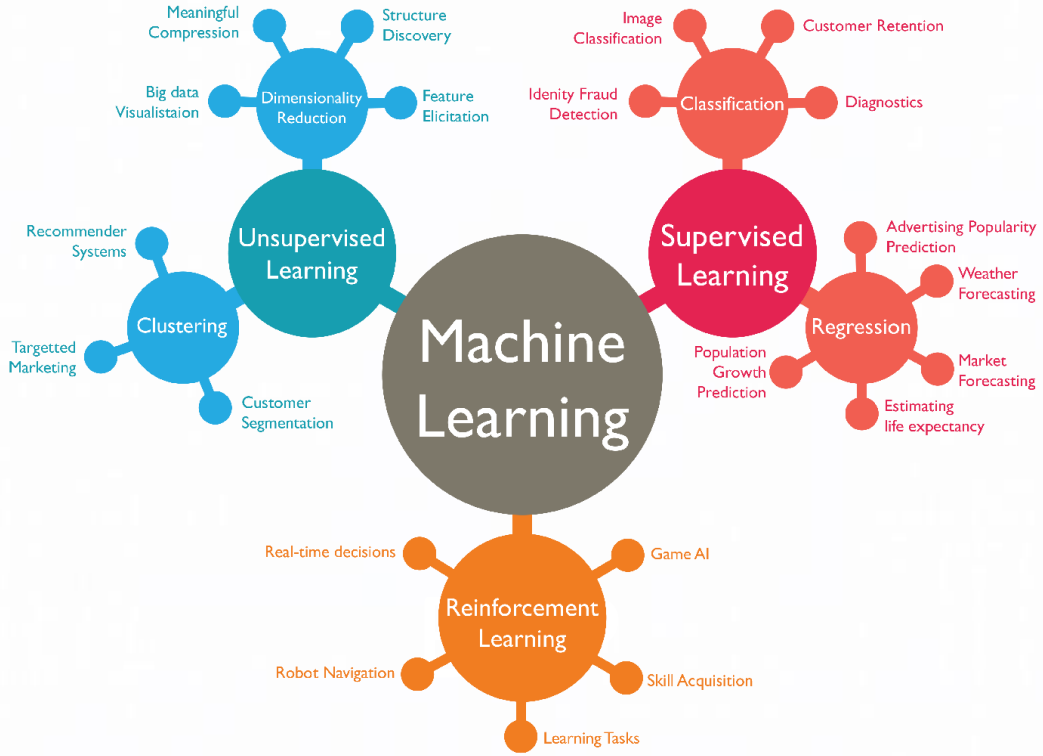
AI in Biomedical Data

Dr. M.B. Khodabakhshi
Amir Hossein Fouladi
Alireza Javadi

github.com/mbkhodabakhshi/AI_in_BiomedicalData



یادگیری ماشین در زیست پزشکی

Chapter 1. The Machine Learning Landscape

دکتر محمدباقر خدابخشی

mb.khodabakhshi@gmail.com

Presenter: Dr.Khodabakhshi

What Is Machine Learning?

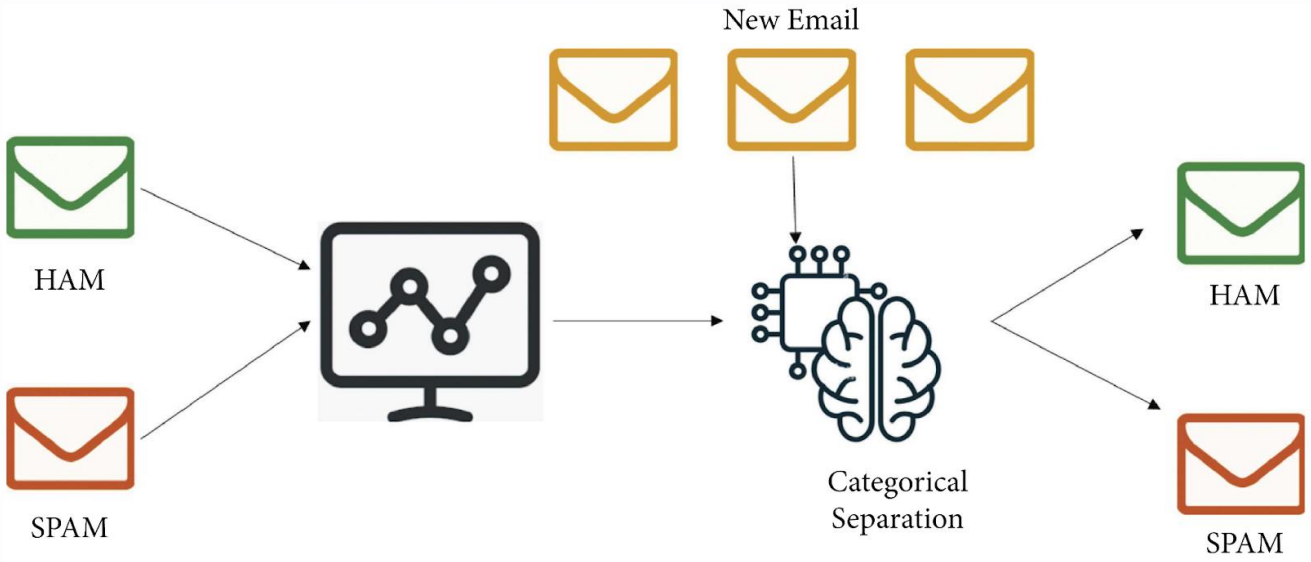
[Machine learning is the] field of study that gives computers the ability to learn without being explicitly programmed.

—Arthur Samuel, 1959

And a more engineering-oriented one:

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.

—Tom Mitchell, 1997



Traditional programming techniques:

1. You might notice that some words or phrases (such as “4U”, “credit card”, “free”, and “amazing”) tend to come up a lot in the subject line. Perhaps you would also notice a few other patterns in the sender’s name, the email’s body, and other parts of the email.

2. You would write a detection algorithm for each of the patterns that you noticed, and your program would flag emails as spam if a number of these patterns were detected.

3. You would test your program good enough to launch.

A long list of complex rules is necessary

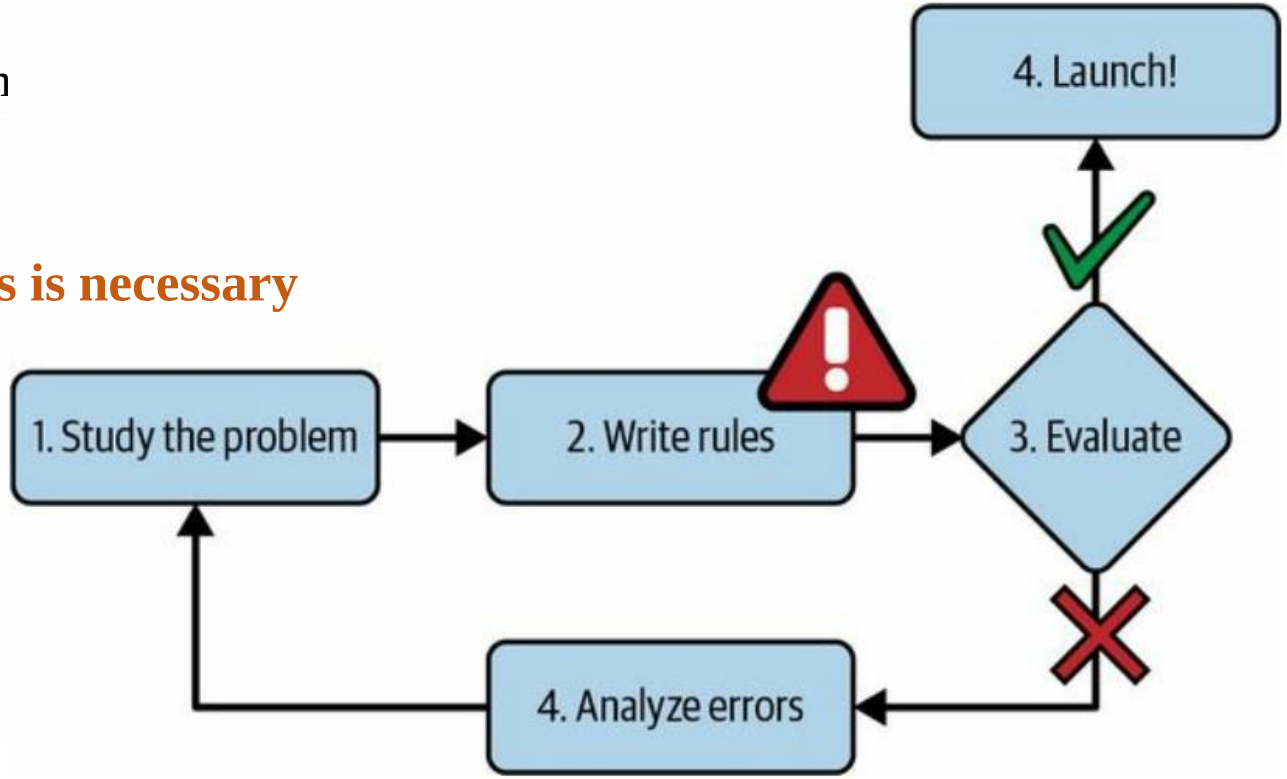


Figure 1-1. The traditional approach

In contrast, a spam filter based on machine learning techniques:

- automatically learns which words and phrases are good predictors of spam by detecting unusually frequent patterns of words in the spam examples compared to the ham examples.

- The program is much **shorter, easier to maintain, and most likely more accurate**.

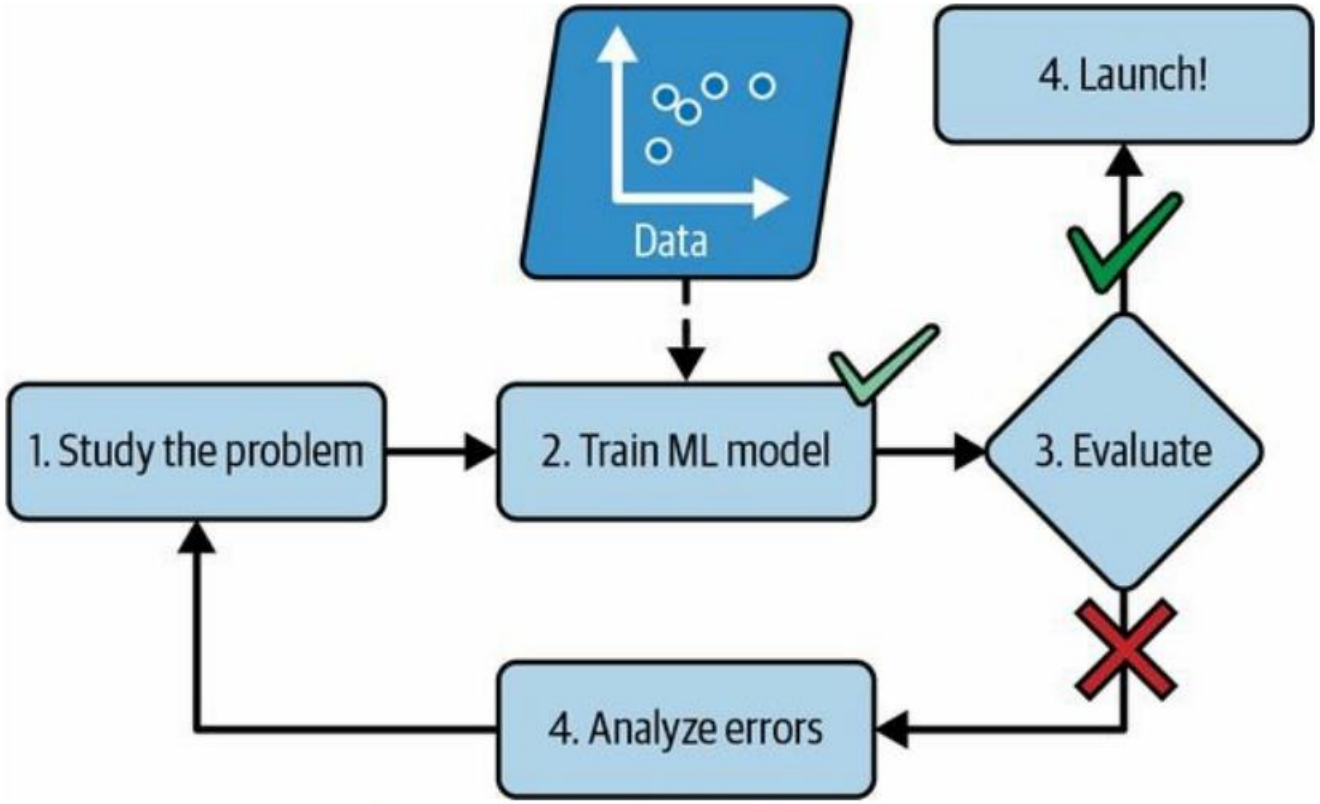


Figure 1-2. The machine learning approach

What if spammers notice that all their emails containing “4U” are blocked?
They might start writing “For U” instead. A spam filter using traditional programming techniques would need to be updated to flag “For U” emails.

