线性回归

===================

一、线性和回归

---》线性：y=a\*x 一次方的变化

---》回归：回归到平均值

（1）简单线性回归

---》即，一元一次方程组

---》复杂线性回归包含多元因素变量，权重可能不同

（2）一元和一次

---》一元指的一个X：影响Y的因素，维度

---》一次指的X的变化：没有非线性的变化

（3）机器学习的目标

---》做机器学习，没有完美解，只有最优解~

---》做机器学习就是要以最快的速度，找到误差最小的最优解！

（4）如何做线性回归

---》最小二乘法？

（5）一个样本的误差：yi^ - yi

---》找到误差最小的时刻，为了去找到误差最小的时刻，需要反复尝试计算a，b

---》根据最小二乘法去求得误差

---》反过来误差最小时刻的a，b就是最优解模型！

（6）多元线性回归

---》本质上就是算法（公式）变换为了多元一次方程组

---》y = w1\*x1 + w2\*x2 + w3\*x3 + ... + wn\*xn + w0\*x0

---》y = wT\*x（一组w的向量转置和一组x的向量求点积）

（7）最大似然估计

---》暂时可以理解为，最大概率估计

---》为什么是求联合似然？

（8）一些术语

---》中心极限定理、概率密度函数、线性回归的损失函数（代价函数）

---》概率密度函数可以基于：均匀分布、正态分布、泊松分布等，但会影响损失函数

2、线性回归的损失函数（代价函数）

---》通过最大似然估计得思想，利用了正太分布的概率密度函数，推导出来了损失函数

===================

二、解析解方法求解线性回归

---》损失函数变形为向量相乘，然后求导，等于0的位置为极值点

---》求得theta的最优解析解：

---》求得的是极小值的前提条件：判定凸函数、两边导数左负右正

---》此种方法虽然可以一步算出结果，但是当数据量大的时候，计算机内存无法完成这样的工作

---》两个方法求解析解

### 方法一，numpy模块，文件名: linear\_regression\_0.py

### 方法二，通过sklearn模块使用LinearRegression（from sklearn.linear\_model import LinearRegression）文件名: linear\_regression\_1.py

====================================

Q：为什么求总似然的时候，要用正太分布的概率密度函数？

A：中心极限定理，如果假设样本之间是独立事件，误差变量随机产生，那么就服从正太分布！

Q：总似然不是概率相乘吗？为什么用了概率密度函数的f(xi)进行了相乘？

A：因为概率不好求，所以当我们可以找到概率密度相乘最大的时候，就相当于找到了概率相乘最大的时候！

Q：概率为什么不好求？

A：因为求得是面积，需要积分，麻烦，大家不用去管数学上如何根据概率密度函数去求概率！

Q：总似然最大和最优解的关系？

A：当我们找到可以使得总似然最大的条件，也就是可以找到我们的DataSet数据集最吻合某个正太分布！即找到了最优解！

Q：何为损失函数？

A：一个模型的损失函数值最小，就对应了模型的最优解！预测最准！

Q：线性回归的损失函数是什么？

A：最小二乘法，MSE（mean squared error），平方均值损失函数，均方误差

Q：线性回归的损失函数有哪些假设？

A：样本独立，随机变量，正态分布

=========================================================

三、梯度下降法

1、梯度下降法

（1）梯度下降法是干嘛的？

---》梯度下降法是一种以最快的速度找到最优解的方法！

（2）梯度下降法流程(重点内容)

1、初始化theta，即一组W（w0...wn）

2、求梯度（导数）gradients

3、调整theta： theta\_t+1 = theta\_t - grad \* learning\_rate

---》learning\_rate是个超参数，太大容易来回振荡，太小步子太短，需要走很长时间，不管太大还是太小，都会迭代次数很多，耗时很长

4、循环往复第2步和第3步，直到迭代收敛，等待grad < threshold（阈值）

（3）两个重要的超参数

---》学习率learning\_rate、预值threshold（θ最小时，g≈0的值）

---------------------------------

2、为什么要推导损失函数导函数？

---》推导线性回归的loss function的导函数，目的是可以更快的求解梯度！

3、批量梯度下降（BGD）

---》grad\_j = (1/m) \* (Xj)^Transpose \* (X\*theta - y)

