EECE5644
ID 001459022
Yu Shun Lin Exam 2

1. Question
   Theory and math calculation of MAP classification

find the MAP classification with minimum

Let xSample be the data with $xSample \in \mathbb{R}^3$

$\Rightarrow \hat{\theta}_{MAP} = \arg\max P(\theta | xSample)$ by maximum likelihood

$= \arg\max_{\theta} \prod_{i=1}^{N} P(\theta | xSample_i) = \arg\max_{\theta} \prod_{i=1}^{N} P(xSample_i | \theta) P(\theta)$

$\Rightarrow \arg\max_{\theta} \prod_{i=1}^{N} \ln P(\theta | xSample_i) = \arg\max_{\theta} \prod_{i=1}^{N} \ln P(xSample_i | \theta) P(\theta)$

$= \arg\max_{\theta} \sum_{i=1}^{N} \ln P(xSample_i | \theta) P(\theta)$

Since the xSample are $\in \mathbb{R}^3$ mixture Gaussian data, $x = xSample$

$\Rightarrow \arg\max_{\theta} \sum_{i=1}^{N} \ln \frac{1}{\sqrt{2\pi}^k \sigma^2} e^{-\frac{1}{2\sigma}[(x_i-\mu)^T (x_i-\mu)]}$

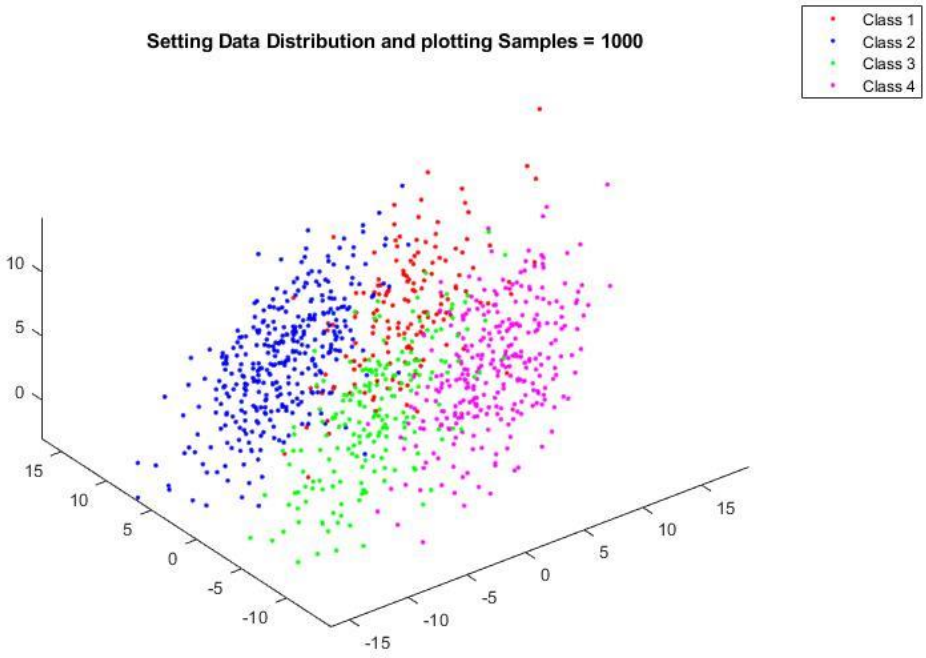
$= \arg\max_{\theta} \sum_{i=1}^{N} \ln \frac{1}{\sqrt{2\pi}^k \sigma^2} - \frac{1}{2\sigma}[(x_i-\mu)^T (x_i-\mu)]$

$= \arg\max_{\theta} \sum_{i=1}^{N} \ln \frac{1}{\sqrt{2\pi}^k} - \ln \sigma - (x_i-\mu)^T (x_i-\mu)$

$= \arg\max_{\theta} \sum_{i=1}^{N} -\ln \sigma - (x_i-\mu)^T \sigma^{-1} (x_i-\mu)$

