什么是机器学习

一、机器学习

机器学习，是近年来比较火的一个概念，它到底是怎么一回事，让我们来一探究竟。

首先我们举一个例子来进行形象的说明。

假如，我有一套房子80㎡，我要出售。那我应该卖多少钱呢，这是个问题。

我用如下方法来解决这个问题：

首先，我参考同路段的房价，然后对我的房子进行定价。

假如我参考了2套房子，如果只按面积一个条件来看的话有如下2所在售房源

面积	**价格**
50㎡	550万
100㎡	1050万

我们**假设**价格和面积的关系是一个一元一次方程的关系：

价格=y,面积=x,a,和b为**参数**

$$y = ax + b$$

代入我们**已知**的2套房源的面积和价格：

得到了如下方程组：

$$\begin{cases} 550 = a \times 50 + b \\ 1050 = a \times 100 + b \end{cases}$$

求解a和b,得到

$$\begin{cases} a = 10 \\ b = 50 \end{cases}$$

得到ab后，我们可以得知价格和面积的**关系**如下

$$y = 10 \times x + 50$$

最后，80㎡的房子应该卖多少钱，有了答案

$$10 \times 80 + 50 = 850$$

所以，**结果**是，我们的房子可以卖850万。

相信，上面这个过程大家都看懂了，

其实上面这**解方程**的过程就可以看做是一个**机器学习**的过程。

解方程中做出的**假设**，就是机器学习中选择的**方法**；

解方程中的**已知**条件，就是机器学习中的**样本**；

解方程中的**求解**过程，就是机器学习中用样本数据进行的**训练**的过程；

而我们解方程中得到的**关系**式，就是机器学习中建立的**模型**；

在解方程中用关系式计算出的**结果**，就是在机器学习中通过模型做出的**预测**

解方程	**机器学习**
假设	方法
已知	样本
求解	训练
关系	模型
结果	预测

以上，这就是机器学习。

机器学习，还有一些相关的概念，像数据挖掘，人工智能，模式识别，等等等等，有兴趣的同学可以google之，万变不离其宗，说到底，他们都是同一个东西。

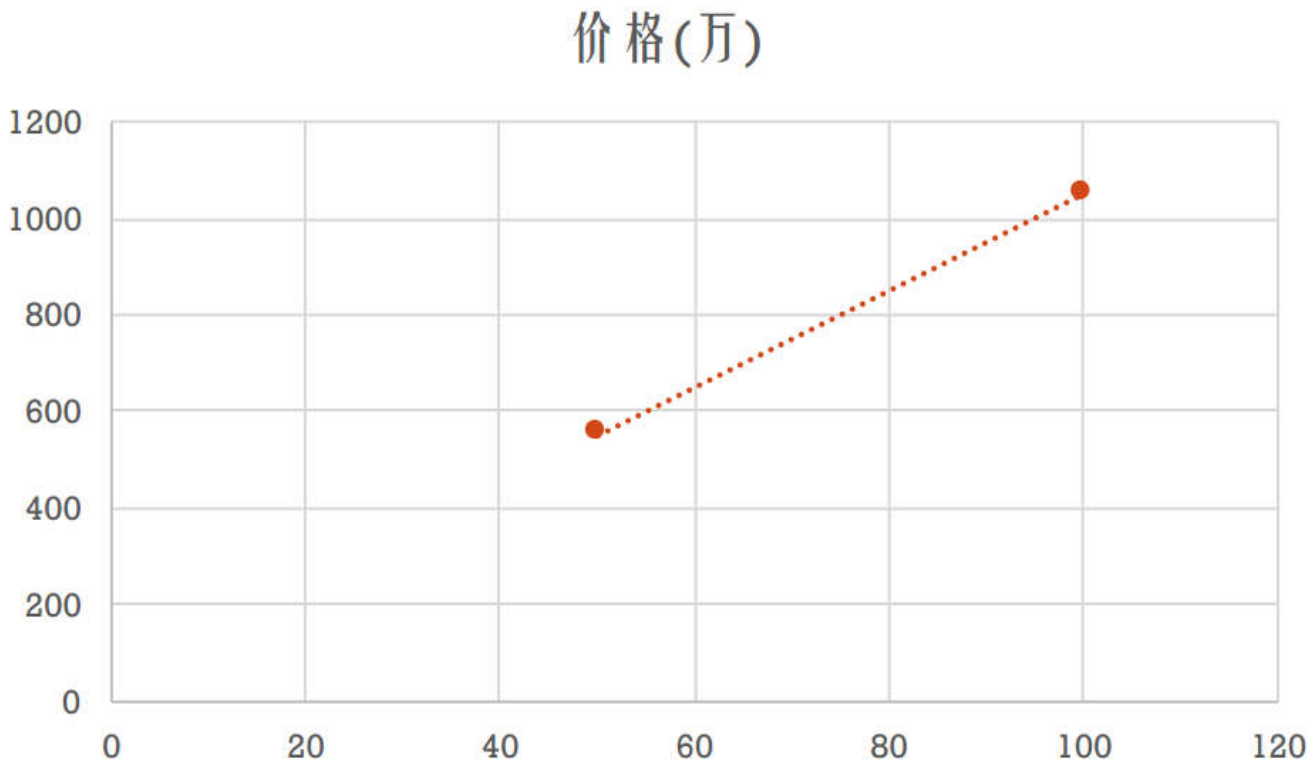
二、样本数量>>参数数量

依然以上文的卖房的例子来看，我们发现在实际卖房子的时候，同地段在售的房源，不会是仅仅只有2套在售的，我们可以参考的已知条件远不止2套房。

当我们的样本只有2个时候，列出的方程和价格趋势图是这样的

面积	**价格**
50㎡	550万
100㎡	1050万

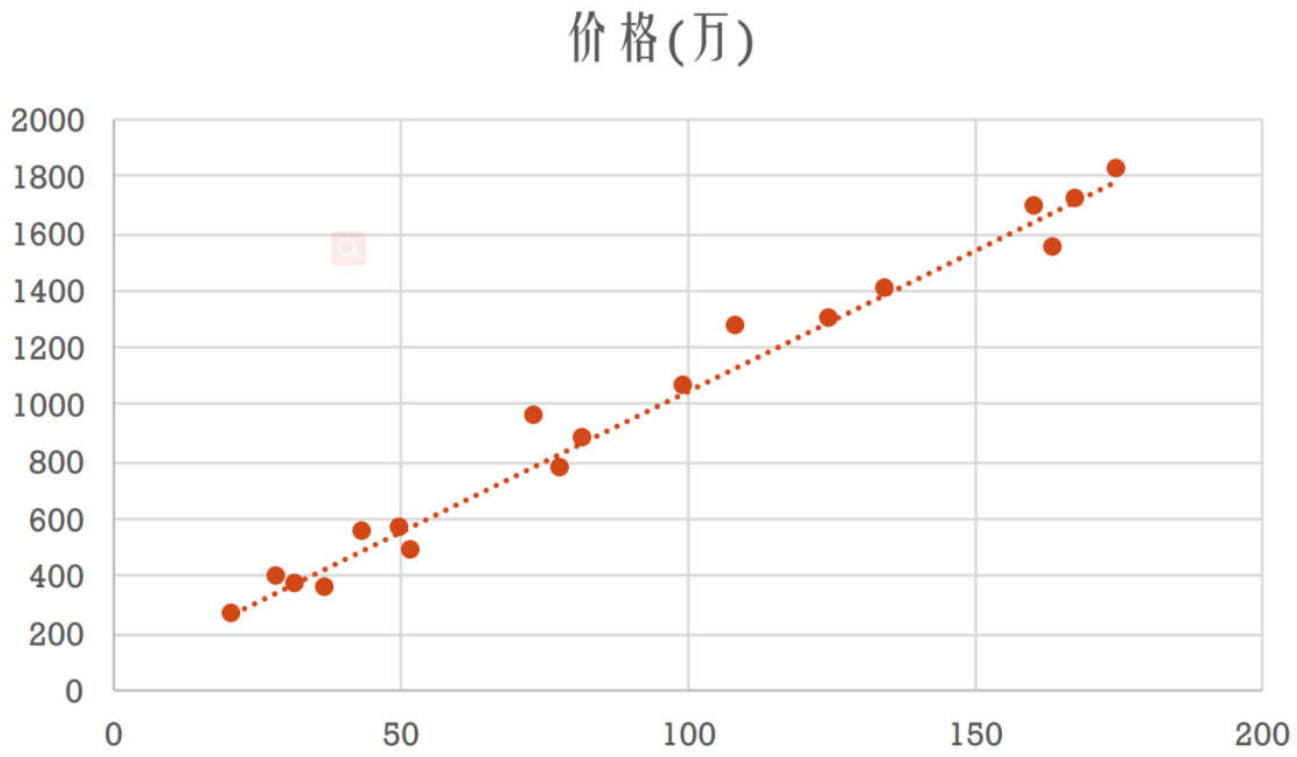
$$y = ax + b$$



当我们的样本很多的时候，我们列出来的方程和价格趋势图是这样的

面积	价格
50	550
100	1050
21	251
52	472
