Lecture9 线性回归&逻辑回归

1. 线性回归 Lineal Regression

介绍

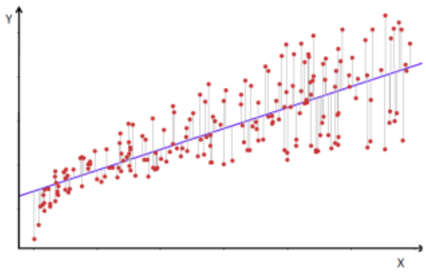
给定：训练数据 $(x_1, y_1), \ldots (x_n, y_n)$

- $x_i \in \mathbb{R}^d, y \in \mathbb{R}$

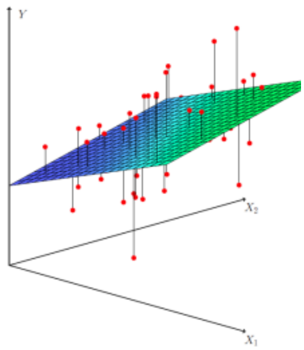
example $x_1 \rightarrow$	x_{11}	x_{12}	$\ldots$	x_{1d}	$y_1 \leftarrow$ label
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
example $x_i \rightarrow$	x_{i1}	x_{i2}	$\ldots$	x_{id}	$y_i \leftarrow$ label
$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$	$\ldots$
example $x_n \rightarrow$	x_{n1}	x_{n2}	$\ldots$	x_{nd}	$y_n \leftarrow$ label

任务：学习一个回归函数 $f: \mathbb{R}^d \rightarrow \mathbb{R}$，使得 $f(x) \rightarrow y$

线性回归 Linear Regression：如果一个回归模型用**线性函数**来表示，那么它就是线性回归模型



$d = 1$, line in $\mathbb{R}^2$



$d = 2$, hyperplane is $\mathbb{R}^3$

线性回归模型

线性回归模型的函数为

$$f(x_i) = \beta_0 + \sum_{j=1}^{d} \beta_j x_{ij} \quad \beta_j \in \mathbb{R}, j \in \{1, \ldots, d\}$$

- 这些 β 也叫做参数 / 系数或者权重
- 学习线性模型即是学习这些 β

最小二乘估计损失函数

使用最小二乘损失函数作为样本的**测试误差 / 损失函数**

$$loss(y_i, f(x_i)) = (y_i - f(x_i))^2$$

我们的目标就是最小化所有样本的损失函数，也就是说，最小化风险 / 成本函数 R

$$R = \frac{1}{2n}\sum_{i=1}^{n}(y_i - f(x_i))^2$$

- 这里使用 $\frac{1}{2n}$ 只是为了方便求导

一次函数线性回归模型

一个很简单的示例，当 $d = 1$，即我们建立的线性回归模型为 $f(x) = \beta_0 + \beta_1 x$

我们希望最小化的是

$$R(\beta_0, \beta_1) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - f(x_i))^2 = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$$

希望寻找到 β_0 和 β_1 最小化 R，最小化它们，也就是说要满足

$$\frac{\partial R}{\partial \beta_0} = 0 \quad \frac{\partial R}{\partial \beta_1} = 0$$

$$\begin{aligned}
\frac{\partial R}{\partial \beta_0} = 2 \times \frac{1}{2n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) \times \frac{\partial}{\partial \beta_0}(y_i - \beta_0 - \beta_1 x_i) \\
\frac{\partial R}{\partial \beta_0} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i) \times (-1) = 0 \\
\beta_0 = \frac{1}{n}\sum_{i=1}^{n} y_i - \beta_1 \frac{1}{n}\sum_{i=1}^{n} x_i
\end{aligned}$$

$$
\frac{\partial R}{\partial \beta_1} = 2 \times \frac{1}{2n} \sum_{i=1}^{n} [(y_i - \beta_0 - \beta_1 x_i) \times \frac{\partial}{\partial \beta_1}(y_i - \beta_0 - \beta_1 x_i)]
$$

$$
\frac{\partial R}{\partial \beta_1} = \frac{1}{n} \sum_{i=1}^{n} [(y_i - \beta_0 - \beta_1 x_i) \times (-x_i)] = 0
$$

$$
\beta_1 \sum_{i=1}^{n} x_i^2 = \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} \beta_0 x_i
$$

解上述表达式，可以解得

$$
\beta_1 = \frac{\sum_{i=1}^{n} y_i x_i - \frac{1}{n} \sum_{i=1}^{n} y_i \sum_{i=1}^{n} x_i}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n} \sum_{i=1}^{n} x_i \sum x_i}
$$

