KIG4068 : Machine Learning

Week 1: Introduction to Machine Learning

Semester 2, Session 2023/2024

What is Machine Learning?

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

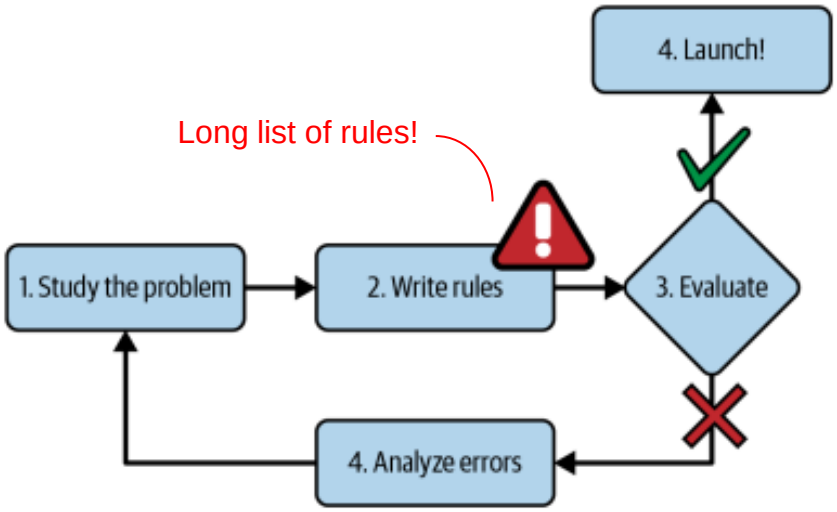
Arthur Samuel, 1959

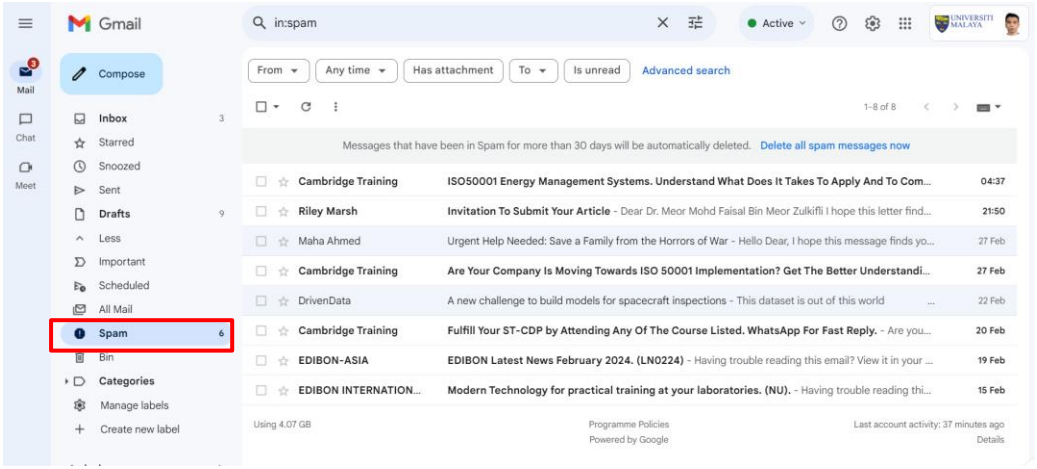
A computer program is said to learn from experience *E* with respect to some task *T* and some performance measure *P*, if its performance on *T*, as measured by *P*, improves with experience *E.*

Tom Mitchell, 1997

Why Use Machine Learning?

Example: A spam filter using traditional programming techniques

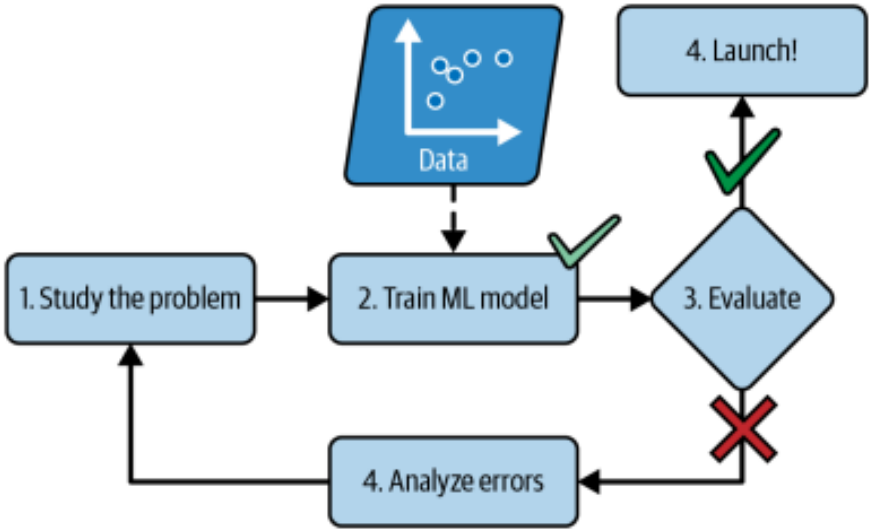




1. Examine the spam patterns – e.g., the subject line, sender's name, emails body etc.
2. Write a detection algorithm (rules) based on the pattern and flag emails as spam if a number of these patterns were detected.
3. Test and repeat steps 1. and 2.

Why Use Machine Learning?

