UNIVERSITY of York
Europe Campus
CITY College

Introduction to Machine Learning

Dr Ioanna Stamatopoulou

All material in these lecture notes is based on our textbook:
Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow*, 3rd ed., O'Reilly, 2022

Welcome to the Machine Learning module!

Introductions:

- Who is who?
- Why are you here?
- What is this module all about?
  - How does ML fit in your programme curriculum?

Outline

- **What** is Machine Learning (ML)?
- **Why** ML?
- **How** does ML fit in the field of AI?
- **Types** of ML systems
- **Challenges** of ML

This session is covered by Chapter 1 of our textbook

What is Learning?

Definitions from Oxford Languages

- the acquisition of knowledge or skills through study, experience, or being taught
- knowledge acquired through study, experience, or being taught
- a thing learned by experience; a lesson
- change (in knowledge, in abilities, in behaviour) through a process

What is Machine Learning?

- "Machine Learning is the science (and art) of programming computers so they can *learn from data*" [A. Géron, 2019]
- "Machine learning is a subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon" [A. Burkov, 2019]
- "Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed" [A. Samuel, 1959]
- "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E" [T. Mitchell, 1997]

Throwback Thursday to AI Techniques and Kefalas...

The role of Classic AI

- Understanding the foundations: many of the concepts used in ML, such as search algorithms, knowledge representation and reasoning
- Problem-solving: Classic AI focuses on the development of algorithms and techniques for solving complex problems that may not have a clear or well-defined solution
- Explainable models: many ML are often referred to as "black boxes" because it can be difficult to understand how they make their predictions; classic AI models are often designed to be more interpretable and transparent
- Complementary approaches: Classic AI and ML are not mutually exclusive
- Historical significance: it can provide a deeper appreciation for the development of the field and the contributions of early researchers and pioneers

Example: The Spam filter

- A program that identifies/flags spam emails

- How would you build it based on what you already know?

Example: The Spam filter

- Traditional approach requires a long list of complex rules that needs to be constantly maintained

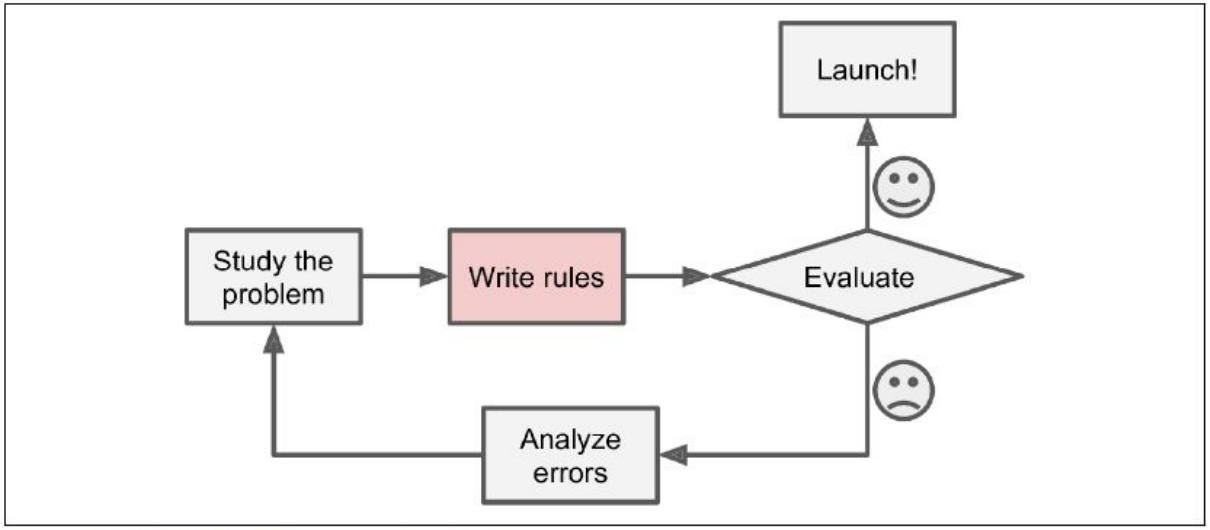


Figure 1-1. The traditional approach

- With the ML approach, the system, given examples of spam and normal emails, learns to identify/flag spam emails

Why Machine Learning?

How does it differ from traditional approaches?

1. Problems for which existing solutions require **a lot of fine-tuning or long lists of rules**: one ML algorithm can often simplify code and perform better than the traditional approach
2. Fluctuating environments: an ML system can **adapt to new data**
3. Complex problems for which using a **traditional approach yields no good solution**: the best ML techniques can perhaps find a solution
4. **Getting insights** about complex problems and large amounts of data

Reason 1

Problems for which existing solutions require a lot of fine-tuning or long lists of rules: one ML algorithm can often simplify code and perform better than the traditional approach

