Johns Hopkins University Data Science Capstone

Zain Naboulsi

2023-03-22

Contents

Abstract

In today's digital landscape, individuals worldwide devote more time, perhaps too much time, to their mobile devices, engaging in various activities such as emailing, social networking, and banking. However, typing on these devices can pose difficulties. SwiftKey, the corporate partner for this capstone project, offers a solution with its smart keyboard that eases typing on mobile devices. A key feature of SwiftKey's keyboard includes predictive text models, which suggest potential words to users based on their previous input. For example, after a user types "I went to the", the keyboard could suggest "gym", "store", or "restaurant". The goal of this capstone project involves both understanding and developing predictive text models similar to those SwiftKey employs.

This project starts with analyzing a large body of text documents to understand data structure and word organization. This process includes cleaning and analyzing text data, followed by the construction and sampling of a predictive text model. The final goal is to create a predictive text product. We will be utilizing all the skills acquired during the Data Science Specialization, with a focus on text data analysis and natural language processing (NLP).

Inital Data Load

In this portion of the project, we are undertaking the first step of our analysis: loading the data. We begin by specifying the location of our data, which resides in text files within the 'SwiftKeyData' folder. To facilitate subsequent analysis, we utilize the 'readtext' function to import the data from these text files into our working environment. The data at this point exists as a collection of separate text documents. To make the data easier to work with and analyze collectively, we then transform this collection of documents into a 'corpus'. In the realm of text analysis, a 'corpus' refers to a structured set of texts, which serves as our comprehensive data set for the subsequent stages of our investigation. Finally, we create a document-feature matrix to get an idea of the word frequency that exists.

References

Appendix: All Source Code

```r
# set a seed in case we use any random items
set.seed(1337)


# Set the names of the packages and libraries you want to install
# Most notably load up all the quanteda packages we will need
required_libraries <- c("quanteda","quanteda.textmodels","quanteda.textstats",
                        "quanteda.textplots")

# Install missing packages and load all required libraries
for (lib in required_libraries) {
  if (!requireNamespace(lib, quietly = TRUE)) {
    install.packages(lib)
  }
  library(lib, character.only = TRUE)
}
```