**Group members:**

* Andrew Adel Labib.
* Hussien Amin Abdelhafez.
* Phoebe Thabit Wadea.
* Sandy Adel Latef.

**Abstract**

Text classification is one of the major tasks of AI in general, NLP in particular. Having five Gutenberg books, this report discusses the methodologies and models with different transformation techniques that have been applied to reach the best accuracy that the champion model achieves by correctly classifying unseen text to the corresponding book.

**Introduction**

Turning written material into useful information and insights is challenging because text is unstructured and not organized in a standardized way. The general goal is to produce categorizations and predictions from the text, compare the results, evaluate the benefits and limitations of different approaches. To achieve this, we will need to implement the approaches, determine the accuracy of each model, and select the most successful one. Python, with its many libraries, is well suited as a programming language for tackling such problems due to its text processing capabilities.

In our task, we perform these modern classifications techniques on some famous fiction novels which made us easily determine the style of each author later from his future writings thus leading to a big revolution in literature field.

**Dataset** The Gutenberg dataset represents a corpus of over 60,000 book texts, their authors and titles. The data has been scraped from the Project Gutenberg website using a custom script to parse all bookshelves. we have taken five different samples of Gutenberg digital books that are of five different authors, that we think are of fiction novels. **The books are**

* The enhanced April.
* Middle March.
* Twenty years after.
* Great expectations.
* Aliece’s adventures.

**Data Preparation:**

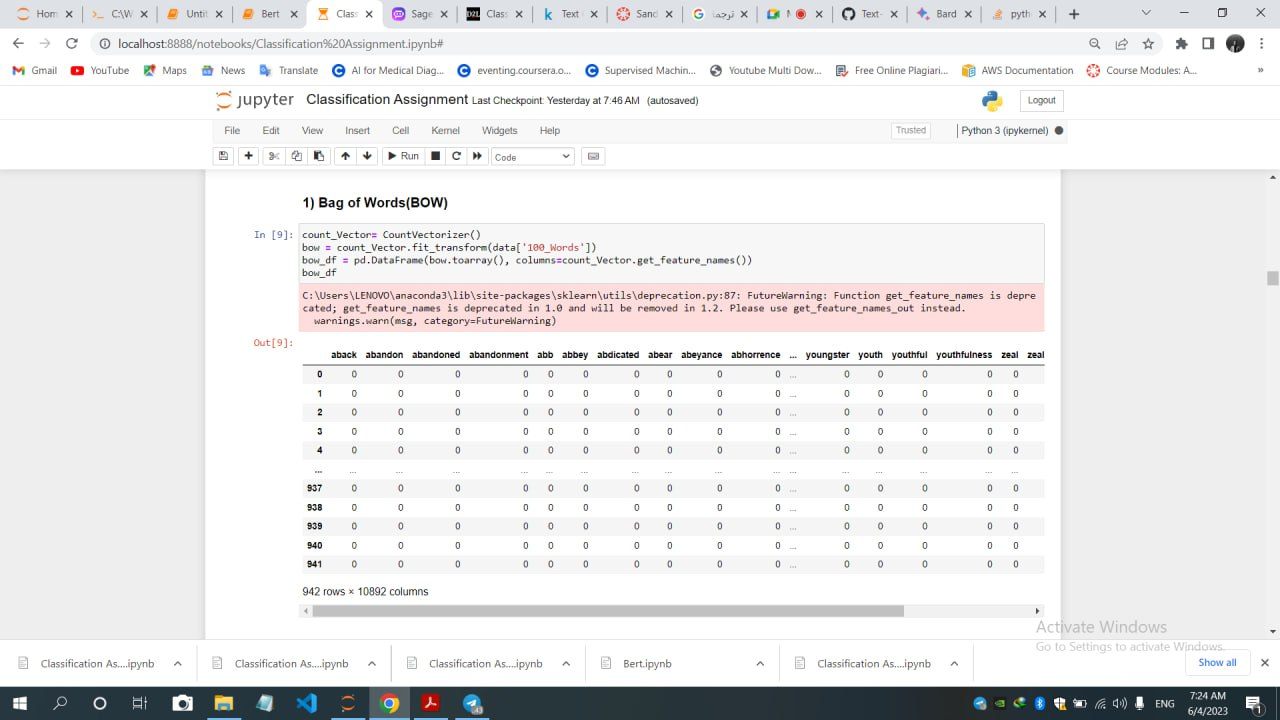
After reading the books, we did some preprocessing methods:

