Bela Bönte Intelligent Systems EIT Data Science 2021/22

NLP Project - Review Writing Helper for Madrid Airbnb's

Aim:
The project aims to illustrate the possibilities of predicting words and generating text. The focus is more on presenting different methodologies than on optimising the individual methods.

Dataset:
https://www.kaggle.com/rusiano/madrid-airbnb-data
The Kaggle dataset contains AirBnB data. Including reviews from customers, regarding apartments in Madrid.
Special characteristic: Not all entries are in the English language.

Github Repository:
https://github.com/belaboe97/madrid_airbnb_nlp

Problems to solve:
→ To facilitate the writing of reviews, a user should be automatically suggested which words could fit the previous text

1) Data clearing: Besides typical operations, like the elimination of line breaks, multilingualism is a problem in this dataset. The goal is to use only english comments.
2) Data exploration & analytics: It is essential to understand the nature of the dataset. There are many possibilities for exploration. The visual one is one of the fastest.
3) Predict words and word groups: A basis must first be created for this. In other words, a text corpus from which various algorithms can extract information.In addition, appropriate algorithms must be identified and applied.
4) Evaluation of the results: In order to evaluate the results, statistical methods are applied on the one hand and a heuristic approach is pursued via the application on the other.

Used technologies (essentials):

CLD3 Neural Network: "CLD3 is a neural network model for language identification."[1] CLD3 can be imported as a library and the already trained network gives information about the language. The neural network reliably recognizes the common languages

Ngrams: NGrams are the occurring frequency of a word or word combination. Mostly, besides filler words and articles, the words that best represent the content of the text are represented here.

Stupid Backoff Algorithm: Stupid Backoff Alogrithm is a relatively simple algorithm, based on the frequency of the occurring words (NGrams). An unscientific explanation but good to get a quick impression: https://stackoverflow.com/questions/16383194/stupid-backoff-implementation-clarification

Markov Chains:
"A Markov chain or Markov process is a stochastic model describing a sequence of possible events in which the probability of each event depends only on the state attained in the previous event."[2] In simple words, the already existing algorithm is trained on the text corpus,

Word2Vec: "Word2vec is a technique for natural language processing published in 2013. The word2vec algorithm uses a neural network model to learn word associations from a large corpus of text."[3] The large word corpus here is not the corpus drawn from the reviews, but an independent model. The assumption is that words in reviews are related in the same way as in other texts. Therefore, a well-known model of Google is taken. "It includes word vectors for a vocabulary of 3 million words and phrases that they trained on roughly 100 billion words from a Google News dataset." [4]

---

[1] Google, https://github.com/google/cld3, Accessed[29.01.2022]
[2] Wikipedia, https://en.wikipedia.org/wiki/Markov_chain, Accessed[29.01.2022]
[3] Wikipedia, https://en.wikipedia.org/wiki/Word2vec, Accessed[29.01.2022]
[4] McCormick, https://mccormickml.com/2016/04/12/googles-pretrained-word2vec-model-in-python/, Accessed[30.01.2022]

Experiment(s) done:

Visual-Analytics:

- *NGrams:* The NGrams show the most frequently occurring words in the text corpus in descending order.

