**智能风控学习笔记六-WOE编码和IV值**

大家好，这里是Python知识学堂。上次推文我们学习了数据编码的方式，具体内容可以从这里查看：

[http://mp.weixin.qq.com/s?\_\_biz=MzU1MzEzOTc4Nw==&mid=2247493371&idx=1&sn=355496516264991a3666dc7d5d937cd7&chksm=fbf5cddfcc8244c9133623d82a4b34c78ef5628ca489e6e05b1650ceb2e937451b068d9f0ba6#rd](http://mp.weixin.qq.com/s?__biz=MzU1MzEzOTc4Nw==&mid=2247493371&idx=1&sn=355496516264991a3666dc7d5d937cd7&chksm=fbf5cddfcc8244c9133623d82a4b34c78ef5628ca489e6e05b1650ceb2e937451b068d9f0ba6" \l "rd)

这次的推文我们继续来讨论一下变量的编码WOE编码，并使用Python代码来进行实现。

**一、数据分箱**

在介绍WOE之前我们先来看看什么是数据分箱。

在给学生成绩打分的时候，我们经常会使用优秀、良好、及格和不及格四种方式来进行划分。这四种打分类别就包含了0-100的情况（假设值百分制）。很显然我们就做了一个分箱的操作。

在建立评分卡的时候，我们通常需要将数值型变量转换为类别型变量。拿用户年龄来说，我们可以将年龄划分为[≤25],(5-35],(35-45],(45-55],(55-65],(>65]等几种类别，这时候我们就把数值变成了类别。跟学生成绩划分等级一样，这种把数值型变量转换为类别变量的过程我们叫做分箱（binning）。

更多数据分箱的知识将在之后的文章中详细的介绍。

**二、什么是WOE编码**

WOE编码全称是Weight of Evidence，即证据权重，其是一种有监督的编码方式,有监督的意思是该编码过程中使用了目标列(traget)的数据。WOE编码既可以对离散变量进行编码，也可以对分箱后的数据（连续变量分箱）进行编码。

这里我们先给出WOE的公式：



**解释一下其中符号的含义：**

：表示某一个变量（或者叫特征）的第个分箱，比如说年龄分箱的某一个类别：[≤25]

：表示在对应的分箱下坏样本的数量

：表示在对应的分箱下好样本的数量

：表示坏样本的总数

: 表示好样本的总数

我们假设一个样例数据，然后进行解释这样比较容易理解了：（假设我们对年龄分为四个箱，相应的分箱中好坏样本数量如下），根据好坏样本的数量我们计算出好坏样本的占比：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Good | Bad | Good\_rate | Bad\_rate |
| 分箱1 [≤25] | 80 | 20 | 80% | 20% |
| 分箱2 (25,50] | 95 | 15 | 95% | 15% |
| 分箱3 (50,75] | 80 | 20 | 80% | 20% |
| 分箱4 (>75) | 75 | 25 | 75% | 25% |
| Total | 320 | 80 | 80% | 20% |

基于上述表格的数据我们来计算WOE:

分箱1：

分箱2：

分箱3：

分箱4：

**这样，年龄的四个区间我们就可以使用对应的来进行替换了，从一定角度上来说我们就将一个离散变量转换为一个数值变量，完成了“年龄”特征的WOE编码工作。**

聪明的小伙伴已经发现我们上述的案例是对“年龄”特征进行了分箱操作，即分箱后再进行WOE的编码。如果原始变量是离散变量比如“性别”，则直接进行统计编码即可，不需要分箱的操作。

通过计算分箱的值，我们可以看出：

1. WOE越大则坏样本的占比就越高；
2. 当前分箱的比例和整体比例相等时WOE为0；
3. 当分箱的比例小于样本整体比例时WOE为负；
4. 当前分箱的比例大于整体比例时WOE为正。

从中我们可以得到以下的结论：

WOE的实质是当前分箱中好坏样本(也即客户)数占总体好坏样本数在比例上的差异。

**三、为什么要进行WOE编码**

首先，WOE的主要作用是描述预测变量与目标变量之间的关系。在评分卡中对编码进行WOE编码不一定能够提升模型的性能，毕竟也就是一个编码的操作，还要考虑特征重要性。

通过WOE转化，我们可以提升数据的可理解性，这对可解释性要求比较高的模型来说是至关重要的。

引入风控中最常使用的模型logistic来进行解释：



可以看出这个逻辑回归（分类）模型，一旦其中的某一个变量发生变化则相应的模型就会发生改变。假设表示的是用户的年龄，则当这个变量前面的参数为正值的时候，显然不会因为年龄越大逾期的概率就越高，毕竟在实际中，年龄和违约率并不是线性相关的。这也是要进行WOE编码的原因。

