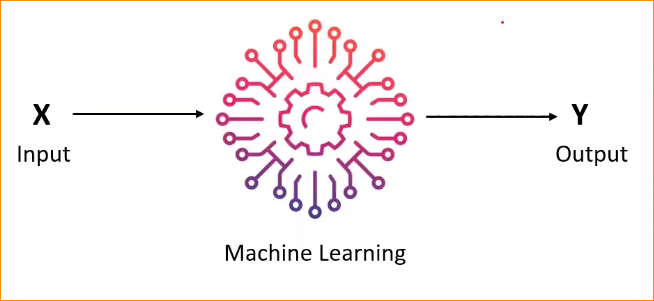
Feature Extraction

1. Machine learning

We give input to the model and model give the output.

Entire process devide 2 steps – 1. Training 2. Testing

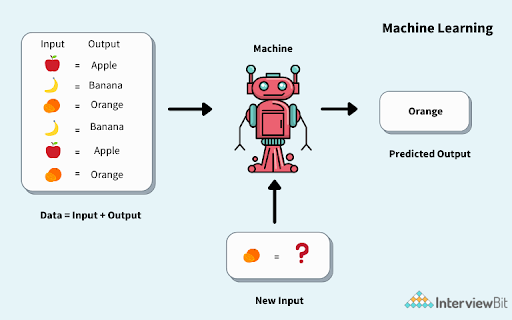


1. Training

With data data set, Data set has both X and Y.

That mean We have input and as well as the corresponding out put. So we refer to that as the data set.

So that bunch of data we send to the model. So we give the relationship to the model and model learn the relationship between X and Y ().



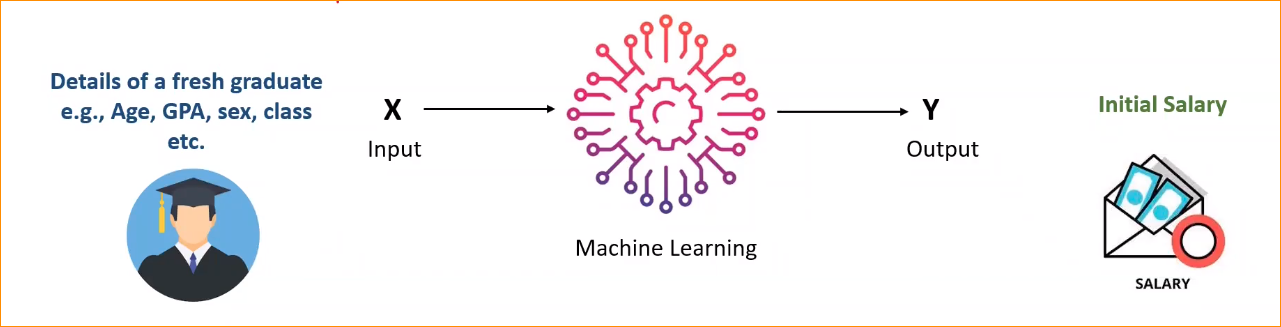
1. Testing or Application process

Applied the trained model in to new environment.

If Give the new Input . X-

Using the relationship that model Learnt , It will try to predict the Y

1. Example of Machine learning problems



Input – giving the detail about fresh graduate (Details – age,GPA,Sex,Class etc. )

Output – Predict the initial salary

For this we are using structured data(can put tom the table)

**In here Example- (Structured data)**

Column -

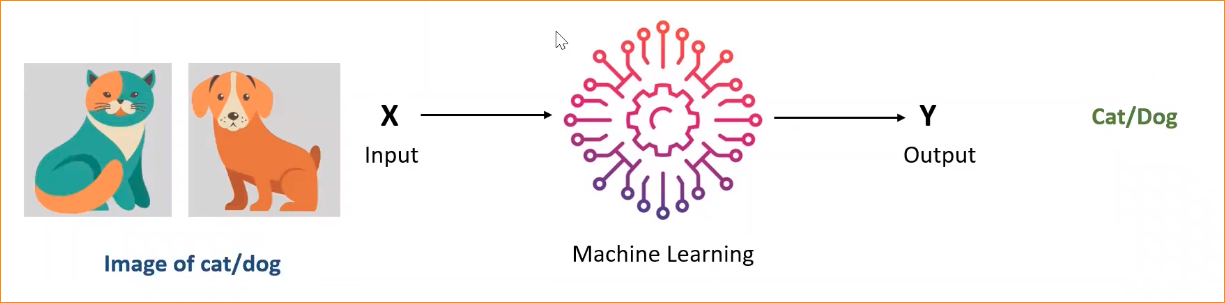
Student ID, Age , Sex ,GPA, Class ()

Rows – Individual Student

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Name** | **Age** | **Sex** | **GPA** | **Class** |
| 001 | Kavinda | 29 | Male | 4.00 | 1 |
| 002 | Shakuni | 29 | Female | 3.50 | 1 |
| 003 | Shantha | 28 | Male | 3.00 | 2 |
| 004 | Ramani | 25 | Female | 2.87 | 3 |

