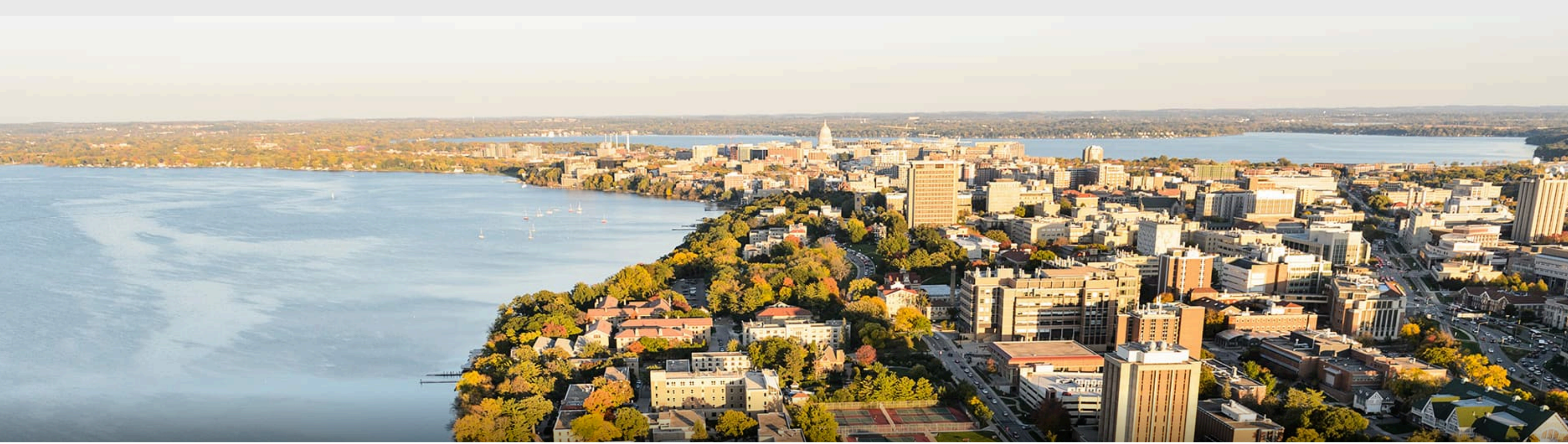


CS 540 Introduction to Artificial Intelligence
Linear Models & Linear Regression

Josiah Hanna
University of Wisconsin-Madison

October 12, 2021

Slides created by Fred Sala [modified by Josiah Hanna]

Announcements

- **Homeworks**:
  - HW 5 due next Tuesday
- **Midterm** coming up on October 28
  - Via Canvas, 24 hours to start.

Tuesday, Oct 12	Machine Learning: Linear Regression	Slides	HW 4 Due, HW 5 Released
Thursday, Oct 14	Machine Learning: K-Nearest Neighbors & Naive Bayes		
Tuesday, Oct 19	Machine Learning: Neural Network I (Perceptron)		HW 5 Due, HW 6 Released
Thursday, Oct 21	Machine Learning: Neural Network II		
Tuesday, Oct 26	Machine Learning: Neural Network III		
MIDTERM EXAM October 28			
Everything below here is tentative and subject to change.			
Tuesday, Nov 2	Machine Learning: Deep Learning I		

Outline

- Supervised Learning & Linear Models
  - Parameterized model, model classes, linear models, train vs. test
- Linear Regression
  - Least squares, normal equations, residuals, logistic regression
  - Gradient descent

Back to Supervised Learning

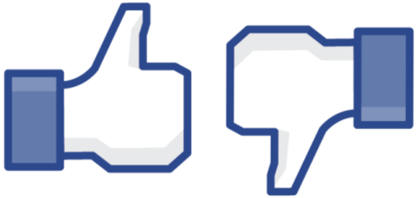
Supervised learning:

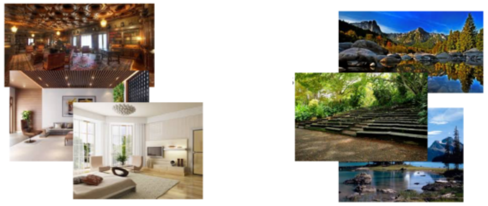
- Make predictions, classify data, perform regression
- Dataset: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$

Features / Covariates / Input

Labels / Outputs

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





indoor outdoor

Back to Supervised Learning

How do we know a function *f* is good?

- Intuitively: "matches" the dataset $f(x_i) \approx y_i$
- More concrete: pick a **loss function** to measure this: $\ell(f(x), y)$
- Training loss/empirical loss/empirical risk

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

Loss / Cost / Objective Function

- Find a *f* that minimizes the loss on the training data
  - Empirical Risk Minimization (ERM)

Loss Functions

What should the loss look like?

