

# DSC 140B

## Representation Learning

Lecture 18 | Part 1

Prediction

convex opt.

non-convex opt.

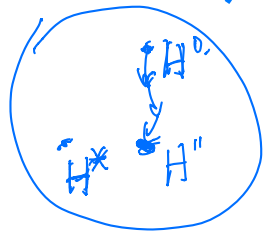
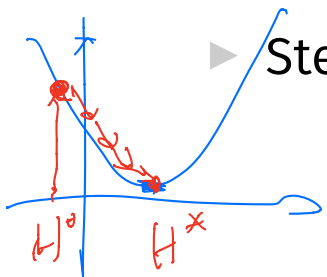
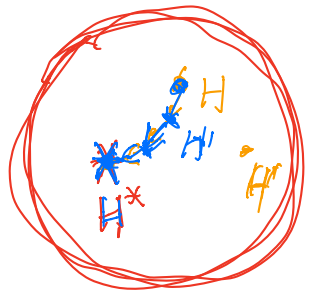
~~ANN~~ ~~CNN~~

# Empirical Risk Minimization (ERM)



- ▶ Step 1: choose a hypothesis class
- ▶ Step 2: choose a loss function
- ▶ Step 3: minimize expected loss (empirical risk)

linear

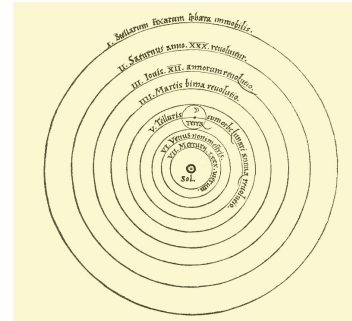
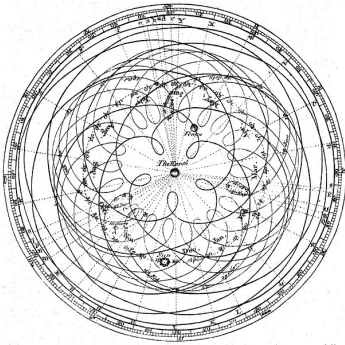


# Hypothesis Classes

- ▶ A **hypothesis class**  $\mathcal{H}$  is a set of possible prediction functions.
- ▶ By choosing a hypothesis class, we are saying something about what the prediction function should look like.
- ▶ Examples:
  - ▶  $\mathcal{H} :=$  linear functions
  - ▶  $\mathcal{H} :=$  functions of the form  $\sin(w_1 x_1 + \dots x_5 x_5)$
  - ▶  $\mathcal{H} :=$  decision trees of depth 10
  - ▶  $\mathcal{H} :=$  neural networks with one layer

# Hypothesis Class Complexity

- ▶ The more complex the hypothesis class, the greater the danger of **overfitting**.
  - ▶ Think: polynomials of degree 10 versus 2.
- ▶ Occam's Razor: assume  $H$  is simple.



