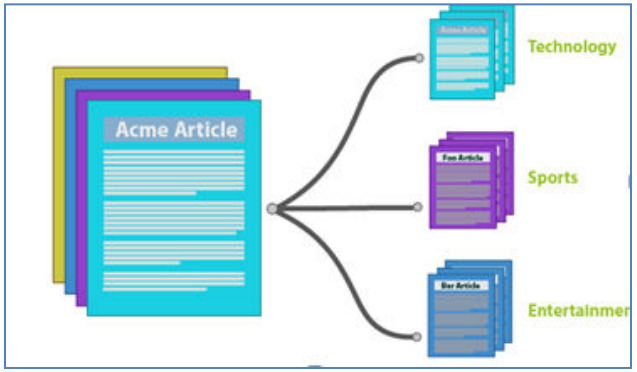
NLP – Text Classification

Text Classification : is the task of assigning predefined categories to free-text documents.

- **Email Spam Identification**
- **Topic classification of news**
- **Sentiment classification**
- **Organization of web pages by search engines**



Ex.

: News-stories are typically organized by subject categories (*topics*) or geographical codes

: Academic papers are often classified by technical domains and sub-domains

: Patient reports in health-care organizations are often indexed from multiple aspects, using taxonomies of disease categories types of surgical procedures, insurance reimbursement codes and so on.

Spam detection Use Case

Business Objective: Create an Intelligent System to detect **SPAM messages** and filter them out to protect the system/mailbox/Inbox etc.

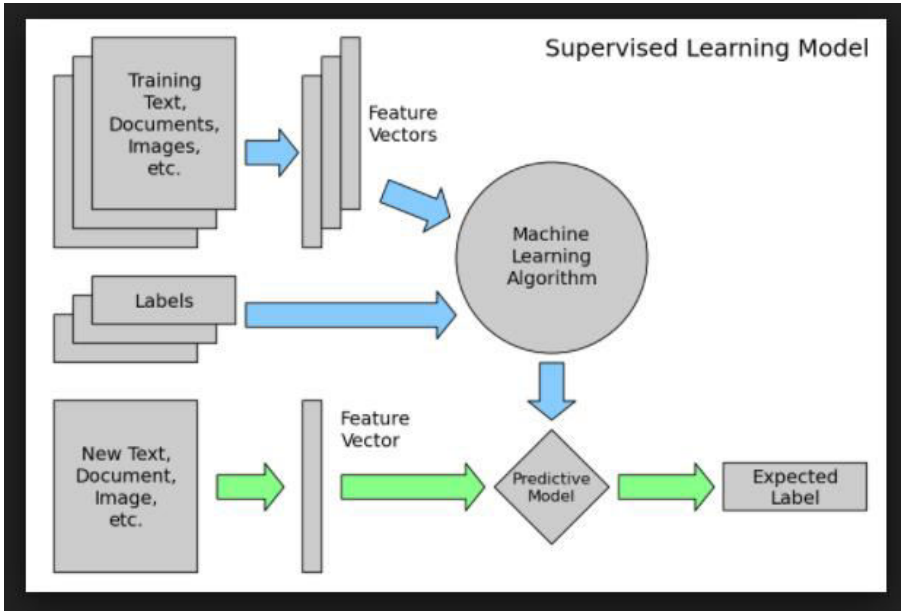
SPAM : Undesirable messages (an advertisement, viruses message etc)

"According to the statistics from **ITU** (International Telecommunication Union), **70% to 80%** of emails in the internet are spams which have become **worldly problem** to the **information infrastructure** ".

Text	Spam_label
Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 0845281007	spam
URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403L	spam
Congrats! 1 year special cinema pass for 2 is yours. call 09061209465 now! C Suprman V, Matrix3, StarWars3, etc all 4 FREE! bx420-ip4-5we. 150pm. ...	spam
Even my brother is not like to speak with me. They treat me like aids patent.	non-spam
I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.	non-spam
Finished class where are you. ? Are you free today to go out ?	non-spam
I call you later, don't have network. If urgnt, sms me	non-spam

Spam detection using Machine Learning

Text	Spam_label
Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 0845281007	spam
URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403L	spam
Congrats! 1 year special cinema pass for 2 is yours. call 09061209465 now! C Suprman V, Matrix3, StarWars3, etc all 4 FREE! bx420-ip4-5we. 150pm. ...	spam
Even my brother is not like to speak with me. They treat me like aids patent.	non-spam
I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.	non-spam
Finished class where are you. ? Are you free today to go out ?	non-spam
I call you later, don't have network. If urgnt, sms me	non-spam



Data Cleaning

Text Features

EDA

Model

Prediction

Text data Pre-processing & Cleaning

Text Pre-processing and Cleaning:

> Text Document : A **text document** is a kind of computer file that is structured as a **sequence of lines** of **electronic text.**

