Machine Learning

HW3：Maximum Likelihood (ML) and Bayesian linear regression

姓名：陳毅

學號：111063577

老師：孫民

日期：5/22

1. train_test_val_split(x,y,random = None)

由於資料集的數據雜亂且隨機，我們必須定義一個函式去把資料分成訓練集與驗證集和測試集，由於題目要求我們分為 70：10：20 的比例。

我們先定義一個叫 **train_test_ val_split(x,y,random = None)**的函式，首先我們設定隨機種子以便產生重複的隨機數序列，確保每次運行的隨機數序列是相同的，接著我們把資料打亂，使用 **np.random.permutation(len(x))**來生成長度為 x(特徵標籤)的隨機排列序列，我們把訓練集*0.7，驗證集*0.1，最後再用總數扣除得到測試集，最後合併回傳。

2. Maximum Likelihood and Least Squares

我們先把訓練集的資料分成為特徵與標籤的部分，再計算 feature vector (Φ)，我們利用公式 wML=(ΦTΦ)-1ΦTt，將訓練標籤帶入 t 將訓練集計算出來的 feature vector 帶入 Φ 來算出 model 的 weight。我們只需要將測試資料計算出的 Φ 與對應的 weight 相乘並加總，就可得到預測機率。我們找的公式 $\Phi(x)=[\phi 1(x),\phi 2(x),\ldots,\phi P+2(x)]T$ 與參考書上的 Φ 差一個轉置，在計算 Φ 時是按照課本上的擺法擺放。由於 Φ 的維度為(n,P+2)，weight 的維度為(P+2,1)，

利用矩陣乘法的特性，便可以直接得到預測的結果 y，維度為 (n,1)。

```
MSE of BLR = 73.17308991079602, MSE of MLR= 75.68989763262593.
```

3. Bayesian Linear Regression

我們利用 get_feature_vector 函式，去計算訓練資料的特徵向量 Φ。該函式的作用是根據指定的基底函數順序 O1、O2、O3、O4、O5 計算出特徵向量。為了將矩陣將用於正規化。使用公式(Φ^TΦ)^-1Φ^Tt 來計算模型的權重 weights。其中，train_phi.T@train_phi 表示特徵向量的轉置矩陣與自身的乘積，np.linalg.inv()表示做反矩陣，最後得到的權重將用於預測。預測結果 y_BLR_prediction 是根據公式$y(x,w)=\sum wj\phi j(x)$來計算的，其中$\phi j(x)$表示基底函數對於測試資料 x 的計算結果。

最後，函式返回預測結果 y_BLR_prediction。

```
MSE of BLR = 73.17308991079602, MSE of MLR= 75.68989763262593.
```

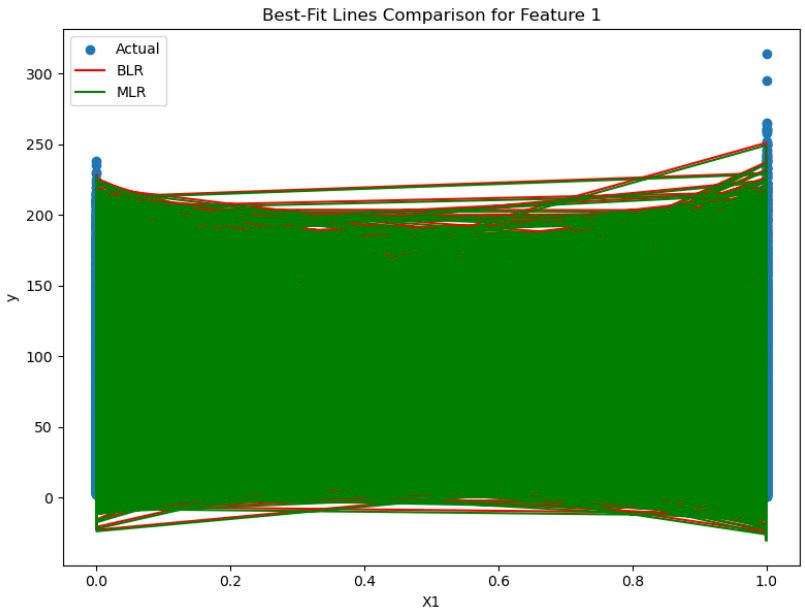
4. Discuss the difference between Maximum Likelihood and Bayesian Linear Regression

