# 分类：Pre梯度下降算法入门

覃雄派

分类：Pre梯度下降算法入门

# 提纲

- 分类：Pre梯度下降算法入门

π@RUC

# 分类：Pre梯度下降算法入门

- 梯度下降法的基本原理
  - 假设有一个函数L，为参数$\theta_0$和$\theta_1$的函数
    - L称为损失函数，我们希望它越小越好
    - 参数数量这里为2个，可以为1个或者多个，不影响讨论
  - 1. 我们计算L对$\theta_0$和$\theta_1$的偏导数
    - 偏导数表示，$\theta_0$和$\theta_1$的微小变化导致L发生如何的变化
  - 2. 我们的目标是最小化损失函数
    - 于是我们按照$\boxed{\theta = \theta - \eta \frac{\partial L}{\partial \theta}}$的方式进行参数的修改，$\theta$为$\theta_0$或者$\theta_1$
    - 即可把目标函数修改小一点点
  - 3. 经过一系列迭代，即可把目标函数最小化(符合一定精度要求)

接下来解释，为什么按照这样的公式修改参数，即可最小化目标函数L

π@RUC

# 分类：Pre梯度下降算法入门

- 理解梯度下降法——举个例子
  - 假设训练数据如下，为了简单起见，这里只有一个样本

| 训练数据 | x | y |
| --- | --- | --- |
| sample1 | 1 | 2 |

π@RUC

# 分类：Pre梯度下降算法入门

- 理解梯度下降法——举个例子
  - 假设训练数据如下，为了简单起见，这里只有一个样本

| 训练数据 | x | y |
|---|---|---|
| sample1 | 1 | 2 |

  - 用没有截距的一元线性回归模型y=wx进行建模，w为系数

π@RUC

# 分类：Pre梯度下降算法入门

- 理解梯度下降法——举个例子
  - 假设训练数据如下，为了简单起见，这里只有一个样本

| 训练数据 | x | y |
|---|---|---|
| sample1 | 1 | 2 |

  - 训练数据体现了数据的2倍数关系，即真实的数据体现的关系是y=f(x)=2x
    - 我们把w设置为某个初始值，比如w=3
    - 现在来看看，梯度下降法如何把3修正到2
    - 损失函数为Loss = $(f(x) - y)^2 = (wx-y)^2$
    - Loss函数针对w求偏导数$\frac{\partial}{\partial w}Loss$=2(wx-y)x
    - 那么权重w的修正公式为w=w-2η(wx-y)x，η为学习率

π@RUC

# 分类：Pre梯度下降算法入门

| 训练数据 | x | y |
|---|---|---|
| sample1 | 1 | 2 |

- 理解梯度下降法——举个例子
  - 列出了前5次迭代过程中，w的值的变化情况

| 迭代 | x | w | y | 2(wx-y)x | η | w=w-2η (wx-y)x |
|---|---|---|---|---|---|---|
| 1 | 1 | 3 | 2 | 2 | 0.1 | 2.8 |
| 2 | 1 | 2.8 | 2 | 1.6 | 0.1 | 2.64 |
| 3 | 1 | 2.64 | 2 | 1.28 | 0.1 | 2.512 |
| 4 | 1 | 2.512 | 2 | 1.024 | 0.1 | 2.4096 |
| 5 | 1 | 2.4096 | 2 | 0.8192 | 0.1 | 2.3277 |
| … | | | | | | |

