

# 实验八报告

姓名：宋卓伦 孔祥昊 刘孙延      组号：11

## 一、实验目的

根据生活习惯预测其 BMI 指数；  
利用 Python 工具包编写数据分析建模程序；  
掌握常用工具包的使用方法及新工具包的自主学习方法；  
结合具体问题选择合适工具包；  
通过团队协作、合理分工高效解决问题。

## 二、实验环境

Python 版本：Python 3.12

使用的库：scikit-learn, numpy, matplotlib, pandas

## 三、实验步骤

**1.数据集：**使用 [Estimation of Obesity Levels Based On Eating Habits and Physical Condition - UCI Machine Learning Repository](#) 公开数据集

Variables Table						
Variable Name	Role	Type	Demographic	Description	Units	Missing Values
Gender	Feature	Categorical	Gender			no
Age	Feature	Continuous	Age			no
Height	Feature	Continuous				no
Weight	Feature	Continuous				no
family_history_with_overweight	Feature	Binary		Has a family member suffered or suffers from overweight?		no
FAVC	Feature	Binary		Do you eat high caloric food frequently?		no
FCVC	Feature	Integer		Do you usually eat vegetables in your meals?		no
NCP	Feature	Continuous		How many main meals do you have daily?		no
CAEC	Feature	Categorical		Do you eat any food between meals?		no
SMOKE	Feature	Binary		Do you smoke?		no

**2.模型：**随机森林回归模型

**3.超参数调优：**使用 GridSearchCV 进行超参数搜索，以找到最佳的 n\_estimators 和 max\_depth 参数

**4.代码：**

## Python

导入所需的 Python 库

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import mean_squared_error
from sklearn.inspection import permutation_importance
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
```

读取 CSV 文件中的数据，并打印出前几行以查看数据集的样式

```
# 读取 CSV 数据

data = pd.read_csv('ObesityDataSet_raw_and_data_sinthetic.csv')

# 查看数据

print(data.head())
```

将数据集中的分类变量（性别、家族史、饮食习惯、吸烟和运动习惯）转换为数值变量，方便模型处理。这里使用了 `apply` 函数和 `lambda` 表达式来进行转换。提取特征变量 `x` 和目标变量 `y`。特征变量包括性别、年龄、身高、家族史等，目标变量 `y` 是根据体重和身高计算得到的 BMI。将数据集划分为训练集和测试集，其中 30% 的数据用作测试集，其余 70% 用作训练集。

```
data['Gender'] = data['Gender'].apply(lambda x: 1 if x == 'Male' else 0)

data['family_history_with_overweight'] =
data['family_history_with_overweight'].apply(lambda x: 1 if x == 'yes'
else 0)

data['FAVC'] = data['FAVC'].apply(lambda x: 1 if x == 'yes' else 0)
data['SMOKE'] = data['SMOKE'].apply(lambda x: 1 if x == 'yes' else 0)
data['SCC'] = data['SCC'].apply(lambda x: 1 if x == 'yes' else 0)
data['CALC'] = data['CALC'].apply(lambda x: 1 if x == 'yes' else 0)

# 提取特征和目标变量
X = data[['Gender', 'Age', 'Height', 'family_history_with_overweight',
'FAVC', 'SMOKE', 'SCC', 'CALC']]

y = data['Weight'] / (data['Height'] / 100) ** 2
```

```
# 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
random_state=13)
print('X_train shape:', X_train.shape)
print('X_test shape:', X_test.shape)
```

定义随机森林回归模型的参数, 并创建模型实例。然后使用训练集数据训练模型。

```
params = {
    "n_estimators": 100,
    "max_depth": 7
}

reg = RandomForestRegressor(**params, random_state=0)
reg.fit(X_train, y_train)
```

使用训练好的模型对测试集进行预测, 并绘制真实 BMI 和预测 BMI 的散点图。

```
# 预测并可视化结果
y_pred = reg.predict(X_test)
plt.scatter(y_test, y_pred)
plt.xlabel('True BMI')
plt.ylabel('Predicted BMI')
plt.title('BMI Prediction')
plt.show()
```

计算并打印测试集上的均方误差 (MSE), 用于评估模型性能。

使用排列重要性方法计算特征的重要性, 并绘制箱线图来展示每个特征的重要性。

```
# 计算 MSE
mse = mean_squared_error(y_test, y_pred)
print("测试集上的 MSE 指标: %.4f" % mse)

# 特征重要性分析
result = permutation_importance(reg, X_test, y_test, n_repeats=10,
random_state=13, scoring='neg_mean_squared_error')
sorted_idx = result.importances_mean.argsort()[::-1] # 降序排序
plt.boxplot(result.importances[sorted_idx].T, vert=False,
labels=np.array(X.columns)[sorted_idx])
plt.xlabel('Feature Importance')
plt.title("Feature Importance Analysis")
plt.show()
```

定义一个参数网格, 用于超参数优化。创建 RandomForestRegressor 模型实例, 并使用 GridSearchCV 进行超参数搜索, 以找到最佳的 n\_estimators 和 max\_depth 参数。