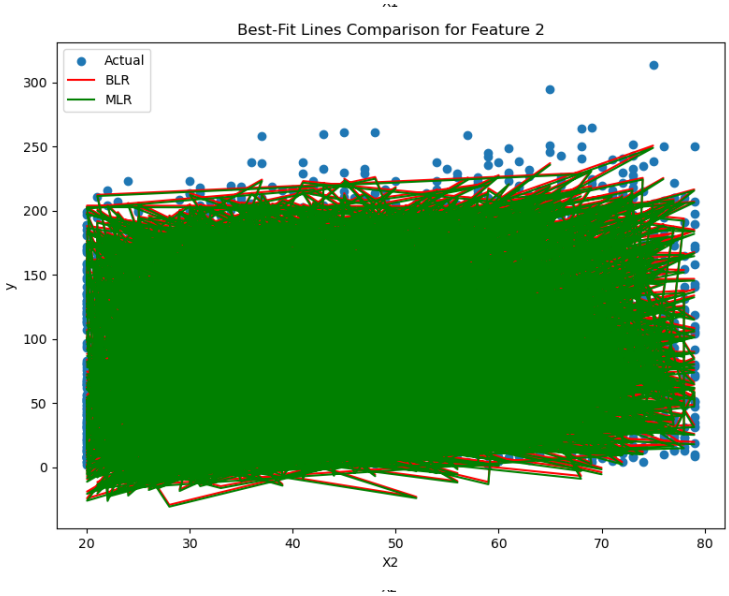
BLR 的優點為具有較好的 Robustness，對於異常值和離群點的影響較小。缺點為計算複雜度較高，需要進行矩陣的求逆運算，尤

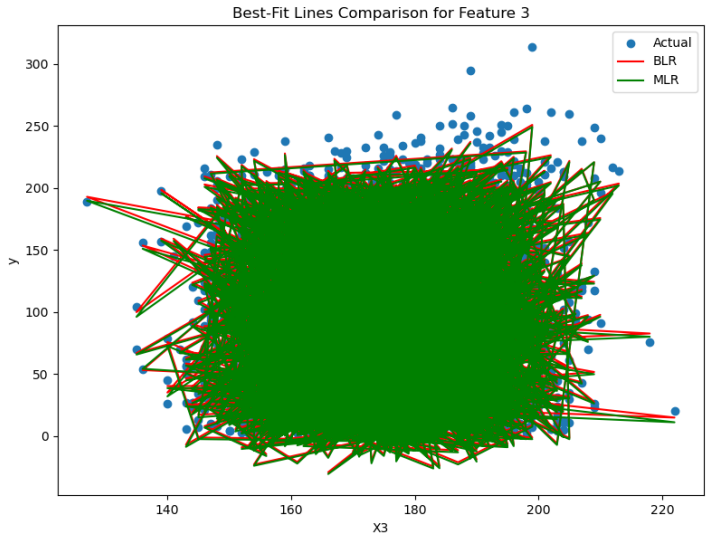
其當特徵維度較大時，計算時間會更長。BLR 對於模型的先驗分佈假設敏感，如果先驗分佈假設不準確，可能導致模型估計不準確。

