Milestone report

Me

Thursday, March 19, 2015

# Main text

## Introduction

This paper presents intermediate results in the construction of a predictive text model. The final objective of this project is to construct a Shiny app which provides users with suggestions for auto completing a sentence they have started.

The training data set has been obtained from the Coursera website, but are originally from a corpus called HC Corpora (www.corpora.heliohost.org). The complete set consists of databases in English, German, Russian and Finnish. In this project we will limit ourselves to the English database. The English database itself is composed of three files, each composed of data sets containing several hundreds of thousands of text lines from blogs, news sites and Twitter.

As a first step, we will summarize the main descriptive statistics for each dataset, and explore how these compare to each other - these statistics will be provided for individual words and for 2-grams (for a definition of this term, see further). As a next step, we will explain the approach used to the cleaning and the organisation of the data. Finally, we will describe the next steps we intend to undertake in order to construct our predictive model.

## Descriptive statistics for the unique words

Before undertaking any concrete modelling steps, it is important to understand the main features of the data. As we shall see below, our understanding of the data will allow us to drastically save on computer resources without jeopardizing predictive accuracy.

On prior grounds, one would expect the three datasets to differ on some important points. For instance, Twitter can be expected to contain relatively more abbreviations, relatively few stopwords, and more non-alphanumerical symbols than the other datasets. We would also expect Twitter to contain less grammatically correct sentences. The differences between the "blogs" and the "news" datasets are probably less pronounced, but one would expect the "news" dataset to use a more formal vocabulary, and to contain less spelling and grammatical mistakes. Therefore, we will analyse the key statistics separately.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Lines of text (10^6) | Word count (10^6) | Unique words (10^3) | Perc90 |
| blogs | 0.9 | 37.9 | 253.0 | 0.61 |
| twitter | 2.4 | 30.6 | 302.7 | 0.92 |
| news | 0.8 | 34.6 | 212.2 | 0.55 |

Key summary statistics for unique words

The table above gives some key information on the "blogs", "Twitter"" and "news" datasets:

* the first column gives, for the "uncleaned" version of the databases, the number of lines of text in millions;
* the second column gives, for the databases restricted to alphanumeric characters, the total number of words in millions;
* the third column gives, for the databases restricted to alphanumeric characters, the number of *unique* words in thousands;
* the fourth column gives (in %), the number of *unique* words corresponding to 90% of *all* word occurrences.

We notice that while the Twitter dataset contains approximately three times more lines of text than the other datasets, it contains less words, which is due to the restrictions imposed on the length of individual tweets. On the other hand, the Twitter dataset contains more *unique* words, reflecting probably idiosyncratic expressions and abbreviations used by the "tweeters".

We also observe that, in all three datasets, the vast majority of word occurrences originates from a very small (data set specific) dictionary. The general skewness of the frequency of words is confirmed when we look at the summary statistics of the counts of *unique* words (see Appendix): these statistics confirm that the vast majority of unique words appears just a few times in the total database, but that there are some words that occur with a very high frequency. This is also illustrated by the plot in Appendix, which shows how the share of individual words in the set of *unique* words changes when we require that they appear at least, respectively, 1, 10, 20, 30, 40 or 50 times in the respective complete dataset. We see a sharp initial decrease in this share. For instance, in all three datasets, the share of words occurring at least ten times is between (approximately) 15 and 25% of all unique words.

Still another way to look at the data is to ask: how many *unique* words appear exactly 1, 2, 3 etc times? In Appendix, we represent the first five and the last five lines of the table containing this information (henceforth the "frequency count tables").

These tables show that, for instance, 170 thousand individual words appear exactly 1 time in the Twitter database, while 1 word appears exactly 937970 times in the Twitter database.

A table in Appendix also gives the 6 most frequent words for each of the three datasets. None of the results are surprising, though it should be noted that the appearance of "you" and "i" in the Twitter top 6 and of "i" in the blogs top 6 are an indication of the more "personal" contents of these datasets.

Finally, we can have a look at a random sample of words that appear very infrequently in the database (here, less than 5 times) - see the table in Appendix for a sample of ten words. These words are generally meaningless in English, or are obvious spelling mistakes.

