



装

订

线

本科生毕业论文（设计）

题目**：二手车交易价格分析及异常检测**

学 院 数学与信息科学学院

学科门类 工学

专 业 软件工程

学 号 2016100408

姓 名 艾乐

指导教师 张辉

2020年5月

摘 要

二手车交易价格分析与异常的检测，在分析与检测的过程中，主要是利用了两个常用的方法；一个是统计学的方式，利用二手车的一些数据，如价格、里程、年份、自动挡手动挡等参考因素，来构建我们的一个多维的线性回归模型，但是一维线性回归方程，主要是通过数字化的方式实现的，而对于那些非数字的如手动挡自动挡、车发动机等因素，我们需要对其做一些特殊的处理，这里我们对其进行one-hot编码的处理方式，构建完成我们one-hot编码的格式之后，我们就需要对纯数字化的参考因素做一个线性回归模型，然后利用线性回归方程，得到我们结果，再与我们每个真实价格做对比，相差较大的，我们就可以认为该价格是异常价格；第二种方式就是我们的，局部离群点检测方法，该方法有基于多种形式的，我们使用一种基于距离的离群检测，而LOF(Local Outlier Factor)，局部异常因子，就是一种基于距离的异常检测方法，我们通过获得每一个样本的LOF值，通过比较其与“1”值的大小，来判断该样本的值，是正常的价格，还是异常的价格

关键字：多维线性回归模型；one-hot编码；局部离群点；局部异常因子

Abstract

Used car transaction price analysis and anomaly detection. In the analysis and detection process, two commonly used methods are used; one is a statistical method, using some data of used cars, such as price, mileage, year, automatic Reference factors such as manual transmission are used to build a multi-dimensional linear regression model, but the one-dimensional linear regression equation is mainly realized by digital methods. , We need to do some special processing on it, here we do one-hot encoding processing method, after building our one-hot encoding format, we need to do a linear regression model of pure digital reference factors, Then use the linear regression equation to get our results, and then compare it with each of our real prices. If the difference is large, we can think that the price is an abnormal price; the second way is ours, the local outlier detection method, This method is based on multiple forms, we use a distance-based outlier detection, and LOF (L ocal Outlier Factor), a local anomaly factor, is a distance-based anomaly detection method. We obtain the LOF value of each sample and compare the value with the "1" value to determine the value of the sample. It is normal Price, or abnormal price.

Key words: Multidimensional linear regression model;generator; one-hot encoding;Local outliers;LOF

目 录

[1、引言 1](#bookmark)

[2、线性回归方程 3](#bookmark1)

[3、生成算法与判别算法 4](#bookmark2)

[3.1生成算法 4](#bookmark3)

[3.2判别算法 4](#bookmark4)

[4 生成对抗网络 5](#bookmark5)

[4.1生成对抗网络的原理 5](#bookmark6)

[4.2 GAN 的模型结构 6](#bookmark7)

[4.2.1生成器模型结构 6](#bookmark8)

[4.2.2判别器模型结构 6](#bookmark9)

[4.2.3生成对抗网络模型结构 7](#bookmark10)

[4.3生成对抗网络的评价指标 7](#bookmark11)

[4.4生成对抗网络的优点 8](#bookmark12)

[4.5生成对抗网络的缺点 8](#bookmark13)

[4.6 生成对抗网络的训练 8](#bookmark14)

[5 条件生成对抗网络 10](#bookmark15)

[5.1条件生成对抗网络的原理 10](#bookmark16)

[5.2深度卷积生成对抗网络 10](#bookmark17)

[5.2.1卷积神经网络 10](#bookmark18)

[5.2.2 深度卷积生成对抗网络原理 11](#bookmark19)

[6 实现算法 12](#bookmark20)

[6.1实验原理 12](#bookmark21)

[6.2实验步骤 12](#bookmark22)

[6.2.1实验数据集介绍 12](#bookmark23)

[6.2.2生成器 13](#bookmark24)

[6.2.3判别器 14](#bookmark25)

[6.2.4训练过程 15](#bookmark26)

[6.2.4生成过程 18](#bookmark27)

[6.3实验结果 18](#bookmark28)

[6.4实验分析 20](#bookmark29)

[参考文献 21](#bookmark30)

[致 谢 22](#bookmark31)

1、引言

近年来随着人们生活水平的提高，人们的出行方式也发生了翻天覆地的变化。又以前的步行、自动车，慢慢的变成了电动车、摩托车；而现如今也越来越多的人选择小轿车的出行方式了，且对小轿车的需求量日益增大；但是也有那么一部分的人，选择了相对较便宜的二手车，虽然二手车的价格相对较便宜，但是二手车与崭新出厂的车的最大的区别，就是它的价格定位难以掌握，并且影响二手车价格的因素较多，比如车的里程、上市年份、车发动机、以及挡位方式是手动挡还是自动挡，二手车的价格都将是围绕这些因素来产生偏差的，但是也不乏有个人的人为因素，所以这个时候，就会出现我们所谓的异常价格，价格偏高的，或者说偏低的这些异常值，我们需要对这些价格做一定必要的整理，来做一些价格的分析与异常值的检测；

