machine Learning-98*, 1998.

[20] G. Batista and D. F. Silva, “How k-Nearest Neighbor Parameters Affect its Performance,” *Argentine Symposium on Artificial Intelligence*, 2009.

[21] R. Feldman and J. Sanger, *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, 2007.

[22] A. McCallum, K. Nigam *et al.*, “A Comparison of Event Models for Naive Bayes text classification,” in *Proceedings of the Association for the Advancement of Artificial Intelligence: Workshop on Learning for Text Categorization*, vol. 752, 1998.

[23] E. Frank and R. R. Bouckaert, “Naive bayes for text classification with unbalanced classes,” in *Proceedings of the European Conference on Principle and Practice of Knowledge Discovery in Databases*, 2006.

[24] J. D. M. Rennie, L. Shih, J. Teevan, and D. R. Karger, “Tackling the Poor Assumptions of Naive Bayes Text Classifiers,” in *Proceedings of the International Conference on Machine Learning*, 2003.

[25] V. N. Vapnik and V. Vapnik, *Statistical learning theory*. Wiley New York, 1998, vol. 1.

[26] M. Konchady, *Text Mining Application Programming*. Charles River Media, 2006.

[27] T. Kudo and Y. Matsumoto, “Fast Methods for Kernel-based Text Analysis,” in *Proceedings of the Annual Meeting on Association for Computational Linguistics*, vol. 1, 2003.

[28] Y. wen Chang, C. jui Hsieh, K. wei Chang, M. Ringgaard, C. jen Lin, and S. Keerthi, “Training and Testing Low-degree Polynomial Data Mappings via Linear SVM,” *Journal of Machine Learning Research*, 2010.

[29] S.-H. Park and J. Fürnkranz, *Efficient Pairwise Classification*, 2007.

[30] N. Homem and J. P. Carvalho, “Authorship Identification and Author “Fingerprints”,” in *Proceedings of the Annual Meeting of the North American Fuzzy Information Processing Society*, 2011.

[31] C. C. Aggarwal and C. X. Zhai, “A Survey of Text Classification Algorithms,” *Mining Text Data*, 2012.

[32] L. Sorensen, “User Managed Trust in Social Networking - Comparing Facebook, MySpace and Linkedin,” in *Proceedings of the International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace Electronic Systems Technology*, 2009.

[33] G. Forman and E. Kirshenbaum, “Extremely Fast Text Feature Extraction for Classification and Indexing,” in *Proceedings of the Association for Computing Machinery Conference on Information and Knowledge Management*, 2008.

[34] J. P. Carvalho and S. Curto, “Fuzzy Preprocessing of Medical Text Annotations of Intensive Care Units Patients,” in *Proceedings of the Institute of Electrical and Electronics Engineers Conference on Norbert Wiener in the 21st Century*, 2014.

[35] E. Clark and K. Araki, “Text Normalization in Social Media: Progress, Problems and Applications for a Pre-Processing System of Casual English,” *Pacific Association for Computational Linguistics*, vol. 27, 2011.

[36] R. Ghani, K. Probst, Y. Liu, M. Krema, and A. Fano, “Text Mining for Product Attribute Extraction,” *Association for Computing Machinery’s Special Interest Group on Knowledge Discovery and Data Mining Explorations Newsletter*, vol. 8, 2006.

[37] S. R. Singh, H. A. Murthy, and T. A. Gonsalves, “Feature Selection for Text Classification Based on Gini Coefficient of Inequality,” *Journal of Machine Learning Research*, vol. 10, 2010.

[38] Y. Yang and J. O. Pedersen, “A Comparative Study on Feature Selection in Text Categorization,” in *Proceedings of the International Conference on Machine Learning*, vol. 97, 1997.

[39] G. Forman, “An Extensive Empirical Study of Feature Selection Metrics for Text Classification,” *Journal of Machine Learning Research*, vol. 3, 2003.

