App: Disease Prediction
Tingyi Wanyan

1. Github Link and Paper Link:

This program is designed for doing disease prediction using a Heterogeneous Graph Embedding technique.
github link: https://github.com/Tingyiwanyan/Disease_prediction

For detailed method and background introduction, please refer to the most recent finished paper:
https://github.com/Tingyiwanyan/Disease_prediction/blob/master/AIME_2020_paper_93.pdf

Paper script also attached

2. Data set:

MIMIC-III Data sets are used in this project, there are mainly three records are used:
a. CHARTEVENTS.csv: This file stores all patient id, admission time and lab test values
b. DIAGNOSES_ICD.csv: This file stores the patient id, and its corresponding diagnosed ICD9 code.
c. D_ICD_DIAGNOSES.csv: This file stores the ICD9 codes and its corresponding text information.
d. D_ITEMS.csv: This file stores the lab test ID and its corresponding text information.

These data files are also attached in the submission.

3. Main files:

First clone the data repositories, the source code contains multiple files, the main 3 scripts related to the disease prediction methods are:
Kg_construction.py: Construction of Heterogeneous Graph Model
kg_model_modify.py: Configuration of Model, training and testing
kg_process_data.py: Process data, divide into train and test.

4. Command Line for Running Scripts:

Running main script for loading and training data:
ipython -i Kg_construction.py
(Be sure to change the file loading path, otherwise it won't be load correctly, the training uses 80% of the whole data, the other 20% data are used for testing)
If runs normal, the training will print loss at every iteration:

```
WARNING:tensorflow:From /home/tingyi/disease_prediction/
 Please use tf.compat.v1.global_variables_initializer i

WARNING:tensorflow:From /home/tingyi/disease_prediction/
Please use tf.compat.v1.local_variables_initializer ins

in get train
25.556526
25.478073
25.553282
25.546057
25.49649
25.509068
25.48351
25.435722
25.410606
25.415092
25.44819
25.45637
25.417164
25.40186
25.359634
```

After the training, the system is ready to do disease prediction. The patient node, diagnosis node and lab test node are stored as three python dictionaries:
kg.dic_patient
kg.dic_diag
kg.dic_item

For example, retrieving the patient id by typing:
kg.dic_patient.keys()

```
In [5]: kg.dic_patient.keys()
Out[5]:
[125207,
 114690,
 196611,
 147462,
 163848,
 147469,
 106510,
 188434,
 114707,
 122900,
 180245,
 116206,
 196634,
 125616,
 163874,
 155684,
 163846,
 114726,
 139304,
 180856,
```

The test patient id is stored in :
hetro_model.test_data.

For each stored patient, the connected diagnoses is stored in key: ‘neighborhood_diag’
The connected lab test is stored in key: ‘neighborhood_item’.

To do a specific patient disease recommendation for a given test patient id, type:
hetro_model.recommandation(127335)
Here the 127335 is a patient test id, then print out the recommended diagnoses:
hetro_model.diagnosis_recom

```
In [6]: hetro_model.recommandation(127335)

In [7]: hetro_model.diagnosis_recom
Out[7]:
[['5856', 'End stage renal disease'],
 ['5859', 'Chronic kidney disease, unspecified'],
 ['2449', 'Unspecified acquired hypothyroidism'],
 ['5849', 'Acute kidney failure, unspecified'],
 ['4280', 'Congestive heart failure, unspecified'],
 ['2724', 'Other and unspecified hyperlipidemia'],
 ['2859', 'Anemia, unspecified'],
 ['53081', 'Esophageal reflux'],
 ['41401', 'Coronary atherosclerosis of native coronary artery'],
 ['5990', 'Urinary tract infection, site not specified'],
 ['2762', 'Acidosis'],
 ['42731', 'Atrial fibrillation'],
```

This gives the most likely diagnoses the system think that the patient may have, the number on the left represents the ICD9 code, the text on the right is the text information associated with the ICD9 code.

For doing the quantitative evaluation, type:
hetro_model.test()
And retrieve the results such as true positive rate, and false positive rate:

```
In [13]: hetro_model.test()

In [14]: hetro_model.tp_test
Out[14]: 0.511829544256535

In [15]: hetro_model.fp_test
Out[15]: 0.14825039154128208

In [16]:
```