扫码签到

神经网络与回归

2022.05.30

目录

1. 人工神经网络介绍

- 梯度下降优化神经网络
  - 假设网络的参数为W与b，采取的损失函数为L
  - 可以计算损失函数对W和b的偏导分别为$\frac{\partial L}{\partial W}$，$\frac{\partial L}{\partial b}$
  - 更新网络参数，公式为：$W = W - \eta\frac{\partial L}{\partial W}$，$b = b - \eta\frac{\partial L}{\partial b}$
  - 其中η为学习率

- 梯度下降优化

1. 假定网络为单层感知机，且没有激活层，没有偏置，此时，网络输出为$y = XW$
2. 设置损失函数为L_{MSE}，并随机初始化网络参数W
3. 当满足终止条件时，终止优化，否则继续
4. 计算网络输出$y = XW$，以及损失$L_{MSE} = \frac{1}{N}(XW - Y)^T(XW - Y)$
5. 求导可得$\frac{\partial L_{MSE}}{\partial W} = \frac{1}{N}X^T(XW - Y)$
6. 根据$W = W - \eta\frac{\partial L_{MSE}}{\partial W}$更新参数$W$
7. 跳转到3

2. 分类和回归

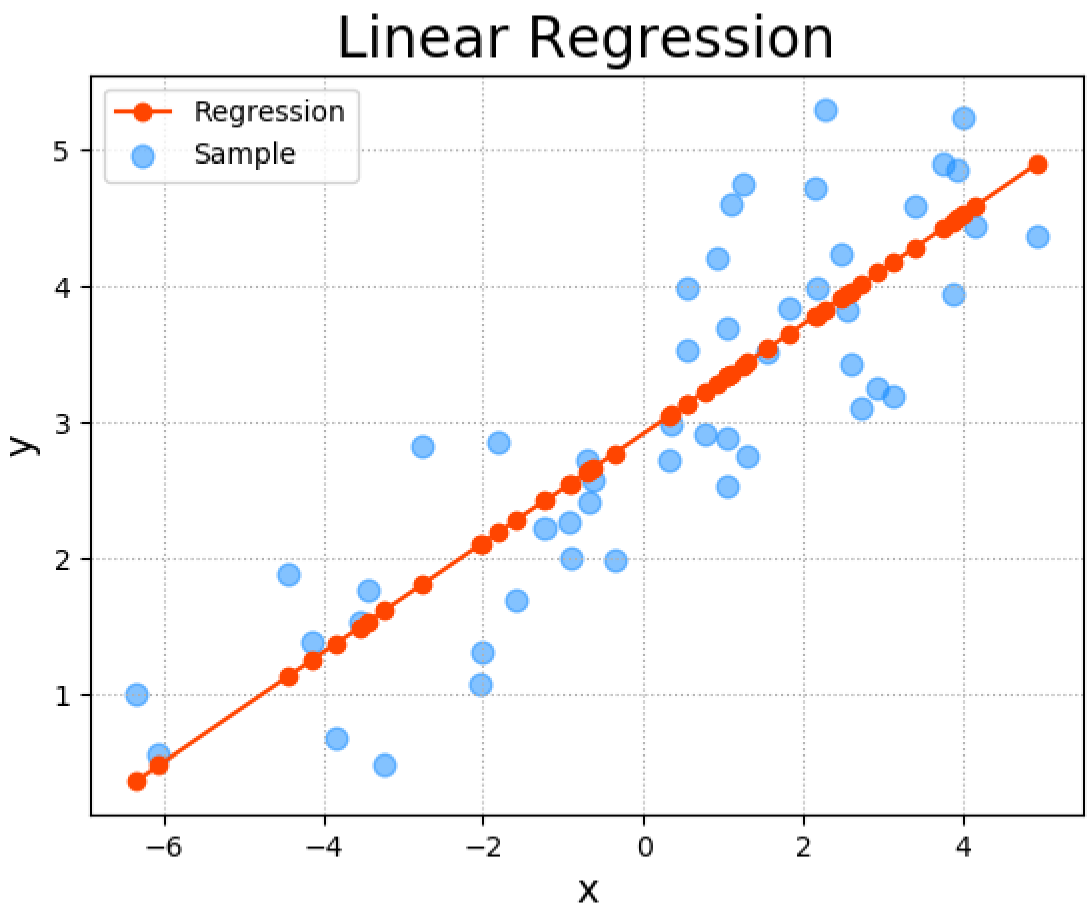
- ✓ 回归（Regression、Prediction）
  - ✓ 如何预测上海浦东的房价？
  - ✓ 未来的股票市场走向？
- ✓ 分类（Classification）
  - ✓ 身高1.85m，体重100kg的男人穿什么尺码的T恤？
  - ✓ 根据肿瘤的体积、患者的年龄来判断良性或恶性？