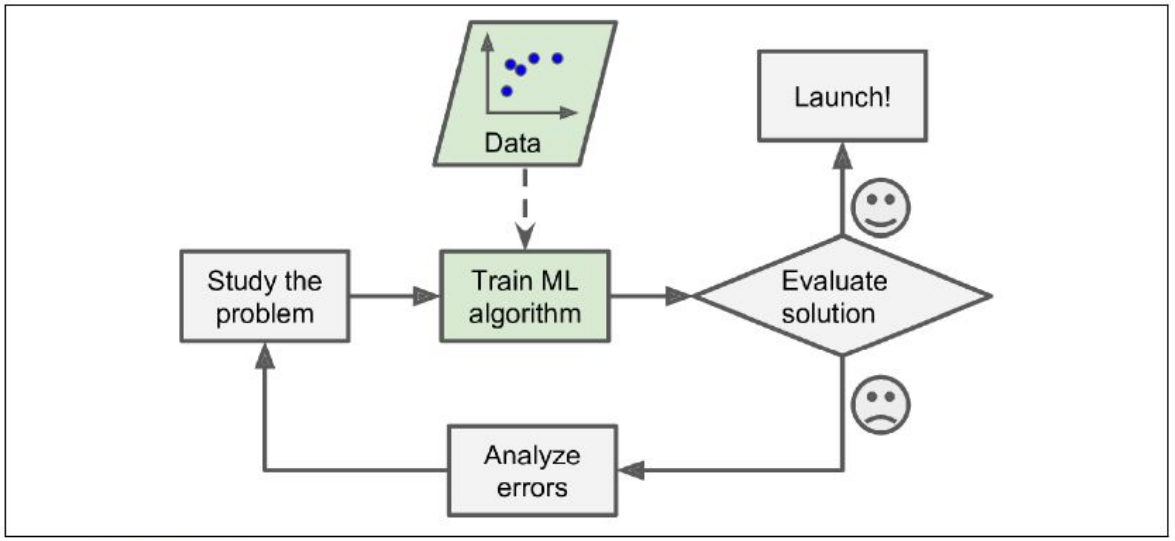


Figure 1-2. The Machine Learning approach

Reason 2

Fluctuating environments: an ML system can adapt to new data

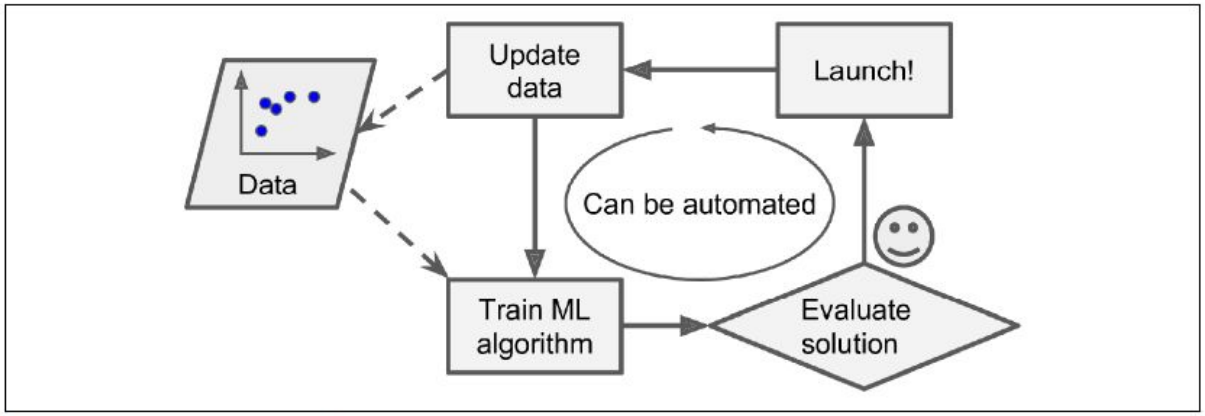


Figure 1-3. Automatically adapting to change

Reason 3

Complex problems for which using a traditional approach yields no good solution: the best ML techniques can perhaps find a solution

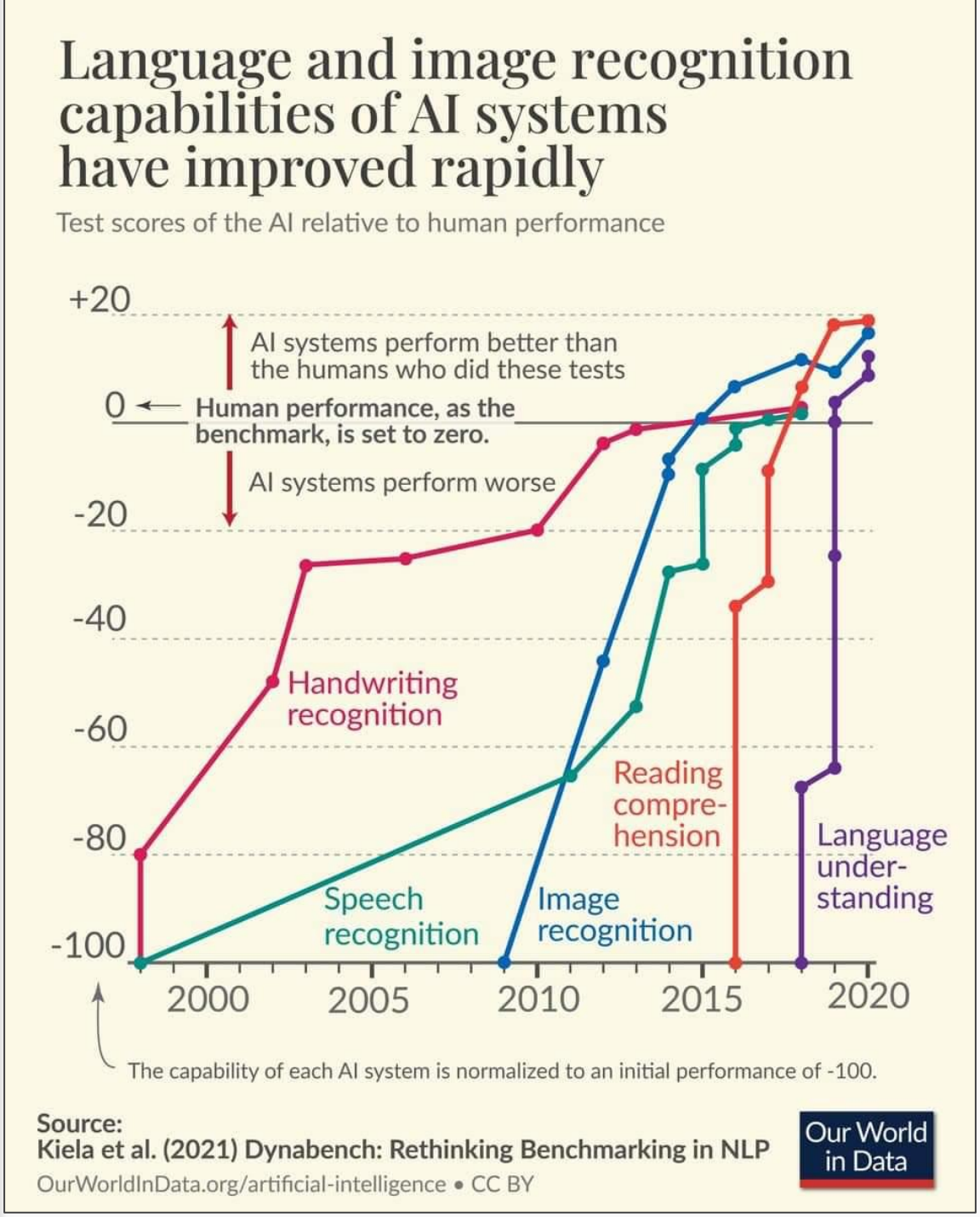
Consider a problem like Speech Recognition

