ML Homework 1 Perceptron

系級: 資工四
姓名: 方琬淳
學號: 406410039

題目

1. Use line equation $y=mx+b$ with particular parameters m and b to generate 30 2D data samples. 15 samples are in the right of the line are marked as positive samples and the others are in the left as negative samples. No samples on the line.

2. Implement Perceptron Learning Algorithm with your own initial w. Discuss if your PLA halts or how many iterations it halts. Generate the data samples three times and calculate the average number of iterations when PLA halts.

3. In Problem 1, generating 1000 positive samples and 1000 negative samples. Implement Pocket Algorithm and compare the execution time to PLA on the same dataset.

1. 使用具有特定參數m和b的線性方程式 y = mx + b 來生成30個2D數據樣本。將右側的15個數據樣本標記為正，左側的15個數據樣本標記為負。並且沒有樣本在線上。

2. 用您自己的初始w實現Perceptron Learning Algorithm。 討論PLA是否停止或它暫停了多少次迭代。生成三次數據樣本併計算平均值PLA暫停時的迭代次數。

3. 在問題1中，生成1000個正樣本和1000個負樣本。 利用Pocket Algorithm，並將執行時間與同一數據集上的PLA進行比較。

Execution description

作業環境

- 作業系統: Windows 10

- Requirments

```
python==3.6.5
numpy==1.14.3
matplotlib==2.2.2
```

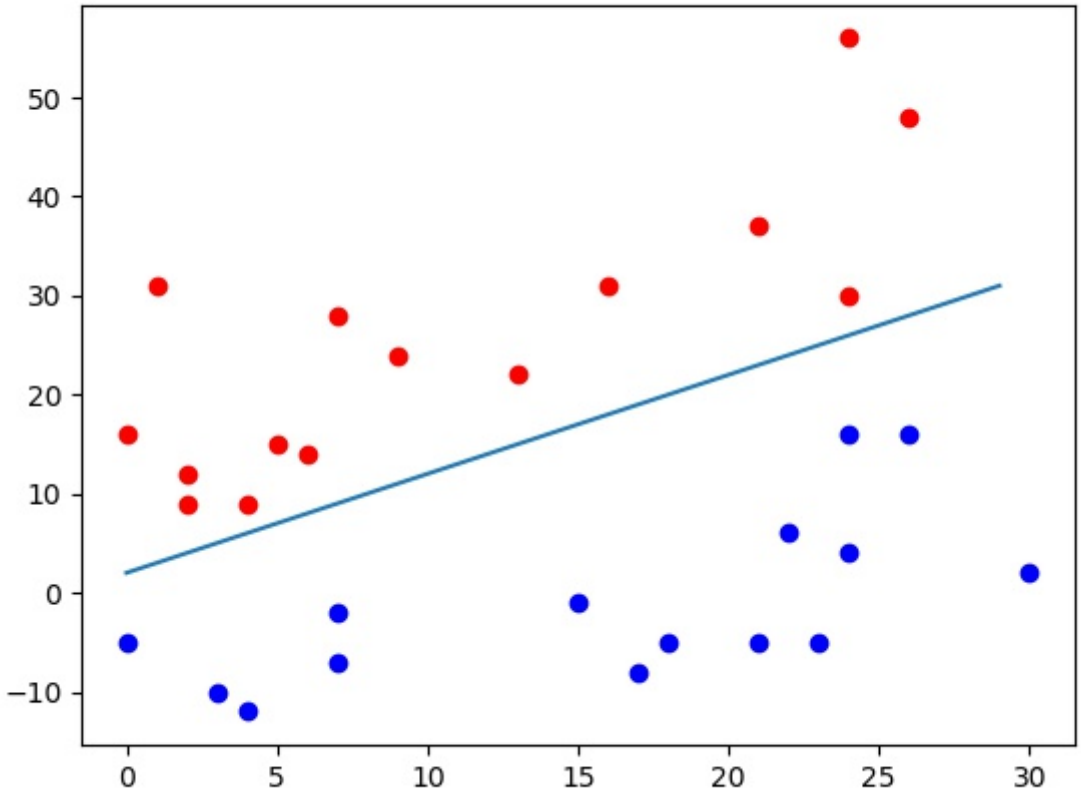
執行方法

- 在作業資料夾中打開命令提示字元
- 輸入以下指令即可執行

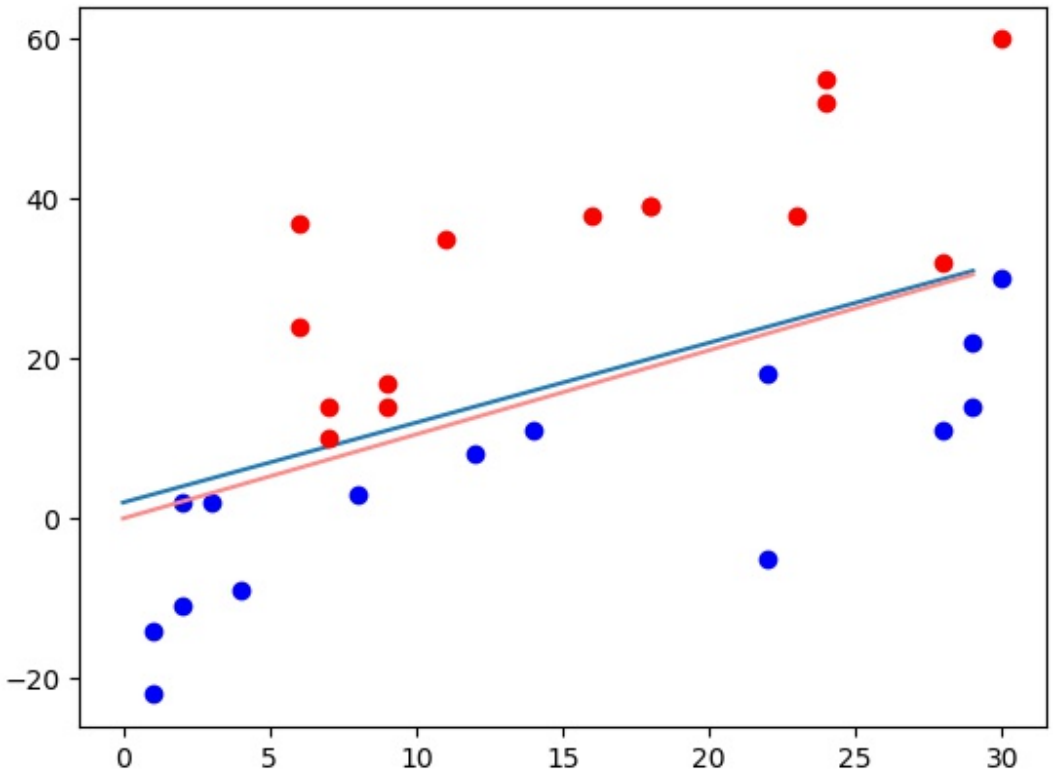
```
python hw1.py
```

Experimental results

1. 畫出線性方程 y = mx + b (m=1, b=2)，以及30個標記點 (藍色為正，紅色為負)