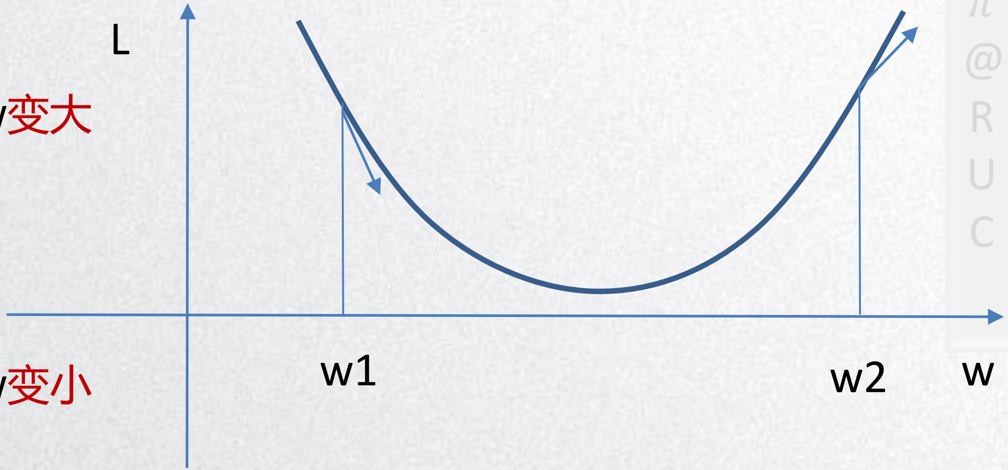
- 可以看到，按照上述公式，w逐渐逼近2.0，即x和y表达出来的正确的数量关系
- 值得注意的是，如果初始化的w小于2，那么它将从另外一个方向逼近2.0

π@RUC

# 分类：Pre梯度下降算法入门

- 梯度下降法直观解释
  - 目标损失函数对于w参数的图像如下
  - 在w1这个点上
    - 梯度$\frac{\partial}{\partial w}Loss$为一个负值
    - 按照w=w-η $\frac{\partial}{\partial w}Loss$调整，w变大
  - 在w2这个点上
    - 梯度$\frac{\partial}{\partial w}Loss$为一个正值
    - 按照w=w-η $\frac{\partial}{\partial w}Loss$调整，w变小

梯度即切线方向的变化量



# 分类：Pre梯度下降算法入门

- 理解梯度下降法——举个例子
  - 从该例子，扩展思路

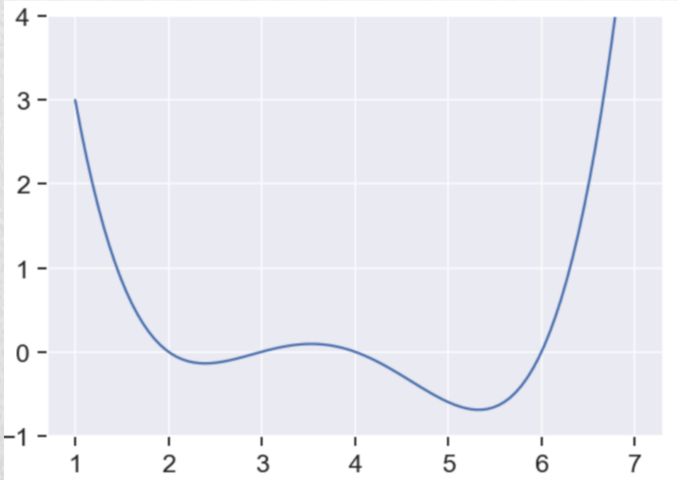
- （1）在这里，我们使用线性函数对梯度下降法进行说明，实际应用中可以用非线性函数表达x和y之间的非线性关系
- （2）在这里，只有一个样本，实际应用中一般有很多的样本
- （3）在这里，x只有一个分量，实际应用中一般x是一个多维向量；但是这些不影响我们对梯度下降法本质的理解

# 分类：Pre梯度下降算法入门

π@RUC

# 分类：Pre梯度下降算法入门

- 梯度下降法的讨论
- 求解一维函数的最小值：考虑下面的一维函数
  - $f(x) = (x^4 - 15x^3 + 80x^2 - 180x + 144)/10$