44	543
37	344
168	1712
74	953
…	…

$$\begin{cases} 550 = \mathbf{a} \times 50 + b \\ 1050 = a \times 100 + b \\ 251 = a \times 21 + b \\ 472 = a \times 52 + b \\ 543 = a \times 44 + b \\ 344 = a \times 37 + b \\ 1712 = a \times 168 + b \\ 953 = a \times 74 + b \\ \cdots \end{cases}$$



细心的同学发现，当我们的样本增多，而我们的参数只有2个，也就是实际情况中，**样本数量远远大于参数**的情况下，

我们所列出的方程式没有解的，如何解决这个问题呢?

这就需要我么找到那个最符合实际情况的趋势，换句话说，找到那个**最优解。**

我们们通过2个样本演化出的多个样本形成的方程组，需要求最优解。

$$\begin{cases} y_1 = ax_1 + b \\ y_2 = ax_2 + b \\ y_3 = ax_3 + b \\ y_4 = ax_4 + b \\ y_5 = ax_5 + b \\ y_6 = ax_6 + b \\ y_7 = ax_7 + b \\ y_8 = ax_8 + b \\ y_9 = ax_9 + b \\ \vdots \end{cases}$$

我们要求它的最优解，

通过他我们可以得出另一个方程组，将等号右侧的都移到等号的左边，

$$\begin{cases} \mathrm{y}_1 - ax_1 - b = 0 \\ \mathrm{y}_2 - ax_2 - b = 0 \\ \mathrm{y}_3 - ax_3 - b = 0 \\ \mathrm{y}_4 - ax_4 - b = 0 \\ \mathrm{y}_5 - ax_5 - b = 0 \\ \mathrm{y}_6 - ax_6 - b = 0 \\ \mathrm{y}_7 - ax_7 - b = 0 \\ \mathrm{y}_8 - ax_8 - b = 0 \\ \mathrm{y}_9 - ax_9 - b = 0 \\ \vdots \end{cases}$$

最完美的情况下，等号的左边都等于0，当然这是不可能的；

但是，当我们想求它的最优的时候，用数学的角度来看，当左侧的数据进行某些运算，最接近于0的时候，此解便是我们所求的**最优解**。

事实上，我们有很多种方法可以得到他的最优解，

上面提到的最接近于0，我们可以通过运算，在正数范围内，求它的最小值，**求最小**

换一种说法，左侧的计算结果我们期望等于0，而实际情况下不可能都等于0，会有一个误差，这个误差的值可能是正数，也可能是负数，我们可以取其绝对值相加除以总数表示，也可以将其平方然后相加除以总数（**方差**），等等。

