Three Components of Deep Learning

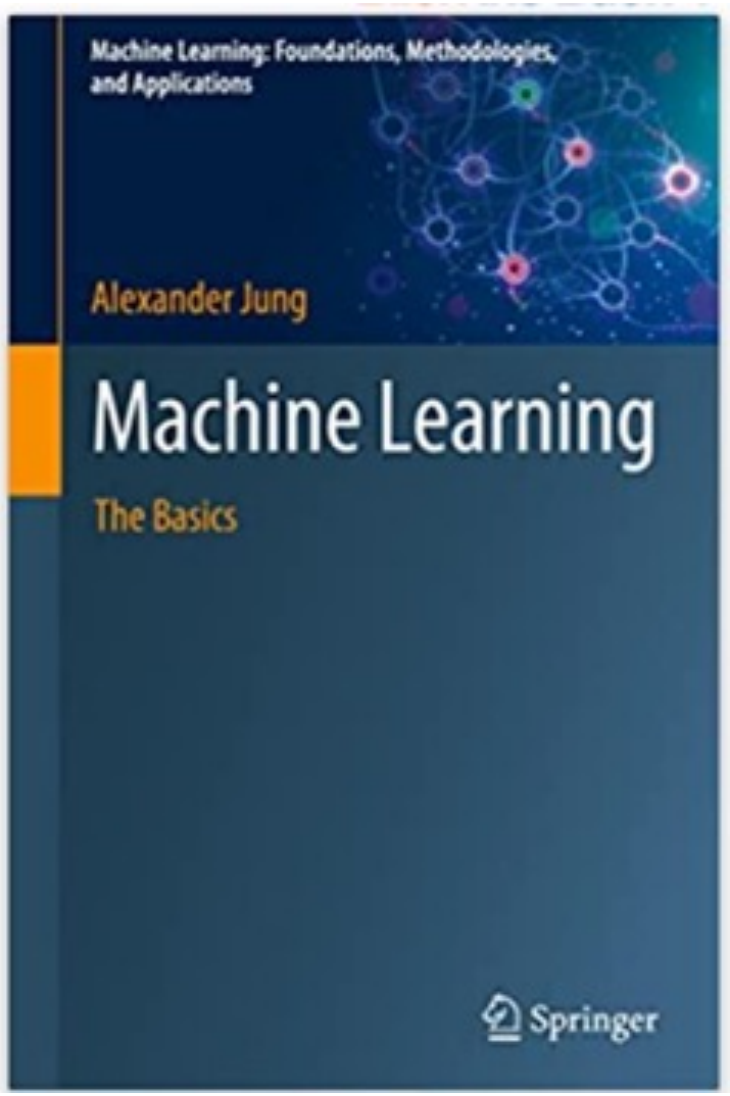
Alex(ander) Jung
Assistant Professor for Machine Learning
Department of Computer Science
Aalto University

Learning Goals

- develop intuition for how ML works
- become familiar with concept of
  - data points (features, labels)
  - model (hypothesis space)
  - loss function (quality measure)

Reading.

- Chapter 1,2 of [MLBook]



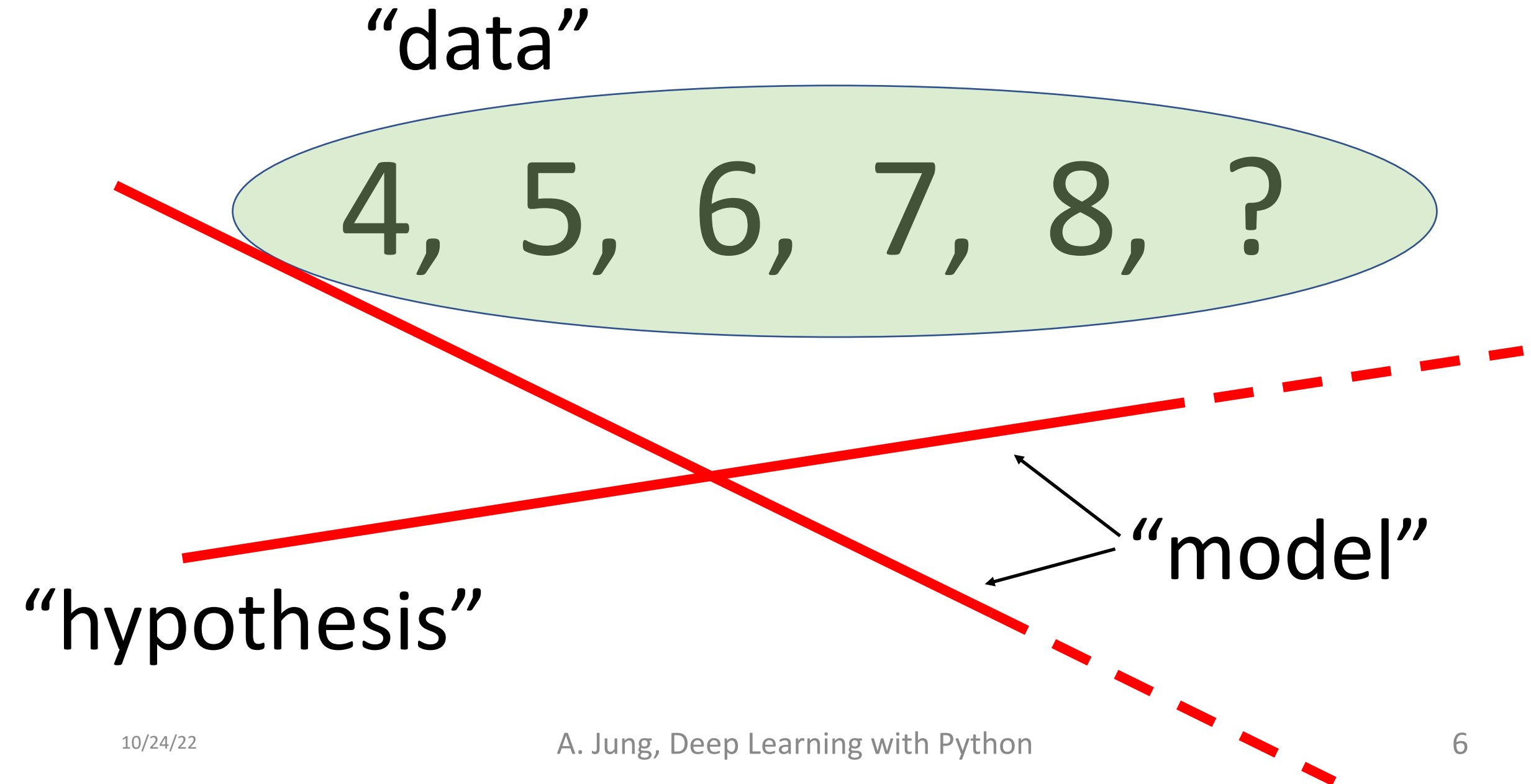
NumPy

https://numpy.org/doc/stable/user/absolute_beginners.html

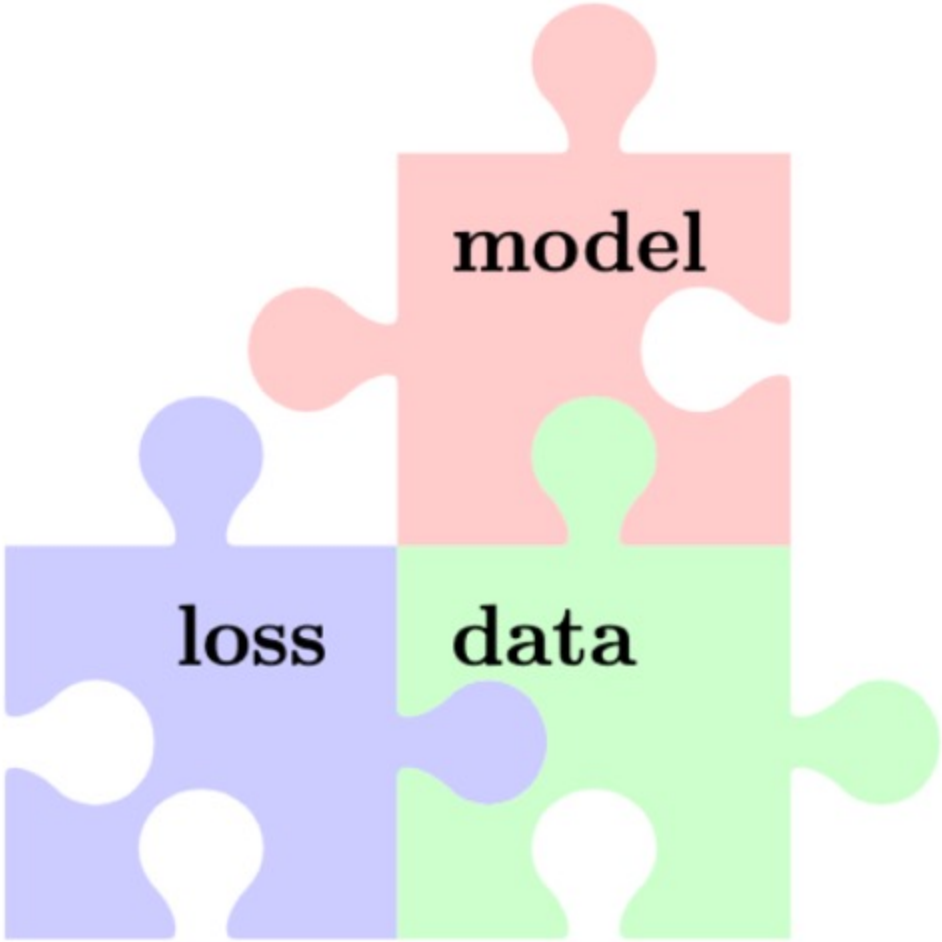
What is it all About ?

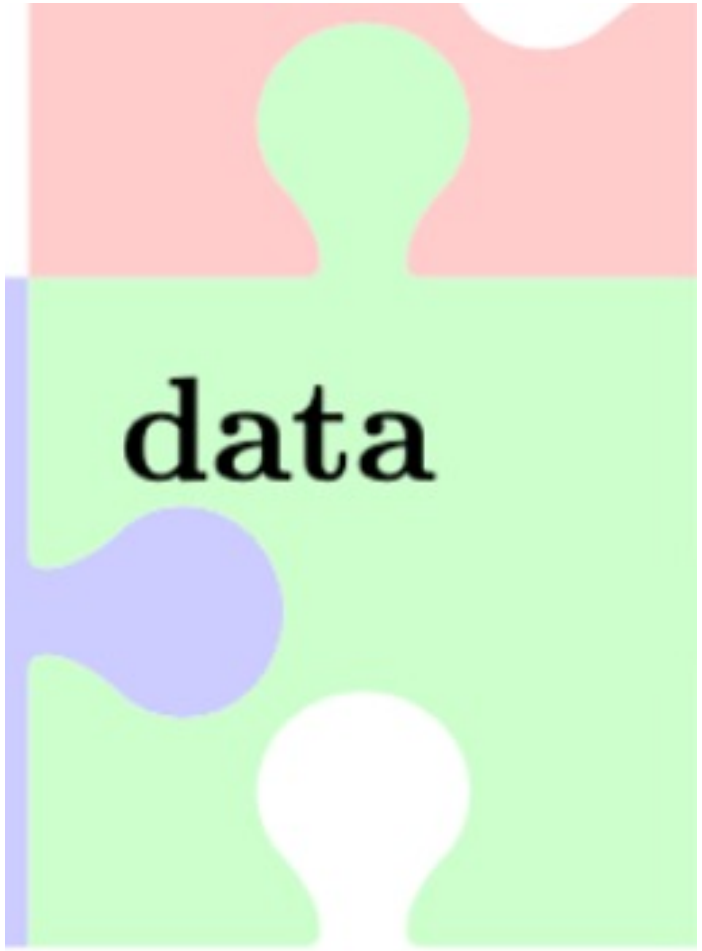
fit model to data to make accurate predictions or forecasts !

4, 5, 6, 7, 8, ?


"data"
4, 5, 6, 7, 8, ?
"model"
"hypothesis"

Three Components of ML





"What I'm finding is that for a lot of problems, it'd be useful to shift our mindset toward not just improving the code but in a more systematic way of improving the data," said Andrew Ng

https://read.deeplearning.ai/the-batch/issue-84/

data

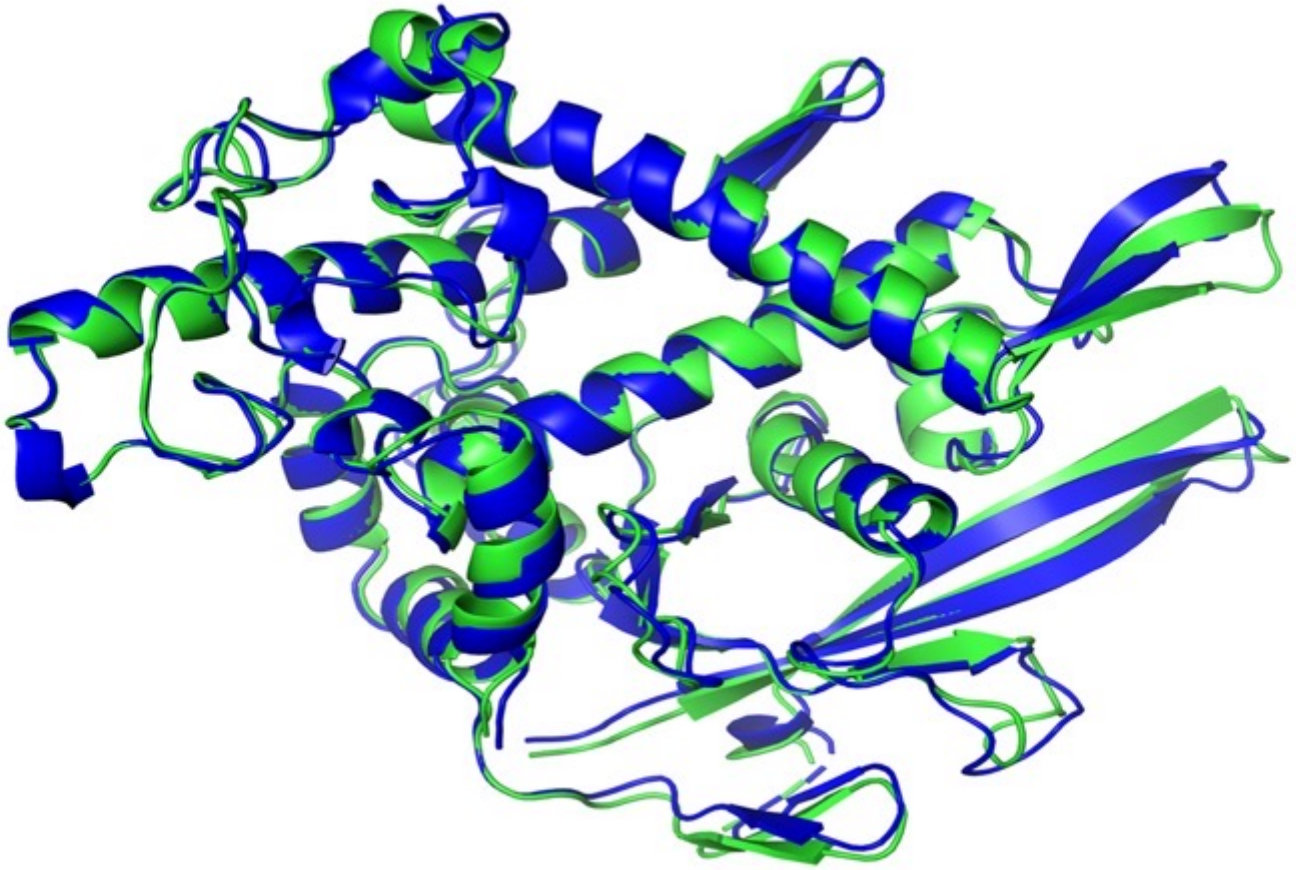
=

set of datapoints

What is a Datapoint?

some object that carries relevant information

Datapoint = Some Protein



Datapoint = A Partial Differential Equation

$$\frac{\partial u}{\partial t}(t,x)+\frac{1}{2}\mathrm{Tr}\left(\sigma\sigma^{\mathrm{T}}(t,x)(\mathrm{Hess}_x u)(t,x)\right)+\nabla u(t,x)\cdot\mu(t,x)$$
$$+f\left(t,x,u(t,x),\sigma^{\mathrm{T}}(t,x)\nabla u(t,x)\right)=0 \qquad \textbf{[1]}$$

RESEARCH ARTICLE

Solving high-dimensional partial differential equations using deep learning

Jiequn Han, Arnulf Jentzen, and Weinan E

+ See all authors and affiliations

https://www.pnas.org/content/115/34/8505/tab-article-info

Datapoint = Some Bridge

https://commons.wikimedia.org/wiki/Category:Bridges

Datapoint = Some Cow

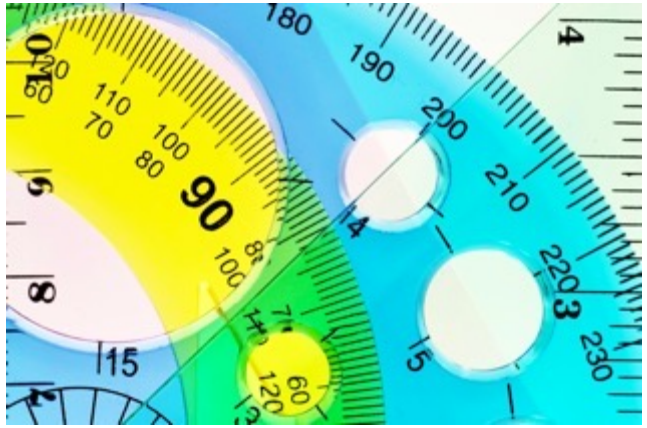
Features and Labels.

datapoint characterized by

- features: low-level properties; easy to measure/compute

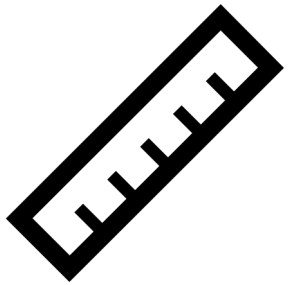
- labels: high-level quantity of interest; difficult to measure/determine

Numeric Features

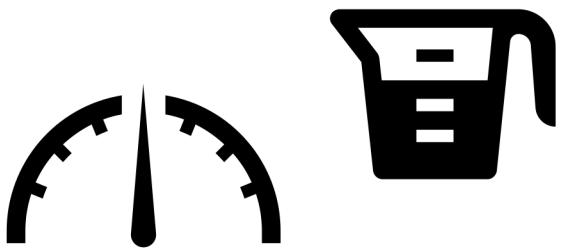


we mainly use numeric features x1,…,xn to characterize a datapoint

stack features into feature vector



Python: use numpy array to store features



discuss feature learning methods later

Features of an Image.

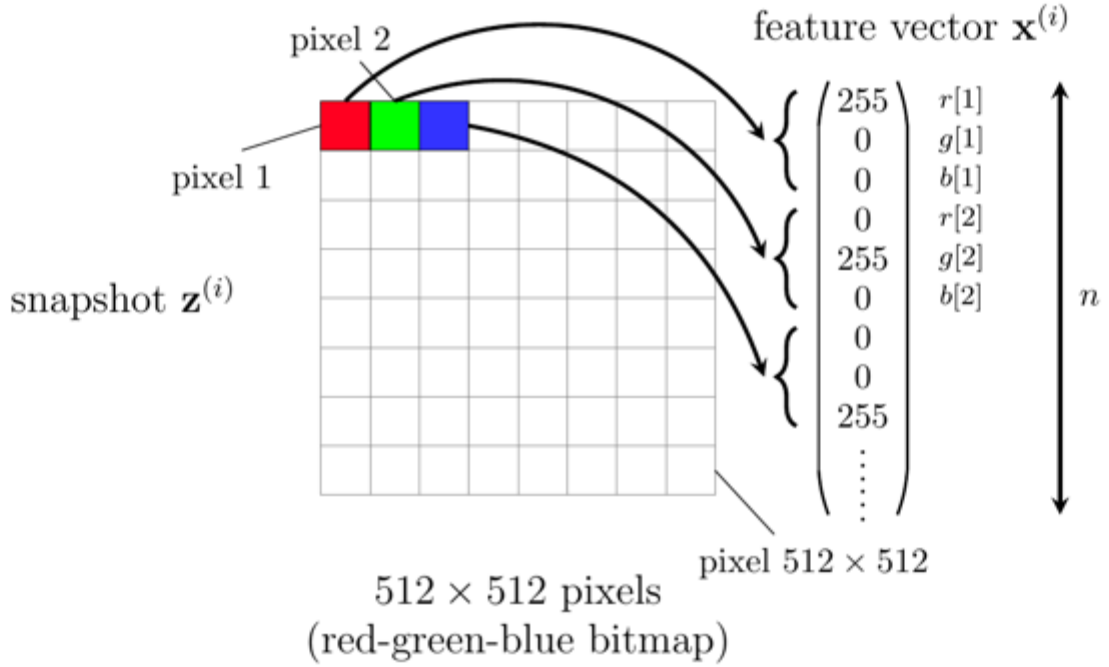
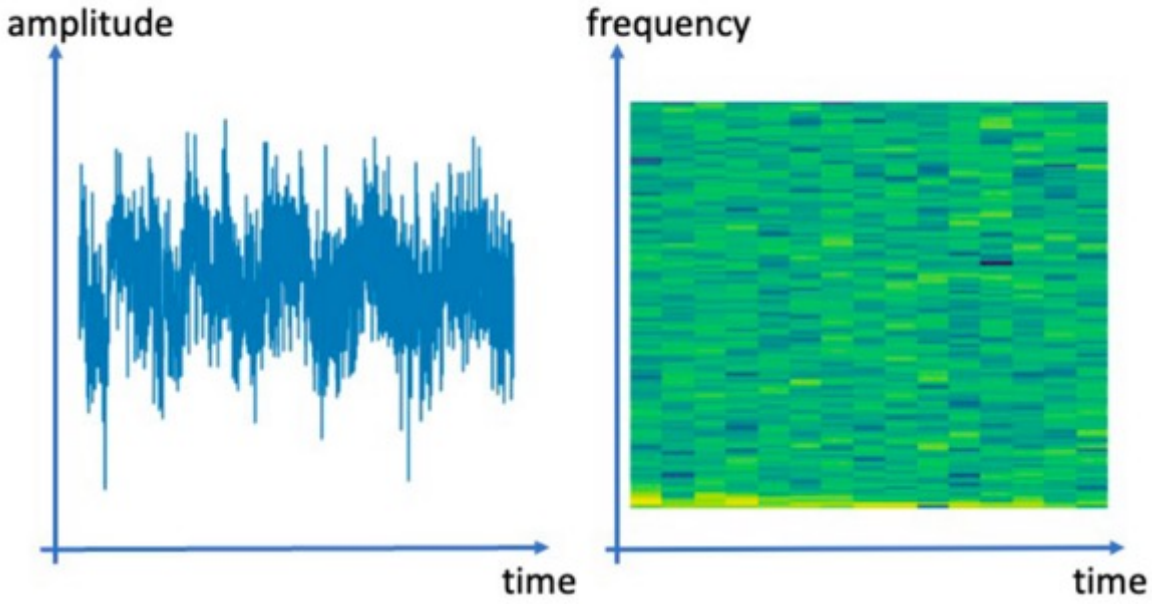


Figure 2.5: If the snapshot $\mathbf{z}^{(i)}$ is stored as a 512×512 RGB bitmap, we could use as features $\mathbf{x}^{(i)} \in \mathbb{R}^n$ the red-, green- and blue component of each pixel in the snapshot. The length of the feature vector would then be $n = 3 \times 512 \times 512 \approx 786000$.

Features of an Audio Recording.



label = song title

Figure 2.4: Two visualizations of a data point that represents an audio recording. The left figure shows a line plot of the audio signal amplitudes. The right figure shows a spectogram of the audio recording.

Datapoint = A Cow

features:

- duration of pregnancy

labels:

- how much milk will it give next year?