如何求解：

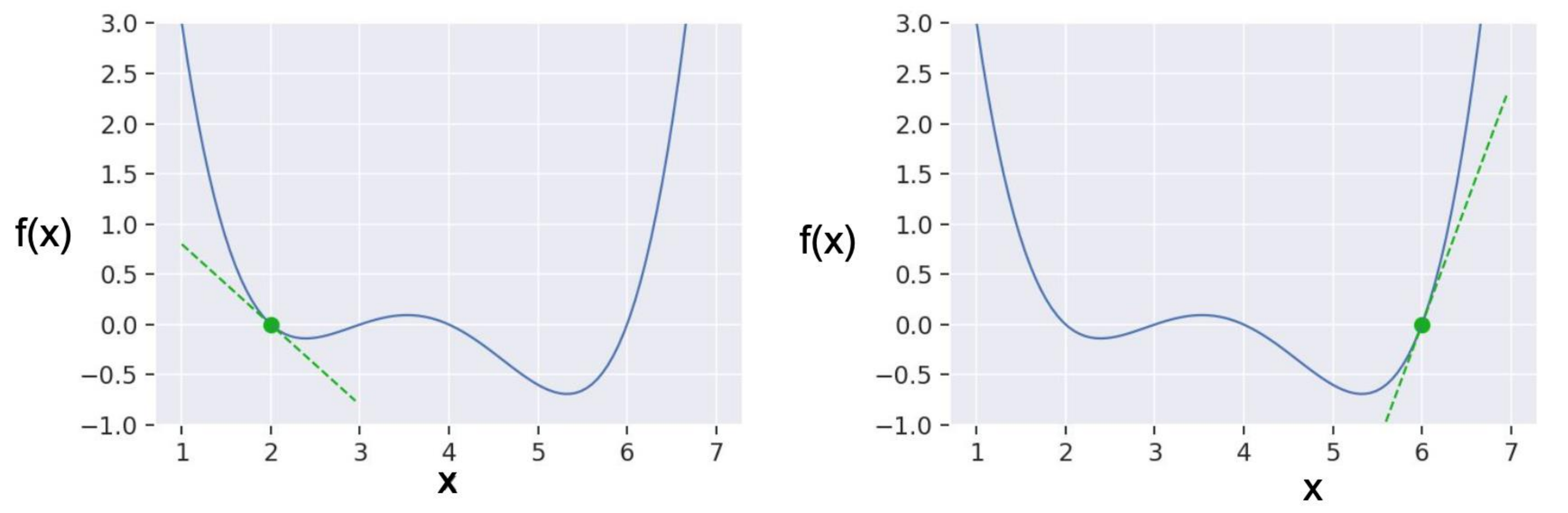
$$\hat{x} = \arg\min_{x} f(x)$$

π@RUC

# 分类：Pre梯度下降算法入门

- 一维梯度下降方法
  - 在最小值点左侧，导数是负数 → $x$应该增大
  - 在最小值点右侧，导数是正数 → $x$应该减小

导数告诉我们：
- 往哪个方向走
- 应该走多远



# 分类：Pre梯度下降算法入门

- 一维梯度下降算法：极简实现版
  - 输入：
    - gradient: 梯度（导数）函数
    - init_guess: 初始猜测
    - learn_rate: 学习率
    - n_iter: 迭代次数
  - 输出：
    - 求解的极小值点$\hat{x}$

$$x^{(t+1)} = x^{(t)} - \alpha \frac{d}{dx} f(x)$$

```
def gradient_descent(gradient, init_guess, learn_rate, n_iter):
    guess = init_guess
    for _ in range(n_iter):
        guess = guess - learn_rate * gradient(guess)
    return guess
```

# 分类：Pre梯度下降算法入门

- 定义梯度函数gradient

导数

```python
def derivative_arbitrary(x):
    return (4*x**3 - 45*x**2 + 160*x - 180)/10
```

$$f(x) = (x^4 - 15x^3 + 80x^2 - 180x + 144)/10$$

- 选择超参数
  - init_guess：一般是随机选择；作为示例，可以选择 **1**
  - n_iter: 根据实际情况，选择迭代次数；作为示例，请选择 **20**
  - learn_rate: 应该如何选择？做一些尝试

