Data 100 Principles and Techniques of Data Science

Summer 2019

Final Solutions

INSTRUCTIONS

- You have 2 hours and 50 minutes to complete the exam.
- The exam is closed book, closed notes, closed computer, closed calculator, except for the provided midterm reference sheet and up to two 8.5" × 11" sheets of notes of your own creation.
- There are 12 pages on this exam and a total of 110 points possible.
- Write your name at the top of each sheet of the exam.
- Mark your answers **on the exam itself**. We will *not* grade answers written on scratch paper.

Last name	
First name	
Student ID number	
CalCentral email (_@berkeley.edu)	
Name of the person to your left	
Name of the person to your right	
All the work on this exam is my own. **(please sign)**	

Terminology and Notation Reference:

$\exp(x)$	e^x
$\log(x)$	$\log_e x$
Linear regression model	$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{x} \cdot \boldsymbol{\theta} = \boldsymbol{\theta} \cdot \boldsymbol{x}$
Logistic (or sigmoid) function	$\sigma(t) = \frac{1}{1+\exp(-t)}$
Logistic regression model	$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = P(Y=1 \mid X) = \sigma(\boldsymbol{\theta} \cdot \boldsymbol{x})$
Squared error loss function	$\ell(\boldsymbol{x}, y, \boldsymbol{\theta}) = (y - f_{\boldsymbol{\theta}}(\boldsymbol{x}))^2$
Absolute error loss function	$\ell(\boldsymbol{x}, y, \boldsymbol{\theta}) = \lvert y - f_{\boldsymbol{\theta}}(\boldsymbol{x}) \rvert$
Cross-entropy loss function	$\ell(\boldsymbol{x}, y, \boldsymbol{\theta}) = -y \log f_{\boldsymbol{\theta}}(\boldsymbol{x}) - (1-y)\log(1 - f_{\boldsymbol{\theta}}(\boldsymbol{x}))$
Bias	$\text{Bias}(\hat{\theta}, \theta^*) = E[\hat{\theta}] - \theta^*$
Variance	$\text{Var}(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$

This page is intentionally left blank, but feel free to use it as scratch paper.

1. **(16 points) Flights**

Fill in both the Python code and the SQL query to produce each result below, assuming that the following tables are stored both as Pandas DataFrames and SQLite tables. **Only the first few rows are shown for each table.** The `acc` table contains one row per user account registered at an airline company's website. The `fly` table contains one row per flight that the airline offers. The `fact` table is a fact table containing one row per flight booked by a user—its `uid` and `fid` columns are foreign keys to the corresponding columns in `acc` and `fly`. Some accounts have never booked a flight, and some flights have never been booked by a user.

Note: On this problem, some blanks should be filled in using more than one keyword or expression.

`acc`

uid	name
1	Sam
4	Leo
3	Steph
6	Manana

`fact`

uid	fid
4	1
4	2
3	4
2	1

`fly`

fid	orig	dest	price
1	LA	SF	110
2	LA	SF	90
3	SF	MN	240
4	SD	NY	370

(a) **(4 pt)** Find the names of all users that did not book any flights.

Python: `acc.loc[~acc['uid'].isin(fact['uid']), 'name']`

SQL: `SELECT name FROM acc WHERE uid NOT IN (SELECT uid FROM fact);`

(b) **(6 pt)** Find the average flight price for each unique pair of origin and destination cities.

Python:
```
(fly[['orig', 'dest', 'price']].groupby(['orig', 'dest'])

    .mean().reset_index())
```

SQL:
```
SELECT orig, dest, AVG(price) AS price FROM fly

  GROUP BY orig, dest
```

(c) **(6 pt)** Find all unique origin and destination city pairs that can be reached in a sequence of two consecutive flights. For example, LA → MN can be reached using `fid` 1 and 3. The result should have two columns: the first column holds origin cities (e.g. LA) and the second holds destination cities (e.g. MN).

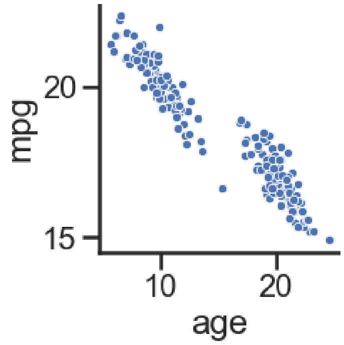
Hint: The `suffixes=(1, 2)` argument to `pd.merge` appends a 1 to column labels in the first table and a 2 to column labels in the second table if the merged tables share column names.

