第二次pj报告

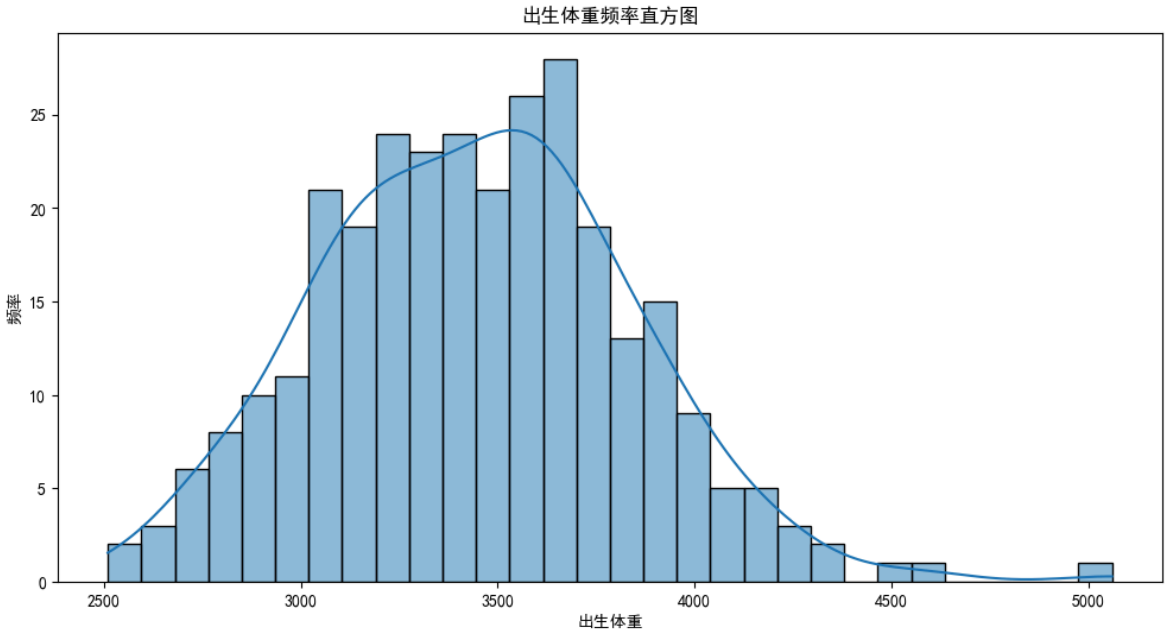
姓名：常雅静 学号：23307130470

第一部分 频率直方图和箱线图的绘制

In [35]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
import statsmodels.api as sm
import scipy.stats as stats
from IPython.display import Image

plt.rcParams['font.sans-serif'] = ['SimHei'] # 使其表格能正常显示中文
plt.rcParams['axes.unicode_minus'] = False # 解决负号现实问题

