Introduction to machine learning

CSCI-P 556

ZORAN TIGANJ

Today

- Chapter 1 from the *Hands On ML* textbook (The machine learning landscape)
  - Applications of ML
  - Types of ML
  - Main challenges in ML
- Next time we will cover chapter 2 (End to end machine learning project).

Announcements

- Download slides and watch videos of previous lectures on Canvas
- See all deadlines for assingments on Canvas
- Stay tuned for our first quiz – it will be released today (with deadline in a week).
- Use Q&A community to ask questions

Examples of Applications

- Analyzing images of products on a production line to automatically classify them
- Detecting tumors in brain scans
- Automatically classifying news articles
- Automatically flagging offensive comments on discussion forums
- Summarizing long documents automatically
- Creating a chatbot or a personal assistant
- Forecasting your company's revenue next year, based on m any performance metrics
- Making your app react to voice commands
- Detecting credit card fraud
- Segmenting clients based on t heir purchases so that you can design a different marketing strategy for each segment

Examples of Applications

- Representing a complex, high-dimensional dataset in a clear and insightful diagram
- Recommending a product that a client may be interested in, based on past purchases
- Building an intelligent bot for a game

Common divisions of ML algorithms

- Most common division of ML algorithms is based on the amount and type of supervision:
  - supervised learning
  - unsupervised learning
  - semisupervised learning
  - reinforcement Learning

Supervised learning

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

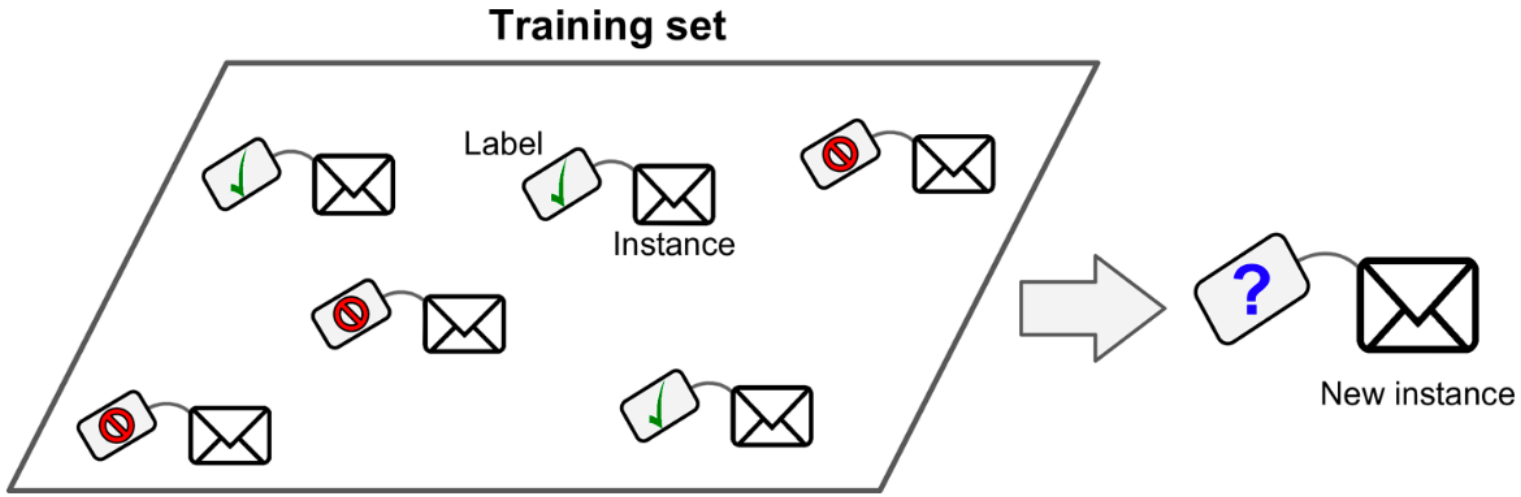

Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

Supervised learning

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

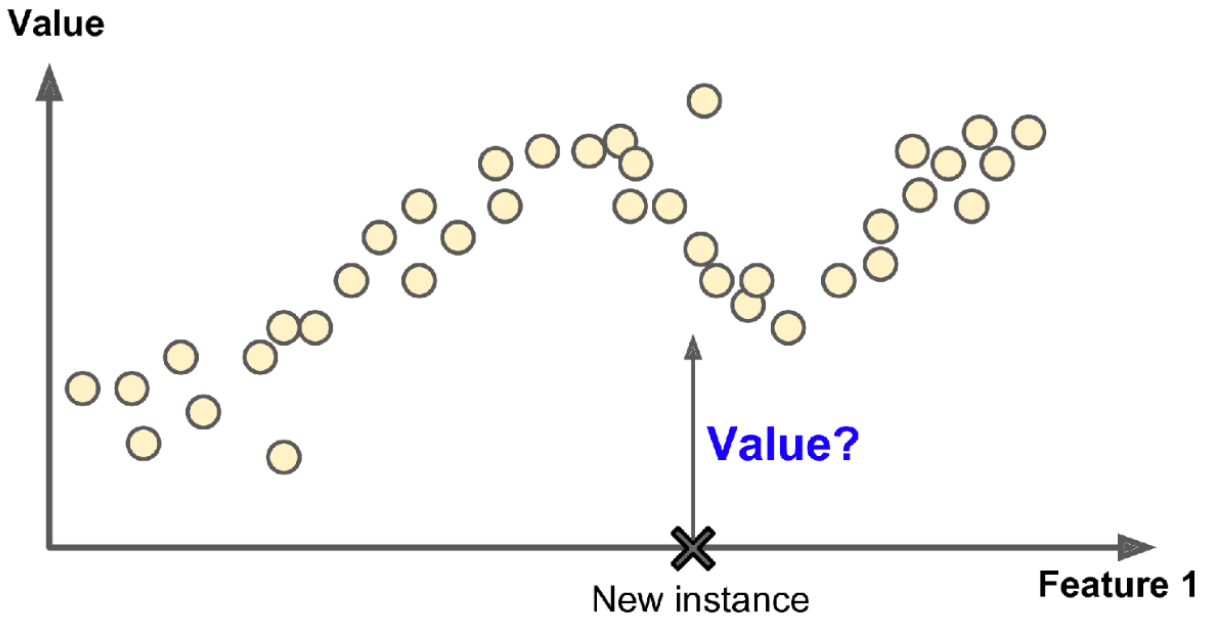

Figure 1-6. A regression problem: predict a value, given an input feature (there are usually multiple input features, and sometimes multiple output values)

Supervised learning algorithms

- Here are some of the most important supervised learning algorithms
  - k-Nearest Neighbors
  - Linear Regression
  - Logistic Regression
  - Support Vector Machines (SVMs)
  - Decision Trees and Random Forests

Unsupervised learning

- In unsupervised learning the training data is unlabeled.

