A Brief Introduction to Machine Learning

Runyu Tang

Feburary 22, 2023

What is Machine Learning?

Machine learning (ML) is a field of inquiry devoted to understanding and building methods that 'learn', that is, methods that leverage data to improve performance on some set of tasks.

Machine Learning v.s. Statistical Learning.

Wiki treats them the same!

Email Spam Filtering

TABLE 1.1. *Average percentage of words or characters in an email message equal to the indicated word or character. We have chosen the words and characters showing the largest difference between* spam *and* email.

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

Classification: Spam or not spam?

Handwritten Digit Recognition

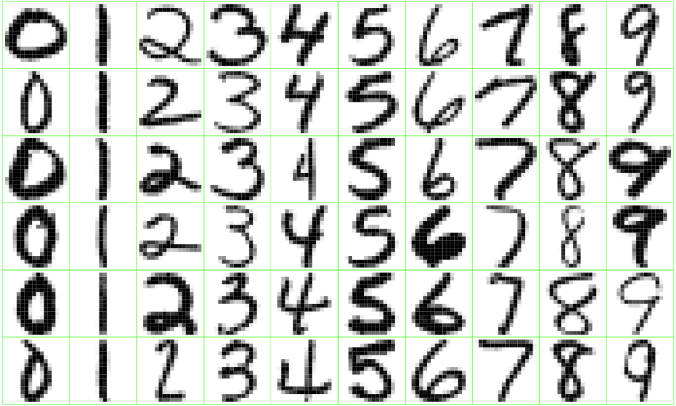


FIGURE 1.2. *Examples of handwritten digits from U.S. postal envelopes.*

Customer Segmentation and Recommendation System

Self-driving vehicles

AlphaGo

ChatGPT (and **Copilot**)

OpenAI   API   RESEARCH   BLOG   ABOUT

ChatGPT: Optimizing Language Models for Dialogue

We've trained a model called ChatGPT which interacts in a conversational way. The dialogue format makes it possible for ChatGPT to answer followup questions, admit its mistakes, challenge incorrect premises, and reject inappropriate requests. ChatGPT is a sibling model to InstructGPT, which is trained to follow an instruction in a prompt and provide a detailed response.

November 30, 2022
13 minute read

GPT: Generative Pre-trained Transformer

Transformers typically undergo self-supervised learning involving unsupervised pre-training followed by supervised fine-tuning.

(Natural Language Processing) NLP:

- RNN (recurrent neural network), LSTM (long short term memory)
- Transformer: Vaswani. et. al. (2017). Attention is all you need. *NeurIPS*. (65000+ citations). Originally for language translation.
- A good illustration site http://jalammar.github.io/illustrated-transformer/
- BERT (Bidirectional Transformers, 2018, 59000+ citations), 0.3 billion parameters. It rapidly starts to power Google Search.
- GPT (Masked Self Attention), GPT-2 1.5 billion parameters, GPT-3 175 billion parameters,

Exclusive: OpenAI Used Kenyan Workers on Less Than $2 Per Hour to Make ChatGPT Less Toxic

https://time.com/6247678/openai-chatgpt-kenya-workers/

Supervised Learning:

- Regression
- LDA, SVM, kNN
- Tree Models
- Neural Networks

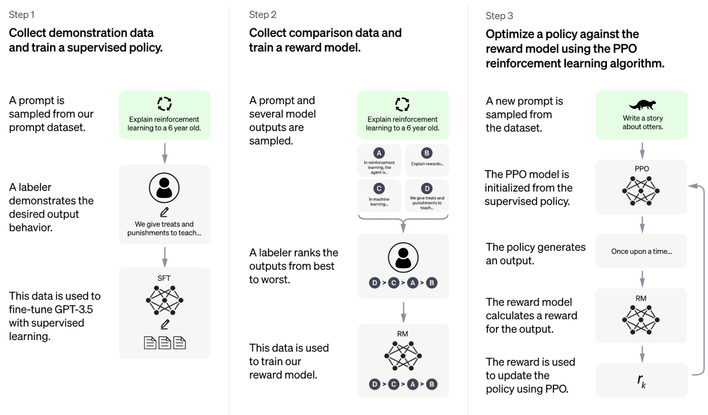
Unsupervised Learning:

- Clustering
- Dimension Reduction

Reinforcement Learning:

- Q-learning

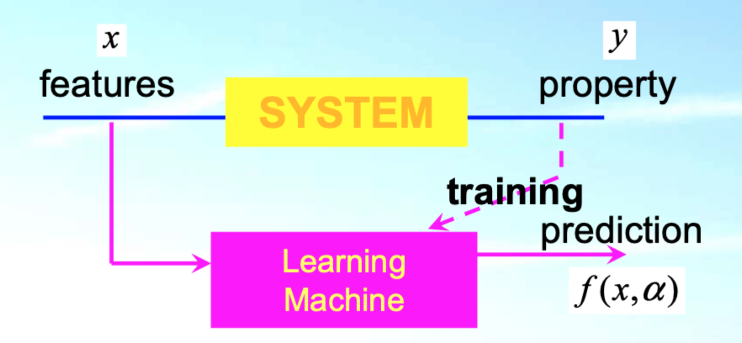
Semi-supervised (both labelled and unlabelled data),
Self-supervised learning (divide the data into x and y).



fine-tune the model using **Proximal Policy Optimization.**

https://openai.com/blog/chatgpt/

Training data: $(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), (\mathbf{x}_N, \mathbf{y}_N)$



Regression

- Linear Regression
- Logistic Regression
- Kernel Regression

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \epsilon$$
$$Y = \beta^T \mathbf{X} + \epsilon$$

We can use the least square method to estimate β.

