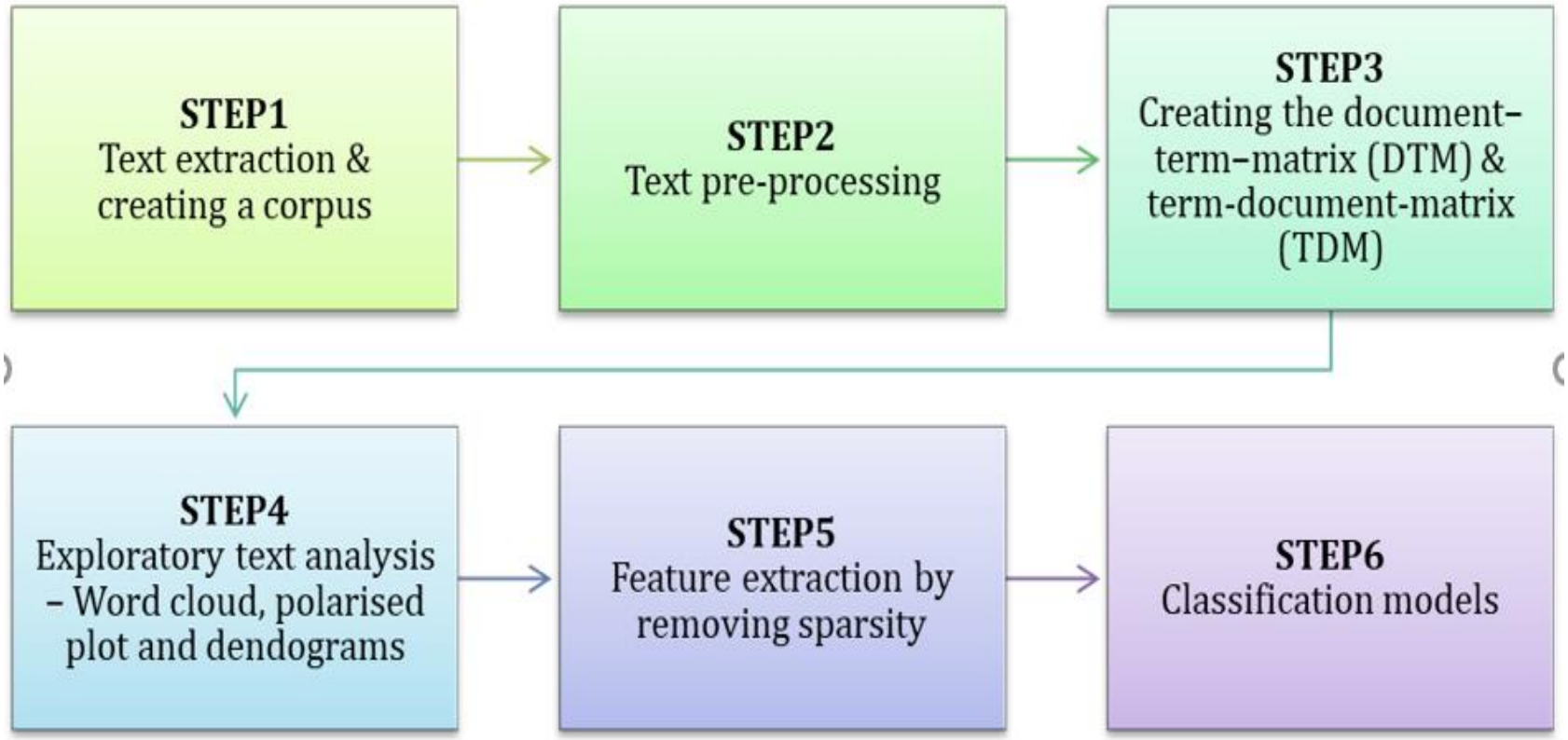
CSDA 1050
Capston Project Podcast

YORK
UNIVERSITÉ
UNIVERSITY

Objectives

- To help Canadian podcast publishers and provide recommendations to understand different segments of podcast listeners.
- To inform business opportunities in this growing media sector.
- To identify growth areas compare with previous research.

High level approach



STEP1 — Text extraction & creating a corpus

- The column **Review.Text** contains the customer reviews received for various products. This is the focus for our analysis. We will now try to understand how to represent text as a data frame.
- First, the review.text is converted into a collection of text documents or "*Corpus*".
- To convert the text into a corpus, we use the "tm" package in R.
- In order to create a corpus using tm, we need to pass a "Source" object as a parameter to the VCorpus method.

