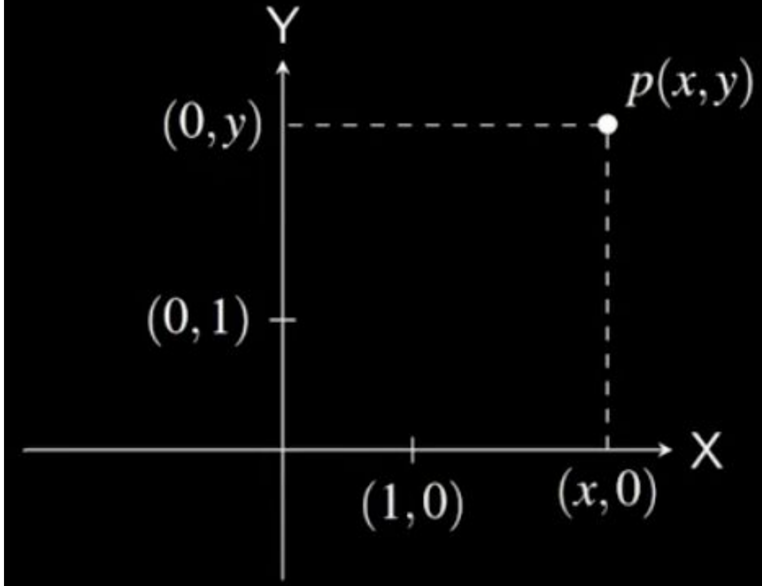
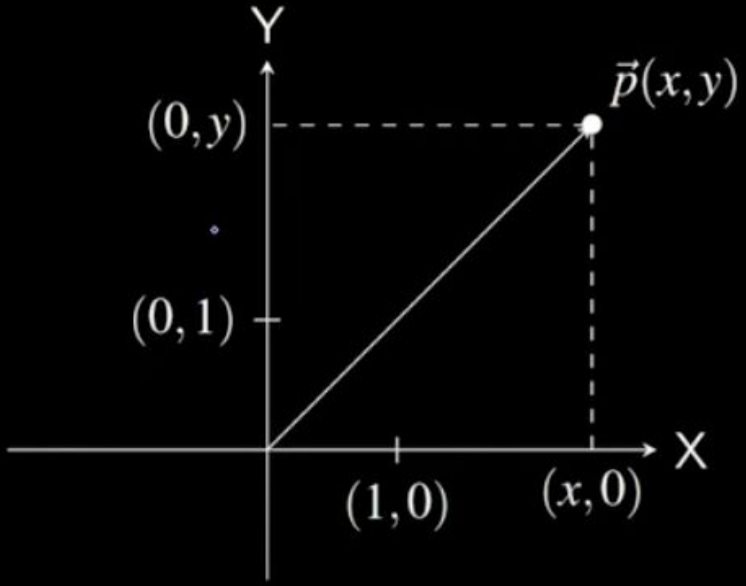
VECTOR SPACE MODELS

