Pattern Recognition

Introduction

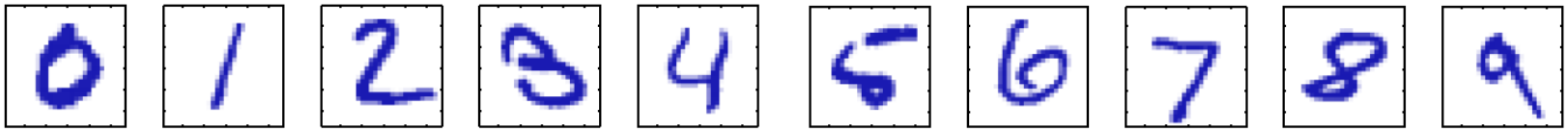
林彥宇 教授
Yen-Yu Lin, Professor

國立陽明交通大學 資訊工程學系
Computer Science, National Yang Ming Chiao Tung University

Some slides are modified from Prof. Sheng-Jyh Wang, Prof. Hwang-Tzong Chen, and Prof. Yung-Yu Chuang

Pattern recognition

- Pattern recognition is the automated recognition of patterns and regularities in data
  - Discover pattern regularities
  - Take actions, such as classification or regression, with regularities
- Data: A set of hand-written digits and the class ground truth



- Computer algorithm: It extracts features from each image, analyze the patterns and regularities in data
- Model: Given a new hand-written digit, predict its class label

Pattern recognition and its related fields

- Pattern recognition is closely related to machine learning and artificial intelligence
- Machine learning: To design and develop algorithms that allow computers to predict data based on empirical data
- Artificial intelligence: To build machines capable of performing tasks that typically require human intelligence

Pattern recognition vs. Machine learning

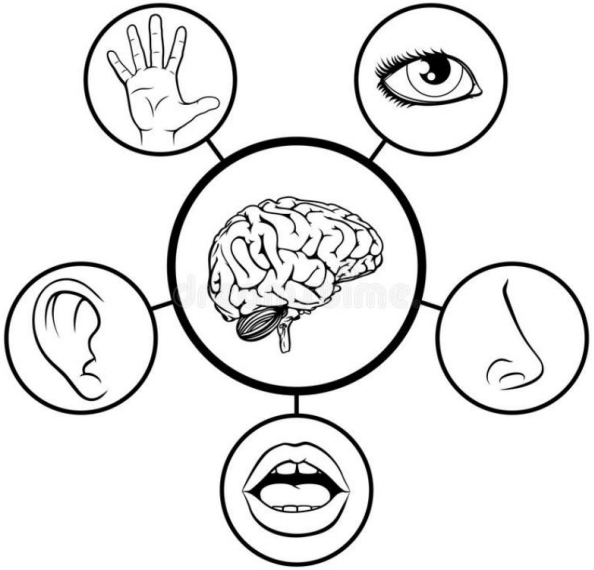
- Machine learning: to design and develop algorithms that allow computers to predict data based on empirical data
  - Try to explore certain patterns or regularities
  - Learn models from the given data
  - Based on the given data, the learner produces a useful output in new cases

- Machine learning is one approach to pattern recognition, while other approaches include hand-crafted (not learned) rules or heuristics

- Machine learning $\subset$ Pattern recognition

Pattern recognition vs. Artificial intelligence

- Artificial intelligence: To build machines capable of performing tasks that typically require human intelligence
- Pattern recognition is one approach to artificial intelligence, while other approaches include symbolic AI
- Pattern recognition ⊂ Artificial intelligence



Applications of pattern recognition

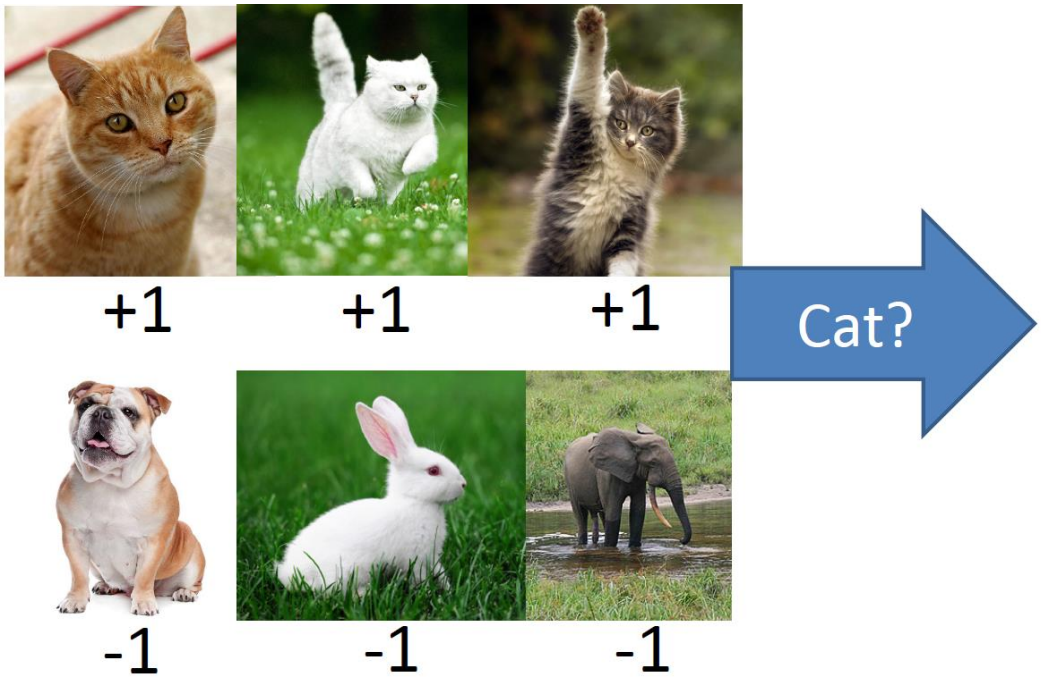
- Computer vision
- Speech recognition
- Search engine
- Natural language processing
- Robotics
- Bioinformatics
- Data Mining
- Finance
- …

Problem definition of a PR task

- Training data
  - A set of N training data $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$, sometimes together with their target vectors $\{t_1, t_2, \ldots, t_N\}$, is provided for a PR task
- Feature extraction
  - Original input variables are usually transformed into some new space of variables, where the problem can be better handled
- Model learning
  - We learn a proper model for the problem
- Generalization or testing
  - To correctly predict new examples (testing data) that differ from those used for training

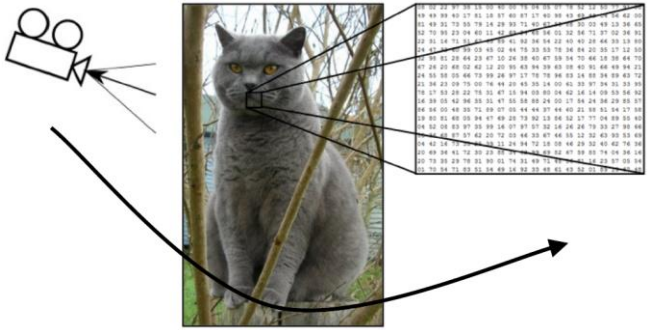
Cat image classification: Training data

- Collect a set of training data with target vectors



Cat image classification: Feature extraction

- Feature extraction is crucial
  - Need to take feature variations into account



viewpoint variations

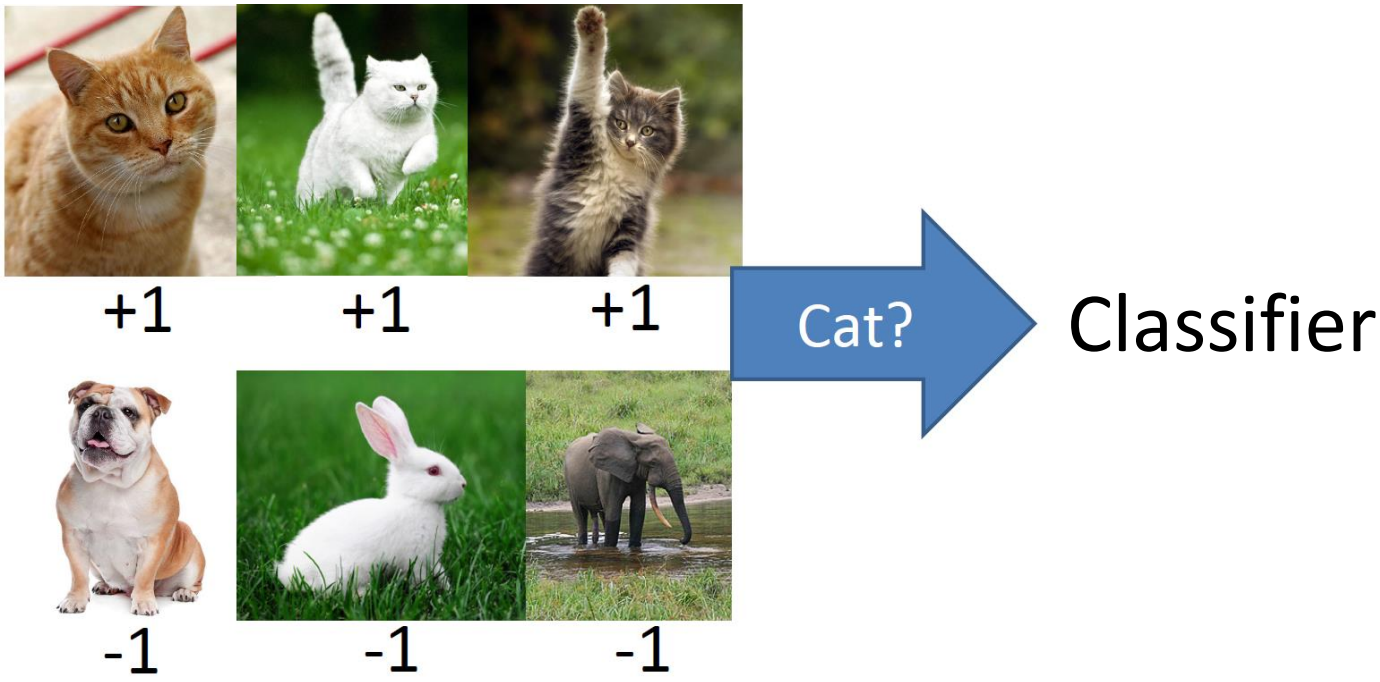
Illumination variations

background variations

pose variations

Cat image classification: Model learning

- Based on the given training data and the extracted features, we learn a classifier



Cat image classification: Testing

- Apply the learned classifier to the testing images

Cat image classification: Testing

- Apply the learned classifier to the testing images and make prediction

Supervised vs. Unsupervised learning

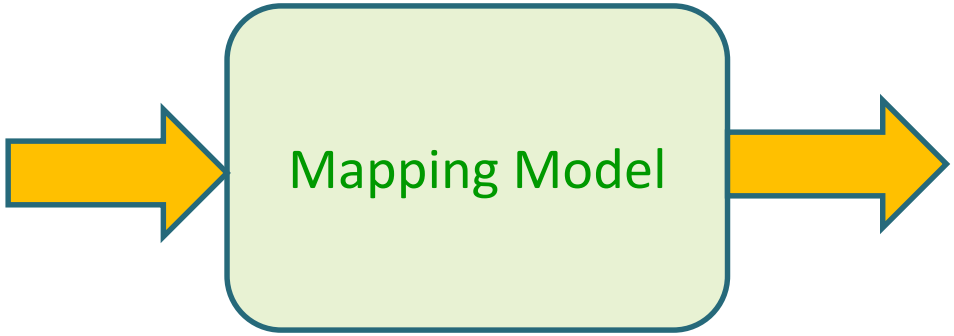
- Supervised learning: the training data comprises examples of the input vectors along with their corresponding target vectors
  - Classification: assign each input vector to one of a finite number of discrete categories
  - Regression: assign each input vector to one or more continuous variables
  - Methods: linear regression, linear classification, neural networks, support vector machine, ensemble learning, dimensionality reduction, deep learning, …

Supervised learning for classification

A finite number of labels

x

input data



C_k

predicted label

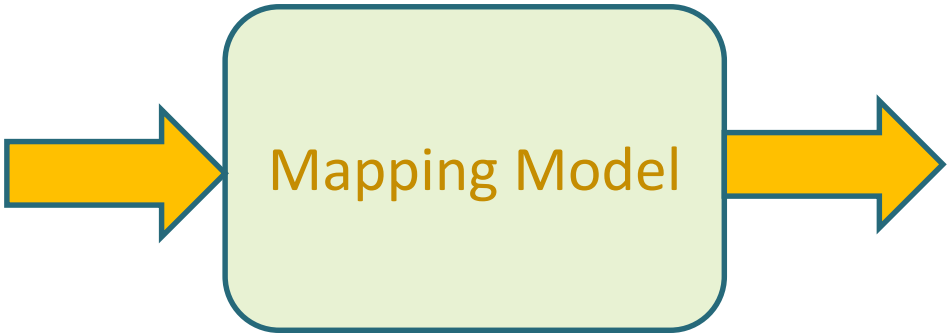
flower

Supervised learning for regression

A large (or infinite) number of values

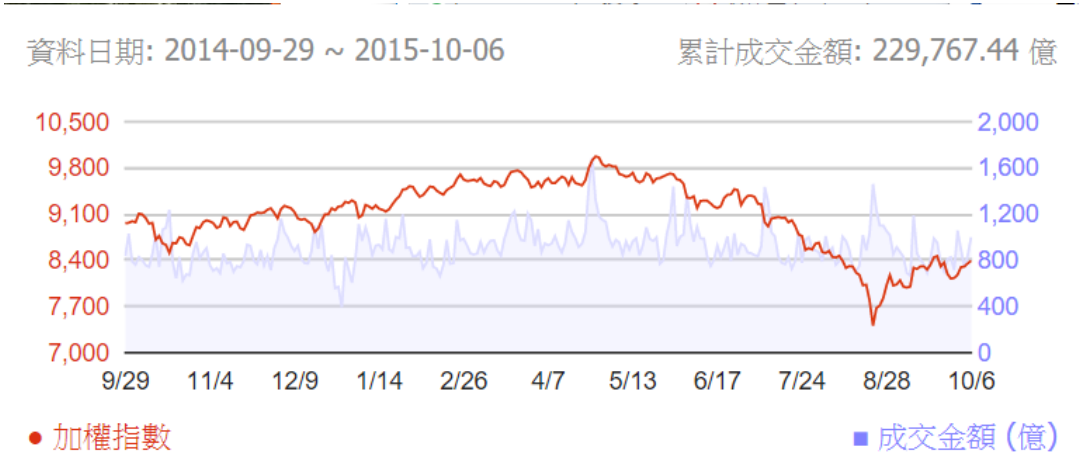
x

input data



y

predicted value



estimated TAIEX on 11/5

Taiwan Capitalization Weighted Stock Index

Training data collection

Input | **Expected Label**

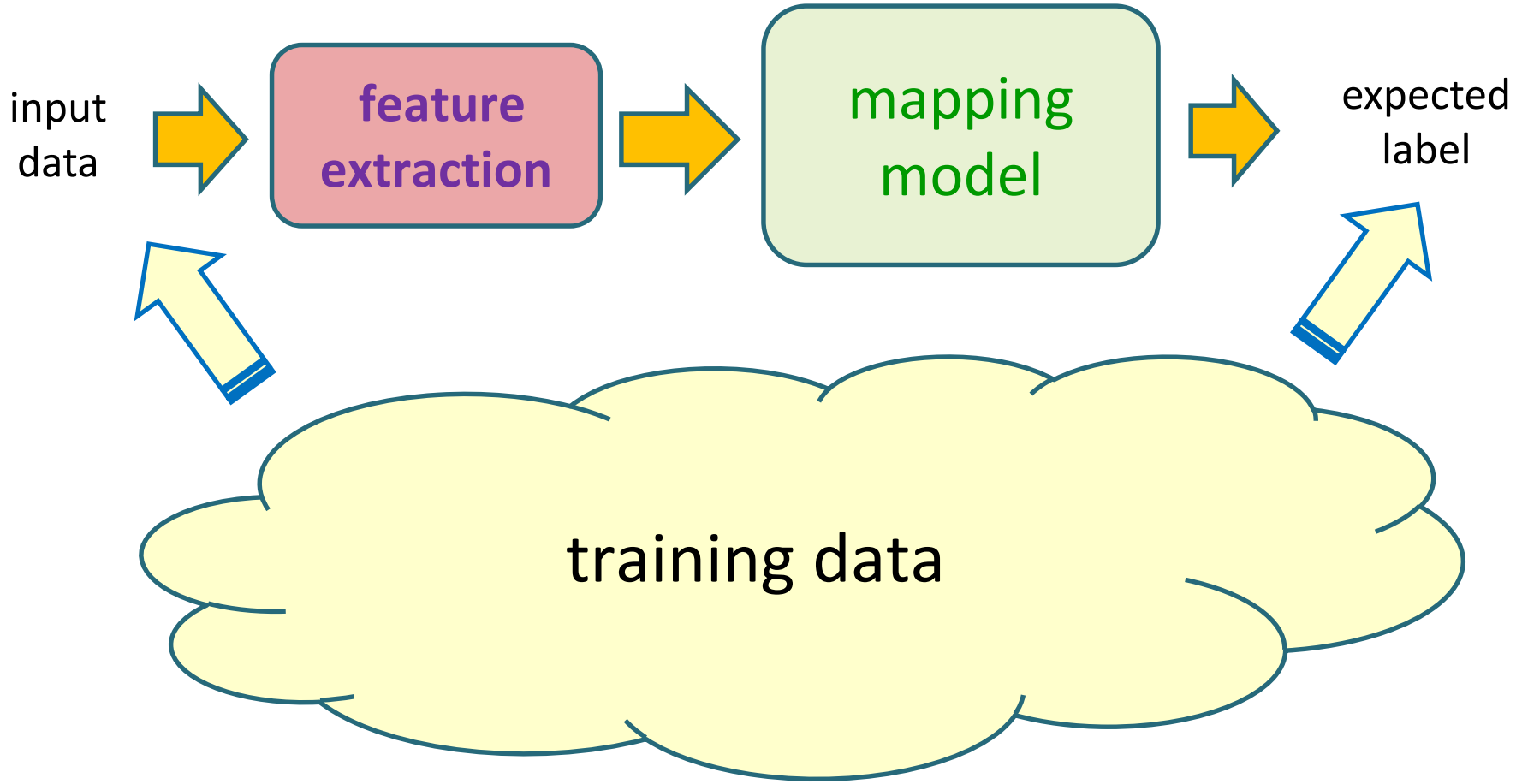
Palm  ... C_1

Flower  ... C_2

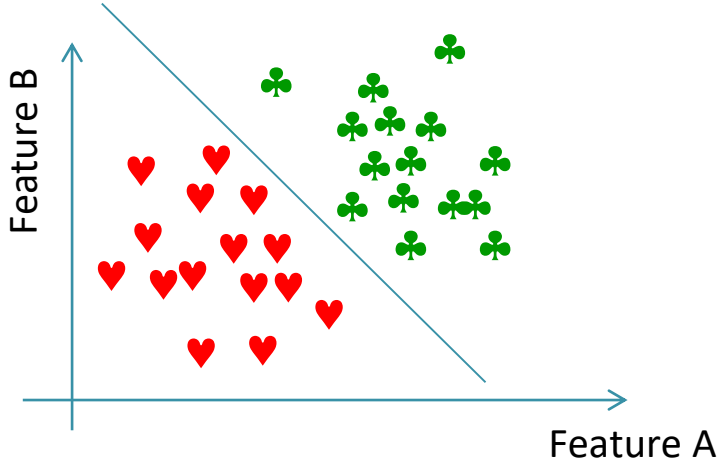
⋮

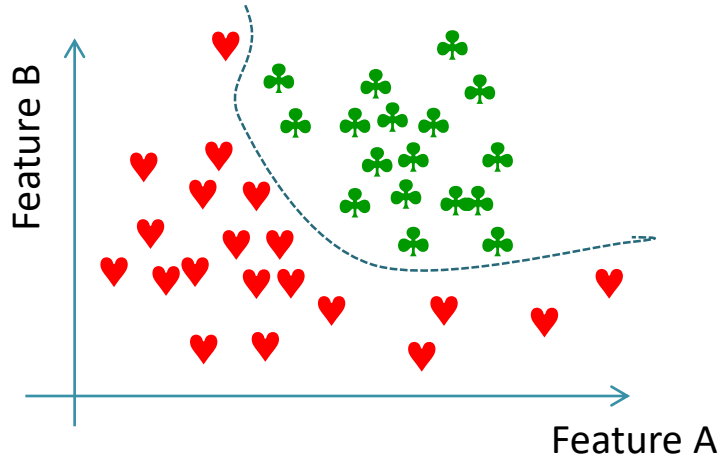
Else  ... C_{N+1}

Feature extraction and model learning

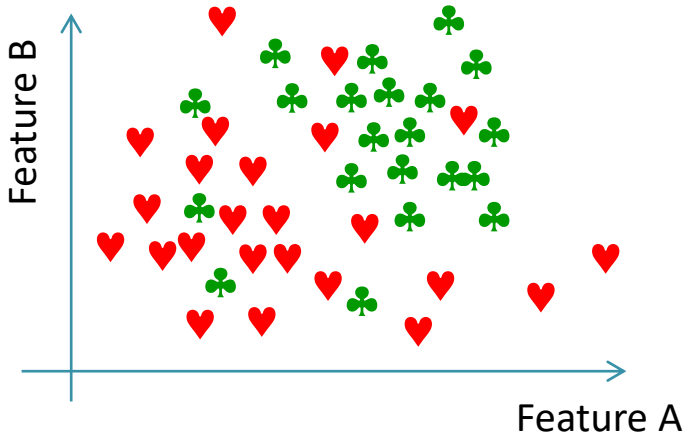


Good vs. bad features for classification



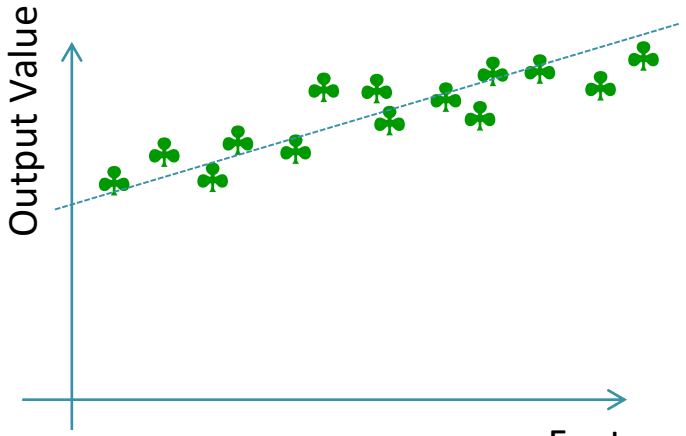


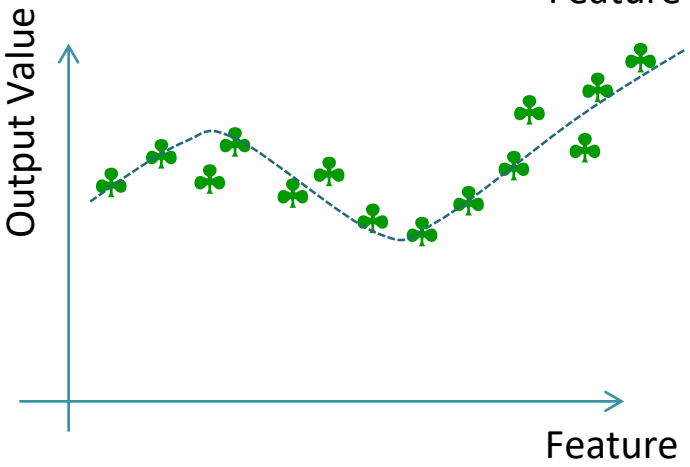
good features



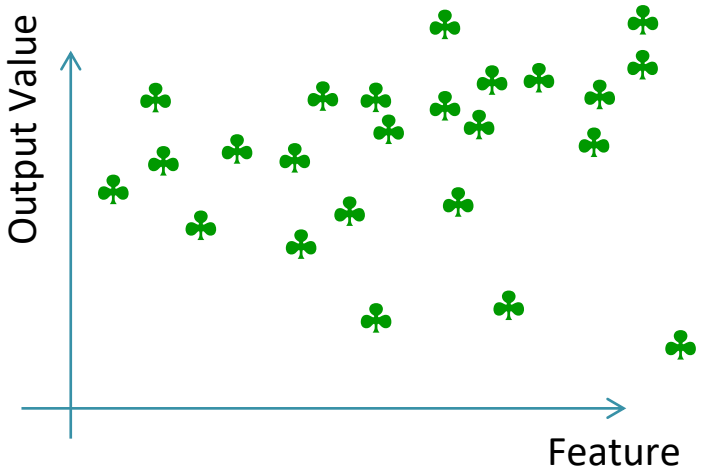
bad features

Good vs. bad features for regression



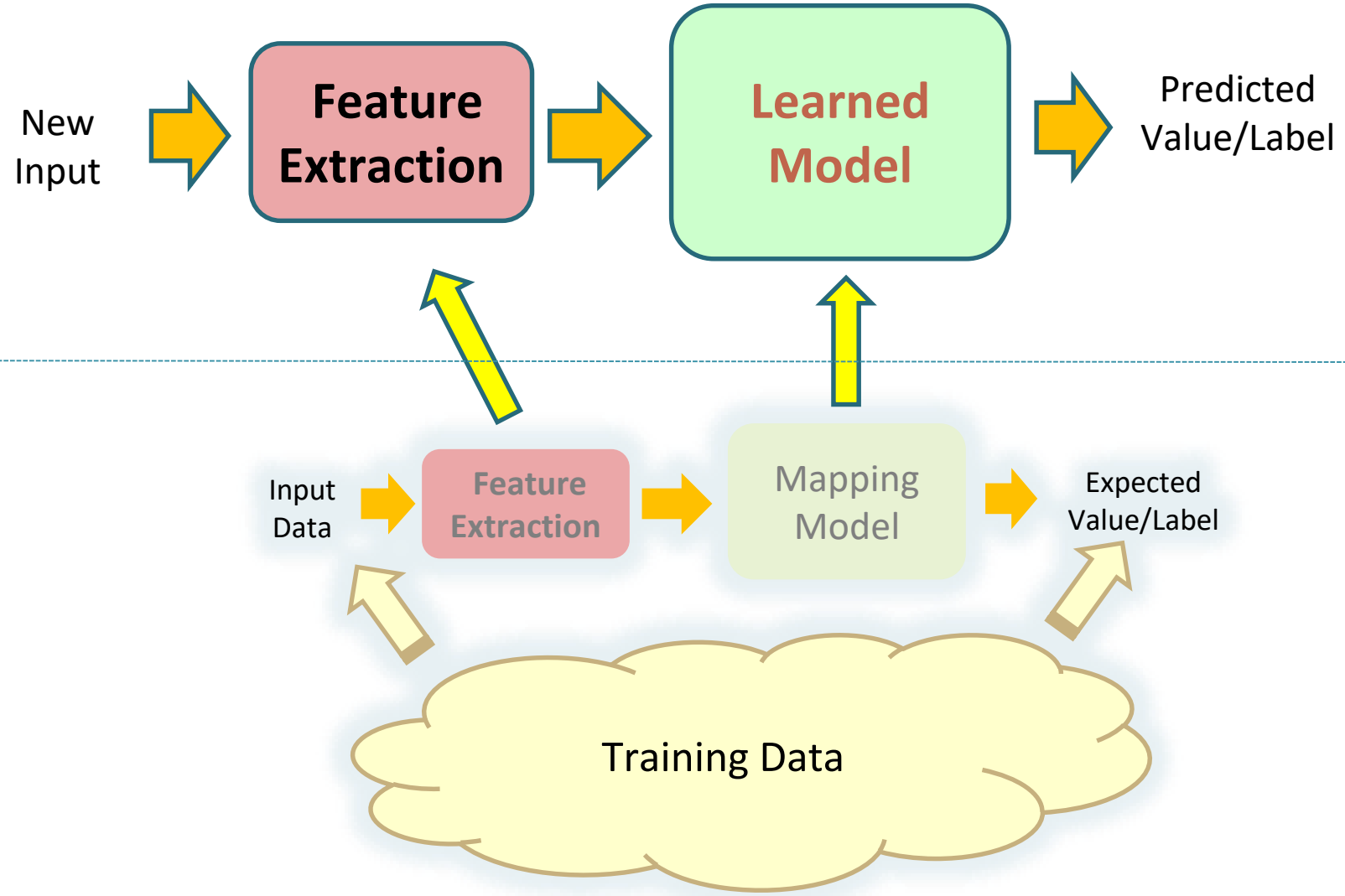


good feature



bad feature

Testing

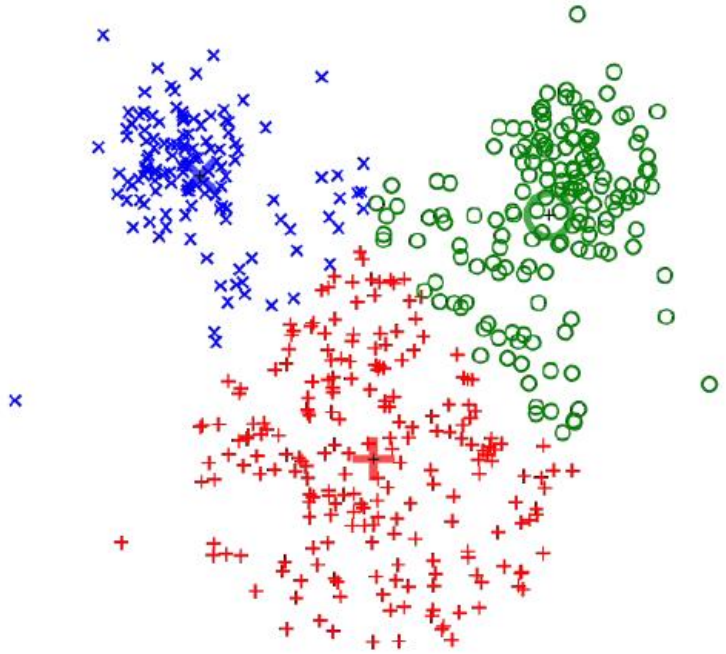

New Input
Feature Extraction
Learned Model
Predicted Value/Label
Input Data
Feature Extraction
Mapping Model
Expected Value/Label
Training Data

Supervised vs. Unsupervised learning

- Unsupervised learning: the training data consists of a set of input vectors x without any corresponding target values
  - Clustering: to discover groups of similar examples within the data
  - Density estimation: to determine the distribution of data within the input space
  - Dimensionality reduction: to project the data from a high-dimensional space down to a low-dimensional space

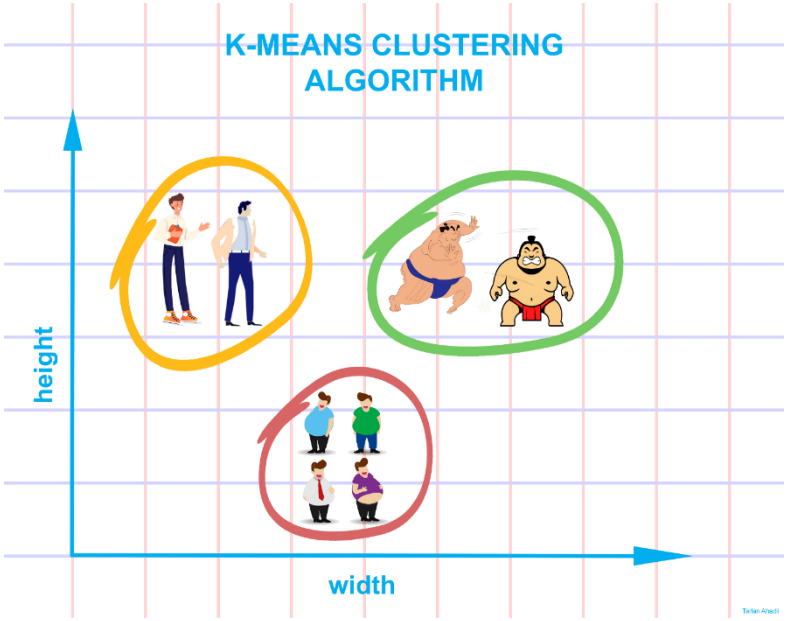
Unsupervised learning for clustering

- Clustering: To group a set of data in such a way that data points in the same group, called a **cluster**, are more similar to each other than to those in other clusters

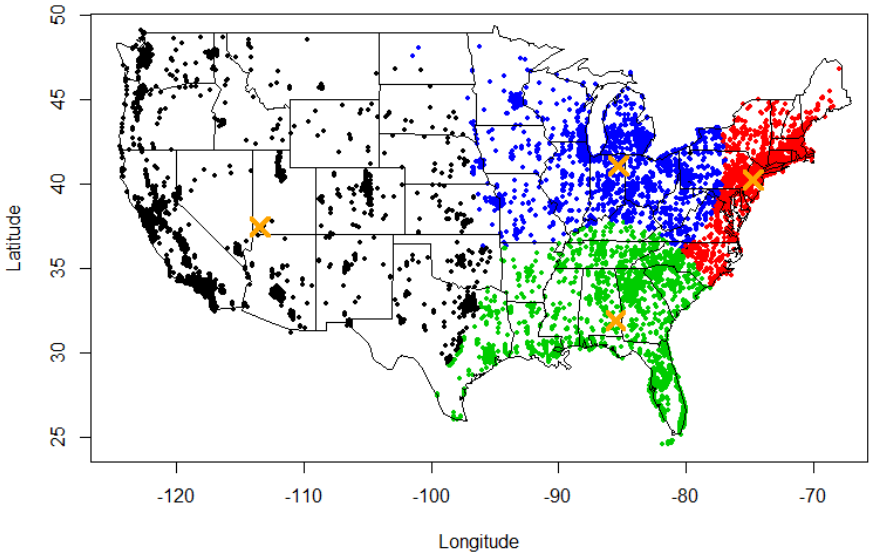


***k*-mean clustering**

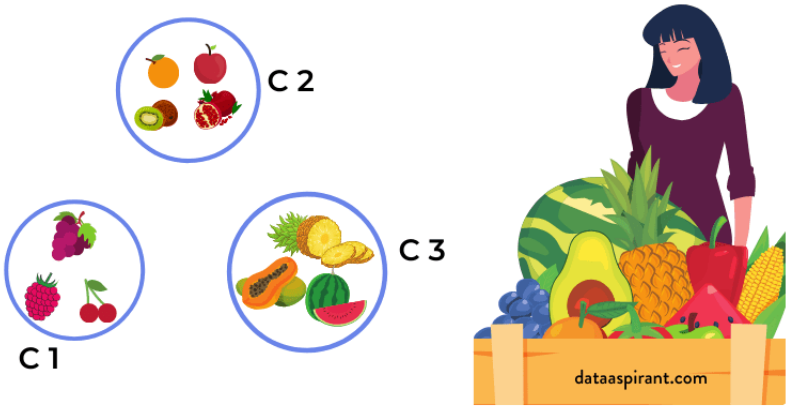
Examples of clustering



https://medium.com/@tarlanahad/a-friendly-introduction-to-k-means-clustering-algorithm-b31ff7df7ef1



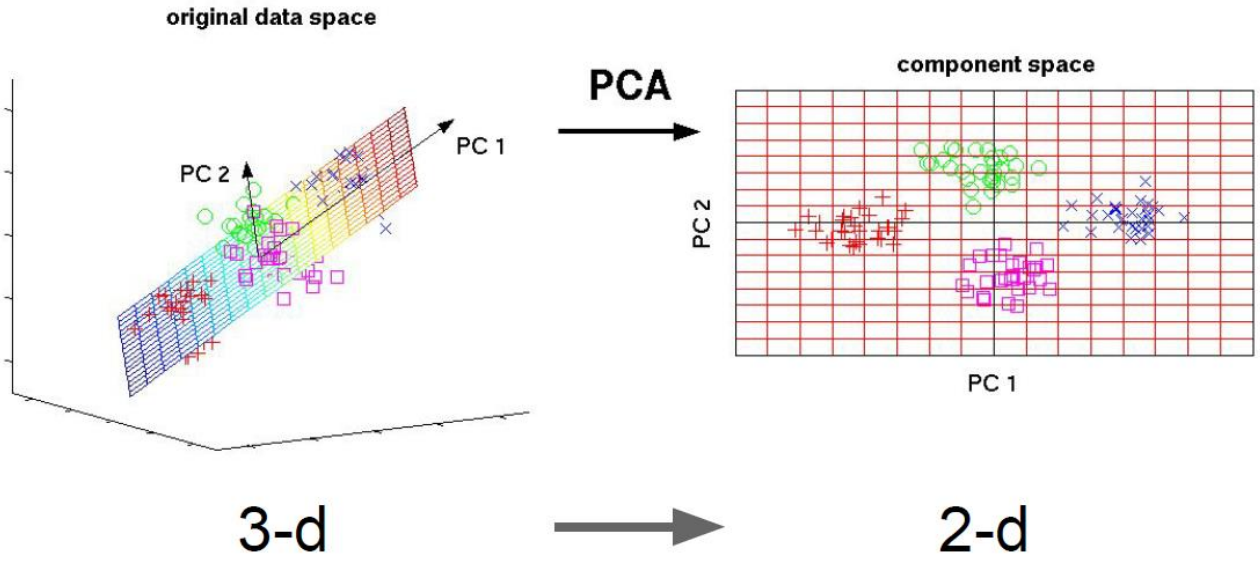
https://sites.wustl.edu/neumann/courses/cse427s/final-project/project-3/



https://dataaspirant.com/k-means-clustering-algorithm/

Unsupervised learning for dimensionality reduction

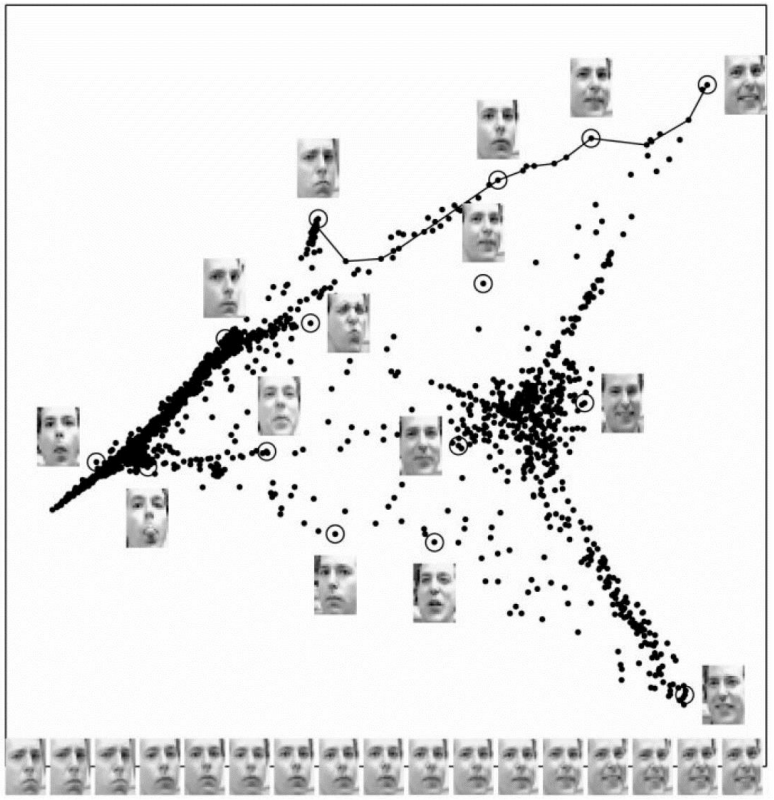
- Dimensionality reduction: To project data from a high-dimensional space to a low-dimensional one



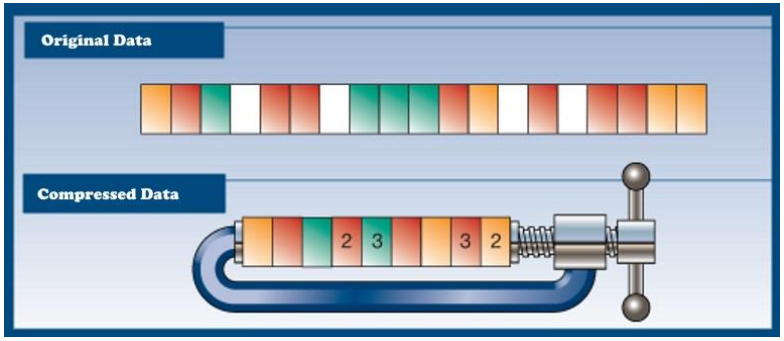
PCA: Principal component analysis

More examples

- Face image analysis
  - LLE (Locally linear embedding)
- High-dimensional input
- 2D representation
  - Expression analysis (x-axis)
  - Pose analysis (y-axis)



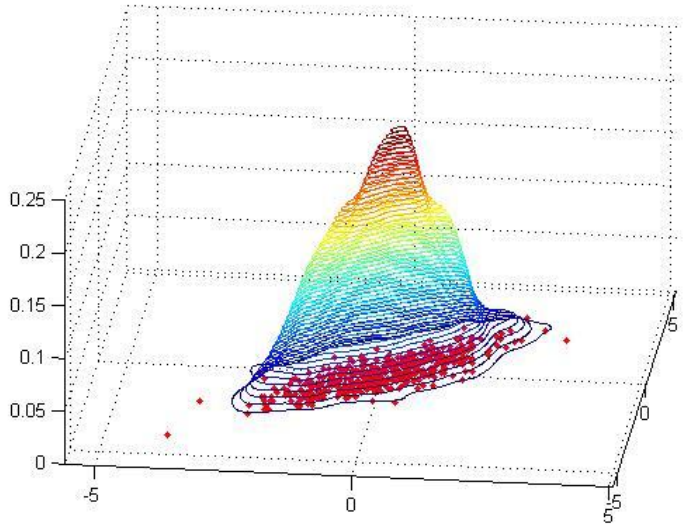
- Compression
  - Data storage
  - Data transmission



https://terabytegarage.com/linux/analysis-compression-options-in-linux/

Unsupervised learning for density estimation

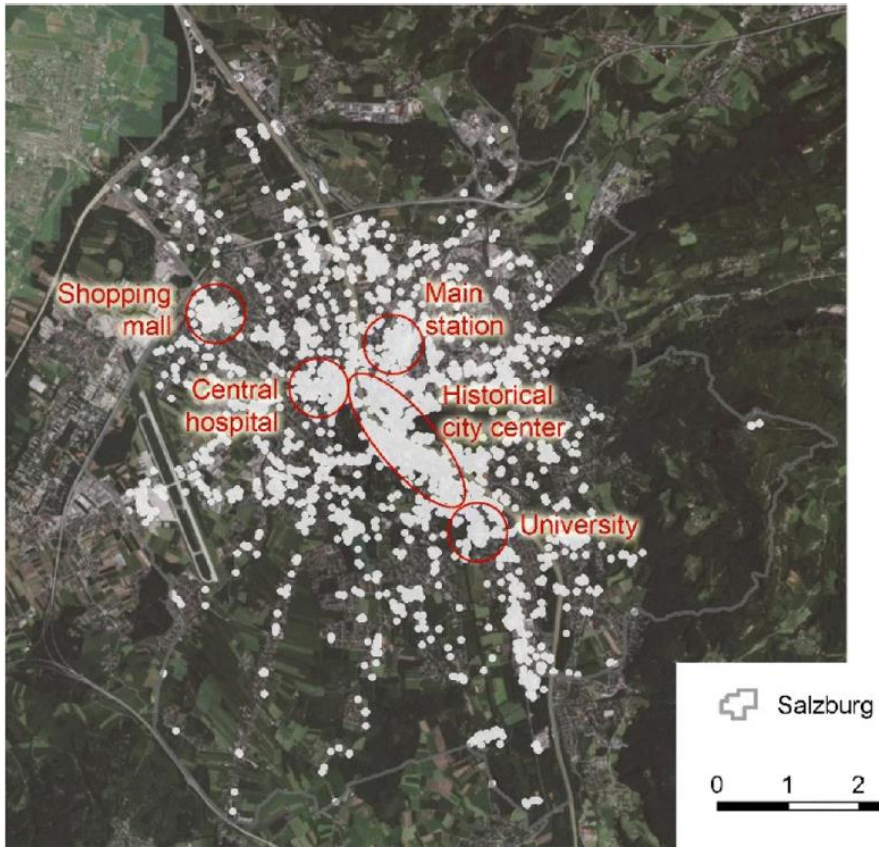
- Density estimation: Based on given data, estimate the underlying probability density function



kernel density estimation (KDE)

Application

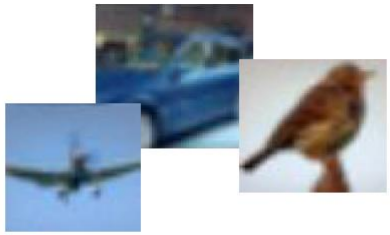
- Station-based bike sharing system planning



Loild et al. ETRR 2019

Unsupervised learning for data generation

- Given a set of natural images, we try to generate new images that look natural and photorealistic



Training data ~ $p_{data}(x)$

Generated samples ~ $p_{model}(x)$

Generative Adversarial Networks (GAN): Given a set of images, generate new images from the same distributions

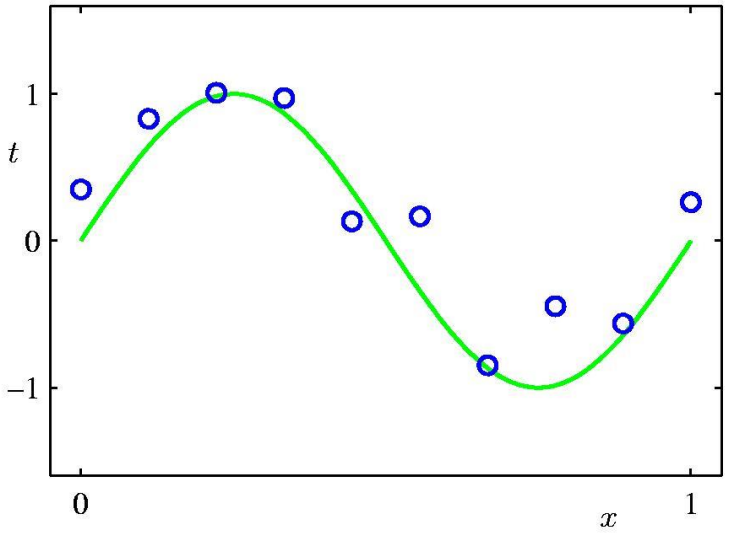
Applications

- Face generation

Tero Karras et al. “Progressive Growing of GANs for Improved Quality, Stability, and Variation”

Polynomial curve fitting: Problem definition

- Training data (observations)
  - 10 blue circles, each of which has
    - One-dimensional input
    - One target output
- Green curve $\sin(2\pi x)$ is the function used to generate these data, which is unknown



- Each point is sampled from the function with a random Gaussian noise
- Goal of curve fitting: To exploit the training data to discover the underlying function so that we can make predictions of the value $\hat{t}$ for some new input $\hat{x}$

Polynomial curve fitting: Choose a fitting function

- Fit the data using a polynomial function of the form:

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

  - This function is parametrized by **w**
  - w_0 is the bias term
  - Its input is a data point, while the output is estimated target
  - *M* is the order of the polynomial function

Polynomial curve fitting: Error function

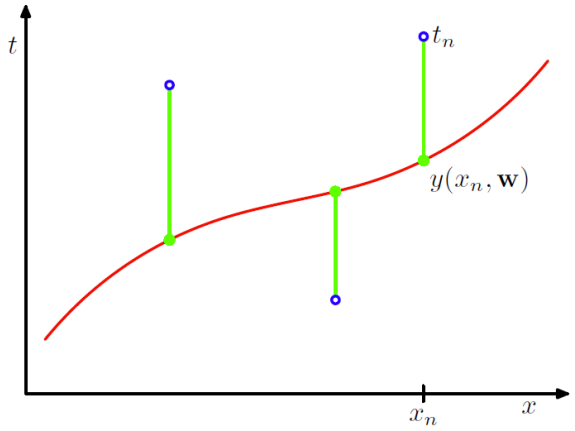
- An error function (objective function) is used to determine the parameters