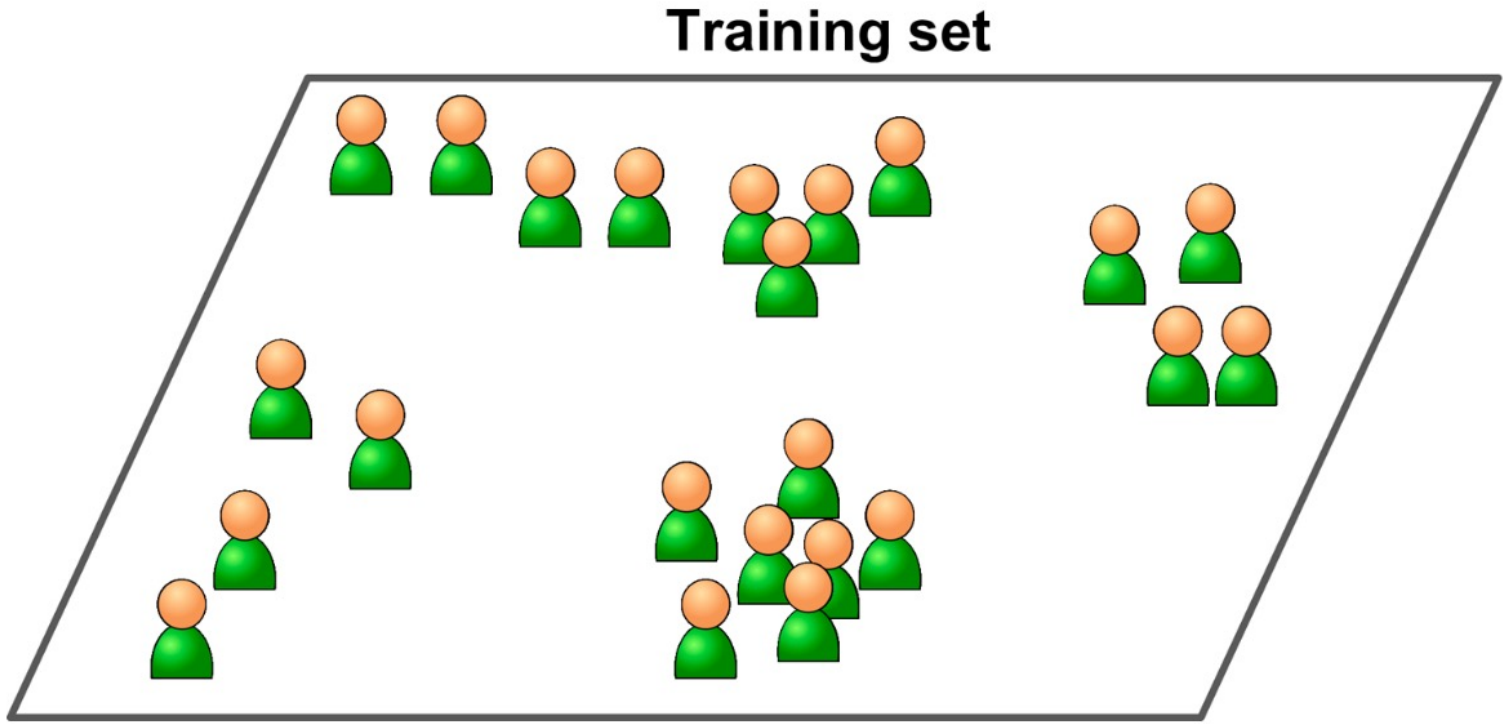

Figure 1-7. An unlabeled training set for unsupervised learning

Unsupervised learning

- In unsupervised learning the training data is unlabeled.

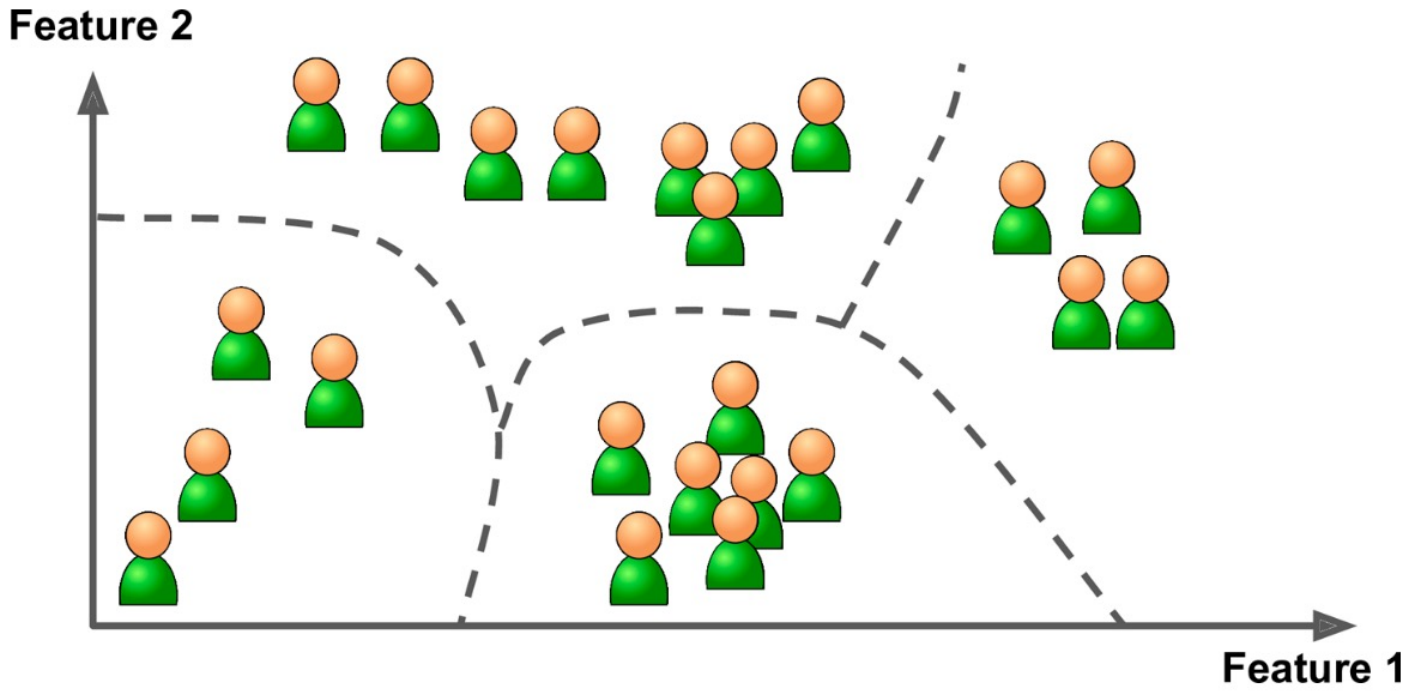

Figure 1-8. Clustering

Unsupervised learning

- In unsupervised learning the training data is unlabeled.
- Here are some of the most important unsupervised learning algorithms
  - Clustering
    - K-Means
  - Anomaly detection and novelty detection
    - One-class SVM
    - Isolation Forest
  - Visualization and dimensionality reduction
    - Principal Component Analysis (PCA)
    - Kernel PCA
    - Locally Linear Embedding (LLE)
    - t-Distributed Stochastic Neighbor Embedding (t-SNE)

