# Text Analytics Assignment Report

Name: Fareed Hassan Khan

ERP ID: 25367

Assignment 1

## Dataset Description

I explored the BBC News dataset on Kaggle, which contains over 14,000 news articles published by the British Broadcasting Corporation (BBC) over a six-year period. The dataset covers five categories, namely business, entertainment, politics, sport, and technology. Moreover, each news article is accompanied by a short description, giving a quick insight into its contents.

## Project Overview

In my project, I started with the traditional Bag of Words model, followed by Word2Vec, GloVe, and Customized Word2Vec models. To improve the models' performance, I tuned their parameters and experimented with dimensionality reduction techniques. I also compared the models' results to observe the effects of each parameter and technique. Furthermore, I implemented the K-Means clustering algorithm to cluster similar documents, which significantly reduced computation resources and processing time. I compared this approach to the traditional complete knowledge base approach to evaluate the effectiveness of the K-Means algorithm.

Overall, my project highlighted the significance of selecting the right NLP models, tuning their parameters, and dimensionality reduction techniques to improve their performance. Additionally, using clustering algorithms like K-Means can significantly reduce computation resources and improve the processing time.

## Bag of Words

After preprocessing the data by lemmatizing and stemming it, I applied bag of words with both tf-idf and count vectorizer and found that tf-idf outperformed count vectorizer in terms of accuracy.

To evaluate the models, I used a metric that involved finding the top three most similar documents to a single document from the dataset. I discovered that the use of max_features affected the performance of both tf-idf and count vectorizer, with models using the complete number of features performing better than those that were limited.

**[With Kmeans]** - Furthermore, I found that using a combination of both uni-grams and bi-grams as n-grams was more efficient than using only uni-grams or bi-grams. Interestingly, when I used k-means clustering with the best model (tf-idf), I found that the computation time for finding the cosine similarity of a new document from a single cluster was not significantly reduced compared to finding the similarity from the complete dataset. This could be due to the fact that the majority of the documents belonged to the same clusters, accounting for almost 72% of the data.

## Word2Vec Pre-Trained

I decided to use lemmatized news instead of stemmed while working with word2vec. For this purpose, I applied a pre-trained word2vec model using the Google News corpus with binary True parameter and a limit of 50000. I chose this limit parameter because going above this value and making the binary value false would result in a memory allocation error, making the word2vec matrix too large.

However, when compared to our best bag-of-words (tfidf) model, pre-trained word2vec did not perform better. The accuracy matrix remained the same, which involved finding the top three most similar documents to a single document from the dataset. **[With Kmeans]** It is worth noting that the k-means approach reduced computation time since each cluster contained an equal number of documents, unlike tfidf.

Although word2vec is a useful tool, it may not always be the optimal choice for every scenario.

## GloVe Pre-Trained

I used lemmatized data while working with GloVe, just as I did with word2vec. I applied two different pre-trained GloVe models, one with 50 dimensions and the other with 300 dimensions, with a binary False parameter. Both models produced some positive results, depending on the matrix used with previous approaches.

When compared to our best bag-of-words (tfidf) model and word2vec, the GloVe 300-dimension pre-trained model performed better than word2vec, but showed approximately the same results as compared to BOG. The accuracy matrix remained the same, which involved finding the top three most similar documents to a single document from the dataset. **[With Kmeans]** Additionally, the k-means approach was effective in reducing computation time, as each cluster contained an equal number of documents with this pre-trained GloVe approach, unlike tfidf.

Overall, my experiment suggests that using lemmatized data with GloVe can produce promising results and may be a suitable alternative to word2vec in certain scenarios. Additionally, the k-means approach can be useful in reducing computation time when working with large datasets.

## Customized Word2Vec

I trained a customized Word2Vec model using two approaches, Skipgram and CBOW, and evaluated their accuracy using a metric that involved finding the top three most similar documents to a single document from the dataset and repeating this step for ten documents. The results showed that Skipgram performed slightly better than CBOW in capturing semantic similarity between words in the dataset. Additionally, the customized Word2Vec model using Skipgram performed almost as well as pre-trained models such as GloVe and tf-idf, suggesting that it is a viable alternative for analyzing text data.

**[With Kmeans]** Furthermore, I explored the effectiveness of the k-means clustering approach in reducing computation time when analyzing large text datasets. The results

showed that the k-means clustering approach effectively reduced computation time, particularly with the pre-trained GloVe approach.

In conclusion, this approach determines the effectiveness of a customized Word2Vec model using the Skipgram algorithm for analyzing text data. The model performed almost as well as pre-trained models such as GloVe and tf-idf, and the k-means clustering approach proved to be an effective tool in reducing computation time.

## SVD Dimensionality Reduction

I applied pre-trained Word2Vec models with SVD dimensionality reduction. We reduced the dimensionality of each document using SVD to 50. Our results showed that the performance of the pre-trained Word2Vec models was almost the same with and without SVD dimensionality reduction, as evaluated by the accuracy matrix we used earlier. This suggests that SVD may not be necessary for improving the performance of pre-trained Word2Vec models on text datasets.

**[With Kmeans]** We also evaluated the effectiveness of k-means clustering in reducing computation time when analyzing large text datasets. Our results showed that k-means clustering did reduce computation time, but the variation was not very high.

Moreover, same behavior we had seen with our best bag of words model i.e., tf-idf.