

# Machine Learning Basics

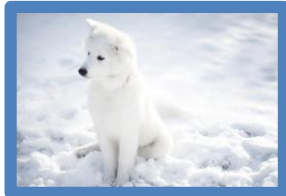
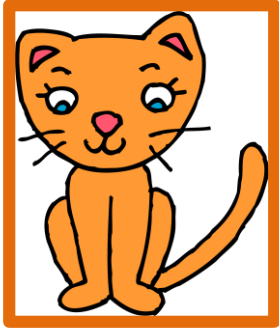
# Machine Learning



Task



# Image Classification





Cute



And Kittens



Clipart



Drawing



Cute Baby



White Cats And Kittens



Pose

Illumination

Appearance



# Image Classification

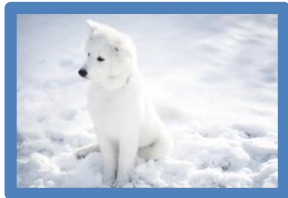


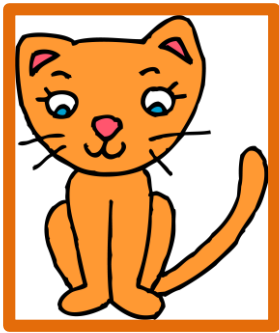
Occlusions



# Image Classification

Background clutter





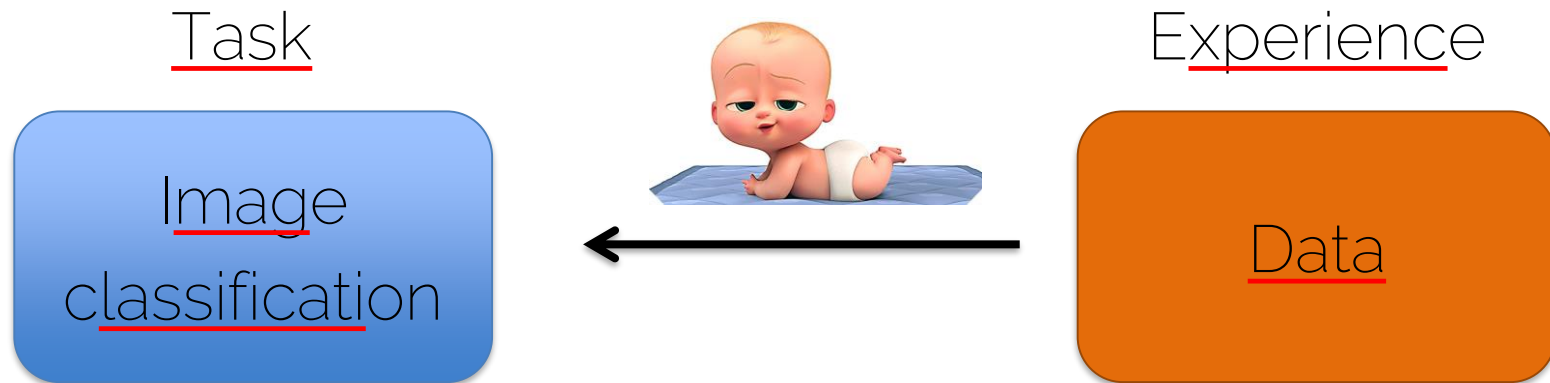
# Image Classification

Representation



# Machine Learning

- How can we learn to perform image classification?





# Machine Learning → RL

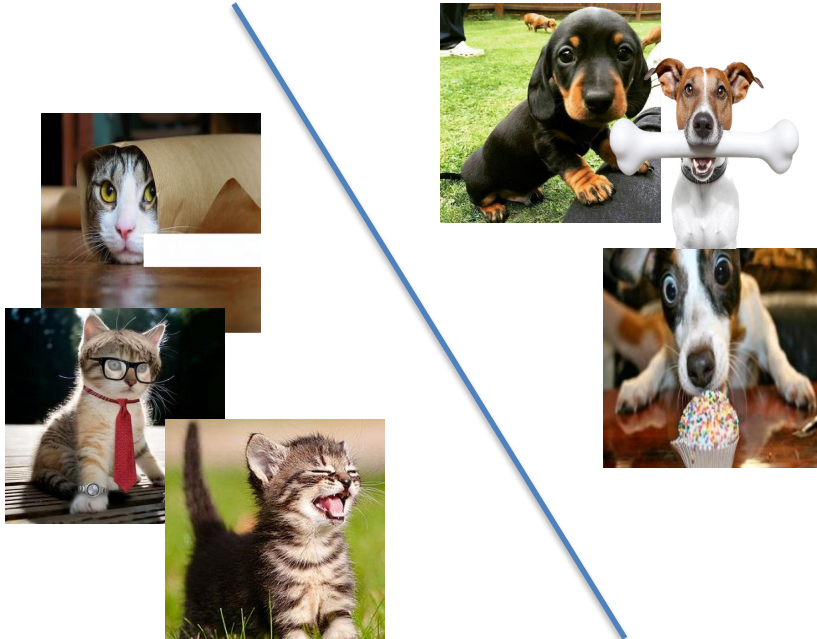
## Unsupervised learning

- No label or target class
- Find out properties of the structure of the data
- Clustering (k-means, PCA, etc.)

## Supervised learning

# Machine Learning

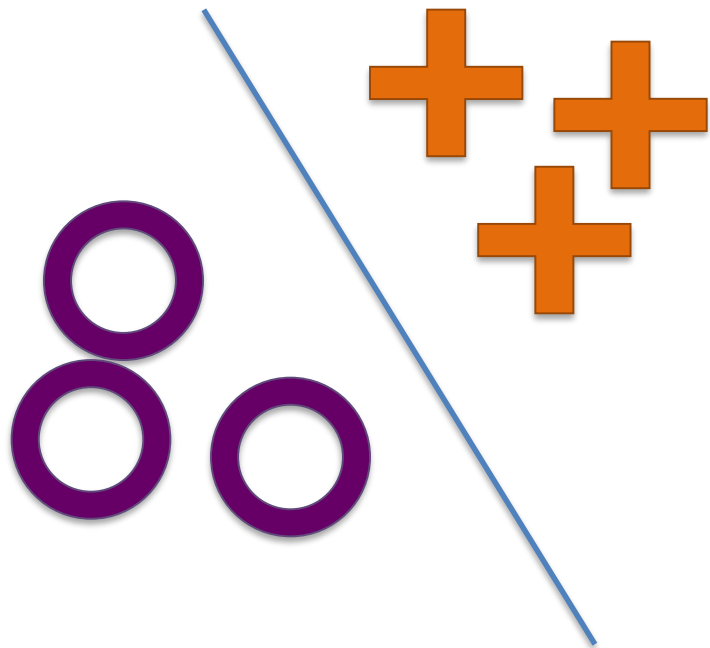
Unsupervised learning



Supervised learning

# Machine Learning

Unsupervised learning



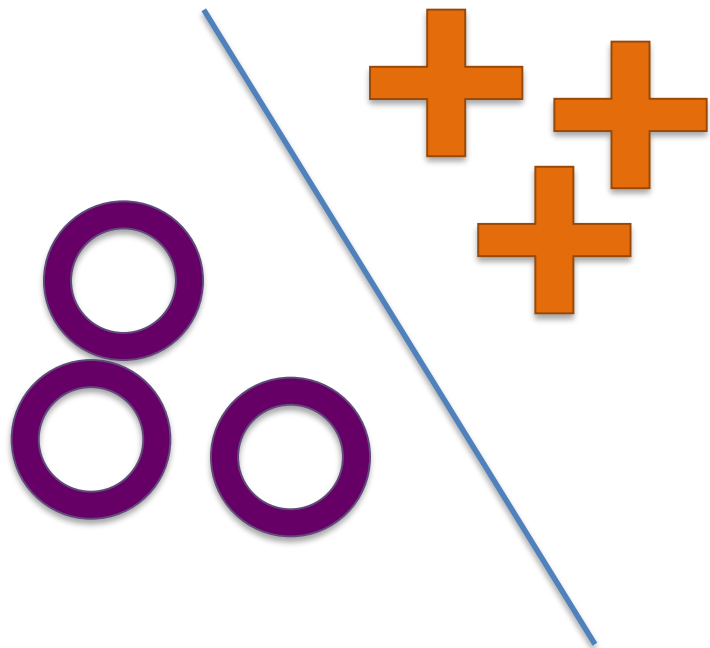
Supervised learning

*Already given*

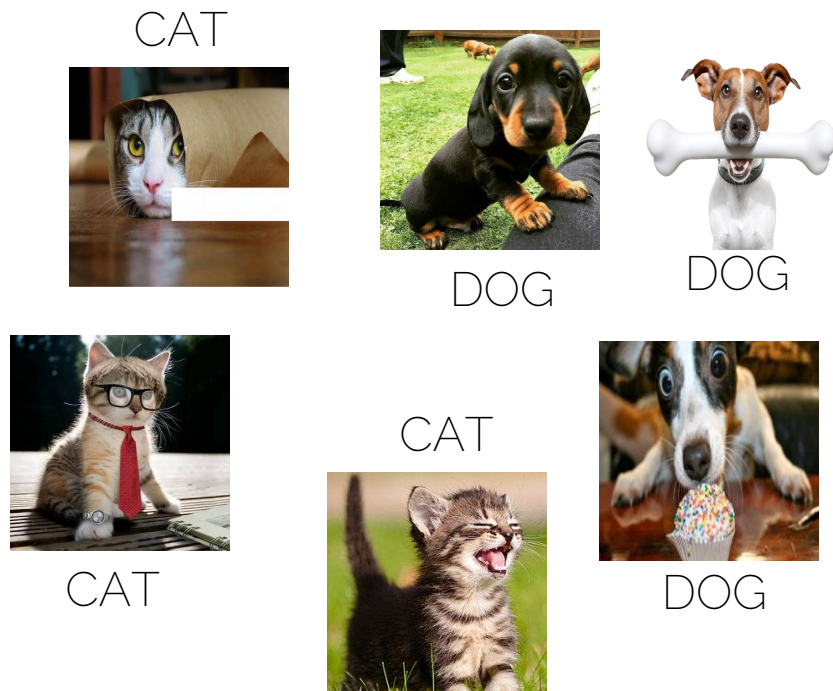
- Labels or target classes

# Machine Learning

## Unsupervised learning



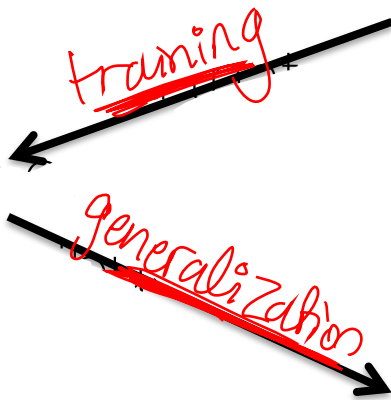
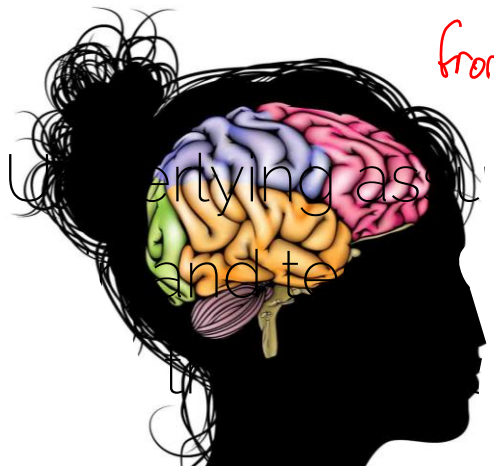
## Supervised learning



# Machine Learning

- How can we learn to perform image classification?