Datapoint = A Partial Differential Equation

features = a solution of the PDE

label = formula of PDE

datapoints, their features and labels are design choices!

raw data from FMI

https://en.ilmatieteenlaitos.fi/download-observations

Year	m	d	Time	precp	snow	airtmp	mintmp	maxtmp
2020	1	2	00:00	0,4	55	2,5	-2	4,5
2020	1	3	00:00	1,6	53	0,8	-0,8	4,6
2020	1	4	00:00	0,1	51	-5,8	-11,1	-0,7
2020	1	5	00:00	1,9	52	-13,5	-19,1	-4,6
2020	1	6	00:00	0,6	52	-2,4	-11,4	-1
2020	1	7	00:00	4,1	52	0,4	-2	1,3
2020	1	8	00:00	4,3	51	0,8	0,1	1,8
2020	1	9	00:00	-1	51	-0,6	-1,9	1,6
2020	1	10	00:00	-1	51	-6,2	-11	-1,4
2020	1	11	00:00	2,8	50	-4,8	-10,7	-2,1
2020	1	12	00:00	-1	53	-1,3	-3,5	0,9
2020	1	13	00:00	-1	53	-6,4	-12,9	-3,1
2020	1	14	00:00	9,7	52	-2,8	-9	-0,7
2020	1	15	00:00	-1	63	0,2	-0,7	0,6
2020	1	16	00:00	0,4	62	-3,9	-5,2	0,1
2020	1	17	00:00	2	62	-5,2	-8,4	-0,7

features

Year	m	d	Time	precp	snow	airtmp	mintmp	maxtmp
2020	1	2	00:00	0,4	55	2,5	-2	4,5
2020	1	3	00:00	1,6	53	0,8	-0,8	4,6
2020	1	4	00:00	0,1	51	-5,8	-11,1	-0,7
2020	1	5	00:00	1,9	52	-13,5	-19,1	-4,6
2020	1	6	00:00	0,6	52	-2,4	-11,4	-1
2020	1	7	00:00	4,1	52	0,4	-2	1,3
2020	1	8	00:00	4,3	51	0,8	0,1	1,8
2020	1	9	00:00	-1	51	-0,6	-1,9	1,6
2020	1	10	00:00	-1	51	-6,2	-11	-1,4
2020	1	11	00:00	2,8	50	-4,8	-10,7	-2,1
2020	1	12	00:00	-1	53	-1,3	-3,5	0,9
2020	1	13	00:00	-1	53	-6,4	-12,9	-3,1
2020	1	14	00:00	9,7	52	-2,8	-9	-0,7
2020	1	15	00:00	-1	63	0,2	-0,7	0,6
2020	1	16	00:00	0,4	62	-3,9	-5,2	0,1
2020	1	17	00:00	2	62	-5,2	-8,4	-0,7

data point

label

data point, features and label are design choices!

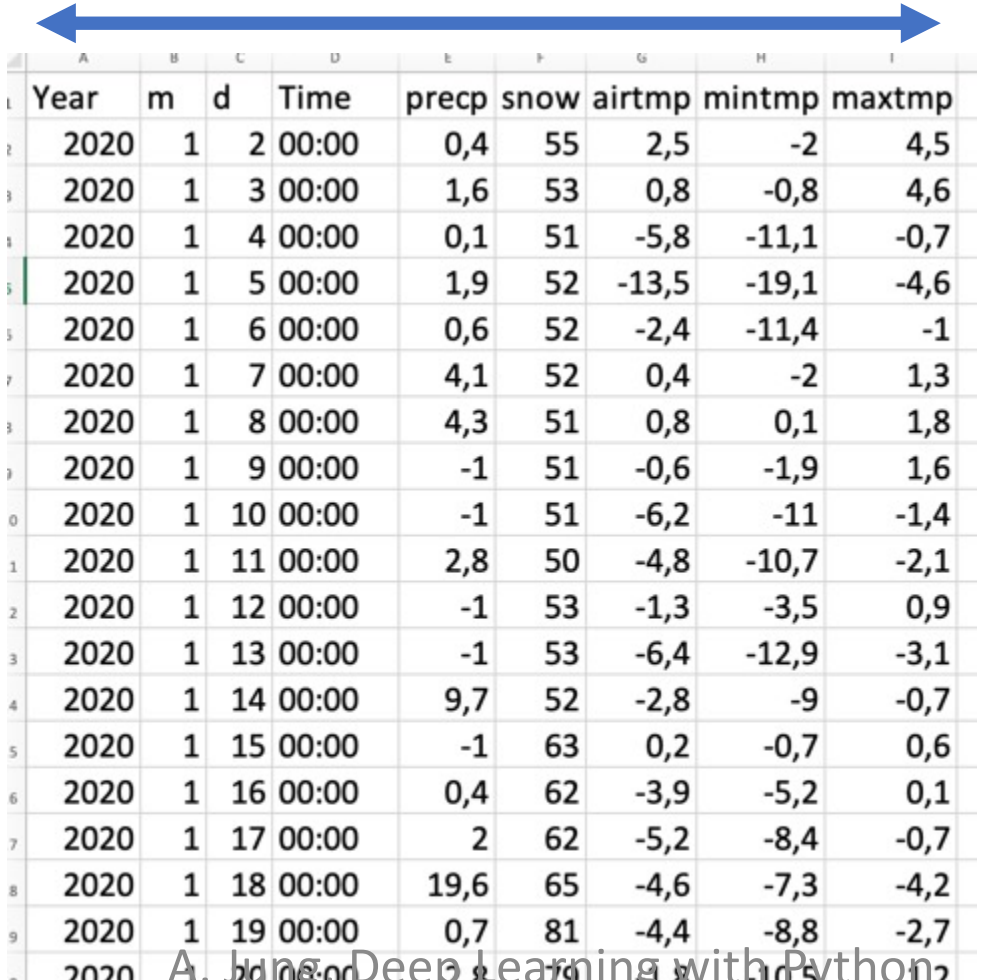
```
newdataset= somedata[somedata['date'] == '2021-06-01'] ;
print(newdataset)
```

```
          date   time  temperature
0   2021-06-01  00:00          6.2
1   2021-06-01  01:00          6.4
2   2021-06-01  02:00          6.4
3   2021-06-01  03:00          6.8
4   2021-06-01  04:00          7.1
5   2021-06-01  05:00          7.6
6   2021-06-01  06:00          7.5
7   2021-06-01  07:00          8.1
8   2021-06-01  08:00         10.3
9   2021-06-01  09:00         12.8
10  2021-06-01  10:00         15.0
11  2021-06-01  11:00         14.1
12  2021-06-01  12:00         16.5
13  2021-06-01  13:00         13.6
14  2021-06-01  14:00         14.2
15  2021-06-01  15:00         13.3
16  2021-06-01  16:00         14.5
17  2021-06-01  17:00         13.8
```

Key Parameters of a Data Set

number n of features

number m of data points "sample size"



Year	m	d	Time	precp	snow	airtmp	mintmp	maxtmp
2020	1	2	00:00	0,4	55	2,5	-2	4,5
2020	1	3	00:00	1,6	53	0,8	-0,8	4,6
2020	1	4	00:00	0,1	51	-5,8	-11,1	-0,7
2020	1	5	00:00	1,9	52	-13,5	-19,1	-4,6
2020	1	6	00:00	0,6	52	-2,4	-11,4	-1
2020	1	7	00:00	4,1	52	0,4	-2	1,3
2020	1	8	00:00	4,3	51	0,8	0,1	1,8
2020	1	9	00:00	-1	51	-0,6	-1,9	1,6
2020	1	10	00:00	-1	51	-6,2	-11	-1,4
2020	1	11	00:00	2,8	50	-4,8	-10,7	-2,1
2020	1	12	00:00	-1	53	-1,3	-3,5	0,9
2020	1	13	00:00	-1	53	-6,4	-12,9	-3,1
2020	1	14	00:00	9,7	52	-2,8	-9	-0,7
2020	1	15	00:00	-1	63	0,2	-0,7	0,6
2020	1	16	00:00	0,4	62	-3,9	-5,2	0,1
2020	1	17	00:00	2	62	-5,2	-8,4	-0,7
2020	1	18	00:00	19,6	65	-4,6	-7,3	-4,2
2020	1	19	00:00	0,7	81	-4,4	-8,8	-2,7
2020	1	20	00:00	2,8	79	-3,8	-10,5	-1,2

Feature Deluge.

modern information technology provides huge number of raw features

- smartphones
- webcams
- social networks
- smart watch
- ….

use only most relevant features but not fewer.

missing relevant features bad for accuracy

using many irrelevant features wastes computation and might result in overfitting

```
newdataset= somedata[somedata['date'] == '2021-06-01'] ;
print(newdataset)
```

```
          date   time  temperature
0   2021-06-01  00:00          6.2
1   2021-06-01  01:00          6.4
2   2021-06-01  02:00          6.4
3   2021-06-01  03:00          6.8
4   2021-06-01  04:00          7.1
5   2021-06-01  05:00          7.6
6   2021-06-01  06:00          7.5
7   2021-06-01  07:00          8.1
8   2021-06-01  08:00         10.3
9   2021-06-01  09:00         12.8
10  2021-06-01  10:00         15.0
11  2021-06-01  11:00         14.1
12  2021-06-01  12:00         16.5
13  2021-06-01  13:00         13.6
14  2021-06-01  14:00         14.2
15  2021-06-01  15:00         13.3
16  2021-06-01  16:00         14.5
17  2021-06-01  17:00         13.8
```

data point = some day at FMI station

feature = nr of hourly observations

want to predict maximum daytime temperature

missing relevant features bad for accuracy

```
newdataset= somedata[somedata['date'] == '2021-06-01'] ;
print(newdataset)
```

```
          date   time  temperature
0   2021-06-01  00:00          6.2
1   2021-06-01  01:00          6.4
2   2021-06-01  02:00          6.4
3   2021-06-01  03:00          6.8
4   2021-06-01  04:00          7.1
5   2021-06-01  05:00          7.6
6   2021-06-01  06:00          7.5
7   2021-06-01  07:00          8.1
8   2021-06-01  08:00         10.3
9   2021-06-01  09:00         12.8
10  2021-06-01  10:00         15.0
11  2021-06-01  11:00         14.1
12  2021-06-01  12:00         16.5
13  2021-06-01  13:00         13.6
14  2021-06-01  14:00         14.2
15  2021-06-01  15:00         13.3
16  2021-06-01  16:00         14.5
17  2021-06-01  17:00         13.8
```

data point = some day at FMI station

feature = hourly temp. 00:00 – 15:00

want to predict temp at 16:00

using irrelevant features wastes comp. resources

- regression: labels are numbers (temperature, distance, duration, …)
- classification: labels are discrete-valued; represents category such as “Cat” vs. “No Cat”
- reg. /class. methods use diff. loss function (see later)

Label is Design Choice!

- by choosing/defining label you define the ML problem or learning task !

- **Human agency and oversight**: ….proper oversight mechanisms need to be ensured…

https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai

Regression. Numeric Labels.

	date	time	temperature
0	2021-06-01	00:00	6.2
1	2021-06-01	01:00	6.4
2	2021-06-01	02:00	6.4
3	2021-06-01	03:00	6.8
4	2021-06-01	04:00	7.1
5	2021-06-01	05:00	7.6
6	2021-06-01	06:00	7.5
7	2021-06-01	07:00	8.1
8	2021-06-01	08:00	10.3
9	2021-06-01	09:00	12.8
10	2021-06-01	10:00	15.0
11	2021-06-01	11:00	14.1
12	2021-06-01	12:00	16.5
13	2021-06-01	13:00	13.6
14	2021-06-01	14:00	14.2
15	2021-06-01	15:00	13.3
16	2021-06-01	16:00	14.5
17	2021-06-01	17:00	13.8

datapoint
“2021-06-01 at some FMI station”

label = tmp at 15:00

Binary Classification.

	date	time	temperature
0	2021-06-01	00:00	6.2
1	2021-06-01	01:00	6.4
2	2021-06-01	02:00	6.4
3	2021-06-01	03:00	6.8
4	2021-06-01	04:00	7.1
5	2021-06-01	05:00	7.6
6	2021-06-01	06:00	7.5
7	2021-06-01	07:00	8.1
8	2021-06-01	08:00	10.3
9	2021-06-01	09:00	12.8
10	2021-06-01	10:00	15.0
11	2021-06-01	11:00	14.1
12	2021-06-01	12:00	16.5
13	2021-06-01	13:00	13.6
14	2021-06-01	14:00	14.2
15	2021-06-01	15:00	13.3
16	2021-06-01	16:00	14.5
17	2021-06-01	17:00	13.8

datapoint
“2021-06-01 at some FMI station”

label =

- “hot” if tmp at 15:00 > 10
- “cold” if … <= 10

Multi-Class Classification

	date	time	temperature
0	2021-06-01	00:00	6.2
1	2021-06-01	01:00	6.4
2	2021-06-01	02:00	6.4
3	2021-06-01	03:00	6.8
4	2021-06-01	04:00	7.1
5	2021-06-01	05:00	7.6
6	2021-06-01	06:00	7.5
7	2021-06-01	07:00	8.1
8	2021-06-01	08:00	10.3
9	2021-06-01	09:00	12.8
10	2021-06-01	10:00	15.0
11	2021-06-01	11:00	14.1
12	2021-06-01	12:00	16.5
13	2021-06-01	13:00	13.6
14	2021-06-01	14:00	14.2
15	2021-06-01	15:00	13.3
16	2021-06-01	16:00	14.5
17	2021-06-01	17:00	13.8

datapoint

"2021-06-01 at some FMI station"

label =

- "nice morning" if tmp at 15:00 < 10 and tmp at 10:00 > 10
- "nice noon" if tmp at 15:00 > 10 and tmp at 10:00 < 10
- "nice day" if tmp at 15:00 > 10 and tmp at 10:00 > 10

Multi-Label Problems

- data point with several different labels
- the choice of label defines the ML task !
- special case of multi-task learning

Multi-Label Regression.

	date	time	temperature
0	2021-06-01	00:00	6.2
1	2021-06-01	01:00	6.4
2	2021-06-01	02:00	6.4
3	2021-06-01	03:00	6.8
4	2021-06-01	04:00	7.1
5	2021-06-01	05:00	7.6
6	2021-06-01	06:00	7.5
7	2021-06-01	07:00	8.1
8	2021-06-01	08:00	10.3
9	2021-06-01	09:00	12.8
10	2021-06-01	10:00	15.0
11	2021-06-01	11:00	14.1
12	2021-06-01	12:00	16.5
13	2021-06-01	13:00	13.6
14	2021-06-01	14:00	14.2
15	2021-06-01	15:00	13.3
16	2021-06-01	16:00	14.5
17	2021-06-01	17:00	13.8

datapoint
“2021-06-01 at some FMI station”

label1 = tmp at 10:00

label2= tmp at 15:00

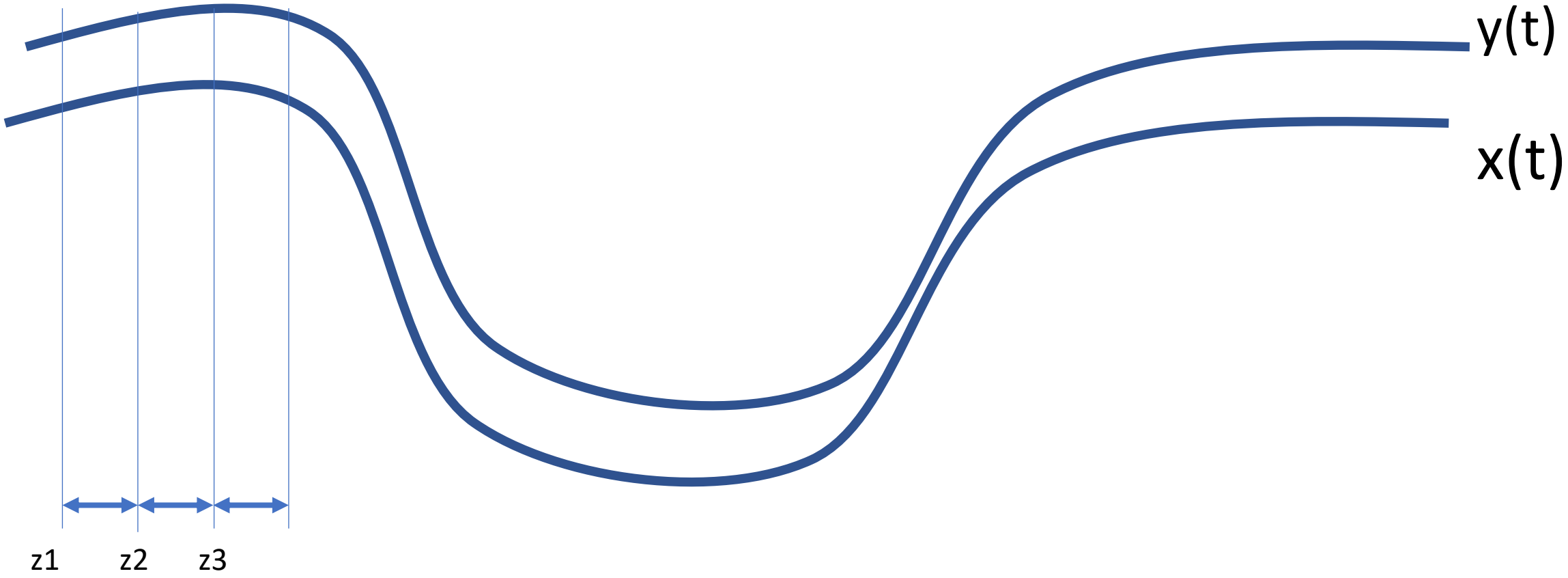
Multi-Label Classification.

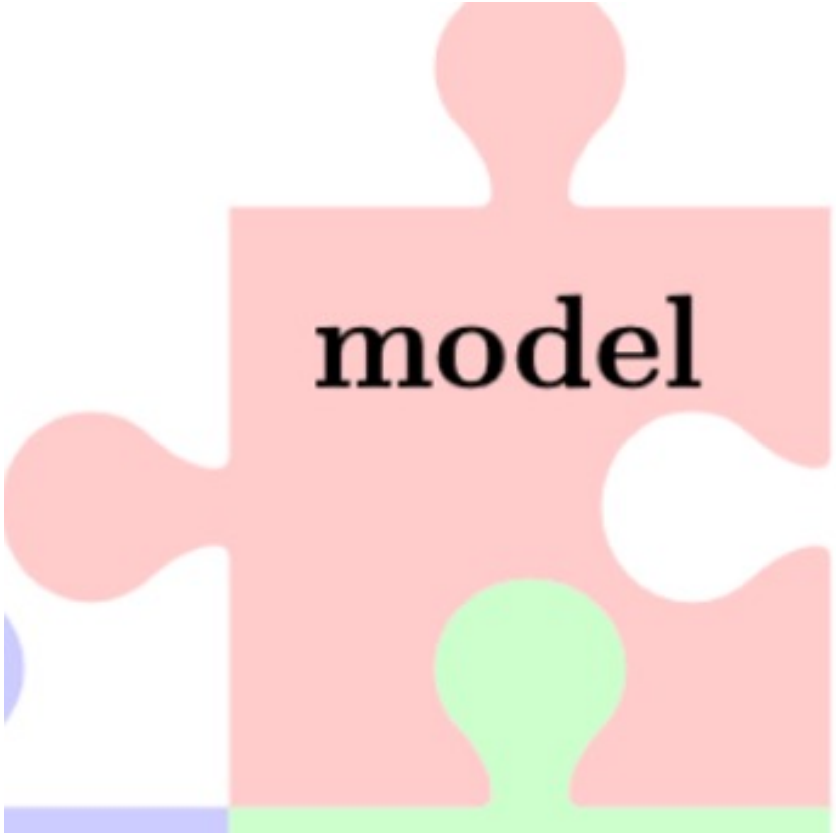
y_1= 1 or 0 if car present or not

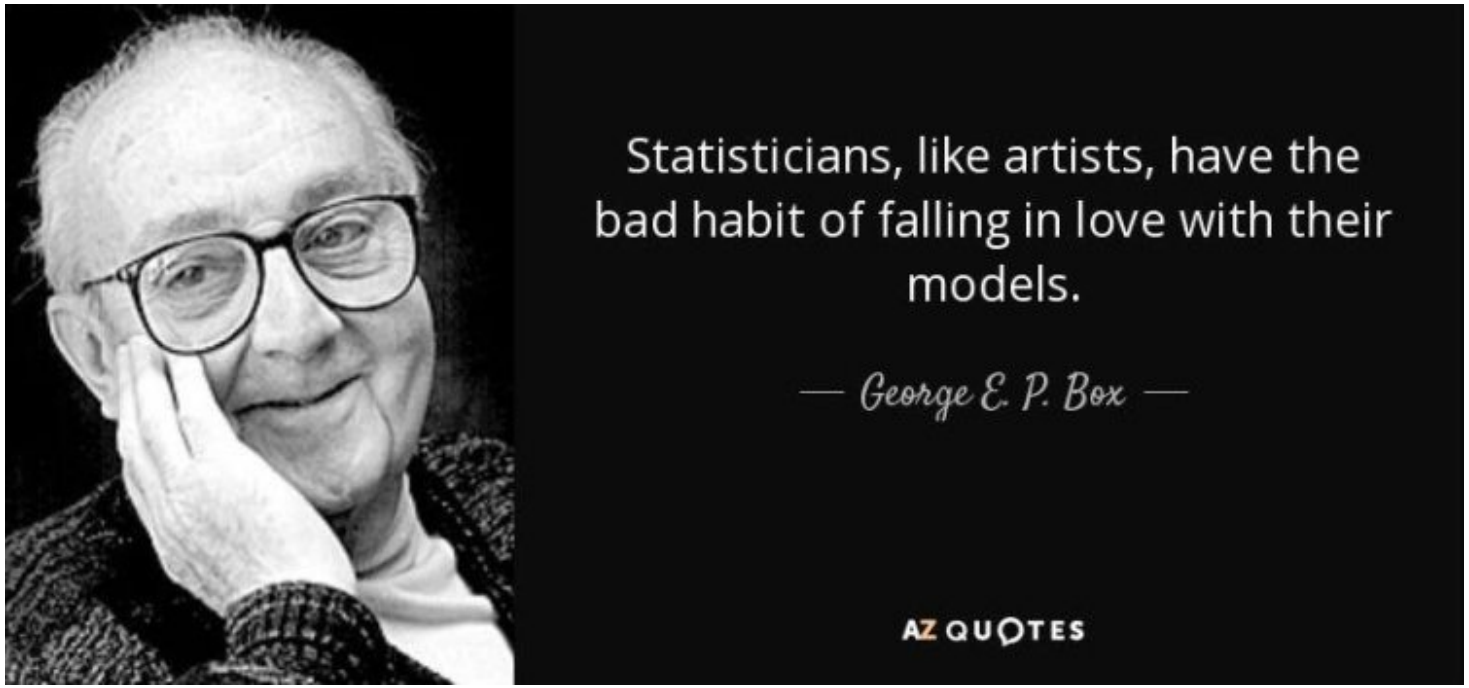
y_2= 1 or 0 if person present or not

y_3= 1 or 0 if tree present or not

Effective Data Size m



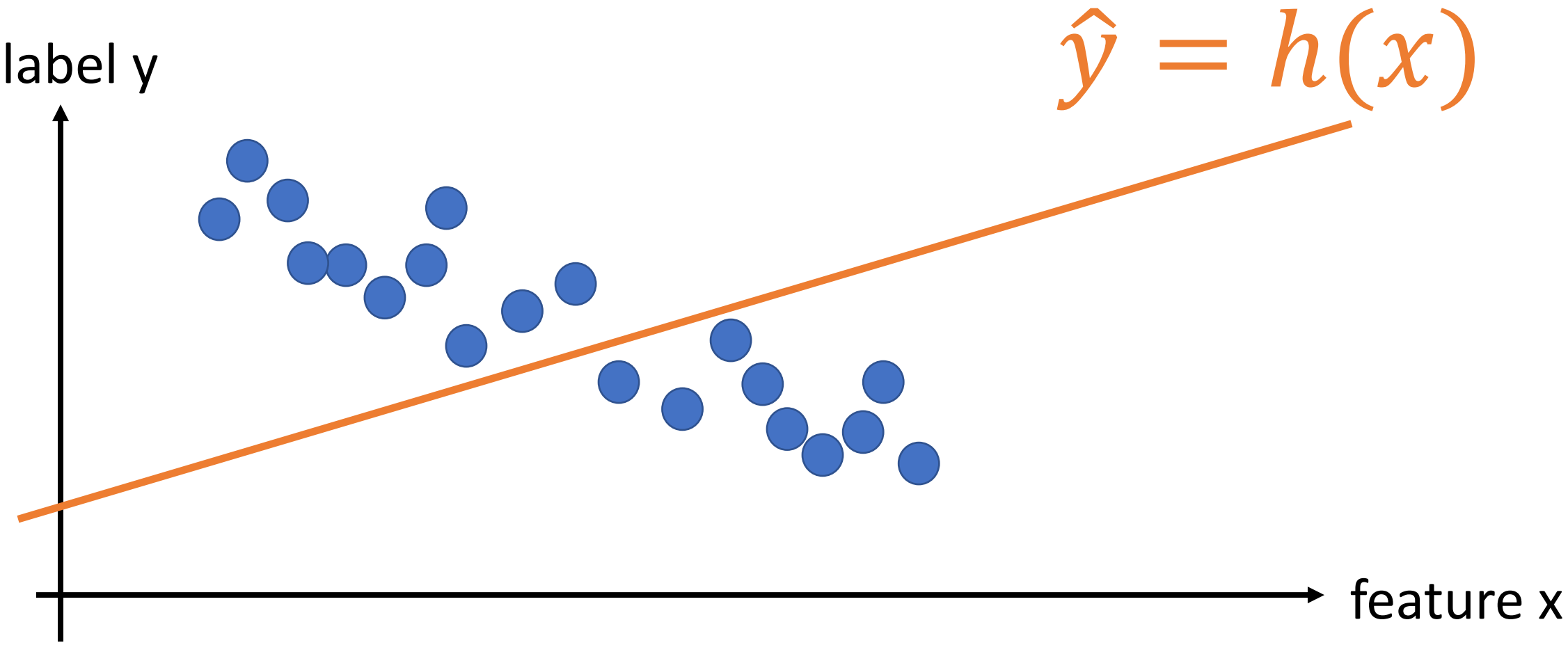

model


Statisticians, like artists, have the bad habit of falling in love with their models.
— George E. P. Box —
AZ QUOTES

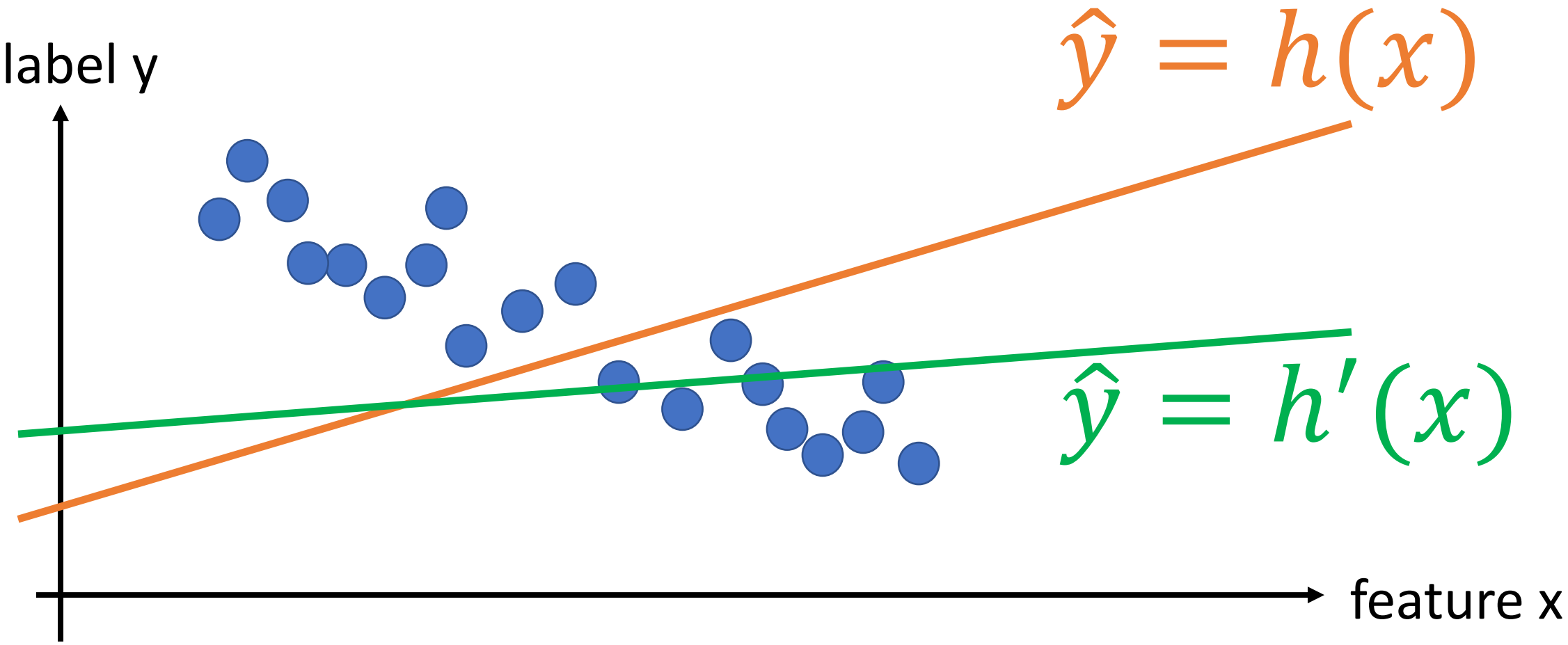
Machine Learning.

