北京交通大学
BEIJING JIAOTONG UNIVERSITY

《大数据概论》
大数据分析与挖掘

鲍鹏
软件学院

目录

- 数据理解与特征工程
- 常用数据挖掘算法
  - 无监督学习
  - 监督学习
    - 线性回归
    - **Logistic**回归
- 高级数据建模技术
- 数据可视化技术

监督学习

- 监督学习从给定的训练数据集中学习一个函数，当新的数据到来时，可以根据该函数预测结果。
- 在监督式学习下，输入数据被称为“训练数据”，每组训练数据有一个明确的标识或结果，如，防垃圾邮件系统中的“垃圾邮件”和“非垃圾邮件”。
- 在建立模型时，监督式学习建立一个学习过程，将预测结果与“测试数据”的实际结果进行比较，不断调整预测模型，直到模型的预测结果达到一个预期的准确率。
- 常见的监督学习算法包括回归和分类。

监督学习-回归&分类

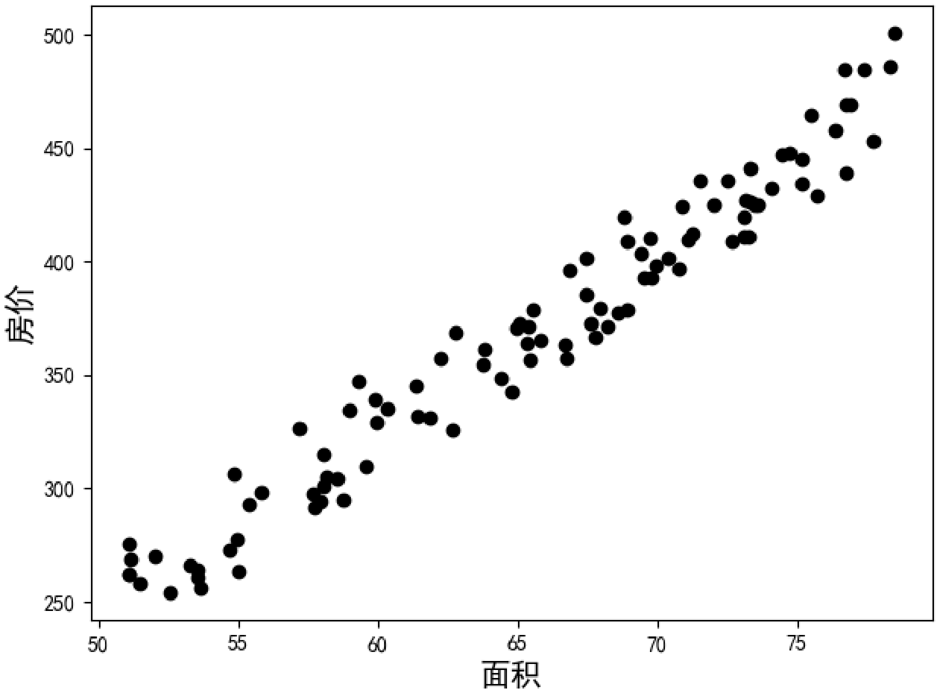
- 线性回归
- Logistic回归
- KNN（最近邻算法）
- 朴素贝叶斯

回归—线性回归

- 线性回归模型建立
- 梯度下降算法求解
- 回归模型评估
- 回归模型非线性变换

线性回归—回归模型建立

- 某市的房价走势图如下图所示，其中横坐标为面积，纵坐标为价格。假如现在随意告诉你一个房屋的面积，怎样才能预测（或计算）出其对应的价格呢？



线性回归—回归模型建立

- 房价与面积相关，根据常识房价的增长更优先符合如下的线性回归模型：

$$\hat{y} = h(x) = wx + b$$

其中，w为权重参数（Weight），b为偏置（Bias）或者截距（Intercept）。

- 当求解得到未知参数之后，预测模型也随之确定，即给定一个房屋面积，能够预测出其对应的房价。

线性回归—回归模型建立

- 当建立好一个模型后，下一步任务为通过给定的数据，即训练集（Training Data），来对模型$\boldsymbol{h(x)}$进行求解。

- 求解$\boldsymbol{h(x)}$的目的是希望输入面积后能够输出“准确”的房价$\hat{y}$。直接求解$\boldsymbol{h(x)}$较为困难，可从“准确”入手间接求解模型。

线性回归—回归模型建立

- 刻画“准确”：计算每个样本的真实房价与预测房价之间的均方误差。

$$J(w,b)=\frac{1}{2m}\sum_{i=1}^{m}(y^{(i)}-\hat{y}^{(i)})^2$$

$$\hat{y}^{(i)}=h(x^{(i)})=wx^{(i)}+b$$

- 其中，m表示样本数数量；$x^{(i)}$表示第i个样本，即第i个房屋的面积；$y^{(i)}$表示第i个房屋的真实价格；$\hat{y}^{(i)}$表示第i个房屋的预测价格。上式即为最小二乘损失函数。

线性回归—回归模型建立

- 当最小二乘损失函数$J(w,b)$取最小值时的参数w、b，即为所求的目标参数。当$J(w,b)$取最小值表示此时所有样本的预测值与真实值之间的误差（Error）最小。极端情况下所有预测值都等同于真实值，则$J(w,b)=0$。
- 如何求解模型$h(x)$的问题转换成了如何最小化函数$J(w,b)$的问题。而$J(w,b)$通常被称为目标函数（Objective Function）、代价函数（Cost Function）或损失函数。