2-D VECTOR SPACE

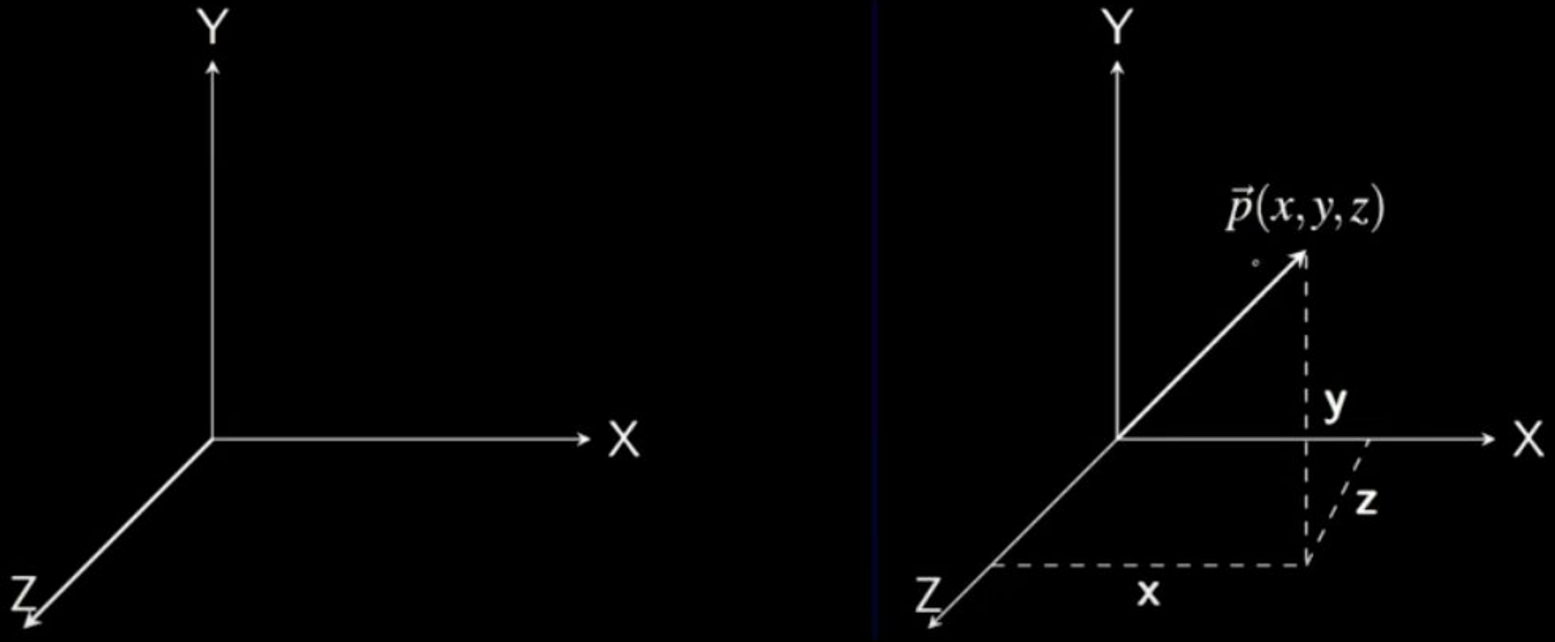
A 2-D vector-space is defined as a set of linearly independent basis vectors with 2 axes. Each axis corresponds to a dimension in the vector-space





3-D VECTOR SPACE

A 3-D vector-space is defined as a set of linearly independent basis vectors with 3 axes. Each axis corresponds to a dimension in the vector-space



Linearly independent vectors of size $\mathcal{N}$ will result in $\mathcal{N}$-dimensional axes which are mutually orthogonal to each other

VECTOR SPACE MODEL FOR WORDS

Let us assume that the words in a corpus are considered as linearly independent basis vectors. If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$. Each word takes an independent axis which is orthogonal to other words/axes. Then $\mathscr{R}$ will contain $|\mathscr{V}|$ axes.

Examples

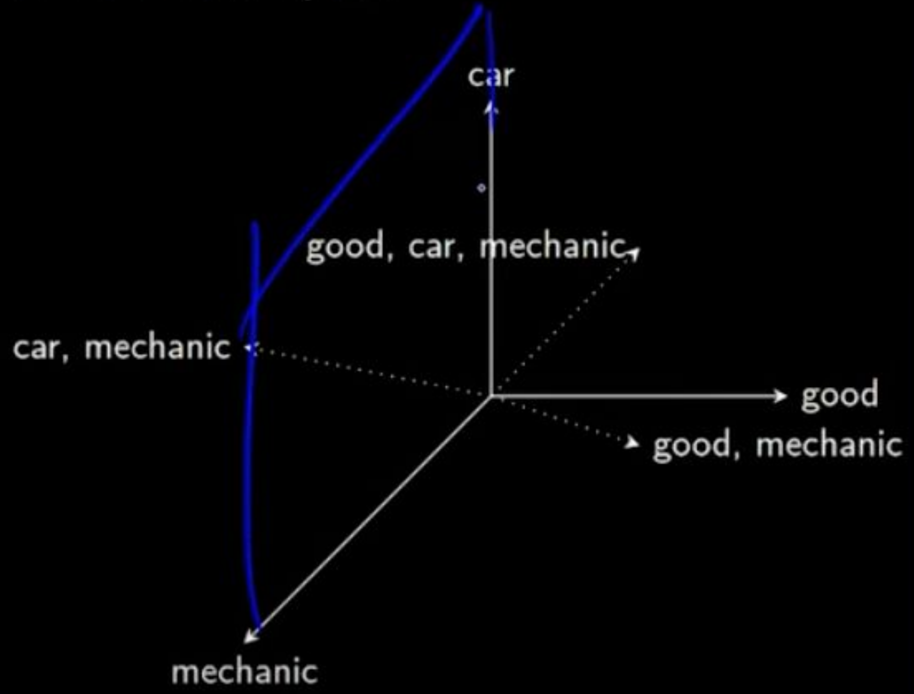
1. The vocabulary size of *emma corpus* is 7079. If we plot all the words in the real space $\mathscr{R}$, we get 7079 axes
2. The vocabulary size of *Google News Corpus corpus* is 3 million. If we plot all the words in the real space $\mathscr{R}$, we get 3 million axes