In contrast, **a spam filter based on machine learning techniques automatically notices** that “For U” has become unusually frequent in spam flagged by users, and **it starts flagging them without your intervention**

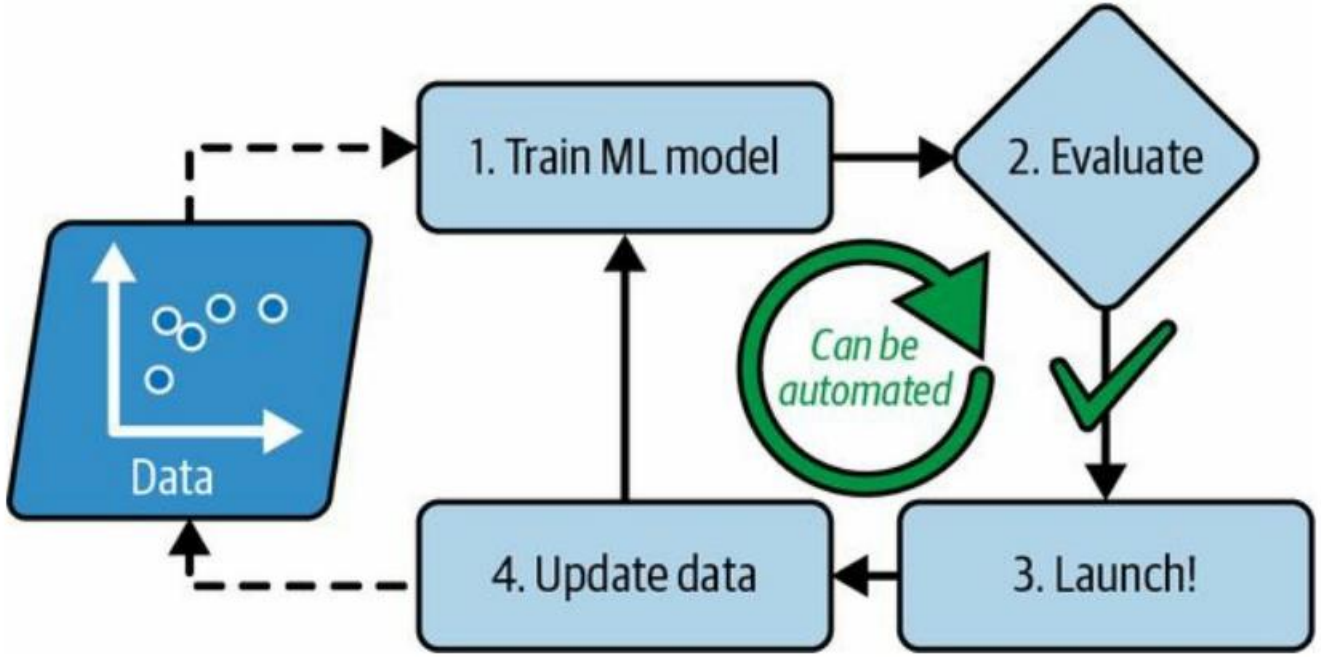


Figure 1-3. Automatically adapting to change

Machine learning can help humans learn

For instance, once a spam filter has been trained on enough spam, it can easily be inspected to reveal the list of words and combinations of words that it believes are the best predictors of spam. Sometimes this will reveal unsuspected correlations or new trends, and thereby lead to a better understanding of the problem.

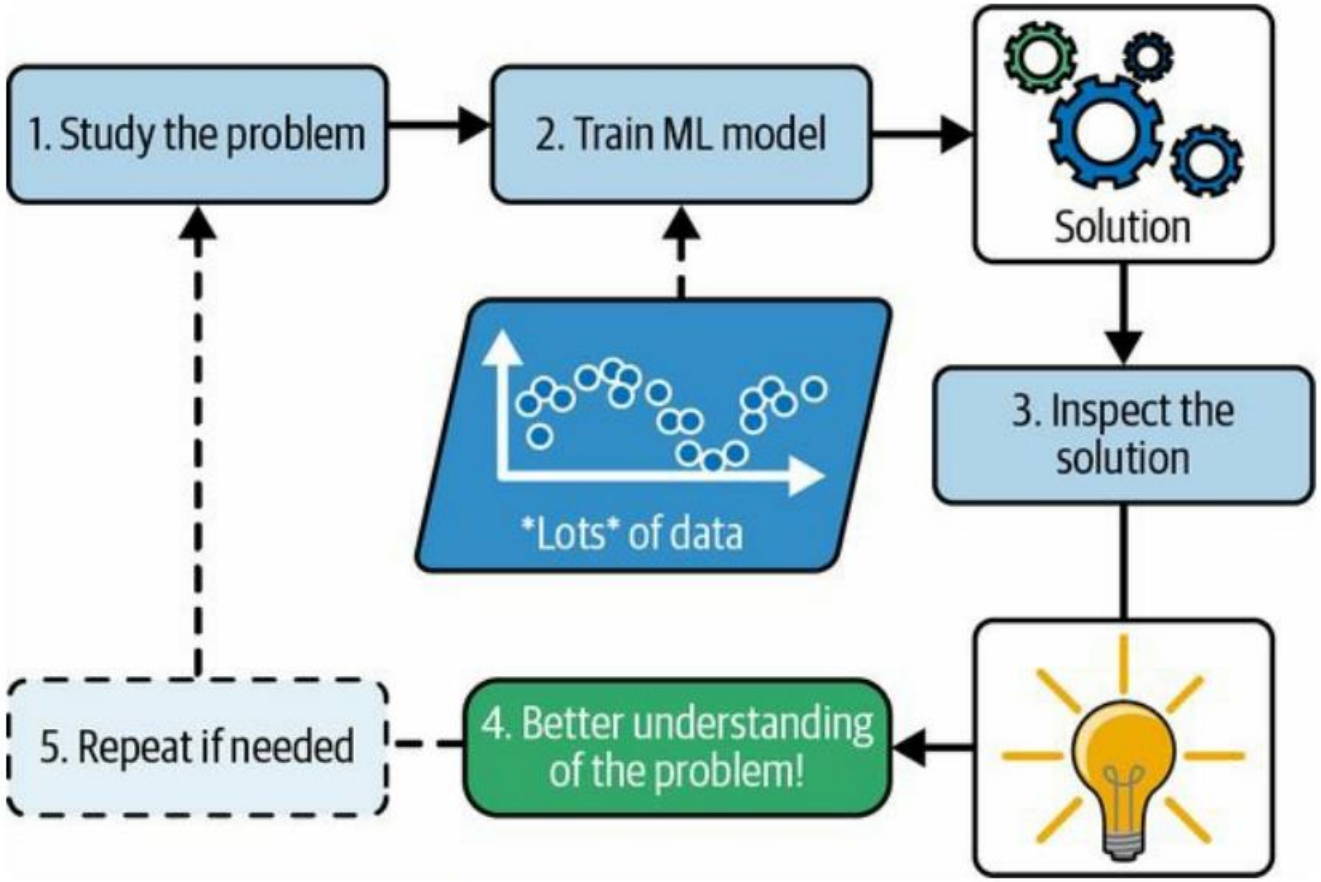


Figure 1-4. Machine learning can help humans learn

Types of Machine Learning Systems

- How they are supervised during training (supervised, unsupervised, semi-supervised, self-supervised, and others)
- Whether or not they can learn incrementally on the fly (online versus batch learning)
- Whether they work by simply comparing new data points to known data points, or instead by detecting patterns in the training data and building a predictive model, much like scientists do (instance-based versus model-based learning)

Types of Machine Learning Systems

- **How they are supervised during training (supervised, unsupervised, semi-supervised, self-supervised, and others)**

- Whether or not they can learn incrementally on the fly (online versus batch learning)

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

Supervised Learning

- In supervised learning, the training set you feed to the algorithm includes the **desired solutions, called labels**

A typical supervised learning task is classification

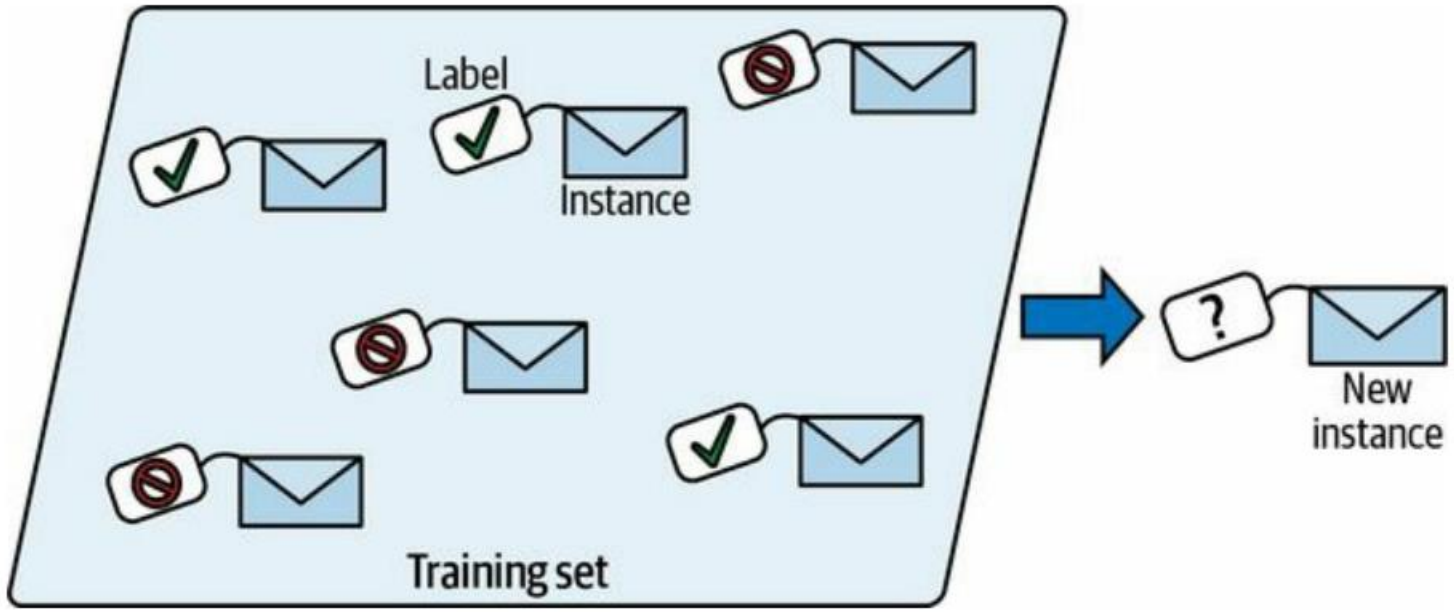
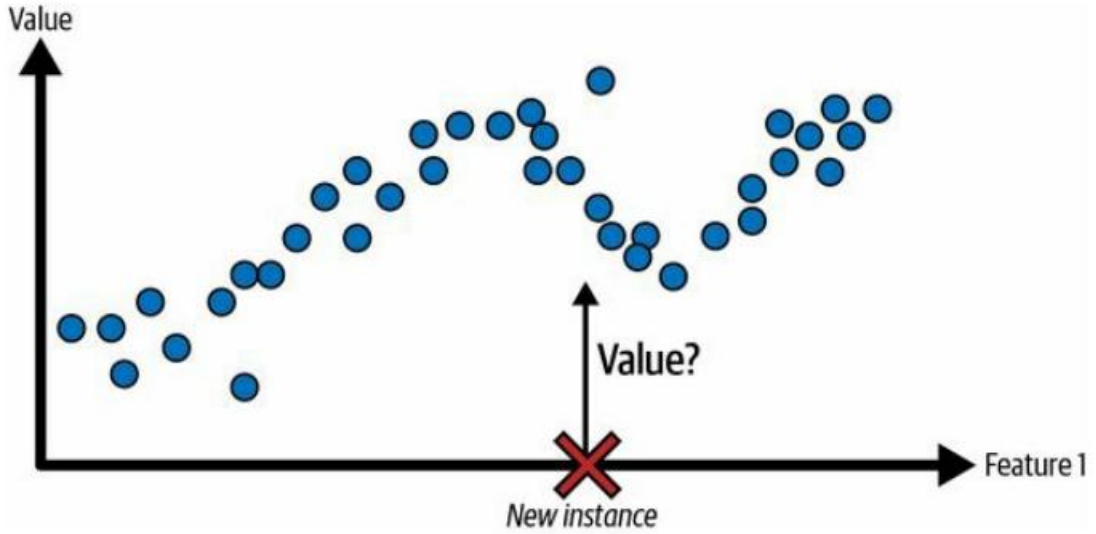


Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

Supervised Learning

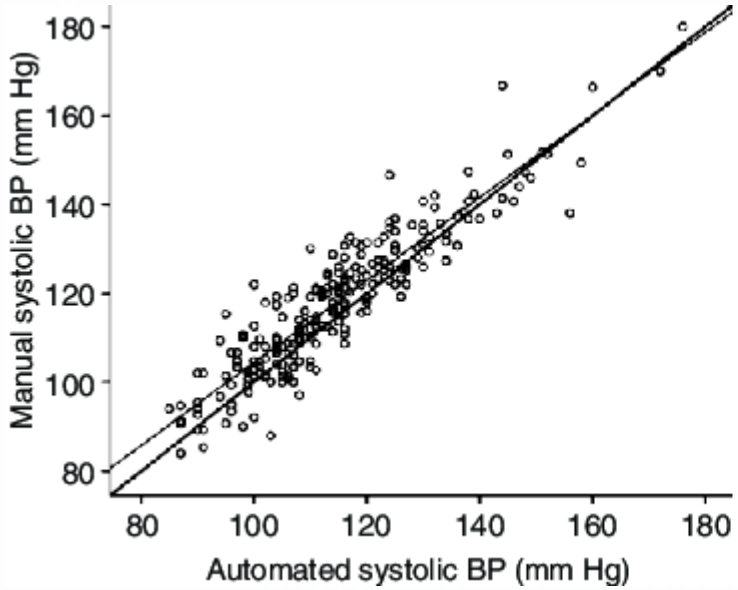
Another typical task is to predict a target numeric value, such as the price of a car, given a set of features (mileage, age, brand, etc.). **This sort of task is called regression**

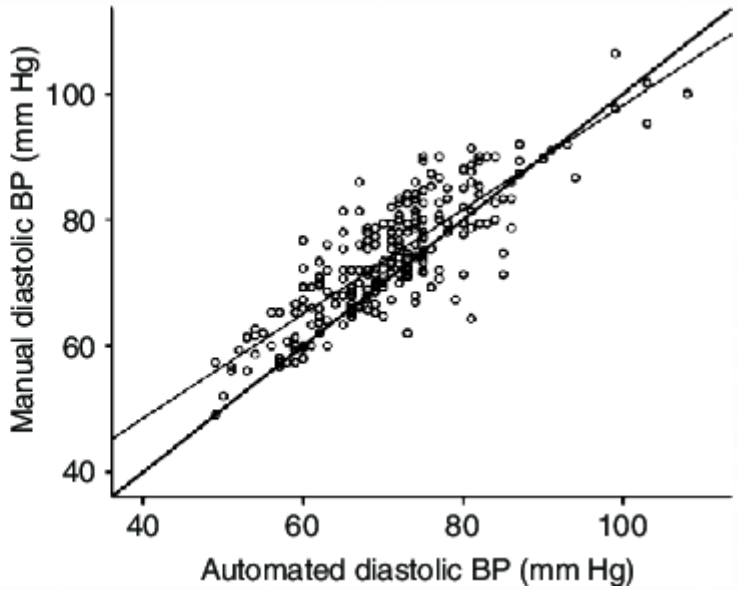


The words **target** and **label** are generally treated as synonyms in supervised learning, but target is more common in regression tasks and label is more common in classification tasks. Moreover, **features are sometimes called predictors or attributes**.

Supervised Learning

Regression: Blood Pressure Estimation





Unsupervised learning

The system tries to learn without a teacher.

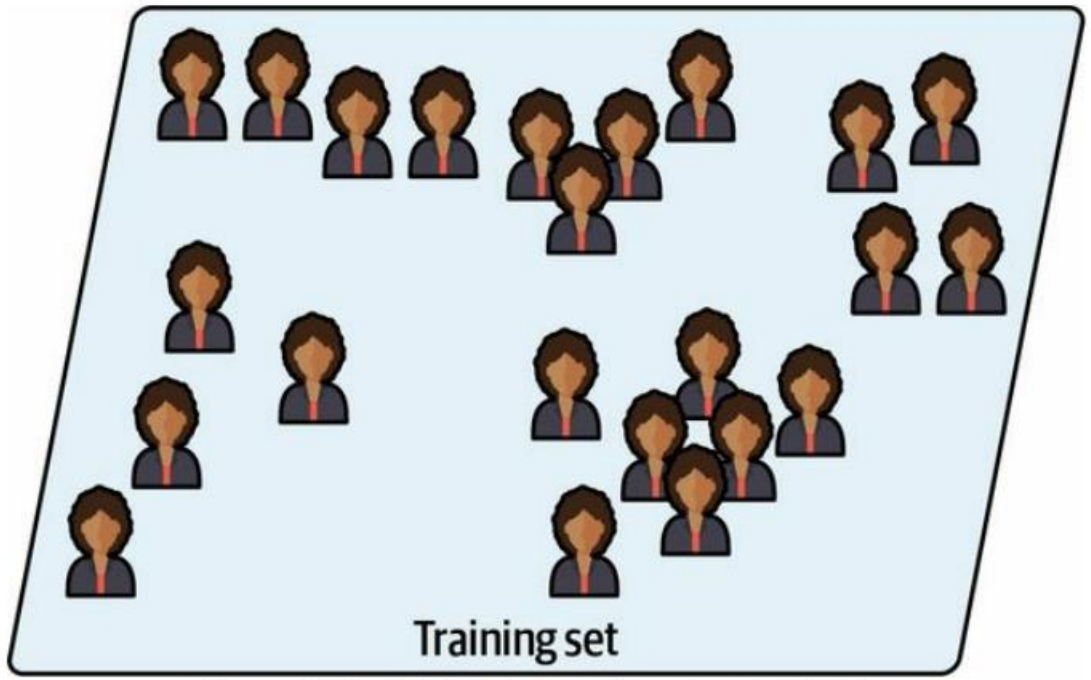


Figure 1-7. An unlabeled training set for unsupervised learning

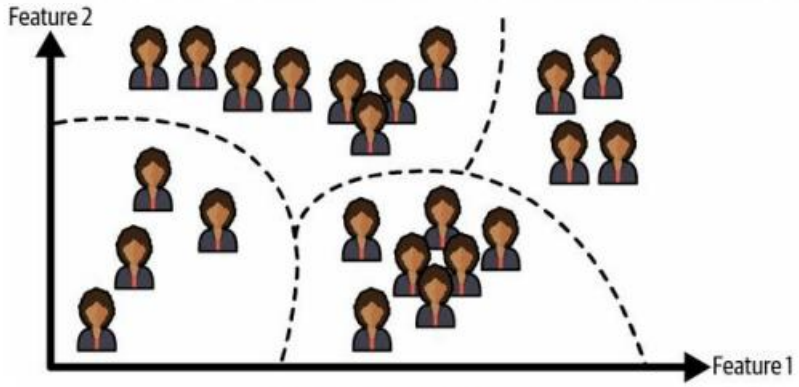
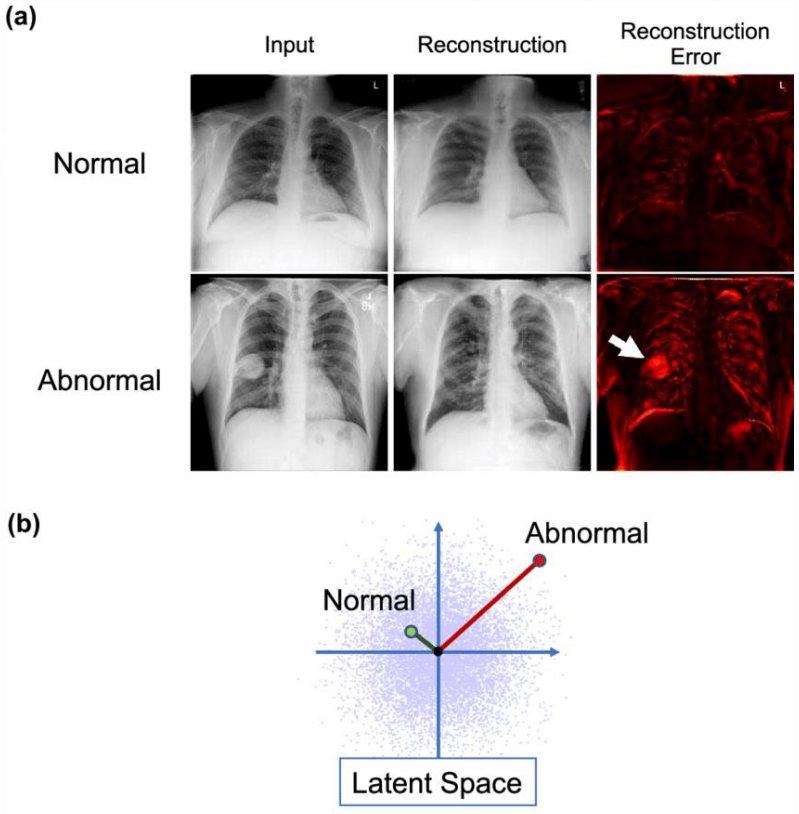


Figure 1-8. Clustering

Unsupervised learning

Another important unsupervised task is anomaly detection—for example, detecting unusual credit card transactions to prevent fraud, catching manufacturing defects, or automatically removing outliers from a dataset before feeding it to another learning algorithm.



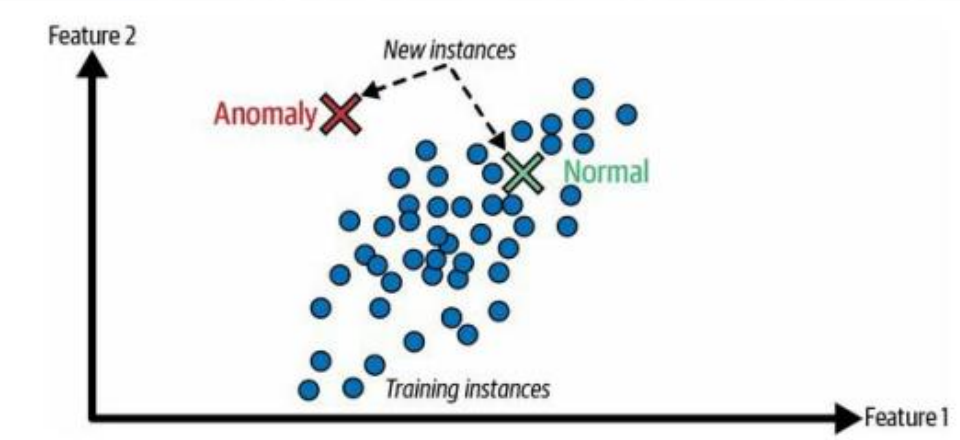


Figure 1-10. Anomaly detection

Unsupervised learning

Association Rule Mining

Association Rule Mining in Medical Image Analysis: A Hypothetical Example

- **Scenario:** A hospital is interested in identifying patterns of co-occurring diseases based on medical images.
- **Data:** The hospital has a large dataset of medical images (e.g., X-rays, MRIs, CT scans) paired with patient medical records. Each image is associated with a set of attributes, such as patient age, gender, symptoms, and diagnoses.
- **Association Rule:** The hospital applies an association rule mining algorithm to this dataset.

Potential Discovery: The algorithm might discover the following rule:

- ✓ **Rule:** Patients with X-ray evidence of pneumonia and a history of smoking are more likely to also have lung cancer.
- ✓ **Support:** 20% of patients with pneumonia and a smoking history also have lung cancer.
- ✓ **Confidence:** 80% of patients with pneumonia and a smoking history who have lung cancer also have X-ray evidence of pneumonia.

Semi-supervised learning

Since labeling data is usually time-consuming and costly, **you will often have plenty of unlabeled instances**, and **few labeled instances**.
Some algorithms can deal with data that's **partially labeled**. This is called **semi-supervised learning**

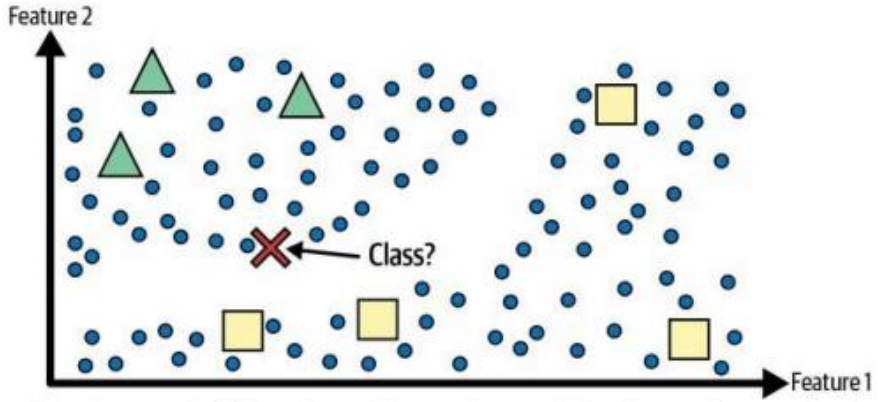


Figure 1-11. Semi-supervised learning with two classes (triangles and squares): the unlabeled examples (circles) help classify a new instance (the cross) into the triangle class rather than the square class, even though it is closer to the labeled squares

Most semi-supervised learning algorithms are **combinations of unsupervised and supervised algorithms**. For example, **a clustering algorithm may be used to group similar instances together, and then every unlabeled instance can be labeled with the most common label in its cluster**. Once the whole dataset is labeled, it is possible to use any supervised learning algorithm.

Semi-Supervised Learning in Medical Image Classification: A Hypothetical Example

Scenario: A medical research institution has a large dataset of brain MRI scans. However, only a small portion of these images are labeled with diagnoses (e.g., healthy, tumor, stroke).

Approach: The institution uses a semi-supervised learning algorithm to leverage the unlabeled data to improve the model's performance.

Method:

1.Initial Training: The algorithm is first trained on the labeled subset of images.

