Machine Learning with Streaming Spark

Project Title: Ham – Spam Classification

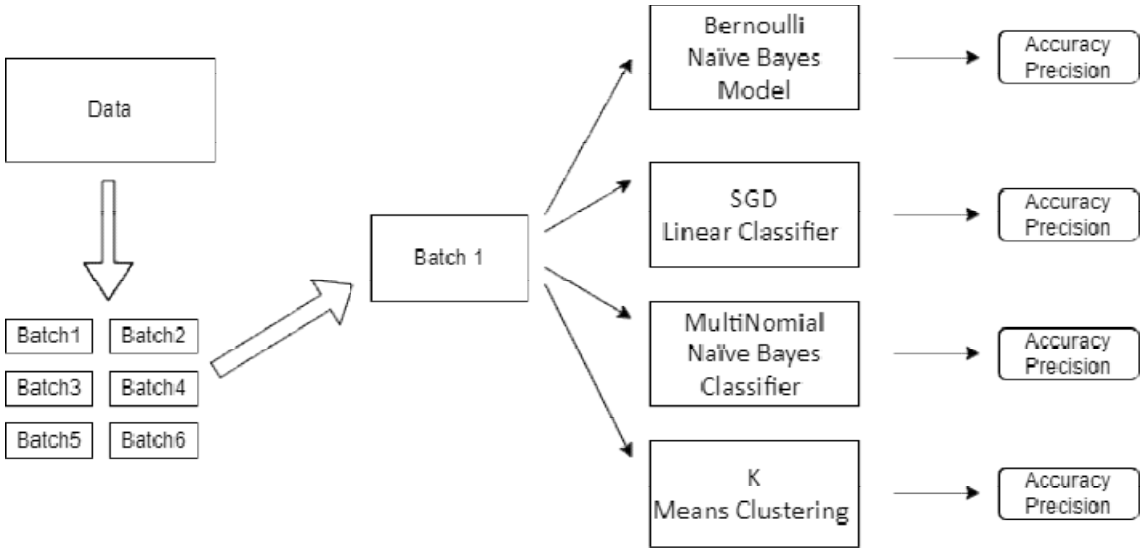
Design Details: We have divided our project into 5 parts in total which includes the preprocessing, the 3 classifiers and a clustering model.

- We have started by creating a session which has been used for processing of the each batch's data. And as the part of preprocessing we have formatted the data in such a way that it is apt for the building of the models.
- The first classification model chosen is the Bernoulli model. We have implemented it using the Bernoulli Naïve Bayes classifier.
- The second classification model chosen is the Linear Classifier. This model has been brought about by the use stochastic gradient descent.
- The third classification model chosen is the Naïve Bayes Classifier. We have implemented it using the multinomial Naïve bays classifier.
- The clustering has been brought about by the use of the K-means clustering. This has been done by considering the number of clusters as 2.

Design Implementation:

1. Preprocessing: The JSON formatted string that we stream in is divided into 3 different features, i.e., subject, message and spam/ham label. Each of these features is appended in a list and passed on to pre processing which has a pipeline implemented for hashing the textual data so that it can use to train the model. We first tokenize the strings, followed by removal of stop words, once this is done the string are clustered into smaller chunks of strings. HashingTF(Term Frequency) is the function used to hash each chunks of strings created in the previous step.
2. Bernoulli Naïve Bayes Model : The inbuilt sklearn model BernoulliNB is used to partially fit the model with each batch data. After each fit this model is saved as a pkl file inside the models directory so that it can be further accessed for training or predicting.
3. SGD Linear Classifier : Another one of the sklearn model SGDClassifier is used to partially built the model for linear classification with stochastic gradient descent.
4. MultiNomial Naïve Bayes Classifier : Sklearn's MultinomialNB() is used to partial build the model. The reason for choosing these models from sklearn is that all these can be iteratively trained.

5. K Means Clustering : MiniBatchMeans module of sklearn is used to do K means clustering. This includes 2 clusters as we have 2 labeled target values i.e., spam or ham.



The data being streamed in is being broken down into several batches of customizable length and these batches are being fed into various models individually and these models are being tested based upon their accuracy, precision and their error rate.

<u>Reason behind design decisions:</u> The reason behind the choice of implementation of the chosen designs is that opted classifier is much simpler and is very easy to use when compared to many other sophisticated models. These models can be trained by using much smaller train data and can predict the output with great accuracies. The naïve bayes classifier is predominantly preferred by many programmers for classification type of problems and it is tailor made fit for spam and ham classification. All these reasons motivated us to stick with our chosen design decision.

