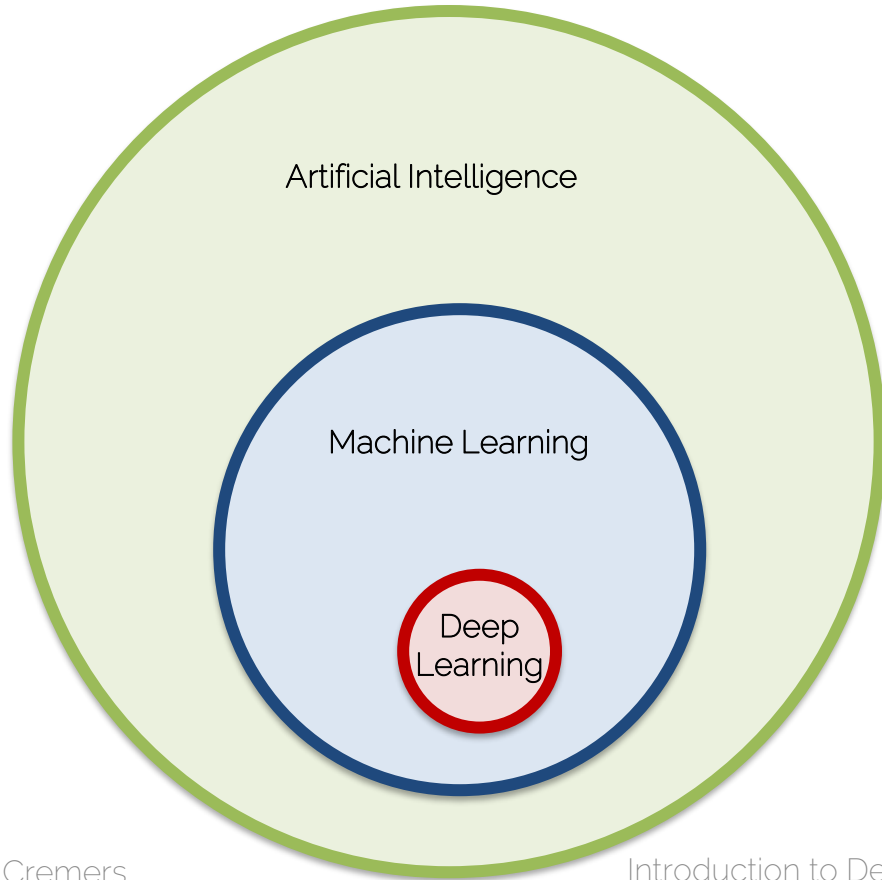


Machine Learning Basics

AI vs ML vs DL



A Simple Task: Image Classification

Image Classification



Task

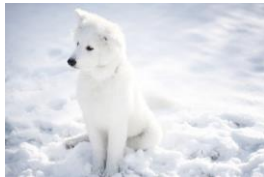


Image Classification

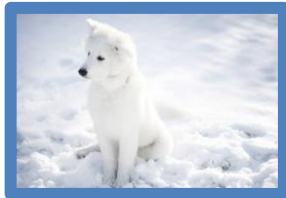
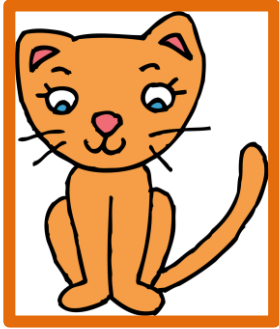


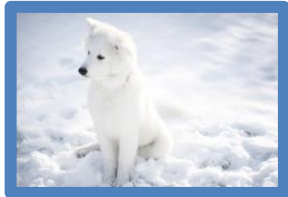
Image Classification



Occlusions

Image Classification

Background clutter





Cute



And Kittens



Clipart



Drawing



Cute Baby



White Cats And Kittens



Pose



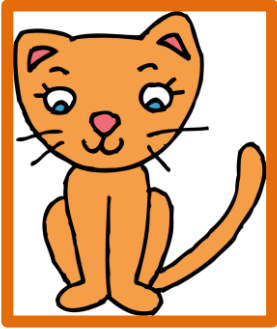
Illumination



Appearance



Image Classification

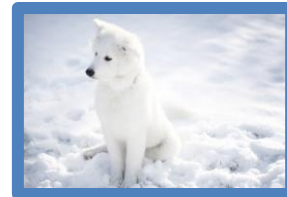
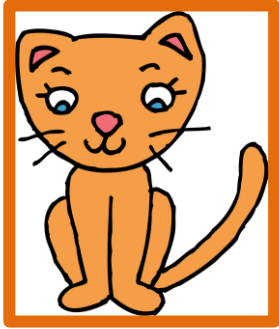


Representation



A Simple Classifier

Nearest Neighbor



Nearest Neighbor

NN classifier = dog



distance

Nearest Neighbor

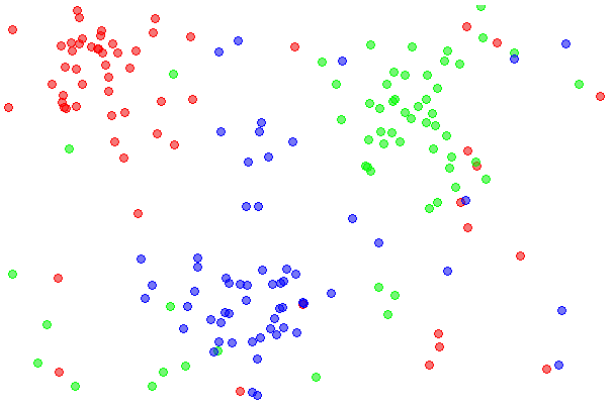
k-NN classifier = cat



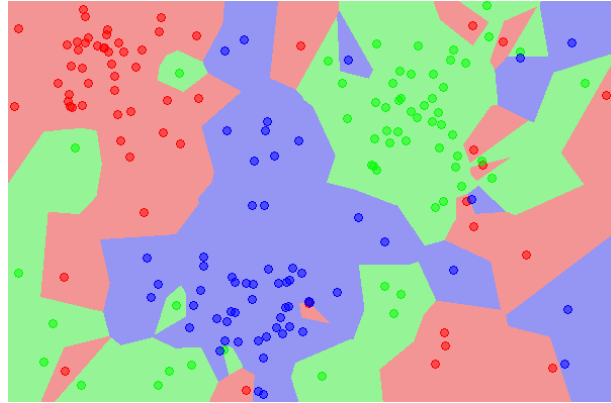
distance

Nearest Neighbor

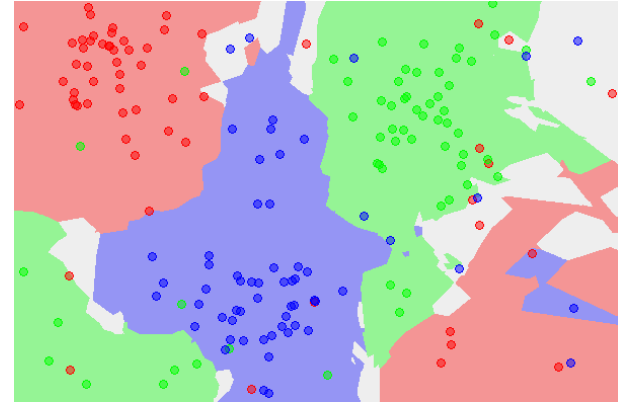
The Data



NN Classifier



5NN Classifier

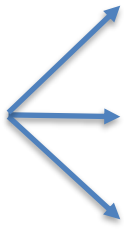


How does the NN classifier perform on training data?

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

What are we actually learning?

