INDENG 242: Applications in Data Analysis, Fall 2022

Assignment 2

November 13, 2022

Problem 1: Predicting Useful Questions on Stack Exchange (70 points)

Stack Overflow is a very popular question-and-answer website, featuring questions on many different computer programming topics. The Stack Exchange Network also features many different spin-off sites on a wide variety of topics. Users of Stack Overflow and its subsidiary Stack Exchange communities can post questions and also provide answers to questions. Users who have earned certain privileges can also vote on the quality/usefulness/helpfulness of both questions and answers. For each question (or answer), privileged users who have read the question can elect to provide either an "upvote" or a "downvote" for the question. An upvote constitutes an endorsement of the question's usefulness for other users, whereas a downvote signals that the question is likely to be especially not useful (or has been sloppily written, or does not show enough research effort, etc.). The "score" of a question is equal to the total number of upvotes for that question minus the total number of downvotes.

In this problem, you will work with question data from Stack Exchange, made available as part of the Stack Exchange Network "data dump".[1] The particular dataset that you will work with contains all of the questions asked during 2020 in the Cross Validated[2] Stack Exchange community, which concerns topics such as statistics, machine learning, data analysis, and data visualization. The data is contained in the files `stack_stats_2020_train.csv` and `stack_stats_2020_test.csv` and has been randomly split, with 70% in the training set and 30% in the testing set. Each observation records text data associated with the title and the body of the associated question, text data for the tags of the question, and also the score of the question. Table 1 describes the dataset in further detail.

After a question is posted, it would be beneficial for Stack Exchange to get an immediate sense of whether the question is useful or not. With such knowledge, the website could automatically promote questions that are believed to be useful to the top of the page. With this goal in mind, in this problem you will build models to predict whether or not a question is useful, based on its tags and the title and body text data associated with each question. To be concrete, let us say that **a question is useful if its score is greater than or equal to one.**

[1] https://archive.org/details/stackexchange
[2] https://stats.stackexchange.com/

Table 1: Description of the dataset `stack_stats_2020`.

Variable	Description
`Title`	Title text of the question
`Body`	Main body text of the question (in html format)
`Tags`	List of tags associated with the question (the format is `<tag1><tag2>...`)
`Score`	Score of the question, equal to total number of upvotes minus total number of downvotes

a) (30 points) Start by cleaning up the dataset to get it into a form where we can apply statistical learning methods to predict our dependent variable of interest. Please briefly describe each step that you took to clean and process the data. Here are some suggestions:

i) The Python library `BeautifulSoup` is useful for dealing with html text. In order to use this library, you will need to install it first by running the following command:

```
conda install beautifulsoup4
```

in the terminal. In the code, you can import it by running the following line:

```
from bs4 import BeautifulSoup
```

ii) When converting html texts that include multiple lines of texts to plain texts using `get_text()` from `BeautifulSoup`, the output texts still contain one type of html string, `\n` or the linebreak character. In other words, `get_text()` does not remove the linebreak characters. You will need to remove them using a proper method of your choice.

iii) You should also inspect several texts from `Body, Tags` and `Title` to see if there is any other transformation needed. You may need to write a custom transformation function.

iv) Once you have all texts in plain texts, you can use the Python library `nltk` to do text cleaning and generate document term matrices as we did in Lab.

v) Notice that there are words that appear in more than one column. For instance, a question might have the term 'regression' in its title, body and tags. However, the terms 'regression' that appear at the three positions of this post may give different implications about this post. (For example, using 'regression' in the title may signal something different than 'regression' as a tag.) For this reason, the three types of text data that we are dealing with should be processed *independently*, and you should use different column names (e.g., `regression_title`, `regression_body`, `regression_tags`).

b) (40 points) Build models with Logistic Regression, LDA, and CART (with cross-validation to select the best `ccp alpha` from 1 to 10 with each step equal to 0.01), and evaluate the models on the test set.

Report on the details of your training procedures and final comparisons on the test set. Use your best judgment to choose a final model and explain your choice. Use the **bootstrap** to asses the performance of your single chosen final model in a way that properly reports on the variability of the relevant performance metrics (accuracy, TPR, and FPR).

Problem 2: Theoretical questions (30 points)

(15 points) Exercise 8.4 in the textbook.

4. This question relates to the plots in Figure 8.12.

   (a) Sketch the tree corresponding to the partition of the predictor space illustrated in the left-hand panel of Figure 8.12. The numbers inside the boxes indicate the mean of Y within each region.

   (b) Create a diagram similar to the left-hand panel of Figure 8.12, using the tree illustrated in the right-hand panel of the same figure. You should divide up the predictor space into the correct regions, and indicate the mean for each region.

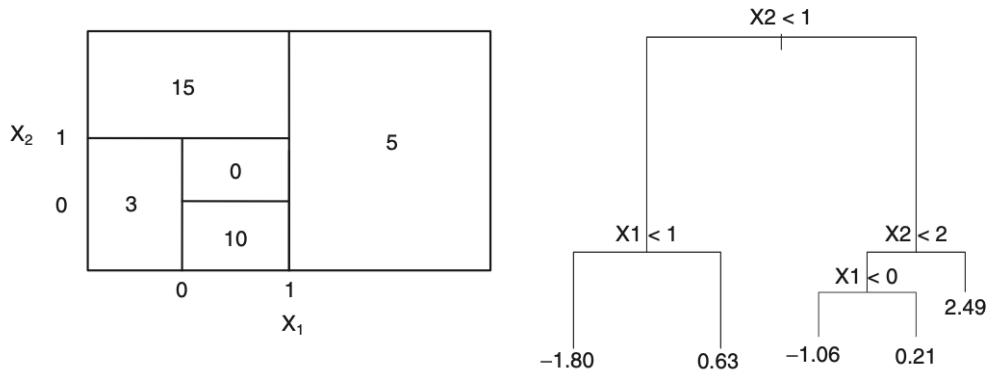

FIGURE 8.12. Left*: A partition of the predictor space corresponding to Exercise 4a.* Right*: A tree corresponding to Exercise 4b.*

Figure 1: ex 8.4 and reletive figures.

(15 points) Exercise 10.3 in the textbook.

3. In this problem, you will perform K-means clustering manually, with $K = 2$, on a small example with $n = 6$ observations and $p = 2$ features. The observations are as follows.

Obs.	X_1	X_2
1	1	4
2	1	3
3	0	4
4	5	1
5	6	2
6	4	0

   (a) Plot the observations.

   (b) Randomly assign a cluster label to each observation. You can use the `sample()` command in `R` to do this. Report the cluster labels for each observation.

   (c) Compute the centroid for each cluster.

   (d) Assign each observation to the centroid to which it is closest, in terms of Euclidean distance. Report the cluster labels for each observation.

   (e) Repeat (c) and (d) until the answers obtained stop changing.

   (f) In your plot from (a), color the observations according to the cluster labels obtained.

Figure 2: ex 10.3.