



Università
Ca' Foscari
Venezia

—
Ca' Foscari
Dorsoduro 3246
30123 Venezia

Corso di Laurea (*vecchio ordinamento,
ante D.M. 509/1999*)

in Scienze dell'Informazione

Tesi di Laurea

Gestione della Clientela tramite analisi di reti sociali

Telecomunicazioni e Twitter

Relatore

Ch. Prof. Salvatore Orlando

Laureando

Bruno Quintavalle

Matricola 761617

Anno Accademico

2012 / 2013

To my mother.

Abstract



The web is reach of interesting and profitable informations, written by humans for other humans in different forms and formats. Blogs, forums and social networks are examples of different environments, where opinions about products, alerts about problems and sentiment about companies and competitors can be harvested for business or other purposes. The difficult is that most of the informations are intended to be read by humans, and there is very often too much to be managed by a single person in a reasonable amount of time. Here a method is described, whose main purpose is to classify Twitter messages into categories of interest, in order to present to the human reader only a greatly reduced number of texts to read and further analyze. The categories of interest considered here are related to telecommunication companies, but the method presented can easily be adapted to any other theme. Automatic classified messages may also simply be counted over periods of time, to reveal significative events, or cross-tabled by telephone company, to make comparison between them. The first step of the method is to extract features from the texts, using dictionaries and a primitive but effective form of stemming. Then a classifier made with an ensemble of three basic classifiers (a decision tree, a neural network and a naïve Bayesian classifier) is used to assign labels to the text. Because of the unavoidable ambiguity of text categories, the classifier has been made to be both hierarchical (able to deal with a taxonomy of labels) and multi label (able to assign more labels to the same text). Twitter messages have a maximum length of 140 characters, and this certainly simplify their analysis. Anyway, with a slight modification of the algorithm, the same set neural networks trained for Twitter has been successfully used to label texts of any length.



Il web è ricco di informazioni utili e talvolta remunerative, scritte da esseri umani per altri esseri umani in molteplici forme e formati. Blogs, forums e social networks sono esempi diversi di ambienti dove opinioni sui prodotti, avvisi relativi a problemi e sentimenti su compagnie e concorrenti possono essere raccolte per motivi di business o altro. La difficoltà sta appunto nel fatto che la maggior parte delle informazioni sono scritte per essere lette da esseri umani, e sono in genere in quantità troppo elevata per essere gestite da un'unica persona in un lasso di tempo ragionevole. Qui un software viene descritto il cui scopo è di classificare messaggi di Twitter in categorie di interesse, per poter presentare al lettore umano solo un numero altamente ridotto di messaggi da leggere ed ulteriormente analizzare. Le categorie di interesse qui considerate sono relative alle compagnie di telecomunicazioni, ma il metodo presentato può facilmente essere adattato a qualsiasi altro soggetto. I messaggi classificati automaticamente possono anche venir semplicemente contati per periododi tempo, per rivelare eventi significativi, o tabulati per compagnia telefonica, per confrontarle tra loro. Il primo passo del metodo consiste nell'estrarre le caratteristiche dal testo, usando dizionari ed una primitiva ma efficace forma di stemming. Quindi un ensamble di classificatori di base (un decision tree, una rete neurale ed un naive Bayesian classifier) è stato usato per assegnare le labels al testo. A causa dell'inevitabile ambiguità delle categorie di testo, il classificatore è stato costruito in modo da poter essere sia gerarchico (capace di gestire tassonomie di etichette) che multi-label (capage di assegnare più etichette allo stesso testo). I messaggi di Twitter hanno una lunghezza massima di 140 caratteri, e questo certamente ne semplifica l'analisi. Tuttavia, con una piccola modifica all'algoritmo, lo stesso set di reti neurali addestrate sui messaggi di Twitter è stato usato con successo per etichettare messaggi di lunghezza qualsiasi.

Contents

1	PROBLEM STATEMENT	1
2	INTRODUCTION	2
2.1	Gathering the data	3
2.2	Features extraction from the text	4
2.3	Classification.....	7
3	MESSAGE DOWNLOADING AND SIMPLE TAGS	9
4	FEATURES EXTRACTION	13
4.1	Features definition and graph.....	13
4.2	Regular verbs stemming	14
4.3	Building the stemming tree.....	16
4.4	Taking surrounding words into account	17
5	ENSEMBLE CLASSIFIER	19
5.1	Complex Tags	20
5.2	Neural Networks.....	22
5.3	Naïve Bayesian Classifiers	29
5.4	Results.....	32
6	SLIDING WINDOWS CLASSIFIER (SWC)	35
6.1	Introduction	35
6.2	Algorithm details.....	36
6.3	Training	37
6.4	Testing.....	39
6.5	Results.....	41
7	OTHER PRACTICAL APPLICATIONS	43
7.1	Some statistics	43

7.2	Graphs.....	44
7.2.1	Telecomitalia - Data network problems	45
7.2.2	Tre - Voice network problem	45
7.3	Ranking.....	47
8	FUTURE DEVELOPMENTS: GET A FREE TRAINING SET!	50
9	CONCLUSIONS	51
10	APPENDIX 1: ESSENTIAL OAUTH KNOWLEDGE	52
11	APPENDIX 2: FEATURE LIST	61
11.1	General.....	61
11.1.1	Time	61
11.1.2	Delay.....	61
11.1.3	Lack.....	62
11.1.4	Money	62
11.1.5	Geo.....	62
11.1.6	Shipment.....	63
11.1.7	Contract.....	64
11.1.8	Communication	64
11.1.9	Sentiment.....	65
11.2	Technical.....	66
11.2.1	Devices.....	66
11.2.2	Malfunctions.....	66
11.2.3	Web.....	67
11.3	Telephone Companies Specific.....	68
11.3.1	People.....	68
11.3.2	Phone Line.....	68
11.3.3	Line Quality.....	69
11.3.4	Companies	69
11.3.5	Variation	70
11.3.6	Ceasing.....	71
11.3.7	Return.....	71
11.4	Other	72
11.4.1	Advertisement	72
11.4.2	Unrelated.....	72
12	APPENDIX 3: COMPLEX TAGS WITH EXAMPLES	73
13	BIBLIOGRAPHY	76
13.1	Books.....	76
13.2	Articles.....	76

1 Problem Statement

If we look into the web for informations about telecommunication companies (Telecomitalia as well as others), we normally find, besides a lot of advertising and their official web sites, three other sorts of things: opinion about products, requests for information and complaints from unsatisfied clients. They are abundant, and they are all valuable, including complaints. Provided most frequently occurring problems are already known by company's customer care services, new problems arises every day as new products go to market. Clusters of similar problems encountered by different clients may highlight company's weakness, things that should be solved immediately to avoid giving any advantages to the competitors. Of course periodical surveys are normally already in action to detect this situations, but surveys are always quite expensive, and almost never give results in real time. Informations on the web, on the contrary, are free and up to date¹. The idea of having a machine to surf the web looking for all possible problem a client can manifest, and automatically identify and report any new situation, is exiting, but unfortunately out of reach (at least for the moment). For this kind of task human intervention is still necessary. But the problem is that the amount of informations to examine is already too big for a single person to deal with, and we expect it to grow huge soon. At least, we would like to have a method that extracts from the huge amount of texts only the small percentage that may be of more interest, even with some tolerance, so that someone can read it in a reasonable amount of time. And this is exactly the theme of this thesis.

¹ And they can also be considered more important, as public complaints, if not solved quickly, can easily generate negative public opinions. There seems to be a growing client awerness about this factor, and this lead to think that in the near future public complaints will grow in number as well.

Reese: I never understood why people put all their information on those sites. Used to make our job a lot easier at the CIA.

Finch: Of course. That's why I created them.

Reese: You're telling me you invented online social networking, Finch?

Finch: The Machine needed more information. People's social graph, their associations. The government had been trying to figure it out for years. Turns out most people were happy to volunteer it. Business wound up being quite profitable, too.

[Person Of Interest, season 1, episode 18]

2 Introduction

Our primary aim is to be able to assign labels to messages downloaded from the Twitter servers, assuming, because of the way we downloaded them, that they are already pertinent, with a certain degree of confidence, to a specific subject (in this case, that of Telecommunications). Twitter messages are easy to get, quite numerous for the subject we are considering (at the moment, about 1000 a day), and with a maximum length of 140 characters. This last characteristic should in theory limit the number of different topics a person can talk about inside a single message, hopefully simplifying the problem.

An algorithm that automatically assign a label to an object (in this case, to a text) is called a classifier. Traditional classifiers operates on features extracted from the object, like for example the count of some specific words inside the text, and produce a single label from a set L .

In our case for example, we want the set L to contain labels like 'BUS' to indicate that the text refers to some kind of business issue (offers, contracts an so on), like 'QUA' to indicate that the issue is about the quality of the service (line or device problem), and many others as well.

Unfortunately, even with the very short Twitter messages, we cannot expect that every message can be classified with a single label: very often for example a business issue will refer to a quality problem and vice versa, and in this case the correct thing to do will be to assign both labels to the same message. In general, even using the best effort to define a set of disjoint concepts, ambiguity and overlapping between them can hardly be avoided. What we need for the problem we are facing is called a Multi-Label classifier. Actually, to use the words of [Zhang - 05]:

“Research of multilabel learning was initially motivated by the difficult of concept ambiguity encountered in text categorization, where each document may belong to several topics (labels) simultaneously.”

For simplicity our approach will be a simple problem transformation method, a binary relevance method BR as defined in [Read - 11] (also known as PT4 [Tsoumakas - 07]) transforming “[...] the multi-label problem into multiple binary problems [...] such that each binary model is trained to predict the relevance of one of the labels”.

An ensemble of three different base classifiers will be used for the purpose: a decision tree, a neural network and a naive Bayesian classifier. The results of each of them will be combined in a simple majority vote scheme to give the final result of the ensemble classifier.

Besides ambiguity, there is another problem we face in defining a good set of labels: the possible lack of sufficient number of examples per label. Every classifier requires a decent number of examples to be able to learn to predict a label correctly, and although a particular text may clearly and immediately suggest a very specific and important label to the reader, there may not be enough other example of the same kind to make the classifier work with it. A simple approach to deal with this problem is to organize the labels in a hierarchy, a taxonomy where the leaves (more basic concepts) may have fewer examples to train with, and for that reason a quite poor prediction quality is expected and accepted for them. At the highers levels of the taxonomy anyway, more examples are accumulated (i.e.: all those of

their subtree), and performance of the classifiers should be much better. In this way, we can keep a very large label set, freely tagging training text in the way we consider more appropriate, and still have a good classifier ready to work from the beginning, at least with the highest level labels. As new examples becomes available for the rarest concepts, classification will improve for them too.

This kind of classifiers are called Hierarchical Multi Label, as defined for example in [Vens - 08]:

“Hierarchical multi-label classification (HMC) is a variant of classification where instances may belong to multiple classes at the same time and these classes are organized in a hierarchy”

Finally we are going to deal with the problem of text segmentation: the ensemble classifier introduced above works with a statistic of the features of the text, i.e.: with a normalized count of each feature found in the text. As long as the statistics are similar, no matter how long the text is, similar results should be expected by the classifier. But obviously, if a short piece of text talking about the interesting subject is “lost” inside a much larger text talking of everything else, statistics changes highly and the classifier is not expected to be able to find it.

Our secondary aim is so to be able to use the same training set of Twitter messages to build a classifier able to detect the pieces of text that matter even when strongly diluted inside a very large unrelated text, as can happen in the case of blogs and long online magazine articles. Using a standard approach, we could first segment the text into coherent segments. Anyway, as in [Beeferman-99] “[...] *such seemingly simple problem can actually prove quite difficult to automate [...]*”.

The solution we propose skips the entire problem of text segmentation, but actually takes the fundamental idea from the subject: a sliding window of N consecutive words. During training, a single text in the training set generates a number of training records, one for each possible position of the window. Each training record consists of the statistics of the features inside the window and of all the labels assigned that piece of sentence. During test, the same kind of sliding window is used to calculate the statistic, and the labels are predicted by the classifier (at the moment, just a neural network, not the entire ensemble), resulting in a sort of graph drawable over the words, with $|L|$ points for every position of the N-words window (where L is the set of labels used). Because the output of the classifier in this situation is often quite confused, a very simple method is proposed to detect the areas of the graph where the decision are sharp enough, and take them as the output of the classifier.

Over the short lenthed Twitter messages, this method has shown to perform worst than the simple ensemble, but it definitely outperform the ensemble in the situations for which it has been made, i.e. in finding short relevant texts diluted inside much larger, irrelevant ones.

2.1 Gathering the data

In extracting data from the web, and assuming we are focusing only on textual informations (so disregarding images or other things), two major kinds of problems are encountered.

The first is finding where in the page the useful informations are placed. What appears obvious at human eyes is quite a difficult task for a computer program. A web page may look like a clean formatted table, with informations properly placed in a very specific order, but having a program to automatically recognize that order inside the HTML code behind the page it is in general not so easy.

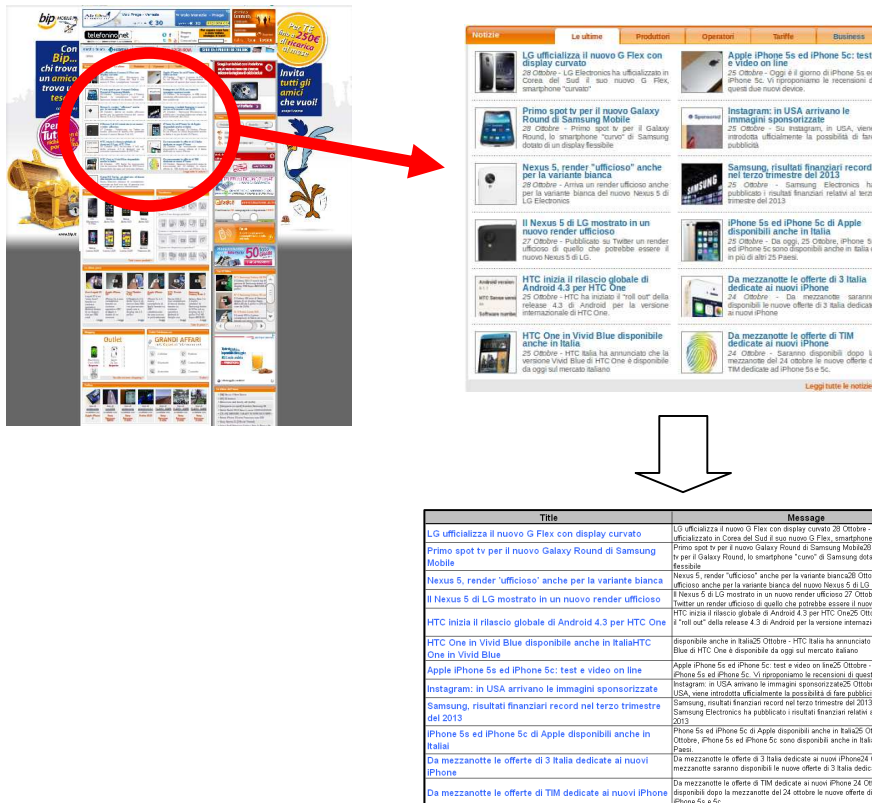


Figure 2-1: the process of automatically extracting information from a web page. After identifying the interesting portion of the page, the software must be able to transform a complex HTML fragment into a simple data table

If the number of interesting websites is small enough, ad hoc routines can be written for each of them (and maintained as needed). If on the contrary the aim is to write a program that behaves totally automatically, discovering the useful stuff and extracting it without any human instruction or intervention, adapting its behavior to whatever the website is, we generally fall into the realm of Artificial Intelligence. Very often anyway, forums pages are formatted by some sort of computer program, making them quite repetitive in structure, so the problem is not really impossible to solve, and some solutions, although non trivial, are suggested for example in [Liu-08].

Lucky enough, sometimes servers provides API (Application Programming Interface) that allows a program to download the text directly via some sort of web service. This is often true for social networks, more rare for forums.

2.2 Features extraction from the text

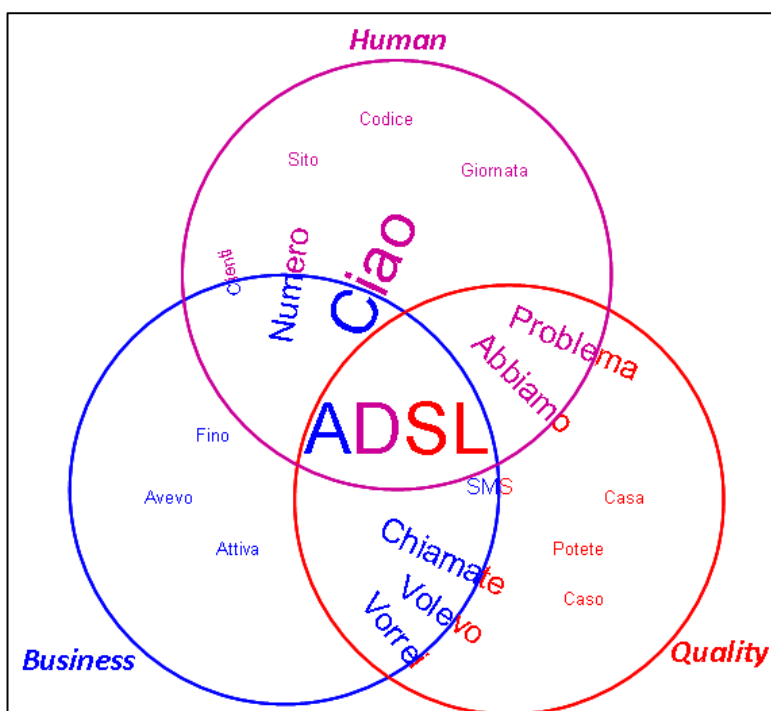
The second and probably major problem is to analyze the text, examining something that is written by humans and meant to be read by other humans. Of course the natural solution would be to have some humans do the task, and provided the amount of text to process is not excessive, this is often the best practice.

But for our purposes this turned out to be unfeasible: at the moment, possible interesting Twitter messages related to telephone companies amount to about a thousand per day. So what we aim, and

what is the purpose of the present thesis, is a **simple method to sort messages into different arguments of interest, specific enough to result in number manageable by a human reader.**

By “simple method” we mean that we are not going to look into the structure of the sentence, but instead we are going to consider sentences as “bag of words”, i.e. we are (almost) not caring about the order of the words and their mutual relationship but only about their presence. This approach may look quite naïve, and in fact we don’t expect to be extremely precise, but it has been proven to be quite successful in the past, as documented for example in [Groot-12]. So basically we are going to classify texts by looking for some specific words, that we know (because of some previous research) to be likely to be present in each of the classes we consider. To illustrate with an example, Figure 2-1 gives a basic, high level idea of the thing. The three categories considered there are: problems related to human factor (like operators bad behavior), business aspects like offer and contracts, and quality of service. Some words that frequently occur in messages of each category are shown, with a size proportional to their average occurrence (based on a set of about 600 manually classified messages).

Figure 2-1: three possible categories of messages, with some of the words often used in them. The size of the words is proportional to its absolute frequency (in a set of about 600 manually classified messages), whilst color and position reflects frequency relative to the category.



What should already have been noticed here, is that there is in general no single word that allows a direct and precise classification of the message into a single category: words that occurs in just one category are too rare to be used alone, and more frequents words are always “shared” among two or more categories.

This is the reason why we need a classifier, an algorithm that will assign labels to the text examining, so to speak, the mixture of interesting words it finds.

But instead of presenting the classifier a huge amount of different words to deal with, a good approach is to group them into equivalent classes, removing for example verb desinences and collecting words with similar meaning into a same class.

This process is called feature extraction. From Wikipedia:

“When the input data to an algorithm is too large to be processed and it is suspected to be notoriously redundant [...] then the input data will be transformed into a reduced representation set of features. [...] If the features extracted are carefully chosen it is expected that the features set will extract the relevant information from the input data in order to perform the desired task using this reduced representation instead of the full size input.”

In our case, feature extraction will consist of two kinds of processes:

- Lemmatization (or Stemming²), that is defined as “the process of grouping together the different inflected forms of a word so they can be analyzed as a single item” [Wikipedia.org]
- Looking for words of similar meaning (synonyms). Notice that words may have similar meanings in a context but not in another: talking about phone lines for example, “abitazione” and “alloggio” can be considered equivalents of “casa”, whilst “patria” and “dinastia” certainly can not.

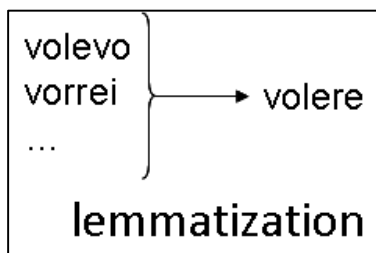


Figure 2-2: Example of lemmatization (stemming²). All conjugations of verb “volere” (“to want”) will be represented with its infinite form



Figure 2-3: Synonym choice. In a specific context some synonyms need to be excluded.

It has to be said here that there is in fact set of labels that can quite easily be given by just looking for very few and very specific words in the text, without the need of a complex form of feature extraction and subsequent classification. Company names and technologies for example, most of the times although not always, have very peculiar names (*telecomitalia*, *vodafone*, ...). Taking advantage of this, we assign the messages a first set of labels from a set that, because of the extreme simplicity of the algorithm, we are going to call the Simple Tags.

² Lemmatization and stemming are to similar processes, whose task is to relate an entire group of words to a single string representation. The difference is that whilst lemmatization requires this word to be a valid lemma (i.e.: to have a meaning), stemming does not. From Wikipedia (“stemming”): “The stem need not be identical to the morphological root of the word; it is usually sufficient that related words map to the same stem, even if this stem is not in itself a valid root”.

Later on, when the classifier will allow us to detect much more complex concepts, like “Human Behavior” and “Quality of Service” (we are going to call these the Complex Tag Set), it would be nice to look at both, counting instances of messages in a cross table, to see for example which company has more problems with employee behavior, and which with its quality of service.

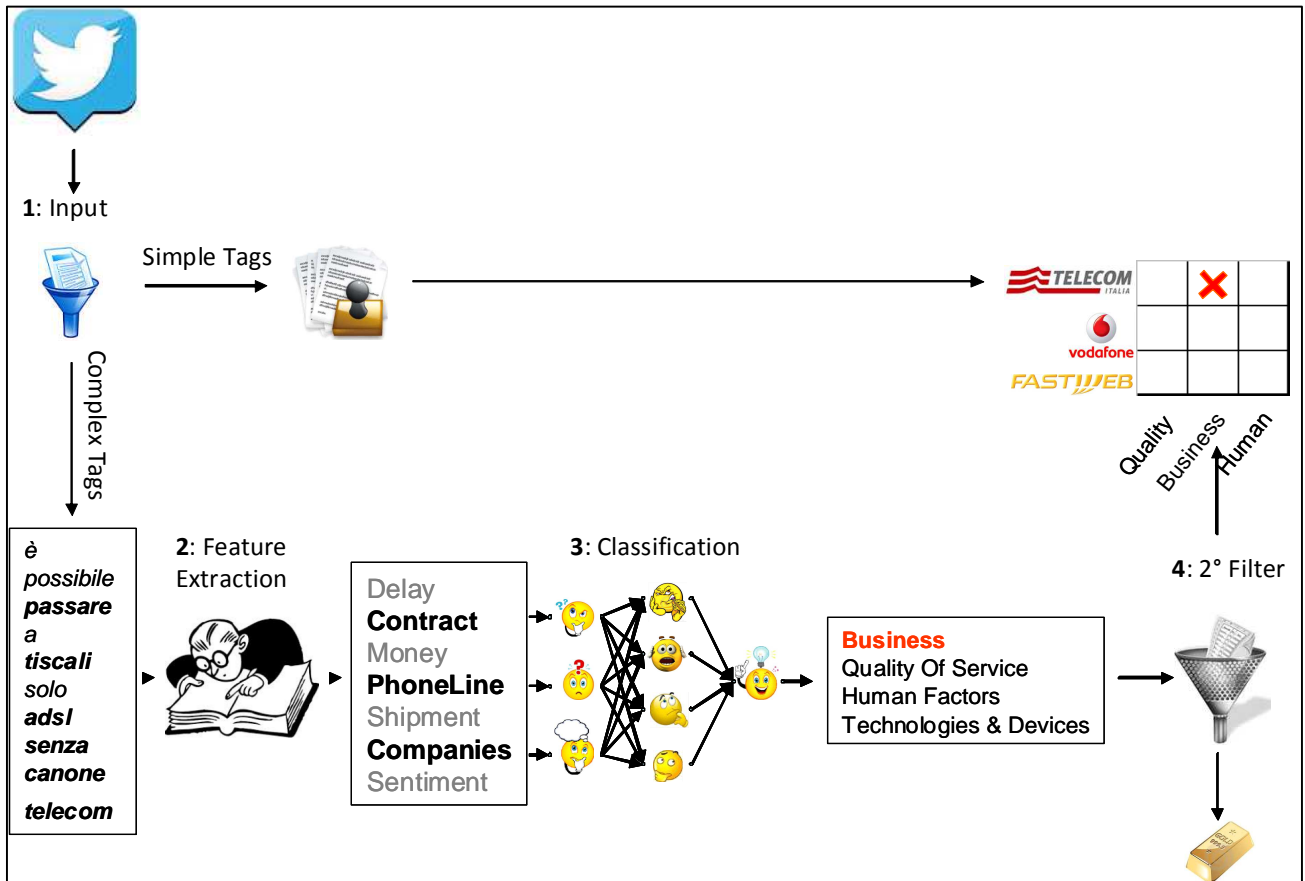


Figure 2-4: Obtaining a cross table of Simple and Complex tags. A first set of tags (Simple Tags) are given immediately after downloading the message from Twitter server, and refer to companies and other things easy to identify in the message. A second set of tags (Complex Tags) is given through by feature extraction and classification. Their cross table is expected to evidence problems and perceived performance of the different telephone company.

