CSC 869: Data Mining, Spring 2015, SFSU Instructor: Dr. Hui Yang

Term Project Report on:

**INSTRUCTOR ASSESSMENT SYSTEM**

By: Pratik Jaiswal

Problem being addressed: To analyze the opinion of students about their instructors in an educational institution. In this project, the students using data mining will evaluate instructors and further prediction of whether instructors will be invited to faculty classes or not can be done.

This is an important aspect since students are expected to get best instructors in their education system.

Dataset:

Initially, I was using instructor data with attributes like instructor degree, instructor experience, assessment score, degree type and the acceptation. Decision tree classifier was providing certain observations out of this like:

**If ((Assessment score == EXCELLENT) && (Experience == FALSE)) then, ACCEPT the instructor**. This is because even the experience is not that great (NEW instructor); assessment score is excellent that assures education manager to promote the instructor in future semesters.

Later on, I found that this dataset is not sufficient in order to create a robust system since the above-mentioned attributes like instructor degree do not give more insights on how they affect the accept/reject decisions from education managers.

Part I: Instructions on compiling and running the program(s)

The source code and the results can be found in:

“**Lack of Proper Data Mistake - Files(corrected)”** folder with ***professor.arff*** as a dataset file and it can be directly imported into the WEKA.

The decision tree classifier output is there in the same folder: ***decisiontree.txt*** and the visualized tree is in image: ***DecisionTreeVisualization.png***

Classifier accuracy: **92.85%** (for smaller dataset)

Confusion Matrix:

a b <-- classified as

9 0 | a = yes

1. 4 | b = no

Result: Mistake was a lack of proper data. So, in order *to correct this* mistake, I did more research and landed up on **RateMyProfessors(RPM)** data with more informative attributes like easiness, helpfulness, clarity, overall ratings(value falls between 0 and 5) and total ratings.

Fetching the data from **RPM** was a complicated task since it requires a scrapper to be written and fetch everything from the URL using python. In order to fetch the entire data, run the program ***fetchdata.py***, which is there in **“Dataset Obtain”** folder. This program uses scraperwiki library that gives the data in wiki like fashion and BeatifulSoup library for navigating and copying the data from specified URL (in my case, ratemyprofessor website).

To run fetchdata.py, you need to install libraries scraperwiki and BeautifulSoup using pip and brew like:

(pip should already be installed: [*https://pip.pypa.io/en/latest/installing.html*](https://pip.pypa.io/en/latest/installing.html) & for brew:

*ruby -e "$(curl -fsSL* [*https://raw.github.com/Homebrew/homebrew/go/install*](https://raw.github.com/Homebrew/homebrew/go/install)*)”* – run this ruby command in terminal)

*brew install poppler*

*pip install scraperwiki*

*Linux: sudo apt-get install python3-bs4, Mac: pip install beautifulsoup*

Depending on the platforms, dependability issues can be resolved and above commands should work.

Running instruction: ***python fetchdata.py***

Above program will take a huge amount of time, since it fetches the information of instructors from A-Z series. Therefore, to show an explicit example of how it works, I have written a *small module* that fetches single professor information.

File to fetch a single instructor is ***fetchprofessor.py*** and it is in the same folder **“Dataset Obtain”**

Running instruction: ***python fetchprofessor.py***

It also gives a data in SQlite form, which can be used in SQL program to fire queries.

tr -s '[:blank:]' ',' <yesitis.txt > modified1.txt

Part I: instructions on compiling and running your program(s).

1. Programming language
2. Development environment

How to run the program ? be concise

GUI nor required but if, + points

Part II: description of the main strategies you've realized for this project.

3 phases:

1. Generation
2. Classifier construction
3. Evaluation

Specify for each step, what software did you use/ implemented your own version

Own version: steps to validate the correctness of the program

Part III: evaluation results and discussions

1. Describe/discuss evaluation results
2. What are the effects of data size, min support, min conf
3. Compare classifier with Naïve Bayes Implemented in homework

Part IV: Conclusions

Part V: Comments and suggestions

Future class improvements - inclusion of R

Comments/thoughts/suggestions on projects/class