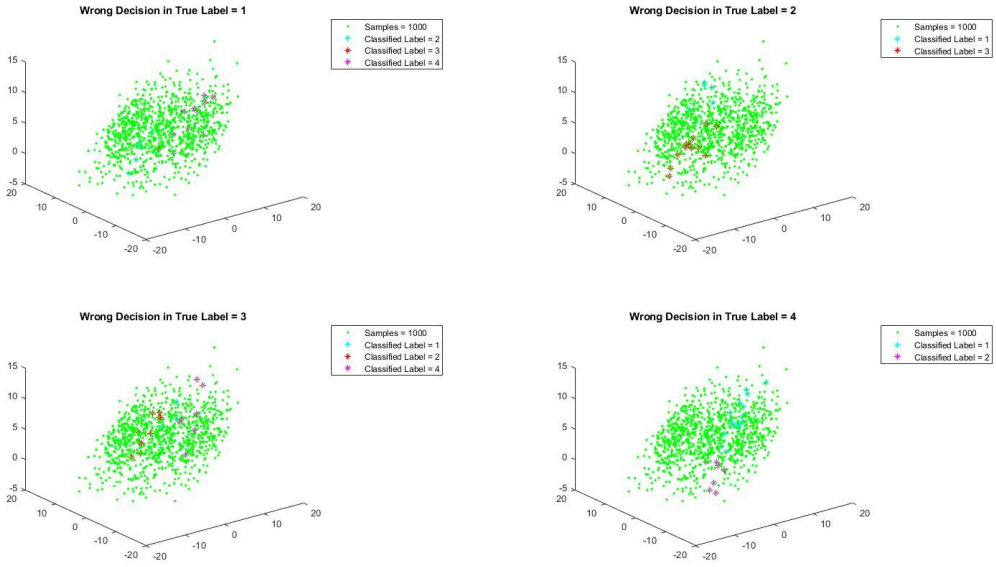
$\Rightarrow \hat{\theta}_{MAP} = \arg\min_{\theta} \sum_{i=1}^{N} \ln \sigma + \frac{(x_i-\mu)^T (x_i-\mu)}{\sigma}$

EECE5644
ID 001459022
Yu Shun Lin Exam 2



The distribution data is generated with
Prior = [0.15,0.35,0.2,0.3] where true sample number = [154,351,197,298]
Sigma = 0.8*[5 1 2;1 5 0;2 0 3] /sqrt(2)* covarianceVector for each class
covarianceVector = [1.3^2 0 0;0 1.2^2 0;0 0 1.4^2]

Apply this data distribution to the MAP classifier:



These plots show the MAP classification result
We get the confusion matrix = [137,6,1,10;6,333,12,0;3,10,177,7;20,0,7,271]

EECE5644
ID 001459022
Yu Shun Lin Exam 2

Where represent the right classification of
Class 1 = 137
Class 2 = 333
Class 3 = 177
Class 4 = 271 in 1000 true samples

Thus, we can calculate the whole MAP classification accuracy rate = (137+333+177+271) /1000 = 91.8%
And we get the decision error of each class is
Class 1 error = 0.11
Class 2 error = 0.05
Class 3 error = 0.10
Class 4 error = 0.09

EECE5644
ID 001459022
Yu Shun Lin Exam 2

Theory and math calculation of neural network:

With neural network = find the maximum likelihood x=Samples l=label

$$\Rightarrow \hat{\theta}_{NN} = \arg\max_{\theta} \prod_{i=1}^{N} P(x_i, l_i|\theta)$$

$$= \arg\max_{\theta} \sum_{i=1}^{N} P(x_i, l_i|\theta) = \arg\max \sum_{i=1}^{N} P(l_i|x_i\theta)P(x_i|\theta)$$

$$= \arg\max_{\theta} \sum_{i=1}^{N} \ln P(l_i|x_i\theta)P(x_i|\theta)$$

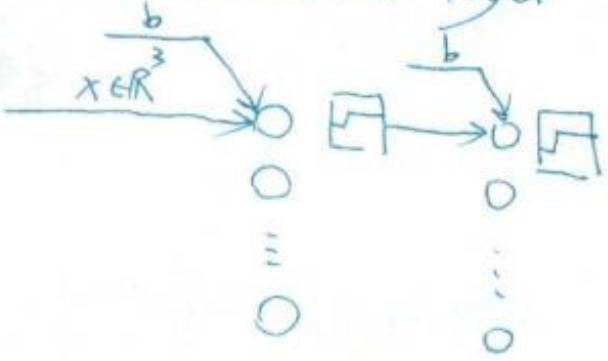
$$\Rightarrow \arg\max_{\theta} \sum_{i=1}^{N} \ln P(l_i|x_i\theta) + \sum_{i=1}^{N} \ln(P(x_i|\theta)) \quad \Rightarrow x_i \text{ is independent}$$

$$\Rightarrow \arg\max_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ln P(l_i|x_i,\theta) = \arg\min_{\theta} -\frac{1}{N} \sum_{i=1}^{N} \ln P(l_i|x_i,\theta)$$

