HW11

612415013 蕭宥羽

1. Execute your code

這邊是助教提供的讀取轉換成 python 列表

```
class HandWritten_Digit_Dataset():
    def __init__(self, mat_file:Path) -> None:
        self.trainX:np.ndarray = None
        self.trainY:np.ndarray = None
        self.testX:np.ndarray = None
        self.testY:np.ndarray = None
        self._read_handwritten_digit(mat_file=mat_file)
```

```
def _read_handwritten_digit(self, mat_file:Path, to_float:bool=True) -> None:
    mat_data = scipy.io.loadmat(mat_file)
    self.trainX = mat_data['train']
    self.trainY = mat_data['train_label'].reshape(-1).astype(np.int32)
    self.testX = mat_data['test']
    self.testY = mat_data['test_label'].reshape(-1).astype(np.int32)
    if to_float:
        self.trainX = self.trainX.astype(np.float32)
        self.testX = self.testX.astype(np.float32)
def get_digits_XY(self, digits:list) -> tuple:
    train_mask = np.isin(self.trainY, digits)
    test_mask = np.isin(self.testY, digits)
    return self.trainX[train_mask], self.trainY[train_mask], self.testX[test_mask], self.testY[test_mask]
```

初始化類別，讀取手寫數字資料集

定義了 HandwrittenDigits 類別，來處理手寫數字資料集， 這個類別初始化時會讀取提供的 .mat 文件，並將其內容存儲在類別變量中，根據給定的數字列表過濾數據集，返回訓練和測試數據及其標籤

定義了 filter_data ，接收一個數字列表，過濾數據集中相應的數字，並將標籤轉換為 1 和-1，以適應二元分類。最後，它會返回訓練和測試數據及其標籤。

```
digits = [6, 9]
trainX, trainY, testX, testY = handwritten_digit_dataset.get_digits_XY(digits)

trainY = np.where(trainY == 6, 1, -1)
testY = np.where(testY == 6, 1, -1)

  trainX_list = trainX.tolist()
  trainY_list = trainY.tolist()
  testX_list = testX.tolist()
  testY_list = testY.tolist()
```

在主程式中，我們選擇 6 和 9 作為要分類的數字，並通過 filter_data 方法將標籤轉換為 1 和-1，以適應二元分類任務。接著，我們將 numpy 數組轉換為 python 列表，以便後續操作。

```
best_accuracy = -1
best_params = {}

for C in C_range:
    for gamma in gamma_range:
        param = f'-v 5 -c {C} -g {gamma} -q'
        cv_accuracy = svm_train(trainY_list, trainX_list, param)
        if cv_accuracy > best_accuracy:
            best_accuracy = cv_accuracy
            best_params['C'] = C
            best_params['gamma'] = gamma
```

```
best_C = best_params['C']
best_gamma = best_params['gamma']
print(f"Best C: {best_C}, Best gamma: {best_gamma}")
```

```
final_param = f'-c {best_C} -g {best_gamma} -q'
model = svm_train(trainY_list, trainX_list, final_param)

p_label_train, p_acc_train, p_val_train = svm_predict(trainY_list, trainX_list, model)
train_accuracy = p_acc_train[0]

p_label_test, p_acc_test, p_val_test = svm_predict(testY_list, testX_list, model)
test_accuracy = p_acc_test[0]

print(f"Training Accuracy: {train_accuracy:.2f}%")
print(f"Testing Accuracy: {test_accuracy:.2f}%")
```

```
min_vals = np.min(trainX, axis=0)
max_vals = np.max(trainX, axis=0)
trainX_scaled = (trainX - min_vals) / (max_vals - min_vals)
testX_scaled = (testX - min_vals) / (max_vals - min_vals)

trainX_scaled_list = trainX_scaled.tolist()
testX_scaled_list = testX_scaled.tolist()

model_scaled = svm_train(trainY_list, trainX_scaled_list, final_param)

p_label_train_scaled, p_acc_train_scaled, p_val_train_scaled = svm_predict(trainY_list, trainX_scaled_list, model_scaled)
train_accuracy_scaled = p_acc_train_scaled[0]

p_label_test_scaled, p_acc_test_scaled, p_val_test_scaled = svm_predict(testY_list, testX_scaled_list, model_scaled)
test_accuracy_scaled = p_acc_test_scaled[0]

print(f"Training Accuracy with scaling: {train_accuracy_scaled:.2f}%")
print(f"Testing Accuracy with scaling: {test_accuracy_scaled:.2f}%")
```

