SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

Hi guys, welcome to the Natural Language Processing lecture!

This is another very important area of Machine Learning and Artificial Intelligence. In this lecture, we will discuss how to manipulate and analyze the language data. The concept behind grouping the articles, legal documents, news etc, based on their relevance. We will also learn how to store the language data in standard format and much more…..!

✅ *Optional Readings and References:*

There are several books and lots of material available on NLP on web for free. You can always google it.
If you are working with NLP in Python, [Natural Language Processing with Python by Steven Bird, Ewan Klein and Edward Loper] is a very good read. It is free on the provided link!
The documentation on official website is always a great resource as well http://www.nltk.org.

Dr. Junaid S. Qazi PhD

Natural Language Processing (NLP)

Suppose:

- you are working with one of the biggest research publication organization (e.g. Springer, ScienceDirect). They want you to group the research articles by the area of research!
- you are a employed by a leading news agency (e.g. BBC, CNN) and your task is to group the news by their headlines or topics!
- You are part of a big legal firm (e.g. Jones Day, Gibson Dunn), where you need to find out the relevant documents from thousands of pages!

Natural Language Processing (NLP)

Suppose:

- you are working with one of the biggest research publication organization (e.g. Springer, ScienceDirect). They want you to group the research articles by the area of research!
- you are a employed by a leading news agency (e.g. BBC, CNN) and your task is to group the news by their headlines or topics!
- You are part of a big legal firm (e.g. Jones Day, Gibson Dunn), where you need to find out the relevant documents from thousands of pages!

These and many other similar tasks are not trivial for human. This is where Natural Language Processing (NLP) can help, when we are dealing with text data!

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

Suppose:

- you are working with one of the biggest research publication organization (e.g. Springer, ScienceDirect). They want you to group the research articles by the area of research!
- you are a employed by a leading news agency (e.g. BBC, CNN) and your task is to group the news by their headlines or topics!
- You are part of a big legal firm (e.g. Jones Day, Gibson Dunn), where you need to find out the relevant documents from thousands of pages!

These and many other similar tasks are not trivial for human. This is where Natural Language Processing (NLP) can help, when we are dealing with text data!

By definition, **Natural-language processing (NLP)** is an area of computer science and artificial intelligence, concerned with the interactions between computers and human (natural) languages, in particular how to program computers to fruitfully process large amounts of natural language data.

Natural Language Processing (NLP)

For the mentioned tasks, we want to do the following:

- compile the documents is some fashion
- get feature from the documents
- compare their features for similarity

Natural Language Processing (NLP)

For the mentioned tasks, we want to do the following:

- compile the documents is some fashion
- get feature from the documents
- compare their features for similarity

In a very simple example, where we have 2 documents with the following text:

- Document 1 with a single sentence: “red tag”
- Document 2 with a single sentence: “green tag”

Natural Language Processing (NLP)

For the mentioned tasks, we want to do the following:

- compile the documents is some fashion
- get feature from the documents
- compare their features for similarity

In a very simple example, where we have 2 documents with the following text:

- Document 1 with a single sentence: “red tag”
- Document 2 with a single sentence: “green tag”

A simple way to featurize the text document is to do the word count. We can transform “text” into a vectorized word counts. In order to do this, we basically create a vector count of all the possible words in all the documents. We then count how many times those words appear in each document. In the above, we have three words, “red”, “green” and “tag”.

Natural Language Processing (NLP)

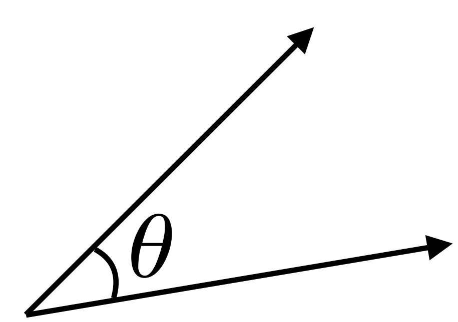
In our example, the featurization of each document, based on the word count, will be:

- "red tag" —> (red, green, tag) —> (1,0,1)
- "green tag" —> (red, green, tag) —> (0,1,1)

(red, green, tag) —> (1,0,1) means, red appeared one time, green 0 time and tag appeared for 1 time and so on!