# DSC 140B

## Representation Learning

Lecture 18 | Part 2

Least Squares Regression

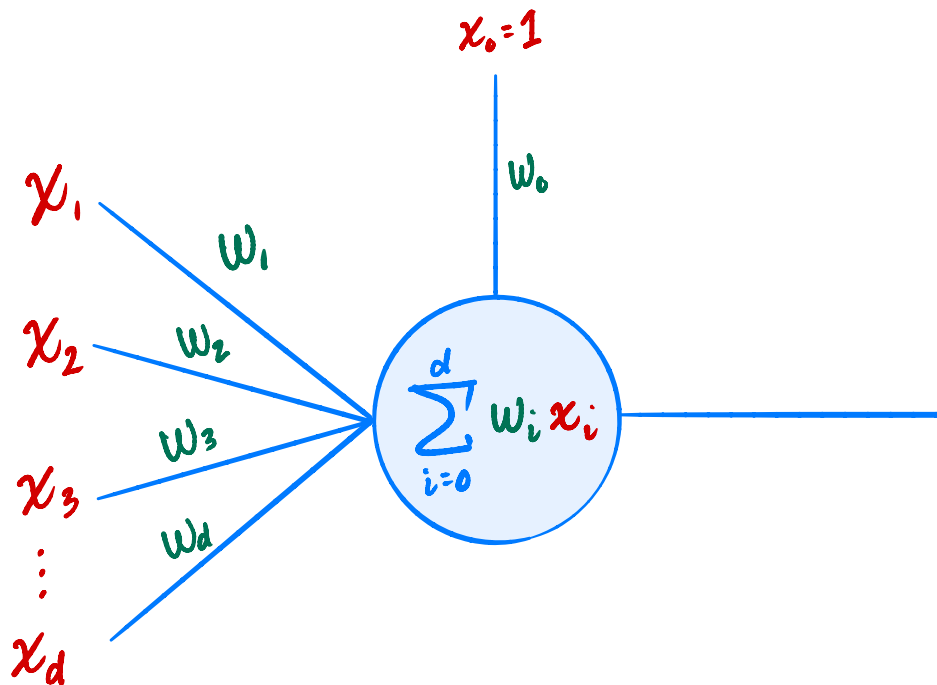
# Linear Prediction Functions

$$H(\vec{X}) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$

Hyp class

- ▶ This is a **linear prediction function**.
- ▶  $w_0, w_1, \dots, w_5$  are the **parameters** or **weights**.
- ▶  $\vec{w} = (w_0, \dots, w_5)^T$  is a **parameter vector**.

# Linear Predictors



# Class of Linear Functions

- ▶ There are infinitely many functions of the form

$$H(\vec{X}) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5$$

- ▶ Each one is completely determined by  $\vec{w}$ .
  - ▶ Sometimes write  $H(\vec{x}; \vec{w})$

- ▶ Example:  $\vec{w} = (8, 3, 1, 5, -2, -7)^T$  specifies

$$H(\vec{x}; \vec{w}) = 8 + 3x_1 + 1x_2 + 5x_3 - 2x_4 - 7x_5$$



$H(\vec{x})$   $\mathbb{R}$   $\theta$   $NN_{\theta}$   $P_{\theta}(\vec{x})$   $P(\vec{x}; \theta)$   
 $P(\vec{x}; \vec{w})$

# “Parameterization”

- ▶ A very useful trick.
- ▶ Searching all linear functions  $\equiv$  searching over  
 $\vec{w} \in \mathbb{R}^6$

$\vec{w}^*$

# In General

- ▶ If there are  $d$  features, there are  $d + 1$  parameters:

$$H(\vec{X}) = W_0 + W_1X_1 + W_2X_2 + \dots + W_dX_d \quad \checkmark$$
$$= W_0 + \sum_{i=1}^d W_iX_i$$

*Handwritten annotations:*

- A red arrow points from the word "bias" to  $W_0$  in the first equation.
- A red arrow points from the word "weights" to  $W_iX_i$  in the second equation.
- A red arrow points from the word "bias" to  $W_0$  in the second equation.

# Linear Prediction and the Dot Product

- ▶ The **augmented feature vector**  $\text{Aug}(\vec{x})$  is the vector obtained by adding a 1 to the front of  $\vec{x}$ :

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix} \quad \text{Aug}(\vec{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$$

# Simplification

- ▶ With augmentation, we can write as dot product:

$$\begin{aligned} H(\vec{x}) &= w_0 + w_1 x_1 + w_2 x_2 + \dots + w_d x_d \\ &= \text{Aug}(\vec{x}) \cdot \vec{w} \end{aligned}$$

$$\vec{w} = \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_d \end{pmatrix} \quad \text{Aug}(\vec{x}) = \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{pmatrix}$$