线性回归—回归模型建立

- 影响房价的主要因素是面积，但其它因素同样也可能影响到房屋的价格。例如，房屋到学校的距离、到医院的距离和到大型商场的距离等。

- 假设存在影响房价的13个因素，机器学习中将这些“因素”称之为特征（**Feature**）。一般化为如下表示：

$$Y(W,X) = h(w,x) = w_1x_1 + w_2x_2 + \cdots w_nx_n$$

线性回归—回归模型建立

- 线性回归的目标函数如下：

$$J(W,b)=\frac{1}{2m}\sum_{i=1}^{m}(y^{(i)}-\hat{y}^{(i)})^2$$

$$\hat{y}^{(i)}=h(x^{(i)})=w_1x_1^{(i)}+\cdots+w_{13}x_{13}^{(i)}+b$$

其中$x_j^{(i)}$表示第i个样本的第j个特征属性，w为一个向量表示所有的权重，b为一个标量表示偏置。

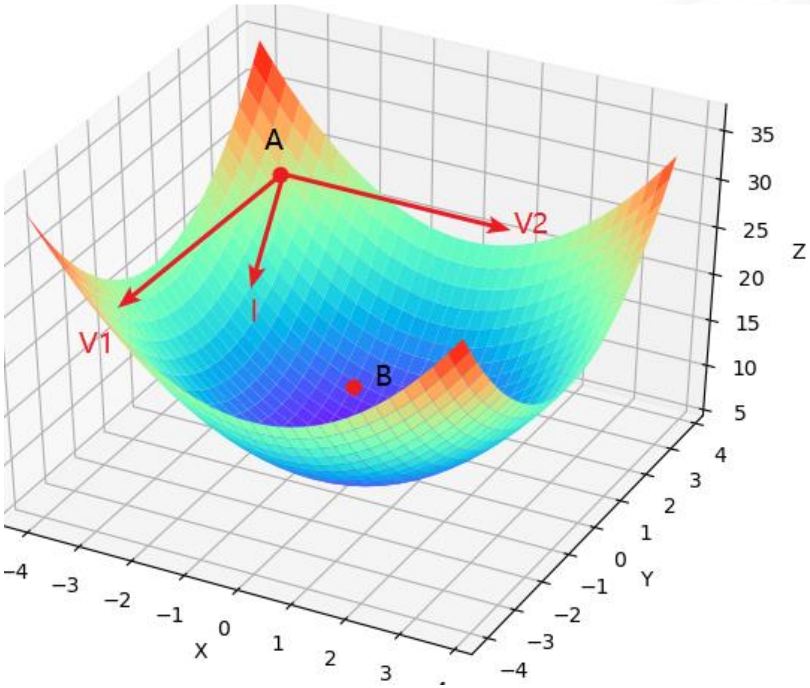
- 通过某种方法最小化目标函数$J(W,b)$后，即可求解出模型对应的参数。

线性回归—梯度下降算法

- 梯度下降
  - 梯度下降算法是求解线性回归的经典方法。
  - 梯度下降算法的目的是最小化目标函数。
  - 当目标函数取到（或接近）全局最小值时，即求解得到了模型所对应的参数。

线性回归—梯度下降算法

- 假设有一个山谷，并且你此时处于位置A处，那么请问以什么样的方向（角度）往前跳，你才能**最快**的到达谷底B处呢?



现在你大致有3个方向可以选择，沿着Y轴的V_1方向，沿着X轴的V_2方向以及沿着两者间的 L方向。

线性回归—梯度下降算法

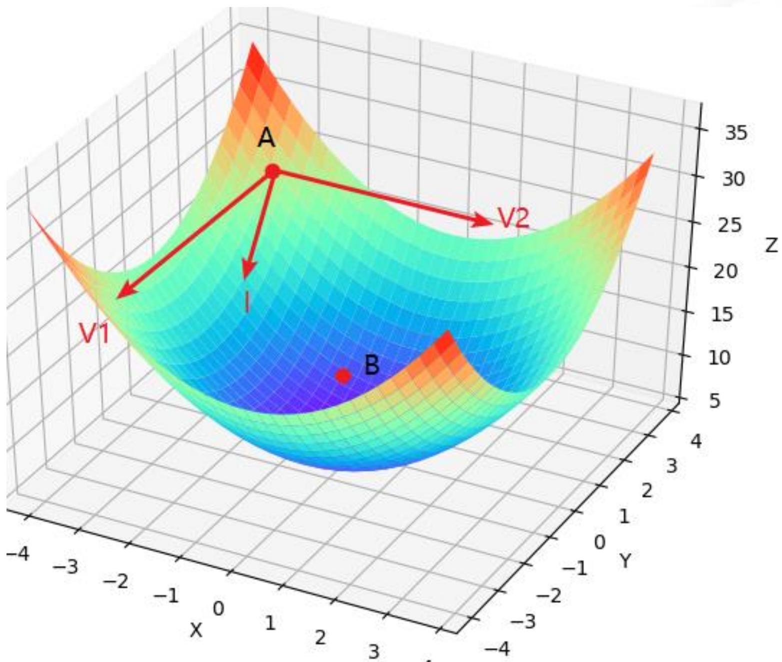
- 在一元函数中，$f(x)$在x_0处的导数反映的是$f(x)$在$x = x_0$处时的变化率；$|f'(x_0)|$越大，意味着$f(x)$在该处的变化率越大，即移动Δx后产生的函数增量Δy越大。同理，在二元函数$z = f(x, y)$中，为了寻找z在A处的最大变化率，应计算函数z在该点的方向导数：

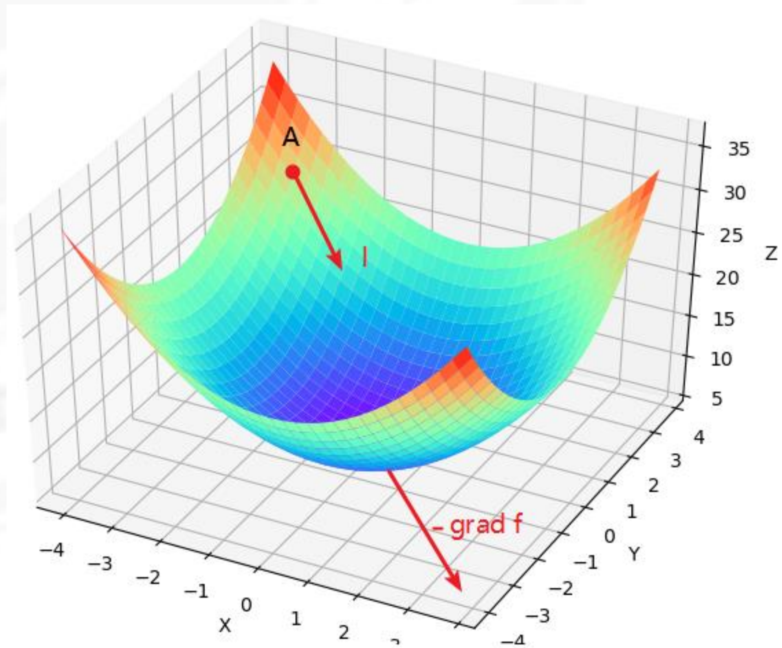
$$\frac{\partial f}{\partial \mathbf{l}} = \{\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\} \cdot \{cos\alpha, cos\beta\} = |\, gradf \,| \cdot |\, \mathbf{l} \,| \cdot \cos\theta$$

- 其中，l为单位向量， α、β分别为l与x和y轴的夹角，θ为梯度方向与 l 的夹角。
- 根据上式可知，若使方向导数取得最大值，则θ必须为0。由此可知，只有当某点处方向导数的方向与梯度的方向一致时，方向导数在该点才会取得最大的变化率。