2.3 Classification

The process of feature extraction introduced in the previous paragraph, roughly speaking, transforms words into numbers. Specifically, it allows to assign to a text a vector of numbers that counts the number of times each feature is found in the text³. The purpose of the classifier is to learn to associate these vectors with the labels that have been manually assigned to the text by a human operator.

³ Actually, because as we are going to see, features are best represented with real values, the counters represents the number of times a feature is found above a certain threshold.

As stated above, for this task we are going to need a Hierarchical Multi-Label Classifier (HMC), and we are going to make one using an ensemble of three basic classifiers:

- A decision tree
- A neural network
- A naïve Bayesian classifier

Details of this will be given in chapter 5.

3 Message Downloading and Simple Tags

This chapter is about downloading messages from Twitter servers, filtering unwanted messages and assigning a first set of tags to the correct ones. This set of tags, called the Simple Tags, are hierarchically organized in a taxonomy that allows groups of them to be referred as single objects in graphs and tables.

Beside some technical complication due to the authentication protocol and to the rate limits imposed by Twitter (see appendix 1 for a quick and dirty approach to the OAuth protocol), querying the server for messages is simply a matter of creating an URL that specifies the words that must be present and, if it is the case, those that must be not (preceding the word to exclude with a minus sign “-“)

The following URL for example makes a request of the last (more recent) 100 Italian tweets containing the word “*tim*” but not the word “*burton*”.

```
https://api.twitter.com/1.1/search/tweets.json?q=tim%20-  
burton&lang=it&count=100&with_twitter_user_id=true
```

The result is a text file in Json format, from which it's quite easy to extract the text of the message as well as other useful informations:

```
{ "statuses":  
  [ { "created_at": "Thu Oct 10 15:06:00 +0000 2013",  
      "id": 388319510861598720,  
      "id_str": "388319510861598720",  
      "text": "@cubovision @TIM_Official peccato che sono a Milano !",  
      "in_reply_to_status_id": 388317598183469056,  
      "in_reply_to_user_id": 134745857,  
      "in_reply_to_screen_name": "cubovision",  
      "user":  
        { "id": 1251328404,  
          "name": "PAPERAIÀ",  
          "screen_name": "paperaia",  
          "location": "Milano",  
          "description": "Leggere, Viaggiare, Margherita",  
          "followers_count": 3,  
          "friends_count": 13,  
          "created_at": "Fri Mar 08 10:20:18 +0000 2013",  
          "geo_enabled": false,  
          "statuses_count": 6,  
          "lang": "it",  
        },  
      "geo": null,  
      "coordinates": null,  
      "place": null,  
      "contributors": null,  
      "retweet_count": 0,  
      "favorite_count": 0,  
    }  
  ]  
}
```

As said earlier, at this stage we are giving a first set of tags to those messages that contains very peculiar names, like brand names (“*telecomitalia*”, “*impresasemplice*”, “*vodafone*”) and technologies (“*wifi*”, “*adsf*”). Because those tags can be given without too much reasoning and almost without errors, we call this first set **Simple Tags**. Later on, the same messages will be given a second set of tags, related to much more complex and uncertain subjects, like quality and behavior.

Unfortunately anyway, not all the brand names are so peculiar: some brands may be named with very common words (like “*wind*”, “*tre*”), or may be wrongly written by the user with spaces inside, transforming them into couples of common words (“*telecom italia*”, “*impresa semplice*”).

If we want to retrieve those messages too, we have to be prepared to receive a load of unwanted, unrelated messages. Sometimes it’s possible to identify some words that frequently occurs in the unwanted messages, and simply ask the server to filter those messages out by specifying those words directly in the query. As an example, looking for the “*wind*” company we are very likely to receive a great amount of American weather forecast, all very similar as can be seen from Table 3-1. In this case, by simply requiring the server to remove all the messages that include the words “*humidity*”, “*temperature*” or “*pressure*” we largely reduces the problem.

Table 3-1: Examples of undesired but easy to remove messages downloaded looking for the Wind telephone company.

21:01 EST: <u>Temperature</u> : 22.8°C, <u>Wind</u> : 0 km\h (ave), 5 km\h (gust), <u>Humidity</u> : 63%, <u>Rain (hourly)</u> 0.0 mm, <u>Pressure</u> : 1018 hPa, rising slowly
17:01 EST: <u>Temperature</u> : 31.8°C, <u>Wind</u> : ENE, 3 km\h (ave), 12 km\h (gust), <u>Humidity</u> : 24%, <u>Rain (hourly)</u> 0.0 mm, <u>Pressure</u> : 1018 hPa, falling
10:55 CET: <u>Temperature</u> : 4.8°C, <u>Wind</u> : 1 kmh (ave), 7 kmh (gust), <u>Humidity</u> : 61%, <u>Rain (hourly)</u> 0.0 mm, <u>Pressure</u> : 986 hPa, falling quickly

In some other cases some sort of client side filtering may be necessary.

Sometimes it’s just a simple matter of looking at the distance between words, requiring them to be very near or exactly adjacent (N-Grams). We can discard for example a downloaded message if it has any word between “*impresa*” and “*semplice*”.

Another quite effective technique is to discard messages that haven’t at least one word from a specific list of words, collected among the most used words in the subject (in our case, a list of words about telecommunications). A similar, more complex idea is implemented in the so called Naïve Bayesian Classifiers, also used here and described later in chapter **Error! Reference source not found.**. Being simpler than their naïve counterpart, we found no better name for this kind of filters than “**Trivial Filters**”. Anyway, these filters include the same kind of lemmatization used later for feature extraction (see chapter 4), that although simplified and imprecise, has been proven to be fast and quite effective. (In fact, Trivial Filters utilize the same algorithms and words of some of the features described later)

To summarize, looking at Figure 3-1 we see that we have three different input situations:

- Those that can safely be tagged “as is” (like “*telecomitalia*” or “*impresasemplice*”)
- Those that can be solved with some simple server side filtering (like “*tim –burton*”)
- Those that requires client side filtering (like “*impresa semplice*” or “*wind*”)

In the picture the yellow nodes represents the **Simple Tags**, i.e.:

- A group of one or more queries related to the same argument is given a specific name
- All the messages resulting from that queries are tagged with that name

As can also be seen in figure Figure 3-1, Simple Tags are hierarchically organized in a tree, with yellow nodes converging into orange ones. In this way, in reports and graphs higher level objects, like “telephone companies”, can be referred as single entities.

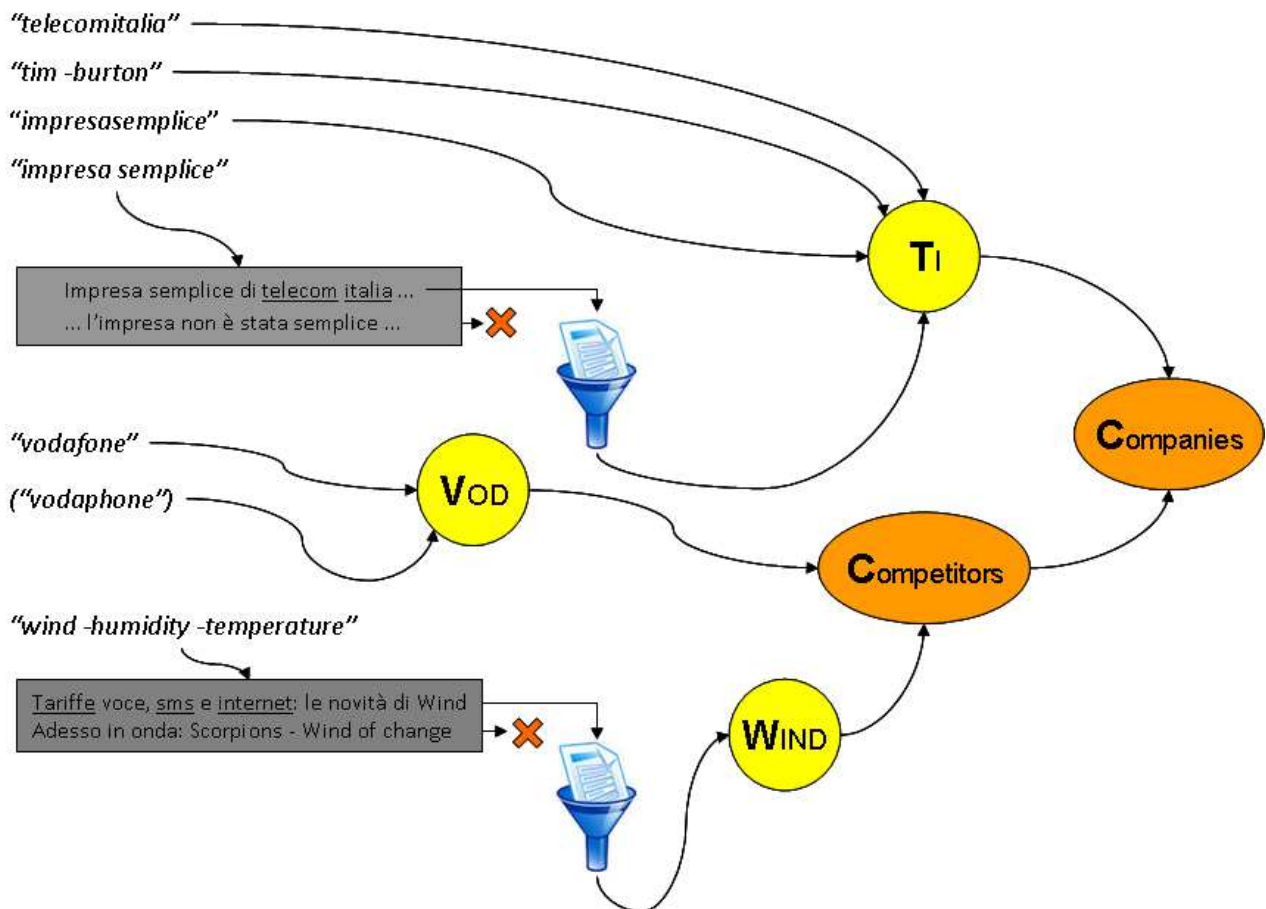


Figure 3-1: Sketch of the input process. Between quotes characters are the query strings, the gray rectangles contains Twitter servers responses. Some queries need local filtering. Simple Tags are assigned on the basis of the queries that generated the message (yellow nodes). For simplicity of use, tags are also organized in a taxonomy (orange nodes).

Even at this stage, before performing the second and more interesting classification of the messages, some useful information can be seen on the data. Twitter allows to download, together with the messages, some information about their writers. A simple graph like one in Figure 3-2 can show for example which argument is talked more about, and by whom⁴: the philosophy of the graph is that the size of the nodes is proportional to the number of messages and the position of the user nodes is such that they are closer to the subject they talk more about.

Specifically:

- The big red nodes represent the tags (Telecom Italia, Competitors, Offers, Technologies and Devices)
- the smaller green, blue and gray nodes represents the users

⁴ The graph is interactive: moving the mouse over a user node makes a tooltip with its name to appear, clicking on a node makes the list of all its messages to appear (see also Figure 3-3).

- the size of a node is proportional to the number of messages it contains
- the initial positions of the tag nodes is around a large circle (they can be moved)
- user nodes in the middle of the graph are positioned in such a way that they will be closer to the subject (tag node) the user writes more about
- user whose messages only relates to a single subject are positioned around that tag (and not in the middle of the graph)

Albeit simple, this graph gives a informative quick view of what has been downloaded, and has been proved useful for a rapid first analysis of the data.

Figure 3-2: A possible way to visualize a set of messages. The size of the nodes is proportional to the number of messages they contain. Red nodes represents Topics of interest (tagged messages), whilst green and blue nodes represents Users. Topics nodes are (by default) placed around a large circle, whilst Users nodes are placed closer to the Topic nodes they talk more about. User that only talk about one Topic are placed around it

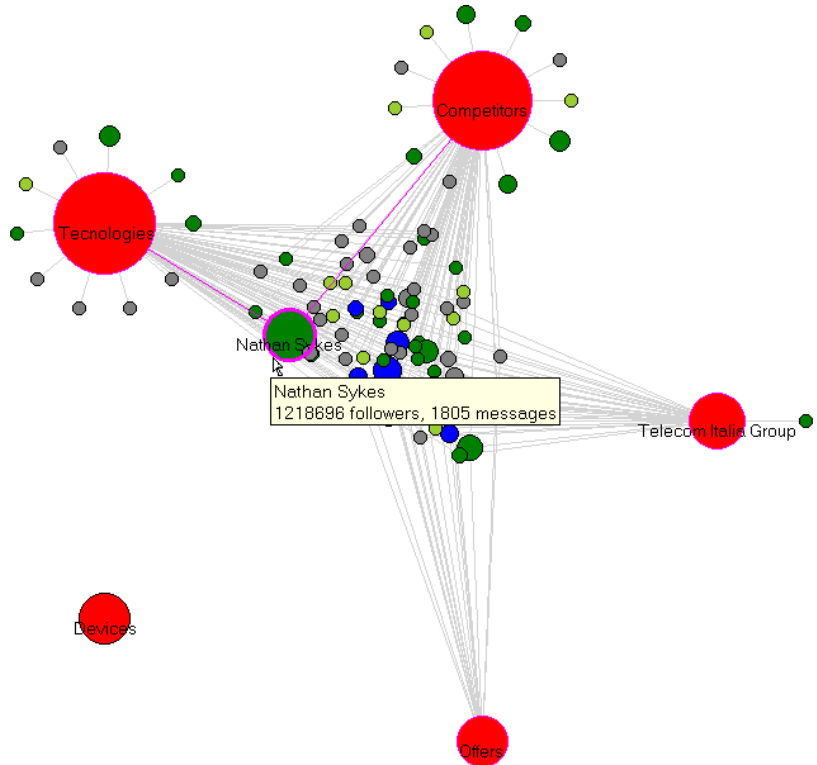


Figure 3-3: The graph of previous figure is interactive. Besides moving the Topic nodes, a list of the contained messages can be requested right-clicking on the node.

User	Message	Date
Maurizio...	@Tre_It ed ancora stamattina irraggiungibile!	02/03/2013 ...
Maurizio...	@Tre_It ho mandato via DM cognome nome e numero di telefono	01/03/2013 ...
Maurizio...	@Tre_It dopo aver passato il numero sono giorni che non posso ricevere nè effettuare chi...	01/03/2013 ...
Maurizio...	@TuriCaggegi @Tre_It grazie non sapevo fossero su tw	01/03/2013 ...
Maurizio...	@Tre_It spento e riacceso il telefono pochi istanti fa.tentativo di chiamata ma ancora in blo...	06/03/2013 ...
Maurizio...	@Tre_It avete tutti i miei recapiti telefonici.attendevo che la sig.na giusy mi richiamasse st...	05/03/2013 ...
Maurizio...	@Tre_It ed anche oggi tutto invariato! Ho spento e riacceso il tel stamattina ma niente tutt...	05/03/2013 ...
Maurizio...	@missbuzzi come ti trovi con tim io l'ho tenuta 1 mese e tolta per disperazione. Ora sono ...	12/03/2013 ...
Maurizio...	@Tre_It siccome mi sono profondamente scocciato di essere introvabile, faccio sostituire l...	07/03/2013 ...

@Tre_It ed ancora stamattina irraggiungibile!	
Id	307792393695010
Created At	02/03/2013 9.59.59
User	MaurizioP
User Name	Maurizio

19 messages Close

4 Features Extraction

This chapter is about extracting features from the text. A matrix of numbers is calculated, one row for each word in the sentence and one column for each feature. Values, between 0 and 1, represent the level of confidence for the couple (Word, Feature). A simplified but effective form of stemming is explained. Then, an algorithm is described that is used in some cases to rise the confidence values when groups of nearby words with the same feature are present in the sentence.

4.1 Features definition and graph

In this context a feature can be defined as a semantic characteristic that can, with some degree of certainty, be derived from a word or from a small set of nearby words.

As an example, any number near a symbol or name of a currency {€ \$ £ euro dollari sterline, ...} almost certainly refers to the concept of money, as does any word in a set like { credito, denaro, pagare, gratuito ... }

Anyway, in some cases the situation may not be so neat: “euro” alone may not refer to money at all, and for words like “somma” or “prelevare” we may never be sure, even after examining the surrounding ones.

For that reason a **feature here is expressed with an arbitrary number between 0 and 1**, with 1 representing a sure presence, 0 the absence, and any value in between a certain degree of presence.

The Figure 4-1 below shows a features graph⁵, with the sentence on the left (one word per row) and a set of vertical lines each representing the values of one feature for every word of the sentence.

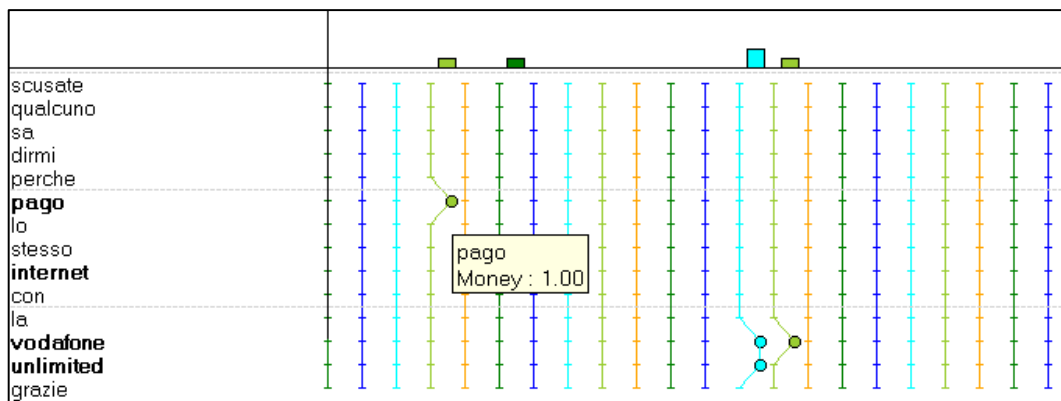


Figure 4-1: A Feature graph. Every feature is represented by a vertical line, with a value between 0 and 1 for each word in the sentence. Istograms on top of the lines represents the number of times the line goes above a specified threshold. Moving the mouse over the graph a tooltip shows the word, the feature name and the value.

⁵ This graph is interactive too. Moving the cursor over the dots tooltips apperas with the word and its calculated feature value (In the picture example, the tooltip shows that the word “pago” has been attributed the value 1.00 for the feature “Money”)

● StopWords	● Number	● Time
● Delay	● Lack	● Money
● Contract	● Web	● Shipment
● Geo	● PhoneLine	● LineQuality
● Device	● Malfuncion	● Companies
● OtherOper	● Variation	● Return
● Ceasing	● People	● Communication
● Sentiment	● Advertisement	● Unrelated

Figure 4-2: Legend of a Feature Graph

Every feature is “artfully” calculated using a function of its own, and although most of those functions share a very similar structure (a simple lookup into 4 different dictionaries, as we will explain shortly), in some cases more specific “tricks” comes in action, in order to recognize, among other things, money, dates, unit measures and addresses. Before we go into any detailed description of the algorithm for feature extraction,

let’s state its driving principle. For our purpose (sorting messages into meaningful categories) **being extremely precise (from the linguistic point of view) is not really the main point**: we are going to deal with errors of various kinds anyway, due to the fact that the authors of the texts we are analyzing, either intentionally or not, do not make a very strict use of the grammar.

On the contrary, expecting to have to deal with a considerable and expanding amount of data, **speed is a real major concern**. We trust that the algorithms on the following layers will be robust enough to deal with all the unavoidable errors.

Said that, in order to assign a value for each word in the sentence we generally consider four different kinds of words (collected in 4 different dictionaries):

1. Regular verbs
2. Regular plurals (names, adjectives, adverbs)
3. Other, non regular, single words
4. N-Grams (sequences of N consecutive words)

Irregular verbs can, if necessary, be treated inserting each of their inflections in the third dictionary, the one for non regular and single words.

4.2 Regular verbs stemming

The regular verbs case is the most complicated of the four. We want to implement for them what is called a stemmer, that is an algorithm that removes all the inflection from the verb and returns a string that uniquely identify the verb. Very roughly speaking, we are going to break the word into two parts: a left part (called the stem) that later on we are going to use in a dictionary lookup, and a right part (the inflection), that must be present in a list of allowed strings and that, once found, is basically discarded. Saying it differently, we first check if the word ends with any of the string in the list, and if so we remove that part and look up in the dictionary for what remains.

Considering that the list of allowed ending strings is not going to be huge, a fast and simple solution is to organize that list of string in a tree where each node either has:

- Exactly 26 children, one for every letter in the English alphabet, obviously with some of them (or better most of them) void.
- A “residual” string to match

To search the string inside the word we start from the end of the word proceeding backward, and from the root of the tree proceeding downward.

At each step, if we are on a node that has children we take another char from the end of the word and proceed on the child with the corresponding index. If instead we are on a residual node, we simply check that the residual string is entirely contained in the word.

Let see that with a simple example. Let say, for simplicity, that the set of allowed substrings to look for is: { "are", "ere", "ire", "erebbe", "erebbero" }. This list, in a way that we are going to explain soon, leads to the tree in Figure 4-3.

Let also say that the word to analyze is “compre**re**bbe”.

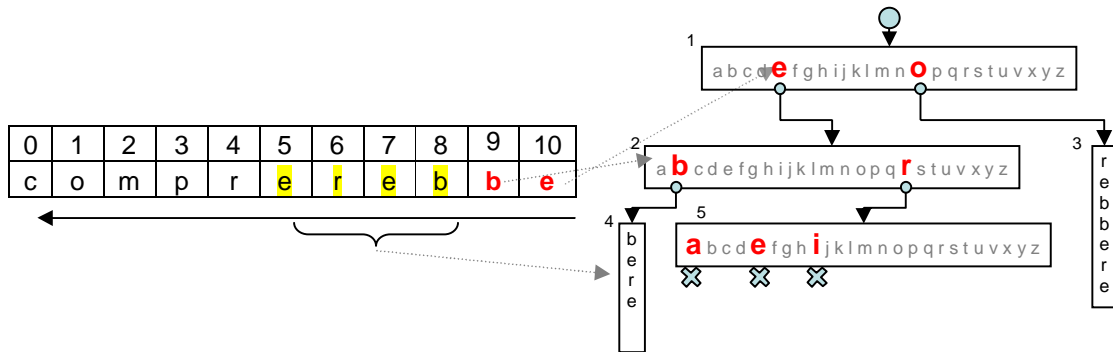


Figure 4-3: A tree used for lemmatization, together with an example. The last char ‘e’ of the word “compre**re**bbe” is found on the root of the tree. Following it on node 2, the second to last ‘b’ is found, and finally all the residual “bere” is also found and removed from the word.

- Starting from character ‘e’ in position 10 in the word and from node 1 in the tree, we follow the link **e** to node 2
- The next character in the word (‘b’ in position 9) bring us to the residual node 4 containing the string “bere”
- Continuing backward on the word, we can match every character of the residual in it (from position 8 to 5)
- The string “compr” is left in the word (position 0 to 4), and will be searched inside the dictionary (regular verb “comprare”, saved as “compr”+“are”)

As an exercise the reader may want to split the word “capire” in “cap”+“ire” following the string “eri” (reverse of “ire”) along the node path {1,2,5} of the same tree.

Notice that this method allows invalid combinations to be recognized as valid words: the strings “comprire” and “comprere” are both recognized as valid conjugation of the verb “comprare”, even if they are not.

Although it would be possible to test the validity of a combination, this sort of grammar check is not always as easy as it may seem at first glance, and probably not so important for our purpose: we assume (or better hope) the number of possible invalid combinations that also have a different meaning (i.e.: a proper entry in a different dictionary) being neglectable.

4.3 Building the stemming tree

The tree can easily be made with a recursive function. Let S be the array of the strings, reversed (“ire” becomes “eri”) and alphabetically sorted in ascending order. In our example S would be:

S	
0	ebbere
1	era
2	ere
3	eri
4	orebbere

Let $\text{makeNode}(\text{int lev}, \text{int beg}, \text{int end})$ be the recursive function that create a node using strings in S , in positions between beg and end , at level lev of the tree. (So the entire tree of the example will be built by calling $\text{makeNode}(0, 0, 4)$).

The first step in makeNode is to check if $\text{beg}=\text{end}$. If so we have only one string, and we create a residual node containing its substring starting at position lev .

