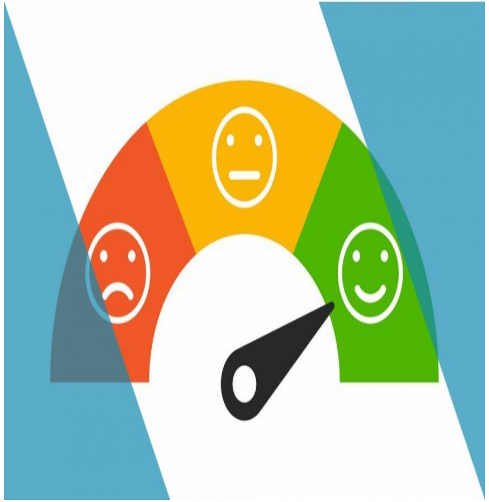
Analyzing the Sentiment of Ice Cream Reviews

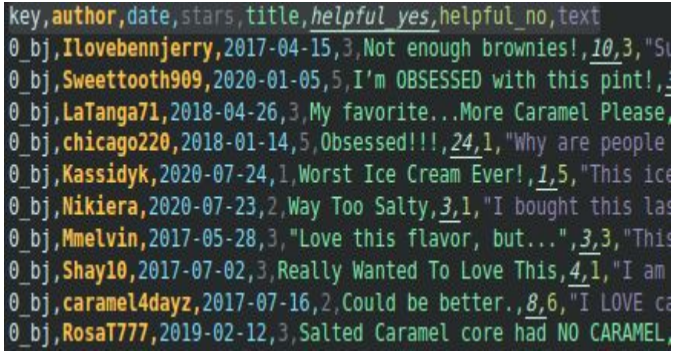
Eric Sclafani

Sentiment analysis: an overview

- A form of **text categorization** which is the act of “assigning a label or category to an entire text or document” (Jurafsky & Martin 2022).
- The classification of data into different user defined classes
- Commonly used by companies to gauge the “sentiment” value of a product review i.e. if someone likes or dislikes something.
- This kind of classification is called **binary classification**, which means classifying things into two classes.
- It is possible to use more classes, however.
- *Many, many* different ways to perform sentiment analysis.

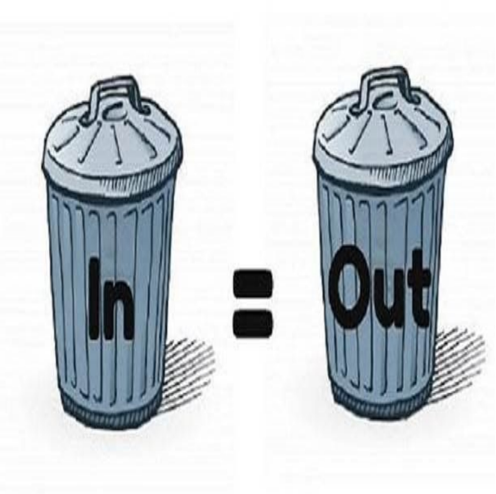


Data preprocessing

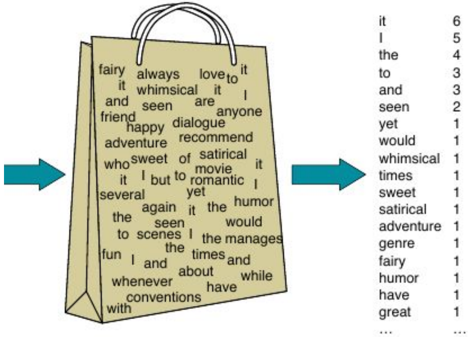


```
key,author,date,stars,title,helpful_yes,helpful_no,text
0_bj,Ilovebennjerry,2017-04-15,3,Not enough brownies!,10,3,"Su
0_bj,Sweettooth909,2020-01-05,5,I'm OBSESSED with this pint!,
0_bj,LaTanga71,2018-04-26,3,My favorite...More Caramel Please,
0_bj,chicago220,2018-01-14,5,Obsessed!!!,24,1,"Why are people
0_bj,Kassidyk,2020-07-24,1,Worst Ice Cream Ever!,1,5,"This ice
0_bj,Nikiera,2020-07-23,2,Way Too Salty,3,1,"I bought this las
0_bj,Mmelvin,2017-05-28,3,"Love this flavor, but...",3,3,"This
0_bj,Shay10,2017-07-02,3,Really Wanted To Love This,4,1,"I am
0_bj,caramel4dayz,2017-07-16,2,Could be better.,8,6,"I LOVE ca
0_bj,RosaT777,2019-02-12,3,Salted Caramel core had NO CARAMEL,
```

- Perhaps the most important part of any computational task involving data
- My steps:
  a. **Drop useless columns** (key, author, date, helpful_yes/no)
  b. **Convert stars** -> 0s and 1s
  c. **Drop rows with NA values**
  d. **Clean text data** (remove punc, stop words, lowercase)
     - Stop words = common words with no real value (the, it, he, she, when, where, etc...)
- Due to dropping NA values, I go from ~8k to 5300 reviews.