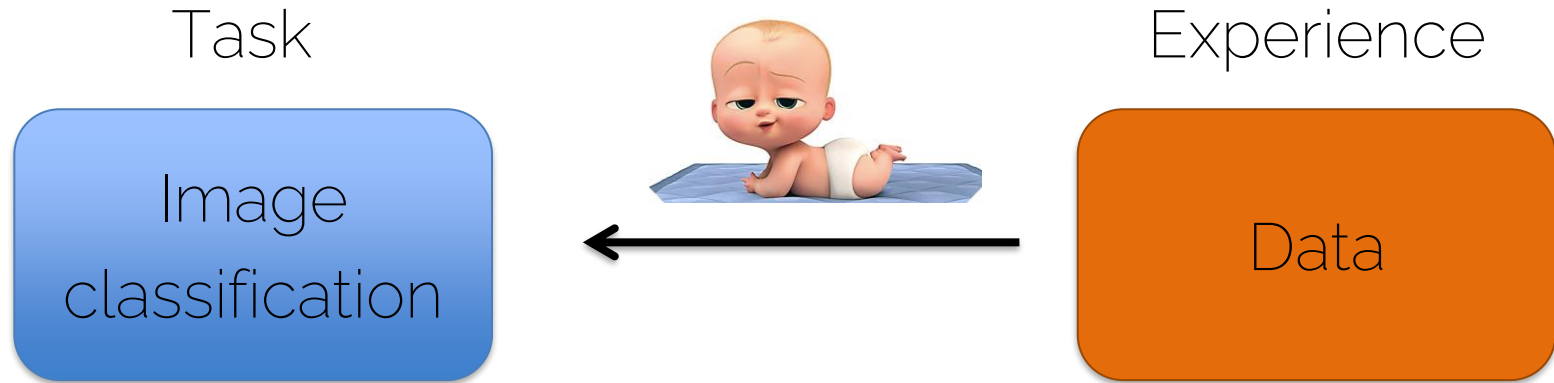
Nearest Neighbor

- Hyperparameters 
 - L1 distance : $|\mathbf{x} - \mathbf{c}|$
 - L2 distance : $\|\mathbf{x} - \mathbf{c}\|_2$
 - No. of Neighbors: k
- These parameters are problem dependent.
- How do we choose these hyperparameters?

Machine Learning for Classification

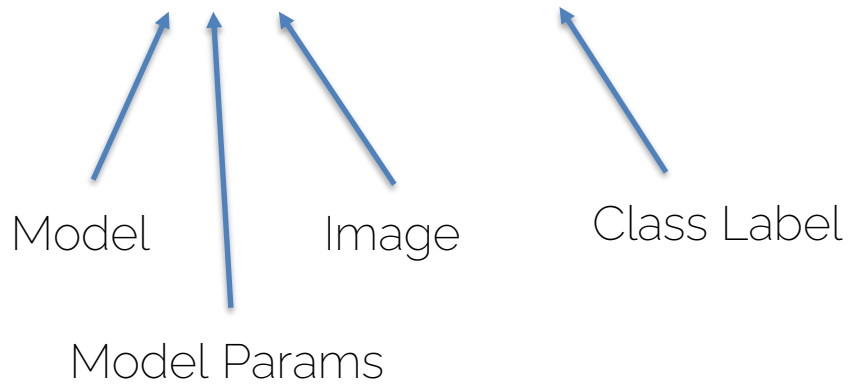
Machine Learning

- How can we learn to perform image classification?



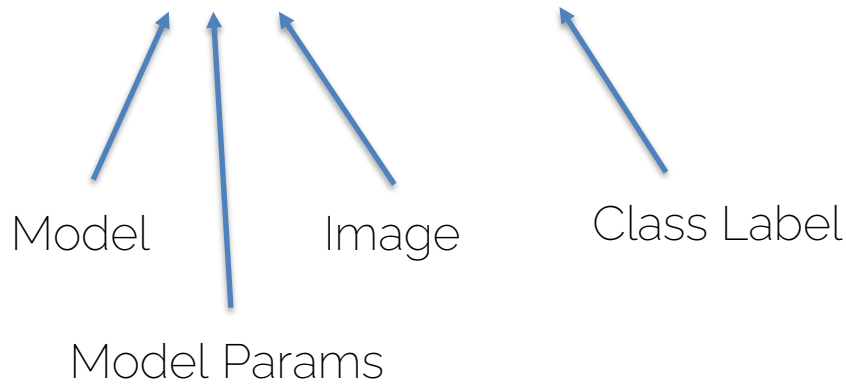
Machine Learning

- $M_{\theta}(I) = \{\text{DOG}, \text{CAT}\}$



Machine Learning

- $M_{\theta}(I) = \{\text{DOG}, \text{CAT}\}$



Given i images with train labels



$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_i D(M_{\theta}(I_i) - Y_i)$$

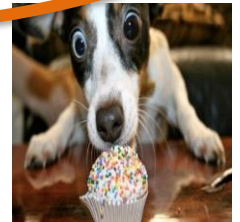
"Distance" function $\{\text{DOG}, \text{CAT}\}$



CAT



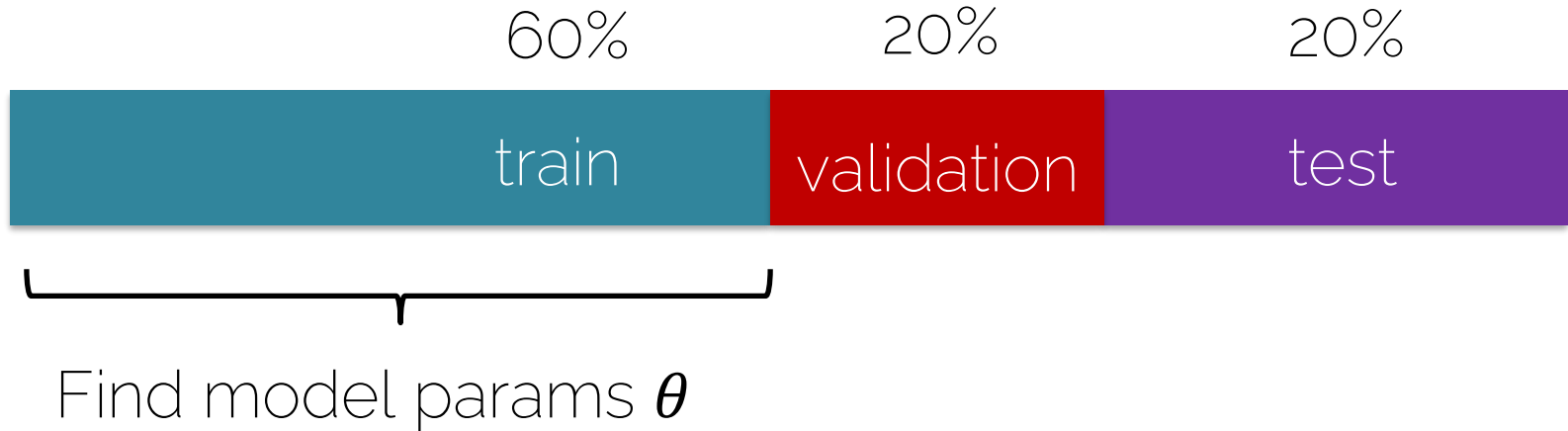
CAT



DOG

Basic Recipe for Machine Learning

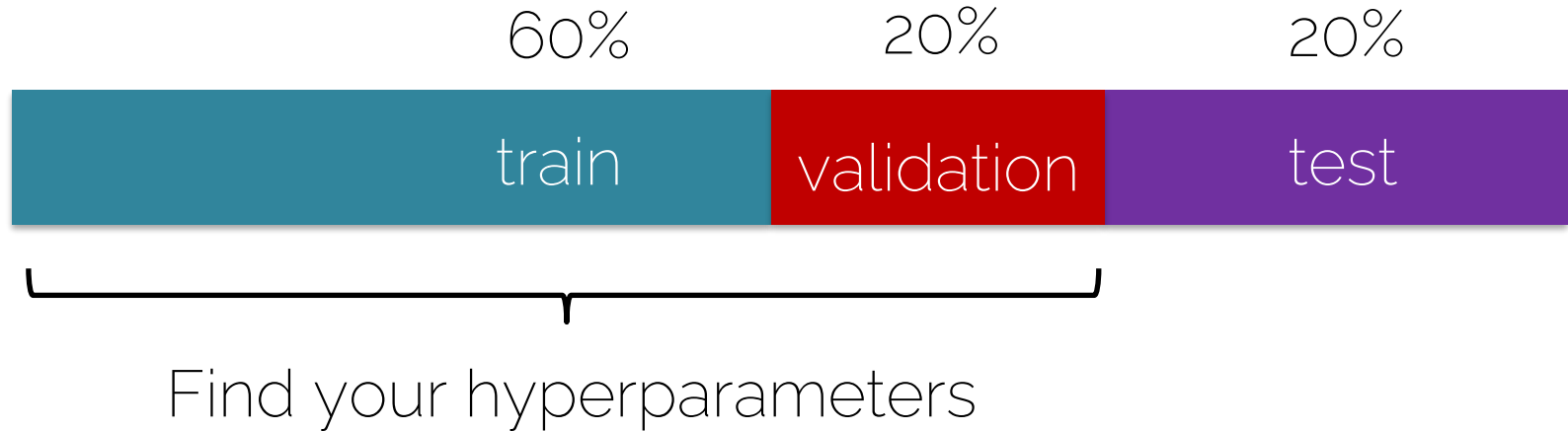
- Split your data



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

Basic Recipe for Machine Learning

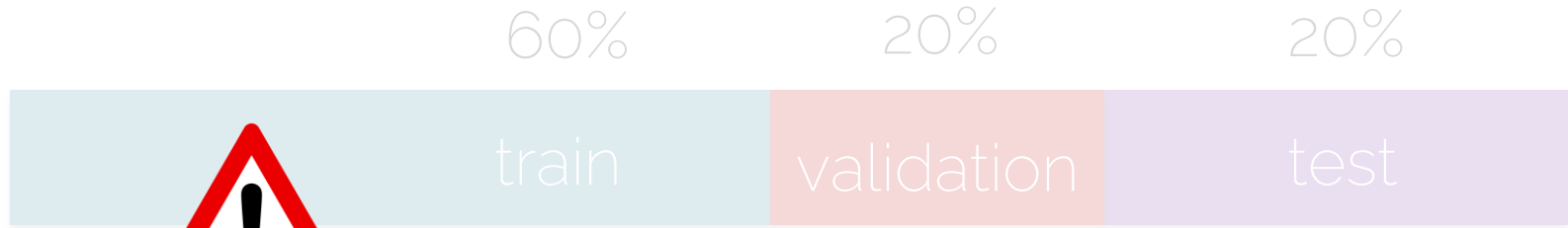
- Split your data



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

Basic Recipe for Machine Learning

- Split your data



Test set is only used once!

Machine Learning

- How can we learn to perform image classification?

