

Machine Learning with Spark MLlib

Dataset chosen : 'Sentiment'.

Team ID : BD_108_111_169_308

Design Details :

- 1)Spark Streaming.
- 2)Preprocessing.
- 3)Supervised Learning.
- 4)Unsupervised Learning.
- 5)Hyperparameter Tuning.
- 6)Graphs.

Surface level Implementation Details:

Spark Streaming : The Streaming is done using Json file,when spark receive the file it is in Dstream of the Json file then we converted this to RDD ,then used `spark.read.json()` to read the json format and converted into a list.In list we take individual columns and convert it into dataframe.

Pre-Processing :Using regular expressions we cleaned the data like removal of URLs ,hashtags, RT,'@', numerical data , punctuations and unwanted spaces. We applied stemming using `PorterStemmer()` which is available in the `nlTK.stem` library. We also used `Hashvectoriser` which removes stopwords,converted tweets to lower case and converted all tweets to vectors(`x_train` matrix). We applied SVD on the hash vectorizer to reduce it 10 features and then trained and tested the model.We found out that the accuracy is very less i.e
Perceptron model : 0.480837, passive aggressive classifier: 0.518985,SGD classifier : 0.496833

Supervised Learning : As the Data is available in batches we have to incrementally learn the model using `partial_fit()`.We created a global variable `flag=0`,when the first batch arrives we train the model and dump it into pickle file(.sav file) change the state of flag variable.So, when the next batch arrives we load the previously stored model and the `partial_fit` the model and then dump it again into the same pickle file.

The models we trained for sentiment analysis: Multinomial NB , SGD(Stochastic Gradient Descent), Perceptron, Passive Aggressive classifier.

1) Multinomial NB : This Classifier is suitable for classification for discrete features.

2) SGD(Stochastic Gradient Descent) Classifier : SGD Classifier is a linear classifier (SVM, logistic regression, a.o.) optimized by the SGD. These are two different concepts. While SGD is an optimization method, Logistic Regression or linear Support Vector Machine is a machine learning algorithm/model .

3) Perceptron : A perceptron consists of four parts: input values, weights and a bias, a weighted sum, and activation function. The idea is simple, given the numerical value of the inputs and the weights, there is a function, inside the neuron, that will produce an output.

4) Passive-Aggressive Classifiers : Passive-aggressive classification is one of the available incremental learning algorithms and it is very simple to implement, since it has a closed-form update rule. ... The core concept is that the classifier adjusts its weight vector for each misclassified training sample it receives, trying to correct it.

Unsupervised Learning :

Here we apply partial fit so for every batch centroids of the clusters are altered. The model here we used is Mini batch K-means algorithm. Mini Batch K-means algorithm's main idea is to use small batches of data of a fixed size, so they can be stored in memory. Each iteration a new sample from the dataset is obtained and used to update the clusters and this is repeated until convergence. Each mini batch updates the clusters using a convex combination of the values of the prototypes and the data, applying a learning rate that decreases with the number of iterations. The number of clusters used here is equal to the number of features which is equal to 2.

Hyperparameter Tuning :

1) Multinomial NB : 'alpha' which is a smoothing parameter.

'alpha'=0.1.

2) SGD Classifier: loss(loss function to be used), penalty(regularization term), 'alpha'(learning rate).

loss='modified_huber', penalty='l2', alpha=0.0001

3) Perceptron : penalty(regularization term), 'alpha'.

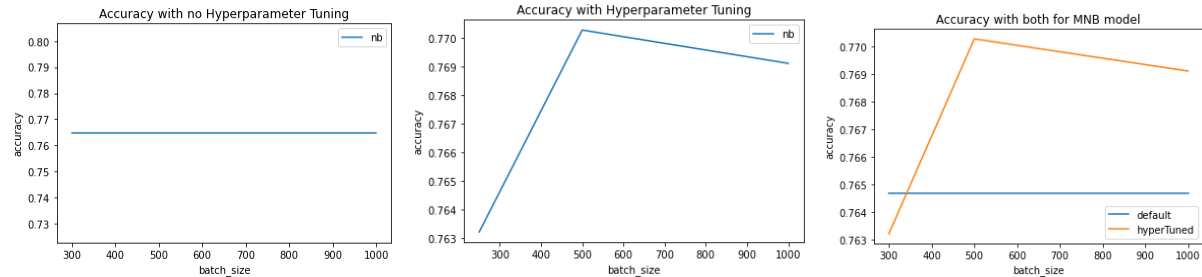
alpha=0.0001

4) Passive-Aggressive Classifiers : C(maximum step size), loss(loss function to be used).

