AI/Machine Learning Kick-off Concepts

Hisham Ihshaish
Birzeit University
May, 2026

Learning Concepts

Agenda

- ➡ A review on the **categories** of Learning
- ➡ Main Concepts
  - ◉ “Representation”
  - ◉ Conventions
  - ◉ Learning process

Types of ML

a review

- ➡ Human supervision
- ➡ Type of output
- ➡ Learning frequency
- ➡ Prediction Mechanism

Types of ML

- Human supervision

Whether or not they are trained with supervision - *supervised, unsupervised, semi, reinforcement.*

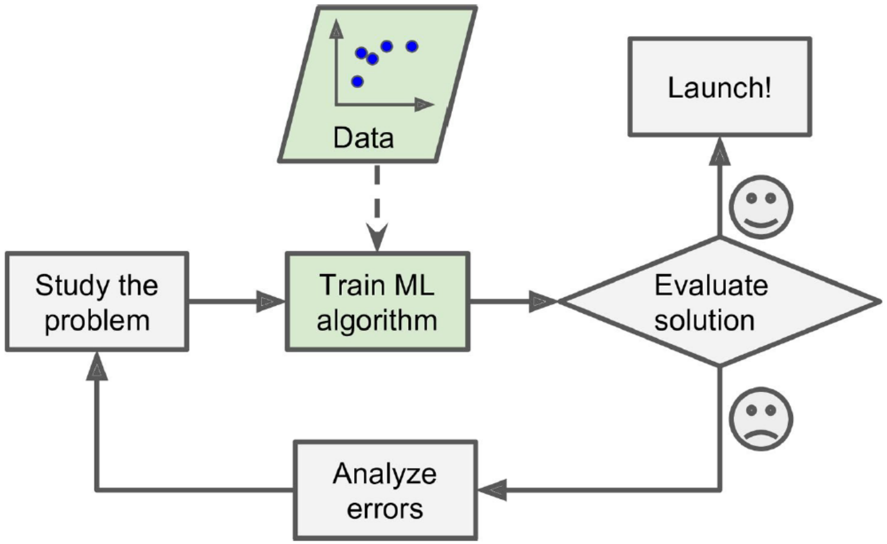
Types of ML

- Type of output

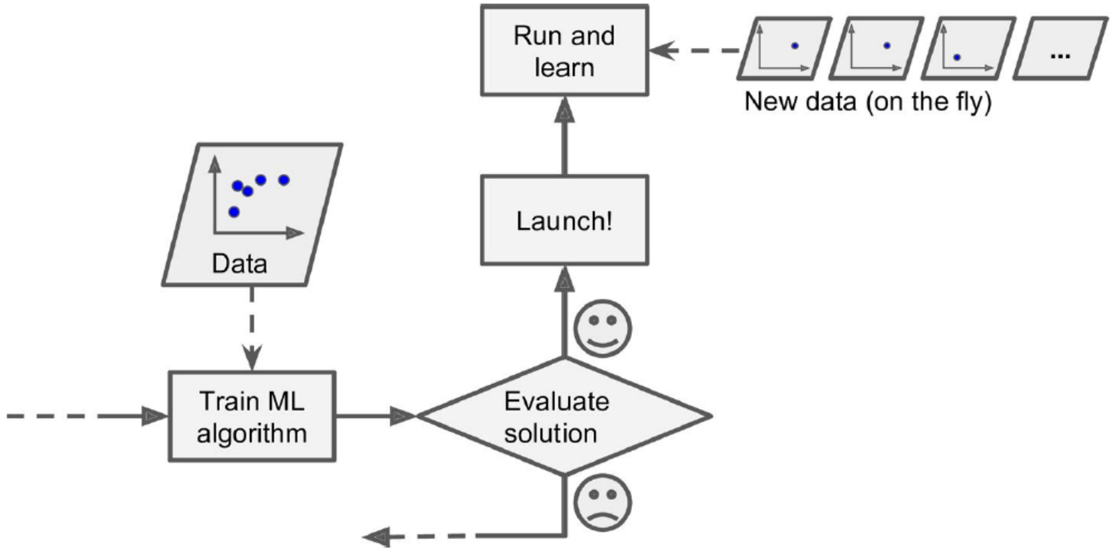
- Classification
- Regression
- Clustering
- Density Estimation
- Dimensionality Reduction

Batch vs Online

- Learning frequency



Batch



Online

Source: A Géron, Hands on ML…

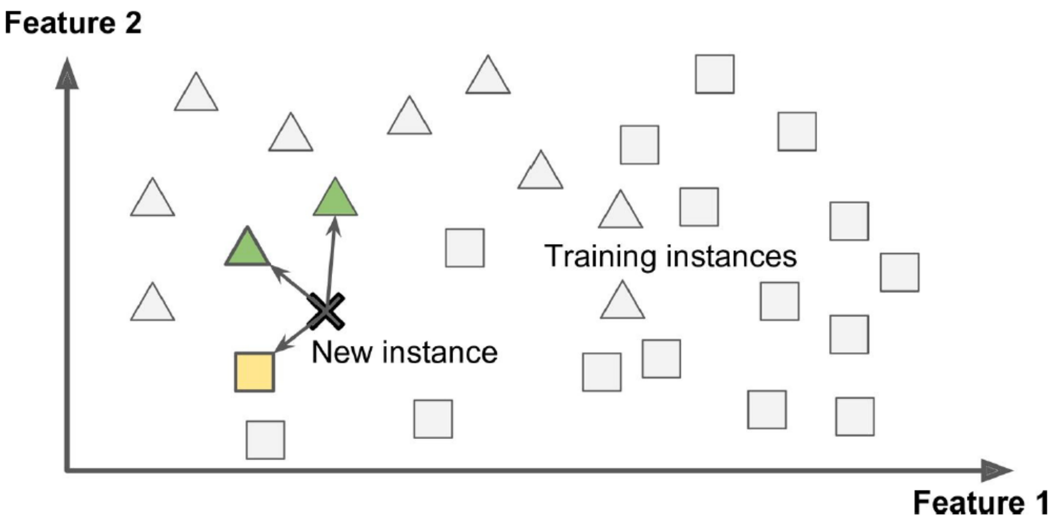
Types of ML

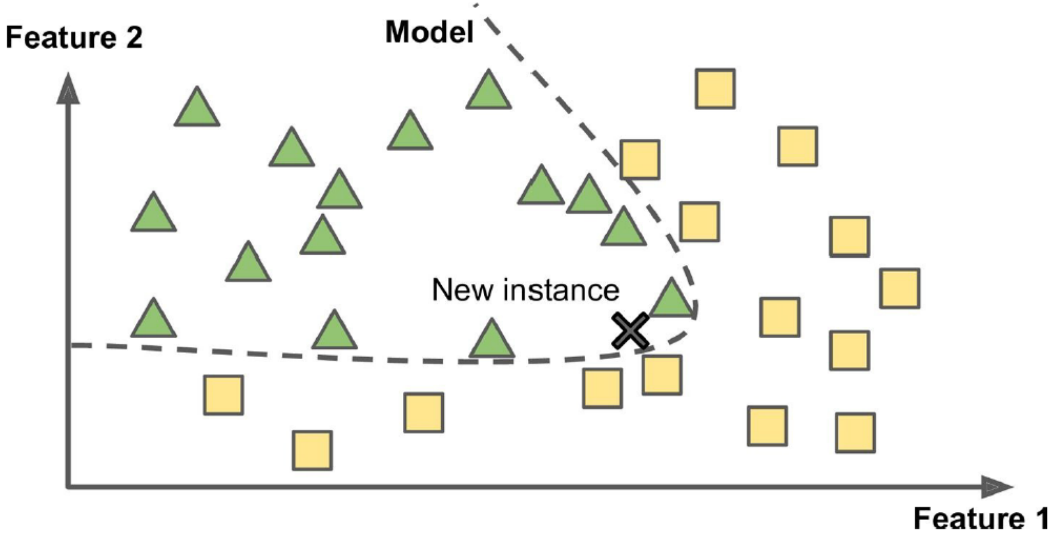
- Prediction Mechanism

Whether they work by simply comparing new data points to known data points, or instead *detect patterns* in the training data and *build a predictive model*, much like scientists do - **instance-based** versus **model-based** learning

Types of ML

- Prediction Mechanism





- KNNs
- Kernel Methods (eg, SVM)- with caution
- RBFs
- Graph Theory - with caution

- Linear regression
- Decision Trees
- RF
- ANNs
- ....

Main Concepts

problem representation

Build a model to classify Oranges from other objects

- A training set of labeled objects (instances)
- We want to classify oranges;
- (?) attributes:{‘**colour**’, ‘**weight**’, ‘**texture**’, ….}

... representation

an instance

x_1

attributes	feature
Colour	3
Weight	130
Texture	2
. . .	...

