Undergraduate AI Capstone NYCU Spr2023 Programming Assignment #1

邵筱庭 109550119

Datasets

1. Dataset1 (A public image dataset): Color Classification
   I. 目標：分辨出物體顏色 (多項分類問題)。
   II. 組成架構：包含黃色、黑色、白色、綠色、紅色、橙色、藍色和紫色共八種顏色的物品，各種顏色約有 7 到 20 張圖片，屬於數量小的資料集。
   III. 資料集大小：共 107 筆資料，拆成訓練集 80 張、測試集 27 張圖片 (預設)。
   IV. 資料示意：  
   V. 資料來源：kaggle
2. Dataset2 (A public non-image dataset): Mobile Price Classification
   I. 目標：預測手機價位，標籤由低至高為 0 ~ 3 (多項分類問題)。
   II. 組成架構：為非圖片資料集，在 csv 檔中包含 battery_power、blue、clock_speed 等共 20 個手機相關特徵可用於判斷，關於各個特徵的詳細資料可至 feature_description.txt 檔案中察看。此外，四種標籤的數量頗為平均，因此不需特別處理。
   III. 資料集大小：共 2000 筆資料，拆成訓練集 1400 筆、測試集 600 筆 (預設)
   IV. 資料示意：

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height
0	1224	0	0.5	0	16	1	6	0.4	109	6	...	747
1	981	1	1.9	1	0	0	2	0.1	136	3	...	75

   V. 資料來源：kaggle
3. Dataset3 (A self-made dataset): my cat or not my cat
   I. 目標：分辨圖片中的貓咪是不是我家養的貓咪閃光 (後面以貓咪名稱：閃光代稱) (二分類問題)。
   II. 組成架構：包含閃光各種動作、角度和光線的照片，以及其他各種花色的貓咪的照片。這些照片中背景也沒有限制，包含乾淨與雜亂的背景。標籤部分正樣本為閃光(1)，負樣本為其他貓咪(0)。
   III. 資料集大小：閃光照片有 121 張，其他貓咪照片有 146 張。訓練集和測試集的比例設為 7:3 (預設)。
   IV. 資料收集 (與 陳存佩 109550119 共同收集此資料集)
      i. 發想與討論：製作此資料集的動機是因為這學期我和存佩同學都有修另一門物連網的課程，我想若能將辨別自家寵物的功能與物連網結合在一起，那應該很適合有養貓的家庭，尤其是多貓家庭。此外，我們當初有討論是否該分類程式閃光與不是閃光(包含

其他貓和非貓)，然而後來認為此分辨主題應該建立於”偵測貓咪位置”的模型後較為合適，先偵測再判別而非直接判別畫面是否為目標貓咪在設計邏輯上應該較為合理。由於本次作業主題並非偵測，因此才選擇只做辨識部分。

ii. 圖片來源：自己和家人用手機拍攝的照片 (閃光和少量其他貓)，其餘來源有 Google 圖片搜尋結果、Dcard 寵物版主人拍攝的日常照、Facebook 貓咪相關粉專和社團，檔案格式皆為 jpg 檔。大部分其他貓咪的資料為取自論壇是由於想讓圖片更接近正樣本的照片。

V. 資料示意：

i. 閃光： 

ii. 其他貓： 

Classifiers

1. Random Forest
2. Decision Tree

I. 決策樹為將數據分成葉子節點(類別)與決策節點(特徵)，將數據切分成多個小子集。隨機森林則是集結多個決策樹的決策結果，並加入隨機抽樣的概念，減少過度擬合的風險。

II. 選擇兩種樹型模型是為了比較兩者差別，也是為能在非圖片辨識，提供較高的可解釋性。

III. 比較

	決策樹	隨機森林
架構	單一樹型模型	多棵決策樹的集成
過度擬合	易	較不易
效率	快	較慢
準確率	較差	較好

3. SVM

I. 使用 kernel function，將資料應設到高為空間，並在此在特徵空間中尋找最佳的分類線。在本次實作中使用了不同的 kernel，可比較不同資料使用不同 kernel 的效果差異。

II. 訓練資料少仍能有不錯的效果。

4. ResNet18
   I. CNN base model，適合圖像辨識與分類。
   II. 由 18 層網路構成，主要使用殘差塊的結構，解決深層網路的梯度消失問題。
   III. 需要調整 learning_rate, gamma 等參數。

p.s. 除 ResNet18 沒有使用 dataset2 (non-image) 外，其餘分類器都有用三種資料集訓練與測試。

Analysis

1. Dataset1 (Color Classification)

資料集特性：類別多，圖片少 (單一類別：7~20 張圖片)，需自行拆分資料

1. 訓練資料數量的影響 (使用 train_test_split 切分) train_size=0.8 => 0.5

Accuracy (in test)	train_size=0.8	train_size=0.5
RandomForest (max_depth=8, n_estimators=500)	0.86	0.667
DecisionTree(max_depth=10, criterion='entropy')	0.727	0.463
SVC(kernel='linear', gamma='auto')	**0.95**	**0.796**
ResNet(pretrain weights=ImageNet) Epoch=10	0.739	0.618

