LONDON
METROPOLITAN
UNIVERSITY

# Machine Learning
# CS7052

## Lecture 11, Working with text data

Dr. Elaheh Homayounvala

Week 11

# Outline of today’s lecture

- Review last week, unsupervised learning
- Working with text data

# Review last week

Unsupervised learning

Clustering

# Unsupervised learning

All kinds of machine learning where:

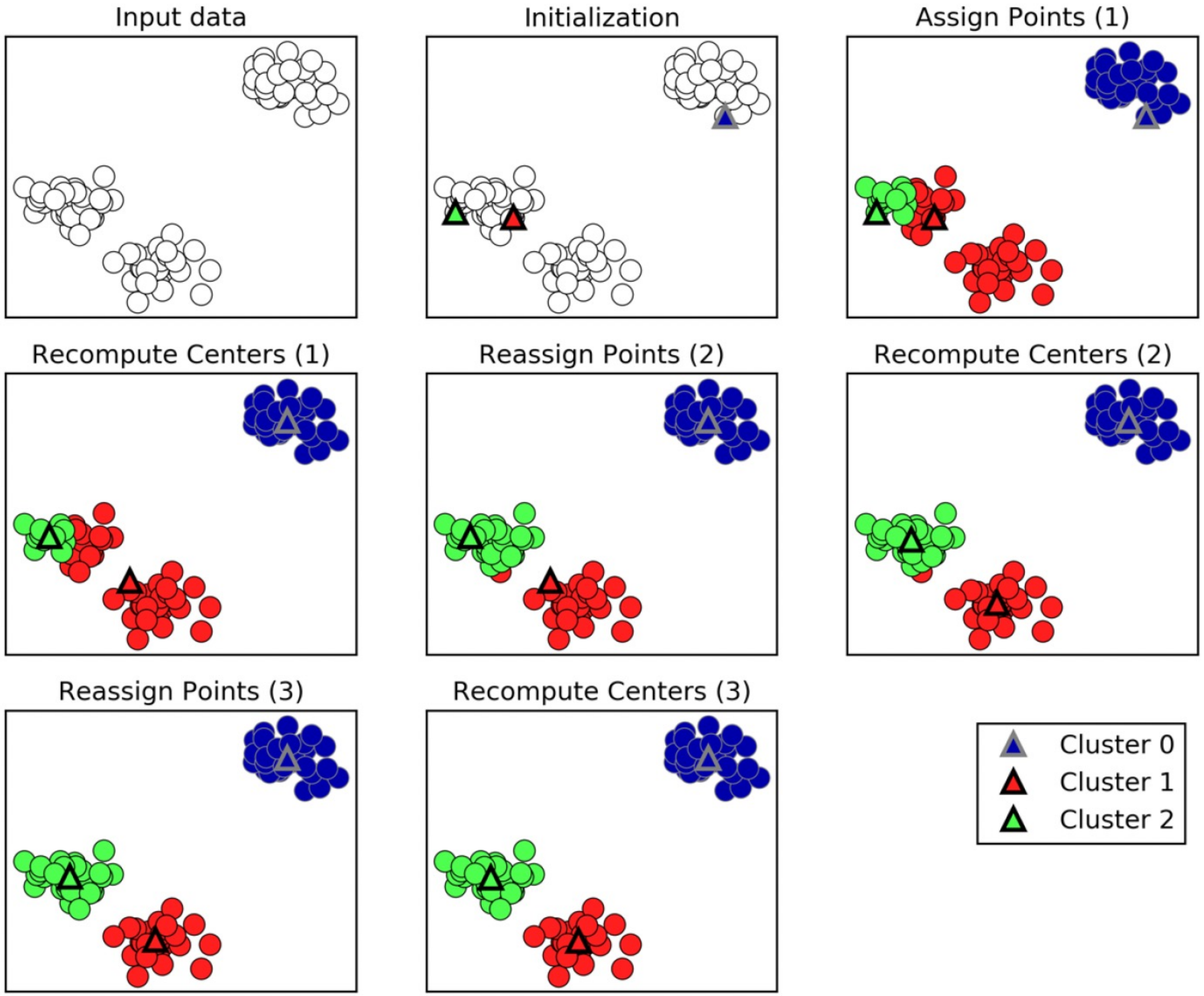
- There is no known output
- No teacher to instruct the learning algorithm.
- The learning algorithm is just shown the input data and asked to extract knowledge from this data.