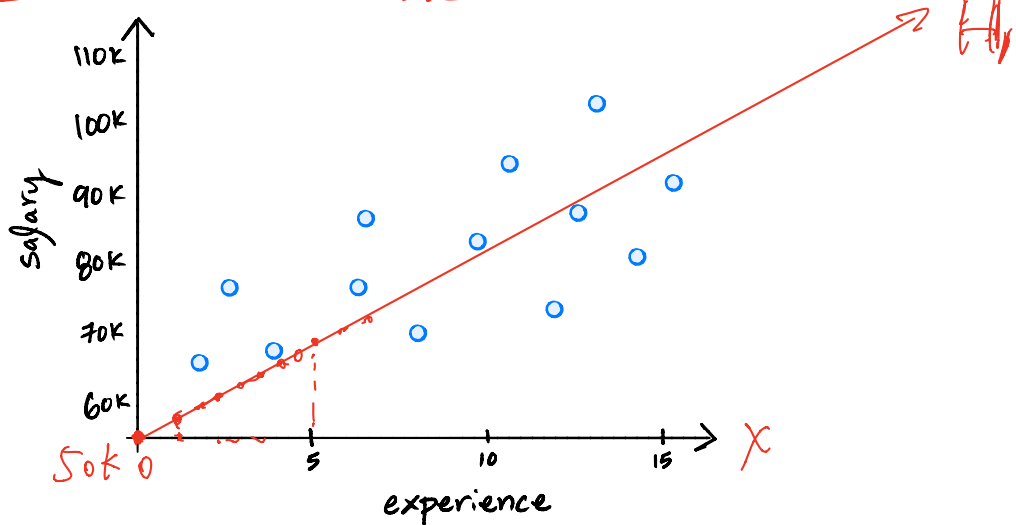
# Geometric Meaning

- ▶ It can be very useful to **think geometrically** when reasoning about prediction algorithms.

# Example

- ▶ A linear prediction function for salary.

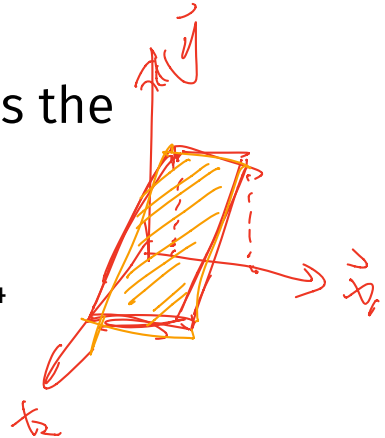
$$H_1(\vec{x}) = \overset{w_0}{\$50,000} + (\text{experience}) \times \overset{w_1}{\$8,000}$$



$$H((x_1, x_2))$$

# Surface

- ▶ The **surface** of a prediction function  $H$  is the surface made by plotting  $H(\vec{x})$  for all  $\vec{x}$ .
- ▶ If  $H$  is a linear prediction function, and<sup>4</sup>
  - ▶  $\vec{x} \in \mathbb{R}^1$ , then  $H(x)$  is a straight line.
  - ▶  $\vec{x} \in \mathbb{R}^2$ , the surface is a plane.
  - ▶  $\vec{x} \in \mathbb{R}^d$ , the surface is a  $d$ -dimensional **hyperplane**.



---



<sup>4</sup>when plotted in the original feature coordinate space!

# Empirical Risk Minimization (ERM)

- ▶ Step 1: choose a **hypothesis class**
  - ▶ Let's assume we've chosen linear predictors
- ▶ Step 2: choose a **loss function**
- ▶ Step 3: minimize **expected loss (empirical risk)**



## Step #2: Choose a loss function

- ▶ Suppose we assume prediction function is linear.
- ▶ There are still infinitely-many possibilities. 
- ▶ We'll pick one that works well on training data.
- ▶ What does “works well” mean? 

# Example: Movie Ratings

| Movie                | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | You |
|----------------------|-------|-------|-------|-------|-------|-----|
| #1                   | 8     | 5     | 9     | 2     | 1     | 6   |
| $i=2 \rightarrow$ #2 | 3     | 5     | 7     | 8     | 2     | 8   |
| #3                   | 1     | 5     | 2     | 3     | 3     | 9   |
| #4                   | 0     | 5     | 3     | 8     | 2     | ?   |

# Quantifying Quality

- ▶ Consider a training example  $(\vec{x}^{(i)}, y_i)$ 
  - ▶ Notation:  $\vec{x}^{(i)}$  is the “ $i$ th training example”
  - ▶  $\vec{x}_j^{(i)}$  is the “ $j$ th entry of the  $i$ th training example”
- ▶ The “right answer” is  $y_i$
- ▶ Our prediction function outputs  $H(\vec{x}^{(i)})$
- ▶ We measure the difference using a **loss function**.

# Loss Function

- ▶ A **loss function** quantifies how wrong a single prediction is.

$$L(\underbrace{H(\vec{x}^{(i)})}_{\text{prediction}}, \underbrace{y_i}_{\text{correct answer}})$$

$L$ (prediction for example  $i$ , correct answer for example  $i$ )

# Empirical Risk

- ▶ A good  $H$  is good on average over entire data set.
- ▶ The expected loss (or empirical risk) is one way of measuring this:

$$R(H) = \frac{1}{n} \sum_{i=1}^n L(H(\vec{x}^{(i)}), y_i)$$

$R(H; D)$       loss function.

