PES UNIVERSITY
UE19CS322 BIG DATA
PROJECT REPORT
Spark Streaming with Machine Learning

Jahnavi Sivaram PES1UG19CS192
Jigisha Narrain PES1UG19CS197
Smruthi G PES1UG19CS488
Suhail F Sheikh PES1UG19CS513
Team ID: BD_192_197_488_513

Project Title Chosen- Spark Streaming with Machine Learning

Design Details-

- We have chosen to take in the streaming dataset and its batch size as command line arguments.
- In our code receiving the RDD through the TCP socket (we have chosen port 6100), we convert each batch to a dataframe before taking any actions on it.
- For convenience, we change all 'Spam' entries to 1 and 'Ham' to 0 in the Spam/Ham column.
- We convert the dataframe into a pandas-like dataframe for easy feature extraction, and then proceed with preprocessing.
- We have utilised Python's NLTK library for standard preprocessing functions - tokenizing, lemmatizing, stemming. From sklearn, we have used the HashVectorizer function to obtain frequencies of words.
- Our modelling is done using sklearn, and incremental learning has been implemented through the partial_fit() function that is present in the same.
- The three algorithms we have modelled are -
    a. Naive Bayes (Bernoulli - for binary classification)
    b. Stochastic Gradient Descent (SGD)
    c. Passive Aggressive Classifier
- All these algorithms are available in scikit-learn and support incremental learning.

Surface level implementation details about each unit-

- <u>Streaming the data</u> - We used the stream.py file provided to take in the train and test datasets for the respective programs. This streaming is done through a TCP socket. The batch size is taken as a command line argument.

- <u>Preprocessing Techniques</u>
    1. Converting target column to binary - Spam and Ham are converted to 0 and 1
    2. Tokenization and punctuation
        Tokenization: Splits sentences into tokens of words
        Punctuation: Removes punctuations from the text
    3. Lowercase conversion - Converts the text into lower case
    4. Lemmatization - Removes the ending part of words and retains only the base word

5. Stemming - Similar to lemmatization for retaining the morphological base words
6. Hash vectorizer - Used to map every word to a numerical value as a vector/array

- Incremental Learning
    1. **partial_fit()**
        - incrementally fits on batches of samples
        - This method is called consecutively for different chunks of the dataset so as to learn incrementally
        - Each time it updates the model with the incremental data
    2. **Pickle files:** Each time the updated models are stored and retrieved from these files

- Modelling - For each of our models (Bernoulli NB, SGD and PAC), we have first used partial_fit to fit each batch incrementally. Then, we've used predict() on the test data to get the predicted Spam/Ham column. After that, we use score() to obtain the accuracy of the predicted values against the true values.

- Clustering **-** We use MiniBatchKMeans for clustering. We take the number of clusters as 2, which is the number of class labels (Spam-1 and Ham-0). Clustering is an unsupervised learning algorithm.

Reason behind design decisions-

- The preprocessing pipeline consists of standard NLP preprocessing techniques, aimed at facilitating predictions and improving the accuracy of our models, using NLTK. The hash vectorizer is used to convert to numerical form and to limit the number of features.
- Incremental learning is implemented using scikit-learn, due to its ease-of-use and it's in-built classification modelling functions. We used partial_fit() to ensure incremental learning with each batch.
- All the models we chose are compatible with incremental learning.
    - Naive Bayes is one of the most popularly used text classification models. It decreases complexity of computation due to its naivety. In streaming analysis in big data, we try to obtain approximate solutions in a short time instead of exact solutions, for which Naive Bayes is an optimal choice.
    - SGD is beneficial in minimising the loss function.
    - The PAC algorithm is generally used for large-scale learning, which is why we have implemented it.
- The MiniBatchKMeans method from scikit facilitates incremental clustering.

Take away from the project-

We familiarised ourselves with NLP techniques and used the same using scikit-learn, NLTK, Spark and incremental learning. We were able to get practice and experience with streaming data analysis and various machine learning models.