*Underlying assumption that  
train & test data come  
from the same distribution*



Experience

Test data

# Machine Learning

- How can we learn to perform image classification?

Task

Image  
classification

Performance  
measure

Accuracy

Experience

Data



# Machine Learning

Unsupervised learning



Supervised learning



Reinforcement learning

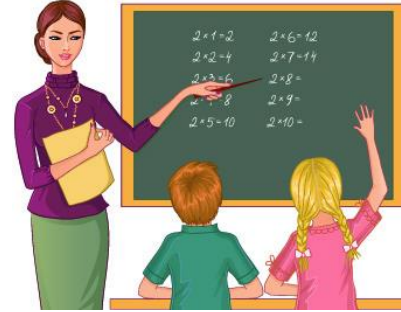


# Machine Learning

Unsupervised learning



Supervised learning



Reinforcement learning

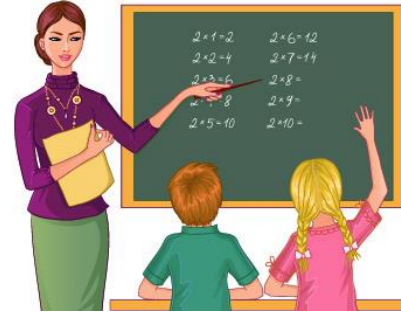


# Machine Learning

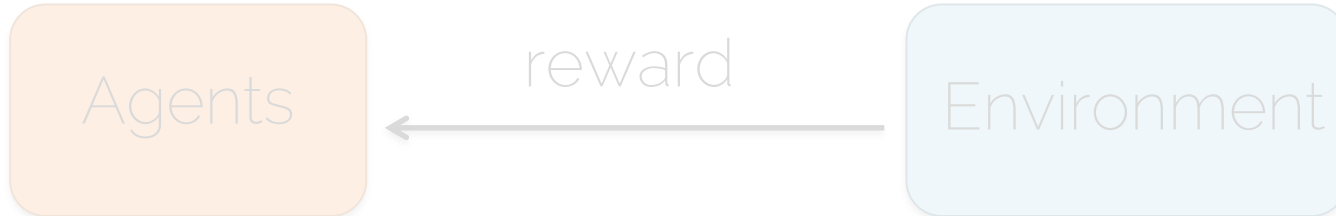
Unsupervised learning



Supervised learning

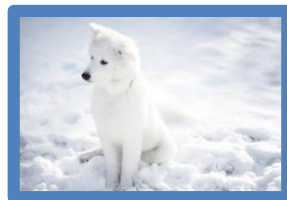
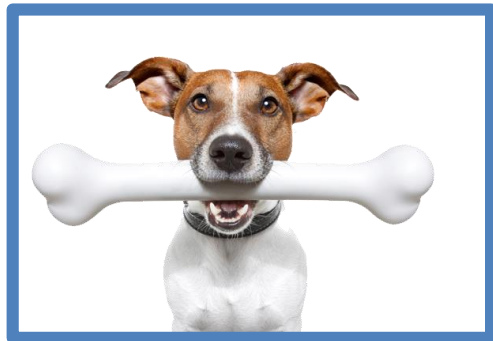
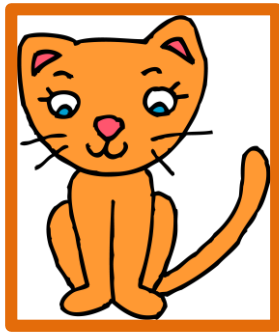


Reinforcement learning



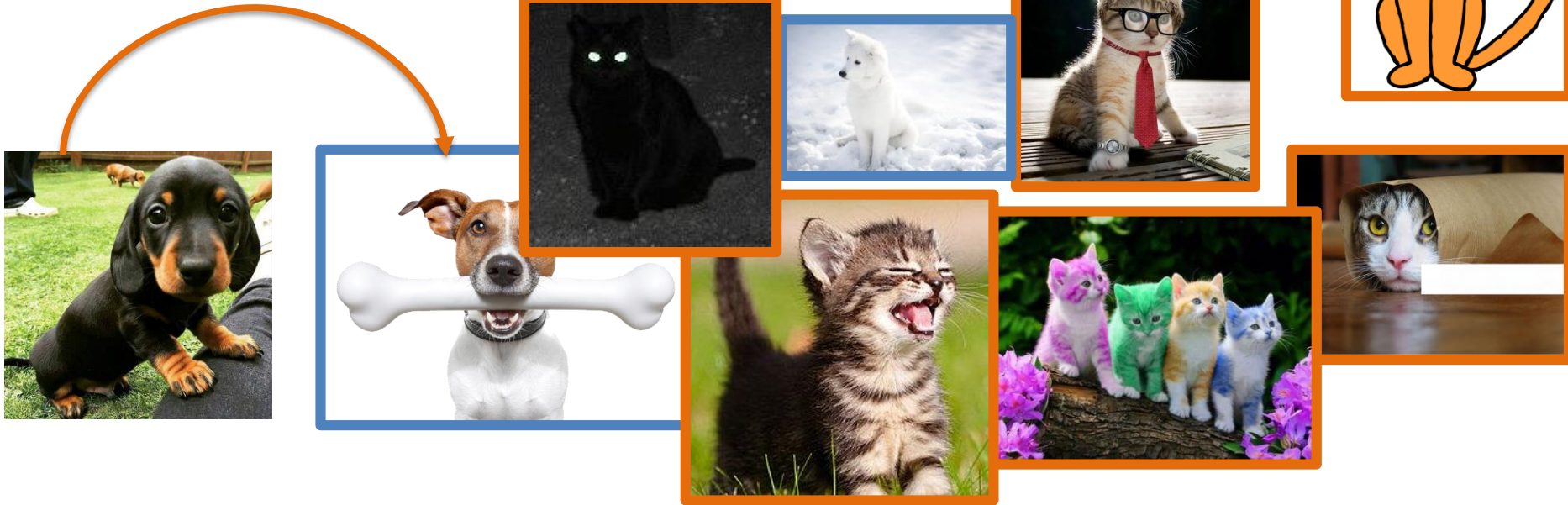
# A Simple Classifier

# Nearest Neighbor



# Nearest Neighbor

1 NN classifier = dog



distance



# Nearest Neighbor

3 NN

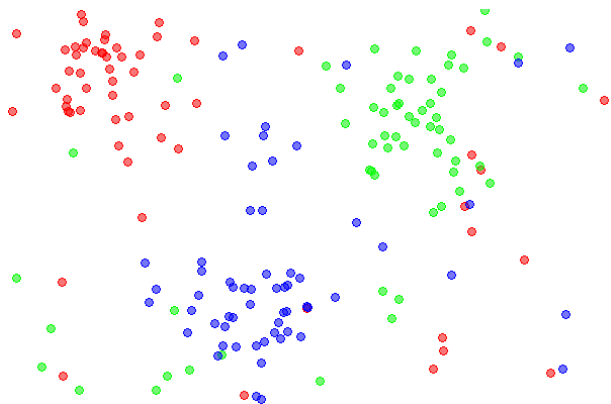


k-NN classifier = cat

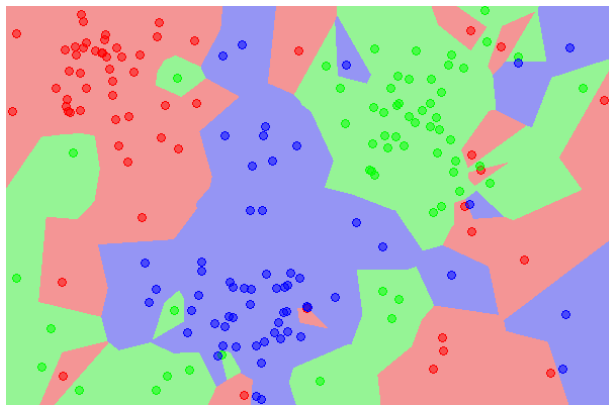
distance

# Nearest Neighbor

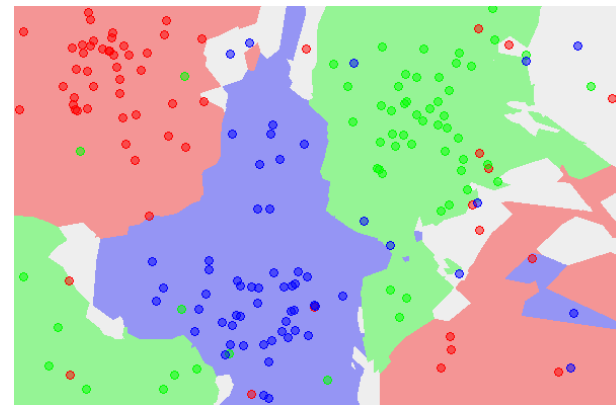
The Data



NN Classifier



5NN Classifier



*reg. effect*

How does the NN classifier perform on training data?

What classifier is more likely to perform best on test data?

