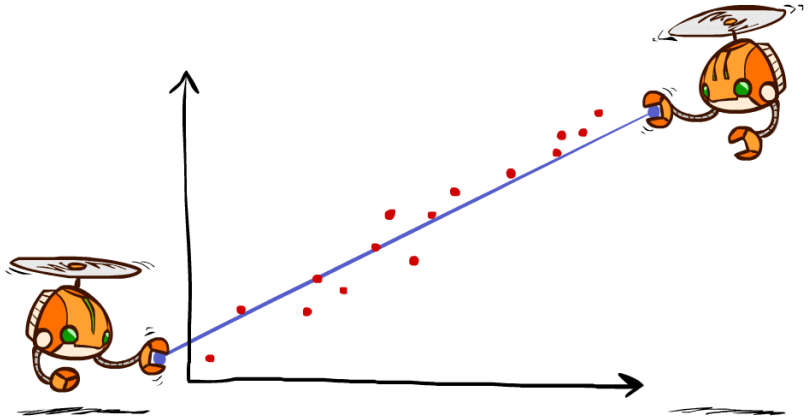
Machine Learning II: Regression

CSE 473: Introduction to Artificial Intelligence



Megan Hazen(mh75@cs)
cs.uw.edu/473
acknowledgements

Welcome!
How is a line like an elephant?

Administrivia

ANNOUNCEMENTS

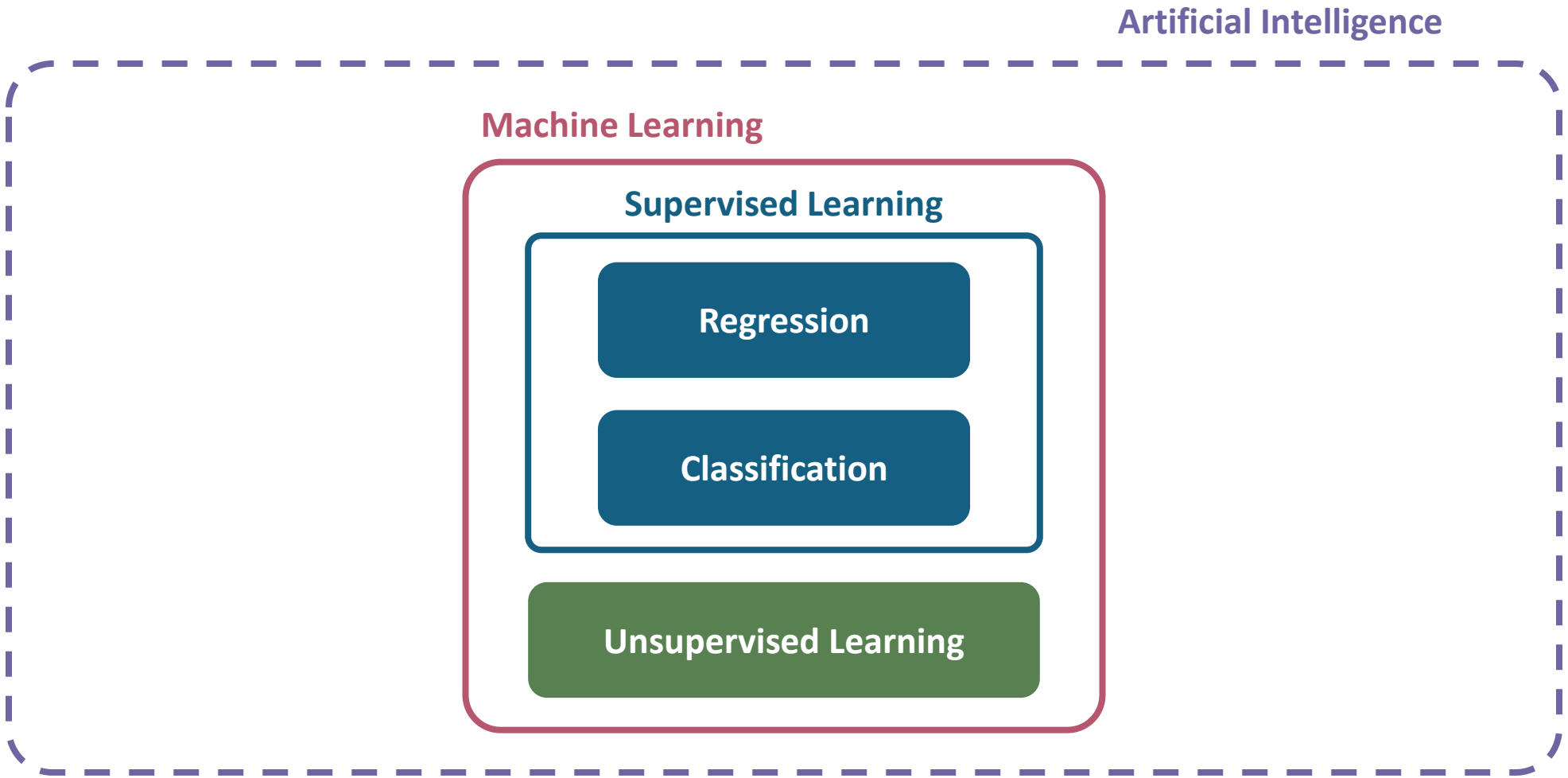
ASSIGNMENTS

- **Project 3** due Friday
- **Homework 4** posted



Mapping Machine Learning

CONTEXT



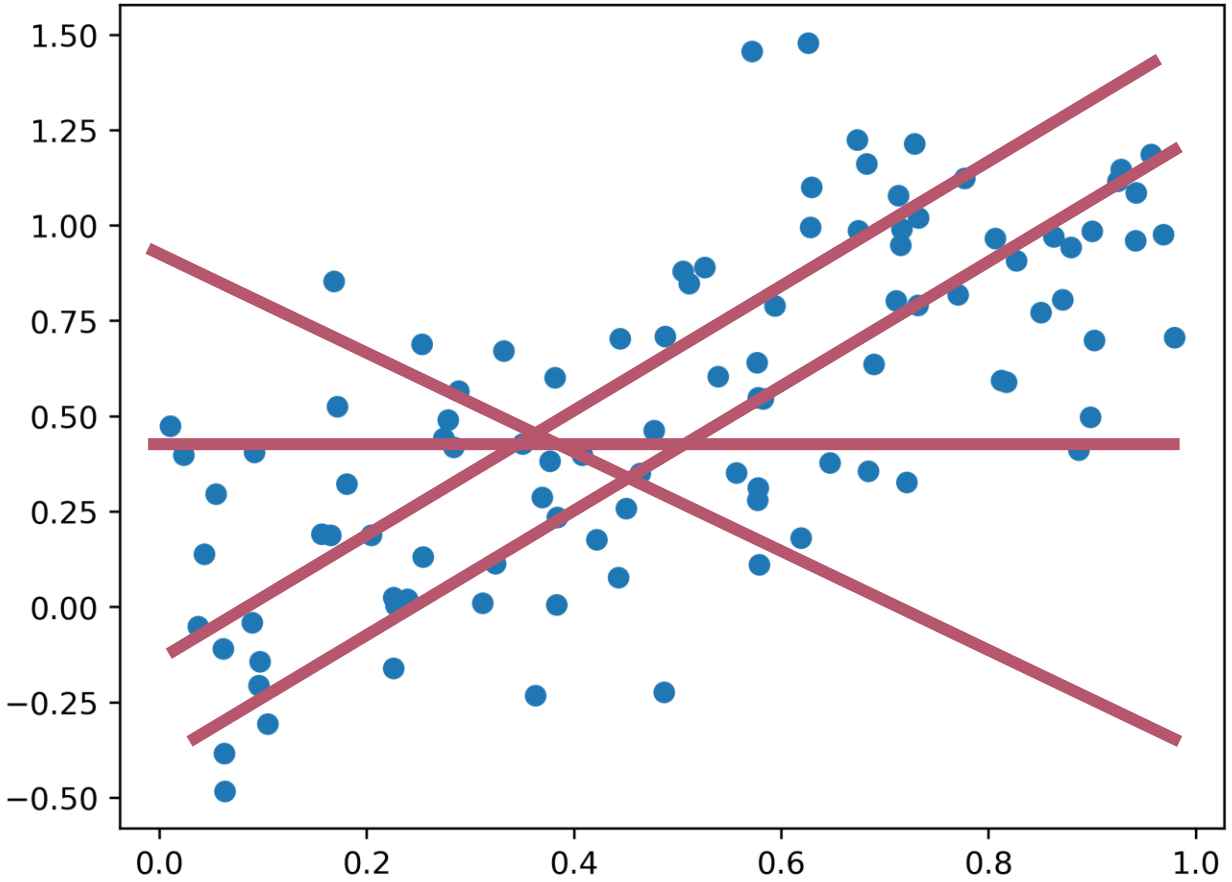
Supervised Learning

FOUNDATIONS

Goal: Learn unknown **target function** f

- Input: **Training data** with *labeled* examples (x_i, y_i), where $f(x_i) = y_i$
- Output: **Hypothesis** h that is "close" to f
  - $h \approx f$ implies that h predicts well on unseen (test set) data
- Knowledge: Family of hypotheses H to choose h from
  - Linear models, logistic regression, neural networks, decision trees, non-parametric models, etc.
- Classification: Learn f with discrete output value
- Regression: Learn f with continuous output value

What is the best fit?