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} (y_i - \boldsymbol{\beta}^T \mathbf{x}_i)^2$$

The estimator of $\hat{\boldsymbol{\beta}}$ is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T Y$$

$Y \in \{0, 1\}$, Classification

$$Z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \epsilon$$
$$Y = \frac{1}{1 + \exp(-Z)} \in [0, 1]$$

We can use the maximum likelihood method (MLE) to estimate β.

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} \prod_{i=1}^{n} P(y_i|\mathbf{x}_i) = \arg\max_{\boldsymbol{\beta}} \prod_{i=1}^{n} \left(\frac{1}{1 + \exp(-\boldsymbol{\beta}^T \mathbf{x}_i)} \right)^{y_i} \left(1 - \frac{1}{1 + \exp(-\boldsymbol{\beta}^T \mathbf{x}_i)} \right)^{1-y_i}$$

Multiclass classification

- One-vs-all Method:
- Multinomial Logistic Regression
  - Softmax function:

$$\sigma(\mathbf{z})_j = \frac{\exp(z_j)}{\sum_{k=1}^{K} \exp(z_k)}$$

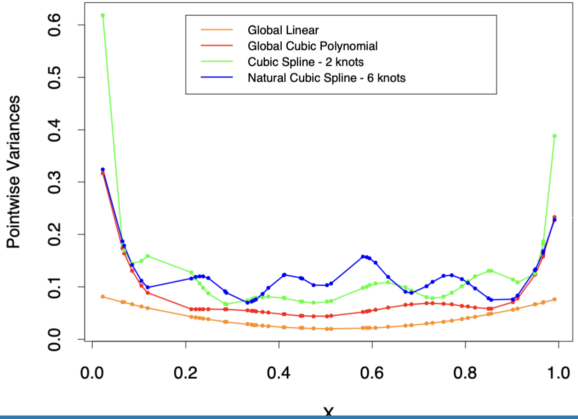
  - Probability of in class c:

$$\Pr(Y_i = c) = \sigma(\mathbf{z})_c = \frac{\exp(\beta_c X_i)}{\sum_{k=1}^{K} \exp(\beta_k X_i)}$$

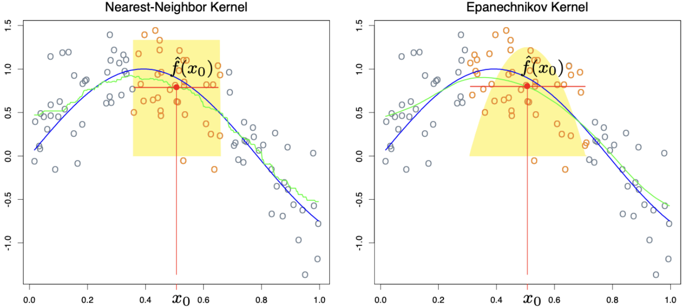
  - $K - 1$ classes:

$$\Pr(Y_i = c) = \frac{\exp(\beta_c X_i)}{1 + \sum_{k=1}^{K-1} \exp(\beta_k X_i)} \text{ if } c \neq K, (= \frac{1}{1 + \sum_{k=1}^{K-1} \exp(\beta_k X_i)} \text{ otherwise.})$$

- Polynomial
- Spline: adds additional constraints, namely that the function is linear beyond the boundary knots



- Kernel methods: a weighting function or kernel $K_\lambda(x_0, x_i)$, which assigns a weight to x_i based on its distance from x_0.



Nadaraya-Watson with Epanechnikov kernel: $\hat{Y}(x) = \frac{\sum_{i=1}^{n} K_\lambda(x,x_i) y_i}{\sum_{i=1}^{n} K_\lambda(x,x_i)}$,

where $K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right), D(t) = \frac{3}{4}(1 - t^2)$, if $|t| \le 1 (0,$ otherwise).

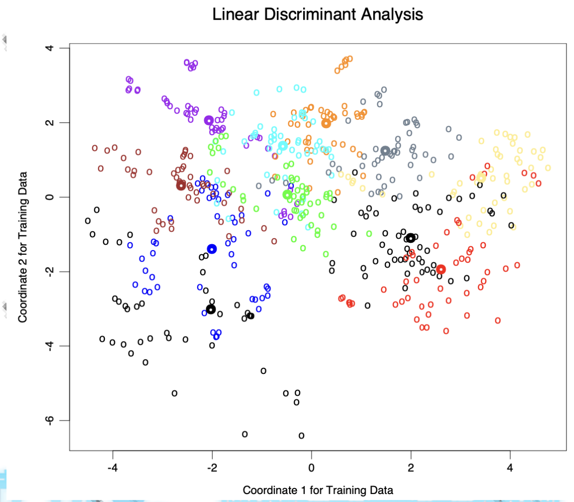
Linear Discriminant Analysis(LDA) (not Latent Dirichlet Allocation)

LDA takes a different approach to classification than logistic regression. Rather than attempting to model the conditional distribution of Y given X, $P(Y = k|X = x)$, LDA models the distribution of the predictors X given the different categories that Y takes on, $P(X = x|Y = k)$.

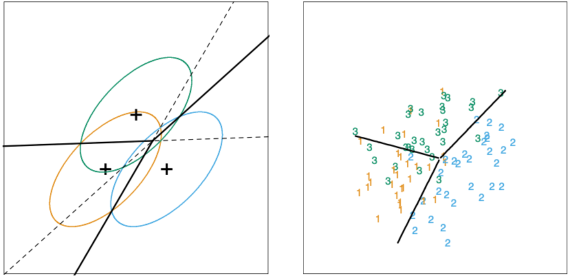
Bayes' theorem:

$$P(Y = k|X = x) = \frac{P(X = x|Y = k)P(Y = k)}{P(X = x)}$$

The Bayes' classifier is then selected. That is the observation assigned to the group for which the posterior probability is the largest.



