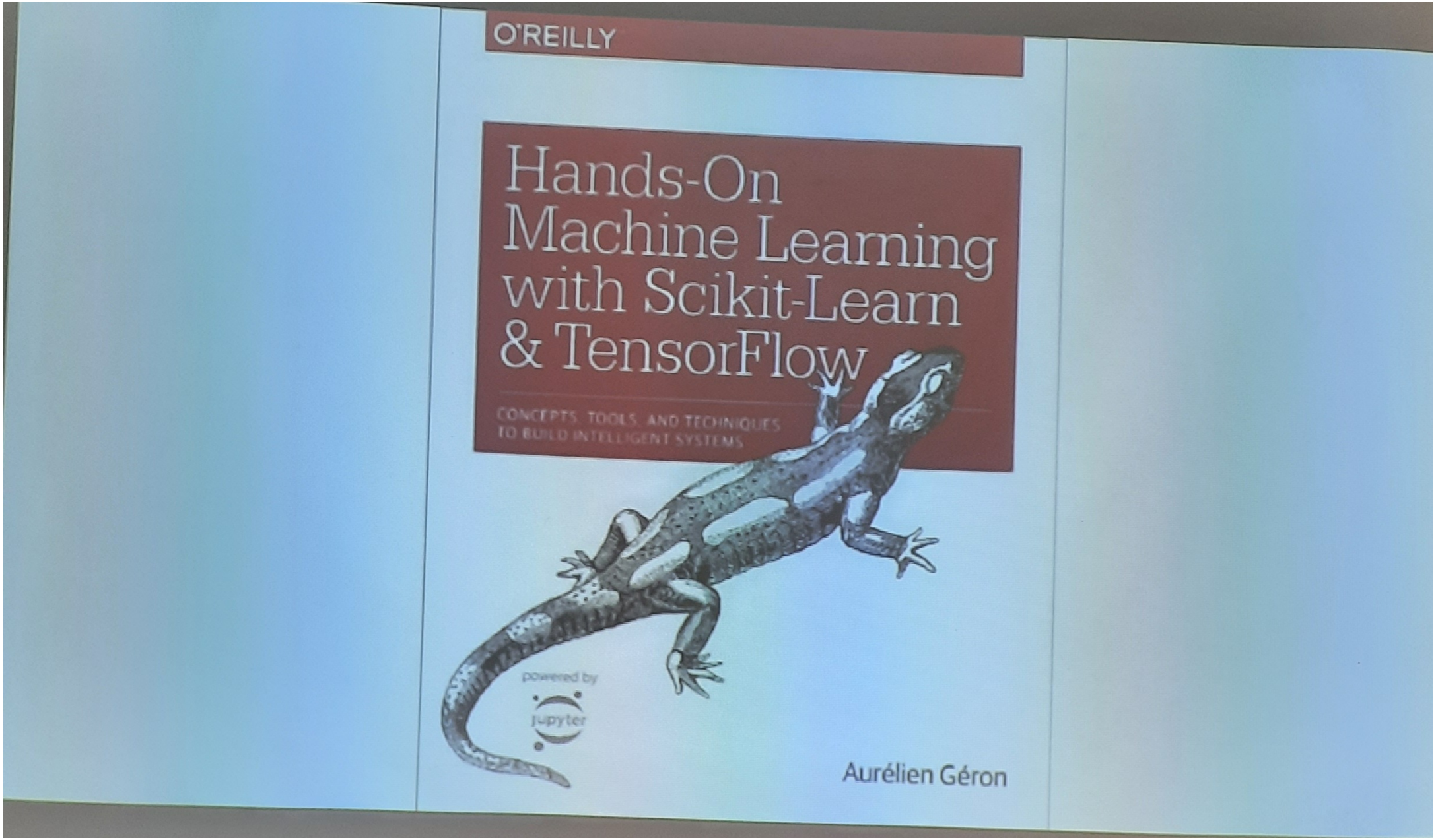
What Is Machine Learning?


O'REILLY
Hands-On Machine Learning with Scikit-Learn & TensorFlow
CONCEPTS, TOOLS, AND TECHNIQUES TO BUILD INTELLIGENT SYSTEMS
powered by
jupyter
Aurélien Géron

- This book assumes that you have some Python programming experience and that you are familiar with Python's main scientific libraries, in particular NumPy, Pandas, and Matplotlib.
- Also, if you care about what's under the hood you should have a reasonable understanding of college- level math as well (calculus, linear algebra, probabilities, and statistics).
- Supplemental material (code examples, exercises, etc.) is available for download at *https://github.com/ageron/handson-ml*.

