股权特征与企业市场价值

吉小浩

2024-05-04

目 录

说 明

为方便大家做数据分析报告，邀请刘瑞雪和吉小浩两同学基于《课程作业模板》各做一份数据分析报告。

欢迎大家阅读、学习和参考，并结合自己的选题完成《Python 数据分析》课程数据分析报告。

《课程作业模板》 在学习通中，请下载使用。

欢迎学习借鉴，鼓励创新，禁止抄袭。

授课教师：程 军

2024 年 5 月 5 日

本分析报告的实验环境设置如下：

```
! python --version
! pip list | findstr "pandas numpy scikit-learn"
```

```
Python 3.12.3
numpy                     1.26.4
pandas                    2.2.2
scikit-learn              1.4.2
```

```
from sklearn.model_selection import train_test_split, cross_val_score,R
andomizedSearchCV
from IPython.core.interactiveshell import InteractiveShell
from sklearn.metrics import (
    median_absolute_error,
    mean_absolute_error,
    mean_squared_error,
    r2_score,
)
import pandas as pd
import numpy as np
from pprint import pprint
```

```
from IPython.core.interactiveshell import InteractiveShell

InteractiveShell.ast_node_interactivity = "all"

import matplotlib_inline

matplotlib_inline.backend_inline.set_matplotlib_formats("svg")

import warnings
warnings.filterwarnings('ignore')
```

1. 引言

1.1 研究问题

本分析报告主要关注 A 股上市公司的股权特征的典型事实及其对企业市场价值的影响。

股权特征：

(1)两权分离度的变化趋势是怎样的？

(2)机构投资者持股特征是怎样的？

(3)国企与非国企，制造业与非制造业的股权集中度是怎样的？

与市值关系：

(4)哪一项股权特征更为重要？

1.2 研究意义

股权特征反映了企业内部控制权的分配和股东之间的利益关系，而企业市值则是市场对企业整体价值的评估。通过研究股权特征与企业市值的关系，可以深入了解企业治理结构对企业价值的影响，以及不同股权结构下企业市值变动的规律。

通过研究，可以发现不同股权特征对企业市值的影响程度和机制。有助于企业根据自身的实际情况，优化股权结构，提高公司治理效率，从而提升企业市值。例如，合理调整国有股、法人股和社会公众股的比例，可以平衡不同股东之间的利益，增强企业的市场竞争力。

通过研究股权特征与企业市值的关系，可以为投资者提供重要的决策参考，帮助他们更准确地评估企业的投资价值和风险。

股权特征和企业市值之间的关系研究，有助于为政策制定者提供有益的参考，帮助他们制定更加科学合理的政策，推动资本市场的健康稳定发展。

1.3 研究思路

本报告旨在探讨股权特征对企业市值的影响。分析实际控制人性质、股东持股比例、机构持股情况等股权特征与企业市值之间的关系。同时，本研究将采用梯度提升决策树（GBDT）和 Lasso 回归模型进行模型分析。

2. 数据准备

2.1 数据来源

本报告使用的数据主要是上市公司中的财务数据和股权特征两部分。数据来自国泰安数据库（CAMAR）。

2.2 样本选择

数据区间：以数据起始时间为起点，截止到 2023 年。其中，参考常规做法，对于数据我们进行了 ST、金融业的剔除。在数据清理期间，填补可填补的数据，删除缺失较多的数据。最终形成 327567 个样本。

其中，

被解释变量为企业市场价值，本报告选择现有文献中常用指标托宾 Q 值。

解释变量即股权特征，选取机构持股数量，机构持股比例，股股东直接持有上市公司股份的数量，公司前 3 位大股东持股比例之和，两权分离率等。

控制变量，选择企业规模，无形资产占比，资产负债率，第一大股东持股比例之和等。

2.3 合并数据

2.3.1 财务指标数据

```
#财务指标
df_fi=pd.read_excel("rawdata\财务指标\FI_T10.xlsx",skiprows=[1,2],header=0)
df_fi.head()
```

	Stkcd	Short Name	Acc per	Indcd	Indnme	F100101B	F10080 1A	F100901A	F100902A	F100903A	F100904A
0	2	深万科 A	1991-12-31	K70	房地产业	NaN	1.505167e+09	2.647639	2.648996	3.243873	3.245536
1	2	深万	199	K7	房	NaN	2.44652	2.587	2.821	3.162	3.447

	Stkcd	ShortName	Accper	Indcd	Indnme	F100101B	F100801A	F100901A	F100902A	F100903A	F100904A
		科 A	2-12-31	0	地产业		2e+09	538	200	024	563
2	2	深万科 A	1993-12-31	K70	房地产业	NaN	3.541231e+09	1.657757	1.659930	1.879755	1.882219
3	2	深万科 A	1994-12-31	K70	房地产业	7.966339	2.797435e+09	1.045697	1.058723	1.073715	1.087090
4	2	深万科 A	1995-12-31	K70	房地产业	6.027127	2.933862e+09	0.907087	0.914923	0.881404	0.889018

```
df_fi["year"]=df_fi["Accper"].str.slice(0,4).astype("int64")

#新建'manufacturingindustry'，根据行业名称，含有制造业，则赋值为 1，其余赋值为 0
df_fi['manufacturingindustry'] = df_fi['Indnme'].str.contains('制造业',
 na=False).astype(int)

df_fi.columns
df_fi.drop(["Accper"],axis=1,inplace=True)
```

```
Index(['Stkcd', 'ShortName', 'Accper', 'Indcd', 'Indnme', 'F100101B',
       'F100801A', 'F100901A', 'F100902A', 'F100903A', 'F100904A', 'year',
       'manufacturingindustry'],
      dtype='object')
```

2.3.2 股权特征数据

```
#股权特征
df_cs=pd.read_excel("rawdata\公司治理\HLD_Contrshr.xlsx",skiprows=[1,2],header=0)
df_cr=pd.read_excel("rawdata\公司治理\HLD_CR.xlsx",skiprows=[1,2],header=0)
df_hi=pd.read_excel("rawdata\公司治理\INI_Holder_IncrOrDesr.xlsx",skiprows=[1,2],header=0)

df_cs.head()
```

```
df_cr.head()
df_hi.head()
```

	Stkcd	Reptdt	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation
0	2	2003-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN
1	2	2003-12-31	2	华润股份有限公司	中国华润总公司	国有法人股,募集法人股	11.19	156151498.0	1100	NaN
2	2	2004-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN
3	2	2004-12-31	2	华润股份有限公司	中国华润总公司	国有法人股,募集法人股,流通A股	12.89	293084169.0	1100	NaN
4	2	2005-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN

	Stkcd	Reptdt	Shrcr1	Shrcr2	Shrcr3
0	2	2003-12-31	11.1868	15.3752	17.4711
1	2	2004-12-31	12.8906	17.1627	21.0274
2	2	2005-12-31	11.8094	15.5075	18.5765
3	2	2006-12-31	14.5428	18.8346	21.4614
4	2	2007-12-31	14.6345	16.9825	19.0465

	InstitutionID	Symbol	EndDate	Status	HolderNumber	Shares	Proportion
0	101775	2	1999-12-31	1	1	3376765.0	0.618979
1	101775	2	1999-12-31	3	1	689200.0	0.126334
2	101775	2	2000-12-31	0	1	6200000.0	0.982611
3	101775	2	2000-12-31	2	1	2287361.0	0.362514
4	101775	2	2000-	3	12	210081	3.32949

	Stkcd	Reptdt	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation
				12-31				59.0	3	

```
df_cs=df_cs.rename(columns={'Reptdt':'Accper'})
df_cr=df_cr.rename(columns={'Reptdt':'Accper'})
df_hi=df_hi.rename(columns={'EndDate':'Accper','Symbol':'Stkcd'})

df_cs.head()
df_cr.head()
df_hi.head()
```

	Stkcd	Accper	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation
0	2	2003-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN
1	2	2003-12-31	2	华润股份有限公司	中国华润总公司	国有法人股,募集法人股	11.19	156151498.0	1100	NaN
2	2	2004-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN
3	2	2004-12-31	2	华润股份有限公司	中国华润总公司	国有法人股,募集法人股,流通A股	12.89	293084169.0	1100	NaN
4	2	2005-12-31	1	0	0	NaN	NaN	NaN	NaN	NaN

	Stkcd	Accper	Shrcr1	Shrcr2	Shrcr3
0	2	2003-12-31	11.1868	15.3752	17.4711
1	2	2004-12-31	12.8906	17.1627	21.0274
2	2	2005-12-31	11.8094	15.5075	18.5765
3	2	2006-12-31	14.5428	18.8346	21.4614
4	2	2007-12-31	14.6345	16.9825	19.0465

	InstitutionID	Stkcd	Accper	Status	HolderNumber	Shares	Proportion

	Stkcd	Accper	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation
0	101775	2		1999-12-31	1	1		3376765.0		0.618979
1	101775	2		1999-12-31	3	1		689200.0		0.126334
2	101775	2		2000-12-31	0	1		6200000.0		0.982611
3	101775	2		2000-12-31	2	1		2287361.0		0.362514
4	101775	2		2000-12-31	3	12		21008159.0		3.329493

```
df_sh1=pd.merge(df_cs,df_cr,how='outer',on=['Stkcd','Accper'])
df_sh=pd.merge(df_hi,df_sh1,how='outer',on=['Stkcd','Accper'])

df_sh.head()
```

	InstitutionID	Stkcd	Accper	Status	HolderNumber	Shares	Proportion	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation	Shrcr1	Shrcr2	Shrcr3
0	1017750	2	1999-12-31	1.0	1.0	337676765.0	0.618979	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	1017750	2	1999-12-	3.0	1.0	68920000	0.126334	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	InstitutionID	Stkcd	Accper	Status	HolderNumber	Shares	Proportion	S0701a	S0702a	S0701b	S0703a	S0704a	S0705a	S0702b	Seperation	Shrcr1	Shrcr2	Shrcr3
			31															
2	1017750	2	2000-12-31	0.0	1.0	6200000.0	0.982611	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	1017750	2	2000-12-31	2.0	1.0	2287361.0	0.362514	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	1017750	2	2000-12-31	3.0	12.0	2100815	3.329493	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
#新建'PropertyRightsNature'列，根据实际控制人性质，若为国有性质，赋值为 1【1100 为国企，2000 为行政机关、事业单位，2100 为中央机构，2120 为地方机构】，其中当实际控制人若有多个，只要其中之一是国有企业，判断为 1；非国有性质，则赋值 0
df_sh['PropertyRightsNature'] = df_sh['S0702b'].str.contains('1100|2000|2100|2120', case=False, na=False).astype(int)

df_sh["accper"]=df_sh["Accper"].str.slice(0,4).astype("int64")

df_sh.columns
df_sh.drop(["Accper"],axis=1,inplace=True)

Index(['InstitutionID', 'Stkcd', 'Accper', 'Status', 'HolderNumber', 'Shares',
       'Proportion', 'S0701a', 'S0702a', 'S0701b', 'S0703a', 'S0704a',
```

```
       'S0705a', 'S0702b', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3',
       'PropertyRightsNature', 'accper'],
      dtype='object')
```

2.3.3 相关控制变量数据