下面列举三种方法：

1、左侧绝对值相加最小

$$
\begin{aligned}
&|y_1 - ax_1 - b| \\
&+|y_2 - ax_2 - b| \\
&+|y_3 - ax_3 - b| \\
&+|y_4 - ax_4 - b| \\
&+|y_5 - ax_5 - b| \\
&+|y_6 - ax_6 - b| \\
&+|y_7 - ax_7 - b| \\
&+|y_8 - ax_8 - b| \\
&+|y_9 - ax_9 - b| \\
&\cdots
\end{aligned}
$$

2、左侧平方和最小

$$
\begin{aligned}
&(y_1 - ax_1 - b)^2 \\
&+(y_2 - ax_2 - b)^2 \\
&+(y_3 - ax_3 - b)^2 \\
&+(y_4 - ax_4 - b)^2 \\
&+(y_5 - ax_5 - b)^2 \\
&+(y_6 - ax_6 - b)^2 \\
&+(y_7 - ax_7 - b)^2 \\
&+(y_8 - ax_8 - b)^2 \\
&+(y_9 - ax_9 - b)^2 \\
&\cdots
\end{aligned}
$$

3、左侧相加的绝对值最小

$$
\begin{aligned}
&(y_1 - ax_1 - b)^2 \\
&+(y_2 - ax_2 - b)^2 \\
&+(y_3 - ax_3 - b)^2 \\
&+(y_4 - ax_4 - b)^2 \\
&+(y_5 - ax_5 - b)^2 \\
&+(y_6 - ax_6 - b)^2 \\
&+(y_7 - ax_7 - b)^2 \\
&+(y_8 - ax_8 - b)^2 \\
&+(y_9 - ax_9 - b)^2 \\
&\cdots
\end{aligned}
$$

实际情况中，在数学上，方法二，最容易得出最优解，所以，我们采用方法二求解。

这个方法在数学上叫做**最小二乘法**

最小二乘法，是一种数学优化技术，它通过最小化误差的平方和寻找数据的最佳函数匹配。

利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。

$$f(a,b) = (y_1 - ax_1 - b)^2 + (y_2 - ax_2 - b)^2 + (y_3 - ax_3 - b)^2 + (y_4 - ax_4 - b)^2 + (y_5 - ax_5 - b)^2 \cdots$$

$$f(a,b) = (550 - 50a - b)^2 + (1050 - 100a - b)^2 + (251 - 21a - b)^2 + (472 - 52a - b)^2 + (543 - 44a - b)^2 \cdots$$

求a,b，使$f(a,b)$最小

$$\begin{cases} \dfrac{\partial f(a,b)}{\partial a} = 0 \\ \dfrac{\partial f(a,b)}{\partial b} = 0 \end{cases}$$

$$\begin{cases} 0 = 2(550 - 50a - b)\times(-50) + 2(1050 - 100a - b)\times(-100) + 2(251 - 21a - b)\times(-21) \cdots \\ 0 = 2(550 - 50a - b)\times(-1) + 2(1050 - 100a - b)\times(-1) + 2(251 - 21a - b)\times(-1) \cdots\cdots \end{cases}$$

$$\begin{cases} -2037382 + 196496a + 1636b = 0 \\ 17235 - 1636a - 18b = 0 \end{cases} \qquad \begin{cases} a = \dfrac{8476416}{860432} \\ b = \dfrac{53451608}{860432} \end{cases} \qquad \begin{cases} a = 9.8153 \\ b = 62.122 \end{cases}$$

