Projects Description

Project 1 & 2 are application-oriented. Project 3 & 4 are algorithm-oriented. You need to select **one** of these projects as your assignment.

Project 1: Visualization of data distribution (updated on MySQL usage and UI)

- Dataset: Bank Marketing Data Set
- Your task: to build an application to show the distribution of attributes (dimensions) in the dataset.

Requirements

- Supportance of slice and dice operation
    - If I choose one or several values of specific attribute(s), the application should be able to show the distribution filtered by these values.
    - For example, If I choose "age = 18-22 and job = student", you need to show the distributions of some other attributes (like education and balance) conditional upon "age = 18-22 and job = student".
- Supportance of MySQL
    - Your application should be able to import the .csv data to MySQL server (without help of management software such as PHPMyAdmin) and load data from it.
    - You can assume that there is an empty database named "bank" with username "test" and empty password in the local MySQL database. This account have all privilege to manipulate the "bank" database.
- Showing 1~9 distribution of attributes at the same time.
    - Users are able to select which attribute(s) to show.
    - However, you need to import **all the data** to MySQL server, not 9 columns of them!
- Interactive UI
    - Users without much programming knowledge can operate your application by select, click and other simple actions.
    - Web or desktop based UI is OK. Command-line interface is not acceptable.

Suggestions

- You may use Python (with matplotlib, mysqldb and so on) or PHP to build the application.

- If you choose PHP, D3.js, a really powerful tool for interactive data visualization, is recommmended as the front-end library.

Project 2: Visualization of data correlation

The dataset and requirements are the same with project 1, but the task is to show the correlation of two selected attributes.

Project 3: "Phrase" mining based on frequent patterns

- Dataset:
    - Gutenberg dataset
    - Amazon review data (optional)
    - DBLP (title) (optional)
    - ACL (title & abstract) (optional)
- Your task: design and implement a data mining algorithm to find the "phrase" in these texts based on frequent patterns.
- The "phrase" here can be more general. Here, two words can be called "phrase" when they always appear together in one sentence, even if they are not adjacent (maybe this will be easier for you). However, it has more actual meaning when we only consider words that are adjacent. That needs some additional techniques and it is put in the "bonus point" part.

Suggestions

The dataset is quite large. You may select **part of them** that you are interested in. For example, all books written by Charles Dickens, all fairy tale books or all books that is written in 18th century. For amazon review data, you can just select the type of reviews that you are interested in.

- **(bonus point)** You will get more score by applying your algorithm to other dataset (such as Amazon review data, DBLP titles and so on).
- **(bonus point)** You can see sentences as "baskets" as the slide says. However, I suggest you to try some different ways, such as:
    - See paragraphs as baskets
    - See chapters as baskets
    - See books as baskets (this might be harder)
    - (For amazon review data) See reviews as baskets
- You may apply some "stop word" methods, which is common in NLP related tasks.
- You may use Expat to read data from xml file (such as DBLP). It is a quite useful skill.

- You may use Python for data preprocessing. However, if you want to make you program more efficient, you may use c++ to implement the core algorithm.

Project 4: Implementation of a "influential" data mining algorithm

- Your task: just as the title says.
- Prepare the dataset by yourself.
- Don't implement your algorighm by a single line with the help of library function, such as:
    - res = xxxlib.Apriori(data)
    - rank = xxxlib.PageRank(data)

Candidate choices (Normal)

- FP algorithm
- PageRank
    - at least 10000 nodes and 100000 edges.

Candidate choices (Hard, with at least 5 bonus point)

- RankClus
    - Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, T. Wu, “RankClus: Integrating Clustering with Ranking for Heterogeneous Information Network Analysis”, EDBT’09
- PathSim
    - Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, “PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks”, VLDB’11