异常价格，我们称之为离群点，通俗来说，就是离其他的数据点比较远的数据点，这样的价格，或者说点，会严重的影响我们后续对价格的分析结果，甚至产生有误导的错误的分析结果，对于二手车来说，其有价格（price）、里程（mileage）、上市年份（year）、档次（trim）、引擎缸数（engine）、换挡方式（transmission）这些影响因素来判断该车的数据是否为异常的数据；而我们的异常值检测方法也有很多种，从统计学的角度出发的话，我们可以利用线性回归模型【】的方式，在上述众多的影响因素中，不乏有一些非数字型的数据，但是在做线性回归模型方程的时候，我们只能出数字型的数据下手，但那些非数字型的因素又不能忽略，所以我们可以采用一种叫做one-hot【】的编码方式对非数字型的因素进行数字化，这种编码方式主要是采用N位状态寄存器进来对N个状态进行编码，每个每个状态都有他独立的寄存器位，并且任意位置有且只有一位有效，他们以分类变量作为二进制向量表示，首先将分类值映射到整数值，然后每个整数被表示为二进制向量，除了帧数的索引之外，其他的都是零值，而它被标记为1；这样，我们就将非数字化的因素转化为数字化了，由于影响因素过多，所以这是一个多维的线性回归模型方程，当我们得出一个关于价格为因变量，而其他影响因素为自变量的线性回归方程后，我们通过方程的方式重新计算一个价格，然后与我们数据中的价格作为一个对比，当二者相差较大时，我们有理由的认为，该价格是异常值，即该数据样本是异常值；而另一种方法是利用一种局部一场因子（LOF）【】这种概念来展开的。异常检测的实质无非就是寻找观测值和参考值之间有意义的偏差，因为某些原有，我们的样本数据会有一些异常的值，我们不可能对其做一个人工检测，所以利用算法解决，而离群点检测是异常检测中最常用的方法之一，离群点检测的主要目的就是为了检测那些与正常数据差别较大的异常数据；而我们的算法也有分类。基于统计的离群检测、基于聚类的离群检测、基于分类的离群检测、基于距离的离群检测、基于密度的离群检测和基于信息熵的离群检测方法，而LOF则是基于距离的离群检测方法之一。我们通过点与其旁边局部的点通过他们的第K距离、距离领域、可达距离等数据，来得出我们的局部可达密度；从而获取我们的局部异常因子，通过该因子与常量值“1”，的大小来判断该点是不是异常点；总结来说，就是比较每个点和其领域点的密度来判断该点是不是异常点，如果点的密度月底，越可能被认为是异常点，而所谓的密度，都是通过距离来计算的，点之间的距离越远，密度越低；距离越近，密度越高，且这些数据都是通过第k领域来计算，即该点的旁边局部点而得出，而不是全局计算，所以称之为“局部”异常因子；所以即使数据都是紧凑的那种，还是散漫的那种；我们通过判断密度来判断，二者都可以被认为是正常的样本数据；

2、one-hot编码

2.1 概念

one-hot编码又被称为有效编码，主要是采用N位寄存器来对N个状态进行编码，每个状态都有独特的寄存器位且在任意时候有且只有一位有效；

在实际的应用任务中，有些样本的特征有时候并不是总是连续值，而是一种分类值，如二手车的样本值中，引擎缸数、档次等信息，就是一种分类值，这个时候我们就需要对其特征进行数字化；拿引擎缸数和档次为例子如下：

* 引擎缸数（engine）：[4 Cyl；4杠,6Cyl：6杠]；
* 档次（trim）：[exl：高配款且带皮革内饰，ex：高配款，lx：低配款]

我们就需要对这种分类值的特征进行one-hot编码数字化

2.2 处理方式

原始数据如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Price | Mileage | Year | Trim | Engine | transmission |
| 0 | 14995 | 67692 | 2006 | ex | 4Cyl | Manual |
| 1 | 11988 | 72728 | 2006 | ex | 4Cyl | Manual |
| 2 | 11999 | 80313 | 2006 | lx | 4Cyl | Automatic |
| 3 | 12995 | 86096 | 2006 | lx | 4Cyl | Automatic |
| 4 | 11333 | 79607 | 2006 | lx | 4Cyl | Automatic |
| 5 | 10067 | 96966 | 2006 | lx | 4Cyl | Automatic |

