CSC 869: Data Mining, Spring 2015, SFSU   Instructor: Dr. Hui Yang
Date: 05/25/2015

Term Project Report on:

INSTRUCTOR ASSESSMENT SYSTEM

By: Pratik Jaiswal

(My Research paper can be found at Github)

Problem being addressed:

To analyze the opinion of students about their instructors in an educational institution. In this project, 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 if 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)

Programming Language: Python

Development Environment: Mac OS X

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* & for brew:

ruby -e "$(curl -fsSL 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 single instructor is named ***fetchprofessor.py*** and it is in the same folder **"Dataset Obtain".**

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

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

Furthermore, after obtaining the data, in order to do analysis, I ***have converted the csv data: instructor.csv into arff format: instructor.arff for WEKA.*** A lot of preprocessing came in with handling of missing values, removing the blank entities, etc.

For row removal with values like "Unknown" & "Other", a program called ***rowremoval.py*** is written in python. File can be located in the folder named **"Row Removal Preprocess"** and it makes use of *instructor.data* as an input that is present in the same folder.

Running instruction: ***python rowremoval.py***

Another preprocessing step was to remove the unwanted blank spaces in the file. This can be achieved using the below terminal command:

```
tr -s '[:blank:]' ',' <instructor.data > trim.data
```

GUI applications:

In order to validate the results, I have written applications for a computer system Windows/Mac/Linux and Android OS (mobile devices).

The video of an Android application run is at: https://youtu.be/lNnFkzoiG4s. The source code for the entire application can be found in "***UI Applications/Android Application"*** folder.

For simply installing application: you can transfer the .apk file to your Android device. Also, computer application can be found in ***"UI Applications --> Windows/Mac Application"*** folder. The file is named **rating.jar** and simply double click on it to run the jar file.

double click rating.jar file

Files in the 'Windows/Mac Application' folder are:

1. Comment.java - For handling the comments/reviews on the instructor.

2. Search.java - For searching the instructor into the dataset

3. rating.sql - This is a SQLite file that can be used to fire queries if you want to rate a professor.

All above 3 files have dependency on JDBC driver and so, it needs a manual configuration for SQL. ***Please run rating.jar in order to see the GUI application of it.***

Naïve Bayesian Classifier:

I have implemented a Naïve Bayesian Classifier that identifies correctly and incorrectly assigned entities. The program for classifier can be located in folder **"Naive Bayes"** by name classifier.py.

Running instruction: ***python classifier.py instructor.data instructor.test***

For an instance, if my *dataset – instructor.data* has 7490 samples of professors then, my classifier gives 81.06% of accuracy in identifying the correctly classified instances.

```
=== Confusion Matrix ===

 a         b   <-- classified as

 6000      0 |   a = accepted

 815     675 |   b = rejected
```

Using 10-fold cross validation (WEKA), classifier's accuracy = 89.07%. The dataset for this is ***instructor.arff***, which is in **"Dataset Obtain"** folder.

Part II: Description of the main strategies for the project

Use of research papers: Learnt a lot by reading research papers and articles in the folder **"Reference Papers"**.

Generation of dataset:

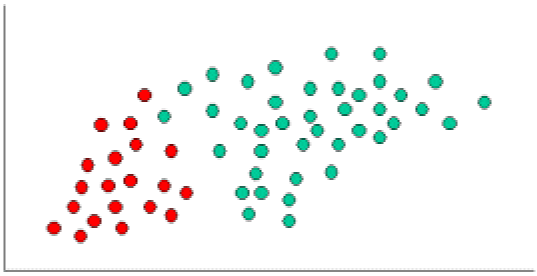
The strategy of **curling the URL** by developing a **scrapper** was useful since it saved a lot of time and manual work. The integration of **scraperwiki** that gives data in wiki like fashion helped me to create a JSON file of instructor information. Preprocessing was intense to check each column for special characters and this was important to create a compatible arff file for WEKA.

Secondly, missing values were cleverly handled by making an array for suspected words and filtering out those entries in order to remove them. More frequent words were: ***Unknown and Other.*** Similarly, trimming out the blank entries with **"tr"** command on Mac platform.

BeautifulSoup library for python made it easy to navigate to different pages of ratemyprofessors.com from terminal and grab the required data (version bs4)

Dataset follows halo effect: Cognitive bias in which you give your opinion based on the feeling/thought developed by looking at the others feedback. Same is observed with students being affected by the others feedback on ratemyprofessors.

Classifier construction:



RED – Rejected, GREEN – Accepted

As indicated, **the objects/instructors can be classified as either GREEN (Accepted) or RED (rejected)**. Our task is to classify new cases as they arrive, i.e., decide to which class label they belong, based on the currently existing instructors. Since there are twice as many GREEN instructors as RED, it is reasonable to believe that a new case (which hasn't been observed yet) is twice as likely to have membership GREEN rather than RED.