Hint: In SQL, `AS t1` and `AS t2` enable retrieving a table's columns using the aliases `t1` and `t2`. For example, `t1.fid` gets the `fid` column of the table referenced by `t1`.

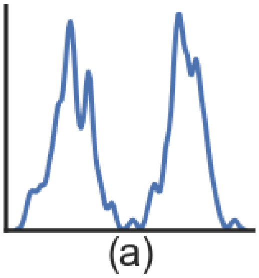
Python:
```
m = pd.merge(fly, fly, left_on='dest', right_on='orig', suffixes=(1, 2))
m.loc[:, ['orig1', 'dest2']].drop_duplicates()
```

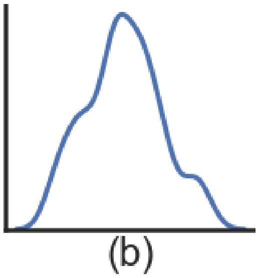
SQL:
```
SELECT DISTINCT t1.orig, t2.dest
 FROM fly AS t1 JOIN fly AS t2 ON t1.dest = t2.orig
```

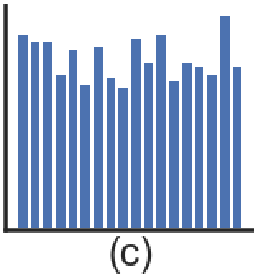
2. (15 points) Lost Labels

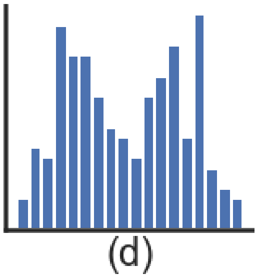


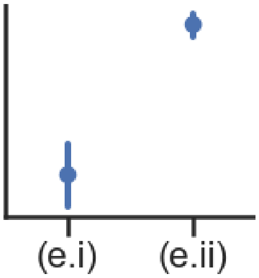
During a data analysis on car attributes, Sam created several plots. However, he has lost the axis labels for all of his plots except for the scatter plot shown on the left. Determine whether the plots below were generated from the same data. If so, mark the axis label that makes each plot consistent with the data in the scatter plot.
Assume that: The KDE plots use the same bandwidth, the histograms use the same number of bins, and point plots show the means of two columns and 95% confidence intervals. The axis limits for each plot were automatically chosen to display all plotted marks.

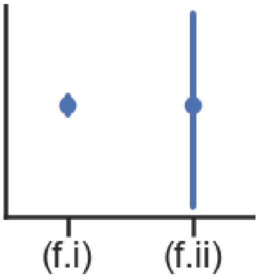


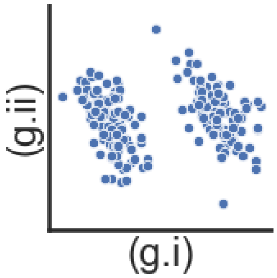


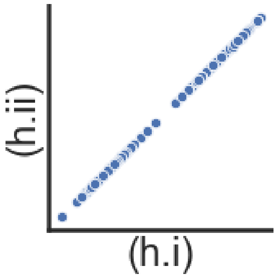












(a) **(7 pt)** Fill the missing axis labels of the 8 plots above using either age or mpg to make the plots consistent with the labeled scatter plot. For example, the first plot shows the distribution of age, so (a) should be filled in with age. If the plot cannot be generated from the data in either age or mpg, select Neither.

	(a)	(b)	(c)	(d)	(e.i)	(e.ii)	(f.i)	(f.ii)	(g.i)	(g.ii)	(h.i)	(h.ii)
age	●	○	○	○	⊗	○	○	○	○	○	⊗	⊗
mpg	○	○	○	⊗	○	⊗	○	○	○	○	○	○
Neither	○	⊗	⊗	○	○		⊗		⊗		○	

(a-d): Notice that both age and mpg are bimodal, but the data have a gap in age and not mpg.
(e-g): The average age is lower than the average mpg.
(h): Either age for both axes or mpg for both axes is correct. The same variable plotted on both x and y axes will give a line with slope 1.