[40] J. M. Gómez Hidalgo, G. C. Bringas, E. P. Sánz, and F. C. García, “Content Based SMS Spam Filtering,” in *Proceedings of the Association for Computing Machinery Symposium on Document Engineering*, 2006.

[41] X.-w. Chen, B. Gerlach, and D. Casasent, “Pruning Support Vectors for Imbalanced Data Classification,” in *Proceedings of the International Joint Conference on Neural Networks*, vol. 3, 2005.

[42] J. Wang, M. Xu, H. Wang, and J. Zhang, “Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding,” in *Proceedings of the International Conference on Signal Processing*, vol. 3, 2006.

[43] C. Chen, A. Liaw, and L. Breiman, “Using Random Forest to Learn Imbalanced Data,” 2004.

[44] X. Hong, S. Chen, and C. J. Harris, “A Kernel-Based Two-Class Classifier for Imbalanced Data Sets,” *Institute of Electrical and Electronics Engineers Transactions on Neural Networks*, vol. 18, 2007.

[45] Y. Tang, Y.-Q. Zhang, N. V. Chawla, and S. Krasser, “SVMs Modeling for Highly Imbalanced Classification,” *Institute of Electrical and Electronics Engineers Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, 2009.

[46] D. P. Williams, V. Myers, and M. S. Silvious, “Mine Classification With Imbalanced Data,” *Institute of Electrical and Electronics Engineers Geoscience and Remote Sensing Letters*, vol. 6, 2009.

[47] V. Ganganwar, “An Overview of Classification Algorithms for Imbalanced Datasets,” *International Journal of Emerging Technology and Advanced Engineering*, vol. 2, 2012.

[48] M. Kumar and H. Sheshadri, “On the Classification of Imbalanced Datasets,” *International Journal of Computer Applications*, vol. 44, 2012.

[49] D. M. Powers, “Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness and Correlation,” School of Informatics and Engineering Flinders University of South Australia, Tech. Rep. SIE-07-001, 2011.

[50] S. Zhang, C. Zhang, and Q. Yang, “Data Preparation for Data Mining,” *Applied Artificial Intelligence*, vol. 17, 2003.

[51] S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. ”O’Reilly Media, Inc.”, 2009.

[52] A. F. Hayes and K. Krippendorff, “Answering the Call for a Standard Reliability Measure for Coding Data,” *Communication Methods and Measures*, vol. 1, 2007.

[53] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text Classification Using String Kernels,” *Journal of Machine Learning Research*, vol. 2, 2002.

[54] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, “Sentiment in Short Strength Detection Informal Text,” *Journal of the Association for Information Science and Technology*, vol. 61, 2010.

[55] D. Hovy, “Demographic Factors Improve Classification Performance,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*, 2015.

[56] R. Bunescu, R. Ge, R. J. Kate, E. M. Marcotte, R. J. Mooney, A. K. Ramani, and Y. W. Wong, “Comparative Experiments on Learning Information Extractors for Proteins and their Interactions,” *Artificial Intelligence in Medicine*, vol. 33, 2005.

[57] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” *Machine Learning*, vol. 6, 1991.

[58] X. Qi and B. D. Davison, “Web Page Classification: Features and algorithms,” *Association for Computing Machinery Computing Surveys*, vol. 41, 2009.

[59] A. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, “Multinomial Naive Bayes for Text Categorization Revisited,” in *Proceedings of the Australian Joint Conference on Advances in Artificial Intelligence*, 2004.

[60] Y. Goldberg and M. Elhadad, “splitSVM: Fast, Space-efficient, Non-heuristic, Polynomial Kernel Computation for NLP Applications,” in *Proceedings of the Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, 2008.

[61] J. C. Platt, “Advances in kernel methods,” in *Fast Training of Support Vector Machines Using Sequential Minimal Optimization*, B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. MIT Press, 1999.

[62] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy, “Improvements to Platt’s SMO Algorithm for SVM Classifier Design,” *Neural Computation*, 2001.

