Project Instruction
"Data Science Research Bot" (a.k.a. SciBot)
Data Science CSE 40647/60647

Last updated: Sept. 28, 2017

Professor-in-charge:

Dr. Meng Jiang, mjiang2@nd.edu
Office: 326C Cushing Hall Phone: (574) 631-7454

Teaching Assistant (TA):

Qi Li, qli8@nd.edu
Office: 212 Cushing Hall

Project goal:

~~Individual project, NOT group project.~~
Team project (after voting in class, we some draw conclusions):

- A team will have at most 2 members (one or two).
- Students will give their partner's name (or N/A – if they do individual projects by their own) in HW3.
- Members in the same team will have the same score.

On a large real-world dataset, students should be able to:

- Process raw data: data cleaning, data integration, data reduction, dimension reduction
- Describe data warehouse, OLAP, data cube concepts and technology that work on multi-dimensional data
- Use Apriori and FP-Growth for frequent pattern mining
- Describe diverse patterns, sequential patterns, graph patterns
- Use Decision Tree, Naïve Bayes, Ensembles for classification
- Describe SVMs and Neural Networks for classification
- Use K-Partitioning Methods (K-Means, etc.) for clustering
- Describe Kernel-based Clustering and Density-based Clustering
- Use appropriate measures to evaluate results of different functionalities

Students are required to accomplish tasks that will be described as "required tasks" below. Students are encouraged to do more tasks as either the recommended ones or the ones they like to do. Basically, the ultimate goal is to enrich the functionalities of the "SciBot" using data science and technology.
One example of the functionalities could be:
>> *What problem do you want to find methods that are strongly associated with?*
>> *(by user) document classification*
>> *The methods that are associated with the problem "document classification" are:*
support_vector_machines (relative support: 0.37, confidence: 0.25)
decision_tree (relative support: 0.32, confidence: 0.21)

…

Students are also required to write a project report/ paper to describe their achievement including the following points *for each task*: (1) Motivation and task definition, (2) Approach, (3) Results, and (4) Discussions.

Grading policy: (25% of the final score)

Students are required to submit their code package + "readme" (.ZIP) and project report/paper (.PDF). There is no paper template requirement.

Students will volunteer to present their SciBot (tech and results) in two lectures. Classmates and the instructor will grade them based on the presentation. For the students who do not present, the instructor will grade their projects after all the lectures end. Note that we will have comparative grading – finishing all the required tasks cannot make sure that you have all the points.

Students are encouraged to implement algorithms such as Apriori, FP-Growth, and K-Means Clustering by themselves instead of calling Python packages.
Students are also encouraged to use Python packages (e.g., numpy and scipy) when they use advanced techniques (e.g., SVMs, Neural Networks, word2vec) to address challenging tasks.
Students are encouraged to compare different methods on the same task and discuss their advantages and disadvantages. Reasoning is always welcome in the paper.
Students are encouraged to share any annotation data (e.g., labels, hand-crafted rules) but not any segment of codes.
Students are encouraged to make a GUI for the SciBot. They are also encouraged to give a better name to their bots than "SciBot".

Graders should have higher expectations on graduates than undergraduates – not only on the project results (more tasks, better performances) but also on writing (a workshop-quality paper of strong reasoning). Undergraduates will be applied with a uniform grading policy no matter what majors they have.
Graders tend to grade individual projects/undergraduates better than team projects/graduates if they generate the same results.

The project due is Nov 30, 2017. There will be NO extension. Significant updates are welcome before the final exam – students can send the updates to the instructor after the due by e-mail but they have to submit one version before the due.

Academic Dishonesty:

- The CSE and du lac honor code will be strictly followed.
- All assignments are individual unless instructed. You can discuss the assignment at a high level, but you should independently and individually write down the answers and/or the program. The sharing and copying of homework solutions or programs or functions or exams will be considered cheating.
- All the references and sources should be carefully provided and cited.

- Entering Notre Dame you were required to study the on-line edition of the Academic Code of Honor, to pass a quiz on it, and to sign a pledge to abide by it. The full Code and a Student Guide to the Academic code of Honor are available at: http://honorcode.nd.edu.
- Perhaps the most fundamental sentence is the beginning of section IV-B: "The pledge to uphold the Academic Code of Honor includes an understanding that a student's submitted work, graded or ungraded – examinations, draft copies, papers, homework assignments, extra credit work, etc. - must be his or her own."