Design a two hidden layer neural network, where two layer with weight and bias.

⇒ first layer with 'tanh', nonlinear.

⇒ second layer with 'softplus' linear.



⇒ R^3 ⇒ first layer ⇒ $R^{\#node}$

$R^{\#node}$ ⇒ second layer ⇒ R^4 (4 classifcation)

then soft $f(x) = \ln(1+e^x)$

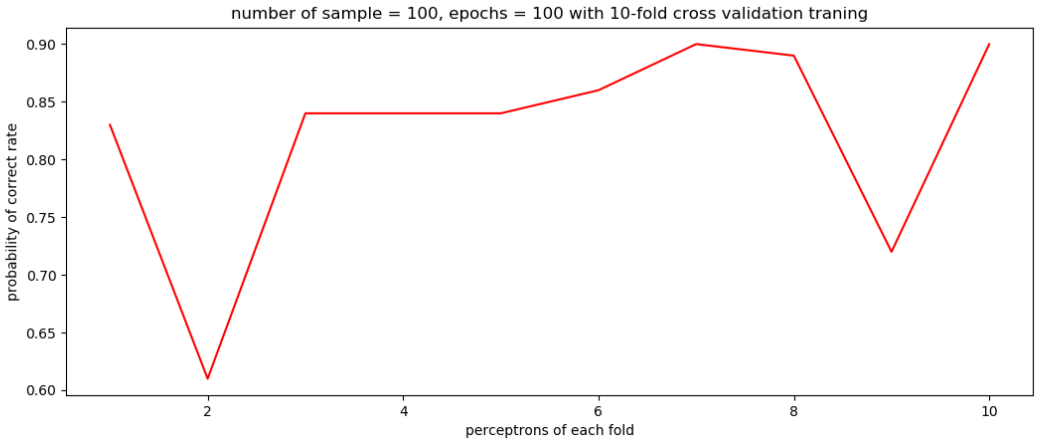
where take the best model for prediction

(softplus will more prefer in higher accuracy of estimation).

EECE5644
ID 001459022
Yu Shun Lin Exam 2

a. Training 100 data for getting the model in Keras library in python

```
2019-11-26 11:56:36.626017: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 11:56:36.628563: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen che
ck.
2019-11-26 11:56:36.630399: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 11:56:36.631867: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compi
led to use: AVX2
2019-11-26 11:56:36.635608: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 11:56:36.637793: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen che
ck.
2019-11-26 11:56:36.639810: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 11:56:37.053994: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2019-11-26 11:56:37.056166: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]      0
2019-11-26 11:56:37.057225: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0:   N
2019-11-26 11:56:37.058849: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1304] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GP
U:0 with 4606 MB memory) -> physical GPU (device: 0, name: GeForce RTX 2060, pci bus id: 0000:01:00.0, compute capability: 7.5)
2019-11-26 11:56:37.761780: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cublas64_100.dll
Accuracy Rate on Train Data 100: 0.8999999761581421
[0.26291084468364717, 0.9200000166893005]
Accuracy Rate on Test Data 100: 0.9200000166893005
The confusion matrix is as below:
[[10  1  1  2]
 [ 1 32  0  0]
 [ 0  1 15  0]
 [ 1  0  1 35]]
Running Time =  186.8040363
Press any key to continue . . .
```



Training the 100 samples data with neural network with 2 layers, with 1st layer setting the ‘tanh’ activation function, and the 2nd layer setting the ‘softplus’ activation function.

After the training, the best node, bias and weight will pass to the model of Keras object.
Let this object test the test data of 100 samples.

