Deep Learning for Vision & Language

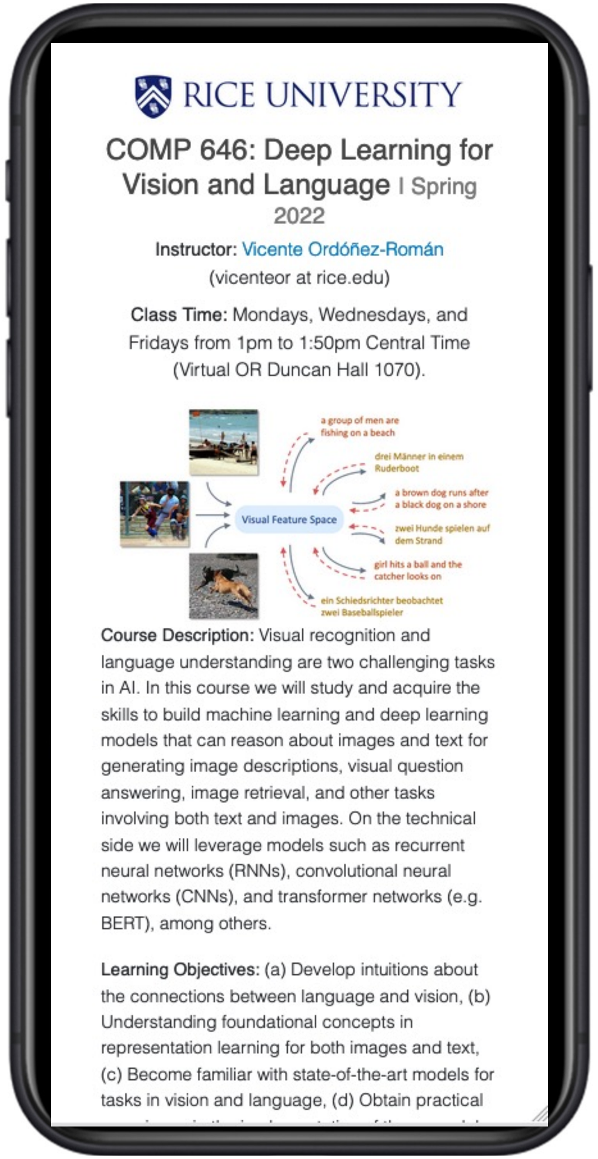
Machine Learning I & II: Supervised vs Unsupervised Learning
Linear Classifiers / Regressors

RICE UNIVERSITY

About the class

- COMP 646: Deep Learning for Vision and Language
- Instructor: **Vicente** Ordóñez (Vicente Ordóñez Román)
- Website: https://www.cs.rice.edu/~vo9/deep-vislang
- Location: Zoom – Rice Canvas has the links **OR**
  Duncan Hall 1070
- Times: Mondays, Wednesdays, and Fridays
  from 1pm to 1:50pm Central Time
- Office Hours: TBD
- Teaching Assistants: TBD
- Discussion Forum: Rice Canvas

https://www.cs.rice.edu/~vo9/deep-vislang/

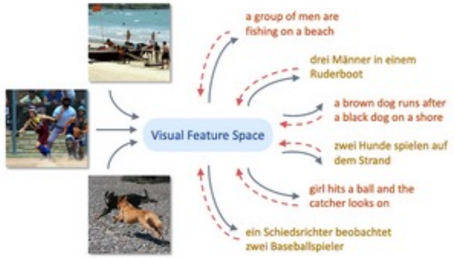


RICE UNIVERSITY

vislang

COMP 646: Deep Learning for Vision and Language | Spring 2022

Instructor: Vicente Ordóñez-Román (vicenteor at rice.edu)
Class Time: Mondays, Wednesdays, and Fridays from 1pm to 1:50pm Central Time (Virtual OR Duncan Hall 1070).



Course Description: Visual recognition and language understanding are two challenging tasks in AI. In this course we will study and acquire the skills to build machine learning and deep learning models that can reason about images and text for generating image descriptions, visual question answering, image retrieval, and other tasks involving both text and images. On the technical side we will leverage models such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and transformer networks (e.g. BERT), among others.

Learning Objectives: (a) Develop intuitions about the connections between language and vision, (b) Understanding foundational concepts in representation learning for both images and text, (c) Become familiar with state-of-the-art models for tasks in vision and language, (d) Obtain practical experience in the implementation of these models.

Prerrequisites: There are no formal pre-requisities for this class. However a basic command of machine learning, deep learning or computer vision will be useful when taking this class. Students should have knowledge of linear algebra, differential calculus, and basic statistics and probability. Moreover students are expected to have attained some level of proficiency in Python programming or be willing to learn Python programming. Students are encouraged to complete the following activity before the first lecture: [Primer on Image Processing].

Grading: Assignments: 30% (3 assignments), Class Project: 50%, Quiz: 10%, Class Participation: 10%.

Schedule

Date	Topic
Mon, Jan 10	Introduction to Vision and Language
Wed, Jan 12	Machine Learning I: Supervised vs Unsupervised Learning, Linear Classifiers
Fri, Jan 14	Machine Learning II: Stochastic Gradient Descent / Regularization
	Assignment on Text and Image Classification
Mon, Jan 17	Martin Luther King, Jr. Day (Holiday - No Scheduled Classes)
Wed, Jan 19	Neural Networks I: Multi-layer Perceptrons and Backpropagation
Fri, Jan 21	Practical Session: Neural Networks Building Blocks

Machine Learning

The study of algorithms that learn from data.

Linear Regression

Example: Hollywood movie data

input variables x					output variables y	
production costs	**promotional costs**	**genre of the movie**	**box office first week**	**total book sales**	**total revenue USA**	**total revenue international**
$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$x_4^{(1)}$	$x_5^{(1)}$	$x_6^{(1)}$	$x_7^{(1)}$
$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$x_4^{(2)}$	$x_5^{(2)}$	$x_6^{(2)}$	$x_7^{(2)}$
$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$x_4^{(3)}$	$x_5^{(3)}$	$x_6^{(3)}$	$x_7^{(3)}$
$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$x_4^{(4)}$	$x_5^{(4)}$	$x_6^{(4)}$	$x_7^{(4)}$
$x_1^{(5)}$	$x_2^{(5)}$	$x_3^{(5)}$	$x_4^{(5)}$	$x_5^{(5)}$	$x_6^{(5)}$	$x_7^{(5)}$

Linear Regression

Example: Hollywood movie data

input variables x					output variables y	
production costs	**promotional costs**	**genre of the movie**	**box office first week**	**total book sales**	**total revenue USA**	**total revenue international**
$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$x_4^{(1)}$	$x_5^{(1)}$	$y_1^{(1)}$	$y_2^{(1)}$
$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$x_4^{(2)}$	$x_5^{(2)}$	$y_1^{(2)}$	$y_2^{(2)}$
$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$x_4^{(3)}$	$x_5^{(3)}$	$y_1^{(3)}$	$y_2^{(3)}$
$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$x_4^{(4)}$	$x_5^{(4)}$	$y_1^{(4)}$	$y_2^{(4)}$
$x_1^{(5)}$	$x_2^{(5)}$	$x_3^{(5)}$	$x_4^{(5)}$	$x_5^{(5)}$	$y_1^{(5)}$	$y_2^{(5)}$

Linear Regression

Example: Hollywood movie data

input variables x | output variables y

	production costs	promotional costs	genre of the movie	box office first week	total book sales	total revenue USA	total revenue international
training data	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$x_4^{(1)}$	$x_5^{(1)}$	$y_1^{(1)}$	$y_2^{(1)}$
	$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$x_4^{(2)}$	$x_5^{(2)}$	$y_1^{(2)}$	$y_2^{(2)}$
	$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$x_4^{(3)}$	$x_5^{(3)}$	$y_1^{(3)}$	$y_2^{(3)}$
test data	$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$x_4^{(4)}$	$x_5^{(4)}$	$y_1^{(4)}$	$y_2^{(4)}$
	$x_1^{(5)}$	$x_2^{(5)}$	$x_3^{(5)}$	$x_4^{(5)}$	$x_5^{(5)}$	$y_1^{(5)}$	$y_2^{(5)}$

Linear Regression

$$\hat{y} = \sum_i w_i x_i$$

$$\hat{y} = W^T x$$

Prediction,
Inference,
Testing

$$D = \{(x^{(d)}, y^{(d)})\}$$

$$L(W) = \sum_{d=1}^{|D|} l(\hat{y}^{(d)}, y^{(d)})$$

$$W^* = \operatorname{argmin} L(W)$$

Training,
Learning,
Parameter estimation
Objective minimization

Linear Regression

$$\hat{y}_j = \sum_i w_{ij} x_i$$

$$\hat{y} = W^T x$$

Prediction,
Inference,
Testing

$$D = \{(x^{(d)}, y^{(d)})\}$$

$$L(W) = \sum_{d=1}^{|D|} \sum_j l(\hat{y}_j{}^{(d)}, y_j{}^{(d)})$$

$$W^* = \operatorname{argmin} L(W)$$

Training,
Learning,
Parameter
estimation
Objective
minimization

Linear Regression – Least Squares

$$\hat{y}_j = \sum_i w_{ji} x_i$$

$$\hat{y} = W^T x$$

$$D = \{(x^{(d)}, y^{(d)})\}$$

$$\boxed{L(W) = \sum_{d=1}^{|D|} \sum_j \left(\hat{y}_j{}^{(d)} - y_j{}^{(d)}\right)^2}$$