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

Features **vector**
for instance (1)

… representation

an instance
x_1

Colour	3
Weight	130
Texture	2
. . .	…

attributes (column) — **feature** (row)

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

Features **vector**
for *instance (1)*

$$\{\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \ldots, \mathrm{x}_n\} \in X$$

... representation

an instance

x_1

Colour	3
Weight	130
Texture	2
. . .	...

feature

attributes

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

Features **vector** for instance (1)

x_2

Colour	7
Weight	40
Texture	2
. . .	...

$$\{\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \ldots, \mathrm{x}_n\} \in X$$

… representation

$$\{x_1, x_2, x_3, \ldots, x_n\} \in X$$

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

$$x_2 = \begin{pmatrix} 7 \\ 40 \\ 2 \\ . \\ . \end{pmatrix}$$

… representation

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

$$x_2 = \begin{pmatrix} 7 \\ 40 \\ 2 \\ . \\ . \end{pmatrix}$$

$$\{\mathrm{x}_1, \mathrm{x}_2, \mathrm{x}_3, \ldots, \mathrm{x}_n\} \in X$$

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix}$$

vector of sample
features' **vectors**

… representation

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix}$$

… representation

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix}$$

colour | weight | texture

… representation

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ \cdot \\ \cdot \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ \cdot\cdot & \cdot\cdot & \cdot\cdot \\ \cdot\cdot & \cdot\cdot & \cdot\cdot \end{pmatrix}$$

colour | weight | texture

```
>>> import numpy as np

>>> X = np.array([[3, 130, 2],
[7, 40, 2]])

>>> X
```

... representation

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix}$$

$$Y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \end{pmatrix}$$

labels vector

... representation

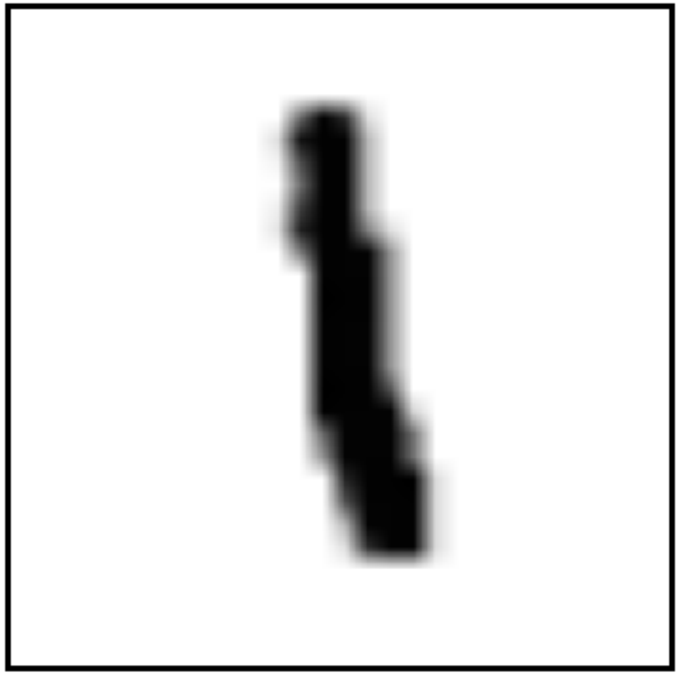
$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix}$$

an orange

$$Y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \end{pmatrix}$$

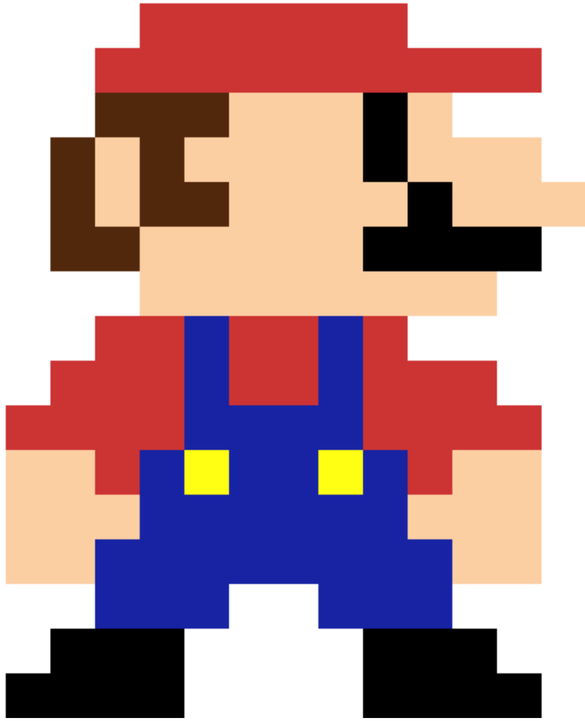
labels vector

Representation (OCR)



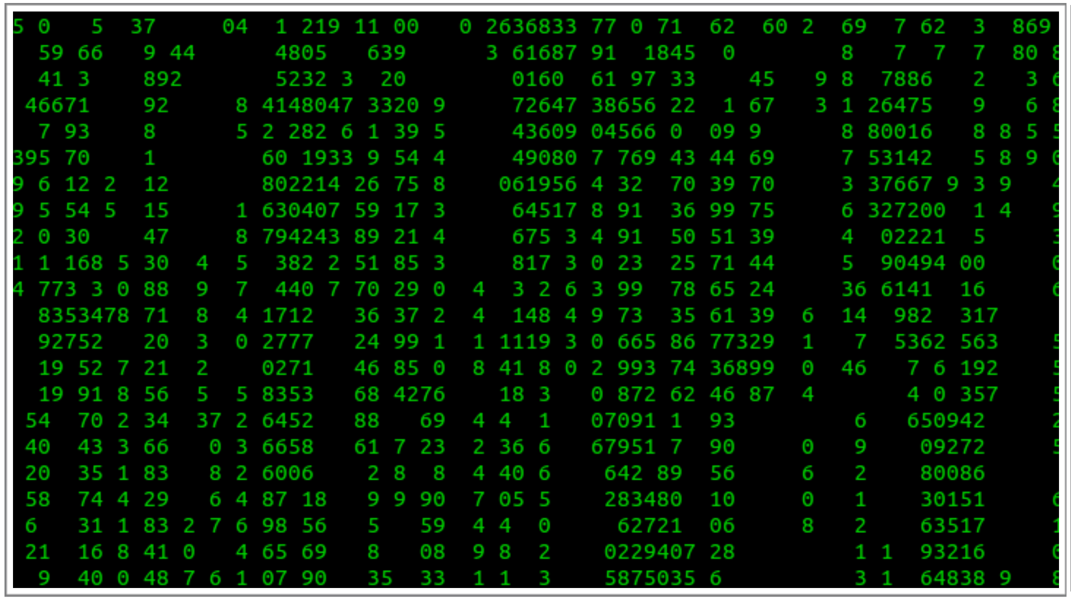
$$
\simeq
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .6 & .8 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .7 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & .5 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .4 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & .7 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .9 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & .3 & 1 & .1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

Representation (graphics)

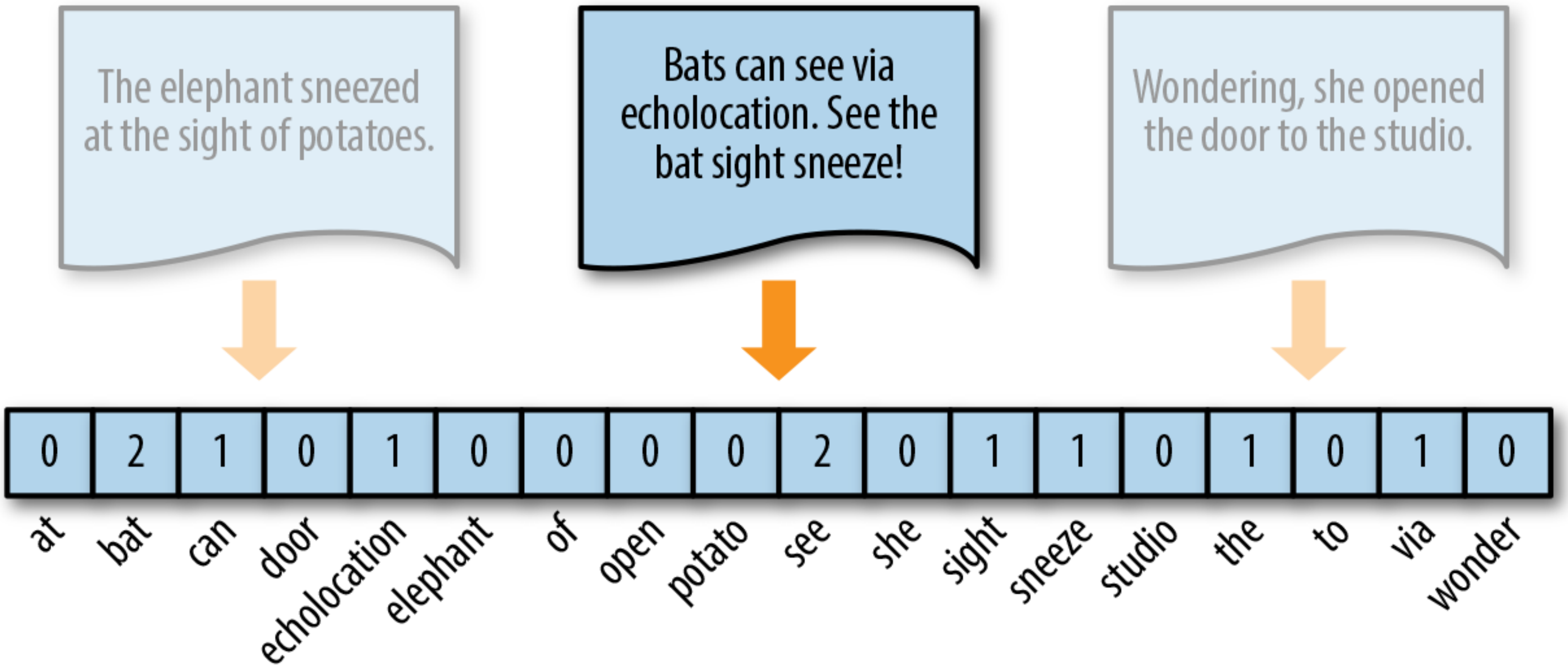


wanna watch me move!?