It has been suggested in the literature that the counts of individuals words in a corpus generally follows a power law, i.e. that the frequency of a word is inversely proportional to its frequency rank. Thus, the second most frequent word occurs half as often the most frequent word, the third most frequent word occurs one third as often as the most frequent word, etc - see <http://en.wikipedia.org/wiki/Power_law>. However, a formal hypothesis has rejected this possibility for the databases we use here (details are available on request). This is confirmed by a visual analysis of the plots of the "frequency count table" - see the Appendix for details.

Nevertheless, from the overall analysis in this section, we can be confident that leaving out words that do not occur with a minimal frequency in each dataset is unlikely to affect the predictive performance of the model. We will discuss below how we have determined this "minimal" frequency.

## Descriptive statistics for 2-grams

The next question we need to consider is the distribution of n-grams, i.e. contiguous sequences of n items in the database. As we shall discuss below, they will be the cornerstone of the predictive model. For the purposes of this project, we have constructed 2-grams, 3-grams and 4-grams. As a full discussion of the properties of all n-grams would take us beyond the scope of this paper, we focus here on the analysis of 2-grams. A full analysis is available on request.

As in the analysis of individual words, we have performed a separate analysis for the "blogs", "Twitter"" and "news" datasets.

|  |  |  |  |
| --- | --- | --- | --- |
|  | 2-gram count (10^6) | Unique 2-gram (10^3) | Perc90 |
| blogs | 37.5 | 7401.8 | 18.18 |
| twitter | 29.6 | 7304.6 | 22.97 |
| news | 34.1 | 7214.0 | 19.64 |

Key summary statistics for 2-grams

The table above gives some key information on "blogs", "Twitter"" and "news" databases:

* the first column gives, for the databases restricted to alphanumeric characters, the total number of 2-grams
* the second column gives, for the databases restricted to alphanumeric characters, the number of *unique* 2-grams
* the third column gives (in %), the number of *unique* 2-grams corresponding to 90% of *all* 2-gram occurrences.

We see thus that the vast majority of 2-gram occurrences originates from a small set of *unique* 2-grams. However, as we should have expected, the share of these *unique* 2-grams in the total dataset is much larger (a factor twenty) than for individual words.

The general skewness of the frequency of 2-grams is confirmed when we look at the summary statistics of the number of counts (see Appendix). These statistics confirm that the vast majority of unique 2-grams appears just a few times in the total database, but that there are some 2-grams that occur with a very high frequency. This is also illustrated by the plot in the Appendix, which shows how the share of 2-grams in the set of *unique* 2-grams changes when we require that they appear at least, respectively, 1, 10, 20, 30, 40 or 50 times in the respective documents. We see that the decrease is even faster than in the case of individuals words.

Ad we did above for individual words, we have also constructed the "frequency count table" for 2-grams - see Appendix for the "head" and the "tail" of this table. These tables show that, for instance, 5519 thousand individual 2-grams appear exactly 1 time in the Twitter database, while 1 individual 2-gram appears exactly 73971 times in the Twitter database.

A table in Appendix gives the most frequent 2-grams for each dataset. As in the case of individual words there are no surprises. It is interesting to note that they all start with a preposition followed by an article.

Finally, we can have a look at a sample of 2-grams that appear very infrequently in each database (see Appendix). A lot of these 2-grams are meaningless in English, but compared to individual words, there are also some "low frequency" 2-grams that do make sense.

In the plot in the Appendix, we have set out the logs of the variables in the "frequency count table". This confirms that, for 2-grams that appear very frequently, the observations deviate substantially from the predictions of a power law, and that we cannot thus maintain the hypothesis of a power law - a rigorous hypothesis test falls outside the scope of this paper.

## Data cleaning

As the next step in our project, we have cleaned the original data:

* First, we have modified the encoding from "UTF-8" to "latin1" - not doing so would lead to the creation of unreadable characters.
* Second, we have removed any numbers present: these are highly context-dependent and unlikely to have a high predictive value (or, conversely, unlikely to be easy to predict).
* Third, we have removed all the words that are banned by Google for their profane or offensive character.
* Fourth, we have removed all white spaces that have resulted from the previous steps.

However, punctuation and other non-alphanumeric characters are an integral part of the sentence (especially in the case of Twitter), and have therefore been kept as they were. Meaningless non-alphanumeric characters are unlikely to appear frequently in the same sequences, and will thus be eliminated by restricting the sets to words and n-grams that appear with a minimal frequency.