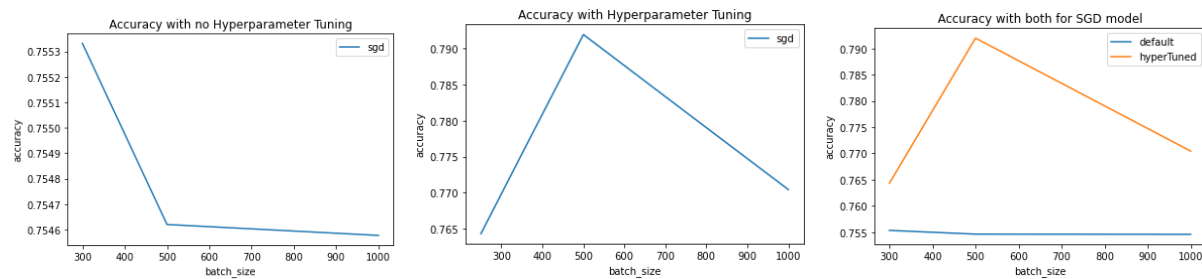
C=0.003,loss='squared_hinge'

Graphs:

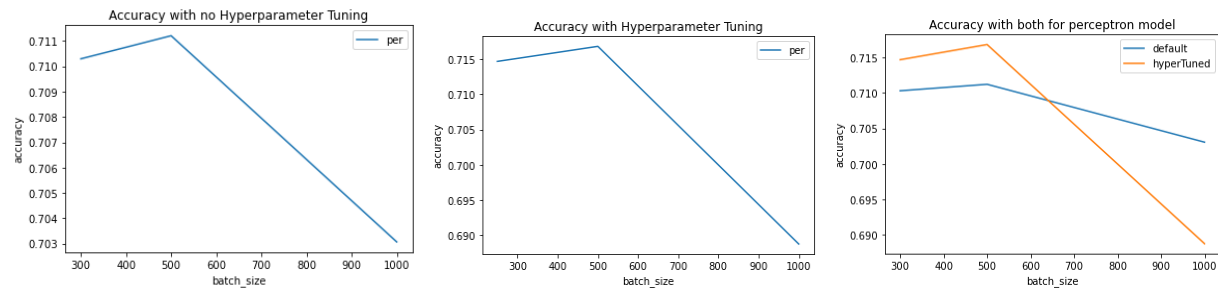
1)Multinomial NB :



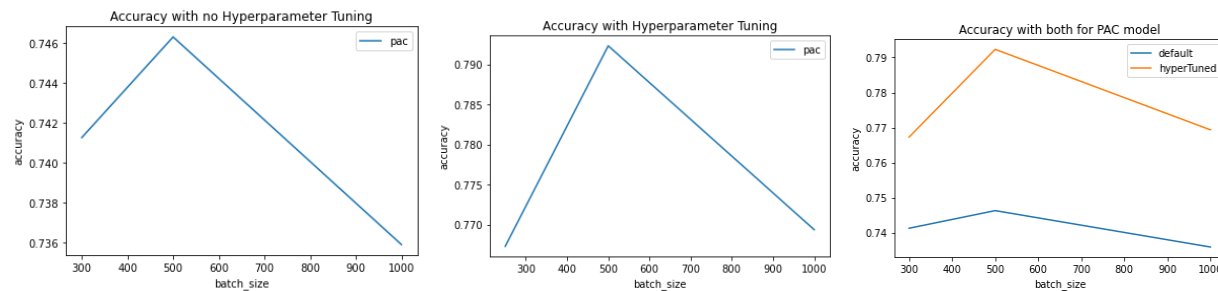
2)SGD Classifier:



3)Perceptron :



4)Passive-Aggressive Classifiers :



Classification Reports :

i) size=500,Default:

Multinomial NB batch :

Actual Values

	Positive	Negative
Positive	TP=32616	FP=7579
Negative	FN=10144	TN=29661

Precision : 0.811
Recall : 0.7627
F1_Score : 0.7863
Accuracy : 0.76465

Perceptron batch size :

Actual Values

	Positive	Negative
Positive	TP=30020	FP=10175
Negative	FN=12311	TN=27494

Precision : 0.7468
Recall : 0.7091
F1_Score : 0.7275
Accuracy : 0.7188

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=32476	FP=9719
Negative	FN=8393	TN=31412

Precision : 0.7582
Recall : 0.7840
F1_Score : 0.7709
Accuracy : 0.7860

Passive-Aggressive Classifiers :

