# Product Review Generator

Intelligent Systems - NLP Assignment

Eleonora Renz

2022-01-30

## Problem to Solve

Review sections are a staple of e-commerce and business advertised online nowadays. Good reviews can convince new potential customers to actually buy a product or service, while bad reviews can deter them from doing so. Because of this, a lot of money is at stake, so companies want to make sure they have good reviews to show. Services offering fake review writers are sometimes hired by companies to boost their appearance online or, in an even more malicious attempt, hired to spam competitors with bad reviews. Automating the task of writing fake reviews, both positive or negative ones, is, while malicious, in high demand. However, researching and understanding the process of writing such fake reviews also offers the opportunities to find ways to combat the fake review wave rushing across the internet.

In this project, a product review generator is built based on Amazon product review data. The data set is taken from Kaggle and can be found under https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products?select=Datafiniti_Amazon_Consumer_Reviews_of_Amazon_Products.csv.

## Experiments Done

Before looking into the models used to create the product review generator, the used data is first explored and then prepared.

### Data Exploration

```
## [1] 28332
```

```
##  [1] "id"                 "dateAdded"          "dateUpdated"
##  [4] "name"               "asins"              "brand"
##  [7] "categories"         "primaryCategories"  "imageURLs"
## [10] "keys"               "manufacturer"       "manufacturerNumber"
## [13] "reviews.date"       "reviews.dateSeen"   "reviews.didPurchase"
## [16] "reviews.doRecommend" "reviews.id"        "reviews.numHelpful"
## [19] "reviews.rating"     "reviews.sourceURLs" "reviews.text"
## [22] "reviews.title"      "reviews.username"   "sourceURLs"
```

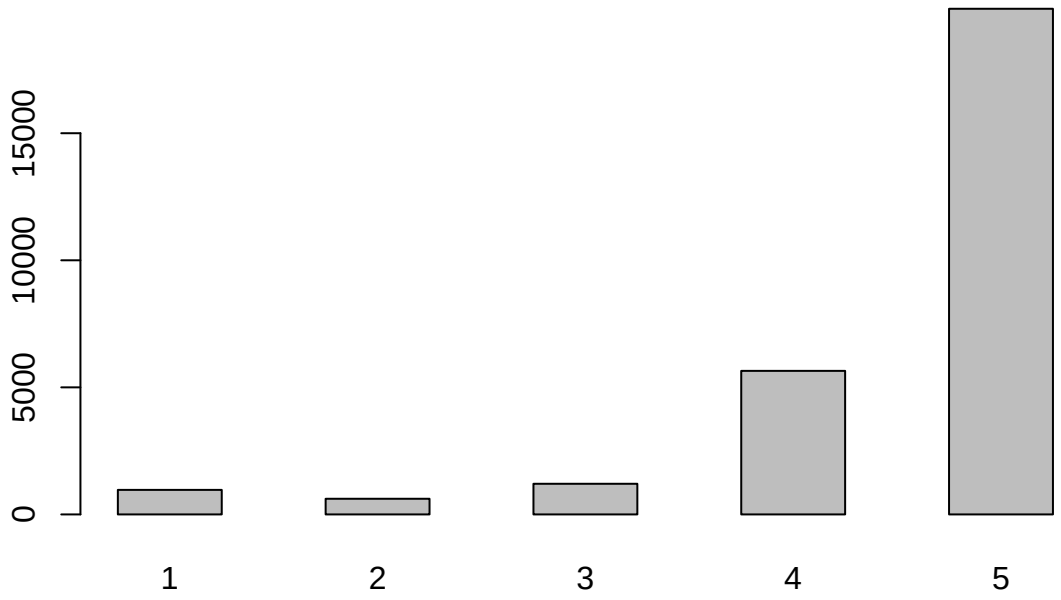
The data set in its raw form contains 28332 rows over various attributes. Many of these attributes are not interesting for the task at hand and will be removed, such as user data and product details.