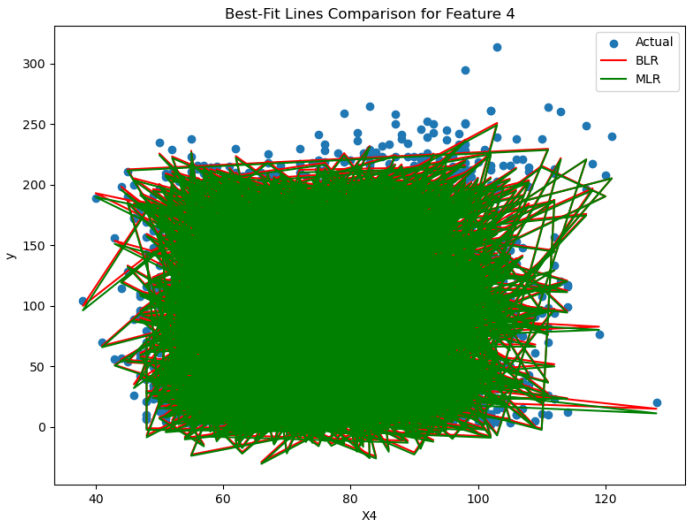
MLR 的優點對於特徵間的線性關係能夠較好地建模，當特徵與標籤之間存在線性關係時，MLR 可以提供較好的結果。但缺點為如果模型參數的不確定性沒有進行明確建模，無法提供參數的機率分佈估計。且 MLR 對於異常值和離群點較為敏感，這些極端值可能對模型的參數估計和預測結果產生較大影響。

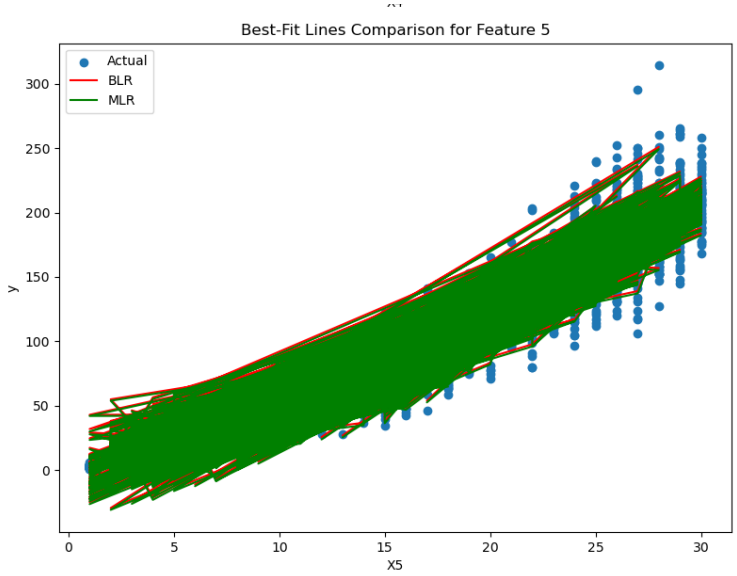
總體來說，BLR 通過引入貝葉斯統計的方法來處理模型的不確定性，提供更準確的結果。然而，BLR 的計算複雜度較高且對於先驗分佈假設敏感。MLR 計算較簡單，但無法提供參數的不確定性估計。

