# BIG DATA PROJECT REPORT
# Machine Learning with Spark Streaming

**TEAM_ID: BD_034_124_477_504**
**TEAM MEMBERS:**

| | |
|---|---|
| Aditya Srinivasa | PES1UG19CS034 |
| Charushree A | PES1UG19CS124 |
| Shweta Patki | PES1UG19CS477 |
| Sri Ramya Priya Vedula | PES1UG19CS504 |

1. **Project Title:**
   Machine Learning with Spark Streaming - Sentiment Analysis

2. **Design Details:**
   - Models are pickled.
   - Hashing Vectorizer from Sklearn module.
   - String Indexer from Mllib module
   - Use of nonlinear, linear and cluster models.

3. **Surface level implementation details about each unit:**

   1) Streaming:
      Tweets are read from port 6100 through tcp connection from provided stream.py and using flatmap transformation all tweets are retrieved and batch is sent for preprocessing and a check if len >0 so to insure if the batch is empty or not and the batch is converted to spark dataframe.

   2) Preprocessing:
      Links and mentions are removed (using regex_replace) from all the tweets .The tweets are then tokenized using Sklearn HashingVectorizer with hyperparameters to remove stopwords, which takes a list of stopwords to be removed. It is used to convert tweets to a matrix of dtype float64.
      All the sentiment labels are changed to 0 and 1 using pyspark StringIndexer and are changed to numpy array .

   3) Classification:
      a) Naive Bayes: Model was depickled and we used partial_fit() for incremental learning.
         Hyperparameter tuning: alpha = 0.1 performs best. A small alpha value causes overfitting.
      b) Perceptron: Model was depickled and we used partial_fit() for incremental learning.

Hyperparameter tuning: alpha=0.01, tol=le-3 and random_state=0 perform best.

c) SGD: Model was depickled and we used partial_fit() for incremental learning.
Hyperparameter tuning: alpha=0.01, max_iter=1000, tol=0.01 and learning_rate='optimal' perform best.

4) Testing: Root mean square error and Accuracy was printed for every model we tested. At the end, we also printed mean absolute error, mean squared error, root mean square error, r2, classification report, and confusion matrix.

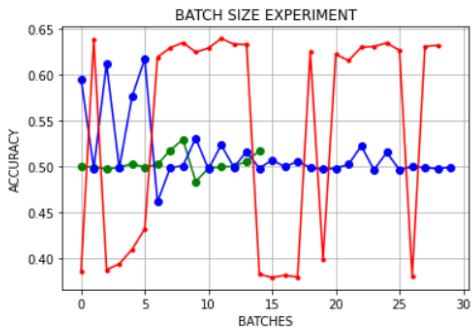
5) Clustering using Mini Batch K Means:
Clustering model was depickled and we used partial_fit() for incremental learning.
Hyperparameter tuning:
The hyperparameters included are n_clusters=2, max_iter=1000, tol=1.0, max_no_improvement=20, reassignment_ratio=0.0001
For best performance, the reassignment ratio should be less and max_no_improvement should be more.

4. **Reason behind design decisions:**
   - Models are saved using the technique of pickling and will be depickled at time of use.
   - Hashing Vectorizer was used to vectorize as there was no need for any conversion to numpy array and also it had a hyperparameter to remove stopwords.
   - Naive Bayes model was used so that we could experiment with both linear and non linear models of sklearn.

5. **Takeaway from the project:**
Our takeaway from this project is that the batch size must be optimal to achieve high accuracy. The best performing classifier on this dataset is Naive Bayes classifier.