```
df_cas=pd.read_excel('rawdata\控制变量\FS_Combas.xlsx',skiprows=[1,2],he
ader=0)
df_cis=pd.read_excel('rawdata\控制变量\FS_Comins.xlsx',skiprows=[1,2],he
ader=0)
df_ena=pd.read_excel('rawdata\控制变量\EN_EquityNatureAll.xlsx',skiprows
=[1,2],header=0)
df_ipo=pd.read_excel('rawdata\控制变量\IPO_IPOCG.xlsx',skiprows=[1,2],he
ader=0)
df_stk=pd.read_excel('rawdata\控制变量\STK_LISTEDCOINFOANL.xlsx',skiprow
s=[1,2],header=0)

df_cas.head()
df_cis.head()
df_ena.head()
df_ipo.head()
df_stk.head()
```

	Stkcd	Short Name	Accper	Typrep	A001212000	A001218000	A001000000	A002000000	A003200000
0	2	深万科 A	1991-12-31	A	7.195234e+07	291259.41	5.684941e+08	3.778345e+08	9.259519e+04
1	2	深万科 A	1992-12-31	A	9.414705e+07	78309636.35	9.455016e+08	7.082929e+08	1.781863e+06
2	2	深万科 A	1993-12-31	A	1.653931e+08	2797303.07	2.136159e+09	1.205853e+09	1.887863e+06
3	2	深万科 A	1994-12-31	A	2.995394e+08	32915749.20	2.675188e+09	1.527296e+09	5.852032e+07
4	2	深万科 A	1995-12-31	A	2.470931e+08	27702130.26	3.234377e+09	1.872121e+09	1.827642e+08

	Stkcd	ShortName	Accper	Typrep	B001101000	B002000000
0	2	深万科 A	1991-12-31	A	NaN	NaN
1	2	深万科 A	1992-12-31	A	6.613562e+08	NaN
2	2	深万科 A	1993-12-31	A	1.084045e+09	NaN
3	2	深万科 A	1994-12-31	A	1.227544e+09	1.829864e+08
4	2	深万科 A	1995-12-31	A	1.503755e+09	1.668964e+08

	Symbol	ShortName	EndDate	LargestHolderRate	TopTenHoldersRate
0	2	万科 A	2003-12-31	11.19	21.45
1	2	万科 A	2004-12-31	12.89	28.17
2	2	G 万科 A	2005-12-31	11.81	24.91
3	2	万科 A	2006-12-31	14.54	27.09
4	2	万科 A	2007-12-31	14.63	22.71

	Symbol	IPODate	DirectorNumber	IndDirectorNum	IsPartTime
0	1201	2021-04-14	5.0	NaN	1
1	1202	2021-04-19	10.0	6.0	1
2	1203	2021-04-20	3.0	NaN	0
3	1205	2021-04-28	8.0	NaN	0
4	1206	2021-05-06	10.0	3.0	1

	Symbol	ShortName	EndDate	EstablishDate
0	2	深万科 A	2000-12-31	1988-11-01
1	2	深万科 A	2001-12-31	1988-11-01
2	2	万科 A	2002-12-31	1988-11-01
3	2	万科 A	2003-12-31	1988-11-01
4	2	万科 A	2004-12-31	1988-11-01

合并表格

```
df_fs=pd.merge(df_cas,df_cis,how="outer",on=["Stkcd","Accper"])
df_fs["year"]=df_fs["Accper"].str.slice(0,4).astype("int64")


df_1=pd.merge(df_ena,df_stk,how="outer",on=["Symbol","EndDate"])
df_1=df_1.rename(columns={'EndDate':'Accper','Symbol':'Stkcd'})
df_1["accper"]=df_1["Accper"].str.slice(0,4).astype("int64")


df_ipo=df_ipo.rename(columns={'Symbol':'Stkcd'})
df_ipo["accper"]=df_ipo["IPODate"].str.slice(0,4).astype("int64")

df_fs.columns
df_fs.drop(["ShortName_x",'Typrep_x','ShortName_y','Typrep_y'],axis=1,i
nplace=True)

Index(['Stkcd', 'ShortName_x', 'Accper', 'Typrep_x', 'A001212000',
       'A001218000', 'A001000000', 'A002000000', 'A003200000', 'ShortNa
me_y',
       'Typrep_y', 'B001101000', 'B002000000', 'year'],
      dtype='object')

df_1.columns
df_1.drop(["ShortName_x",'ShortName_y'],axis=1,inplace=True)

Index(['Stkcd', 'ShortName_x', 'Accper', 'LargestHolderRate',
       'TopTenHoldersRate', 'ShortName_y', 'EstablishDate', 'accper'],
      dtype='object')

df_ipo.columns

Index(['Stkcd', 'IPODate', 'DirectorNumber', 'IndDirectorNum', 'IsPartT
ime',
       'accper'],
      dtype='object')

df_kz=pd.merge(df_fs,df_1,how="left",left_on=["Stkcd","year"],right_on=
["Stkcd","accper"])
df_kz=df_kz.merge(df_ipo,how="left",left_on=["Stkcd","year"],right_on=
["Stkcd","accper"])

df_kz.head()
```

	Stkcd	Accper_x	A001212000	A001218000	A001100000	A002000000	A003200000	B001100000	B002000000	year	Accper_y	LargestHolderRate	TopTenHoldersRate	EstablishDate	accper_x	IPODate	DirectorNumber	IndDirectorNum	IsPartTime	accper_y
0	2	1991-12-31	7.19152e+07	29125941	5.68494e+08	3.77834e+08	9.25951e+08	NaN	NaN	1991	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	2	1992-12-31	9.41470e+07	78309635	9.45501e+08	7.08290e+08	1.78186e+08	6.61356e+07	NaN	1992	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2	1993-12-31	1.65393e+08	27973035	2.13615e+09	1.20585e+09	1.88786e+08	1.08404e+08	NaN	1993	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2	1	2.	3	2.	1.	5.	1.	1.	1	N	Na	Na	N	N	N	Na	Na	N	N

	Stkcd	Accper_x	A001212000	A001218000	A001000000	A002000000	A003200000	B001101000	B002000000	year	Accper_y	LargestHolderRate	TopTenHoldersRate	EstablishDate	accper_x	IPODate	DirectorNumber	IndDirectorNum	IsPartTime	accper_y
	994	994-12-31	2.97518e+08	6.72897e+08	5.27203e+08	8.52744e+08	2.27032e+08	8.29864e+08	1994	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	2	1995-12-31	2.47703e+08	3.23437e+08	1.87216e+08	1.82705e+08	1.50368e+08	1.66896e+08	1995	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
df_kz.columns
df_kz.drop(['Accper_x','Accper_y','accper_x','accper_y'],axis=1,inplace=True)

Index(['Stkcd', 'Accper_x', 'A001212000', 'A001218000', 'A001000000',
       'A002000000', 'A003200000', 'B001101000', 'B002000000', 'year',
       'Accper_y', 'LargestHolderRate', 'TopTenHoldersRate', 'EstablishDate',
       'accper_x', 'IPODate', 'DirectorNumber', 'IndDirectorNum', 'IsPartTime',
       'accper_y'],
      dtype='object')
```

2.3.4 合并数据

```
#合并数据
df=pd.merge(df_fi,df_sh,how='left',left_on=["Stkcd","year"],right_on=["Stkcd","accper"])
df=df.merge(df_kz,how='left',left_on=["Stkcd","year"],right_on=["Stkcd","year"])
```

2.4 清洗数据

```
df.columns
```

```
Index(['Stkcd', 'ShortName', 'Indcd', 'Indnme', 'F100101B', 'F100801A',
       'F100901A', 'F100902A', 'F100903A', 'F100904A', 'year',
       'manufacturingindustry', 'InstitutionID', 'Status', 'HolderNumber',
       'Shares', 'Proportion', 'S0701a', 'S0702a', 'S0701b', 'S0703a',
       'S0704a', 'S0705a', 'S0702b', 'Seperation', 'Shrcr1', 'Shrcr2',
       'Shrcr3', 'PropertyRightsNature', 'accper', 'A001212000', 'A001218000',
       'A001000000', 'A002000000', 'A003200000', 'B001101000', 'B002000000',
       'LargestHolderRate', 'TopTenHoldersRate', 'EstablishDate', 'IPODate',
       'DirectorNumber', 'IndDirectorNum', 'IsPartTime'],
      dtype='object')
```

```
df.head()
```

	Stkcd	ShortName	Indcd	Indnme	F100101B	F100801A	F100901A	F100902A	F100903A	F100904A	...	A003200000	B001101000	B002000000	LargestHolderRate	TopTenHoldersRate	EstablishDate	IPODate	DirectorNumber	IndDirectorNum	IsPartTime
0	2	深万科A	K70	房地产业	NaN	1.505167e+07	2.647639e+09	2.648973e+09	3.243873e+09	3.245536e+09	...	9.259519e+09	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	Stkcd	ShortName	Indcd	Indnm	F100101B	F100801A9	F100901A	F100902A	F100903A	F100904A	...	A003200000	B001101000	B002000000	LargestHolderRate	TopTenHoldersRate	EstablishDate	IPODate	DirectorNumber	IndDirectorNum	IsPartTime
1	2	深万科A	K70	房地产业	NaN	2.446522e+09	2.587538	2.821200	3.162024	3.447563	...	1.781863e+06	6.613562e+08	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	2	深万科A	K70	房地产业	NaN	3.541231e+09	1.655775	1.659930	1.879755	1.882219	...	1.887863e+06	1.084045e+09	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	2	深万科A	K70	房地产业	7.963639	2.797435e+09	1.045697	1.058723	1.073715	1.087090	...	5.852032e+07	1.227544e+09	1.829864e+08	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

	Stkcd	ShortName	Indcd	Indnme	F100101B	F100801A	F100901A	F100902A	F100903A	F100904A	...	A003200000	B001101000	B002000000	LargestHolderRate	ToPTenHoldersRate	EstablishDate	IPODate	DirectorNumber	IndDirectorNum	IsPartTime
42	深万科A	K70	房地产业	6.027127	2.933862e+09	0.907087	0.914923	0.881404	0.889018	...	1.827642e+08	1.503755e+09	1.668964e+08	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
#清洗前样本量
df.shape
```

```
(362440, 44)
```

```
#删除 ShortName 中含有 s,st,*st 的样本
df = df[~df['ShortName'].str.contains(r's|st|\\*st', case=False)]
```

