API - 209 | Problem Set 11

Prof. Dan Levy

Due on Tuesday, November 15, 2022 at 10:00 am.

GENERAL INSTRUCTIONS

For question 1, the "normal" rules for collaboration in API-209 problem sets apply, i.e., you are encouraged to work in a study group, but must write up your own answers. For question 2 (final exercise), you are asked to work in a group and everyone in the group should submit the exact same answers.

INSTRUCTIONS

To successfully complete this problem set, please follow these steps:

1. **Download this RMarkdown document file into your computer.**
2. **Insert all your answers into this document.** Guidance **here** on how to insert objects such as handwritten work or screenshot images in your answers.
3. **SAVE your work frequently**.
4. To make things easier to visualize in RStudio, you can set the view mode as "Visual" instead of as "Source" in the top left of your screen (just below the Save button).
5. Once your document is complete, please save it as a PDF by clicking the **KNIT** button.
6. Please submit an electronic copy of the PDF (and any separate requested files) to the Canvas course page.

   6.a) If you want to check a PDF version of this problem set before starting to work on it, you can always knit it. In fact, you can knit the document at any point.

   6.b) If you cannot Knit and it's time to submit the problem set, submit the RMarkdown file and make an appointment with a member of the teaching team
7. Remember to consult the R resources from math camp, particularly the HKS R cheat sheet (available **here**, which contains many of the commands needed to answer the questions in this problem set.

IDENTIFICATION

1. Your information

```
Last Name:  Chaturvedi
First Name: Shreya
```

2. Group Members (please list below the classmates you worked with on this problem set):

```
Group members:
```

3. Compliance with Harvard Kennedy School Academic Code: Do you certify that my work in this problem set complies with the Harvard Kennedy School Academic Code[1] (mark with an X below)?

[X] YES [ ] NO

[1] We abide by the Harvard Kennedy School Academic code (available here) for all aspects of the course. In terms of problem sets, unless explicitly written otherwise, the norms are the following: You are free (and encouraged) to discuss problem sets with your classmates. However, you must hand in your own unique written work and code in all cases. Any copy/paste of another's work is plagiarism. In other words, you can work with your classmate(s), sitting side-by-side and going through the problem set question-by-question, but you must each type your own answers and your own code. For more details, please see syllabus.

QUESTION 1 - CAN YOU DO BETTER THAN OSHA? PREDICTING WORKER SAFETY IN THE ERA OF MACHINE LEARNING[2]

In the previous problem set, you read a case and analyzed how OSHA selected the sites that it will inspect given that it does not have resources to inspect all sites. Now, you will be given an OSHA data set on injuries and inspections in work establishments.

Logistical Notes:

- Please have available a copy of your answers to this question and the one from your previous problem set to class on November 15. This will allow you to participate more fully in the discussion of the case in class.
- The R exercises requires many steps, but for the most part you will be asked to summarize your key findings in the answer boxes.
- As in OLS, we have prepared a three-part introduction to LASSO and prediction in R that cover the necessary skills and conceptual understanding to complete this problem. The links are provided along the way, and they are also available together **here**.

Case background

OSHA has hired you (Alex Seguro) to help them improve the way it selects establishments to be inspected. They are under increasing pressure because some policymakers are arguing that OSHA inspections frequently represent an unnecessary cost to businesses. Moreover, they will gain political support if they are able to improve their cost-effectiveness. In this exercise, you will be asked to use various algorithms designed to predict two key outcome variables that OSHA could take into account in deciding which establishments to select for inspections.

The dataset contains annual observations for a sample of establishments that were in OSHA's target list of potential sites to inspect through the SST program in the period 2001-2010. These tend to be establishments that are in higher-risk industries that merit OSHA's close attention. Because of resource constraints, OSHA cannot inspect all sites on such list. At the moment, the process that OSHA uses to select from this list is akin to a simple random selection. Your job is to see if you can use prediction models to improve on the procedure that OSHA uses. The key features of the question are summarized in the box below.

Outline of this Question

Outcomes: OSHA wants to predict two outcomes:

- `injury_rate`, a continuous variable, measured as "workplace's annual number of injuries that prompted at least one day away from work (DAFW) per 100 full time equivalent (FTE) employees", and
- `high_injury_rate`, a binary variable, which is 1 if `injury_rate` is larger than 0.3 and 0 otherwise.