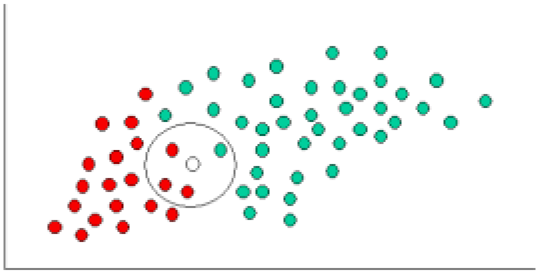
Prior Probability of Accepted: number of GREEN instructors / total number of instructors

Prior Probability of Rejected: number of RED instructors / total number of instructors

For an instance in above figure, since there are total of 60 instructors, 40 of which are GREEN and 20 RED, prior probabilities for class membership are:

Prior Probability for GREEN: 40 / 60, Prior Probability for RED: 20 / 60

Having formulated prior probability, I am ready to classify a new instructor (WHITE circle in the diagram below). Since the instructors are well clustered, it is reasonable to assume that the more GREEN (or RED) instructors in the vicinity of X, the more likely that the new cases belong to that particular color. To measure this likelihood, I have **drawn a circle around X** that encompasses a number (to be chosen a priori) of points irrespective of their class labels. Then, calculated the number of points in the circle belonging to each class label. From this, likelihood will be:



Likelihood of X given GREEN is directly proportional to:
Number of GREEN in vicinity of X / Total number of GREEN cases.
Similarly, calculate for the RED cases.

From the example above, it is clear that Likelihood of X given GREEN is smaller than Likelihood of X given RED, since the circle encompasses 1 GREEN object and 3 RED ones. Therefore, probability of **X given GREEN = 1/40 and of X given RED = 3/20**

As per the definition of Bayesian analysis, combining both sources of information, i.e., the prior and the likelihood, to form a posterior probability using the so-called Bayes' rule and this will produce the final classification.

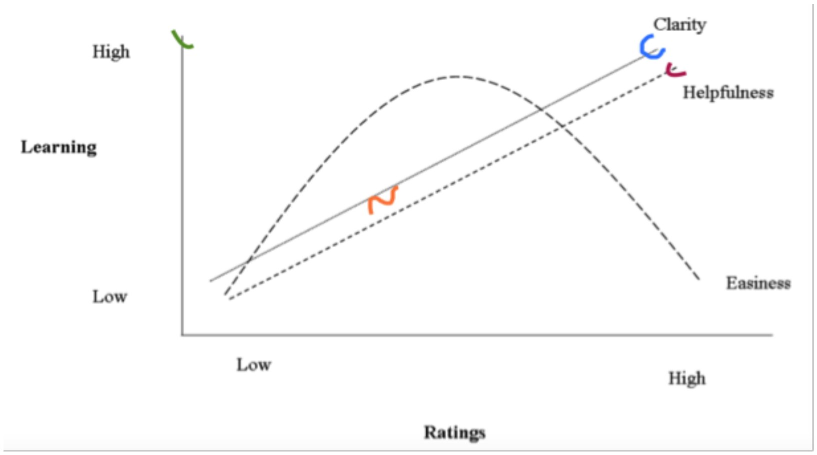
Example: Posterior Prob. Of X being GREEN = Prior Prob. Of GREEN * Likelihood of X given GREEN = (4/6) * (1/40) = 1/60 and,

Posterior Prob. of X being RED = 1/20

Finally, I classify X-instructor as RED (Rejected) since its class membership achieves the largest posterior probability. Based on the similar concept, I have constructed the Naive Bayesian Classifier.

<u>Evaluations & Observations</u>:

1. The graph of colleges vs. total ratings in figure **1.PNG** of folder ***Observations*** shows that students post more about colleges like Duke & University of Texas, Dallas rather than Harvard or Stanford Universities.
2. The plot of total ratings vs. the year clearly shows that the ratemyprofessors has gain a lot of attention over the years and therefore, it should be taken seriously by each and everyone. The statistics shows that currently 84% of students use ratemyprofessors and out of them, 24% posts reviews on the instructors. This can be found in figure **2.PNG** of folder ***Observations.***
3. Review posting frequency of students affects the overall rating of an instructor. This can be clearly depicted by the scatter plot where the higher posting frequency is pulling up the overall quality of an instructor. However, the average value stands in between 2 and 3 where a straight line is shown in figure **3.PNG** of folder ***Observations.***
4. Similar is the case with easiness vs. overall rating plot at figure **4.PNG** in same folder.
5. Another interesting observation could be the impact of type of feedback given by a student on an instructor: positive or negative. Figure **5.PNG** shows that even though the posting frequency is not great but the overall rating is good showing that the feedback given was more on a positive side. To visualize this, ***please check the green remark*** in the same figure.
6. Learning can be more, given that clarity and helpfulness are at their peak values. That said, if an instructor is clear in his/her topics and helps out the students in office hours, then learning by a student could be maximum. This can be observed in the graph below:



7. Several other facts can be seen in figures from **Observations** folder like University of Florida has maximum ratings amongst all the popular colleges in figure **9.PNG**.