- This book is organized in two parts.
  - Part I, *The Fundamentals of Machine Learning*, covers the following topics:
    - What is Machine Learning? What problems does it try to solve? What are the main categories and fundamental concepts of Machine Learning systems?
    - The main steps in a typical Machine Learning project. Learning by fitting a model to data. Optimizing a cost function. Handling, cleaning, and preparing data.
    - Selecting and engineering features. Selecting a model and tuning hyperparameters using cross-validation.
    - The main challenges of Machine Learning, in particular underfitting and overfitting (the bias/variance tradeoff).
    - Reducing the dimensionality of the training data to fight the curse of dimensionality.
    - The most common learning algorithms: Linear and Polynomial Regression, Logistic Regression, k- Nearest Neighbors, Support Vector Machines, Decision Trees, Random Forests, and Ensemble methods.

- Machine Learning is the science (and art) of programming computers so they can *learn from data*.
- Here is a slightly more general definition:

  [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*

- Spam filter is a Machine Learning program that can learn to flag spam given examples of spam emails (e.g., flagged by users) and examples of regular (nonspam, also called “ham”) emails.
- The examples that the system uses to learn are called the *training set*. Each training example is called a *training instance* (or *sample*).
- In this case, the task T is to flag spam for new emails, the experience E is the *training data*, and the performance measure P needs to be defined; for example, you can use the ratio of correctly classified emails.
- This particular performance measure is called *accuracy* and it is often used in classification tasks.

Why Use Machine Learning? – Spam Filter

- First you would look at what spam typically looks like. 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. Perhaps you would also notice a few other patterns in the sender's name, the email's body, and so on.
- 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 are detected.
- You would test your program, and repeat steps 1 and 2 until it is good enough.

- Since the problem is not trivial, your program will likely become a long list of complex rules — pretty hard to maintain.
- 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 (Figure 1-2). The program is much shorter, easier to maintain, and most likely more accurate.

