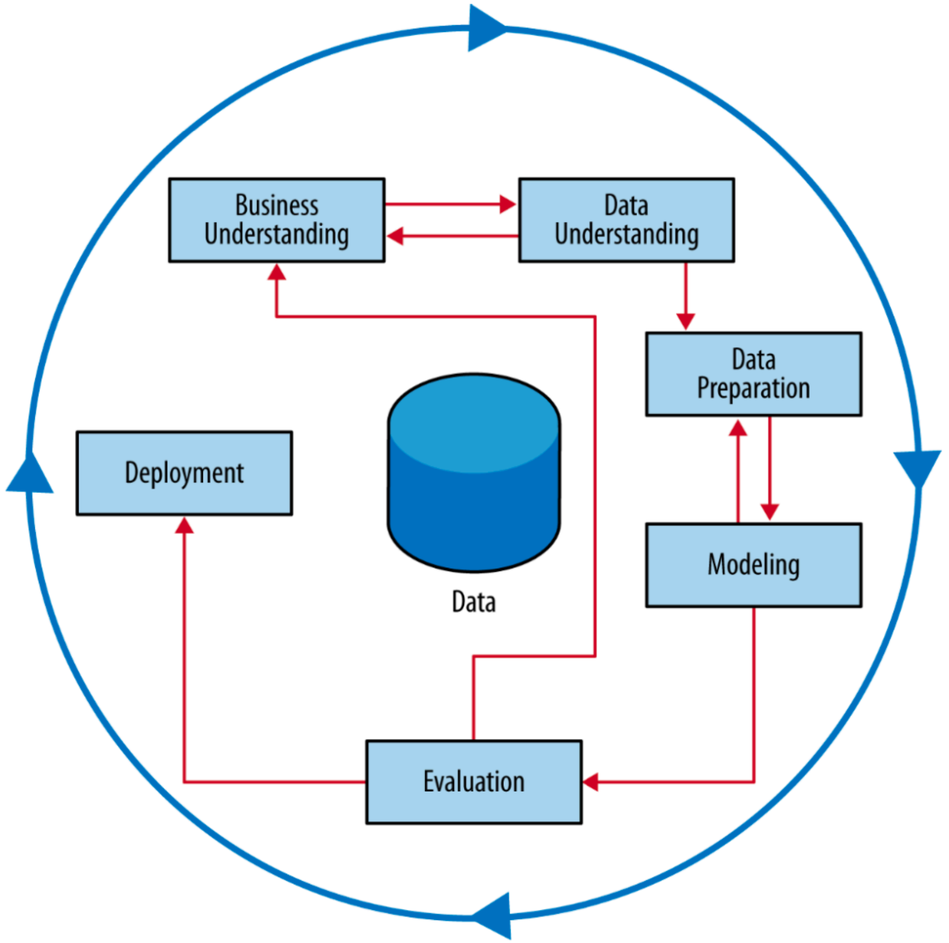
Introduction to Data Science

Lesson 4

Introduction to Machine Learning.
Supervised Learning.

Marija Stankova Medarovska, PhD
marija.s.medarovska@uacs.edu.mk

Data science process



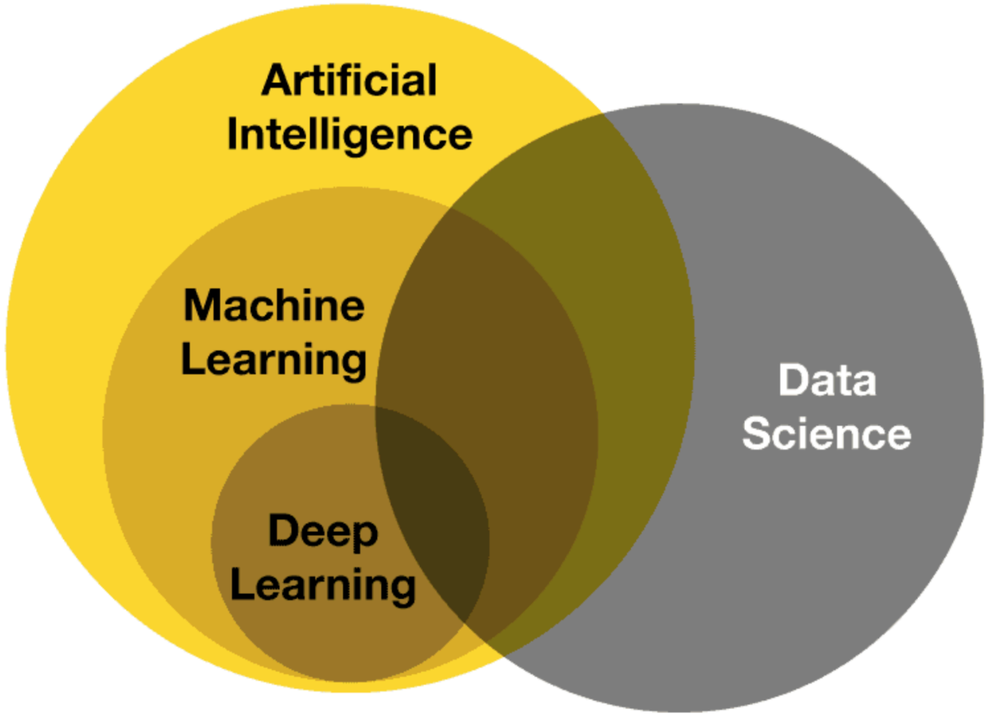
Introduction to Machine Learning

Overview

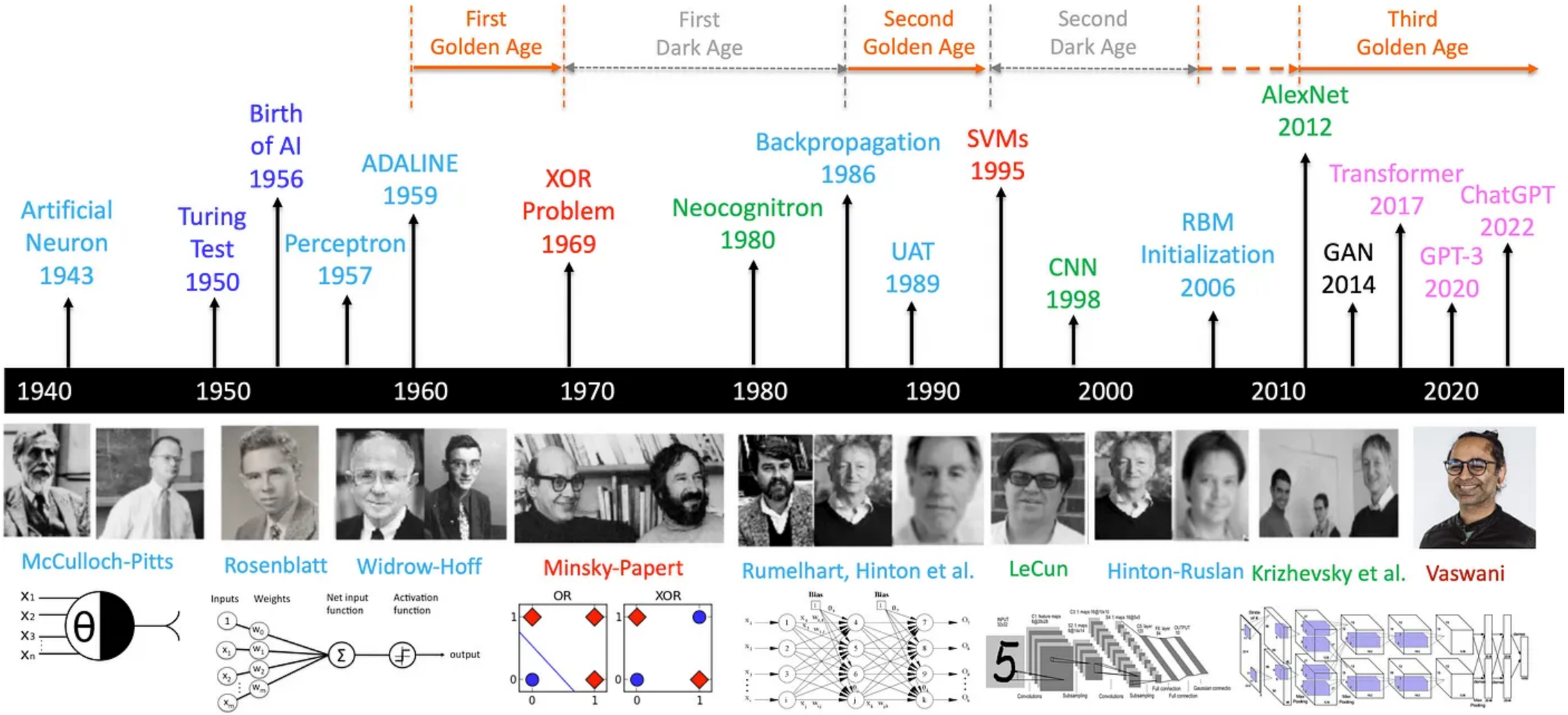
- Artificial Intelligence is the broad concept of developing machines that can simulate human thinking, reasoning and behavior.

- Machine Learning is a subset of AI that enables computer systems to learn from data and improve their performance over time without explicit programming.

