

# 法律声明

---

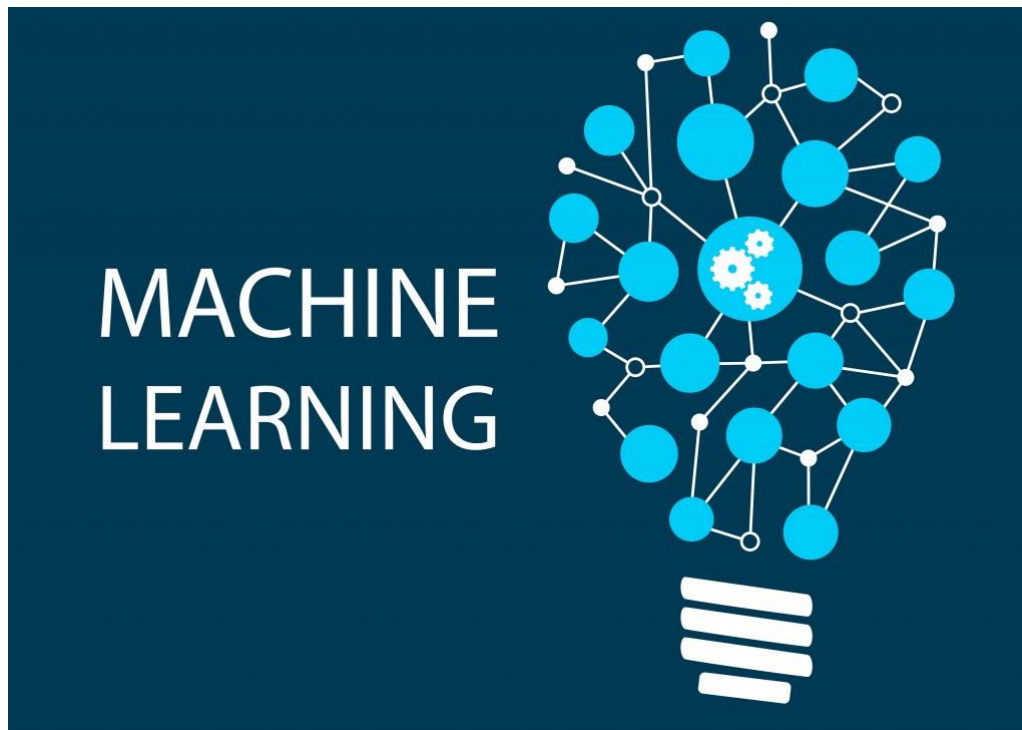
- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。



关注 小象学院

# 第三讲

---



## 机器学习（1）

--Robin

# 目录

---

- 机器学习基本概念与流程
- k-近邻算法（kNN）
- 线性回归（Linear Regression）
- 逻辑回归（Logistic Regression）及Softmax回归
- 正则化
- 实战案例3-1：贷款违约预测 (1)

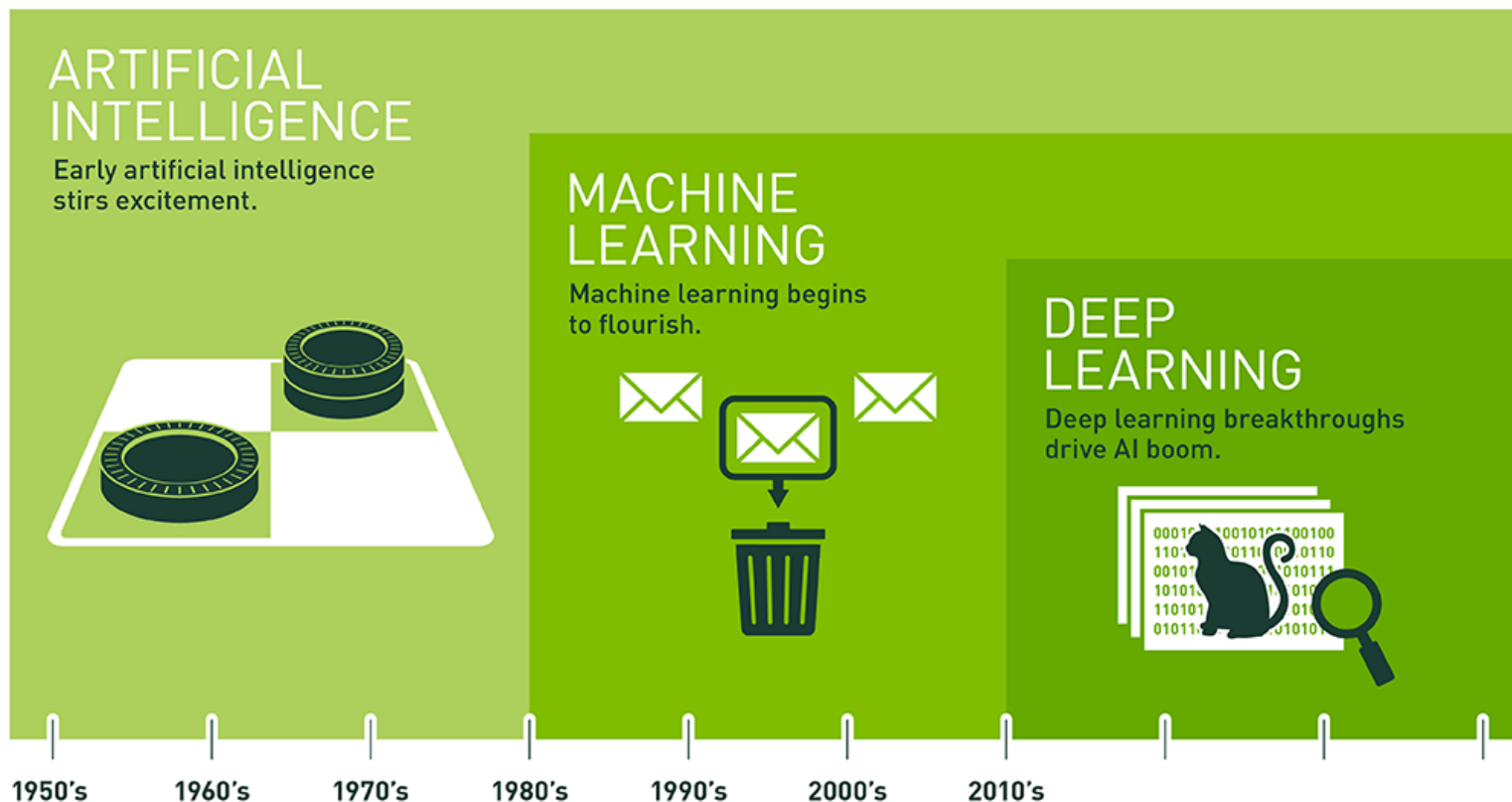
# 目录

---

- 机器学习基本概念与流程
- k-近邻算法 (kNN)
- 线性回归 (Linear Regression)
- 逻辑回归 (Logistic Regression) 及Softmax回归
- 正则化
- 实战案例3-1: 贷款违约预测 (1)

# 机器学习基本概念与流程

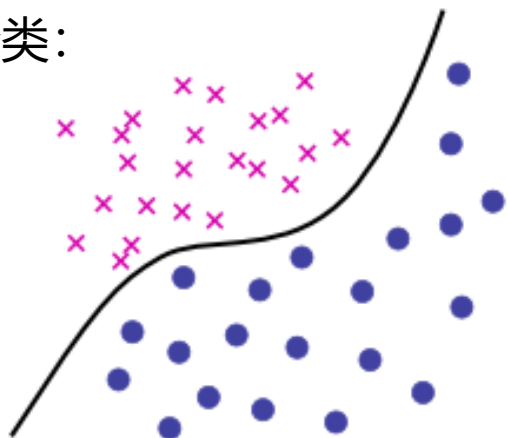
人工智能 vs 机器学习 vs 深度学习



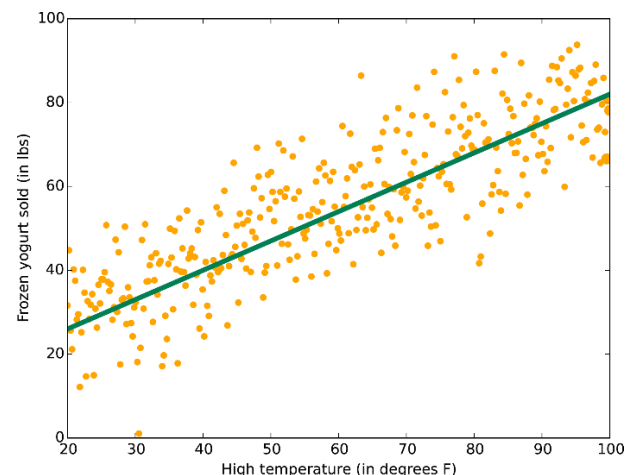
Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

# 机器学习基本概念与流程

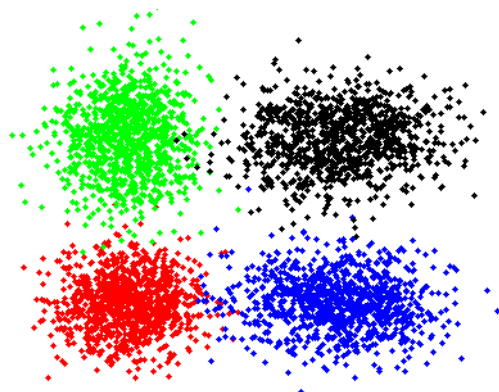
任务分类:



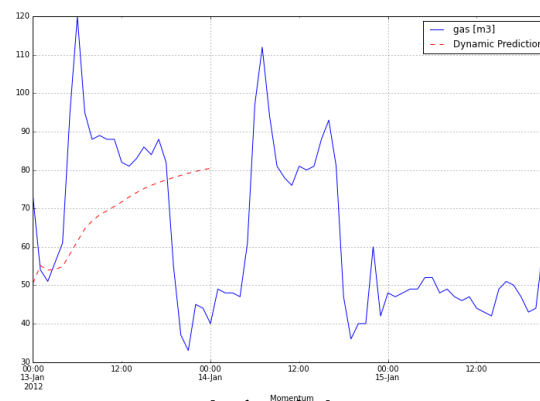
分类



回归



聚类



时序分析

# 机器学习基本概念与流程

---

## 分类与回归

- 应用：信用卡申请人风险评估、预测公司业务增长量、预测房价等
- 原理：

分类，将数据映射到预先定义的群组或类。算法要求基于数据属性值来定义类别，把具有某些特征的数据项映射到给定的某个类别上。

回归，用属性的历史数据预测未来趋势。算法首先假设一些已知类型的函数可以拟合目标数据，然后利用某种误差分析确定一个与目标数据拟合程度最好的函数。

- 区别：分类模型采用离散预测值，回归模型采用连续的预测值。

# 机器学习基本概念与流程

---

## 聚类

- 应用：根据症状归纳特定疾病、发现信用卡高级用户、根据上网行为对客户分群从而进行精确营销等

- 原理：

在没有给定划分类的情况下，根据信息相似度进行信息聚类。

聚类的输入是一组未被标记的数据，根据样本特征的距离或相似度进行划分。

划分原则是保持最大的组内相似性和最小的组间相似性。

挖掘未标记样本的Structure： 1) 聚类相似样本， 2) 异常样本检测

- 监督学习：学习的是带有标记的数据
- 非监督学习：学习的是未被标记的数据



# 机器学习基本概念与流程

---

## 时序模型

- 应用：下个季度的商品销量或库存量是多少？明天用电量是多少？
- 原理：

描述基于时间或其他序列的经常发生的规律或趋势，并对其建模。

与回归一样，用已知的数据预测未来的值，但这些数据的区别是变量所处时间的不同。重点考察数据之间在时间维度上的关联性。

# 机器学习基本概念与流程

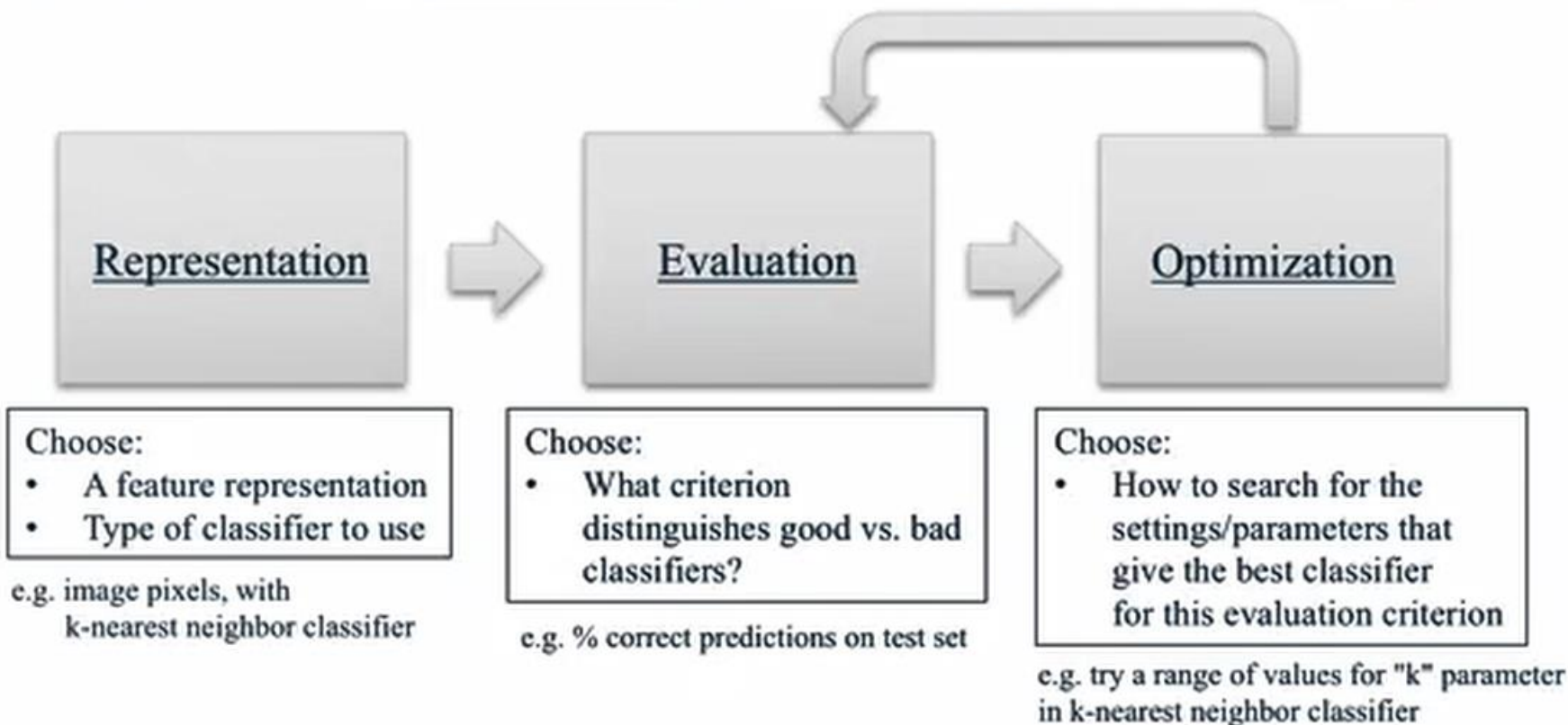
---

## 机器学习：问题描述

- “学习” 问题通常包括 $n$ 个样本数据（训练样本），然后预测未知数据（测试样本）的属性
- 每个样本包含的多个属性（多维数据）被称作“特征”
- 分类：
  - 监督学习，训练样本包含对应的“标签”，如识别问题
    - 分类问题，样本标签属于两类或多类（离散）
    - 回归问题，样本标签包括一个或多个连续变量（连续）
  - 无监督学习，训练样本的属性不包含对应的“标签”，如聚类问题

# 机器学习基本概念与流程

## 基本流程



# 机器学习基本概念与流程

## 特征表示

### Email

To: Chris Brooks  
From: Daniel Romero  
Subject: Next course offering  
Hi Daniel,  
Could you please send the outline for the  
next course offering? Thanks! -- Chris

Feature	Count
to	1
chris	2
brooks	1
from	1
daniel	2
romero	1
the	2
...	

### Feature representation

A list of words with  
their frequency counts

### Picture



A matrix of color  
values (pixels)

### Sea Creatures



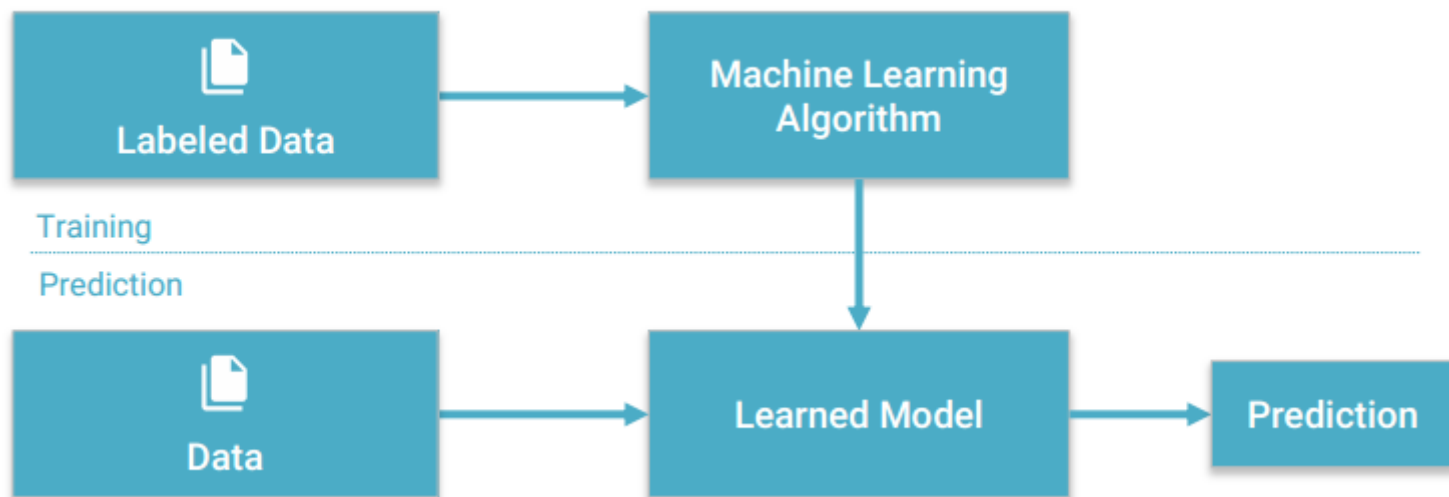
Feature	Value
DorsalFin	Yes
MainColor	Orange
Stripes	Yes
StripeColor1	White
StripeColor2	Black
Length	4.3 cm

A set of attribute values

# 机器学习基本概念与流程

## 定义

- Machine Learning is a type of Artificial Intelligence that provides computers with the ability to **learn without being explicitly programmed**.
- Provides **various techniques** that can learn from and make predictions on **DATA**.

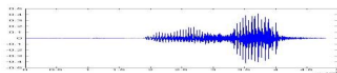


# 机器学习基本概念与流程

≈ 寻找一个函数

- 语音识别

$f(\text{语音波形}) = \text{“你好吗？”}$



- 图像识别

$f(\text{猫的图片}) = \text{“猫”}$



- 围棋对战

$f(\text{围棋棋盘}) = \text{“5-5” (下一步)}$



- 对话系统（如Siri）

$f(\text{“你好！” (用户发问)}) = \text{“您好！” (系统回应)}$

# 机器学习基本概念与流程

如何选择?

图像识别

$$f(\text{猫}) = \text{“猫”}$$



$$f_1(\text{猫}) = \text{“猫”}$$

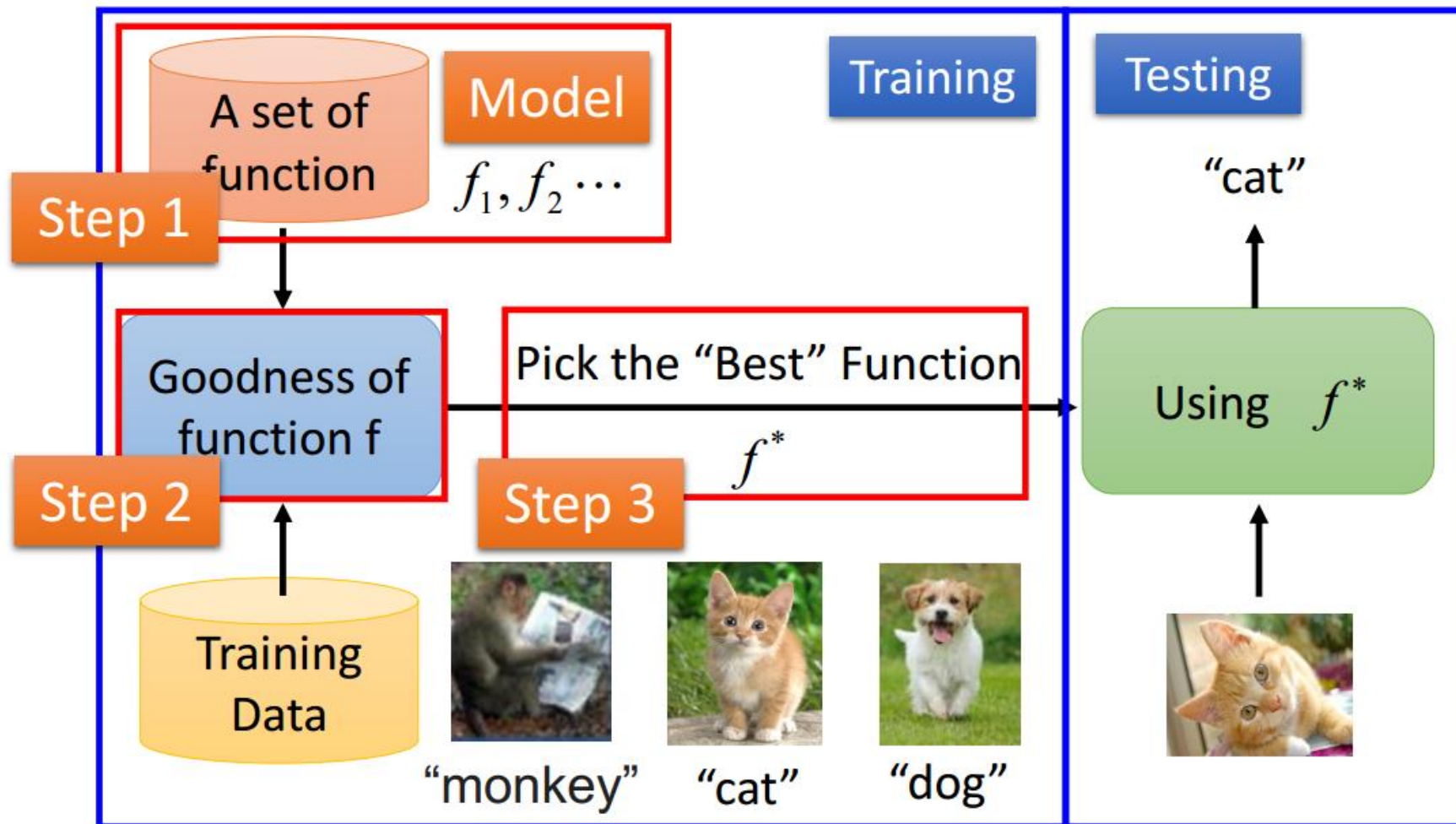
$$f_2(\text{猫}) = \text{“猴子”}$$

$$f_1(\text{狗}) = \text{“狗”}$$

$$f_2(\text{猫}) = \text{“蛇”}$$

# 机器学习基本概念与流程

## 基本框架



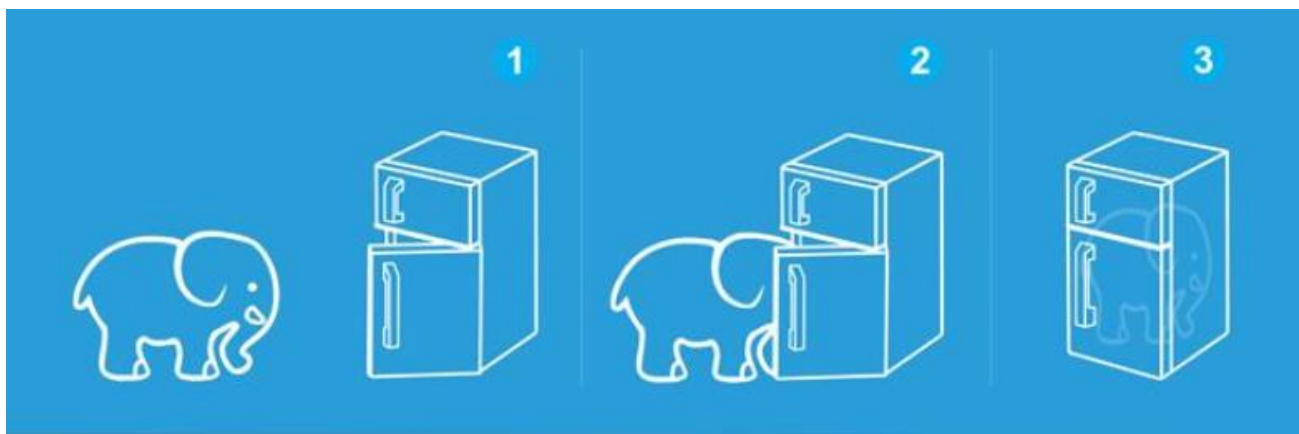


# 机器学习基本概念与流程

## 基本步骤

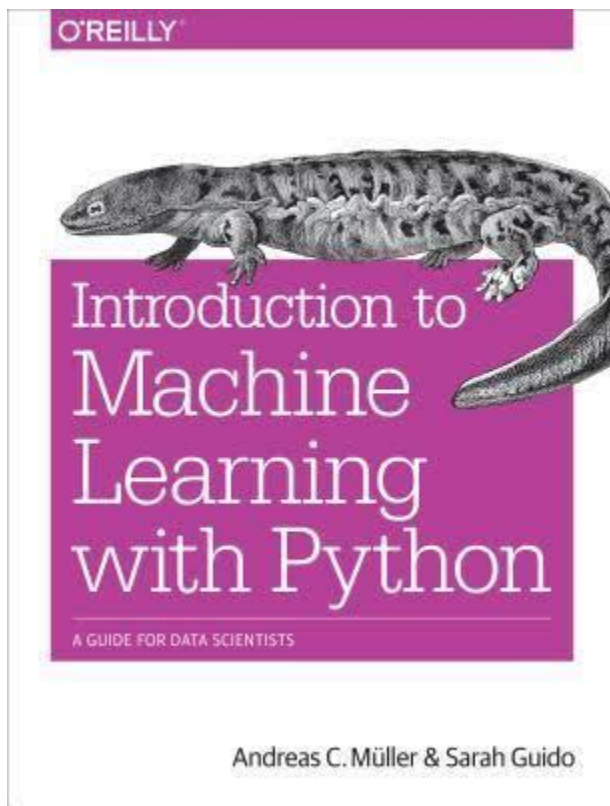


机器学习就是这么简单…



# 机器学习基本概念与流程

---



# Python机器学习库scikit-learn

## Machine Learning



what society thinks I  
do



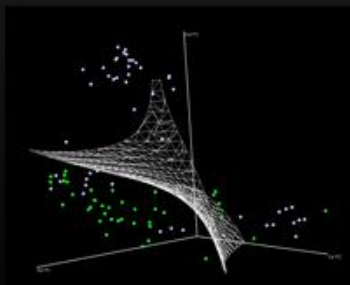
what my friends think  
I do



what my parents think  
I do

$$\begin{aligned} L_p &= \frac{1}{2} \|\mathbf{w}\|^2 - \sum_i \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w} + b) + \sum_i \alpha_i \\ \alpha_i &\geq 0, \forall i \\ \mathbf{w} &= \sum_i \alpha_i y_i \mathbf{x}_i, \sum_i \alpha_i y_i = 0 \\ \nabla \hat{g}(\theta_t) &= \frac{1}{n} \sum_{i=1}^n \nabla \ell(x_i, y_i; \theta_t) + \nabla r(\theta_t) \\ \theta_{t+1} &= \theta_t - \eta_t \nabla \ell(x_{i(t)}, y_{i(t)}; \theta_t) - \eta_t \cdot \nabla r(\theta_t) \\ \mathbb{E}_{i(t)}[\ell(x_{i(t)}, y_{i(t)}; \theta_t)] &= \frac{1}{n} \sum_i \ell(x_i, y_i; \theta_t) \end{aligned}$$

what other programmers  
think I do



what I think I do

```
>>> from sklearn import svm
```

what I really do

# Python机器学习库scikit-learn



# Python机器学习库scikit-learn

---



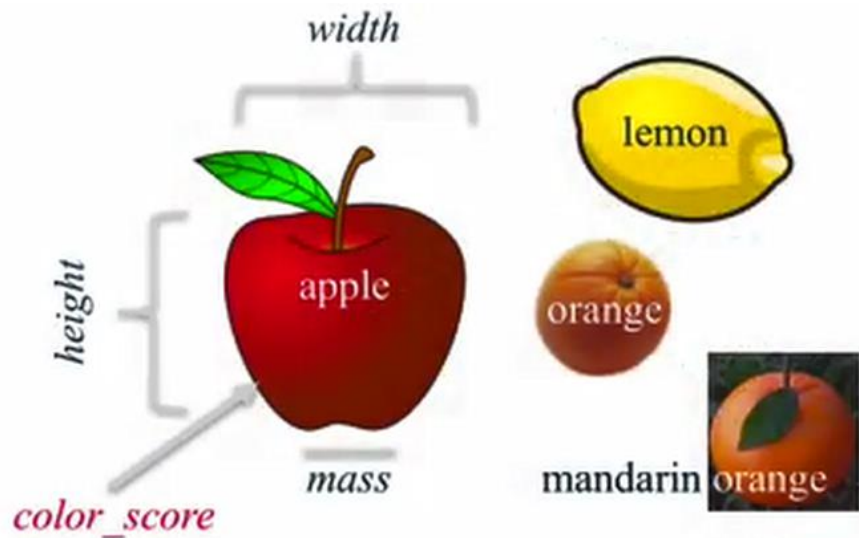
- 面向Python的免费机器学习库
- 包含分类、回归、聚类算法，比如：SVM、随机森林、k-means等
- 包含降维、模型筛选、预处理等算法
- 支持NumPy和SciPy数据结构
- 用户

<http://scikit-learn.org/stable/testimonials/testimonials.html>

- 安装
  - `pip install scikit-learn`
  - `conda install scikit-learn`

# Python机器学习库scikit-learn

## 水果识别



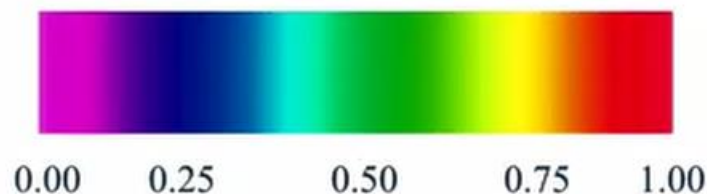
fruit\_data\_with\_colors.txt

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	granny_smith	192	8.4	7.3	0.55
1	1	apple	granny_smith	180	8.0	6.8	0.59
2	1	apple	granny_smith	176	7.4	7.2	0.60
3	2	mandarin	mandarin	86	6.2	4.7	0.80
4	2	mandarin	mandarin	84	6.0	4.6	0.79
5	2	mandarin	mandarin	80	5.8	4.3	0.77
6	2	mandarin	mandarin	80	5.9	4.3	0.81
7	2	mandarin	mandarin	76	5.8	4.0	0.81
8	1	apple	braeburn	178	7.1	7.8	0.92
9	1	apple	braeburn	172	7.4	7.0	0.89
10	1	apple	braeburn	166	6.9	7.3	0.93
11	1	apple	braeburn	172	7.1	7.6	0.92
12	1	apple	braeburn	154	7.0	7.1	0.88
13	1	apple	golden_delicious	164	7.3	7.7	0.70
14	1	apple	golden_delicious	152	7.6	7.3	0.69
15	1	apple	golden_delicious	156	7.7	7.1	0.69

# Python机器学习库scikit-learn

水果识别

color\_score 特征:



Color category	color_score
Red	0.85 - 1.00
Orange	0.75 - 0.85
Yellow	0.65 - 0.75
Green	0.45 - 0.65

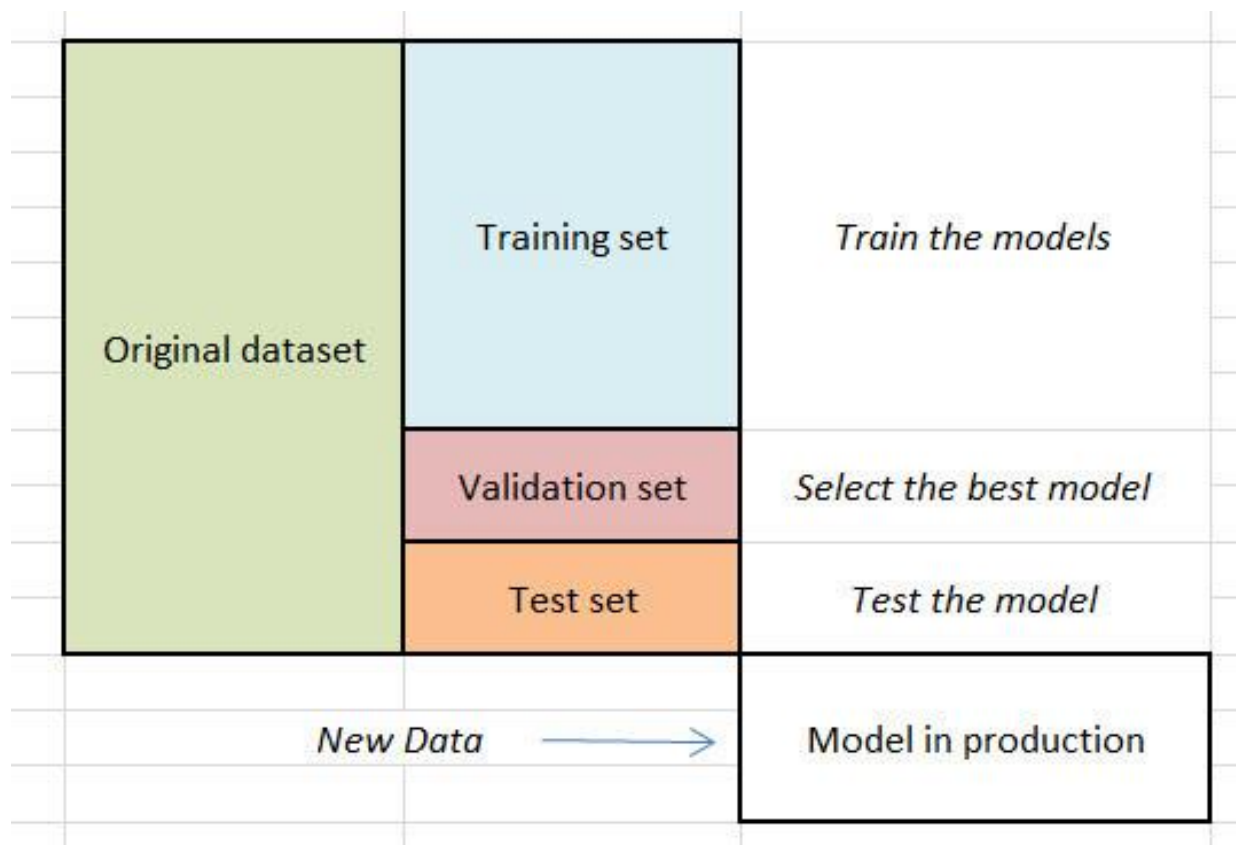
lect04\_eg01.ipynb



# Python机器学习库scikit-learn

## 数据集划分

训练集 vs 验证集 vs 测试集

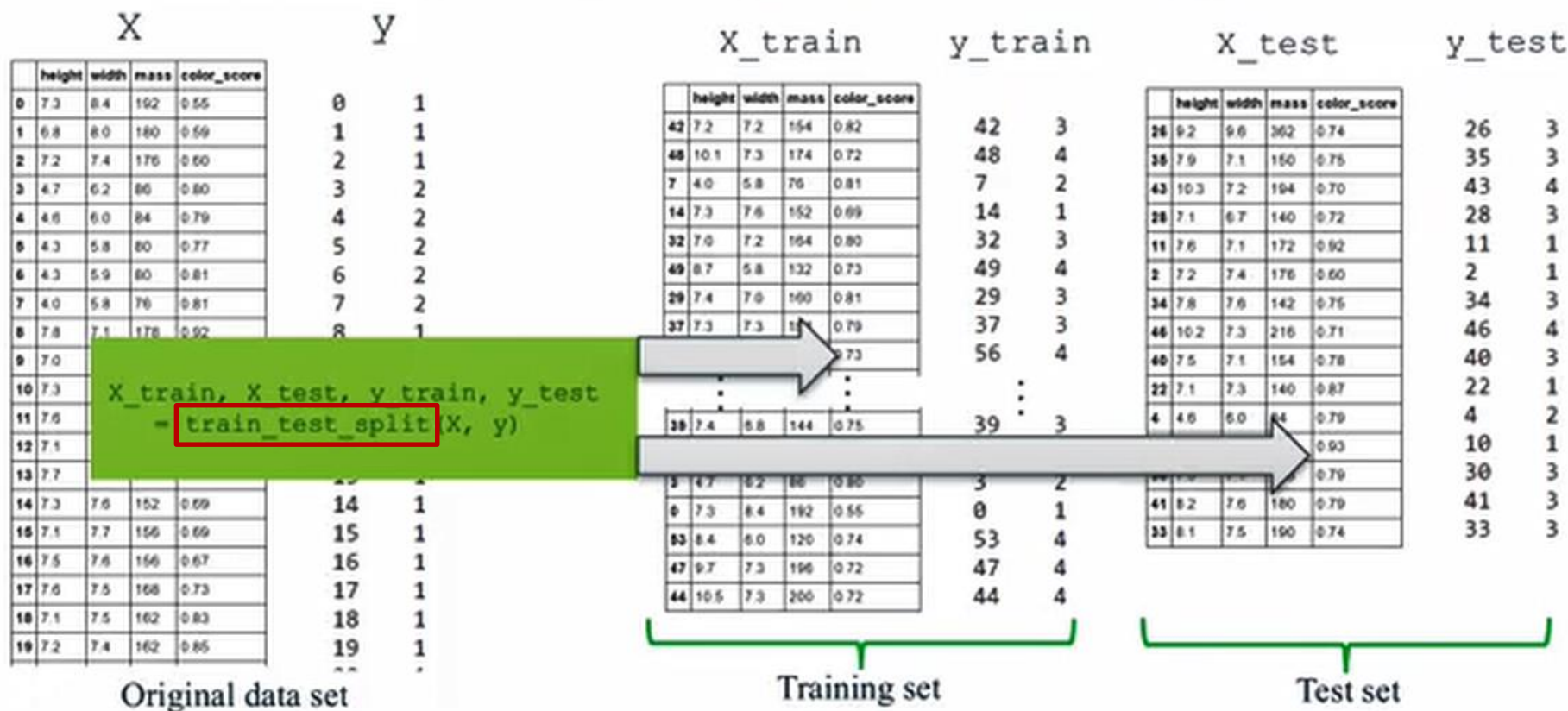




# Python机器学习库scikit-learn

## 数据集划分

### 训练集 vs 测试集



# 目录

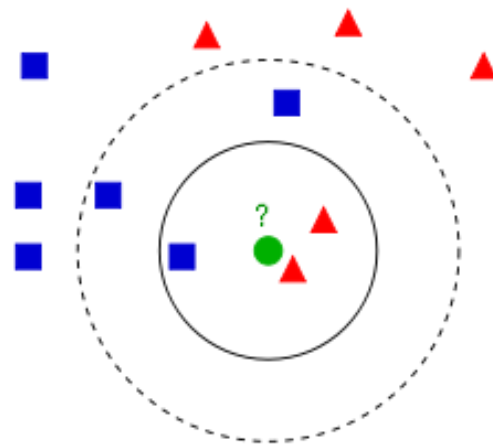
---

- 机器学习基本概念与流程
- **k-近邻算法 (kNN)**
- 线性回归 (Linear Regression)
- 逻辑回归 (Logistic Regression) 及Softmax回归
- 正则化
- 实战案例3-1：贷款违约预测 (1)

# kNN

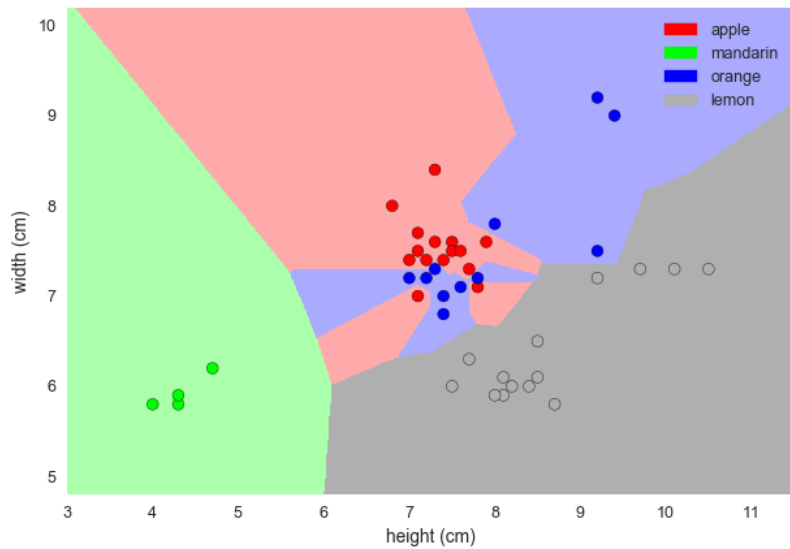
## kNN (k-NearestNeighbor), k-近邻算法

- 是一种基于样本/实例的算法
- 无参（nonparametric）模型
- 可用于分类和回归
- 步骤：
  1. 计算出测试样本和所有训练样本的距离；
  2. 为测试样本选择k个与其距离最小的训练样本；
  3. 统计出k个训练样本中大多数样本所属的分类；
  4. 这个分类就是待分类数据所属的分类
- 问题：
  - k必须是奇数吗？

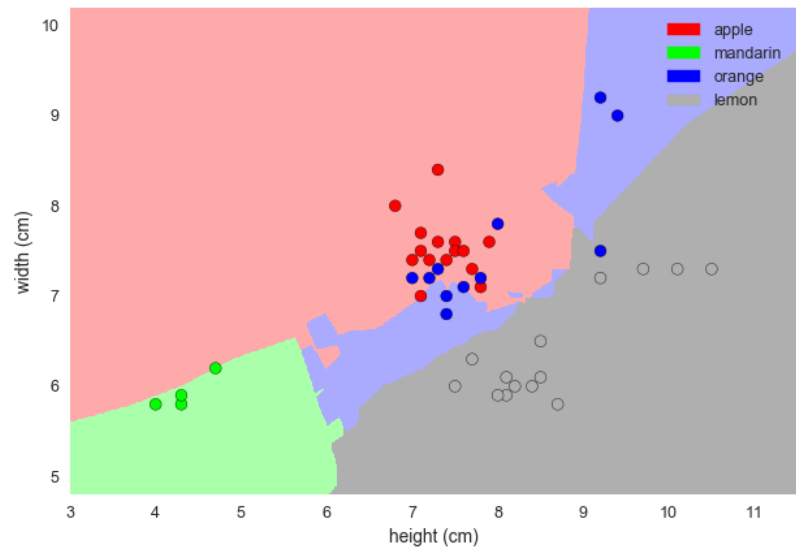


# kNN

- 应用在“水果识别”数据集中



k=1

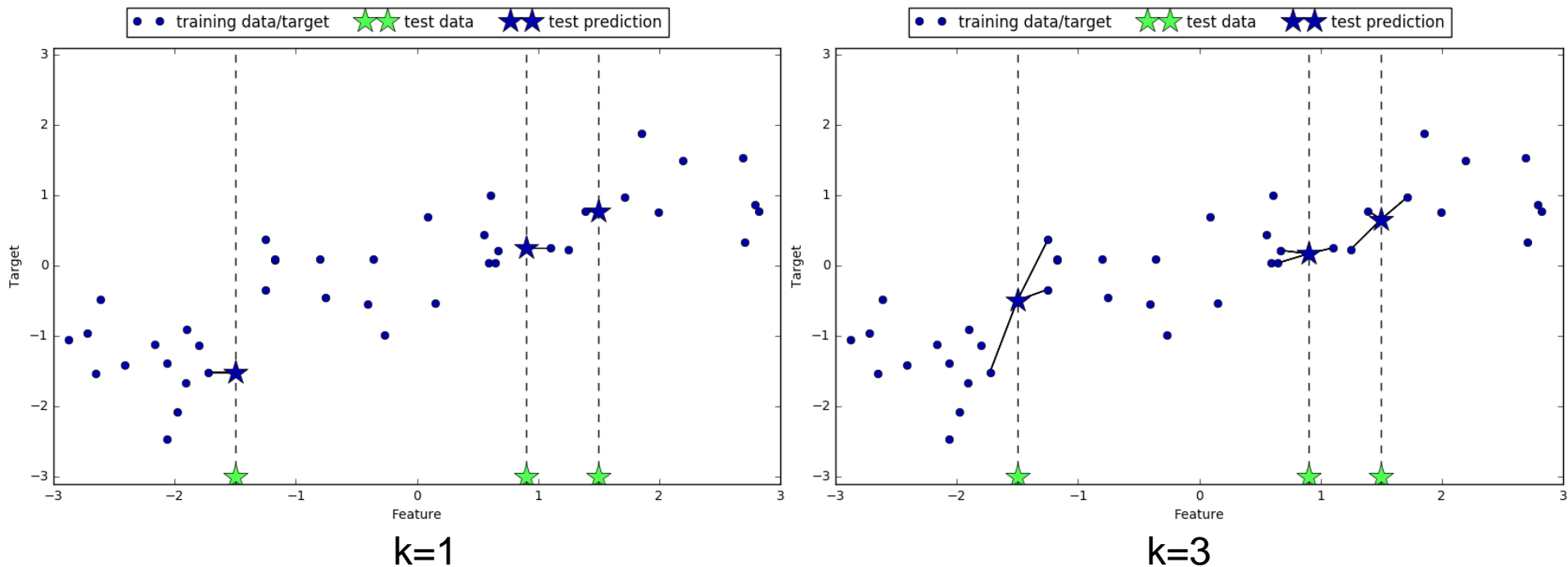


k=10

- 问题:
  - 如果k是无穷大, 结果是怎样的?

# kNN

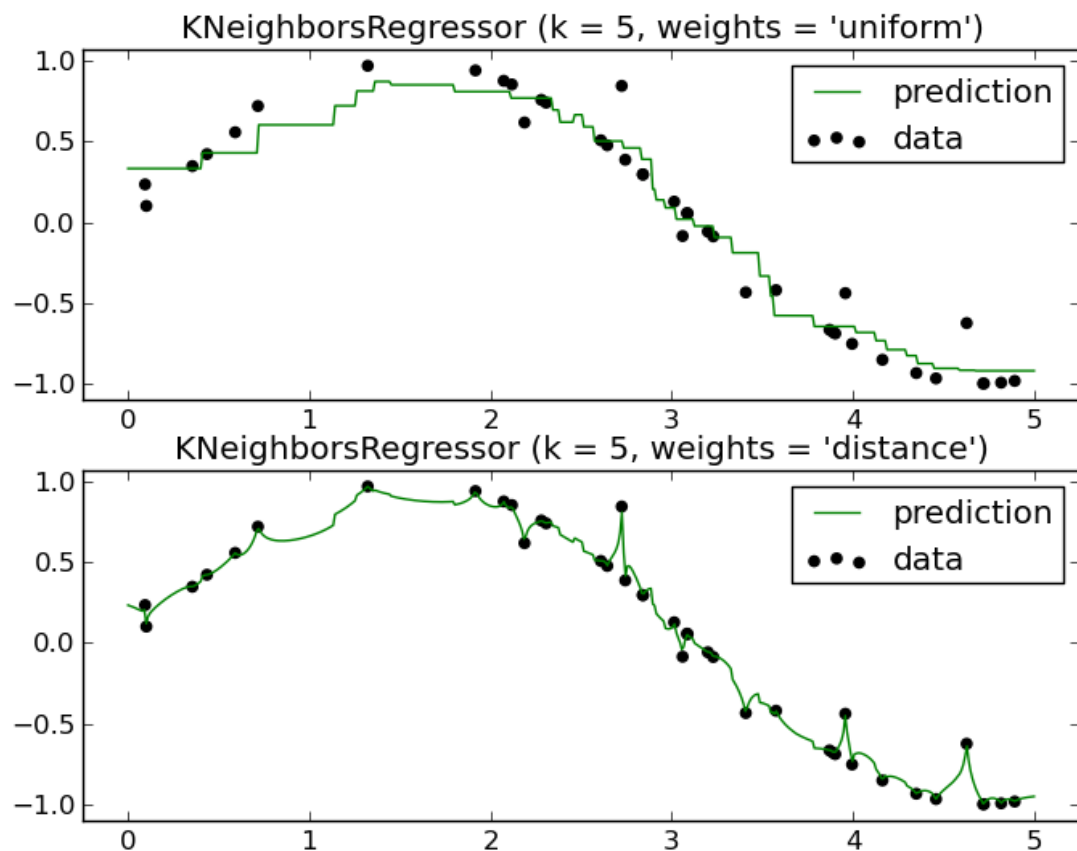
- kNN用于回归



ref: <https://elvinouyang.github.io/study%20notes/python-datasets-and-knn/>

# kNN

- kNN用于回归

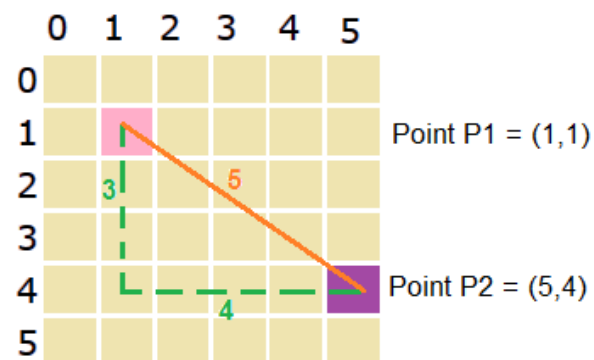


ref: <https://stats.stackexchange.com/questions/104255/why-would-anyone-use-knn-for-regression>

# kNN

- kNN中的距离
  - [相似性度量](#)
  - 闵式距离
  - 欧式距离（默认）
  - $p=1$ ，曼哈顿距离
  - $p=2$ ，欧式距离
  - ...

$$d(\mathbf{x}, \mathbf{y}) = \left( \sum_{i=0}^{n-1} |x_i - y_i|^p \right)^{1/p}$$



$$\text{Euclidean distance} = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

$$\text{Manhattan distance} = |5-1| + |4-1| = 7$$

# Python机器学习库scikit-learn

## kNN (k-NearestNeighbor), k-近邻算法

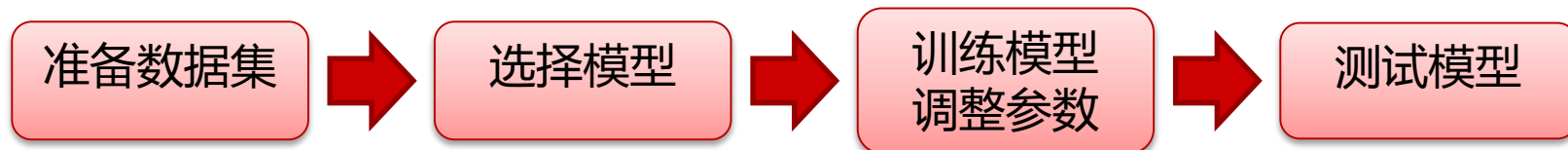
- 需要注意的问题
  1. 相似性度量
  2. 紧邻点个数, 通过交叉验证得到最优紧邻点个数
- kNN优缺点

优点	缺点
<ul style="list-style-type: none"><li>• 算法简单直观, 易于实现</li><li>• 不需要额外的数据, 只依靠数据(样本)本身</li></ul>	<ul style="list-style-type: none"><li>• 计算量较大, 分类速度慢</li><li>• 需要预先指定k值</li></ul>



# Python机器学习库scikit-learn

## 使用scikit-learn的流程



- |         |         |         |         |
|---------|---------|---------|---------|
| • 数据处理  | • 根据任务选 | • 根据经验设 | • 预测    |
| • 特征工程  | 择模型     | 定参数     | • 识别    |
| • 训练集、测 | • 分类模型  | • 交叉验证确 | • ..... |
| 试集分割    | • 回归模型  | 定最优参数   |         |
|         | • 聚类模型  |         |         |
|         | • ..... |         |         |

# Python机器学习库scikit-learn

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
%matplotlib inline

# 加载数据集
fruits_df = pd.read_table('fruit_data_with_colors.txt')

X = fruits_df[['mass', 'width', 'height', 'color_score']]
y = fruits_df['fruit_label']

# 分割数据集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=1/4, random_state=0)

# 建立模型
knn = KNeighborsClassifier(n_neighbors=5)

# 训练模型
knn.fit(X_train, y_train)

# 验证模型
y_pred = knn.predict(X_test)

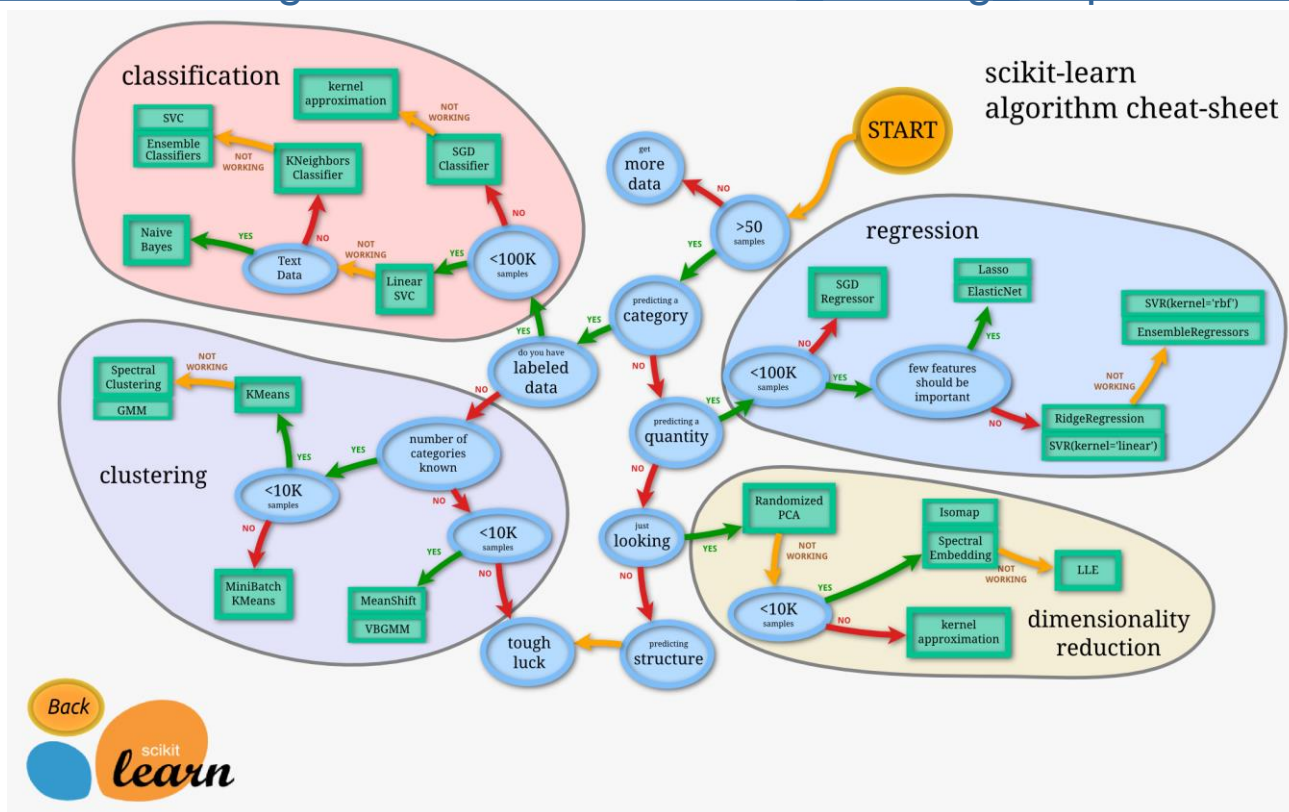
acc = accuracy_score(y_test, y_pred)
print('准确率: ', acc)
```

# Python机器学习库scikit-learn

## 选择模型

- 模型选择路线图

[http://scikit-learn.org/stable/tutorial/machine\\_learning\\_map/index.html](http://scikit-learn.org/stable/tutorial/machine_learning_map/index.html)



# Python机器学习库scikit-learn

---

## 训练模型

- Estimator对象
- 从训练数据学习得到的
- 可以是分类算法、回归算法或者是特征提取算法
- fit方法用于训练Estimator
- Estimator的参数可以训练前初始化，或者之后更新
- get\_params()返回之前定义的参数
- score()对Estimator进行评分
  - 回归模型：使用“决定系数”评分（Coefficient of Determination）
  - 分类模型：使用“准确率”评分（accuracy）

# Python机器学习库scikit-learn

---

## 测试模型

- `model.predict(X_test)`
  - 返回测试样本的预测标签
- `model.score(X_test, y_test)`
  - 根据预测值和真实值计算评分

# 目录

---

- 机器学习基本概念与流程
- k-近邻算法（kNN）
- 线性回归（Linear Regression）
- 逻辑回归（Logistic Regression）及Softmax回归
- 正则化
- 实战案例3-1：贷款违约预测 (1)

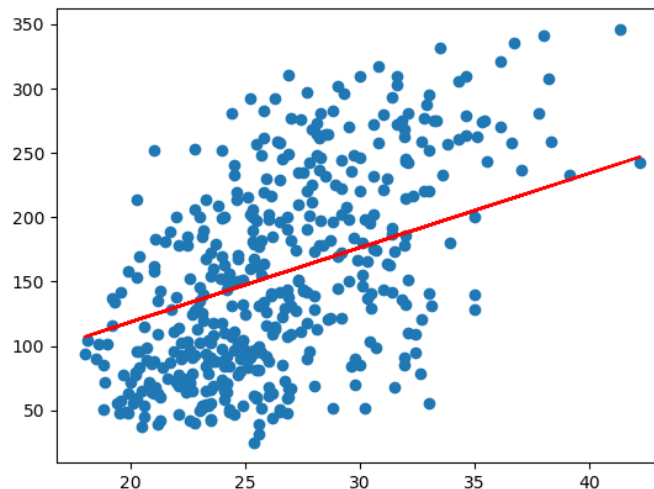
# Linear Regression

## 线型回归

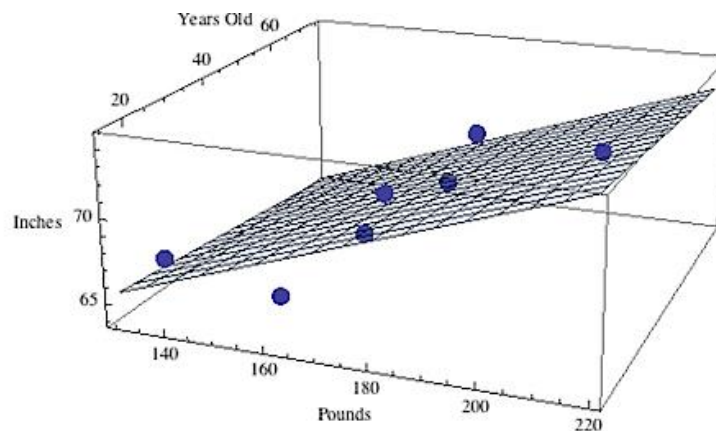
$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

$$h(x) = \sum_{i=0}^n \theta_i x_i = \theta^T x$$

$$J(\theta) = \frac{1}{2} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$



低维



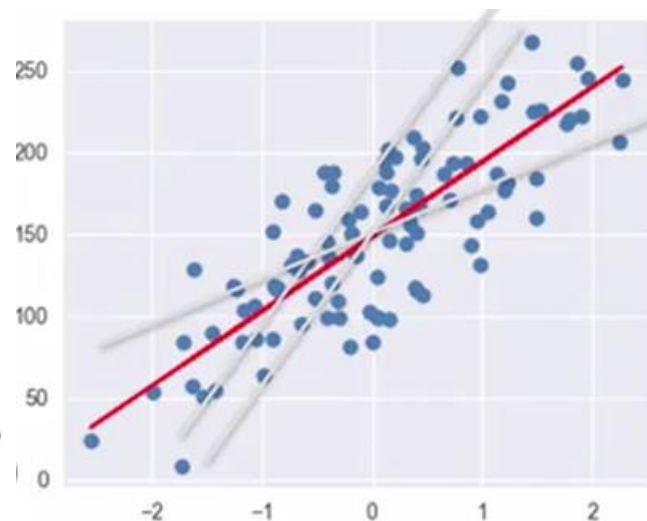
高维

# Linear Regression

## 线型回归

- 如何求解参数theta?
1. 梯度下降法（参考第一讲内容）
  2. 最小二乘法（Least Square）

$$\sum_{j=1}^n X_{ij}\beta_j = y_i, (i = 1, 2, \dots, m), \quad \mathbf{X}\boldsymbol{\beta} = \mathbf{y},$$



$$\mathbf{X} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ X_{21} & X_{22} & \cdots & X_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ X_{m1} & X_{m2} & \cdots & X_{mn} \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix}, \quad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$\hat{\boldsymbol{\beta}} = \arg \min_{\boldsymbol{\beta}} S(\boldsymbol{\beta}), \quad S(\boldsymbol{\beta}) = \sum_{i=1}^m \left| y_i - \sum_{j=1}^n X_{ij}\beta_j \right|^2 = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$$

对beta进行求导  
令导数为0

$$\Rightarrow (\mathbf{X}^T \mathbf{X}) \hat{\boldsymbol{\beta}} = \mathbf{X}^T \mathbf{y}. \quad \Rightarrow \hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$



# Linear Regression

---

## 线型回归

- 如何选择？
  1. 梯度下降法（Gradient Descent）
  2. 最小二乘法（Least Square）
- 数据量小的时候，使用Least Square;
- 数据量大的时候，可以考虑使用Gradient Descent

# Linear Regression

sklearn中调用线性回归

```
from sklearn.linear_model import LinearRegression

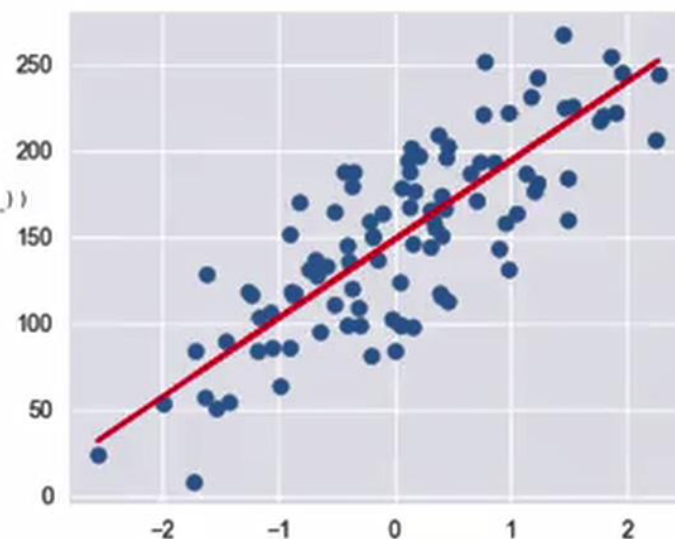
X_train, X_test, y_train, y_test =
    train_test_split(X_R1, y_R1, random_state = 0)

linreg = LinearRegression().fit(X_train, y_train)

print("linear model intercept (b): {}".format(linreg.intercept_))
print("linear model coeff (w): {}".format(linreg.coef_))
```

$$\hat{y} = \mathbf{w}_0 x_0 + b$$

Diagram illustrating the linear regression equation  $\hat{y} = \mathbf{w}_0 x_0 + b$ . The term  $\mathbf{w}_0$  is labeled as `linreg.coef_` and the term  $b$  is labeled as `linreg.intercept_`.



# 目录

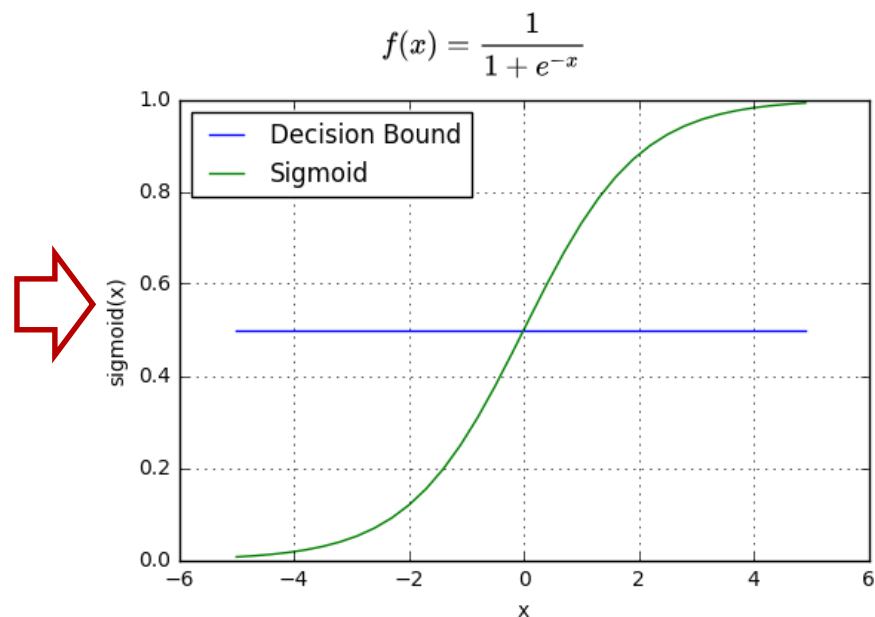
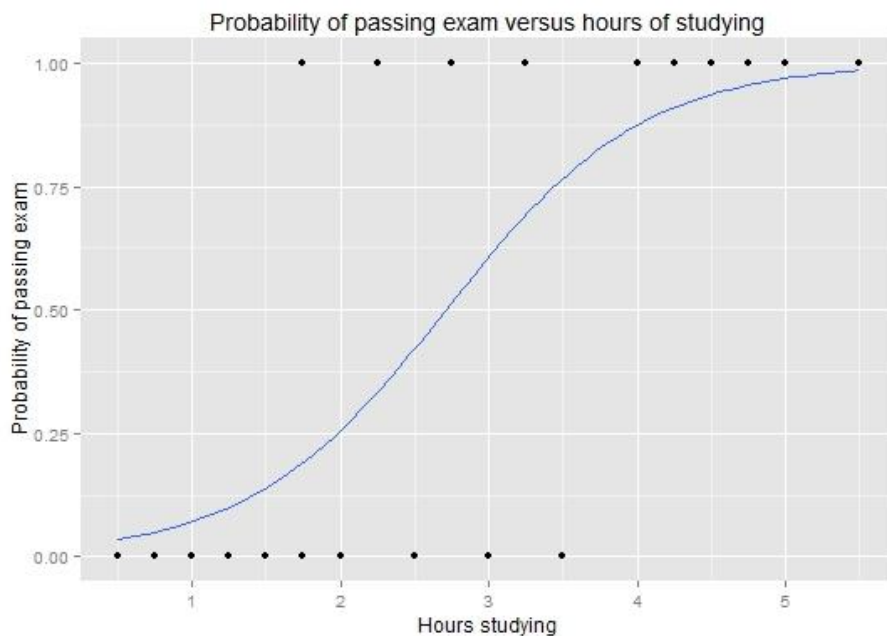
---

- 机器学习基本概念与流程
- k-近邻算法 (kNN)
- 线性回归 (Linear Regression)
- 逻辑回归 (Logistic Regression) 及 Softmax 回归
- 正则化
- 实战案例3-1: 贷款违约预测 (1)

# Logistic Regression

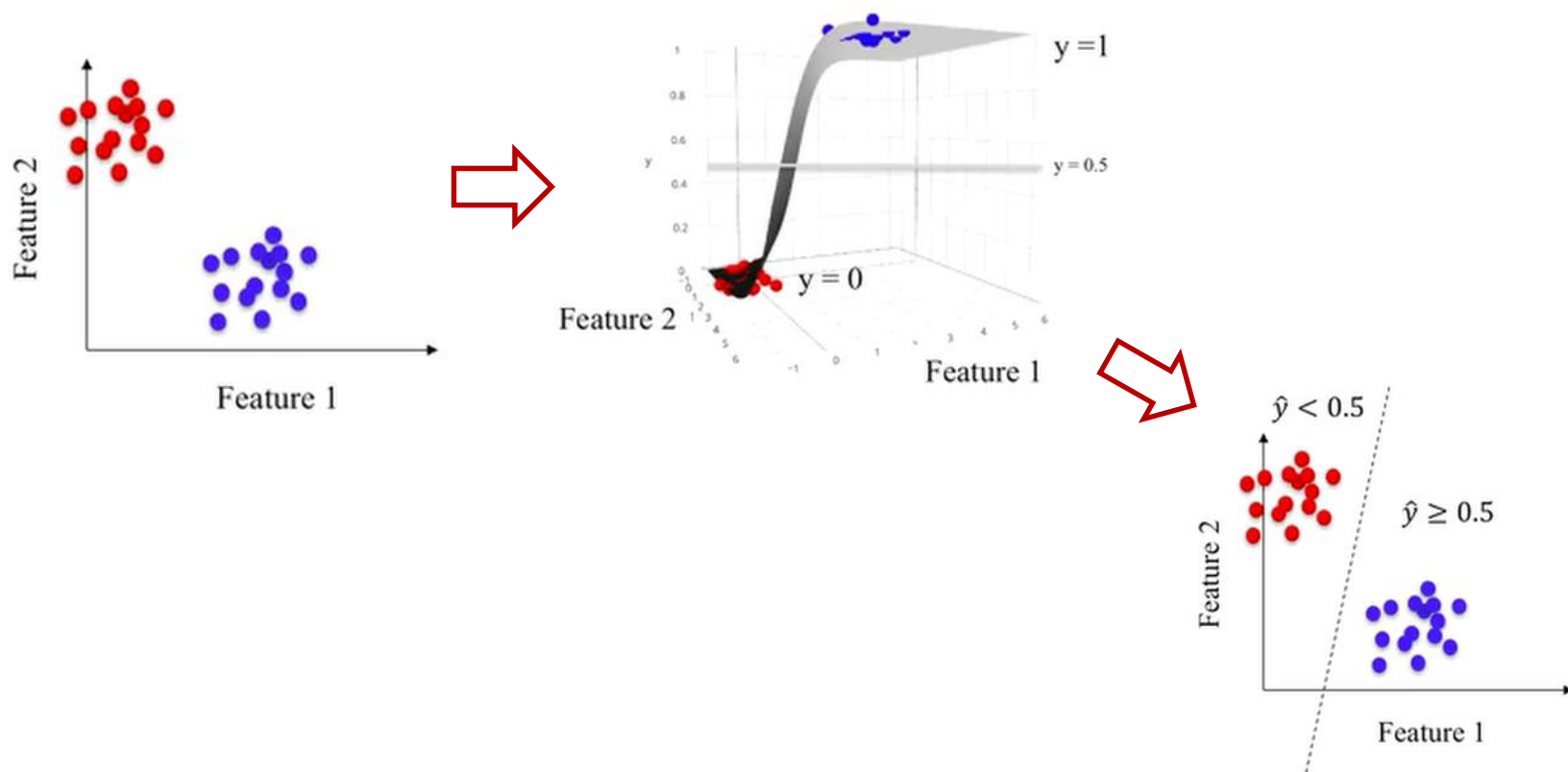
Q: 现有20个学生投入0-6个小时学习课程的记录，分析投入的时间和是否通过考试的概率的关系。

Hours	0.50	0.75	1.00	1.25	1.50	1.75	1.75	2.00	2.25	2.50	2.75	3.00	3.25	3.50	4.00	4.25	4.50	4.75	5.00	5.50
Pass	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1



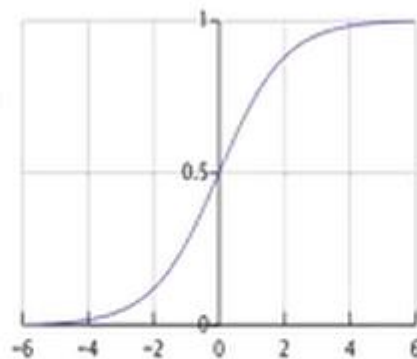
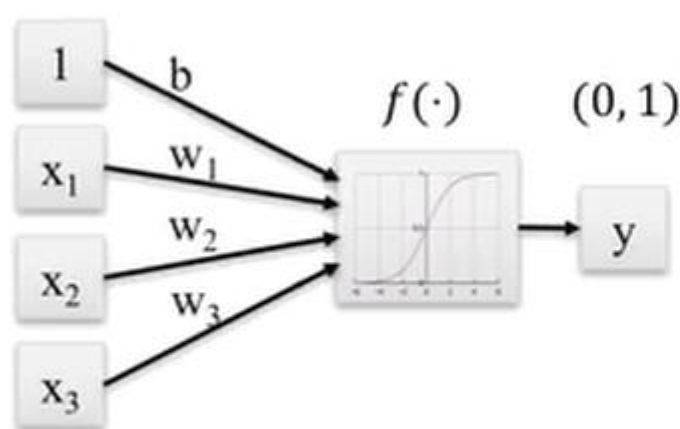
# Logistic Regression

- 例子：样本中包含二维特征



# Logistic Regression

Input features



The logistic function transforms real-valued input to an output number  $y$  between 0 and 1, interpreted as the probability the input object belongs to the positive class, given its input features  $(x_0, x_1, \dots, x_n)$

$$\begin{aligned}\hat{y} &= \text{logistic}(\hat{b} + \hat{w}_1 \cdot x_1 + \dots \hat{w}_n \cdot x_n) \\ &= \frac{1}{1 + \exp[-(\hat{b} + \hat{w}_1 \cdot x_1 + \dots \hat{w}_n \cdot x_n)]}\end{aligned}$$

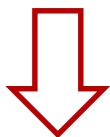
# Logistic Regression

Logistic Regression中的损失函数

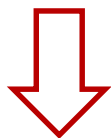
$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{Cost}(h_{\theta}(x^{(i)}), y^{(i)})$$

$$\text{Cost}(h_{\theta}(x), y) = -\log(h_{\theta}(x)) \quad \text{if } y = 1$$

$$\text{Cost}(h_{\theta}(x), y) = -\log(1 - h_{\theta}(x)) \quad \text{if } y = 0$$

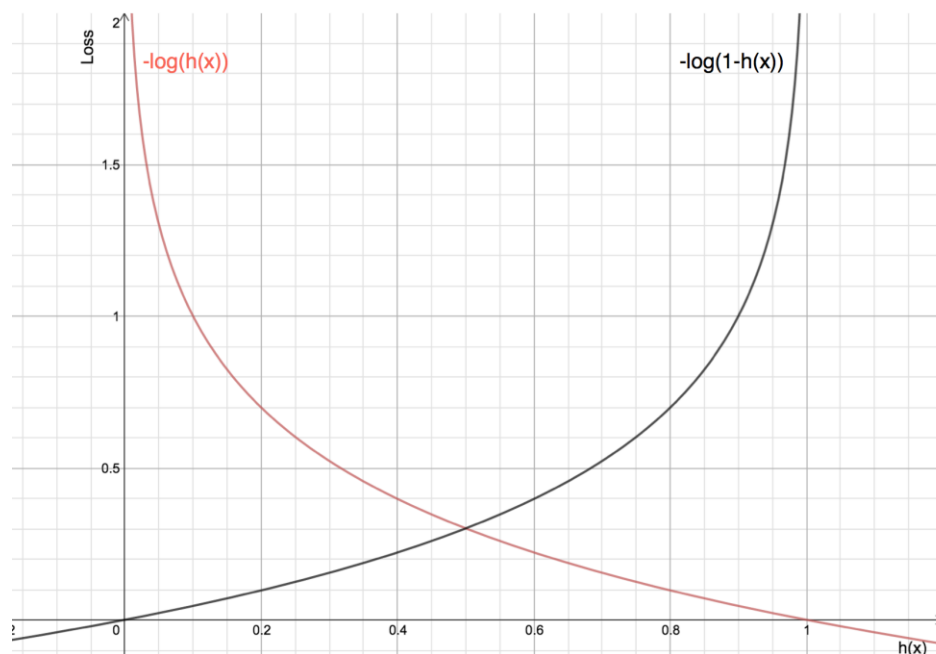


$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)}))]$$



梯度下降求解参数

cross-entropy loss或logloss

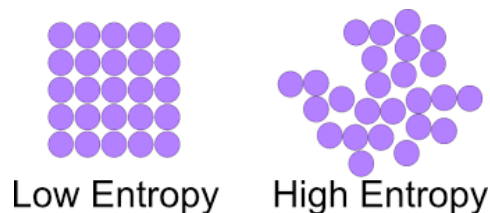


# Logistic Regression

- 熵 (Entropy)

在信息论中，设离散随机变量  $X$  的概率分布为  $P(X = x^{(i)}) = p_i, i = 1, 2, \dots, n$  的熵(Entropy)的定义为：

$$H(p) = - \sum_{i=1}^n p_i \log p_i$$



- 交叉熵 (Cross Entropy)

关于同一组事件  $x^{(1)}, \dots, x^{(n)}$  的两个分布  $p, q$ ，其交叉熵(Cross-Entropy)的定义如下：

$$H(p, q) = - \sum_{i=1}^n p_i \log q_i$$

当两个分布完全相同时，交叉熵取最小值。

交叉熵可以衡量两个分布之间的相似度，交叉熵越小两个分布越相似。



# Softmax Regression

## 二分类问题

- sigmoid function 
$$h_{\theta}(x) = \frac{1}{1 + \exp(-\theta^T x)},$$

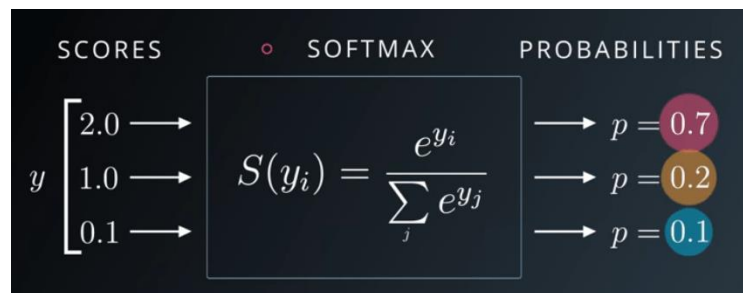
$$h_{\theta}(y = 1 | x; \theta) = \frac{1}{1 + e^{-\theta^T x}} = \frac{e^{\theta^T x}}{e^{\theta^T x} + 1} = \frac{e^{y=1}}{e^{y=0} + e^{y=1}}$$

$$h_{\theta}(y = 0 | x; \theta) = 1 - h_{\theta}(y = 1 | x; \theta) = \frac{1}{e^{\theta^T x} + 1} = \frac{e^{y=0}}{e^{y=1} + e^{y=0}}$$



## 多分类问题

- softmax function



# Softmax Regression

---

## Softmax Regression

$$h_{\theta}(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)}; \theta) \\ p(y^{(i)} = 2|x^{(i)}; \theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)}; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^k e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

↓

$$\theta = \begin{bmatrix} -\theta_1^T \\ -\theta_2^T \\ \vdots \\ -\theta_k^T \end{bmatrix}$$

# 线性模型

---

- 线型模型的优缺点

优点	缺点
<ul style="list-style-type: none"><li>• 模型简单，容易训练</li><li>• 能快速预测</li><li>• 适用于较大（数据量，特征维度）的数据集</li><li>• 对于预测结果能很好地解释原因</li></ul>	<ul style="list-style-type: none"><li>• 对于低维度的数据，可能没有其他分类器表现的好</li><li>• 不能很好地去分类不能线性分割的数据集（考虑使用非线性核函数，即 <b>kernel SVM</b>）</li></ul>

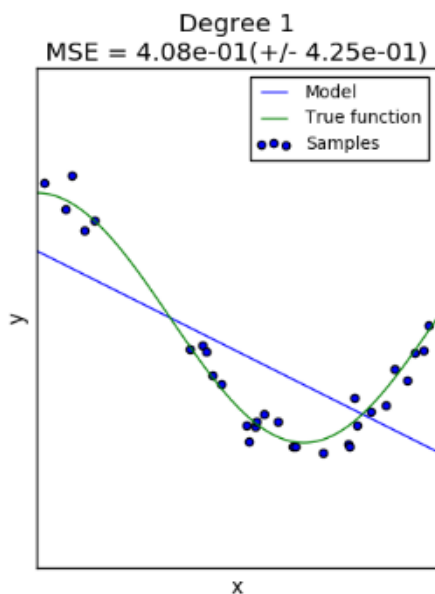
# 目录

---

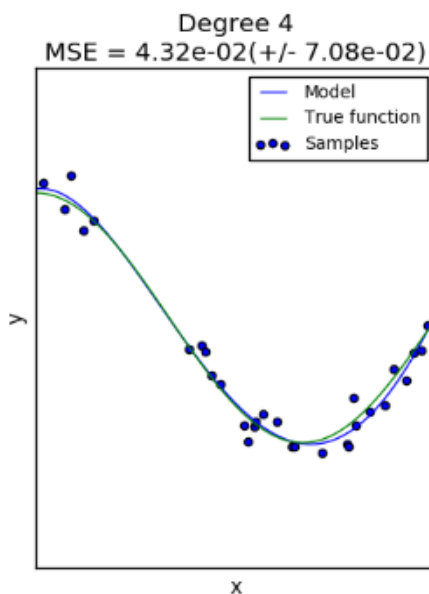
- 机器学习基本概念与流程
- k-近邻算法（kNN）
- 线性回归（Linear Regression）
- 逻辑回归（Logistic Regression）及Softmax回归
- 正则化
- 实战案例3-1：贷款违约预测 (1)

# 正则化

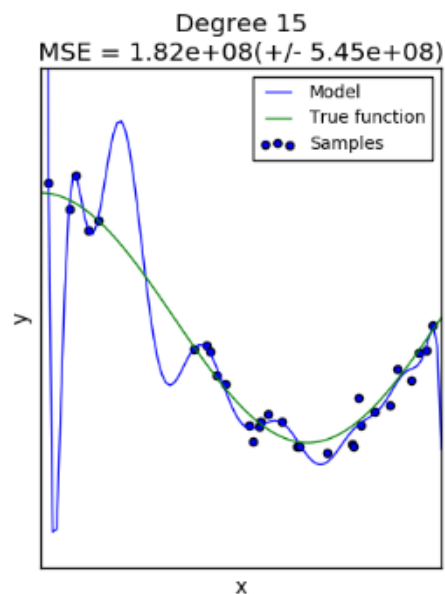
- 过拟合
  - 是指在调适一个统计模型时，使用**过多**参数。模型对于训练数据拟合**程度过当**，以致太适应训练数据而非一般情况。
  - 在训练数据上表现非常好，但是在测试数据或验证数据上表现很差。



欠拟合



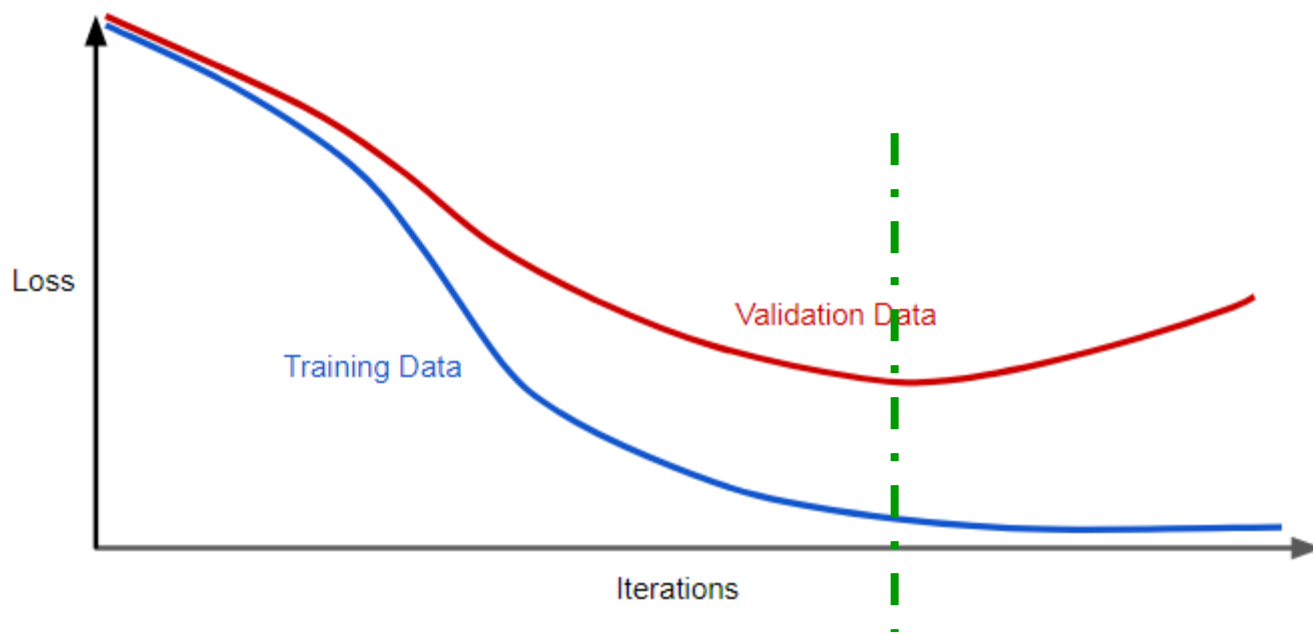
“刚刚好”



过拟合

# 正则化

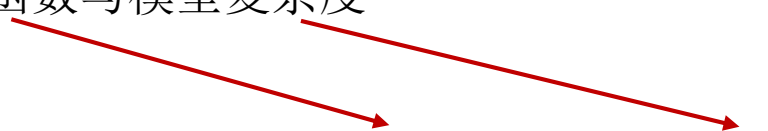
- 正则化



# 正则化

---

- 正则化
  - 控制模型复杂度，模型复杂度越高，越容易过拟合
  - 平衡损失函数与模型复杂度


$$\text{minimize: } Loss(Data | Model) + complexity(Model)$$

- 衡量模型复杂度
  - 模型学习得到的权重越大，模型复杂度越高
  - L2 正则化
    - $complexity(model) = \text{sum of the squares of the weights}$
    - 惩罚特别大的权重项

# 正则化

---

- 正则化 A Loss Function with  $L_2$  Regularization

$$L(\mathbf{w}, D) + \lambda || \mathbf{w} ||_2^2$$

Where:

$L$ : Aim for low training error

$\lambda$ : A scalar value that controls how weights are balanced

$\mathbf{w}$ : Balances against complexity

$_2^2$ : The square of the  $L_2$  normalization of  $\mathbf{w}$

- $\lambda$  值越大，正则化越强，表示需要更多关注模型的复杂度，适用于测试集中的样本与训练集中的样本相差比较大时；
- $\lambda$  值越小，正则化越弱，表示需要更多关注损失函数，适用于测试集中的样本与训练集中的样本相差不是很大



# 正则化

---

- 正则化
- 例子:

$$L_2 \text{ regularization term} = \|\mathbf{w}\|_2^2 = w_1^2 + w_2^2 + \dots + w_n^2$$

$$\{w_1 = 0.2, w_2 = 0.5, w_3 = 5, w_4 = 1, w_5 = 0.25, w_6 = 0.75\}$$

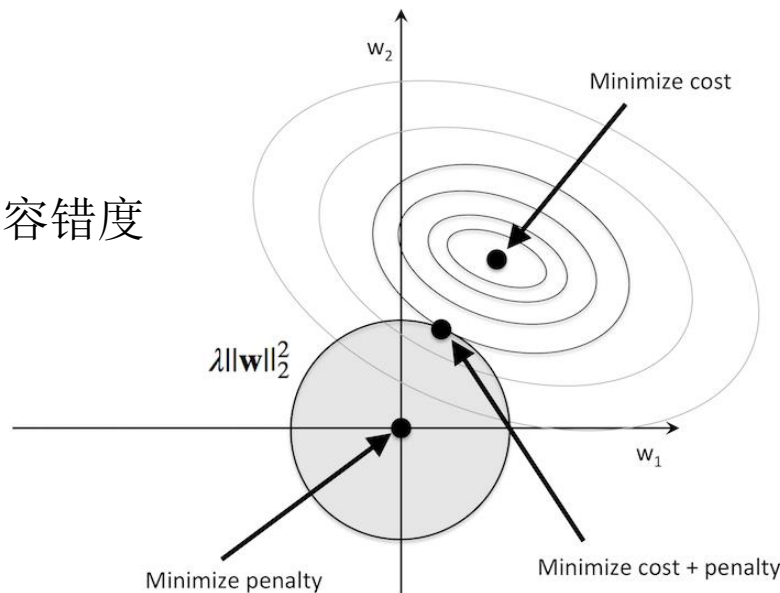
$$\begin{aligned} & w_1^2 + w_2^2 + \mathbf{w_3^2} + w_4^2 + w_5^2 + w_6^2 \\ &= 0.2^2 + 0.5^2 + \mathbf{5^2} + 1^2 + 0.25^2 + 0.75^2 \\ &= 0.04 + 0.25 + \mathbf{25} + 1 + 0.0625 + 0.5625 \\ &= 26.915 \end{aligned}$$

# 正则化

- 正则化

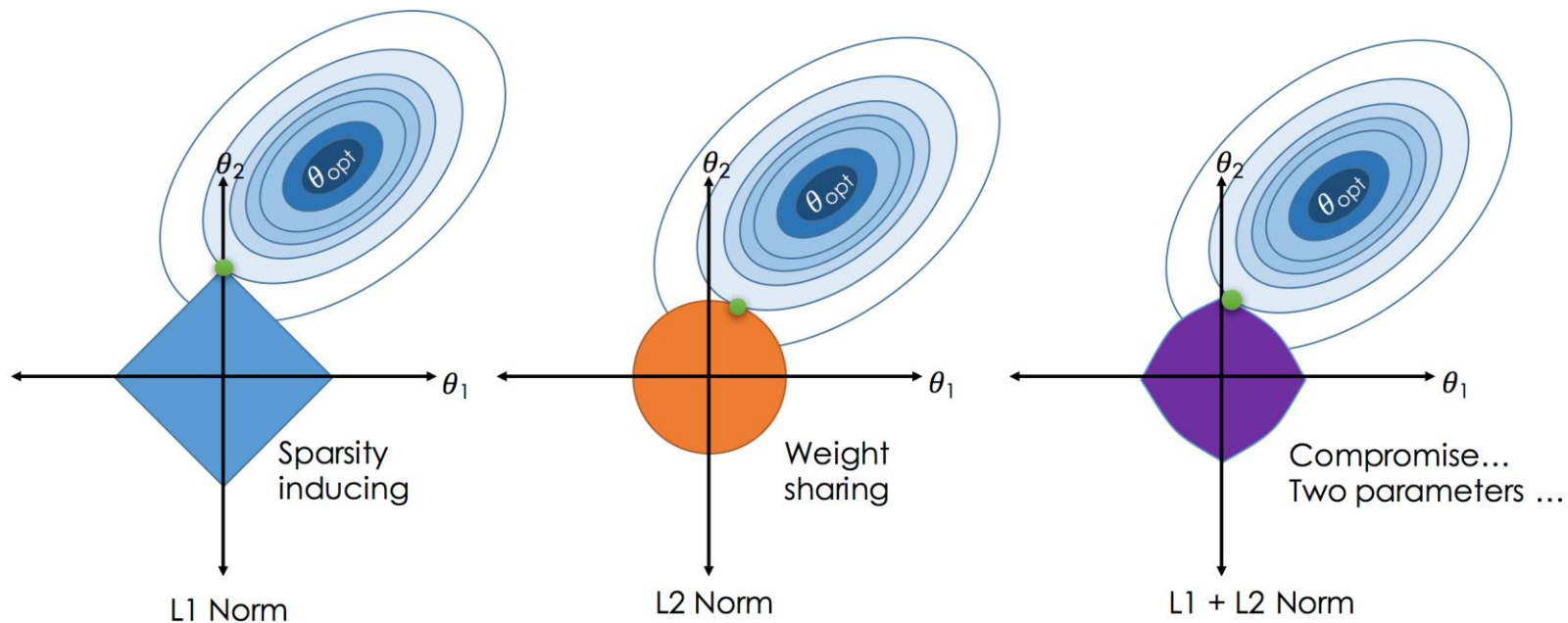
$$L = - \sum_{i=1}^n \log g(y_i z_i) + \frac{\lambda}{2} \sum_{k=1}^l w_k^2 \quad \text{正则项}$$

- 注意：sklearn中，logistic regression的参数C是正则项系数的倒数， $C=1/\lambda$
- 正则项中的C值决定了正则化的强度
- $\lambda$  值越大（C值越小），正则化越强
  - 对于单个样本的错误分类具有较强的容错度
- $\lambda$  值越小（C值越大），正则化越弱
  - 尽可能地去拟合训练样本的数据
  - 对于分类器来说，每个样本都很重要



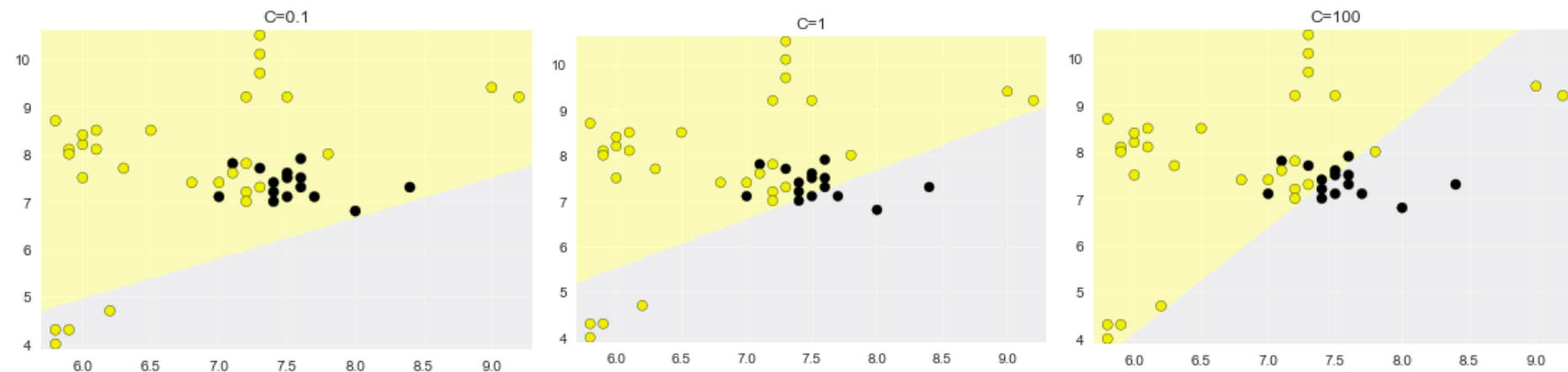
# 正则化

- 正则化



# 正则化

- 正则化



# 目录

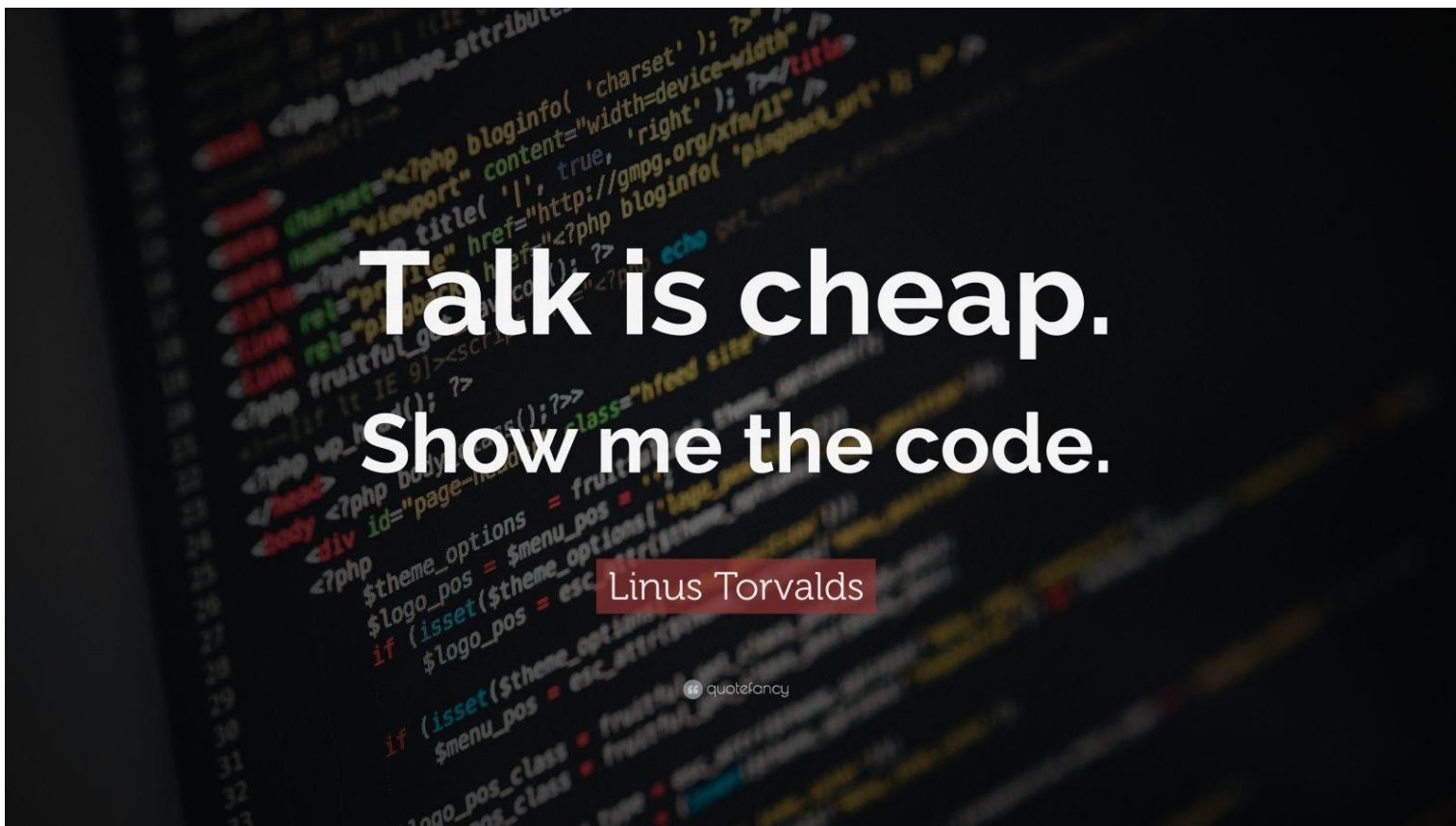
---

- 机器学习基本概念与流程
- k-近邻算法（kNN）
- 线性回归（Linear Regression）
- 逻辑回归（Logistic Regression）及Softmax回归
- 正则化
- 实战案例3-1：贷款违约预测 (1)

## 实战案例 3-1

# 项目名称：贷款违约预测 (1)

- 请参考相应的配套代码及案例讲解文档



# 问答互动

在所报课的课程页面，

- 1、点击“全部问题”显示本课程所有学员提问的问题。
- 2、点击“提问”即可向该课程的老师 and 助教提问问题。



# 联系我们

---

小象学院：互联网新技术在线教育领航者

— 微信公众号：**小象学院**