通过计算，我们求得了参数a和b。

这里我们求得的关系式，或者说我们建立的**模型**可表示为：

$$y = 9.8153x + 62.122$$

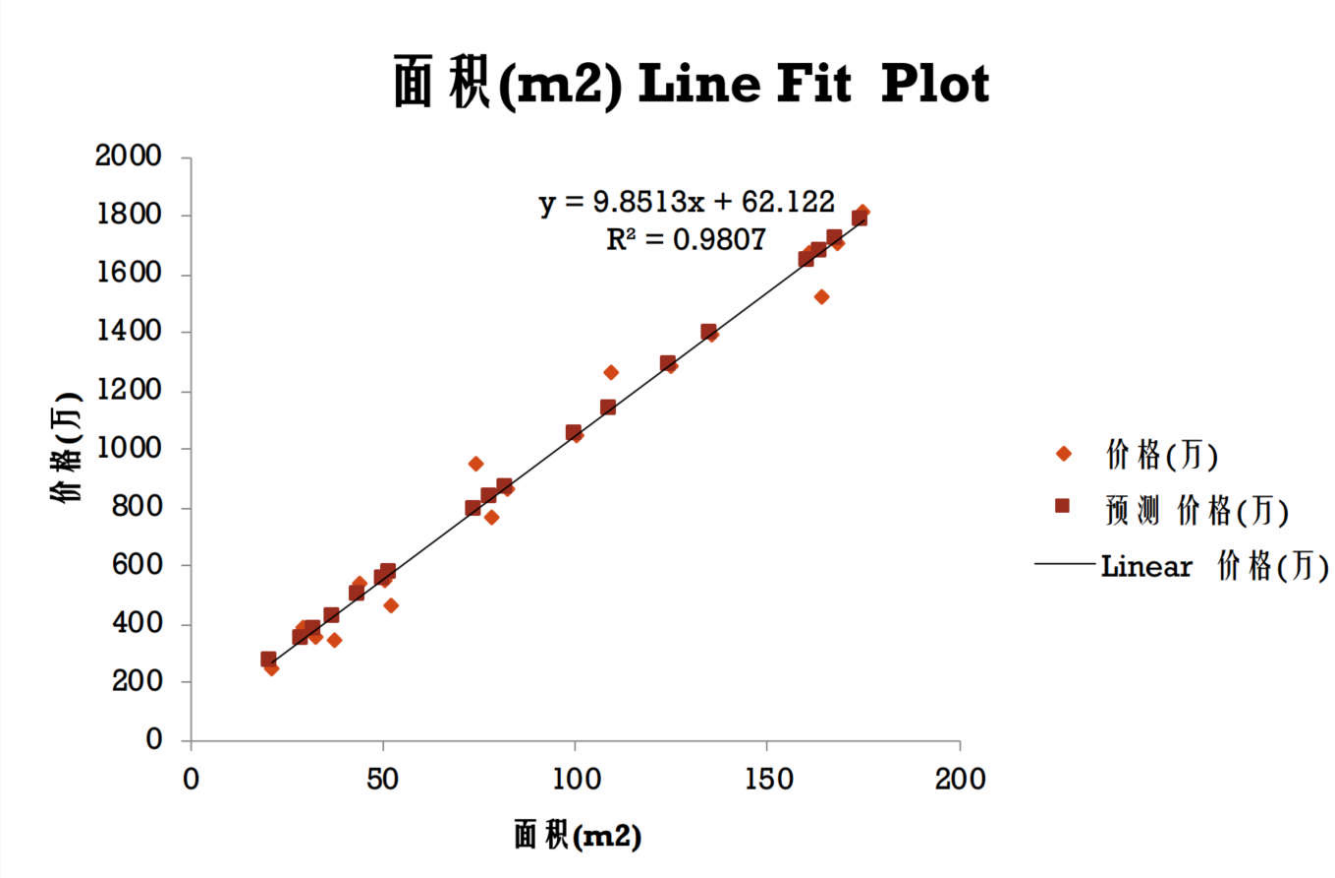
那我们80㎡的房子售价的预测结果通过计算可以知，

$$9.8153 \times 80 + 62.122 = 847.346$$

预测售价为847.346万元。

这里我们用到的方法，在统计学上来说，叫做回归分析，这里只有房间面积一个单一变量，求价格。

因此，适用于**一元线性回归**



SUMMARY OUTPUT

回归统计	
Multiple R	0.990314396
R Square	0.980722603
Adjusted R Square	0.979517765
标准误差	75.49337613
观测值	18

方差分析

	df	*SS*	*MS*	*F*	*Significance F*
回归分析	1	4639118.503	4639118.503	813.9875655	3.77781E-15
残差	16	91187.99744	5699.24984		
总计	17	4730306.5			

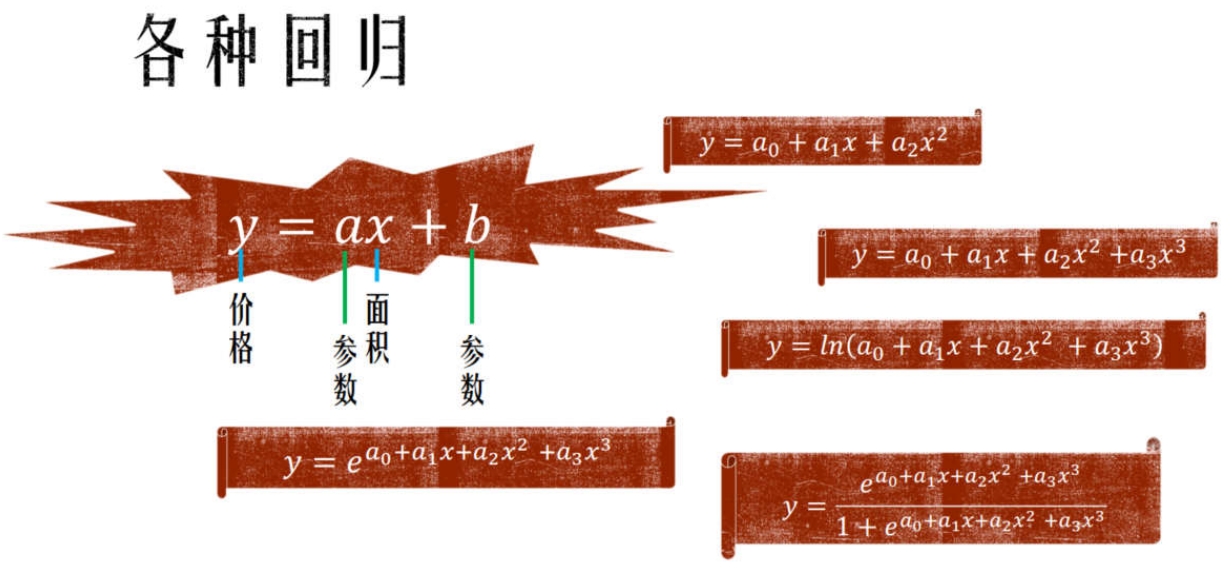
	Coefficients	*标准误差*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*
Intercept	62.12182718	36.07675175	1.721935156	0.104353302	-14.35747005	138.6011244
面积(m2)	9.85134909	0.345292244	28.53046732	3.77781E-15	9.119362231	10.58333595

