基于机器与深度学习的企业ST概率预测

ST Probability of Stock based on Deep Learning

西南财经大学 SWUFE

Repoter: Wenjie Lan　　Mentor: Jun Wang

Time: 2022-8-21

目录

CONTENTS

1.1 研究背景
background

研究意义:财务困境预测是财务管理和投资管理领域的一个重要研究方向。

目标:基于当前历史信息预测公司下一季度被ST的概率。

→ **二分类问题：下一个季度是否ST**

主要观点：公司是否会陷入财务困境会受到当前财务等指标的影响。

研究的问题：（1）如何构建选取预警指标，除传统指标外，能否构建是否新指标对是否企业ST有预测作用？

（2）如何根据不同的数据格式选择相应的模型？

2.1 数据构建|传统特征选择

Dataset Contribution

表 1 ： 初步财务指标选取图

类型	指标	类型	指标
盈利能力	税前收入/平均总资产	营运能力	销售收入/应收账款平均余额
	净利润/平均总资产		销售成本/平均库存量
	净利润/平均流动资产		应收账款周转天数+库存周转天数
	净利润/平均固定资产		销售成本/应付账款平均余额
	净利润/平均所有者权益		销售收入/平均流动资产
	税前收入/投资资本量		销售收入/平均固定资产
	(销售收入-销售成本)/销售收入		销售收入/平均总资产
	销售利润/销售收入	发展能力	本年总资产增加量/去年总资产量
	净利润/销售收入		本年净资产增加量/去年净资产量
	销售成本-销售收入		本年净利润增加量/去年净利润
偿债能力	流动资产/流动负债		本年总利润增加量/去年净利润
	(流动资产-库存)/流动负债		本年销售收入增加量/去年销售收入
	流动资产-流动负债		本年所有者权益增加量/去年所有者权益
	净经营性现金流/流动负债	权益比例	净利润/年末普通股股数
	总负债/总资产		现金及现金等价物的净增长量/年末普通股股数
	总负债/(总资产-净无形资产)		销售收入/年末普通股股数
	总资产/所有者权益		销售利润/年末普通股股数
	总负债/所有者权益		所有者权益/年末普通股股数
	所有者权益/总负债		资本留存量/年末普通股股数
	净经营性现金流/流动负债		(盈余留存量+未分配利润)/年末普通股股数
金融结构	流动资产/总资产		净经营性现金流/年末普通股股数
	现金流/总资产		
	固定资产/总资产		
	所有者权益/固定资产		