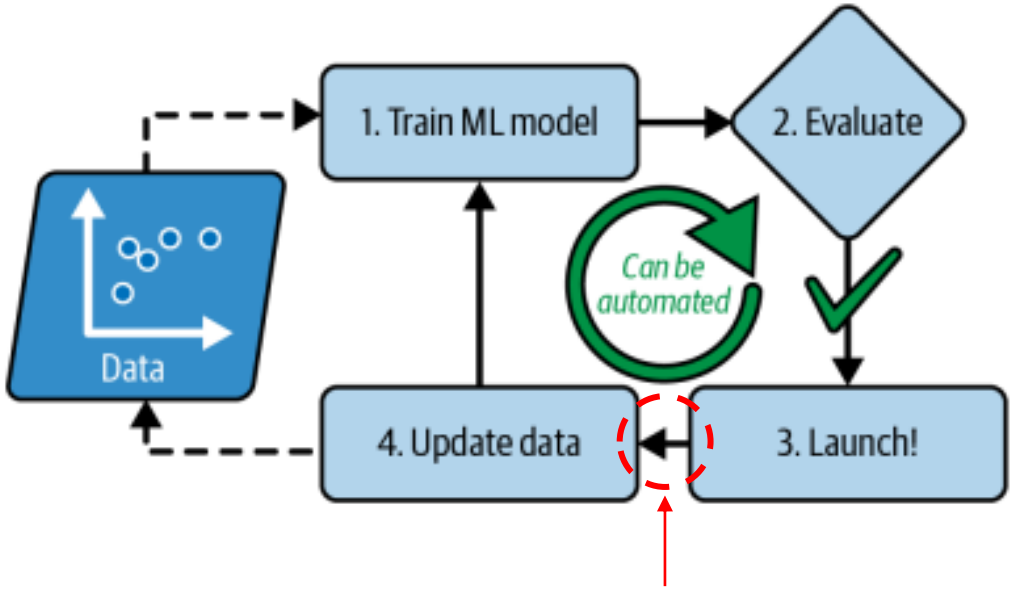
Example: A spam filter using machine learning approach



1. Examine the spam patterns – e.g., the subject line, sender's name, emails body etc.
2. Automatically learns which words and phrases are good predictors of spam based on the input data.
3. Test and repeat steps 1. and 2.

Why Use Machine Learning?

What if spammers use different words/ approaches?



Spam emails flagged by the user, and the ML model can be automatically re-trained

Why Use Machine Learning?

To summarize, machine learning is great for:

- Problems for which existing solutions require a lot of **fine-tuning or long lists** of **rules** (e.g. email spam detection)
- Complex problems for which using traditional approach yields **no good solution** (e.g. autonomous driving)
- **Fluctuating environment** - a machine learning system can easily be retrained on new data, **keeping it up to date** (e.g. stock price prediction)
- Getting insights about complex problems and **large** amounts of **data** (e.g. customer churn prediction)

