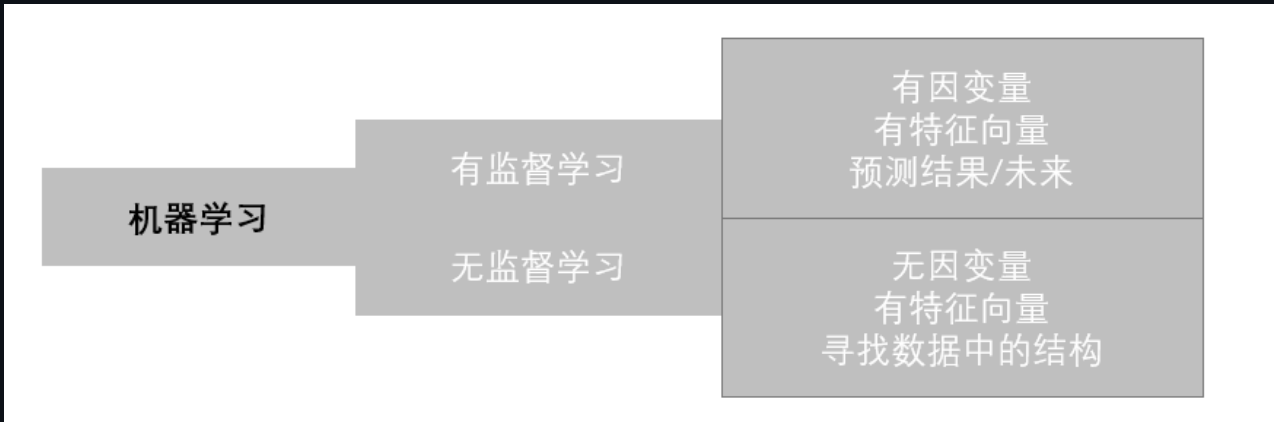
什么是机器学习

机器学习是一门致力于研究如何通过计算的手段，利用经验来改善系统自身性能的学科。在计算机系统中，「经验」通常以「数据」形式存储机器学习的一个重要的目标就是利用数学模型来理解数据，发现数据中的规律，用作数据的分析和预测。在计算机中，数据通常由一组向量组成，这组向量中的每个向量都是一个样本，我们用x_i来表示一个样本，其中$i = 1, 2, 3, \ldots, N$，共N个样本，每个样本$x_i = (x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)$共p+1个维度，前p个维度的每个维度我们称为一个特征，最后一个维度y_i我们称为因变量(响应变量)。特征用来描述影响因变量的因素，如：我们要探寻身高是否会影响体重的关系的时候，身高就是一个特征，体重就是一个因变量。通常在一个数据表dataframe里面，一行表示一个样本x_i，一列表示一个特征。
根据数据是否有因变量，机器学习的任务可分为：**有监督学习**和**无监督学习**。

- 有监督学习：给定某些特征去估计因变量，即因变量存在的时候，我们称这个机器学习任务为有监督学习。如：我们使用房间面积，房屋所在地区，环境等级等因素去预测某个地区的房价。
- 无监督学习：给定某些特征但不给定因变量，建模的目的是学习数据本身的结构和关系。如：我们给定某电商用户的基本信息和消费记录，通过观察数据中的哪些类型的用户彼此间的行为和属性类似，形成一个客群。注意，我们本身并不知道哪个用户属于哪个客群，即没有给定因变量。



根据因变量的是否连续，有监督学习又分为**回归**和**分类**：

- 回归：因变量是连续型变量，如：房价，体重等。
- 分类：因变量是离散型变量，如：是否患癌症，西瓜是好瓜还是坏瓜等。

为了更好地叙述后面的内容，我们对数据的形式作出如下约定：

第i个样本：$x_i = (x_{i1}, x_{i2}, \ldots, x_{ip}, y_i)^T, i = 1, 2, \ldots, N$

因变量$y = (y_1, y_2, \ldots, y_N)^T$

第k个特征:$x^{(k)} = (x_{1k}, x_{2k}, \ldots, x_{Nk})^T$

特征矩阵$X = (x_1, x_2, \ldots, x_N)^T$

在机器学习中，我们通常会使用 `sklearn` 工具库来探索机器学习项目。

```
# 引入相关科学计算包
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use("ggplot")
import seaborn as sns
```