# 读取数据
data = pd.read_excel('fetal_weight_data.xlsx') #此处将胎重数据文件名改为了fetal_w

# 查看数据前几行
print(data.head())
```

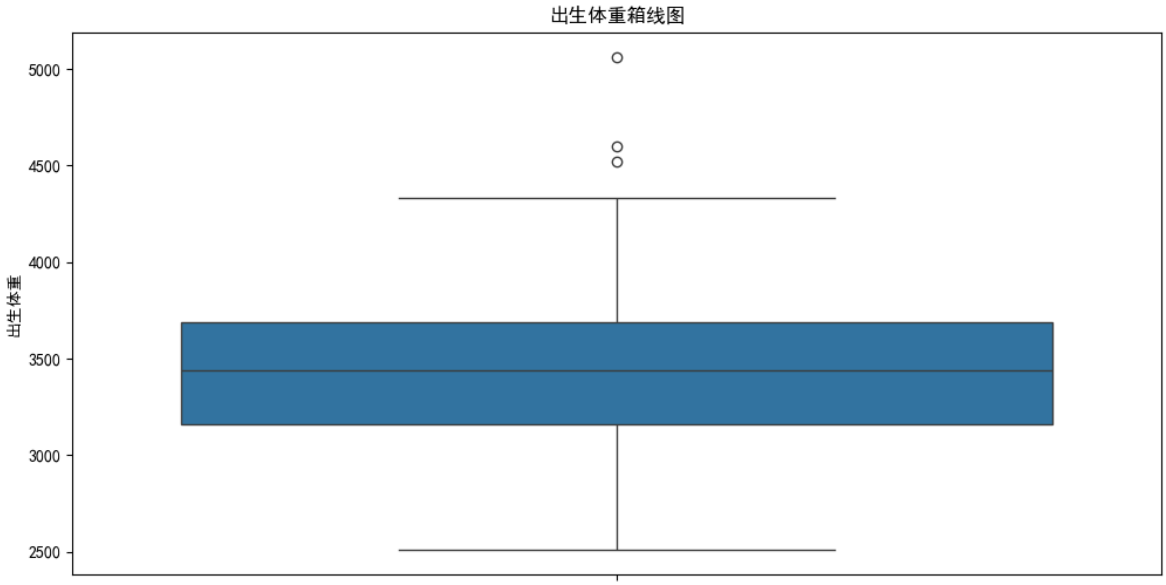
```
   出生体重  BPD(双顶径)  HC(头围)  AC(腹围)  FL(股骨长度)
0  3115       9.3    32.2    32.2       7.3
1  3220       9.9    33.0    33.0       7.1
2  2800       9.3    32.1    32.1       6.7
3  2870       9.0    31.2    31.2       6.4
4  3420       9.9    32.0    32.2       7.2
```

In [36]:
```
# 绘制频率直方图
plt.figure(figsize=(12, 6))
sns.histplot(data['出生体重'], bins=30, kde=True)
plt.title('出生体重频率直方图')
plt.xlabel('出生体重')
plt.ylabel('频率')
plt.show()
```



In [37]:

```
# 绘制箱线图
plt.figure(figsize=(12, 6))
sns.boxplot(y=data['出生体重'])
plt.title('出生体重箱线图')
plt.ylabel('出生体重')
plt.show()
y=data['出生体重']