**Example 02 – Given image Cat or dog and Classified as Cat or Dog (image)**

Indicate diction of the model use Class



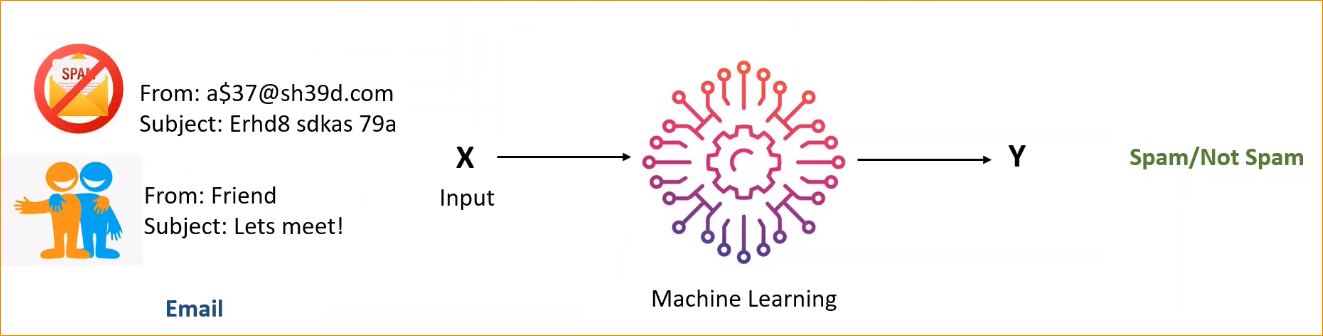
Class

Input – image of the Cat or Dog

Output – indicating the Cat or Dog

**Example 03 – NLP Spam classification (Text)**

How to differentiate Good email and Spam email – Spam Classification



Input – Email Address and Email Topic

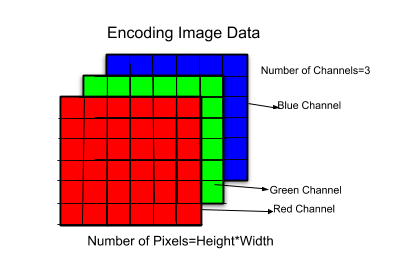
Output- Spam / Not Spam

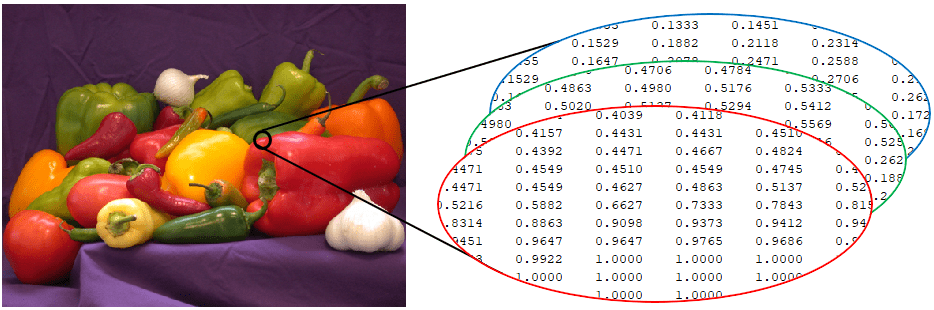
Consider this example - Change the Input - Structured data, Image , text

How to do that –

Machine only can understand numbers so all this types should convert into numbers.

Example image – image divided in to pixel location. Can get the RGB values , Like wise can represent image as A numbers. So that model can understand.





1. Features

* Features are Actual input in the system
* Process of generating the input or features that call features extraction (text, image etc)
* Depend on what input that data type (Structured or un Structured data) depend output.

1. Feature Extraction
2. Text Vectorization

In this form Transform text in to number.

There is 2 steps –

1. Defining a vocabulary
2. Converting each text to vector representation

**Part 01 Vocabulary –**

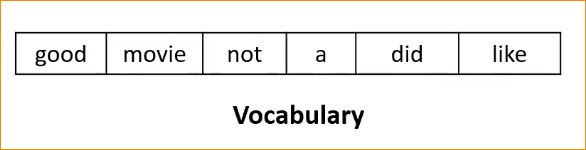
Vocabulary is subset of unique tokens in the **corpus** (Document collection)

Spam detector = Bunch of email – corpus (Collection of Document)

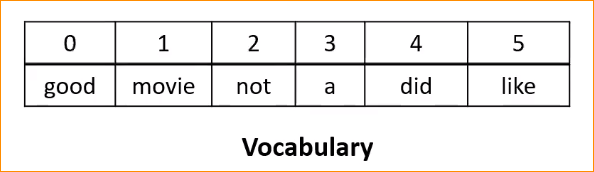
Example -



* Subset of unique token(With Nrgram)call vocabulary



* Vocabulary – the valid word that contribute to the document creations
* In here We will giving number to each word in vocabulary.