SVM scores in test (在 SVM 的例子中，各項評分分數都差不多)

train_size=0.8 / 0.5	precision	recall	F1-score
macro avg	0.96 / 0.83	0.97 / 0.77	0.96 / 0.77
weighted avg	0.97 / 0.82	0.95 / 0.80	0.96 / 0.78

總結：由於初始資料數少，在減少後許多類別只剩個位數的圖片，因此當訓練資料變更少時，正確率下降幅度更大。

2. 不同分類器的效果和比較 (test)

macro avg / weighted avg	Random Forest (max_depth=8, n_estimators=500)	DecisionTree (max_depth=10, criterion='entropy')	SVC (kernel='rbf', gamma=1e-5, C=100)	ResNet (pretrained weights=ImageNet) Epoch=10
accuracy	0.86	0.73	**0.95**	0.739
Precision	0.85 / 0.90	0.69 / 0.77	**0.94 / 0.98**	X

(avg)				
Recall (avg)	0.88 / 0.86	0.68 / 0.73	**0.94 / 0.95**	X
F1-score (avg)	0.85 / 0.87	0.66 / 0.73	**0.92 / 0.95**	X

- 理論上 ResNet 會比 SVM 更善於處理圖片分類問題，但由於此類別的資料集很小，因此對於需要大量圖片資料的 ResNet 就無法有就好的效果，甚至效果很差，所以反而是 SVM 有更好的表現。
- 樹型模型架構簡單，而圖片資料的特徵與細節過多，像素間關係複雜，因此效果不佳。
- 不同分類器中的四項評分指標分數都差不多，因此可知這些分類器對每個類別的分類效果相對一致 (各資料樣本數較一致)。

3. 使用不同超參數的效果 (SVM in test)

SVM 使用不同 kernel (C=1) & 使用 GridSearchCV 找到的最佳參數 (kernel 選擇有 rbf, linear)

macro avg / weighted avg	linear	polynomial	RBF	RBF C=100, gamma=1e-5
accuracy	**0.95**	0.82	0.82	**0.95**
Precision (avg)	**0.96 / 0.97**	0.62 / 0.80	0.71 / 0.83	**0.94 / 0.98**
Recall (avg)	**0.97 / 0.95**	0.69 / 0.82	0.74 / 0.82	**0.94 / 0.95**
F1-score (avg)	**0.96 / 0.96**	0.64 / 0.79	0.70 / 0.80	**0.92 / 0.95**

- 使用 GridSearchCV 時，將 cv 參數由 5 改成 10 後，效果也有明顯進步，應該是因為 cv 調大較適合小型資料集
- 儘管不同 kernel 比較中，RBF 效果普通，但透過 GridSearchCV 找到適合參數後效果有明顯進步。

4. Data Augmentation (ResNet18)
   1. 由於訓練集太小，因此使用 torchvision 套件實作 data augmentation，增加訓練資料數量(不用於 Validation 部分)。
   2. 觀察圖片後發現顏色物件在畫面的佔比極大，因此就算裁剪了圖片仍可以清楚辨認圖片標籤。此外由於偵測目標是物體顏色，因此物體旋轉和裁剪而不完整理論上並不影響圖片特徵。基於觀察到的這些圖片特性，我使用隨機裁剪與隨機水平和垂直翻轉，增加圖片的同時盡量不影響到圖片特徵。

 =>  code: ./dataset1/data_augmentaion.ipynb

3. 比較與分析

```
optimizer = torch.optim.Adam(model.parameters(), lr=1e-2, weight_decay=1e-7)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer=optimizer, step_size=100, gamma=0.1)
loss_fn = nn.CrossEntropyLoss()
```

epoch = 10	只使用原訓練資料	加入處理過的資料
Training accuracy	0.897	**0.91**
Validation accuracy	0.571	**0.917**
Training loss	0.422	**0.289**
Validation loss	1.5	**0.208**
Testing accuarcy	0.618	**0.913**