(b) **(8 pt)** After conducting PCA, Sam projected each point onto the two principal component axes. He stored the projections onto the first and second principal components in the columns pc1 and pc2, respectively. As in the previous part, fill in each of the missing axis labels using either pc1 or pc2 if the plots were generated using the points projected onto the first or second principal component, or select Neither.

	(a)	(b)	(c)	(d)	(e.i)	(e.ii)	(f.i)	(f.ii)	(g.i)	(g.ii)	(h.i)	(h.ii)
pc1	⊗	○	○	○	○	○	○	⊗	⊗	○	⊗	⊗
pc2	○	⊗	○	○	○	○	⊗	○	○	⊗	○	○
Neither	○	○	⊗	⊗	⊗		○		○		○	

The first PC points from the upper left of the scatter plot to the lower right. The second PC is perpendiular to the first and points from lower left to upper right.
(a): After projecting on the first PC, there is a gap between the two clusters of data.
(b): The points generally lie along the first PC with large variations occurring less frequently than small variations.
(c-d): Neither of these could have been generated from the projected points.
(e-f): Remember that we subtract the average value from each column before conducting SVD. This means that the points are always centered at 0 after projection, ruling out choice (e). The first PC captures more variance than the second PC, so (f.ii) is pc1 and (f.i) is pc2.
(g): pc1 is the x-axis because the points are divided into two clusters along the first PC.
(h): As in the previous part, either pc1 for both axes or pc2 for both axes are correct.

3. (5 points) Parking Problems

A parking lot on campus has a \$10 parking fee per day. You find out that every morning, a police officer flips three fair coins. If all three coins land heads, the officer will go to the parking lot and give a \$64 parking ticket to all cars that did not pay the parking fee.

Let X be a random variable for the dollar amount you will pay on a particular day if you decide to **never** pay the parking fee. Note that all fractions shown in this problem are fully simplified.

(a) (2 pt) What is $E(X)$?

○ 0 ○ $\frac{1}{8}$ ○ $\frac{5}{4}$ ○ 5 ○ 8 ○ 24 ○ 32 ○ Other

(b) (2 pt) What is $Var(X)$?

○ 0 ○ $\frac{1}{8}$ ○ $\frac{7}{64}$ ○ 7 ○ 56 ○ 64 ○ 448 ○ 512 ○ Other

(c) (1 pt) Based on the calculations in (a) and (b), which parking strategy will save you the most money in the long run?

○ Never pay the parking fee.
○ Flip a fair coin and pay the parking fee only if the coin lands heads.
○ Always pay the parking fee.

4. (5 points) Derive It

To estimate a population parameter from a sample $(x_1, \ldots, x_n)$, we select the following empirical risk:

$$L(\theta) = -\frac{1}{n}\sum_{i=1}^{n} \log(\theta e^{-\theta x_i})$$

Find the estimator $\hat{\theta}$ that minimizes the empirical risk. Show all your work within the space provided below and **draw a box** around your final answer.

$$\begin{aligned} L(\theta) &= -\frac{1}{n}\sum_{i=1}^{n} \log(\theta e^{-\theta x_i}) \\ &= -\frac{1}{n}\sum_{i=1}^{n} (\log(\theta) - \theta x_i) \\ \nabla_\theta L(\theta) &= -\frac{1}{n}\sum_{i=1}^{n} \left(\frac{1}{\theta} - x_i\right) \\ &= -\frac{1}{\theta} + \frac{1}{n}\sum_{i=1}^{n} x_i \end{aligned}$$

$$\frac{1}{\hat{\theta}} - \frac{1}{n}\sum_{i=1}^{n} x_i = 0$$

$$\frac{1}{\hat{\theta}} = \overline{x}$$

$$\hat{\theta} = \frac{1}{\overline{x}}$$

or: $\hat{\theta} = \frac{1}{\frac{1}{n}\sum_{i=1}^{n} x_i}$

5. (2 points) Modeling

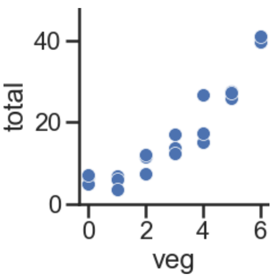
Shade in the box for **all the models** that are appropriate for the modeling problems described below.