2.Prediction: The trained model is used to predict the labels of the unlabeled images.

3.Confidence Thresholding: Only the predictions with high confidence are added to the training set.

4.Retraining: The model is retrained with the expanded dataset, including the newly labeled images.

Iterative Process: This process can be repeated multiple times, gradually incorporating more unlabeled data into the training set.

Self-supervised learning

Actually, generating a fully labeled dataset from a **fully unlabeled** one. Again, once the whole dataset is labeled, any supervised learning algorithm can be used.

Masked Prediction

1. **Pre-training:** The model is initially trained on a large dataset of unlabeled images to predict masked parts.
2. **Additional Layers:** A new classification layer (like SVM or FC) is added to the pre-trained model.
3. **Fine-tuning:** The entire model is then trained on a smaller labeled dataset to optimize its performance for the classification task.

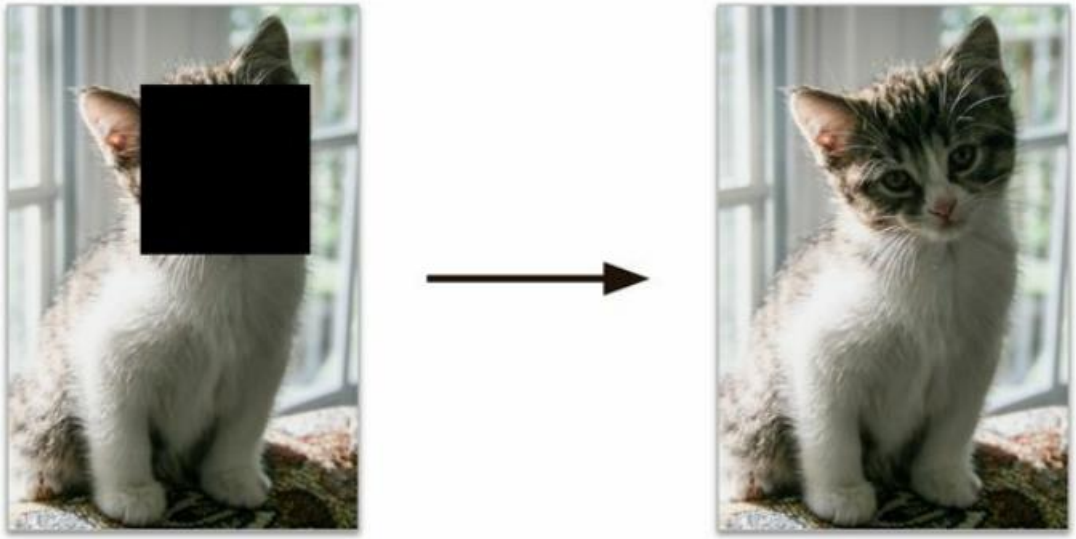
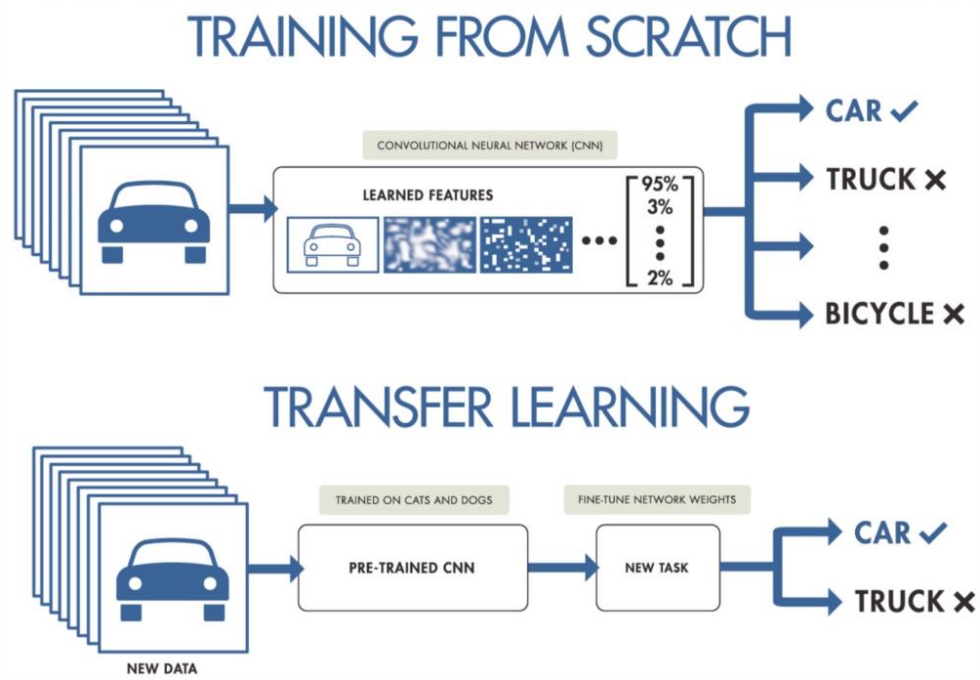


Figure 1-12. Self-supervised learning example: input (left) and target (right)

Self-supervised vs Semi-supervised??

Transferring knowledge from one task to another is called **transfer learning**, and it's one of the most important techniques in machine learning today, especially when using deep neural networks (i.e., neural networks composed of many layers of neurons).



Mask prediction has employed a transfer learning approach

Reinforcement learning

The learning system, called an agent in this context, can **observe the environment**, **select and perform actions**, and **get rewards** in return. A **policy** defines what action the agent should choose when it is in a given situation.

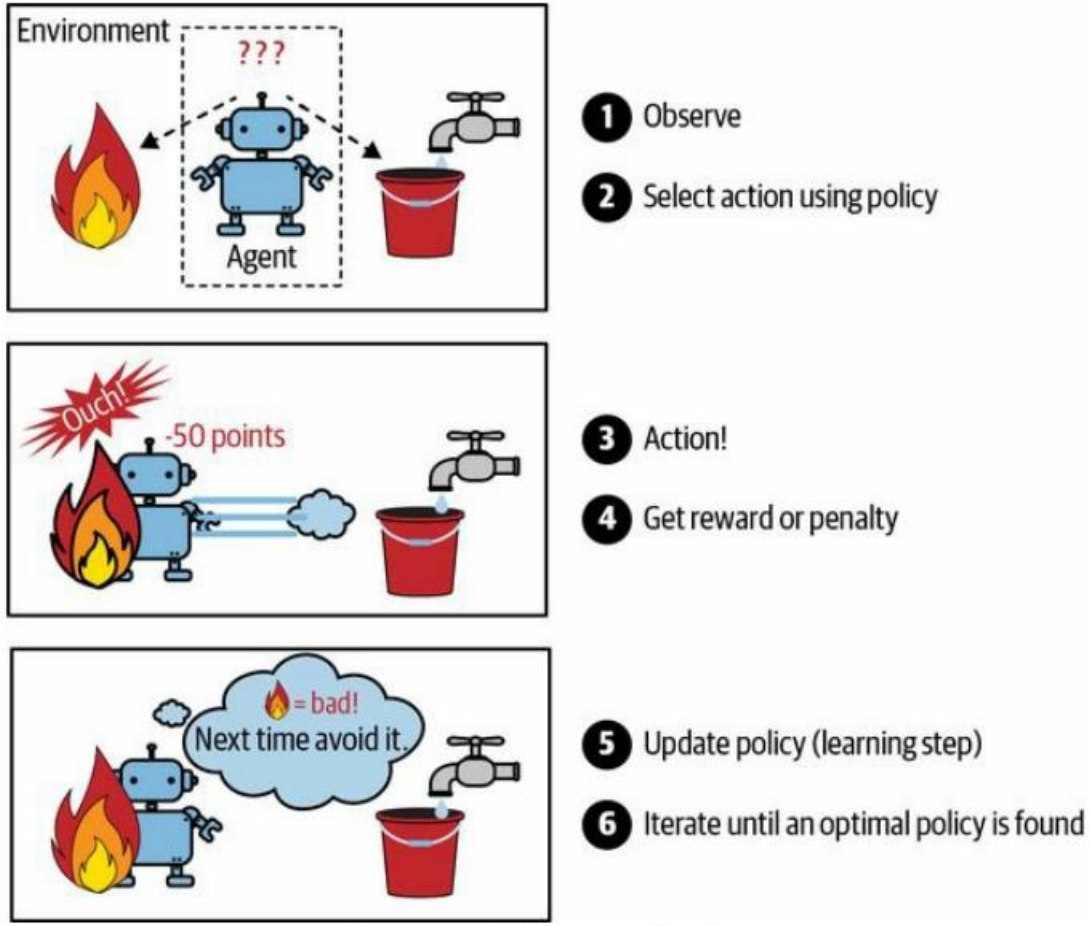
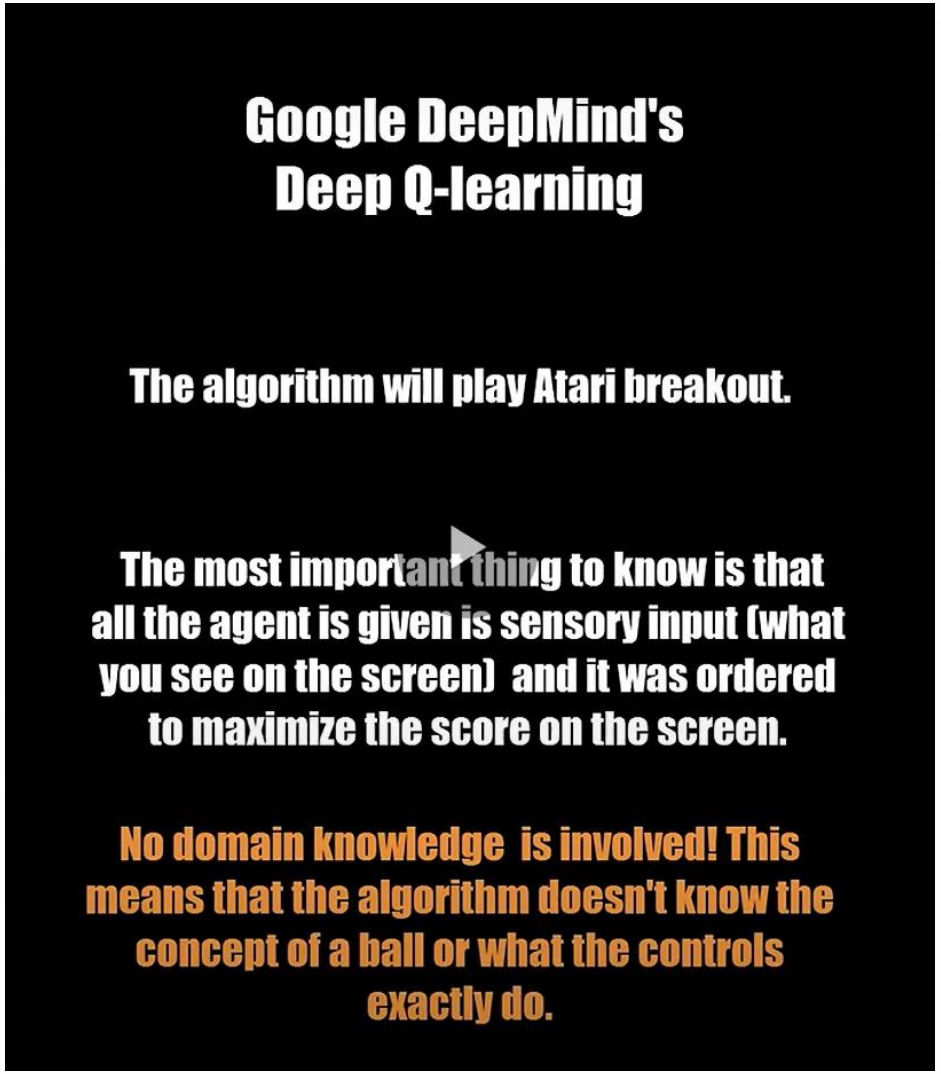


Figure 1-13. Reinforcement learning


Google DeepMind's
Deep Q-learning
The algorithm will play Atari breakout.
The most important thing to know is that all the agent is given is sensory input (what you see on the screen) and it was ordered to maximize the score on the screen.
No domain knowledge is involved! This means that the algorithm doesn't know the concept of a ball or what the controls exactly do.

Types of Machine Learning Systems

- How they are supervised during training (supervised, unsupervised, semi-supervised, self-supervised, and others)
- **Whether or not they can learn incrementally on the fly (online versus batch learning)**
- Whether they work by simply comparing new data points to known data points, or instead by detecting patterns in the training data and building a predictive model, much like scientists do (instance-based versus model-based learning)

Batch Versus Online Learning

- In ***batch learning***, the system is incapable of learning incrementally: it must be trained using all the available data.
- Batch learning is often referred to as ***offline learning*** because the model is trained on a fixed dataset before being deployed into a production environment. This means that the model doesn't learn from new data as it encounters it in real-time.
- If the model deals with **fast-evolving systems**, for example making predictions on the financial market, then it is likely to decay quite fast.
- Even a model trained to classify pictures of cats and dogs may need to be retrained regularly, not because cats and dogs will mutate overnight, but **because cameras keep changing**, along with image formats, sharpness, brightness, and size ratios.
- This solution is simple and often works fine, but training using the full set of data can take many hours, so you would typically train a new system only every 24 hours or even just weekly. **If your system needs to adapt to rapidly changing data (e.g., to predict stock prices), then you need a more reactive solution**

Batch Versus Online Learning

- In ***online learning***, you train the system incrementally by feeding it data instances sequentially, either individually or in small groups called ***mini batches***.
- Each learning step is fast and cheap, so the system can learn about new data on the fly, as it arrives

One important parameter of online learning systems is *how fast they should adapt to changing data*: this is called the *learning rate*.