Example of ML Applications

Manufacturing

- Predictive maintenance
- Quality control
- Supply chain optimization

Energy

- Grid optimization
- Predictive maintenance
- Renewable energy forecasting

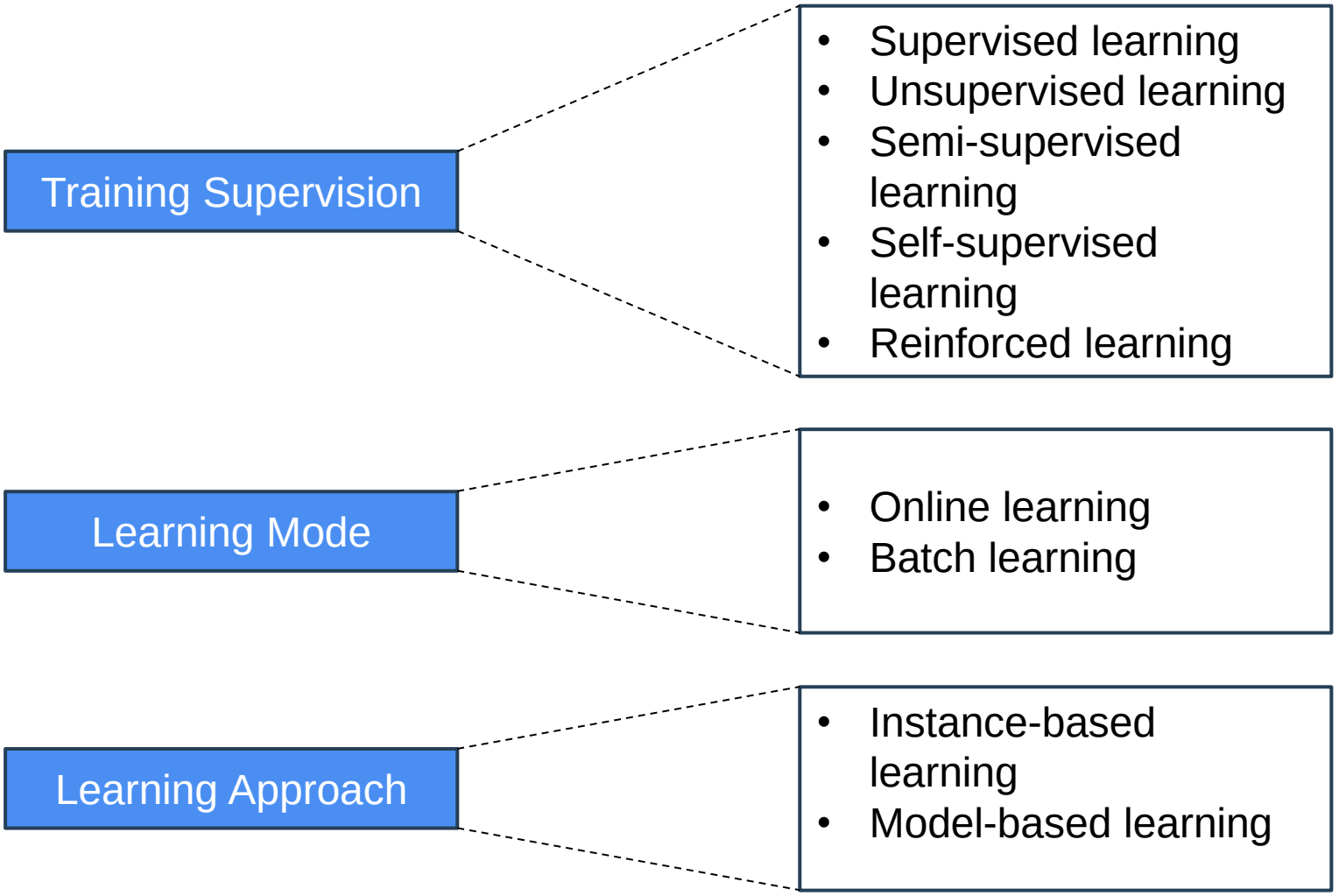
Banking

- Fraud detection
- Credit risk assessment
- Customer service automation

Healthcare

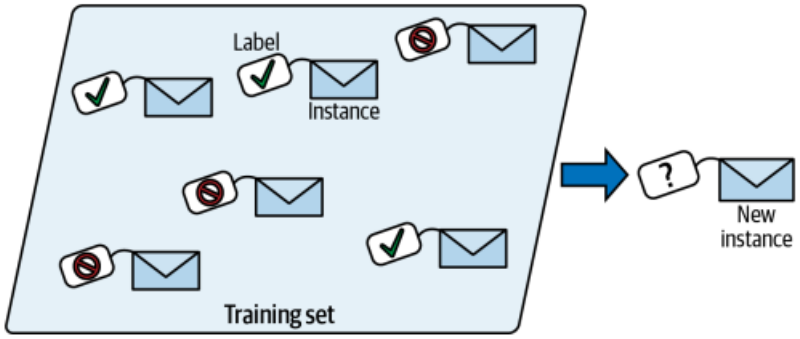
- Medical image analysis
- Drug discovery
- Personalized medicine

Types of Machine Learning Systems



Supervised Learning

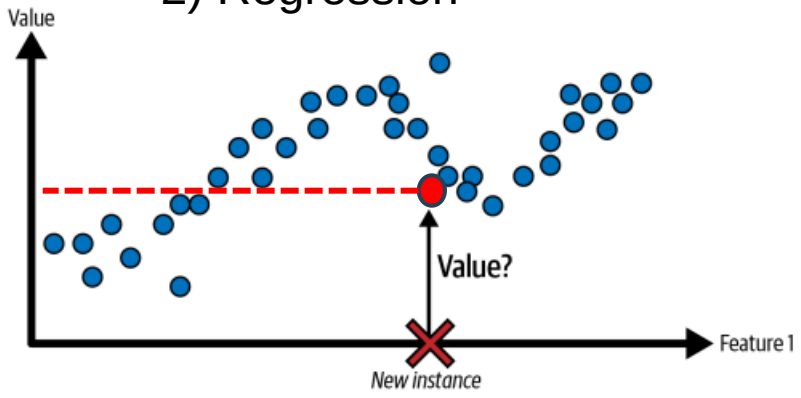
1) Classification



The model is trained with many e.g. (emails) each label with a ***class*** (spam/ not spam), and it must learn how to classify new emails to predict their ***labels***.

2) Regression

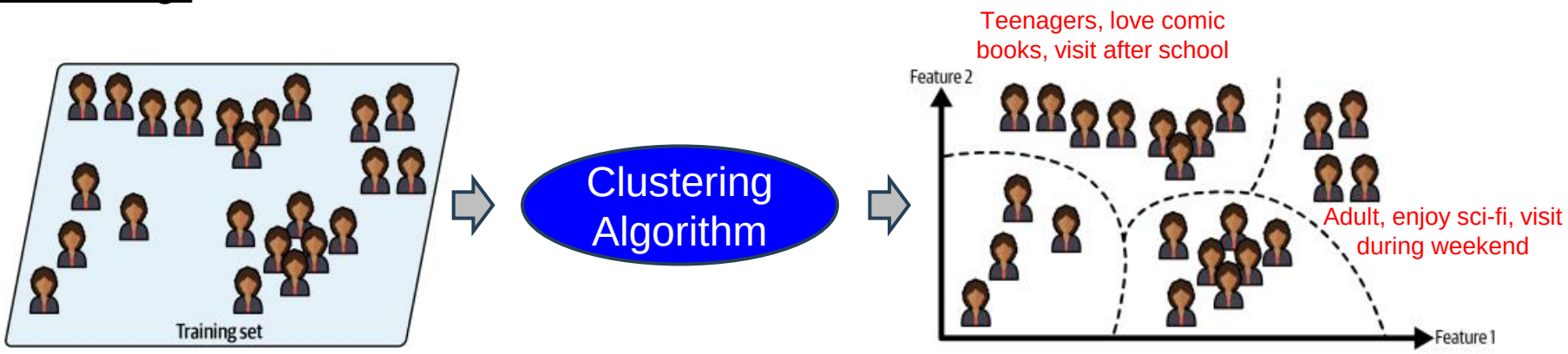
The model is trained with many e.g, (cars), each described by a set of ***features*** (milage, age, brand etc.) and associated with a ***target*** numeric value (e.g., car price). It learns how to predict a continuous numeric value, such as the price of a car.



