Take Home Final - BUSN 20800

Connor Dowd

May 28th, 2021

Setup

You will need the following packages loaded:

```
library(tidyverse)
library(lubridate)
library(rpart)
library(glmnet)
```

You will also need access to the following datasets:

```
#Q1
datasets::airquality #Usually installed with base R
#Q2
jtpa #used for HW7 -- see Q2 for download location
#Q4
semiconductors #used for Lecture 5 -- see Q4 for download location
```

Reminders

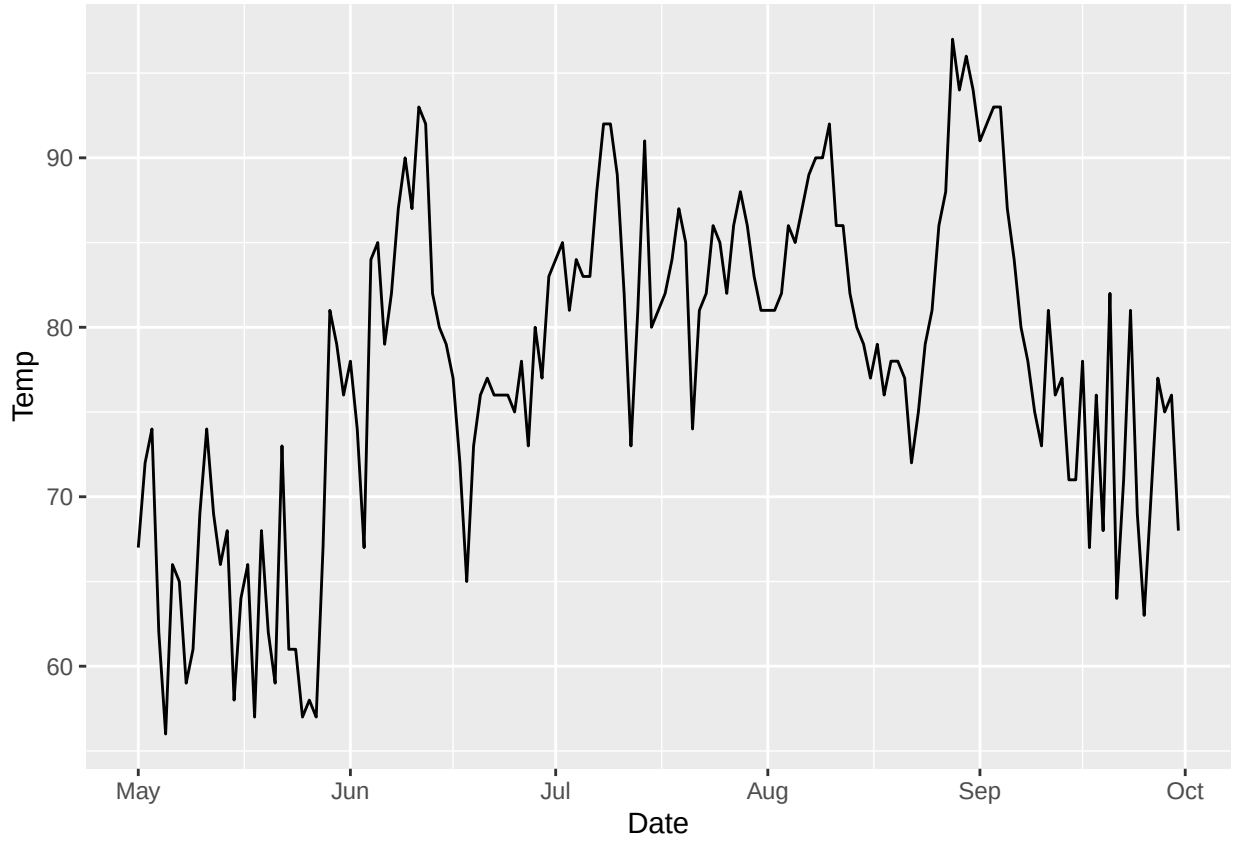
- If, after reading a question 3 times, you are struggling with *interpreting* the question, with understanding what it wants, please email me your question. Don't just stare at it for 5 hours.
  - I will note however, that I will not be working during non-business hours next week. So if your question doesn't arise until Thursday at 9pm, you won't be able to get an answer. For this reason *I strongly suggest you read through questions ahead of time and make sure you understand them.*
- In addition – If I do get questions that seem likely to be important to everyone, I will post on the discussion board. To that extent – it is in your interest to make sure you check the discussion board periodically.
- This is an open-book, open-internet, open-everything examination. However, you are not permitted to discuss the contents of this exam with other members of the class until June 10th.
  - To emphasize the point *there is a lot of likely helpful code in your past homeworks and my solutions.* You are welcome to use that code.
    - If/when you do use code from your group homeworks please make sure you give your group some attribution.

Questions

Q1: Rolling CV

We have data on daily air temperatures in New York City for 5 months (in 1973).

```
temp = datasets::airquality %>% select(Temp,Month,Day)
temp = temp %>%
  mutate(Date = mdy(paste0(Month,"-",Day,"-1973"))) %>%
  select(-Month,-Day)
ggplot(temp,aes(Date,Temp)) + geom_line()
```



I would like to make 3-day ahead forecasts of air temperatures. I'm considering two different models.

- Model 1 takes an average air temperature over the last 3 days (today, yesterday, day before), and uses that average as the forecast. This model believes that future temperatures are likely similar to recent temperatures.
- Model 2 finds the change in temperature between three days ago and today, adds that change to todays temperature, and takes that number as the forecast. This model is based on the idea that recent trends may continue.

Each of those models is fully specified, so I've attached some code below which makes predictions for each one based on a vector of recent temperatures.

```
# functions that take data to make predictions. Give them vectors where the most recent data point is t
pred_mod1_data = function(temps_vec) {
  data = tail(temps_vec,3) #Find the last three days in the data
  mean(data) #Take mean of most recent 3 days
}
pred_mod2_data = function(temps_vec) {
  n = length(temps_vec) #Find length of data
  delta = temps_vec[n]-temps_vec[n-3] #Find change over last 3 days
  temps_vec[n]+delta #Project delta
}
```

So if I wanted a prediction for Day 7, I would give each of those models the data for days 1-4.

```
pred_mod1_data(temp$Temp[1:4])
```

```
## [1] 69.33333
```

```
pred_mod2_data(temp$Temp[1:4])
```

```
## [1] 57
```

```
#Comparison
temp$Temp[7]
```

```
## [1] 65
```

Using a for-loop (or otherwise), loop through the data making OOS 3-day ahead temperature forecasts, and finding the errors in those forecasts for each model. Bear in mind that Day 7 is the first day for which we can make a forecast using both models (so we don't need predictions for days 1-6).

Which model performs better out-of-sample if we care about minimizing the MSE of the forecasts? Which model performs better if we care about minimizing the fraction of the time the errors are bigger than 10 degrees in either direction? Do you have any concerns about drawing strong conclusions choosing between these models using this data?

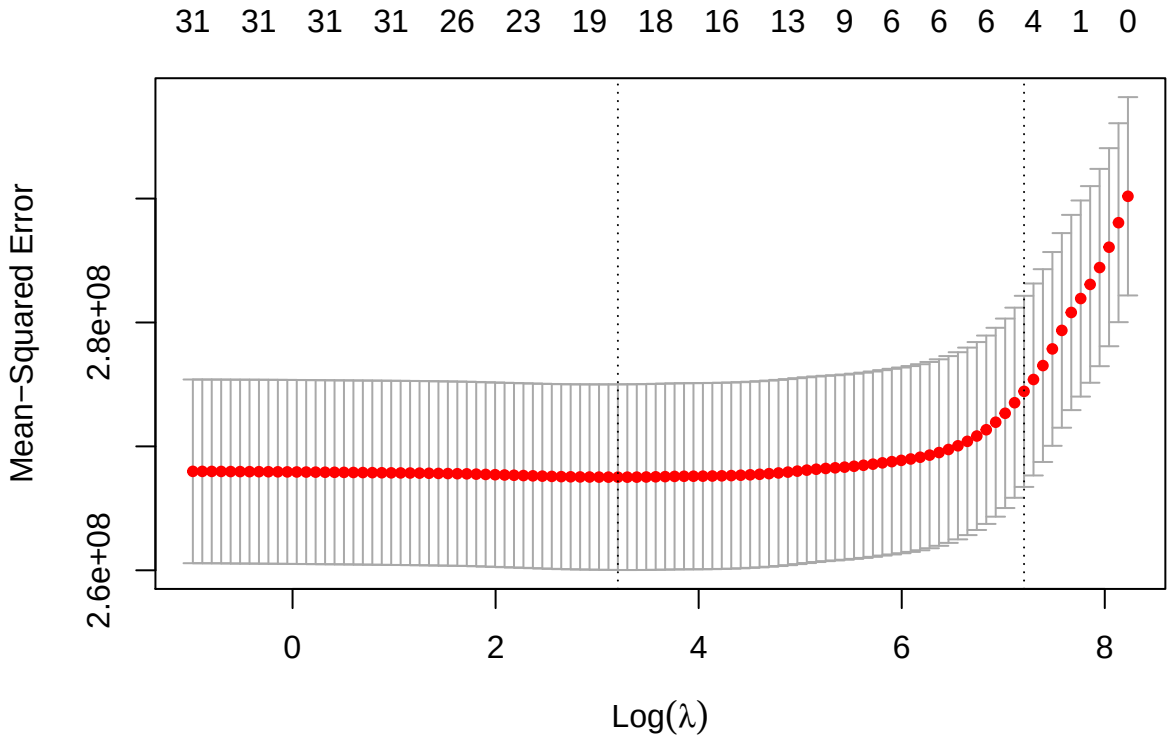
Q2: CV LASSO

In Q4 of the homework last week we estimated fully interacted linear models for the treated observations in the JTPA data (available at https://codowd.com/bigdata/lectures/l15/jtpa/jtpa.RData). We found that the OOS performance of the targeting model we generated was poor – likely a result of bad predictions.

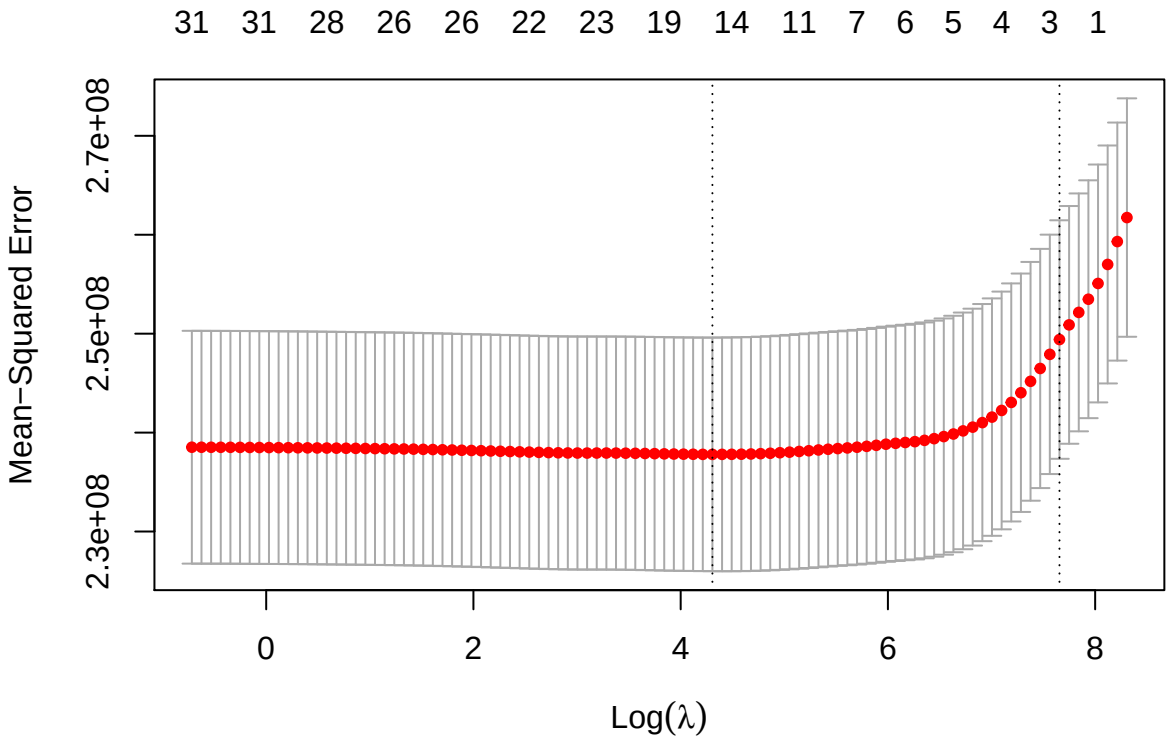
```
load("../../lectures/l15/jtpa/jtpa.RData") #your file location will differ
jtpa = jtpa %>% select(-male,-black,-hispanic)
jtpa = jtpa %>% select(-train,-classroom,-Index,-OJT_JSA,-f2sms)
mod_treat = lm(y~(.-offer)^5,data=jtpa %>% filter(offer==1))
mod_cont = lm(y~(.-offer)^5,data=jtpa %>% filter(offer==0))
```

Perhaps by using a LASSO we can improve our predictions and thus our targeting model.

```
library(glmnet)
x_treat   = model.matrix(y~(.-offer)^5,data=jtpa %>% filter(offer==1))
x_control = model.matrix(y~(.-offer)^5,data=jtpa %>% filter(offer==0))
y_treat   = jtpa %>% filter(offer==1) %>% pull(y)
y_control = jtpa %>% filter(offer==0) %>% pull(y)
lasso_treat   = cv.glmnet(x_treat  ,y_treat)
lasso_control = cv.glmnet(x_control,y_control)
plot(lasso_treat)
```




```
plot(lasso_control)
```



1. Do these plots provide evidence that LASSO will improve our predictions?
2. Do these plots provide evidence of overfit or underfit for our full linear models from HW7?
3. What do these plots tell you about the comparison between the full linear model from the homework and a simple model based on a mean?
4. Are there any other conclusions you can draw from these plots?

Q3: Boosting

The basic idea behind boosting is that we can improve our predictions by building models that give more weight to hard-to-predict observations. Boosting iterates, repeating that basic idea, and averages across all the models it builds.

Suppose we have a dataset of income, which has 3 outlier observations. These are observations which are 10x bigger than other nearby data points.

Sam and Vera have a disagreement about what causes these outliers.

Sam says the reason that there are outliers is that the mean function we want to model spikes up sharply. Other future data points that are *very* close (in covariates X) to the outliers we observe would also experience that sharp spike in the underlying mean – and thus are likely to appear to be outliers.

Vera says the reason there are outliers is that incomes have a distribution which has very fat tails. Some points (in this case 3), are just very very far from the mean, in ways that are totally unpredictable. Other future data points that are *very* close (in covariates X) to the outliers we observe would not experience any change in the mean – and thus are unlikely to appear to be outliers.

1. Would you expect boosting to give better out-of-sample predictions if Sam is right or if Vera is right? In <3 sentences, why?

2. Would you expect boosting to give better in-sample predictions if Sam is right or if Vera is right? In <3 sentences, why?

Q4: Bootstraps, Bagging

Bagging – or bootstrap aggregating, is the process of repeatedly resampling data and estimating a model on the resampled data, and then aggregating (usually by an average) across those models.

We're going to consider the performance improvements of two bagged models using the Semiconductor data (available at https://codowd.com/bigdata/lectures/l5/semiconductors/semiconductor.csv) – one is a bagged logistic regression. One is bagged trees. First, lets split the data into test and training data.

```
semi = read_csv("https://codowd.com/bigdata/lectures/l5/semiconductors/semiconductor.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double()
## )
## i Use `spec()` for the full column specifications.
```

```
#Get a holdout sample
holdout_inds = read_csv("https://codowd.com/bigdata/hw/final/holdouts.csv")$x
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   x = col_double()
## )
```

```
# If there is difficulty getting that file, it was generated with the following code:
# set.seed(123911)
# holdout_inds = which(runif(nrow(semi)) < 0.2) #20% holdout


train = semi[-holdout_inds,]
test  = semi[holdout_inds,]
```

Lets make a single logit model and a single tree to consider their performance.

```
library(rpart)
tree  = rpart(FAIL~.,data=train)
logit =   glm(FAIL~.,data=train,family="binomial")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
tree_preds  = predict(tree ,newdata=test)
logit_preds = predict(logit,newdata=test,type="response")

binom_deviance = function(probs,actual,epsilon=2e-16) {
  probs[probs >= 1] = 1-epsilon
```

```
  probs[probs <= 0 ] = epsilon
  -2*sum(actual*log(probs)+(1-actual)*log(1-probs))
}

tree_dev  = binom_deviance(tree_preds ,test$FAIL)
logit_dev = binom_deviance(logit_preds,test$FAIL)

tree_dev
```

```
## [1] 276.8754
```

```
logit_dev
```

```
## [1] 278.6932
```

Now I'll make some a bagged tree model.

```
#Simple function that makes a single bootstrapped tree.
resampled_trees = function(i,formula=FAIL~.,data=train){
  resampled_inds = sample.int(nrow(data),replace=T)
  rpart(formula,data=data[resampled_inds,])
}
# Run that function 20 times, store the output in a list
bag_tree_submods = lapply(1:20,resampled_trees)
# For each element of that list, make a prediction OOS.
bag_tree_subpreds = sapply(bag_tree_submods,predict,newdata=test)
# Average the predictions to get the ensemble prediction
bag_tree_preds = rowMeans(bag_tree_subpreds)
#Take the ensemble predictions, calculate the binomial deviance.
bag_tree_dev = binom_deviance(bag_tree_preds,test$FAIL)
bag_tree_dev
```

```
## [1] 123.4108
```

Bagging the trees shows a solid performance improvement. Will a logit do the same?

Q4A: Bag the logit.

1. Resample the data, and estimate a logit on the resampled data, 20 times. (hint: don't forget to use `glm` and `family="binomial"` – I suggest tweaking the `resampled_trees` function above)
2. Make OOS predictions for each of the 20 logit models (hint: don't forget to include `type="response"` when predicting logits – if you use sapply that can just be added to the end of the function)
3. Average those predictions to get the ensemble predictions.
4. Find the OOS binomial deviance of the ensemble predictions.

Does the logit performance improve when we bag it? Explain why in <3 sentences.

Q4B: Compare predictions

Plot the OOS bagged logit ensemble predictions against the OOS bagged tree ensemble predictions. Does this plot suggest that you could improve by combining the models rather than choosing between them? Explain why in <3 sentences.

Q4C: Average the Ensembles

Lets consider a naive average of the predictions of our logit ensemble and our tree ensemble. Average them (this should be `(bag_tree_preds+bag_logit_preds)/2`) and find the OOS deviance. What percent improvement is this over the logit ensemble? Explain the source of this improvement in <3 sentences.

Q4D: A 4 tree ensemble

As above, build a bagged model that contains only 4 trees, rather than 20 (you could just take the first four from above if you like).

What is the OOS performance of the four tree bagged model? What fraction of the improvement in deviance from a single tree to 20 trees do we get from using 4 trees? In <3 sentences explain why we would expect to see such a fraction.

Submission

Due Thurs June 3rd at 11:59:59 pm. Submit online through canvas.