线性回归—梯度下降算法

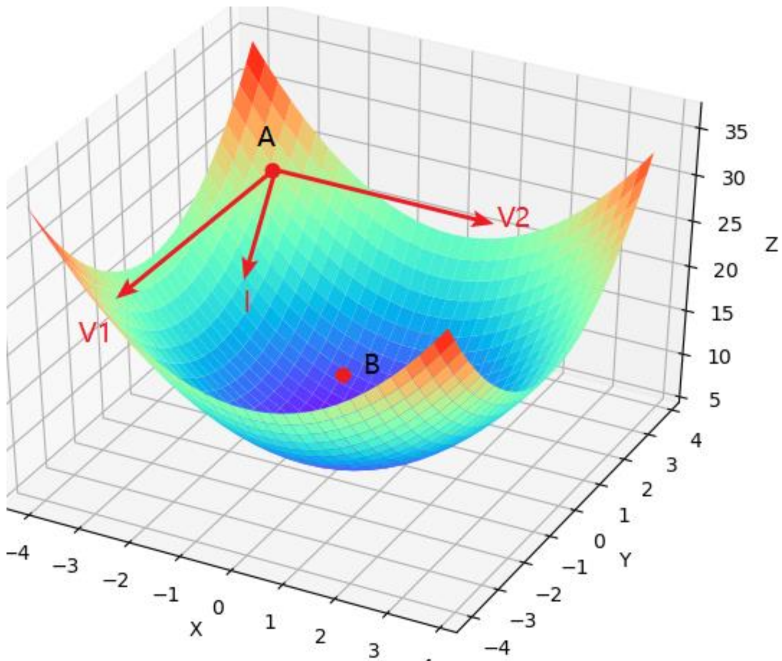
- 假设$z = x^2 + y^2+5$，A为(-3, 3, 23)，则$\frac{\partial z}{\partial x} = 2x, \frac{\partial z}{\partial y} = 2y$ 。此时点A处梯度的方向为(-6, 6)。故，在A点沿各个方向往前跳同样大小的距离时，只有沿着$(\frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2})$这个方向（单位化，且取了相反方向，需要负增量）才会产生最大的函数增量Δz。





线性回归—梯度下降算法

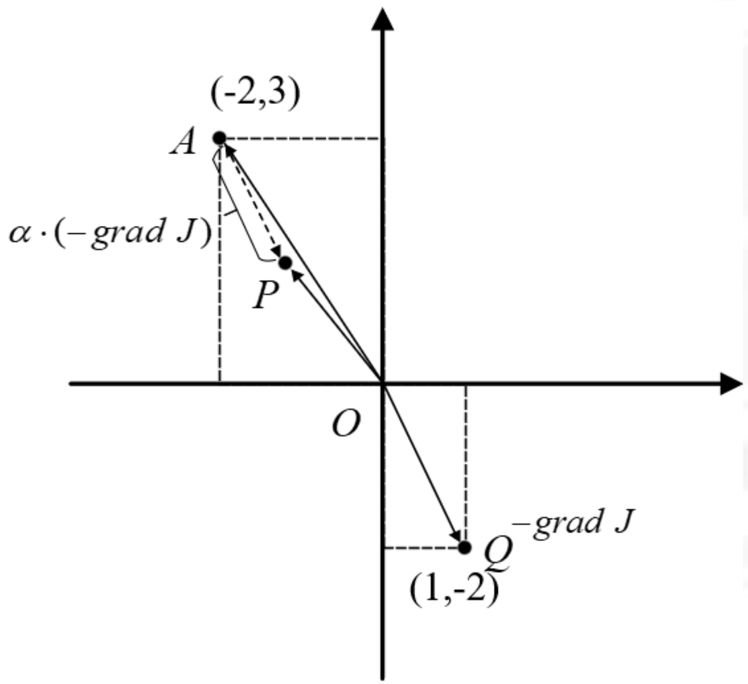
- 假设现在有一个模型的目标函数 $J(w_1, w_2) = w_1^2 + w_2^2 + 2w_2 + 5$（为了方便可视化此处省略了参数$b$，但原理都一样），其中$w_1, w_2$为待求解的权重参数，且随机初始化点A为初始权重值。下面就一步步的通过梯度下降法来进行求解。



设初始点 $A = (w_1, w_2) = (-2, 3)$，则 $J(-2, 3) = 24$，且点A第1次往前跳的方向为 $-grad\,J = -(2w_1, 2w_2 + 2) = (1, -2)$。

线性回归—梯度下降算法

- 如图所示，**OQ**为平面上梯度的反方向，**AP**为其平移后的方向，但是长度为之前的α倍。因此，根据梯度下降的原则，此时曲面上的**A**点就该沿着其梯度的反方向跳跃，而投影到平面则为**A**应该该沿着**AP**的方向移动。假定曲面上**A**点跳跃到了**P**点，那么对应在投影平面上就是图中的**AP**部分，同时权重参数也从**A**的位置更新到了**P**点的位置。



从图中可以看出，向量**AP,OA,OP**三者的关系为

$$\mathbf{OP = OA - PA}$$

可进一步改写为

$$\mathbf{OP = OA} - \alpha \cdot grad\ J$$

线性回归—梯度下降算法

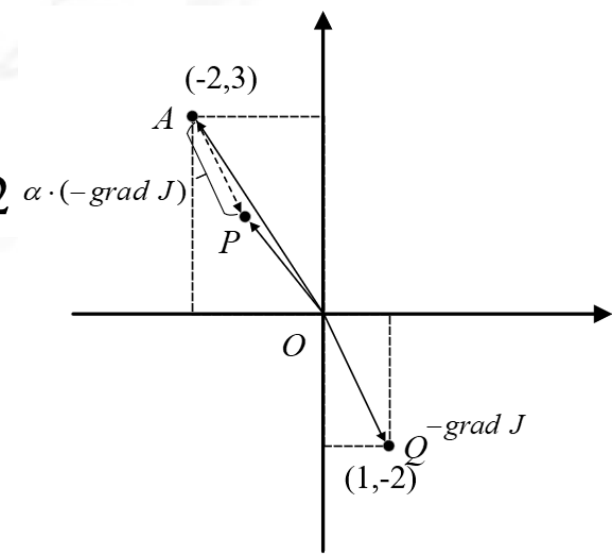
- 由于OP,OA本质上就是权重参数w_1, w_2更新后与更新前的值，所以便可以得出梯度下降的更新公式为：

$$W = W - \alpha \cdot \frac{\partial J}{\partial W}$$

- 其中 $\mathbf{W} = (w_1, w_2)$， $\frac{\partial J}{\partial W}$ 为权重的梯度方向，α 为步长，用来放缩每次向前跳跃的距离。同时，将上式代入具体数值后可以得出， 曲面上的点A在第1次跳跃后的着落点为：