# Clustering

- Is the task of partitioning the dataset into groups, called clusters.

- The goal is to split up the data in such a way that:
  - points within a single cluster are very similar and
  - points in different clusters are different.

# K-means clustering example



Muller and Guido book page 171

*Figure 3-23. Input data and three steps of the k-means algorithm*

# Agglomerative Clustering

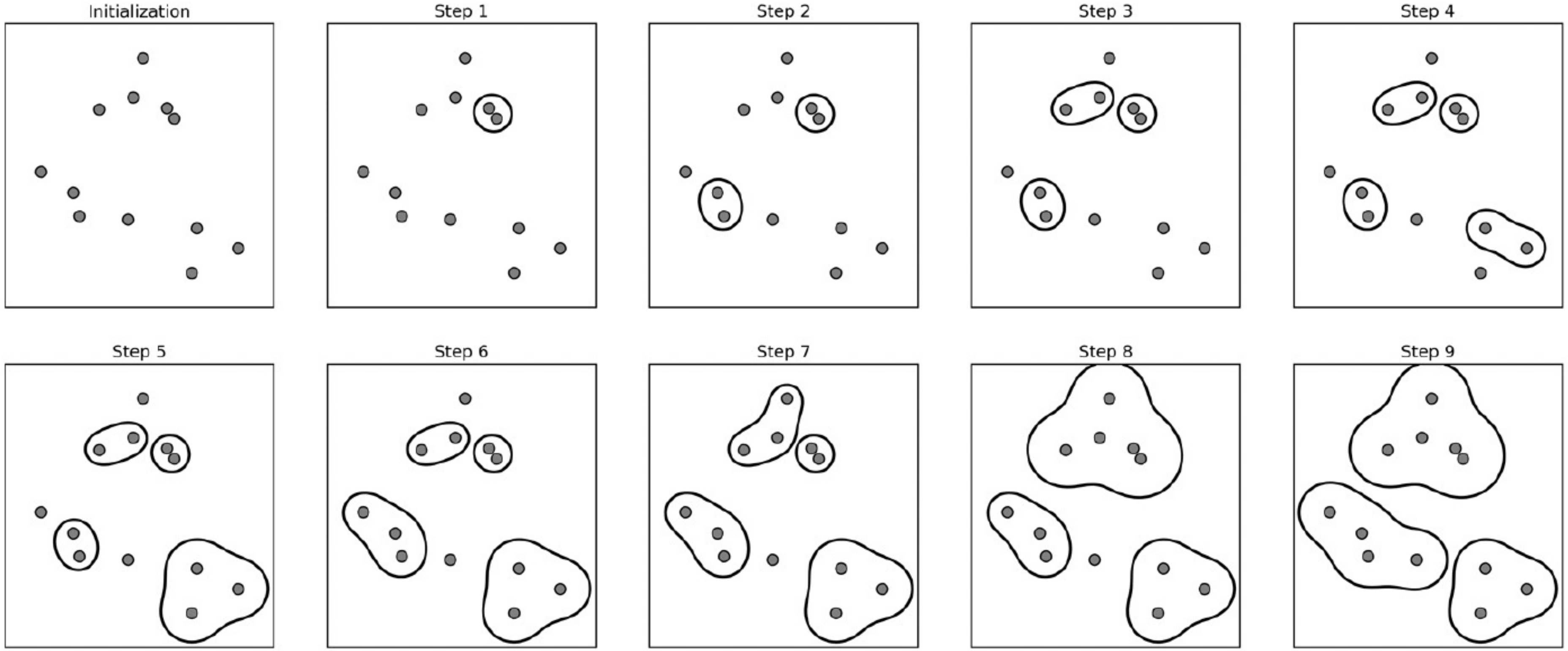


*Figure 3-33. Agglomerative clustering iteratively joins the two closest clusters*