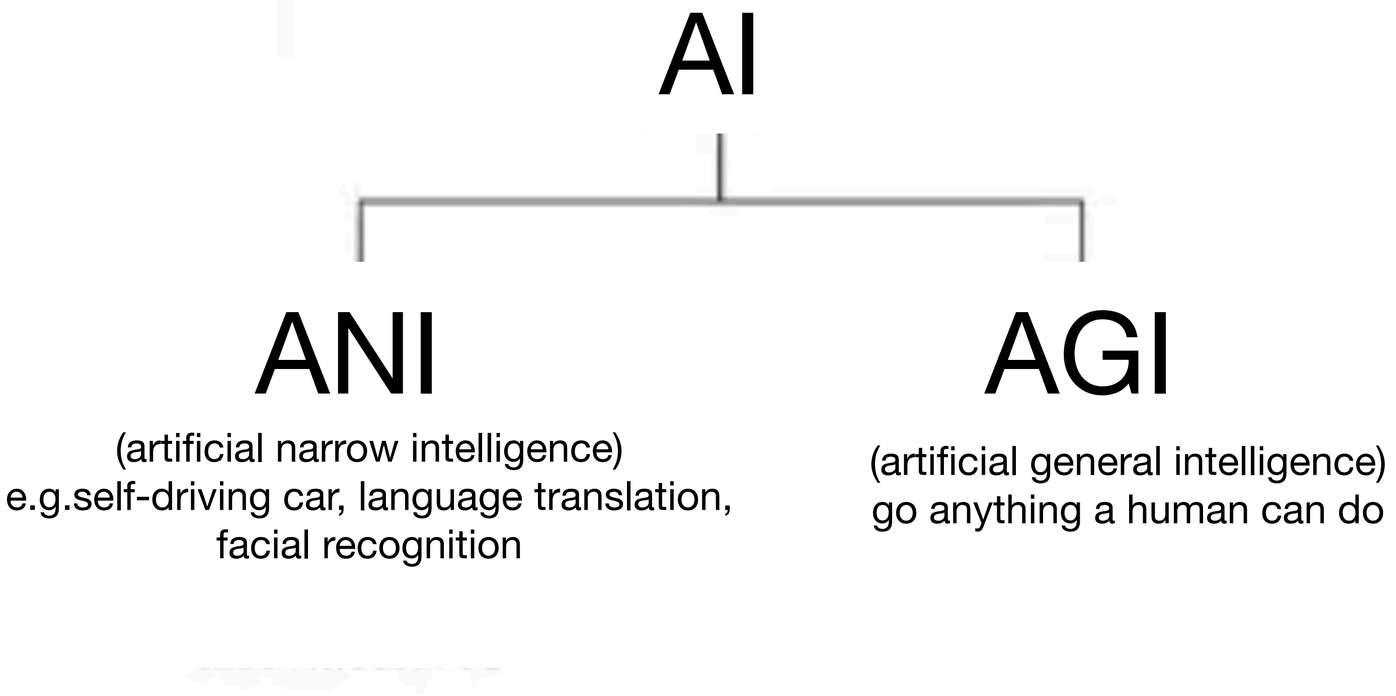
- Deep Learning is is part of a broader family of machine learning methods based on artificial neural networks.



Brief history of AI



Narrow vs. General Intelligence



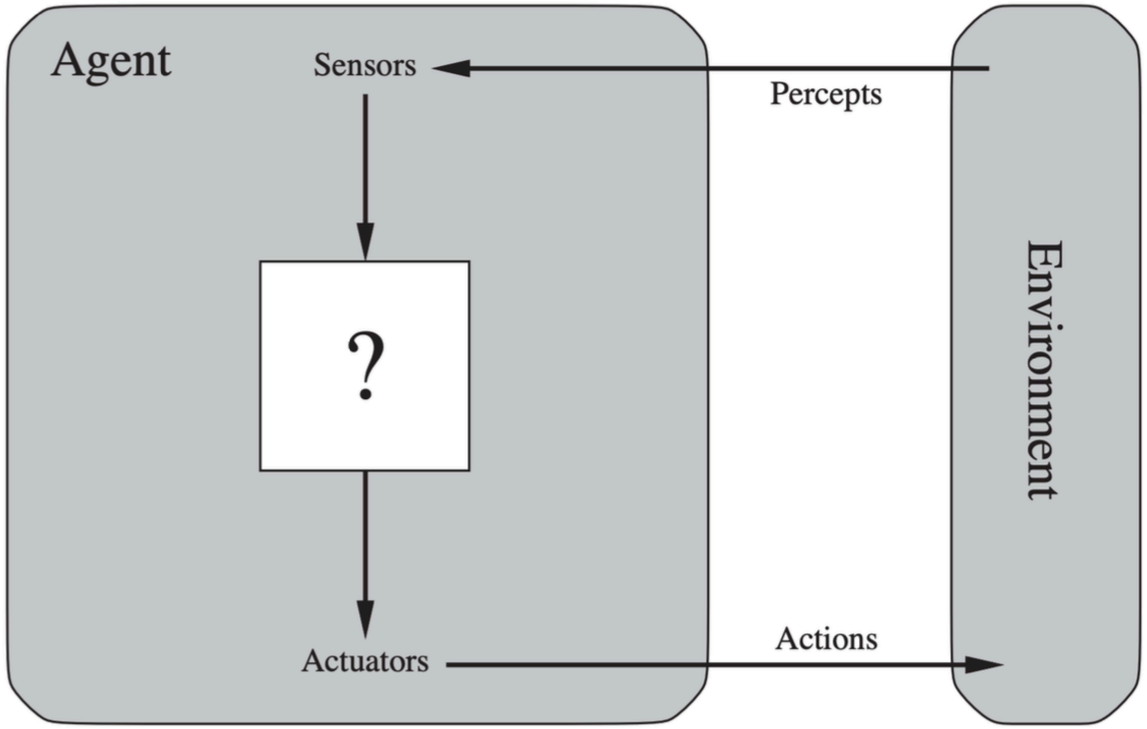
- Superintelligence: AI that surpasses human intelligence in all domains

https://www.microsoft.com/en-us/research/publication/sparks-of-artificial-general-intelligence-early-experiments-with-gpt-4/

Agent

Agent is an entity that perceives its environment through sensors, processes information, and takes actions to achieve specific goals.

Agents can be humans, animals, robots, software programs, etc.



Machine Learning

An agent is learning if it improves its performance after making observations about the world. When the agent is a computer, we call it machine learning.

Arthur Samuel's definition on Machine Learning (1959):
"Field of study that gives computers the ability to learn without being explicitly programmed."

Tom Mitchell's definition on Machine Learning (1997):
"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Original Image Via IBM

Arthur Samuel's checkers program developed in the late 1950s, utilized self-play machine learning to autonomously improve its performance over time, showcasing the potential of machine learning algorithms in artificial intelligence

Classes of Learning Problems

Supervised Learning

The agent observes input-output pairs and learns a function that maps from input to output

Data: (*x,y*)
x is data, *y* is label ("correct answer")

Goal: Learn function *f* to map $x \rightarrow y$

Apple example:

This thing is an apple.

Unsupervised Learning

The agent learns patterns in the input without any explicit feedback

Data: *x*
x is data, no labels available

Goal: Learn underlying structures

Apple example:


This thing is like the other thing.

Reinforcement Learning

The agent learns from a series of reinforcements: rewards and punishments

Data: state-action pairs

Goal: Maximize future rewards over many steps

Apple example:

Eat this thing because it will keep you alive.

Classes of Learning Problems

Supervised Learning

The agent observes input-output pairs and learns a function that maps from input to output

Data: (*x*,*y*)
x is data, *y* is label ("correct answer")

Goal: Learn function *f* to map $x \rightarrow y$

Apple example:

This thing is an apple.

Unsupervised Learning

The agent learns patterns in the input without any explicit feedback

Data: *x*
x is data, no labels available

Goal: Learn underlying structures

Apple example:


This thing is like the other thing.

Reinforcement Learning

The agent learns from a series of reinforcements: rewards and punishments

Data: state-action pairs

Goal: Maximize future rewards over many steps

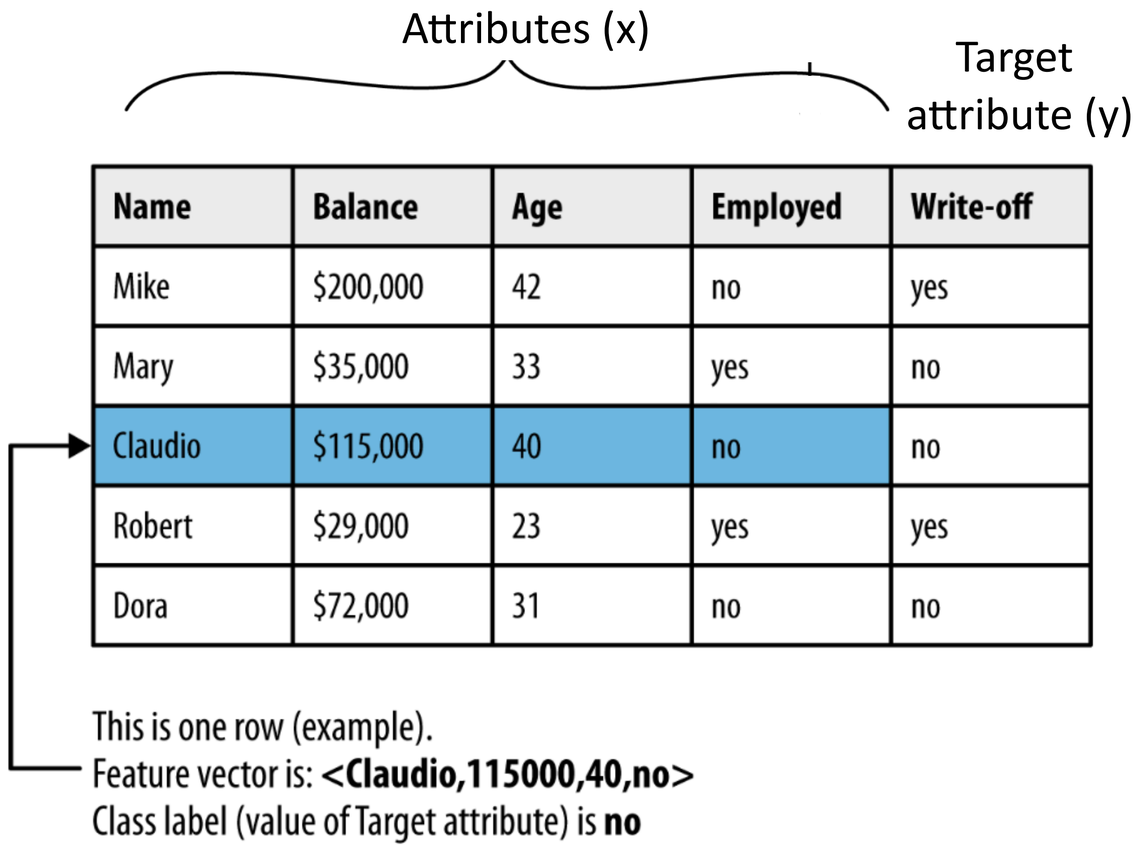
Apple example:

Eat this thing because it will keep you alive.

