Final Exam

MSBD 5002 Data Mining, Spring 2020

Due: 8:00am June 1st

Exam Guidelines

- This is an open book examination.
- Exam Duration: 8:00pm May 29 to 8:00am Jun 1.
- Turn In: You must submit your solutions via **canvas** before 8:00am Jun 1. In oder to avoid network congestion and submission failure, please submit your solutions in advance. **Any late submission will lead to zero mark directly.**
- You must pack all solutions together into one zip file, named as itsc_studentID_final.zip. The example of directory structure is shown in the Fig 1.
- **Academic integrity:**
  - Your program and report should be based on your own effort. Students cannot collaborate with anyone.
  - In case you seek help from any reference source, you should state it clearly in your report. Failure to do so is considered plagiarism which will lead to appropriate disciplinary actions.
  - **Plagiarism will lead to zero mark directly.**

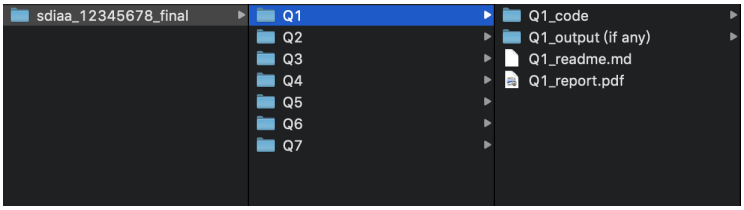

Figure 1: An example of the directory structure to be submitted.

Datasets

The data sets for Q1, Q2, Q5, Q6 and Q7 can be downloaed via the link or the canvas. The data set for Q3 can be downloaded via the link.

Grading

Your score will be given based on:

1. Correctness, completeness and clarity in your report.
2. Codes must be runnable, and code comments are necessary. Missing the necessary comments will be deducted a certain score.
3. Evaluation performance (if any).
4. You can get bonus if you propose a novel approach to any question other than Q7.

Notes

Please read the following notes carefully.

1. Please read Exam Guidelines carefully.
2. Please work hard and arrange your time reasonably.
3. For programming language, in principle, python is preferred.
4. Computation of some questions is very large, students might use cloud computing platform, such as azure.
5. If your codes or answer refer to any blog, github, paper and so on, please provide the links in corresponding **QX_report.pdf**.
6. If you have any question that Google cannot answer, please send email to {*hlicg,lwangcg,sdiaa*}@connect.ust.hk. Questions from other channels will not be answered.

1 Dimensionality Reduction on Data Feature (10 Points)

In real-world problems, some data has a high dimensional feature that contains noise and impedes the performance of machine learning models. In this question, for simplicity, we give a small toy dataset and you are required to design **two** dimensionality reduction algorithms to reduce the dimension of features.

You are required to code for your dimensionality reduction algorithms manually, but you are allowed to use any existing package of the classification model or clustering model to test the effect of your algorithms. For example, you feed the features obtained by different dimension reduction algorithms to the same classification or clustering model and compare the performance.

1.1 Data Description

The dataset contains 3686 points where each point has 400 dimension feature. Also, in the dataset, X = Multi-dimensional point data, y = labels (0 or 1).

1.2 Submission

1. Pack all code files in folder **Q1_code** .
2. You should write the pseudo code of your dimensionality reduction algorithms in the report **Q1_report.pdf** and describe them in detail.
3. You should write the experiment settings in your report **Q1_report.pdf**. For example, if you split the dataset, you should write it clearly
4. You should compare the performance of your algorithms and analyze them in the report **Q1_report.pdf**.
5. Please state clearly how to run your program in **Q1_readme.md**.

1.3 Notes

1. If your pseudo code includes the classification model or the cluster model, you can use a function to denote it directly instead of writing it in detail.
2. How many dimensions you want to reduce depends on you.

2 Imbalanced Data Classification (15 Points)

In this task, you are required to model classifiers for 5 imbalanced data sets.

2.1 Data Description

You are provided with 5 training imbalanced dataset as follows:

1. **Bi-class Datasets** *v_train.csv* and *p_train.csv* are data sets with binary classes (e.g., positive, negative).
2. **Multi-class Datasets** *y_train.csv*, *e_train.csv* and *a_train.csv* are data sets with multi-classes.

2.2 Submission

1. Pack all code files in folder **Q2_code**.
2. Predict the test data and put your prediction results in folder **Q2_output**.
3. Please report the algorithm you utilized in details, the experiment setting, and training accuracy in your report **Q2_report.pdf**.
4. Please state clearly how to run your program in **Q2_readme.md**.

2.3 Notes

1. You are allowed to use any existing package for this question. But please clearly state your reference link in **Q2_report.pdf**.

3 Fake Video Detection (15 Points)

Nowadays, people all enjoy sharing their lives via video platforms like TikTok and Kuaishou. However, due to the development of deep fake technologies, many online videos are actually synthesised intentionally. In this problem, you need to design and implement a fake video detection algorithm by yourself.