三、多变量

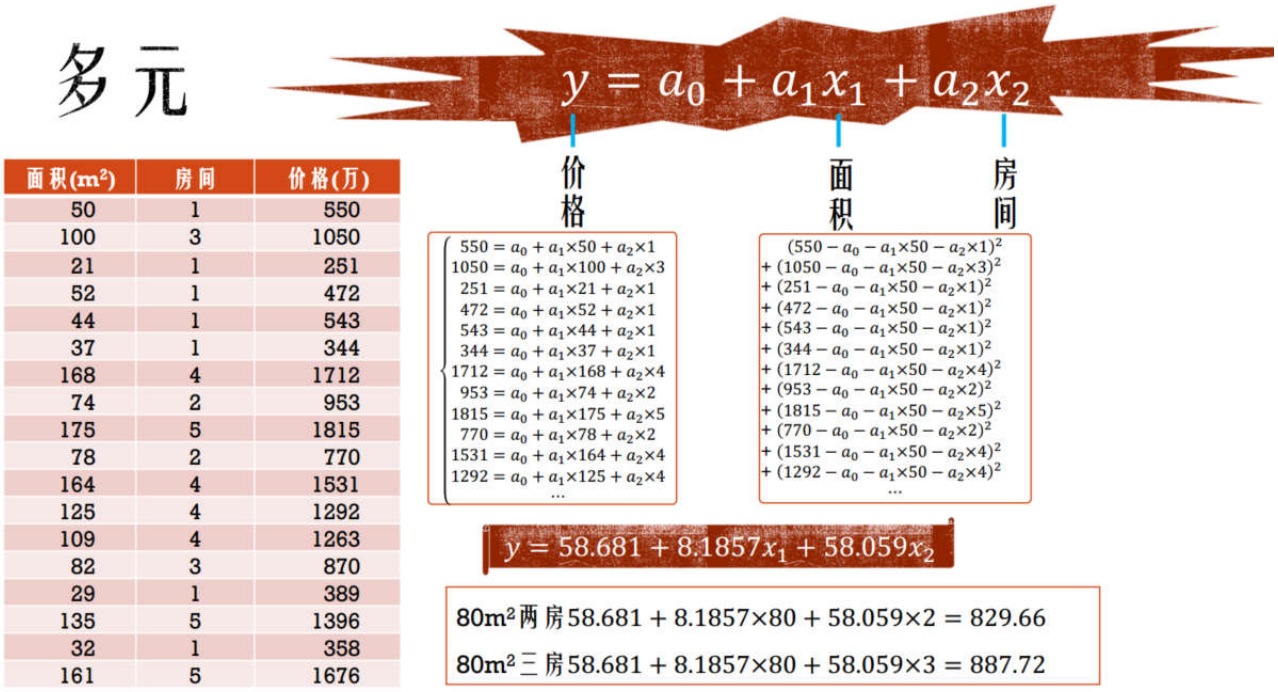
上文举到的例子，我们只考虑了房屋面积这一个因素，即只有一个自变量，在实际情况中，变量的个数往往不止一个。

这样就需要采不同的方法，来训练样本，建立模型。会用到各种不同的回归分析方法

例如：



还是以卖房为例，我们如果将面积，和房间数都作为考虑因素，我们获取到的样本，就会有面积，房间，价格这三组数据，而我们采用的方法将是**复回归分析（多变量回归分析）**



面积(m²)	房间	价格(万)
50	1	550
100	3	1050
21	1	251
52	1	472
44	1	543
37	1	344
168	4	1712
74	2	953
175	5	1815
78	2	770
164	4	1531
125	4	1292
109	4	1263
82	3	870
29	1	389
135	5	1396
32	1	358
161	5	1676

四、损失函数

损失函数（loss function）是用来估量你模型的预测值f(x)与真实值Y的不一致程度，它是一个非负实值函数,通常使用L(Y, f(x))来表示，损失函数越小，模型的鲁棒性就越好。

我们希望我们预测的公式与实际值差值越小越好，所以就定义了一种衡量模型好坏的方式，即损失函数（用来表现预测与实际数据的差距程度）。

于是乎我们就会想到这个方程的损失函数可以用绝对损失函数表示：

公式Y-实际Y的绝对值，

$$L(y, f(x)) = |y - f(x)|$$

为后续数学计算方便，我们通常使用平方损失函数代替绝对损失函数：

公式Y-实际Y的平方，

平方损失函数

$$L(y, f(x)) = (y - f(x))^2$$

当样本个数为n时，此时的损失函数变为：

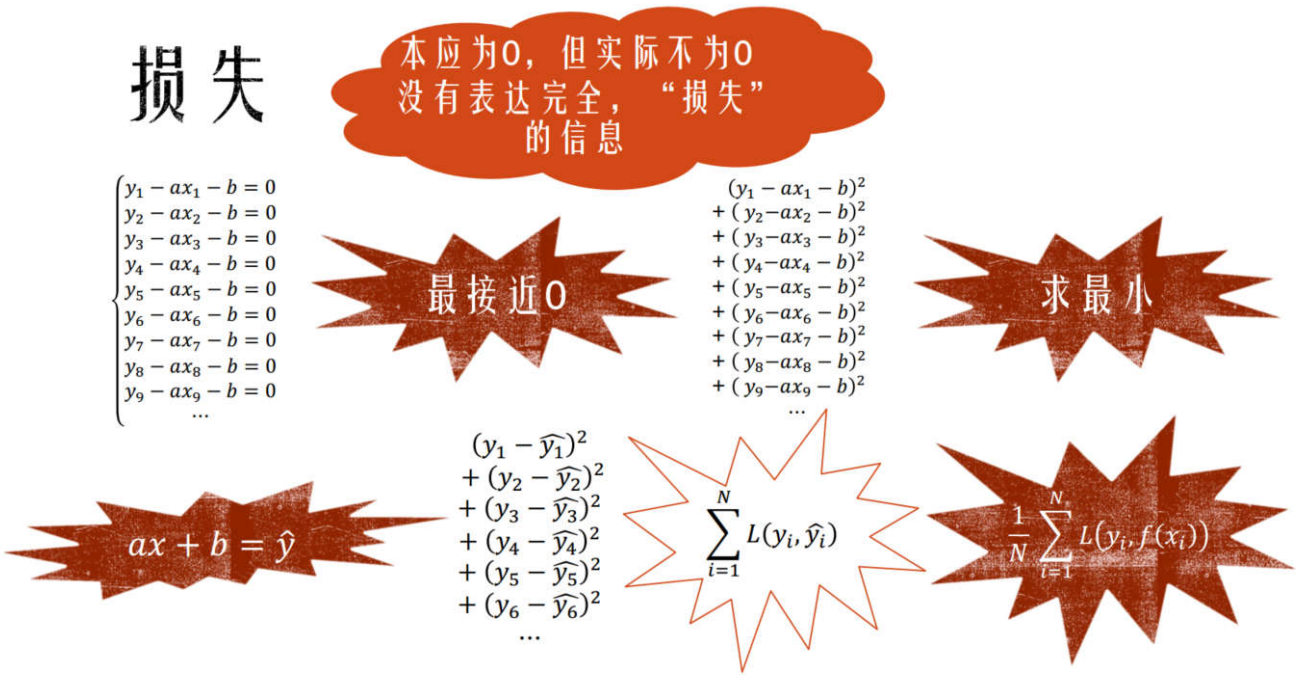
$$L(y, f(x)) = \sum_{i=1}^{n} (y - f(x))^2$$