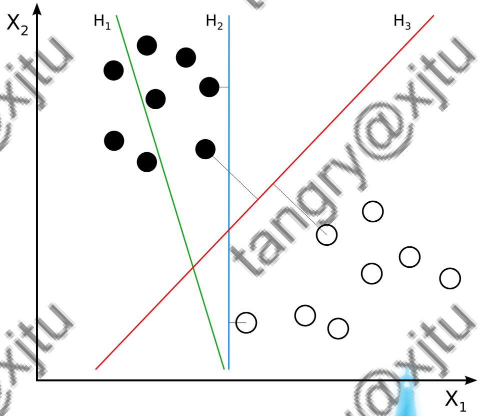
- A two-dimensional plot of the vowel training data. There are eleven classes with $X \in \mathbb{R}^{10}$, and this is the best view in terms of a LDA model. The heavy circles are the projected mean vectors for each class. The class overlap is considerable.



The left panel shows three Gaussian distributions, with the same covariance and different means. Included are the contours of constant density enclosing 95% of the probability in each case. The Bayes decision boundaries separating all three classes are the thicker solid lines. On the right we see a sample of 30 drawn from each Gaussian distribution, and the fitted LDA decision boundaries.

Support Vector Machine(SVM)

- Separate p–dimensional points with a $(p-1)$-dimensional hyperplane.
- For the RHS figure, H_1 does not separate the classes.
- H_2 does, but with a small margin, H_3 separates them with the maximal margin.



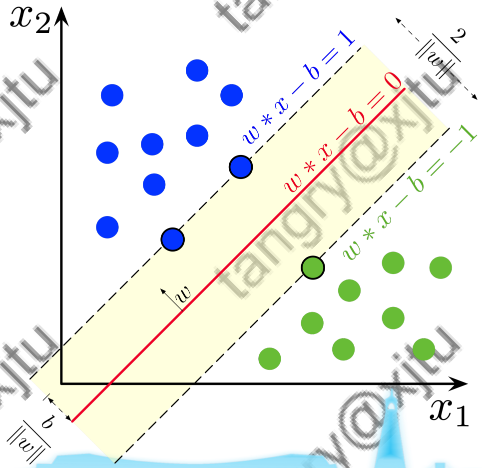
Support Vector Machine(SVM)

- Data points: $(\mathbf{x}_i, y_i)$, $y_i \in \{-1, 1\}$.
- Hard-margin:

$$\min_{\mathbf{w}, b} \quad \frac{1}{2}\|\mathbf{w}\|^2$$
$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1,\ i = 1, \ldots, n$$

- Soft-margin (not linearly separable):

$$\min_{\mathbf{w}, b} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i,\ i = 1, \ldots, n$$
$$\xi_i \geq 0,\ i = 1, \ldots, n$$

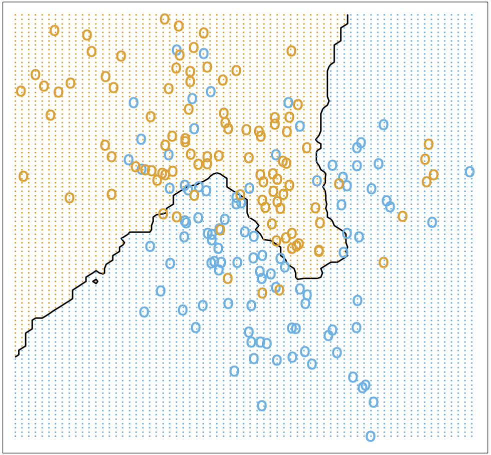


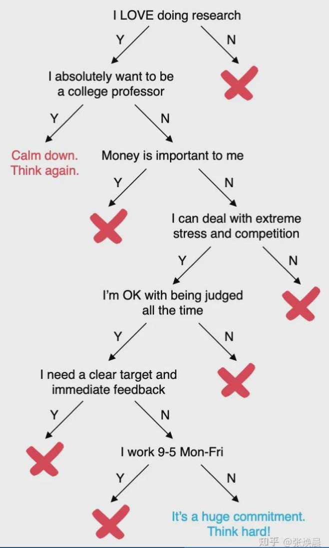
K-nearest neighbors (KNN)

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$$

single parameter: k.

15-Nearest Neighbor Classifier





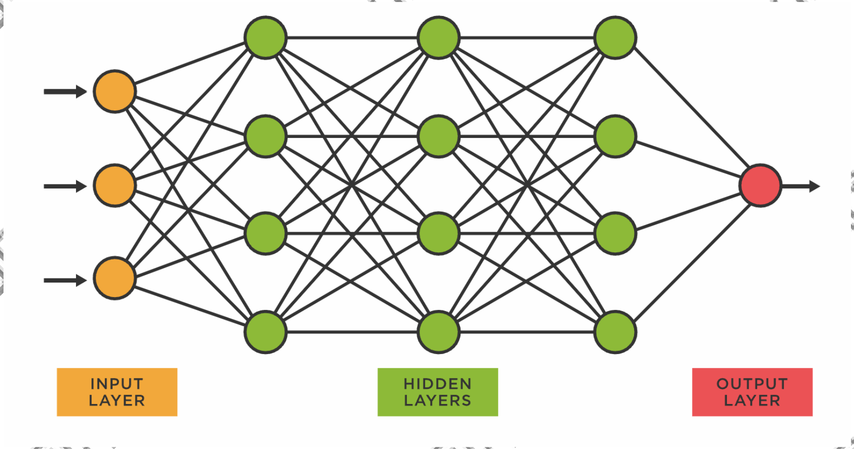
Decision tree classifier

- Python: sklearn.tree.DecisionTreeClassifier
- R: rpart
- Pros: easy to understand, easy to interpret, fast to train, can handle both numerical and categorical data, can handle multi-output problems, can handle missing values
- Cons: can be unstable, can be biased, can be overfitting

Ensemble methods: use multiple learning algorithms to obtain better predictive performance than could be obtained from any of the constituent learning algorithms alone.