回归

寻找数据中规律的问题是一个基本的问题，有着很长的很成功的历史。机器学习领域关注的是利用计算机算法自动发现数据中的规律，以及使用这些规律采取将数据分类等行动。我们以波士顿房价为例：

```
from sklearn import datasets
boston = datasets.load_boston()     # 返回一个类似于字典的类
X = boston.data
y = boston.target
features = boston.feature_names
boston_data = pd.DataFrame(X,columns=features)
boston_data["Price"] = y
```

sklearn中所有内置数据集都封装在datasets对象内：

返回的对象有：

- data:特征X的矩阵(ndarray)
- target:因变量的向量(ndarray)
- feature_names:特征名称(ndarray)

让我们来看一看数据

```
1 boston_data.head()
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	Price
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2

```
1 boston_data
```

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	Price
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33	36.2
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
501	0.06263	0.0	11.93	0.0	0.573	6.593	69.1	2.4786	1.0	273.0	21.0	391.99	9.67	22.4
502	0.04527	0.0	11.93	0.0	0.573	6.120	76.7	2.2875	1.0	273.0	21.0	396.90	9.08	20.6
503	0.06076	0.0	11.93	0.0	0.573	6.976	91.0	2.1675	1.0	273.0	21.0	396.90	5.64	23.9
504	0.10959	0.0	11.93	0.0	0.573	6.794	89.3	2.3889	1.0	273.0	21.0	393.45	6.48	22.0
505	0.04741	0.0	11.93	0.0	0.573	6.030	80.8	2.5050	1.0	273.0	21.0	396.90	7.88	11.9

506 rows × 14 columns

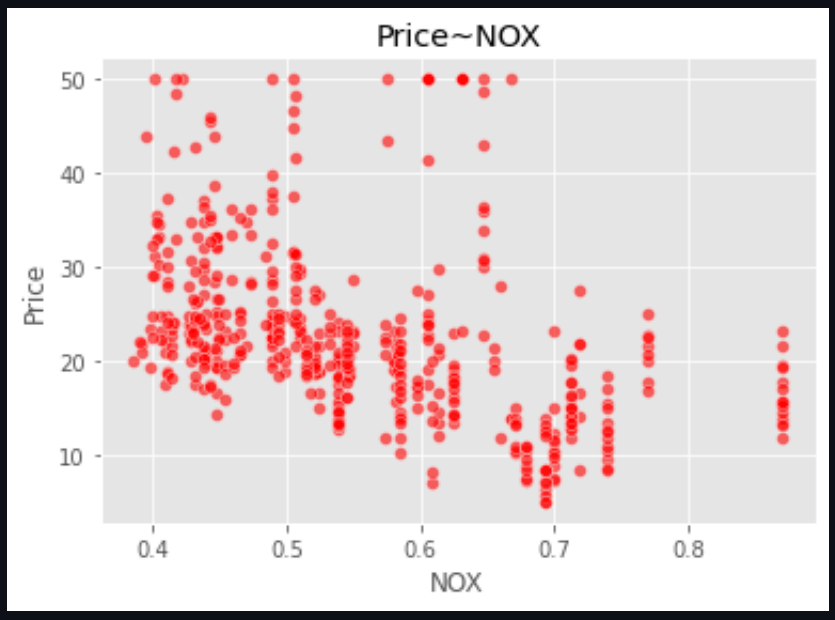
各个特征的相关解释：

- CRIM：各城镇的人均犯罪率
- ZN：规划地段超过25,000平方英尺的住宅用地比例
- INDUS：城镇非零售商业用地比例

- CHAS：是否在查尔斯河边(=1是)
- NOX：一氧化氮浓度(/千万分之一)
- RM：每个住宅的平均房间数
- AGE：1940年以前建造的自住房屋的比例
- DIS：到波士顿五个就业中心的加权距离
- RAD：放射状公路的可达性指数
- TAX：全部价值的房产税率(每1万美元)
- PTRATIO：按城镇分配的学生与教师比例
- B：1000(Bk - 0.63)^2其中Bk是每个城镇的黑人比例
- LSTAT：较低地位人口
- Price：房价

既然是回归任务，那让我们看看给定自变量（feature）的条件下因变量(target)的变化情况

```
1 sns.scatterplot(boston_data['NOX'],boston_data['Price'],co
  lor="r",alpha=0.6)
2 plt.title("Price~NOX")
3 plt.show()
```



分类

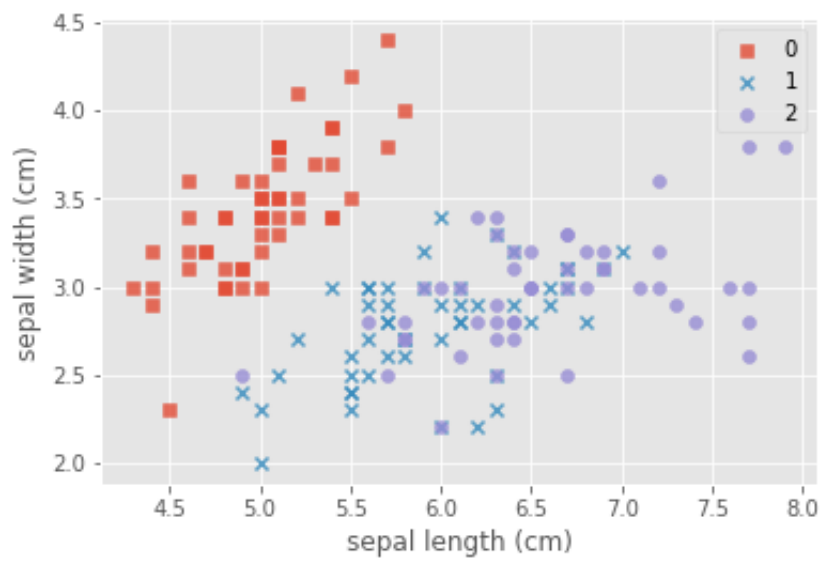
给每个输入向量分配到有限数量离散标签中的一个，被称为分类(classification)问题。如果要求的输出由一个或者多个连续变量组成，那么这个任务被称为回归(regression)。

我们可以看一个鸢尾花类别分类的例子，来自大名鼎鼎的iris数据集：

```
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
y = iris.target
features = iris.feature_names
iris_data = pd.DataFrame(X,columns=features)
iris_data['target'] = y
iris_data.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	target
0	5.1	3.5	1.4	0.2	0
1	4.9	3.0	1.4	0.2	0
2	4.7	3.2	1.3	0.2	0
3	4.6	3.1	1.5	0.2	0
4	5.0	3.6	1.4	0.2	0

```
# 可视化特征
marker = ['s','x','o']
for index,c in enumerate(np.unique(y)):
    plt.scatter(x=iris_data.loc[y==c,"sepal length (cm)"],y=iris_data.loc[y==c,"sepal width (cm)"],alpha=0.8,label=c,marker=marker[c])
plt.xlabel("sepal length (cm)")
plt.ylabel("sepal width (cm)")
plt.legend()
plt.show()
```



我们可以看到：每种不同的颜色和点的样式为一种类型的鸢尾花，数据集有三种不同类型的鸢尾花。因此因变量是一个类别变量，因此通过特征预测鸢尾花类别的问题是一个分类问题。

各个特征的相关解释：

- sepal length (cm)：花萼长度(厘米)
- sepal width (cm)：花萼宽度(厘米)
- petal length (cm)：花瓣长度(厘米)
- petal width (cm)：花瓣宽度(厘米)