**那么进行WOE编码的好处有哪些呢？**

（1）增加了变量的可解释性，很显然我们对变量考虑到了其每一个可能的取值（连续变量进行分箱）

（2）异常值不敏感（也可处理缺失值）。如果对出现的异常值进行WOE编码就会转换为正常值，很多频次较少的变量（比如说借贷中客户年龄较小分箱）也可以通过WOE转换进行合并。

（3）可以指示自变量对应变量的预测能力，**样本的概率值与WOE值有着密切的关系(不在论证可参考其他教程)**

**四、IV值**

说到WOE的时候就不得不说IV值了。

IV的全称是Information Value，中文意思是信息价值，或者信息量。**信息值是预测模型中选择重要变量的方式之一，它能根据预测变量的重要性对预测变量进行排序。**

在机器学习算法模型中使用逻辑回归、决策树进行建模的时候，我们需要进行变量的筛选。毕竟我们不可能把收集到的很多变量（如100个）都放入到模型中进行建模。因此需要选择一种方法挑选一些变量出来放入模型。但是变量的选择是有很多问题需要考虑的，如变量的预测能力，变量之间的相关性、变量的可解释性等等。“变量的预测能力”是我们认为自己自定出来的一种指标，那么在实际中量化这个指标才是要考虑的问题，那么IV值就被发明出来了。

计算后不同的IV值的预测能力如下：

|  |  |
| --- | --- |
| **IV值** | **预测能力** |
| IV<=0.02 | 无预测能力 |
| 0.02 - 0.1 | 弱预测能力 |
| 0.1 - 0.3 | 中预测能力 |
| 0.3 - 0.5 | 强预测能力 |
| 大于0.5 | 超强预测能力 |

IV的计算公式如下：



很显然就能看出IV值的计算与WOE是有联系的。我们可以在WOE的基础上进一步计算就可以得到IV值。之前说过WOE考虑到了每一个分箱的相对风险，并没有考虑到每一个分箱样本数占总样本数量之比，很显然IV就能反应每一个分箱的贡献。

**五、其他问题**

4.1 对于特征变量来说有空值（NULL）怎么处理：

这应该涉及到数据预处理的知识，可以将空值进行填充操作或者将NULL值单独作为一个特征或者分箱。

4.2 WOE编码后非线性、非单调的变量的处理

在使用Logistic模型进行评分卡建模的时候，往往希望WOE编码结果是线性的和单调的，毕竟Logistic回归的本质是对数线性模型。那么在WOE编码后非线性、非单调的变量的处理方式一般为：

（1）离散变量可以进行合并，连续变量进行再分箱。

（2）删除该变量不加入模型训练。

为什么Logistic模型较难从非线性和非单调的模型中学习到有用的“知识”？就拿年龄分箱来说，WEO结果大体呈现为中间部分的WOE值较小，表示该范围年龄的客户发生违约的概率较低，两端的WOE较大表示违约的可能较大。

4.3 WOE为什么被称为证据权重？

大家可以从其他文章中学习，这里我就不再赘述了，主要结合后验概率的知识。

4.4 WOE结果线性

我们可以不对变量进行WOE编码，毕竟有的模型就可以直接使用原始的特征数据就可以。当是WOE编码可以将非线性的特征转换为线性的特征。这个线性的特征对于像广义的线性回归模型来说是很有必要的，有利于这类模型的构建。