Muller and Guido book page 185

# Agglomerative Clustering, an example

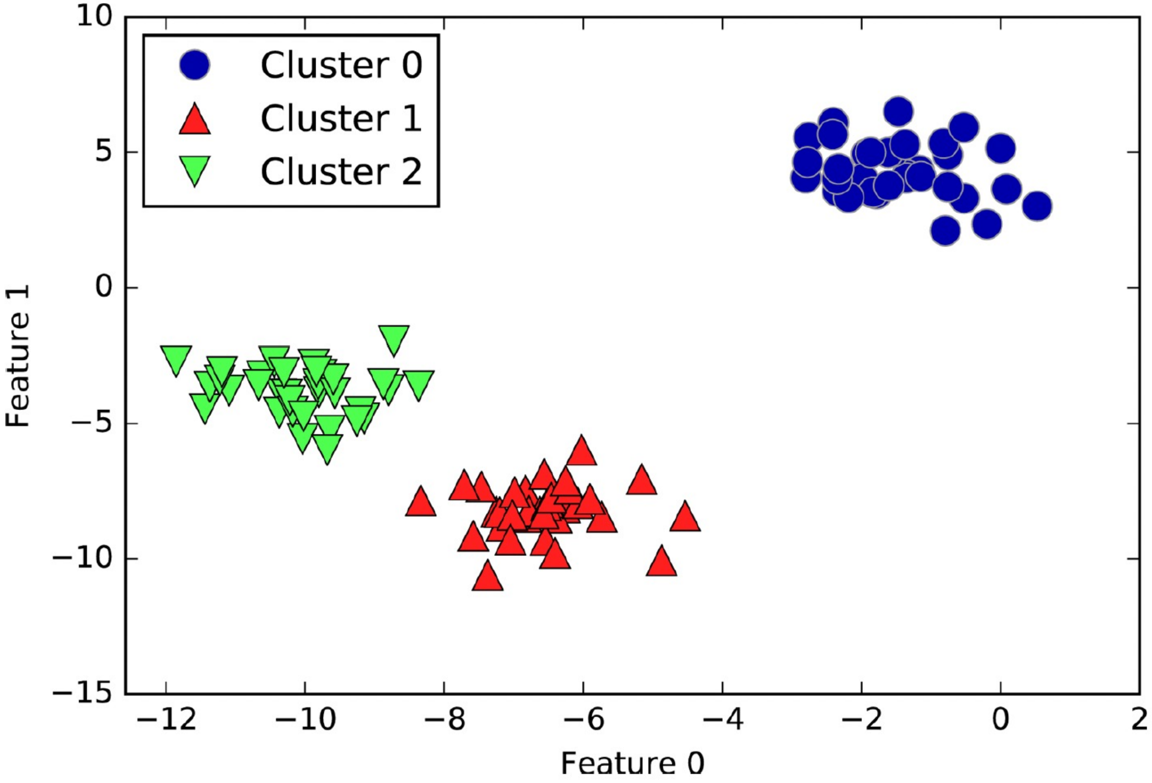


*Figure 3-34. Cluster assignment using agglomerative clustering with three clusters*