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- In this case, we minimize the sum-of-squares error

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

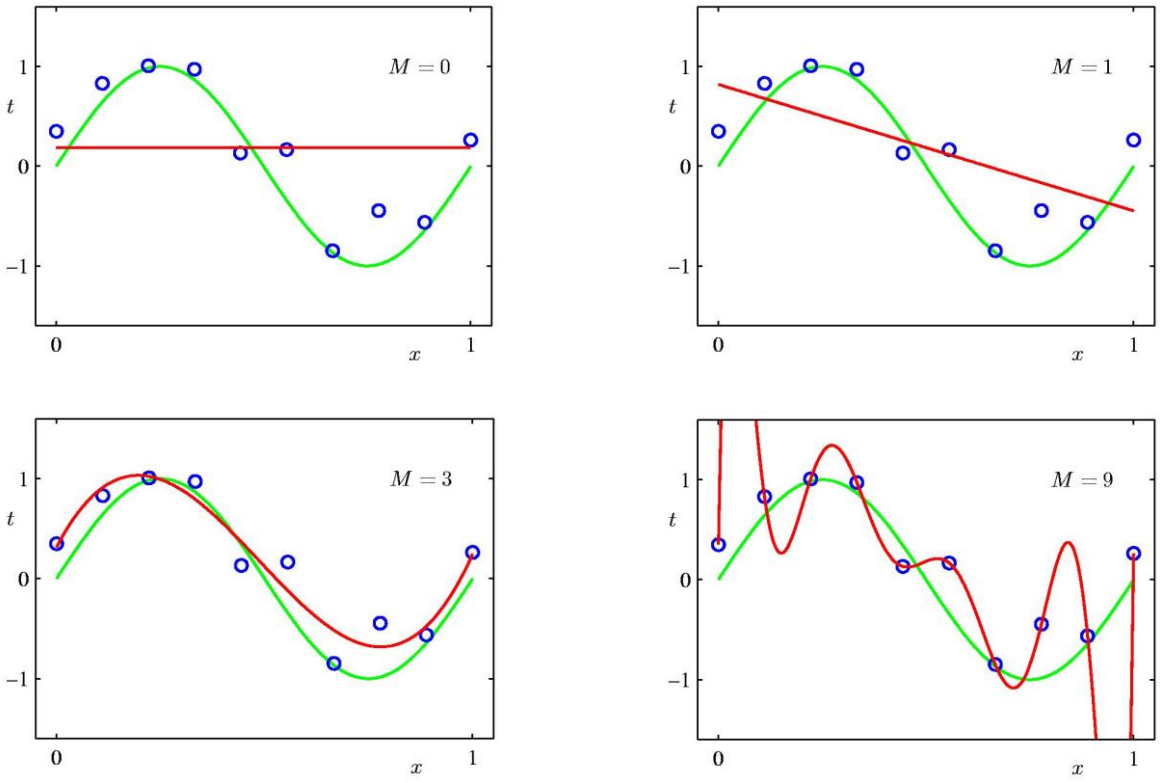
  - Differentiable
  - Closed form solution



Polynomial curve fitting: Model selection

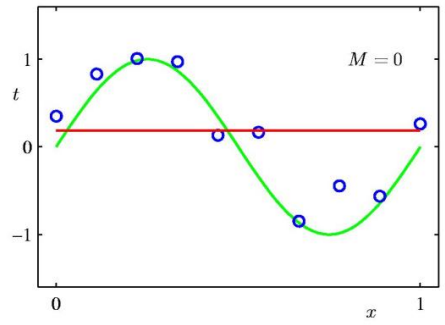
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

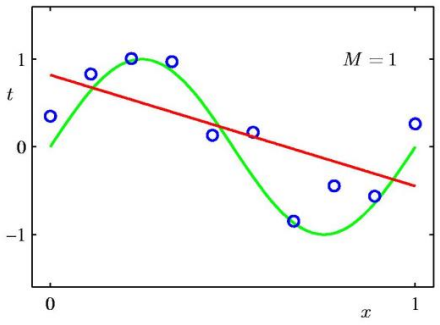
- Models with different values of hyperparameter *M*

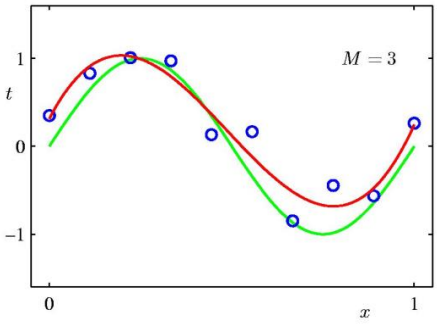


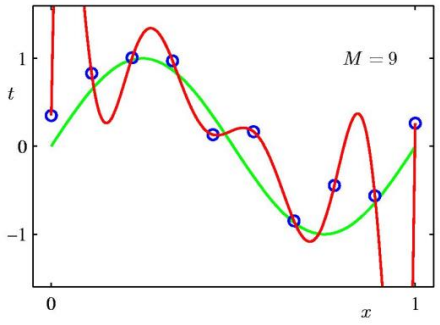
- Model selection: To choose a proper value of *M*

Polynomial curve fitting: Model selection







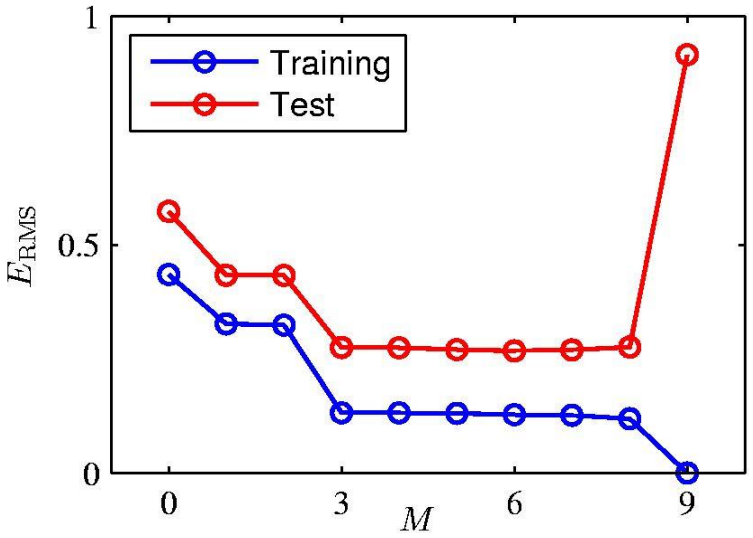


- Under-fitting: $M = 0$ or $M = 1$
  - The constant or first order polynomial gives poor fit due to insufficient flexibility
- The third order polynomial gives the best fit
- Over-fitting: $M = 9$
  - Each training points are perfectly fitted
  - Poor representation of the green curve
  - The generalization is poor

Polynomial curve fitting: Generalization

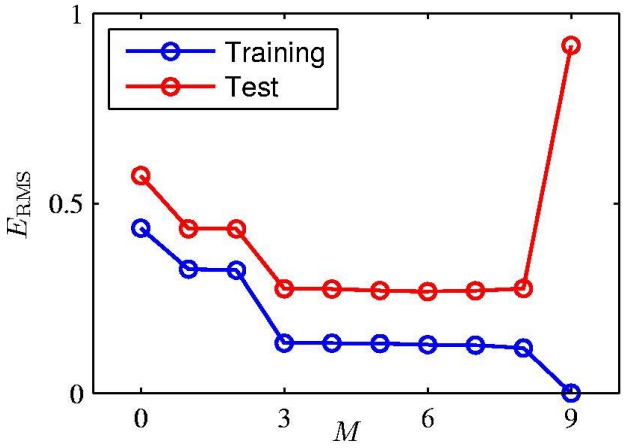
- Suppose we are given a set of training data and a separate set of 100 test data
- Evaluate the generalization for each choice of *M* via root-mean-square (RMS) error

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$



	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

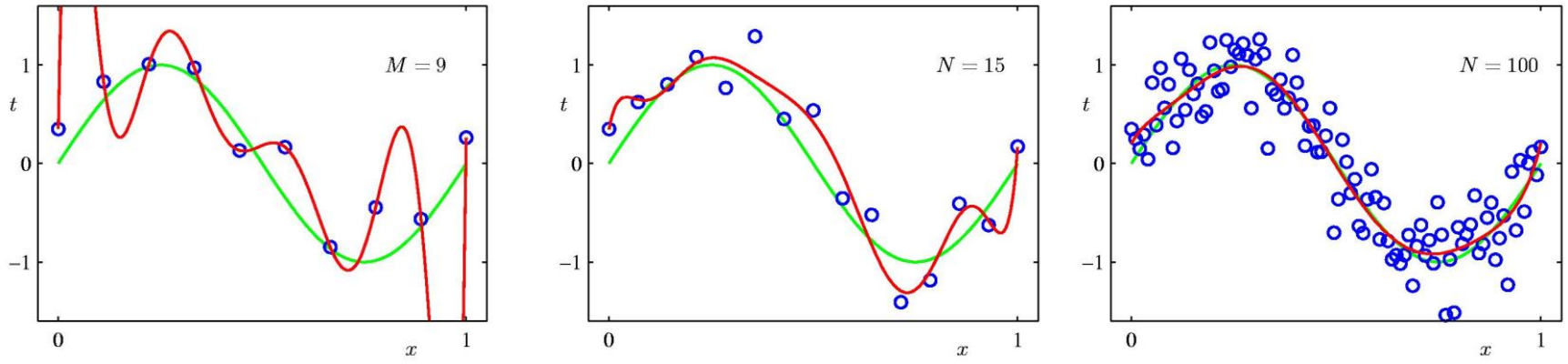
Polynomial curve fitting: Generalization



- Small values of *M* give relatively large values of training and test errors
- When *M* is between 3 and 8, reasonable representations are obtained
- For *M*=9, the training error goes to zero, but the test error increases significantly

Polynomial curve fitting: Data size vs. Over-fitting

$M = 9$



- Over-fitting becomes less severe as the data size increases
- In general, the number of data points should be no less than some multiple (say 5 or 10) of the number of adaptive parameters in the model
- Regularization is often used to control the over-fitting phenomenon

Polynomial curve fitting: Regularization

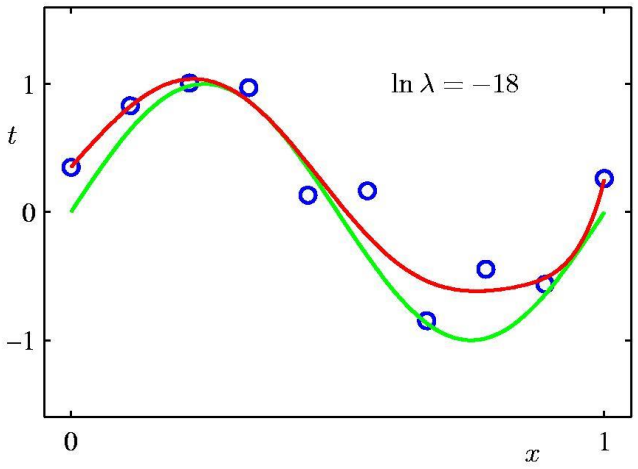
- Regularization: Add a penalty term to the error function to discourage the coefficients from reaching large values

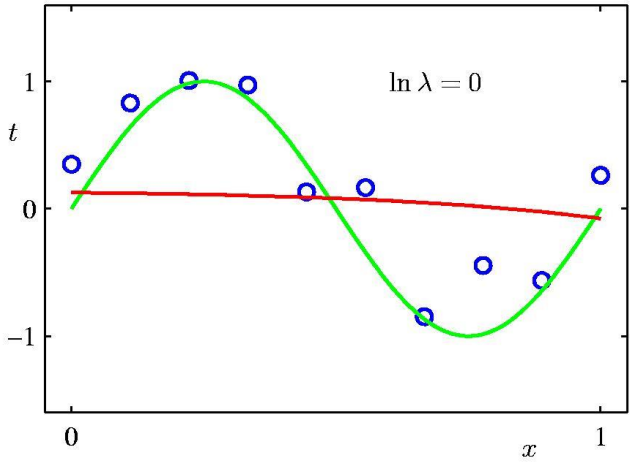
$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

where $\|\mathbf{w}\|^2 = \mathbf{w}^T\mathbf{w} = \omega_0^2 + \omega_1^2 + \cdots + \omega_M^2$

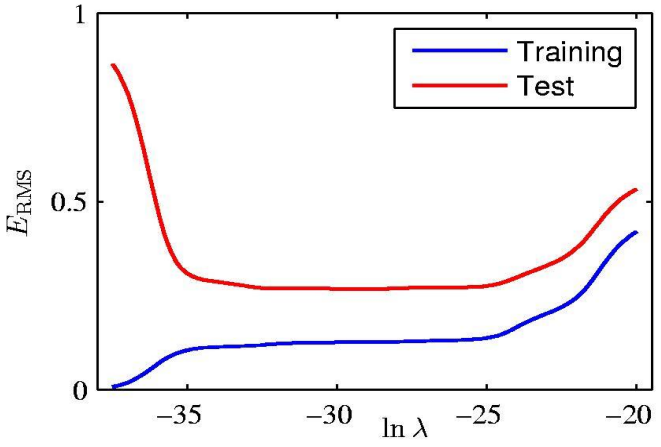
  - The coefficient ω_0 is usually omitted
  - This kind of techniques is called shrinkage methods in the statistics literature
  - A quadratic regularizer is called ridge regression
  - In neural networks, this approach is known as weight decay

Polynomial curve fitting: Regularization





	$\ln\lambda = -\infty$	$\ln\lambda = -18$	$\ln\lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

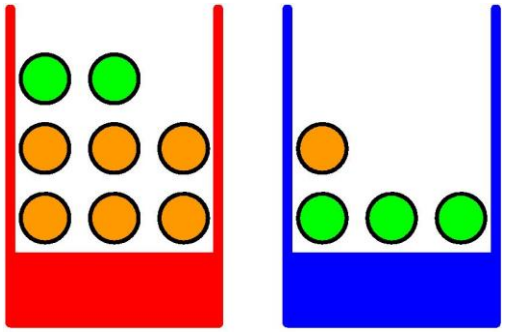


Probability theory

- Pattern recognition handles data uncertainties, which result from
  - Noise on measurement
  - Finite size of data sets

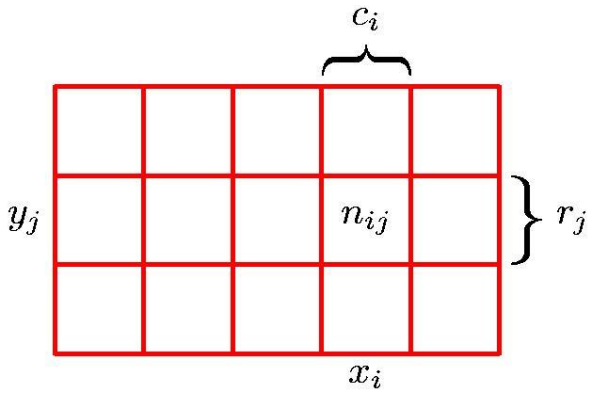
- Probability theory provides a consistent framework to manipulate uncertainties, and hence is essential to pattern recognition research

A toy examples



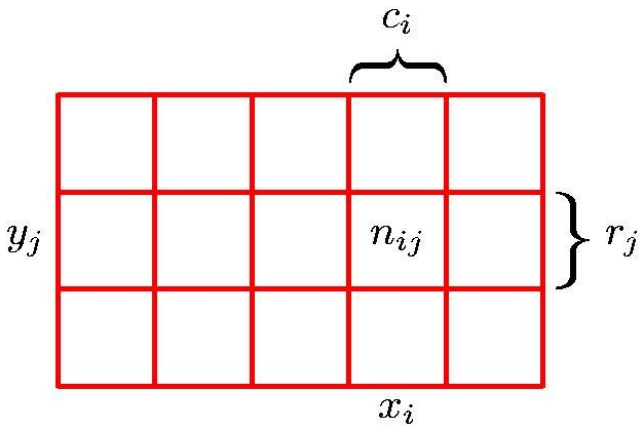
- Two boxes: *r* (red box) and *b* (blue box)
- Two types of fruits: *a* (apple) and *o* (orange)
- A trial: Randomly selecting a box from which we randomly picking a fruit
- Introduce one variable *B* for box and one variable *F* for fruit
- Many trials: Repeat the process many times
- Question 1: What is the probability that an apple is picked
  - Marginal probability
- Question 2: Given that we have picked an orange, what is the probability that the box we chose was the blue one?
  - Conditional probability

Probability theory: A two-variable case



- Two random variables: X and Y
- Each variable has a set of discrete states
  - X can take any value x_i where $i = 1, 2, \dots, M$
  - Y can take any value y_j where $j = 1, 2, \dots, L$
- N trails where both variables X and Y are sampled
- Some notations
  - Let the number of trails where $X = x_i$ and $Y = y_j$ be n_{ij}
  - Let the number of trails where X takes value x_i be c_i
  - Let the number of trails where Y takes value y_j be r_j

Joint, marginal, and conditional probabilities



- The probability that X takes value x_i and Y takes value y_j is called joint probability
- It is defined by the fraction of points (trails) falling in the cell *i,j*

$$p(X = x_i, Y = y_j) = \frac{n_{ij}}{N}$$

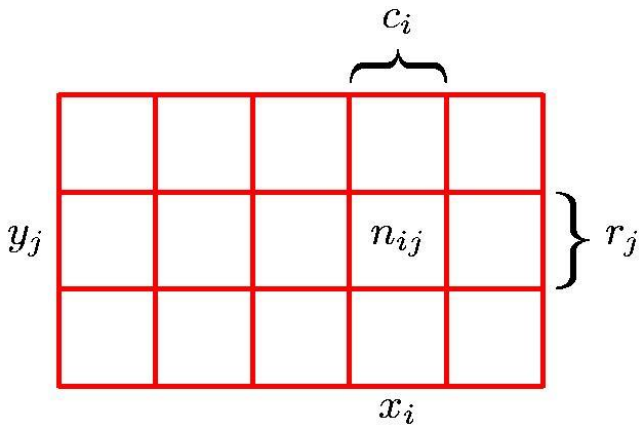
Joint, marginal, and conditional probabilities

- The probability that X takes value x_i irrespective of the value of Y is called marginal probability and is written as $p(X = x_i)$
- It is defined by the fraction of the number of points that fall in column *i*, namely

$$p(X = x_i) = \frac{c_i}{N}$$

- With the joint probability and $c_i = \sum_j n_{ij}$, we have

$$p(X = x_i) = \sum_{j=1}^{L} p(X = x_i, Y = y_j)$$



- The sum rule

Joint, marginal, and conditional probabilities

- If we consider only those cases where X takes value x_i, the fraction of those cases where $Y = y_j$ is written as $p(Y = y_j | X = x_i)$. It is called conditional probability
- It is defined by

$$p(Y = y_j | X = x_i) = \frac{n_{ij}}{c_i}$$

- Relationships among joint, marginal, and conditional probabilities:

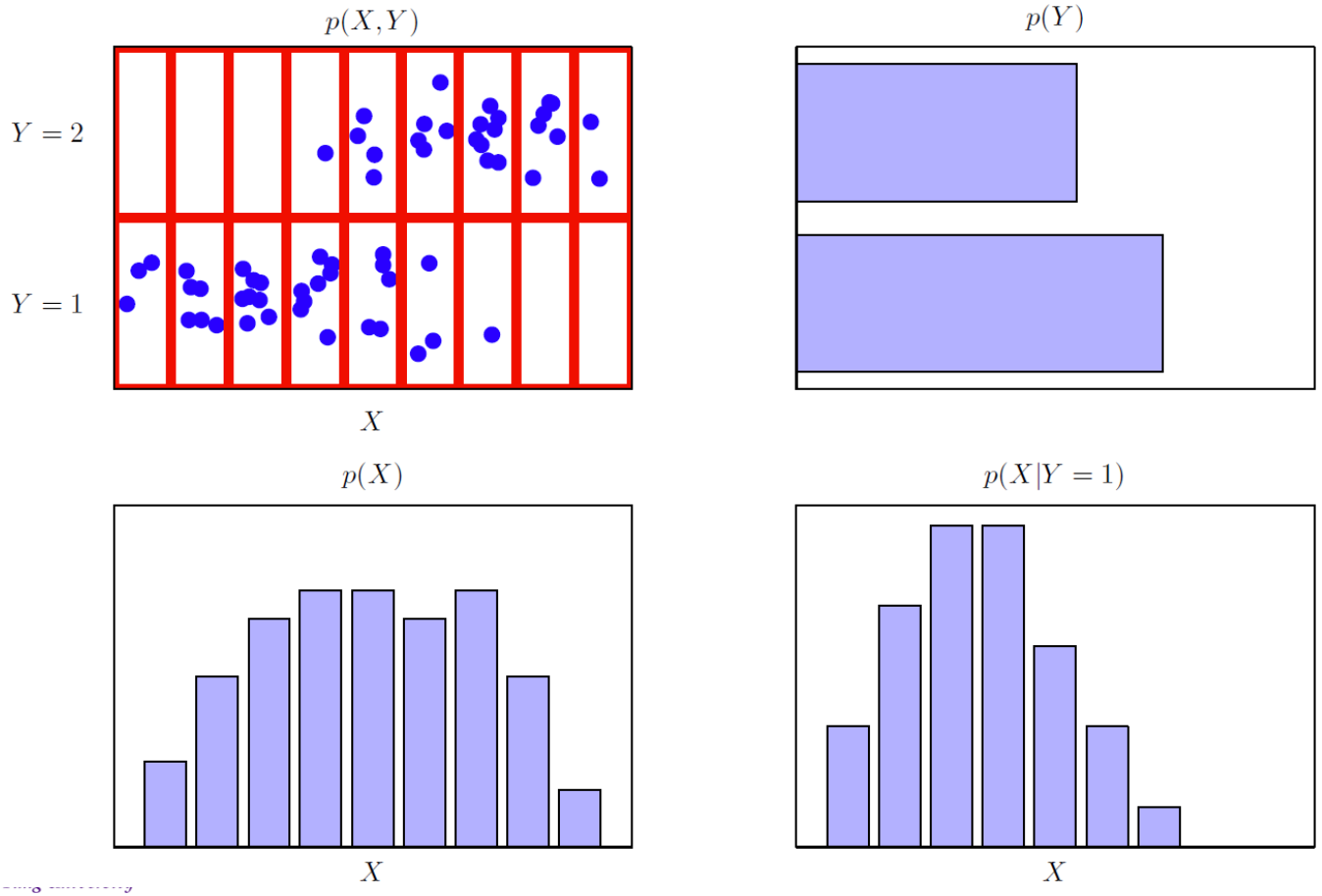
$$\begin{aligned} p(X = x_i, Y = y_j) &= \frac{n_{ij}}{N} = \frac{n_{ij}}{c_i} \cdot \frac{c_i}{N} \\ &= p(Y = y_j | X = x_i) p(X = x_i) \end{aligned}$$

- The product rule

Joint, marginal, and conditional probabilities

$$\textbf{sum rule} \qquad p(X) = \sum_{Y} p(X, Y)$$

$$\textbf{product rule} \qquad p(X, Y) = p(Y|X)p(X)$$



Bayes' theorem

- By using the product rule and the symmetry property $p(X,Y)=p(Y,X)$, we have

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

Probability with continuous variables

- The probability density $p(x)$ over a continuous variable x must satisfy the two conditions:

$$p(x) \geqslant 0$$

$$\int_{-\infty}^{\infty} p(x)\,\mathrm{d}x = 1$$

  - Nonnegative: Probabilities are nonnegative
  - Sum-to-1: The value of x must lie somewhere on the real axis

- The cumulative distribution function defines the probability that x lies in the interval $(-\infty, z)$ via

$$P(z) = \int_{-\infty}^{z} p(x)\mathrm{d}x$$

Sum rule and product rule

- Sum rule in discrete cases

$$p(X) = \sum_{Y} p(X, Y)$$

- Sum rule in continuous cases

$$p(x) = \int p(x, y)\, \mathrm{d}y$$

- Product rule in discrete cases

$$p(X, Y) = p(Y|X)p(X)$$

- Product rule in continuous cases

$$p(x, y) = p(y|x)p(x)$$

Expectations and covariances

- The average value of some function $f(x)$ under a probability distribution $p(x)$ is called the expectation of $f(x)$

- For a discrete distribution, the expectation of $f(x)$ is

$$\mathbb{E}[f] = \sum_x p(x) f(x)$$

- For a continuous probability, the expectation of $f(x)$ is

$$\mathbb{E}[f] = \int p(x) f(x) \, \mathrm{d}x$$

Expectations and covariances

- The variance of $f(x)$ under a probability distribution $p(x)$ is

$$\mathrm{var}[f] = \mathbb{E}\left[(f(x) - \mathbb{E}[f(x)])^2\right]$$

- It is a measure of how much variability there is in $f(x)$ around its mean $\mathbb{E}[f(x)]$

- For two random variables x and y, the covariance is defined by

$$\begin{aligned}\mathrm{cov}[x, y] &= \mathbb{E}_{x,y}\left[\{x - \mathbb{E}[x]\}\{y - \mathbb{E}[y]\}\right] \\ &= \mathbb{E}_{x,y}[xy] - \mathbb{E}[x]\mathbb{E}[y]\end{aligned}$$

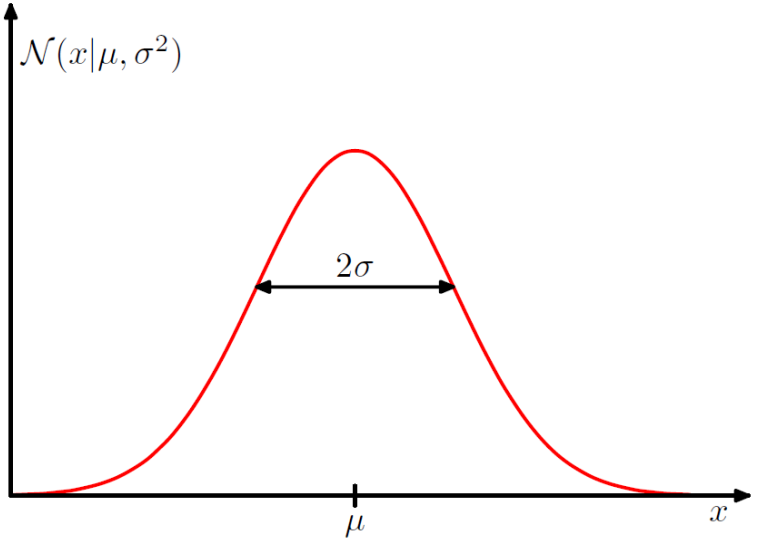
- It expresses the extent to which x and y vary together.

Gaussian distribution

- For a single continuous variable, the Gaussian or normal distribution is defined by

$$\mathcal{N}\left(x|\mu,\sigma^2\right) = \frac{1}{(2\pi\sigma^2)^{1/2}}\exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

which is specified by two parameters: mean μ and variance σ^2



$$\mathcal{N}(x|\mu,\sigma^2) > 0$$

$$\int_{-\infty}^{\infty}\mathcal{N}\left(x|\mu,\sigma^2\right)\,\mathrm{d}x = 1$$

Mean and variance of a Gaussian distribution

- The average value of a random variable x whose distribution is Gaussian

$$\mathbb{E}[x] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x \, \mathrm{d}x = \mu$$

- The second order moment of variable x

$$\mathbb{E}[x^2] = \int_{-\infty}^{\infty} \mathcal{N}\left(x|\mu,\sigma^2\right) x^2 \, \mathrm{d}x = \mu^2 + \sigma^2$$

- The variance of variable x

$$\mathrm{var}[x] = \mathbb{E}[x^2] - \mathbb{E}[x]^2 = \sigma^2$$

Multivariate Gaussian

- The multivariate Gaussian distribution defined over a *D*-dimensional vector **x** of continuous variables:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

where *D* x *D* matrix is called the co-variance matrix while $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$

Bayes' theorem for polynomial curve fitting

- Recall the curve fitting problem
  - Given a set of *N* observations $D = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ and their target values $\{t_1, t_2, \ldots, t_N\}$
  - Polynomial curve fitting: Determine the values of $\boldsymbol{w}$

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

- Prior probability $p(\boldsymbol{w})$: Express our assumption about $\boldsymbol{w}$ before observing any data
- Likelihood function $p(D|\boldsymbol{w})$: Express how probable the observed data *D* is under $\boldsymbol{w}$. It is evaluated after the observations *D* are given

Bayes' theorem for polynomial curve fitting

- Bayes' theorem takes the form

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

  which allows us to evaluate the uncertainty after we have observations *D*

- $p(D)$ is the normalization constant. Thus, we have

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

Determining Gaussian parameters by maximum likelihood

- Given a set of *N* observations: $\mathbf{x} = (x_1, \ldots, x_N)$
- Assume these observations are sampled from a Gaussian distribution with mean μ and variance σ^2 (unknown)
- Our goal is to determine μ and σ^2 based on the observations
- We assume that data are sampled independently from the same distribution, namely independent and identically distributed, or i.i.d. for short

Determining Gaussian parameters by maximum likelihood

- Since the data are i.i.d., the likelihood function of data given mean μ and variance σ^2 is

$$p(\mathbf{x}|\mu,\sigma^2) = \prod_{n=1}^{N} \mathcal{N}\left(x_n|\mu,\sigma^2\right)$$

- The log likelihood function is

$$\ln p\left(\mathbf{x}|\mu,\sigma^2\right) = -\frac{1}{2\sigma^2}\sum_{n=1}^{N}(x_n-\mu)^2 - \frac{N}{2}\ln\sigma^2 - \frac{N}{2}\ln(2\pi)$$

- Maximum likelihood solution:

$$\mu_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N} x_n \qquad \sigma^2_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(x_n-\mu_{\mathrm{ML}})^2$$

Probabilistic perspective of polynomial curve fitting

- Given *N* data for regression: $\mathbf{x} = (x_1, \ldots, x_N)^{\mathrm{T}}$ & $\mathbf{t} = (t_1, \ldots, t_N)^{\mathrm{T}}$
  - Fit the data using a polynomial function of the form:

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

  - This function is parametrized by $\mathbf{w}$

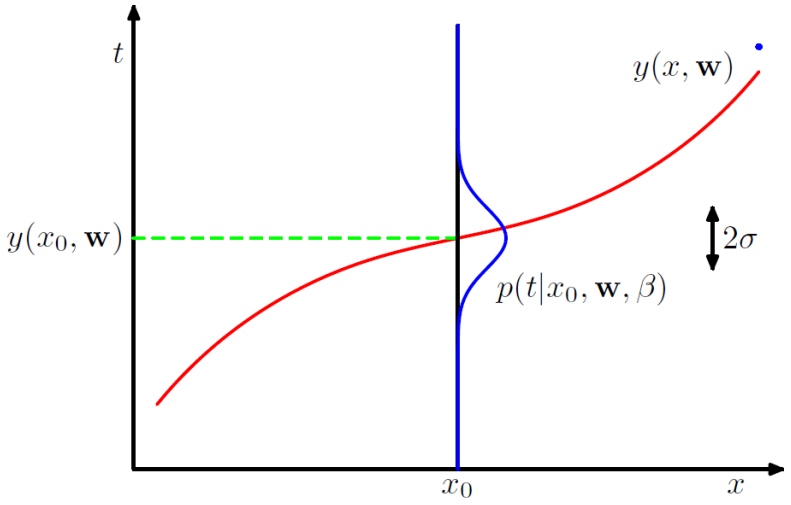
- Given the value of x, we assume the corresponding value of t has a Gaussian distribution with a mean equal to $y(x, \boldsymbol{w})$

$$p(t|x, \mathbf{w}, \beta) = \mathcal{N}\left(t|y(x, \mathbf{w}), \beta^{-1}\right)$$

where β^{-1} is the variance σ^2 (β is called precision)

Probabilistic perspective of polynomial curve fitting

- The Gaussian conditional distribution for t given x



- If data are i.i.d., the likelihood function is

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n|y(x_n, \mathbf{w}), \beta^{-1}\right)$$

Maximum likelihood solution

- The log likelihood function

$$\ln p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = -\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi)$$

- Maximum likelihood (ML) solution for determining $\mathbf{w}$ and β
  - Compute the gradient of the log likelihood function w.r.t. $\mathbf{w}$. And set it to 0. We can get $\mathbf{w}_{\mathrm{ML}}$.

$$\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

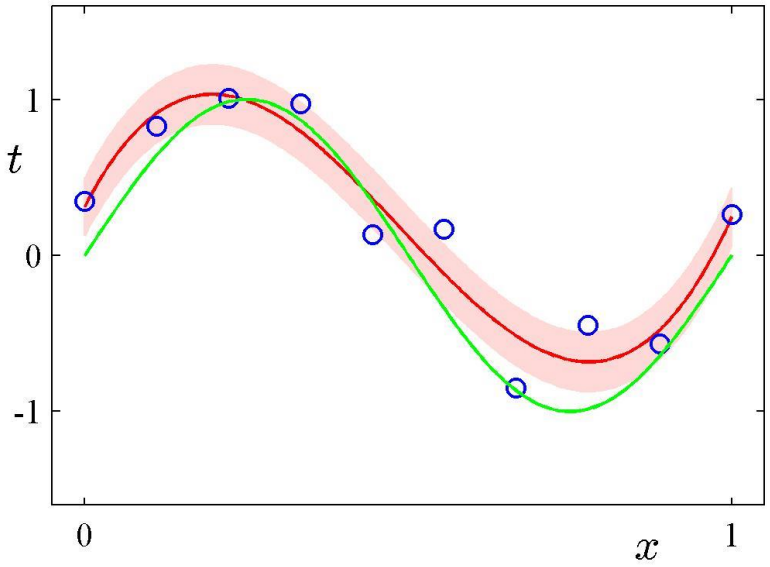
  - By setting the gradient of the log likelihood function w.r.t. β to 0, β_{ML} is obtained by solving

$$\frac{1}{\beta_{\mathrm{ML}}} = \frac{1}{N}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}_{\mathrm{ML}}) - t_n\}^2$$

Maximum likelihood solution

- After determining the values of $\mathbf{w}_{\mathrm{ML}}$ and β_{ML}, we can make predictions for a new value of x

$$p(t|x, \mathbf{w}_{\mathrm{ML}}, \beta_{\mathrm{ML}}) = \mathcal{N}\left(t|y(x, \mathbf{w}_{\mathrm{ML}}), \beta_{\mathrm{ML}}^{-1}\right)$$



Maximum a posterior (MAP) solution

- While ML solution is obtained by maximizing the likelihood, MAP solution is by maximizing the posterior
- Recall $\text{posterior} \propto \text{likelihood} \times \text{prior}$
- Introduce a prior distribution over the curve parameters $\mathbf{w}$

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left\{-\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right\}$$

  - M is the order of the polynomial
  - α is a hyperparameter
- The posterior distribution for $\mathbf{w}$

$$p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta) \propto p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)p(\mathbf{w}|\alpha).$$

Maximum a posterior (MAP) solution

- The MAP solution, $\mathbf{w}_{\mathrm{MAP}}$ and β_{MAP}, is obtained by maximizing the posterior function, or equivalently by minimizing

$$\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w})-t_n\}^2+\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}$$

Bayesian curve fitting

- We make a point estimation of **w** no matter in ML and MAP solutions
- In a full Bayesian approach, we integrate over all possible values of **w** for regression, i.e.,

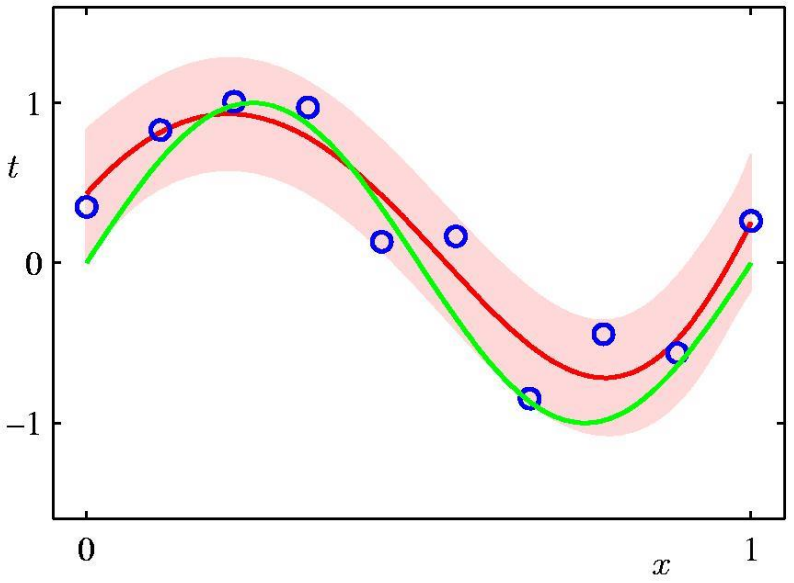
$$p(t|x,\mathbf{x},\mathbf{t}) = \int p(t|x,\mathbf{w})p(\mathbf{w}|\mathbf{x},\mathbf{t})\,\mathrm{d}\mathbf{w} = \mathcal{N}\left(t|m(x),s^2(x)\right)$$

$$m(x) = \beta\boldsymbol{\phi}(x)^{\mathrm{T}}\mathbf{S}\sum_{n=1}^{N}\boldsymbol{\phi}(x_n)t_n$$

$$s^2(x) = \beta^{-1} + \boldsymbol{\phi}(x)^{\mathrm{T}}\mathbf{S}\boldsymbol{\phi}(x).$$

$$\phi(x_n) = (x_n^0,\ldots,x_n^M)^T$$

$$\mathbf{S}^{-1} = \alpha\mathbf{I} + \beta\sum_{n=1}^{N}\boldsymbol{\phi}(x_n)\boldsymbol{\phi}(x)^{\mathrm{T}}$$



Probabilistic polynomial curve fitting

- Given the assumption $p(t|x, \mathbf{w}, \beta) = \mathcal{N}\left(t|y(x, \mathbf{w}), \beta^{-1}\right)$
  - ML solution: Find $\mathbf{w}$ that maximizes the likelihood function

$$p(t \mid x, D) = p(t \mid x, \mathbf{w}_{\mathbf{ML}}, \beta^{-1})$$

  - MAP solution: Find $\mathbf{w}$ that maximizes the posterior probability

$$p(t \mid x, D) = p(t \mid x, \mathbf{w}_{\mathbf{MAP}}, \beta^{-1})$$

  - Bayesian solution: Integrate over $\mathbf{w}$

$$p(t \mid x, D) = p(t|x, \mathbf{x}, \mathbf{t}) = \int p(t|x, \mathbf{w}) p(\mathbf{w}|\mathbf{x}, \mathbf{t}) \, \mathrm{dw}$$

Model selection

- Hyperparameters, such as M in polynomial curve fitting, control the model behavior complexity

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

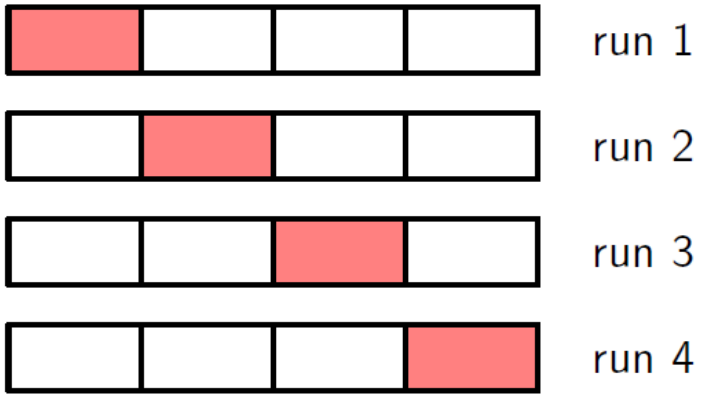
- Model selection: determine the values of hyperparameters that achieve the best predictive performance on new (testing) data
- Idea: split training data into a training set and a validation set
  - Training set: Used to learn the model with particular hyperparameters values
  - Validation set: Used to evaluate the performance of the learned model

Model selection

- About the size of the validation set
  - A large validation set: Less training data for model learning
  - A small validation set: Less reliable performance evaluation

Model selection via cross validation

- *S*-fold cross-validation
  - Partition training data into *S* equal-sized groups
  - *S*-1 groups are used to train the model that is evaluated on the remaining group
  - Repeat the procedure for all *S* possible runs
  - Average the performance

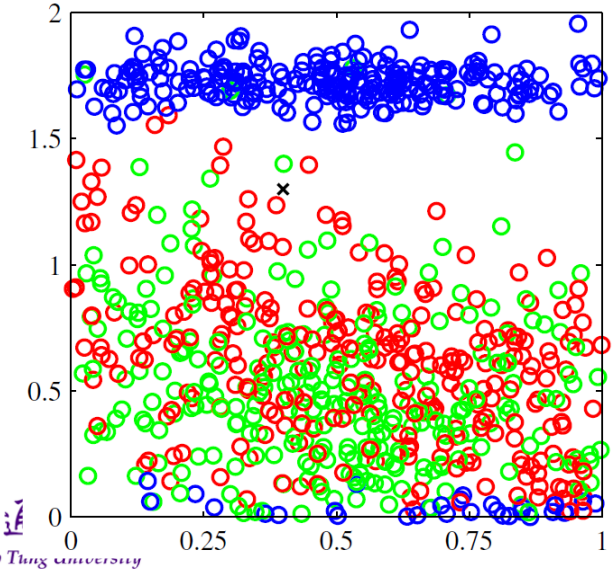


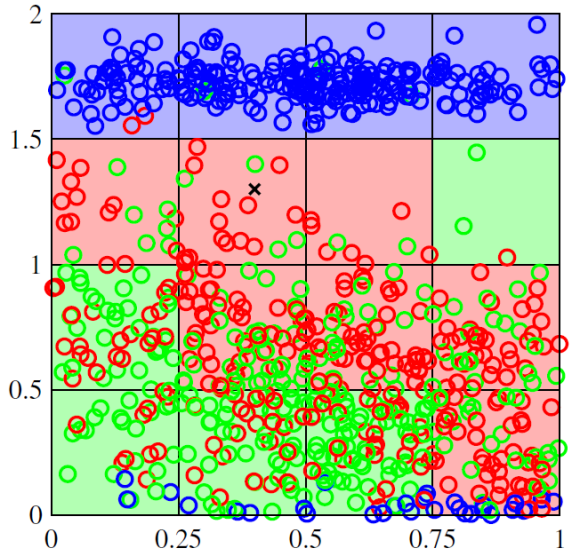
Drawbacks of model selection

- If training data are limited, a large value of *S* is appropriate
- At the extreme, setting *S*=*N* (number of training data), it gives the leave-one-out technique
- Some drawbacks
  - The number of training runs increases by a factor of *S*
  - The number of hyperparameter value combinations increases exponentially

Curse of dimensionality

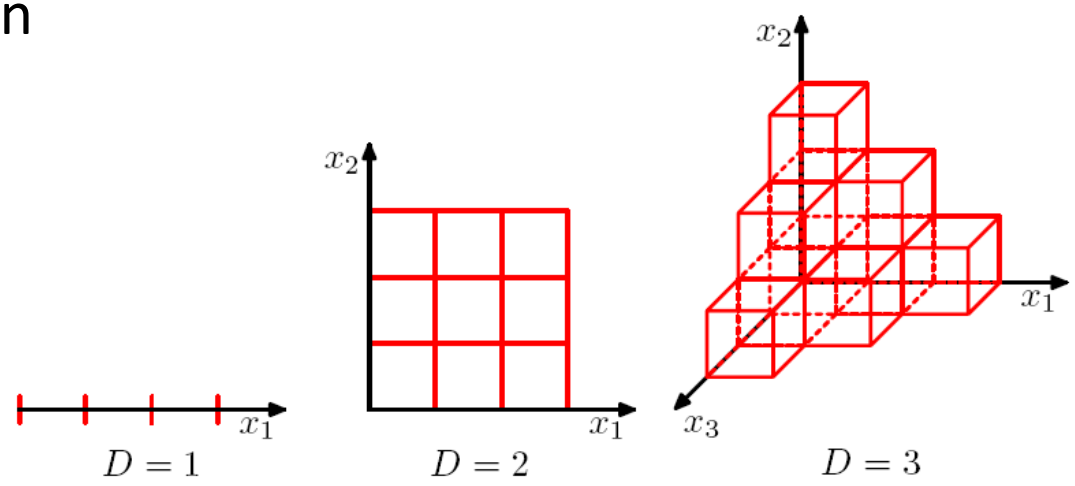
- Consider a three-class classification problem
  - What is the class of that cross?
  - Its class strongly depends on its surrounding training data
- A simple algorithm for classification
  - Divide the input space into regular cells
  - The cross class is that having the most data in the same cell





Curse of dimensionality

- In the cases where data are in a high-dimensional space, does this naïve method work well?
- Reliable classification needs a sufficient number of training data in each cell
- The number of cells grows exponentially w.r.t. the space dimension



Curse of dimensionality

- In polynomial curve fitting, consider the case where data in a *D*-dimensional space and the polynomial order is set to 3

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{D} w_i x_i + \sum_{i=1}^{D}\sum_{j=1}^{D} w_{ij} x_i x_j + \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{k=1}^{D} w_{ijk} x_i x_j x_k$$

  - Exponential growth of polynomial coefficients
  - The number of parameters grows exponentially
  - At the risk of over-fitting

Curse of dimensionality

- Good news
  - Real data can often be confined to a subspace of a lower effective dimension
  - Real data typically exhibit some smoothness properties (at least locally) so we can exploit local interpolation-like techniques for the prediction of the target variables

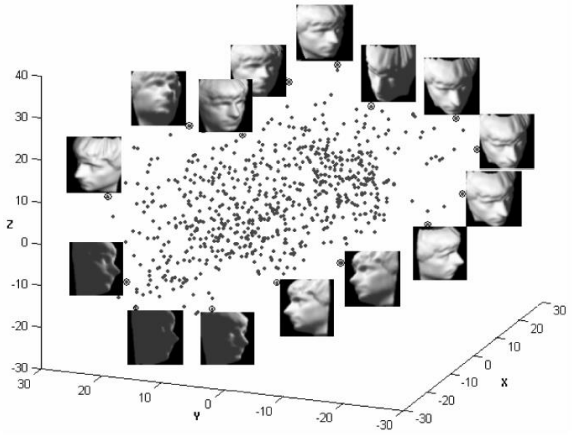


Fig. 19. Three-dimensional embedding of ISOMAP face data using RML.

Ref: T. Lin & H. Zha, "Reimannian Manifold Learning", PAMI, May 2008

Summary

- Polynomial curve fitting for regression
  - Fitting by minimizing the sum-of-squares error

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

  - Regularization for alleviating overfitting

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

- Probability density
  - Expectation, variance, and covariance
  - Gaussian distribution

Summary

- Bayes' theorem

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

- When applying Bayes' theorem to polynomial curve fitting,
  - ML solution: Find $\mathbf{w}$ that maximizes the likelihood function
  - MAP solution: Find $\mathbf{w}$ that maximizes the posterior probability
  - Bayesian solution: Integrate over $\mathbf{w}$
- Model selection by cross-validation
- Curse of dimensionality

References

- Chapters 1.1, 1.2, 1.3, and 1.4 in the PRML textbook

Thank You for Your Attention!

Yen-Yu Lin (林彥宇)

Email: lin@cs.nctu.edu.tw

URL: https://www.cs.nctu.edu.tw/members/detail/lin