In [1]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import Lasso
from sklearn.model_selection import GridSearchCV, KFold
from sklearn.metrics import mean_squared_error, r2_score
import pickle

# 设置随机种子以获得可重复的结果
np.random.seed(42)

data = pd.read_csv('/home/featurize/Sample_fake_Zscore.csv')

X = data.drop(['Age'], axis=1)
y = data['Age']

lasso = Lasso()

param_grid = {'alpha': np.logspace(-4, 4, 9)}

n_bootstrap = 500
n_samples = len(data)
predicted_ages = []
model_details = []

for i in range(n_bootstrap):
    bootstrap_indices = np.random.choice(range(n_samples), n_samples, replace=Tr
    X_boot = X.iloc[bootstrap_indices]
    y_boot = y.iloc[bootstrap_indices]

    cv = KFold(n_splits=5, shuffle=True, random_state=42)
    grid_search = GridSearchCV(lasso, param_grid, cv=cv)
    grid_search.fit(X_boot, y_boot)

    best_alpha = grid_search.best_params_['alpha']

    # 使用最佳超参数训练模型
    lasso = Lasso(alpha=best_alpha)
    lasso.fit(X_boot, y_boot)

    #进行模型评估
    y_pred = lasso.predict(X)
    mse = mean_squared_error(y, y_pred)
    r2 = r2_score(y, y_pred)

    #保存模型评估结果和模型细节
    model_details.append({
        'bootstrap_iteration': i+1,
        'best_alpha': best_alpha,
        'MSE': mse,
        'R2': r2
    })

    predicted_ages.append(y_pred)

# 取预测年龄的平均值
mean_predicted_age = np.mean(predicted_ages, axis=0)

# 将模型详情保存到磁盘
```

```
with open('model_details.pkl', 'wb') as file:
    pickle.dump(model_details, file)

# 将平均预测年龄保存到CSV文件中
average_age_df = pd.DataFrame({
    'predicted_age': mean_predicted_age
})
average_age_df.to_csv('/home/featurize/predicted_ages.csv', index=False)
```

In [3]:
```
average_age_df
```

Out[3]:

	predicted_age
0	57.766067
1	54.737502
2	59.845198
3	31.203989
4	36.407598
...	...
195	61.831887
196	57.451063
197	56.278840
198	71.123238
199	33.191743

200 rows × 1 columns

In [8]:
```
! pip install statsmodels
```

```
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Collecting statsmodels
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/39/88/d8cd64c8c56131a796
215ce7f80ebb73e97200e6e6de26580cd20ae56e3e/statsmodels-0.14.1-cp310-cp310-manylin
ux_2_17_x86_64.manylinux2014_x86_64.whl (10.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 10.8/10.8 MB 127.5 MB/s eta 0:00:00a
0:00:01
Requirement already satisfied: numpy<2,>=1.18 in /environment/miniconda3/lib/pyth
on3.10/site-packages (from statsmodels) (1.24.1)
Requirement already satisfied: scipy!=1.9.2,>=1.4 in /environment/miniconda3/lib/
python3.10/site-packages (from statsmodels) (1.11.3)
Requirement already satisfied: pandas!=2.1.0,>=1.0 in /environment/miniconda3/li
b/python3.10/site-packages (from statsmodels) (2.1.2)
Collecting patsy>=0.5.4 (from statsmodels)
  Downloading https://pypi.tuna.tsinghua.edu.cn/packages/43/f3/1d311a09c34f14f597
3bb0bb0dc3a6e007e1eda90b5492d082689936ca51/patsy-0.5.6-py2.py3-none-any.whl (233
kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 233.9/233.9 kB 136.3 MB/s eta 0:00:00
Requirement already satisfied: packaging>=21.3 in /environment/miniconda3/lib/pyt
hon3.10/site-packages (from statsmodels) (23.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /environment/miniconda3/
lib/python3.10/site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /environment/miniconda3/lib/python
3.10/site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2023.3.post1)
Requirement already satisfied: tzdata>=2022.1 in /environment/miniconda3/lib/pyth
on3.10/site-packages (from pandas!=2.1.0,>=1.0->statsmodels) (2023.3)
Requirement already satisfied: six in /environment/miniconda3/lib/python3.10/site
-packages (from patsy>=0.5.4->statsmodels) (1.16.0)
Installing collected packages: patsy, statsmodels
Successfully installed patsy-0.5.6 statsmodels-0.14.1
```

In [9]:

```python
from statsmodels.nonparametric.smoothers_lowess import lowess

# 已经有了预测年龄和实际年龄的数据
predicted_ages_df = pd.read_csv('/home/featurize/predicted_ages.csv')
chronological_ages = data['Age']

# 使用Lowess函数进行局部回归拟合
fraction = 2/3  # 给定的分数参数
lowess_results = lowess(chronological_ages, predicted_ages_df['predicted_age'],