結論：可以看出當加入處理後資料後，ResNet 效果有明顯的提升。

5. 與 kaggle 上的比較

kaggle 上的 SVM 範例 (Accuracy =92% , train_size=0.75) 只有簡單的使用 linear kernel 並將 gamma 設為 auto，並沒有對資料做太多的處理。

因此我使用 train_size=0.75 測試，先採用了一樣的構造 (AUC=88.9%)，又額外使用 GridSearchCV 做交叉驗證，包含不同的 kernel (linear, rbf)以及 C 和 gamma 選項，找出更佳的參數 (AUC=92.6%)。此外，因為資料集較小，因此我將 cv=5 改成 cv=10 後，效果也有明顯的進步。

2. Dataset2 (Mobile Price Classification)

資料集特性：非圖片資料集，高達 2000 筆資料，20 項特徵。

1. 訓練資料數量的影響 (training data 1400 => 700)

Accuracy	1400 筆	700 筆
RandomForest (max_depth=8, n_estimators=500)	0.872	0.807
DecisionTree(max_depth=10)	0.83	0.78
SVC(kernel='linear', C=1)	**0.972**	**0.963**

總結：資料減少，正確率幾乎都呈現下降，尤其是樹型模型變化較明顯。

然而幾次測試中 SVM 有時卻會略微上升，猜測可能的因素是每次切分資料皆為隨機，而且 SVM 在半數資料時就達到頗高的正確率，因此應屬於正常浮動。

p.s. 在 ./dataset3/get_train_test.ipynb 程式中可直接根據需求分割所需比例及數量的訓練與測試集。

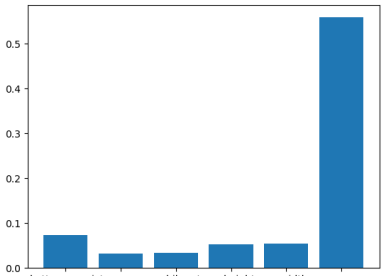
2. 不同分類器的效果和比較

macro avg / weighted avg	Random Forest (max_depth=8, n_estimators=500)	DecisionTree (max_depth=10)	SVC (kernel='rbf', gamma=1e-10, C=1e7)
train_accuracy	**0.99**	**0.99**	0.98
test_accuracy	0.88	0.85	**0.97**
Precision (avg)	0.88 / 0.88	0.85 / 0.85	**0.97 / 0.97**
Recall (avg)	0.88 / 0.88	0.85 / 0.85	**0.97 / 0.97**
F1-score (avg)	0.88 / 0.88	0.85 / 0.85	**0.97 / 0.97**

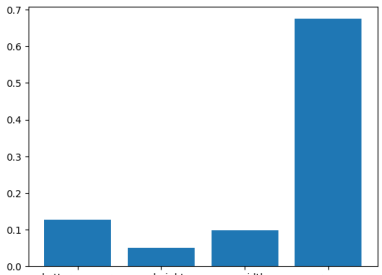
- 儘管在訓練時樹型模型都正確率都高於 SVM，但在測試時 SVM 的各項表現仍為最佳。
- 決策樹於訓練和測試的正確率差距最大，猜測是因為有點過度擬合。

3. 資料特徵重要性 (Feature Importance)

儘管樹型模型效果都沒有特別突出，但它卻能有解釋性的處理分類問題，以下是隨機森林和決策樹產出的特徵重要性直方圖 (各特徵分數加總為 1，大於 0.03 才顯示於圖表中)。



(隨機森林)



(決策樹)

由圖表可知，各項特徵中 ram 大小對於手機價格影響最大，其次是手機電量和手機像素，這些特徵也確實與實際上手機價格關聯性較強，所以儘管樹型模型效能並非最佳，卻更貼近實際分辨的思維邏輯。

4. 使用不同超參數的效果 (SVM in test)

SVM 使用不同 kernel (C=1) & 使用 GridSearchCV 找到的最佳參數 (kernel=rbf)

macro avg / weighted avg	linear	polynomial	RBF	RBF C=1e7, gamma=1e-10
accuracy	**0.967**	0.95	0.95	**0.971**
Precision (avg)	**0.97 / 0.97**	0.95 / 0.95	0.95 / 0.95	**0.97 / 0.97**
Recall (avg)	**0.97 / 0.97**	0.95 / 0.95	0.95 / 0.95	**0.97 / 0.97**
F1-score (avg)	**0.97 / 0.97**	0.95 / 0.95	0.95 / 0.95	**0.97 / 0.97**

