New York University Shanghai

# Problem Set 4

**BUSF-SHU 210: Business Analytics (Spring 2019)**

## 1. Please briefly answer the following questions:

(a) (Existence of a pure tree.) For a general classification problem with $p-$dimensional covariate $X = (X_1, X_2, \cdots X_p)$ and dependent variable $Y \in \{0, 1\}$. Show that, as long as in the training data set any two data points with the same covariate $X$ have the same outcome $Y$, we can always build a CART tree with pure leafs. Consider a classification problem with two independent variables $X_1$ and $X_2$. Please try to plot a CART tree with *pure leafs only* using the following training data set (for a pictorial illustration, see Figure 1):

| Data Point Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $X_1$ | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |
| $X_2$ | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| $Y$ | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

(b) (Gini index of a tree.) We assume that the dependent variable $Y$ takes a binary value, i.e., $Y \in \{0, 1\}$. There are two independent variables, $X_1$ and $X_2$. Consider a CART tree as plotted in Figure 2. For the training data set that are used to train this tree, the number of outcomes in each leaf is summarized in the following table:

| | $L_1$ | $L_2$ | $L_3$ | $L_4$ | $L_5$ |
|---|---|---|---|---|---|
| $Y = 0$ | 30 | 15 | 5 | 45 | 10 |
| $Y = 1$ | 10 | 25 | 25 | 20 | 25 |

What is the Gini index of the tree in Figure 2? What is the Gini index of the training data set?

(c) (Constructing a CART tree with the help of Gini index.) Consider a classification problem with two covariates $X_1$ and $X_2$, and a binary outcome $Y \in \{0, 1\}$. The training data set has nine data points (for a pictorial illustration, see Figure 1):

| Data Point Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| $X_1$ | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 3 |
| $X_2$ | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| $Y$ | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |

Please try to build CART tree by recursively minimizing the Gini index of the tree. We set the maximum depth of the tree to be 2. Recall that, we start with all the data on the root node of the CART tree, and then recursively split the tree until the depth of the tree reaches 2.

(d) (Regression tree). In regression tree, we exploit the tree structure to make predictions about the outcome. In each leaf of a regression tree, we use the *sample mean* of all the outcomes in this leaf (in the training set) as the prediction. As in linear regression, we use the sum of squared errors $SSE$[1] (also called the residual sum of squares $RSS$):

$$SSE = \sum_{i=1}^{n}(Y_i - \hat{f}(X_i))^2$$

as a measure of model performance in regression tree. Consider a training data set of size 10, which we use to fit the regression tree as plotted in Figure 3. Each leaf of the tree has 5 data points. The outcomes (i.e., $Y$) of the data points in each leaf is summarized in the following table:

| Data Point Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Leaf | $L_1$ | $L_1$ | $L_1$ | $L_1$ | $L_1$ | $L_2$ | $L_2$ | $L_2$ | $L_2$ | $L_2$ |
| $Y$ | 13.4 | 12.1 | 15.3 | 14.8 | 11.7 | 2.3 | 3.5 | 1.7 | 3.2 | 0.8 |

(i) What is the prediction of the outcome $Y$ for a testing data with covariate $X = 2.2$?

(ii) Calculate the in-sample $SSE$ of the training data set for the tree in Figure 3. Compare the in-sample $SSE$ with the total sum of squares

$$SST = \sum_{i=1}^{n}(Y_i - \bar{Y})^2$$

of the training data set. Do you think the in-sample performance of the tree is better than the baseline model where we predict the outcome using the sample mean of the training set $\bar{Y}$ for any covariate $X$?

(e) (Building a regression tree.) Consider a regression problem with one covariate $X$, and a continuous outcome $Y \in \mathbb{R}$. The training data set has 5 data points (for a pictorial illustration, see Figure 4):

| Data Point Index | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| $X$ | 0.6 | 1.2 | 2.5 | 3.7 | 4.9 |
| $Y$ | 10 | 8 | -4 | 3 | 4 |

To build a regression tree, we also adopt the idea of recursively splitting the data by one of the covariate. Specifically, please try to build a regression tree by recursively minimizing the $SSE$ of the tree. We start with all the training data on the root node of the CART tree, and then split the tree *once*. The model therefore has a single root and two leaves. Please draw the fitted regression tree model and report the in-sample-$R^2$ (defined as $1 - SSE/SST$).

