简单线性回归算法公式推导

$$\text{求损失函数}\sum_{i=1}^{m}(y^{(i)}-ax^{(i)}-b)^2\ \text{的最小值}$$

最小二乘法求解:

本质是试图找到一条直线，使得样本上的点到直线的欧式距离之和最小

$$J(a,b)=\sum_{i=1}^{m}(y^{(i)}-ax^{(i)}-b)^2$$

对J(a,b)求导并且令 $\frac{\partial J(a,b)}{\partial a}=0$ ，$\frac{\partial J(a,b)}{\partial b}=0$

(一)

$$\frac{\partial J(a,b)}{\partial b}=\sum_{i=1}^{m}2(y^{(i)}-ax^{(i)}-b)(-1)\ =0$$

化简

$$\sum_{i=1}^{m}y^{(i)}-a\sum_{i=1}^{m}x^{(i)}-\sum_{i=1}^{m}b=0$$

$$\sum_{i=1}^{m}y^{(i)}-a\sum_{i=1}^{m}x^{(i)}-mb=0$$

$$\sum_{i=1}^{m}y^{(i)}-a\sum_{i=1}^{m}x^{(i)}=mb$$

两边同除 m

$$\mathbf{b=\bar{y}-a\bar{x}}$$

（二）

$$\frac{\partial J(a,b)}{\partial b}=\sum_{i=1}^{m}2(y^{(i)}-ax^{(i)}-b)(-x^{(i)})\ =0$$

化简并将 b 带入

$$\sum_{i=1}^{m}(y^{(i)}-ax^{(i)}-\bar{y}+a\bar{x})\,x^{(i)}=0$$

$$\sum_{i=1}^{m}\left(x^{(i)}y^{(i)}-ax^{(i)}x^{(i)}-x^{(i)}\bar{y}+a\bar{x}\,x^{(i)}\right)=0$$

$$\sum_{i=1}^{m}\left(x^{(i)}y^{(i)}-x^{(i)}\bar{y}+a\bar{x}\,x^{(i)}-ax^{(i)}x^{(i)}\right)=0$$

$$\sum_{i=1}^{m}\left(x^{(i)}y^{(i)}-x^{(i)}\bar{y}\right)\ =a\sum_{i=1}^{m}(x^{(i)}x^{(i)}-\bar{x}\,x^{(i)})$$

$$a=\frac{\sum_{i=1}^{m}\left(x^{(i)}y^{(i)}-x^{(i)}\bar{y}\right)}{\sum_{i=1}^{m}(x^{(i)}x^{(i)}-\bar{x}\,x^{(i)})}$$

因为

$$\sum_{i=1}^{m}x^{(i)}\bar{y}=\bar{y}\sum_{i=1}^{m}x^{(i)}=m\bar{y}\cdot\bar{x}=\ \bar{x}\sum_{i=1}^{m}y^{(i)}=\ \sum_{i=1}^{m}x^{(i)}\bar{y}\ =\sum_{i=1}^{m}\bar{y}\,\bar{x}$$

所以

$$a=\frac{\sum_{i=1}^{m}\left(x^{(i)}y^{(i)}\ -x^{(i)}\bar{y}-\ \bar{x}y^{(i)}+\ \bar{x}\bar{y}\right)}{\sum_{i=1}^{m}(x^{(i)}x^{(i)}-\bar{x}\,x^{(i)}-\ \bar{x}\,x^{(i)}+\bar{x}^{2})}$$

$$\boldsymbol{a=\frac{\sum_{i=1}^{m}(x^{(i)}-\ \bar{x})(y^{(i)}-\bar{y})}{\sum_{i=1}^{m}(x^{(i)}-\ \bar{x})^{2}}}$$

模型输出$\mathbf{y=ax_i+b}$

多元线性回归公式推导

样本数据集 $\begin{pmatrix}x_1^1,x_2^1,x_3^1,\cdots,x_n^1,y^1\\x_1^2,x_2^2,x_3^2,\cdots,x_n^2,y^2\\\cdots\\x_1^n,x_2^n,x_3^n,\cdots,x_n^n,y^n\end{pmatrix}=(X_1^{(i)},X_2^{(i)},X_3^{(i)},\cdots,X_n^{(i)},Y^{(i)})$