Supervised Learning

Supervised learning involves building a model for predicting, or estimating, an output based on one or more inputs.

For each input (instance with attributes x_i, $i = 1, \ldots, n$) there is an associated output (target attribute y). We wish to fit a model that relates the response to the input variables, with the aim of accurately predicting the response for future observations (prediction)



Name	Balance	Age	Employed	Write-off
Mike	$200,000	42	no	yes
Mary	$35,000	33	yes	no
Claudio	$115,000	40	no	no
Robert	$29,000	23	yes	yes
Dora	$72,000	31	no	no

This is one row (example).
Feature vector is: **<Claudio,115000,40,no>**
Class label (value of Target attribute) is **no**

Example: Supervised learning problem of predicting which customers would not repay their loan (default/write-off). An instance (example or row) represents a historical customer who had been given credit. It is described by a set of attributes (also known as variables or features).

Examples of Supervised Learning

Input (*x*)	Output (*y*)	Application
email	spam (*yes*/*no*)	spam filtering
audio	text transcript	speech recognition
english	spanish	machine translation
ad, user info	visit link (yes/no)	online advertising
MRI (Magnetic Resonance Imaging) data	brain tumor detection (yes/no)	medical diagnosis

Dear Sir.

First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial and top secret. ...

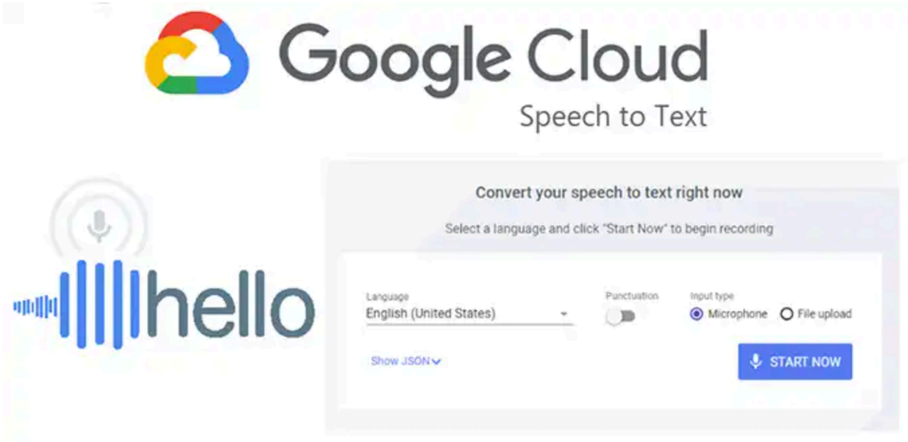
TO BE REMOVED FROM FUTURE MAILINGS, SIMPLY REPLY TO THIS MESSAGE AND PUT "REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES FOR ONLY $99

Ok, Iknow this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Examples of Supervised Learning

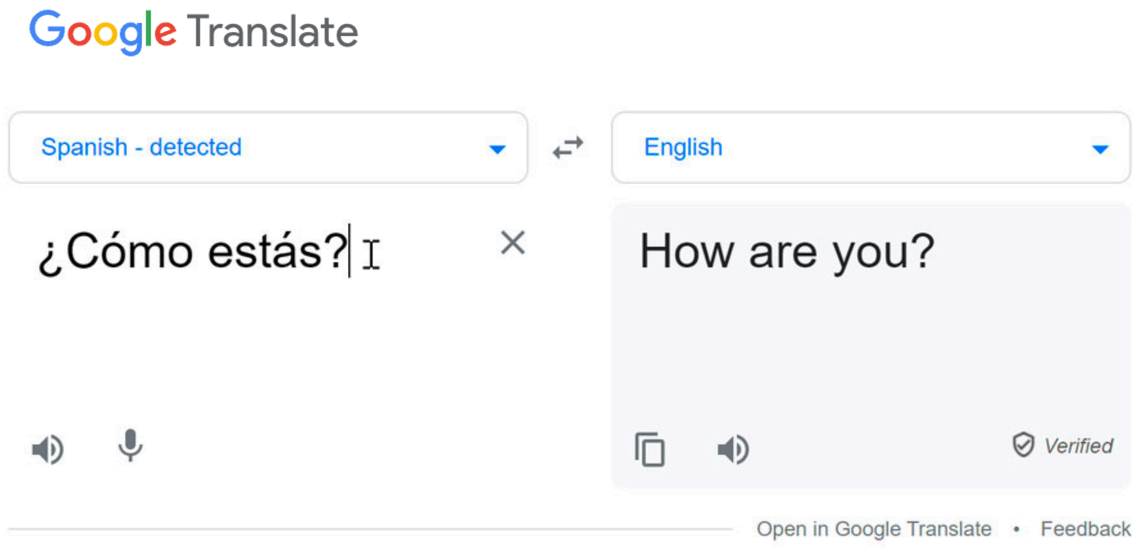
Input (*x*)	Output (*y*)	Application
email	spam (*yes*/*no*)	spam filtering
audio	text transcript	speech recognition
english	spanish	machine translation
ad, user info	visit link (yes/no)	online advertising
MRI (Magnetic Resonance Imaging) data	brain tumor detection (yes/no)	medical diagnosis



https://cloud.google.com/speech-to-text?hl=en

Examples of Supervised Learning

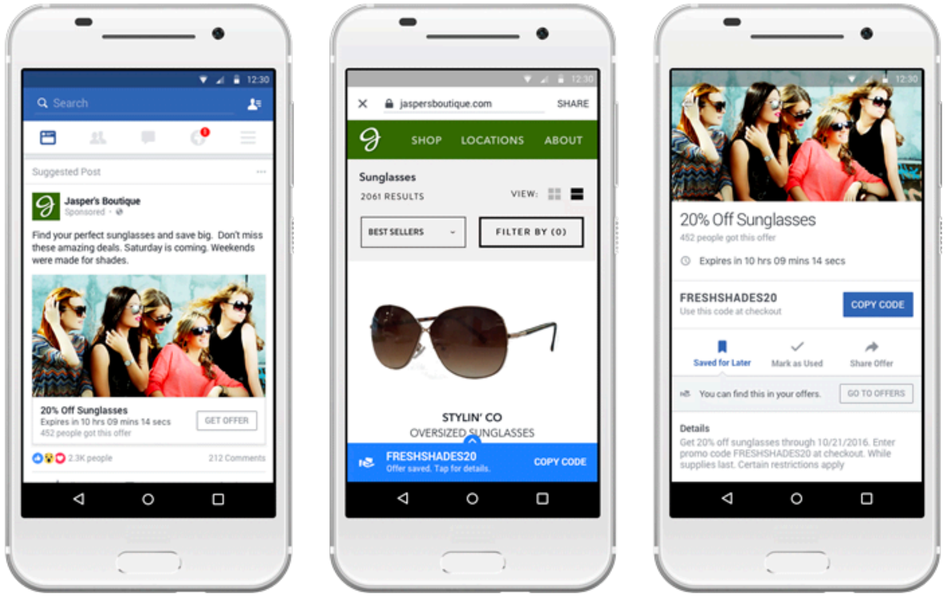
Input (*x*)	Output (*y*)	Application
email	spam (*yes*/*no*)	spam filtering
audio	text transcript	speech recognition
english	spanish	machine translation
ad, user info	visit link (yes/no)	online advertising
MRI (Magnetic Resonance Imaging) data	brain tumor detection (yes/no)	medical diagnosis

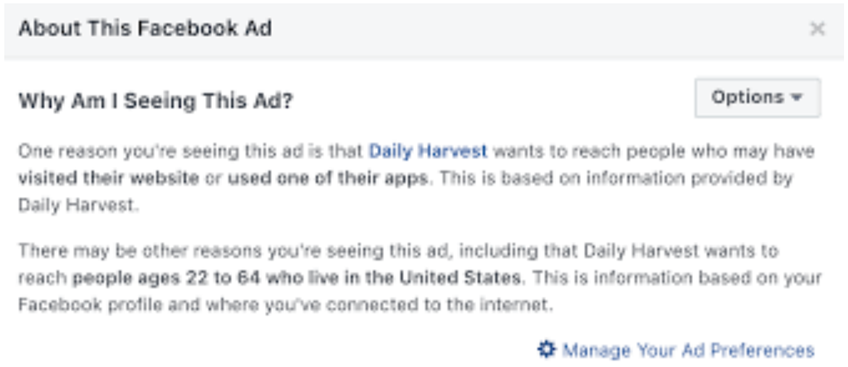


https://translate.google.com/

Examples of Supervised Learning

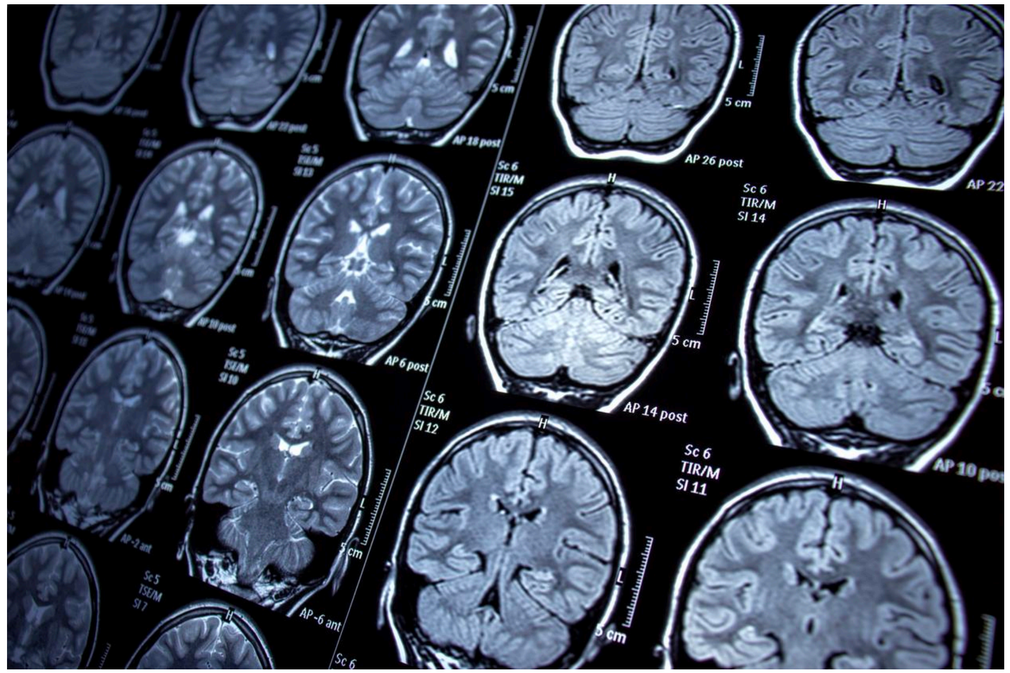
Input (*x*)	Output (*y*)	Application
email	spam (*yes*/*no*)	spam filtering
audio	text transcript	speech recognition
english	spanish	machine translation
ad, user info	visit link (yes/no)	online advertising
MRI (Magnetic Resonance Imaging) data	brain tumor detection (yes/no)	medical diagnosis