高次函数线性回归模型

如果是高维的特征，即

$$
f(x_i) = \beta_0 + \sum_{j=1}^{d} \beta_j x_{ij}
$$

我们要找到一组 $\beta_0, \ldots, \beta_j$ 满足最小化

$$
R = \frac{1}{2n} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{d} \beta_j x_{xj})^2
$$

可以使用矩阵表达

- 让 $\boldsymbol{X}$ 是一个 $n \times (d+1)$ 的矩阵，其中每一行的第一个数都是 1
- 让 $\boldsymbol{y}$ 是每个样本的标签
- 让 $\boldsymbol{\beta}$ 是权重矩阵

$$
\boldsymbol{X} := \begin{pmatrix} 1 & x_{11} & \cdots & x_{1j} & \cdots & x_{1d} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{i1} & \cdots & x_{ij} & \cdots & x_{id} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n1} & \cdots & x_{nj} & \cdots & x_{nd} \end{pmatrix} \quad \boldsymbol{y} := \begin{pmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_n \end{pmatrix} \quad \boldsymbol{\beta} := \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_j \\ \vdots \\ \beta_d \end{pmatrix}
$$

现在我们想要找到一个合理的 $\boldsymbol{\beta}$ 来最小化 R

$$R(\boldsymbol{\beta}) = \frac{1}{2n}||(\boldsymbol{y} - \boldsymbol{X\beta})||^2$$

$$\begin{aligned}
R(\boldsymbol{\beta}) &= \frac{1}{2n}(\boldsymbol{y} - \boldsymbol{X\beta})^T(\boldsymbol{y} - \boldsymbol{X\beta}) \\
\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{\beta}} &= -\frac{1}{n}\boldsymbol{X^T}(\boldsymbol{y} - \boldsymbol{X\beta})
\end{aligned}$$

我们解得 $X^T(y - X\beta) = 0$，最终

$$\boldsymbol{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}$$

梯度下降 Gradient descent

梯度下降是一种优化方法

重复下述方法指导收敛，以一次函数线性回归为例

同时更新所有的 β_j

$$\begin{aligned}
\beta_0 &:= \beta_0 - \alpha\frac{\partial}{\partial\beta_0}R\left(\beta_0, \beta_1\right) \\
\beta_1 &:= \beta_1 - \alpha\frac{\partial}{\partial\beta_1}R\left(\beta_0, \beta_1\right)
\end{aligned}$$

- α： 学习率

我们上面已经给出了

$$\begin{aligned}
\frac{\partial R}{\partial\beta_0} &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \beta_0 - \beta_1 x_i\right) \times (-1) \\
\frac{\partial R}{\partial\beta_1} &= \frac{1}{n}\sum_{i=1}^{n}\left[\left(y_i - \beta_0 - \beta_1 x_i\right) \times \left(-x_i\right)\right]
\end{aligned}$$

那么我们得到了梯度下降的公式

$$\begin{aligned}
\beta_0 &:= \beta_0 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(\beta_0 + \beta_1 x_i - y_i\right) \\
\beta_1 &:= \beta_1 - \alpha\frac{1}{n}\sum_{i=1}^{n}\left(\beta_0 + \beta_1 x_i - y_i\right)\left(x_i\right)
\end{aligned}$$

实际的考虑

- **规范化**：让特征的值规范在一个相似的比例，例如 $x_i := \frac{x_i - u_i}{stdev(x_i)}$
- **学习率**：不要选择过小或过大的学习率
- R 应该在每一次**迭代中逐渐降低**
- 声明收敛的定义，如果它的梯度下降小于 ϵ，它是收敛的
- 什么时候 X^TX 不是可逆的？
  - 特征过多，比如 50 个样本，500 个特征
  - 特征线性相关（重量同时存在磅和千克两个特征）

优缺点

标准解法：解方程

- 优点：不需要指定收敛速度或迭代
- 缺点：只有在 X^TX 可逆的时候可以求，且在 d 非常大的时候，计算 $(X^TX)^{-1}$ 时间复杂度为 $O(d^3)$

迭代解法：梯度下降

- 优点：高维度的时候效率高
- 缺点：需要很多次迭代才能收敛，需要选择学习率 α

2. 逻辑回归 Logistic Regression

介绍

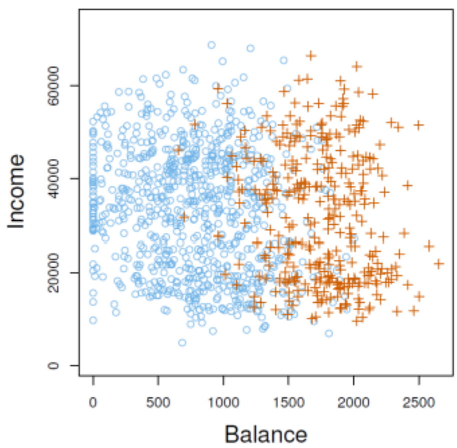
给定：训练数据 $(x_1, y_1), \dots (x_n, y_n)$

- $x_i \in \mathbb{R}^d$
- y_i 是离散的（可分类的）$y_i \in Y$（例如 $Y = \{-1, +1\}, Y = \{0, 1\}$）

任务：学习一个分类函数 $f : \mathbb{R}^d \to Y$

线性分类 Linear Classification：如果一个分类模型用一个**线性函数**来表示，那么它就是线性分类模型

示例



- 我们无法准确预测信用卡违约，假设我们想要预测客户违约的可能性，也就是输出 0 到 1 之间客户违约的概率
- 在这种情况下，输出是实数的（回归），但是是被划分的（分类）
  - 比如输出的概率是 0.65 > 0.5，被分类为可能是信用卡违约用户
- 回归学习是的发生某件时期的概率

$$P(y|x)$$

- 在这个问题中，我们预测的是 $P(default = yes|balance)$

线性回归模型转为逻辑回归模型

是否能用**线性回归**拟合？

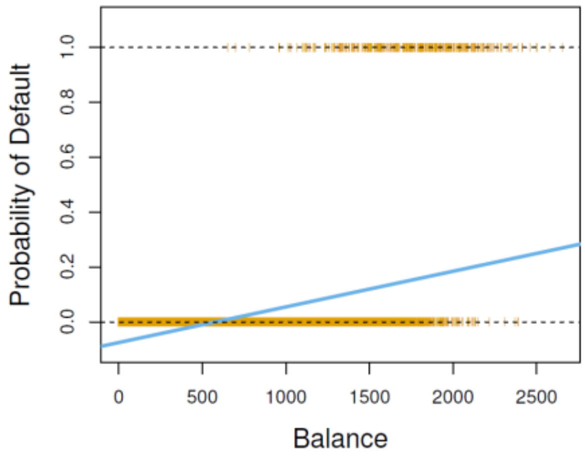
一次线性回归模型中，如果要构建，我们初始构建的是

$$f_{old}(x_i) = \beta_0 + \beta_1 \times x_i$$

- 在这个问题中 $f_{old}(x_i) = \beta_0 + \beta_1 \times balance$
- 需要满足 $0 \leq f(x_i) \leq 1, \ \ f(x_i) = P(y = 1|x_i)$

线性回归可以使用，但是存在问题

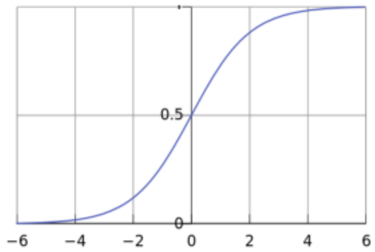
- 如果使用线性回归，其它情况下一些预测的结果可能超出 $[0, 1]$ 的范围
- 模型效果不是很好



- 如图是线性拟合的结果，由于 y_i 是离散的，不是 1 就是 0，如果按照 0.5 划分成不会违约，那么所有的样本都是不违约的，效果很差

Sigmoid 函数

我们使用一种叫做 `Sigmoid` 的激活函数



$$g(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}}$$

Sigmoid 函数有几个好的特点

- $g(z) \to 1$ 当 $z \to +\infty$
- $g(z) \to 0$ 当 $z \to -\infty$
- $g(z)' = g(z)(1-g(z))$

逻辑回归模型

我们可以巧妙的把原来的线性函数转换成 Sigmoid 函数

$$g(f_{old}(x_i)) = \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}$$

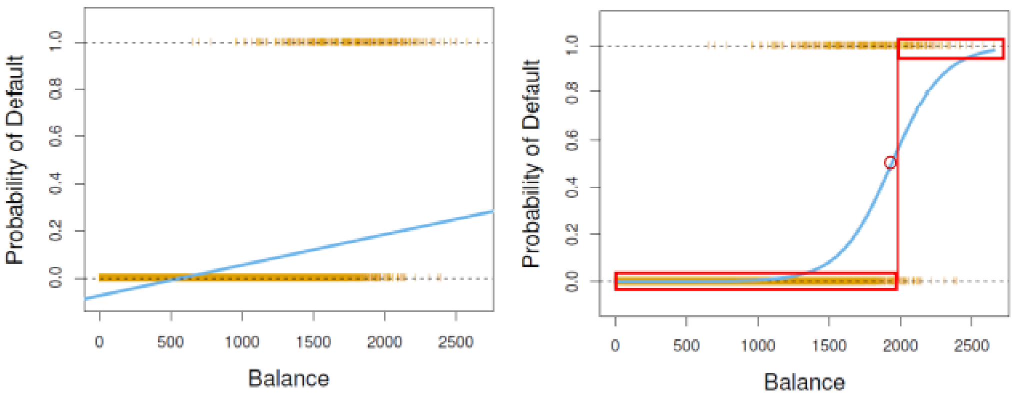
我们把这个函数作为新的**逻辑回归模型**

$$f(x_i) = g(\beta_0 + \beta_1 x_i) = \frac{e^{\beta_0+\beta_1 x_i}}{1+e^{\beta_0+\beta_1 x_i}}$$

当然，如果是**高次逻辑回归模型**的话，写法就是

$$f(x_i) = g(\sum_{j=1}^{d} \beta_j x_j + \beta_0)$$

- 换句话说，对逻辑回归模型对输出进行强制转换，使线性函数值介于 0 和 1 之间
- 不仅如此，因为 sigmoid 函数的良好的求导特性，使得梯度下降非常简单计算



- 如图所示，此时如果按照 y_i 是否大于 0.5 划分是否违约，就可以很好的划分一个较为合理的界限

逻辑回归模型预测示例

如何做出预测，假设 $\beta_0 = -10.65$，$\beta_1 = 0.0055$，那么如果有一个 balance = $1000 的客人，他是否会违约?

$$P(default = yes|balance = 1000) = \frac{1}{1 + e^{10.65 - 0.0055 * 1000}} = 0.00576$$

- $g(x_i) \leq 0.5$，预测客人不会违约

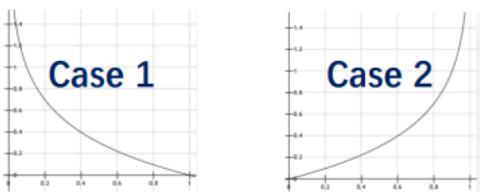
逻辑回归模型中损失函数的定义

在线性回归中，我们的损失函数被定义为

$$loss = \frac{1}{2}(y_i - f(x_i))^2$$
$$R(\beta) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - f(x_i))^2$$

- $\frac{1}{2}$ 只是为了方便求导，没有什么实际意义
- 注意，如果仍然使用这种方法的话，这里 $y_i - f(x_i)$ 事实上不是一个线性函数了，它可能存在很多个局部最优解，因此梯度下降无法找到全局最优解
- 我们不能不能以最小化这样一个 $R(\beta)$ 来找到合适的一组 β
- 我们需要另一个凸函数

这里我们找到了新的一个损失函数，由于 $Y = \{0, 1\}$



$$\text{loss}(f(x_i), y_i) = \begin{cases} -\log\left(f(x_i)\right) & \text{if } y_i = 1 \\ -\log\left(1 - f(x_i)\right) & \text{if } y_i = 0 \end{cases}$$

- 如果 $y = 1$ 且 $f(x_i) = 1$，那么 $loss = 0$
- 如果 $y = 1$ 且 $f(x_i) = 0$，那么 $loss \to \infty$
- 如果 $y = 0$ 且 $f(x_i) = 0$，那么 $loss = 0$
- 如果 $y = 0$ 且 $f(x_i) = 1$，那么 $loss \to \infty$

合并一下这两个情况，可以得到

$$loss(f(x_i), y_i) = -y_i log f(x_i) - (1 - y_i) log(1 - f(x_i))$$
$$R(\boldsymbol{\beta}) = -\frac{1}{n}[\sum_{i=1}^{n} y_i log f(x_i) + (1 - y_i) log(1 - f(x_i))]$$

我们的目标是找到一组 $\boldsymbol{\beta}$，使得这样的 $R(\boldsymbol{\beta})$ 最小

梯度下降

与线性回归相同，对所有的 β_i 进行更新

$$\beta_j := \beta_j - \alpha \frac{\partial}{\partial \beta_0} R(\boldsymbol{\beta})$$

计算可得

$$\beta_j := \beta_j - \alpha \sum_{i=1}^{m} (f(x_i) - y_i) x_{ij}$$

- x_{ij}：样本 x_i 的第 j 个特征
- 注意这里的 $f(x_i)$ 是逻辑回归模型