CS 422: Data Mining

Department of Computer Science
Illinois Institute of Technology
Vijay K. Gurbani, Ph.D.

Fall 2018: Homework 1 (10 points)

Due date: Wed, Sep 19, 2018 11:59:59 PM Chicago Time

Please read all of the parts of the homework carefully before attempting any question. If you detect any ambiguities in the instructions, please let me know right away instead of waiting until after the homework has been graded.

This is the final version of the homework.

1 **Recitation problems** (total points: 3)

1.1 Tan, Chapter 1 (1.5 point divided evenly among the questions)

Besides the lecture, make sure you read Chapter 1. After doing so, answer the following questions at the end of the chapter: 1, 3.

1.2 Tan, Chapter 2 (1.5 point divided evenly among the questions)

Besides the lecture, make sure you read Chapter 2, sections 2.1 – 2.3. After doing so, answer the following questions at the end of the chapter: 2, 3, 7, 12.

2 **Practicum problems** (total points: 7)

2.1 Problem 1 (3 points)

This exercise relates to the College data set, which can be found in the file College.csv. It contains a number of variables for 777 different universities and colleges in the US. The variables are

- Private : Public/private indicator
- Apps : Number of applications received
- Accept : Number of applicants accepted
- Enroll : Number of new students enrolled
- Top10perc : New students from top 10 % of high school class
- Top25perc : New students from top 25 % of high school class
- F.Undergrad : Number of full-time undergraduates
- P.Undergrad : Number of part-time undergraduates
- Outstate : Out-of-state tuition

- Room.Board : Room and board costs
- Books : Estimated book costs
- Personal : Estimated personal spending
- PhD : Percent of faculty with Ph.D.'s
- Terminal : Percent of faculty with terminal degree
- S.F.Ratio : Student/faculty ratio
- perc.alumni : Percent of alumni who donate
- Expend : Instructional expenditure per student
- Grad.Rate : Graduation rate

Before reading the data into R , you should view in a text editor or imported in a spreadsheet.

(a) (0.1 points) Use the read.csv() function to read the data into an R dataframe, called 'college.df' . Make sure that you have the directory set to the correct location for the data. Print the top 6 observations in this dataframe.

(You can use the setwd() function to set the current working directory. For example, assume that College.csv is located in /home/vkg/CS422/Homework-1. Then, setwd("/home/vkg/CS422/Homework-1") will set the current working directory appropriately. Once you do this, you can simply invoke read.csv("College.csv", …) instead of prefxing the path name.)

(b) (0.1 points) Count and print the number of private colleges and the number of public colleges. You may use the table() function to build a contingency table of counts at each combination of factor levels. Recall that a factor in R is a constrained type that can take only a certain predefined values (an enum in Java or C).

(c) (0.7 points) Create two new data frames: one contains all public colleges and the other contains all private colleges. Plot the histogram of the PhD holders in private colleges and public colleges. (You will produce 2 histograms). For each plot, overlay the plot with a density curve. Make sure you use color and labeling to make your graph look attractive. Provide some comments on the histograms (i.e., are private colleges top-heavy with respect to PhD faculty? Are public colleges top-heavy?)

(Hint: Use dplyr::filter() to create the data frames. To create a histogram, use the hist() function. To overlay a density curve on top of a histogram, use lines(density(…)). The lines() command plots the result on the existing plot, i.e., it will overlay the line on top of the existing plot.

(d) (0.6 points) Create a new data frame that contains the data in the college.df dataframe, but sorted on the attribute "Grad.Rate". Then, print the top 5 colleges that have the minimum graduation rates (print the name and graduation rate), and the top 5 colleges that have the maximum graduation rate (again, print the name and graduation rate).

(Hint: To sort the data frame on attributes, use the dplyr::arrange() method. To print the two attributes --- name and graduation rate --- use the dplyr::select() method.)

(e) (1.5 points divided evenly)

i. Use the summary() function to produce a numerical summary of the variables in the data set.

ii. Use the pairs() function to produce a scatterplot matrix of the first ten columns or variables of the data. Recall that you can reference the first ten columns of a matrix A using A[,1:10] .

iii. Use the boxplot() function to produce side-by-side boxplots that help answers the following question: Which alumni donate more to their colleges --- those who go to public schools or those who go to private schools?

To do this, you will use the boxplot() function, but you will group the attribute you are interested in by a control group. The control group here is the attribute Private (which is ‘Yes’ or ‘No’), and the attribute of interest here is perc.alumni. The format is boxplot(x, data=), where x is a formula and data= denotes the data frame providing the data. An example of a formula is y~group where a separate boxplot for numeric variable y is generated for each value of group. Label the X- and Y-axes appropriately and provide a main= parameter to the boxplot() command for a graph title. You should see two boxplots if all works.

As an added resource, take a look at https://www.statmethods.net/graphs/boxplot.html

iv Use the boxplot() function to produce side-by-side boxplots that help answers the following question: Which colleges --- public or private --- employ more Ph.D.’s?

v. Create a new qualitative variable, called Elite by binning the Top10perc variable. We are going to divide universities into two groups based on whether or not the proportion of students coming from the top 10 % of their high school classes exceeds 50 %.

```
      > Elite <- rep(“No”, nrow(college))
      > Elite[college$Top10perc > 50] <- “Yes”
      > Elite <- as.factor(Elite)
      > college <- data.frame(college, Elite)
```

Use the summary() function to see how many elite universities there are.

vi. Use the hist() function to produce some histograms with differing numbers of bins for a few of the quantitative variables. You may find the command par(mfrow=c(2,2)) useful: it will divide the print window into four regions so that four plots can be made simultaneously. Modifying the arguments to this function will divide the screen in other ways.

vii. Continue exploring the data, and provide a brief summary of what you discover. (Yes, I know this is an open ended question, but discovery is an important part of data mining.)

2.2 Linear regression (4 points divided among subparts as indicated)

You are given 398 observations of in nine dimensions of a city-cycle fuel consumption in miles per gallon, to be predicted in terms of three multivalued discrete and 5 five continuous attributes. The target attribute to be predicted is “mpg”, and it is one of the nine dimensions.

Information on the dimensions is as follows:

mpg (continuous)	weight (continuous)
cylinders (multi-valued, discrete)	acceleration (continuous)
displacement (continuous)	model.year (multi-valued, discrete)
horsepower (continuous)	car.name (string, factor, unique for each instance)

You are to run linear regression models on the dataset to predict the miles per gallon (**mpg**).

The dataset is provided to you in auto-mpg.csv.

Please follow the directions below carefully.

(a) [0.3 points divided evenly by sub-parts] **Clean the dataset**.

(i) The dataset is mostly clean except that there are 6 "?" in the **horsepower** attribute. Print the indexes of the rows that contain the "?" in the **horsepower** attribute, and then clean the dataset by removing the rows that contain the "?".

(Hint: Study the Help page of which() to figure out what these indexes to these rows may be.)

(ii) Once you have removed the rows that contained "?", run str() on the dataframe and observe the type of the attribute **horsepower**. You will note that it is of type "Factor". Change the type of the attribute **horsepower** from "Factor" to "integer". (Hint: Take a look at the Help page for as.integer()). Print the output of the str() command to ensure that all attributes (except **car.name**) are either numeric or integer.

All of the remaining sub-parts will use the cleaned dataset resulting from this answer.

(b) [0.7 points divided evenly by sub-parts] **Simple regression** (univariate)

Study the attributes of the dataset. Pick one attribute that you think will be strongly correlated with the response variable (**mpg**). (You may run pairwise correlations to see which predictor is positively or negatively correlated with the response.) Print the variable chosen and the correlation plot. (For the correlation plot, please use psych::pair.panels()).

Perform a simple regression on **mpg** using the predictor. Print the summary of the regression model and comment on how well the model fits the data by studying the R^2, RSE and RMSE.

(c) [0.2 points] For the model in (b), plot the X-Y (or scatterplot) of the predictor and the regressor. On that plot, overlay the the regression line obtained from the model. Label the X and Y axes appropriately and provide a main title for the graph.

For the remaining questions below, set the seed as follow and divide the dataset in two parts: a training part and a test part. The training part will be used to build your regression model, and the testing part will be used to test your regression model to see how effective it is. The following commands are to be entered exactly as shown.

```
      > set.seed(1122)
      > index <- sample(1:nrow(cars.df), 0.80*dim(cars.df)[1])
      > train.df <- df[index, ]
      > test.df <- df[-index, ]
```

Above, the first statement sets the seed for the pseudo-random number generator; **use the same seed as shown above**. The second statement samples (without replacement) about 314 numbers between 1 and the total rows of the data frame that you read the dataset in. The third statement creates the **training** subset from the dataset, and the final statement creates the **test** subset from the dataset (note the use of -index as the row variable to gather all the rows that were not in the sampled index). At the end of these statements, you have two new dataframes, one containing the training data and the other containing the test data.

(d) [0.2 points] You are to pick 3-4 attributes that will act as regressors (predictors, or the independent variable) in your regression model. Using the correlation plot from (a), study the correlation of the attributes versus the response variable to narrow down the list of regressors you will use.

Once you have the list of regressors, plot the correlation between the response variable and the chosen regressors on the **training dataset**.

(e) [1.0 points divided evenly by sub-parts] **Multiple regression (multi-variate)**

Run your regression model using the regressors picked in (d).

(i) Print a summary of your model.

(ii) Determine which predictors are statistically significant and which are not. Eliminate those that are not statistically significant by going back to (d) and examining other options. Note that if you end up with an R^2 of 1.0, you may be overfitting (this means that the model is picking up patterns in the training set that may not be in the test set); if that happens, you will need to change the combination of regressors to give you a R^2 that is less than 1.0. (Please do not ask what a good value of R^2 should be; you should have notes from class that guide you on choosing R^2.)

(iii) Print the summary of the regression model and comment on how well the model fits the data by studying the R^2, RSE and RMSE.

(f) [0.3 points] Plot the residuals of the model. Comment on the shape of the graph of the residuals.

(g) [0.3 points] Plot a histogram of the residuals. Does the histogram follow a Gaussian distribution?

(h) [0.4 points] Using the predict() method, fit the **test** dataset to the model. To do so, you will create a new data frame as follows: get the predictions (fitted values) and put them in a new dataframe as a column vector. Put the test response variable as the second column vector in the new dataframe. Use R commands to find out how many of the fitted values matched (exactly) the **mpg** in the **test** dataset.

(i) [0.6 points] Use R to calculate the residual vector for the predictions on the **test** dataset. Then, using the residual vector, calculate and print the following statistics:

1. RSS (Residual Sum of Errors).
2. TSS (Total Sum of Errors).
3. The F-statistic.
4. The RSE (Residual Standard Error).
5. The RMSE (Root Mean Square Error)

What can you say about how well your model is performing based on the statistics above?