[63] T. Hastie and R. Tibshirani, “Classification by Pairwise Coupling,” in *Proceedings of the Conference on Advances in Neural Information Processing Systems 10*, 1998.

[64] W. McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the Python in Science Conference*, 2010.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, 2011.

[66] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, “The NumPy Array: A Structure for Efficient Numerical Computation,” *Computing in Science and Engineering*, vol. 13, 2011.

[67] B.-k. Wang, Y.-f. Huang, W.-x. Yang, and X. Li, “Short Text Classification Based on Strong Feature Thesaurus,” *Journal of Zhejiang University Science C*, vol. 13, 2012.

[68] Y. Li, A. Tripathi, and A. Srinivasan, “Challenges in Short Text Classification: The Case of Online Auction Disclosure,” in *Proceedings of the Mediterranean Conference on Information Systems*, 2016.

Appendix A

English	French	Italian	German	Spanish
about	~~à propos~~	circa	ungefähr	~~sobre~~
airport	aéroport	~~aeroporto~~	flughafen	aeropuerto
answer	répondre	risposta	antwort	respuesta
arrival	arrivée	arrivo	ankunft	llegada
assistance	assistance	assistenza	unterstützung	asitencia
bye	aurevoir	ciao	tschüss	adios
cancelled	annulé	cancellato	storniert	~~cancelado~~
~~confirm~~	comfirmé	conferma	bestätigen	~~confirmado~~
~~contact~~	~~contact~~	contattare; contattato	kontakt	~~contacto~~
delay	retard	ritardo	verspätung	retraso
delayed	retardé	ritardato	verspätet	retrasado
departure	départ	partenza	abflug	salida
flies	voler	volare	fliegen	volar
flight	~~un vol~~	~~volo~~	flug	vuelo
~~fly~~	voler	volare	fliegen	volar
have	avoir	avere; ~~ho~~; avete; hanno	haben	tener
hello	bonjour	ciao	hallo	hola
help	aide	aiuto	hilfe	ayuda
hi	salut	ciao	~~hi~~	~~hola~~
how	comment	~~quanto~~; ~~come~~	wie	cómo
luggage	~~bagage~~	bagaglio	gepäck	equipaje
luggages	bagages	bagagli	gepäckstücke	equipajes
my	~~mon~~; ~~ma~~	~~mio~~	mein/e	~~mio~~
~~no~~	~~non~~	~~no~~	nein	~~no~~
overweight	surpoids	sovrappeso	übergewicht	sobrepeso
pay	payer	pagare; pagato	zahlen	pagar
payment	payement	pagamento	bezahlung	pago
receive	recevoir	ricevuto	erhalten	recibido
said	(j'ai) dis	disse; detto	gesagt	dicho
say	dire	dire; dice	sagen	decir
strike	grève	sciopero	streik	huelga
suitcase	valise	valigia	koffer	maleta
really	vraiment	davvero	wirklich	~~realmente~~
thank	remercier	grazie, ringrazio	danken	gracias
thanks	merci	grazie	danke	gracias
that	~~que~~	quella; che; ~~la~~	~~das~~	eso
this	~~que~~	~~questo~~; ~~questa~~	dies	esto
travel	voyager	viaggio; viaggiare	reise	viaje
understanding	compréhension	comprensione	verständnis	comprensión
validate	valider	convalidare	validieren	~~validar~~
very	très	molto; ~~tanto~~	sehr	mucho
wait	attendez	attesa, aspettando	warten	~~espera~~
we	nous	~~noi~~	wir	nosotros
weight	poid	~~peso~~	gewicht	~~peso~~
with	avec	con	mit	con
you	toi; vous	~~tu~~; ~~voi~~	~~du~~	~~tu~~; vosotros
yours	votre	vostro	dein(e)	vuestro

Table A.1: Words related to the subject airlines / airport in English, French, German, Italian and Spanish. The words strike-trough were discarded due to its closeness to Portuguese or due to their frequent use by the Portuguese population.