Source: <https://commons.wikimedia.org/wiki/File:Data3classes.png>

# Nearest Neighbor

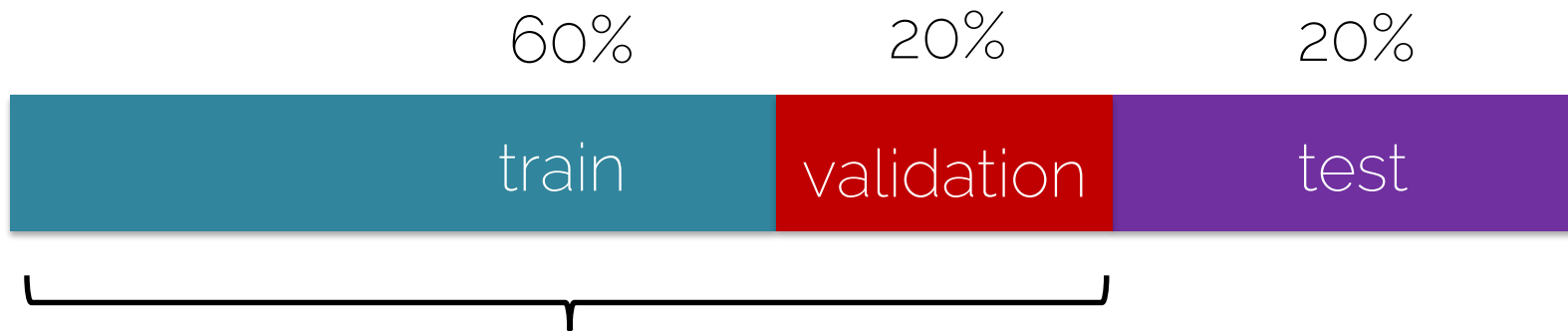
- Hyperparameters
  - L1 distance :  $|x - c|$
  - L2 distance :  $||x - c||_2$
  - No. of Neighbors:  $k$

\* These parameters are problem dependent.

- How do we choose these hyperparameters?

# Basic Recipe for Machine Learning

- Split your data

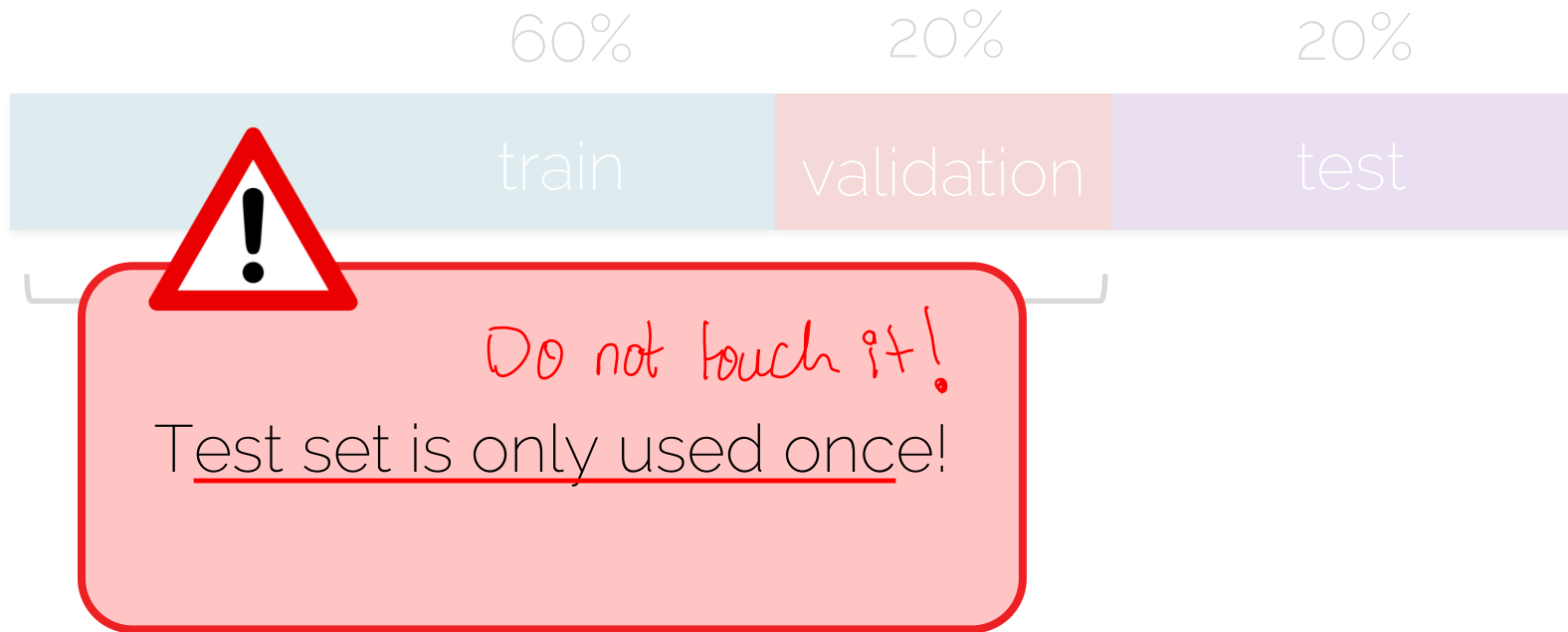


Find your hyperparameters

Other splits are also possible (e.g., 80%/10%/10%)

# Basic Recipe for Machine Learning

- Split your data



# Cross Validation

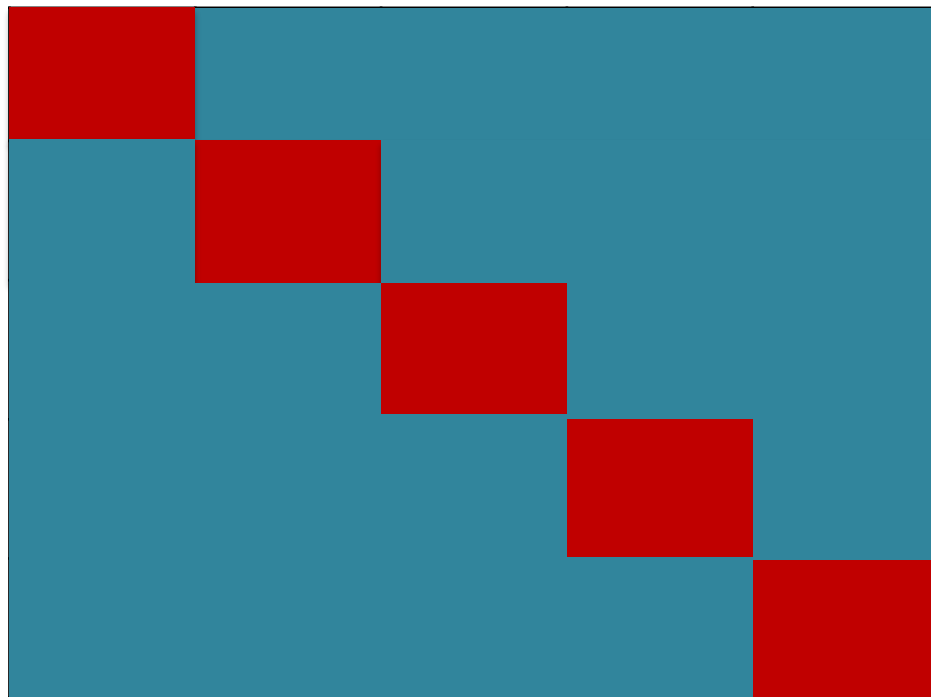
Run 1

Run 2

Run 3

Run 4

Run 5



train

validation



Split the training data into N folds



# Cross Validation



Find your hyperparameters

Why do cross  
validation?  
Why not just train  
and test?

# Cross Validation

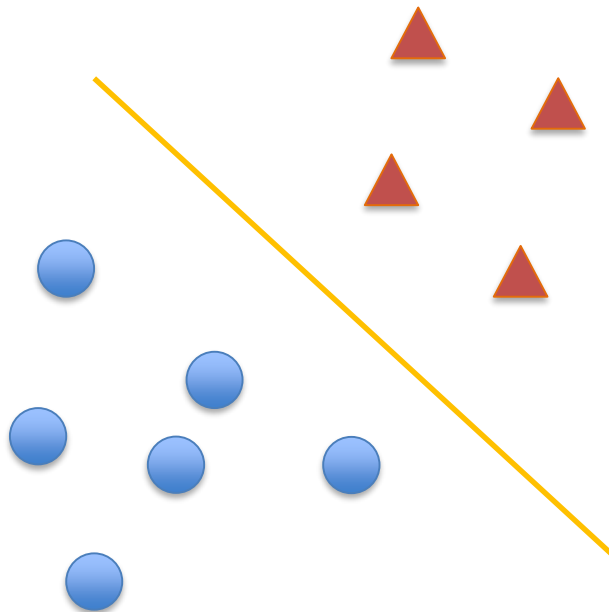


Test set is only used once!

Why do cross validation? Why not just train and test?

# Linear Decision Boundaries

This lecture

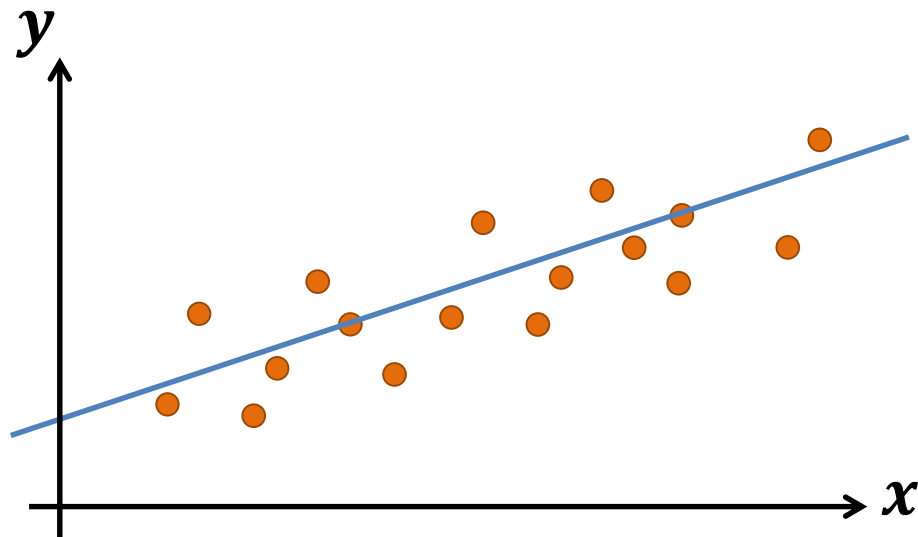


What are the pros and cons for using linear decision boundaries?