A document represented as a vector of word counts is called "**Bag of Words**". Treating each document as a vector of features is useful because, once, we have the bag of words vectors, we can perform mathematical operations on them. For example, we can compute cosine similarity using the equation below. We can also compute other similarity metrics in order to figure out how similar two text documents are to each other.

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



SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

We can improve the Bag of Words (BoW) by adjusting word counts, based on their frequency in corpus (a collection of written text or a group of all the documents). We can use ***TF-IDF*** (Term Frequency - Inverse Document Frequency) in order to do this.

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

We can improve the Bag of Words (BoW) by adjusting word counts, based on their frequency in corpus (a collection of written text or a group of all the documents). We can use ***TF-IDF*** (Term Frequency - Inverse Document Frequency) in order to do this.

- **Term Frequency *(TF)*** measures how frequently a term occurs in a document. *TF(t, d)* depends upon the number of occurrences of term *"t"* in the document *"d"*. This suggest how important the term is within that document.

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

We can improve the Bag of Words (BoW) by adjusting word counts, based on their frequency in corpus (a collection of written text or a group of all the documents). We can use ***TF-IDF*** (Term Frequency - Inverse Document Frequency) in order to do this.

- **Term Frequency *(TF)*** measures how frequently a term occurs in a document. *TF(t, d)* depends upon the number of occurrences of term *"t"* in the document *"d"*. This suggest how important the term is within that document.
- **Inverse Document Frequency *(IDF)*** measures the importance of the term in the corpus (group of all the documents).

$$IDF(t, D) = log\ \frac{D}{|\{d \in D : t \in d\}|}$$

"*D*" is the total number of documents

$|\{d \in D : t \in d\}|$ is the number of documents in *"D"* where the term *"t"* appeared

Natural Language Processing (NLP)

We can improve the Bag of Words (BoW) by adjusting word counts, based on their frequency in corpus (a collection of written text or a group of all the documents). We can use ***TF-IDF*** (Term Frequency - Inverse Document Frequency) in order to do this.

- **Term Frequency *(TF)*** measures how frequently a term occurs in a document. *TF(t, d)* depends upon the number of occurrences of term *"t"* in the document *"d"*. This suggest how important the term is within that document.

- **Inverse Document Frequency *(IDF)*** measures the importance of the term in the corpus (group of all the documents).

$$IDF(t, D) = log\ \frac{D}{|\{d \in D : t \in d\}|}$$

"D" is the total number of documents

$|\{d \in D : t \in d\}|$ is the number of documents in *"D"* where the term *"t"* appeared

IDF diminishes the weight of terms that occur very frequently in the corpus and increases the weight of terms that occur rarely.

Natural Language Processing (NLP)

Consider, we have two documents (d_1 & d_2) with some terms and their frequencies as given in the table, let's compute TF and IDF for "this" and "example".

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

Natural Language Processing (NLP)

Consider, we have two documents (d_1 & d_2) with some terms and their frequencies as given in the table, let's compute TF and IDF for "this" and "example".

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

TF("this", d_1) = freq. of "this" / total no. of words in d_1 = 1 / 5 = **0.2**
TF("this", d_2) = freq. of "this" / total no. of words in d_2 = 1 / 7 ~ **0.14**
d_2 has more words, the relative frequency of "this" is smaller

Natural Language Processing (NLP)

Consider, we have two documents (d_1 & d_2) with some terms and their frequencies as given in the table, let's compute TF and IDF for "this" and "example".

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

TF("this", d_1) = freq. of "this" / total no. of words in d_1 = 1 / 5 = **0.2**

TF("this", d_2) = freq. of "this" / total no. of words in d_2 = 1 / 7 ~ **0.14**

d_2 has more words, the relative frequency of "this" is smaller

IDF is constant per corpus:

IDF ("this", D) = log(total no of docs. in corpus) / (no of docs with "this")

= log (2/2) = 0

Natural Language Processing (NLP)

Consider, we have two documents (d_1 & d_2) with some terms and their frequencies as given in the table, let's compute TF and IDF for "this" and "example".