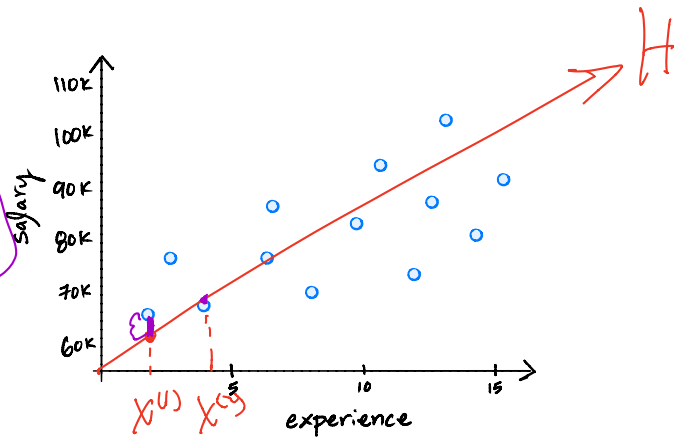
- ▶ Note: depends on  $H$  and the data!

# Loss Functions for Regression

▶ We want  $H(\vec{x}^{(i)}) \approx y_i$ .

▶ **Absolute loss:**  $|H(\vec{x}^{(i)}) - y_i|$

▶ **Square loss:**  $(H(\vec{x}^{(i)}) - y_i)^2$

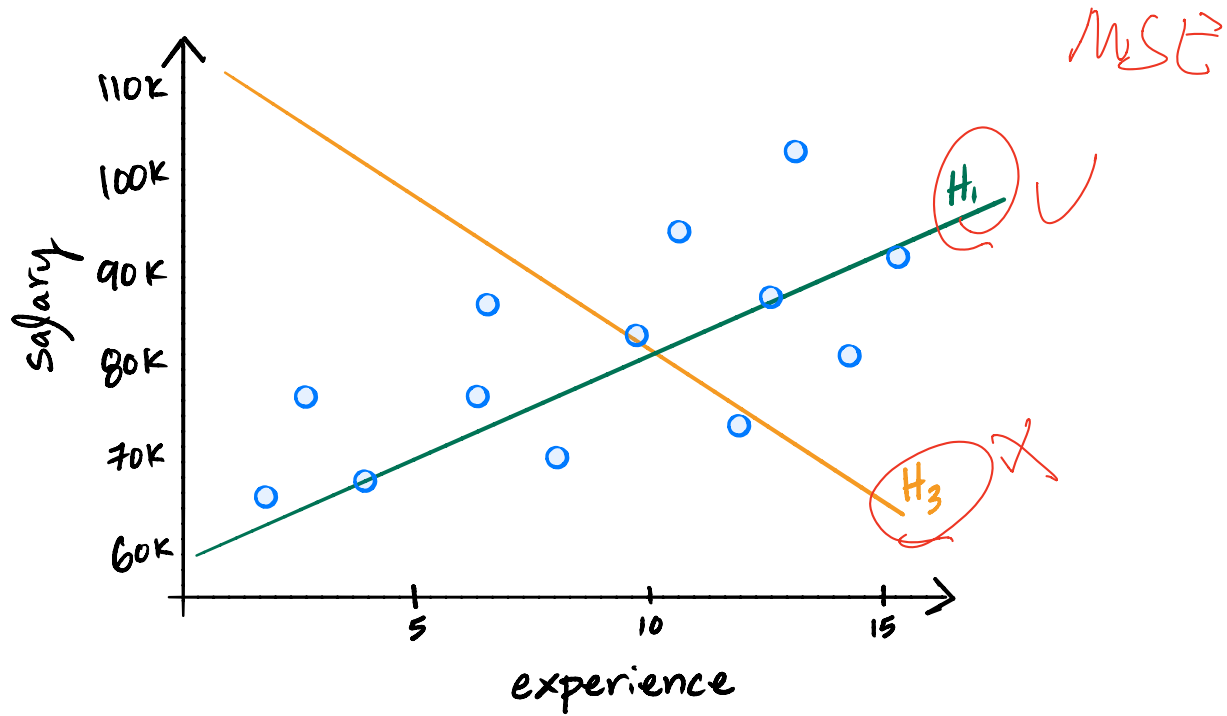


# Mean Squared Error

- ▶ **Expected square loss** (mean squared error):

$$R_{\text{sq}}(H) = \frac{1}{n} \sum_{i=1}^n (H(\vec{x}^{(i)}) - y_i)^2$$

- ▶ This is the **empirical risk** for the square loss.
- ▶ Goal: find  **$H$**  minimizing **MSE**.