Predictors: All variables are described within the case (Data Appendix). Each observation in the data set is a workplace at risk for Site-Specific Targeting (SST) inspection. Some variables represent outcomes in the prediction year, some represent characteristics that are available before the prediction year, and some represent characteristics that are fixed over time (industry, region of the country, etc.).

Predictive Models: We will ask you to evaluate at least three models: A simple OLS considering only a handful of predictors, a "kitchen-sink", exhaustive OLS which considers all the predictors available, and a LASSO regression which also considers all the predictors but runs a penalized regression, instead of ordinary least squares.

Optional Questions: If you have time, we encourage you to evaluate several other well-known machine learning models: A ridge regression, post-LASSO with data-driven tuning parameters, and a regression tree model, along with a null model. If you do any of these, please email me your problem set before discussing the case on Tuesday.

1. To prepare, learn the basics of pre-processing for machine learning packages in R with this **screencast** we have made for this problem set. This screencast roughly covers parts (1) through (3).

[2] I am very grateful to former MPA/IDs Astrid Camille Pineda, José Ramón Morales, and Shiro Kuriwaki for their help in designing and writing this problem set question.

Start by downloading the dataset called `osha.csv`, reading it in, and examining its dimensions. We will refer to this original dataset as `osha` in the code snippets below.

We have pre-processed the dataset to make it a bit easier to use:

- We have already transformed factor (categorical) variables into multiple dummy variables.
- We re-scaled all variables that are not binary so that each of them has a sample mean of 0 and a standard deviation of 1[3].

One of the benefits of an object-oriented programming language like R is that you can build up complex processes from its component pieces. Start by preparing the formula objects that will represent the regression specification for two OLS models we will estimate. We have provided code for you for this part, so all you need to do is read and use the code provided.

A. A defensible choice for the short OLS is to start from indicators that the site has gone through inspections or has received some complaints in the past. Store a formula object for this regression with `injury_rate` as the outcome:

```
osha <- read_csv("osha.csv")

f_short_c <- injury_rate ~ has_tmin1_odi + any_insp_prior +
  any_complaint_tmin13 + num_nonfat_comp_insp_cy_tc99mm1 +
  initial_pen_cy_mzmm1 + ln_initial_pen_cy_mzmm1 +
  dafw_analysis_rec_tc99mm1
```

B. The kitchen-sink regression should include all the available variables in the data EXCEPT for the following: `sst_year` (the year), `estab_id_duns` (the workplace identifier), injuries, `injury_rate`, and `high_injury_rate` (the three outcome measures). Create this formula object, again with the outcome injury_rate.

This formula will include 178 terms, which is too many to type out by hand. The trick we'll use here is to extract the terms as a vector and concatenate them by the `str_c()`. You can use the code below to do so, and the description following the code explains what the code is doing.

```
rm_vars <- c("sst_year",
             "estab_id_duns",
             "injuries",
             "injury_rate",
             "high_injury_rate")

long_vars  <- setdiff(colnames(osha), rm_vars)

f_long_rhs <- str_c(long_vars, collapse = " + ")

f_long_c   <- as.formula(str_c("injury_rate ~ ", f_long_rhs))
```

- rm_cars manually creates a character vector composed of the six variables to remove.
- long_vars creates a character vector composed of the variables you wish to put as predictors. `colnames()` gives a vector of the column names of a dataframe, and `setdiff()` takes the first argument and removes the elements named in the second argument.
- f_long_rhs uses the long_vars vector with the `str_c()` function with the argument collapse = " + ", which will concatenate (collapse) all the elements of a vector with the + character as glue. This is your RHS (right-hand side).
- Write out the LHS and tilde, and concatenate that to the RHS.
- Coerce the string into a formula object f_long_c with `as.formula()`.

2. Split the data 70-30 into a training and test data, respectively. Splitting must be random. Here, try using the function `rsample::initial_split`, and please use the seed 02138 as shown in the screencast[4].

[3] Here, as in the first screencast here, `.` stands for "everything else".

[4] It is called this way because binary outcomes can be modeled as a Binomial distribution. `family = "gaussian"` would corresponding to continuous variables. In `binomial`, the logit link function is used as a default.