與 Dataset1 不同，三種 kernel 的表現都不錯且接近，推測可能的原因

i. 資料本身可能是可線性分類的

ii. 資料特徵和分類具有明顯的函數關係 (根據特徵重要性，可以發現 ram 對類別影響頗大，而且次要重要的特徵也不多)

5. 與 kaggle 上的比較

Kaggle 上的範例也使用了決策樹、隨機森林、SVM，在經比較後也是 SVM 效果更好。調整參數後，範例獲得 0.983 的正確率，而我使用相同的參數卻獲得 0.965。在觀察後發現範例使用了 2000 筆資料，但我因為沒有 test.csv 的解答檔案，因此是將 train.csv 在拆成訓練和測試資料，導致訓練資料只有 1400 筆。

綜上所述，若能使用更多資料進行訓練，再加上使用 GridSearchCV 尋找合適參數，我認為很可能也能夠達到差不多的正確率。

3. Dataset3 (my cat or not my cat)

資料集特性：二分類問題，各類別資料數量平衡

1. 訓練資料數量的影響 (training data 186 => 133)

Accuracy in test	186 筆	133 筆
RandomForest (max_depth=8, n_estimators=200)	0.80	0.761
DecisionTree(max_depth=3)	0.77	0.739
SVC(kernel=rbf, C=1)	0.815	0.783
ResNet epoch=10	**0.851**	**0.813**

總結：資料減少，正確率都呈現下降。但若在減少訓練資料後調整參數或許效果能有些許提升。

I. 不同分類器的效果和比較 (in test)

macro avg / weighted avg	Random Forest (max_depth=8, n_estimators=500)	DecisionTree (max_depth=3)	SVC (kernel='rbf', C=1)	ResNet epoch=10
accuracy	0.85	0.703	0.81	**0.85**
ROCAUC	0.84	0.695	0.81	x
precision	0.85 / 0.85	0.70 / 0.70	**0.81 / 0.81**	0.766 / x
recall	0.85 / 0.85	0.70 / 0.70	0.81 / 0.81	**0.973 / x**
F1-score	0.85 / 0.85	0.70 / 0.70	0.81 / 0.81	**0.857 / x**

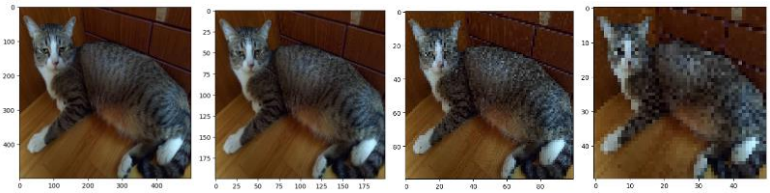
除了 precision 是 SVM 高一點，其他皆是 RestNet18 表現較佳。雖然資料總數看似與 dataset1 差不多，但由於只有兩個類別，因此單一類別還是會有 100 張左右的訓練資料。

綜上所述，儘管與 dataset2 一樣是圖片資料，但因為資料數較充足，因此 ResNet 的效果優於 SVM。

II. 使用不同超參數的效果 (SVM in test)

i. Image Size (Random Forest and Decision Tree)

Accuracy in test	Random Forest (max_depth=8, n_estimators=500)	DecisionTree (max_depth=3)	SVC (kernel='rbf', C=1)
500 x 500	0.79	0.728	0.81
200 x 200	0.81	0.80	0.81
100 x 100 (default)	0.85	0.703	0.81
50 x 50	0.84	0.74	0.82


(500~50)

- 隨機森林和決策樹中，發現正確率和圖像尺寸並非呈現正相關。猜測是因為圖片特徵量太多，因此樹型模型難以學習圖像豐富的特徵，而當圖片尺寸縮小，特徵數下降，能有好一點的表現。但當圖片縮太小後又會開始失真 (如上方右圖)，因此儘管圖像內特徵關係變簡單但正確率卻反而會下降。
- 圖像尺寸變化對於 SVM 的影響較小，但尺寸調大後訓練時間有明顯增長，因此 SVM 較另兩個模型適合使用較小的圖片尺寸。

ii. ResNet18 Pretraining?

epoch = 10	x	pretrain weights=ImageNet
Training accuracy	**0.856**	0.833
Validation accuracy	**0.851**	0.796
Training loss	**0.273**	0.362
Validation loss	**0.323**	0.575
Testing accuarcy	**0.852**	0.753
Precision (test)	**0.766**	0.707
Recall (test)	**0.973**	0.784
F1 score (test)	**0.857**	0.743