Examples of Supervised Learning

Input (*x*)	Output (*y*)	Application
email	spam (*yes*/*no*)	spam filtering
audio	text transcript	speech recognition
english	spanish	machine translation
ad, user info	visit link (yes/no)	online advertising
MRI (Magnetic Resonance Imaging) data	brain tumor detection (yes/no)	medical diagnosis



Classification problem

If the output has a finite set of values (ex. such as yes/*no* or *sunny/cloudy/rainy*), the task is called classification.

For example, predicting whether a client will default, i.e. not repay its bank loan (*yes/no*).

Attributes (x)

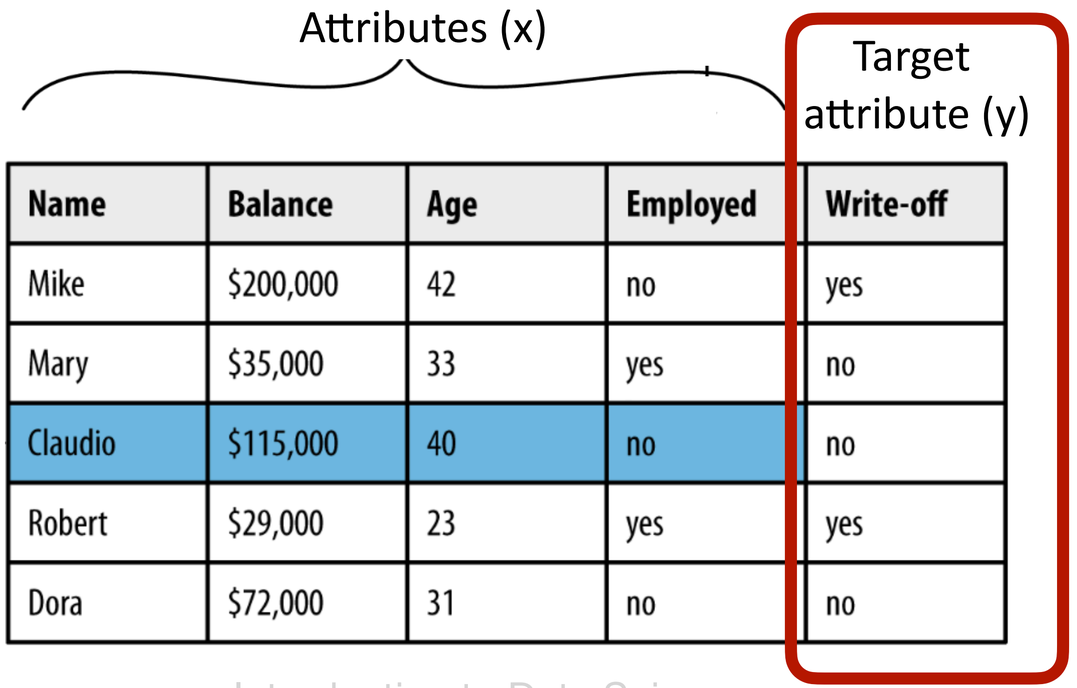
Target attribute (y)

Name	Balance	Age	Employed	Write-off
Mike	$200,000	42	no	yes
Mary	$35,000	33	yes	no
Claudio	$115,000	40	no	no
Robert	$29,000	23	yes	yes
Dora	$72,000	31	no	no

Binary vs multiclass classification

The distinct values that the output can have in a classification task are called classes (or labels)

If the output has only two classes, then the task is called binary classification (ex. *yes/no*). For more then two classes, the task is multiclass classification (ex. *sunny/cloudy/rainy*).



Name	Balance	Age	Employed	Write-off
Mike	$200,000	42	no	yes
Mary	$35,000	33	yes	no
Claudio	$115,000	40	no	no
Robert	$29,000	23	yes	yes
Dora	$72,000	31	no	no

Regression problem

When the output is a number (such as tomorrow's temperature), the task is called regression.

For example, predicting the price of a house in US dollars ($), based on information about the location, area of the house, number of bedrooms. condition, garden size, closeness to school, etc.

Selecting a model

In supervised learning, we assume there is some relationship between the output Y and the input X = (X_1,X_2,...,X_p), which can be written in the very general form:

$$Y = f(X) + \epsilon.$$

Here f is some fixed but unknown function of $X_1, \ldots, X_p$ and ε is a random *error term*, which is independent of X.

Based on the observed points we can estimate a function $\hat{f}$ that approximates the true function f and use it to predict the values of Y as $\hat{Y}$

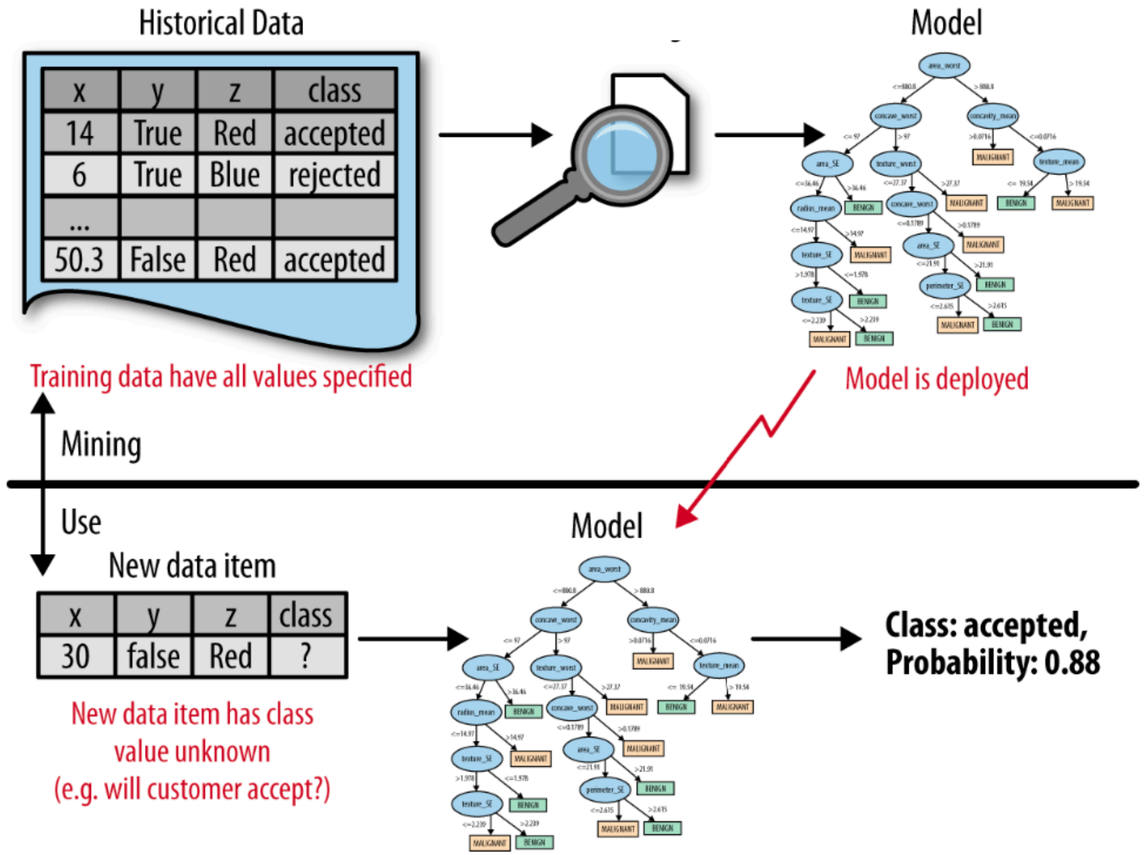
$$\hat{Y} = \hat{f}(X)$$

The accuracy of Y^ as a prediction for Y depends on two quantities, which we will call the *reducible error (*can be reduced using better techniques) and the *irreducible error* (ε may contain unmeasured variables that are useful in predicting Y, and ε may also contain unmeasurable variation).