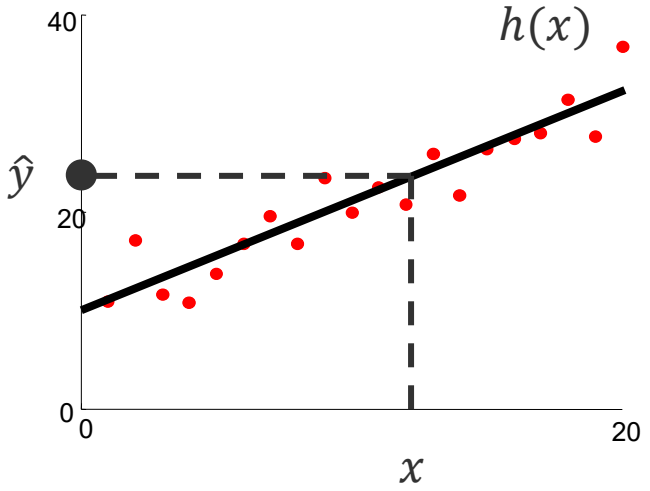
Linear Regression

FOUNDATIONS

- A regression problem finds h to approximate

$$f\colon \mathbb{R}^n \to \mathbb{R}^1$$

  - $y = f(\boldsymbol{x}) + \epsilon$, irreducible noise $\epsilon \sim N(0, \sigma^2)$
  - $h \in \mathcal{H}_{\text{linear}}$



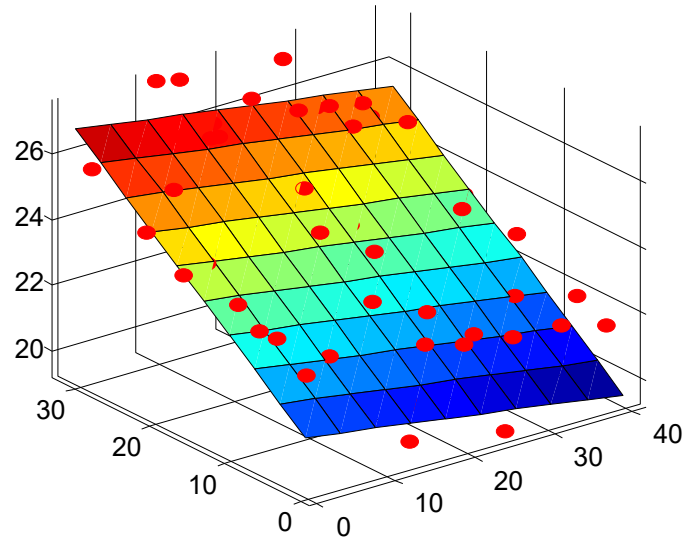
- Approximation is linear model of the form

$$\hat{y} = h_{\boldsymbol{w}}(\boldsymbol{x}) = w_0 + w_1 x_1 + \cdots + w_n x_n$$

- Best hypothesis minimizes

$$\hat{h}^* = \text{argmin}_{h \in \mathcal{H}} \mathcal{L}(h, x, y)$$

  - for loss function $\mathcal{L}$

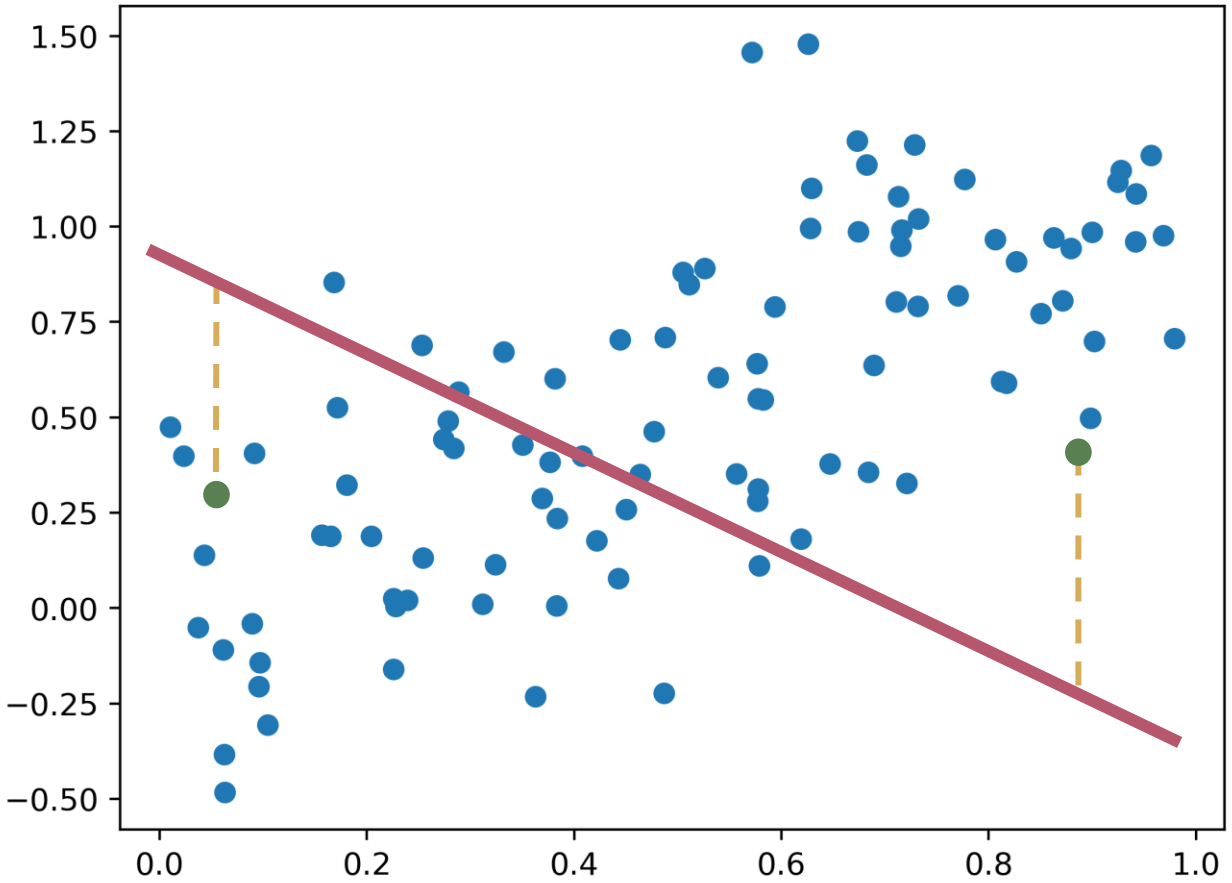


Measuring Loss

CONTEXT

Error measures the difference between prediction and ground truth:

$$\text{error} = \text{y}_{\text{actual}} - h(x)$$



Loss describes the *total* (unsigned) deviation from the desired values

$$\text{SSE} = \sum_j (y_j - h(x_j))^2$$

Least Squares Regression

DEFINITION

- Define $\mathcal{L}$ as sum of squared error between predicted and actual

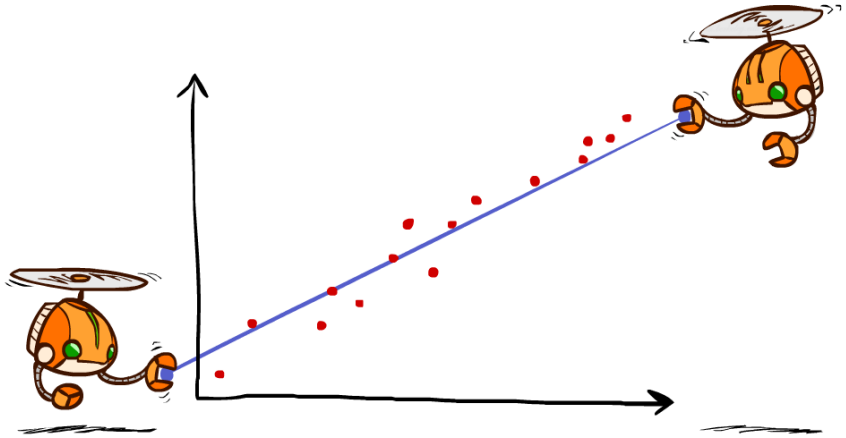
$$\mathcal{L}_2 = \sum_{j=1}^{N} (y_j - h_{\boldsymbol{w}}(\boldsymbol{x}_j))^2 = \sum_{j=1}^{N} (y_j - (w_0 + w_1 x_{j,1} + \cdots + w_n x_{j,n}))^2$$

- We want weights $\boldsymbol{w}^*$ that minimize $\mathcal{L}_2$
  - Solving via iterative optimization:

$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathcal{L}_2(\boldsymbol{w})$$

  - Solving via linear algebra:

$$\boldsymbol{w}^* = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$$

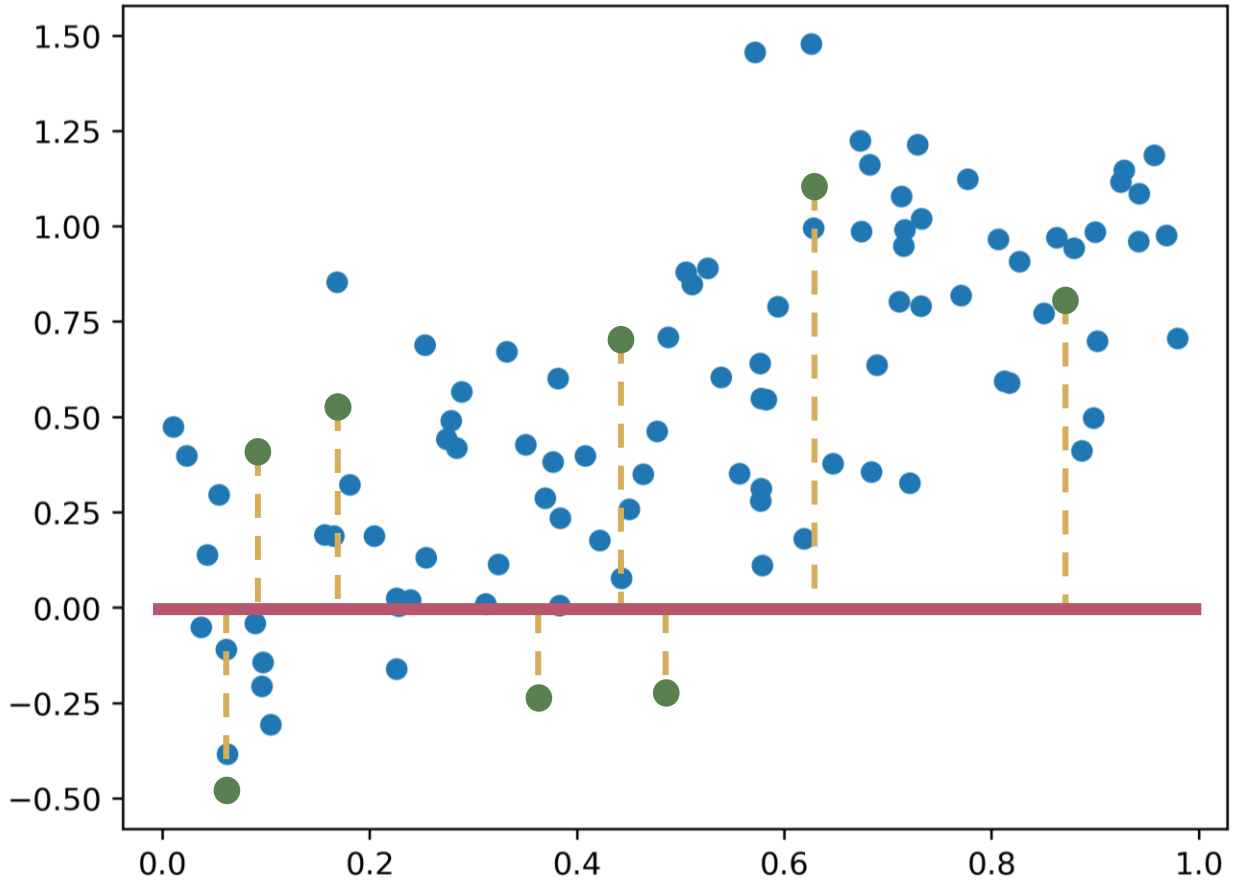


Questions?

Least Squares in Action

EXAMPLE

Initialize: $w_0 = w_1 = 0$



$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathcal{L}_2(\boldsymbol{w})$$

$$\mathcal{L}_2 = \sum_j (y_j - (w_0 + w_1 x))^2 = 46.36$$

$$\frac{\partial}{\partial w_1} \mathcal{L}_2 = -69.5$$

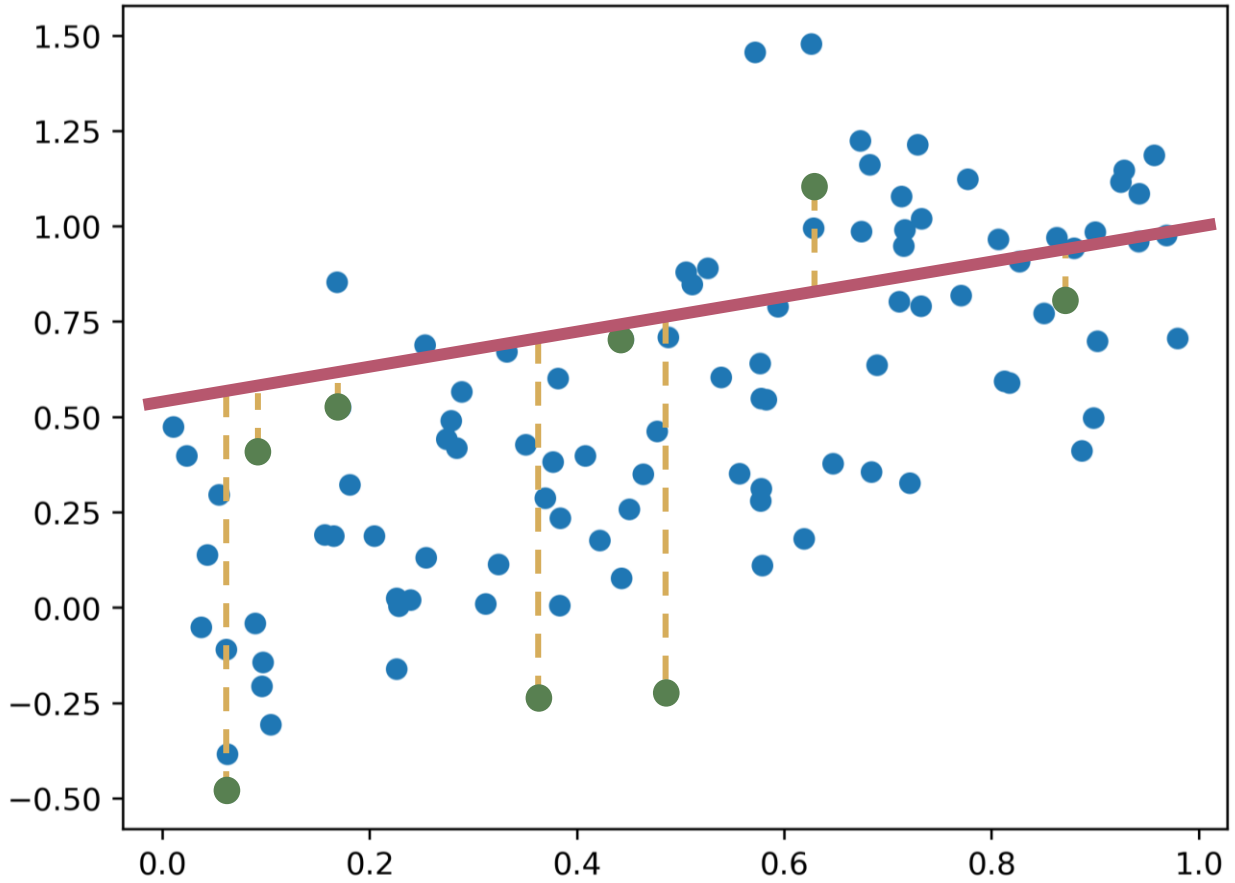
$$w_1 \leftarrow w_1 - (0.005) \cdot (-69.5) = 0.35$$

$$w_0 \leftarrow w_0 - (0.005) \cdot \frac{\partial}{\partial w_0} \mathcal{L}_2 = 0.53$$

Least Squares in Action (2)

EXAMPLE

Update:
$w_0 = 0.53$
$w_1 = 0.35$



$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathcal{L}_2(\boldsymbol{w})$$

$$\mathcal{L}_2 = \sum_j (y_j - (w_0 + w_1 x))^2 = 17.07$$

$$\frac{\partial}{\partial w_1} \mathcal{L}_2 = 6.72$$

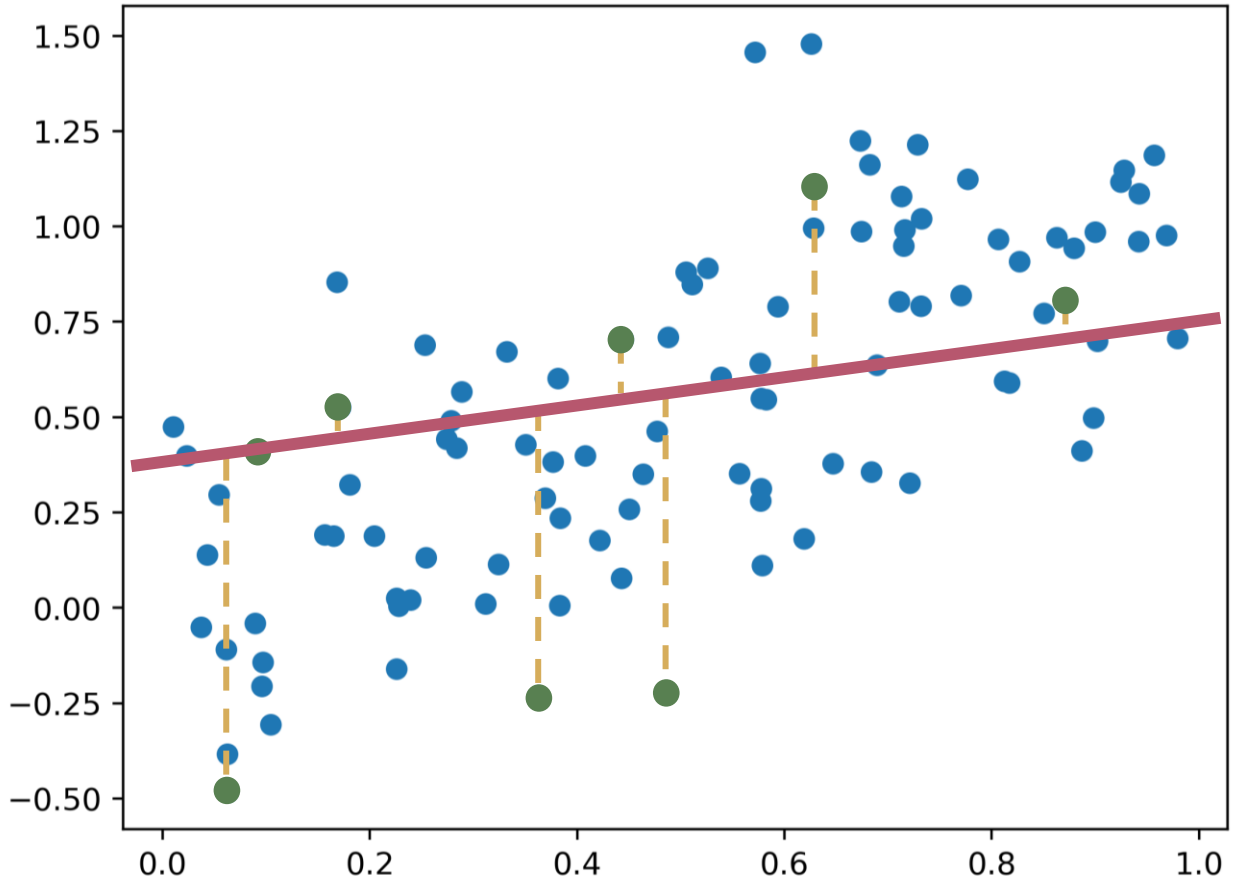
$$w_1 \leftarrow w_1 - (0.005) \cdot (6.72) = 0.32$$

$$w_0 \leftarrow w_0 - (0.005) \cdot \frac{\partial}{\partial w_0} \mathcal{L}_2 = 0.35$$

Least Squares in Action (3)

EXAMPLE

Update:
$w_0 = 0.35$
$w_1 = 0.32$



$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathcal{L}_2(\boldsymbol{w})$$

$$\mathcal{L}_2 = \sum_j (y_j - (w_0 + w_1 x))^2 = 14.14$$

$$\frac{\partial}{\partial w_1} \mathcal{L}_2 = -13.3$$

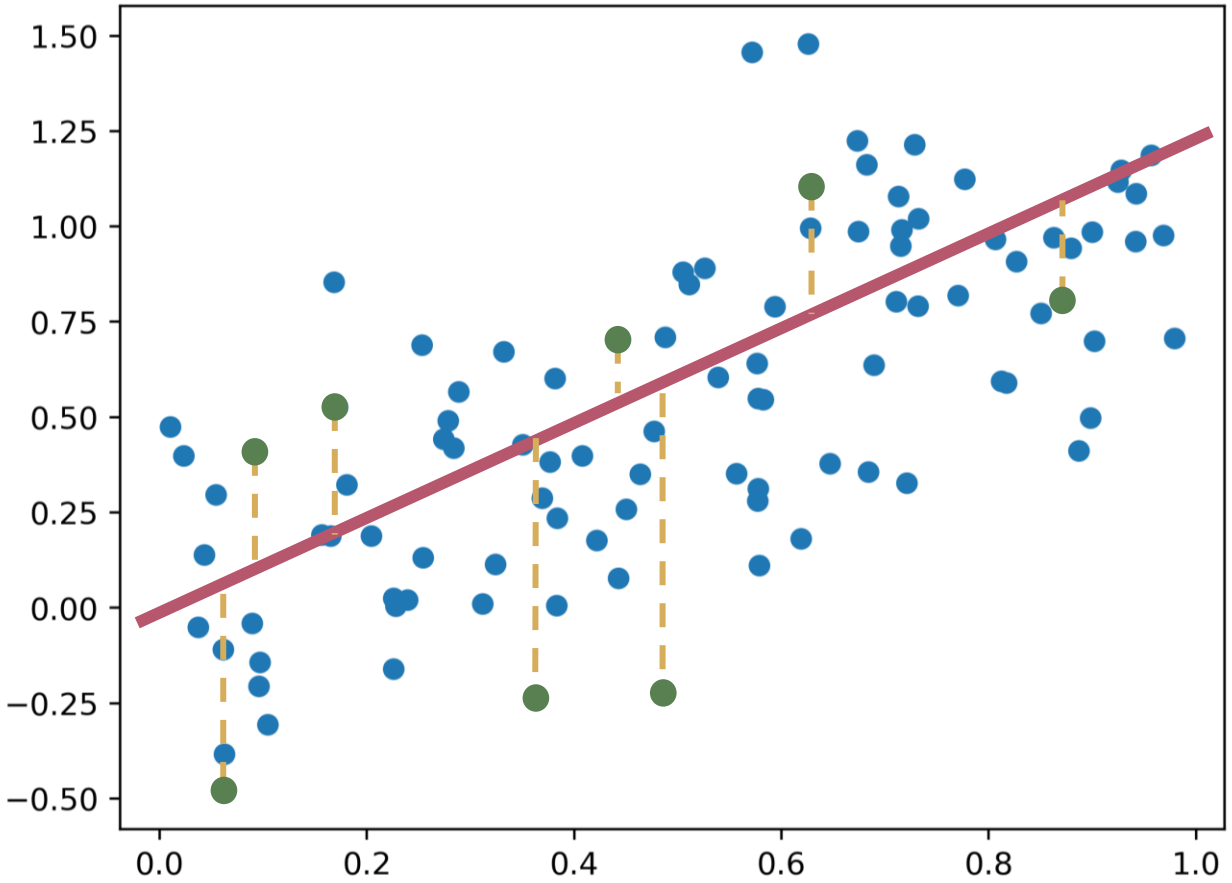
$$w_1 \leftarrow w_1 - (0.005) \cdot (-13.3) = 0.39$$

$$w_0 \leftarrow w_0 - (0.005) \cdot \frac{\partial}{\partial w_0} \mathcal{L}_2 = 0.36$$