Actual Values

	Positive	Negative
Positive	TP=30626	FP=9569
Negative	FN=10020	TN=29785

Precision : 0.7619
Recall : 0.75340
F1_Score : 0.7576
Accuracy : 0.755

ii) size=500,hyperparameter tuning:

Multinomial NB batch :

Actual Values

	Positive	Negatives
Positive	TP=31893	FP=8302
Negative	FN=9636	TN=30169

Precision : 0.7934

Recall : 0.76796

F1_Score : 0.7805

Accuracy : 0.775

Perceptron batch size :

Actual Values

	Positive	Negative
Positive	TP=32668	FP=7527
Negative	FN=15767	TN=24038

Precision : 0.8127

Recall : 0.674

F1_Score : 0.737

Accuracy : 0.7088

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=31486	FP=8709
Negative	FN=8409	TN=31396

Precision : 0.7833

Recall : 0.7892

F1_Score : 0.7862

Accuracy : 0.7860

Passive-Aggressive Classifiers :

Actual Values

	Positive	Negative
Positive	TP=31140	FP=9055
Negative	FN=8261	TN=31544

Precision : 0.7744

Recall : 0.790

F1_Score : 0.782

Accuracy : 0.7835

iii) size=1000,Default :

Multinomial NB batch :

Actual Values

	Positive	Negative
Positive	TP=35519	FP=4676
Negative	FN=8479	TN=31326

Precision : 0.883
Recall : 0.8072
F1_Score : 0.8437
Accuracy : 0.76465

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=29859	FP=10336
Negative	FN=7774	TN=32031

Precision : 0.7428
Recall : 0.7934
F1_Score : 0.7673
Accuracy : 0.7736

Perceptron batch size :

Actual Values

	Positive	Negative
Positive	TP=30019	FP=10176
Negative	FN=12116	TN=27689

Precision : 0.7468
Recall : 0.7124
F1_Score : 0.7292
Accuracy : 0.7213

Passive-Aggressive Classifiers :

Actual Values


	Positive	Negative
Positive	TP=29549	FP=10646
Negative	FN=9918	TN=29887

Precision : 0.7351
Recall : 0.7487
F1_Score : 0.7418
Accuracy : 0.7429

iv) size=1000,hyperparameter tuning :

Multinomial NB batch :

Actual Values

	Positive	Negative
Positive	TP=31893	FP=8302
Negative	FN=9636	TN=30169 

Precision : 0.7934
Recall : 0.7679
F1_Score : 0.7805
Accuracy : 0.7670

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=31193	FP=9002
Negative	FN=8140	TN=31665

Precision : 0.7760
Recall : 0.7930
F1_Score : 0.7844
Accuracy : 0.7840

Perceptron batch size :


Actual Values

	Positive	Negative
Positive	TP=31697	FP=8498
Negative	FN=15157	TN=24648

Precision : 0.7885
Recall : 0.6765
F1_Score : 0.7282
Accuracy : 0.7413

Passive-Aggressive Classifiers :

Actual Values

	Positive	Negative
Positive	TP=31099	FP=9096
Negative	FN=8232	TN=31573 

Precision : 0.7737
Recall : 0.7906
F1_Score : 0.7821
Accuracy : 0.7812

v)size=300,Default:

Multinomial NB batch :

Actual Values

	Positive	Negative
Positive	TP=32620	FP=7575
Negative	FN=10144	TN=29661

Precision : 0.81
Recall : 0.7627
F1_Score : 0.7864
Accuracy : 0.76465

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=30531	FP=9664
Negative	FN=8435	TN=31370

Precision : 0.7595
Recall : 0.7835
F1_Score : 0.7713
Accuracy : 0.7737

Perceptron batch size :

Actual Values

	Positive	Negative
Positive	TP=26289	FP=13906
Negative	FN=9227	TN=30578

Precision : 0.788
Recall : 0.676
F1_Score : 0.7282
Accuracy : 0.704

Passive-Aggressive Classifiers :

Actual Values

	Positive	Negative
Positive	TP=31181	FP=9014
Negative	FN=10990	TN=30578

Precision : 0.7757
Recall : 0.7393
F1_Score : 0.7571
Accuracy : 0.7499

vi)size=300,hyperparameter tuning:

Multinomial NB batch :

Actual Values

	Positive	Negative
Positive	TP=31893	FP=8302
Negative	FN=9636	TN=30169

Precision : 0.7934
Recall : 0.7679
F1_Score : 0.7805
Accuracy : 0.775

SGD classifier :