Otherwise:

- create a node N with 26 children
- group all the strings between beg and end by their character in position lev ; in other words, for every resulting group G_c :
 - All the strings in G_c has the same character c in their position lev .
 - Because S is sorted, all strings in G_c are contiguous in S , in positions between some beg_c and end_c indexes
- For every non empty groups G_c recursively call $\text{makeNode}(\text{lev}+1, \text{beg}_c, \text{end}_c)$ and assign the result to the child of index $(c - 'a')$ of the node n

To continue with our example, the first level of recursion in the call to $\text{makeNode}(0, 0, 4)$ produces a node with 26 children, 2 of which are not null:

- child ‘o’ is a residual node containing the string “rebbere”
- child ‘e’ is to further be expanded with the set of string between 0 and 3

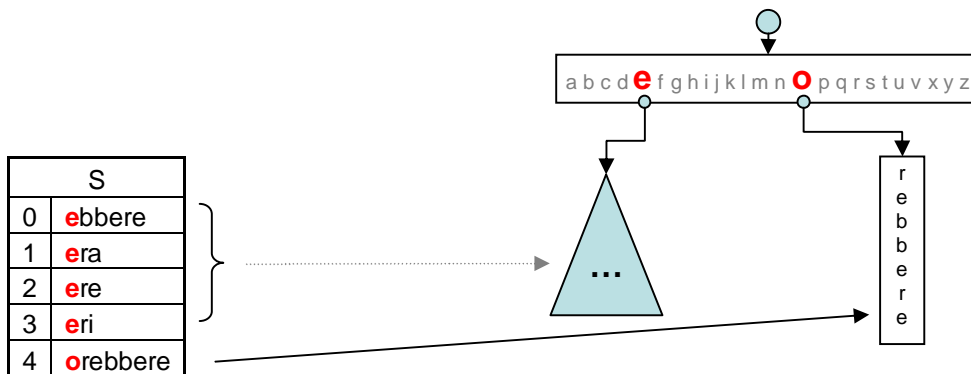


Figure 4-4: The first iteration in building the lemmatization tree: assuming only the 5 words in the table are to be looked for in the strings, the first four will go in a subtree whose root is attached to the ‘e’ child of the root of the tree, whilst the last one will form a residual attached to the ‘o’ child of the root of the tree.

4.4 Taking surrounding words into account

If in a text we read words like “*adsl*”, “*voip*” or “*lte*”, we are immediately quite sure that the sentence has something to do with a phone line. But what if we read words like “*voce*”, “*linea*” or “*collegamento*”? We can’t say anything for sure, unless we already know what the context is. In other words, for the purpose of determining the semantic content of a string, clusters of them must be considered together.

Having no resources for a thorough statistical research, we resort to a reasonable empiric formula, together with a tool for testing its results.

To calculate the values for a specific feature (meaning) for every word in the sentence:

- First, we assign a weight to every word that may be used to express that specific meaning. The highest is the weight, the more likely that word may have that meaning. For simplicity, the labels High, Medium and Low has been used for the purpose.
- Then, considering a moving window of n contiguous words in the sentence, we count how many words we find for each of the 3 possible weights. Let that counters be (nH, nM, nL)
- Next, given 3 arbitrary weights (wH, wM, wL) a score is calculated:

$$sco = wH * nH + wM * nM + wL * nL$$
- Finally, if the score reaches an arbitrary threshold thr the features values of the words in the windows are raised (never lowered) to the values in the set (vH, vM, vL) , depending on the label of the word (i.e.: if the word is labeled as High its new value will be the maximum between vH and its previous one, and so on for Medium and Low labeled words)

The following set of 8 parameters seems to work quite well with all the features in which they have been used:

Window Size	5
wH	1
wM	0.5
wL	0.33
thr	0.9
vH	1
vM	0.8
vL	0.6

It basically can be read as: “*If inside a window of 5 words you find at least*

- *1 High*
- *or 2 Mediums*
- *or 3 Lows*
- *or 1 Medium and 2 Lows*

then assign 1 to the Highs, 0.8 to the Mediums and 0.6 to the Lows, unless they are already higher than that.”

A tool has been developed to allow experimenting and tests. Figure 4-5 and Figure 4-6 shows the effects on the feature “PhoneLine” in a sentence where 2 of its characteristic words around the word

“voce” has been temporarily removed: 0.4 is the default value assigned to the word. After the two words “linea” and “adsl” has been reintroduced its value is raised to 0.8.

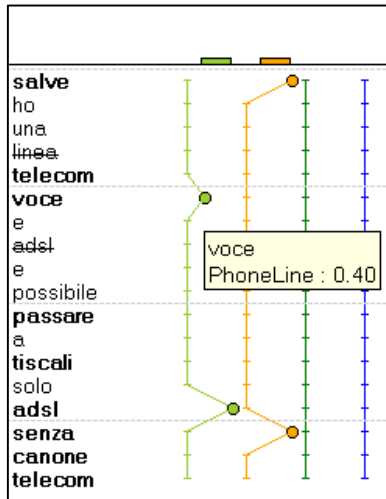


Figure 4-5: PhoneLine” feature (in light green) with the words “linea” and “adsl” removed. The word “voce” gets a score of 0.4

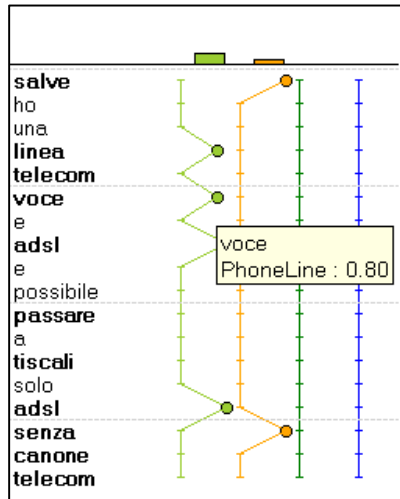


Figure 4-6: Same feature with the words “linea” and “adsl” included. Now the score of “voce” is 0.8

5 Ensemble classifier

In this chapter we first analyze the set of labels that we want to be automatically assigned to the texts. Despite every effort to choose a set of disjoint concepts, messages that express more than a single one are frequently found. For that reason we need a classifier that allows multiple labels to be assigned to the same text at the same time (called a Multi-Label classifier).

Also, the nature of the labels themselves suggests to organize them in a taxonomy. Beside clearness, this gives the advantage of allowing to work with an initial quite small training set, if compared with the number of labels. As we are going to see, in our case the classifier start to work well when at least a couple of hundreds examples are given for each of the labels. Without a label's taxonomy, we would be forced either to find all the necessary examples for each label, that sometimes may be infeasible because of their rarity, or to give up the label itself. In a taxonomy anyway, all the examples of a node are, by definition, also examples of its parent. So even if at its lower level the classifier may not perform so well, at least at the top level we can expect it to be good. And of course in time, as long as new examples for the rarest cases are added, the performance of the classifier will improve for them too. For this last reason, our classifier is also going to hierarchical, becoming a so called Hierarchical Multi-label Classifier (HMC). To use the words of ([Vens-08]):

“Hierarchical multi-label classification (HMC) differs from normal classification in two ways: (1) a single example may belong to multiple classes simultaneously; and (2) the classes are organized in a hierarchy: an example that belongs to some class automatically belongs to all its superclasses (we call this the hierarchy constraint).”

About the multiple label ability, different approaches are known for the problem. Although we are going to use the simplest one, few words can usefully be spent about the others.

According to [Tsoumakas-07], existing methods for multi label classification can be grouped into two main categories: problem transformation methods and algorithm adaption methods:

“We define problem transformation methods as those methods that transform the multi-label classification problem either into one or more single-label classification or regression problems [...] We define algorithm adaptation methods as those methods that extend specific learning algorithms in order to handle multi-label data directly.”

The most simple and common problem transformation method, which is also the one we are adopting here, is to consider one label at a time, making a different classifier for each of them. The training set for each label l will have all the examples of the original training set, labeled as true if the original example contains l and false otherwise. This kind of approach is called Binary Relevance (BR) in [Read-11], Single Label Classification (SC) in [Vens-08] and referred to as PT4 in to [Tsoumakas-08]. Beside possible inefficiency (the learner has to be run $|\mathcal{L}|$ times, once for each label in $|\mathcal{L}|$), the main disadvantage of this method is claimed to be the fact that it does not model the correlation that may exist between labels in the training data.

The opposite approach is to consider every possible combination of labels, represent it with a single new label and make a classifier for that one. This is called Label Combination or Label Power Set (LC) in [Read-11], and referred to as PT3 in to [Tsoumakas-08]. The problem with this method is that set of possible combination can be huge ($2^{|\mathcal{L}|}$, if \mathcal{L} is the set of labels). Of course only the combinations found in the training set are considered (as the others have no examples to train with), but this is a problem too, as for some combinations too few examples may be available.

Other methods in between this two has been suggested, as the binary Pairwise approach (PW), where a binary model is trained for each pair of labels, and various Classifier Chain models ([Read-11]), where

basically a chain of binary classifiers are made, augmenting the attribute space of each classifier with the binary predictions of the previous ones.

In order to obtain a good classification accuracy, an aggregation of three basic classifiers has been used. This kind of aggregations are called **ensembles** or classifiers combinations, and do their work by taking a vote (sometimes weighted) on the prediction made by each base classifier. Although not required, three different algorithms have here been used:

- A decision tree
- A neural network
- A naïve Bayesian classifier

After a brief discussion about the label set we have decided to use, the following sub-chapter will briefly describe each of the basic classifier in details.

5.1 Complex Tags

Although features extracted in the previous layer could be considered tags themselves, they are not enough. What we aim to find are sentences expressing even more complex concepts like “bad quality of something” or “bad behavior of someone”, even if we know that a text may clearly state that an operator or a salesman have had a bad behavior of some sort without even mentioning the words “operator” or “salesman” or “behavior” at all. To distinguish them from the previous set of labels, we are going to call this second set the **Complex Tags**.

Although we expect the classifier to do some magic, it’s always a good thing to be realistic in what we pretend, so in order to decide the set of possible meanings to look for, a trade off between at least three factors is considered:

1. What we would like to find (the boss requests)
2. What is actually expressed in the texts
3. What we can reasonably expect to be able to discern effectively

After quite a lengthy process of trial and error, the set shown in **Error! Reference source not found.** has been defined.

Business labels refer to everything that has to do with business issues, from the description of a new offer to the complaints for a bad one. Quality of Service includes issues about line malfunctions, bad coverage (both mobile and broadband), as well as every problem that may occur with any device connected to the line. In the Human Factor subtree we include everything that has to do with the behavior of salesmen and operators. This is actually a critical point of the research, and not yet well working as (luckily enough) not enough examples has been found so far. The Internet & Technology subset include everything that has to do with computer and devices, from characteristics to malfunctions (there is in fact a little overlapping with the Quality of Service node, as we do not expect, at the moment, to be able to effectively distinguish the two). Finally an Other subgroup has been included to consider everything that may be on interest but that do non fall into the previous categories. In **Error! Reference source not found.** we take the opportunity to show that, although not so frequently, with the labels we have defined some messages actually requires multiple labeling. **Error! Reference source not found.** shows part of the tool that allows to highlight and assign different labels to different portions of the same sentence (or even to assign different labels the same portion of the sentence, if that is the case).

Table 5-1 Complex Tags

Business
Offers (A description or a request of information about the characteristic of a specific offer, or a comment or comparison between two of them)
Contract (Something about a contract that has been already made. Most of the time a complaint about money)
Activation, Cessation, Portability (Everything related to the activation or cessation of a line or porting of a number from one company to another)
Human Factor
Difficult Communication
Late or No Response
Bad Behavior
Positive comment / Thanks
Operator Reply
Shipment - Lost Packages
Other
Internet, Technologies & Devices
Device characteristics
Device malfunctions
Official Operator App / Web Site
Other
Quality of service
Coverage
Line Quality
Malfunction / Set Up / Delay
Other
Advertisements
Comment on Advertisements

Table 5-2: Some messages requires more than a single tag.

Message	Tags
<i>Ho attivato l'abbonamento Vodafone Unlimited ma a volte la connessione dati non si attiva e per inviare un messaggio dice errore</i>	Activation + Malfunction
<i>@Tre_It quando creerete una tariffa internet flat da abbinare al webpocket per quei poveri disgraziati che non hanno una adsl decente?</i>	Offers + Line Quality
<i>Ciao, puoi attivare Vodafone UNLIMITED dal sito o chiamando il numero gratuito 42070. Tks.</i>	Activation + Web Site
<i>@Tre_It È un problema di tariffe, vostri operatori avevano promesso di richiamarmi più volte ma tutto tace.</i>	Contract + Difficult Communications
<i>mi sa che comunque è ora di cambiare provider. #fastweb ha calato la reale banda disponibile un'altra volta. umpf!</i>	Cessation + Line Quality
<i>@FASTWEB sono senza linea da questo pomeriggio. Mi fate contattare dall'assistenza tecnica, please?</i>	Line Quality + Difficult Communications
<i>Ho un piano #FastWeb da 20 MB, 20 MB. Ho un download EFFETTIVO di 500KB da SEMPRE, UN QUARANTESIMO di ciò che PAGO. COMPLIMENTI. @FASTWEB</i>	Contract + Line Quality
<i>consiglio abbonamento Adsl: Attualmente ho Telecom e nonostante va discretamente costa decisamente troppo.</i>	Offers + Line Quality

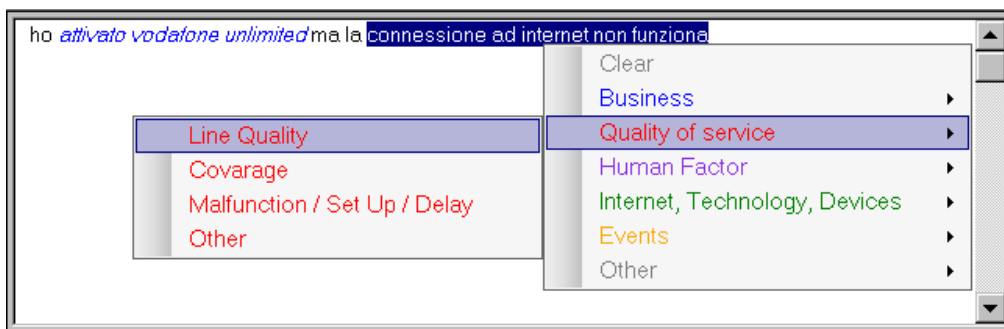


Figure 5-1 the sentence “ho attivato vodafone unlimited ma la connessione ad internet non funziona” has been assigned two different tags: a “Activation” (blue) tag for the first portion (“attivato vodafone unlimited”) and a “Line Quality” (red) tag for the last one (“la connessione ad internet non funziona”)

5.2 Neural Networks

Neural Networks are mathematical models of networks of real neurons. They have been successfully used for many applications, as described for example in [Berry-97]:

“... from predicting financial series to diagnosing medical conditions, from identifying clusters of valuable customers to identifying fraudulent credit card transactions, from recognizing number written on checks to predicting the failure rates of engines”.

As also pointed out in [Berry-97], good problems to be solved with neural networks are the one where input and output are very well understood, but their relationships are not: “You have a good idea of which features of the data are important, but not necessarily how to combine them”.

And this happen to be exactly our case: reading a sentence with the purpose of finding in which category it should fall, the words that may be important for the classification are usually very easy to spot. But the same words appear in sentences of very different categories. Of course we know that it is their combination that matters, and we trust the neural network to find it.

Although many different types of neurons are known at the time, in term of functionality most of them share a similar structure:

- A set of inputs from other neurons, called dendrites
- A cell body, called the soma
- One output to other neurons , called the axon

In real cells, the incoming signal through the other neurons has the effect of changing the electrical potential inside the body of the receiving cell. If this potential reaches e threshold, a pulse is sent over the axon. A simple mathematical model of a neuron, called perceptron, is shown in Figure 5-2: it computes a weighted sum of its inputs from other units and outputs +1 or -1 according to whether this sum is above or below a certain threshold:

$$Out(t + 1) = Sgn(\sum_i w_i * Inp_i(t))$$

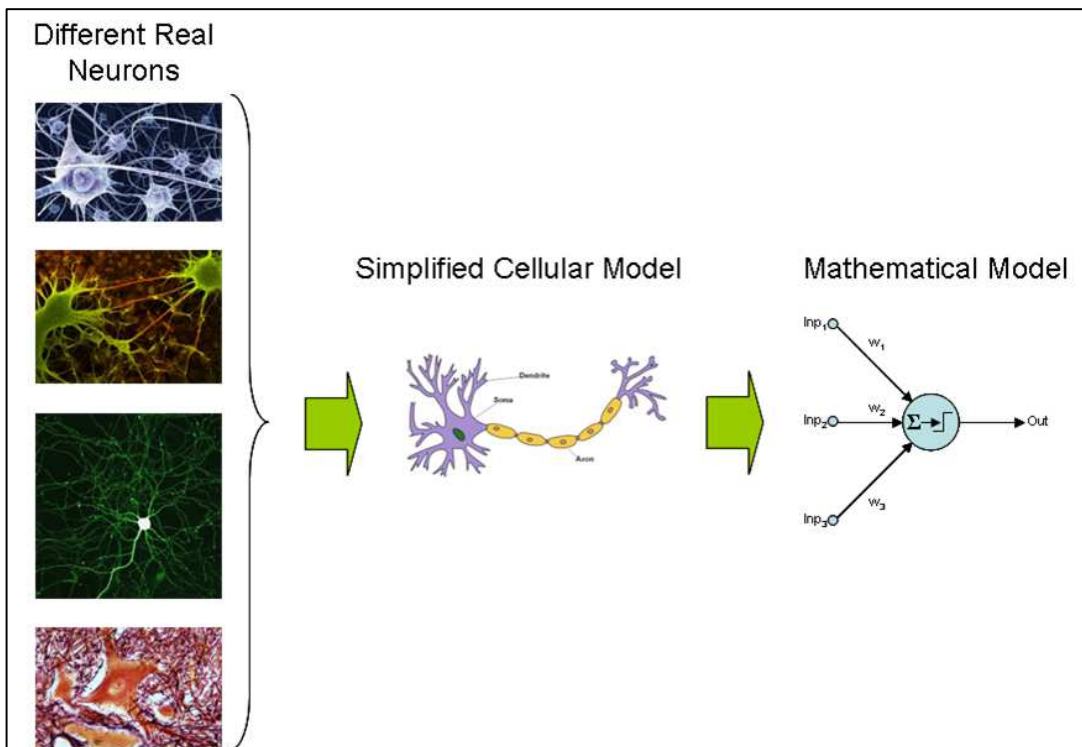


Figure 5-2: From “real” neurons, to a simplified cellular model, to its mathematical model.

Let's test this idea with our problem. Let's say that we want to make a neuron that is able to make a distinction between two set of messages:

- Those who talks about **Offers** characteristics (most of the times enthusiastic claims about nice things to buy)
- Those who talks about **Contracts** (very often quite angry complains about things that have already been acquired)

Of course this distinction may not always be very sharp, and in some cases may it result unclear even for a human being, but we are willing to accept a certain amount of errors anyway.

For the sake of simplicity, let's consider only two features:

- **Contract**: terminology typically found in contracts
- **Money**: every string that may represent a money value, plus other terms that imply some kind of payment (“costi”, “bolletta”, “pagamento”, ...)

So let say that the set of messages in Table 5-3 is given, whilst Table 5-4 reports the input (calculated features) and the respective desired output for the same set of messages.

Table 5-3: Some messages with the desired tags. In bold and color the words recognized as features: the “Contract” words in pink, the “Money” words in blue.

ID	Message	Desired Tag
1	<i>Navighi, senza limiti di traffico, con una velocità fino a 42.2 Mbps per il primo GB del mese.</i>	Offer
2	<i>forse la 3 è meglio! Vodafone nn solo mi addebita costi inesistenti ma adesso mi dice che ho finito il GB dopo 2 giorni!</i>	Offer
3	<i>puoi risparmiare fino al 70% sulla bolletta telefonica senza cambiare numero di telefono.</i>	Offer
4	<i>un piccolo canone per un numero telefonico che può ricevere ed effettuare 50 chiamate in contemporanea.</i>	Offer
5	<i>Solo online 2 mesi di internet fino a 5GB con chiavetta Wi-Fi inclusa a soli 29€ #soloconvodafone</i>	Offer
6	<i>Un'offerta da non perdere! Un #Galaxy con 400 min & SMS e 1GB da 15€ mese... Buttatevi ragazzi!</i>	Offer
7	<i>Scusate qualcuno sa dirmi perchè pago lo stesso internet con la Vodafone Unlimited? GRAZIE**</i>	Contract
8	<i>#Vodafone #Unlimited boiata pazzesca. E ti chiedono pure 10 euro per attivarla.</i>	Contract
9	<i>Attenzione: se si attiva #VodafoneUnlimited il 190 diventa a pagamento (1 Euro a chiamata)</i>	Contract
10	<i>#posteitaliane non recapita le bollette, ti staccano il telefono e @telecomitalia che fa? ti addebita oltre 13€ per la riattivazione. Ottimo, no?</i>	Contract
11	<i>Ho un piano #FastWeb da 20 MB. Ho un download EFFETTIVO di 500KB da SEMPRE, UN QUARANTESIMO di ciò che PAGO. COMPLIMENTI.</i>	Contract

Given that data, the question is: can we build a perceptron able to make a distinction among the two kinds of messages in the set? The answer is: ... almost!

Graph in Figure 5-3 shows a plot of the records where the X axis represent the Money feature and the Y axis represent the Contract one. Green diamonds represent Offers messages, whilst orange squares represent Contract ones.

ID	Money Feature	Contract Feature	Desired Tag	Desired Output	Network Output
1	0.000	0.100	Offer	+1	+1
2	0.125	0.000	Offer	+1	+1
3	0.071	0.000	Offer	+1	+1
4	0.000	0.111	Offer	+1	+1
5	0.125	0.000	Offer	+1	+1
6	0.143	0.143	Offer	+1	+1
7	0.143	0.000	Contract	-1	-1
8	0.250	0.125	Contract	-1	-1
9	0.375	0.000	Contract	-1	-1
10	0.143	0.000	Contract	-1	-1
11	0.091	0.000	Contract	-1	+1

Table 5-4: Numeric values of “Money” and “Contract” features for the same messages of the previous table, plus a comparison between the desired and the real output of the perceptron. Shown in red, for record 11, the wrong one.

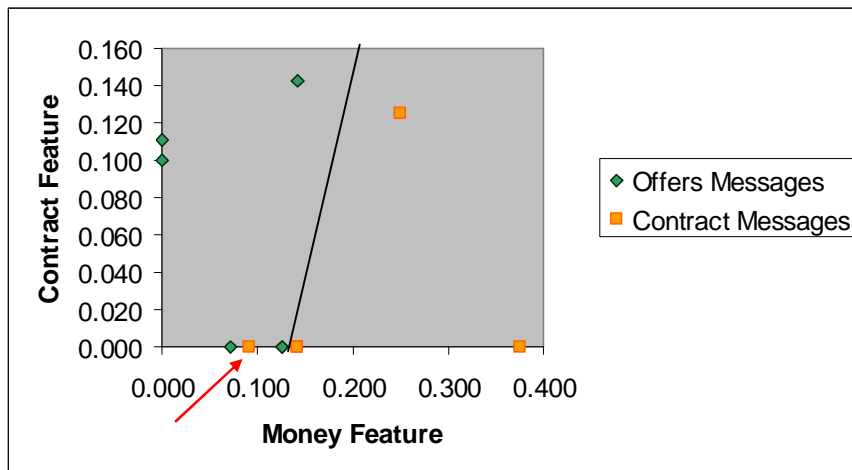


Figure 5-3: Plot of the Money and Contract features of the messages in the Cartesian plane. A line can be drawn that separates all the messages of one kind from all of the other, except for one (record 11, shown by the arrow)

As can be seen, a straight line can be drawn in the graph that separates the two kinds of point (except for the one pointed by the red arrow), so that each green point is left/above of the line and each orange one is right-below it.

Let $y = m * x + k$ be the equation of that line (in this particular instance, $m=2.388$ and $k=-0.317$)

Obviously a point (x_i, y_i) is above that line if $y_i > y$, i.e. if $y_i > m * x_i + k$, so the sign of the quantity $y_i - m * x_i - k$ will be positive for Offer messages and negative for the others.

Substituting Money and Contract for X and Y, our neuron must be able to calculate the sign of the quantity $Contract - m * Money - k$, and this can be done arranging the inputs and the weight as in Figure 5-4 (notice that an extra input has been used, clamped to -1).

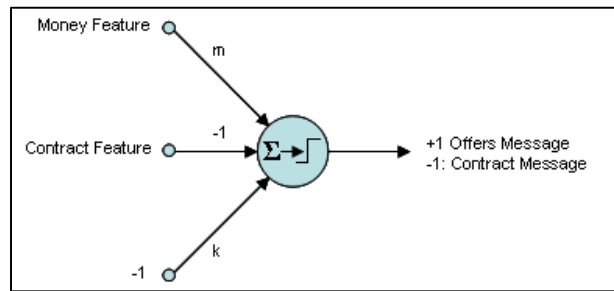


Figure 5-4: A perceptron able to classify all but one of the messages in the example. (m= 2.388 and k=0.317)

The last column of Table 5-4 shows the result of such calculation, and it can be seen to be correct for all but one record, as the 11-th message lies on the wrong side of the line.

