5-8 PM

Welcome to the

EECS 398 Review Session (Part II)

- see problems on course website
  (annotated slides + recording will be posted tonight)
- ask questions anonymously @ q.practicaldsc.org
- survey completion only at 75%! need 85% tonight

Final Review: Post-Midterm Content

← return to study.practicaldsc.org

The problems in this worksheet are taken from past exams in similar classes. Work on them **on paper**, since the exams you take in this course will also be on paper.

We encourage you to attempt these problems **before** Tuesday's exam review session, so that we have enough time to walk through the solutions to all of the problems.

We will enable the solutions here after the review session, though you can find the written solutions to these problems in other discussion worksheets.

Problem 1

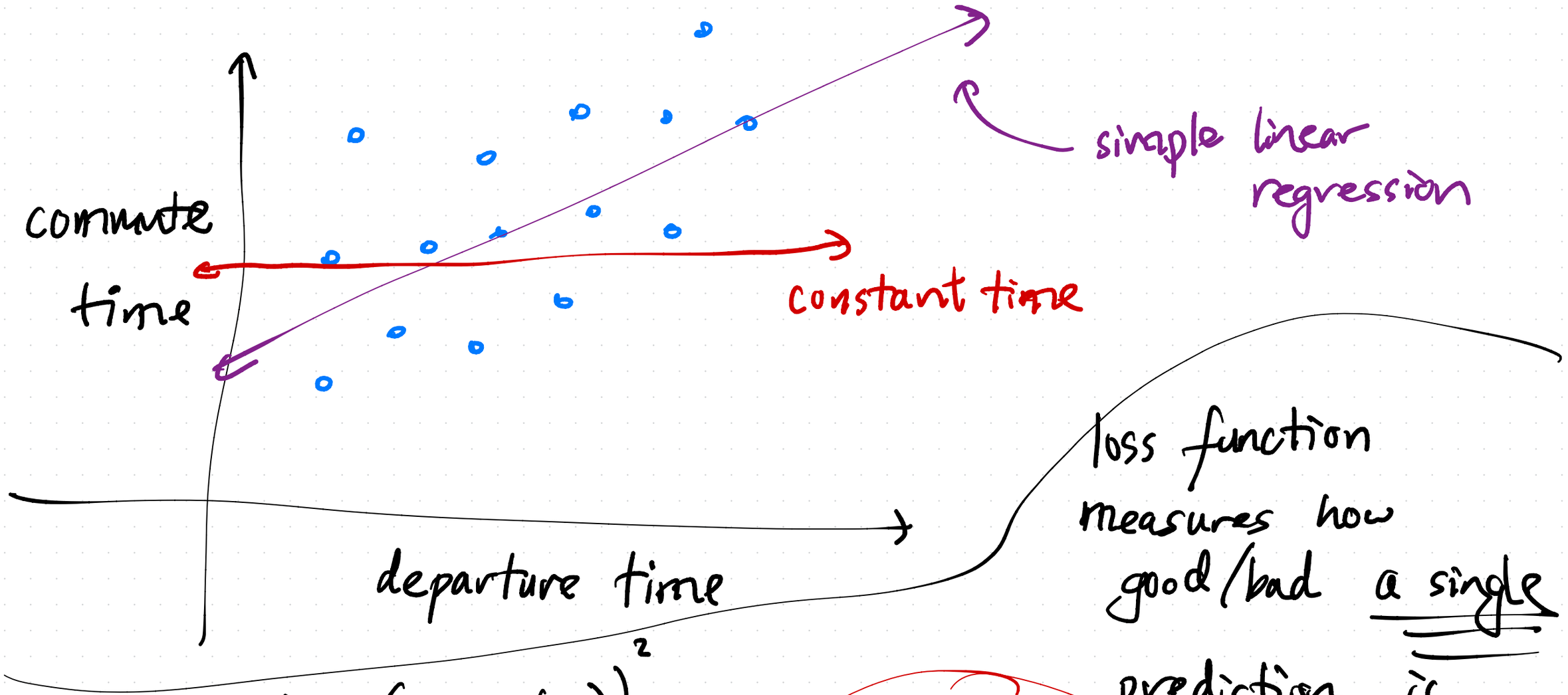
Consider a dataset of n values, $y_1, y_2, ..., y_n$, all of which are non-negative. We're interested in fitting a constant model, $H(x) = h$, to the data, using the new "Wolverine" loss function:

$$L_{\text{wolverine}}(y_i, h) = w_i \left(y_i^2 - h^2\right)^2$$

Here, w_i corresponds to the "weight" assigned to the data point y_i, the idea being that different data points can be weighted differently when finding the optimal constant prediction, h^*.

For example, for the dataset $y_1 = 1, y_2 = 5, y_3 = 2$, we will end up with different values of h^* when we use the weights $w_1 = w_2 = w_3 = 1$ and when we use weights $w_1 = 8, w_2 = 4, w_3 = 3$.

Loss functions: what's the point?



$$L_{sq}(y_i, H(x_i)) = (y_i - H(x_i))^2$$

(actual − pred)^2

we want small loss!

to find the best predictions (optimal parameters), we minimize AVERAGE LOSS,

also known as

EMPIRICAL RISK (R)

$$R_{sq}(H) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - H(x_i)\right)^2$$

n points in training data

e.g. squared loss

$$R_{sq}(h) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h)^2$$

e.g. constant model $H(x_i) = h$

to find the best constant h^*, minimize $R_{sq}(h)$

h^*: optimal parameter

Problem 1.1

Find $\frac{\partial L_{\text{wolverine}}}{\partial h}$, the derivative of the Wolverine loss function with respect to h. Show your work.

$$L(y_i, h) = w_i (y_i^2 - h^2)^2$$

$$\frac{dL}{dh} = w_i(2)(y_i^2 - h^2)(-2h)$$
$$= -4 w_i h (y_i^2 - h^2)$$

Problem 1.2

Prove that the constant prediction that minimizes average loss for the Wolverine loss function is:

$$h^* = \sqrt{\frac{\sum_{i=1}^n w_i y_i^2}{\sum_{i=1}^n w_i}}$$

$$R(h) = \frac{1}{n} \sum_{i=1}^n L(y_i, h) \rightarrow \boxed{\frac{dR}{dh} = 0}$$

Problem 1.3

For a dataset of non-negative values $y_1, y_2, ..., y_n$ with weights $w_1, 1, ..., 1$, evaluate:

$$\lim_{w_1 \to \infty} h^*$$

$$= \lim_{w_1 \to \infty} \sqrt{\frac{w_1 y_1^2 + \sum_{i=2}^n y_i^2}{w_1 + (n-1)}}$$

$$= \lim_{w_1 \to \infty} \sqrt{\frac{y_1^2 + \frac{\sum_{i=1}^n y_i^2}{w_1}}{1 + \frac{(n-1)}{w_1}}} = \sqrt{y_1^2} = \boxed{y_1}$$

