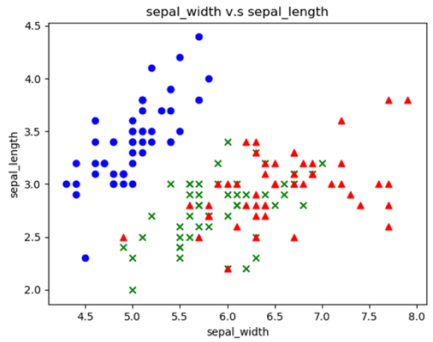
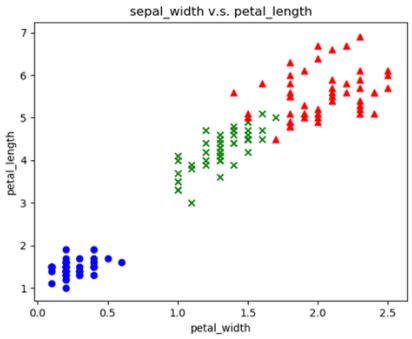
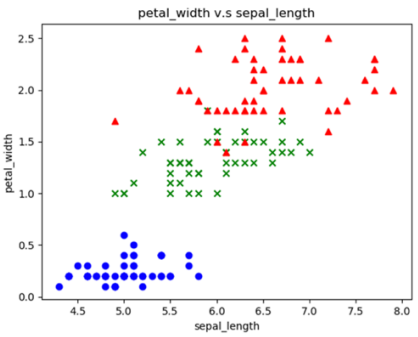
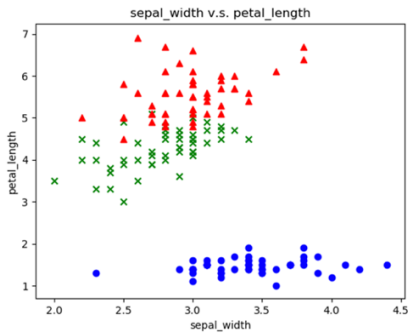
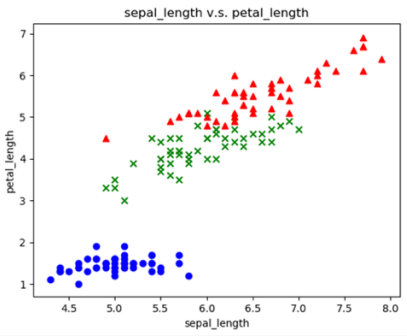
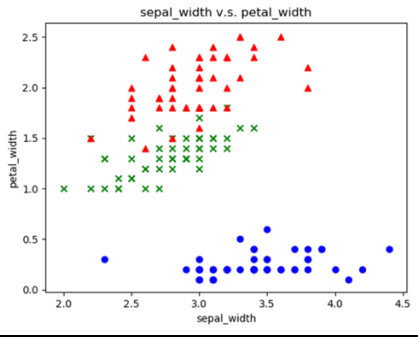
NYCU Machine Learning HW1

312605001

機器人碩士 歐庭維

1. Scatter plot

我以 python 的 matplot 套件繪製出六張 scatter plot 來對資料分佈情況進行分析，結果如下六圖所示，圖表中三種 label 的意義分別為：
label 1(Setosa) = blue circle / label 2(Versicolor) = green x / label 3(Virginica) = red triangle