Note: R's double colon (::), a notation we will be using repeatedly, is a standard way to specify not just the function (on the RHS) but also the package that comes from (on the LHS). It is a useful shorthand that also runs as R code.

```
# insert your code here

set.seed(02138)
split <- initial_split(osha, prop = 0.7)
osha_train <- training(split)
osha_test <- testing(split)
```

3. Our third model is LASSO, implemented in the `glmnet` package written by Friedman, Hastie, and Tibshirani et al. (the authors of The Elements of Statistical Learning, a standard Machine Learning text).

`glmnet` requires users to input a numeric X matrix and a y vector directly, instead of specifying a formula as in `lm`. Create these objects, again with the code provided:

```
y_c_train <- osha_train$injury_rate
x_train   <- as.matrix(select(osha_train, !!!long_vars))
```

The rationale for the code is:

- A concise tidyverse way to subset a dataframe to a set of columns is to use the familiar select function but instead of naming each variable, enter the vector of variable names preceded with three exclamation marks[5]
- Then coerce the dataframe into a matrix by `as.matrix`.
- Create the outcome vector by either pull, $, or double square brackets. We can name this `y_c_train` to denote that this is a continuous outcome.

4. Estimating predictive models of the training set

For the next few problems, watch the second screencast to learn about the fitting and interpreting a LASSO regression.

Predict the continuous variable `injury_rate` in the training set with the following models. There are three required and four optional models (described in optional question A1).

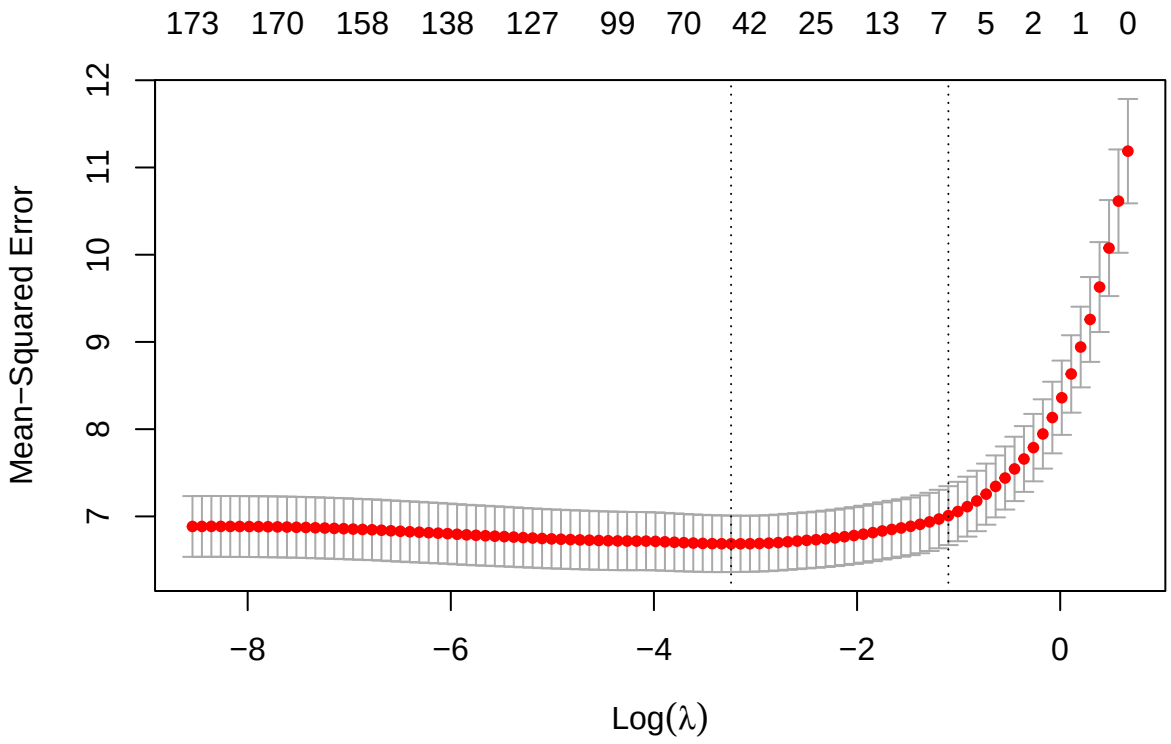
i. The simple model using the variables in (2) A estimated by OLS in `lm`, which we'll call the short model.
ii. The "kitchen-sink" model using the variables in (2) B computed by OLS in `lm`, which we'll refer to the long model.
iii. A LASSO model (lasso) that considers the same variables as the long model, where the penalty parameter is chosen by cross-validation.