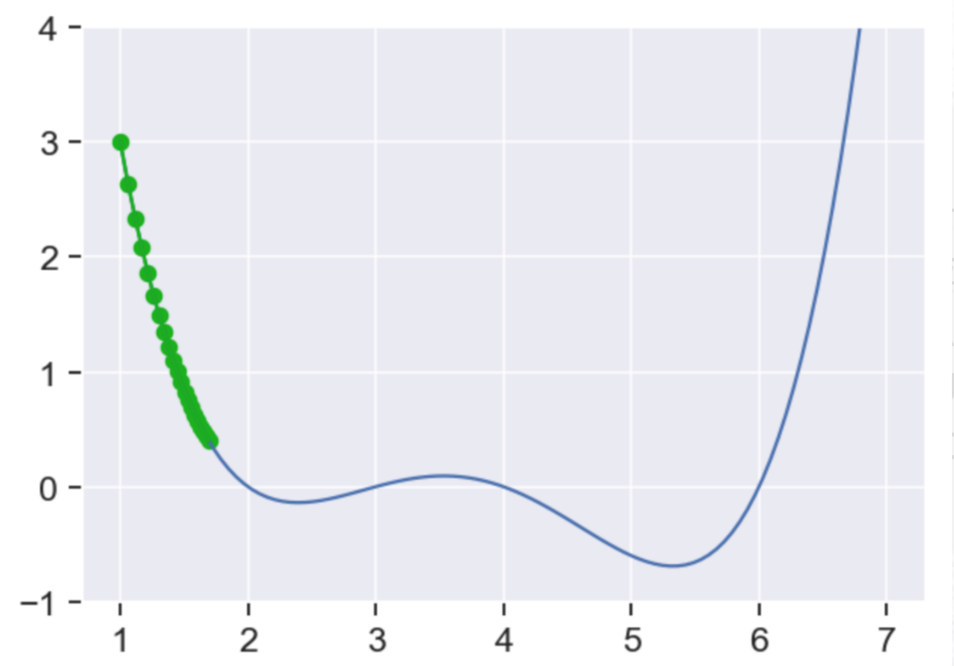
```python
def gradient_descent(gradient, init_guess, learn_rate, n_iter):
    guess = init_guess
    for _ in range(n_iter):
        guess = guess - learn_rate * gradient(guess)
    return guess
```

π@RUC

# 分类：Pre梯度下降算法入门

- 学习率的选择
- 选择一个很小的学习率
  - $\alpha = 0.01$

```
0-th iteration: guess = 1, gradient=-6.1
1-th iteration: guess = 1.06, gradient=-5.61
2-th iteration: guess = 1.12, gradient=-5.18
3-th iteration: guess = 1.17, gradient=-4.81
4-th iteration: guess = 1.22, gradient=-4.47
5-th iteration: guess = 1.26, gradient=-4.17
6-th iteration: guess = 1.3, gradient=-3.9
7-th iteration: guess = 1.34, gradient=-3.66
8-th iteration: guess = 1.38, gradient=-3.44
9-th iteration: guess = 1.41, gradient=-3.24
10-th iteration: guess = 1.45, gradient=-3.06
11-th iteration: guess = 1.48, gradient=-2.9
12-th iteration: guess = 1.51, gradient=-2.75
13-th iteration: guess = 1.53, gradient=-2.61
14-th iteration: guess = 1.56, gradient=-2.48
15-th iteration: guess = 1.58, gradient=-2.36
16-th iteration: guess = 1.61, gradient=-2.25
17-th iteration: guess = 1.63, gradient=-2.14
18-th iteration: guess = 1.65, gradient=-2.05
19-th iteration: guess = 1.67, gradient=-1.96
```



