核心技术报告

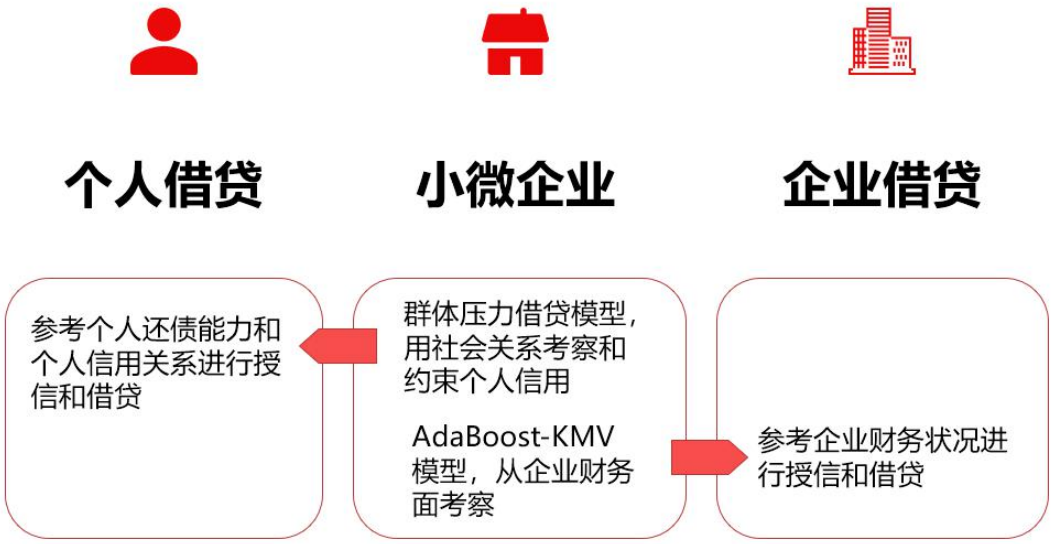
在疫情的大背景之下，企业的运营受到极大的冲击。尤其是小微企业自身抵抗风险的能力相对较弱，受影响更为严重。小微企业推动复产复工亟需贷款资金。而作为银行，考虑到小微企业公开数据远小于上市公司，因而难以评估企业风险，同时审核成本也更高。因此双方都需要一个借贷系统，能够对企业进行合理地估分，从而帮助小微企业借贷以及银行放贷，加速疫情后恢复，减少审核步骤，提高金融系统运营效率。

模型探究

根据调研，在我国，目前银行针对小微企业提供的贷款服务主要是抵押。即，小微企业需要提供相应的抵押物（如房产等），银行才会放贷。由于小微企业数据的相对不透明，很难评估其信用。银行很少能提供基于授信的贷款，即给定一个不需要抵押的额度进行放贷。

查阅论文，我们了解到，传统评估信用的风险度量模型有 5C 模型（偏向主观）、Z 值评分（财务数据）和 ZETA 模型 （Z 值拓展）。现代风险度量模型有 KMV 模型，Risk-Metric 模型，Credit Metrics 模型和麦肯锡模型。

我们认为，小微企业借贷的借贷逻辑介于个人借贷和企业借贷之间。针对个人借贷，需要看重个人的借贷能力。而由于无法简单地判断个人的借贷能力，我们通过群体压力这样的方式转换，可以很好地反应甚至提高个人的偿债能力。针对企业借贷，更看重财务指标，而这也是我们模型建立的基础。