Store each of these into separate objects.

```
# insert your code here

short_model <- lm(f_short_c, osha_train)
long_model <- lm(f_long_c, osha_train)
lasso_model <- cv.glmnet(x = x_train, y = y_c_train)
plot(lasso_model)
```

[5] In a nutshell, the econometric way to think about a binary outcome model is to assume that there is a latent, continuous, and unbounded utility that can be modeled as in OLS. We then say this gets passed through a particular function — $\frac{\exp(u)}{1+\exp(u)}$ in the logit case — that rescales the continuous utility into a monotonically increasing function conveniently bounded between 0 and 1. This is then interpreted as the rate parameter in a Bernoulli distribution, which generates the outcome.



Note: This is a good place to consider trying optional question A2.

5. Predicting a continuous outcome

Watch the **third screencast** in the LASSO series to learn about generating predicted values with regression objects.

Generate predicted values from the three models you estimated, again on the training dataset. As the second screencast discussed, the LASSO model you estimated stores multiple models for each value of the penalty parameter λ, so please pick the LASSO coefficients at the value of λ that `cv.glmnet` has determine to achieve the lowest Mean Squared Error. Store each set of predictions as additional columns on the training dataframe, as shown in the screencast.

```
# insert your code here

osha_train <- osha_train %>%
  mutate(
    pred_short = predict(short_model),
    pred_long = predict(long_model),
    pred_lasso = as.vector(predict(lasso_model, newx = x_train, s = "lambda.min"))
  )
```

6. Predictive performance for continuous outcome variables

A. Summarize the predictive performance for **each model** (Simple, Kitchen Sink and LASSO) on **the training set** by computing the Mean Square Error (MSE) between the predicted value and the observed outcome, and display these values below.

MSE for the length-n observed outcome vector y and the corresponding predicted values $\hat{y}$ is defined as:

$$\text{MSE}(y, \hat{y}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

```r
# insert your code here
options(digits = 3)
osha_train_mse <- osha_train %>%
  mutate(
    squared_error_short = (injury_rate - pred_short)^2,
    squared_error_long = (injury_rate - pred_long)^2,
    squared_error_lasso = (injury_rate - pred_lasso)^2,
  ) %>%
  summarize(
    mse_short = mean(squared_error_short),
    mse_long = mean(squared_error_long),
    mse_lasso = mean(squared_error_lasso)
    )
gt(osha_train_mse)
```

mse_short	mse_long	mse_lasso
7.41	6.25	6.48

B. Now repeat the procedures in 5 and 6A but for the **test (holdout) set**. Note that you will need to create the equivalent of `X_train` but for the test set. Please report below your MSE for **each model** on the **test (holdout) set**. Remember you will be using the parameters you estimated using observations in the training set to make predictions about the observations in the test set. We provide the following table as a guidance on the results you are expected to produce after answering questions 6A and 6B. **Please note that you do not need to report your results in the exact format of this table; it is displayed below to help ensure that you know which numbers you are being asked to calculate.**

Table 1 – Predictive Performance Indicators for Injury Rates

Model	Training Set prediction MSE	Test Set prediction MSE
Simple		
Kitchen Sink		
LASSO		

Note: Simple = 5 predictors, Kitchen Sink = 178 predictors

* Instructions: Please round your numbers reasonably – e.g., 2 significant digits

```r
# insert your code here
y_c_test <- osha_test$injury_rate
x_test   <- as.matrix(select(osha_test, !!!long_vars))

osha_test <- osha_test %>%
  mutate(
    pred_short = predict(short_model, newdata = osha_test),
    pred_long = predict(long_model, newdata = osha_test),
    pred_lasso = as.vector(predict(lasso_model, newx = x_test, s = "lambda.min"))
  )

osha_test_mse <- osha_test %>%
  mutate(
    squared_error_short = (injury_rate - pred_short)^2,
    squared_error_long = (injury_rate - pred_long)^2,
    squared_error_lasso = (injury_rate - pred_lasso)^2,
  ) %>%
  summarize(
```

```
    mse_short = mean(squared_error_short),
    mse_long = mean(squared_error_long),
    mse_lasso = mean(squared_error_lasso)
    )
gt(osha_test_mse)
```

mse_short	mse_long	mse_lasso
7.42	6.81	6.69

C. What are 2-3 key conclusions you draw from your results above (i.e., MSE for the 3 different models) regarding your ability to predict injury rates?

