BIG DATA PROJECT

UE19CS322

MACHINE LEARNING WITH SPARK MLLIB

Team Details:

TeamID: BD_375_397_439_502

Name	SRN
Rajeshwari R	PES1UG19CS375
Ruchita V R	PES1UG19CS397
Sayonika Das	PES1UG19CS439
Sravya Yepuri	PES1UG19CS502

PROJECT TITLE CHOSEN : MACHINE LEARNING WITH SPARK MLLIB

DATASET CHOSEN:

The dataset chosen for this project is Spam. It has 3 features. The first column stores the subject of the email, the second column contains the content of the email and the third column indicates whether the mail is spam or ham.We have two datasets, they are train.csv and test.csv.The train dataset contains approx 30,000 rows and the test dataset contains 3372 rows.

DESIGN DETAILS:

This project aims at classifying the given dataset into the categories spam or ham. The different steps followed by us in this project is streaming the data, preprocessing, model fitting, training and testing the data using incremental learning. In preprocessing, the removal of the stopwords, numbers, punctuations, and special characters is done. The spam or ham column is encoded into the values 0 and 1 using the LabelEncoder. Hashing vectoriser is used to convert the text documents to a matrix of token occurrences. We have used partial fit to split the large dataset into several batches and train the model consecutively. The metrics such as accuracy, precision, recall and the confusion matrix can be computed for each model using the scikit-learn library.

The models implemented in this project are

1 Naïve Bayes Bernoulli

2 Naïve Bayes Multinomial

3 SGD classifier

4 MiniBatch KMeans Clustering

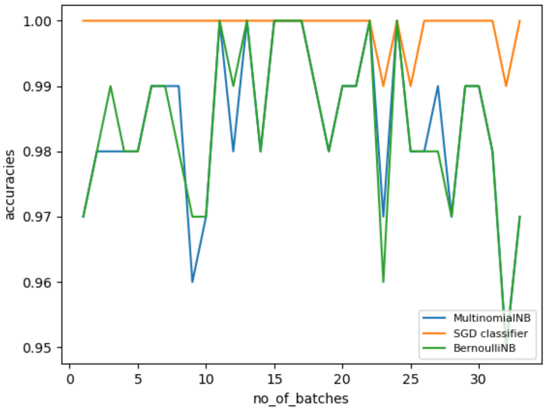
SURFACE LEVEL IMPLEMENTATION ABOUT EACH UNIT

In our project, we have imported some modules from the libraries such as pyspark, sklearn, matplotlib for preprocessing, model fitting, computing the metrics, plotting the graphs and implementing the different classifiers. The data (both train and test) is streamed by stream.py that streams the data in batches (of size 100, by default, 500, 1000 and 1500 by changing it in the command line), this data is received by client.py through the tcp connection and each batch is converted to a dataframe in the df_fun user defined function. We preprocess the dataframe by removing all the stop words, punctuations, numbers, new line characters, tab spaces, etc. Then we binary encode feature2 with the values 0 and 1 for the data classified as spam or ham respectively. We have used partial fit as we are implementing an incremental model which learns based on prior knowledge of batches already used to train the model. We have built 3 models using binomial and multinomial naive bayes and Stochastic gradient descent and fit the data using train.csv and tested using test.csv.

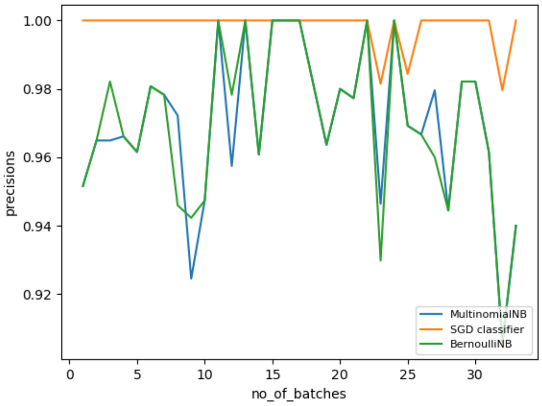
We train 3 different models and calculate the accuracy, precision and recall for each of the three models on the test data and compare them. We plot three graphs with the number of batches on the x-axis and each of the metrics on the y-axis as shown below.