Representation (graphics)

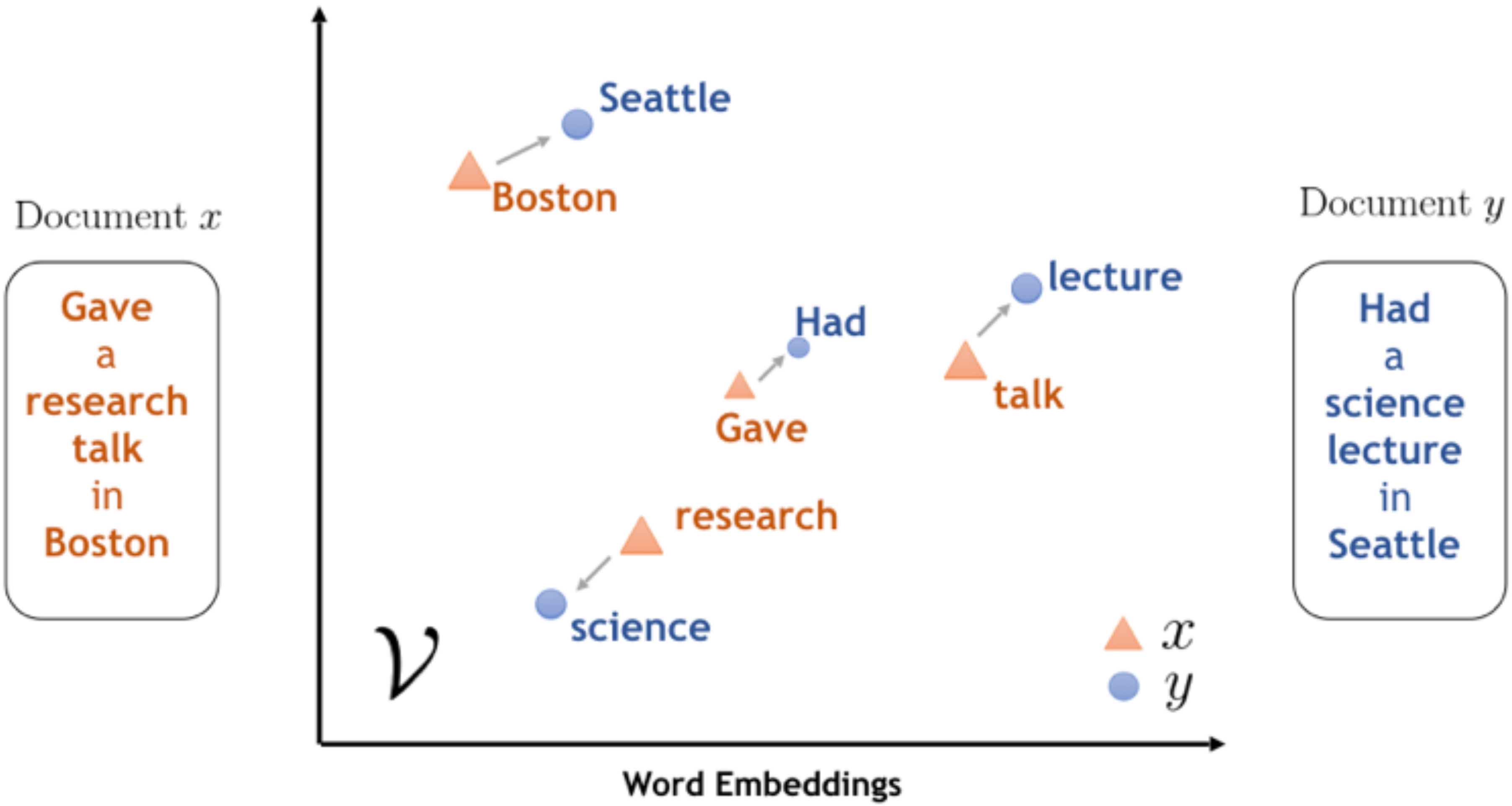


Documents / Texts



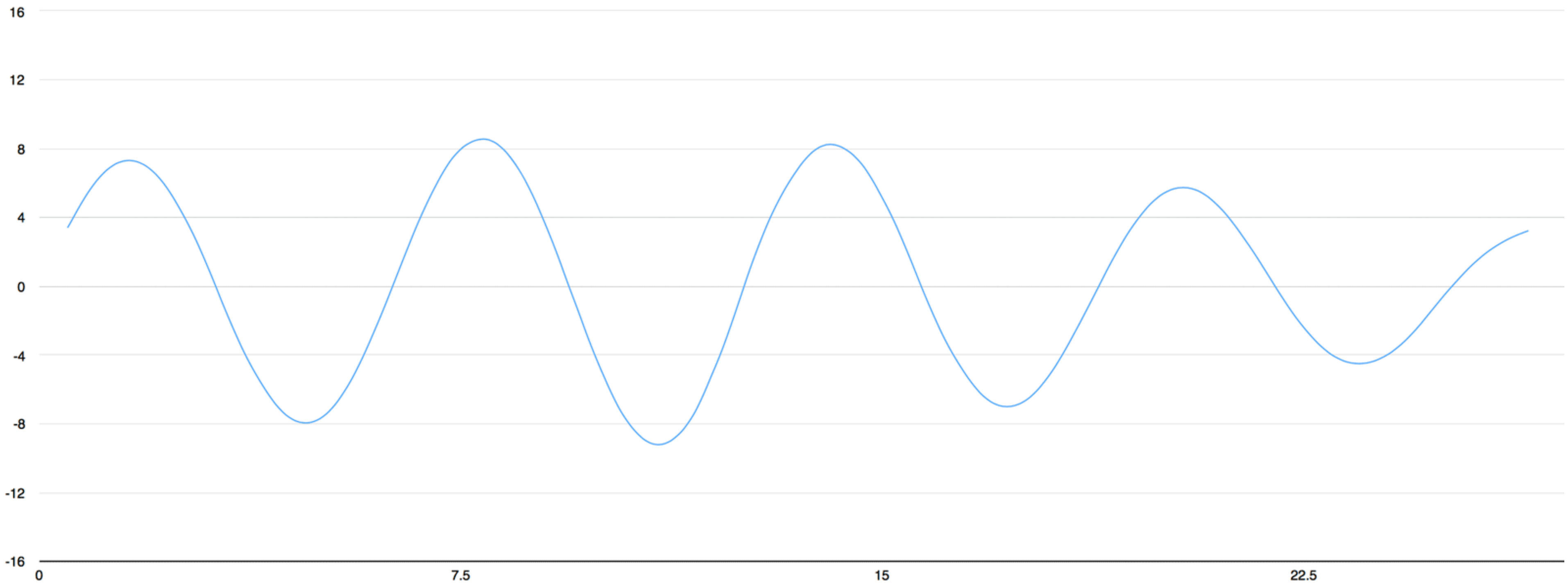
excerpt from Applied Text Analysis with Python, Ed. 2018

Documents / Texts



excerpt from EMNLP 2018 paper by IBM; “Word Mover’s Embedding: From Word2Vec to Document Embedding”.

Representation (speech)




```
[-1274, -1252, -1160, -986, -792, -692, -614, -429, -286, -134, -57, -41, -169, -456, -450, -541, -761, -1067, -1231, -1047, -952, -645, -489, -448
, -397, -212, 193, 114, -17, -110, 128, 261, 198, 390, 461, 772, 948, 1451, 1974, 2624, 3793, 4968, 5939, 6057, 6581, 7302, 7640, 7223, 6119, 5461,
 4820, 4353, 3611, 2740, 2004, 1349, 1178, 1085, 901, 301, -262, -499, -488, -707, -1406, -1997, -2377, -2494, -2605, -2675, -2627, -2500, -2148, -
1648, -970, -364, 13, 260, 494, 788, 1011, 938, 717, 507, 323, 324, 325, 350, 103, -113, 64, 176, 93, -249, -461, -606, -909, -1159, -1307, -1544]
```

Each value represents the amplitude of the sound wave at 1/16000th of a second intervals

check this out: https://openai.com/blog/musenet/

'simpleS'

Data… representation

$$X = \begin{pmatrix} (x_1)^T \\ (x_2)^T \\ (x_3)^T \\ . \\ . \\ (x_n)^T \end{pmatrix} = \begin{pmatrix} 3 & 130 & 2 \\ 7 & 40 & 2 \\ .. & .. & .. \\ .. & .. & .. \end{pmatrix}$$

$$Y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \end{pmatrix}$$

an orange

labels vector

Output is not always $\begin{pmatrix} 1 \\ 0 \\ 0 \\ . \end{pmatrix}$

Classification

Y discrete

Will you pass the exam?

Regression

Y continuous

How many points will you get in the exam?

Data… representation

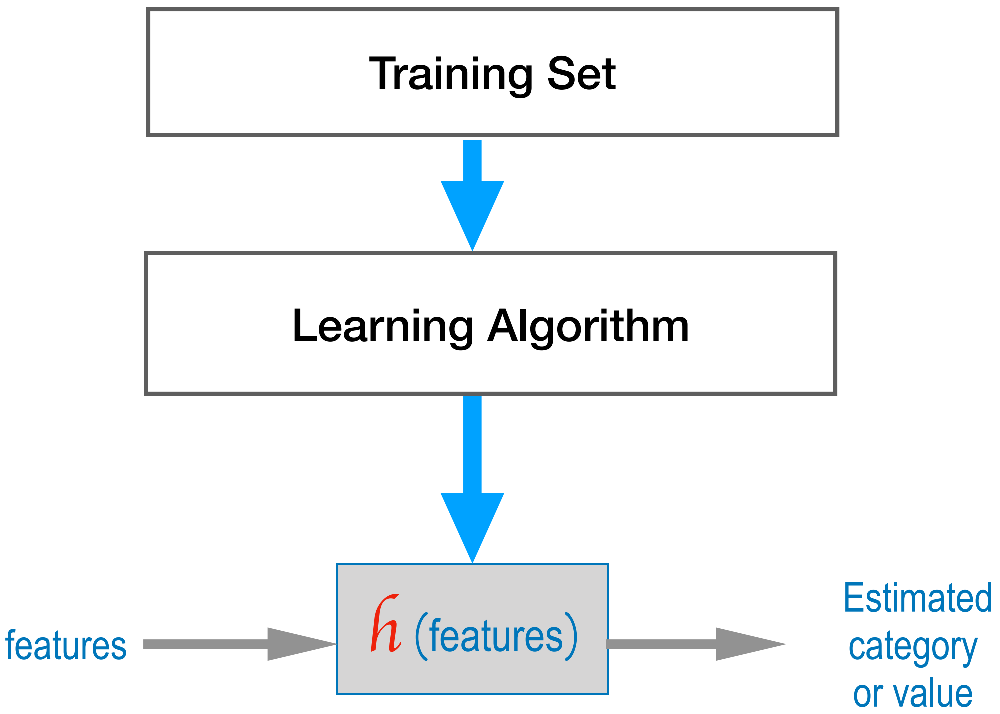
X =

1.1	2.2	3.4	5.6	1.0
6.7	0.5	0.4	2.6	1.6
2.4	9.3	7.3	6.4	2.8
1.5	0.0	4.3	8.3	3.4
0.5	3.5	8.1	3.6	4.6
5.1	9.7	3.5	7.9	5.1
3.7	7.8	2.6	3.2	6.3

