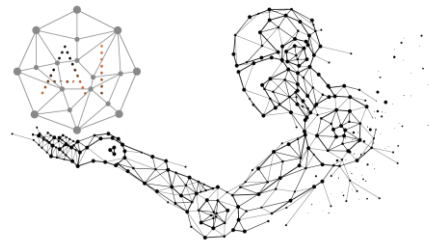
Applied Machine Learning

Chapter 4- Introduction to Machine Learning

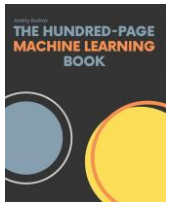
دانشگاه تربیت مدرس
Tarbiat Modares University

Hossein Homaei
Department of Electrical & Computer Engineering

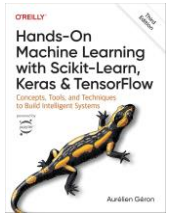


Some resources

- Books



A. Burkov, The Hundred-Page Machine Learning Book, Andriy Burkov, 2019.



A. Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 3rd ed. O'Reilly Media, 2023.

- Online
  - Machine Learning Specialization- Coursera
    - Instructors: Andrew Ng, Geoff Ladwig, and Aarti Bagul
    - Stanford University and DeepLearning.AI

What is Machine Learning?

- Programmer view:
  - Machine learning is the "*field of study that gives computers the ability to learn without being explicitly programmed*." [Arthur Samuel, 1959]

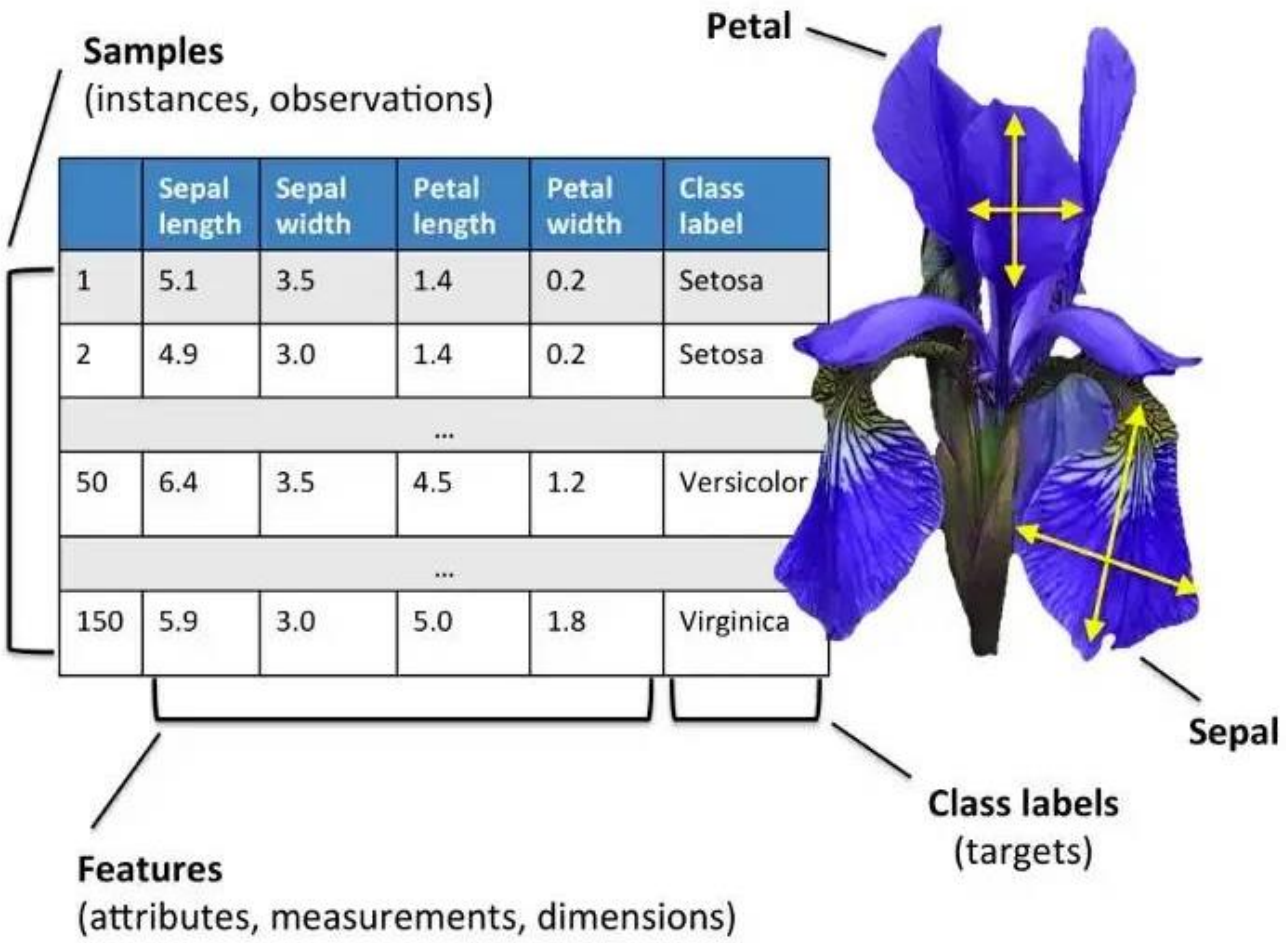
- More formal:
  - "*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.*" [Tom Mitchell, 1997]

Basic Terms

- **Training set**: The examples that the system uses to learn
- **Instance**, **Sample**, or **Observation**: Each training example
- **Feature**, **Attribute**, or **Dimension**: A measurable property of a data set
- **Model**: The part of a machine learning system that learns and makes predictions
  - Example: Neural networks, random forest,…
- **Target**, or **Label**: what the model is expected to **predict**
  - Target is more common in **Regression** tasks
  - Label is used in **Classification** tasks

...Basic Terms

- Iris Dataset



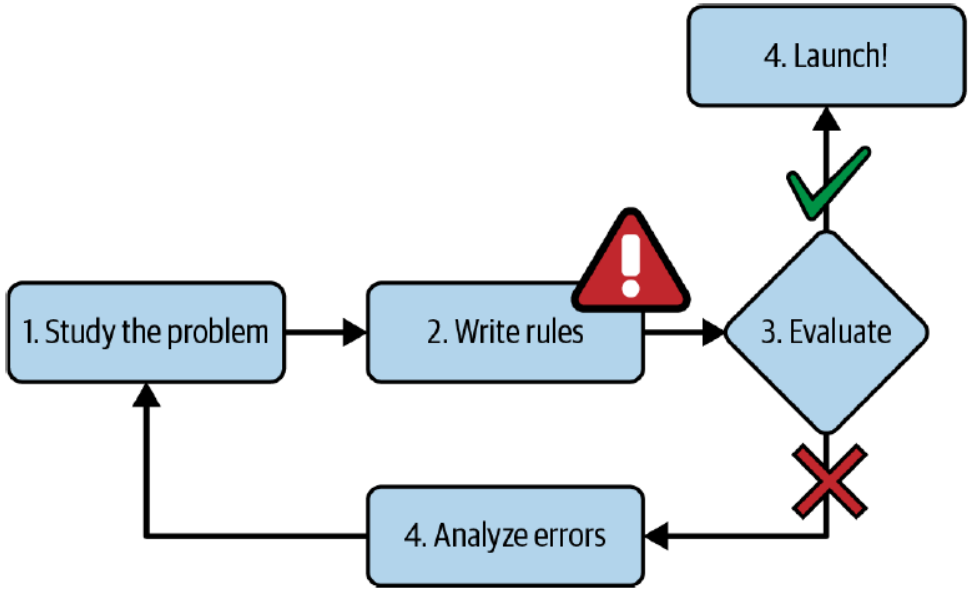
	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
			...		
50	6.4	3.5	4.5	1.2	Versicolor
			...		
150	5.9	3.0	5.0	1.8	Virginica

Example

- Spam filter is an ML program.
  - Given examples of spam emails (flagged by users) and examples of regular emails (also called “ham”), can learn to flag spam.
    - Task T = flag spam for new emails
    - Experience E = training data
    - Performance measure P = accuracy (the ratio of correctly classified emails)

ML vs traditional programming

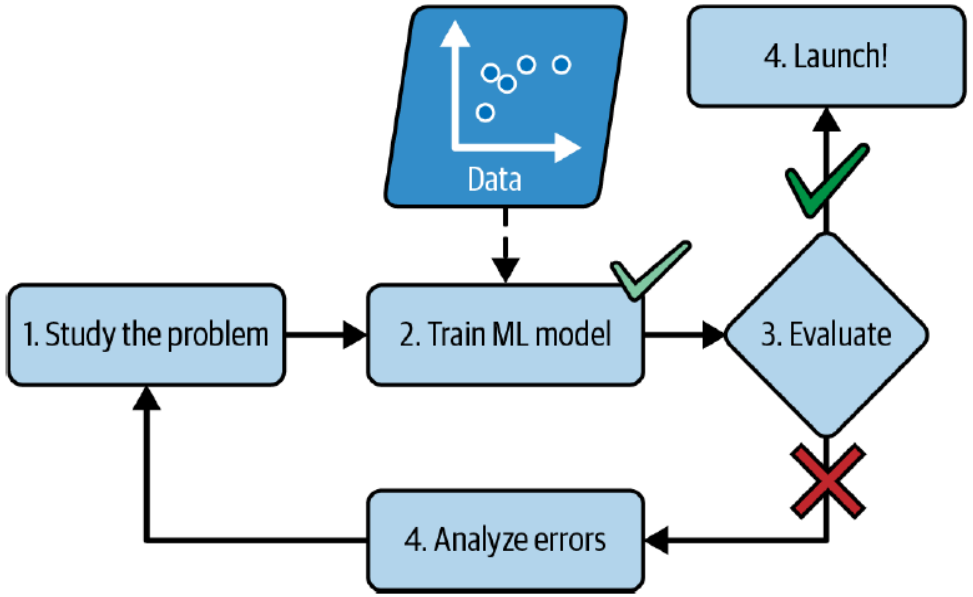
Spam filter- traditional program



Examine what spam typically looks like and extract some patterns.