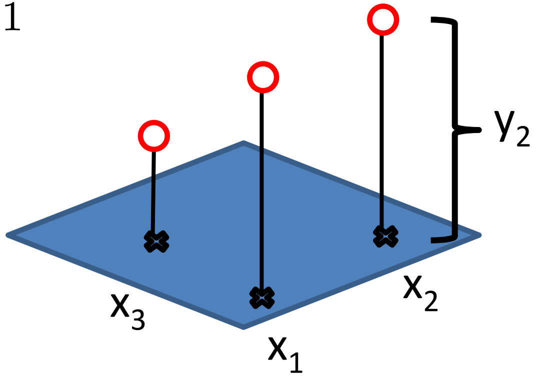
- If $f(x_i) \approx y_i$, should be small (0 if equal!)
- For classification: 0/1 loss $\ell(f(x), y) = 1\{f(x_i) \neq y_i\}$
- For regression, square loss $\ell(f(x), y) = (f(x_i) - y_i)^2$

Others too! We'll see more.

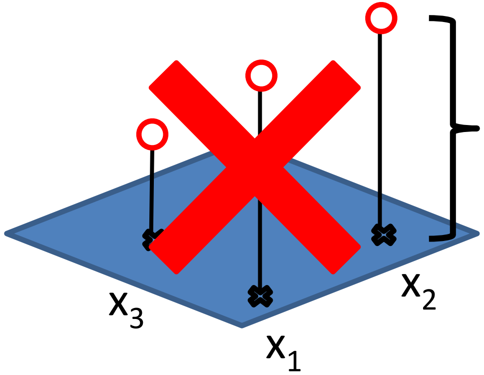
Functions/Models

The function *f* is usually called a model

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



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* →

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

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

← Linear model class *f*

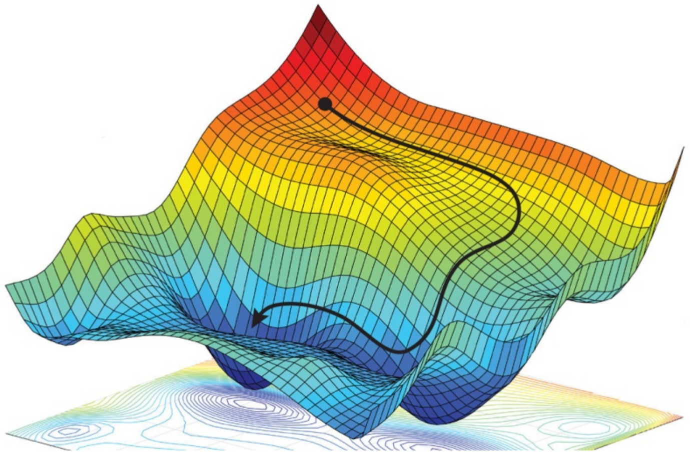
$$= \frac{1}{n} \sum_{i=1}^{n} (\theta_0 + x_i^T \theta - y_i)^2$$

← Square loss

How Do We Minimize?

- Need to solve something that looks like $\min_{\theta} g(\theta)$
- Generic optimization problem; many algorithms
  - A popular choice: **stochastic gradient descent (SGD)**

- Most algorithms iterative: find some sequence of points heading towards the optimum



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, create a **test** set. Do **not** train on it!

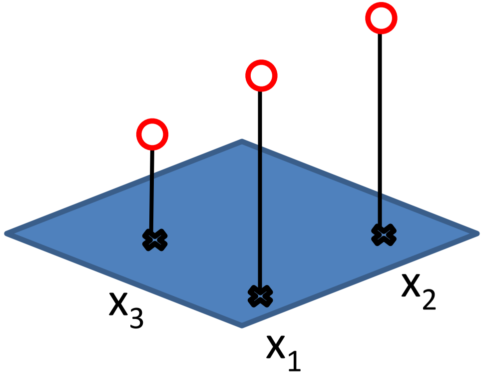
$$\underbrace{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)}_{\text{Training Data}} \quad \underbrace{(\mathbf{x}_{n+1}, y_{n+1}), \ldots, (\mathbf{x}_{n+p}, y_{n+p})}_{\text{Test Data}}$$

Do NOT train on your test set!!!