y-f(x)表示的是残差，整个式子表示的是残差的平方和，而我们的目的就是最小化这个目标函数值。

而在实际应用中，通常会使用均方差（MSE）作为一项衡量指标，公式如下：

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y - f(x))^2$$

在我们的例子中，参考如下：



这里例子中对应前面提到的损失函数有如下关系

$$ax + b = \hat{y} = f(x)$$

$L(y_i, \hat{y_i})$即$L(y_i, f(x_i))$

其他损失函数：

统计学习中常用的损失函数有以下几种：

(1) 0-1损失函数(0-1 lossfunction):

(2) 平方损失函数(quadraticloss function)

(3) 绝对损失函数(absoluteloss function)

(4) 对数损失函数(logarithmicloss function)或对数似然损失函数(log-likelihood loss function)

损失函数越小，模型就越好。

(5) 指数损失函数

(6) 合叶损失函数-SVM

各算法的损失函数整理

线性回归

$$L(Y, f(X)) = \sum_{i=1}^{n}(Y - f(X))^2$$

逻辑回归

$$J(\theta) = -\frac{1}{N}\sum_{i=1}^{N} y^{(i)} \log\left(h_\theta\left(x^{(i)}\right)\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_\theta\left(x^{(i)}\right)\right)$$

决策树

$$C_\alpha(T) = \sum_{t=1}^{|T|} N_t H_t(T) + \alpha|T|$$

SVM

$$\sum_{i}^{N}[1 - y_i(w \cdot x_i + b)]_+ + \lambda||w||^2$$

AdaBoost

$$L(y, f(x)) = \frac{1}{N}\sum_{i=1}^{n} exp[-y_i f(x_i)]$$

大家想深入了解，可以参考相关资料，或者google之。

参考资料如下：

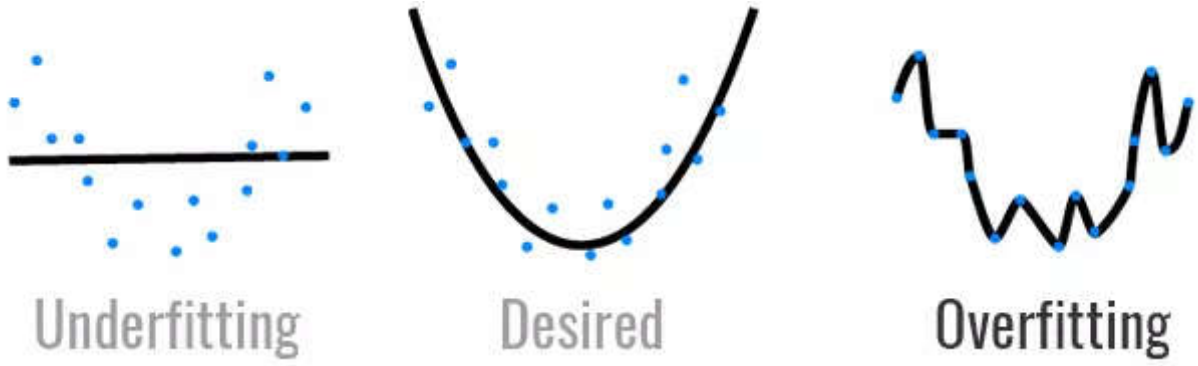
《统计学》，《统计学方法》，《The Elements of Statistical Learning》

五、过拟合

过拟合（overfitting）

在统计学和机器学习中，overfitting一般在描述统计学模型随机误差或噪音时用到。它通常发生在模型过于复杂的情况下，如参数过多等。overfitting会使得模型的预测性能变弱，并且增加数据的波动性。

过拟合是指模型为了得到一致假设而使假设变得过于严格，也就是说模型对训练数据的学习有点过头。模型并没有学习数据的整体分布，而是学习了每个数据点的预期输出。



这就好比你在做数学题的时候，你只记准了某些特定问题的答案是什么，但不知道解题的公式。这就造成模型无法泛化。

如何防止过拟合

1 获取更多数据

你的模型可以存储很多很多的信息，这意味着你输入模型的训练数据越多，模型就越不可能发生过拟合。原因是随着你添加更多数据，模型会无法过拟合所有的数据样本，被迫产生泛化以取得进步。 收集更多的数据样本应该是所有数据科学任务的第一步，数据越多会让模型的准确率更高，这样也就能降低发生过拟合的概率。

2 数据增强&噪声数据

收集更多的数据会比较耗时耗力。如果没有时间和精力做这个，应该尝试让你的数据看起来更多元化一些。利用数据增强的方法可以做到这一点，这样模型每次处理样本的时候，都会以不同于前一次的角度看待样本。这就提高了模型从每个样本中学习参数的难度。

另一个好方法是增加噪声数据：

- 对于输入：和数据增强的目的相同，但是也会让模型对可能遇到的自然扰动产生鲁棒性。
- 对于输出：同样会让训练更加多元化。

注意：在这两种情况中，你需要确保噪声数据的量级不能过大。否则最后你获取的输入信息都是来自噪声数据，或者导致模型的输出不正确。这两种情况也都会对模型的训练过程带来一定干扰。