- Long list of complex rules.
- Hard to maintain
- Error prone

Spam filter based on ML



Automatically learns which words and phrases are good predictors of spam

- Much shorter
- Easier to maintain
- More accurate

ML Candidate Problems

- There exists a pattern
  - But it is not a well-known mathematical pattern
- Data is available

Types of Machine Learning Systems

- Criterion 1: Based on the amount and type of supervision they get during training
  - Supervised
  - Unsupervised
  - Semi-supervised
  - Self-supervised
  - Reinforcement

- Criterion 2: whether or not the system can learn incrementally from a stream of incoming data
  - Batch
  - Online

- Criterion 3: Do they compare new data points to known ones or build a predictive model? How do they generalize?
  - Instance-based
  - Model-based

… Types of Machine Learning Systems

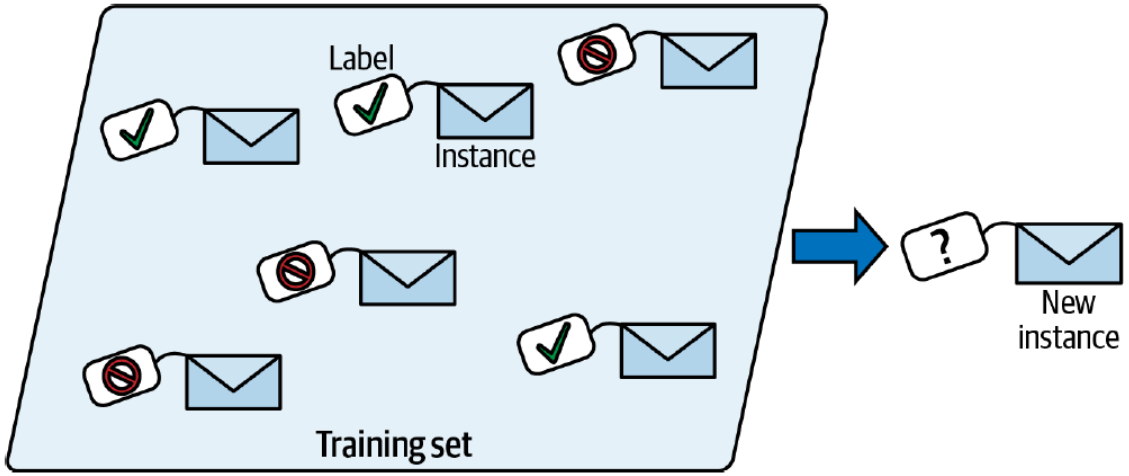
- The criteria are not exclusive and can be combined.

- Example: Spam filter that learn on the fly using a deep neural network model trained using human-provided examples of spam and ham.
  - Supervised
  - Online
  - Model-based

Supervised learning

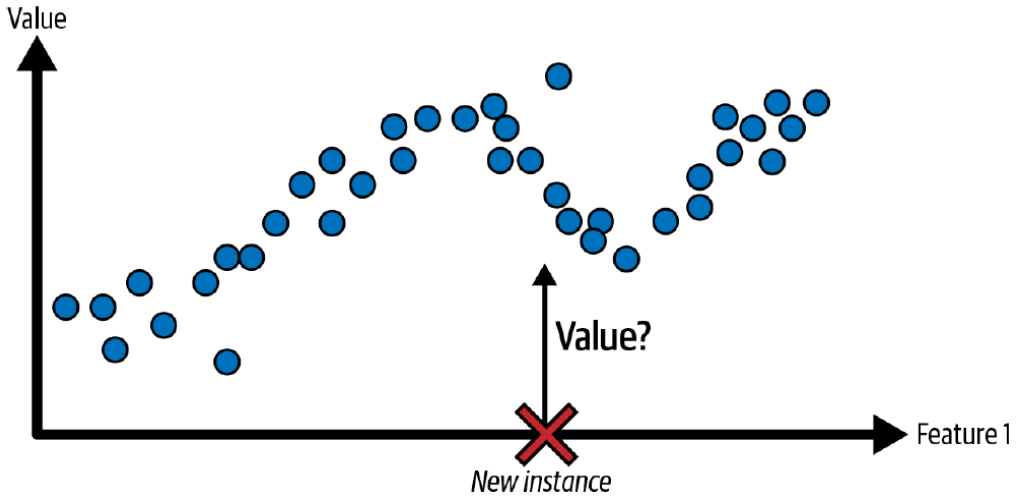
- Training set contains the desired solutions

- Typical tasks
  - **Classification**: assign a **label** to the new instance
    - Desired solutions = Discrete classes
    - Example: Spam filter



… Supervised learning

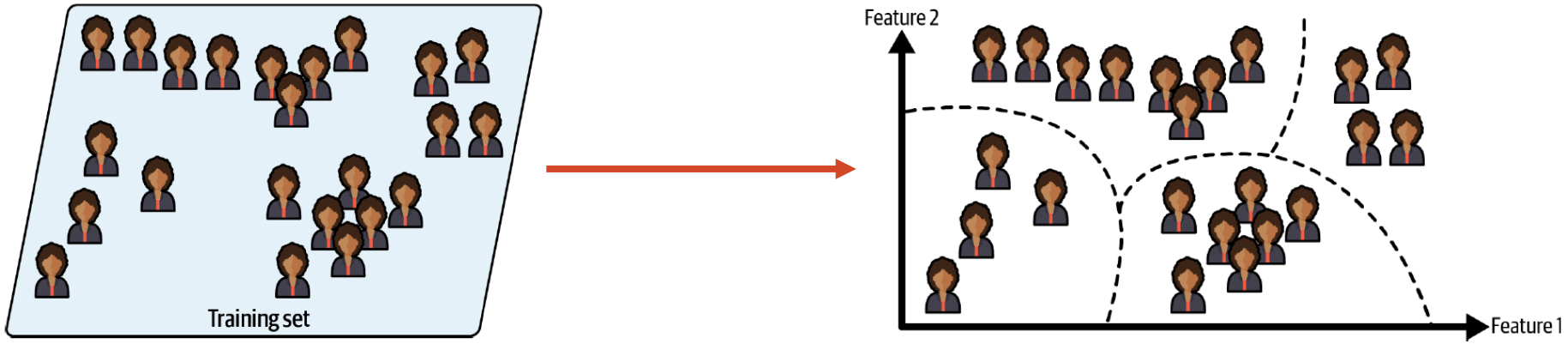
- … Typical tasks
  - **Regression**: predict a **target** numeric value
    - Desired solutions = Continuous values
    - Example: Predict the price of a car, given a set of **features** (mileage, age, brand, etc.).



Unsupervised learning

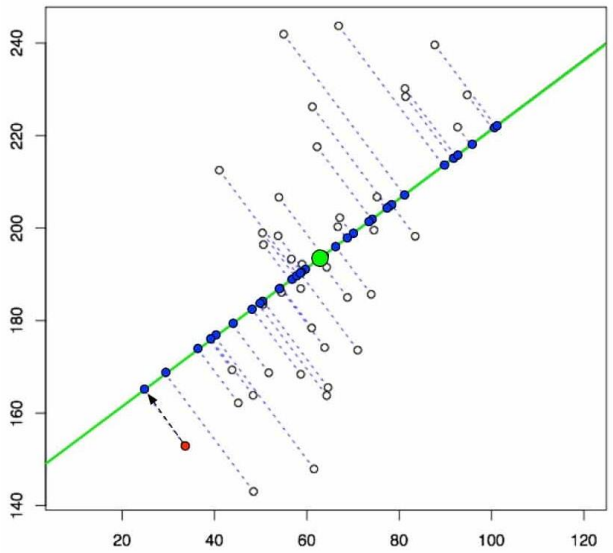
- Training data is unlabeled.
  - The system tries to learn without a teacher.

- Typical tasks
  - **Clustering**: try to detect groups of similar instances.
    - Example: detect groups of similar visitors to your website.



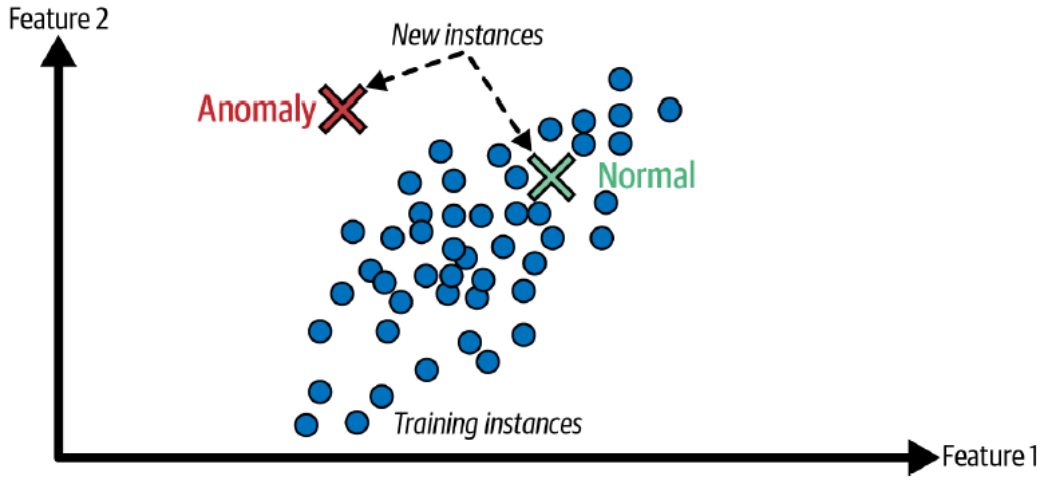
… Unsupervised learning

- … Typical tasks
  - **Dimensionality reduction**: simplify the data without losing too much information; For example by merging several correlated features into one.
    - Example: The car's mileage is correlated with its age ⇒ Merge them into one feature representing the car's wear and tear.



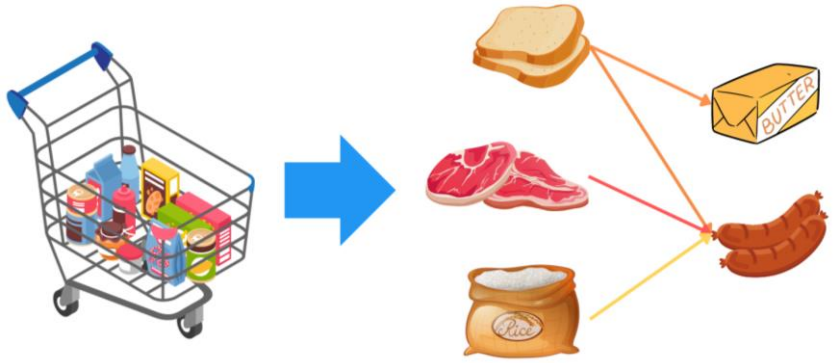
… Unsupervised learning

- … Typical tasks
  - **Anomaly detection**: The system learns to recognize normal instances. When it sees a new instance, it can tell whether it looks like a normal one or an anomaly.
    - Example 1: Fraud detection
    - Example 2: Outlier detection. Automatically removing outliers from a dataset before feeding it to another learning algorithm.