The result shows the confusion matrix, which allow us to find the accurate estimation
In 100 testing sample and 100 10-fold training.
True number of elements = [14,33,16,37]

Correct in label 1 = 10
Correct in label 2 = 32
Correct in label 3 = 15

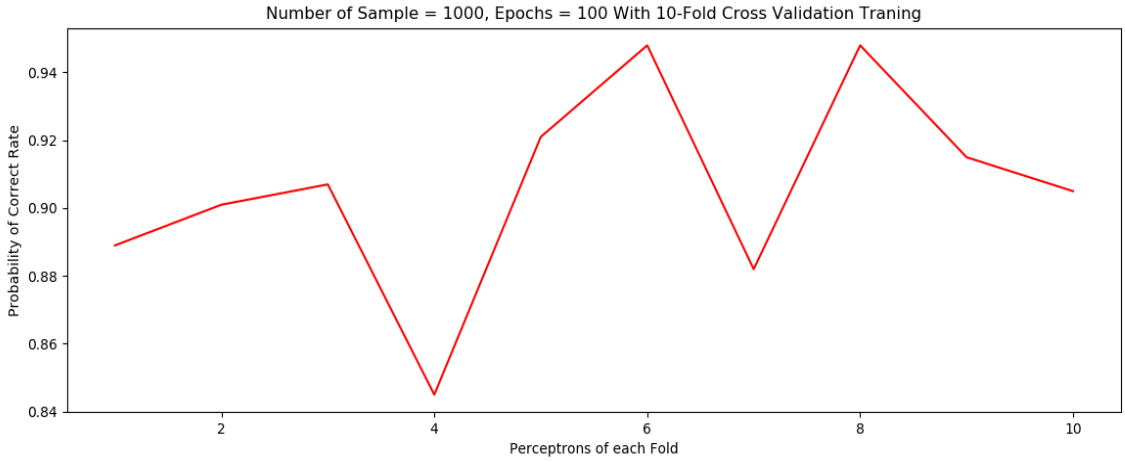
EECE5644
ID 001459022
Yu Shun Lin Exam 2

Correct in label 4 = 35 Accuracy = (10+32+15+35)/100 = 0.92

In this case, we can observe that there might have bad tanning case beget to low probability of accuracy rate.

b. Training 1000 data for getting the model in Keras library in python

```
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 11:15:14.389384: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen check.
2019-11-26 11:15:14.391043: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 11:15:14.392416: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2019-11-26 11:15:14.396127: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 11:15:14.398924: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen check.
2019-11-26 11:15:14.400861: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 11:15:14.838404: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2019-11-26 11:15:14.840284: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]      0
2019-11-26 11:15:14.841407: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0:   N
2019-11-26 11:15:14.842676: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1304] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 4606 MB
memory) -> physical GPU (device: 0, name: GeForce RTX 2060, pci bus id: 0000:01:00.0, compute capability: 7.5)
2019-11-26 11:15:15.533489: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cublas64_100.dll
Accuracy Rate on Train Data 1000: 0.9300000071525574
[0.20013089179992677, 0.9340000152587891]
Accuracy Rate on Test Data 1000: 0.9340000152587891
The confusion matrix is as below:
[[124   1   0  16]
 [  8 330  15   0]
 [  0  12 191   0]
 [ 13   0   1 289]]
Running Time =  2318.3332645
Press any key to continue . . .
```



Same as the former training in 100 samples. We take the first layer with activation function ‘tanh’, and the second layer with activation function ‘softplus’.
After getting the best model in the training, fit the model to test samples.

The result shows the confusion matrix, which allow us to find the accurate estimation
In 1000 testing sample and 1000 10-fold training.
True number of elements = [141,353,203,303]