类型	指标
股票交易信息	开盘价
	收盘价
	成交量
	成交额
	换手率
	最高价
	最低价

◆ 特征选取原则

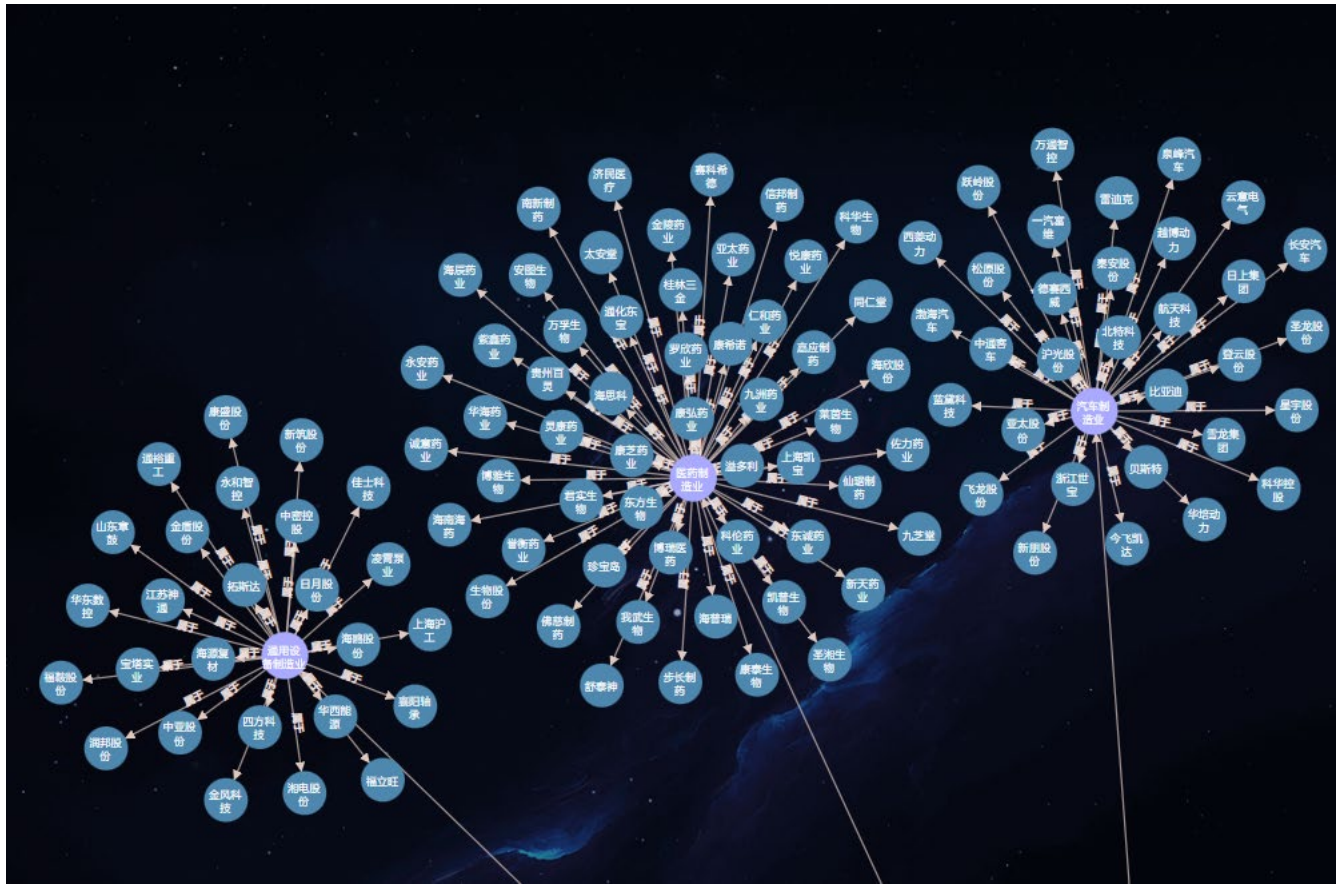
- 在之前的研究中被使用过
- 指标数据可获取
- 指标对预测ST有效果

◆ 数据来源

- CSMAR（财务指标：8）
- BaoStock（股票信息：41）
- 季度数据（17-21年）

2.2 数据构建|图的构建

Dataset Contribution



◆ 图的构建

每只股票的行业分类信息，输出邻接矩阵，每两个股票之间的相关关系。

◆ 模型使用

LSTM+GCN

面板数据，邻接矩阵

◆ 后续

知识图谱搭建

2.3 数据构建|数据规整

model establishment

	code	dateY	quarter
0	sh. 600000	2017	q1
1	sh. 600000	2017	q2
16	sh. 600000	2021	q1
17	sh. 600000	2021	q2
18	sh. 600000	2021	q3

isST
0
0
0
0
0

construction of dataset

- 缺失值处理：填充为0
- 当季度是否ST，滞后一期，下一季度是否被ST
- 以一年四个季度为一个时间跨度或忽略时间跨度
- 截面数据样本不平衡处理：（ADASYN（Adaptive Synthetic Sampling））优点：可调用现成的包。在示例密度低的区域生成更多的合成示例

```
Counter({0: 55875, 1: 1961})
Counter({1: 56130, 0: 55875})
```