… Unsupervised learning

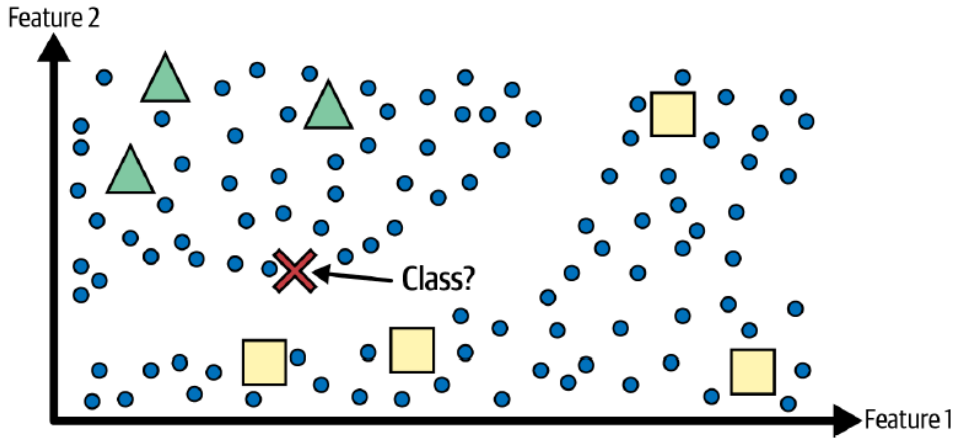
- … Typical tasks
  - **Association rule learning**: dig into large amounts of data and discover interesting relations between attributes.
    - Example: Find which items are purchased together in the supermarket.



Purchase item A => also purchase item B

Semi-supervised learning

- Dealing with data that's partially labeled.
  - Labeling data is usually time-consuming and costly.
  - We often have plenty of unlabeled instances, and few labeled instances.



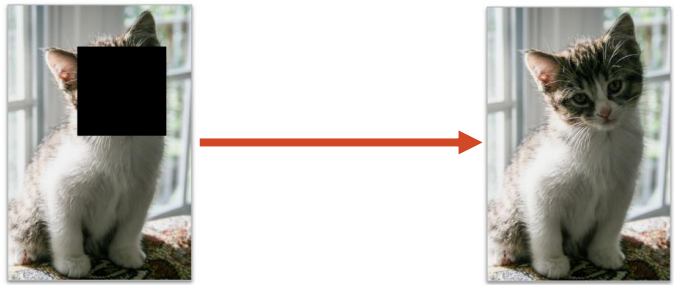
Semi-supervised learning with two classes (triangles and squares)
The unlabeled examples (circles) help classify a new instance into the triangle class rather than the square class, even though it is closer to the labeled squares

… Semi-supervised learning

- Most semi-supervised learning algorithms are combinations of unsupervised and supervised algorithms.
  - Example: Google Photos photo-hosting service.
    - You upload all your family photos to the service
    - It automatically recognizes the same persons in the photos (unsupervised clustering)
    - Add only one label per person (supervised part)
    - Now, the system can name everyone in every photo
      - Useful for searching photos

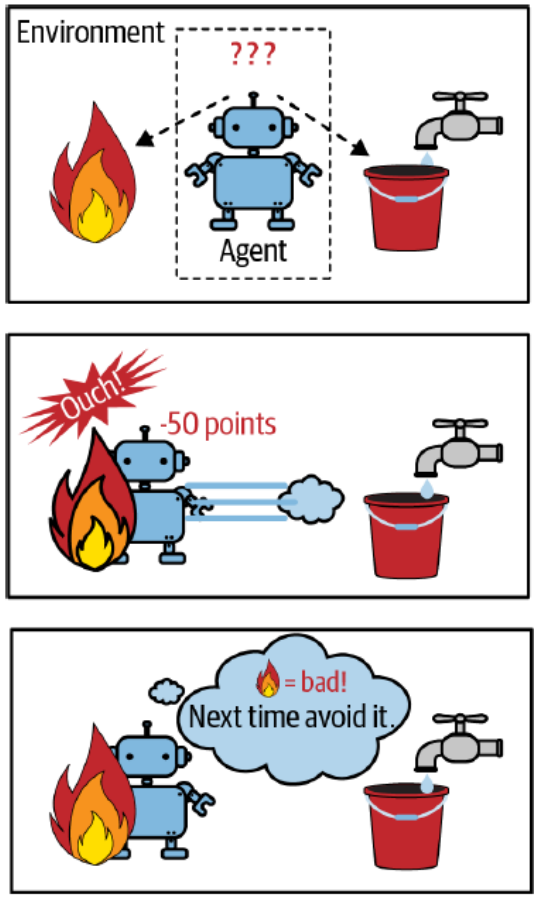
Self-supervised learning

- The model trains itself to learn one part of the input from another part of the input.
  - Example: repair images
    - A large dataset of unlabeled images is available.
    - Mask a small part of each image.
    - Train a model to recover the original image.
      - During training, the masked images are used as the inputs to the model, and the original images are used as the labels.
    - The resulting model can repair damaged images.



Reinforcement learning

- Learning the optimal behavior in an environment to obtain maximum reward.
- How?
  - An **agent** observes the environment,
  - selects and performs **actions**, and
  - get **rewards** in return (or **penalties**)
  - Learn **policy**
    - The best strategy to get the most reward over time
- Example: Learn to walk



Batch learning

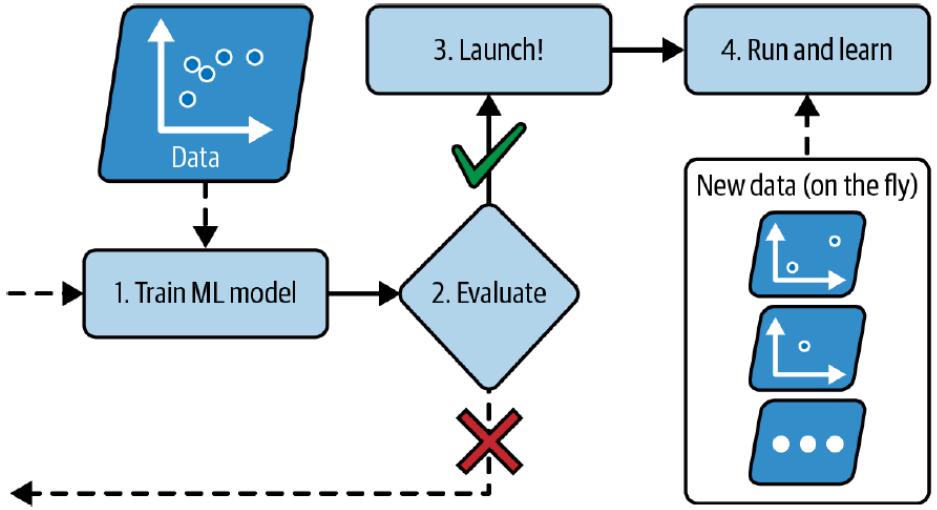
- First the system is trained, and then it is launched into production and runs without learning anymore.
  - Also called **offline learning**
  - Problem: the world continues to evolve while the model remains unchanged
    - This phenomenon is often called model rot or data drift.
  - Solution: Regularly retrain the model on up-to-date data.
    - How often do you need to update data?
      - It depends on the use case.
      - Example: cats and dogs classifier will decay very slowly, but predicting the financial market, is a fast-evolving system and hence will decay quite fast.

… Batch learning

- Some notes
  - Note 1:
    - 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.
  - Note 2:
    - You need to train a new version of the system from scratch on the full dataset
      - A lot of computing resources (CPU, memory space, disk space, disk I/O, network I/O, etc.)

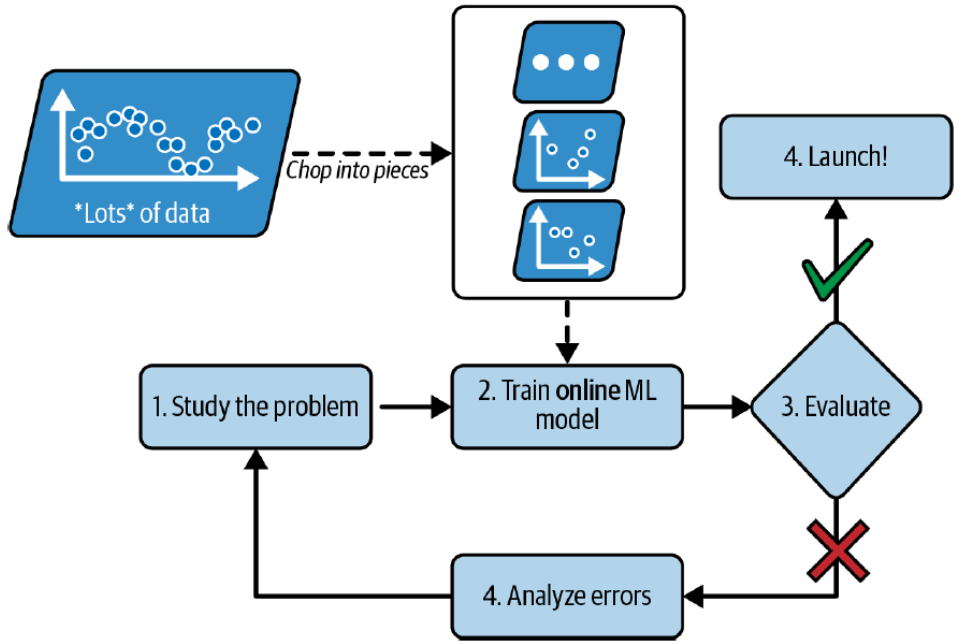
Online learning

- Train the system incrementally by feeding it data instances sequentially, either individually or in small groups called mini-batches.
- Useful cases:
  - Systems that need to adapt to change extremely rapidly (e.g., to detect new patterns in the stock market).
  - Only limited computing resources are available.



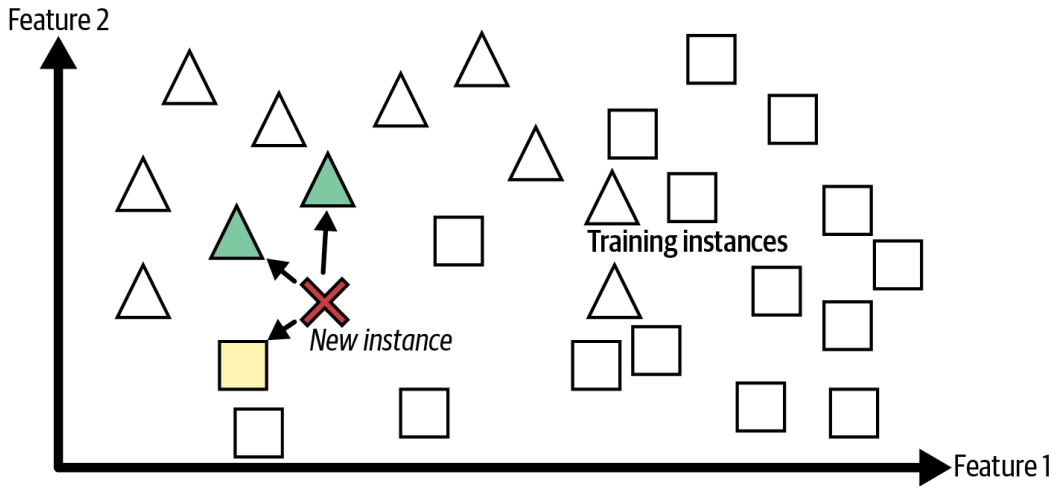
… Online learning

- … Useful cases:
  - Train models on huge datasets that cannot fit in one machine's main memory.
    - This is called out-of-core learning
    - Note: Out-of-core learning is usually done offline (i.e., not on the live system), so online learning can be a confusing name.
      - Think of it as incremental learning.