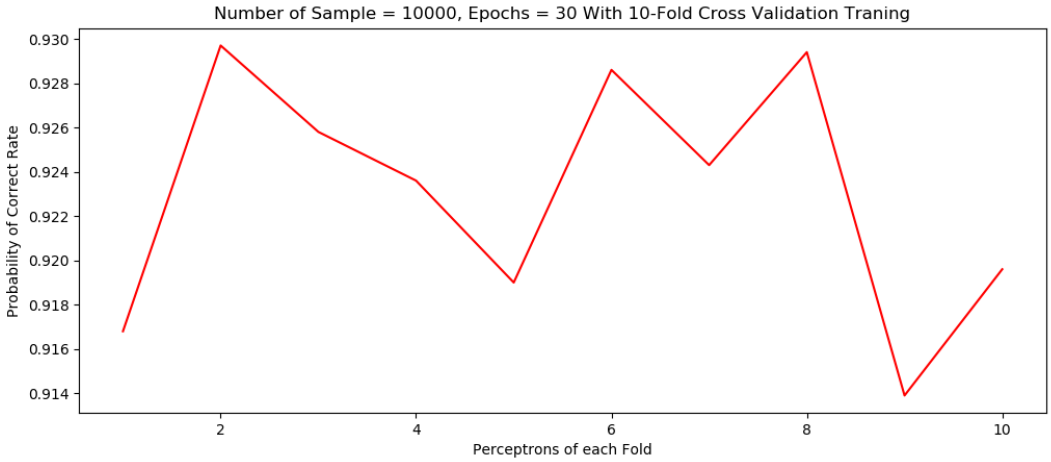
Correct in label 1 = 124
Correct in label 2 = 330
Correct in label 3 = 191
Correct in label 4 = 289 Accuracy = (124+330+191+289)/1000 = 0.934

Same here we could observe that some bas case might be caught for training, but the accuracy rate is better than 100 sample case. Which means the probability of pick bad training module would be lower.

EECE5644
ID 001459022
Yu Shun Lin Exam 2

c. Training 10000 data with epoch 30 for getting the model in Keras library in python

```
Training Samples with Neural Network
Training for 10000 samples...
2019-11-26 12:08:52.870565: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library nvcuda.dll
2019-11-26 12:08:52.889310: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 12:08:52.891498: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen check.
2019-11-26 12:08:52.893127: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 12:08:52.894493: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not compiled to use: AVX2
2019-11-26 12:08:52.898533: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 12:08:52.901131: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen check.
2019-11-26 12:08:52.902839: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 12:08:53.314299: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2019-11-26 12:08:53.315878: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]      0
2019-11-26 12:08:53.316784: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0:   N
2019-11-26 12:08:53.318008: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1304] Created TensorFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 4606 MB memory
) -> physical GPU (device: 0, name: GeForce RTX 2060, pci bus id: 0000:01:00.0, compute capability: 7.5)
2019-11-26 12:08:54.053277: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cublas64_100.dll
Accuracy Rate on Train Data 10000: 0.9279999732971191
[0.21498292227983476, 0.930899977684021]
Accuracy Rate on Test Data 10000: 0.930899977684021
The confusion matrix is as below:
[[1308   58   14  137]
 [   4 3346  129    0]
 [  14  173 1789   49]
 [ 103    0   10 2866]]
Running Time =  11690.229969499998
Press any key to continue . . .
```



Same method as former two experiment
We get the confusion matrix that we could get the accurate rate and the wrong estimation number
True number of elements in 10000 data = [1517,3479,2025,2979]
Correct in label 1 = 1308
Correct in label 2 = 3346
Correct in label 3 = 1789
Correct in label 4 = 2866 Accuracy = 9309/10000 = 0.9309

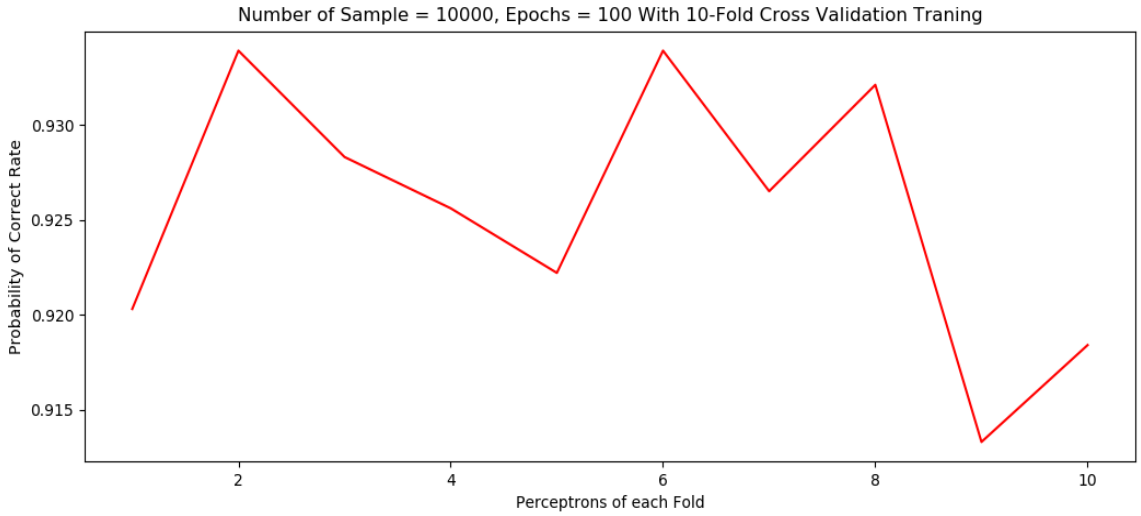
This result does not show a better accuracy than previous two experiment since the epoch = 30 which is lower than 100