Java one-hot编码代码：

**private** List<OneHotSample> readFile2(String path) **throws** Exception {  
 InputStream is = **new** FileInputStream(path);  
 List<OneHotSample> list = **new** ArrayList<>();  
 **try**{  
 *//换成你的文件名* BufferedReader reader = **new** BufferedReader(**new** FileReader(path));  
 reader.readLine();*//第一行信息，为标题信息，不用* String line = **null**;  
 **while**((line=reader.readLine())!=**null**){  
 *//CSV格式文件为逗号分隔符文件，这里根据逗号切分* String item[] = line.split(**","**);  
 OneHotSample sample = **new** OneHotSample();  
 sample.setPrice(Integer.*valueOf*(item[0]));  
 sample.setMileage(Integer.*valueOf*(item[1]));  
 sample.setYear(Integer.*valueOf*(item[2]));  
 **switch** (item[3]) {  
 **case "ex"** :  
 sample.setTrimEx(1);  
 **break**;  
 **case "lx"**:  
 sample.setTrimLx(1);  
 **break**;  
 **default**:  
 sample.setTrimExl(1);  
 }  
 **switch** (item[4]) {  
 **case "4 Cyl"** :  
 sample.setEngine4Cyl(1);  
 **break**;  
 **default**:  
 sample.setEngine6Cyl(1);  
 }  
 **switch** (item[5]) {  
 **case "Automatic"** :  
 sample.setTransmissionAutomatic(1);  
 **break**;  
 **default**:  
 sample.setTransmissionManual(1);  
 }  
 list.add(sample);  
 }  
 } **catch** (Exception e){  
 System.***out***.println(**"文件转换出错:"** + e);  
 } **finally** {  
 **if** (is != **null**) {  
 **try** {  
 is.close();  
 } **catch** (Exception e){  
 System.***out***.println(**"输入流关闭异常"**);  
 }  
 }  
 }  
 **return** list;  
}

经过one-hot编码后（由于空间问题，价格price列未展示）：

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Mileage | Year | Trim-ex | Trim-lx | Trim-exl | Engine-4Cyl | Engine-6Cyl | Tr-manu | Tr-auto |
| 0 | 67692 | 2006 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 1 | 72728 | 2006 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 2 | 80313 | 2006 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 3 | 86096 | 2006 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 4 | 79607 | 2006 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| 5 | 96966 | 2006 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |

从编码结果来说，对于如trim这种非数字化的特征，我们将其one-hot编码后，就变成了数字化，且在trim的属性中，有且只有一位是有效的，如 1 0 0，010；这个时候，我们就将影响因素数值化了，后续对其进行构建线性回归模型；

3、线性回归模型

3.1一维线性回归方程

对于表面上的线性，我们给定一组输入值x和输出值y；（x,y），我们假设他们的关系是线性的关系，这时就可以得到关系 y = kx + b，只要我们得到k和b的值，就可以得到关于这个关系的方程，即得到我们的线性。

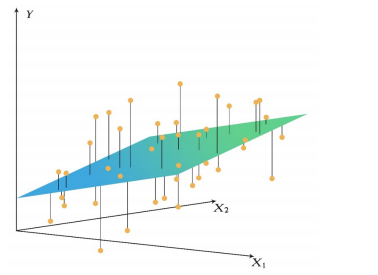


我们所有的（x，y），都是围绕这条线分布的，这时，我们通过输入的自变量x，可以一个值y，将该得到的y和实际值y比较，当二者的差距在我们置信水平范围内（误差范围），我们就有理由的认为该组数据是正常值，否住认为是异常值；

3.2多维线性回归方程

上述一维的线性回归方程，其实是相当于多维线性回归模型中特殊在一种情况，即自变量只有一种的情况；线性回归模型真正的含义是：利用线性函数对一个或多个自变量（x或者x0、x1、x2、x3...xn）和因变量（y）之间的关系进行拟合的模型，即线性回归模型构建成功后，这个模型表现线性函数的形式，所以说一维的，就是特殊的多维线性回归方程；

* 自变量只有一个：y = f(x) = a + bx;
* 自变量多个：y = f(x1,x2,,x3,...,xn) = a + b1x1 + b2x2 + b3x3 + ...+ bnxn;



如果特征属性是一维的，那么他就是一条直线，如果是二维的就是一个平面，三位的就是一个体，以此类推下去。

上述多元线性回归方程中，这些x和y的关系整体的图像还并不可以完全的满足我们任意两点之间的关系的，但是这条直线又是我们综合所有的点里面最合适的用来描述他们共同特征的直线了，毕竟它到我们所有点的距离的和是最小的即是误差最小的所以我们的多元线性回归模型方程的表达式又可以写成：

y = w0x0 + w1x1 + w2x2 + ... + wnxn

由于我们无法确定多元线性回归模型函数中预测到的回归函数是否经过原点，于是我们在多元线性回归模型函数中，就需要保留一项常数项来作为我们的截距，于是就有了新的多元线性回归模型函数：

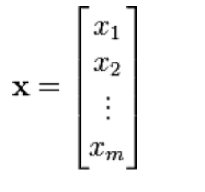
y = w0 + w1x1 + w2x2 + ... + wnxn

当我们没有w0这一项的时候，我们的多元线性回归模型函数就是一条完全由n+1个自变量构成且经过坐标原点的图像的函数，这将导致我们一直用一个经过原点的图像来概括或者说描述一些散列点的分布情况，这显然是不可理的，增大了局限性，这将造成我们预测出的结果的函数准确率将会大幅度的下降。在这里我们默认x0 = 1而不是x2或者其他的x，是因为我们的目标就是为了让函数y = w0 + w1x1 + w2x2 + ... + wnxn是一个包含常数项的线性回归函数模型，无论如何的去选取这个x都是可以的，放在第一个位置，这样便于理解。