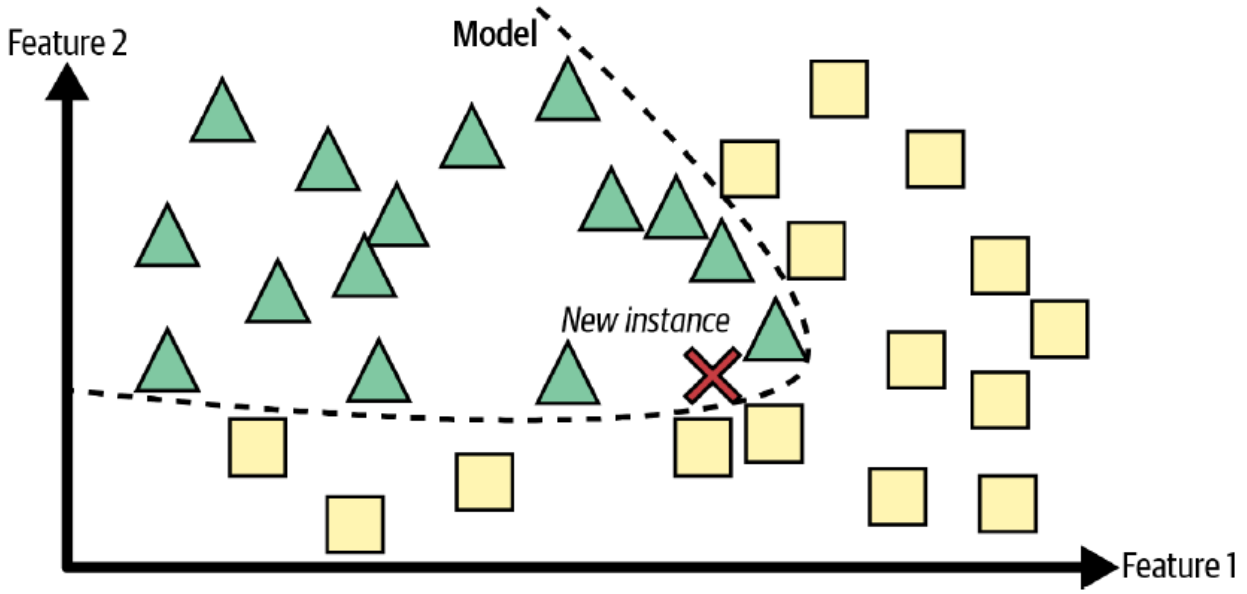
Instance-based learning

- Compare new instances with instances seen in training, which have been stored in memory.
  - Computation is postponed until a new instance is observed,
    - Known as "**Lazy**" models
- The system is generalized using a similarity measure to compare new instances to the learned ones.



Model-based learning

- Build a model of training samples and then use that model to make predictions.
- The model generalizes from the training data.
  - Example: linear regression



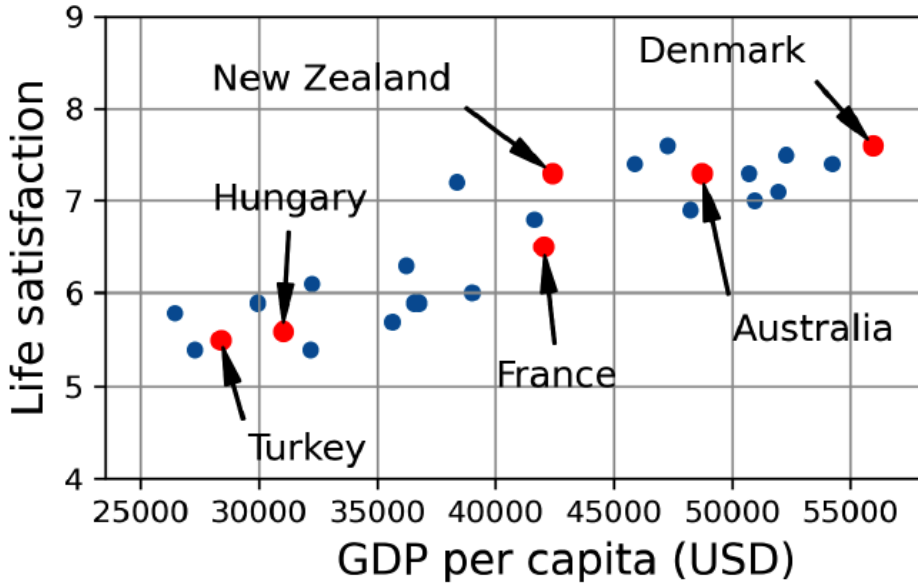
A typical machine learning workflow- Example

- Problem description: We want to know if money makes people happy.
- Solution steps
  - Data preparation
    - Download the Better Life Index data from the OECD's website.
    - Download the World Bank stats about Gross Domestic Product (GDP) per capita.
    - Join the tables and sort by GDP per capita.

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

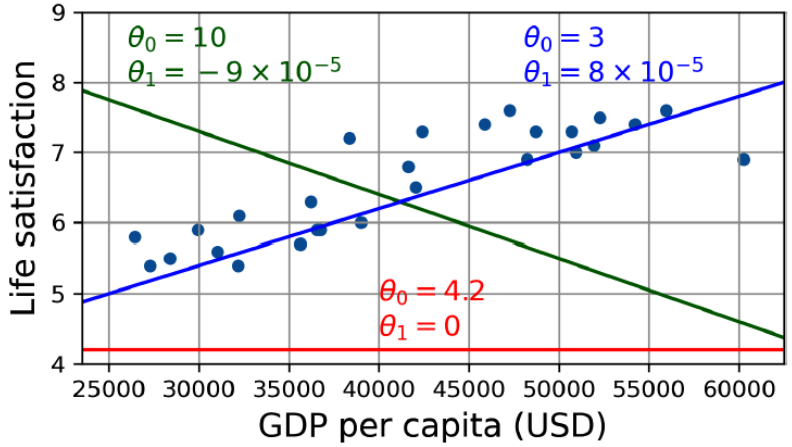
… A typical machine learning workflow- Example

- … Solution steps
  - Study the data
    - Visualize the data to understand it better!
      - Although the data is **noisy** it looks like life satisfaction goes up more or less linearly as the country's GDP per capita increases.



… A typical machine learning workflow- Example

- … Solution steps
  - … Study the data
    - **Model selection** (choosing the type of model): You decide to model life satisfaction as a linear function of GDP per capita.
      - You selected a *linear model* with just one attribute:
        - $\text{life}-\text{satisfaction} = \theta_0 + \theta_1 \times \text{GDP}-\text{per}-\text{capita}$
      - This model has two parameters: θ_0 and θ_1.
        - By tweaking these parameters, you can make your model represent any linear function

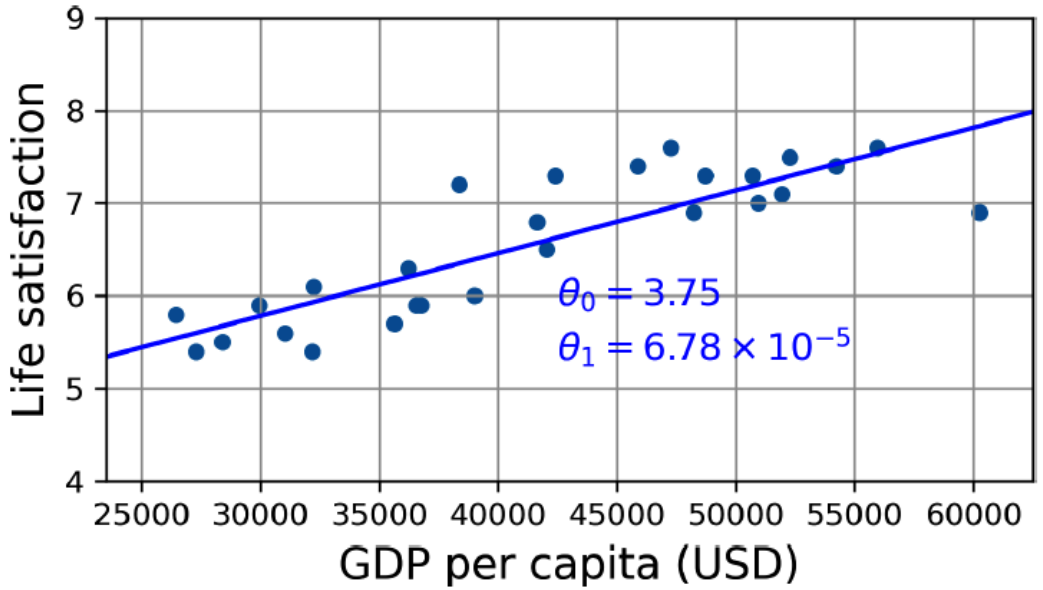


… A typical machine learning workflow- Example

- … Solution steps
  - Train the model: Assign the best values to the model parameters, θ_0 and θ_1.
    - Specify a performance measure and find a model that optimizes the metric
      - Define a **utility function** (or **fitness function**): measures how good your model is
      - Define a **cost function**: measures how bad your model is
    - linear regression problems → Cost function = the distance between the linear model's predictions and the training examples.
      - Objective = Minimize the distance
    - Feed the linear regression algorithm your training samples, and it finds the parameters that make the linear model fit best to your data.
      - This is called **training** the model.
    - $\theta_0 = 3.75$ and $\theta_1 = 6.78 \times 10^{-5}$

… A typical machine learning workflow- Example

- … Solution steps
  - Inference: Run the model to make predictions.
    - Example: We know that Cyprus's GDP per capita is $37,655. How happy Cypriots are?
      - $\text{Cyprus}-\text{life}-\text{satisfaction} = 3.75 + 37655 \times 6.78 \times 10^{-5} = 6.30$