无监督学习

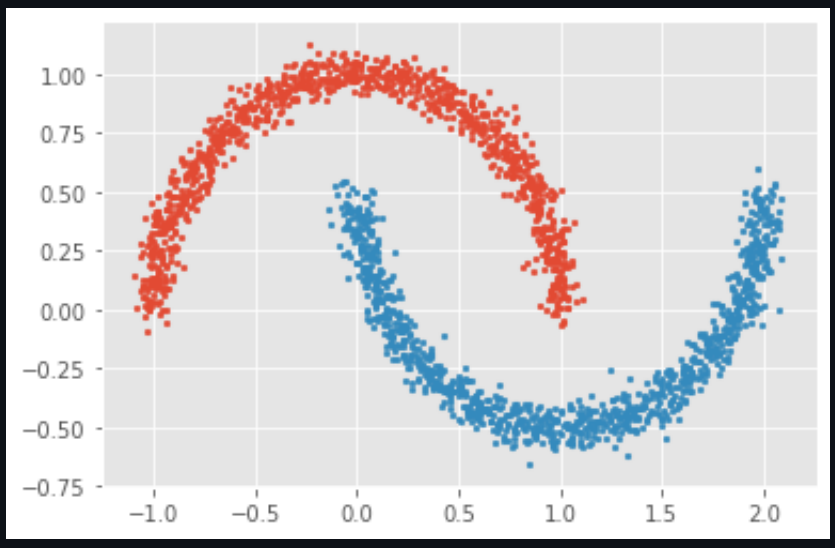
在其他的机器学习问题中，训练数据由一组输入向量x组成，没有任何对应的目标值。 在这样的`无监督学习(unsupervised learning)`问题中，目标可能是发现数据中相似样本的 分组，这被称为`聚类(clustering)`，或者决定输入空间中数据的分布，这被称为`密度估计 (density estimation)`，或者把数据从高维空间投影到二维或者三维空间，为了`数据可视化 (visualization)`。

我们可以使用sklearn生成符合自身需求的数据集，下面我们用其中几个函数例子来生成无因变量的数据集：

https://scikit-learn.org/stable/modules/classes.html?highlight=datasets#module-sklearn.datasets

Samples generator	
datasets.make_biclusters(shape, n_clusters, *)	Generate an array with constant block diagonal structure for biclustering.
datasets.make_blobs([n_samples, n_features, ...])	Generate isotropic Gaussian blobs for clustering.
datasets.make_checkerboard(shape, n_clusters, *)	Generate an array with block checkerboard structure for biclustering.
datasets.make_circles([n_samples, shuffle, ...])	Make a large circle containing a smaller circle in 2d.
datasets.make_classification([n_samples, ...])	Generate a random n-class classification problem.
datasets.make_friedman1([n_samples, ...])	Generate the “Friedman #1” regression problem
datasets.make_friedman2([n_samples, noise, ...])	Generate the “Friedman #2” regression problem
datasets.make_friedman3([n_samples, noise, ...])	Generate the “Friedman #3” regression problem
datasets.make_gaussian_quantiles(*[, mean, ...])	Generate isotropic Gaussian and label samples by quantile
datasets.make_hastie_10_2([n_samples, ...])	Generates data for binary classification used in Hastie et al.
datasets.make_low_rank_matrix([n_samples, ...])	Generate a mostly low rank matrix with bell-shaped singular values
datasets.make_moons([n_samples, shuffle, ...])	Make two interleaving half circles
datasets.make_multilabel_classification([...])	Generate a random multilabel classification problem.
datasets.make_regression([n_samples, ...])	Generate a random regression problem.
datasets.make_s_curve([n_samples, noise, ...])	Generate an S curve dataset.
datasets.make_sparse_coded_signal(n_samples, ...)	Generate a signal as a sparse combination of dictionary elements.
datasets.make_sparse_spd_matrix([dim, ...])	Generate a sparse symmetric definite positive matrix.
datasets.make_sparse_uncorrelated([...])	Generate a random regression problem with sparse uncorrelated design
datasets.make_spd_matrix(n_dim, *[, ...])	Generate a random symmetric, positive-definite matrix.
datasets.make_swiss_roll([n_samples, noise, ...])	Generate a swiss roll dataset.

```
# 生成月牙型非凸集
from sklearn import datasets
x, y = datasets.make_moons(n_samples=2000, shuffle=True,
                           noise=0.05, random_state=None)
for index,c in enumerate(np.unique(y)):
    plt.scatter(x[y==c,0],x[y==c,1],s=7)
plt.show()
```




```python
# 生成符合正态分布的聚类数据
from sklearn import datasets
x, y = datasets.make_blobs(n_samples=5000, n_features=2, centers=3)
for index,c in enumerate(np.unique(y)):
    plt.scatter(x[y==c, 0], x[y==c, 1],s=7)
plt.show()
```