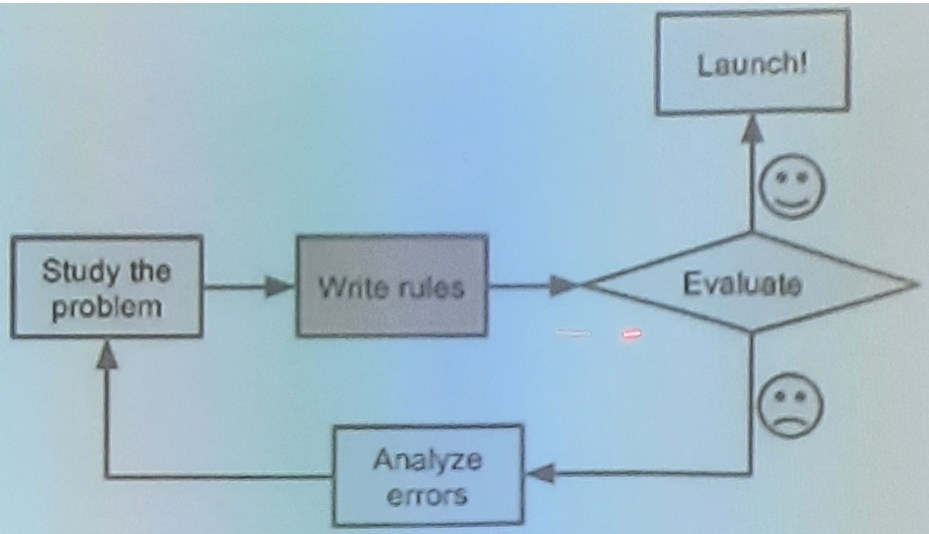


Figure 1-1. The traditional approach

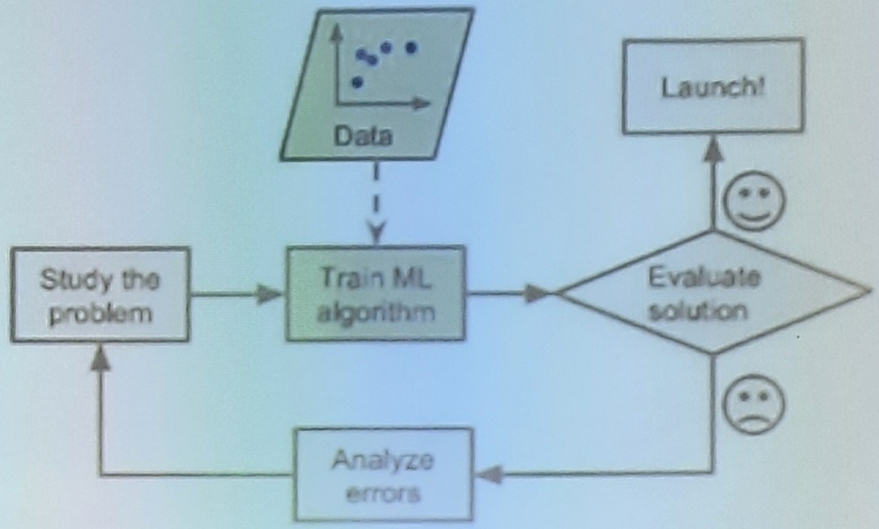


Figure 1-2. Machine Learning approach

- 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. If spammers keep working around your spam filter, you will need to keep writing new rules forever.
- 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).

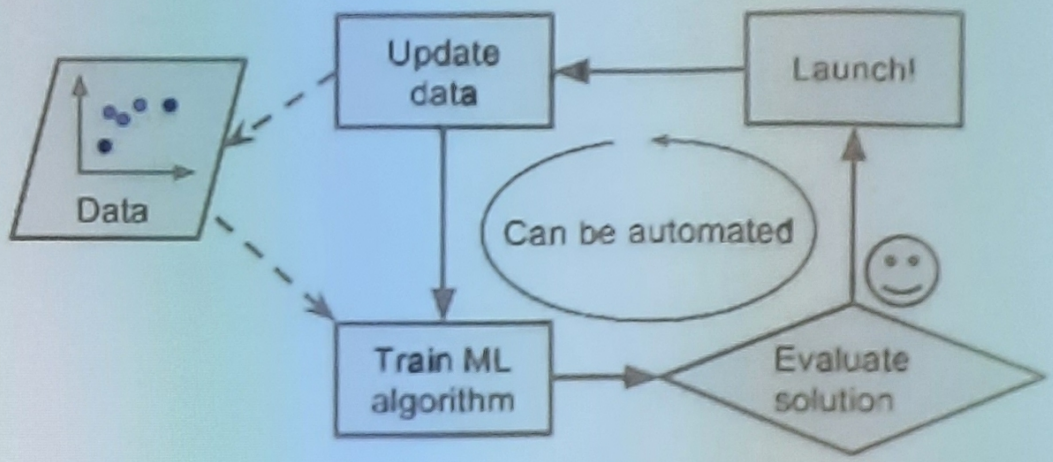


Figure 1-3. Automatically adapting to change

- Finally, Machine Learning can help humans learn (Figure 1-4): ML algorithms can be inspected to see what they have learned (although for some algorithms this can be tricky). For instance, once the 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.
- Applying ML techniques to dig into large amounts of data can help discover patterns that were not immediately apparent. This is called *data mining*.

