UG PROJECT

By **Y Rushikesh Kumar, 19124043** Under the guidance of **Prof. Dr. Sunil Kumar**

Department of Mathematical Sciences
Indian Institute of Technology (BHU) Varanasi
Varanasi-221005, India

Topic - i) Working on Recommender systems, Natural Language processing and Neural networks.

1) Recommender Systems

Introduction:

A **recommender system**, or a **recommendation system** (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

The two most common types of recommender systems are Content-Based and Collaborative Filtering (CF).

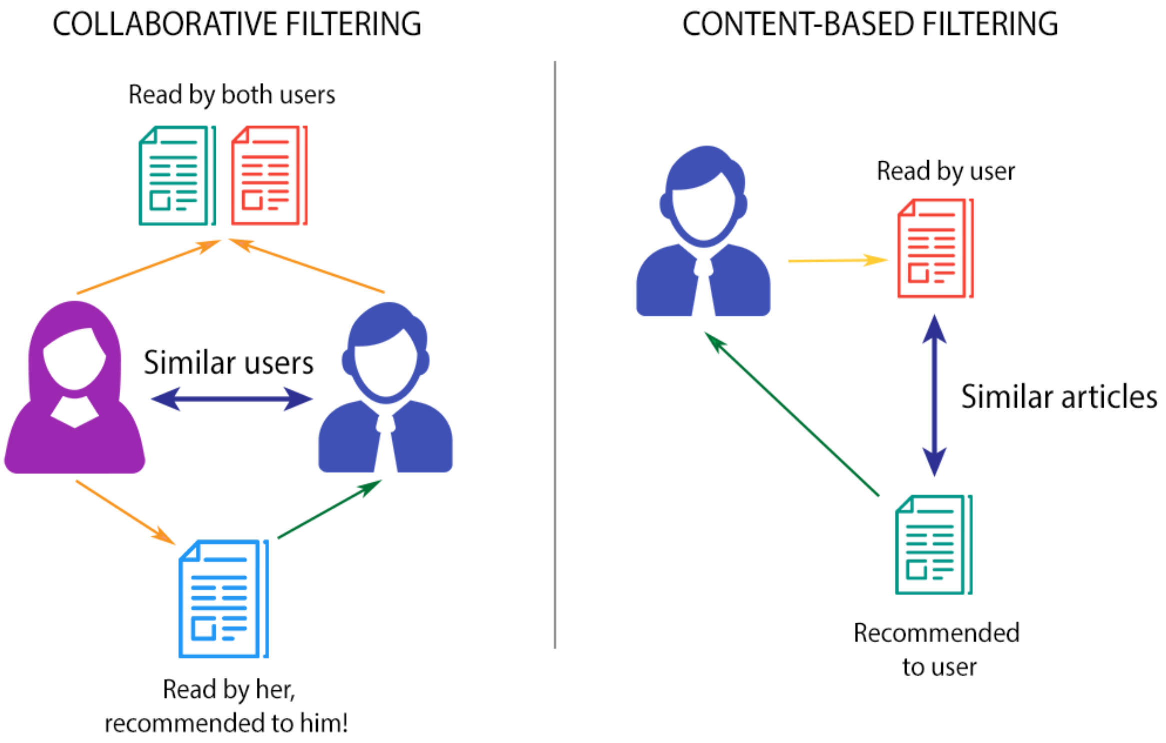
➔Collaborative filtering produces recommendations based on the knowledge of users' attitude to items that if it uses the “wisdom of the crowd” to recommend items.

➔Content-based recommender systems focus on the attributes of the items and give you recommendations based on the similarity between them.

➔ In general, collaborative filtering (CF) is more commonly used than content-based systems because it usually gives better results and is relatively easy to understand (from an overall implementation perspective).

➔The algorithm has the ability to do feature learning on its own which means that it can learn for itself what features to use.

➔CF can be divided into memory-based Collaborative Filtering and Model-based dCollaborative filtering.



