Data Science Applications  
Classification Assignment

short line

Abdelrahman Ibrahim

Lamis Sayed

Minah Ghanem

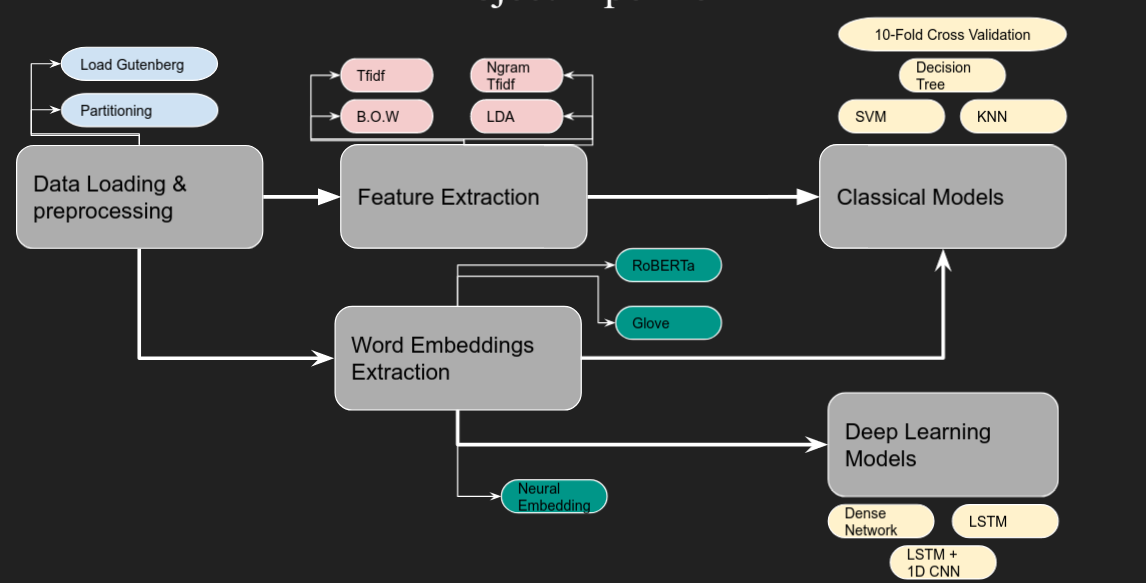
Mohammed El Namory

29th May, 2021

# Introduction

In this assignment we have classified different samples of five different books from the Gutenberg digital books. Our problem is focused on classifying text into five different categories or authors: **milton-paradise , shakespeare-caesar, melville-moby\_dick, chesterton-brown, whitman-leaves**

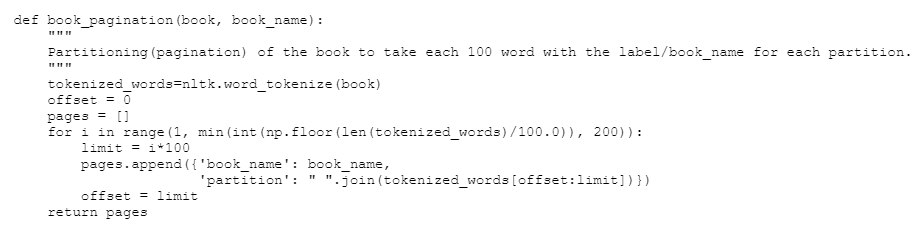
Through the whole process of our task, we have passed through multiple steps which will be introduced in the next pages:

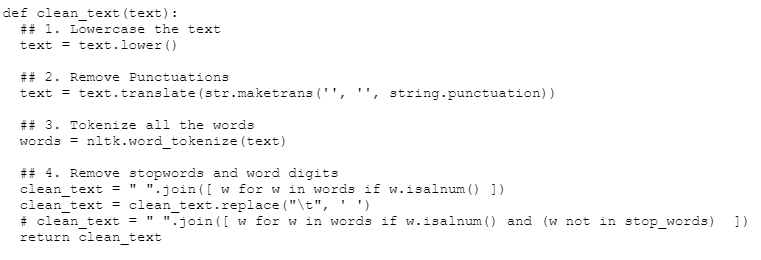


# Data Preprocessing

## Main Functions:

**Book\_pagination** is used to make partitions out of the text. It splits each book into 200 different partition that each one of it consists of 150 word.