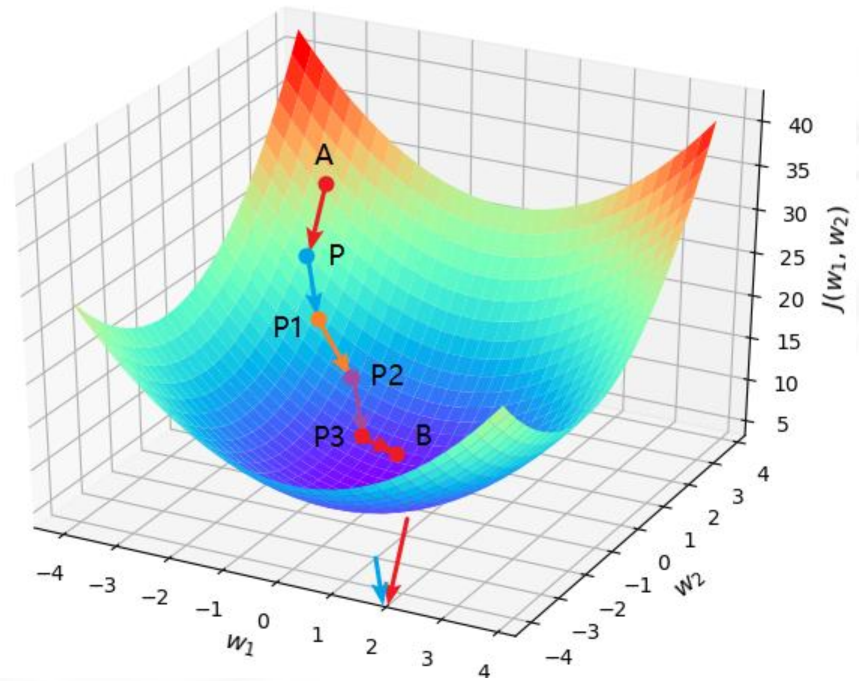
$$w_1 = w_1 - 0.1 \times 2 \times w_1 = -2 - 0.1 \times 2 \times (-2) = -1.6$$

$$w_2 = w_2 - 0.1 \times (2 \times w_2 + 2) = 3 - 0.1 \times (2 \times 3 + 2) = 2.2$$



线性回归—梯度下降算法

- 此时，权重参数便从(-2,3)更新到了(-1.6,2.2)。当然其目标函数$J(w_1, w_2)$也从24更新到了16.8。至此，我们便详细的完成了1轮梯度下降的计算。当跳跃到新的点之后，又可以再次利用梯度下降算法进行跳跃，直到跳到谷底（或附近），如下图所示。



线性回归—梯度下降算法

- 目标函数推导—求解梯度
  - 设$y^{(i)}$表示第i个样本的真实值；$\hat{y}^{(i)}$表示第i个样本的预测值；W表示权重（列）向量，W_j表示其中一个分量；X表示数据集形状为$m \times n$，m为样本个数，n为特征维度；$x^{(i)}$ 为一个（列）向量，表示第i个样本，$x_j^{(i)}$为第j维特征。
  - 目标函数如下：

$$J(W,b)=\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)}-\hat{y}^{(i)}\right)^2=\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)}-(W^T x^{(i)}+b)\right)^2$$

线性回归—梯度下降算法

- 目标函数推导—求解梯度
  - 目标函数关于W_j的梯度求解过程为：

$$\frac{\partial J}{\partial W_j} = \frac{\partial}{\partial W_j}\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)} - (W_1x_1^{(i)} + W_2x_2^{(i)} \cdots W_nx_n^{(i)} + b)\right)^2$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - (W_1x_1^{(i)} + W_2x_2^{(i)} \cdots W_nx_n^{(i)} + b)\right)\cdot(-x_j^{(i)})$$

$$= \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - (W^Tx^{(i)} + b)\right)\cdot(-x_j^{(i)})$$

  - 目标函数关于b的梯度求解过程为：

$$\frac{\partial J}{\partial b} = \frac{\partial}{\partial b}\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)} - (W^Tx^{(i)} + b)\right)^2$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - (W^Tx^{(i)} + b)\right)$$

线性回归—梯度下降算法

- 目标函数推导—求解梯度
  - 目标函数关于参数的梯度计算公式：

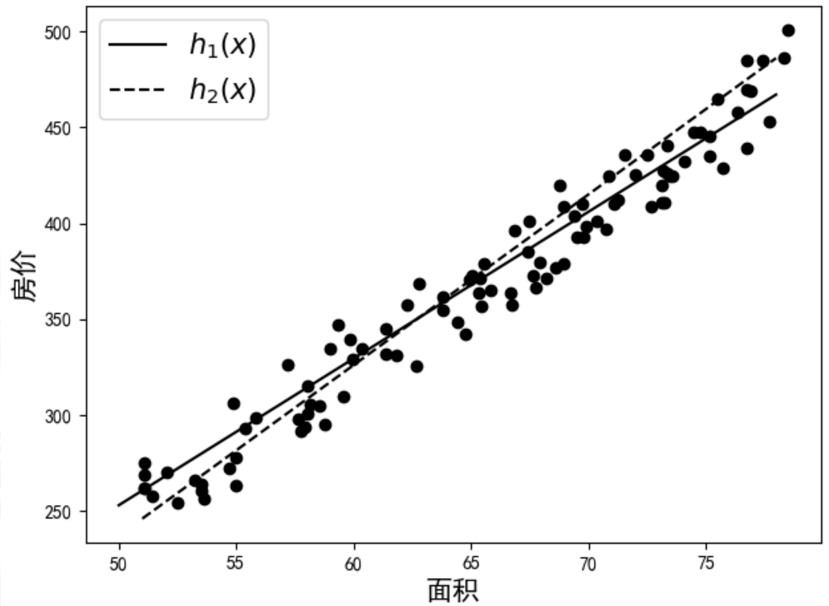
$$J(W,b)=\frac{1}{2m}\sum_{i=1}^{m}\left(y^{(i)}-(W^{T}x^{(i)}+b)\right)^{2}$$

$$\frac{\partial J}{\partial W_j}=\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}-(W^{T}x^{(i)}+b)\right)\cdot(-x_j^{(i)})$$

$$\frac{\partial J}{\partial b}=-\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)}-(W^{T}x^{(i)}+b)\right)$$

线性回归—模型评估

- 以房价预测为例，假设你求解得到了下图所示的两个模型$h_1(x)$与$h_2(x)$，那么应该选哪一个呢？



线性回归—模型评估

- 评估指标
  - 在回归任务中，常见的评估指标（Metric）有平均绝对误差（Mean Absolute Error, MAE）、均方误差（Mean Square Error, MSE）、均方根误差（Root Mean Square Error, RMSE）、平均绝对百分比误差（Mean Absolute Percentage Error, MAPE）和决定系数（Coefficient of Determination），其中使用最为广泛的是**MAE**和**MSE**。