Unsupervised learning: Data visualization with t-SNE

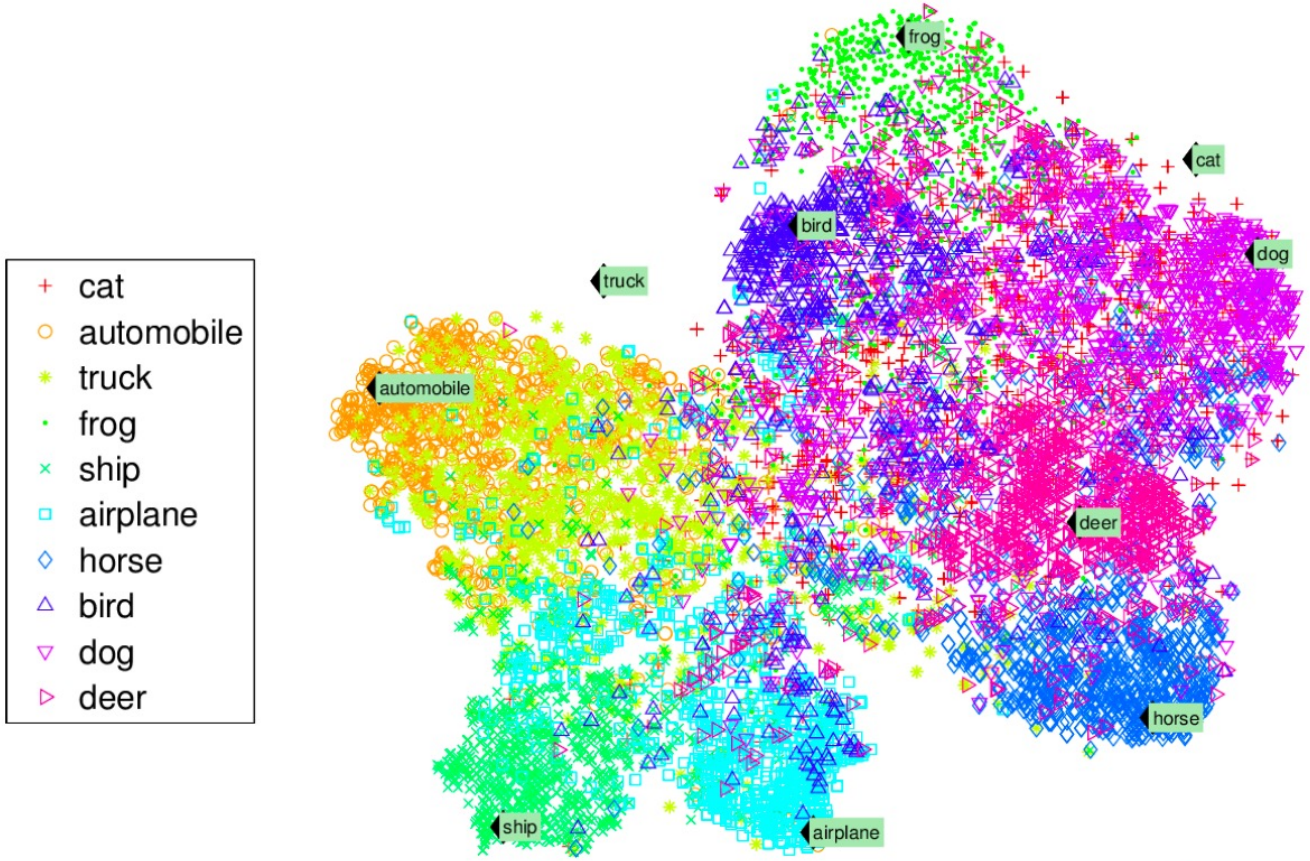

Figure 1-9. Example of a t-SNE visualization highlighting semantic clusters[3]

Unsupervised learning: anomaly detection

Figure 1-10. Anomaly detection

Semisupervised leanring

- Semisupervised leanring: plenty of unlabeled instances, and few labeled instances.

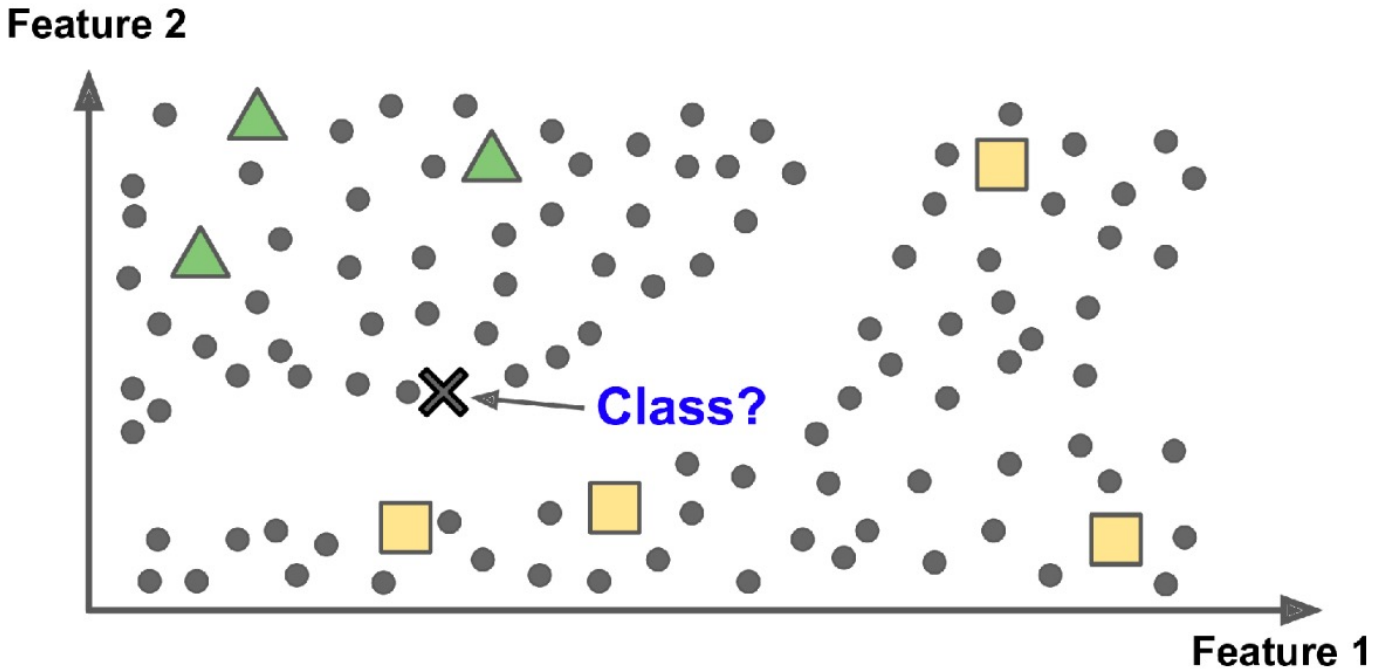

Figure 1-11. Semisupervised learning with two classes (triangles and squares): the unlabeled examples (circles) help classify a new instance (the cross) into the triangle class rather than the square class, even though it is closer to the labeled squares