```
#删除 s,st,*st 的样本后
df.shape
```

```
(354324, 44)
```

```
df.isnull().sum()
```

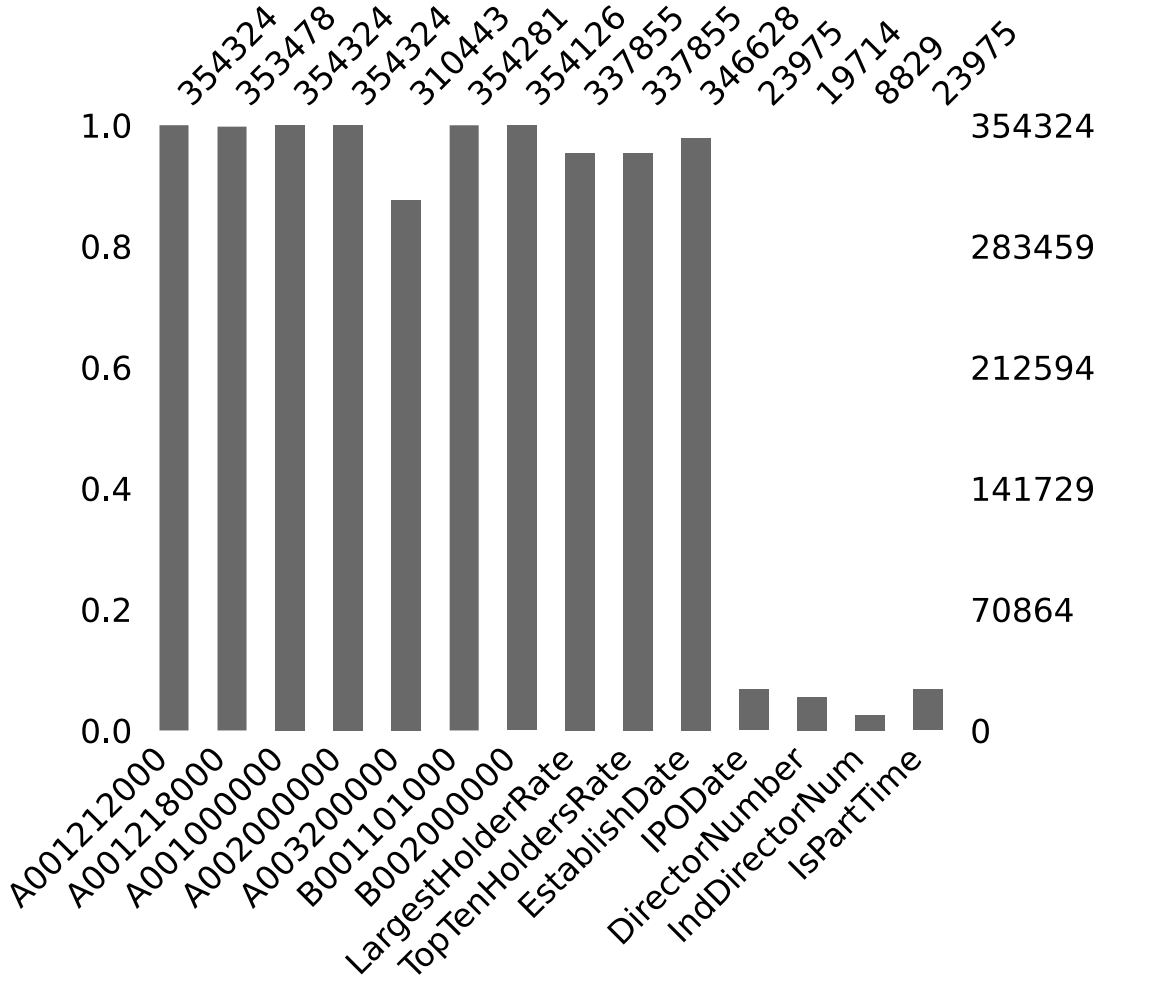
```
Stkcd                        0
ShortName                    0
Indcd                        0
Indnme                       0
F100101B                 34353
F100801A                     0
F100901A                     0
F100902A                     0
F100903A                     0
F100904A                     0
year                         0
manufacturingindustry        0
```

```
InstitutionID              4587
Status                     4587
HolderNumber               4587
Shares                     4587
Proportion                 4587
S0701a                    17443
S0702a                    17443
S0701b                    17443
S0703a                    23660
S0704a                    23694
S0705a                    23713
S0702b                    23737
Seperation                31493
Shrcr1                     6974
Shrcr2                     6974
Shrcr3                     6974
PropertyRightsNature       4402
accper                     4402
A001212000                    0
A001218000                  846
A001000000                    0
A002000000                    0
A003200000                43881
B001101000                   43
B002000000                  198
LargestHolderRate         16469
TopTenHoldersRate         16469
EstablishDate              7696
IPODate                  330349
DirectorNumber           334610
IndDirectorNum           345495
IsPartTime               330349
dtype: int64
```

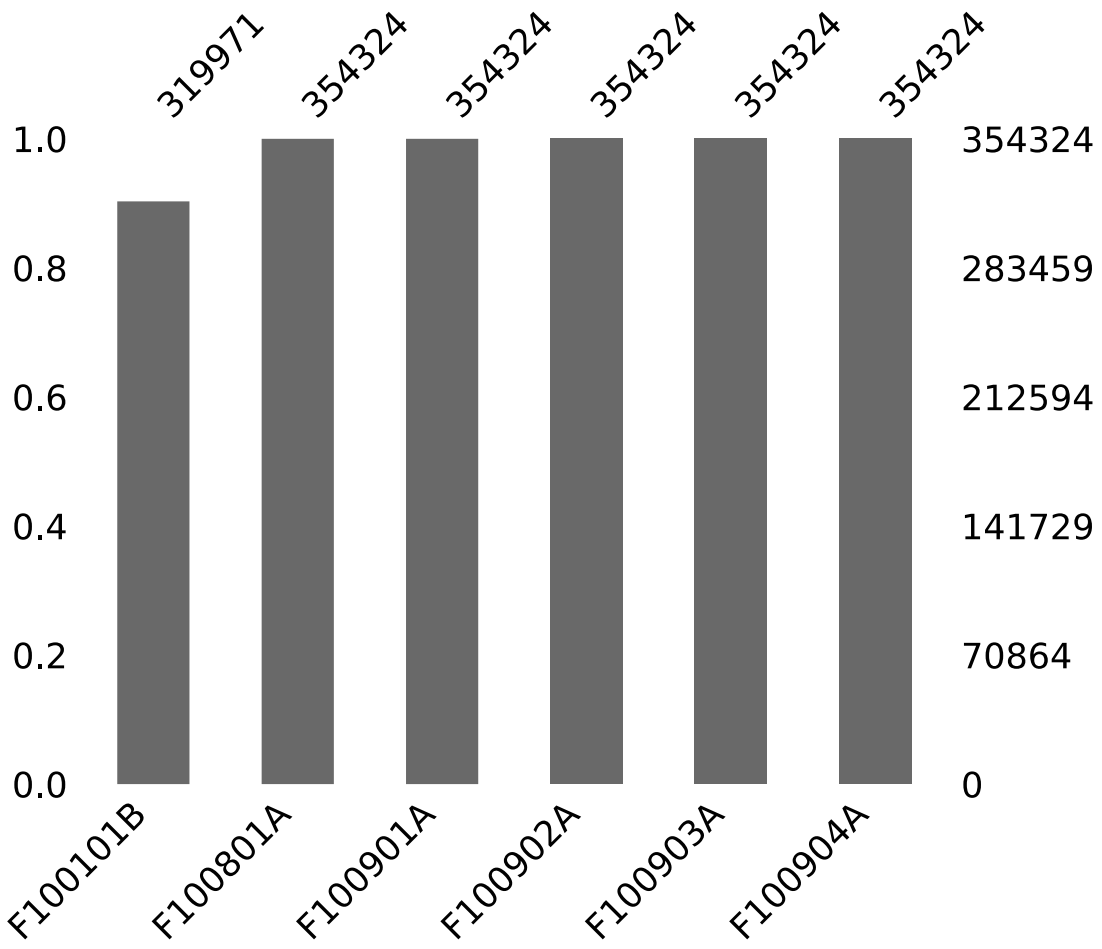
```
df.columns
```

```
Index(['Stkcd', 'ShortName', 'Indcd', 'Indnme', 'F100101B', 'F100801A',
       'F100901A', 'F100902A', 'F100903A', 'F100904A', 'year',
       'manufacturingindustry', 'InstitutionID', 'Status', 'HolderNumbe
r',
       'Shares', 'Proportion', 'S0701a', 'S0702a', 'S0701b', 'S0703a',
       'S0704a', 'S0705a', 'S0702b', 'Seperation', 'Shrcr1', 'Shrcr2',
       'Shrcr3', 'PropertyRightsNature', 'accper', 'A001212000', 'A0012
18000',
       'A001000000', 'A002000000', 'A003200000', 'B001101000', 'B002000
000',
       'LargestHolderRate', 'TopTenHoldersRate', 'EstablishDate', 'IPOD
ate',
       'DirectorNumber', 'IndDirectorNum', 'IsPartTime'],
      dtype='object')
```

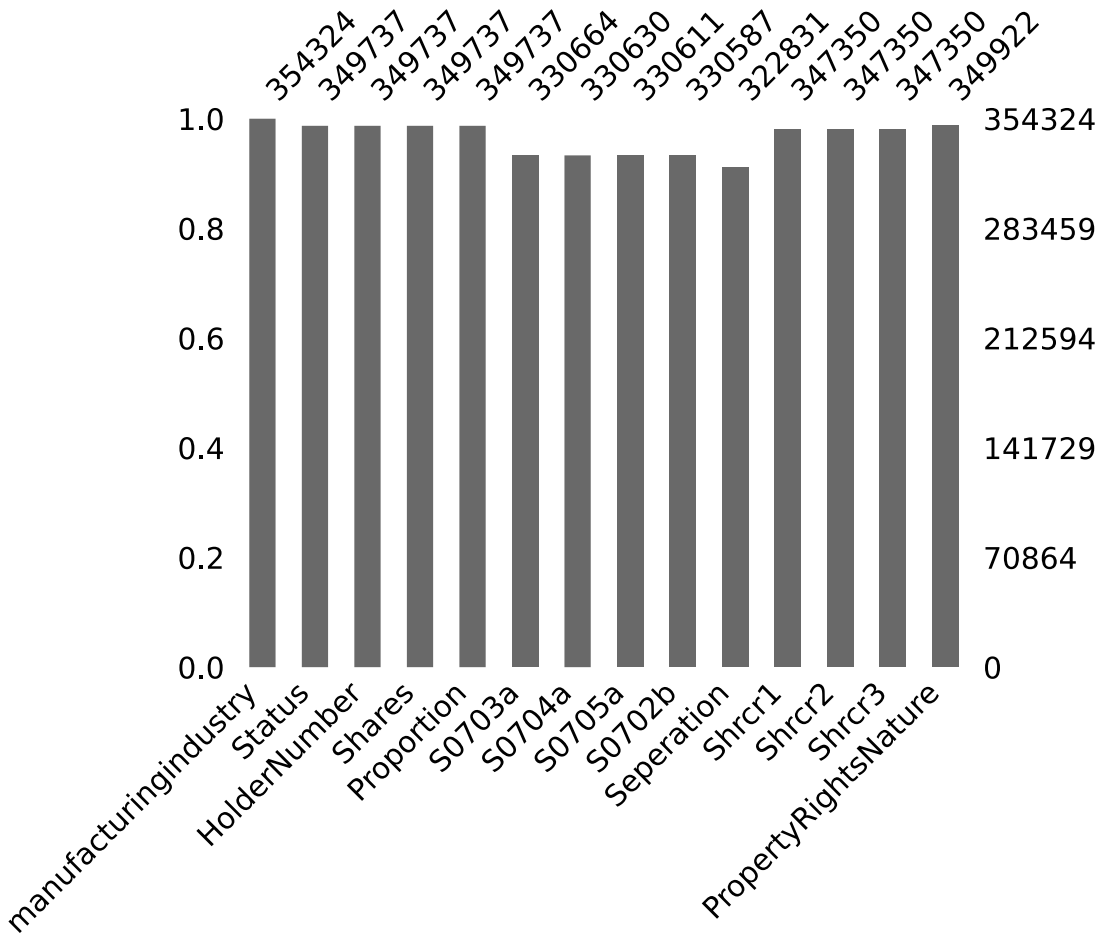
```
#控制相关变量缺失值可视化
import missingno as msno
msno.bar(df=df[['A001212000', 'A001218000',
       'A001000000', 'A002000000', 'A003200000', 'B001101000', 'B002000
000',
       'LargestHolderRate', 'TopTenHoldersRate', 'EstablishDate', 'IPOD
ate',
       'DirectorNumber', 'IndDirectorNum', 'IsPartTime']],figsize=(8,6),
fontsize=18)
```




```
#'IPODate','DirectorNumber', 'IndDirectorNum', 'IsPartTime'缺失值较多，
同时'A003200000'由于在后续控制变量计算中需作分母，无法填补为0，直接删除
df.drop(['A003200000','IPODate','DirectorNumber', 'IndDirectorNum', 'Is
PartTime'],axis=1,inplace=True)

#市值指标缺失值可视化
import missingno as msno
msno.bar(df=df[['F100101B', 'F100801A',
       'F100901A', 'F100902A', 'F100903A', 'F100904A']],figsize=(8,6),f
ontsize=18)
```




```
#股权特征缺失值可视化
import missingno as msno
msno.bar(df=df[['manufacturingindustry','Status', 'HolderNumber', 'Shar
es', 'Proportion', 'S0703a', 'S0704a', 'S0705a', 'S0702b', 'Seperation',
       'Shrcr1', 'Shrcr2', 'Shrcr3','PropertyRightsNature']],figsize=(8,
6),fontsize=18)
```




```
# F100101B [市盈率（PE）1] 缺失过多，删除
df.drop(["F100101B"],axis=1,inplace=True)

#填补股权特征中实际股权性质，两权分离度，直接控股股东持股比例,机构持股比例缺失值
df["PropertyRightsNature"]=df["PropertyRightsNature"].fillna(0)
df["Seperation"]=df["Seperation"].fillna(0)
df["S0704a"]=df["S0704a"].fillna(0)
df["Proportion"]=df["Proportion"].fillna(0)

#其余缺失，直接删除
df=df.dropna()

df.isnull().sum()

Stkcd                   0
ShortName               0
Indcd                   0
Indnme                  0
F100801A                0
F100901A                0
F100902A                0
```

```
F100903A                  0
F100904A                  0
year                      0
manufacturingindustry     0
InstitutionID             0
Status                    0
HolderNumber              0
Shares                    0
Proportion                0
S0701a                    0
S0702a                    0
S0701b                    0
S0703a                    0
S0704a                    0
S0705a                    0
S0702b                    0
Seperation                0
Shrcr1                    0
Shrcr2                    0
Shrcr3                    0
PropertyRightsNature      0
accper                    0
A001212000                0
A001218000                0
A001000000                0
A002000000                0
B001101000                0
B002000000                0
LargestHolderRate         0
TopTenHoldersRate         0
EstablishDate             0
dtype: int64
```

计算控制变量

```
df.columns
```

```
Index(['Stkcd', 'ShortName', 'Indcd', 'Indnme', 'F100801A', 'F100901A',
       'F100902A', 'F100903A', 'F100904A', 'year', 'manufacturingindust
ry',
       'InstitutionID', 'Status', 'HolderNumber', 'Shares', 'Proportion
',
       'S0701a', 'S0702a', 'S0701b', 'S0703a', 'S0704a', 'S0705a', 'S07
02b',
       'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNatur
e',
       'accper', 'A001212000', 'A001218000', 'A001000000', 'A002000000',
       'B001101000', 'B002000000', 'LargestHolderRate', 'TopTenHoldersR
ate',
```

```
       'EstablishDate'],
      dtype='object')

df['lnAssets']=np.log(df['A001000000'])
df['Lev']=df['A002000000']/df['A001000000']
df['PPE_TA']=df['A001212000']/df['A001000000']
df['Intass']=df['A001218000']/df['A001000000']
df['OwnCon1']=df['LargestHolderRate']
df['OwnCon2_10']=df['TopTenHoldersRate']-df['LargestHolderRate']
df['SOE']=df['PropertyRightsNature']
df['ROA']=df['B002000000']/df['A001000000']

df["Establish_year"]=df["EstablishDate"].str.slice(0,4).astype("int64")
df['Age']=df['year']-df['Establish_year']

#.shift(1) 用于将数据沿轴（默认是行轴，即 axis=0）移动指定的数量。在这里，1 表示向下移动一行
df['Last_Year_Revenue'] = df['B001101000'].shift(1)
df['SalesGrowth'] = (df['B001101000'] / df['Last_Year_Revenue'] - 1).fillna(0)
```

2.5 数据描述性统计

```
df.columns

Index(['Stkcd', 'ShortName', 'Indcd', 'Indnme', 'F100801A', 'F100901A',
       'F100902A', 'F100903A', 'F100904A', 'year', 'manufacturingindustry',
       'InstitutionID', 'Status', 'HolderNumber', 'Shares', 'Proportion',
       'S0701a', 'S0702a', 'S0701b', 'S0703a', 'S0704a', 'S0705a', 'S0702b',
       'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNature',
       'accper', 'A001212000', 'A001218000', 'A001000000', 'A002000000',
       'B001101000', 'B002000000', 'LargestHolderRate', 'TopTenHoldersRate',
       'EstablishDate', 'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1',
       'OwnCon2_10', 'SOE', 'ROA', 'Establish_year', 'Age',
       'Last_Year_Revenue', 'SalesGrowth'],
      dtype='object')
```

2.5.1 全描述性统计

```
#最终样本量
df.shape

(327567, 50)

df.describe()
```

	Stkcd	F100801A	F100901A	F100902A	F100903A	F100904A	year	manufacturing	Institution ID	Status	...	PPE_TA	Intass	OwnCon1	OwnCon2_10	SOE	ROA	Age	Establish_year	Last_Year_Revenue	Sales Growth
count	32756	32756	32756	32756	32756	32756	32756	3275670	32756	32756	...	32756	32756	32756	32756	32756	32756	32756	32756	32756	32756
	7.00000	7.00000e+05	7.00000	7.00000	7.00000	7.00000	7.00000		7.00000e+05	7.00000		7.00000	7.00000	7.00000	7.00000	7.00000	7.00000	7.00000	7.00000	7.00000e+05	7.00000
mean	3136	2.17	2.06	2.25	2.66	2.90	2015.95	0.510747	2.84	1.59	...	0.21	0.04	34.79	25.22	0.33	0.04	19	17.15	1.10	0.10
	7835	4972e+10	1301	4699	2088	9907	6695		4354e+06	7511		7483	6089	9067	2727	8267	4226	9802	6893	0030e+10	3264
std	2746	8.79	2.25	2.46	2.88	3.15	5.09	0.499885	4.50	1.22	...	0.16	0.06	15.27	13.57	0.47	0.09	5.55	6.34	7.85	17.12
	5973	5379	0992	5757	3675	2868	0264		4082	3992		2245	1325	9419	6366	3121	3831	4898	4083	8401e	4252

	Stkcd	F100801A	F100901A	F100902A	F100903A	F100904A	year	manufacturingindustry	InstitutionID	Status	...	PPE_TA	Intass	OwnCon1	OwnCon2_10	SOE	ROA	Establish_year	Age	Last_Year_Revenue	Sales Growth
	3189	e+10							e+06											+10	
min	2.000000	2.005420e+08	0.624542	0.672131	0.512040	0.523741	2003.000000	0.000000	1.000000e+02	0.000000	...	-0.206255	-0.019847	1.840000	0.000000	0.000000	-8.753399	1958.000000	0.000000	-1.148569e+08	-2.733488
25%	2340.000000	3.203148e+09	1.233935	1.313528	1.392143	1.493675	2012.000000	0.000000	1.021670e+05	0.000000	...	-0.093320	0.015757	22.900000	14.400000	0.000000	0.017797	1994.000000	13.000000	8.000052e+08	0.000000
50%	3004.000000	6.043000e+09	1.559600	1.726600	2.020400	2.217300	2017.000000	1.000000	1.050000e+05	2.000000	...	-0.018400	0.031000	32.650000	24.446000	0.000000	0.004200	1999.000000	17.000000	1.882000e+08	0.000000

	Stkcd	F100801A	F100901A	F100902A	F100903A	F100904A	year	manufacturing industry	InstitutionID	State	...	PPE_TA	Intass	OwnCon1	OwnCon2_10	SOE	ROA	Age	Establish_year	Last_Year_Revenue	Sales Growth
	29.000000	2240e+09	411	827	5095	1766	0000	00	67506e+05	000		418	686	00000	00000	0000	926	0000	00000	8667e+09	0000
75%	6006450.000000	1.49513 9e+10	2.285835	2.508344	3.093654	3.377984	2020.000000	1.000000 00	1.010216e+07	3.000000	...	0.306808	0.055047	45.000000	34.940000	1.000000	0.074058	2002.000000	21.000000	5.088103e+09	0.000000
max	873833.000000	2.98990 7e+12	259.145916	259.145916	349.085718	349.085718	2023.000000	1.000000 00	1.060939e+07	3.000000	...	0.970921	0.938255	89.990000	72.900000	1.000000	0.785865	2017.000000	64.000000	3.318168e+12	7088.006858

2.5.2 描述性统计分析可视化

2.5.2.1 两权分离度可视化探究

```
import matplotlib.pyplot as plt

period1 = (2003, 2013)
period2 = (2013, 2023)

mean_separation1 = df[(df['year'] >= period1[0]) & (df['year'] <= perio
d1[1])].groupby('year')['Seperation'].mean()
mean_separation2 = df[(df['year'] >= period2[0]) & (df['year'] <= perio
d2[1])].groupby('year')['Seperation'].mean()

# 设置中文显示
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False

#2003-2013
plt.figure(figsize=(10, 6))
plt.subplot(2, 1, 1)
plt.plot(mean_separation1.index, mean_separation1.values, marker='o', l
abel='2003-2013')
plt.xlabel('年份')
plt.ylabel('两权分离度年度均值')
plt.title('2003-2013 两权分离度年度均值趋势')
plt.legend()

#2013-2023
plt.subplot(2, 1, 2)
plt.plot(mean_separation2.index, mean_separation2.values, marker='o', l
abel='2013-2023')
plt.xlabel('年份')
plt.ylabel('两权分离度年度均值')
plt.title('2013-2023 年两权分离度年度均值趋势')
plt.legend()


plt.tight_layout()  # 调整子图参数，使之填充整个图像区域
plt.show()
```

```
Text(0.5, 0, '年份')
```

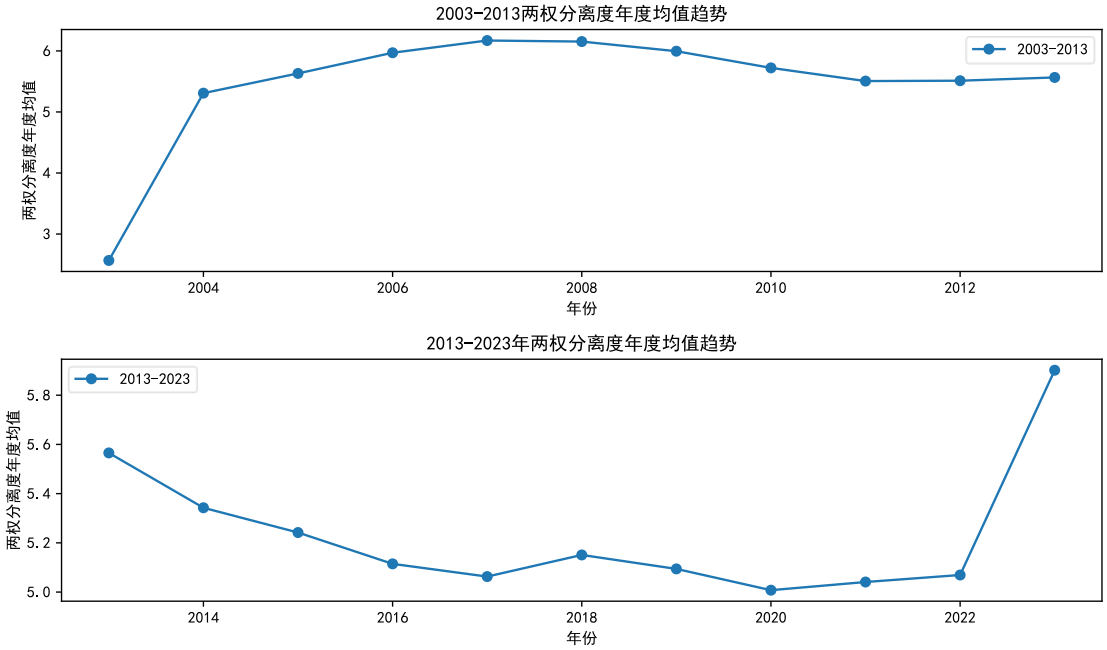
```
Text(0, 0.5, '两权分离度年度均值')
```

