Machine Learning HW2

R04942070 吳冠融

1. Logistic regression function.

```
for i in range(200000):
    # forward
    y_pred_rows = special.expit(np.dot(x_data_rows, w))  # shape = (x_row_num, 1)
    y_diff_rows = (y_data_rows - y_pred_rows)  # shape = (x_row_num, 1)
    w_grad = -np.dot(x_data_rows_trans, y_diff_rows)

    # write model
    if total_epoch % 10000 == 0:
        train_error_count = np.sum(np.abs(to_bool(y_pred_rows) - y_data_rows))
        train_accuracy = float(x_row_num - train_error_count) / x_row_num

        model_file_info = "epo" + str(total_epoch) + "_acc" + str(train_accuracy)[:6]
        print(model_file_info)
        np.savez(MODEL_FILE_PATH, w=w, vw=vw, lr=lr, total_epoch=total_epoch, x_train_means=x_train_means, x_train_stds=x_train_stds)

    # update weights. using momentum
    vw = lr * w_grad + gamma * vw
    w = w - vw

    total_epoch += 1
```

Note: I have appended a '1' in each data row, so I don't need a constant b here.

2. Describe your another method, and which one is best.

Method 2: Deep Neural Network

- 在 Method 2 裡，我實作了很多功能：
  - 資料預處理
    - Normalization
    - Selecting Data：
      - 我嘗試了兩種版本：一種使用 data 的全部 column ，另一種是只用與 output 相關性最高的 32 個 column（相關性判斷依據是 logistic regression 訓練出來的 model 的 weight）。
  - 任意形狀的 Neural Network
    - 初期我是把層數寫死。到後期會了方便測試，我用一個 list（layers = [x_col_num, 30, 30, 30, 30, 1]）來設定 NN 的結構，要變動結構時只要改寫這個 list 再重新 train 即可。
  - Dropout
    - 在 train 時，Hidden layer 每一層都會都會根據我設定的 dropout rate 而使部分 node 失去作用，以避免 overfitting。
    - dropout rate 我嘗試過 0.5 與 0.75 兩個數值，結果是：0.75 的 training accuracy 上升得較快（收斂速度快），而當兩者的 training accuracy 一樣時，testing accuracy 通常是 0.5 的略高一點。

- 因為我有充足的時間 train，所以後來我都採用 0.5。
- 在上傳的 code 中，為了在 10 分鐘內達到最好的訓練效果，我將 dropout rate 設為 0.75。

Experements on DNN structures

3 layers, dropout rate = 0.5, using 32 columns of data

Number of Neurons	30 * 3	40 * 3	50 * 3
Train Accuracy	0.9317	0.9653	0.9725
Test Accuracy (Public)	0.93	0.95	0.93333

可以看出在同為三層的情況下，neurons = 40*3 效果最好，到 50*3 就 overfit 了。

4 layers, dropout rate = 0.5, using all columns of data

Number of Neurons	30 * 4	40 * 4
Train Accuracy	0.9780	0.9810
Test Accuracy (Public)	0.95333	0.93667

在同為四層的情況下，30 * 4 剛好能達到最高的 accuracy，而 40*4 就 overfit 了。

Results

Method	Logistic Regression	Final Submission 1 DNN using 32 columns neurons = 40 * 3	Final Submission 2 DNN using all columns neurons = 30 * 4
Public Score	0.92667	0.95000	0.95333
Private Score	0.91667	0.94000	0.92333
Average Score	0.92167	0.945	0.93833

- 毫無疑問的，DNN 的準確度比 Logistic Regression 高。
- 令我意外的是 40*3 的 test accuracy 比 30*4 的高，可能是因為前者我使用的是 32 column 的資料，後者是用全部的資料，所以後者稍微 overfit。（也可能只是 test data 太少筆的關係）
- Final Result:
  - Scored 0.94 and ranked 27/276 in private leaderboard.

27	↑11	r04942070_RogerW_大家名字都好有創意害我好慚愧QQ	0.94000	35	Wed, 26 Oct 2016 20:02:26 (-2.7d)