---》grads = (1/m) \* X^Transpose \* (X\*theta - y)

上面就是批量梯度下降的时候，去求解gradients梯度的公式！

不管是批量梯度下降，还是随机梯度下降，流程里面的1，3，4都是一样的，只有第二步求梯度稍有不同！

----------------------------------

4、随机梯度下降（SGD）

Q：随机梯度下降，怎么随机的呢？

A：其实就是在求梯度的时候，不再用所有的m个样本数据来计算，而是随机的选择一条数据来计算梯度！

Q：随机梯度下降的好处是什么？缺点是什么？

A：在求梯度的时候快，迭代次数有可能更多，最终可能落不到全局最优解上

Q：Mini-Batch GD是什么？

A：就是在求梯度的时候做了一个折中，不用所有的数据，而是随机选择一部分数据来求梯度！

---》上面代码里面除了随机抽取一条数据来求解梯度，还随着迭代次数的增多，不断减小步长（learning\_rate）！

Q：为什么要不断的调整步长？

A：就是为了让越接近最优解的时候，调整的幅度越小，避免来回震荡！

Q：如果我们不人为的调小步长，会不会随着迭代的次数增多，调整的幅度自动减小？

A：调整的幅度取决于谁？却决于学习率和梯度，梯度事实上越接近最优解，梯度的绝对值越小

==============================================

1. 归一化和过拟合

------------------

1、归一化的必要性

Q：为什么要做归一化？

A：只要是基于梯度来进行下降求解最优解，都需要归一化，目的是各个维度梯度可以同时收敛

Q：如果不做归一化，会产生的问题是什么？

A：如果X1<<X2，那么W1>>W2，那么要到达最优解的位置，W1需要调整的幅度>>W2需要调整的幅度！

---》因为X1<<X2，则g1=(y^-y)\*x1 ，g2=(y^-y)\*x2，所以g1<<g2

---》因为g1<<g2，而θ调整的幅度等于：θ\_t+1 - θ\_t = - alpha \* g，所以g越小，调整的幅度就越小

---》总结一下上面的推导：

X1<<X2，W1调整的幅度<<W2调整的幅度，但是W1需要调整的幅度>>W2需要调整的幅度；矛盾就产生了，如果此时不做归一化，去使用梯度下降求解最优解的话，产生的效果，即会是同样的迭代次数下，W2已经调整好了，W1还在慢慢的往前挪。整体看起来，就比先做归一化，再做梯度下降，需要的迭代次数要多了！！！

---------------------------------

2、如何归一化

Q：怎么让多个维度对应的W基本上在同一时刻收敛？

A：对多个维度X来进行统一的归一化，比如说，最大值最小值归一化的方法

Q：何为最大值最小值归一化呢？

A：(X-Xmin)/(Xmax-Xmin)，最大值最小值归一化的特点是一定可以把一列数据归到0到1之间

---------------------------------

3、过拟合

Q：什么是过拟合？

A：拟合过度，用算法生成的模型，很好的拟合了你的训练集数据，但是当来新的数据的时候，比如测试集的数据，预测的准确率反而降低了很多，那这个时候就是发生了过拟合现象

Q：如何防止过拟合呢？

A：防止过拟合，等价于提高模型的泛化能力，或者说白了就是举一反三的能力！提高了模型的容错能力！

============================================

五、岭回归、Lasso回归和ElasticNet的选择

1、线性回归算法选择和正则化

Q：如何在机器学习里面防止过拟合呢？

A：模型参数W个数，越少越好；

模型参数W的值越小越好，这样如果X输入有误差，也不会太影响y预测结果

通过正则化惩罚项，人为的修改已有的损失函数，比如使用L1、L2正则添加到loss func里面去

L1 = n个维度的w绝对值加和

L2 = n个维度的w平方和

让我们的SGD，在找最优解的过程中，考虑惩罚项的影响

Q：当使用惩罚项，会产生什么影响？

A：使用惩罚项，会提高模型的泛化能力，但是因为人为的改变了损失函数，所有在一定程度上牺牲了正确率，即对训练集已有数据的拟合效果。但是没关系，因为我们计算模型目的是对未来新的数据进行预测。在惩罚项里有个alpha，即惩罚项的权重，我们可以通过调整alpha超参数，根据需求来决定是更看重模型的正确率还是模型的泛化能力！