* Vocabulary can be made of Ngrams(can give more meaningful information) as well.
* Add Bigram for this –



* If add trigram for this vocabulary goes high.
* Subset of token can be chosen base on frequency and by removing stop word etc.
* We can keep threshold (that we can put limitation – ex if this word “The ” word 3 time in the paragraphs ignore that word)
* So using that kind of technique we can reduce the complex the model.

**Part 2 - Converting each text to vector representation**

Techniques –

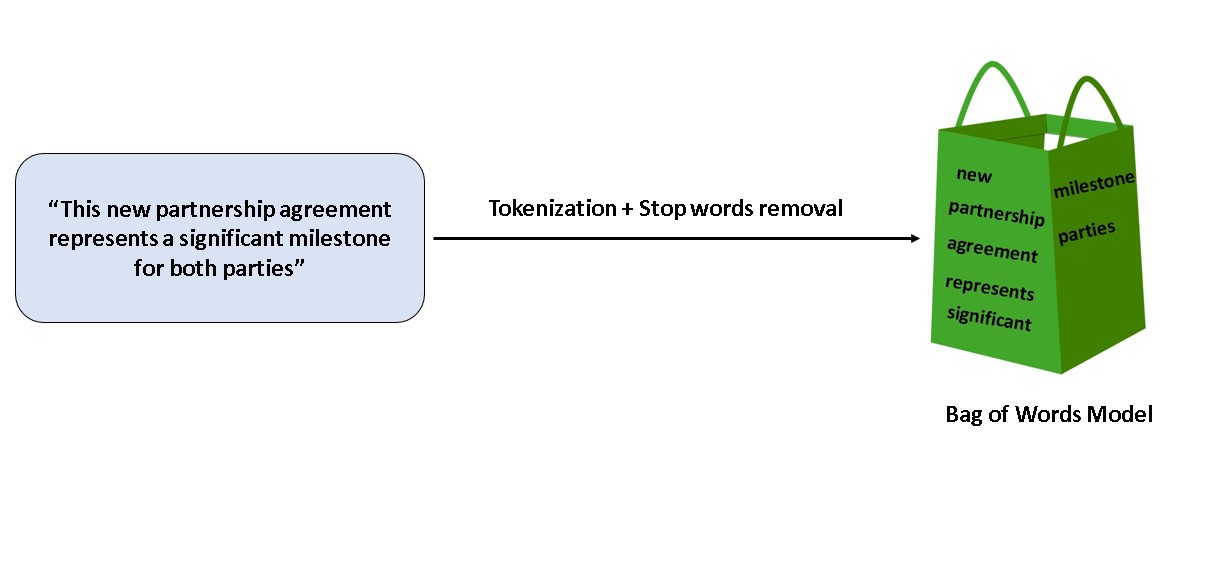
1. Bag of words
2. Ti-IDF Vector representation

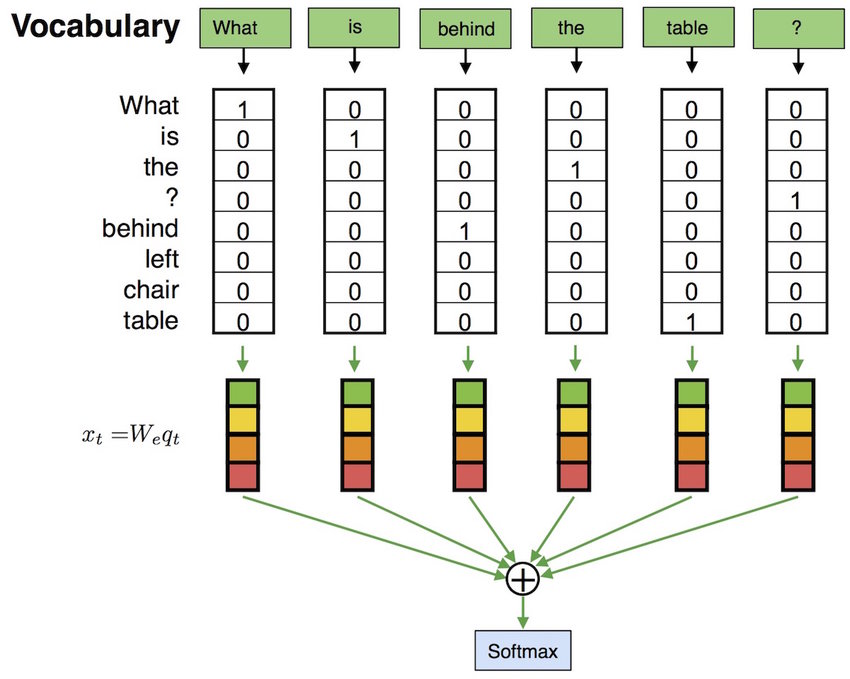
Note – What are this-

Converting text into vector representations is a foundational step in many natural language processing (NLP) tasks. These vectors capture semantic meanings of words, phrases, or documents, allowing machines to understand and process text more effectively. Here are some popular techniques to convert text to vector representations:

1. Bag of Words (BoW):

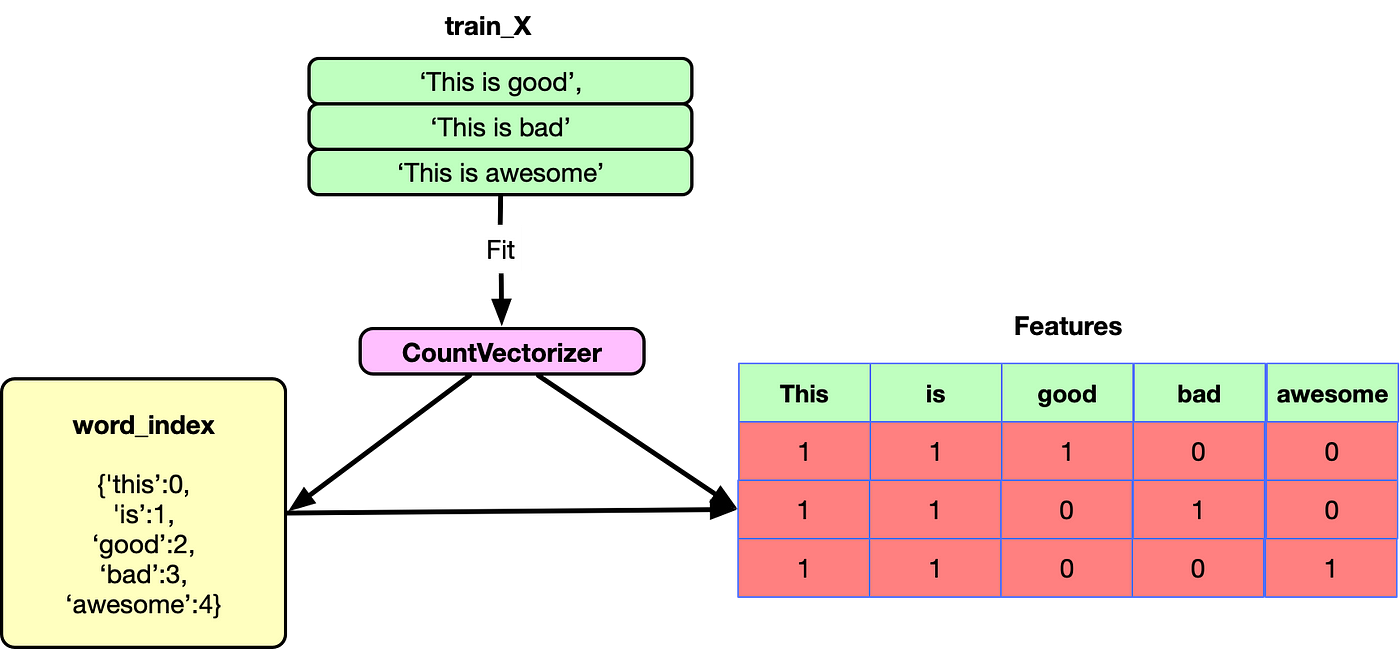
* Treats each document as an unordered set of words.
* Represents each word with an index.





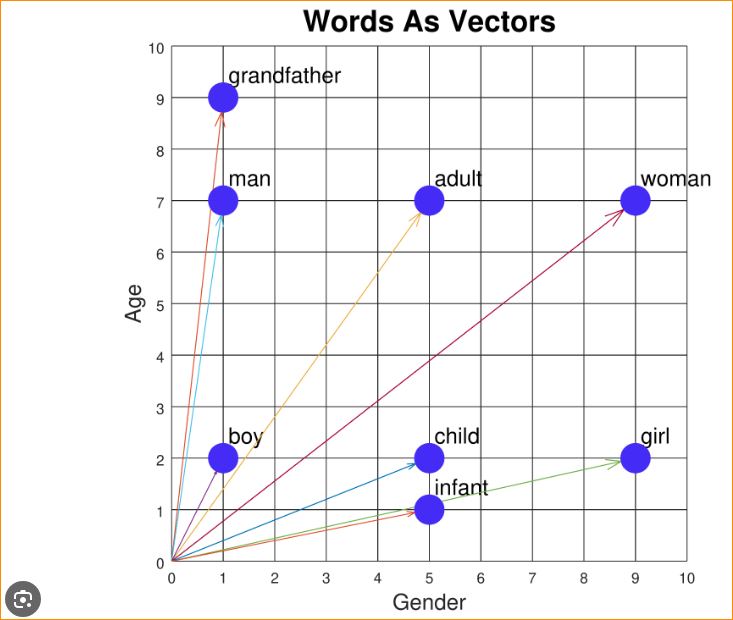
1. TF-IDF (Term Frequency-Inverse Document Frequency):

* Considers not only the frequency of a word in a document but also its frequency across all documents in the corpus.
* A word's TF-IDF score increases proportionally to its frequency in the document but is offset by the frequency of the word in the corpus.



