Data 100 Midterm 2 Review Session

Suraj Rampure, Neil Shah, Allen Shen

Slides credit: Sona Jeswani, Josh Hug, Joseph Gonzalez

Review Session Format/Overview

5 Sections with Practice Problems

- Probability: Expectation, Variance, Covariance
- Modeling/Gradient Descent: L1 vs. L2 Loss, Gradient Descent Update Equation, Stochastic/Batch Gradient Descent
- Linear Regression: Normal Equations (and Ridge Regression Variant), Feature Engineering, Least Squares Geometric Intuition
    - Bias Variance Tradeoff/Regularization: Bias Variance Decomposition, Tradeoff Curves, (K-Fold) Cross Validation, LASSO vs. Ridge
- Logistic Regression/Classification: Precision vs. Recall (TP, TN, FP, FN), Linear Separability, Cross Entropy Loss, Sigmoid Function

Disclaimer

- This will not be a comprehensive coverage of topics.
- Concepts covered in this review session may or may not appear on the exam, and concepts on the exam may or may not appear in this review session.
- This review session is meant to rehash concepts from lecture and enhance your understanding on those concepts.

Notation Changes

- In past semesters, Φ was used to represent the feature matrix given data records
- This semester, we are using X to represent the feature matrix
    - In some of the slides and problems, Φ is used, so just replace it with X

Probability

Probability Overview

- The **expectation** (ie. expected value) $E(X)$ of a random variable X is the average.
- The variance $var(X)$ is a measure of spread.

$$E(X) = \mu = \sum_{x \in \mathbb{X}} x \cdot P(X = x)$$

$$var(X) = E[(X - E(X))^2] = E(X^2) - E(X)^2$$

$$SD(X) = \sqrt{var(X)}$$

Covariance

$$Cov(X, Y) = E[XY] - E[X]E[Y]$$

Properties of Expectation and Variance

Linearity of Expectation

$$E[X+Y] = E[X] + E[Y]$$
$$E[aX] = aE[X]$$

Properties of Variance

$$var(X+Y) = var(X) + var(Y) + 2cov(X,Y)$$

$$var(aX+b) = a^2 var(X)$$

If X, Y are independent:

$$cov(X,Y) = 0 \Rightarrow E[XY] = E[X]E[Y]$$

$$var[X+Y] = var(X) + var(Y)$$

Suppose we model students' grades using

$$Y = X\beta + \epsilon$$

where X, β are (non-random) constants, and ϵ is a random variable such that $E[\epsilon] = 0$ and $var[\epsilon] = \sigma^2$.

a) What is $E[Y]$?

b) What is $var[Y]$?

Suppose we model students' grades using

$$Y = X\beta + \epsilon$$

where X, β are (non-random) constants, and ϵ is a random variable such that $E[\epsilon] = 0$ and $var[\epsilon] = \sigma^2$.

The key to this problem is realizing that $X\beta$ is just a constant. The expectation of a constant is the constant itself, and the variance of a constant is 0.

a) What is $E[Y]$?

$E[Y] = E[X\beta + \epsilon] = E[X\beta] + E[\epsilon] = X\beta$

b) What is $var[Y]$?

$var[Y] = \sigma^2$, since $var[X\beta] = 0$

8. Suppose the random variable X can take on values -1, 0, and 1 with chance p^2, $2p(1-p)$ and $(1-p)^2$, respectively, for $0 \le p \le 1$.

   What is the expected value of X?

   A. $2p(1-p)$

   B. $p^2(1-p)^2$

   C. 0

   D. $1-2p$

   E. 1

$$E[X] = \sum_{x \in \mathcal{X}} x \cdot P(X=x)$$

6. Suppose X, Y, and Z are random variables that are independent and have the same probability distribution. If $\mathrm{Var}(X) = \sigma^2$, then $\mathrm{Var}(X+Y+Z)$ is:

   A. $9\sigma^2$

   B. $3\sigma^2$

   C. σ^2

   D. $\frac{1}{3}\sigma^2$

8. Suppose the random variable X can take on values -1, 0, and 1 with chance p^2, $2p(1-p)$ and $(1-p)^2$, respectively, for $0 \leq p \leq 1$.

   What is the expected value of X?

   A. $2p(1-p)$

   B. $p^2(1-p)^2$

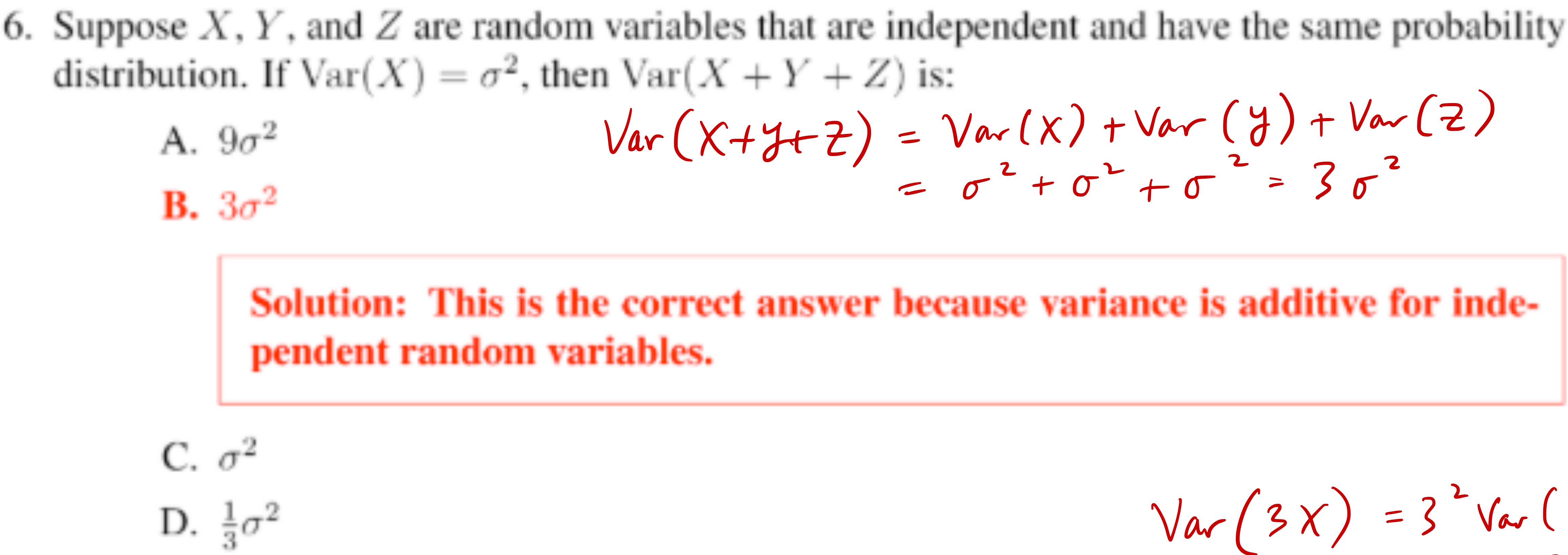
   C. 0

   D. $1-2p$

   E. 1

Solution: The expected value of X is

$$\begin{aligned} E(X) &= \sum_{i=1}^{m} v_i P(X = v_i) \\ &= -1\overset{-p^2}{P(X = -1)} + 0P(X = 0) + 1\underset{(1-p)^2}{P(X = 1)} \\ &= -p^2 + (1-p)^2 \\ &= 1 - 2p \end{aligned}$$



6. Suppose X, Y, and Z are random variables that are independent and have the same probability distribution. If $\mathrm{Var}(X) = \sigma^2$, then $\mathrm{Var}(X + Y + Z)$ is:

   A. $9\sigma^2$

   B. $3\sigma^2$

   Solution: This is the correct answer because variance is additive for independent random variables.

   C. σ^2

   D. $\frac{1}{3}\sigma^2$

$$\mathrm{Var}(X+Y+Z) = \mathrm{Var}(X) + \mathrm{Var}(Y) + \mathrm{Var}(Z)$$
$$= \sigma^2 + \sigma^2 + \sigma^2 = 3\sigma^2$$

$$\mathrm{Var}(3X) = 3^2\,\mathrm{Var}(X)$$
$$= 9\,\mathrm{Var}(X)$$

but $X+Y+Z \neq 3X$

Modeling / Gradient Descent

Loss Functions and Risk

A **loss function** is a function that characterizes the cost, error, or loss resulting from a particular choice of model or model parameters.

- **Loss Function:** Measures loss for a particular observation.
- **Empirical Risk:** Average loss on the training set of observations.

The choice of loss function **depends on the estimation task**. Questions to consider:

- Qualitative or quantitative predictions?
- Sensitivity to outliers
- The cost of error (how bad are false negatives?)
- Does the loss function have an analytical solution?

Common Loss Functions

Suppose y represents the true value of a variable, and $\hat{y}$ represents our predicted value.

L1 Loss

$$L_1(y, \hat{y}) = |y - \hat{y}|$$

L2 Loss

$$L_2(y, \hat{y}) = (y - \hat{y})^2$$

Huber Loss

$$L_\alpha(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & |y - \hat{y}| \le \alpha \\ \alpha\left(|y - \hat{y}| - \frac{1}{2}\alpha\right), & \text{else} \end{cases}$$

Minimizing Risk

- Finding the parameter values that minimize the empirical risk on training data
- Generally, we are interested in convex loss functions (as they have global minimums)
- To find the analytical solution, we perform our favorite procedure:

```
1. Take the derivative/gradient
2. Set equal to zero
3. Solve for optimal parameters
```

Problem: Many loss functions do not have an analytical solution! In other words, we cannot solve for our optimal parameters mathematically... what do we do?