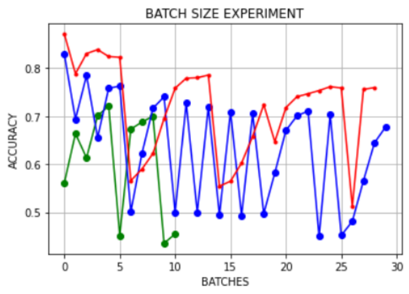
## Plots

**Batch size vs Accuracy experiments:**
**Linear model**
**With increasing batch size**



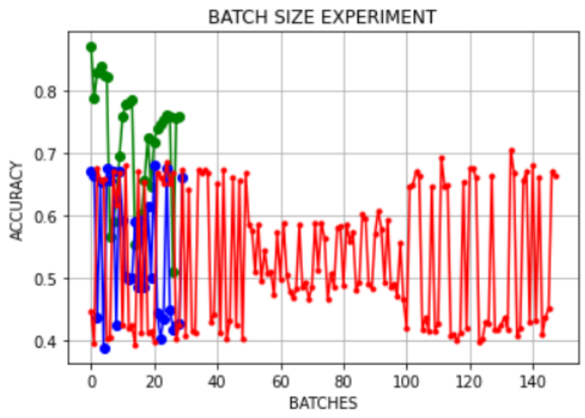
Red line: batch size of 10000
Blue line: batch size of 50000
Green line: batch size of 100000

**Batch size vs Accuracy experiments:**
**Non Linear model**
**With decreasing batch size**



Red line: batch size of 10000
Blue line: batch size of 5000
Green line: batch size of 1000

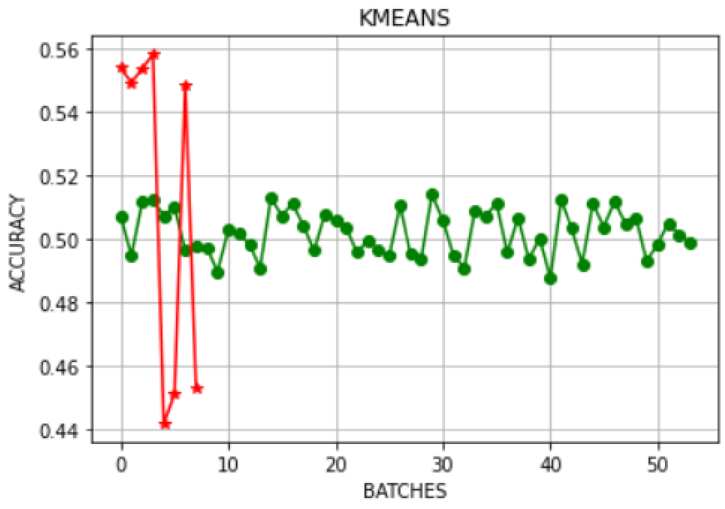
**Batch size vs Accuracy experiments:**
**Non Linear model**
**With decreasing batch size**



Red line: batch size of 1000
Blue line: batch size of 5000
Green line: batch size of 10000