截面数据

Mechine Learning

LightGBM　Xgboost　Logestic

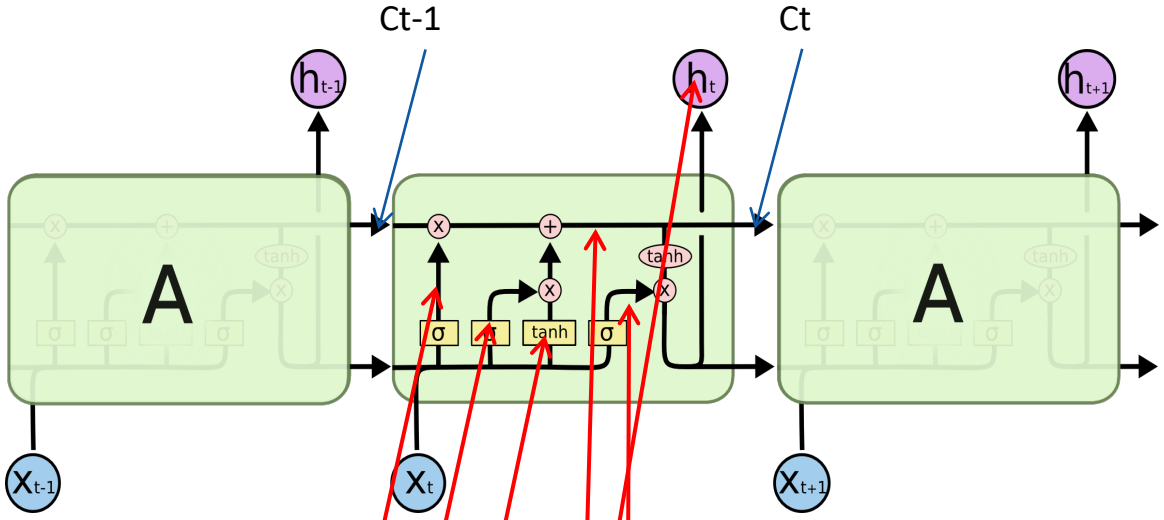
面板数据

Deep Learning

LSTM　GAT

3.1 模型建立|深度学习

print the presentation



$$f^t = \sigma(W^f x^t + U^f h^{t-1})$$
$$i^t = \sigma(W^i x^t + U^i h^{t-1})$$
$$o^t = \sigma(W^o x^t + U^o h^{t-1})$$
$$\tilde{c}^t = tanh(W^c x^t + U^c h^{t-1})$$
$$c^t = f^t \odot c^{t-1} + i^t \odot \tilde{c}^t$$
$$h_t = o^t \odot tanh(c^t)$$

内部流程
上个状态向下个状态传递Ct-1、ht-1
本期输入Xt
同时　ht-1与Xt进行拼接

①遗忘信息ft：选择上阶段忘记什么信息
②选择信息it：选择本期学习什么信息
③更新状态~Ct：本期更新后的Ct
④输出Ct：通过遗忘和选择两个阶段，对上阶段的Ct-1及本期更新的~Ct进行计算得到新的Ct，进而传递到下个状态

①输出门ot
②计算隐含阶段的输出ht
由ot及生成的Ct计算出

预测Y值由ht计算得到

3.1 模型建立|LSTM

model establishment

```python
class LSTM(nn.Module):

    def __init__(self, input_size, output_size, num_classes=2, classification=False):
        # 输入维度（dataframe的特征维度）；隐藏层设置
        super(LSTM, self).__init__()  # 参数编写
        self.hidden_size = output_size
        self.cell_size = output_size  # 与GRU不同
        self.tanh = nn.Tanh()
        self.sigmoid = nn.Sigmoid()
        self.gate = nn.Linear(input_size + output_size, output_size)
        self.classification = classification
        if self.classification:
            self.output_dense = nn.Linear(output_size, num_classes)

    def forward(self, x, h_t, c_t):  # 对应上面公式
        combined = torch.cat((x, h_t), 1)

        f = self.sigmoid(self.gate(combined))
        i = self.sigmoid(self.gate(combined))
        o = self.sigmoid(self.gate(combined))
        c = torch.add(torch.mul(c_t, f), torch.mul(self.tanh(self.gate(combined)), i))

        h = torch.mul(self.tanh(c), o)
        if self.classification:
            output = self.output_dense(h)
        else:
            output = h

        return output, h, c

    def init_hidden(self):
        return Variable(torch.zeros(1, self.hidden_size))  # 初始化

    def init_cell(self):
        return Variable(torch.zeros(1, self.cell_size))  # 初始化
```