Gradient Descent Intuition

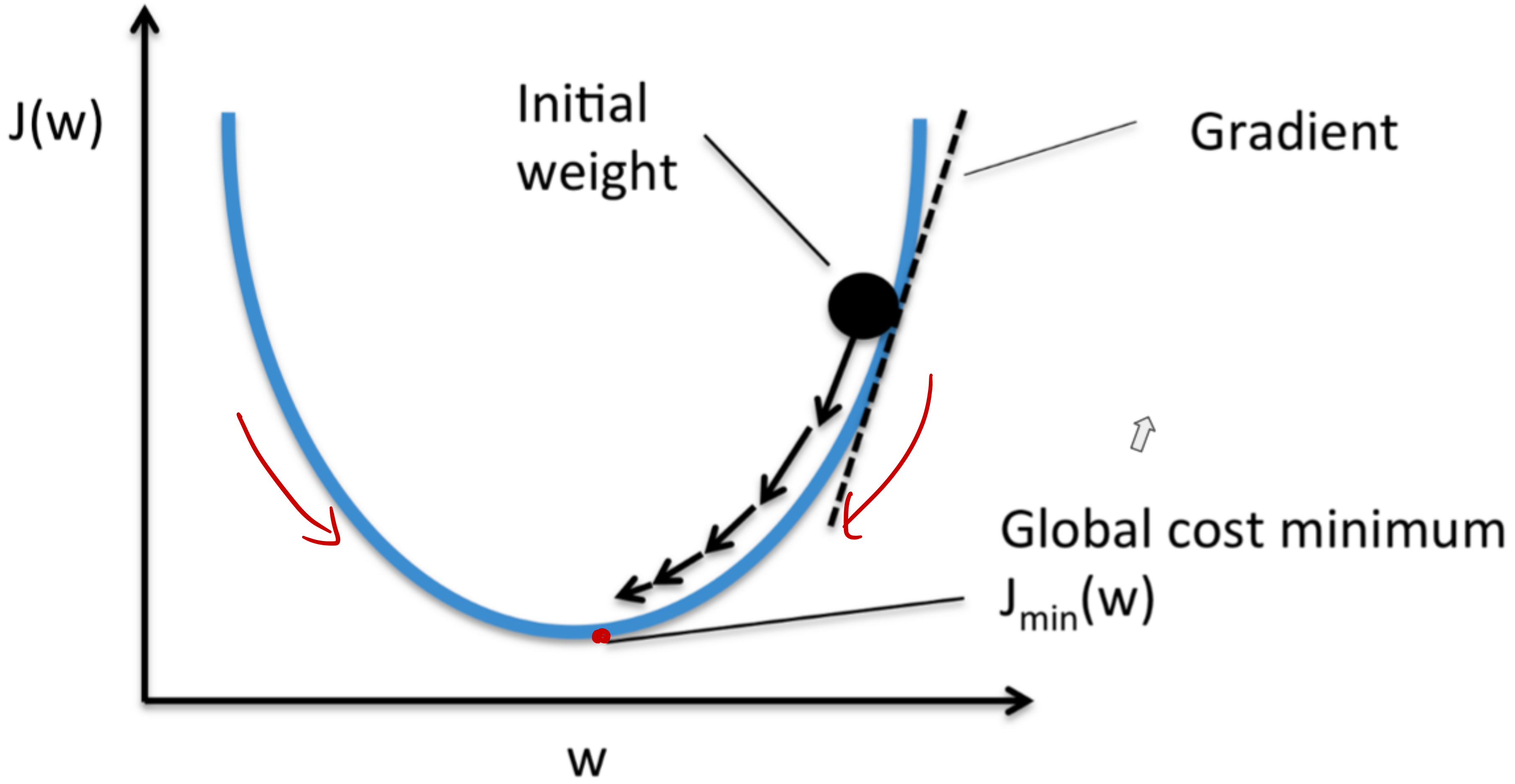
- Gradient Descent is an algorithm to find iteratively find the parameters/weights that minimize the risk.
- The gradient $\nabla L(\beta)$ points in direction of greatest increase or steepest ascent.
- So $-\nabla L(\beta)$ (negative gradient) moves along direction of steepest descent.

Iterative optimization: given a candidate minimizer β^t at step t, we want an improved version β^{t+1}.

Update Rule:

$$\boxed{\beta^{t+1} = \beta^t - \alpha \nabla L(\beta^t)}$$

where α is the *learning rate* which determines how large the steps are.



Considerations:

- What if α is too big? What if α is too small?
- What's the risk of running GD on a non-convex function?
- How computationally efficient is computing $\nabla L(\beta^t)$? This uses all data points.

Variations of Gradient Descent

- Standard (batch) gradient descent: Use all n data points when computing the gradient $\nabla L(\beta^t)$
- Stochastic gradient descent: Randomly choose 1 single data point and compute the gradient $\nabla L(\beta^t)$ only using this data point
  - More computationally efficient than batch GD
  - This approximates the standard gradient descent
- Mini-batch gradient descent: Randomly choose b points to compute gradient on
  - More computationally efficient than batch GD.

$$L(\beta) = \sum_{i=1}^{\boxed{n}} L(y_i, \hat{y}_i)$$

for standard GD

stochastic : don't sum, just take single pt

mini-batch : choose some subset of b points,
and sum over those points

25. Let $x_1, \ldots, x_n$ denote any collection of numbers with average $\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$.

(a) [3 Pts] $\sum_{i=1}^{n}(x_i - \overline{x})^2 \leq \sum_{i=1}^{n}(x_i - c)^2$ for all c.

☑ True ○ False

$$f(c) = \sum_{i=1}^{n} (x_i - c)^2$$

$$\frac{\partial}{\partial c} f(c) = \sum_{i=1}^{n} 2(x_i - c)(-1) = 0$$

$$-2\sum_{i=1}^{n}(x_i - c) = 0$$

$$\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} c = 0$$

$$\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} c = nc$$

$$\boxed{c = \frac{\sum_{i=1}^{n} x_i}{n}}$$

$$\therefore \ c = \overline{x}$$

minimizes $\sum_{i=1}^{n} (x_i - c)^2$

(b) [3 Pts] $\sum_{i=1}^{n} |x_i - \overline{x}| \leq \sum_{i=1}^{n} |x_i - c|$ for all c.

○ True ✓ False

Minimizing value of c here is median (x)

Consider the following loss function based on data $x_1, \ldots, x_n$:

$$\ell(\mu, \sigma) = \log(\sigma^2) + \frac{1}{n\sigma^2}\sum_{i=1}^{n}(x_i - \mu)^2.$$

$$\frac{\partial \ell}{\partial \mu} = 0 + \frac{1}{n\sigma^2}\sum_{i=1}^{n} 2(x_i - \mu)^2 = 0$$

$$\Rightarrow 2\sum_{i=1}^{n}(x_i - \mu) = 0 \quad \longleftarrow \text{fixed}$$

from previous slides :

$$\boxed{\hat{\mu} = \frac{\sum_{i=1}^{n} x_i}{n}}$$

~~θ~~ B

25. [2 Pts] For this problem, recall that stochastic gradient descent is very similar to normal gradient descent, except that the gradient of the loss function is computed on a random sample of the data instead of the entire dataset. Which of the following are true? Select all that apply.

- □ A. At a particular iteration, stochastic gradient descent will often update θ more accurately compared to an update with regular gradient descent.
- □ B. For a convex loss function, a single step of gradient descent always decreases the loss.
- □ C. For a convex loss function, a single step of stochastic gradient descent always decreases the loss.
- ☑ D. Suppose it takes t seconds for one update of regular gradient descent, Stochastic gradient descent can usually perform more than one update in t seconds.
- □ E. None of the Above

Linear Regression

Gradients – Review

Suppose $a, x \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n\times n}$.

Recall the gradients of each of the following functions.

$$f(x) = a^T x$$

$$\Rightarrow \nabla_x f(x) = a$$

$$f(x) = x^T x$$

$$\Rightarrow \nabla_x f(x) = 2x$$

$$f(x) = x^T A x$$

$$\Rightarrow \nabla_x f(x) = (A + A^T)x$$

A symmetric

$= 2Ax$

Linear Regression in 2D

Suppose we're given $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, and want to fit a linear model $\underline{y = \beta_1 x + \beta_0,}$ using MSE (i.e. L2) loss.

Our objective function is

$$L(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \beta_1 x_i - \beta_0)^2$$

One way to solve: Take partial derivatives with respect to β_0, β_1. Solve for β_0 and β_1.

$$= \frac{1}{n}\left((y_1 - \beta_1 x_1 - \beta_0)^2 + (y_2 - \beta_1 x_2 - \beta_0)^2 + \ldots\ldots\right)$$

$$E[y \mid \underset{\sim}{X}] = X^T\beta$$

single point

Let's try and rewrite this in vector form.

$$\hat{y} = X\beta \qquad \begin{bmatrix} y_1 - \hat{y}_1 \\ \vdots \\ y_n - \hat{y}_n \end{bmatrix}$$

$$L(\beta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \beta_1 x - \beta_0)^2$$

$$y - X\beta = \begin{bmatrix} y_1 - \beta_0 - \beta_1 x_1 \\ \vdots \\ y_n - \beta_0 - \beta_1 x_n \end{bmatrix}$$

$$\Rightarrow L(\beta) = \frac{1}{n}\|y - X\beta\|_2^2$$

We can say the following:

$$\beta = \begin{bmatrix}\beta_0 & \beta_1\end{bmatrix}^T$$

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_n \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 & y_2 & \dots & y_n \end{bmatrix}^T$$

$$\underset{\sim}{X}\beta = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

$$= \begin{bmatrix} \beta_0 + \beta_1 x_1 \\ \beta_0 + \beta_1 x_2 \\ \vdots \\ \beta_0 + \beta_1 x_n \end{bmatrix} = y$$