Muller and Guido book page 186

# Dendrogram of the clustering

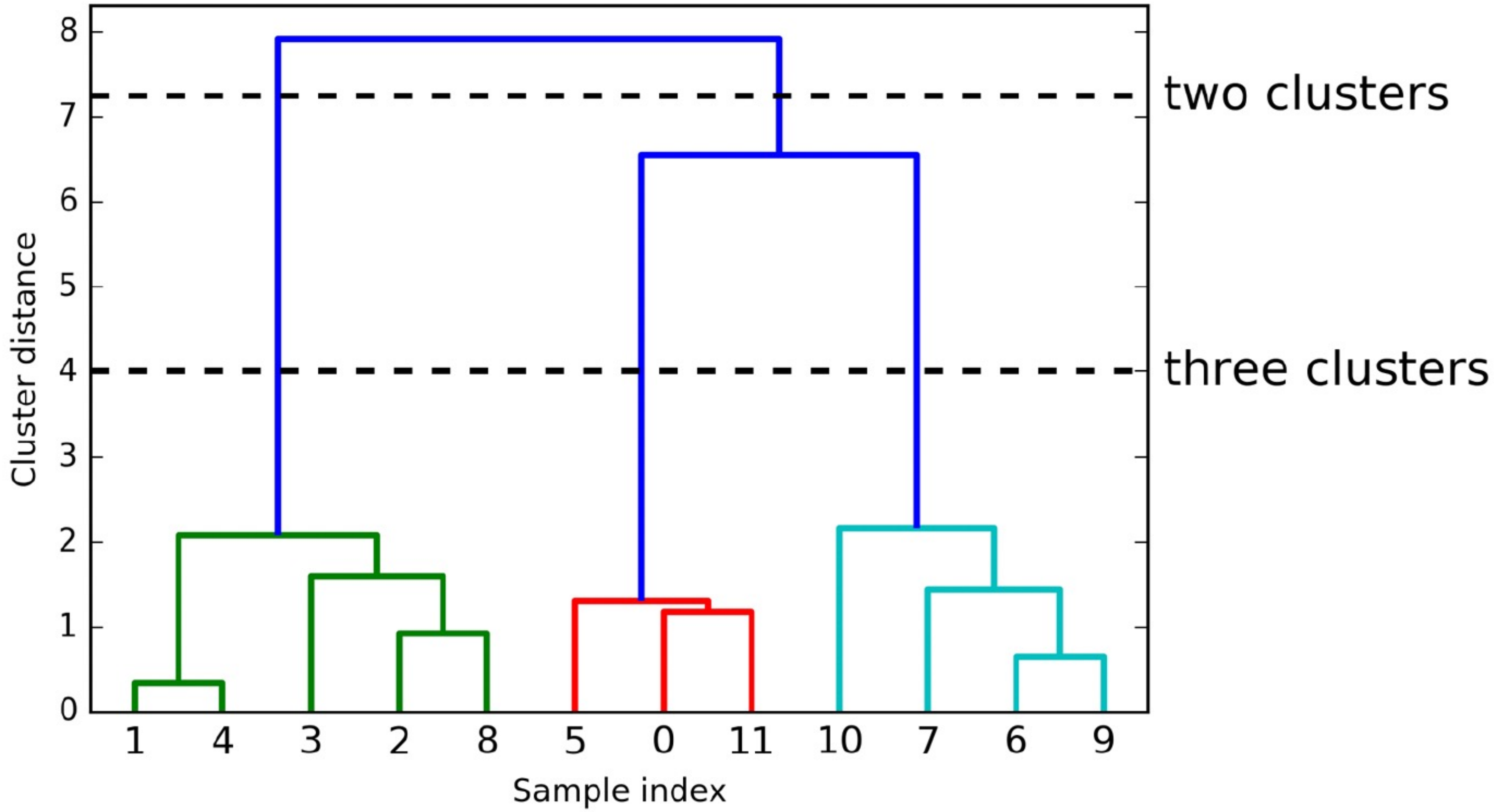


*Figure 3-36. Dendrogram of the clustering shown in Figure 3-35 with lines indicating splits into two and three clusters*

Muller and Guido book page 188

# Working with text data

Sentiment analysis

Bag-of-word

n-gram

# Features representing properties of the data

- Continuous  features that describe a quantity,
- Categorical features that are items from a fixed list
- Text/String data

# Example applications

- Classify an email message as either a legitimate email or spam
- The opinion of a politician on the topic of immigration (individual’s speeches or tweets)
- In customer service, if a message is a complaint or an inquiry (the subject line and content of a message)

# Four kinds of string data

- Categorical data
  - Your favourite colour from a drop-down menu
- Free strings that can be semantically mapped to categories
  - Type your favourite colour
- Structured string data
  - addresses, names of places or people, dates, telephone numbers
- Text data
  - Phrases, sentences such as tweets, chat logs, and hotel reviews, etc.

# Corpus and document

In the context of text analysis:

- The dataset is called the **corpus**,
- Each data point, represented as a single text, is called a **document**
- From the information retrieval (IR) and natural language processing (NLP) community

# Sentiment Analysis of Movie Reviews

- A dataset of movie reviews
- The IMDb (Internet Movie Database) website collected by Stanford researcher.
- This dataset contains the text of the reviews, and a “positive” or “negative” label.
- The IMDb website itself contains ratings from 1 to 10.
- Reviews with a score of 7 or higher labelled as positive
- Reviews with a score 4 or lower is labelled as negative
- Neutral reviews are not included in the dataset

http://ai.stanford.edu/~amaas/data/sentiment/

# An example of a text data (movie review)

"This movie has a special way of telling the story, at first i found it rather odd as it jumped through time and I had no idea what’s happening.<br /><br />Anyway the story line was although simple, but still very real and touching. You met someone the first time, you fell in love completely, but broke up at last and promoted a deadly agony. Who hasn't go through this? but we will never forget this kind of pain in our life. <br /><br />I would say i am rather touched as two actor has shown great performance in showing the love between the characters. I just wish that the story could be a happy ending.”