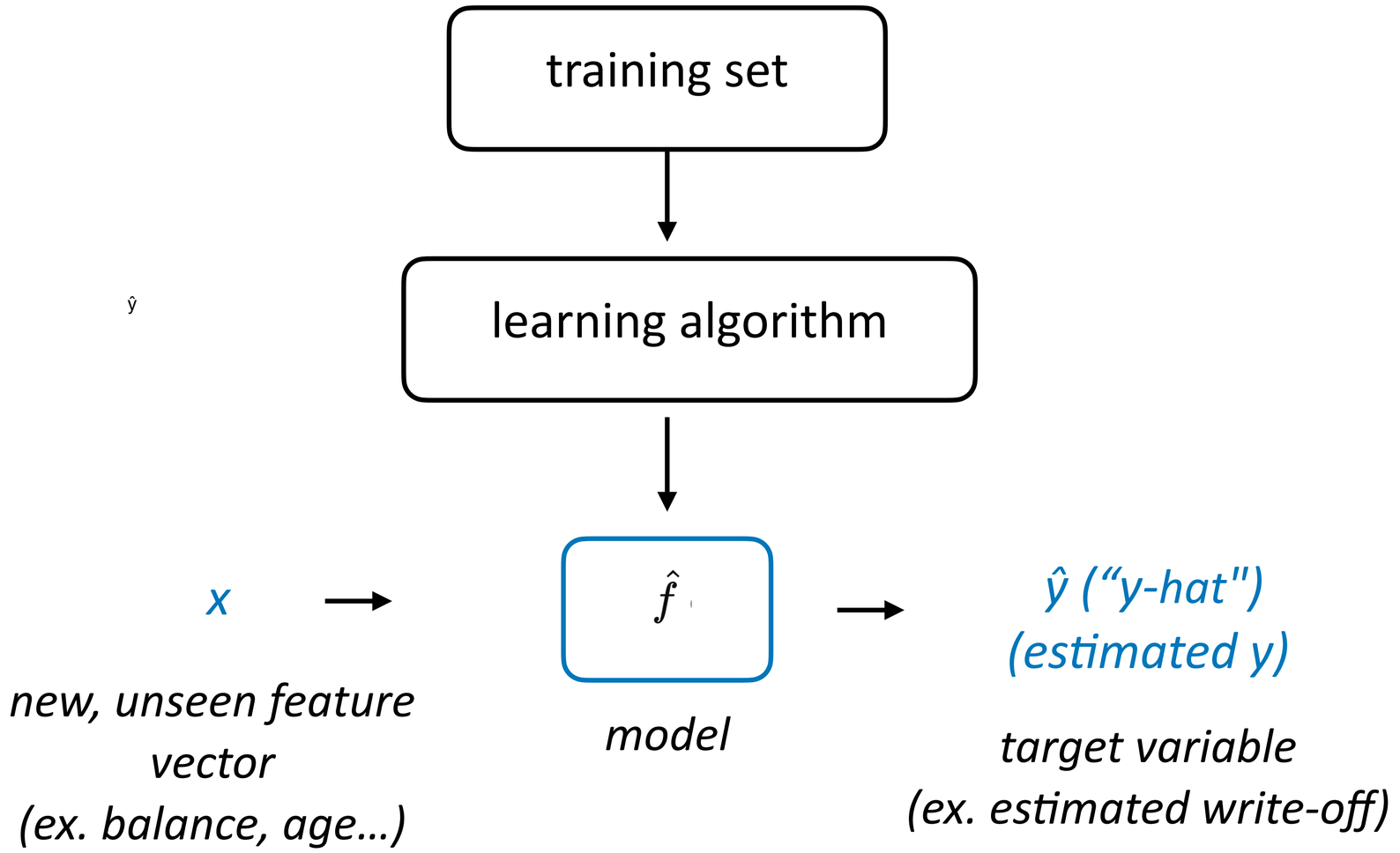
Selecting a model

How do we choose a good approximation for f?

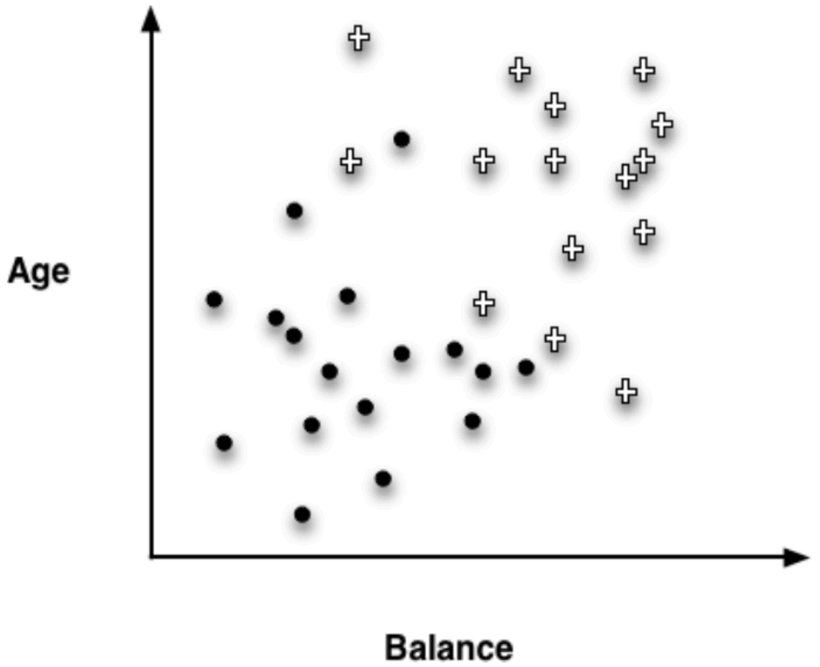
The true measure of a model not how it performs on the historical data (training set), but rather how well it handles inputs it has not yet seen. We can evaluate that with a second sample of (x_i, y_i) pairs called a test set. We say that $\hat{f}$ generalizes well if it accurately predicts the outputs of the test set.



Selecting a model

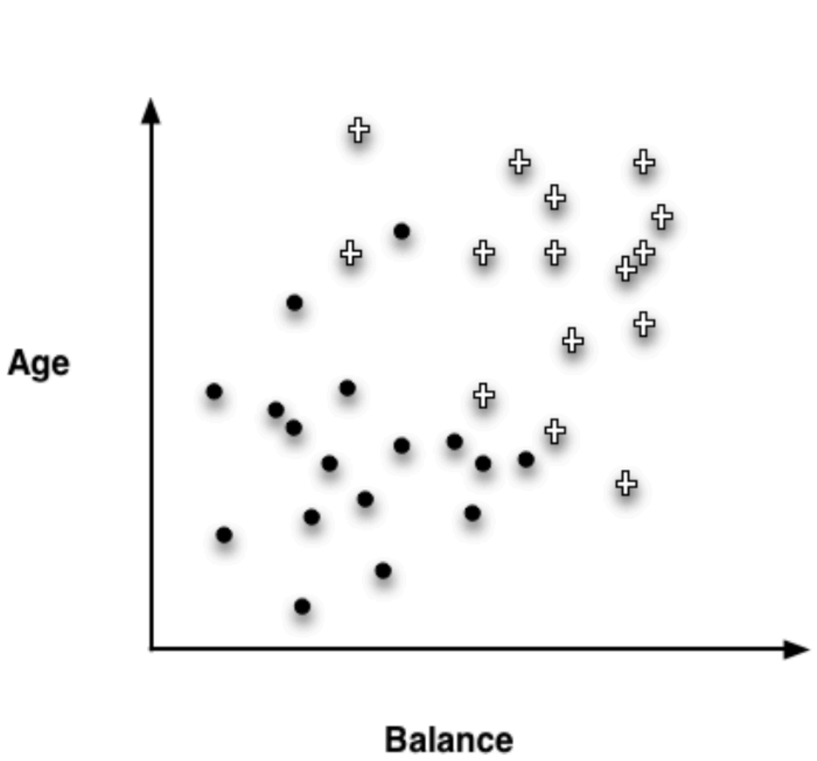


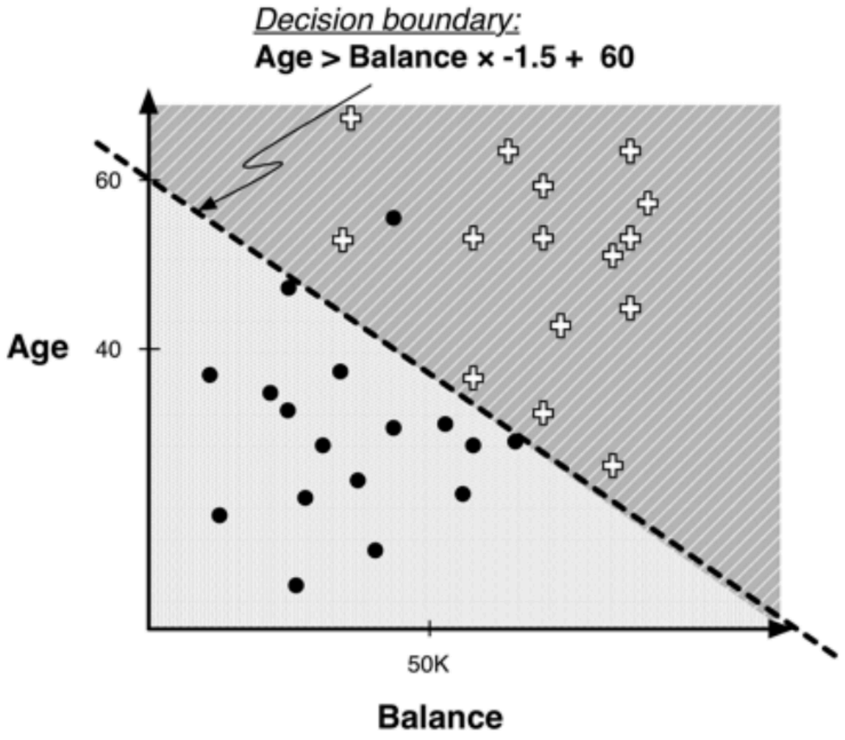
Selecting a model



The raw data points from our credit default prediction example, using only two variables Balance and Age. Customers who have repaid their loan and have a class label "no" are denoted by a sign (+), while those who have defaulted and have a class label "yes" are represented by a dot (•).