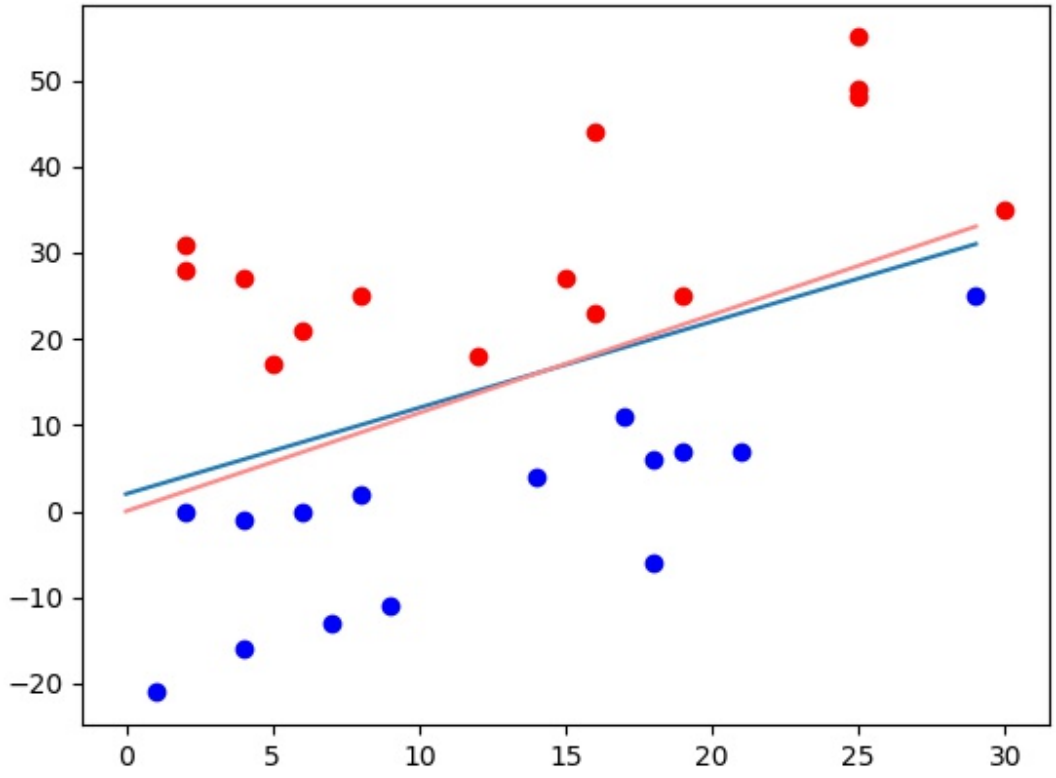


2. 自訂初始 w([1,1])，實作PLA，並重複三次計算每次執行PLA的迭代次數以及三次平均次數
    - 第一次: 213次



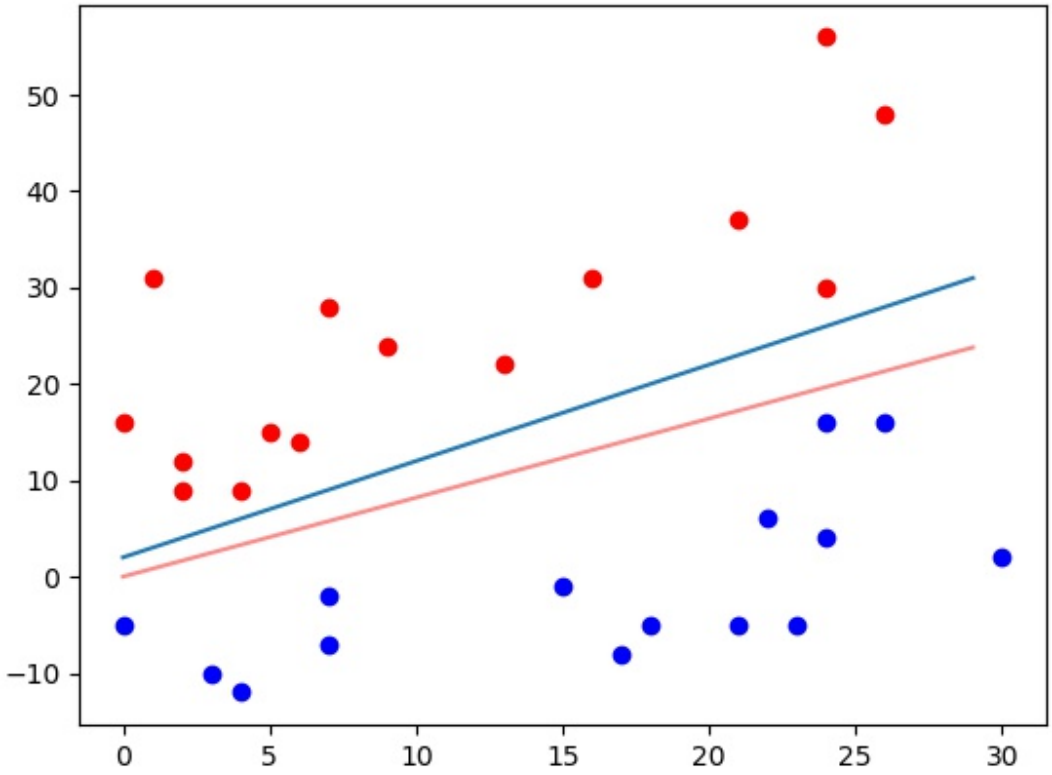
> 藍線為 y = mx + b (m=1, b=2)
> 紅線為PLA最終答案