# Linear Regression

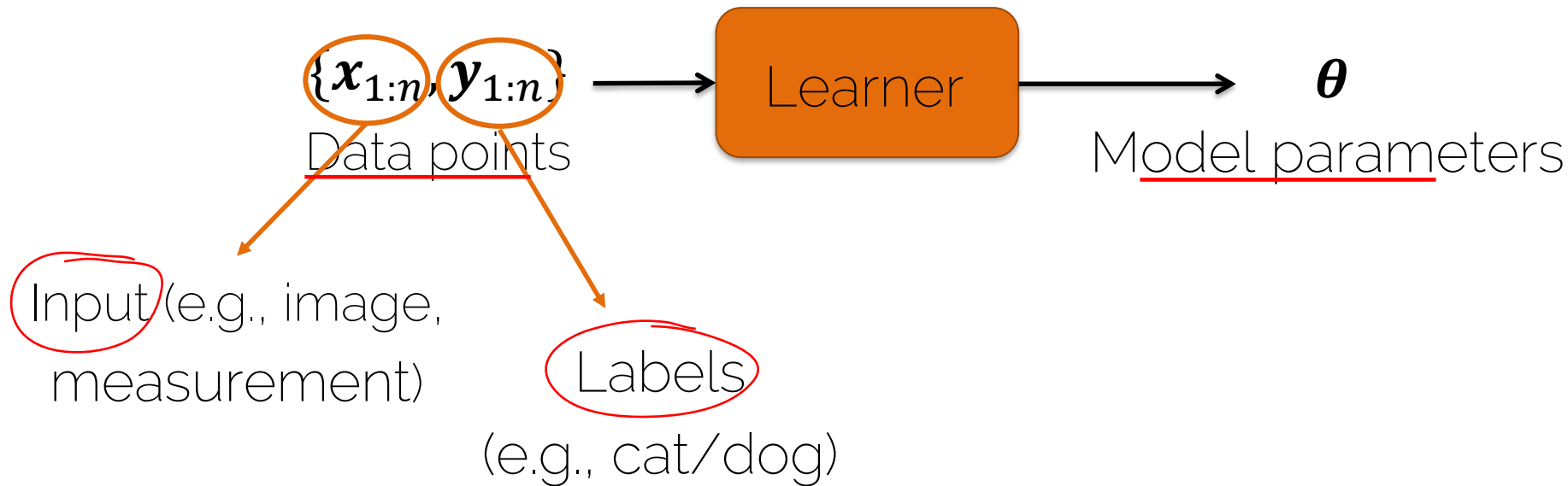
# Linear Regression

- Supervised learning
- Find a linear model that explains a target  $y$  given inputs  $x$



# Linear Regression

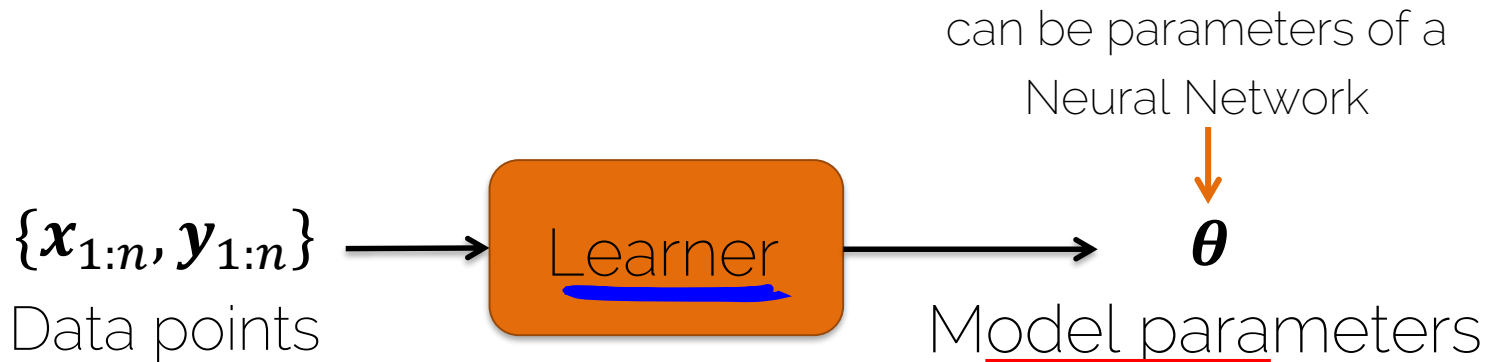
Training





# Linear Regression

Training



Testing



# Linear Prediction

- A linear model is expressed in the form

The diagram shows the equation  $\hat{y}_i = \sum_{j=1}^d x_{ij} \theta_j$  with several annotations:

- A red circle around  $\hat{y}_i$  with three red arrows pointing towards it from the left.
- A purple arrow pointing from the text "input dimension" to the superscript  $d$ .
- An orange circle around  $x_{ij}$  with an orange arrow pointing from the text "Input data, features" below to it.
- A blue circle around  $\theta_j$  with a blue arrow pointing from the text "weights (i.e., model parameters)" below to it.

Input data, features

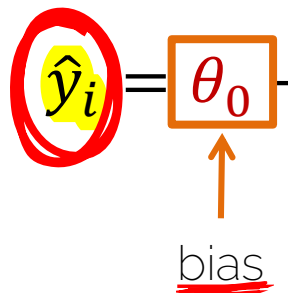
weights (i.e., model parameters)

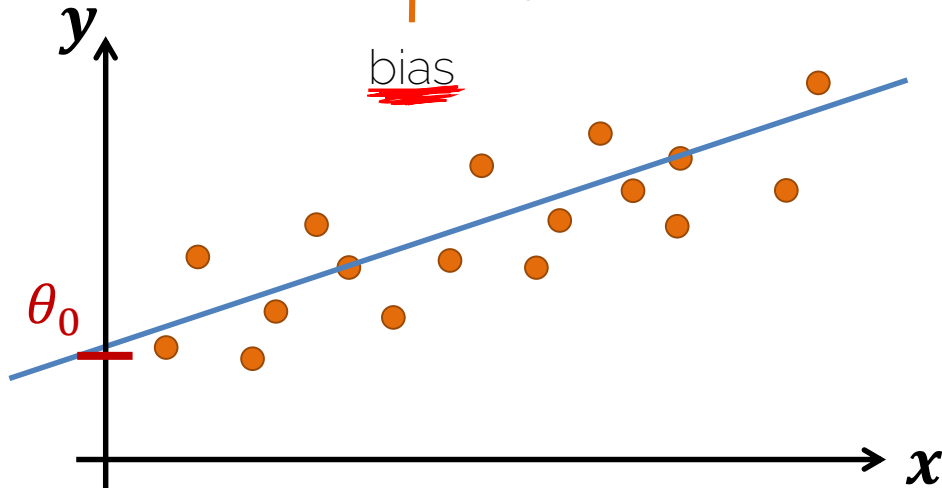
input dimension

# Linear Prediction

- A linear model is expressed in the form

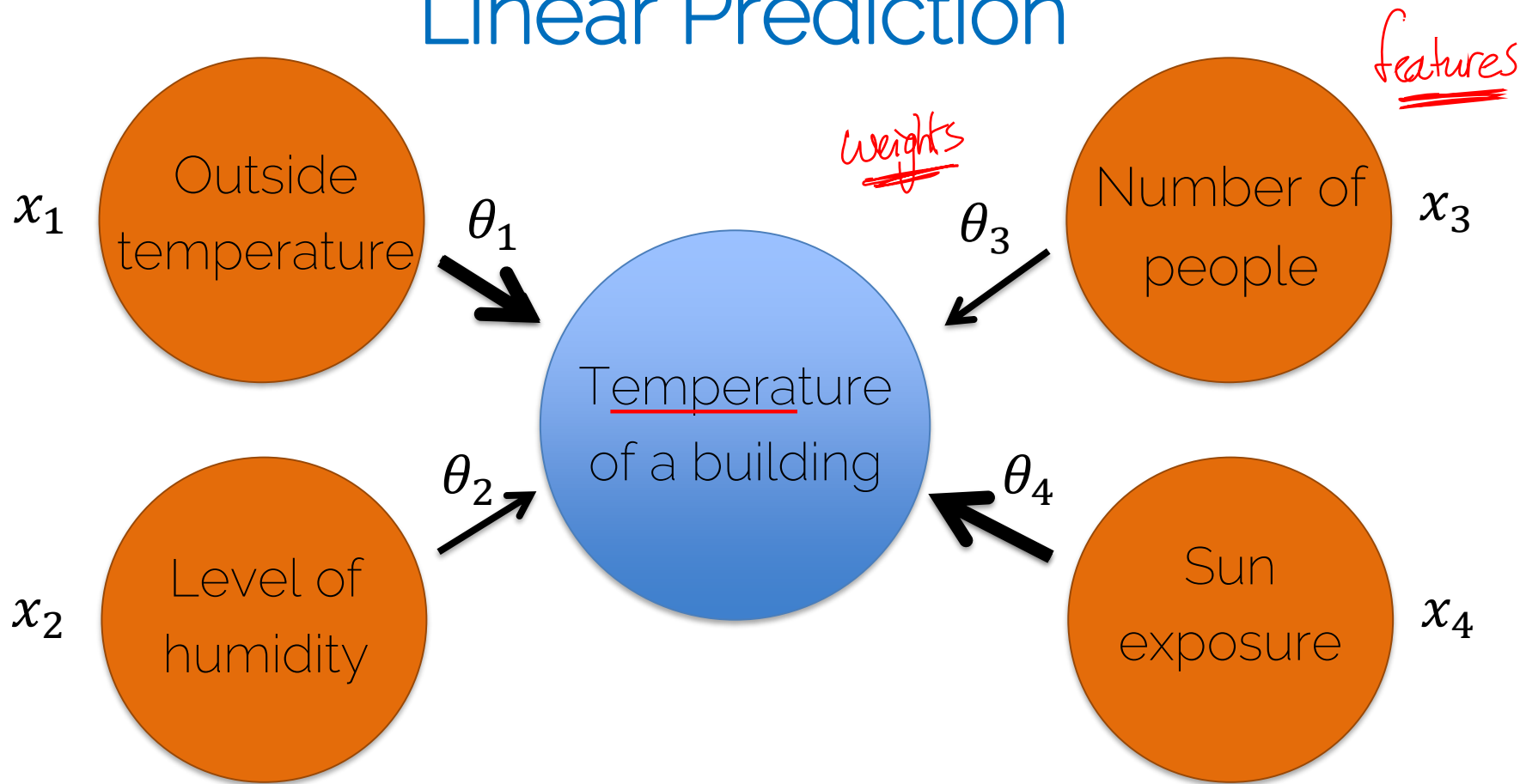
$$\hat{y}_i = \theta_0 + \sum_{j=1}^d x_{ij}\theta_j = \theta_0 + x_{i1}\theta_1 + x_{i2}\theta_2 + \dots + x_{id}\theta_d$$





NB Bias is independent  
of input features

# Linear Prediction



# Linear Prediction

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \theta_0 + \begin{bmatrix} x_{11} & \cdots & x_{1d} \\ x_{21} & \cdots & x_{2d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_d \end{bmatrix}$$

*bias*

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}$$

*Samples*      *features*

$$\Rightarrow \hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta}$$

*Matrix Notation*

# Linear Prediction

$$\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\theta}$$

Input features

(one sample has  $d$  features)

Prediction



$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1d} \\ 1 & x_{21} & \cdots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix}$$

Model

parameters

( $d$  weights and  $1$  bias)

