Data-intensive Programming 2022–2023

Programming Assignment

Version 1.0.1

Description

The course has a compulsory programming assignment that is done in groups of two students. Working alone is also ok. Groups are created in Moodle, and you need to create a group even if you work alone. When creating a group, you must agree on the programming language, Scala, or Python, that you want to use in the assignment. There are basic tasks that must be implement in order achieve accepted assignment. In addition, there are additional tasks that can give up to 5 course points. And finally, a bonus point is available for those groups who have implemented all the basic tasks with Scala.

Deadline for the grouping is 10th of November.

Code repositories for the groups will be generated by the course staff after the grouping deadline. The repositories are created in https://course-gitlab.tuni.fi/. The data, a template for the assignment, and a test suite are published via GitLab. The template project is not expected to update often so the course personnel will notify you if it is changed.

You are given a data set which, in general, looks like this:

```
a,b,LABEL
0.83039,4.63513,Fatal
-0.013,1.39797,Ok
-0.28272,5.70507,Fatal
…
```

Even if the amount of data is not actually that big, you should write your implementation as if it were. Moreover, Spark should be programmed with Scala or Python.

The assignments are first evaluated with scale fail or pass. If you fail, you need to fix your submission. The group should commit a working project in the group's repository for submission. The code should be commented on at least to some extent.

A basic passed solution implements the basic tasks 1–4. Such a solution is accepted but will not give any points. However, it is possible to gain course points by implementing the additional tasks. The points will be added to the total sum along with the exam and weekly exercises points. See the Grading section in the course Moodle page for details about the passing requirements.

A simple test suite is given for testing your solution. You can add your own tests to the suite.

Submission

The assignment is submitted by committing the source files into the group's repository and submitting the latest commit hash to Moodle. In addition, mention the programming language you used and if you are aiming for the bonus points, mention which additional tasks you have done. The Moodle submission page will contain an example on how to give the required information.

The deadline for the submission is by the end of Sunday 11th of December.

Compulsory basic tasks

Basic task 1: Basic 2D K-means

The task is to implement k-means clustering with Apache Spark for two-dimensional data. Example data can be found from file `data/dataD2.csv` (ignore the LABEL column in this task). The task is to compute cluster means using DataFrames and MLlib. The number of means (`k`) is given as a parameter. Data for k-means algorithm should be scaled, but it is not required to scale the resulting cluster centers back to the original scale to complete this basic task (see Additional Task #6).

See `task1` in the `Assignment.scala` or `assignment.py` file.

Basic task 2: Three Dimensions

The task is to implement k-means clustering with Apache Spark for three-dimensional data. Example data can be found in file `data/dataD3.csv` (ignore the LABEL column in this task). The task is to compute cluster means with DataFrames and MLlib. The number of means (`k`) is given as a parameter. Remember to scale your data for the algorithm similarly to task 1.

See `task2` in the `Assignment.scala` or `assignment.py` file.

Basic task 3: Using Labels

K-means clustering has been used in medical research. For instance, our example data could model some laboratory results of patients and the label could imply whether he/she has a fatal condition or not. The labels are `Fatal` and `Ok`.

Use two-dimensional data (like in the file `data/dataD2.csv`), map the LABEL column to a numeric scale, and store the resulting data frame to `dataD2WithLabels` variable. And then, cluster in three dimensions (including columns `a`, `b`, and the numeric value of `LABEL`) and return two-dimensional clusters means (the values corresponding to columns `a` and `b`) for those two clusters that have the largest count of Fatal data points. You can assume that the input data frame and the total number of clusters (`k`) is selected in such a way that there will always be at least 2 cluster centers which contain Fatal data points. Remember to scale your data similarly to task 1.

See `task3` in `Assignment.scala` or `assignment.py` file.

Basic task 4: Silhouette Method

The silhouette method can be used to find the optimal number of clusters in the data. Implement a function which returns an array of (`k`, `score`) pairs, where `k` is the number of clusters, and `score` is the silhouette score for the clustering. You can assume that the data is given in the same format as the data for Basic task 1, i.e., two-dimensional data with columns `a` and `b`.

See `task4` in `Assignment.scala` or `assignment.py` file.

Additional tasks

Additional task 1: Functional style – 0.5 Points

Try to write your code in functional programming style. Use, for example, immutable variables, pure functions, higher-order functions, recursion, and mapping. Avoid looping through data structures. It is quite possible that you will achieve this task without much effort just by following the style used in the course material.

Additional task 2: Efficient usage of data structures – 1 Point

Use data structures efficiently. For example:

- Use caching or persisting if it is sensible.
- Consider defining schemas instead of inferring them.
- Avoid unnecessary operations.
- Adjust the amount of shuffle partitions if it is sensible. Reason in comments why or why not to adjust the amount of shuffle partitions.

Additional task 3: Dirty data – 1 Point

The program should handle dirty or erroneous data somehow. Add a few additional test cases using erroneous data files. You can use the file `data/dataD2_dirty.csv` as an example on what a dirty data could look like, but you can also create your own erroneous data files.

Additional task 4: ML (Machine Learning) pipeline – 0.5 Points

Chain your ML tasks as a ML pipeline with multiple stages. (For example, VectorAssembler, MinMaxScaler, Kmeans). To get the point from this task you have to use ML pipelines when calculating the results in the basic tasks.

Additional task 5: Visualization – 1 Point

Make programmatically a graph that presents the silhouette score as a function of k (the result of Basic task 4). You can save the graph as an image or make your application wait for a while when the graph is visible.

With Scala the following two libraries have been tested to work with the assignment template:

- breeze-viz (version 1.3): https://github.com/scalanlp/breeze
  - Spark already includes the breeze library, only breeze-viz library needs to be added
- nspl (version 0.5.0): https://github.com/pityka/nspl

With Python you can use, for example use the matplotlib library: https://pypi.org/project/matplotlib/

Additional task 6: Scaling back to original scale - 1 Point

Scale the cluster centers back to the original scale. I.e., if the values for column `X` were originally in the range `[X_min, X_max]`, then the returned cluster centers should also be in that same range. To get the point from this task all the basic tasks must include this feature and return the cluster centers in the original scale.

Bonus point

Usage of Scala – 1 Point

One point will be given for implementing all the basic tasks in the assignment with Scala.
You can get this bonus point by fulfilling one of the following conditions:

a) you have used Scala as the main language and implemented the basic tasks and the additional tasks of your choice with Scala
b) you have used Python as the main language and implemented the basic tasks and the additional tasks of your choice with Python; **and** you have implemented the basic tasks with Scala