STEP2 — Text Pre-processing

To ensure that the DTM and TDM are cleaned up and represent the core set of relevant words, a set of pre-processing activities need to be performed on the corpus. This is similar to the data clean-up done for structured data before data mining. The following are some of the common pre-processing steps:

1. Convert to lower case
2. Remove punctuation
3. Remove stopwords
4. Stemming a document
5. Frequently used words

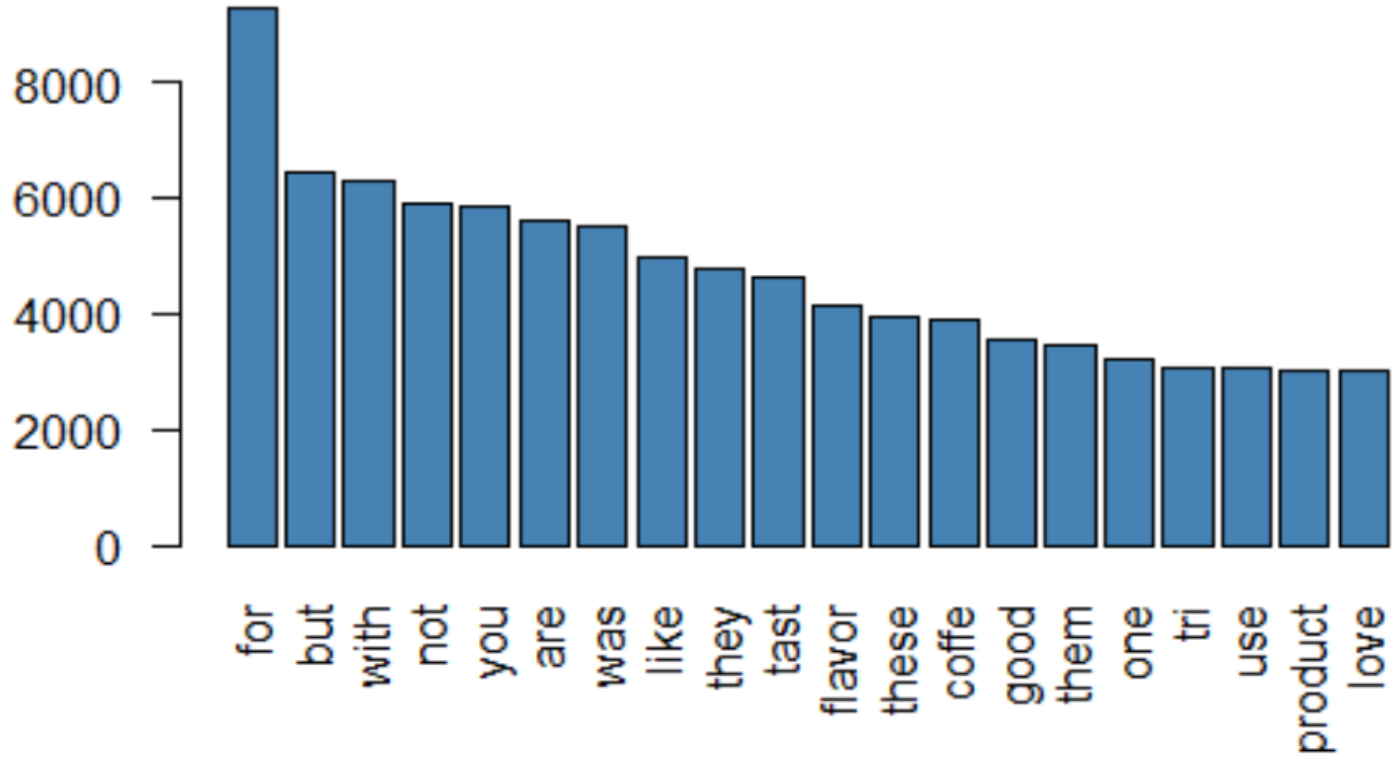
STEP3 — Create the DTM & TDM from the corpus

The pre-processed and cleaned up corpus is converted into a matrix called the document term matrix.

A document-term matrix is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. In a document-term matrix, rows correspond to documents in the collection and columns correspond to terms.

The term-document matrix is a transpose of the document-term matrix. It is generally used for language analysis. An easy way to start analyzing the information is to change the DTM/TDM into a simple matrix using as.matrix().