Reinforcement Learning

- The learning system, called an agent, can observe the environment, select and perform actions, and get rewards in return (or penalties in the form of negative rewards).
- It must then learn by itself what is the best strategy, called a policy, to get the most reward over time.
- A policy defines what action the agent should choose when it is in a given situation

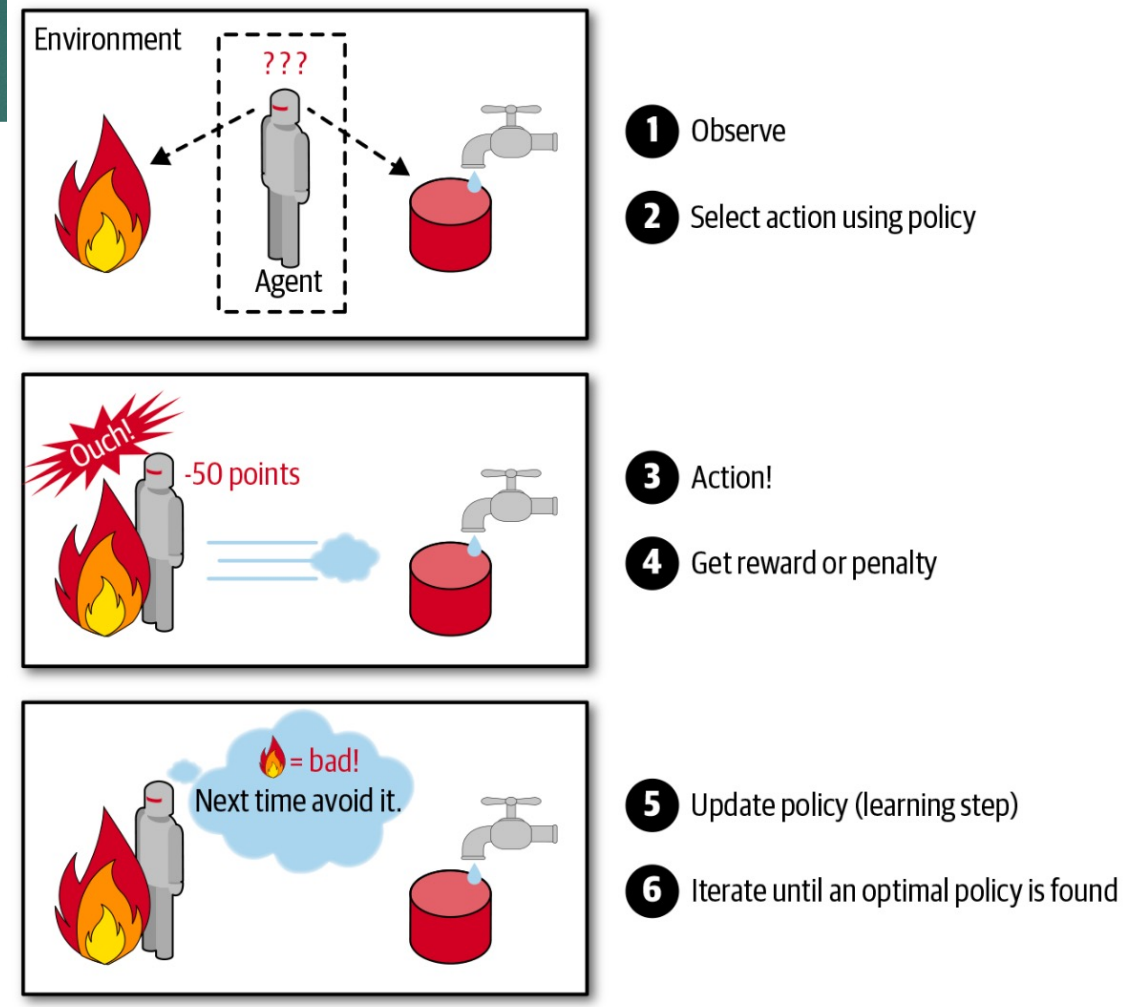

Figure 1-12. Reinforcement Learning

Batch and Online Learning

- In **batch learning**, the system is incapable of learning incrementally: it must be trained using all the available data.
- If you want a batch learning system to know about new data (such as a new type of spam), you need to train a new version of the system from scratch on the full dataset (not just the new data, but also the old data).

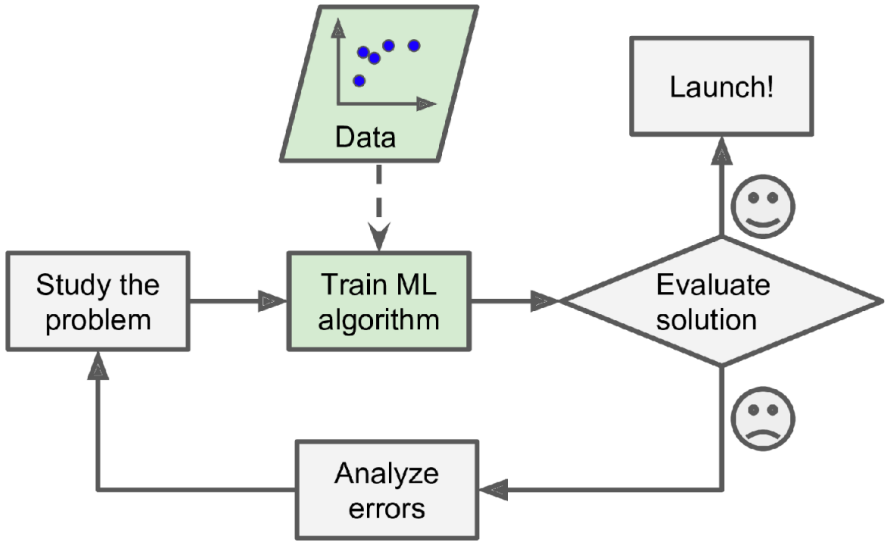

Figure 1-2. The Machine Learning approach

Batch and Online Learning

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

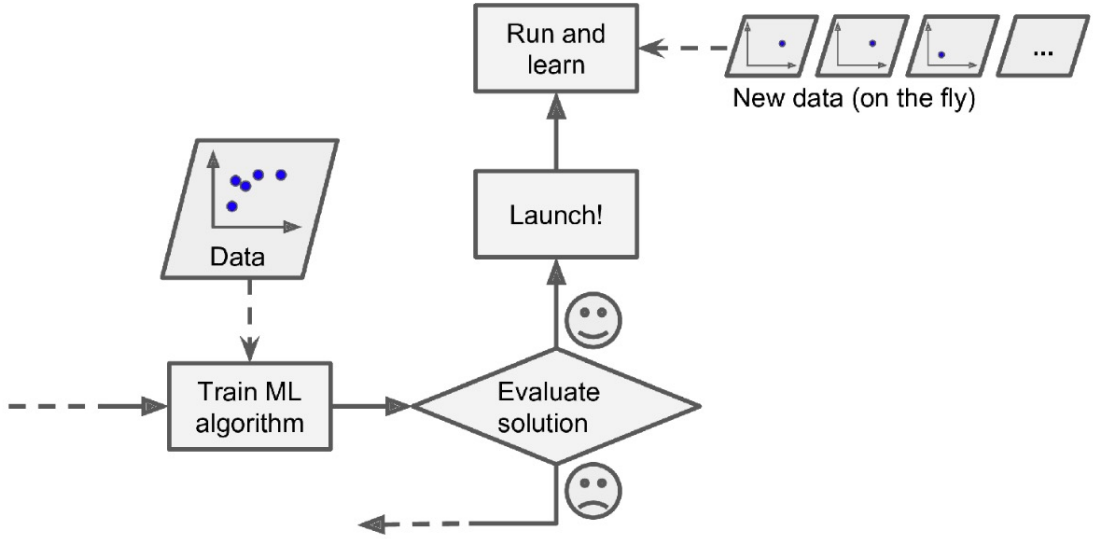

Figure 1-13. In online learning, a model is trained and launched into production, and then it keeps learning as new data comes in

Instance-Based Versus Model-Based Learning

- **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).

