Lab Course: distributed data analytics
Exercise Sheet 8

Nghia Duong-Trung, Mohsan Jameel
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: Friday, June 16, 23:59PM (on LearnWeb, course code: 3117)
Note: this exercise sheet is for the first term students only.

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit a zip or a tar file containing two things a) python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission should be made before the deadline, only through learnweb.
4. If you are M.Sc. Data Analytics summer 2017 intake student, you should submit to "First term students" link on LearnWeb.
5. And if you are M.Sc. Data Analytics winter 2016 intake student, you should submit to "Second term students" link on LearnWeb.
6. If you are not M.Sc. Data Analytics student, you can submit to anyone of the links above.

Exercise 1: Find an image histogram using aggregation (6 points)

First of all, please refer to exercise sheet 3 to remind yourself about what are image histogram and OpenCV. In this exercise, you have to find an image histogram using aggregation in Spark. Along with your code you also have to explain following:

1. Initialize your spark context for gray and RGB color.
2. How you design your sequence and combination operation functions.
3. Implement for both RGB and gray scale histogram.

Remember to include your image example in the submission.

Exercise 2: Machine Learning using Spark MLlib: Decision Tree on Iris dataset (8 points)

In this exercise, you are going to implement a well-known machine learning algorithm, decision tree, on a basic dataset, the Iris, using Spark MLlib. You might find further information about the Iris dataset in [1] and a good tutorial of decision tree classifier [4] on the Iris is provided in [2].

MLlib is Apache Spark's scalable machine learning library which is developed for ease of use, high performance and easy to deploy. Its goal is to make practical machine learning scalable and easy [3].

Along with your code, you might have to follow the proposed procedure.

1. Load the Iris dataset into a RDD and/or a DataFrame using the provided `iris.csv` file.
2. Split data into training and test sets.
3. Create a decision tree model.

4. Train the decision tree model on the training set, predict on the test set.
5. Report evaluation on prediction precision or F1 score.
6. Change the values of hyperparameters in step 3 and 4 to see if you can improve the precision or F1 score.

Exercise 3: Machine Learning using Spark MLlib: Naive Bayes on Spam dataset (6 points)

As you have already learned about Pipeline API and ML workflows in exercise sheet 7. In that exercise, you learn how to design a pipeline for textual data pre-processing with TF-IDF weights. In this exercise, you are going to develop it a step further by integrating your current developed pipeline with a machine learning model to accomplish a machine learning problem.

Naive Bayes classifiers are a popular statistical technique of e-mail filtering. They typically use bag of words features to identify spam e-mail, an approach commonly used in text classification [5]. Naive Bayes classifiers work by correlating the use of tokens (typically words, or sometimes other things), with spam and non-spam e-mails and then using Bayes' theorem to calculate a probability that an email is or is not spam [6].

In this exercise, you are going to develop a pipeline of textual data pre-processing and feed it to Naive Bayes classifiers. The dataset `spam.csv` is used for this exercise. Along with your code, you might have to follow the proposed procedure.

1. Load the `spam.csv`.
2. Split data into training and test sets.
3. Develop a pipeline for textual data pre-processing and Naive Bayes model.
4. Train the Naive Bayes model on the training set, predict on the test set.
5. Report evaluation on prediction precision or F1 score.

Bonus: Aggregation in MongoDB (10 points)

This is the second and also the last bonus for Lab Course. It is optional. You are encouraged to do the bonus that you can gain a better score at the end of the semester. However, if you want to skip it, you are free to do. This bonus section is the following part from the exercise sheet 7's bonus section.

Use dataset `twitter.txt` for this exercise. Complete the following tasks using single aggregation and/or aggregation pipeline.

1. How many times the VIEW, UPDATE and DOWNLOAD actions have been performed in the whole collection? (1 point)
2. Summarize the minimum request time, the average request time, and the maximum request time of VIEW, UPDATE and DOWNLOAD actions in the whole collection. (1 point)
3. Summarize the number of requests, the minimum request time, the average request time, the maximum request time of DOWNLOAD action for each month in the collection. The final output is sorted by the average request time in ascending order. (2 points)
4. How many times the owner whose `ObjectId` is `543f227c0208373c53ba4c4a` performs the UPDATE action? (1 point)
5. Display all `request ip` made by the owner whose `ObjectId` is `543f227c0208373c53ba4c4a`. (1 point)
6. Display all `request ip` together with how many times the `request method SET` has been performed. (1 point)

7. Each document in the collection contains a location field `loc`, e.g. `"loc": [-99.1499, 19.4667]`, where the first value is latitude and the second value is longitude. It means that a particular `request ip` is from a location specified by a latitude and longitude coordinate pair.

   The decimal latitude and longitude coordinates for Hildesheim are 52.15077 and 9.95112 respectively. Let display two documents in the collection that their locations are the nearest to Hildesheim's location. The distance should be also calculated and reported. (3 points)

Annex

1. Sklearn iris visualization `http://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html`
2. Decision Tree classifier Explanation & Example using Iris dataset `https://codingmachinelearning.wordpress.com/2016/06/23/decision-tree-classifier-explanation-example-using-iris-dataset-6/`
3. MLlib official page `http://spark.apache.org/docs/latest/ml-guide.html`
4. How To Implement The Decision Tree Algorithm From Scratch In Python `http://machinelearningmastery.com/implement-decision-tree-algorithm-scratch-python/`
5. 6 Easy Steps to Learn Naive Bayes Algorithm (with code in Python) `https://www.analyticsvidhya.com/blog/2015/09/naive-bayes-explained/`
6. How To Build a Simple Spam-Detecting Machine Learning Classifier `https://hackernoon.com/how-to-build-a-simple-spam-detecting-machine-learning-classifier-4471fe6b816e`