Least Squares in Action (100)

EXAMPLE

Update:
$w_0 = -0.02$
$w_1 = 1.08$



$$w_i \leftarrow w_i - \alpha \frac{\partial}{\partial w_i} \mathcal{L}_2(\boldsymbol{w})$$

$$\mathcal{L}_2 = \sum_j (y_j - (w_0 + w_1 x))^2 = 9.69$$

Gradient Descent

DEFINITION

Gradient descent is a *greedy* optimization approach

- Take small steps in the direction of the steepest descent (negative gradient)

$$w \leftarrow w - \eta \frac{\partial}{\partial w} \mathcal{L} \qquad \eta \sim \alpha$$

  - Stop when updates become too small

- Considerations
  - How to find, calculate gradient?
  - Step size η matters a lot!
  - When to stop?



Questions?

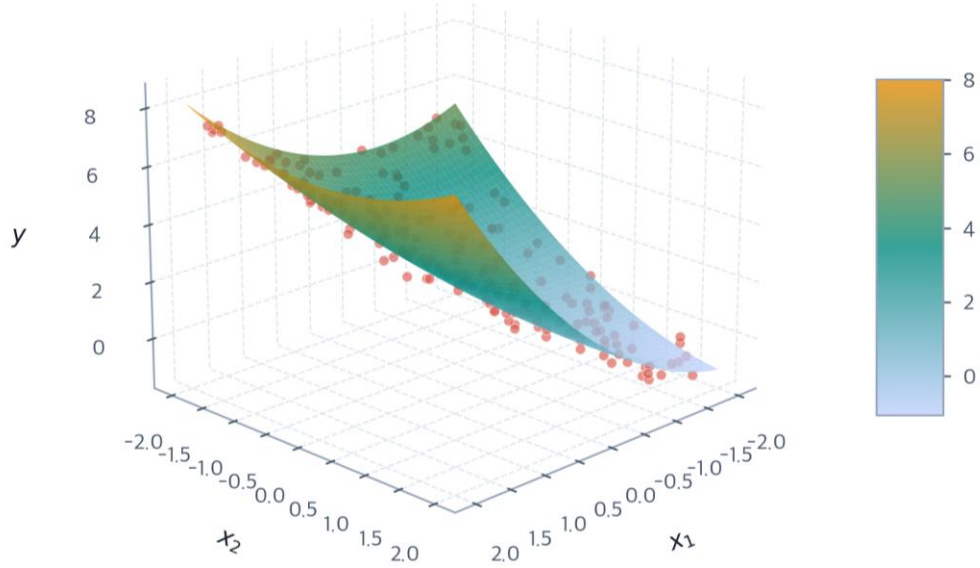
Polynomial Regression

DEFINITION

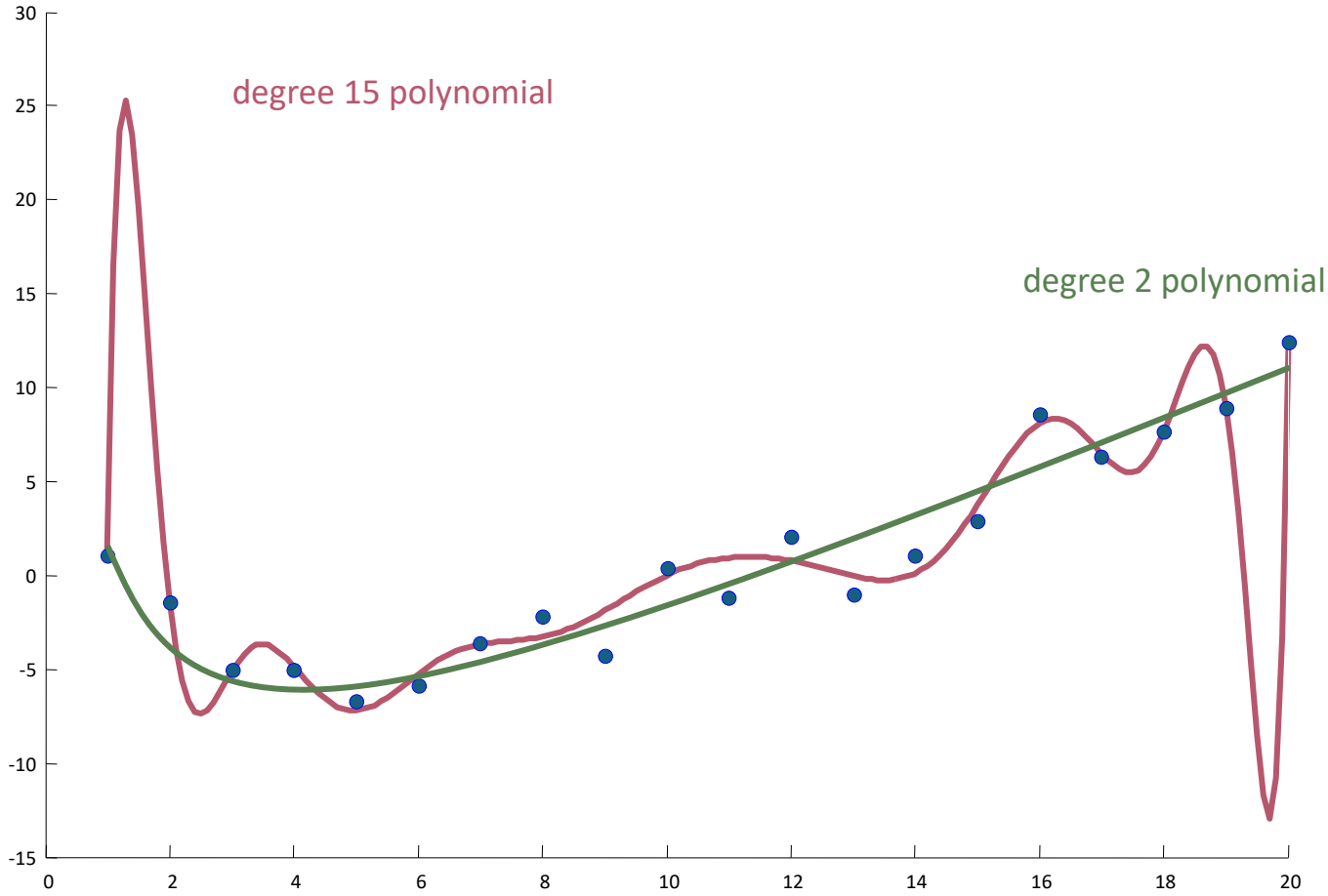
Goal: Find the curve (surface) that minimizes the loss function

- Model is polynomial, coefficients are still linear

$$h(x) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 = w_0 + w_1 x + w_2 z_2(x) + w_3 z_3(x)$$



What is the best "fit"?



