Math 342W / 642 / RM742 Spring 2025
Midterm Examination Two

Professor Adam Kapelner

May 15, 2025

Full Name ______________________________

Code of Academic Integrity

Since the college is an academic community, its fundamental purpose is the pursuit of knowledge. Essential to the success of this educational mission is a commitment to the principles of academic integrity. Every member of the college community is responsible for upholding the highest standards of honesty at all times. Students, as members of the community, are also responsible for adhering to the principles and spirit of the following Code of Academic Integrity.

Activities that have the effect or intention of interfering with education, pursuit of knowledge, or fair evaluation of a student's performance are prohibited. Examples of such activities include but are not limited to the following definitions:

Cheating Using or attempting to use unauthorized assistance, material, or study aids in examinations or other academic work or preventing, or attempting to prevent, another from using authorized assistance, material, or study aids. Example: using an unauthorized cheat sheet in a quiz or exam, altering a graded exam and resubmitting it for a better grade, etc.

I acknowledge and agree to uphold this Code of Academic Integrity.

______________________________ ______________
signature date

Instructions

This exam is 110 minutes (variable time per question) and closed-book. You are allowed **two** pages (front and back) of a "cheat sheet", blank scrap paper (provided by the proctor) and a graphing calculator (which is not your smartphone). Please read the questions carefully. Within each problem, I recommend considering the questions that are easy first and then circling back to evaluate the harder ones. No food is allowed, only drinks.

Problem 1 This question is about two datasets from the `nycflights13` package called `planes` and `flights`. Below are `skimr::skim` reports of both data frames:

```
> skim(flights)
-- Data Summary ------------------------
                           Values
Name                       flights
Number of rows             336776
Number of columns          19
_______________________
Column type frequency:
  factor                   4
  numeric                  14
________________________
Group variables            None

-- Variable type: factor -------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 carrier               0         1     FALSE         16 UA: 58665, B6: 54635
2 tailnum            2512         0.993 FALSE       4043 N72: 575, N72: 513
3 origin                0         1     FALSE          3 EWR: 120835, JFK: 111279
4 dest                  0         1     FALSE        105 ORD: 17283, ATL: 17215

-- Variable type: numeric ---------------------------------------------
   skim_variable  n_missing complete_rate    mean      sd    p0  p25  p50  p75 p100
 1 year                   0         1     2013       0     2013 2013 2013 2013 2013
 2 month                  0         1        6.55    3.41     1    4    7   10   12
 3 day                    0         1       15.7     8.77     1    8   16   23   31
 4 dep_time            8255         0.975 1349.    488.       1  907 1401 1744 2400
 5 sched_dep_time         0         1     1344.    467.     106  906 1359 1729 2359
 6 dep_delay           8255         0.975   12.6    40.2    -43   -5   -2   11 1301
 7 arr_time            8713         0.974 1502.    533.       1 1104 1535 1940 2400
 8 sched_arr_time         0         1     1536.    497.       1 1124 1556 1945 2359
 9 arr_delay           9430         0.972    6.90   44.6    -86  -17   -5   14 1272
10 flight                 0         1     1972.   1632.       1  553 1496 3465 8500
11 air_time            9430         0.972  151.     93.7     20   82  129  192  695
12 distance               0         1     1040.    733.      17  502  872 1389 4983
13 hour                   0         1       13.2     4.66     1    9   13   17   23
14 minute                 0         1       26.2    19.3      0    8   29   44   59
```

```
> skim(planes)
-- Data Summary ------------------------
                           Values
Name                       planes
Number of rows             3322
Number of columns          9
_______________________
Column type frequency:
  factor                   5
  numeric                  4
________________________
Group variables            None

-- Variable type: factor -------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 tailnum               0             1 FALSE       3322 N10: 1, N10: 1
2 type                  0             1 FALSE          3 Fix: 3292, Fix: 25, Rot: 5
3 manufacturer          0             1 FALSE         35 BOE: 1630, AIR: 400, BOM: 368
4 model                 0             1 FALSE        127 737: 361, A32: 256
5 engine                0             1 FALSE          6 Tur: 2750, Tur: 535, Rec: 28

-- Variable type: numeric ---------------------------------------------
  skim_variable n_missing complete_rate    mean      sd   p0  p25  p50  p75 p100
1 year                 70       0.979   2000.     7.19  1956 1997 2001 2005 2013
2 engines               0       1          2.00   0.118    1    2    2    2    4
3 seats                 0       1        154.    73.7      2  140  149  182  450
4 speed              3299       0.00692  237.   150.      90  108.  162  432  432
```

(a) [3 pt / 3 pts] The missing data mechanism for the `flights` data frame is most likely...
Circle one: MCAR / MAR / NMAR

(b) [2 pt / 5 pts] The missing data mechanism for the `planes` data frame is most likely...
Circle one: MCAR / not MCAR

(c) [2 pt / 7 pts] In the `planes` data frame, the likely recommended procedure to do with the `speed` variable is to ... Circle one:
drop it / impute it using the value $\bar{x}_j$ / impute it using `missForest`

(d) [3 pt / 10 pts] In the `flights` data frame, assume we drop all rows with `dep_time` missing. After doing so, the `arr_time` variable has very little missingness. The likely recommended procedure to do with the `arr_time` variable then is to ... Circle one:
drop it / impute it using the value $\bar{x}_j$ / impute it using `missForest`

Despite your answers above, in the `flights` data frame, we dropped all rows with `dep_time` missing and then imputed all other values using `missForest`. And in the `planes` data frame, we dropped `speed` and then used listwise deletion. Here is the updated `skimr::skim` reports for the imputed data frames:

```
> skim(flights)
-- Data Summary ------------------------
                           Values
Name                       flights
Number of rows             328521
Number of columns          18
-----------------------
Column type frequency:
  factor                   4
  numeric                  14
------------------------
Group variables            None

-- Variable type: factor ---------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 carrier               0             1 FALSE         16 UA: 57979, B6: 54169
2 tailnum               0             1 FALSE       4037 N72: 546, N72: 487
3 origin                0             1 FALSE          3 EWR: 117596, JFK: 109416
4 dest                  0             1 FALSE        104 ATL: 16898, ORD: 16642

-- Variable type: numeric --------------------------------------------
   skim_variable  n_missing complete_rate    mean      sd   p0  p25  p50  p75 p100
 1 year                   0             1 2013       0    2013 2013 2013 2013 2013
 2 month                  0             1    6.56    3.41    1    4    7   10   12
 3 day                    0             1   15.7     8.78    1    8   16   23   31
 4 dep_time               0             1 1349.    488.      1  907 1401 1744 2400
 5 sched_dep_time         0             1 1341.    467.    500  905 1355 1729 2359
 6 dep_delay              0             1   12.6    40.2   -43   -5   -2   11 1301
 7 arr_time               0             1 1502.    533.      1 1105 1535 1940 2400
 8 sched_arr_time         0             1 1533.    498.      1 1122 1555 1944 2359
 9 arr_delay              0             1    6.90   44.6   -86  -17   -5   14 1272
10 flight                 0             1 1945.   1622.      1  544 1471 3416 8500
11 air_time               0             1  151.     93.5    20   82  130  191  695
12 distance               0             1 1049.    736.     80  509  888 1389 4983
13 hour                   0             1   13.1     4.66    5    9   13   17   23
14 minute                 0             1   26.2    19.3     0    8   29   44   59

> skim(planes)
-- Data Summary ------------------------
                           Values
Name                       planes
Number of rows             3252
Number of columns          8
-----------------------
Column type frequency:
  factor                   5
  numeric                  3
```

```
------------------------
Group variables           None

-- Variable type: factor ----------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 tailnum               0             1 FALSE       3252 N10: 1, N10: 1, N10: 1
2 type                  0             1 FALSE          3 Fix: 3230, Fix: 17, Rot: 5
3 manufacturer          0             1 FALSE         28 BOE: 1603, AIR: 390
4 model                 0             1 FALSE        121 737: 355, A32: 253
5 engine                0             1 FALSE          6 Tur: 2697, Tur: 526

-- Variable type: numeric ---------------------------------------------
  skim_variable n_missing complete_rate    mean     sd    p0  p25  p50  p75 p100
1 year                  0             1 2000.    7.19  1956 1997 2001 2005 2013
2 engines               0             1    2.00  0.102    1    2    2    2    4
3 seats                 0             1  155.   73.3      2  140  149  182  450
```

The `tailnum` variable is a unique serial number for each airplane. Consider left-joining the `flights` data frame to the `planes` data frame to create a data frame called `flights_with_plane_information`.

(e) [3 pt / 13 pts] How many rows will `flights_with_plane_information` have? If the answer cannot be determined, write "cannot be determined".

328,521 (or, if you read the question differently, cannot be determined)

(f) [3 pt / 16 pts] Consider the following calculation:

```
> setdiff(planes$tailnum, flights$tailnum)
[1] "N347SW" "N728SK" "N768SK" "N862DA" "N865DA" "N939DN"
```

If we left-join the `planes` data frame to the `flights` data frame, how many rows will the joined data frame have? If the answer cannot be determined, write "cannot be determined".

cannot be determined (or, if you read the question differently, 328,521)

(g) [3 pt / 19 pts] Assume we want to plot the density distribution of `arr_delay` by plane `manufacturer` and automatically generate a legend in `ggplot`'s `geom_density`. Provide four rows of an example input data frame for this operation below. We need to generate a few sample rows from the *long* data frame derived from `flights_with_plane_information` with metric variables `arr_delay` and `manufacturer`:

variable	value
manufacturer	BOE
arr_delay	-17
manufacturer	AIR
arr_delay	14

We now wish to build a model to predict $\boldsymbol{y}$ = `arr_delay`, the numeric amount of time (in min.) that a flight is delayed (positive values of y indicate the flight arrived late and negative values indicate the flight arrived early). We can only use features that we know in advance of the flight and we drop obviously useless features and obviously duplicative information. Below is the subset used with comments about what the variables mean compiled into matrix $\boldsymbol{X}$ and a `skimr::skim` report:

```
> n = nrow(flights_with_plane_information)
> y = flights-with-plane-information$arr-delay
> X = flights-with-plane-information[, c(
  "sched_dep_time",  #in 24hr format
  "sched_arr_time",  #in 24hr format
  "carrier",         #the airline
  "origin",          #the code of airport
  "dest",            #the code of airport
  "distance",        #in miles
  "model",           #the plane's model
  "year.y"           #the year the plane was constructed
)]

> skim(X)
-- Data Summary ------------------------
                           Values
Name                       X
Number of rows             274796
Number of columns          8
-----------------------
Column type frequency:
  factor                   4
  numeric                  4
------------------------
Group variables            None

-- Variable type: factor ----------------------------------------------
  skim_variable n_missing complete_rate ordered n_unique top_counts
1 carrier               0             1 FALSE         16 UA: 56117, B6: 52502
2 origin                0             1 FALSE          3 EWR: 109983, JFK: 92571
3 dest                  0             1 FALSE        104 LAX: 15373, ATL: 14064
4 model                 0             1 FALSE        121 A32: 45122, EMB: 25647

-- Variable type: numeric ---------------------------------------------
  skim_variable  n_missing complete_rate  mean     sd   p0  p25  p50  p75 p100
1 sched_dep_time         0             1 1343. 472.    500  901 1355 1730 2359
2 sched_arr_time         0             1 1529. 507.      1 1118 1550 1946 2359
3 distance               0             1 1077. 764.     80  529  937 1416 4983
4 year.y                 0             1 2001.   6.40 1956 1999 2002 2006 2013
```

(h) [2 pt / 21 pts] If we use $\boldsymbol{X}$ as the feature matrix to model, what information (that we have access to from the raw data frame) are we leaving out that might be important?

The $\boldsymbol{M}$ matrix of missing dummy variables. Also `seats` could be an answer as well.