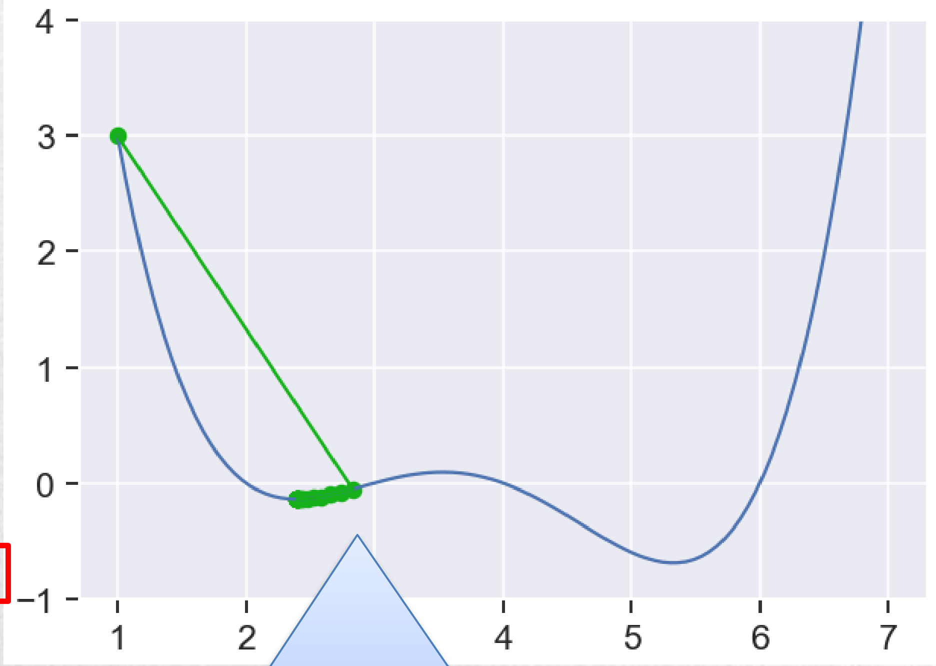

参数变化缓慢，应该如何解决？

# 分类：Pre梯度下降算法入门

- 学习率的选择
- 选择一个略小的学习率
  - $\alpha = 0.3$

```
0-th iteration: guess = 1, gradient=-6.1
1-th iteration: guess = 2.83, gradient=0.31
2-th iteration: guess = 2.74, gradient=0.28
3-th iteration: guess = 2.65, gradient=0.24
4-th iteration: guess = 2.58, gradient=0.2
5-th iteration: guess = 2.52, gradient=0.15
6-th iteration: guess = 2.48, gradient=0.1
7-th iteration: guess = 2.45, gradient=0.07
8-th iteration: guess = 2.43, gradient=0.04
9-th iteration: guess = 2.41, gradient=0.03
10-th iteration: guess = 2.41, gradient=0.02
11-th iteration: guess = 2.4, gradient=0.01
12-th iteration: guess = 2.4, gradient=0.01
13-th iteration: guess = 2.4, gradient=0.0
```