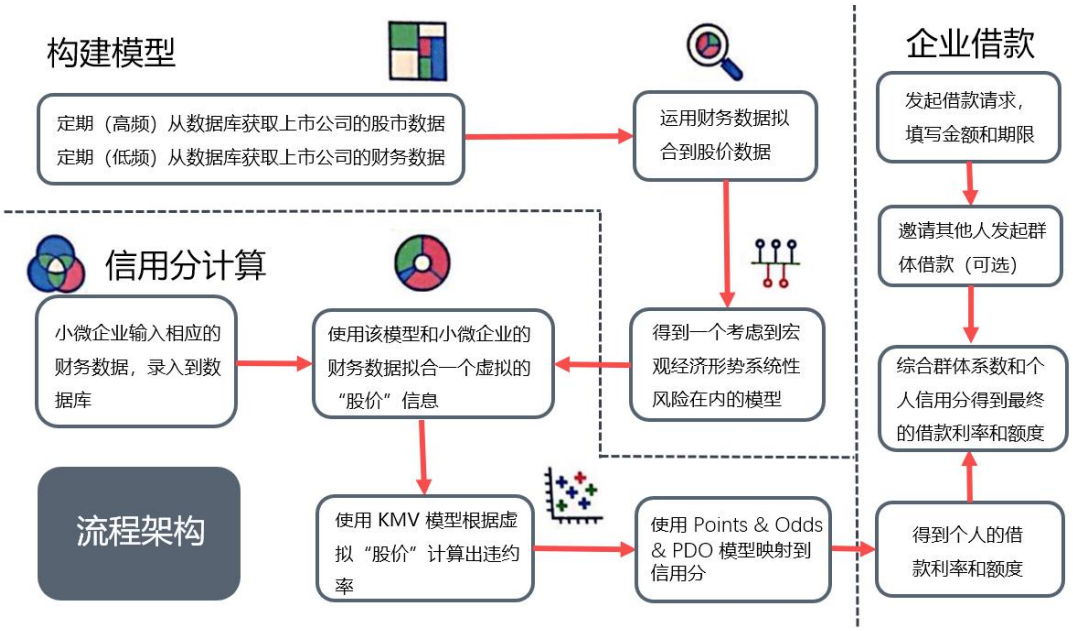


我国现阶段采用的传统信用风险度量模型定性、主观，主要以小微企业的财务状况作为衡量依据，会导致风险评估结果准确性低、存在时滞等问题。同时，违约公司和正常经营公司的财务比例差异不显著，所以财务数据模型对国内适用性不是特别强。

KMV 公司开发的 KMV 模型是基于现代公司理财和期权理论的结构性模型。其理论依据是，贷款的信用风险是在给定负债的情况下由债务人的资产市场价值决定的。它假设将公司股权看作为以公司资产价值为标的的看涨期权，公司的负债为看涨期权的执行价格当公司的资产价值大于负债时，公司股东选择不违约，并获得剩余索取权，这就相当执行了看涨期权，反之亦然。当公司的资产价值（股权价值+债务价值）低于一定水平（违约点 Default Point），则公司违约。它的优势在于，将资本市场的信息纳入考核，不像传统方法依赖于历史账面资料，采用股票市场的数据，更新很快；一旦给定了公司资产结构，和资产价值的随机过程，便可以得到任一时间单位的实际违约概率。

财务报告反映公司的理事情况，市场价格更能反映公司发展的未来趋势，最准确的信用风险度量方法应该同时使用这两种数据资源。国内也有诸多文献证实了 KMV 模型在我国证券市场的适用性，因此我们的基础模型选用 KMV 模型。由于小微企业往往没有上市，所以股价信息无从获得。为了解决这一问题，我们首先使用 Adaboost 机器学习模型来先从财务信息得到股价信息，然后使用 Black Scholes 计算期权价格的逆过程，使用数值分析的方法求解，即可得到 KMV 模型下的违约概率。再利用 Points & Odds & PDO 将概率映射到信用分。

因此，我们的模型可以概括为，通过建立数据库，评估哪些变量对于预测违约率有效，调用机器学习算法建立评分模型，结合群体压力模型完成借贷。



数据来源

建模时，数据来源于多家真实银行对于小微企业的借贷信息和所有上市公司的数据。

对于小微企业的借贷信息，我们拥有贷款科目号、贷款科目名称、产品代码、产品名称、贷款用途、用途代码名称、客户号、年龄、借款人类型名称、企业规模、贷款合同号、借据序号　　担保方式、贷款期限、贷款合同金额、贷款余额、月日均余额、季日均余额、年日均余额、合同状态、行业分类一级名称、行业分类、行业投向一级名称、行业投向、放款渠道、还款方式、发放金额、利率调整方式、定价方式、结息周期、执行利率、四级形态、五级形态、公司类十级形态、预计损失率、预计损失额、上次分类人员、上次分类时间、是否分类、贷款发放日期、贷款到期日期、还贷还息日期、还款金额、贷款利息收入、表内欠息余额、表外欠息余额、上次结息日期、下一结息日期、预结利息、昨日存款余额、存贷挂钩标志、审批责任人甲柜员号。