Task



Performance
measure



Experience



Machine Learning

Unsupervised learning

Supervised learning

- Labels or target classes

Machine Learning

Unsupervised learning

Supervised learning

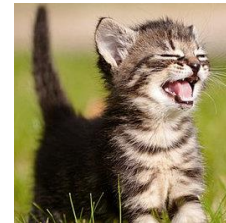


DOG

DOG



CAT



DOG

Machine Learning

Unsupervised learning

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

Supervised learning

CAT



DOG



DOG



CAT

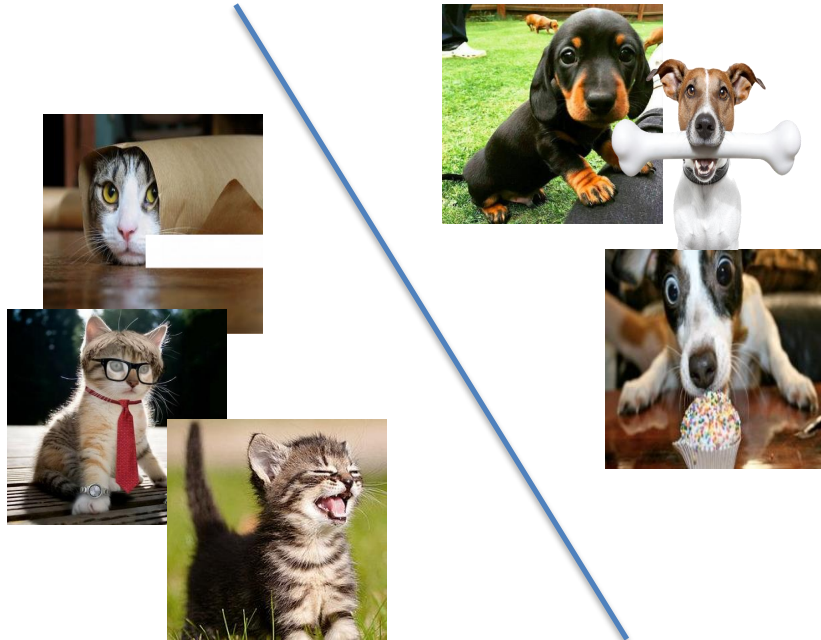
CAT



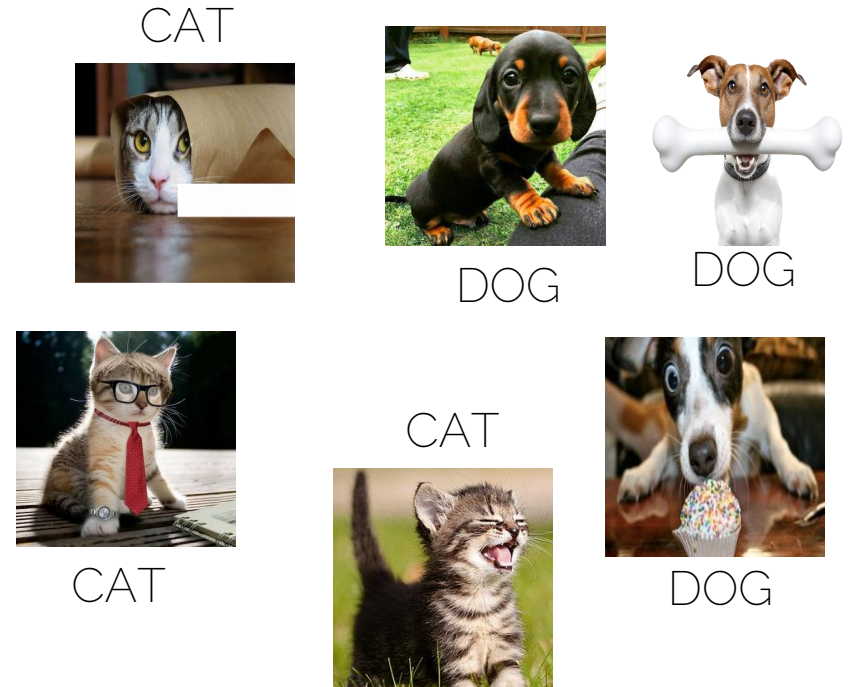
DOG

Machine Learning

Unsupervised learning

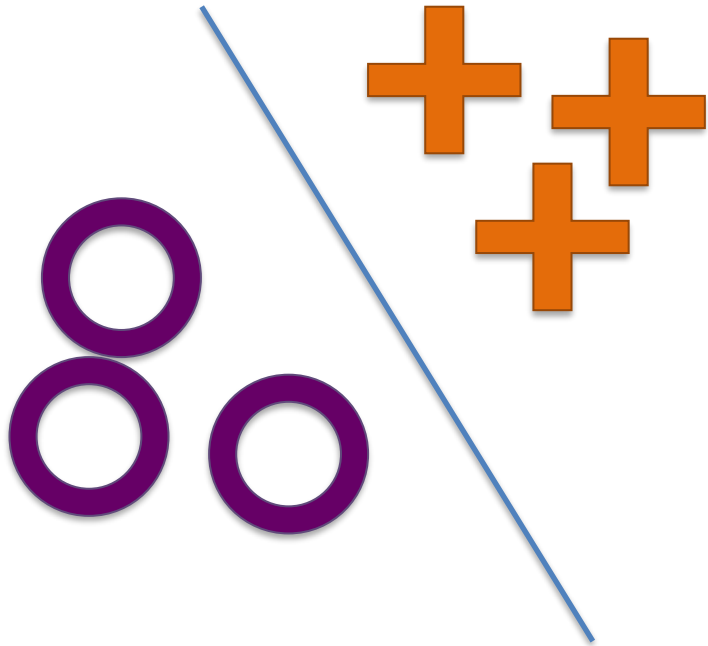


Supervised learning



Machine Learning

Unsupervised learning



Supervised learning

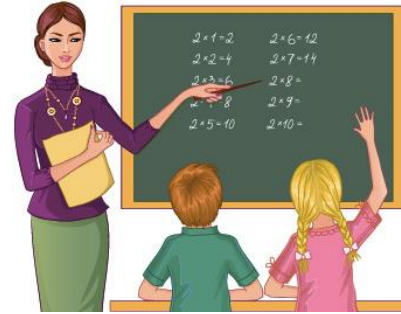


Machine Learning

Unsupervised learning



Supervised learning



Reinforcement learning



Machine Learning

Unsupervised learning



Supervised learning



Reinforcement learning

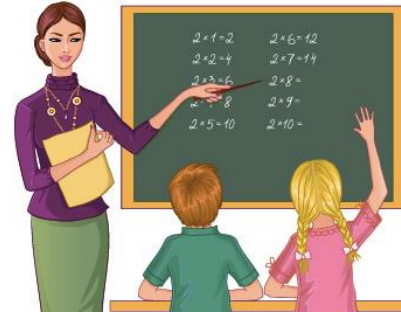


Machine Learning

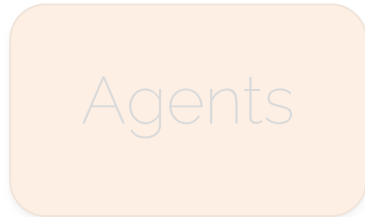
Unsupervised learning



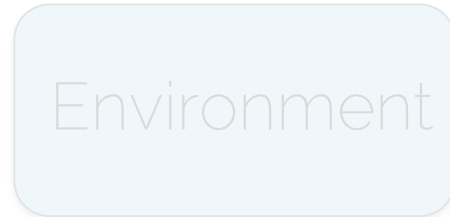
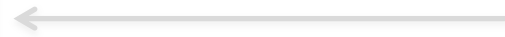
Supervised learning



Reinforcement learning

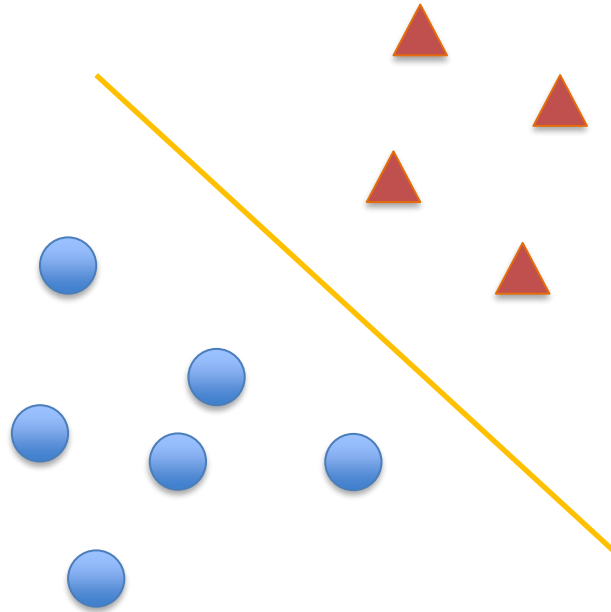


reward



Linear Decision Boundaries

Let's start with a simple linear Model!

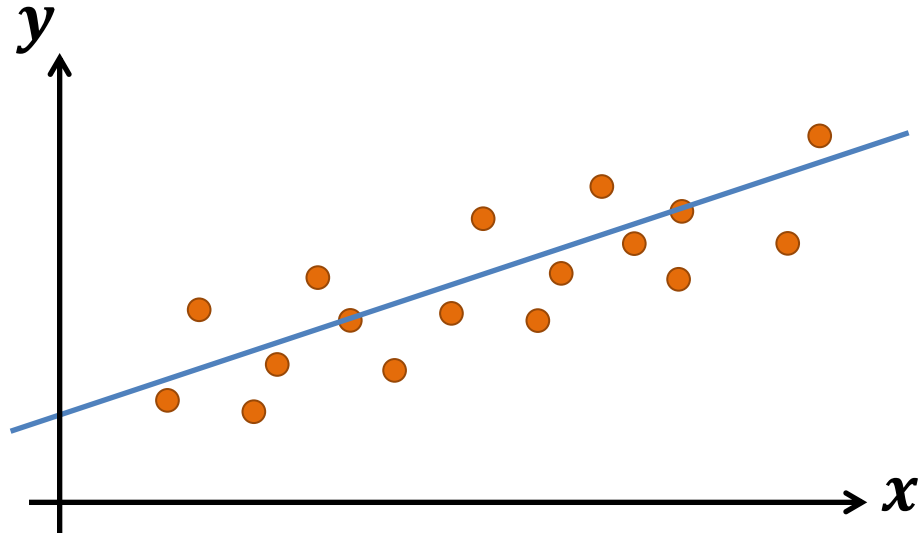


What are the pros and cons for using linear decision boundaries?