Typically, first implementations of text generators are based on a common theme, such as text in the style of a specific author. When thinking about how to build a product review generator, it needs to be considered that there is no one common theme. Every product has its own intecracies that are described in a review

and the writing style is different between all users leaving reviews. Because of this, a look is taken on the primary categories of all reviews at hand to try to categorize them as such.

```
##      Animals & Pet Supplies                  Electronics
##                           6                        13995
##       Electronics,Furniture            Electronics,Media
##                           2                          185
##             Health & Beauty                Home & Garden
##                       12071                            2
##             Office Supplies Office Supplies,Electronics
##                           9                          386
##    Toys & Games,Electronics
##                        1676
```

Another idea obvious in conjunction with product reviews is that these reviews can either be positive or negative. The data set includes a 1-5 star rating to each review. The distribution is rather unbalanced with the vast majority being positive reviews with either 4 or 5 stars, which should be noted.




## Data Preparation

Based on these first thoughts, the following attributes are selected to be worked with.

```
reviews <- reviews %>% select(id, name, categories, primaryCategories,
                              reviews.rating, reviews.text, reviews.title)
reviews_text <- as.character(reviews$reviews.text)
```

The corpus used for training the model will be based on the attribute reviews.text, and thus this is the most important part of the data set for this project. Additionally, new columns based on the ideas of importance of categories and review positivity are created.

```
# add column based on category
electronics <- c("Electronics", "Electronics,Media", "Electronics,Furniture",
                 "Supplies,Electronics", "Toys & Games,Electronics")
reviews$mergedCategory <- with(reviews, ifelse(reviews$primaryCategories
                            %in% electronics, "electronics", "non-electronics"))
# add column based on rating
reviews$reviews.rating_positivity <- with(reviews, ifelse(
                            reviews$reviews.rating >3, "positive", "negative"))
```

Also, the corpus is checked for utf8 encoding validity as seen in class.

```
reviews_text[!utf8_valid(reviews_text)] # 0 is good
```

```
## character(0)
```

Next, some tests with subword tokenaziation are completed.

```
bpe_model <- bpe(unlist(reviews_text[1:14166]))
subtoks <- bpe_encode(bpe_model, x = reviews_text[14167:28332], type = "subwords")
```

And, a document feature matrix is built to analyze the most and least frequent features in the corpus.

```
# Top features
dfm_reviews_text <- dfm(tokens(reviews_text))
topfeatures(dfm_reviews_text)
```

```
##     .   the   and     i    to     ,   for     a    it    is
## 57270 28869 22360 20745 19540 18084 17933 17627 16956 11487
```

```
# Top features without punctuation and stop words
dfm_reviews_text_1 <- dfm(tokens(reviews_text, remove_punct = TRUE))
dfm_reviews_text_2 <- dfm_remove(dfm_reviews_text_1, stopwords("en"))
topfeatures(dfm_reviews_text_2)
```

```
##     great batteries    tablet      good     price       use    amazon      love
##      9409      8093      6632      5880      5186      4434      3729      3535
##    bought       can
##      3177      2999
```

As we saw earlier, the most common category is electronics, so the top features in the data set seem to be very logically fitting this category in a review style text.

## Models

All models created are based on Markov Chains. The principle idea of Markov Chains is that every word in the corpus is connected to every other word with varying probabilities.

The first model created is a general model based on the entire corpus.

**General Model**

```r
# Build the model
markov_model <-
  generate_markovify_model(
    input_text = reviews_text,
    markov_state_size = 2L,
    max_overlap_total = 25,
    max_overlap_ratio = .85
  )

# Test model
markovify_text(
  markov_model = markov_model,
  maximum_sentence_length = NULL,
  output_column_name = 'textProductReview',
  count = 5,
  tries = 100,
  only_distinct = TRUE,
  return_message = FALSE
)
```

```
## # A tibble: 5 x 2
##   idRow textProductReview
##   <int> <chr>
## 1     1 Seems to be independant of my remotes
## 2     2 Duracell/Energizers last 3-4 time longer.
## 3     3 We bought this so the replacement to a friend, and he is still working ~
## 4     4 I had to replace kindle.
## 5     5 This allows me to use and small enough to read from another popular tab~
```

The second experiment split the corpus based on the rating_positivity engineered feature into a positive and negative corpus from which two models were trained respectively. The model code blocks are excluded from here on to save space and as they are very similar.