Note: This is a good place to consider trying optional question A3.

Please enter your answers here

XX

---

7. Predicting a binary outcome (i.e. classification)

In the previous section, you predicted a continuous variable (`injury_rate`). The next set of questions are about predicting a binary variable (`high_injury_rate`). While the logic of prediction is the same, we will need to adapt our models and our metric for assessing performance.

Consideration 1. Adapt the outcome vectors and the formulas by swapping `injury_rate` with `high_injury_rate`. An easy way to change only a part of the formula is to use the `update()` function. For example, `update(y ~ x1 + x2 + x3, z ~ .)` will produce `z ~ x1 + x2 + x3`.[6]

```
y_b_train <- osha_train$high_injury_rate
f_short_b <- update(f_short_c, high_injury_rate ~ .)
f_long_b  <- update(f_long_c, high_injury_rate ~ .)
```

Consideration 2. Now you can use the formulas to fit new models – the binary outcome equivalents of part (4). You will want to replace the models you used above to its binary equivalents.

- Change the OLS regressions into a logit regression. Switch `lm` to `glm` (for generalized linear model) with the argument `family = "binomial"`[7]
- For `cv.glmnet`, keep the function but add to it an option `family = "binomial"`.

```
fit_null_b  <- glm(high_injury_rate ~ 1, osha_train, family = "binomial")
fit_short_b <- glm(f_short_b, osha_train, family = "binomial")
fit_long_b  <- glm(f_long_b, osha_train, family = "binomial")
fit_lasso_b <- cv.glmnet(x = x_train, y = y_b_train, alpha = 1, family  = "binomial")
```

Consideration 3. You can generate predicted values from your models with the same `predict` function. There is, however, one critical difference in predictions from a binary outcome model: the predicted values are probabilities even though the observed outcomes are binary. Prediction therefore happens in three steps:

[6] Here, as in the first screencast here, `.` stands for "everything else".

[7] It is called this way because binary outcomes can be modeled as a Binomial distribution. `family = "gaussian"` would corresponding to continuous variables. In `binomial`, the logit link function is used as a default.

- When using `predict`, set the argument `type = "response"`. This prompts `predict` to output the predictions on the probability scale, instead of R's default to give them on a latent utility scale.[8]
- You might want to verify that your predicted values are between 0 and 1 by graphing them in a histogram.
- **Create a binary variable (with the tag `pred_hi`)** that takes the value of 1 for those establishments that are in the top 30% (i.e., the 70% quantile and above) in terms of your predicted probability of having a high injury rate, and a value of 0 otherwise. If your predicted probability from the short regression is called `pred_short`, in a dplyr context "if-else" rule is concisely given by `pred_hi_short = pred_short > quantile(pred_short, 0.70)`. The establishments with `pred_hi_short = 1` represent the ones you would recommend OSHA to select for inspection.

Remark. As you might have noticed, the choice of the 70th percentile as our cutoff is arbitrary. We can evaluate the predictive performance for any cutoff between 0 and 100%. The ROC curve (left as an optional question below) is a way to visualize all those possibilities.

```
# insert your code here
osha_train <- osha_train %>%
  mutate(
    pred_short_b = predict(fit_short_b, newdata = osha_train, type="response"),
    pred_long_b = predict(fit_long_b, newdata = osha_train, type="response"),
    pred_lasso_b = as.vector(predict(fit_lasso_b, newx = x_train, s = "lambda.min", type="response")),
    pred_hi_short = pred_short_b > quantile(pred_short_b, 0.70),
    pred_hi_long = pred_long_b > quantile(pred_long_b, 0.70),
    pred_hi_lasso = pred_lasso_b > quantile(pred_lasso_b, 0.70)
  )
```