y =

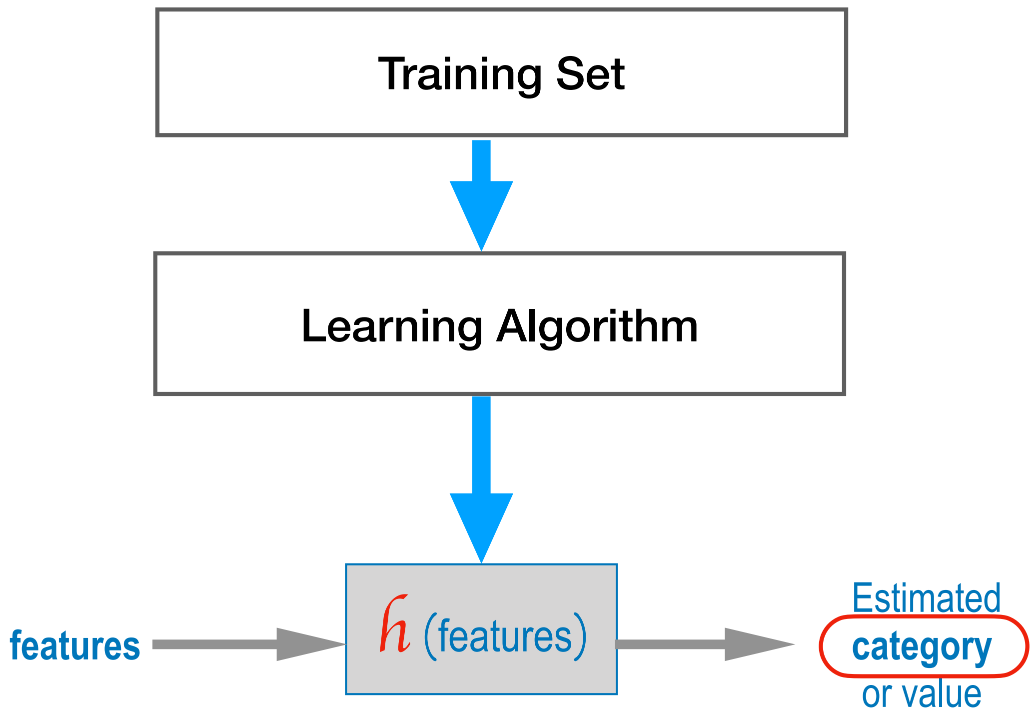
1.6
2.7
4.4
0.5
0.2
5.6
6.7

Process


Training Set
Learning Algorithm
features
h (features)
Estimated
category
or value


ORANGE
JUICE

A Hypothesis (guess)

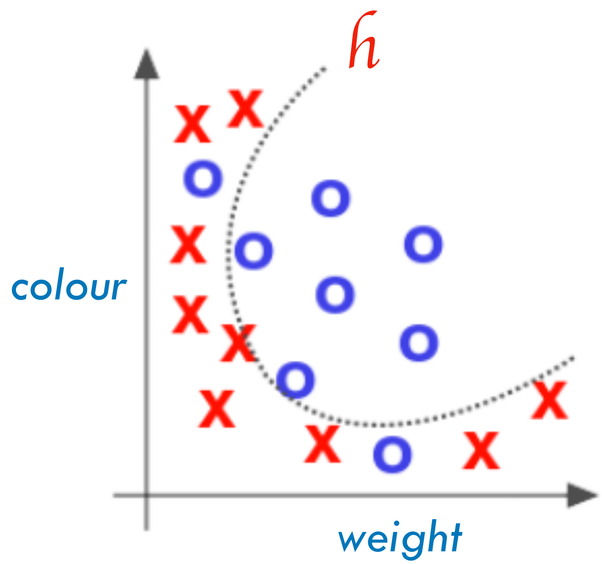


a model

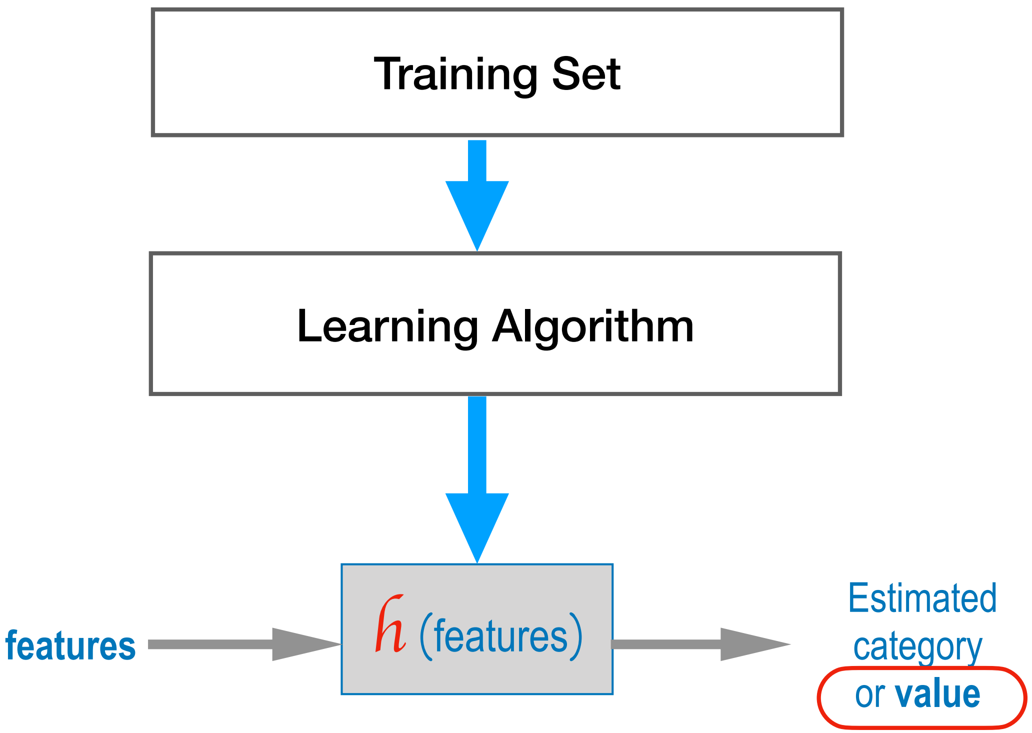
h(weight, colour): IF $\theta_0 + \theta_{1*}$(**weight**) > Val_0 AND

(**colour**) == Val_1,

THEN **1** ELSE **0**

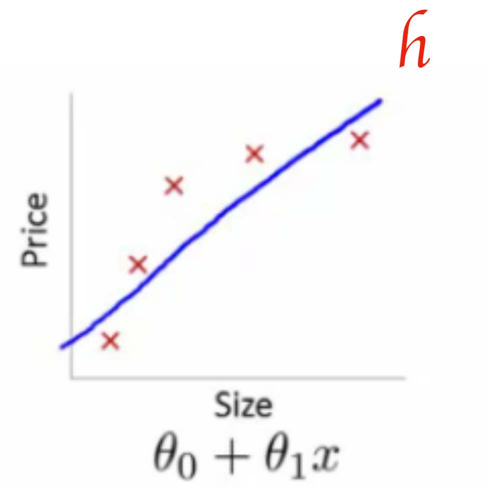


A Hypothesis (guess)



[a model]

$h(x_1) = \theta_0 + \theta_1 x_1$



Definitions & Conventions

- $\theta_0, \theta_1, \theta_2, \ldots, \theta_{m-1}$: model parameters.
- $\{x_0, x_1, x_2, \ldots, x_{m-1}\} \in X$: training set (of size *m*) - each **x** *is an instance.*
- each **x** is basically a vector ($\vec{x}$) which represents the features of the corresponding instance.
- h: the hypothesis (sometimes we use *f()*) - which is the model that maps $\vec{x}$ to y (in supervised ML)

> The process of finding h that maps $\vec{x}$ to y (in supervised ML) is called *fitting* model h or (f).

$$x_1 = \begin{pmatrix} 3 \\ 130 \\ 2 \\ . \\ . \end{pmatrix}$$

$$x_2 = \begin{pmatrix} 7 \\ 40 \\ 2 \\ . \\ . \end{pmatrix}$$

a model

$$h(x_1) = \theta_0 + \theta_1 x_1$$

$$Y = \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \end{pmatrix}$$

Definitions & Conventions

- $\theta_0, \theta_1, \theta_2, \ldots, \theta_{m-1}$: model parameters.
- $\{x_0, x_1, x_2, \ldots, x_{m-1}\} \in X$: training set (of size *m*) - each **x** *is an instance.*
- each **x** is basically a vector ($\vec{x}$) which represents the features of the corresponding instance.
- h: the hypothesis (sometimes we use *f()*) - which is the model that maps $\vec{x}$ to y (in supervised ML)

The process of finding h that maps $\vec{x}$ to y (in supervised ML) is called *fitting* model h or (f).

$$(x_i, y_i) \propto p(x, y) \quad \text{i.i.d.}$$

$$x_i \in \mathbb{R}^n$$

$$y_i \in \mathbb{R}$$

$$f(x_i) \approx y_i$$

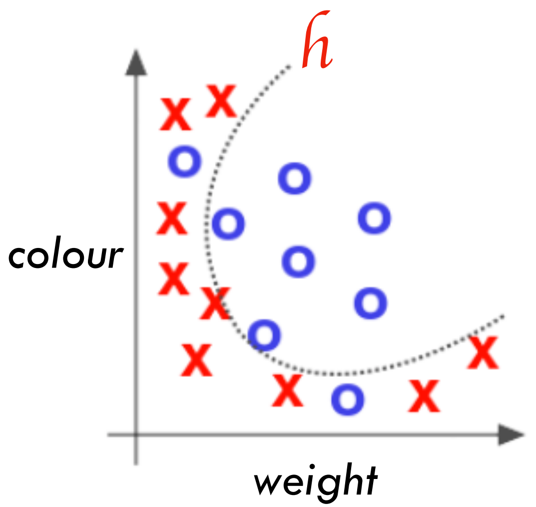
Which is then the “optimum” model/guess?