<u>Implementation and Working :</u> Let us now try and understand the detailed steps which were involved in the implementation of the code to obtain the results from the model.

1. As the base of our models we have created two files namely traingmodels.py and testingmodels.py these along with the provided stream.py are the necessary files which have been used by us.
2. The files trainingmodels.py and testingmodels.py have been used for the purpose of training and testing of the machine learning models respectively.
3. The trainingmodels.py file creates the machine learning models. The models are created by training them with the provided train data. These models were created with the use of pickle and the models are created and stored with .pkl extension.

4. The testingmodels.py file makes use of the models which were created earlier and computes the accuracy and precision of the models.
5. The accuracy and precision of the computed data are thus written into the corresponding files which are present in testingmodels.py.
6. These files have been used in order to plot the graphs for the purpose of visualizations.

The steps involved in the execution of the program are:

1. By choosing the appropriate batch size as the parameter the stream.py file will be ran first prior to the others.
2. The dataset field in stream.py is set to train by commenting test and then the stream.py is run.
3. After the stream.py is all set and running the trainingmodel.py file would be run with the help of spark-submit.
4. After the completion of execution of the stream.py i.e. when the stream has completed all the batches and then the trainingmodels.py is stopped.
5. Now the stream.py is ran again but now changing the dataset field to test instead of train.
6. After the stream is up and running then testingmodels.py is ran again with the aid of spark-submit.
7. Penultimately the testingmodels.py is closed after all the batches of the stream.py have completed or the stream.py has achieved 100% streaming of the data.
8. Finally the accuracy and the precision values can be found written into the files named 'burnacc.txt', 'SGD.txt', 'Naïve.txt' and 'kmeans.txt'.

Observations and Results :

The following observations were achieved by the end of the project.

The first classification model which we worked upon was the Bernoulli Naïve Bayes Model. This model was trained and tested several times against varying batch sizes ranging from 100 to 300. The accuracy of this model is found to be increasing as the batch size increases which becomes approximately equal after reaching a particular point.

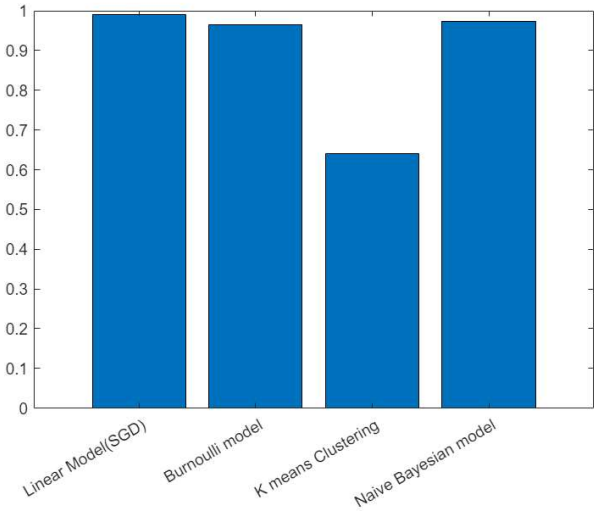
The second classification model was the SGD Linear Classifier. This model similar to the Bernoulli was trained and tested against the varying batch sizes. The insights observed from the results obtained from this model are that this model's behavior was somewhat similar to that of the prior model.

The third classification model was the MultiNomial Naïve Bayes Classifier. This model when trained and tested against varying batch sizes reached a point after which the further increase in the accuracy of the model was not possible.

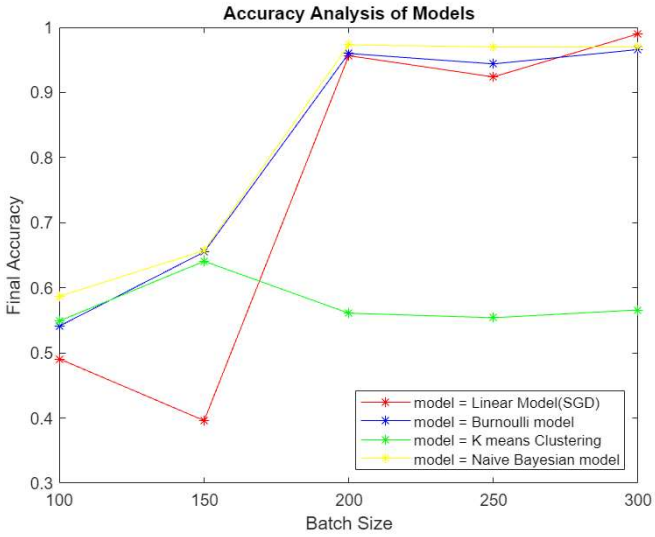
The clustering model was the K- Means Clustering model. This model like all the classification models was trained and tested against batch sizes ranging from 100 to 300. The results of this classifier weren't much promising and the accuracy obtained from this model was amongst lowest of the accuracies.