**Clean\_text** is used for cleaning by ( lowering text - removing punctuations - keeping only alphanumeric tokens - removing any tabs - removing stop words )



**After the cleaning stage is finished** our data is transformed into a dataframe to be easily dealt with. And splitted randomly into training and testing datasets with a percentage of 80% to 20% respectively

# Feature Extraction

In the feature extraction phase, we have tried multiple methods and compared their results:

## CountVectorizer Features:

It is a simple way of building a vocabulary of known words. The feature output of that method is an encoded vector with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.

## CountVectorizer Using n-grams:

The same idea is implemented using n-grams. Which is simply a sequence of N words.instead of getting the frequency of one word, the same method is implemented on N sequence of words.

## TF-IDF Features:

In this method we have calculated the Term Frequency – Inverse Document (TF-IDF)

Which consists of **Term Frequency:** This summarizes how often a given word appears within a document. **And the Inverse Document Frequency:** This downscales words that appear a lot across documents.

It overcomes the issue of the simple count of having large counts for words that do not have meaningful insights, instead it gives more weight for words that matter.

## 

## LDA Features:

Latent Dirichlet Allocation ( LDA ) is an algorithm to extract topic modeling from large documents. We Made a vector out of the LDA model to use it as input for the supervised classifier. The feature vector for each partition consists of a set of probabilities of all the topics. We used the soft prediction of the topic classification model as an input feature for the supervised classifier

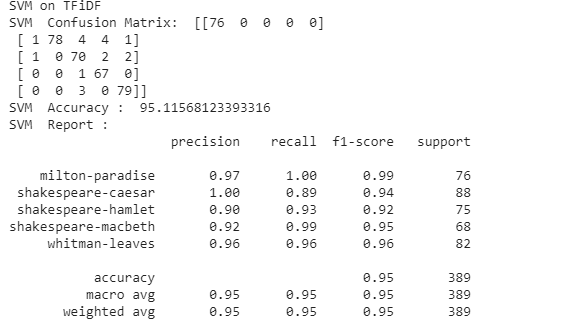
# Classical Models

We used 3 different classifiers algorithms SVM, Decision Tree and K-nearest neighbors.

* **Support Vector Machine(SVM):**

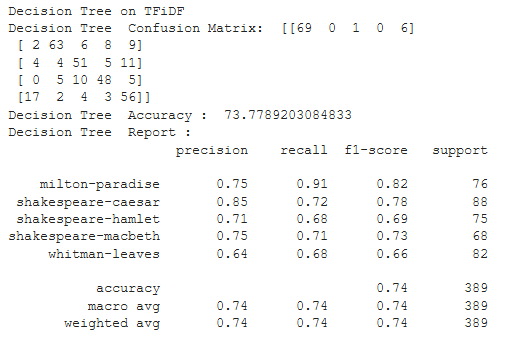
SVM is a relatively simple Supervised Machine Learning Algorithm used for classification and/or regression. It is more preferred for classification. In text classification, It determines the best decision boundary between vectors that belong to a given category and vectors that do not belong to it. So, the texts have to be transformed into vectors. We applied it on TFiDF, BOW and n-grams.

**TDiDF Results:**



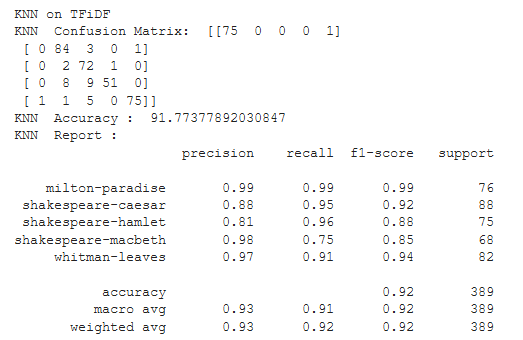
* **Decision Tree:**

A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label.