标签连续

标签离散

2. 分类和回归

- 线性回归（Linear Regression）
  - 是一种通过属性的线性组合来进行预测的线性模型，其目的是找到一条直线或者一个平面或者更高维的超平面，使得预测值与真实值之间的误差最小化。



2. 分类和回归—符号定义

m 代表训练集中样本的数量

n 代表特征的数量

x 代表特征/输入变量

y 代表目标变量/输出变量

(x, y) 代表训练集中的样本

$(x^{(i)}, y^{(i)})$ 代表第i个观察样本

h 代表学习算法的解决方案或函数也称为假设（**hypothesis**）

$\hat{y} = h(x)$,代表预测的值

建筑面积	总层数	楼层	实用面积	房价
143.7	31	10	105	36200
162.2	31	8	118	37000
199.5	10	10	170	42500
96.5	31	13	74	31200
……	……	……	……	……

$x^{(i)}$是特征矩阵中的第i行，是一个**向量**。

上图的：$x^{(2)} = \begin{bmatrix} 162.2 \\ 31 \\ 8 \\ 118 \end{bmatrix}$ $y^{(2)}$ =37000

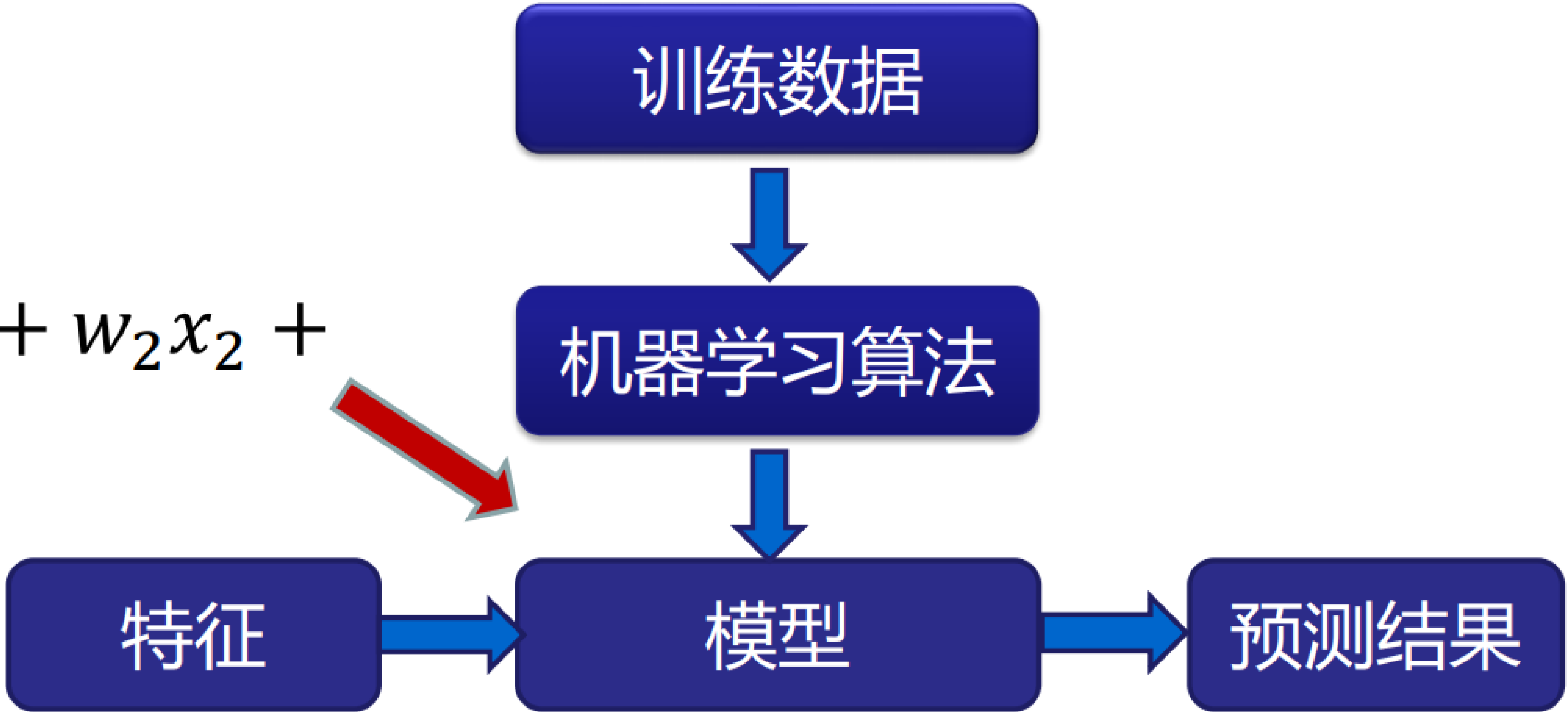
$x_j^{(i)}$代表特征矩阵中第 i 行的第 j 个特征

上图的$x_2^{(2)} = 31, x_3^{(2)} = 8$

2. 分类和回归—算法流程

x 和 y 的关系

$$h(x) = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n$$



可以设$x_0 = 1$

则：$h(x) = w_0x_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n = w^{\mathrm{T}}X$