a = np.quantile(y, 0.75)  # 上四分之一数
b = np.quantile(y, 0.25)  # 下四分之一数
print("（1）平均数：", np.mean(y))  # 打印均值
print("（2）中位数：", np.median(y))  # 打印中位数
print("（3）上四分之一数：", a)  # 打印上四分之一数
print("（4）下四分之一数：", b)  # 打印下四分之一数
up_limit = a + 1.5 * (a - b)  # 异常值判断标准
low_limit = b - 1.5 * (a - b)  # 异常值判断标准
y = np.sort(y)  # 对原始数据排序
shangjie = y[y < up_limit][-1]  # 除了异常值外的最大值
xiajie = y[y > low_limit][0]  # 除了异常值外的最小值
print("（5）上边界：", shangjie)  # 打印上界
print("（6）up:", up_limit)
print("（7）down:", low_limit)
print("（8）下边界：", xiajie)  # 打印下界
print('（9）异常值有：')
print(y[(y < low_limit) + (y > up_limit)])
```



（1）平均数： 3443.9133333333334
（2）中位数： 3440.0
（3）上四分之一数： 3690.5
（4）下四分之一数： 3158.75
（5）上边界： 4335
（6）up: 4488.125
（7）down: 2361.125
（8）下边界： 2510
（9）异常值有：
[4520 4600 5060]

根据这张图片可知，上面各个数据在图中的对应关系如下，中位数为3440.0，下四分之一数为3158.75，上四分之一数为3690.5,上边界为4335，下边界为2510，异常值包括4520,4600,5060

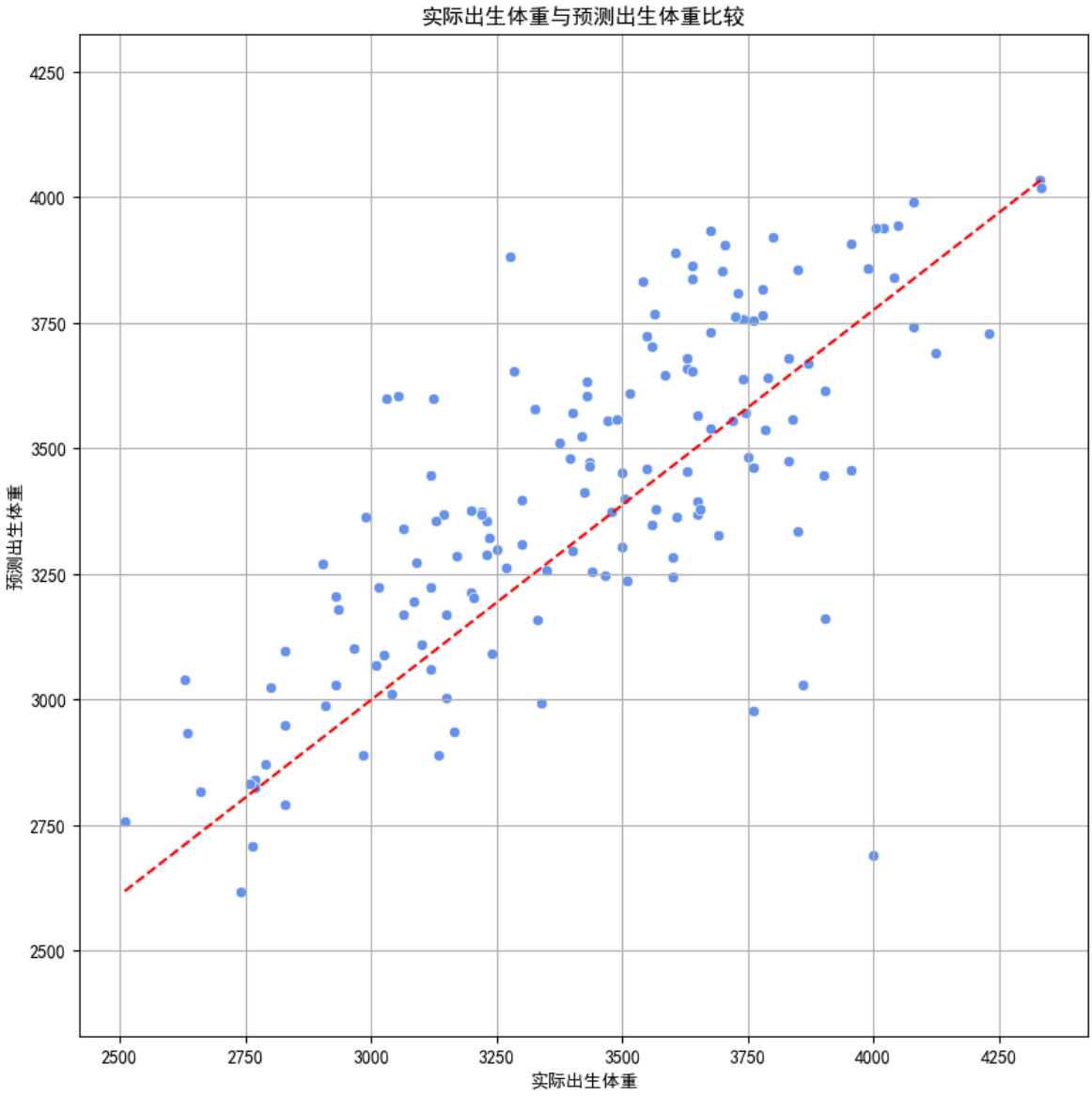
box_plot

第二部分 划分为训练集和测试集并进行线型回归拟合

In [41]:
```
# 划分训练集和测试集
X = data[['BPD(双顶径)', 'HC(头围)', 'AC(腹围)', 'FL(股骨长度)']]  # 参数数据
y = data['出生体重']

# 使用50%作为训练集，50%作为测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_

# 线性回归拟合
model = LinearRegression()
model.fit(X_train, y_train)

# 预测
y_pred = model.predict(X_test)

# 返回系数和截距
print("回归系数:", model.coef_)
print("截距:", model.intercept_)
```

回归系数: [223.04912922 -23.60198959 117.13927861 315.32509975]
截距: -4164.297946676423

In [42]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# 创建一个散点图，展示实际出生体重与模型预测值
plt.figure(figsize=(10, 10))
sns.scatterplot(x=y_test, y=y_pred,color='cornflowerblue')
plt.plot([y_test.min(), y_test.max()], [y_pred.min(), y_pred.max()], color='red'
plt.title('实际出生体重与预测出生体重比较')
plt.xlabel('实际出生体重')
plt.ylabel('预测出生体重')
plt.axis('equal')  # 使x轴和y轴比例一致
plt.grid()
plt.show()
```