线性回归—模型评估

- 平均绝对误差（MAE）
  - MAE用来衡量预测值与真实值之间的平均绝对误差。

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

  - $MAE \in [0,+\infty)$，MAE的值越小表明模型越好。
- 均方根误差（ RMSE ）
  - MSE的基础之上开根号得到，定义如下：

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

  - $RMAE \in [0,+\infty)$，RMAE的值越小表明模型越好。

线性回归—模型评估

- 均方误差（MSE）
  - MSE用来衡量预测值与真实值之间的误差平方。

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  - $MSE \in [0,+\infty)$，MSE的值越小表明模型越好。

线性回归—非线性变换

- 线性回归模型$Y(W,X) = w_1x_1 + w_2x_2 + \cdots w_nx_n$，对参数W而言，输入X($x_1$，$x_2$...$x_n$)并非一定是线性函数，可以通过一系列的基函数$\phi_i(X)$，对输入进行线性变换:

$$Y(W,X) = \sum_{i=1}^{n} w_i \phi_i(X)$$

线性回归—非线性变换

- 多项式函数
  - 多项式函数是由常数与自变量经过有限次乘法与加法运算得到的。
  - 定义如下：通过一系列的基函数$\phi_i(X)$，对输入进行线性变换：

$$\phi(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_0$$

  - 其中，$a_i(i = 0, 1, \ldots n)$是常数，当n=1时，多项式函数为一次函数 $\phi(x) = a_1 x + a_0$。

线性回归—非线性变换

- 高斯函数
  - 高斯函数公式如下：

$$\boldsymbol{\phi}(\boldsymbol{x}) = \boldsymbol{a} \cdot \boldsymbol{exp}\left(-\frac{(\boldsymbol{x}-\boldsymbol{b})^2}{\boldsymbol{c}^2}\right)$$

  - 其中， a， b， c 均是实常数，且a>0。

- Sigmod函数
  - Sigmod函数是一个常见的S型函数，公式如下：

$$\boldsymbol{\phi}(\boldsymbol{x}) = \frac{\mathbf{1}}{\mathbf{1}+\mathrm{e}^{-\mathrm{x}}}$$

监督学习—分类

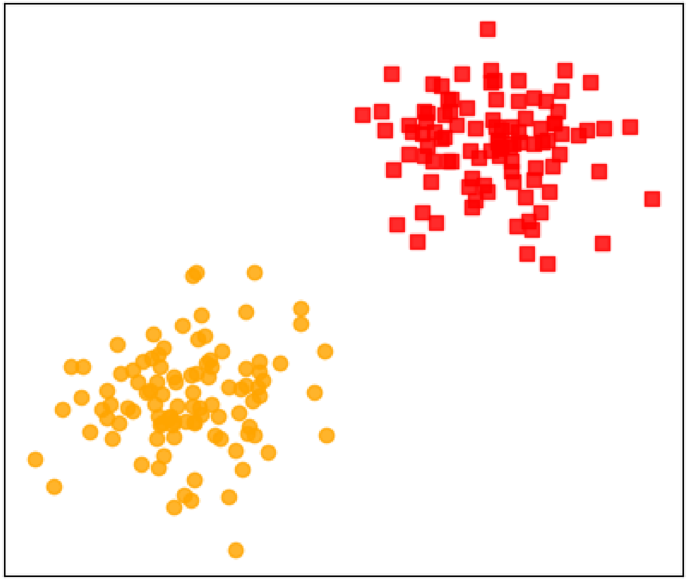
- 分类的目的是提出一个分类函数或分类模型（即分类器），通过分类器将数据对象映射到某一个给定的类别中。
- 数据分类可分为两个步骤：
  - 建立模型，用于描述给定的数据集合。通过分析由属性描述的数据集合来建立反映数据集合特性的模型。该步骤称作有监督的学习，导出模型基于训练数据集，训练数据集是已知类标记的数据对象。
  - 使用模型对数据对象进行分类。首先评估模型的分类准确度，若模型准确度可接受，用其来对未知类标记的对象进行分类。

Logistic 回归

- Logistic回归(逻辑回归)一般用于分类问题，而其本质是线性回归模型，只是在回归的连续值结果上加了一层函数映射。
  - 模型建立
  - 模型求解
  - 模型评估

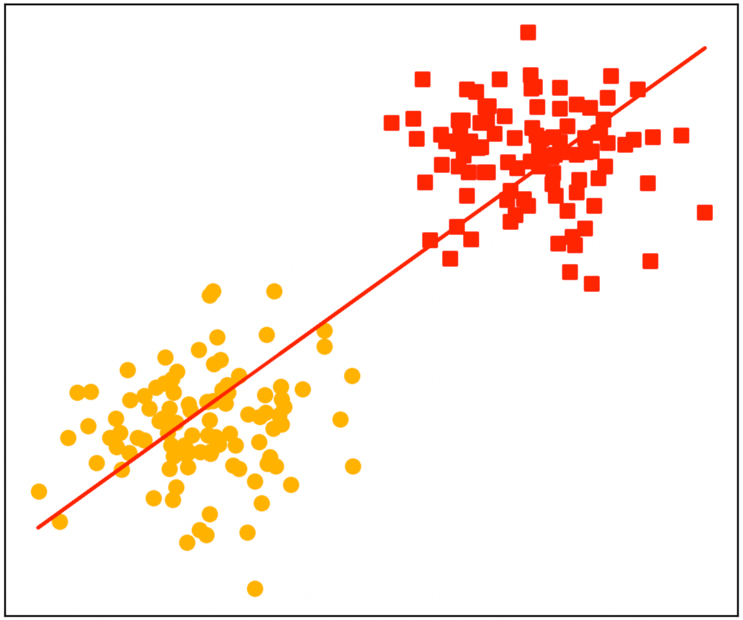
Logistic 回归—模型建立

- 如图所示，现在有两堆样本点，需要建立一个模型来对新输入的样本进行预测，判断其应属于那个类别，即二分类问题（**Binary Classification**）。该问题是否能通过前面的线性回归模型来解决？



