Assignment 3

Name: Harsha Chandwani

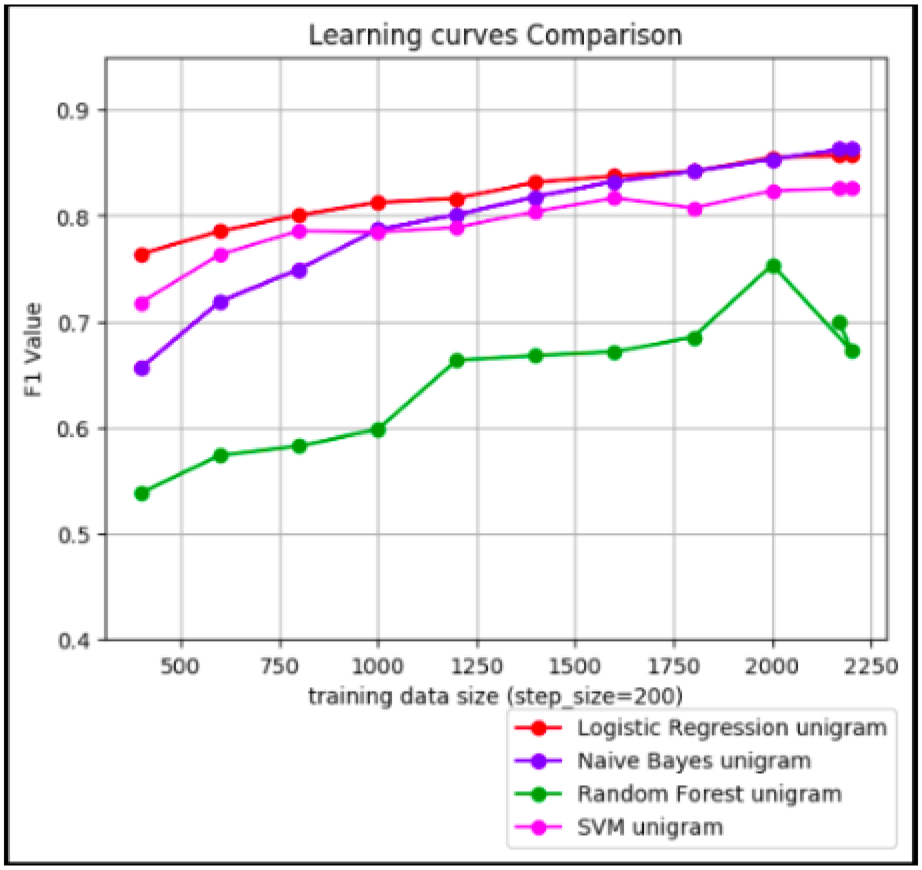
SBU ID: 111481387

**Part A: Table showing the precision, recall and F1 score for Unigram and Bigram baselines for each of the four methods i.e. Naïve Bayes, Random Forest, SVM, Logistic Regression**

| **Algorithm** | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| NB, UB | 0.896916 | 0.854615 | 0.863727 |
| NB, BB | 0.896075 | 0.809464 | 0.821813 |
| RF, UB | 0.732685 | 0.69038 | 0.679321 |
| RF, BB | 0.765069 | 0.687324 | 0.678349 |
| SVM, UB | 0.828992 | 0.824525 | 0.825892 |
| SVM, BB | 0.835914 | 0.824363 | 0.827275 |
| LR, UB | 0.862466 | 0.854009 | 0.856882 |
| LR, BB | 0.864179 | 0.851758 | 0.855306 |

**Part B: Learning curve plot**

b. A learning curve (LC) result, showing the performance of each classifier with just the unigram representation. The learning curve is the plot of the performance of the classifier (F1 on the y-axis) on the dev fold, when trained on different amounts of training data (size of training data on the x-axis).



**Part C: Explanation of the findings**

So, based on the above graph plot, we can see, that the algorithms perform in the following order from best to worst performance.

1. Naive Bayes
2. Logistic Regression
3. SVM
4. Random Forest

The two bold assumptions that Naïve Bayes makes, contribute to its better performance in terms of text classification

1.The probability of occurrence of any word given the class label, is independent of the probability of occurrence of any other word, given that label.

2.The probability of occurrence of a word in a document, is independent of the location of that word within the document.

Also when the size of data is less, general algorithms like Naïve Bayes perform better than SVM and Logistic regression(link)

Moreover, to our surprise, here we see that the performance is better with unigrams than the bigrams. Bigrams provide better context information than unigrams but also need a considerable amount of training dataset so that the frequency of the bigrams does not decrease, and we get better performance results. The more sparse your data set, the worse you can model it. It is for this reason that even if we have a higher-order n-gram model, and it contains more information about the context, it cannot generalize and classify the new test data sets  to high accuracy because it has seen a fewer instances at the time of its training. Hence, the overfitting. To get better results with bigrams, we need larger training data.

https://www.quora.com/Why-does-Naive-Bayes-classifier-works-so-well-with-text-data

https://stackoverflow.com/questions/36542993/when-are-uni-grams-more-suitable-than-bi-grams-or-higher-n-grams

https://ai.stanford.edu/~ang/papers/nips01-discriminativegenerative.pdf

**Part D**

**: Exploring different configurations and finding the best model**

**Based on Part 1, we figure out that the model that worked best was NB, Unigram. Tried different configurations with this as the base model.**

| Model Configuration | Precision | Recall | F-1 |
|---|---|---|---|
| NB : Unigram: CountVectorizer: Stemmer: No Stopper | 0.91573 | 0.910966 | 0.912922 |
| NB: Unigram: CountVectorizer: Stemmer: Stopper | 0.914559 | 0.908966 | 0.911201 |
| NB : Unigram: CountVectorizer: Stemmer: No Stopper: Feature:SelectPercentile = 80 | 0.916529 | 0.910954 | 0.91303 |
| NB: Unigram: TFIDFVectorizer: Stemmer: No Stopper | 0.912448 | 0.900716 | 0.904641 |
| NB: Unigram: TFIDFVectorizer: Stopper: No Stemmer | 0.913611 | 0.90221 | 0.906122 |
| **NB: Unigram: TFIDFVectorizer: No Stemmer: No Stopper** | **0.921745** | **0.908723** | **0.913141** |
| NB : Unigram: TFIDFVectorizer: No Stemmer: No Stopper: Feature:SelectPercentile = 80 | 0.918985 | 0.904723 | 0.909483 |
| NB : Unigram: TFIDFVectorizer: No Stemmer: Stopper: Feature:SelectPercentile = 80 | 0.919585 | 0.907466 | 0.91162 |

**Explanations:**

Out of the above 8 models, model with the configuration **NB: Unigram: TFIDFVectorizer: No Stemmer: No Stopper** gives the best performance.

The reason why no stemming gives better performance is because sometimes stemming completely changes the results of classification, because of incorrect understanding of the context as explained here. phrase "runs today" may refer to a runner, while "long running" may be about phone battery lifetime. In this case stemming makes classification worse, not better.

Stop-words are words like 'a, the, an, it'. Not removing stop-words sometimes increases the accuracy of the classification. It helps when we classify based on topics that are well defined by nouns and adjectives.

It is possible that certain stop words are really more associated with certain topics. Perhaps they occur in common phrases associated with the topic. You could check feature importance from the weights and see which of the stop words were being used most heavily by the model.

https://stackoverflow.com/questions/22603332/stemming-in-text-classification-degrades-accuracy

https://stats.stackexchange.com/questions/280441/stopword-removal-suprisingly-decreases-accuracy-of-naive-bayes-model