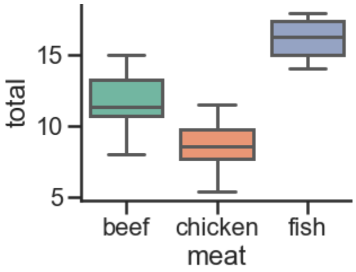
	Linear Regression	Logistic Regression	Random Forest
Predict day of the week from the number of shoppers at a store.	□	■	■
Predict total revenue today for a store from weather forecast of either sunny or rainy.	■	□	■
Predict number of apples sold from number of chickens sold.	■	□	■
Predict fastest checkout line $(1, 2, \ldots, 8)$ from number of people in each line.	□	■	■

6. (23 points) Grocery Associations

Every week, Manana goes to her local grocery store and buys a varying amount of vegetables but always buys exactly one pound of meat. We use a linear regression model to predict her total grocery bill. We've collected a dataset containing the pounds of vegetables bought, the type of meat bought (either beef, chicken, or fish), and the total bill. Below we display the first few rows of the dataset and two plots generated using the entire dataset.

veg	meat	total
1	beef	13
3	fish	19
2	beef	16
0	chicken	9



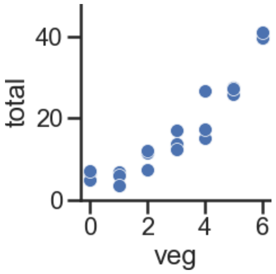


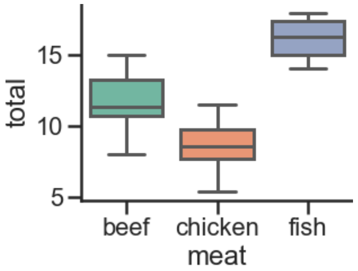
(a) **(8 pt)** Suppose we fit the following linear regression models to predict `total`. Based on the data and visualizations shown above, determine whether the fitted model weights are positive (+), negative (-), or exactly 0. The notation `meat=beef` refers to the one-hot encoded `meat` column with value 1 if the original value in the `meat` column was `beef` and 0 otherwise.

Model	Weight	+	-	0	Not enough info
$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0$	θ_0	⊗	○	○	○
$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 \cdot \texttt{veg}^2$	θ_0	⊗	○	○	○
	θ_1	⊗	○	○	○
$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 \cdot (\texttt{meat=beef}) + \theta_2 \cdot (\texttt{meat=chicken})$	θ_0	⊗	○	○	○
	θ_1	○	⊗	○	○
	θ_2	○	⊗	○	○
$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 \cdot (\texttt{meat=beef})$	θ_0	○	○	○	⊗
$+\theta_2 \cdot (\texttt{meat=chicken}) + \theta_3 \cdot (\texttt{meat=fish})$	θ_1	○	○	○	⊗
	θ_2	○	○	○	⊗
	θ_3	○	○	○	⊗

The data and plots from the previous page are reproduced here for convenience:

veg	meat	total
1	beef	13
3	fish	19
2	beef	16
0	chicken	9





Suppose we fit the model: $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \theta_0 + \theta_1 \cdot \texttt{veg} + \theta_2 \cdot (\texttt{meat=beef}) + \theta_3 \cdot (\texttt{meat=fish})$.
After fitting, we find that $\hat{\boldsymbol{\theta}} = [-3, 5, 8, 12]$. Calculate:

(b) (1 pt) The prediction of this model on the **first** point in our dataset.

○ -3 ○ 2 ○ 5 ○ [✓ 10] ○ 13 ○ 22 ○ 25

(c) (2 pt) The loss of this model on the **second** point in our dataset using squared error loss.

○ 0 ○ 1 ○ 5 ○ 6 ○ 8 ○ 24 ○ [✓ 25] ○ 169

(d) (2 pt) The loss on the **third** point in our dataset using absolute loss with L_1 regularization and $\lambda = 1$.