Although foreign words have crept inside the database, we have left them as they were. Indeed, some foreign words are deliberately used in English to convey a specific message. If foreign sentences have been included inadvertently in the database, it is unlikely that similar sentences (or sentences "contaminating" English content) have appeared frequently, and they are also likely to disappear once we eliminate low frequency words and n-grams.

Next, we have used the "cleaned" version of the database (which differs thus from the "raw" version discussed above) to create a list of words and their occurrence. Moreover, we have constructed a list of all n-grams (for n = 4,3,2) and counted how often each individual n-gram occurs. Thus, for instance, the sentence: "This is how to count two-grams" would be split in: "this is", "is how", "how to", "to count", "count two-grams", and then we would count how often each 2-gram "occurs in the dataset.

For the purposes of the predictive model, we have used the Markov assumption, this is, we have assumed that the probability that a given word is the next word in a sentence only depends on the *n* preceding words, rather than on the complete sentence.

For instance, we have used the 4-grams to predict the probability of a word occurring, conditional on the three preceding words in the sentence. In order to do so, we have split each 4-gram in two strings, one representing the first three words and one the final word (which needs to be predicted). Because we know the count for all 4-grams and all 3-grams, it is straightforward to calculate the conditional probability that a given word will follow a given 3-gram.

As stated above, because of the general skewness of the distribution of the n-grams and in order to save computer resources, we have limited ourselves to the n-grams that appeared at least 3 times in each dataset. It remains to be tested how this choice affects the overall predictive power of our model.

For any given sentence, the prediction algorithm will then first predict the three most likely "next" words, first using the last three words of the sentence, then (if no results are obtained from the previous step) the last two words, etc.

## Next steps

With the work performed until now, two major steps remain to be done. First, test the actual performance of the model with alternative datasets. Second, to develop the "Shiny" app. It is possible that, due to the technical limitations of Shiny, our model will need to be "culled" in order to be implementable on-line.

# Appendix

## Summary statistics for words and 2-gram counts

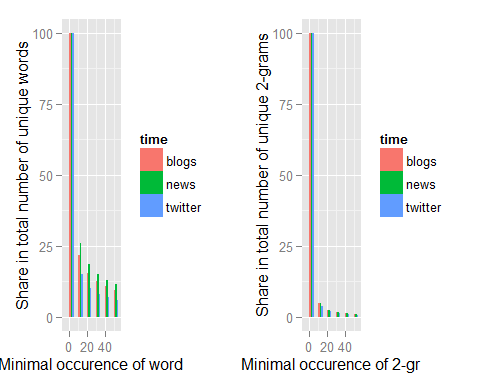
|  |  |  |  |
| --- | --- | --- | --- |
|  | news | twitter | blogs |
| Min. | 1.0 | 1 | 1.0 |
| 1st Qu. | 1.0 | 1 | 1.0 |
| Median | 2.0 | 1 | 2.0 |
| Mean | 163.1 | 101 | 149.7 |
| 3rd Qu. | 10.0 | 4 | 7.0 |
| Max. | 1975000.0 | 938000 | 1861000.0 |

Summary statistics for word counts

|  |  |  |  |
| --- | --- | --- | --- |
|  | news | twitter | blogs |
| Min. | 1.000e+00 | 1.000 | 1.000e+00 |
| 1st Qu. | 1.000e+00 | 1.000 | 1.000e+00 |
| Median | 1.000e+00 | 1.000 | 1.000e+00 |
| Mean | 4.726e+00 | 4.057 | 5.069e+00 |
| 3rd Qu. | 2.000e+00 | 1.000 | 2.000e+00 |
| Max. | 1.848e+05 | 73970.000 | 1.843e+05 |

Summary statistics for 2-grams

## Share of unique words/2-grams as function of minimal occurence



## Frequency count tables for low frequency words and 2-grams

|  |  |  |  |
| --- | --- | --- | --- |
| # of unique words in twitter | # of unique words in blogs | # of unique words in news | counts |
| 169816 | 116015 | 81854 | 1 |
| 36519 | 31710 | 27381 | 2 |
| 17002 | 16686 | 14798 | 3 |
| 10559 | 10269 | 9846 | 4 |
| 7264 | 7132 | 7163 | 5 |
| 5427 | 5497 | 5255 | 6 |

