Turing Machine and Deep Learning
Lecture 1: Introduction

Satchit Chatterji
satchit.chatterji@gmail.com

MSc Artificial Intelligence
University of Amsterdam

12th May 2023

Who are we?

Lecturer: Satchit Chatterji

TAs: Kami, Chris, Sara

Intended Learning Outcomes

- Familiarity with basic concepts of machine and deep learning.
- Familiarity with basic implementations in code.
- Some intuition of how ML and DL work in practice and how to tune them.
- Experience working on a semester-long machine learning project with a small group.
- Familiar with working with git and GitHub.

Prerequisites

There are no *formal* prerequisites, but the following is assumed:

1. Basic-intermediate knowledge in linear algebra (...eigenvalues).
2. Basic-intermediate knowledge in multivariate calculus (...PDEs).
3. Basic knowledge in probability theory and statistics (...distributions).
4. Working knowledge and experience in Python (...classes)

Practical Matters

The course is comprised of the following components:

- 5 lectures
- 5 tutorials
- 4 homework assignments
- 1 semester project

Grading

- Homework assignments
- Semester project code, report
- Semester project presentation

Lectures Overview

Date	**Lecture**
12/04/2023	Introduction, Recap of PDS
19/04/2023	Supervised Learning Methods
26/04/2023	Unsupervised Learning Methods
02/05/2023	Deep Learning: Intro to Neural Networks
09/05/2023	ML Best Practices + extra

Semester Project

The semester project is intended to be a hands on application of the concepts you learned during this course.

- Final project in groups of 3-4.
- **Topic of your choice**, incorporating aspects of machine learning and/or neural networks.
- 10 minute presentation + 5 minute questions (14/05/2023)
- Hand-in notebook on GitHub (14/05/2023, 23:59)
- Scientific report on the project (max. 4000 words) (details on Canvas + more coming soon)

Strong recommendation

Hand in a 1 page (max) project proposal by 26/05/2023 describing your motivation, data sources and intended methods.

Recommended Readings/References

- Jaeger, H (2022) *Lecture Notes in Neural Networks*, University of Groningen.
  `https://www.ai.rug.nl/minds/uploads/LN_NN_RUG.pdf`
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep Learning*. Nature.
  `https://www.deeplearningbook.org/`
- Deisenroth, M. P., Faisal, A. A., & Ong, C. S. (2020). Mathematics for Machine Learning. Cambridge University Press.
  `https://mml-book.github.io/`
- Russell, S. J. Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Pearson Education, Inc..

Let's Begin!

Recap: *Python for Data Science*

Introduction to Machine Learning

The Hype (as per February 2023)

Fortune

A robot's $100 billion error: Alphabet shares tank after its ChatGPT rival makes a mistake in its very first ad

Google's new chatbot to challenge ChatGPT made a mistake causing the company's market value to slide.

2 days ago

The Guardian

Well, I never: AI is very proficient at designing nerve agents

Researchers for a pharmaceutical company stumbled upon a nightmarish realisation, proving there's nothing intrinsically good about machine...

3 hours ago

CNN

Getty Images suing the makers of popular AI art tool for allegedly stealing photos

These systems include Stable Diffusion and DALL-E, from OpenAI. Shutterstock, a Getty Images competitor and fellow stock image platform,...

3 weeks ago

Reuters

Microsoft co-founder Bill Gates: ChatGPT 'will change our world'

Microsoft co-founder Bill Gates believes ChatGPT, a chatbot that gives strikingly human-like responses to user queries, is as significant as...

1 day ago

The Guardian

'There is no standard': investigation finds AI objectify women's bodies

Images posted on social media are analyzed by artificial intelligence (AI) algorithms that decide what to amplify and what to suppress.

2 days ago

Machine Learning In Theory

Comprising fields

- Statistics
- Linear Algebra
- Probability Theory
- Multivariable Calculus
- ...

Not what this course is about!

Machine Learning In *Practice*

Elements of

- Statistics
- Linear Algebra
- Probability Theory
- Multivariable Calculus
- ...

The good news...

People have done it for you!*

*Still needs work from ML practitioners like you to make it better!

What is *Learning*?

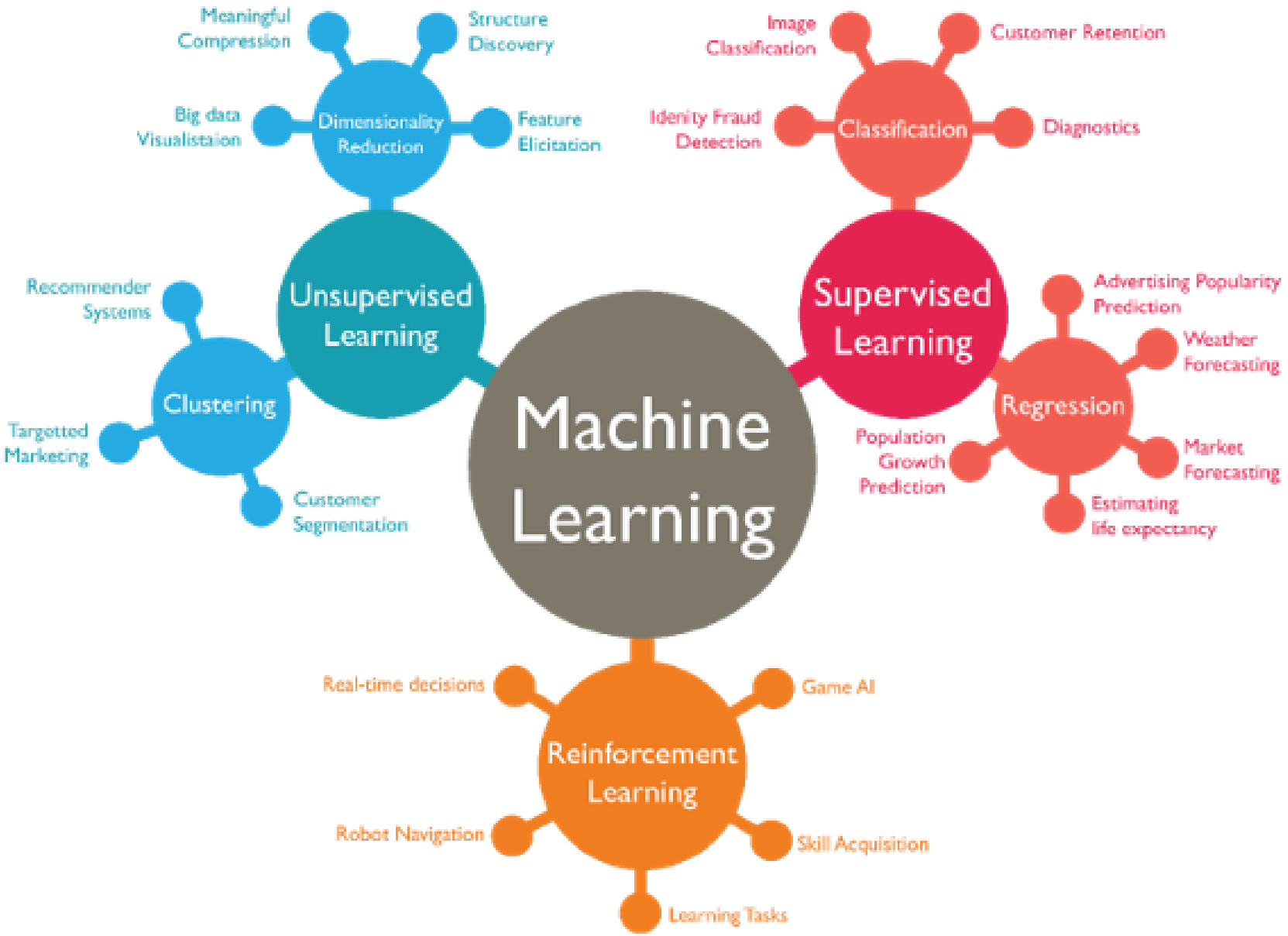
What does it mean to *learn*?