DOCUMENT VECTOR SPACE MODEL

- Vector space models are used to represent words in a continuous vector space $\mathscr{R}$
- Combination of Terms represent a document vector in the word vector space
- Very high dimensional space - several million axes, representing terms and several million documents containing several terms

EXAMPLE - BINARY INCIDENCE MATRIX

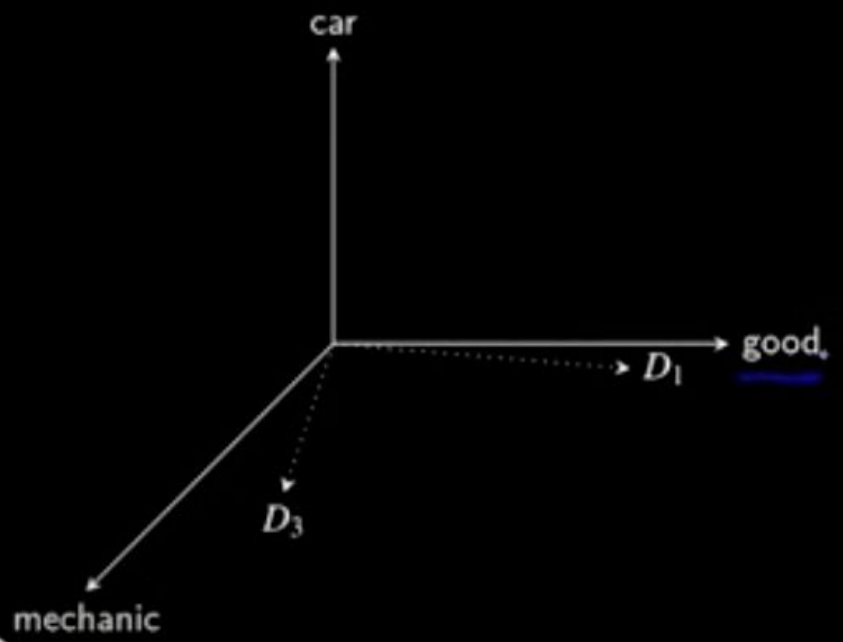
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	1	1	1
D2	1	0	1
D3	0	1	1

EXAMPLE - TF-IDF INCIDENCE MATRIX

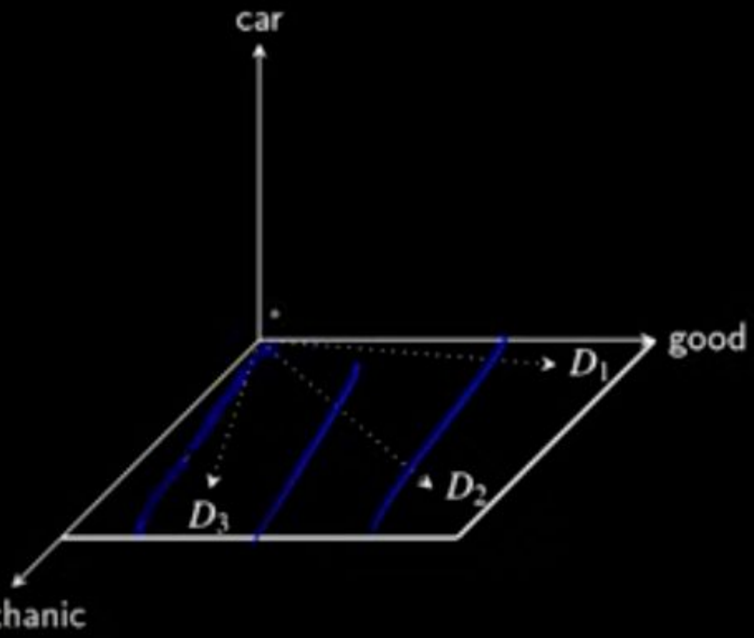
Let us consider three words - *good, car, mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