第三部分 新方程的皮尔逊相关系数、MPE、MAPE分析和Hadlock4预测及其MPE、MAPE

皮尔逊

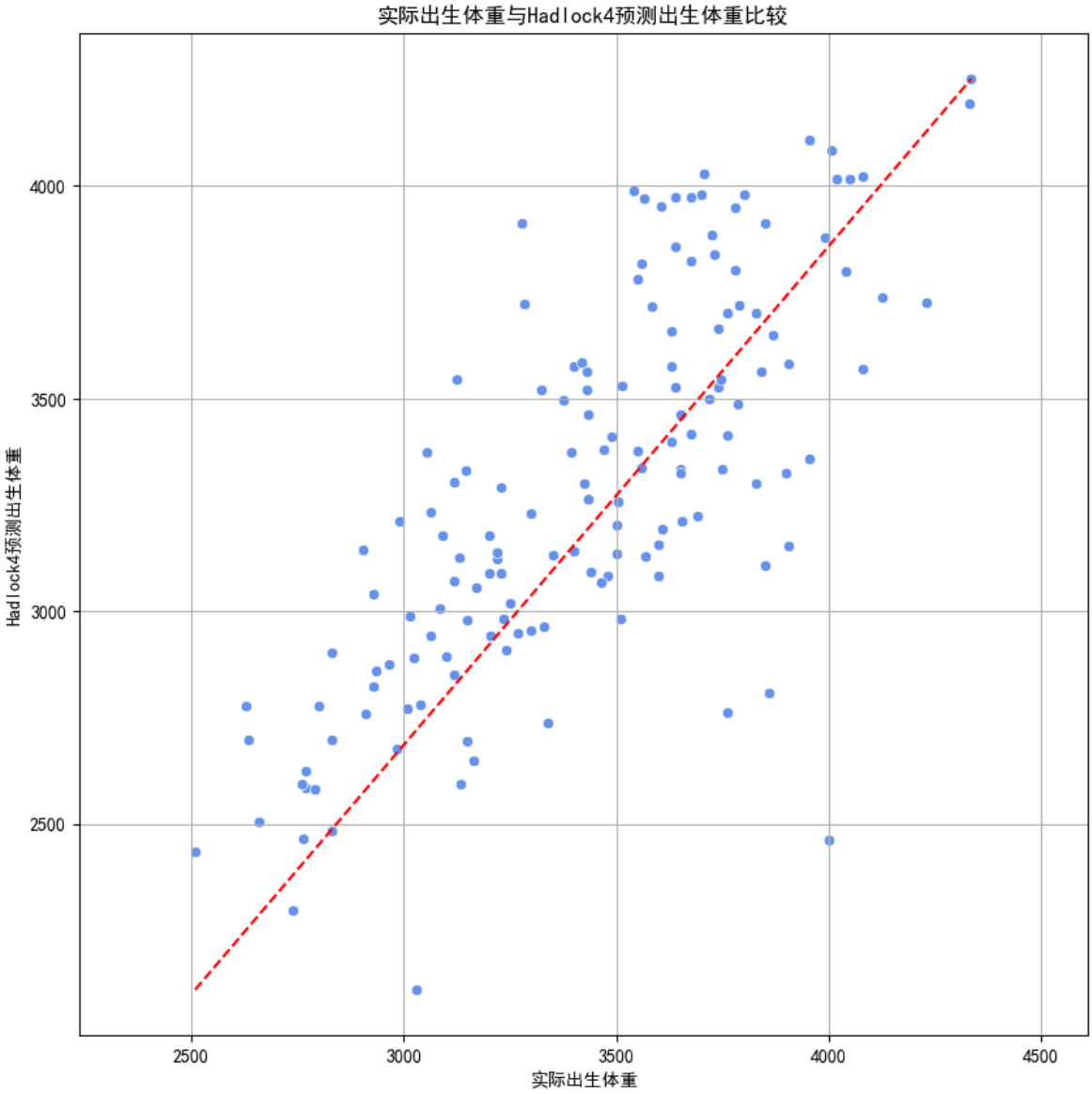
In [45]:
```
import numpy as np

# 计算MPE和MAPE
```

```
mpe = np.mean((y_pred - y_test) / y_test) * 100
mape = np.mean(np.abs(y_pred - y_test) / y_test) * 100

# 计算Pearson相关系数
pearson_r = np.corrcoef(y_pred, y_test)[0, 1]

# 输出结果
print(f'MPE（Mean Percentage Error 平均百分比误差）: {mpe:.4f}%')
print(f'MAPE（Mean Absolute Percentage Error 平均绝对百分比误差）: {mape:.4f}%')
print(f'Pearson correlation coefficient (R，皮尔逊相关系数): {pearson_r:.4f}')
```

```
MPE（Mean Percentage Error 平均百分比误差）: -0.1388%
MAPE（Mean Absolute Percentage Error 平均绝对百分比误差）: 5.7258%
Pearson correlation coefficient (R，皮尔逊相关系数): 0.7351
```

