ITCS 3190– Cloud Computing For Data Analysis
Project 2
Due by 11:59:59pm on Sunday, April 7, 2019

First task: *Running wordcount in Spark*

Step 1:
To connect to the dsba-hadoop cluster (from a Unix machine)
$ ssh dsba-hadoop.uncc.edu -l <username>

For example $ ssh dsba-hadoop.uncc.edu -l dyang33

Step 2:

Before you run the sample, you must create input locations in HDFS. Use the following commands to create the input directory/user/<username>/input in HDFS:

$ **hadoop fs -mkdir /user/<username>/input**

And then put input file alice29.txt to your input directory in HDFS.
$ **hadoop fs -put alice29.txt /user/<username>/input**
Then use the -ls command to check if the input file exist or not.
$ **hadoop fs -ls /user/<username>/input**

Step 3:
Run the wordcount.py program, passing the paths to the input directories in HDFS:
$ **spark2-submit wordcount.py /user/<username>/input/alice29.txt**

Step 4:

Take a screenshot of resluts.

This is the first thing you should submit!!

Second task: Implement Linear regression on Spark:

1. Introduction to linear model

Suppose the data consists of n observations $\{\boldsymbol{y}_i, \boldsymbol{x}_i\}_{i=1}^{n}$. Each observation i includes a scalar response $\boldsymbol{y}_i$ and a column vector $\boldsymbol{x}_i$ of values of p predictors (regressors) x_{ij} for j = 1, ..., p. In a linear regression model, the response variable, $\boldsymbol{y}_i$, is a linear function of the regressors:

$$y_i = \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i ,$$

where β is a $p\times1$ vector of unknown parameters; the ε_i are unobserved scalar random variables (errors) which account for influences upon the responses $\boldsymbol{y_i}$ from sources other than the explanators x_i; and $\boldsymbol{x_i}$ is a column vector of the ith observations of all the explanatory variables. This model can also be written in matrix notation as:

$$y = X\beta + \varepsilon,$$

where y and ε are $n\times1$ vectors of the values of the response variable and the errors for the various observations, and X is an $n\times p$ matrix of regressors, also sometimes called the design matrix, whose row i is x_i^T and contains the ith observations on all the explanatory variables.

As a rule, the constant term is always included in the set of regressors X, say, by taking $x_{i1} = 1$ for all $i = 1, \ldots, n$. The coefficient β_1 corresponding to this regressor is called the *intercept*.

Consider an overdetermined system

$$\sum_{j=1}^{p} X_{ij}\beta_j = y_i,\ (i = 1, 2, \ldots, n),$$

of n linear equations in p unknown coefficients, $\beta_1,\beta_2,\ldots,\beta_p$, with $n > p$. (Note: for a linear model as above, not all of contains information on the data points. The first column is populated with ones, , only the other columns contain actual data, so here p = number of regressors + 1.) This can be written in matrix form as

$$\mathbf{X}\boldsymbol{\beta} = \mathbf{y},$$

where

$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & X_{22} & \cdots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{np} \end{bmatrix}, \qquad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}, \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}.$$

Such a system usually has no solution, so the goal is instead to find the coefficients $\boldsymbol{\beta}$ which fit the equations "best", in the sense of solving the quadratic minimization problem

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\arg\min}\, S(\boldsymbol{\beta}),$$

where the objective function S is given by

$$S(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left| y_i - \sum_{j=1}^{p} X_{ij}\beta_j \right|^2 = \left\| \mathbf{y} - \mathbf{X}\boldsymbol{\beta} \right\|^2.$$

A justification for choosing this criterion is given in properties below. This minimization problem has a unique solution, provided that the p columns of the matrix **X** are linearly independent, given by solving the normal equations

$$(\mathbf{X}^{\mathrm{T}}\mathbf{X})\hat{\boldsymbol{\beta}} = \mathbf{X}^{\mathrm{T}}\mathbf{y}.$$

The matrix $\mathbf{X}^\mathbf{T}\mathbf{X}$ is known as the Gramian matrix of **X**, which possesses several nice properties such as being a positive semi-definite matrix, and the matrix $\mathbf{X}^\mathbf{T}\mathbf{Y}$ is known as the moment matrix of regressand by regressors. Finally, $\widehat{\boldsymbol{\beta}}$ is the coefficient vector of the least-squares hyperplane**, the closed form expression for the ordinary least squares estimate of the linear regression coefficient** can be expressed as

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}.$$

Then X^T= $[x_1, x_2, \ldots\ x_\mathrm{n}]$, and $X = \begin{bmatrix} x_1^T \\ x_2^T \\ \ldots \\ x_\mathrm{n}^T \end{bmatrix}$. So $X^T X$ will be:

Let a = $X^T X = x_1 * x_1^T + x_2 * x_2^T + \ldots + x_\mathrm{n} * x_\mathrm{n}^T$ = $\boldsymbol{\sum_i^n x_\mathrm{i} * x_1^T}$**.**

Correspondingly, b = $X^T Y = \ x_1 * y_1 + x_2 * y_2 + \ldots + x_\mathrm{n} * y_\mathrm{n}$ = $\boldsymbol{\sum_i^n x_\mathrm{i} * y_\mathrm{i}}$**.**