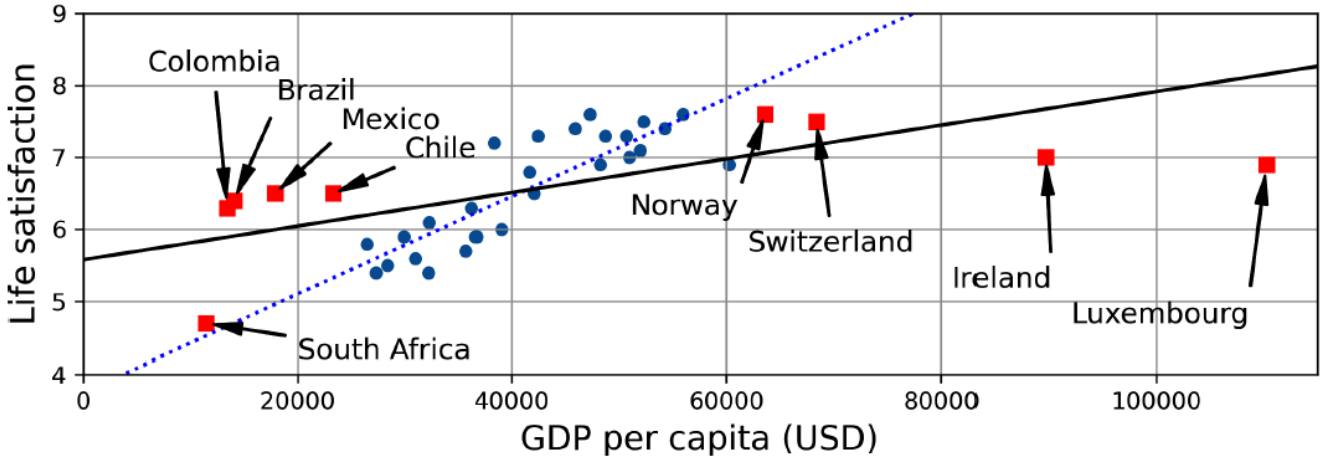


Main Challenges of ML

- Insufficient Quantity of Training Data
  - It takes lots of data for most ML algorithms to work properly.
  - Example: Image or speech recognition needs millions of examples.
- Poor-Quality Data
  - The training data is full of errors, outliers, and noise (e.g., due to poor quality measurements)
  - What should we do? Clean up the training data.
    - Discard outliers manually if it is possible.
    - If some instances are missing a few features
      - Ignore the features
      - Ignore the instances
      - Fill in the missing values (e.g. with median or mean,…)

… Main Challenges of ML

- Nonrepresentative Training Data
  - To generalize well, it is crucial that your training data be representative of the new cases you want to generalize to.
  - Example: In the previous example, the dataset we used for training is not representative; it did not contain any country with a GDP per capita lower than $23,500 or higher than $55,938.
    - Completely, different answer!

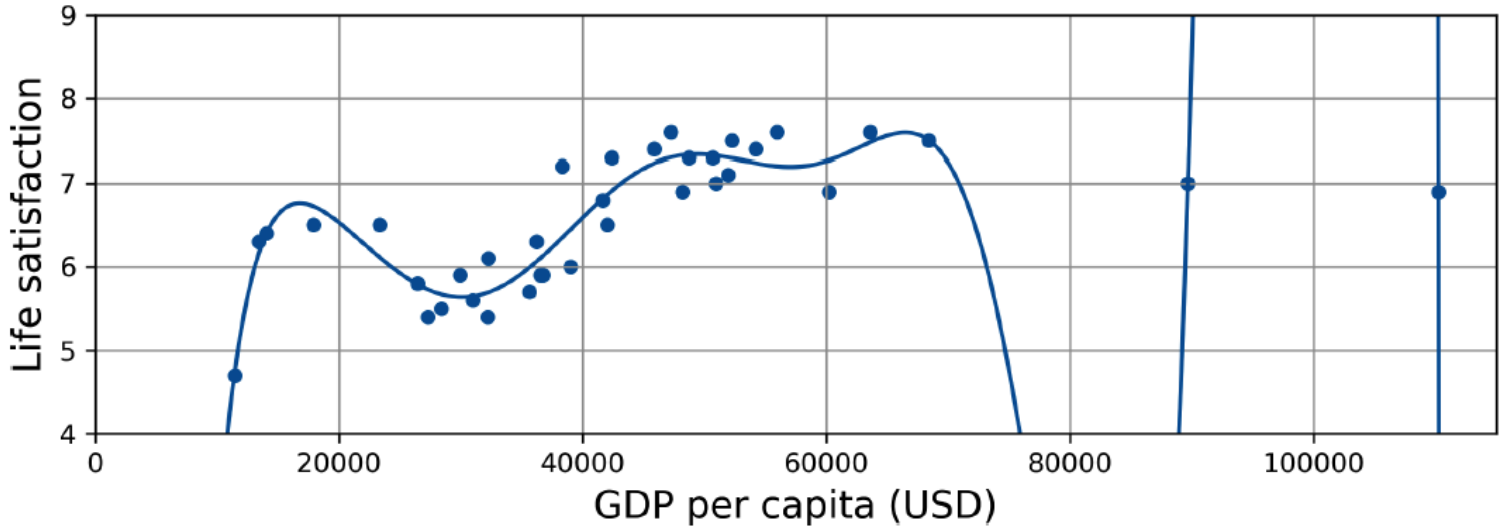


… Main Challenges of ML

- Irrelevant Features
  - The training data should contain enough relevant features and not too many irrelevant ones.
  - **Feature engineering**:
    - **Feature selection**: selecting the most useful features to train on among existing ones
    - **Feature extraction**: combining existing features to produce a more useful one
    - Creating new features by gathering new data

… Main Challenges of ML

- Overfitting the Training Data
  - The model performs well on the training data, but it does not generalize well.
  - If the training set is noisy, the model is likely to detect patterns in the noise itself
  - Why happens?
    - The model is too complex relative to the amount and noisiness of the training data.

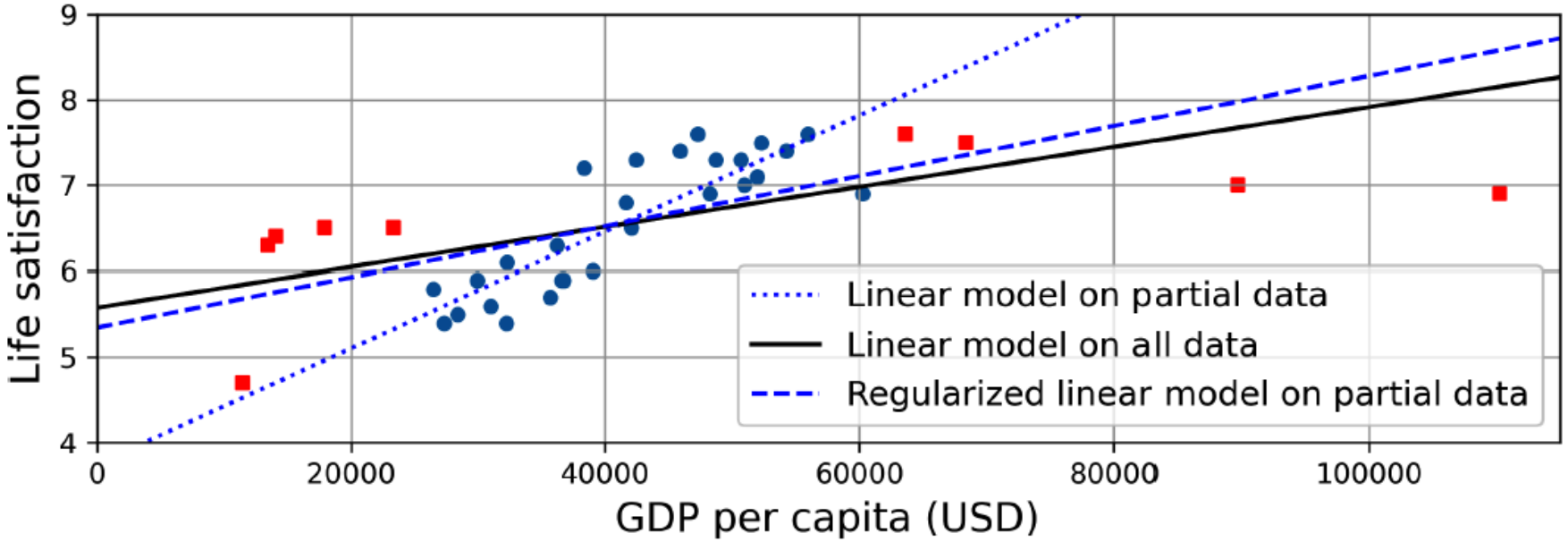


… Main Challenges of ML

- … Overfitting
  - Solutions
    - Gather more training data
    - Reduce the noise in the training data
    - Simplify the model by
      - Selecting one with fewer parameters (e.g., a linear model rather than a high-degree polynomial model)
      - Reducing the number of attributes in the training data
      - Constraining the model
        - Called **regularization**
        - Example: A linear algorithm has two degrees of freedom to adapt to the training data: we can tweak both θ_0 and θ_1 of the line. Regularization may force θ_1 to be equal to a specific value or restrict θ_1 between two specific values ($a \leq \theta_1 \leq b$)

… Main Challenges of ML

- … Overfitting



… Main Challenges of ML

- Underfitting the Training Data
  - The model is too simple to learn the underlying structure of the data
  - Example: The linear model of life satisfaction is prone to underfit
  - Solutions
    - Select a more powerful model, with more parameters.
    - Feed better features to the learning algorithm (feature engineering).
    - Reduce the constraints on the model

Bias-Variance Tradeoff

All Data

ID	F_1	F_2	F_3	…	F_P
1	v_{1-1}	v_{1-2}	v_{1-3}		v_{1-P}
2	v_{2-1}	v_{2-2}	v_{2-3}		v_{2-P}
⋮	⋮	⋮	⋮	⋱	⋮
N	v_{N-1}	v_{N-2}	v_{N-3}		v_{N-P}

Sample-1

ID	F_1	F_2	F_3	…	F_P
S_{1_1}	$v_{S_{1_1}-1}$	$v_{S_{1_1}-2}$	$v_{S_{1_1}-3}$		$v_{S_{1_1}-P}$
S_{1_2}	$v_{S_{1_2}-1}$	$v_{S_{1_2}-2}$	$v_{S_{1_2}-3}$		$v_{S_{1_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{1_n}	$v_{S_{1_n}-1}$	$v_{S_{1_n}-2}$	$v_{S_{1_n}-3}$		$v_{S_{1_n}-P}$

Sample-2

ID	F_1	F_2	F_3	…	F_P
S_{2_1}	$v_{S_{2_1}-1}$	$v_{S_{2_1}-2}$	$v_{S_{2_1}-3}$		$v_{S_{2_1}-P}$
S_{2_2}	$v_{S_{2_2}-1}$	$v_{S_{2_2}-2}$	$v_{S_{2_2}-3}$		$v_{S_{2_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{2_m}	$v_{S_{2_m}-1}$	$v_{S_{2_m}-2}$	$v_{S_{2_m}-3}$		$v_{S_{2_m}-P}$