(i) [3 pt / 24 pts] What is the value of $p_{raw} + 1$? That is, if we were to consider the model `lm(y ~ ., X)`, how many coefficients would it fit? Assume none of the features are collinear. If the answer cannot be determined, write "cannot be determined".

$$p_{raw} + 1 = \underbrace{4}_{\text{numeric features}} + \underbrace{16 - 1}_{\text{carrier}} + \underbrace{3 - 1}_{\text{origin}} + \underbrace{104 - 1}_{\text{dest}} + \underbrace{121 - 1}_{\text{model}} + \underbrace{1}_{\text{intercept}} = 245$$

(j) [4 pt / 28 pts] If we were to consider the model `lm(y ~ . * ., X)`, what is the value of $p + 1$? That is, how many coefficients would it fit? Assume none of the features are collinear. If the answer cannot be determined, write "cannot be determined". Advice: leave this problem for last as it is difficult!

$$\begin{aligned} p + 1 &= 4 + 15 + 2 + 103 + 120 + \binom{4}{2} + \\ &\quad 4 * (15 + 2 + 103 + 120) + \\ &\quad 15 * (2 + 103 + 120) + \\ &\quad 2 * (103 + 120) + \\ &\quad 103 * 120 + 1 \\ &= 17,392 \end{aligned}$$

The first line is the answer to (i) plus the interactions among the four numeric variables. The next line is the numeric variables interacted with all the categorical variables' levels. The next line is carrier cross others; the next is origin cross others and the next is dest cross model and finally, the intercept. In class I erroneously stated that `R`'s ". * ." notation includes squares (but it does not). So if you include the squares for the numeric variables, you would add four more to arrive at 17,396 and that would be marked correct.

We are going to do model selection, so we split the data into train-test:

```
> n_train = 5000
> idx_train = sample(1 : n, n_train); idx_test = setdiff(1 : n, idx_train)
> y_train = y[idx_train];             y_test = y[idx_test]
> X_train = X[idx_train, ];           X_test = X[idx_test, ]
```

Let's take a look at some simple models first in the training data. Remember since we took a subset, we may not have the same number of levels in the categorical variables as the entire dataset.

(k) [3 pt / 31 pts] If we were to do the model selection procedure using cross validation, would there likely need to be an "outer loop"? Yes / no and explain your answer.

No. The test set has $n - 5000 \approx 270,000$ observations. Thus, the stability of our oos measurement will be very good and thus there is no need to cross-validate.

```
> round(coef(lm(y_train ~ carrier * sched_dep_time, X_train)), 3)
            (Intercept)               carrierAA               carrierAS 
                -19.278                  -8.321                 -26.328 
              carrierB6               carrierDL               carrierEV 
                  7.927                   2.404                   9.152 
              carrierF9               carrierFL               carrierHA 
                  5.789                  42.223                 -94.050 
              carrierMQ               carrierUA               carrierUS 
                 -0.901                  -3.454                  13.552 
              carrierVX               carrierWN               carrierYV 
                -21.895                 -13.496                 -26.498 
         sched_dep_time carrierAA:sched_dep_time carrierAS:sched_dep_time 
                  0.017                   0.004                   0.013 
carrierB6:sched_dep_time carrierDL:sched_dep_time carrierEV:sched_dep_time 
                 -0.005                  -0.004                   0.001 
carrierF9:sched_dep_time carrierFL:sched_dep_time carrierHA:sched_dep_time 
                 -0.003                  -0.018                   0.092 
carrierMQ:sched_dep_time carrierUA:sched_dep_time carrierUS:sched_dep_time 
                  0.011                   0.004                  -0.010 
carrierVX:sched_dep_time carrierWN:sched_dep_time carrierYV:sched_dep_time 
                  0.021                   0.016                   0.022 
```

(l) [2 pt / 33 pts] Using the above model, predict the delay for an American Airlines flight (`carrier` = AA) leaving at 10AM (`sched_dep_time` = 1000) to three decimals. This carrier has 9,787 flights in the total dataset. If the answer cannot be determined, write "cannot be determined".