梯度已经减为0！



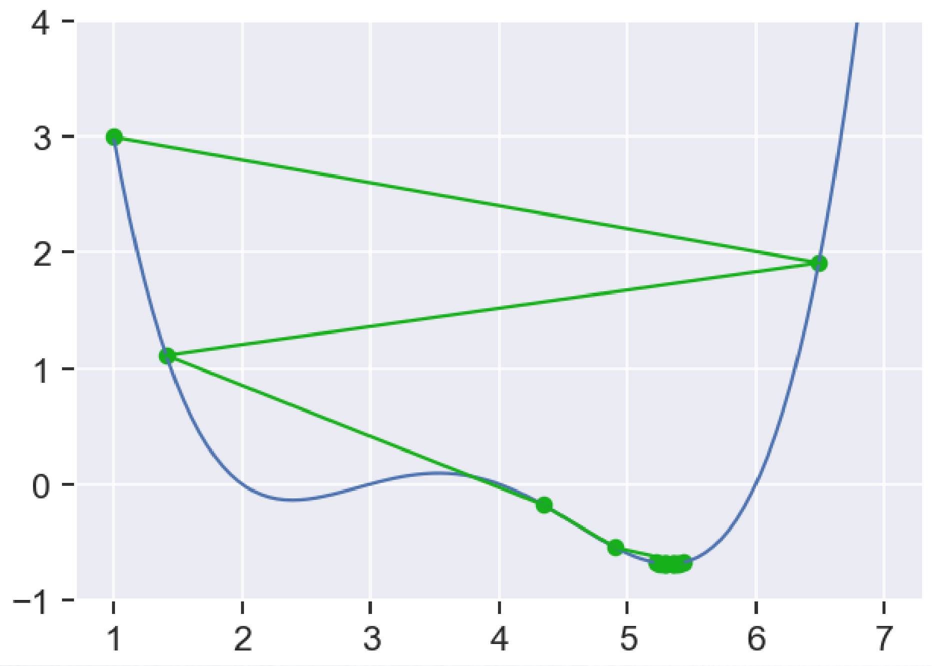

梯度下降可能陷入
局部最优（Local Minima）

π@RUC

# 分类：Pre梯度下降算法入门

- 学习率的选择
- 选择一个大一些的学习率
  - $\alpha = 0.9$

```
0-th iteration: guess = 1, gradient=-6.1
1-th iteration: guess = 6.49, gradient=5.64
2-th iteration: guess = 1.41, gradient=-3.26
3-th iteration: guess = 4.34, gradient=-0.62
4-th iteration: guess = 4.91, gradient=-0.58
5-th iteration: guess = 5.43, gradient=0.24
6-th iteration: guess = 5.22, gradient=-0.21
7-th iteration: guess = 5.4, gradient=0.18
8-th iteration: guess = 5.25, gradient=-0.16
9-th iteration: guess = 5.39, gradient=0.14
10-th iteration: guess = 5.26, gradient=-0.12
11-th iteration: guess = 5.38, gradient=0.11
12-th iteration: guess = 5.28, gradient=-0.1
13-th iteration: guess = 5.37, gradient=0.09
14-th iteration: guess = 5.29, gradient=-0.08
15-th iteration: guess = 5.36, gradient=0.07
16-th iteration: guess = 5.3, gradient=-0.06
17-th iteration: guess = 5.35, gradient=0.05
18-th iteration: guess = 5.3, gradient=-0.05
```




更大的学习率可以使算法在更大的范围内进行试探

# 分类：Pre梯度下降算法入门

- 学习率的选择
- 选择一个再大一些的学习率
  - $\alpha = 0.95$

```
0-th iteration: guess = 1, gradient=-6.1
1-th iteration: guess = 6.79, gradient=8.44
2-th iteration: guess = -1.22, gradient=-45.07
3-th iteration: guess = 41.59, gradient=21643.42
4-th iteration: guess = -20519.65, gradient=-3457865914535.88
5-th iteration: guess = 3284972598289.43, gradient=1.4179314815282369e+37
6-th iteration: guess = -1.3470349074511825e+37, gradient=-9.776795748433592e+110

---------------------------------------------------------------------------
OverflowError                             Traceback (most recent call last)
<ipython-input-39-6bf613c13fd5> in <module>
      1 plot_arbitrary()
----> 2 guess = gradient_descent_with_plot(derivative_arbitrary, arbitrary, 1, 0.95, 20)
      3 print(guess)
```

请你解释出现上述情况的原因？

# 分类：Pre梯度下降算法入门

- 学习率的选择
- 选择一个再大一些的学习率
  - $\alpha = 0.95$

```
0-th iteration: guess = 1, gradient=-6.1
1-th iteration: guess = 6.79, gradient=8.44
2-th iteration: guess = -1.22, gradient=-45.07
3-th iteration: guess = 41.59, gradient=21643.42
4-th iteration: guess = -20519.65, gradient=-3457865914535.88
5-th iteration: guess = 3284972598289.43, gradient=1.4179314815282369e+37
6-th iteration: guess = -1.3470349074511825e+37, gradient=-9.776795748433592e+110

---------------------------------------------------------------------------
OverflowError                             Traceback (most recent call last)
<ipython-input-39-6bf613c13fd5> in <module>
      1 plot_arbitrary()
----> 2 guess = gradient_descent_with_plot(derivative_arbitrary, arbitrary, 1, 0.95, 20)
      3 print(guess)
```

请你解释出现上述情况的原因？

梯度爆炸，甚至溢出

π@RUC

# 分类：Pre梯度下降算法入门

- 学习率的选择
  - 过大的学习率导致学习的不稳定，甚至不能收敛
    - 如之前的“螺旋上升”现象
  - 过小的学习率导致训练步数多读
    - 导致过长的训练时间
  - 复杂的工程问题！
    - 你会设计什么机制?

$$\boxed{x^{(t+1)} = x^{(t)} - \alpha \frac{d}{dx} f(x)}$$

```python
def gradient_descent(gradient, init_guess, learn_rate, n_iter):
    guess = init_guess
    for _ in range(n_iter):
        guess = guess - learn_rate * gradient(guess)
    return guess
```

