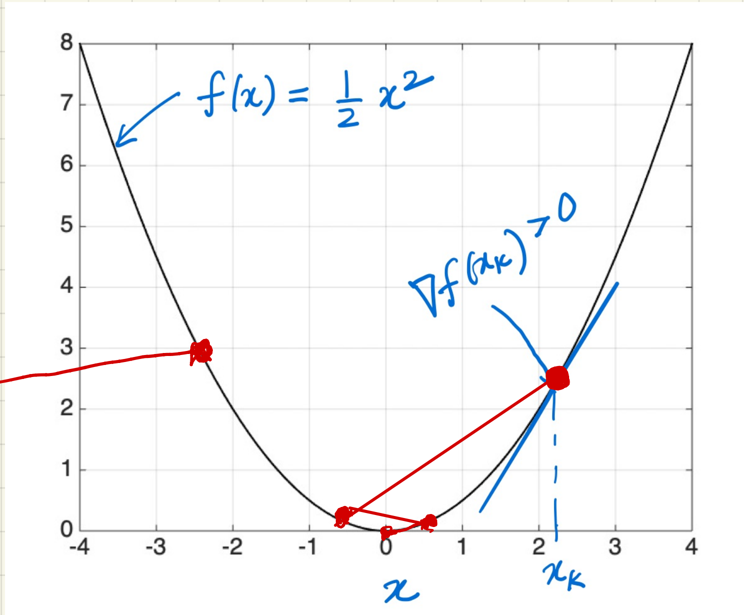
# Lecture 11, 10/1/24

ECE 364
Fall 2024

Ex.

$$f(x) = \frac{1}{2} x^2 \; ; \qquad x \in \mathbb{R}$$




$$\nabla f(x) = x$$

$$x_{k+1} = x_k - \alpha \nabla f(x_k)$$

$$= x_k (1 - \alpha)$$

Case 1: $\alpha = 1.5$, Then $x_{k+1} = x_k(-0.5)$

$$\Longrightarrow x_k = x_0 (-0.5)^k \longrightarrow 0 \text{ as } k \to \infty$$

Case 2: $\alpha = 2.5$, then $x_{k+1} = x_k(-1.5)$

$$\Longrightarrow x_k = x_0(-1.5)^k \Longrightarrow |x_k| \to \infty$$

Case 3: $\alpha = 2$; then $x_{k+1} = x_k(-1)$

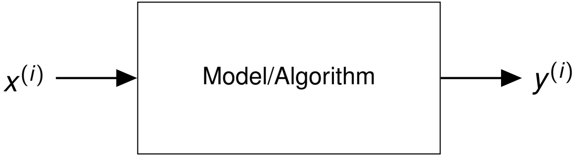
$\Longrightarrow x_k = x_0(-1)^k \Longrightarrow$ oscillation between $-x_0, x_0$

# Example:

Let’s get a computer to recognize whether there is a cat in the image.

Formulation of pattern recognition for first part of the course:

- input data/value/vector: $x^{(i)}$
- label/output: $y^{(i)}$




How do we call this process?

- Inference
- Prediction

What is learning?

$$x^{(i)} \longrightarrow \boxed{\text{Model/Algorithm}} \longrightarrow y^{(i)}$$

- Model/Algorithm depends on parameters $\boldsymbol{w}$
- Learning/Fitting of parameters $\boldsymbol{w}$
- Based on dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$

How could we recognize a cat? What should the algorithm do?

How is an image represented in a computer?