注意：若表达式 $h(x) = w_0 + w_1x_1 + w_2x_2 + \ldots + w_nx_n + b$，　则$b$可以融入到$w_0$

2. 分类和回归—算法流程

$$h(x) = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n$$

损失函数采用平方和损失：

$$l(x^{(i)}) = \frac{1}{2}(h(x^{(i)}) - y^{(i)})^2$$

要找到一组 $w(w_0, w_1, w_2, \ldots, w_n)$，

使得$J(w) = \frac{1}{2}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2$

(残差平方和) 最小

损失函数(Loss Function)度量单样本预测的错误程度，损失函数值越小，模型就越好。常用的损失函数包括：0-1损失函数、平方损失函数、绝对损失函数、对数损失函数等。

代价函数(Cost Function)度量全部样本集的平均误差。常用的代价函数包括均方误差、均方根误差、平均绝对误差等。

目标函数(Object Function)代价函数和正则化函数，最终要优化的函数。

备注：损失函数的系数1/2是为了便于计算，使对平方项求导后的常数系数为1，这样在形式上稍微简单一些。有些教科书把系数设为1/2，有些设置为1，这些都不影响结果。

2. 分类和回归—回归Loss

均方误差（Mean Square Error,MSE）

$$\mathrm{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2$$

均方根误差 RMSE(Root Mean Square Error,RMSE)