# Sentiment analysis task

- Given a review, assign the label “positive” or “negative” based on the text content of the review.

- The text data is not in a format that a machine learning model can handle
- Hence, convert the string representation of the text into a numeric representation that we can apply our machine learning algorithms to

# Bag-of-words

- One of the most simple but effective and commonly used ways to represent text for machine learning is:
- bag-of-words representation

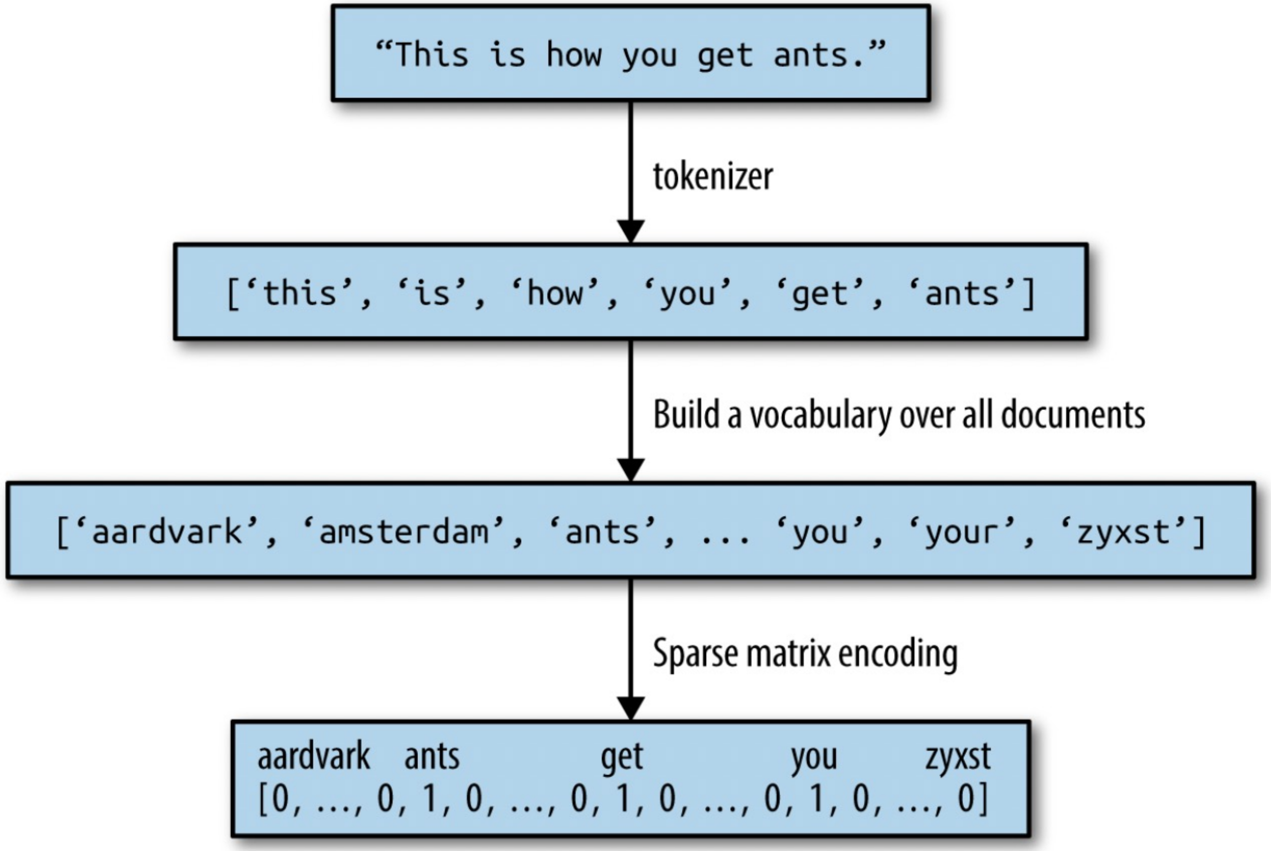
- Bag of words: “how often each word appears in each text”

# Bag-of-words processing

1. Tokenisation.
   - Split each document into the words that appear in it (called tokens), for example by splitting them on whitespace and punctuation.