Unsupervised Learning

- **No predefined labels or target output** provided to the algorithm
- Primary objective is to **find hidden patterns or intrinsic structures** within the data

Clustering

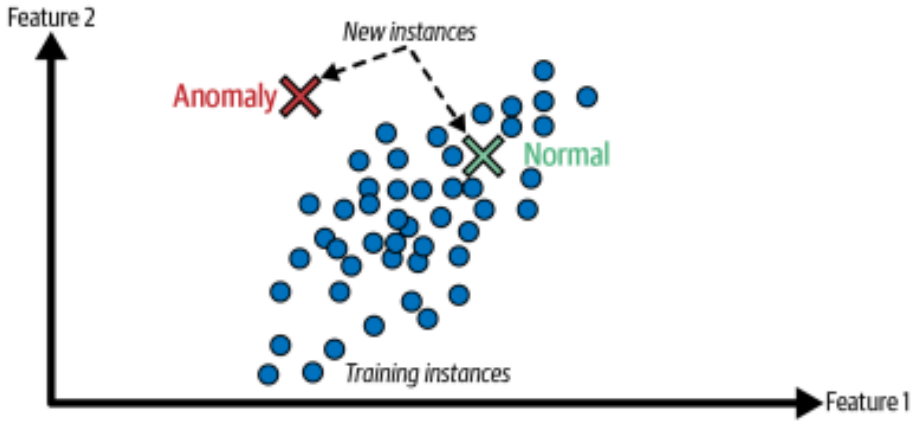


e.g. data of a blog's visitors (age, visiting time, country, interest etc)

Unsupervised Learning

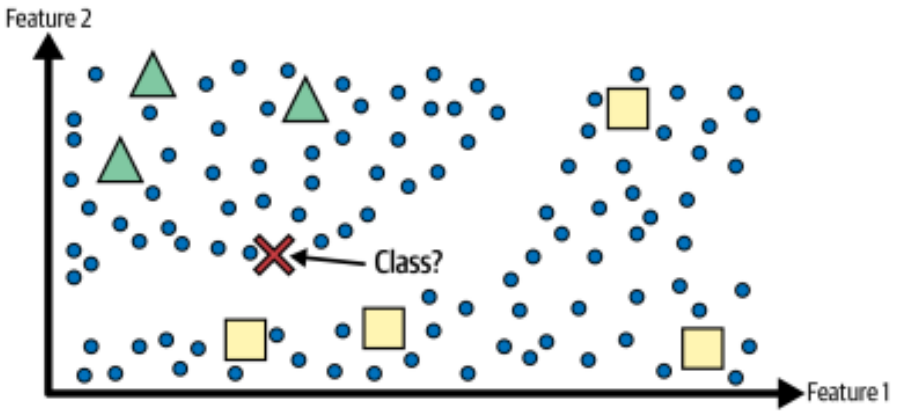
Anomaly detection

- The goal is to **detect deviations from the norm or unexpected patterns** that may indicate potential anomalies or anomalies in the data



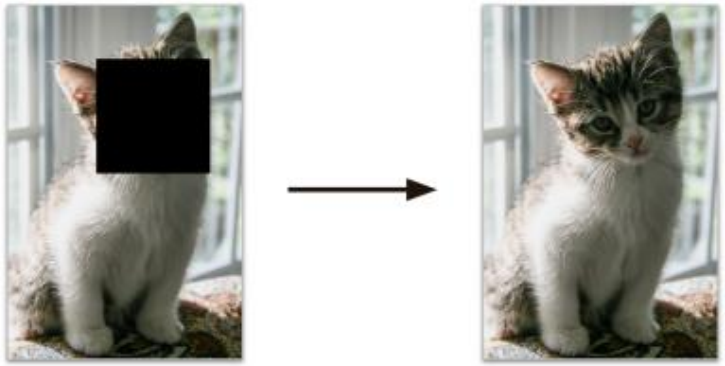
Semi-Supervised Learning

- Utilizes **a small amount of labeled data** combined with a larger **pool of unlabeled data**
- Leverage the additional unlabeled data to improve model's performance where obtaining data is costly or time-consuming
- Application examples: text and speech recognition, image classification etc.



