Big Data - Fall 2019
Homework Assignment #2
Responsible Data Mining
Due: 9:00AM, Oct 21, 2019
NO LATE SUBMISSIONS WILL BE ACCEPTED

Goal

This assignment is about responsible data mining. In the first part of this assignment, you will solve some analytical problems using the Gradiance platform. In the second part of this assignment you are asked to analyze a post-processed version of the ProPublica dataset that is provided for you, by implementing a SparkSQL program.

Grading

This assignment is worth 100 points or 10% of the overall course grade. If this assignment is submitted late, you will receive no credit.

Part 1: Gradiance is worth 30 points
Part 2: SparkSQL is worth 70 points

This assignment is to be completed individually. Please consult the course syllabus for a description of our academic honesty policy.

Part 1: Gradiance (30 points)

In this assignment you will be using a free online system called Gradiance to work on the problem and to submit your answers. To access Gradiance, create an account at http://www.newgradiance.com/services/servlet/COTC, using your NYU email address with ID to sign up for the service. If your id is less than 6 characters, pad it with *x* on the right. For example, if your id is *ab123*, you should use *ab123x* as your Gradiance username.

Then you can use your account to log on to the system. Having logged on, sign up for a new class, specifying the following class token.

Big Data Section A (Prof. Stoyanovich): your class token is **7975FA43**
Big Data Section B (Prof. Freire): your class token is **538FD953**

When you enter the token above, the appropriate section of CS-GY6513 will appear under "Your Classes". Click on the link for CS-GY6513, and then follow the link to homeworks in the menu on the left. You will see the currently assigned homework,

and will be able to open it and submit solutions. Gradiance will automatically grade your submissions and record your scores. **You do not need to submit anything to us for this part of the assignment. We will retrieve your scores directly from Gradiance.**

You may submit your assignment as many times as you wish before the deadline. Whenever you answer a question incorrectly, an explanation will be provided to help you. The score you receive on your last attempt is the score that will count. All questions carry an equal number of points.

(Note: for the gradiance question about evil-doers, ignore "as in the slides for the 1/9/08 lecture")

Part 2: SparkSQL (70 points)

Details

To get started, study the formulation of classification rules (CARS) and of potentially discriminatory rules (PD-CARs) in the data mining lecture slides. Also familiarize yourself with ProPublica's "Machine Bias", available at https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing .

Your goal in this assignment is to compute CARs that have *vscore* on the right-hand-side, and one or several of the following attributes on the left-hand-side: *race, age*, *gender* or *marriage*. For example, you may compute the rules:

age=30, gender=1 => vdecile=1
age=40, race=1 => vdecile=2

But you should **not** compute the rules:

gender=0 vdecile=1 => age=30
age=30 => marriage=2

In other words, all rules you produce must have vdecile=1 or vdecile=2 on the right hand side, and should not have either vdecile=1 or vdecile=2 on the left hand side.

We provide a template for **hw2sql.py**, along with two shell scripts you should use to run and test your code. Your implementation must be entirely in hw2sql.py**.** You should add to this script, but you should not remove or change any code in hw2sql.py.

Your program must place all computed rules into relations **R2**, **R3** and **PD_R3**. R2 and R3 should each contain classification association rules of sizes 2 and 3, respectively, with vdecile=0 or vdecile=1 on the right. Only output rules that pass the support and confidence thresholds.

For each rule in R2 and R3, compute and output its support and confidence. These should be placed into the appropriate columns in the output relations.

PD_R3 should contain a subset of the rules from R3. Each rule in PD_R3 (1) has race as one of the attributes on the left-hand-side and (2) meets the protection threshold.

Make sure that, for relations R2 and R3, attribute names are stored in alphabetical order. That is, if your rule contains attributes age and gender on the left-hand-side, then it should be the case that R3.attr1 = 'age', and R3.attr2='gender', in this specific order.
You should not explicitly mention by name (hard-code) attributes other than vdecile and race anywhere in your code. Assume that your code should still work if another attribute is added, e.g., income or education. It is OK to refer to race and vdecile by name, since these attributes have special meaning.

Your grade will be based on a combination of correctness (the content of the final output relations R2, R3 and PD_R3) and efficiency, where by efficiency we mean the size of the intermediate result sets.

Code, Data, and Submission Instructions

You will submit your Spark SQL program as a single python file named **hw2sql.py**. You do not need to submit the input to your program or the output of your program, only submit the python code itself.

We provide a template for **hw2sql.py**, along with two shell scripts you should use to run and test your code. To get the template and the scripts, log on to **dumbo** and execute:

```
[jds405@login-2-1 hw2]$ hfs -get /user/jds405/HW2code/*

[jds405@login-2-1 hw2]$ chmod u+x test.sh testall.sh hw2sql.py

[jds405@login-2-1 hw2]$ ls -ltr
total 16
-rwxr--r-- 1 jds405 users  917 Oct  7 10:48 test.sh
-rwxr--r-- 1 jds405 users  179 Oct  7 10:48 testall.sh
-rwxr--r-- 1 jds405 users 6670 Oct  7 10:49 hw2sql.py
```

You should implement the required functionality in **hw2sql.py**. **Your submission should run on dumbo when invoked with testall.sh.** If your program does not run in this way, you will not receive full credit.

Take a look at both **test.sh** and **testall.sh** to understand how to invoke **hw2sql.py** as you implement / debug your program.

We are also providing you with two test cases and correct output. These test cases are computed over the /user/jds405/HW2data/**PP_small.csv** dataset. When we test your submission, we will check that these two test cases produce the correct output. We will also test your code on additional datasets and combinations of parameters.

The input and output datasets are available to you on HDFS. You may copy these files to your local directory and inspect them.

```
[jds405@login-2-1 sandbox]$ hfs -ls /user/jds405/HW2data
Found 3 items
-rw-r--r--+  3 jds405 users    1995913 2019-10-06 16:39 /user/jds405/HW2data/PP_large.csv
-rw-r--r--+  3 jds405 users     400750 2019-10-06 16:38 /user/jds405/HW2data/PP_small.csv
drwxr-xr-x+  - jds405 users          0 2019-10-06 17:10 /user/jds405/HW2data/results
```

```
[jds405@login-2-1 sandbox]$ hfs -ls /user/jds405/HW2data/results
Found 2 items
drwxr-xr-x+  - jds405 users          0 2019-10-06 17:10 /user/jds405/HW2data/results/res_500_055_1
drwxr-xr-x+  - jds405 users          0 2019-10-06 17:11 /user/jds405/HW2data/results/res_600_07_05
```

Note in particular that there are two subdirectories under results: /user/jds405/HW2data/results/res_500_055_1 contains results of executing hw2sql.py with support=500, confidence=0.55, protection=1.

Similarly, /user/jds405/HW2data/results/res_600_07_05 contains results for support=600, confidence=0.7, protection-0.5.