# Linear Prediction

Temperature  
of the building

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \end{bmatrix} = \begin{bmatrix} 1 & 25 & 50 & 2 & 50 \\ 1 & -10 & 50 & 0 & 10 \end{bmatrix} \cdot \begin{bmatrix} 0.2 \\ 0.64 \\ 0 \\ 1 \\ 0.14 \end{bmatrix}$$

Bias      Outside temperature      Humidity      Number people      Sun exposure (%)      MODEL

# Linear Prediction



Temperature  
of the building

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \end{bmatrix}$$

=

$$\begin{bmatrix} 1 & 25 & 50 & 2 & 50 \\ 1 & -10 & 50 & 0 & 10 \end{bmatrix}$$

Bias

Outside temperature

Humidity

Number people

Sun exposure (%)

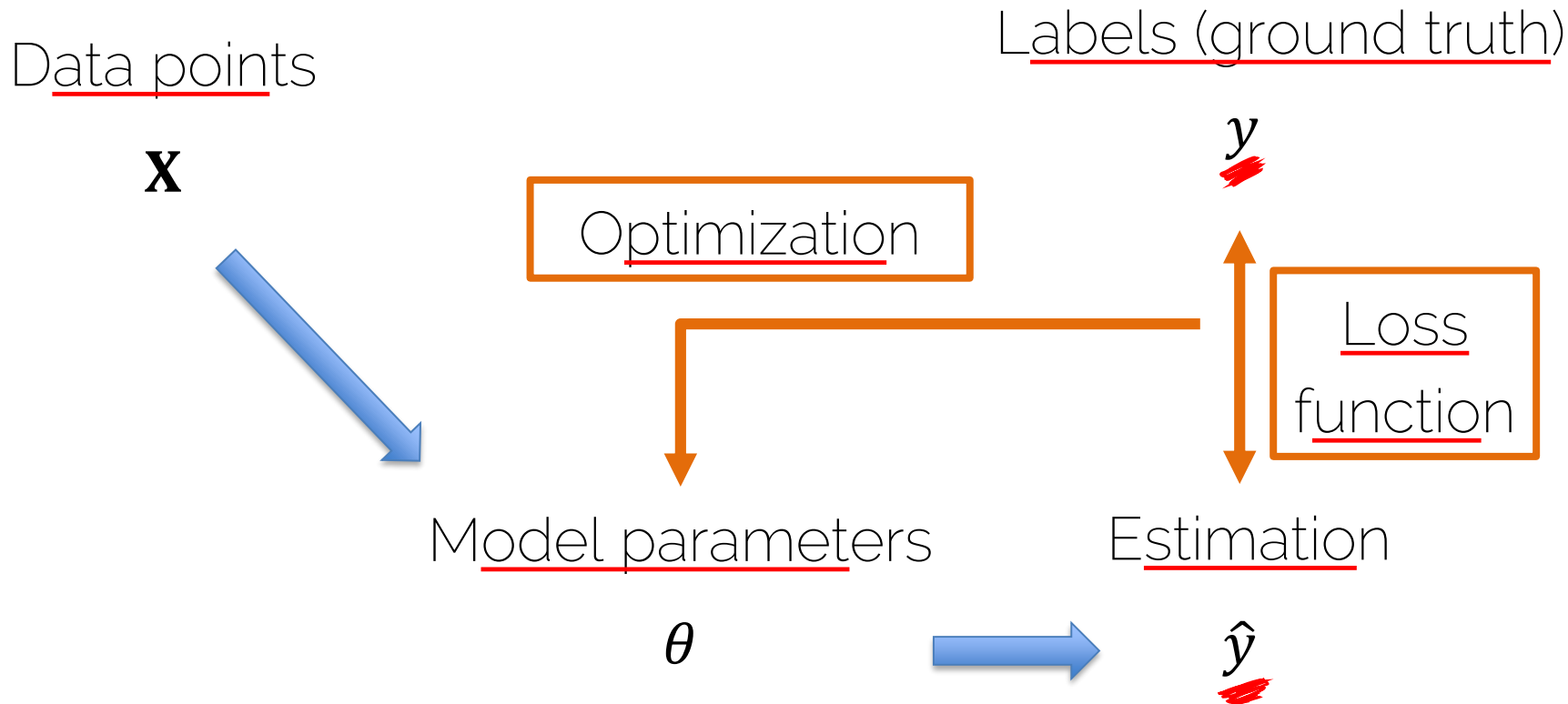
MODEL

$$\begin{bmatrix} 0.2 \\ 0.64 \\ 0 \\ 1 \\ 0.14 \end{bmatrix}$$

How do we  
obtain the  
model?



# How to Obtain the Model?

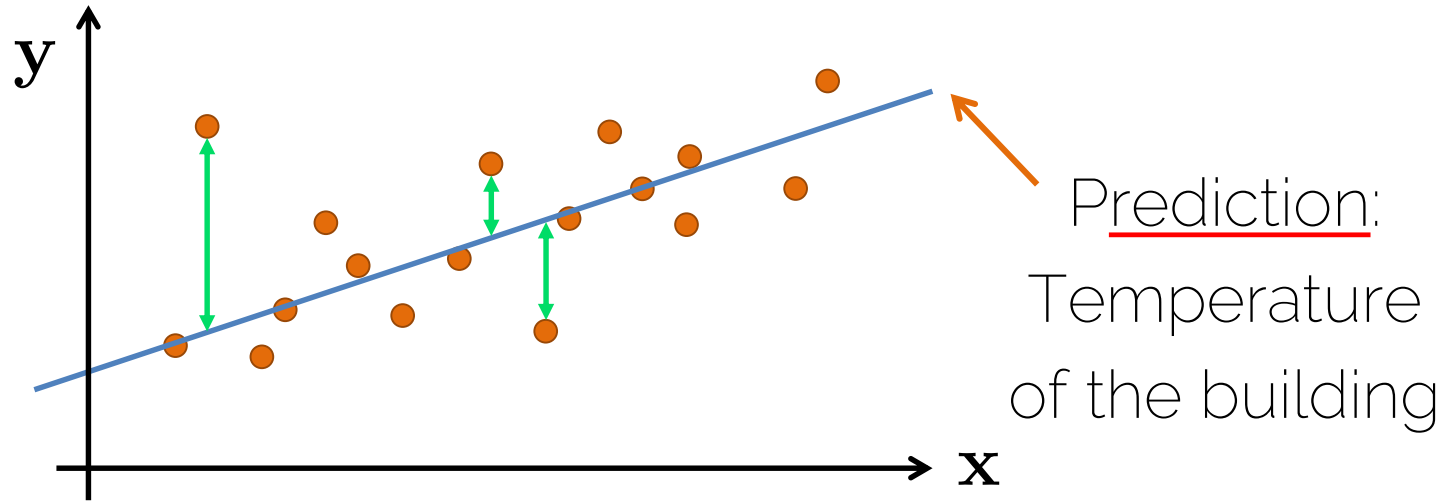


# How to Obtain the Model?

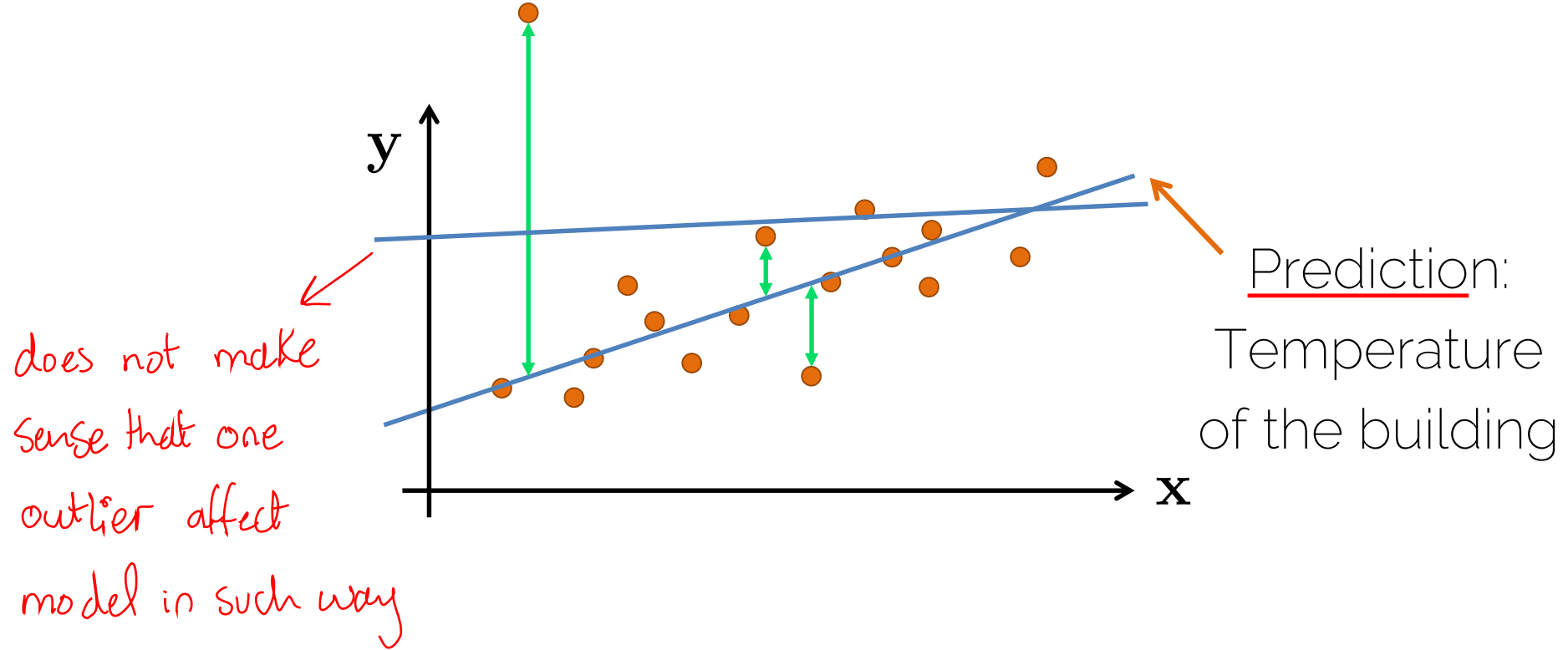
- **Loss function:** measures how good my estimation is (how good my model is) and tells the optimization method how to make it better.
- **Optimization:** changes the model in order to improve the loss function (i.e., to improve my estimation).