STEP4 — Exploratory text analysis

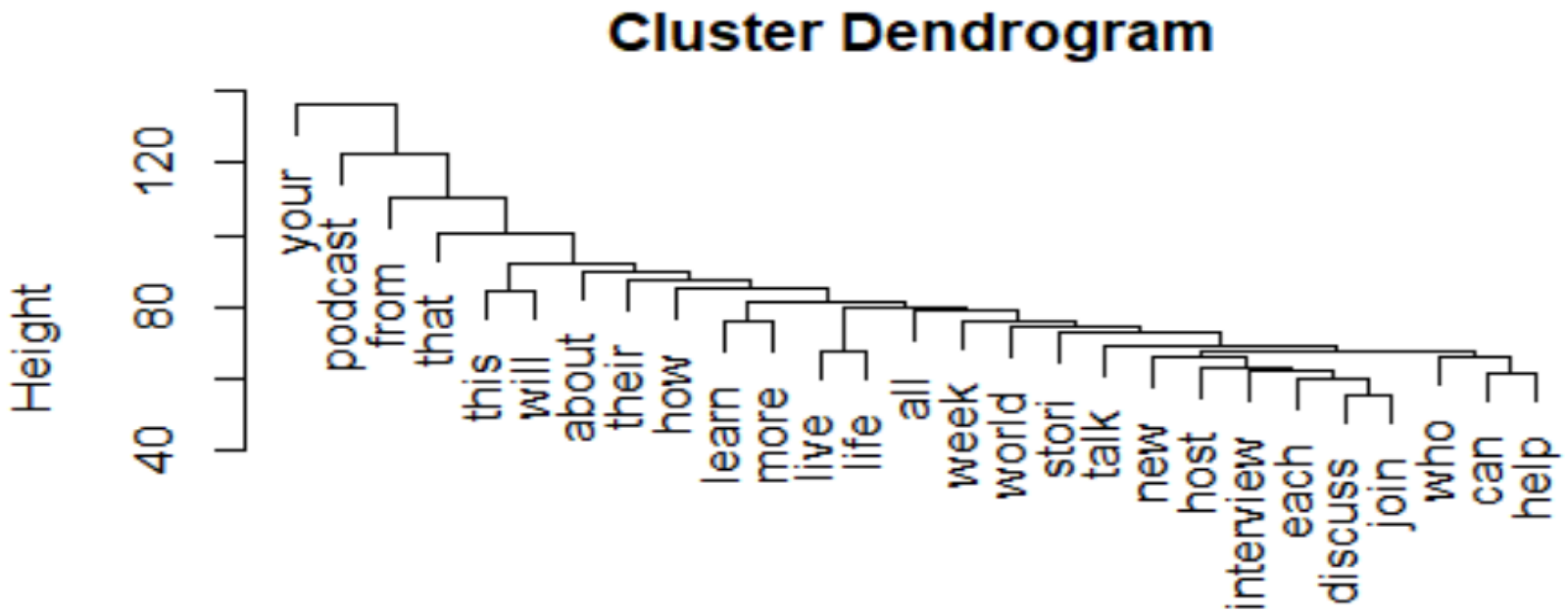


Word clouds

Word clouds:- Word cloud is a common way of visualizing a text corpus to understand the frequently used words. The word cloud varies the size of the words based on the frequency. The word cloud can receive a set of colors or a color palette as input to distinguish between the more and the lesser frequent words in the cloud.

Simple word clustering: -

- Simple word clustering**: -** Word clustering is used to identify word groups used together, based on frequency distance. This is a dimension reduction technique. It helps in grouping words into related clusters. Word clusters are visualized with dendrograms.