- 2. Vocabulary building.
  - Collect a vocabulary of all words that appear in any of the documents, and number them (say, in alphabetical order).
- 3. Encoding
  - For each document, count how often each of the words in the vocabulary appear in this document.

# Bag-of-words processing



“This is how you get ants.”

tokenizer

[‘this’, ‘is’, ‘how’, ‘you’, ‘get’, ‘ants’]

Build a vocabulary over all documents

[‘aardvark’, ‘amsterdam’, ‘ants’, ... ‘you’, ‘your’, ‘zyxst’]

Sparse matrix encoding

aardvark ants get you zyxst
[0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0]

*Figure 7-1. Bag-of-words processing*

Muller and Guido book page 335

# Sparse matrix

- The bag-of-words representation is stored in a SciPy sparse matrix that only stores the entries that are nonzero

# Bag-of-words feature extraction

- Each word corresponds to a feature

- Example: movie review dataset
  - Size of the training data: 25,000 x 74,849, indicating that the vocabulary contains 74,849 entries

# Classification

- For high-dimensional, sparse data like this, linear models like Logistic Regression often work best.

# Improvements (performance or processing time)

- Numbers are not words
    - 007 is a word in the movie context
- Soon, soOn, sOOn are all same token/word

# Improvements, set min_df

- Only use tokens that appear in at least two documents (or at least five documents or do on).
- A token that appears only in a single document is unlikely to appear in the test set and is therefore not helpful.
- **min_df** parameter: minimum number of documents a token needs to appear in with the
- For movie review dataset, we can bring down the number of features to 27,271 when min_df = 5

# Improvements, Stop-words

- Get rid of uninformative words is by discarding words that are too frequent to be informative
- How to choose stop-words:
  - use a language specific list of stop words (built-in list of English stopword)
  - discard words that appear too frequently

- Examples of English language stop words:
  ['above', 'elsewhere', 'into', 'well', 'rather', 'fifteen', 'had', 'enough', 'herein', 'should', 'third', 'although', 'more', 'this', 'none', 'seemed', 'nobody', 'seems', 'he', 'also', 'fill', 'anyone', 'anything', 'me', 'the', 'yet', 'go', 'seeming', 'front', 'beforehand', 'forty', 'i']

# Rescaling the Data with tf–idf

- Instead of dropping features that are deemed unimportant, another approach is to rescale features by how informative we expect them to be.
- One of the most common ways to do this is using tf-idf
- tf-idf is term frequency–inverse document frequency

# tf–idf

- The intuition of this method is to give high weight to any term that appears often in a particular document, but not in many documents in the corpus
- If a word appears often in a particular document, but not in very many documents, it is likely to be very descriptive of the content of that document

# Movie review dataset

- Features with lowest tfidf:

  ['poignant' 'disagree' 'instantly' 'importantly' 'lacked' 'occurred' 'currently' 'altogether' 'nearby' 'undoubtedly' 'directs' 'fond' 'stinker' 'avoided' 'emphasis' 'commented' 'disappoint' 'realizing' 'downhill' 'inane']

- Features with highest tfidf:

  ['coop' 'homer' 'dillinger' 'hackenstein' 'gadget' 'taker' 'macarthur' 'vargas' 'jesse' 'basket' 'dominick' 'the' 'victor' 'bridget' 'victoria' 'khouri' 'zizek' 'rob' 'timon' 'titanic']

Features with low tf–idf are those that either are

# tf–idf score

- The tf–idf score for word w in document d:

$$\mathrm{tfidf}(w, d) = \mathrm{tf} * \ \log\left(\frac{N+1}{N_w+1}\right) + 1$$

- where N is the number of documents in the training set, Nw is the number of documents in the training set that the word w appears in, and tf (the term frequency) is the number of times that the word w appears in the query document d (the document you want to transform or encode).