If you set a high learning rate, then your system will rapidly adapt to new data, but it will also tend to quickly forget the old data.

Conversely, **if you set a low learning rate, the system will have more inertia; that is, it will learn more slowly**, but it will also be less sensitive to noise in the new data or to sequences of nonrepresentative data points (outliers).

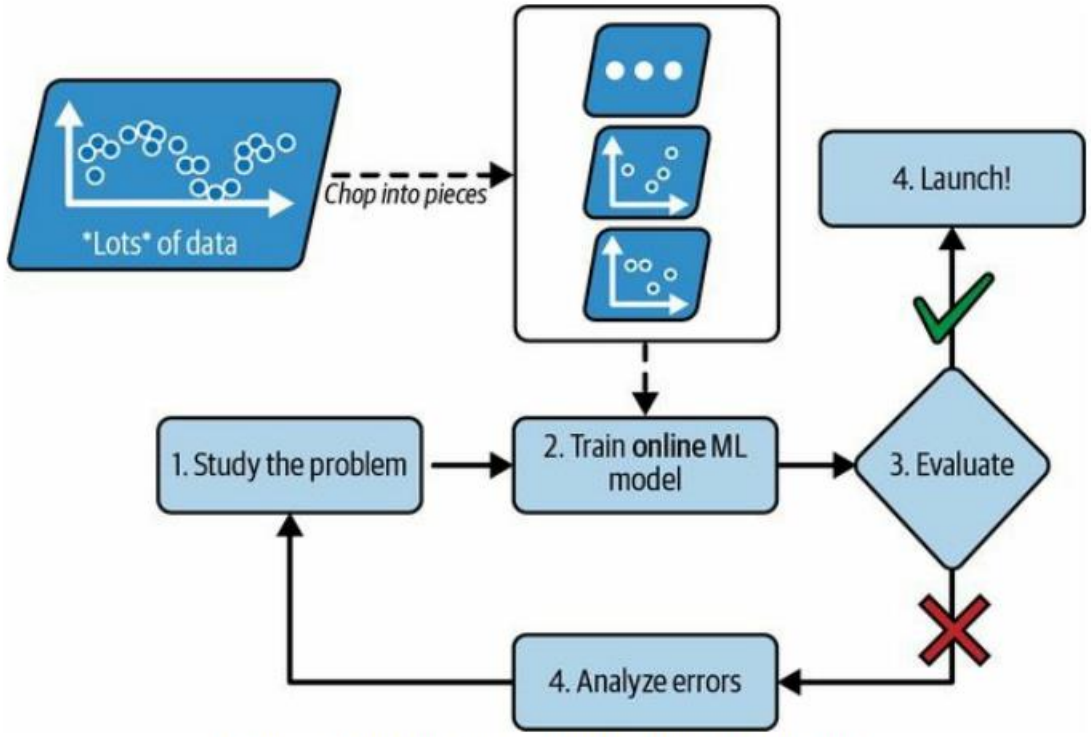


Figure 1-15. Using online learning to handle huge datasets

Types of Machine Learning Systems

- How they are supervised during training (supervised, unsupervised, semi-supervised, self-supervised, and others)
- Whether or not they can learn incrementally on the fly (online versus batch learning)
- **Whether they work by simply comparing new data points to known data points, or instead by detecting patterns in the training data and building a predictive model, much like scientists do (instance-based versus model-based learning)**

Instance-Based Versus Model-Based Learning

How ML approaches *generalize*

Instance-based learning: the system learns the examples by heart, then generalizes to new cases by using a similarity measure to compare them to the learned examples (or a subset of them). For example, in Figure 1-16 the new instance would be classified as a triangle because **the majority of the most similar instances belong to that class**.

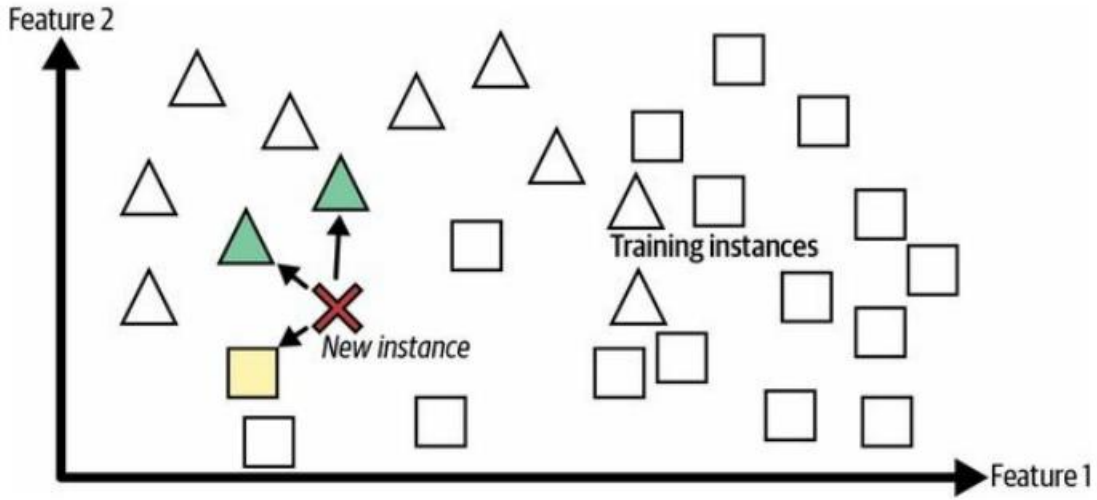


Figure 1-16. Instance-based learning

Instance-Based Versus Model-Based Learning

Another way to generalize from a set of examples is to build a model of these examples and then use that model to make predictions. This is called ***model-based learning***.

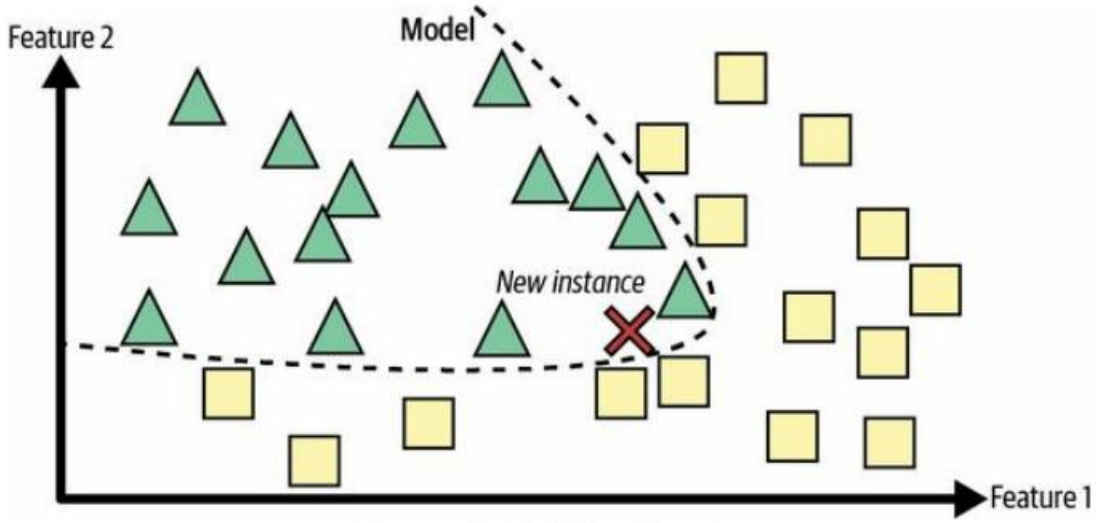


Figure 1-17. Model-based learning

Example: Model-Based Learning

Table 1-1. Does money make people happier?

Country	GDP per capita (USD)	Life satisfaction
Turkey	28,384	5.5
Hungary	31,008	5.6
France	42,026	6.5
United States	60,236	6.9
New Zealand	42,404	7.3
Australia	48,698	7.3
Denmark	55,938	7.6

Linear Regression

Equation 1-1. A simple linear model

life_satisfaction = θ 0 + θ 1 × GDP_per_capita

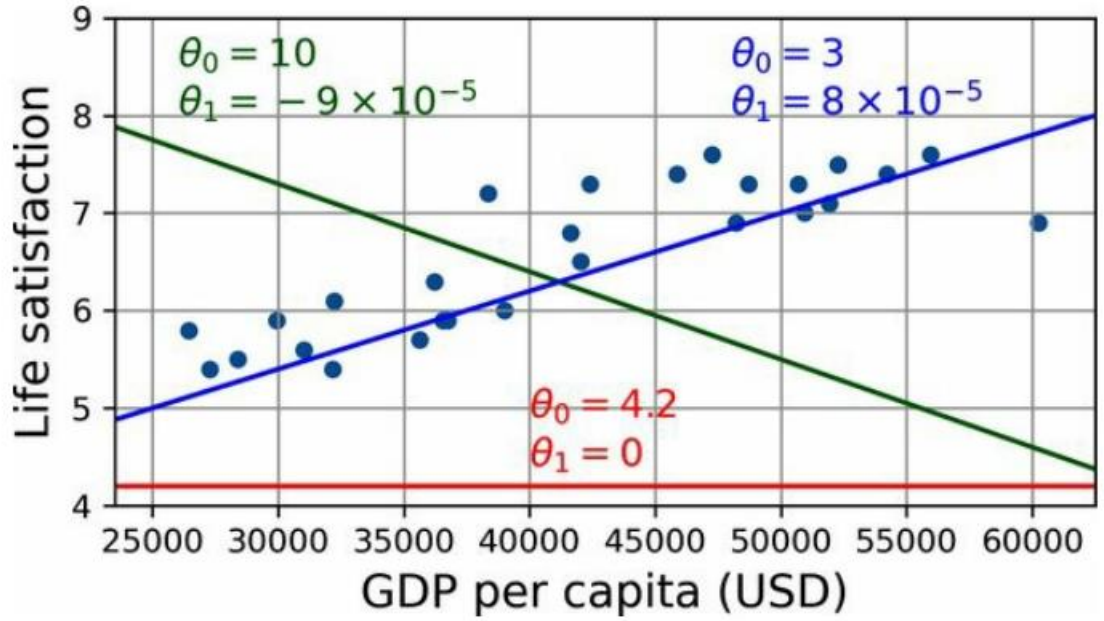


Figure 1-19. A few possible linear models

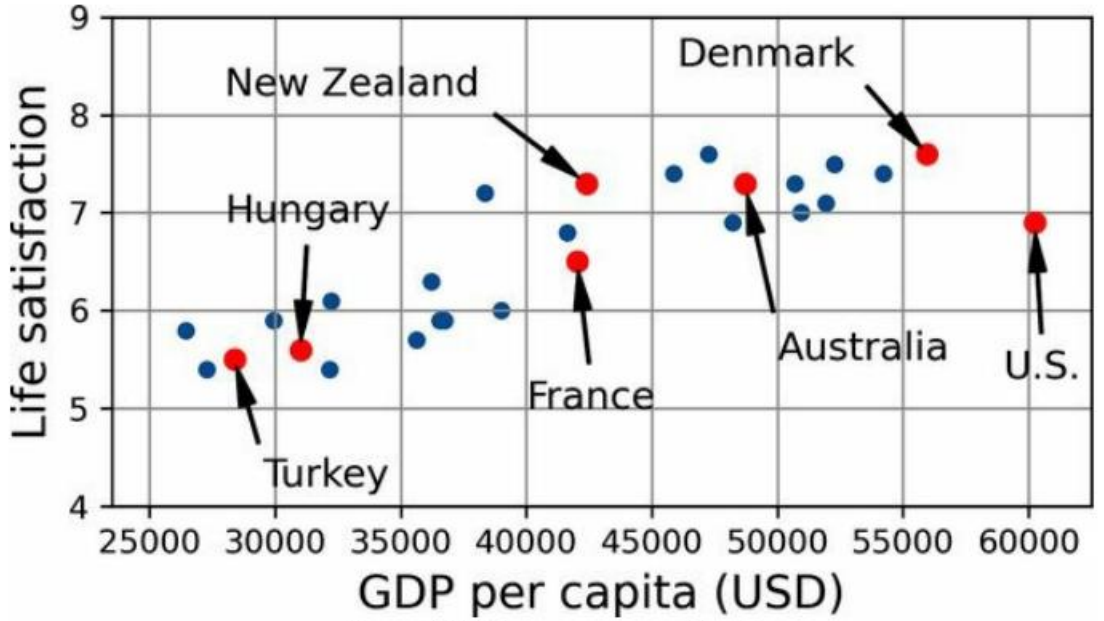


Figure 1-18. Do you see a trend here?