* **Text cleaning** – converting text into lowercase and cleaning it by removing unsubstantial parts such as HTML tags, symbols, or sometimes numbers, ensures that words with less than 3 characters like “ye” are removed as they have no special meaning (Removing non-alphabetic characters)
* **Stop words removal** – excluding some common words that don’t provide useful information.
* [**Lemmatization and stemming**](https://www.baeldung.com/cs/stemming-vs-lemmatization) – simplifying words to their base or root form by following some rules from dictionaries, cutting off common prefixes and suffixes, and similar.
* **Tokenization** – when we separate cleaned text into smaller units, such as words, characters, or some combinations of them.
* A screenshot of a computer

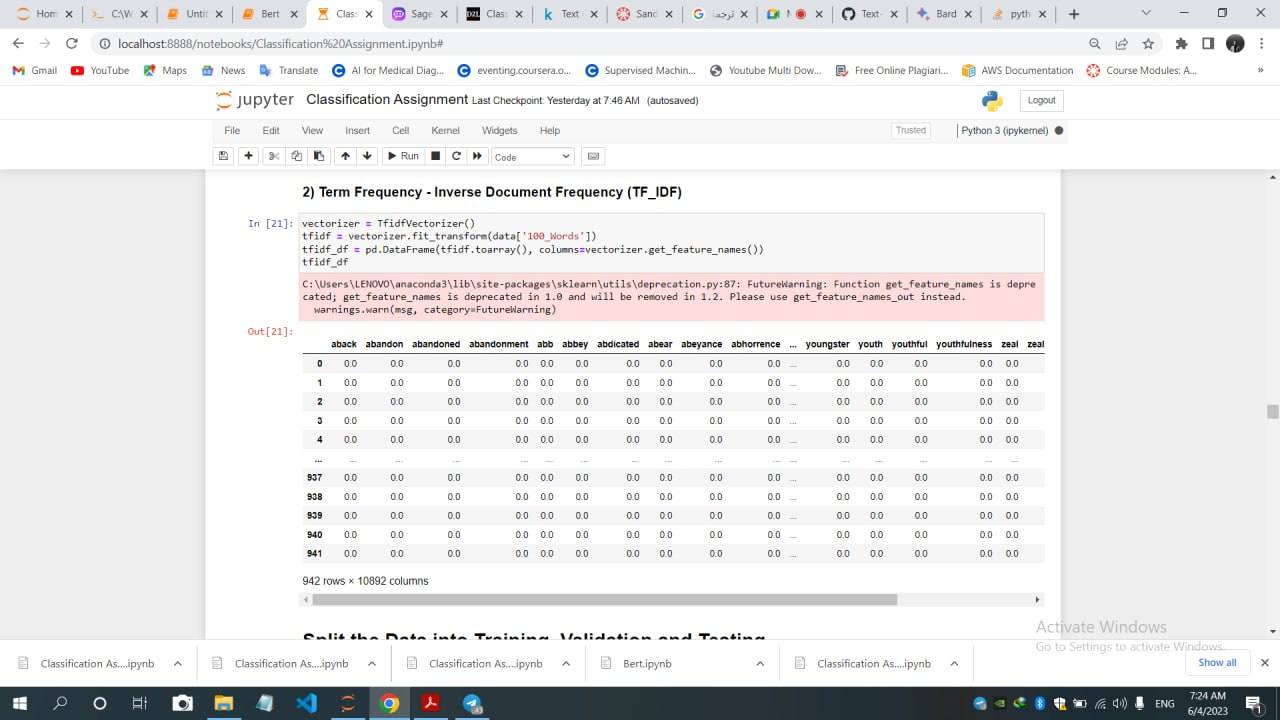
  Description automatically generated with low confidence**Data Partitioning** – partition each book into 200 documents, each document is a 100-word record.