"learn to predict the label of a data point solely from its features"

A Hypothesis.

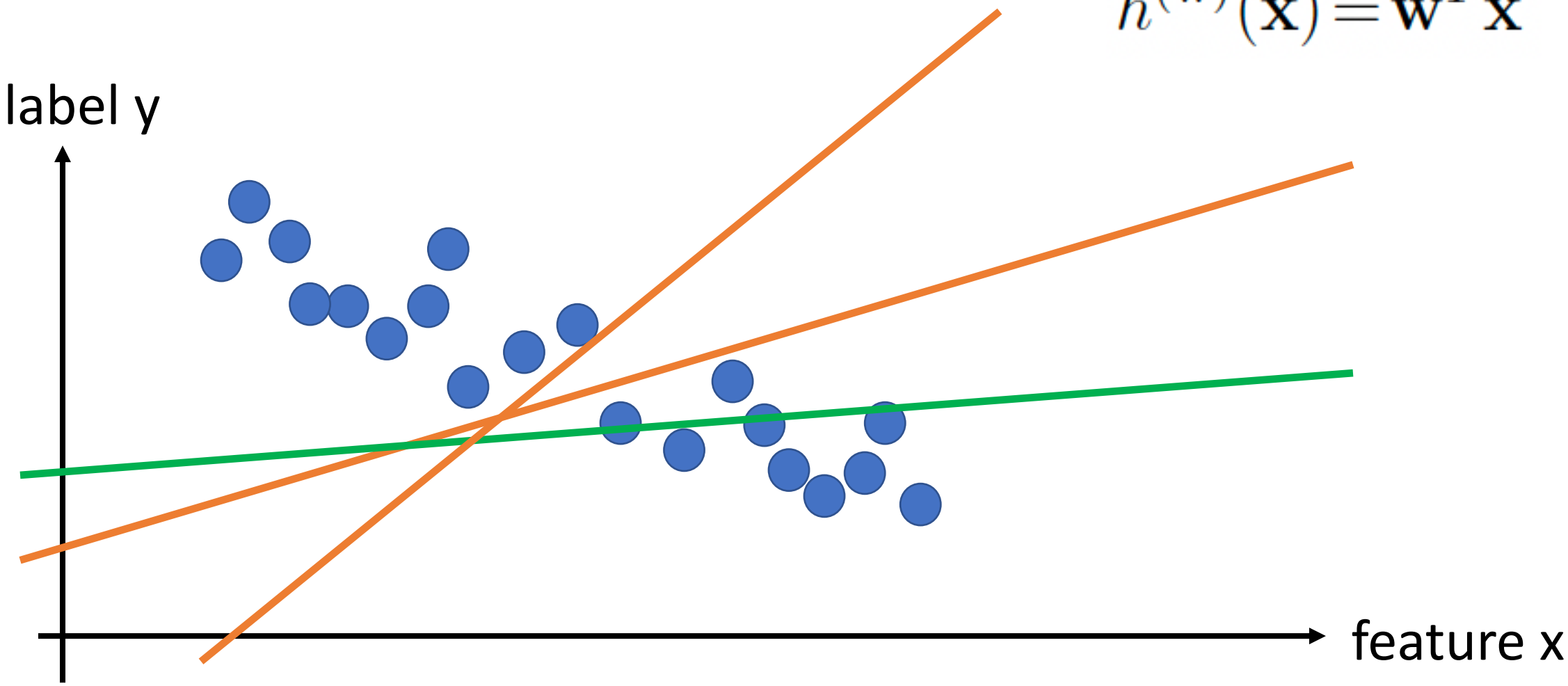


Model = Several Hypotheses.

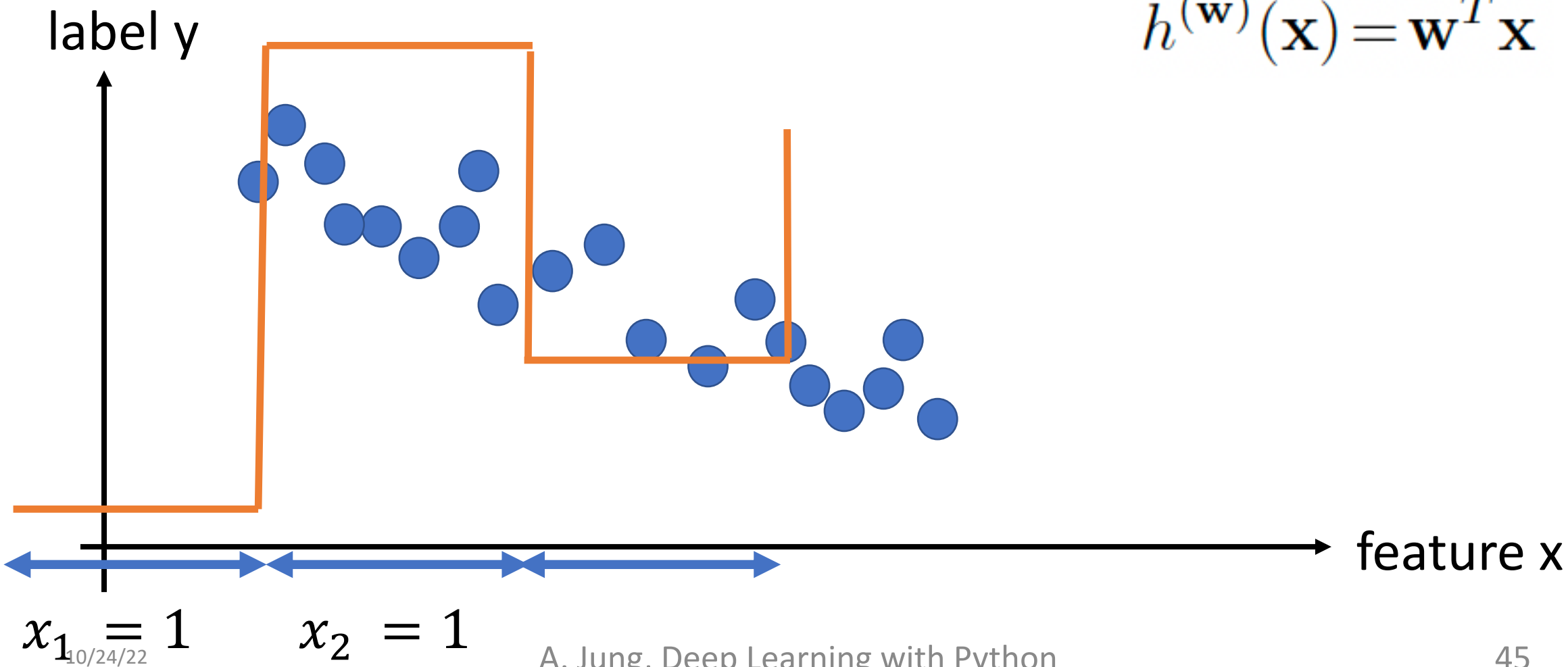


Linear Model

$$h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$



Linear Model is Versatile!



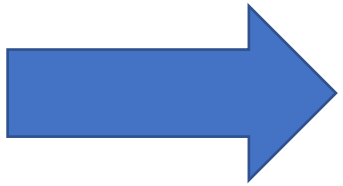
$$h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

Linear + Feature Map

feature map

$\phi(\cdot)$

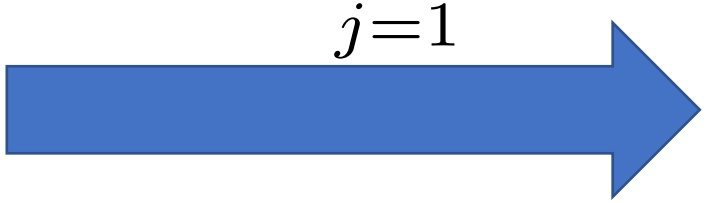
single feature x



$$\begin{pmatrix} x_1 = \phi_1(x) \\ x_2 = \phi_2(x) \\ \vdots \\ x_n = \phi_n(x) \end{pmatrix}$$

linear map

$$\mathbf{w}^T \mathbf{x} = \sum_{j=1}^{n} w_j x_j$$

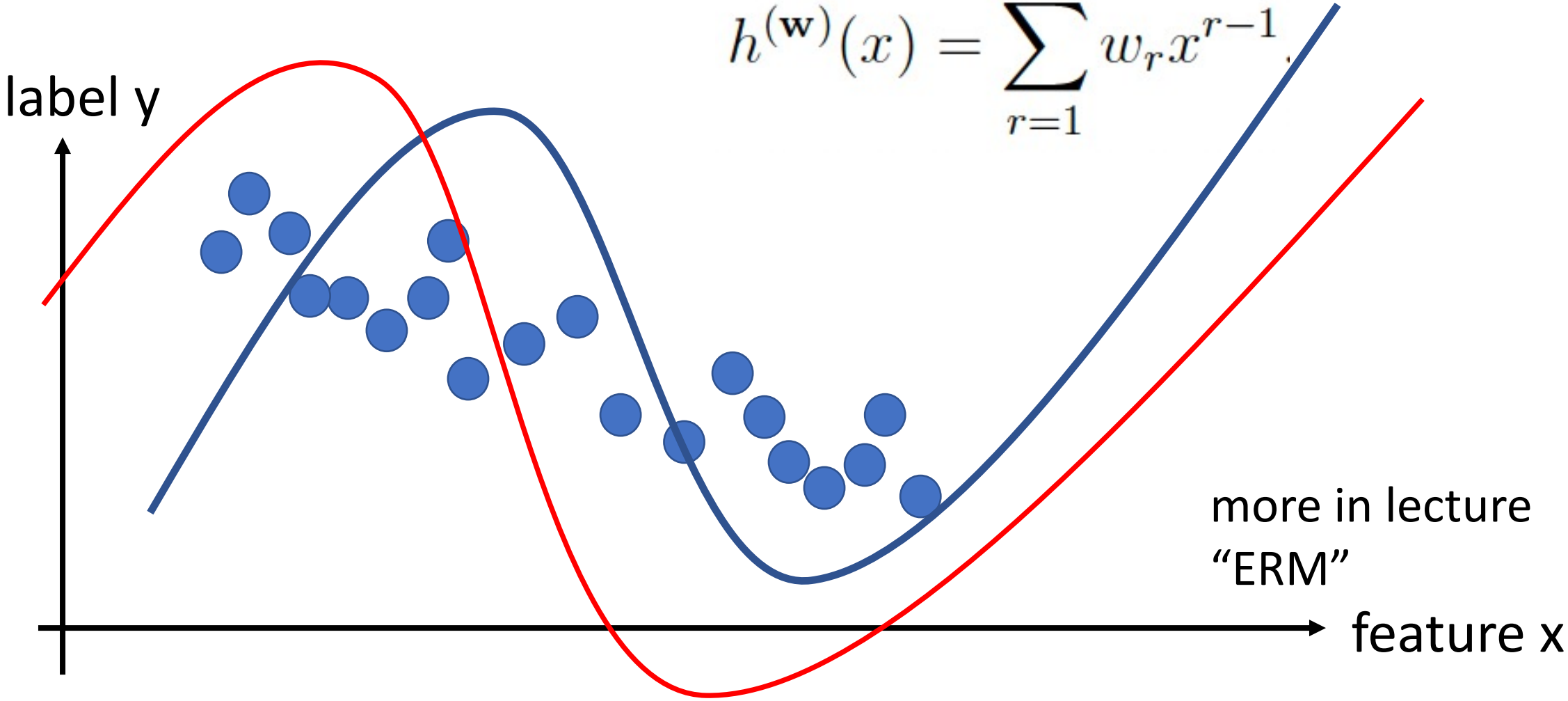


h(x)

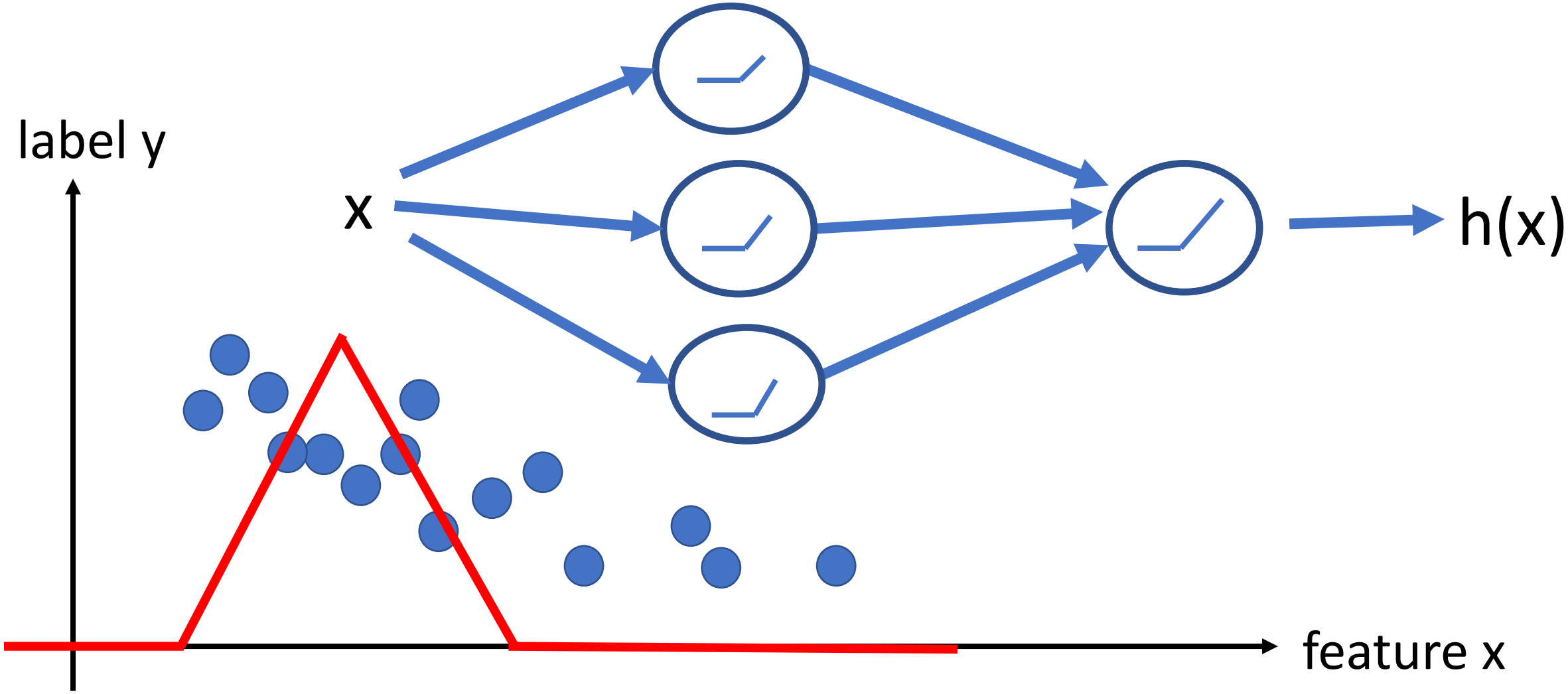
$$h(x) = \sum_{j=1}^{n} w_j \phi_j(x)$$

Polynomials

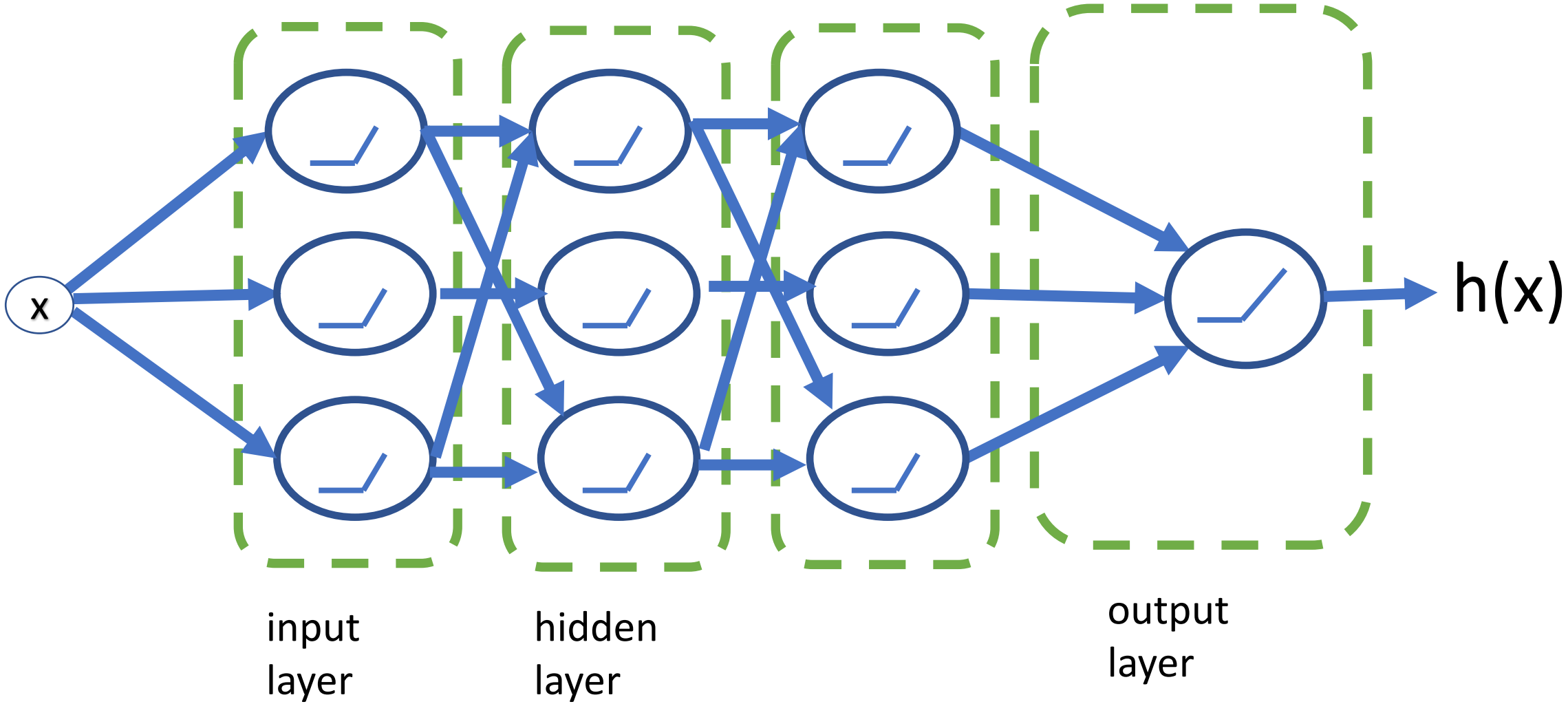
$$h^{(\mathbf{w})}(x) = \sum_{r=1}^{n} w_r x^{r-1}$$



