Machine Learning Homework 03

陳柏翔 4109061012

Problem 01：

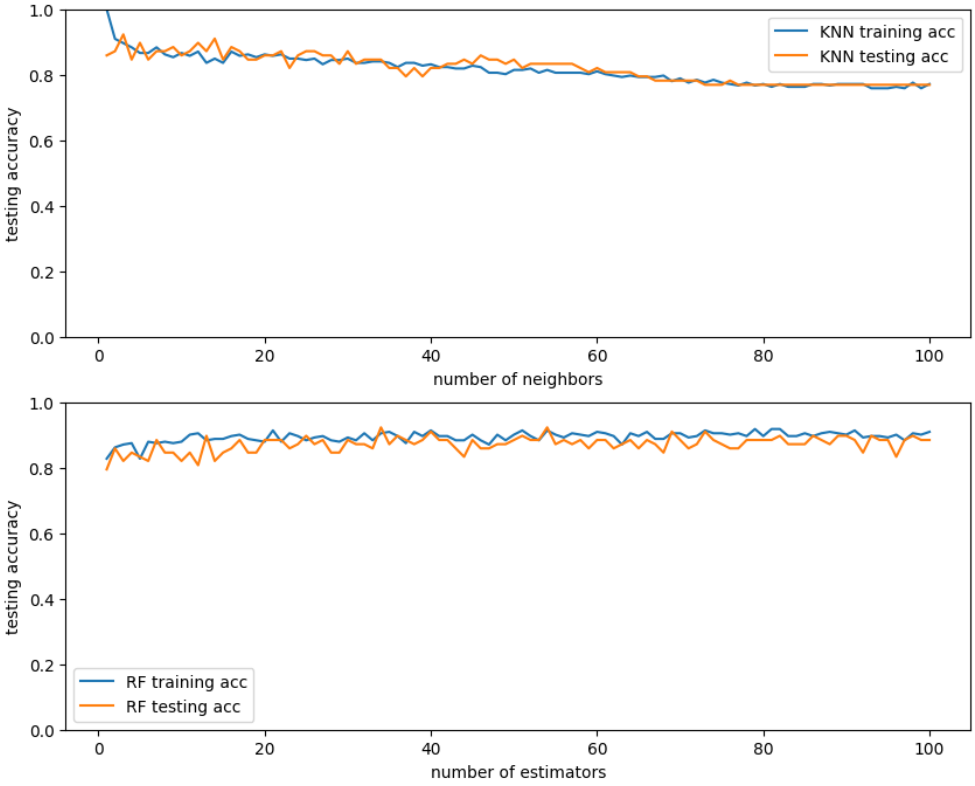
Code：

(詳見附檔 problem_1.py 或 GitHub 連結) (EEGuizhi 是我的 GitHub user name)

Result：

```
>> KNN algorithm
Best pred. n_neighbors = 3, Acc = 92.30769%

>> Random Forest algorithm
Best pred. n_estimators = 34, Acc = 92.30769%
```



“Number of neighbors 在 KNN 中的影響” 及 “Number of estimators 在 Random Forest 中的影響” 如上方輸出結果所示