$$W^* = \operatorname{argmin} L(W)$$

Training,
Learning,
Parameter
estimation
Objective
minimization

Linear Regression – Least Squares

$$L(W) = \sum_{d=1}^{|D|} \sum_{j} \left(\hat{y}_j{}^{(d)} - y^{(d)} \right)^2$$

$$\hat{y}_j^{(d)} = \sum_i w_{ji} x_i^{(d)}$$

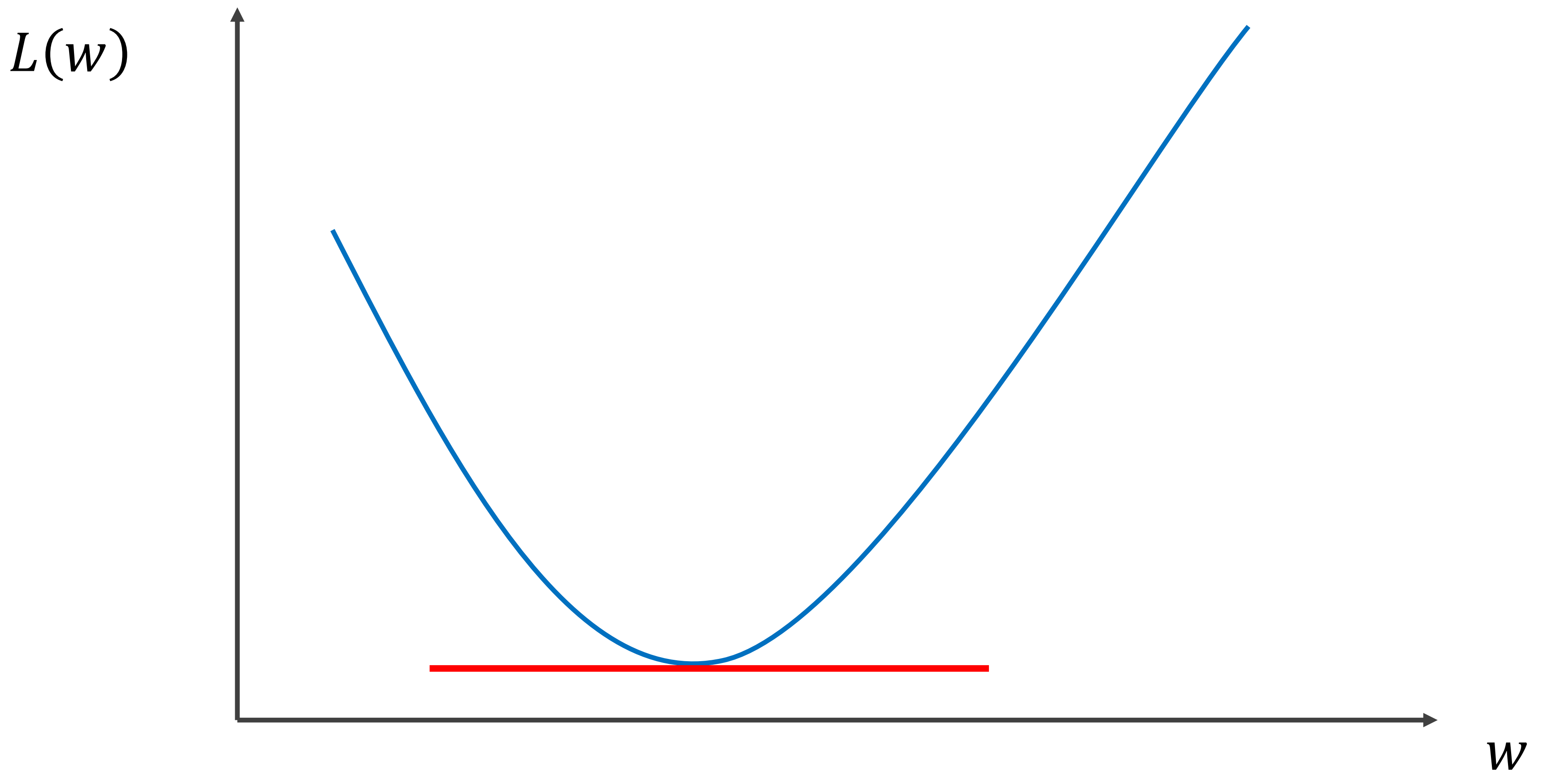
$$L(W) = \sum_{d=1}^{|D|} \sum_{j} \left(\sum_i w_{ji} x_i^{(d)} - y^{(d)} \right)^2$$

Linear Regression – Least Squares

$$L(W) = \sum_{d=1}^{|D|} \sum_{j} \left(\sum_{i} w_{ji} x_i^{(d)} - y^{(d)} \right)^2$$

$$W^* = \operatorname{argmin} L(W)$$

How to find the minimum of a function L(W)?



$$\frac{\partial L(w)}{\partial w} = 0$$

Linear Regression – Least Squares

$$\frac{\partial L(W)}{\partial w_{ji}} = \frac{\partial}{\partial w_{ji}} \left(\sum_{d=1}^{|D|} \sum_j \left(\sum_i w_{ji} x_i^{(d)} - y^{(d)}\right)^2\right)$$

$$\frac{\partial L(W)}{\partial w_{ji}} = 0$$

…

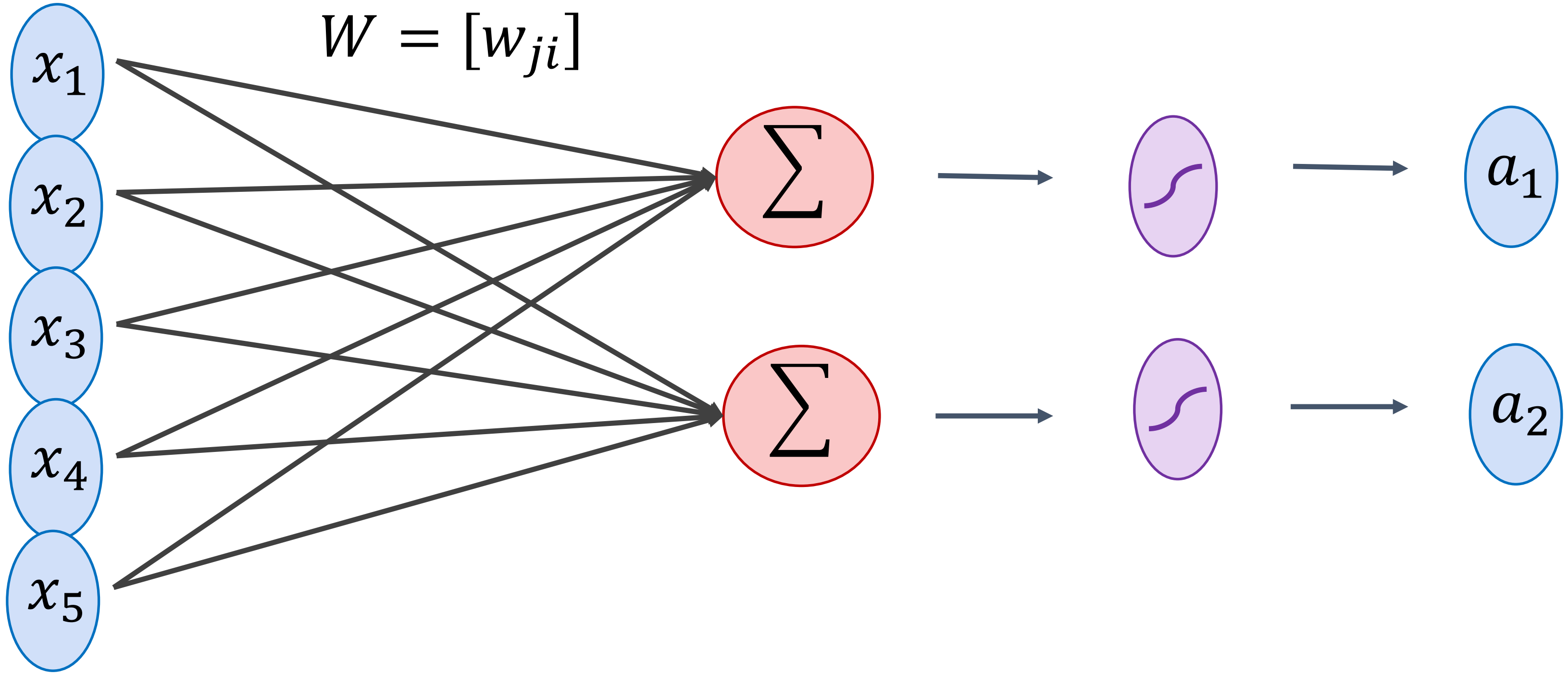
$$W = (X^T X)^{-1} X^T Y$$

ML Classifier / Regression models

- K-nearest neighbors
- Linear classifier / Linear regression
- Naïve Bayes classifiers
- Decision Trees
- Random Forests
- Boosted Decision Trees
- Neural Networks

Neural Network with One Layer

$$sigmoid(z) = \frac{1}{1 + e^{-z}}$$



$$a_j = sigmoid(\sum_i w_{ji} x_i + b_j)$$

Neural Network with One Layer

$$L(W, b) = \sum_{d=1}^{|D|} \left(a^{(d)} - y^{(d)}\right)^2$$

$$a_j^{(d)} = sigmoid(\sum_i w_{ji} x_i^{(d)} + \boxed{b_j})$$

Bias parameters

$$L(W, b) = \sum_{j,d} \left(sigmoid(\sum_i w_{ji} x_i^{(d)} + b_j) - y_j^{(d)} \right)^2$$