$\hat{y} = -19.278 + -8.321 + 0.017 * (1000) + 0.004 * (1000) = -6.599$

(m) [3 pt / 36 pts] Using the above model, predict the delay for an Endeavor Air flight (`carrier` = 9E) leaving at 8PM (`sched_dep_time` = 2000) to three decimals. This carrier has 17,377 flights in the total dataset. If the answer cannot be determined, write "cannot be determined".

If you assumed 9E was the reference category, then the answer is

$$\hat{y} = -19.278 + 0.017 * (2000) = 14.722.$$

However since only 14 coefficients were given (due to one being dropped in the $\boldsymbol{X}_{train}$ sub dataset by mistake), the answer really "cannot be determined".

```
> round(coef(lm(y_train ~ poly(distance, 3), X_train)), 3)
      (Intercept) poly(distance, 3)1 poly(distance, 3)2 poly(distance, 3)3
            7.621           -244.235            -84.548             56.173
```

(n) [3 pt / 39 pts] Using the above model, predict the delay for a flight with distance of 1,000mi to three decimals. If the answer cannot be determined, write "cannot be determined".

Cannot be determined (as you don't have the formula for orthogonal polynomials).

```
> round(coef(lm(y_train ~ poly(distance, 3, raw = TRUE), X_train)), 3)
                  (Intercept) poly(distance, 3, raw = TRUE)1
                        8.460                          0.005
poly(distance, 3, raw = TRUE)2 poly(distance, 3, raw = TRUE)3
                        0.000                          0.000
```

(o) [2 pt / 41 pts] Using the above model, predict the delay for a flight with distance of 1,000mi to three decimals. If the answer cannot be determined, write "cannot be determined".

$\hat{y} = 8.460 + 0.005 * (1000) + 0.000 * (1000^2) + 0.000 * (1000^3) = 13.460$

(p) [3 pt / 44 pts] What is the danger of using high degree polynomial models and why?

Large extrapolation errors due to Runge's phenomenon.

```
> round(coef(lm(y_train ~ log(distance), X_train)), 3)
  (Intercept) log(distance)
       29.848        -3.313
```

(q) [2 pt / 46 pts] Using the above model, interpret the effect of $x =$ `log(distance)` on the predicted arrival time, $\hat{y}$. If the answer cannot be determined, write "cannot be determined".

A 1% increase in distance decreases $\hat{y}$, the predicted arrival time, by $3.313 \times 1\% = 0.0313$ minutes.

We now fit a model from forward stepwise OLS regression procedure. To do so, we split the 5,000 training observations *further* into a subtraining set of 4,500 observations and a selection set of 500 observations. We select the model using the oosR^2 error metric.

The pool of features we consider is given by R's formula notation ". * .", i.e. all first order interactions from part (j). Regardless of your answer to (j), assume that the answer to (j) is greater than 4,500, the number of observations in the subtraining set.

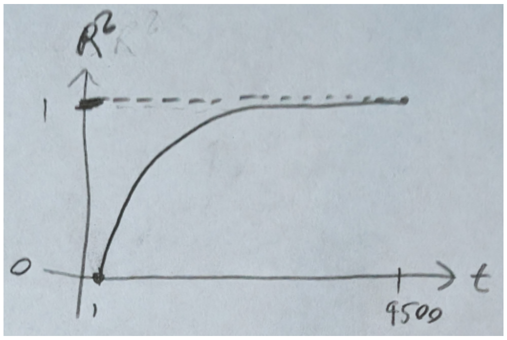
(r) [1 pt / 47 pts] If we were to do this forward stepwise OLS regression procedure using K-fold cross-validation with the subtraining set of 4,500 observations and a selection set of 500 observations, what is K (how many folds would we do)?

5,000 / 500 = 10

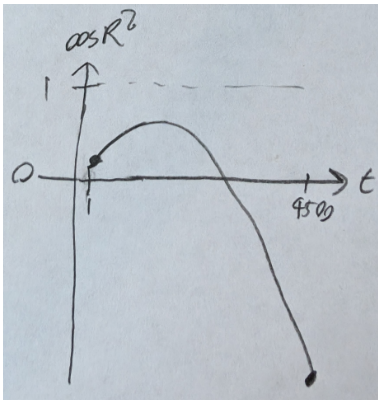
(s) [3 pt / 50 pts] What would be the benefit to doing this forward stepwise selection procedure using K-fold cross validation? Explain.