Evaluation

A

a model

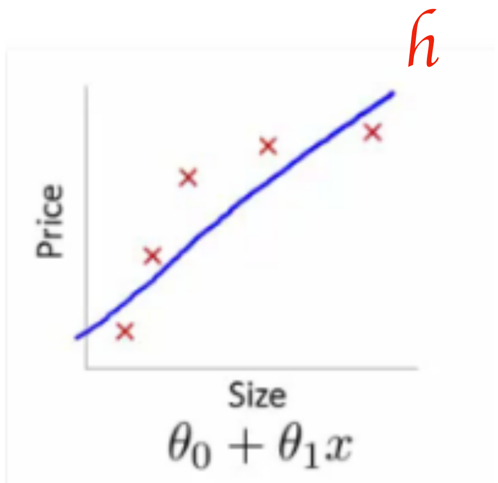
h(weight, colour): IF $\theta_0 + \theta_1 *$**(weight)** > *Val_0* AND **(colour)** == *Val_1*, THEN **1** ELSE **0**



B

a model

$h(x_1) = \theta_0 + \theta_1 x_1$

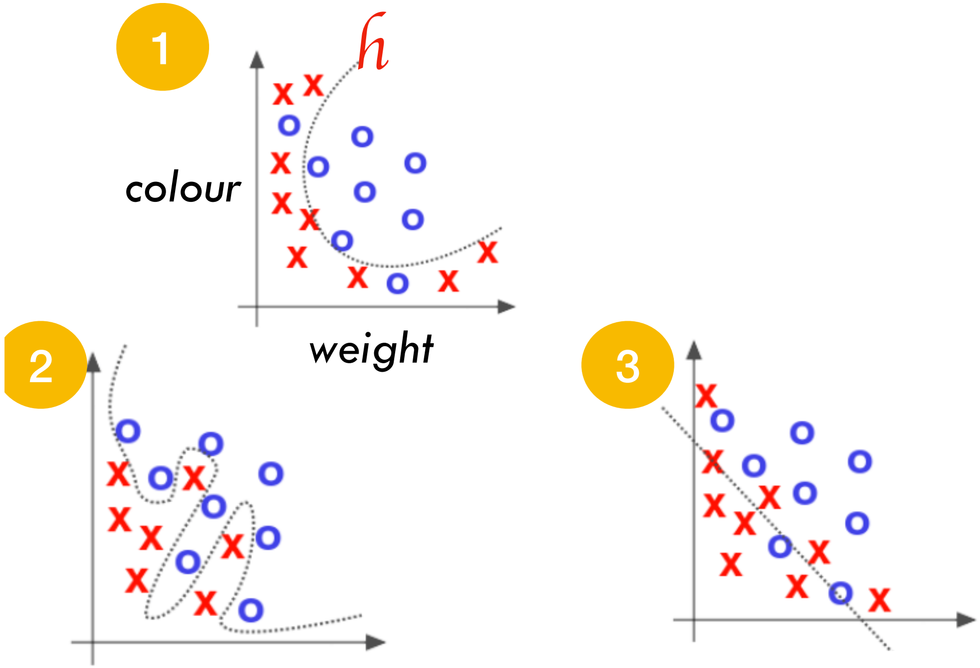


Evaluation

A

a model

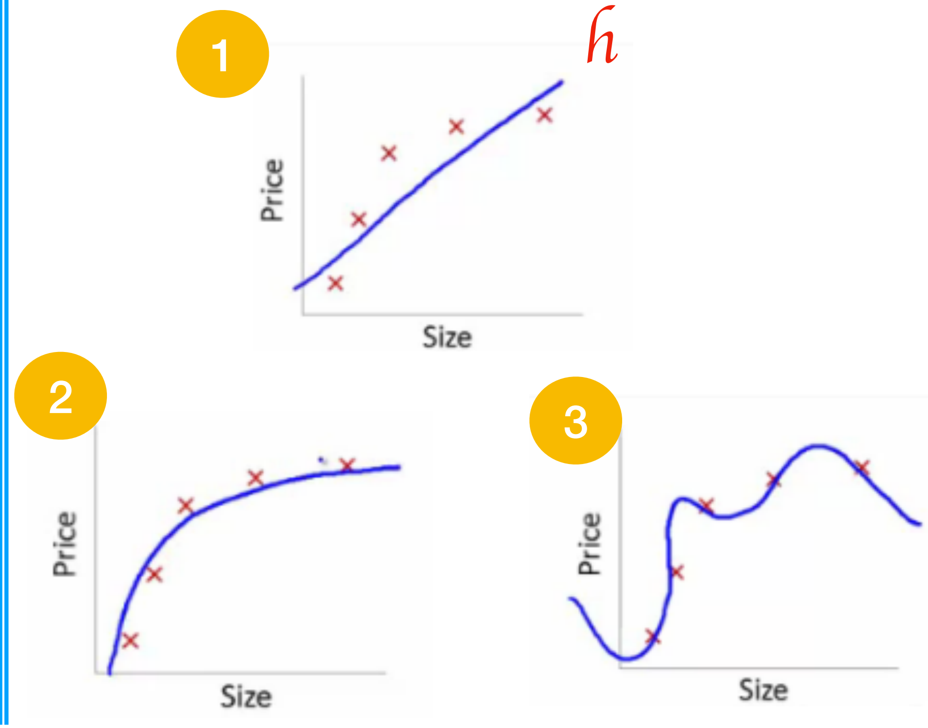
h(weight, colour): IF $\theta_0 + \theta_{1*}$(**weight**) > Val_0 AND (**colour**) $==$ Val_1, THEN **1** ELSE **0**



B

a model

$h(x_1) = \theta_0 + \theta_1 x_1$



Generalisation

not only $f(x_i) \approx y_i$

but also $f(x) \approx y$

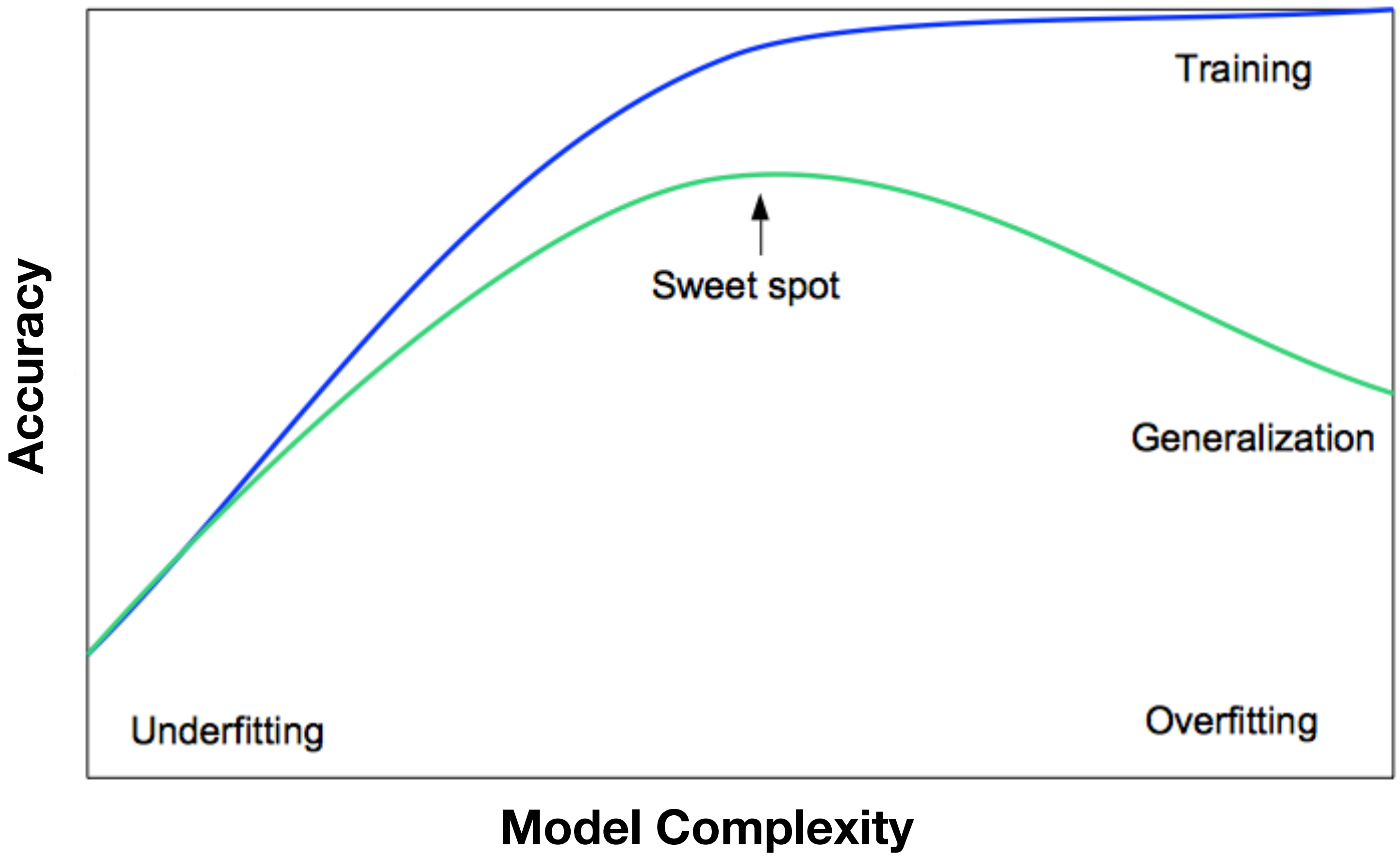
'general model'