How can you scale a solution to recognise the same word in all possible accents, with any possible background noise?

Only by **learning by examples**!

Reason 3

Complex problems for which using a traditional approach yields no good solution: the best ML techniques can perhaps find a solution



Reason 4

Getting insights about complex problems and large amounts of data

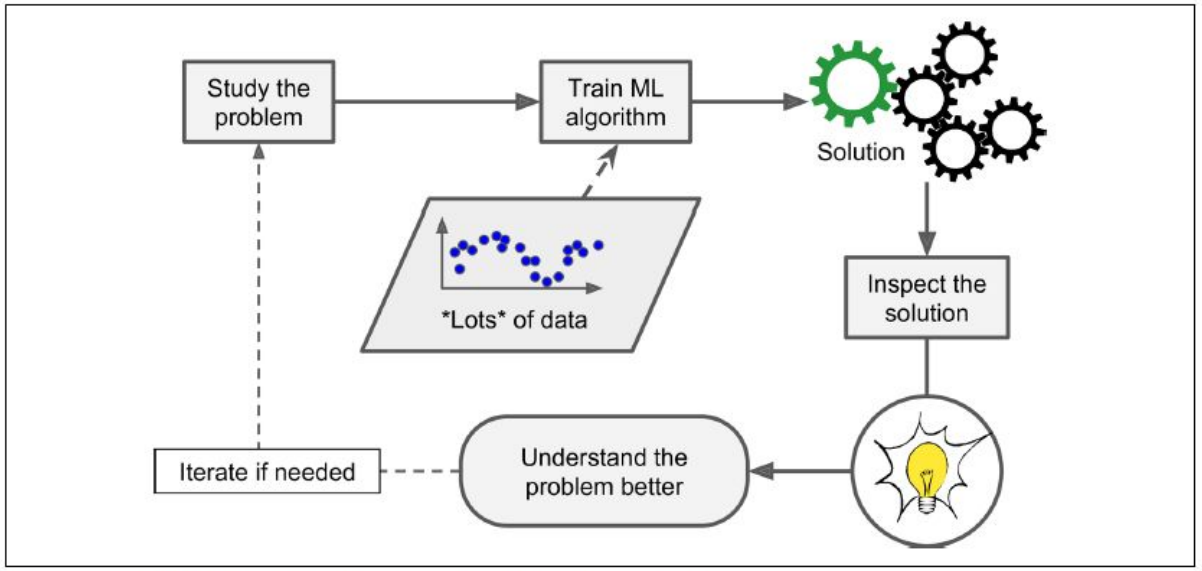


Figure 1-4. Machine Learning can help humans learn

Data mining is the application of ML techniques to large amounts of data for the purpose of identifying patterns that are not otherwise apparent

Examples of ML Applications

TASK	APPLICATION
Classification	Spam Filter
Reinforcement Learning	Building an intelligent bot for a game
Clustering	Grouping clients based on their purchases so that you can design a different marketing strategy for each group
Regression	• Forecasting your company's revenue next year, based on many performance metrics • Predicting the price of a car based on some features
Anomaly Detection	Detecting credit card fraud

**not including NN-related tasks/applications covered in the Deep Learning module*

Activity: Think of or google for applications relating to particular fields, such as healthcare, finance, entertainment, etc.

Types of ML Systems

Types of ML Systems

Categorisation based on three criteria

1. Human Supervision Required
   a. Supervised
   b. Unsupervised
   c. Reinforcement Learning
   d. Semi-supervised and self-supervised
2. Learning incrementally, on the fly (or not)
   a. Batch (offline) learning
   b. Online learning
3. How do they generalise?
   a. Instance-based learning
   b. Model-based learning

Terminology

Training Set or **Sample**
the set of all examples (training instances) that the system uses in order to learn

Training Instance
Each one of the examples in the Training Set

Feature or **Attribute**
An individual measurable property of an instance

Model
The part of the ML system that learns and makes the predictions

1 Supervision required

1a Supervised Learning

- The Training Set includes the solution, i.e. a **label** per training instance

- Out of all the **features** per instance, one is selected as the **class/target**

1a Supervised Learning

Typically used for the following TASKS:

Classification

- The Spam Filter system classifies emails as “spam” or “not spam”
- The email instances in the Spam Filter system Training Set are labeled as “spam” or “not spam”
- The label is also called the **class** (ergo classification…)