对于上市公司的数据，我们提取了股票代码、收盘价、流通股数、流通市值、年份、每股净资产、基本每股收益、净利润增长率、总资产周转率 TTM、总资产报酬率、无风险利率、速动比率、资产负债率、营业利润率 TTM。

	特征	获取渠道
上市企业	贷款余额、月日均余额、季日均余额、年日均余额、合同状态、行业分类一级名称、行业分类、行业投向一级名称、行业投向、放款渠道、还款方式、发放金额、利率调整方式、定价方式、结息周期、执行利率、四级形态、五级形态、公司类十级形态、预计损失率、预计损失额、上次分类人员、上次分类时间、是否分类、贷款发放日期、贷款到期日期、还贷还息日等	非公开渠道
小微企业	股票代码、收盘价、流通股数、流通市值、年份、每股净资产、基本每股收益、净利润增长率、总资产周转率TTM、总资产报酬率、无风险利率、速动比率、资产负债率、营业利润率TTM	公开渠道

数据预处理

我们对原始数据做了大量的预处理，包括缺失值处理、异常值处理、数值化处理和探索性分析。通过分箱、合并、基于统计构造新特征等特征工程，整理出来了结构性数据。通过（净利润/基本每股收益 - 流通股数）* 每股净资产 + 流通股市场价值得到公司基本价值。通过流动负债合计 + 长期负债 *0.5 得到违约点。通过计算（净利润本年本期金额—净利润上年同期金额)/（净利润上年同期金额）得到净利润增长率。通过（利润总额＋财务费用）/ 平均资产总额，平均资产总额=（资产合计期末余额+资产合计上年期末余额）/2 计算总资产报酬率等等。

部分数据处理的过程如下。

资产负债率和速动比率

数据来源于CSMAR数据库中的公司研究系列 /财务指标分析/偿债能力

In [4]:
```
company = pd.read_csv("data/FI_T1.csv")
```

使用合并报表而不是母公司报表

In [5]:
```
company = company[company['Typrep'] == 'A']
company.drop(columns = ['Typrep'],inplace=True)
```

剔除缺乏速动比率和资产负债率的数据

In [6]:
```
company.dropna(inplace=True)
```

改列名，同时将日期转为 pandas 日期格式

In [7]:
```
company.columns = ['股票代码','时间','速动比率','资产负债率']
company['时间'] = pd.to_datetime(company['时间'])
company['年份'] = company['时间'].dt.year
```

仅保存每个公司年底(或2020年最晚)的财报

In [8]:
```
company.sort_values(['股票代码','时间'],inplace = True)
company.drop_duplicates(['股票代码','年份'],keep='last',inplace=True)
company
```

	股票代码	时间	速动比率	资产负债率	年份
50804	1	1990-12-31	14.289268	0.019279	1990
50794	1	1991-12-31	0.428590	0.017736	1991
50755	1	1992-12-31	0.322646	0.026040	1992
9	2	1991-12-31	0.875832	0.664623	1991
10	2	1992-12-31	0.623260	0.749119	1992
...	...	...	...	...	...
226674	900957	2016-12-31	1.404382	0.661961	2016
253963	900957	2017-12-31	1.925383	0.584235	2017
281418	900957	2018-12-31	2.100740	0.551612	2018
313467	900957	2019-12-31	1.964754	0.531432	2019
324290	900957	2020-06-30	2.356481	0.518559	2020

50271 rows × 5 columns

KMV 模型

参考多篇论文，为了变量的可解释性，我们没有挑选所有的变量进行回归分析，而是选择了能够反应企业信用信息的一些变量，并且将其应用于股价预估。具体而言，总资产报酬率、营业利润率、净利润增长率、速动比率、总资产周转率、资产负债率作为变量预测违约率。使用 KMV 模型，根据股价信息计算违约概率。因为 KMV 模型基于一个类似 Black Scholes 的期权定价模型，我们可以用类似的方法计算。但是 KMV 计算是 Black Scholes 的逆过程，因此没有解析解。我们使用数值分析的方法得到其拟合解