In [46]:

```
hadlock_pred = (10**(1.3596 - 0.00386 * X_test['AC(腹围)'] * X_test['FL(股骨长度
                     0.0064 * X_test['HC(头围)'] +
                     0.00061 * X_test['BPD(双顶径)']* X_test['AC(腹围)'] +
                     0.0424* X_test['AC(腹围)'] +
                     0.174* X_test['FL(股骨长度)']))

X = data[['BPD(双顶径)', 'HC(头围)', 'AC(腹围)', 'FL(股骨长度)']]  # 参数数据
y = data['出生体重']

# 使用50%作为训练集，50%作为测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_
plt.figure(figsize=(10, 10))
sns.scatterplot(x=y_test, y=hadlock_pred,color='cornflowerblue')
plt.plot([y_test.min(), y_test.max()], [hadlock_pred.min(), hadlock_pred.max()],
plt.title('实际出生体重与Hadlock4预测出生体重比较')
plt.xlabel('实际出生体重')
plt.ylabel('Hadlock4预测出生体重')
plt.axis('equal')  # 使x轴和y轴比例一致
plt.grid()
plt.show()


# 计算Hadlock方程的MPE和MAPE
mpe_hadlock = np.mean((hadlock_pred - y_test) / y_test) * 100
mape_hadlock = np.mean(np.abs(hadlock_pred - y_test) / y_test) * 100

# 输出Hadlock的指标
print(f'Hadlock MPE（Mean Percentage Error 平均百分比误差）: {mpe_hadlock:.4f}%'
print(f'Hadlock MAPE（Mean Absolute Percentage Error 平均绝对百分比误差）: {mape_
#计算并输出Hadlock的Pearson相关系数
pearson_r = np.corrcoef(hadlock_pred, y_test)[0, 1]
print(f'Pearson correlation coefficient (R，皮尔逊相关系数): {pearson_r:.4f}')
# 进行方差分析
f_value, p_value = stats.f_oneway(y_pred, hadlock_pred)
print(f"F值: {f_value:.2f}, p值: {p_value:.4f}")
```




```
Hadlock MPE（Mean Percentage Error 平均百分比误差）: -4.0545%
Hadlock MAPE（Mean Absolute Percentage Error 平均绝对百分比误差）: 7.4641%
Pearson correlation coefficient (R，皮尔逊相关系数): 0.7442
F值: 7.07, p值: 0.0083
```