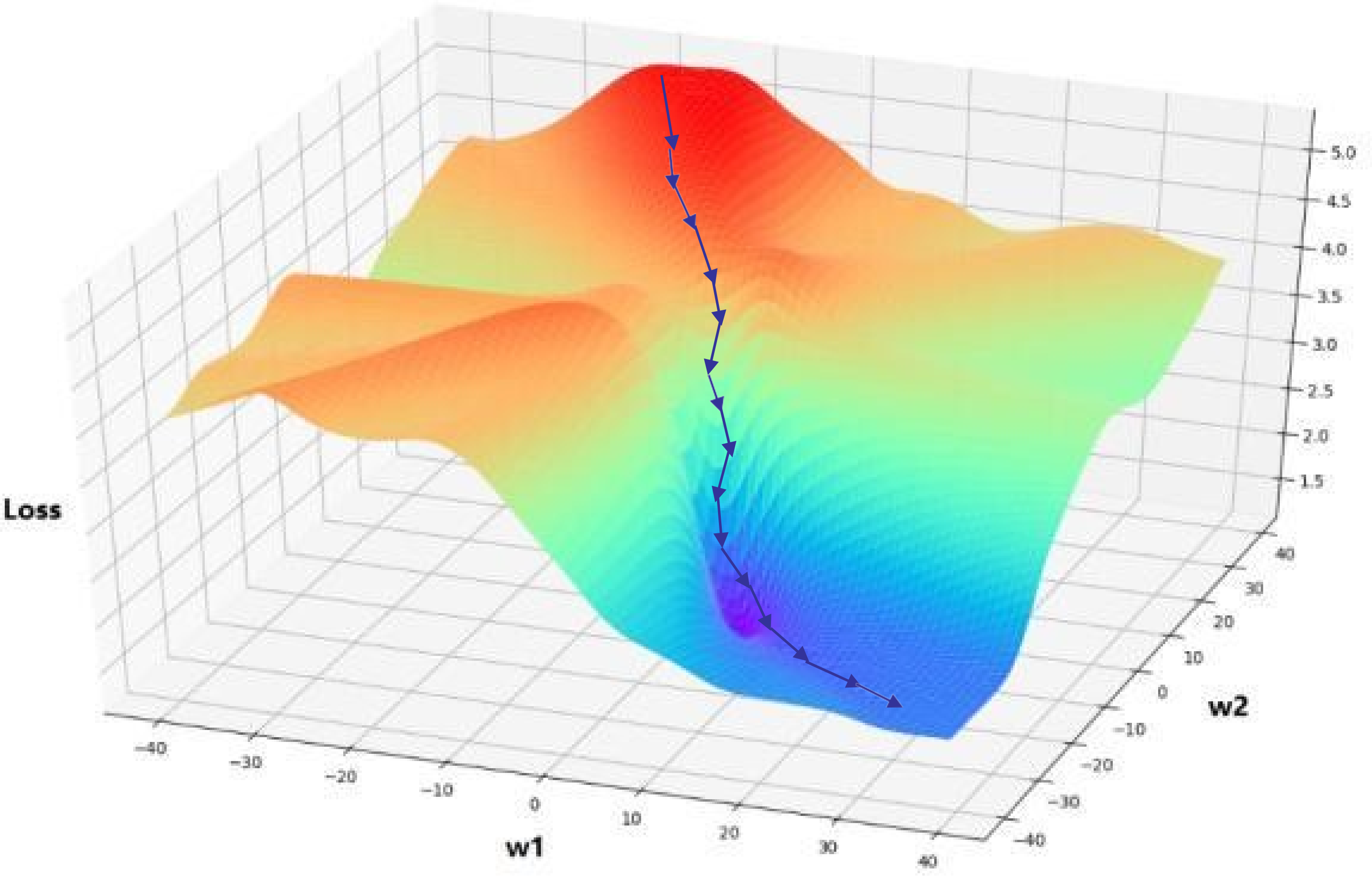
$$\mathrm{RMSE}(y,\hat{y}) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - \hat{y}^{(i)})^2}$$

平均绝对误差（Mean Absolute Error,MAE）

$$\mathrm{MAE}(y,\hat{y}) = \frac{1}{m}\sum_{i=1}^{n}\left|y^{(i)} - \hat{y}^{(i)}\right|$$

其中，$y^{(i)}$ 和$\hat{y}^{(i)}$分别表示第i个样本的真实值和预测值，m 为样本个数。

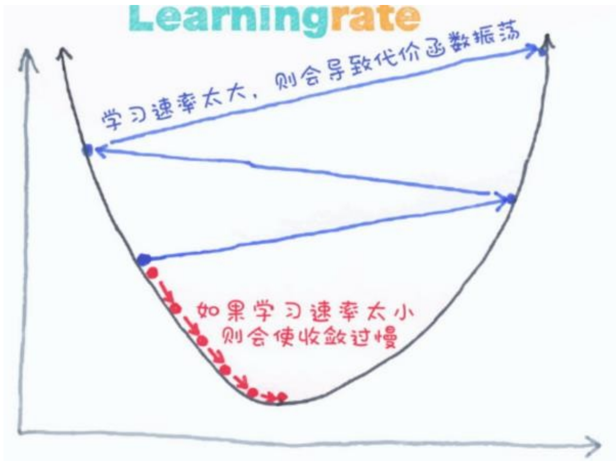
2. 分类和回归—梯度下降



学习率

α

步长



2. 分类和回归—梯度下降

- 批量梯度下降（Batch Gradient Descent,BGD）
  - 梯度下降的每一步中，都用到了所有的训练样本

参数更新

$$w_j := w_j - \underbrace{\alpha}_{\text{学习率}} \frac{1}{m}\sum_{i=1}^{m}\underbrace{\left(\left(h(x^{(i)}) - y^{(i)}\right)\cdot x_j^{(i)}\right)}_{\text{梯度}}$$

(同步更新w_j，(j=0,1,...,n))

- 随机梯度下降（Stochastic Gradient Descent,SGD）
  - 梯度下降的每一步中，用到一个样本，在每一次计算之后便更新参数 ，而不需要首先将所有的训练集求和

参数更新

$$w_j := w_j - \alpha\left(h(x^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

(同步更新w_j，(j=0,1,...,n))

- 小批量梯度下降（Mini-Batch Gradient Descent,MBGD）
  - 梯度下降的每一步中，用到了一定批量的训练样本

$$w_j := w_j - \alpha\frac{1}{b}\sum_{k=i}^{i+b-1}\left(h(x^{(k)}) - y^{(k)}\right)x_j^{(k)}$$

(同步更新w_j，(j=0,1,...,n))

b=1（随机梯度下降,SGD）
b=m（批量梯度下降,BGD）
b=*batch_size*，通常是2的指数倍，常见有32,64,128等。（小批量梯度下降,MBGD）

2. 分类和回归—数据标准化和归一化

- **归一化**：数据归一化的目的是使得各特征对目标变量的影响一致，会将特征数据进行伸缩变化，所以数据归一化是会改变特征数据分布的。
- 将数据映射到[0,1]区间

$$x^* = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

- **Z-Score**：数据标准化为了不同特征之间具备可比性，经过标准化变换之后的特征数据分布没有发生改变。
- 就是当数据特征取值范围或单位差异较大时，最好是做一下标准化处理
- 处理后的数据均值为0，方差为1

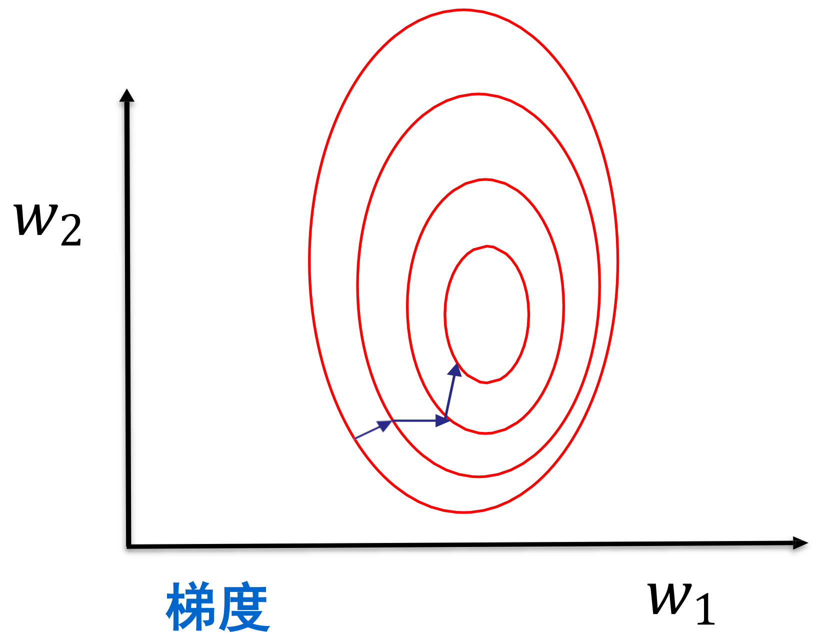
$$x^* = \frac{x - \mu}{\sigma}$$

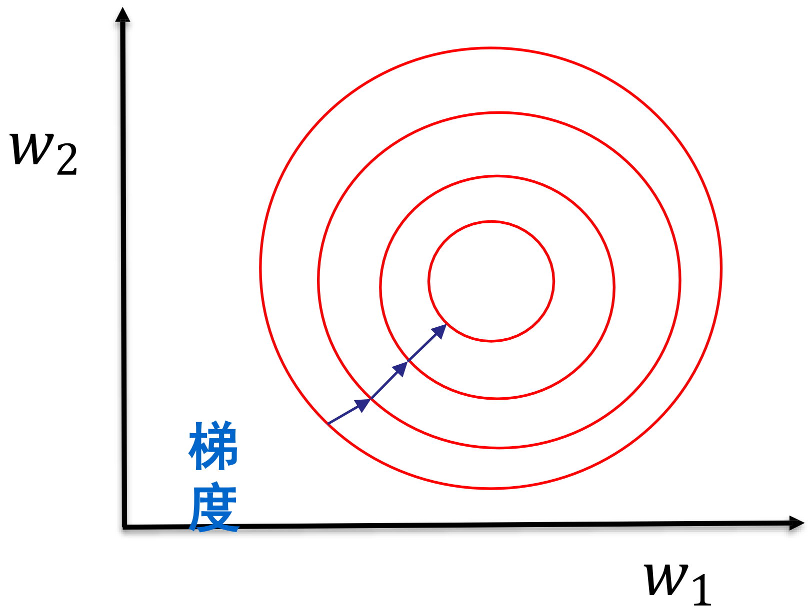
$$\sigma^2 = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)} - \mu)^2$$

$$\mu = \frac{1}{m}\sum_{i=1}^{m} x^{(i)}$$

2. 分类和回归—数据标准化和归一化

- **提升模型精度：**不同维度之间的特征在数值上有一定比较性，可以大大提高分类器的准确性
- **加速模型收敛：**最优解的寻优过程明显会变得平缓，更容易正确的收敛到最优解





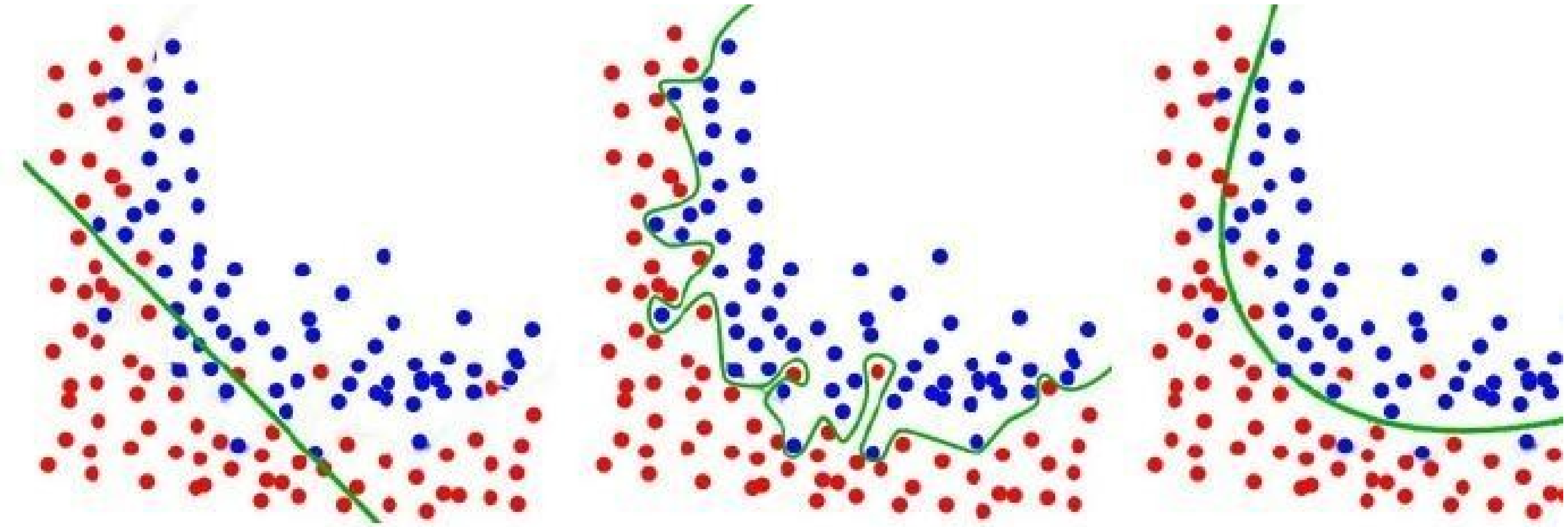
如果某一个特征值特别大，就会使对应的权重具有更大的梯度，若使用相同的学习率，则Error surface 不一定朝着最优点，如左侧。

2. 分类和回归—数据标准化和归一化

- **需要做数据归一化/标准化**
  - 线性模型，如基于距离度量的模型包括KNN(K近邻)、K-means聚类、感知机和SVM。另外，线性回归类的几个模型一般情况下也是需要做数据归一化/标准化处理的

- **不需要做数据归一化/标准化**
  - 决策树、基于决策树的Boosting和Bagging等集成学习模型对于特征取值大小并不敏感，如随机森林、XGBoost、LightGBM等树模型，以及朴素贝叶斯，以上这些模型一般不需要做数据归一化/标准化处理。