Regression

- Car Price Prediction system aims to predict a **target** numeric value (price) based on the values of various other features (mileage, age, brand)
- The Training Set contains instances that include their features as well their targets

1b Unsupervised Learning

- The Training Set is **unlabelled**

1b Unsupervised Learning

Typically used for the following TASKS:

Clustering

- You have gathered data about your blog's visitors
- A clustering algorithm can identify groups of similar visitors
- We do not guide the algorithm wrt these groups
- Hierarchical clustering can further subdivide each group into smaller ones

Anomaly Detection and Novelty Detection

- Anomaly: Detecting unusual credit card transactions to prevent fraud
- Novelty: Detecting new instances that are totally different from all instances in the training set

1b Unsupervised Learning

Typically used for the following TASKS:

Data Visualisation and Dimensionality Reduction

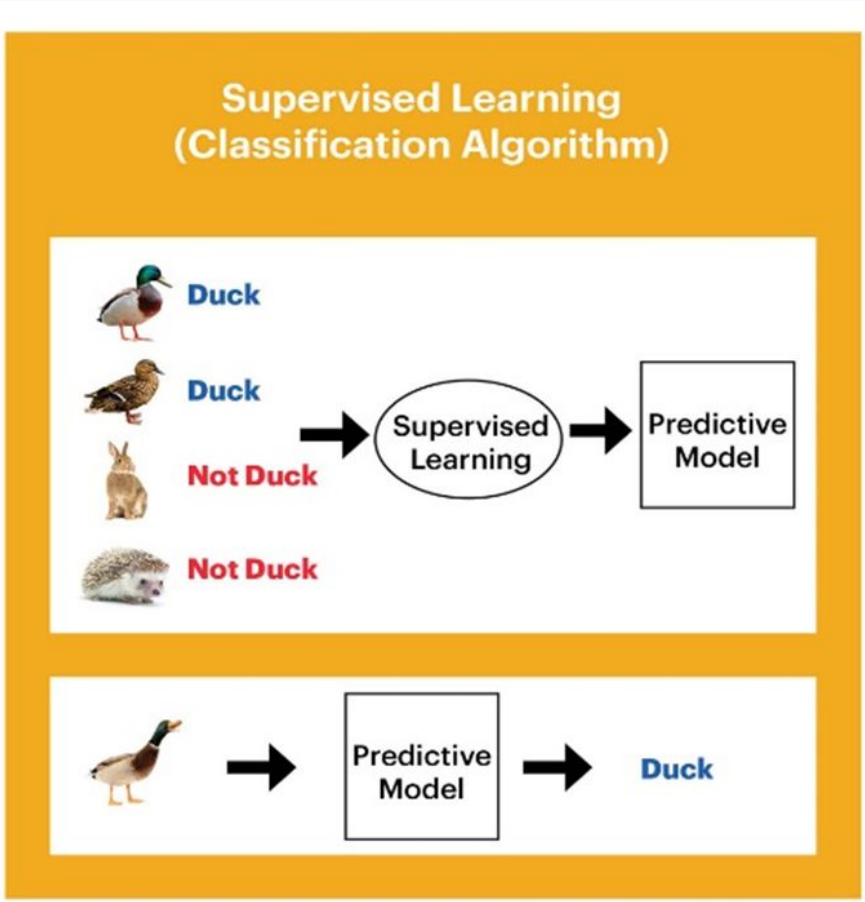
- Dimensionality Reduction* includes the processes of Feature Engineering that reduce the number of features used by the ML system
- e.g. Feature Creation: Identifying strongly correlated features that may be merged into one feature
- Example: car's mileage and age

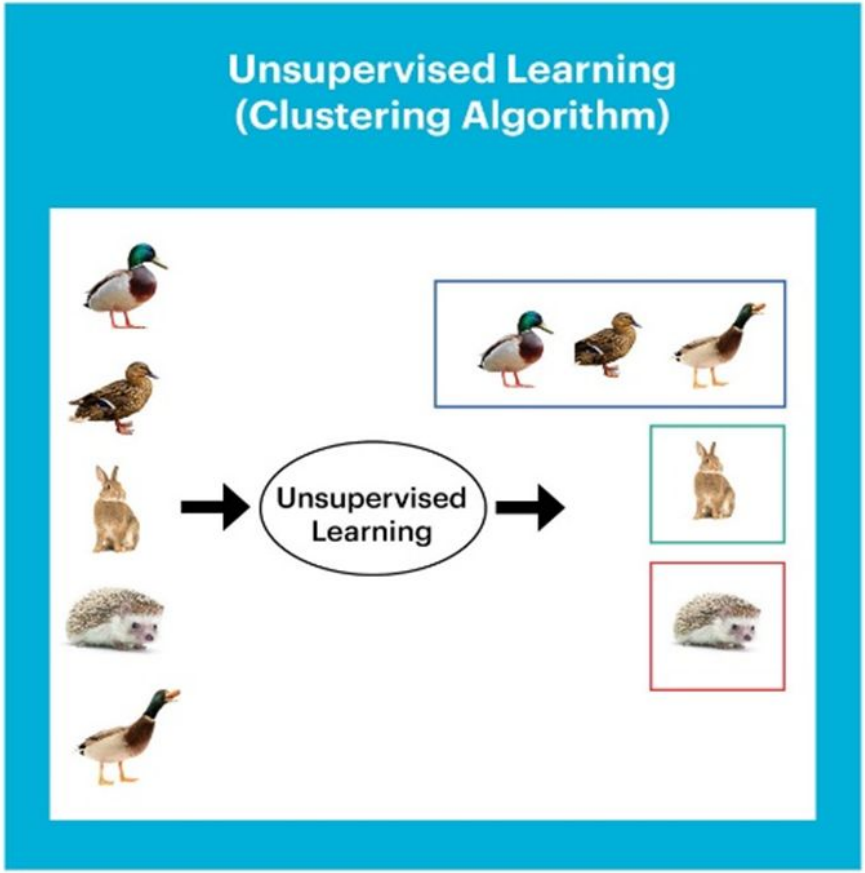
Association Rule learning

- Discovering interesting relations between attributes
- Example: which supermarket products are typically purchased together so that are physically positioned close one to the other