Solution to Normal Equation, using Vector Calculus

$$L(\beta) = \frac{1}{n}||y - X\beta||_2^2 = \frac{1}{n}\left((y - X\beta)^T(y - X\beta)\right)$$

$$= \frac{1}{n}\left(y^T y - y^T X\beta - (X\beta)^T y - \beta^T X^T X\beta\right)$$

$$= y^T y - 2(X^T y)^T\beta - (X\beta)^T(X\beta)$$

Taking the gradient and setting it equal to 0:

$$\nabla L(\beta) = 0 - 2X^T y - 2X^T X\beta = 0$$

$$\Rightarrow X^T X\beta = X^T y$$

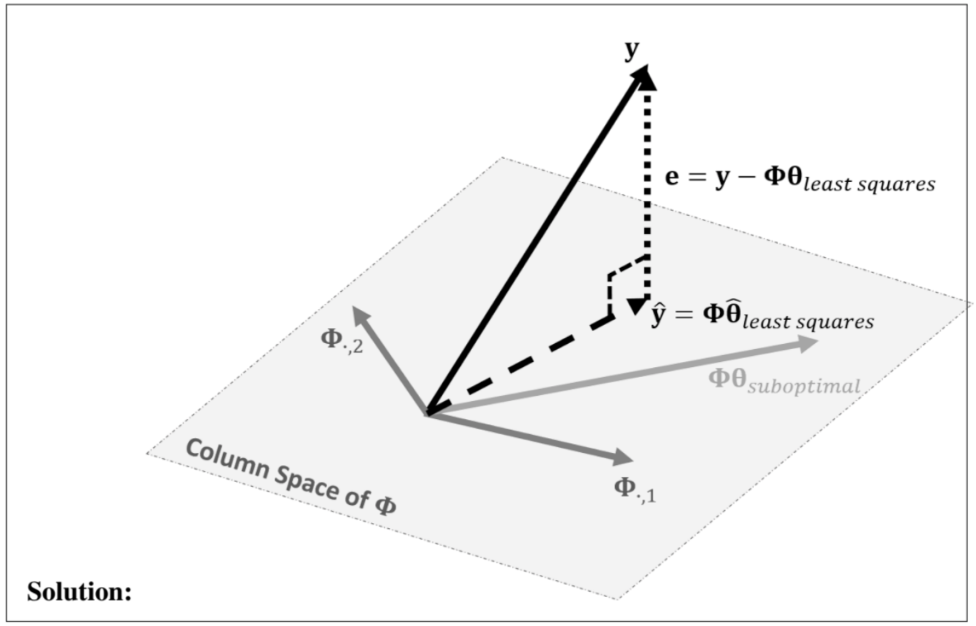
$$\Rightarrow \beta^* = (X^T X)^{-1}X^T y$$

$a^T b = a \cdot b$

Solution to Normal Equation, using Geometry

$e = y - X\beta$

$\underbrace{X^T(y - X\beta)} = 0$


$\mathbf{y}$
$\mathbf{e} = \mathbf{y} - \mathbf{\Phi\theta}_{least\ squares}$
$\hat{\mathbf{y}} = \mathbf{\Phi}\hat{\mathbf{\theta}}_{least\ squares}$
$\mathbf{\Phi\theta}_{suboptimal}$
$\mathbf{\Phi}_{:,2}$
$\mathbf{\Phi}_{:,1}$
Column Space of Φ
Solution:

We see that to minimize e, e must be orthogonal to the column space of X (or, in the picture, Φ).

- The Discussion 7 walkthrough talks about this in significant detail.

Regularization

$$L(\beta) = \frac{1}{n}\|y - X\beta\|_2^2$$

$$\beta^* = (X^TX)^{-1}X^Ty$$

Issues with OLS:

- Solution doesn't always exist (if X is not full-rank, X^TX will not be full rank)
- Numerical issues with inversions
- Potential overfitting to training set – model can be too complex

To fix: Add penalty on magnitude of β. However, we could use either the L2 norm, or L1 norm!

Using L2 (Ridge): $L(\beta) = \frac{1}{n}||y - X\beta||_2^2 + \lambda||\beta||_2^2$

Using L1 (LASSO): $L(\beta) = \frac{1}{n}||y - X\beta||_2^2 + \lambda||\beta||_1$

Note: $||x||_2 = \sqrt{x_1^2 + x_2^2 + ... + x_n^2}$, and $||x||_1 = |x_1| + |x_2| + ... + |x_n|$.

$$\|\beta\|_2^2 = \beta_1^2 + \beta_2^2 + \cdots + \beta_p^2$$

$$\beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{bmatrix}$$

Regularization – Ridge Regression

When we use the L2 vector norm for the penalty term, our objective function becomes

$$\min_{\beta} \frac{1}{n}||y - X\beta||_2^2 + \lambda||\beta||_2^2$$

This is called "ridge regression."

$$\lambda = 0$$

Solution can also be determined using vector calculus.

$$\Rightarrow \beta^*_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

- λ represents the penalty on the size of our model. We will discuss this more later in the review.
- Unlike OLS, Ridge Regression always has a unique solution!

Regularization – LASSO Regression

$$\Rightarrow \|\beta\|_1 = |\beta_1| + |\beta_2| + \cdots + |\beta_n|$$

$$\|\beta\|_2^2 = \beta_1^2 + \beta_2^2 + \cdots + \beta_n^2$$

When we use the L1 vector norm for the penalty term, our objective function becomes

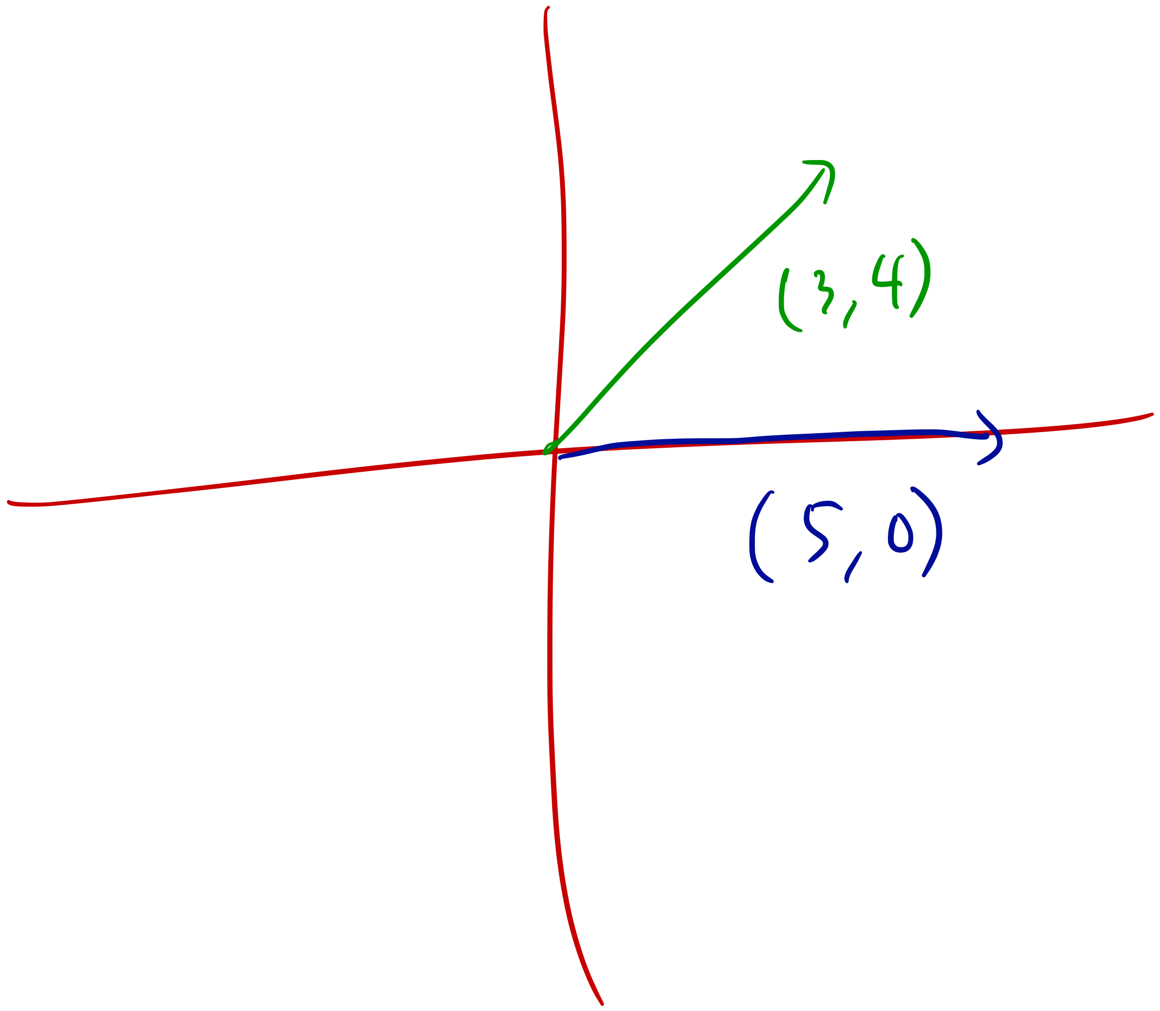
$$\min_{\beta} \frac{1}{n} ||y - X\beta||_2^2 + \lambda \boxed{||\beta||_1}$$

This is called "LASSO regression." *Fun fact: LASSO stands for Least Absolute Shrinkage and Selection Operator.*

Unlike OLS and Ridge Regression, there is (in general) no closed form solution. Need to use a numerical method, such as gradient descent.

- LASSO regression encourages sparsity, that is, it sets many of the entries in our β vector to 0. LASSO effectively selects features for us, and also makes our model less complex (many weights set to 0 ——> less features used ——> less complex)
- Again, λ represents the penalty on the size of our model.