B

Additional Results

In this appendix are available more results to complement Section 4.

B.1 kNN

kNN	Removal Stop-words	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
1NN	False	**88.70%**	**89.90%**	**88.90%**	**88.68%**
	True. NLTK PT.	**88.70%**	89.80%	**88.90%**	**88.68%**
	True. Customized list.	87.80%	88.80%	87.90%	87.79%

Table B.1: kNN performance removing stop-words considering the 1000 most relevant features. Split 90% / 10%.

kNN	Numbers	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
1NN	Remain	88.70%	89.90%	88.90%	88.68%
	Replaced				
	Removed				

Table B.2: kNN performance upon different approaches on numbers considering the 1000 most relevant features. Split 90% / 10%.

kNN	Removal Punctuation	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
1NN	False	**88.70%**	**89.90%**	**88.90%**	**88.68%**
	True. Except "!" and "?".	88.60%	**89.90%**	88.80%	88.60%
	True	88.70%	89.80%	88.80%	88.66%

Table B.3: kNN performance upon different approaches on punctuation considering the 1000 most relevant features. Split 90% / 10%. Confusion matrices (in Table B.5) complement this table.

kNN	Removal Stop-words	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
1NN	False	**88.70%**	**89.90%**	**88.90%**	**88.68%**
	True. NLTK PT.	**88.70%**	89.80%	**88.90%**	**88.68%**
	True. Customized list.	87.80%	88.80%	87.90%	87.79%
3NN	False	77.50%	77.40%	73.20%	77.49%
	True. NLTK PT.	77.90%	77.30%	74.50%	77.93%
	True. Customized list.	77.20%	76.70%	73.20%	77.15%
5NN	False	75.80%	77.10%	70.40%	75.81%
	True. NLTK PT.	75.70%	77.50%	70.10%	75.69%
	True. Customized list.	74.90%	75.00%	69.00%	74.91%
7NN	False	73.90%	75.40%	66.80%	73.90%
	True. NLTK PT.	74.60%	77.90%	67.70%	74.58%
	True. Customized list.	72.90%	76.00%	65.10%	72.90%

Table B.4: kNN performance removing stop-words considering the 1000 most relevant features. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	560	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	566	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table B.5: Confusion matrices for 1NN considering the 1000 most relevant words. On the left we removed all the punctuation, while on the right we removed all the punctuation except "?" and "!". Split — 90% / 10%.

kNN	Words To Keep	Split	Weighted Avg. Recall	Weighted Avg. Precision	Weighted Avg. F-Measure	Accuracy
1	1000	67% / 33%	74.40%	73.90%	72.20%	74.44%
		90% / 10%	88.70%	89.90%	88.90%	88.68%
	3000	67% / 33%	75.00%	74.50%	72.60%	75.02%
		90% / 10%	88.60%	89.80%	88.90%	88.57%
5	1000	67% / 33%	66.80%	70.30%	58.20%	66.82%
		90% / 10%	75.80%	77.10%	70.40%	75.81%
	3000	67% / 33%	65.50%	66.00%	55.90%	65.51%
		90% / 10%	75.70%	78.60%	70.30%	75.69%

Table B.6: kNN performance applying baseline preprocessing techniques.

A	E	I	O	P	S	R	<– classified as
145	2	0	15	3	0	0	A (acknowledgments)
6	93	2	66	3	0	3	E (praises)
0	2	15	19	1	0	1	I (insults)
20	27	7	1504	23	0	16	O (others)
23	1	3	185	156	2	15	P (questions)
1	2	0	11	4	6	0	S (suggestions)
4	14	1	163	37	3	139	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	567	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table B.7: Confusion matrices for 1NN applying baseline preprocessing techniques, considering the 1000 most relevant words (on the right) and the 3000 (on the left). Split — 67% / 33%.