4 多元线性回归的推导过程以及求解

4.1向量表示形式

向量就是一个数组，如[1,2,3,4,5]这是一个有五个元素的向量，向量有行向量和列向量之分，行向量就是前面的那种，数字横向排列的：X = [1,2,3,4,5]，而列向量的话，就是数字是竖着排列的，如图：



现在，我们假设有这么一个向量W = [w1,w2,w3,...,wn]为行向量，另一个列向量

X = [x1,x2,x3,...,xn]，行向量和列向量相乘的法则就是对位相乘且相加，结果就会得到一个实数，这就符合我们的预测结果等于y了，所以我们就可以将表达式写成 y = W \* X了。

在这种情况下，多元线性回归模型函数 y = w0 + w1x1 + w2x2 + ... + wnxn，方程就化简成为y = W \* X了，但是如果两个向量中，一个是行向量，一个是列向量的话，很容易就混淆了，所以我们就直接规定我们的W和X都是列向量，这时表达式就可以改写成WT（列向量的转置，即WT，所以WT是行向量）与列向量相乘，最后的最终结果的线性回归模型方程的表达式就变为：

y = WT \* X；但是WT也经常用θT表示

如果W和X都行向量的话， y = W \* XT也是和前面一样的，只是单纯的为了统一表达式，而选择的一种形式而已，二者并没有差别

4.2最大似然估计

最大似然估计就是最大可能性估计的意思，主要内容为：如果事件A和事件B相互独立，于是我们可以说事件A和事件B同时发生的概率会满足一个公式

P(A,B) = P(A) \* P(B)

P(X)就表示事件X发生的概率；

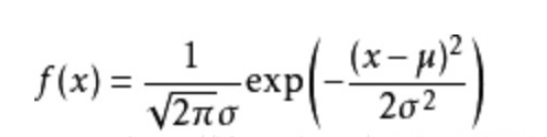
所谓两个独立的事件的意思就是：这两个事件不相关，即A事件发生与否，对B事件的发生与否完全没有影响，即两事件完全不存在关系；这个时候我们就可以称这两个事件是相互独立的；我们使用多元线性回归模型函数的目的就是为了总结一些不相关因素的规律，总结相互独立事件发生的概率也就可以认为是在总结所有相互独立事件同时发生的概率，当所有的事情发生的概率越大时，那么我们预测到的规律就会变得更加的准确。当影响的因素或者说观察到的维度更多的话，则对预测到的维度发生的机率就会越大，也就是说我们的结果更准确，根据最大似然估计表达式：

P(y) = P(x1,x2,x3,..xn.) = P(x1) \* P(x2) \*P(x3) ...P(xn)

当我们所有的事件发生的概率最大时，就可以认为所得函数最符合这些独立事件发生的实际规律，于是我们就把所有样本点的基本分布规律转变成了一个求最大似然估计P(y)的概率,即P(x1,x2,x3...xn) = P(x1) \* P(x2) \* P(x3)...\* P(xn)发生的最大概率。

4.3 概率密度函数

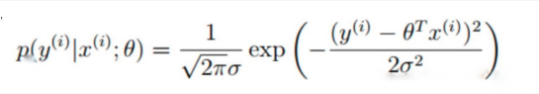
虽然我得到了分布规律的最大似然值，但是在数学当中，其实并没有某种方法来让我们直接求得在何种情况下几个独立事件同时发生的最大的概率，于是我们又引入了概率密度函数；但是在此之前还有一个重要的概念就是：一个随机的变量发生的概率一定会符合正态分布或者说高斯分布，而高斯分布的概率分布密度还是高斯分布，公式：



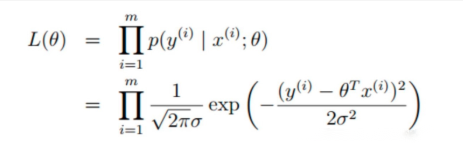
在这个公式中，x就是我们的实际值，μ就是我们的预测值，所以在多元线性回归模型函数中，x就是实际的y，而μ就是我们的θT \* X；前面说到我们需要总结的事件是相互独立的事件，于是这里的每个事件就都是一个随机的事件，或者称他们为随机变量，所以我们最后总结的每个事件的发生概率都将会符合我们的高斯分布；

概率密度函数，就是某个事件发生的概率有多大，当将事件x带入到上述的公式中得到的值越大时，就证明该事件发生的概率越大，但是又值得注意的是，得到的并不是该事件发生的概率，而只是让我们知道该高斯分布公式的值同该事件发生的概率是个正相关而已；

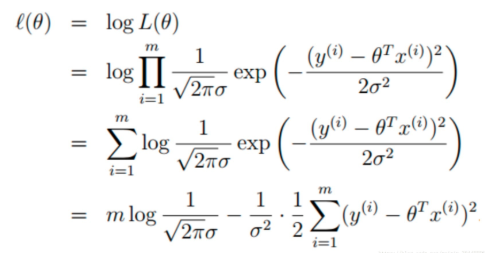
如果将y = θT\*T中的每个自变量x带入到这个公式，将得到如下函数



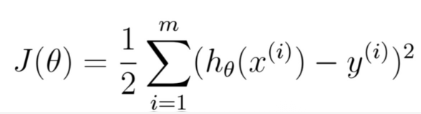
求得所得的独立事件的发生概率最大就是求得左右独立事件概率密度函数结果的乘机的最大值，于是可以得到如下公式：