The following are the graphical visualizations of the above models :

The corresponding graph represents the variation of the accuracy of all the models with increase in the size of the batches provided as input. It is pretty much evident from the graph the all the classification models are performing much better than the clustering models. The graphs here represent that accuracy of the clustering models tend to reach stagnant point at batch size of 200. The following bar graph is the representation of the highest accuracy achieved by each of the models when compared to all the batches run for all the models.

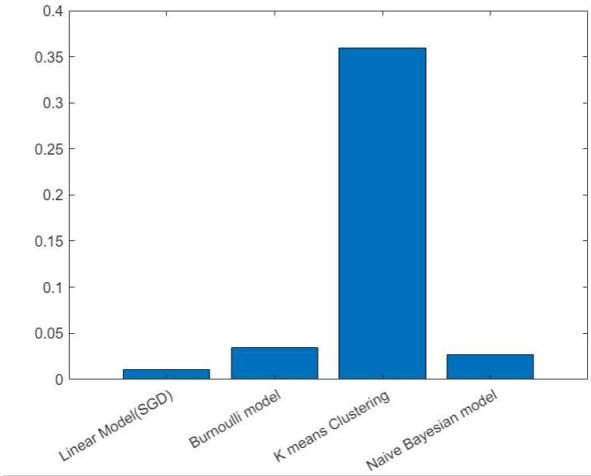


Maximum accuracy achieved by each of the models

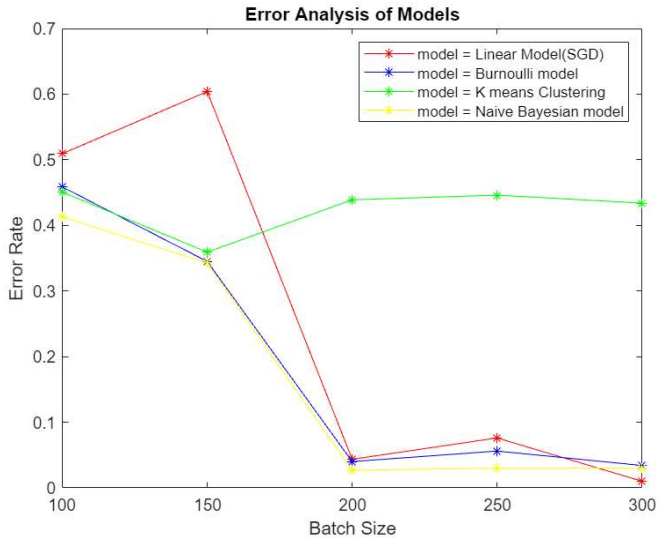


Final Accuracy vs Batch Size for all models

The following graph is the representation of the error rate achieved by each of the models with increase in the batch sizes. The graph clearly states that the error rate of k-means clustering is much larger when compared to other models. The bar graph represents min error attained by the each model when executed for all the batches.



Minimum Error achieved by each of the models



Error Rate vs Batch Size for all models

Results:

The results obtained from the project are as follows:

The classification models were much better than the clustering model. They were able to attain high accuracies unlike the clustering. The linear classifier outperformed all the others to achieve a high accuracy of 99% when ran over a batch size of 300. The other highest accuracies achieved by the models are 97% achieved by the naïve bayes classifier and 96.6% accuracy achieved by Bernoulli classifier when ran over batch size of 300. The best accuracy achieved by the K-Means clustering was only about 64.08% when the model was trained and tested over batch size of 150. Thus we conclude that SGD linear classifier had the best accuracy of them all.

Take away From the Project: By being a part of this project we are now pretty much familiar with the process of spark streaming and how the streamed data can be processed upon. By extending the spark streaming and integrating with machine learning we were able to find out how live streams of data can be processed in order to be classified as well as how meaningful insights could be drawn from them. This way we understood that the spark's main implementation which is to derive the approximate solution in a very short span of time.

Group Number : BD_246_409_453_574
Member 1: Lohith V PES1UG19CS246
Member 2: Sachin M S PES1UG19CS409
Member 3 : Shashank Yadav K PES1UG19CS453
Member 4 : Vishnu J G PES1UG19CS574