Sepal length v.s. sepal width	Petal width v.s. petal length
	
Petal width v.s. sepal length	**Sepal width v.s. petal length**
	
Sepal length v.s. petal length	**Sepal width v.s. petal width**
	

2. Classification rate
   - K = 1
     由 CR 可以得知 14 約為 96.53%，是所有數據組中最好的

K = 1

Combinations	Classification Rate
1. Sepal Length only	59.72
2. Sepal Width only	45.14
3. Petal Length only	94.44
4. Petal Width only	94.44
5. Sepal Length + Sepal Width	72.22
6. Sepal Length + Petal Length	93.06
7. Sepal Length + Petal Width	88.89
8. Sepal Width + Petal Length	93.06
9. Sepal Width + Petal Width	93.75
10. petal Length + Petal Width	95.14
11. Sepal Length + Sepal Width + Petal Length	92.36
12. Sepal Length + Sepal Width + Petal Width	93.06
13. Sepal Length + Petal Length + Petal Width	94.44
14. sepal Width + Petal Length + Petal Width	96.53
15. Sepal Length + Sepal Width + Petal Length + Petal Width	94.44

   - K = 3
     由 CR 可以得知 14 約為 97.22%，是所有數據組中最好的

K = 3

Combinations	Classification Rate
1. Sepal Length only	62.50
2. Sepal Width only	54.86
3. Petal Length only	93.75
4. Petal Width only	95.14
5. Sepal Length + Sepal Width	78.47
6. Sepal Length + Petal Length	92.36
7. Sepal Length + Petal Width	93.75
8. Sepal Width + Petal Length	91.67
9. Sepal Width + Petal Width	95.14
10. petal Length + Petal Width	95.14
11. Sepal Length + Sepal Width + Petal Length	92.36
12. Sepal Length + Sepal Width + Petal Width	90.97
13. Sepal Length + Petal Length + Petal Width	95.14
14. sepal Width + Petal Length + Petal Width	97.22
15. Sepal Length + Sepal Width + Petal Length + Petal Width	93.75

   - CR 與 Scatter Plot 的關係
     - 這次作業的散佈圖使用兩組特徵作為組合進行繪製，共有六張，使用 k-nn 算法計算出 CR 後，分別對應到 5~10 項。而可以由這些分佈圖大致推測出由 Petal width v.s. petal length 這組數據來看，每個群的數據分佈都較另外五張集中，因此我推測其在 CR 的表現來說會是六組中較好的，由上個部分的圖可以驗證（k=1 : 95.14% / k=3 : 95.14%）。

     - 由 sepal width & sepal length 的 scatter plot 可以得知 Versicolor 與 Virginica 的 Distribution 重疊區域相當多，可以預見由於不明顯的邊界，會使 knn 在找尋最近點時會有較大的機率判錯，若採用這個關係作為 classifier 則會有較低的 CR，而結果顯示 CR 約為 72.22%(k = 1)以及 78.47%（k = 3）。

     - 當以單一特徵組合表現來看（1 ~ 4），3&4 的表現基本都維持在 90%以上，但即使調整 k 值，其表現也大致持平，其性能上限約為 95%左右，我認為這是因為單一個特徵會無法同時考慮到所有 label 的複雜性和多變性，當特徵不明顯時（1&2），表現就會非常差，下圖為我自行嘗試調整當 k=9 時的 CR 表現，4 號數據在 1~4 中的表現最好：

K = 9

Combinations	Classification Rate
1. Sepal Length only	64.58
2. Sepal Width only	48.61
3. Petal Length only	94.44
4. Petal Width only	95.83
5. Sepal Length + Sepal Width	79.86
6. Sepal Length + Petal Length	92.36
7. Sepal Length + Petal Width	94.44
8. Sepal Width + Petal Length	94.44
9. Sepal Width + Petal Width	94.44
10. petal Length + Petal Width	96.53
11. Sepal Length + Sepal Width + Petal Length	90.97
12. Sepal Length + Sepal Width + Petal Width	94.44
13. Sepal Length + Petal Length + Petal Width	94.44
14. sepal Width + Petal Length + Petal Width	95.83
15. Sepal Length + Sepal Width + Petal Length + Petal Width	95.83

- 其餘五張 scatter plot 的重疊區域與 sepal width & sepal length 的 scatter plot 相比較為分散，但是 Versicolor 與 Virginica 兩個群的 boundary 較為曖昧，因此可以想像要以 knn 作為 classifier，其最大準確度一定沒辦法太高，而我後續也嘗試調整了其他的 K 值，得到的最高成功率如下所示，也同樣是 97.22%