Results :

Here I created a content based recommender system for a data set of movies.
We focussed on providing a basic recommendation system by suggesting items that are most similar to a particular item, in this case, movies.

2) Natural Language Processing

Introduction:

NLP algorithms are **typically based on machine learning algorithms**. Instead of hand-coding large sets of rules, NLP can rely on machine learning to automatically learn these rules by analyzing a set of examples (i.e. a large corpus, like a book, down to a collection of sentences), and making a statistical inference.

We first compile documents, Featurize documents, Compare their features

Let us consider an example:

You have two documents

- Red house
- Blue house

Featurize based on word count:

- Blue house - (red, blue, house) - (0,1,1)
- Red house - (red, blue, house) - (1,0,1)

- ❖ A document represented as a vector of word counts is called a “Bag of Words”
- ❖ You can use a cosine similarity on the vectors made to determine their similarity.
- ❖ We can improve on Bag of Words by adjusting word counts based on their frequency in corpus.
- ❖ We can use TF-IDF (which is term frequency - inverse document frequency)
- ❖ Term Frequency - Importance of the term within that document

TF(d,t) = Number of occurrences of term t in document d

- ❖ Inverse Document Frequency - Importance of the term in the corpus

IDF(t) = log(D/t) where
D = total number of documents
t = number of documents with the term

- TF - IDF is expressed as

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

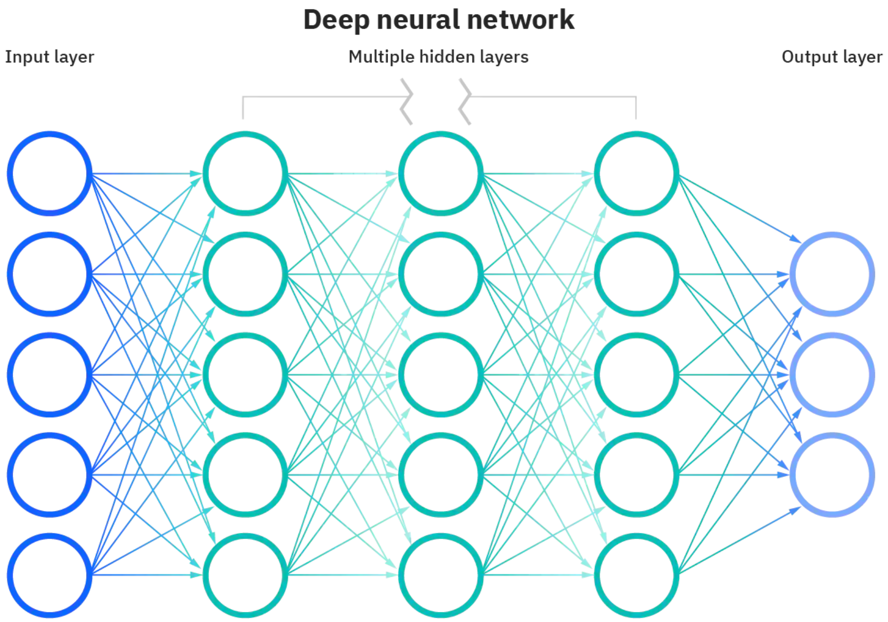
Results :
Here I attempted to classify Yelp Reviews into 1 star or 5 star categories based on the text content in the reviews.

I used the Yelp Review DataSet from Kaggle.

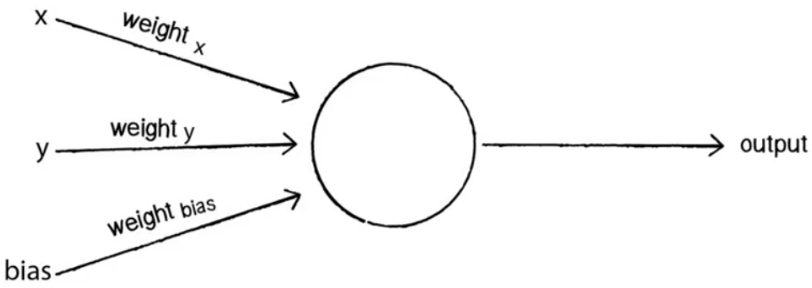
Each observation in this dataset is a review of a particular business by a particular user.