Neural Network with One Layer

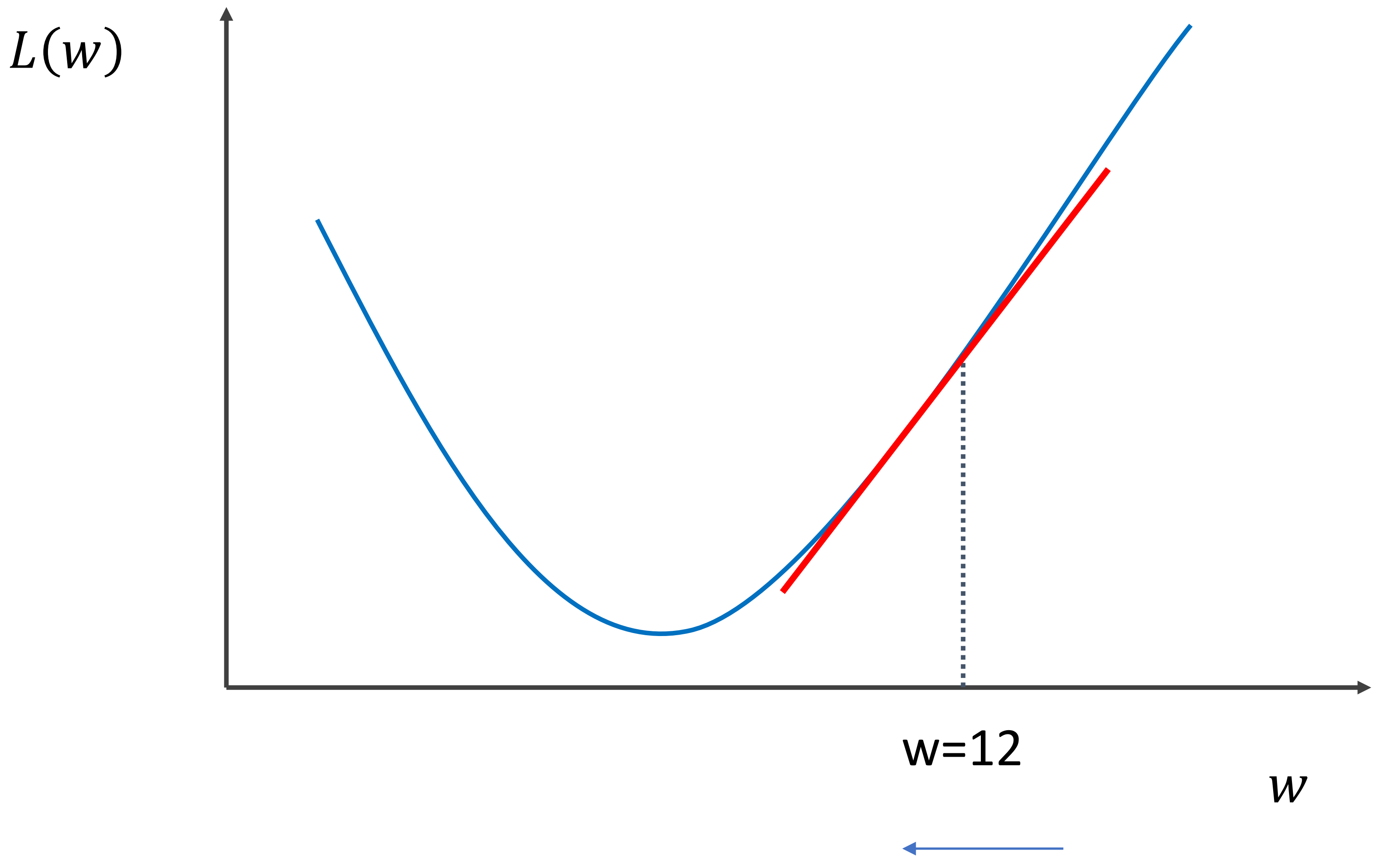
$$L(W, b) = \sum_{j,d} \left(sigmoid(\sum_i w_{ji} x_i^{(d)} + b_j) - \ y_j^{(d)} \right)^2$$

$$\frac{\partial L}{\partial w_{ji}} = 0$$

(1) We can compute this derivative but often there will be no closed-form solution for W when dL/dw = 0

(2) Also, even for linear regression where the solution was $W = (X^T X)^{-1} X^T Y$, computing this expression might be expensive or infeasible. e. g. think of computing $(X^T X)^{-1}$ for a very large dataset with a million x_i

Gradient Descent



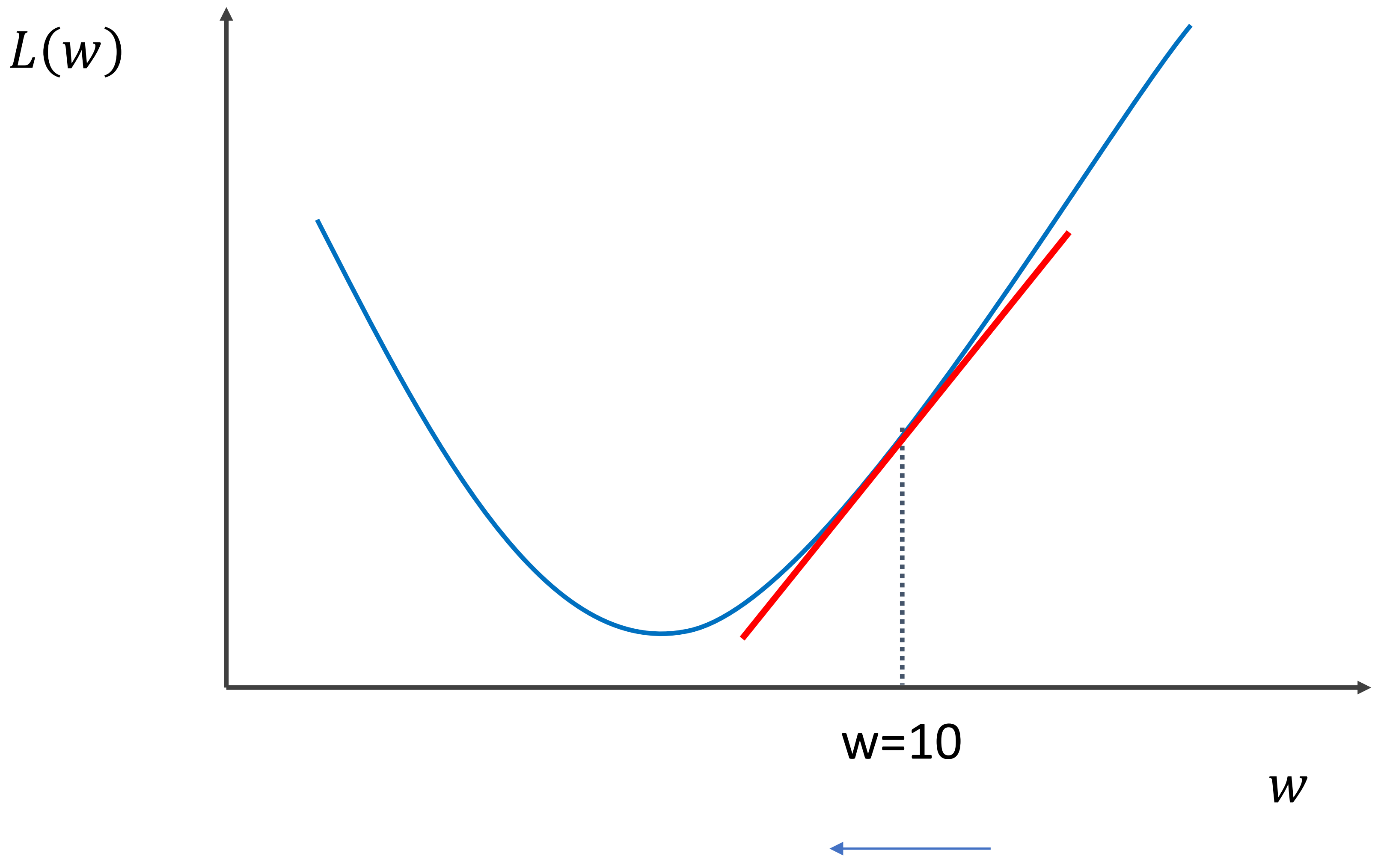
1. Start with a random value of w (e.g. w = 12)

2. Compute the gradient (derivative) of L(w) at point w = 12. (e.g. dL/dw = 6)

3. Recompute w as:

w = w – lambda * (dL / dw)

Gradient Descent

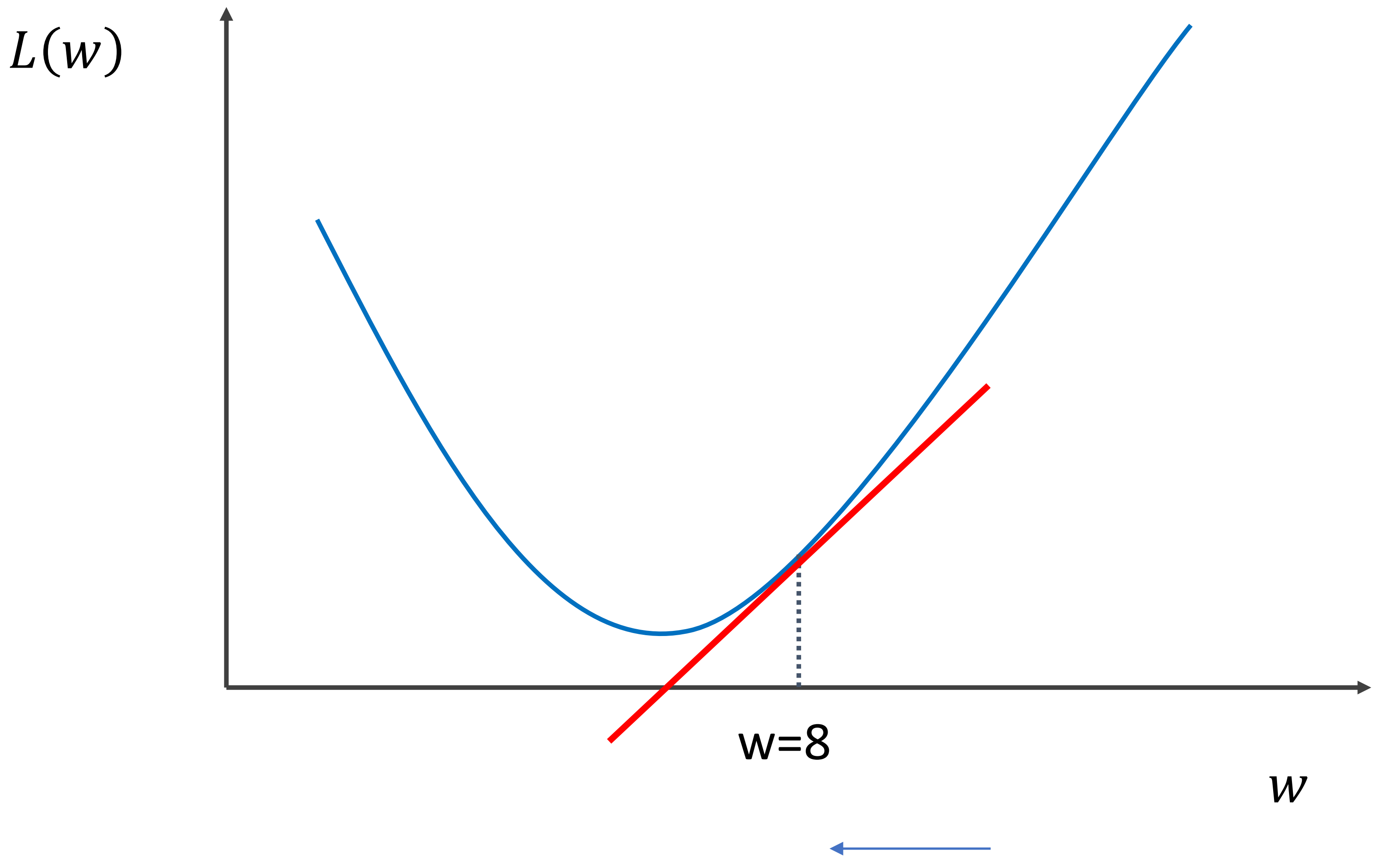


2. Compute the gradient (derivative) of L(w) at point w = 12. (e.g. dL/dw = 6)

3. Recompute w as:

w = w – lambda * (dL / dw)

Gradient Descent



2. Compute the gradient (derivative) of L(w) at point w = 12. (e.g. dL/dw = 6)

3. Recompute w as:

w = w – lambda * (dL / dw)

Gradient Descent

expensive

$$L(w,b) = \sum_{i=1}^{n} l(w,b)$$

$\lambda = 0.01$

Initialize w and b randomly

for e = 0, num_epochs **do**

Compute: $dL(w,b)/dw$ and $dL(w,b)/db$

Update w: $w = w \; - \lambda \, dL(w,b)/dw$

Update b: $b = b \; - \lambda \, dL(w,b)/db$

Print: $L(w,b)$ // Useful to see if this is becoming smaller or not.

end

Stochastic Gradient Descent (mini-batch)

$\lambda = 0.01$

$$L_B(w, b) = \sum_{i=1}^{B} l(w, b)$$

Initialize w and b randomly

for e = 0, num_epochs **do**

for b = 0, num_batches **do**

Compute: $dL_B(w, b)/dw$ and $dL_B(w, b)/db$

Update w: $w = w \ - \lambda \, dl(w, b)/dw$

Update b: $b = b \ - \lambda \, dl(w, b)/db$

Print: $L_B(w, b)$ // Useful to see if this is becoming smaller or not.

end

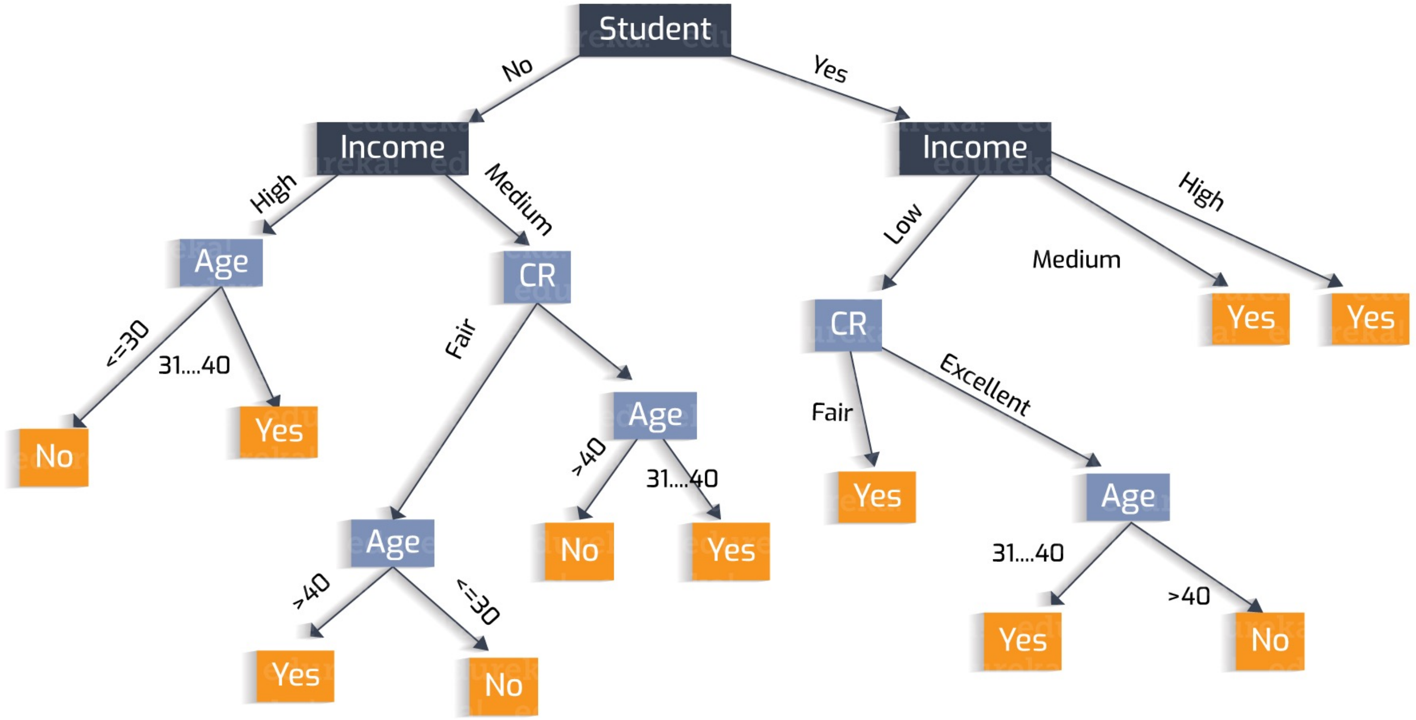
end

In this class we will mostly rely on…

- K-nearest neighbors
- Linear classifiers
- Naïve Bayes classifiers
- Decision Trees
- Random Forests
- Boosted Decision Trees
- Neural Networks

Why?

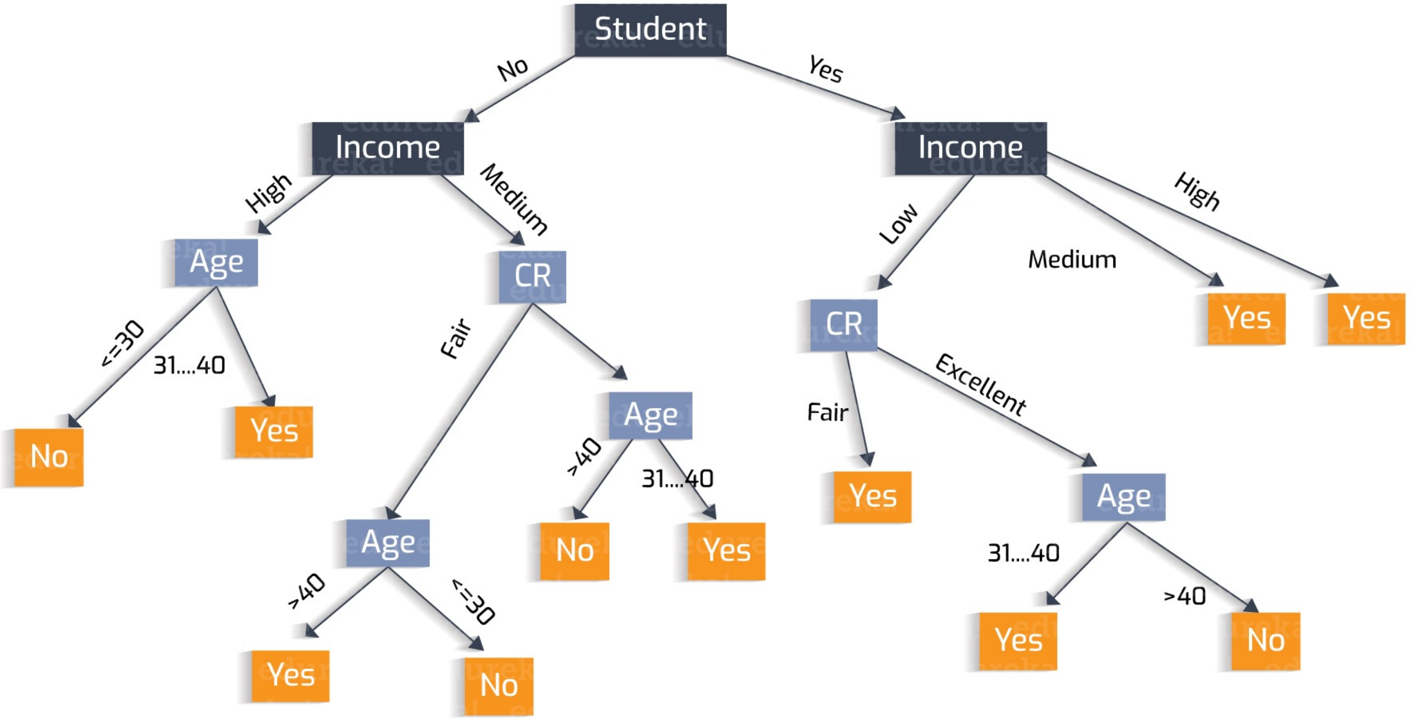
- Decisions Trees



https://heartbeat.fritz.ai/understanding-the-mathematics-behind-decision-trees-22d86d55906 by Nikita Sharma

Why?