*always hand-in-hand*

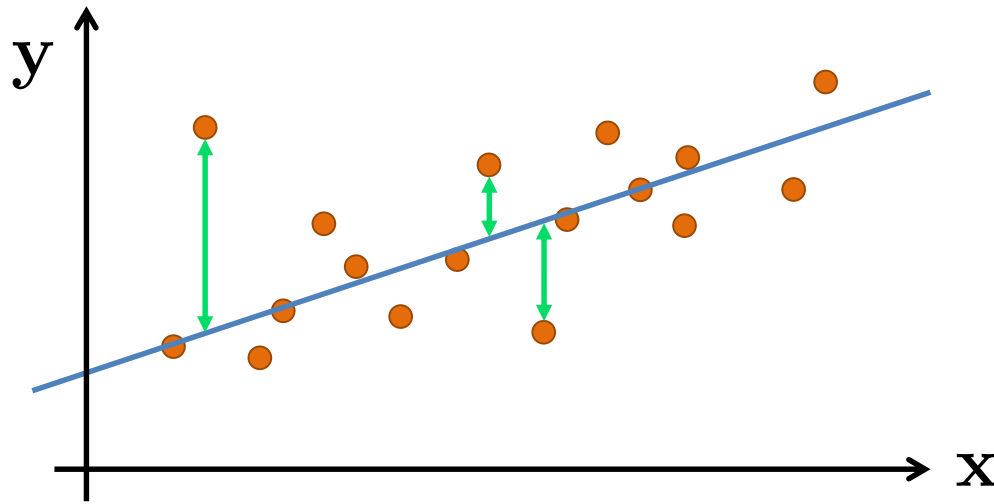
# Linear Regression: Loss Function



# Linear Regression: Loss Function



# Linear Regression: Loss Function



Minimizing

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$


Objective function

Energy

Cost function

# Optimization: Linear Least Squares

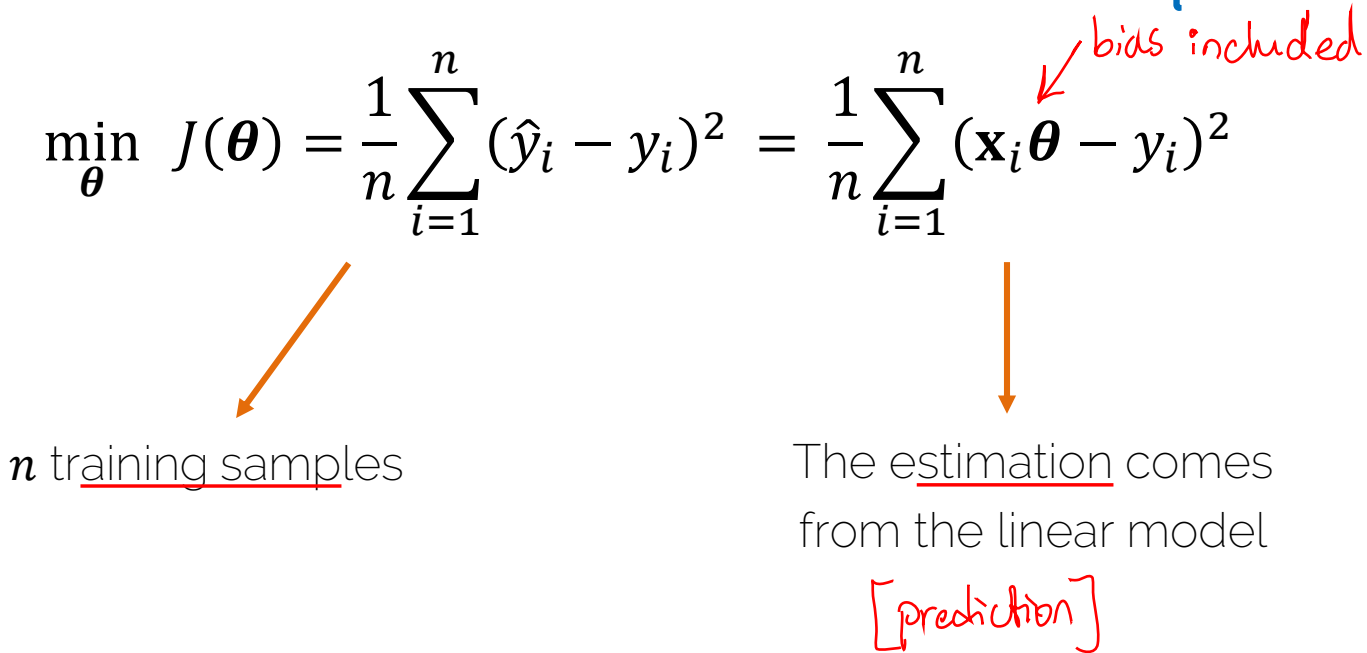
- Linear least squares: an approach to fit a linear model to the data


$$\min_{\theta} J(\theta) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

\* Convex problem, there exists a closed-form solution that is unique.

NB we normalize to make our model independent of # samples

# Optimization: Linear Least Squares

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2$$


The diagram illustrates the components of the Linear Least Squares optimization problem. It shows the cost function  $J(\boldsymbol{\theta})$  as the average squared error over  $n$  samples. The first term,  $\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$ , is linked by an orange arrow to the text " $n$  training samples". The second term,  $\frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2$ , is linked by an orange arrow to the text "The estimation comes from the linear model [prediction]". A red handwritten note "bids included" with an arrow points to the  $\mathbf{x}_i$  term in the second equation, indicating that the input features represent bids.

$n$  training samples

The estimation comes  
from the linear model

[prediction]

# Optimization: Linear Least Squares

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2$$

→  $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$

$\mathbf{X}$   $n$  training samples,  
each input vector has  
size  $d$

$\mathbf{y}$   $n$  labels

Matrix notation



# Optimization: Linear Least Squares

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2$$

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$

Matrix notation


More on matrix notation in the next exercise session

# Optimization: Linear Least Squares

$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 = \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i \boldsymbol{\theta} - y_i)^2$$

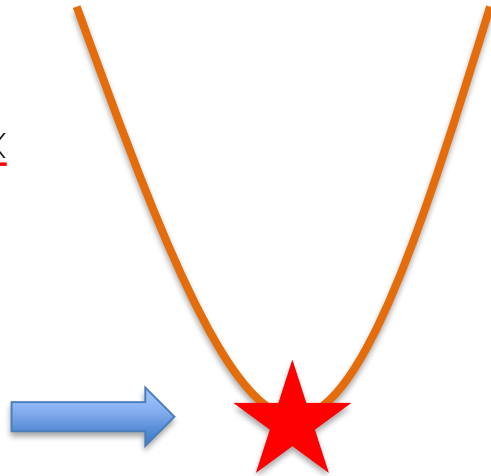
$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$

gradient-based optimization


$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0$$

Convex


Optimum



# Optimization

Details in the  
exercise  
session!

$$\frac{\partial J(\theta)}{\partial \theta} = 2\mathbf{X}^T \mathbf{X} \theta - 2\mathbf{X}^T \mathbf{y} = 0$$


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

Moore Penrose Inv  $\mathbf{X}^+$

We have found  
an analytical  
solution to a  
convex problem

Inputs: Outside  
temperature,  
number of people,  
...

True output:  
Temperature of  
the building  
ground truth

# Is this the best Estimate?

- Least squares estimate

$$J(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

Important

# Maximum Likelihood

# Maximum Likelihood Estimate

$$p_{data}(\mathbf{y}|\mathbf{X})$$

True underlying distribution



$$p_{model}(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$$

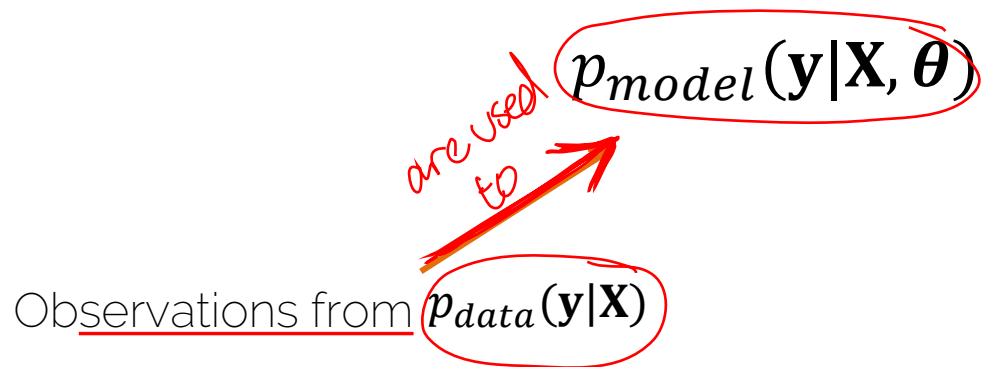
Parametric family of distributions

Controlled by parameter(s)

→ Model shall correlate to data

# Maximum Likelihood Estimate

- A method of estimating the parameters of a statistical model given observations,



# Maximum Likelihood Estimate

\* A method of estimating the parameters of a statistical model given observations, by finding the parameter values that maximize the likelihood of making the observations given the parameters.

\*  $\theta_{ML} = \arg \max_{\theta} p_{model}(\mathbf{y}|\mathbf{X}, \theta)$



# Maximum Likelihood Estimate

\* MLE assumes that the training samples are independent and generated by the same probability distribution

$$p_{model}(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^n p_{model}(y_i|\mathbf{x}_i, \boldsymbol{\theta})$$

"i.i.d." assumption

independent & identically distributed

# Maximum Likelihood Estimate

$$\theta_{ML} = \arg \max_{\theta} \prod_{i=1}^n p_{model}(y_i | \mathbf{x}_i, \theta)$$

$$\theta_{ML} = \arg \max_{\theta} \sum_{i=1}^n \log p_{model}(y_i | \mathbf{x}_i, \theta)$$



Logarithmic property  $\log ab = \log a + \log b$

# Back to Linear Regression

$$\boldsymbol{\theta}_{ML} = \arg \max_{\boldsymbol{\theta}} \sum_{i=1}^n \log p_{model}(y_i | \mathbf{x}_i, \boldsymbol{\theta})$$

What shape does our  
probability distribution  
have?

# Back to Linear Regression

$$p(y_i | \mathbf{x}_i, \boldsymbol{\theta})$$

What shape does our probability distribution have?

# Back to Linear Regression

$$p(y_i | \mathbf{x}_i, \boldsymbol{\theta})$$

Gaussian / Normal  
distribution

→ Assuming  $y_i = \mathcal{N}(\mathbf{x}_i \boldsymbol{\theta}, \sigma^2) = \mathbf{x}_i \boldsymbol{\theta} + \mathcal{N}(0, \sigma^2)$

mean

Gaussian.