A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

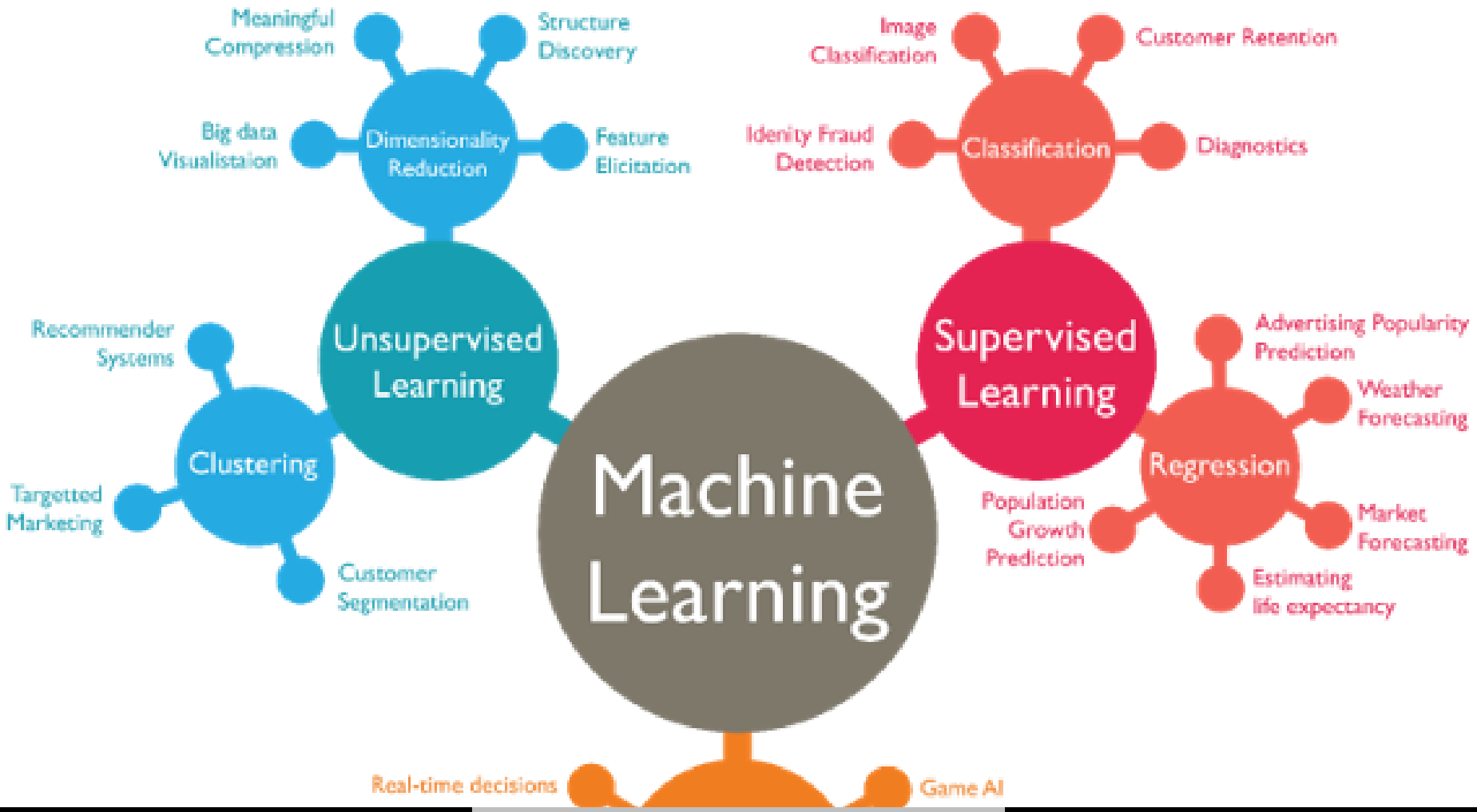
- Tom Mitchell

Domains of ML

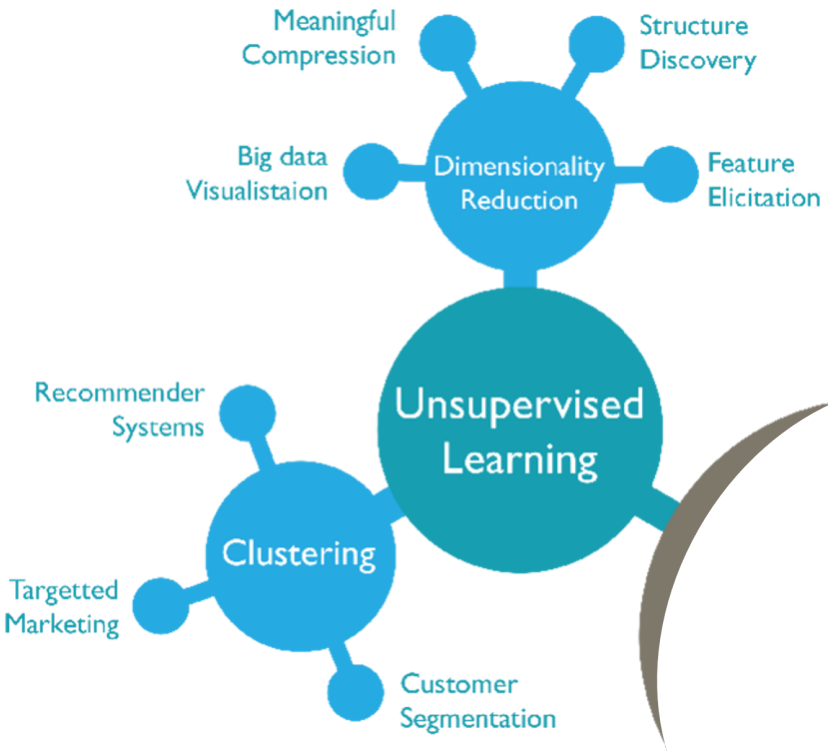
Domains of ML



Domains of ML



Domain: Unsupervised ML



Given

Unlabelled, possibly unstructured data, notated simply as:

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

Goal

Several, including:

- Dimensionality reduction
- Model probability densities $P(X)$
- Find general patterns/structures/groups

e.g.: K-means clustering, PCA, autoencoders

Domain: Supervised ML

Given

Input-output pairs of the form:

$$S = (x_i, y_i)$$

$$x_i \in \mathbb{R}^K, y_i \in \mathbb{R}^M, i = 1, ..., N$$

Assumption

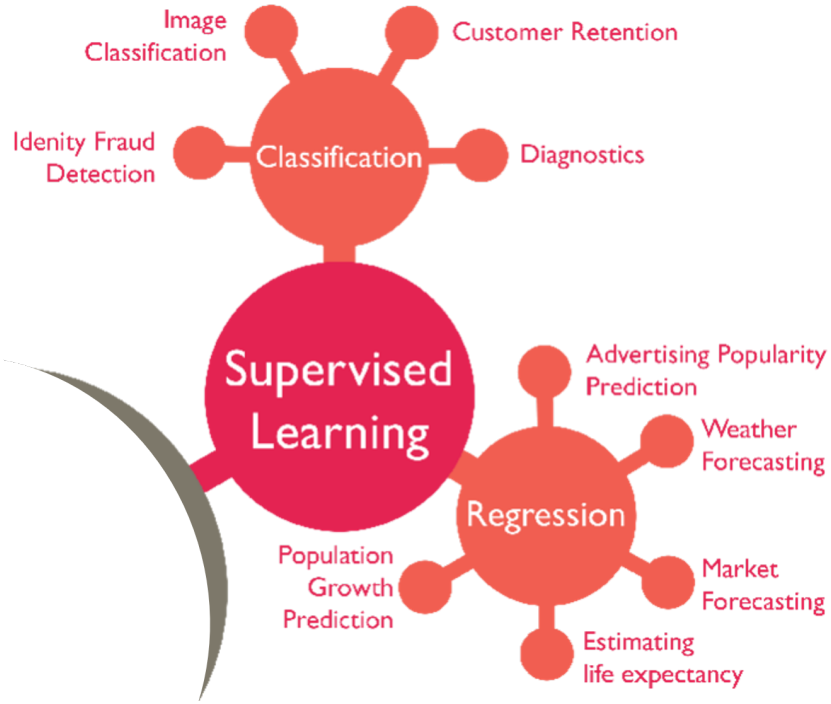
Data is generated by a 'true' function f with some noise:

$$y_i = f(x_i) + \nu_i$$

Goal

'Learn' an approximation $\hat{f}$ that is close to the real function f over all S:

$$\hat{f}(x_i) \approx f(x_i) \quad \forall i \in 1, ..., N$$



Tangent: Representations

Goal (Unsupervised)

Model probability densities $P(X)$, given examples $x_i \in R^K$.