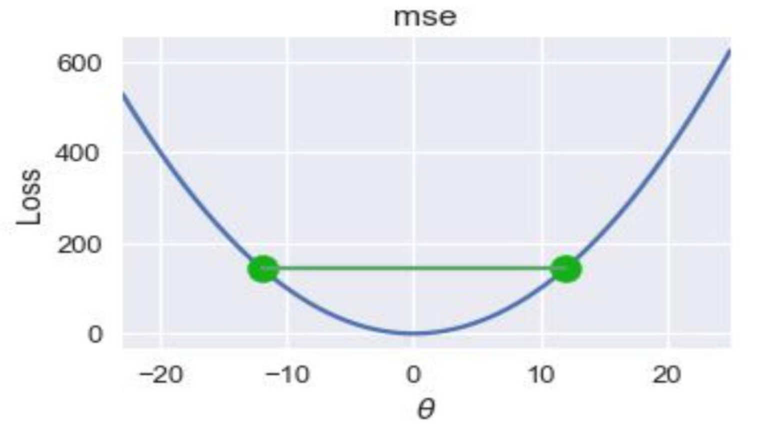
# 分类：Pre梯度下降算法入门

π@RUC

# 分类：Pre梯度下降算法入门

- 函数的凸性（Convexity）
- 定义
  - 函数$f(x)$为凸函数的充分必要条件是给定定义域中任意两个点$x_1$和$x_2$及某个常数$t \in [0,1]$，都有：

$$t \cdot f(x_1) + (1-t) \cdot f(x_2) \geq f(t \cdot x_1 + (1-t)x_2)$$



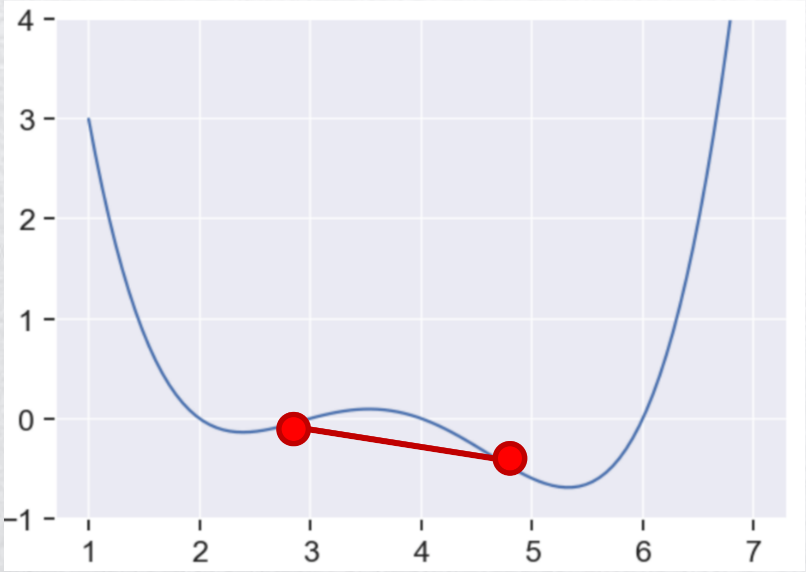

在函数$f(x)$为凸函数的情况下，梯度下降方法能够找到函数$f(x)$的最小值点

**MSE目标函数是凸函数吗？**
**A. 是  B. 否**

π@RUC

# 分类：Pre梯度下降算法入门

- 函数的凸性（Convexity）
- 请给出此一维函数不符合凸性的例子
  - $f(x) = (x^4 - 15x^3 + 80x^2 - 180x + 144)/10$




π@RUC

# 分类：Pre梯度下降算法入门

- 梯度下降法的参考代码

| 名称 | 类型 | 大小 | 修改日期 |
|---|---|---|---|
| 01gradient_descent_algorithm_intro.ipynb | IPYNB 文件 | 129 KB | 2022/2/21 14:44 |

请打开代码运行与分析

@RUC

# 回归：一元线性回归（解析解与梯度下降算法）