

# 输过败过不曾怕过

积跬步以至千里，积懈怠以致深渊。

- 博客园
- 首页
- 新随笔
- 联系
- 订阅
- 管理
- 随笔-43

## 公告

昵称：Sirius、武...  
园龄：2年9个月  
粉丝：3  
关注：4  
+加关注

|    |          |    |    |    |    |    |  |   |
|----|----------|----|----|----|----|----|--|---|
| <  | 2018年10月 |    |    |    |    |    |  | > |
| 日  | 一        | 二  | 三  | 四  | 五  | 六  |  |   |
| 30 | 1        | 2  | 3  | 4  | 5  | 6  |  |   |
| 7  | 8        | 9  | 10 | 11 | 12 | 13 |  |   |
| 14 | 15       | 16 | 17 | 18 | 19 | 20 |  |   |
| 21 | 22       | 23 | 24 | 25 | 26 | 27 |  |   |
| 28 | 29       | 30 | 31 | 1  | 2  | 3  |  |   |
| 4  | 5        | 6  | 7  | 8  | 9  | 10 |  |   |

## 搜索

找找看

谷歌搜索

## 常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签

## 随笔档案

- 2017年7月 (1)
- 2017年6月 (9)
- 2017年5月 (11)
- 2017年4月 (15)
- 2017年3月 (7)

## 最新评论

- 1. Re:Android系统架构支持支持
- 牛腩

## 阅读排行榜

- 1. 机器学习-随机梯度下降（Stochastic gradient descent）和 批量梯度下降（Batch gradient descent） (11254)
- 2. 前端开发之旅- 移动端HTML5实现文件上传(2120)
- 3. android接收mjpg-streamer软件视频流(1193)
- 4. Raspberry Pi开发之旅-光照强度检测（BH1750） (1150)

## 机器学习-随机梯度下降（Stochastic gradient descent）和 批量梯度下降（Batch descent）

梯度下降（GD）是最小化风险函数、损失函数的一种常用方法，随机梯度下降和批量梯度下降是两种迭代求解思路，从实现的角度对两者进行分析，如有哪个方面写的不对，希望网友纠正。

下面的h(x)是要拟合的函数，J(theta)损失函数，theta是参数，要迭代求解的值，theta求解出来了那最终要拟合的函数来了。其中m是训练集的记录条数，i是参数的个数。

$$h(\theta) = \sum_{j=0}^n \theta_j x_j$$

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

### 1、批量梯度下降的求解思路如下：

（1）将J(theta)对theta求偏导，得到每个theta对应的的梯度

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

（2）由于是要最小化风险函数，所以按每个参数theta的梯度负方向，来更新每个theta

$$\theta_j' = \theta_j + \frac{1}{m} \sum_{i=1}^m (y^i - h_{\theta}(x^i)) x_j^i$$

（3）从上面公式可以注意到，它得到的是一个全局最优解，但是每迭代一步，都要用到训练集所有的数据，如果m很知这种方法的迭代速度！！所以，这就引入了另外一种方法，随机梯度下降。

### 2、随机梯度下降的求解思路如下：

（1）上面的风险函数可以写成如下这种形式，损失函数对应的是训练集中每个样本的粒度，而上面批量梯度下降对应训练样本：

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

$$\text{cost}(\theta, (x^i, y^i)) = \frac{1}{2} (y^i - h_{\theta}(x^i))^2$$

（2）每个样本的损失函数，对theta求偏导得到对应梯度，来更新theta

$$\theta_j' = \theta_j + (y^i - h_{\theta}(x^i)) x_j^i$$

（3）随机梯度下降是通过每个样本来迭代更新一次，如果样本量很大的情况（例如几十万），那么可能只用其中几万样本，就已经将theta迭代到最优解了，对比上面的批量梯度下降，迭代一次需要用到十几万训练样本，一次迭代不迭代10次的话就需要遍历训练样本10次。但是，SGD伴随的一个问题是噪音较BGD要多，使得SGD并不是每次迭代都优化方向。

### 3、对于上面的linear regression问题，与批量梯度下降对比，随机梯度下降求解的会是最优解吗？

（1）批量梯度下降---最小化所有训练样本的损失函数，使得最终求解的是全局的最优解，即求解的参数是使得风险函

（2）随机梯度下降---最小化每条样本的损失函数，虽然不是每次迭代得到的损失函数都向着全局最优方向，但是大的向全局最优解的，最终的结果往往是在全局最优解附近。

### 4、梯度下降用来求最优解，哪些问题可以求得全局最优？哪些问题可能局部最优解？

对于上面的linear regression问题，最优化问题对theta的分布是unimodal，一个peak，所以梯度下降最终最优解。然而对于multimodal的问题，因为存在多个peak值，很有可能梯

### 5、随机梯度和批量梯度的实现差别

2

0

5. 前端开发之旅-zopim在线即时聊天客服(770)