使用 scipy.optimize 求解非线性方程

In [ ]:
```
from scipy.optimize import fsolve
import scipy.stats as sts
```

定义优化方程

res是两个为0的条件

In [ ]:
```
def equation(x,Ve,T,DP,sigma_e,r):
    d1 = (math.log(abs(x[0]/DP)) + (r+x[1]**2/2)*T) / (x[1] * math.sqrt(T))
    d2 = d1 - x[1] * math.sqrt(T)
    Nd1 = sts.norm(0,1).cdf(d1)
    Nd2 = sts.norm(0,1).cdf(d2)
    res1 = x[0] * Nd1  - math.exp(-r*T) * DP * Nd2 - Ve
    res2 = x[0] * Nd1 * x[1] - Ve * sigma_e
    return [res1,res2]
```

In [ ]:
```
def fn(row):
    Ve,DP,sigma_e,r = row['总市值'],row['违约点'],row['年波动率'],row['无风险利率']
    return fsolve(lambda x:equation(x,Ve=Ve,T=1,DP=DP,sigma_e=sigma_e,r=r), [Ve,sigma_e])
```

In [ ]:
```
data['Va'],data['sigma_a'] = zip(*data.apply(lambda row:fn(row),axis=1))
data
```

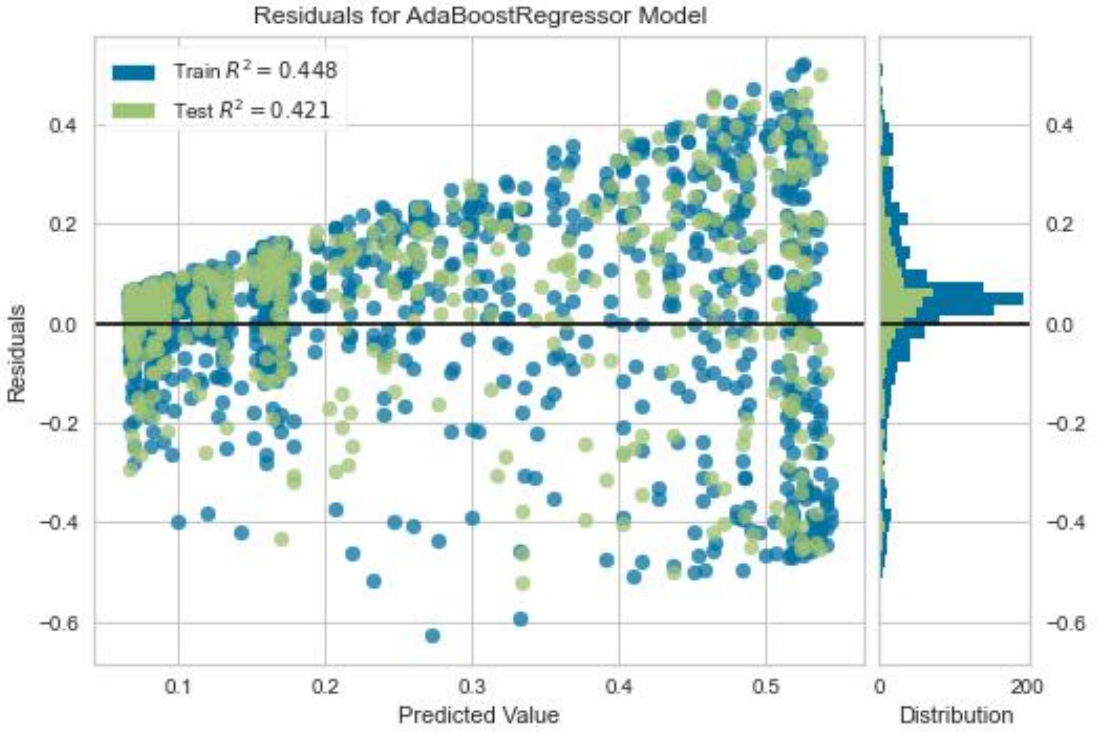
机器学习模型

我们使用 pycaret 框架，交叉测试了 AdaBoost、Gradient Boosting、随机森林、支持向量机、Extra Trees、CatBoost 等大量机器学习模型，最终得到 AdaBoost 模型取得了最优的拟合效果。