- The maximum of $y_1, y_2, ..., y_n$
- The mean of $y_1, y_2, ..., y_{n-1}$
- The mean of $y_2, y_3, ..., y_n$
- The mean of $y_2, y_3, ..., y_n$, multiplied by $\frac{n}{n-1}$
- y_1
- y_n

$$\frac{dR}{dh} = \frac{1}{n} \sum_{i=1}^n \frac{dL}{dh}$$

$$0 = \frac{1}{n} \sum_{i=1}^n (-4 w_i h)(y_i^2 - h^2)$$

solve for h

$$h^* = \sqrt{\frac{\sum w_i y_i^2}{\sum w_i}}$$

Problem 2

Suppose we want to fit a hypothesis function of the form:

$$H(x) = w_0 + w_1 x^2$$

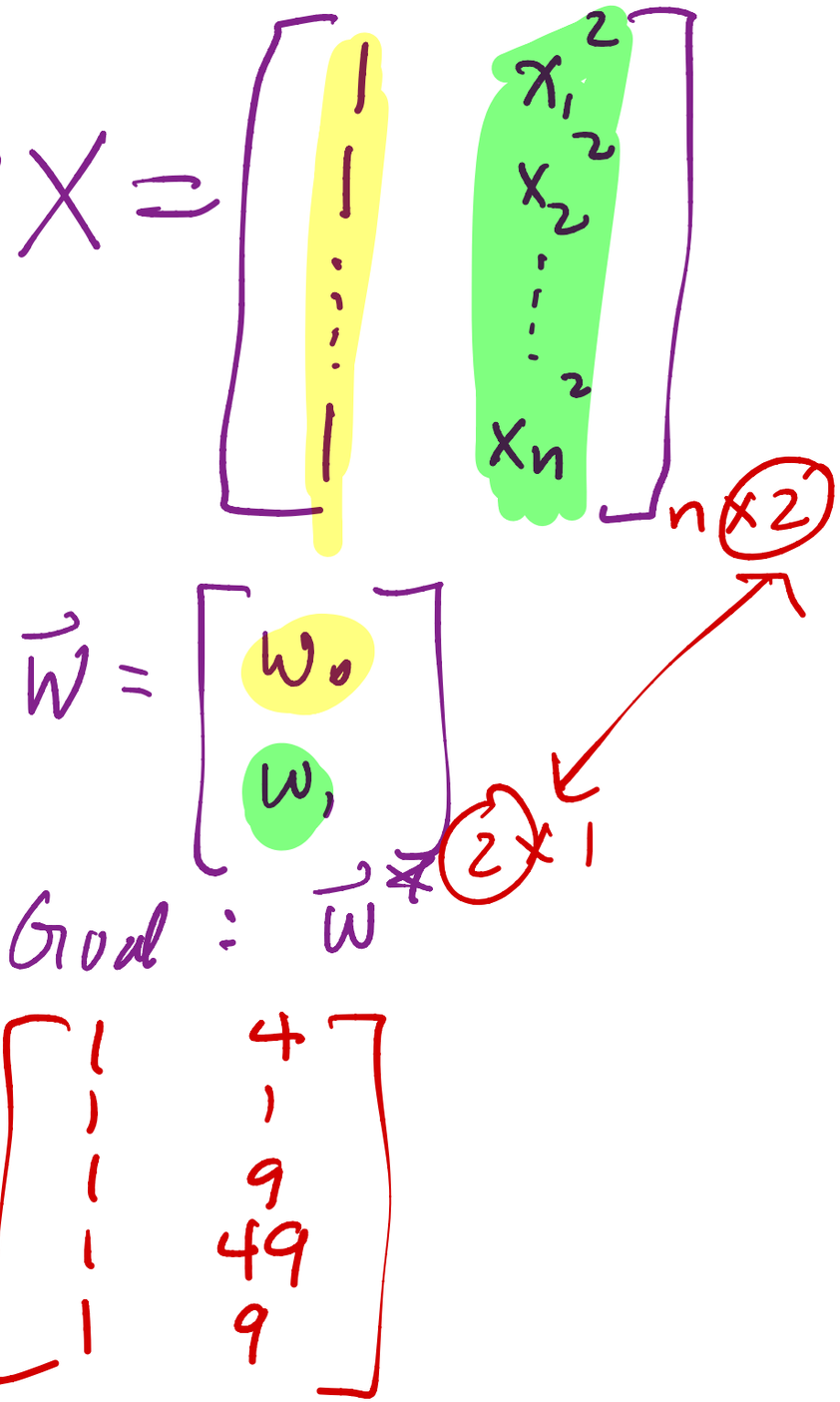
Note that this is *not* the simple linear regression hypothesis function, $H(x) = w_0 + w_1 x$.

To do so, we will find the optimal parameter vector $\vec{w}^* = \begin{bmatrix} w_0^* \\ w_1^* \end{bmatrix}$ that satisfies the normal equations.

The first 5 rows of our dataset are as follows, though note that our dataset has n rows in total.

x	y
2	4
-1	4
3	4
-7	4
3	4

Suppose that $x_1, x_2, ..., x_n$ have a mean of $\bar{x} = 2$ and a variance of $\sigma_x^2 = 10$.



Problem 2.1

Write out the first 5 rows of the design matrix, X.

Problem 2.2

Why linear algebra? To find these parameters!!!

→ so we can use multiple input variables, i.e. features!

each individual (row in dataset) represented by a feature vector, $\vec{x}_i \in \mathbb{R}^d$ (d features)

Example: Predict commute time given departure hour, day of month, and temperature

$$\vec{x}_i = \begin{bmatrix} \text{departure hour}_i \\ \text{day of month}_i \\ \text{temp}_i \end{bmatrix}_{3\times 1} = \begin{bmatrix} x_i^{(1)} \\ x_i^{(2)} \\ x_i^{(3)} \end{bmatrix}$$

this model has 4 parameters:

w_0, w_1, w_2, w_3

Suppose, just in part (b), that after solving the normal equations, we find $\vec{w}^* = \begin{bmatrix} 2 \\ -5 \end{bmatrix}$. What is the predicted y value for $x = 2$? Give your answer as an integer with no variables. Show your work.

$X=2 \rightarrow X^2 = 4$

$\vec{w}^* \cdot \text{Aug}(\vec{x})$

Here,

$\vec{x}_i = [x_i^2]$

$\text{Aug}(\vec{x}_i) = \begin{bmatrix} 1 \\ x_i^2 \end{bmatrix}$

Here, $x_i = 2$, so

$\vec{w}^* \cdot \text{Aug}(\vec{x}_i)$

$= \begin{bmatrix} 2 \\ -5 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2^2 \end{bmatrix}$

$= (1)(2) + (2^2)(-5)$

$= \boxed{-18}$

Problem 2.3

Let $X_{\text{tri}} = 3X$. Using the fact that $\sum_{i=1}^{n} x_i^2 = n\sigma_x^2 + n\bar{x}^2$, determine the value of the bottom-left value in the matrix $X_{\text{tri}}^T X_{\text{tri}}$, i.e. the value in the second row and first column. Give your answer as an expression involving n. Show your work.

