CSE8803: Big Data Analytics in Healthcare
Homework 3

Jimeng Sun

Deadline: February 28, 2016

- Discussion is encouraged, but each student must write his/her own answers and explicitly mention any collaborators.
- Each student is expected to respect and follow the GT Honor Code.
- Please type the submission with LaTeX or Microsoft Word. We **will not** accept hand written submissions.
- Please **do not** change the filenames and function definitions in the skeleton code provided, as this will cause the test scripts to fail and subsequently no points will be awarded.

Overview

Accurate knowledge of a patient's disease state is crucial which requires knowing accurate phenotypes about patients based on their electronic health records. There are several strategies for phenotyping, including supervised rule-based methods as well as unsupervised methods. In this homework, you will implement both type of phenotyping algorithms. You will be required to implement these using Spark.

Prerequisites [0 points]

This homework is mainly about Spark with Scala. You can download and install spark on your local machine by following instructions on http://spark.apache.org/docs/1.3.1/ if you want to to try spark-shell. Programming assignments in this homework actually doesn't require Spark installation. We suggest you develop the code using IDE like IntelliJ(free community edition is enough for developing homework code, but as a student, you can get education license for full edition).

For programming problem, you will be given the code skeleton from a GT GitHub repo. You need to download the skeleton by

```
cd hw3 #navigate to hw directory
git clone https://github.gatech.edu/hsu34/bdh-hw3.git code
```

Then you need to download data from S3, unzip that and put that into your code directory

```
cd code
wget https://s3.amazonaws.com/cse8803/2016hw3/data.tar.gz
tar -zxvf data.tar.gz
```

Note that the data folder should be inside your code folder.

If you are a mac user you should be able to use below command to compile and run the code by

```
sbt/sbt compile run
```

Otherwise, you will need to refer to SBT installation manual to update *sbt/sbt* script first. Then you can call in above way.

1 Programming: Rule based phenotyping [30 points]

Phenotyping can be done using a rule-based method. The Phenotype Knowledge Base (PheKB) (https://phekb.org) provides a set of rule-based methods (typically in the form of decision trees) for determining whether or not a patient fits a particular phenotype.

Please implement a phenotyping algorithm for type-2 diabetes based on the flowchart below that

- takes as input event data for diagnoses, medications, and lab results and
- return RDD of patients with labels (*label*=1 if the patient is case, *label*=2 if control, 3 otherwise).

Implement the following algorithms from PheKB (We have reduced the rules for simplicity. Thus, you should follow this simplified flowchart for your homework description, and refer to this for more details http://jamia.oxfordjournals.org/content/19/2/219.long):

- Diabetes Mellitus Type 2

The following files in *data* folder will be used as inputs:

- **encounter_INPUT.csv**: Each line represents an envounter. The encounter ID and the patient ID (Member ID) are separate columns. *Hint: sql join*
- **encounter_dx_INPUT.csv**: Each line represents an encounter. The disgnoses (ICD9 codes) are in this file.

- **medication_orders_INPUT.csv**: Each line represents a medication order. The name of medication is found in one of the columns on this file.
- **lab_results_INPUT.csv**: Each line represents a lab result. The name of the lab (use 'Result_Name' column), the units for the lab, and the value for the lab are found in specific columns on this file.

You are responsible for transforming above CSV files into RDDs.

The detailed rules for phenotyping of Diabetes Mellitus Type 2 are shown below. These rules were based off of the criteria from the PheKB phenotypes, which have been placed in the folder `/phenotyping_resources/`. As it is stated above, these rules were simplified for this homework.