Goal (Supervised)

'Learn' an approximation $\hat{f}$ that is close to the real function f over all S:

$$\hat{f}(x_i) \approx f(x_i) \;\; \forall i \in 1, ..., N$$

Tangent: Representations

Goal (Unsupervised)

Model probability densities $P(X)$, given examples $x_i \in R^K$.

Goal (Supervised)

'Learn' an approximation $\hat{f}$ that is close to the real function f over all S:

$$\hat{f}(x_i) \approx f(x_i) \;\; \forall i \in 1, ..., N$$

What even *are* $P(X)$ and $\hat{f}(x_i)$? What is x_i?

- Some fancy math that describes the data.
- Some approximation of the underlying generating function.
- Some condensed *representation* of our data.

Tangent: Representations

Gödel, Escher, Bach by Douglas Hofstadter.

- 🧔 You may know more about Chinese cuisine than I do, Mr.T, I'll bet I know more about Japanese poetry than you do. Have you ever read any haiku?
- 🐢 I'm afraid not. What is a haiku?
- 🧔 A haiku is a Japanese seventeen-syllable poem – or minipoem rather, which is evocative in the same way, perhaps, as a fragrant petal is, or a lily pond in a light drizzle. It generally consists of groups of five, then seven, then five syllables.
- 🐢 Such compressed poems with seventeen syllables can't have much meaning...
- 🧔 Meaning lies as much in the mind of the reader as in the haiku.

Tangent: Representations

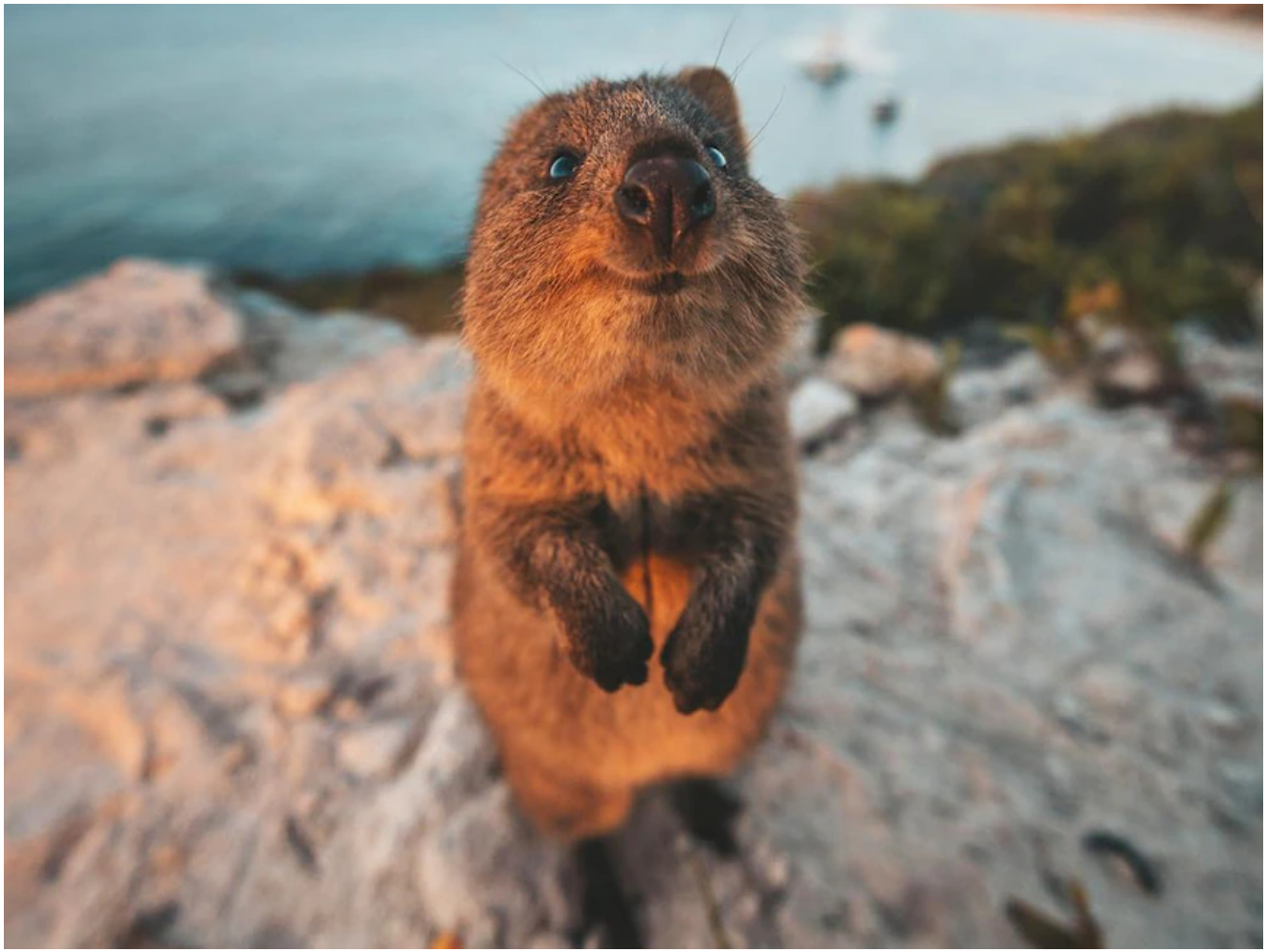
Such compressed poems
with seventeen syllables
can't have much meaning

Meaning lies as much
in the mind of the reader
as in the haiku.

Tangent: Representations

Where is the music?

A breather



Quok

Supervised ML: What is 'Learning'? (I)

Loss function: A distance measure $\mathcal{L}$ between $\hat{f}(x)$ and $f(x)$, considered to be 'good' if $\mathcal{L}$ is low across many instances of S, and is non-negative by convention:

$$\mathcal{L} : \mathbb{R}^M \times \mathbb{R}^M \rightarrow \mathbb{R}^{\geq 0}$$

Expected risk: Assuming x_i and y_i come from random variables X and Y, the expected risk of a model $\hat{f}$ is defined as:

$$R(\hat{f}) = \mathbb{E}[\mathcal{L}(\hat{f}(X), Y)]$$

Thus, learning algorithms should minimise the *expected risk*.

Supervised ML: What is 'Learning'? (II)

Empirical Risk: Since we do not know the *actual* joint distribution $P_{X,Y}$, we cannot actually minimise the expected risk. So, we gather sampled from the real world and have the algorithm minimise the *empirical risk*:

$$R^{emp}(\hat{f}) = \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(\hat{f}(x_i), y_i)$$

Thus, if we have a model $h \in \mathcal{H}$ ($\mathcal{H}$ is called the '**hypothesis space**'), we can find an **optimal** model h_{opt} as:

$$\hat{f} = h_{opt} = \arg\min_{h\in\mathcal{H}} \frac{1}{N}\sum_{i=1}^{N}\mathcal{L}(h(x_i), y_i)$$

We just did machine learning! Right?

