规则篇

《100天成为风控专家》

规则生成(1)：单变量(含实操)

出品：东哥起飞

解锁风控课程

关注我的公众号

目录

三个步骤

单一变量规则从筛选到制定生成，一般有以下三个步骤：

对变量进行描述性统计分析，进行初步筛选；

对筛选后保留的变量进行分箱处理，计算分箱下的统计量和指标，基于IV值再次对变量进行筛选；

基于分箱IV分析法、极端值检测两种方法确认规则的阈值；

扫码加我微信

一、变量初筛

1.1. 变量初筛

什么要进行变量筛选?

假如，我们现在手上有几千个变量，很明显不可能全部用于制定规则，一是变量效果有好有坏，并且部分变量之间效果也存在效果重叠；二是规则多太复杂会导致稳定性变差；三是规则多也会大大提升不必要的工作量。因此要对变量进行层层筛选。

变量初筛的过程，主要对变量进行描述性统计（如平均值、最大值、最小值、标准差等）以及缺失率、众数占比等指标计算，**对不符合要求的变量先进行一轮剔除，这样可以减少后面二轮筛选规则时的压力。**

变量名	变量含义	缺失率	众数占比	平均值	最大值	最小值	标准差
A	现行消费贷款机构数	5%	2%	2.5	8	0	1
B	近12个月旅游出行次数	60%	90%	0.1	1	0	0
C	月均夜间短信数	10%	5%	3	50	0	1.5
D	同WIFI地址一个月出现贷款次数	8%	6%	2	15	0	0.5

1.2. 变量初筛

```
def simple_statics():
    # 读入数据
    df = pd.read_csv('data_rule.csv')
    stats = []
    for col in df.columns:
        stats.append((col, df[col].nunique(),
                      (df[col].isnull()).sum() * 100 / df.shape[0],
                      (df[col]==-999).sum() * 100 / df.shape[0],
                      df[col].value_counts(normalize=True, dropna=False).values[0] * 100,
                      df[col].dtype))

    stats_df = pd.DataFrame(stats, columns=['Feature', 'Unique_values', 'Percentage_of_null','Percentage_of_999',
                                            'Percentage_of_mode', 'Type'])
    stats_df.sort_values('Unique_values', ascending=False, inplace=True)
    return stats_df
```

```
sts_df = simple_statics()
sts_df.head(100)
```

	Feature	Unique_values	Percentage_of_null	Percentage_of_999	Percentage_of_mode	Type
0	lmt	1718	0.0	0.000000	2.160889	float64
1	job	13	0.0	0.000000	47.766021	int64
3	basicLevel	6	0.0	1.712503	34.837043	int64
2	ncloseCreditCard	3	0.0	0.235554	79.341660	int64
4	unpayNormalLoan	3	0.0	0.235554	84.530671	int64
5	target	2	0.0	0.000000	99.273644	int64