Batch Size: 100

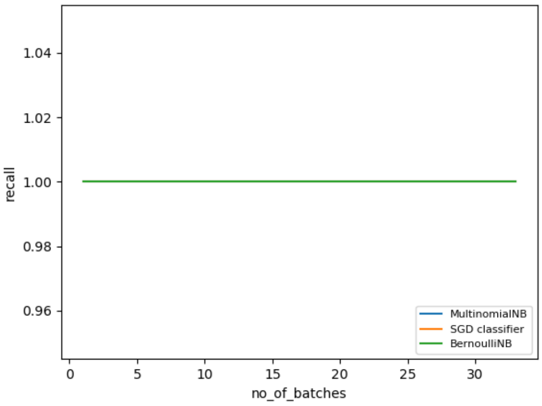
Accuracy Comparison for 3 different Models



Precision Comparison for 3 different Models

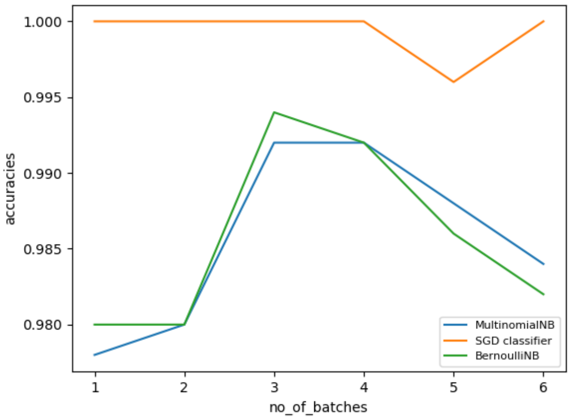


Recall Comparison for 3 different Models

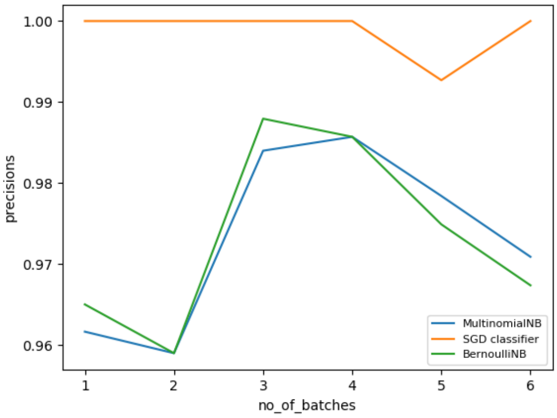


Batch Size: 500

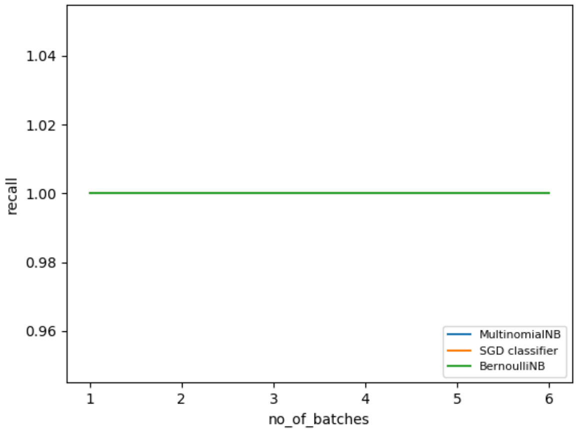
Accuracy Comparison for 3 different Models