求L(θ)最大的时候，W的值，则能总结出所有的独立事件符合的规律，需要对上述公式进行求解，因为需要求得的是该函数在什么情况下函数所得到的值最大，而不是求该函数的所有解：



将函数进行一个Log计算，这样将连乘化解成为了连加,这样便于后续的计算，在该公式中，m就是样本的个数，π和σ都是常数，不会影响上述表达式的大小，于是去掉所有的常数项后得到的新的公式如下：



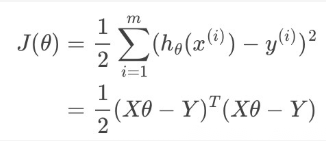
上述得到的公式实质上是一个常数减去减去该公式，所以求概率密度函数的最大值就相当于求这个公式的最大值，而这个公式又是一个数的平方，在数学上，我们称其为最小二乘公式，所以，我们的多元线性模型回归函数的本质就是一个最小二乘。

4.4 多元线性模型回归函数解析解表达

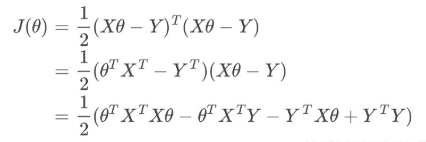
在多线线性回归模型函数中有种解析解求解法，即我们可以通过一个指定的公式得到方程的解，当我们把函数方程的参数带入到公式中，计算公式得到的结果就可以得到方程的解，而不需要一步一步的去化简从而求解；最简单且最熟悉的莫过于学过的一元二次方程的解析解了我们只需要将上述的a、b、代入就可以得到方程的解；所以我们需要得到最小二乘函数的解析解表达方式。

求某个函数方程在它某一点上面的导数，就相当于在该方程函数上，过该点多做切线得到的斜率，而该点的导函数就是切线的函数，如果能够找到函数图像上面切线是零的点的话，就可以得到函数的解，而函数上某点求导就相当于通过该点在图像上做切线，做出来得到的切线就是我们所求导得到的导函数的图像，即切线的函数就是对函数方程求导得到的一个导函数，则我们只要获取到该导函数为0的点，就可以得到该图像的解；于是我们对最小二乘函数求导，然后让导函数的结果等于0；最后得到的结果就是最小二乘函数的解；过程如下：

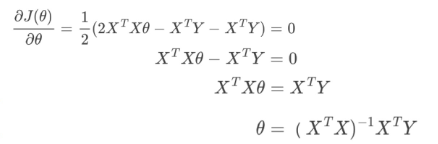
首先对最小二乘做一个变形，变成一个矩阵的表达形式：



展开矩阵函数：



对展开的矩阵J(θ）求导，然后让导数等于0



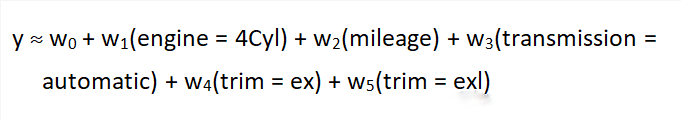
最后我们就求得了解析解的表达式为：



其中θ就是我们线性回归模型方程的系数向量，而X,Y则是样本的影响因素值的向量和样本结果值的向量，即二手车中，X是非价格列，而Y就是样本的价格，当将所有的样本值代入后，就可以得到向量θ的值，即线性回归模型方程的系数向量。

4.5 多元线性函数系数求解

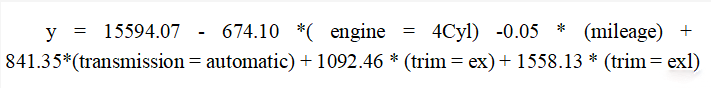
得到多线线性模型函数的解析解的表达式，后续我们只需要将one-hot编码后的样本数据带入即可，但是在样本数据当中，某些特殊因素会影响求解过程导致无法求解，如年份一列，样本值中年份列的所有值都是一样的，这样导致由样本组成矩阵会有线性相关侧从而导致该举证是一个奇异矩阵而无法求逆，最后无法得到系数的解，于是我们需要删除这些特殊的列，结果最后筛选我们得出线性回归模型的函数表达式如下