```
Text(0.5, 1.0, '2003-2013 两权分离度年度均值趋势')
```

```
Text(0.5, 0, '年份')
```

```
Text(0, 0.5, '两权分离度年度均值')
```

```
Text(0.5, 1.0, '2013-2023 年两权分离度年度均值趋势')
```

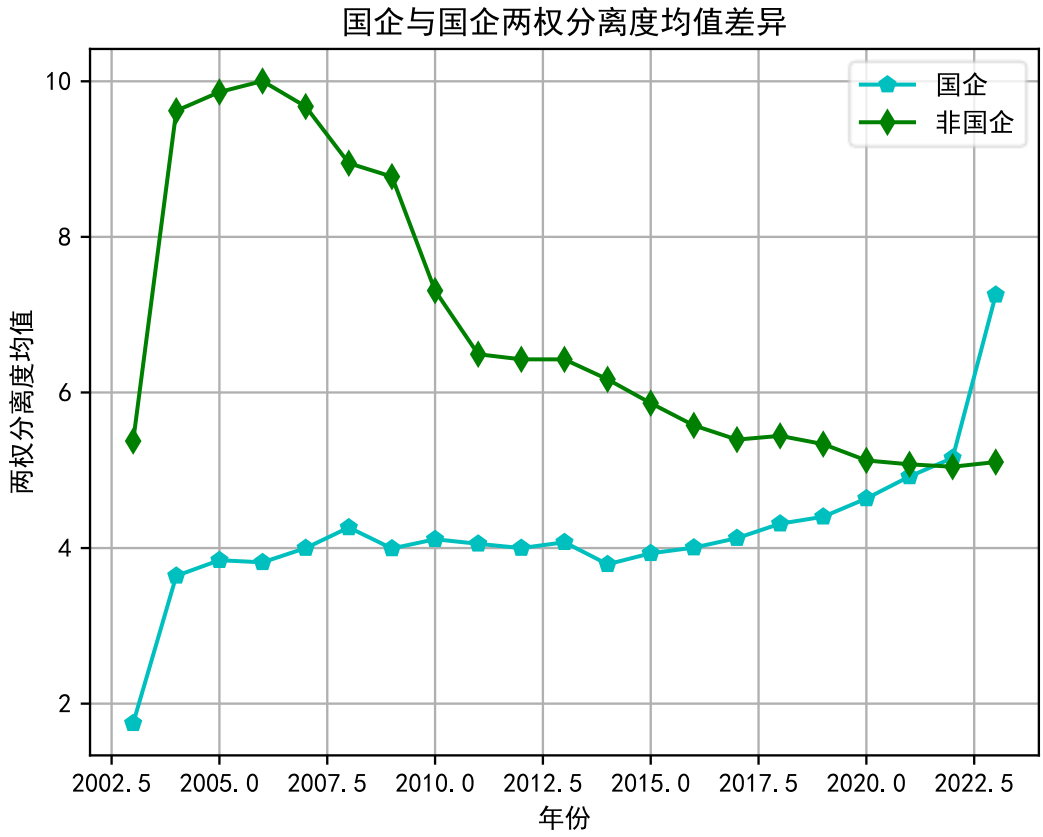



```
import matplotlib.pyplot as plt

means = df.groupby(['year', 'PropertyRightsNature'])['Seperation'].mean
().reset_index()
yes_means = means[means['PropertyRightsNature'] == 1]
no_means = means[means['PropertyRightsNature'] == 0]

# 绘制国企的数据
plt.plot(yes_means['year'], yes_means['Seperation'], label='国企', mark
er='p',color='c')

# 绘制非国企的数据
plt.plot(no_means['year'], no_means['Seperation'], label='非国企', mark
er='d',color='g')

plt.legend()
plt.xlabel('年份')
plt.ylabel('两权分离度均值')
plt.title('国企与国企两权分离度均值差异')
plt.grid(True)

plt.show()
```

```
Text(0.5, 0, '年份')
```

```
Text(0, 0.5, '两权分离度均值')
```

```
Text(0.5, 1.0, '国企与国企两权分离度均值差异')
```



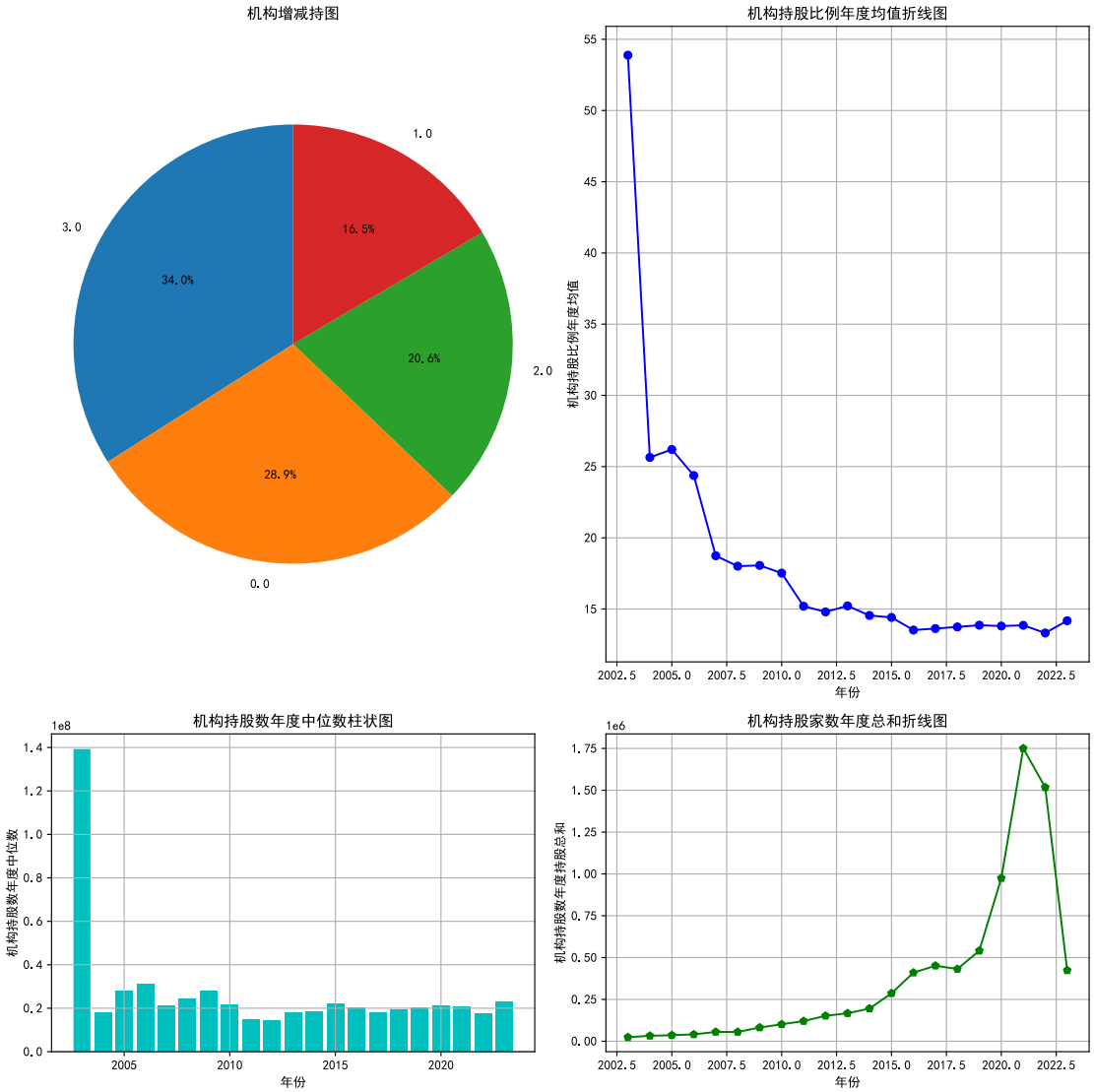
2.5.2.2 机构持股信息可视化

```
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import gridspec

#作图数据
status_counts = df['Status'].value_counts()
labels = status_counts.index.tolist()
sizes = status_counts.values.tolist()

annual_proportion_mean = df.groupby('year')['Proportion'].mean().reset_
index()

annual_median = df.groupby('year')['Shares'].median().reset_index()

annual_sum=df.groupby('year')['HolderNumber'].sum().reset_index()

fig = plt.figure(figsize=(12,12))
grid = gridspec.GridSpec(2, 2, height_ratios=[2, 1], width_ratios=[1,
1])
```

```
# 绘制机构增减持饼图
# 0：维持；1：增持；2：减持；3：新进
ax1 = fig.add_subplot(grid[0, 0])
ax1.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
ax1.axis('equal')  # 确保饼图是圆的
ax1.set_title('机构增减持图')

# 绘制构持股数年度比例均值折线图
ax2 = fig.add_subplot(grid[0, 1])
ax2.plot(annual_proportion_mean['year'], annual_proportion_mean['Propor
tion'], marker='o', linestyle='-', color='blue')
ax2.set_title('机构持股比例年度均值折线图')
ax2.set_xlabel('年份')
ax2.set_ylabel('机构持股比例年度均值')
ax2.grid(True)

# 绘制机构持股数年度中位数柱状图
ax3 = fig.add_subplot(grid[1, 0])
ax3.bar(annual_median['year'], annual_median['Shares'], color='c')
ax3.set_title('机构持股数年度中位数柱状图')
ax3.set_xlabel('年份')
ax3.set_ylabel('机构持股数年度中位数')
ax3.grid(True)

# 绘制机构持股数家数年度总和折线图
ax4 = fig.add_subplot(grid[1, 1])
ax4.plot(annual_sum['year'], annual_sum['HolderNumber'], marker='p', li
nestyle='-', color='g')
ax4.set_title('机构持股家数年度总和折线图')
ax4.set_xlabel('年份')
ax4.set_ylabel('机构持股数年度持股总和')
ax4.grid(True)

# 调整子图间距
plt.tight_layout()

plt.show()
```

```
([<matplotlib.patches.Wedge at 0x19a3ab3b200>,
  <matplotlib.patches.Wedge at 0x19a32cccc20>,
  <matplotlib.patches.Wedge at 0x19a38e19fa0>,
  <matplotlib.patches.Wedge at 0x19a33067d70>],
 [Text(-0.9640959812767127, 0.5296403863812623, '3.0'),
  Text(-0.1072154587189917, -1.094762460724552, '0.0'),
  Text(1.0928611753575848, -0.12511775012378865, '2.0'),
  Text(0.54534764312417, 0.9552988789582626, '1.0')],
 [Text(-0.5258705352418432, 0.28889475620796123, '34.0%'),
```

```
  Text(-0.058481159301268196, -0.5971431603952102, '28.9%'),
  Text(0.5961060956495916, -0.06824604552206652, '20.6%'),
  Text(0.2974623507950018, 0.5210721157954159, '16.5%')])

(-1.0999998785819671,
 1.0999981517220176,
 -1.0999996551377191,
 1.0999999835779866)

Text(0.5, 1.0, '机构增减持图')

Text(0.5, 1.0, '机构持股比例年度均值折线图')

Text(0.5, 0, '年份')

Text(0, 0.5, '机构持股比例年度均值')

Text(0.5, 1.0, '机构持股数年度中位数柱状图')

Text(0.5, 0, '年份')

Text(0, 0.5, '机构持股数年度中位数')

Text(0.5, 1.0, '机构持股家数年度总和折线图')

Text(0.5, 0, '年份')