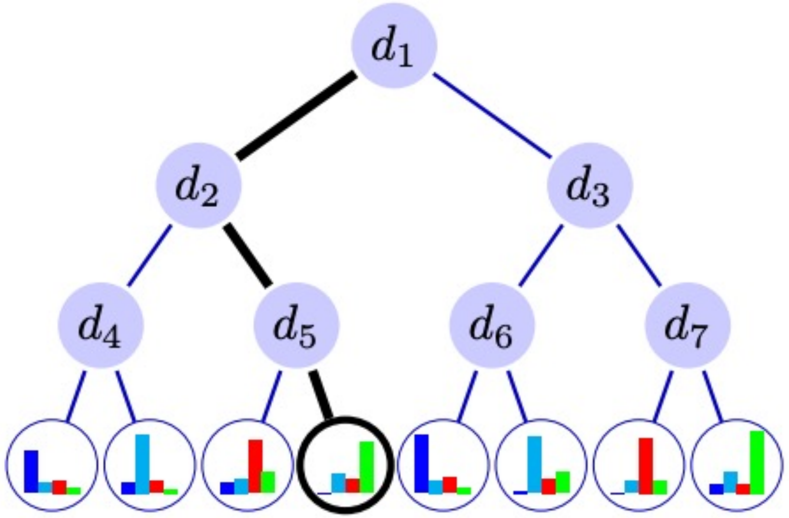
- Decisions Trees are great because they are often interpretable.
- However, they usually deal better with categorical data – not input pixel data.



https://heartbeat.fritz.ai/understanding-the-mathematics-behind-decision-trees-22d86d55906 by Nikita Sharma

That said

- There have been efforts to combine neural networks and decision trees, where pixels first go through a neural network and then a decision tree structure.



Deep Neural Decision Forests

Peter Kontschieder[1] Madalina Fiterau[*,2] Antonio Criminisi[1] Samuel Rota Bulò[1,3]

Microsoft Research[1]
Cambridge, UK

Carnegie Mellon University[2]
Pittsburgh, PA

Fondazione Bruno Kessler[3]
Trento, Italy

Review

- Image Classification Assignment from the Deep Learning for Visual Recognition class

- NOTE: This is not an assignment for this class. Do at your own pace, no need to hand out anything. You can always ask us questions about it during office hours.

Regression vs Classification

Regression
- Labels are continuous variables – e.g. distance.
- Losses: Distance-based losses, e.g. sum of distances to true values.
- Evaluation: Mean distances, correlation coefficients, etc.

Classification
- Labels are discrete variables (1 out of K categories)
- Losses: Cross-entropy loss, margin losses, logistic regression (binary cross entropy)
- Evaluation: Classification accuracy, etc.

Stochastic Gradient Descent

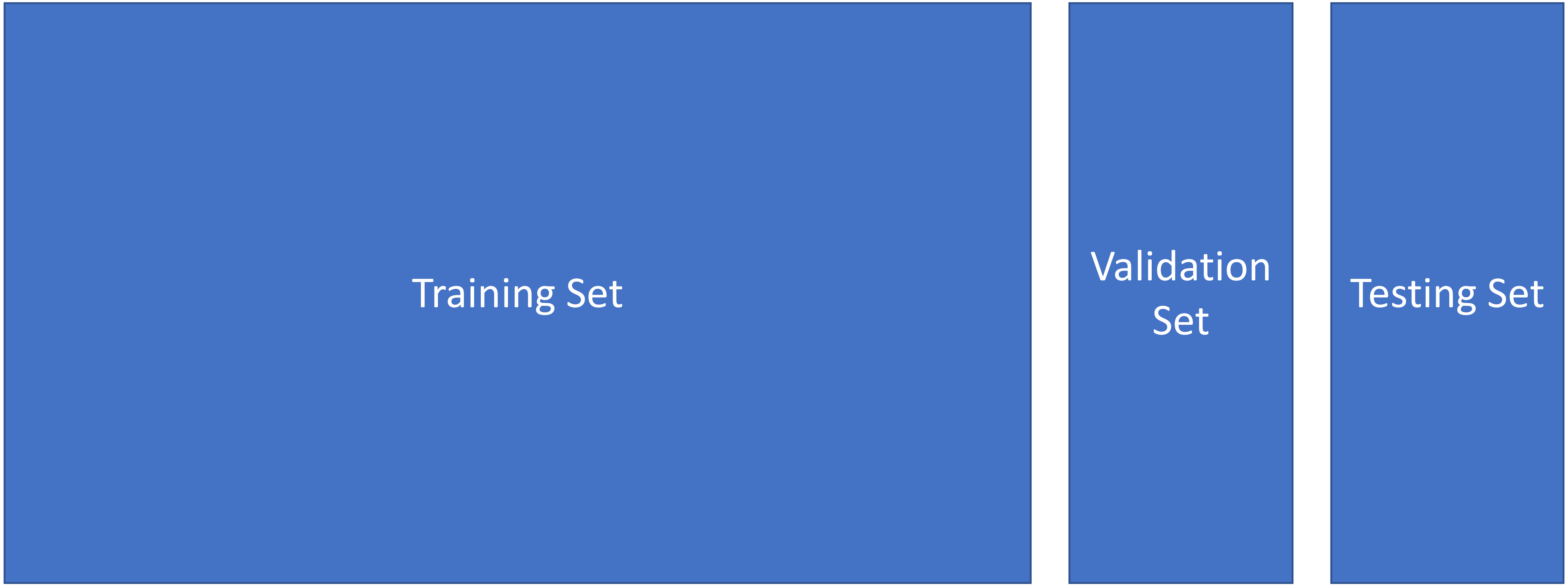
- How to choose the right batch size B?
- How to choose the right learning rate lambda?
- How to choose the right loss function, e.g. is least squares good enough?
- How to choose the right function/classifier, e.g. linear, quadratic, neural network with 1 layer, 2 layers, etc?

Linear Regression

Example: Hollywood movie data

	input variables x					output variables y	
	production costs	promotional costs	genre of the movie	box office first week	total book sales	total revenue USA	total revenue international
training data	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$x_4^{(1)}$	$x_5^{(1)}$	$y_1^{(1)}$	$y_2^{(1)}$
	$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$x_4^{(2)}$	$x_5^{(2)}$	$y_1^{(2)}$	$y_2^{(2)}$
	$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$x_4^{(3)}$	$x_5^{(3)}$	$y_1^{(3)}$	$y_2^{(3)}$
test data	$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$x_4^{(4)}$	$x_5^{(4)}$	$y_1^{(4)}$	$y_2^{(4)}$
	$x_1^{(5)}$	$x_2^{(5)}$	$x_3^{(5)}$	$x_4^{(5)}$	$x_5^{(5)}$	$y_1^{(5)}$	$y_2^{(5)}$