A	E	I	O	P	S	R	<– classified as
145	1	1	18	0	0	0	A (acknowledgments)
10	59	1	103	0	0	0	E (praises)
0	0	5	33	0	0	0	I (insults)
33	11	1	1549	1	0	2	O (others)
42	4	8	303	26	0	2	P (questions)
2	2	0	20	0	0	0	S (suggestions)
17	10	10	298	13	0	13	R (complaints)

A	E	I	O	P	S	R	<– classified as
149	0	0	16	0	0	0	A (acknowledgments)
12	62	1	98	0	0	0	E (praises)
0	2	5	31	0	0	0	I (insults)
29	15	1	1550	1	0	1	O (others)
32	7	5	295	44	0	2	P (questions)
0	1	0	23	0	0	0	S (suggestions)
14	19	5	292	8	0	23	R (complaints)

Table B.8: Confusion matrices for 5NN applying baseline preprocessing techniques, considering the 1000 most relevant words (on the right) and the 3000 (on the left). Split — 67% / 33%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	3	1	0	0	1	I (insults)
2	14	0	565	5	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
0	94	0	9	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	14	0	567	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	8	0	1	69	R (complaints)

Table B.9: Confusion matrices for 1NN applying baseline preprocessing techniques, considering the 1000 most relevant words (on the right) and the 3000 (on the left). Split — 90% / 10%.

A	E	I	O	P	S	R	<– classified as
10	0	0	3	0	0	0	A (acknowledgments)
6	44	1	62	0	0	0	E (praises)
0	1	2	2	0	0	0	I (insults)
3	6	2	601	0	0	0	O (others)
3	2	2	40	9	1	0	P (questions)
0	1	0	14	0	0	0	S (suggestions)
2	1	4	61	0	0	10	R (complaints)

A	E	I	O	P	S	R	<– classified as
10	0	0	3	0	0	0	A (acknowledgments)
4	46	1	62	0	0	0	E (praises)
0	1	3	1	0	0	0	I (insults)
5	6	1	599	1	0	0	O (others)
3	2	0	40	11	1	0	P (questions)
0	1	0	14	0	0	0	S (suggestions)
3	3	2	60	2	0	8	R (complaints)

Table B.10: Confusion matrices for 5NN applying baseline preprocessing techniques, considering the 1000 most relevant words (on the right) and the 3000 (on the left). Split 90% / 10%.

B.2 MNB

A	E	I	O	P	S	R	<– classified as
136	2	0	2	17	0	8	A (acknowledgments)
4	127	0	24	2	0	16	E (praises)
1	0	13	8	2	0	14	I (insults)
41	127	21	1060	196	14	138	O (others)
1	2	0	28	304	0	50	P (questions)
0	3	0	3	3	7	8	S (suggestions)
1	16	2	19	36	1	286	R (complaints)

A	E	I	O	P	S	R	<– classified as
12	0	0	0	0	0	1	A (acknowledgments)
1	101	0	6	0	0	5	E (praises)
0	0	4	0	0	0	1	I (insults)
8	30	4	503	23	9	35	O (others)
0	1	0	4	40	3	9	P (questions)
0	5	0	2	0	6	2	S (suggestions)
0	0	0	3	2	1	72	R (complaints)

Table B.11: Confusion matrices for MNB considering the 10000 most relevant words employing baseline preprocessing. On the left, split 67% / 33%. On the right, split 90% / 10%.

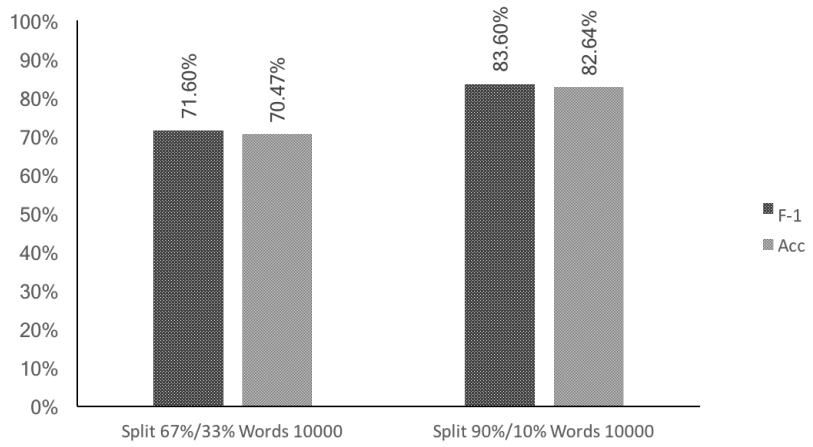

Figure B.1: F1 and Accuracy for MNB employing baseline preprocessing.