(f) Do you think whether the scales/units of the covariates matter in building a CART tree? Why or why not?

[1] In the lecture, we use the average sum of squares. The two measures are equivalent, different from each other by a scale of sample size.

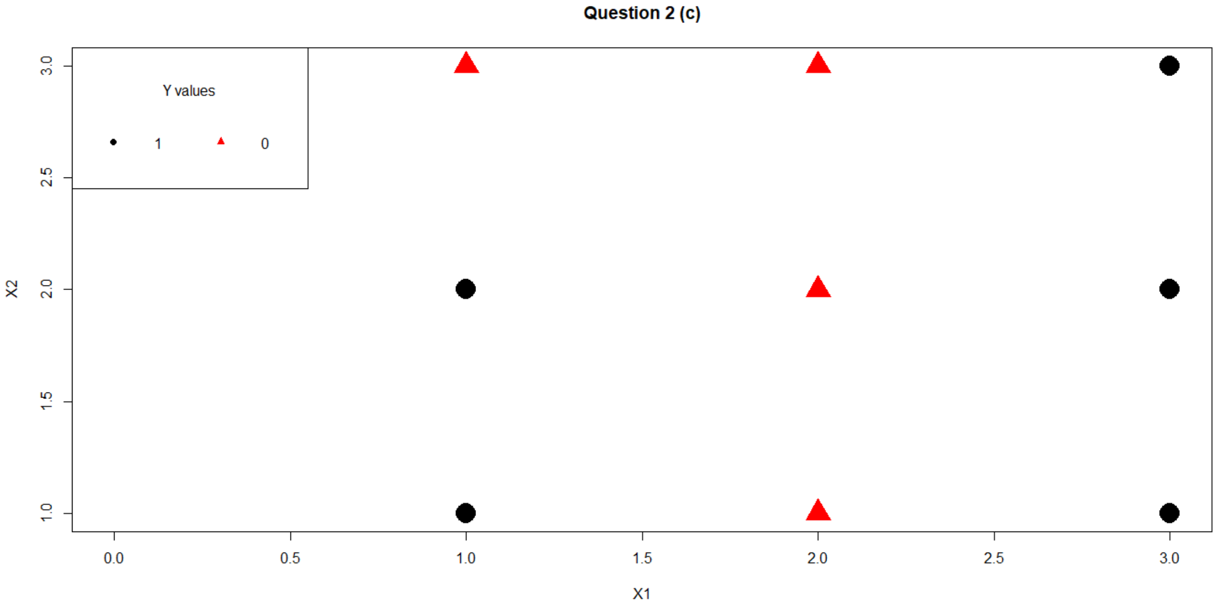

Figure 1: Question 1(a,c)