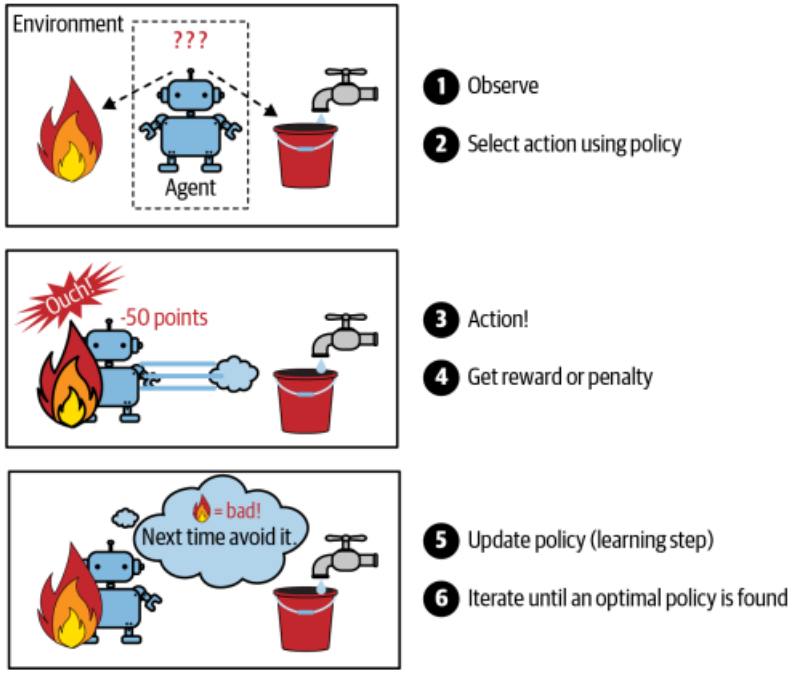
Self-Supervised Learning

- The model learns from the inherent structure of the input data, without requiring externally provided labels
- Generates labels from the input data using predefined tasks
- The goal is to leverage the intrinsic relationship within the data to learn meaningful representations, which can then be used for downstream tasks
- Application examples: autoencoding, predicting missing parts of data etc.



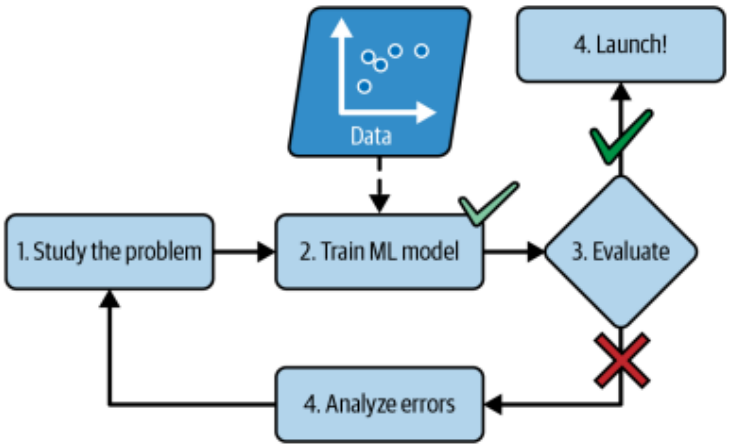
Reinforcement Learning

- The learning system called an ***agent*** can observe the environment, select and performed actions, and get rewards (or penalties) in return.
- Learn by itself what is the best strategy, called a ***policy***, to get most reward over time. A policy defines what action the agent should choose when it is in each situation.

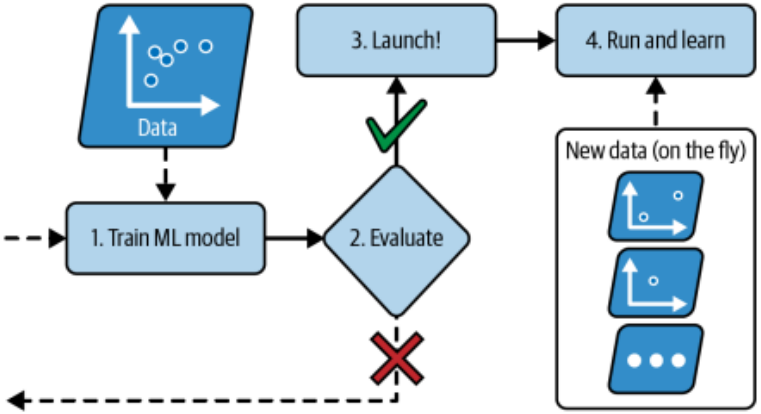


Batch (offline) vs Online Learning

Batch Learning



Online Learning



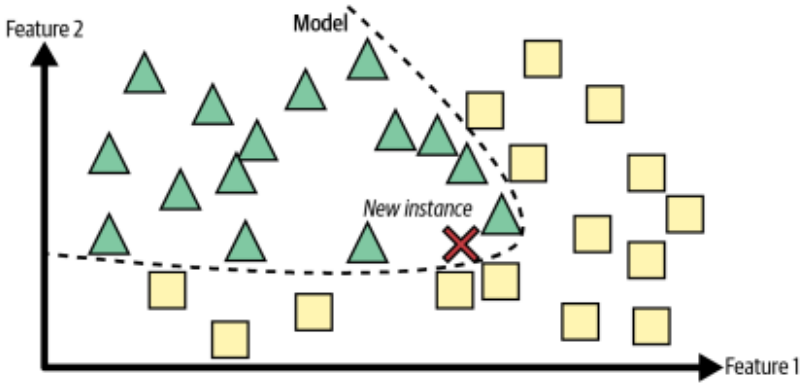
- In online learning, the ML system is trained incrementally by feeding it data instances sequentially, either individually or in small group called *mini-batches*
- Useful for system that need to adapt to change extremely rapidly (e.g. stock market trading)/ for system that has limited computing resources

Instance-based Learning

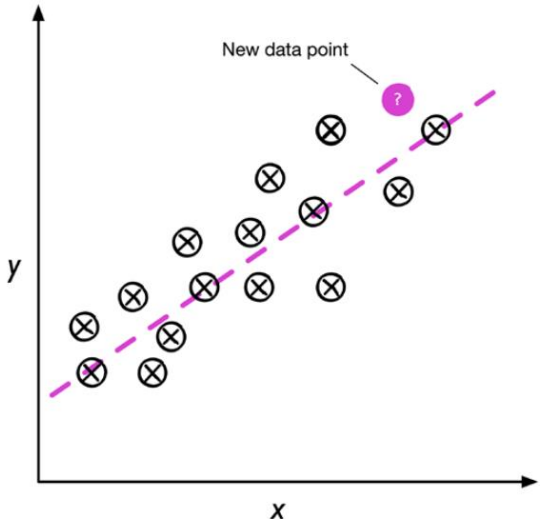
- The system learns the examples by heart, then generalizes to new cases by using a similarity measure to compare to the learned examples
- e.g. The new instance would be classified as a triangle because the majority instances belong to that class.