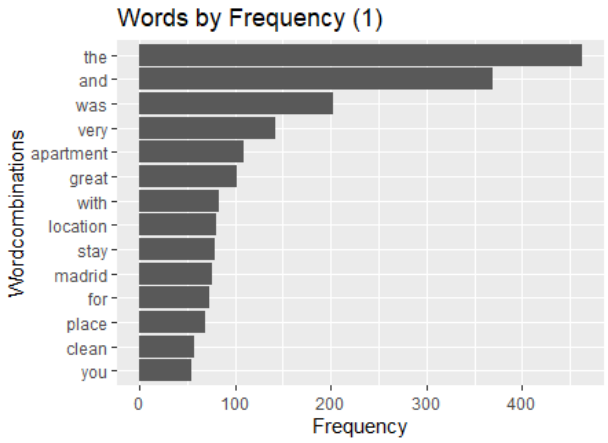

Figure 1: Frequency Barplot for one word

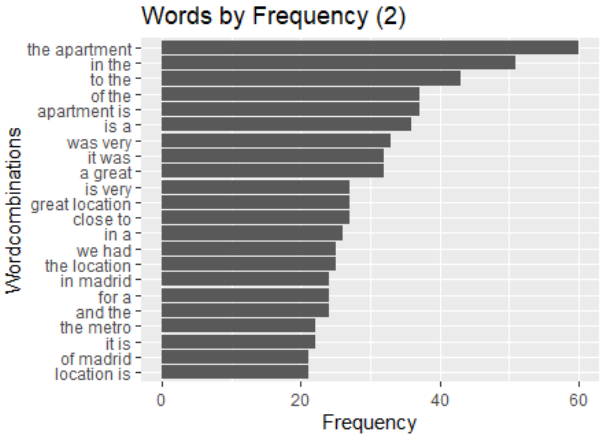

Figure 2: Frequency Barplot for two words

- *Wordcloud:* The Wordcloud displays frequently occurring words in the text corpus. It helps to get an overview of the nature of the text

Figure 3: Word Cloud on cleaned data

Hands-on:

To get an impression of the words suggested by the methods, a console application was developed.

How it works:

1) Enter word or sentence as initial state
2) A List of predictions is displayed
3) Choose an operation: End the sentence, Enter text on your own (no matching prediction), select word or word group
4) Labels give information as which method predicted the word
5) The last word (next initial state) will be compared to the other predicted words. And the similarity is calculated.

```
Please enter a starting word or sentence: 100
[1] "Please enter an english word or sentence"
Please enter a starting word or sentence: When we arrived
[1] "The Review:"
[1] "When we arrived"
[1] "_____"
[1] "This are the predictions"
      predictions          labels
1           <EOS> End of sentence
2      Enter text    Custom input
3             and             sbo
4             was             sbo
5            from             sbo
6          almost          markov
7           hours          markov
8             and          markov
9  And go further          markov
10   And has free          markov
[1] "_____"
Choose a prediction by number: 5
[1] "and"
[1] "and"
[1] "_____"
[1] "This are the similiarities to the chosen (last) word: fro
        was from    almost      hours   further      free
1 0.1528511    1 0.3157228 0.09808804 0.1916077 0.2181916
[1] "_____"
[1] "The Review:"
[1] "When we arrived from"
[1] "_____"
[1] "This are the predictions"
                predictions          labels
1                <EOS> End of sentence
2               Enter text    Custom input
3                    there             sbo
4                      the             sbo
5                    plaza             sbo
6                      the          markov
7                     host          markov
8                definitely          markov
9     Juan's associate and          markov
10 The beautiful apartment          markov
[1] "_____"
Choose a prediction by number: 4
```

Figure 4: Demonstration of word prediction

Analysis of results

Analytics:

- SBO: To evaluate the SBO algorithm there is built-in functionality in the library. In this process, the data is divided into a training and a test data set. The accuracy for correct word predictions in the test data set is then calculated.
  Accuracy: 0.424, It is also possible to plot which words are responsible for the accuracy.
  The graphic shows that especially the 50th most frequently occurring words are responsible for a large part of the correct predictions.

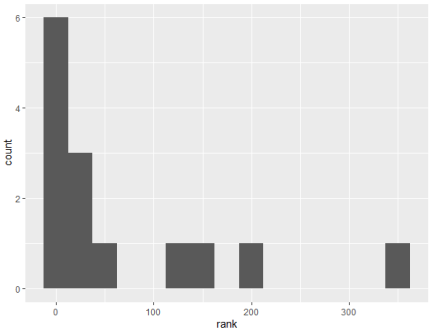

Figure 5: Accurate predictions with SBO dictionary.

- Similarity Check: This method aims to give the user a constant feedback on how good the other predictions were, based on the final word selected.
  On a purely empirical approach, no serious differences were found after word selection. The expectation was that there might be a stronger similarity between words predicted by SBO and words predicted by Markov Chains .

Hands-on & Conclusion:

The experiments performed and the console application show how different methods, words and groups of words can be predicted. The empirical data suggests the potential of such technologies, but some steps still need to be optimised.
A sample review that was generated with the console application:

```
'_____________________________________________________"
'The Review:"
'This flat was great <eos> we had everything you need for the children <eos>"
'_____________________________________________________"
```

Every now and then, the words only fit conditionally well with the sentence that already exists. The console application also allows you to insert your own sentence groups. In this sentence, "This flat" and "we had" were entered manually.
A good way to improve the predictions would be to use more reviews and a uniform cleaning of the data as well as word stemming of the existing words.