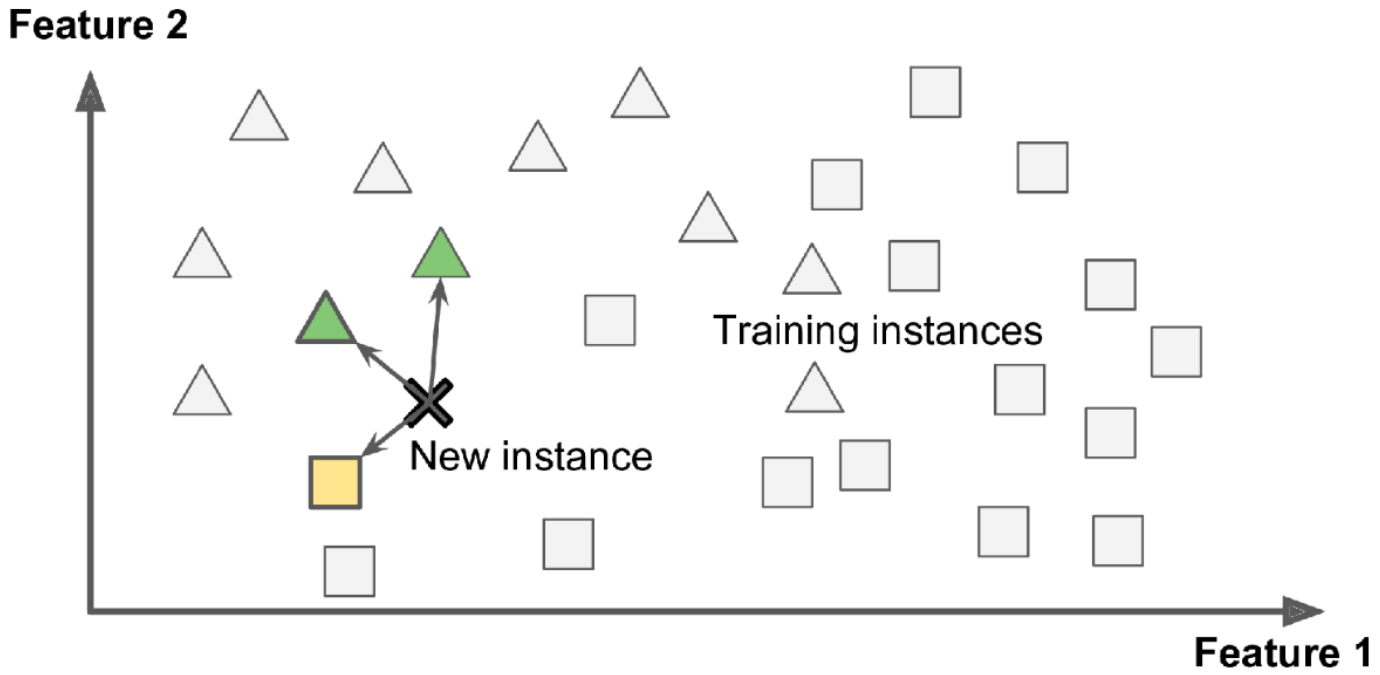

Figure 1-15. Instance-based learning

Instance-Based Versus Model-Based Learning

- **Model-based learning:** build a model of the available examples and then use that model to make predictions.

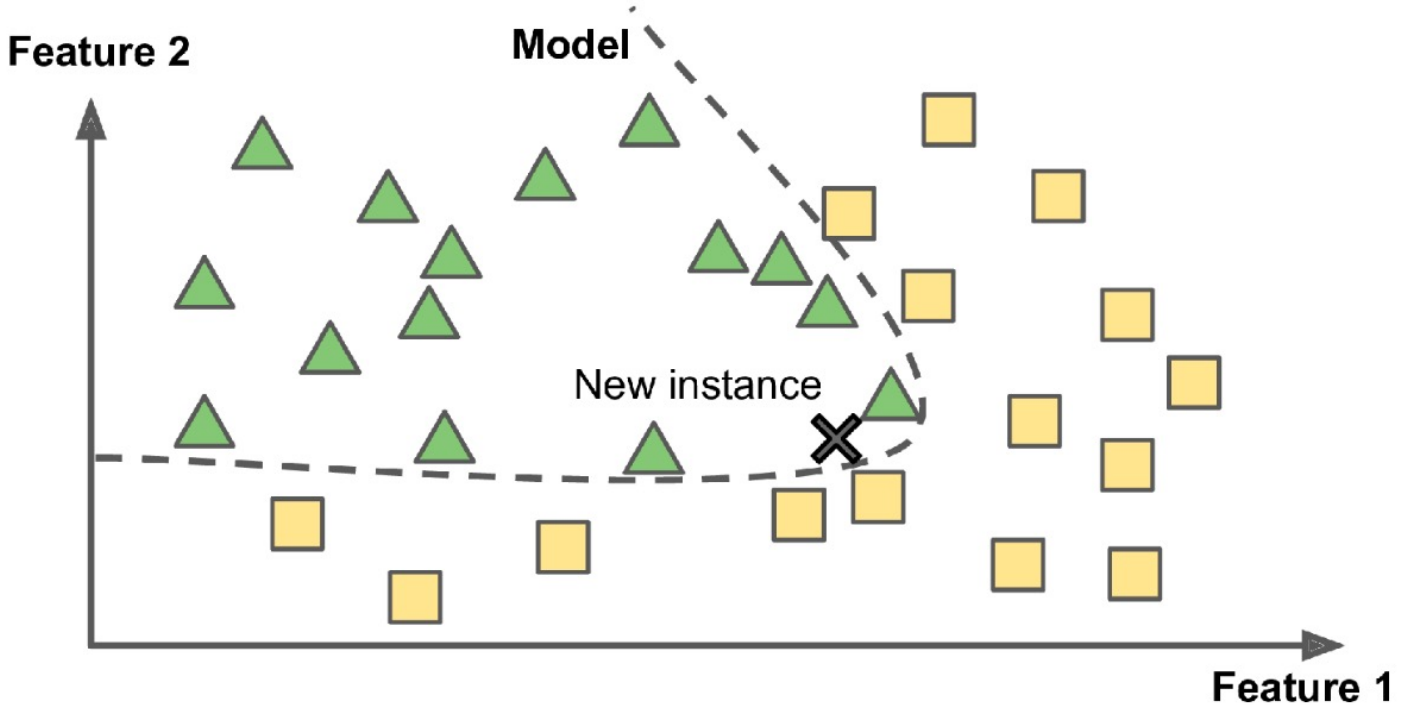

Figure 1-16. Model-based learning

Model-based learning: example

- Suppose you want to know if money makes people happy, so you download the Better Life Index data from the OECD's website and stats about gross domestic product (GDP) per capita from the IMF's website.