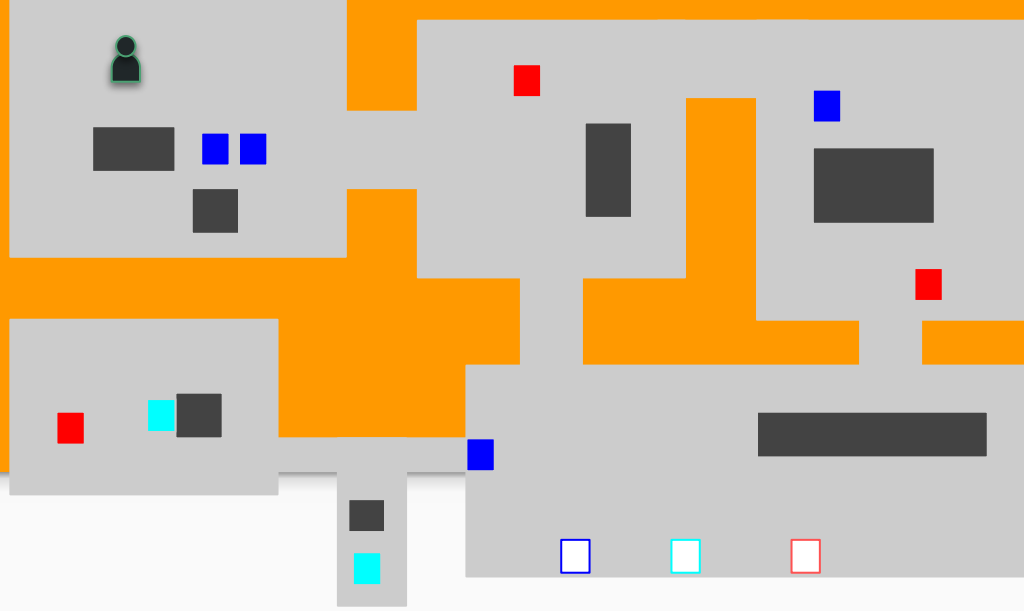
**for more on this, see Chapter 8 of the Textbook*

Supervised vs Unsupervised





1c Reinforcement Learning



- A very different beast!
- The learning system is an **agent** that
  - Observes the environment
  - Selects an action using some **policy**
  - Acts!
  - Gets **reward** or **penalty**
  - **Updates policy** (learning step, i.e. the goal is to learn the policy)
- …and iterates the above until an optimal policy is found

2 Learning on-the-fly or not

2a Batch (offline) Learning

- System is trained **offline** on **all** data **before** it is launched
- If new training data is available, you train the system on all data (old and new) again, and launch a new version
- Not suitable for when:
  - The full data set is of substantial size (increased time and computing resources required for the training)
  - The resources are restricted to begin with (e.g. mobile phone)
  - The system needs to adapt to rapidly changing data (e.g. predicting stock prices)

2b Online Learning

- The system is trained incrementally by feeding it sequential data instances (in mini-batches)
- Learning on-the-fly
- Great for systems that:
  - receive data as a continuous flow
  - have limited computing resources

2b Online Learning (cont'd)

- The **challenge** with online learning
  - If bad data is fed to the system (e.g. a malfunctioning sensor), performance will decline
  - Solution: react to abnormal data by using an *anomaly detection algorithm*
- **Learning rate**: the speed of adapting to new data
  - High: adapts to new data quickly, "forgets" the old data
  - Low: learns more slowly but is less sensitive to **outliers**

2b Online Learning (cont'd)

Variation: Out-of-core Learning

- Takes place **offline** (weird, I know… Think of it as **incremental** learning)
- Useful for huge datasets (cannot fit on one machine's memory)
- Data is split into pieces
- Training algorithm runs on each data part separately
- Process is repeated until it has run on all of the data

3 How do they generalise?

3a Instance-based Learning

- In instance-based learning, a **measure of similarity** is selected
  - Example: in the Spam Filter system, counting the number of common words
- The system generalises by using this measure to compare new instances to the learned examples

3b Model-based Learning

Table 1-1. Does money make people happier?

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

- Generalise by building a **model** of the examples and use the model to make predictions about new instances
- Example: predicting life satisfaction

