CS 540 Introduction to Artificial Intelligence
Linear Models & Linear Regression

University of Wisconsin-Madison
Spring 2026 Sections 1 & 2

Announcements

- HW4 due on **Wednesday Feb. 25th at 11:59 PM**
- Class roadmap:

ML Linear Regression	Supervised Learning
Machine Learning: K - Nearest Neighbors & Naive Bayes	
Machine Learning: Neural Networks I (Perceptron)	

Outline

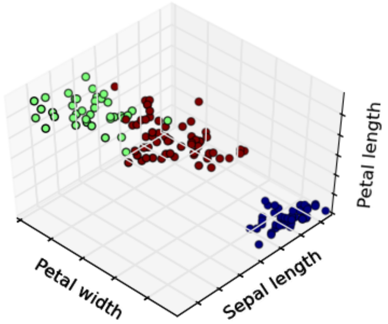
- Unsupervised Learning Recap
- Supervised Learning with Linear Models
  - Parameterized model, model classes, linear models, train vs. test
- Linear Regression
  - Least squares, normal equations, residuals, logistic regression

Unsupervised Learning Recap

Dataset: $x_1, x_2, \ldots, x_n$ **No labels**;

Goal: find patterns & structures that help better understand data.

- Clustering:
  - Hierarchical Clustering
  - Center-based Clustering (*k-means*)
  - Spectral clustering
- Visualization - T-SNE
- Density Estimation
- PCA



Supervised Learning

Supervised learning:

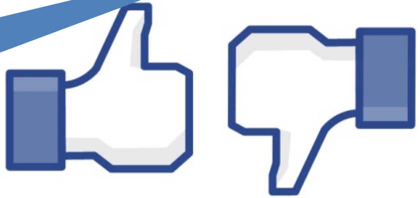
- Make predictions, classify data, perform regression
- Dataset: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

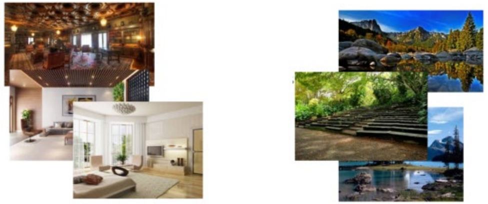
Feature vector / Covariates / Input

Label

- Goal: find function $f : X \rightarrow Y$ to predict label on **new** data

Select a **model** f from a class of possible models





indoor

outdoor

Regression

- Continuous label $y \in \mathbb{R}$
- Squared loss function $\ell(f(x), y) = (f(x) - y)^2$
- Finding f that minimizes the empirical risk

$$\frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Basic Recipe for Supervised Learning

1. Select model class

2. Select loss

3. Optimize parameters

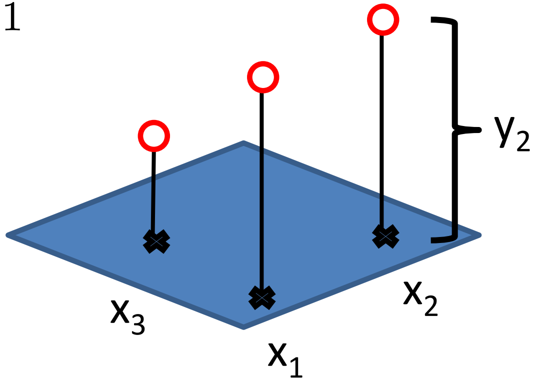
4. Evaluate output

Functions/Models

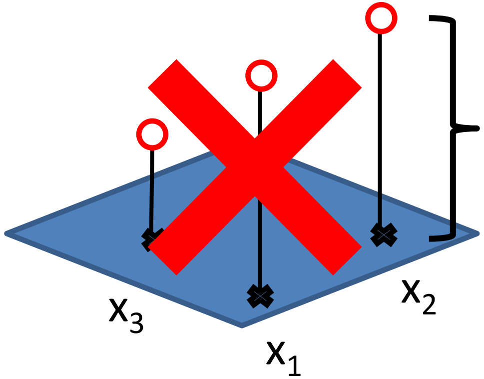
The function *f* is usually called a model

- Which possible functions should we consider?
- One option: **all functions**
  - Not a good choice. Consider $$f(x) = \sum_{i=1}^{n} 1\{x = x_i\} y_i$$
  - Training loss: **zero.** Can't do better!
  - How will it do on *x* not in the training set?

