Big Data Project Report

Team - BD_182_195_249_804
Github link - https://github.com/jjc1805/BD_182_195_249_804

I. PROJECT TITLE CHOSEN:

We have chosen Machine Learning with Spark MLib as our project title. We will perform our analysis on the Spam dataset.

II. DESIGN DETAILS:

We are provided with a stream that streams data in batches. Since we have to build machine learning model on all the data in the stream and it is not possible to store all the data sent by the stream due to lack of memory space , we resort to incremental machine learning techniques to build a model for the data in the stream.

We parse each batch passed in the stream inorder to select the necessary details.We then extract the features in order to perform classification. Feature extraction for text classification can be done using various methods such as TF-IDF, count vectorizer, Word2Vec, etc.

We then scale the data to a common scale using any of the normalization or standardisation methods. We then pass the extracted features and the labels to an incremental machine learning model. These incremental machine learning model are capable of improving by repeated training on new small batches of data.So we train the model on the features extracted for the batch. We then save fitted model using any of the model saving techniques which serialize the models trained and store them in a file on the local storage.To incrementally train the model on the next batch, we load the model from the file and incrementally fit the data.

On doing this repeatedly over many batch of data in the stream , the model will have effectively learned the whole data. Thus this method of incrementally learning solves the problem of machine learning over huge datasets.

III. SURFACE LEVEL IMPLEMENTATION DETAILS:

The input data is received through the stream application running at socket 6100 and the data received is in json format. With the help of spark stream processing modules ,

we process the data in this stream. We create a Dstream from the current stream using the spark unstructured streaming modules which is socketTextStream in this case. We then iterate over each rdd in the stream with the help of foreachRDD function of the Dstream object.

Each batch sent in the socket stream is received as rdd. And this rdd is processed to perform the required learning.We now have to extract the features for each rdd. We extract the text features from the body of the e-mail using hashingVectorizer provided by sklearn which generates feature vectors by hashing the given text over a defined hashing function. The number of features chosen for hashing must be large enough as hashing based feature selection techniques suffer from key collisions. Hence the number of features chosen is 1000. We then normalise the features using MaxAbsScaler provided by sklearn and encode the classification labels using labelEncoder provided by sklearn.

We feed in the features extracted and the labels to the incremental model. Hence logistic regression can be applied for classification. It generates several mini-batches which are trained for the specified training set . A passive aggressive classifier is used which will help in demarcating the correct and incorrect classifications by incrementally feeding the training instances. Subsequently a perceptron classifier has been implemented which is used for binary classification.For clustering the instances, we have implemented k-means clustering, specifically the MiniBatchKMeans algorithm which is faster than the traditional K-means model.

After we model the data , we store the model using pickle provided by python which serializes the given model into a local file so that we can incrementally learn on the next batch of data.For analyzing the performance of the model, we first predict on the new batch using the previously learned model and compare it with the actual classifications given in the batch and access the performance using various performance metrics.

IV. REASON BEHIND DESIGN DECISIONS:

Feature extraction from text can be done using various methods such as TF-IDF, count vectorization , etc. But for incremental learning, not all are suitable. We have to either choose a hashing based feature extractor or a stateful feature extractor which remembers the state across all the batches. In our case, we implement feature extraction using one of the hashing techniques due to its easy implementation and understandability.

We save the model , so that we can load the model for the next batch to incrementally learn. If not saved will cause the weights to be reset to initial values.

V. TAKEAWAY FROM THIS PROJECT:

This project helped us delve deeper and understand the working of the spark framework and also taught us about the various libraries (Mlib, sklearn) that could be used to implement a machine learning model. It also helped us to implement machine learning over large datasets that cannot be processed by a single computer due lack of memory or storage. It helped us gain a deeper understanding of implementing incremental machine learning over batches of data.