Linear Regression

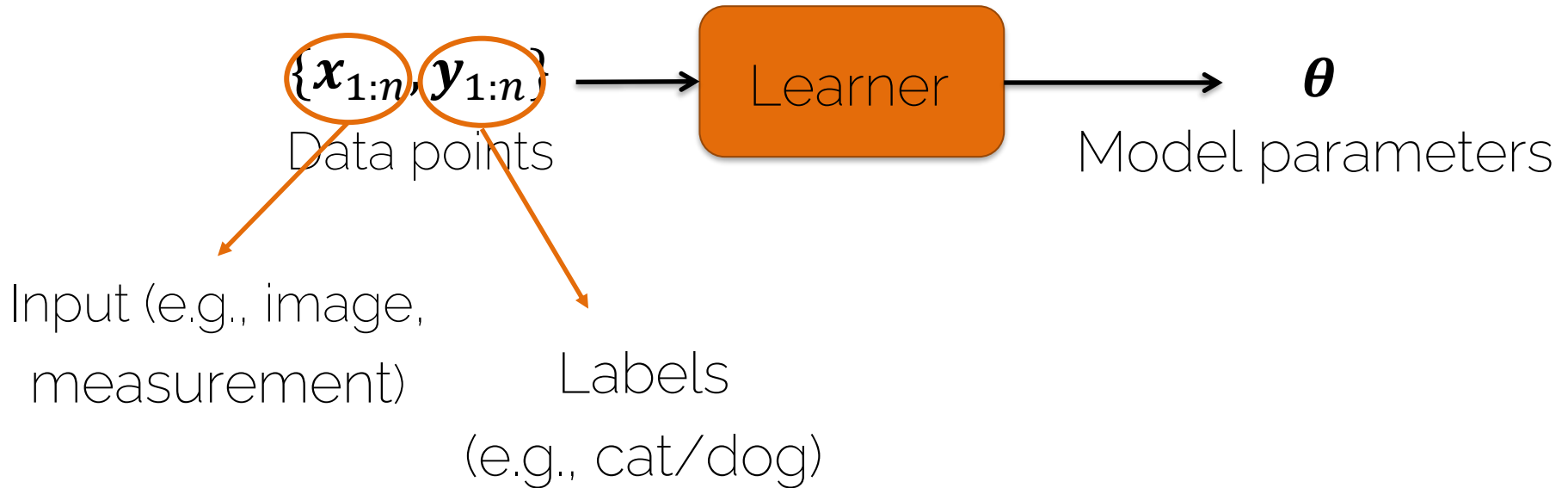
Linear Regression

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



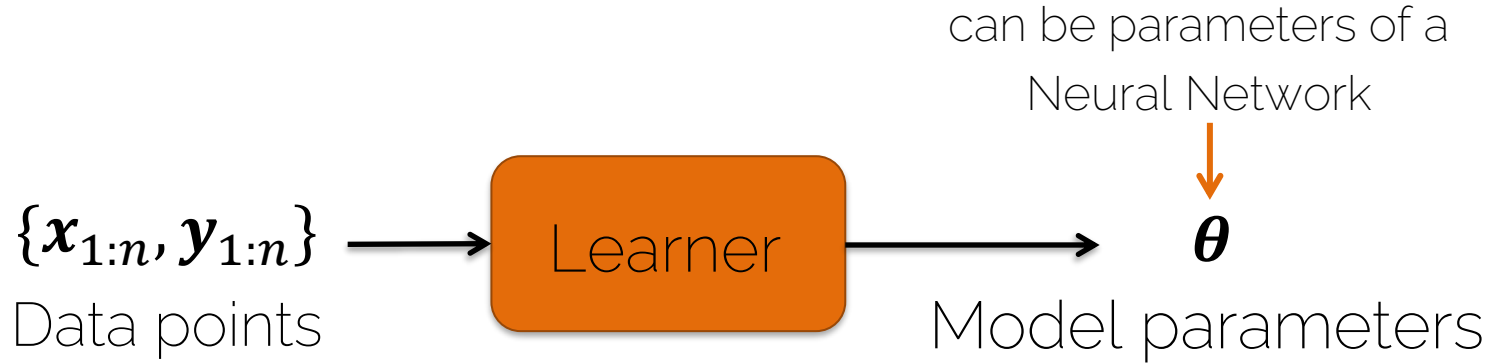
Linear Regression

Training



Linear Regression

Training



Testing



Linear Prediction

- A linear model is expressed in the form

$$\hat{y}_i = \sum_{j=1}^d x_{ij} \theta_j$$

The diagram shows the equation $\hat{y}_i = \sum_{j=1}^d x_{ij} \theta_j$ with several annotations. A purple arrow points from the text "input dimension" to the superscript d . An orange circle highlights the term x_{ij} , with an orange arrow pointing from the text "Input data, features" below to it. A blue circle highlights the term θ_j , with a blue arrow pointing from the text "weights (i.e., model parameters)" to it.

input dimension

Input data, features

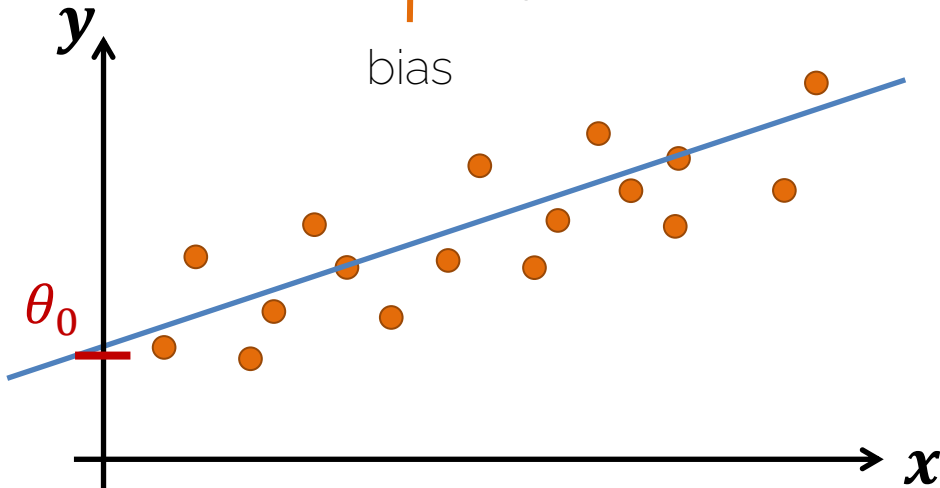
weights (i.e., model parameters)

Linear Prediction

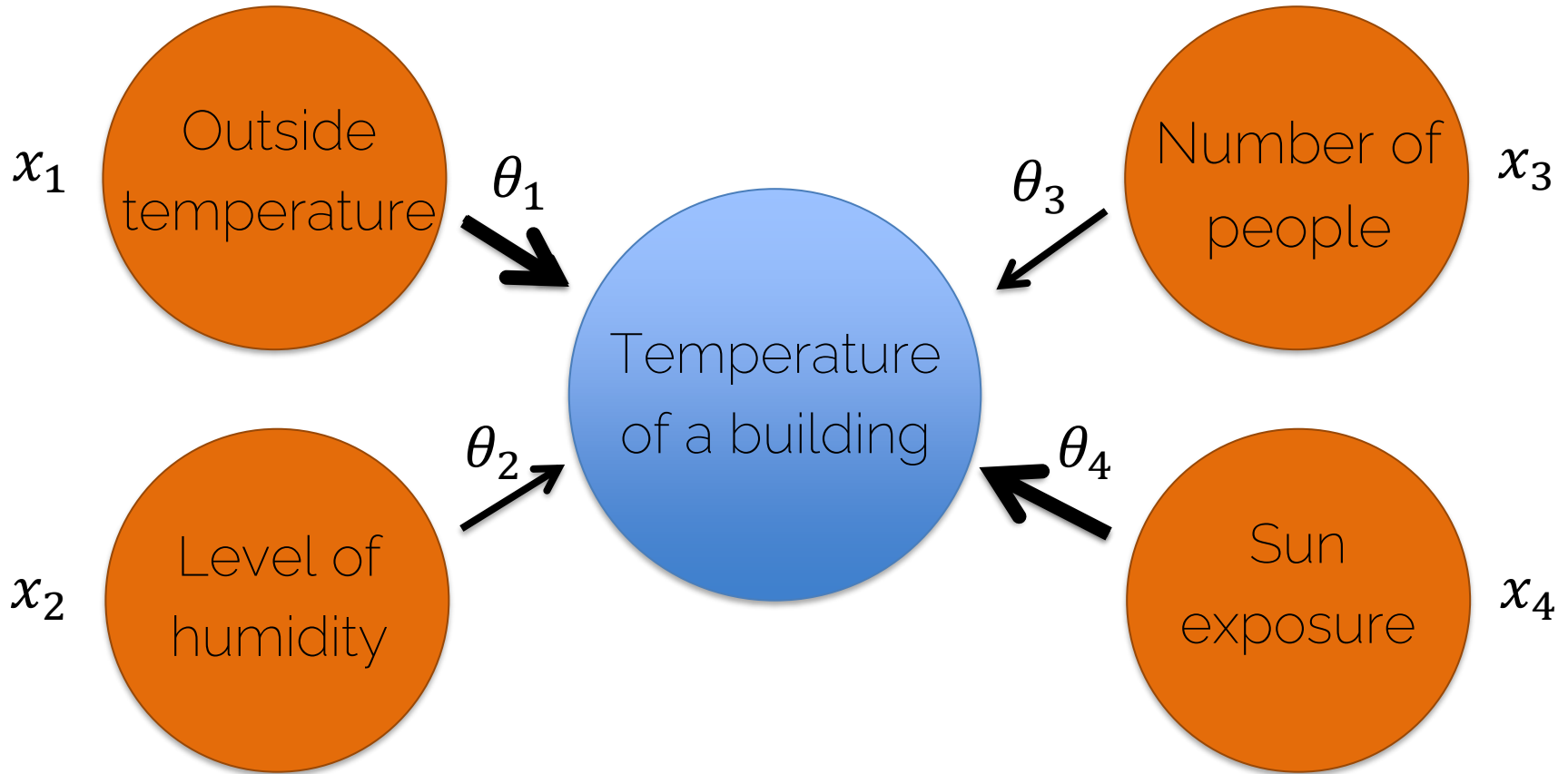
- A linear model is expressed in the form

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


↑
bias



Linear Prediction



Linear Prediction

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


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

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

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

$$\begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \end{bmatrix} = \begin{bmatrix} \text{Bias} & \text{Outside temperature} & \text{Humidity} & \text{Number people} & \text{Sun exposure (\%)} \\ 1 & 25 & 50 & 2 & 50 \\ 1 & -10 & 50 & 0 & 10 \end{bmatrix} \cdot \begin{bmatrix} \text{MODEL} \\ 0.2 \\ 0.64 \\ 0 \\ 1 \\ 0.14 \end{bmatrix}$$

Linear Prediction

How do we
obtain the
model?