We’ll have to evaluate…

the ‘goodness of fit’

let’s call it (for now): the “*accuracy*” of the guess (learnt model)

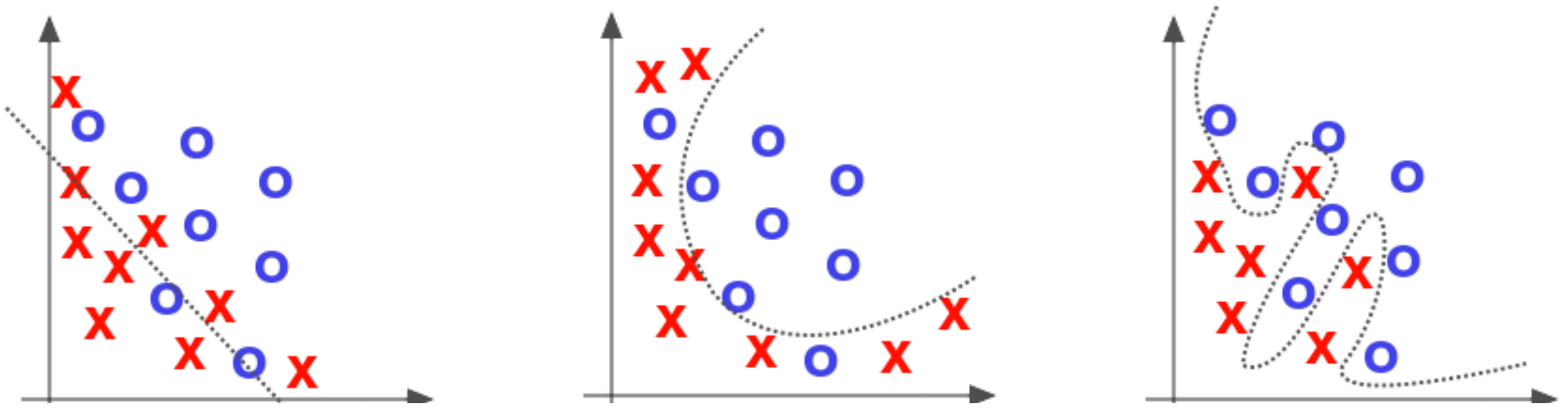
Generalisation



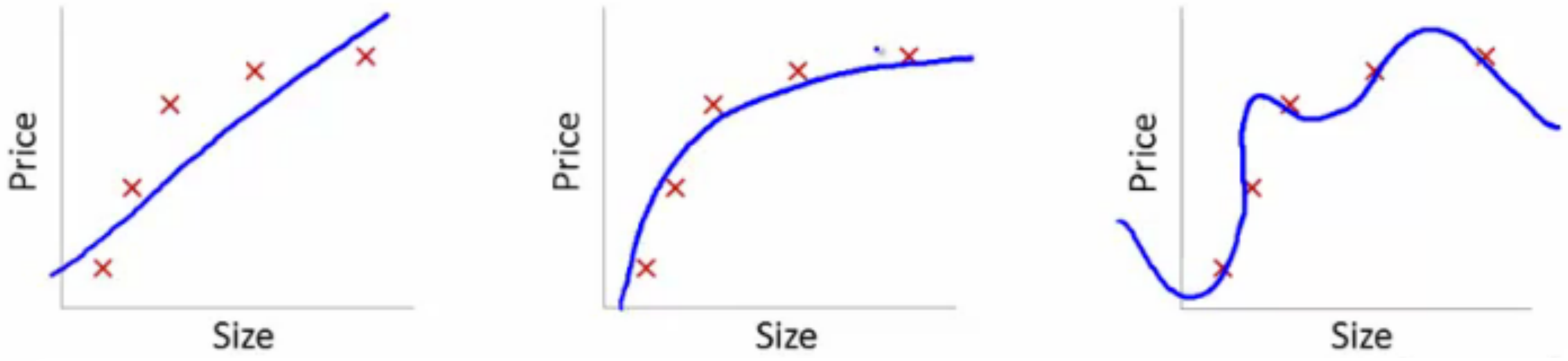
Underfitting & Overfitting

Tradeoff

A

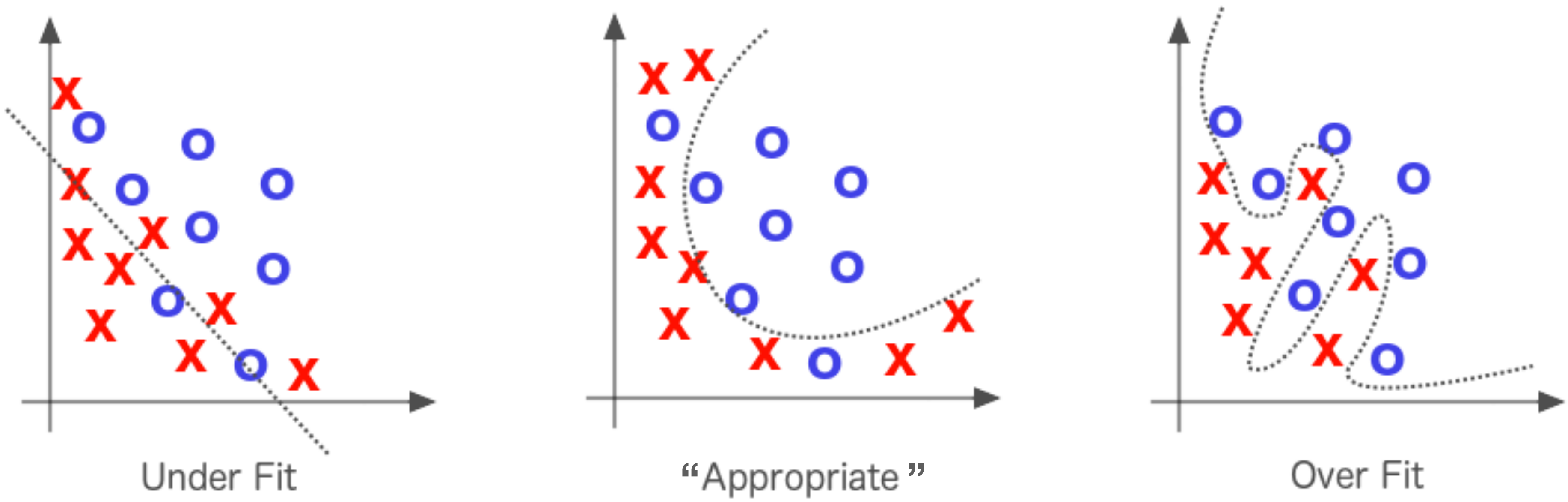


B



Generalisation

A



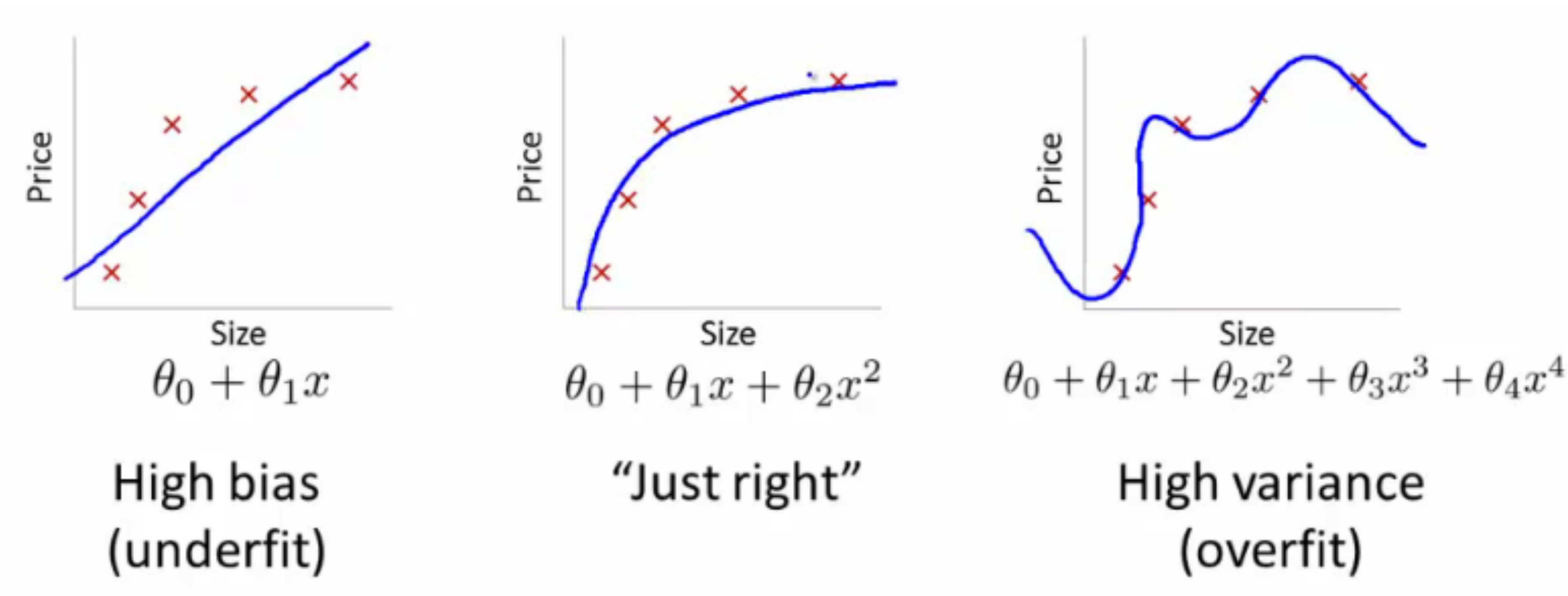
Underfitting — on TikTok!