Problem 3

categorical

Suppose we have one qualitative variable that that we convert to numerical values using one- hot encoding. We've shown the first four rows of the resulting design matrix below:

	a	b	c	
→	1	0	0	a
→	1	0	0	a
→	0	0	1	c
→	0	1	0	b

↓

Problem 3.1

Say we train a linear model m_1 on these data. Then, we replace all of the 1 values in column **a** with 3's and all of the 1 values in column **b** with 2's and train a new linear model m_2. Neither m_1 nor m_2

$X_{tri} = 3X$

$= 3\begin{bmatrix} 1 & x_1^2 \\ \vdots & x_2 \\ & \vdots \\ 1 & x_n^2 \end{bmatrix}_{n\times 2}$

Goal: Find $X_{tri}^T X_{tri}$

$= (3X)^T(3X)$

$= 9X^TX$

$= 9\begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1^2 & x_2^2 & \cdots & x_n^2 \end{bmatrix}_{2\times n}$

$\begin{bmatrix} 1 & x_1^2 \\ 1 & x_2^2 \\ \vdots & \vdots \\ 1 & x_n^2 \end{bmatrix}_{n\times 2}$

$= 9\begin{bmatrix} n & \sum x_i^2 \\ \sum x_i^2 & \sum x_i^4 \end{bmatrix}$

bottom-left is

$\sum_{i=1}^{n} x_i^2 \overset{\text{hint}}{=} n\sigma_x^2 + n\bar{X}^2$

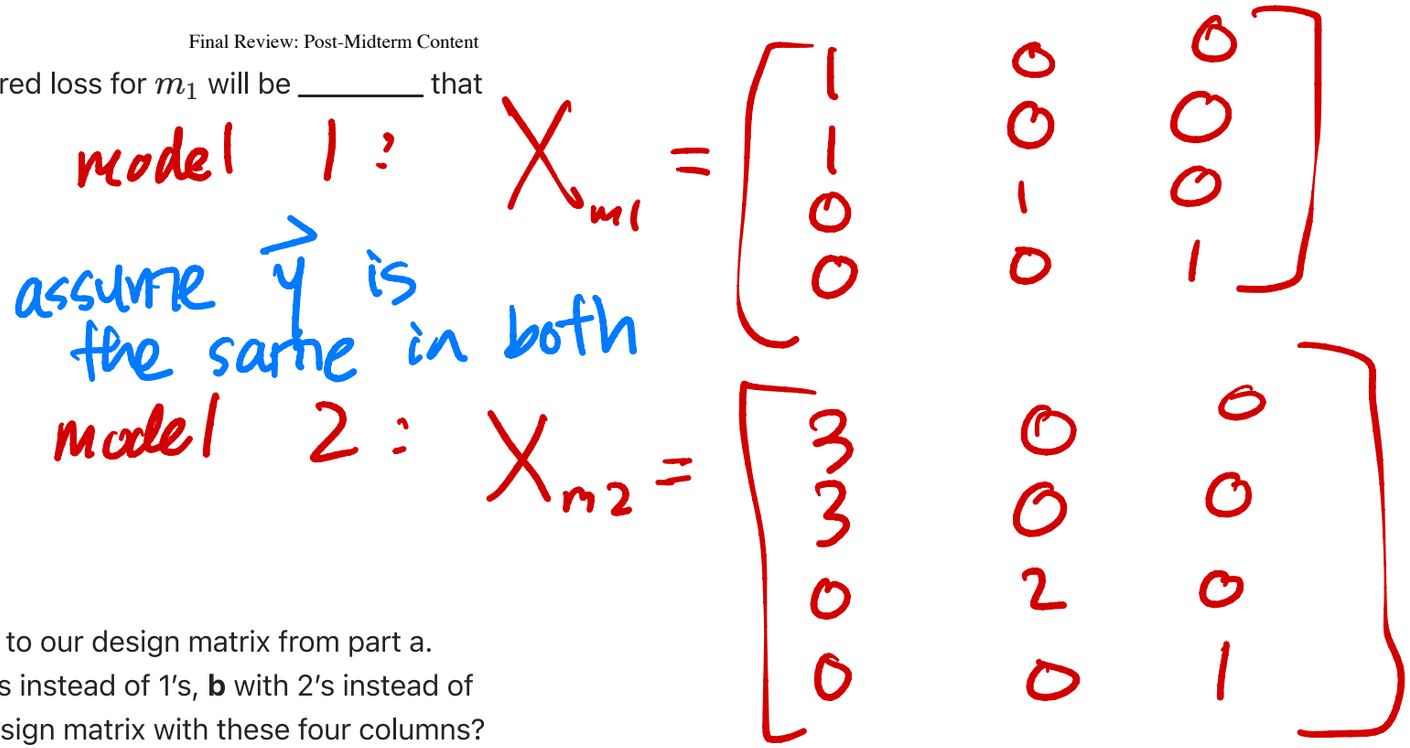
$= n \cdot 10 + n(2)^2$

$= 10n + 4n$

$= 14n$

have an intercept term. On the training data, the average squared loss for m_1 will be _____ that of m_2.

- greater than
- less than
- equal to
- impossible to tell



Problem 3.2

To account for the intercept term, we add a column of all ones to our design matrix from part a. That is, the resulting design matrix has four columns: **a** with 3's instead of 1's, **b** with 2's instead of 1's, **c**, and a column of all ones. What is the rank of the new design matrix with these four columns?

- 1
- 2
- 3
- 4

Problem 3.3

Suppose we divide our sampling frame into three clusters of people, numbered 1, 2, and 3. After we survey people, along with our survey results, we save their cluster number as a new feature in our design matrix. Before training a model, what should we do with the cluster column? (Note: This part is independent of parts a and b.)

Recall:

$$X\vec{w} = \vec{h} = \begin{bmatrix} H(\vec{x}_1) \\ H(\vec{x}_2) \\ \vdots \\ H(\vec{x}_n) \end{bmatrix}$$

- X: design matrix
- $\vec{w}$: param vector
- $\vec{h}$: hypothesis vector
- $H(\vec{x}_2)$: e.g. predicted y-value for 2nd row in X

How do we find $\vec{w}$? By minimizing MSE:

$$R(\vec{w}) = \frac{1}{n} \left\| \vec{y} - X\vec{w} \right\|^2$$

$\vec{y} - X\vec{w}$: error vector, $\vec{e}$

Key idea:

$$X_{m1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad X_{m2} = \begin{bmatrix} 3 & 0 & 0 \\ 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Key idea: $X_{m1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ and $X_{m2} = \begin{bmatrix} 3 & 0 & 0 \\ 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

have the same span!

a vector in span(X_{m1}) looks like

$$w_1 \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} + w_2 \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} + w_3 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Why (drop = "First" when one hot encoding?

Let's revisit X_{m1}, which didn't have [1].

$$X_{m1} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

rank 3

m_1 no intercept term

Typically, we will have an intercept term

$$X_{m_1}' = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

d a b c

rank 3

columns are not linearly independent!

e.g. $c = d - a - b$

issue: multicolinearity

- ○ Leave as is
- ○ One-hot encode it
- ○ Normalize it
- ○ Use bag of words

Problem 4

One piece of information that may be useful as a feature is the proportion of SAT test takers in a state in a given year that qualify for free lunches in school. The Series `lunch_props` contains 8 values, each of which are either `"low"`, `"medium"`, or `"high"`. Since we can't use strings as features in a model, we decide to encode these strings using the following `Pipeline`:

```
# Note: The FunctionTransformer is only needed to change the result
# of the OneHotEncoder from a "sparse" matrix to a regular matrix
# so that it can be used with StandardScaler;
# it doesn't change anything mathematically.
pl = Pipeline([
    ("ohe", OneHotEncoder(drop="first")),
    ("ft", FunctionTransformer(lambda X: X.toarray())),
    ("ss", StandardScaler())
])
```

After calling `pl.fit(lunch_props)`, `pl.transform(lunch_props)` evaluates to the following array:

```
array([[ 1.29099445, -0.37796447],
       [-0.77459667, -0.37796447],
       [-0.77459667, -0.37796447],
       [-0.77459667,  2.64575131],
       [ 1.29099445, -0.37796447],
       [ 1.29099445, -0.37796447],
       [-0.77459667, -0.37796447],
       [-0.77459667, -0.37796447]])
```

low med

OneHotEncoder → StandardScaler

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \rightarrow \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$

$$z_i = \frac{x_i - \bar{x}}{\sigma_x}$$

and `pl.named_steps["ohe"].get_feature_names()` evaluates to the following array:

```
array(["x0_low", "x0_med"], dtype=object)
```

Fill in the blanks. Given the above information, we can conclude that `lunch_props` has **(a)** value(s) equal to `"low"`, **(b)** value(s) equal to `"medium"`, and **(c)** value(s) equal to `"high"`. *(Note: You should write one positive integer in each box such that the numbers add up to 8.)*

2 4 3 (6)

What goes in the blanks?

