WSTA Workshop

9-March-2018

Shivashankar Subramanian
shivashankar@student.unimelb.edu.au

Discussion I

- Give some examples of text processing applications that you use on a daily basis.

Text processing applications

There are lots!

Web search engines: Google, Bing

Speech-to-text: Siri, Cortana

Other commonly used applications: predictive messaging, spelling correction, machine translation, and so on.

Discussion II

What is **tokenisation** and why is it important?

Discussion II

What is **tokenisation** and why is it important?

- Segmenting text into linguistic units, such as words, punctuations and numbers
  - How many tokens are there
    - The wolf said: "Little pig, little pig, let me come in.“
    - You’re -> you + ‘ + re

- Important to normalize text. Also can give better features, say for classification.
  - Depends on the language (**Language identification)**

Discussion II

What are **stemming** and **lemmatisation**, and how are they different?

Discussion II

What are **stemming** and **lemmatisation**, and how are they different?

- Both stemming and lemmatisation are mechanisms for transforming a token into a canonical (base, normalised) form. For example, turning the token *walking* into its base form *walk.*

- Both operate by applying a series of rewrite operations to remove or replace (parts of) affixes (primarily suffixes) (In English).

- However, lemmatisation works in conjunction with a **lexicon** : a list of valid words in the language. The goal is to turn the input token into an element of this list (a valid word) using the rewrite rules. If the re-write rules can't be used to transform the token into a valid word, then the token is left alone.

- Stemming simply applies the rewrite rules, even if the output is a garbage token.

Inflectional Vs Derivational Morphology

Inflectional morphology is the systematic process by which tokens are altered to conform to certain grammatical constraints:

*E.g., English noun **teacher** --- plural form is **teachers**.*

The idea is that these changes don’t really alter the meaning of the term.

Consequently, both stemming and lemmatisation attempt to remove this kind of morphology.

Inflectional Vs Derivational Morphology

- **Derivational morphology** is the process by which we transform terms of one **class** into a different class.

- E.g., if we would like to make the English verb *teach* into a noun (someone who performs the action of *Teaching*), then it must be represented as *teacher*.

- This kind of morphology tends to produce terms that differ (perhaps subtly) in meaning, and the two separate forms are usually **both** listed in the lexicon.

- Consequently, lemmatisation doesn’t usually remove derivational morphology in its normalisation process, but stemming usually does.

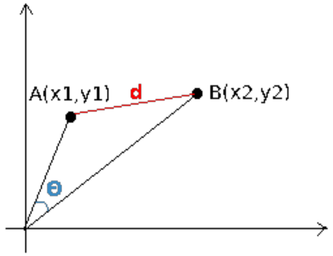
Discussion III

- What is **text-classification**? Give some examples
  - Classifying a piece of text into pre-defined set of categories (in general)
    - Sentiment analysis, author identification, fake news detection, etc

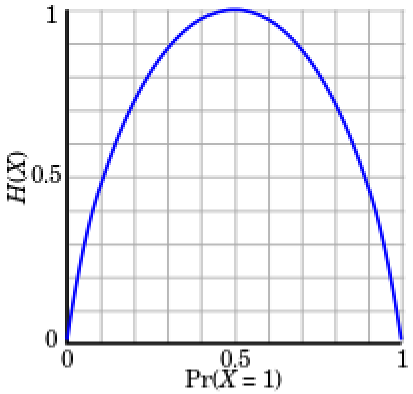
Discussion III

- Why is text classification generally a difficult problem? What are some hurdles that need to be overcome?
  - Representation --- bag-of-words, bag-of-ngrams
  - Sparsity (feature selection, transformation)

Discussion III



- Supervised text classification problem:
  - k-Nearest Neighbour using Euclidean distance
  - k-Nearest Neighbour using Cosine similarity
    - Curse of dimensionality
    - Hyper parameters – k, distance metric

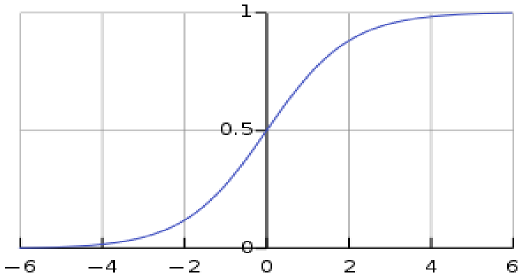


  - Decision Trees using Information Gain
    - Interpretable
    - Information gain = entropy(parent) – [average entropy(children)]
    - Not suitable for large dimensions (spurious correlations)
    - Information Gain is a poor choice because it tends to prefer **rare** features; in this case, this would correspond to features that appear only in a handful of documents

Discussion III

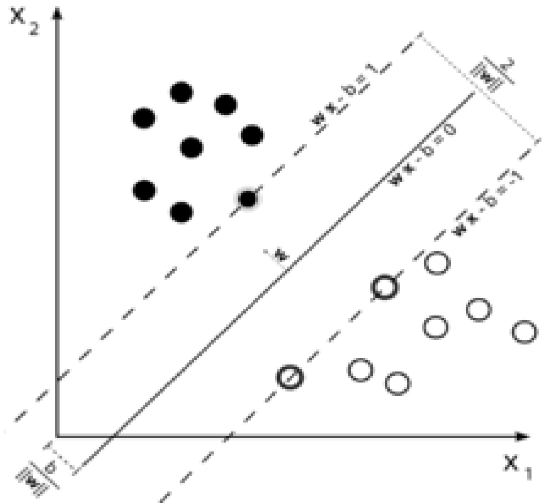
Naive Bayes

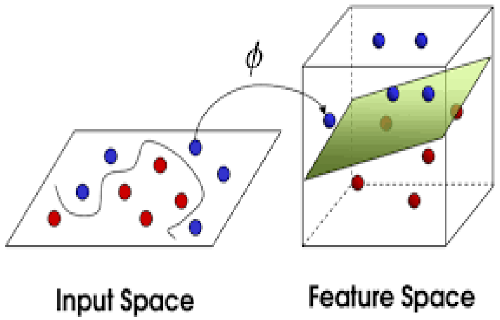
- Simple and fast
- Independence assumption
- Not suitable for large dimensions



Logistic Regression

- linear





Support Vector Machines

- Linear/non-linear kernel
- Multi-class