IR ASSIGNMENT-1

FIRST SEMESTER 2016-2017

BY

IPSIT SAHOO - 2014A7PS095H

KOUSHIK KRISHNA - 2014A7PS014H

[VEDIRE RAGHAVENDER REDDY](https://www.facebook.com/raghavenderreddyvedire)

[NIRMALKUMAR JULURU](https://www.facebook.com/nirmalkumar.juluru)



**OBJECTIVE**

The main objective of this assignment was to build our own text-based retrieval system. The first part of the assignment was to take a large collection of data and produce a dictionary of words. The second part of the assignment was to building the searching module with Vector Space Model of information Retrieval

**TECHNOLOGY USED**

**Python** is the programming language that was used by us to design the entire retrieval system. One of the major reasons we chose Python is because of its vast amount of in-built functions for handling and maintaining data structures as well as strings. This makes the code easier to handle for debugging compared to other programming languages. Other libraries like **NumPy** and **SciPy** were for performing matrix manipulations.

* CORPUS  
  **NLTK Data** is the corpus used for creating our dictionary.
* JUPYTER Notebook has been used to create a proper documentation for the entire project.

**ARCHITECTURE**

* The retrieval system was based on a vector-space retrieval model. NLTK Data was used for getting the file IDs of the existing files in the database that the user might search for. The file-IDs were stored in a list. This Python list serves as a reference for searches made by the user. Samples of text and words were collected from different formats of text like Shakespearean Language, names, twitter language etc.
* Porter Stemmer Algorithm is used for normalization of words and removes certain endings of a word to convert it to a common root form. The code uses a inbuilt Porter Stemmer algorithm which is stored in a matrix.
* Then we calculate the Inverse doc frequency for calculating tf for the tf-idf matrix.
* The final part of building the search database is creating a normalized tf-idf matrix by calculating Euclidean distance.
* Get the query and calculate Cosine Similarities
* Output the file-IDs and Corresponding Corpus by sorting the cosine similarities.

**MAJOR DATA STRUCTURES USED ARE**:

* LIST: The most basic data structure in Python is the **sequence**. Each element of a sequence is assigned a number - its position or index. The first index is zero, the second index is one, and so forth.
* Matrix: 2D array formats which have multiple lists forming its basic building blocks. It is used to build the sparse matrix.
* np.array: A numpy array is a grid of values of the same data-type and is indexed by nonnegative integers. The number of dimensions is the rank of the array. the shape of an array is a tuple of integers giving the size of the array along each dimension
* SciPy Compressed Sparse Matrix
* List of tuples at the end.