Big Data Project
Spark Streaming for Machine Learning

Team ID: BD_171_177_185_391
Guhan K PES1UG19CS171
Hanuraag Baskaran PES1UG19CS177
Harshita Vidapanakal PES1UG19CS185
Rohan M PES1UG19CS391

Design Details:

Before we specify the design details, we need to define the vectorisers used:

1) **Tokenizer**: Splits up the given sentence into smaller words, such as individual words or phrases.

2) **Stopwords Remover** using the spark.ml NLTK(Natural Language Toolkit), removes most used phrases or words before processing.

3) **CountVectorizer**: Transforms the given text into vectors based on the frequency of each word that occurs in the given document

4) **IDF with HashingTF**: This is used to assign weights to feature vectors, where less frequently used words are upweighted, and more frequently used words are down weighted.

The first step in this process is using one-hot encoding on the spam-ham column to mark one as spam and zero as ham. Following this, we use Tokenizer to split the sentence in the message column into individual words. We then use the Stopword remover to remove the most commonly occurring phrases or words. After removing these words, we then use HashingTF to convert these filtered set of words to feature vectors. We then use TF-IDF to apply weights to the words, where less commonly used words will be assigned higher weights, and more frequently occurring words will be assigned lower weights.
Following pre-processing, we get the feature vectors and feature sets which are accordingly weighted. We pass the obtained data into 4 models, which are as follows:

K-means clustering is a clustering algorithm that classifies the given set of data points into k clusters, which are predefined before training. Clustering is the process of grouping data samples into clusters based on similarities in specific features they share. The idea behind k means is that we have to add k points to the given data. The point, in this case, is known as a Centroid and will try and center itself in the middle of each cluster. The moment the centroids stops moving, the clustering algorithm stops immediately.

Multinomial Naïve-Bayes is a probabilistic learning algorithm that works on the principle of the Bayes theorem, which can predict the probability of a class occurring again based on the probability of other classes occurring. The algorithm works by creating a frequency table of the given training set, find the probabilities of each class occurring in the table, creating a likelihood table, and finally calculating the posterior probability of each class occurring using the Naïve-Bayes theorem. The highest probability of all the calculated probabilities will be the final outcome.

Bernoulli Naïve-Bayes works on the Bernoulli classifier, which gives only a binary output, hence requiring the feature vectors to be binary valued. This largely simplifies classification massively, and has the added benefit of increased accuracy.

Stochastic Gradient Descent Classifier(SGD Classifier) is a method of classification that uses only one point while changing weights, instead of taking into account the entire data set. This reduces the computation time by a huge margin, especially with huge datasets.

Incremental learning has been applied to the above models for the best results.

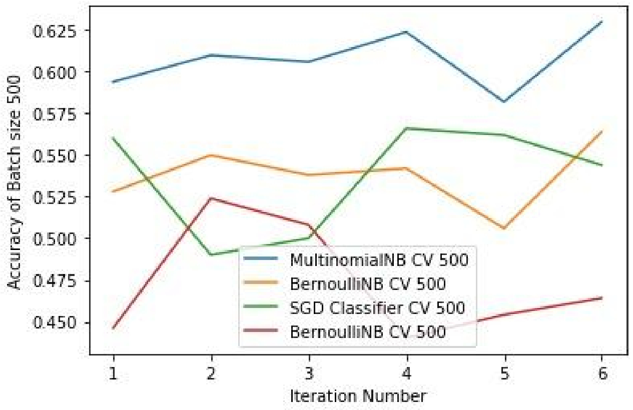
Reasons for Design Decisions:

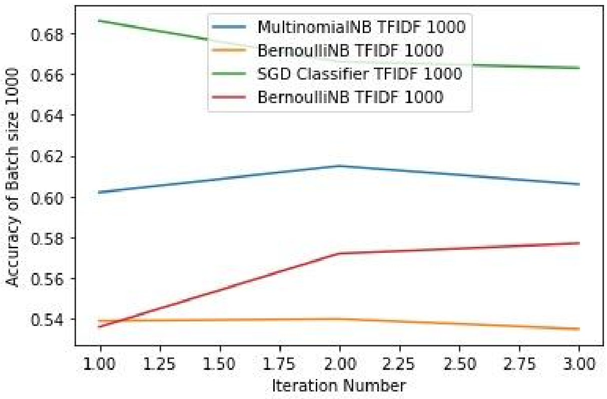
We have chosen TF-IDF and CountVectorizer over Word2Vec because the former gives much better accuracy than using Word2Vec for our specific dataset, especially with spam-ham detection. We also obtained negative vectors for Word2Vec, hence we were unable to effectively use it to map out feature vectors. These results have been quantified in the graphs given below

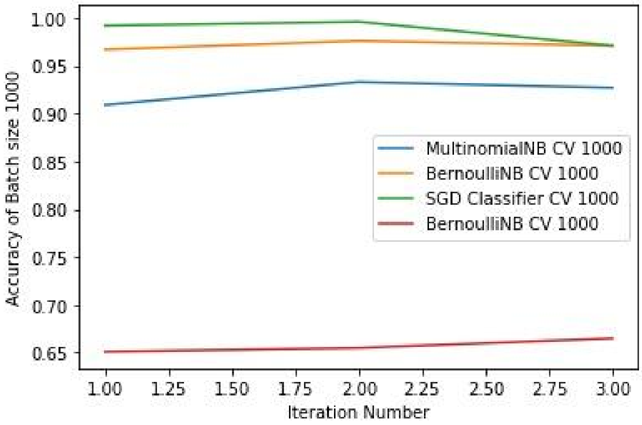
Takeaways from this project:

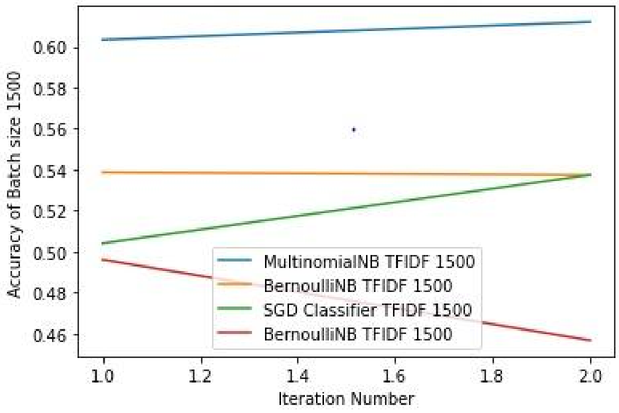
We have learnt the following from this project:

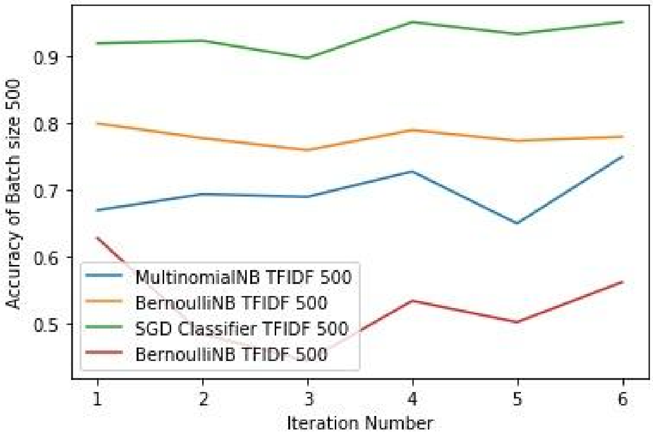
1) We obtain better accuracy if we consider a higher number of features. When we made our predictions with a batch of 500 with 20 feature sets, we obtained an accuracy of about 50%. With increasing batch size to 2000 with 50 features(features and batch size gradually increasing), we got a higher accuracy of 60-70% accuracy. But the highest accuracy was with a batch size of about 1000-1500 with 100 features, we got the highest accuracy of about 90%. This proves accuracy relies more on the feature size than the batch size.
2) TF-IDF is one of the better algorithms to use, especially for spam/ham, compared to other algorithms such as Word2Vec
3) We conclude that SGD Classifier is the best among all the classifiers, providing a consistent accuracy at any given batch size
4) We infer that the K means clustering isn’t the ideal classifier considering it has the least amount of accuracy.

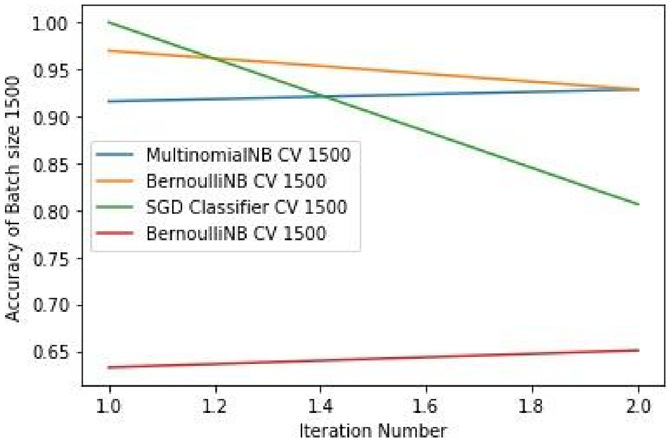

0.625
0.600
0.575
0.550
0.525
0.500
0.475
0.450
Accuracy of Batch size 500
1
2
3
4
5
6
Iteration Number
MultinomialNB CV 500
BernoulliNB CV 500
SGD Classifier CV 500
BernoulliNB CV 500


0.68
0.66
0.64
0.62
0.60
0.58
0.56
0.54
Accuracy of Batch size 1000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Iteration Number
MultinomialNB TFIDF 1000
BernoulliNB TFIDF 1000
SGD Classifier TFIDF 1000
BernoulliNB TFIDF 1000


1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
Accuracy of Batch size 1000
1.00
1.25
1.50
1.75
2.00
2.25
2.50
2.75
3.00
Iteration Number
MultinomialNB CV 1000
BernoulliNB CV 1000
SGD Classifier CV 1000
BernoulliNB CV 1000


0.60
0.58
0.56
0.54
0.52
0.50
0.48
0.46
Accuracy of Batch size 1500
1.0
1.2
1.4
1.6
1.8
2.0
Iteration Number
MultinomialNB TFIDF 1500
BernoulliNB TFIDF 1500
SGD Classifier TFIDF 1500
BernoulliNB TFIDF 1500


0.9
0.8
0.7
0.6
0.5
Accuracy of Batch size 500
1
2
3
4
5
6
Iteration Number
MultinomialNB TFIDF 500
BernoulliNB TFIDF 500
SGD Classifier TFIDF 500
BernoulliNB TFIDF 500


1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
Accuracy of Batch size 1500
1.0
1.2
1.4
1.6
1.8
2.0
Iteration Number
MultinomialNB CV 1500
BernoulliNB CV 1500
SGD Classifier CV 1500
BernoulliNB CV 1500