Project 4

New Attempt

Due Oct 28 by 11:59pm **Points** 100 **Submitting** a file upload

KMeans Clustering on Spark

Description

The purpose of this project is to develop a data analysis program using Apache Spark.

This project must be done individually. No copying is permitted. **Note: We will use a system for detecting software plagiarism, called Moss (http://theory.stanford.edu/~aiken/moss/), which is an automatic system for determining the similarity of programs.** That is, your program will be compared with the programs of the other students in class as well as with the programs submitted in previous years. This program will find similarities even if you rename variables, move code, change code structure, etc.

Note that, if you use a Search Engine to find similar programs on the web, we will find these programs too. So don't do it because you will get caught and you will get an F in the course (this is cheating). Don't look for code to use for your project on the web or from other students (current or past). Just do your project alone using the help given in this project description and from your instructor and GTA only.

Platform

As in the previous projects, you will develop your program on SDSC Expanse. Optionally, you may use your laptop or IntelliJIdea or Eclipse to help you develop your program, but you should test your programs on Expanse before you submit them.

Using your laptop to develop your project

You may use your laptop to develop your program and then test it and run it on Expanse.

To install Spark and the project on your laptop:

```
cd
wget https://downloads.apache.org/spark/spark-3.1.2/spark-3.1.2-bin-hadoop3.2.tgz
tar xfz spark-3.1.2-bin-hadoop3.2.tgz
wget http://lambda.uta.edu/cse6331/project4.tgz
tar xfz project4.tgz
```

To compile and run project4 on small data:

```
cd project4
mvn install
rm -rf output
~/spark-3.1.2-bin-hadoop3.2/bin/spark-submit --class KMeans target/cse6331-project4-0.1.jar points-small.txt centroids.txt
```

Your output should be similar to (but not necessarily the same as) the results in `solution-small.txt`. You may also run it for large data:

```
~/spark-3.1.2-bin-hadoop3.2/bin/spark-submit --class KMeans target/cse6331-project4-0.1.jar points-large.txt centroids.txt
```

Your output should be similar to (but not necessarily the same as) the results in `solution-large.txt`.

Installing the Project on Expanse

Login into Expanse and download and untar project4:

```
wget http://lambda.uta.edu/cse6331/project4.tgz
tar xfz project4.tgz
chmod -R g-wrx,o-wrx project4
```

Go to project4/examples and look at the Spark example JoinSpark.scala. You can compile JoinSpark.scala using:

```
run joinSparkScala.build
```

and you can run it in local mode using:

```
sbatch joinSpark.local.run
```

File join.local.out will contain the trace log of the Spark evaluation.

Project Description

You are asked to implement one step of the Lloyd's algorithm for K-Means clustering using Spark and Scala. The goal is to partition a set of points into k clusters of neighboring points. It starts with an initial set of k centroids. Then, it repeatedly partitions the input according to which of these centroids is closest and then finds a new centroid for each partition. That is, if you have a set of points P and a set of k centroids C, the algorithm repeatedly applies the following steps:

1. Assignment step: partition the set P into k clusters of points P_i, one for each centroid C_i, such that a point p belongs to P_i if it is closest to the centroid C_i among all centroids.
2. Update step: Calculate the new centroid C_i from the cluster P_i so that the x,y coordinates of C_i is the mean x,y of all points in P_i.

The datasets used are random points on a plane in the squares (i*2+1,j*2+1)-(i*2+2,j*2+2), with 0≤i≤9 and 0≤j≤9 (so k=100 in k-means). The initial centroids in `centroid.txt` are the points (i*2+1.2,j*2+1.2). So the new centroids should be in the middle of the squares at (i*2+1.5,j*2+1.5).