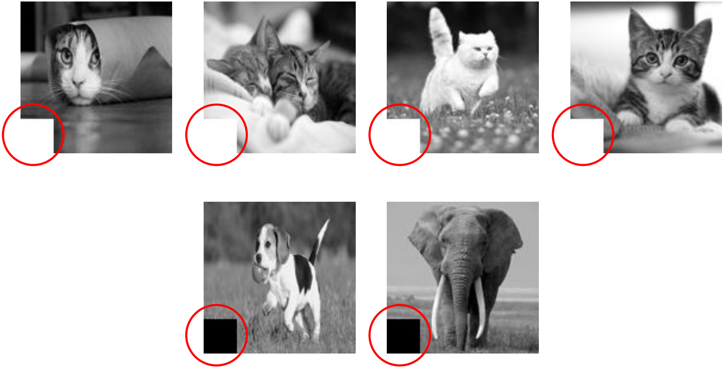
| | | | | | | |
|---|---|---|---|---|---|---|
| 69 | 69 | 70 | 70 | 71 | 72 | 72 |
| 79 | 80 | 80 | 81 | 82 | 83 | 83 |
| 87 | 88 | 88 | 89 | 90 | 91 | 91 |
| 95 | 96 | 96 | 97 | 98 | 99 | 99 |
| 108 | 108 | 109 | 110 | 110 | 111 | 112 |
| 119 | 119 | 120 | 120 | 121 | 122 | 122 |
| 130 | 131 | 131 | 132 | 133 | 134 | 134 |
| 144 | 144 | 145 | 145 | 146 | 147 | 148 |
| 158 | 158 | 159 | 160 | 162 | 163 | 164 |
| 172 | 172 | 173 | 174 | 175 | 176 | 177 |
| 187 | 188 | 189 | 190 | 191 | 192 | 193 |
| 198 | 198 | 199 | 200 | 202 | 203 | 204 |
| 207 | 207 | 208 | 209 | 210 | 212 | 212 |

Given that we know how a computer 'sees' an image, how can it recognize cats?

Still very hard to describe what a cat looks like.

Let's look at tons of examples (Dataset).

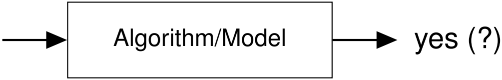
Dataset (Thousands of examples):



Instead of asking to recognize cats in general, let's recognize cats in this dataset

Algorithm: if bottom left corner is black (0) say no, otherwise say yes

Algorithm: if bottom left corner is black (0) say no, otherwise say yes




Works perfect on our dataset. :)
But does not generalize to other data. :(

**Conclusion:**

We designed a simple "classifier" that works on this dataset but doesn't work on real data.

**To our rescue:**

Machine learning found mechanisms to search for mappings which generalize.

**Scope of this class:**

In this class we talk about algorithms and models. A detailed treatment about generalization is left to lectures on learning theory.

Categorization of pattern recognition algorithms according to

- Available annotated data (supervised vs. unsupervised)
- Complexity of model (linear vs. non-linear)
- Structure of output (independent vs. structured)
- Modeling of data ($x^{(i)}$) or label ($y^{(i)}$) (generative vs. discriminative)

# Classification Framework: Formalism

**Main Components**

- an *input* (also called *observation* or *data point*), denoted by $x$
- a discrete *output* (also called *label* or *class*), denoted by $y$
- a *decision function* (also called *classification rule*), $y = f(x)$

The main problem in Pattern Recognition is to construct the *decision function*.

# Linear Regression

**Goals of this lecture**

- Math Intro
- Getting to know linear regression
- Understanding how linear regression works
- Examples for linear regression

**Reading Material**

- K. Murphy; Machine Learning: A Probabilistic Perspective; Chapter 7

## Math Intro:

- Vector: $\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$
- Matrix: $\mathbf{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots \\ x_{n,1} & \cdots & x_{n,m} \end{bmatrix} \in \mathbb{R}^{n \times m}$
- Norm: $\|x^{(1)} - x^{(2)}\|_2^2 = \sum_{i=1}^{n} (x_i^{(1)} - x_i^{(2)})^2$ distance between two points in $n$ dimensions
- Transpose: $\mathbf{X}^T = \begin{bmatrix} x_{1,1} & \cdots & x_{n,1} \\ \vdots & \ddots & \vdots \\ x_{1,m} & \cdots & x_{m,n} \end{bmatrix} \in \mathbb{R}^{m \times n}$

$$\boldsymbol{x}^T = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix} \in \mathbb{R}^{1 \times n}$$

- Matrix multiplication: $\mathbf{X}^T \boldsymbol{x}$ or $\mathbf{X}\boldsymbol{x}$?