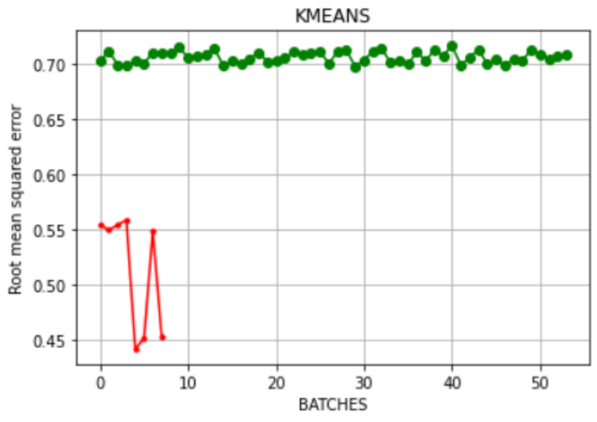
MINI BATCH K MEANS

1. PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000



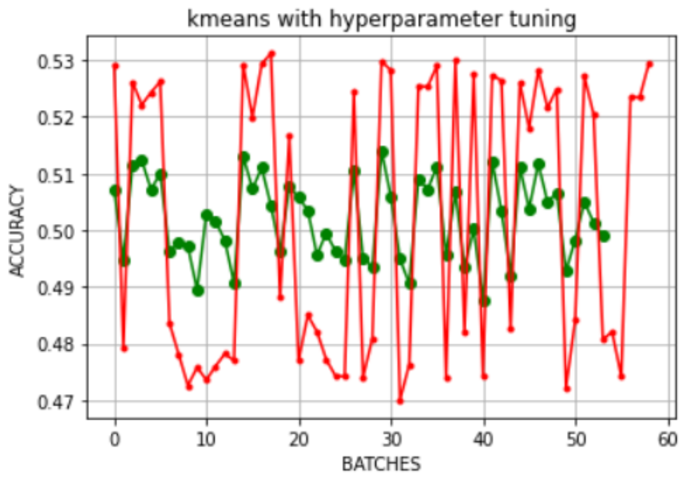
Green line: raining data
Red line: testing data

2. PLOT BETWEEN ROOT MEAN SQUARED ERROR AND BATCHES FOR BATCH SIZE =10000



Green line: raining data
Red line: testing data

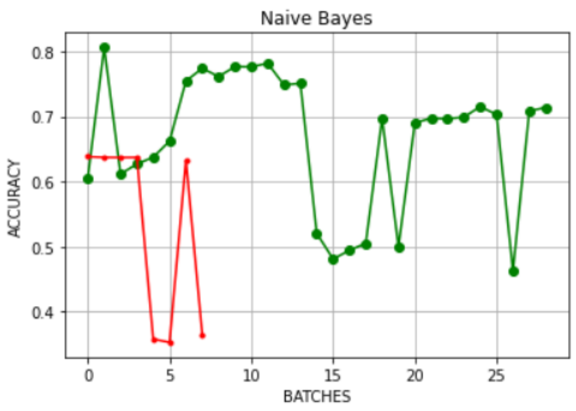
3. PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000 WITH CHANGE IN LEARNING RATE.



Green line: raining data
Red line: testing data

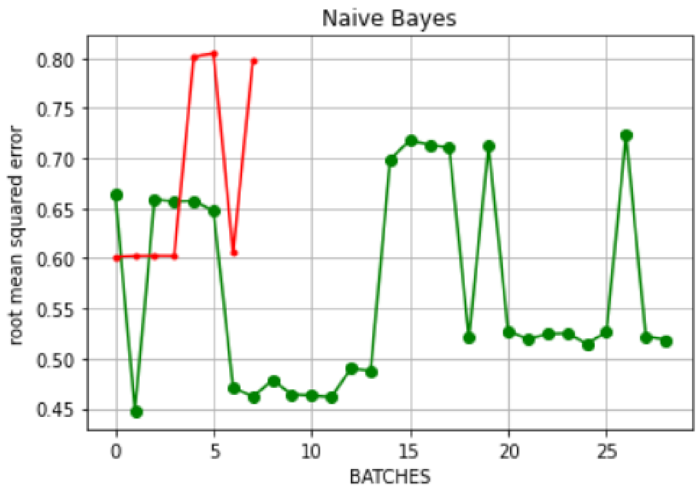
## NAIVE BAYES

1. PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000



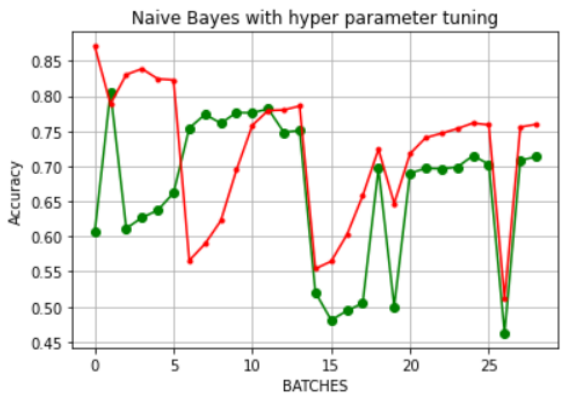
Green line: raining data
Red line: testing data