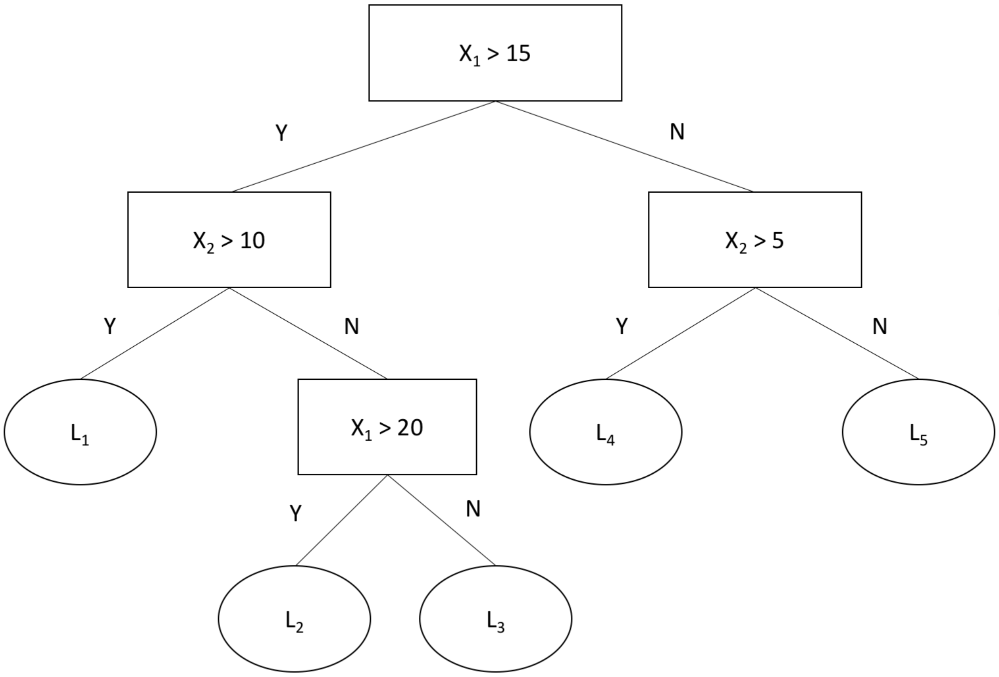

Figure 2: Question 1(b)

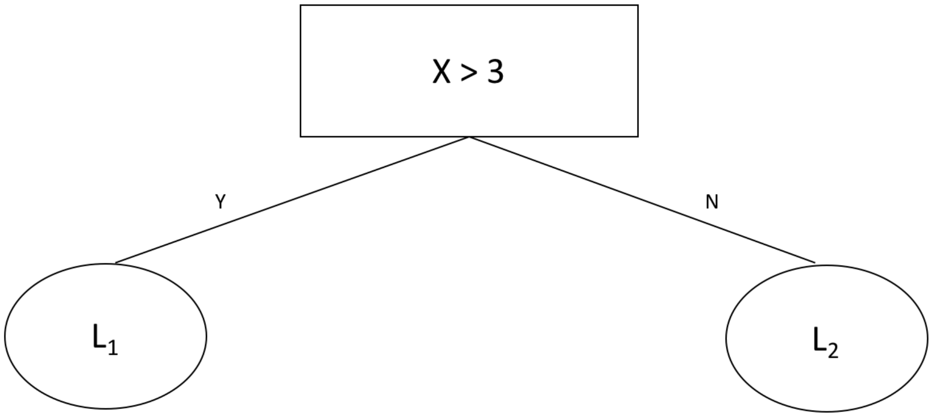

Figure 3: Question 1(d)

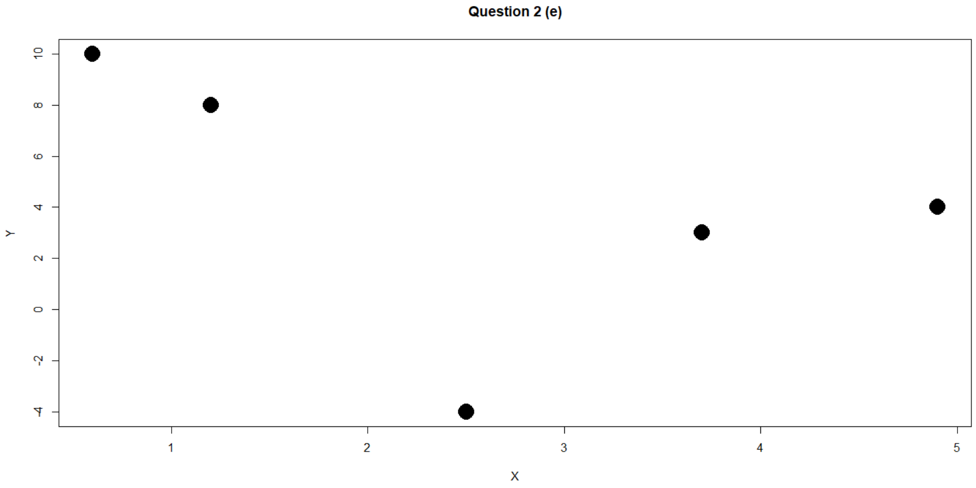

Figure 4: Question 1(e)

### 2. Letter Recognition

In this problem, we will build a model that uses statistics of images of four letters – A, B, P, and R – to predict which letter a particular image corresponds to. The data set `letters_ABPR.csv` contains 3116 observations, each of which corresponds to a certain image of one of the four letters A, B, P and R. The images came from 20 different fonts, which were then randomly distorted to produce the final images; each such distorted image is represented as a collection of pixels, each of which is“on” or “off”. For each such distorted image, we have available certain statistics of the image in terms of these pixels, as well as which of the four letters the image is.