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE	TT (Sec)
0	AdaBoost Regressor	0.1349	0.0361	0.1894	0.3804	0.1387	3.4144	0.0629
1	Gradient Boosting Regressor	0.1267	0.0367	0.1908	0.3703	0.1372	2.7643	0.2778
2	Random Forest	0.1304	0.0377	0.1935	0.3522	0.1401	2.7693	0.3194
3	Support Vector Machine	0.1306	0.0380	0.1943	0.3521	0.1376	3.2214	0.0528
4	Extra Trees Regressor	0.1315	0.0382	0.1950	0.3443	0.1407	2.8291	0.1787
5	CatBoost Regressor	0.1306	0.0388	0.1963	0.3329	0.1410	2.6381	2.3180
6	Light Gradient Boosting Machine	0.1366	0.0419	0.2040	0.2842	0.1476	2.7114	0.1985
7	K Neighbors Regressor	0.1355	0.0420	0.2044	0.2777	0.1477	2.9236	0.0051
8	Extreme Gradient Boosting	0.1405	0.0449	0.2112	0.2269	0.1531	2.8459	0.2442
9	Elastic Net	0.1794	0.0591	0.2422	-0.0045	0.1781	6.4174	0.0039
10	Lasso Least Angle Regression	0.1794	0.0591	0.2422	-0.0045	0.1781	6.4174	0.0051
11	Lasso Regression	0.1794	0.0591	0.2422	-0.0045	0.1781	6.4174	0.0043
12	Huber Regressor	0.1332	0.0566	0.2320	-0.0114	0.1506	2.3580	0.0303
13	Random Sample Consensus	0.1377	0.0579	0.2302	-0.0578	0.1471	2.8942	0.1235
14	Decision Tree	0.1597	0.0638	0.2516	-0.1015	0.1823	2.8744	0.0117
15	Orthogonal Matching Pursuit	0.1526	0.0610	0.2372	-0.1134	0.1547	4.3911	0.0030
16	TheilSen Regressor	0.1385	0.0631	0.2390	-0.1497	0.1472	2.6399	1.3184
17	Ridge Regression	0.1545	0.0632	0.2409	-0.1533	0.1549	4.2713	0.0042
18	Bayesian Ridge	0.1548	0.0641	0.2420	-0.1744	0.1552	4.2821	0.0077
19	Linear Regression	0.1545	0.0646	0.2424	-0.1844	0.1550	4.2558	0.0101
20	Least Angle Regression	0.1545	0.0646	0.2424	-0.1844	0.1550	4.2562	0.0049
21	Passive Aggressive Regressor	0.4039	0.2774	0.4981	-3.8819	0.3028	14.4702	0.0043

	MAE	MSE	RMSE	R2	RMSLE	MAPE
0	0.1363	0.0334	0.1829	0.3655	0.1385	3.1757
1	0.1203	0.0305	0.1745	0.4015	0.1271	3.8368
2	0.1590	0.0482	0.2196	0.2132	0.1619	5.3135
3	0.1307	0.0357	0.1889	0.5379	0.1329	2.7172
4	0.1201	0.0289	0.1701	0.3640	0.1275	2.9908
5	0.1163	0.0281	0.1678	0.4516	0.1246	2.5815
6	0.1441	0.0411	0.2027	0.3961	0.1450	3.5100
7	0.1540	0.0430	0.2074	0.3191	0.1524	3.8112
8	0.1336	0.0338	0.1839	0.3296	0.1359	3.8521
9	0.1389	0.0380	0.1949	0.4364	0.1416	3.2700
Mean	0.1353	0.0361	0.1893	0.3815	0.1387	3.5059
SD	0.0136	0.0062	0.0161	0.0828	0.0113	0.7414