Bias Vs Variance tradeoff

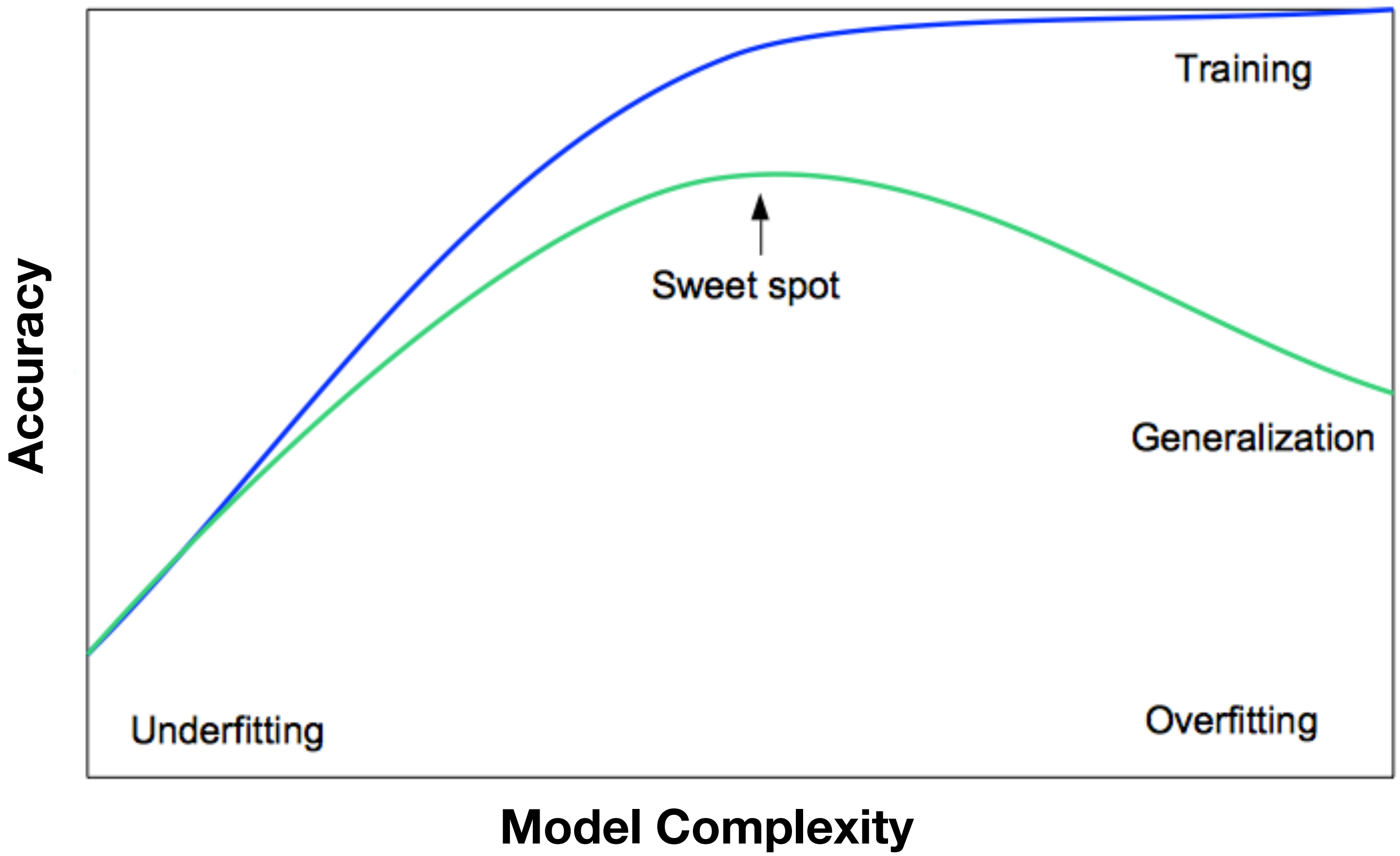
Underfitting is the high bias. Overfitting is the high variance.

B

“**optimum**” model?

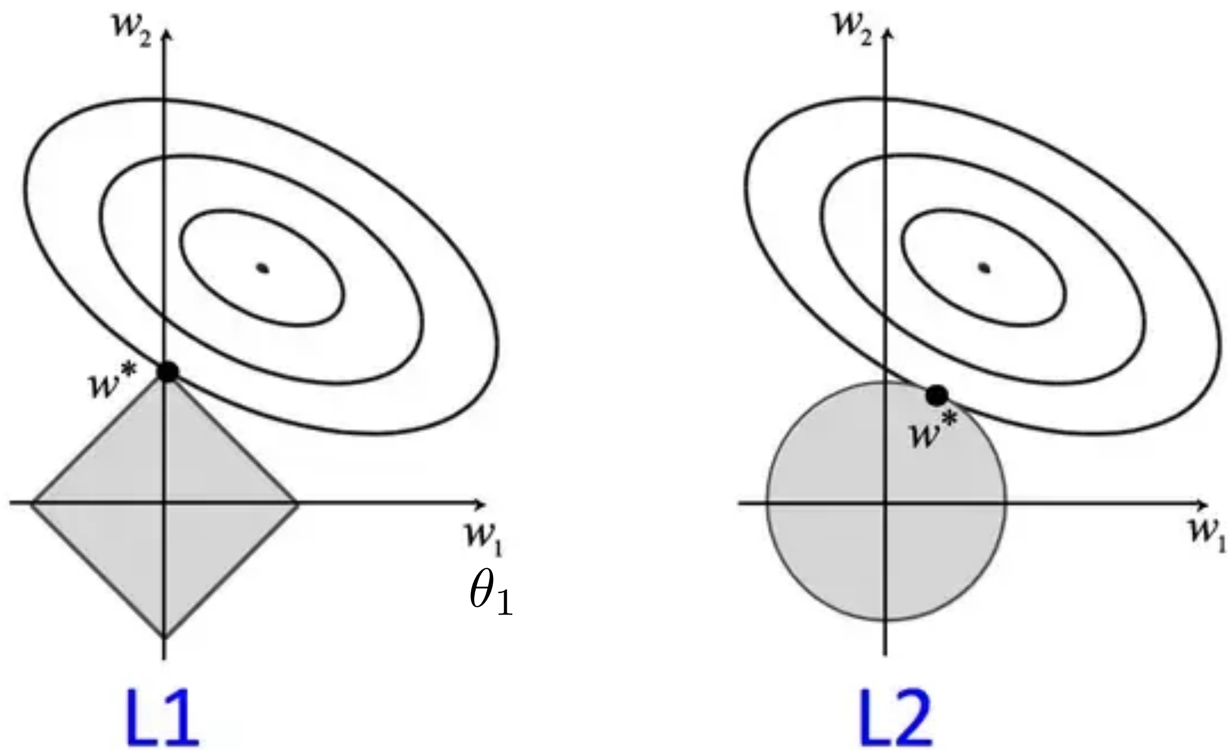


Generalisation



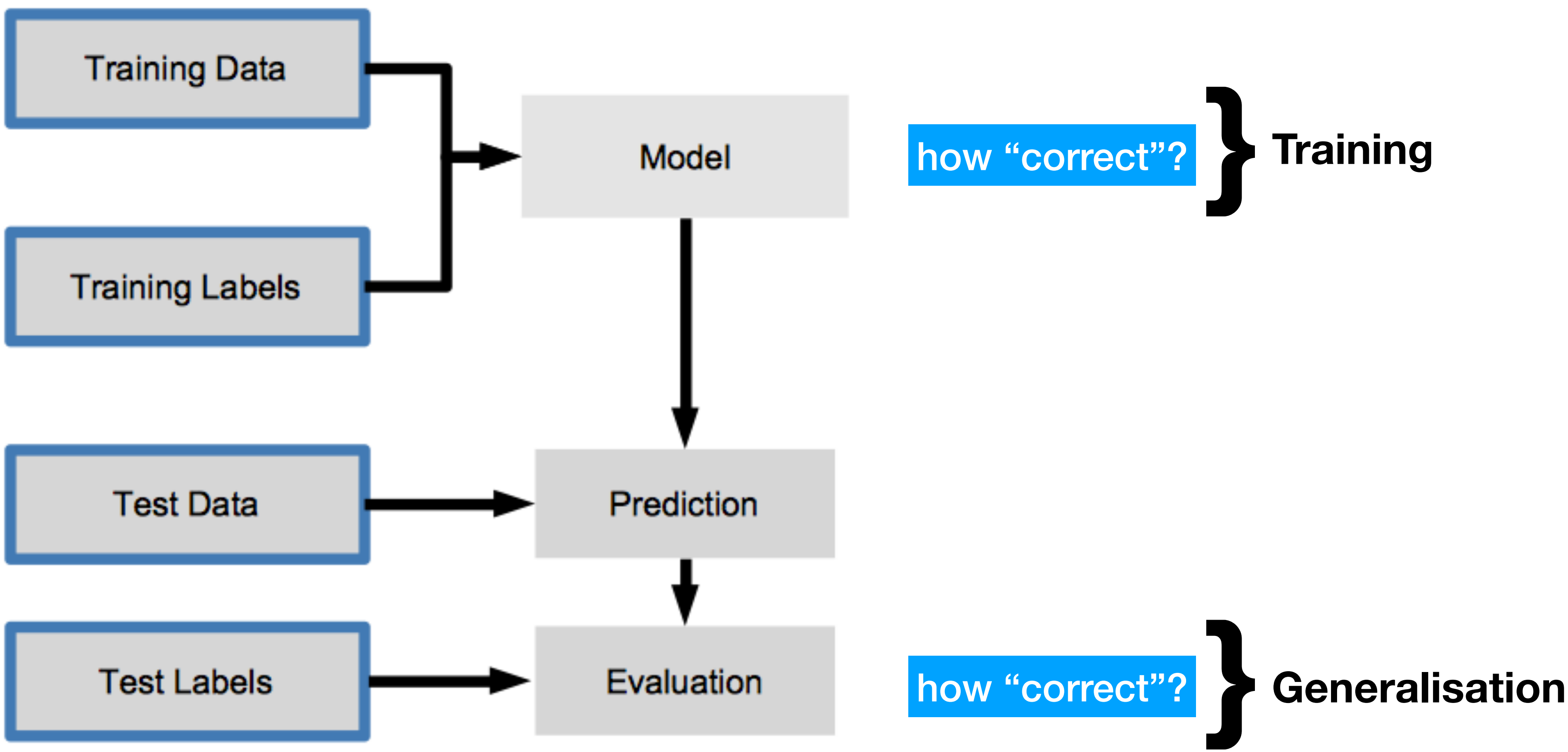
Regularisation

(Leave for now)