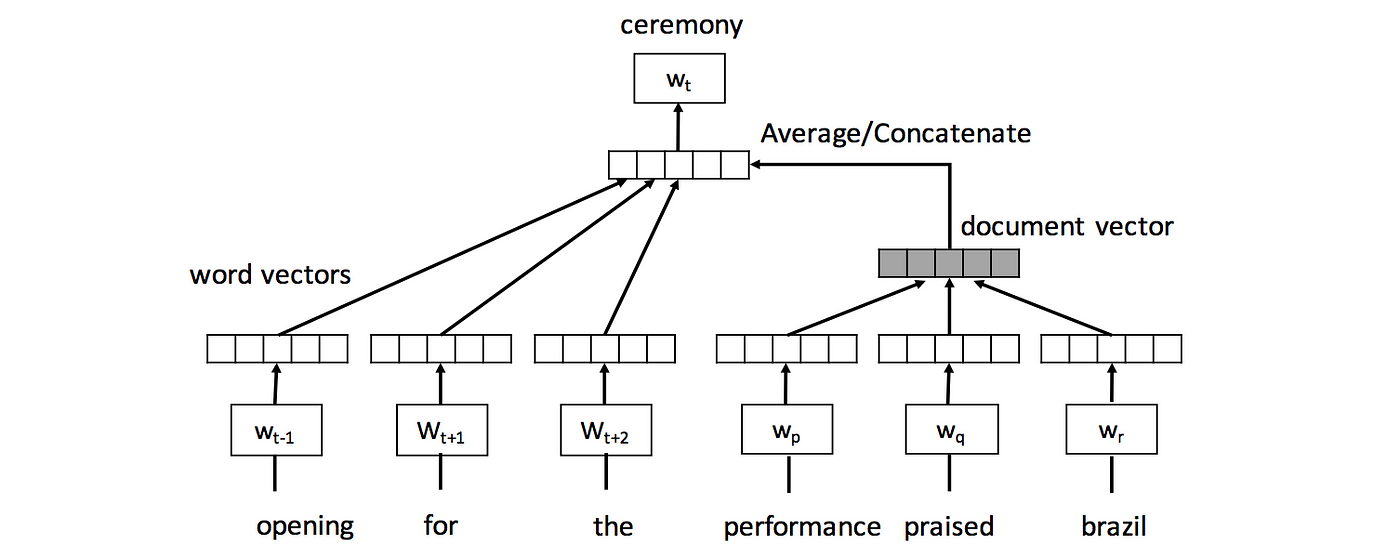
1. Word Embedding:

* Word2Vec: Uses neural networks to produce dense vector representations where semantically similar words have similar vectors.
* GloVe (Global Vectors for Word Representation): Constructs word vectors based on co-occurrence statistics from the corpus.
* FastText: Builds on Word2Vec by considering subword information, which can capture morphological structures of words.



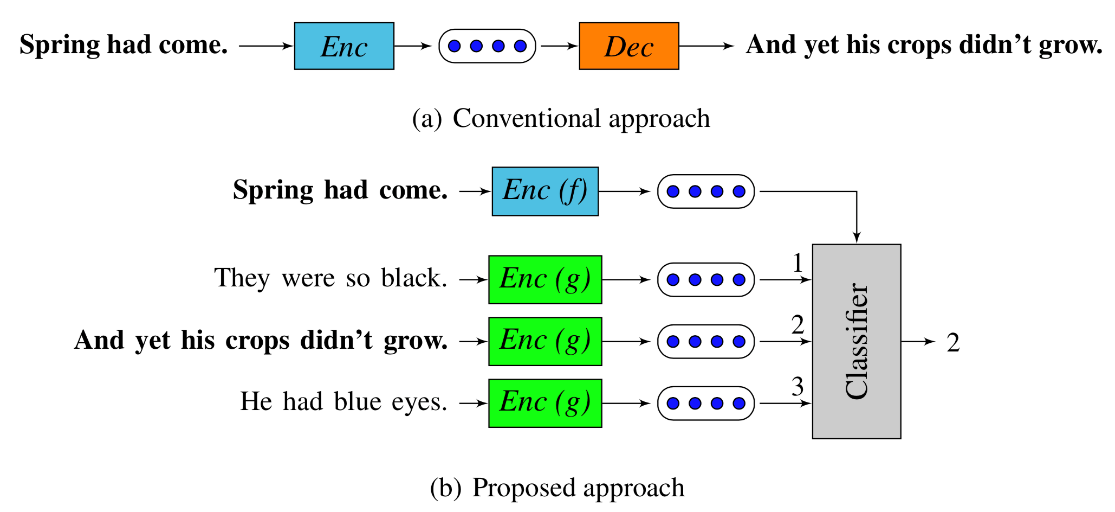
1. Document Embeddings:

* Doc2Vec (or Paragraph Vectors): Extends Word2Vec to produce vector representations for entire documents.
* BERT (Bidirectional Encoder Representations from Transformers): Uses transformer architecture to generate embeddings for words, sentences, or entire documents.