# Step #3: Minimize MSE

- ▶ We want to find an  $H$  minimizing this:

$$\underline{R_{\text{sq}}(H)} = \frac{1}{n} \sum_{i=1}^n (H(\vec{x}^{(i)}) - y_i)^2$$

- ▶ It helps to use linear assumption:

$$\underline{R_{\text{sq}}(\vec{W})} = \frac{1}{n} \sum_{i=1}^n (\vec{W} \cdot \underline{\text{Aug}(\vec{x}^{(i)})} - y_i)^2$$

# Calculus

- ▶ We want to find  $\vec{w}$  that minimizes the average square loss:

$$R_{sq}(\vec{w}) = \frac{1}{n} \sum_{i=1}^n (\vec{w} \cdot \text{Aug}(\vec{x}^{(i)}) - y_i)^2$$

~~max~~  
min  $R_{sq}(\vec{w})$

- ▶ Take the gradient, set to  $\vec{0}$ , solve.

$$\frac{\nabla R_{sq}(\vec{w})}{\nabla \vec{w}} = \vec{0}$$

$\vec{w}^*$

- ▶ Solution: the **Normal Equations**,  $\vec{w} = (X^t X)^{-1} X^t \vec{y}$

~~$\vec{w}$~~

# Design Matrix

- ▶  $X$  is the **design matrix**  $X$ :

$$X = \begin{pmatrix} \text{Aug}(\vec{x}^{(1)}) \longrightarrow & & & & \\ \text{Aug}(\vec{x}^{(2)}) \longrightarrow & & & & \\ \vdots & & & & \\ \text{Aug}(\vec{x}^{(n)}) \longrightarrow & & & & \end{pmatrix} = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \dots & x_d^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \dots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_1^{(n)} & x_2^{(n)} & \dots & x_d^{(n)} \end{pmatrix}$$

# Note

- ▶ There was a closed-form solution!
- ▶ This is a direct consequence of using the **mean squared error**.
- ▶ Not true if we use, e.g., the **mean absolute error**.

# Why linear?

- ▶ Easy to work with mathematically.
- ▶ Harder to overfit. ✓ *underfit*
- ▶ But still quite powerful.

# DSC 140B

## Representation Learning

Lecture 18 | Part 3

Least Squares Classifiers

# Movie Ratings

- ▶ Five of your friends rate a movie from 0-10:

- ▶  $x_1$ : 9

- ▶  $x_2$ : 3

- ▶  $x_3$ : 7

- ▶  $x_4$ : 2

- ▶  $x_5$ : 8

- ▶ **Task:** Will you like the movie? (yes / no)

# Classification

- ▶ Linear prediction functions can be used in classification, too.

$$H(\vec{X}) = \underbrace{w_0 + w_1 X_1 + w_2 X_2 + \dots + w_d X_d}$$

- ▶ Same ERM paradigm also useful.



# A Classifier from a Regressor

- ▶ Binary classification can be thought of as regression where the targets are 1 and -1
  - ▶ (or 0 and 1, or ...)
- ▶  $H(\vec{x})$  outputs a real number. Use the **sign** function to turn it into -1, 1:

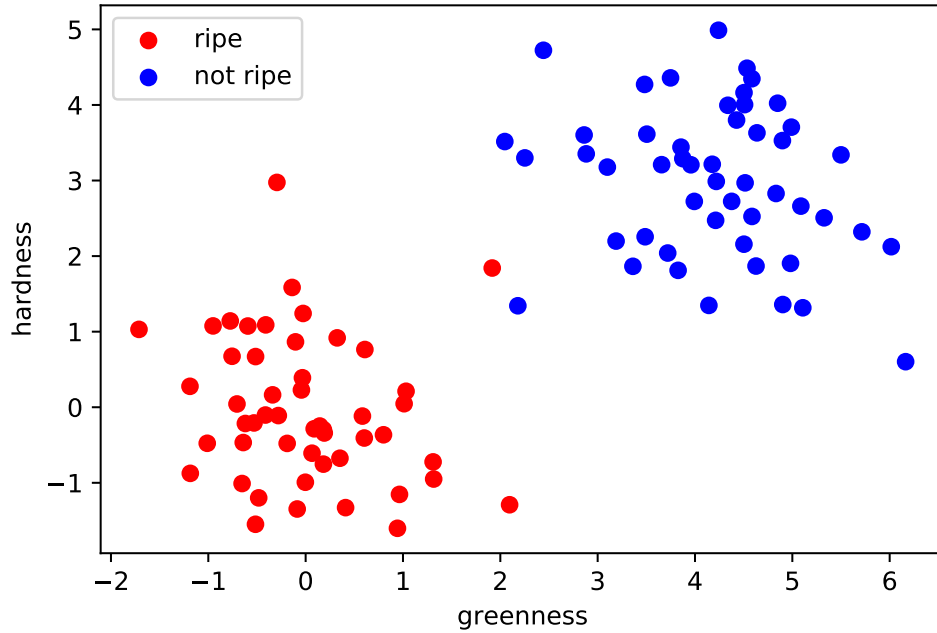
$$\text{sign}(z) = \begin{cases} 1 & z > 0 \\ -1 & z < 0 \\ 0 & \text{otherwise} \end{cases}$$

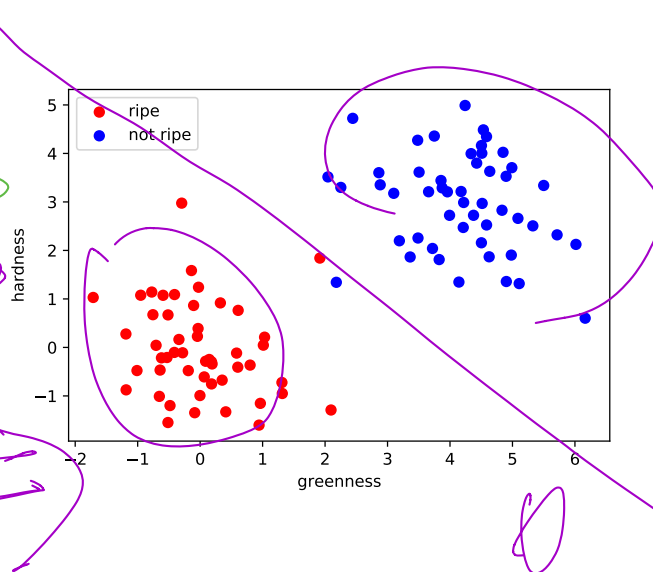
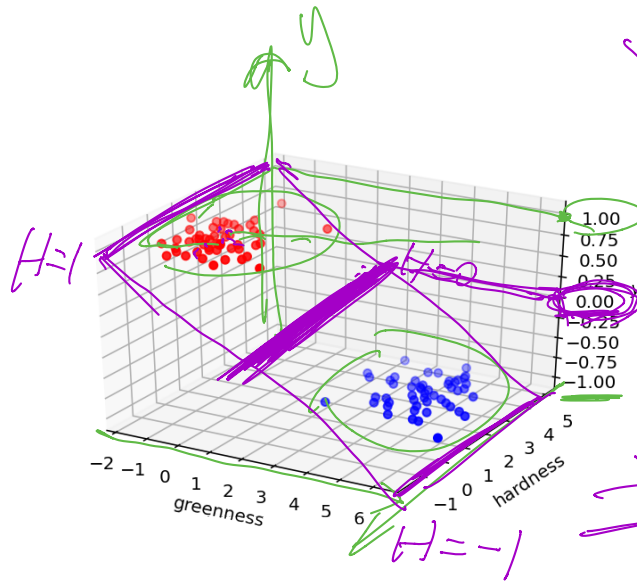
- ▶ Final prediction:  $\text{sign}(H(\vec{x}))$

# Example: Mango Ripeness

- ▶ Predict whether a mango is ripe given greenness and hardness.
- ▶ Idea: gather a set of labeled **training data**.
  - ▶ Inputs along with correct output (i.e., “the answer”).