…

Sample-K

ID	F_1	F_2	F_3	…	F_P
S_{K_1}	$v_{S_{K_1}-1}$	$v_{S_{K_1}-2}$	$v_{S_{K_1}-3}$		$v_{S_{K_1}-P}$
S_{K_2}	$v_{S_{K_2}-1}$	$v_{S_{K_2}-2}$	$v_{S_{K_2}-3}$		$v_{S_{K_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{K_q}	$v_{S_{K_q}-1}$	$v_{S_{K_q}-2}$	$v_{S_{K_q}-3}$		$v_{S_{K_m}-P}$

...Bias-Variance Tradeoff

Sample-1

ID	F_1	F_2	F_3	...	F_P
S_{1_1}	$v_{S_{1_1}-1}$	$v_{S_{1_1}-2}$	$v_{S_{1_1}-3}$		$v_{S_{1_1}-P}$
S_{1_2}	$v_{S_{1_2}-1}$	$v_{S_{1_2}-2}$	$v_{S_{1_2}-3}$		$v_{S_{1_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{1_n}	$v_{S_{1_n}-1}$	$v_{S_{1_n}-2}$	$v_{S_{1_n}-3}$		$v_{S_{1_n}-P}$

Sample-2

ID	F_1	F_2	F_3	...	F_P
S_{2_1}	$v_{S_{2_1}-1}$	$v_{S_{2_1}-2}$	$v_{S_{2_1}-3}$		$v_{S_{2_1}-P}$
S_{2_2}	$v_{S_{2_2}-1}$	$v_{S_{2_2}-2}$	$v_{S_{2_2}-3}$		$v_{S_{2_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{2_m}	$v_{S_{2_m}-1}$	$v_{S_{2_m}-2}$	$v_{S_{2_m}-3}$		$v_{S_{2_m}-P}$

...

Sample-K

ID	F_1	F_2	F_3	...	F_P
S_{K_1}	$v_{S_{K_1}-1}$	$v_{S_{K_1}-2}$	$v_{S_{K_1}-3}$		$v_{S_{K_1}-P}$
S_{K_2}	$v_{S_{K_2}-1}$	$v_{S_{K_2}-2}$	$v_{S_{K_2}-3}$		$v_{S_{K_2}-P}$
⋮	⋮	⋮	⋮	⋱	⋮
S_{K_q}	$v_{S_{K_q}-1}$	$v_{S_{K_q}-2}$	$v_{S_{K_q}-3}$		$v_{S_{K_m}-P}$

...

Model-1 → $M_1(X)$

Model-2 → $M_2(X)$

Model-K → $M_K(X)$

New instance X

Bias-Variance Tradeoff talks about how similar are these models to each other and how strong these models are in making a predictions about an unknow new sample.

…Bias-Variance Tradeoff

- How to build the model?
  - Strategy-1: choose an overly simple model
    - Example: Decide only based on the mean of the sample values
      - Because each training set is completely randomized, the average of the values for each feature is (hopefully) very similar for each of the training sets.
      - All the models will be very similar to each other
        - If you ask about an unknown new sample, all the models are going to predict pretty much the same way

Underfit

Difference between predictions from different models

- The models have low variance → Good
  - We don't want the model to change a lot based on little changes in our training sets

Average distance between predictions and the truth

- The model has high bias → Bad
  - Because the model is very basic, it probably don't capture all the truth

…Bias-Variance Tradeoff

- …How to build the model?
  - Strategy-2: choose an overly complex model
    - Example: Incorporates all the features in a very complex way
      - Because each training set has its own specific patterns and noise, we cannot generalize the models that are trained based on each dataset to the general population.
      - Small change in training set leads to big change in the model
        - If you ask about an unknown new sample, each model is going to predict in a different way

- The models have high variance → Bad

- The model has low bias → Good
  - If we average over all of the models, we will get the right answer; but if we look at each model independently, it is going to be variable from truth and from each other

…Bias-Variance Tradeoff

- Summary
  - Bias measures how far the model's predictions are from the true underlying function
  - Variance measures how sensitive the model is to fluctuations in the training data
  - You cannot minimize both simultaneously
  - The goal is to find a balanced model that minimizes total error

Testing

- How well a model will generalize to new cases?
  - Method 1: Put your model in production!
  - Methods 2:
    - Split your data into two sets:
      - Training set
      - Test set
    - Evaluate your model on the test set

  - The error rate on new cases is called the **generalization error**

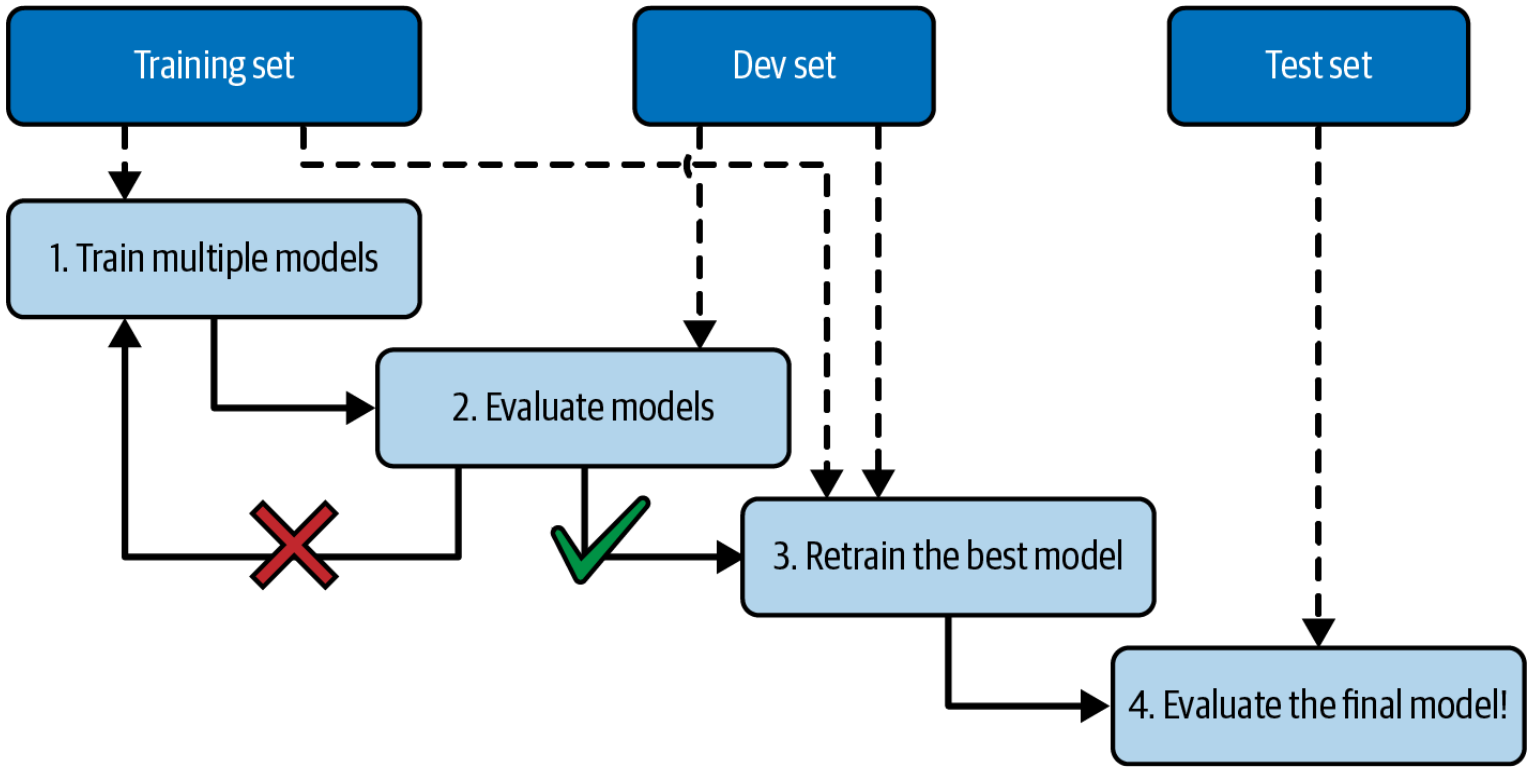
If the training error is low but the generalization error is high, it means that your model is overfitting the training data.

…Testing

- Common cases:
  - 80% for training
  - 20% for testing
- More general:
  - Depends on the size of the dataset
    - Example: If the dataset contains 10 million instances, then holding out 1% means your test set will contain 100,000 instances
      - Probably more than enough to get a good estimate of the generalization error

Hyperparameter Tuning and Model Selection

- **Holdout validation:** Hold out part of the training set for evaluation
  - This part is called the **validation set** or **development set** (or dev set).



… Hyperparameter Tuning and Model Selection

- Why do not use the test set?
  - You measure the generalization error multiple times on the test set, and adapt the model to produce the best results for that particular set. This means the model is unlikely to perform as well on new data.

… Hyperparameter Tuning and Model Selection

- Which model is best for the data?
  - We don't know.
  - No model is a priori guaranteed to work better
  - When you select a particular type of model, you are implicitly making assumptions about the data.
    - For example, if you choose a linear model, you are implicitly assuming that the data is fundamentally linear.
  - The only way to know the best model, is to evaluate them all.
    - It is not possible!
  - In practice, you make some reasonable assumptions about the data and evaluate only a few reasonable models.
    - Example: For simple tasks, use linear models. For complex problems, use neural networks.