```
# 超参数优化
params_grid = {'n_estimators': [50, 100, 150], 'max_depth': [5, 7, 10]}
rf = RandomForestRegressor(random_state=0)
reg_search = GridSearchCV(estimator=rf, param_grid=params_grid, cv=5,
scoring='neg_mean_squared_error')
```

```
reg_search.fit(X_train, y_train)
print('最优超参数: ', reg_search.best_params_)
```

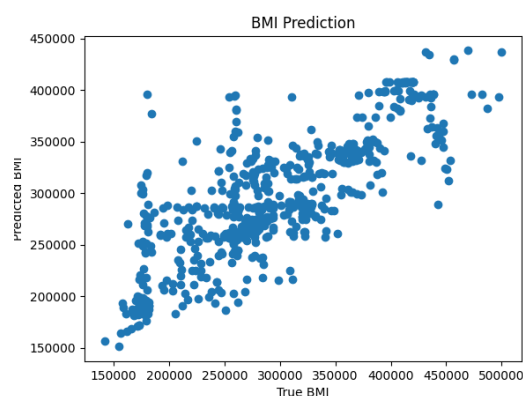
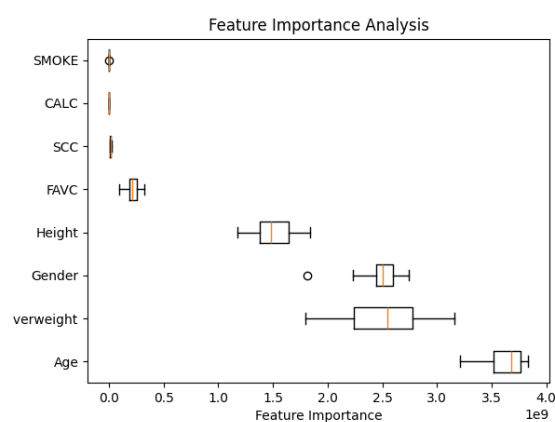
使用找到的最优超参数重新训练模型，并使用优化后的模型对测试集进行预测。计算并打印优化模型在测试集上的 MSE。

```
# 使用优化模型进行预测
y_pred_optimized = reg_search.predict(X_test)
mse_optimized = mean_squared_error(y_test, y_pred_optimized)
print("优化模型在测试集上的 MSE 指标: %.4f" % mse_optimized)
```

## 四、实验结果

```
D:\代码\py\pythonProject\venv\scripts\python.exe D:\代码\py\pythonProject\BMI12.py
  Gender  Age  Height  ...      CALC      MTRANS      NObyesdad
0  Female  21.0   1.62  ...      no  Public_Transportation  Normal_Weight
1  Female  21.0   1.52  ...  Sometimes  Public_Transportation  Normal_Weight
2   Male  23.0   1.80  ...  Frequently  Public_Transportation  Normal_Weight
3   Male  27.0   1.80  ...  Frequently      Walking  Overweight_Level_I
4   Male  22.0   1.78  ...  Sometimes  Public_Transportation  Overweight_Level_II