Multinomial Naive Bayes



it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…

- Why is it called **naive**? We make the naive assumption that all probabilities are independent of each other.
- One way to use naive bayes is as a **bag of words** model.
- Basic idea (simplified):
  - Count each unique word in a review
  - **Prior probability:** Out of all our documents, how many belong to each class?
  - **Likelihood probability:** For each word in a document, what is the probability of that word occurring given each class?
  - For each document, add up the **log** likelihood probabilities of each word given the class and then add that to the **log** prior probability of the class
  - Do this for both classes and then take the argmax (returns whichever class gives the higher probability score)

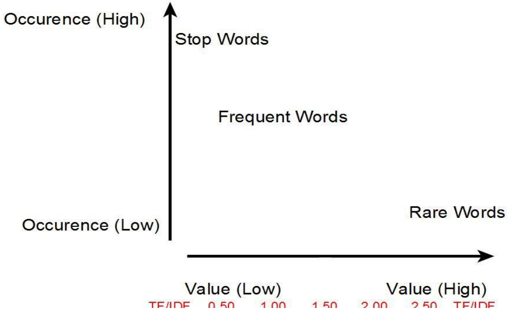
$$c_{NB} = \underset{c \in C}{\operatorname{argmax}} \log P(c) + \sum_{i \in positions} \log P(w_i|c)$$

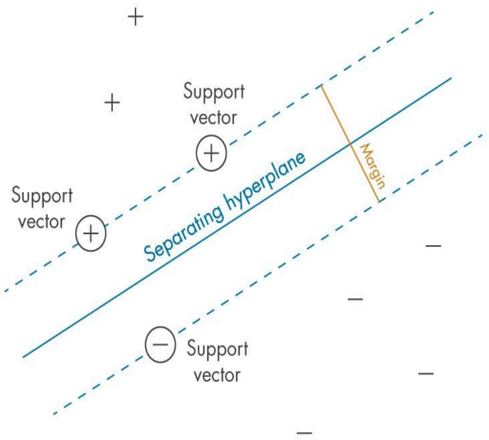
Support Vector Machine

- ❖ Unlike my naive bayes model, I used **sklearn** for the SVM classifier
- ❖ Also unlike NB, instead of a bag of words (BoW) model, I chose to use **TF-IDF** (**T**erm **F**requency **I**nverse **D**ocument **F**requency) to process my data for the SVM.
  - ➢ **BoW** focuses on **word frequencies**
  - ➢ **TF-IDF** puts more emphasis on the # of times a word appears in a single document rather than across all documents.
  - ➢ TF-IDF **vectorizes** the data in preparation for ML
- ❖ After vectorizing, the data gets fed into and fit to sklearn's SVM model.
- ❖ **Goal of SVMs**: find a hyperplane that best separates the data.
  - ➢ Hyperplanes use **support vectors** to determine the maximum margin between the two classes