零假设（H0）：新方程和Hadlock4方程的预测结果在均值上没有显著差异。

由于 p 值为0.0083，小于 0.05，所以可以拒绝零假设，认为新方程和Hadlock4方程的预测结果有显著差异。

而新方程的MPE、MAPE分别为-0.1388%， 5.7258；Hadlock的MPE、MAPE分别为-4.0545%， 7.4641。

前者误差值更小，说明回归模型更好，但值得注意的是Hadlock的皮尔逊相关系数略高，相关性较好%%

第四部分 K折交叉验证

K折交叉验证：用于模型的鲁棒性测试，通过将数据分为K个子集，以不同的训练-测试集组合多次训练模型，最终得到模型性能的平均得分。

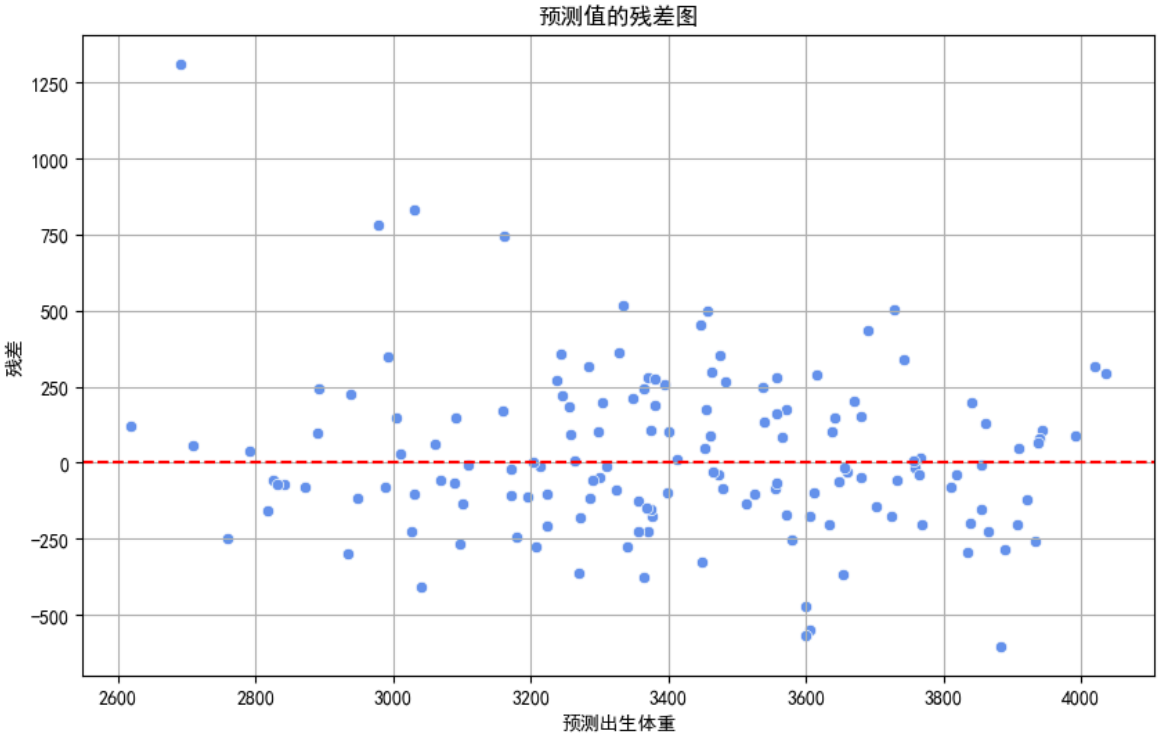
In [49]:
```
from sklearn.model_selection import cross_val_score

# 创建模型
```

```
model = LinearRegression()

# 使用K折交叉验证
cv_scores = cross_val_score(model, X, y, cv=10)

print(f'K折交叉验证得分: {cv_scores}')
print(f'平均得分: {np.mean(cv_scores):.8f}')
```

```
K折交叉验证得分: [0.67238772 0.74711468 0.68614961 0.3327542  0.72252574 0.18212485
 0.48599352 0.49406052 0.40983628 0.50805257]
平均得分: 0.52409997
```