Dataset introduction:

The dataset has both structured and unstructured information of over five thousand data science research papers. It includes three zip files:

1. pdf.zip (4.7GB; unzip ~5.5 GB): raw unstructured data (actually you don't have to use this huge file)
   https://www.dropbox.com/s/460h772tpuceew5/pdf.zip?dl=0
   It has 64 folders/proceedings. Each folder is named as "[CONF][YEAR]":
   - CONF: {icdm, kdd, wsdm, www}
     icdm: IEEE International Conference on Data Mining
     kdd: ACM SIGKDD Conference on Knowledge Discovery and Data Mining
     wsdm: ACM Conference on Web Search and Data Mining
     www: International Conference on World Wide Web
   - YEAR: {94, 95, …, 99, 00, 01, …, 16}
     from 1994 to 2016

   Each folder has an incomplete set of papers of the proceeding of CONF-YEAR. The papers are named as "[PDFID].pdf":
   - PDFID: {icdm01-d0, …}

2. text.zip (~95MB; unzip ~270MB): raw unstructured data
   https://www.dropbox.com/s/o0f7qjb5mobmfvt/text.zip?dl=0
   It has the same folder names and file names as pdf.zip. The only difference is the files' ext. name (".txt" here, ".pdf" in pdf.zip). A Python package was used to transfer *.pdf into *.txt, but the text looks incomplete and noisy.
   - Practitioners are recommended to skip the REFERENCE section when they mine knowledge from the text data.

3. microsoft.zip (~24MB; unzip ~100MB): raw structured data
   https://www.dropbox.com/s/o9qzhbdd0pmk5wm/microsoft.zip?dl=0
   It has five files. All except "index.txt" were provided by Microsoft Academic Search (MAS) engine. "index.txt" was created by the instructor to bridge the structured and unstructured data with entry id (PDFID and PID).
   (1) index.txt

Folder name in pdf.zip/txt.zip	PDFID (file name) in pdf.zip (*.pdf) /txt.zip (*.txt): (paper id in PDFs)	PID (paper id in MAS database)	TITLE (lower case)

(2) Papers.txt

PID (paper id in MAS database)
TITLE_CASE (case sensitive)
TITLE (lower case)
YEAR (year of proceeding)
DATE_OF_PROCEEDING (not recommended to use)
DOI (not recommended to use)
CONF_FULL_NAME (not recommended to use)
CONF (abbreviation, lower case)*
N/A
CID (conference id; not proceeding id!)**
N/A

* The dataset is noisy. NOT every entry can be correlated across the files, for example, CONF has conference names that are not included in {icdm, kdd, wsdm, www}.
** One-to-one mapping between CONF and CID.

(3) PaperKeywords.txt

PID (paper id in MAS database)
KEYWORD (keyword in lower case)*
KID (keyword id; not recommended to use)

** It has a very limited set of keywords. We will process the text data for more structured semantic information of the papers.*

(4) PaperAuthorAffiliation.txt

PID (paper id in MAS database)
AID (author id in MAS database)
FID (affiliation id in MAS database)
AFF_ORG (original affiliation name, not recommended to use)
AFF (normalized affiliation name)*
SID (author sequence number: “1” = the first author, “3” = the 3rd author)**

* One-to-one mapping between AFF and FID.
** The author information of a paper may not be complete. It may only have the 1st, 2nd, and 4th authors.

(5) Authors.txt

AID (authored in MAS database)	AUT (author name in lower case)

Required tasks:

<u>Task 1: Data preprocessing</u>

Q1: Given the above files from multiple sources, can we integrate the data, clean the data (work with incomplete/missing entries and redundancy/unnecessary entries), describe the data using statistics and visualization (distributions, etc.)? What are the data objects and what are the attributes?

Techs: Data cleaning, data integration, data description, statistical analysis, data visualization.
Hints:

1. PDFID-PID mapping in index.txt can be used to integrate paper text (txt.zip), Papers.txt, PaperKeywords.txt and PaperAuthorAffiliation.txt.
2. AID can be used to integrate PaperAuthorAffiliation.txt and Authors.txt.