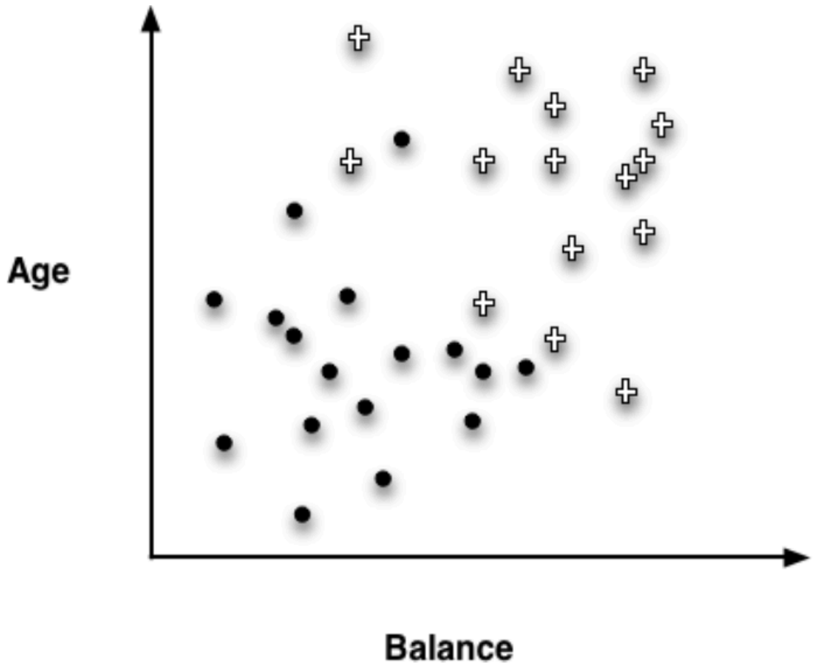
Selecting a model

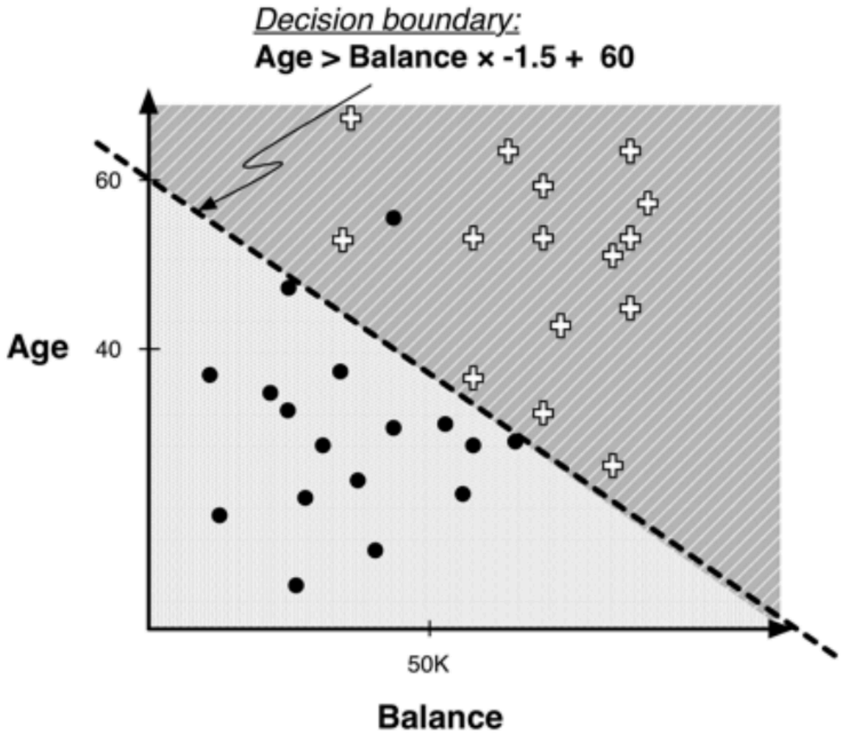


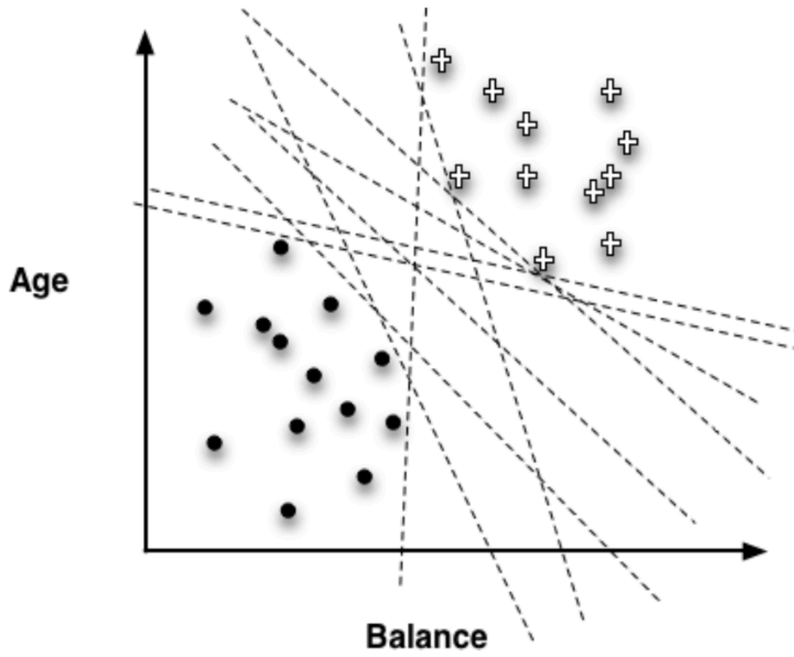


We want to partition the instance space using decision boundary/ies, in order to create as homogeneous as possible regions, so that we can predict the target variable of a new, unseen instance by determining which segment it falls into. For example, we can separate the instances almost perfectly (by class) if we introduce a boundary that is a straight line (linear classifier). if a new customer falls into the lower-left segment, we can conclude that the target value is very likely to be "•". Similarly, if it falls into the upper-right segment, we can predict its value as "+".

Selecting a model



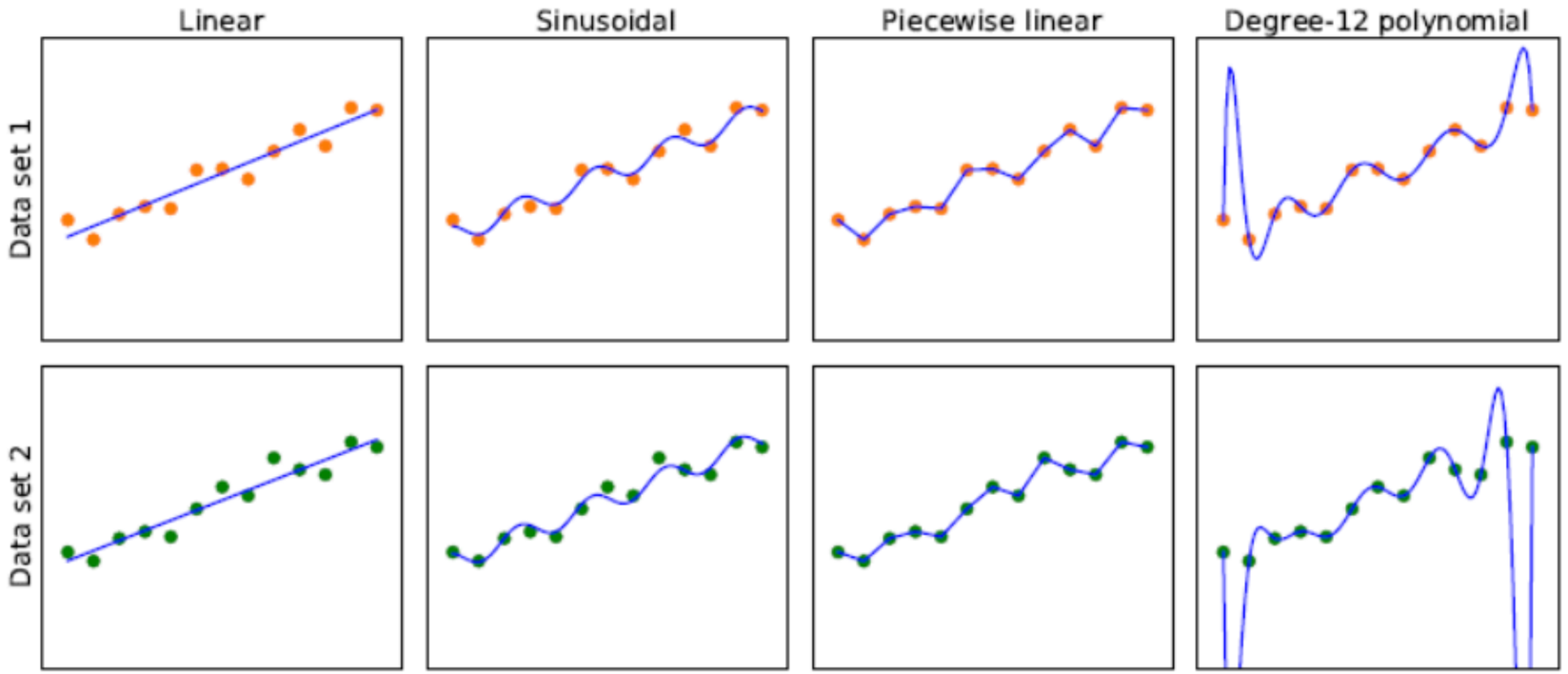




There are many different straight lines (linear discriminants) that can separate the classes from this example perfectly. They have very different slopes and intercepts, and each represents a different model of the data. In fact, there are infinitely many lines (models) that classify this training set perfectly.

Selecting a model

- **Linear**: Straight lines; functions of the form $h(x) = w_1 x + w_0$
- **Sinusoidal**: functions of the form $h(x) = w_1 x + \sin(w_0 x)$
- **Piecewise-linear**: functions where each line segment connects the dots from one data point to the next.
- **Degree-12 polynomials**: $h(x) = \sum_{i=0}^{12} w_i x^i$.



Top row: *four plots of best-fit functions from four different hypothesis spaces trained on data set 1.*
Bottom row: *the same four functions, but trained on a slightly different data set (sampled from the same f (x) function).*

Parametric and non-parametric methods

Parametric methods reduce the problem of estimating *f* down to one of estimating a set of parameters.

Assuming a parametric form for *f* simplifies the problem of estimating *f* because it is generally much easier to estimate a set of parameters, than it is to fit an entirely arbitrary function *f*.

The potential disadvantage of a parametric approach is that the model we choose will usually not match the true unknown form of *f*. If the chosen model is too far from the true *f*, then our predictions will be poor.