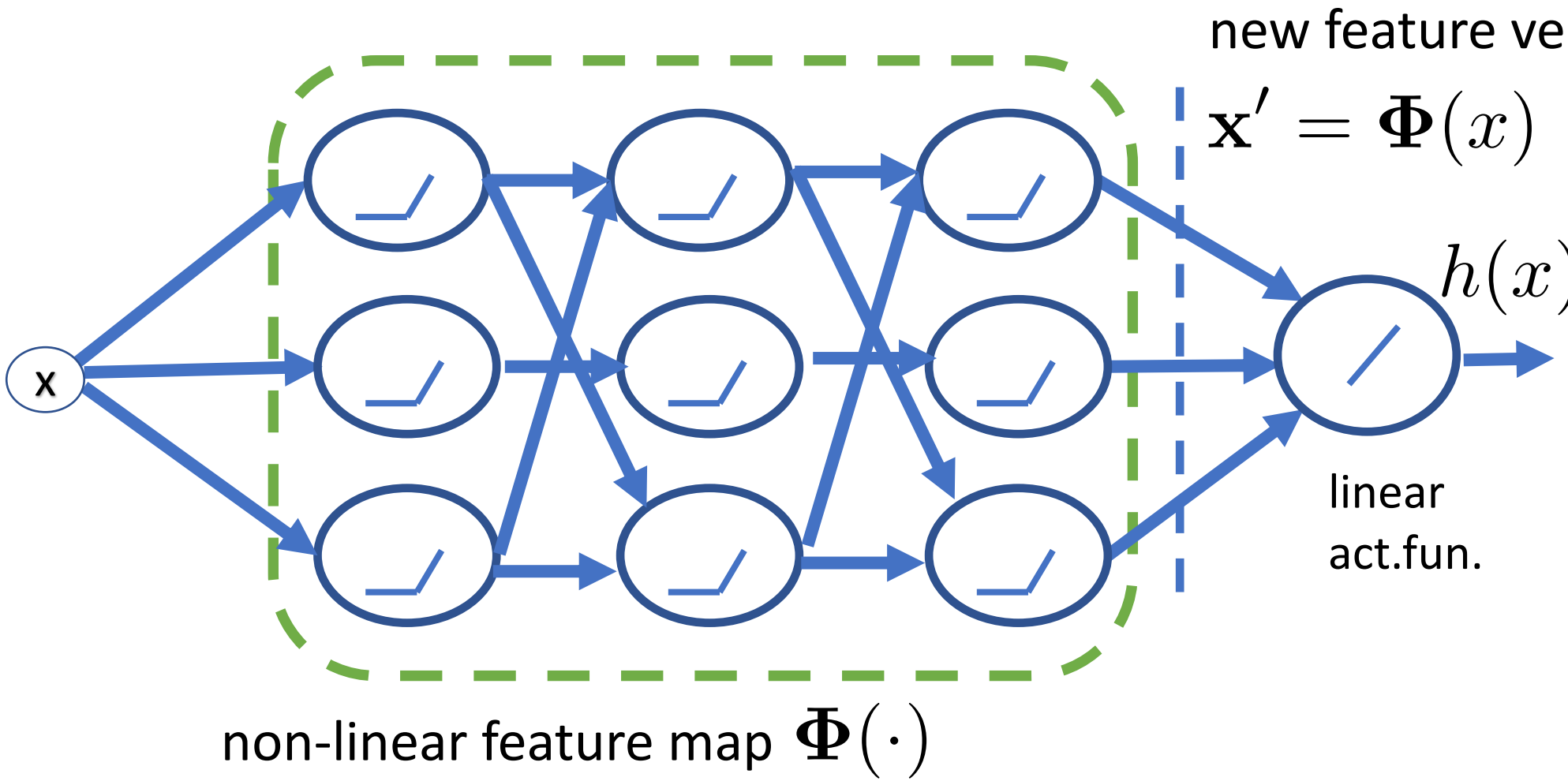
Artificial Neural Network



Deep ANN



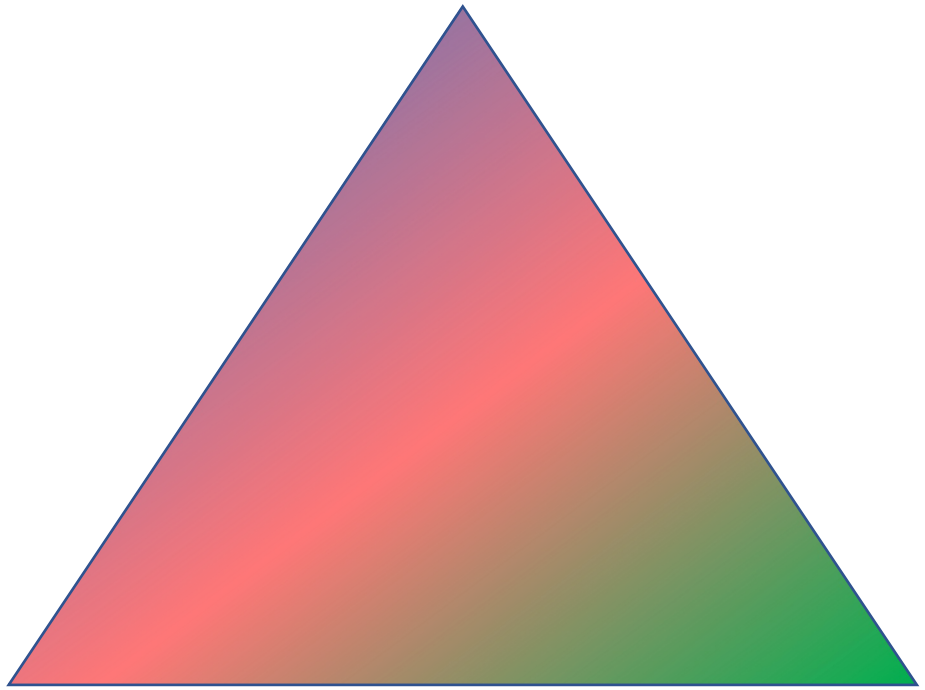
Deep Net= Feature Map+Lin.Model



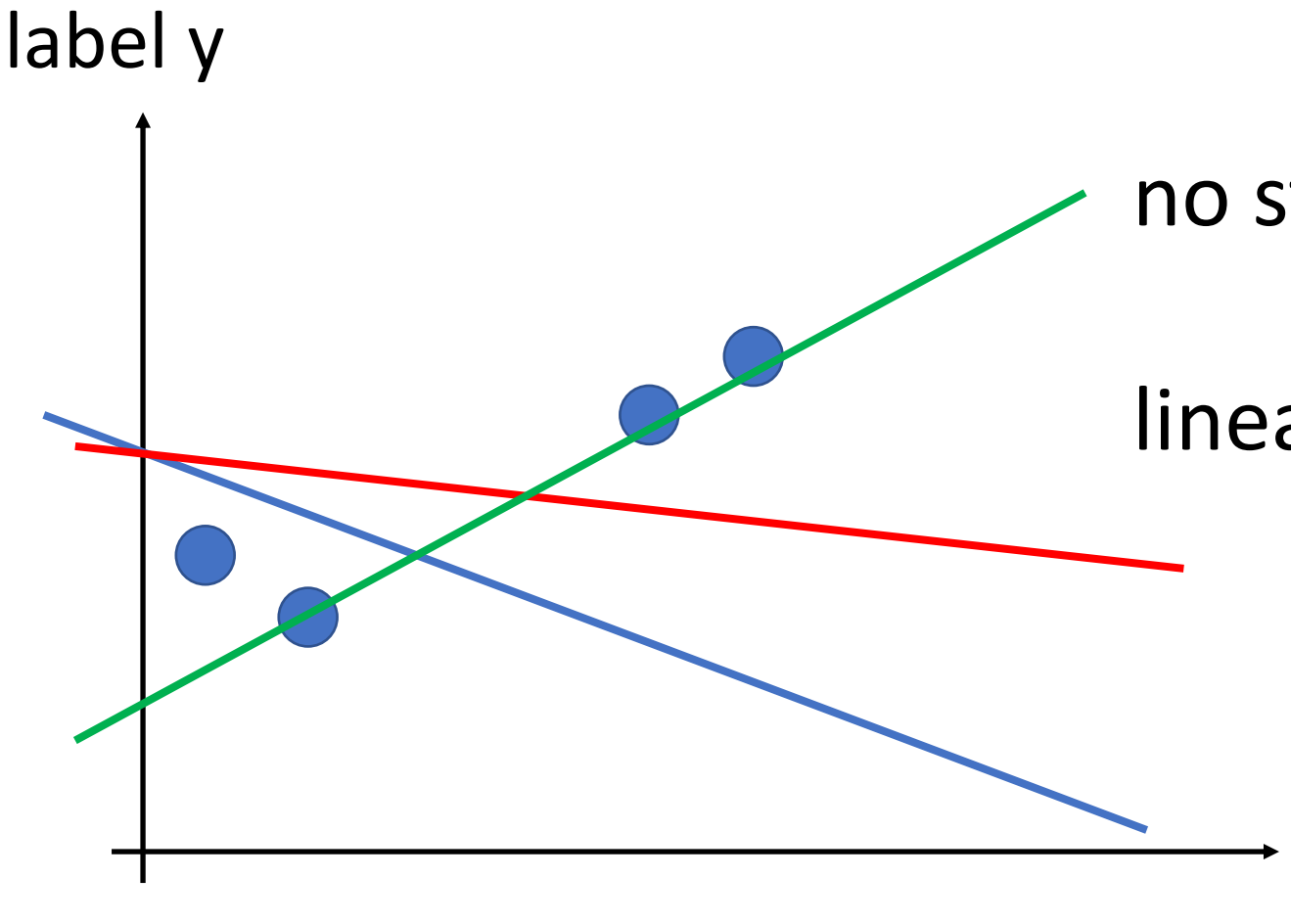
Which Model To Choose?

- large to contain a suitable hypothesis
- small to fit computational resources
- simple or interpretable

Design Choice: Model



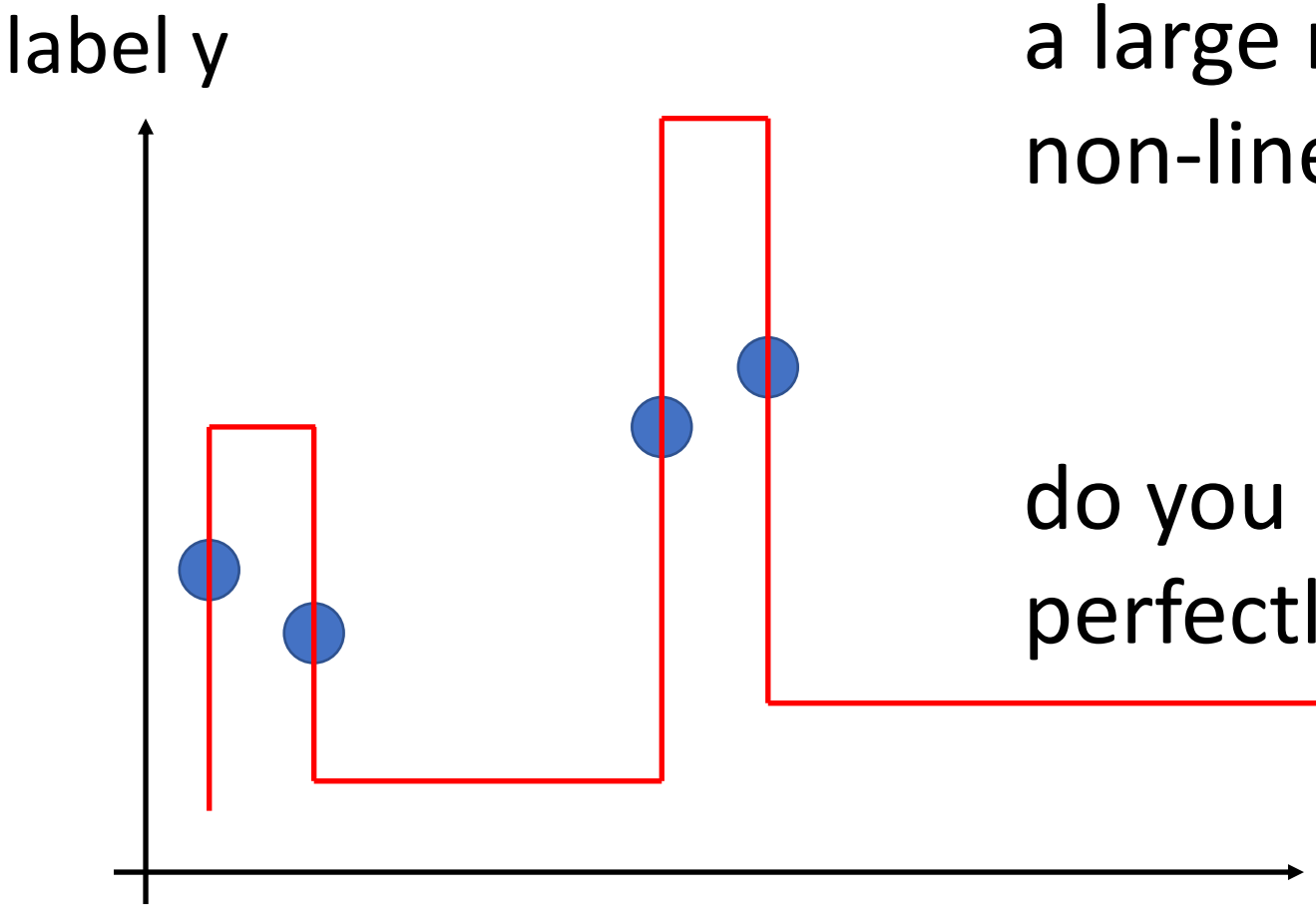
Sufficiently Large



no straight line that fits well data

linear model is too small

Sufficiently Small (Stat.)



a large model contains some very non-linear hypothesis maps

do you like this hypothesis which perfectly fits the data points ?

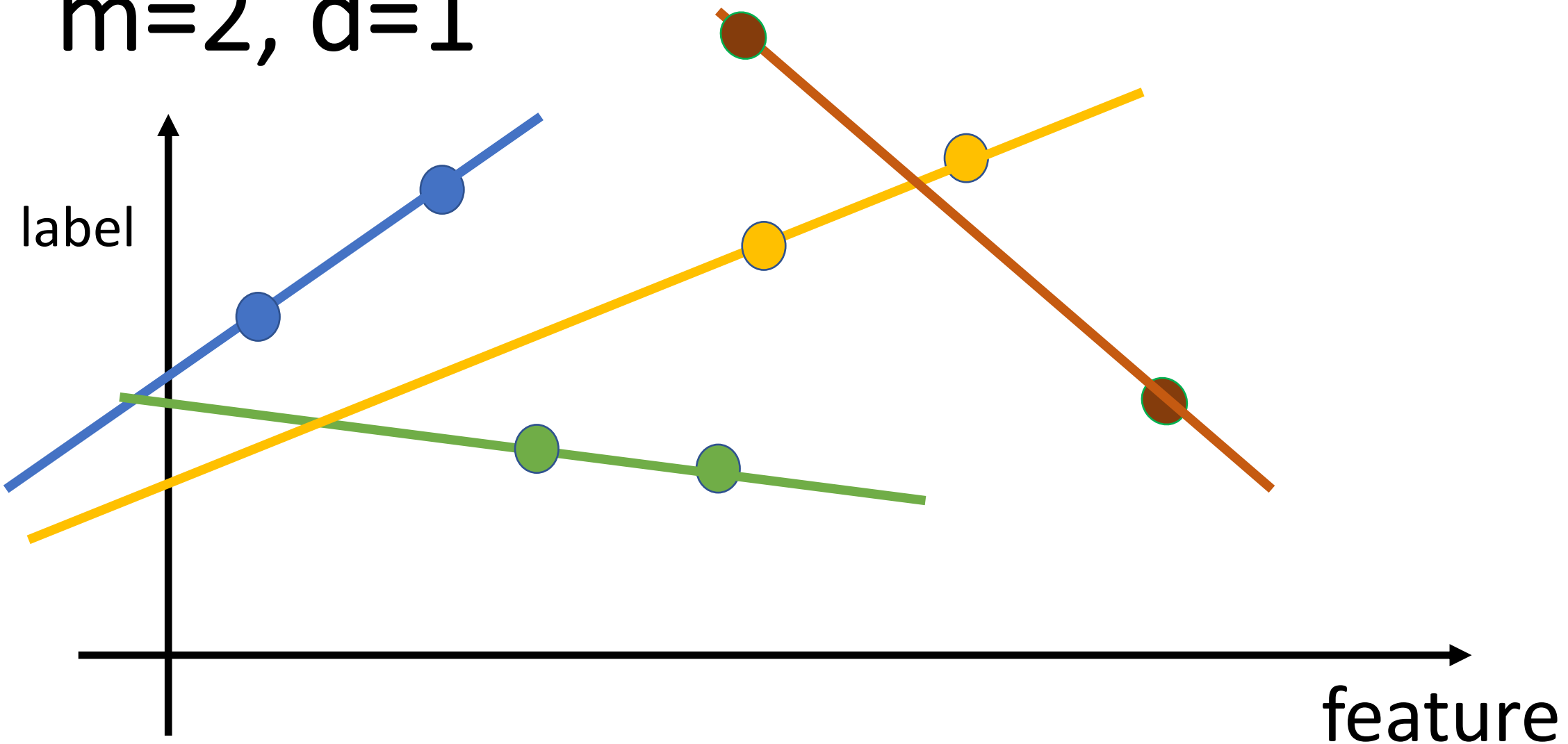
Sufficiently Small (Comput.)

- consider linear regression using n features
- fit linear model on m > n datapoints
- need to invert “n by n” matrix !

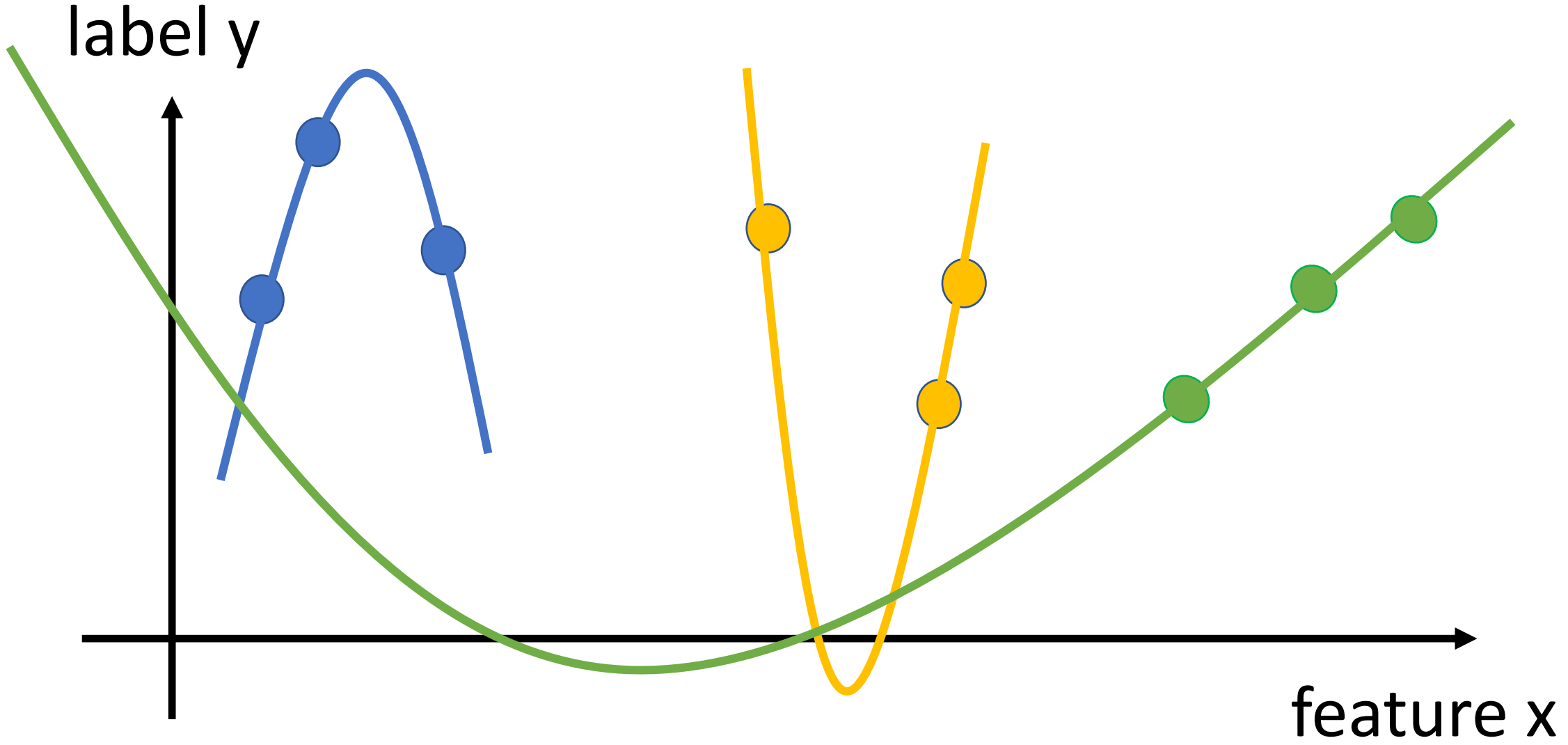
Size of a Model?

- different measures for model size
- e.g., nr. of hypotheses in the model
- what about linear model ? VC-dimension
- eff. model size $\approx$nr of datapoints being perfectly fit

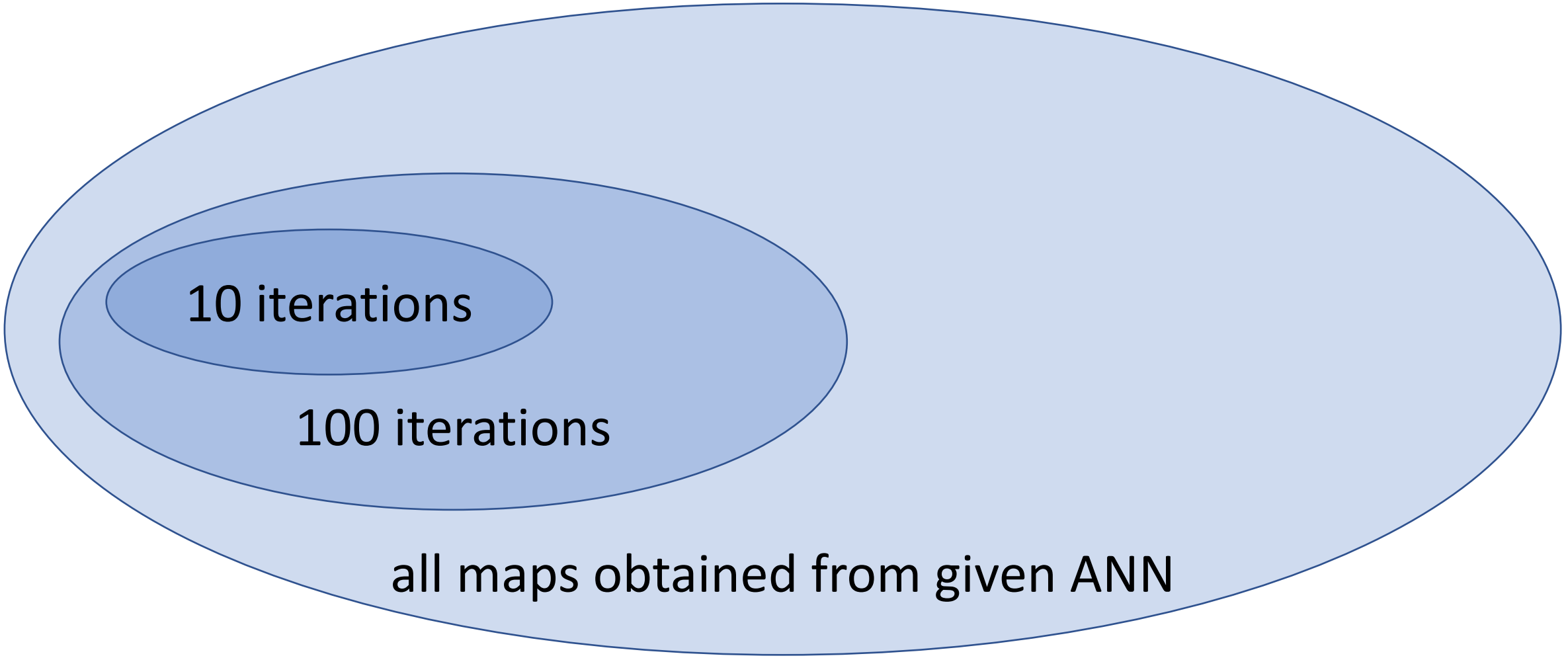
m=2, d=1



m=3, degree d=2 polynomial

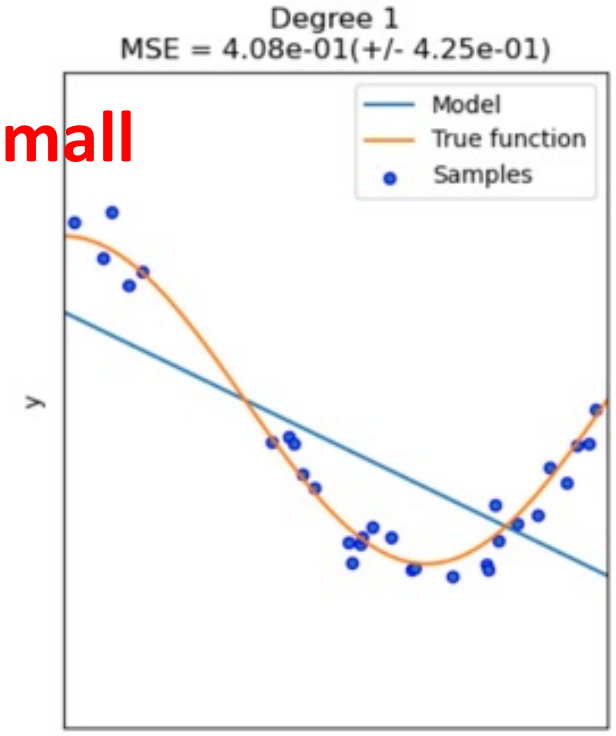


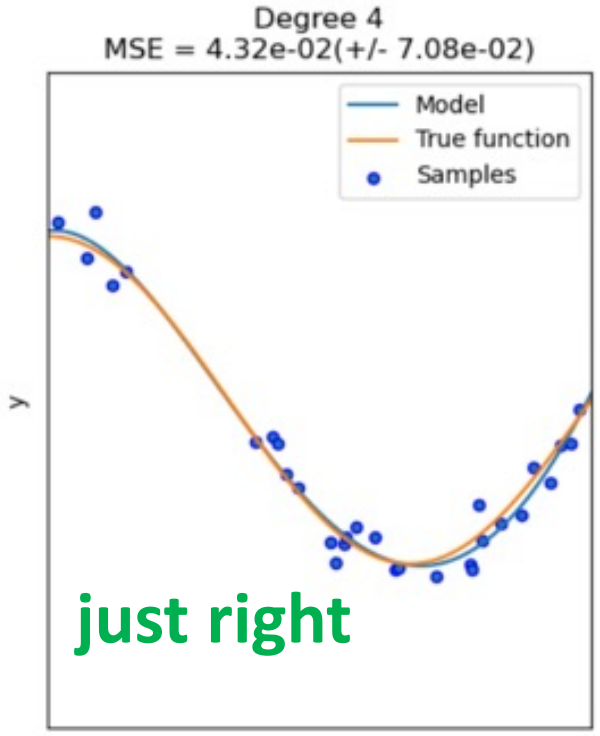
Effective Model Size d



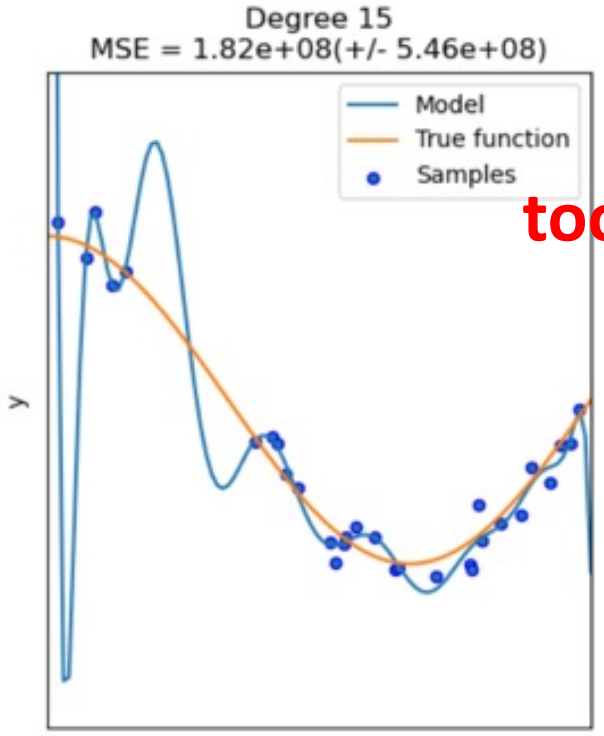
Design Choice Relations

too small





just right



too large

source: https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html

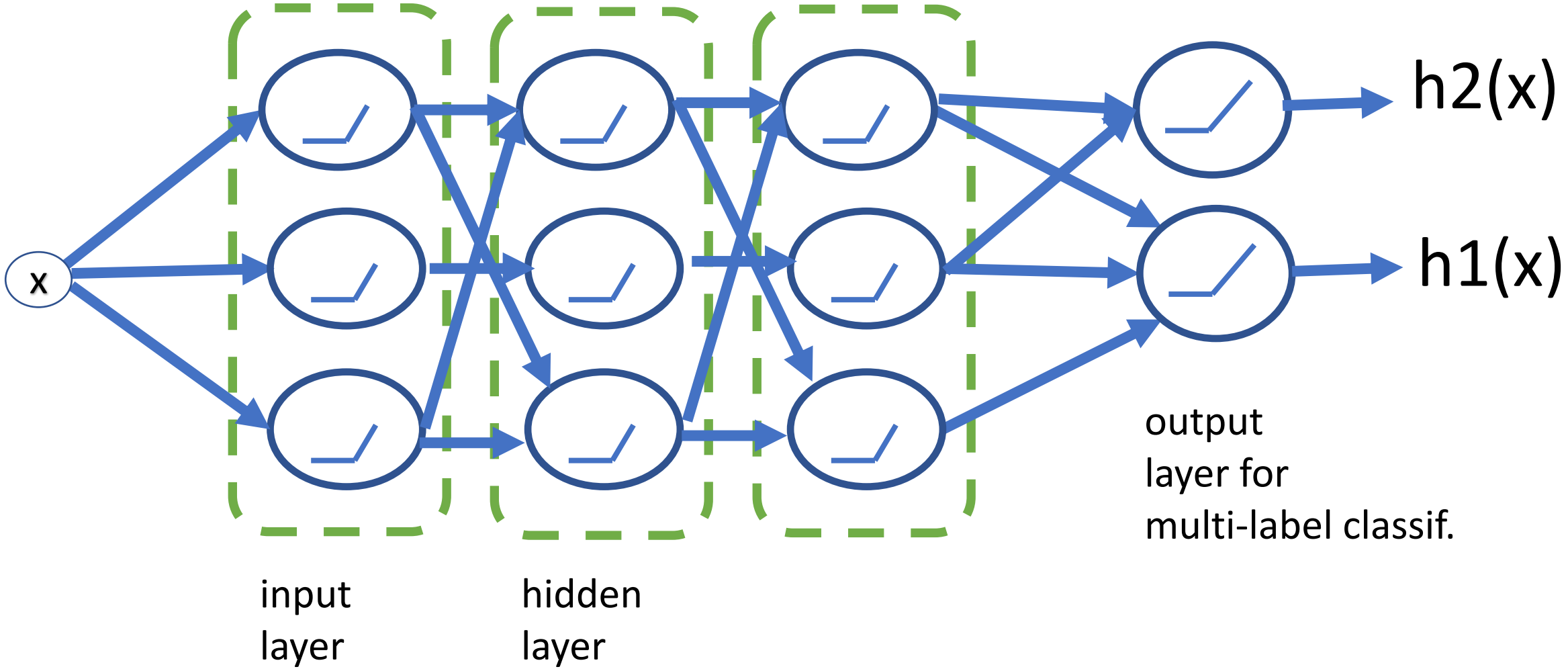
Alex' rule of thumb:

m >= 10 * d

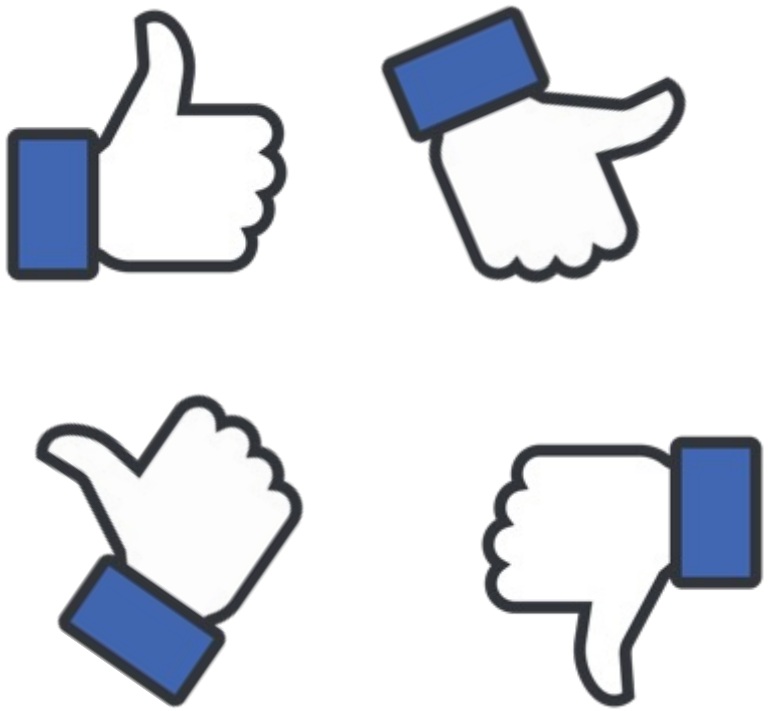
Design Choice Relations

- model must be suitable for data
- consider final layer of deep net
- for binary classif., cannot use ReLU or linear
- for regression, cannot use Sigmoid or SoftMax

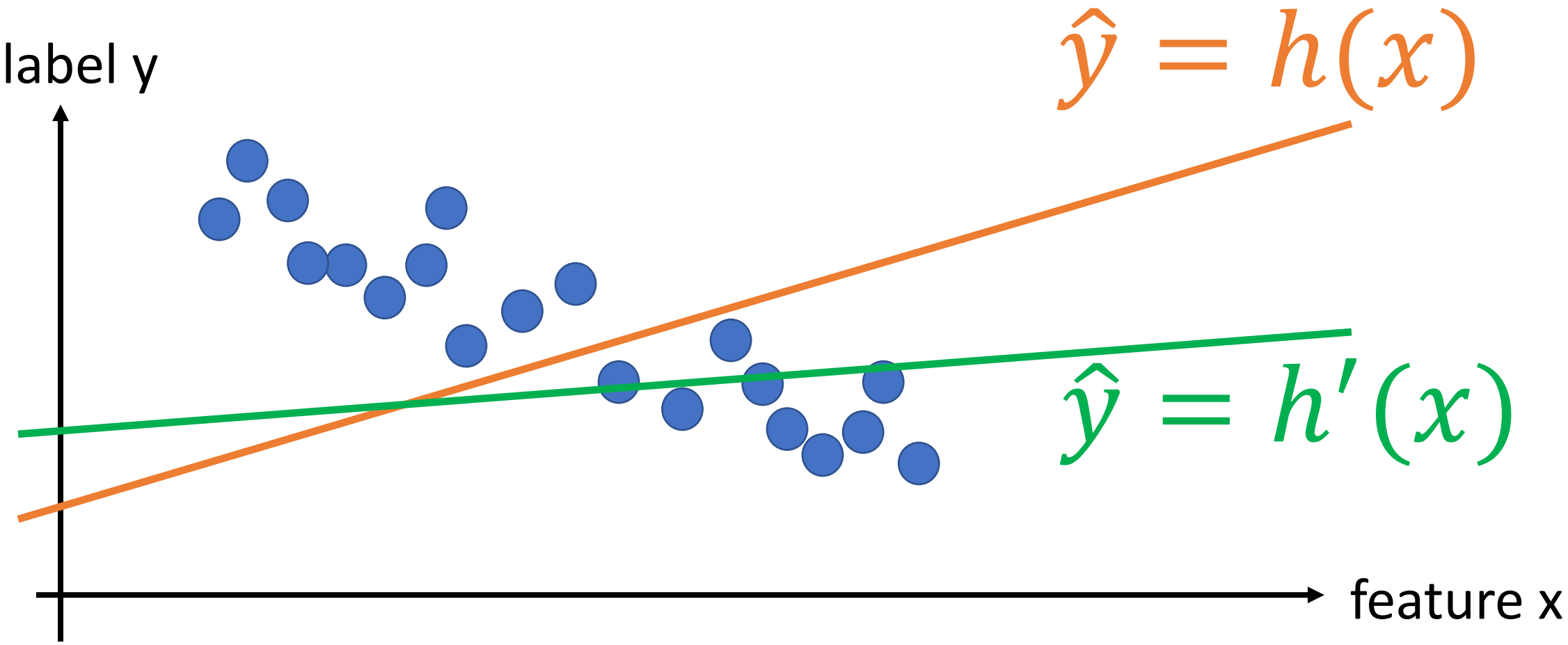
Design Choice Relations



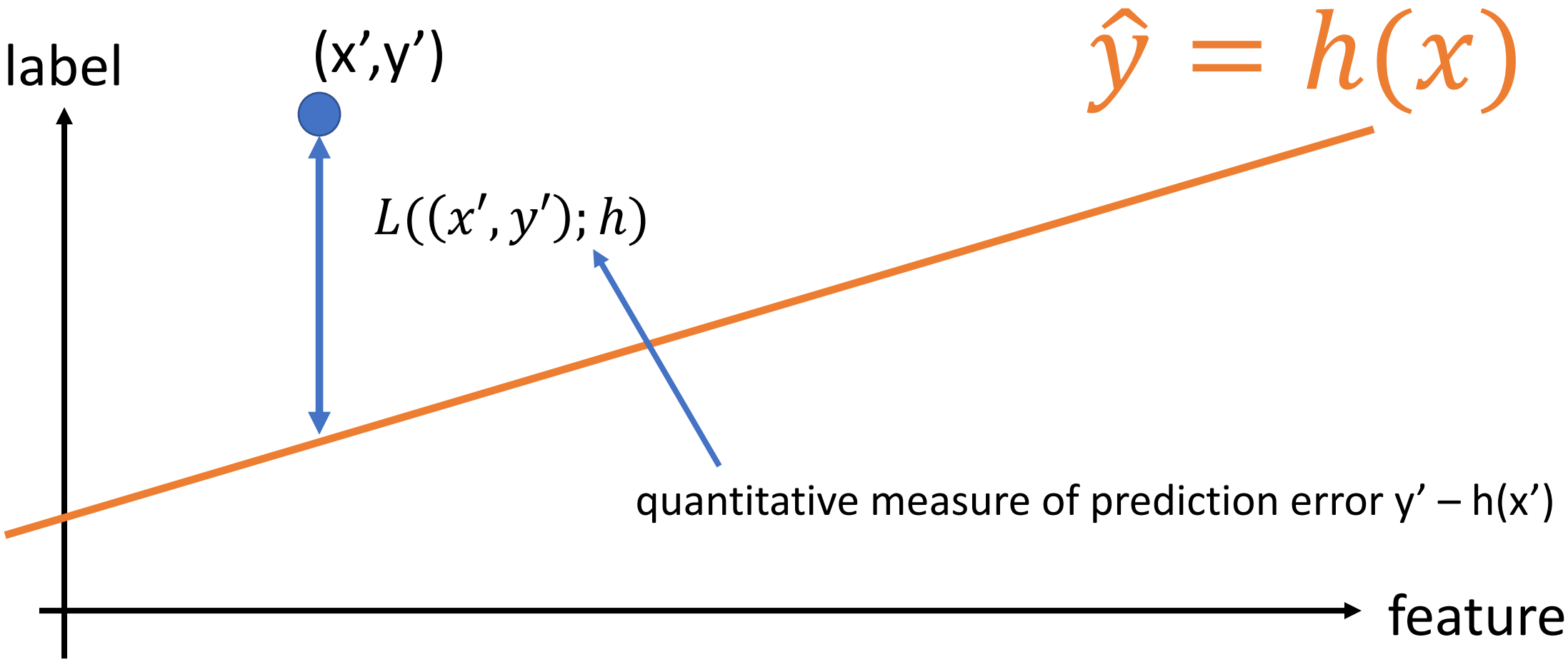

loss



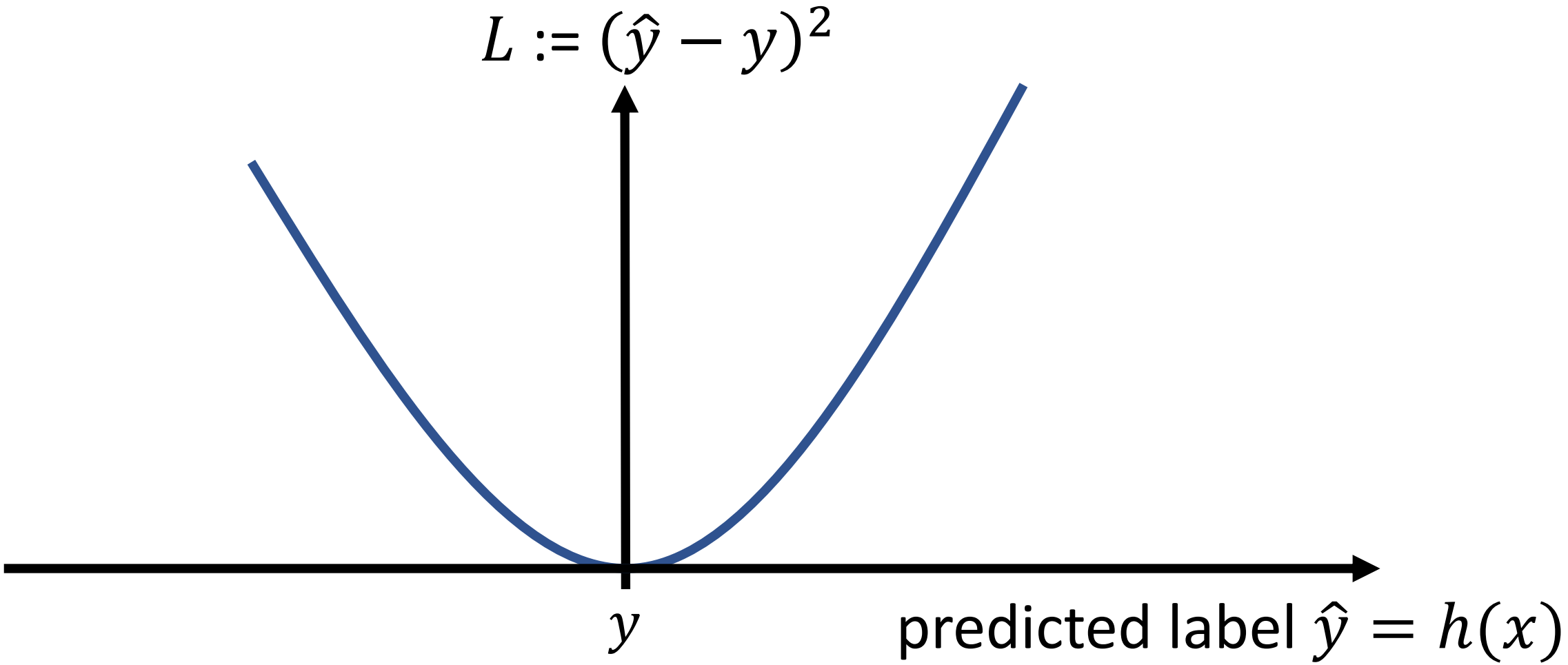
Which Hypothesis is Better?



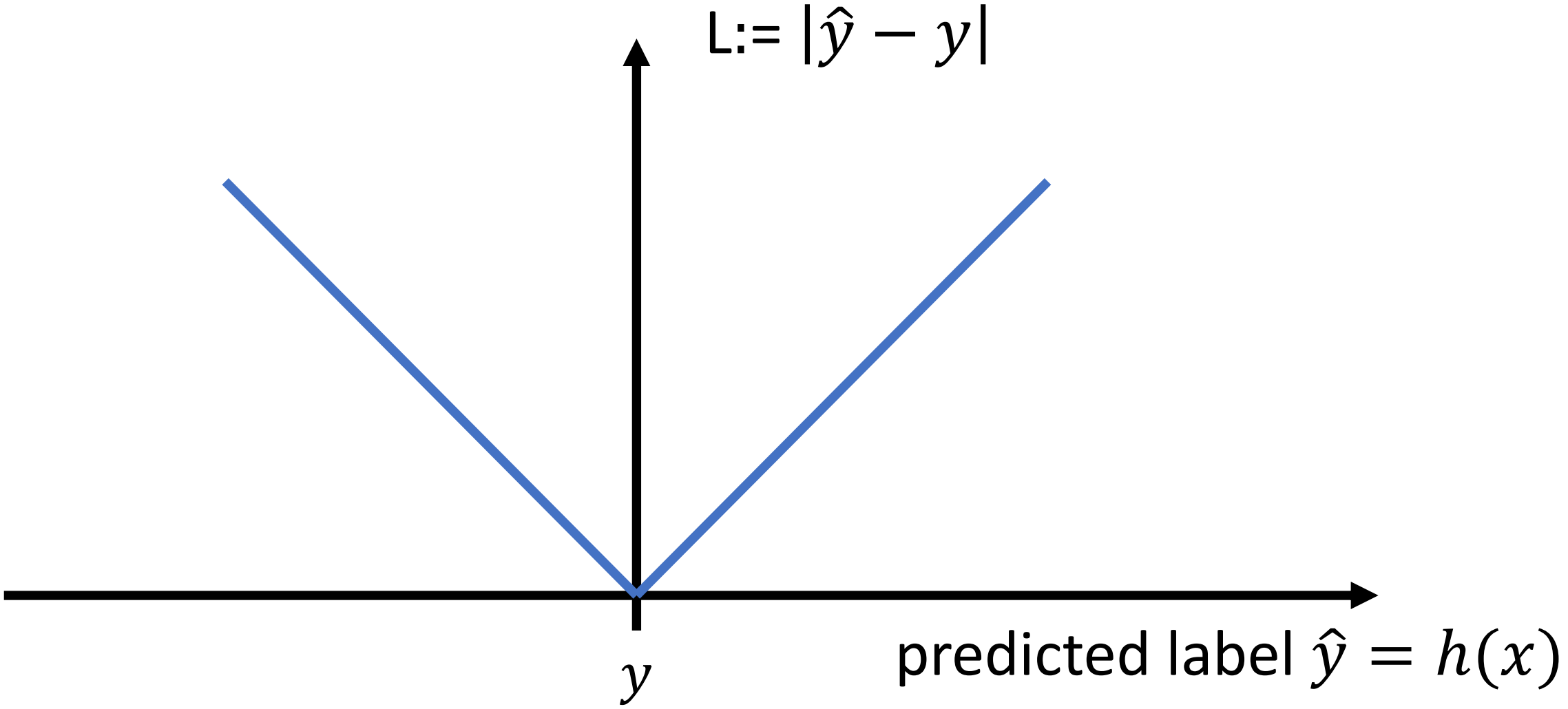
A Loss Function



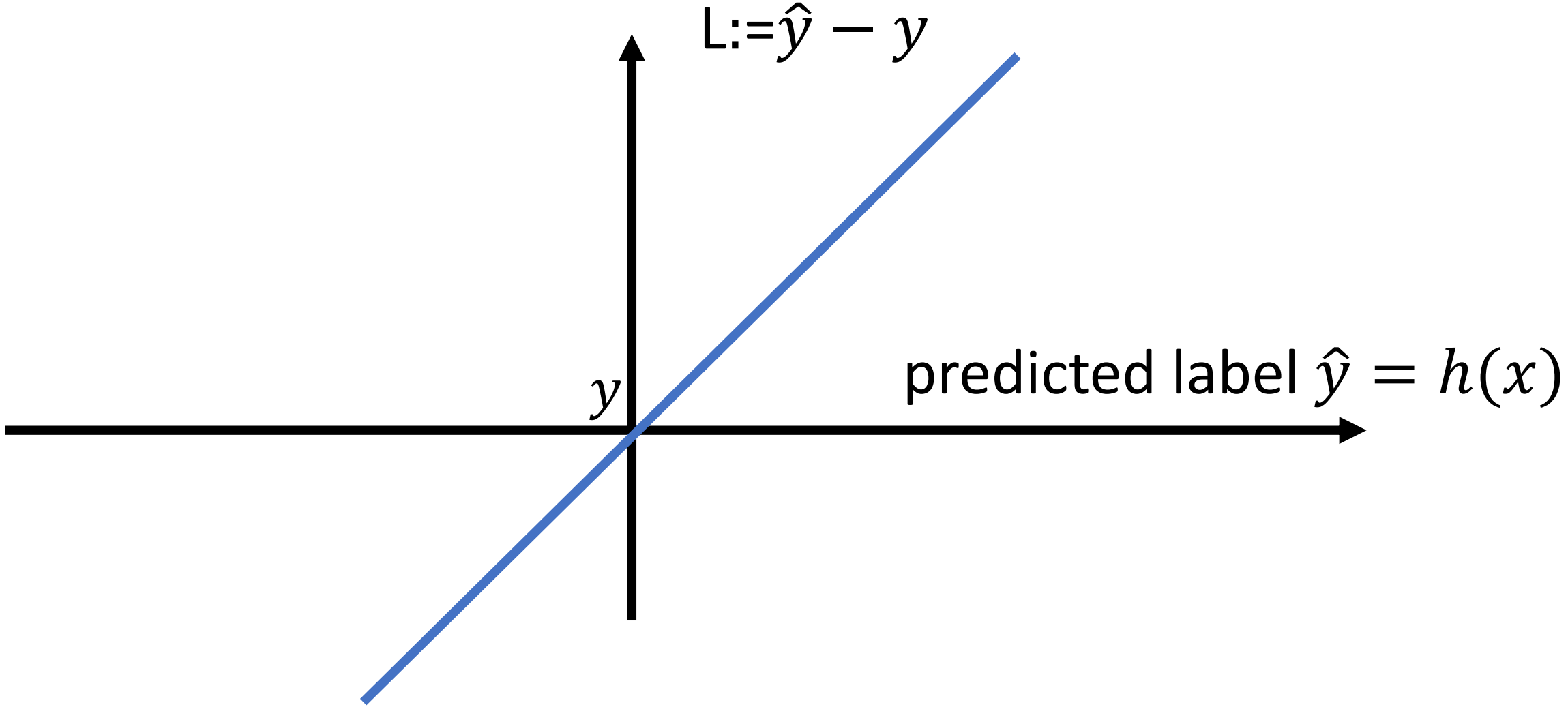
The Squared Error Loss



The Absolute Error Loss

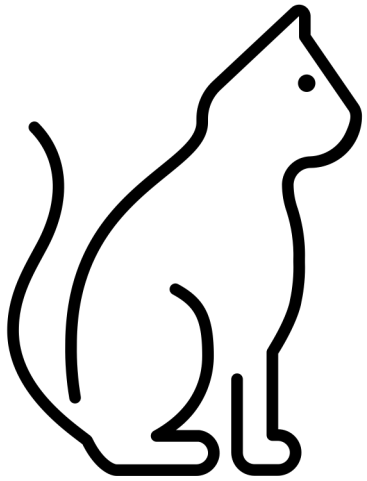


The Difference Loss (?)