Logistic 回归—模型建立

- 利用线性回归模型可得到一条向右倾斜的直线，而二分类问题需要一条向左倾斜的直线区分两个区域。



解决思路：通过建立一个模型来预测每个样本点属于其中一个类别的概率p，如果$p > 0.5$则可认为该样本点属于某个类别。

Logistic 回归—模型建立

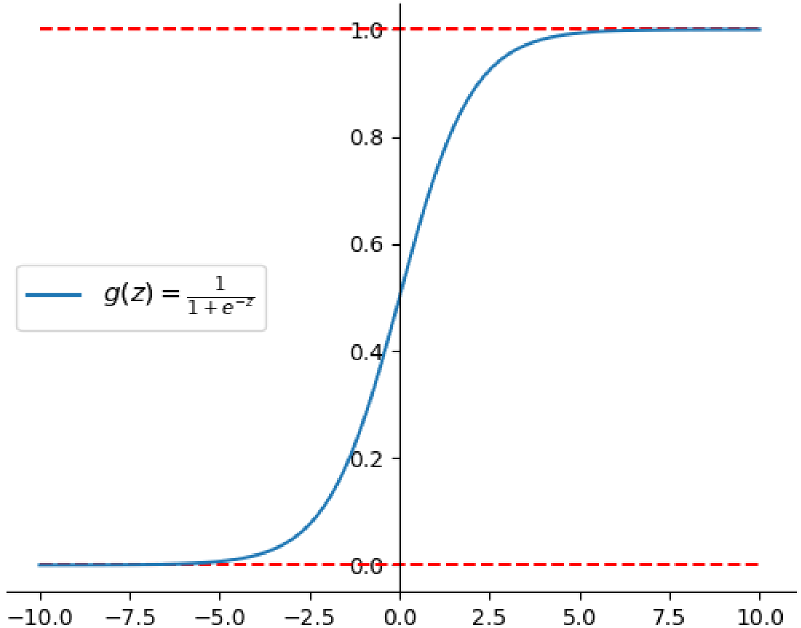
- 在线性回归中，通过建模$h(x) = wx + b$来对新样本进行预测，其输出可为任意实数。但此方法需得到一个样本所属类别的概率，直接手段为通过一个函数$g(z)$，将$h(x)$映射至[0,1]的范围。因此，可得到逻辑回归中的预测模型：

$$\hat{y} = h(x) = g(wx + b)$$

其中，w, b为未知参数；$h(x)$称为假设函数，当$h(x_i)$大于某个值（通常设为0.5）时，便可认为样本x_i属于正类，反之则属于负类。同时，将$wx + b = 0$称为两个类别间的决策面（Decision Boundary）。

Logistic 回归—模型建立

- 映射函数
  - 函数$g(z)$将特征的线性组合$z = Wx + b$映射到区间[0,1]。
  - $g(z)$ 也被称为Sigmoid函数，函数图像如下所示：



其中$z \in (-\infty, +\infty)$，而之所以选择Sigmoid的原因在于：①连续光滑且处处可导；②Sigmoid关于点(0，0.5)中心对称；③Sigmoid求导过程简单，其导数为 $g'(z) = g(z) \cdot \left(1 - g(z)\right)$。

Logistic 回归—模型建立

- 通过Sigmod函数二值化后，Logistic回归模型如下：

$$h_w(X) = Y(W, X) = g\left(W^T X\right) = \frac{1}{1 + e^{-W^T X}}$$

- Logistic回归模型通常用于二分类，面对多分类问题可以使用Softmax函数将连续的回归结果映射为多分类标签。

$$Softmax(z_i) = \frac{e^{z_i}}{\sum_{c=1}^{C} e^{z_c}}$$

Logistic 回归—模型求解

- 求解Logistic回归
  - 与线性回归相同，通过目标函数来刻画预测标签与真实标签之间的差距。当最小化目标函数后，能得到需要求解的参数：

$$J(w,b) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1-y^{(i)}) \log(1-h(x^{(i)}))\right]$$

$$h(x^{(i)}) = g(wx^{(i)} + b)$$

其中，m表示样本总数，$x^{(i)}$表示第i个样本，$y^{(i)}$表示第i个样本的真实标签，$h(x^{(i)})$表示第i个样本为正类的预测概率。

Logistic 回归—模型求解

- 求解梯度
  - 通过梯度下降算法可以最小化某个目标函数。当目标函数取得（或接近）最小值时，可得到目标函数中对应的未知参数。
  - 求解目标函数$J(W,b)$关于参数W的梯度：

$$
\begin{aligned}
\frac{\partial J}{\partial W_j} &= -\frac{\partial}{\partial W_j}\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log h(x^{(i)})+(1-y^{(i)})\log(1-h(x^{(i)}))\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\frac{h'(x^{(i)})}{h(x^{(i)})}+(1-y^{(i)})\frac{-h'(x^{(i)})}{1-h(x^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\frac{g(z^{(i)})(1-g(z^{(i)}))}{g(z^{(i)})}x_j^{(i)}-(1-y^{(i)})\frac{g(z^{(i)})(1-g(z^{(i)}))}{1-g(z^{(i)})}x_j^{(i)}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}(1-g(z^{(i)}))-(1-y^{(i)})g(z^{(i)})\right]x_j^{(i)} \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}-h(x^{(i)})\right]x_j^{(i)}
\end{aligned}
$$

Logistic 回归—模型求解

- 求解梯度
  - 求解目标函数$J(W,b)$关于参数b的梯度：

$$
\begin{aligned}
\frac{\partial J}{\partial b} &= -\frac{\partial}{\partial b}\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log h(x^{(i)}) + (1-y^{(i)})\log(1-h(x^{(i)}))\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\frac{h'(x^{(i)})}{h(x^{(i)})} + (1-y^{(i)})\frac{-h'(x^{(i)})}{1-h(x^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\frac{g(z^{(i)})(1-g(z^{(i)}))}{g(z^{(i)})} - (1-y^{(i)})\frac{g(z^{(i)})(1-g(z^{(i)}))}{1-g(z^{(i)})}\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}(1-g(z^{(i)})) - (1-y^{(i)})g(z^{(i)})\right] \\
&= -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)} - h(x^{(i)})\right]
\end{aligned}
$$

Logistic 回归—模型评估

- 在分类任务中，常见的模型评价指标有：准确率（Accuracy）、精确率（Precision）、召回率（Recall）与F值（F_{score}），其中应用最为广泛的是准确率和召回率。

真实＼预测	P	N
P	TP	FN
N	FP	TN

$$Accurcay = \frac{TP+TN}{TP+FP+FN+TN}$$

$$Precision = \frac{TP}{TP+FP}$$

$$Recall = \frac{TP}{TP+FN}$$

$$F_{score} = (1+\beta^2)\frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

KNN（最近邻算法）

- K最近邻（K-nearest beighbor，KNN）分类算法。KNN方法的出发点：如果一个样本在特征空间中的k个最相似（特征空间中最近邻）的样本中的大多数属于某一个类别，则该样本也属于这个类别，并具有这个类别样本的特性。
  - KNN思想
  - KNN原理