3 简化模型

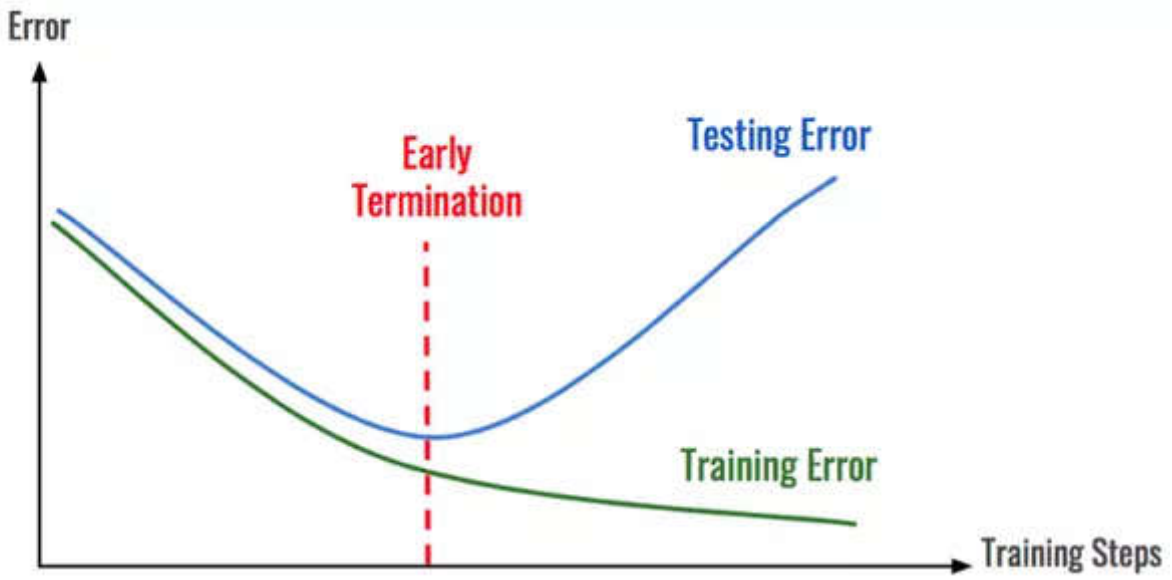
即时你现在手中获取了所有需要的数据，如果你的模型仍然过拟合训练数据集，可能是因为模型过于强大。那么你可以试着降低模型的复杂程度。

如前所述，模型只能过拟合部分数据。通过不断降低模型的复杂度（比如随机森林中的估计量，神经网络中的参数），最终达到一个平衡状态：模型足够简单到不产生过拟合，又足够复杂到能从数据中学习。这样操作时一个比较方便的方法是根据模型的复杂程度查看模型在所有数据集上的误差。

简化模型的另一个好处是能让模型更轻便，训练速度更快，运行速度也会更快。

4 提前终止

大部分情况下，模型会首先学习数据的正确分布，然后在某个时间点上开始对数据过拟合。通过识别模型是从哪些地方开始发生转变的，那么就可以在过拟合出现之前停止模型的学习过程。和前面一样，通过查看随着时间推移的训练错误，就可以做到这一点。



(当测试错误开始增加时，就该停止训练了)

六、正则项

监督机器学习就是规则化**参数**的同时最小化**误差**。

有监督学习的样本都是带有标签的样本，用 y 来表示样本的标签，我们通过算法来提取样本特征并对其进行分类或回归，得到结果 $y_1 = W^T x$, 这里x为样本、W 即是参数，此时有目标函数 $z = y - y_1$,我们希望对于相同的样本，其结果输出与其标签一样，于是我们通过优化算法使得z 尽可能的小，即$min(z) = min(y - W^T x)$, 优化算法即是更新参数W 的值使得分类输出更加接近于标签y ,但是由于种种原因比如样本量过少的问题会导致过拟合

过拟合导致参数向量WW变大， 我们可以给目标函数zz 加上一个正则化项，指约束模型的学习以减少过拟合，常见的正则化项有L0范数、L1范数以及L2范数，下面简单的介绍一下范数的概念。

L0范数:

其表示向量中非零元素的个数。如果我们使用L0来规则化参数向量W，就是希望W的元素大部分都为零。L0范数的这个属性，使其非常适用于机器学习中的稀疏编码。在特征选择中，通过最小化L0范数来寻找最少最优的稀疏特征项。但是，L0范数的最小化问题是NP难问题。L1范数是L0范数的最优凸近似，它比L0范数要更容易求解。因此，L0优化过程将会被转换为更高维的范数（例如L1范数）问题。