Text(0, 0.5, '机构持股数年度持股总和')
```



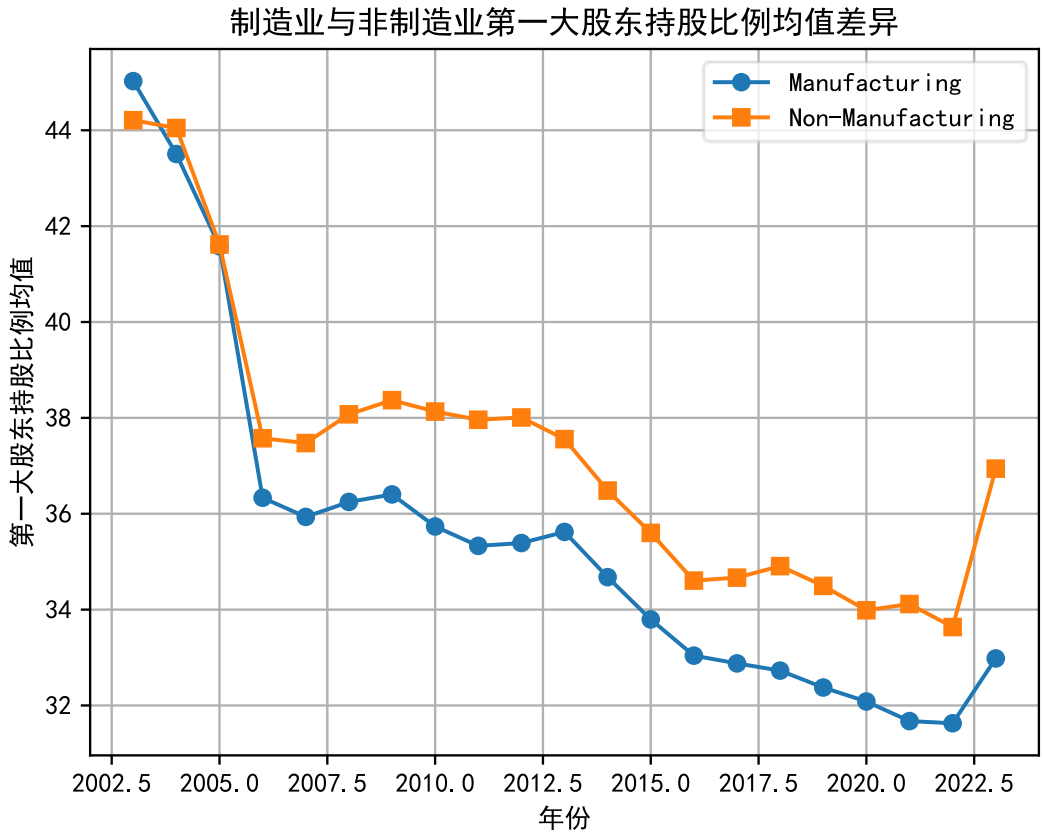
2.5.2.3 分组指标探究

2.5.2.3.1 制造业与非制造业

```
#探寻制造业与非制造业第一大股东持股比例均值差异
import matplotlib.pyplot as plt

means = df.groupby(['year', 'manufacturingindustry'])['Shrcr1'].mean().
reset_index()
yes_means = means[means['manufacturingindustry'] == 1]
no_means = means[means['manufacturingindustry'] == 0]

# 绘制制造业的数据
plt.plot(yes_means['year'], yes_means['Shrcr1'], label='Manufacturing',
 marker='o')

# 绘制非制造业的数据
```

```
plt.plot(no_means['year'], no_means['Shrcr1'], label='Non-Manufacturing
', marker='s')

plt.legend()
plt.xlabel('年份')
plt.ylabel('第一大股东持股比例均值')
plt.title('制造业与非制造业第一大股东持股比例均值差异')
plt.grid(True)

plt.show()
```

```
Text(0.5, 0, '年份')

Text(0, 0.5, '第一大股东持股比例均值')

Text(0.5, 1.0, '制造业与非制造业第一大股东持股比例均值差异')
```



2.5.2.3.2 国企与非国企

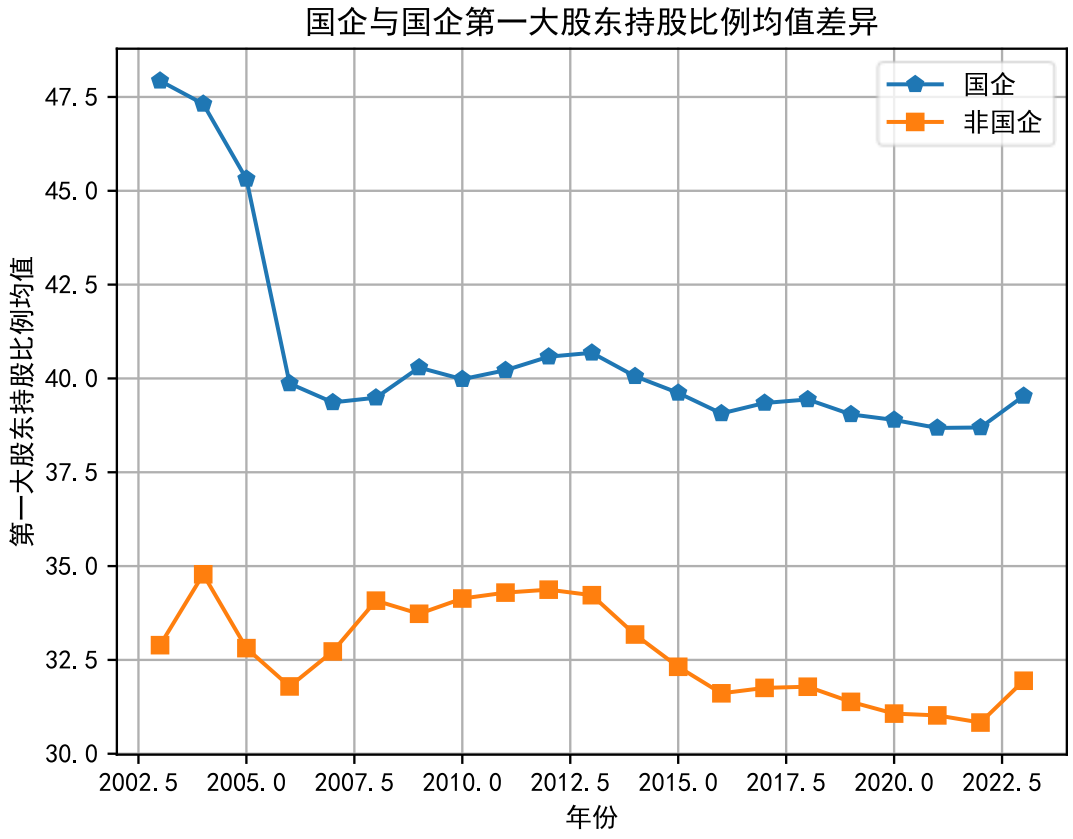
```
#探寻国企与非国企第一大股东持股比例均值差异
import matplotlib.pyplot as plt

means = df.groupby(['year', 'PropertyRightsNature'])['Shrcr1'].mean().r
eset_index()
yes_means = means[means['PropertyRightsNature'] == 1]
```

```
no_means = means[means['PropertyRightsNature'] == 0]


plt.plot(yes_means['year'], yes_means['Shrcr1'], label='国企', marker='
p')
plt.plot(no_means['year'], no_means['Shrcr1'], label='非国企', marker='
s')


plt.legend()
plt.xlabel('年份')
plt.ylabel('第一大股东持股比例均值')
plt.title('国企与国企第一大股东持股比例均值差异')
plt.grid(True)
plt.show()
```

```
Text(0.5, 0, '年份')
```

```
Text(0, 0.5, '第一大股东持股比例均值')
```

```
Text(0.5, 1.0, '国企与国企第一大股东持股比例均值差异')
```



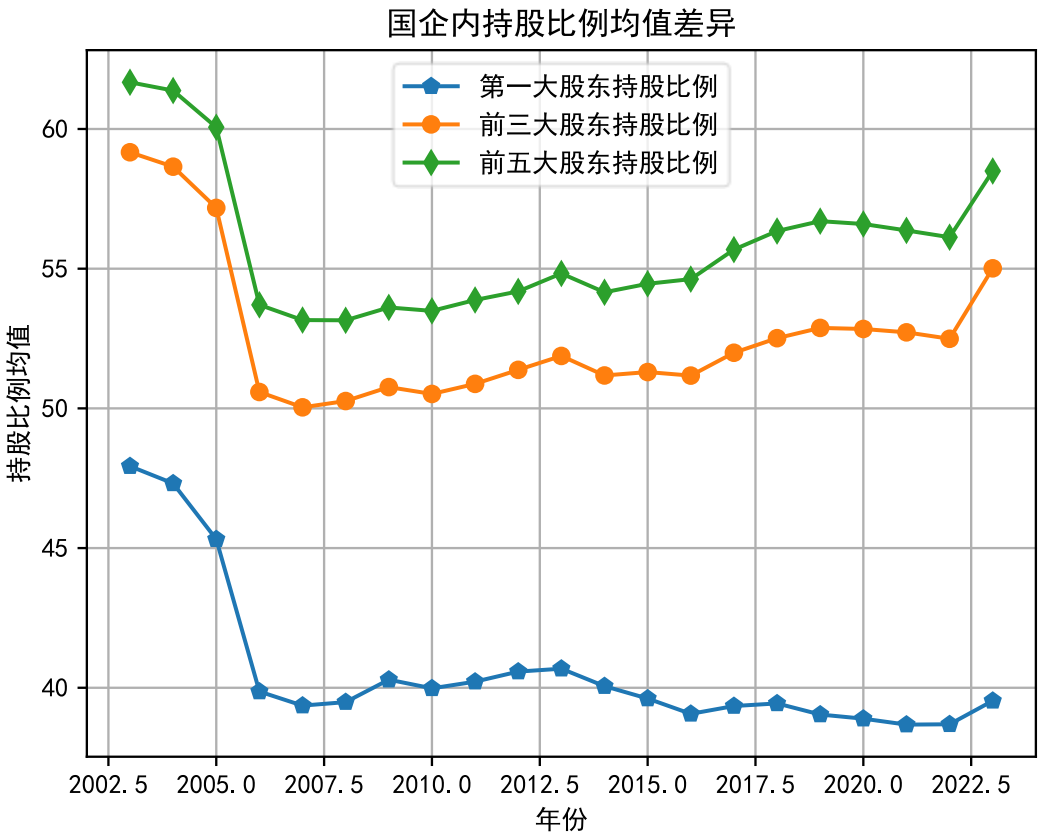

```
#探寻国企内持股比例均值差异
import matplotlib.pyplot as plt
```

```
#计算第一大股东比例均值
means1 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr1'].mean().
reset_index()
yes_means1 = means1[means['PropertyRightsNature'] == 1]
#计算前三大股东比例均值
means2 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr2'].mean().
reset_index()
yes_means2 = means2[means['PropertyRightsNature'] == 1]
#计算前五大股东比例均值
means3 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr3'].mean().
reset_index()
yes_means3 = means3[means['PropertyRightsNature'] == 1]


plt.plot(yes_means1['year'], yes_means1['Shrcr1'], label='第一大股东持股
比例', marker='p')
plt.plot(yes_means2['year'], yes_means2['Shrcr2'], label='前三大股东持股
比例', marker='o')
plt.plot(yes_means3['year'], yes_means3['Shrcr3'], label='前五大股东持股
比例', marker='d')


plt.legend()
plt.xlabel('年份')
plt.ylabel('持股比例均值')
plt.title('国企内持股比例均值差异')
plt.grid(True)
plt.show()
```

