Machine Learning with Spark Streaming - Sentiment Analysis

Team Details:

Team ID : BD_304_310_315

Team Member Details:

Name	SRN
Om Amitesh B R	PES1UG19CS310
Nishanth M S	PES1UG19CS304
P Girivinay	PES1UG19CS315

Project Brief:

In this project, we aim to explore incremental learning via spark streaming to classify tweets based on their sentiment as positive/negative.

For the task of classification, the following three classifier(supervised learning) have been implemented:

1. Naive Bayes Classifier
2. Stochastic Gradient Descent Classifier
3. Passive Aggressive Classifier.

For the unsupervised learning task, the following model has been implemented:

1. MiniBatch K Means Clustering.

These models have been chosen from the given list of models in the sklearn library that support incremental learning via the partial_fit function

Design & Implementation :

Model Instantiation:

A simple python code creates model objects and pickles them so that they can be trained while streaming. Command line arguments can be used for selection of models.

Data Flow for incremental learning via spark streaming:

The following stages are to stream and train models on given batches of data:

Stage 1: Receiving and processing data

1. The data is received via streaming context's socketTextStream to receive a dstream, each RDD of which is processed using the process function.

2. Each RDD is collected to give the json in string format, this is converted to a dictionary via the json library
3. The dictionary is then converted to a spark dataframe to make it easier to obtain features.
4. The features are selected via the dataframe.select() and made into 2 numpy arrays(of tweets and labels separetely) for the preprocessing step.

Stage 2: Preprocessing data:

Following steps are used to the preprocess given tweets, and make them ready for our models to fit:

1. The tweets are each sent into a preprocess function that:(done via re library)
    a. Removes punctuations from all words
    b. Removes special symbols, twitter ids and links
    c. Removes stop words(like ‘a’, ‘the’ etc)
    d. Converts all words in the tweet to lowercase
2. The tweets are then vectorised using a HashingVectorizer(other vectorisers were count-dependent and could not be used for incremental learning) to obtain a vector of pre-set no. of features.
3. The training data is split into training and validation sets via train_test_split.

Stage 3: Training models:

1. The model is loaded from its saved file and the partial_fit function is called over the training subset.
2. The model is evaluated via the validation subset to obtain training metrics.
3. The model is then trained over the validation data so that it doesn't miss out on it.
4. The trained model is saved to its file path so that it can be retrieved for the next batch.

This data flow is common to training both the classifiers and the clustering algorithm(centroid estimation).
For testing, the same stages are followed except instead of stage 3, we directly evaluate the model on the testing data via model.predict() to obtain metrics.
The metrics are written to separate files so that they can be analysed later as well.

Takeaways:

This Project helped us to:

> Learn spark streaming to stream batches of data which probably cannot reside on a single machine.

> Implement incremental learning to train models on ‘Big Data’

> Understand hyperparameter tuning to observe different results