3 low
1 medium
4 high

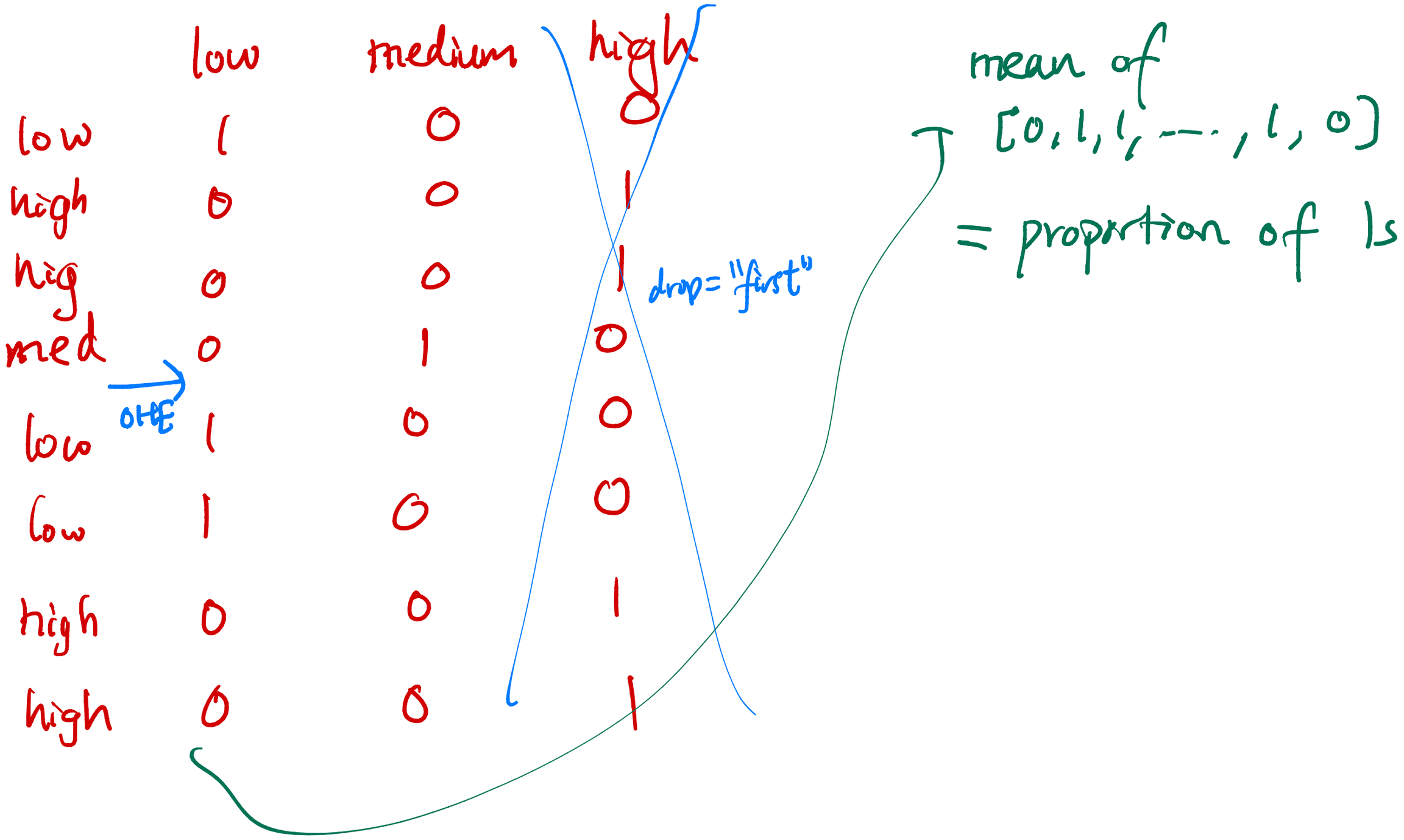
Problem 5

Consider the least squares regression model, $\vec{h} = X\vec{w}$. Assume that X and $\vec{h}$ refer to the design matrix and hypothesis vector for our training data, and $\vec{y}$ is the true observation vector.

Let $\vec{w}^*_{\text{OLS}}$ be the parameter vector that minimizes mean squared error without regularization. Specifically:

$$\vec{w}^*_{\text{OLS}} = \arg\min_{\vec{w}} \frac{1}{n} \|\vec{y} - X\vec{w}\|_2^2$$

Let $\vec{w}^*_{\text{ridge}}$ be the parameter vector that minimizes mean squared error with L_2 regularization, using a non-negative regularization hyperparameter λ (i.e. ridge regression). Specifically:


low
medium
high
low 1 0 0
high 0 0 1
hig 0 0 1
med 0 1 0
one
low 1 0 0
low 1 0 0
high 0 0 1
high 0 0 1
drop="first"
mean of
[0, 1, 1, ---, 1, 0]
= proportion of 1s

$$\vec{w}^*_{\text{ridge}} = \arg\min_{\vec{w}} \frac{1}{n} \|\vec{y} - X\vec{w}\|_2^2 + \lambda \sum_{j=1}^{p} w_j^2$$

For each of the following problems, fill in the blank.

Problem 5.1

If we set $\lambda = 0$, then $\|\vec{w}^*_{\text{OLS}}\|_2^2$ is ____ $\|\vec{w}^*_{\text{ridge}}\|_2^2$

- less than
- equal to
- greater than
- impossible to tell

length of $\vec{w}^*_{\text{OLS}}$, squared

same length because same vector!

Problem 5.2

For each of the remaining parts, you can assume that λ is set such that the predicted response vectors for our two models ($\vec{h} = X\vec{w}^*_{\text{OLS}}$ and $\vec{h} = X\vec{w}^*_{\text{ridge}}$) is different.

$\lambda > 0$

The **training** MSE of the model $\vec{h} = X\vec{w}^*_{\text{OLS}}$ is $<$ than the model $\vec{h} = X\vec{w}^*_{\text{ridge}}$.