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

we need to estimate the parameters $\beta_0, \beta_1, \ldots, \beta_p$ *such that*

$$Y \approx \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

Example: First, we make an assumption about the functional form, or shape, of f. For example, one very simple assumption is that f is linear in X. Instead of having to estimate an entirely arbitrary p-dimensional function f(X), one only needs to estimate the coefficients $\beta_0, \beta_1, \ldots, \beta_p$.

Parametric and non-parametric methods

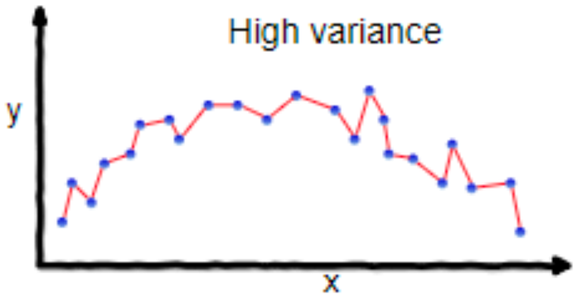
Non-parametric methods do not make explicit assumptions about the functional form of IInstead they seek an estimate of I that gets as close to the data points as possible without overfitting

Since they do not reduce the problem of estimating *f* to a small number of parameters, a very large number of observations (far more than is typically needed for a parametric approach) is required in order to obtain an accurate estimate for *f.*

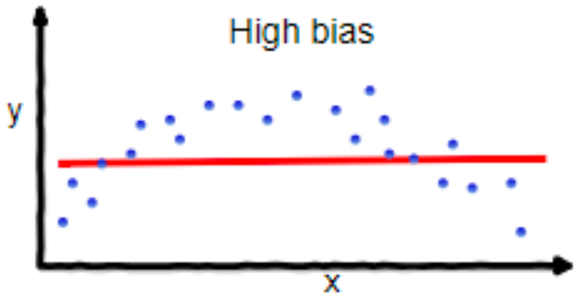
Overfitting and underfitting

Overfitting occurs when a model learns to capture noise and irrelevant details from the training data, making it perform well on the training set but poorly on new, unseen data. Essentially, the model becomes too complex and fits the training data too closely. se more complex models can lead to a phenomenon known as *overfitting* the data, which essentially means they follow the errors, or *noise*, too closely.

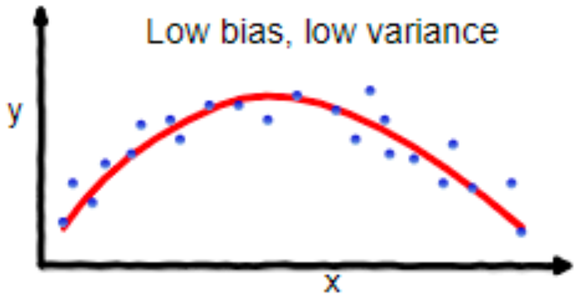
Underfitting, happens when a model is too simplistic to capture the underlying structure of the data. In this case, the model fails to learn the patterns present in the training data, resulting in poor performance both on the training set and on new data.



overfitting



underfitting



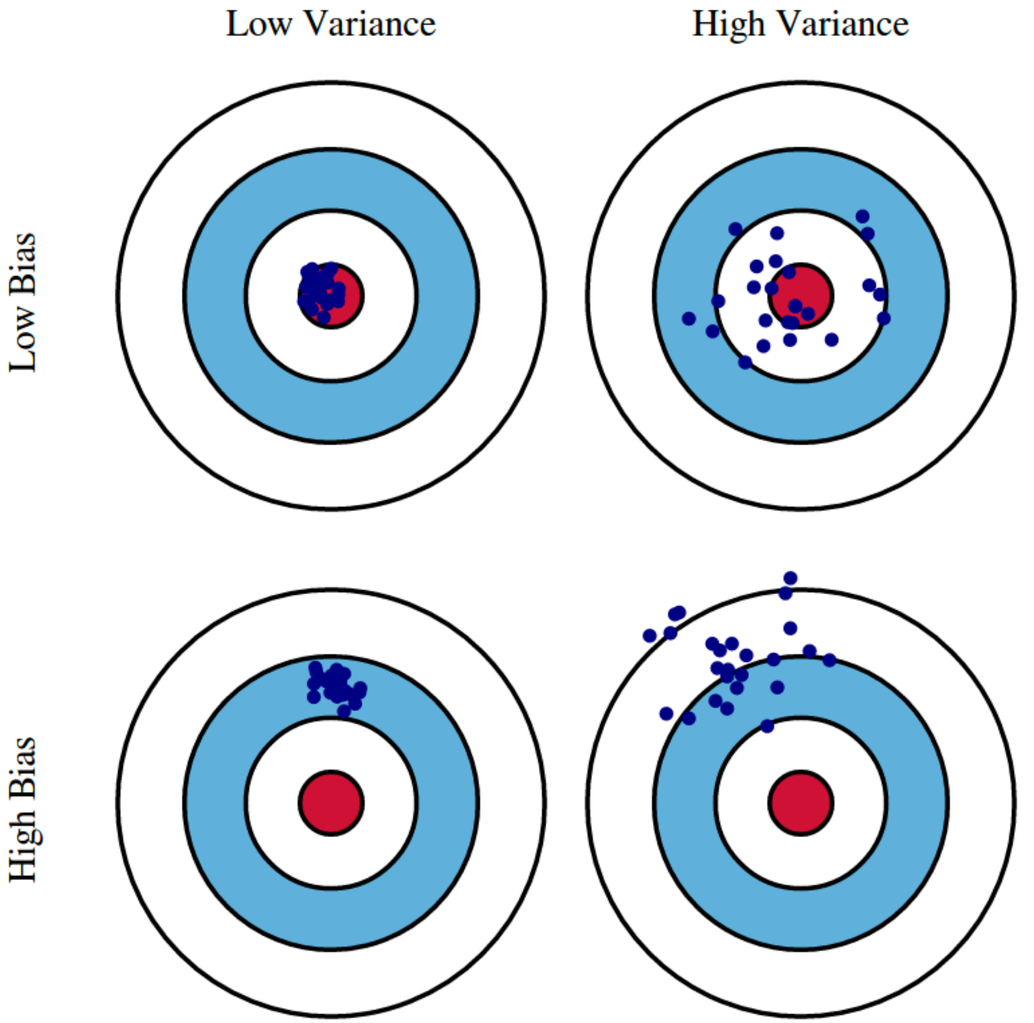
Good balance

Bias and variance

Bias occurs when a model simplifies a real-world problem too much, making strong assumptions about the data. High bias often leads to underfitting, where the model fails to capture the true relationship between features and the target variable.

Variance refers to the variability of a model's predictions for a given data point. A model with high variance is overly sensitive to the noise in the training data and captures random fluctuations as meaningful patterns, which can lead to overfitting.

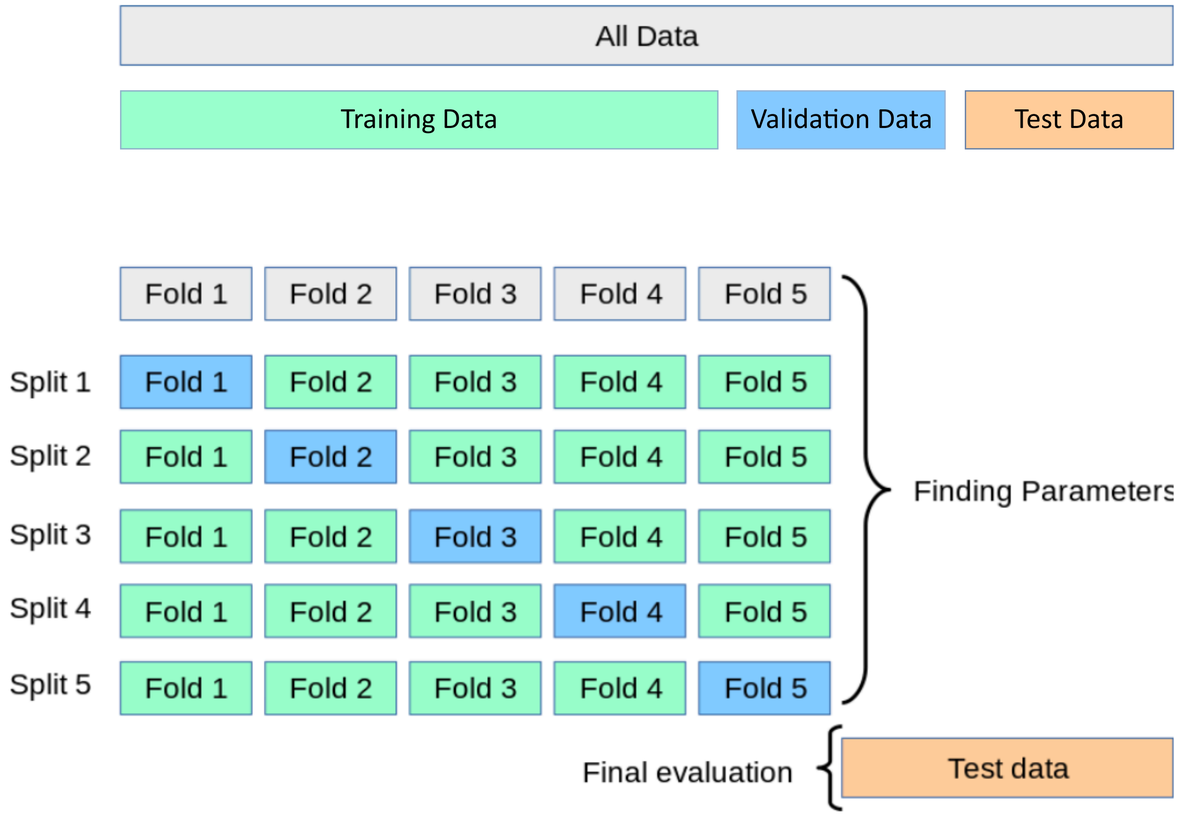
Often there is a bias–variance tradeoff: a choice between more complex, low-bias models that fit the training data well and simpler, low-variance models that may generalize better.