Precision Comparison for 3 different Models

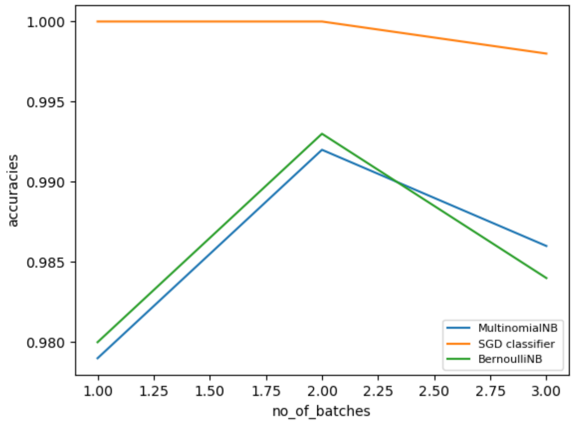


Recall Comparison for 3 different Models

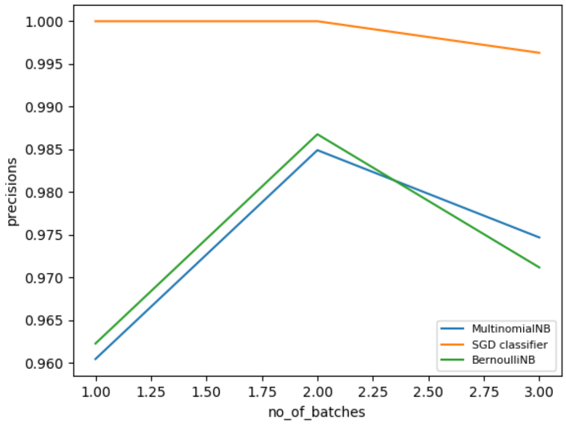


Batch Size: 1000

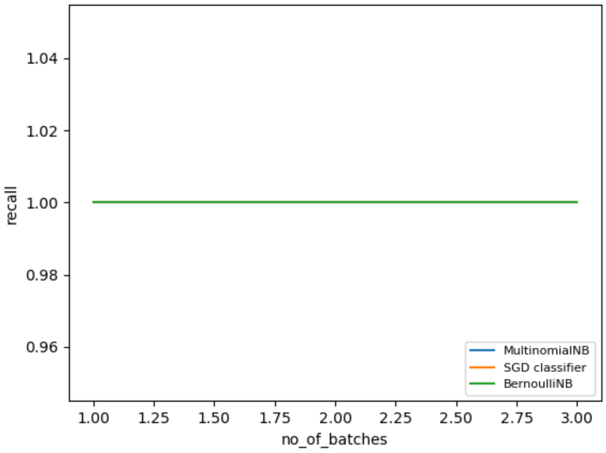
Accuracy Comparison for 3 different Models



Precision Comparison for 3 different Models

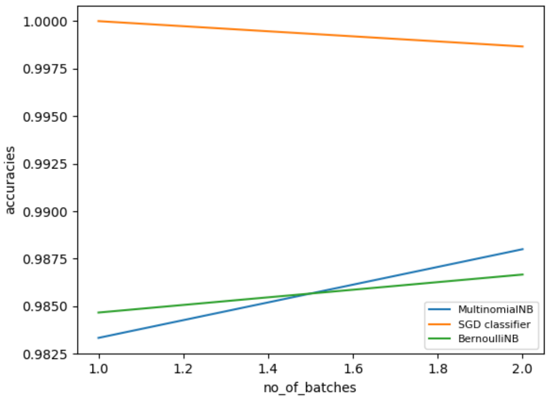


Recall Comparison for 3 different Models

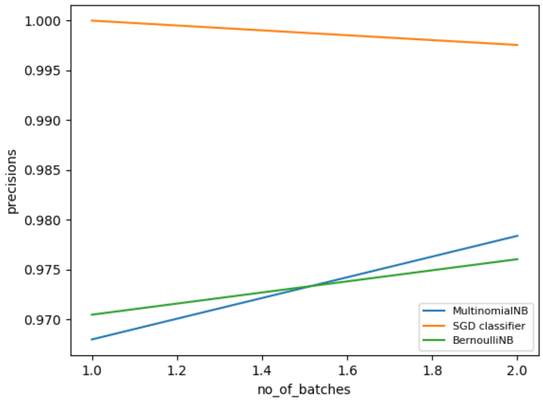


Batch Size: 1500

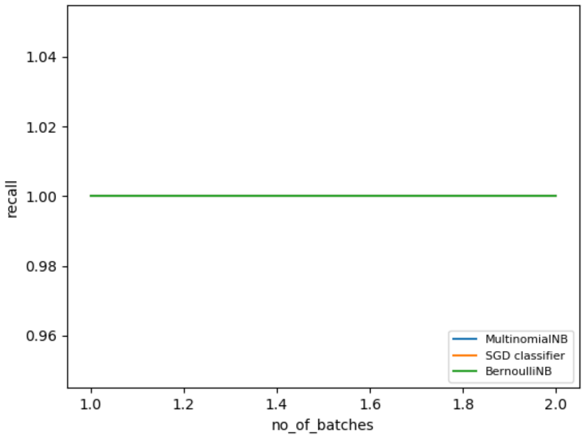
Accuracy Comparison for 3 different Models