- less than
- equal to
- greater than
- impossible to tell

Why? $\vec{w}^*_{\text{OLS}}$ is chosen to minimize MSE

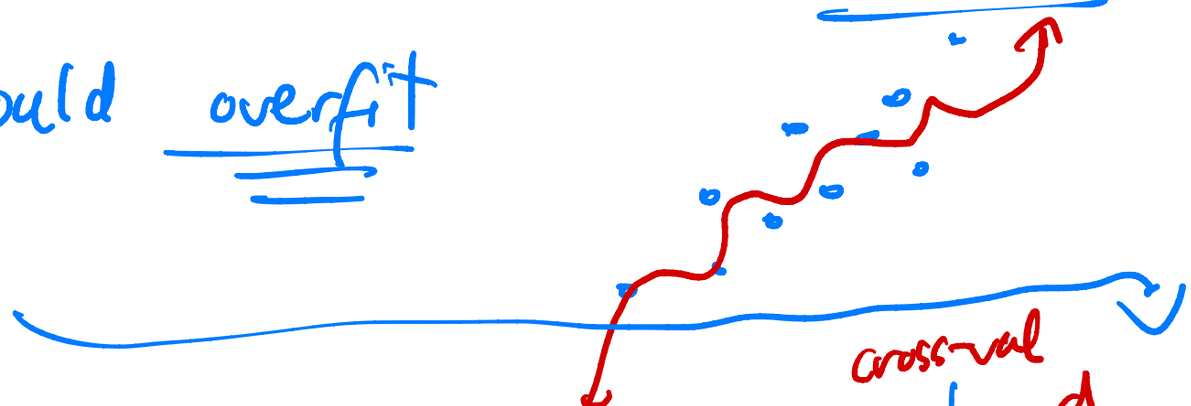
Regularization

find $\vec{w}^*$ that minimizes $\frac{1}{n}\|\vec{y} - X\vec{w}\|^2$

no regularization:
"ordinary least squares" (OLS)

$\frac{1}{n}\|\vec{y} - X\vec{w}\|^2$ mean squared error on training set

$\Rightarrow$ could overfit



LASSO: L_1 regularization
(L_2 regularization)

ridge regression:

find $\vec{w}^*$ that minimizes $\frac{1}{n}\|\vec{y} - X\vec{w}\|^2 + \lambda \sum_{j=1}^{d} w_j^2$

cross-val

$\frac{1}{n}\|\vec{y} - X\vec{w}\|^2$: MSE

$\lambda \sum_{j=1}^{d} w_j^2$: penalty on size of w_s

Problem 5.3

Now, assume we've fit both models using our training data, and evaluate both models on some unseen testing data.

The **test** MSE of the model $\vec{h} = X\vec{w}^*_{\text{OLS}}$ is ____ than the model $\vec{h} = X\vec{w}^*_{\text{ridge}}$.

- less than
- equal to
- greater than
- impossible to tell

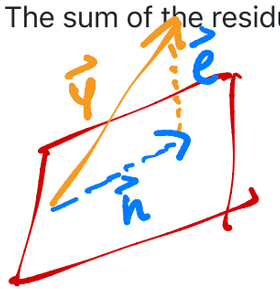
likely worse on unseen data, because likely overfit on training data

⇒ but we don't know for sure!

Problem 5.4

Assume that our design matrix X contains a column of all ones. The sum of the residuals of our model $\vec{h} = X\vec{w}^*_{\text{ridge}}$ ____.

- equal to 0
- not necessarily equal to 0



→ error vector, $\vec{e}$ orthogonal to span(X)

→ true without regularization, with regularization, unclear

Problem 5.5

As we increase λ, the bias of the model $\vec{h} = X\vec{w}^*_{\text{ridge}}$ tends to ____.

- increase

- stay the same
- decrease

Problem 5.6

As we increase λ, the model variance of the model $\vec{h} = X\vec{w}^*_{\text{ridge}}$ tends to ____.

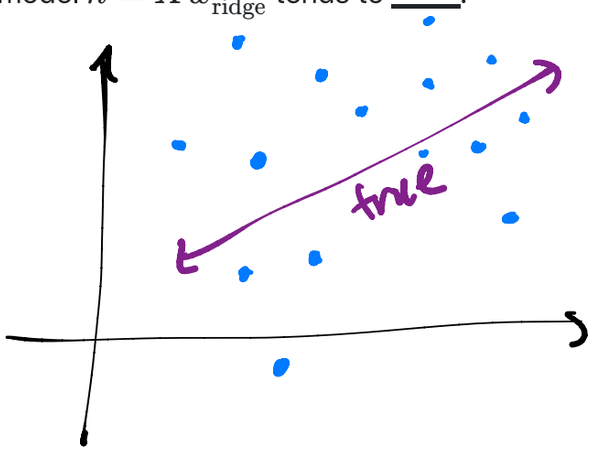
- increase
- stay the same
- decrease

Problem 5.7

As we increase λ, the observation variance of the model $\vec{h} = X\vec{w}^*_{\text{ridge}}$ tends to ____.

- increase
- stay the same
- decrease

you can't change the data



Problem 6

Neerad wants to build a model that predicts the number of open rooms a hotel has, given various other features. He has a training set with 1200 rows available to him for the purposes of training his model.

Cross-validation

hyperparameter: something about the model that we choose, e.g.
- polynomial degree
- k in k-nn classification
- decision tree depth...
- λ in regularization

cross-validation helps us choose hyperparameters without looking at the test data!

k-fold cross-val

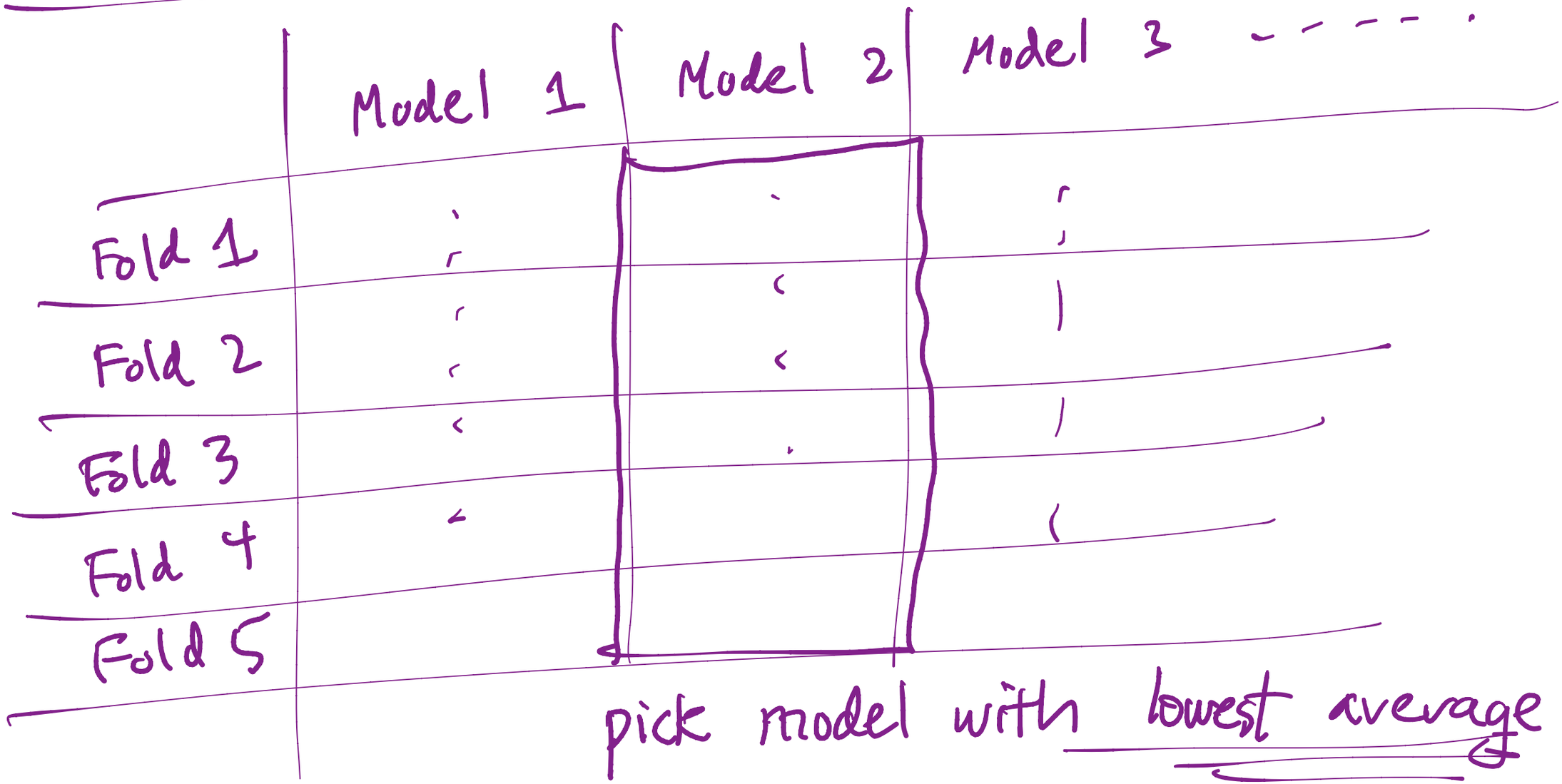
for every model I'm considering, I'll train it 5x