2. 分类和回归—过拟合和欠拟合



欠拟合　　　　过拟合　　　　正合适

2. 分类和回归—过拟合和欠拟合

- **过拟合的处理**

  1.**获得更多的训练数据。**使用更多的训练数据是解决过拟合问题最有效的手段，因为更多的样本能够让模型学习到更多更有效的特征，减小噪声的影响

  2.**降维。**即丢弃一些不能帮助我们正确预测的特征。可以是手工选择保留哪些特征，或者使用一些模型选择的算法来帮忙（例如PCA）。

  3.**正则化。**正则化(regularization)的技术，保留所有的特征，但是减少参数的大小（magnitude），它可以改善或者减少过拟合问题。

  4.**集成学习方法。**集成学习是把多个模型集成在一起，来降低单一模型的过拟合风险。

2. 分类和回归—过拟合和欠拟合

- **欠拟合的处理**

  1.添加新特征。当特征不足或者现有特征与样本标签的相关性不强时，模型容易出现欠拟合。通过挖掘组合特征等新的特征，往往能够取得更好的效果。

  2.增加模型复杂度。简单模型的学习能力较差，通过增加模型的复杂度可以使模型拥有更强的拟合能力。例如，在线性模型中添加高次项，在神经网络模型中增加网络层数或神经元个数等。

  3.减小正则化系数。正则化是用来防止过拟合的，但当模型出现欠拟合现象时，则需要有针对性地减小正则化系数。

2. 分类和回归—过拟合和欠拟合

- **正则化**

L_1正则化：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x^{(i)})-y^{(i)})^2+\lambda\sum_{j=1}^{n}|w_j|$，Lasso Regression（Lasso回归）

L_2正则化：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x^{(i)})-y^{(i)})^2+\lambda\sum_{j=1}^{n}w_j^2$，Ridge Regression （岭回归）

Elastic Net：$J(w)=\frac{1}{2}\sum_{i=1}^{m}(h(x^{(i)})-y^{(i)})^2+\lambda(\rho\cdot\sum_{j=1}^{n}|w_j|+(1-\rho)\cdot\sum_{j=1}^{n}w_j^2)$

(弹性网络)

其中：

- λ为正则化系数，调整正则化项与训练误差的比例，λ>0。
- 1≥ρ≥0为比例系数，调整L_1正则化与L_2正则化的比例。

2. 分类和回归—实例

- **房价预测**

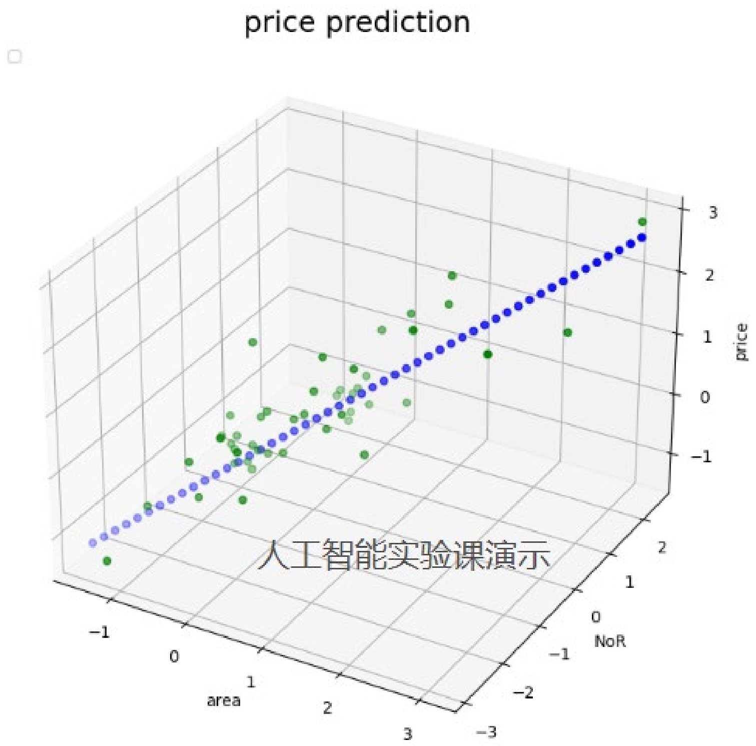
1. 假定网络为单层感知机，且没有激活层，没有偏置，此时，网络输出为 $\hat{y} = X_{train}W$,X为房子的特征，W为神经网络参数，$\hat{y}$为预测的房价
2. 设置损失函数为L_{MSE}，并随机初始化网络参数W
3. 当满足终止条件时，终止优化，否则继续
4. 计算网络输出$\hat{y} = X_{train}W$，以及损失$L_{MSE} = \frac{1}{N}(X_{train}W - Y)^T(X_{train}W - Y)$，Y为真实房价
5. 求导可得$\frac{\partial L_{MSE}}{\partial W} = \frac{1}{N}X_{train}{}^T(X_{train}W - Y)$
6. 根据$W = W - \eta\frac{\partial L_{MSE}}{\partial W}$更新参数$W$，$\eta$为学习率（步长）
7. 跳转到3