Normalized	Words To Keep	Minimum Word Frequency	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
False	1000	1	76.60%	82.40%	78.50%	76.59%
		2				
		3				
	3000	1				
		2				
		3				
	5000	1	80.80%	82.50%	80.80%	79.95%
		2	80.00%	82.50%	80.80%	79.95%
		3	79.70%	82.60%	80.70%	79.75%
	10000	1	80.00%	82.50%	80.80%	79.95%
		2	80.00%	82.50%	80.80%	79.95%
		3	79.70%	82.60%	80.70%	79.73%

Table B.12: MNB performance applying baseline preprocessing techniques. Split 90% / 10%.

Normalized	Words To Keep	Numbers	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
True	5000	Remain	82.60%	87.20%	83.70%	82.64%
		Replaced	81.80%	86.70%	83.00%	81.83%
		Removed	82.20%	86.90%	83.30%	82.17%
	10000	Remain	82.60%	86.50%	83.60%	82.64%
		Replaced	82.00%	86.20%	83.00%	81.95%
		Removed	82.20%	86.30%	83.20%	82.17%

Table B.13: MNB performance upon different approaches on numbers. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
151	1	0	5	7	0	1	A (acknowledgments)
7	126	0	30	2	0	8	E (praises)
1	3	13	17	1	0	3	I (insults)
14	29	4	1490	36	0	24	O (others)
6	1	0	81	270	1	26	P (questions)
0	4	0	7	4	6	3	S (suggestions)
3	11	1	72	38	1	235	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	103	0	4	0	0	5	E (praises)
0	0	4	0	0	0	1	I (insults)
7	37	6	499	15	7	41	O (others)
0	1	0	3	41	3	9	P (questions)
0	4	1	0	0	6	4	S (suggestions)
0	1	0	2	2	1	72	R (complaints)

Table B.14: Confusion matrix for MNB considering the 5000 most relevant words employing baseline preprocessing. On the left we split the data in 67% / 33% and 90% / 10% on the right.

B.3 SVM

Classifier	Words to Keep	Minimum Word Frequency	Weighted Recall	Weighted Precision	Weighted F-Measure	Accuracy
Poly, $d = 1$	1000	1	88.80%	89.70%	88.80%	88.80%
		2				
		3				
	3000	1				
		2				
		3				

Table B.15: SVM performance using different minimum word frequency. Split 90% / 10%.

Kernel	Degree	Numbers	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
Polynomial	1.0	False	**88.80%**	**89.70%**	**88.80%**	**88.80%**
		Replaced	**88.80%**	89.60%	**88.80%**	88.78%
		Removed	**88.80%**	89.60%	**88.80%**	88.78%
	2.0	False	88.20%	89.20%	88.20%	88.24%
		Replaced	88.20%	89.20%	88.10%	88.22%
		Removed	88.20%	89.20%	88.10%	88.22%

Table B.16: SVM performance upon different approaches on numbers considering the 3000 most relevant features. Split 90% / 10%.