L_1 vs. L_2 vector norms: $(3,4)$ vs. $(5,0)$



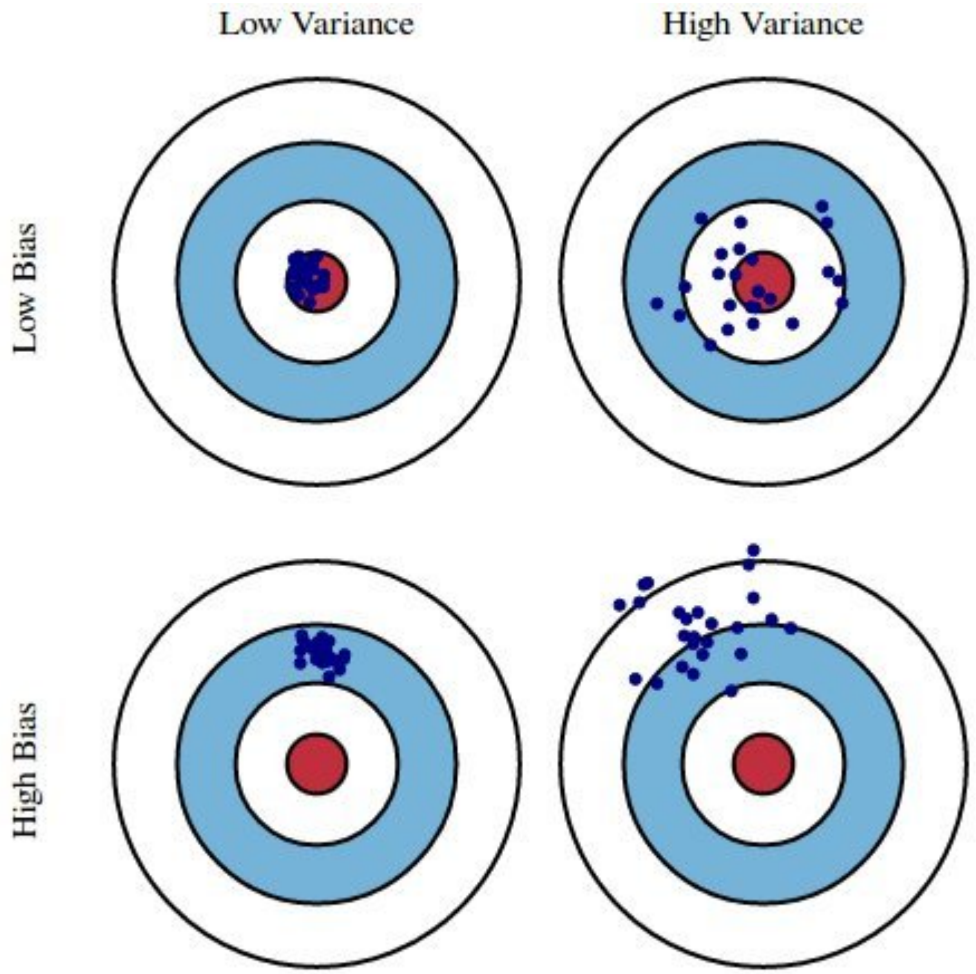
$$L_1(5,0) : |5|+|0|$$
$$= 5$$

$$L_2(5,0) = \sqrt{5^2+0^2}$$
$$= 5$$

$$L_2(3,4) = \sqrt{3^2+4^2}$$
$$= 5$$

$$L_1(3,4) = |3|+|4|$$
$$= 7$$

Bias-Variance



Bias-Variance Decomposition

Suppose ϵ is some random variable such that $\mathbb{E}[\epsilon] = 0$ and $var[\epsilon] = \sigma^2$. Also, suppose we have Y generated as follows:

e.g. $f_B(x) = \beta_0 + \beta_1 x$

$$Y = h(x) + \epsilon$$

We collect some sample points $\{(x_i, y_i)\}_{i=1}^n$, and want to fit a model $f_\beta(x)$. We define the model **risk** as $\mathbb{E}[(Y - f_\beta(x))^2]$.

$$\mathbb{E}[(Y - f_\beta(x))^2] = \underbrace{(h(x) - \mathbb{E}[f_\beta(x)])}_{\text{bias}^2})^2 + \mathbb{E}[\underbrace{(\mathbb{E}[f_\beta(x)] - f_\beta(x))}_{\text{model variance}}{}^2] + \underbrace{\sigma^2}_{\text{obs. error}}$$

This is sometimes referred to as the **bias-variance decomposition.**

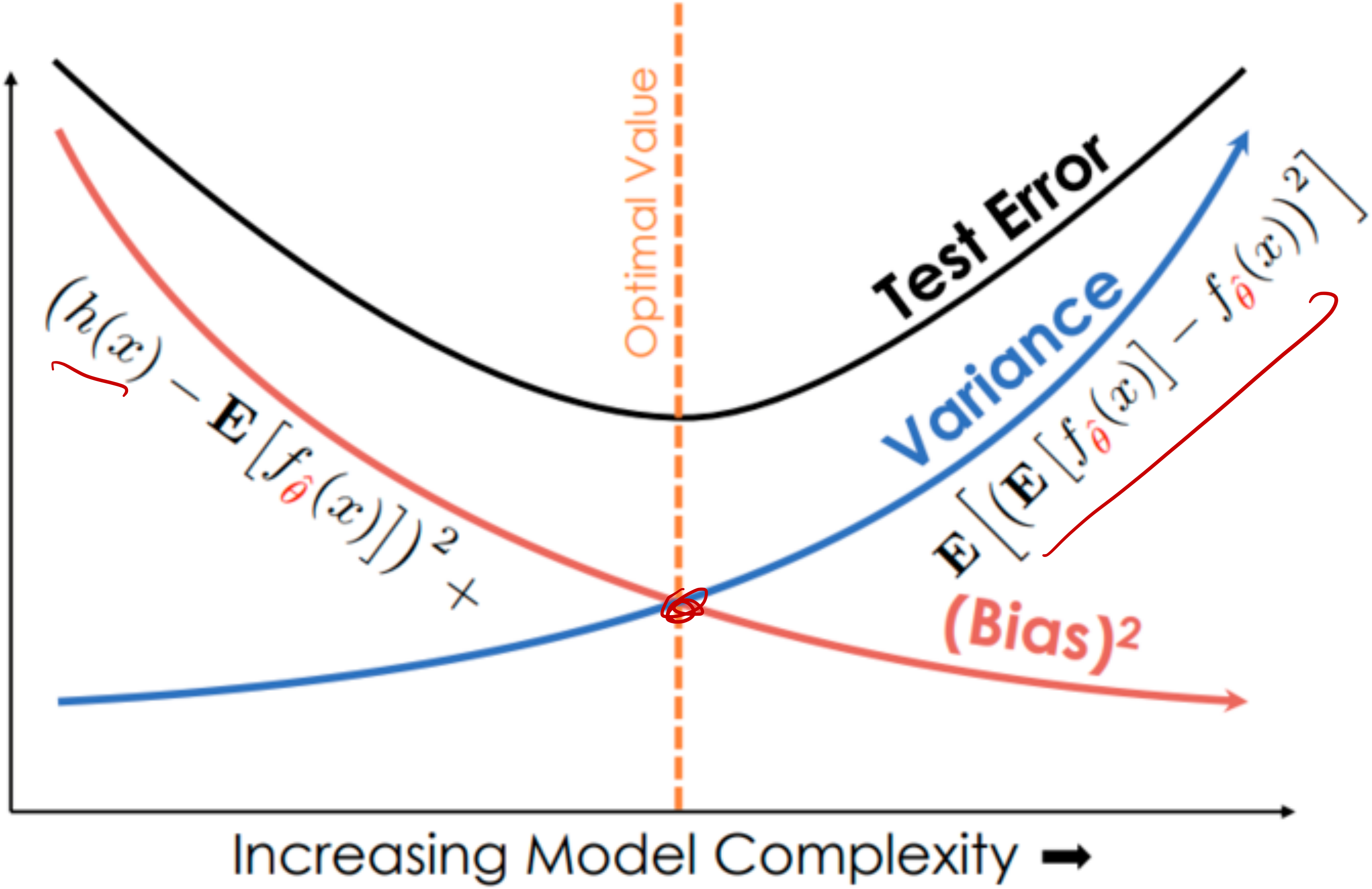
Let's analyze the objective function for ridge regression (however, the analysis is the same for

$$\min_{\theta} \frac{1}{n}\left|\left|y - X\beta\right|\right|_2^2 + \lambda\left|\left|\beta\right|\right|_2^2$$

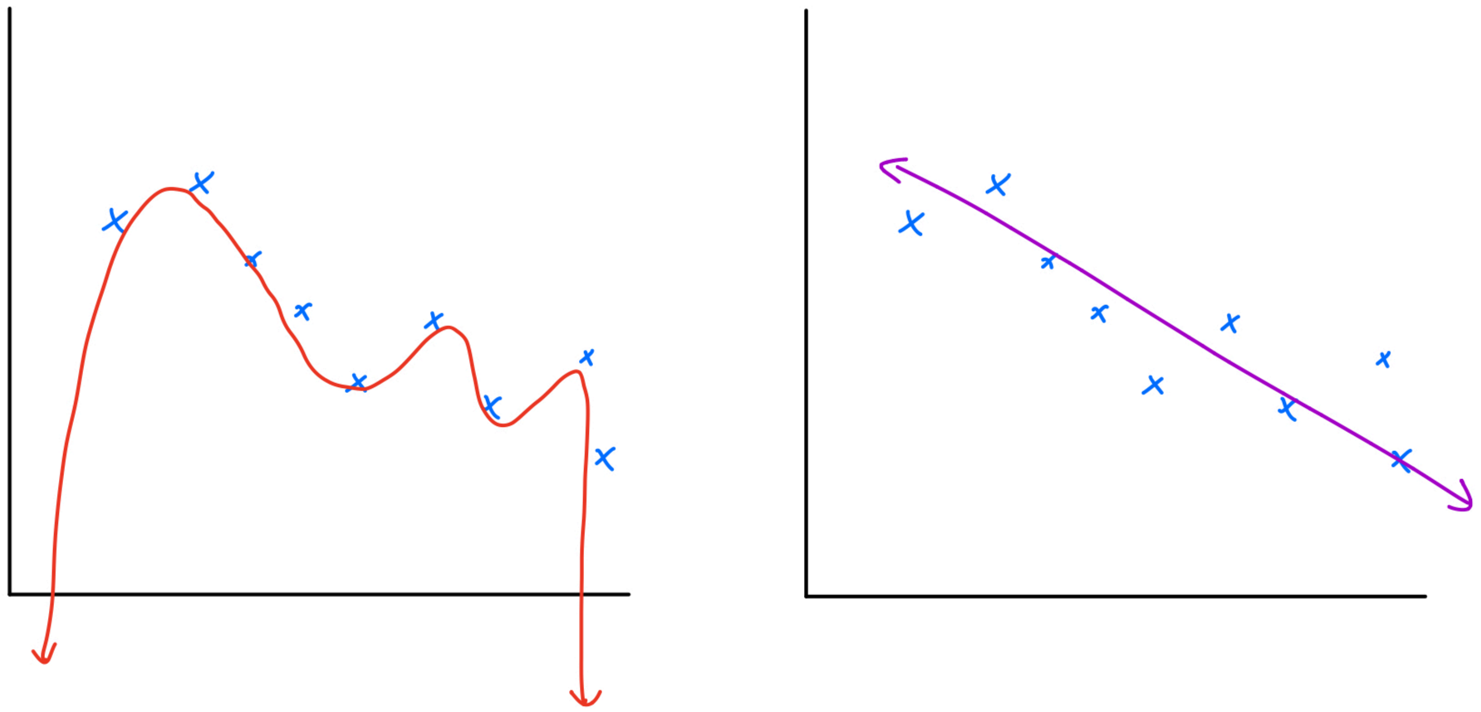
As λ increases, model complexity decreases. This is because increasing λ increases the penalty on the magnitude of β. Since we are trying to minimize the objective, if λ increases, $\left|\left|\beta\right|\right|_2^2$ must decrease.