**六、代码实现**

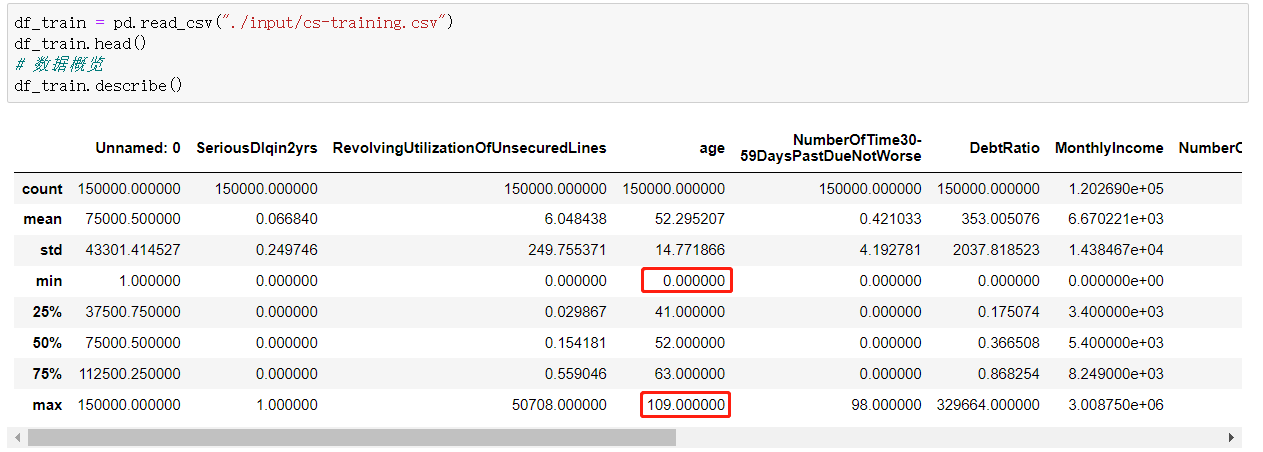
我们来看一个比较综合的例子，例子使用的数据来Kaggle链接如下：

<https://www.kaggle.com/c/GiveMeSomeCredit/data>

这里仅仅使用cs-training.csv数据来说明即可，这个数据中一共有12列数据，各个变量的含义如下：

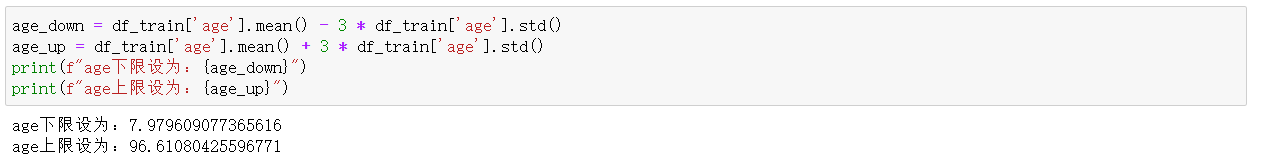


我们来整体看一下数据的情况：

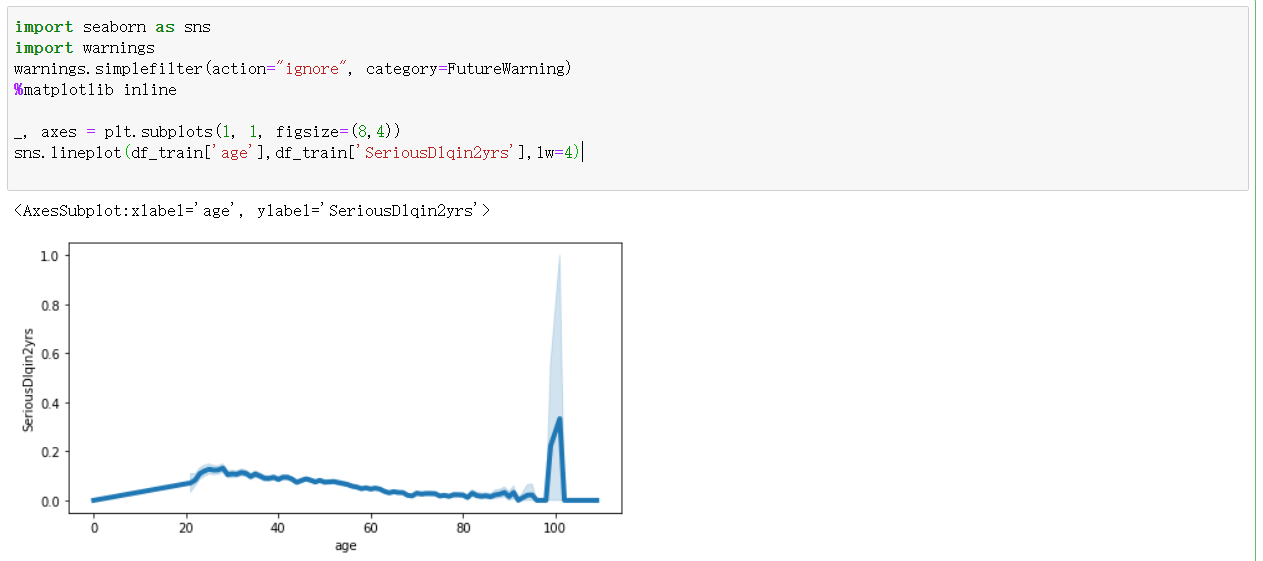


可以看出Age这一列中数据已有异常值得，最小值为0，最大值为109。

那么正常的年龄数据应该的分布是什么范围呢？我们可以使用3倍标准差理论来判断一下：



我们先查看一下年龄与违约情况之间的关系：



可以看出年龄与违约状态的折线图，在一定程度上是符合实际的业务常识的。

接下来对Age列进行WOE编码：

def bin\_woe(tar, var, n=None):

total\_bad = tar.sum()

total\_good =tar.count()-total\_bad

totalRate = total\_good/total\_bad

msheet = pd.DataFrame({tar.name:tar,var.name:var,'var\_bins':pd.qcut(var, n, duplicates='drop')})

grouped = msheet.groupby(['var\_bins'])

groupBad = grouped.sum()[tar.name]

groupTotal = grouped.count()[tar.name]

groupGood = groupTotal - groupBad

groupRate = groupGood/groupBad

groupBadRate = groupBad/groupTotal

groupGoodRate = groupGood/groupTotal

woe = np.log(groupRate/totalRate)

iv = np.sum((groupGood/total\_good-groupBad/total\_bad)\*woe)

dictmap = {}

for x in woe.index:

dictmap[x] = woe[x]

genervar, cut = var.map(dictmap), woe.index

return woe, iv, cut, genervar

# 调用

woe,iv,cut,genervar = bin\_woe(df\_train['SeriousDlqin2yrs'], df\_train['age'],5)