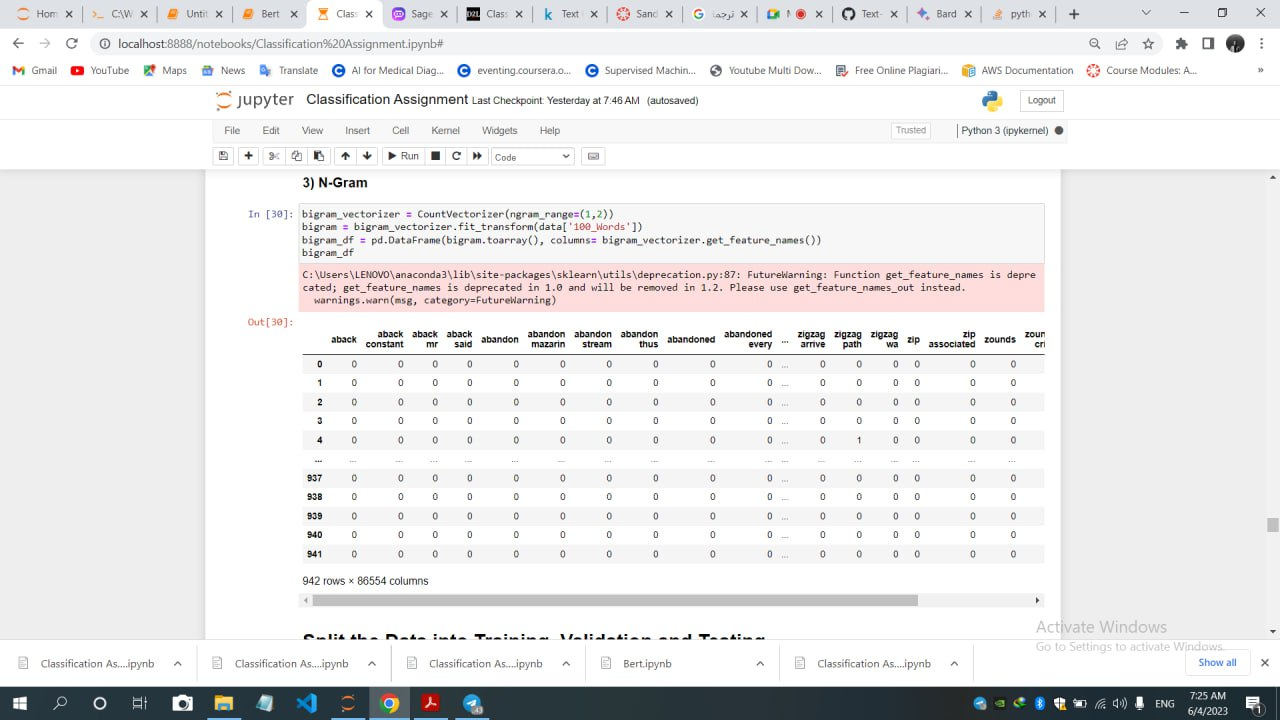
**Feature engineering:**

* **BOW**: one type of transformation that count of the total occurrences of most frequently used words, to convert the text into numbers so that the algorithm can deal with it.



* **TF\_IDF**:  term frequency-inverse document frequency is a measure for estimating the importance of words in a document among a collection of documents.



* **N-Gram**: may be used to create probabilistic language models called n-gram models, **which** predict the occurrence of a word based on its N – 1 previous word.

**Classification:**

For each technique of the above, these following models are trained and tested:

* SVM.
* Random Forest.

