基于沪深京市场A股上市公司2018年至2024年的年度财务数据，本文尝试对企业的成本与费用结构进行建模分析。

模型部分使用 scikit-learn 中的线性回归和梯度提升决策树（GBDT）进行对比实验，旨在比较线性假设与非线性方法在拟合企业成本行为上的表现差异，初步评估不同模型在解释费用结构变化中的适用性。

环境设置如下：

In [108…

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

```
Python 3.13.2
matplotlib                3.10.3
matplotlib-inline         0.1.7
pandas                    2.3.0
scikit-learn              1.7.0
```

In [109…

```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

目 录

In [110…

```
import numpy as np
import pandas as pd
```

1. 数据合并

数据导入与合并

In [111…

```
df_18_19=pd.read_excel("rawdata\智能查询_沪深京股票(18-19年频).xlsx",skiprows=[1
df_20_24=pd.read_excel("rawdata\智能查询_沪深京股票(20-24年频).xlsx",skiprows=[1

df_18_19.head()
df_20_24.head()
```

```
c:\Users\Lenovo\AppData\Local\Programs\Python\Python313\Lib\site-packages\openpyx
l\styles\stylesheet.py:237: UserWarning: Workbook contains no default style, appl
y openpyxl's default
  warn("Workbook contains no default style, apply openpyxl's default")
c:\Users\Lenovo\AppData\Local\Programs\Python\Python313\Lib\site-packages\openpyx
l\styles\stylesheet.py:237: UserWarning: Workbook contains no default style, appl
y openpyxl's default
  warn("Workbook contains no default style, apply openpyxl's default")
```

Out[111…

	code	stknme	listingDate	EndDate	FS_Comins-B001209000	FS_Combas-A001000000	FS_Comins-B001210000
0	59	华锦股份	1997-01-30	2018	2.525935e+08	3.246068e+10	1.642411e+09
1	301	东方盛虹	2000-05-29	2018	1.740349e+08	2.186924e+10	1.394549e+08
2	408	藏格矿业	1996-06-28	2018	4.661615e+08	9.385461e+09	6.286074e+07
3	420	吉林化纤	1996-08-02	2018	5.762423e+07	7.140258e+09	8.851527e+07
4	422	湖北宜化	1996-08-15	2018	5.067766e+08	2.392735e+10	1.088275e+09

Out[111…

	code	stknme	listingDate	EndDate	FS_Comins-B001209000	FS_Combas-A001000000	FS_Comins-B001210000
0	59	华锦股份	1997-01-30	2020	3.264887e+08	2.789965e+10	1.314843e+09
1	301	东方盛虹	2000-05-29	2020	4.737093e+07	6.293361e+10	2.594956e+08
2	408	藏格矿业	1996-06-28	2020	3.808878e+07	8.677639e+09	6.838218e+07
3	420	吉林化纤	1996-08-02	2020	3.621608e+07	8.688283e+09	9.938830e+07
4	422	湖北宜化	1996-08-15	2020	4.618002e+07	2.201567e+10	4.313263e+08

In [112…

```
df_18_19 = df_18_19.rename(columns={"EndDate": "year"})
df_18_19.columns

df_20_24 = df_20_24.rename(columns={"EndDate": "year"})
df_20_24.columns
```

Out[112…

```
Index(['code', 'stknme', 'listingDate', 'year', 'FS_Comins-B001209000',
       'FS_Combas-A001000000', 'FS_Comins-B001210000', 'FS_Comins-B001216000',
       'FS_Comins-B001201000', 'FS_Comins-B001101000',
       'FS_Comscfd-C001000000'],
      dtype='object')
```

Out[112…

```
Index(['code', 'stknme', 'listingDate', 'year', 'FS_Comins-B001209000',
       'FS_Combas-A001000000', 'FS_Comins-B001210000', 'FS_Comins-B001216000',
       'FS_Comins-B001201000', 'FS_Comins-B001101000',
       'FS_Comscfd-C001000000'],
      dtype='object')
```

In [113…