Loss Functions for Binary Classification

label y = “cat”



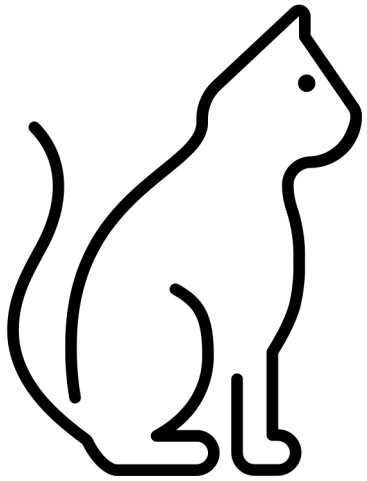
features x = pixels

h(x) = “dog”

Loss = 100

Loss Functions for Binary Classification

label y = “cat”



features x = pixels

h(x) = “cat”

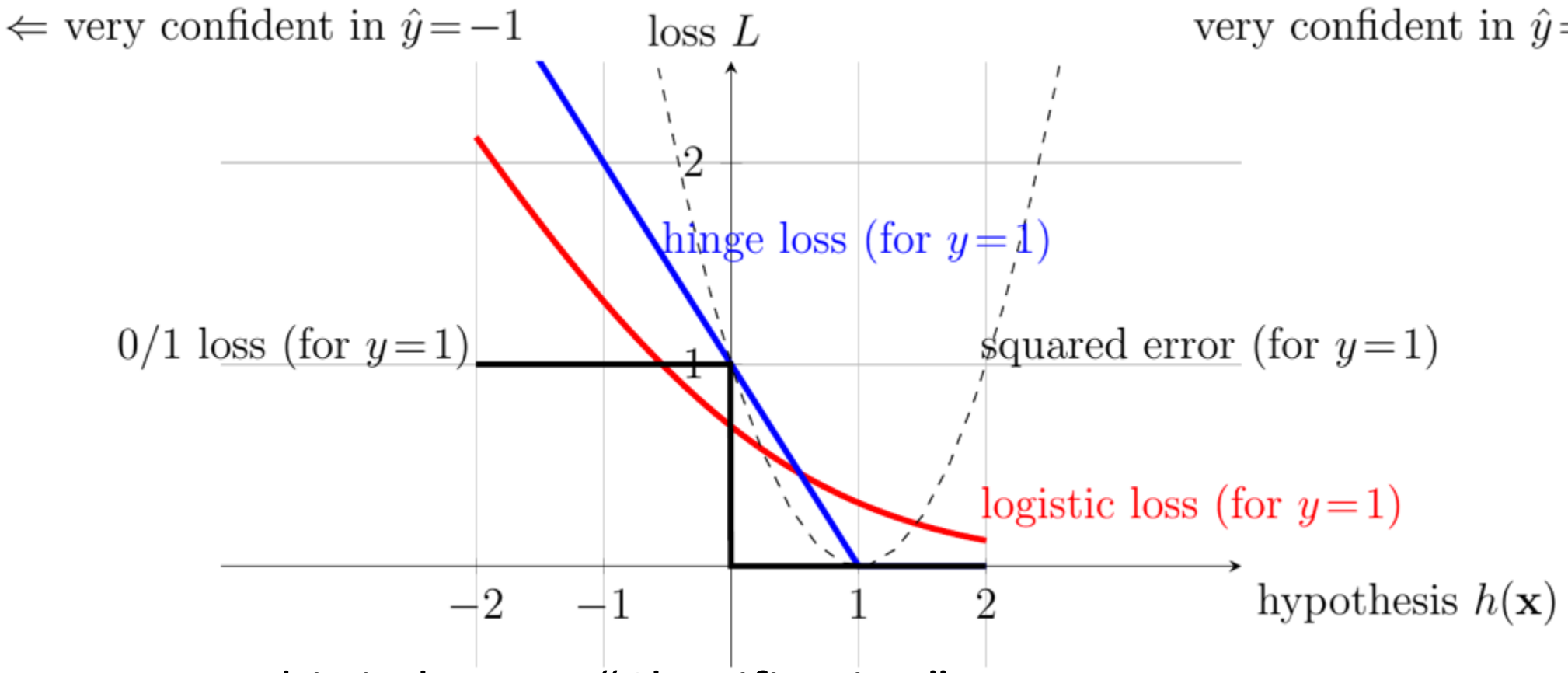
Loss = 0

Classifiers

- consider label values either “cat” or “dog”
- features vector x = pixels values
- can we use linear hypothesis maps h(x)?

- YES!
- use sign h(x) to classify: h(x) > 0 → “dog”
- use |h(x)| as confidence measure

Loss Functions for Binary Classification



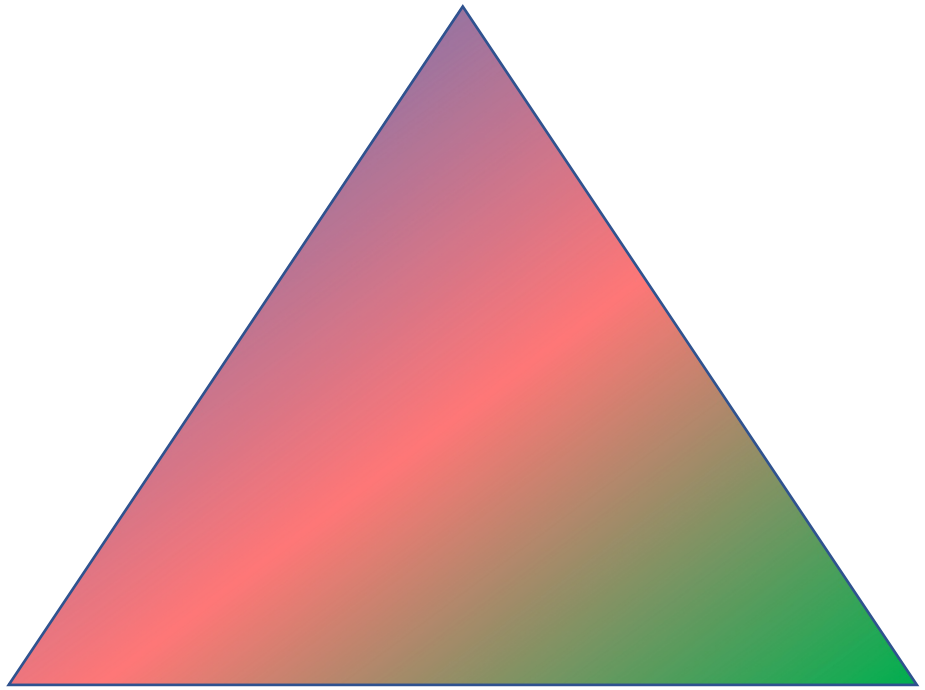
more on this in lecture “Classification”

Which Loss Function ?

- statistical aspects (should favour “reasonable” hypothesis)
- computational aspects (must be able to minimize them)
- interpretation (what does log-loss = -3 mean ?)

…..choosing a suitable loss function is often non-trivial !

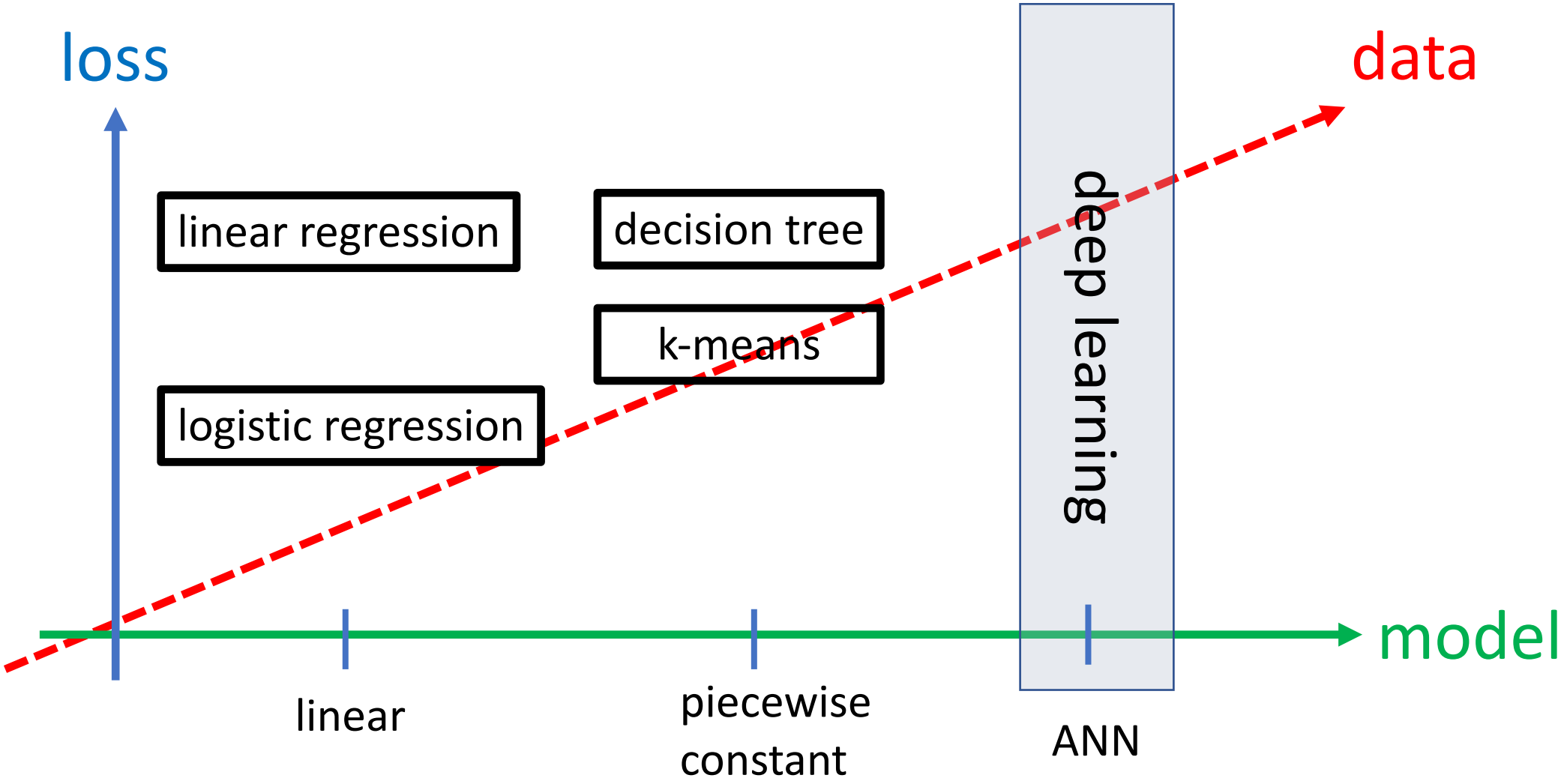
Design Choice: Loss



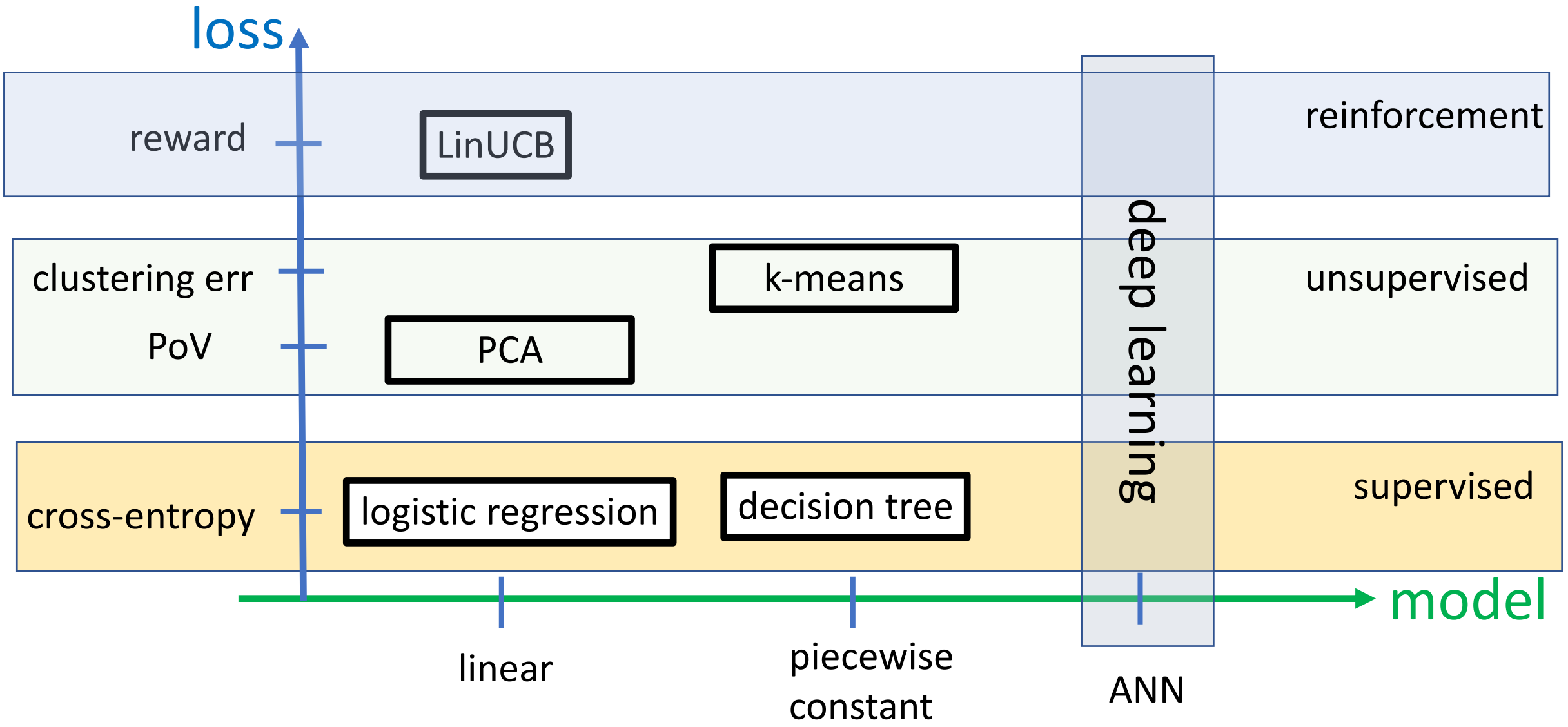
Main Components of ML

- data
- model
- loss

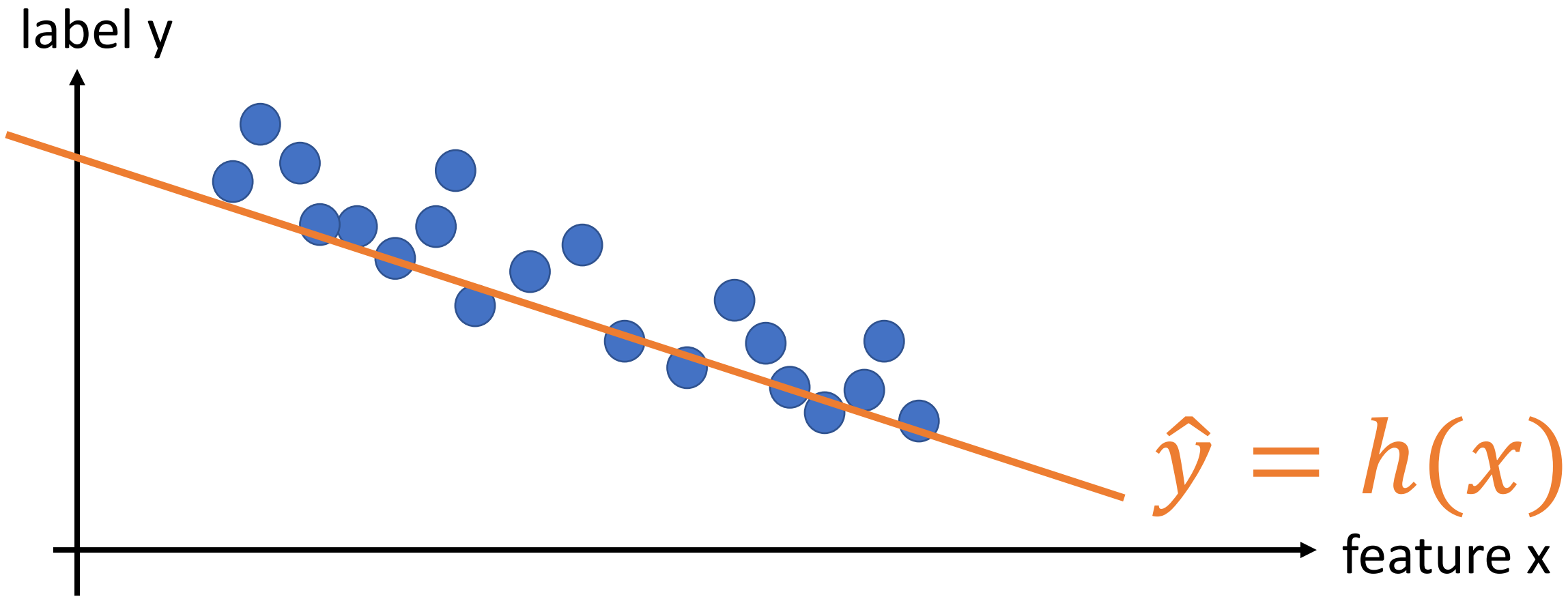
Landscape of ML Methods



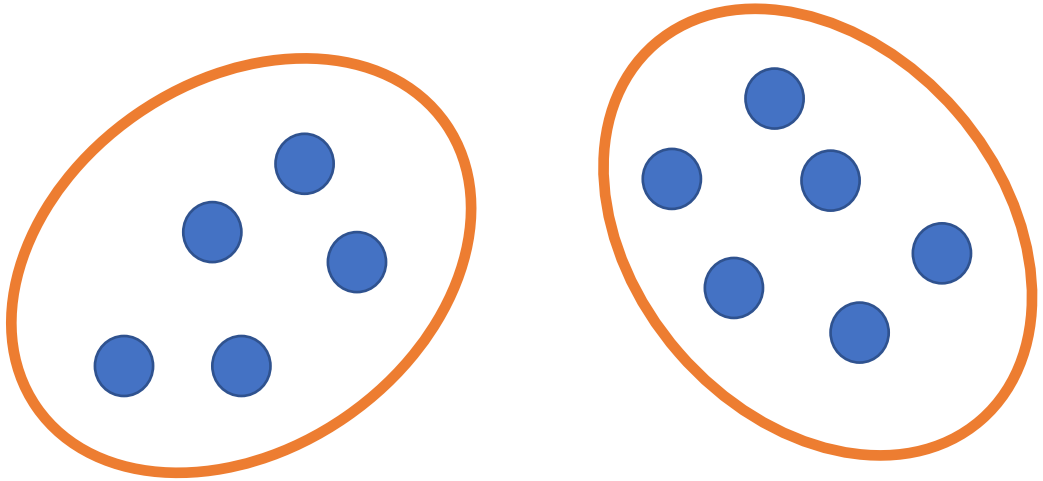
Three Main Flavours of ML



Supervised Learning



Unsupervised Learning



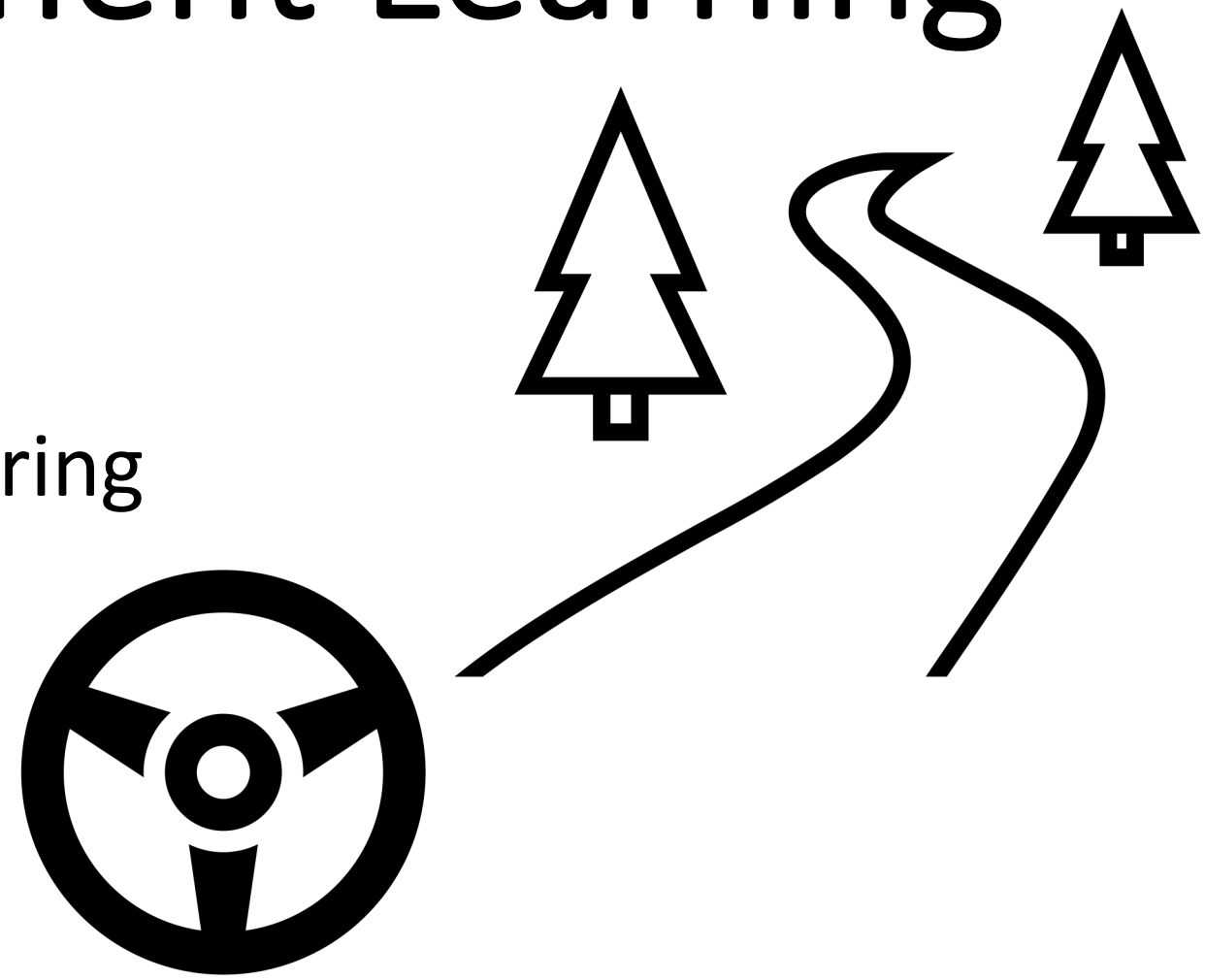
label = cluster index (1 or 2)

clustering methods (such as “k-means”) do not require true label for any data point !

Reinforcement Learning

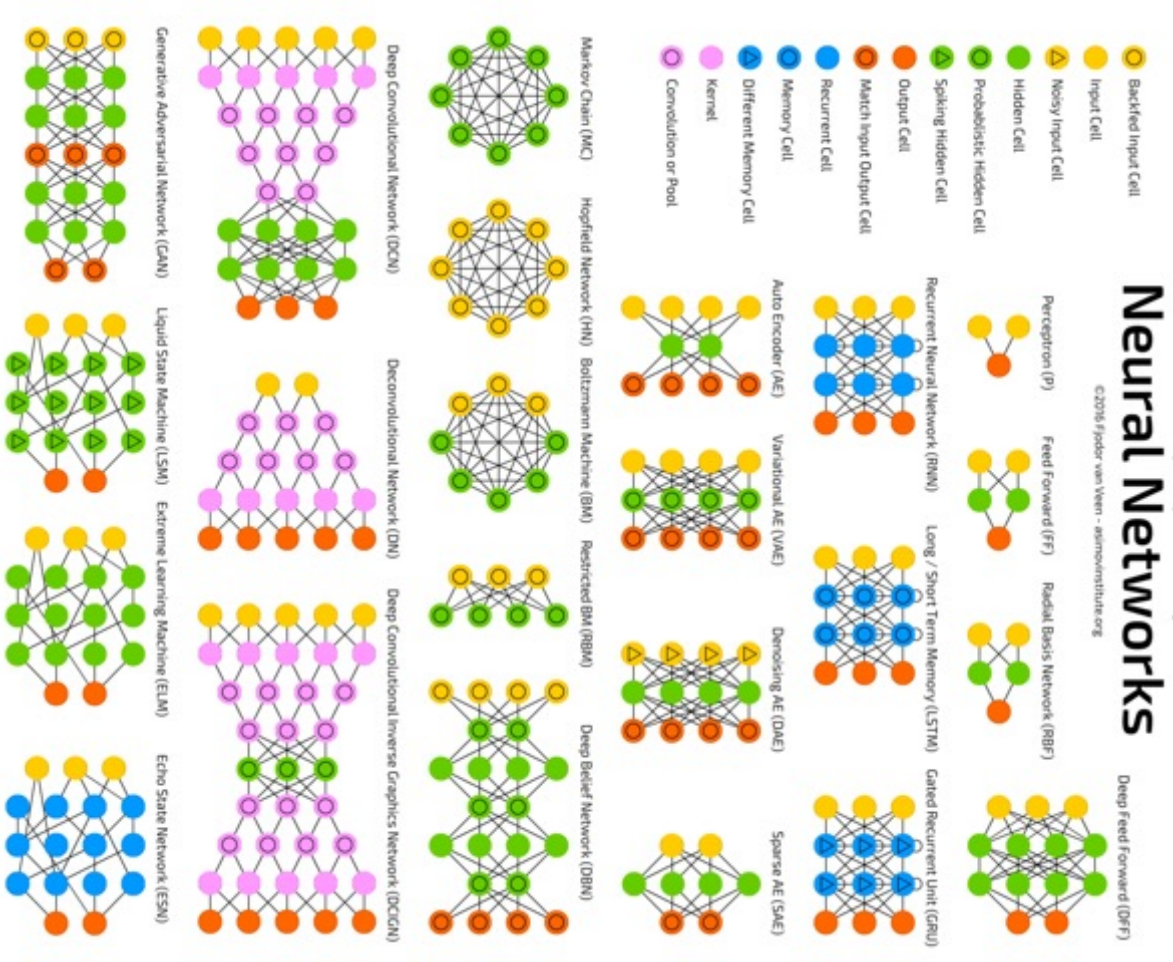
features = on-board camera video

label = “optimal steering direction”



not covered in this course !