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

Model-based learning: example

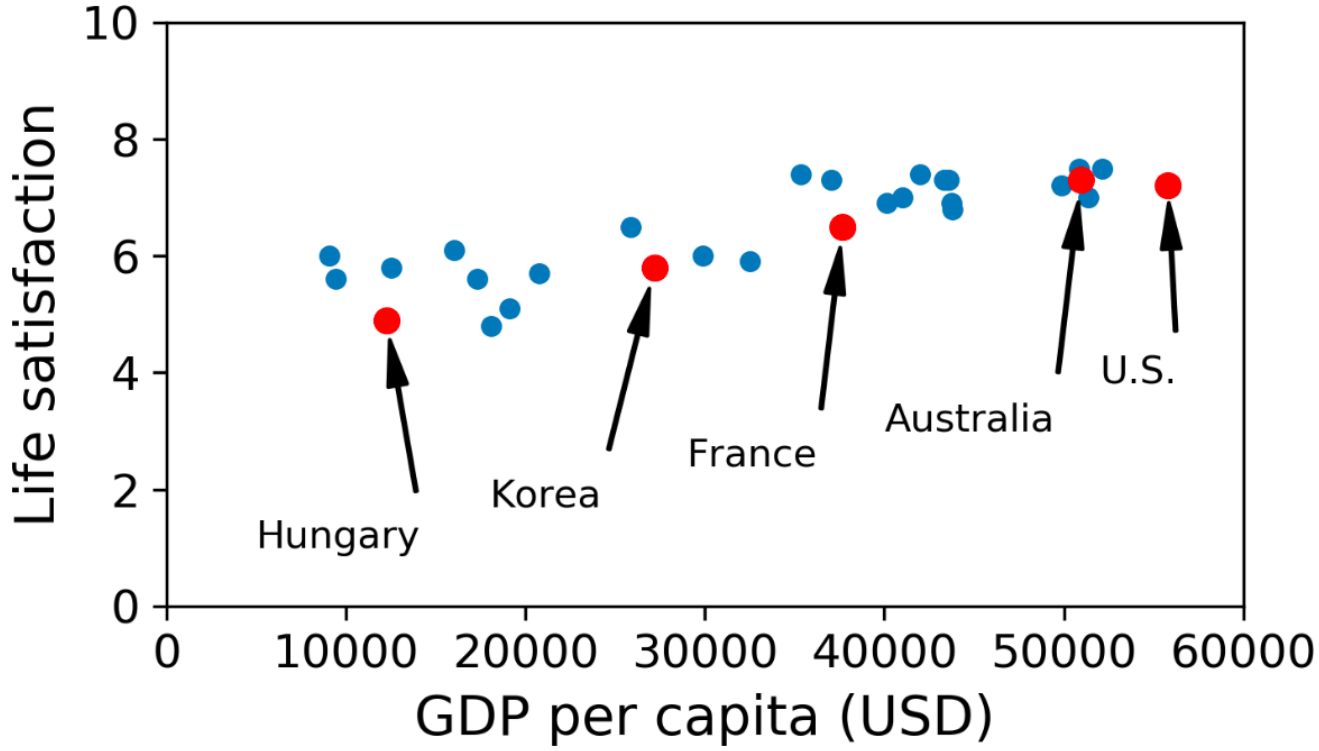

Figure 1-17. Do you see a trend here?

Model selection: we select a simple linear model with two parameters, θ_0 and θ_1 :

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

Model-based learning: example

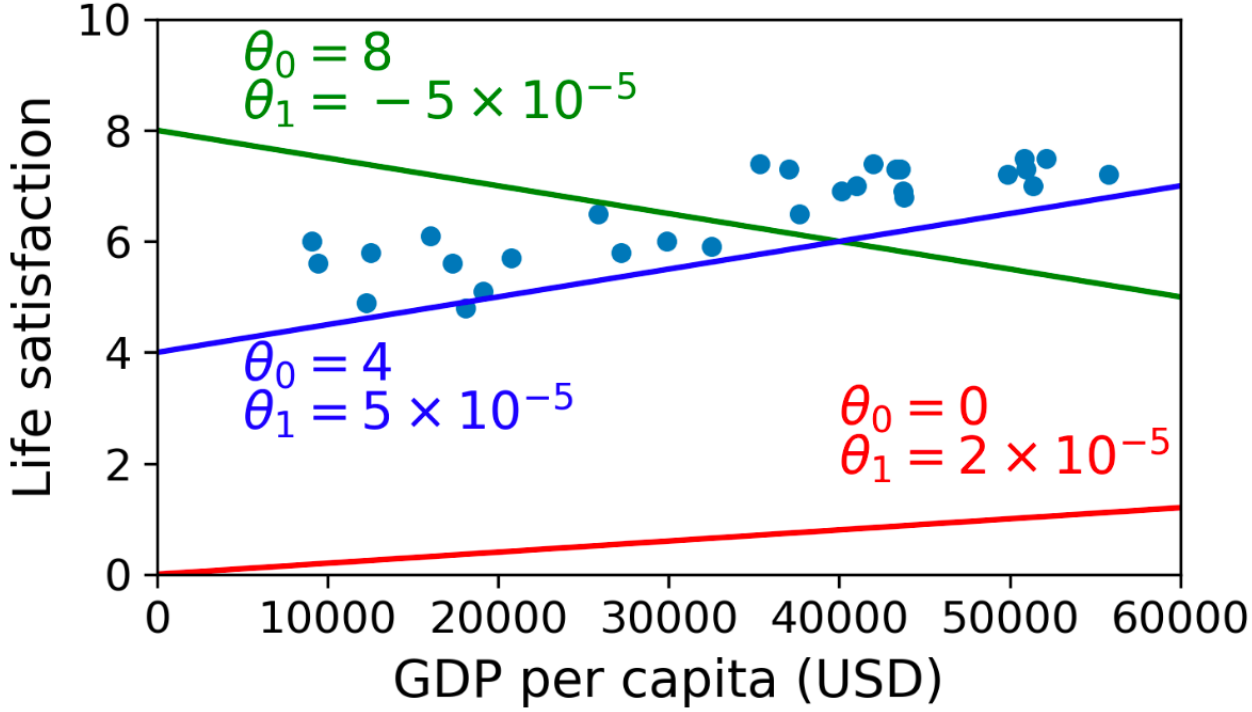

Figure 1-18. A few possible linear models

Model selection: we select a simple linear model with two parameters, θ_0 and θ_1:

life_satisfaction = θ_0 + θ_1 × GDP_per_capita

Model-based learning: example

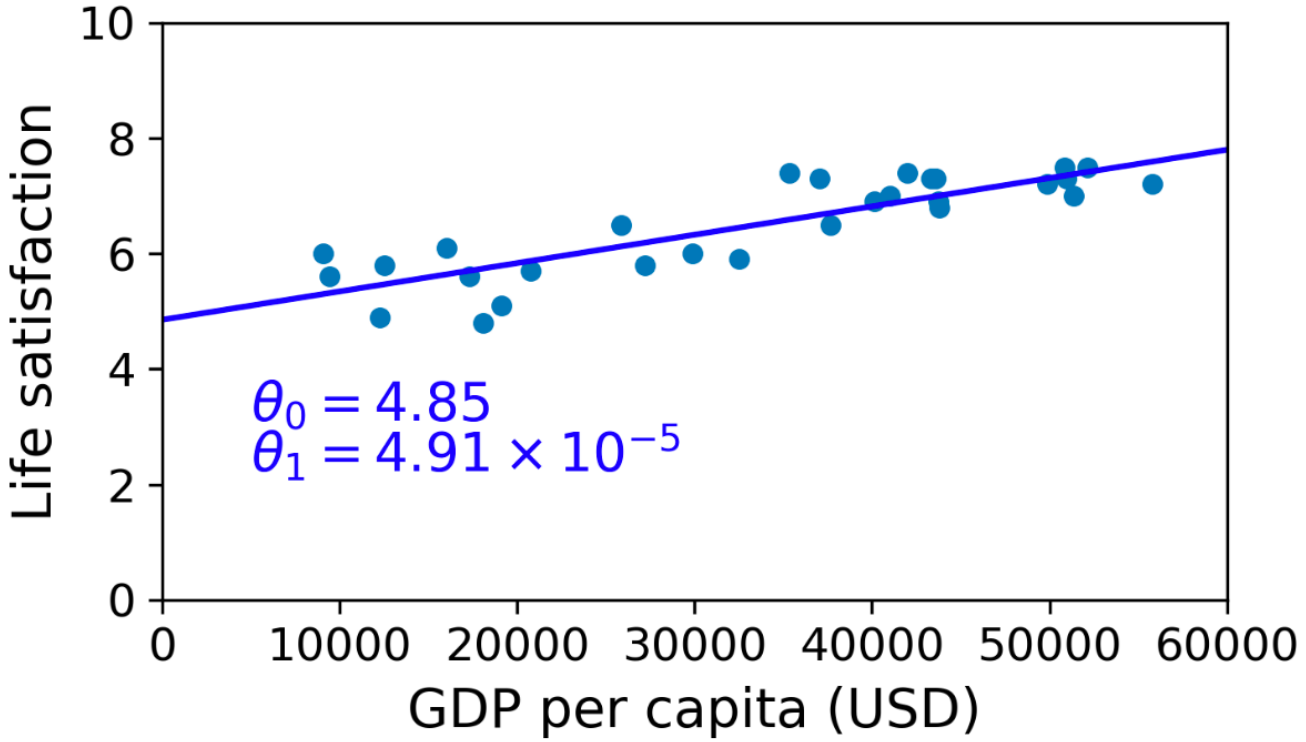

Figure 1-19. The linear model that fits the training data best

Model selection: we select a simple linear model with two parameters, θ_0 and θ_1 :

life_satisfaction = θ_0 + θ_1 × GDP_per_capita

Model-based learning: example

- Now we can use the model to predict life satisfaction:
  - For example, say you want to know how happy Cypriots are, and the OECD data does not have the answer. Fortunately, you can use your model to make a good prediction: you look up Cyprus's GDP per capita, find $22,587, and then apply your model and find that life satisfaction is likely to be somewhere around $4.85 + 22{,}587 \times 4.91 \times 10 = 5.96$.

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
import sklearn.linear_model

# Load the data
oecd_bli = pd.read_csv("oecd_bli_2015.csv", thousands=',')
gdp_per_capita =
pd.read_csv("gdp_per_capita.csv",thousands=',',delimiter='\t',
                                encoding='latin1', na_values="n/a")


# Prepare the data
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita)
X = np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# Visualize the data
country_stats.plot(kind='scatter', x="GDP per capita", y='Life
satisfaction')
plt.show()