```
df_data = pd.concat([df_18_19, df_20_24], ignore_index=True)
df_data = df_data.sort_values(by=["code", "year"]).reset_index(drop=True)
df_data.head()
```

Out[113…

	code	stknme	listingDate	year	FS_Comins-B001209000	FS_Combas-A001000000	FS_Comins-B001210000	
0	59	华锦股份	1997-01-30	2018	2.525935e+08	3.246068e+10	1.642411e+09	8.
1	59	华锦股份	1997-01-30	2019	3.000648e+08	2.935920e+10	1.530138e+09	1.
2	59	华锦股份	1997-01-30	2020	3.264887e+08	2.789965e+10	1.314843e+09	1.
3	59	华锦股份	1997-01-30	2021	3.165466e+08	3.211587e+10	1.654941e+09	1.
4	59	华锦股份	1997-01-30	2022	3.402182e+08	3.263326e+10	1.117396e+09	1.

In [114…

```
df_data.shape
```

Out[114…

```
(3969, 11)
```

缺失值处理

In [115…

```
df_data.isnull().sum()
```

Out[115…

```
code                      0
stknme                    0
listingDate               0
year                      0
FS_Comins-B001209000    708
FS_Combas-A001000000    706
FS_Comins-B001210000    706
FS_Comins-B001216000    748
FS_Comins-B001201000    708
FS_Comins-B001101000    708
FS_Comscfd-C001000000   706
dtype: int64
```

In [116…

```
# 剔除资产总计缺失的样本
df_data=df_data.dropna(axis=0,subset="FS_Combas-A001000000")
```

In [117…

```
df_data.isnull().sum()
```

Out[117…

```
code                     0
stknme                   0
listingDate              0
year                     0
FS_Comins-B001209000     2
FS_Combas-A001000000     0
FS_Comins-B001210000     0
FS_Comins-B001216000    42
FS_Comins-B001201000     2
FS_Comins-B001101000     2
FS_Comscfd-C001000000    0
dtype: int64
```

In [118…

```
# 研发费用缺失值用0进行填补
df_data["FS_Comins-B001216000"]=df_data["FS_Comins-B001216000"].fillna(0)

# 剩余缺失之间剔除
df_data=df_data.dropna()

df_data.isnull().sum()
```

Out[118…

```
code                    0
stknme                  0
listingDate             0
year                    0
FS_Comins-B001209000    0
FS_Combas-A001000000    0
FS_Comins-B001210000    0
FS_Comins-B001216000    0
FS_Comins-B001201000    0
FS_Comins-B001101000    0
FS_Comscfd-C001000000   0
dtype: int64
```

生成变量

In [119…

```
df_data.columns
```

Out[119…

```
Index(['code', 'stknme', 'listingDate', 'year', 'FS_Comins-B001209000',
       'FS_Combas-A001000000', 'FS_Comins-B001210000', 'FS_Comins-B001216000',
       'FS_Comins-B001201000', 'FS_Comins-B001101000',
       'FS_Comscfd-C001000000'],
      dtype='object')
```

In [120…

```
df_data["CFO_it"]=df_data["FS_Comscfd-C001000000"]

df_data["PROD_it"]=df_data["FS_Comins-B001201000"]

df_data["DISEXP_it"]=df_data["FS_Comins-B001210000"]+df_data["FS_Comins-B0012160

df_data["REV_it"]=df_data["FS_Comins-B001101000"]
df_data['REV_it-1'] = df_data.groupby('code')['REV_it'].shift(1)
df_data['ΔREV_it'] = df_data.groupby('code')['REV_it'].diff()
df_data['ΔREV_it-1'] = df_data.groupby('code')['ΔREV_it'].shift(1)

df_data["A_it"]=df_data["FS_Combas-A001000000"]
df_data['A_it-1'] = df_data.groupby('code')['A_it'].shift(1)

#y
df_data["CFO_it/A_it-1"]=df_data["CFO_it"]/df_data["A_it-1"]
df_data["PROD_it/A_it-1"]=df_data["PROD_it"]/df_data["A_it-1"]
df_data["DISEXP_it/A_it-1"]=df_data["DISEXP_it"]/df_data["A_it-1"]

#x
df_data["1/A_it-1"]=1/df_data["A_it-1"]
df_data["REV_it/A_it-1"]=df_data["REV_it"]/df_data["A_it-1"]
df_data["ΔREV_it/A_it-1"]=df_data["ΔREV_it"]/df_data["A_it-1"]
df_data["ΔREV_it-1/A_it-1"]=df_data["ΔREV_it-1"]/df_data["A_it-1"]
df_data["REV_it-1/A_it-1"]=df_data["REV_it-1"]/df_data["A_it-1"]
```

导出数据

In [121…