Consideration 4. Although it is technically possible to compute a MSE with the predicted binary variable and the observed binary variable, it does not make much sense to do so because the outcome and model output is essentially on different scales. Therefore, we ask you to report the precision and recall (see formulas from class) instead for each of the 3 models for the training sample. This can be done either via `summarize` or by generating cross-tabs that show the count of true positives, true negatives, false positives, and false negatives.

```
# insert your code here
osha_train_table <- osha_train %>%
  summarize(
    precision_short = sum(high_injury_rate*pred_hi_short)/sum(pred_hi_short),
    precision_long  = sum(high_injury_rate*pred_hi_long)/sum(pred_hi_long),
    precision_lasso = sum(high_injury_rate*pred_hi_lasso)/sum(pred_hi_lasso),
    recall_short = sum(high_injury_rate*pred_hi_short)/sum(high_injury_rate),
    recall_long  = sum(high_injury_rate*pred_hi_long)/sum(high_injury_rate),
    recall_lasso = sum(high_injury_rate*pred_hi_lasso)/sum(high_injury_rate)
  )
gt(osha_train_table)
```

precision_short	precision_long	precision_lasso	recall_short	recall_long	recall_lasso
0.741	0.803	0.799	0.513	0.556	0.552

Consideration 5. As above, you will need to repeat the prediction and assessment for the **test set** and report the precision and recall for each of the 3 models for the test set.

```
# insert your code here
osha_test <- osha_test %>%
  mutate(
    pred_short_b = predict(fit_short_b, newdata = osha_test, type="response"),
    pred_long_b = predict(fit_long_b, newdata = osha_test, type="response"),
    pred_lasso_b = as.vector(predict(fit_lasso_b, newx = x_test, s = "lambda.min", type="response")),
    pred_hi_short = pred_short_b > quantile(pred_short_b, 0.70),
```

[8] In a nutshell, the econometric way to think about a binary outcome model is to assume that there is a latent, continuous, and unbounded utility that can be modeled as in OLS. We then say this gets passed through a particular function — $\frac{\exp(u)}{1+\exp(u)}$ in the logit case — that rescales the continuous utility into a monotonically increasing function conveniently bounded between 0 and 1. This is then interpreted as the rate parameter in a Bernoulli distribution, which generates the outcome.

```
    pred_hi_long = pred_long_b > quantile(pred_long_b, 0.70),
    pred_hi_lasso = pred_lasso_b > quantile(pred_lasso_b, 0.70)
  )

osha_test_table <- osha_test %>%
  summarize(
    precision_short = sum(high_injury_rate*pred_hi_short)/sum(pred_hi_short),
    precision_long  = sum(high_injury_rate*pred_hi_long)/sum(pred_hi_long),
    precision_lasso = sum(high_injury_rate*pred_hi_lasso)/sum(pred_hi_lasso),
    recall_short = sum(high_injury_rate*pred_hi_short)/sum(high_injury_rate),
    recall_long  = sum(high_injury_rate*pred_hi_long)/sum(high_injury_rate),
    recall_lasso = sum(high_injury_rate*pred_hi_lasso)/sum(high_injury_rate)
  )

gt(osha_test_table)
```

precision_short	precision_long	precision_lasso	recall_short	recall_long	recall_lasso
0.728	0.763	0.766	0.493	0.517	0.519

We provide the following table as a guidance on the results you are expected to produce after answering sub-questions above of question 7. **Please note that you do not need to report your results in the exact format of this table; it is displayed below to help ensure that you know which numbers you are being asked to calculate.**

Description of what "Precision" and "Recall" represent in the context of this case:

Table 2 – Predictive Performance Indicators for a Flag for High Injury Rates

	Training Set		Test Set	
Model	**Precision**	**Recall**	**Precision**	**Recall**
Simple				
Kitchen Sink				
LASSO				

* Instructions: Please round your numbers reasonably – e.g., 2 significant digits

8. Based on the above results, OSHA asks you to indicate which of your 3 prediction algorithms (or models) you would recommend using to select the sites they should inspect and why.

Please enter your answers here

XX

---

9. Given OSHA's current approach of choosing randomly to inspect 30% of sites, compute the precision and recall of their current approach. Note: You don't need to use R or the data set to answer this question. Goal is that you apply the definition of precision and recall to OSHA's current algorithm.

Please enter your answers here

OSHA's Precision: XX

OSHA's Recall: XX

---

10. Are your precision and recall rates better than those for OSHA's current approach (previous question)?

Please enter your answers here

XX

---

11. OSHA would need to overcome several operational and political obstacles in order to change their algorithm for targeting which sites to inspect. They ask you to explain to them how much better the algorithm you chose performs relative to their status quo operation. How would you respond?