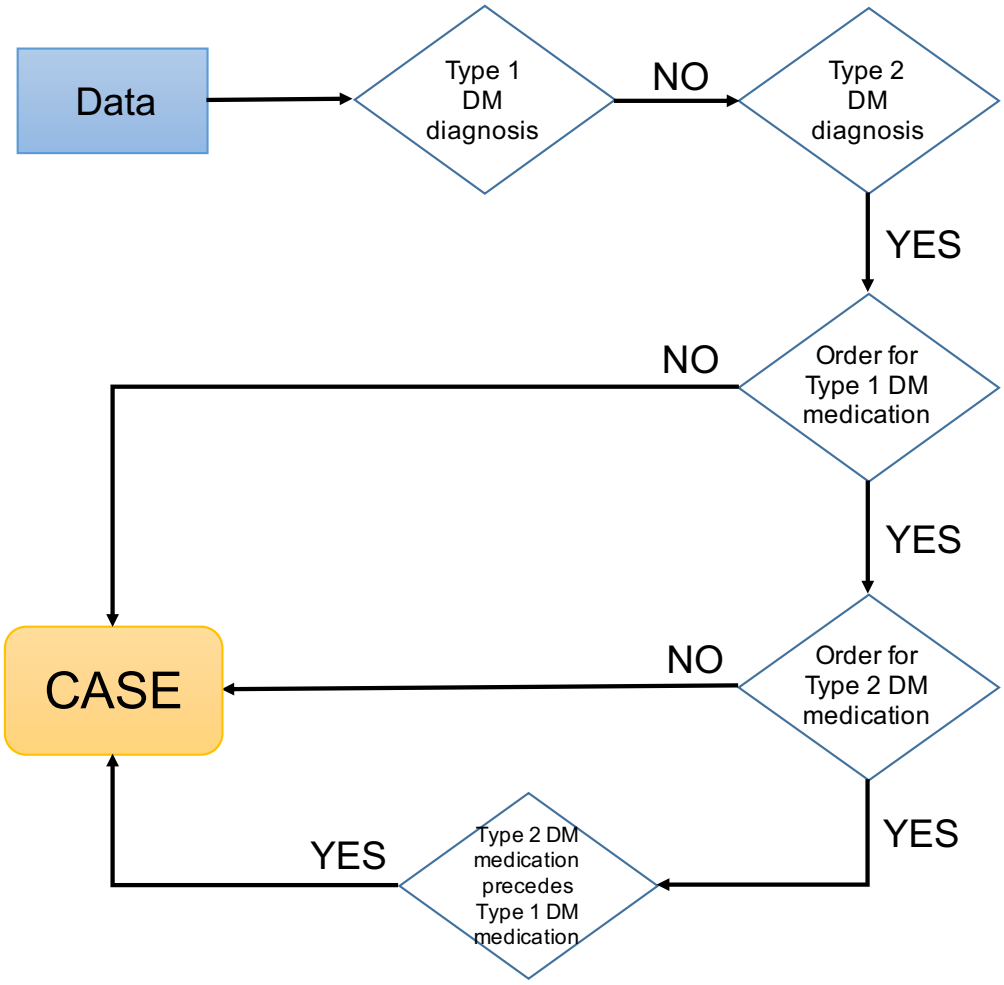

Figure 1: Determination of cases

- **Requirements for Case patients**: Figure 1 details the rules for determining whether a patient is case. Certain parts of the flowchart involve criteria that you will find the handout folder `/phekb_criteria/`.

- `/phekb_criteria/T1DM_DX.csv`: Any of the ICD codes present in this file will be sufficient to result in `YES` for the Type 1 DM diagnosis criteria.
- `/phekb_criteria/T1DM_MED.csv`: Any of the medications present in this file will be sufficient to result in `YES` for the Order for Type 1 DM medication criteria. Please also use for the criteria Type 2 DM medication precedes Type 1 DM medications.
- `/phekb_criteria/T2DM_DX.csv`: Any of the ICD codes present in this file will be sufficient to result in `YES` for the Type 2 DM diagnosis criteria.
- `/phekb_criteria/T2DM_MED.csv`: Any of the medications present in this file will be sufficient to result in `YES` for the Order for Type 2 DM medication criteria. Please also use for the criteria Type 2 DM medication precedes Type 1 DM medications.