Can the output of this simple neuron be made to be always correct, calculating its weight in some other way? As demonstrated in [Hertz-91] the answer is no:

“The condition for solvability of a problem by a simple perceptron with threshold units is whether or not that problem is linearly separable”.

A linearly separable problem is one in which a plane can be found in the input space separating the two sets of input patterns.

To overcome this limitation one or more layers of unit can be inserted between the input and the output (called hidden layers), to make what is called a feed-forward multilayer network.

Figure 5-5 shows an example of such a network, with a single hidden layer made of 6 units.

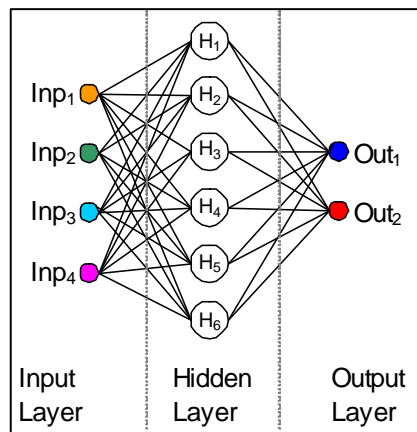


Figure 5-5: Example of a two layers feed-forward neural network with 4 inputs, 6 element in the hidden layer and 2 outputs. (Conventionally, the input layer is not considered in the count of the number of layers of a neural network).

A network is called feed-forward if the outputs of the elements of each layer are connected only to the elements of the next one: backward connections, or connections to the same layer or to layers past the next one are not allowed.

Feed-forward networks are not the only models of neural networks studied so far, but are the one best suited for our purpose. Also, for them various training algorithms (methods to calculate the weights) are known and available in free libraries and tools. And although the use of a library is not mandatory, it certain saves a lot of time.

Figure 5-7: A possible activation function of a neural network element (hyperbolic tangent)



Here a library called **Encog 3** [<http://www.heatonresearch.com/>] has been used, that happen to be free, quite well documented (books are available on Amazon for both the C# and Java implementations), and, most important, it seems to work.⁶ As can be seen from Table 5-6, the network in Figure 5-8 with the weights in Table 5-5 correctly calculate the desired value for all the records in the set. In Table 5-6 the columns with names like ΣX_i contains the weighted sum of the inputs of the element X_i :

Figure 5-6: Encog Library Logo

$$\Sigma X_i = \sum_j v_j w_{i,j}$$

The output value of an element is the result of the application to that sum of what is called the **activation function** (next columns), that for the perceptron used in the previous example was the simple $\text{sign}(x)$ function.

The training algorithm anyway requires the activation function to be continuous and derivable. There are different functions with that characteristic, here the choice has been to use the hyperbolic tangent $\tanh(x)$, whose graph can be seen in Figure 5-9.

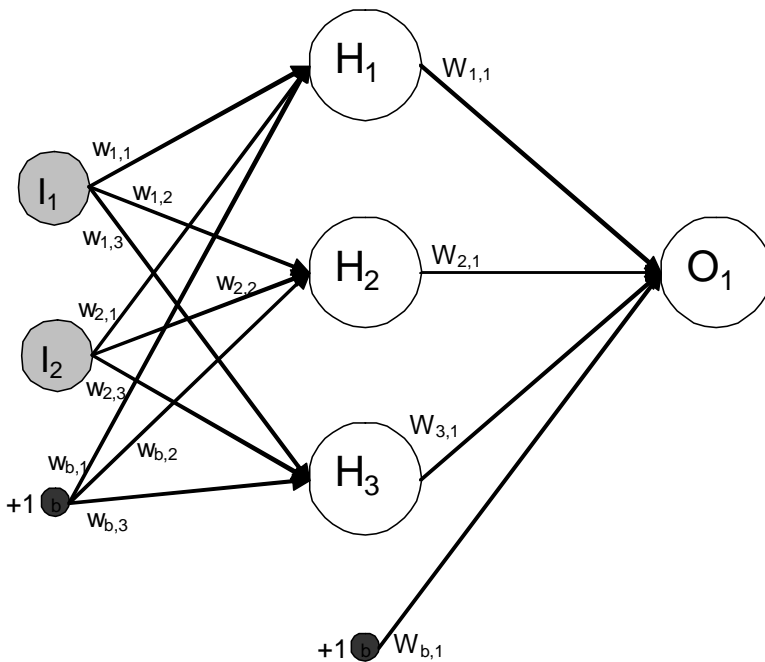


Table 5-5: Weights of the neural network in Figure 5-8

O ₁	W _{1,1}	-43.3732
	W _{1,2}	-38.7171
	W _{1,3}	-95.5844
	W _{1,b}	0.045141
H ₁	w _{1,1}	20.42473
	w _{1,2}	3.329065
	w _{1,b}	-3.36646
H ₂	w _{2,1}	143.4564
	w _{2,2}	-80.5905
	w _{2,b}	-9.28541
H ₃	w _{3,1}	-4.28717
	w _{3,2}	1.013559
	w _{3,b}	0.416504

Figure 5-8: This 2 layer neural network, with the weight in Table 5-5: Weights of the neuralTable 5-5 correctly classifies the messages in Table 5-3

⁶ Actually, because of company policies forbidding the use or unauthorized code, including libraries, only the tools for training the networks, the Encog Workbench, has been used and tested, in an offline computer. The output of the Workbench is just a text file, and the simple routine that uses the networks has been rewritten.

Table 5-6: Result of the application of the neural network in Figure 5-8 to the features extracted from the messages in Table 5-3. Columns like ΣX_i contains the activation value of the element (the weighted sum of its inputs), whilst the columns X_i that immediately follows contains the output of the same element (result of the activation valued applied to the activation function). The last column contains the sign of the output value of the network, and is exactly the same as the desired input.

ID	I_1 : Money Feature	I_2 : Contract Feature	Type of Message	Desired Output	ΣH_1	H_1	ΣH_2	H_2	ΣH_3	H_3	ΣO_1	O_1	Out Sgn(O_1)
1	0.000	0.100	Offer	+1	-3.034	-0.995	-17.344	-1.000	0.518	0.476	36.432	1.000	1
2	0.125	0.000	Offer	+1	-0.813	-0.671	8.647	1.000	-0.119	-0.119	1.809	0.948	1
3	0.071	0.000	Offer	+1	-1.908	-0.957	0.961	0.745	0.110	0.110	2.208	0.976	1
4	0.000	0.111	Offer	+1	-2.997	-0.995	-18.240	-1.000	0.529	0.485	35.589	1.000	1
5	0.125	0.000	Offer	+1	-0.813	-0.671	8.647	1.000	-0.119	-0.119	1.809	0.948	1
6	0.143	0.143	Offer	+1	0.027	0.027	-0.305	-0.295	-0.051	-0.051	15.203	1.000	1
7	0.143	0.000	Contract	-1	-0.449	-0.421	11.208	1.000	-0.196	-0.193	-1.928	-0.959	-1
8	0.250	0.125	Contract	-1	2.156	0.974	16.505	1.000	-0.529	-0.484	-34.605	-1.000	-1
9	0.375	0.000	Contract	-1	4.293	1.000	44.511	1.000	-1.191	-0.831	-2.603	-0.989	-1
10	0.143	0.000	Contract	-1	-0.449	-0.421	11.208	1.000	-0.196	-0.193	-1.928	-0.959	-1
11	0.091	0.000	Contract	-1	-1.510	-0.907	3.756	0.999	0.027	0.027	-1.853	-0.952	-1

For multilayer feed-forward network weights have to be calculated iteratively:

1. Feed the network with a random input vector from the training set and calculate its output
2. Calculate the error, that is the difference between the desired output and the calculated one
3. Slightly adjust the weights to get a little lower error
4. Repeat to step 1 until the error goes below a specified threshold

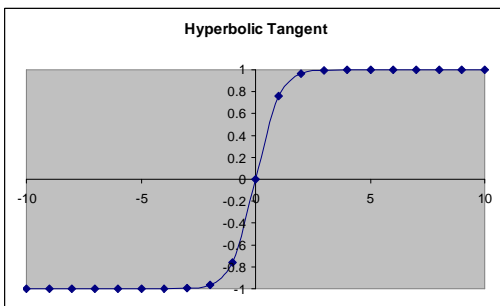


Figure 5-9: A possible activation function of a neural network (hyperbolic tangent function)

Because of this iterative “blind” process for weights calculation, for multilayer feed forward networks at least by now there is no way to interpret the result of the training. The network becomes a sort of black box, to trust on real situation because it has shown to succeed in a limited amount of training and test data. This may or may not be a problem, depending mostly on the importance of the decisions that has to be taken on the basis of the network result. In our case, misjudging a sentence would not result in any harm.

Of course the difficult part is point 3. The back propagation algorithm is probably the most famous one that does the job (genetic algorithms being another possibility). It is detailed in [Hertz-91], together with a formal demonstration of its validity.

Because of this iterative “blind” process for weights calculation, for multilayer feed forward networks at least by now there is no way to interpret the result of the training. The network becomes a sort of black box, to trust on real situation because it has shown to succeed in a limited amount of training and test data. This may or may not be a problem, depending mostly on the importance of the decisions that has to be taken on the basis of the network result. In our case, misjudging a sentence would not result in any harm.

5.3 Naïve Bayesian Classifiers

From Wikipedia: A naïve Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem with strong (naïve) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model".

To illustrate the idea with some examples, let's consider the two groups of sentences in **Error! Reference source not found.**, obtained searching Twitter for messages containing both the words "impresa" and "semplice". The first group is only made of sentences related to the brand ImpresaSemplice™ (the correct ones), whilst the other contains only unrelated ones. Let's call them the Correct and the Wrong sets respectively. Our classifier should be able to assign new sentences to one set or the other.

Table 5-7: Training set: some result from the search of the words "impresa" and "semplice" on Twitter. The first group of 5 messages talks about the brand ImpresaSemplice (Correct sentences), the last set of 4 does not (Wrong sentences).

	Correct
1	Tutte le novità da Impresa Semplice per la tua <u>azienda</u> arriveranno domani da Milano dove si riuniranno i #talenti
2	Autotre srl a ponzano ha scelto Impresa Semplice per la sua <u>azienda</u> con <u>TIM</u> senza problemi! Così può parlare e navigare illimitatamente!
3	Furti di Gasolio in <u>azienda</u> sempre più comuni! La soluzione è Localizz@ di Impresa Semplice!
4	domanda Come e quando e se pensa di <u>fare</u> Telecom una grande impresa non come oggi Impresa Semplice . Quale e' il futuro di <u>Tim</u> ?
5	La scelta migliore per la tua <u>azienda</u> è affidarti a Impresa Semplice di Telecom Italia. Scopri come <u>fare</u> .
	Wrong
6	intrappolata in pastoie burocratiche, se fosse più semplice <u>fare</u> impresa in Italia forse la situazione sarebbe meno grigia.
7	Purtroppo tutto è diventato finanza già da molti anni: è molto più semplice che non <u>fare</u> impresa .
8	per <u>fare</u> una famiglia ci vogliono coraggio, devozione e pazienza. non e' un impresa semplice . A volte viene perfino voglia di rinunciare
9	No, ma è un modo semplice per capire come mai una <u>azienda</u> strategica e sanata non potrà <u>fare</u> impresa in Italia

To begin with, let's focus on some words frequently present in both sets: "azienda", "fare" and "tim", and count how many times we find them in each of the two sets. The result is in **Error! Reference source not found.**

Table 5-8: Count of the number of times three specific words appears in the two sets of messages

	Correct	Wrong
<u>azienda</u>	4	1
<u>fare</u>	2	4
<u>tim</u>	2	0

By definition, if S and W are two random variables, $P(W=w | S=s)$ indicates the conditional probability that W takes on a certain value w if it's already known that S takes on value s.

Let S be the random variable representing the set (so S can assume values in {Correct, Wrong}). Let also W_0 , W_1 and W_2 be three random variable representing the presence in the sentence of the words “azienda”, “fare” and “tim” respectively (so they all can take on values from the set {0,1}, where “1” indicate presence and “0” absence).

With this definitions $P(W_0=1 | S=Correct)$ indicates for example the conditional probability of finding the word “azienda” ($W_0=1$) in a sentence belonging to the set Correct. We can estimate that value by simply dividing the number of sentences in the set Correct containing that word by the total number of sentences in that set, i.e. $P(W_0=1 | S=Correct)=\frac{3}{4}$.

In the same way we can calculate:

<u>azienda</u>	$P(W_0=1 S=Correct)=\frac{4}{5}$	$P(W_0=1 S=Wrong)=\frac{1}{4}$
<u>fare</u>	$P(W_1=1 S=Correct)=\frac{2}{5}$	$P(W_1=1 S=Wrong)=1$
<u>tim</u>	$P(W_2=1 S=Correct)=\frac{2}{5}$	$P(W_2=1 S=Wrong)=0$

The idea behind naïve Bayes classifiers is to use this values (calculated from the training set, i.e. from the two sets of sentences) together with the known values of the W_i variables (measured in the sentence to classify) to calculate the probabilities that that sentence belongs to the two sets, and assign the label (Correct or Wrong) according to the highest value of the two. In other words, first find the three values w_0 , w_1 and w_2 , and then:

Let $P_C=P(S=Correct | W_0=w_0, W_1=w_1, W_2=w_2)$

Let $P_W=P(S=Wrong | W_0=w_0, W_1=w_1, W_2=w_2)$

IF $P_C > P_W$ THEN Label=Correct

ELSE Label=Wrong

Before explaining the method, it's worthwhile notice that it would be possible, in theory, to make a direct estimation of those probabilities by simply counting the number of correct and wrong sentences for every possible combination of the random variables W_i . Anyway such approach would clearly require a training set of exponential size (proportional to 2^n , where n is the number of words considered). Using Bayes formula instead:

$$P(Y | X) = \frac{P(X | Y)P(Y)}{P(X)}$$

we can write:

$$P(S = Correct | W_0 = w_0, W_1 = w_1, W_2 = w_2) = \frac{P(W_0 = w_0, W_1 = w_1, W_2 = w_2 | S = Correct)P(S = Correct)}{P(W_0 = w_0, W_1 = w_1, W_2 = w_2)}$$

Assuming independence of the random variables W_i , we have:

$$P(W_0 = w_0, W_1 = w_1, W_2 = w_2) = P(W_0 = w_0)P(W_1 = w_1)P(W_2 = w_2)$$

and also:

$$\begin{aligned} &P(W_0 = w_0, W_1 = w_1, W_2 = w_2 | S = Correct) \\ &= P(W_0 = w_0 | S = Correct)P(W_1 = w_1 | S = Correct)P(W_2 = w_2 | S = Correct) \end{aligned}$$

So P_C becomes:

$$P(S = \text{Correct} | W_0 = w_0, W_1 = w_1, W_2 = w_2) = \frac{\prod P(W_i = w_i | S = \text{Correct})P(S = \text{Correct})}{\prod P(W_i = w_i)}$$

And in the same way P_W becomes:

$$P(S = \text{Wrong} | W_0 = w_0, W_1 = w_1, W_2 = w_2) = \frac{\prod P(W_i = w_i | S = \text{Wrong})P(S = \text{Wrong})}{\prod P(W_i = w_i)}$$

Noticing that the two expressions have the same denominator, in order to compare P_C with P_W we only need to calculate:

$$P_C' = P(S = \text{Correct}) \prod P(W_i = w_i | S = \text{Correct})$$

$$P_W' = P(S = \text{Wrong}) \prod P(W_i = w_i | S = \text{Wrong})$$

$P(S=\text{Correct})$ and $P(S=\text{Wrong})$ are called the prior probabilities of the events Correct and Wrong respectively, and can be estimated with their relative frequencies. Having 5 Correct sentences and 4 Wrong ones we have $P(S=\text{Correct})=5/9$ and $P(S=\text{Wrong})=4/9$.

A problem arise with the right part of the formulas when infrequent words are considered: if at least one word is not present in one of the two sets, estimating its conditional probability using the frequency gives a zero, and that makes the entire product to be zero too. In other words, it becomes very likely to be in the situation where both P_C' and P_W' are zero, making the classification of the sentence impossible.

To overcome this problem a simple formula called **additive smoothing** can be used. If N is the total number of sentences in a set and x_i is the number of sentences in the same set that contains the i^{th} word, then instead of the frequency $F=x_i/N$ the following fraction can be used:

$$P = \frac{x_i + \alpha}{N + \alpha d}$$

where d is the total number of words considered (i.e.: the size of the parameter vector), and α is an arbitrary parameter >0 . “[...] some authors have argued that α should be 1 (in which case the term add-one smoothing is also used), though in practice a smaller value is typically chosen”.

[from http://en.wikipedia.org/wiki/Additive_smoothing]

Error! Reference source not found. shows the additive smoothing estimation for our example compared, with the simple frequency estimation, having $d=3$ and using $\alpha=1/3$.

Table 5-9: Estimation of the conditional probabilities of finding a certain word in a certain set $P(W_i=1|S=s_j)$. The conditional probabilities of not finding the same word are obviously their complement: $P(W_i=0|S=s_j)=1-P(W_i=1|S=s_j)$

		Freq.	Additiv e Smooth		Freq.	Additive Smoothin g
<i>aziend</i> <i>a</i>	$P(W_0=1 S=\text{Correct})$	0.60	0.56	$P(W_0=1 S=\text{Wrong})$	0.25	0.27

<i>fare</i>	P(W ₁ =1 S=Correct)	0.20	0.22	P(W ₁ =1 S=Wrong)	1.00	0.87
<i>tim</i>	P(W ₂ =1 S=Correct)	0.40	0.39	P(W ₂ =1 S=Wrong)	0.00	0.07

Using the values in the **Error! Reference source not found.**, let's now try to classify the two test sentences of **Error! Reference source not found.** Sentence 10 only contains the word “azienda”, so its parameter vector is (W₀=1, W₁=0, W₂=0).

$$P_C^{10} = 5/9 * 0.56 * (1 - 0.22) * (1 - 0.39) = 0.148$$

$$P_W^{10} = 4/9 * 0.27 * (1 - 0.87) * (1 - 0.07) = 0.015$$

Table 5-10 Two test sentences

10	<i>Offerta di lavoro: Agente Impresa Semplice: Glik SRL - Palermo - Glik srl, <u>azienda</u></i>
11	<i>Come e quando e se pensa di <u>fare</u> Telecom una grande impresa non come oggi Impresa Semplice. <u>Quale e' il futuro di Tim?</u></i>

The posterior probability of the sentence 10 belonging to the Correct set (0.148) is about ten time greater than the one of it belonging to the Wrong set, so the sentence is correctly classified as a Correct one. Sentence 11 contains the words “fare” and “tim”, so its parameter vector is (W₀=0, W₁=1, W₂=1).

$$P_C^{11} = 5/9 * (1 - 0.56) * 0.22 * 0.39 = 0.026$$

$$P_W^{11} = 4/9 * (1 - 0.27) * 0.87 * 0.07 = 0.020$$

In this case too, being P_C>P_W the sentence is correctly classified as a Correct one. As expected the margin is much smaller than before, as the word “fare” very frequently appears in the Wrong set. A sentence containing only the word “tim”, with parameter vector (W₀=0, W₁=0, W₂=1) would result in:

$$P_C = 5/9 * (1 - 0.56) * (1 - 0.22) * 0.39 = 0.091$$

$$P_W = 4/9 * (1 - 0.27) * (1 - 0.87) * 0.07 = 0.003$$

Again it would be classified as Correct with a very great margin.

5.4 Results

The parameter used here to evaluate the classifier are the classical Precision and Recall. First of all, lets give some definitions: From [Tan-06]:

- True Positives (TP): the number of positive examples correctly predicted
- False Negative (FN): the number of positive examples wrongly predicted as negative
- False Positive (FP): the number of negative examples wrongly predicted as positive
- True Negative (TN): the number of negative example correctly predicted

Again from [Tan-06]:

“Precision determines the fraction of records that actually turns out to be positive in the group the classifier has declared as a positive class. [...] Recall measures the fraction of positive examples correctly predicted by the classifier”

So

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

For the purpose of testing the ensemble classifier, two different method has been used: a traditional 10 fold cross validation and a second, totally new test set, with example collected over a longer period of time (with respect to the training set).

The 10 fold cross validation method works as follows: for 10 times a random subset of about 10% of the examples has been removed from the training set, to be used for testing. The remaining 90% of examples has then been used to train the classifier. Finally, the average of the evaluation parameters (precision and recall) over the 10 folds trial has been calculated.

About the other test set (let’s call it the extended test set), it has been made with the purpose of testing the classifier performances over time, to measure the negative influence of the unavoidable changes in subject and writing styles. Whilst the training set has been made with messages collected over only two month of the year (February and March 2013), the extended one spans over ten months of the same year, obviously avoiding messages present in the training set. To be precise, a set of 1000 sentences, randomly extracted from over 300000 messages downloaded from February to October 2013, has been manually tagged using only four tags, and the resulting distribution is shown in **Error! Reference source not found.** The choice of the four tags, limited in number for time reasons, has been related to the size of their respective training set: with a sample of the performance of the classifier with different training set sizes we can have an idea of how large it need to be. We notice that the presence of the four selected labels inside the test set is quite rare, ranging from a 2.4% to a 9.7%⁷. This means that, in real situations, **we can expect the number of interesting messages to be below the 10% of the total messages downloaded.**

Name	TAG	Level of Taxonomy	Description	Presence % (percent of messages)	Training Set Size
Business	BUS	Top	Offers, Contracts, Activation, Cessation, Portability	9.7%	281
Quality	QUA	Top	Coverage, Line quality and malfunctions, difficult setup and delays.	7.6%	208
Internet, Tech. & Devices	ITD	Top	Device characteristics and malfunctions, Official Operator App and Web Site	4.1%	91
Bad Behavior	HUM-H	Below “Human Factor”	Human bad behavior (operators, agents, ...)	2.4%	35

⁷ The other messages are mostly silly affirmations or jokes of no practical use. Being able to use someone else unprotected wifi connections seems to be a common reason for great pride.

Table 5-11 Extended test set: 1000 messages randomly chosen (form Feb to Oct 2013) and manually labeled with 4 selected labels. As can be seen, most of the messages have had no label at all. The size of the training set for each of the label, measured in sentence fragments, is also reported in last column.

The result of the 10 fold cross validation test is given in Table 5-12, and that of the validation with the extended test set in Table 5-13.

Label	Decision Tree		Neural Network		Bayesian Classifier		Ensemble	
	Prec%	Rec%	Prec%	Rec%	Prec%	Rec%	Prec%	Rec%
@BUS	57.59	68.83	68.59	57.08	72.84	53.63	75.9	60.12
@EVT	47.21	38.76	71.64	31.81	69.16	31.53	69.03	34.31
@HUM	59.42	67.6	64.75	50.37	72.67	56.99	71.95	53.68
@ITD	41.01	35.64	31.58	23.47	45.43	23.61	51.79	22.45
@OTH	57.99	50.6	74.75	47.67	76	44.1	79.38	46.6
@QUA	49.17	62.93	59.12	58.91	66	64.16	65.98	64.07
@BUS-C	23.58	32.73	35.5	20.52	51.65	21.06	50.48	16.85
@BUS-O	28.81	27.29	25.23	12.61	29.74	12.33	29.15	9.49
@BUS-P	43.52	29.66	41.91	16.39	47.34	31.26	57.97	25.43
@EV002	32.35	45.83	75	59.16	58.32	61.66	74.16	59.16
@EV003	23.11	29.66	35	16.49	0	0	40	13.16
@HUM-C	19.85	21.5	0	0	25	5.75	0	0
@HUM-H	11.38	13.08	0	0	14.16	10.35	10	1.42
@HUM-N	12.21	25.71	23.33	9.51	30.83	17.84	40	16.18
@HUM-P	6.75	15.33	5	2.5	5	5	0	0
@HUM-R	57.4	75.48	65.31	64.91	71.49	75.96	71.17	74.95
@ITD-D	0.25	10	24.16	16.66	34.16	18.32	30	9.16
@ITD-M	0.38	10	5	2.5	15	13.33	10	3.33
@ITD-W	5.64	13.33	19.16	17.33	0	0	3.33	2
@OTH-P	66.68	86.25	77.77	77.66	87.5	73.07	79.38	77.66
@QUA-C	28.95	34.4	55.83	25.88	48.95	45.06	64.21	32.97
@QUA-L	19.74	33.54	35.16	19.1	45.49	47.18	52.94	29.73

Table 5-12: Results of the 10 fold cross validation process, with details of the single algorithms on the first columns and results for the ensemble on the last two.

Tag	Extended Test Set								10 Fold	
	Decision Tree		Neural Network		Bayesian Classifier		Ensemble		Ensemble	
	Prec%	Rec%	Prec%	Rec%	Prec%	Rec%	Prec%	Rec%	Prec%	Rec%
@BUS	85.7	49.4	57.2	60.8	81.6	50.5	80	53.6	75.9	60.12
@ITD	60	14.6	41.1	34.1	50	34.1	43.7	17	51.79	22.45
@QUA	73.3	43.4	51.4	47.3	54.9	51.3	64.7	43.4	65.98	64.07
@HUM-H		0	0	0	100	4.1		0	10	1.42

Table 5-13: Result of the validation process over the Extended test set. The last columns reports for comparison the results of the 10 fold process (ensemble part only)

6 Sliding Windows Classifier (SWC)

6.1 Introduction

Interesting things found on the web are not always in the form of messages of 140 characters maximum length. They may present themselves hidden inside blogs, Facebook pages or forums, and we would like to be able to capture them all.