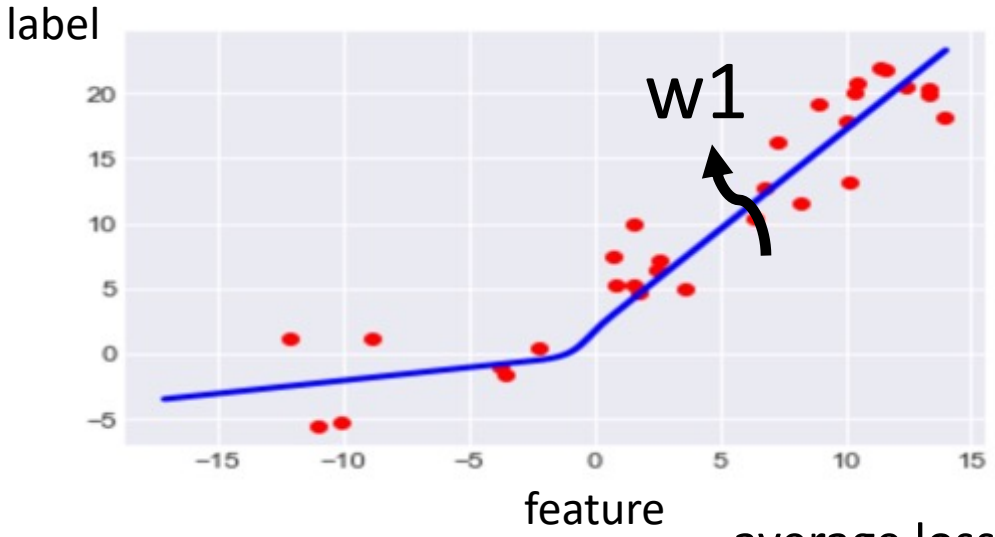
Landscape of Deep Learning



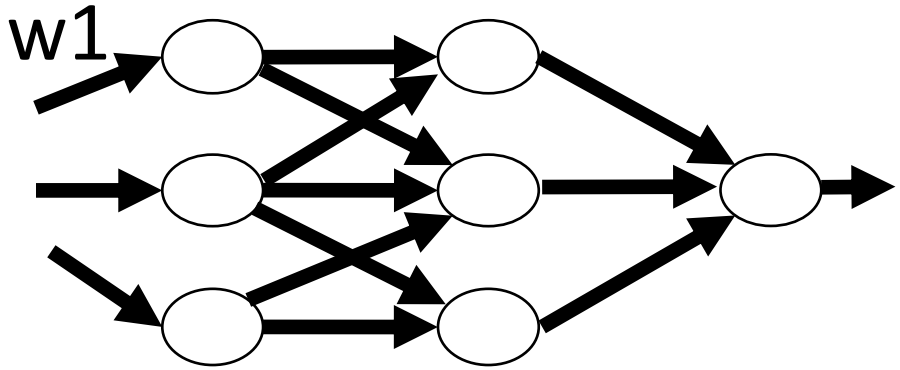
https://www.asimovinstitute.org/neural-network-zoo/

Three Views on Deep Learning

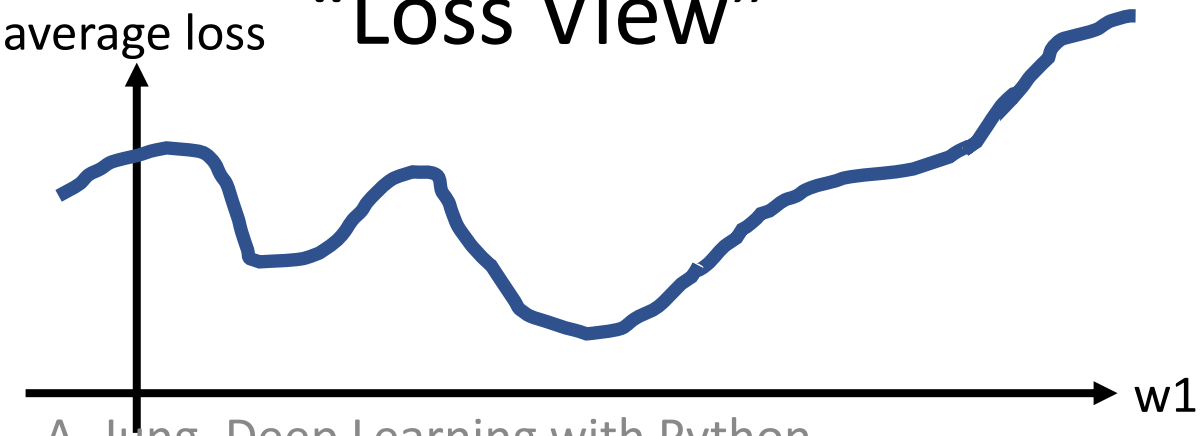
"Data View"



"Model View"



"Loss View"



Python code typically starts with:

1. choose/load data
2. choose/build model (ANN structure)
3. choose/construct loss function

1. choose/load data

```
image_size = (180, 180)
batch_size = 32

train_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "PetImages",
    validation_split=0.2,
    subset="training",
    seed=1337,
    image_size=image_size,
    batch_size=batch_size,
)
val_ds = tf.keras.preprocessing.image_dataset_from_directory(
    "PetImages",
    validation_split=0.2,
    subset="validation",
    seed=1337,
    image_size=image_size,
    batch_size=batch_size,
)
```

2. choose/build model (ANN structure)

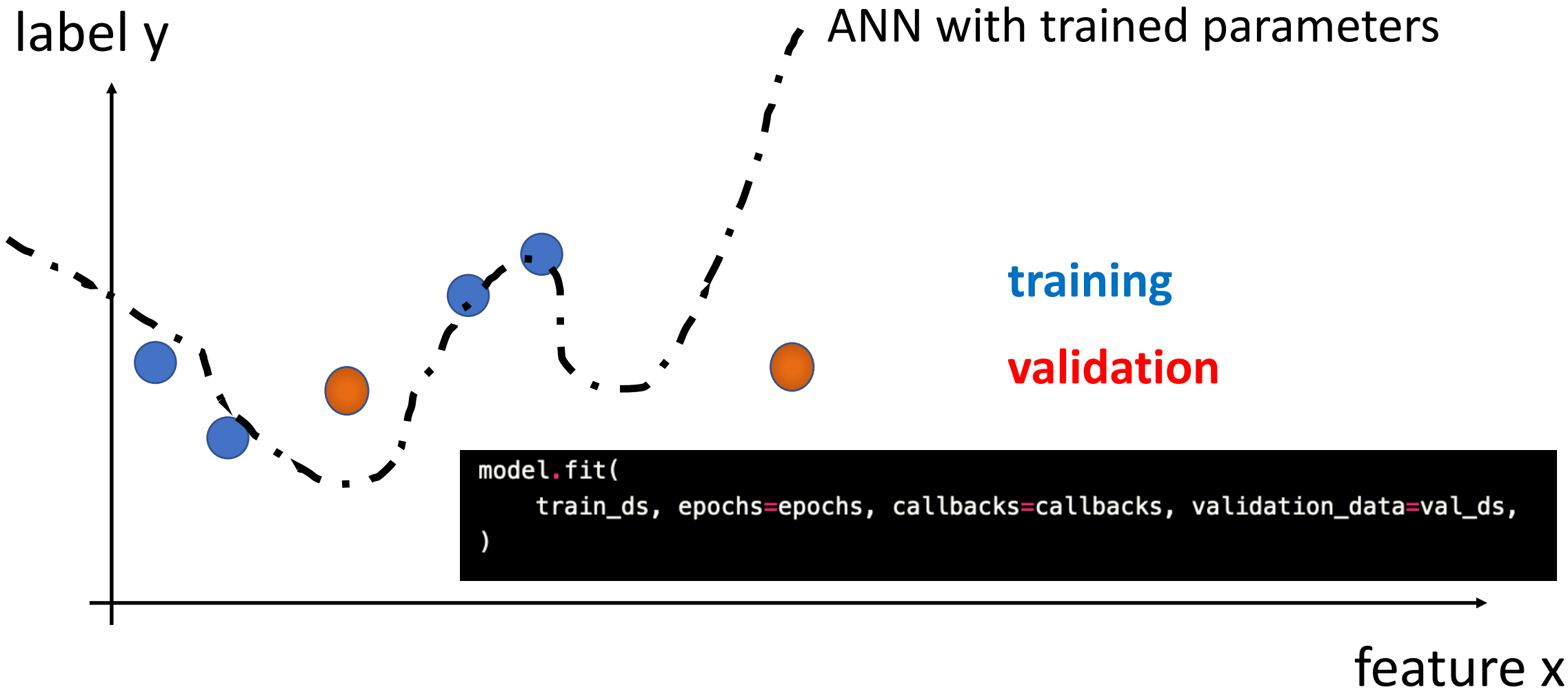
```
# Entry block
x = layers.Rescaling(1.0 / 255)(x)
x = layers.Conv2D(32, 3, strides=2, padding="same")(x)
x = layers.BatchNormalization()(x)
x = layers.Activation("relu")(x)

x = layers.Conv2D(64, 3, padding="same")(x)
x = layers.BatchNormalization()(x)
x = layers.Activation("relu")(x)
```

3. choose/construct loss function

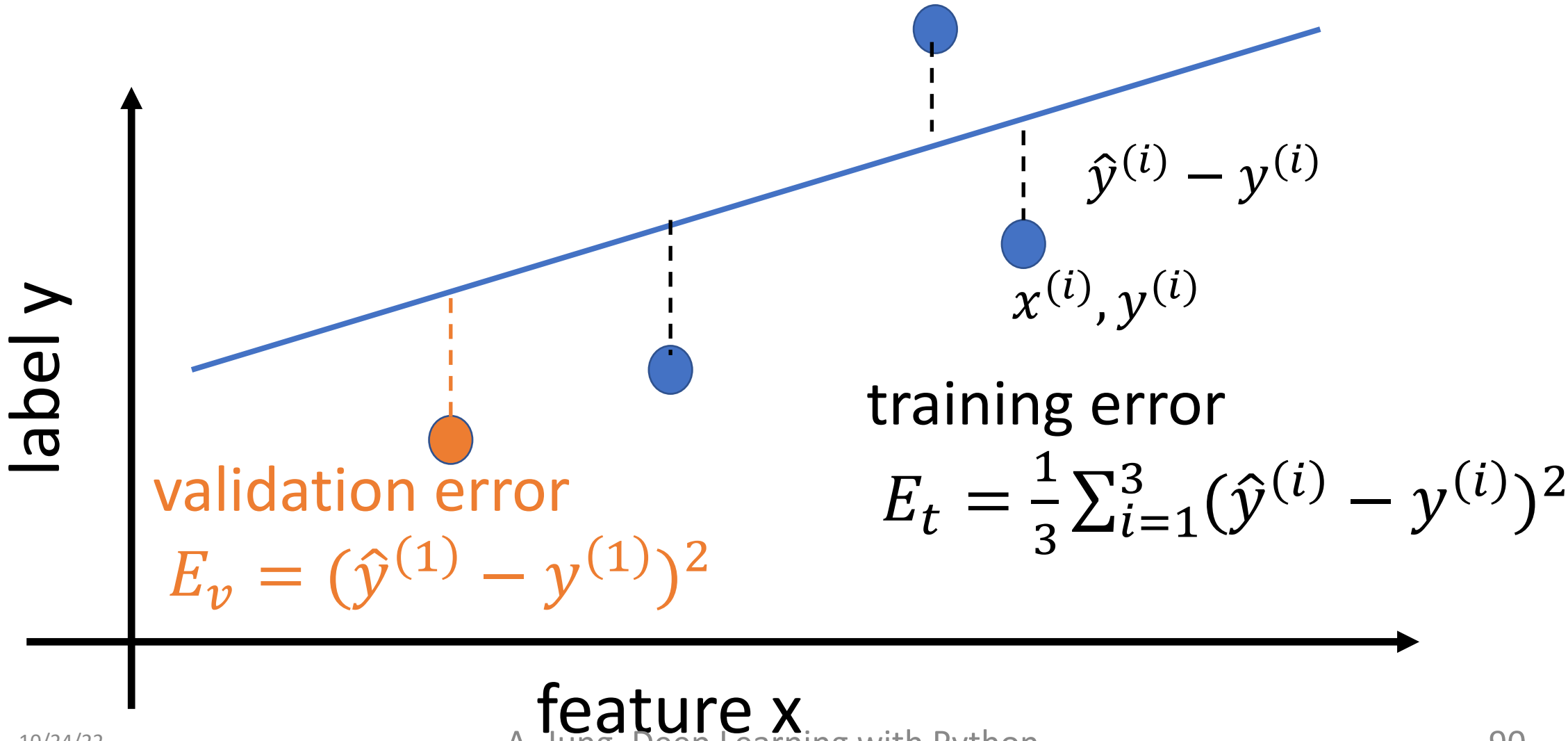
```
model.compile(
    optimizer=keras.optimizers.Adam(1e-3),
    loss="binary_crossentropy",
    metrics=["accuracy"],
)
```

Putting Things Together

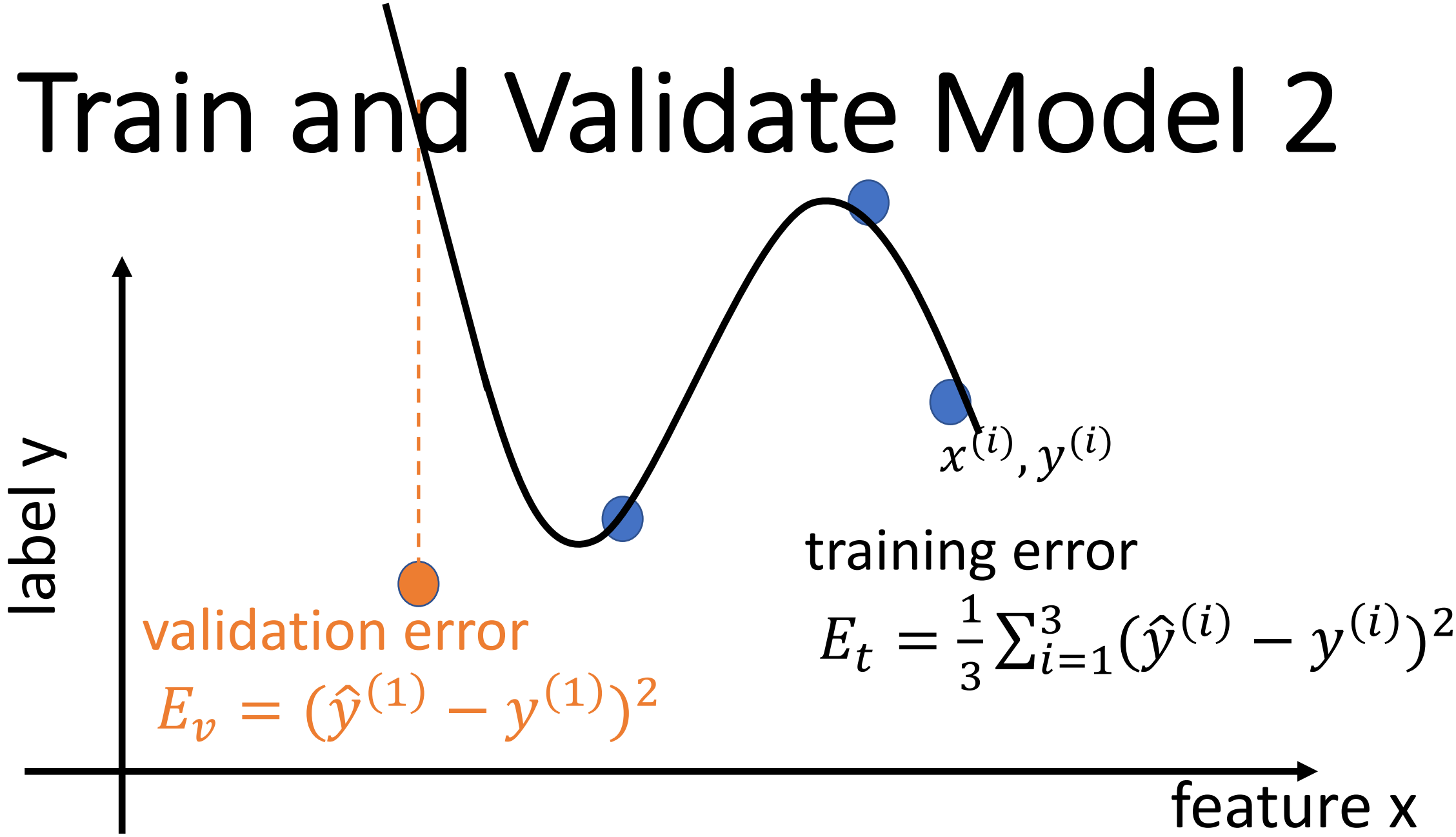
Train and Validate !