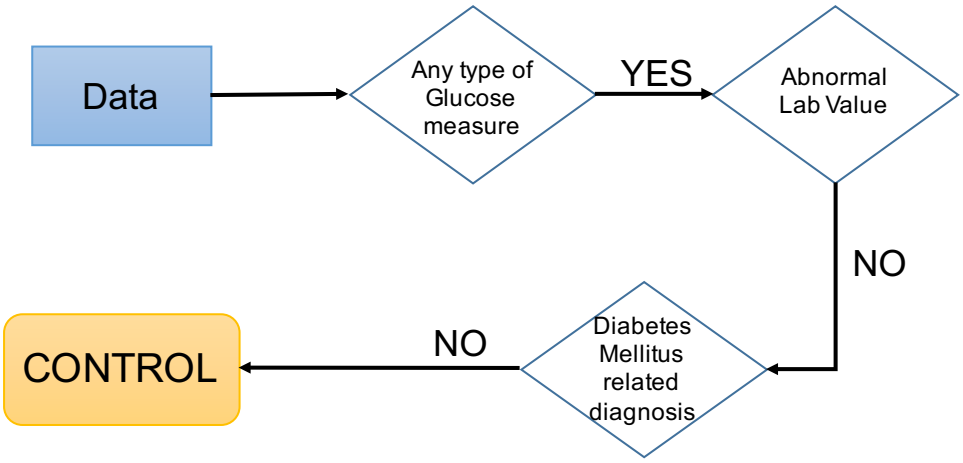

Figure 2: Determination of controls

- **Requirements for Control patients**: Figure 2 details the rules for determining whether a patient is control. Certain parts of the flowchart involve criteria that you will find the handout folder `/phekb_criteria/`.
  - `/phekb_criteria/ABNORMAL_LAB_VALUES_DX.csv`: You can refer to these abnormal lab values criteria for controls.
  - `/phekb_criteria/DM_RELATED_DX.csv`: Any of the ICD codes present in this file will be sufficient to result in `YES` for the Diabetes Mellitus related diagnosis criteria.

a. Implement *edu.gatech.cse8803.main.Main.loadRddRawData* to load above CSV files as structured RDD. Follow instructions for turning this in.[5 points]

b. Implement *edu.gatech.cse8803.phenotyping.T2dmPhenotype*. Follow instructions for turning this in.

- Finding case patients [10 points]
- Finding control patients [10 points]
- Finding other patients [5 points]

2 Programming: Unsupervised phenotyping via clustering [35 points]

At this point you have implemented a supervised, rule-based phenotyping algorithm. Those type of methods are great for picking out specific diseases, in our case diabetes and rheumatoid arthritis. However, they are not good for discovering new, complex phenotypes. Such phenotypes can be disease subtypes (i.e. severe hypertension, moderate hypertension, mild hypertension) or they can reflect combinations of diseases that patients may present with (e.g. a patient with hypertension and renal failure).

2.1 Feature Construction [14 points]

Given the raw data, you need to start with feature construction. You will need to implement ETL using Spark with similar function as what you did in last homework using Pig. Given that you know which diagnoses (in the form of ICD-9 codes) each patient exhibits, and which medication each patient took, these can be used as features in a clustering model. Using the RDDs that you created in *edu.gatech.cse8803.main.Main.loadRddRawData*, you need to construct features using COUNT aggregation for medication and diagnostics, AVERAGE aggregation for lab test values.

a. Implement feature construction in *edu.gatech.cse8803.features.FeatureConstruction*. Implement two kinds of feature construction, one construct features using all available icd codes, lab and medication, and another with only features related to the phenotype. See comments of the source code for details.

2.2 Evaluation Metric [7 points]

Purity is a metrics to measure the quality of clustering, it's defined as

$$purity(\Omega, C) = \frac{1}{N} \sum_{k} \max_{j} |w_k \cap c_j|$$

where N is the number of samples, k is index of clusters and j is index of class. w_k denotes the set of samples in k-th cluster and c_j denotes set of samples of class j. **a .** Implement *edu.gatech.cse8803.clustering.Metrics*.

In this homework you will perform some clustering using Spark. Spark contains MLLib library with implementation of the k- means clustering algorithm and the Gaussian Mixture Model algorithm.