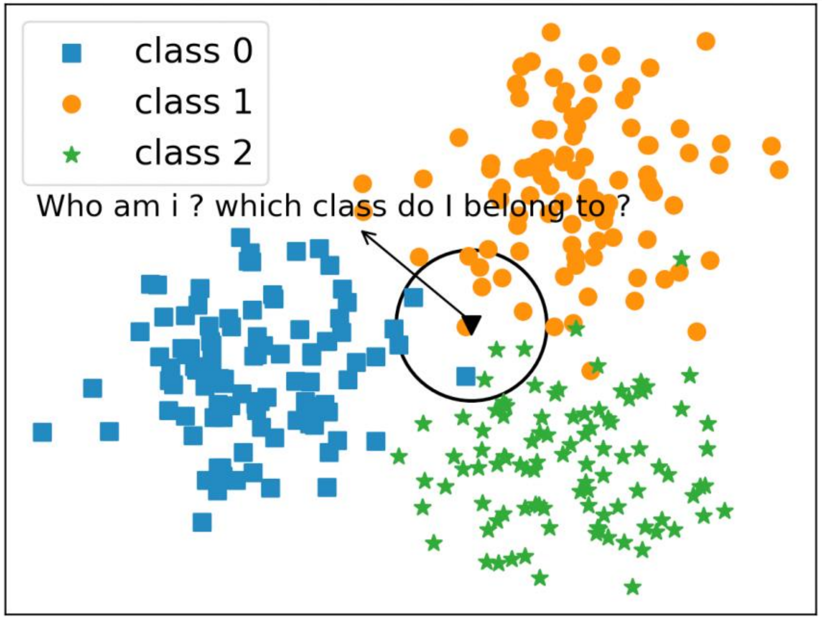
KNN思想

- 某天，你和几位朋友准备去外面聚餐，但是就晚上吃什么菜一直各持己见。最后，无奈的你提出用多数服从少数的原则来进行选择。于是你们每个人都将自己想要吃的东西写在了纸条上，最后的统计情况是：3个人赞成吃火锅、2个人赞成吃炒菜、1个人赞成吃自助。当然，最后你们一致同意按照多数人的意见去吃了火锅。

- 尽管吃火锅跟K近邻没关系，但是整个决策的过程却完全体现了K最近邻算法的决策过程。

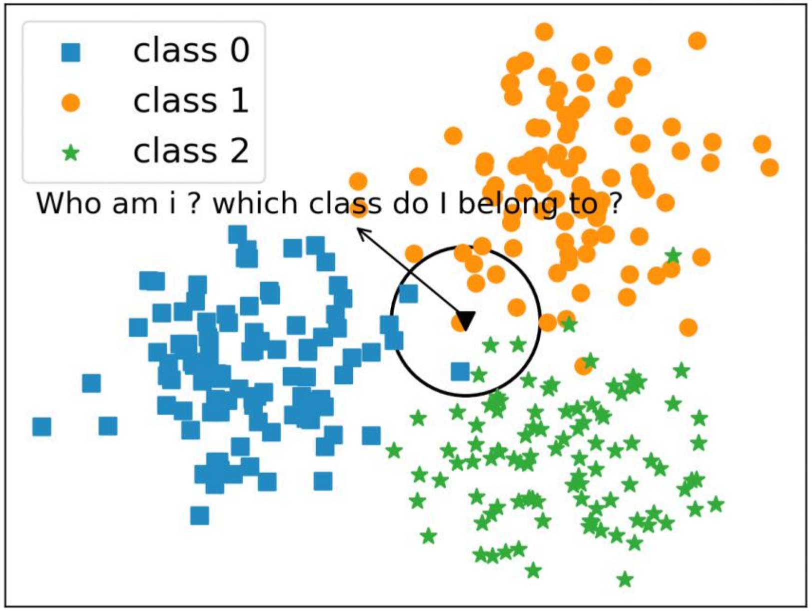
KNN原理

- 如图所示，彩色样本点为原始的训练数据，并且包含了0, 1, 2 这 3 个类别（分别为图中不同形状的样本点）。现在拿到一个新的样本点（图中黑色倒三角），需要对其所属类别进行分类。



KNN原理

- **KNN工作原理**
  - 首先 KNN 会确定一个 K 值；然后选择离自己最近的 K 个样本点；最后，根据投票的规则（Majority Voting Rule）确定新样本应该所属的类别。



如图所示，示例中选择了离三角形样本点最近的 14 个样本点（方形4个、圆形7个、星形3个）。

离三角形样本点最近的 14 个样本中，数量最多的为圆形样本，所以 KNN 算法将新样本归类为类别 1。

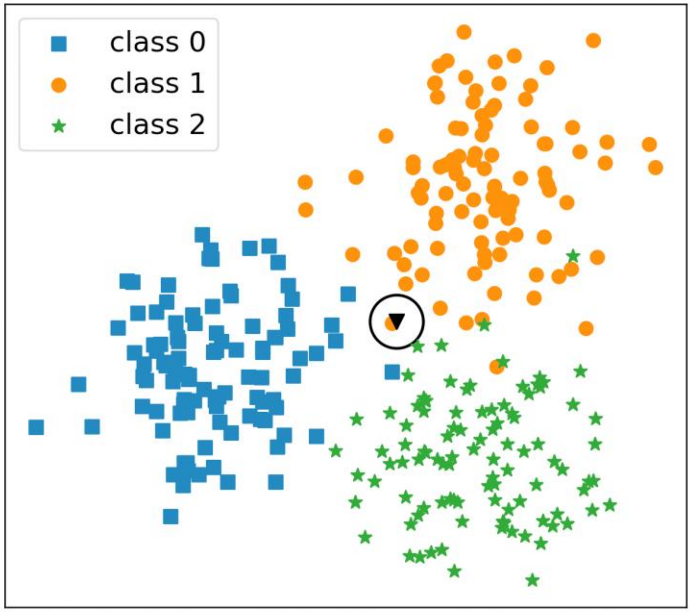
KNN原理

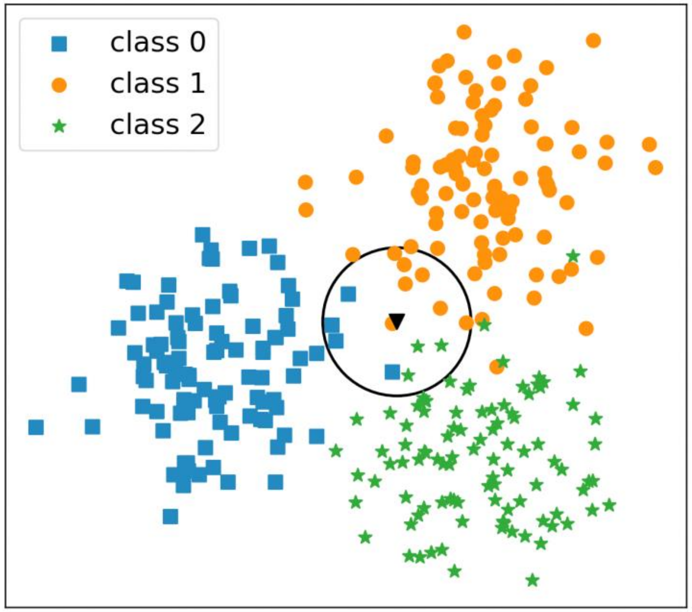
- **KNN算法的三个步骤**
  - 首先确定一个 K 值，用于选择离自己（三角形样本点）最近的样本数。
  - 然后选择一种度量距离，用于计算得到离自己最近的K个样本（例如，使用最为广泛的欧氏距离）。
  - 最后确定一种决策规则，用于判定新样本所属类别（例如，示例中采用了基于投票的分类规则）。