输入数据：

①不考虑时间序列，仅当作面板数据输入

且将公司首次被ST前四个季度均标为ST

②考虑时间序列，选择有完整5年时间的股票进行训练

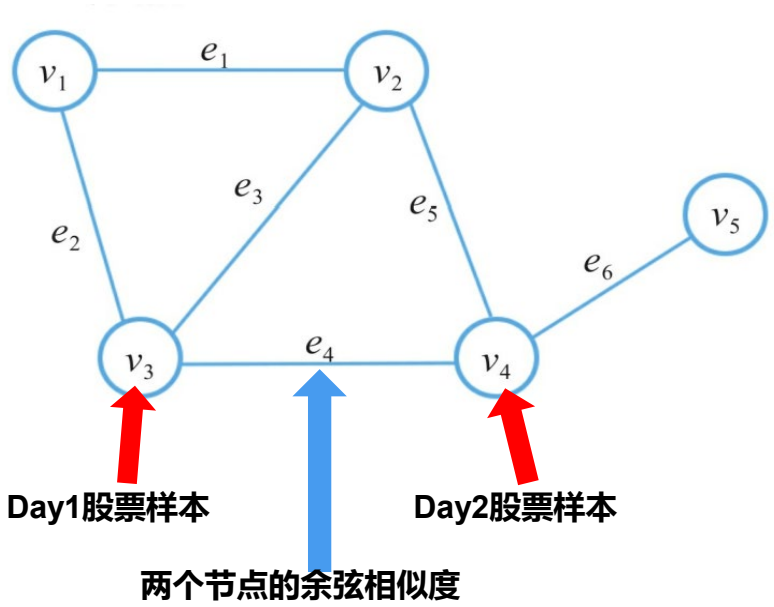
输入数据中ST与非ST比例在25：1左右

模型结果

	Accuracy	Precision	AUC
①不考虑时间序列	72.90%	51.96%	62.20%
②考虑时间序列	94.41%	56.25%	54.82%
③GAT	74.19%	26.44%	52.71%

3.1 模型建立|GAT

model establishment



对行数据归一化，保证所有邻居的权重系数加和为1

对角线加1：`adj + sp.eye(adj.shape[0])`

Adj:邻接矩阵

	v_1	v_2	v_3
v_1		0.51	0.32
v_2	0.02		0.61
v_3	0.06	0.68	

样本量*样本量

定义节点为季度的ST数据，图中的边由torch.cosine_similarity()方法构建，
sim_threshold表示连接边的阈值，股票间关系小于这一阈值的边都不计入图

```
for i, j in index_combinations:
    cos_sim = torch.cosine_similarity(torch.Tensor(idx_features_labels[:, :-1][i]), torch.Tensor(idx_features_labels[:, :-1][j]), dim=0)
    if cos_sim >= args.sim_threshold:
        adj[i][j] = cos_sim.numpy()
```

3.1 模型建立|GAT

model establishment

参数设定

```python
import argparse

parser = argparse.ArgumentParser(description='Args for Stock_GAT_predict')
parser.add_argument('--cuda', default=True)
parser.add_argument('--fastmode', action='store_true', default=False, help='Validate during training pass.')
parser.add_argument('--sparse', action='store_true', default=False, help='GAT with sparse version or not.')
parser.add_argument('--seed', type=int, default=2022, help='Random seed.')
parser.add_argument('--epochs', type=int, default=1000, help='Number of epochs to train.')
parser.add_argument('--lr', type=float, default=0.005, help='Initial learning rate.')
parser.add_argument('--weight_decay', type=float, default=5e-4, help='Weight decay (L2 loss on parameters).')
parser.add_argument('--hidden', type=int, default=8, help='Number of hidden units.')
parser.add_argument('--nb_heads', type=int, default=3, help='Number of head attentions.')
parser.add_argument('--dropout', type=float, default=0.6, help='Dropout rate (1 - keep probability).')
parser.add_argument('--alpha', type=float, default=0.2, help='Alpha for the leaky_relu.')
parser.add_argument('--patience', type=int, default=100, help='Patience')
parser.add_argument('--sim_threshold', type=float, default=0.6)

args = parser.parse_args()
```

nb_heads：多头图注意力层，调用K组相互独立的注意力机制，然后将输出结果拼接或平均。能够将注意力的分配放到中心节点与邻居,节点之间多处相关的特征上，可使得系统的学习能力更加强大。

原论文以及衍生应用中均设置nb_heads=8，但我们尝试发现该场景中设置为3效果最佳

A two-layer GAT model

我们采用两层GAT模型的架构。
第一层由K = 3个注意头组成，每个注意头计算F= 8个特征(总共24个特征)，然后连接一个指数线性单元(ELU)。
第二层用于分类:单个注意力头计算C特征(C是Label的种类数量)，然后激活log_softmax。

经济解释

对股票数据而言，某两个样本（行）的相似度高意味着价、量、基本面情况等处于高度类似的状态，那么我们有理由认为历史不会重复但会惊人的相似，所以我们利用相似的历史走势做出对未来是否会st的预测是有道理的。

4.1 论文总结
Summary

模型结果		
	Accuracy	AUC
不考虑时间序列	72.90%	62.20%
考虑时间序列	94.41%	54.82%
GAT	74.19%	52.71%
Lightgbm	92.65%	90.16%

◆ 展望

文本特征：非结构化的文本大数据（情绪因子）
管理特征：股东持股比例
基于企业关系图的特征：LSTM+GCN

恳请老师 批评指正

THANKS FOR LISTENING

西南财经大学

Repoter: Wenjie Lan

Mentor: Jun Wang

Time: 2022-8-21