回归残差图



预测和实际违约率做对比

index	总资产报酬率	营业利润率TTM	净利润增长率	速动比率	总资产周转率TTM	资产负债率	违约率	Label
0	-0.2067	-2.072802	0.108306	0.531122	0.108306	0.819505	0.312713	0.5084
1	0.0463	0.050168	0.549039	0.698595	0.549039	0.505634	0.138138	0.1889
2	0.0744	0.067757	0.947269	0.661535	0.947269	0.286011	0.008458	0.0748
3	0.0494	0.155213	0.218061	0.402045	0.218061	0.626002	0.358269	0.4628
4	0.1444	0.426275	0.340059	1.256601	0.340059	0.500599	0.176210	0.1801
...	...	...	...	...	...	...	...	...
207	0.1165	0.090152	0.739009	1.483228	0.739009	0.407718	0.081755	0.0789
208	0.0452	0.139935	0.315713	1.138978	0.315713	0.562481	0.253859	0.3434
209	0.0867	0.137513	0.479876	1.007126	0.479876	0.589455	0.100081	0.3835
210	0.0473	0.101820	0.511929	1.146327	0.511929	0.520198	0.105864	0.2065
211	0.0615	0.257071	0.104729	2.100740	0.104729	0.551612	0.004668	0.2605

评分卡模型

我们应用评分卡模型，也就是基于 Points & Odds & PDO 将违约率映射到分数。

$$\text{Score} = \text{A}-\text{B} * \ln\left(\frac{\text{P}_{y=1}}{1-\text{P}_{y=1}}\right)$$

```
data['odds'] = data['违约率']/(1-data['违约率'])
data['odds']
```

```
0       1.168485e+09
1       1.326539e-03
2       5.660813e-02
3       7.263741e-01
4       2.853214e-02
            ...
2319    2.494380e+00
2320    2.107179e-01
2321    1.808687e+00
2322    4.975545e+00
2323    4.690003e-03
Name: odds, Length: 2324, dtype: float64
```

使用拟合函数对 A 和 B 求解。

定义拟合的范围

In [8]:
```
low,up = data['lnodds'].min(),data['lnodds'].max()
score_low,score_up = 350,900
```

In [9]:
```
low,up
```

```
(-6.895848765205113, 4.543239251879722)
```

定义拟合的函数形式

定义为 score = A - B * ln(p/(1-p))

In [10]:
```
from scipy.optimize import fsolve
```

In [11]:
```
def equation(x,score_low,score_up,low,up):
    res1 = score_up - (x[0] - x[1] * up)
    res2 = score_low - (x[0] - x[1] * low)
    return [res1,res2]
```

In [12]:
```
A,B = fsolve(lambda x:equation(x,score_low,score_up,low,up), [0,0])
```