扫码加我微信

二、变量分箱、指标计算

2.1. 分箱-概念

分箱定义

① 将连续变量离散化；

② 将多状态的离散变量合并为数量更少的几箱；

分箱的用处

① 对变量分层，易于对变量进行统计分析；

② 避免了异常值的干扰，鲁棒性好；

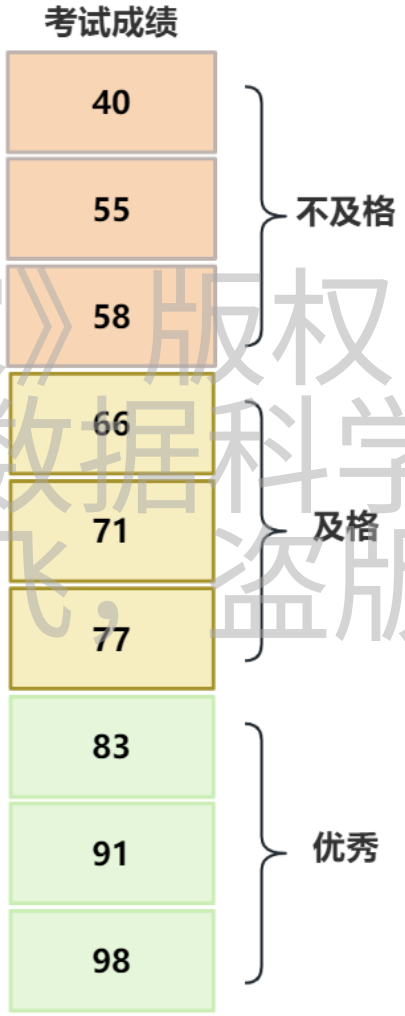
分箱的注意事项

① 分箱必须包括变量的所有数值，不能丢失信息；

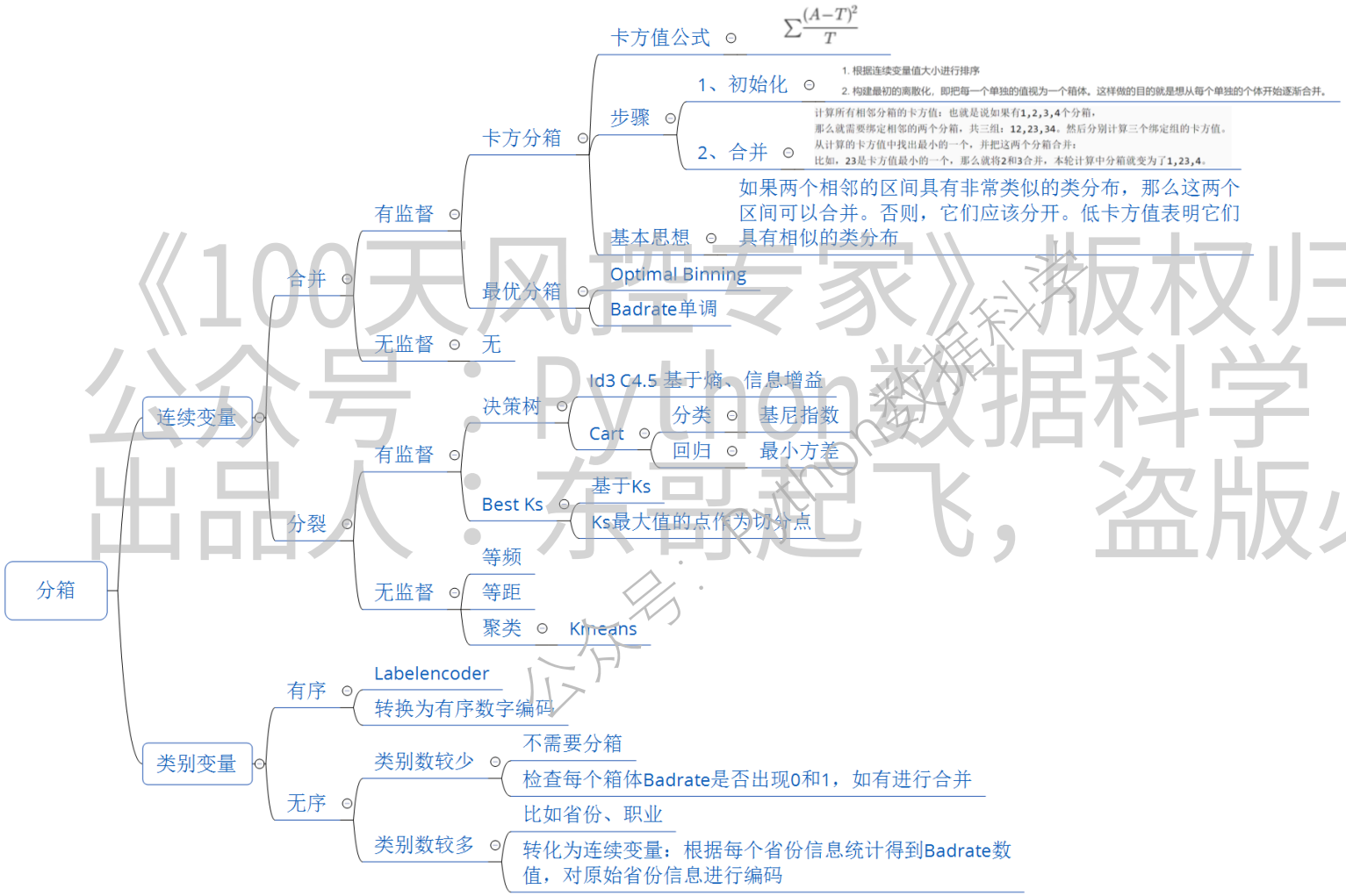
② 分箱数量不宜太多，一般控制在5-8之间；

③ 每箱数量占比至少在5%以上，数量太少没有统计意义；

④ 如果有缺失值，需要单独分一箱；



2.2. 分箱-算法分类



扫码加我微信

2.3. 分箱-统计量

比如“最近3个月新型非银金融机构的查询次数”征信变量，数值类型范围从0-16，下面分为6箱。

分箱	好客户数	坏客户数	总客户数	好客户占比	坏客户占比	总客户占比	区间坏账率
[0.0,0.5)	796	28	824	40.20%	20.59%	38.94%	3.40%
[0.5,1.5)	533	36	569	26.92%	26.47%	26.89%	6.33%
[1.5,2.5)	301	23	324	15.20%	16.91%	15.31%	7.10%
[2.5,3.5)	166	18	184	8.38%	13.24%	8.70%	9.78%
[3.5,4.5)	98	11	109	4.95%	8.09%	5.15%	10.09%
[4.5,16.0)	86	20	106	0.0434	14.71%	5.01%	18.87%
总计	1980	136	2116	100%	100.01%	100.00%	6.43%

① **数量统计**：各分箱下客户的数量求和，比如好/坏/总客户数量。

② **边际占比**：**分箱下的XX数量/XX总数量**，XX可以是好/坏/总客户。比如，[0.0,0.5)分箱下边际好客户占比=分箱下的好客户数/总的好客户数=796/1980=40.2%，同理边际坏客户占比=28/136=20.59%，边际总客户占比=824/2116=38.94%。