The ensemble classifier we have described so far takes as input a normalized statistic of the feature of a certain text, and no matter how long the text is, we expect that similar statistics will give similar result. But of course, if the piece of text we are looking for is inside a much longer document that talks about all sorts of other things, the statistic change, and we expect the classifier to perform much poorly (as in fact it does). We can in theory solve the problem by slicing the document into coherent segments and calculate the statistics on them. This is called a “text segmentation” approach, and it is quite a complex problem itself. Hearst for example (in [Hearst-94]) uses lexical cohesion to partition expository texts into multi-paragraph segments that reflect their subtopic structure. His approach is called TextTiling, and it basically works by considering pairs of adjacent groups of tokens and measuring their relative lexical cohesion: “*The [...] method compares, for a given window size, each pair of adjacent blocks of text according to how similar they are lexically. [...] if two adjacent blocks of text are dissimilar, this implies a change in subtopic flow.*”

The graph in Figure 6-1 shows the result of the algorithm applied to a reference text (Stargazer). Below the graph there is the judgment of seven persons about the same text. The vertical lines in the graph indicates the points where the algorithm splits the text.

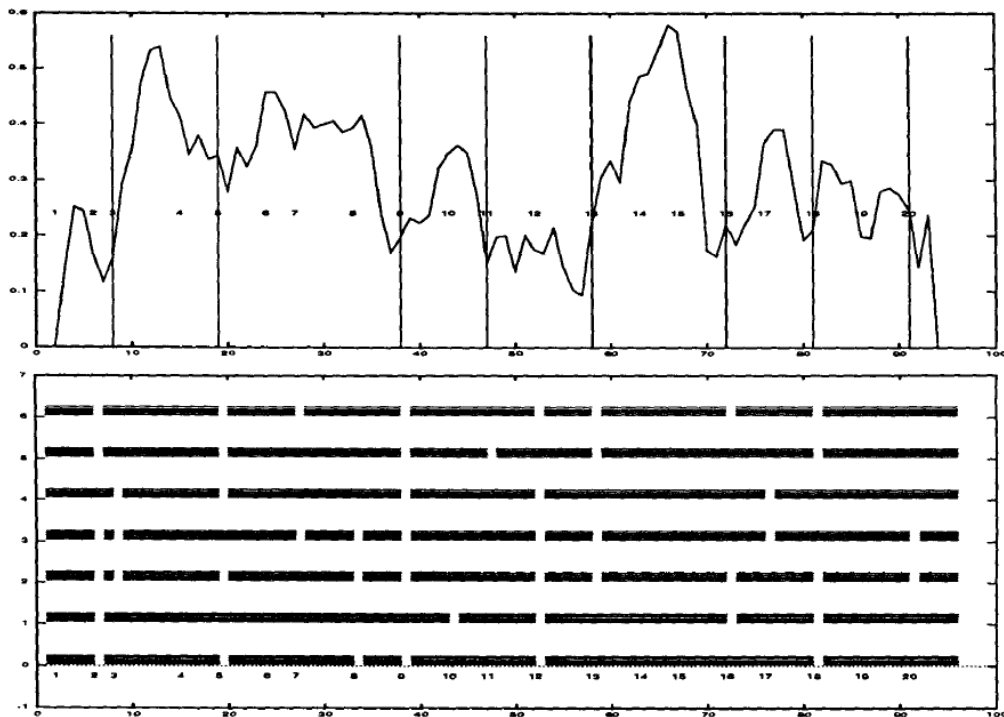


Figure 6-1 On top: result of Hearst TextTiling algorithm applied on the Stargazer text. Vertical lines near the points of minimum indicates where the algorithm splits the text . On bottom: judgment of seven readers on the same text.

The algorithm proposed here does not segment the text, but keep the idea of a sliding window of words and builds on it. It uses a neural network as it basic classifier, but in principle every other Multi Label classifier can be used in place. In brief: feature statistic are calculated for each position of the window inside the text. The $|L|$ outputs of the neural network are recorded for each position of the window, then the resulting graph, often quite messy, is analyzed to find zones where the networks express sharp results, and those are the output of the algorithm⁸.

6.2 Algorithm details

Before explaining the algorithm, let's first have a look of what its output is going to be: Figure 6-2 shows the result of applying three neural networks to the same sentence. Three neural networks have been trained in windows of sizes 5, 9 and 17 respectively. Then three sliding windows (of 5, 9 and 17 words) have been moved over the sentence, and for each position a normalized statistic of the features has been calculated (not shown in the graph). The windows statistics becomes the input of their respective neural networks, and the output of each network is calculated and showed in the graph. In this way, the three resulting columns of graphs have a line for each of the outputs of the neural network, with points in correspondence of each words of the sentence.

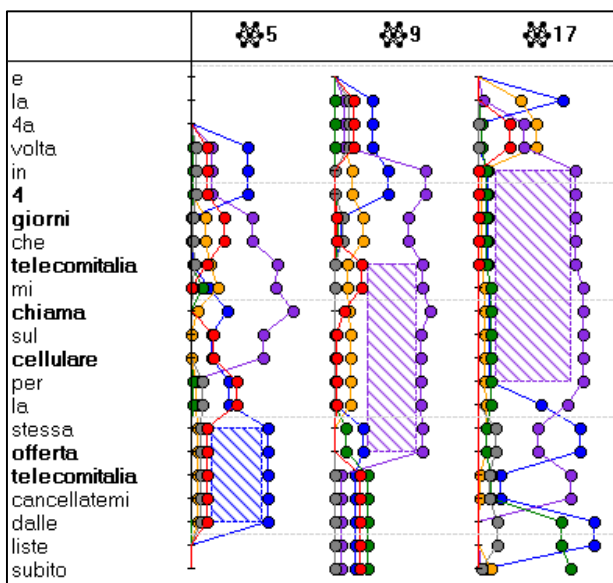


Figure 6-2. The Sliding Windows Classifier (SWC) applied to a Twitter message . The three different neural networks are applied to the sentence on the left column. Each neural network considers a fixed number of words at a time (in this case: 5, 9 and 17). The networks are “moved” along the sentence (vertically), and their outputs are calculated in each point and arbitrarily attributed to the central word of the set. In this way each network generate a graph from the sentence, with a curve for each network output. Rectangles represent zones of the graphs where the output is clear, i.e. where a single output is high and all the others are low. The area of these rectangles is taken as the actual output of the neural network set. Notice that some tags like “Offers” (the blue rectangle) are better found with smaller windows (5 words in this case), whilst other like Behavior (the purple rectangles) need larger ones (9 or even better 17).

Figure 6-3 should give the idea of the process, applied to a 5 words window containing the sentence fragment “*ho attivato vodafone unlimited ma*”. For simplicity of illustration the network has only four inputs { **Web**, **Malfunction**, **Companies**, **Variation** } and two outputs { **Business**, **Quality of service** }. We count 2 **Companies** words and 1 **Variation** word, so the input vector before normalization is { 0,0,2,1 }, after normalization is { 0,0,2/3,1/3 } and the desired output vector is { 1,0 }

⁸ Actually, the algorithm considers multiple windows of different sizes at the same time. The union of their results is the actual result of the algorithm.

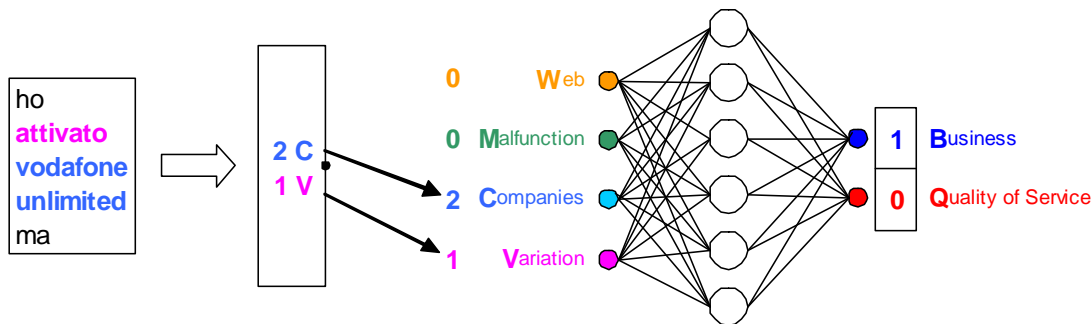


Figure 6-3 A neural network (4 inputs and 2 outputs) operating over sequences of 5 words. It means that 5 consecutive words at a time are considered. Features extracted from that words are counted and provides the input (after normalization, not shown here). The calculated output vector is attributed (arbitrarily) to the central word of the five.

The outputs of the algorithm are going to be a function of the area of the rectangles in the graph (more on this later). But a question comes first: how long should the window be? How many consecutive words should we consider to have the best result? A possible approach would be to try with different lengths and take the best one. Anyway, we suspect that different concepts require different length of text to be expressed, and so maybe there is no “better length” at all. As can be seen for example in Figure 6-2 and Figure 6-7, this is often exactly the case, so we are going to follow this slightly different approach: consider multiple windows of different length and combine their results by simply taking their union (the union of each network output set).

6.3 Training

To train each network we need transform the training set to take its window size into account. For each example of the original training set an entire set of new examples will be generated, with one element for every possible position of the window. In detail:

- for each sentence in the training set:
 - calculate the features
 - for each net
 - let N be the size of the window the net is going to be trained for
 - for each sequence of N consecutive words of the sentence:
 - calculate the input vector (counting for each feature the number times its greater than a certain threshold and then normalizing)
 - calculate the output vector, simply considering the central word of the sequence: a 1 is put in the output vector in the position relative to a certain label *l* if the central word of the sequence has been labeled with *l*
 - add the couple of input and output vectors to a list
 - collect all the couples of vectors in the list that has the same input vector and substitute them with a single couple where the output vector is the average of the group.

In Figure 6-4 the entire sentence “*ho attivato vodafone unlimited ma la connessione ad internet non funziona*” is shown again, with the desired output tags set to “**Business**” for the fragment “*attivato vodafone unlimited*” and to “**Quality of service**” for the fragment “*connessione ad internet non funziona*”. The arrow shows the first couple of I/O vectors, where the output vector is picked in correspondence with the central word of the 5-word group. In this example a total of 7 couples are found, that grouped by input vector results in a total of 5 couples to be added to the training set list (Table 6-1).

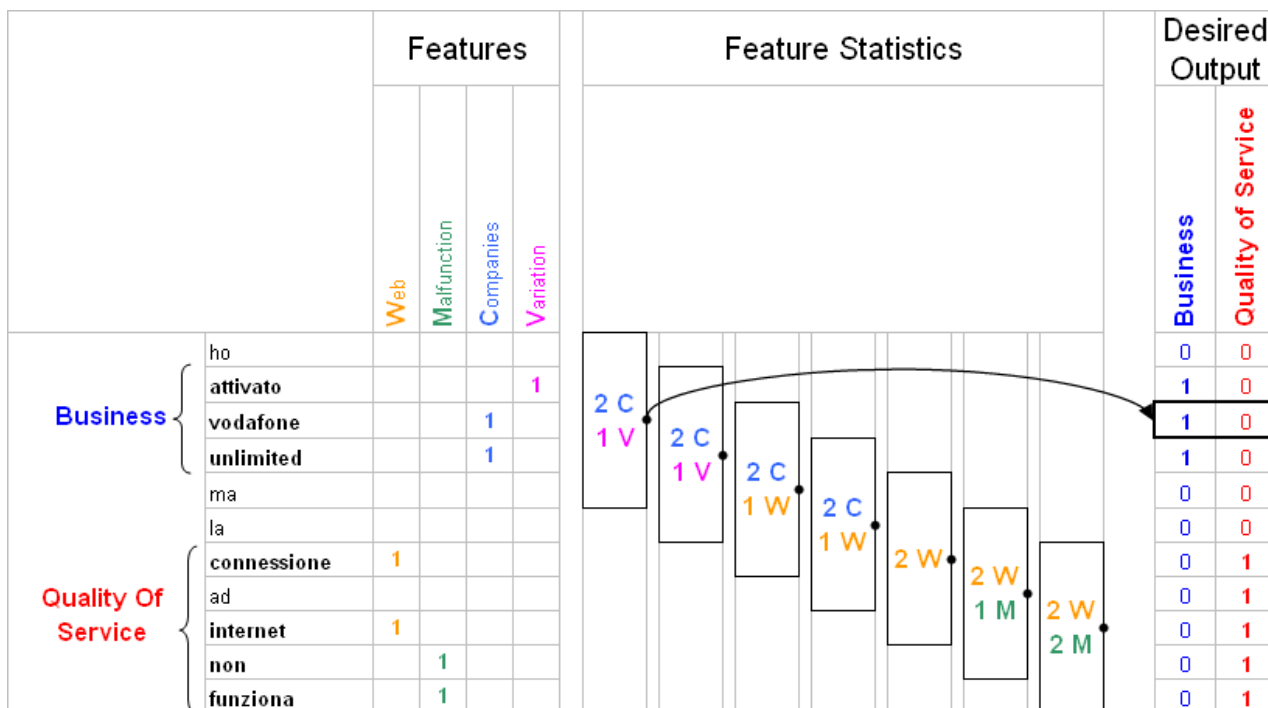


Figure 6-4: Training set generation for a 5-words neural network. Every group of 5 consecutive words in the sentence is considered. The count of the features become the input vector, whilst the desired values in correspondence with the central word of the 5 becomes the output vector.

Input				Output	
Web	Multifunction	Companies	Variation	Business	Quality of Service
		2	1	1	0
1		2		0	0
2				0	1
2	1			0	1
2	2			0	1

Table 6-1 Resulting training set for the network and sentence in Figure 6-4 (before normalization). The 7 records of the example has been grouped by input vector (becoming 5), and the output has been averaged.

6.4 Testing

To assign labels to a sentence we follow a process very similar to the one used to train the classifier:

- calculate the features of the entire sentence
- for every sequence of N contiguous word:
 - calculate the input vector (as before)
 - use the network to calculate the output vector and save it in a list (the graph G)
- extract the tags examining the graph

Figure 6-5 shows an example of such a graph G, (obtained from a working network, not a toy example), trained in sequences of 5 words (with a training set of about 800 sentences), applied to the same sentence used above. The sentence, beside its simplicity, has been chosen on purpose from the training set. The resulting graph is quite clean, and an obvious way to tag the sentence would be to set a threshold halfway in the graph (0.5) and assign to the sentence a tag for every curve that reaches that threshold (in this case, the blue and the red curves, respectively the “Business” and “Quality of service” ones).

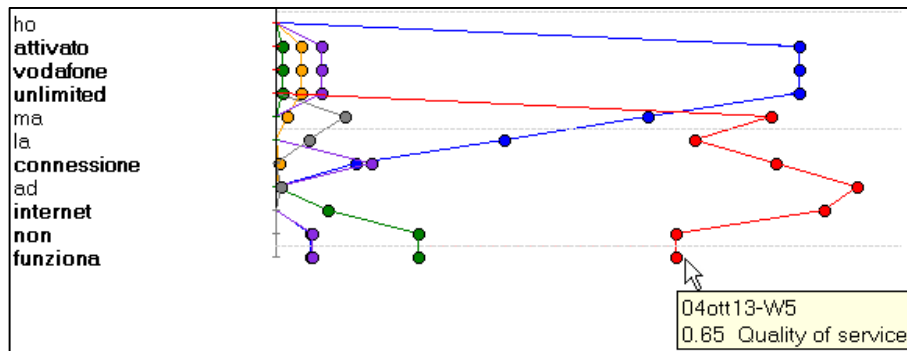


Figure 6-5: Output of a neural network (of 5-words window size) applied to the same sentence of Figure 6-4. The network has been trained with a training set of about 800 examples. The sentence itself is part of the training set, so the graph is exceptionally clean and the outputs are remarkably correct.

But with more complex or not in training set sentences, the graph usually comes out much more messy, and a different approach is needed. In Figure 6-6 for example, even the simple sentence “*balottelli è passato da vodafone a 3*” generates quite a confusing graph, with many unwanted tag values well above the threshold (assuming as before a threshold at 0.5).

The idea to solve the problem is quite simple: **find the regions of the graph where the network is “sure” about its output**, i.e. where the output is quite stable in moving from one sequence of N words to the next.

A way to implement this idea is to find the largest rectangles that lies inside the graphs between a line of a certain output (on its right) and whatever values is on its left, without touching any curve in the graph.

In Figure 6-6 the blue rectangle between the blue and the purple values that extends over the sentence fragment “*passato da vodafone*” correctly identify that fragment as “Business” related, whilst red (“Quality of service”) and purple (“Human Factor”) curves, that are both over the same threshold, not enclosing any rectangle do not qualify as valid tags.

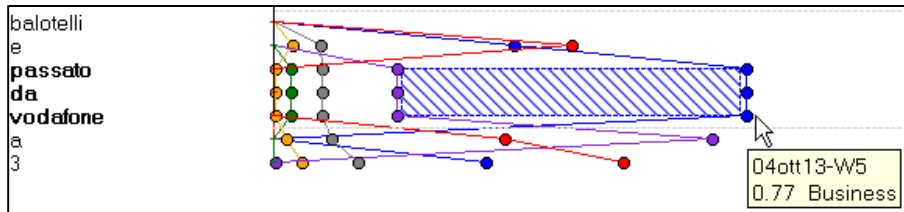



Figure 6-6: A SWC (of size 5) applied to a sentence that was not in the training set. Many outputs go high, resulting in a confusing graph. Anyway, there is a central part of the graph where a single output is above all other (the blue Business output, which is also the correct one), and this happens for 3 consecutive words, and so is taken as valid.

Obviously even with this approach a threshold must be decided: the minimum area that a rectangle must have in order of not to be discarded. This is the most important parameter we have to tune in order to get the desired precision and recall⁹. Lower thresholds result in having more positive cases, which in turn causes the precision to lower and the recall to increase. Empirically a value of 1.6 seems to work quite well, which means that we request a rectangle to be of at least 2 words and large 0.8, or 3 words and large 0.53, and so on.

As said before, sometime a concept is well expressed in few words, whilst some other times, even for the same concept, a longer piece of the sentence is involved. Figure 6-7 shows an example of “Line Quality” (red) tag that has been detected by a network trained in sequences of 17 words, whilst the graph for 5 and 9 word sequences were too messy to analyze (the number near the icon  on top of each graph is the size of the sequence of words used to train the network).

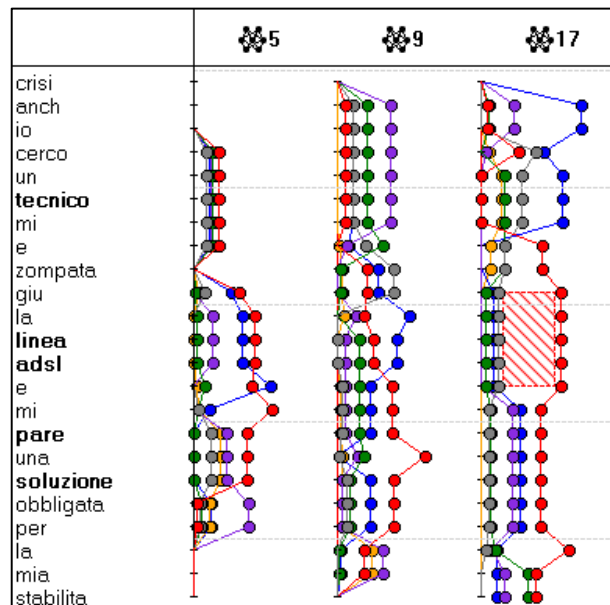


Figure 6-7: An example of category that has only correctly been recognized by the 17-words network of the SWC (the red rectangle on the right of the graph). The 5-words network found some correct high values (the red line), but has been “confused” by the blue line, whilst the 9-words network is even more messy and unuseful (for this sentence).

⁹ At this stage of the algorithm we desire a high recall, accepting a low precision. Appropriate filters in the next stage will be used to improve the precision.

6.5 Results

To test the SWC algorithm and compare it with the previous non-windowed form, we are going to use the same 10 fold cross validation used before, but with a variation: each message in the test set will be “diluted” within a longer text, same for all the messages, and chosen in such a way that, if classified by itself, it gives no positives in any of the two algorithm to test. Table 6-3 shows the 448 words text used for the test, taken from a free online novel. All the messages in the test set are inserted contiguously and in the same position of the dilution one¹⁰.

Tag	Ensemble (Non-Windowed)		SWC	
	Prec%	Rec%	Prec%	Rec%
@BUS	53.32	7.18	40.76	92.52
@QUA	5.92	14.54	36.32	79.84
@ITD	0	0	26.26	34.2
@HUM	0	0	42.76	94.1

Table 6-2 10 fold “diluted” cross validation of the Sliding Windows Classifier and the (non-windowed) ensemble algorithms. Test set text are inserted inside a 448 words unrelated text before features calculations.

Figure 6-8 shows the graphs of a message inserted after the 5th word of the dilution text. Because the features statistic is local (made inside of the two windows, one of 5 and the other of 9 words), the message is correctly detected (see the colored rectangles in between the lines of the graphs).

It is clear so that, although not very precise, the windowed method can still hold in presence of a very diluted text, whilst the non-windowed version can not.

Guidavo da solo, di notte in una strada alberata. I fari illuminavano la striscia d’asfalto, i rami e le foglie ai lati, nient’altro; sembrava che qualcosa avesse inghiottito il paesaggio. La mia anima era rapita e assente. I dolori, quelli veri, erano dimenticati, sarebbero tornati solo dopo l’arrivo. L’orologio segnava le tre e un quarto ed io ero veramente stanco. Non sarei dovuto partire, ma Giorgia voleva scaricarmi. Io stavo da lei da quasi due anni e così decisi di partire subito per Rimini, non avevo altro posto dove andare. Cercai di mantenere la calma, anche se la cosa mi faceva impazzire. Avevo conosciuto Giorgia perché a lei piaceva il jazz e veniva spesso ai concerti che facevo a Bologna. Non è che io fossi una star, ma mi aveva notato ed io avevo notato lei: alta, capelli lunghi e scuri, poi era sempre vestita veramente poco, insomma l’avrebbe notata anche una statua. Mi colpì che faceva la parrucchiera in provincia ed aveva una vera passione per il jazz e soprattutto per il pianoforte. Non riuscivo a vedere legami fra la tintura per capelli e Miles Davis, forse perché lui adoperava una parrucca. Dopo mesi chestavamo insieme abitavo già da lei e mi ero fatto un giro di locali dove suonavo spesso. Bologna mi era sempre piaciuta, ci sono tante occasioni per uno come me che si adatta. Le cose difficilmente vanno come le pensiamo. Quanta energia sprechiamo per prevedere quello che sarà, senza capire che non è possibile capire niente, soprattutto se i nostri piani riguardano altre persone. Giorgia mi disse che si era stancata di vedermi solo nei locali dove suonavo o che io partissi per chissà dove sempre quando lei aveva bisogno di me. Avevamo una vita troppo diversa, le poche volte che ci vedevamo facevamo l’amore, e dormivamo, insomma non potevo darle torto. Cercai di accorciare l’addio e le dissi che dovevo partire per Rimini. A trentadue anni stava iniziando un nuovo tema della mia vita, solo dopo avrei scoperto se era un tema ritmico ed agitato o melodico e cantabile. Avevo abbandonato la statale per una strada secondaria. La mia vecchia Ford non reggeva il ritmo delle Mercedes e Bmw che sfrecciavano a centocinquanta verso il divertimento, verso l’eccesso, verso la fine della notte: verso la riviera romagnola. Era l’ultimo sabato di giugno, forse il primo che faceva sentire il cambio dell’atmosfera. Il passaggio dalla quasi quiete all’improvvisa eruzione. Di lì a qualche giorno la riviera sarebbe esplosa di persone in cerca di forti emozioni e di felicità. Fiumi di alcol, droga e eccessi di ogni genere avrebbero attraversato quel lungo tratto di costa. Tutti quelli che vi erano diretti non potevano non essere eccitati.