PDFID	(PID)	CONF	YEAR	LIST_AUTHOR	LIST_KEYWORD	…	TEXT	(PDF)

Task 2: Entity mining: Candidate generation and quality assessment
Q2: Given the text data, can we mine entities, e.g., "text categorization", "document classification", "naïve bayes", "decision tree", "support vector machines", "SVM", "SVMs"? Can we propose at least one measure of entity quality and rank them by it?
Techs: Frequent pattern mining, measures (outlier-ness like Z-score), hand-crafted rule matching
Hints:

1. Entity names are a subset of words or phrases. Relational phrases or stop phrases are not entity names, e.g., "turn_out_to_be", "in_this_paper".
2. Use the given keyword list as entity/phrase candidates. Suppose a document has 1,000,000 words. It has 1,000 "decision" and 1,000 "tree". Assuming an even distribution of the words, we may only have one "decision tree". The observed number could be much bigger.
3. Rules of lexical features are useful for generating entity candidates. For example, we may often see "… <u>S</u>upport <u>V</u>ector <u>M</u>achines …", "<u>s</u>upport <u>v</u>ector <u>m</u>achines (<u>SVMs</u>)", "… <u>n</u>on-negative <u>m</u>atrix <u>f</u>actorization (<u>NMF</u>) …" Then we use outlier-ness to evaluate the quality of the candidates.
4. We may generate N-grams (N=2,3,4…) as phrase/entity candidates. However, the number could be huge. Can we use heuristics (like the rules above) to generate a proper-sized set of candidates?
5. If we consider phrase/entity candidates as patterns (word itemsets), another possible quality measure is *absolute support* of the pattern (i.e., count of the pattern in the text data) if we consider each sentence/paragraph/paper as a transaction and words as items.
6. Label a set of quality entities. Evaluate the performance of different measures and different candidate sources/generation methods.

PDFID	LIST_ENTITY

Milestone 1: In HW3, we request students to report some numbers of their results on Task 1 and 2. (Oct. 3 due)

Task 3: Entity typing
Q3: Given quality entities and text data, can we assign types to the entities? Basically, we consider four major types: $Problem, $Method, $Metric, $Dataset.

$Problem	text categorization, document classification, fraud detection …
$Method	naïve bayes, decision tree, support vector machines …
$Metric	accuracy, precision, recall, F1 score …
$Dataset	netflix, youtube, movielens, facebook, twitter, dblp …

Techs: Measures (outlier-ness like Z-score), dimension reduction, classification

Hints:

1. Take an entity as a data object. The attributes are contextual words around the entity in the text data. Suppose we have an N-size window and take each word in the window as an attribute. Then we can measure the probability of assigning a type to an entity. We assume that if the word "method", "model" or "approach" has a high Z-score to appear in the context of an entity, the entity is likely to be typed as "$Method".
2. Here we carefully type entities. We want high accuracy but not good coverage. We use supervised methods (for classification – Naïve Bayes or decision tree or others) and feed with the set of contextual words (attributes) to type other entities. If the number of attributes is too large, we can consider to use dimension reduction (like PCA or SVD).

PDFID	LIST_PROBLEM	LIST_METHOD	LIST_METRIC	LIST_DATASET

<u>Task 4: Collaboration discovery</u>

Q4: Given the paper-author data, find frequent author-sets (as patterns): which two/three/four authors often collaborate together?

Techs: Frequent pattern mining (Apriori, FP-Growth).

Hints:

1. Here each paper is considered as a transaction. Each author is an item.

<u>Task 5: Problem-method association mining</u>

Q5: Given the paper-problem-method data, find strong association rules, problem X → method Y, or method X → problem Y, of high support and confidence.

Techs: Association rule mining.

Hints:

1. Here each paper is considered as a transaction. Each problem/method is an item.

Milestone 2: In HW4, we request students to report some numbers of their results on Task 3, 4, and 5. (Nov. 9 due)

<u>Task 6: Problem/method/author-to-conference classification</u>

Q6: Given a problem/method/author, predict if a conference has papers of it.