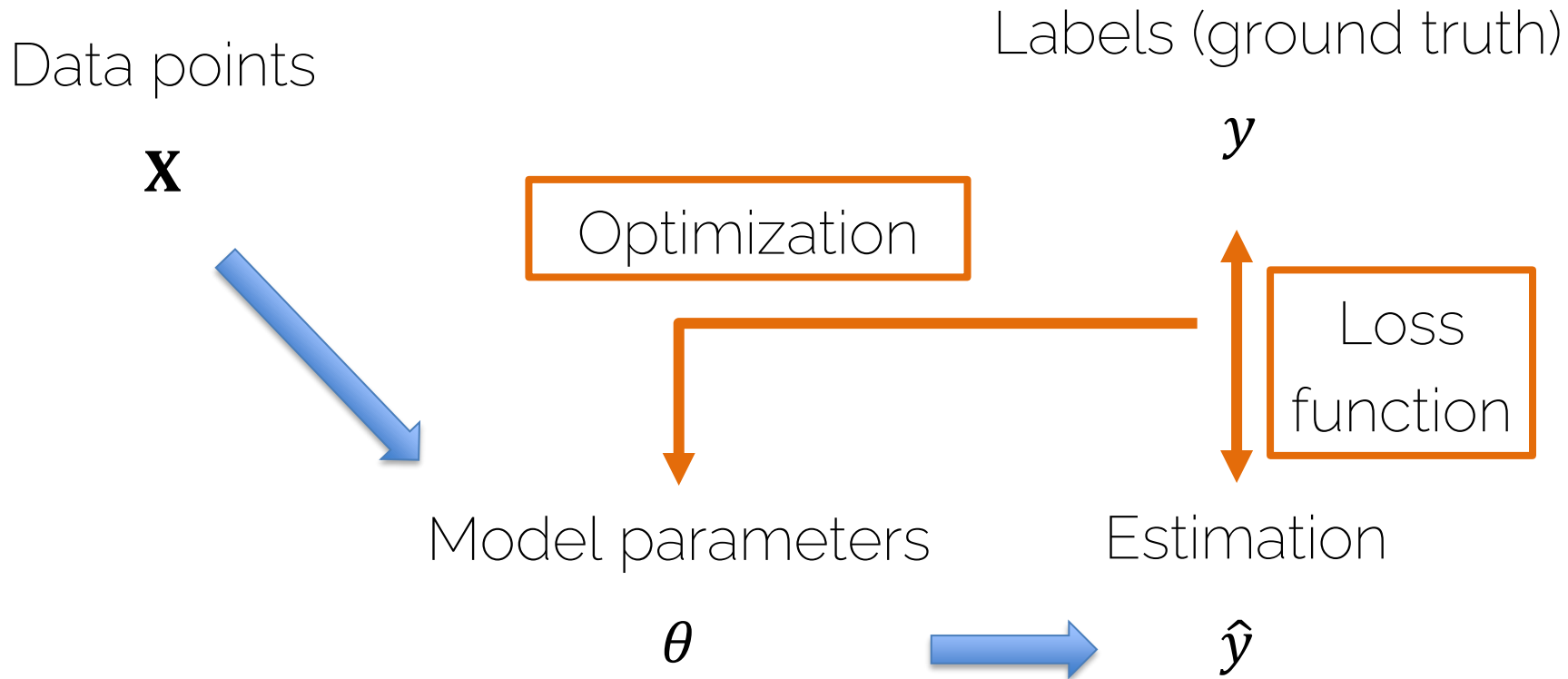
Temperature
of the building

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

The vector $\begin{bmatrix} 0.2 \\ 0.64 \\ 0 \\ 1 \\ 0.14 \end{bmatrix}$ is circled in orange and labeled "MODEL".

Labels for the matrix columns: Bias, Outside temperature, Humidity, Number people, Sun exposure (%)

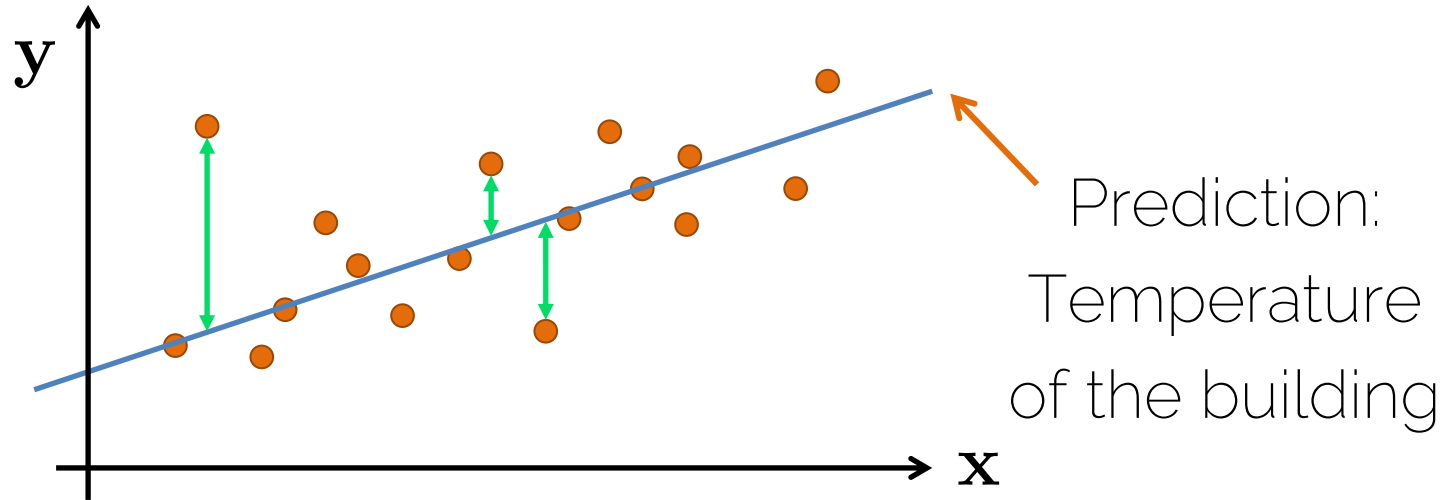
How to Obtain the Model?



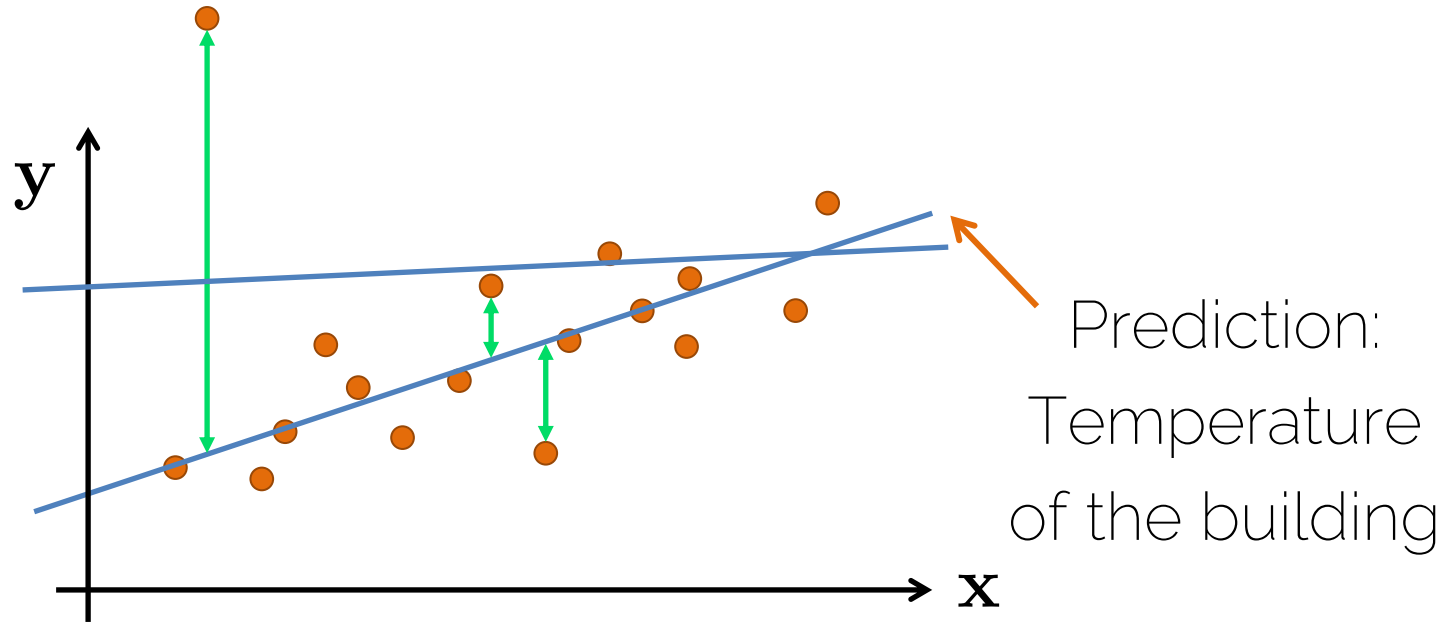
How to Obtain the Model?