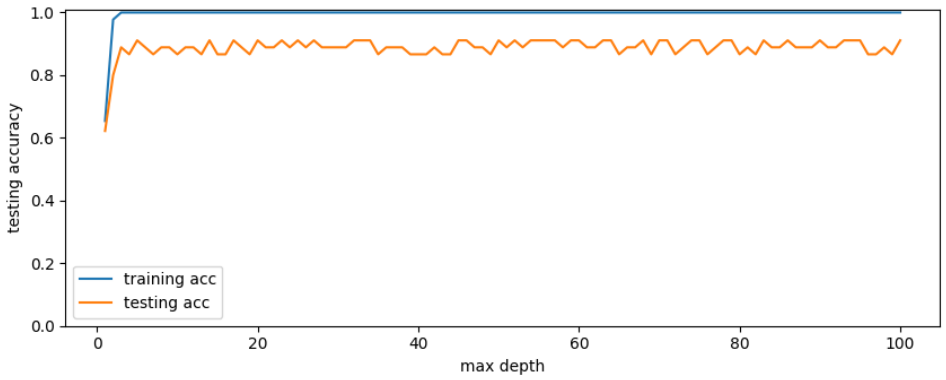
Problem 02：

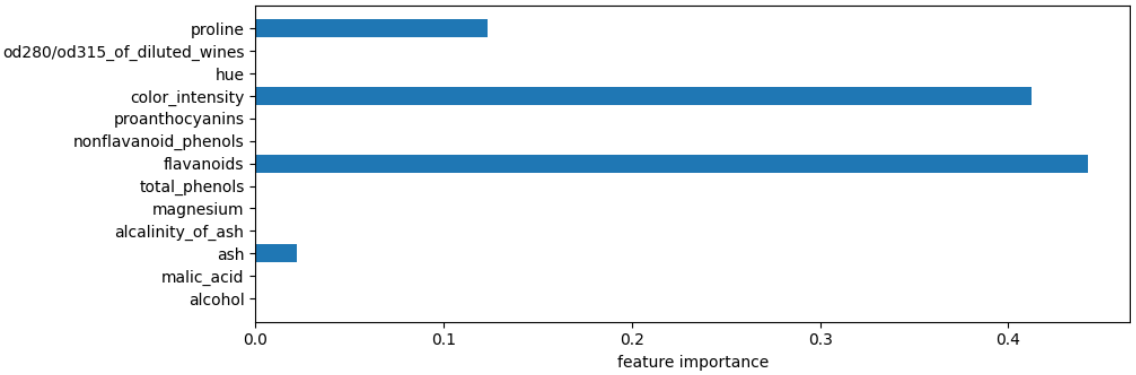
Code：

(詳見附檔 problem_2.py 或 GitHub 連結)

Result：

```
Best pred. max_depth = 5, Acc = 91.11111%
```





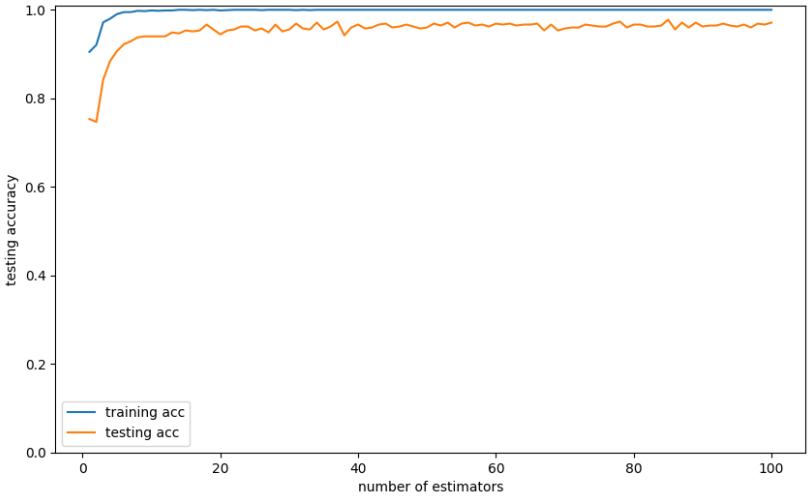
Problem 03：

Code：

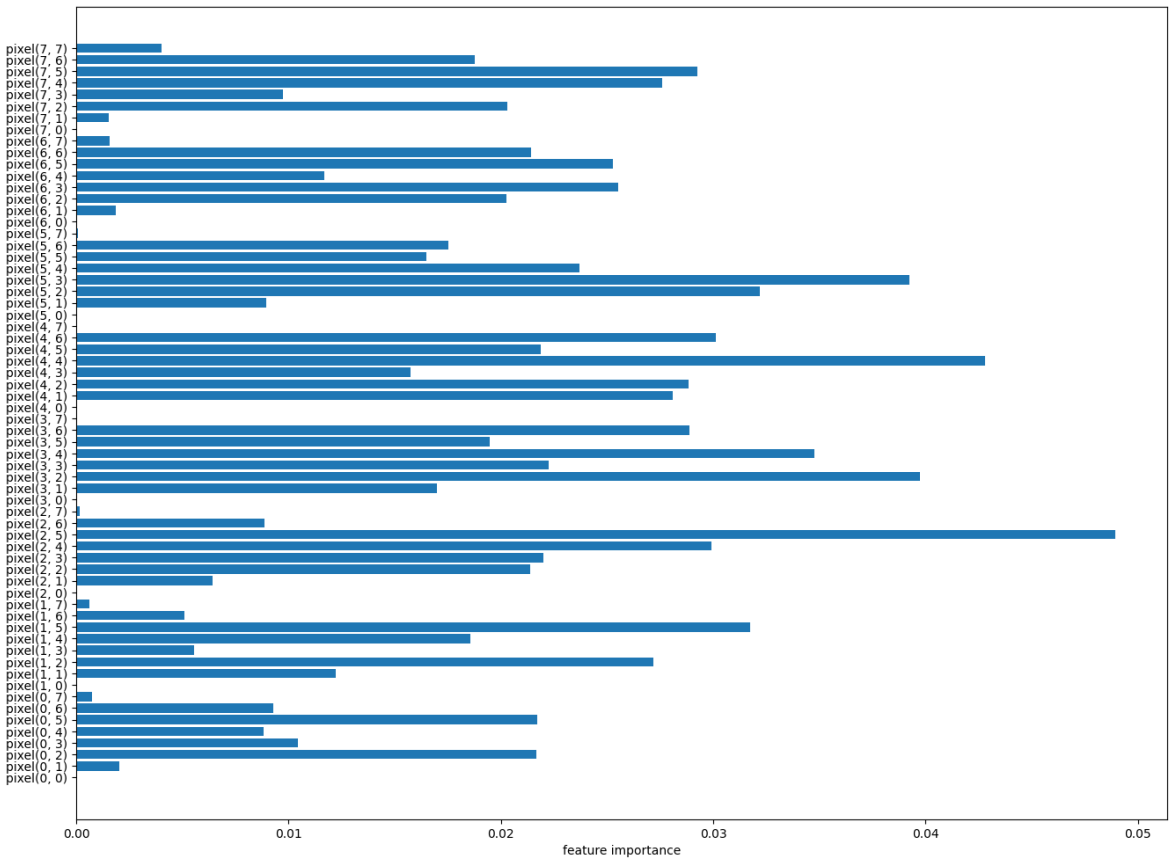
(詳見附檔 problem_3.py 或 GitHub 連結)