① treat f2 ⊕ f3 ⊕ f4 ⊕ f5 as training; compute validation error on f1

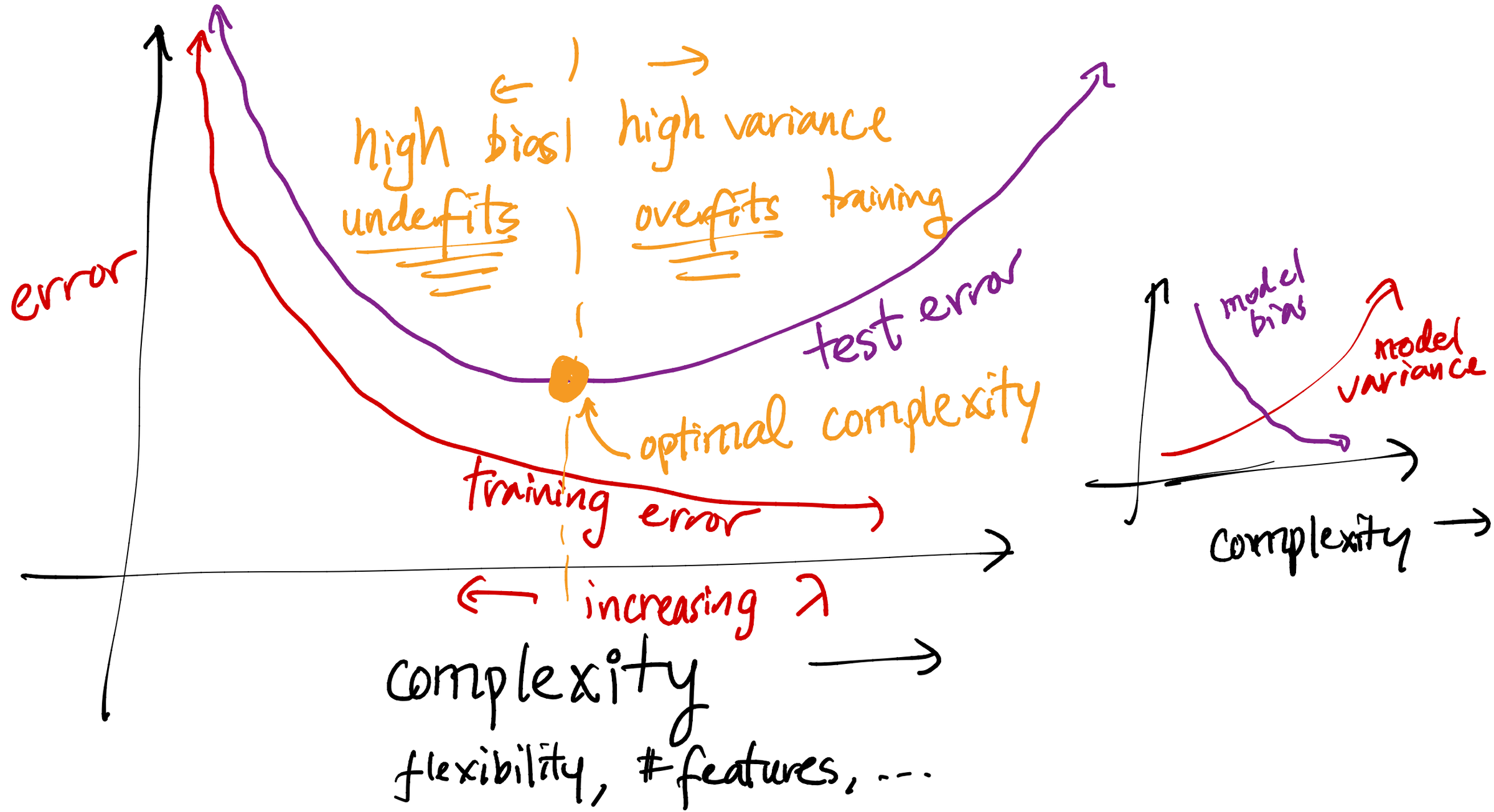
② treat f1 ⊕ f3 ⊕ f4 ⊕ f5 as training; compute validation error on f2

5 validation errors per model:

5 validation errors per model



pick model with lowest average validation error


high bias
high variance
underfits
overfits
training
error
test error
optimal complexity
training error
increasing λ
complexity
flexibility, #features, ...
model bias
model variance
complexity

Problem 6.1

Neerad fits a regression model using the `GPTRegression` class. `GPTRegression` models have several hyperparameters that can be tuned, including `context_length` and `sentience`.

To choose between 5 possible values of the hyperparameter `context_length`, Neerad performs k-fold cross-validation.

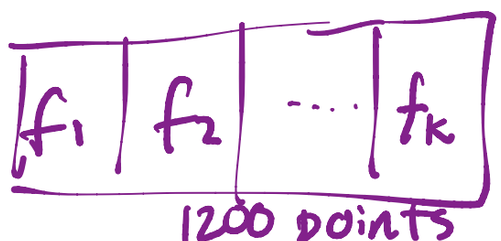
1. How many total times is a `GPTRegression` model fit?

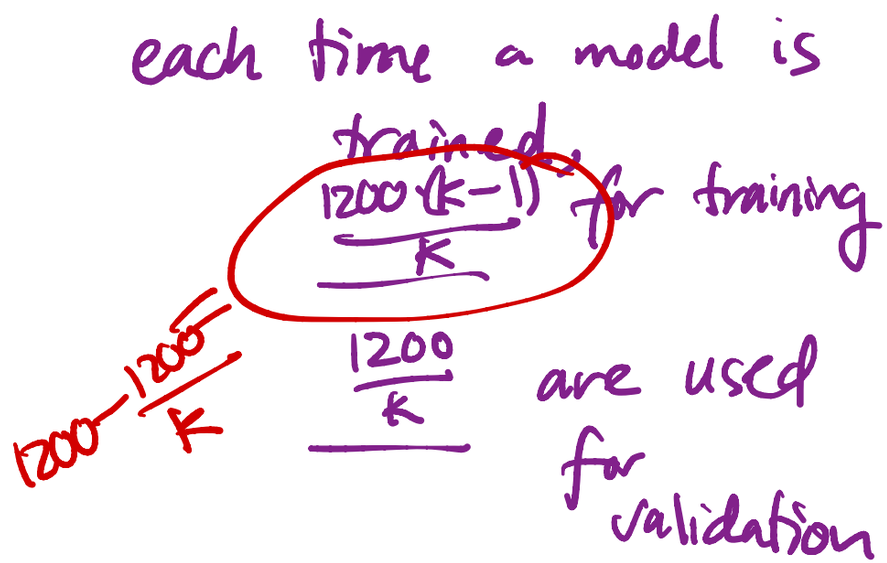
   - $4k$
   - $5k$
   - $240k$
   - $6000k$
   - $4(k-1)$
   - $5(k-1)$
   - $240(k-1)$
   - $6000(k-1)$

2. Suppose that every time a `GPTRegression` model is fit, it appends the number of points in its training set to the list `sizes`. Note that after performing cross- validation, `len(sizes)` is equal to your answer to the previous subpart.