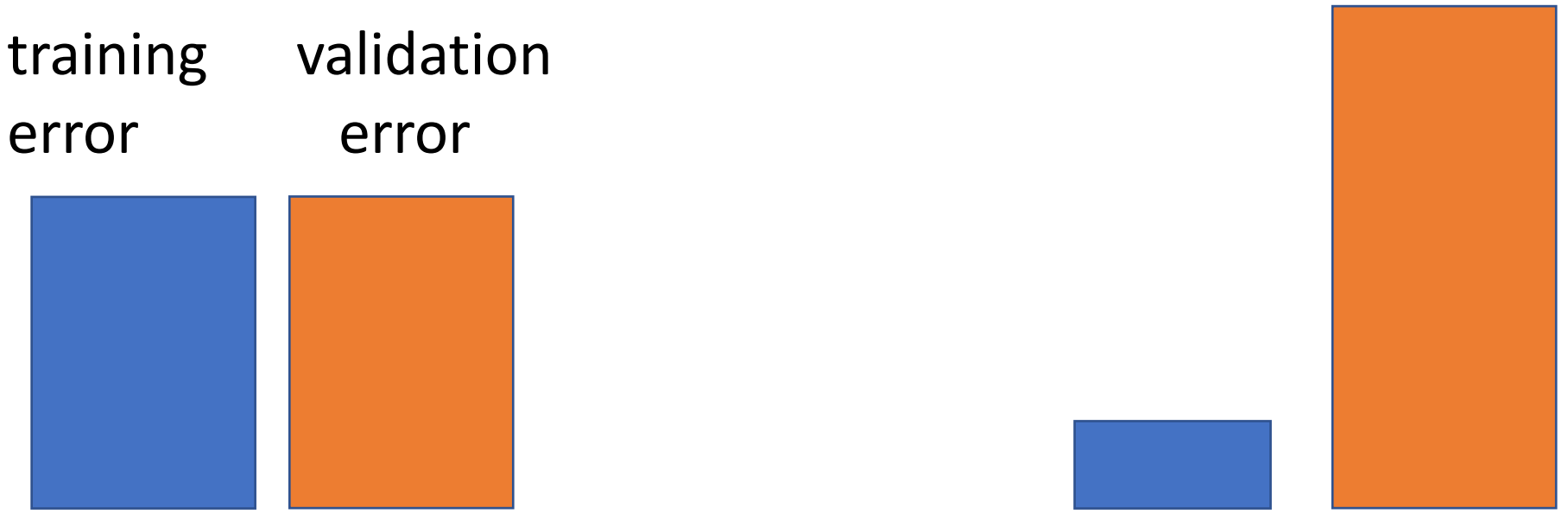
Train and Validate Model 1



Train and Validate Model 2

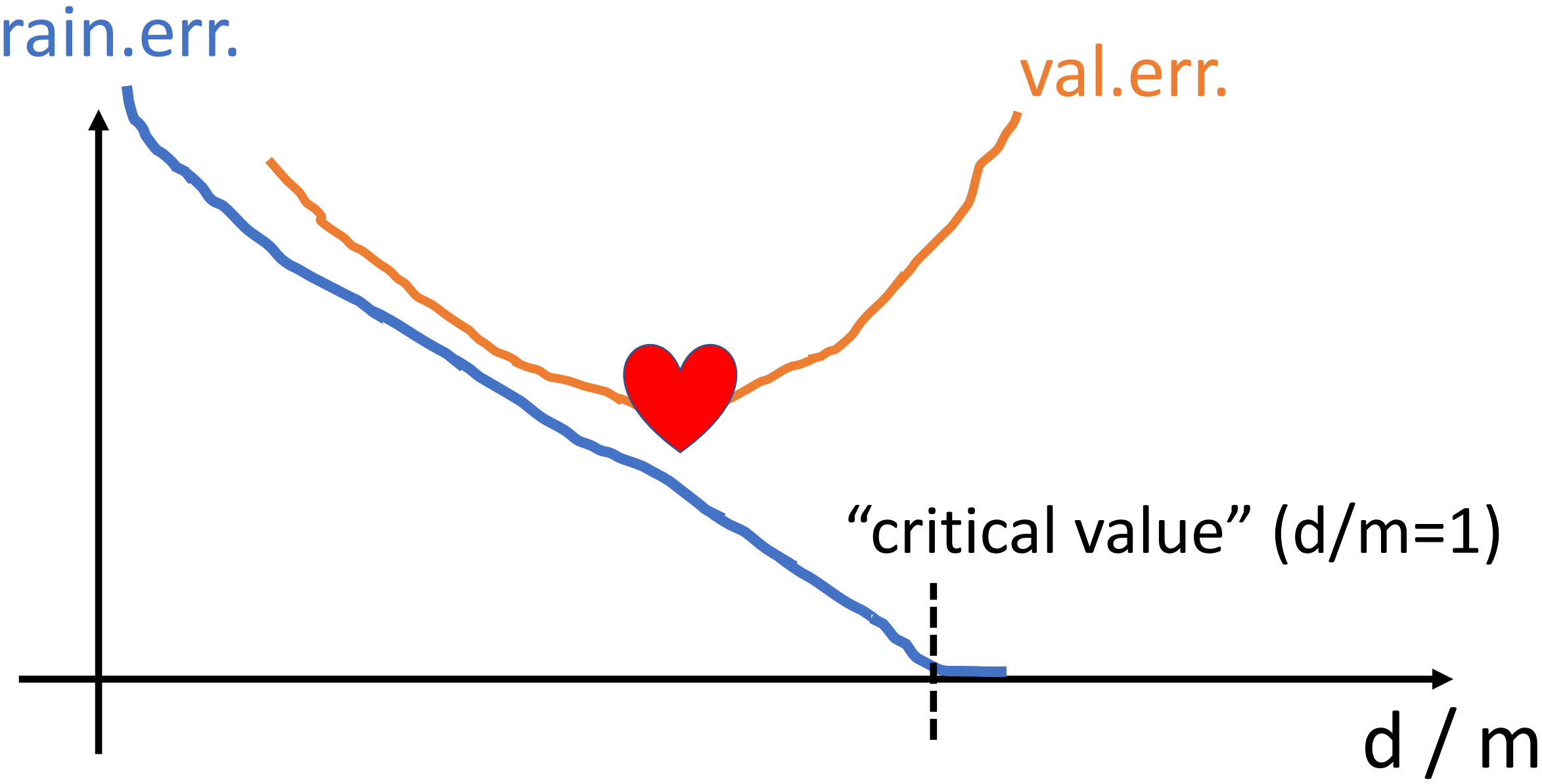


Choose Model via Val.Error

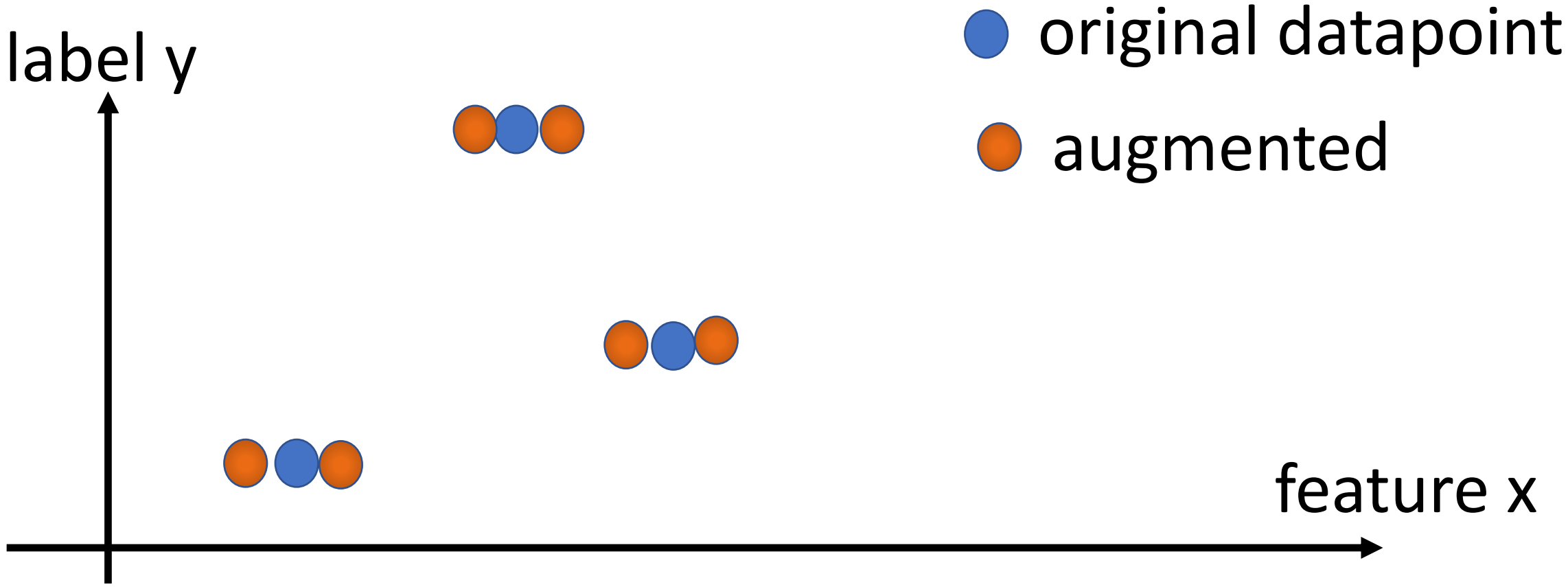


model 1: ANN 1 layer

model 2: ANN 3 layers

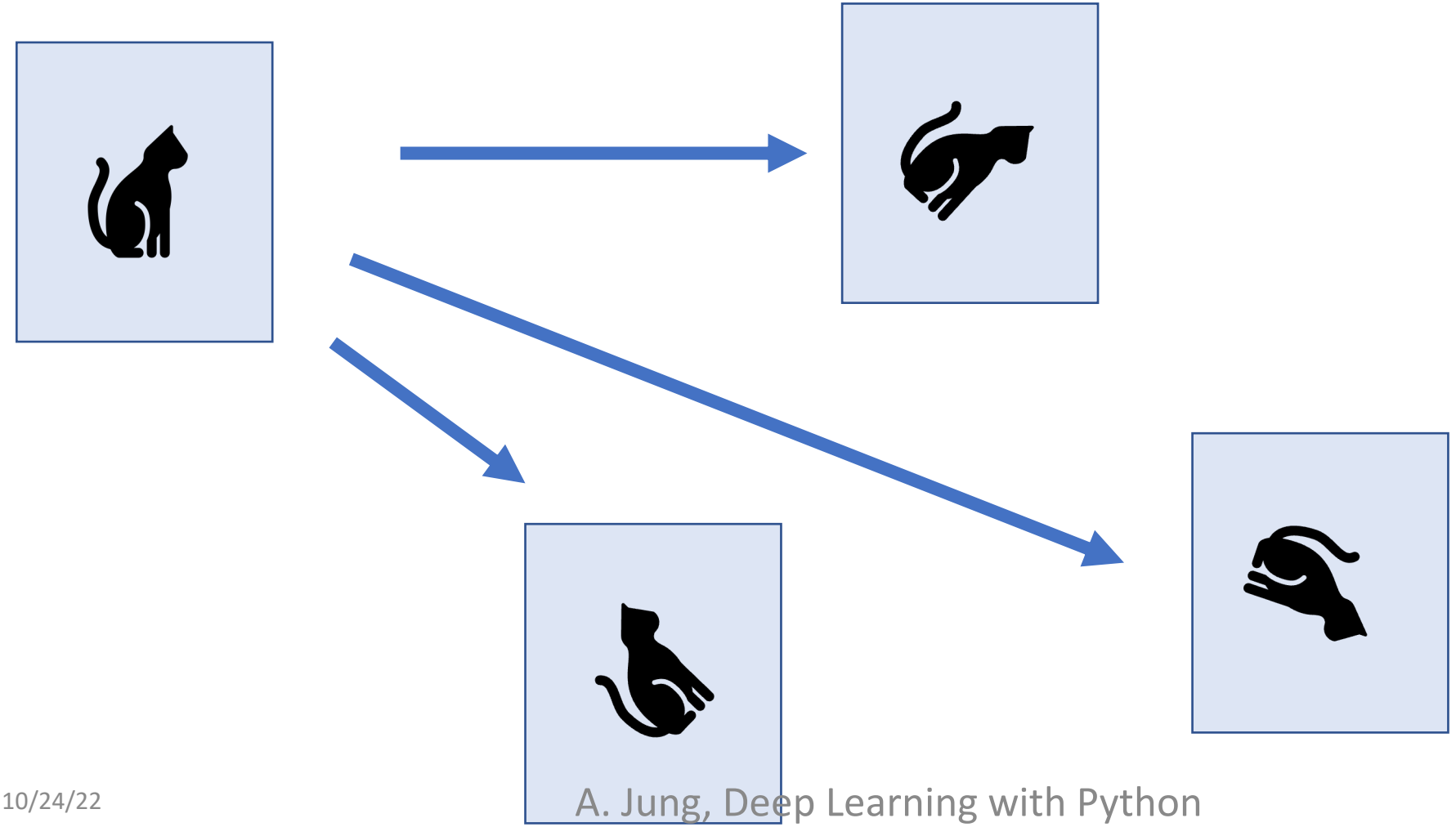

train.err.
val.err.
“critical value” (d/m=1)
d / m

increasing effective data size

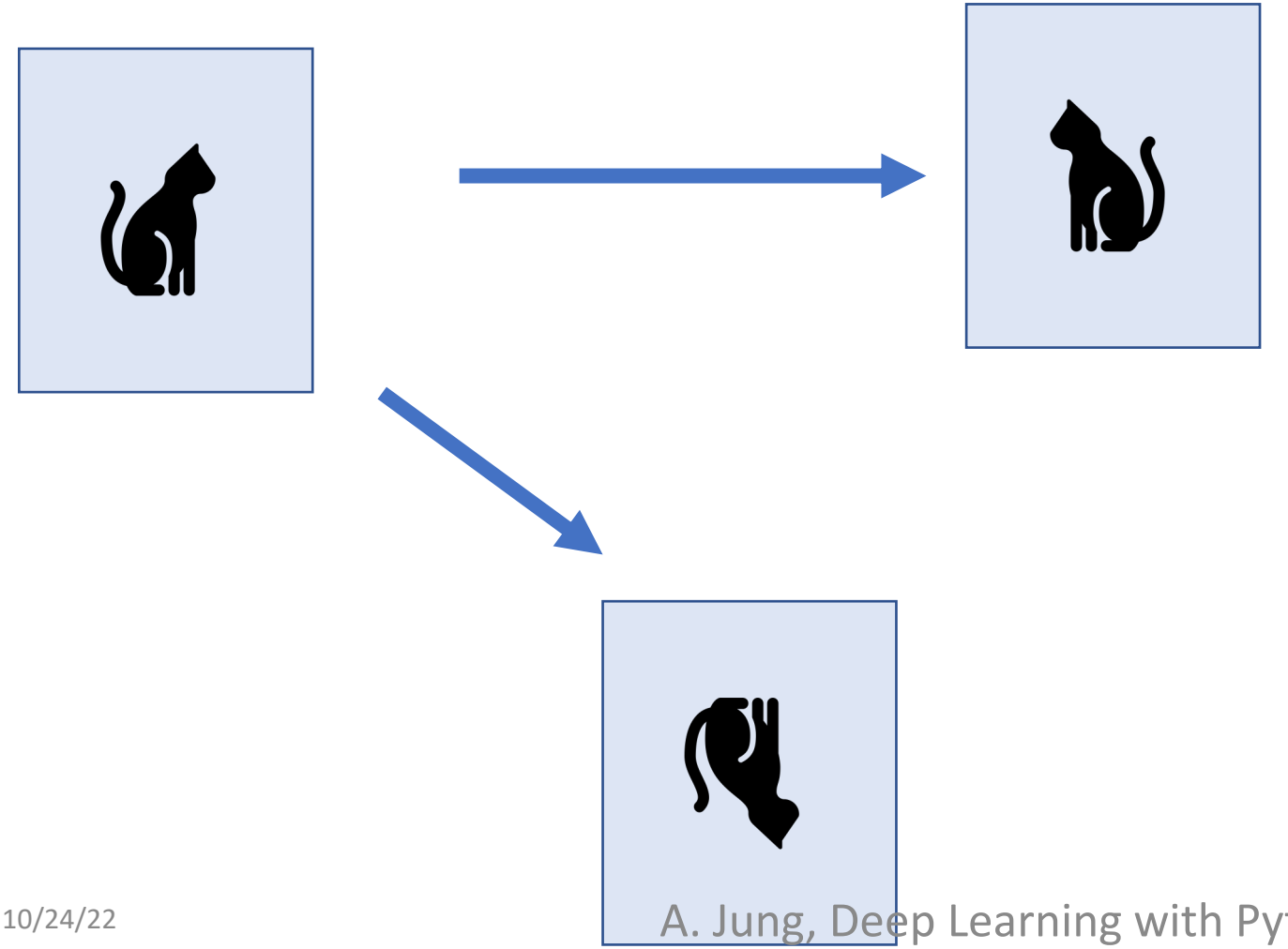


we have increased the dataset by factor 3 !

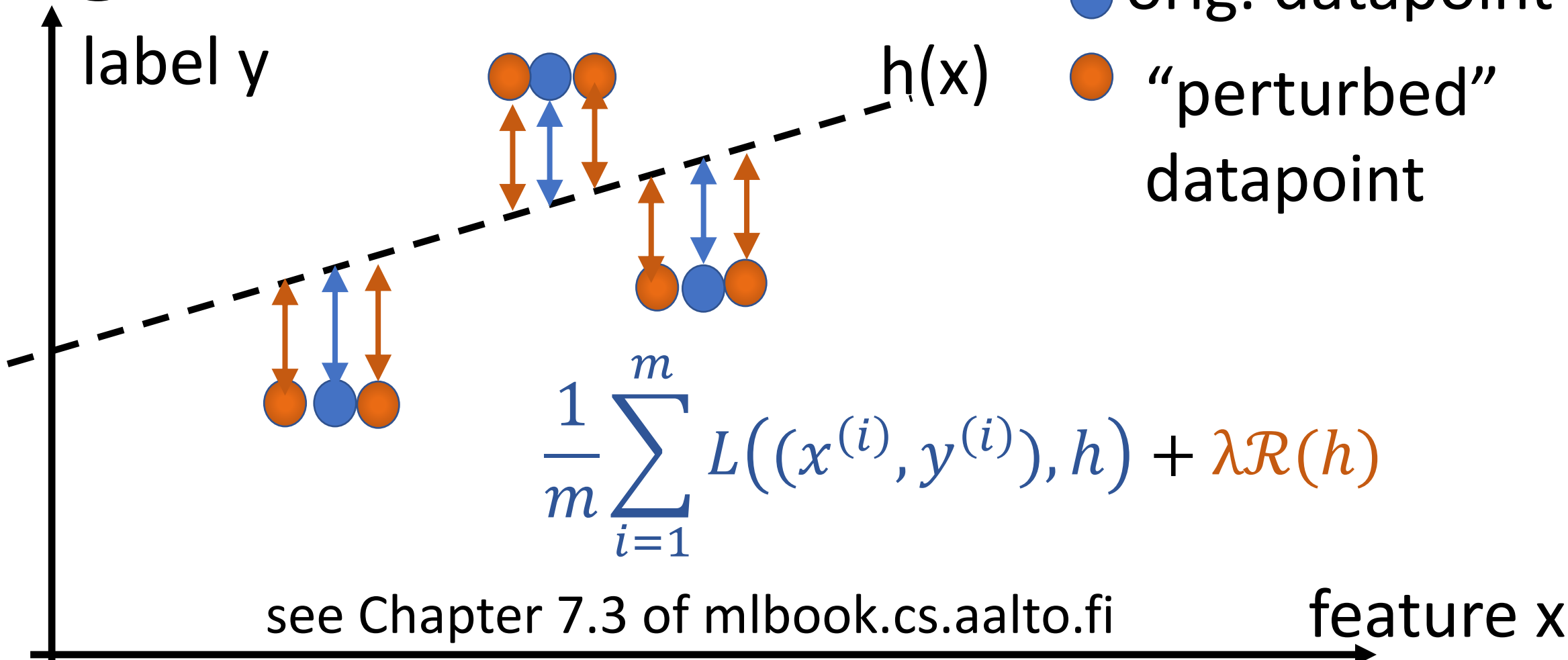
rotated cat image is still cat image



flipped cat image is still cat image

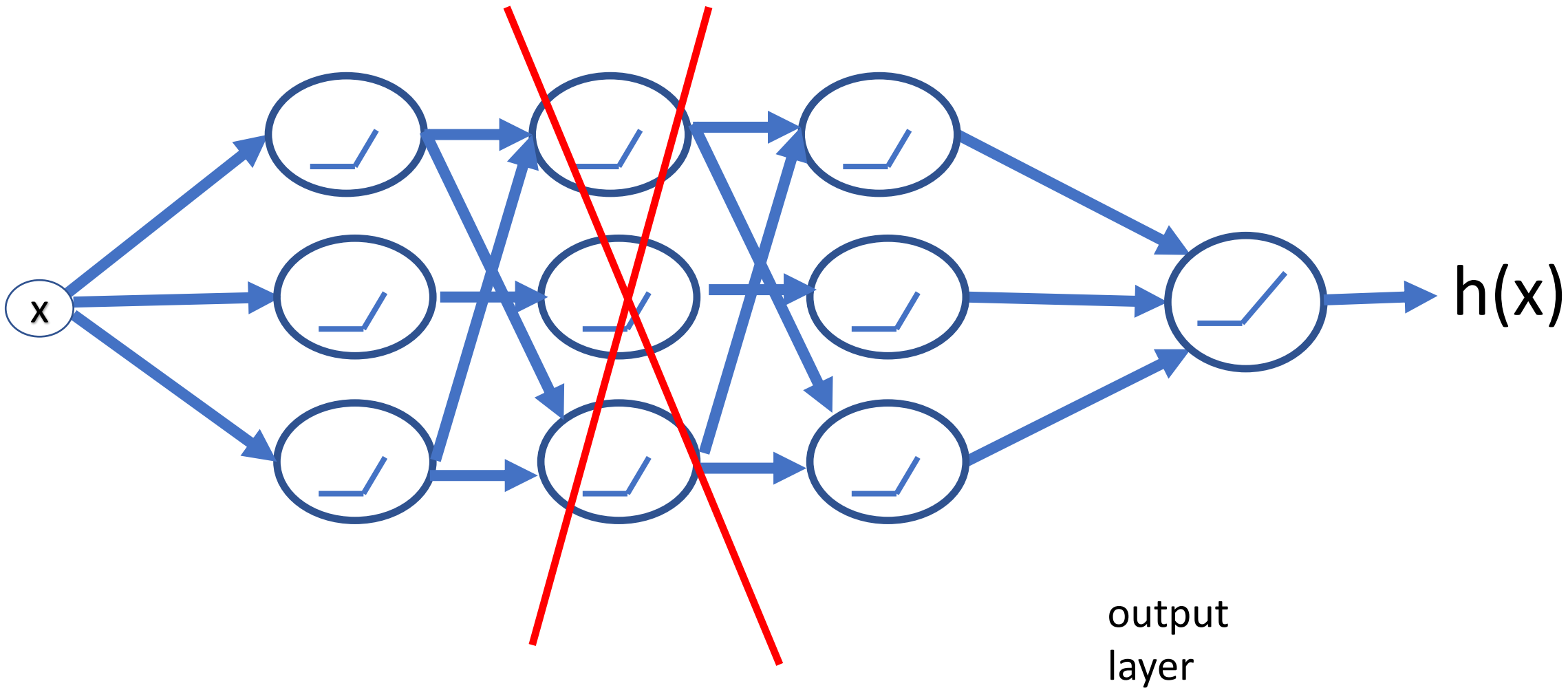


Data Augmentation via Regularization

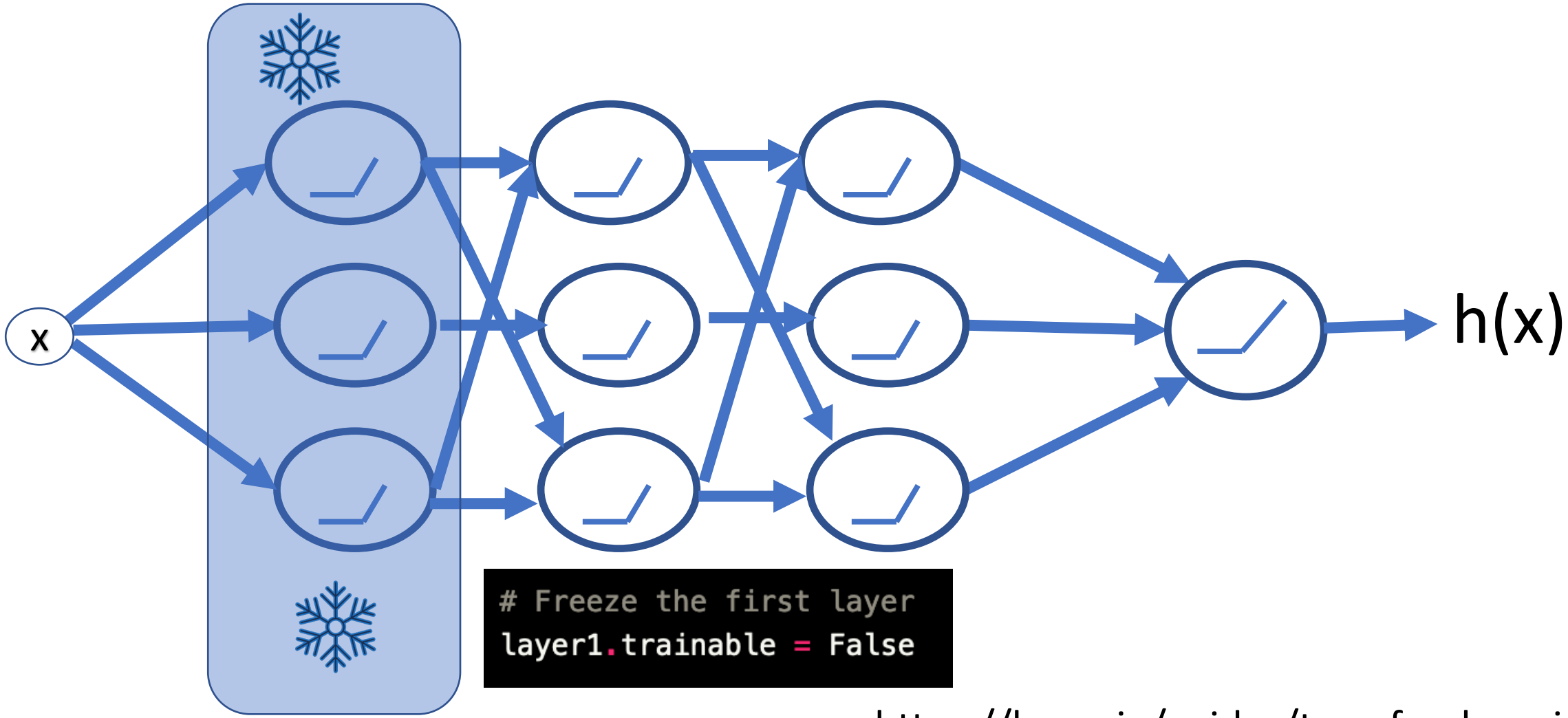


see Chapter 7.3 of mlbook.cs.aalto.fi

decrease eff. model size



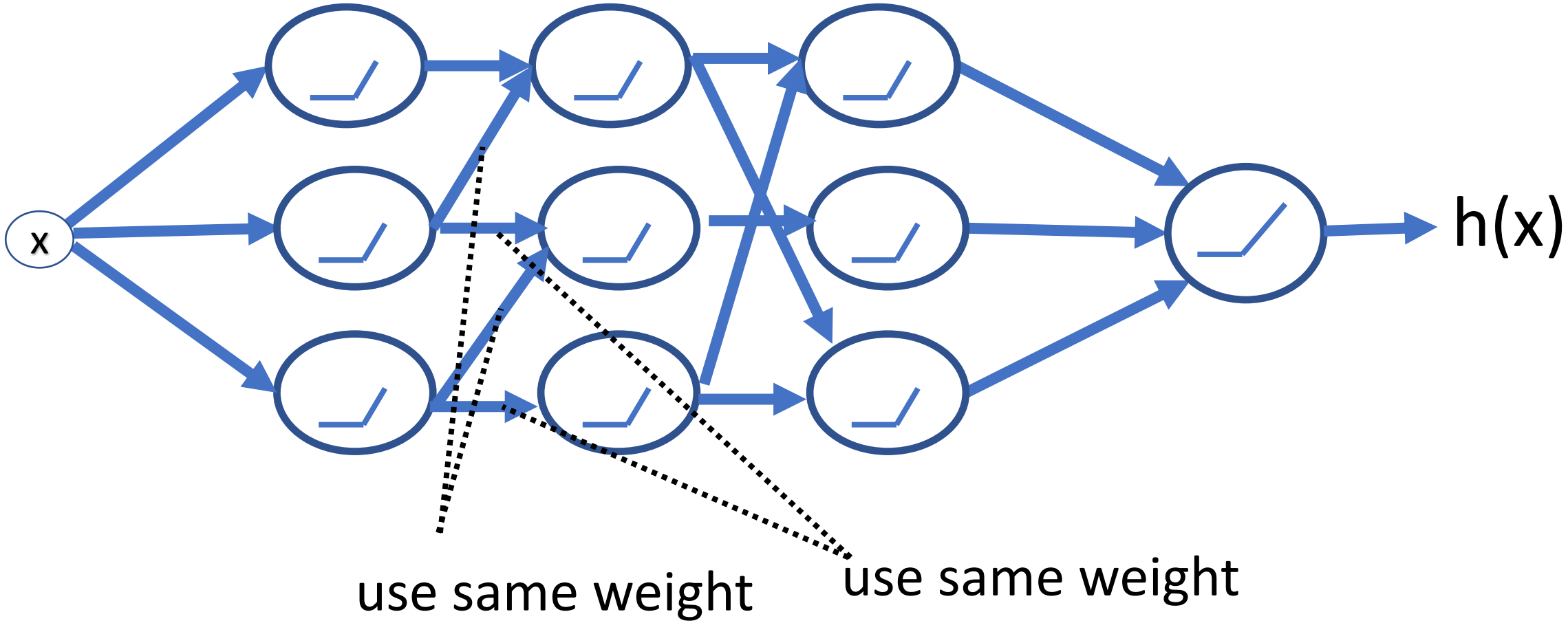
decrease eff. model size



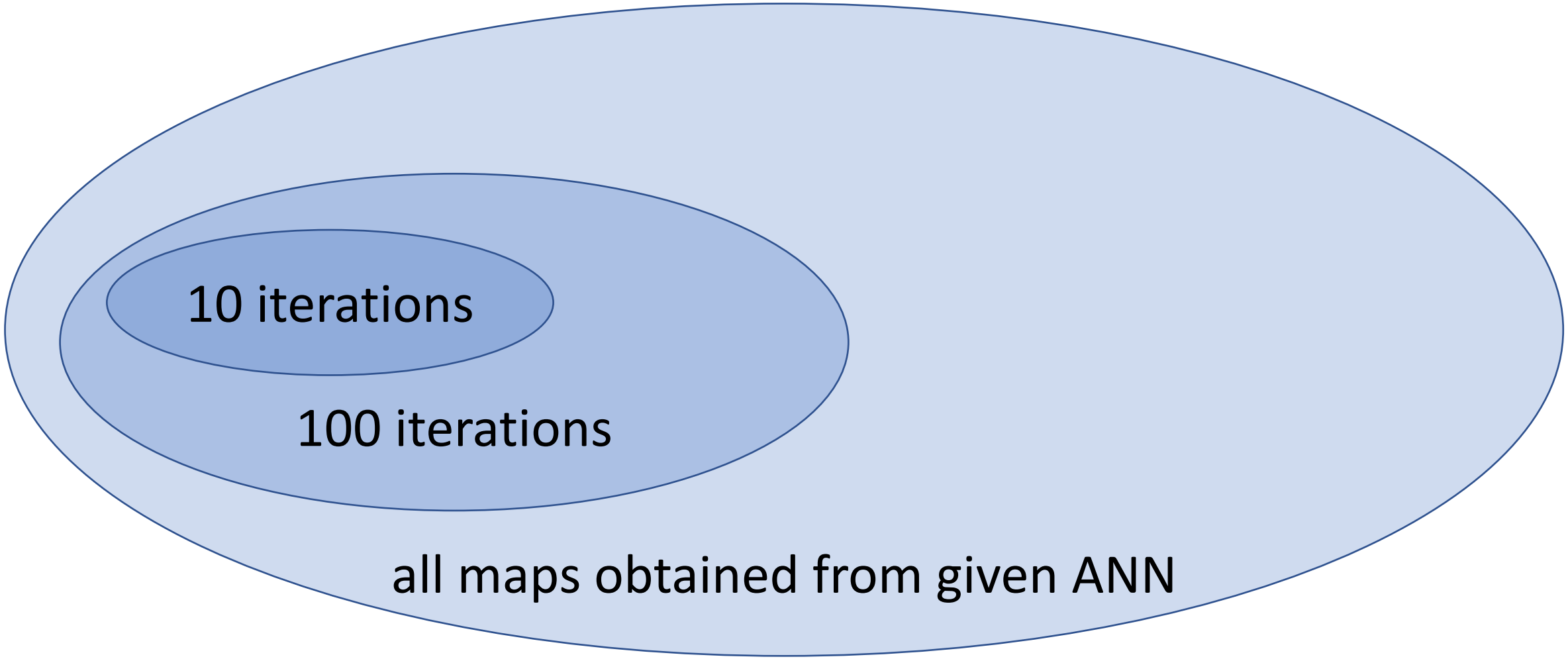

```
# Freeze the first layer
layer1.trainable = False
```

https://keras.io/guides/transfer_learning/

decrease eff. model size with weight sharing



decrease eff. model size



Wrap Up

- data points characterized by features and label
- features $\approx$ low-level properties
- labels $\approx$ high-level properties (quantity of interest)
- GOAL of ML: learn a hypothesis map h(.) such that h(x) $\approx y$
- ML model = comp. tractable subset of possible maps h(.)
- ML quantifies prediction error y-h(x) with a loss function

What’s Next ?

- Zoom Support Session 27.10 at 14:00 pm
- Support Session Fr. 12-16:00 pm hall AS4 in TUAS building (Maarintie 8)
- join our Slack !