Result：

```
>> Random Forest algorithm
Best pred. n_estimators = 85, Acc = 97.77778%
```



Number of estimators 對於 Random Forest 準確率的影響如上方輸出結果所示



Problem 04：

Code：

(詳見附檔 problem_4.py 或 GitHub 連結)

Result：

```
>> KNN
For n_neighbors = 2:  training acc = 0.99183, testing acc = 0.98667
For n_neighbors = 3:  training acc = 0.98961, testing acc = 0.99111
For n_neighbors = 4:  training acc = 0.98961, testing acc = 0.98667
For n_neighbors = 5:  training acc = 0.99035, testing acc = 0.98667
For n_neighbors = 6:  training acc = 0.98738, testing acc = 0.98444
For n_neighbors = 7:  training acc = 0.98589, testing acc = 0.98667
For n_neighbors = 8:  training acc = 0.98589, testing acc = 0.98444
For n_neighbors = 9:  training acc = 0.98515, testing acc = 0.98667
For n_neighbors = 10:  training acc = 0.98218, testing acc = 0.98444
Best prediction model (n_neighbors = 3) Testing Acc. = 99.11111%

>> Logistic Regression
For C = 0.001:  training acc = 0.97327, testing acc = 0.95333
For C = 0.01:  training acc = 0.9948, testing acc = 0.96444
For C = 0.1:  training acc = 1.0, testing acc = 0.96222
For C = 1:  training acc = 1.0, testing acc = 0.96
For C = 10:  training acc = 1.0, testing acc = 0.95778
For C = 100:  training acc = 1.0, testing acc = 0.95556
Best prediction model (C = 0.01) Testing Acc. = 96.44444%

>> Linear Support Vector Classification
For C = 0.001:  training acc = 0.98367, testing acc = 0.95556
For C = 0.01:  training acc = 0.9948, testing acc = 0.95556
For C = 0.1:  training acc = 0.99555, testing acc = 0.95556
For C = 1:  training acc = 0.99703, testing acc = 0.95333
For C = 10:  training acc = 0.99852, testing acc = 0.94
For C = 100:  training acc = 0.99926, testing acc = 0.93778
Best prediction model (C = 0.001) Testing Acc. = 95.55556%

>> Guassian Naive Bayes
training acc = 0.85152, testing acc = 0.80667
Testing Acc. = 80.66667%
```

(a)(b)(c)在上方輸出中顯示

(d) 在此題中，KNN 演算法相較於其他演算法成果最佳

Problem 05：

Code：

(詳見附檔 problem_5.py 或 GitHub 連結)

Result：

```
>> Linear Regression
training acc = 0.54618, testing acc = 0.41419
Testing Acc. = 41.419%

>> Ridge Regression
For alpha = 0.01:  training acc = 0.54509, testing acc = 0.41202
For alpha = 0.1:  training acc = 0.54018, testing acc = 0.41153
For alpha = 1:  training acc = 0.45746, testing acc = 0.34675
For alpha = 10:  training acc = 0.17413, testing acc = 0.12509
For alpha = 100:  training acc = 0.02413, testing acc = 0.0142
Best prediction model (alpha = 0.01) Testing Acc. = 41.20158%

>> Lasso Regression
For alpha = 0.01:  training acc = 0.54471, testing acc = 0.41188
For alpha = 0.1:  training acc = 0.53504, testing acc = 0.41932
For alpha = 1:  training acc = 0.39059, testing acc = 0.30702
For alpha = 10:  training acc = 0.0, testing acc = -0.00352
For alpha = 100:  training acc = 0.0, testing acc = -0.00352
Best prediction model (alpha = 0.1) Testing Acc. = 41.93208%
```