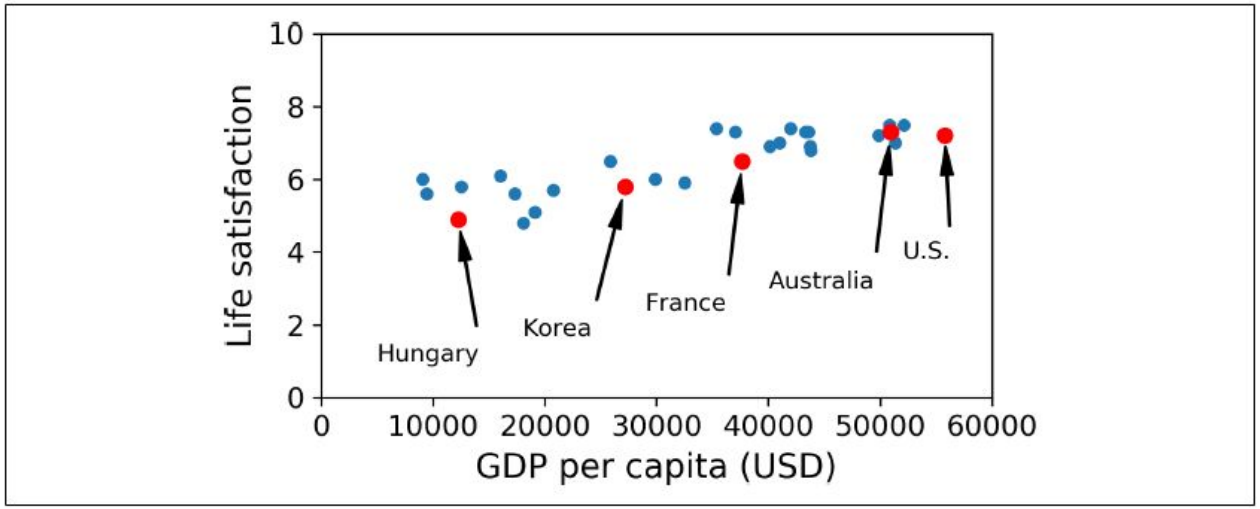


Figure 1-17. Do you see a trend here?

3b Model-based Learning (cont'd)

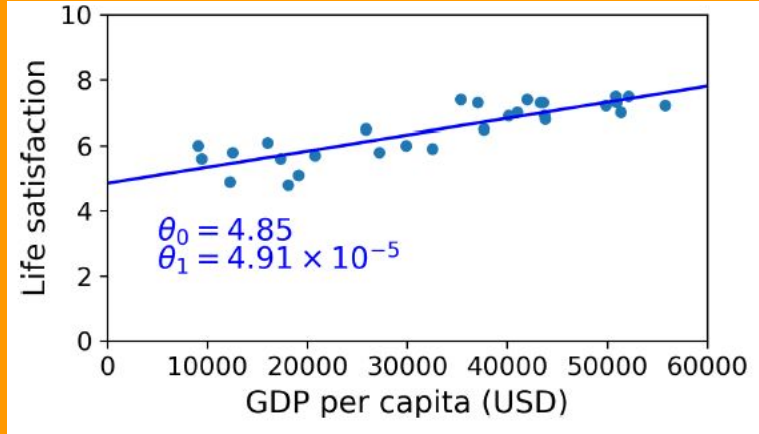


Figure 1-19. The linear model that fits the training data best

- Select type of model, e.g. linear:

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

- By tweaking the model parameters θ_0 and θ_1, you can represent any linear function
- Select a **performance measure**. Typically :
  - a **fitness function**: measures how *good* your model is
  - A **cost function**: measures how *bad* your model is, i.e. the distance between the predictions and the training examples
- **Train the model**: use linear regression algorithm to find the θ parameters that make the model fit best to the data

Challenges of ML

Challenges of ML

Insufficient Quantity of Training Data

- We typically need
  - thousands of examples for moderate complexity problems
  - millions of examples for complex problems such as image or speech recognition
- Evidence suggests that various different algorithms can perform equally well when trained with enough data:
- **The Unreasonable Effectiveness of Data**! [Halevy, Norvig, and Pereira, 2009]

see also the No Free Lunch Theorem at the end of this session

Challenges of ML

Nonrepresentative Training Data

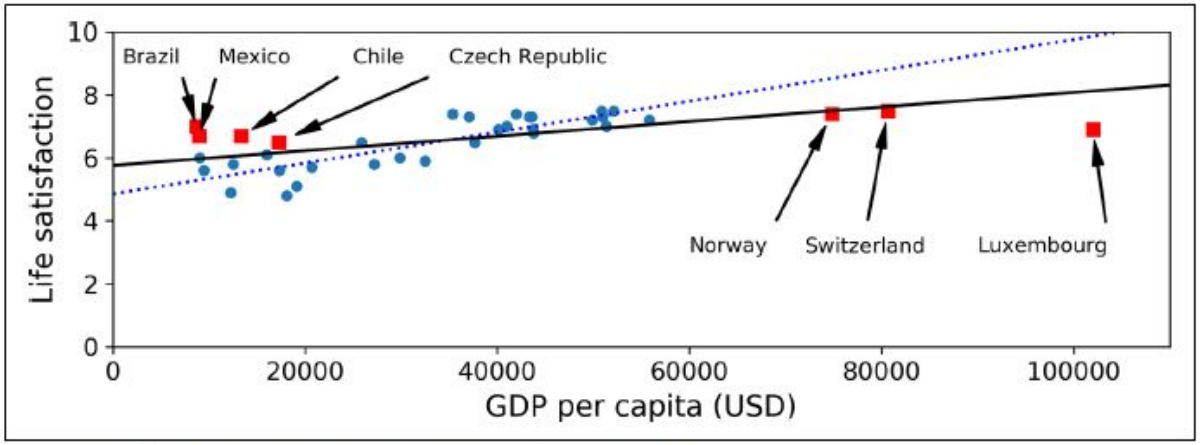


Figure 1-21. A more representative training sample

- In the above
  - the dotted line is the "old" model
  - the solid line is the one that used more representative training data
- Non-representative training sets will make poor predictions (particularly for the extreme cases: very poor, very rich countries)
- The Training Set must be **representative of the cases you want to generalise to!**

Terminology

Sampling Noise
Undesirable, incorrect, even **garbage data** in our Training Set (a lot of times as a result of chance) that must be removed

Sampling Bias
Even in large samples,
bias may exist as a result of a flawed sampling method

Sampling Method
The method used to extract/select the subset of the all the data that will be included in the sample

Outlier
A **valid instance** in the Training Set that lies an abnormal distance from other values in a sample

Example: Sampling bias
(the 1936 US presidential elections and the *Literary Digest* poll)

https://www.youtube.com/watch?v=R2vhjC5qCQk

Example: Sampling bias
(the 1936 US presidential elections and the *Literary Digest* poll)

- Two major sampling problems with the *Literary Digest* poll:
  - The use of telephone directories and car registries for the **selection of the sample** (sampling method) favoured wealthier people in the sample, which were more likely to vote Republican
  - The **nonresponse bias**: there were many wealthy voters with telephones and cars who backed the Democrats (Franklin D. Roosevelt) but chose not to participate in the poll

Example: Outlier

- In a small sample of people's annual income, add Elon Musk
  - The data about Elon is valid
  - But as a result the entire sample is on average millionnaires

Challenges of ML

Poor-Quality Data

- Poor-quality is the data that contains
  - Noise
  - Outliers
  - Missing Data
- **Cleaning up the sample** is a very important process
- Cleaning up examples:
  - In the case of noise either discard them or fix them manually
  - In the case of instances missing a feature (or more):
    - ignore the instances
    - ignore the entire feature
    - fill in the missing values (e.g. with the median value)

Challenges of ML

Irrelevant Features

- **Garbage in, garbage out!**
- **Feature Engineering** is the process of selecting a good set of features
- It is a very critical part of the success of an ML project
- It includes (among others):
  - Feature Selection: selecting the most useful features to train on
  - Feature Creation: combining existing features to produce a more useful one
  - Feature Extraction: transforming raw data (e.g. from images, sound signals) into numerical features that can be processed

Challenges of ML

Overfitting the Training Data

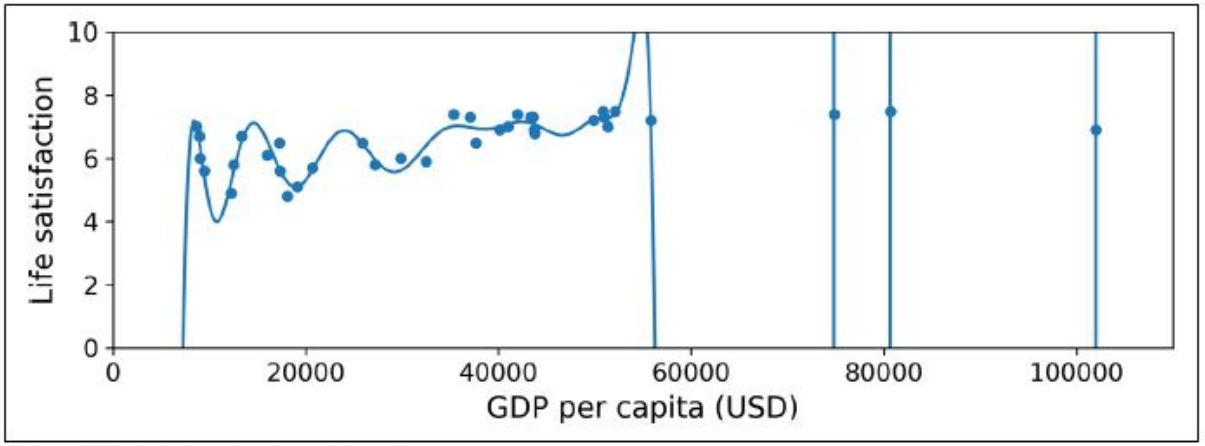


Figure 1-22. Overfitting the training data

- Overfitting is the result of a very **complex model**
- It means that:
  - The model performs well (outstandingly even) on the training data
  - But does not generalise well, i.e. does not perform well on new data
- Solution: **Regularisation**

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

Regularisation

The process of simplifying a model (to avoid overfitting)

- The linear model we saw has two parameters:
  - θ_0, the height of the line
  - θ_1, the slope of the line
- Model parameters are called **degrees of freedom**
- By tweaking the model parameters we can adapt the model to the training data

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

Regularisation (cont'd)

- Consider that θ_1 becomes 0:
  you are left with one degree of freedom, therefore
  - A line parallel to the x-axis
  - A much less flexible model
- Consider that θ_1 is restricted to a small value:
  you are left with (let's say) 1½ degrees of freedom…
  - A line with a restricted slope
  - A moderately flexible model

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

Regularisation (cont'd)

- Regularisation is about finding the right **balance** between
  - Fitting the training data perfectly, and
  - Keeping the model simple enough to generalise well

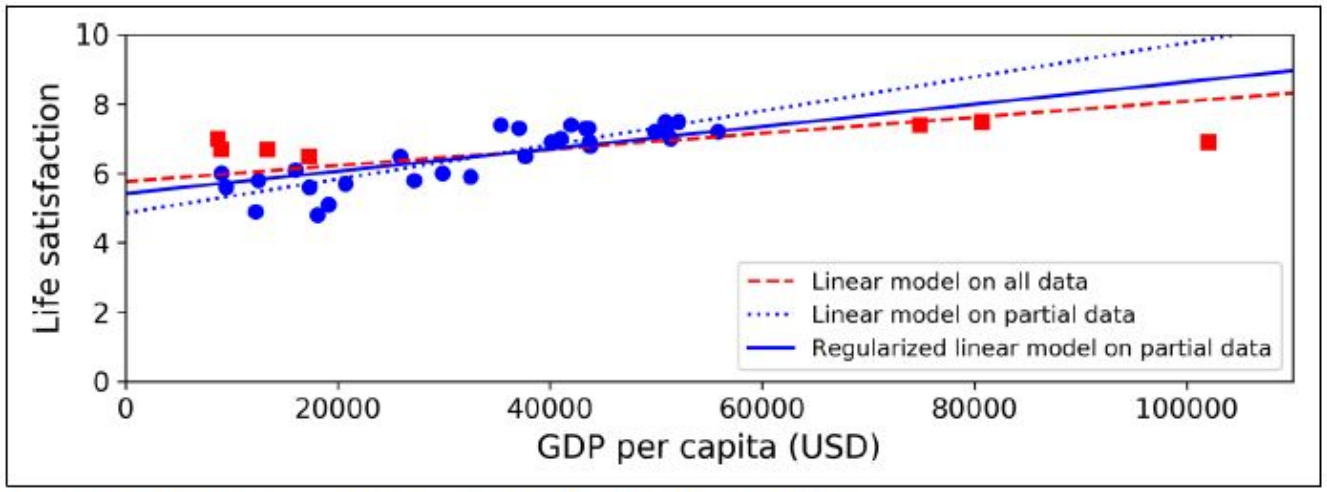


Figure 1-23. Regularization reduces the risk of overfitting

Regularisation (cont'd)

Hyperparameters

Equation 1-1. A simple linear model

$$\text{life\_satisfaction} = \theta_0 + \theta_1 \times \text{GDP\_per\_capita}$$

- A hyperparameter is something that can control the amount of regularisation while learning
- **A hyperparameter is a parameter of the learning algorithm**, not of the model

Higher hyperparameter
⇒ Higher regularisation
⇒ Simpler model
(e.g. θ_1 close or equal to 0)

Challenges of ML

Underfitting the Training Data

- Underfitting is a result of a very **simple model** that cannot learn the underlying structure of the data
- Reality is more complex that the model…
- The model underperforms even on the training data
  - e.g. the original linear model for life satisfaction
- How to fix:
  - Increase degrees of freedom, i.e. the complexity of the model
  - Reduce regularisation hyperparameter
  - Feature Engineering

Model Testing
Model Evaluation, and
Fine-Tuning

Testing your model

- Evaluating your model after it goes live is a bad idea
- Better option: split your data into a Training Set and a **Test Set**
- After training, test your model on the Test Set
- This will give you an indication of how well your model will perform on cases it has not seen before

Terminology

Test Set
the set of instances that you run the model on, after training, to evaluate its performance

Training Error
The error rate of your model
on the instances of the Training Set

Generalisation Error or **Out-of-Sample Error**
The error rate of your model
on the new instances of the Test Set

Training/Test Set Size Ratio
Rule of Thumb:

- 80% of available data for Training
- 20% of available data for Testing

Testing your model

What is the challenge?

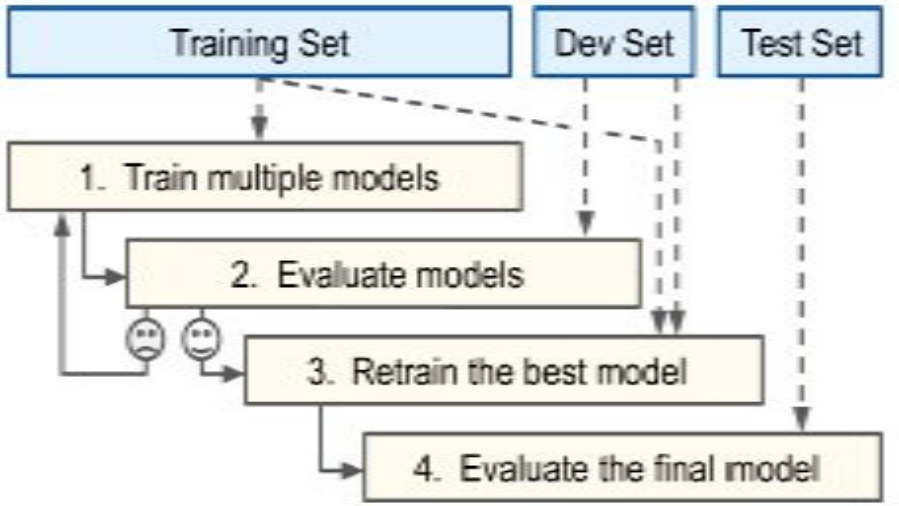
- Model Selection
  - Train all candidate models on the Training Set
  - Test them on the Test Set
- Choose the best model and apply regularisation to avoid overfitting
  - Train various versions of the model with different values for the hyperparameter
  - Test all of them on the Test Set
  - Choose the model with the hyperparameter value that results in the lowest generalisation error
- **Problem! You have selected the model that is best adapted *to the particular* Test Set!**

Holdout Validation of the model

An answer to the testing challenge

- **Train** various versions of the model with different values for the hyperparameter **on a reduced training set** (complete training set minus **validation set**)
- Choose the model that performs best on the validation set
- **Retrain** the model on the complete training set
- **Test** the model on the test set

Terminology



Validation Set or **Dev Set**
Part of the training set, which you put aside only for validating the several candidate models

Training/Validation/Test Set Size Ratio
Rule of Thumb:

- 60% of available data for Training
- 20% of available data for Validation
- 20% of available data for Testing

The Differences Between Training, Validation & Test Datasets

Training, Validation, Test Split for Machine Learning Datasets

No Free Lunch Theorem

[David Wolpert, 1996]

- When you select a model, you are making an *assumption* about the data
  - Choose a linear model, you are assuming data is linear
- Make no assumptions about the data

  ⟹ There is no reason to prefer one model over another!
- There is no model that is *a priori* guaranteed to work better!

Thank you!

Coming up next:
Reinforcement Learning