(cannot generalize)



Functions/Models



Don't want all functions

- Instead, pick a specific class
- Parametrize it by weights/parameters
- **Example:** linear models

$$f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \theta_0 + x^T \theta$$

Weights/ Parameters

Training The Model

- Parametrize it by weights/parameters
- Minimize the loss

Best parameters = best function *f* →

$$\min_\theta \frac{1}{n}\sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Linear model class *f*

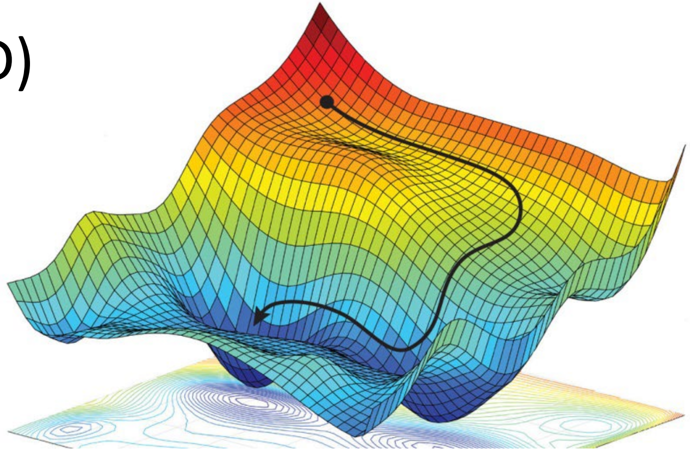
$$= \frac{1}{n}\sum_{i=1}^{n} \ell(\theta_0 + x_i^T\theta, y_i)$$

Square loss

$$= \frac{1}{n}\sum_{i=1}^{n} (\theta_0 + x_i^T\theta - y_i)^2$$

How Do We Optimize Parameters?

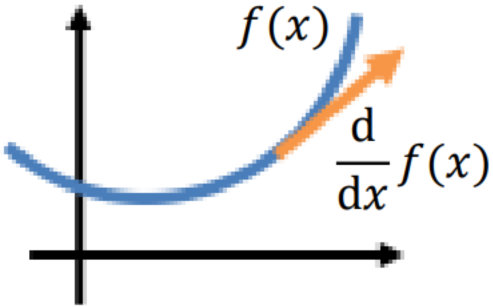
- Need to solve something that looks like $\min_{\theta} g(\theta)$
- Generic optimization problem; many algorithms
- Most popular: variants of **gradient descent**
  - Stochastic Gradient Descent (SGD)
  - Momentum
  - Adam



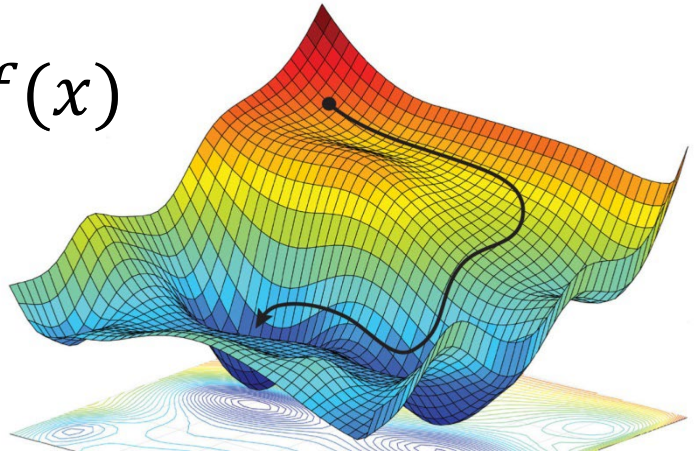
M. Hutson

Gradients & Gradient Descent

- One dimension: derivative $\frac{\mathrm{d}}{\mathrm{d}x} f(x)$
  - How to shift x to make $f(x)$ larger