Discrete Probability: $y \in \{1, \ldots, 6\}$

- Discrete probability distribution: $p(Y = y) \in [0, 1]$ with $\sum_{y \in \{1,\ldots,6\}} p(Y = y) = 1$
- Abbreviation: $p(Y = y) = p(y) \in [0, 1]$
- Expectation: $\mathbb{E}_{p(y)}[f(y)] = \sum_{y \in \{1,\ldots,6\}} p(y)f(y)$

Continuous probability: $y \in \mathbb{R}$

- $p(Y = 1) = 0$
- Probability density function: $p(y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{-1}{2\sigma^2}(y - \mu)^2\right)$
- Mean: $\mathbb{E}_{p(y)}[y] = \int_{-\infty}^{\infty} yp(y)dy = \mu$
- Variance: $\mathbb{E}_{p(y)}[(y - \mu)^2] = \sigma^2$

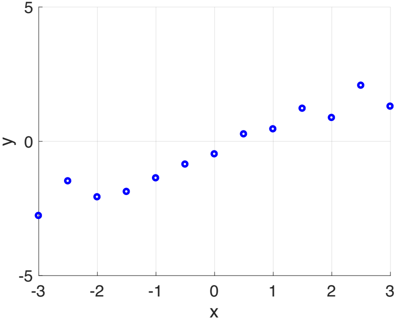
Multivariate continuous probability: $\boldsymbol{y} \in \mathbb{R}^n$ $\mu \in \mathbb{R}^n$

- n-dimensional density: $p(\boldsymbol{y}) = p(y_1, \ldots, y_n) = \frac{1}{\sqrt{(2\pi)^n|\Sigma|}} \exp\left(\frac{-1}{2}(\boldsymbol{y} - \mu)^T \Sigma^{-1} (\boldsymbol{y} - \mu)\right)$
- Covariance matrix: $\Sigma$

Multivariate calculus: $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{w} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n \times n}$

- Multivariate function: $f(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$
- Derivative: $\frac{\partial f}{\partial \boldsymbol{x}} = \boldsymbol{w}$ (e.g., Eq. (69))
- Multivariate function: $f(\boldsymbol{x}) = \boldsymbol{x}^T \mathbf{A} \boldsymbol{x}$
- Derivative: $\frac{\partial f}{\partial \boldsymbol{x}} = (\mathbf{A} + \mathbf{A}^T)\boldsymbol{x}$ (e.g., Eq. (78) or Eq. (81))

## Linear Regression - The Problem:




Given outcomes $y^{(i)} \in \mathbb{R}$ for covariates $x^{(i)} \in \mathbb{R}$,

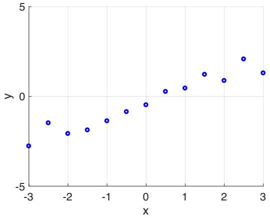
what is/are the underlying system/model/model parameters?

Let's assume a linear model with parameters $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$

$$y = w_1 \cdot x + w_2$$

Given a dataset of $N$ pairs $(x, y)$:

$$\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$$




How do we find the parameters $w_1$, $w_2$?

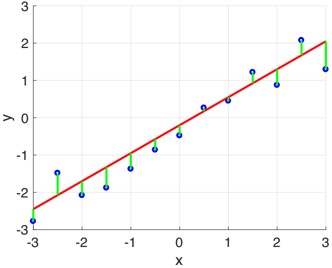
Assuming model

$$y = w_1 \cdot x + w_2$$

Find parameters $w_1$, $w_2$ such that the squared error is small

$$\arg\min_{w_1, w_2} \frac{1}{2} \sum_{i=1}^{N} \left(y^{(i)} - w_1 \cdot x^{(i)} - w_2\right)^2$$

What exactly is the error?