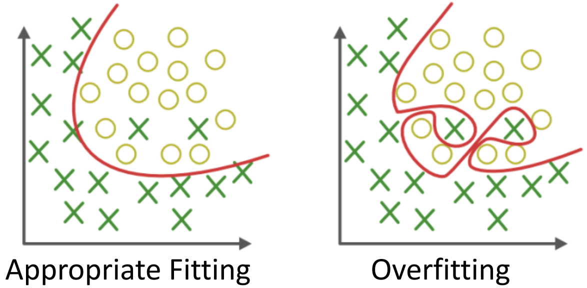
Train vs Test

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 *xi* 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

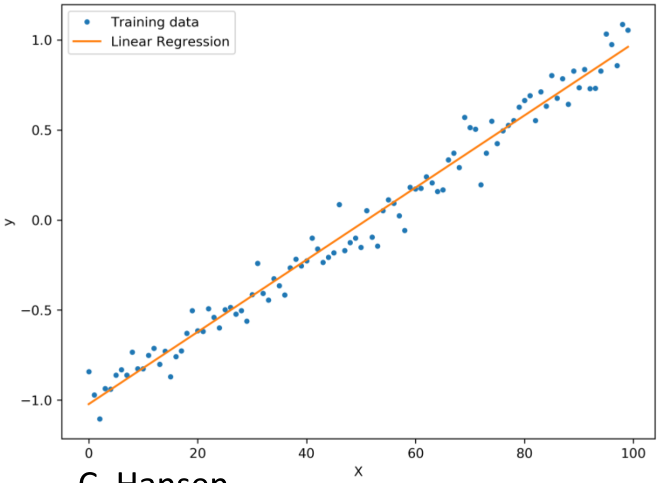
- **Q 1.3**: You have trained a classifier, and you find there is significantly **lower** loss on the test set than the training set. What is likely the case?

- **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 type of regression problem.

- **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 $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)$
  - Since $f(x) = \theta_0 + x^T\theta$, wrap intercept: $f(x) = x^T\theta$
  - Take train data and make it a matrix/vector:
  - Then, square loss is

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$$

$$\frac{1}{n}\sum_{i=1}^{n}(x_i^T\theta - y_i)^2 = \frac{1}{n}\|X^T\theta - y\|^2$$

Finding The Optimal Parameters

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

- Could optimize it with SGD, etc…
- No need: minimum has a solution (easy with vector calculus)

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

Hat: indicates an estimate

Not always invertible…

“Normal Equations”

How Good are the Optimal 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!

Train/Test for Linear Regression?

So far, residuals measure error on **train** set

- Sometimes that's all we care about (**Fixed Design LR**)
  - Data is deterministic.
  - Goal: find best linear relationship on dataset

- Or, create a test set and check (**Random Design LR**)
  - Common: assume data is $y = \theta^T x + \varepsilon$
  - The more noise, the less linear

0-mean
Gaussian noise

Solving With Gradient Descent

What if we don't know the exact solution?

- Use one of the iterative algorithms to do $\min_{\theta} \ell(\theta)$
- Among the most popular: **gradient descent**
- Basic idea: start at $\theta^{(0)}$
  - Next step: do $\theta^{(j+1)} = \theta^{(j)} - \gamma \nabla \ell(\theta^{(j)})$

Next solution

Current solution

Learning Rate (a constant)

Gradient of the loss, evaluated at current sol.

  - Run till convergence. (You'll implement this in HW5!)

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]

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

Logistic function

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 regression dataset created from a quadratic process (i.e., $y_i = ax_i^2 + bx_i + c$). Predict what might happen if you run linear regression on this data set.

- A. Linear regression will overfit the data.
- B. Linear regression will under-fit the data.
- C. Linear regression will neither overfit nor under-fit the data.
- D. Not enough information to say.

Break & Quiz

- **Q 2.1**: You have a regression dataset created from a quadratic process (i.e., $y_i = ax_i^2 + bx_i + c$). Predict what might happen if you run linear regression on this data set.

- A. Linear regression will overfit the data.
- **B. Linear regression will under-fit the data.**
- C. Linear regression will neither overfit nor under-fit the data.
- D. Not enough information to say.

Break & Quiz

- **Q 2.1**: You have a regression dataset created from a quadratic process (i.e., $y_i = ax_i^2 + bx_i + c$). Predict what might happen if you run linear regression on this data set.

- A. Linear regression will overfit the data.
- B. Linear regression will under-fit the data. (Cannot represent non-linear relationship)
- C. Linear regression will neither overfit nor under-fit the data.
- D. Not enough information to say.

Causal Interpretation

- Linear regression captures associations between features and outputs.
- Require causal assumptions to draw causal conclusions.
- 2021 Nobel Prize in Economics
  - Joshua Angrist and Guido Imbens
  - *“for their methodological contributions to the analysis of causal relationships”*