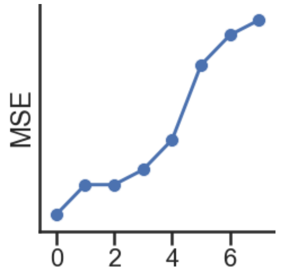
○ 0 ○ 1 ○ 15 ○ 24 ○ 26 ○ 27 ○ 28 ○ [✓ 29]

(e) (4 pt) Determine how each change below affects model bias and variance compared to the model described at the top of this page. **Shade in all the boxes that apply.**

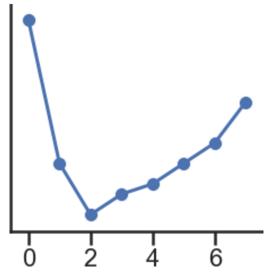
	Increase bias	Decrease bias	Increase variance	Decrease variance
Add degree 3 polynomial features	□	■	■	□
Add a feature of random numbers between 0 and 1	□	□	■	□
Collect 100 more sample points	□	□	□	■
Remove the `veg` column	■	□	□	■

(f) (4 pt) Suppose we predict `total` from `veg` using 8 models with different degree polynomial features (degrees 0 through 7). Which of the following plots display the training and validation errors of these models? Assume that we plot the degree of polynomial features on the x-axis, mean squared error loss on the y-axis, and the plots share y-axis limits.

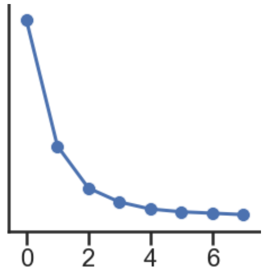
Plot A



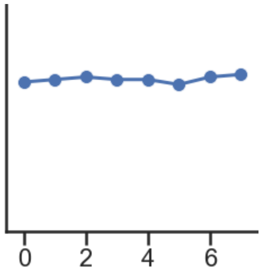
Plot B



Plot C



Plot D



Training error: ○ A ○ B ○ [✓ C] ○ D Validation error: ○ A ○ [✓ B] ○ C ○ D

(g) (2 pt) Suppose that we fit 8 degree-4 polynomial models using ridge regression and the x-axis for the plots in the previous part show λ instead of polynomial degree. Which plots show the training and validation errors of the models?

Training error: ○ [✓ A] ○ B ○ C ○ D Validation error: ○ A ○ [✓ B] ○ C ○ D

7. (20 points) Logistic Regression

(a) (8 pt) Suppose we use the following regression model with a single model weight θ and loss function:

$$\begin{aligned} f_\theta(x) &= \sigma(\theta - 2) \\ \ell(\theta, x, y) &= -y \log f_\theta(x) - (1 - y)\log(1 - f_\theta(x)) + \frac{1}{2}\theta^2 \end{aligned}$$

Derive the **stochastic gradient descent update rule** for this model and loss function, assuming that the learning rate $\alpha = 1$. Your answer may only use the following variables: $\theta^{(t+1)}, \theta^{(t)}, y$, and the sigmoid function σ. Show all your work within the space provided and **draw a box around your final answer**.

Let $\sigma_i = f_\theta(x) = \sigma(\theta - 2)$.
First, we take the gradient of σ_i w.r.t θ. From lecture, we know that:

$$\nabla_\theta \sigma_i = \sigma_i(1 - \sigma_i)$$

Then, we find the gradient of the loss w.r.t θ:

$$\begin{aligned} \ell(\theta, x, y) &= -y \log \sigma_i - (1 - y)\log(1 - \sigma_i) + \frac{1}{2}\theta^2 \\ \nabla_\theta \ell(\theta, x, y) &= -\frac{y}{\sigma_i}\nabla_\theta \sigma_i - \frac{1 - y}{1 - \sigma_i}(-1)\nabla_\theta \sigma_i + \theta \\ &= -(y)(1 - \sigma_i) + (1 - y)(\sigma_i) + \theta \\ &= -(y - \sigma_i) + \theta \\ &= -(y - \sigma(\theta - 2)) + \theta \end{aligned}$$

This gives the SGD update rule:

$$\begin{aligned} \theta^{(t+1)} &= \theta^{(t)} + \alpha(y - \sigma(\theta^{(t)} - 2) - \theta^{(t)}) \\ \theta^{(t+1)} &= y - \sigma(\theta^{(t)} - 2) & \text{Since } \alpha = 1 \end{aligned}$$