Supervised ML: An Example

SML: Training Data

Given

Input-output pairs of the form:

$$S = (x_i, y_i)$$

$$x_i \in \mathbb{R}^K, y_i \in \mathbb{R}^M, i = 1, ..., N$$

	x	y
0	1.256637	1.146804
1	5.654867	-0.505666
2	0.000000	0.352810
3	2.513274	0.961297
4	4.398230	-0.981328
5	5.026548	-0.971700
6	0.628319	0.667817
7	6.283185	0.028809
8	3.141593	-0.195456
9	3.769911	-0.397768
10	1.884956	1.399235

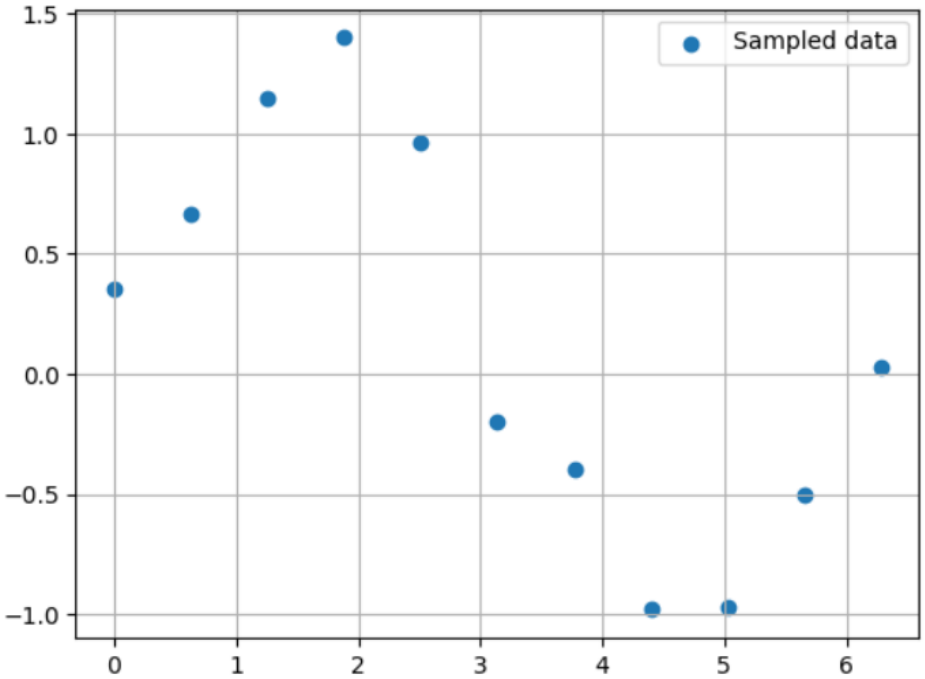
SML: Training Data

Given

Input-output pairs of the form:

$$S = (x_i, y_i)$$

$$x_i \in \mathbb{R}^K, y_i \in \mathbb{R}^M, i = 1, ..., N$$

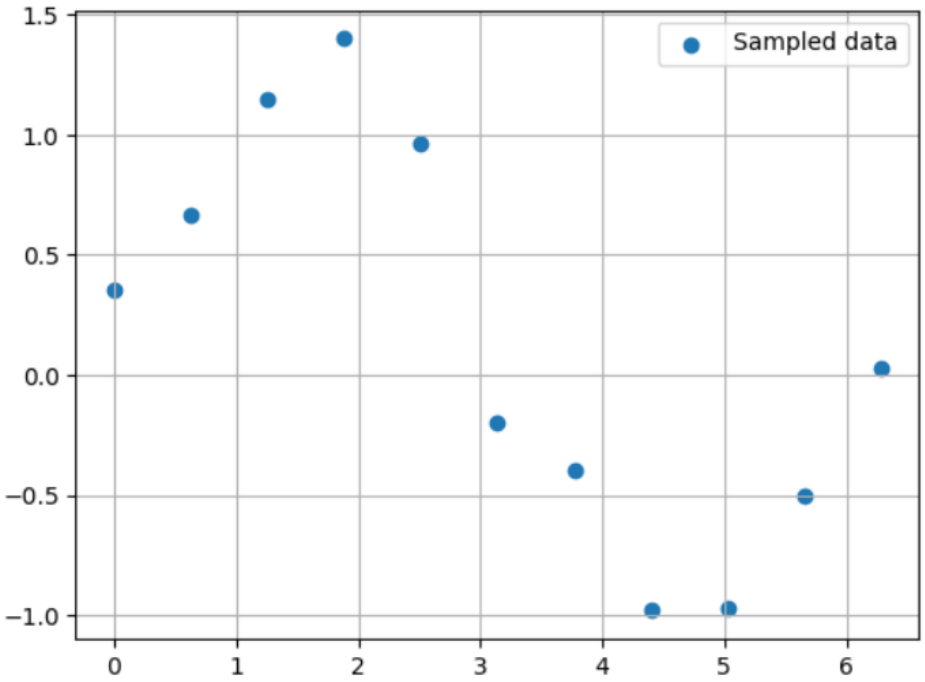


SML: Training Data

Assumption

Data is generated by a 'true' function f with some noise:

$$y_i = f(x_i) + \nu_i$$

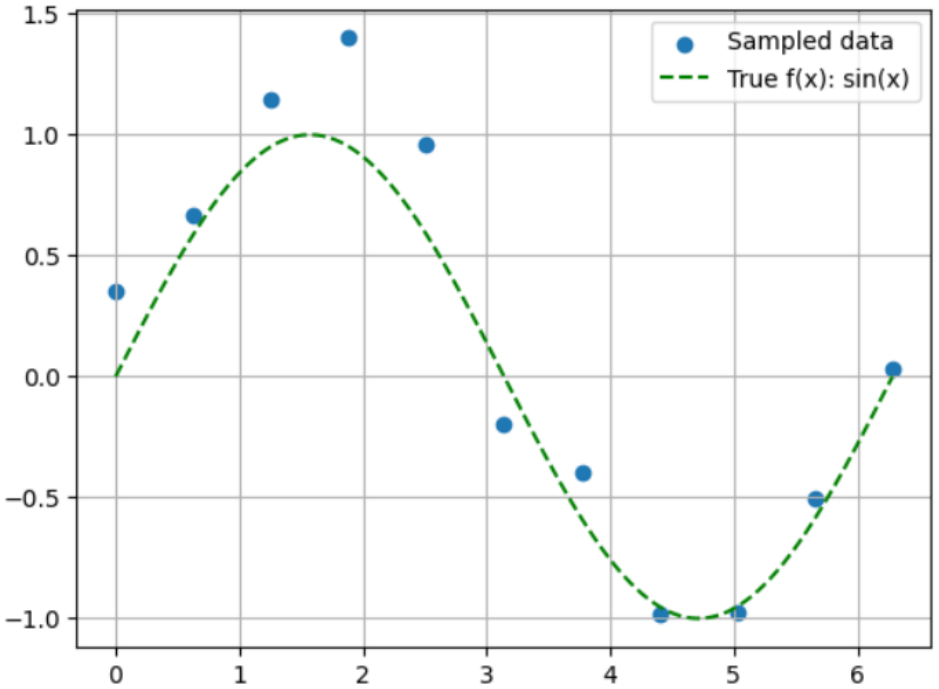


SML: Training Data

Assumption

Data is generated by a 'true' function f with some noise:

$$y_i = f(x_i) + \nu_i$$



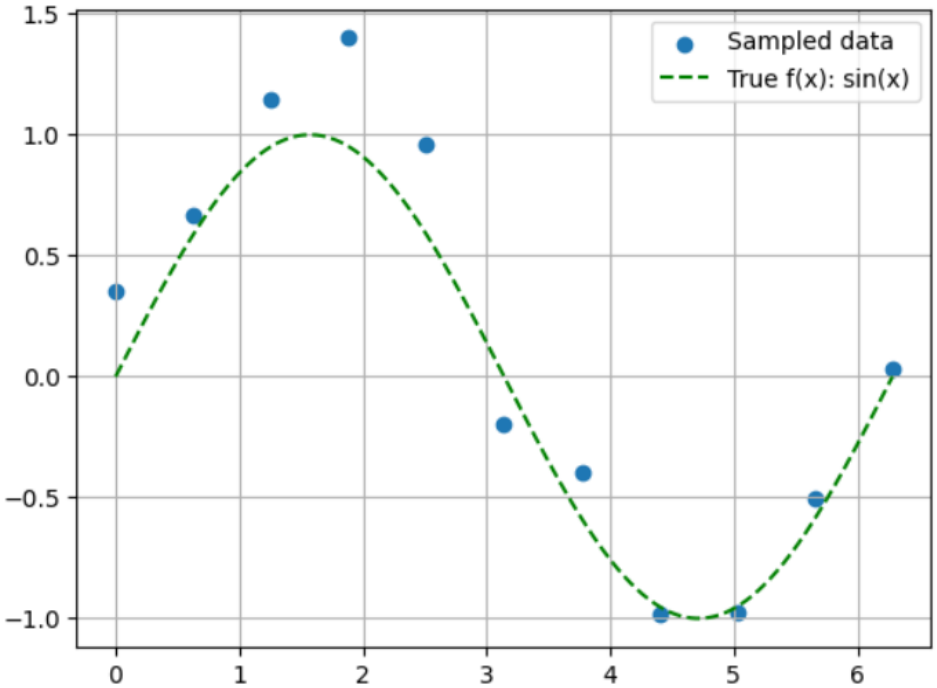
$$y_i = sin(x_i) + \nu_i,\ \nu_i \sim \mathcal{N}(0, \sigma),\ x \in [0, 2\pi]$$

SML: Goal

Goal

'Learn' an approximation $\hat{f}$ that is close to the real function f over all S:

$$\hat{f}(x_i) \approx f(x_i) \quad \forall i \in 1, ..., N$$



First attempt: Find a straight line $\hat{f}$ that minimises the *empirical risk*.