Model-based Learning

- A model is built from a set of examples (training data), and the trained model is used to make predictions for new instances.
- Example of model-based classification and regression:



Classification

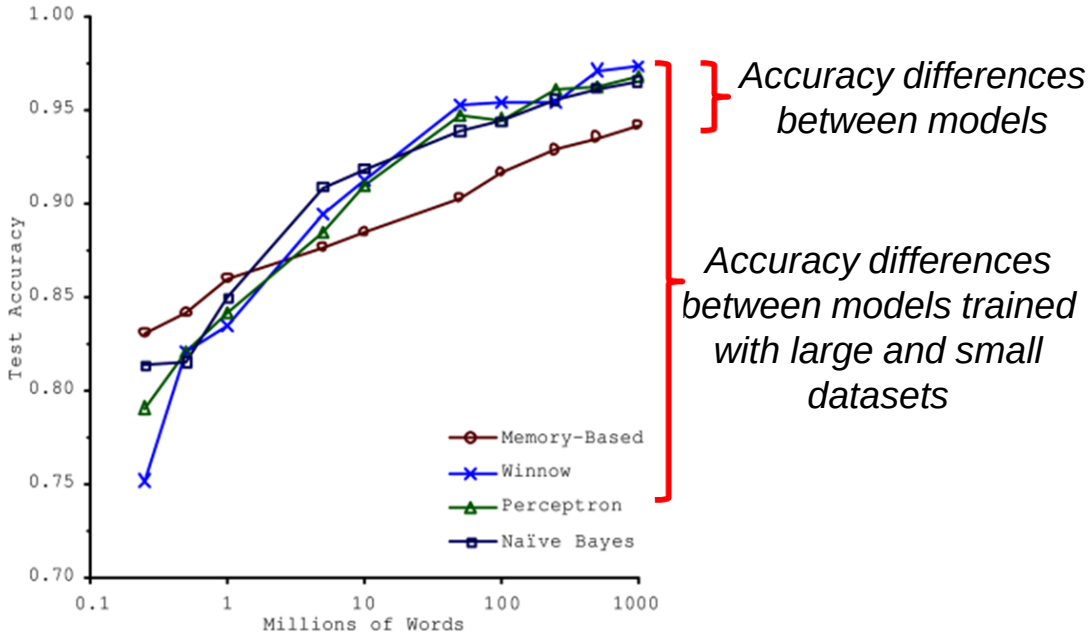


Regression

Main Challenges of ML

Insufficient Quantity of Training Data

- The more data for training the better!
- It can take a lot of data for most ML algorithms to work.
- “Simple” problems often require >10k of samples.
- “Complex” problems often require > 1m of samples.