经过一系列函数求解后得（结果都是保留两位小数）：

* W0 = 15594.07
* W1 = -674.10
* W2 = -0.05
* W3 = 841.35
* W4 = 1092.46
* W5 = 1558.13

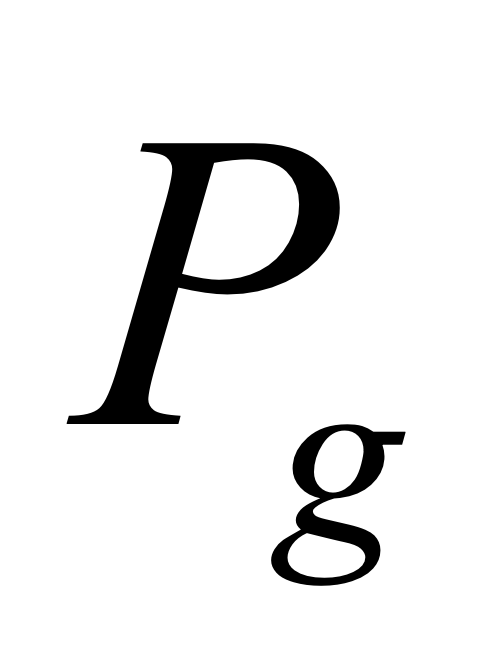
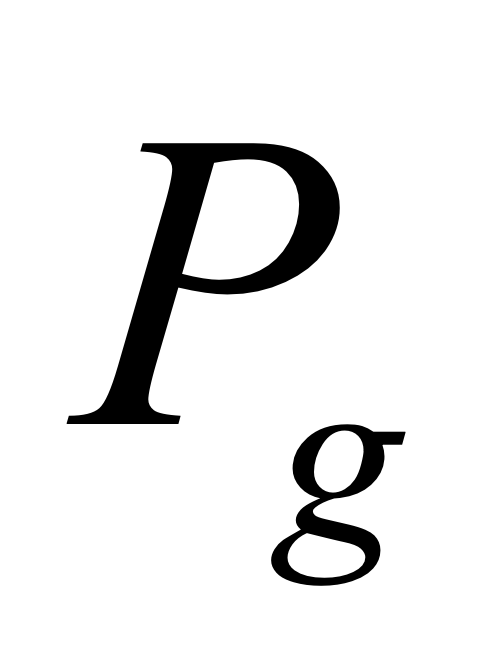
于是我们的关于该样本的线性回归函数的表达式为：



代码如下  
**import** org.ujmp.core.DenseMatrix;  
**import** org.ujmp.core.Matrix;  
**import** java.util.List;  
  
**public class** LinearRegression {  
  
 Matrix **denseX**;  
 Matrix **densext**;  
 Matrix **denseY**;  
 **private** List<OneHotSample> **oneHotSamples**;  
  
 **public** LinearRegression() **throws** Exception {  
 UsedCarNnalysis usedCarNnalysis = **new** UsedCarNnalysis();  
 **oneHotSamples** = usedCarNnalysis.readFile2(**"汽车价格离群值检测/dataset/accord\_sedan\_training.csv"**);  
 *//获取输入训练数据文本的 行数* **int** rowoffile=**oneHotSamples**.size();  
 *//获取输入训练数据文本的 列数* **int** columnoffile = OneHotSample.**class**.getDeclaredFields().**length**;  
 *//去掉某些会造成线性相关的列* **denseX** = DenseMatrix.***Factory***.zeros(rowoffile, columnoffile - 4);  
 **denseY** = DenseMatrix.***Factory***.zeros(rowoffile, 1);  
 initMatrix();  
 }  
 **private void** initMatrix(){  
 **for** (**int** i = 0; i < **oneHotSamples**.size();i++) {  
 **denseX**.setAsDouble(1.0, i, 0);  
 **denseX**.setAsDouble(**oneHotSamples**.get(i).getEngine4Cyl(), i, 1);  
 **denseX**.setAsDouble(**oneHotSamples**.get(i).getMileage(), i, 2);  
 **denseX**.setAsDouble(**oneHotSamples**.get(i).getTransmissionAutomatic(), i, 3);  
 **denseX**.setAsDouble(**oneHotSamples**.get(i).getTrimEx(), i, 4);  
 **denseX**.setAsDouble(**oneHotSamples**.get(i).getTrimExl(), i, 5);  
 **denseY**.setAsDouble(**oneHotSamples**.get(i).getPrice(), i, 0);  
 }  
 **densext** = **denseX**.transpose();  
 }  
  
 **public void** result(){  
 Matrix denseXtX = **densext**.mtimes(**denseX**);  
 Matrix denseXtXInv = denseXtX.inv();  
 Matrix denseXtXInvXt = denseXtXInv.mtimes(**densext**);  
 Matrix denseXtXInvXtY = denseXtXInvXt.mtimes(**denseY**);  
 System.***out***.println(**"常数项w0："**+ denseXtXInvXtY.getAsDouble(0, 0));  
 System.***out***.println(**"Engine4Cyl w1："**+ denseXtXInvXtY.getAsDouble(1, 0));  
 System.***out***.println(**"Mileage w2："**+ denseXtXInvXtY.getAsDouble(2, 0));  
 System.***out***.println(**"TransmissionAutomatic w3："**+ denseXtXInvXtY.getAsDouble(3, 0));  
 System.***out***.println(**"TrimEx w4："**+ denseXtXInvXtY.getAsDouble(4, 0));  
 System.***out***.println(**"TrimExl w5："**+ denseXtXInvXtY.getAsDouble(5, 0));  
 }  
  
 **public static void** main(String[] args) **throws** Exception {  
 LinearRegression m = **new** LinearRegression();  
 m.result();  
 }  
}