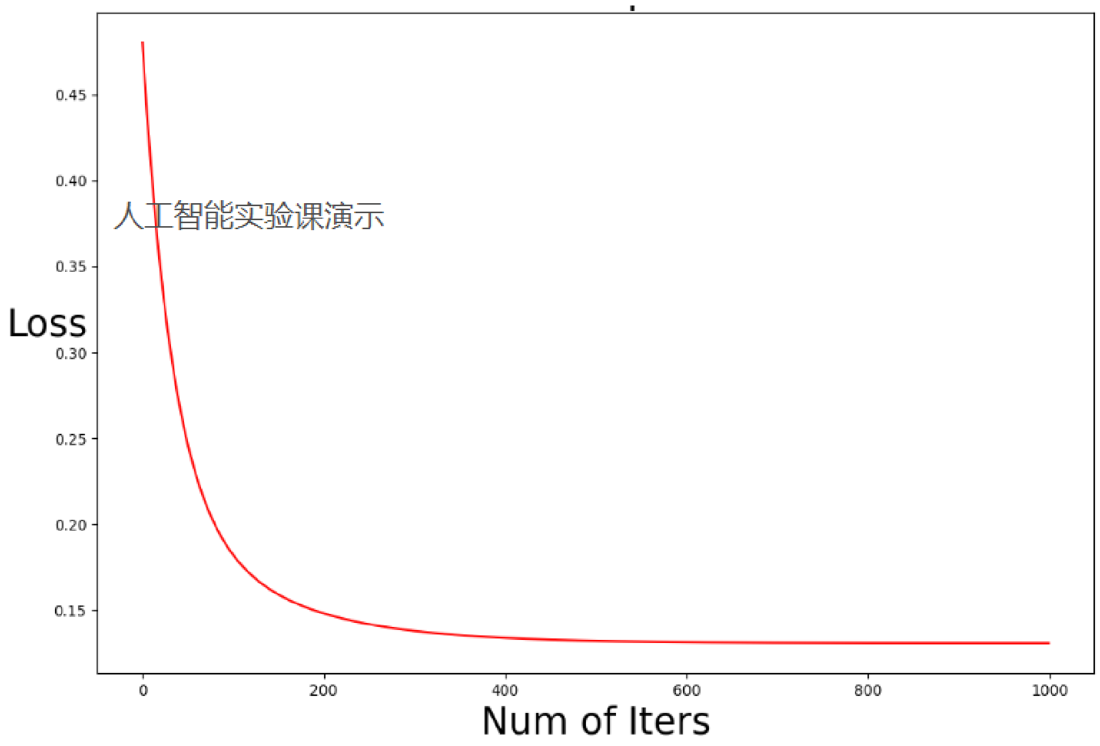
3. 实验任务

- 在给定数据集（regress_data2.csv）完成房价预测回归训练，画出训练loss曲线图、预测函数图。
- 要求
  - 设计合适的网络结构，选择合适的损失函数，利用训练集完成网络训练，画出loss曲线图、预测函数图(x:[房价数,面积]—y:预测房价)。
  - 与上周分类任务二选一完成即可

3. 实验结果示例

y = b + w0*x0 + w1*x1





附录

矩阵求导： https://zhuanlan.zhihu.com/p/137702347

矩阵运算库Numpy教程：

https://www.runoob.com/numpy/numpy-tutorial.html

Matplotlib可视化教程:

https://www.runoob.com/matplotlib/matplotlib-tutorial.html

Thanks!