It would yield a better selected model as the oosR^2 error metric will be more stable.

(t) [3 pt / 53 pts] Draw the in-sample R^2 by the iteration number. Label your axes (y is R^2 and x is t). Label critical points on both the x and y axes.



(u) [4 pt / 57 pts] Draw the oosR^2 by the iteration number. Label your axes (y is oosR^2 and x is t). Label critical points on both the x and y axes.

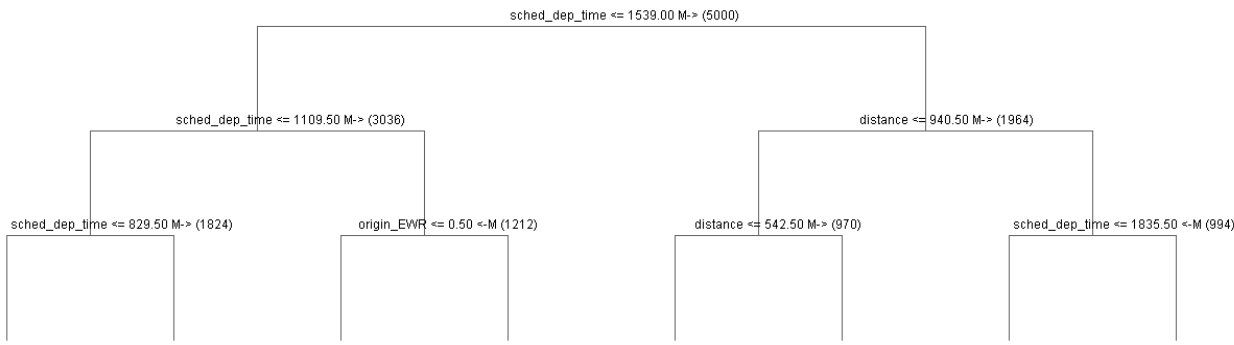


Note: the oosR^2 point at $t = 1$ will be positive or negative but approximately zero but its true value is unknown and the oosR^2 point at $t = 4500$ will be negative but its true value is unknown.

We now model arrival delay using a regression tree with the default hyperparameter value of $N_0 = 5$ for regression:

```
> tree_mod = YARFCART(X_train, y_train)
```

and print out an illustration of tree_mod to depth 3:



(v) [1 pt / 58 pts] Assuming the tree model above, what is the most important variable?

sched_dep_time

(w) [1 pt / 59 pts] Assuming the tree model above, what is the second most important variable?

distance

Also compute a linear model:

```
> ols_mod = lm(y_train ~ ., X_train)
```

(x) [1 pt / 60 pts] Assume the real function f has many non-linearities and interactions among the p_{raw} features. Which model would likely perform better in the future?
Circle one... ols_mod / tree_mod

(y) [1 pt / 61 pts] Assume the real function f does not have many non-linearities and interactions among the p_{raw} features. Which model would likely perform better in the future?
Circle one... ols_mod / tree_mod

(z) [3 pt / 64 pts] Assuming you cannot use different training data, how can you improve the oos performance during fitting a single regression tree?

Use the model selection procedure to optimize the hyperparameter value N_0.

We now model arrival delay using a bag-of-trees model and a Random Forests model where in the latter we use the default hyperparameter value of floor($p_{raw}/3$):

```
> bag_tree_mod = YARFBAG(X_train, y_train, num_trees = 2000)
> bag_tree_mod$rmse_oob
[1] 43.3813
> rf_mod = YARF(X_train, y_train, num_trees = 2000)
> rf_mod$rmse_oob
[1] 42.6371
```

(aa) [1 pt / 65 pts] Which model is more interpretable?
Circle one... `rf_mod` / `bag_tree_mod` / `tree_mod`

(bb) [1 pt / 66 pts] Which model would likely perform better in the future?
Circle one... `rf_mod` / `bag_tree_mod` / `tree_mod`

(cc) [2 pt / 68 pts] If `rf_mod` was instead fit with $m_{try} = 1$, which model would likely perform better in the future?
Circle one... `rf_mod` / `bag_tree_mod` / `tree_mod`

(dd) [2 pt / 70 pts] Assuming you cannot use different training data, and assuming 2,000 trees is sufficient, how can you improve the oos performance during fitting the random forest model?