STEP5 — Feature extraction by removing sparsity

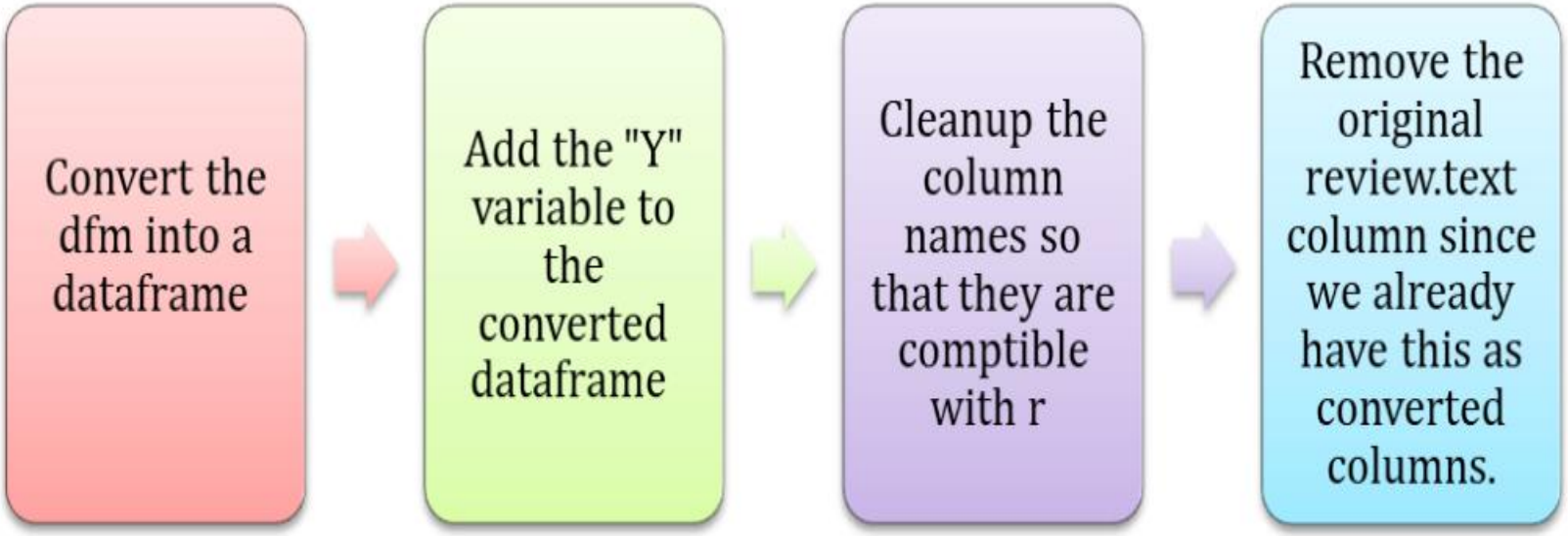
Concept of sparsity

- Sparsity is related to the document frequency of a term. In DTM, since the terms form the columns, every document will have several columns each representing one term—a unigram, bi-gram, tri-gram, etc.

Feature extraction

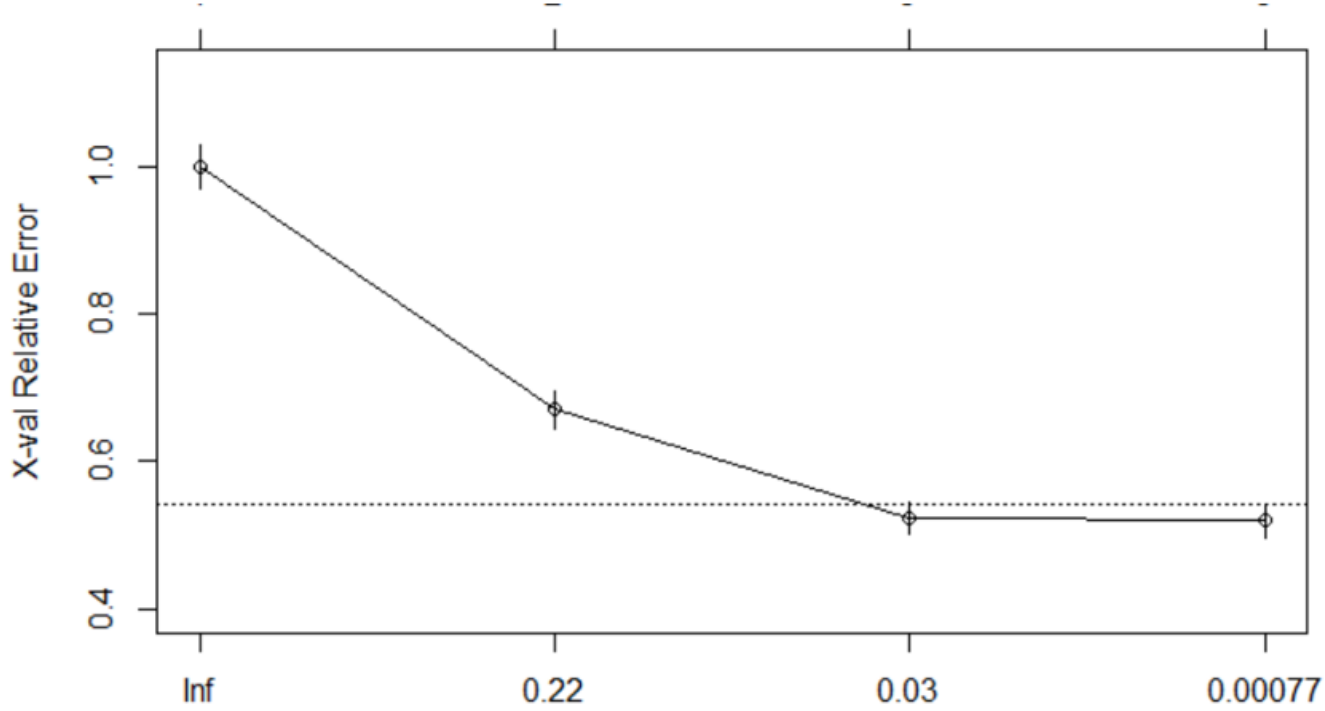
- The exploratory text analysis has given several insights based on the customer reviews. We will now use the same review text as predictor variable to predict whether the product will be recommended by the customer. In terms of classification algorithms used, there is not much of a difference between data and text input. We will try 3 of the most popular classification algorithms—CART, Random forest