由於預訓練模型是在大數量的圖像資料上訓練的，然而此次只是一個小型資料集，預訓練模型的參數量太多，可能因此而效果反而沒有那麼好。

Discussion

1. Based on your experiments, are the results and observed behaviors what you expect?

大部分都符合我原先的預期，不過在 Dataset1 顏色分類的部分，因為資料數真的太少，SVM 能夠有很高的正確率讓我很意外。此外，在調整圖片尺寸部分，一開始測試決策樹時我就發現尺寸小而變得模糊的圖像，正確率有時不降反升。

此外，在自製資料集(dataset3)部分，我原本會擔心效果不好，畢竟圖片的背景大多很雜亂，拍攝光線也不一致，貓咪也呈現各種形狀。不過最後在 SVM 和 ResNet 卻有 80%以上的正確率，比我預期的效果還要更好。

2. Discuss factors that affect the performance, including dataset characteristics.
    1. 各種超參數：在這四種分類器中，只要參數改變都可能會使正確率有顯著改變，因此調整參數雖然費時但也是影響性能的重要因素。
    2. 資料處理：在本次的圖片處理都是調整尺寸和正規化，並無特別處理噪聲，只有在數據擴充 (Dataset1) 的部分有使用翻轉和裁剪。在 dataset1 中，因為重點在顏色所以裁剪並不會有太大的影響，但在 dataset3 中，因為目標是辨認貓咪，所以裁剪圖片就可能會對正確率有影響。
    3. 針對資料集的分類器選擇：
        i. 像是隨機森林和決策樹都只適合數據資料的分類器，雖然在此沒有突出的表現，但呈現特徵重要性的部分能提升模型的可解釋性。
        ii. SVM 在三個資料集中都有不錯的表現，尤其在資料少的 dataset1 中明顯的比其他分類器更為突出。
        iii. ResNet 雖然是不錯的神經網路模型，但在 dataset1 中卻明顯的非常劣勢。
        iv. 分類器的選擇應考慮資料集的數據類型、資料大小、資料分布是否平均等因素。
    4. 資料集樣本分布：如果數據中樣本分佈不均，容易讓模型過度擬合或欠擬合。因此若有不均則需要特別處理 (採樣、數據合成、加權等等) 或只為足夠的樣本建模，而將問題變成 openset problem。
    5. 數據量：若不考慮訓練時間，則越高的數據量通常能讓模型更好的學習並有更好的表現。
3. Describe experiments that you would do if there were more time available.
    1. 若能有更多時間，我想嘗試更多種數據擴充的方法，尤其是調整亮度的部分我覺得很適合用在自製的資料集上。最後再將資料用於兩個資料集以及四種分類器。

2. 嘗試更多參數組合，尤其是 SVM 和 ResNet18 這兩個表現較佳但訓練時間較長的模型。
3. 套用特徵降維，避免過度擬合，也能加快訓練速度，並且觀察不同降維方式的影響。
4. 嘗試物件偵測，並將物件偵測模型配合 dataset3 所做的貓咪辨識，讓分類器的辨識結果有所價值。

4. Indicate what you have learned from the experiments and remaining questions.
   1. 這次的作業讓我對於分類器的選擇更有概念，這次的三個資料集都有各自的特色，因此也顯示出就算分類器的性能再好，用於不適合的資料也不會有好的效果。
   2. 之前在評估模型時我大多是使用 AUC，但這次的 precision, recall, F1, AUROC 讓我對於評估模型更有概念。不過因為這次選擇的資料集資料分佈都很平均，因此沒有從實做中明顯觀察到這些指標的影響，都是從網路介紹了解概念。
   3. 處理與觀察資料部分我也在本次作業變得熟悉許多，尤其在 dataset1 時，有個 bug 找很久，後來才發現在此資料集中居然有不同通道數的圖片，而我一直都在檢查第一張圖片的通道數，而圖片又剛好是 RGB，後來是將通道數不為三的圖片都轉成 RGB 才成功。

Appendix

程式碼位於 appendix 資料夾中，使用.ipynb 檔案，有四個分類器檔案和三個存放資料集的資料夾。在.ipynb 檔案中皆有 Markdown 以及分出區塊，輸出結果也都有保留。

p.s. 正確率等數值可能跟報告不同，因為是隨機切分資料，因此重新切資料後結果可能會略有不同。