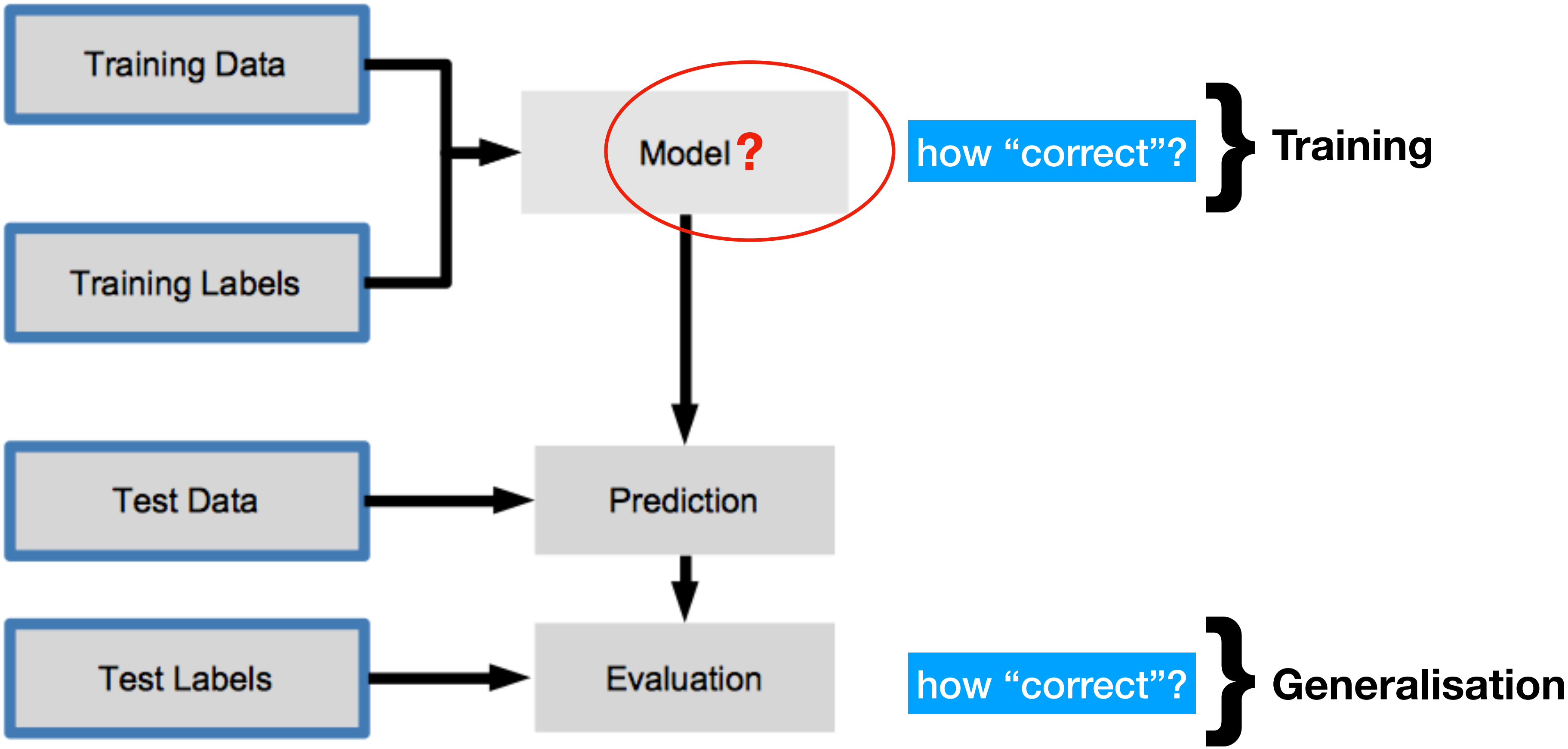
Evaluate,

in training and in testing



Evaluate,

in training and in testing, ...and



Performance Measures

Specific metrics will be covered and applied later

Regression

- MSPE
- MSAE
- R Square
- Adjusted R Square

Classification

- Precision-Recall
- ROC-AUC
- Accuracy
- Log-Loss

Unsupervised Models

- Rand Index
- Mutual Information

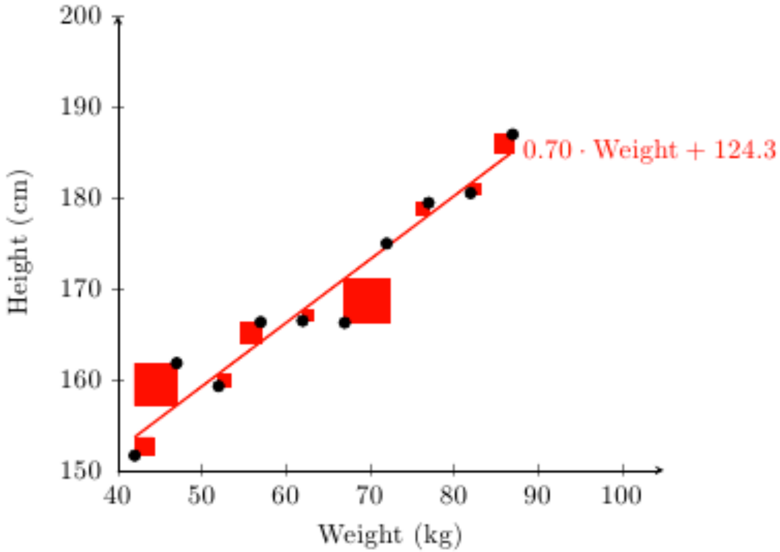
Others

- CV Error
- Heuristic methods to find K
- BLEU Score (NLP)

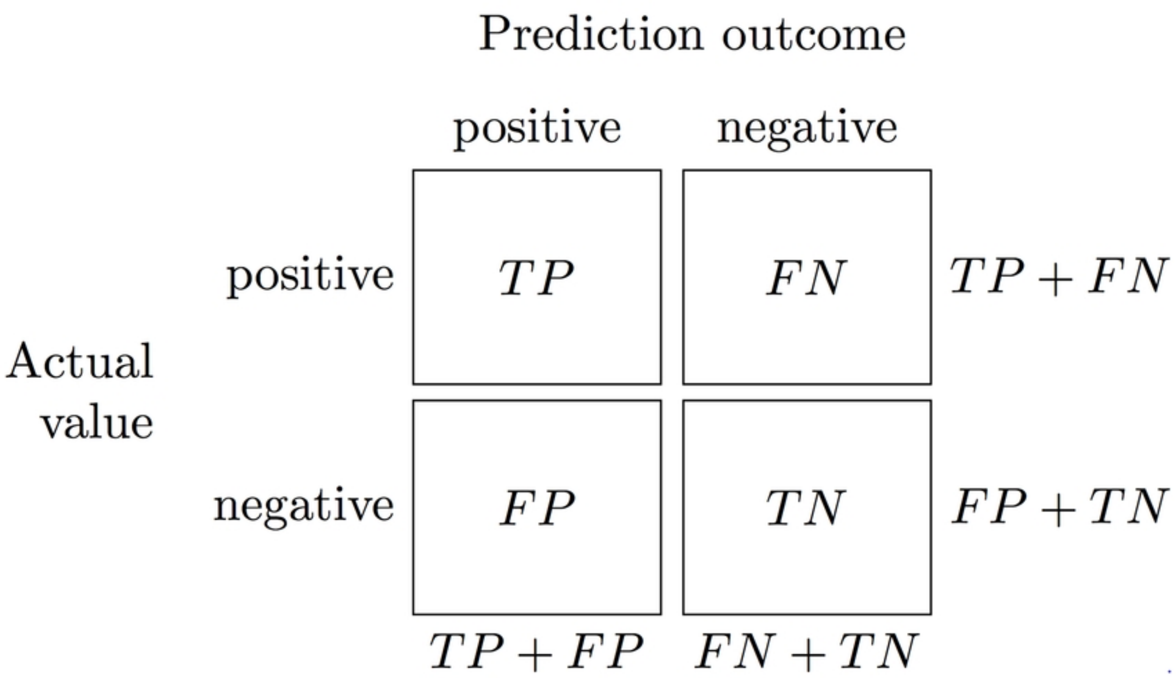
Performance Measures

Mean squared error: 6.41243

x	y	regression	error
42	151.62	153.64	−2.03
47	161.79	157.13	4.65
52	159.25	160.63	−1.37
57	166.31	164.12	2.19
62	166.46	167.61	−1.15
67	166.24	171.1	−4.86
72	174.96	174.59	0.37
77	179.44	178.08	1.36
82	180.51	181.57	−1.06
87	186.97	185.06	1.91



MSE



Confusion Matrix

Recap

- Categories of Learning
- Main Concepts
    - Problem Representation
    - Data Representation - features vector, hypotheses, labels vector, etc.
    - Introduction to Evaluation - when, why & where!

Reading

- **E. Alpaydin (Intro ML):** Section 2.7 (pp. 37–41)
- **J Gurus (ML from Scratch):** Ch 11 (pp. 142–145)
- **A Géron (Hands on ML..):** Ch 1&2 (pp. 3–31 & 33—39)

Interesting and simplified

Jason Brownlee's blog:

- A good summary about 'A tour of Machine Learning Algorithms' - https://goo.gl/xhQvp8

- 'Gentle Introduction to the Bias-Variance Tradeoff in ML' - https://goo.gl/YtAVPS

Next week more on: evaluation, regularisation and similarity.