Training, Validation (Dev), Test Sets



Training, Validation (Dev), Test Sets

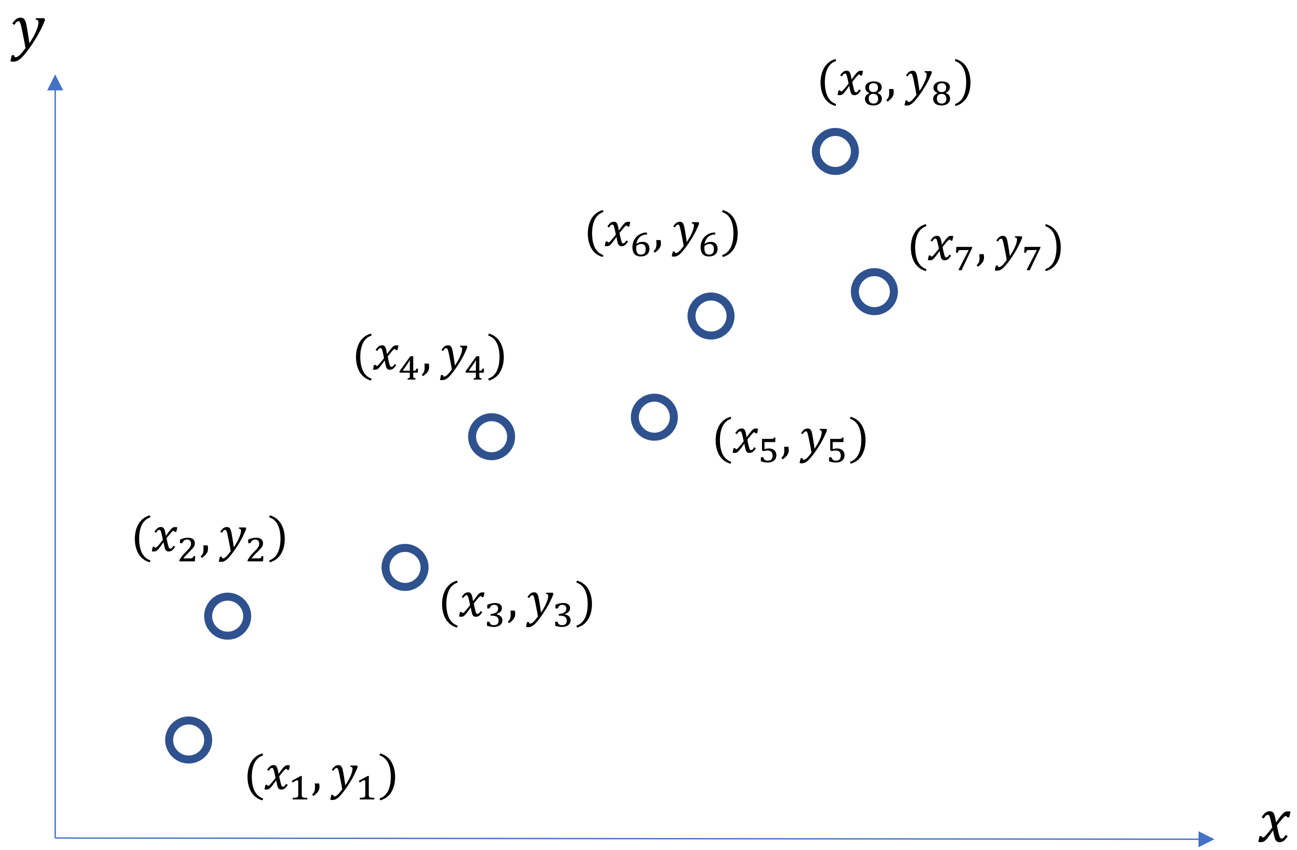
Used during development

Training, Validation (Dev), Test Sets

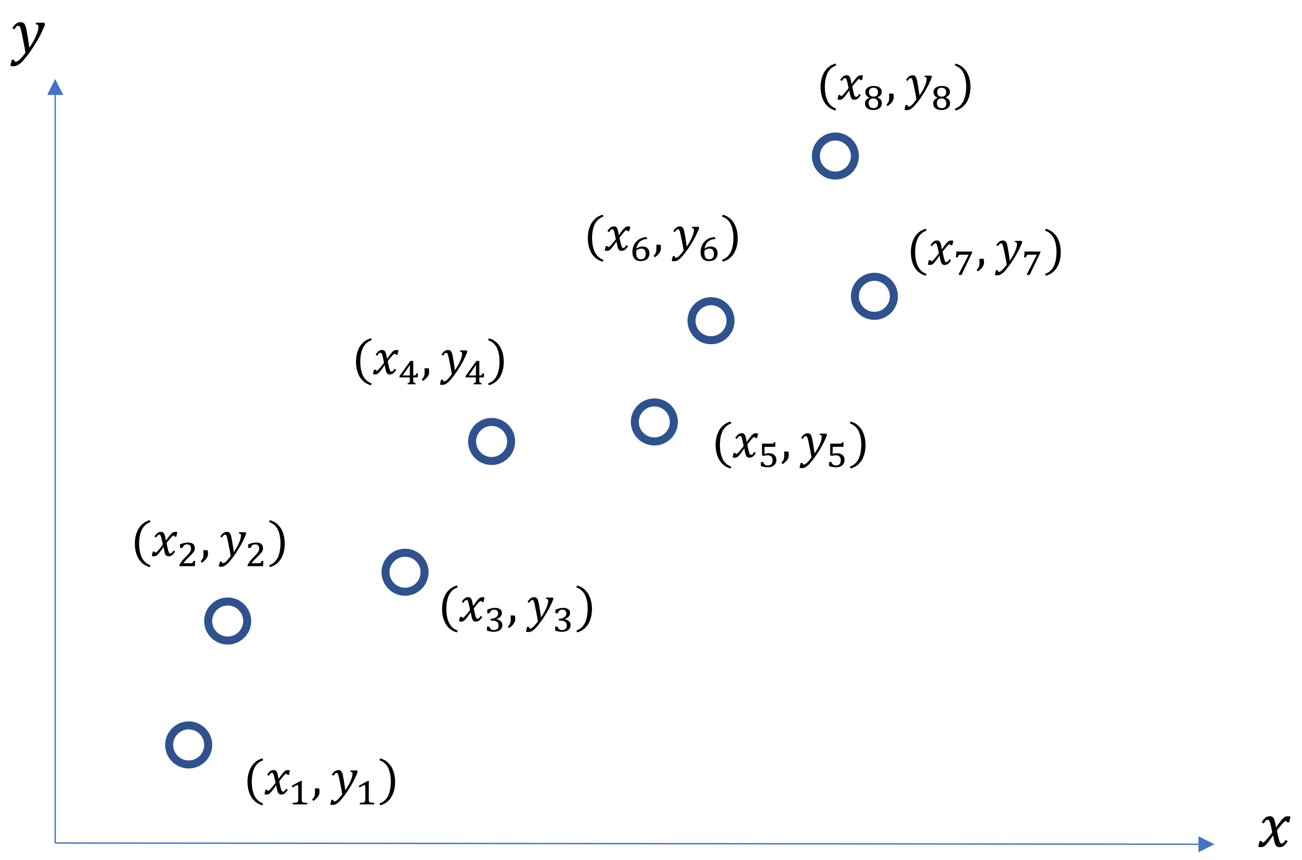
Only to be used for evaluating the model at the very end of development and any changes to the model after running it on the test set, could be influenced by what you saw happened on the test set, which would invalidate any future evaluation.

How to pick the right model?

