PES UNIVERSITY

UE19CS322 BIG DATA PROJECT REPORT

3.1 PROJECT TITLE: MACHINE LEARNING WITH SPARK MLLIB
Spam Dataset

Sahana Ramesh - PES1UG19CS413
Sonali Kadham - PES1UG19CS493
V M Dhruthi - PES1UG19CS550
Vaibhav Jamadagni - PES1UG19CS553

3.2 Design details

Preprocessing used :

1. **Tokenizer** We tokenized the data which converts the input to lowercase and then splits the text by whitespaces.
2. **Stop word removal** We removed all the stop words which are constantly repeated.
3. **Count vectorizer** We created a vocabulary by extracting from the document collections
4. **String indexer,** We used a label indexer that maps a string column of labels to an ML column of label indices. In the numeric input column, we cast it to string and index the string values. And converted the classification of spam/ham to 0/1
5. **Vector assembler** We used the transformer which combined all the features of different columns into a single feature vector column.
6. **Pipeline** we then pipelined the data and fed it to the model after cleaning the data and flattening it.

Models used from SciKit Learn :

1. Sklearn MultinomialNB (Naive Bayes) model
2. SGD model
3. Perceptron

- Pickle module used for storing model
- Matplotlib module used to plot graphs to compare the models and the metrics

Clustering - MiniBatchKMeans is compared with KMeans. MiniBatchKMeans is faster, but gives slightly different results with some batches of data as it converges faster than KMeans.

3.3 Surface level implementation details about each unit

All preprocessing steps are common to all models

Tokenizer -> Stop Word Removal -> Count Vectorizer -> String indexer -> Vector Assembler -> Pipeline the preprocessing and fit and transform 'data' to save as clean data

The following models from SciKit Learn are used:

1. NaiveBayes Model - Model with 2 files for training data and then testing the data. The model is saved with a Pickle file and it runs incrementally on the training data.
2. Perceptron model - Model with 2 files for training data and then testing the data. The model is saved with a Pickle file and it runs incrementally on the training data.
3. SGD model - Model with 2 files for training data and then testing the data. The model is saved with a Pickle file and it runs incrementally on the training data.
4. Clustering - Clustering was done with MiniBatchKMeans with incremental learning using Pickle and is plotted in comparison to KMeans.

3.4 Reason behind design decisions

1. **The naive Bayes Model**

   a. gives low false positive spam detection rates that are generally acceptable to users
   b. One of the oldest and trusted methods used for spam detection.
   c. All the features in the model are independent of each other.

2. **Perceptron:**

   The perceptron model enables machines to automatically learn coefficients of weight which helps in classifying the inputs. Also recognized as the Linear Binary Classifier, the perceptron model is extremely efficient and helpful in arranging the input data and classifying the same in different classes
   Other advantages include

1. Adaptive learning: An ability to learn how to do tasks based on the data given for training or initial experience.
3. MLP/Neural networks do not make any assumption regarding the underlying probability density functions or other probabilistic information about the pattern classes under consideration in comparison to other probability-based models.
4. They yield the required decision function directly via training.
5. A two-layer backpropagation network with sufficient hidden nodes has been proven to be a universal approximator

3. **SGD:**

a) It is easier to fit in the memory due to a single training example being processed by the network.
b) It is computationally fast as only one sample is processed at a time.
c) For larger datasets, it can converge faster as it causes updates to the parameters more frequently.

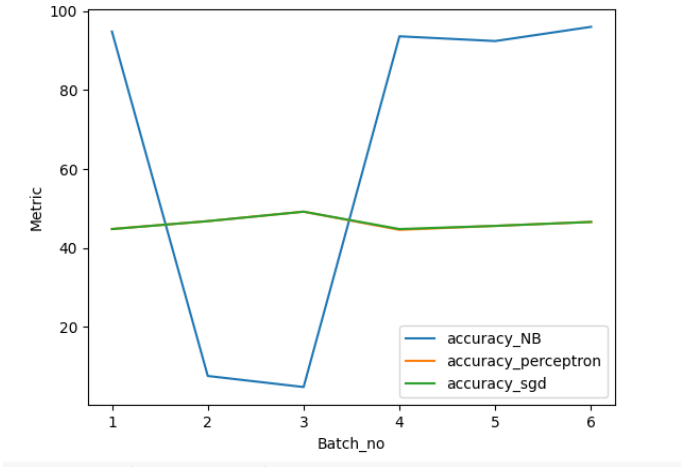
3.5 Takeaway from the project

We learned how to build a full model right from taking the streaming data in batches and converting the data to the type which is compatible with the architecture used and applying all the necessary preprocessing techniques to making and using a machine learning algorithm/model to train the data and also to test the accuracy of the model by running it on the test data to see how good the model is in predicting the future data which might need to be classified. We also learned how to fine-tune the batch size which different values affect the data prediction and accuracy and all tuning the hyperparameters for each model varies the accuracy, but when tuning the hyperparameters we should be careful and keep it at an optimal level to not decrease the accuracy of the model.

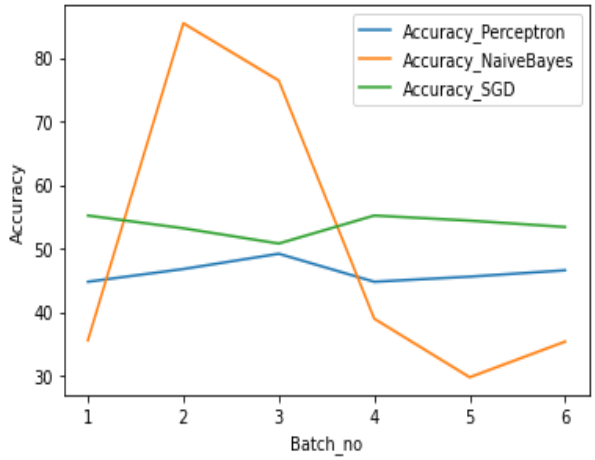
We learned to work with data frames, work with models in Scikit Learn and Pyspark's machine learning model. We visualized how varying batch sizes can affect the accuracy, precision, recall,f1 score of a model.
We learned to implement MinBatchKMeans and KMeans with streaming data and understood and visualized the difference between them.

Accuracy for various models (batch size 500) :

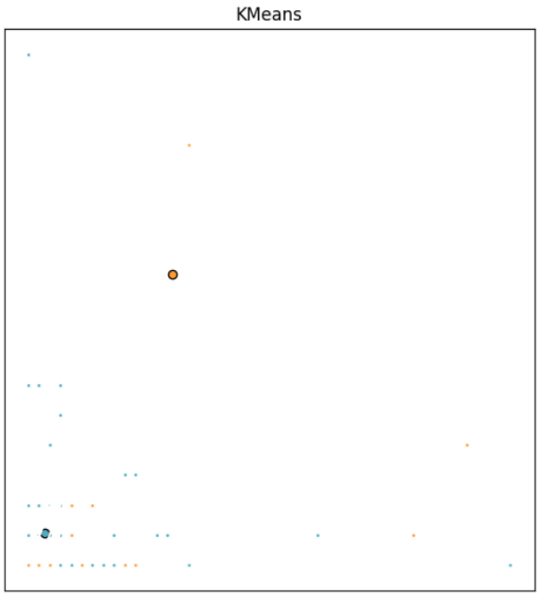
With StopWord Removal

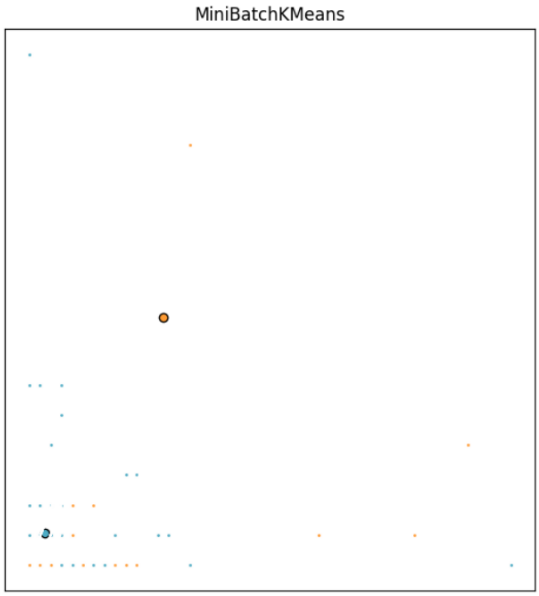


Without StopWord Removal



KMeans vs MiniBatchKMeans comparison is done





Naive Bayesian Multinomial Model With Varying Batch sizes.