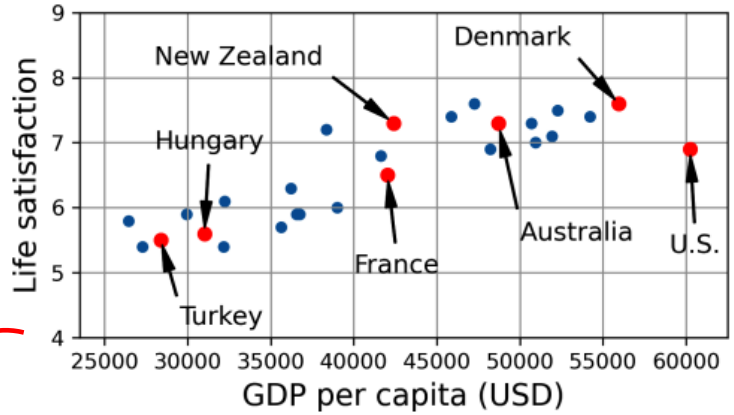
The importance of data vs algorithms

Main Challenges of ML

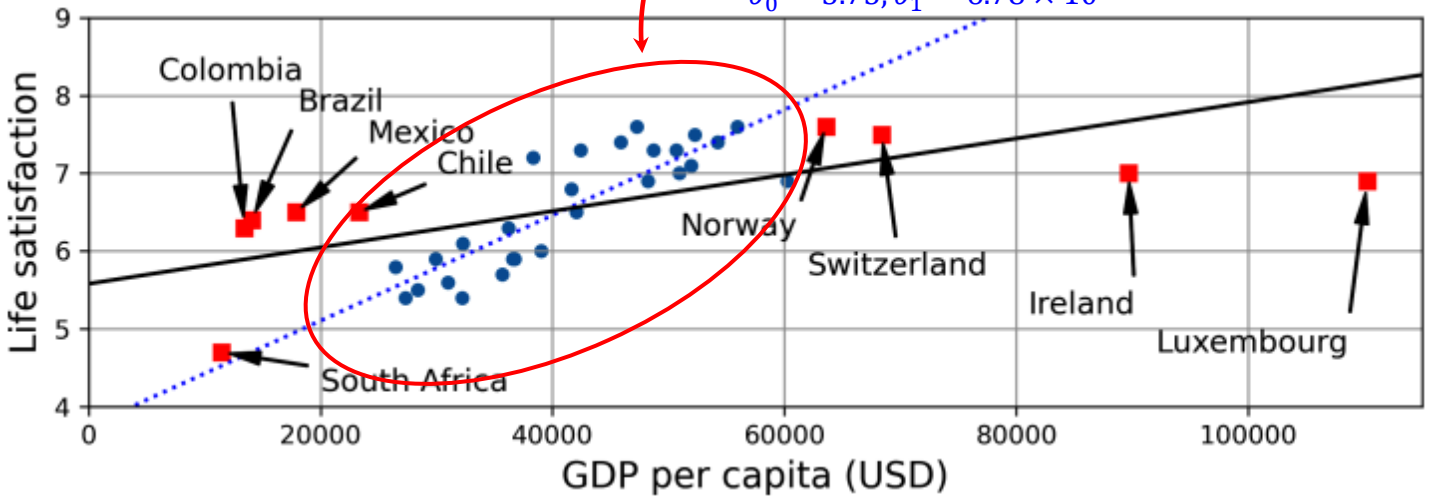
Non-representative training data

using simple linear model:

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



$\theta_0 = 3.75, \theta_1 = 6.78 \times 10^{-5}$



Main Challenges of ML

Poor quality training data

- Data can be full of errors, outliers, and noise (e.g., due to poor-quality measurements).
- Dirty data => hard for any algorithm to detect patterns.
- Significant amount of time will be spent on understanding the data and data cleaning:
  1. Data types? Numerical features? Ordinal features? Categorical features?
  2. Look for outliers in the data: Remove? Fix manually?
  3. Look for missing data: Remove? Impute values?

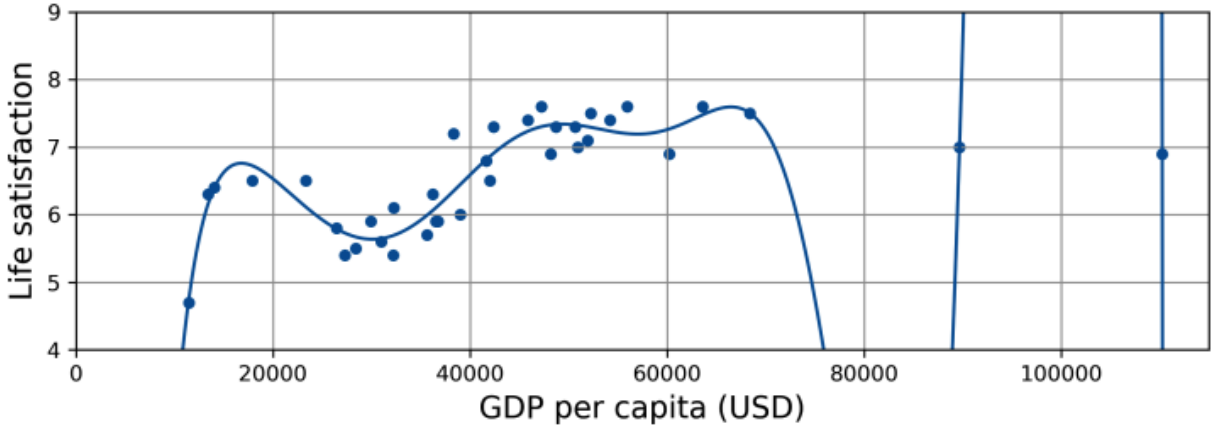
Main Challenges of ML

Overfitting the training data

What is *overfitting*?

- Overfitting is when model performs well on training data but poorly on new data.
- If model is complex or training data is limited, model will detect spurious patterns.
- Constraining a complex model to make it simpler is called *regularization.*

**Regularization – technique used to prevent overfitting in machine learning model*



Main Challenges of ML

Underfitting the training data

What is *underfitting*?

- Underfitting is when a model is too simple to learn the underlying structure of the data.
- Linear models will often underfit (but often a good place to start).

How to reduce underfitting?

- Select mode complex (more parameters) model.
- Feed better features to the model (*feature engineering*).
- Reduce the constrains on model (reduce *regularization*).

Validation and Testing

Generalization error

Why measure generalization error?

- Only way to know if your model is good is to measure performance on new data
- Split the dataset into *train* and *test* sets: error on the test set is an estimation of *generalization error*.
- Low training error, high generalization error => overfitting!

Some train-test split heuristics:

- For datasets smaller than 100k samples, take 80% for train and holdout 20% for test.
- For larger datasets (1m samples), holdout 1-10% of the dataset for test.