- Random forest
- Gradient Boosting Machine (GBM):
  - XGBoost
  - lightGBM: by Microsoft

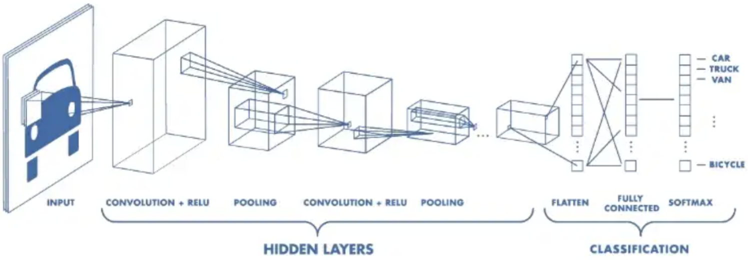
Generally, GBM > RF > DT.



An easy way to “feel” NN: `https://playground.tensorflow.org/`

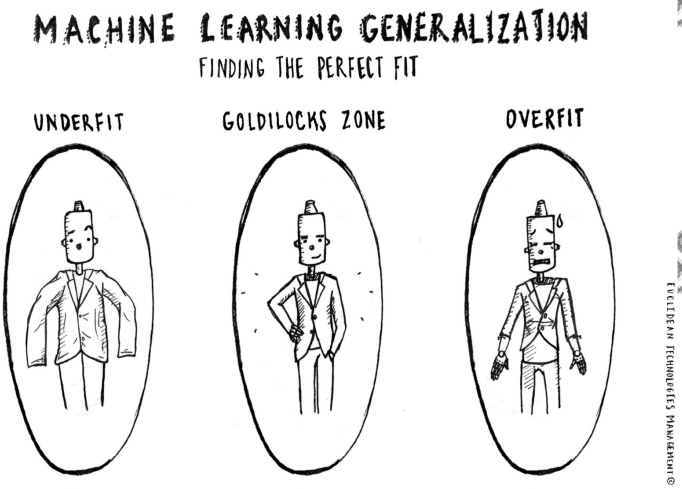
Convolutional Neural Networks (CNN): a specialized kind of Feed forward Neural Networks

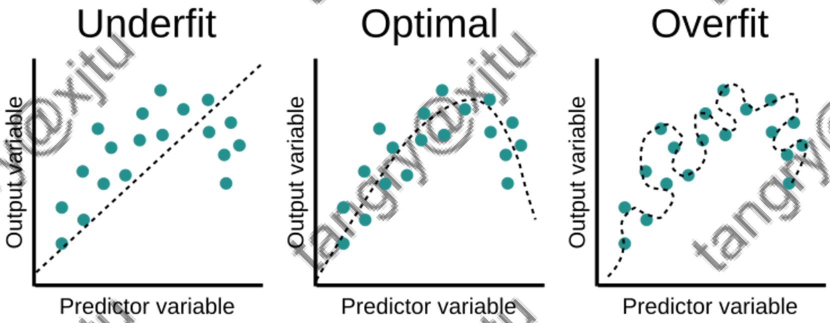
- Convolution Layer
- Pooling Layer
- ReLu (Rectified Linear Unit) $(max(0, x))$
- Sigmoid $(\frac{1}{1+e^{-x}})$



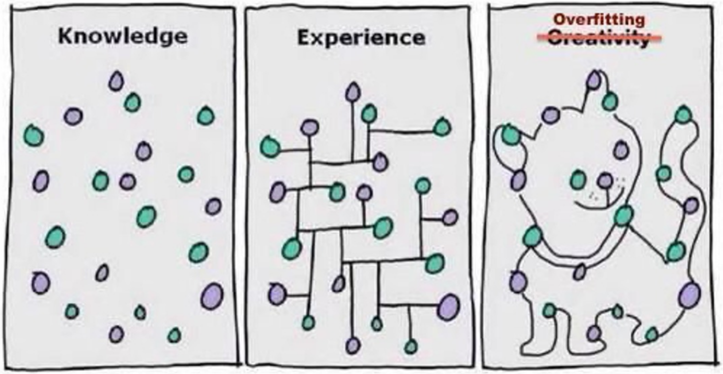
Modern CNN architectures:

- Network in Network
- Inception
- ResNet, ResNeXt
- ShuffleNet
- DenseNet
- CondenseNet
- SENet...


MACHINE LEARNING GENERALIZATION
FINDING THE PERFECT FIT
UNDERFIT
GOLDILOCKS ZONE
OVERFIT
EUCLIDEAN TECHNOLOGIES MANAGEMENT ©


Underfit
Optimal
Overfit
Output variable
Output variable
Output variable
Predictor variable
Predictor variable
Predictor variable

Overfitting



How to resolve overfitting?