| <u>Greenness</u> | <u>Hardness</u> |  | Ripe |
|------------------|-----------------|--|------|
| 0.7              | 0.9             |  | 1    |
| 0.2              | 0.5             |  | -1   |
| 0.3              | 0.1             |  | -1   |
| ⋮                | ⋮               |  | ⋮    |





$H(\vec{x})$   
 Surface

# Decision Boundary

- ▶ The **decision boundary** is the place where the output of  $H(x)$  switches from “yes” to “no”.
  - ▶ If  $H > 0 \mapsto$  “yes” and  $H < 0 \mapsto$  “no”, the decision boundary is where  $H = 0$ .
  
- ▶ If  $H$  is a linear predictor and<sup>5</sup>
  - ▶  $\vec{x} \in \mathbb{R}^1$ , then the decision boundary is just a number.
  - ▶  $\vec{x} \in \mathbb{R}^2$ , the boundary is a straight line.
  - ▶  $\vec{x} \in \mathbb{R}^d$ , the boundary is a  $d - 1$  dimensional (hyper) plane.

---

<sup>5</sup>when plotted in the original feature coordinate space!

# Empirical Risk Minimization

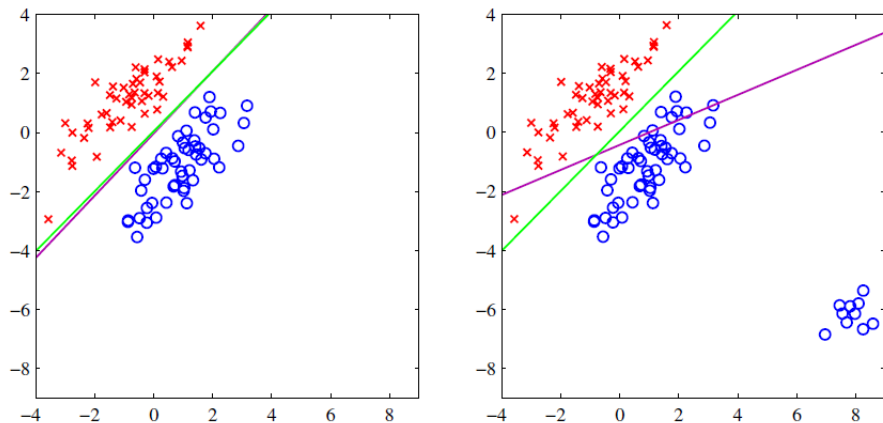
- ▶ Step 1: choose a **hypothesis class**
  - ▶ Let's assume we've chosen linear predictors
- ▶ Step 2: choose a **loss function**
- ▶ Step 3: minimize **expected loss (empirical risk)**

## Exercise

Can we use the square loss for classification?

$$(H(\vec{x}^{(i)}) - y_i)^2$$

# Least Squares and Outliers



**Figure 4.4** The left plot shows data from two classes, denoted by red crosses and blue circles, together with the decision boundary found by least squares (magenta curve) and also by the logistic regression model (green curve), which is discussed later in Section 4.3.2. The right-hand plot shows the corresponding results obtained when extra data points are added at the bottom left of the diagram, showing that least squares is highly sensitive to outliers, unlike logistic regression.



# Square Loss for Classification

- ▶ We **can** use the square loss for classification
  - ▶ The “least squares classifier”
- ▶ However, the square loss penalizes being “too correct”
- ▶ **Example:** suppose the correct label is 1. What is the square loss of predicting 10? -9?

# Loss Functions

- ▶ There are many different loss functions for classification.
- ▶ Each leads to a different classifier:
  - ▶ Logistic Regression
  - ▶ Support Vector Machine
  - ▶ Perceptron
  - ▶ etc.
- ▶ But that's for another class... (DSC 140A)

# DSC 140B

## Representation Learning

Lecture 18 | Part 4

**Linear Limitations**

# Linear Predictors

- ▶ Last time, we saw linear prediction functions:

$$\begin{aligned} H(\vec{x}; \vec{w}) &= w_0 + w_1 x_1 + \dots + w_d x_d \\ &= \text{Aug}(\vec{x}) \cdot \vec{w} \end{aligned}$$