2. PLOT BETWEEN ROOT MEAN SQUARED ERROR AND BATCHES FOR BATCH SIZE =10000



Green line: raining data
Red line: testing data

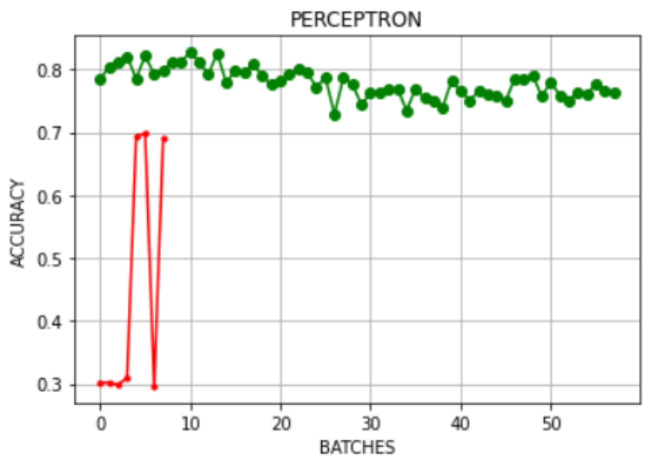
3. PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000 WITH CHANGE IN LEARNING RATE.



Green line:raining data
Red line:testing data

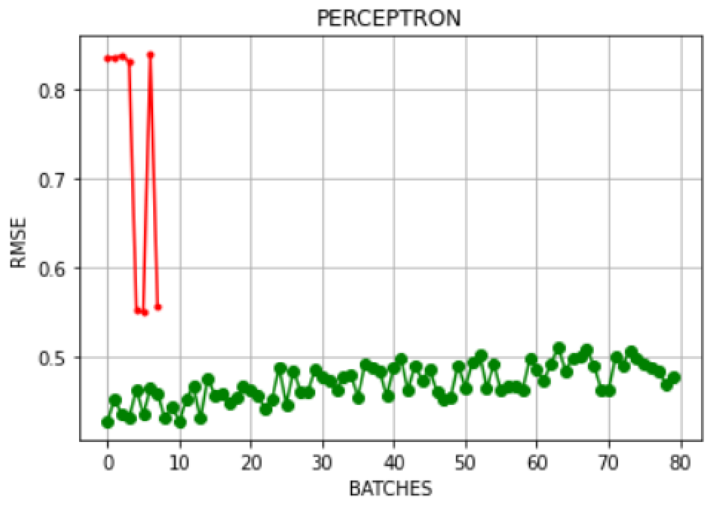
## PERCEPTRON

### 1.PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000



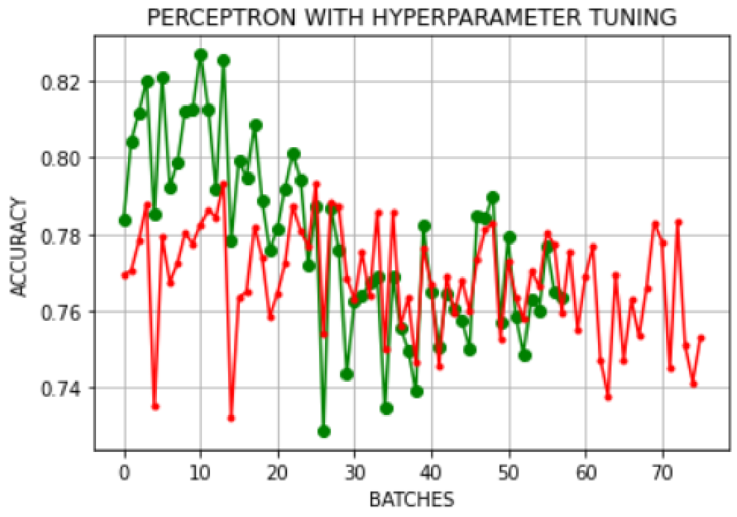
Green line:raining data
Red line:testing data