我們使用 5-fold 交叉驗證和網格搜尋來找到最佳的參數組合，並紀錄最佳的準確率以及其對應的參數，並輸出最佳的兩個值，使用最佳參數訓練最終模型， 訓練最終模型，並在訓練和測試數據上評估其準確率。對數據進行縮放，將數值範圍轉換到 0 到 1 之間，並重新訓練評估準確率，使用縮放後的數據重新訓練模型，將結果進行比較。

2. Experimental results

```
Cross Validation Accuracy = 60.4651%
```

進行 5-fold 交叉驗證時，得到的準確率是 60.4651%

```
Best C: 0.03125, Best gamma: 3.0517578125e-05
```

最佳參數組合

```
Accuracy = 60.4651% (104/172) (classification)
Accuracy = 38.2716% (62/162) (classification)
Training Accuracy: 60.47%
Testing Accuracy: 38.27%
```

訓練準確率：60.47%（172 個訓練樣本中，有 104 個分類正確）。

測試準確率：38.27%（162 個測試樣本中，有 62 個分類正確）。

```
Accuracy = 39.5349% (68/172) (classification)
Accuracy = 61.7284% (100/162) (classification)
Training Accuracy with scaling: 39.53%
Testing Accuracy with scaling: 61.73%
```

訓練準確率（縮放後）：39.53%（172 個訓練樣本中，有 68 個分類正確）。

測試準確率（縮放後）：61.73% （162 個測試樣本中，有 100 個分類正確）。

3. Conclusion

- ✓ 訓練階段的準確率顯示，即使嘗試了不同 C 和 gamma 值進行交叉驗證，得到的準確率都是相同的 60.4651%，這表明所有組合都沒有改變交叉驗證的準確率。
- ✓ 最終找到的最佳參數組合是 C = 0.03125 和 gamma = 3.0517578125e-05。這是基於交叉驗證準確率最高的參數組合
- ✓ 在模型上的訓練資料有一定的準確度但不高，且在測試集上表現很差，可能存在過擬合或模型在測試數據上泛化能力不足。
- ✓ 縮放後的準確率顯示，訓練的準確率下降了，但在測試集卻提升了，說明數據縮放提高了模型在測試數據上的表現，改善了模型的泛化能力

```
/home/rvl/ccu/ML/hw11/assigment_11/123.py:77: RuntimeWarning: invalid value encountered in divide
  trainX_scaled = (trainX - min_vals) / (max_vals - min_vals)
/home/rvl/ccu/ML/hw11/assigment_11/123.py:78: RuntimeWarning: divide by zero encountered in divide
  testX_scaled = (testX - min_vals) / (max_vals - min_vals)
/home/rvl/ccu/ML/hw11/assigment_11/123.py:78: RuntimeWarning: invalid value encountered in divide
  testX_scaled = (testX - min_vals) / (max_vals - min_vals)
```

進行數據縮放時，出現除以零或無法除的錯誤。有可能是因為某些特徵最小值和最大值相等，導致分母為 0。

4. Dicussion

這些結果表明數據縮放對模型性能的提升有明顯的幫助。經過縮放處理後，模型在測試數據上的準確率從 38.27%提升至 61.73%，顯示出顯著的性能改進。此外，未縮放數據時模型在訓練資料上有過擬合現象，而經過數據縮放

後，過擬合情況有所減少。這強調了數據預處理在機器學習模型訓練中的重要性。