# Naïve Bayes.

# K-Nearest Neighbour.

# XG-Boost.

# (SGD)Stochastic Gradient Descent.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Testing accuracy** | | | | | | |
|  | **SVM** | **Random**  **forest** | Naïve Bayes | **K-NN** | **XG-Boost** | **SGD** |
| **BOW** | 0.9788 | 0.9524 | 0.9841 | 0.8942 | 0.9471 | 0.9735 |
| **TF\_IDF** | 0.9788 | 0.9683 | 0.9735 | 0.9577 | 0.9365 | 0.9788 |
| **N-gram** | 0.9683 | 0.9683 | 0.9841 | 0.8942 | 0.9471 | 0.9788 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Cross-Validation accuracy** | | | | | | |
|  | **SVM** | **Random**  **forest** | Naïve Bayes | **K-NN** | **XG-Boost** | **SGD** |
| **BOW** | 0.9894 | 0.9894 | 0.9947 | 0.9365 | 0.9577 | 0.9947 |
| **TF\_IDF** | 0.9947 | 0.9788 | 0.9894 | 0.9683 | 0.9788 | 0.9947 |
| **N-gram** | 0.9894 | 0.9894 | 0.9947 | 0.9206 | 0.9630 | 0.9735 |

**Champion model:**

According to the previous tables of calculating the accuracy for both testing and validation,

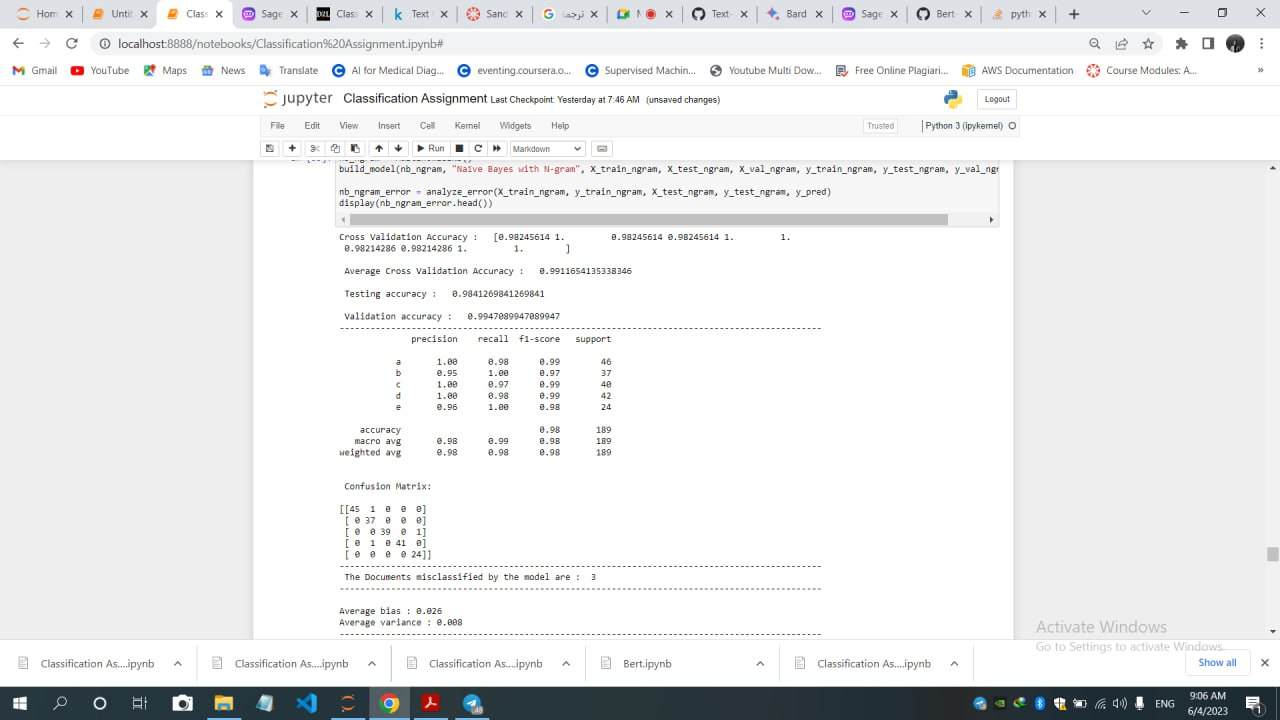
# The champion model is: Naïve Bayes based on (N-gram), which gives us the accuracy of 0.9841 (The same accuracy of Naïve Bayes based on BOW) , but the average variance of this model is lower (0.008 (N-gram) < 0.013(BOW)).