Before you can use your model, **you need to define the parameter values θ_0 and θ_1** . How can you know which values will make your model perform best?
To answer this question, you need to specify a **performance measure**.
You can either define a utility function (or **fitness function**) that measures how good your model is, or you can define a cost function that measures how bad it is

You look up Cyprus's GDP per capita, find $37,655, and then apply your model and find that life satisfaction is likely to be somewhere around 3.75 + 37,655 × 6.78 × e(-5) = **6.30**.

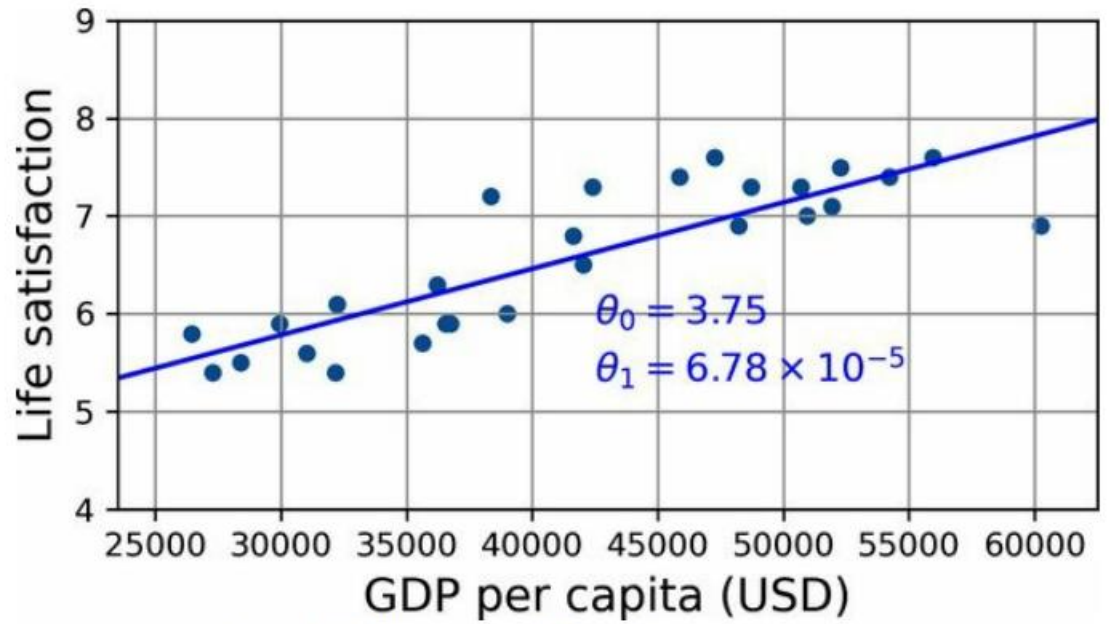


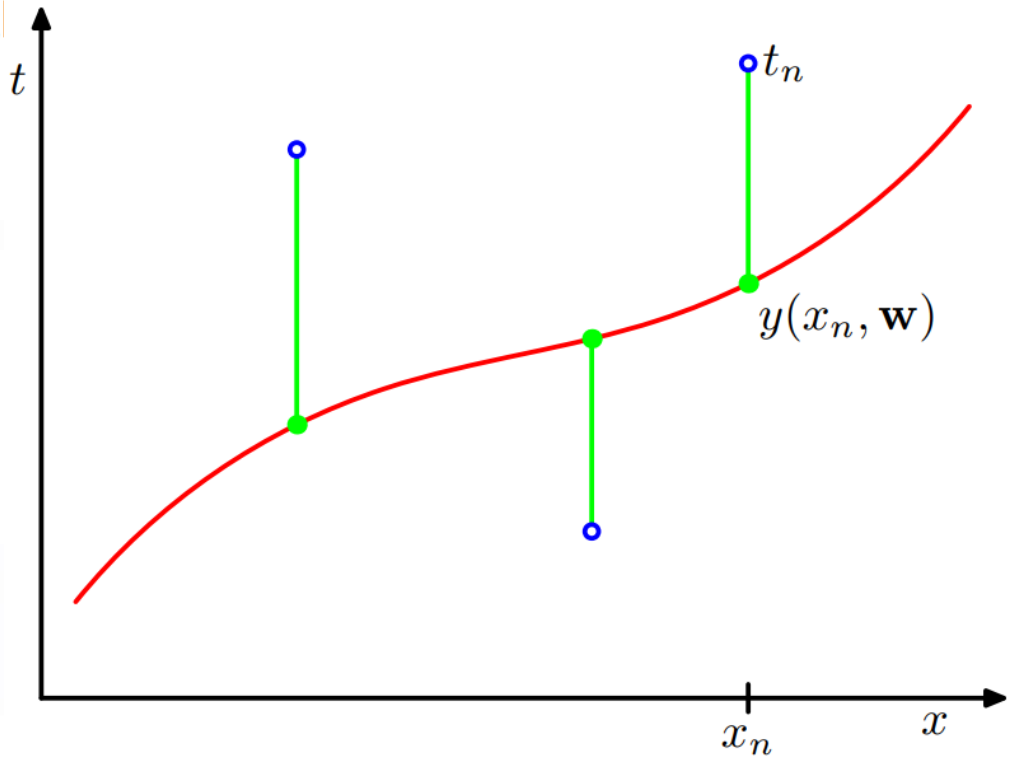
Figure 1-20. The linear model that fits the training data best

Polynomial Curve Fitting (theory)

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

M is order of polynomial

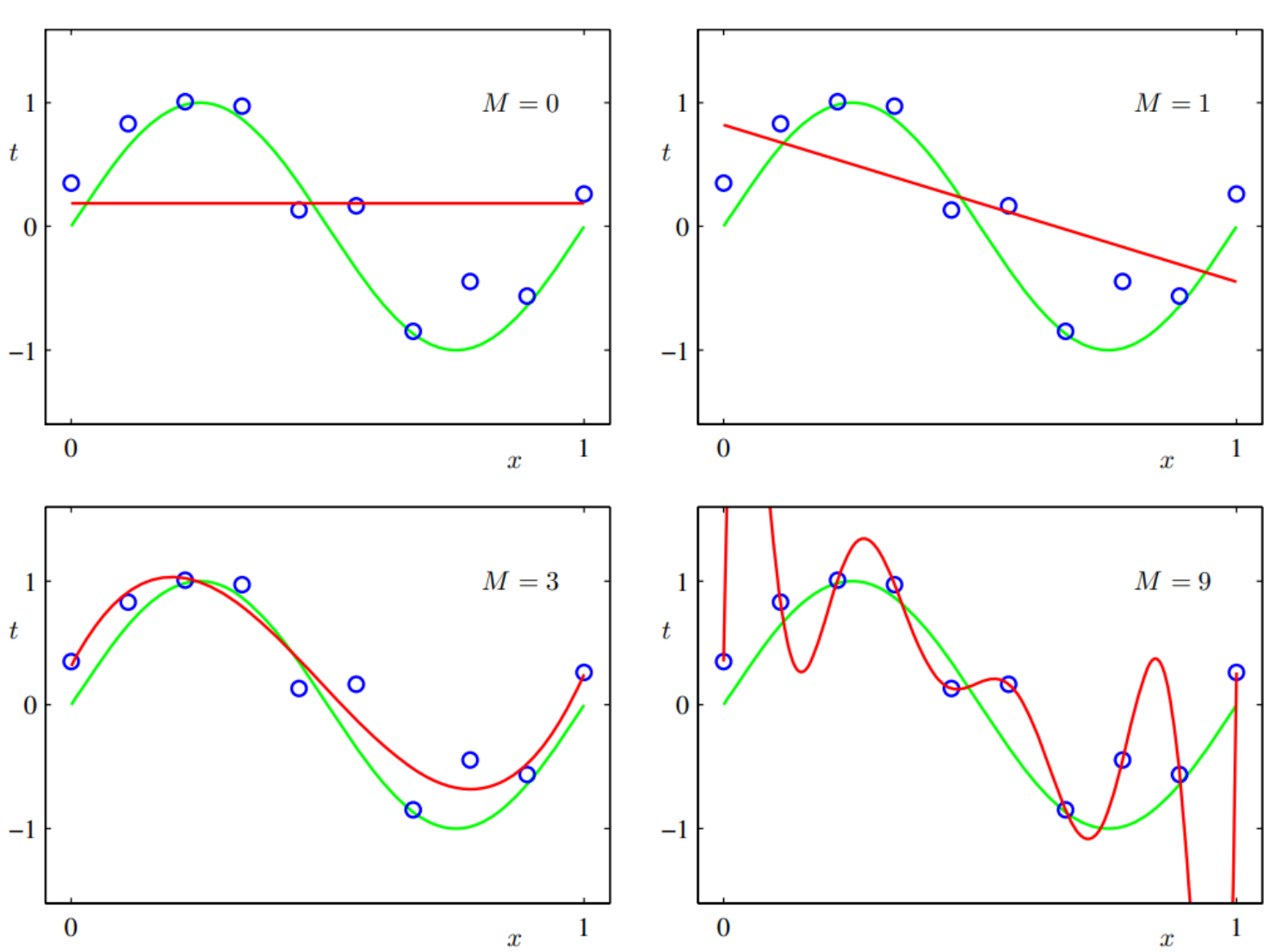
- Training set: $\mathbf{x} \equiv (x_1, \ldots, x_N)$ AND $\mathbf{t} \equiv (t_1, \ldots, t_N)$
- Goal: predict the target $\hat{t}$ for some new input $\hat{x}$
- *Probability theory* allows to express the uncertainty of t target.
- *Decision theory* allows to make optimal predictions.



- Minimize:

$$E(\boldsymbol{w}) = \frac{1}{2} \sum_{i=1}^{N} \{y(x_n, \boldsymbol{w}) - t_n\}^2$$

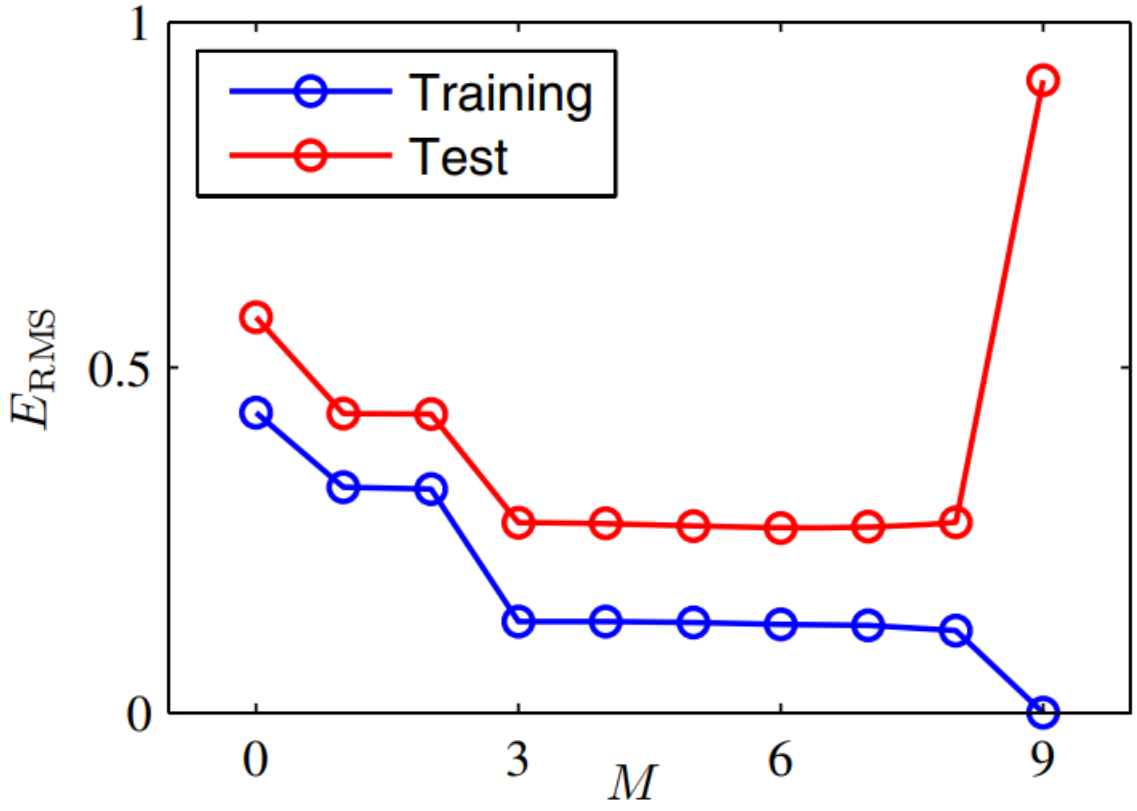
Polynomial Curve Fitting (theory)



(RMS) error defined by

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$

Graphs of the root-mean-square error, defined by (1.3), evaluated on the training set and on an independent test set for various values of M.



In $M = 9$ overfitting is occurred.

Observe how the typical **magnitude of the coefficients increases** dramatically as the **order of the polynomial (*M*)** increases.

	$M = 0$	$M = 1$	$M = 6$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Observe how the typical **magnitude of the coefficients increases** dramatically as the **order of the polynomial (*M*)** increases.