In [13]:
```
fn = lambda ln_odds:A - B * ln_odds
```

In [14]:
```
data['信用分'] = data['lnodds'].apply(fn)
```

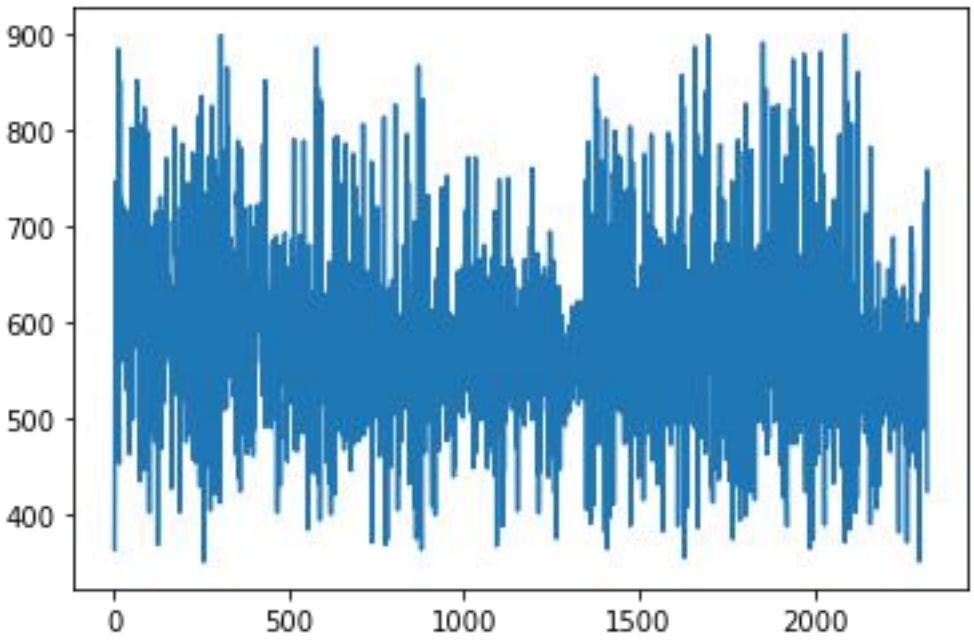
评分卡模型的评分结果

In [14]:
```
data[['股票代码','违约率','信用分']]
```

	股票代码	违约率	信用分
1	4	0.001325	363.013869
2	5	0.053575	543.488796
3	6	0.420751	666.186678
4	8	0.027741	510.547646
5	9	0.795727	746.937631
...	...	...	...
2319	900948	0.713826	725.505371
2320	900951	0.174044	606.684593
2321	900953	0.643962	710.050323
2322	900956	0.832651	758.704876
2323	900957	0.004668	423.733118

2120 rows × 3 columns

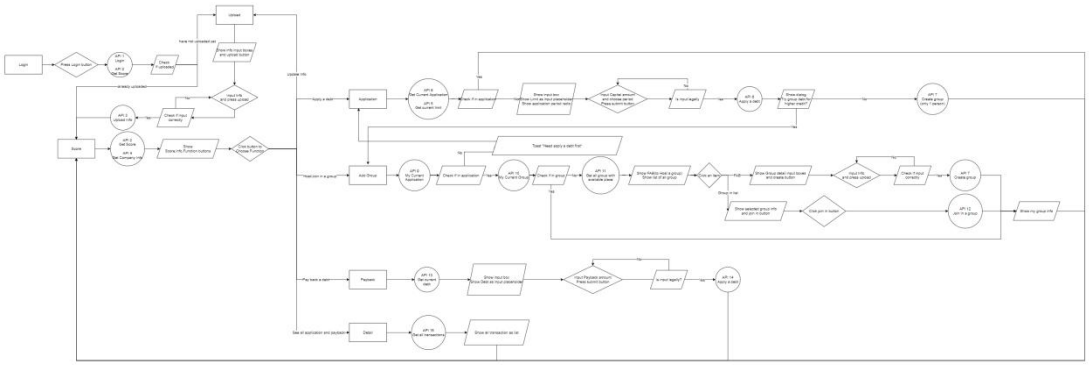
评分卡模型的评分结果分布



群体压力与借贷流程

现在，已经拥有了财务信息 -> 股价信息 -> 违约率 -> 信用分这样的完整的一条链。我们可以根据小微企业的基本财务信息来得到其信用分。但是，这样没有从根本提高小微企业的信用。参考论文，我们采用了群体压力的模型。在这个模型下，多人参与借款，他们共担违约的信用风险。通过共担风险，可以提高小微企业的还款意愿。对于小微企业来说，多人共担风险可以提供一定的额度宽限和利率减免。使用小微企业数量作为一个变量，建立一个对数函数，输出一个可靠的系数来作为额度宽限和利率减免的指标。

产品实现逻辑



（https://borealin.cn/citi/CitiCupFrontEndProcess.html）

未来展望

我们的模型量化了小微企业的信用状况，给借贷提供了一种新的思路，同时用群体压力模型来降低了违约率。

我们可以从下面几个角度对模型做进一步优化和提升。

考虑到疫情的冲击对于不同行业和不同地区的影响不同，可以再加入风险溢价，对贷款利率作微调，以更好地帮助到受到冲击的小微企业。

同时，不同行业的运营模式不同，使得不同行业的利率和额度都有区别。我们可以对不同行业添加不同的系数，同时不同季度不同行业也有周期波动，所以行业系数也可以随着时间变动，以更好地刻画。

最后，为了应对单人借贷无法应用群体压力进行约束，我们可以加入舆情分析。通过 NLP 技术，监测与借款人相关的舆情分析，作为辅助诊断。还有一种关联的信用分析是可以使用区块链技术，对小微企业的供应链上下游进行企业信用分析，预测小微企业供应链断裂的概率。