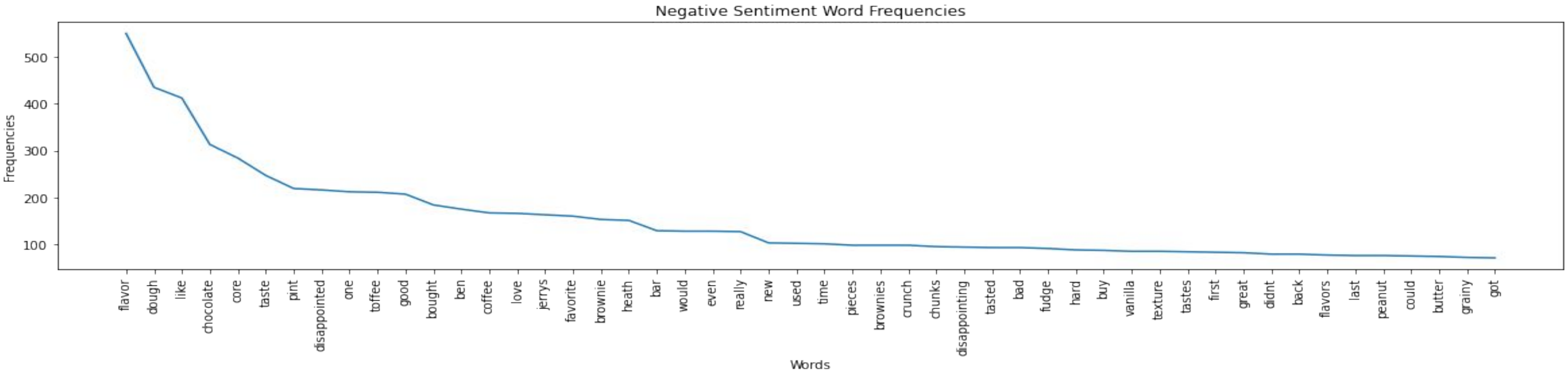
$$tf(t, d) = \log(1 + freq(t, d))$$

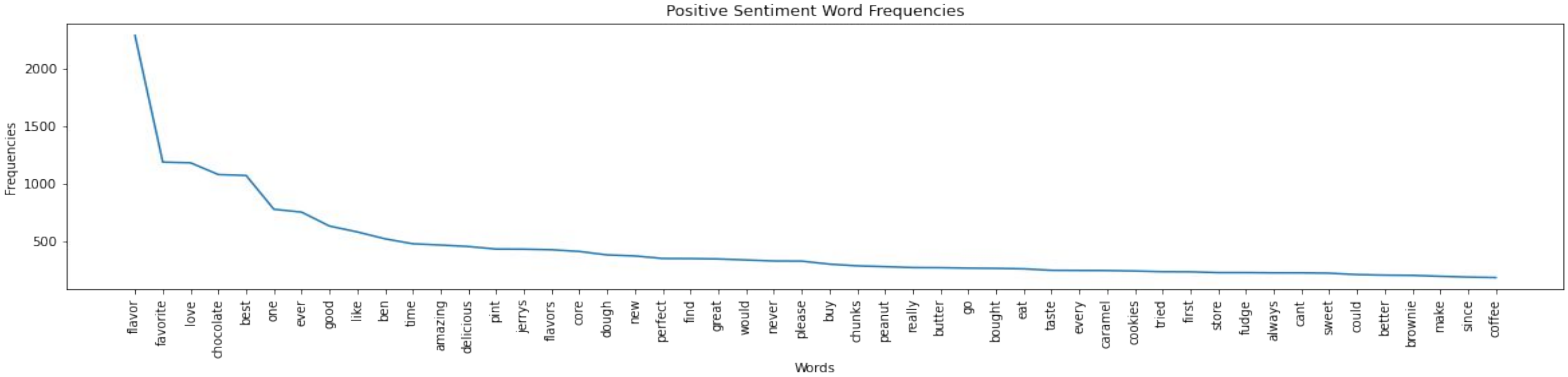
$$idf(t, D) = \log\left(\frac{N}{count(d \in D : t \in d)}\right)$$





Word frequencies: long tail distributions





Results

- As expected, the SVM outperformed the NB model. However, the NB did surprisingly well given its simplicity.
- One difference in the models' results is that sklearn calculates the precision and recall for each class.
- My NB just calculates them regarding whether the model makes a correct prediction or not.

- **Naive Bayes Classifier Results:**
  - Accuracy: 70%
  - Precision: 81%
  - Recall: 80%
  - F1-score: 80%
- **SVM:**
  - Negative class:
    - Precision: 81%
    - Recall: 57%
    - F1-score: 67%
  - Positive class:
    - Precision: 90%
    - Recall: 97%
    - F1-score: 93%

References

- Images:
  - Bag of words image: Jurafsky & Martin p.59 (citation below)
  - https://www.memind.eu/sentiment-analysis-emotion-detection/
  - https://medium.com/@seanjpan/garbage-in-garbage-out-2e781f4d014a
  - https://towardsdatascience.com/tfidf-for-piece-of-text-in-python-43feccaa74f8
  - https://www.mathworks.com/discovery/support-vector-machine.html
- Texts:
  - Jurafsky, M. J. D. H. (2022). *Speech and Language Processing An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition* (2nd ed.). PEARSON INDIA.
  - Stecanella, B. (2019, May 10). *Understanding TF-ID: A Simple Introduction*. MonkeyLearn. https://monkeylearn.com/blog/what-is-tf-idf/