# 从lowess_results中提取拟合的预期年龄值
expected_ages = lowess_results[:, 1]

# 计算 Age Gap，每个样本的预测年龄与预期年龄之间的差异
age_gaps = predicted_ages_df['predicted_age'] - expected_ages

# 将Age Gap结果添加到预测年龄的DataFrame中
predicted_ages_df['Age_Gap'] = age_gaps

# 查看Age Gap结果
print(predicted_ages_df[['predicted_age', 'Age_Gap']])

# 输出到CSV文件查看完整数据
predicted_ages_df.to_csv('/home/featurize/predicted_ages_with_age_gaps.csv', ind
```

```
     predicted_age    Age_Gap
0        57.766067  34.562375
1        54.737502  31.413822
2        59.845198  35.444471
3        31.203989   6.408346
4        36.407598  11.461033
..             ...        ...
195      61.831887 -10.648916
196      57.451063 -15.089878
197      56.278840 -16.430109
198      71.123238  -2.105516
199      33.191743 -40.968112

[200 rows x 2 columns]
```

In [11]:

```
from scipy import stats
import statsmodels.api as sm

# 计算 predicted_age 和 Age 的 Pearson 相关系数
orig_r, _ = stats.pearsonr(predicted_ages_df['predicted_age'], chronological_age
print("Pearson correlation coefficient (orig_r):", orig_r)

# 加载 'Sample_fake_Zscore_CDR.csv' 数据
cdr_data = pd.read_csv('/home/featurize/Sample_fake_Zscore_CDR.csv')

cdr_data['Age_Gap'] = predicted_ages_df['Age_Gap']

# 准备回归分析的数据 -- 'CDR-GLOB' 作为响应变量，'Age_Gap' 作为主要预测变量，并控
X = cdr_data[['Age_Gap', 'Sex', 'Age']]  # 解释变量
y = cdr_data['CDR-GLOB']  # 响应变量
X = sm.add_constant(X)  # 添加一个常数项

# 执行回归分析
model = sm.OLS(y, X).fit()

# 打印回归结果的摘要
print(model.summary())

# 将 Age_Gap 与 CDR-GLOB 的相关系数命名为 orig_beta
orig_beta = model.params['Age_Gap']
print("Adjusted correlation coefficient (orig_beta):", orig_beta)
```

```
Pearson correlation coefficient (orig_r): 0.8312965141085761
                            OLS Regression Results
==============================================================================
Dep. Variable:               CDR-GLOB   R-squared:                       0.861
Model:                            OLS   Adj. R-squared:                  0.859
Method:                 Least Squares   F-statistic:                     406.3
Date:                Sun, 03 Mar 2024   Prob (F-statistic):           7.67e-84
Time:                        19:53:03   Log-Likelihood:                -86.351
No. Observations:                 200   AIC:                             180.7
Df Residuals:                     196   BIC:                             193.9
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          1.0402      0.107      9.766      0.000       0.830       1.250
Age_Gap        0.0550      0.002     31.367      0.000       0.052       0.058
Sex            0.0217      0.053      0.408      0.684      -0.083       0.127
Age           -0.0011      0.002     -0.553      0.581      -0.005       0.003
==============================================================================
Omnibus:                       17.892   Durbin-Watson:                   0.888
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               20.629
Skew:                           0.786   Prob(JB):                     3.31e-05
Kurtosis:                       3.056   Cond. No.                         203.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
Adjusted correlation coefficient (orig_beta): 0.054984522485060305
```

In [12]:

```
orig_beta
```

Out[12]:

```
0.054984522485060305
```

In [14]:

```
from scipy import stats
import statsmodels.api as sm

# 预先定义变量
perm_r_values = []
perm_beta_values = []
n_permutations = 5

for i in range(n_permutations):
    # 生成特征 "Pro1" 的置换版本
    data_permuted = data.copy()
    data_permuted['Pro1'] = np.random.permutation(data['Pro1'])

    # 重新计算与置换特征相关的预测年龄
    X_perm = data_permuted.drop(['Age'], axis=1)
    lasso.fit(X_perm, y)
    predicted_ages_permuted = lasso.predict(X_perm)

    # 计算置换 Pearson 相关系数
    perm_r, _ = stats.pearsonr(predicted_ages_permuted, y)
    perm_r_values.append(perm_r)

    # 使用 Lowess 拟合预测年龄曲线
    lowess_results = lowess(y, predicted_ages_permuted, frac=fraction)
    expected_ages_permuted = lowess_results[:, 1]
```

```
    age_gaps_permuted = predicted_ages_permuted - expected_ages_permuted

    # 更新CDR数据集的Age_Gap
    cdr_data['Age_Gap'] = age_gaps_permuted

    # 重新运行回归分析
    X_cdr_permuted = cdr_data[['Age_Gap', 'Sex', 'Age']]
    X_cdr_permuted = sm.add_constant(X_cdr_permuted)
    model_permuted = sm.OLS(cdr_data['CDR-GLOB'], X_cdr_permuted).fit()

    # 获取置换回归系数
    perm_beta = model_permuted.params['Age_Gap']
    perm_beta_values.append(perm_beta)

# 计算置换的平均 Pearson 相关系数和回归系数
mean_perm_r = np.mean(perm_r_values)
mean_perm_beta = np.mean(perm_beta_values)

# 根据初步计算和置换后计算的结果得到特征重要性
fi_chrono = orig_r - mean_perm_r
feat_imp_bio = orig_beta - mean_perm_beta

print("Feature importance based on chronological age (fi_chrono):", fi_chrono)
print("Feature importance based on biological age (feat_imp_bio):", feat_imp_bic
```

```
Feature importance based on chronological age (fi_chrono): 0.29034881998812434
Feature importance based on biological age (feat_imp_bio): -1.2884288845061986
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: