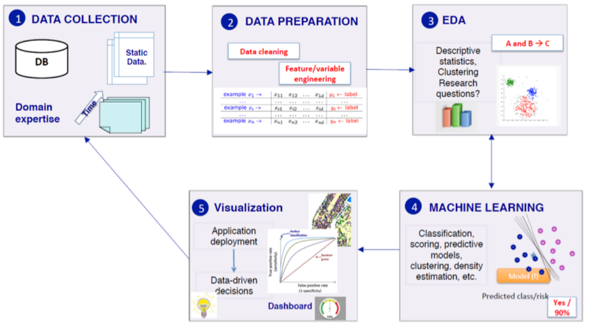
Data types

Data comes in different sizes and also flavors (types):

- Texts
- Numbers
- Clickstreams
- Graphs
- Tables
- Images
- Transactions
- Videos
- Some or all of the above!

一般分为两种:
vector
scalar

The Data Science process



①数据收集
②数据准备: a. 清洗
b. 特征工程

ML vs. Statistics

Statistics:

- Hypothesis testing
- Experimental design
- Anova
- Linear regression
- Logistic regression
- GLM
- PCA

Machine Learning:

- SVMs
- Neural Networks
- Decision trees
- Rule induction
- Clustering method
- Association rules
- Feature selection
- Visualization
- Graphical models
- Genetic algorithm

Machine Learning definition

"How do we create computer programs that improve with experience?"

Tom Mitchell

"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*. "

Tom Mitchell. Machine Learning 1997.

Supervised vs. Unsupervised

Given: Training data: $(x_1, y_1), \dots, (x_n, y_n)$, $x_i \in \mathbb{R}^d$ and y_i is the label.

$$
\begin{array}{l|cccc|l}
\text{example } x_1 \rightarrow & x_{11} & x_{12} & \dots & x_{1d} & y_1 \leftarrow \text{label} \\
\dots & \dots & \dots & \dots & \dots & \dots \\
\hline
\text{example } x_i \rightarrow & x_{i1} & x_{i2} & \dots & x_{id} & y_i \leftarrow \text{label} \\
\hline
\dots & \dots & \dots & \dots & \dots & \dots \\
\text{example } x_n \rightarrow & x_{n1} & x_{n2} & \dots & x_{nd} & y_n \leftarrow \text{label}
\end{array}
$$

fruit	length	width	weight	label
fruit 1	165	38	172	Banana
fruit 2	218	39	230	Banana
fruit 3	76	80	145	Orange
fruit 4	145	35	150	Banana
fruit 5	90	88	160	Orange
...				
fruit n	...	...	...	...

11

Unsupervised learning:
Learning a model from **unlabeled** data.

Supervised learning:
Learning a model from **labeled** data.

无监督：数据没有label
有监督：数据有label

Unsupervised Learning

Training data: "examples" x.

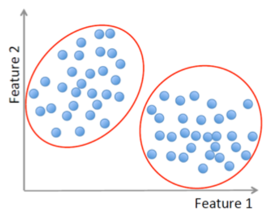
$$x_1, \dots, x_n, \; x_i \in X \subset \mathbb{R}^d$$

- **Clustering/segmentation:**

$$f: \mathbb{R}^d \rightarrow \{C_1, \dots, C_k\} \text{ (set of clusters).}$$

Example: Find clusters in the population, fruits, species.

聚类



Methods: K-means, Gaussian mixtures, hierarchical clustering, spectral clustering, etc.

Supervised Learning

Training data: "examples" x with "labels" y.

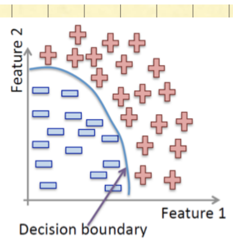
$$(x_1, y_1), \dots, (x_n, y_n), \; x_i \in \mathbb{R}^d$$

- **Classification**: y is discrete. To simplify, $y \in \{-1, +1\}$

$$f: \mathbb{R}^d \rightarrow \{-1, +1\} \quad (f \text{ is called a } \textbf{binary classifier})$$

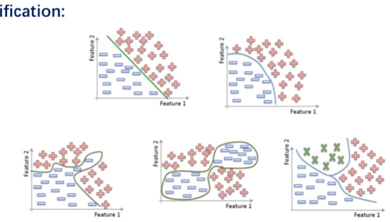
Example: Approve credit yes/no, spam/ham, banana/orange.

分类

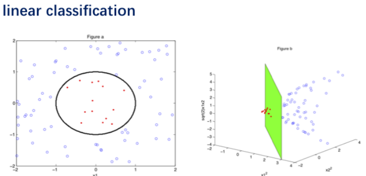


Methods: Support Vector Machines, neural networks, decision trees, K-nearest neighbors, naive Bayes, etc.

Classification:



Non linear classification



Supervised Learning

Training data: "examples" x with "labels" y.

$$(x_1, y_1), \ldots, (x_n, y_n), x_i \in \mathbb{R}^d$$

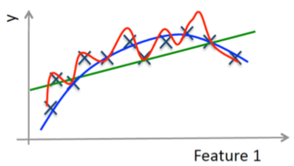
- **Regression:** y is a real value, $y \in \mathbb{R}$.

  $f: \mathbb{R}^d \to \mathbb{R}$ (f is called a **regressor**)

  Example: amount of credit, weight of fruit.

回归

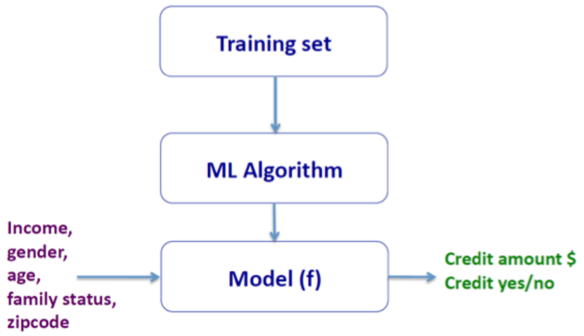
Regression:



拟合曲线

Example: Income in function of age, weight of the fruit in function of its length.

Training and Testing



K-nearest neighbors

- Not every ML method builds a model!
- Our first ML method: KNN.
- Main idea: Uses the **similarity** between examples.
- Assumption: Two similar examples should have same labels.
- Assumes all examples (instances) are points in the d dimensional space $\mathbb{R}^d$.

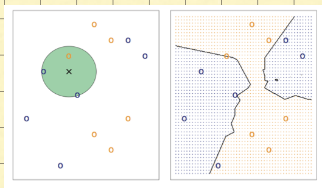
利用相似度

假设相似的样本之间有相同的label

- KNN uses the standard **Euclidian distance** to define nearest neighbors. Given two examples x_i and x_j:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{d} (x_{ik} - x_{jk})^2}$$

欧拉距离



Training algorithm:

Add each training example (x, y) to the dataset D.

$x \in \mathbb{R}^d$, $y \in \{-1, +1\}$.

Classification algorithm:

Given an example x_q to be classified. Suppose $N_k(x_q)$ is the set of the K-nearest neighbors of x_q.

$$\hat{y}_q = sign\left(\sum_{x_i \in N_k(x_q)} y_i\right)$$

Question: What are the pros and cons of K-NN?

Pros:

+ Simple to implement.
+ Works well in practice.
+ Does not require to build a model, make assumptions, tune parameters.
+ Can be extended easily with news examples.

Cons:

- Requires large space to store the entire training dataset.
- Slow! Given n examples and d features. The method takes $O(n \times d)$ to run.
- Suffers from the curse of dimensionality.

Application of KNN

1. Information retrieval.
2. Handwritten character classification using nearest neighbor in large databases.
3. Recommender systems (user like you may like similar movies).
4. Breast cancer diagnosis.
5. Medical data mining (similar patient symptoms).
6. Pattern recognition in general.

Training and Testing

- We calculate E^{train} the in-sample error (training error or empirical error/risk).

$$E^{train}(f) = \sum_{i=1}^{n} \ell oss(y_i, f(x_i))$$

- Examples of loss functions:
  - **Classification error:**

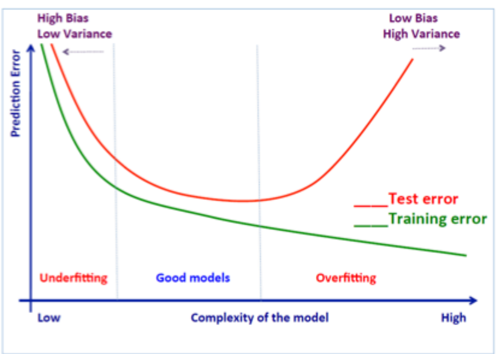
$$\ell oss(y_i, f(x_i)) = \begin{cases} 1 & \text{if } sign(y_i) \neq sign(f(x_i)) \\ 0 & \text{otherwise} \end{cases}$$

  - **Least square loss:**

$$\ell oss(y_i, f(x_i)) = (y_i - f(x_i))^2$$

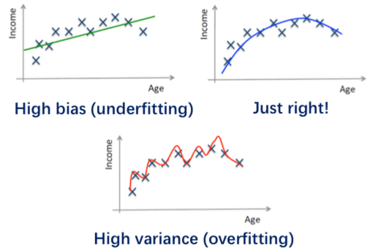
- We aim to have $E^{train}(f)$ small, i.e., minimize $E^{train}(f)$
- We hope that $E^{test}(f)$, the out-sample error (test/true error), will be small too.

Structural Risk Minimization



IMPORTANT

Overfitting / Underfitting



Avoid overfitting

In general, use simple models!
- **Reduce the number** of features manually or do feature selection.
- Do a **model selection**.
- Use **regularization** (e.g., keep the features but reduce their importance by setting small parameter values).
- Do a **cross-validation** to estimate the test error.

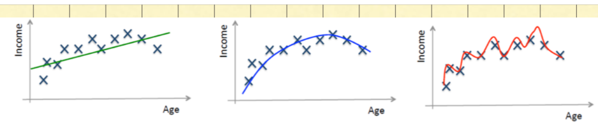
①减少特征
②模型选择
③正则化
④交叉验证

Regularization: Intuition

We want to minimize:

Classification term + $C \times$ Regularization term

$$\sum_{i=1}^{n} \ell oss(y_i, f(x_i)) + C \times R(f)$$



$f(x) = \lambda_0 + \lambda_1 x$ ... (1)
$f(x) = \lambda_0 + \lambda_1 x + \lambda_2 x^2$ ... (2)
$f(x) = \lambda_0 + + \lambda_1 x + \lambda_2 x^2 + \lambda_3 x^3 + \lambda_4 x^4$ ... (3)

Hint: Avoid high-degree polynomials.

尽量避免高次多项式

Train, Validation and Test

1. Training set is a set of examples used for learning a model (e.g., a classification model).
2. Validation set is a set of examples that cannot be used for learning the model but can help tune model parameters (e.g., selecting K in K-NN). Validation helps control overfitting.
3. Test set is used to assess the performance of the final model and provide an estimation of the test error.

Note: Never use the test set in any way to further tune the parameters or revise the model.

K-fold Cross Validation

A method for estimating test error using training data.

Algorithm:

Given a learning algorithm A and a dataset D

Step 1: Randomly partition D into k equal-size subsets $D_1, \ldots, D_k$

Step 2:

For $j = 1$ to k

Train A on all D_i, $i \in 1, \ldots, k$ and $i \neq j$, and get f_j.

Apply f_j to D_j and compute E^{D_j}

Step 3: Average error over all folds: $\sum_{j=1}^{k} (E^{D_j})$

Evaluation metrics

		Actual Label	
		Positive	Negative
Predicted Label	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Accuracy	(TP + TN) / (TP + TN + FP + FN)	The percentage of predictions that are correct
Precision	TP / (TP + FP)	The percentage of positive predictions that are correct
Sensitivity (Recall)	TP / (TP + FN)	The percentage of positive cases that were predicted as positive
Specificity	TN / (TN + FP)	The percentage of negative cases that were predicted as negative

准确率：预测正确所占比例
精度：
敏感度
特征性