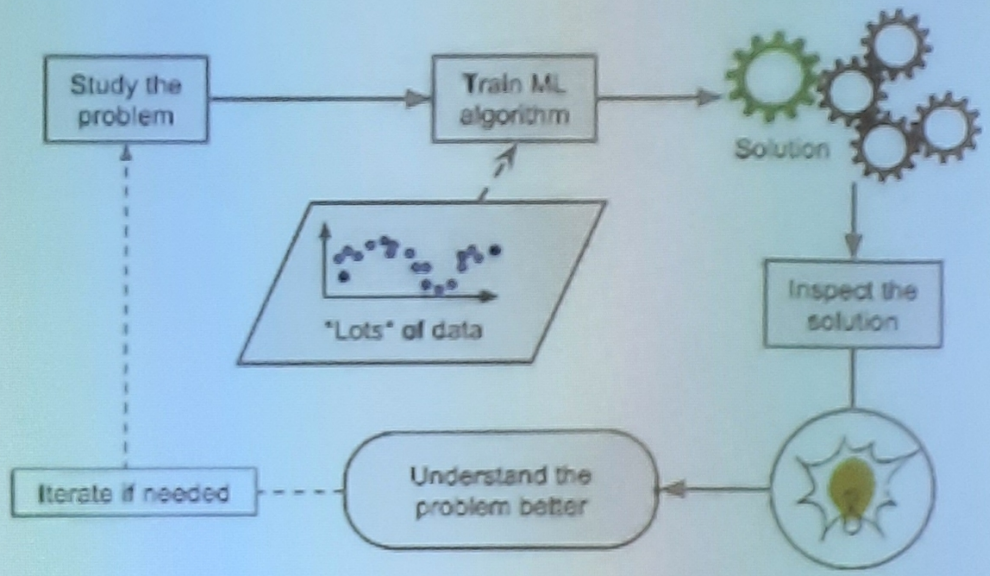


Figure 1-4. Machine Learning can help humans learn

- Machine Learning is great for:
  - Problems for which existing solutions require a lot of hand-tuning or long lists of rules: one Machine Learning algorithm can often simplify code and perform better.
  - Complex problems for which there is no good solution at all using a traditional approach: the best Machine Learning techniques can find a solution.
  - Fluctuating environments: a Machine Learning system can adapt to new data. Getting insights about complex problems and large amounts of data.

Types of Machine Learning Systems

- Whether or not they are trained with human supervision (**supervised, unsupervised, semisupervised, and Reinforcement Learning**)
- 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 detect patterns in the training data and build a predictive model, much like scientists do (**instance-based versus model-based learning**)

Supervised/Unsupervised Learning

- Machine Learning systems can be classified according to the amount and type of supervision they get during training.
- There are four major categories:
  - supervised learning,
  - unsupervised learning,
  - semisupervised learning,
  - Reinforcement Learning.

Supervised learning

- In *supervised learning*, the training data you feed to the algorithm includes the desired solutions, called *labels* (Figure 1-5).
- A typical supervised learning task is *classification*. The spam filter is a good example of this: it is trained with many example emails along with their *class* (spam or ham), and it must learn how to classify new emails.
- 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.) called *predictors*. This sort of task is called *regression* (Figure 1-6).

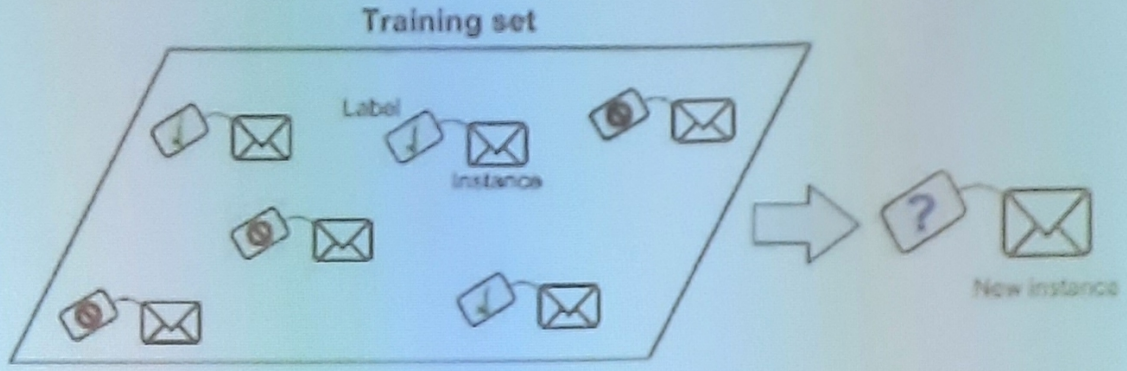


Figure 1-5. A labeled training set for supervised learning (e.g., spam classification)

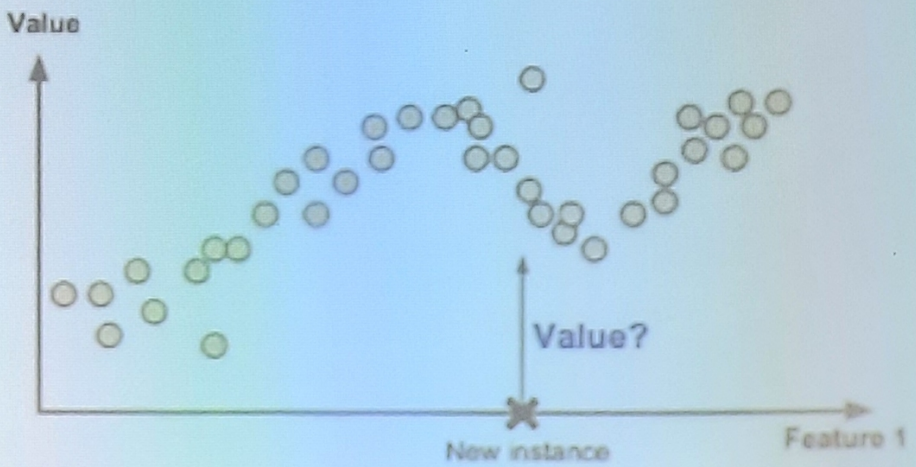


Figure 1-6. Regression

Supervised learning

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

Unsupervised learning

- *Unsupervised learning*, as you might guess, the training data is unlabeled. The system tries to learn without a teacher.