What is `sum(sizes)`?

- $4k$





- $5k$
- $240k$
- $6000k$
- $4(k-1)$
- $5(k-1)$
- $240(k-1)$
- $6000(k-1)$

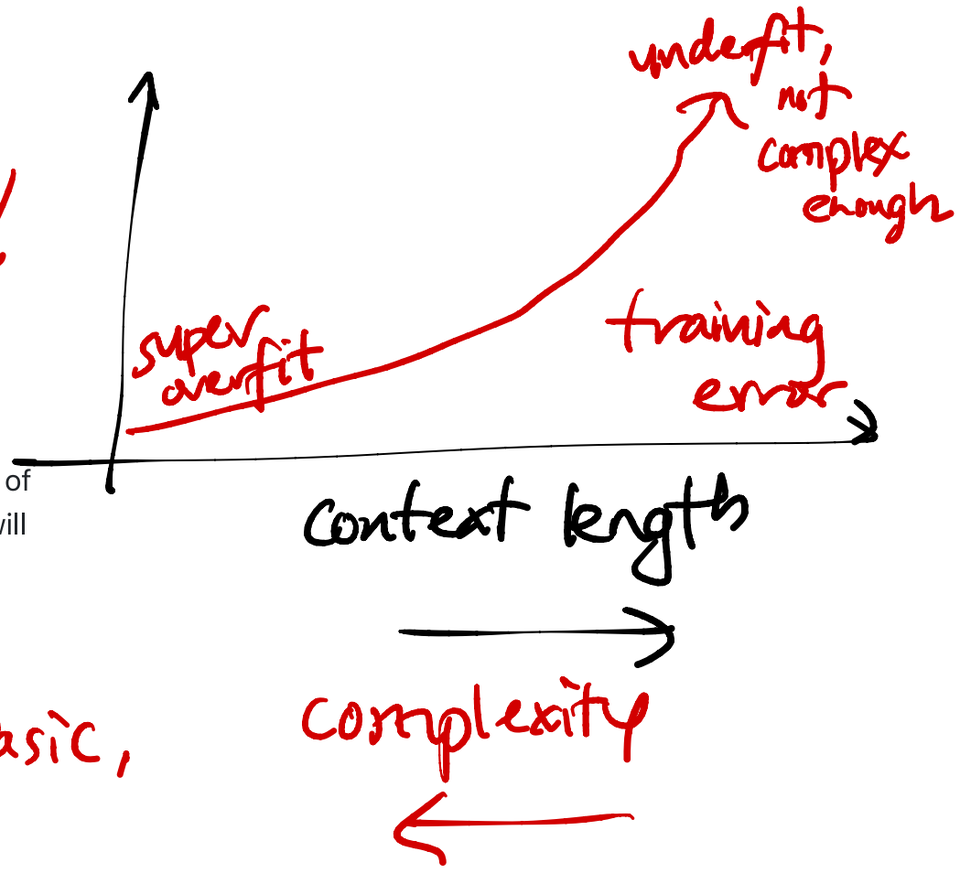
$\text{sum(sizes)} = \underbrace{5k}_{\#\text{ times model is trained}} \cdot \underbrace{\frac{1200(k-1)}{k}}_{\text{number of points used each time a model is trained}}$

$= 5(1200)(k-1)$

Problem 6.2

The average training error and validation error for all 5 candidate values of `context_length` are given below.

context_length	Mean Training RMSE	Mean Validation RMSE
0.1	1.7	100.9
1	4.3	88.2
10	12.2	34.7
100	17.4	21.5
1000	25.8	49.6



Fill in the blanks: As `context_length` increases, model complexity `__(i)__`. The optimal choice of `context_length` is `__(ii)__`; if we choose a `context_length` any higher than that, our model will `__(iii)__`.

1. What goes in blank (i)?

- increases
- decreases

2. What goes in blank (ii)?

- 0.1
- 1
- 10
- 100

context length ↑
model gets more basic,
more bias,
less variance,
less complexity

- 1000

3. What goes in blank (iii)?

- overfit the training data and have high bias
- underfit the training data and have high bias
- overfit the training data and have low bias
- underfit the training data and have low bias

if context length > 100, model is too basic

Problem 7

Suppose we want to use logistic regression to classify whether a person survived the sinking of the Titanic. The first 5 rows of our dataset are given below.

	Age	Survived	Female
0	22.0	0	0
1	38.0	1	1
2	26.0	1	1
3	35.0	1	1
4	35.0	0	0

$P(y=1 \mid \vec{x}) = \sigma(\vec{\omega} \cdot \text{Aug}(\vec{x}))$

class 1

logistic function "sigmoid"

Suppose after training our logistic regression model we get $\vec{w}^* = \begin{bmatrix} -1.2 \\ -0.005 \\ 2.5 \end{bmatrix}$, where -1.2 is an intercept term, -0.005 is the optimal parameter corresponding to passenger's age, and 2.5 is the optimal parameter corresponding to sex (1 if female, 0 otherwise).

Problem 7.1

Consider Sīlānah Iskandar Nāsīf Abī Dāghir Yazbak, a 20 year old female. What chance did she have to survive the sinking of the Titanic according to our model? Give your answer as a probability in terms of σ. If there is not enough information, write "not enough information."

Problem 7.2

Sīlānah Iskandar Nāsīf Abī Dāghir Yazbak actually survived. What is the cross-entropy loss for our prediction in the previous part?

Problem 7.3

At what age would we predict that a female passenger is more likely to have survived the Titanic than not? In other words, at what age is the probability of survival for a female passenger greater than 0.5?

see solutions

Hint: Since $\sigma(0) = 0.5$, we have that $\sigma\left(\vec{w}^* \cdot \text{Aug}(\vec{x})\right) = 0.5 \implies \vec{w}^* \cdot \text{Aug}(\vec{x}) = 0$.

Problem 7.4

Let m be the **odds** of a given non-female passenger's survival according to our logistic regression model, i.e., if the passenger had an 80% chance of survival, m would be 4, since their odds of survival are $\frac{0.8}{0.2} = 4$.

see solutions

It turns out we can compute f, the odds of survival for a female passenger of the same age, in terms of m. Give an expression for f in terms of m.

$$\vec{w}^* = \begin{bmatrix} -1.2 \\ -0.005 \\ 2.5 \end{bmatrix} \qquad \vec{x} = \begin{bmatrix} 20 \\ 1 \end{bmatrix}$$

$$\sigma(-1.2 + 20(-0.005) + 1(2.5)) = \sigma(1.2)$$

$$L_{ce}(y_i, p_i) = -y_i \log(p_i) - (1 - y_i)\log(1 - p_i)$$

$y_i = 0$ or 1

$$L_{ce}(1, \sigma(1.2)) = -\log(\sigma(1.2))$$

Problem 8

We decide to build a classifier that takes in a state's demographic information and predicts whether, in a given year:

- The state's mean math score was greater than its mean verbal score (1), or
- the state's mean math score was less than or equal to its mean verbal score (0).