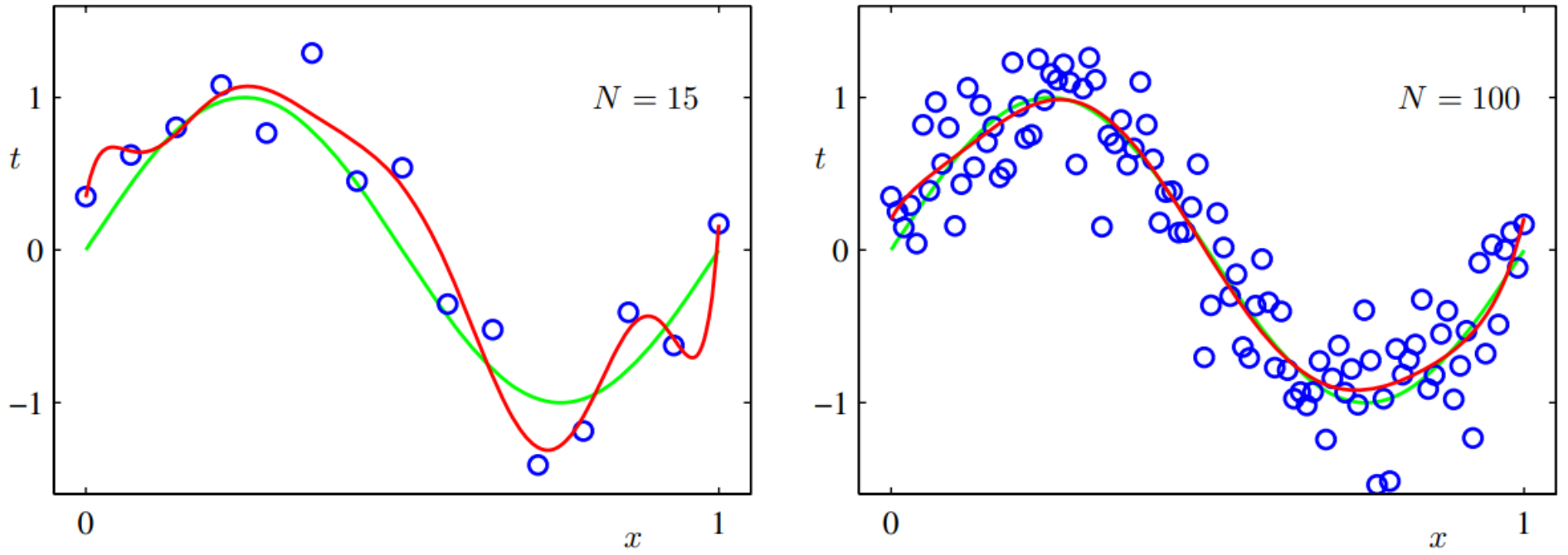


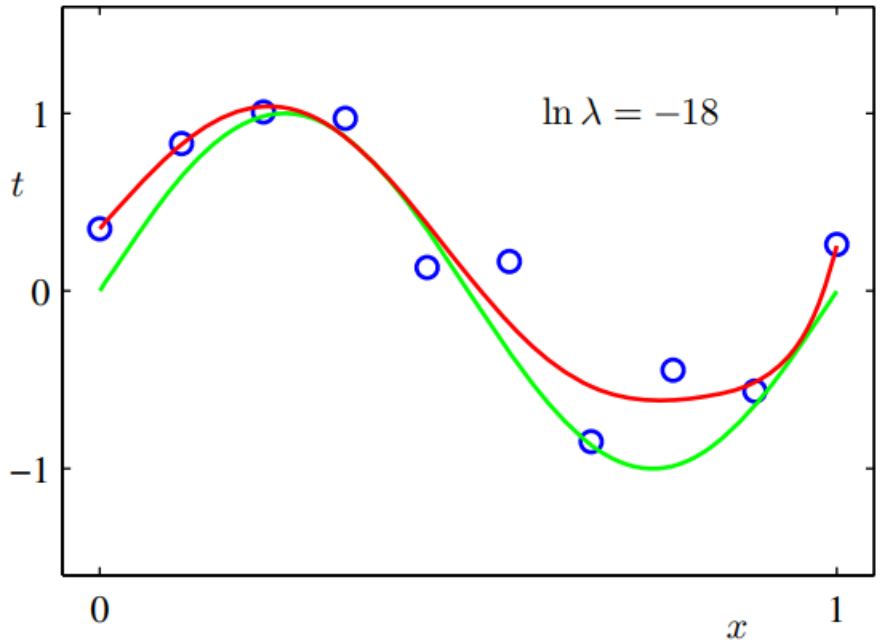
Figure 1.6 Plots of the solutions obtained by minimizing the sum-of-squares error function using the $M = 9$ polynomial for $N = 15$ data points (left plot) and $N = 100$ data points (right plot). We see that increasing the size of the data set reduces the over-fitting problem.

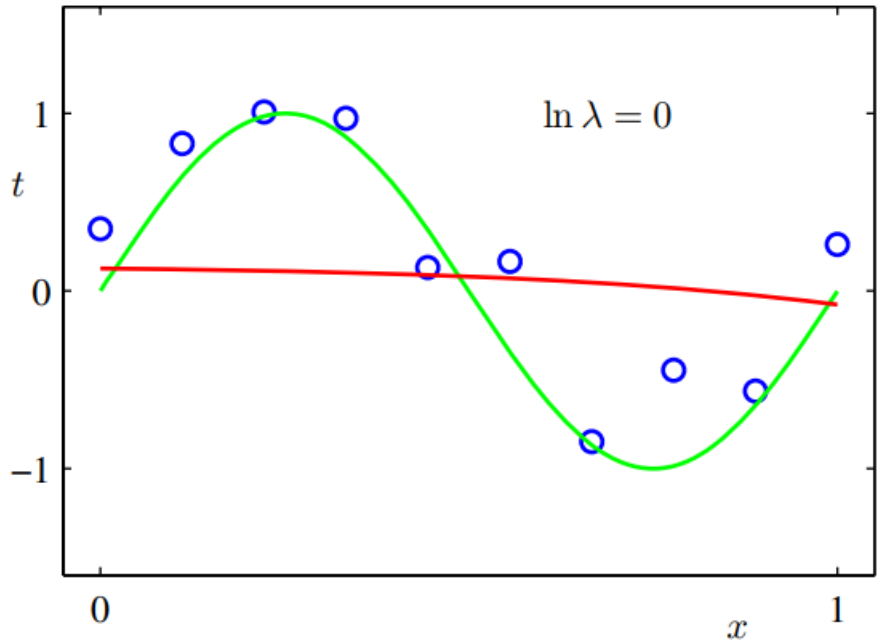
Regularization term to avoid overfitting

One technique that is often used to control the over-fitting phenomenon in such cases is that of ***regularization***, which involves adding a penalty term to the error function in order to **discourage the coefficients from reaching large values.**

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

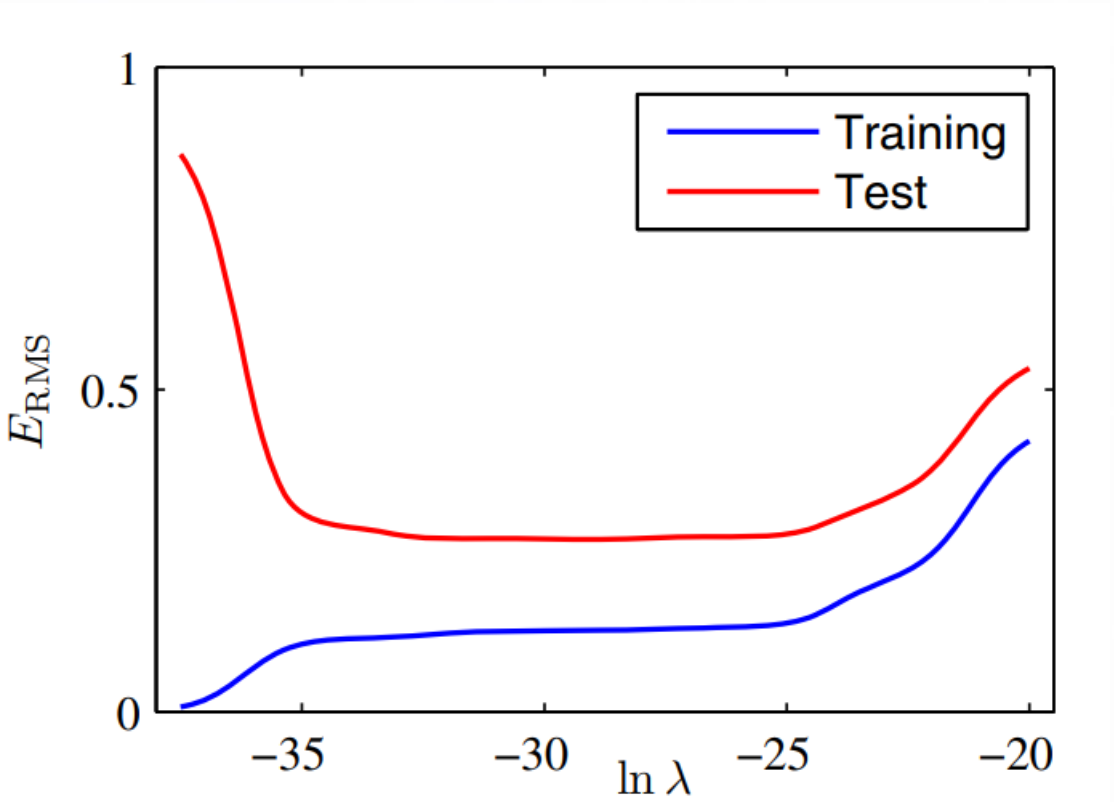
$$\|\mathbf{w}\|^2 \equiv \mathbf{w}^{\mathrm{T}}\mathbf{w} = w_0^2 + w_1^2 + \ldots + w_M^2$$





Regularization term to avoid overfitting

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01



Linear Regression as a model-based ML

Example 1-1. Training and running a linear model using Scikit-Learn

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression

# Download and prepare the data
data_root = "https://github.com/ageron/data/raw/main/"
lifesat = pd.read_csv(data_root + "lifesat/lifesat.csv")
X = lifesat[["GDP per capita (USD)"]].values
y = lifesat[["Life satisfaction"]].values

# Visualize the data
lifesat.plot(kind='scatter', grid=True,
             x="GDP per capita (USD)", y="Life satisfaction")
plt.axis([23_500, 62_500, 4, 9])
plt.show()

# Select a linear model
model = LinearRegression()

# Train the model
model.fit(X, y)

# Make a prediction for Cyprus
X_new = [[37_655.2]]  # Cyprus' GDP per capita in 2020
print(model.predict(X_new)) # output: [[6.30165767]]
```

K-neareset neighbor as an Instance-based ML

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

with these two:

```
from sklearn.neighbors import KNeighborsRegressor
model = KNeighborsRegressor(n_neighbors=3)
```

Main Challenges of Machine Learning

Insufficient Quantity of Training Data

Even for very simple problems you typically need thousands of examples, and for complex problems such as image or speech recognition **you may need millions of examples**.

In a famous paper published in 2001, Microsoft researchers Michele Banko and Eric Brill showed that very different machine learning algorithms, including fairly simple ones, performed almost identically well on a complex problem of natural language disambiguation once they were given enough data.

The idea that data matters more than algorithms for complex problems was further popularized by Peter Norvig et al. in a paper titled "The Unreasonable Effectiveness of Data", published in 2009. **It should be noted, however, that small and medium-sized datasets are still very common, and it is not always easy or cheap to get extra training data — so don't abandon algorithms just yet**.

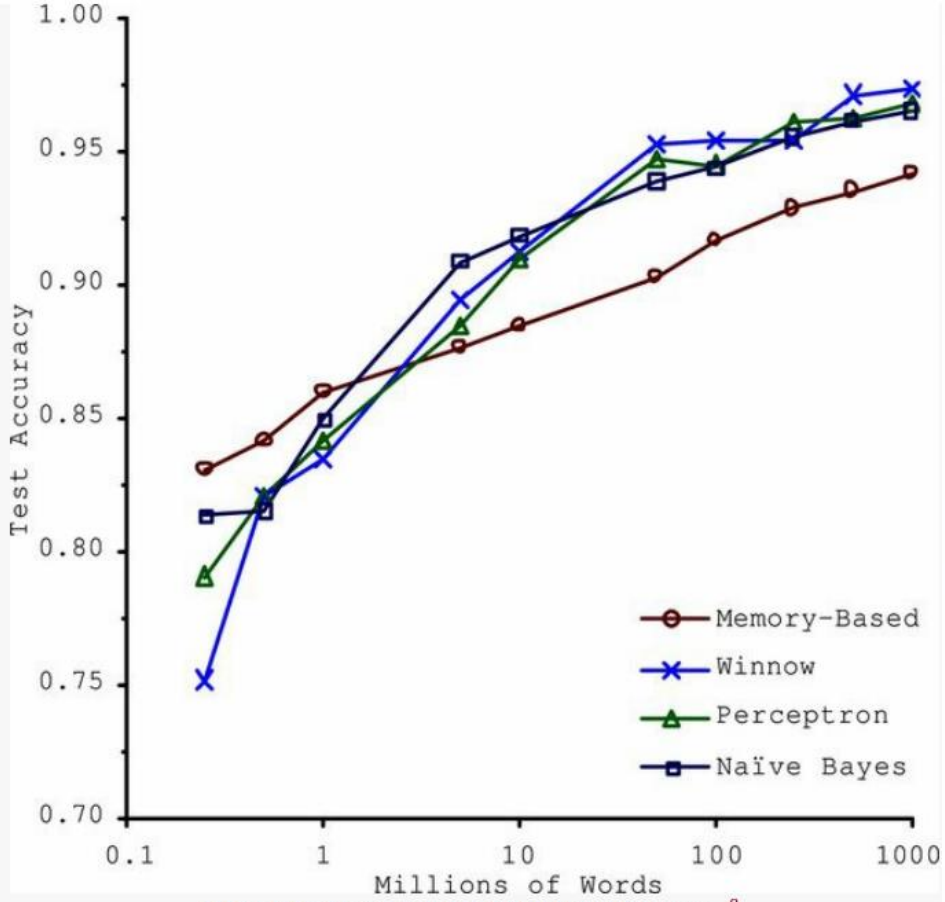


Figure 1-21. The importance of data versus algorithms [8]

Nonrepresentative Training Data

In order to generalize well, it is crucial that your **training data be representative of the new cases you want to generalize to**. This is true whether you use instance-based learning or model-based learning.

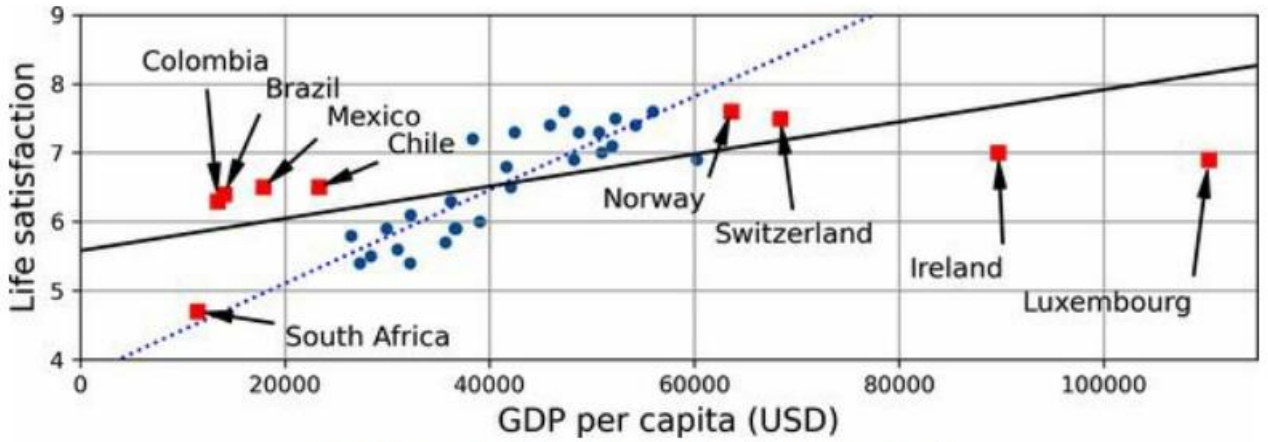


Figure 1-22. A more representative training sample

Poor-Quality Data

Obviously, if your training data is full of errors, outliers, and noise (e.g., due to poor-quality measurements), it will make it harder for the system to detect the underlying patterns, so your system is less likely to perform well.
It is often well worth the effort **to spend time cleaning up your training data**. The truth is, most data scientists spend a significant part of their time doing just that. The following are a couple examples of when you'd want to clean up training data:

- If some instances are clearly **outliers**, it may help to simply discard them or try to fix the errors manually.

- If some instances are missing a few features (e.g., 5% of your customers did not specify their age), you must decide whether you want to ignore this attribute altogether, ignore these instances, fill in the missing values (e.g., with the median age), or train one model with the feature and one model without it

Overfitting the Training Data

overfitting: it means that the model performs well on the training data, but it does not generalize well.

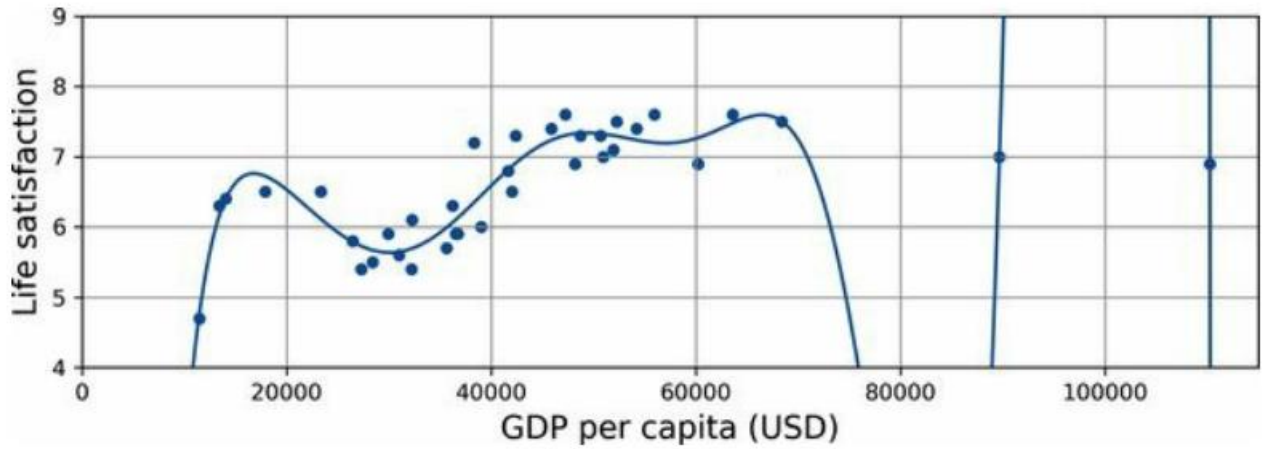


Figure 1-23. Overfitting the training data

For example, say you feed your life satisfaction model many more attributes, including uninformative ones such as **the country's name**. In that case, a complex model may detect patterns like the fact that all countries in the training data with a w in their name have a life satisfaction greater than 7: New Zealand (7.3), Norway (7.6), Sweden (7.3), and Switzerland (7.5). How confident are you that the w-satisfaction rule generalizes to Rwanda or Zimbabwe?

WARNING

Overfitting happens when the model is too complex relative to the amount and noisiness of the training data. Here are possible solutions:

- Simplify the model by selecting one with fewer parameters (e.g., a linear model rather than a high-degree polynomial model), by reducing the number of attributes in the training data, or by constraining the model.
- Gather more training data.
- Reduce the noise in the training data (e.g., fix data errors and remove outliers).

Constraining a model to make it simpler and reduce the risk of overfitting is called ***regularization***.

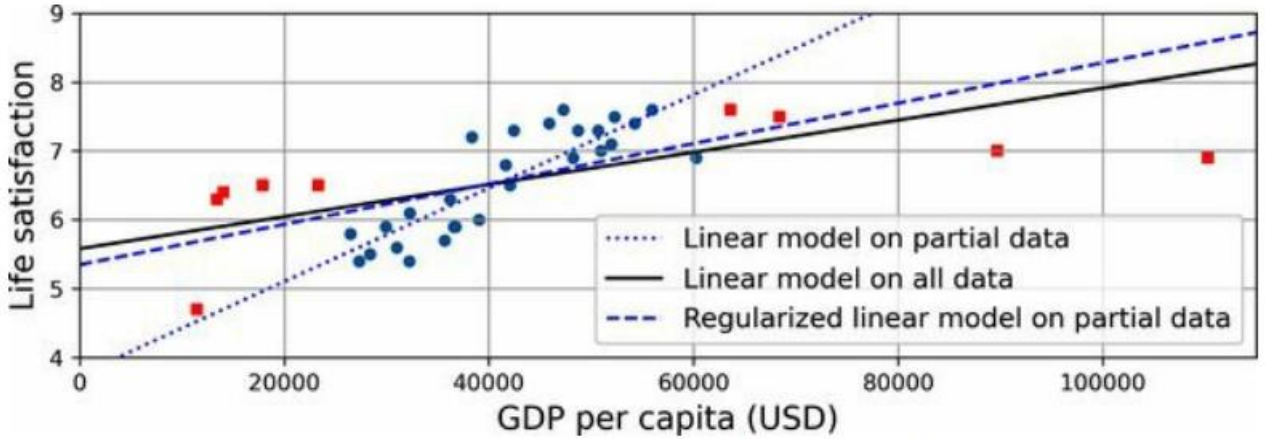
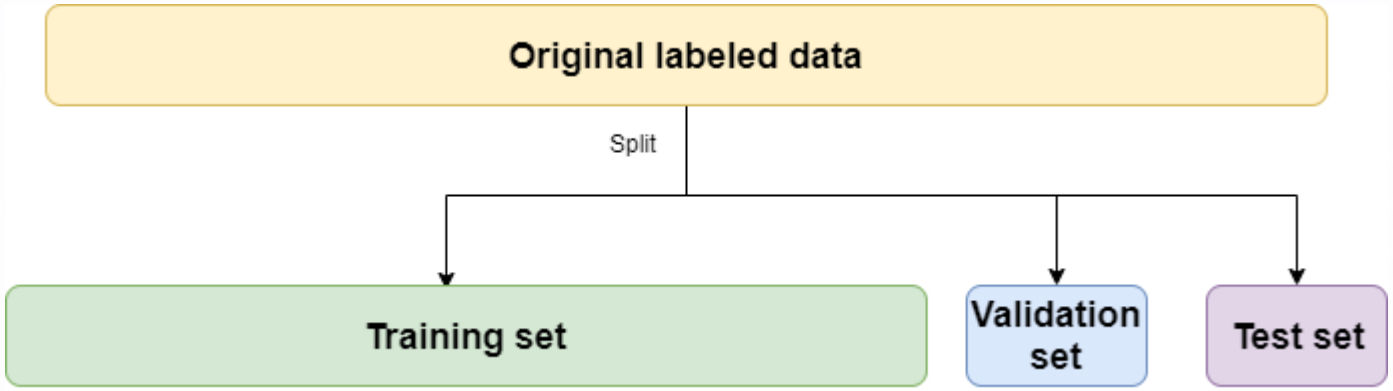


Figure 1-24. Regularization reduces the risk of overfitting

The amount of regularization to apply during learning can be controlled by a hyperparameter. A hyperparameter is a parameter of a learning algorithm (not of the model).

Testing and Validating



Hyperparameter Tuning and Model Selection

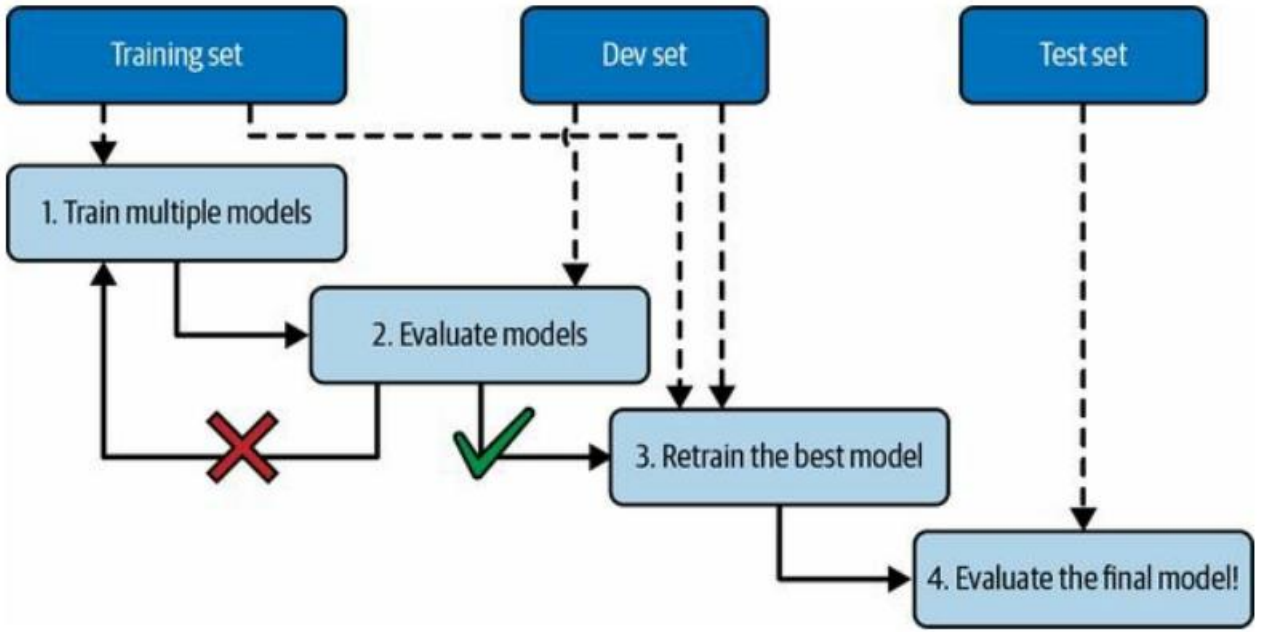


Figure 1-25. Model selection using holdout validation