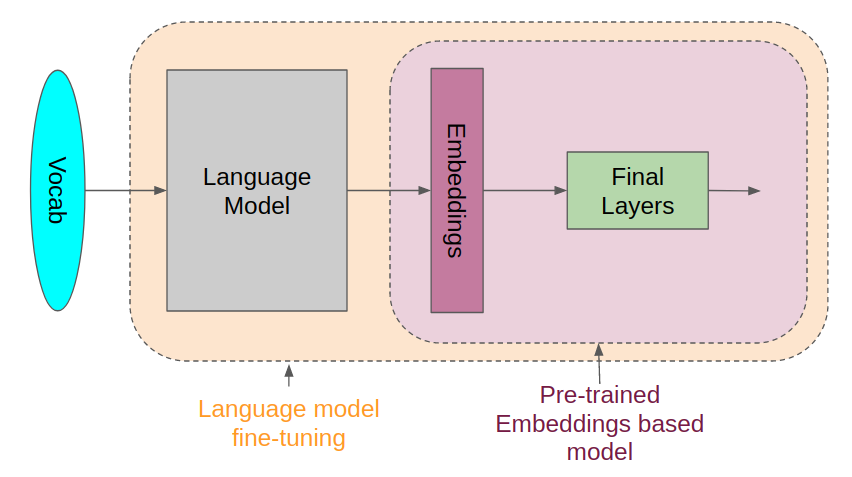
Sentence Embeddings:

Universal Sentence Encoder: Generates embeddings for sentences or short texts. It can capture semantic similarity between sentences.



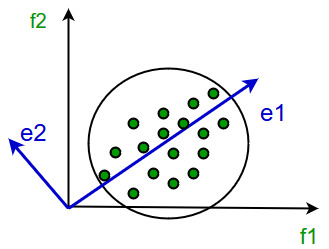
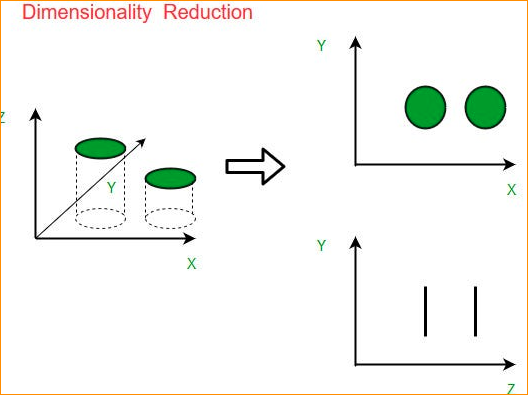
Pre-trained Language Models:

* Models like GPT (Generative Pre-trained Transformer) or BERT can be fine-tuned on specific tasks or used to extract contextual embeddings from text.



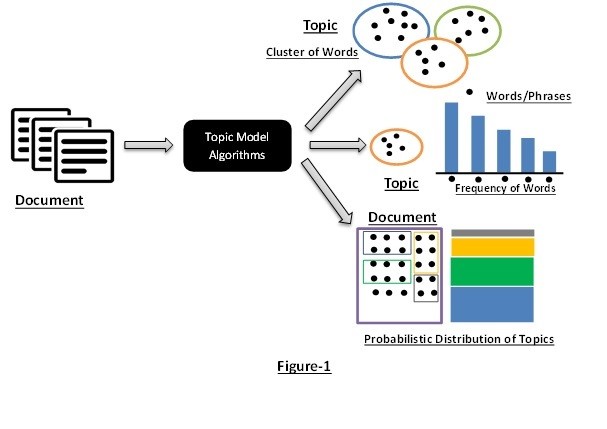
Dimensionality Reduction:

* Techniques like Principal Component Analysis (PCA) or t-SNE can be applied to reduce the dimensionality of vector representations while preserving their semantic meaning.

Topic Modeling:

* Algorithms like Latent Dirichlet Allocation (LDA) can be used to identify topics in a corpus and represent documents in terms of these topics.



Note End –

1. Bag of Words

Count the occurrence of tokens in the vocabulary.

Ex – corpos and unigram vocabulary

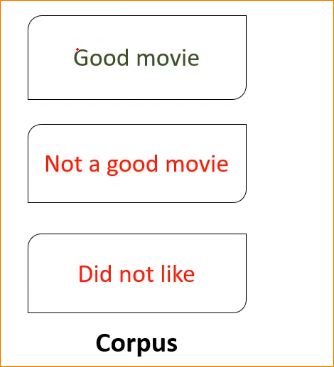
In here Word “Good” Appears as 1

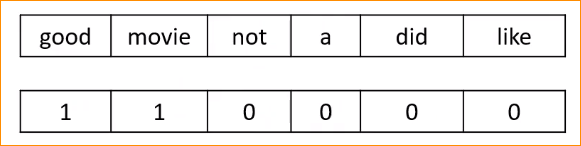
“Movies“ as 1 both Appear 1 time(Double times)

Other “not, a did and like” Appear 0 times

So we can get the count of each document.