K = 11

Combinations	Classification Rate
1. Sepal Length only	71.53
2. Sepal Width only	48.61
3. Petal Length only	94.44
4. Petal Width only	95.83
5. Sepal Length + Sepal Width	79.17
6. Sepal Length + Petal Length	92.36
7. Sepal Length + Petal Width	93.06
8. Sepal Width + Petal Length	94.44
9. Sepal Width + Petal Width	95.14
10. petal Length + Petal Width	97.22
11. Sepal Length + Sepal Width + Petal Length	92.36
12. Sepal Length + Sepal Width + Petal Width	93.75
13. Sepal Length + Petal Length + Petal Width	95.14
14. sepal Width + Petal Length + Petal Width	95.14
15. Sepal Length + Sepal Width + Petal Length + Petal W	94.44

- 以表現普遍較好的第 14 號的組合為例，可以發現當 k>5 時，K-NN 在 CR 的表現會開始下降，我推測這是由於當 k 值越大，會使整個模型過於模糊，失去局部敏感性，使其不足以應付數據的複雜性，導致性能下降。

- 當參考的點數越多（k 越高），代表參照圓的半徑越大，這會使投票驗證的點數變多，使其更加的保險，但過大的 k 會造成在處理時的計算複雜度變高（耗時間），以及決策邊界變得更加的模糊，進而造成性能下降，因此需要針對不同的數據集選則適合的 k 值。

- 考慮到我一開始忽略了如果遇到超過 k 個相同數據時，該如何處理，我將程式碼由原本的

```
def knn_classifier(train_data, train_labels, test_data, k):

    predictions = []



    for test_point in test_data:
```

```
        distances = [np.sqrt(np.sum((train_point - test_point) ** 2)) for train_point in
train_data]



        k_indices = np.argsort(distances)[:k]



        k_nearest_labels = [train_labels[i] for i in k_indices]

        # 原本的會只取 k 個進行投票，當數據集中出現多個相同距離的鄰居時，
            會降低準確性

        prediction = np.bincount(k_nearest_labels).argmax()

        predictions.append(prediction)

    return predictions
```

改成遇到超過 k 個距離相同的鄰居時，使用所有相同的鄰居進行投票：

```
def knn_classifier(train_data, train_labels, test_data, k):

    predictions = []



    for test_point in test_data:

        distances = [np.sqrt(np.sum((train_point - test_point) ** 2)) for train_point in
train_data]



        k_indices = np.argsort(distances)

        k_nearest_labels = [train_labels[i] for i in k_indices[:k]]



        # 統計相同距離鄰居的數量
```

```
        same_distance_count = k_nearest_labels.count(k_nearest_labels[0])


        # 如果相同距離鄰居的數量大於 k，則使用所有相同距離的鄰居進行投票
        if same_distance_count > k:
            prediction = np.bincount(k_nearest_labels[:same_distance_count]).argmax()
        else:
            prediction = np.bincount(k_nearest_labels).argmax()


        predictions.append(prediction)


    return predictions
```

3. 心得

這次的作業對我而言是一次非常寶貴的學習經驗，讓我深入了解了 K-NN 演算法的整體架構、運作機制以及其獨特的特點。尤其是使用較低維度的 Iris 數據集時，我發現 K-NN 演算法特別適用。這主要是因為 Iris 數據集中的特徵之間的距離不僅計算簡單，而且具有高度的解釋性和實用性。此外，透過親手實作了 K-NN 分類器，讓我對算法的實際運作過程有了更全面和深入的認識。這個過程讓我更明確地看到了每一步驟是如何影響最終結果的，並且也讓我更加明白了在實際應用中會遇到的問題，以及該如何調整不同的參數以獲得最佳效果。