Then, the beta will be: $\widehat{\boldsymbol{\beta}}$ **=a**$^{-1}$***b** $= (\sum_\mathrm{i}^\mathrm{n} \mathbf{x}_\mathrm{i} * \mathbf{x}_1^\mathbf{T})^{-1}(\sum_\mathrm{i}^\mathrm{n} \mathbf{x}_\mathrm{i} * \mathbf{y}_\mathrm{i})$**.**

2. Your task

Implement linear regression in MapReduce logic using Spark. You are required to compute $\hat{\beta}$ for datasets from both yxlin.csv and yxlin2.csv seperately. You will use the closed form expression for the ordinary least squares estimate of the linear regression coefficient to compute $\hat{\beta}$ in linreg.py:

Step 1:
Read the dataset. The first column will be ***y*** values and the second column will be ***x*** value. Partition the dataset into a number of subsets $(x_1, y_1), \ldots (x_n, y_n)$.

Step 2:
You should implement Map functions to take (x_i, y_i) as input:
Assign the same key **a** to $\boldsymbol{x_i * x_i}$, ***Emits (Key a,*** $\boldsymbol{x_i * x_i}$***)***
Assign the same key **b** to $\boldsymbol{x_i * y_i}$, ***Emits (Key b,*** $\boldsymbol{x_i * y_i}$***)***

Tips: you might use this operation: map(lambda x: x*x)

Step 3:
You should implement Reduce functions to group the results by key:

Group by key **a**, and compute **a** = $\sum_i^n x_\mathrm{i} * x_1^T$.
Group by key **b**, and compute **b** = $\sum_i^n x_\mathrm{i} * y_\mathrm{i}$.
Tips: you might use this operation: reduceByKey(lambda x,y: x+y).collect()

Step 4:
Then compute $\hat{\beta} = a^{-1} * b$ and print the result.

Submissions:
Upload to Canvas a zip file named your_uncc_id.zip (e.g. dyang33.zip if your email address is dyang33@uncc.edu). The file should contain:

1. Screenshot from workcount.py on alice29.txt.
2. Screenshot from linreg.py on yxlin.csv
3. Screenshot from linreg.py on yxlin2.csv
4. Source code for linreg.py

Guidelines:

Please make sure your source code is adequately commented, and also make sure each of your files has your name and email id included at the top of the file.

Assignments are to be done individually. See course syllabus for late submission policy and academic integrity guidelines.

Please make sure your screenshot includes your account information. For example:

```
tinkling: 1
lasted: 1
rule: 2
pictured: 1
questions,: 1
needn't: 3
not!': 1
savage!': 1
Grief,: 1
trial,: 1
feathers,: 1
ordered';: 1
lives: 1
skimming: 1
undertone: 1
`You!': 1
19/03/19 21:29:03 INFO server.AbstractConnector: Stopped Spark@64ea36b3{HTTP/1.1,[http/1.1]}{0.0.0.0:4041}
19/03/19 21:29:03 INFO ui.SparkUI: Stopped Spark web UI at http://mba-i2.uncc.edu:4041
19/03/19 21:29:03 INFO cluster.YarnClientSchedulerBackend: Interrupting monitor thread
19/03/19 21:29:03 INFO cluster.YarnClientSchedulerBackend: Shutting down all executors
19/03/19 21:29:03 INFO cluster.YarnSchedulerBackend$YarnDriverEndpoint: Asking each executor to shut down
19/03/19 21:29:03 INFO cluster.SchedulerExtensionServices: Stopping SchedulerExtensionServices
(serviceOption=None,
 services=List(),
 started=false)
19/03/19 21:29:03 INFO cluster.YarnClientSchedulerBackend: Stopped
19/03/19 21:29:04 INFO spark.MapOutputTrackerMasterEndpoint: MapOutputTrackerMasterEndpoint stopped!
19/03/19 21:29:04 INFO memory.MemoryStore: MemoryStore cleared
19/03/19 21:29:04 INFO storage.BlockManager: BlockManager stopped
19/03/19 21:29:04 INFO storage.BlockManagerMaster: BlockManagerMaster stopped
19/03/19 21:29:04 INFO scheduler.OutputCommitCoordinator$OutputCommitCoordinatorEndpoint: OutputCommitCoordinator stopped!
19/03/19 21:29:04 INFO spark.SparkContext: Successfully stopped SparkContext
19/03/19 21:29:04 INFO util.ShutdownHookManager: Shutdown hook called
19/03/19 21:29:04 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-81e75d9d-1908-468c-be60-8d64b8b7b3c7
19/03/19 21:29:04 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-f2b51cab-cf0c-4509-b56f-b1c045949fde
19/03/19 21:29:04 INFO util.ShutdownHookManager: Deleting directory /tmp/spark-81e75d9d-1908-468c-be60-8d64b8b7b3c7/pyspark
[dyang33@mba-i2 ~]$ 
```

Grading Rubric:

Total 100.

1. **Output from running wordcount.py (30 pts).**
2. **Output from running linreg.py using yxlin.csv as input. (20 pts).**
3. **Output from running linreg.py using yxlin2.csv as input. (20 pts).**
4. **Source code linreg.py (30 pts).**