Regularization

FOUNDATIONS

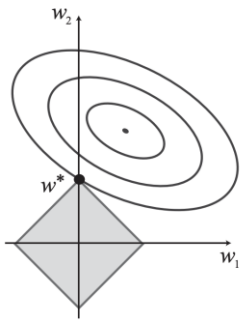
Regularization limits overfitting (variance) by constraining parameter values

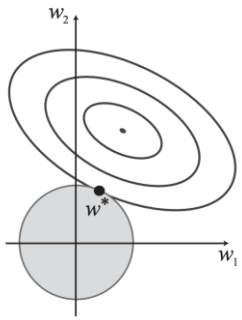
$$\mathcal{L}_{\text{regularized}} = \mathcal{L}_{\text{error}} + \lambda \cdot \text{Complexity}$$

LASSO

Complexity is $L_1(\boldsymbol{w}) = \sum_i |w_i|$

Encourages sparse weights (i.e. most weights will be 0)





Ridge

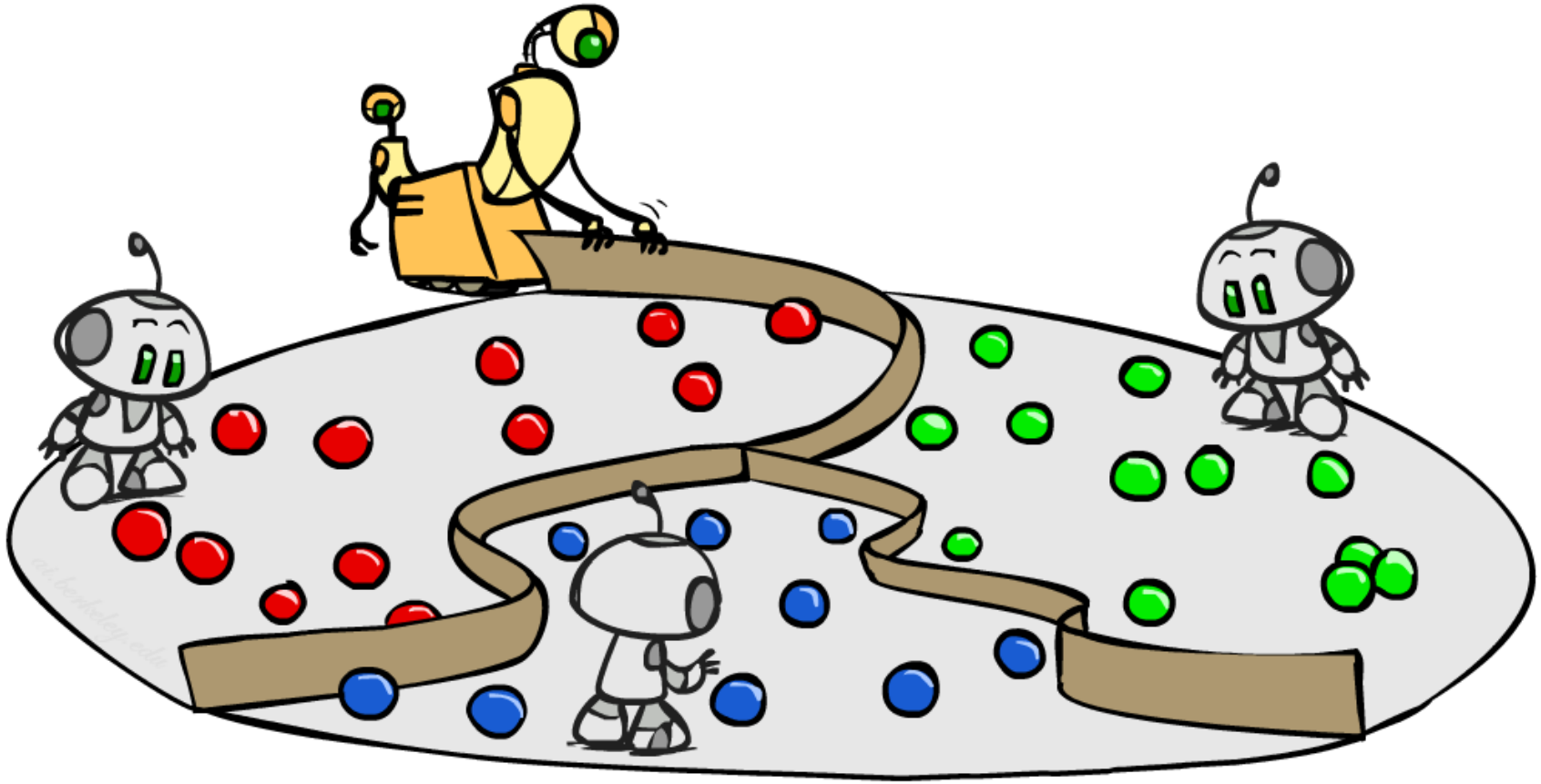
Complexity is $L_2(\boldsymbol{w}) = \sum_i (w_i)^2$

Encourages small weights

Questions?