Low frequency words

|  |  |  |  |
| --- | --- | --- | --- |
| # of unique 2-grams in twitter | # of unique 2-grams in blogs | # of unique 2-grams in news | Counts in each dataset |
| 5519157 | 5355250 | 5105961 | 1 |
| 756455 | 828897 | 855739 | 2 |
| 297863 | 337474 | 348869 | 3 |
| 161059 | 185305 | 191777 | 4 |
| 101610 | 118067 | 122150 | 5 |
| 70429 | 82088 | 84715 | 6 |

Low frequency 2-grams

## Frequency count tables for high frequency words and 2-grams

|  |  |  |  |
| --- | --- | --- | --- |
| # of unique words | counts in twitter | counts in blogs | counts in news |
| 1 | 438745 | 876848 | 679242 |
| 1 | 601330 | 904594 | 774525 |
| 1 | 617520 | 906917 | 889612 |
| 1 | 788952 | 1069565 | 894899 |
| 1 | 918858 | 1094859 | 906198 |
| 1 | 937970 | 1860686 | 1975163 |

High frequency words

|  |  |  |  |
| --- | --- | --- | --- |
| # of unique 2-grams | counts in twitter | counts in blogs | counts in news |
| 1 | 41475 | 54400 | 53460 |
| 1 | 45247 | 67089 | 65669 |
| 1 | 45792 | 70912 | 69699 |
| 1 | 54730 | 84541 | 82845 |
| 1 | 71627 | 143331 | 167229 |
| 1 | 73971 | 184311 | 184752 |

High frequency 2-grams

## Sample of low frequency words and 2-grams

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| count | news | count | twitter | count | blogs |
| 1 | feminazis | 1 | wheresmypersonalassistant | 1 | tme |
| 3 | securityplus | 1 | kanna | 1 | pranava |
| 1 | impulsiveness | 1 | predilections | 1 | oughton |
| 1 | thehotelito | 1 | mufugga | 1 | zbf |
| 1 | vezetti | 1 | boshra | 1 | pascho |
| 4 | assemblers | 1 | tvless | 1 | puzey |
| 2 | mafias | 1 | easyway | 2 | mazzucchelli |
| 3 | titters | 1 | artistadmin | 1 | naivetes |
| 4 | mccreanor | 1 | getyofactsstraight | 1 | fairus |
| 2 | ketogenic | 1 | whatexam | 3 | centsable |

Random sample of infrequent words

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| count | news | count | twitter | count | blogs |
| 1 | Tour So | 1 | wars exist | 2 | that Oprah |
| 1 | rarity to | 1 | coffee twitter | 1 | of Nowruz |
| 1 | called Glor | 1 | markett Not | 1 | manufactured a |
| 1 | studies Garfield | 2 | gotten spam | 4 | years Next |
| 2 | trial enters | 2 | Fire Chief | 2 | minutes Two |
| 1 | CRUSH LIKE | 1 | theme w | 2 | or UKIP |
| 1 | favors urged | 1 | ReMega Millions | 4 | harmony as |
| 1 | swigging beer | 1 | Brut We | 1 | is jelly |
| 2 | friendly dog | 1 | Whirlwind ride | 1 | am exclusively |
| 1 | cousin's husky | 1 | waistoftaxpayermoney emorydisc | 1 | Massacre narrowly |

Random sample of infrequent 2-grams

## Most frequents words and 2-grams

`

|  |  |  |
| --- | --- | --- |
| news | twitter | blogs |
| a | a | a |
| and | and | and |
| in | i | i |
| of | the | of |
| the | to | the |
| to | you | to |

Most frequent words in the three datasets

|  |  |  |
| --- | --- | --- |
| news | twitter | blogs |
| at the | for the | for the |
| for the | in the | in the |
| in the | of the | of the |
| of the | on the | on the |
| on the | to be | to be |
| to the | to the | to the |

Most frequent 2-grams in the three datasets

## Graphical representation of the frequency count tables

In the plot below, we have set out the logs of the variables in the "frequency count table". Thus, on the horizontal axis, we represent the log of the number of different words/2-grams occurring a given number of times in each dataset (where the vertical represents the log of the count of this word/2-grams in the dataset).

If the counts would follow a power law, the plots would follow a linear trend. However, we observe that this linear trend only holds (approximately) for unique words and 2-grams that occur less than 100 times in the total dataset. After this threshold, we see that there is an increasing spread around the linear trend: for words that occur very often, it is very unlikely that there is another word with exactly the same count in the dataset (and log(Number of words) is thus very likely to be zero).