```
Text(0.5, 0, '年份')
```

```
Text(0, 0.5, '持股比例均值')
```

```
Text(0.5, 1.0, '国企内持股比例均值差异')
```



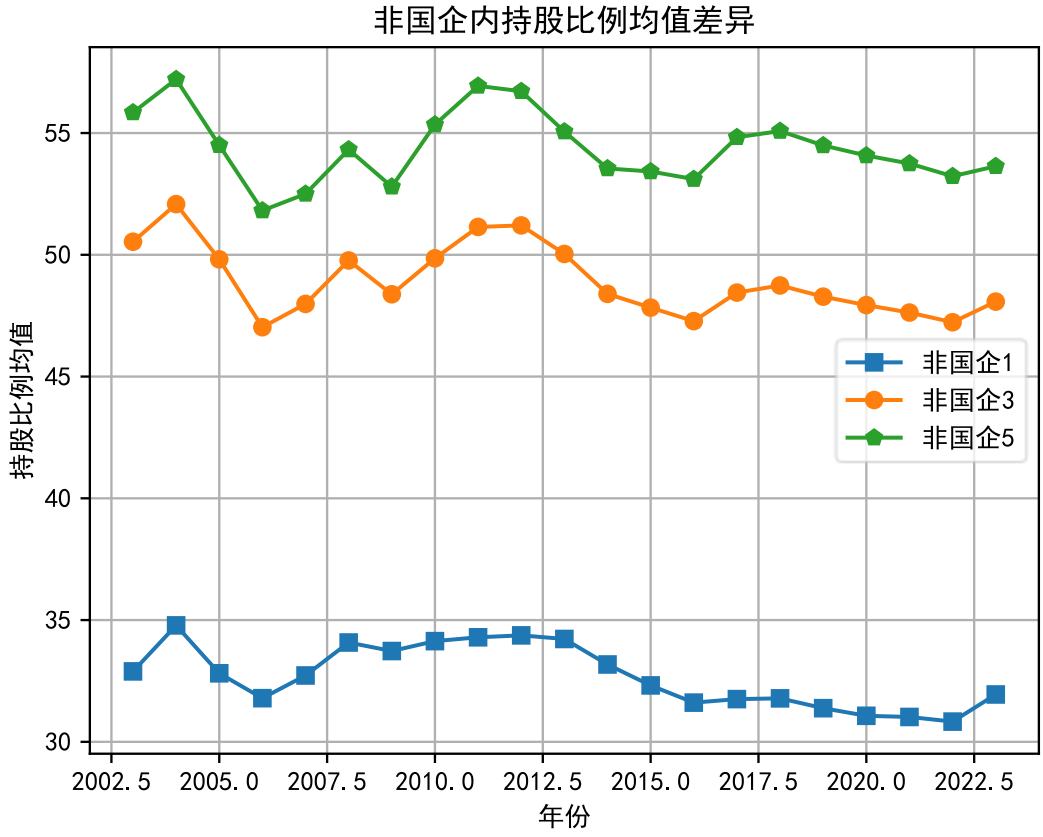

```
#探寻非国企内持股比例均值差异
import matplotlib.pyplot as plt

#计算第一大股东比例均值
means1 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr1'].mean().
reset_index()
no_means1 = means1[means['PropertyRightsNature'] == 0]
#计算前三大股东比例均值
means2 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr2'].mean().
reset_index()
no_means2 = means2[means['PropertyRightsNature'] == 0]
#计算前五大股东比例均值
means3 = df.groupby(['year', 'PropertyRightsNature'])['Shrcr3'].mean().
reset_index()
no_means3 = means3[means['PropertyRightsNature'] == 0]


plt.plot(no_means1['year'], no_means1['Shrcr1'], label='非国企1', marke
r='s')
plt.plot(no_means2['year'], no_means2['Shrcr2'], label='非国企3', marke
r='o')
plt.plot(no_means3['year'], no_means3['Shrcr3'], label='非国企5', marke
r='p')
```

```
plt.legend()
plt.xlabel('年份')
plt.ylabel('持股比例均值')
plt.title('非国企内持股比例均值差异')
plt.grid(True)
plt.show()
```

```
Text(0.5, 0, '年份')
```

```
Text(0, 0.5, '持股比例均值')
```

```
Text(0.5, 1.0, '非国企内持股比例均值差异')
```



3. 模型训练与评估

3.1 算法基本介绍

3.1.1 模型一、梯度提升树 GBDT 模型

梯度提升树（Gradient Boosting Decision Tree, GBDT）是一种集成学习方法，通过逐步构建并结合多个弱学习器（通常是决策树）来构建一个强大的预测模型。

核心思想是通过迭代的方式逐步减少残差（即预测值与真实值之间的差距）。在每一轮迭代中，模型会聚焦于当前模型预测结果与真实值之间残差的最大部分，然后训练一个新的决策树来拟合这部分残差。新树的输出被加到现有的模型预测上，从而逐步改善整体模型的预测能力。

在训练每棵决策树时，GBDT 采用了梯度下降的思想来最小化损失函数。具体来说，模型通过计算损失函数关于当前预测值的负梯度，确定残差的方向和大小，然后以该梯度作为指导构建决策树。这使得新树能够针对性地拟合残差，有效地减少整体损失。

3.1.2 模型二、Lasso 回归

Lasso 回归（Least Absolute Shrinkage and Selection Operator），也称为 L1 正则化线性回归，是一种数据挖掘方法，用于在常用的多元线性回归中添加惩罚函数，以压缩系数，从而达到精简模型的目的。

核心原理是在传统的线性回归损失函数中加入了一个 L1 正则化项（即参数的绝对值之和），通过正则化项对模型的参数进行惩罚，使得参数趋向于较小的值，同时也有助于产生稀疏模型，即某些特征的权重为零。

优势：

参数收缩与特征选择：通过 L1 正则化项，Lasso 回归可以将某些回归系数精确地压缩到 0，从而实现特征选择的目的。这使得模型更为简洁，减少了模型的复杂度。

防止过拟合：在拟合过于复杂的模型时，Lasso 回归通过正则化项对系数进行惩罚，有助于防止过拟合现象。

3.2 模型训练

```
df.columns
```

```
Index(['Stkcd', 'ShortName', 'Indcd', 'Indnme', 'F100801A', 'F100901A',
       'F100902A', 'F100903A', 'F100904A', 'year', 'manufacturingindust
ry',
       'InstitutionID', 'Status', 'HolderNumber', 'Shares', 'Proportion
',
       'S0701a', 'S0702a', 'S0701b', 'S0703a', 'S0704a', 'S0705a', 'S07
02b',
```

```
       'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNatur
e',
       'accper', 'A001212000', 'A001218000', 'A001000000', 'A002000000',
       'B001101000', 'B002000000', 'LargestHolderRate', 'TopTenHoldersR
ate',
       'EstablishDate', 'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1
',
       'OwnCon2_10', 'SOE', 'ROA', 'Establish_year', 'Age',
       'Last_Year_Revenue', 'SalesGrowth'],
      dtype='object')
```

3.2.1 模型一

```
import pandas as pd
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.model_selection import train_test_split


X_value= ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S0704a', '
S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNat
ure',
          'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1', 'OwnCon2_10
', 'SOE', 'ROA', 'Establish_year', 'Age',  'SalesGrowth']
X1 = df[X_value]

y1 = df['F100901A']

X_train, X_test, y_train, y_test = train_test_split(
    X1, y1, test_size=0.2, random_state=42
)


gbreg = GradientBoostingRegressor(max_depth=2, n_estimators=3, learning
_rate=0.15,random_state=0)


gbreg.fit(X_train, y_train)


gbreg.predict(X_test[:5])


f"GradientBoostingRegressor 在训练集上的 R2:{gbreg.score(X_train, y_trai
n):.3f}"
f"GradientBoostingRegressor 在测试集上的 R2:{gbreg.score(X_test, y_test):.
3f}"
```

```
GradientBoostingRegressor(learning_rate=0.15, max_depth=2, n_estimators
=3,
                          random_state=0)
```

```
array([2.0508443, 2.0508443, 2.0508443, 2.0508443, 2.0508443])

'GradientBoostingRegressor 在训练集上的 R2:0.327'

'GradientBoostingRegressor 在测试集上的 R2:0.221'
```

3.2.2 模型二

```
from sklearn.linear_model import Lasso

X_value= ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S0704a', '
S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNat
ure',
          'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1', 'OwnCon2_10
', 'SOE', 'ROA', 'Establish_year', 'Age',  'SalesGrowth']
X2 = df[X_value]

y2 = df['F100901A']
X_train, X_test, y_train, y_test = train_test_split(X2, y2,test_size=0.
2,random_state=42)


lasso = Lasso(alpha=0.95)


lasso .fit(X_train, y_train)


lasso .predict(X_test[:5])


lasso.score(X_train, y_train)
lasso.score(X_test, y_test)

lasso = Lasso(alpha=0.95).fit(X_train, y_train)
f"Training set score: {lasso.score(X_train, y_train):.3f}"
f"Test set score: {lasso.score(X_test, y_test):.3f}"

Lasso(alpha=0.95)

array([1.9151229 , 2.15419853, 1.81000124, 2.23009166, 2.10087588])

0.008929994856189283

0.011396955874669712

'Training set score: 0.009'

'Test set score: 0.011'
```

3.3 模型评估

3.3.1 模型一

```
from sklearn.metrics import (
    median_absolute_error,
    mean_absolute_error,
    mean_squared_error,
    r2_score,
)
from sklearn.model_selection import train_test_split
from sklearn.model_selection import RandomizedSearchCV
from IPython.core.interactiveshell import InteractiveShell

def scoring_reg(y_true, y_pred):
    print("-" * 10, "\n")
    print("median_absolute_error:", median_absolute_error(y_true, y_pre
d))
    print("mean_absolute_error:", mean_absolute_error(y_true, y_pred))
    print("mean_squared_error:", mean_squared_error(y_true, y_pred))
    print("r2_score:", r2_score(y_true, y_pred))
```

3.3.2 模型二

```
def myscoring(y_true, y_pred):
    print("-" * 10, "\n")
    print("median_absolute_error:", median_absolute_error(y_true, y_pre
d))
    print("mean_absolute_error:", mean_absolute_error(y_true, y_pred))
    print("mean_squared_error:", mean_squared_error(y_true, y_pred))
    print("r2_score:", r2_score(y_true, y_pred))
```

4. 模型调参

4.0.1 模型一

```
from scipy.stats import uniform
from scipy.stats import randint


params = {
    "max_depth": [3, 4, 5, 6,7,8],
    "min_samples_leaf": uniform(0.0001, 0.3),
    "learning_rate": uniform(0.0001, 0.5),
    "n_estimators": randint(10, 100)
}

gbdt_select = ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S0704
a', 'S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRigh
tsNature',
                   'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1', 'Ow
```

```
nCon2_10', 'SOE', 'ROA', 'Establish_year', 'Age',  'SalesGrowth']
X1 = df[gbdt_select]

y1 = df['F100901A']

X_train, X_test, y_train, y_test = train_test_split(
    X1, y1, test_size=0.2, random_state=42
)

from sklearn.ensemble import GradientBoostingRegressor

rvestimator = GradientBoostingRegressor(random_state=0)
grid = grid = RandomizedSearchCV(
    estimator=rvestimator,
    param_distributions=params,
    n_iter=10,
    scoring="r2",
    n_jobs=-1,
    cv=5,
    random_state=0
)
grid.fit(X_train, y_train)
```

```
RandomizedSearchCV(cv=5, estimator=GradientBoostingRegressor(random_sta
te=0),
                   n_jobs=-1,
                   param_distributions={'learning_rate': <scipy.stats._
distn_infrastructure.rv_continuous_frozen object at 0x0000019A345B24E0>,
                                        'max_depth': [3, 4, 5, 6, 7, 8],
                                        'min_samples_leaf': <scipy.stat
s._distn_infrastructure.rv_continuous_frozen object at 0x0000019A32FF44
A0>,
                                        'n_estimators': <scipy.stats._d
istn_infrastructure.rv_discrete_frozen object at 0x0000019A34D98170>},
                   random_state=0, scoring='r2')
```

```
print("Best parameters: {}".format(grid.best_params_))
print("Best cross-validation score: {:.3f}".format(grid.best_score_))
print("Best estimator:\n{}".format(grid.best_estimator_))
```