- Higher dimensions: gradient $\nabla f(x)$
  - Direction where f grows fastest

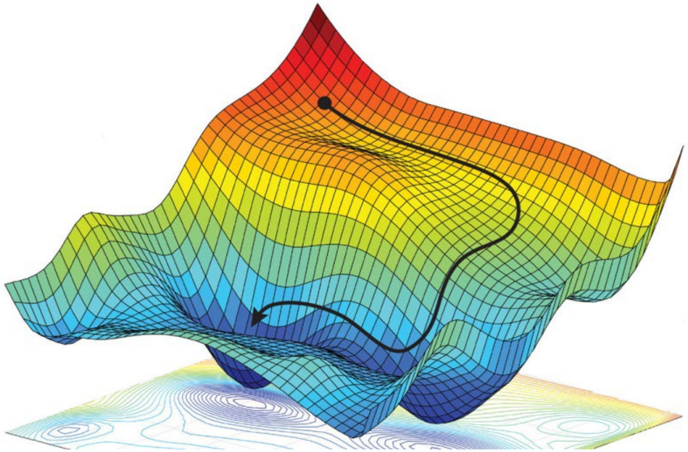


M. Hutson

Gradients & Gradient Descent

- Gradient descent takes iterative steps to make loss function smaller

Gradient Descent
<u>Input:</u> dataset (X, y), loss function L, number of steps T, step size η
1. Initialize θ_0 2. For $t = 1,2, \dots, T$ 3. Calculate $g_t = \nabla L(\theta_{t-1}; X, y)$ 4. Update $\theta_t \leftarrow \theta_{t-1} - \eta g_t$ 5. Return θ_T



M. Hutson

Train vs Test

Now we've trained, have some *f* parametrized by θ

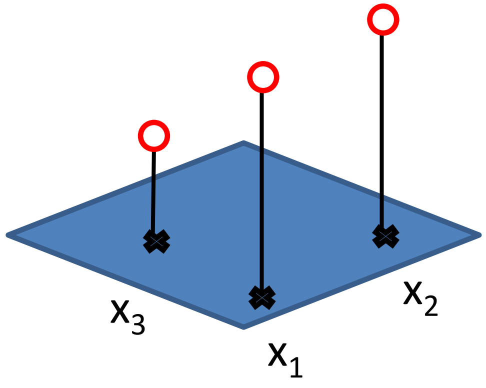
- Train loss is small → *f* predicts most x_i correctly
- How does *f* do on points not in training set? **"Generalizes!"**
- To evaluate this, reserve a **test** set. Do **not** train on it!

$$\underbrace{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)}_{\text{Training Data}} \quad \underbrace{(\mathbf{x}_{n+1}, y_{n+1}), \ldots, (\mathbf{x}_{n+p}, y_{n+p})}_{\text{Test Data}}$$

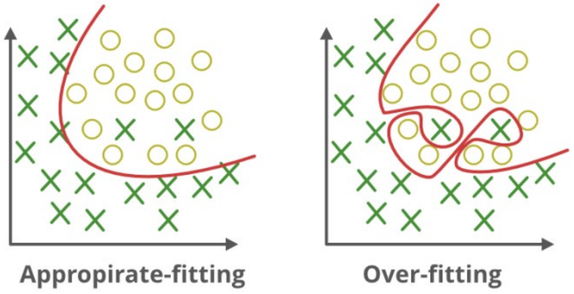
Train vs Test

Use the test set to evaluate *f*

- Why? Back to our "perfect" train function
- Training loss: 0. Every point matched perfectly
- How does it do on test set? **Fails completely**!



- Test set helps detect **overfitting**
  - Overfitting: too focused on train points
  - "Bigger" class: more prone to overfit
    - Need to consider **model capacity**



GFG

Break & Quiz

Q 1.1: When we train a model, we are

- A. Optimizing the parameters and keeping the features fixed.
- B. Optimizing the features and keeping the parameters fixed.
- C. Optimizing the parameters and the features.
- D. Keeping parameters and features fixed and changing the predictions.