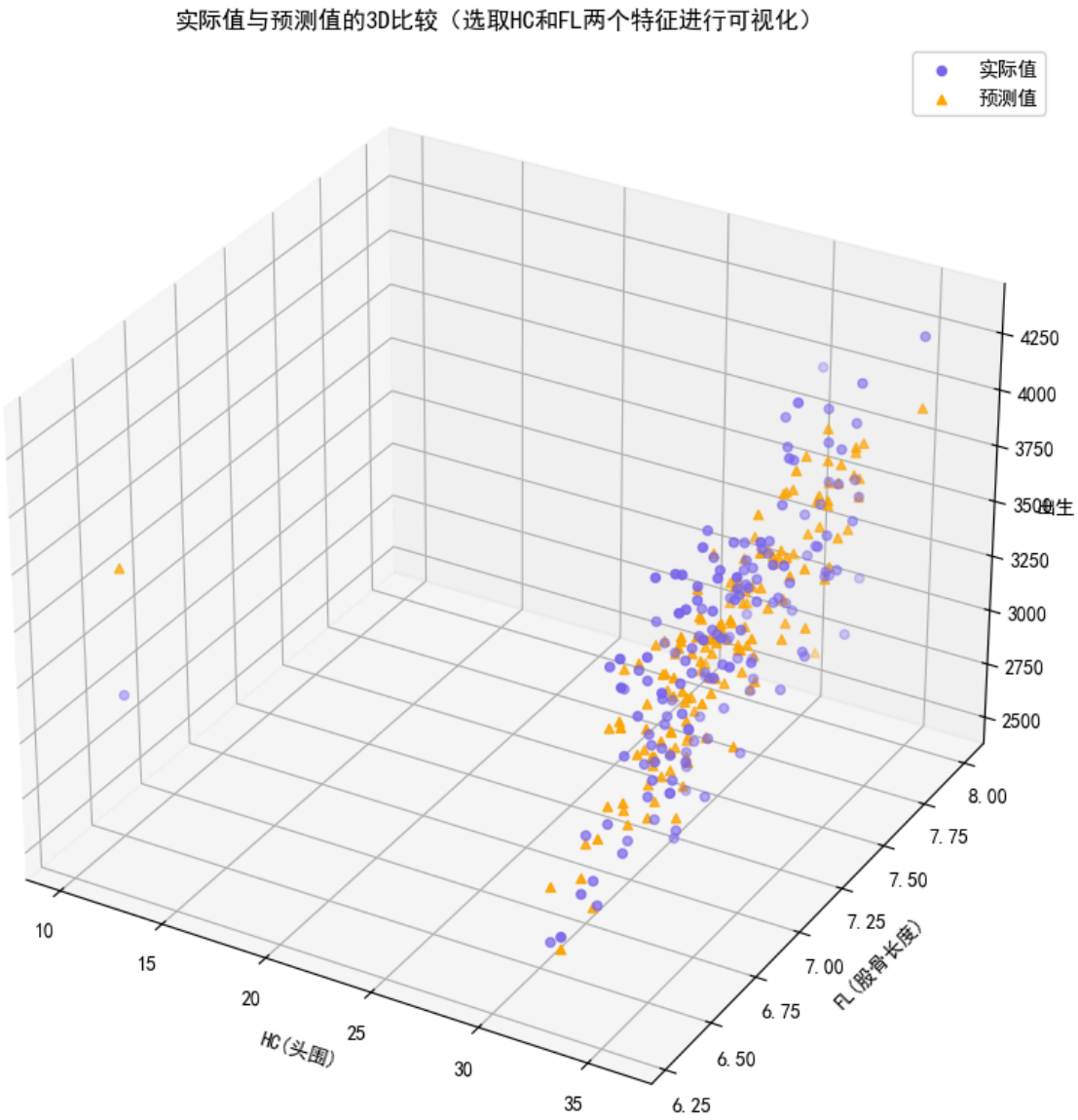
第五部分 扩展分析

In [51]:
```
# 计算残差
residuals = y_test - y_pred

# 创建残差图
plt.figure(figsize=(10, 6))
sns.scatterplot(x=y_pred, y=residuals,color='cornflowerblue')
plt.axhline(0, color='red', linestyle='--')
plt.title('预测值的残差图')
plt.xlabel('预测出生体重')
plt.ylabel('残差')
plt.grid()
plt.show()
```



In [52]:
```
from mpl_toolkits.mplot3d import Axes3D

# 创建3D图
fig = plt.figure(figsize=(15, 10))
ax = fig.add_subplot(111, projection='3d')

# 选取两个特征进行可视化（例如HC和FL）
ax.scatter(X_test['HC(头围)'], X_test['FL(股骨长度)'], y_test, c='mediumslateblu
```

```
ax.scatter(X_test['HC(头围)'], X_test['FL(股骨长度)'], y_pred, c='orange', marke

ax.set_title('实际值与预测值的3D比较（选取HC和FL两个特征进行可视化）')
ax.set_xlabel('HC(头围)')
ax.set_ylabel('FL(股骨长度)')
ax.set_zlabel('出生体重')
plt.legend()
plt.show()
```



In [53]:

```
from mpl_toolkits.mplot3d import Axes3D

# 创建3D图
fig = plt.figure(figsize=(15, 10))
ax = fig.add_subplot(111, projection='3d')

# 选取两个特征进行可视化（例如HC(头围)和AC）
ax.scatter(X_test['HC(头围)'], X_test['AC(腹围)'], y_test, c='royalblue', label=
ax.scatter(X_test['HC(头围)'], X_test['AC(腹围)'], y_pred, c='r', marker='^', la

ax.set_title('实际值与预测值的3D比较（选取HC和AC两个特征进行可视化）')
ax.set_xlabel('HC(头围)')
ax.set_ylabel('AC(腹围)')
ax.set_zlabel('出生体重')
```

```
plt.legend()
plt.show()
```