求损失函数$\sum_{i=1}^{m}(\boldsymbol{y^{(i)}-\hat{y}^{(i)}})^2$ 的最小值

$$\mathrm{y}=\ \theta_0+\ \theta_1x_1+\theta_2x_2+\theta_3x_3+\cdots+\theta_nx_n$$ 截距为θ_0

$$\hat{y}^{(i)}=\ \theta_0+\ \theta_1X_1^{(i)}+\theta_2X_2^{(i)}+\theta_3X_3^{(i)}+\cdots+\theta_nX_n^{(i)}$$

找出$\theta_0,\theta_1,\theta_2,\cdots,\theta_n$,使得$\sum_{i=1}^{m}(\boldsymbol{y^{(i)}-\hat{y}^{(i)}})^2$ **最小**

$$令\boldsymbol{\theta}=(\theta_0,\theta_1,\theta_2,\cdots,\theta_n)^T$$

$$\hat{y}^{(i)}=\ \theta_0X_0^{(i)}+\ \theta_1X_1^{(i)}+\theta_2X_2^{(i)}+\theta_3X_3^{(i)}+\cdots+\theta_nX_n^{(i)},X_0^{(i)}\equiv1$$

$$X_b=X_0^{(i)},X_1^{(i)},X_2^{(i)},X_3^{(i)},\cdots,X_n^{(i)}$$

$$X_b\ =\begin{pmatrix}1,x_1^1,x_2^1,x_3^1,\cdots,x_n^1,y^1\\1,x_1^2,x_2^2,x_3^2,\cdots,x_n^2,y^2\\\cdots\\1,x_1^n,x_2^n,x_3^n,\cdots,x_n^n,y^n\end{pmatrix}\quad\boldsymbol{\theta}=\begin{pmatrix}\theta_0\\\theta_1\\\theta_2\\\cdots\\\theta_n\end{pmatrix}$$

$$\sum_{i=1}^{m}(y^{(i)}-\hat{y}^{(i)})^2=(\mathrm{y}-X_b\cdot\boldsymbol{\theta})^T(\mathrm{y}-X_b\cdot\boldsymbol{\theta})$$

最终多元线性回归方程的正规方程解　　$\boldsymbol{\theta} = (X_b^T X_b)^{-1} X_b^T y$

模型输出$\hat{y}^{(i)} = X_b \boldsymbol{\theta}$

参数

超参数	解释	数值类型(默认值)	optional
fit_intercept	是否计算此模型的截距。 如果设置为 False，则不会在计算中使用截距（例如，预计数据已经居中	Boolean（True）	True/False
normalize	fit_intercept 设置为 False 时，将忽略此参数。 如果为真，则回归量 X 将在回归之前通过减去平均值并除以 l2 范数来归一化。 如果您希望标准化，请 在使用估算器 sklearn.preprocessing.StandardScaler 之前 fit 使用 normalize=False。	Boolean（False）	True/False
copy_X	如果为 True，则将复制 X; 否则，它可能会被覆盖。(不太懂)	Boolean（True）	True/False
n_jobs	确定 cpu 的核数，-1 表示使用所有处理器	Int（ None）	

属性	解释	类型	Shape
coef_	回归系数（斜率）	array	（n_features，）（n_targets，n_features）
intercept_	截距	Float	

方法	解释	类型
fit(X, y, sample_weight=None)	训练线性模型	X : array-like or 稀疏矩阵，y : array_like，sample_weight : numpy array
get_params (*deep=True*)	如果为 True，将返回此估计器的参数并包含作为模型的子对象。	***boolean***
predict(X)	使用线性模型预测	X : array-like or 稀疏矩阵
score(X, y[, sample_weight])	评分标准$R^2 = 1 - \frac{(\sum_{i=1}^{m}(\hat{y}^{(i)} - y^{(i)})^2)/m}{(\sum_{i=1}^{m}(y^{(i)} - \bar{y})^2)/m}$	float
set_params(**params)	设置参数	