3.1 Data description

1. You can download the whole dataset from here.
2. Each sub-folder denotes one category of videos. For example, for videos in ./ApplyEyeMakeup folder, their labels are all 'ApplyEyeMakeup'. These videos are all real instead of synthesised.

3.2 Submission

1. Please write down your algorithm's principles and details in **Q3_report.pdf**. If your code refers to blogs, GitHub codes, papers or anything else, please cite them clearly.
2. Please put all your code of this question in folder **Q3_code**.
3. You need put your fake video detection results in **Q3_output.csv**. You can choose arbitrary fake and real videos online to do the validation. The output format should be like

The_Name_of_Video	**Is_This_Video_real?**
Diving_Bird.avi	False
Obama_Presenting.avi	True

4. Please state clearly how to run your program in **Q3_readme.md**.

3.3 Notes

1. You can use any algorithm that you know. You can NOT directly use complete models that others have already trained to do classification without any detailed process.
2. We will test your algorithm on separated dataset and use this metric as the grading criteria.
3. The term fake video means that the video is not originated from the nature world, instead it is generated by some algorithms from noise or some distributions.

4 Stock Prediction (15 Points)

Everybody wants to earn money from the stock market. Is it possible to predict stock prices in the future with data mining and machine learning technologies? In this problem, you need to design and implement an stock price prediction algorithm to predict the closing prices of AAPL (Apple company), PG (P&G) and GM (General Motors) on June 2, 2020.

4.1 Data Description

1. You are free to collect arbitrary data and information from the internet like stock prices, twitters, newspapers etc.

2. Your data collecting method is also not restricted.

4.2 Submission

1. Please write down your algorithm's principles and details in **Q4_report.pdf**. If your code refers to blogs, GitHub codes, papers or anything else, please cite them clearly.

2. Please put all your codes of this question in **Q4_code** folder.

3. You need give you prediction for the closing prices of AAPL (Apple company), PG (P&G) and GM (General Motors) on June 2 in **Q4_output.txt**. The .txt file should only contains one float number representing the stock price like 318.25.

4. Please state clearly how to run your program in **Q4_readme.md**.

4.3 Notes

1. You can use any algorithm that you know. You can NOT directly use complete models that others have already trained to do sentiment analysis without any detailed process.

5 Frequent Itemset Mining (15 Points)

You are required to write programs to mine the **frequent itemsets** (min sup = 100) in the provided data. Based on the frequent itemsets you mined, please write program to mine the **closed frequent itemsets** and **maximal frequent itemsets**.

5.1 Data Description

The benchmark data set is supported by the IBM Almaden Quest research group, which contains 1,000 items and 100,000 transactions. For simplicity, each number uniquely identifies an item.

5.2 Task Description

1. Task 1 (5 points) Mine the frequent itemsets (min sup = 100).
2. Task 2 (5 points) Mine the closed frequent itemsets.
3. Task 3 (5 points) Mine the maximal frequent itemsets.

5.3 Submission

1. Pack all code files in folder **Q5_code**.
2. Following the sample submission, put your outputs in folder **Q5_output**.
3. Describe the algorithm you utilized for task 1 and the running time of each task in **Q5_report.pdf**.
4. Please state clearly how to run your program in **Q5_readme.md**.

5.4 Notes

1. You are allowed to use any existing package for this question. But please clearly state your reference link in **Q5_report.pdf**.

6 Community Detection (10 Points)

In many social networks, many different relations usually exist between a same set of users. In this task, given several datasets, you are required to detect the communities among 419 tweet users. The ground truth consists of five communities.

6.1 Data Description

419 users are referenced by their unique Twitter IDs in data *ids.mtx*. You are provided with three different sources of these users social networks:

1. **follow/followedby** This folder records "follow" information between users.
2. **mention/mentionedby** This folder records "mention" information between users.
3. **retweet/retweetedby** This folder records "retweete" information between users.

6.2 Task Description

1. Task 1 (10 points) Detect 5 communities between 419 users.

6.3 Submission

1. Pack all code files in folder **Q6_code**.
2. Following the sample submission, put your outputs in folder **Q6_output**.
3. Describe in detail the algorithm you are using in **Q6_report.pdf**.
4. Please state clearly how to run your program in **Q6_readme.md**.

6.4 Notes

1. You are allowed to use any existing package for this question. But please clearly state your reference link in **Q6_report.pdf**.
2. Do NOT use any external data.