- 第二次: 107次



> 藍線為 y = mx + b (m=1, b=2)
> 紅線為PLA最終答案

- 第三次: 70次



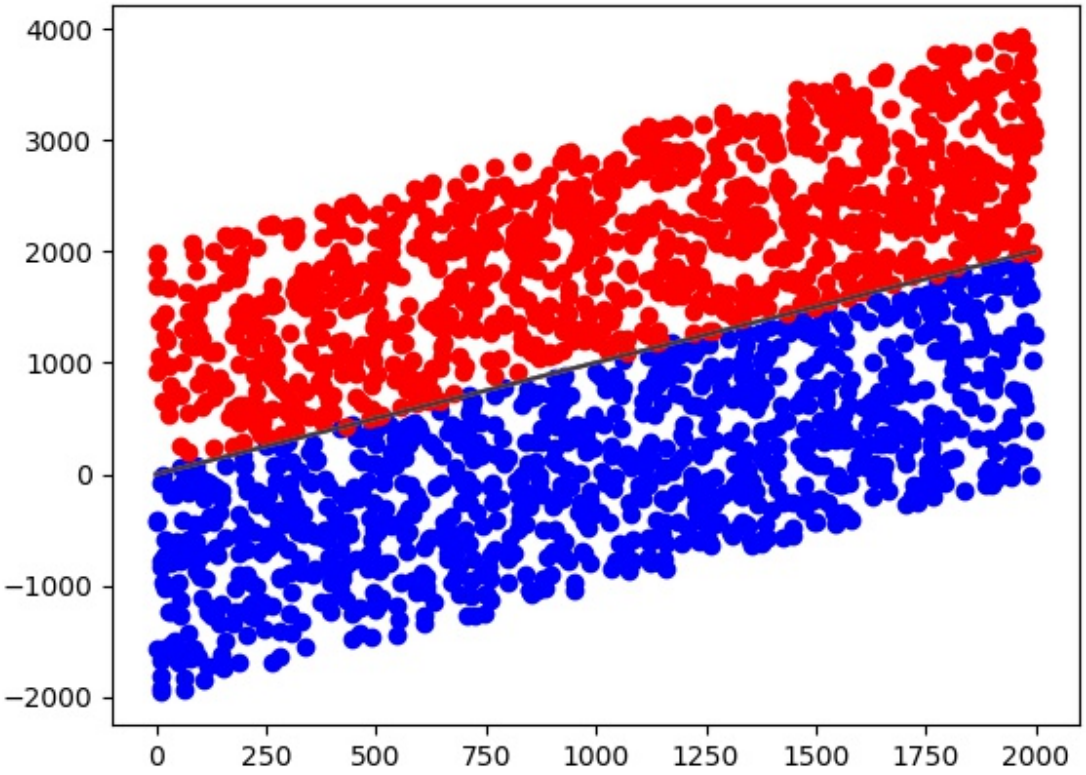
> 藍線為 y = mx + b (m=1, b=2)
> 紅線為PLA最終答案