# 

**Error analysis:**

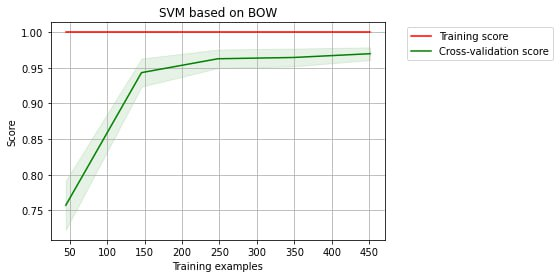
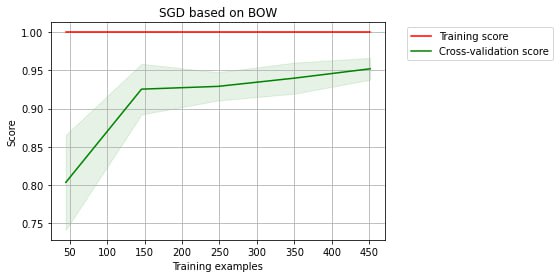
Error analysis is an important step in improving the performance of text classification models. It involves analyzing the errors made by the model on the validation or test set and identifying the patterns or characteristics of the misclassified instances. Based on the analysis, we can identify the areas where the model needs improvement and take appropriate actions to address them.  
Here are some common techniques We used in error analysis for our text classification problem:

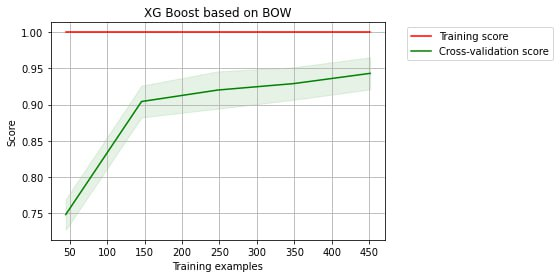
* **Confusion matrix**: A confusion matrix can be used to visualize the number of true positives, true negatives, false positives, and false negatives. This can help identify which classes the model is struggling with and where it is making the most errors.
* **Precision, recall, and F1 score**: These metrics can be used to evaluate the performance of the model on a per-class basis. This can help identify which classes the model is struggling with and where it needs improvement.
* **Visualizing misclassified instances**: Misclassified instances can be visualized to identify patterns or characteristics that may be causing the errors.

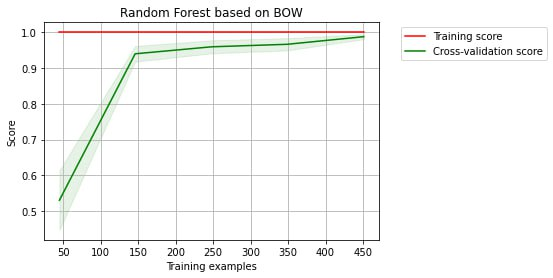
****

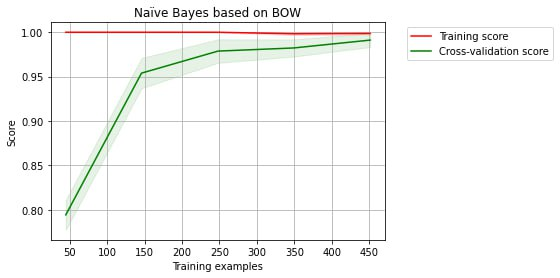
**Testing results :**

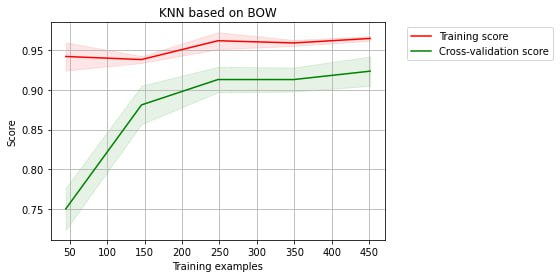
**Learning curve:** a learning curve is a plot of the model's performance as a function of the amount of training data. It is used to diagnose whether the model has a high bias or high variance. A high bias model is one that is too simple to capture the underlying patterns in the data, resulting in underfitting. A high variance model is one that is too complex and overfits the training data, resulting in poor generalization performance. A learning curve can help determine whether the model would benefit from more training data or if it is too complex and needs to be simplified.