# Bag-of-Words with More Than One Word

- One of the main disadvantages of using a bag-of-words representation is that word order is completely discarded.

  “it’s bad, not good at all” “it’s good, not bad at all”

- have exactly the same representation, even though the meanings are inverted.

# n-gram

- Not only considering the counts of single tokens
- but also the counts of pairs or triplets of tokens that appear next to each other.
- Pairs of tokens are known as bigrams
- triplets of tokens are known as trigrams
- sequences of tokens are known as n-grams

# Example

- Vocabulary size: 13
- Vocabulary:

['be', 'but', 'doth', 'fool', 'he', 'himself', 'is', 'knows', 'man', 'the', 'think', 'to', 'wise’]

- Vocabulary size: 14
- Vocabulary:

['be fool', 'but the', 'doth think', 'fool doth', 'he is', 'himself to', 'is wise', 'knows himself', 'man knows', 'the fool', 'the wise', 'think he', 'to be', 'wise man']

# n-gram and feature space

- The number of bigrams could be the number of unigrams squared
- The number of trigrams could be the number of unigrams to the power of three, leading to very large feature spaces.

# Accuracy, n-gram and parameter C

- using bigrams increases performance quite a bit
- adding trigrams is not increasing accuracy that much

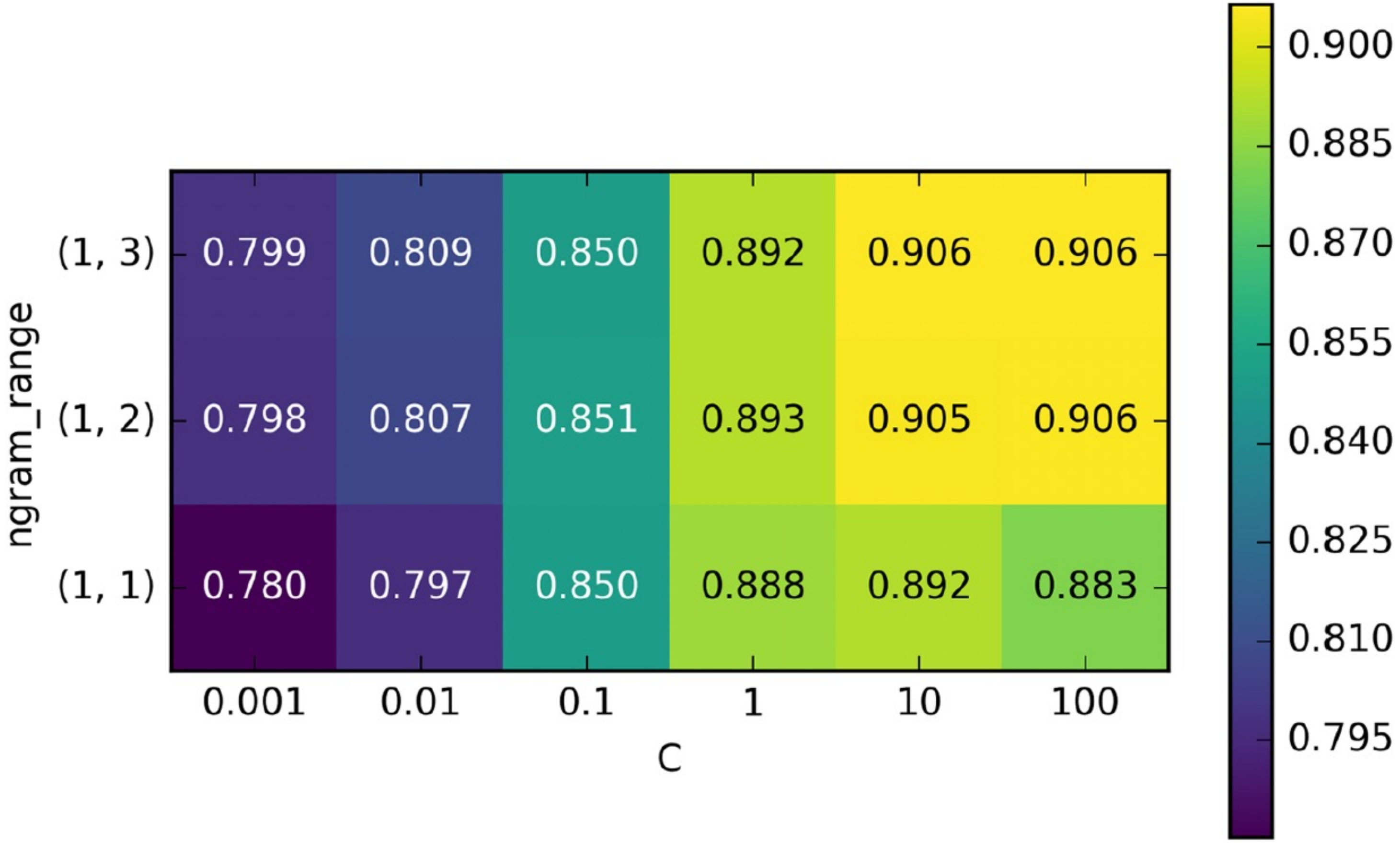


*Figure 7-3. Heat map visualization of mean cross-validation accuracy as a function of the parameters ngram_range and C*

Muller and Guido book page 349

# Coefficients’ magnitudes of the best model

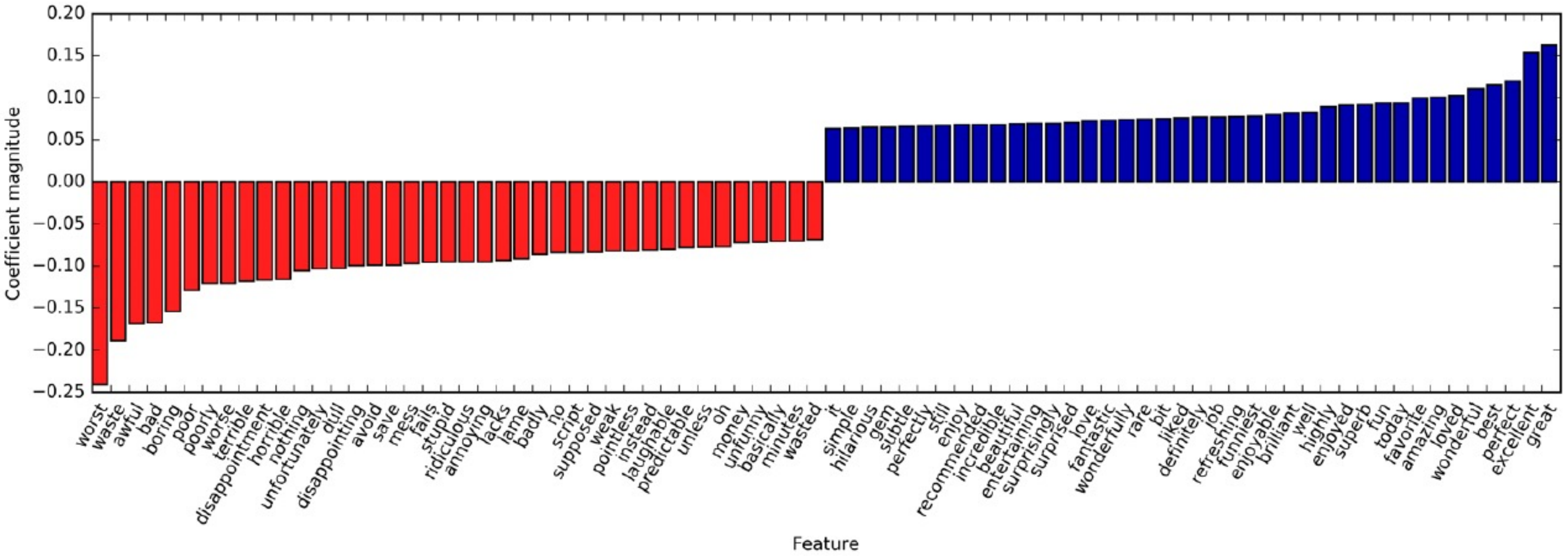


*Figure 7-2. Largest and smallest coefficients of logistic regression trained on tf-idf features*

Muller and Guido book page 350