From Continuous to Discrete

CONTEXT

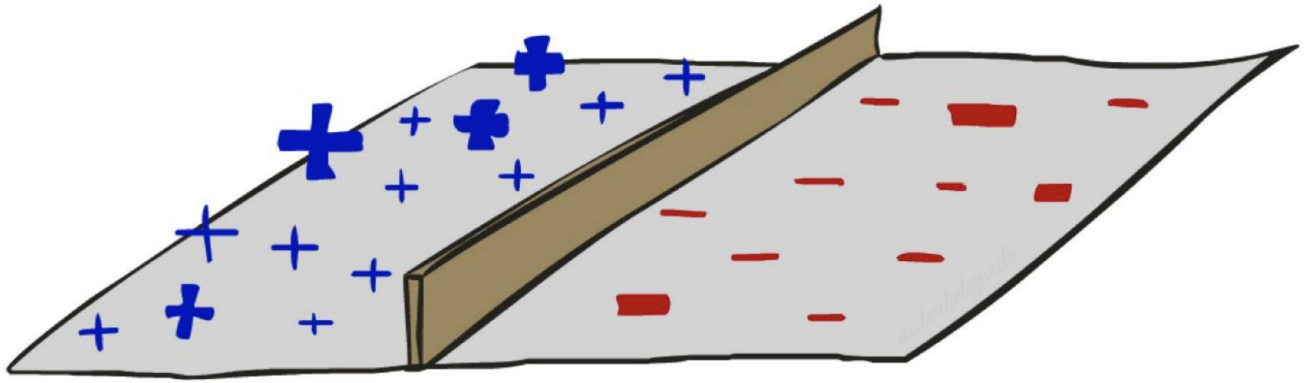


Decision Boundaries

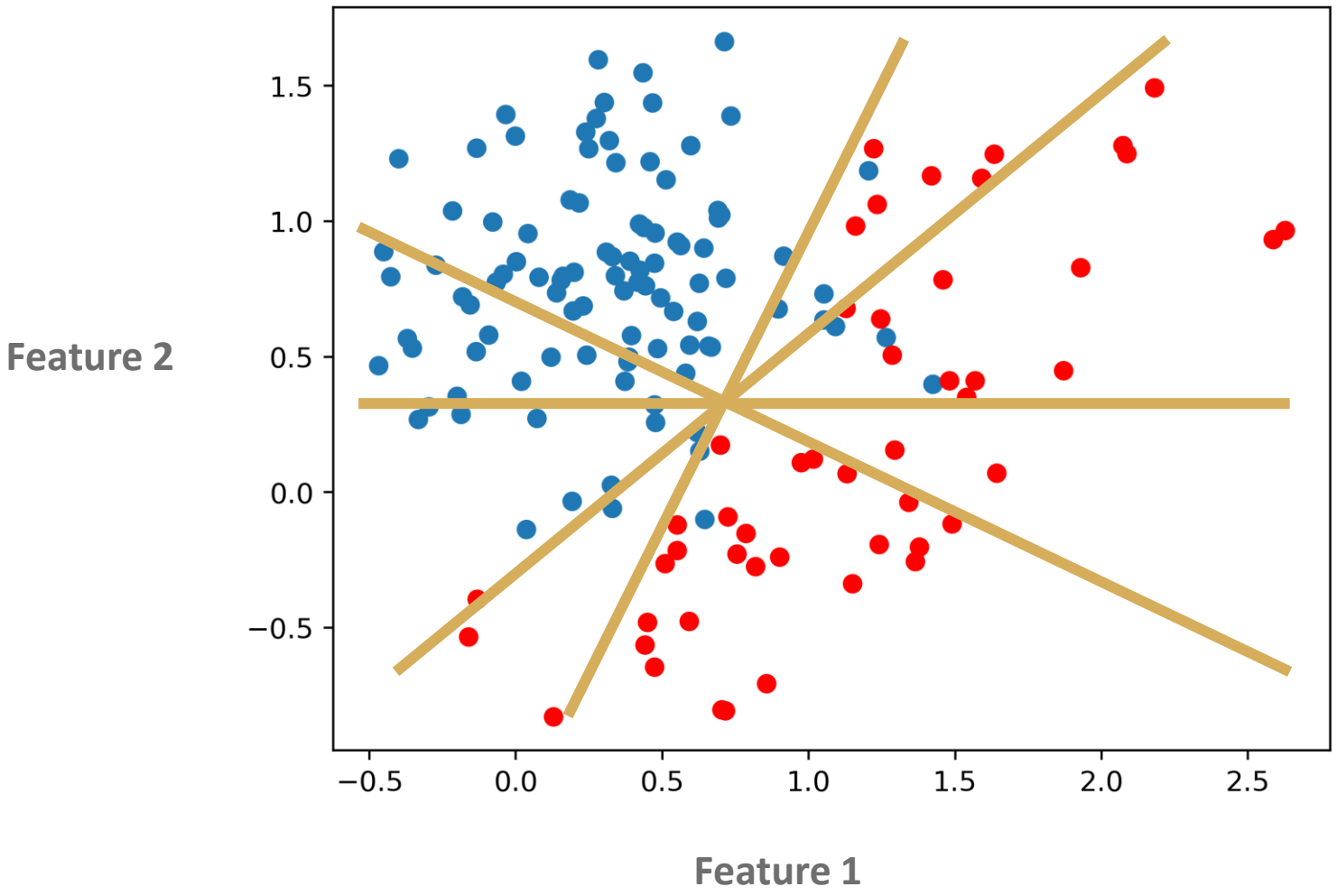
FOUNDATIONS

A **decision boundary** demarcates where in the feature space the classification changes

- Small movements to either side of the boundary changes the classification
- Not necessarily parametric



What is the best "fit"?



Binary Classification

FOUNDATIONS

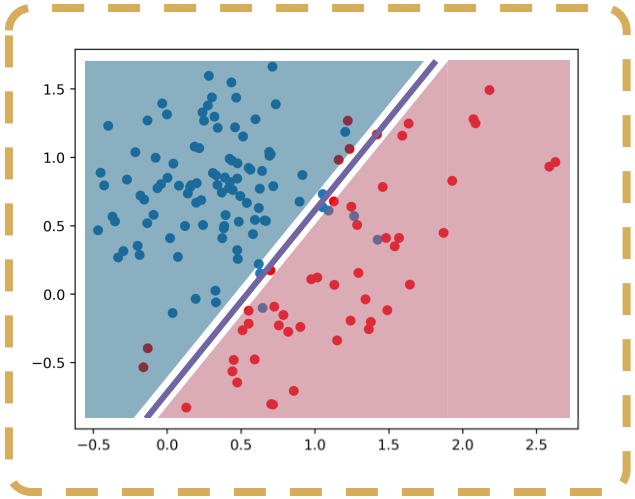
With a linear classification model, the line is the decision boundary

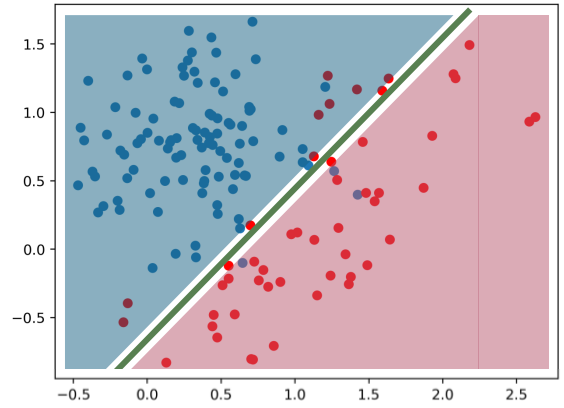
$$h(x) = w_0 + w_1 x_1 + w_2 x_2 \ldots = 0$$

- $h(x) \geq 0 \Longrightarrow$ positive class
- $h(x) < 0 \Longrightarrow$ negative class

Ideally, classes are linearly separable. If not:

- **Option 1**: Minimize misclassifications
- **Option 2**: Maximize distance from decision boundary



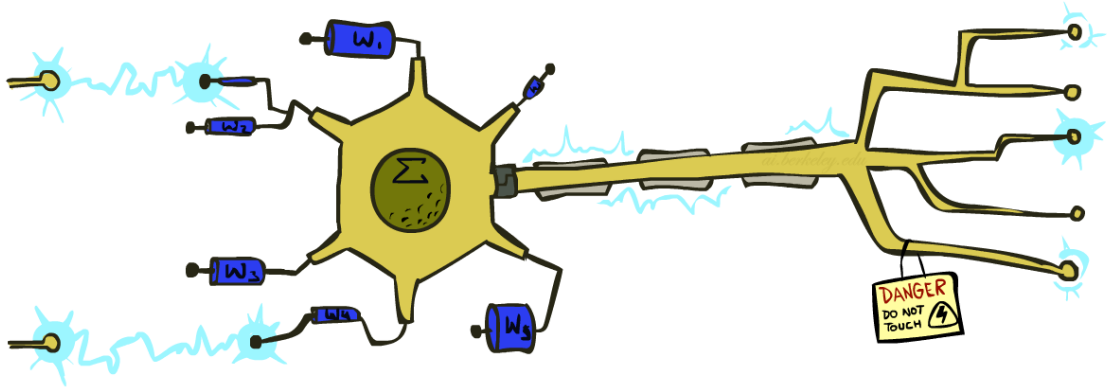


Perceptron

FOUNDATIONS

A **perceptron** is a linear classifier with output

$$h_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_i w_i x_i = \boldsymbol{w} \cdot \boldsymbol{x}$$



Perceptron Learning

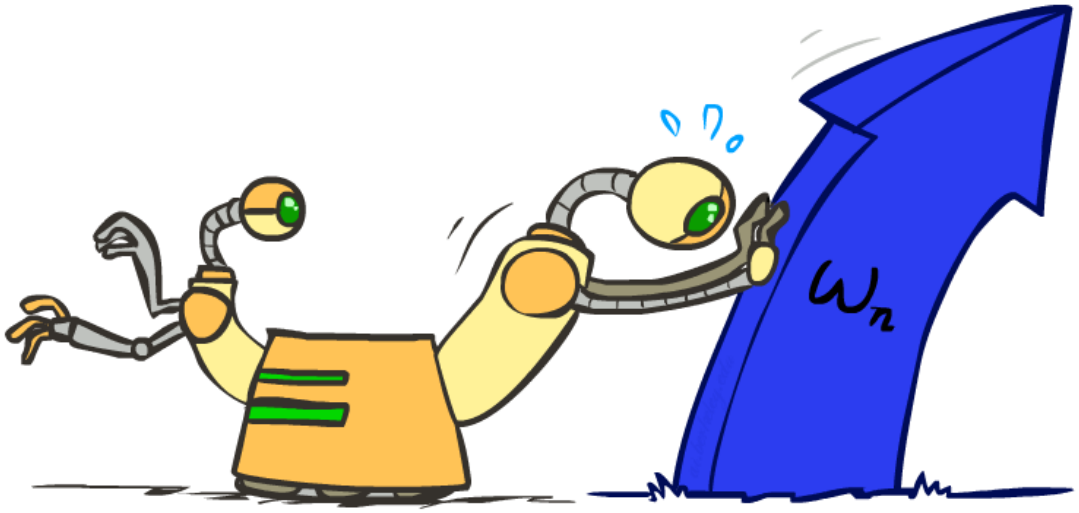
ALGORITHM

Repeat while not converged ($\exists i \;\; y_i \neq h_{\boldsymbol{w}}(\boldsymbol{x}_i)$):

- For each data point i do:
  - $w \leftarrow w - \eta(y_i - h_{\boldsymbol{w}}(\boldsymbol{x}_i))\boldsymbol{x}_i$

Gradient

Equivalent to least squares regression with $\mathcal{L} = \sum_i (y_i - h_{\boldsymbol{w}}(\boldsymbol{x}_i))^2$



Remember?