From clustering, we can discover groups of patients with similar characteristics. Please cluster the patients based upon diagnoses, labs and medications. If there are d distinct diagnoses, l distinct medications and m medications, then there should be $d + l + m$ distinct features.

2.3 K-Means Clustering [7 points]

a. Run k-means clustering for $k = 2, 3, 4, 5$ and report purity for two kinds of feature construction. Skeleton code is provided in *edu.gatech.cse8803.main.Main.scala*. Follow instructions to complete the necessary TODO part in the function *testClustering*

b. Perform the comparison of the clustering for the $k = 3$ case with the ground truth phenotypes that you computed for the rule-based PheKB algorithms. Specifically, for each cluster, report percentage of *case*, *control* and *unknown* in Table 1 and Table 2 for two feature construction strategies. Report the numbers in the table below.

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 1: K-Means with 3 centers characteristics using all features

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 2: K-Means with 3 centers characteristics using filtered features

2.4 Clustering with Gaussian Mixture Model [7 points]

a. Run GaussianMixture for $k = 2, 3, 4, 5$ and report purity for two kinds of feature construction. Skeleton code is provided in *edu.gatech.cse8803.main.Main.scala*. Follow instructions to complete the necessary TODO part in the function *testClustering*

b. Perform the comparison of the clustering for the $k = 3$ case with the ground truth phenotypes that you computed for the rule-based PheKB algorithms. Specifically, for each cluster,

report percentage of *case*, *control* and *unknown* in Table 3 and Table 4 for two feature construction strategies.

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 3: Gaussian Mixture Model with 3 centers characteristics using all features

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 4: Gaussian Mixture Model with 3 centers characteristics using filtered features

3 Advanced phenotyping with NMF [25 points]

Given a feature matrix $\boldsymbol{V}$, the objective of NMF is to minimize the Euclidean distance between the original non-negative matrix $\boldsymbol{V}$ and its non-negative decomposition $\boldsymbol{W} \times \boldsymbol{H}$, which can be formulated as follows,

$$\underset{\boldsymbol{W} \geq 0, \boldsymbol{H} \geq 0}{\operatorname{argmin}} \quad \frac{1}{2}||\boldsymbol{W}\boldsymbol{H} - \boldsymbol{V}||_2^2 \tag{1}$$

where $\boldsymbol{V} \in \mathbb{R}_{\geq 0}^{n \times m}$, $\boldsymbol{W} \in \mathbb{R}_{\geq 0}^{n \times r}$ and $\boldsymbol{H} \in \mathbb{R}_{\geq 0}^{r \times m}$. $\boldsymbol{V}$ can be considered as a dataset comprised of m number of n-dimensional data vectors, and r is generally smaller than n. To obtain such $\boldsymbol{W}$ and $\boldsymbol{H}$ via Multiplicative Update (MU), we alternately update the values of one while fixing the values of the other. We can reformulate Eq.(1) for obtaining a column of $\boldsymbol{H}$ while fixing $\boldsymbol{W}$ as follows,

$$F(\boldsymbol{h}) = \frac{1}{2}||\boldsymbol{W}\boldsymbol{h} - \boldsymbol{v}||_2^2 \tag{2}$$

where $\boldsymbol{v} \in \mathbb{R}_{\geq 0}^{n}$ is a column of $\boldsymbol{V}$ and $\boldsymbol{h} \in \mathbb{R}_{\geq 0}^{r}$ a column of $\boldsymbol{H}$.

Use your feature matrix $\boldsymbol{V}$ from previous feature construction result, you can decompose $\boldsymbol{V}$ into $\boldsymbol{W}$ and $\boldsymbol{H}$. Here the $\boldsymbol{W}$ will be alike cluster assignment in result of Gaussian Mixture. As each row a $\boldsymbol{V}$ represent one patient's features, corresponding row in $\boldsymbol{W}$ is cluster assignment. For example, let $r = 3$ to find three phenotype(cluster), if row 1 of $\boldsymbol{W}$ is $(0.23, 0.45, 0.12)$, you can say this patient should be group to second phenotype as 0.45 is the largest element.