③ **区间占比**：**分箱下的XX数量/分箱内总客户数**， XX可以是好/坏/总客户，一般我们只关注坏客户，也叫**区间坏账率**，比如[0.0,0.5)分箱下的区间坏账率=坏客户数/总客户数=28/824=3.4%。

2.4. 分箱-统计量含义

比如“最近3个月新型非银金融机构的查询次数”征信变量，数值类型范围从0-16，下面分为6箱。

分箱	好客户数	坏客户数	总客户数	好客户占比	坏客户占比	总客户占比	区间坏账率
[0.0,0.5)	796	28	824	40.20%	20.59%	38.94%	3.40%
[0.5,1.5)	533	36	569	26.92%	26.47%	26.89%	6.33%
[1.5,2.5)	301	23	324	15.20%	16.91%	15.31%	7.10%
[2.5,3.5)	166	18	184	8.38%	13.24%	8.70%	9.78%
[3.5,4.5)	98	11	109	4.95%	8.09%	5.15%	10.09%
[4.5,16.0)	86	20	106	4.34%	14.71%	5.01%	18.87%
总计	1980	136	2116	100%	100.01%	100.00%	6.43%

① **总客户占比**：代表每个分箱内的样本数据占总量的比例，比如最后一箱[4.5,16)，如果我们将4.5设置规则阈值，那么此时总客户占比就等同于规则的命中率(hit_rate)了。

② **区间坏账率**：该示例中最后一列可以明显看到一个从小到大的排序。这个就是我们前面总提到的“**排序性**”，**一般要求排序具有明显的单调性，这样符合业务的可解释性。**

③ **好/坏客户占比**：可用于评估制定规则的阈值，比如，我们设计该变量值大于4时为拒绝，那么将会拒掉最后一箱的客户，其中坏账客户占全部坏客户中的14.71%，但也会误拒掉全部好客户中的4.34%。

2.5. WOE-公式和含义

WOE（Weight of Evidence）叫做证据权重，计算公式如下：

$$WOE_i = ln(\frac{Bad_i}{Bad_T} / \frac{Good_i}{Good_T}) = ln(\frac{Bad_i}{Bad_T}) - ln(\frac{Good_i}{Good_T})$$