**Positive Model Output**

```
## # A tibble: 5 x 2
##   idRow textProductReview
##   <int> <chr>
## 1     1 Strong wi fi range.
## 2     2 As a first tablet and I am pleased with them
## 3     3 Good value will be using these batteries seem to have and own to watch ~
## 4     4 It is extremely beneficial.
## 5     5 I very satisfied EXCEPT it just goes blank repeatedly and for the money.
```

**Negative Model Output**

```
## # A tibble: 5 x 2
##   idRow textProductReview
##   <int> <chr>
## 1     1 I will not charge.
## 2     2 arrived on time ....now I have to walk my dog has a kid proof case.
## 3     3 But I am lucky if a new charger and it freezes up at times.
## 4     4 All the dying batteries are faulty.
## 5     5 I've usually had very short life in every way, options are limited and ~
```

A third try splits the corpus based on the primary category of electronic or non-electronic.

**Electronic Category Model Output**

```
## # A tibble: 5 x 2
##   idRow textProductReview
##   <int> <chr>
## 1     1 We bought the fire tablet is great.
## 2     2 Very satisfied with my amazon fire.
## 3     3 Upgraded from a tablet.
## 4     4 This kindle has been very durable and tough.
## 5     5 Plus, it has page advance buttons on both accounts and they are on stor~
```

**Non-Electronic Category Model Output**

```
## # A tibble: 5 x 2
##   idRow textProductReview
##   <int> <chr>
## 1     1 Ok, batteries but they do work... but not as good as the big name brand~
## 2     2 And the Xbox everyday and I would not recommend this product!
## 3     3 These are great for what we used before and was quite amazed at how wel~
## 4     4 So far batteries seem to be good.
## 5     5 Now he is in a battery factory, that these will last longer than durace~
```

Lastly, an attempt at a product aware review generator is made. This is done by forcing the generated review to start with the desired product name.

**Product Aware Model Output**

Product that reviews should be generated for is: *Batteries*.

```
## # A tibble: 4 x 3
##   idRow wordStart     textProductReview
##   <int> <chr>         <chr>
## 1     1 The batteries The batteries themselves come pre-wrapped in groups of 4 ~
## 2     2 The batteries The batteries themselves come pre-wrapped in groups of fo~
## 3     1 Batteries     Batteries usually leak after they were leaking so I don't~
## 4     2 Batteries     Batteries seem to be independant of my daughter uses it e~
```

## Analysis of Results

While some of the product reviews generated sound very realistic, especially because humans reviewing product do not always follow grammar rules themselves; other generated reviews are very much out there and do not make any sense whatsoever. The positive and negative trained models clearly show a sentiment connotation in the generated text, although this is working better for the positive reviews. A reason for this can be that the negative reviews include 1-3 starts, which is a wide range and technically includes rather neutral reviews as well. For the category based approach, the results for both primary categories are heavily leaning towards electronic goods. This is probably due to the data set being primarily focused on electronic goods in the first place and the non-electronic category while not having electronics as their main category, still including electronics as a side-category. The generated reviews based on starting words do talk about the inputted topic, but not so much because of context awareness and rather because valid sentences were formulated while being forced to include the inputted product name in the generated review.

### Ideas for Further Steps

*Improvements on Current Model*

While some sentences sounded very realistic, others lost track of the message they were conveying and did not make any sense. One option to improve the current models would be to use more training data. An alternative approach to getting better sounding sentences would be to include pre-trained models, such as the Open AI models. These, however, are not limited to reviews and change the style of text generated.

*Fake Review Predictor (Discriminator)*

Under the aspect of wanting to understand more on fake review generators in order to combat them, a discriminator can be implemented. The discriminator would try to distinguish between real and fake reviews and can find meaning when wanting to remove fake reviews from sites.

*Sentiment Analysis*

Building on the existing model, a sentiment model can be trained on the real data and used to give a 1-5 rating on the generated review text.

## Link to GitHub

https://github.com/Ellomarshmallow/product-review-generator