1. 分類問題(part a, d)

一個是二分類問題，另一個是四分類問題。這邊我分別使用了Deep Neural Network和Random Forest來進行分類。下面是我經過多次實驗所定下的架構與參數:

1. Binary Classification:
2. DNN (test\_acc = 0.871)

Inputs: 2class\_training.csv, feature1~118

Outputs: 2class\_training.csv, label

Validation: 2class\_test.csv

Model: each layer units is (200, 100, 50, 3, 1)

Dropout: each layer with (0.5, 0.5, 0.5, 0.3)

Optimizer: Adam(lr=0.001, beta1=0.9, beta2=0.99)

Kernel initializer: glorot uniform (bias = zeros)

Activation: Relu in first four layers and sigmoid in last

Layer

Loss function: Mean Square Error

1. Random Forest (test\_acc = 0.91)

Inputs: 2class\_training.csv, feature1~118

Outputs: 2class\_training.csv, label

Criterion: gini (分類依據)

Bootstrap: 隨機放回抽樣

Max\_feature: sqrt(feature數) (每個決策樹最大特徵數量)

1. Four Categories Classification:
2. DNN (test\_acc = 0.756)

Inputs: 2class\_training.csv, feature1~118

Outputs: 2class\_training.csv, label

Validation: 2class\_test.csv

Model: each layer units is (200, 100, 50, 32, 8, 4)

Dropout: each layer with (0.5, 0.5, 0.5, 0.5, 0.5)

Optimizer: Adam(lr=0.001, beta1=0.9, beta2=0.99)

Kernel initializer: glorot uniform (bias = zeros)

Activation: Relu in first five layers and softmax in last

Layer

Loss function: Mean Square Error

1. Random Forest(test\_acc = 0.641)

Inputs: 2class\_training.csv, feature1~118

Outputs: 2class\_training.csv, label

Criterion: gini (分類依據)

Bootstrap: 隨機放回抽樣

Max\_feature: sqrt(feature數) (每個決策樹最大特徵數量)

經過多次實驗，可以發現決策樹對於2class的分類比DNN還要精準，且穩定。不過在4class的部分則是輸給了DNN。當我嘗試去改進決策樹的準確率時發現，能夠調整的參數很有限，這部分是決策樹和DNN最大的差別，DNN做得夠好是能夠超越決策樹的，決策樹的優點在於容易使用，需要較少的前處理，訓練過程也較簡單。

1. 添加雜訊(part b, c)

在機器學習訓練的過程中，常常會在訓練的圖片上添加雜訊，因為可

以防止過擬合且增加泛化性，尤其是對於資料量較小的訓練幫助更大。這次作業所要做的就是在資料中加上雜訊並觀察現象。這邊我用了兩種方法來觀察結果，一是用一般訓練完的model來預測加上雜訊的資料，二是用加入雜訊在資料中訓練出來的model來預測資料。這邊統一用SNR(db)=10

1. 一般模型:

2class: test acc = 0.871

4class: test acc = 0.756

在input加入雜訊:

2class: test acc = 0.846

4class: test acc = 0.692

在output加入雜訊:

2class: test acc = 0.795

4class: test acc = 0.756

觀察出當輸入或輸出被雜訊影響時，都會造成正確率的下降。

1. 加入雜訊模型:

2class: test acc = 0.859

4class: test acc = 0.743

在input加入雜訊:

2class: test acc = 0.808

4class: test acc = 0.680

在output加入雜訊:

2class: test acc = 0.628

4class: test acc = 0.731

可以觀察到，加入雜訊訓練的模型準確率有些微的下降，且在

Testing時加入雜訊準確率下降得更多了。

我們一般會以為加入雜訊訓練模型可以達到更好的結果、更高的準確率，但是似乎並不是在所有情況都適用這個方法，例如這次的表格資料。在訓練中加入雜訊反而造成準確率的下降。而利用雜訊訓練出的模型對抗雜訊的能力也沒有比較好。當我用含雜訊的input做testing時，準確率還是有明顯的下降。所以我認為加入雜訊訓練模型對於影像處理方面是比較好的方法，對於表格資料則不一定。