STA130H1S – Winter 2020

Tutorial 7 Practice Problems - Sample Answers

L. Bolton and N. Moon

Instructions

How do I hand in these problems for the March 5th deadline?

Your complete .Rmd file that you create for at least *Questions 1 and 2* of these practice problems AND the resulting pdf (i.e., the one you 'Knit to PDF' from your .Rmd file) must be uploaded into a Quercus assignment (link: https://q.utoronto.ca/courses/138992/assignments/284432) by 11:59PM, Thursday, March 5th.

What should I bring to tutorial on March 6th?

R output (e.g., output and explanations) for *Questions 1 and 2*. You can either bring a hardcopy or bring your laptop with the output.

Tutorial Grading

Tutorial grades will be assigned according to the following marking scheme.

	Mark
Completion of required problems (due on Quercus the day before your tutorial)	1
Attendance for the entire tutorial	1
In-class exercises	4
Total	6

Practice Problems

Question 1

Using data from the Gallup World Poll (and the World Happiness Report), we are interested in predicting which factors influence life expectancy around the world. These data are in the file `happinessdata_2017.csv` (we first looked at these data in Week 1).

```
happiness2017 <- read.csv("happinessdata_2017.csv")
```

(a) Begin by creating a new variable called `life_exp_category` which takes the value "Good" for countries with a life expectancy higher than 65 years, and "Poor" otherwise.

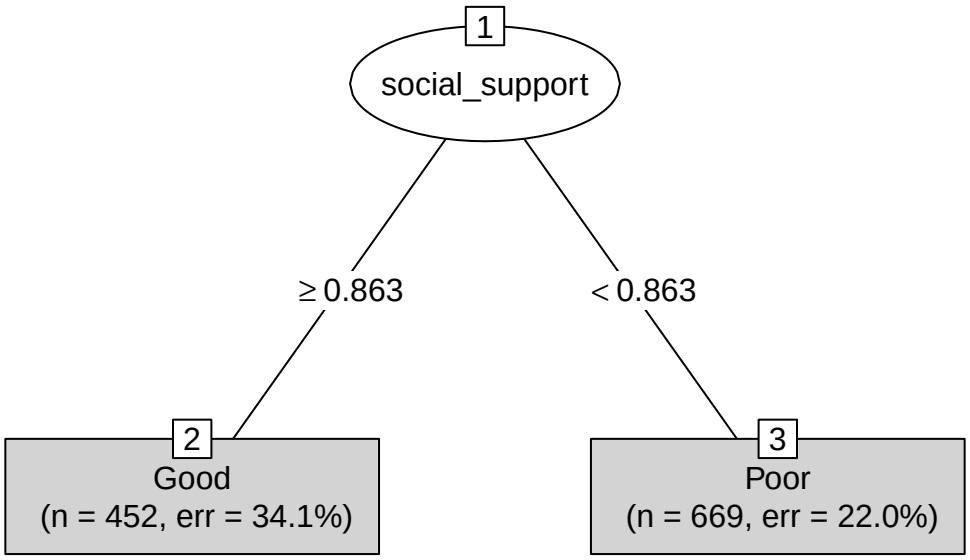
```
happiness2017  <- happiness2017 %>% mutate(life_exp_category = ifelse(life_exp > 65, yes="Good", no="Poo
```

(b) Divide the data into training (80%) and testing (20%) datasets. Build a classification tree using the training data to predict which countries have `Good` vs `Poor` life expectancy, using the only the `social_support` variable as a predictor. Use the last 3 digits of your student ID number for the random seed.

```
set.seed(834);
n <- nrow(happiness2017); # number of obs in the full dataset
n
```

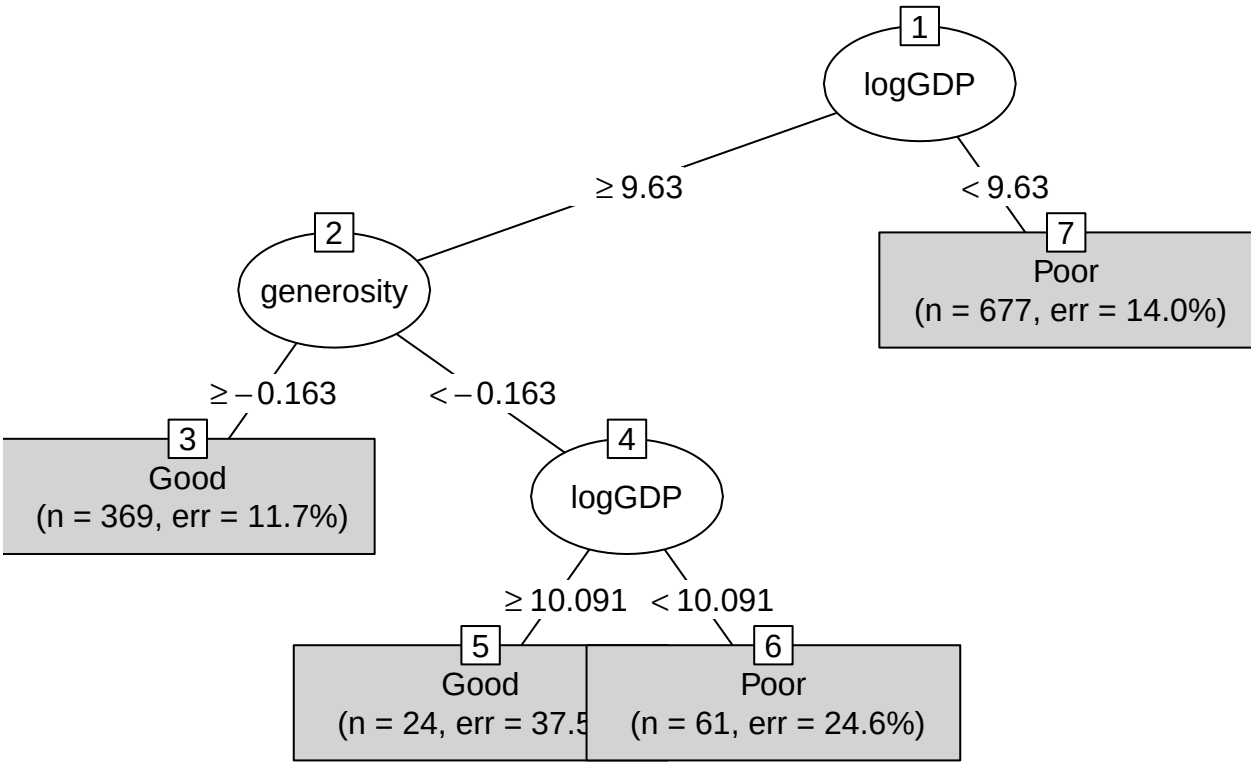
```
## [1] 1420
```

```
# Add row IDs to happiness data
happiness2017 <- happiness2017 %>% rowid_to_column()

# Random sample of 20% of row indices
training_indices <- sample(1:n, size=round(0.8*n))

train <- happiness2017 %>% filter(rowid %in% training_indices)
test <- happiness2017 %>% filter(!(rowid %in% training_indices))

# Fit the tree based on training data
tree1 <- rpart(life_exp_category ~ social_support, data=train)
plot(as.party(tree1), type="simple")
```



(c) Use the same training dataset created in (b) to build a second classification tree to predict which countries have good vs poor life expectancy, using `logGDP`, `social_support`, `freedom`, and `generosity` as potential predictors.

```
# Fit the tree based on training data
tree2 <- rpart(life_exp_category ~ logGDP + social_support + freedom +
                 generosity, data=train)
plot(as.party(tree2), type="simple")
```



(d) Use the testing dataset you created in (b) to calculate the confusion matrix based on the "majority rules" cutpoint (i.e. 0.5) for the trees you built in (b) and (c). Report the sensitivity (True positive rate), specificity (True negative rate) and accuracy for each of the trees. Here you will treat "Good" life expectancy as a positive response/prediction.

```
### Tree from (b)
predicted_tree1 <- predict(tree1, newdata = test, type="prob")
head(predicted_tree1)
```

```
##        Good      Poor
## 1 0.2197309 0.7802691
## 2 0.2197309 0.7802691
## 3 0.2197309 0.7802691
## 4 0.2197309 0.7802691
## 5 0.6592920 0.3407080
## 6 0.6592920 0.3407080
```

```
tree1_preds_majorityRules <- predicted_tree1 %>% as_tibble() %>%
  mutate(prediction = ifelse(Good >= 0.5, "Predict Good", "Predict Poor"))

m <- table(tree1_preds_majorityRules$prediction, test$life_exp_category)
m
```

```
##
##                Good Poor
##   Predict Good   81   39
##   Predict Poor   41  119
```

```
tp <- m[1,1] / sum(m[,1])
tn <- m[2,2] / sum(m[,2])
accuracy <- sum(diag(m))/sum(m)
c(tp, tn, accuracy)
```

```
## [1] 0.6639344 0.7531646 0.7142857
```

For the tree built in part (b), the sensitivity (TPR) is 0.66, the specificity (TNR) is 0.75, and the overall accuracy is 0.71.

```
### Tree from (c)
predicted_tree2 <- predict(tree2, newdata = test, type="prob")
head(predicted_tree2)
```

```
##        Good      Poor
## 1 0.1403250 0.8596750
## 2 0.1403250 0.8596750
## 3 0.1403250 0.8596750
## 4 0.1403250 0.8596750
## 5 0.8834688 0.1165312
## 6 0.8834688 0.1165312
```

```
tree2_preds_majorityRules <- predicted_tree2 %>% as_tibble() %>%
  mutate(prediction = ifelse(Good >= 0.5, "Predict Good", "Predict Poor"))

m <- table(tree2_preds_majorityRules$prediction, test$life_exp_category)
m
```

```
##
##                Good Poor
##   Predict Good   94   11
##   Predict Poor   28  147
```

```
tp <- m[1,1] / sum(m[,1])
tn <- m[2,2] / sum(m[,2])
accuracy <- sum(diag(m))/sum(m)
c(tp, tn, accuracy)
```

```
## [1] 0.7704918 0.9303797 0.8607143
```

For the tree built in part (c), the sensitivity (TPR) is 0.77, the specificity (TNR) is 0.93, and the overall accuracy is 0.86.

(e) Fill in the following table using the tree you constructed in part (c). Does the fact that some of the values are missing (NA) prevent you from making predictions for the life expectancy category for these observations?

	logGDP	social_support	freedom	generosity	Predicted life expectancy category
Obs 1	9.56	0.74	NA	-0.25	**Poor**
Obs 2	10.1	0.84	0.80	10.9	**Poor**
Obs 3	11.2	0.88	0.77	0.1	**Good**

Even though the value of `freedom` is missing for observation 1, we can still predict the life expectancy category for this country because none of the decisions in the tree in part (c) involve the `freedom` variable.

(f) In most cases, two classification trees will make different predictions for some new observations. Using the classification trees you built in parts (b) and (c), fill in the table below with values which would lead the the specified predictions.

logGDP	social_support	freedom	generosity	Pred life expectancy category based on (b)	Pred life expectancy category based on (c)
11	0	*	0.25	**Poor**	**Good**
9	1	*	*	**Good**	**Poor**

The *s can take any numerical value, as the specified values of `logGDP` and `generosity` lead to the specified predictions using the trees we built in (b) and (c) respectively.

Question 2

A classification tree was built to predict which individuals have a disease. Using two different cutpoints, we use this tree to predict disease status for 100 new individuals. Below are confusion matrices corresponding to these two cutpoints.

Cutpoint A

	Disease	No disease
Predict disease	35	20
Predict no disease	3	42

Cutpoint B

	Disease	No disease
Predict disease	23	4
Predict no disease	15	58

a) Calculate the accuracy, false-positive rate, and false negative rate for each classifier. Here, a "positive" result means we predict an individual has the disease and a "negative" result means we predict they do not.

Classifier A - Overall accuracy: (35 + 42) / 100 = 0.77

- False-positive rate: 20 / (20 + 42) = 0.32
- False-negative rate: 3 / (35 + 3) = 0.079

Classifier B - Overall accuracy: (23 + 58) / 100 = 0.81

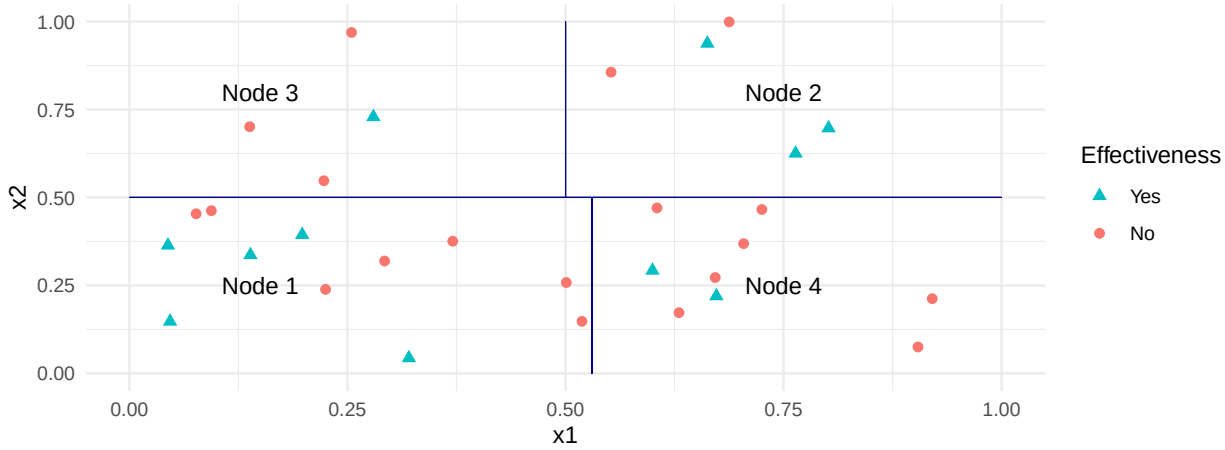
- False-positive rate: 4 / (58 + 4) = 0.064
- False-negative rate: 15 / (15 + 23) = 0.39

b) Suppose the disease is very serious if untreated. Explain which classifier you would prefer to use.

Since the disease is very serious if untreated, we would prefer to use a classifier which does not miss many true cases of disease (in other words, one with a low false-negative rate). Classifier A has a false-negative rate of 0.079, so over 90% of individuals who actually have the disease are correctly diagnosed, on the other hand Classifier B's false-negative rate is almost five times larger (0.39).

Question 3

Data was collected on 30 cancer patients to investigate the effectivness (Yes/No) of a treatment. Two quantitative variables, $x_i \in (0, 1), i = 1, 2$, are considered to be important predictors of effectiveness. Suppose that the rectangles labelled as nodes in the scatterplot below represent nodes of a classification tree.



(a) Suppose we use a cutpoint of 0.5 to make predictions. In other words, if the drug is effective for at least 50% of observations in a terminal node (based on the training data), then we would predict that the drug would be effective for future patients whose characteristics put them in this node. What is the predicted class of each node?

Node	Proportion of "Yes" values in each node	Prediction based on cutpoint of 0.5
1	5/12	Not effective
2	3/5	Effective
3	1/4	Not effective
4	2/9	Not effective

(b) Since the treatment was only effective for 11/30 of the patients in our sample (training data), we could instead use a cutpoint of $11/30 = 0.367$. In other words, if the drug is effective for at least 36.7% of observations in a terminal node (based on the training data), then we would predict that the drug would be effective for future patients whose characteristics put them in this node. What is the predicted class of each node?

Node	Proportion of "Yes" values in each node	Prediction based on cutpoint of 0.5
1	5/12	Effective
2	3/5	Effective
3	1/4	Not effective
4	2/9	Not effective

(c) Did changing the cutpoint from 0.5 to 0.367 change the prediction for any of the nodes?

Yes, changing the cutpoint from 0.5 to 0.367 changed the prediction from "Not Effective" to "Effective" for Node 1.

Question 4

In order to avoid clogging their users' inboxes, email service providers filter out junk (or spam) emails before they clog a user's inbox. To do this, they must predict whether an incoming email is spam (in which case they would direct it to a "junk" folder) or a real email.

```
spam <- read.csv("spamdata.csv") %>% select(-X)
spam <- spam %>% rowid_to_column()
```

(a) Using a training dataset of 3500 email messages you will built 3 trees to classify messages as "spam" or "email".

(i) Divide the `spam` data frame into a training dataset with 3500 observations, and the remaining observations will form the testing dataset. Before randomly splitting the `spam` dataset into the training and testing datasets, use `set.seed()` with the last 3 digits of your student ID number.

```
# Random sample of 3500 row indices
set.seed(145)

training_indices <- sample(1:nrow(spam), size=3500, replace=FALSE)
train <- spam %>% filter(rowid %in% training_indices)
nrow(train)
```

```
## [1] 3500
```

```
# Testing dataset includes all observations NOT in the training data
test <- spam %>% filter(!(rowid %in% training_indices))
nrow(test)
```

```
## [1] 1101
```

(ii) Build `tree1` using the training data, where the only potential predictors are the frequency of dollar signs (`char_freq_dollar`) and the frequency of exclamation marks (`char_freq_exclamation`). We expect that messages with very long strings of these characters are more likely to be spam.

```
tree1 <- rpart(type ~ char_freq_dollar + char_freq_exclamation, data=train)
#plot(as.party(tree1), type="simple")
```

(iii) Build `tree2` using the training data, where the only potential predictors are the frequency of semicolons (`char_freq_semicolon`), the frequency of parentheses marks (`char_freq_parentheses`), and the frequency of hashtag characters (`char_freq_hashtag`). We expect that messages with very long strings of these characters are more likely to be spam.

```
tree2 <- rpart(type ~ char_freq_semicolon + char_freq_parentheses + char_freq_hashtag, data=train)
#plot(as.party(tree2), type="simple")
```

(b) Consider the tree you built in part a(ii), `tree1`. Using this tree, predict whether each message in the testing dataframe is a spam message or a normal email using two cutpoints: 0.5 (i.e. a message is classified as 'spam' if there is at least a 50% chance that it is spam) and 0.95 (i.e. a message is classified as 'spam' only if there is at least a 95% chance that it is spam). For each of these cutpoints, calculate the true positive rate (TPR) and false positive rate (FPR). We will treat "spam" as a positive response for this problem (since the goal is to identify spam emails).

```
# 50% cutpoint
pred_probs50 <- predict(object = tree1, newdata = test, type = "prob")
head(pred_probs50)
```

```
##        email      spam
## 1 0.28666667 0.7133333
## 2 0.04570637 0.9542936
## 3 0.04570637 0.9542936
## 4 0.28666667 0.7133333
## 5 0.29310345 0.7068966
## 6 0.04570637 0.9542936
```

```
preds_cutpoint50 <- predict(object = tree1, newdata = test, type = "prob") %>% as_tibble() %>%
  mutate(prediction = ifelse(spam >= 0.5, "Predicted Spam", "Predicted Email"))

m50 <- table(preds_cutpoint50$prediction, test$type)
m50
```

```
##
##                   email spam
##   Predicted Email   623  102
##   Predicted Spam     63  313
```

```
tpr50 <- m50[2,2] / sum(m50[,2])
fpr50 <- m50[2,1] / sum(m50[,1])
c(tpr50, fpr50)
```

```
## [1] 0.75421687 0.09183673
```

```
# 95% cutpoint
preds_cutpoint95 <- predict(object = tree1, newdata = test, type = "prob") %>% as_tibble() %>%
  mutate(prediction = ifelse(spam >= 0.95, "Predicted Spam", "Predicted Email"))

m95 <- table(preds_cutpoint95$prediction, test$type)
m95
```

```
##
##                   email spam
##   Predicted Email   678  205
##   Predicted Spam      8  210
```