```
Best parameters: {'learning_rate': 0.36041632736295837, 'max_depth': 4,
 'min_samples_leaf': 0.04310598622271392, 'n_estimators': 42}
Best cross-validation score: 0.208
Best estimator:
GradientBoostingRegressor(learning_rate=0.36041632736295837, max_depth=
4,
                          min_samples_leaf=0.04310598622271392, n_estim
ators=42,
                          random_state=0)
```

4.0.2 模型二

```
param_lasso = {
    'alpha': [1e-4, 1e-3, 1e-2, 1e-1, 1, 10, 100],  # 正则化强度
    'fit_intercept': [True, False]  # 是否计算截距
}

lasso_select = ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S070
4a', 'S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRig
htsNature',
                  'lnAssets', 'Lev', 'PPE_TA', 'Intass', 'OwnCon1', 'Ow
nCon2_10', 'SOE', 'ROA', 'Establish_year', 'Age',  'SalesGrowth']
X2 = df[lasso_select]

y2 = df['F100901A']

X_train, X_test, y_train, y_test = train_test_split(
    X2, y2, test_size=0.2, random_state=42
)

from sklearn.linear_model import Lasso
lasso = Lasso()

la=la= RandomizedSearchCV(
    estimator=lasso,
    param_distributions=param_lasso,
    n_iter=10,
    scoring='neg_mean_squared_error',
    cv=5,
    random_state=42,
    n_jobs=-1
)

la.fit(X_train, y_train)

RandomizedSearchCV(cv=5, estimator=Lasso(), n_jobs=-1,
                   param_distributions={'alpha': [0.0001, 0.001, 0.01,
0.1, 1,
                                                  10, 100],
                                        'fit_intercept': [True, False]},
                   random_state=42, scoring='neg_mean_squared_error')

print("Best parameters: {}".format(la.best_params_))
print("Best cross-validation score: {:.3f}".format(la.best_score_))
print("Best estimator:\n{}".format(la.best_estimator_))

Best parameters: {'fit_intercept': True, 'alpha': 0.0001}
Best cross-validation score: -4.965
Best estimator:
Lasso(alpha=0.0001)
```

5. 模型再训练与评估

5.0.1 模型一

```
gbreg = GradientBoostingRegressor(
    learning_rate=grid.best_params_["learning_rate"],
    max_depth=grid.best_params_["max_depth"],
    min_samples_leaf=grid.best_params_["min_samples_leaf"],
    n_estimators=grid.best_params_["n_estimators"],
    random_state=0
)

gbreg.fit(X_train, y_train)

y_train_pred = gbreg.predict(X_train)
y_test_pred = gbreg.predict(X_test)

scoring_reg(y_train, y_train_pred)
scoring_reg(y_test, y_test_pred)
```

```
GradientBoostingRegressor(learning_rate=0.36041632736295837, max_depth=
4,
                          min_samples_leaf=0.04310598622271392, n_estim
ators=42,
                          random_state=0)

----------

median_absolute_error: 0.43055519778276263
mean_absolute_error: 0.7104879538160958
mean_squared_error: 4.208617129023834
r2_score: 0.2097799142268003
----------

median_absolute_error: 0.4322560874143442
mean_absolute_error: 0.7097417459687295
mean_squared_error: 2.996913691821297
r2_score: 0.2565593402969356
```

5.0.2 模型二

```
lareg= Lasso(
    alpha=la.best_params_["alpha"],
    fit_intercept=la.best_params_["fit_intercept"],
    random_state=42
)

lareg.fit(X_train, y_train)

y_train_pred = lareg.predict(X_train)
y_test_pred = lareg.predict(X_test)
```

```
myscoring(y_train, y_train_pred)
myscoring(y_test, y_test_pred)

Lasso(alpha=0.0001, random_state=42)

----------

median_absolute_error: 0.6243878081397505
mean_absolute_error: 0.8640897125574449
mean_squared_error: 4.963253847374214
r2_score: 0.06808750695364008
----------

median_absolute_error: 0.6260719397293559
mean_absolute_error: 0.8583693732472447
mean_squared_error: 3.692201588441123
r2_score: 0.08408013478719556
```

6. 模型解释与应用

6.1 股权特征重要性

6.1.1 模型一

```
from sklearn.ensemble import GradientBoostingRegressor
import pandas as pd

X_value = ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S0704a',
'S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNa
ture']
X1 = df[X_value]
y1 = df['F100901A']


gbdt_model = GradientBoostingRegressor()
gbdt_model.fit(X1, y1)


importance_gbdt = gbdt_model.feature_importances_

import matplotlib.pyplot as plt

indices = np.argsort(importance_gbdt)[::-1]  # 索引从大到小排序
names = [X_value[i] for i in indices]  # 获取对应的特征名称

plt.figure(figsize=(10, 5))
plt.bar(range(10), importance_gbdt[indices[:10]])
```

```
plt.title('GBDT 特征重要性(前十)')
plt.xlabel('特征')
plt.ylabel('重要性')
plt.xticks(range(10), [X_value[i] for i in indices[:10]], rotation='ver
tical')
plt.tight_layout()
plt.show()
```

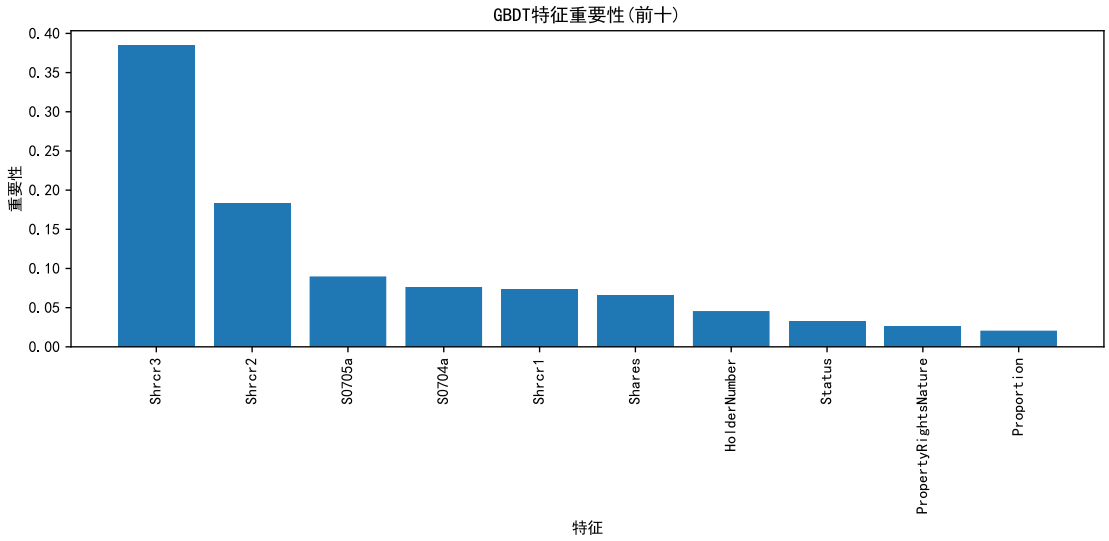
```
GradientBoostingRegressor()
```

```
Text(0.5, 1.0, 'GBDT 特征重要性(前十)')
```

```
Text(0.5, 0, '特征')
```

```
Text(0, 0.5, '重要性')
```

```
([<matplotlib.axis.XTick at 0x19a330142c0>,
  <matplotlib.axis.XTick at 0x19a34faf350>,
  <matplotlib.axis.XTick at 0x19a34f25df0>,
  <matplotlib.axis.XTick at 0x19a34f1bb60>,
  <matplotlib.axis.XTick at 0x19a34ef7d70>,
  <matplotlib.axis.XTick at 0x19a38447920>,
  <matplotlib.axis.XTick at 0x19a38446ff0>,
  <matplotlib.axis.XTick at 0x19a38447680>,
  <matplotlib.axis.XTick at 0x19a384466f0>,
  <matplotlib.axis.XTick at 0x19a38445d30>],
 [Text(0, 0, 'Shrcr3'),
  Text(1, 0, 'Shrcr2'),
  Text(2, 0, 'S0705a'),
  Text(3, 0, 'S0704a'),
  Text(4, 0, 'Shrcr1'),
  Text(5, 0, 'Shares'),
  Text(6, 0, 'HolderNumber'),
  Text(7, 0, 'Status'),
  Text(8, 0, 'PropertyRightsNature'),
  Text(9, 0, 'Proportion')])
```



6.1.2 模型二

```
from sklearn.linear_model import Lasso
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np


X_value = ['Status', 'HolderNumber', 'Shares', 'Proportion', 'S0704a',
'S0705a', 'Seperation', 'Shrcr1', 'Shrcr2', 'Shrcr3', 'PropertyRightsNa
ture']
X2 = df[X_value]
y2 = df['F100901A']


lasso_model = Lasso(alpha=0.95)
lasso_model.fit(X2, y2)

# 获取特征重要性（系数）
coef = lasso_model.coef_


print('前十特征重要性:')
for f in range(min(10, len(coef))):
    print(f'{X_value[f]}: {coef[f]}')
```

```
Lasso(alpha=0.95)
```

```
前十特征重要性:
Status: -0.0
HolderNumber: 0.0004987237506142459
Shares: -6.74619606217727e-12
Proportion: -0.0
```

```
S0704a: -0.00022074640644622007
S0705a: -1.5185300711834792e-11
Seperation: -0.0
Shrcr1: -0.004225190556922066
Shrcr2: -0.004841190641558022
Shrcr3: -0.0
```

7. 研究结论

7.1 股权特征典型事实

7.1.1 两权分离度

（1）总体上，在 2003-2013 时间段呈现稳健上升特征，但在 2014-2022 时间段则是逐渐减少，2022 年后又再次上升。

（2）国企与非国企之间，两者则是相反的差异，国企总体在提高两权分离度，而非国企总体在降低两权分离度。

7.1.2 机构投资者

（1）在机构增减股份方面，选择维持与新进比例最多，而选择增持方面，占比最低。

（2）每年度的机构持股数中位数分布上，除一开始的极端情况，总体分布较为均匀。

（3）机构持股比例年度均值上呈现逐步下降的趋势，而机构持股数年度持股总和上先逐步上升，2018-2021 前后，持股总和快速上升，但与此同时，也以 2021 前后为转折点，持股总和又呈现出急剧下降的特点。

7.1.3 股权集中度

（1）国企与非国企

在国企与非国企第一大股东持股比例均值上，国企均值明显大于非国企。趋势上，国企大体呈现下降趋势，最后在 2022 后半年上升，非国企则是上升，下降，上升。

在第一，前三，前五持股比例之和均值的变化趋势:国企方面，三者基本呈现出先快速下降，但在除了第一大股东在 2006-2022 区间上是缓和下降的趋势，其余两者都是逐步上升。在 20022 后半年，出现了较为快速的上升；非国企方面，前三，前五持股比例之和的均值波动比第一大股东更为明显，第一大股东均值变化较为缓和。

（2）制造业与非制造业

在制造业与非制造业第一大股东持股比例均值上，2006 左右，两者从几乎没有差异发展到差异逐步拉开，但与此同时，两者总体都呈现出先下降，后在 2022 年左右又上升的趋势。

7.2 股权特征重要性

在 GBDT 中，Shrcr2 - 公司前 3 位大股东持股比例之和，与 Shrcr3 - 公司前 5 位大股东持股比例之和显著高于其他股权特征

报告承诺

本项目报告参考了：

[1]中国企业的专利行为:特征事实以及来自创新政策的影响（寇宗来 刘学悦）

[2]大数据应用对中国企业市场价值的影响———来自中国上市公司年报文本分析的证据（张叶青 陆 瑶 李乐芸）

[3]数字技术创新对企业市场价值的影响研究（陶 锋 朱 盼 邱楚芝 王欣然）