- Reduce the number of features
- Increase the number of training samples
- Regularization
- Cross-validation

Questionare results

基本信息搜集(匿名)

第1题： 你未来想做学术吗？ [单选题]

选项	小计	比例
是， 而且我想做优化相关的研究工作	5	20.83%
是，但我不想做优化相关的工作，只想有所了解	7	29.17%
否，我想去企业做偏技术的工作	7	29.17%
否，我想考公务员（或者其他不太技术相关的工作）	2	8.33%
我还没想好	3	12.5%
本题有效填写人次	**24**	

表格 饼状 圆环 柱状 条形 折线

第2题： 你有学过优化和建模相关课程吗？ [多选题]

选项	小计	比例
线性规划	23	95.83%
凸优化	12	50%
整数规划与其他非线性优化	20	83.33%
动态规划	15	62.5%
鲁棒优化	0	0%
(空)	1	4.17%
本题有效填写人次	**24**	

查看多选题百分比计算方法

表格 饼状 圆环 柱状 条形 折线

Linear Regression:

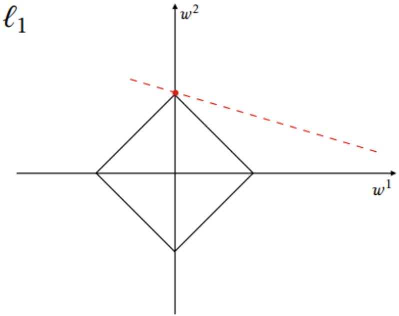
$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

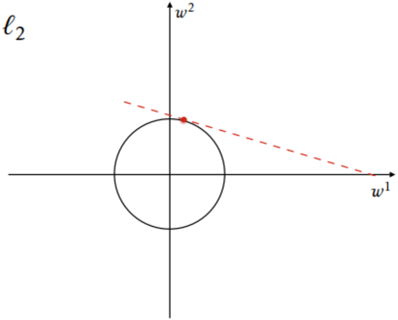
LASSO:

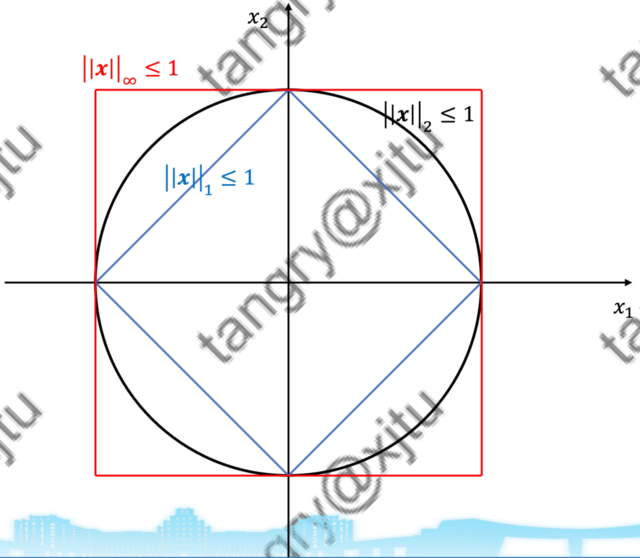
$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_1$$

Ridge:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda\|\mathbf{w}\|_2^2$$


ℓ_1
w^2
w^1


ℓ_2
w^2
w^1


x_2
$||\boldsymbol{x}||_\infty \leq 1$
$||\boldsymbol{x}||_2 \leq 1$
$||\boldsymbol{x}||_1 \leq 1$
x_1

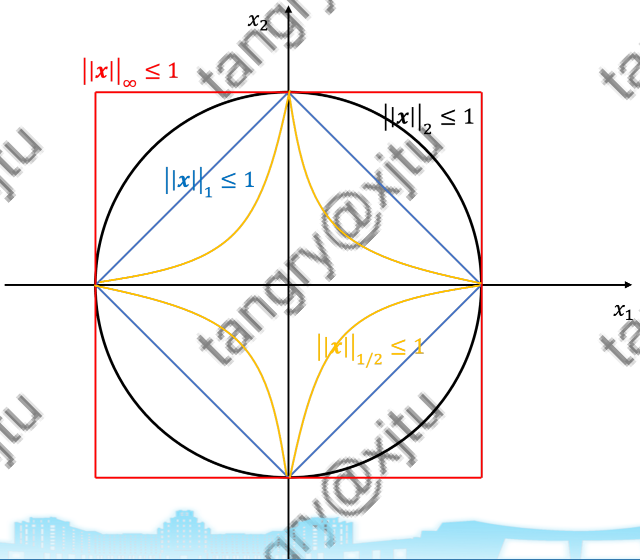
p–norm:

$$\|\mathbf{w}\|_p = (\sum_{i=1}^{p} |w_i|^p)^{1/p}, \text{ for } p \geq 1$$

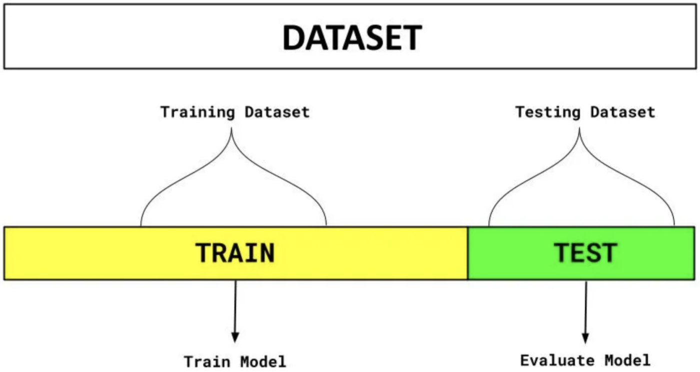
homogeneous	subadditive	positive definite
$\|k\boldsymbol{x}\| = \|k\|\|\boldsymbol{x}\|$	$\|\boldsymbol{x}+\boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$	$\|\boldsymbol{x}\| \geq 0$

Prof. Zongben Xu:

$L_{1/2}$ norm: finding sparse solutions.


x_2
$||\boldsymbol{x}||_\infty \leq 1$
$||\boldsymbol{x}||_2 \leq 1$
$||\boldsymbol{x}||_1 \leq 1$
x_1
$||\boldsymbol{x}||_{1/2} \leq 1$

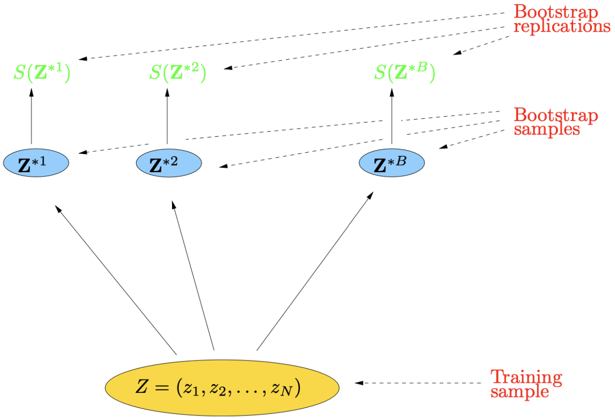
hold-out method: involves splitting the data into multiple parts and using one part for training the model and the rest for validating and testing it. (normally 70/30)



***K*-fold cross validation**: randomly divide the training set into K folds without replacement, then use fold k as the validation set and the union of the other $K-1$ folds as the training set. Repeat this process K times, and average the performance over the K folds.

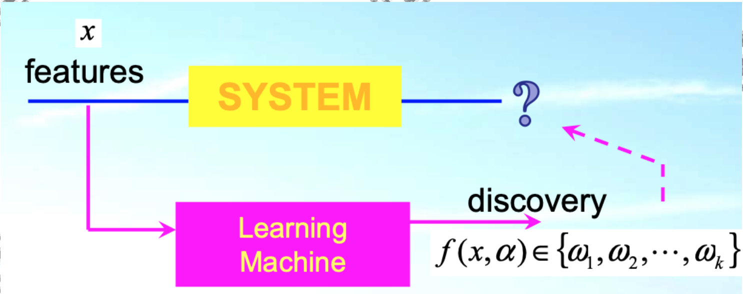
1	2	3	4	5
Train	Train	Validation	Train	Train

Bootstrap: randomly sample n observations with replacement from the training set to create a bootstrap sample. Then fit a model on the bootstrap sample and evaluate it on the out-of-bag observations. Repeat this process B times, and average the performance over the B bootstrap samples.



Unsupervised learning is a type of algorithm that learns patterns from untagged data.
Learning hidden structures of unlabeled data

- Unlabeled data
- No error or reward signal to evaluate a solution

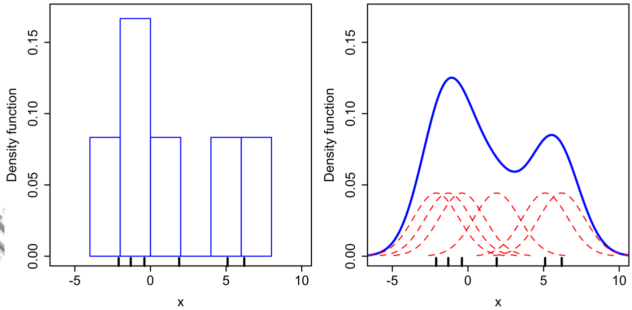


Major tasks

- Density estimation
- Association rules
- Cluster analysis
- Latent variable analysis

Suppose we have a random sample $x_1, \ldots, x_N$ drawn from a probability density $f_X(x)$, and we wish to estimate f_X at a point x_0.

- Local estimation
- Kernel Density Estimation



$$\hat{f}(x_0) = \frac{1}{N\lambda}\{\#x_i \in \mathcal{N}(x_0)\}$$

where $\mathcal{N}(x_0)$ is the neighborhood of x_0 of width λ.

With kernel:

$$\hat{f}(x_0) = \frac{1}{N\lambda}\sum_{i=1}^{N} K_\lambda(x_0, x_i)$$

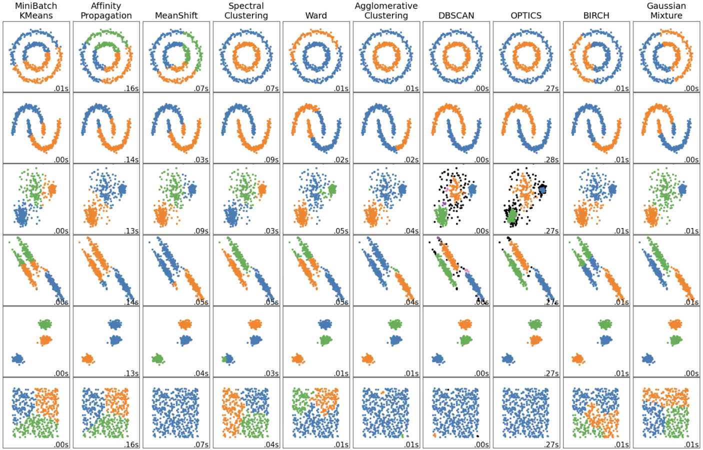
where $K_\lambda(x_0, x) = \phi(|x - x_0|/\lambda)$ and ϕ is the Gaussian density with mean zero and standard deviation λ.

$X \Rightarrow Y$ means that if an itemset contains X, then it is likely to also contain Y.
E.g. $X = \{\text{Beer}\}, Y = \{\text{Diaper}\}$

- $\text{Dsupport}(X \Rightarrow Y) = \frac{\text{Number of transactions containing both X and Y}}{\text{Total number of transactions}}$
- $\text{Dconfidence}(X \Rightarrow Y) = \frac{\text{Number of transactions containing both X and Y}}{\text{Number of transactions containing X}}$
- $X \Rightarrow Y$ is a strong rule if $\text{Dconfidence}(X \Rightarrow Y) > \beta$ and $\text{Dsupport}(X \Rightarrow Y) > \alpha$

Apriori algorithm: Market Basket Analysis

Clustering



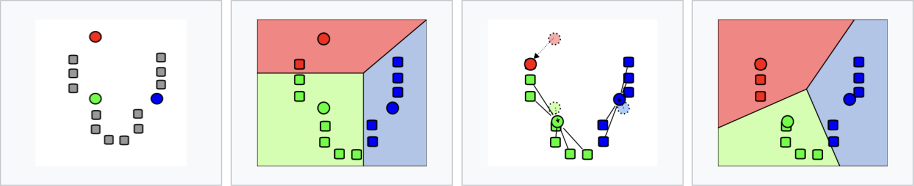
A comparison of the clustering algorithms in scikit-learn

https://scikit-learn.org/stable/modules/clustering.html

K-means clustering

K-means clustering: aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster.