Linear Regression – 1 output, 1 input

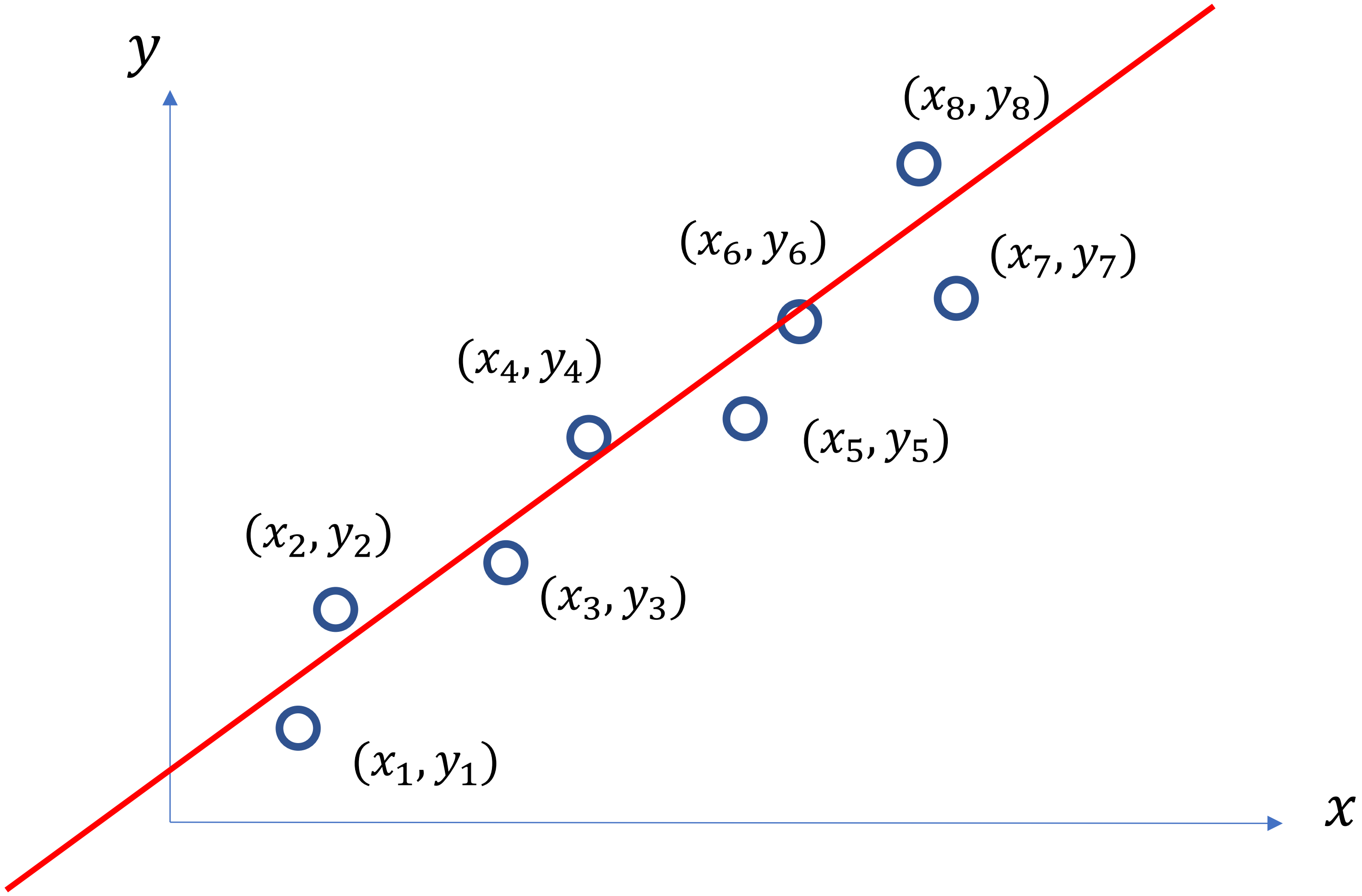


Linear Regression – 1 output, 1 input



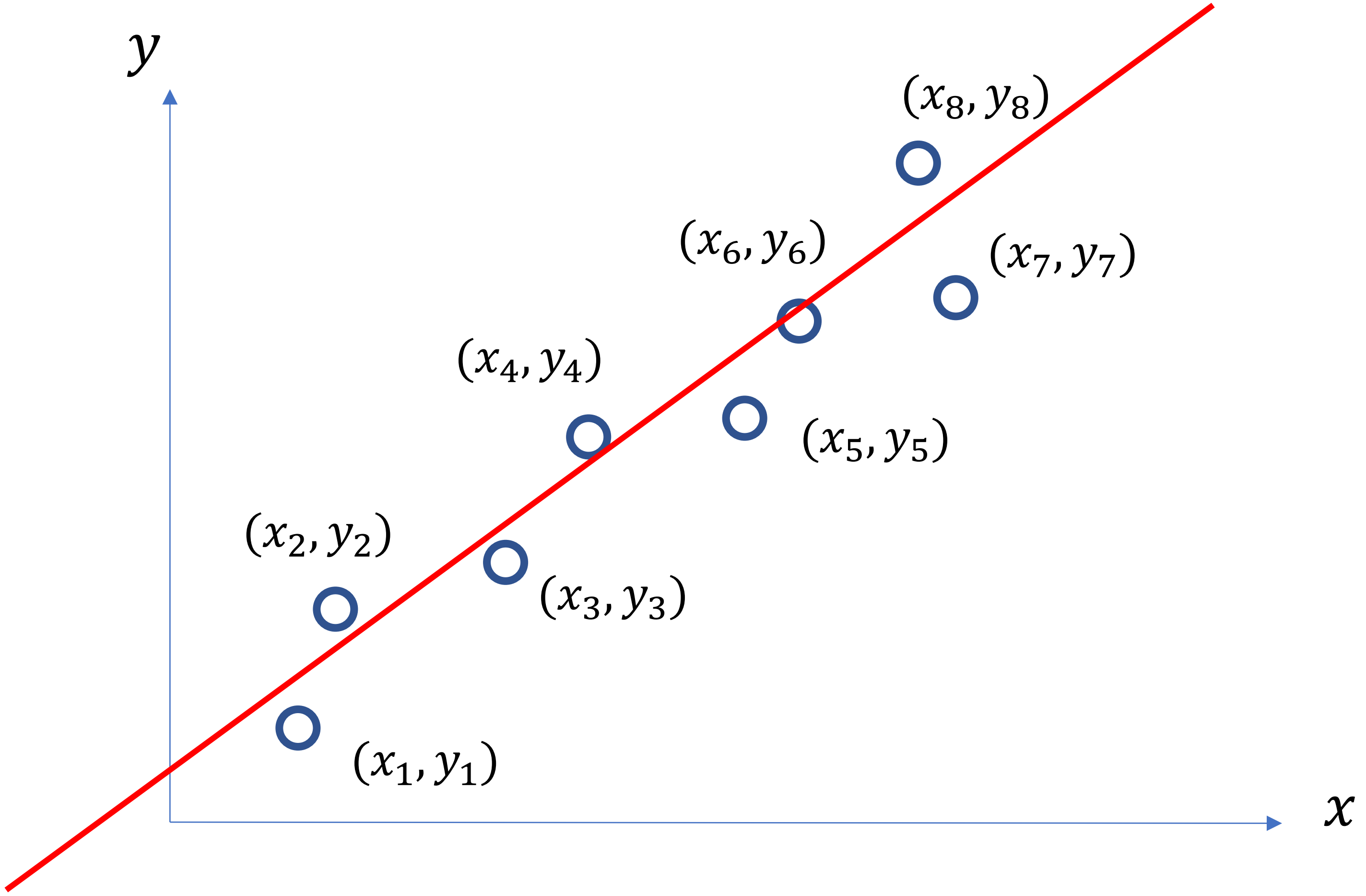
Model: $\hat{y} = wx + b$

Linear Regression – 1 output, 1 input



Model: $\hat{y} = wx + b$

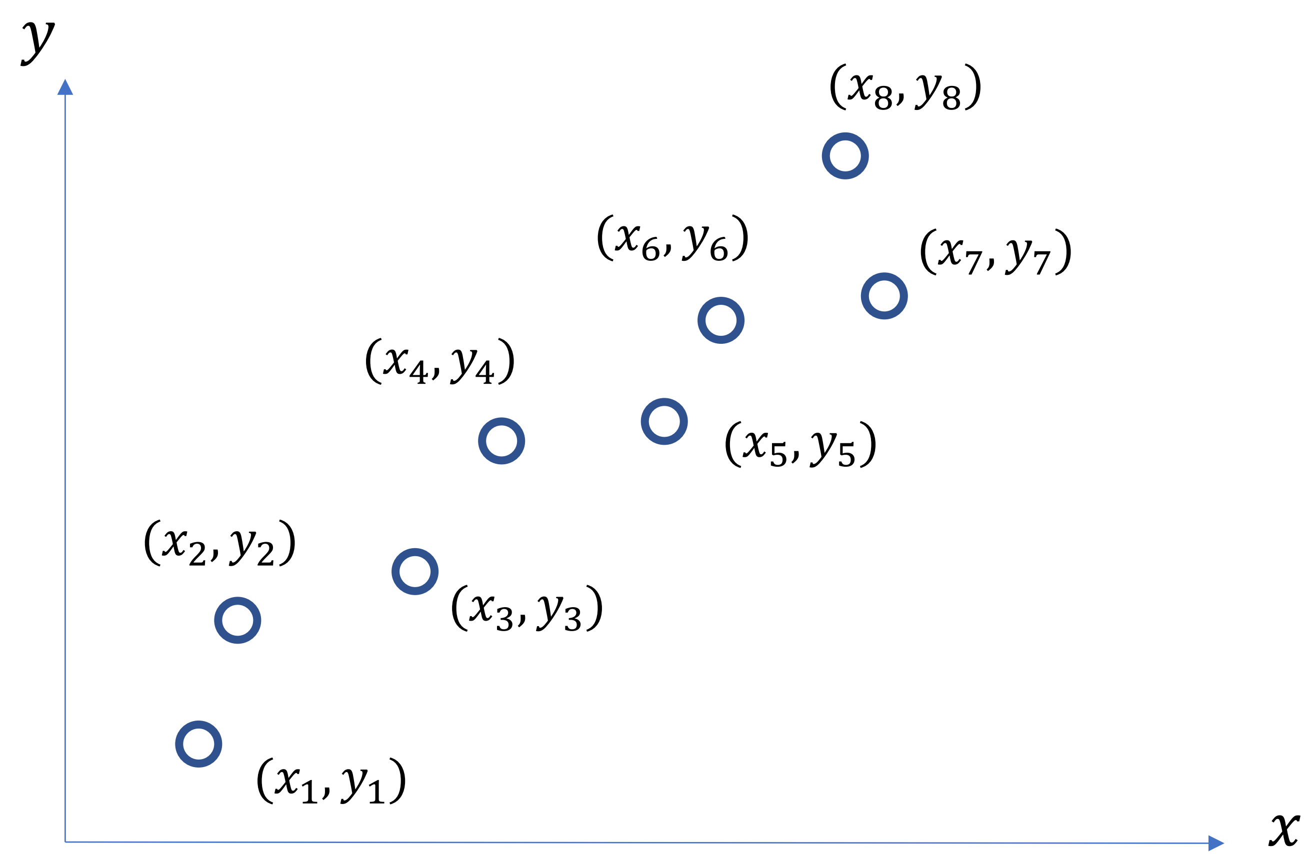
Linear Regression – 1 output, 1 input



Model: $\hat{y} = wx + b$

Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

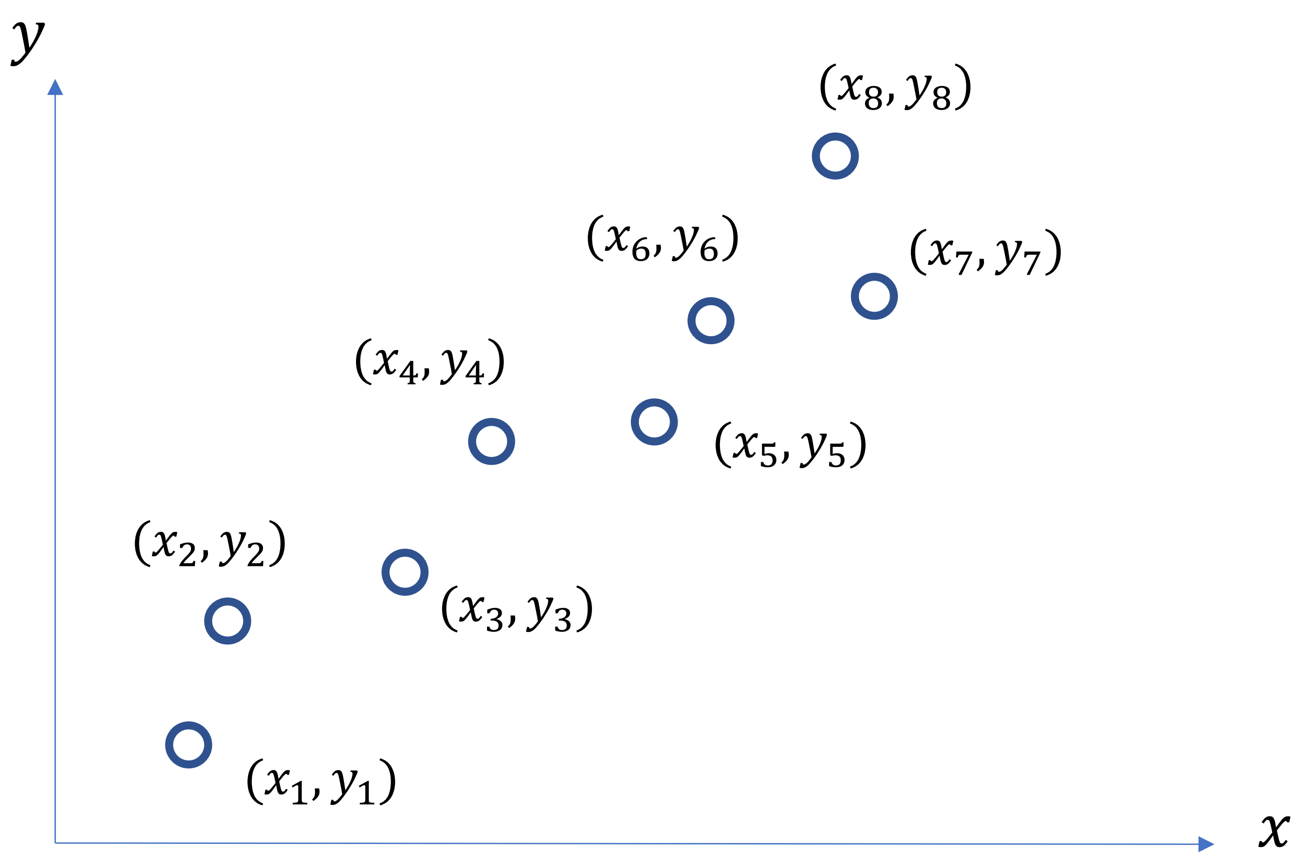
Quadratic Regression



Model: $\hat{y} = w_1 x^2 + w_2 x + b$

Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

n-polynomial Regression

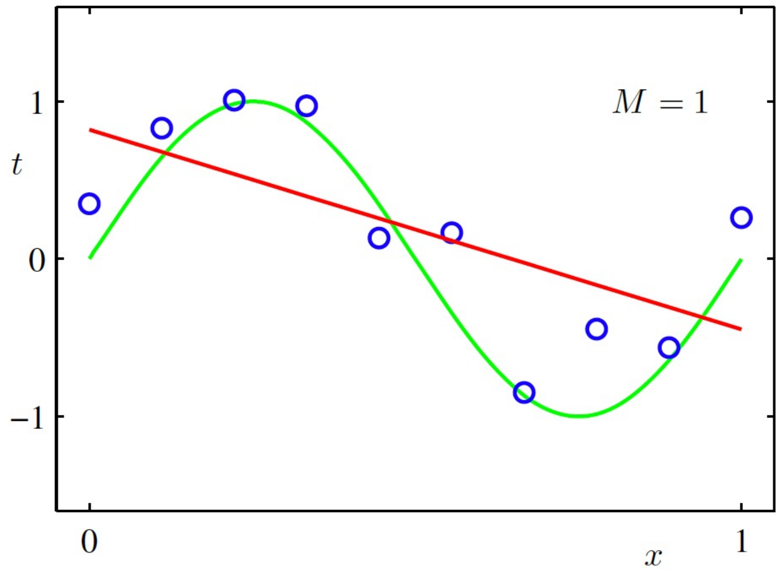


Model: $\hat{y} = w_n x^n + \cdots + w_1 x + b$

Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

Overfitting

f is linear

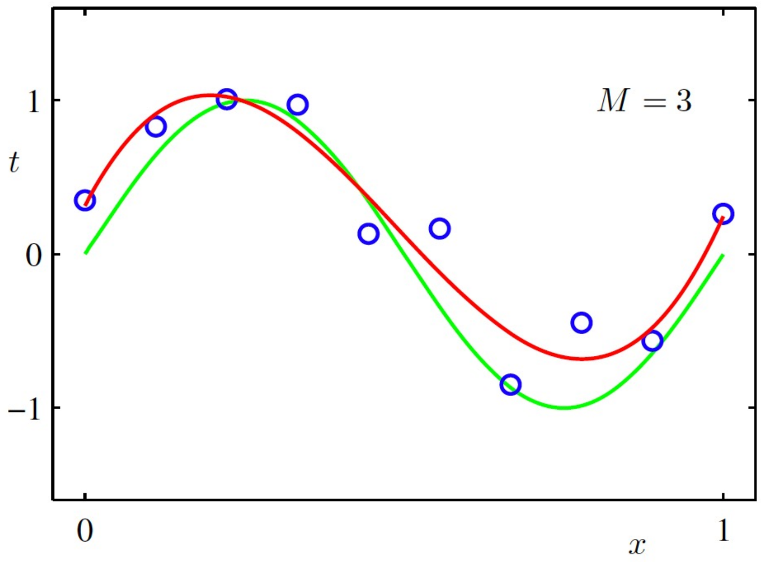


$Loss(w)$ is high

Underfitting

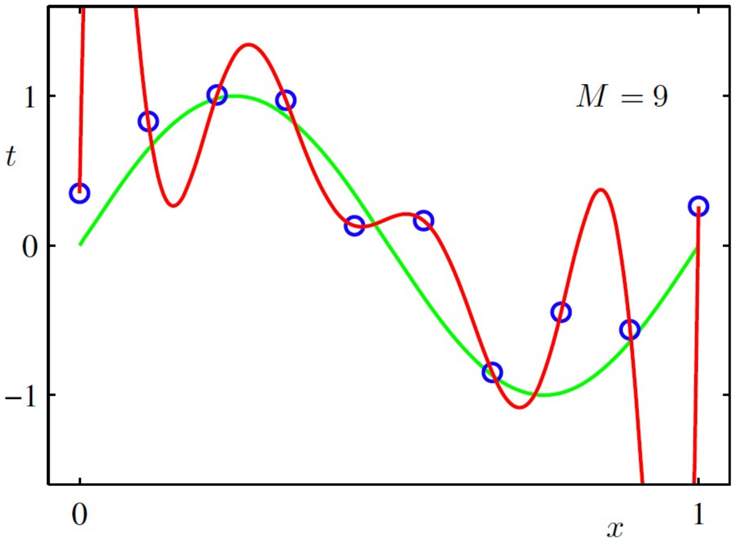
High Bias

f is cubic



$Loss(w)$ is low

f is a polynomial of degree 9



$Loss(w)$ is zero!

Overfitting

High Variance

Christopher M. Bishop – Pattern Recognition and Machine Learning

Supervised Learning - Classification

Training Data

cat

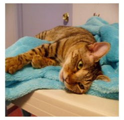
dog

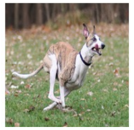
cat

.
.
.

bear

Test Data





.
.
.

Supervised Learning - Classification

Training Data

$x_1 = [$  $]$ $y_1 = [$cat$]$

$x_2 = [$  $]$ $y_2 = [$dog$]$

$x_3 = [$  $]$ $y_3 = [$cat$]$

.
.
.

$x_n = [$  $]$ $y_n = [$bear$]$

Supervised Learning - Classification

Training Data

inputs

targets / labels / ground truth

predictions

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $\quad y_1 = 1 \quad \hat{y}_1 = 1$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $\quad y_2 = 2 \quad \hat{y}_2 = 2$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $\quad y_3 = 1 \quad \hat{y}_3 = 2$

.
.
.

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $\quad y_n = 3 \quad \hat{y}_n = 1$

We need to find a function that maps *x* and *y* for any of them.

$$\hat{y}_i = f(x_i; \theta)$$

How do we "learn" the parameters of this function?