Break & Quiz

Q 1.1: When we train a model, we are

- **A. Optimizing the parameters and keeping the features fixed.**
- B. Optimizing the features and keeping the parameters fixed.
- C. Optimizing the parameters and the features.
- D. Keeping parameters and features fixed and changing the predictions.

Break & Quiz

Q 1.1: When we train a model, we are

- **A. Optimizing the parameters and keeping the features fixed.**
- B. Optimizing the features and keeping the parameters fixed) (Feature vectors *x*i don't change during training).
- C. Optimizing the parameters and the features. (Same as B)
- D. Keeping parameters and features fixed and changing the predictions. (We can't train if we don't change the parameters)

Break & Quiz

- **Q 1.2**: You have trained a classifier, and you find there is significantly **higher** loss on the test set than the training set. What is likely the case?

- A. You have accidentally trained your classifier on the test set.
- B. Your classifier is generalizing well.
- C. Your classifier is generalizing poorly.
- D. Your classifier is ready for use.

Break & Quiz

- **Q 1.2**: You have trained a classifier, and you find there is significantly **higher** loss on the test set than the training set. What is likely the case?

- A. You have accidentally trained your classifier on the test set.
- B. Your classifier is generalizing well.
- **C. Your classifier is generalizing poorly.**
- D. Your classifier is ready for use.

Break & Quiz

- **Q 1.2**: You have trained a classifier, and you find there is significantly **higher** loss on the test set than the training set. What is likely the case?

- A. You have accidentally trained your classifier on the test set. **(No, this would make test loss lower)**
- B. Your classifier is generalizing well. **(No, test loss is high means poor generalization)**
- **C. Your classifier is generalizing poorly.**
- D. Your classifier is ready for use. **(No, will perform poorly on new data)**

Break & Quiz

- **Q 1.3**: You have trained a classifier, and you find there is significantly **lower** loss on the test set than the training set. What is likely the case?

- A. You have accidentally trained your classifier on the test set.
- B. Your classifier is generalizing well.
- C. Your classifier is generalizing poorly.
- D. Your classifier needs further training.

Break & Quiz

- **Q 1.3**: You have trained a classifier, and you find there is significantly **lower** loss on the test set than the training set. What is likely the case?

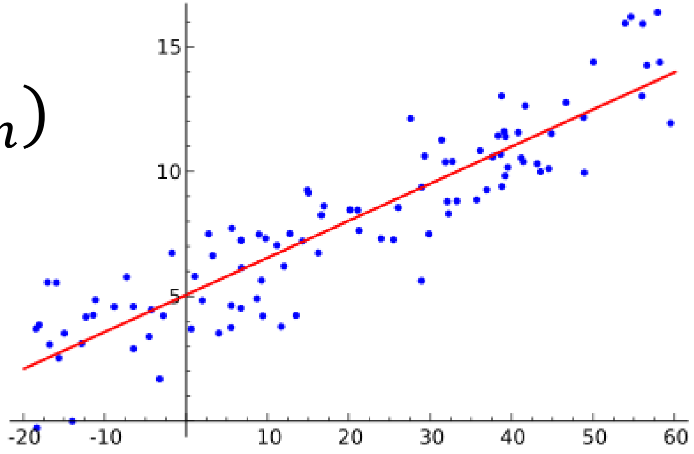
- **A. You have accidentally trained your classifier on the test set.** (This is very likely, loss will usually be the lowest on the data set on which a model has been trained)
- B. Your classifier is generalizing well.
- C. Your classifier is generalizing poorly.
- D. Your classifier needs further training.