Use the model selection procedure to optimize the hyperparameter value m_{try}.

(ee) [6 pt / 76 pts] Fill in the blanks in following terms using one of the following symbols $>, <, \geq, \leq, =, ?$ where "?" should be used only if the comparison is not possible. Express the relationships that are expected (those that we spent time studying in class).

Bias of `tree_mod` $=$ Bias of `bag_tree_mod` $=$ Bias of `rf_mod` $<$ Bias of `ols_mod`

Var of `tree_mod` $>$ Var of `bag_tree_mod` $>$ Var of `rf_mod` $>$ Var of `ols_mod`

MSE of `tree_mod` $>$ MSE of `bag_tree_mod` $>$ MSE of `rf_mod` $<$ MSE of `ols_mod`

We now model arrival delay using the boosting model we discussed in class with 2,000 shallow trees as the base learner:

```
> Xmm = model.matrix(~ ., X)
> boost_mod = xgboost(
    data = Xmm[idx_train, ],
    label = y_train,
    objective = "reg:squarederror", #for a regression problem
    booster = "gbtree", #shallow trees, default depth = 6
    nrounds = 2000, #num trees (should be cross-validated)
    eta = 0.2, #learning rate (should be cross-validated)
    print_every_n = 100
  )

> n_test = 500
> y_hat_test = predict(boost_mod, Xmm[idx_test, ][1 : n_test, ])
> sqrt(mean((y_test[1 : n_test] - y_hat_test)^2)) #oosRMSE calculation
[1] 39.06686
```

(ff) [1 pt / 77 pts] Which model would likely perform better in the future?
Circle one... `rf_mod` / `boost_mod`

(gg) [1 pt / 78 pts] Which model likely has less estimation error?
Circle one... `rf_mod` / `boost_mod`

(hh) [2 pt / 80 pts] Which model likely has less stability in its estimate of future RMSE?
Circle one... `rf_mod` / `boost_mod`

(ii) [2 pt / 82 pts] Which model has more hyperparameter values to optimize?
Circle one... `rf_mod` / `boost_mod`

(jj) [3 pt / 85 pts] Fill in the blanks in following terms using one of the following symbols $>, <, \geq, \leq, =, ?$ where "?" should be used only if the comparison is not possible. Express the relationships that are expected (those that we spent time studying in class).

Bias of `rf_mod` ? Bias of `boost_mod`

Var of `rf_mod` ? Var of `boost_mod`

MSE of `rf_mod` ? MSE of `boost_mod`

We are now interested in changing the response to the following binary response: 1 if the flight is delayed by more than a half hour (i.e. if the previous metric $y > 30$) or if the flight is not (i.e. if the previous metric $y \leq 30$). We fit a logistic regression to the raw features.

```
> y_train_bin = as.numeric(y_train > 30)
> table(y_train_bin)
y_train_bin
   0    1
4205  795
> logistic_mod = glm(y_train_bin ~ . - model, X_train, family = "binomial")
```

Below is a subset of the coefficients in the fit $\boldsymbol{b}$:

```
> coef(logistic_mod)
   (Intercept) sched_dep_time sched_arr_time
 -1.975068e+01   1.317949e-03   3.681219e-05
      carrierAA      carrierAS      carrierB6
 -3.699871e-01  -1.802762e-01   3.718844e-02
      carrierDL      carrierEV      carrierF9
 -5.471695e-01   3.698079e-01   7.164400e-01
      carrierFL      carrierHA      carrierMQ
  1.374622e-01  -1.399129e+01  -2.033716e-01
      carrierUA      carrierUS      carrierVX
 -2.642582e-01  -3.790116e-01   8.489270e-02
      carrierWN      carrierYV      originJFK
  2.818394e-01   3.628071e-01  -3.211934e-01
      originLGA        destACK        destALB
 -4.340381e-01   4.986711e+01   4.930801e+01
```

(kk) [1 pt / 86 pts] circle one: Assuming you use the model `logistic_mod`, relative to Newark airport (the reference level of variable `origin`), you are ...

more / less

likely to be delayed by 30min or more if the origin is JFK.