# Linear Decision Functions

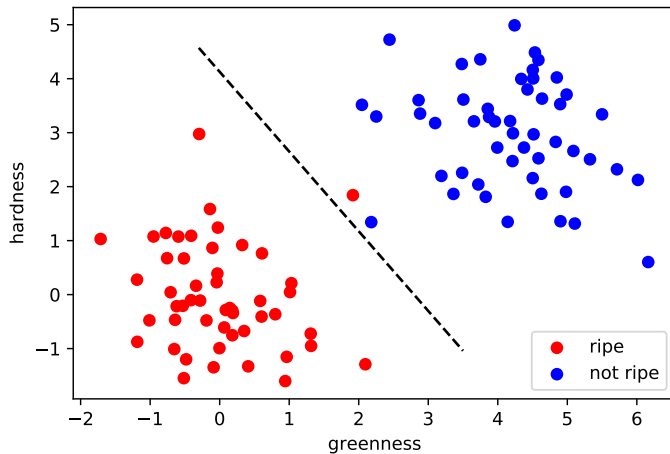
- ▶ A linear prediction function  $H$  outputs a number.
- ▶ What if classes are +1 and -1?
- ▶ Can be turned into a **decision function** by taking:

$$\text{sign}(H(\vec{x}))$$

- ▶ **Decision boundary** is where  $H = 0$ 
  - ▶ Where the sign switches from positive to negative.

# Decision Boundaries

- ▶ A linear decision function's decision boundary is linear.
  - ▶ A line, plane, hyperplane, etc.



# An Example: Parking Predictor

- ▶ **Task:** Predict (yes / no): Is there parking available at UCSD right now?
- ▶ What training data to collect? What features?

# Useful Features

- ▶ Time of day?
- ▶ Day's high temperature?
- ▶ ...



## Exercise

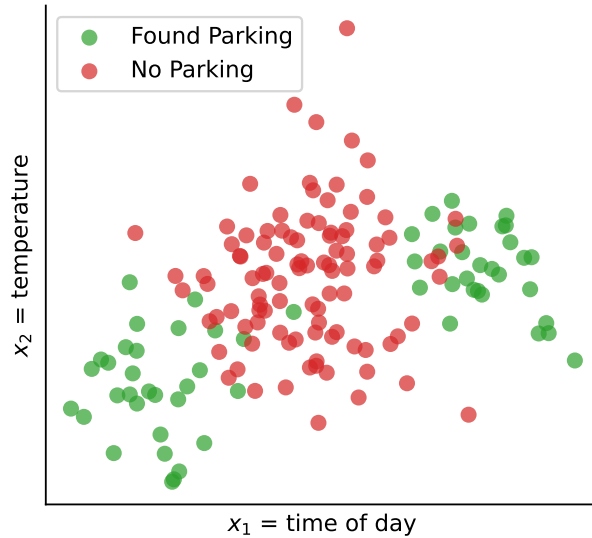
Imagine a scatter plot of the training data with the two features:

- ▶  $x_1$  = time of day
- ▶  $x_2$  = temperature

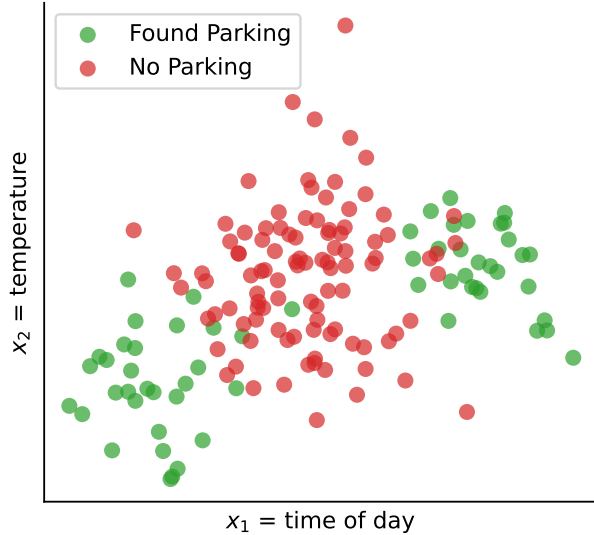
“yes” examples are green, “no” are red.

What does it look like?

# Parking Data



# Uh oh



- ▶ A linear decision function won't work.
- ▶ What do we do?