Linear Regression

Linear Regression: Building blocks

Hypothesis space $\mathcal{H}$:

Linear Regression: Building blocks

Hypothesis space $\mathcal{H}$: All functions of the form:

$$h(x) = ax + b$$

where $a \in \mathbb{R}$ and $b \in \mathbb{R}$ are called *parameters*, *coefficients* or *weights*.

Linear Regression: Building blocks

Hypothesis space $\mathcal{H}$: All functions of the form:

$$h(x) = ax + b$$

where $a \in \mathbb{R}$ and $b \in \mathbb{R}$ are called *parameters*, *coefficients* or *weights*.

Loss function: Standard in regression tasks to use a variant of squared error, a.k.a. 'L2 loss', e.g. Mean Squared Error:

$$MSE_{train} = \frac{1}{N} \sum_{i=1}^{N} (y_i - h(x_i))^2$$

Linear Regression: Building blocks

Hypothesis space $\mathcal{H}$: All functions of the form:

$$h(x) = ax + b$$

where $a \in \mathbb{R}$ and $b \in \mathbb{R}$ are called *parameters*, *coefficients* or *weights*.

Loss function: Standard in regression tasks to use a variant of squared error, a.k.a. 'L2 loss', e.g. Mean Squared Error:

$$MSE_{train} = \frac{1}{N} \sum_{i=1}^{N} (y_i - h(x_i))^2$$

Thus, we must find h_{opt} that minimises our **empirical risk**:

$$\hat{f} = h_{opt} = \arg\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} (y_i - h(x_i))^2$$

Re: What does it mean to *learn*?

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
>
> - Tom Mitchell

Linear regression:

E :

Re: What does it mean to *learn*?

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
>
> - Tom Mitchell

Linear regression:

E : Our data $S = (x_i, y_i)$

T :

Re: What does it mean to *learn*?

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
>
> - Tom Mitchell

Linear regression:

- E : Our data $S = (x_i, y_i)$
- T : Find line of best fit $\hat{f}$
- P :

Re: What does it mean to *learn*?

> *A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.*
>
> - Tom Mitchell

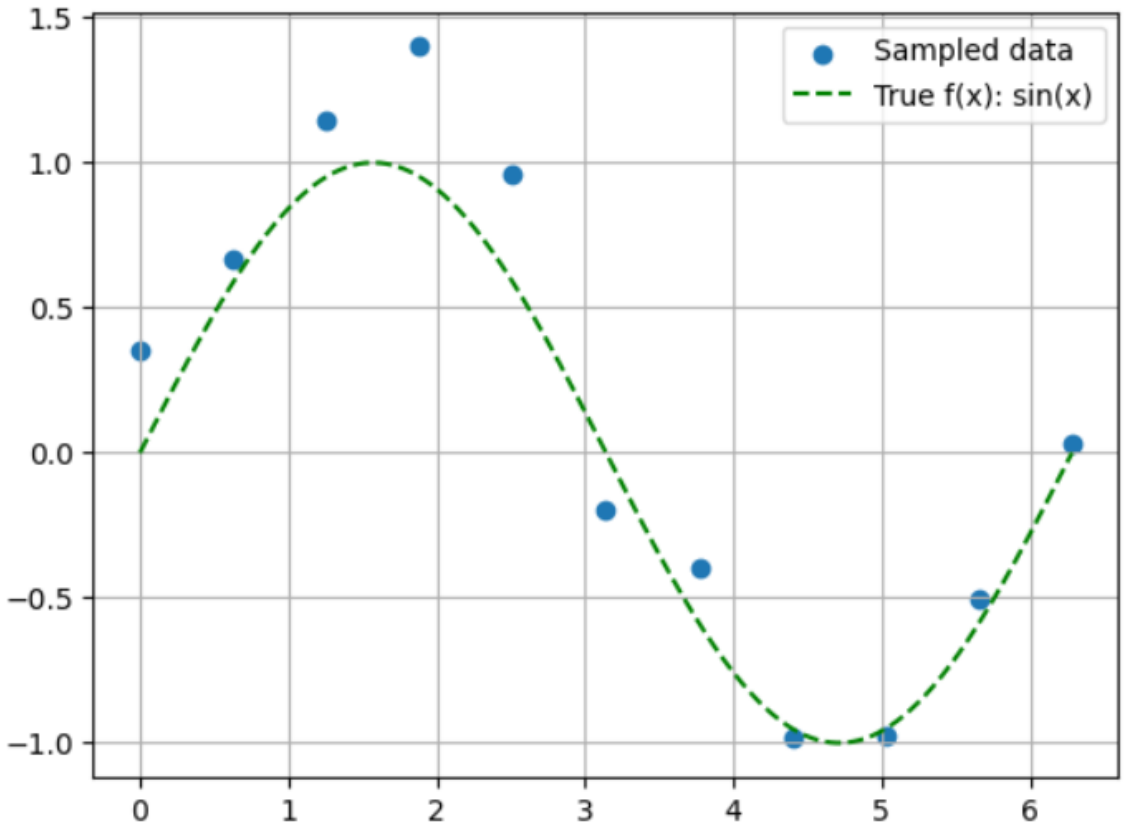
Linear regression:

- E : Our data $S = (x_i, y_i)$
- T : Find line of best fit $\hat{f}$
- P : Average distance between line and data points (i.e. MSE loss).

Linear Regression: Results

$$\hat{f}(x) = ax + b$$

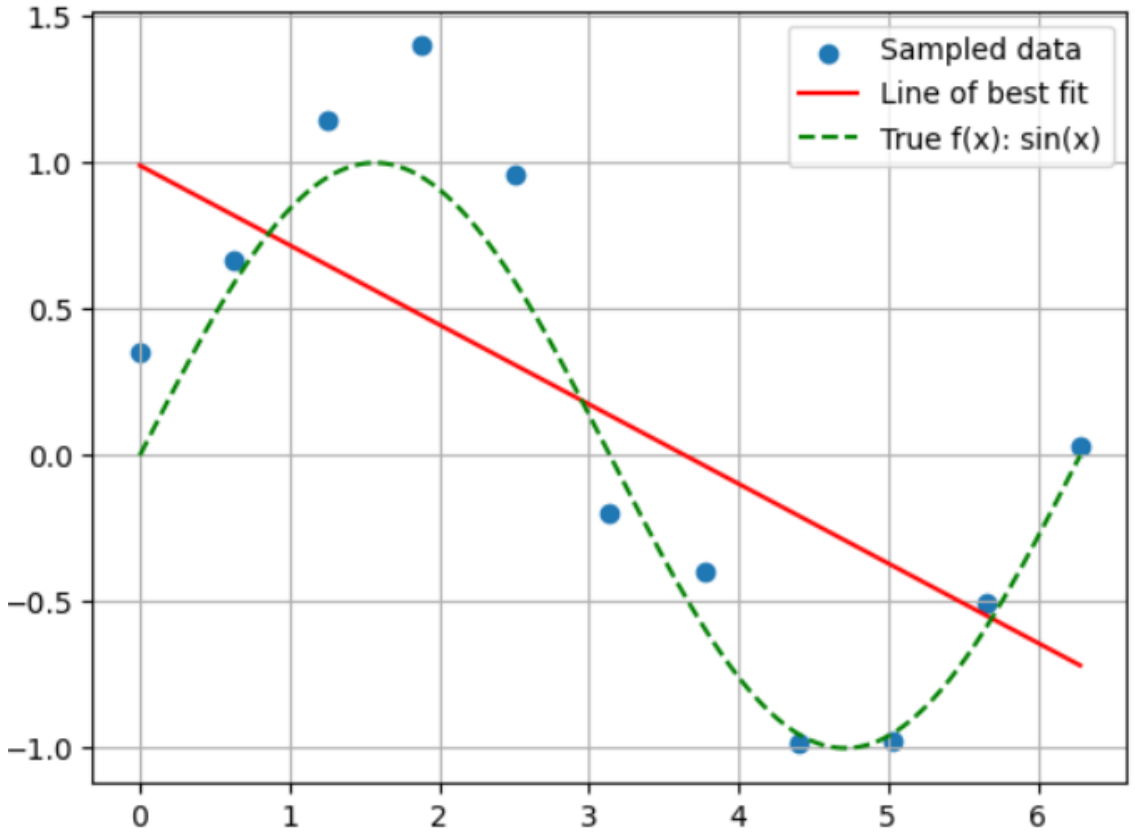
$$MSE_{train} = \mathbb{R}^{\geq 0}$$



Linear Regression: Results

$$\hat{f}(x) = -0.27x + 0.99$$

$$MSE_{train} = 0.3374$$



Can we do better?