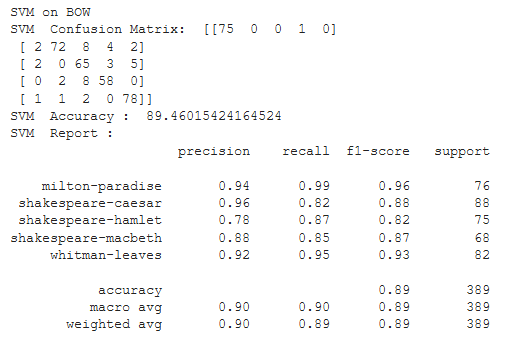


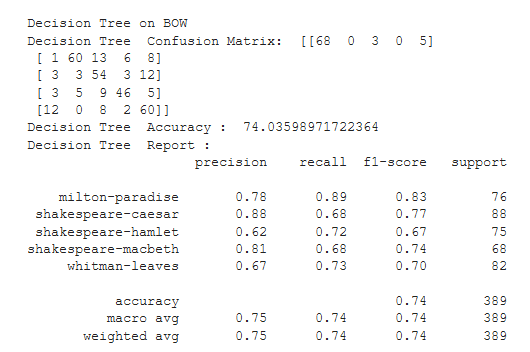
* **K-nearest neigbours(KNN):**

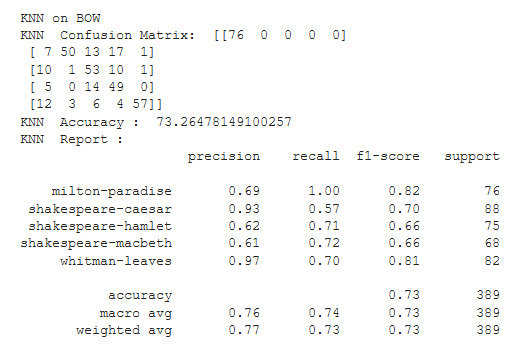
The closeness or proximity amongst samples of data determines their neighborhood and this is done by calculating the distance between points. The K in KNN indicates the number of categories that the data will be classified to. We use K=15 because as we increase the number of classes the classification become easier so the accuracy increases.