3) Neural Networks

The human brain has interconnected neurons with dendrites that receive inputs, and then based on those inputs, produce an electrical signal output through the axon.



The simplest Neural Network possible is the perceptron.



A perceptron follows 4 main steps: receive inputs, Weight inputs, Sum inputs, Generate output.

The output is generated by passing that sum through an activation function like trigonometric, logistic, step etc.

To avoid the case of input being 0 there is a third input known as Bias.

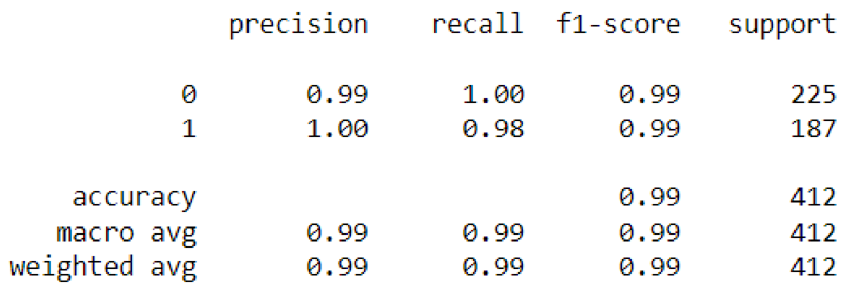
To actually train the perceptron we use the following steps:

1. Provide the perceptron with inputs for which there is a known answer.
2. Ask the perceptron to guess an answer.
3. Compute the error. (How far off from the correct answer?)
4. Adjust all the weights according to the error.
5. Return to Step 1 and repeat!

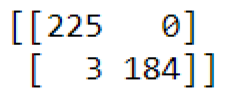
Results:

Here we used the Bank Authentication Data Set from the UCI repository.
The data here consists of variance, skewness, curtosis Entropy and class
Where class indicates whether or not a Bank Note was authentic.
And in the end we created a Random forest classifier and compared the confusion matrix and classification report to the DNN model.

Result:
DNN -

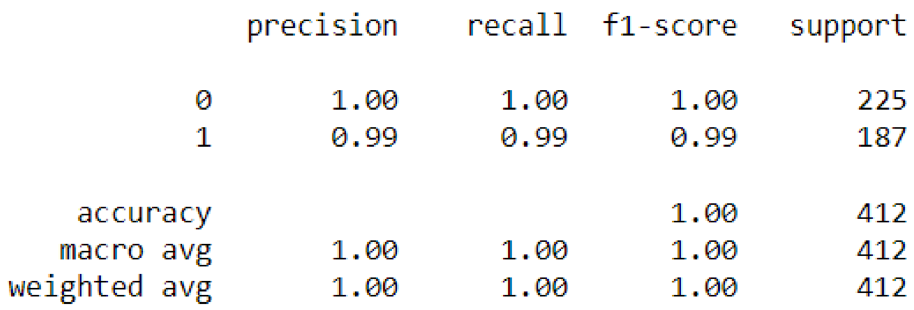


```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       225
           1       1.00      0.98      0.99       187

    accuracy                           0.99       412
   macro avg       0.99      0.99      0.99       412
weighted avg       0.99      0.99      0.99       412
```

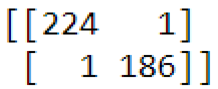


```
[[225   0]
 [  3 184]]
```

Random Forest -



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       225
           1       0.99      0.99      0.99       187

    accuracy                           1.00       412
   macro avg       1.00      1.00      1.00       412
weighted avg       1.00      1.00      1.00       412
```



```
[[224   1]
 [  1 186]]
```