Please enter your answers here

XX

---

12. Please enter the following information in this link for our discussion on Tuesday:

https://forms.gle/xJz3hSvrHmu2bTJa7

Congratulations! Through this exercise, you have now learned the basics of how to use machine learning to do prediction with real world data in a real world setting. Do not be intimidated anymore by the terms data science, machine learning, predictive analytics, etc. Some of the methods are for sure more sophisticated than the ones you used here, but you now have a basic foundation to understand what they do. There is of course more to learn. If you are interested in doing so, please come and speak with any member of our teaching team!

OPTIONAL

The preceding questions allowed you to do the basics of machine learning. Congratulations! If you are interested in expanding your knowledge or skills in this area further, you might consider doing some of the optional questions below.

A1. Optional Models

In addition to the three required, estimate the following.

4. A **null** model as a benchmark, computed with only the intercept (no covariates) in OLS. The formula syntax for this is `injury_rate ~ 1` where `1` represents an intercept that is implicitly assumed when there are covariates.
5. A **ridge** model similar to the `glmnet`'s LASSO. In fact, this is simple as setting the `alpha` argument to `0` instead of `1` in `cv.glmnet`.
6. Another version of the LASSO (`hdm::rlasso`) where the penalty is chosen by the data. We will call this **post**-LASSO. The name comes from the fact that variables selected by this LASSO are later (post) run as OLS.
7. A regression tree, which we'll call **forest**. The function `randomForest::randomForest` is a standard package.
8. A model of your choosing that you think will achieve the best out of sample prediction accuracy. (If you do this option and achieve better predictive accuracy than all seven of the above, please email us with your metric and specification!)

```
# In-sample
```

```
# Out-sample


# Compare side by side
```

A2. Interpreting LASSO coefficients

Let's summarize the results of the LASSO in three parts to understand what is happening.

A. Check the change in MSE as the values of the penalty term λ changes. What value of λ gives the smallest MSE?

B. Examine the coefficient estimates when λ is the value given in part A. How many variables (columns) did we enter into either regression? Compare the variable selection with the long regression: How many coefficients have been zeroed out (are exactly zero) in each regression?

C. Continuing the comparison with the long model, fill in the table that shows the five largest non-zero LASSO coefficients, with the coefficient estimates from the long model side-by-side. For the OLS, also note the p-value associated with the five coefficients.

Finally, interpret your table by touching on the following points:

- LASSO not only selects variables, but tends to shrink them towards 0 to make up for the dropped variables. Do you see that here?
- Are any variables that the long regression would deem as insignificant at conventional p-value levels selected as significant in LASSO?

A3. Bias-Variance

Note, *the instructions and formulas for this optional problem were not entirely correct. In general the MSE can be decomposed into the sum of bias^2 and variance. Bias however is difficult to estimate in practice because for each model we only obtain a single prediction to compare to the observed value.*

In contrast to the original instructions, the below table estimates the squared bias as

$Bias^2 = MSE - Var(\hat{Y})$

The MSE is a useful metric because it can be decomposed into a bias component and a variance component. For any value of the empirical MSE comparing the observed ("true") data y and the model estimate $\hat{y}$, we can decompose

$\text{MSE}(y, \hat{y}) = \text{Bias}(y, \hat{y})^2 + \text{Var}(\hat{y}) + \text{Var}(\epsilon)$

where $\text{Bias}(y, \hat{y})$ can be estimated as the average of the absolute deviation between predicted and observed values for each observation, $\text{Var}(\hat{y})$ can be estimated as the variance of the predicted values (the noisiness of the predictions), and the $\text{Var}(\epsilon)$ is what Tibishirani and Hastie call "irreducible error".

Therefore, the MSE tries to reconcile the fundamental bias-variance trade-off in any kind of statistics.

Investigate a claim that was made in lecture: that OLS is generally unbiased, whereas ML methods are lower variance. Do this by making side-by-side comparisons of the MSE, the Variance, and the Bias of each model, all in the test set. As the formula above indicates, the sum of the variance and bias squared should be strictly less than the observed MSE.

What patterns stand out?

A4. Binary Outcomes

- For `rlasso`, switch to the function `rlassologit`.
- For `randomForest`, you will need to signal that it is a classification task by using transforming the binary outcome variable into a factor in the left-hand-side of the formula.