Techs: Binary classification (Naïve Bayes, Decision Tree).

Hints:

1. What are the attributes (features) you want to use?
2. How to set up training and testing? Please evaluate the performance on different features, different models, and different setups.

<u>Task 7: Paper clustering</u>

Q7: Given a set of papers, cluster them into K groups.

Techs: K-partitioning clustering methods (K-Means).

Hints:

1. What are the attributes (features) you want to use?
2. Suppose K = 4 and the ground-truth is the conference. Please evaluate the performance on different features and different methods.

Final due: Nov. 30. Volunteer to present!

Recommended tasks:

Task 1+: Data preprocessing
Q1+: Study the data distributions: Do you find power-law, Poisson, or normal distributions between variables? Can you explain them?
Techs: Statistical analysis.

Task 2+: Entity mining: Candidate generation and quality assessment
Q2+: Can you use auxiliary sources (e.g., stop word list) or auxiliary criteria to further improve the quality of entity names you mined?
Techs: Classification (good entity name: "yes", "no").

Task 3+: Entity typing
Q3+: Can you use cluster analysis on the entities to type entities by clusters? Given what kind of features, the entities might be grouped together if they had the same type?
Techs: Clustering

Task 4+: Advisor-advisee discovery
Q4+: Can you find advisor-advisee relations from collaborations?
Techs: Measures (like Kulc).

Task 8: Manage the data with data cube
Q8: Given enriched structured data, can we construct a data cube and compute iceberg cubes for query-based applications? E.g., expert recommendation: Given a problem, list authors, papers and other information that help related research.
Techs: Data cube, iceberg cube, closed cells, etc.
Hints:

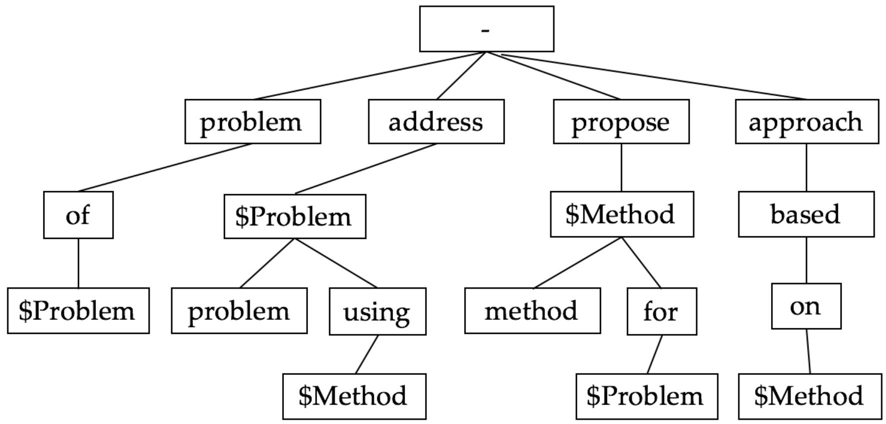
1. Each paper is considered as a transaction. The cell maintains a set of papers. A paper may be in multiple cells. We count the size of paper set for the cube computation.
2. For a list of entities, the attribute types are dimensions (e.g., problem, method, dataset, author, conference); the attribute values are the dimension values (e.g., "naïve bayes", "decision tree").
3. More functionalities of the data cube, and efficiency analysis are welcome.

Task 9: Pattern-based entity recognition and typing
Q9-1: Given entity names, can we find frequent patterns around the entities? We replace concrete entity names as "$Entity". You can find more entities via pattern matching.
Q9-2: Given seed typed entities (methods, problems, etc.), can we find concrete frequent patterns around the typed entities? We replace concrete method/problem entities as "$Method"/"$Problem". Those patterns indicate that you may be able to find more entities of the specific types.
Techs: Constraint-based frequent pattern mining.
Hints:

1. An iterative process that first generate and evaluate the support of patterns such as "*problem of $Problem*", "*address $Problem problem*", "*propose $Method method*", "*approach based on $Method*", "*propose $Method for $Problem*", "*address $Problem*

using $Method" and then recognize more entities and their types by matching the patterns in the text and repeat until convergence.