STEP6 — Building the Classification Models

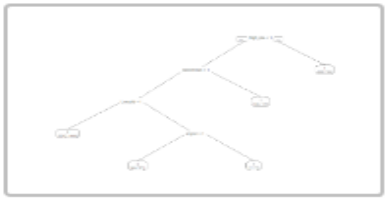


CART model

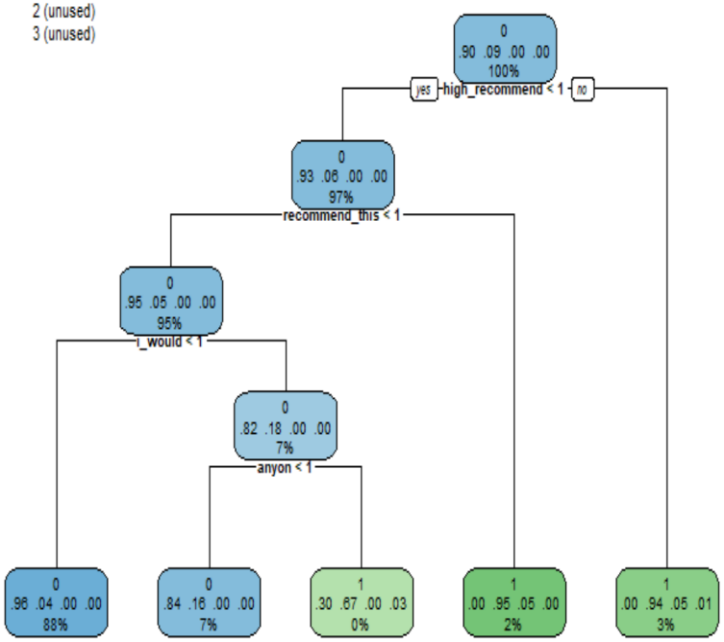
```
## Build the CART model
tree=rpart(formula = recommend ~ ., data = reviewtokensdf, method="class",control =
rpart.control(minsplit = 200,  minbucket = 30, cp = 0.0001))
printcp(tree)
plotcp(tree)
```