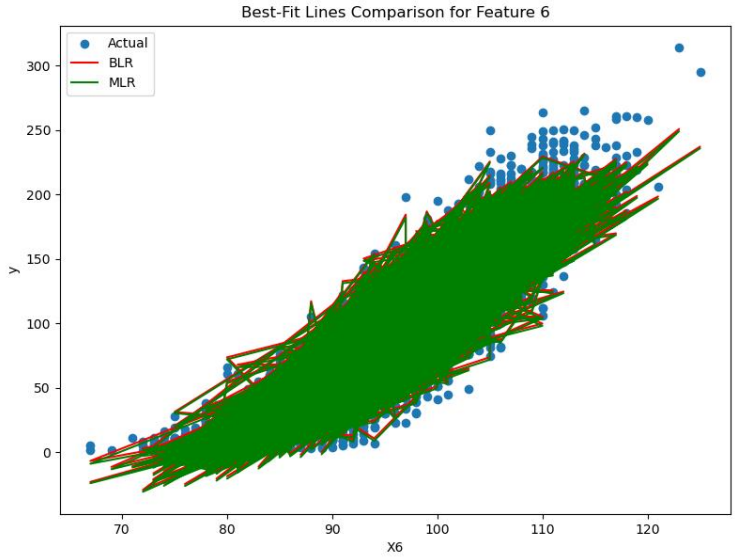


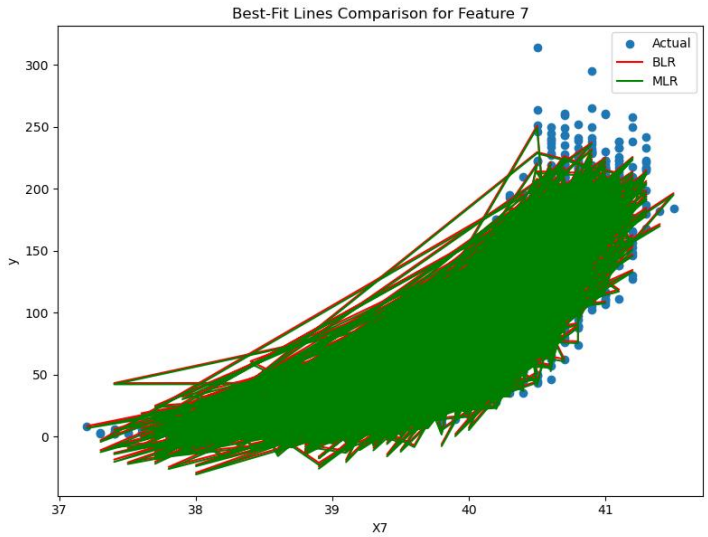

Best-Fit Lines Comparison for Feature 2
Actual
BLR
MLR
y
X2


Best-Fit Lines Comparison for Feature 3
Actual
BLR
MLR
y
X3


Best-Fit Lines Comparison for Feature 4
Actual
BLR
MLR
y
X4







以上為七種特徵別的預測圖，可以從圖中發現 BLR 與 MLR 大

致上都重疊在一起，而少部分誤差。

5. Implement any regression model

XGBRegressor 在計算效率和內存使用方面表現出色。它使用了優化的算法和數據結構，並且可以並行處理多個子模型，從而實現了高效的訓練和預測速度，並且通過使用多個樹模型的集成學習方式，能夠捕捉到數據中的非線性關係和交互作用，從而提供準確的預測結果。它能夠自適應地學習數據的特徵和分佈，並生成更強大的模型，且對於噪聲數據和異常值具有較好的 Robustness。它使用了樹模型的集成方法，可以有效地降低單個模型對於異常值和噪聲數據的過度擬合。

In [2]:
```
import pandas as pd
import numpy as np
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split


data_train = pd.read_csv('D:/train_data7.csv')
data_test = pd.read_csv('D:/test_data7.csv')


data_train['0'] = data_train['0'].map({'male': 1, 'female': 0})
data_test['0'] = data_test['0'].map({'male': 1, 'female': 0})


X_train = data_train.iloc[:, 1:-1].values
y_train = data_train.iloc[:, -1].values

X_test = data_test.iloc[:, 1:-1].values
y_test = data_test.iloc[:, -1].values


X_train, X_valid, y_train, y_valid = train_test_split(X_train, y_train, test_size=0.2, random_state=42)


model = XGBRegressor()
model.fit(X_train, y_train)


y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print('MSE:', mse)
```

```
MSE: 6.199811119152599
```

MSE：6.19

參考資料：

1. https://github.com/Mahe-999/Calories-Burnt-Predictor/blob/main/Calories_burnt_Predictor_.ipynb
2. https://github.com/Mahe-999/Calories-Burnt-Predictor