print(woe.tolist())

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print(iv)

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print(cut)

print('\_\_\_\_\_\_\_\_\_\_\_\_\_\_')

print(genervar)

# 输出为：

[-0.49659289366137954, -0.24692219665998896, -0.07273396131143318, 0.4327260133213886, 1.0637379266101183]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

0.2500393230671014

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

CategoricalIndex([(-0.001, 39.0], (39.0, 48.0], (48.0, 56.0], (56.0, 65.0],

(65.0, 109.0]],

categories=[(-0.001, 39.0], (39.0, 48.0], (48.0, 56.0], (56.0, 65.0], (65.0, 109.0]], ordered=True, dtype='category', name='var\_bins')

\_\_\_\_\_\_\_\_\_\_\_\_\_\_

0 -0.246922

1 -0.246922

2 -0.496593

3 -0.496593

4 -0.072734

...

149995 1.063738

149996 -0.246922

149997 0.432726

149998 -0.496593

149999 0.432726

Name: age, Length: 150000, dtype: float64

上述结果中我们没有对Age变量进行异常值的处理。可以看出分箱的数量为5，WOE的结果的数量也为5.IV(age)=0.2500,说明age这个变量具有中等的预测能力。其他变量的预测能力大家可以照着学习。

更多的关于分箱的知识点可以从之后的推文中再学习。

更多关于变量编码的知识可以从这学习到：

<https://github.com/scikit-learn-contrib/category_encoders>

1. **总结**

以上就是WOE编码和IV值得相关知识。WOE更深层次的理论知识推文中没有介绍，大家可以从其他文献中进行学习。文章找那个有什么不对的地方还请大家指正，我们下期见。