-------------------------------------

2、线性回归算法选择顺序

（1）Ridge Regression (L2正则化)

（2）ElasticNet (即包含L1又包含L2)

（3）Lasso Regression (L1正则化)

Q：正则化L1和L2有什么区别？

A：L1是w绝对值加和，L2是w平方加和。

---》L1的有趣的现象是会使得w有的接近于0，有的接近于1

---》L1更多的会用在降维上面（但其实降维在预处理阶段就好了，所以该算法很少用），因为有的是0有的是1，我们也称之为稀疏编码。

---》L2是更常用的正则化手段，它会使得所有w整体变小

3、各种线性回归算法中超参数注释

Rideg类中，alpha = L2正则的权重

Lasso类中，alpha = L1正则的权重

ElasticNet和SGDRegressor类中，l1\_ration = 损失函数的ρ，系数整体分别表示 L1和L2正则的权重

==============================================

六、多项式回归

情景：当遇到非线性的数据，需要拟合时：方法一，采用非线性模型；方法二，将数据升维成多元线性数据

---》叫回归但并不是去做拟合的算法，而是一种线性模型的扩展

---》PolynomialFeatures是来做预处理的，来转换我们的数据，把数据进行升维！

---》比如：将非线性的y = x^2+x+1曲线，其中x^2预处理时，当成新的特征，当多元成线性处理

Q：升维有什么用？

A：升维就是增加更多的影响Y结果的因素，这样考虑的更全面，最终是要增加准确率！

还有时候，就像PolynomialFeatures去做升维，是为了让线性模型去拟合非线性的数据！

Q：PolynomialFeatures是怎么升维的？

A：可以传入degree超参数，如果等于2，那么就会在原有维度基础之上增加二阶的数据变化！

更高阶的以此类推

Q：如果数据是非线性的变化，但是就想用线性的模型去拟合这个非线性的数据，怎么办？

A：1，非线性的数据去找非线性的算法生成的模型去拟合

2，可以把非线性的数据进行变化，变成类似线性的变化，然后使用线性的模型去拟合，PolynomialFeatures类其实就是这里说的第二种方式

polynomial\_regression.py

上面代码里面其实通过画图让同学们理解，更高阶的数据变化，可以让线性模型拟合的更好，但是也容易过拟合！

==============================================

七、保险的案例：

目的：未来来个新的人，可以通过模型来预测他的医疗花销

所以，就把charges列作为y，其他列作为X维度

Q：为什么每行没有人名？

A：人名不会对最终的Y结果产生影响，所以可以不用

Q：为什么要观测注意数据多样性，采样要均匀？

A：就是因为你要的模型的功能是对任何年龄段的人都有一个好的预测，那么你的模型在训练的时候

读取的数据集，就得包含各个年龄段得数据，而且各个年龄段也得数据均匀，防止过拟合！

Q：什么是Pearson相关系数？（皮尔逊系数）

A：Pearson相关系数是来测量两组变量之间的线性相关性的！Pearson相关系数的区间范围是-1到1之间

如果越接近于-1，说明两组变量越负相关，一个变大，另一个变小，反之如果越接近于1，说明两组

变量越正相关，一个变大，另一个也跟着变大，如果越接近于0，说明越不相关

通过Pearson相关系数，如果发现两个维度之间，相关系数接近于1，可以把其中一个去掉，做到降维！

通过Pearson相关系数，如果发现某个维度和结果Y之间的相关系数接近于0，可以把这个维度去掉，降维！