```
df_data.columns
```

Out[121…

```
Index(['code', 'stknme', 'listingDate', 'year', 'FS_Comins-B001209000',
       'FS_Combas-A001000000', 'FS_Comins-B001210000', 'FS_Comins-B001216000',
       'FS_Comins-B001201000', 'FS_Comins-B001101000', 'FS_Comscfd-C001000000',
       'CFO_it', 'PROD_it', 'DISEXP_it', 'REV_it', 'REV_it-1', 'ΔREV_it',
       'ΔREV_it-1', 'A_it', 'A_it-1', 'CFO_it/A_it-1', 'PROD_it/A_it-1',
       'DISEXP_it/A_it-1', '1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1',
       'ΔREV_it-1/A_it-1', 'REV_it-1/A_it-1'],
      dtype='object')
```

In [122…

```
df_data.isnull().sum()
```

Out[122…

```
code                        0
stknme                      0
listingDate                 0
year                        0
FS_Comins-B001209000        0
FS_Combas-A001000000        0
FS_Comins-B001210000        0
FS_Comins-B001216000        0
FS_Comins-B001201000        0
FS_Comins-B001101000        0
FS_Comscfd-C001000000       0
CFO_it                      0
PROD_it                     0
DISEXP_it                   0
REV_it                      0
REV_it-1                  553
ΔREV_it                   553
ΔREV_it-1                1091
A_it                        0
A_it-1                    553
CFO_it/A_it-1             553
PROD_it/A_it-1            553
DISEXP_it/A_it-1          553
1/A_it-1                  553
REV_it/A_it-1             553
ΔREV_it/A_it-1            553
ΔREV_it-1/A_it-1         1091
REV_it-1/A_it-1           553
dtype: int64
```

In [123…

```
df=df_data[['code', 'stknme', 'listingDate', 'year','CFO_it/A_it-1', 'PROD_it/A_
       'DISEXP_it/A_it-1', '1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1',
       'ΔREV_it-1/A_it-1', 'REV_it-1/A_it-1']]
# 方便后续训练模型
df_all = df[(df['year'] >= 2020) & (df['year'] <= 2024)]
df_all.head()
```

Out[123…

	code	stknme	listingDate	year	CFO_it/A_it-1	PROD_it/A_it-1	DISEXP_it/A_it-1	
2	59	华锦股份	1997-01-30	2020	0.095869	0.908668	0.059721	3.
3	59	华锦股份	1997-01-30	2021	0.100530	1.092918	0.074634	3.
4	59	华锦股份	1997-01-30	2022	0.054622	1.275073	0.049885	3.
5	59	华锦股份	1997-01-30	2023	0.029732	1.204440	0.045197	3.
6	59	华锦股份	1997-01-30	2024	0.014414	1.003558	0.057146	3.

In [124…

```
df_all.isnull().sum()
```

Out[124…

```
code                  0
stknme                0
listingDate           0
year                  0
CFO_it/A_it-1       169
PROD_it/A_it-1      169
DISEXP_it/A_it-1    169
1/A_it-1            169
REV_it/A_it-1       169
ΔREV_it/A_it-1      169
ΔREV_it-1/A_it-1    345
REV_it-1/A_it-1     169
dtype: int64
```

In [125…

```
df_cleaned=df_all.dropna()
df_cleaned.head()
df_cleaned.shape
```

Out[125…

	code	stknme	listingDate	year	CFO_it/A_it-1	PROD_it/A_it-1	DISEXP_it/A_it-1	
2	59	华锦股份	1997-01-30	2020	0.095869	0.908668	0.059721	3.
3	59	华锦股份	1997-01-30	2021	0.100530	1.092918	0.074634	3.
4	59	华锦股份	1997-01-30	2022	0.054622	1.275073	0.049885	3.
5	59	华锦股份	1997-01-30	2023	0.029732	1.204440	0.045197	3.
6	59	华锦股份	1997-01-30	2024	0.014414	1.003558	0.057146	3.

Out[125…

```
(2170, 12)
```

In [126…

```
df_missing = df_all[df_all.isnull().any(axis=1)]
df_missing.head()
df_missing.shape
```

Out[126…

	code	stknme	listingDate	year	CFO_it/A_it-1	PROD_it/A_it-1	DISEXP_it/A_it-1
346	1207	联科科技	2021-06-23	2021	NaN	NaN	NaN
347	1207	联科科技	2021-06-23	2022	0.028780	0.854740	0.058648
353	1217	华尔泰	2021-09-29	2021	NaN	NaN	NaN
354	1217	华尔泰	2021-09-29	2022	0.080521	0.757693	0.053792
360	1218	丽臣实业	2021-10-15	2021	NaN	NaN	NaN

Out[126…

