BD PROJECT

PROJECT TITLE : Machine Learning with Spark MLlib

TEAM ID: BD_442_458_557_585

NAMES	SRN
Seshank I	PES1UG19CS442
Shourya N Kumar	PES1UG19CS458
Y K Ujwal	PES1UG19CS585
Vanshika Taneja	PES1UG19CS557

Project Title: Detection of spam emails
DATASET CHOSEN: SPAM

Design details:

We have implemented the following steps in three different functions:

- **Pre-processing** – Remove all the stop words and remove the alphabets from the given subject and body of the email (we combine both subject and body together as one)

- **Representation-** Conversion of an array of sentences into a sequence of numbers (into an attribute-value pairs' vector)
  Here the sentences are converted into tokens and passed through HashingVectorizer

- **Learning-** Passing the hashingvectorized stream into the 3 models and 1 clustering model and partially fitting these as we have a continuous incoming data as batches (so we can retain the previous learning)

Surface level implementation:

- **HashingVectorizer:** It is used to convert a collection of text documents to a matrix of token occurrences.

- **LabelEncoder:** It is used to encode target labels with a value between 0 and 1 (number_of_unique_classes = 2).

- **Classifiers:**
  - **Perceptron:** The Perceptron is a type of linear classifier used for supervised learning of binary classifiers. It is suitable for large-scale learning.
  - **Multinomial NB:** It is an instance of Naïve Bayes classifier. It is suitable for classification with discrete features It assumes that the features are drawn from a simple Multinomial distribution.
  - **SGD Classifier**: This estimator implements regularized linear models with stochastic gradient descent (SGD) learning. The gradient

of the loss is estimated for each sample at a time instead of the sum of the gradient of the cost function of all the samples.

- **Clustering algorithms:**

    - **MiniBatchKMeans:** K means groups unlabelled data points into distinct non-overlapping clusters. The MiniBatchKMeans is a variant of the KMeans which uses mini-batches(subsets of the input data, randomly sampled in each training iteration) to reduce the computation time while optimizing the objective function.

The reason behind design decisions:

We chose scikit-learn's models for our project because they are very easy to implement

- HashingVectorizer maintains no vocabulary and determines the index of a word in an array of fixed size via hashing. Since no vocabulary is maintained, the presence of new or misspelled words doesn't create any problem. Also, the hashing is done on the fly and memory need is diminished.

- The above mentioned 3 models are used since it supports incremental analysis using partial fit and finally predict it on the test.csv

- MiniBatchKMeans was chosen because we are clustering the data into two groups i.e., spam and ham.

The takeaway from the project:

The importance of big data in everyday lives. A better understanding for different models of **sklearn** and **pyspark** such as *HashingVectorizer*, *LabelEncoder*, *MultinomialNB*, *Perceptron*, *SGDClassifier*, *KMeans*

We would like to thank our teachers and TA's for providing us with this opportunity to work and learn from the project and the importance of big data in our everyday lives.