Part III: Evaluation results and discussions

Describe/discuss evaluation results:

1. Evaluation shows that the instructors are mostly accepted with higher values of easiness, helpfulness and clarity factors. On an average, the rating falls between 2.5 and 3.5 and the result is accepted for major number of faculties.
2. In fact, in our school San Francisco State University, rating falls between 3 and 4 resulting into the acceptance of all the faculties.
3. Another interesting discussion would be on the specific set of instructors in the school that are responsible for major ratings. For an instance, Arizona State University has certain instructors that contribute to 11% of the ratings. This can be seen in the figure **8.PNG** of **"Observations"** folder.

What are the effects of data size and attribute reduction?

Data size made a major difference in obtaining useful results. Initially, the dataset was having attributes like degree of instructor, instructor experience, etc. that did not provide sufficient information in drawing relationships.

Regression analysis was possible on ratemyprofessors data with relationships between easiness, helpfulness, clarity and ratings of the instructor. This helped in creating Naïve Bayesian Classifier since it provided attributes to decide class of ACCEPTED / REJECTED.

Attribute reduction: Easiness of an instructor was excluded in the beginning phase with the classifier giving an accuracy of 75%. However, on understanding the current statistics of ratemyprofessors, it shows that easiness can play an important role in affecting the overall quality of an instructor promoting them in the future semesters. On inclusion of it, overall rating was greatly affected in pulling up most of the instructors. This agrees with the Wilson's research in 1998 that good graders are always evaluated well.

Compare classifier with Naïve Bayes Implemented in homework:

On comparison with the Naïve Bayesian Classifier implemented in the homework, I found out that the classifier accuracy is nearly the same at around 82% for the multiple attributes in the dataset like college name, instructor name, easiness, total ratings, helpfulness, clarity and overall ratings.

In an instance of 7490 samples, 6000 instructors were accepted with others being rejected.

This accuracy can be greatly improved given that the instructor experience is taken into consideration.

Part IV: Conclusions

By developing an Instructor Assessment System, I have addressed an important problem of analyzing the opinion of students about the instructors. This is something, which is always being less focused and needs analysis so that the students can get best instructors in educational institutions.

Top 10 Data Mining mistakes like ‘Lack of proper data’ and ‘Believe the best model’ should be corrected in the early stages since it saves a lot of time in preprocessing phase.

UI applications can be effective in validation of results. For an instance, UFL has better ratings than CSULA and this can be validated by selecting the universities in the application (future work)

Neglecting an attribute can make a major difference (in my case, consideration of an easiness attribute)

Assigning weights can lead to validity and then, we can normalize it so that the value falls between 1 and 5 as per the restrictions enforced by the dataset.

Part V: Comments and suggestions

Comments: Overall, it is a very informative class and best part was the students being allowed to choose their own system to be developed as a part of Data Mining term project. I, myself, learned a lot from using the mining technique in educational system and drawing observations from them.

The allocated time for the mini projects completion was just sufficient since it take some time to understand and develop our own algorithm. Also, the way research paper presentations were scheduled during the semester helped us to improve our confidence to present in front of the class, obtain information from real world examples presented by other groups, and I personally enjoyed being in a group, practicing together and getting problems solved from instructor.

Since it was a tough semester for my mom being sick, instructor was always supportive boosting my confidence and motivating me to smoothly carry out the academics during my hard time.

Teaching Assistance was helpful and has given us time apart from the office hours in resolving the issues.

Future class improvements: My suggestion would be to include R language as an option to write programs in mini projects and term projects.

During this semester, I have written a lot in R (addition to Java, python) and it looks like it helped me in doing additional analysis. For an instance, I could develop classifier in R by making use of RandomForest library. Also, ***R language has a less complexity*** in importing of the files (easily accepts .txt, .xlsx, .csv and much more) as compared to arff format of WEKA.

Graphical plots construction is easier in R and gives better visualization as compared to other tools. ***Orange tool*** for data mining would be a good idea to explore since it is handy and provides result by drag and drop (http://orange.biolab.si)

***Entire Instructor Assessment System can be found here: Instructor Assessment System