```
(345, 12)
```

剔除缺失

In [127…

```
df_cleaned.to_excel("data/df_cleaned.xlsx",index=False)
```

缺失表

In [128…

```
df_missing.to_excel("data/df_missing.xlsx",index=False)
```

2. 期间费用管理模型

操控性支出

In [129…

```
df_cleaned=pd.read_excel("data/df_cleaned.xlsx")
```

sk包--线性模型

In [140…

```
X=df_cleaned[['1/A_it-1', 'REV_it-1/A_it-1']]
y=df_cleaned['DISEXP_it/A_it-1']

# 训练集：2020 - 2023
X_train = X[df_cleaned['year'].between(2020, 2023)]
y_train = y[df_cleaned['year'].between(2020, 2023)]

# 测试集：2024
X_test = X[df_cleaned['year'] == 2024]
y_test = y[df_cleaned['year'] == 2024]

# 导入估计器
```

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()

lr.fit(X_train, y_train)
f"LinearRegression在训练集上的R2: {lr.score(X_train, y_train):.3f}"
f"LinearRegression在测试集上的R2: {lr.score(X_test, y_test):.3f}"
```

Out[140…

```
▾ LinearRegression  ⓘ ?

▶ Parameters
```

Out[140…

```
'LinearRegression在训练集上的R2: 0.054'
```

Out[140…

```
'LinearRegression在测试集上的R2: 0.040'
```

GBDT

In [141…

```
from sklearn.ensemble import GradientBoostingRegressor

X=df_cleaned[['1/A_it-1', 'REV_it-1/A_it-1']]
y=df_cleaned['DISEXP_it/A_it-1']

# 训练集: 2020 - 2023
X_train = X[df_cleaned['year'].between(2020, 2023)]
y_train = y[df_cleaned['year'].between(2020, 2023)]

# 测试集: 2024
X_test = X[df_cleaned['year'] == 2024]
y_test = y[df_cleaned['year'] == 2024]


# 导入估计器
gbreg = GradientBoostingRegressor(max_depth=2, n_estimators=3, learning_rate=0.1

# 训练模型（拟合数据）
gbreg.fit(X_train, y_train)

# 预测数据（应用模型）
gbreg.predict(X_test[:5])

# 评估模型
f"GradientBoostingRegressor在训练集上的R2: {gbreg.score(X_train, y_train):.3f}"
f"GradientBoostingRegressor在测试集上的R2: {gbreg.score(X_test, y_test):.3f}"
```

Out[141…

```
▾ GradientBoostingRegressor  ⓘ ?

▶ Parameters
```

Out[141…

```
array([0.07795326, 0.07390233, 0.06970382, 0.07230166, 0.07795326])
```

Out[141…

```
'GradientBoostingRegressor在训练集上的R2: 0.056'
```

Out[141…

```
'GradientBoostingRegressor在测试集上的R2: 0.026'
```

3. 生产成本管理模型

营业成本

sk包--线性模型

In [142…

```
df_cleaned.columns
```

Out[142…

```
Index(['code', 'stknme', 'listingDate', 'year', 'CFO_it/A_it-1',
       'PROD_it/A_it-1', 'DISEXP_it/A_it-1', '1/A_it-1', 'REV_it/A_it-1',
       'ΔREV_it/A_it-1', 'ΔREV_it-1/A_it-1', 'REV_it-1/A_it-1'],
      dtype='object')
```

In [143…

```
X=df_cleaned[['1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1','ΔREV_it-1/A_it-1']]
y=df_cleaned['PROD_it/A_it-1']

# 训练集：2020 - 2023
X_train = X[df_cleaned['year'].between(2020, 2023)]
y_train = y[df_cleaned['year'].between(2020, 2023)]

# 测试集：2024
X_test = X[df_cleaned['year'] == 2024]
y_test = y[df_cleaned['year'] == 2024]

# 导入估计器
from sklearn.linear_model import LinearRegression

lr=LinearRegression()

lr.fit(X_train, y_train)
f"LinearRegression在训练集上的R2: {lr.score(X_train, y_train):.3f}"
f"LinearRegression在测试集上的R2: {lr.score(X_test, y_test):.3f}"
```

Out[143…

▾ LinearRegression ⓘ ?