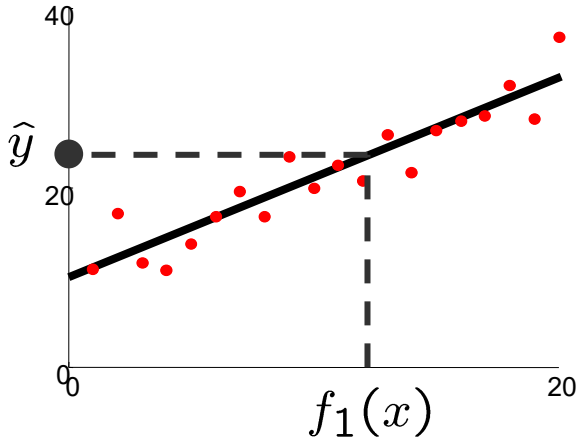
Update Q-values or Least-square weights

FOUNDATIONS

- **Approximate Q-Values:**

$$Q(s,a) = w_1 f_1(s,a) + w_2 f_2(s,a) + \cdots + w_n f_n(s,a)$$

$$w_i \leftarrow w_i + \alpha \cdot difference \cdot f_i(s,a)$$



$$\hat{y} = w_0 + w_1 f_1(x)$$

$$w_k \leftarrow w_k + \alpha \left(y - \sum_k w_k f_k(x) \right) f_k(x)$$

Or

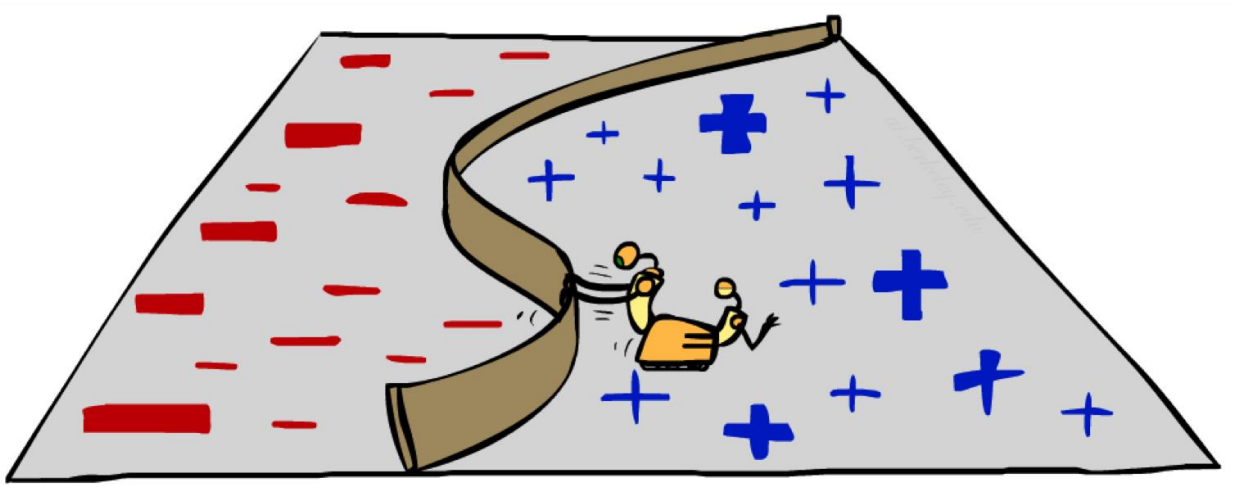
$$w_k \leftarrow w_k + \alpha([r + \gamma \cdot \max_{a'} Q(s',a')] - Q(s,a))f_k(x)$$

Convergence?

DEFINITION

Depends on linear separability

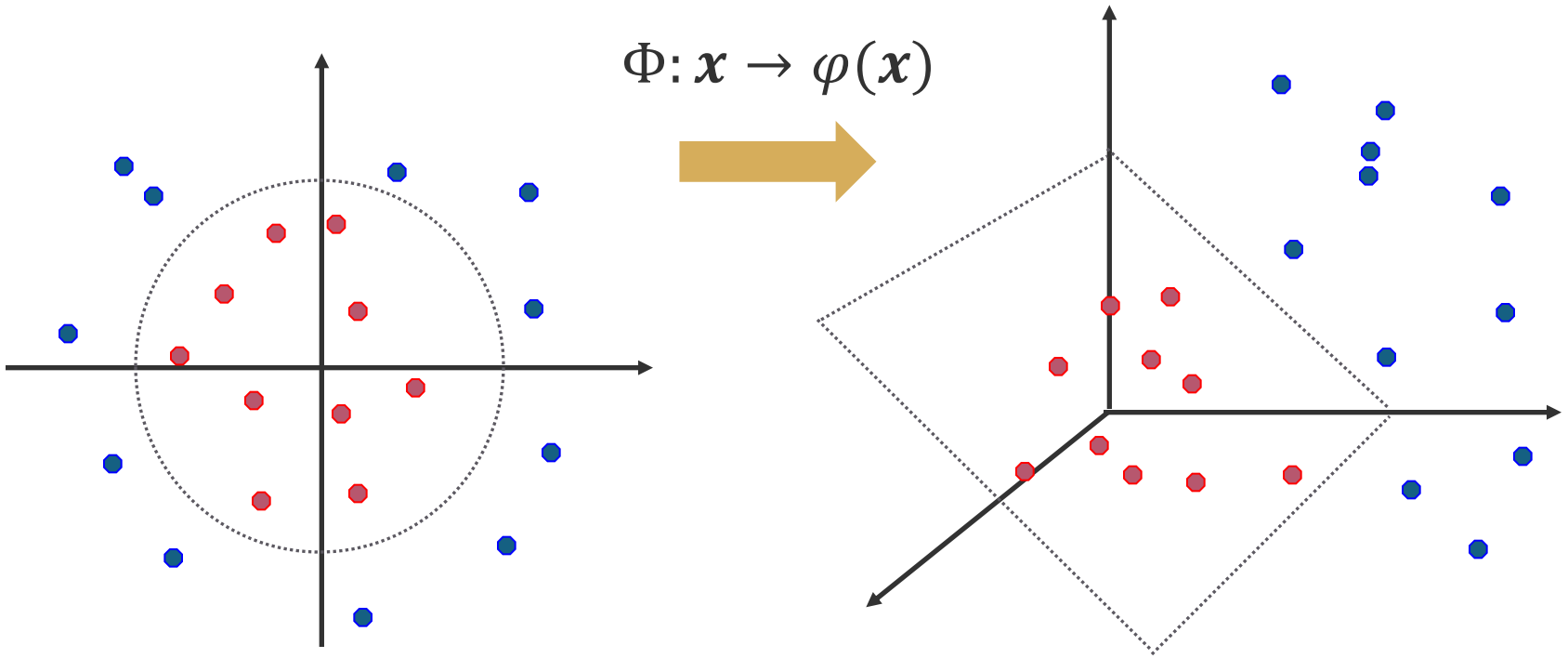
- If data are separable:
  - Perceptron learning will eventually converge to a separator with no misclassifications
- If data are not separable:
  - Perceptron learning will converge to a minimum-error solution (provided learning rate is decayed)



Non-Linear Separators

FOUNDATIONS

Goal: Map original feature space to higher-dimensional feature space where training set is separable



Kernels

DEFINITION

Kernels *implicitly* map vectors to higher-dimensional space, take the dot product, and hand back the result

- Linear Kernel: $K_L(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x} \cdot \boldsymbol{x}' = \sum_i x_i x_i'$
- Polynomial Kernel: $K_P(\boldsymbol{x}, \boldsymbol{x}') = (\boldsymbol{x} \cdot \boldsymbol{x}' + c)^d$
- Radial Basis Function Kernel: $K_{RBF}(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \parallel \boldsymbol{x} - \boldsymbol{x}' \parallel^2)$
- Neural Tangent Kernel: $K_{NT}(\boldsymbol{x}, \boldsymbol{x}') = \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}) \cdot \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(\boldsymbol{x}')$

Questions?

that's it for today!

SUMMARY

- Regression is an optimization problem with the goal of minimizing the loss function
- Regularization limits overfitting by controlling parameter sizes
- The same optimization process can be used to fit a decision boundary

UPCOMING

- Classification
- Neural Networks

REMINDERS

- Complete **Project 3**
- Start **Homework 4**