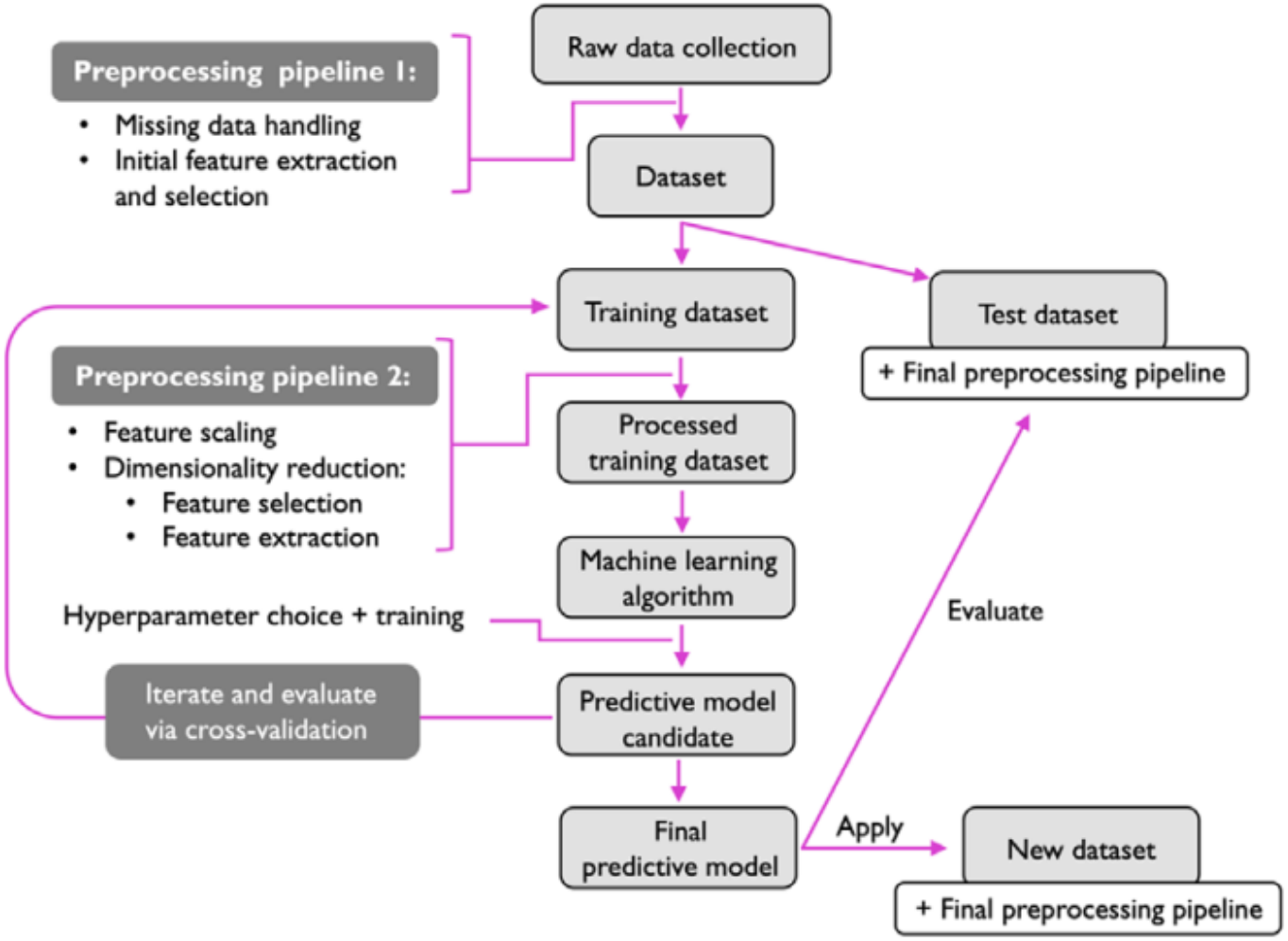
**Generalization – ability of a machine learning model to perform well on unseen data*

Validation and Testing

Model Selection

- Often need to tune *hyperparameters* to find a good model within a particular class of models.
- Split the training dataset into training set and validation set.
- Always compare tuned models using the test set.
- Validation set too small => might select “bad” model by mistake.
- Validation set too large => training set too small!
- Cross validation: Create lots of small validation sets, evaluate model on each validation set, measure average performance across validation sets

A Roadmap for Building ML Systems



Machine Learning Platforms

- Statistics and Machine Learning Toolbox
- Deep Learning Toolbox
- Computer Vision Toolbox
- Curve Fitting Toolbox
- Optimisation Toolbox

Installing Python

1. Download Anaconda:
   - Go to the Anaconda website: Anaconda.com
   - Click on "Download" and choose the appropriate version for your operating system (Windows, macOS, or Linux).
   - Follow the prompts to download the installer.

2. Install Anaconda:
   - Once the installer is downloaded, double-click it to start the installation process.
   - Follow the installation wizard instructions.
   - Select the installation directory (the default directory is usually fine).
   - Choose whether to add Anaconda to your system PATH (recommended for easier command-line access).
   - Complete the installation process by clicking "Install" or "Next" as per the prompts.

3. Launch Anaconda Command Line:
   - After installation, you can launch Anaconda Command Line from the Start menu (Windows) or from the Applications folder (macOS).
   - Alternatively, you can also launch Anaconda Prompt or Anaconda PowerShell Prompt from the Start menu (Windows) or Spotlight search (macOS).

4. Create a Python Environment:
   - In Anaconda Command Line (Anaconda Prompt or Anaconda PowerShell Prompt), type the following command to create a new environment:

     conda create --name myenv

     Replace "myenv" with the desired environment name.

Installing Python

5. After creating the environment, activate it using the command:
   conda activate myenv
   Replace "myenv" with the name of your environment.

6. Install pip:
   conda install pip

7. With the environment activated, install the required Python packages using pip. In Anaconda Command Line, type the following commands one by one:
   pip install numpy
   pip install scipy
   pip install scikit-learn
   pip install matplotlib
   pip install pandas
   pip install spyder
   pip install jupyter

8. Launch spyder or jupyter to start/ continue your coding using either of these commands:
   spyder
   jupyter notebook