线性回归与逻辑回归

Machine Learning Engineer

机器学习工程师

讲师：寒小阳

目 录

CONTENTS

01 线性模型、线性回归与广义线性模型

02 逻辑回归

03 工程应用经验

04 数据案例讲解

01

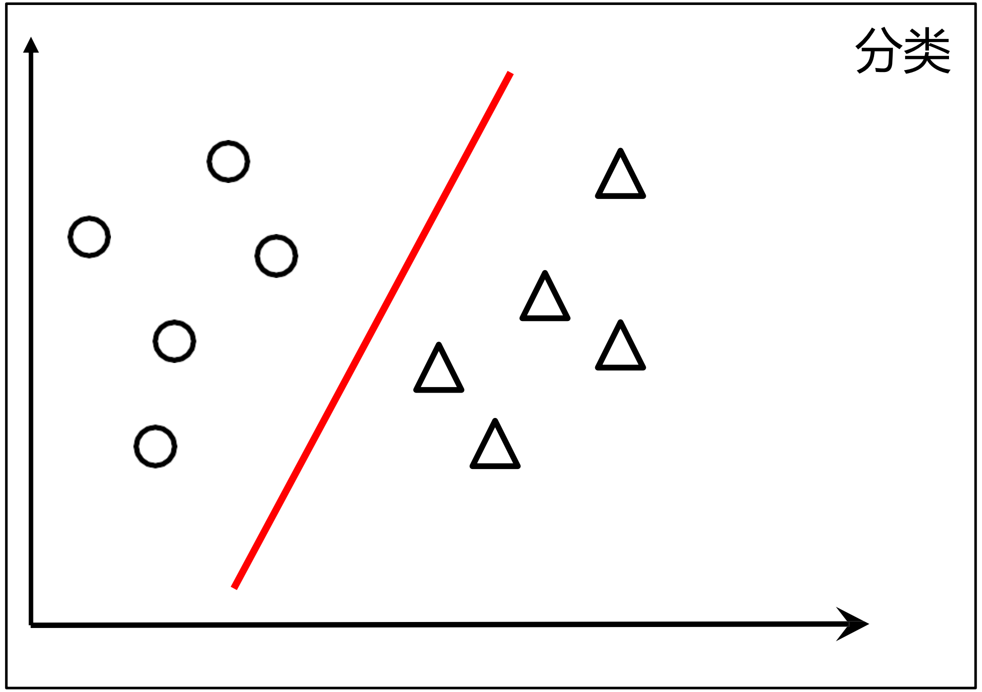
线性模型、线性回归与广义线性回归

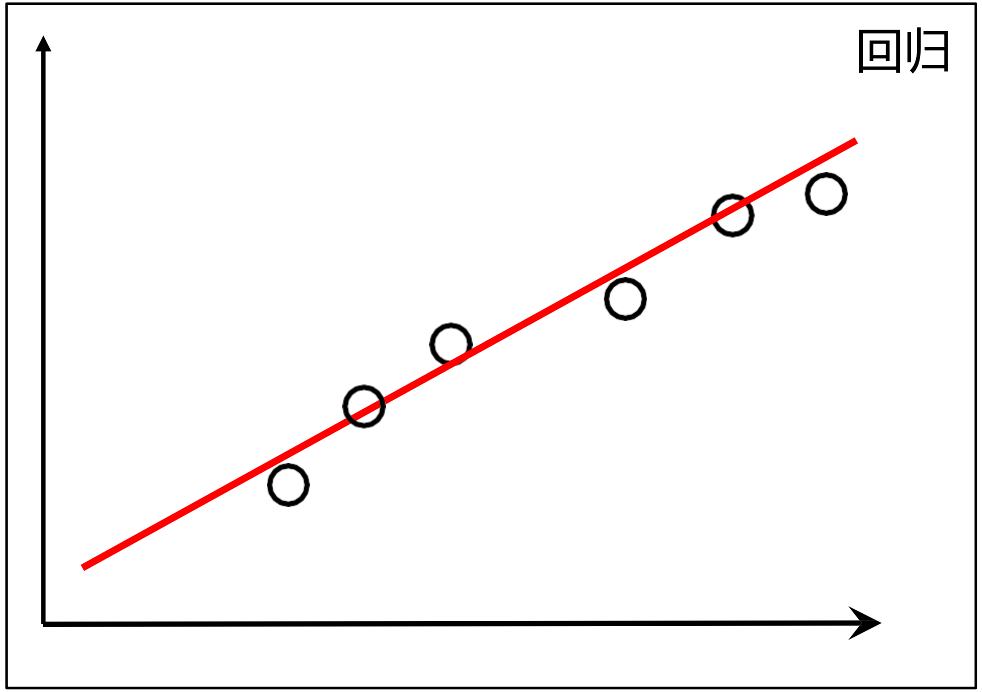
1.1 线性模型

1.2 线性回归

1.3 广义线性模型

1.1 线性模型



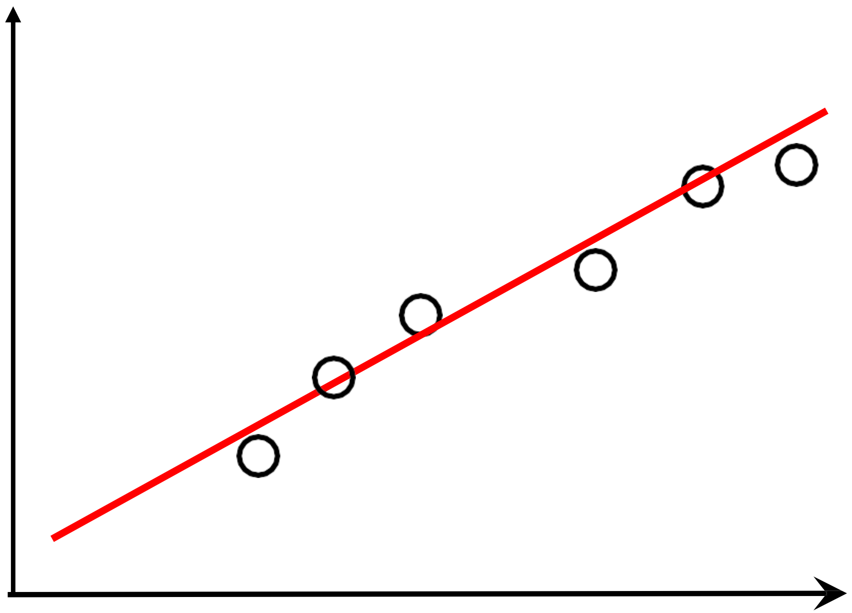


线性模型(linear model)试图学得一个通过属性的线性组合来进行预测的函数 $f(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d + b$

向量形式：$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b$

简单、基本、可解释性好

1.2 线性回归

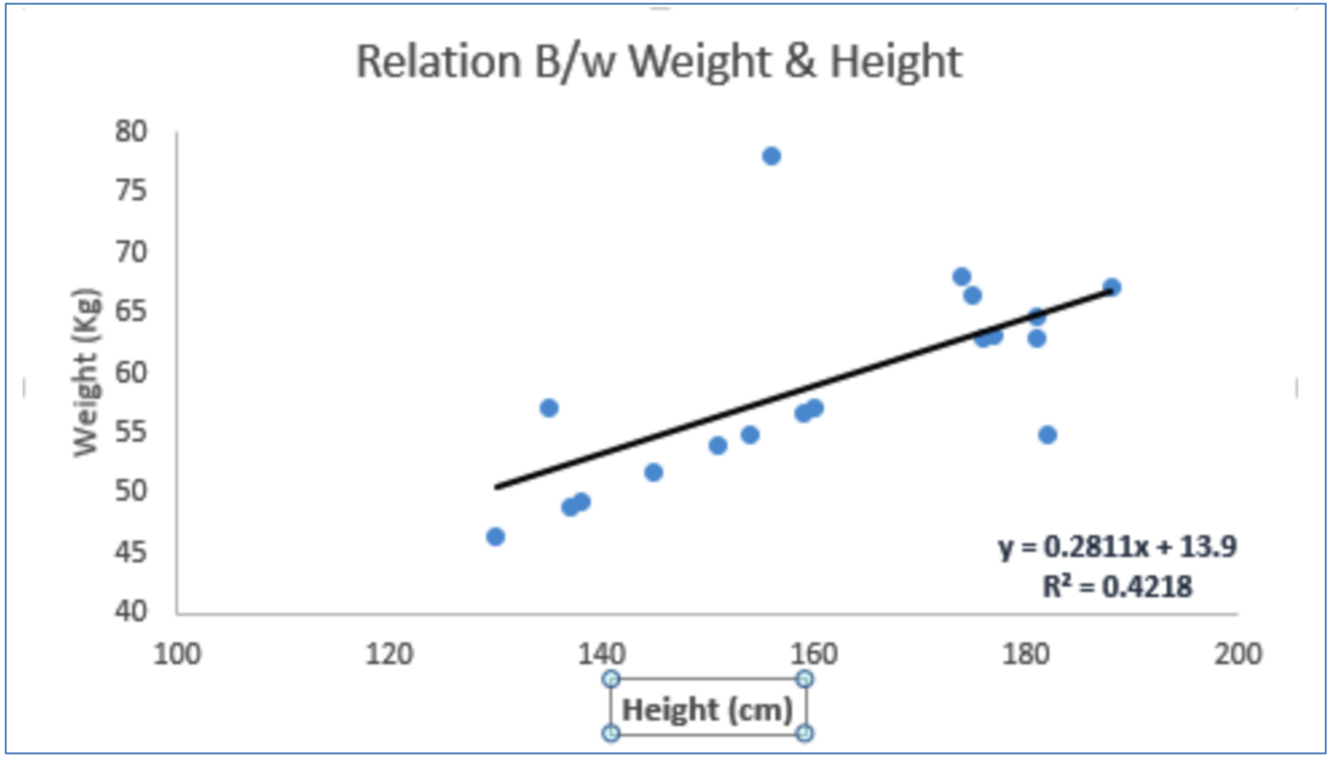


- 有监督学习→学习样本为 $\mathcal{D}=\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$
- 输出/预测的结果y_i为连续值变量
- 需要学习映射
- 假定输入x和输出y之间有线性相关关系 $f:\mathcal{X}\rightarrow\mathcal{Y}$

线性回归

- 一个简单的例子

让一个六年级的孩子在不问同学具体体重多少的情况下，把班上同学按照体重从轻到重排队。这个孩子会怎么做呢？

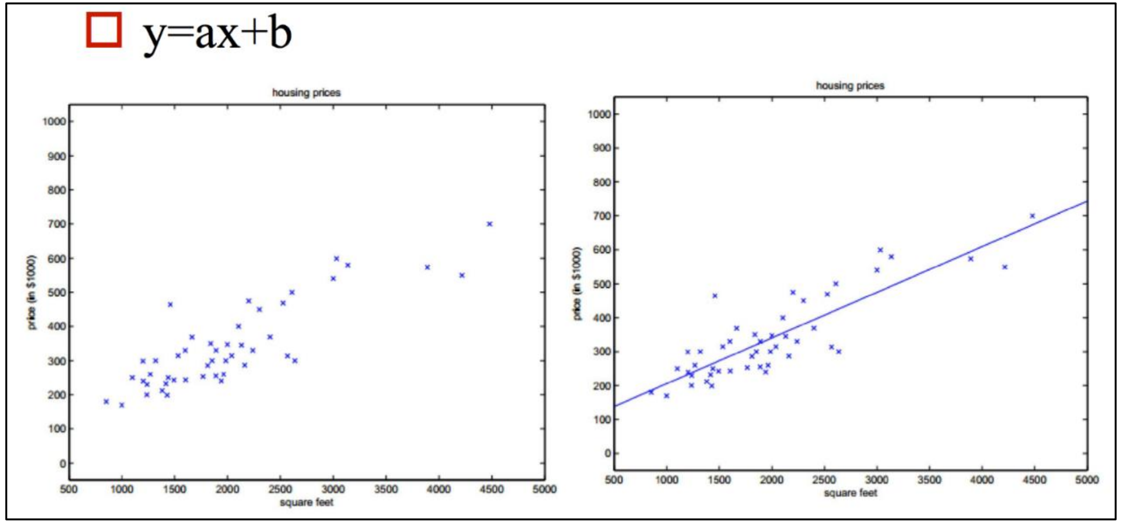


他有可能会通过观察大家的身高和体格来排队。

1.2 线性回归

- 房价预测例子（一元）

面积 (x，平方英尺)	**价格** (y，千美元)
2104	460
1416	232
1534	315
852	178
…	…



- 房价预测例子（多元）

训练集

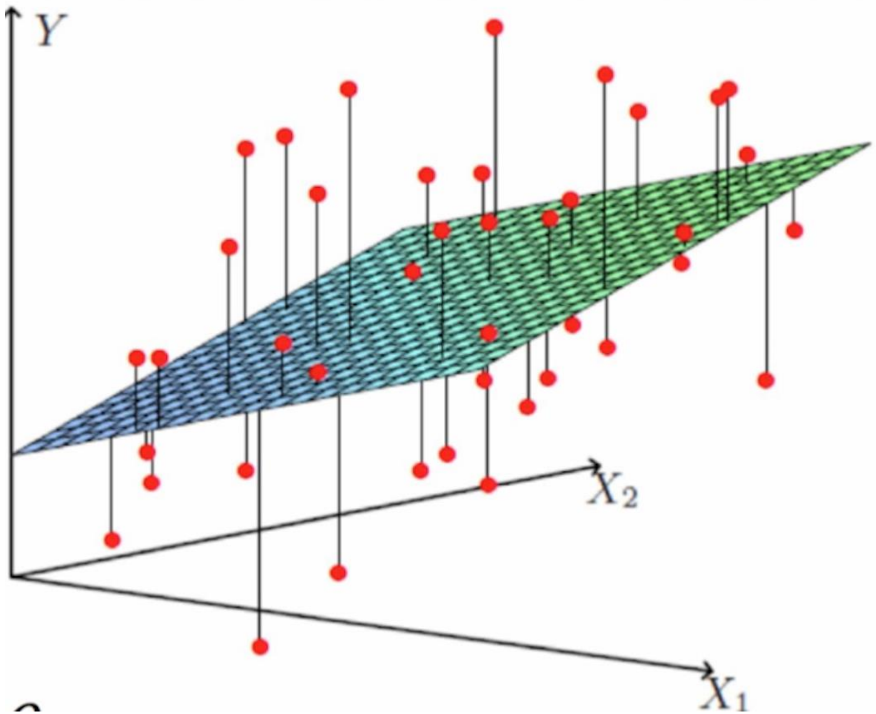
面积 (x1，平方英尺)	卧室个数 (x2，个)	楼层 (x3，层)	房龄 (x4，年)	…	价格 (y，千美元)
2104	5	1	45	…	460
1416	3	2	40	…	232
1534	3	2	30	…	315
852	2	1	36	…	178
…	…	…	…	…	…

测试集

面积 (x1，平方英尺)	卧室个数 (x2，个)	楼层 (x3，层)	房龄 (x4，年)	…	价格 (y，千美元)
1500	3	2	3		?

线性回归

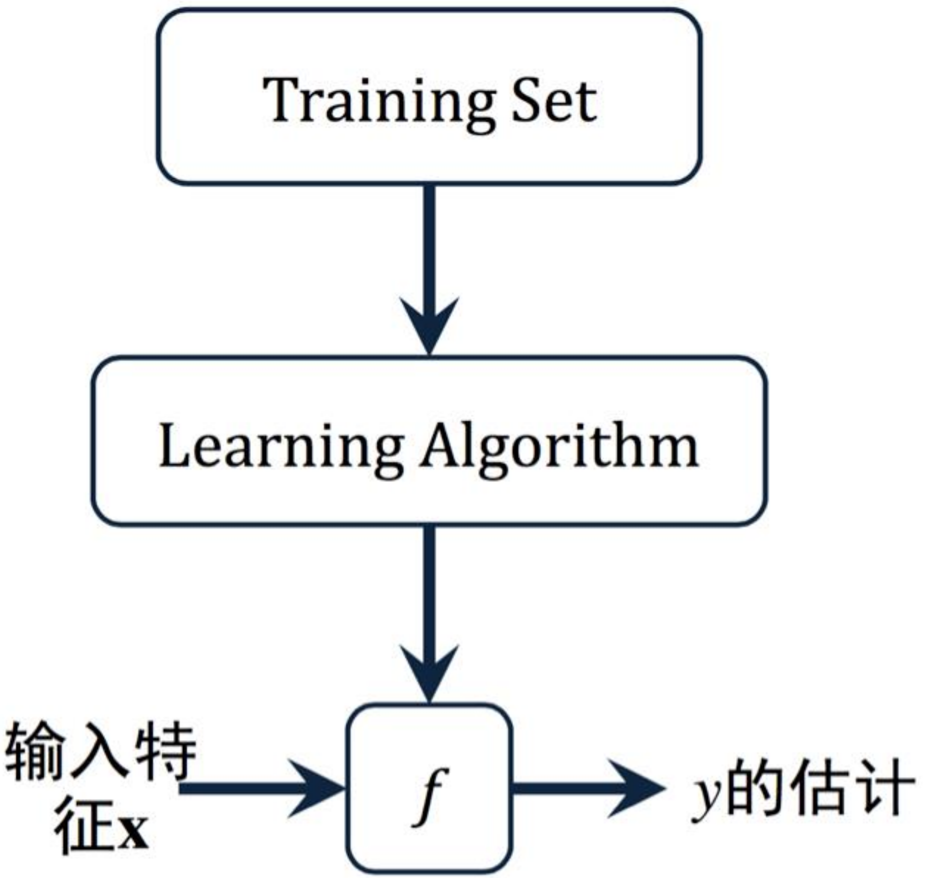
- 房价预测例子（多元）



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

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

线性回归



问题：如何表示f？

线性回归：假设函数f为输入$\mathbf{x}$的线性函数：

$$f(\mathbf{x}) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

写成向量形式（在特征$\mathbf{x}$中增加一维$x_0 = 1$，表示截距项）：

$$f(\mathbf{x}) = \theta^T \mathbf{x}$$

1.2 线性回归

- 损失函数（loss function）

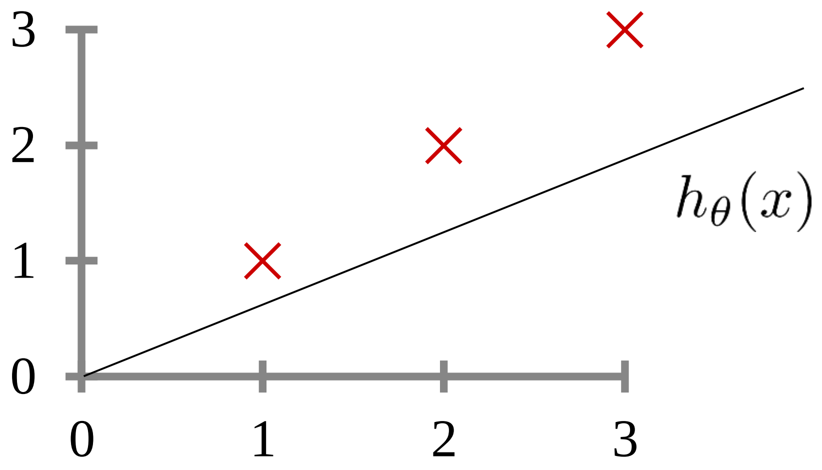
我们希望找到最好的权重/参数$\theta = [\theta_0, \theta_1, \ldots, \theta_n]$

如何衡量“最好”？

我们把x到y的映射函数f记作 θ 的函数 $h_\theta(x)$

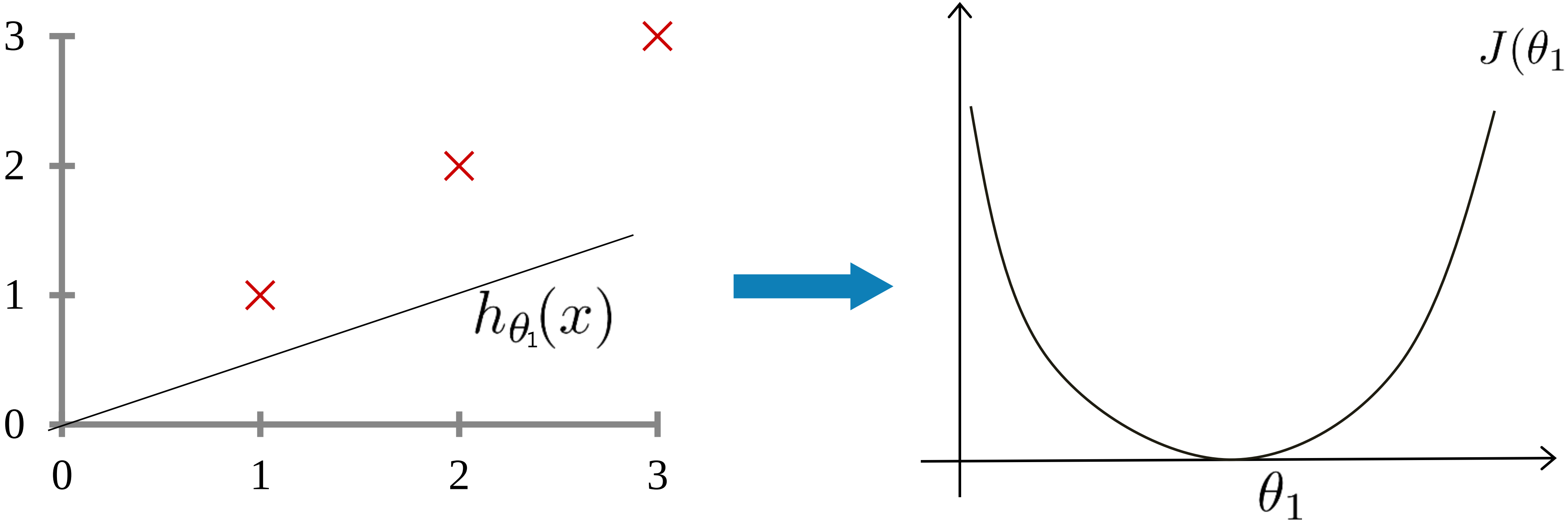
定义损失函数为：

$$J(\theta_0, \theta_1, \ldots, \theta_n) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$



- 最小化损失函数

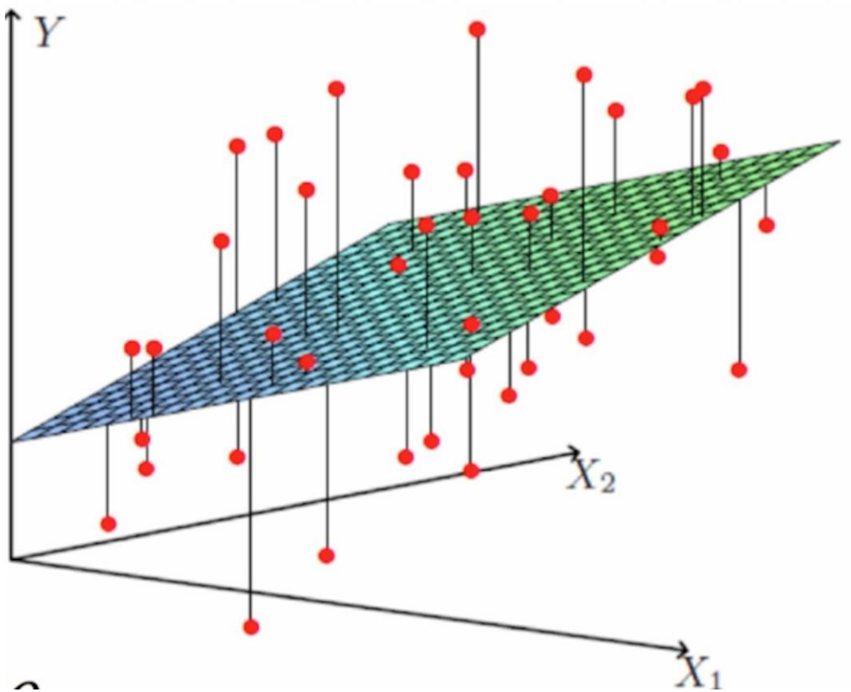
  均方误差损失是一个凸函数

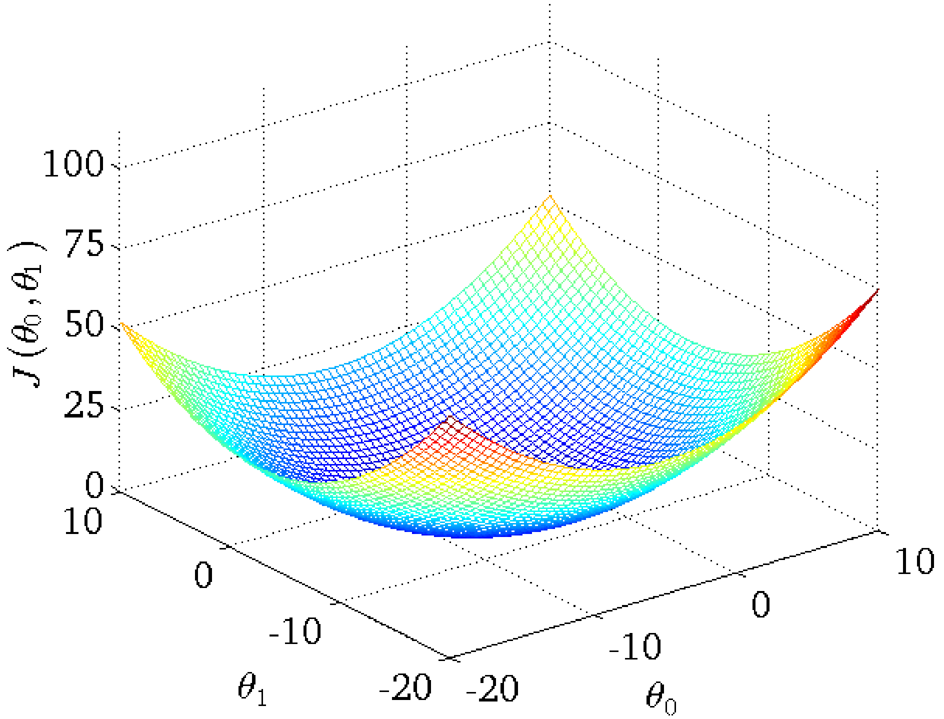


1.2 线性回归

- 最小化损失函数

  均方误差损失是一个凸函数

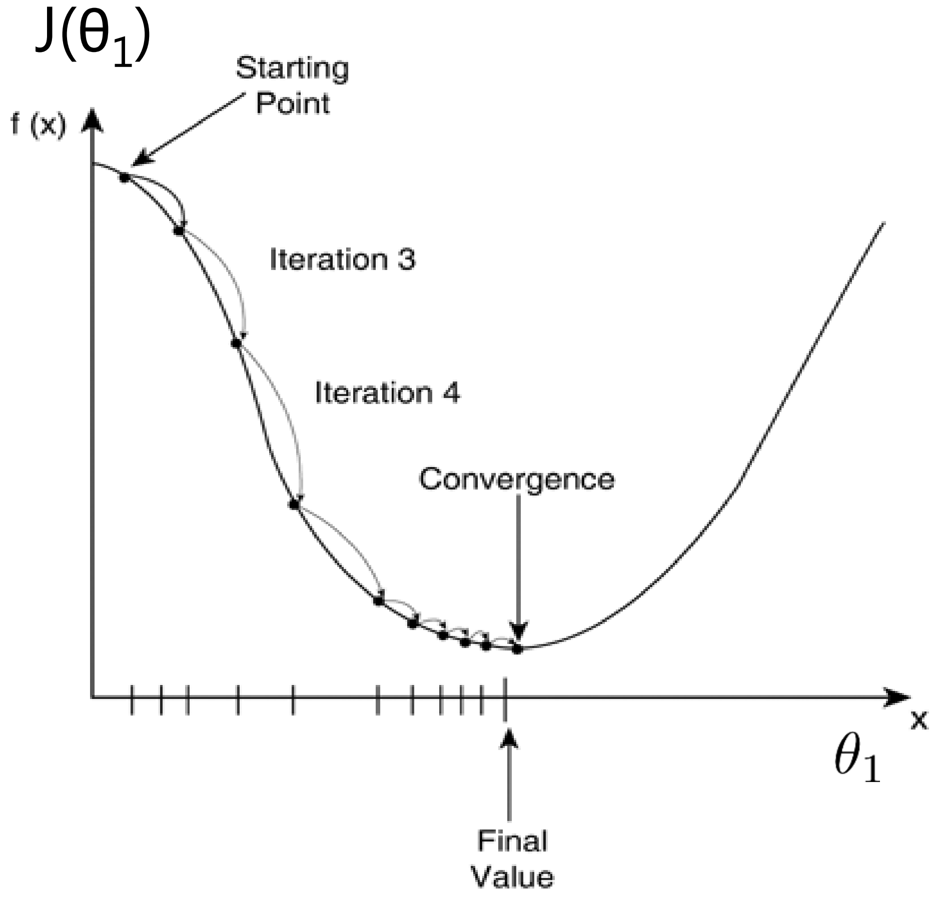




线性回归

- 梯度下降

  逐步迭代减小损失函数(凸函数)
  如同下山，找准方向(斜率)，每次迈进一小步，直至山底



一元的损失函数

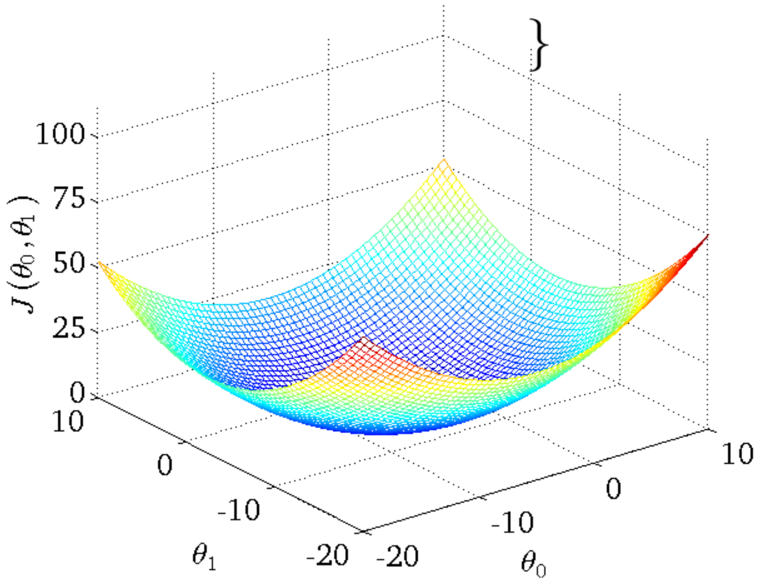
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

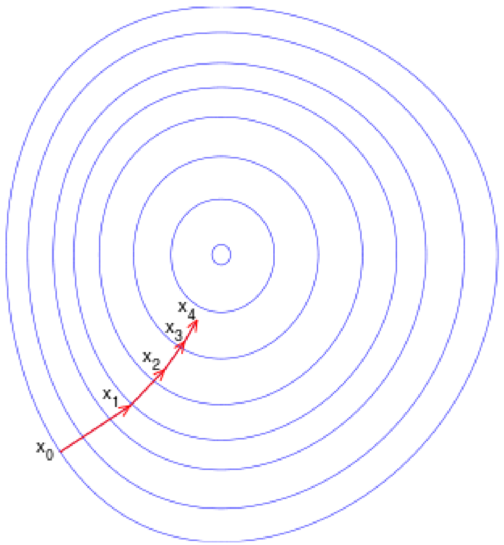
1.2 线性回归

- 梯度下降

二元的损失函数

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&\quad \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)} \\
&\}
\end{aligned}
$$



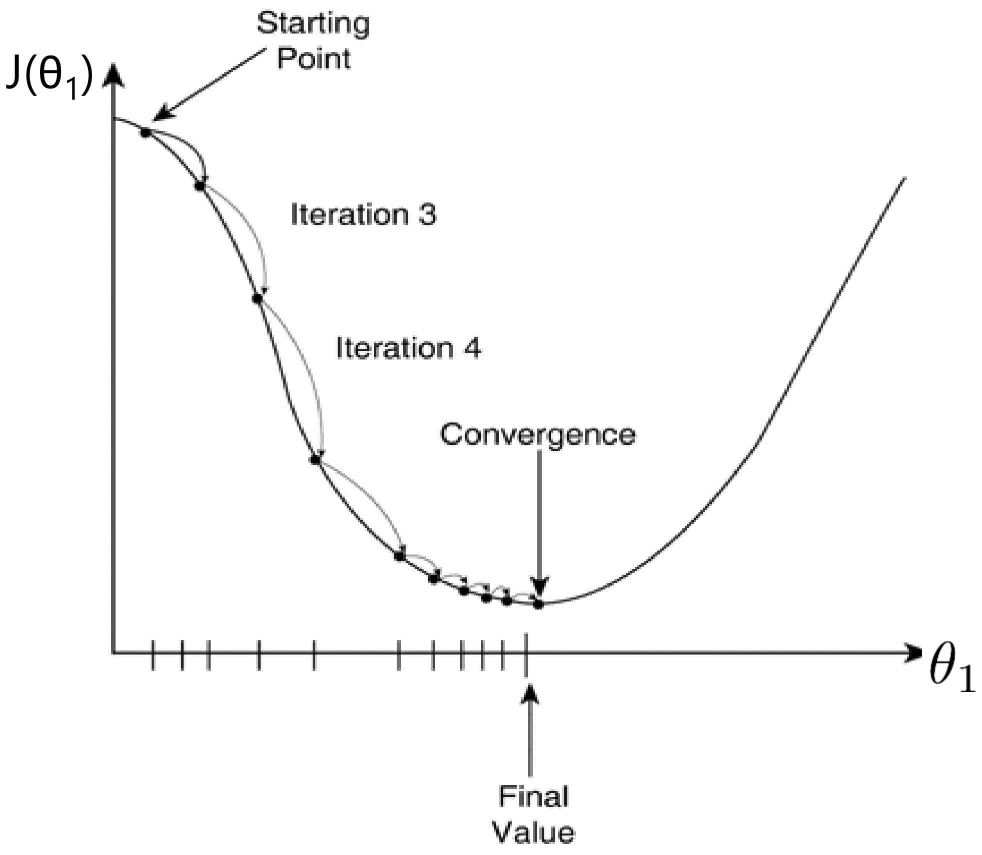


- 梯度下降学习率的影响

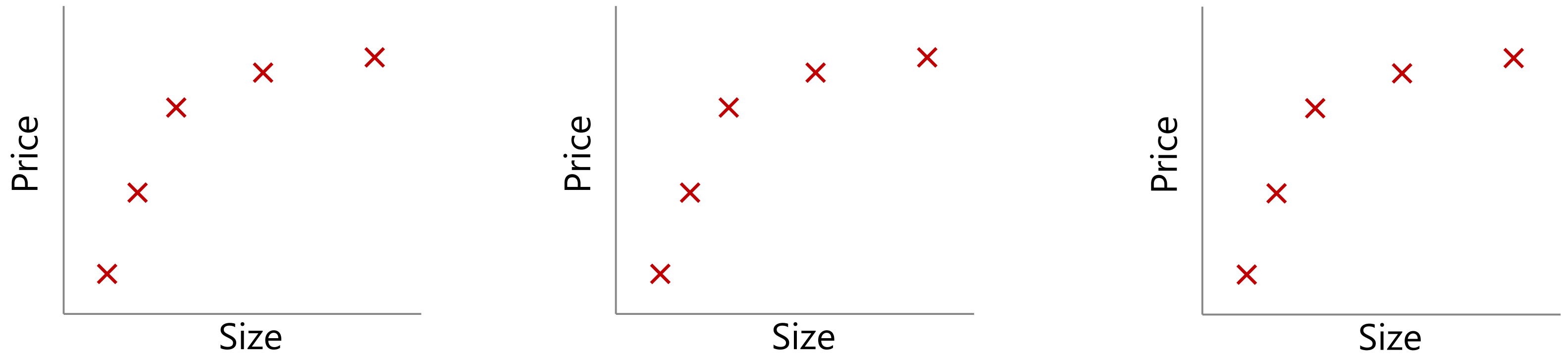
太小收敛速度太慢
太大会震荡甚至不收敛

一元的损失函数

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



- **欠拟合与过拟合**(以多项式回归为例)



$\theta_0 + \theta_1 x$ $\qquad$ $\theta_0 + \theta_1 x + \theta_2 x^2$ $\qquad$ $\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

<u>欠拟合</u>：模型没有很好地捕捉到数据特征，不能够很好地拟合数据
<u>过拟合</u>：把样本中的一些噪声特性也学习下来了，泛化能力差

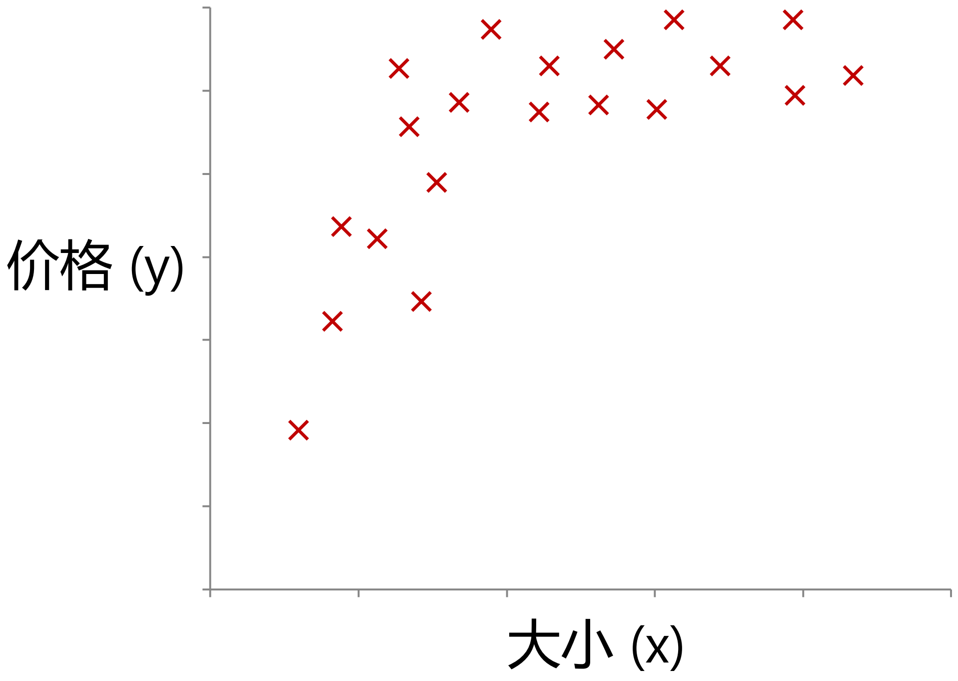
实际工业界使用的各种模型都存在过拟合的风险：

- 更多的参数/特征，更复杂的模型，通常有更强的学习能力，但是更容易“失去控制”
- 训练集中有一些噪声，并不代表全量真实数据的分布，死记硬背会丧失泛化能力

线性回归

- 过拟合与正则化

通知正则化添加参数“惩罚”，控制参数幅度
限制参数搜索空间，减小过拟合风险



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

$$\Downarrow$$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

广义线性模型

对于样本 $(\boldsymbol{x}, y)$，$y \in \mathbb{R}$，如果我们希望用线性的映射关系去逼近y值

可以得到线性回归模型 $y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

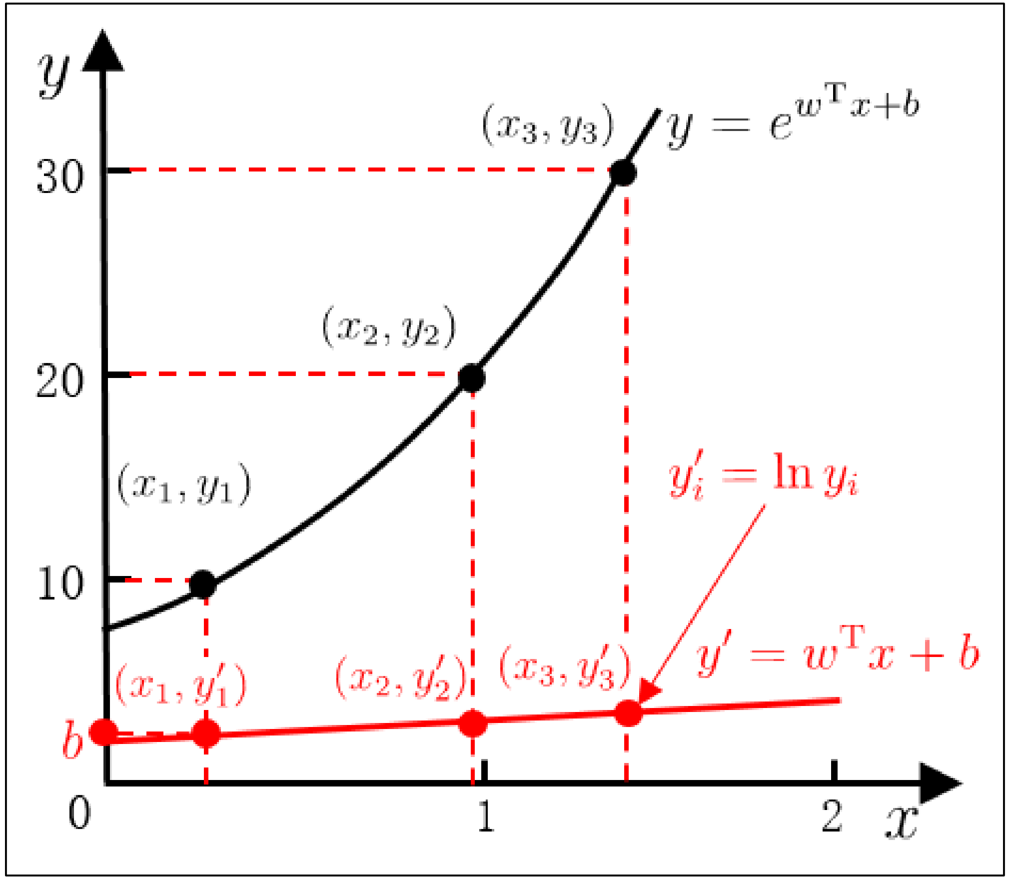
有时候关系不一定是线性的
如何逼近y 的衍生物？

比如令 $\ln y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

则得到对数线性回归

(log-linear regression)

实际是在用 $e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}$ 逼近y



要点总结

- 线性回归
  - 线性映射关系
    - $\hat{y} = \theta^T X$
  - 损失函数
    - MSE：评估与标准答案之间的差距
  - 梯度下降
    - 沿着损失函数梯度方向逐步修正参数
    - 学习率影响
  - 模型状态
    - 欠拟合
    - 过拟合
- 广义线性回归
  - 对线性映射的结果进行数学变换，去逼近y值
    - 指数(exp)或者对数(log)变换处理

02

逻辑回归

2.1 从线性回归到逻辑回归

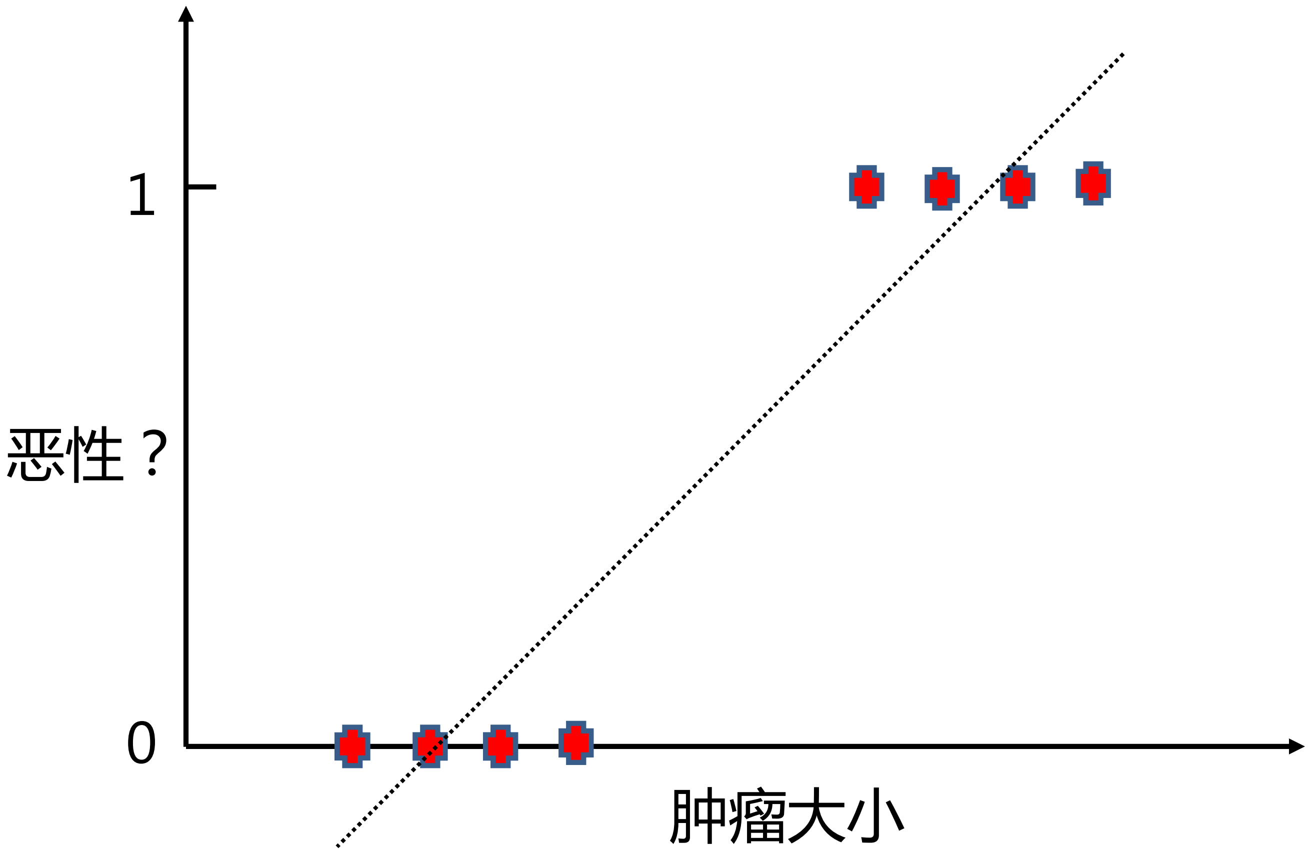
2.2 逻辑回归决策边界

2.3 逻辑回归损失函数

2.4 从二分类到多分类

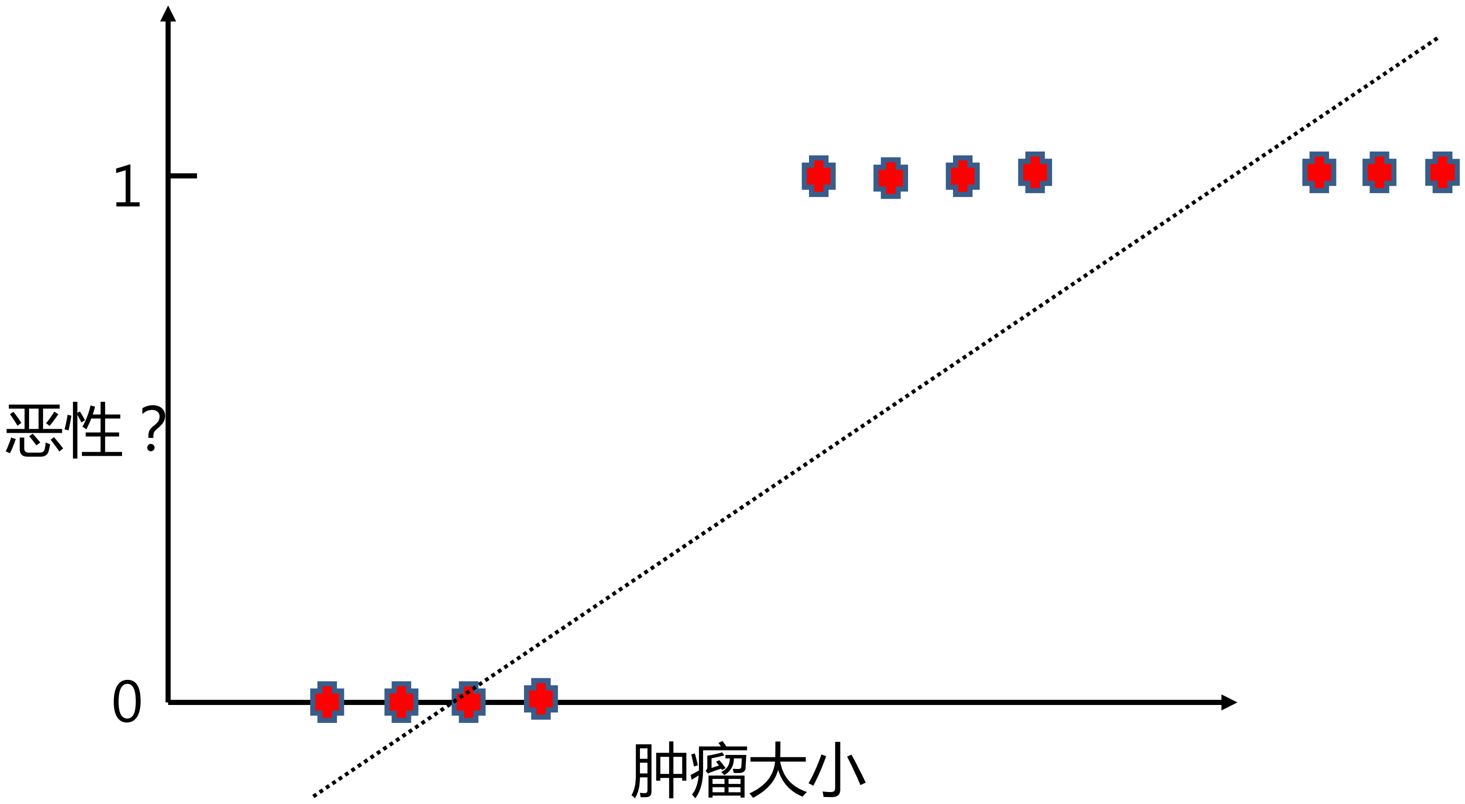
2.1 从线性回归到逻辑回归

分类问题可以通过 **线性回归+阈值** 去解决吗？



从线性回归到逻辑回归

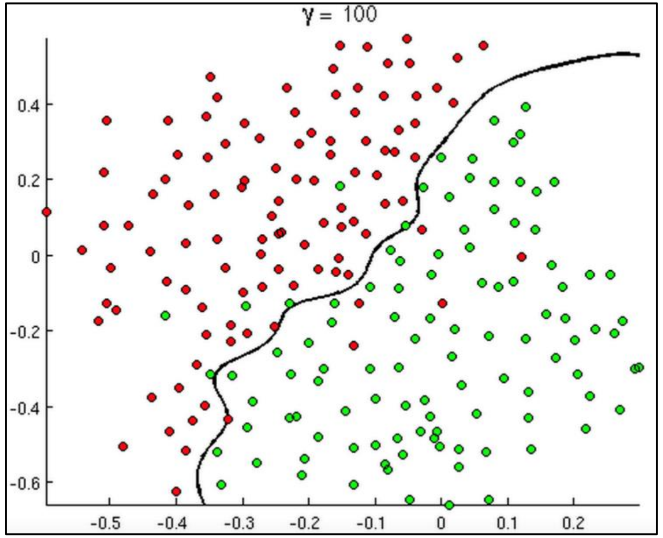
分类问题在有噪声点的情况下，阈值偏移大，健壮性不够



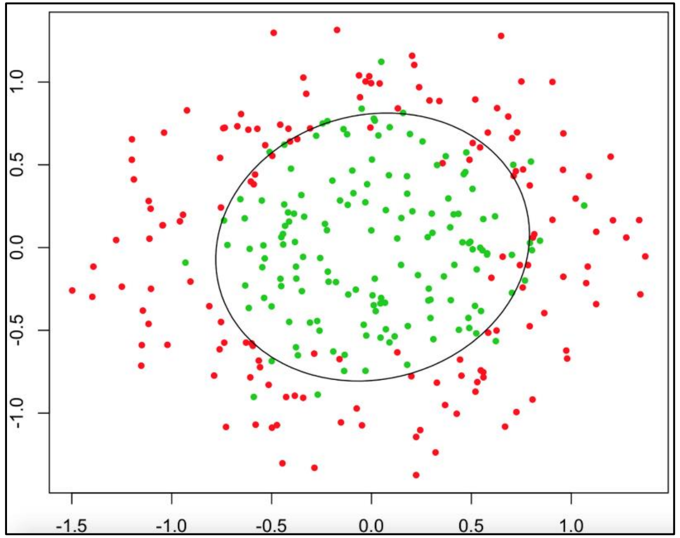
逻辑回归决策边界

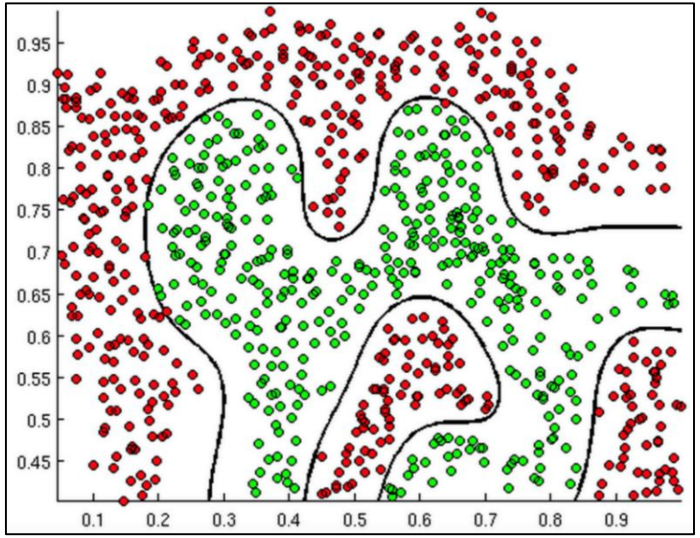
在逻辑回归(Logistic Regression)里，通常我们并不拟合样本分布，而是确定决策边界

下面为各式各样的决策边界



线性决策边界

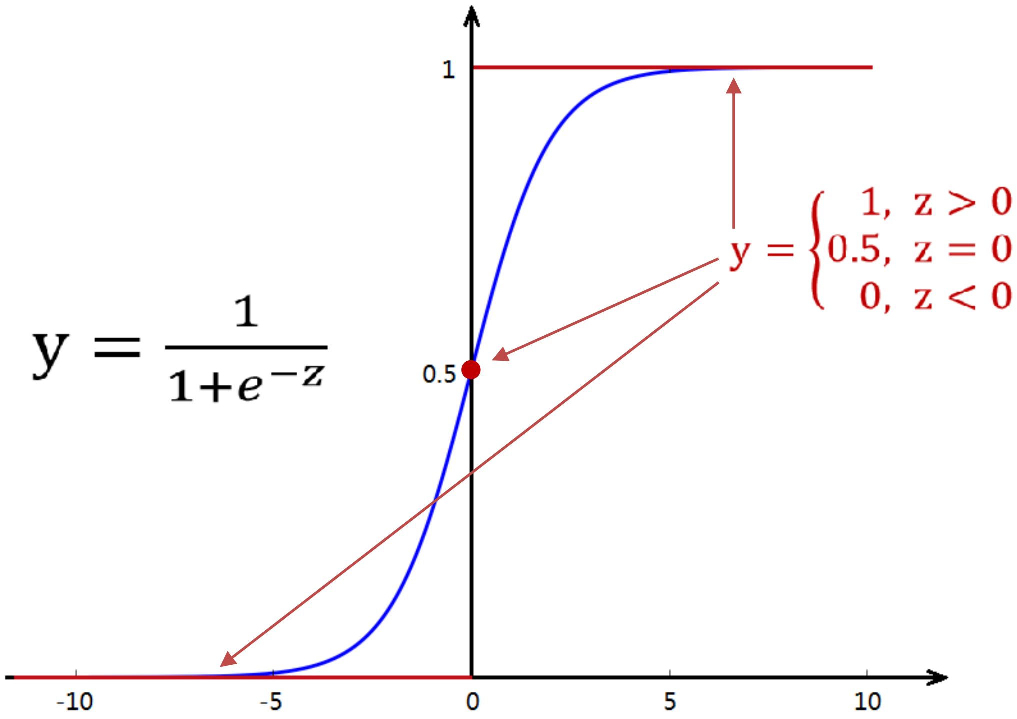




非线性决策边界

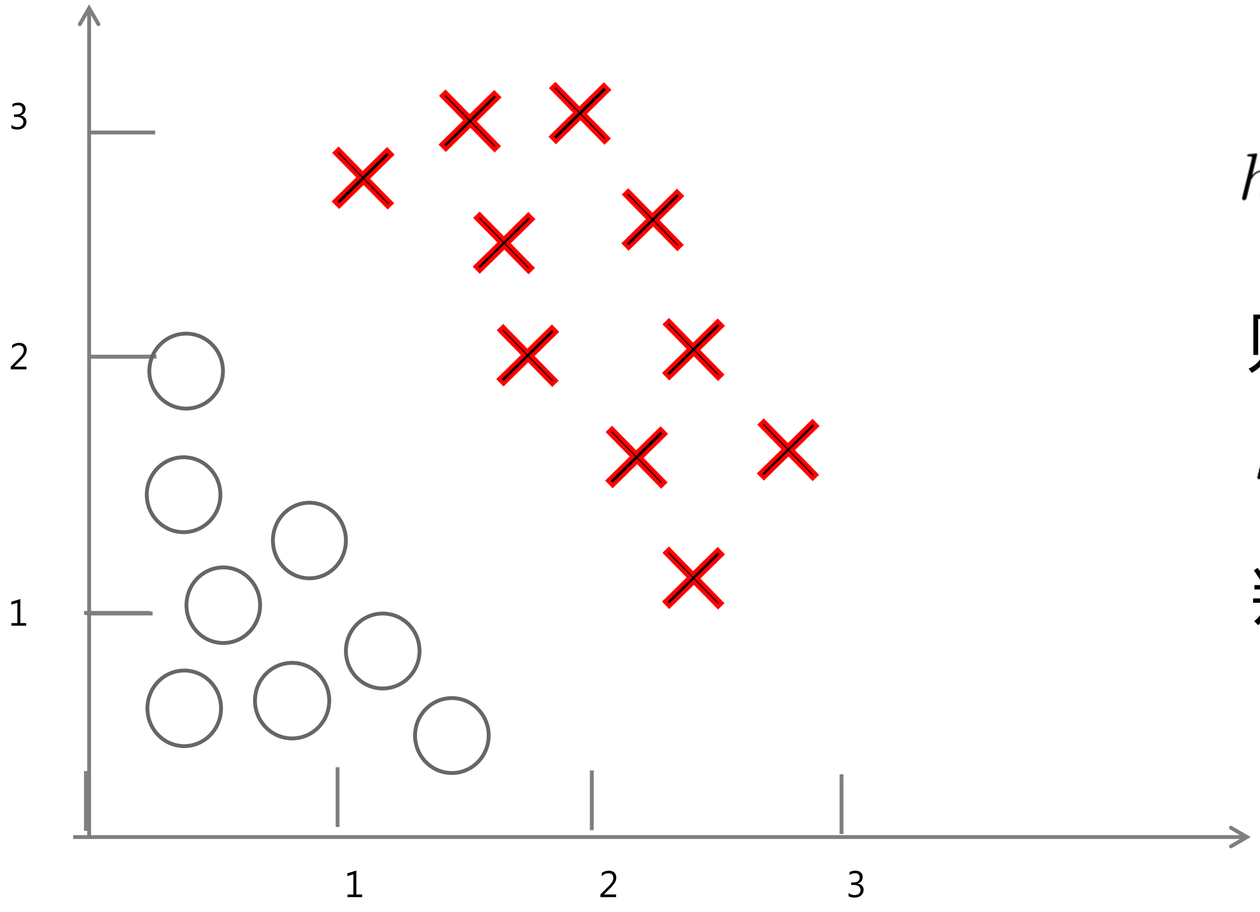
逻辑回归决策边界

$$y = \frac{1}{1+e^{-z}}$$



逻辑回归决策边界

- 线性决策边界



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

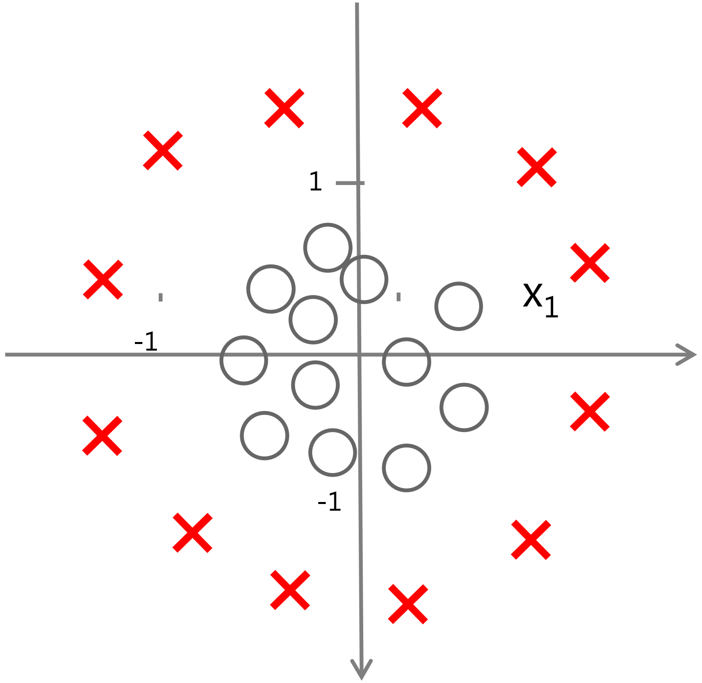
则 $-3 + x_1 + x_2 \geq 0$ 时

$h_\theta(x)$ 结果如何？

判定结果如何？

2.2 逻辑回归决策边界

- 非线性决策边界



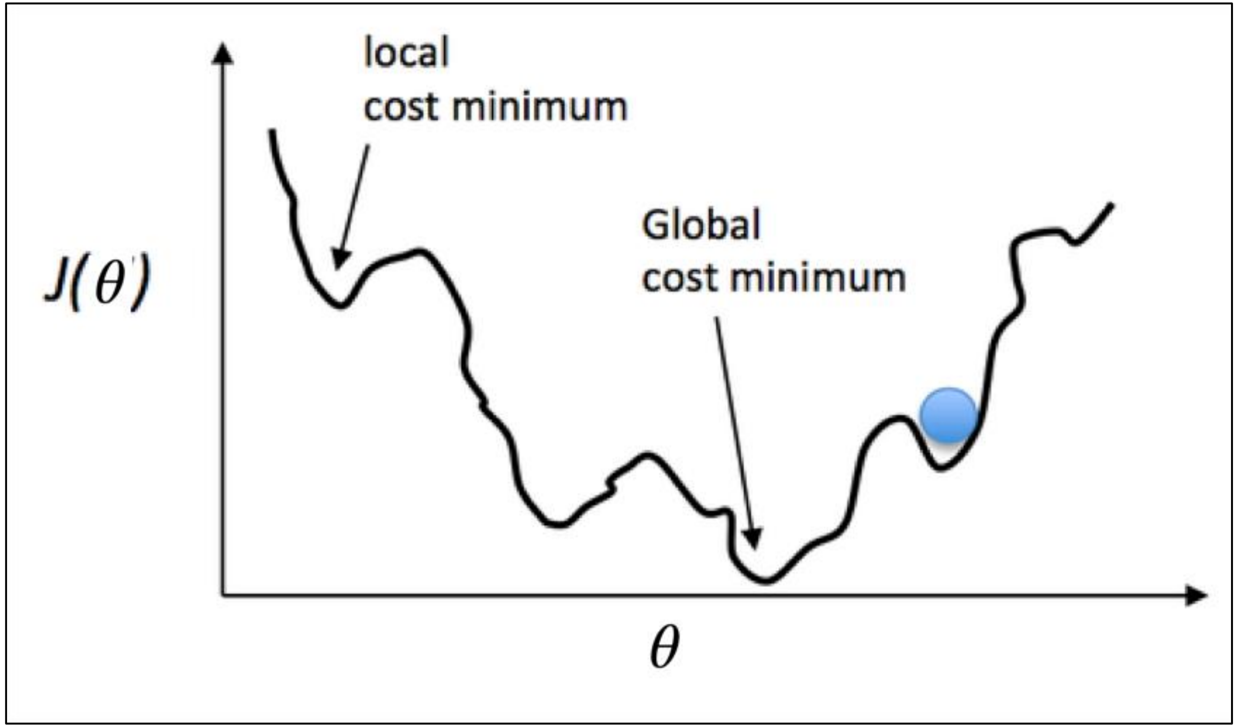
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^2 x_2^2 + \theta_6 x_1^3 x_2 + \ldots)$$

逻辑回归损失函数

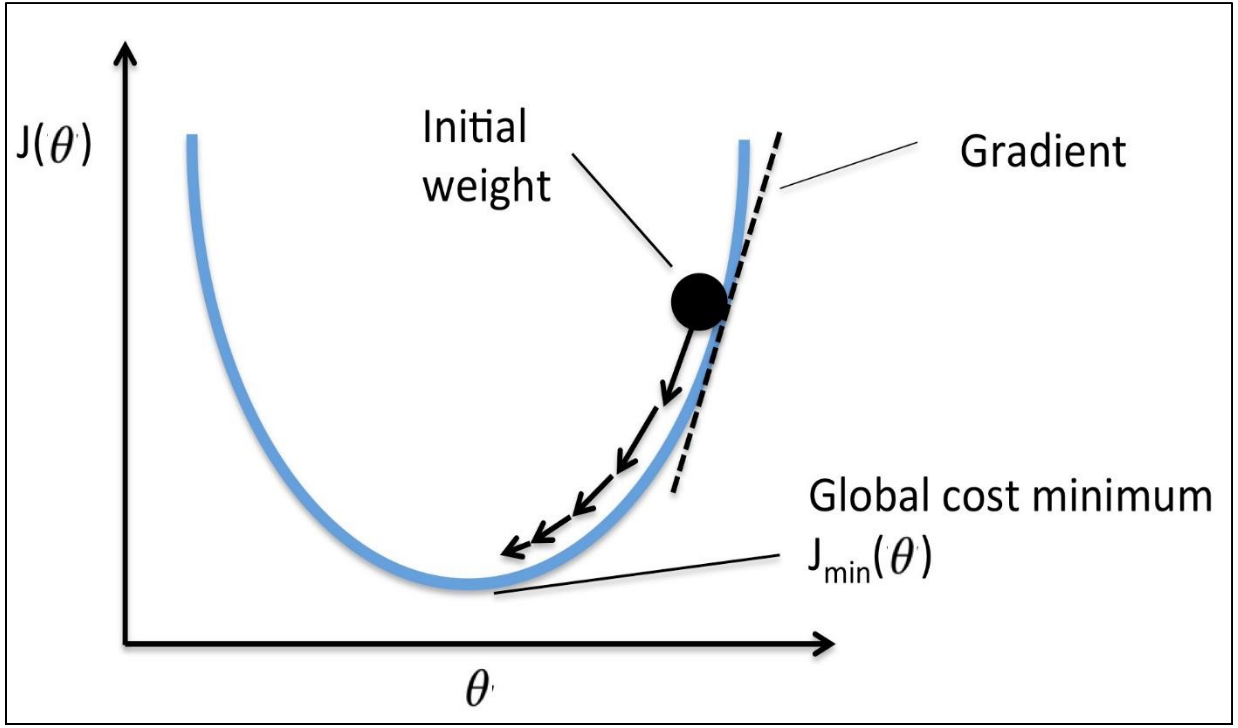
- 均方差损失(MSE)？

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



逻辑回归损失函数

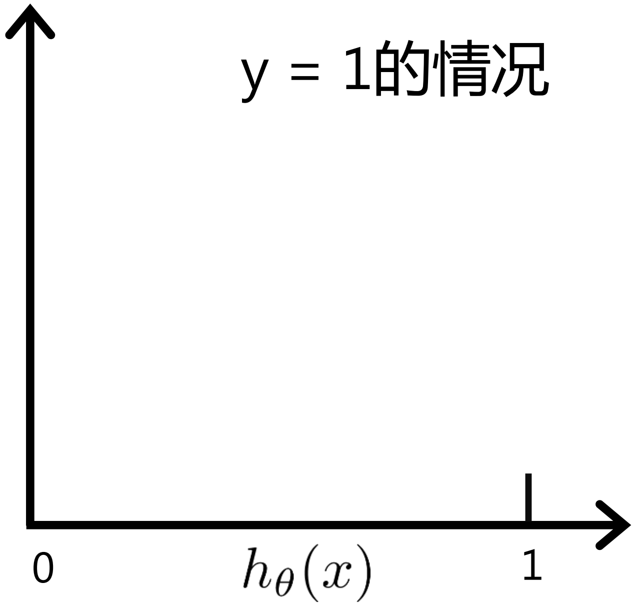
我们希望损失函数是凸函数

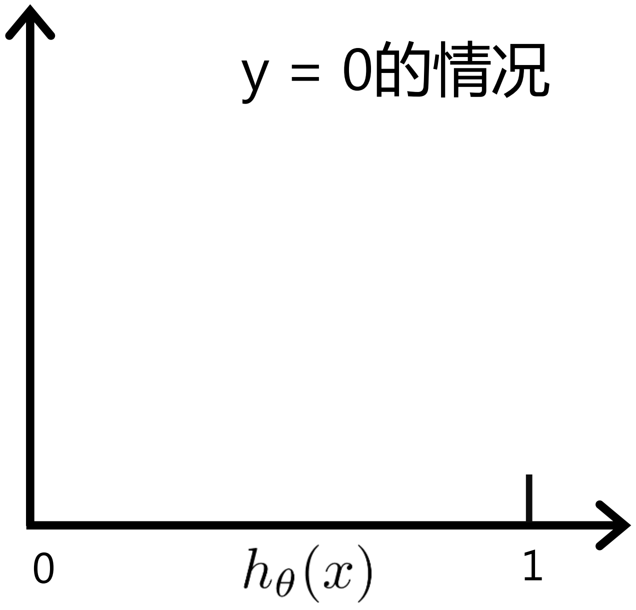


2.3 逻辑回归损失函数

- 对数损失/二元交叉熵损失

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$





2.3 逻辑回归损失函数

- 损失函数与正则化

依旧存在过拟合问题，决策边界可能“抖动很厉害”！

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

$$\Big\Downarrow \text{添加正则化项}$$

$$J(\theta)=\left[-\frac{1}{m}\sum_{i=1}^{m}y^{(i)}\log(h_\theta(x^{(i)})+(1-y^{(i)})\log 1-h_\theta(x^{(i)})\right]+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

2.3 逻辑回归损失函数

- 如何最小化损失函数？

对于凸函数，依旧可以用梯度下降！

$$J(\theta)=\left[-\frac{1}{m}\sum_{i=1}^{m}y^{(i)}\log\left(h_\theta(x^{(i)})+(1-y^{(i)})\log 1-h_\theta(x^{(i)})\right]+\frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$$

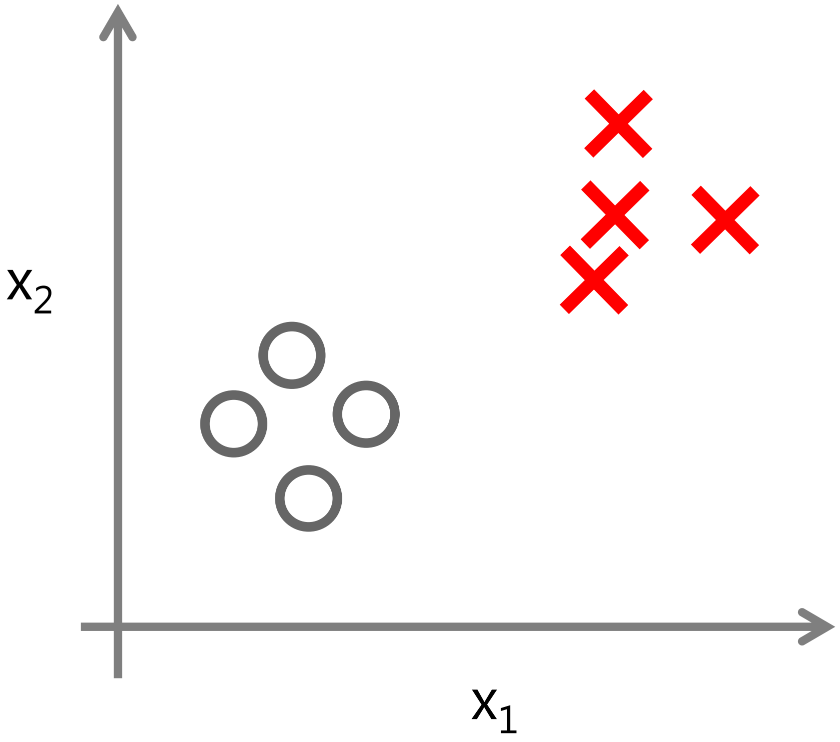
$$\downarrow$$

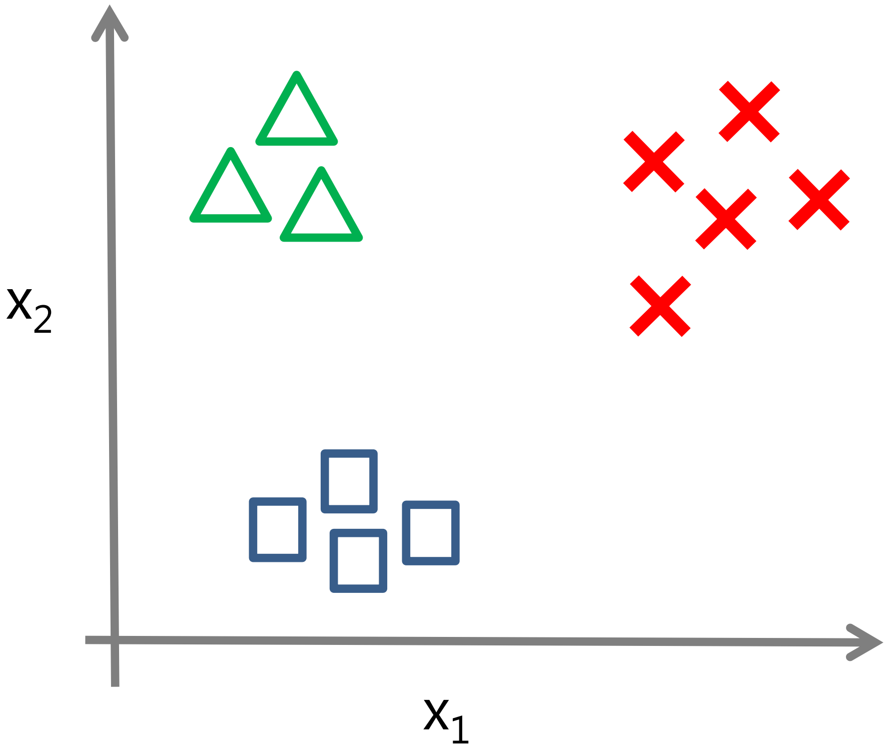
$$\theta_j := \theta_j - \alpha\frac{\partial}{\partial\theta_j}J(\theta)$$

2.4 从二分类到多分类

- 多分类

  我们已经知道二分类问题如何处理了，那么多分类呢？





要点总结

- 逻辑回归
  - 线性回归+阈值
    - 解决分类问题鲁棒性不OK
  - Sigmoid函数与决策边界
    - Sigmoid函数：压缩至0-1之间
    - 根据阈值，产生对应的决策边界
  - 损失函数
    - 最大似然到对数损失
  - 梯度下降
    - 沿着损失函数梯度方向逐步修正参数
  - 二分类到多分类
    - one vs one
    - one vs rest

03 工程应用经验

3.1 逻辑回归 VS 其他模型

3.2 样本处理

3.3 工具包与库

3.1 逻辑回归 VS 其他模型

LR 弱于 SVM/GBDT/RandomForest... ？

- 模型本身并没有好坏之分
  1. LR能以概率的形式输出结果，而非只是0,1判定
  2. LR的可解释性强，可控度高
  3. 训练快，特征工程(feature engineering)之后效果赞
  4. 因为结果是概率，可以做排序模型
  5. 添加特征非常简单...
- 应用
  1. CTR预估/推荐系统的learning to rank/各种分类场景
  2. 很多搜索引擎厂的广告CTR预估基线版是LR
  3. 电商搜索排序/广告CTR预估基线版是LR
  4. 新闻app的推荐和排序基线也是LR

3.2 样本处理

- 样本特征处理
  离散化后用独热向量编码(one-hot encoding)处理成0,1值
  LR训练连续值，注意做幅度缩放(scaling)
- 处理大样本量
  试试spark MLlib
  试试采样(注意是否需要分层采样)
- 注意样本的平衡
  对样本分布敏感
  欠采样，过采样
  修改损失函数，给不同权重

Liblinear

https://www.csie.ntu.edu.tw/~cjlin/liblinear/

LIBLINEAR -- A Library for Large Linear Classification

Machine Learning Group at National Taiwan University
Contributors

Introduction

LIBLINEAR is a linear classifier for data with millions of instances and features. It supports

- L2-regularized classifiers
  L2-loss linear SVM, L1-loss linear SVM, and logistic regression (LR)
- L1-regularized classifiers (after version 1.4)
  L2-loss linear SVM and logistic regression (LR)
- L2-regularized support vector regression (after version 1.9)
  L2-loss linear SVR and L1-loss linear SVR.

Main features of LIBLINEAR include

- Same data format as LIBSVM, our general-purpose SVM solver, and also similar usage
- Multi-class classification: 1) one-vs-the rest, 2) Crammer & Singer
- Cross validation for model evaulation
- Automatic parameter selection
- Probability estimates (logistic regression only)
- Weights for unbalanced data
- MATLAB/Octave, Java, Python, Ruby interfaces

3.3 工具包与库

Spark Mllib

Spark 2.2.0 Overview Programming Guides API Docs Deploying More

Scala Java **Python**

The following example shows how to load a sample dataset, build Logistic Regression model, and make predictions with the resulting model to compute the training error.

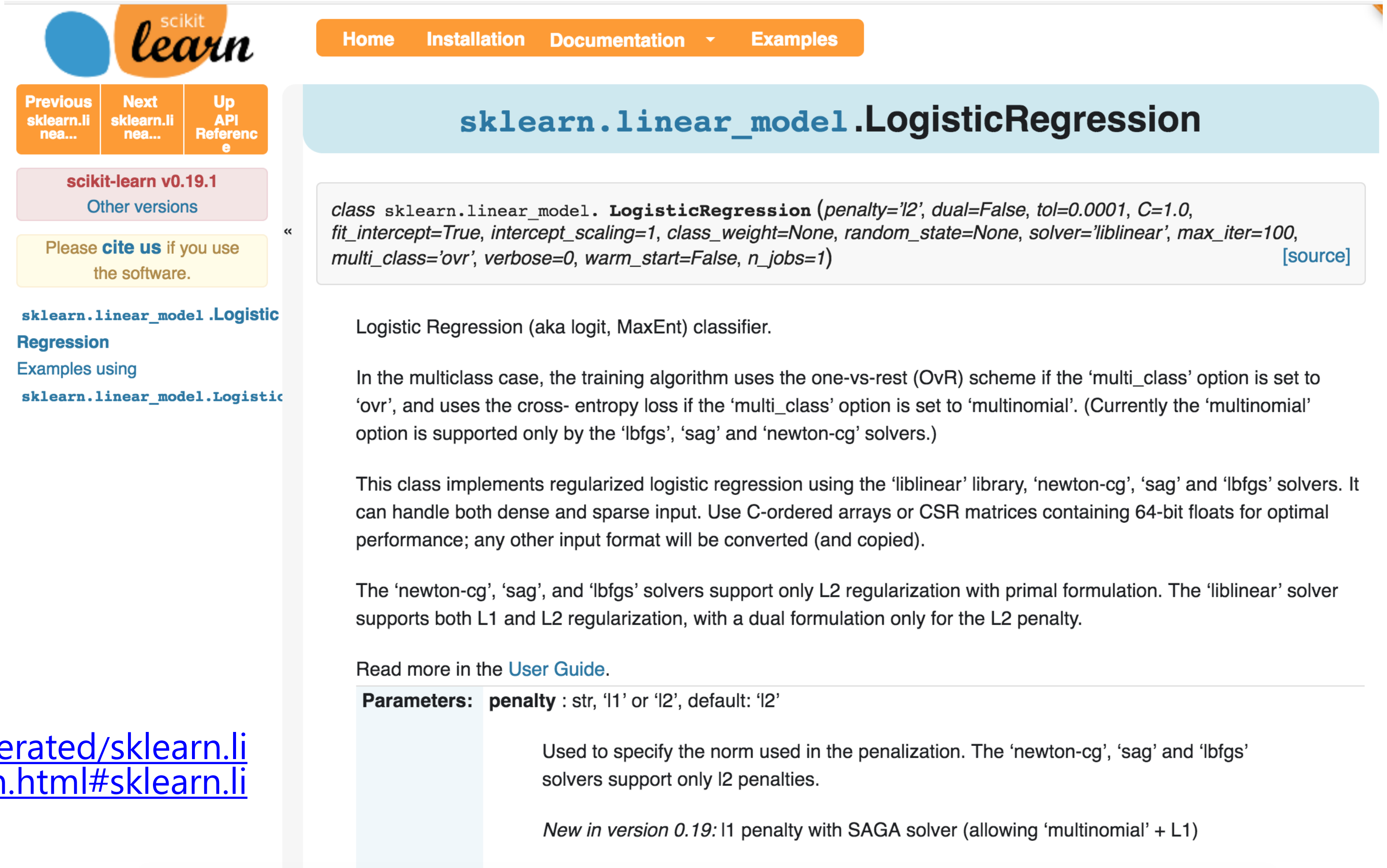
Note that the Python API does not yet support multiclass classification and model save/load but will in the future.

Refer to the `LogisticRegressionWithLBFGS` Python docs and `LogisticRegressionModel` Python docs for more details on the API.

```
from pyspark.mllib.classification import LogisticRegressionWithLBFGS, LogisticRegressionModel
from pyspark.mllib.regression import LabeledPoint

# Load and parse the data
def parsePoint(line):
    values = [float(x) for x in line.split(' ')]
    return LabeledPoint(values[0], values[1:])

data = sc.textFile("data/mllib/sample_svm_data.txt")
parsedData = data.map(parsePoint)

# Build the model
model = LogisticRegressionWithLBFGS.train(parsedData)

# Evaluating the model on training data
labelsAndPreds = parsedData.map(lambda p: (p.label, model.predict(p.features)))
trainErr = labelsAndPreds.filter(lambda lp: lp[0] != lp[1]).count() / float(parsedData.count())
```

http://spark.apache.org/docs/latest/mllib-linear-methods.html#logistic-regression

Scikit-learn



scikit learn

Home | Installation | Documentation | Examples

Previous sklearn.linear_model... | Next sklearn.linear_model... | Up API Reference

scikit-learn v0.19.1
Other versions

Please **cite us** if you use the software.

sklearn.linear_model.LogisticRegression
Examples using sklearn.linear_model.Logistic

sklearn.linear_model.LogisticRegression

class `sklearn.linear_model.` **`LogisticRegression`**(*penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='liblinear', max_iter=100, multi_class='ovr', verbose=0, warm_start=False, n_jobs=1*) [source]

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the 'multi_class' option is set to 'ovr', and uses the cross- entropy loss if the 'multi_class' option is set to 'multinomial'. (Currently the 'multinomial' option is supported only by the 'lbfgs', 'sag' and 'newton-cg' solvers.)

This class implements regularized logistic regression using the 'liblinear' library, 'newton-cg', 'sag' and 'lbfgs' solvers. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted (and copied).

The 'newton-cg', 'sag', and 'lbfgs' solvers support only L2 regularization with primal formulation. The 'liblinear' solver supports both L1 and L2 regularization, with a dual formulation only for the L2 penalty.

Read more in the User Guide.

Parameters: **penalty** : str, 'l1' or 'l2', default: 'l2'

Used to specify the norm used in the penalization. The 'newton-cg', 'sag' and 'lbfgs' solvers support only l2 penalties.

New in version 0.19: l1 penalty with SAGA solver (allowing 'multinomial' + L1)

http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression

要点总结

- 逻辑回归
  - 优缺点
    - 优点：可解释性强、输出概率结果、可用于排序、添加特征方便
    - 缺点：模型效果与特征工程程度有关系、数据要做好预处理
  - 样本与数据处理
    - 数据样本采样
    - 特征离散化、独热向量编码
  - 工具包
    - Liblinear | Spark | Scikit-learn

04

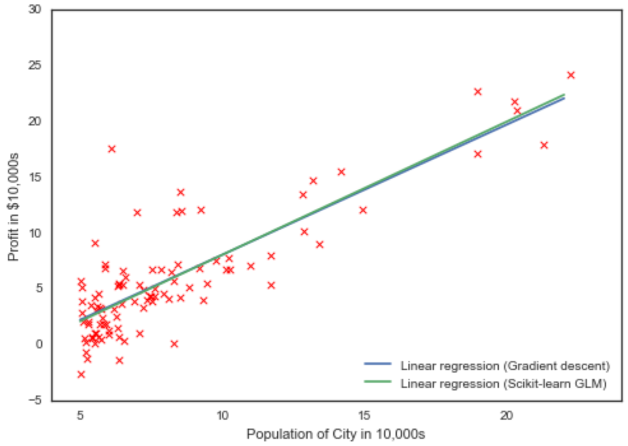
数据案例讲解

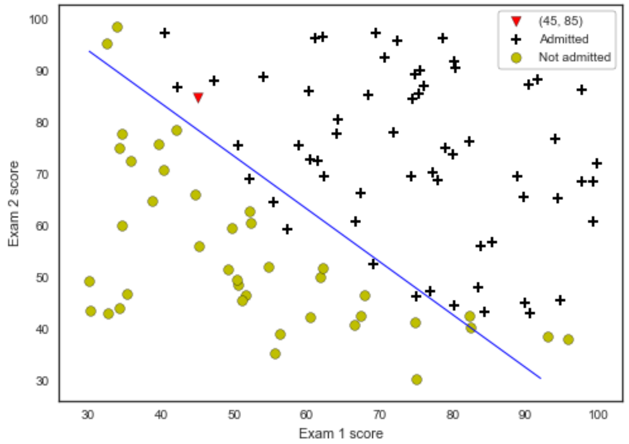
4.1

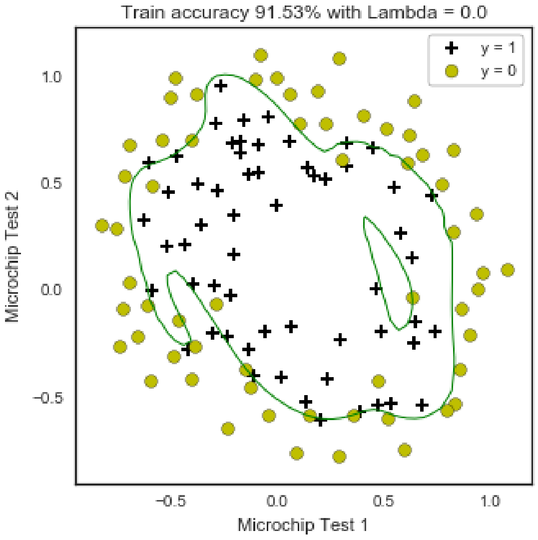
Python完成线性回归与逻辑回归

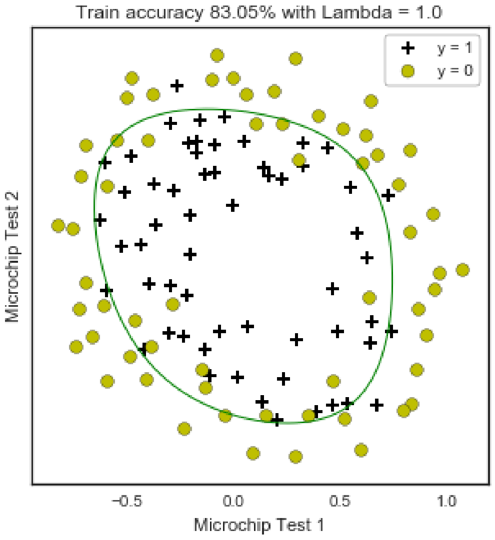
4.1 Python完成线性回归与逻辑回归

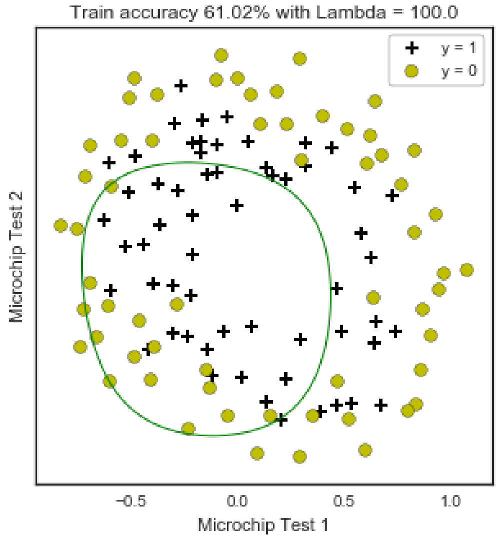
详见课程案例讲解











附

总结

- 逻辑回归
  - 优缺点
    - 优点：可解释性强、输出概率结果、可用于排序、添加特征方便
    - 缺点：模型效果与特征工程程度有关系、数据要做好预处理
  - 样本与数据处理
    - 数据样本采样
    - 特征离散化、独热向量编码
  - 工具包
    - Liblinear
    - Spark
    - Scikit-learn

参考文献/Reference

- Prof. Andrew Ng. Machine Learning. Stanford University
- 李航，统计学习方法，清华大学出版社，2012
- 周志华，机器学习，清华大学出版社，2016
- Thomas M. Cover, Joy A. Thomas. Elements of Information Theory. 2006
- Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer-Verlag. 2006

THANK YOU！

Machine Learning Engineer
机器学习工程师微专业

稀牛学院 網易 NETEASE