Table 6-3 Text used to dilute the messages, taken from a free online novel: “Voci dalla rete 2” Longanesi & C

¹⁰ After the 5-th word

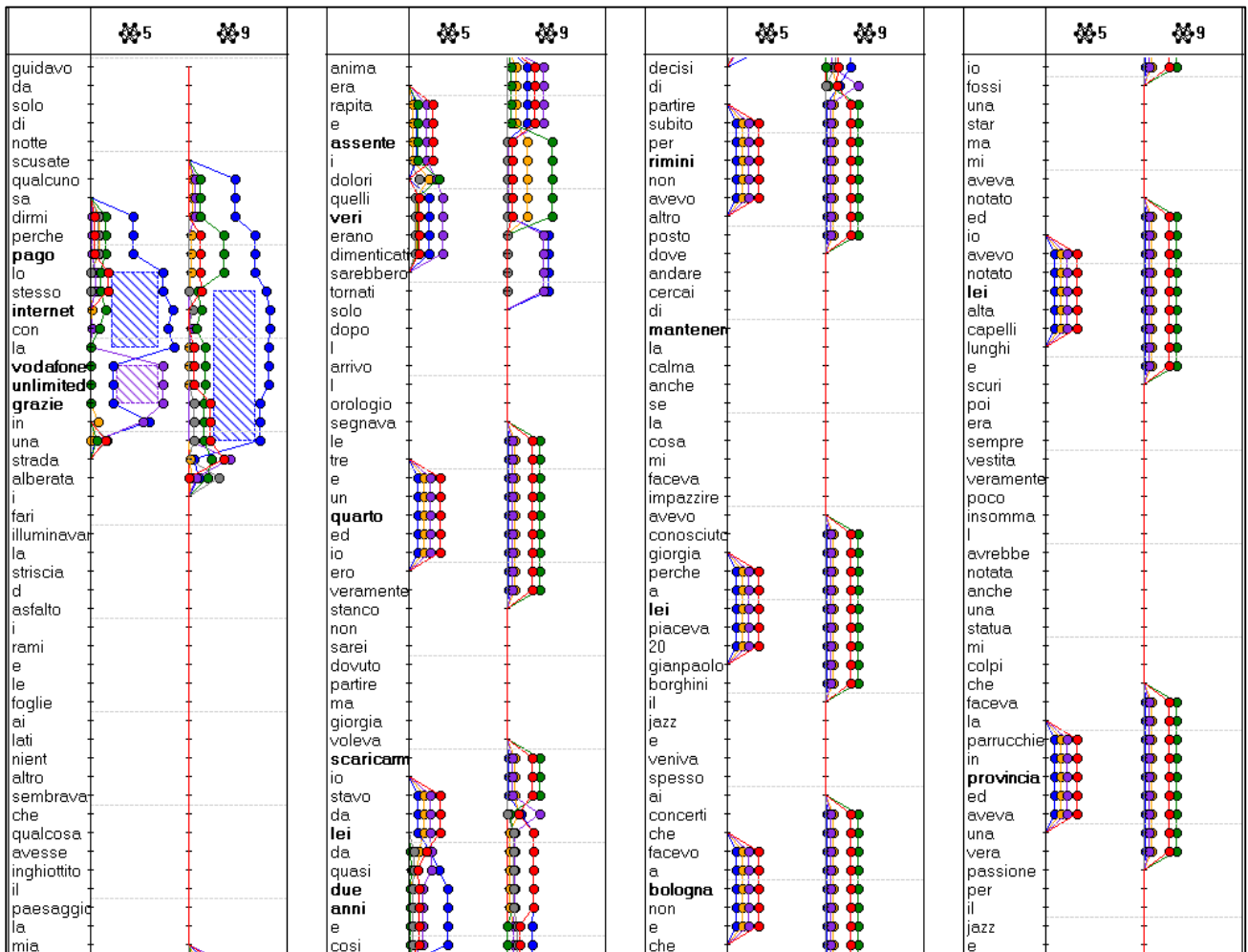


Figure 6-8: The graphs of SWC applied to a message diluted inside a much longer text. Message starts at the 6th word of the text. The message is correctly found, as colored rectangles over it indicates. (The message is “Scusate qualcuno sa dirmi perchè pago lo stesso internet con la Vodafone Unlimited? GRAZIE”)

7 Other Practical Applications

Although both the precision and the recall obtained so far are not exceptional, they are enough for the purpose of information retrieval, where a person is supposed to read the text after their classification (and in case correct it). But is the error still acceptable for other task as well? The next paragraphs examine this issue.

7.1 Some statistics

As stated before, two sets of tags are given to the messages:






- The Simple Tags, given by simply looking for the presence of some very specific words, and that identify, among other things, the telephone companies the message is talking about
- The Complex Tags, calculated by the neural networks and filtered with the naïve Bayesian classifiers, that refers to more complex concepts like Quality or Behavior

Even if the precision and the recall in assigning the Complex Tags are both not exceptional, assuming error are equally distributed, we can cross table the two sets to have a first view of the differences between telephone companies.

To this purpose, a set of about 400000 messages has first been automatically tagged. Then, the messages with Complex Tags different from Quality and Business have been removed, together with all messages that had multiple Simple Tags or multiple Complex Tags.

The result is a set of about 64000 messages, cross tabled in Table 7-1.

Table 7-1 Comparison between operators. Of about 400000 tweets, about 64000 has been automatically tagged with a Quality or Business tags. Telecomitalia and Fastweb have more Quality messages, whilst Vodafone and Wind more Business ones. We may be tempted to conclude that Fastweb and Telecomitalia have more problems with the quality of their lines, whilst Vodafone and Wind have more issues regarding contract and money. But further digging is of course necessary.

Complex Tags \ Simple Tags	Quality	Business	Tot	Quality %	Business %
	1234	2846	4080	30%	70%
	1835	15063	16898	11%	89%
	3115	14771	17886	17%	83%
	4122	11522	15644	26%	74%
	3720	6486	10206	36%	64%
Total	14026	50688	64714	22%	78%

As we can see, on the average we have 78% of Business messages against 22% of Quality ones. Looking at the difference between companies:

- Tre is close to the average
- Vodafone and Wind seems have more Business related messages. They can talk about problem or not, because in this category falls both the complaints for contracts and the request of informations for tariffs.
- Telecom Italia and Fastweb both have a number of messages regarding Quality above the average. These are normally complaints, as the client hardly manifests any happiness about the quality of the service (We can think he likes to keep it for himself). Anyway, messages regarding new coverage area, that of course are always positive ones, also falls into this category.

Clearly **this result must still be considerate just indicative, until the sub-categories of Quality and Business will be made precise enough.**

7.2 Graphs

Even with a not very high precision, graphs of the count of messages over a period of time can still be useful. Again we assume that mistakes in the assignments of the Complex Tags are independent from the assignment of the Simple Tags. In fact, peaks in the graphs quite often reveal interesting events, as the following to examples shows. Anyway, care must be taken if retweets are included in the count. Retweets are messages received from a user and forwarded, almost identical, to all his followers¹¹. In term of statistics, they normally should be included in the count, as they represent an agreement to an opinion of some sort. Anyway, a misclassified and high retweeted message will give rise to irrelevant peaks, so every peak in the graphs should be carefully examined before being considered en evidence of some event.

• ¹¹ From Twitter FAQ: “A Retweet is a re-posting of someone else's Tweet. Twitter's Retweet feature helps you and others quickly share that Tweet with all of your followers. Sometimes people type RT at the beginning of a Tweet to indicate that they are re-posting someone else's content. This isn't an official Twitter command or feature, but signifies that they are quoting another user's Tweet.”

7.2.1 Telecomitalia - Data network problems

On March 28 2013 Google search engine didn't work for user connected to Telecom Italia network. In that week Quality messages count raised from an average of 14 to a value of 61. (42 messages contains the string "Google", some are reported in

Table 7-3).

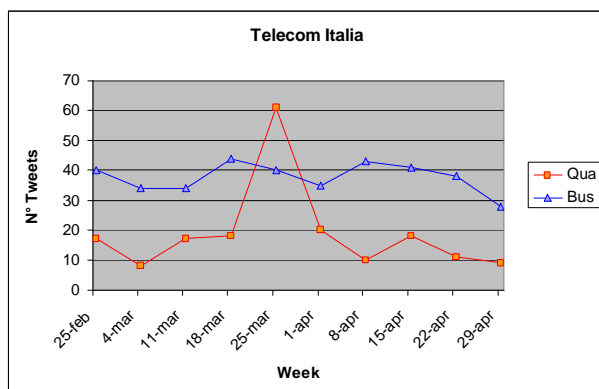


Figure 7-1: Peek of the Quality tagged messages (red line) on the week of March 28 2013, when Google search engine stopped working for Telecomitalia clients

	Qua	Bus
25-feb	17	40
4-mar	8	34
11-mar	17	34
18-mar	18	44
25-mar	61	40
1-apr	20	35
8-apr	10	43
15-apr	18	41
22-apr	11	38
29-apr	9	28

Table 7-2: Number of Quality and Business tagged messages for Telecomitalia in the period around March 28, 2013

Table 7-3: Some messages tweeted on March 28, 2013, when Google search engine stopped working for clients connected to Telecomitalia network,

#googledown per chi è connesso con @telecomitalia per problemi ai loro DNS. Usate l'IP 173.194.35.31 nel vs file hosts o #OpenDNS.

ok..ho aggiornamenti. Mi dicono che il **#googledown** è colpa dei dns di Telecom. E' sempre colpa di @telecomitalia #google

Evviva i problemi di #TelecomItalia. Con **#GoogleDown** credevo fosse morta tutta la mia rete di casa. Invece...

#GoogleDown forse problema di dns @telecomitalia, non potendo verificare chiediamo a loro. Che non abbiano pagato la bolletta? Può capitare.

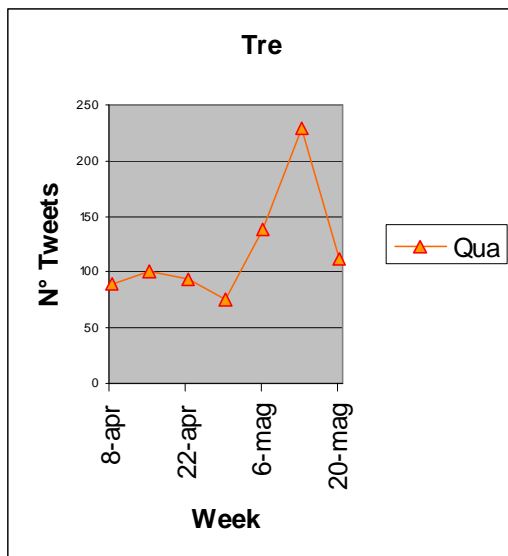
Stamattina la mia connessione ADSL é in crisi (pare sia un problema Telecom e di accesso ai servizi di **Google**). Spero di tornare tra voi.

Il motore di ricerca di **Google** non è raggiungibile da ADSL Telecom Italia a causa di un problema tecnico.

E' la rete Telecom a bloccare **Google**? O il down del motore è dovuto ai DNS?

7.2.2 Tre - Voice network problem

On May 16, 2013 the Tre voice network stopped working in Sicily and other regions of Italy. The count of Quality messages rises from an average of the period of 87 to a peak of 229.



	Qua
8-apr	89
15-apr	100
22-apr	93
29-apr	76
6-mag	138
13-mag	229
20-mag	112

Figure 7-2: Quality tagged messages for the Tre company, with a peak on the week of May 16 2013, when its network stopped working in Sicily and other region of Italy

Table 7-4: Number of Quality tagged messages for the Tre company in that period

Table 7-5: Some of the Quality tagged tweets regarding the Tre company on May 16 2013

Caltagirone la rete @Tre_It è morta...
@Tre_It continua a non prendermi il cel da stamattina. C'è qualche problema? Zona Davoli (CZ). ho bisogno urgente
@AlianteBlu @Tre_It confermo! Rende (Cs) senza servizio...
@Tre_It disservizio completo a Catania. Alcune zone con \nessun servizio\, altre invece in cui si ha \errore di chiamata\.
Buongiorno @Tre_It c'è un modo per parlare con voi da un numero non 3 o fisso? Non c'è linea!
@Tre_It salve . dale ore 8:00 di questa mattina tutta la provincia di palermo è senza rete ... ma è mai possibile ?
@Tre_It Confermato... problemi in Sicilia ma anche Sardegna
Si è verificato un problema di copertura ma è stato preso subito in gestione. Vi daremo riscontro su tempistiche di risoluzione quanto prima
@Tre_It da stamattina nessun segnale a Reggio Calabria. Solo chiamate d emergenza. #helpus

SICILIAINFORMAZIONI.COM

LOOKING FAR , LOOKING DEEP

CELLULARI, MISTERIOSO GUASTO FERMA LA COMPAGNIA

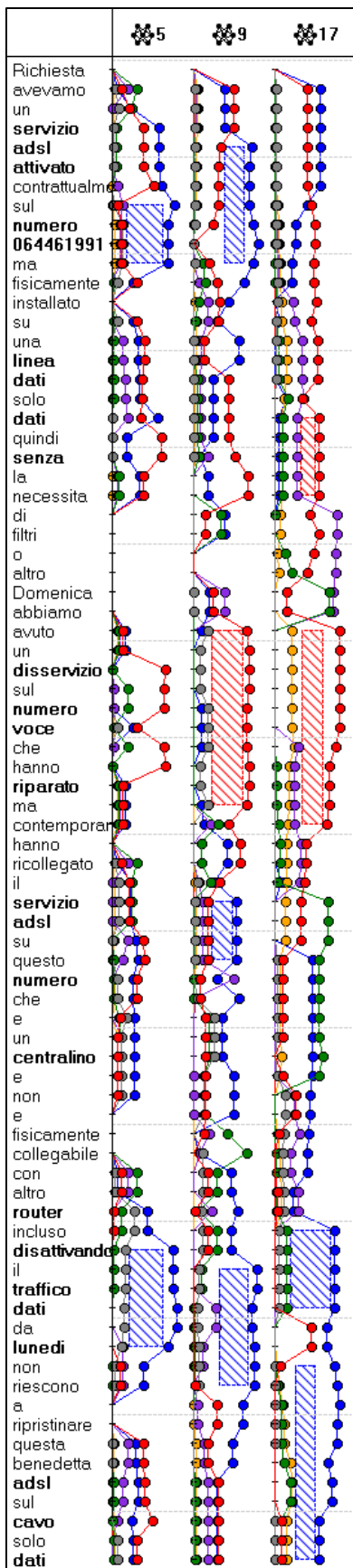
LA TRE NON FUNZIONA A PALERMO "TANTI DANNI, RISARCIRE SUBITO"

16 maggio 2013 - 14:26 - Cronaca Regionale



Per un attimo qualcuno è tornato indietro di 25 anni. Inizio di giornata tormentato per gli **utenti della compagnia Tre**. Telefoni senza rete a Palermo e in gran parte della Sicilia. L'azienda è stata tempestata di telefonate. Migliaia di cellulari che utilizzano le schede di H3G, dall'alba di oggi hanno cercato la rete senza trovarla, **mandando in bestia tantissimi palermitani**, rimasti "soli" per svariate ore. Una giornata atipica per migliaia di clienti che hanno dovuto fare a meno del telefonino.

Figure 7-3: An article on an online magazine talking about Tre network failure on May 16 2013.



7.3 Ranking

Even with a windowed classifier, that can work on text much longer than the text with which it has been trained, a problem arises: the longer the text is, the higher is the number of subjects that we are likely to find inside it. For this reason, long text are likely to have assigned most of the labels, and result all similar to each others. Anyway, different labels may appear in different proportions, so some simple ratios may give interesting informations.

For example, we can rank a set of records with the ratio between the Quality and the Business outputs of the Neural Network Layers. Hopefully, we will find the most Quality related texts on one side of the list and the most Business related one on the other, leaving the center of the list as “unclassified”.

Table 7-6 and Table 7-7 show the two extremes of the list resulting by applying this idea to a small bunch of Facebook messages, addressed by some clients to the Telecom Italia account. Needless to say, they are all complaints. As can be seen, despite his simplicity, this method is quite successful in finding what people is complaining about: Quality issues in Table 7-6 and Business issues in Table 7-7 are very well separated.

Unfortunately the other two tags (the Human Behavior and the Technology & Devices) are not precise enough yet. We argue that when they will be, a graph like the one in Figure 7-5 will be possible, where the text can be represented by points in a plane:

- The value of the X axis is given by the ratio Quality/Business
- The value of the Y axis is given by the ratio Behavior/Technology

Figure 7-4: A three neural networks set applied to a long sentence (longer than 140 characters)

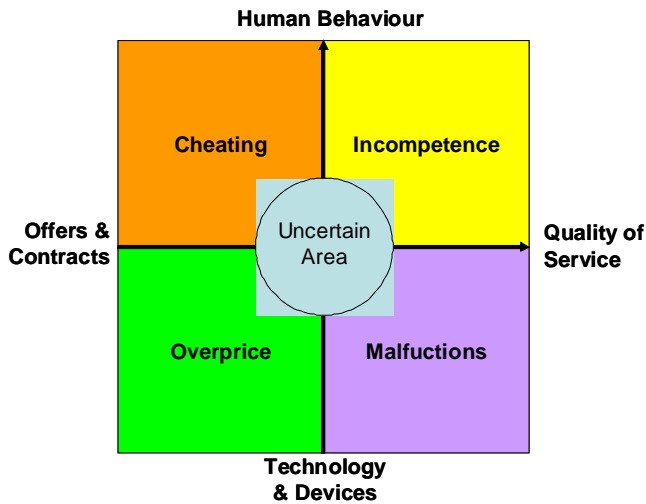


Figure 7-5: Possible graph, once other outputs work. The ratio Quality of service over Offer and contract is represented on X axis. The ratio Human Behaviour over Technologies and devices is on the Y axis. We expect to find quite specific and different messages in the four quadrants.

Assuming that all the texts represent some kind of problem or complaint:

- The texts in the top-right quadrant, with high Behavior and high Quality issues, will probably talk about the incompetence of some technician or operator
- The ones in the top left quadrant, with high Behavior and high Business issues, will probably be about some sort of cheating or at least misunderstanding between the seller and the client.
- The complaints in the bottom left quadrant, referring to both Business and Technology issues, will probably highlight overpricing or inadequacy of some device
- The last bottom right quadrant, with high Technology and high Quality problems, will finally contain complains about malfunctions of different sort.

Table 7-6: Top of the list of Facebook messages directed to the Telecomitalia account, ranked with the ratio Quality/Business. They clearly all express quality issues.

Quality	Rank
<i>Buonasera, mi rivolgo a voi perché ancora una volta al 191 non sanno risolvere il problema... Ho problemi con il server di uscita smtp per le mail... Compare sempre il messaggio "nessuna risposta dal server"... Al 191 mi hanno fatto modificare l'apn i.box.it ma senza successo... Anzi rivorrei i parametri originali visto che la posta non parte comunque, ed invece la connessione specialmente quella LTE è peggiorata tantissimo... Aspetto un vostro riscontro, saluti...</i>	3.215
<i>Buongiorno!!!! rieccomi qui, per un nuovo problemino: allora quando chiamano a studio ed il numero è occupato, chi chiama trova cmq la linea libera, non so se dipende dalla linea che abbiamo che è una isdn.... dato che la segretaria a studio è una nn avremmo la necessita di attaccare un cordless per passare la telefonata sotto.... Mi chiamate? nn ho line a sul cell qui, nn riesco a parlare cn il 191 chiamate vanessa grazieeeee!!!!</i>	1
<i>Richiesta: Buongiorno ho problemi con la visualizzazione delle fatture sul sito 191, mi dice: Email o cellulare non memorizzati su Ebill. ho scaricato il modulo per la richiesta attivazione E-Bill e inviato tramite fax. L'email è info@immobiliariesanizi.com e il numero di cellulare è 366*****. Entro quanto tempo è attivo il servizio? Grazie e buona giornata</i>	1
<i>ciao scusate o un problema con la play 3 mi fa giocare lento online mi uccidono sempre uncharted 3</i>	1
<i>Richiesta: avevamo un servizio adsl attivato contrattualmente sul numero 06*****, ma fisicamente installato su una "linea dati" (solo dati quindi senza la necessità di filtri o altro) Domenica abbiamo avuto un disservizio sul numero voce che hanno riparato ma contemporaneamente hanno ricollegato il servizio adsl su questo numero - che è un centralino e non è fisicamente collegabile con altro (router incluso) disattivando il traffico dati. da lunedì non riescono a ripristinare questa benedetta adsl sul cavo solo dati!</i>	0.918

Table 7-7: Bottom of the list of Facebook messages directed to the Telecomitalia account, ranked with the ratio Quality/Business. They clearly all express business issues (money and contract problems).

Business	Rank
<p><i>Salve, da novembre 2012 sono passata ad altro operatore, non sono più con Telecom Italia. il 6 dicembre 2012 ho ricevuto una vostra comunicazione di un credito a mio favore di 122,53 euro che mi sarebbe stato rimborsato entro 90 gg. siamo a luglio... i 90 gg sono passati e nonostante le numerose telefonate fatte al 191, non ho ricevuto ancora niente. I vostri operatori continuano a ripetermi che hanno aperto la pratica di rimborso... ma nessuno è in grado di dirmi che fine abbia fatto questa pratica e soprattutto quando riceverò questo rimborso. Gradirei sapere cosa bisogna fare perchè mi venga restituito quanto mi spetta.... grazie Tel 02***** CF 975***** n° pratica aperta per il rimborso: I-600*****</i></p>	0.001
<p><i>pratica numero: 7-149***** da contattare urgentemente. Sono stanca e delusa. Se non mi contattate subito attuerò una denuncia immediata nei vostri confronti. arriverderci Telefono 331***** P.iva 0727*****</i></p>	0.001
<p><i>Richiesta: Buongiorno, avanzo un credito nei vostri confronti di circa 123 euro (risulta da fattura emessa il 5 aprile). Ho visto che avete 90 gg di tempo per rimborsarla, ma volevo anche sapere in che modo mi verrà rimborsata. Grazie.</i></p>	0.001
<p><i>Richiesta: Come posso inserire il mio numero sull'elenco telefonico? Non so per quale ragione al passaggio da Poste Impresa a Impresa Semplice è stato cancellato.</i></p>	0.001
<p><i>Richiesta: salve ho sottoscritto un contratto a tariffa unica il 10/05 vi ho mandato via fax la copia dei documenti firmati. oggi ho ricevuto la vostra ultima fattura ancora con i consumi della vecchia tariffa. vorrei sapere se avete ricevuto tutto e se è andato in porto il mio nuovo piano tariffario.inoltre ancora non siete riusciti ad abbinare la mia p.i. al mio codice fiscale</i></p>	0.001
<p><i>Buongiorno vorrei un informazione:dovrei fare portabilità di 2 linee telefoniche mobili da Vodafone ad impresa semplice e la mia linea telecom Italia passarla a partita iva,ci sono offerte?inoltre e'possibile avere su ogni linea due iphone con rata su conto telecom Italia?grazie vorrei un po di info!! Ok il mio recapito e'335***** se possibile in orari serali grazie mille</i></p>	0.001

8 Future Developments: get a free training set!

As [Berry-97] stated: “A neural network is only as good as the training set used to generate it”.

Making a good training set anyway, with at least some hundreds examples for each category of sentence, is a real time consuming task, and consequentially a quite expensive option. But what if we could have not just a good, but an excellent training set totally for free?

The trick here is simple: we just have to take the last layer of the machine, the Naïve Bayesian Classifier, and move it into the web, allowing as many people as possible to use it for their own purpose. Web data mining is an activity of great and increasing interest for many companies and users. Provided we extend the domain of the machine to other areas of interest, such as food and real estate, the interest is expected to be great. If we build a good, although initially imprecise classifiers for each area of interest, and allow a free use of it on the web, the end user will be faced with the problem of refining its result.

So the user will be given a web tool to tag the wrong and correct results (simply that, a yes/no tag on otherwise already classified records). Then, pushing a button, a client side java applet can easily build a user “private” Bayesian classifier to clean the set.

But of course the user tags are not going to be lost! Saving them into the server, together with those of many other users, provides exactly the rich and cost free training set we were looking for!

That training set will obviously be used to improve the neural network and the default naïve Bayesian classifier of the machine (almost automatically), and can also be examined to look for improvements in the feature extraction layer (those need to be manual).

Also, we must expect that different users give different tags to the same record. Measuring for each category the entropy of the distribution of the yes/no users tags, we are probably going to find some category for which it is very high. This means that they are unclear, probably ill defined, and so hints for improvement of the category tree can also be derived from the users training sets.

In synthesis, this “Free analysis for free training set in return” is a kind of the virtuous cycle that may give great expectation about a rapid growth in term of precision and effectiveness of the machine itself.



Figure 8-1: A proposed virtuous cycle that can lead to a good training set at no expense. Users on the web improve they free analysis tagging wrong records for their (local) naïve Bayesian classifier. The server saves their tagging to improve the training set.

9 Conclusions

We have seen that the fraction of messages of some interest is in general well below the 10% of the total downloaded messages (the other being mostly silly affirmations or jokes of no practical use).

The two methods exposed here have shown to be able to extract this low percentage of interesting messages from the thousands daily downloaded from Twitter servers, with quite a good precision and a good recall, at least for the higher levels of the taxonomy of subjects adopted here. For some topics anyway the training sets are yet not big enough, so precision and recall are also yet not exceptional.

Besides selecting the interesting Twitter messages to read, which was the major aim of this thesis, the labeling system proposed here has also shown to be useful to classify messages from other sources and of different lengths, to make comparison between telephone companies and to rise alerts on events to be investigated further. Of course a lot of work has still to be done to improve both the training sets and the feature extraction part of the method, especially to improve precision and recall for the rarest labels of the taxonomy. Certainly far to be perfect, still quite a useful tool anyway!

10 Appendix 1: Essential OAuth Knowledge



Figure 10-1: OAuth logo

According to the official web site [oauth.net] OAuth is “An open protocol to allow secure authorization in a simple and standard method from web, mobile and desktop applications.”