EXAMPLE - TF-IDF INCIDENCE MATRIX

Let us consider three words - *good*, *car*, *mechanic* and we will represent these words in a 3-D vector space



	good	car	mechanic
D1	0.91	0	0.0011
D2	0.21	0	0.1
D3	0.15	0	0.921

DOCUMENT-TERM MATRIX

	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12
t1	0.1	0.0	0.4	0.1	0.2	0.0	0.1	0.9	0.9	0.3	0.0	0.8
t2	0.1	0.0	0.4	0.1	0.2	0.0	0.1	0.9	0.9	0.3	0.0	0.8
t3	0.0	0.9	0.0	0.2	0.3	0.1	0.7	0.0	0.2	0.7	0.5	0.5
t4	0.0	0.9	0.3	0.9	0.5	0.1	0.9	0.3	0.8	0.4	0.1	0.4
t5	0.4	0.0	0.3	0.2	0.5	0.9	0.3	0.7	0.4	0.6	0.0	0.3
t6	0.6	0.0	0.4	0.7	0.3	0.3	0.9	0.1	0.9	0.0	0.0	0.3
t7	0.0	0.8	0.5	0.6	0.6	0.6	0.0	0.1	0.4	0.9	0.3	0.1
t8	0.4	0.0	0.6	0.5	0.5	0.1	0.7	0.1	0.5	0.3	0.8	0.1
t9	0.3	0.0	0.7	0.9	0.8	0.7	0.7	0.8	0.6	0.6	0.8	0.0
t10	0.0	0.5	0.5	0.0	0.2	0.0	0.0	0.1	0.3	0.4	0.5	0.3

The columns of the matrix represent the document as vectors. A document vector is represented by the terms present in the document

WEIGHTED-TF-IDF

Every element in the matrix represent tf-idf either in the plain form or in some of the weighted forms as given below:

$$tf.idf = tf \times log_{10}\left(\frac{N}{df_t}\right) \text{ or} \tag{1}$$

$$= w_{t,d} \times \left(\frac{N}{df_t}\right) \tag{2}$$

$$\text{where } w_{t,d} = \begin{cases} (1 + log_{10} tf_t), & \text{if } tf_{t,d} > 0 \\ 0 & otherwise \end{cases} \tag{3}$$

QUERY MODELING

Each query is modeled as a vector using the same attribute space of the documents.

$$q = \begin{bmatrix} q_{t_1} & q_{t_2} & q_{t_3} & \dots & q_{t_n} \end{bmatrix} \tag{4}$$

The relevancy ranking of a document depends on the distance of the document with respect to the query. The proximity of the query with every document is computed using distance measures.

DOCUMENT SIMILARITY

Earlier, using the binary incidence matrix, a query returned a set of documents whether the query keywords were found in documents or absent. It did not give any ranking for the retrieved documents. A similarity measure is a real-valued function that quantifies the similarity between two objects [1]. Some of the methods are given below.

Euclidean Distance - $\mathscr{E}(\vec{d}_1, \vec{d}_2) = \sqrt{d_1^2 - d_2^2}$ (5)

Cosine similarity– $\cos\left(\vec{d}_1, \vec{d}_2\right) = \frac{\vec{d}_1 . \vec{d}_2}{\left\|\vec{d}_1\right\| \left\|\vec{d}_2\right\|} = \frac{\vec{d}_1}{\left\|\vec{d}_1\right\|} . \frac{\vec{d}_2}{\left\|\vec{d}_2\right\|}$ (6)