Data Mismatch

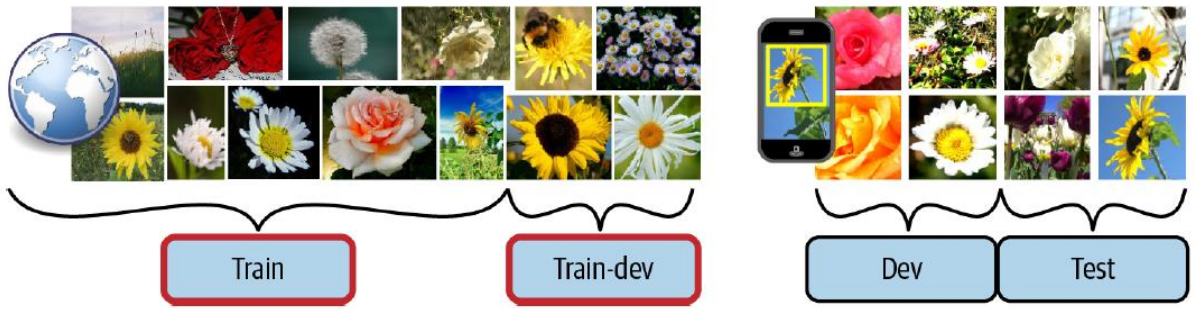
- Both the validation and test sets must be as representative as possible of the data you expect to use in production.
- In some cases, getting a large amount of data for training is easy, but this data won't perfectly represent the data used in production.
- Example:
  - Mobile app for taking flower pictures and detecting their species
  - There exist millions of pictures of flowers on the web
  - Web pictures won't be perfectly representative of the ones taken by the mobile

… Data Mismatch

- The problem
  - Assume that you train a model on the web pictures and get poor performance on the validation set.
  - So what? Does it happen due to the model overfitting problem, or the mismatch between the web and mobile app pictures?
- Solution:
  - Make a new set, **train-dev set**
  - Train the model on the training set
  - Evaluate it on the train-dev set
    - Prevent overfitting
  - Evaluate it on the dev set
    - Prevent data mismatch
  - Evaluate it on the test set
    - Know how well it performs in production

… Data Mismatch

- Example:
  - Make the train-dev set from the web
  - Train the model on the training set (from the web)
  - Evaluate the model on the train-dev set
    - If the model performs poorly, then it must have overfit the training set
      - Solve the problem using the methods previously explained.
  - Evaluate the model on the dev set
    - If it performs poorly, then the problem is coming from the data mismatch



… Data Mismatch

- How to tackle the problem?
  - Use representative data!
  - Example:
    - Make a preprocess on the web images to make them look more like the pictures taken by the mobile app, then retrain the model.

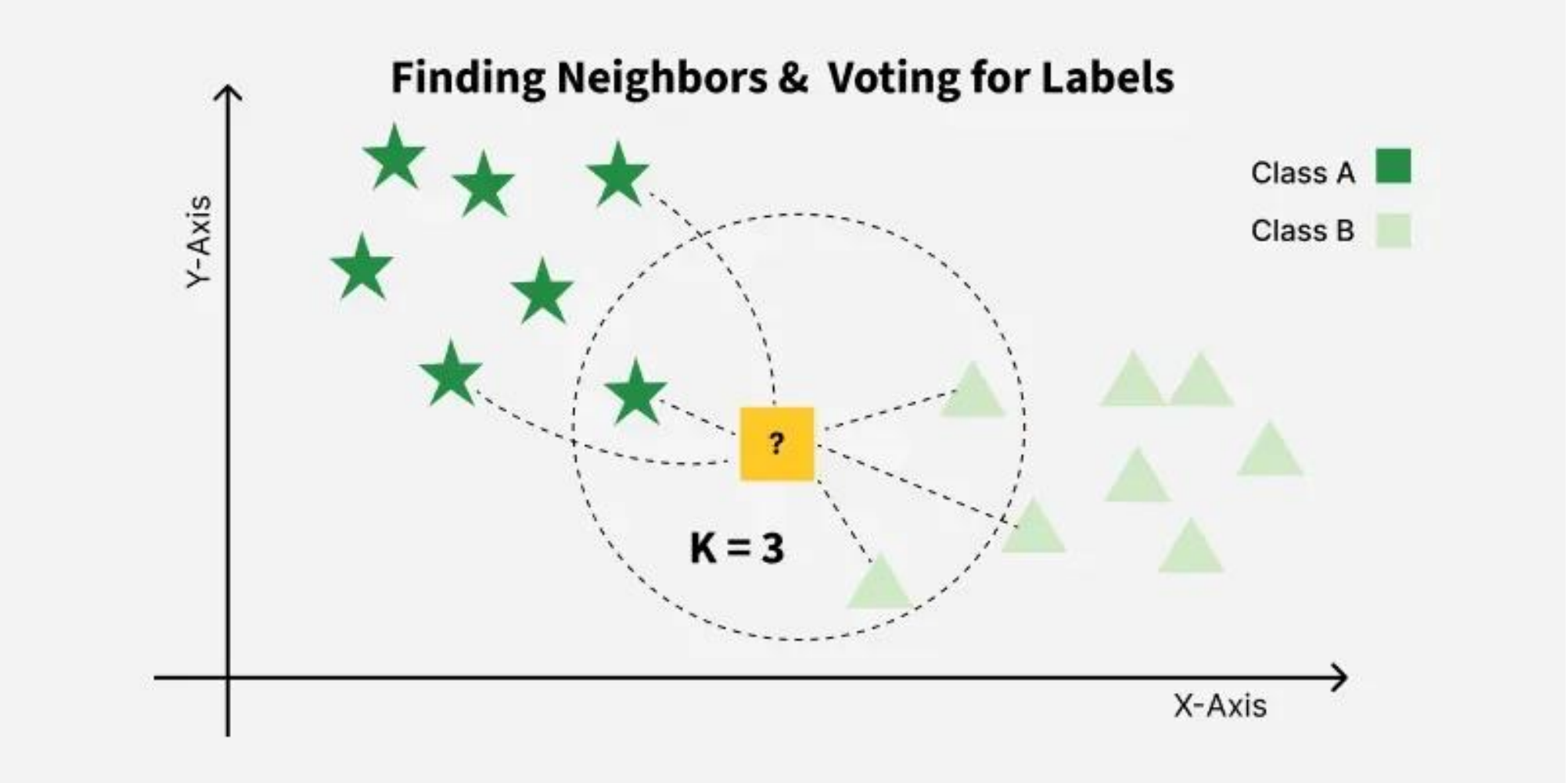
Classification with KNN

- What is it?
  - KNN = K-Nearest Neighbors
  - A supervised instance-based machine learning algorithm
  - One of the easiest to understand classification algorithms

…Classification with KNN

- How does it work?
  - We have a set of data points for which we know the correct class labels
  - When we get a new data point, we measure the distance between it and each existing point
  - Take the "k" closest data points (neighbors) to the given input
  - Make a prediction based on the majority class

...Classification with KNN



…Classification with KNN

- Hyperparameters (We should decide about them before running)
  - K = number of neighbors
    - Usually a number between 2 and 20
  - Distance metric to decide which point is closer to this point
    - Euclidean, Manhattan,…

…Classification with KNN

- Notes
  - No process for training
  - The decision can be too sensitive to the selected K hyperparameter.
    - Small K
      - The classifier memorizes training points
      - Predictions are very close to training data → Low bias
    - larger K
      - Predictions average over many neighbors.
      - The model is more stable → Low variance

KNN- Example-1

- Predict the diabetic patient given BMI and Age

ID	BMI	Age	Diabetic
1	33.6	50	Yes
2	26.6	30	No
3	23.4	40	No
4	43.1	67	No
5	35.3	23	No
6	35.9	67	Yes
7	36.7	45	Yes
8	25.7	46	No
9	23.3	29	No
10	31	56	Yes

…KNN- Example-1

- Does a 40-year-old person with a BMI of 43.6 have diabetes?
  - Using KNN:
    - Calculate the distance between the new instance and all the existing ones
    - Select the K nearest instances
    - Predict by voting
  - We should decide on the distance measurement mechanism and the K
    - Distance: Euclidean
    - K = 3

…KNN- Example-1

- Measuring the distance

ID	BMI	Age	Diabetic	Distance to (43.6, 40)
1	33.6	50	Yes	$\sqrt{(43.6-33.6)^2+(40-50)^2}=14.14$
2	26.6	30	No	$\sqrt{(43.6-26.6)^2+(40-30)^2}=19.72$
3	23.4	40	No	$\sqrt{(43.6-23.4)^2+(40-40)^2}=20.2$
4	43.1	67	No	$\sqrt{(43.6-43.1)^2+(40-67)^2}=27$
5	35.3	23	No	$\sqrt{(43.6-35.3)^2+(40-23)^2}=18.92$
6	35.9	67	Yes	$\sqrt{(43.6-35.9)^2+(40-67)^2}=28.08$
7	36.7	45	Yes	$\sqrt{(43.6-36.7)^2+(40-45)^2}=8.52$
8	25.7	46	No	$\sqrt{(43.6-25.7)^2+(40-46)^2}=18.88$
9	23.3	29	No	$\sqrt{(43.6-23.3)^2+(40-29)^2}=23.09$
10	31	56	Yes	$\sqrt{(43.6-31)^2+(40-56)^2}=20.37$

…KNN- Example-1

- Find the 3 nearest instances

ID	BMI	Age	Diabetic	Distance to (43.6, 40)	Rank
1	33.6	50	Yes	$\sqrt{(43.6-33.6)^2+(40-50)^2}=14.14$	2
2	26.6	30	No	$\sqrt{(43.6-26.6)^2+(40-30)^2}=19.72$	5
3	23.4	40	No	$\sqrt{(43.6-23.4)^2+(40-40)^2}=20.2$	6
4	43.1	67	No	$\sqrt{(43.6-43.1)^2+(40-67)^2}=27$	9
5	35.3	23	No	$\sqrt{(43.6-35.3)^2+(40-23)^2}=18.92$	4
6	35.9	67	Yes	$\sqrt{(43.6-35.9)^2+(40-67)^2}=28.08$	10
7	36.7	45	Yes	$\sqrt{(43.6-36.7)^2+(40-45)^2}=8.52$	1
8	25.7	46	No	$\sqrt{(43.6-25.7)^2+(40-46)^2}=18.88$	3
9	23.3	29	No	$\sqrt{(43.6-23.3)^2+(40-29)^2}=23.09$	8
10	31	56	Yes	$\sqrt{(43.6-31)^2+(40-56)^2}=20.37$	7

…KNN- Example-1

- Predict: 2 Yeses vs. 1 No → Yes

ID	BMI	Age	Diabetic	Distance to (43.6, 40)	Rank
1	33.6	50	Yes	$\sqrt{(43.6-33.6)^2+(40-50)^2}=14.14$	2
2	26.6	30	No	$\sqrt{(43.6-26.6)^2+(40-30)^2}=19.72$	5
3	23.4	40	No	$\sqrt{(43.6-23.4)^2+(40-40)^2}=20.2$	6
4	43.1	67	No	$\sqrt{(43.6-43.1)^2+(40-67)^2}=27$	9
5	35.3	23	No	$\sqrt{(43.6-35.3)^2+(40-23)^2}=18.92$	4
6	35.9	67	Yes	$\sqrt{(43.6-35.9)^2+(40-67)^2}=28.08$	10
7	36.7	45	Yes	$\sqrt{(43.6-36.7)^2+(40-45)^2}=8.52$	1
8	25.7	46	No	$\sqrt{(43.6-25.7)^2+(40-46)^2}=18.88$	3
9	23.3	29	No	$\sqrt{(43.6-23.3)^2+(40-29)^2}=23.09$	8
10	31	56	Yes	$\sqrt{(43.6-31)^2+(40-56)^2}=20.37$	7