4.6 离群值检测

在本样本案例中，我们选择置信水平为0.95，即认为只要超过了95%百分位数的样本数据，都被认为是异常数据。

要负责对输入数据进行判断，识别是生成器生成的数据还是真实数据【3】。当训练结束后，生成器可以根据输入的随机噪声生成十分真实清晰的图像。生成器学习数据的分布情况是该网络的关键。首先给出数据集X的分布参数,以及输入先验噪声变量。用 表示潜在空间到数据空间的映射，输出单个标量作为判断x来自真实数据而不是的概率。训练判别器以最大化正确标注实际数据和生成样本的概率。训练生成器用于最小化。尽量减少判别器得出正确答案的概率。可以将这样的训练任务看作具有值函数的极大极小博弈

4.2 GAN 的模型结构

4.2.1生成器模型结构

在gan的基础上，通过增加条件将原有的模型扩展为条件模型【4】。条件模型中生成器和判别器都依赖增加的条件y，y可以是类标签或者其他模式的数据。相对生成对抗网络，生成器和判别器的输入信息都增加了来自条件y的限制。在生成器中，之前的输入噪声pz(z)和y以联合隐藏表示的形式组合在一起，在判别器中，x和y表示为输入，并表示为一个鉴别函数，博弈的目标函数为：

 （3）

其主要的网络结构如下：

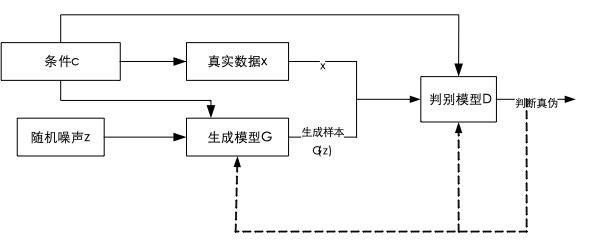


图5-1 条件对抗网络结构

5.2深度卷积生成对抗网络

5.2.1卷积神经网络

近两年卷积神经网络得到了深入的研究。卷积神经网络属于机器学习中的监督学习，主要应用于计算机视觉领域。卷积神经网络主要采用卷积计算对图像的特征进行提取，通过将采样层对减小训练参数，多层卷积池化操作能够对特征信息进行深入解析。卷积神经网络主要包括数据输入层、卷积层、激活层、池化层和全连接层。此外，卷积神经网络还具有参数共享、局部连接、多核卷积等特点，这些都是卷积神经网络能够表现效果好的关键。目前表现效果良好且常用的卷积神经完网络包括LeNet，其最早用于数字识别，ALexNet在2013年的LILSVRC比赛上取得了惊人的成绩，他比LeNet层次更深，用多层小卷积层叠加替换单大卷积层，VGGNet在多图像转化学习问题上效尤其突出。

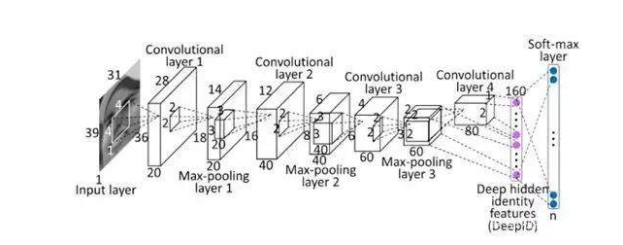


图5-2 卷积神经网络

5.2.2 深度卷积生成对抗网络原理

卷积神经网络在监督学习上得到了广发的应用，相比之下，卷积神经网络在无监督学习上的应用给很少。深度卷积生成对抗网络的诞生弥补了CNNs在监督学习和非监督学习方面缺陷。深度卷积神经网络具有一定的架构约束，结合了深度神经网络和生成对抗网网络，使得该算法既能应用于半监督学习又能应用于监督学习，且表现出了不错的效果。介绍反卷积神经网络：深度卷积生成对抗网络主要有以下几个特点：对于所有的判别网络，将采样层都使用步幅卷积，对于所有的生成网络，都采用微步幅度卷积，二者进行替换。为了确保输入随机噪声的区间，生成网络和判别网络的输入数据都要进行归一化批处理。

对于更深的架构移除全连接隐藏层，减少训练参数，在生成网络输出层使用tanh激活函数，其余所有层使用RelU激活函数，判别网络的所有层上都使用LeakyReLU激活函数。

图5-3 深度卷积生成对抗网络的结构

6 实现算法

6.1实验原理

本实验主要采用深度卷积生成对抗网络算法实现对keraas的Fashion-MNIST数据集中的灰度图像进行训练，生成10终类别的图像

现简单，计算高效，对内存需求少,参数的更新不受梯度的伸缩变换影响,此外超参数具有很好的解释性，且通常无需调整或仅需很少的微调，更新的步长能够被限制在大致的范围内， 能自然地实现步长退火过程，适用于不稳定目标函数，适用于梯度稀疏或梯度存在很大噪声的问题综合。损失函数采用的是binary\_crossentropy。 上述激活函数采用的是sigmoid函数，对于单个神经元的输入输出，其代价函数可以表示为：

 （7）

 （8）

 （9）

其中a为经过激活函数后的实际输出，y是期望输出，z是神经元经过网络处理后得到的值。在训练神经网络过程中，通过梯度下降算法来更新w和b，因此需要计算代价函数对w和b的导数，不断更新w、b的值,公式如下：