******BOW:**

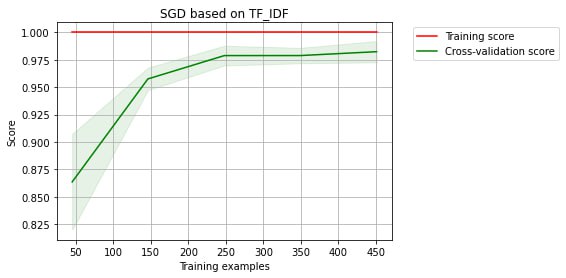
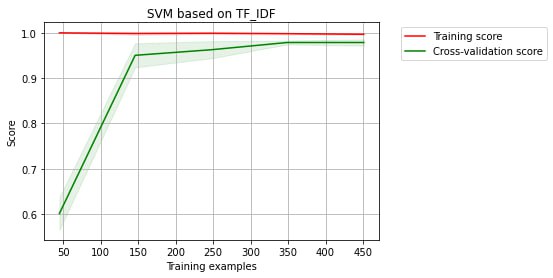


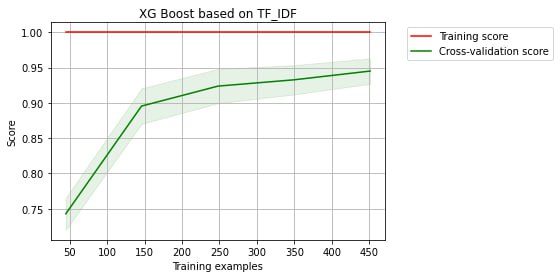
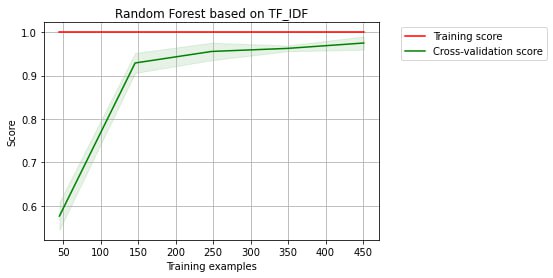


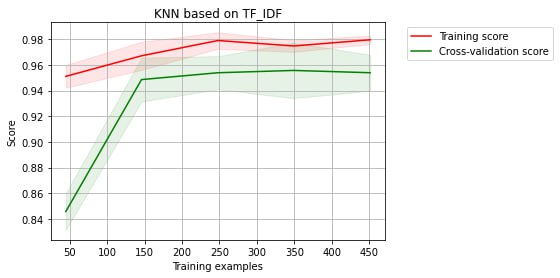
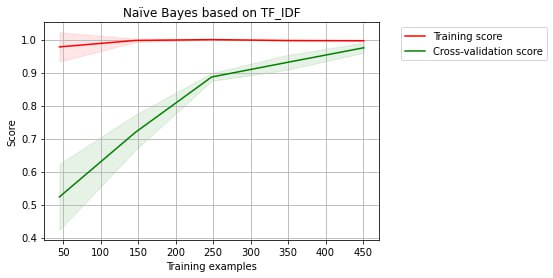




**TF-IDF:**





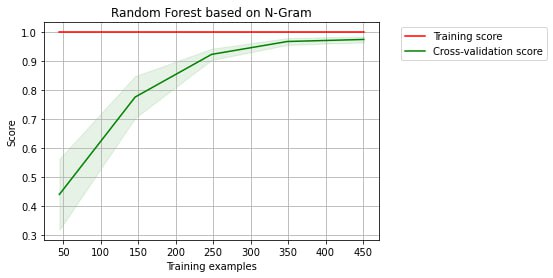


**N-Gram:**

A graph with green and red lines

Description automatically generated with low confidenceA picture containing line, text, plot, diagram

Description automatically generated

 A graph with a green line

Description automatically generated with low confidence

A picture containing text, line, plot, diagram

Description automatically generated A graph with red and green lines

Description automatically generated with low confidence