Program:

$$\arg\min_{w_1, w_2} \frac{1}{2} \sum_{i=1}^{N} \left( y^{(i)} - w_1 \cdot x^{(i)} - w_2 \right)^2$$

Vector notation:

$$\arg\min_{w_1, w_2} \frac{1}{2} \left\| \underbrace{\begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathbb{R}^N} - \underbrace{\begin{bmatrix} x^{(1)} & 1 \\ \vdots & \vdots \\ x^{(N)} & 1 \end{bmatrix}}_{\mathbf{X}^\top \in \mathbb{R}^{N \times 2}} \cdot \underbrace{\begin{bmatrix} w_1 \\ w_2 \end{bmatrix}}_{\mathbf{w} \in \mathbb{R}^2} \right\|_2^2$$

Program:

$$\arg\min_{\boldsymbol{w}} \underbrace{\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}^\top \boldsymbol{w}\|_2^2}_{\text{cost/loss function}}$$

$(Y - X^T w)^T (Y - X^T w)$

$= Y^T Y - Y^T X^T w - w^T X Y + w^T X X^T w$

How to solve the program:

- Take derivative w.r.t. $\boldsymbol{w}$ of cost function
- Set derivative w.r.t. $\boldsymbol{w}$ to zero
- Solve for $\boldsymbol{w}$

$\frac{\partial}{\partial w} = 0 - XY - XY + 2XX^T w = 0$

$\Rightarrow XX^T w = XY$

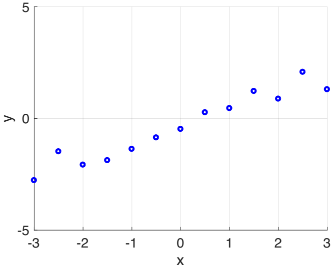
Derivative:

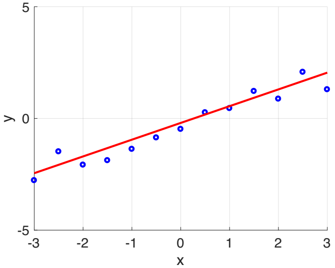
$$\boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{w}^* - \boldsymbol{X}\boldsymbol{Y} = 0$$

Solution:

$$\boldsymbol{w}^* = \left(\boldsymbol{X}\boldsymbol{X}^\top\right)^{-1} \boldsymbol{X}\boldsymbol{Y}$$

Linear regression:



Extensions:

- Higher dimensional problems ($\boldsymbol{x}^{(i)} \in \mathbb{R}^d$)
- Regularization
- Higher order polynomials

Higher dimensional problems ($\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, $y^{(i)} \in \mathbb{R}$)
Model:

$$y^{(i)} = w_0 + \sum_{k=1}^{d} \boldsymbol{x}_k^{(i)} w_k$$

Program:

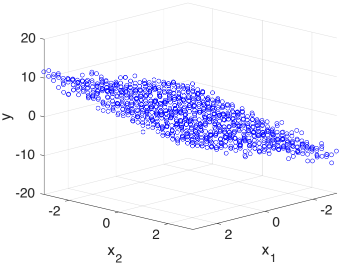
$$\arg\min_{\boldsymbol{w}} \frac{1}{2} \| \underbrace{\boldsymbol{Y}}_{\in \mathbb{R}^N} - \underbrace{\boldsymbol{X}^\top}_{\in \mathbb{R}^{N \times (d+1)}} \underbrace{\boldsymbol{w}}_{\in \mathbb{R}^{d+1}} \|_2^2$$