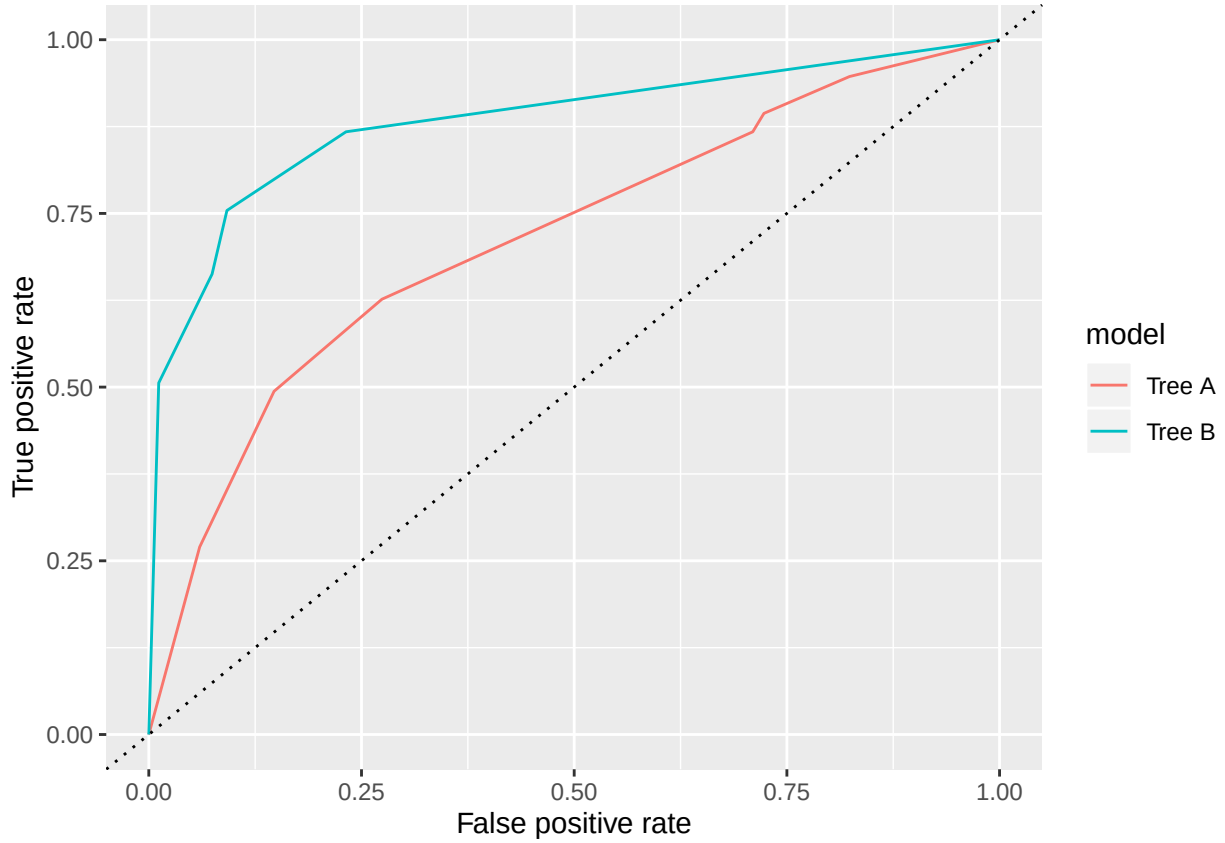
```
tpr95 <- m95[2,2] / sum(m95[,2])
fpr95 <- m95[2,1] / sum(m95[,1])
c(tpr95, fpr95)
```

```
## [1] 0.50602410 0.01166181
```

When we use a cutpoint of 0.5 to identify spam messages, the true positive rate is 0.75 and the false positive rate is 0.09. When we use a cutpoint of 0.95 to identify spam messages, the true positive rate is 0.51 and the false positive rate is 0.01. In other words, when we use a higher cutpoint, a smaller proportion of regular

emails end up in the spam folder (1% vs 9%), but a higher proportion of spam emails make it to our inbox.

(c) Below is a plot of the ROC curves for the two trees you built in parts (a). In part (b), you calculated the TPR and FPR for `tree1`, for two cutpoints. Which of the ROC curves corresponds to `tree1`.



(d) Which of the ROC curves would you prefer to use to predict which incoming messages are spam / regular emails?

Since the ROC curve for Tree B is always above the curve for Tree A, we know that Tree B (which in fact corresponds to `tree1` built earlier) makes more accurate predictions than "Tree A", no matter what cutpoint is used to make predictions. I would prefer to use the Tree B to predict which incoming messages are spam / regular emails.