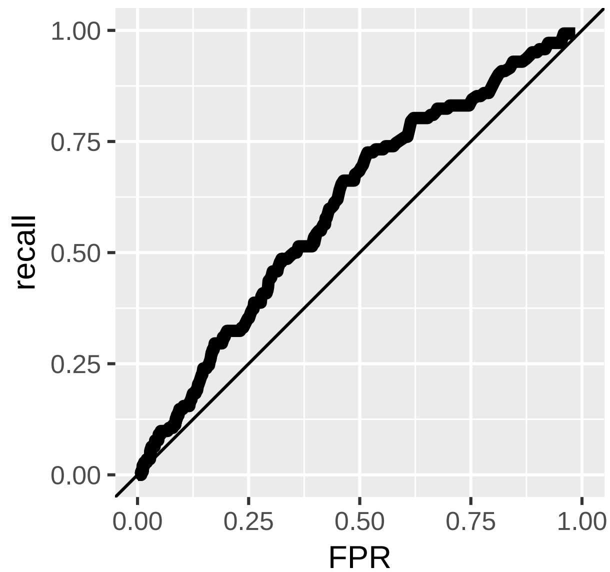
(ll) [3 pt / 89 pts] Using the model `logistic_mod`, you predict for your flight tomorrow (with measurements $\boldsymbol{x}_\star$) and compute $\boldsymbol{x}_\star \boldsymbol{b} = 3.171$. What is the predicted probability of a more than 30 min arrival delay to the nearest three decimals?

$$\frac{e^{3.171}}{1 + e^{3.171}} = 0.960 = 95.973\%$$

(mm) [2 pt / 91 pts] Our `logistic_mod`'s oos Brier score is -0.121. Does `logistic_mod` beat the naive probability estimation model where you let $\hat{p} = 0.5$ for all predicted units? Explain.

Yes. It has a higher Brier score than the naive model whose Brier score is -0.25.

We now compute the oos ROC curve for `logistic_mod` and illustrate it below.



(nn) [1 pt / 92 pts] Estimate the oos AUC.

Since the upper triangle has area 0.5, I estimate the curve has about 15% of the area, so I'm estimating 0.55-0.65.

(oo) [1 pt / 93 pts] Circle one: the in-sample AUC is likely...

lower than / higher than the oos AUC.

(pp) [2 pt / 95 pts] Circle one: the easiest way to improve the AUC is likely to...

increase the training set size and fit the same model /
add interactions and quadratic terms and fit a different model

```
> table(y_test_bin, as.numeric(p_hat_test > 0.025))

y_test_bin   0   1
         0  19 839
         1   1 141

> table(y_test_bin, as.numeric(p_hat_test > 0.05))

y_test_bin   0   1
         0  96 762
         1   7 135
> table(y_test_bin, as.numeric(p_hat_test > 0.1))

y_test_bin   0   1
         0 419 439
         1  41 101
> table(y_test_bin, as.numeric(p_hat_test > 0.2))

y_test_bin   0   1
         0 692 166
         1  98  44
> table(y_test_bin, as.numeric(p_hat_test > 0.3))

y_test_bin   0   1
         0 804  54
         1 128  14
> table(y_test_bin, as.numeric(p_hat_test > 0.4))

y_test_bin   0   1
         0 846  12
         1 139   3
```

(qq) [2 pt / 97 pts] We are interested in minimizing the error that occurs in the future when we (1) predict the flight is not delayed by more than 30min and then (2) in reality it is delayed by more than 30min. Of the asymmetric models shown above, which value of p_{th} would optimize for our goal?

0.025

(rr) [3 pt / 100 pts] Assuming you chose the model in the previous question, what is the tradeoff if you use this model in the future?

You make many false positive errors.