- **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).

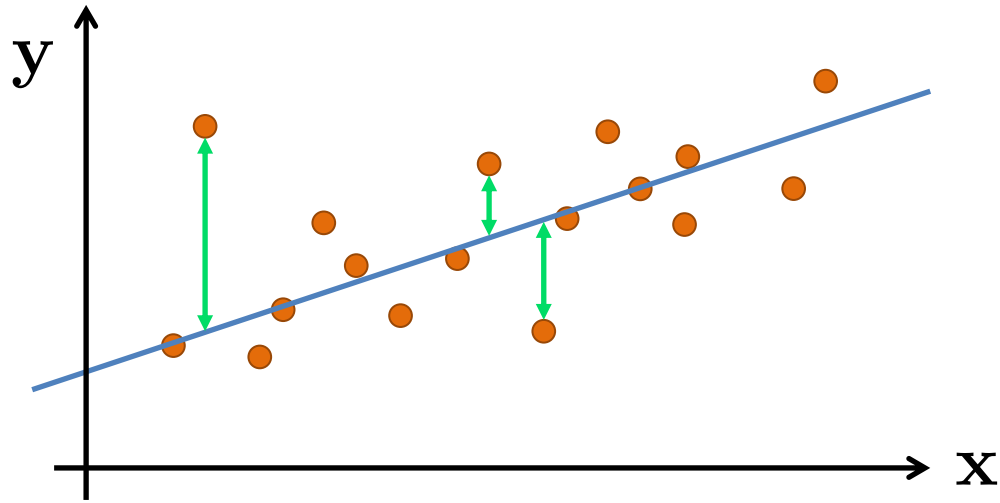
Linear Regression: Loss Function



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.

Optimization: Linear Least Squares

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



n training samples




The estimation comes from the linear model

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



n training samples,
each input vector has
size d



n labels

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

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

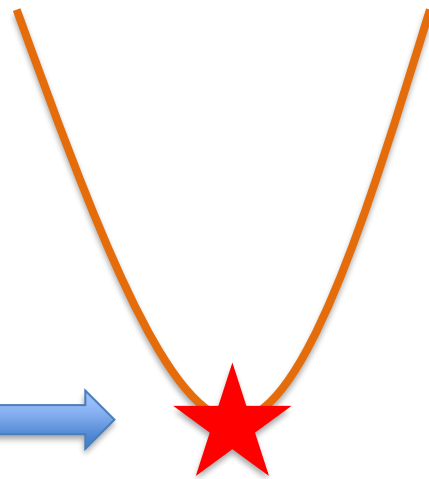
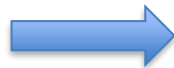
$$\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})^T (\mathbf{X}\boldsymbol{\theta} - \mathbf{y})$$



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

Convex

Optimum



Optimization

Details in the
exercise
session!

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

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

We have found
an analytical
solution to a
convex problem

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

True output:
Temperature of
the building

...

Is this the best Estimate?

- Least squares estimate

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

Maximum Likelihood

Maximum Likelihood Estimate

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

True underlying distribution



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

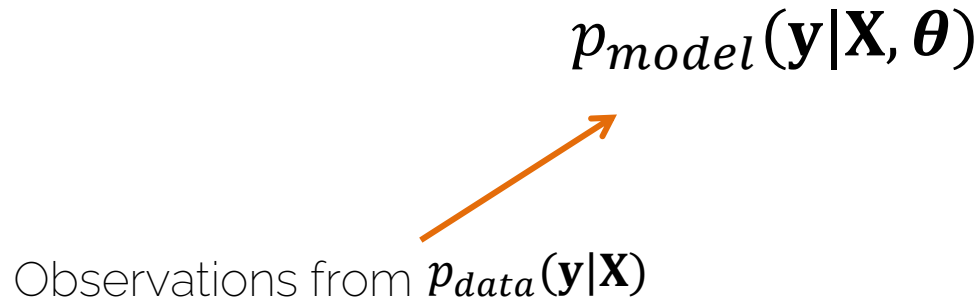
Parametric family of distributions



Controlled by parameter(s)

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.

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

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

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

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

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

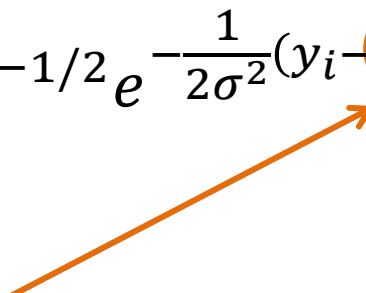
Gaussian:

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


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

mean


Back to Linear Regression

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


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

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

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!

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

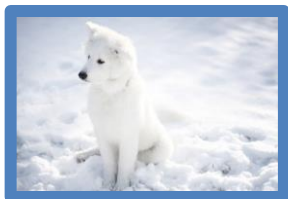
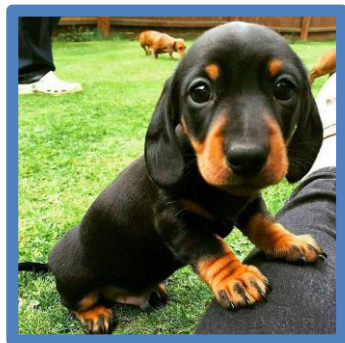
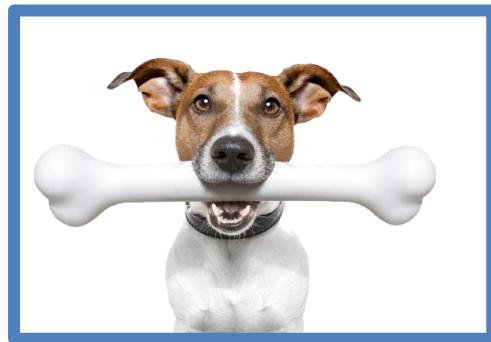
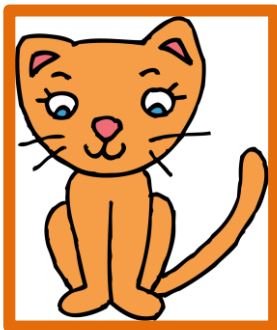
$$\theta = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

How can we find
the estimate of
theta?