$$L0\text{范数：}\ ||X||_0 = \sum_{i=0}^{n} X_i^0$$

L1范数

L1范数是向量中各个元素绝对值之和，也被称作“Lasso regularization”（稀疏规则算子）。

$$||x||_1 = \sum_{i=1}^{N} |x_i|$$

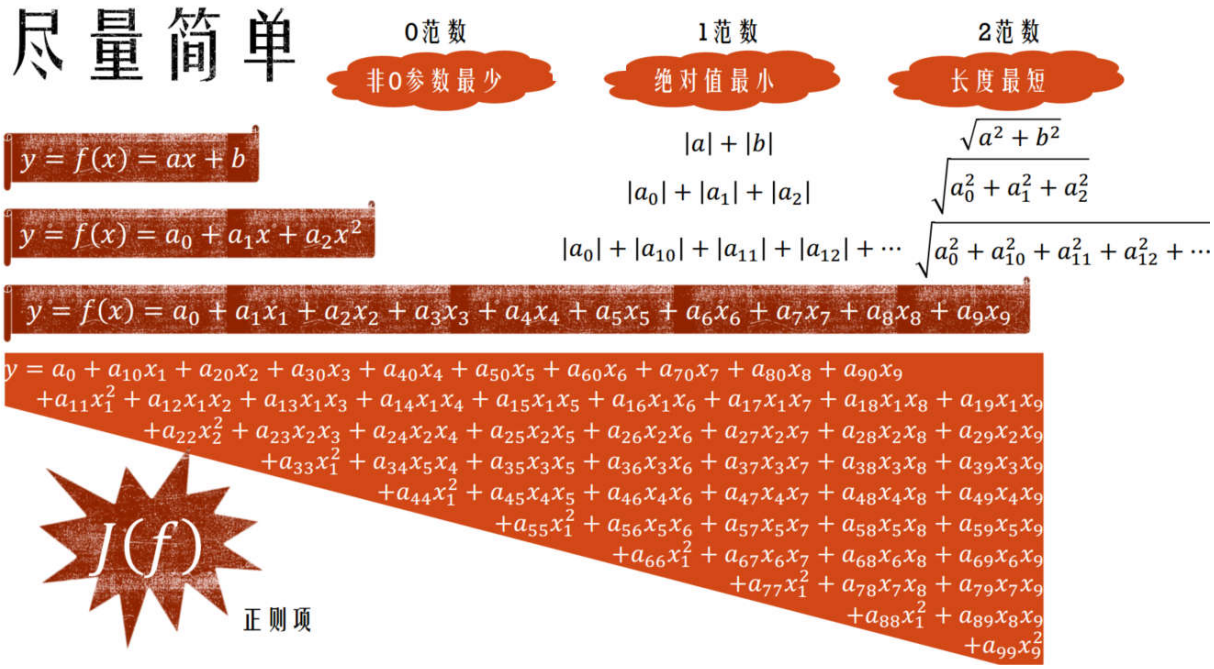
L2范数

Euclid范数（欧几里得范数，常用计算向量长度），即向量元素绝对值的平方和再开方。

我们让L2的规则化项最小，可以使W中的每个元素都很小，但不像L1范数那样使元素等于0，而是接近于零。越小的参数说明模型越简单，越简单的模型越不容易产生过拟合的现象。即通过L2范数可以防止过拟合，提升模型的泛化能力。

$$||\mathbf{x}||_2 = \sqrt{\sum_{i=1}^{N} x_i^2}$$

前文提到的例子来做参考，0范数，1范数，2范数如下：

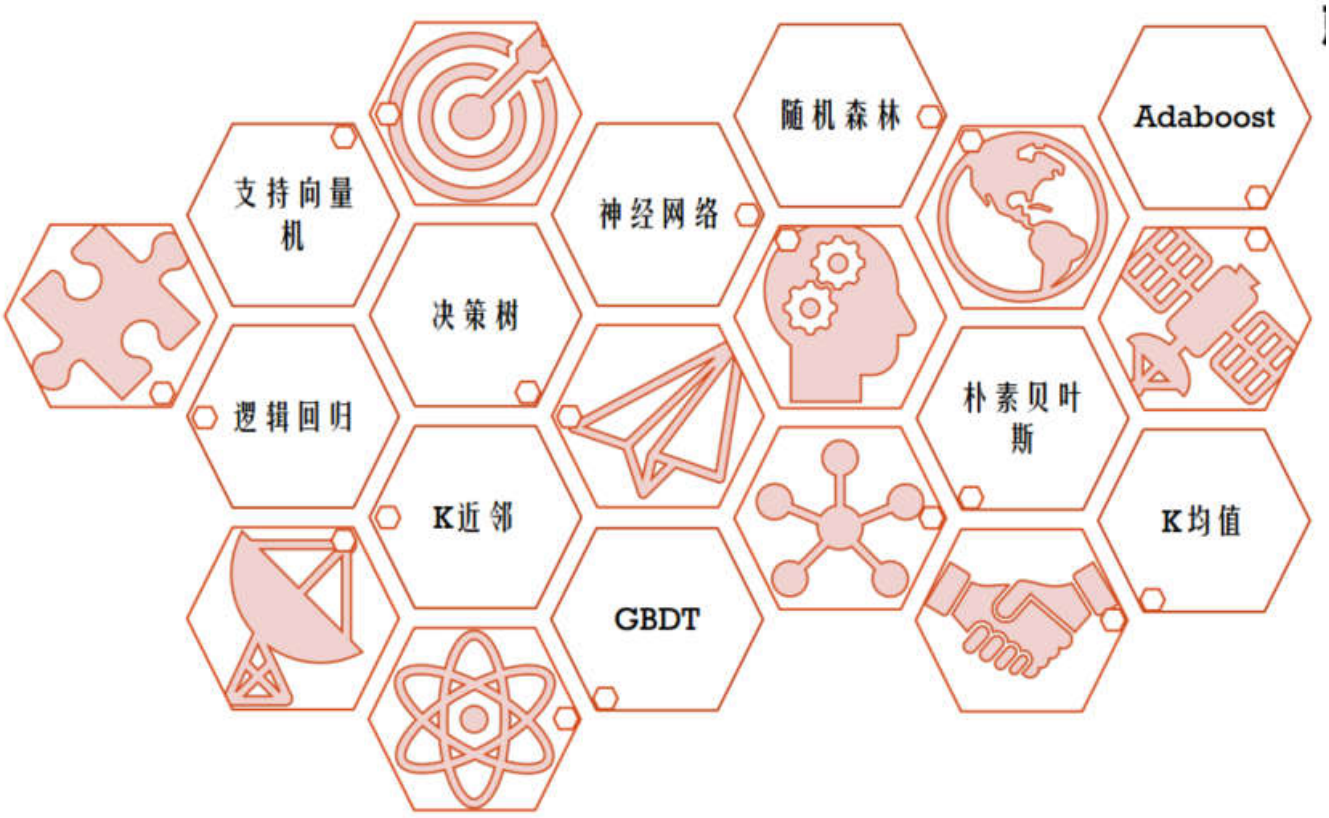


七、目标函数

上文中提到的正则项 J(f)，结合我们的损失函数，就得出目标函数,这里引入一个正则项系数，λ随机数。目标函数表示如下：

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

八、其他



大家想深入了解，可以参考相关资料，或者google之。

参考资料如下：

《统计学》，《统计学方法》，《The Elements of Statistical Learning》