- 迭代平均次數: 130次

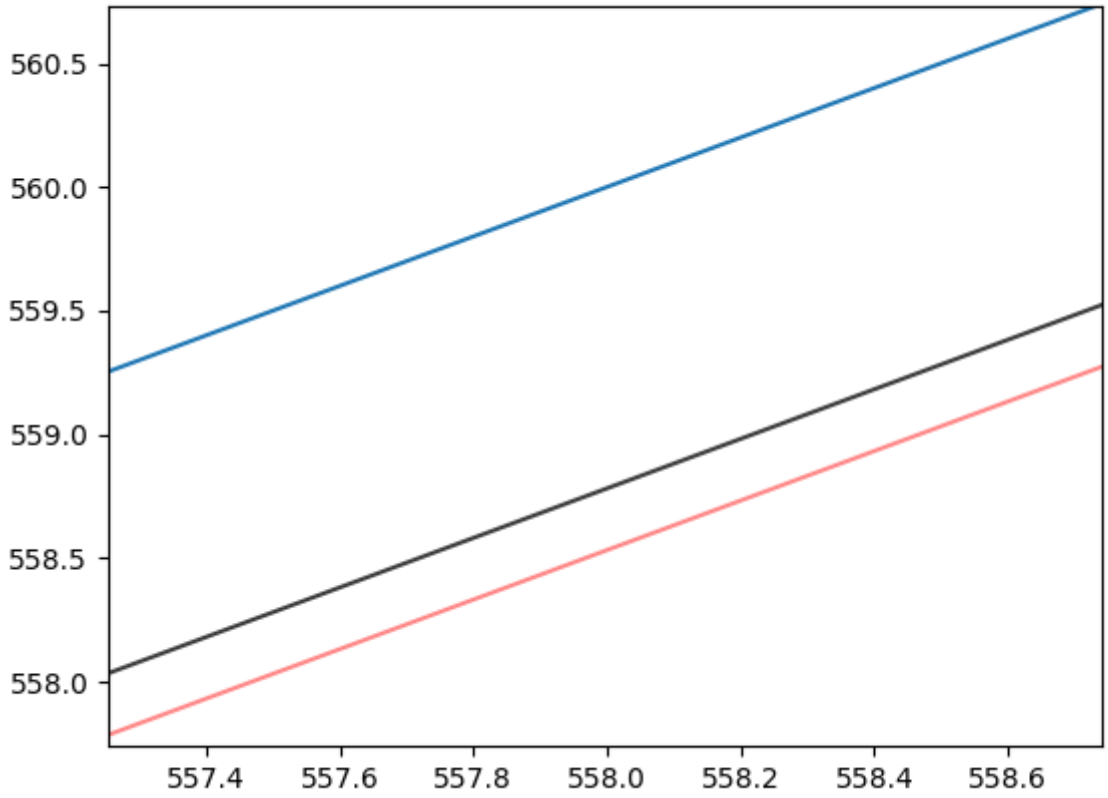
```
[redacted]ML\hw1>python hw1.py
1: 213
2: 107
3: 70
average iteration: 130.00
Run PLA Alogrithm in 0.0625 seconds
Run POCKET Alogrithm in 1.3750 seconds
```

3. 生成1000個正樣本和1000個負樣本。 自訂初始 w([1,1])，實作Pocket Algorithm，並將執行時間與同一數據集上的PLA進行比較。

    - 結果圖片



因為看不清楚有三條線所以我們把線的部分放大



> 藍線為 y = mx + b (m=1, b=2)
> 紅線為PLA最終答案
> 灰線為Pocket Algorithm答案

- 執行時間

```
[redacted]ML\hw1>python hw1.py
1: 213
2: 107
3: 70
average iteration: 130.00
Run PLA Alogrithm in 0.0625 seconds
Run POCKET Alogrithm in 1.3750 seconds
```

  - PLA: 0.0625 seconds
  - Pocket Algorithm: 1.3750 seconds

Conclusion

What is Perceptron Learning Algorithm?

- 用在可以完全二分的問題中，經由不斷的修正錯誤，求得一個 完美 的分類器。

- 如何判斷錯誤?
    - 將w_t 與 $x_{n(t)}$做內積，看是不是我們要的正負號。

$$sign(w_t^T x_{n(t)}) \neq y_{n(t)}$$

- 如何修正錯誤?
    - 根據正負耗去調整w，如果是正的，那就把w轉的靠近x一點，如果是負的，就把w轉的遠離x一點。

$$w_{t+1} \leftarrow w_t + y_{n(t)} x_{n(t)}$$

- 不斷重複上述兩步驟直到沒有錯誤

What is Pocket Algorithm?

- 為PLA的變形，目的是找出一個相對好的分類器
- 在每次進行修正的時候都把新的w_t跟舊的w做比較，如果w_t的誤差值較小，則更新目前口袋中的w值，經過不斷的迭代，w中的值必定可以找到當下最好的分類器

Discussion

- 為什麼PLA有時候會無法停止？
    - 因為資料樣本不是線性可分

- 為什麼使用Pocket Algorithm的時候，會出現沒被準確分類的樣本？
    - 因為Pocket Algorithm的目的是找出相對好的w，為了避免遇到不是線性可分的資料，有限制迭代的上限次數，所以有時候會出現沒被準確分類的樣本。

- 為什麼Pocket Algorithm常常比PLA慢?
    - 因為Pocket Algorithm每次調整w時，都會重新計算一次w的正確率，所以通常會比較久