### 评论排行榜

1. Android系统架构(1)

### 推荐排行榜

1. 机器学习-随机梯度下降

( Stochastic gradient descent ) 和 批量梯度下降 ( Batch gradient descent ) (2)

2. 基于云计算感应器的智能盆栽监测系统(1)

以前一篇博文中NMF实现为例，列出两者的实现差别（注：其实对应Python的代码要直观的多，以后要练习多写pyth

```

1 // 随机梯度下降，更新参数
2 public void updatePQ_stochastic(double alpha, double beta) {
3     for (int i = 0; i < M; i++) {
4         ArrayList<Feature> Ri = this.dataset.getDataAt(i).getAllFeature();
5         for (Feature Rij : Ri) {
6             // eij=Rij.weight-PQ for updating P and Q
7             double PQ = 0;
8             for (int k = 0; k < K; k++) {
9                 PQ += P[i][k] * Q[k][Rij.dim];
10            }
11            double eij = Rij.weight - PQ;
12
13            // update Pik and Qkj
14            for (int k = 0; k < K; k++) {
15                double oldPik = P[i][k];
16                P[i][k] += alpha
17                    * (2 * eij * Q[k][Rij.dim] - beta * P[i][k]);
18                Q[k][Rij.dim] += alpha
19                    * (2 * eij * oldPik - beta * Q[k][Rij.dim]);
20            }
21        }
22    }
23 }
24
25 // 批量梯度下降，更新参数
26 public void updatePQ_batch(double alpha, double beta) {
27
28     for (int i = 0; i < M; i++) {
29         ArrayList<Feature> Ri = this.dataset.getDataAt(i).getAllFeature();
30
31         for (Feature Rij : Ri) {
32             // Rij.error=Rij.weight-PQ for updating P and Q
33             double PQ = 0;
34             for (int k = 0; k < K; k++) {
35                 PQ += P[i][k] * Q[k][Rij.dim];
36             }
37             Rij.error = Rij.weight - PQ;
38         }
39     }
40
41     for (int i = 0; i < M; i++) {
42         ArrayList<Feature> Ri = this.dataset.getDataAt(i).getAllFeature();
43         for (Feature Rij : Ri) {
44             for (int k = 0; k < K; k++) {
45                 // 对参数更新的累积项
46                 double eq_sum = 0;
47                 double ep_sum = 0;
48
49                 for (int ki = 0; ki < M; ki++) { // 固定k和j之后,对所有i项加和
50                     ArrayList<Feature> tmp = this.dataset.getDataAt(i).getAllFeature();
51                     for (Feature Rj : tmp) {
52                         if (Rj.dim == Rij.dim)
53                             ep_sum += P[ki][k] * Rj.error;
54                     }
55                 }
56                 for (Feature Rj : Ri) { // 固定k和i之后,对多有j项加和
57                     eq_sum += Rj.error * Q[k][Rj.dim];
58                 }
59
60                 // 对参数更新
61                 P[i][k] += alpha * (2 * eq_sum - beta * P[i][k]);
62                 Q[k][Rij.dim] += alpha * (2 * ep_sum - beta * Q[k][Rij.dim]);
63             }
64         }
65     }
66 }

```

好文要顶

关注我

2

0





Sirius、武...  
关注 - 4  
粉丝 - 3

+加关注

« 上一篇：[机器学习-监督学习应用：梯度下降](#)

» 下一篇：[机器学习-最小二乘法](#)

posted @ 2017-06-02 11:52 Sirius、武... 阅读(11255) 评论

[刷新评论](#) [刷新](#)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

- 【推荐】超50万VC++源码: 大型组态工控、电力仿真CAD与GIS源码库！
- 【推荐】华为云11.11普惠季 血拼风暴 一促即发
- 【拼团】腾讯云服务器拼团活动又双叒叕来了！
- 【推荐】腾讯云新注册用户域名抢购1元起



腾讯云

腾讯云AMD云服务器

节省IT成本30%

1核1G AMD机型0.57元/天起

立即抢购

- 最新IT新闻:
- 腾讯上线音乐短视频应用音兔
  - 亚马逊更新Kindle Paperwhite，带来IPX8级防水和最高32GB空间
  - 从此，天上有一颗星星叫“南仁东星”
  - 特斯拉债务即将到期 银行主动上门资助
  - “DJ” 丁磊推电音业务 情怀还是搅局？
- » 更多新闻...



爱护乳腺, AI不宜迟

英特尔® 用人工智能解决大问题



了解更多 人工智能

技术生态构建 | IT汇算智能 从实践 开始



- 最新知识库文章:
- 为什么说 Java 程序员必须掌握 Spring Boot ？
  - 在学习中，有一个比掌握知识更重要的能力
  - 如何招到一个靠谱的程序员
  - 一个故事看懂 “区块链”
  - 被踢出去的用户
- » 更多知识库文章...