Text	Spam_label
Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 0845281007	spam
URGENT! You have won a 1 week FREE membership in our å£100,000 Prize Jackpot! Txt the word: CLAIM to No: 81010 T&C www.dbuk.net LCCLTD POBOX 4403L	spam
Congrats! 1 year special cinema pass for 2 is yours. call 09061209465 now! C Suprman V, Matrix3, StarWars3, etc all 4 FREE! bx420-ip4-5we. 150pm. ...	spam
Even my brother is not like to speak with me. They treat me like aids patent.	non-spam
I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.	non-spam
Finished class where are you. ? Are you free today to go out ?	non-spam
I call you later, don't have network. If urgnt, sms me	non-spam

- **Splitting the whole document into smaller chunks of sentences & words**

  Sentence tokenization - Splitting the paragraph into sentences .

  Word tokenization - Splitting the sentence into words.

```
In [4]: # tokenize sentence into words
        text = "This is a very simple text data for tokenizatoin example "
        word_tokenize(text)

Out[4]: ['This',
         'is',
         'a',
         'very',
         'simple',
         'text',
         'data',
         'for',
         'tokenizatoin',
```

Stop Words Removal

➢ Stop Words Removal (Noise Entity)

Noise : Any text that doesn't add relevance value to context of the text data is known as Noise:
Example : is, am, are, was, were, of , that , in etc, social media hastags, punctuations.

```
import nltk
from nltk.corpus import stopwords
stop_words = stopwords.words('english')
# For english, 179 stop word list given in NLTK
```

```
stop_words
```

```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

```
text_1 = "This is a very simple text data for tokenization example"
[word for word in text_1.split(" ") if word not in stop_words]
```

```
['This', 'simple', 'text', 'data', 'tokenization', 'example']
```

Customized Stop Words List

- **Customized Stop words list :**

- Some words are very common in each domain and those words present in almost all the document. Ex **'patient', 'dr.', 'mcg'** are very common in **clinical text**, so these could be treated as potential stop words

```
custom_stop_words = set(stop_words)
```

```
# added three more stop words in the given list for Clinical Text domain
custom_stop_words.update(['patient', 'dr', 'mcg'])
print(len(custom_stop_words)) # now total stop words list = 182
custom_stop_words
```

```
 "don't",
 'down',
 'dr',
 'during',
 'each',
 'few',
 'for',
 'from',
```

- Most frequent terms can be treated as distracting terms(stop words)

- least frequent terms can be treated as distracting terms(stop words)

Stemming & Lemmatization

Stemming - is a technique to **remove the affixes** from the word.

In **Linguistic morphology** and **information retrieval**, stemming is the process of **reducing a derived(inflected)** words to their **base form.**

Lemmatizing - Lemma is a **root word** and similar to stemming.

Lemmatizer looks **meaning of the word** while **stemmer looks form of the word**.

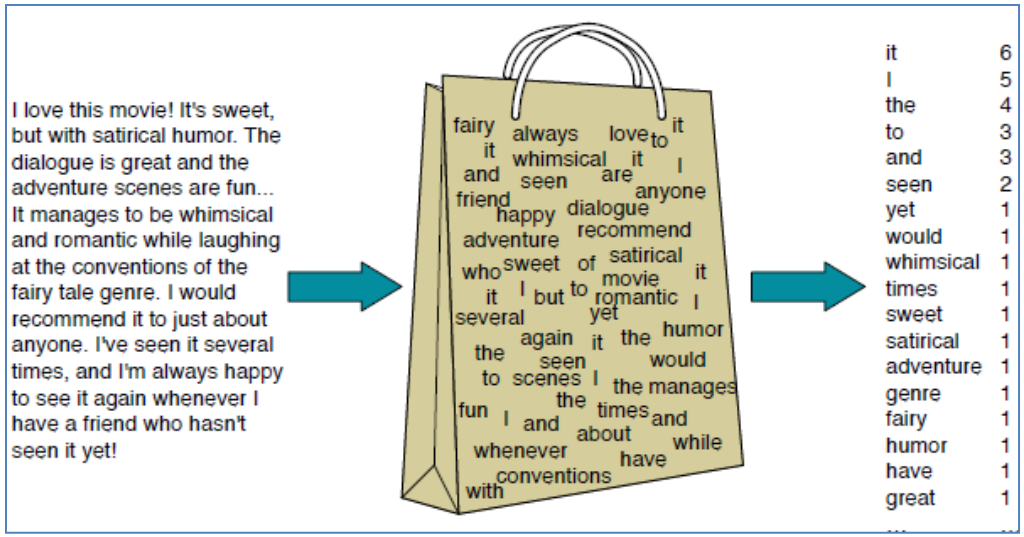
Example : lemmatizer will transform the word **'believes'** to it's root word **'belief'**
stemmer will transform the word **'believes'** to it's base form **'believe'**

```
# Lemmatization & stemming
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
print('Lemmatization : ', lemmatizer.lemmatize('believes'))
print('stemming : ', Lanc_stemmer.stem('believes'))
```