Demonstration of the standard algorithm



1. *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).

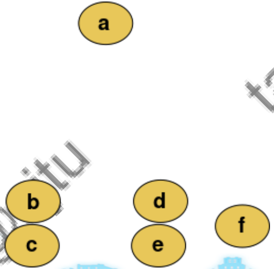
2. *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

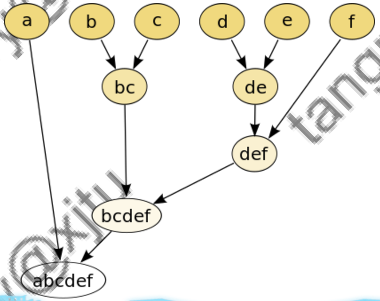
3. The centroid of each of the *k* clusters becomes the new mean.

4. Steps 2 and 3 are repeated until convergence has been reached.

- Agglomerative: bottom-up approach, each observation starts in its own cluster, and clusters are successively merged together.
- Divisive: top-down approach

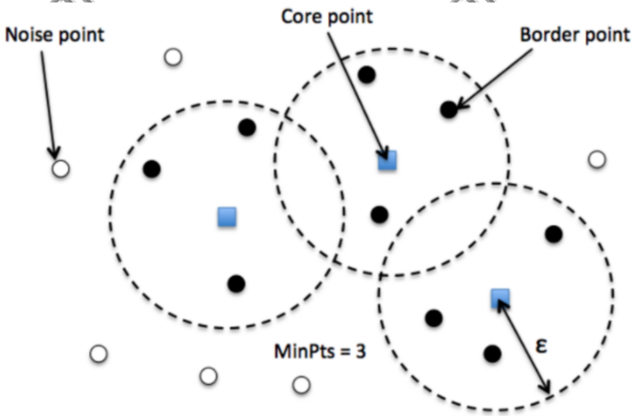
Dendrogram: a tree-like diagram used to illustrate the arrangement of the clusters produced by hierarchical clustering.





DBSCAN: Density-based Spatial Clustering of Applications with Noise. Given a set of points in some space, it groups together points that are closely packed together (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions (whose nearest neighbors are too far away)

- Find core samples of high density
- Expand clusters from core samples
- Assign noise samples to nearest cluster



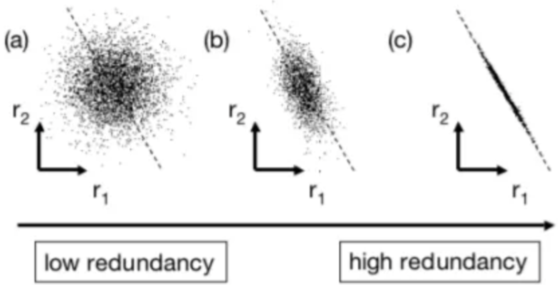
Pros & Cons

- Pros: arbitrary shape clusters, no need to specify number of clusters, robust to outliers
- Cons: sensitive to parameters, not suitable for large datasets

A visualization: `https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/`

Principal Component Analysis (PCA):

- Maximize the variance of the projected data
- Minimize the distance between the data and projections
- Equivalent to fine the eigenvectors corresponding to the largest eigenvalues of the covariance matrix



Multistage decision problems:

- DP: Dynamic programming
- MDP: Markov Decision Process
- RL: Reinforcement learning

The main difference between the classical dynamic programming methods and reinforcement learning algorithms is that the latter do not assume knowledge of an exact mathematical model of the MDP and they target large MDPs where exact methods become infeasible.

- state: the current state of the environment $s_t \in \mathcal{S}$
- action: the action taken by the agent $a_t \in \mathcal{A}$.
- reward: the reward received by the agent $r(s_t, a_t)$
- state transition: the transition from the current state to the next state. $s_{t+1} = f(s_t, a_t)$

Value function

$$V^\pi(s) = \mathbb{E}_\pi\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \tag{1}$$

where π is the policy, γ is the discount factor.

For the finite-horizon problem:
Backward induction:

$$V_t(s_t) = \max_{a_t} \mathbb{E}_\pi \left[r(s_t, a_t) + \gamma V_{t+1}(s_{t+1}) \right]$$

For the infinite horizon problem: $T \to \infty$:
Bellman equation

$$V^*(s) = \max_{\pi} \mathbb{E}_\pi \left[r(s, a) + \gamma V^*(f(s, a)) \right]$$

We can write is as

$$TV^* = V^*$$

where the operator T satisfies $TV(s) = \max_\pi \mathbb{E}_\pi \left[r(s, a) + \gamma V(f(s, a)) \right]$.

Fixed point theory

- Monotonicity: T is monotone if $TV_1 \le TV_2$ whenever $V_1 \le V_2$.
- Contraction mapping: T is a contraction mapping if $||TV_1 - TV_2|| \le \gamma ||V_1 - V_2||$ for all V_1, V_2.

see **Bertsekas**, Abstract dynamic programming 2013, page 7 & 8 for details.

Algorithms for solving the infinite horizon problem.

- Policy iteration: we start by choosing an arbitrary policy $\boldsymbol{\pi}$. Then, we iteratively evaluate and improve the policy until convergence:
- Value iteration: In value iteration, we compute the optimal state value function by iteratively updating the estimate $\mathbf{V(s)}$:

In general, policy iteration is more efficient (fewer iterations) than value iteration, but it requires more memory.