Linear Regression

- Simplest form of regression
- Find a line that fits the data

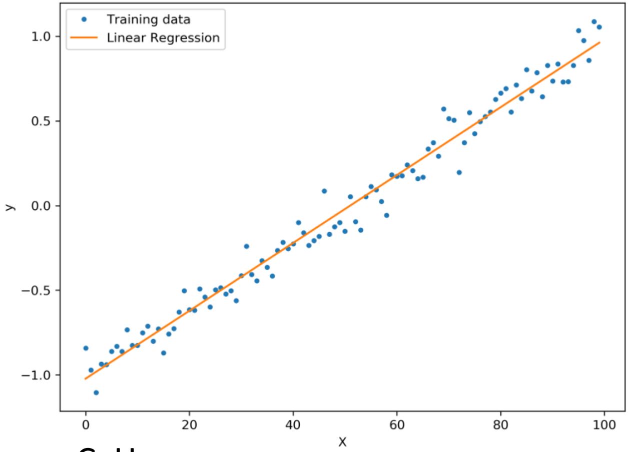
- Example:
  - Training data $(x_1, y_1), \ldots, (x_n, y_n)$
  - Find $a, b \in \mathbb{R}$ such that

$$\forall i,\ y_i \approx a x_i + b$$



Linear Regression

- **Inputs**: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$
  - *x*'s are vectors, *y*'s are scalars.
  - "**Linear**": predict a linear combination of x components + intercept



C. Hansen

$$f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \theta_0 + x^T \theta$$

- **Want**: parameters θ

Linear Regression Setup

Problem Setup

- Goal: figure out how to minimize square loss
- Let's organize it. Train set

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$$

Notational Trick

- When x is a scalar: $f_{(a,b)}(x) = ax + b$
- When x is a vector:

  $f(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_d x_d = \theta_0 + x^T\theta$
- Give x a "dummy dimension" to simplify notation

Old

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

New

$$x = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \qquad f(x) = \begin{bmatrix} 1 & x_1 & x_2 & \cdots & x_d \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ x_d \end{bmatrix} = \langle x, \theta \rangle = x^T\theta$$

Linear Regression Setup

Problem Setup

- Train set $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$
- Take train features and make it a n*(d+1) matrix, and y a vector:

$$X = \begin{bmatrix} x_1^T \\ \ldots \\ x_n^T \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ \ldots \\ y_n \end{bmatrix}$$

- Then, the empirical risk is $\frac{1}{n}\|X\theta - y\|^2$

Finding The Estimated Parameters

Have our loss: $\frac{1}{n}\|X\theta - y\|^2$

- Could optimize it with SGD, etc…
- But the minimum also has a closed-form solution

  (vector calculus, proof in the Suggested Readings-optional):

$$\hat{\theta} = (X^T X)^{-1} X^T y$$

Hat: indicates an estimate

Not always invertible…

“Normal Equations”

How Good are the Estimated Parameters?

Now we have parameters $\hat{\theta} = (X^T X)^{-1} X^T y$

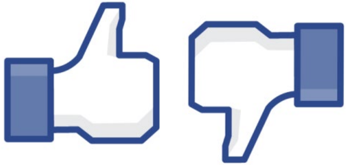
- How good are they?
- Predictions are $f(x_i) = \hat{\theta}^T x_i = ((X^T X)^{-1} X^T y)^T x_i$
- Errors ("residuals")

$$|y_i - f(x_i)| = |y_i - \hat{\theta}^T x_i| = |y_i - ((X^T X)^{-1} X^T y)^T x_i|$$

- If data is linear, residuals are 0. Almost never the case!
- Mean squared error on a test set $\frac{1}{m} \sum_{i=n+1}^{n+m} (\hat{\theta}^T x_i - y_i)^2$

Linear Regression → Classification?

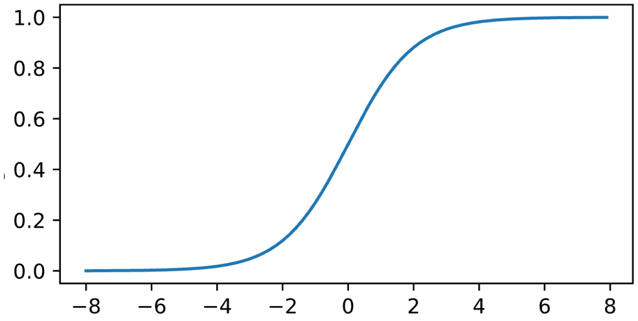
What if we want the same idea, but *y* is 0 or 1?