2. How to generate the patterns *efficiently*? How to match the text with patterns *efficiently*? The data structure of Trie Tree (https://en.wikipedia.org/wiki/Trie) is recommended. Suppose we have the above 6 patterns. We can construct a tree below. It is easier to search in the tree than to match string patterns.



PDFID	LIST_PROBLEM	LIST_METHOD	LIST_METRIC	LIST_DATASET

Task 10: Problem/method/author clustering

Q10: Given a set of problems/methods/authors, cluster them into K groups. Evaluate the clustering results in a proper way.

Techs: K-partitioning clustering methods (K-Means).

Task 11: Attribute discovery

Q11: Suppose we use rules to type digit number as $Digit. Can we find the size of datasets used in the papers? Can we find the performance of methods?

Techs: Constraint-based frequent pattern mining.

Task 12: Ensemble learning

Q12: Suppose we have multiple models/methods for a specific task (actually you do have if you've finished the required tasks). Can we use ensemble methods to further improve the performance?

Techs: Ensemble methods (bagging, Adaboost, etc.).

Task 13: Practice with advanced classification and clustering methods

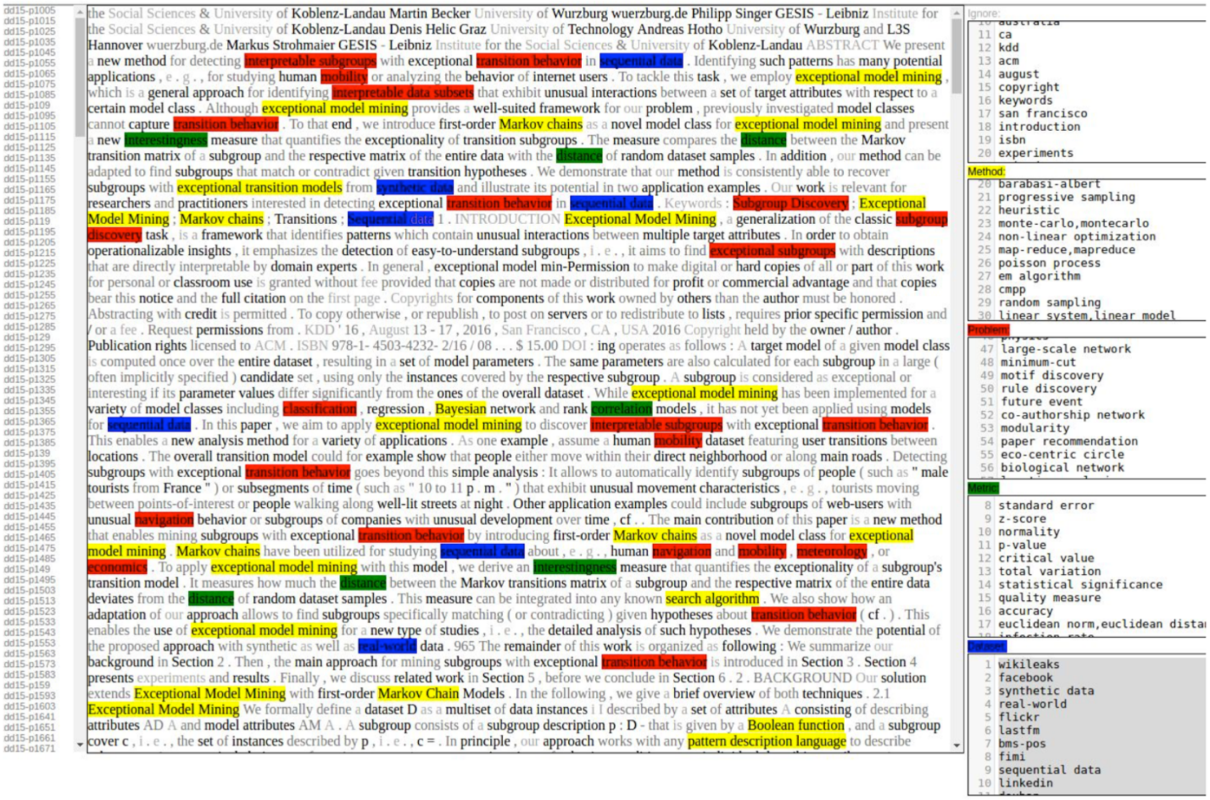
Q13: Can you solve the above tasks with advanced classification models (e.g., SVMs, Neural Networks) and clustering methods (e.g., spectral clustering)?