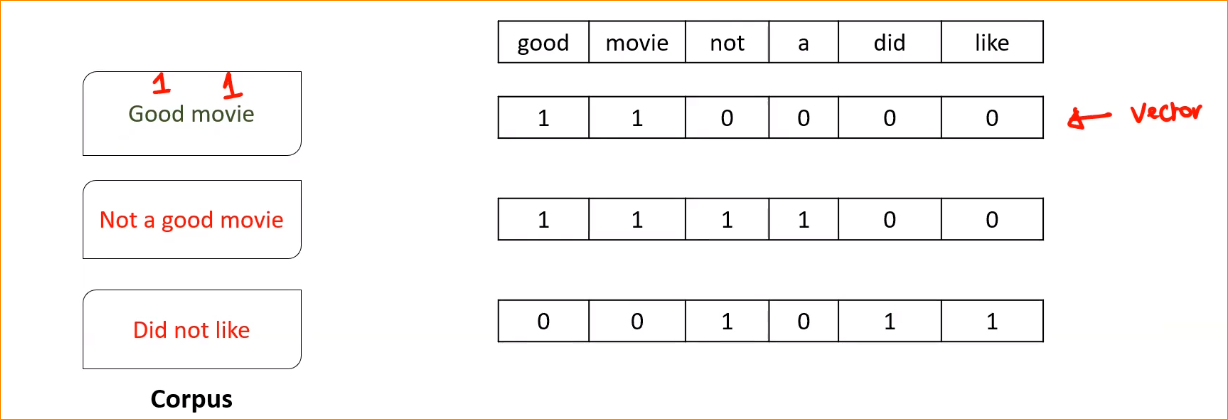
That count corresponding positioning of the vactor





Vector

Similar can create vector representation of document



Now we successfully convert text to numbers.

Problems -

In here If you get large Document there may be may “0” s.

Higher frequent words dominate. That mean word can be occurring many times. Example Domain specific words (in here example “Movies”)

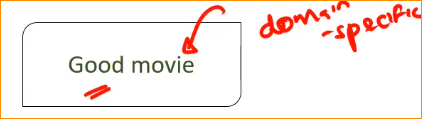
Sentiment Analyze

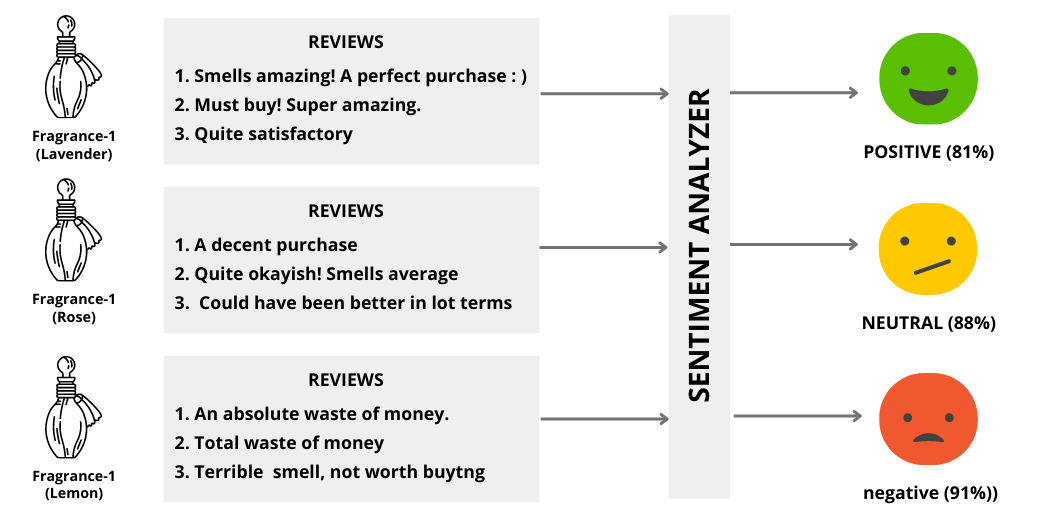
In our example using sentiment Analyze we can get Idea, that review positive or negative.

In here normally have 3 classes. (positive, neutral and nagative).

We can identify the sentiment express in the text.

If You use Sentiment Analyze , Can consider word like “Good”





But in “movies” Word appears so may time. Model think this word is very important to predict the output. So the model confused.

To avoid that We use TF-IDF technique.