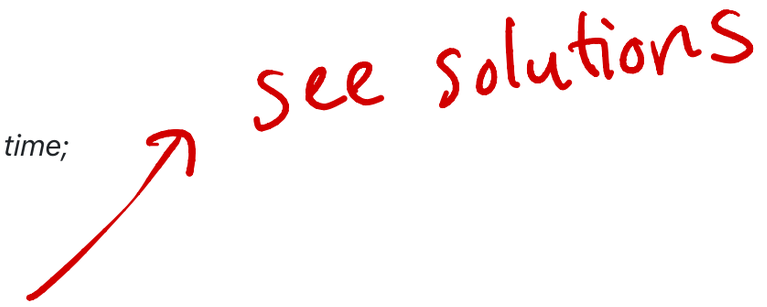
Problem 8.1

The simplest possible classifier we could build is one that predicts the same label (1 or 0) every time, independent of all other features.

Consider the following statement:

If `a > b`, then the constant classifier that maximizes training accuracy predicts 1 every time; otherwise, it predicts 0 every time.

For which combination of `a` and `b` is the above statement **not guaranteed** to be true?



Note: Treat `sat` as our training set.

Option 1:

```
a = (sat['Math'] > sat['Verbal']).mean()
b = 0.5
```

Option 2:

```
a = (sat['Math'] - sat['Verbal']).mean()
b = 0
```

Option 3:

```
a = (sat['Math'] - sat['Verbal'] > 0).mean()
b = 0.5
```

Option 4:

```
a = ((sat['Math'] / sat['Verbal']) > 1).mean() - 0.5
b = 0
```

- Option 1
- Option 2
- Option 3
- Option 4

see solution

Problem 8.2

Suppose we train a classifier, named Classifier 1, and it achieves an accuracy of $\frac{5}{9}$ on our training set.

Typically, root mean squared error (RMSE) is used as a performance metric for regression models, but mathematically, nothing is stopping us from using it as a performance metric for classification models as well.

What is the RMSE of Classifier 1 on our training set? Give your answer as a **simplified fraction**.

Problem 8.3

While Classifier 1's accuracy on our training set is $\frac{5}{9}$, its accuracy on our test set is $\frac{1}{4}$. Which of the following scenarios is most likely?

- Classifier 1 overfit to our training set; we need to increase its complexity.
- Classifier 1 overfit to our training set; we need to decrease its complexity.
- Classifier 1 underfit to our training set; we need to increase its complexity.
- Classifier 1 underfit to our training set; we need to decrease its complexity.

For the remainder of this question, suppose we train another classifier, named Classifier 2, again on our training set. Its performance on the training set is described in the confusion matrix below. Note that the columns of the confusion matrix have been separately normalized so that each has a sum of 1.

	Actually 0	Actually 1
Predicted 0	0.9	0.4
Predicted 1	0.1	0.6

definition of recall = TP / (TP+FN)

axis=0

Problem 8.4

Suppose `conf` is the DataFrame above. Which of the following evaluates to a Series of length 2 whose only unique value is the number `1`?

- `conf.sum(axis=0)`
- `conf.sum(axis=1)`

that are actually positive

Problem 8.5

Fill in the blank: the ___ of Classifier 2 is guaranteed to be 0.6.

- precision
- recall

see previous slide

	Actually 0	Actually 1
Pred 0	0.9B	0.4A
Pred 1	0.1B	0.6A

For your convenience, we show the column-normalized confusion matrix from the previous page below. You will need to use the specific numbers in this matrix when answering the following subpart.

	Actually 0	Actually 1
Predicted 0	0.9	0.4
Predicted 1	0.1	0.6

B values actually 0

A values actually 1

$$\text{accuracy} = \frac{0.9B + 0.6A}{0.9B + 0.1B + 0.4A + 0.6A}$$

$$= \frac{0.9B + 0.6A}{B + A}$$

$1-\alpha$ α

$$= 0.9(1-\alpha) + 0.6\alpha$$

Problem 8.6

Suppose a fraction α of the labels in the training set are actually 1 and the remaining $1 - \alpha$ are actually 0. The accuracy of Classifier 2 is 0.65. What is the value of α?

Hint: If you're unsure on how to proceed, here are some guiding questions:

- Suppose the number of y-values that are actually 1 is A and that the number of y-values that are actually 0 is B. In terms of A and B, what is the accuracy of Classifier 2? Remember, you'll need to refer to the numbers in the confusion matrix above.
- What is the relationship between A, B, and α? How does it simplify your calculation for the accuracy in the previous step?

Problem 9

Let $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$. Consider the function $g(\vec{x}) = (x_1 - 3)^2 + (x_1^2 - x_2)^2$.

Problem 9.1

Find $\nabla g(\vec{x})$, the gradient of $g(\vec{x})$, and use it to show that $\nabla g\left(\begin{bmatrix} -1 \\ 1 \end{bmatrix}\right) = \begin{bmatrix} -8 \\ 0 \end{bmatrix}$.

treat x_2 as const.

$$\frac{\partial g}{\partial x_1} = 2(x_1 - 3) + 2(x_1^2 - x_2)\,2x_1$$

$$\nabla g(\vec{x}) = \begin{bmatrix} \frac{\partial g}{\partial x_1} \\ \frac{\partial g}{\partial x_2} \end{bmatrix}$$

$$\frac{\partial g}{\partial x_2} = 2(x_1^2 - x_2)(-1)$$

$$\nabla g(\vec{x}) = \begin{bmatrix} 2(x_1 - 3) + 4x_1(x_1^2 - x_2) \\ -2(x_1^2 - x_2) \end{bmatrix}$$

Problem 9.2

We'd like to find the vector $\vec{x}^*$ that minimizes $g(\vec{x})$ using gradient descent. Perform one iteration of gradient descent by hand, using the initial guess $\vec{x}^{(0)} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$ and the learning rate $\alpha = \frac{1}{2}$. In other words, what is $\vec{x}^{(1)}$?

In general:

$$\vec{x}^{(t+1)} = \vec{x}^{(t)} - \alpha \nabla g(\vec{x}^{(t)})$$

Here:

$$\vec{x}^{(1)} = \vec{x}^{(0)} - \frac{1}{2} \nabla g(\vec{x}^{(0)})$$

$$= \begin{bmatrix} -1 \\ 1 \end{bmatrix} - \frac{1}{2} \begin{bmatrix} -8 \\ 0 \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$$

$\vec{x}^{(1)}$

Problem 9.3

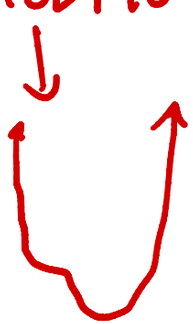
Consider the function $f(t) = (t - 3)^2 + (t^2 - 1)^2$. Select the true statement below.

$= t^2 - 6t + 9 + t^4 - 2t^2 + 1$

$= t^4 - t^2 - 6t + 10$

- $f(t)$ is convex and has a global minimum.
- $f(t)$ is not convex, but has a global minimum.
- ~~$f(t)$ is convex, but doesn't have a global minimum.~~
- ~~$f(t)$ is not convex and doesn't have a global minimum.~~

need second derivative to always be ≥ 0.



What's the point: gradient descent?

imagine this is a graph of average loss

GD finds the minimum of a function

⟹ finds the optimal parameters that minimize average loss!

👋 Feedback: Find an error? Still confused? Have a suggestion? Let us know here.