So it's not used just in Twitter. On the contrary, at the moment Wikipedia lists more than 50 well known service providers that use it, including among the others: Amazon, Facebook, Google, LinkedIn, Microsoft, PayPal, Twitter and Yahoo!

[<http://en.wikipedia.org/wiki/Oauth>]

One of his main features, according to Twitter FAQ pages, is that it [...] allows users to approve application to act on their behalf without

sharing their passwords. [<https://dev.twitter.com/docs/auth/oauth/faq>]

The analogy often used to explain this thing is the one of the valet key, a special key that most expensive cars are provided with, that allows only limited use of the car itself (like moving for just few kilometers at low speed with the sole purpose of parking), restricting other unwanted functionalities (like running at high speed, or using the car phone)

Before protocols like OAuth were introduced, the only way for a client program (the valet) to interface with a server (the car) on behalf of somebody else (the owner of the car) was to acquire his credentials (the only keys, i.e. the username and password), with two obvious main drawbacks:

- There was a risk of potential damages caused by the client program (or programmer) that, quite possibly, could use the same credentials for different purposes
- It was impossible to revoke the access to a specific client without affecting the others (the password had to be changed)

OAuth solution, actually the first part of the OAuth specifications, starts by defining three rules and three sets of credentials. The rules are:

- The service provider (a server containing some resources)
- The resource owner, or user (normally a person)
- The consumer, or client, that is not a person but a software that interacts with the server, requesting or modifying the resources on behalf of the owner.

The sets of credentials are:

- The **consumer credentials**, that identify the consumer program to the server. During its development the programmer registers it to the server and receives backs two codes: the consumer key and its secret.
- The **access credentials**, also known as token credentials, that are used to identify the user in place of his username and password, that may be limited in scope and duration, and that can easily be revoked at any time.
- The **temporary credentials** (also known as request credentials), used between the client and the server to generate the access credentials.

The process to generate the access credentials depends on some factors, like:

- Where the client application runs (in a server through a browser, in a PC as a stand alone application or in a smartphone or other portable device)
- Whether or not a browser is available to the user on his device

Anyway the basic idea is always the same: somewhere during the data exchange between the client and the server, the user is directed by the client to a server page, where he authenticates himself and instructs the server to accept the client requests on his behalf. When the user does so, the server associates the client with the user, takes note of any user choice, and sends the client the access credentials to operate. Of course all this is done in such a way that the client never has to see the user credentials.

As already said, for the programmer point of view obtaining the consumer key and secret is a simple matter of using a browser to navigate to the Twitter developers web site, manually register the application and copy and paste the two returned codes somewhere in his program.

On the contrary, to get the access credentials a program interacting with Twitter servers must be written. Lucky enough, for trial purposes or if there is no need to deploy the application, Twitter automatically provides them “pre calculated” on the credentials of the user that has registered the application, in a page like that in Figure 10-8.

The second part of the protocol specification defines the method for making HTTP request using the credentials acquired in the first. This is the part that must be implemented, so let see it with an example. Let say that what we want is to download all the messages (“statuses” in Twitter terminology) that contain a certain given word. In the list of available API functions [<https://dev.twitter.com/docs/api/1.1>] we find that the appropriate function is the “GET search/tweets”.

Resource Information	
Rate Limited?	Yes
Requests per rate limit window	180/user 450/app
Authentication	Required
Response Formats	json
HTTP Methods	GET
Resource family	search
Response Object	Tweets
API Version	v1.1

Twitter online documentation has a page for every function in its API, specifying the characteristics of the function, like HTTP method and rate limits (see Figure 10-2), together with the list of all parameter it accepts. (Actually there are two sets of API, of which we are using the “REST” one, the other being the streaming one).

Before writing any line of code it’s a good thing to take advantage of the tool Twitter has made available online for testing purposes, which is able to generate correct API calls ready to be used (although for a limited period of time, due to a mandatory time stamp field), or simply to be compared with those generated by the program (see Figure 10-3). Using the tool, in the “cURL” section of the response (Figure 10-4) we discover that we have to generate a HTTP GET

Figure 10-2: Example of informations about a Twitter API function (in this case, the GET search/tweets function)

request to the url <https://api.twitter.com/1.1/search/tweets.json> adding a parameter at its end (q=telecomitalia), and inserting 7 other parameters in its header. This can easily be done in .NET using `HttpRequest` objects, as seen in Figure 10-5:

- An object of type `HttpRequest` is made by calling the static function `Create()` of the class `WebRequest` and type casting the result to `HttpRequest`.
- The HTTP method GET is specified assigning the string "GET" to the field `Method` of that object.
- Headers are simply added using the method `Add` of the field `Headers`.
- Network access credentials are specified if necessary.

The request is then sent over the net calling `GetResponse()`, and the result is returned in a `WebResponse` object, that in turn contains a stream with the text in json format.

Clearly now the only difficult part is to build the header with all the parameters, signature included.

To begin with, the `oauth_consumer_key` is the public part of the consumer credentials, whilst the `oauth_token` is the public part of the access credentials. The `oauth_nonce` is a random string whose purpose, together with the `oauth_timestamp`, is to avoid what is called a “replay attack”: the same request being used in identical form multiple times.

Figure 10-3: Twitter OAuth tool. Allows to generate the correct parameters of a request, for testing purposes.

Request URI:

The full URI, without parameters. For example: https://api.twitter.com/1/statuses/home_timeline.json

Request query:

The parameters for your request. For example: `include_entities=true&page=2`. Note these parameters will be sent on the querystring for GET requests, and in the request body for POST requests.

[See OAuth signature for this request](#)



Figure 10-4: A section of the output from Twitter OAuth tool

cURL command	<pre>curl --get 'https://api.twitter.com/1.1/search/tweets.json' --data 'q=telecomitalia' --header 'Authorization: OAuth oauth_consumer_key="s7Hhj09duWBCIJ5bNpCFg", oauth_nonce="9a13570151dbb5e5f15c83ce7053cc5c", oauth_signature="yOTyv3qRmstb0iZSErRM%2BRSZ9Sg%3D", oauth_signature_method="HMAC-SHA1", oauth_timestamp="1381764999", oauth_token="871962258-GFG0khV1i6gBYeY2wZDsGfqa2a0T2OdHaktldkB", oauth_version="1.0"' --verbose</pre>
--------------	--



Figure 10-5: Output from OAuth tool embedded inside a C# code

```
HttpWebRequest req=(HttpWebRequest)WebRequest.Create(
    "https://api.twitter.com/1.1/search/tweets.json?q=telecomitalia");
req.Method="GET";
req.Credentials= CredentialCache.DefaultCredentials;
req.Proxy.Credentials= CredentialCache.DefaultCredentials;
req.ContentLength = 0;
req.ServicePoint.Expect100Continue = false;
req.Headers.Add("Authorization",
    "OAuth "+
    "oauth_consumer_key=\"s7Hhj09duWBCIJ5bNpCFg\", "+
    "oauth_nonce=\"4ea3855020fa70facc7f9a2e05f5d94d\", "+
    "oauth_signature=\"XT8ZCjX01t9ND7BxG1jtL%2FP4TBI%3D\", "+
    "oauth_signature_method=\"HMAC-SHA1\", "+
    "oauth_timestamp=\"1381753453\", "+
    "oauth_token=\"871962258-GFG0khV1i6gBYeY2wZDsGfqa2a0T2OdHaktldkB\", "+
    "oauth_version=\"1.0\"");
WebResponse res=req.GetResponse();
StringBuilder sb=new StringBuilder();
using(Stream st= res.GetResponseStream())
{
    int b;
    while((b=st.ReadByte())>=0) sb.Append((char)b);
}
```

According to Twitter documentation: “*The `oauth_timestamp` parameter indicates when the request was created. This value should be the number of seconds since the Unix epoch at the point the request is generated [...]. Twitter will reject requests which were created too far in the past, so it is important to keep the clock of the computer generating requests in sync with NTP.*”

[<https://dev.twitter.com/docs/auth/authorizing-request>]

(NB: NTP stands for “Network Time Protocol”, and it’s actually a protocol, not a specific time server. Anyway, according to Wikipedia: “*NTP can usually maintain time to within tens of milliseconds over the public Internet*”, so being the timestamp expressed in seconds, any time server should suffice.)

Unix epoch is 1 January 1970, so assuming the system clock of the PC is correct, a C# command to calculate the timestamp can be the following:

Figure 10-6: C# code to generate a valid Twitter timestamp (assuming the system clock error not larger than a second)

```
timestamp = Convert.ToInt64(  
    (DateTime.UtcNow - new DateTime(1970, 1, 1, 0, 0, 0))  
    .TotalSeconds, CultureInfo.CurrentCulture)  
    .ToString(CultureInfo.CurrentCulture);
```

About signature methods, OAuth can potentially use three different types (PLAINTEXT, HMAC-SHA1 and RSA-SHA1), but Twitter servers can only handle HMAC-SHA1, and so that must be the value of the parameter `oauth_signature_method`. Also, by now the only OAuth version implemented by Twitter is the 1.0, so that value is mandatory for the `oauth_version` parameter.

Before proceeding with the most difficult part, generating the signature, a few definitions are useful.

UTF-8 Encoding: [from Wikipedia] *is a variable-width encoding that can represent every character in the Unicode character set.* The algorithm requires strings to be translated into UTF-8. Anyway “*the first 128 characters of Unicode [...] are encoded using a single octet with the same binary value as ASCII*”. If we assume for simplicity that we are only dealing with ASCII strings, no action is needed.

URL Encoding (also known as Percent Encoding) : [again from Wikipedia] “*Percent-encoding a reserved character involves converting the character to its corresponding byte value in ASCII and then representing that value as a pair of hexadecimal digits [...] preceded by a percent sign ‘%’*”.

URL encoding is clearly a very simple thing to do. Anyway, as noted in the very good Hueuniverse OAuth 1.0 online guide: “[...] *the parameters are URL-encoded [...] in a specific way that is often not fully compatible with existing URL-encoding libraries*”. And this in fact is true for the `.NET Uri.EscapeDataString()` function, that do not consider for example the exclamation mark to be a reserved character. Twitter documentation is not explicit about the set of characters to be considered reserved, but the following set (plus the space) seems to work quite well:

Figure 10-7: Characters that needs to be percent encoded in Twitter OAuth protocol

```
!*'();:@&=+$,/?#[ ]
```

Now, in order to calculate the signature:

- Collect all the parameters, sorted by their name and with the values URL-encoded

oauth_consumer_key	s7Hhj09duWBCIJ5bNpCFg
oauth_nonce	287F2549
oauth_signature_method	HMAC-SHA1
oauth_timestamp	1382019093
oauth_token	871962258-GFG0khV11i6gBYeY2wZDsGfqa2a0T2OdHaktldkB
oauth_version	1.0
q	telecomitalia

- Concatenate all the parameters together to form a single string. Each parameter's name is separated from the corresponding value by an '=' character (even if the value is empty), and each name-value pair is separated by an '&' character. Let's call this the **parameter string**.

```
oauth_consumer_key=s7Hhj09duWBCIJ5bNpCFg&oauth_nonce=3D3297F&oauth_signature_method=HMAC-SHA1&oauth_timestamp=1382019796&oauth_token=871962258-GFG0khV11i6gBYeY2wZDsGfqa2a0T2OdHaktldkB&oauth_version=1.0&q=telecom%20italia
```

- Create a new sting, called the “**signature base string**”, with the following pieces in the exact order:
 - The HTTP method in upper case (“GET”)
 - The ‘&’ character
 - The percent encoded base URL (the URL to which the request is directed, minus any query string or hash parameters)
 - Again a ‘&’ character
 - The percent encoded parameter string

```
GET&https%3A%2F%2Fapi.twitter.com%2F1.1%2Fsearch%2Ftweets.json&oauth_consumer_key%3Ds7Hhj09duWBCIJ5bNpCFg%26oauth_nonce%3D3D3297F%26oauth_signature_method%3DHMAC-SHA1%26oauth_timestamp%3D1382019796%26oauth_token%3D871962258-GFG0khV11i6gBYeY2wZDsGfqa2a0T2OdHaktldkB%26oauth_version%3D1.0%26q%3Dtelecom%2520italia
```

As suggested in Twitter documentation: “*Make sure to percent encode the parameter string! The signature base string should contain exactly 2 ampersand '&' characters. The percent '%' characters in the parameter string should be encoded as %25 in the signature base string.*”

- Generate the signing key by simply concatenate the percent encoded consumer secret, followed by an ampersand character '&', followed by the percent encoded token secret

- Finally, calculate the signature by passing the signature base string and signing key to the HMAC-SHA1 hashing algorithm. In .NET the class `System.Security.Cryptography.HMACSHA1` does the job:


```
using(HMACSHA1 hm=new HMACSHA1(K))
{
    O=hm.ComputeHash(B);
    string s=Convert.ToBase64String(O);
}
```

(here K and B are two bytes array containing the ASCII character of the signing key and of the signature base string respectively). The result is always a string of 28 characters:

```
7euApbjd13s7XfsXFgQH6H5+D2M=
```

TwitaTuto

[Details](#)
[Settings](#)
[OAuth tool](#)
[@Anywhere domains](#)
[Reset keys](#)
[Delete](#)



Twitter Application
<http://www.twitatuto.com>

Organization

Information about the organization or company associated with your application. This information is optional.

Organization	None
Organization website	None

OAuth settings

Your application's OAuth settings. Keep the "Consumer secret" a secret. This key should never be human-readable in your application.

Access level	Read-only About the application permission model
Consumer key	s7Hhj09duWBC1J5bNpCFg
Consumer secret	[REDACTED]
Request token URL	https://api.twitter.com/oauth/request_token
Authorize URL	https://api.twitter.com/oauth/authorize
Access token URL	https://api.twitter.com/oauth/access_token
Callback URL	None
Sign in with Twitter	No

Your access token

Use the access token string as your "oauth_token" and the access token secret as your "oauth_token_secret" to sign requests with your own Twitter account. Do not share your oauth_token_secret with anyone.

Access token	871962258-GFG0khV1i6gBYeY2wZDsGfqa2a0T2OdHaktldkB
Access token secret	[REDACTED]
Access level	Read-only

[Recreate my access token](#)

Figure 10-8: Application page on Twitter developer web site. Notice that an access token (token that identifies a user) has been provided for convenience for the user who registered the application.

Table 10-1 Parameters specific to the GET search/tweets function. They are all optional, except obviously the q parameter, that specifies the query string.

q required	A UTF-8, URL-encoded search query of 1,000 characters maximum, including operators. Queries may additionally be limited by complexity. Example Values: @noradio
geocode optional	Returns tweets by users located within a given radius of the given latitude/longitude. The location is preferentially taking from the Geotagging API, but will fall back to their Twitter profile. The parameter value is specified by "latitude, longitude, radius", where radius units must be specified as either "mi" (miles) or "km" (kilometers). Note that you cannot use the near operator via the API to geocode arbitrary locations; however you can use this geocode parameter to search near geocodes directly. A maximum of 1,000 distinct "sub-regions" will be considered when using the radius modifier. Example Values: 37.781157,-122.398720,1mi
lang optional	Restricts tweets to the given language, given by an ISO 639-1 code. Language detection is best-effort. Example Values: eu
locale optional	Specify the language of the query you are sending (only ja is currently effective). This is intended for language-specific consumers and the default should work in the majority of cases. Example Values: ja
result_type optional	Optional. Specifies what type of search results you would prefer to receive. The current default is "mixed." Valid values include: * mixed: Include both popular and real time results in the response. * recent: return only the most recent results in the response * popular: return only the most popular results in the response. Example Values: mixed, recent, popular
count optional	The number of tweets to return per page, up to a maximum of 100. Defaults to 15. This was formerly the "rpp" parameter in the old Search API. Example Values: 100
until optional	Returns tweets generated before the given date. Date should be formatted as YYYY-MM-DD. Keep in mind that the search index may not go back as far as the date you specify here. Example Values: 2012-09-01
since_id optional	Returns results with an ID greater than (that is, more recent than) the specified ID. There are limits to the number of Tweets which can be accessed through the API. If the limit of Tweets has occurred since the since_id, the since_id will be forced to the oldest ID available. Example Values: 12345
max_id optional	Returns results with an ID less than (that is, older than) or equal to the specified ID. Example Values: 54321
include_entities optional	The entities node will be disincluded when set to false. Example Values: false
callback optional	If supplied, the response will use the JSONP format with a callback of the given name. The usefulness of this parameter is somewhat diminished by the requirement of authentication for requests to this endpoint. Example Values: processTweets

Table 10-2: OAuth specific parameters. All mandatory in every Twitter function where authentication is required.

Consumer key oauth_consumer_key	The <code>oauth_consumer_key</code> identifies which application is making the request. Obtain this value from checking the settings page for your application on dev.twitter.com/apps .
Nonce oauth_nonce	The <code>oauth_nonce</code> parameter is a unique token your application should generate for each unique request. Twitter will use this value to determine whether a request has been submitted multiple times. The value for this request was generated by base64 encoding 32 bytes of random data, and stripping out all non-word characters, but any approach which produces a relatively random alphanumeric string should be OK here.
Signature oauth_signature	The <code>oauth_signature</code> parameter contains a value which is generated by running all of the other request parameters and two secret values through a signing algorithm. The purpose of the signature is so that Twitter can verify that the request has not been modified in transit, verify the application sending the request, and verify that the application has authorization to interact with the user's account
Signature method oauth_signature_method	The <code>oauth_signature_method</code> used by Twitter is HMAC-SHA1. This value should be used for any authorized request sent to Twitter's API.
Timestamp oauth_timestamp	The <code>oauth_timestamp</code> parameter indicates when the request was created. This value should be the number of seconds since the Unix epoch at the point the request is generated, and should be easily generated in most programming languages. Twitter will reject requests which were created too far in the past, so it is important to keep the clock of the computer generating requests in sync with NTP.
Token oauth_token	The <code>oauth_token</code> parameter typically represents a user's permission to share access to their account with your application. There are a few authentication requests where this value is not passed or is a different form of token, but those are covered in detail in Obtaining access tokens . For most general-purpose requests, you will use what is referred to as an access token . You can generate a valid access token for your account on the settings page for your application at dev.twitter.com/apps .
Version oauth_version	The <code>oauth_version</code> parameter should always be 1.0 for any request sent to the Twitter API.

11 Appendix 2: Feature List

11.1 General

11.1.1 Time

This feature tries to capture everything that can be related to time. Dates and time intervals are also searched inside the sentence.

Time Intervals	Months ^(*)	Days of Week ^(*)	Relative	Other
secondo, secondi, minuto, minuti, ora, ore, giorno, giorni, gg, settimana, settimane, mese, mesi, anno, anni	gennaio, febbraio, marzo, aprile, maggio, giugno, luglio, agosto, settembre, ottobre, novembre, dicembre	lunedì, martedì, mercoledì, giovedì, venerdì, sabato, domenica	altroieri, ieri, oggi, domani, dopodomani	tempo

(*) Abbreviations of 3 chars are also inserted in the dictionary

11.1.2 Delay

This feature tries to catch the idea of delay. For words that may refer to other things we look at the possible subjects, actually trying to recognize some common sentences used to express delay. If we find one of the main keywords in the sentence we look for at least one of the related possible words, in a defined direction (after, before or around) and not too far from the keyword itself (4 or 5 words).

Word	Regular Verb	Regular Plural
gingillarsi, tardi	ritardare, rinviare, rimandare, prorogare, procrastinare, tardare, attendere, aspettare, temporeggiare, indugiare, dilazionare	dilazione, ritardo, rinvio, procrastinazione, gingillamento, attesa

Common Sentences			
Look For	Where	Main Keywords	Examples
tempo, ore, ora, minuto, minuti, secondo, secondi, giorno, giorni, giornata, giornate, mese, mesi, anno, anni	around	Verbs: prendere, perdere Words: preso, perso	<i>“perdere del tempo”</i>
comodo, comoda, comodamente	after	Words: prendersela	<i>“Prendersela con comodo”</i>
lungo, lunghe	after	Verbs: tirare	<i>“tirava troppo per le lunghe”</i>
non	before	ancora	<i>“non è ancora arrivato”</i>

11.1.3 Lack

Here we would simply like to model the idea of something lacking, something missing.

Word	Regular Verb	Regular Plural
senza, niente, assenza, mancanza, nulla, indisponibilità, indisponibilità, zero, inesistenza	mancare, scarseggiare	carente, inesistente, scarso, carenza

11.1.4 Money

Words related to money and payments falls into this feature, together with numbers surrounded or attached to any symbol or name of currency (€, euro, euri, \$, dollaro, dollari, dollars, £, sterline, sterlina, pound, pounds)

High confidence			
Word	Regular Verb	Regular Plural	N-Gram
banca, banche, free, gratis, iban, iva, ricarica, ricariche, saldo	accreditare, acquistare, addebitare, costare, fatturare, incassare, noleggiare, pagare, rateizzare, rimborsare, riscuotere, sborsare, scontare, spendere, stornare	accredito, acquisto, addebito, amministrativo, assegno, bancaria, bolletta, bollettino, bonifico, conto, contributo, costo, costosa, costoso, credito, denaro, domiciliazione, esborso, extrasconto, fattura, fatturazione, gratuito, importo, incasso, insolvente, insolvenza, interesse, maxisconto, multa, noleggio, onere, pagamento, prelevamento, prelievo, prezzo, rata, rateizzazione, rimborso, riscossione, sconto, soldi, somma, spesa, storno, supersconto, versamento	conto corrente, estratto conto
Medium Confidence			
	Regular Verb	Regular Plural	
	prelevare, ricaricare, scalare, versare	cara	

11.1.5 Geo

Addresses and other geographical references often appear in two kinds of sentences: complaints about shipment and comments on coverage and line quality. At the moment we only search each word inside a dictionary of about 7000 city names, taking care of the fact that many of them are made of 2 or more tokens and can be written in different ways. A dictionary is made using the first token of each city name as key and all the remaining tokens as value (actually a list of possible remaining tokens is used, as there are in general many city names that begin with the same token). Whenever a word from the sentence is found inside the dictionary a check is made to see if all the other tokens of the city name are present, and if so all the matching words in the sentence are tagged as "Geo" (value=1.0). Differences

in stop words (*di, del, sul, ...*) are tolerated. Typical abbreviations are considered in groups of equivalences (for example: *s,san,sant,santo,santa* are all considered equivalents).

11.1.6 Shipment

Shipments of articles bought by phone or on the web are very often a cause of complaints. In this feature we consider words that can be related to shipment of physical objects. Because part of that same terminology is often used in “email” related sentences (*spedire, destinatario, recapito, mittente, ...*) we use a little trick here: we look at the “Web” feature already calculated before, and if around the word we are considering there are words with some “Web” meaning we decrease its “shipment” one.

In details, the algorithm is the following:

- Considering a window of ± 3 words around the “shipment” one (skipping the stop words) we count the numbers of other “shipment” words with a high, medium and low level of confidence. If we have enough other “shipment” words we set a “bRise” flag to true (at the moment enough means: at least 1 high or 2 mediums or 3 lows or 1 mediums and 2 lows)
- Inside the same window if we find any “web” related word we set a “bLower” flag to true
- If only the bRise flag is true we rise the “shipment” meaning of the word to 0.8
- If only the bLower flag is true we lower the “shipment” meaning of the word to 0.2
- If both or none of them are true we do nothing

High confidence		
Word	Regular Verb	Regular Plural
bartolini, destinatari, dhl, plichi, plico, poste, sda, stoccaggi, stoccaggio, tnt	consegnare, recapitare, trasportare	consegna, corriere, destinatario, prodotto, scatola, spedizione, spedizioniere, trasportatore, trasporto
Medium confidence		
Word	Regular Verb	Regular Plural
corrispondenza, distribuzione, domicili, inoltrare, pacchi, posta, recapito	confezionare, giacere, imbustare, impacchettare, indirizzare, inscatolare, inviare, smistare, spedire, stoccare	busta, collo, confezione, domicilio, giacente, giacenza, incustodito, indirizzo, inoltro, invio, lettera, mittente, pacchetto, pacco, smistamento
Low confidence		
Word	Regular Verb	Regular Plural
distribuire	abbandonare, aprire, ordinare, portare, prelevare, prendere	abbandonato, abitazione, aperto, casa, custodia, deposito, dimora, fornitura, magazzino, mercanzia, merce, ordine

11.1.7 Contract

Words used in and to describe contracts, especially those taken with phone companies.

High confidence			
Word	Regular Verb	Regular Plural	N-Gram
bundle, bundles, business, flat, legalmente, legge, mnp, modulistica, modulistiche, pec, portabilita, portabilità, portability, pratica, pratiche, rinunce, roaming, traffico	addebitare, attivare, cessare, chiudere, disdire, migrare, recedere, rientrare, rimborsare, rinnovare, rinunciare, risarcire, sottoscrivere, stipulare, subentrare	abbonamento, abbonato, addebito, amministrativo, assistenza, attivazione, aziendale, canone, cessazione, chiusura, cliente, consenso, contratto, disdetta, documentazione, documento, legale, migrazione, modulo, norma, onere, penale, procedura, profilo, recesso, residenziale, ricaricabile, rientro, rimborso, rinnovato, rinnovo, rinuncia, risarcimento, sollecito, sottoscritto, sottoscrizione, stipula, subentro, tariffa, tariffazione, utenza	decreto bersani, partita iva, piano tariffario, ragione sociale
Medium Confidence			
Word	Regular Verb	Regular Plural	
pacchetto, tempo	abilitare, aderire, disabilitare	adesione, anticipo, commercializzazione, consumo, offerta, opzione, promozione, scatto	

11.1.8 Communication

Words that may refer to a form of communication of some kind are included in this feature.