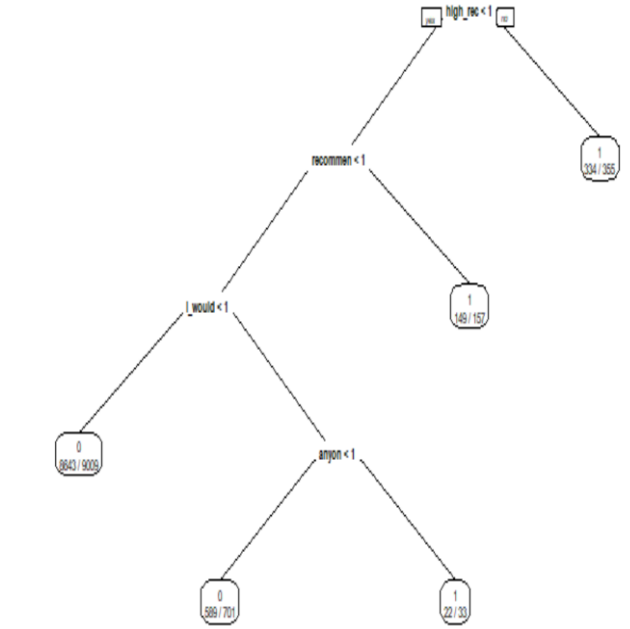


CART model



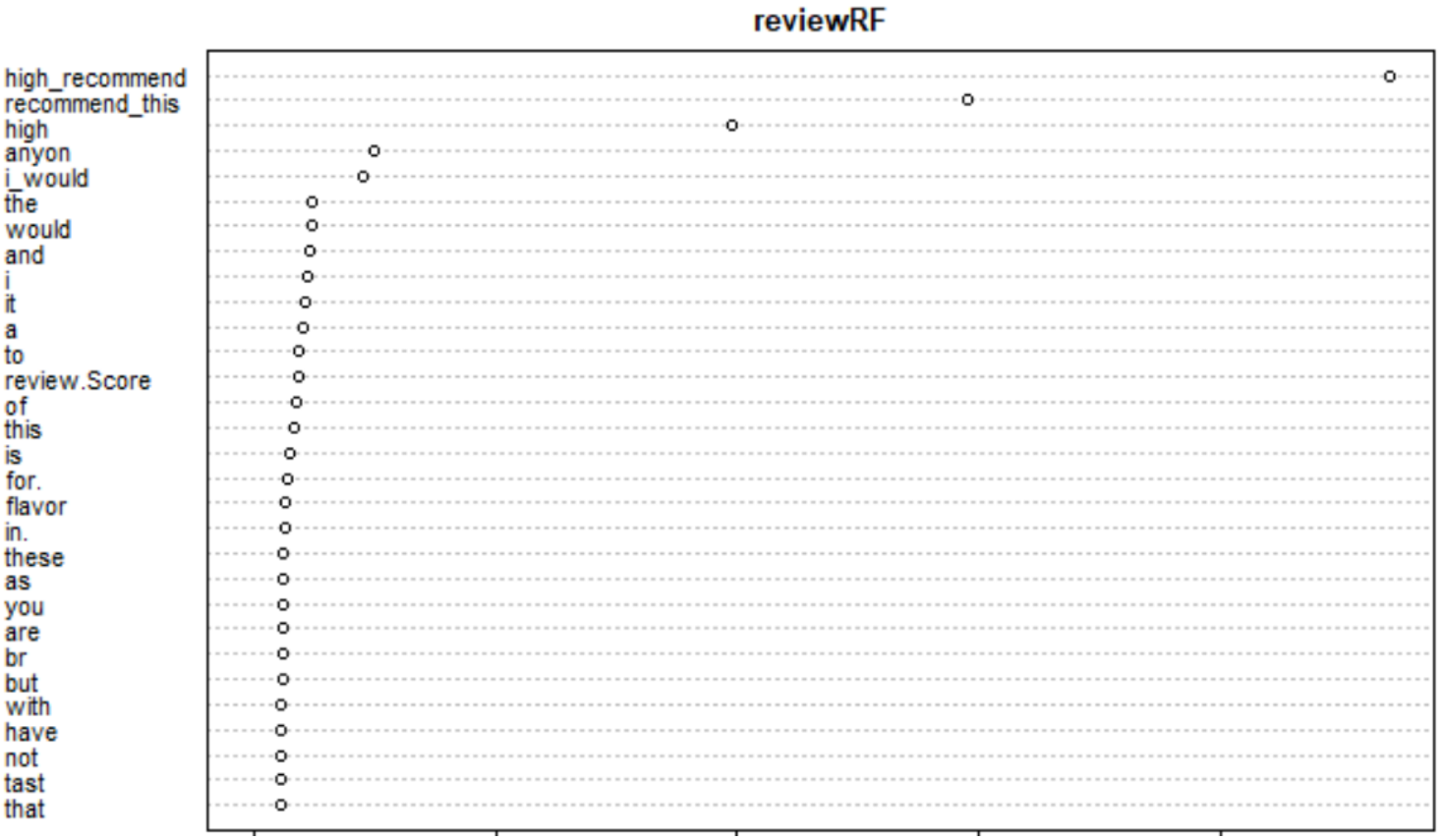
1
2 (unused)
3 (unused)





Random forest

The next classification algorithm we will use is the Random forest. We will examine the varimp plot of the random forest model to understand which words affect the classification the most.



Conclusion

In sync with the CART model, the varimp plot of the Random forest model also , words like “High Rec”, “recommend”, “I would”, etc are used by happy customers — i.e., customers do recommend the product. The tree can be interpreted further to understand the word patterns used by customers who recommend the product vs those who don’t.