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

TF("this", d_1) = freq. of "this" / total no. of words in d_1 = 1 / 5 = **0.2**
TF("this", d_2) = freq. of "this" / total no. of words in d_2 = 1 / 7 ~ **0.14**
d_2 has more words, the relative frequency of "this" is smaller

IDF is constant per corpus:
IDF ("this", D) = log(total no of docs. in corpus) / (no of docs with "this")
= log (2/2) = 0

TF-IDF ("this", d_1) = TF("this", d_1) x IDF ("this", D) = 0.2 x 0 = 0
TF-IDF ("this", d_2) = TF("this", d_2) x IDF ("this", D) = 0.14 x 0 = 0

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

Consider, we have two documents (d_1 & d_2) with some terms and their frequencies as given in the table, let's compute TF and IDF for "this" and "example".

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

TF("this", d_1) = freq. of "this" / total no. of words in d_1 = 1 / 5 = **0.2**
TF("this", d_2) = freq. of "this" / total no. of words in d_2 = 1 / 7 ~ **0.14**
d_2 has more words, the relative frequency of "this" is smaller

IDF is constant per corpus:
IDF ("this", D) = log(total no of docs. in corpus) / (no of docs with "this")
= log (2/2) = 0

TF-IDF ("this", d_1) = TF("this", d_1) x IDF ("this", D) = 0.2 x 0 = 0
TF-IDF ("this", d_2) = TF("this", d_2) x IDF ("this", D) = 0.14 x 0 = 0

TF-IDF is zero for the word "this", which implies that the word is not very informative as it appears in all documents.

Natural Language Processing (NLP)

For the word “example”, the situation is interesting, it appeared three times but only in d_2 !

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

For the word "example", the situation is interesting, it appeared three times but only in d_2 !

TF("example", d_1) = 0 / 5 = **0**
TF("example", d_2) = 3 / 7 ~ **0.429**

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

Natural Language Processing (NLP)

For the word "example", the situation is interesting, it appeared three times but only in d_2 !

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

TF("example", d_1) = 0 / 5 = **0**
TF("example", d_2) = 3 / 7 ~ **0.429**

IDF is constant per corpus:
IDF ("example", D) = log (2/1) = 0.301 *(using base 10 logarithm)*

Natural Language Processing (NLP)

For the word “example”, the situation is interesting, it appeared three times but only in d_2 !

TF(“example”, d_1) = 0 / 5 = **0**
TF(“example”, d_2) = 3 / 7 ~ **0.429**

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

IDF is constant per corpus:
IDF (“example”, D) = log (2/1) = 0.301 *(using base 10 logarithm)*

TF-IDF (“example”, d_1) = TF(“example”, d_1) x IDF (“example”, D
TF-IDF (“example”, d_1) = 0 x 0.301 = 0

TF-IDF (“example”, d_2) = TF(“example”, d_2) x IDF (“example”, D)
TF-IDF (“example”, d_2) = 0.429 x 0.301 ~ 0.13

SCIENCE ACADEMY
Excellence in Teaching & Learning

Natural Language Processing (NLP)

For the word "example", the situation is interesting, it appeared three times but only in d_2 !

TF("example", d_1) = 0 / 5 = **0**
TF("example", d_2) = 3 / 7 ~ **0.429**

d_1	
Term	**Term Count**
this	1
is	1
a	2
sample	1

d_2	
Term	**Term Count**
this	1
is	1
another	2
example	3

IDF is constant per corpus:
IDF ("example", D) = log (2/1) = 0.301 *(using base 10 logarithm)*

TF-IDF ("example", d_1) = TF("example", d_1) x IDF ("example", D
TF-IDF ("example", d_1) = 0 x 0.301 = 0

TF-IDF ("example", d_2) = TF("example", d_2) x IDF ("example", D)
TF-IDF ("example", d_2) = 0.429 x 0.301 ~ 0.13

Hence, "example" is informative in the corpus and, in this case, more relevant to the document d_2.

We have gone through the theory behind Natural Language Processing (NPL). Let's move on and learn with a practical example using Python.

In order to do the, we need to install and additional library in Python which deals with NLP. Go to the terminal or command line install nltk:

- **conda install nltk**

  or
- **pip install nltk**