```
Lemmatization :  belief
stemming :  believ
```

Text Features Engineering

- **Bag of words**
- **Bi-Gram, Tri-Gram , N-Gram**
- **TF-IDF**
- **LDA, LSI Low Dimensional Topics generation**
- **Word Text Vectors**
- **POS, NER**



Bag of Word – Vectorizer

- **Bag of Word Vectorizer (TF) :** Convert a collection of text documents to a matrix of token counts

- sklearn.feature_extraction.text.CountVectorizer

- vectorizer.fit_transform(...) creates the vocabulary and returns a term-document matrix

- each column in the matrix represents a word in the vocabulary while each row represents the document in our dataset where the values in this case are the word counts

```
vectorizer=CountVectorizer(max_df=.85, stop_words=stopwords, max_features=1000,token_pattern='[a-zA-Z\-][a-zA-Z\-]{2,}' )
x = vectorizer.fit_transform(temp_data_old)
```

	claim	comp	cup	dbuk	entry	final	jackpot	lccltd	may	membership	net	pobox	prize	question	rate	receive	std	text	tkts	urgent	week	win
0	0	1	1	0	2	1	0	0	1	0	0	0	0	1	1	1	1	1	1	0	0	1
1	1	0	0	1	0	0	1	1	0	1	1	1	1	0	0	0	0	0	0	1	1	0

TF-IDF (Statistical Features)

TF-IDF stands for "**Term Frequency**, **Inverse Document Frequency**."

- It's a way to score the **importance of words** (or "terms") in a document based on **how frequently** they appear across multiple documents.
- If a word appears **frequently** in a document, it's important, give the word a **high score**.
- But if a word appears in **many documents**, it's **not a unique identifier**, give the word a **low score**.
- Therefore, **common words** like "the" and "for," which appear in many documents, will be **scaled down**. Words that appear frequently in a *single* document will be scaled up.

TF IDF formula gives the **relative importance** of a **term** in a corpus (list of documents) and **convert** the **text documents** into **vector models** on the basis of occurrence of words in the documents.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

tf_{ij} = number of occurrences of i in j
df_i = number of documents containing i
N = total number of documents

- Documents:
  - d1: data mining and social media mining
  - d2: social network analysis
  - d3: data mining
- tf-idf representation:

	analysis	data	media	mining	network	social
df(w)	1	2	1	2	1	2
log(N/df(w))	0.48	0.18	0.48	0.18	0.48	0.18
d1, tf	0	1	1	2	0	1
d2, tf	1	0	0	0	1	1
d3, tf	0	1	0	1	0	0
d1, tf-idf	0.00	0.18	0.48	0.35	0.00	0.18
d2, tf-idf	0.48	0.00	0.00	0.00	0.48	0.18
d3, tf-idf	0.00	0.18	0.00	0.18	0.00	0.00

BOW & TF-IDF

Bow : tf(t,d)

- **Raw** word counts
- **bow(w, d**) = # times word w appears in document d

TF-IDF: tf(t,d) * idf(t)

- **Normalized** word counts
- tf-idf(w, d) = bow(w, d) * log (N / # documents in which word w appears)
- if a word appear in every single document, will effectively zeroed out (log1=0) and word appears in few document will have larger count than before. (ex – mining word)
- makes **rare words more prominent** and effectively **ignores common words**
- this feature scaling can improve the linear classification model logreg

- Documents:
  - d1: data mining and social media mining
  - d2: social network analysis
  - d3: data mining
- tf-idf representation:

	analysis	data	media	mining	network	social
df(w)	1	2	1	2	1	2
log(N/df(w))	0.48	0.18	0.48	0.18	0.48	0.18
d1, tf	0	1	1	2	0	1
d2, tf	1	0	0	0	1	1
d3, tf	0	1	0	1	0	0
d1, tf-idf	0.00	0.18	0.48	0.35	0.00	0.18
d2, tf-idf	0.48	0.00	0.00	0.00	0.48	0.18
d3, tf-idf	0.00	0.18	0.00	0.18	0.00	0.00

Chi-squared Test of Independence

Two random variables x and y are called **independent** if the probability distribution of one variable is not affected by the presence of another.

Assume f_{ij} is the observed frequency count of events belonging to both i-th category of x and j-th category of y. Also assume e_{ij} to be the corresponding expected count if x and y are independent. The null hypothesis of the independence assumption is to be rejected if the p-value of the following Chi-squared test statistics is less than a given significance level α.

$$\chi^2 = \sum_{i,\ j} \frac{(f_{ij} - e_{ij})^2}{e_{ij}}$$

$$X^2 = \sum \frac{(o-e)^2}{e}$$

where

X^2 is Chi-squared,
$\sum$ stands for summation,
o is the observed values, and
e is the expected values.

Text Classification work Flow

Data Cleaning | Text Features | EDA | Model | Prediction