Actual Values

	Positive	Negative
Positive	TP=31355	FP=8840
Negative	FN=8282	TN=31523

Precision : 0.7800
Recall : 0.7910
F1_Score : 0.7855
Accuracy : 0.7859

Perceptron batch size :

Actual Values

	Positive	Negative
Positive	TP=27428	FP=12767
Negative	FN=9564	TN=30241

Precision : 0.6823
Recall : 0.7414
F1_Score : 0.7106
Accuracy : 0.7208

Passive-Aggressive Classifiers :

Actual Values

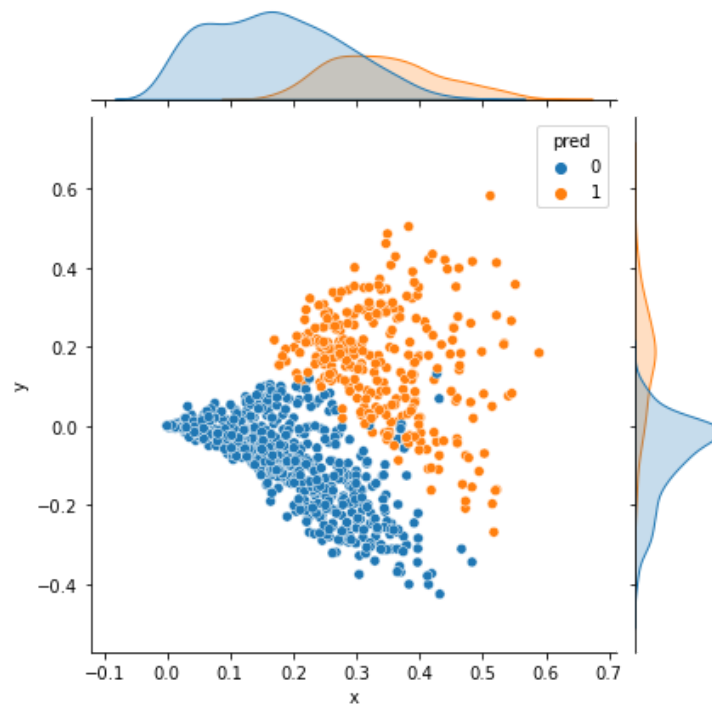
	Positive	Negative
Positive	TP=31116	FP=9079
Negative	FN=8240	TN=31565

Precision : 0.7741
Recall : 0.7906
F1_Score : 0.7822
Accuracy : 0.7835

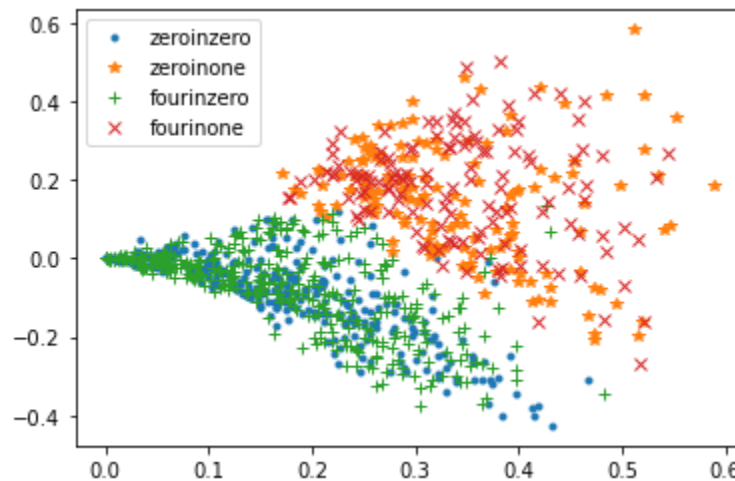
Reason Behind Design Decisions:

Since we are streaming data in batches each batch is stored as an RDD. For every RDD we are applying pre-processing, and calling different model functions, and checking their accuracy. In order to achieve the best model.

We have two labels for the dataset namely 0 and 4 ,so we also divided clusters into two groups namely 0 and 1.



The first figure depicts how the data points are divided among the two clusters.



The second figure is the analysis of different labels belonging to different classes. That is plotted the figure for number of data points of label 0 belonging to cluster 0 and 1 and similarly to label 4.

The number of data points of label zero belonging to cluster zero : 27857

The number of data points of label zero belonging to cluster one : 12338

The number of data points of label four belonging to cluster zero : 29478

The number of data points of label four belonging to cluster one : 10327

Takeaway From Project :

We learnt about batch streaming, incremental learning, implementation of partial fit model. How exactly pyspark works. About clustering techniques etc.