① **WOE含义**：从公式上理解，**WOE为每个分箱里的坏客分布相对于好客分布之间的差异性。**

② **WOE计算示例**

分箱	好客户数	坏客户数	总客户数	好客户占比	坏客户占比	总客户占比	区间坏账率	WOE
[0.0,0.5)	796	28	824	40.20%	20.59%	38.94%	3.40%	-0.6692
[0.5,1.5)	533	36	569	26.92%	26.47%	26.89%	6.33%	-0.0168
[1.5,2.5)	301	23	324	15.20%	16.91%	15.31%	7.10%	0.1066
[2.5,3.5)	166	18	184	8.38%	13.24%	8.70%	9.78%	0.4566
[3.5,4.5)	98	11	109	4.95%	8.09%	5.15%	10.09%	0.4911
[4.5,16.0)	86	20	106	4.34%	14.71%	5.01%	18.87%	1.2196
总计	1980	136	2116	100%	100.01%	100.00%	6.43%	1.5879

公式中的计算项正是我们前面介绍的**边际好坏客户的占比**，因此我们只需取ln然后相减即可得到WOE结果。比如，[0.0,0.5)分箱下**WOE=ln(20.59%)-ln(40.20%)=-0.6692**，其他分箱同理。

2.6. IV-公式和含义

IV（Information Value）信息价值，是基于WOE计算出来的一个指标，在风控策略和模型**中常用来评估变量对目标变量Y的预测能力**，其公式如下。

每个分箱下的IV值

$$IV_i = (\frac{Bad_i}{Bad_T} - \frac{Good_i}{Good_T}) * WOE_i$$

$$= (\frac{Bad_i}{Bad_T} - \frac{Good_i}{Good_T}) * ln(\frac{Bad_i}{Bad_T} / \frac{Good_i}{Good_T})$$

所有分箱IV值求和

$$IV = \sum_{i=1}^{n} IV_i$$

分箱	好客户数	坏客户数	总客户数	好客户占比	坏客户占比	总客户占比	区间坏账率	WOE	IV
[0.0,0.5)	796	28	824	40.20%	20.59%	38.94%	3.40%	-0.6692	0.1313
[0.5,1.5)	533	36	569	26.92%	26.47%	26.89%	6.33%	-0.0168	0.0001
[1.5,2.5)	301	23	324	15.20%	16.91%	15.31%	7.10%	0.1066	0.0018
[2.5,3.5)	166	18	184	8.38%	13.24%	8.70%	9.78%	0.4566	0.0222
[3.5,4.5)	98	11	109	4.95%	8.09%	5.15%	10.09%	0.4911	0.0154
[4.5,16.0)	86	20	106	4.34%	14.71%	5.01%	18.87%	1.2196	0.1264
总计	1980	136	2116	100%	100.01%	100.00%	6.43%	1.5879	0.2972

基于每箱的WOE结果再乘以坏客户和好客户的边际占比之差，可得到每箱的IV值，最后将所有分箱的IV值求和得到最终IV结果，此例中基于该分箱的最终IV值=0.2972。

扫码加我微信

2.7. IV-使用标准

① IV值的衡量标准是什么?

IV范围	预测效果
IV<0.02	几乎没有
0.02<=IV<0.1	弱
0.1<=IV<0.3	中
0.3<=IV<0.5	强
0.5<=IV	需要排查，可能穿越

变量名	IV	预测能力
最近12月贷款信用卡查询次数，	0.4246	强
最近6月线款信用卡查询欠数	0.3489	强
现行贷记卡（R2）账户数（人民币）	0.3032	强
现行当前月度房贷应还金额（房贷）	0.2372	中
最近24个月信贷账户最大持续道期月份数	0.2069	中
货记卡现行账户计数	0.2028	中
最近3月贷款信用卡查询机构数	0.1871	中
最近3月贷款信用卡查询欠数	0.1871	中
现行经营性账户数	0.1697	中
学历	0.1586	中
最近12个月所有信贷产品最大的历史道朝期数	0.1303	中
现行房贷账户数	0.0803	弱
现行贷记卡（R2）燥近6个月1-29天无期欠数	0.0762	弱
金近公积金纳比的健个人缴存比例+公司缴存比例	0.0548	弱