As a result, as λ increases, model bias increases, and variance decreases.

- Bias increases because our model becomes less complex, and thus more general.
- Variance decreases because, again, our model becomes more general.


Optimal Value
Test Error
Variance
$\mathbf{E}\left[\left(\mathbf{E}\left[f_{\hat{\theta}}(x)\right] - f_{\hat{\theta}}(x)\right)^2\right]$
$\left(h(x) - \mathbf{E}\left[f_{\hat{\theta}}(x)\right]\right)^2 +$
(Bias)^2
Increasing Model Complexity ➡

Polynomial regression with large d, small d:

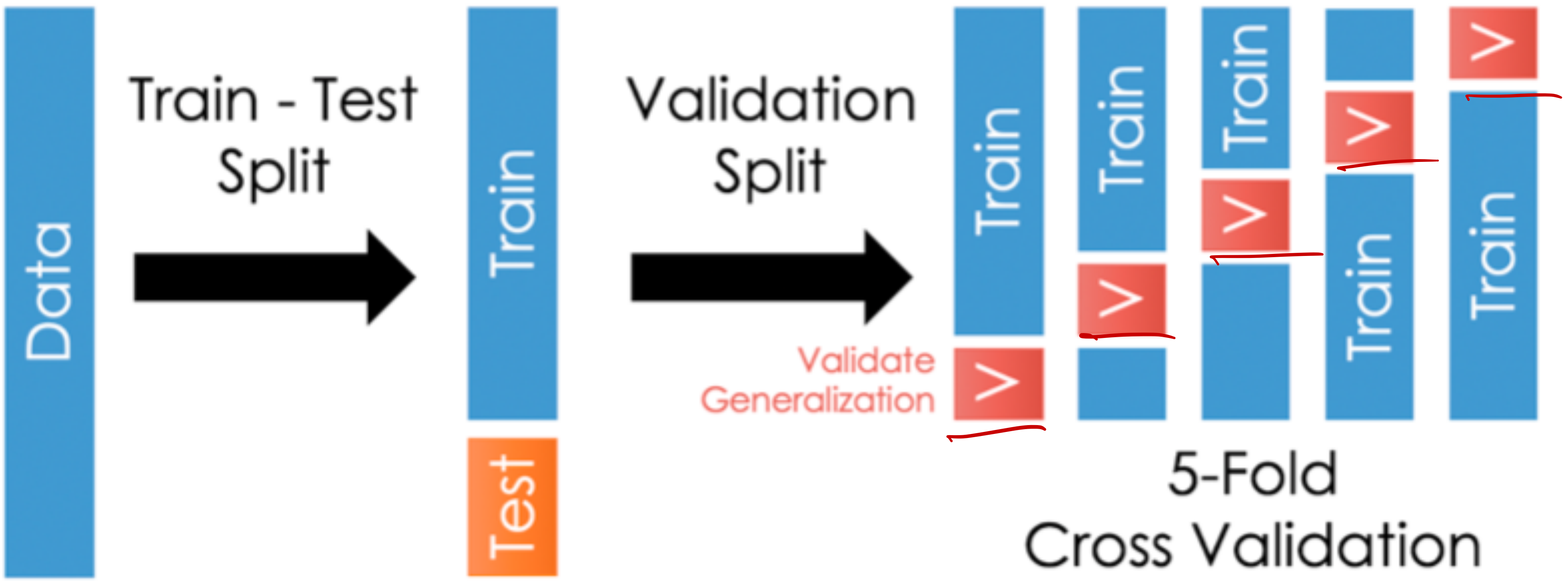


The high degree polynomial model has lower bias, but higher variance, than the model on the right.

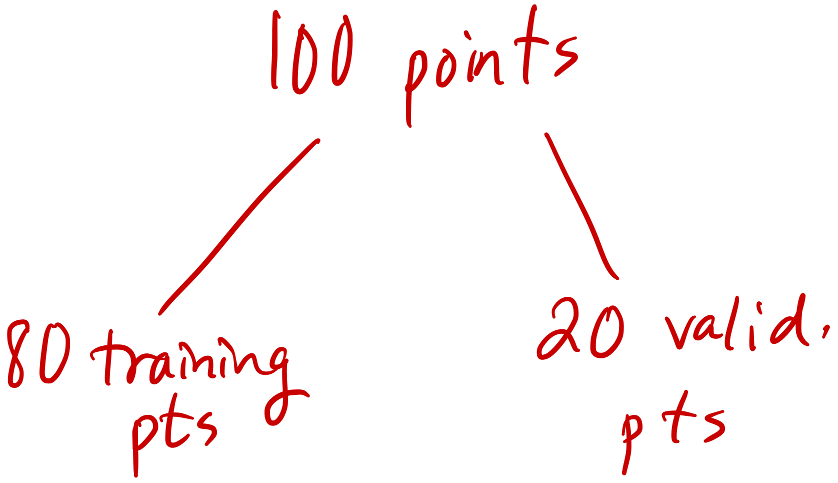
One way to interpret variance: In the model on the left, if we were to introduce a new point, our polynomial model would change significantly. However, on the right, introducing a new point is unlikely to change our model by much.

λ

Cross Validation



Cross validation simulates multiple train-test splits on the training data. This helps us find the best hyperparameters for our model.



Procedure:

- Split data in to training/test data
- For each training fold, split into training/validation data
- Train on training data, get score/error on validation data
- Repeat for different values of hyperparameters

Pros

- Helps us choose hyperparameters for model
- Makes the most of a limited amount of data (ie helpful when it is hard to collect more data)

Cons

- Takes more time to run, especially as number of hyperparameters grow

take λ
with lowest
column average

Fold #	$\lambda=0.1$	$\lambda=0.2$	...
0	—	—	
1	—	—	
2	—	—	
3	—	—	
⋮	⋮	⋮	

$\boxed{\lambda=0.1}$

$\lambda=0.2$

Suppose you are working with a partner to train a model with one hyperparameter λ. Together, you and your partner run 5-fold cross validation and compute mean squared errors for each fold and value of λ from a set of 4 candidate values for λ. However, your partner forgets to send you the results for the last two folds! The table below contains the mean squared errors for the first three of five total folds.

Fold Num	$\lambda = 0.1$	$\lambda = 0.2$	$\lambda = 0.3$	$\lambda = 0.4$	Row Avg
1	64.2	60.1	77.7	79.2	70.3
2	76.8	66.8	88.8	98.8	82.8
3	81.5	71.5	86.5	88.5	82.0

(a) [3 Pts] Your partner uses the full table containing data for all five folds to create a final model to use on test data. Given the information above, what can you conclude about the final model? Select all that apply.

- □ A. Our final model should use $\lambda = 0.4$.
- □ B. Our final model should be trained on fold 1, since it achieves the lowest row average.
- □ C. Our final model should be trained on fold 2, since it achieves the highest row average.
- ☑ D. None of the above.

we were never given column averages!

(b) [2 Pts] Let's say we know the row averages for all 5 folds. Which of the following are valid conclusions we can draw? Select all that apply.

- □ A. We can determine which fold number to use for our model.
- □ B. We can determine which λ value to use in our model.
- ☑ C. None of the above.

Practice: Regularization, B-V (T/F)

[✓ True]/False: L_1 reguarlization can help us select a subset of the features that are important. sparsity

True/[✓ False]: After regularization, we expect the training accuracy to increase and the test accuracy to decrease. no: regularization makes our model more general, having opp. effect

True/[✓ False]: In ridge regression, if we let $\lambda \to \infty$, our model will become more and more complex. model goes to 0

True/[✓ False]: As we improve our model to reduce bias, we often run the risk of under-fitting.

True/[✓ False]: Training error is typically larger than test error. typically, the opposite (not always)

fixed

Practice: Regularization

Consider the following general loss formulation.

$$\arg\min_{\theta}\left[\sum_{i=1}^{n}(y_i - x_i^T\beta)^2 + \lambda\sum_{p=1}^{d}\beta^2\right]$$

a) How many data points are there? n

b) What dimension is our data, i.e. how many features are we using? d

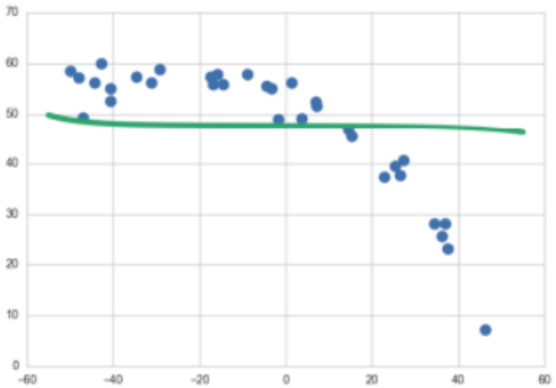
c) Is this a classification or regression problem? regression

d) What type of regularization is being used? L_2 (ridge)

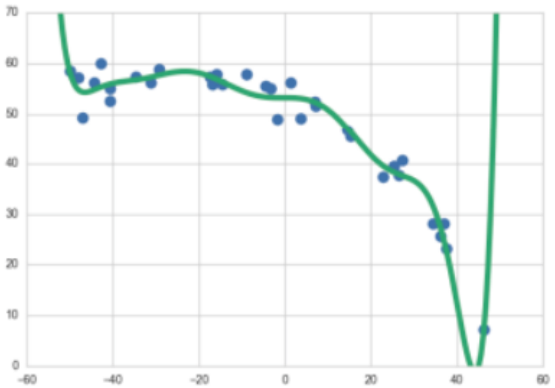
e) As λ increases, what will happen to bias? increase

f) As λ increases,what will happen to variance? decrease

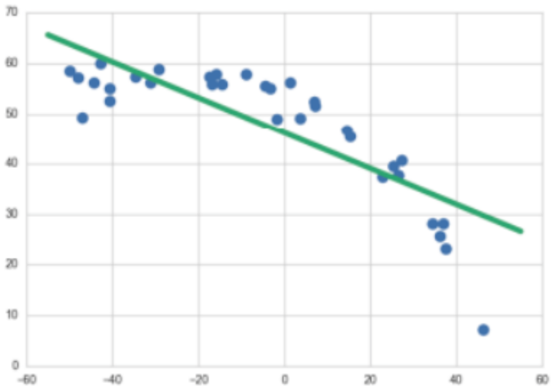
Practice: Identifying types of regression



(a)



(b)



(c)

Determine the plot above that best represents each of the following models.

i. Linear regression (c)

ii. Regularized linear regression, using polynomial features and a large λ (a)

iii. Linear regression, using degree 10 polynomial features (b)

Classification

Regression vs. Classification

Regression is the problem of creating a model that takes in a point and outputs a real number. We've seen regression in the form of Ordinary Least Squares, Ridge Regression, and LASSO Regression.