- Need to convert the $\theta^T x$ to a probability in [0,1]

Logistic function

$$p(y=1|x) = \frac{1}{1+\exp(-\theta^T x)}$$



Why does this work?

- If $\theta^T x$ is really big, $\exp(-\theta^T x)$ is really small → *p* close to 1
- If really negative exp is huge → *p* close to 0

"Logistic Regression"

Break & Quiz

Q 2.1: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

What are the labels, number of points (n), and dimension of the features (d)?

- A. labels are 2 and 4; n=3, and d=2.
- B. labels are 2 and 4; n=2, and d=3.
- C. labels are [-1,0,1] and [2,3,1]; n=2, and d=4.
- D. labels are 2 and 3; n=4, and d=2.

Break & Quiz

Q 2.1: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

What are the labels, number of points (n), and dimension of the features (d)?

- A. labels are 2 and 4; n=3, and d=2.
- **B. labels are 2 and 4; n=2, and d=3.**
- C. labels are [-1,0,1] and [2,3,1]; n=2, and d=4.
- D. labels are 2 and 3; n=4, and d=2.

Break & Quiz

Q 2.1: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

What are the labels, number of points (n), and dimension of the features (d)?

- A. labels are 2 and 4; n=3, and d=2.
- **B. labels are 2 and 4; n=2, and d=3.**
- C. labels are [-1,0,1] and [2,3,1]; n=2, and d=4.
- D. labels are 2 and 3; n=4, and d=2.

There are two data points, each x has 3 features, and the labels are the y-values.

Break & Quiz

Q 2.2: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. Predict $\hat{\boldsymbol{y}}$ for $x = [1, 10, 1]$

- A. 15
- B. 9
- C. 13
- D. 21

Break & Quiz

Q 2.2: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. Predict $\hat{\boldsymbol{y}}$ for $x = [1, 10, 1]$

- A. 15
- B. 9
- **C. 13**
- D. 21

Break & Quiz

Q 2.2: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. Predict $\hat{\boldsymbol{y}}$ for $x = [1, 10, 1]$

- A. 15
- B. 9
- **C. 13**
- D. 21

$\hat{y} = 1 * \beta_0 + 1 * \beta_1 + 10 * \beta_2 + 1 * \beta_3 = 13$

Break & Quiz

Q 2.3: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and$(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. What is the mean squared error (MSE) on the training set?

- A. 9
- B. 13/2
- C. 25/2
- D. 25

Break & Quiz

Q 2.3: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. What is the mean squared error (MSE) on the training set?

- A. 9
- B. 13/2
- C. **25/2**
- D. 25

Break & Quiz

Q 2.3: You have a dataset for regression given by $(x_1, y_1) = ([-1,0,1], 2)$ and $(x_2, y_2) = ([2,3,1], 4)$.

We have the weights $\beta_0 = 0, \beta_1 = 2, \beta_2 = 1, \beta_3 = 1$. What is the mean squared error (MSE) on the training set?

- A. 9
- B. 13/2
- C. **25/2**
- D. 25

Compute the predicted label for each data point, then compute the squared error for each data point, then take the mean error of the two points:

$$f(x_1) = \beta_0 \; - 1 * \beta_1 + 0 * \beta_2 + 1 * \beta_3 = -1$$
$$\ell(f(x_1), y_1) = (-1 - 2)^2 = 9$$

$$f(x_2) = \beta_0 + 2 * \beta_1 + 3 * \beta_2 + 1 * \beta_3 = 8$$
$$\ell(f(x_2), y_2) = (8 - 4)^2 = 16$$
$$MSE = (16 + 9)/2 = 2$$

Suggested Reading

- Linear regression, logistic regression, stochastic gradient descent by Prof. Zhu https://pages.cs.wisc.edu/~jerryzhu/cs540/handouts/regression.pdf