**BOW Results**



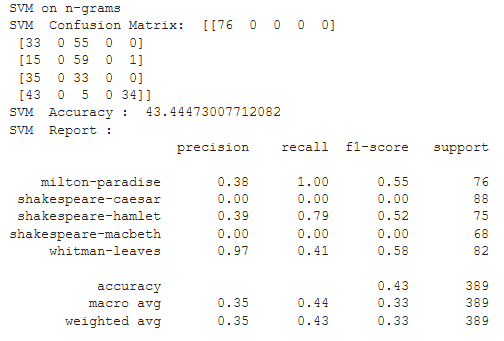


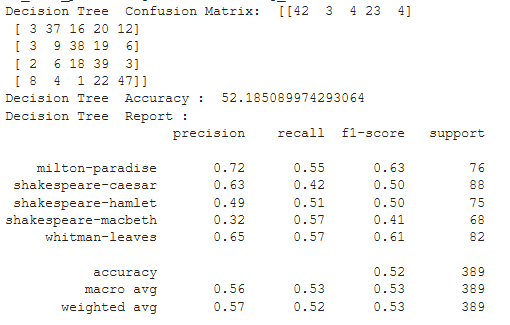


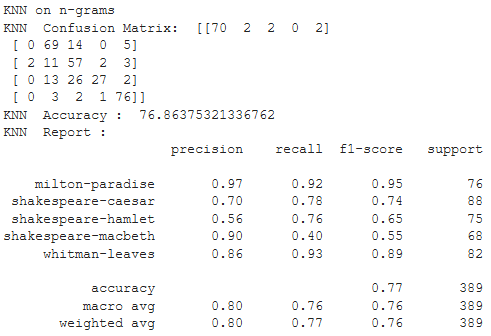
**Bi-grams Results:**

In Bi-grams we used TFiDf feature extraction. It produces accuracy better than

the BOW

****

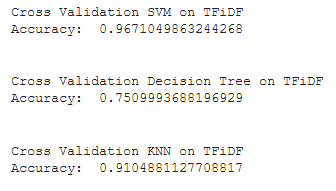
****

****

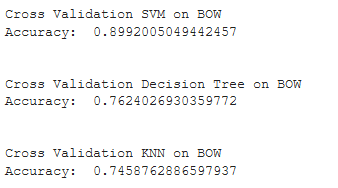
# Cross-Validation k-fold:

We split the data-set into k number of subsets(known as folds) then we perform training on the all the subsets but leave one(k-1) subset for the evaluation of the trained model. In this method, we iterate k times with a different subset reserved for testing purpose each time. We 10 folds

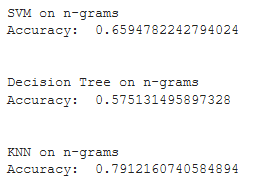
**TFiDF Results:**



**BOW Results:**

****

**N-grams Results:**

****

# 

# 

# 

# 

# 

# 

# 

# 

# Word Cloud Representation:

**We have made a word cloud representation of our corpus to visualize our input.**



# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

# 

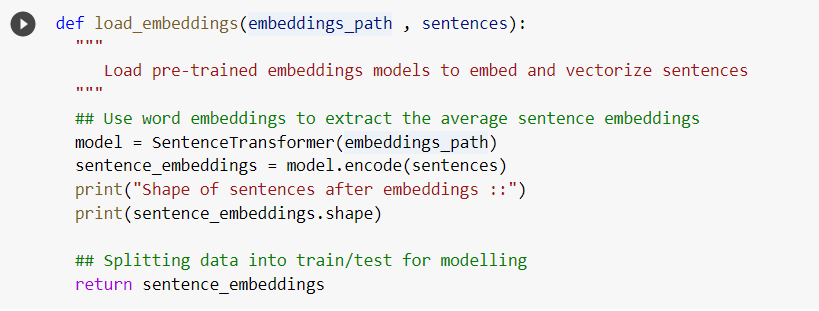
# 

# 

# Word Embeddings Extraction:

## 

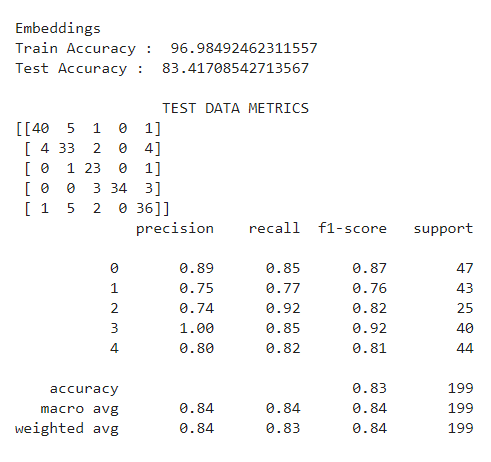
We used the function load\_embeddings to load pertained models to embed and vectorize sentences



## GLoVe

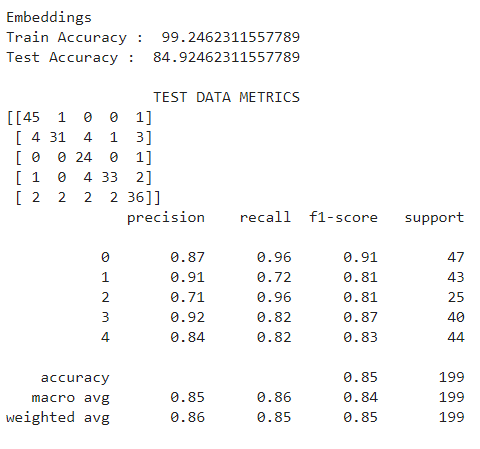
GLoVe: global vector for word representation is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space.