Cosine Distance = 1 - $\cos\left(\vec{d}_1, \vec{d}_2\right)$

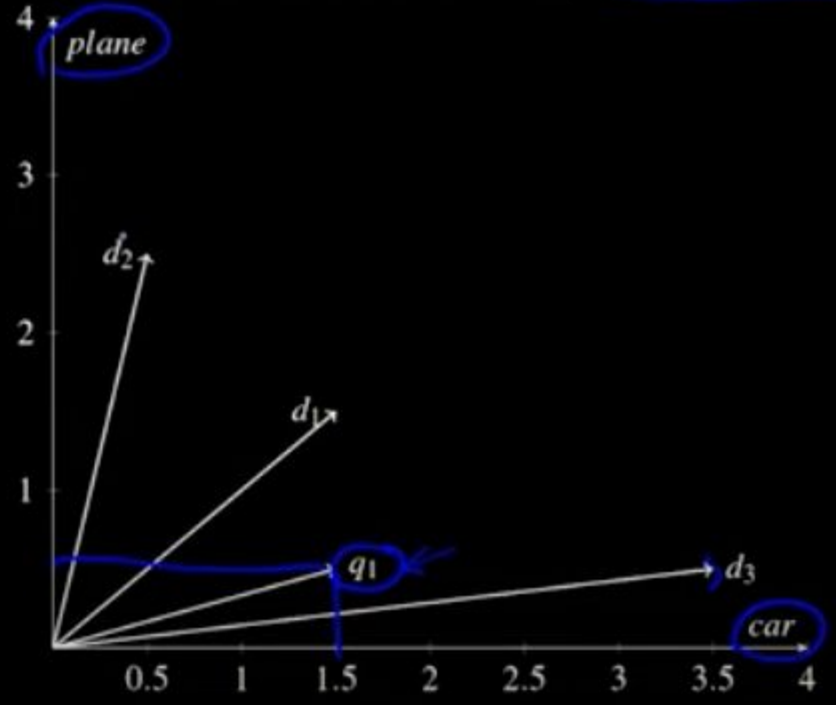
Cluster similarity-$\mathscr{L}(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 . \vec{d}_2}{\left\|\vec{d}_1\right\|_1}$ (7)

Jaccard Similarity - $\mathscr{J}(\vec{d}_1, \vec{d}_2) = \left|\frac{\vec{d}_1 \cap \vec{d}_2}{\vec{d}_1 \cup \vec{d}_2}\right|$ (8)

WHICH MEASURE?

Euclidean measure does not work well for unequal sized vectors as rthe vectors are not normalized. We often use a normalized correlation coefficient, Cosine distance for the

osine Distance = 1 - $\cos\left(\vec{d}_1, \vec{d}_2\right)$



similarity measure.

PROXIMITY SCORE

A query is considered as a document vector[2]. The proximity of the query with every document is computed using a distance measure.
Cosine distance is preferred and it is easy to compute if the document vector distances are normalized. Proximity score is listed in the descending order and the documents within a predefined proximity score (angle) will be considered as relevant and retrieved.

Cosine similarity is used instead of Cosine distance.
The Github containes the modified version of this program

	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
D0	0.0	4.0	90.0	45.7	50.6	64.8	41.9	64.6	74.6	72.1	56.9
D1	4.0	0.0	90.0	46.9	52.6	66.0	42.3	67.3	75.3	73.7	59.0
D2	90.0	90.0	0.0	56.5	66.1	71.8	59.5	81.4	57.6	41.7	61.7
D3	45.7	46.9	56.5	0.0	39.5	46.6	28.5	58.5	53.9	45.2	49.7
D4	50.6	52.6	66.1	39.5	0.0	29.5	48.9	53.8	60.8	31.9	36.1
D5	64.8	66.0	71.8	46.6	29.5	0.0	58.1	54.3	66.9	40.5	61.2
D6	41.9	42.3	59.5	28.5	48.9	58.1	0.0	63.0	56.4	53.5	50.5
D7	64.6	67.3	81.4	58.5	53.8	54.3	63.0	0.0	54.3	51.1	69.1
D8	74.6	75.3	57.6	53.9	60.8	66.9	56.4	54.3	0.0	50.3	69.2
D9	72.1	73.7	41.7	45.2	31.9	40.5	53.5	51.1	50.3	0.0	44.5
D10	56.9	59.0	61.7	49.7	36.1	61.2	50.5	69.1	69.2	44.5	0.0

Document Rank for the query D0

D0	0.0
D1	4.0
D6	41.9
D3	45.7
D4	50.6
D10	56.9
D7	64.6
D5	64.8
D9	72.1
D8	74.6
D2	90.0

Demo link

https://github.com/Ramaseshanr/anlp

Contextual Understanding of Words

CONTEXTUAL UNDERSTANDING OF WORDS

- The study of *meaning* and *context* should be central to linguistics
- Exploiting the context-dependent nature of words
- Language patterns cannot be accounted for in terms of a single system
- The *collocation*, gives enough clue to understand a word and its meaning
- *No study of meaning apart from context can be taken seriously* [2]

UNDERSTANDING A WORD FROM ITS CONTEXT

The view from the top of the mountain was

awesome
breathtaking
amazing
stunning
astounding
astonishing
awe-inspiring
extraordinary
incredible
unbelievable
magnificent
wonderful
spectacular
remarkable

Collocations & Dense word Vectors

COLLOCATIONS

Collocations is a juxtaposition of two or more words that more often occur together than by chance.