Linear Regression

- Maximum Likelihood Estimate (MLE) corresponds to the Least Squares Estimate (given the assumptions)
- Introduced the concepts of loss function and optimization to obtain the best model for regression

Image Classification

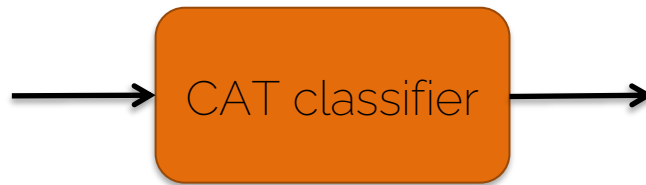


Regression vs Classification

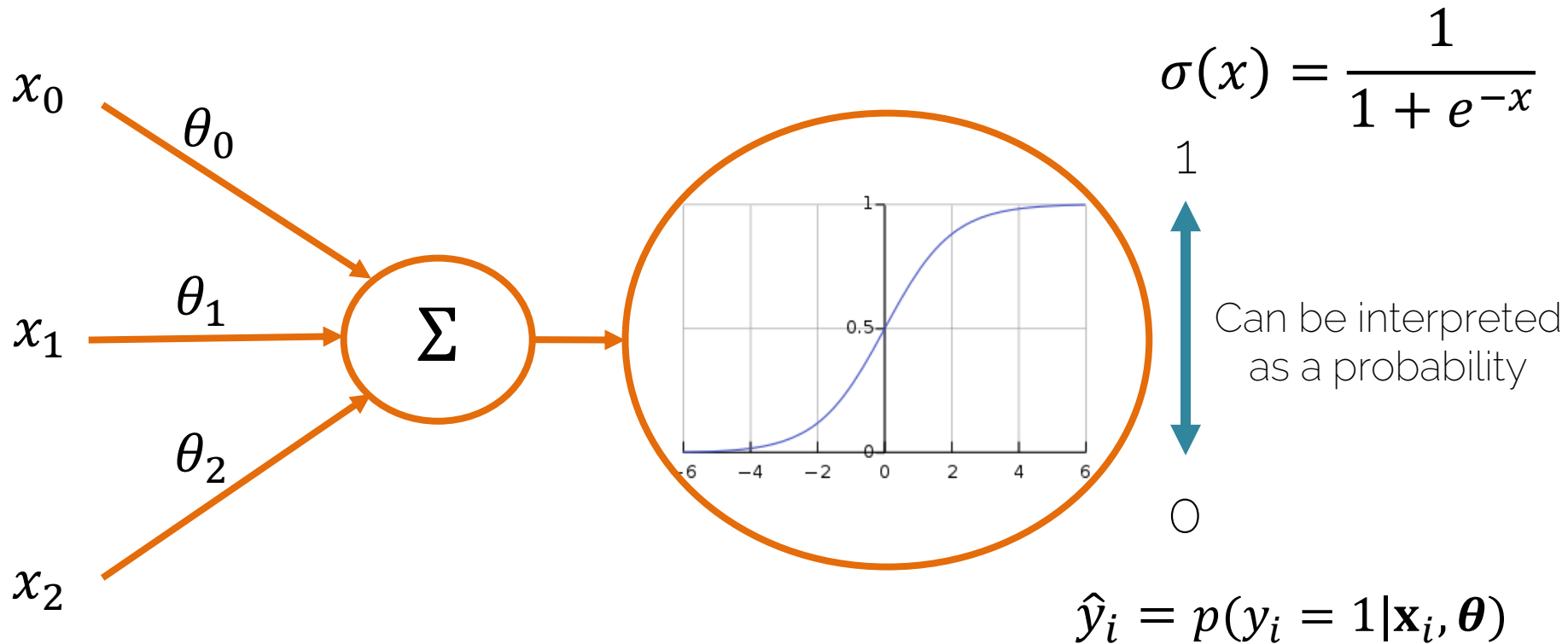
- Regression: predict a continuous output value (e.g., temperature of a room)
- Classification: predict a discrete value
 - Binary classification: output is either 0 or 1
 - Multi-class classification: set of N classes



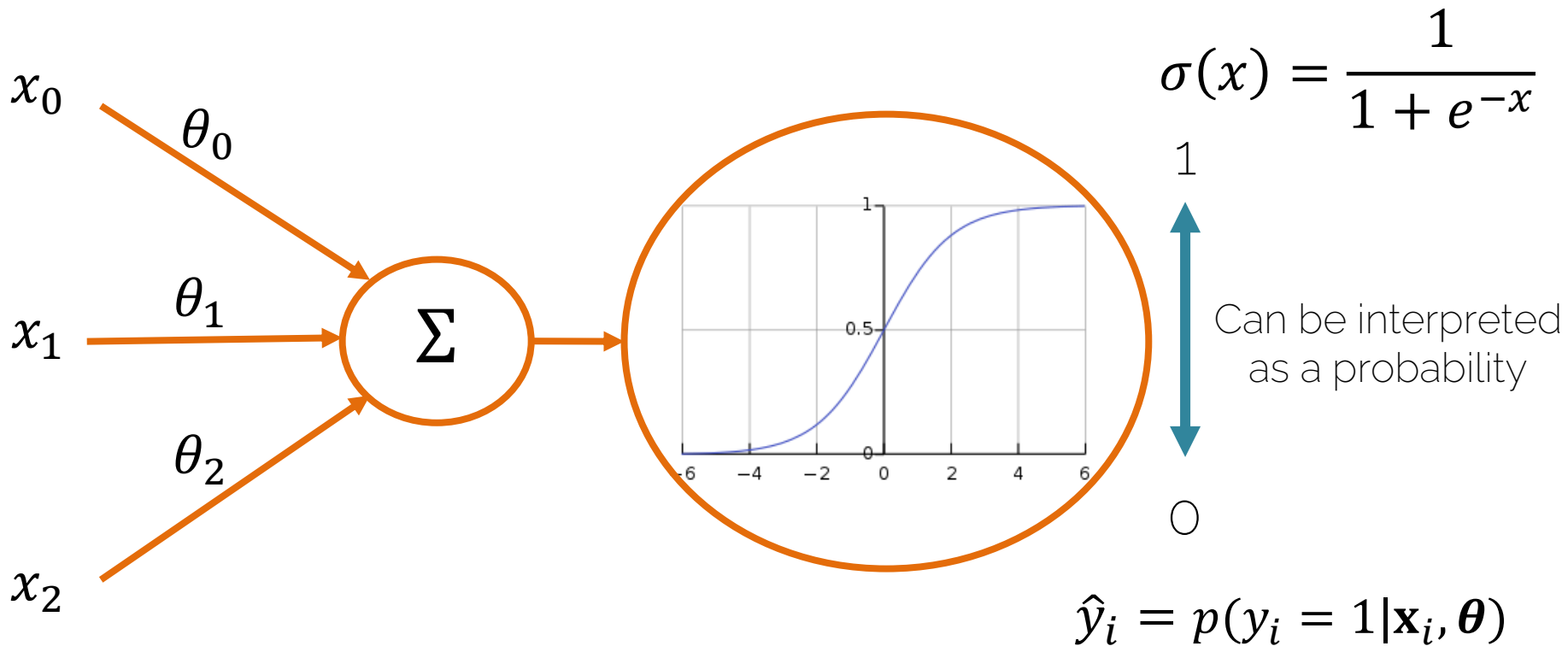
Logistic Regression



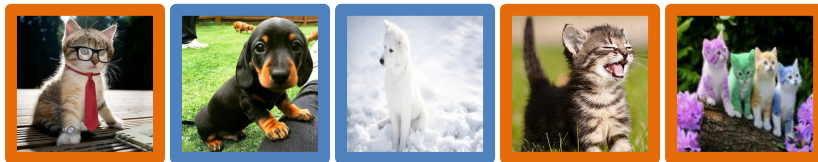
Sigmoid for Binary Predictions



Spoiler Alert: 1-Layer Neural Network



Logistic Regression: Max. Likelihood



- Probability of a binary output

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

- Maximum Likelihood Estimate

$$\hat{y}_i = p(y_i = 1 | \mathbf{x}_i, \boldsymbol{\theta})$$

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

Logistic Regression: Loss Function

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

$$\begin{aligned} \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\theta}) &= \sum_{i=1}^n \log (\hat{y}_i^{y_i} (1 - \hat{y}_i)^{(1-y_i)}) \\ &= \sum_{i=1}^n y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \end{aligned}$$

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*)

Logistic Regression: Optimization

- Loss for each training sample:

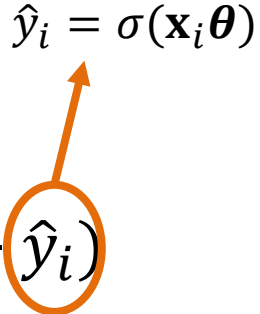
$$\mathcal{L}(\hat{y}_i, y_i) = -[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]$$

- Overall loss

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

Minimization



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


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

Logistic Regression: Optimization

- No closed-form solution
- Make use of an iterative method → gradient descent

Gradient descent –
later on!

Insights from the first lecture

- 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

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.