→  $p(y_i) = \frac{1}{\sqrt{(2\pi\sigma^2)}} e^{-\frac{1}{2\sigma^2}(y_i - \mu)^2}$

$$y_i \sim \mathcal{N}(\mu, \sigma^2)$$

# Back to Linear Regression

$$p(y_i | \mathbf{x}_i, \boldsymbol{\theta}) = ?$$

Assuming  $y_i = \mathcal{N}(\mathbf{x}_i \boldsymbol{\theta}, \sigma^2) = \mathbf{x}_i \boldsymbol{\theta} + \mathcal{N}(0, \sigma^2)$

Gaussian:

$$p(y_i) = \frac{1}{\sqrt{(2\pi\sigma^2)}} e^{-\frac{1}{2\sigma^2}(y_i - \mu)^2}$$

$$y_i \sim \mathcal{N}(\mu, \sigma^2)$$

mean

# Back to Linear Regression

$$\rightarrow p(y_i | \mathbf{x}_i, \boldsymbol{\theta}) = (2\pi\sigma^2)^{-1/2} e^{-\frac{1}{2\sigma^2}(y_i - \mathbf{x}_i\boldsymbol{\theta})^2}$$

Assuming  $y_i = \mathcal{N}(\mathbf{x}_i\boldsymbol{\theta}, \sigma^2) = \mathbf{x}_i\boldsymbol{\theta} + \mathcal{N}(0, \sigma^2)$

Gaussian:

$$p(y_i) = \frac{1}{\sqrt{(2\pi\sigma^2)}} e^{-\frac{1}{2\sigma^2}(y_i - \mu)^2}$$


$$y_i \sim \mathcal{N}(\mu, \sigma^2)$$

mean

# Back to Linear Regression

$$p(y_i|\mathbf{x}_i, \boldsymbol{\theta}) = (2\pi\sigma^2)^{-1/2} e^{-\frac{1}{2\sigma^2}(y_i - \mathbf{x}_i\boldsymbol{\theta})^2}$$

Original  
optimization  
problem

$$\boldsymbol{\theta}_{ML} = \arg \max_{\boldsymbol{\theta}} \sum_{i=1}^n \log p_{model}(y_i|\mathbf{x}_i, \boldsymbol{\theta})$$
An orange oval highlights the log term in the summation of the ML equation. An orange arrow points from this oval to the entire likelihood function equation shown above.



# Back to Linear Regression

$$\sum_{i=1}^n \log \left[ (2\pi\sigma^2)^{-\frac{1}{2}} e^{-\frac{1}{2\sigma^2}(y_i - \mathbf{x}_i\boldsymbol{\theta})^2} \right]$$

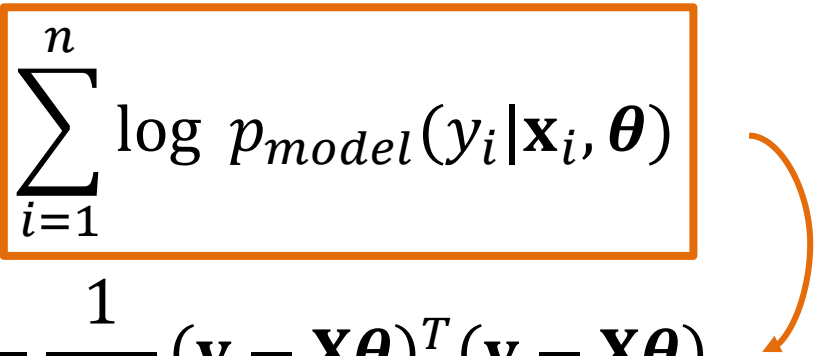
Canceling log and e

$$\sum_{i=1}^n -\frac{1}{2} \log (2\pi\sigma^2) + \sum_{i=1}^n \left( -\frac{1}{2\sigma^2} \right) (y_i - \mathbf{x}_i\boldsymbol{\theta})^2$$


Matrix notation

$$-\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})^T (\mathbf{y} - \mathbf{X}\boldsymbol{\theta})$$

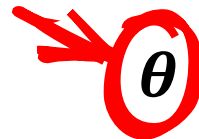
# Back to Linear Regression

$$\theta_{ML} = \arg \max_{\theta} \sum_{i=1}^n \log p_{model}(y_i | \mathbf{x}_i, \theta)$$
$$-\frac{n}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} (\mathbf{y} - \mathbf{X}\theta)^T (\mathbf{y} - \mathbf{X}\theta)$$


Details in the  
exercise session!

$$\frac{\partial J(\theta)}{\partial \theta} = 0$$


How can we find  
the estimate of  
theta?


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

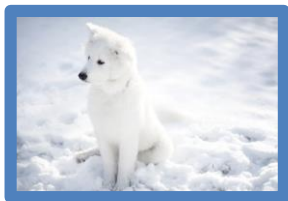
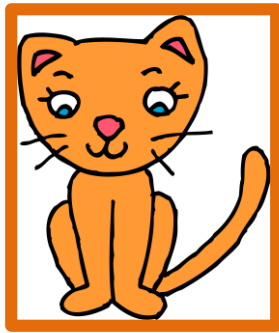
# Linear Regression

\* Maximum Likelihood Estimate (MLE) corresponds to the Least Squares Estimate (given the assumptions)

[important conclusion]

- Introduced the concepts of loss function and optimization to obtain the best model for regression

# Image Classification



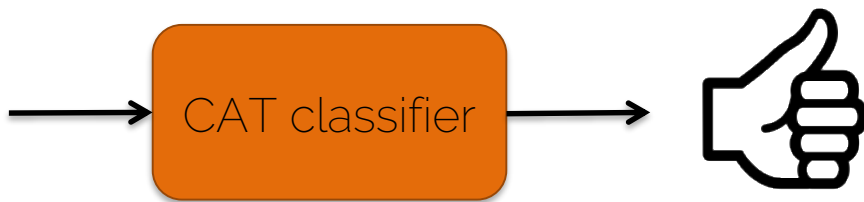
# Regression vs Classification

- **Regression:** predict a continuous output value (e.g., temperature of a room)
- **Classification:** predict a discrete value
  - Binary classification: output is either 0 or 1
  - Multi-class classification: set of N classes