- Poverty is a ***major problem*** for many countries
- Ram has a ***powerful computer***
- I had a ***brief chat*** with Raj
- I could not see anything in the room, it was ***pitch dark*** inside
- The crime was committed in ***broad daylight*** - We don't use wide, large, big daylight
- I wish I had a ***strong tea*** - we don't use powerful, tough
- The ***heavy rain*** prevented us from playing outside - We don't use strong rain
- Someone ***knocked*** on the front ***door***

CREATION OF SEMANTICALLY CONNECTED VECTORS

- Identify a model that enumerates the relationships between terms and documents
- Identify a model that tries to put similar items closer to each other in some space or structure
- A model that discovers/uncovers the semantic similarity between words and documents in the latent semantic domain
- Develop a distributed word vectors or dense vectors that captures the linear combination of word vectors in the transformed domain

METHODS TO CREATE DENSE VECTORS

- Latent Semantic Analysis or Latent Semantic Indexing
- Neural networks using skip grams and CBOW
  - CBOW - uses surrounding words to predict the center of words
  - Skip grams use center of words to predict the surrounding words
- Brown clustering - statistical algorithms for assigning words to classes based on the frequency of their co-occurrence with other words

WHY DENSE VECTORS?

- Sparse vectors are too long and not very convenient as features machine learning
- Abstracts more than just frequency counts
- It captures neighborhood words that are connected by synonyms
  - Consider these two documents (1) Automobile association (2) car driver
  - Connects the neighbor of Automobile and the neighbor of car
  - "Automobile association" with "car driver" - driver and association could be connected using the similar words ***Automobile and car***

Vector Space models

VECTOR SPACE MODEL FOR WORDS

Let us assume that the words in a corpus are considered as linearly independent basis vectors.
If a corpus contains $|\mathscr{V}|$ words which are linearly independent, then every word represents an axis in the continuous vector space $\mathscr{R}$.
Each word takes an independent axis which is orthogonal to other words/axes.
Then $\mathscr{R}$ will contain $|\mathscr{V}|$ axes.

Examples

1. The vocabulary size of *emma corpus* is 7079. If we plot all the words in the real space $\mathscr{R}$, we get 7079 axes
2. The vocabulary size of *Google News Corpus corpus* is 3 million. If we plot all the words in the real space $\mathscr{R}$, we get 3 million axes

The fourth one is to really understand what
are the words and what those words convey

DOCUMENT VECTOR SPACE MODEL

- Vector space models are used to represent words in a continuous vector space $\mathscr{R}$
- Combination of Terms represent a document vector in the word vector space
- Very high dimensional space - several million axes, representing terms and several million documents containing several terms

Binary Incidence Matrix TF-IDF Incidence matrix Query modeling Document similarity Information Extraction Named Entity Recognition

CREATION OF SEMANTICALLY CONNECTED VECTORS

- Identify a model that enumerates the relationships between terms and documents
- Identify a model that tries to put similar items closer to each other in some space or structure
- A model that discovers/uncovers the semantic similarity between words and documents in the latent semantic domain
- Develop a distributed word vectors or dense vectors that captures the linear combination of word vectors in the transformed domain

HUMAN/MACHINE LEARNING

- How do we solve problems when we lack sufficient knowledge?
- Finding Examples and using experience gained are useful
- Examples provide certain underlying patterns
- Patterns give the ability to predict some outcome or help in constructing an approximate model
- The model may help resolve some problems, though may not be an ideal one
- **Learning** is the key to the ambiguous world
- Linear and non-linear classification
- Perceptron, perceptron learning, cost function, feed forward neural network, back propagation algorithm

WORD EMBEDDING

- Process each word in a Vocabulary of words to obtain a respective numeric representation of each word in the Vocabulary
- Reflect semantic similarities, Syntactic similarities, or both, between words they represent
- Map each of the plurality of words to a respective vector and output a single merged vector that is a combination of the respective vectors

1. Continuous bag of words (CBOW) Model
2. Skip-gram model
3. Discuss Word2Vec model