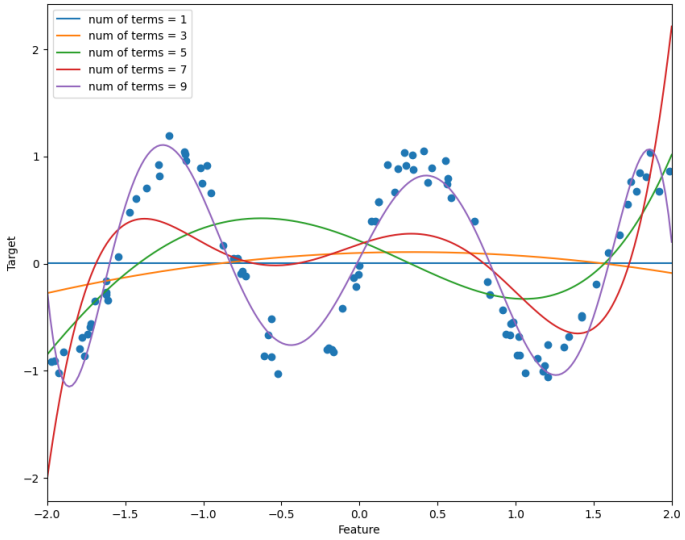
不論是 Ridge 或是 Lasso，限制參數(alpha) 基本上有越大精準度越差的趨勢。

Problem 06：

Code：

(詳見附檔 problem_6.py 或 GitHub 連結)

Result：

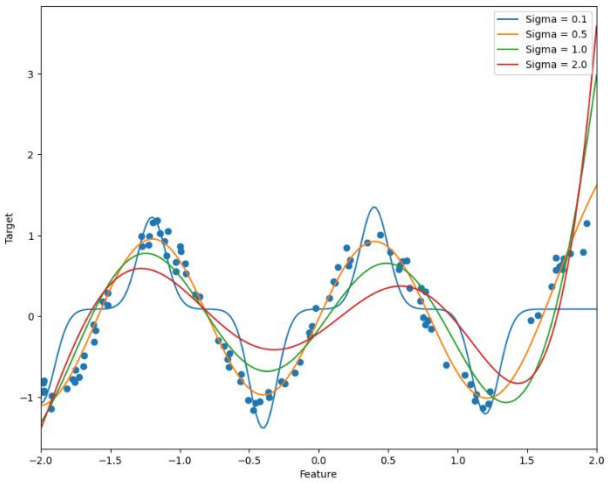


Problem 07：

Code：

(詳見附檔 problem_7.py 或 GitHub 連結)

Result：



Problem 08：

Code：

(詳見附檔 problem_8.py 或 GitHub 連結)

Result：

```
>> Logistic Regression
For C = 0.001:  training acc = 0.78125, testing acc = 0.72396
For C = 0.01:  training acc = 0.78819, testing acc = 0.72396
For C = 0.1:  training acc = 0.78646, testing acc = 0.73438
For C = 1:  training acc = 0.78472, testing acc = 0.72917
For C = 10:  training acc = 0.78472, testing acc = 0.72917
For C = 100:  training acc = 0.78299, testing acc = 0.72917
Best prediction model (C = 0.1) Testing Acc. = 73.4375%

>> Linear Support Vector Classification
For C = 0.001:  training acc = 0.72049, testing acc = 0.6875
For C = 0.01:  training acc = 0.73611, testing acc = 0.6875
For C = 0.1:  training acc = 0.79167, testing acc = 0.71354
For C = 1:  training acc = 0.78472, testing acc = 0.73958
For C = 10:  training acc = 0.78125, testing acc = 0.73438
For C = 100:  training acc = 0.78125, testing acc = 0.73438
Best prediction model (C = 1) Testing Acc. = 73.95833%

>> Random Forest algorithm
For n_estimators = 2:  training acc = 0.88368, testing acc = 0.70312
For n_estimators = 4:  training acc = 0.9375, testing acc = 0.69271
For n_estimators = 6:  training acc = 0.95833, testing acc = 0.72917
For n_estimators = 8:  training acc = 0.96701, testing acc = 0.73958
For n_estimators = 10:  training acc = 0.9809, testing acc = 0.75
Best prediction model (n_estimators = 10) Testing Acc. = 75.0%
```

(a)(b)(c)顯示於上方輸出結果。(a)(b)較難以看出規律；(c)可見隨著 estimators 數量上升，精準度也又提升。