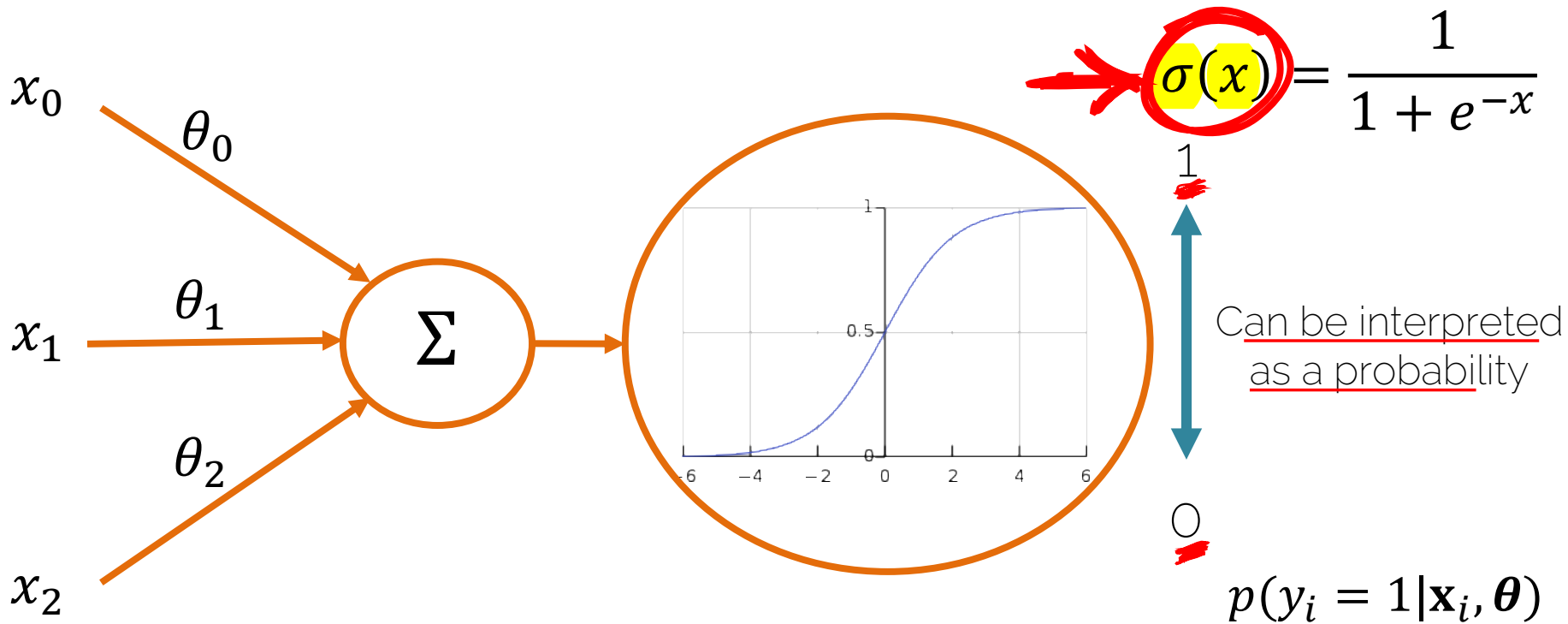


Important

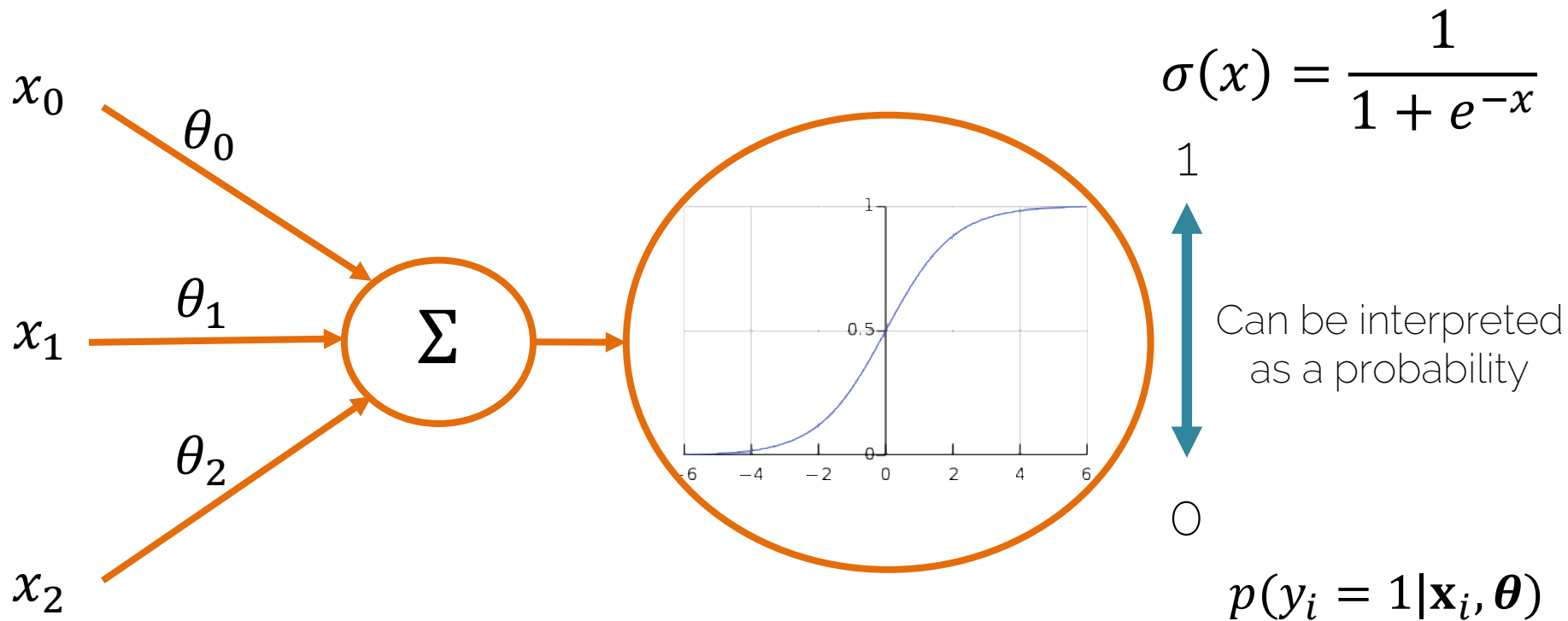
# Logistic Regression



# Sigmoid for Binary Predictions



# Spoiler Alert: 1-Layer Neural Network






# Logistic Regression

- Probability of a binary output

$$\hat{\mathbf{y}} = p(\mathbf{y} = 1 | \mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^n p(y_i = 1 | \mathbf{x}_i, \boldsymbol{\theta})$$

The prediction of  
our sigmoid


$$\hat{y}_i = \sigma(\mathbf{x}_i \boldsymbol{\theta})$$

# Logistic Regression

- Probability of a binary output

$$\hat{\mathbf{y}} = p(\mathbf{y} = 1 | \mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^n p(y_i = 1 | \mathbf{x}_i, \boldsymbol{\theta})$$

Bernoulli trial

Model for  
coins

The diagram illustrates the components of a Bernoulli trial in the context of a coin model. A red arrow points from the 'Bernoulli trial' label to the term  $p(z|\phi)$  in the equation below. Another red arrow points from the 'Model for coins' label to the same term. The term  $p(z|\phi)$  is circled in red. An orange arrow points from the product term in the equation above to the term  $\phi^z(1-\phi)^{1-z}$  in the equation below. A second orange arrow points from the same term to the text 'The prediction of our sigmoid'.

$$p(z|\phi) = \phi^z (1 - \phi)^{1-z} = \begin{cases} \phi & , \text{ if } z = 1 \\ 1 - \phi & , \text{ if } z = 0 \end{cases}$$

The prediction of  
our sigmoid

# Logistic Regression

- Probability of a binary output

$$\hat{\mathbf{y}} = p(\mathbf{y} = 1 | \mathbf{X}, \boldsymbol{\theta}) = \prod_{i=1}^n p(y_i = 1 | \mathbf{x}_i, \boldsymbol{\theta})$$

$\hat{\mathbf{y}} = \prod_{i=1}^n \hat{y}_i^{y_i} (1 - \hat{y}_i)^{(1-y_i)}$

Prediction of the Sigmoid: continuous

True labels: 0 or 1

Model for coins

# Logistic Regression: Loss Function

- Probability of a binary output

$$p(y|\mathbf{X}, \boldsymbol{\theta}) = \hat{\mathbf{y}} = \prod_{i=1}^n \hat{y}_i^{y_i} (1 - \hat{y}_i)^{(1-y_i)}$$



- Maximum Likelihood Estimate

$$\boldsymbol{\theta}_{ML} = \arg \max_{\boldsymbol{\theta}} \log p(y|\mathbf{X}, \boldsymbol{\theta})$$

# Logistic Regression: Loss Function

$$p(y|\mathbf{X}, \boldsymbol{\theta}) = \hat{\mathbf{y}} = \prod_{i=1}^n \hat{y}_i^{y_i} (1 - \hat{y}_i)^{(1-y_i)}$$

$$\sum_{i=1}^n \log (\hat{y}_i^{y_i} (1 - \hat{y}_i)^{(1-y_i)})$$


$$\sum_{i=1}^n y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$


# Logistic Regression: Loss Function

$$\mathcal{L}(\hat{y}_i, y_i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

# 1

$$y_i = 1 \longrightarrow \mathcal{L}(\hat{y}_i, 1) = \log \hat{y}_i$$

Maximize!

$$\boldsymbol{\theta}_{ML} = \arg \max_{\boldsymbol{\theta}} \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta})$$

# Logistic Regression: Loss Function

$$\mathcal{L}(\hat{y}_i, y_i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

$$y_i = 1 \longrightarrow \mathcal{L}(\hat{y}_i, 1) = \log \hat{y}_i$$

NB

We want  $\log \hat{y}_i$  large; since logarithm is a monotonically increasing function, we also want large  $\hat{y}_i$ .

1 is the largest value our model's estimate can take!

# Logistic Regression: Loss Function

$$\mathcal{L}(\hat{y}_i, y_i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

#2

$$y_i = 1 \longrightarrow \mathcal{L}(\hat{y}_i, 1) = \log \hat{y}_i$$


$$y_i = 0 \longrightarrow \mathcal{L}(\hat{y}_i, 0) = \log(1 - \hat{y}_i)$$

We want  $\log(1 - \hat{y}_i)$  large; so we want  $\hat{y}_i$  to be small

(0 is the smallest value our model's estimate can take!)



# Logistic Regression: Loss Function


$$\mathcal{L}(\hat{y}_i, y_i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

Referred to as *binary cross-entropy* loss (BCE)

- Related to the multi-class loss you will see in this course (also called *softmax loss*)


$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^K e^{z_j}}$$

# Logistic Regression: Optimization

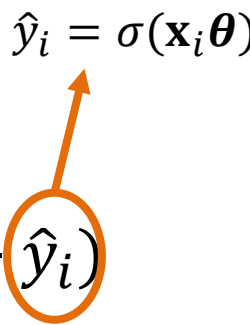
- Loss function

$$\mathcal{L}(\hat{y}_i, y_i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

- Cost function


$$C(\theta) = -\frac{1}{n} \sum_{i=1}^n \mathcal{L}(\hat{y}_i, y_i)$$

Minimization

$$= -\frac{1}{n} \sum_{i=1}^n y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$


# Logistic Regression: Optimization

 No closed-form solution

- Make use of an iterative method → gradient descent

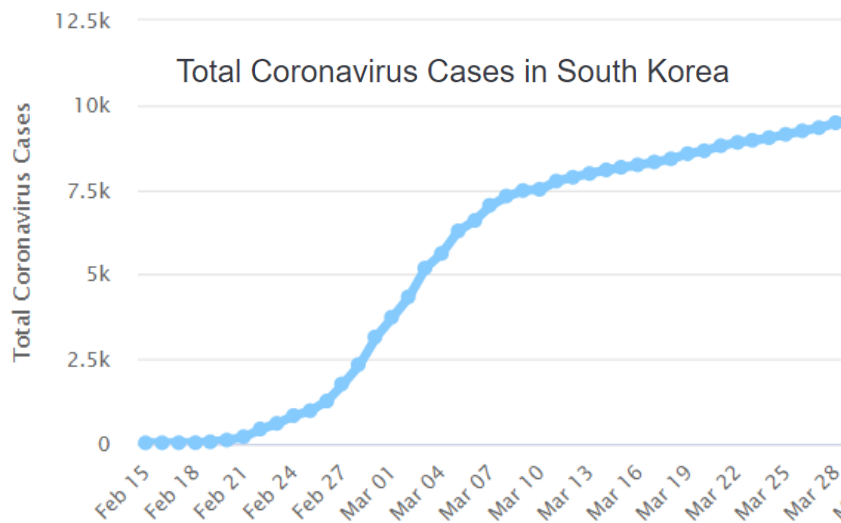
Gradient descent –  
later on!

# Why Machine Learning so Cool

- We can learn from experience
  - > Intelligence, certain ability to infer the future!
- Even linear models are often pretty good for complex phenomena: e.g., weather:
  - Linear combination of day-time, day-year etc. is often pretty good

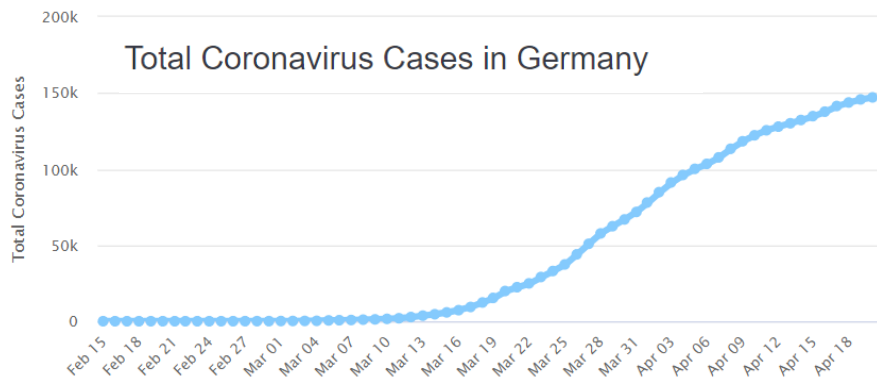
# Many Examples of Logistic Regression

- Coronavirus models behave like logistic regressions
  - Exponential spread at beginning
  - Plateaus when certain portion of pop. is infected/immune



# Many Examples of Logistic Regression

- Coronavirus models behave like logistic regressions
  - Exponential spread at beginning
  - Plateaus when certain portion of pop. is infected/immune



Think about good features:

- Total population
- Population density
- Implementation of Measures
- Reasonable government 😊 ?
- Etc. (many more of course)

# The Model Matters

- Each case requires different models; linear vs logistic
- Many models:
  - #coronavirus\_infections cannot be  $>$  #total\_population
  - Munich housing prizes seem exponential though
    - No hard upper bound  $\rightarrow$  prizes can always grow!

# Next Lectures

- Next exercise session: Math Recap II
- Next Lecture: Lecture 3:
  - Jumping towards our first Neural Networks and Computational Graphs



# References for further Reading

- Cross validation:
  - <https://medium.com/@zstern/k-fold-cross-validation-explained-5aeba90ebb3>
  - <https://towardsdatascience.com/train-test-split-and-cross-validation-in-python-80b61beca4b6>
- General Machine Learning book:
  - Pattern Recognition and Machine Learning. C. Bishop.

See you next week 😊