KNN Algorithm

- Training Algorithm
  - For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

- Classification Algorithm
  - Given a query instance x_q to be classified,
    - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
    - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{K} \delta\big(v, f(x_i)\big)$$

Where

$V = \{v_1, v_2, \ldots, v_s\}$ is the finite set of all labels

$$\delta(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases}$$

…KNN- Example-1

- What happens if we choose K=4?
  - Votes are equal → The algorithm faces a tie

ID	BMI	Age	Diabetic	Distance to (43.6, 40)	Rank
1	33.6	50	Yes	$\sqrt{(43.6-33.6)^2+(40-50)^2}=14.14$	2
2	26.6	30	No	$\sqrt{(43.6-26.6)^2+(40-30)^2}=19.72$	5
3	23.4	40	No	$\sqrt{(43.6-23.4)^2+(40-40)^2}=20.2$	6
4	43.1	67	No	$\sqrt{(43.6-43.1)^2+(40-67)^2}=27$	9
5	35.3	23	No	$\sqrt{(43.6-35.3)^2+(40-23)^2}=18.92$	4
6	35.9	67	Yes	$\sqrt{(43.6-35.9)^2+(40-67)^2}=28.08$	10
7	36.7	45	Yes	$\sqrt{(43.6-36.7)^2+(40-45)^2}=8.52$	1
8	25.7	46	No	$\sqrt{(43.6-25.7)^2+(40-46)^2}=18.88$	3
9	23.3	29	No	$\sqrt{(43.6-23.3)^2+(40-29)^2}=23.09$	8
10	31	56	Yes	$\sqrt{(43.6-31)^2+(40-56)^2}=20.37$	7

It is better to choose an odd K!

Common tie-breaking strategies

- Choose randomly

- Pick the first label
  - scikit-learn uses this strategy in the unweighted case
    - Rank the instances and pass the resulting array to the `scipy.stats.mode` function
      - This function returns the most common value in the passed array
      - If there is more than one such value, only the first is returned
      - Which one is the first?
        - Depends on the order of the samples in the training data

- Use the distance-weighted nearest neighbor algorithm

Distance-Weighted Nearest Neighbor

- Instead of a simple majority, weight votes by inverse distance
  - The closer neighbor has more influence

- Training Algorithm
  - For each training example $\langle x, f(x) \rangle$, add the example to the list *training_examples*

- Classification Algorithm
  - Given a query instance x_q to be classified,
    - Let $x_1 \ldots x_k$ denote the k instances from *training_examples* that are nearest to x_q
    - Return

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{K} w_i \delta(v, f(x_i))$$

Where

$$w_i = \frac{1}{d(x_q, x_i)}$$

KNN- Example-2

- What is the label for this data: (157, 54)?

ID	Height	Weight	Label
1	150	50	Medium
2	155	55	Medium
3	160	60	Large
4	161	59	Large
5	158	65	Large

…KNN- Example-2

- What is the label for this data: (157, 54)?
  - Distance = Euclidean
  - K = 3

ID	Height	Weight	Label	Distance	Rank
1	150	50	Medium	8.06	4
2	155	55	Medium	2.24	1
3	160	60	Large	6.71	2
4	161	59	Large	6.40	3
5	158	65	Large	11.05	5

…KNN- Example-2

- What is the label for this data: (157, 54)?
  - Without weighing, the label is Large

ID	Height	Weight	Label	Distance	Rank
1	150	50	Medium	8.06	4
2	155	55	Medium	2.24	1
3	160	60	Large	6.71	2
4	161	59	Large	6.40	3
5	158	65	Large	11.05	5

…KNN- Example-2

- What is the label for this data: (157, 54)?
  - With weighing, the label is Medium
    - 0.45 vs 0.31

ID	Height	Weight	Label	Distance	Rank	1/Distance
1	150	50	Medium	8.06	4	
2	155	55	Medium	2.24	1	0.45
3	160	60	Large	6.71	2	0.15
4	161	59	Large	6.40	3	0.16
5	158	65	Large	11.05	5	

KNN- Example-3

- Predict high-risk heart disease patients

ID	Gender	Smoke	cholesterol	Diabetic	Blood Pressure	Heart Disease
1	Male	Yes	High	No	Normal	No
2	Female	No	High	No	High	Yes
3	Male	No	No	Yes	Yes	Yes
4	Male	No	Normal	Yes	Normal	No
5	Male	Yes	Yes	No	Normal	Yes
6	Female	Yes	Normal	No	Normal	No
7	Female	No	Normal	No	High	No
8	Male	No	Normal	Yes	Normal	Yes
9	Female	No	Normal	Yes	Normal	No
10	Male	Yes	High	No	High	Yes

…KNN- Example-3

- Categorical instead of numeric data
  - How can we calculate the distance?
    - Hamming distance
      - Same values → distance = 0
      - Different values → distance = 1

…KNN- Example-3

- Does a smoking non-diabetic male with high blood pressure and cholesterol level have a risk of heart disease?

ID	Gender	Smoke	cholesterol	Diabetic	Blood Pressure	Heart Disease	Distance to (Male, Yes, High, No, High)	Rank
1	Male	Yes	High	No	Normal	No	$0+0+0+0+1=1$	1
2	Female	No	High	No	High	Yes	$1+1+0+0+0=2$	2
3	Male	No	Normal	Yes	High	Yes	$0+1+1+1+0=3$	3
4	Male	No	Normal	Yes	Normal	No	$0+1+1+1+1=4$	4
5	Male	Yes	High	No	Normal	Yes	$0+0+0+0+1=1$	1
6	Female	Yes	Normal	No	Normal	No	$1+0+1+0+1=3$	3
7	Female	No	Normal	No	High	No	$1+1+1+0+0=3$	3
8	Male	No	Normal	Yes	Normal	Yes	$0+1+1+1+1=4$	4
9	Female	No	Normal	Yes	Normal	No	$1+1+1+1+1=5$	5
10	Male	No	High	No	High	Yes	$0+1+0+0+0=1$	1

...KNN- Example-3

- K=3 → 2 Yeses vs. 1 No → Yes

ID	Gender	Smoke	cholesterol	Diabetic	Blood Pressure	Heart Disease	Distance to (Male, Yes, High, No, High)	Rank
1	Male	Yes	High	No	Normal	No	$0+0+0+0+1=1$	1
2	Female	No	High	No	High	Yes	$1+1+0+0+0=2$	2
3	Male	No	Normal	Yes	High	Yes	$0+1+1+1+0=3$	3
4	Male	No	Normal	Yes	Normal	No	$0+1+1+1+1=4$	4
5	Male	Yes	High	No	Normal	Yes	$0+0+0+0+1=1$	1
6	Female	Yes	Normal	No	Normal	No	$1+0+1+0+1=3$	3
7	Female	No	Normal	No	High	No	$1+1+1+0+0=3$	3
8	Male	No	Normal	Yes	Normal	Yes	$0+1+1+1+1=4$	4
9	Female	No	Normal	Yes	Normal	No	$1+1+1+1+1=5$	5
10	Male	No	High	No	High	Yes	$0+1+0+0+0=1$	1

…KNN- Example-3

- What about k=1?
  - More than K points are equally close (same distance)
  - If several points are exactly the same distance as the K-th neighbor, most implementations (including Scikit-learn) will include them all
    - This means you might end up with slightly more than K neighbors

Homework

- How should we handle a dataset that contains both numerical and categorical features at the same time?
  - Note: The Euclidean distance for numerical variables may differ in scale compared to the Hamming distance for categorical features.
    - You may need to combine them by weighted sum to balance categorical vs numerical influence.

Coding

- Scikit-Learn (sklearn): Scientific toolkit for machine learning
  - One of the most popular ML libraries
  - Free and open sourse
  - Built on NumPy, SciPy, and matplotlib
  - Can be used for
    - Classification
    - Regression
    - Clustering
    - Dimensionality reduction
    - Model selection and validation
    - Data preprocessing
    - …

scikit-learn

Original author	David Cournapeau
Developer	Google Summer of Code project
Initial release	June 2007; 18 years ago
Stable release	1.7.2[1] / 9 September 2025; 27 days ago
Repository	github.com/scikit-learn/scikit-learn
Written in	Python, Cython, C and C++[2]
Operating system	Linux, macOS, Windows
Type	Library for machine learning
License	New BSD License
Website	scikit-learn.org

Scikit-Learn Main Interface

- Estimators
  - The core interface of scikit-learn
  - The classes which can learn and estimate some parameters based on a dataset
  - Use `fit()` method to learn
  - The method takes
    - A dataset as a parameter, OR
    - Two datasets for supervised learning algorithms—the second dataset contains the labels.
  - Any other parameter is a hyperparameter
  - General syntax:
    - `estimator.fit(data, targets) #supervised learning`
    - `estimator.fit(data) #unsupervised learning`

...Scikit-Learn Main Interface

- Transformers
  - The estimators which can also transform a dataset based on the learned parameters
  - Use `transform()` method for transformation
  - The method takes a dataset and returns the transformed dataset
  - This transformation generally relies on the learned parameters
  - All transformers also have the `fit_transform()` method.
    - Equivalent to calling `fit()` and then `transform()`
      - Sometimes `fit_transform()` is optimized and runs faster
  - General syntax:
    - `new_data = transformer.transform(data)`
    - `new_data = transformer.fit_transform(data)`

...Scikit-Learn Main Interface

- Predictors
  - The estimators which are capable of making predictions
  - Use `predict()` method for prediction
  - The method takes a dataset of new instances and returns the corresponding predictions
  - It also has a `score()` method that measures the quality of the predictions
  - General syntax:
    - `prediction = predictor.predict(new_data)`
    - `score = predictor.score(new_data)`

…Scikit-Learn Main Interface

- Example
  - `SimpleImputer` is a transformer used to fill missing values
    - The "strategy" hyperparameter specifies how the missing values are filled
    - E.g. by the median of the values for the same feature
    - The `SimpleImputer`
      1. Learn the specified value (e.g. median) by the `fit()` method
      2. Fill missing values with the learnt median value by using `transform()` method
    - `fit_transform()` method is also applicable

```
from sklearn.impute import SimpleImputer
imputer = SimpleImputer(strategy="median")
imputer.fit(X)
transformed_X = imputer.transform(X)
```

…Scikit-Learn Main Interface

- Extra notes
  - Estimator's hyperparameters are accessible via public instance variables
    - Example: `imputer.strategy`
  - Estimator's learned parameters are accessible via public instance variables with an underscore suffix
    - Example: `imputer.statistics_`

Get Your Feet Wet

- Start with a simple toy project to see a typical ML workflow in scikit-learn

All the codes are available online:

https://github.com/hhomaei/aml

let's start coding…