Linear regression (visually) doesn't seem to model our training data too well. How do we do better?

Can we do better?

Linear regression (visually) doesn't seem to model our training data too well. How do we do better?

- Use polynomials instead!

Can we do better?

Linear regression (visually) doesn't seem to model our training data too well. How do we do better?

- Use polynomials instead!
- $\approx$Increase the size of $\mathcal{H}$!

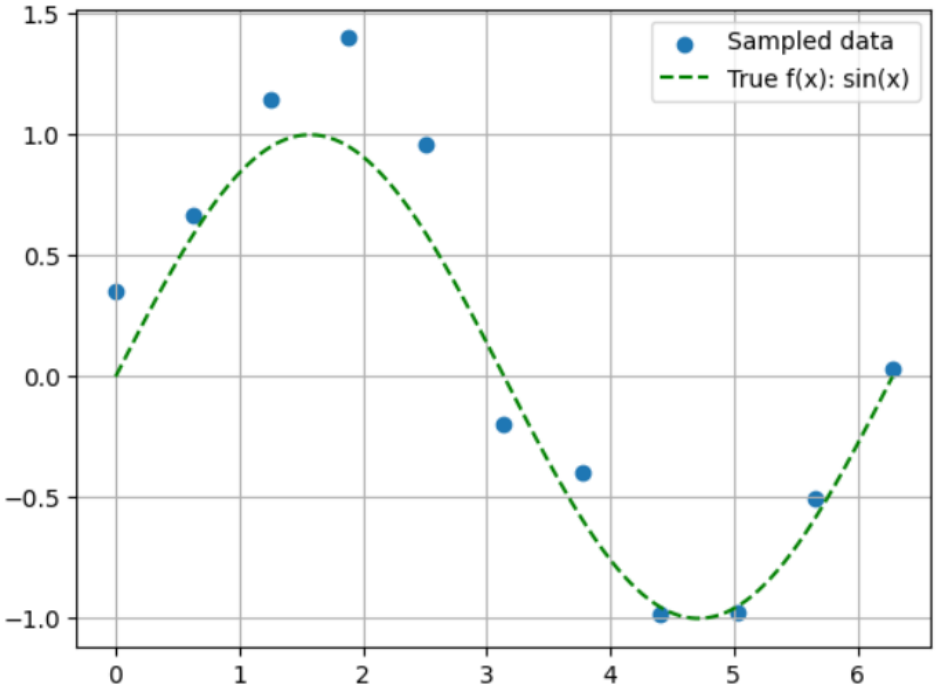
Polynomial Regression

SML: Goal Revisited

Goal

'Learn' an approximation $\hat{f}$ that is close to the real function f over all S:

$$\hat{f}(x_i) \approx f(x_i) \quad \forall i \in 1, ..., N$$



Next attempt: Find a polynomial $\hat{f}$ of degree d that minimises the empirical risk.

Polynomial Regression: Building blocks

Hypothesis space $\mathcal{H}$: All functions of the form:

$$h(x) = w_0 + w_1 x + w_2 x^2 + ... + w_d x^d$$

where $w_i \in \mathbb{R}$ for a polynomial of degree d.

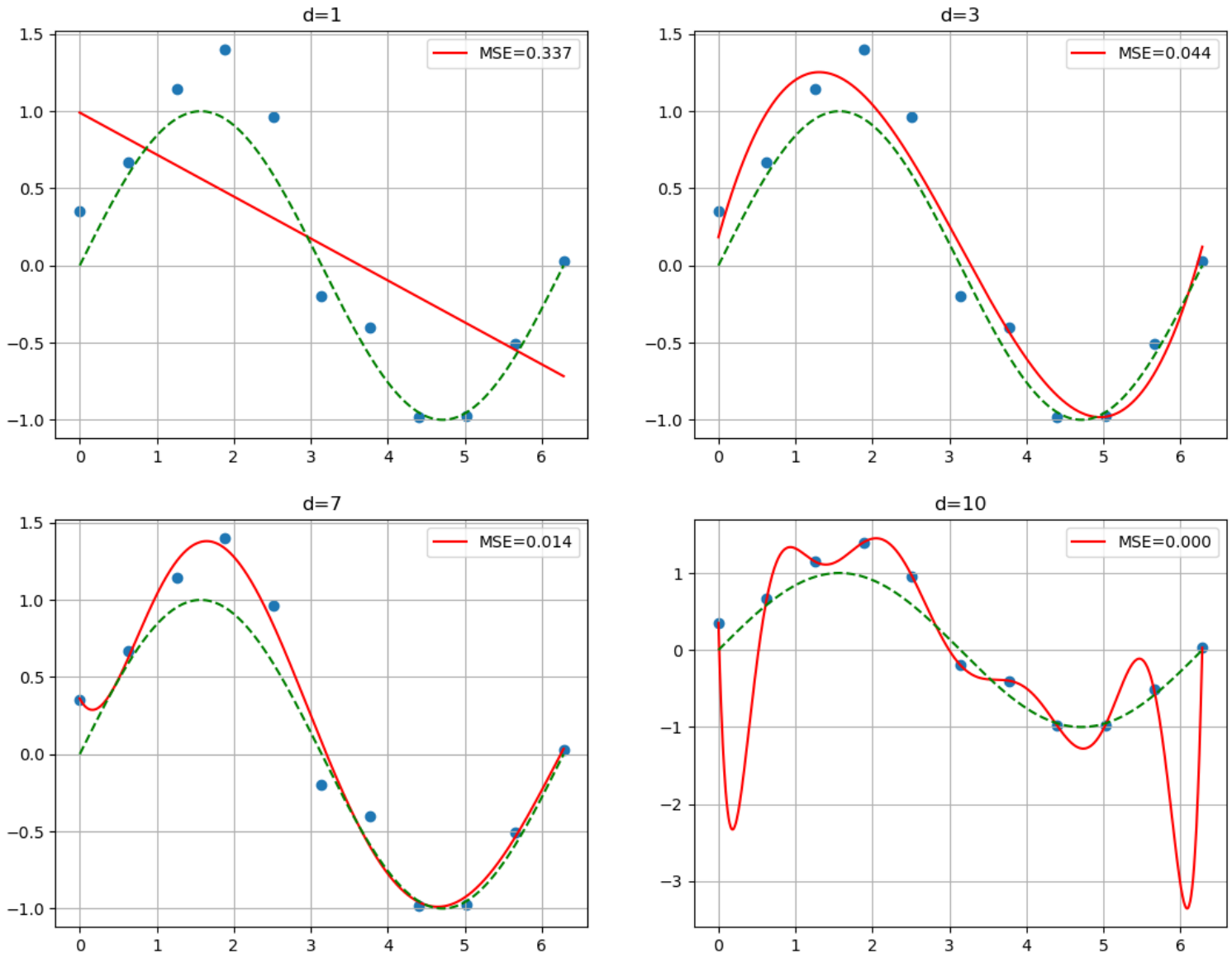
Loss function: Standard in regression tasks to use a variant of squared error, e.g. Mean Squared Error:

$$MSE_{train} = \frac{1}{N} \sum_{i=1}^{N} (y_i - h(x_i))^2$$

Thus, we must find h_{opt} that minimises our **empirical risk**:

$$\hat{f} = h_{opt} = \arg\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} (y_i - h(x_i))^2$$

Polynomial Regression: Results



Polynomial Regression: Analysis

Questions to think about:

- Which polynomial fits the data the best?
  - Which function minimises the *empirical risk*?

