DS 5230 Unsupervised Machine Learning
Project: Plagiarism Detection

Team Members:
Aishwarya Vantipuli
Spatika Krishnan
Nandini Jampala

Milestone 1:

The goal of this project is to find the list of students with their extent of plagiarised work for each question when their answers are given as input.

Pre-Processing and EDA:
For Text preprocessing we made sure that all the words are in lowercase with punctuations and stopwords removed.There are various latin words in answers so we used ”latin1” encoding for reading the text. Plots are generated for most frequent unigrams(fig 1), bigrams(fig 2) for each question answered by the students. We want to compare two different methods for the task plagiarism detection. Algorithms used and results obtained so far using both the methods are discussed in subsequent sections.

Method 1:

Glove Representation of Words:
We used Glove representation as proposed in the abstract. The GloVe is a log-bilinear model which is trained on the non-zero entries of a global word-word co-occurrence matrix, which tabulates how frequently words co-occur with one another in a given corpus.The advantage of GloVe is that, unlike Word2vec, GloVe does not rely just on local statistics (local context information of words), but incorporates global statistics (word co-occurrence) to obtain word vectors.
After performing GloVe on our preprocessed data we get vectors of n (=50) dimensions for each unique word in our corpus. Using the **K-means algorithm**, we cluster these word vectors into K clusters based on their contextual meaning.The output of this step is K word clusters .

Vector Representation for document- level embeddings:
From the obtained clusters and its respective labels from the K-means clustering, we now use these clusters to convert our data in each document to vectors. For this, we do the following:

- Determine the cluster label for each of the words in sentences and substitute the word with its respective cluster label.
- From the above step, we merge all the words and form sentence vectors.
- From these sentence vectors, we form an answer (document) vector by assigning a unique number if two sentence vectors are different. If two sentence vectors are the same we assign each of them the same number.

Thus we get a vector for each answer(document). This vector can be viewed as an item-set and each answer can be viewed as a transaction.Thus we get a list of transactions which contain item-sets. We can feed this into the FP-Growth Algorithm to get frequent item sets,i.e., plagiarised answers.

Scope for Milestone 2:

With the resultant document vectors as transactions in *Method 1*, we want to apply FP-Growth Algorithm to mine frequent documents based on sentence similarities which ultimately gives a list of documents which were plagiarised. We also want to apply hierarchical clustering and compare results with K-means.

***Method 2* (Not mentioned in the abstract):**

After implementing the pre-processing steps such as stop-word removal, tokenizing and stemming we do the following steps to cluster the similar documents together to detect plagiarism.

Tf-Idf (Term frequency–Inverse document frequency)Vector :
This is a document term matrix which tells us how important a word is in the corpus. This results in a matrix where rows is equal to the number of documents and column is the number of words present. So basically this represents every document as a row vector.The tf-idf weight is the product of two terms: the first computes the normalized TF, aka. the number of times a word appears in a document, divided by the total number of words in that document; the second term is the IDF,computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

Cosine Similarity :
The distance is a very important measure to find the similarity between clusters as closely spaced documents should belong to one cluster. Thus using Cosine distance we create a similarity matrix. So here we obtain a matrix(N*N) matrix where each value is the distance between two answers and N being our number of answers.

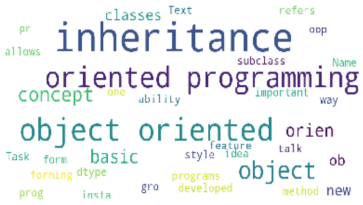
K-means Clustering Algorithm :
When K-means was performed with a tf-idf vector it resulted in a very high SSE(Sum of Squared Errors) and gave incorrect clusters thus failing our aim. Hence we decided to use k-means on our cosine similarity matrix as it gave us a low SSE and similar answers resulted in one cluster. The answers which are not in the same cluster as the original(reference) answer would be classified as non-plagiarised work.

Scope for milestone 2:
In our resultant clusters we see that many answers are in the same cluster as the original(reference) answer so, to classify those clusters into heavily or lightly plagiarised similarity score within each cluster is measured using either tri-gram sequence matching or Levenshtein's Edit Distance. So according to the similarity score we can classify the extent of plagiarism. We also want to apply hierarchical clustering and compare results with K-means in both the methods.

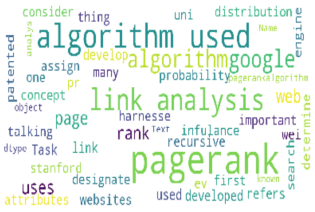
Visualizations

1. Word clouds:

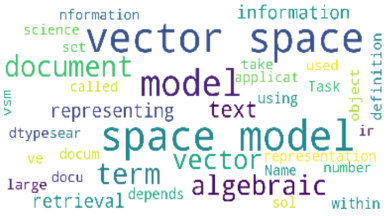
Inheritance



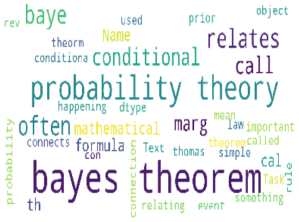
Page Rank



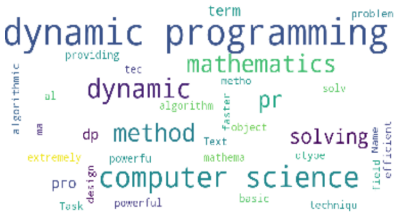
Vector space model



Bayes theorem



Dynamic programming



2. Top Frequent Bigrams: