Table of Contents

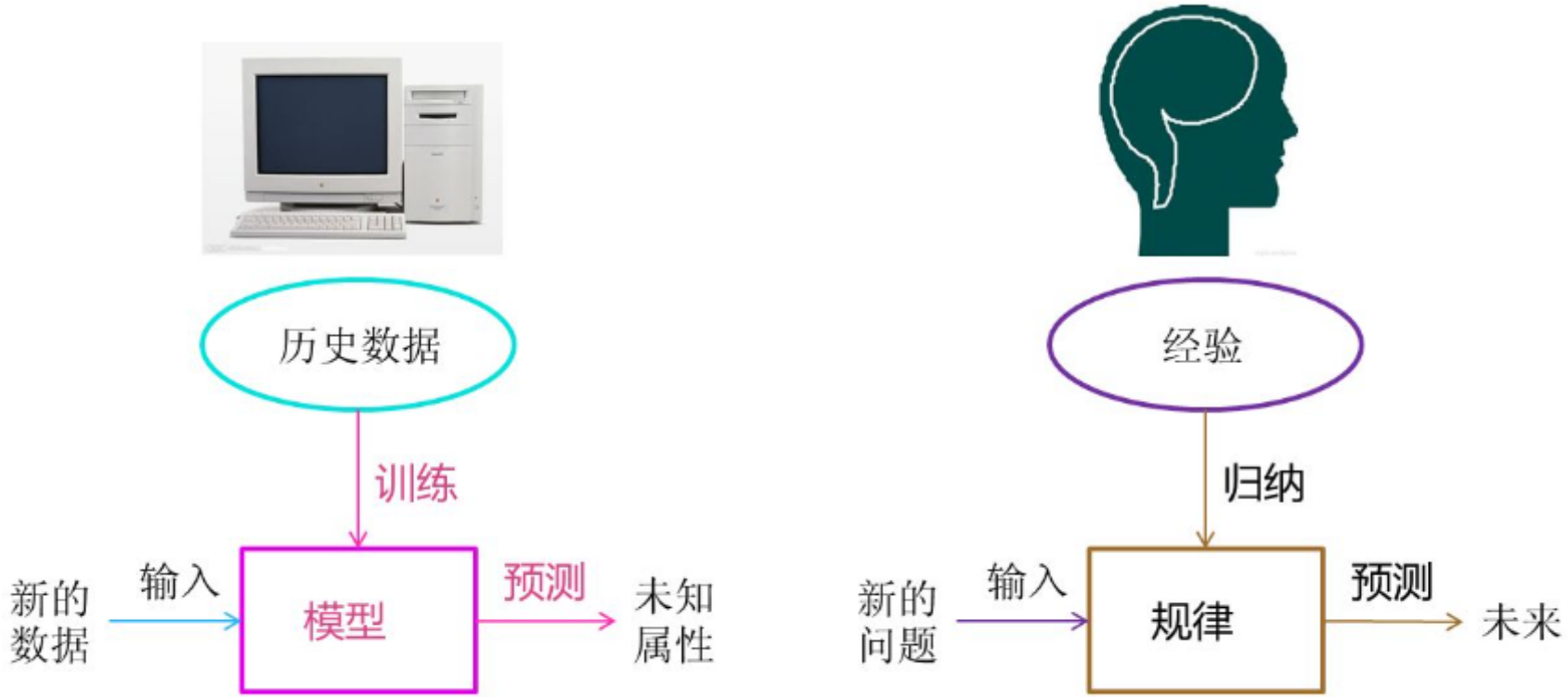
1 什么是机器学习?

- ## 1.1 最早定义:

Arthur Samuel最早定义机器学习为：*通过不显式地编程方式使计算机获得学习能力的研究领域*
Machine learning is the field of the study that gives computers the ability to learn without being explicity.
Samuel的定义可以回溯到50年代，他编写了一个西洋棋程序。

- ## 1.2 现今比较公认的定义:

来自卡内基梅隆大学的Tom Mitchell，提出机器学习是:
一个程序被认为能从经验E中学习，解决任务T，达到性能度量值P，当且仅当，有了经验E后，经过P评判，程序在处理T时的性能有所提升
经验E 就是程序上万次的自我练习的经验而任务T 就是下棋。性能度量值P呢，就是它在与一些新的对手比赛时，赢得比赛的概率。

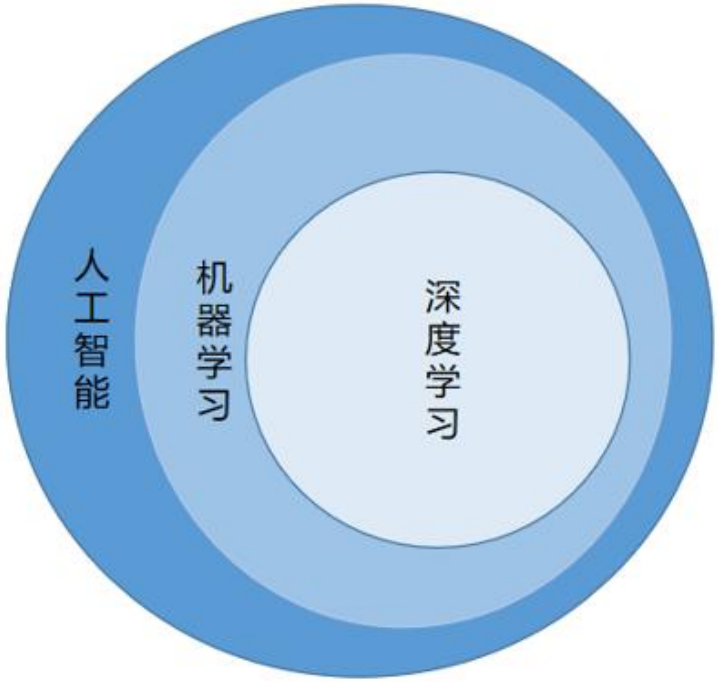


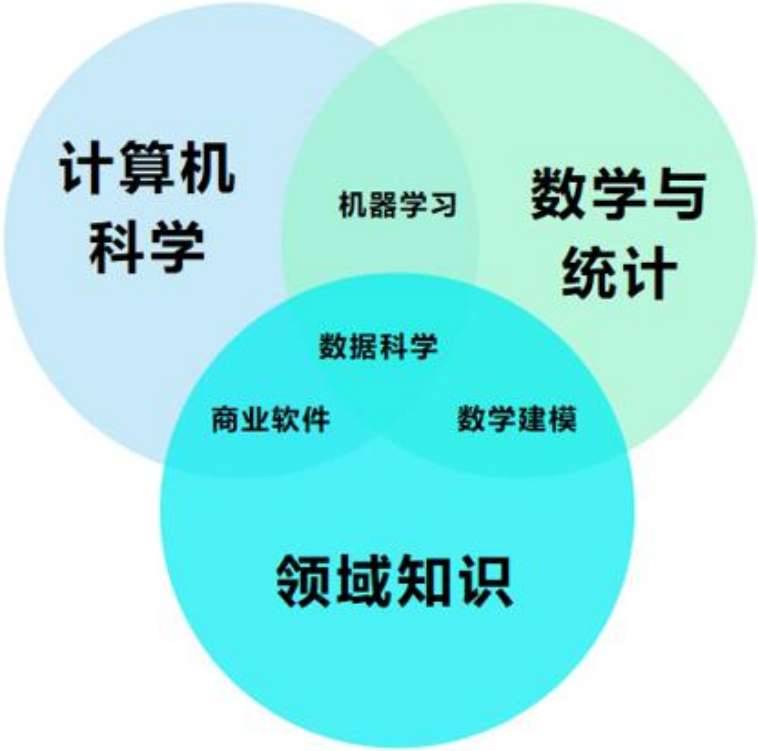
*注：机器学习不是基于编程形成的直接结果，而是通过归纳得出来的模型

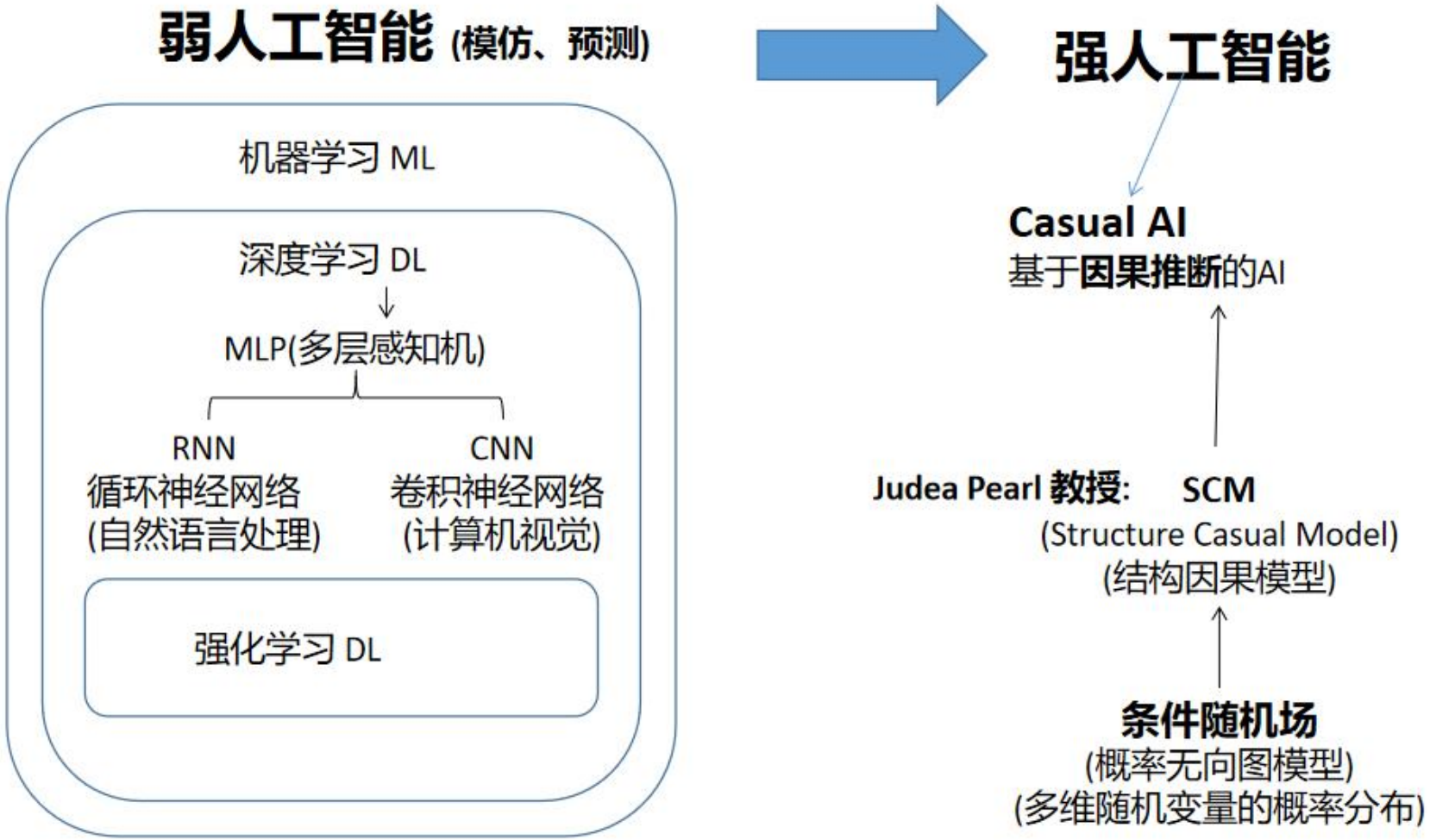
- ## 1.3 学习:

如果一个系统能通过执行某个过程改善它的性能，这就是学习。——赫伯特·西蒙

- ## 1.4 机器学习与相关知识的关系

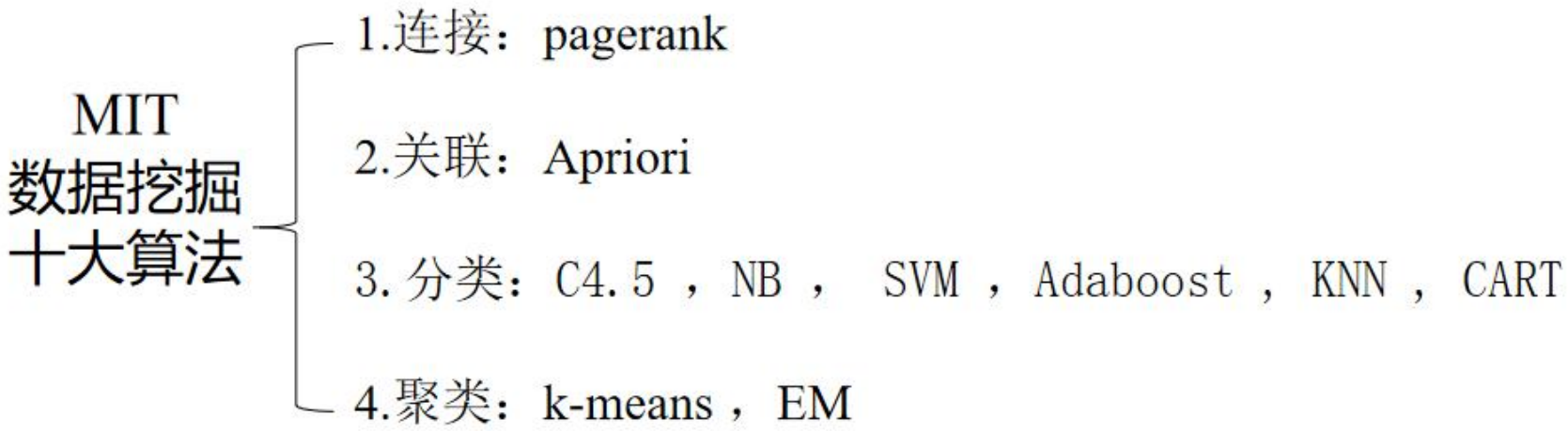






1.5 机器学习与数据挖掘的区别

- **机器学习**:
  - 2002年左右兴起，应用与学术界，偏重方法（知识发展 和 模式识别）
- **数据挖掘**:
  - 1990年兴起，应用于工业界，偏重流程。

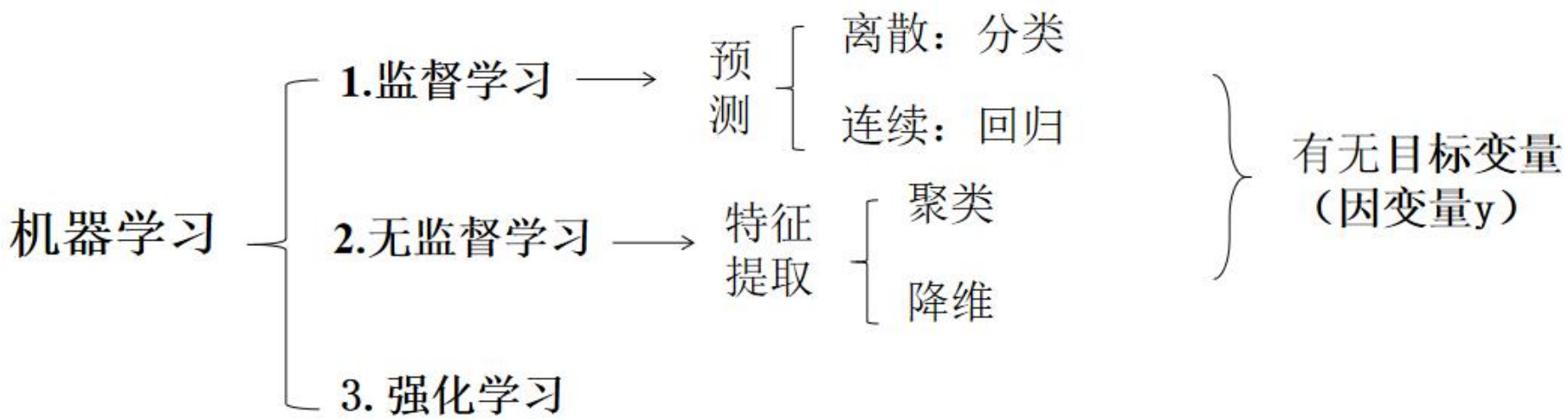


- ## 1.6 机器学习的基本步骤：

- 1. 获取训练数据集合
- 2. 确定假设空间，即所有可能的模型的集合
- 3. 确定模型选择的准则（什么是最优模型的标准），即学习的策略
- 4. 实现求解最优模型的算法（如何获取最优模型），即学习的算法
- 5. 通过算法中假设空间中，找到最优模型
- 6. 使用该模型对新数据进行预测和分析

2 机器学习的分类

- ## 2.1 基本分类:



3 机器学习三要素：机器学习=模型+策略+方法

3.1 模型 model：

- 学习的**条件概率分布**或**决策函数**
  - **概率分布**：$P(Y \mid X)$
    -
    -
  - **函数空间**：$y = f(x)$
    - 假设空间：
    - 假设参数空间：

$$\mathcal{F} = \{P \mid P(Y \mid X)\}$$
$$\mathcal{F}_\theta = \{P \mid P_\theta(Y \mid X), \theta \in \mathbf{R}^n\}$$

$$\mathcal{F} = \{f \mid Y = f(X)\}$$
$$\mathcal{F}_\theta = \{f \mid Y = f_\theta(X), \theta \in \mathbf{R}^n\}$$

3.2 策略 Strategy：

- 评价标准和依据：寻找使得**风险函数**最小的model,**损失值越小，风险越小，模型越好。**

- **3.2.1 损失函数Loss Function/代价函数cost function：——> 一次模型预测的好坏**

  (1) 0-1 损失函数 (0-1 loss function)

$$L(Y, f(X)) = \begin{cases} 1, & Y \neq f(X) \\ 0, & Y = f(X) \end{cases}$$

  (2) 平方损失函数 (quadratic loss function)

$$L(Y, f(X)) = (Y - f(X))^2$$

  (3) 绝对损失函数 (absolute loss function)

$$L(Y, f(X)) = |Y - f(X)|$$

  (4) 对数损失函数 (logarithmic loss function) 或对数似然损失抖数 elihood loss function)

$$L(Y, P(Y \mid X)) = -\log P(Y \mid X)$$

- **3.2.2 风险函数risk function/期望损失expected loss：——> 模型预测的平均好坏**
  - 即为**损失函数的关于联合分布期望**:

$$R_{\exp}(f) = E_p[L(Y, f(X))] = \int_{x \times y} L(y, f(x)) P(x, y) \mathrm{d}x \mathrm{d}y$$

- **3.2.3 经验风险最小化ERM 结构风险最小化SRM：**
  - 但是问题在于，计算**风险函数**知道联合分布$P(X, Y)$，但是这个是无法知道的，否则也不用学习了，所以监督学习就成了一个**病态问题（ill-formed problem）**，即无法精确的判断模型真正的效果
  - 因此此时**退而求其次取**其关于数据的平均损失称为**经验风险empirical risk/经验损失empirical loss**:

$$R_{\mathrm{emp}}(f) = \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right)$$

  - **结构风险structural risk minimization**:

$$R_{\mathrm{srm}}(f) = \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right) + \lambda J(f)$$

    其中$\lambda J(f)$是模型复杂度，是定义在假设空间F上的泛函
  - 而机器学习的目标，就是**经验风险或结构风险的最优化问题（找到最小值）**

- **3.3 算法 Algorithm：**
  - 实现上述最优化问题的过程。
    - **解析**
    - **数值：随机梯度下降SGD、Boosting**

4 模型评估 与 模型选择

4.1 训练误差与测试误差 --- > 模型评估

4.1.1 训练误差：

- **解释**：模型关于**训练集train_set**的平均损失

$$R_{\text{emp}}(f) = \frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right))$$

其中N是训练集容量

4.1.2 测试误差：

- **解释**：模型关于**测试集test_set**的平均损失

$$R_{\text{emp}}(f) = \frac{1}{N'}\sum_{i=1}^{N'} L\left(y_i, f\left(x_i\right)\right))$$

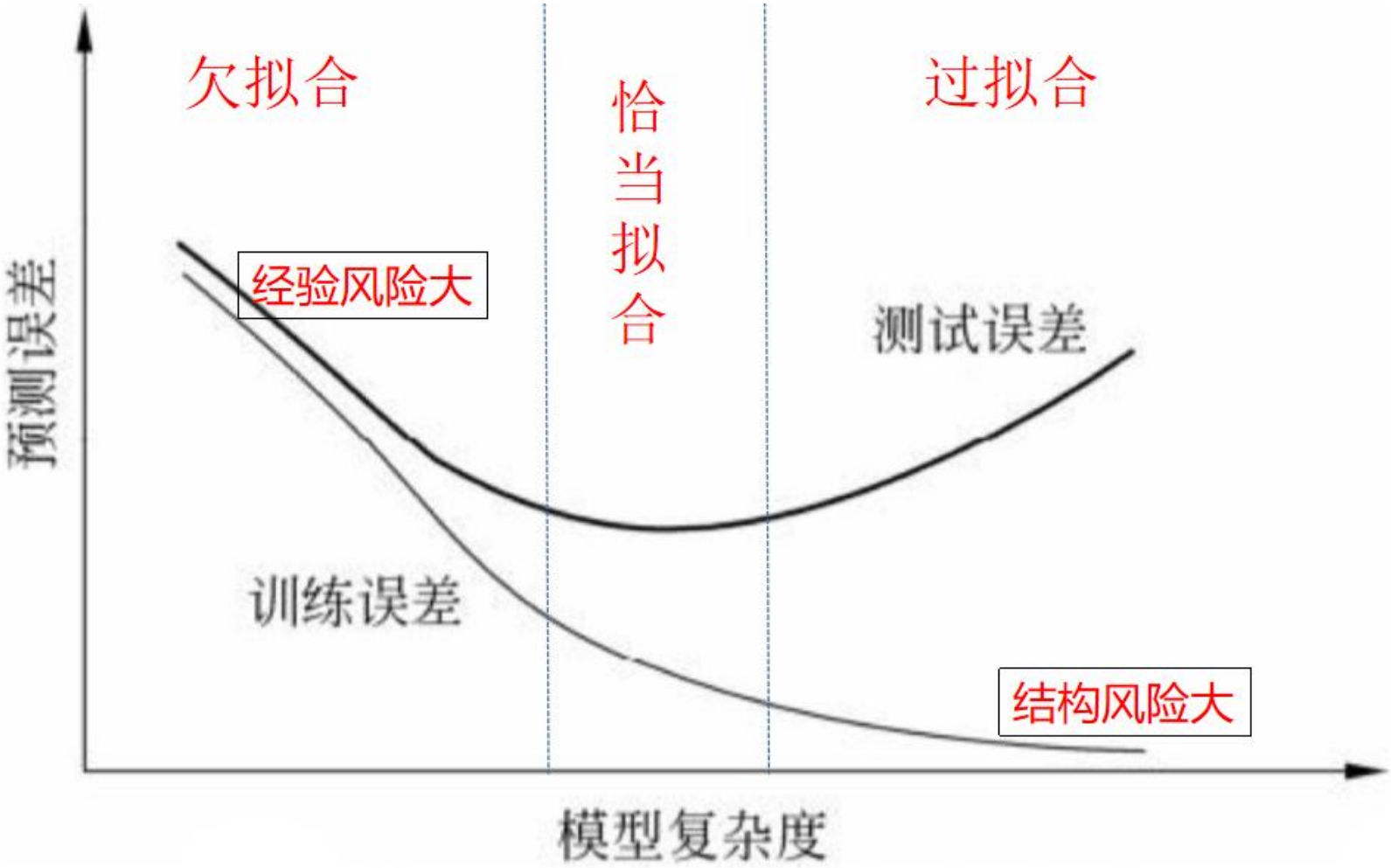
其中N'是测试集容量

4.2 过拟合over-fitting、欠拟合、恰当拟合 --- > 模型选择

4.2.1 过拟合：

过分追求经验风险最小化 即 $R_{\text{emp}}(f) = min\frac{1}{N}\sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right))$ ， 而求得的模型往往存在的现象

4.2.2 模型选择



训练误差和测试误差与模型复杂度的关系

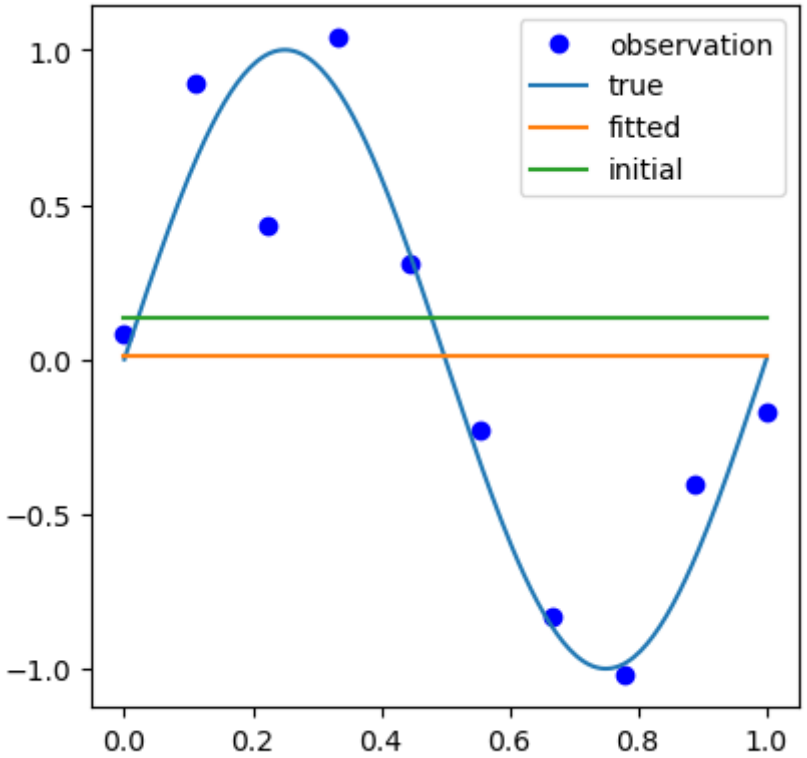
- **4.3 过拟合python代码练习： （每一次运行图都不一样）**

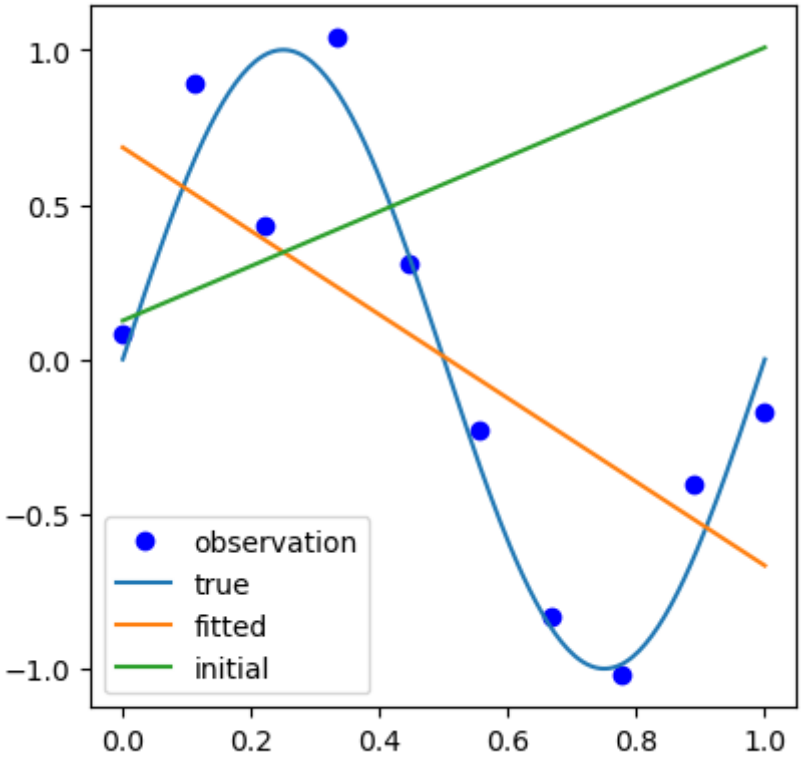
In [1]:

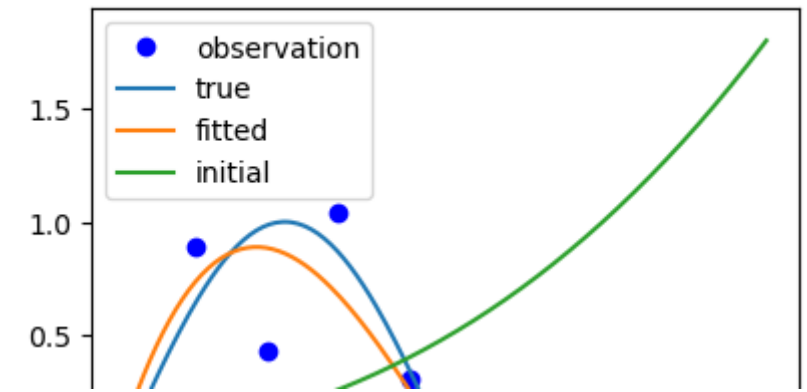
```
import numpy as np
from scipy.optimize import leastsq#最小二乘法
import matplotlib.pyplot as plt

x=np.linspace(0,1,10)
y=np.sin(2*np.pi*x)+np.random.normal(0,0.2,len(x))#正弦函数加上随机项

xx=np.linspace(0,1,200)
yy=np.sin(2*np.pi*xx)

#定义模型
def f_model(p,x):
    f_poly=np.poly1d(p)
    return f_poly(x)

#给定策略
def f_residual(p,x,y):
    return y-f_model(p,x)

def f_fitting(m):
    p_ini=np.random.rand(m+1)
    p_est=leastsq(f_residual,p_ini,args=(x,y))
    print('estimation:',p_est[0])
    plt.plot(x,y,'bo',label='observation')
    plt.plot(xx,yy,label='true')
    plt.plot(xx,f_model(p_est[0],xx),label='fitted')
    plt.plot(xx,f_model(p_ini,xx),label='initial')
    plt.legend()

fig=plt.figure(num=1,figsize=(10,10))
fig.add_subplot(221)
f_fitting(0)
fig.add_subplot(222)
f_fitting(1)
fig.add_subplot(223)
f_fitting(3)
fig.add_subplot(224)
f_fitting(9)
```

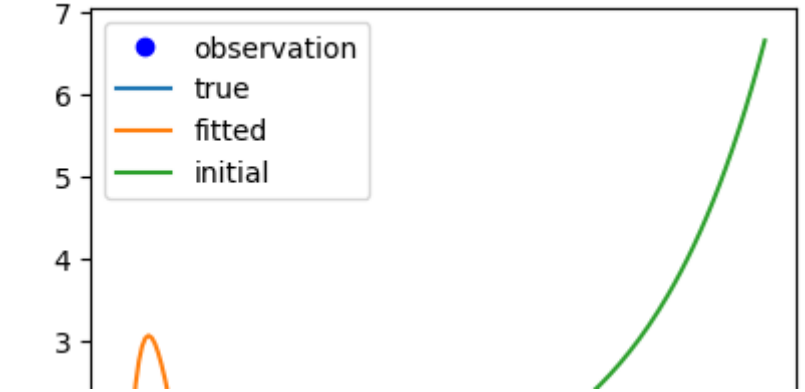
```
estimation: [0.00888225]
```

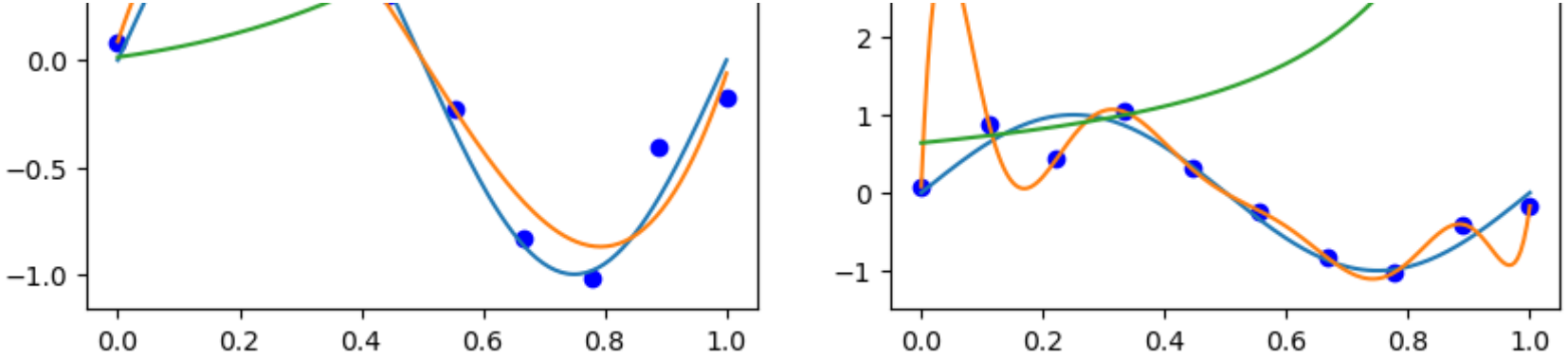
```
estimation: [-1.35148903  0.68462677]
estimation: [ 17.38297714 -26.04861166   8.51593156   0.08756397]
estimation: [ 5.15170142e+04 -2.42560671e+05  4.82290031e+05 -5.26409625e+05
  3.42819855e+05 -1.35223200e+05  3.11609546e+04 -3.77828779e+03
  1.83673252e+02  8.19723250e-02]
```











4.4 训练误差、测试误差与模型复杂度的关系python代码练习（每一次运行图都不一样）

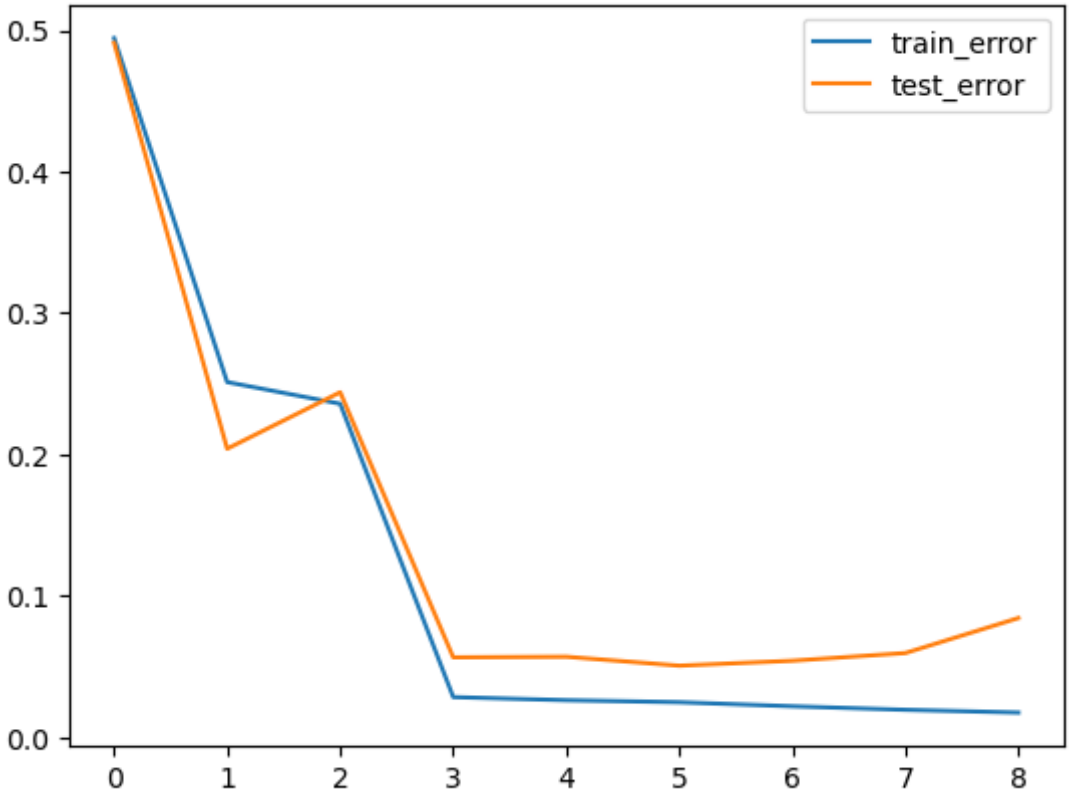
- **小结论**:
  - 模型训练误差可能会**大于**测试误差
  - 训练误差一定是**单调递减的**

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import leastsq #最小二乘法
from numpy import random

x=np.linspace(0,1,40)
y=np.sin(2*np.pi*x)+np.random.normal(0, 0.2,len(x))

train_set=list(range(len(x)))
test_set=[]
test_num=int(len(x)/2)

for i in range(test_num):
    temp=int(random.uniform(0,len(train_set)))
    test_set.append(train_set[temp])
    del(train_set[temp])

x_train=x[train_set]
y_train=y[train_set]

x_test=x[test_set]
y_test=y[test_set]

def f_model(p,x):
    f_poly=np.poly1d(p)#多项式拟合
    return f_poly(x)

def f_residual(p,x,y):
    return y-f_model(p,x)

train_error=[]
test_error=[]

#comp=list([0,2,4,6,8,10,12,14,16])

for m in range(9):
    p_ini=np.random.rand(m+1)
    p_est=leastsq(f_residual,p_ini,args=(x_train,y_train))
    train_error.append(np.dot(f_residual(p_est[0],x_train,y_train),f_residual(p_est[0],x_train,y_train))/test_num)
    test_error.append(np.dot(f_residual(p_est[0],x_test,y_test),f_residual(p_est[0],x_test,y_test))/test_num)

```