Polynomial Regression: Analysis

Questions to think about:

- Which polynomial fits the data the best?
  - Which function minimises the *empirical risk*?
- Which polynomial fits the *function* the best?
  - Which function minimises the *expected risk*?

Overfitting and Underfitting

Choosing a polynomial model

Q: What polynomial models the true function best?

Choosing a polynomial model

Q: What polynomial models the true function best?
Ans: Ideally, the one that minimies the *expected* risk.

Choosing a polynomial model

Q: What polynomial models the true function best?
Ans: Ideally, the one that minimies the *expected* risk.

Problem

In real-world scenarios, we usually do **not** have access to the true function we are trying to approximate!

Choosing a polynomial model

Q: What polynomial models the true function best?
Ans: Ideally, the one that minimies the *expected* risk.

Problem

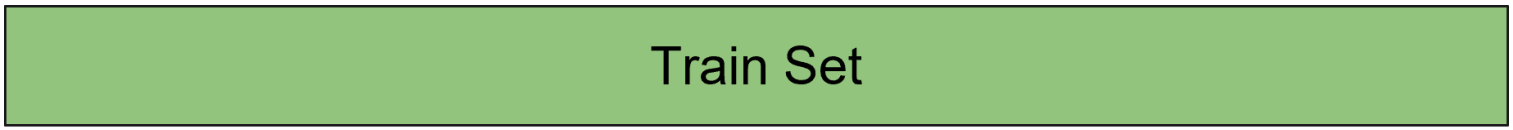
In real-world scenarios, we usually do **not** have access to the true function we are trying to approximate!

Solution (the most common)

Test the model on a set of data that you do not train a model with, that has the same distributional properties as the training data.

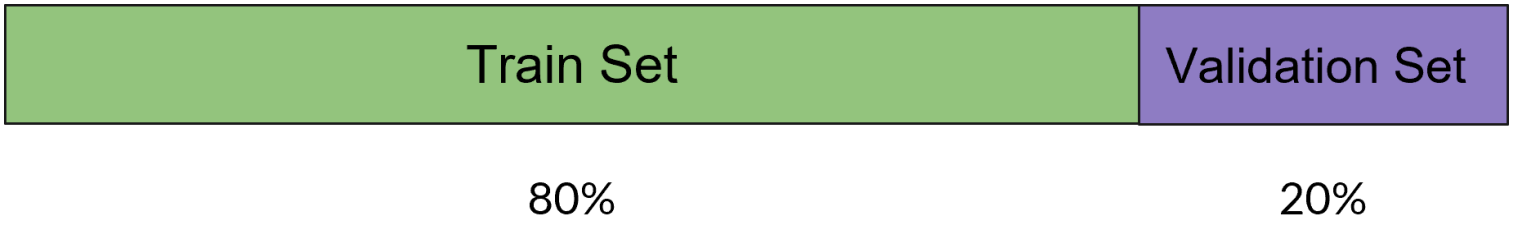
Train and Validation Sets

Assume we have a training set of N examples:

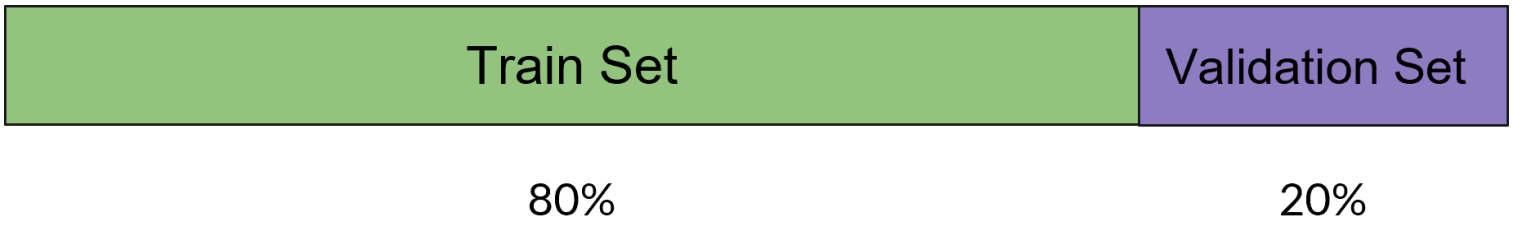


Train and Validation Sets

Now split it in two!

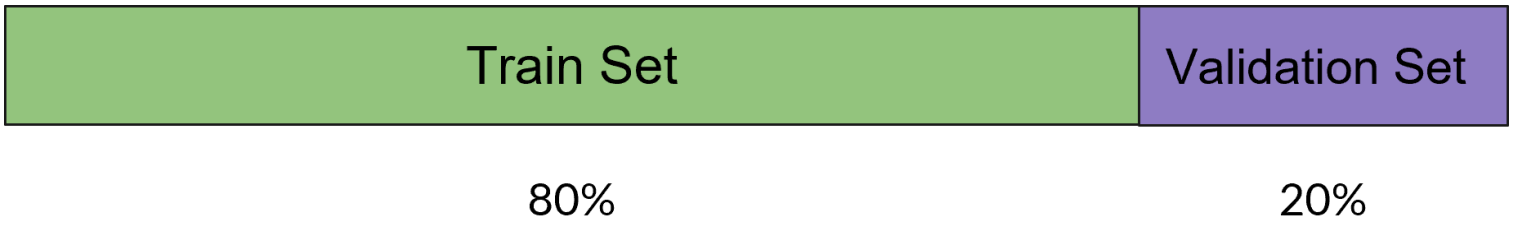


Train and Validation Sets



Use the new 'reduced' train set to train your model, and test the performance (read: '*compute the loss*') on the unseen **validation set** data.