- KNN算法的三个步骤对应了三个超参数的选择。通常，对于决策规则的选择基本上都是采用基于投票的分类规则。

KNN原理

- K 值的选择会极大程度上影响 KNN 的分类结果。





若分类过程中选择较小的K值，将会使得模型的训练误差减小而使得模型的泛化误差增大，即模型过于复杂而产生了过拟合现象。

KNN原理

- 距离度量
  - 在样本空间中，任意两个样本点之间的距离可看作是两个样本点之间相似性的度量。两个样本点的距离越近，意味着这两个样本点越相似。一般情况下KNN使用欧式距离，也可采用其它距离，例如更一般的L_p距离。
  - 设训练样本 $X=\{x^{(1)},x^{(2)},\dots,x^{(n)}\}$，其中 $x^{(i)}=\{x_1^{(i)},x_2^{(i)},\dots,x_m^{(i)}\}\in R^m$，即每个样本包含$m$个特征维度，$L_p$距离定义为：

$$L_p(x^{(i)},x^{(j)})=\left(\sum_{k=1}^{m}|x_k^{(i)}-x_k^{(j)}|^p\right)^{\frac{1}{p}};p\geq 1$$

朴素贝叶斯

- 朴素贝叶斯分类算法利用统计学中的贝叶斯定理来预测类成员的概率。给定一个样本，计算该样本属于一个特定的类的概率。
- 朴素贝叶斯分类基于的假设：每个属性之间都是相互独立，并且每个属性对分类问题产生的影响相同。

朴素贝叶斯—基本概念

- 先验概率
  - 先验概率指根据历史经验得出来的概率。
  - 假设在某二分类数据集中，正样本有4个，负样本有 6个，那么通过该数据集能够学习到的先验知识为任取一个样本，其为正样本的可能性为40%，负样本的可能性为60%，该先验知识中的可能性被称为先验概率。

朴素贝叶斯—基本概念

- 后验概率
  - 后验概率指通过贝叶斯公式推断得到的结果。
  - 上述例子中，不能因为负样本出现的可能性为60%就判定任意取出的样本为负样本。先验知识只能先取得一个大致的判断，而事实情况需要根据先验概率和条件概率来进行计算。

朴素贝叶斯—基本概念

- 极大后验概率
  - 极大后验概率指在所有后验概率中选择其中最大的一个。
  - 上述例子中，根据先验概率和条件概率可以计算出每个样本属于正样本还是负样本的后验概率。在判断该样本属于何种类别时，应挑选后验概率最大的类别。

朴素贝叶斯—基本概念

- 极大似然估计
  - 极大似然估计是用来估计能使得当前已知结果最有可能发生的模型参数的过程。
  - 例如，某次抛硬币的结果为正面4次，反面6次。那么什么样的模型参数能够使得这一结果最可能发生呢？
  - 假设p为正面向上的概率，令该结果最可能发生只需最大化下式即可：

$$\binom{10}{4} p^4 (1-p)^6$$

朴素贝叶斯—原理

- 贝叶斯公式
  - 假设B为最终的分类标签，A为一系列的特征属性，在使用朴素贝叶斯进行样本分类的时候，实际计算的应为每个样本在当前的特征取值为A的情况下，其属于类别B的概率。

$$P(B|A)=\frac{P(AB)}{P(A)}$$

  - 在实际情况中，**A, B** 之间的联合概率分布 **P(A, B)** 是未知的，因此将其转换成先验概率分布**P(A)**乘以条件概率分布**P(B|A)**来得到联合分布，即：

$$P(B|A)=\frac{P(B)P(A|B)}{P(A)}$$

朴素贝叶斯—原理

- 通过学习数据的先验分布，再学习数据的条件概率分布，即可得到联合概率分布P(X,Y)。对于每个类别，其先验概率分布为：

$$P(Y=c_k)=\frac{\#c_k}{m},k=1,2,\ldots,K$$

其中，$\#c_k$表示该类别中包含的样本数，m表示所有的样本总数。

- 对于已知类标下的条件概率分布为：

$$P(X=x\mid Y=c_k)=P(X^{(1)}=x^{(1)},\ldots,X^{(n)}=x^{(n)}\mid Y=c_k)$$

其中，$x^{(i)}$表示第i个特征的取值。

朴素贝叶斯—原理

- 由于在实际情况中条件概率未知，朴素贝叶斯对条件概率分布进行了**条件独立性假设**，即**P(AB|D)=P(A|D)P(B|D)**，此为“朴素”一词的由来。

$$P(X=x|Y=c_k)=P(X^{(1)}=x^{(1)},\ldots,X^{(n)}=x^{(n)}|Y=c_k)$$

$$\Downarrow$$

$$P(X=x|Y=c_k)=\prod_{i=1}^{n}P(X^{(i)}=x^{(i)}|Y=c_k)$$

朴素贝叶斯—原理

- 在已知特征属性$X = x$的条件下，其属于类别$Y = c_k$的后验概率为：

$$P(Y=c_k \mid X=x)=\frac{P(X=x \mid Y=c_k)P(Y=c_k)}{\sum_{k=1}^{K} P(X=x \mid Y=c_k)P(Y=c_k)}$$

$$\Big\Downarrow \text{基于条件独立性假设变换}$$

$$P(Y=c_k \mid X=x)=\frac{P(Y=c_k)\prod_{i=1}^{n} P(X^{(i)}=x^{(i)} \mid Y=c_k)}{\sum_{k=1}^{K} P(Y=c_k)\prod_{i=1}^{n} P(X^{(i)}=x^{(i)} \mid Y=c_k)}$$

朴素贝叶斯—原理

- 朴素贝叶斯分类器：
  - 计算出任意样本属于类别c_k的概率后，选择其中概率最大者作为其分类的类标。
  - 朴素贝叶斯分类器可以表示为：

$$y = \arg\max_{c_k} = \frac{P(Y=c_k)\prod_{i=1}^{n} P(X^{(i)}=x^{(i)} \mid Y=c_k)}{\sum_{k=1}^{K} P(Y=c_k)\prod_{i=1}^{n} P(X^{(i)}=x^{(i)} \mid Y=c_k)}$$

  - 进一步可得：

$$y = \arg\max_{c_k} P(Y=c_k)\prod_{i=1}^{n} P(X^{(i)}=x^{(i)} \mid Y=c_k)$$