1. TF-IDF vector representation

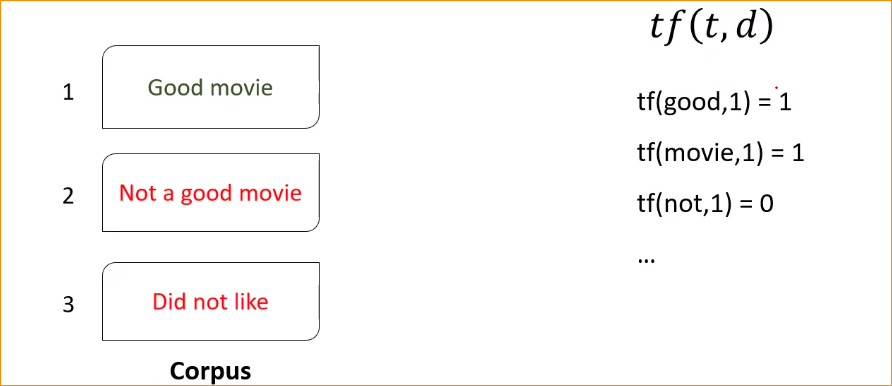
Team Frequency (TF) is computed as number of times a token *t* appears in document.

**IDF – Inverse document frequency**

**TF – Team Frequency is like bag of word technique.**

TF – We simply calculate the number of occurrences

Example –

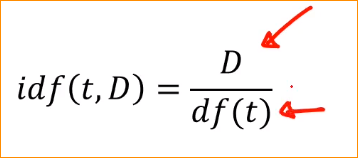


In 1st document that “movies” appearing 1 time

In 1st document that “not” appearing 0 time

**IDF – Inverse document frequency**

Here eliminate the previous problem that word that bag of word approach , where that word that occurring all the document start to dominate. Here we calculate inverse document frequency. In here we divide the total number of document that which the t appear



D- total number of document

df(t) – Denominator that number of times that term “t” appear

in here that terms that appearing in many document that denominator “df(t)” going to increase.

So that ideal score – decrees

Example –

There are 1000 document available in together.

A term is appearing only one document. For that document that term in very important.

So the score of the document = 1000 / 1 == 1000

If term appears that 500 Document(half) - In here that tem is not really important –

Because it appearing the half of document.

1000/ 500 = 2

Appearing one document that value 1000

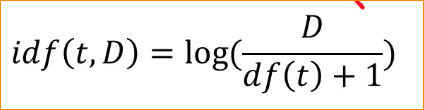
Appearing half of document that value 2

**That mean the term appearing many time in many document – that is not important**

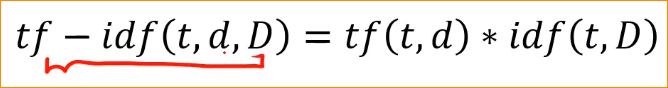
Using this we can easily handle, Domain Specific stop words

What are the problems that we have to face –

* D - Copas is very very large in the real word. If that document large that mean D is large. For that we are taking the log value with the euation.
* Denominator can be zero . So here we adding one



Multiplying the tf value and idf value give the **TF-IDF** score.



So that TF-IDF value larger mean tf value larger and idf value is larger.

Tf lager mean – that termas appear so many time in the Document

And idf value is larger (that mean denominator become smaller)

That mean , that terams appear so many time in the document (tf - high) and it should apperar less number of time in the other document. That mean that term is very important to the document. That is the idea of TF-IDF

High tf-idf score is obtained for a term with high frequency in a document and low document frequency in the corpus.

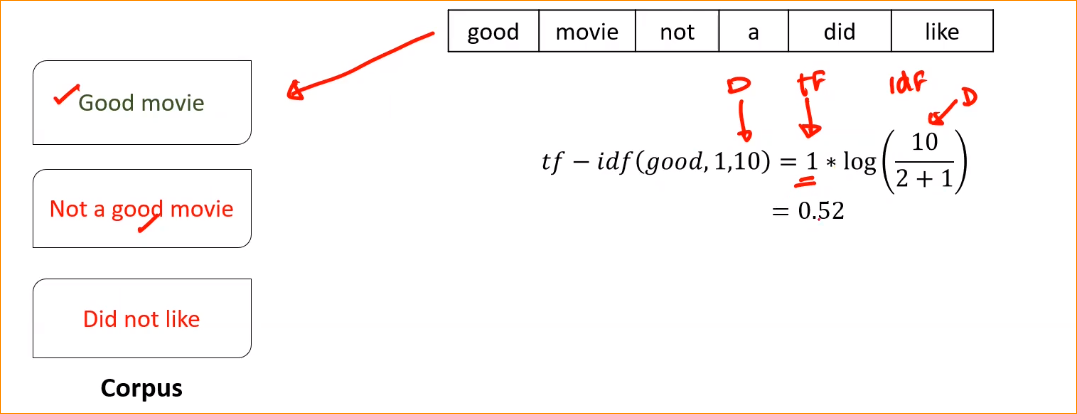
1. TF-IDF Vector Representation

Replace counters with TF-IDF scores

Calculate the TF-IDF value –

Word – “good”

Corpus size - 10



Consider of 1 st document – TF = 1