Recall that in lecture we derived the following batch gradient descent (BGD) and stochastic gradient descent (SGD) update rules for logistic regression with L_2 regularization. This expression uses the same notation used in class where $\boldsymbol{X}$ is the $(n \times p)$ design matrix, $\boldsymbol{X_i}$ is a vector containing the values in the i'th row of $\boldsymbol{X}$, $\boldsymbol{y}$ is the length-n vector of outcomes, y_i is a single outcome (either 0 or 1), and $\boldsymbol{\theta}$ is a vector containing the model weights.

$$\text{Batch Gradient Descent: } \boldsymbol{\theta}^{(t+1)} = (1-2\lambda)\boldsymbol{\theta}^{(t)} + \alpha \left[\frac{1}{n} \sum_{i=1}^{n} (y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)}))\boldsymbol{X_i} \right]$$

$$\text{Stochastic Gradient Descent: } \boldsymbol{\theta}^{(t+1)} = (1-2\lambda)\boldsymbol{\theta}^{(t)} + \alpha \left[(y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)}))\boldsymbol{X_i} \right]$$

(b) (4 pt) What are the dimensions of the expressions below?

	Scalar	Length-n vector	Length-p vector	$(n \times p)$ matrix
$\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)}$	⊗	○	○	○
$y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)})$	⊗	○	○	○
$(y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)}))\boldsymbol{X_i}$	○	○	⊗	○
$\alpha \left[\frac{1}{n} \sum_{i=1}^{n} (y_i - \sigma(\boldsymbol{X_i} \cdot \boldsymbol{\theta}^{(t)}))\boldsymbol{X_i}\right]$	○	○	⊗	○

(c) (8 pt) Suppose we use SGD to fit a logistic regression model with L_2 regularization with one model weight and no intercept term. Combinations of $\theta^{(t)}, X_i, y_i, \lambda$ and α are listed below. Complete the combination such that $\theta^{(t+1)}$ will be the **most positive** after one iteration of SGD among the choices provided. The notation $\lambda \in \mathbb{R}$ means that λ is a fixed, unknown real number (which can be either positive, zero, or negative). If more than one choice produces the most positive value, select Tie. If you need more information on unknown variables to solve the problem, select Need Info.

(c.i) $\theta^{(t)} = 1,\ X_i = 5,\ y_i = 0,\ \lambda = 1,$

$\alpha =$ ○ -1 ○ 0 ○ 1 ○ Tie ○ Need Info

(c.ii) $\theta^{(t)} = -1,\ X_i = 5,\ \lambda \in \mathbb{R},\ \alpha = 1$

$y_i =$ ○ 0 ○ 1 ○ Tie ○ Need Info

(c.iii) $\theta^{(t)} \in \mathbb{R},\ X_i = 0,\ \lambda \in \mathbb{R},\ \alpha = 1$

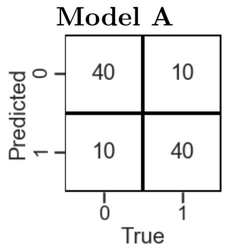
$y_i =$ ○ 0 ○ 1 ○ Tie ○ Need Info

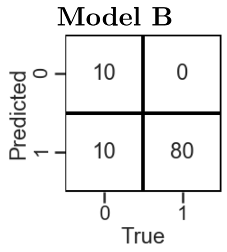
(c.iv) $\theta^{(t)} = 0,\ X_i \in \mathbb{R},\ \lambda \in \mathbb{R},\ \alpha = 1$

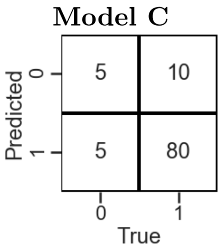
$y_i =$ ○ 0 ○ 1 ○ Tie ○ Need Info

8. (5 points) Classifiers

(a) (2 pt) Suppose we fit three classifiers which produce the following confusion matrices:

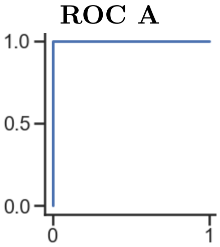


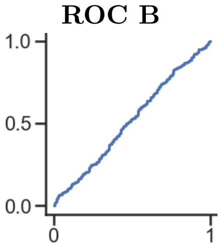


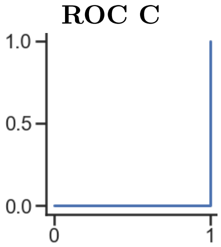


	Model A	Model B	Model C
Which model has the highest precision?	○	○	⊗
Which model has the highest recall?	○	⊗	○

(b) (3 pt) Suppose we fit three more classifiers and plot the ROC curves for each classifier on the test set. The test set contains 100 points: the first 50 points are labeled 0 and the second 50 points are labeled 1. Determine which models produce each ROC curve.