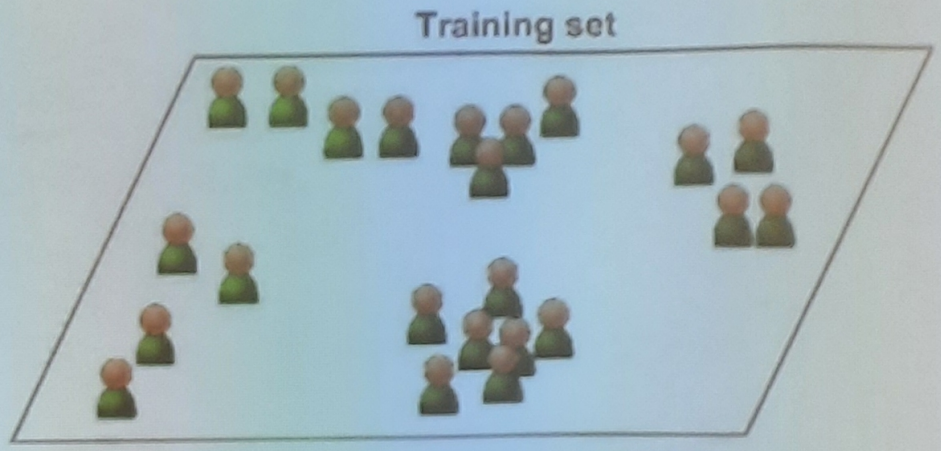


Figure 1-7. An unlabeled training set for unsupervised learning

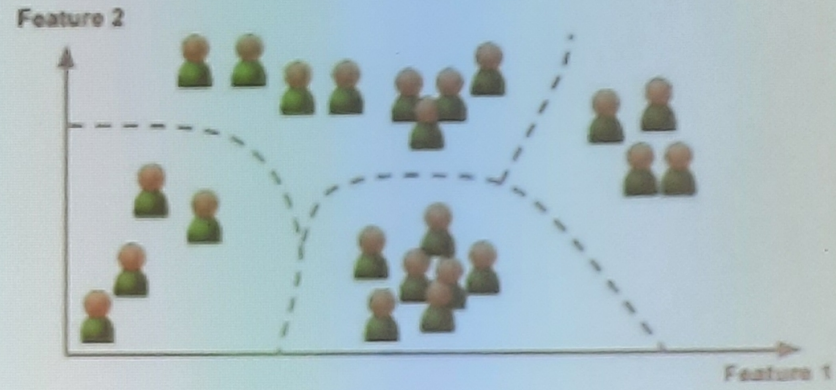


Figure 1-8. Clustering

Unsupervised learning

- Some of the most important unsupervised learning algorithms
  - Clustering
    - k-Means
    - Hierarchical Cluster Analysis (HCA)
    - Expectation Maximization
  - Visualization and dimensionality reduction
    - Principal Component Analysis (PCA)
    - Kernel PCA
    - Locally-Linear Embedding (LLE)
    - t-distributed Stochastic Neighbor Embedding (t-SNE)
  - Association rule learning
    - Apriori
    - Eclat