We used a pretrained model from glove “ average\_word\_embeddings\_glove.6B.300d” which is trained on 6 billion tokens and has a feature vector of a length of 300.



## RoBERTa

## RoBERTa is a language based model based on BERT. It improves on Bidirectional Encoder Representations from Transformers, or BERT, the self-supervised method released by Google in 2018.



# 

# 

# 

# 

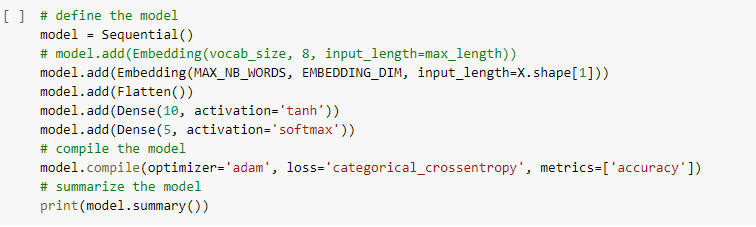
# Deep Learning Models

**Embedding Layer**

* In each DL model we used; we added an embedding layer for our architecture before passing it to the deep neural network.
* The aim of using a neural embedding layer is to build the context vector and learn more about the semantic and syntactic meaning of each partition sequence passed to neural network.

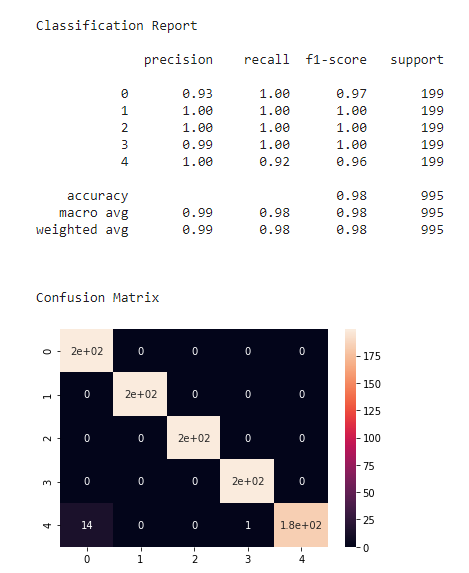
## Model-1

* + Our first deep learning model is not so deep, just 10 units of dense neural network with a softmax layer.



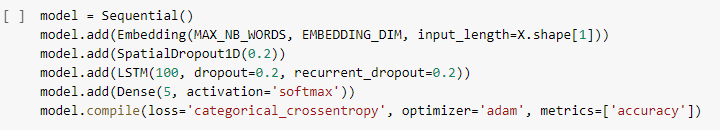
#### Error Analysis:

#### 

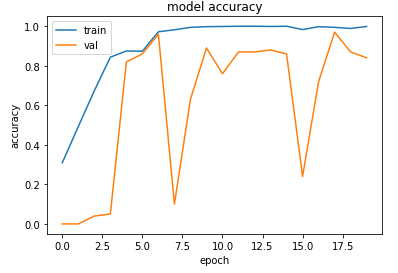


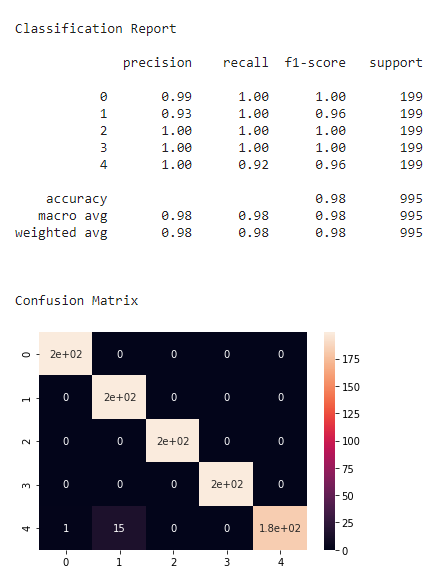
## Model-2

* + Here we used a new RNN-based model. We used the LSTM unit to deal with the sequence data better than the dense network which is widely used in the NLP work flow.



