Data Science Capstone Project – Milestone 1

Ramaa Nathan

Sep 19, 2014

The main goal of this project is to build predictive models within the realm of natural language processing.

To do, this we are given 3 corpus in three 3 different languages – English, German, and Finnish. At this milestone, we will only consider the English language.
There are three different documents that make up the corpus - texts from Twitter feed, news feed, and a blog.

Data Study and basic statistics:
Each of the documents are long and the line, word and character count for these documents are as follows:

```
  899288 37334131 210160014 en_US.blogs.txt
 1010242 34372530 205811889 en_US.news.txt
 2360148 30373583 167105338 en_US.twitter.txt
```

Preprocessing steps:
Now, each of these documents need to be cleaned up. They could either be loaded directly into the corpus using the tm module of R and then the pre-processing could be done there. But, this results in major memory problems and the machines are crashing.

So, basic unix tools have been used to do the following pre-processing:

1. remove punctuation
2. convert to lower case
3. remove digits or numbers
4. convert URL strings to URL
5. remove non printable characters

Unix command:
```
cat en_US.news.txt | tr -d "[:punct:]"  | tr "[A-Z]" ["a-z"] |  tr -d "[:digit:]" | tr –d "[:print:]" | sed -Ee "s/http.* |http.*$/URL /g" > clean_en_US.news.txt
```

The above commands are run for all the three documents.

As we are building a predictor model, it is important to have training, validation and test sets. Again, doing all these within the Corpus object makes it very memory intensive. So, the following steps have been taken:

1. Read each file into R using readLines
2. Take a random sampling of these lines (used 1% for training set, 0.5% for validation, 0.5% for test set).
3. Write the sampled sentences into separate documents – train_news.txt, val_news.txt, test_news.txt. Similar steps are taken for the twitter and blog documents.

The documents are now ready to be read into the Corpus. This is done using the DirSource, and a readPlain reader. As a precaution, tm_map is used to remove punctuation, strip white spaces.

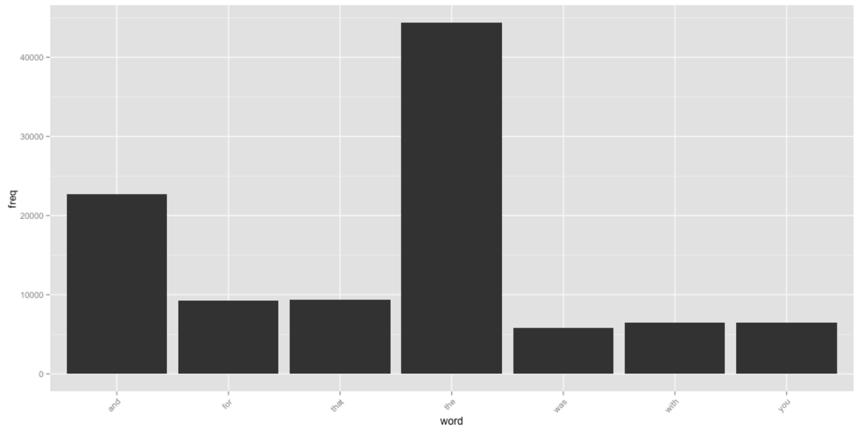
We want to build a n-gram model here. So, we use the RWeka package to build the N-gram tokenizer using unigrams, bigrams, and trigrams

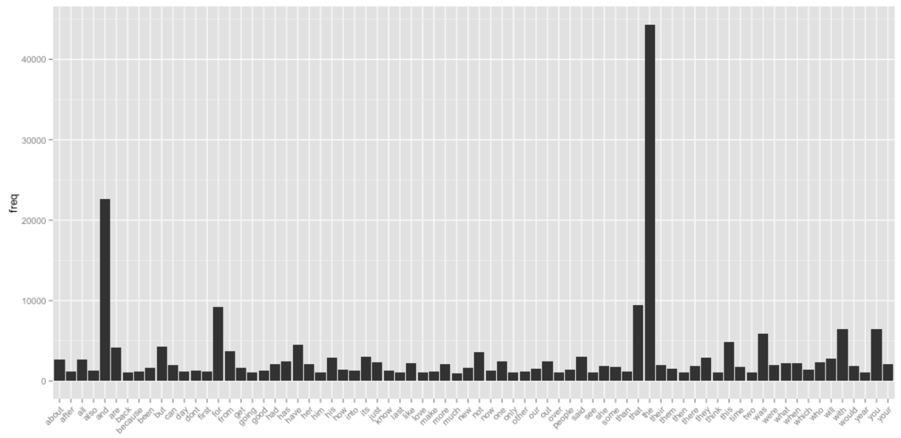
Three differnet TermDocumentMatrices (TDM) are now built using the three different n-grams. A frequency of frequencies table is next built by first calculating the rowsums in the TDM and then ordering those.

Then a datamatrix is created from this that shows the count for each word or phrase. Now, that we have the counts of the various single words, two words and three word sequences, we can start trying out the various language models.

This project was started out pretty late and a lot of time was spent on overcoming memory issues. At this time, I have the frequency table and am trying out the various language models. The goal is to try the different smoothing models including the Kneser-Nay.

One important part of data exploration is to visualize the data. Here, we plot the histogram to look for specific words that might occur most frequently. Here is the histogram plot (done using ggplot2) of words occurring more 500 times and 5000 times


freq
and
for
that
the
was
with
you
word


freq
about after all also and are back because been but can day dont first for from get going good had has have her him his how into its just know last like love make more much new not now one only other out over people said see she some than that the their them then there they think this time two was were what when which who will with would year you your