

Natural Language Processing

Let's learn something!

Python and Spark

- Let's now learn about the basics of Natural Language Processing!
- This is the field of machine learning that focuses on creating models from a text data source (straight from articles of words).

Python and Spark

- This is a very large field of machine learning with its own unique challenges and sets of algorithms and features, so what we cover here will be scratching just the surface!

Python and Spark

- Optional Reading Suggestions:
 - Wikipedia Article on NLP
 - NLTK Book (separate Python library)
 - Foundations of Statistical Natural Language Processing (Manning)

Python and Spark

- Examples of NLP
 - Clustering News Articles
 - Suggesting similar books
 - Grouping Legal Documents
 - Analyzing Consumer Feedback
 - Spam Email Detection

Python and Spark

- Our basic process for NLP:
 - Compile all documents (Corpus)
 - Featurize the words to numerics
 - Compare features of documents

Python and Spark

- A standard way of doing this is through the use of what is known as “TF-IDF” methods.
- TF-IDF stands for Term Frequency - Inverse Document Frequency
- Let’s explain how it works!

NLP

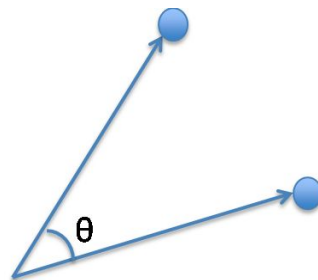
Simple Example:

- You have 2 documents:
 - “Blue House”
 - “Red House”
- Featurize based on word count:
 - “Blue House” -> (red,blue,house) -> (0,1,1)
 - “Red House” -> (red,blue,house) -> (1,0,1)

NLP

- A document represented as a vector of word counts is called a “Bag of Words”
 - “Blue House” -> (red,blue,house) -> (0,1,1)
 - “Red House” -> (red,blue,house) -> (1,0,1)
- These are now vectors in an N-dimensional space, we can compare vectors with cosine similarity:

$$\text{sim}(A, B) = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|}$$



NLP

- We can improve on Bag of Words by adjusting word counts based on their frequency in corpus (the group of all the documents)
- We can use TF-IDF (Term Frequency - Inverse Document Frequency)

NLP

- Term Frequency - Importance of the term within that document
 - $TF(\mathbf{x}, \mathbf{y})$ = Number of occurrences of term \mathbf{x} in document \mathbf{y}
- Inverse Document Frequency - Importance of the term in the corpus
 - $IDF(\mathbf{t}) = \log(\mathbf{N}/\mathbf{dfx})$ where
 - \mathbf{N} = total number of documents
 - \mathbf{dfx} = number of documents with the term

NLP

- Mathematically, TF-IDF is then expressed:

$$w_{x,y} = \text{tf}_{x,y} \times \log \left(\frac{N}{\text{df}_x} \right)$$

TF-IDF

Term x within document y

$\text{tf}_{x,y}$ = frequency of x in y

df_x = number of documents containing x

N = total number of documents

Python and Spark

- Spark has a lot of `pyspark.ml.feature` tools to help out with this entire process and make it all easy for you!