#### Error Analysis:





## Model-3

* + Here we used a more complex model leveraging two types of neural networks LSTM + CNN. We used 1D CNN to pass on the sequence and treat it as an image and pass a filter on the text sequence to highlight the important parts in the text.

#### 

#### 

#### 

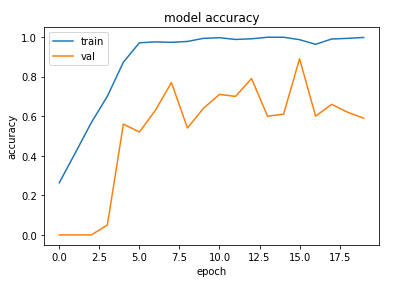
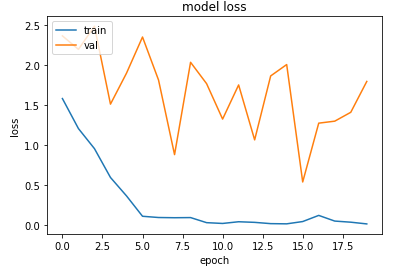
#### 

#### 

#### 

#### **Error Analysis:**

# 



# 

# 

# 

# 

# 

# 

# 

# 

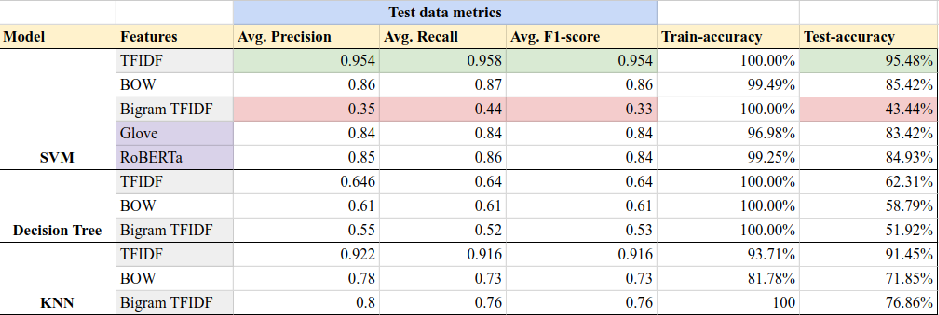
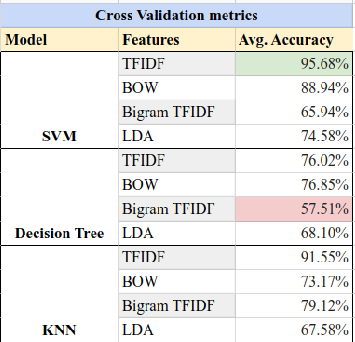
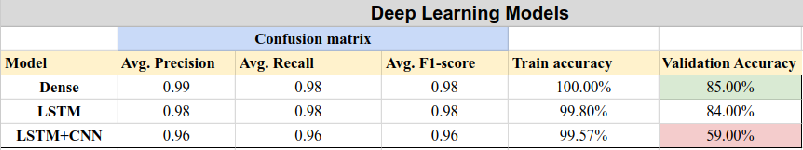
# 

# 

# 

# 

# Comparing Results



# References:

* Results Sheet: <https://docs.google.com/spreadsheets/d/13BLKEPEv0fWEbEFswhIal5dQV6FNDevyl27sSdOih8g/edit?usp=sharing>
* <https://towardsdatascience.com/unsupervised-nlp-topic-models-as-a-supervised-learning-input-cf8ee9e5cf28>
* <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html>
* <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html>
* <https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html>
* <https://www.machinelearningplus.com/nlp/topic-modeling-gensim-python/>
* <https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/>
* <https://realpython.com/python-keras-text-classification/#convolutional-neural-networks-cnn>
* <https://machinelearningmastery.com/sequence-classification-lstm-recurrent-neural-networks-python-keras/>
* Lecture notes from Dr. Arya Rahgozar and Notebook samples for visualization and loading the pre-trained models

short dash