Multiplicative Update(MU) is one way to solve the NMF. It defines the update rule for both $\boldsymbol{W}_{ij}$ and $\boldsymbol{H}_{ij}$ as follows,

$$\boldsymbol{W}_{ij}^{t+1} = \boldsymbol{W}_{ij}^{t} \frac{(\boldsymbol{V}\boldsymbol{H}^\top)_{ij}}{(\boldsymbol{W}^t\boldsymbol{H}\boldsymbol{H}^\top)_{ij}}$$

$$\boldsymbol{H}_{ij}^{t+1} = \boldsymbol{H}_{ij}^{t} \frac{(\boldsymbol{W}^\top\boldsymbol{V})_{ij}}{(\boldsymbol{W}^\top\boldsymbol{W}\boldsymbol{H}^t)_{ij}}$$

Pseudo-code for both update rules is listed below.

```
Initialize W, H randomly;
repeat
    /* Updating W[i,:]                                          */
    Save HH^T as a global variable H_s;
    W^{t+1}[i,:] = W^t[i,:] ⊙ V[i,:]H^T ⊙ (W^t[i,:]H_s)^{-1}
    /* Updating H[:,i]                                          */
    Save W^T W as a global variable W_s;
    H^{t+1}[:,i] = H^t[:,i] ⊙ W^T V[:,i] ⊙ (W_s H^t[:,i])^{-1}
until (1/2)||V − WH||_2^2 < ε;
```

Assuming $\boldsymbol{W}$ is very long, i.e. billion patients, which must be distributed, and $\boldsymbol{H}$ is small and can fit memory. You will see in skeleton code these two types are considered as distributed RowMatrix and local dense Matrix respectively.

a. Implement *edu.gatech.cse8803.clustering.NMF*.Follow instructions from homework handout. [15 points]

b. Run NMF clustering for $k = 2, 3, 4, 5$ and report purity for two kinds of feature construction. [5 points]

c. Perform the comparison of the clustering for the $k = 3$ case with the ground truth phenotypes that you computed for the rule-based PheKB algorithms. Specifically, for each cluster, report percetage of *case*, *control* and *unknown* in Table 5 and Table 6 for two feature construction strategies. [5 points]

d. [10 points bonus] Show why we can use MU update, derive the equation behind the update rule.

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 5: NMF with 3 centers characteristics using all features

	Cluster 1	Cluster 2	Cluster 3
Percentage Case	x%	y%	z%
Percentage Control	xx%	yy%	zz%
Percentage Unknown	xxx%	yyy%	zzz%

Table 6: NMF with 3 centers characteristics using filtered features

4 Submission [5 points]

The folder structure of your submission should be as below. You can display fold structure using *tree* command. All other unrelated files will be discarded during testing. You may add additional methods, additional dependencies, but make sure existing methods signature doesn't change. It's your duty to make sure your code is compilable with provided sbt. Be aware that writeup is within code root.

```
<your gtid>-<your gt account>-hw3
|-- homework3answer.pdf
|-- build.sbt
|-- project
|   |-- build.properties
|   \-- plugins.sbt
|-- sbt
|   \-- sbt
\-- src
    |-- main
    |   |-- java
    |   |-- resources
    |   \-- scala
    |       \-- edu
    |           \-- gatech
    |               \-- cse8803
    |                   |-- clustering
    |                   |   |-- NMF.scala
    |                   |   |-- Metrics.scala
    |                   |   \-- package.scala
    |                   |-- features
    |                   |   \-- FeatureConstruction.scala
    |                   |-- ioutils
    |                   |   \-- CSVUtils.scala
    |                   |-- main
    |                   |   \-- Main.scala
    |                   |-- model
    |                   |   \-- models.scala
    |                   \-- phenotyping
    |                       \-- PheKBPhenotype.scala
    \-- test
        |-- java
        |-- resources
```

```
        \-- scala
```

Create a tar archive of the folder above with the following command and submit the tar file.

```
tar -czvf <your gtid>-<your gt account>-hw3.tar.gz \
  <your gtid>-<your gt account>-hw3
```