# Select a linear model
model = sklearn.linear_model.LinearRegression()

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

# Make a prediction for Cyprus
X_new = [[22587]]  # Cyprus's GDP per capita
print(model.predict(X_new)) # outputs [[ 5.96242338]]
```

Main Challenges of Machine Learning

- Insufficient Quantity of Training Data
  - 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.

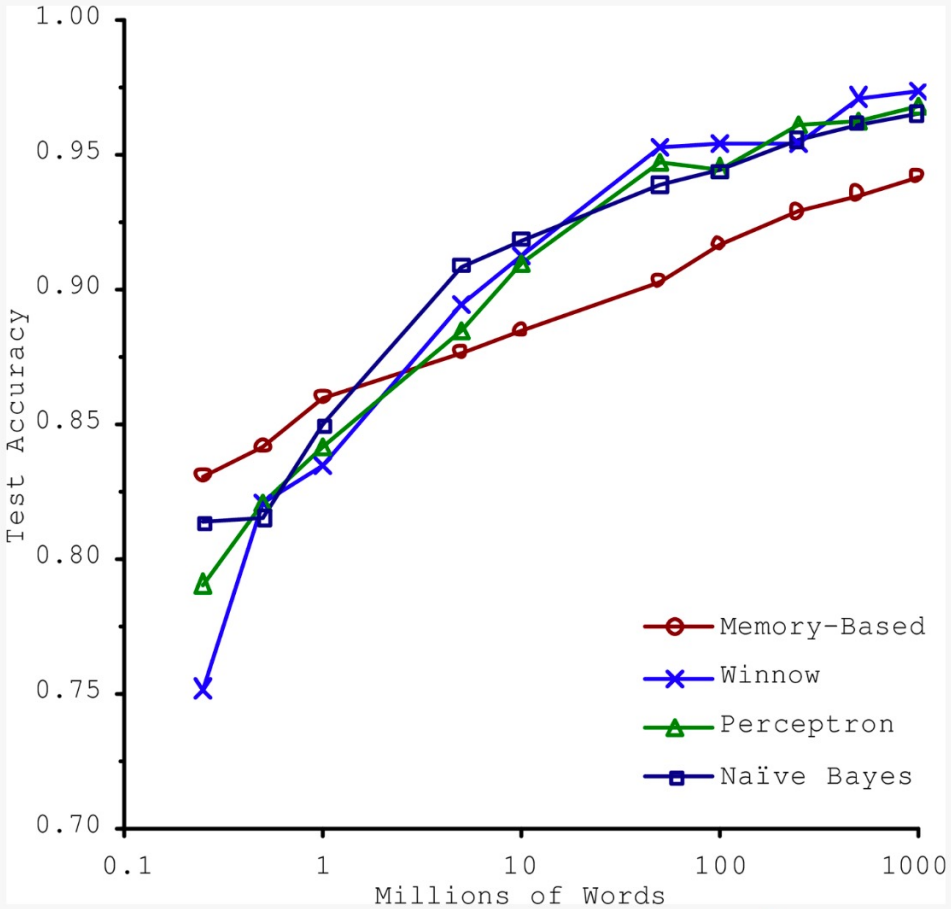

Figure 1-20. The importance of data versus algorithms[9]

Main Challenges of Machine Learning

- Nonrepresentative Training Data (sampling bias)

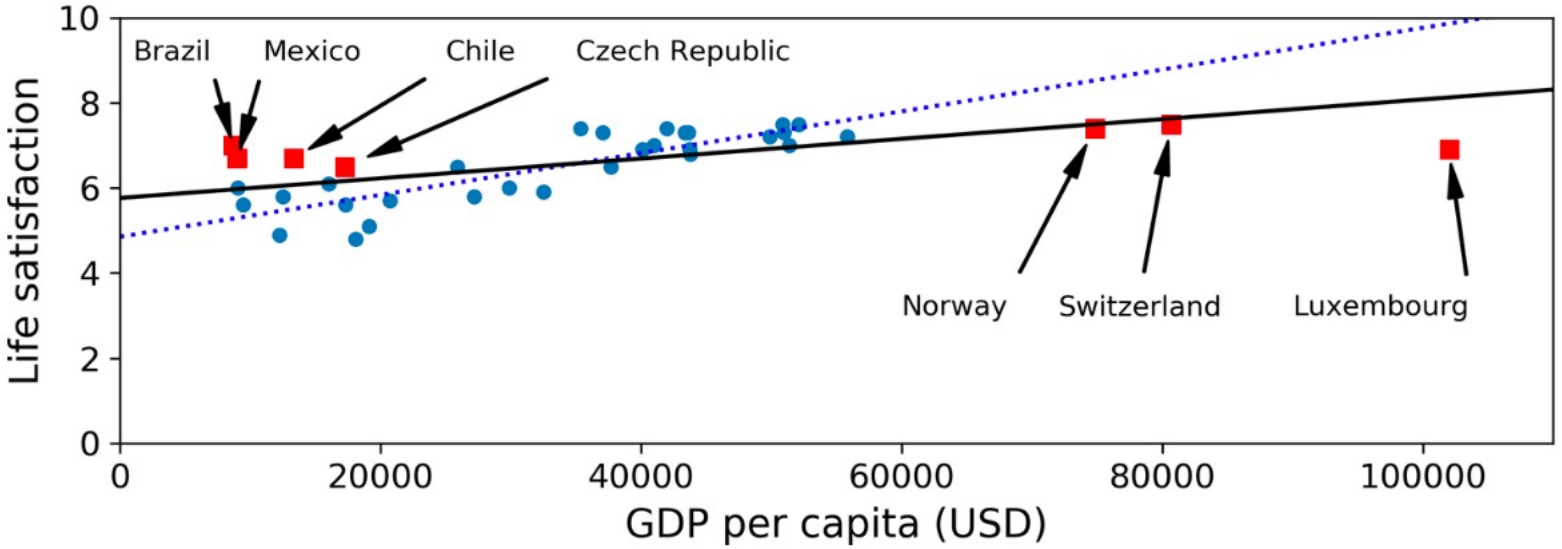

Figure 1-21. A more representative training sample

Main Challenges of Machine Learning

- Poor-Quality 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.

Main Challenges of Machine Learning

- Irrelevant Features:
  - garbage in, garbage out
  - feature engineering can help (selecting the most useful features to train on among existing features)

Main Challenges of Machine Learning

- Overfitting the Training Data
  - Example of a high-degree polynomial life satisfaction model that strongly overfits the training data. Even though it performs much better on the training data than the simple linear model.

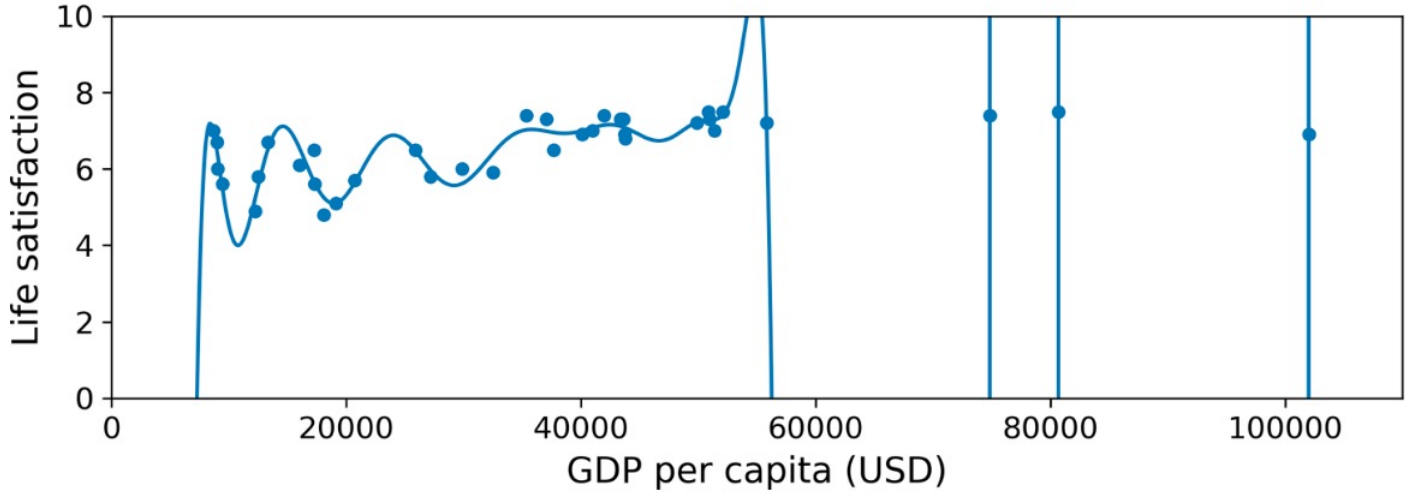

Figure 1-22. Overfitting the training data

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

Main Challenges of Machine Learning

- Underfitting the Training Data
  - opposite of overfitting: it occurs when your model is too simple to learn the underlying structure of the data.
  - For example, a linear model of life satisfaction is prone to underfit; reality is just more complex than the model, so its predictions are bound to be inaccurate, even on the training examples.

Testing and Validating

- To ensure that our algorithm is not overfitting the training data it is common to split the data into two sets:
  - the training set and
  - the test set
- You train your model using the training set, and you test it using the test set.
- The error rate on new cases is called the generalization error (or out-of-sample error),

Hyperparameter Tuning and Model Selection

- How do you choose the value of the regularization hyperparameter?
  - One option is to train 100 different models using 100 different values for this hyperparameter.
  - If you do this and evaluate the model on test set, it is likely going to lead to overfitting.
  - The problem is that you measured the generalization error multiple times on the test set, and you adapted the model and hyperparameters to produce the best model for that particular set
  - A common solution to this problem is called holdout validation: you simply hold out part of the training set to evaluate several candidate models and select the best one.

Next time

- Chapter 2 of the textbook: End-to-End Machine Learning Project