 （10）

 （11）

根据上述sigmoid函数图像可以，在z取大部分值时会很小，因此w和b的更新速度非常慢，交叉熵代价函数克服了这一缺点，

与方差代价函数一样，交叉熵代价函数同样有两个性质，第一非负性，当真实输出a与期望输出y接近的时候，代价函数接近于0。第二它可以克服方差代价函数更新权重过慢的问题。对权重和偏置分别求偏导数：

 （12）

 （13）

可以看出导数中没有σ‘(z)这一项，所以权重的更新只受σ(z)−y影响，即受误差的影响。所以当误差大的时候，权重更新就快，当误差小的时候，权重的更新就慢。主要代码如下：

g.compile(loss='binary\_crossentropy', optimizer="Adam")

d\_on\_g.compile(loss='binary\_crossentropy', optimizer="Adam")

d.compile(loss='binary\_crossentropy', optimizer="Adam")

判别器D采用常规的批量训练方法,由于生成器和判别器是相互博弈的，所以判别器的训练过程中加入了对抗过程，该过程采用将生成器的生成的数据手动标记为假数据的方式对判别器D进行训练。生成器的训练也依赖于判别器，生成器首先根据输入噪声生成图像，然后输入到判别器中，由判别器给出训练损失函数的值，并据此对生成器参数进行调整，该过程对生成器的训练是单独的。

6.2.4生成过程

随机生成100个数据作为输入噪声，这些值都在（-1，1）之间， 然后和条件数据进行拼接，将拼接后的数据输入生成器中，生成器经过处理生成图片，将图片送入判别器，判别器对图片进行判断，给出各个种类的概率，取结果最优值作为最终结果。

6.3实验结果

针对输入图片中的服装类型，本文主要输入了10中图片，实验结果如图：

6.4实验分析

从实验结果中我们可以看出能够很好的生成图片，且生成数据样本与真实数据样本十分逼近。同时表明在监督学习和生成模型上对抗网络可以为图像学习到很好的特征表示。

DCGAN确实在一定程度上提高了GAN训练的稳定性(不太容易发生mode collapse的情况),而且生成的图片质量如果数据集数量较高、训练充分，还是很不错的。

我认为将这个框架扩展到其他领域,如视频（帧预测）和音频（预训练功能的语音合成）应该是非常值得进一步研究的。

参考文献

[1] 生成式对抗网络GAN的研究进展与展望[J]. 自动化学报, 2017(3).

[2] Ledig C , Theis L , Huszar F , et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network[J]. 2016.

[3] Santana E, Hotz G. Learning a Driving Simulator[J]. 2016.

[4] 唐贤伦, 杜一铭, 刘雨微, et al. 基于条件深度卷积生成对抗网络的图像识别方法[J]. 自动化学报, 2018, v.44(05):90-99.

[5] Shmelkov K, Schmid C, Alahari K. How good is my GAN?[J]. 2018.

[6] Goodfellow I J, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[C]// International Conference on Neural Information Processing Systems. 2014.

[7] 生成式对抗网络GAN的研究进展与展望[J]. 自动化学报, 2017(3).

[8] 杜秋平, 刘群. 基于图像云模型语义标注的条件生成对抗网络[J]. 模式识别与人工智能, 2018, v.31；No.178(04):89-98.

[9] 蒋芸, 谭宁, 张海, et al. 基于条件生成对抗网络的咬翼片图像分割[J]. 计算机工程.

[10] 刘玉杰, 窦长红, 赵其鲁, et al. 基于条件生成对抗网络的手绘图像检索[J]. 计算机辅助设计与图形学学报, 2017(12):186-192.

[11] 基于条件深度卷积生成对抗网络的图像识别方法\_唐贤伦

[12] 佚名. 基于条件生成对抗网络的咬翼片图像分割[J]. 计算机工程, 2018.

[13] Odena A . Semi-Supervised Learning with Generative Adversarial Networks[J]. 2016.

[14] Radford A , Metz L , Chintala S . Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks[J]. Computer Science, 2015.

[15] He K, Zhang X, Ren S, et al. Identity Mappings in Deep Residual Networks[J]. 2016.

致 谢

我首先要感谢我的论文指导老师河北大学数学与信息科学学院的王涛老师。王涛老师对我论文的研究方向做出了指导性的意见和推荐，在论文撰写过程中及时对我遇到的困难和疑惑给予悉心指点，提出了许多有益的改善性意见，投入了超多的心血和精力。我要对王涛老师对我的帮忙和关怀表示诚挚的谢意!同时，还要感谢河北大学数学与信息科学学院软件工程专业的授课老师们和所有同学们，大家在学习中互相学习，互相帮忙，共同度过了一段完美难忘的时光。

此外，还要感谢朋友以及同学们在论文编写中带给的大力支持和帮忙，给我带来极大的启发。也要感谢参考文献中的作者们，透过他们的研究文章，使我对研究课题有了很好的出发点。

　　最后，谢谢论文评阅老师们的辛苦工作。衷心感谢我的家人、朋友，以及同学们，真是在他们的鼓励和支持下我才得以顺利完成此论文。