```
42
43 plt.plot(range(9), train_error, label='train_error')
44 plt.plot(range(9), test_error, label='test_error')
45 plt.legend()
```

Out[2]: <matplotlib.legend.Legend at 0x15ef8ee6c70>



5 正则化 与 交叉验证： 模型选择的方法

5.1 正则化/惩罚项：

- **解释**：基于经验风险、结构风险的最小化的策略进行模型选择。

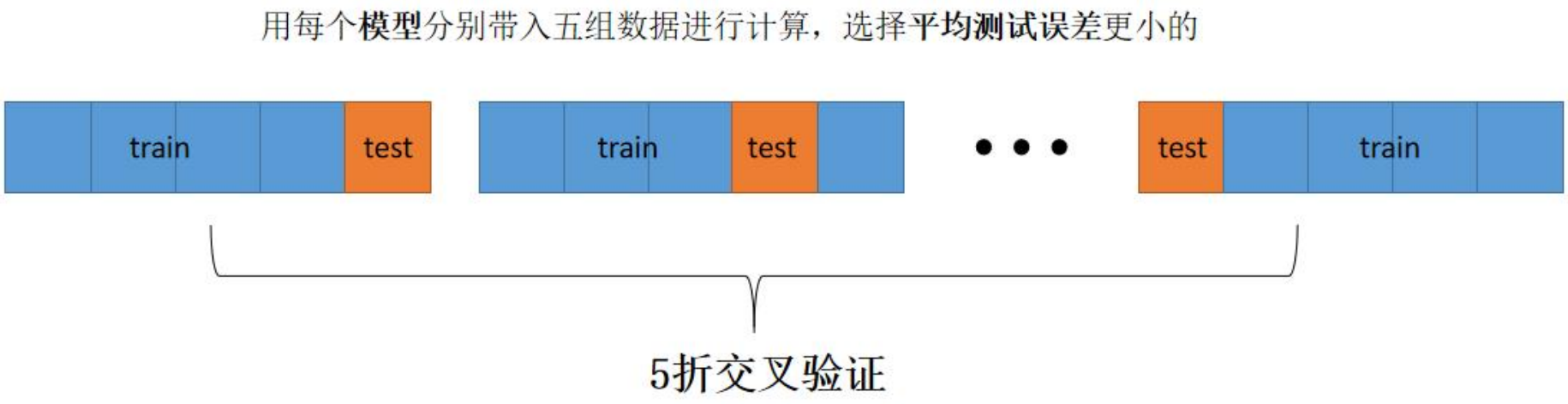
$$\min_{f \in F} \frac{1}{N} \sum_{i=1}^{N} L\left(y_i, f\left(x_i\right)\right) + \lambda J(f)$$

其中第1项是经验风险，第2项是正则项/惩罚项，$\lambda >= 0$为调整二者之间关系的系数

- **举例**：
  - 岭回归 ridge regression ， 正则项为 L_2 范数
  - Lasso回归 Lasso regression ， 正则项为 L_1 范数

5.2 交叉验证(CV) cross validation ：

- **解释**：在模型**训练、测试**过程中，数据的作用是平等且对称的
- **分类**：
  - (1) 简单交叉验证：随机分割训练集和测试集
  - (2) k 折交叉验证：把有 N 个数据分层成 k 份，其中 $k < N$ ，取其中 $k-1$ 个作为训练集，剩下的作为测试集，重复 k 次,找到误差最小的模型
  - (3)留一交叉验证：就是 $k = N$ 的 k 折交叉验证
    - 刀切法：Jackknife方法
    - bootstrap：随机抽样生成样本

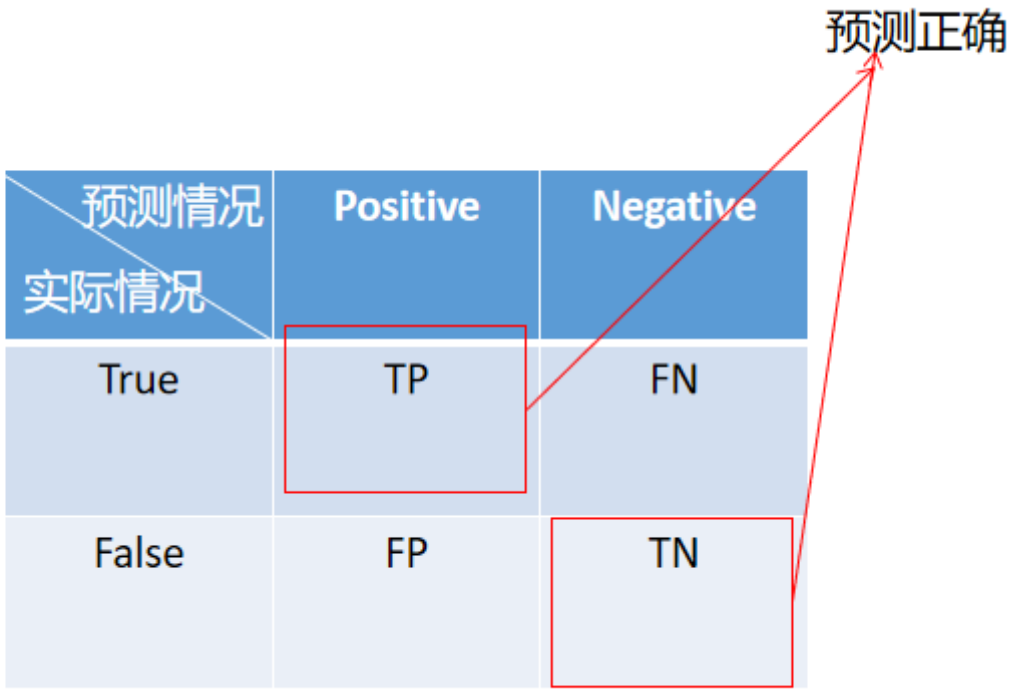


6 生成模型和判别模型

判别模型	生成模型
直接学习的是决策函数$f(X)$或者条件概率分布 $P(Y\|X)$	学习出**联合概率分布** $P(X,Y)$ **间接**得到的是条件概率分布 $P(Y\|X)$
线性回归、逻辑回归、感知机、决策树、支持向量机.....	朴素贝叶斯、HMM(隐马尔科夫链)、深度信念网络(DBN).....

7 模型评估的标准

7.1 分类模型



准确率 Accuracy：$\frac{TP+TN}{\text{总和}}$

召回率 recall：$\frac{TP}{TP+FN}$

精确率 precision：$\frac{TP}{TP+FP}$

F1 - score：$\frac{2}{\frac{1}{\text{精确率}}+\frac{1}{\text{召回率}}}$

In [ ]:
```
1
```