	ROC A	ROC B	ROC C
Predicts $P(Y = 1\|X)$ using a random number between 0 and 1	○	⊗	○
Assigns $P(Y = 1\|X) = 0.3$ to the first 50 points and $P(Y = 1\|X) = 0.4$ to the second 50 points.	⊗	○	○
Assigns $P(Y = 1\|X) = 0.8$ to the first 50 points and $P(Y = 1\|X) = 0.6$ to the second 50 points.	○	○	⊗

9. (12 points) If a Forest Falls...

(a) (8 pt) Suppose we fit decision trees of varying depths to predict `y` using `x1` and `x2`. For this question, a decision tree with depth 0 is a tree with no splitting (all points in a single node). What is the:

x1	**x2**	**y**
S	1	0
S	2	1
M	3	0
M	4	1
S	1	0
S	2	1
M	3	0
M	4	1

	0	0.5	1	2
Lowest possible entropy of a node in a fitted tree with depth 0?	○	○	⊗	○
Lowest possible entropy of a node in a fitted tree with depth 1?	⊗	○	○	○
Lowest possible entropy of a node in a fitted tree with depth 2?	⊗	○	○	○
Depth of a fitted decision tree with no depth limit?	○	○	○	⊗

Depth 0: all points are in a single node. Since half the points are class 0 and half are 1, the entropy is 1.

Depth 1: Splitting on x1 gives half 0 and half 1 for all children. However, using x2 can give a pure child if we use the rule `x2 <= 1`. The original solutions had 1 as the answer so we marked either 0 or 1 correct.

Depth 2: A tree of depth two can have all pure children (use `x1 == S` for the first split, then split using `x2`).

The depth of a fitted decision tree is 2 since at depth 2 it is possible for all child nodes to be pure.

(b) **(4 pt)** Select true or false for each statement about the bootstrap.

T F

⊗ ○ Increasing the number of bootstrap resamples increases the model bias of a random forest.

○ ⊗ Increasing the number of bootstrap resamples causes a confidence interval for the mean of a population to decrease in width.

⊗ ○ Increasing the number of bootstrap resamples does not change the center of a sampling distribution.

⊗ ○ After fitting any regression model, we can bootstrap the test set to create a confidence interval for the population error of a model.

The answer to the first statement is True. However, False was marked on the original solutions. When grading, either T or F for the first question was marked correct.

10. **(7 points)** **Thinking in Parallel**

For this question, assume that the following functions are written in Python to compute the average of a column in the DataFrame `df` and fit a decision tree to a design matrix. Assume that all other code outside these functions runs instantly, and that there is no limit on the number of functions that can be run in parallel unless otherwise stated.

```
@ray.remote
def avg(col):
    ... # Takes 1 second to run
```

```
@ray.remote
def fit_tree(df):
    ... # Takes 2 seconds to run
```

(a) **(1 pt)** If `df` contains 10 columns, how many seconds will it take to compute the average for all columns in `df` if we call `avg` serially?

○ 1 ○ 2 ○ 10 ○ 20

(b) **(2 pt)** If `df` contains 10 columns, how many seconds will it take to run the following code?

```
vals = []
for col in df.columns:
    vals.append(avg.remote(col))
ray.get(vals)
```

○ 1
○ 2
○ 10
○ 20

(c) **(2 pt)** How many seconds will it take to run the following code?

```
frst = []
for _ in range(10):
    frst.append(fit_tree.remote(df))
ray.get(frst)
```

○ 1
○ 2
○ 10
○ 20

(d) **(2 pt)** How many seconds will it take to run the code in part (c) if we can only run a maximum of four functions in parallel at a time?

○ 1 ○ 2 ○ 3 ○ 4 ○ 5 ○ 6

11. (0 points) Optional: Draw a Picture About Berkeley Data Science (or use this page for scratch work)