Precision Comparison for 3 different Models

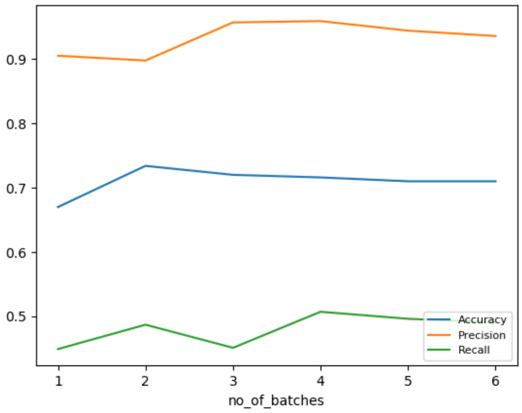


Recall Comparison for 3 different Models

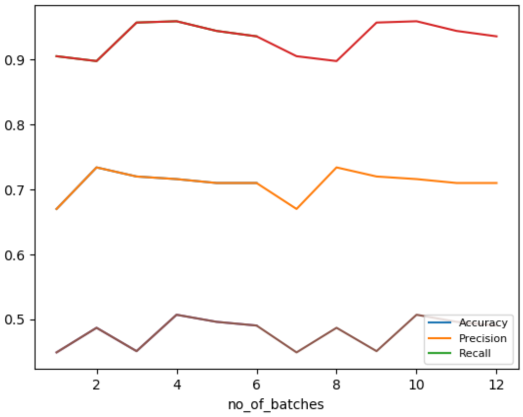


Minibatch k-means clustering is used for large single cell sequencing data. MBKmeans is used to return a list object including centroids, wcss, clusters, etc. Based on the number of batches we consider the training or test dataset. The fit_transform function is used to transform the text to numbers which makes it easier to plot a graph for the clusters. Given below are the graphs plotted for clustering using the matplotlib library.

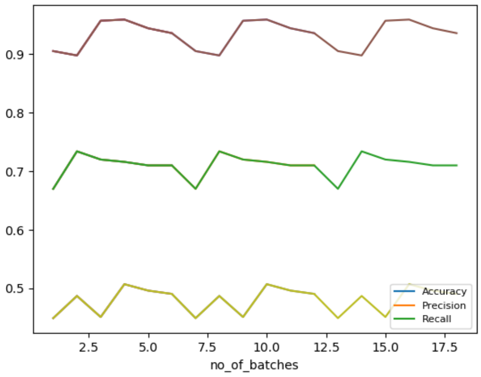
Cluster Performance Metrics after one stream of train and test dataset



Cluster Performance Metrics after two streams of train and test dataset



Cluster Performance Metrics after three streams of train and test dataset



REASONS BEHIND DESIGN DECISIONS

The Naïve Bayes algorithm is used for classification problems and is faster than other classification algorithms. It predicts the class of a dataset using the prior knowledge. Multinomial bayes algorithm works for both continuous and discrete data. It is scalable and can handle enormous datasets with ease.

Stochastic Gradient Descent (SGD) is easier to fit in the memory and is computationally faster than other data models as only one model is processed at a time.

K-means clustering is an unsupervised learning algorithm which is simple and easy to implement and is suitable for handling large datasets.It is an iterative algorithm that divides the unlabelled dataset into k different clusters.

TAKE AWAY FROM THE PROJECT

The two main concepts learned from this project are streaming the data and incremental learning which is a machine learning paradigm. We also learnt how to implement these concepts using Pyspark.

Acknowledgement

We are highly indebted to our teacher Dr. KV Srinivas sir and the Teaching Assistants and the entire Big Data team, for providing with significant knowledge and resources to gain more in depth knowledge on the learnt concepts for the guidance and constant supervision as well as for providing necessary information regarding the project and also for the support in completing the project.We would like to thank the H.O.D of Computer Science Department, Ms. Shylaja Sharath for giving us opportunities to do such projects. Our thanks and appreciations also go to our teammates in developing the project and people who have willingly helped us out with their abilities.