### 2.PLOT BETWEEN ROOT MEAN SQUARED ERROR AND BATCHES FOR BATCH SIZE =10000



Green line:raining data
Red line:testing data

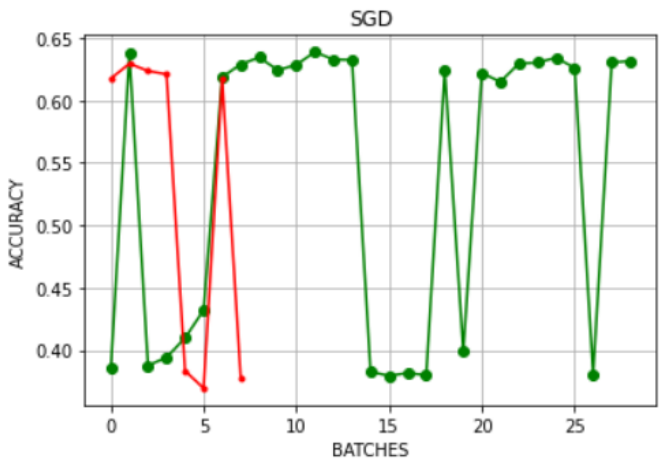
3.PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000 WITH CHANGE IN LEARNING RATE.



Green line: training data
Red line: testing data

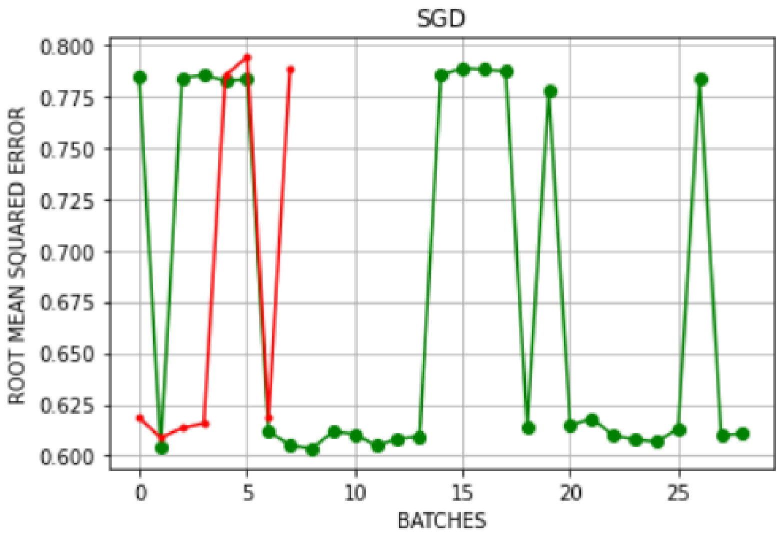
## <u>SGD</u>

1.PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000



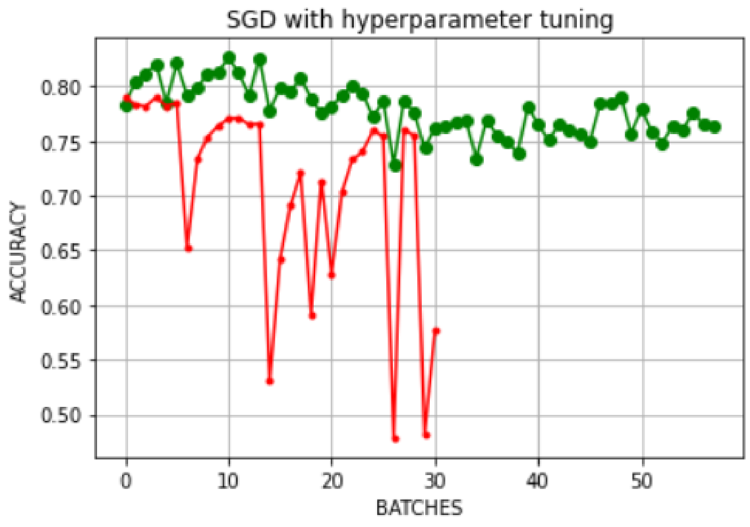
Green line:raining data
Red line:testing data

2.PLOT BETWEEN ROOT MEAN SQUARED ERROR AND BATCHES FOR BATCH SIZE =10000



Green line:raining data
Red line:testing data

3.PLOT BETWEEN ACCURACY AND BATCHES FOR BATCH SIZE =10000 WITH CHANGE IN LEARNING RATE.



Green line:raining data
Red line:testing data