② 筛选变量的标准：

好而不同，单体预测能力强，相互关联性弱。因此除了用IV值评估变量预测能力以外，还要考虑变量之间的相关性，选择IV值高的并且彼此之间相关性低的规则组合，和做模型筛选入模变量是一个道理。同时也需要考虑变量的区间坏账排序性，即业务可解释性。

2.8. IV-Python代码实现

一些用于做模型的三方Python包，如toad、scorecardpy已经封装好了分箱、WOE和IV的计算函数，可直接调用函数实现IV的计算。此外，如果对三方包的函数功能不满意，也可以基于WOE和IV公式自行手写一个，过程不算复杂。


```python
def cal_iv(x, y):
    """
    IV计算函数
    :param x: feature
    :param y: label
    :return:
    """
    crtab = pd.crosstab(x, y, margins=True)
    crtab.columns = ['good', 'bad', 'total']
    crtab['factor_per'] = crtab['total'] / len(y)
    crtab['bad_per'] = crtab['bad'] / crtab['total']
    crtab['p'] = crtab['bad'] / crtab.loc['All', 'bad']
    crtab['q'] = crtab['good'] / crtab.loc['All', 'good']
    crtab['woe'] = np.log(crtab['p'] / crtab['q'])
    crtab2 = crtab[abs(crtab.woe) != np.inf]

    crtab['IV'] = sum(
        (crtab2['p'] - crtab2['q']) * np.log(crtab2['p'] / crtab2['q']))
    crtab.reset_index(inplace=True)
    crtab['varname'] = crtab.columns[0]
    crtab.rename(columns={crtab.columns[0]: 'var_level'}, inplace=True)
    crtab.var_level = crtab.var_level.apply(str)
    return crtab
```

扫码加我微信

三、制定规则阈值

3.1. 规则效果-两个期限

对于规则效果而言，我们有以下两点期望：

a）拒绝客户占总体客户的比例（命中率）不易过高；

一是拒绝客户占比过高会影响通过率，进而影响业务规模；

二是整个风控策略流程中有很多个判断风险的环节，类似一个漏斗，对客户进行层层筛选，各个环节维度不同，可以对风险识别互为补充，是一种协作的关系。因此每个环节只会做最有把握的事情，即拒绝最差的客户，剩下的部分交给其它环节来判断。如果单个规则的拒绝率过高，那么将会损失过多好的客户。

因此，规则阈值的设置一般会比较极端，要实现的效果就是**通过大部分客户，而只拒绝一少部分客户，一般拒绝比例不超过5%。**

b）拒绝客户中，坏客户占比越高越好，同时好客户占比越低越好；

我们希望尽量抓到更多的坏客户，而减少对好客户的误杀。

3.2. 规则效果-5个评估项

① 命中率 (hit_rate)	② 精准率 (bad_rate)	③ 召回率 (recall_rate)	④ 排序性	⑤ 可解释性
a）**定义**：指的是规则拒绝的客户占总客户的比例 b）**解释**：一般拒绝比例不超过5%，否则将对业务通过率有很大影响。	a）**定义**：指的是规则拒绝的人中坏用户的占比 b）**解释**：理想情况越接近100%越好，但实际情况中由于坏样本浓度过低，很难达到一个很高的精确率。因此一般精准率的高低是和整体样本的坏账率作比较的。比如，整体坏账率为6%，精准率为18%，高于3倍。	a）**定义**：指的是规则拒绝的坏用户占总的坏用户的比例，简单说就是规则抓出了多少坏人。 b）**解释**：比如，一个规则精确率达到了80%，但拒绝的坏用户只占了0.1%，那这条规则对于降低风控坏账其实没什么用，大部分坏用户还是被它放过了。	a）**定义**：指的是区间坏账率的排序性，是针对弱规则的评估指标。 b）**解释**：例如一个评分规则，我们希望分箱下的区间坏账率和分数大小是呈现单调性的，即分数越低，区间坏账率越高。这样我们就能根据业务需求对它进行收紧或放松，实现对贷后逾期的可控性。	可解释性指的是评估规则本身是否符合业务感知，比如征信机构查询次数一般是拒绝查询次数非常多的客群，但规则是拒绝查询次数少的客户，那这条规则就不具有解释性了。 另一方面可解释性是用于向业务人员解释规则拒绝是否科学合理。