7 Matrix Factorization in Graph Representation and Recommendation System (20 Points)

7.1 Preliminary

Matrix Factorization (MF) (e.g., Probabilistic Matrix Factorization and Non-Negative Matrix Factorization) techniques have become the crux of many real-world scenarios, including graph representation and recommendation system (RecSys), because they are powerful models to find the hidden properties behind the data. More specifically, Non-Negative Matrix Factorization (NNMF) is a group of models in multivariate analysis and linear algebra where a matrix $\mathbf{A} \in \mathbb{R}^{|B|\times|C|}$ is decomposed into $\mathbf{B} \in \mathbb{R}^{|B|\times d}$ and $\mathbf{C} \in \mathbb{R}^{|C|\times d}$ according to:

$$\mathbf{B}, \mathbf{C} = \arg\min_{\mathbf{B}', \mathbf{C}'} \left\| \mathbf{A} - \mathbf{B}'\mathbf{C}'^{\top} \right\|_F^2, \tag{1}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

Graph Representation Given a graph $\mathcal{G} = (E, V)$ (V is the set of nodes and E is the set of edges), the adjacency matrix of $\mathcal{G}$ is represented by $\mathbf{A} \in \mathbb{R}^{|V|\times|V|}$, where $x_{ij} = 1$ if there is an edge $e \in E$ between the node v_i with the node v_j, otherwise $x_{ij} = 0$. Following Eq. 1, you can represent nodes V into $\mathbf{V} \in \mathbb{R}^{|V|\times d}$ by minimizing the loss $\mathcal{L}_g = \left\| \mathbf{A} - \mathbf{V}\mathbf{V}^{\top} \right\|_F^2$. You may refer to the last lecture slides for more details.

User-Movie RecSys Given a collection of users U and a collection of movies V, let $\mathbf{X} \in \mathbb{R}^{|U|\times|V|}$ denotes the user history of rating items, where $x_{ij} = 0$ if i-th user u_i has not rated the j-th movie v_j, otherwise $x_{ij} \in (0, 5]$ represents the ratings of u_i to the movie v_j. Following Eq. 1, you can represent users U and movies V into $\mathbf{U} \in \mathbb{R}^{|U|\times d}$ and $\mathbf{V} \in \mathbb{R}^{|V|\times d}$ respectively, by minimizing the loss $\mathcal{L}_r = \left\| \mathbf{X} - \mathbf{U}\mathbf{V}^{\top} \right\|_F^2$.

User-Movie RecSys with Movie Tags In real-world applications, it is important to handle the cold-start issue in RecSys since there are new movies coming out every day. The above user-movie RecSys cannot well handle this issue since no user have seen a new movie before. This is a variant of MF techniques to alleviate the cold-start issue by incorporating the information of movie tags. Given a collection of movie tags T and a collection of movies V, let $\mathbf{Y} \in \mathbb{R}^{|T|\times|V|}$ denote movie tag information, where $y_{ij} = 1$ if movie v_j is tagged with t_i, otherwise $x_{ij} = 0$. Therefore, you can represent users U, movies V, and tags T into $\mathbf{U} \in \mathbb{R}^{|U|\times d}$, $\mathbf{V} \in \mathbb{R}^{|V|\times d}$, and $\mathbf{T} \in \mathbb{R}^{|T|\times d}$ respectively, by minimizing the loss $\mathcal{L}_t = \left\| \mathbf{X} - \mathbf{U}\mathbf{V}^{\top} \right\|_F^2 + \left\| \mathbf{Y} - \mathbf{T}\mathbf{V}^{\top} \right\|_F^2$ as:

$$\mathbf{U}, \mathbf{V}, \mathbf{T} = \arg\min_{\mathbf{U}', \mathbf{V}', \mathbf{T}'} \left(\left\| \mathbf{X} - \mathbf{U}'\mathbf{V}'^{\top} \right\|_F^2 + \left\| \mathbf{Y} - \mathbf{T}'\mathbf{V}'^{\top} \right\|_F^2 \right).$$

7.2 Task Description

You are required to code three different matrix factorization techiniques as introduced above.

1. Task 1 (5 points) Given the graph data *graph.npy*, please represent nodes V by minimizing $\mathcal{L}_g$.
2. Task 2 (5 points) Given the rating data *rating.csv*, please represent users U and movies V by minimizing $\mathcal{L}_r$.
3. Task 3 (10 points) Given the rating data *rating.csv* and the movie data *movie.csv*, please represent users U, movies V, and tags T by minimizing $\mathcal{L}_t$.

7.3 Submission

1. Pack all code files in folder **Q7_code**.
2. Please report the experiment settings (e.g., learning rate), your final losses ($\mathcal{L}_g$, $\mathcal{L}_r$, and $\mathcal{L}_t$), and the corrsponding learning curves (X-axis: iteration and Y-axis: loss) for three tasks in **Q7_report.pdf**.
3. Please state clearly how to run your program in **Q7_readme.md**.

7.4 Notes

1. You are allowed to use any existing package for this question. But please clearly state your reference link in **Q7_report.pdf**.
2. Hints: Learning in Eq. 1 is usually more stable if $\mathbf{U}$ or $\mathbf{V}$ is updated in each subproblem based on gradient descent by reducing $\mathcal{L}$ slightly, such as:
    - while not converged:
        - Minimizing $\mathcal{L}$ over $\mathbf{U}$ by gradient descent with $\mathbf{V}$ fixed.
        - Minimizing $\mathcal{L}$ over $\mathbf{V}$ by gradient descent with $\mathbf{U}$ fixed.