Validation set

Three data sets are needed:

1. A training set to train candidate models.
2. A validation set to evaluate the candidate models and choose the best one.
3. A test set to do a final unbiased evaluation of the best model.



k-fold cross-validation

- split the data into *k* equal subsets
- perform *k* rounds of learning
- on each round 1/*k* of the data are held out as a validation set and the remaining examples are used as the training set.
- Popular values for *k* are 5 & 10
- leave-one-out cross-validation (LOOCV), when *k*=*n*

Classes of Learning Problems

Supervised Learning

The agent observes input-output pairs and learns a function that maps from input to output

Data: (*x*,*y*)
x is data, *y* is label ("correct answer")

Goal: Learn function *f* to map $x \rightarrow y$

Apple example:

This thing is an apple.

Unsupervised Learning

The agent learns patterns in the input without any explicit feedback

Data: *x*
x is data, no labels available

Goal: Learn underlying structures

Apple example:


This thing is like the other thing.

Reinforcement Learning

The agent learns from a series of reinforcements: rewards and punishments

Data: state-action pairs

Goal: Maximize future rewards over many steps

Apple example:

Eat this thing because it will keep you alive.

Unsupervised Learning

Unsupervised learning involves situations in which we only observe input variables, with no corresponding output

Unlike in the previous examples, here we are not trying to predict an output variable. Instead of predicting a particular output variable, we are interested in determining whether there are groups, or clusters, on the basis of the variables measured.

Identifying such groups can be of interest because it might be that the groups differ with respect to some property of interest.

Example: In a marketing setting, we might have demographic information for a number of current or potential customers. We may wish to understand which types of customers are similar to each other by grouping individuals according to their observed characteristics. It might be that the groups differ with respect to some property of interest, such as spending habits.

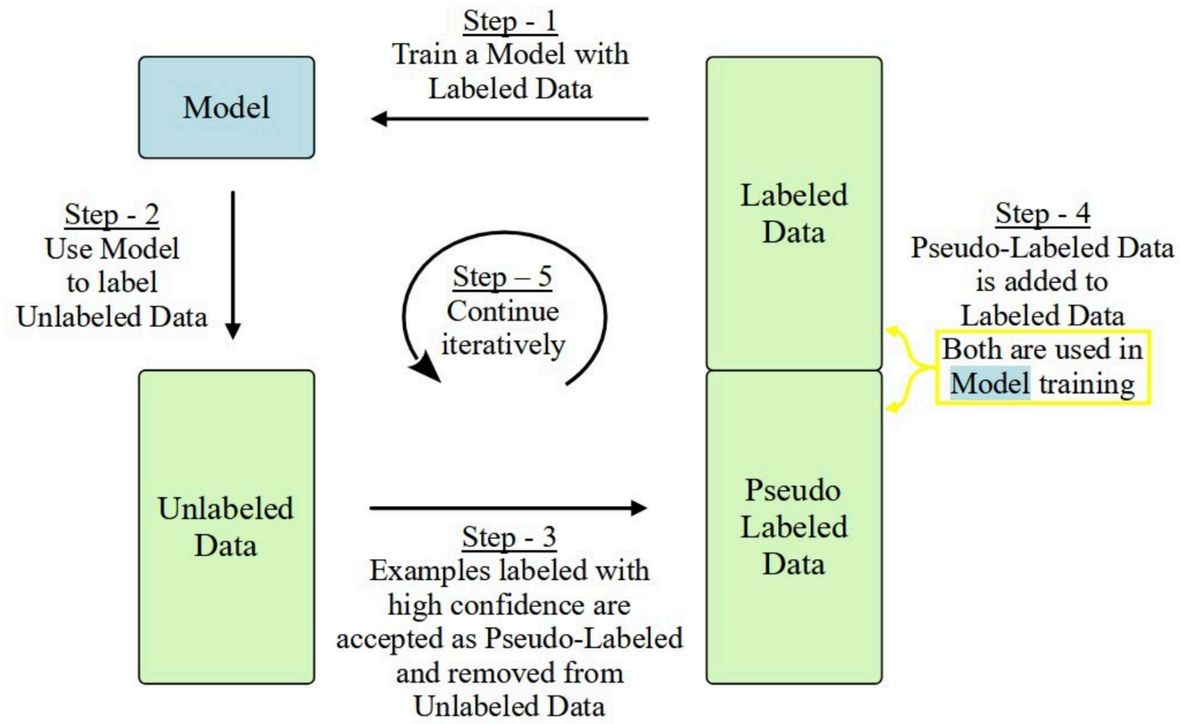
Semi-supervised Learning

Semi-supervised learning is a type of machine learning that lies between supervised and unsupervised learning. In this setting, a small portion of the dataset is labeled, where the correct output (class or target value) is known and for **a** larger portion of the dataset the correct output is unknown.

Such a scenario can arise if the variables can be measured relatively cheaply but the corresponding is much more expensive to collect (ex. labelling medical images).

Goal: Improve the accuracy of predictions by combining both labeled and unlabeled data, learning from the labeled data while also extracting useful patterns from the unlabeled data

Semi-supervised Learning



Example: Self-Training (or Self-Learning): A model is initially trained on the labeled data, then used to label the unlabeled data. The most confident predictions from the unlabeled data are added to the labeled dataset, and the model is retrained.

Classes of Learning Problems

Supervised Learning

The agent observes input-output pairs and learns a function that maps from input to output

Data: (*x*,*y*)
x is data, *y* is label ("correct answer")

Goal: Learn function *f* to map $x \rightarrow y$

Apple example:

This thing is an apple.

Unsupervised Learning

The agent learns patterns in the input without any explicit feedback

Data: *x*
x is data, no labels available

Goal: Learn underlying structures

Apple example:


This thing is like the other thing.

Reinforcement Learning

The agent learns from a series of reinforcements: rewards and punishments

Data: state-action pairs

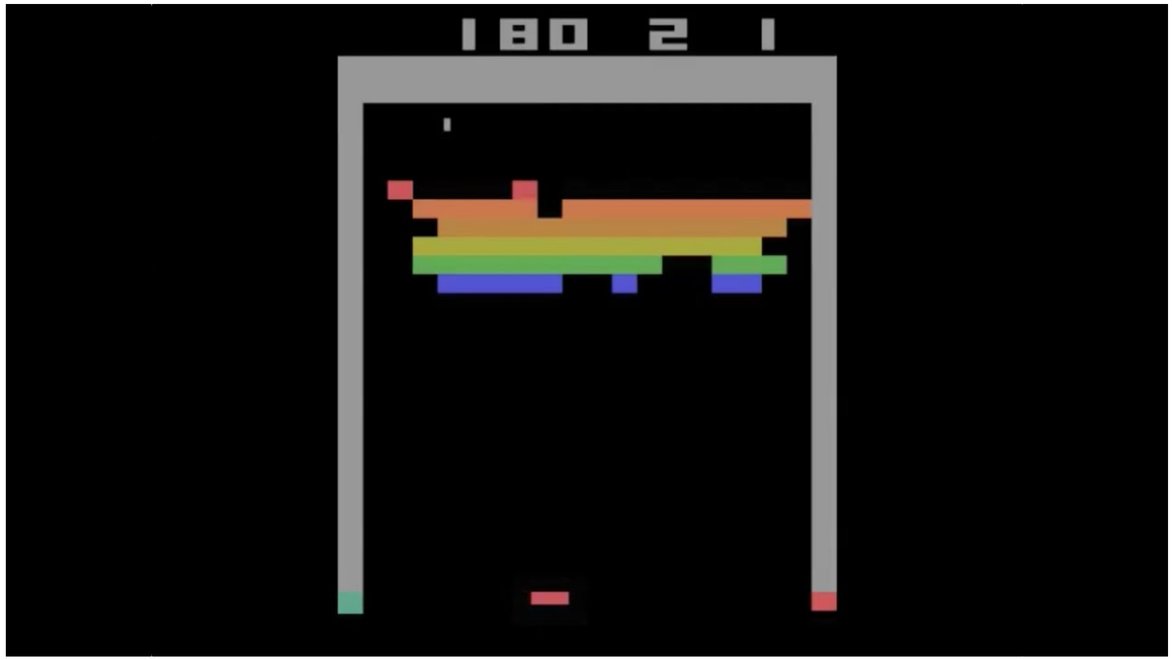
Goal: Maximize future rewards over many steps

Apple example:

Eat this thing because it will keep you alive.

Reinforcement Learning

In reinforcement learning (RL) the agent takes actions in an environment to maximize some notion of cumulative reward. It learns to make decisions by interacting with the world and receiving feedback In the form of reinforcement (rewards or penalties).



Deep Q Network (DQN), introduced by researchers at DeepMind, utilised reinforcement learning to achieve human-level performance in playing Atari 2600 games directly from raw visual input.

Appendix

History of AI*

1940-1950: Early days: neural and computer science meet

1943 McCulloch & Pitts: Boolean circuit model of brain

1950 Turing's "Computing Machinery and Intelligence"

1950—70: Logic-driven approach

1950s Early AI programs, including Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine

1956 Dartmouth meeting: "Artificial Intelligence" adopted

1965 Robinson's complete algorithm for logical reasoning

History of AI*

1970—90: Knowledge-based approaches

1969–79	Early development of knowledge-based systems
1980–88	Expert systems industry booms
1988–93	Expert systems industry busts: “AI Winter”
1985–95	Neural networks return to popularity

1990—: Statistical approaches

1988-	Resurgence of probability; focus on uncertainty
1995-	Agents, agents, everywhere…
1997	Deep Blue defeats Kasparov in chess
2000–	Human-level AI back on the agenda

History of AI*

2000 —: AI Growth

2002 iRobot launches Roomba, autonomous vacuum cleaner

2011 IBM Watson wins in the US show Jeopardy

2011 Apple launches Siri

2017 AlphaGo beats world GO champion Ke Jie

2018 European Commission publishes white paper on Ethics Guidelines for Trustworthy Artificial Intelligence

2022 OpenAI launches ChatGPT

…

Examples of Reinforcement Learning: Playing Go Game (Google DeepMind)

2016: Google's AlphaGo (later also AlphaZero and MuZero) used deep learning, including convolutional neural networks and reinforcement learning with Monte Carlo Tree Search to master the game of Go, achieving unprecedented success in strategic decision-making tasks.