Train and Validation Sets

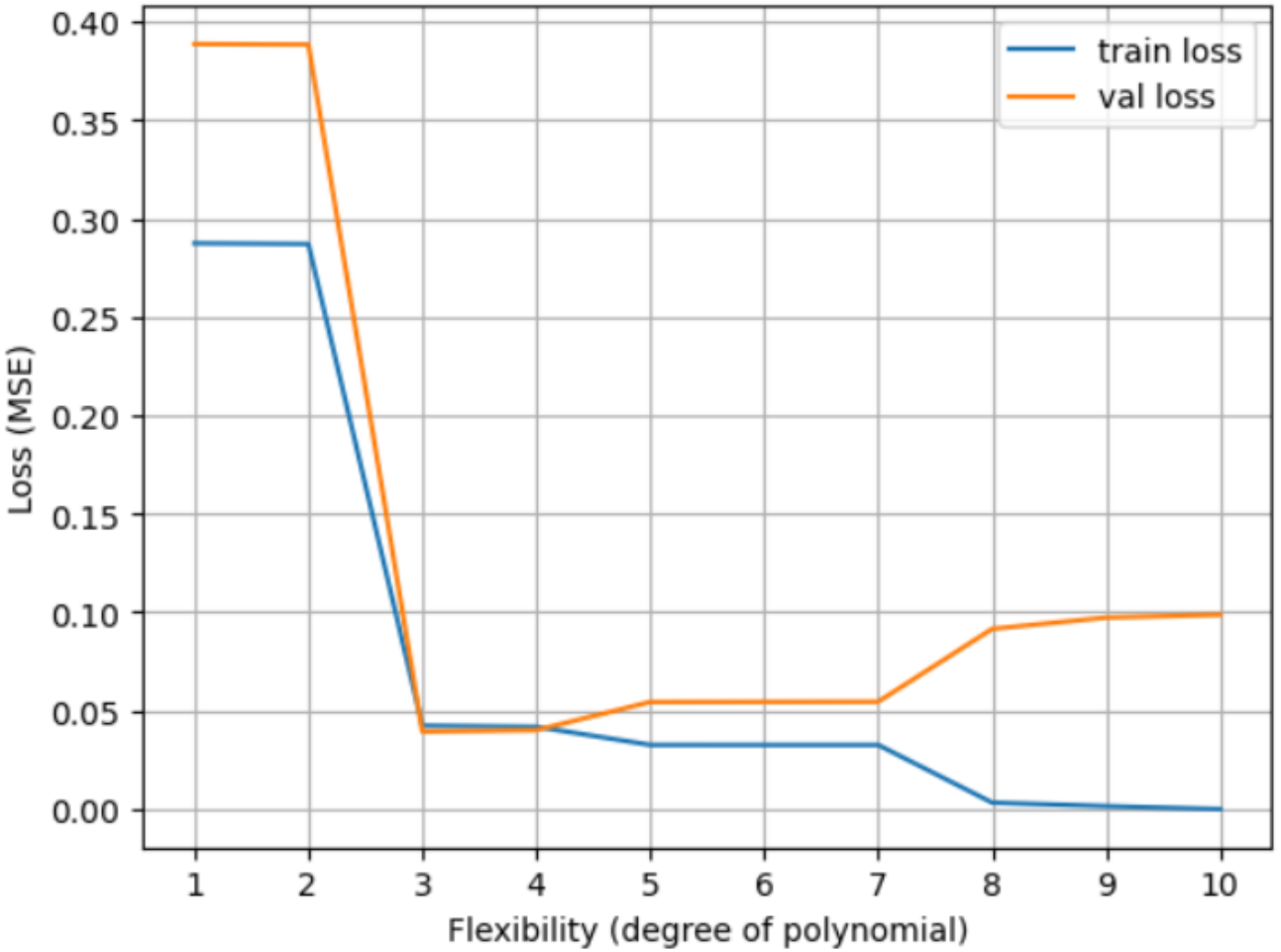


Use the new 'reduced' train set to train your model, and test the performance (read: '*compute the loss*') on the unseen **validation set** data.

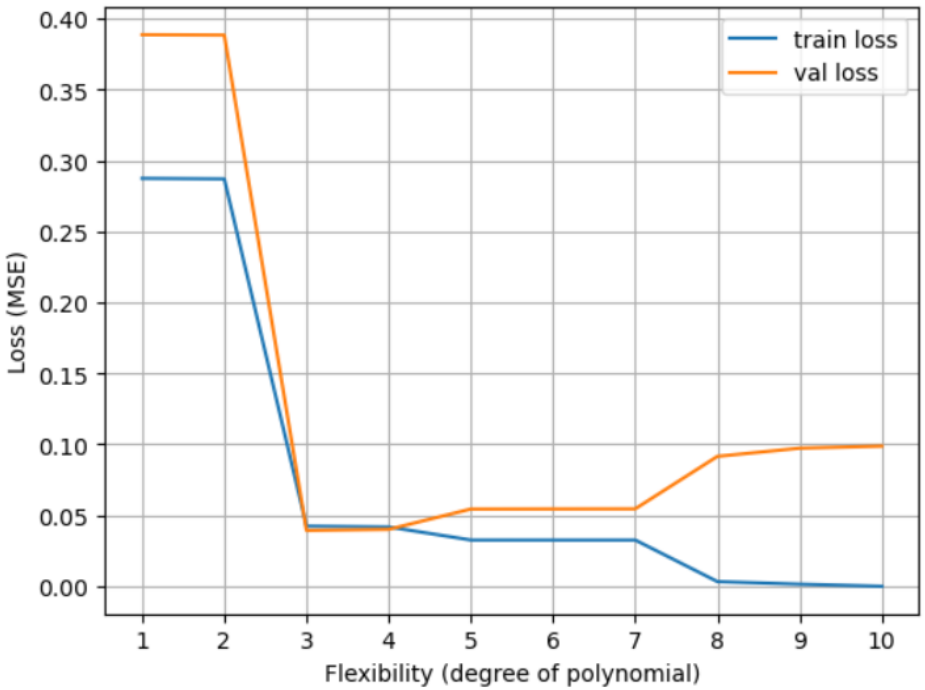
Note

Split ratio depends on model, task and data.

Train and Validation Losses



Overfitting and Underfitting



Underfitting

- The model does not fit the data well enough.
- Empirical risk is high, expected risk is high.

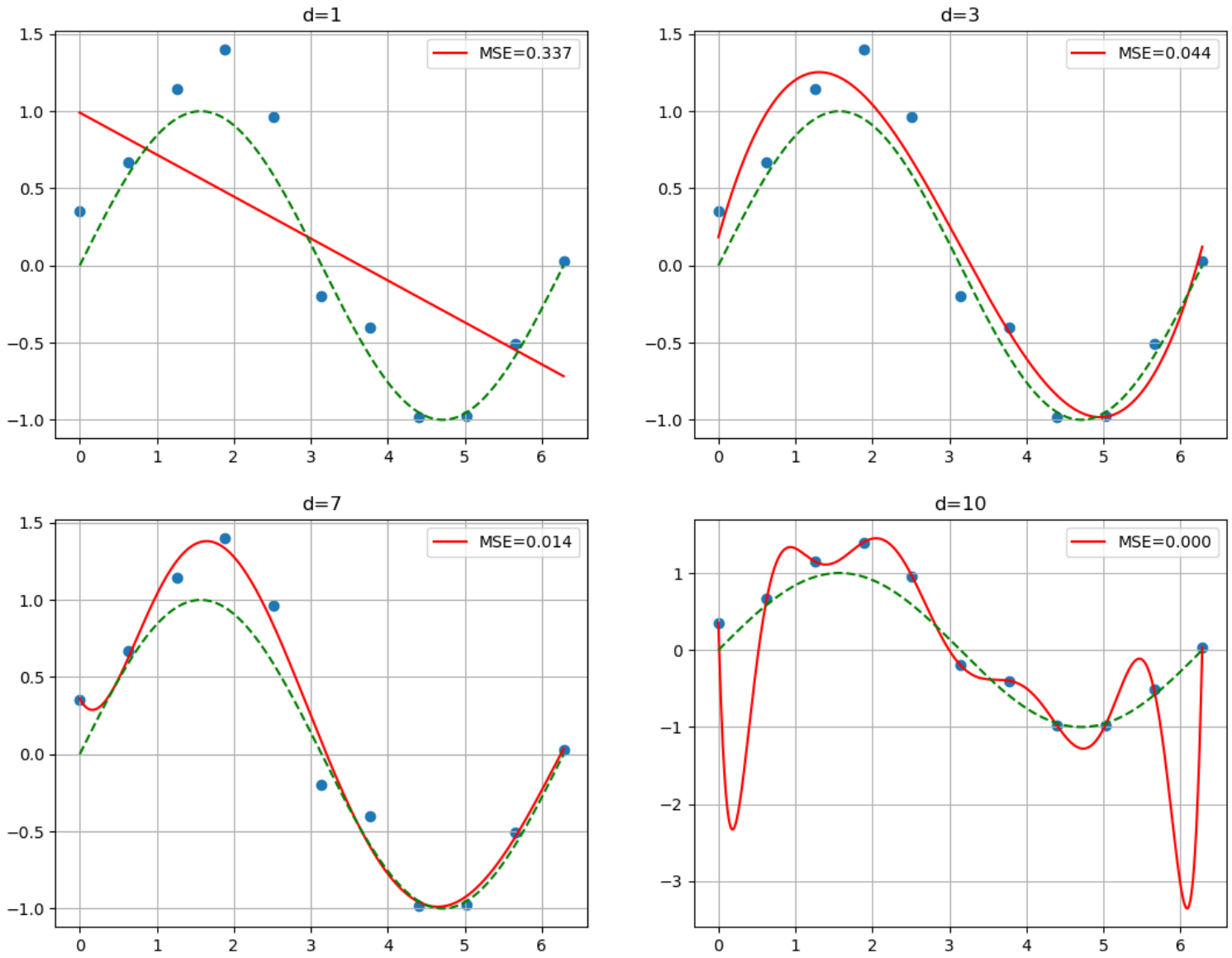
Balanced

- The model performs well on both data sets.
- Empirical risk is low, expected risk is optimal.

Overfitting

- The model fits the training data *too well*, including noise.
- Empirical risk is low, expected risk is high/not optimal.

Polynomial Regression: Over/underfitting



Assignment

Assignment 1

About the basic ML concepts we learned today:

- Linear+polynomial regression
- Over/underfitting
- Training and testing/validation datasets
- Recreating most of the plots in this lecture
- Math+programming practice

Due: 16/05/2023, 23:59

Semester Project

- Find a group of 3-4 and register on Canvas.
- Create a GitHub repository and share it with us.

Image credits

1. Slide 21: https://zappy.ai/ai-blogs/reinforcement-learning-how-intelligent-bots-are-built
2. Slide: 26: https://www.abc.net.au/news/science/2021-03-12/quokka-babies-daddy-long-legs-australian-animal-myths/13188740