A5. Response-Operator Curves (AUC) and the Area under the Curve (AUC)

For this part, you may only consider the LASSO regression on the test set.

A. Instead of considering the top 30% of the observations as "predicted high injury", what would be the precision and accuracy if you had considered only the top 1% of the observations as "predicted high injury"?

B. Plot the Response Operator curve (ROC) of the LASSO model. Then compute the out of sample Area under the Curve (AUC) from all models you estimated. The easiest way is to use `pROC::auc`, which can take a formula object as its first argument where the LHS is the observed binary outcome and the RHS is the predicted probability from the model, and the second argument is the dataframe.

Verify that:

- You can identify the Sensitivity and the Specificity values you computed with the two cutoffs as coordinates on the plot.
- Visually, the AUC appears to correspond to the area under the ROC curve.

Which model(s) achieve the highest AUC?

The graph shows the LASSO model (red), the long model (black), and the null model (gray).

A6. Email

If you did any of the optional extensions above, please email me a copy of your problem set to see if we can incorporate some of your work into the class plan. The earlier you can email it to me the better.

QUESTION 2 - FINAL EXERCISE

This part of the problem set is designed to be completed with your final exercise team. As opposed to other problem set questions where you are asked to write your answers in your own words, all team members can submit identical answers for this question. But please submit answers individually (as part of the problem set you submit) this time to facilitate the grading.

Note: The goal of this question is to help you advance in the final exercise, so you increase your chances of producing a final product you are proud of. Don't feel too constrained by the specific prompts you see below. You should try to answer each of the prompts but your team should decide how much time it is worth to spend at this time on each of the items below. Ultimate goal is to nudge you in the direction of making progress.

Link to final exercise is here.

OPTION 1 - Mongolia (Macro)

1. Review the list of final exercise items from the previous problem set and complete those items that you were not able to complete (skip this step if you did them all).
2. Finalize your analysis plan. This implies translating each of the points listed in the assignment into tasks that can be operationalized, i.e. list the tabulations, cross-tabulations, graphs, and regressions you envision doing for the final memo. You don't have to complete all tests/analyses, but you should have completed some and have in mind the key things you have yet to analyze. We recommend going over Miguel Santos' book before finalizing the tests you're going to run (the 4 types of tests outlined in the project description). Remember that this is a statistics class, and we're looking for statistically significant results, not only trends.
3. If you're struggling to find statistically significant results in the microdata for Mongolia, how about global evidence in labor force productivity in which Mongolia is an outlier? Does that provide a clue?
4. What are the key findings (using bullet points and/or visual aids) that you plan to highlight in the main body of the memo?
5. Produce a draft outline of your technical appendix. This need not be the final version, but it should contain sections designed to explain the technical aspects underpinning your work and the limitations of your analysis. It should also contain templates of tables you plan to put in this appendix.
6. The most effective messages are relayed via good visualizations. Start thinking about what kind of graphs you plan to incorporate into your memo and presentation.
7. Decide how you plan to organize the rest of the work. Who will do what? Establish some deadlines.

Please enter your answers here

OPTION 2 - Health in Brazil (Micro)

1. Review the list of final exercise items from the previous problem set and complete those items that you were not able to complete (skip this step if you did them all).
2. Finalize your analysis plan. This implies translating each of the points listed in the assignment into tasks that can be operationalized, i.e. list the tabulations, cross-tabulations, graphs, and regressions you envision doing for the final memo. You don't have to complete all tests/analyses, but you should have completed some and have in mind the key things you have yet to analyze.
3. If you're struggling to deal with the complexity of this final exercise keep calm, breath and keep it simple. It is better to start simple and then build up complexity than to get stuck.
4. What are the key findings (using bullet points and/or visual aids) that you plan to highlight in the main body of the memo?
5. Produce a draft outline of your technical appendix. This need not be the final version, but it should contain sections designed to explain the technical aspects underpinning your work and the limitations of your analysis. It should also contain templates of tables you plan to put in this appendix.
6. The most effective messages are relayed via good visualizations. Start thinking about what kind of graphs you plan to incorporate into your memo and presentation.
7. Decide how you plan to organize the rest of the work. Who will do what? Establish some deadlines.

Please enter your answers here