In this project, you are asked to implement one step of the K-means clustering algorithm using Spark and Scala. A skeleton file `project4/src/main/scala/KMeans.scala` is provided, as well as scripts to build and run this code on Expanse. **You should modify KMeans.scala only**. Your main program should take two arguments: the text file that contains the points (points-small.txt or points-large.txt) and the centroids.txt file. The resulting centroids will be written to the output. This time, the process of finding new centroids from previous centroids using KMeans must be repeated 5 times. Note: you need to broadcast the centroids to worker nodes using the Spark broadcast method (see **Broadcast Variables (https://sparkbyexamples.com/spark/spark-broadcast-variables/)** on p58 in the class slides):

```
centroids = /* initial centroids from the file centroids.txt */

for ( i <- 1 to 5 ) {
   val cs = sc.broadcast(centroids)
   centroids = points.map { p => (cs.value.minBy(distance(p,_)), p) }
                     .groupByKey().map { /* ... calculate a new centroid ... */ }
}
```

where `distance(x,y)` calculates the distance between two points x and y.

You can compile KMeans.scala on Expanse using:

```
run kmeans.build
```

and you can run it in local mode over the small data using:

```
sbatch kmeans.local.run
```

You should modify and run your programs in local mode until you get the correct result. Your output should be similar to (but not necessarily the same as) the results in `solution-small.txt`. After you make sure that your program runs correctly in local mode, you run it in distributed mode using:

```
sbatch kmeans.distr.run
```

This will work on the moderate-sized data and will print the results to the output. Your output should be similar to (but not necessarily the same as) the results in `solution-large.txt`.

Optional: Use an IDE to develop your project

If you have a prior good experience with an IDE (IntelliJ IDEA or Eclipse), you may want to develop your program using an IDE and then test it and run it on Expanse. Using an IDE is optional; you shouldn't do this if you haven't used an IDE before.

On IntelliJ IDEA, go to New→Project from Existing Sources, then choose your project4 directory, select Maven, and the Finish. To compile the project, go to Run→Edit Configurations, use + to Add New Configuration, select Maven, give it a name (eg, build), use Working directory: your project4 directory, Command line: install, then Apply. To run your project in local mode, you need to add the line conf.setMaster("local[2]") in the main program before you create SparkContex (you should remove this line before you test your project on Expanse). Go to Run→Edit Configurations, use + to Add New Configuration, select Application, give it a name (eg, run), use the Main class: KMeans, Program arguments: points-small.txt centroids.txt.

On Eclipse, you first need to install **m2e (https://projects.eclipse.org/projects/technology.m2e)** (Maven on Eclipse), if it's not already installed. Then, install Scala on Eclipse from **scala-ide.org (http://scala-ide.org/download/current.html)** using Install New Software... and then cut-and-paste the update site URL. Then go to Open File...→Import Project from File System, then choose your project4 directory. To compile your project, right click on the project name at the Package Explorer, select Run As, and then Maven install. To run your project in local mode, you need to add the line conf.setMaster("local[2]") in the main program before you create SparkContex (you should remove this line before you test

your project on Expanse). Right-click on KMeans.java→Run As→Run Configurations, select Scala Application, press the New button to create a new configuration, give it a name (eg, run), add the main class KMeans, and go to Arguments. Add 2 arguments: points-small.txt and centroids.txt (separate lines). Now you can run it in local mode by hitting Run.

Documentation

You can learn more about Scala at:

- **A Scala Tutorial for Java Programmers (http://docs.scala-lang.org/tutorials/scala-for-java-programmers.html)**
- **A Tour of Scala (https://docs.scala-lang.org/tour/tour-of-scala.html)**

You can learn more about Spark at:

- **Spark Quick Start (http://spark.apache.org/docs/latest/quick-start.html)**
- **Spark Programming Guide (http://spark.apache.org/docs/latest/programming-guide.html)**
- **Spark by Examples (https://sparkbyexamples.com/)**
- **RDD API (http://spark.apache.org/docs/latest/api/scala/org/apache/spark/rdd/RDD.html)**
- **PairRDDFunctions API (http://spark.apache.org/docs/latest/api/scala/org/apache/spark/rdd/PairRDDFunctions.html)**

What to Submit

Submit the zipped project4 directory, which must contain the files:

```
project4/src/main/scala/KMeans.scala
project4/kmeans.local.out
project4/kmeans.distr.out
```