Kernel	Degree	Removal Punctuation	Wei. Avg Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
Polynomial	1.0	False	**88.80%**	**89.70%**	**88.80%**	**88.80%**
		True. Except "!" and "?"	**88.80%**	89.60%	**88.80%**	88.78%
		True	88.70%	89.60%	**88.80%**	88.71%
	2.0	False	88.20%	89.20%	88.20%	88.24%
		Replaced	88.50%	89.50%	88.60%	88.42%
		Removed	88.40%	89.40%	88.50%	88.37%

Table B.17: SVM performance upon different approaches on punctuation considering the 3000 most relevant features. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
11	0	0	2	0	0	0	A (acknowledgments)
4	15	0	85	0	0	9	E (praises)
0	0	0	5	0	0	0	I (insults)
4	0	0	601	0	0	7	O (others)
1	0	0	27	24	0	5	P (questions)
0	0	0	13	0	0	2	S (suggestions)
0	0	0	38	2	0	38	R (complaints)

Table B.18: Confusion matrix for RBF kernel, $d = 1$, considering the 3000 most relevant words employing baseline preprocessing. Split 90% / 10%

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	90	0	13	0	0	9	E (praises)
0	0	3	1	0	0	1	I (insults)
2	9	0	577	2	2	20	O (others)
0	0	0	9	38	3	7	P (questions)
0	4	0	2	0	6	3	S (suggestions)
0	0	0	11	0	1	66	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	86	0	16	0	0	10	E (praises)
0	0	3	1	0	0	1	I (insults)
1	9	0	573	4	2	23	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

Table B.19: Confusion matrices of Polynomial kernel 3000 words employing baseline preprocessing. On the left $d = 1$, and $d = 2$ on the right. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
47	0	0	116	2	0	0	A (acknowledgments)
1	12	0	150	0	0	10	E (praises)
0	0	0	37	1	0	0	I (insults)
1	0	0	1576	6	0	14	O (others)
2	0	0	200	170	0	13	P (questions)
0	0	0	20	0	0	4	S (suggestions)
0	1	0	174	26	26	160	R (complaints)

A	E	I	O	P	S	R	<– classified as
11	0	0	2	0	0	0	A (acknowledgments)
4	16	0	85	0	0	8	E (praises)
0	0	0	5	0	0	0	I (insults)
3	0	0	602	0	0	7	O (others)
1	0	0	27	24	0	5	P (questions)
0	0	0	13	0	0	2	S (suggestions)
0	0	0	34	2	0	42	R (complaints)

Table B.20: Confusion matrices for RBF, $d = 1$, considering the 10000 most relevant words employing baseline preprocessing. On the left, split 67% / 33%. On the right, split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
11	0	0	2	0	0	0	A (acknowledgments)
7	10	0	87	1	0	8	E (praises)
0	0	0	5	0	0	0	I (insults)
3	0	0	605	1	0	3	O (others)
1	0	0	29	23	0	4	P (questions)
0	0	0	14	0	0	1	S (suggestions)
0	0	0	39	2	0	37	R (complaints)

A	E	I	O	P	S	R	<– classified as
11	0	0	2	0	0	0	A (acknowledgments)
4	15	0	85	0	0	9	E (praises)
0	0	0	5	0	0	0	I (insults)
4	0	0	600	0	0	8	O (others)
1	0	0	27	24	0	5	P (questions)
0	0	0	13	0	0	2	S (suggestions)
0	0	0	35	2	0	41	R (complaints)

Table B.21: Confusion matrices for RBF, $d = 1$, considering the 5000 most relevant words. On the left, removed Portuguese NLTK stop-words. On the right, just the baseline preprocessing. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	90	0	14	0	0	8	E (praises)
0	0	4	0	0	0	1	I (insults)
2	8	0	576	2	2	22	O (others)
0	0	0	9	38	3	7	P (questions)
0	4	0	2	0	6	3	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	90	0	13	0	0	9	E (praises)
0	0	4	0	0	0	1	I (insults)
2	9	0	575	2	2	22	O (others)
0	0	0	9	38	3	7	P (questions)
0	4	0	2	0	6	3	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

Table B.22: Confusion matrices for Poly, $d = 1$, considering the 5000 most relevant words. On the left, removed Portuguese NLTK stop-words. On the right, just the baseline preprocessing. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	85	0	17	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	11	0	572	3	2	23	O (others)
0	1	0	6	39	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
1	87	0	15	0	0	10	E (praises)
0	0	2	2	0	0	1	I (insults)
1	11	0	570	4	2	24	O (others)
0	1	0	5	40	4	7	P (questions)
0	4	0	2	0	7	2	S (suggestions)
0	0	0	9	0	1	68	R (complaints)

Table B.23: Confusion matrices for Poly, $d = 2$, considering the 5000 most relevant words. On the left, removed Portuguese NLTK stop-words. On the right, just the baseline preprocessing. Split 90% / 10%.