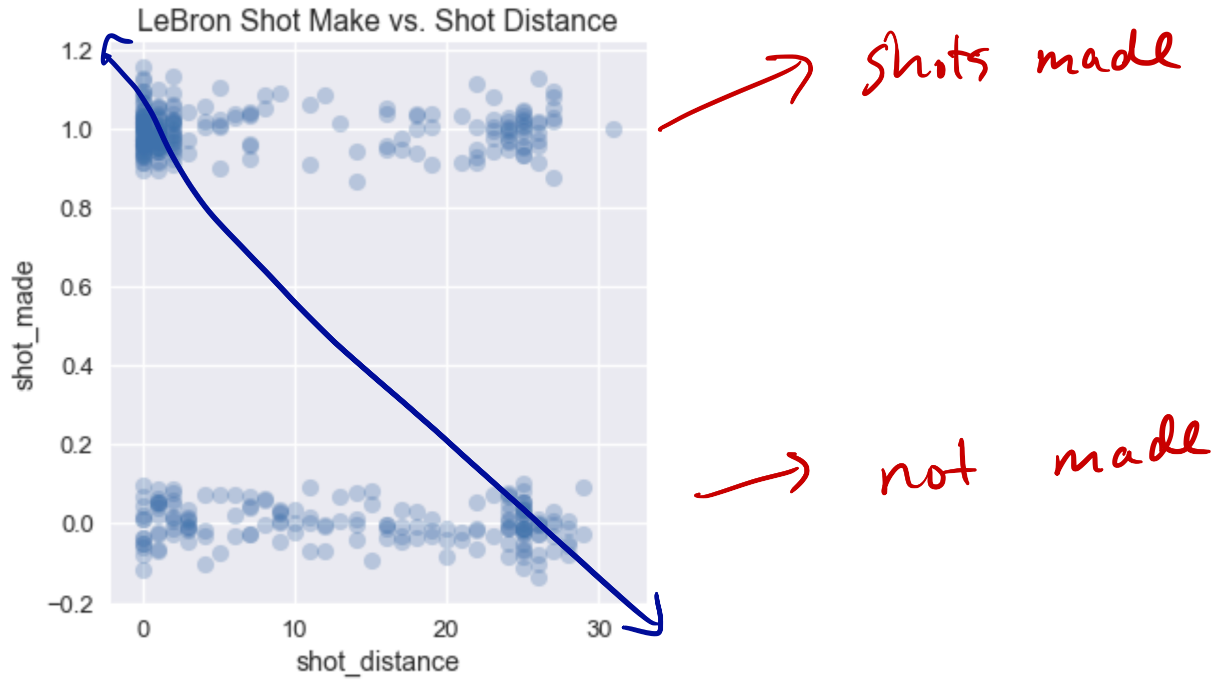
On the other hand, **classification** is the problem of creating a model that takes in a point and outputs a discrete **label**.

A very basic example:

- Regression would allow us to predict a student's final exam grade, given their grades on the midterm and homeworks.
- Classification would allow us to predict whether or not a patient has a disease.

Example (from textbook): Suppose we have LeBron's shot data for a particular season. Specifically, we have the distance from which the shot was taken, and whether or not it went in.

We want to build a model that will allow us to predict whether or not a new shot will go in.



Sure, we *can* fit a standard linear regression model to this, and interpret the output as the *probability that the shot will go in*. For example, we can say if the predicted value is over 0.5, he will make the shot. **However, values aren't restricted to $[0, 1]$!**

Probability Recap, Sigmoid Function

A probability density function $f(x)$ is valid iff it satisfies the following conditions:

- $0 \leq f(x) \leq 1$, for all $x \in \mathbb{X}$
- $\sum_{x \in \mathbb{X}} f(x) = 1$

We need some function that maps $\mathbb{R} \rightarrow [0, 1]$. Our choice is $\sigma(x)$:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

This is known as the **sigmoid** function, and it satisfies the following property:

$$\frac{d}{dx}\sigma(x) = \sigma(x)(1 - \sigma(x))$$

With standard linear regression, to fit our model $E[Y|x] = x^T\beta$, we found the β that minimizes

$$\min_{\beta} \frac{1}{n}||y - X\beta||_2^2 \Leftarrow$$

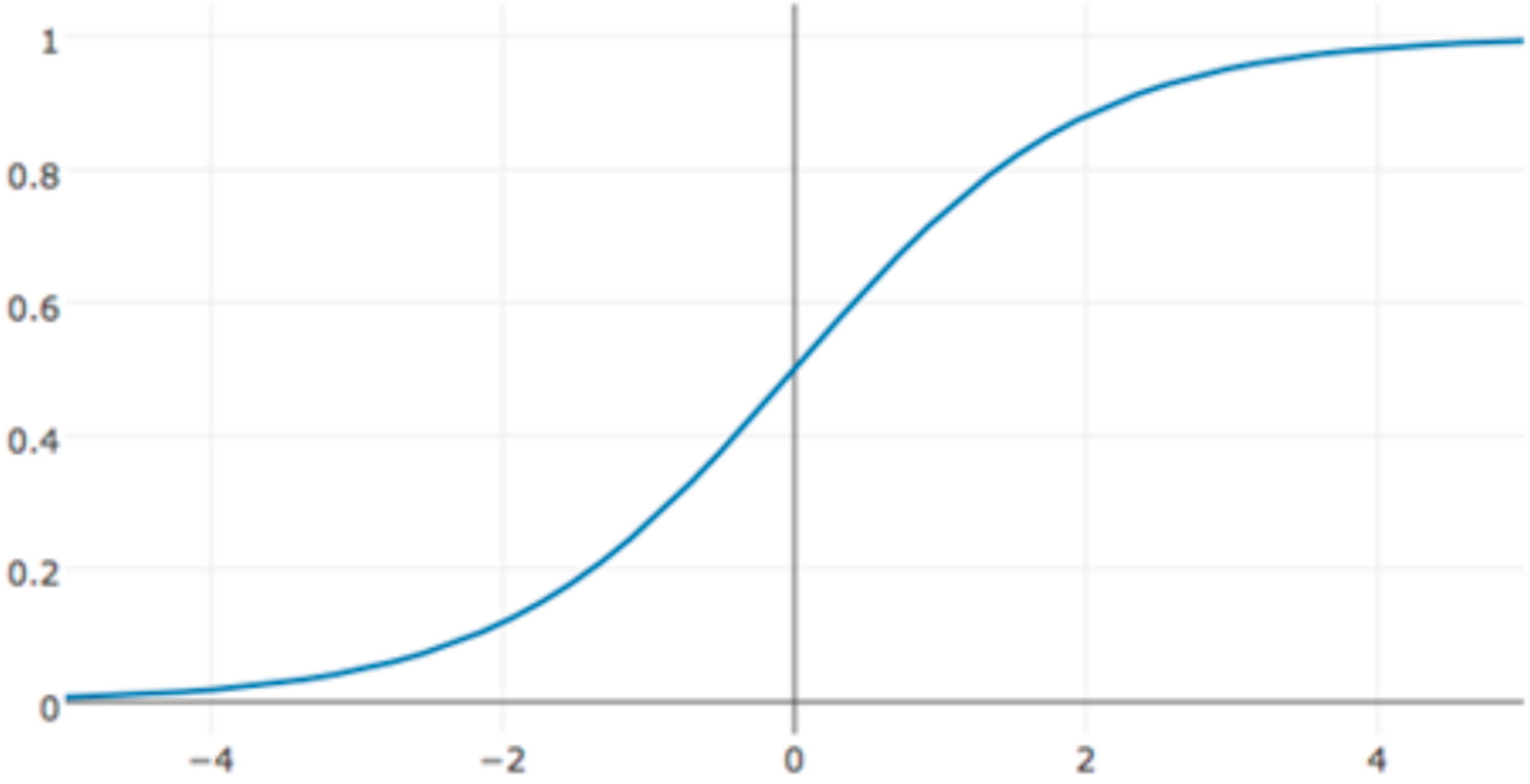
However, with **logistic regression**, we instead model using the following (assuming our classification is binary):

$$\mathbb{P}_\beta(Y = 1|\overset{\substack{\text{test point}\\ \Downarrow}}{x}) = \sigma(x^T\beta) = \frac{1}{1 + \exp(-x^T\beta)}$$

This is the probability that our test point x belongs to class 1 (as opposed to class 0).

Important: The output of logistic regression on its own is not a classification. We need to decide a *cutoff*, or decision boundary, in order to complete our classifier.

Output of $\sigma(x)$



Loss Function for Logistic Regression

"log-loss"

The loss function we use for logistic regression is what is known as **cross entropy loss**. It has information-theoretic foundations, and is preferred over L2 loss for a number of reasons.