This dataset contains the following 17 variables:

- *letter* = the letter that the image corresponds to (A, B, P or R)
- *xbox* = the horizontal position of where the smallest box covering the letter shape begins
- *ybox* = the vertical position of where the smallest box covering the letter shape begins
- *width* = the width of this smallest box
- *height* = the height of this smallest box
- *onpix* = the total number of “on” pixels in the character image
- *xbar* = the mean horizontal position of all of the “on” pixels
- *ybar* = the mean vertical position of all of the “on” pixels
- *x2bar* = the mean squared horizontal position of all of the “on” pixels in the image
- *y2bar* = the mean squared vertical position of all of the “on” pixels in the image
- *xybar* = the mean of the product of the horizontal and vertical position of all of the “on” pixels in the image
- *x2ybar* = the mean of the product of the squared horizontal position and the vertical position of all of the “on” pixels
- *xy2bar* = the mean of the product of the horizontal position and the squared vertical position of all of the “on” pixels
- *xedge* = the mean number of edges (the number of times an “off” pixel is followed by an “on” pixel, or the image boundary is hit) as the image is scanned from left to right, along the whole vertical length of the image
- *xedgeycor* = the mean of the product of the number of horizontal edges at each vertical position and the vertical position
- *yedge* = the mean number of edges as the images is scanned from top to bottom, along the whole horizontal length of the image

- *yedgexcor* = the mean of the product of the number of vertical edges at each horizontal position and the horizontal position

To build classification models to recognize the letters, you may transform the data type of the variable *letter* into an integer variable by running the code: `letters = letters.replace('A',1)`, `letters = letters.replace('B',2)`, `letters = letters.replace('P',3)`, and `letters = letters.replace('R',4)` (assuming that you load the data and store it in the data frame named "`letters`"). Remember to split the data into training and testing sets.

(a) Build a CART tree model with all other variables as covariates. The tree should be with pure leaves only. Please report the out-of-sample accuracy. According to your model, what is the most important factor to recognize the letter?

(b) Use cross validation to train a CART model with different maximum depths. What is the model you eventually build with cross validation? Report the out-of-sample overall accuracy of the model you build.

(c) Now build a random forest model on the training data with the same set of independent variables. Also vary the value of maximum depth as part (b) to find the best random forest. You can set the number of trees to 25. Report the out-of-sample accuracy for the random forest models. For the best CART model and the best random forest model, which one performs better?

(d) State in your own words why, in general, the random forest model could help improve the prediction accuracy over a CART model.

## 8. Predicting PM 2.5

In this problem, you will try to use both time series and regression models to predict the PM 2.5 level of Beijing. We use the data set `PM25.csv`, which contains the PM2.5 and weather information of Beijing from 2010 to 2014. This data set has the following variables:

- $No$: Data point index
- $year$: year of data in this row
- $month$: month of data in this row
- $day$: day of data in this row
- $hour$: hour of data in this row
- $pm2.5$: PM2.5 concentration ($\mu g/m^3$)
- $DEWP$: Dew Point (°C)
- $TEMP$: Temperature (°C)
- $PRES$: Pressure (hPa)
- $cbwd$: Combined wind direction
- $Iws$: Cumulated wind speed (m/s)
- $Is$: Cumulated hours of snow
- $Ir$: Cumulated hours of rain

Use cross validation and feature engineering techniques to build and select regression model(s) and predict the concentration of PM 2.5 level. You may need to examine the performance of a few candidate models. Please clearly write your training and validation steps. Report the estimated mean squared error of your model under the “true” data generating process.