▶ Parameters

Out[143…

```
'LinearRegression在训练集上的R2: 0.934'
```

Out[143…

```
'LinearRegression在测试集上的R2: 0.920'
```

GBDT

In [144…

```
from sklearn.ensemble import GradientBoostingRegressor

X=df_cleaned[['1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1','ΔREV_it-1/A_it-1']]
y=df_cleaned['PROD_it/A_it-1']

# 训练集：2020 - 2023
X_train = X[df_cleaned['year'].between(2020, 2023)]
y_train = y[df_cleaned['year'].between(2020, 2023)]

# 测试集：2024
X_test = X[df_cleaned['year'] == 2024]
y_test = y[df_cleaned['year'] == 2024]
```

```
# 导入估计器
gbreg = GradientBoostingRegressor(max_depth=2, n_estimators=3, learning_rate=0.1

# 训练模型（拟合数据）
gbreg.fit(X_train, y_train)

# 预测数据（应用模型）
gbreg.predict(X_test[:5])

# 评估模型
f"GradientBoostingRegressor在训练集上的R2: {gbreg.score(X_train, y_train):.3f}"
f"GradientBoostingRegressor在测试集上的R2: {gbreg.score(X_test, y_test):.3f}"
```

Out[144…

```
▾ GradientBoostingRegressor
▶ Parameters
```

Out[144…

```
array([0.70054893, 0.58172215, 0.52488238, 0.52488238, 0.63185631])
```

Out[144…

```
'GradientBoostingRegressor在训练集上的R2: 0.504'
```

Out[144…

```
'GradientBoostingRegressor在测试集上的R2: 0.402'
```

4. 销售管理模型

CFO

sk包--线性模型

In [145…

```
X=df_cleaned[['1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1']]
y=df_cleaned["CFO_it/A_it-1"]

# 训练集: 2020 - 2023
X_train = X[df_cleaned['year'].between(2020, 2023)]
y_train = y[df_cleaned['year'].between(2020, 2023)]

# 测试集: 2024
X_test = X[df_cleaned['year'] == 2024]
y_test = y[df_cleaned['year'] == 2024]

# 导入估计器
from sklearn.linear_model import LinearRegression

lr=LinearRegression()

lr.fit(X_train, y_train)
f"LinearRegression在训练集上的R2: {lr.score(X_train, y_train):.3f}"
f"LinearRegression在测试集上的R2: {lr.score(X_test, y_test):.3f}"
```

Out[145…

```
▾ LinearRegression
▶ Parameters
```

```
Out[145…  'LinearRegression在训练集上的R2: 0.185'

Out[145…  'LinearRegression在测试集上的R2: -0.054'
```

GBDT

```
In [146…  from sklearn.ensemble import GradientBoostingRegressor

          X=df_cleaned[['1/A_it-1', 'REV_it/A_it-1', 'ΔREV_it/A_it-1']]
          y=df_cleaned["CFO_it/A_it-1"]

          # 训练集：2020 - 2023
          X_train = X[df_cleaned['year'].between(2020, 2023)]
          y_train = y[df_cleaned['year'].between(2020, 2023)]

          # 测试集：2024
          X_test = X[df_cleaned['year'] == 2024]
          y_test = y[df_cleaned['year'] == 2024]


          # 导入估计器
          gbreg = GradientBoostingRegressor(max_depth=2, n_estimators=3, learning_rate=0.1

          # 训练模型（拟合数据）
          gbreg.fit(X_train, y_train)

          # 预测数据（应用模型）
          gbreg.predict(X_test[:5])

          # 评估模型
          f"GradientBoostingRegressor在训练集上的R2：{gbreg.score(X_train, y_train):.3f}"
          f"GradientBoostingRegressor在测试集上的R2：{gbreg.score(X_test, y_test):.3f}"
```

```
Out[146…  ▾ GradientBoostingRegressor   i  ?

          ▶ Parameters

Out[146…  array([0.07276299, 0.07276299, 0.06186525, 0.06186525, 0.07276299])

Out[146…  'GradientBoostingRegressor在训练集上的R2：0.280'

Out[146…  'GradientBoostingRegressor在测试集上的R2：-0.093'
```