Average cross entropy loss is of the form

$$L(\beta) = -\frac{1}{n}\sum_{i=1}^{n}(y_i x_i^T \beta \underset{\Uparrow}{+} \log(\sigma(-x_i^T \beta)))$$

This cannot be determined analytically, unlike the solution to OLS. We must use a numerical method, such as gradient descent, to determine β^*.

$$\log\left(\frac{1}{1+e^{-x}}\right)$$

y_i : scalar

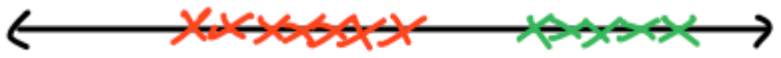
x_i : feature vector

β : weights vector

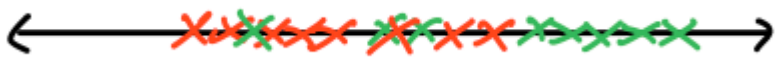
Linear Separability

The goal of logistic regression is to model the probability of a point belonging to a class. We only do this when there is some level of uncertainty, i.e. overlap, in our training set.

In some cases, we are able to draw a *linear decision boundary* to separate our data.

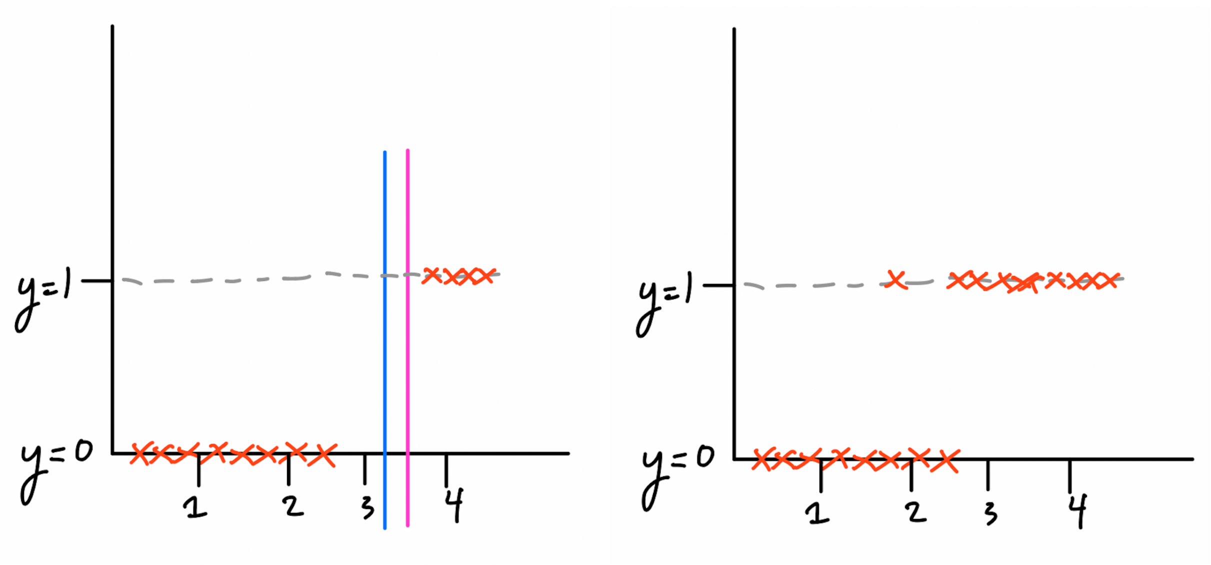


Above, we see that our data is **linearly separable**. We can draw a line (infinitely many lines, in fact) between the clusters of red dots and green dots.



This is not the case in the second example.

Again, let's look at data in 1D, but plotted in 2D (one dimension is the value of our variable, the second dimension is the label, 0 or 1 --- this is equivalent to the drawings on the previous slide).

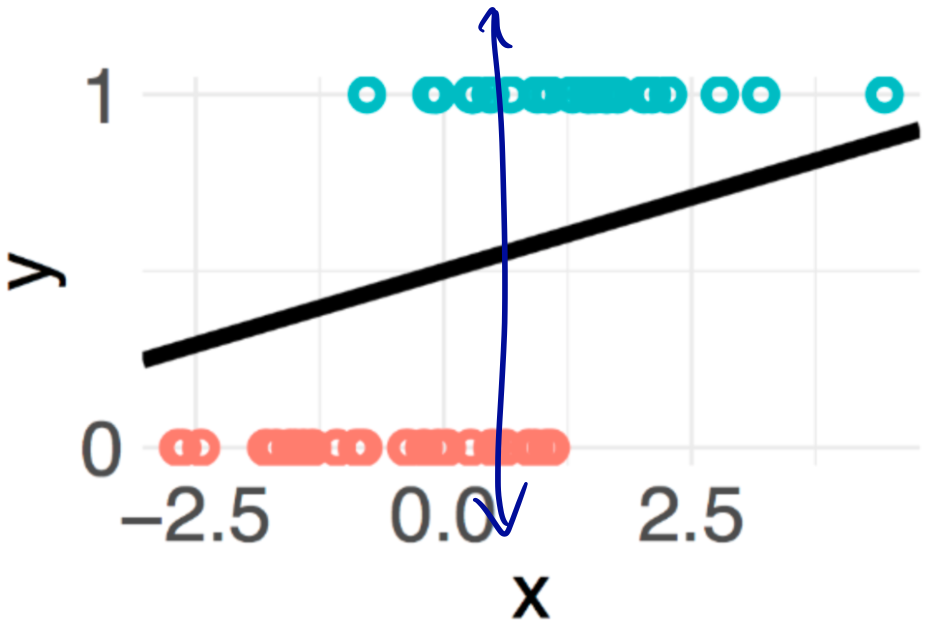


On the left, we have an example of linearly separable data. In the blue and purple are two possible hyperplanes that can separate our data. There would be no use in using logistic regression here.

However, on the right, we have non-linearly separable data. In this case we would use a tool like logistic regression to model probabilities.

IMPORTANT: Linear Separability Depends on the Dimension of the Data!

A set of d-dimensional points is linearly separable iff we can draw a degree $d-1$ hyperplane to separate the points.



$d = 1$

$d=0$ $x = a$

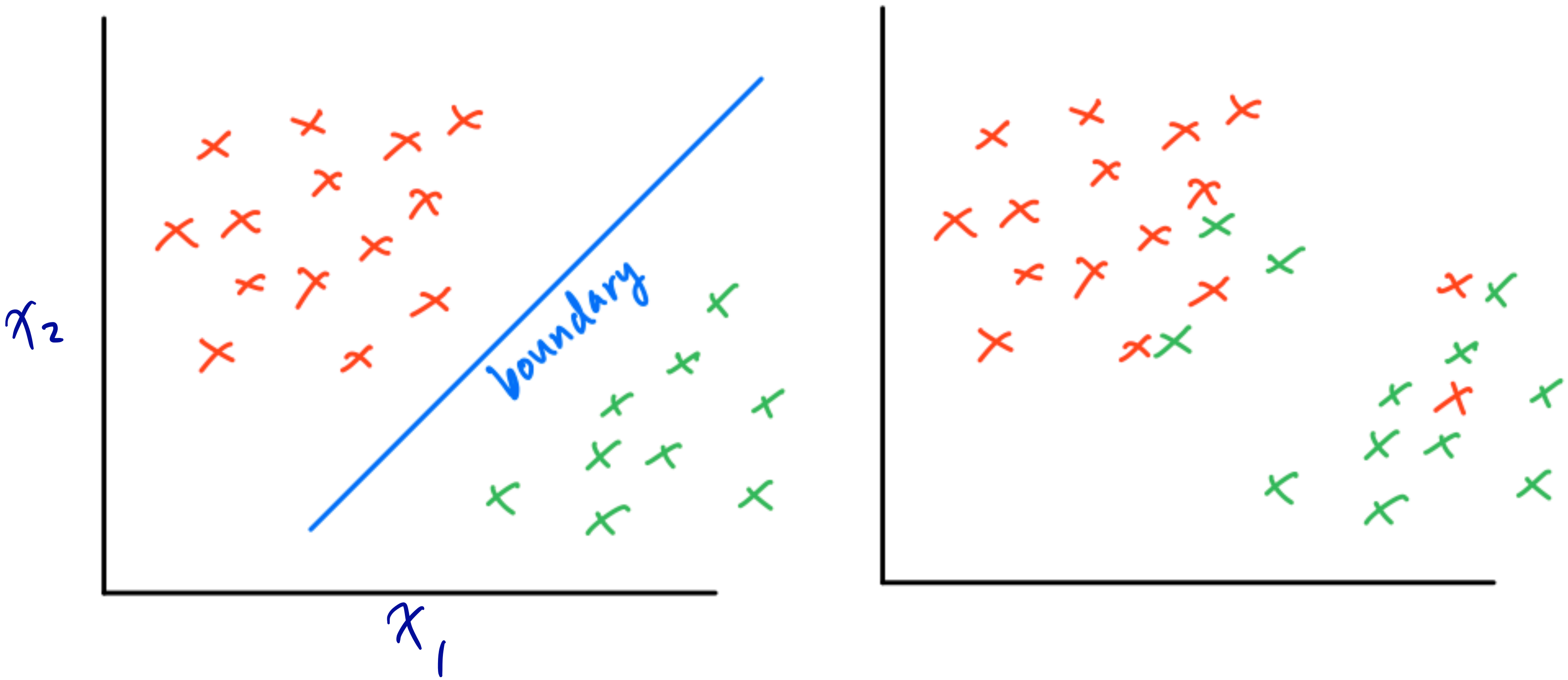
$d=1$ $y = ax+b$

This data **is not** linearly separable. This problem (from Disc 8) is intentionally misleading; the points are in 1D, however the class labels are represented in two ways (y axis and color). We cannot draw a degree 0 line (i.e. something of the form $x = a$) to separate this data.

$d=2$

$d=1 \quad y = ax+b$

Linear Separability in Two Dimensions



Here, we have examples of both linearly separable and non-linearly separable data in two dimensions. Here, our data is truly two dimensional, as our feature space has two components --- an x_1 and a x_2. The class is represented by the color (Y).

Evaluating the Effectiveness of a Classifier

Suppose we train a binary classifer, and suppose `y` represents actual values, and `y_pred` represents predicted values.