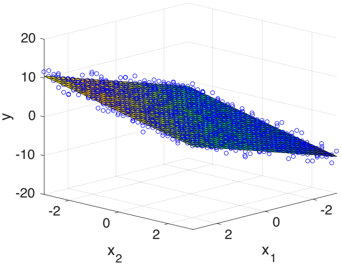
Solution: (obviously the same as before)

$$\boldsymbol{w}^* = \left( \boldsymbol{X} \boldsymbol{X}^\top \right)^{-1} \boldsymbol{X} \boldsymbol{Y}$$

Example:

$$\boldsymbol{w}^* = \left(\boldsymbol{X}\boldsymbol{X}^\top\right)^{-1} \boldsymbol{X}\boldsymbol{Y}$$




What if $N < d + 1$?

Regularization:

we want to make sure that the parameters are not too large

we want to make sure we can invert the matrix

Program:

$$\arg\min_{\boldsymbol{w}} \underbrace{\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}^\top \boldsymbol{w}\|_2^2 + \frac{C}{2}\|\boldsymbol{w}\|_2^2}_{\text{cost function}}$$

Solution:

$$\boldsymbol{w}^* = \left(\boldsymbol{X}\boldsymbol{X}^\top + C\boldsymbol{I}\right)^{-1}\boldsymbol{X}\boldsymbol{Y}$$

Higher order polynomials ($x^{(i)} \in \mathbb{R}$, $y^{(i)} \in \mathbb{R}$)
Model:

$$y^{(i)} = w_2 \cdot \left(x^{(i)}\right)^2 + w_1 \cdot x^{(i)} + w_0$$

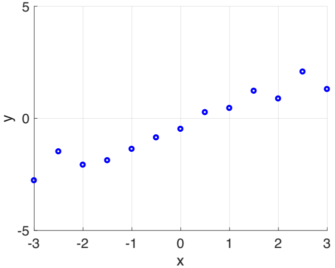
Program:

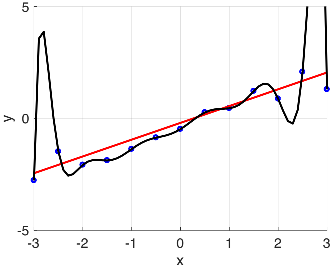
$$\arg\min_{w_0,w_1,w_2} \frac{1}{2} \left\| \underbrace{\begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(N)} \end{bmatrix}}_{\mathbf{Y} \in \mathbb{R}^N} - \underbrace{\begin{bmatrix} \left(x^{(1)}\right)^2 & x^{(1)} & 1 \\ \vdots & \vdots & \vdots \\ \left(x^{(N)}\right)^2 & x^{(N)} & 1 \end{bmatrix}}_{\Phi^\top \in \mathbb{R}^{N\times M}} \cdot \underbrace{\begin{bmatrix} w_2 \\ w_1 \\ w_0 \end{bmatrix}}_{\mathbf{w} \in \mathbb{R}^M} \right\|_2^2$$

Solution:

$$\boldsymbol{w}^* = \left(\Phi\Phi^\top\right)^{-1} \Phi \boldsymbol{Y}$$

Example:




Which model is more reasonable?

Generalizing all aforementioned cases:

- $x^{(i)}$ is some data (e.g., images)
- $\phi(x^{(i)}) \in \mathbb{R}^M$ is a transformation into a feature vector

Model:

$$y^{(i)} = \phi(x^{(i)})^\top \boldsymbol{w}$$

Program:

$$\arg\min_{\boldsymbol{w}} \frac{1}{2} \sum_{i=1}^{N} \left( y^{(i)} - \phi(x^{(i)})^\top \boldsymbol{w} \right)^2$$

Solution:

$$\boldsymbol{w}^* = \left( \Phi \Phi^\top \right)^{-1} \Phi \boldsymbol{Y} \quad \text{where} \quad \Phi = \left[ \phi(x^{(1)}), \cdots, \phi(x^{(N)}) \right] \in \mathbb{R}^{M \times N}$$