B.4 FFP adapted to short texts

k	T2S2	Wei. Avg. Recall	Wei. Avg. Precision	Wei. Avg. F-Measure	Accuracy
175	0.05	69.31%	**81.05%**	73.26%	69.31%
	0.10	75.81%	80.78%	**77.49%**	75.81%
	0.15	**77.82%**	78.31%	76.33%	**77.82%**
	0.20	74.80%	75.30%	70.36%	74.80%

Table B.24: FFP adapted to short texts performance upon different $T2S2$ scores, employing baseline preprocessing. Split 90% / 10%. The confusion matrices (in Table B.25, B.26) complement this information.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
3	92	4	8	0	3	3	E (praises)
0	0	3	1	0	0	1	I (insults)
34	36	19	443	19	25	36	O (others)
7	6	1	4	26	4	9	P (questions)
1	3	2	2	0	5	2	S (suggestions)
7	13	4	5	2	10	37	R (complaints)

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
3	88	2	16	0	2	2	E (praises)
0	0	3	1	0	0	1	I (insults)
23	25	10	509	8	16	21	O (others)
6	2	0	14	24	4	7	P (questions)
0	3	1	7	0	3	1	S (suggestions)
5	7	4	15	2	2	37	R (complaints)

Table B.25: Confusion matrices for FFP adapted to short texts, $k = 175$, employing baseline preprocessing. On the left, we considerer $T2S2 = 0.05$. On the right, $T2S2 = 0.10$. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
2	68	1	40	0	1	1	E (praises)
0	0	3	2	0	0	0	I (insults)
14	13	5	564	3	5	8	O (others)
2	0	0	33	18	1	3	P (questions)
0	2	0	10	0	2	1	S (suggestions)
2	4	2	37	1	5	27	R (complaints)

A	E	I	O	P	S	R	<– classified as
10	0	0	3	0	0	0	A (acknowledgments)
1	46	0	66	0	0	0	E (praises)
0	0	2	3	0	0	0	I (insults)
9	6	3	586	1	3	4	O (others)
2	0	0	44	8	0	3	P (questions)
0	1	0	12	0	1	1	S (suggestions)
1	1	0	57	0	4	15	R (complaints)

Table B.26: Confusion matrices for FFP adapted to short texts, $k = 175$, employing baseline preprocessing. On the left, we considerer $T2S2 = 0.15$. On the right, $T2S2 = 0.20$. Split 90% / 10%.

A	E	I	O	P	S	R	<– classified as
13	0	0	0	0	0	0	A (acknowledgments)
4	84	2	17	0	2	4	E (praises)
0	0	3	1	0	0	1	I (insults)
22	21	10	511	9	14	24	O (others)
5	2	0	18	23	3	6	P (questions)
0	3	1	7	0	3	1	S (suggestions)
5	8	4	14	2	7	38	R (complaints)

Table B.27: Confusion matrix for FFP adapted to short texts, $k = 175$, removing all the punctuation except "?" and "!". Split 90% / 10%.

B.5 Dataset Retrieved Knowledge

Common observed questions while labeling the corpus:

- "My passport has expired. Can I use the old passport number to buy a ticket plane?"
- "The passport with which I bought the trip will expire before traveling, how should I proceed?"

- “Can I bring a trolley plus a handbag with my computer?”
- “I got an error while paying the ticket planes. How should I proceed?”
- “What are the conditions and prices to transport domestic animals?”