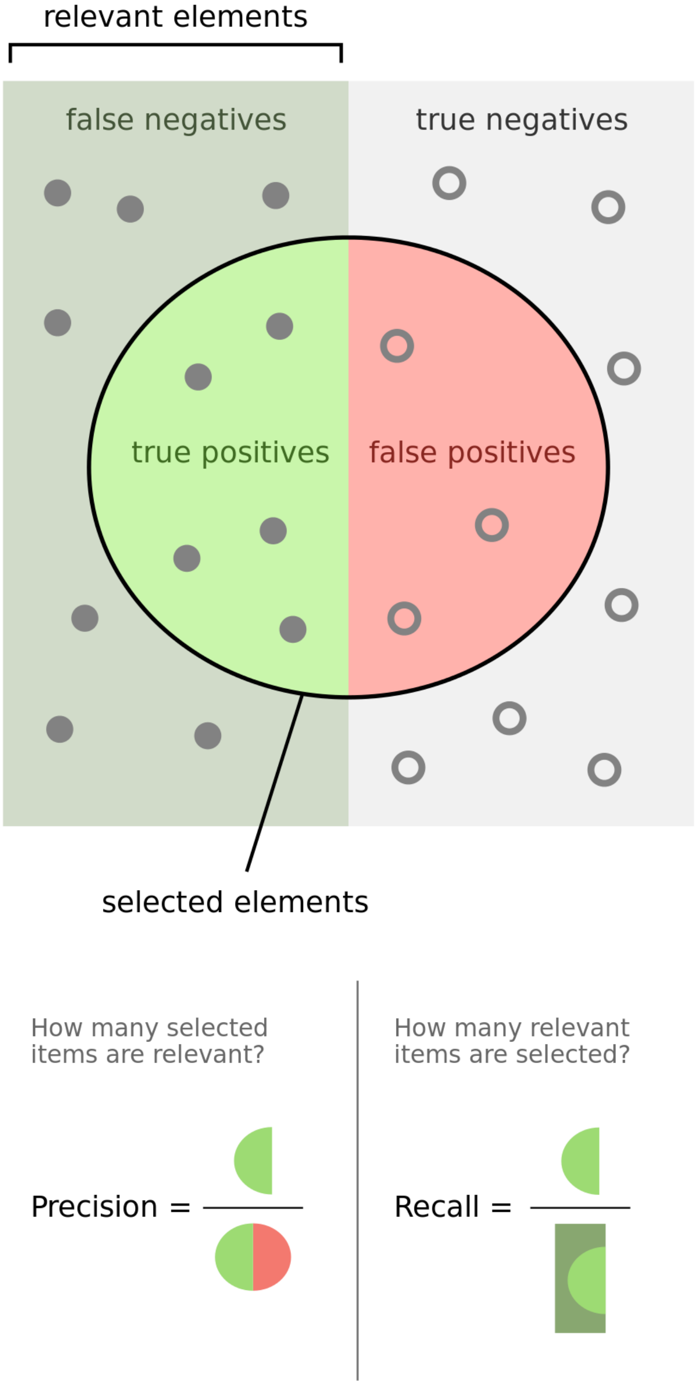
Recall (no pun intended) the following definitions:

- True Positives: `TP = np.count_nonzero((y == y_pred) & (y_pred == 1))`
- True Negatives: `TN = np.count_nonzero((y == y_pred) & (y_pred == 0))`
- False Positives: `FP = np.count_nonzero((y != y_pred) & (y_pred == 1))`
- False Negatives: `FN = np.count_nonzero((y != y_pred) & (y_pred == 0))`

all terms involve p

Then, we have the following definitions:

- Precision = $\frac{TP}{TP+FP}$, measures the number of predicted true values that are actually true
- Recall = $\frac{TP}{TP+FN}$, measures the number of actually true values that are marked true ("detection rate")



relevant elements

false negatives

true negatives

true positives

false positives

selected elements

How many selected items are relevant?

Precision =

How many relevant items are selected?

Recall =

Example

Suppose you create a classifier to predict whether or not an image contains a picture of a goat. You test it on 23 images.

- There were 12 true images of goats. Your classifier predicted 9 of them to be goats, and 3 to not be a goat.
- There were 11 images that did not contain goats. Your classifier predicted 3 of them to be goats, and the remaining 8 to not be goats.

Determine the precision and recall of your goat classifier.

TP : 9
FN : 3
FP : 3
TN : 8

$$\text{Precision} = \frac{TP}{TP+FP} = \frac{9}{9+3} = \frac{3}{4}$$

$$\text{Recall} = \frac{TP}{TP+FN} = \frac{9}{9+3} = \frac{3}{4}$$

(coincidence in this case)

Practice: Classification (T/F)

True/[✓ False]: In logistic regression, predictor variables (x) are continuous, with values in the range $[0, 1]$.

No restriction on inputs x

True/[✓ False]: In two-class binary classification, the output is continuous, with values in the range $[0, 1]$.

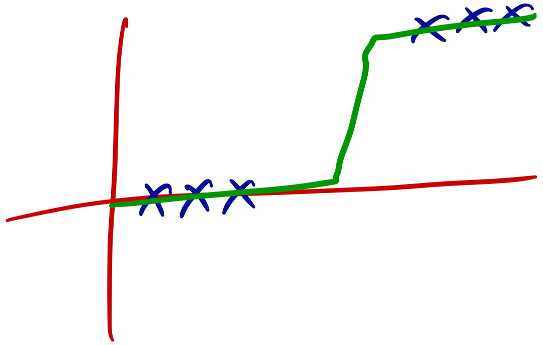
output is 0 or 1

→ this is tricky. if this said "logistic regression", then true

True/[✓ False]: In logistic regression, we calculate the weights β^* as $\beta^* = (X^T X)^{-1} X^T y$ and then fit responses as $y = \sigma(x^T \beta)$.

Need to fit cross entropy loss.

Practice: Logistic Regression (T/F)



[✓ True]/False: If no regularization is used and the training data is linearly separable, the parameters will tend towards positive or negative infinity.

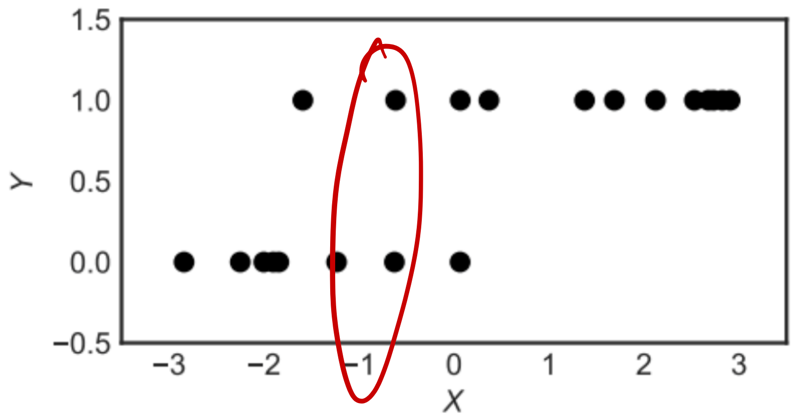
[✓ True]/False: L_1 reguarlization can help us select a subset of the features that are important.

True/[✓ False]: After regularization, we expect the training accuracy to increase and the test accuracy to decrease.

last two questions were repeated from earlier, sorry

Practice: Interpreting Logistic Regression

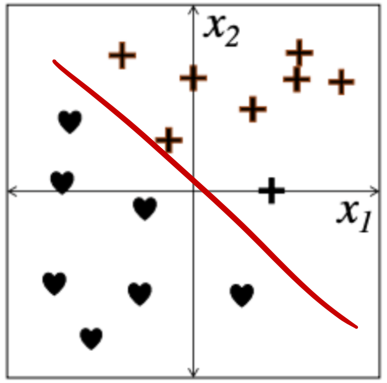
[2 Pts] Suppose you are given the following dataset $\{(x_i, y_i)\}_{i=1}^{n}$ consisting of x and y pairs where the covariate $x_i \in \mathbb{R}$ and the response $y_i \in \{0, 1\}$.



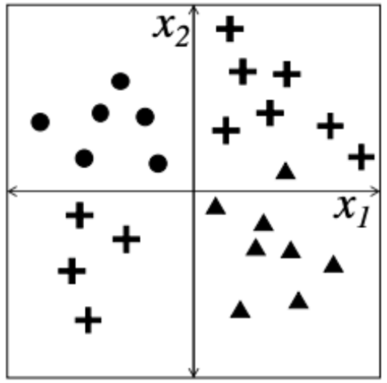
Given this data, the value $\mathbf{P}\,(Y = 1 \,|\, x = -1)$ is likely closest to:

○ 0.95 [✓ ○ 0.50] ○ 0.05 ○ -0.95

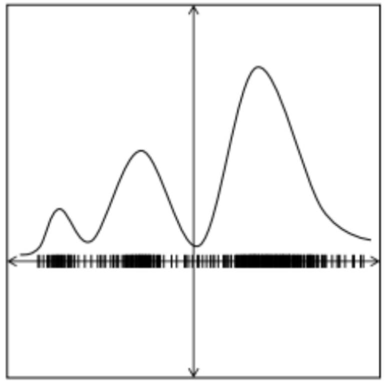
Practice: Separability



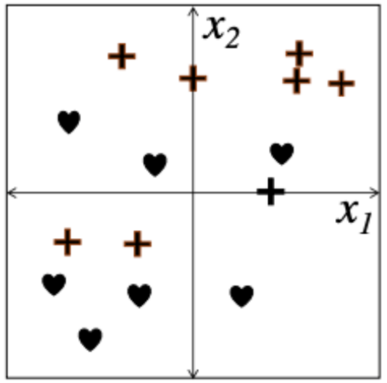
(A)



(B)



(C)



(D)

(1) Which of the above plots represents a **linearly separable binary classification** task?

[✓ ○ (A)] ○ (B) ○ (C) ○ (D)

(2) Which of the above plots represents a **binary classification** task that is **not linearly separable**?

○ (A) ○ (B) ○ (C) [✓ ○ (D)]

(3) Which of the above plots represents a **multi-class classification task**?

○ (A) [✓ ○ (B)] ○ (C) ○ (D)

good luck!