[5 rows x 17 columns]
X_train shape: (1477, 8)
X_test shape: (634, 8)
```



## 五、实验分析

在运行代码的过程中，出现了均方误差巨大的现象。在排除了不是模型引用的问题之后，我们发现在单位转换上结果出现了不一致。将上面原代码中的“/100”去掉之后，身高的单位变为米，BMI 回落到正常范围，MSE 也控制在了一个较小的范围。

更改的实验代码如下：

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, GridSearchCV

from sklearn.metrics import mean_squared_error
from sklearn.inspection import permutation_importance
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression

# 读取CSV数据

data = pd.read_csv('ObesityDataSet_raw_and_data_synthetic.csv')

# 查看数据

print(data.head())

# 选择特征和目标变量

# 特征：性别（编码为0或1）、年龄、身高、家族史（编码为0或1）、其他可能的特征

# 目标变量：BMI（通过体重（kg）/ 身高（m）^2 计算得到）

# 编码分类变量

data['Gender'] = data['Gender'].apply(lambda x: 1 if x == 'Male' else 0)

data['family_history_with_overweight'] = data['family_history_with_overweight'].apply(lambda x: 1 if x == 'yes' else 0)
```

```
data['FAVC'] = data['FAVC'].apply(lambda x: 1 if x == 'yes' else 0)

data['SMOKE'] = data['SMOKE'].apply(lambda x: 1 if x == 'yes' else 0)

data['SCC'] = data['SCC'].apply(lambda x: 1 if x == 'yes' else 0)

data['CALC'] = data['CALC'].apply(lambda x: 1 if x == 'yes' else 0)

# 提取特征和目标变量

X = data[['Gender', 'Age', 'Height', 'family_history_with_overweight', 'FAVC', 'SMOKE', 'SCC', 'CALC']]

y = data['Weight'] / (data['Height']) ** 2 # 注意: 身高可能需要从米转换为厘米, 或者保持单位一致

# 划分训练集和测试集

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=13)

print('X_train shape:', X_train.shape)

print('X_test shape:', X_test.shape)

# 训练随机森林回归模型

params = {

    "n_estimators": 150,

    "max_depth": 10

}

reg = RandomForestRegressor(**params, random_state=15)

reg.fit(X_train, y_train)

# 预测并可视化结果
```

```
y_pred = reg.predict(X_test)

plt.scatter(y_test, y_pred)

plt.xlabel('True BMI')

plt.ylabel('Predicted BMI')

plt.title('BMI Prediction')

plt.show()

# 计算MSE

mse = mean_squared_error(y_test, y_pred)

print("测试集上的 MSE 指标: %.4f" % mse)

# 特征重要性分析

result = permutation_importance(reg, X_test, y_test, n_repeats=10,
                                random_state=13, scoring='neg_mean_squared_error')

sorted_idx = result.importances_mean.argsort()[::-1] # 降序排序

plt.boxplot(result.importances[sorted_idx].T, vert=False, labels=p.array(X.columns)[sorted_idx])

plt.xlabel('Feature Importance')

plt.title("Feature Importance Analysis")

plt.show()

# 超参数优化

params_grid = {'n_estimators': [50, 100, 150], 'max_depth': [5, 7, 10]}

rf = RandomForestRegressor(random_state=0)

reg_search = GridSearchCV(estimator=rf, param_grid=params_grid, cv=5, scoring='neg_mean_squared_error')
```

```

reg_search.fit(X_train, y_train)

print('最优超参数: ', reg_search.best_params_)

# 使用优化模型进行预测

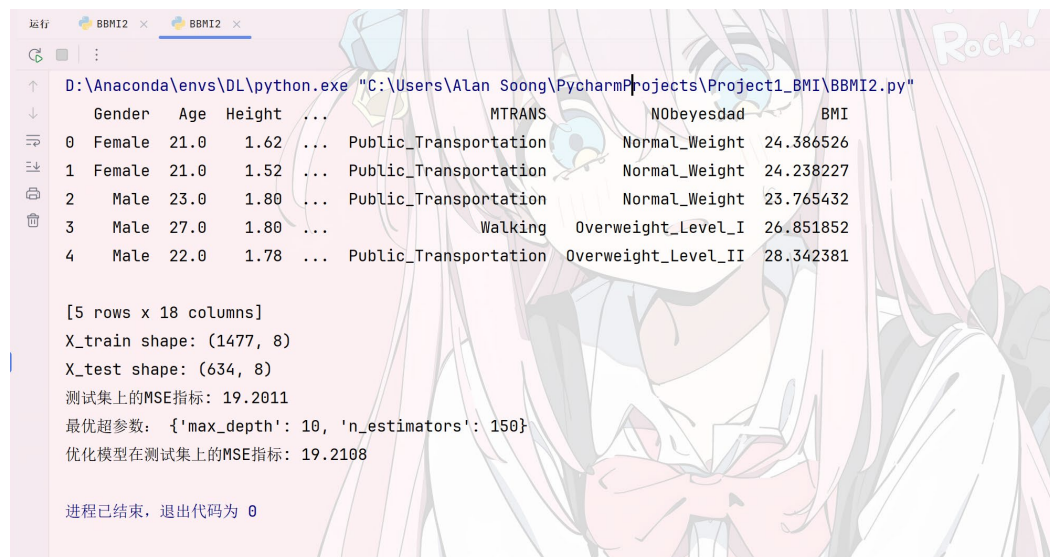
y_pred_optimized = reg_search.predict(X_test)

mse_optimized = mean_squared_error(y_test, y_pred_optimized)

print("优化模型在测试集上的MSE指标: %.4f" % mse_optimized)

```

实验代码输出结果如下：



```

运行 BBMI2 x BBMI2 x
D:\Anaconda\envs\DL\python.exe "C:\Users\Alan Soong\PycharmProjects\Project1_BMI\BBMI2.py"

```

	Gender	Age	Height	...	MTRANS	NObeyesdad	BMI
0	Female	21.0	1.62	...	Public_Transportation	Normal_Weight	24.386526
1	Female	21.0	1.52	...	Public_Transportation	Normal_Weight	24.238227
2	Male	23.0	1.80	...	Public_Transportation	Normal_Weight	23.765432
3	Male	27.0	1.80	...	Walking	Overweight_Level_I	26.851852
4	Male	22.0	1.78	...	Public_Transportation	Overweight_Level_II	28.342381

```

[5 rows x 18 columns]
X_train shape: (1477, 8)
X_test shape: (634, 8)
测试集上的MSE指标: 19.2011
最优超参数: {'max_depth': 10, 'n_estimators': 150}
优化模型在测试集上的MSE指标: 19.2108

进程已结束，退出代码为 0

```