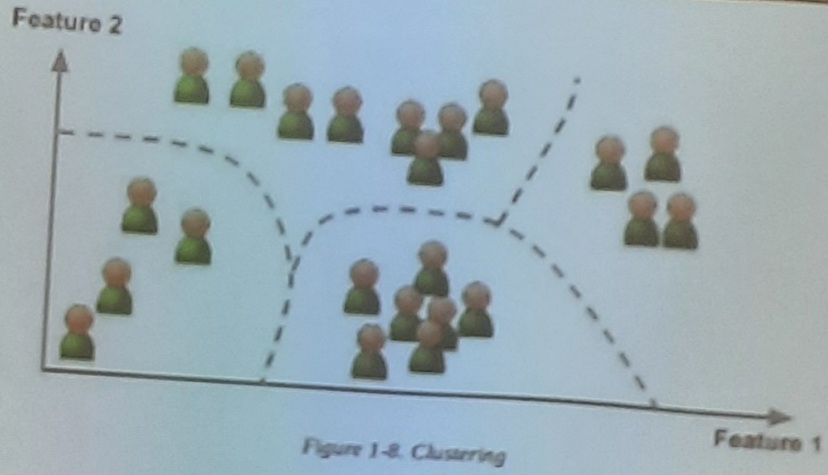


Figure 1-8. Clustering

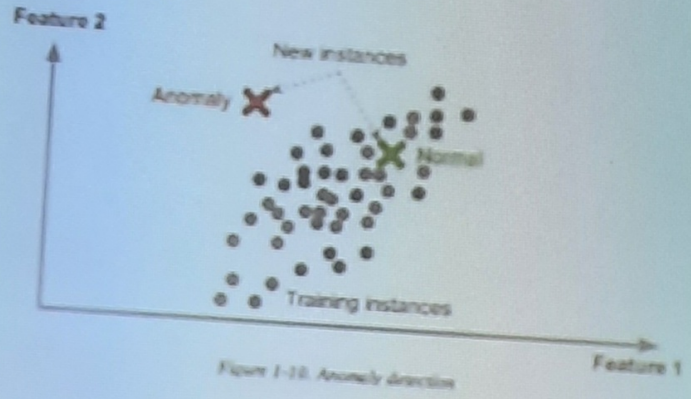


Figure 1-10. Anomaly detection

Semisupervised learning

- Algorithms can deal with partially labeled training data, usually a lot of unlabeled data and a little bit of labeled data. This is called *semisupervised learning* (Figure 1-11).
- Semisupervised learning algorithms are combinations of unsupervised and supervised algorithms

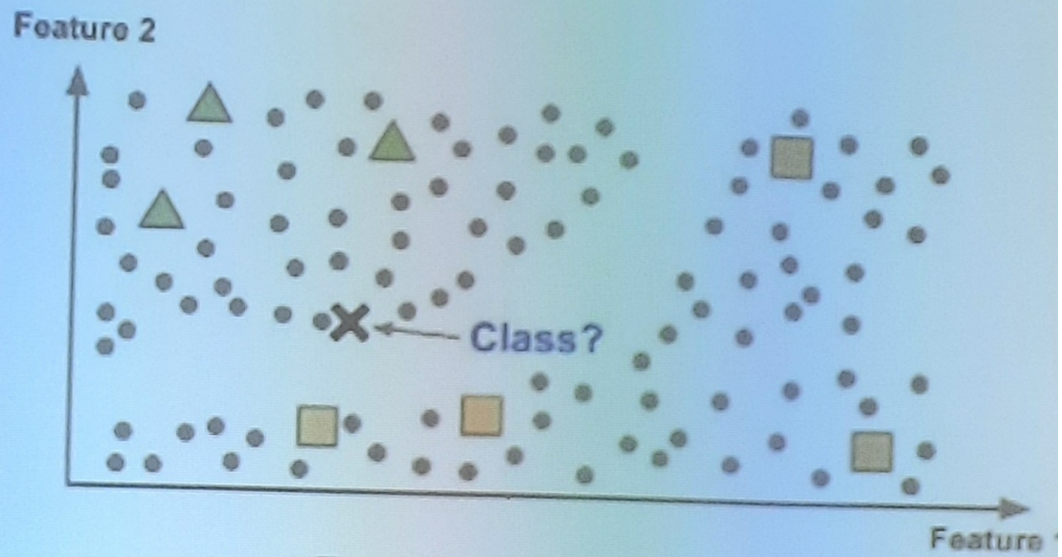


Figure 1-11. Semisupervised learning

Reinforcement Learning

- *Reinforcement Learning* is a very different beast.
- The learning system, called an *agent* in this context, can observe the environment, select and perform actions, and get *rewards* in return (or *penalties* in the form of negative rewards, as in Figure 1-12).
- It must then learn by itself what is the best strategy, called a *policy*, to get the most reward over time.

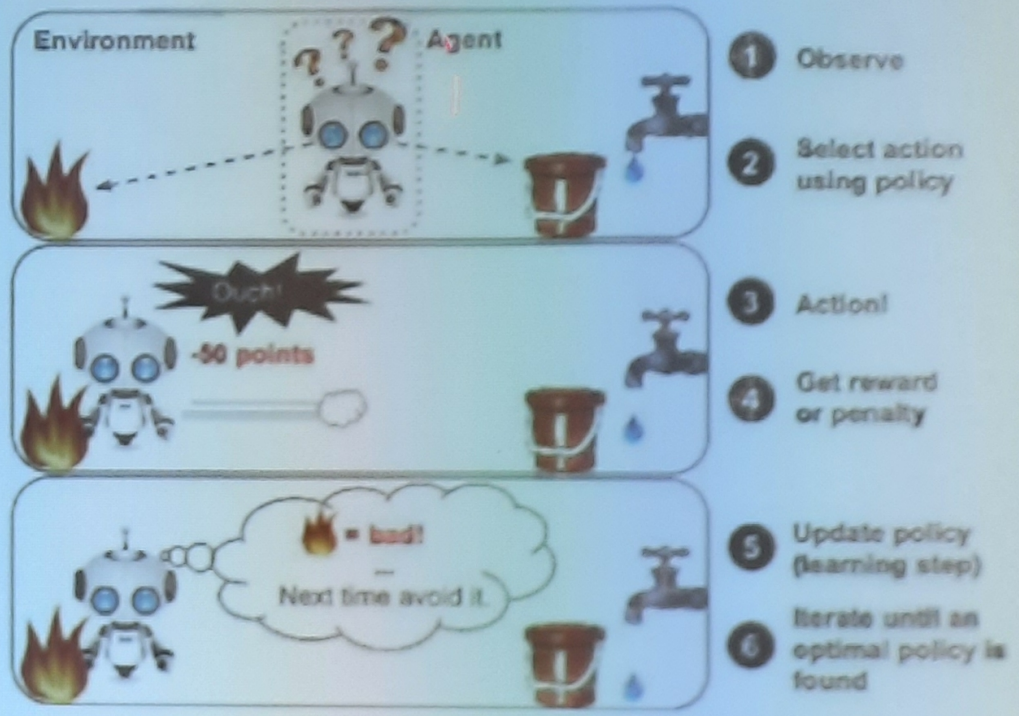


Figure 1-12. Reinforcement Learning

Batch and Online Learning

- **Batch learning**
  - In *batch learning*, the system is incapable of learning incrementally: it must be trained using all the available data. This will generally take a lot of time and computing resources, so it is typically done offline. First the system is trained, and then it is launched into production and runs without learning anymore; it just applies what it has learned. This is called *offline learning*
- **Online learning**
  - In *online learning*, you train the system incrementally by feeding it data instances sequentially, either individually or by 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

Online Learning

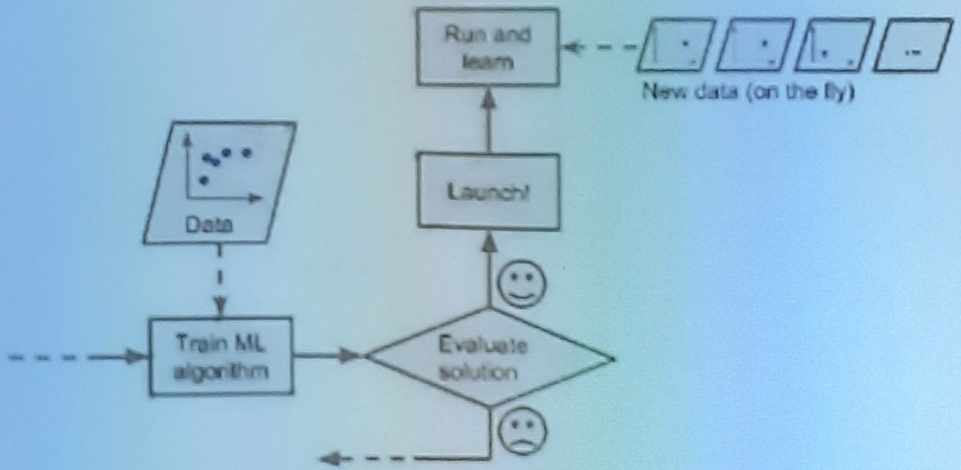


Figure 1-13. Online learning

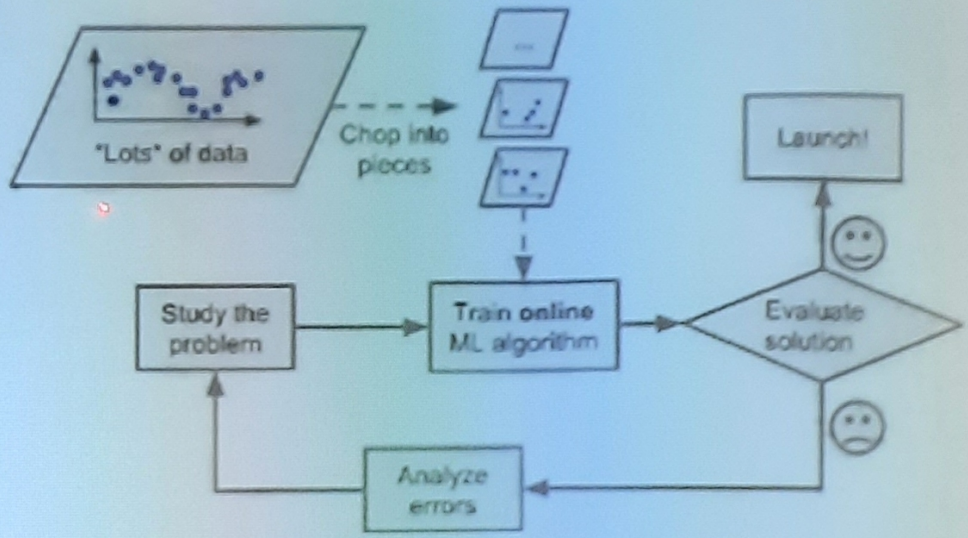


Figure 1-14. Using online learning to handle huge datasets

Online learning is great for systems that receive data as a continuous flow (e.g., stock prices) and need to adapt to change rapidly or autonomously. It is also a good option if you have limited computing resources: once an online learning system has learned about new data instances

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

if you set a low learning rate, the system will have more inertia; that is, it will learn more slowly, but it will also

Instance-Based Versus Model-Based Learning

- *Instance-based learning*: the system learns the examples by heart, then generalizes to new cases using a similarity measure (Figure 1-15).
- *Model-based learning:* generalize from a set of examples is to build a model of these examples, then use that model to make *predictions* (Figure 1-16).

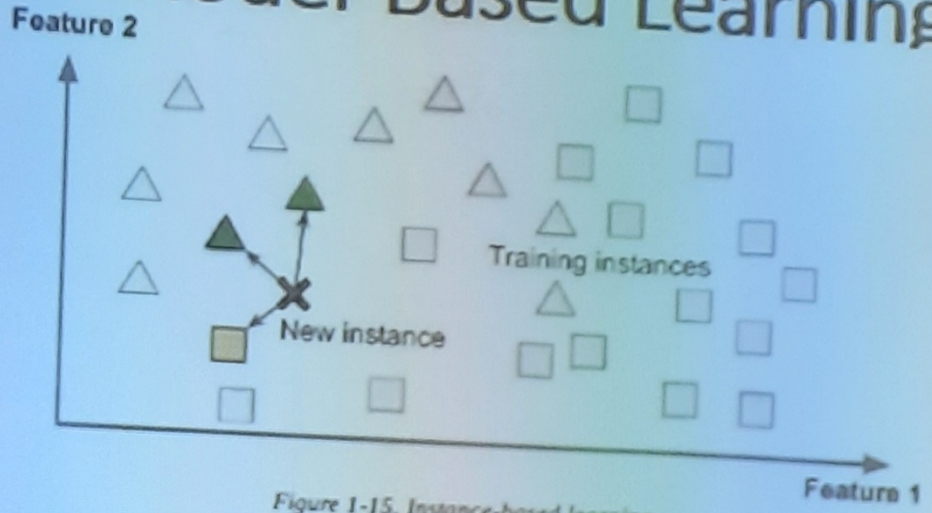


Figure 1-15. Instance-based learning

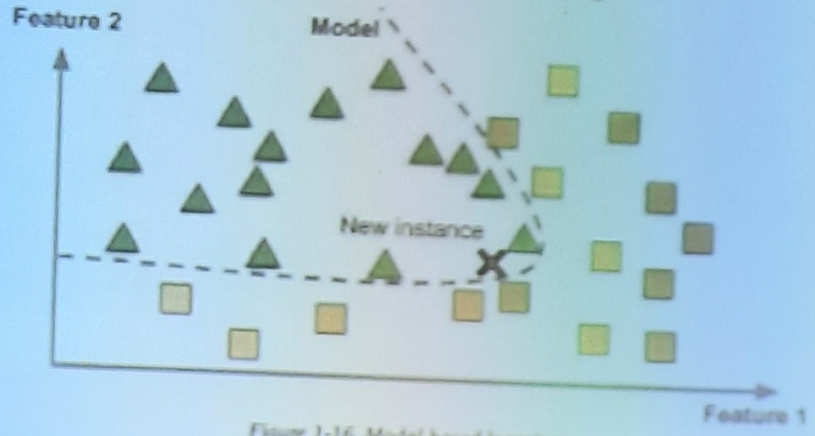


Figure 1-16. Model-based learning