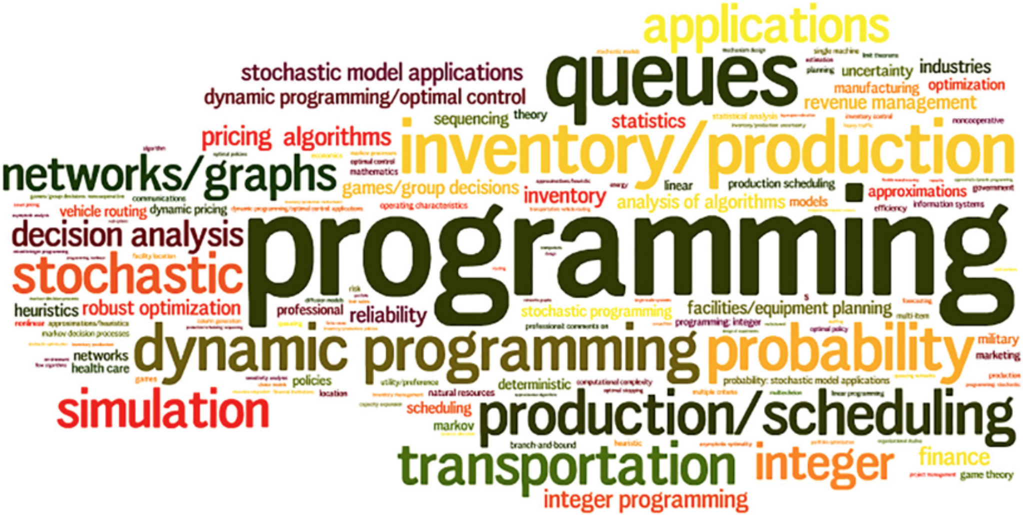
Top five research method published by OR since 1981:

- Math programming
- Queueing
- Dynamic programming
- Simulation
- Game theory

In the past 10 years (2010-2019), dynamic programming has the strongest community, followed by pricing.

Angelito Calma , William Ho , Lusheng Shao , Huashan Li (2021) Operations Research: Topics, Impact, and Trends from 1952-2019. Operations Research 69(5):1487-1508.

Figure 2. (Color online) Author Keywords Tag Cloud



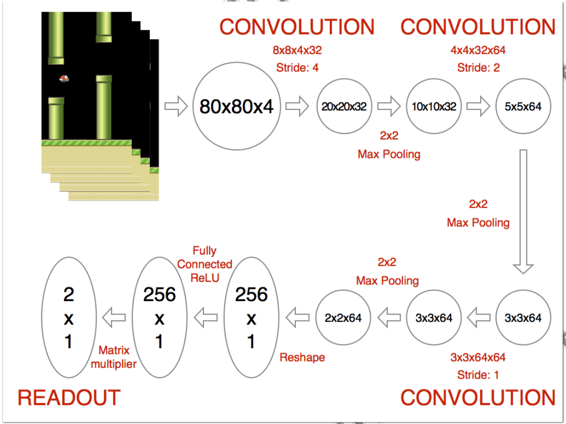
- Q-learning is a model-free reinforcement learning algorithm
- It can be used to find the optimal policy for a given MDP
- It is an off-policy algorithm

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{current value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Bigg(\underbrace{\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}}}_{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{current value}} \Bigg)}^{\text{temporal difference}}$$

DQN: Deep Q-Network: a Q-Learning framework with a neural network.

- The network is trained to predict the Q-value of each action given the current state

```
Initialize replay memory D to size N
Initialize action-value function Q with random weights
for episode = 1, M do
        Initialize state s_1
        for t = 1, T do
                With probability ϵ select random action a_t
                otherwise select a_t=argmax_a  Q(s_t,a; θ_i)
                Execute action a_t in emulator and observe r_t and s_(t+1)
                Store transition (s_t,a_t,r_t,s_(t+1)) in D
                Sample a minibatch of transitions (s_j,a_j,r_j,s_(j+1)) from D
                Set y_j:=
                        r_j for terminal s_(j+1)
                        r_j+γ*max_(a^' )  Q(s_(j+1),a'; θ_i) for non-terminal s_(j+1)
                Perform a gradient step on (y_j-Q(s_j,a_j; θ_i))^2 with respect to θ
        end for
end for
```



The values at the output layer represent the Q function given the input state for each valid action. `https://github.com/yenchenlin/DeepLearningFlappyBird`

- Hastie, T., Tibshirani, R. & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*. New York: springer.
- Dimitri P. Bertsekas. (2018) *Reinforcement Learning and Optimal Control.* Athena Scientific.
- Zhihua Zhou.(2015) *Machine Learning*. Tsinghua University Press.

- Zhiwei (Tony) Qin, Xiaocheng Tang, Yan Jiao, Fan Zhang, Zhe Xu, Hongtu Zhu, Jieping Ye (2020) Ride-Hailing Order Dispatching at DiDi via Reinforcement Learning. *INFORMS Journal on Applied Analytics* 50(5):272-286.
- J. G. Dai, Mark Gluzman (2021) Queueing Network Controls via Deep Reinforcement Learning. *Stochastic Systems* 12(1):30-67.
- Nooshin Salari, Sheng Liu, Zuo-Jun Max Shen (2022) Real-Time Delivery Time Forecasting and Promising in Online Retailing: When Will Your Package Arrive?. *Manufacturing & Service Operations Management* 24(3):1421-1436.
- Meng Qi, Yuanyuan Shi, Yongzhi Qi, Chenxin Ma, Rong Yuan, Di Wu, Zuo-Jun (Max) Shen (2022) A Practical End-to-End Inventory Management Model with Deep Learning. *Management Science* 0(0).