3.3. 规则阈值制定方法-基于分箱的IV分析法

① 基于分箱的IV分析法

第一种是比较常用的方法，就是基于分箱计算各统计量和IV值。

比如下面这个变量，基于分箱结果，我们将最后一箱（区间坏账率最高的）定为规则的阈值：**最近3个月新型非银金融机构的查询次数>4，触发则拒绝，反之通过。**

分箱	好客户数	坏客户数	总客户数	好客户占比	坏客户占比	总客户占比	区间坏账率	
[0.0,0.5)	796	28	824	40.20%	20.59%	38.94%	3.40%	
[0.5,1.5)	533	36	569	26.92%	26.47%	26.89%	6.33%	
[1.5,2.5)	301	23	324	15.20%	16.91%	15.31%	7.10%	
[2.5,3.5)	166	18	184	8.38%	13.24%	8.70%	9.78%	
[3.5,4.5)	98	11	109	4.95%	8.09%	5.15%	10.09%	通过
[4.5,16.0)	86	20	106	4.34%	14.71%	5.01%	18.87%	拒绝
总计	1980	136	2116	100%	100.01%	100.00%	6.43%	

1）命中率（hit_rate）为5.01%；

2）拒绝客户的坏账率（精准率）为18.87%，Lift=18.87%/6.43%=2.93；

3）坏客户占比（召回率）为14.71%，即抓到了坏客户中的14.71%，并且仅拒绝了好客户中的4.34%；

4）区间坏账率有明显的排序性，且有业务可解释性。

3.4. 规则阈值制定方法-极端值检测

② 极端值检测

极端值检测的方法的思想是：**高风险客群属于一群异常的客户，他们的变量特征是异于常人的，会集中在比较极端的范围内**，即变量值越大或者越小，客户的风险越高。极端值检测通过“**枚举分位数**”的方式，**枚举可能得极端值，作为备选的阈值**，以此制定出规则。

比如下面设置了8个分位点 [0.005, 0.01, 0.02, 0.05, 0.95, 0.98, 0.99, 0.995])，分别基于这些点位制定出规则，然后输出评估指标。我们看到，综合hit_rate、hit_bad_rate、lift等指标，该变量阈值定为4.25为最佳。

	var	rule	total_size	total_bad_size	total_bad_rate	hit_rate	hit_size	hit_bad_size	hit_bad_rate	lift
0	最近3个月新型非银金融机构的查询次数	is missing	2116	136	0.064272	0.000000	0	0	NaN	NaN
1	最近3个月新型非银金融机构的查询次数	<= 0.0	2116	136	0.064272	0.389414	824	28	0.033981	0.528698
2	最近3个月新型非银金融机构的查询次数	<= 0.0	2116	136	0.064272	0.389414	824	28	0.033981	0.528698
3	最近3个月新型非银金融机构的查询次数	<= 0.0	2116	136	0.064272	0.389414	824	28	0.033981	0.528698
4	最近3个月新型非银金融机构的查询次数	<= 0.0	2116	136	0.064272	0.389414	824	28	0.033981	0.528698
5	最近3个月新型非银金融机构的查询次数	>= 4.25	2116	136	0.064272	0.050095	106	20	0.188679	2.935627
6	最近3个月新型非银金融机构的查询次数	>= 6.0	2116	136	0.064272	0.029301	62	0	0.000000	0.000000
7	最近3个月新型非银金融机构的查询次数	>= 9.0	2116	136	0.064272	0.010870	23	0	0.000000	0.000000
8	最近3个月新型非银金融机构的查询次数	>= 11.0	2116	136	0.064272	0.006144	13	0	0.000000	0.000000

扫码加我微信

四、Python代码实操