We choose ones that makes the following quantity small:

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

Supervised Learning – Linear Softmax

Training Data

inputs

targets /
labels /
ground truth

$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$ $\quad y_1 = 1$

$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$ $\quad y_2 = 2$

$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$ $\quad y_3 = 1$

$\vdots$

$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$ $\quad y_n = 3$

Supervised Learning – Linear Softmax

Training Data

inputs	targets / labels / ground truth	predictions
$x_1 = [x_{11} \quad x_{12} \quad x_{13} \quad x_{14}]$	$y_1 = [1 \quad 0 \quad 0]$	$\hat{y}_1 = [0.85 \quad 0.10 \quad 0.05]$
$x_2 = [x_{21} \quad x_{22} \quad x_{23} \quad x_{24}]$	$y_2 = [0 \quad 1 \quad 0]$	$\hat{y}_2 = [0.20 \quad 0.70 \quad 0.10]$
$x_3 = [x_{31} \quad x_{32} \quad x_{33} \quad x_{34}]$	$y_3 = [1 \quad 0 \quad 0]$	$\hat{y}_3 = [0.40 \quad 0.45 \quad 0.15]$
$\vdots$		
$x_n = [x_{n1} \quad x_{n2} \quad x_{n3} \quad x_{n4}]$	$y_n = [0 \quad 0 \quad 1]$	$\hat{y}_n = [0.40 \quad 0.25 \quad 0.35]$

Supervised Learning – Linear Softmax

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $y_i =$ [1 0 0] $\hat{y}_i = [f_c \quad f_d \quad f_b]$

$$g_c = w_{c1}x_{i1} + w_{c2}x_{i2} + w_{c3}x_{i3} + w_{c4}x_{i4} + b_c$$

$$g_d = w_{d1}x_{i1} + w_{d2}x_{i2} + w_{d3}x_{i3} + w_{d4}x_{i4} + b_d$$

$$g_b = w_{b1}x_{i1} + w_{b2}x_{i2} + w_{b3}x_{i3} + w_{b4}x_{i4} + b_b$$

$$f_c = e^{g_c}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_d = e^{g_d}/(e^{g_c}+e^{g_d} + e^{g_b})$$

$$f_b = e^{g_b}/(e^{g_c}+e^{g_d} + e^{g_b})$$

How do we find a good w and b?

$x_i = [x_{i1} \quad x_{i2} \quad x_{i3} \quad x_{i4}]$ $\quad y_i =$ [1 0 0] $\quad \hat{y}_i = [f_c(w,b) \quad f_d(w,b) \quad f_b(w,b)]$

We need to find w, and b that minimize the following:

$$L(w,b) = \sum_{i=1}^{n}\sum_{j=1}^{3} -y_{i,j}\log(\hat{y}_{i,j}) = \sum_{i=1}^{n} -\log(\hat{y}_{i,label}) = \sum_{i=1}^{n} -\log f_{i,label}(w,b)$$

Why?

Questions?