Task 14: Other interesting tasks related to other data entries/attribute like "affiliation ranking on a specific method/problem".

Task 15: Data visualization is encouraged.

Examples: Project results from UIUC Summer 2017 Data mining course (10 weeks, 3 lectures per week, 5 written assignments).

(1) A Web UI to manually label "problems", "methods", "metrics", etc.



(2) Evaluating clustering analysis (K = 3) based on two PCA features.

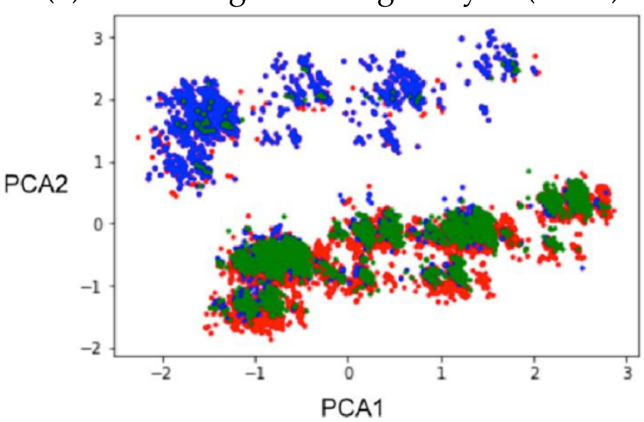


Fig.6 True label

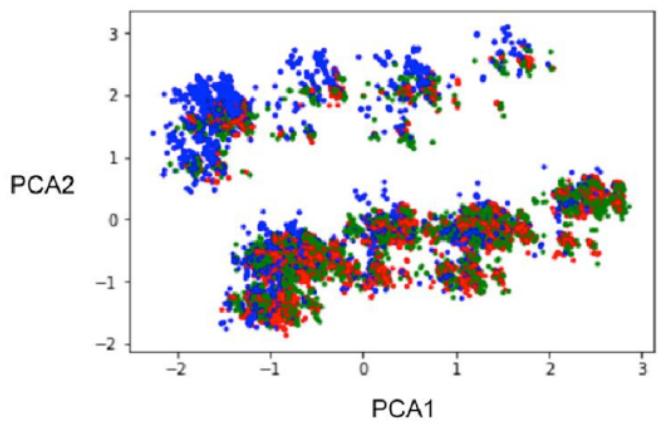
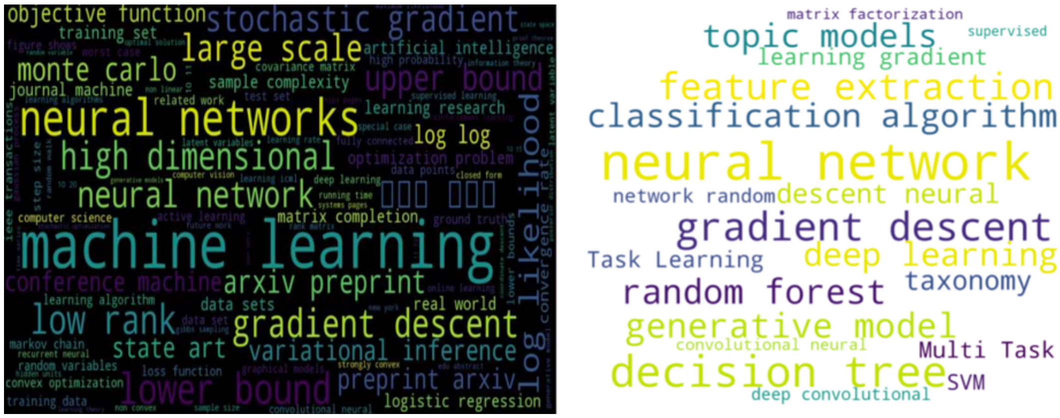


Fig.7 Clustering labels

(3) Word cloud of entity clusters



(4) Classification performance (F1 score) vs training size (%) and #PCA features.