Word	Regular Verb	Regular Plural
annunci, annuncio, notifica, notifiche	annunciare, annunciare, ascoltare, avvertire, avvisare, chiamare, comunicare, confessare, contattare, controbattere, dichiarare, diffondere, discorrere, divulgare, esclamare, esprimere, esternare, informare, inoltrare, notificare, parlare, pronunciare, recapitare, recitare, replicare, ribattere, riferire, rispondere, rivelare, scrivere, segnalare, smentire, sollecitare, sostenere, spiegare, telefonare, trasmettere	avvertimento, avviso, chiamata, comunicazione, confessione, contatto, dichiarazione, diffusione, discorso, divulgazione, esclamazione, espressione, indirizzo, informazione, inoltro, recapito, replica, repliche, riscontro, risposta, rivelazione, segnalazione, smentita, sollecito, spiegazione, telefonata, trasmissione

11.1.9 Sentiment

This feature considers the words most frequently used in expressing sentiments (most of the time negative) or to describe problems in a strongly emotional way. Again the value assigned depends on the level of confidence of the words and on the clusters of words found: 1.0 for high confidence words, 0.8 for medium confidence in clusters and 0.4 for medium confidence alone.

High confidence		
Word	Regular Verb	Regular Plural
assurdità, bene, benissimo, bestialità, cacca, cacche, calvari, calvario, ciniche, cinismo, fanculo, fastidi, fastidio, illudere, lerciume, letame, luridume, male, malissimo, pattume, piaga, piaghe, popò, porcheria, pupu, pupù, schifo, serietà, serietà, sterco, strazi, strazio, sudiciume, supplizi, supplizio, travagli, travaglio	abbindolare, accalappiare, affliggere, bidonare, buggerare, cagare, carpire, chiavare, ciulare, commiserare, compatire, defecare, deludere, derubare, disperare, fottere, fottersene, fregare, imbrogliare, inculare, infastidire, infinocchiare, ingannare, penare, propinare, raggirare, rapinare, schifare, scopare, scroccare, sgraffignare, straziare, tormentare, torturare, tradire, tribolare, trombare, truffare, turlupinare	affanno, afflizione, amarezza, angoscia, angosciante, ansia, asinata, assillo, assurdo, baggianata, balla, bidone, boiata, bravo, cacata, cagata, cappella, casino, castroneria, cavolata, cazzata, cazzo, cinico, coglionata, coglione, commiserazione, compassione, cretineria, culo, disgrazia, disperazione, dispiacere, eresia, escremento, fesseria, fesso, fottuto, fregatura, furto, grana, idiozia, immondezza, immondizia, ladro, merda, monata, mortificazione, ossessione, panzana, patema, pena, peripezia, pessimo, rognà, scemata, scemenza, scempiaggine, schifezza, sciagura, scoglionamento, spazzatura, sproposito, stronzo, stupidaggine, stupido, sventura, tormento, tortura, tribolazione, tristezza, troiata, truffa, vendetta
Medium confidence		
Word	Regular Verb	Regular Plural
palle	frantumare, girare, rompere, scassare	giramento, rottura, scassamento

11.2 Technical

11.2.1 Devices

This feature regard the words that refers to a physical device of some kind. This category is of course one that needs to be updated frequently to add new brands and new models.

High confidence		
Word	Regular Plural	N-Grams
android, apparecchi, apparecchio, galaxy, huawei, ipad, iphone, iphone4, iphone5, lg, lumia, microsím, mobile, modem, modems, nokia, smartphone, tablet	apparato, apparecchiatura, brandizzato, centralino, dispositivo, impianto, telefonino, telefono	smart phone

11.2.2 Malfunctions

Here we look at words that may describe a malfunction of some kind (device, line or other). Together with a list of words that can be used to describe a malfunction of some kind we try to detect two words forms like:

- The word “*non*” followed by a verb in { *chiamare, ricevere, funzionare* }
- The verb “*cadere*” followed by a word in { *linea, connessione, collegamento, rete* }

High confidence			
Word	Regular Verb	Regular Plural	N-Gram
ticket, tickets	danneggiare, riparare, rompere, sistem, spaccare, spezzare	danneggiamento, danneggiato, danno, difettoso, disservizio, fracassato, guasto, infranto, magagna, malfatto, malfunzionamento, manomesso, riparazione, rotto, rottura, spaccato, spezzato	mettere a posto

11.2.3 Web

In this feature we try to catch everything that may relate to computers and web. Besides the words in the dictionaries we also look for units of measure of memory and speed, and for number with a unit attached.

High confidence			
Word	Regular Verb	Regular Plural	N-Gram
adsl, apn, app, appstore, browser, browsers, chat, chrome, click, computer, computers, download, downloads, ebill, email, emails, explorer, facebook, fibra, firefox, form, forms, google, ibox, internet, ipad, linux, log, login, lte, mac, mbps, microsoft, modem, password, pc, ping, psw, pwd, reboot, router, routers, server, servers, tablet, tablets, twitter, unix, upload, uploads, usb, user, username, usr, wap, web, webcube	autenticare, cliccare, navigare	autenticazione, credenziale, navigazione, portale, schermata, sito	app store, area clienti, calcolatore elettronico, internet key, jelly bean
Medium Confidence			
Word	Regular Verb	Regular Plural	
applicazione, chiavetta, crash, dati, servizi, servizio	accedere, archiviare, cancellare, caricare, comparire, connettere, consultare, copiare, immettere, inserire, introdurre, memorizzare, registrare, scaricare, visualizzare	accesso, aggiornamento, connessione, consultazione, immissione, inserimento, introduzione, memoria, pagina, registrazione, riavvio, scaricamento, schermo, versione, visualizzazione	

11.3 Telephone Companies Specific

11.3.1 People

The purpose of this feature is to catch sentences or parts of them where something is said about operators or other workers of a company (bad behavior, or less frequently some compliments).

High confidence words are given a value of 1.0 whilst medium confidence ones are initially given a 0.4. Then a second pass rises the medium confidence value to 0.8 if more words are found in the same cluster (in 5 consecutive words at least 1 high or 2 medium).

High confidence		
Word	Regular Plural	N-Grams
119, 133, 187, 191, operai, promoter, promoters, venditore, venditrice	agente, centralinista, commesso, operaio, operatore, operatrice, rappresentante, referente, rivenditore, rivenditrice, signorina, tecnico	customer care, responsabile commerciale, servizio clienti
Medium confidence		
	Regular Plural	
	addetto, dipendente, impiegato, incaricato	

11.3.2 Phone Line

Basically we consider here all the words that may be related to the concept of “phone line”. Also, we give that tags to words that may be a phone number, requiring a length of at least 5 digit but also considering that sometimes phone numbers are split for readability in contiguous groups of 3 or 4 digit each.

High confidence			
Word	Regular Verb	Regular Plural	N-Gram
adsl, bandalarga, download, edge, fax, fibra, lte, ping, portante, sim, telefonia, telefoniche, umts, upload, voip		cellulare, connessione, copertura, telefonico, telefono	banda larga
Medium Confidence			
Word	Regular Verb	Regular Plural	
apparecchi, apparecchio, card, cards, filo, fruscii, fruscio, instabilita, instabilità, latenza, link, montaggi, montaggio, numerazione, video, voce, wired	connettere, crollare, ricevere	allacciamento, apparato, apparecchiatura, attenuazione, cabina, cavetto, cavo, centrale, collegamento, congegno, coperto, cordone, dati, dispositivo, fisso, impianto, instabile, installazione, lento, linea, numero, raggiungibile, rete, ricezione, rumore, segnalatore, segnale, spia, tratto	

11.3.3 Line Quality

Here we try to catch the parts of the sentences where the quality of the line is talked about. Using also two features that must be already calculated (“PhoneLine” and “Geo”) we proceed as follows:

- If a word in the sentence is found in the high confidence group we tag that word with the highest value (1.0)
- If that word has already been recognized as a “PhoneLine” word we check the words around it (with a radius of ± 5 words) and if they are in the medium confidence set or if they were already been recognized as “Geo” words we tag both with a reasonable high value (0.9)

The reason for using the “Geo” feature is that very often discussions about the quality of the line, and especially about the coverage, contains indications about a specific place where such quality may be good or poor.

High confidence		
Word	Regular Verb	Regular Plural
connettersi, offline, online	connettere	connessione, saturo, veloce
Medium confidence		
Word	Regular Verb	Regular Plural
bene, benissimo, decenza, efficienza, indecenza, inefficienza, insufficienza, irregolarita, lentezza, male, scarsita, senza, velocita	agganciare, arrivare, collegare, funzionare, prendere	accelerato, attiva, decente, efficiente, fulmineo, funzionamento, funzionante, inchiodato, insufficiente, irregolare, lentissimo, lento, malissimo, rapido, scadente, scarso, spedito, ultraveloce, velocissimo

11.3.4 Companies

We use this feature to tag words that may refer to phone companies, including both their names and the names of their offers.

Algorithm:

1. First we look inside the sentence to find company names
2. Next we look for offers names alone. We characterize every offer name with three parameters ($minNec1$, $minNec2$, $minOpt$) and with two set of words: necessary and optional. In order to recognize the offer we require that in a set of (almost) contiguous words we must be able to find:
 - a. At least $minNec1$ necessary words OR
 - b. At least $minNec2$ necessary words together with $minOpt$ optional ones
3. If the offer name can't be recognized in step 2 but we found at least one of the necessary words we check its surroundings for the company name eventually already found in step 1, and if present we tag the words of the offer too.

So for example, with the offers in the following table, the string “*smart*” alone will not be identified as a name of a Vodafone offer, but both the sets { “*smart*”, “250” } and { “*vodafone*”, “*smart*” } will be.

Here for “set of almost contiguous words” we mean a set where the number of unrelated words is not higher than a specified threshold (at the moment 2 unrelated words at maximum).

Company Name (single word)	Company Name (Bi-gram)	Generic Word
telecom, telecomitalia, impresasemplice, tim, vodafone, voda, vodaphone, wind, infostrada, fastweb, fw, h3g, telepiu, tiscali, tiscalimobile, bt	telecom italia impresa { semplice facile } { 3 tre } italia fast web tele piu british telecom	aom , operatore, operatori, provider, providers

Offers					
Company	Necessary Words	Optional Words	minNec1	minNec2	minOpt
Telecom Italia	official				
Telecom Italia	young				
Telecom Italia	senza limiti				
Vodafone	square	smart	-1	1	1
Vodafone	unlimited				
Vodafone	you				
Vodafone	all inclusive				
Vodafone	smart	250	-1	1	1
Tre	super internet				
Tre	top	400	-1	1	1
Fastweb	homepack	full	-1	2	1

11.3.5 Variation

Here we want to highlight any variation that may regard a contract or a phone line, especially if they involve a change of company or a main change in the kind of contract (from business to consumer and vice versa). Some words (*traslocare, subentrare, ...*) are alone clear expressions of this kind of variation. For others (*cambiare, passare, variare, ...*) a better look around the word itself should be done in order to try to identify the object of the change. Here we don't try any grammatical analysis of any kind, we simply search around the word for other words in a list of possible objects.

Algorithm:

1. If the word is one of those that do not require a object we tag it with a very high value(1.0)
2. If an object is required:
 - a. we first look around for the name of a company, using the feature "Company" that must be already calculated. If found we also tag the word with and high value (1.0)
 - b. Otherwise we look around for one of the possible objects. If found both words are tagged with a quite high value (0.8), otherwise the word is given a "low confidence" value (0.4).
3. If the word itself does non describe a variation but it is in the set of words that may describe a business/consumer situation, we look around for another word in the same set, and if found we tag both with a quite high value (0.8)

Words that require no subject		
Word	Regular Verb	Regular Plural
mnp, traslochi, trasloco	subentrare, traslocare	disdetta, subentro
Words that require an object		
Word	Regular Verb	Regular Plural
cambi, cambio, passaggi, portabilità, portabilità, sostituzione	attivare, cambiare, commutare, convertire, disdire, mantenere, migrare, mollare, mutare, passare, permutare, recedere, reintegrare, rimpiazzare, sostituire, tramutare, trasferire, trasformare, trasmutare, variare	attivazione, commutazione, conversione, migrazione, mutamento, passaggio, permuta, permutazione, recesso, reintegro, rimpiazzo, tramutazione, trasferimento, trasmutazione, variazione
Possible objects		
Word	Regular Verb	Regular Plural
domicili, domicilio, sim		abitazione, casa, civico, denominazione, dimora, indirizzo, linea, nome, numerazione, numero, recapito, residenza, scheda, sede
Business / Consumer		
Word	Regular Verb	Regular Plural
business, consumer		affari, commerciale, domestico, privato, residenziale

11.3.6 Ceasing

This feature is about the action of ceasing a line or an entire contract. Words in the high confidence set are tagged with the maximum value of 1.0. Words in the medium confidence set are also tagged with a 1.0 if they are found near a word in the “possible objects” list, with 0.7 otherwise.

High confidence			
Word	Regular Verb	Regular Plural	N-Grams
revoca, revoche	cessare, disdire, revocare	cessazione, disdetta, sfratto	porre fine
Medium confidence, need object			
Word	Regular Verb	Regular Plural	
annullamento, rinunce, ritirarsi, sciogliere	annullare, chiudere, concludere, dismettere, eliminare, estinguere, interrompere, sbaraccare, smantellare, sospendere	chiusura, conclusione, dismessi, dismissione, eliminazione, estinzione, rinuncia, scioglimento, smantellamento, sospensione	
Possible Objects			
Word	Regular Verb	Regular Plural	
attività, attività		contratto, linea, numero	

11.3.7 Return

This feature instead is about the action of coming back to a previous telephone company. If a word in the set is found near a word previously tagged as “Company” that word is tagged with a value of 1.0.

Word	Regular Verb	Regular Plural
ritorno	rientrare, ritornare	gestore, rientrati, rientro

11.4 Other

11.4.1 Advertisement

Many of the Twitter messages happen to be comments on the last TV advertisement of a particular company. Although such tweets may not be so interesting, detecting them may help to refine the result.

Word	Regular Plural
elio, musica, musiche, pubblicità, publicitá, pubblicità, pubblicitari, reclame, spot, spots	canzone, canzonetta, pinguino, pubblicitaria

11.4.2 Unrelated

As well as for of the Advertisement feature, detecting frequent unrelated or uninteresting text may help in getting good results with the others.

Word	N-Gram
compleanno	beppe grillo

12 Appendix 3: Complex Tags with examples

Business
Offers
A description or a request of information about the characteristic of a specific offer, or a comment or comparison between two of them
<i>Ho pensato che con Vodafone Unlimited avrei avuto Internet gratis, ma dovrei pagare le chiamate, quindi penso che...</i>
<i>Gente, pareri su Vodafone Unlimited? Elio misà che mi ha convinto.</i>
<i>Vodafone unlimited.... Prrr meglio la tim young xl, ci sono anche i minuti! #sapevatelo</i>
<i>Vodafone Unlimited: se ho 1 GB di internet con cui sfruttare whatsapp, a che mi servono gli sms?</i>
Contract
Something about a contract that has been already made. Most of the time a complaint about money.
<i>ho attivato vodafone unlimited e connettendomi a internet tramite 3g dopo l'attivazione mi sono stati scalati lo stesso 4e..come mai?</i>
<i>Salve, l'ordine effettuato correttamente il 21\02 sembrerebbe bloccato sui sistemi. Richiedo immediato sblocco. Grazie.</i>
<i>come faccio a sapere quanto è la mia penale per il recesso anticipato?</i>
<i>Se si attiva #VodafoneUnlimited il 190 diventa a pagamento (1 Euro a chiamata).</i>
Activation, Cessation, Portability
Everything related to the activation or cessation of a line or porting of a number from one company to another.
<i>mi sa che comunque è ora di cambiare provider. #fastweb ha calato la reale banda disponibile un'altra volta. unpf!</i>
<i>Codice di Migrazione non Conforme..i soliti problemi x passar da @VodafoneIT a @FASTWEB con #HomePack!</i>
<i>buongiorno ho sottoscritto ieri un nuovo contratto mobile ma il numero non risulta attivo. cosa devo fare per verifca attivazione?</i>
<i>voglio sapere perché continuate a bocciarmi la migrazione a @FASTWEB per motivi fittizzi senno vi denuncio.</i>
Human Factor
Difficult Communication
<i>@Tre_It finalmente sono riuscito a riattivare il rinnovo automatico di Super Internet New: ho dovuto chiamare il 4040; dal sito non funge.</i>
<i>Velocita adsl ridicol meno di 1 mbit , chiedo assistenza qui visto che al 130 non rispondono</i>
<i>volete chiamarmi? Non riesco a mettermi in contatto con nessuno di voi! Mi sembra la #sagradegliorrori</i>
<i>30 minuti per parlare con un operatore senza risolvere nulla!!! NON ARRIVA IL SEGNALE!!! SVEGLIA!!!</i>
Late or No Response
<i>@TiscaliHelpDesk due giorni senza ADSL, ho aperto 2 ticket e dopo 40 min di attesa è caduta la linea con l'OP.</i>
<i>@FASTWEB volentieri, ma io sto ancora aspettando che mi ricontattino dopo aver mandato il contratto firmato. Da due settimane.</i>
<i>posso avere notizie? Vi è troppo disturbo rispondermi?</i>
Bad Behavior
<i>@FASTWEB visto che il mio cod. cl. non é riconosciuto e i vostri gentili operatori mi chiudono il telefono in faccia..Come si fa a parlarvi?</i>
<i>ma la signorina della vodafone, che disturba il mio sonno di domenica mattina per propinarmi un pacchetto ADSL, che problema ha?No, ditemi!</i>
<i>E' la 4a volta in 4 giorni che #telecomitalia mi chiama sul cellulare per la stessa offerta. @telecomitalia cancellatemi dalle liste subito!</i>

Positive comment / Thanks
<i>@telecomitalia finalmente dopo anni il problema è stato risolto! Grazie grazie! Finalmente pago 7mb e mi collego a 7mb!! :)</i>
<i>Orgoglioso di far parte della famiglia #tre @Tre_It #Auguri3 #10AnniDi3</i>
<i>@FASTWEB grande ragazzi. Problema risolto! :)</i>
Operator Reply
<i>Ciao Pietro, come possiamo aiutarti?</i>
<i>Salve, per supporto ci invii in DM i suoi dati e un recapito per il contatto.Grazie.</i>
<i>Ho sollecitato un intervento dei colleghi competenti. le auguro una buona serata.</i>
Shipment - Lost Packages
<i>Salve, non abbiamo una data certa, ma l' apparato è già in consegna e stiamo attendendo ultimo OK da Telecom entro il 05\03.</i>
Other
<i>possibile che nessuno mi sa risolvere il mio problema su vodafone calcio da 15gg? Cliente relax e mi rispondono stranieri</i>
<i>@VodafoneIT oggi sono stato contattato per fare adsl a casa, ma ce l'ho già dall'inizio! A molti fanno offerte con sconti a me no! Attendo!!</i>
Internet & Devices
Device characteristics
<i>Aggiornamento SW, arriva LTE sui Nokia Lumia 820 e 920 a marchio 3 Italia</i>
<i>Qual è l'ultima versione di Android disponibile per SII brandizzato 3?</i>
<i>ho visto uno Z10 da Vodafone (finalmente!), ma il tipo dice che è brand (a me non sembra). Come faccio a riconoscerlo?</i>
Device malfunctions
<i>@TiscaliHelpDesk NON funziona ancora la linea!! LED Broadband lampeggia sul modem,provato anche senza filtro ADSL.</i>
<i>Porca miseria, ho cambiato il modem, quello di Fastweb il bianco e mo mi da problemi</i>
<i>@Tre_It nuova SIM che non si attiva. Configurato cell da menù ass senza esito. help!</i>
Official Operator App / Web Site
<i>Ciao, UNLIMITED prevede un'assistenza dedicata:utilizza la Chat sul mini sito oppure l'App gratuita My 190</i>
<i>Ciao, Open 200 può essere disattivata chiamando il 40211 o nell'Area FaidaTe. Qui trovi gli APN inclusi</i>
<i>@Tre_It non mi risulta la lista completa delle ultime fatture. Problemi tecnici?</i>
Other
<i>Huawei presenta FusionNet per LTE-B per il futuro della banda larga</i>
<i>Facebook per iPhone: nuova release con chiamate Voip: L'aggiornamento tanto atteso dai Facebook-dipendenti</i>
<i>salve ma LTE SAREBBE IL 4G?</i>

Quality of service
Coverage
<i>@telecomitalia Possibile che non si riesca ad avere l'ADSL perché non c'è posto sulla centralina?? Una lista attesa no, eh??</i>
<i>Informazione di servizio: qualcuno sa se a Benevento, zona centro storico, c'è copertura Fastweb?</i>
<i>Paesino con ADSL scadente si stende la fibra da sé - Zeus News</i>
Line Quality
<i>Ho un piano #FastWeb da 20 MB, 20 MB. Ho un download EFFETTIVO di 500KB da SEMPRE, UN QUARANTESIMO di ciò che PAGO. COMPLIMENTI.</i>
<i>qualcuno ha #Vodafone #unlimited? come va la velocità di internet?</i>
<i>@Tre_It io passerei a 3 sul cellulare. Ma la pessima esperienza che sto avendo con la chiavetta internet mi inibisce.</i>
<i>LTE è una scheggia. Aspettiamo la copertura completa</i>
<i>Sono in un ristorante. Io (TIM) prendo in EDGE. Il mio vicino (Vodafone) prende in LTE. \Oggi volete farmi arrabbiare, eh?</i>
Malfunction / Set Up / Delay
<i>La Fastweb sarà una delle migliori aziende nel campo delle connessioni, ma prima di mandare il tecnico, passano settimane.</i>
<i>Lasciare una società senza adsl e fax dall'82...questa è Telecomitalia!</i>
<i>@TiscaliHelpDesk due giorni senza ADSL, ho aperto 2 ticket e dopo 40 min di attesa è caduta la linea con l'OP.</i>
Other
Advertisements
<i>Andare più veloce dell' #Adsl, in auto! @GM annuncia la #banda ultralarga sui nuovi modelli. Ecco l' \auto connessa\</i>
Comment on Advertisements
<i>La Vodafone con le nuove pubblicità sta invogliando a passare alla concorrenza...</i>

13Bibliography

13.1 Books

- [Liu-08] Bing Liu “Web Data Mining. Exploring Hyperlinks, Contents and Usage Data”, Springer 2008
- [Tan-06] Pang-Ning Tan, Michael Steinbach, Vipin Kumar “Introduction to Data Mining” Pearson 2006
- [Berry-97] Michael J.A. Berry, Gordon Linoff “Data Mining Techniques for Marketing, Sales, and Customer Support”, Wiley 1997
- [Groot-12] Roy De Groot, “Data Mining for Tweet Sentiment Classification, Twitter Sentiment Analysis”, Lambert Academic Publishing 2012
- [Hertz-91] John Hertz, Andres Krough, Richard G.Palmer, “Introduction to the theory of neural computation”, Perseus Book 1991
- [Russel-11] Matthew A. Russel “Mining the Social Web. Analyzing Data from Facebook, Twitter, LinkedIn and Other Social Media Sites”, O’Reilly, 2011
- [Heaton-11] Jeff Heaton, “Programming neural networks with Encog 3 in C#”, Heaton Research, 2011
- [Tsvetovat-11] Maksim Tsvetovat, Alexander Kouznetsov, “Social Network Analysis for Startups”, O’Reilly 2011
- [Jurafsky-08] Daniel Jurafsky, James H. Martin “Speech and Language Processing”, 2008

13.2 Articles

- [Tsoumakas-07] G.Tsoumakas, I. Katakis: Multi-Label Classification: An Overview. International Journal of Data Warehousing & Mining, 3(3), 1-13, July-September 2007
- [Vens-08] C.Vens, J.Struyf, L.Schietgat, S.Dzeroski, H.Blockeel: Decision Trees for Hierarchical Multi-label Classification. Machine Learning, 2008 – Springer
- [Hearst-94] M.Hearst - Multi-paragraph segmentation of expository text. ACL '94 Proceedings of the 32nd annual meeting on Association for Computational Linguistics.
- [Read-11] J.Read, B.Pfahring, G.Holmes, E.Frank: Classifier chains for multi-label classification. - Machine Learning, 2011 - Springer
- [Zhang-05] ML Zhang, ZH Zhou: A k-nearest neighbor based algorithm for multi-label classification. IEEE International Conference on Granular Computing, 2005 (Volume:2)
- [Tsoumakas-08] KTG Tsoumakas, G Kalliris: Multi-label classification of music into emotions. ISMIR 2008: Proceedings of the 9th International Conference of Music
- [Beeferman-99] D. Beeferman, A.Berger, J.Lafferty: Statistical Models for Text Segmentation. Machine Learning 1999 Kluwer Academic Publishers.