Thus, take epochs = 100 in the following experiment

EECE5644
ID 001459022
Yu Shun Lin Exam 2

d. Training 10000 data with epoch 100 for getting the model in Keras library in python

```
2019-11-26 17:19:50.339493: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cudart64_100.dll
Using TensorFlow backend.
Training Samples with Neural Network
Training for 10000 samples...
2019-11-26 17:19:53.494953: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library nvcuda.dll
2019-11-26 17:19:53.516182: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 17:19:53.518668: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen ch
eck.
2019-11-26 17:19:53.521407: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 17:19:53.523411: I tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports instructions that this TensorFlow binary was not comp
iled to use: AVX2
2019-11-26 17:19:53.527470: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1618] Found device 0 with properties:
name: GeForce RTX 2060 major: 7 minor: 5 memoryClockRate(GHz): 1.83
pciBusID: 0000:01:00.0
2019-11-26 17:19:53.530077: I tensorflow/stream_executor/platform/default/dlopen_checker_stub.cc:25] GPU libraries are statically linked, skip dlopen ch
eck.
2019-11-26 17:19:53.532934: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1746] Adding visible gpu devices: 0
2019-11-26 17:19:53.963873: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1159] Device interconnect StreamExecutor with strength 1 edge matrix:
2019-11-26 17:19:53.965539: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1165]      0
2019-11-26 17:19:53.966486: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1178] 0:   N
2019-11-26 17:19:53.967837: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1304] Created TensorFlow device (/job:localhost/replica:0/task:0/device:G
PU:0 with 4606 MB memory) -> physical GPU (device: 0, name: GeForce RTX 2060, pci bus id: 0000:01:00.0, compute capability: 7.5)
2019-11-26 17:19:54.664314: I tensorflow/stream_executor/platform/default/dso_loader.cc:44] Successfully opened dynamic library cublas64_100.dll
Accuracy Rate on Train Data 10000: 0.9300000071525574
[0.17136851172447204, 0.9384999871253967]
Accuracy Rate on Test Data 10000: 0.9384999871253967
The confusion matrix is as below:
[[1323   26   25  143]
 [  11 3397   71    0]
 [  22  197 1787   19]
 [  76    0   25 2878]]
Running Time =  41445.527304999996
Press any key to continue . . .
```



Here I tried the different if it can get a better result for accurate rate
Increase the epoch from 30 to 100.
The accurate rate: 0.9384
This accurate is a little better than the former 1000 training case

Conclusion of Question 1:
By the previous experiment, NN training is a little better than MAP classification. But it depends on the amount of training sample. As the number of training epochs increase, the accuracy supposed to be increase.

EECE5644
ID 001459022
Yu Shun Lin Exam 2

Question 2

Similar as Question 1.'s part 3.

create a neural network but with only 1 layer.

$\Rightarrow$ data $= [X_1, X_2]^T$ X_1 is the input X_2 is the target

$$\hat{\theta}_{NN} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \ln P(X_i, d_i|\theta) \quad \text{where } d \approx f(x,\theta)$$

$$\Rightarrow \hat{\theta}_{NN} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \ln P(d_i|X_i,\theta) \quad \text{since independent model}$$

$\Rightarrow$ prior $P(X_i|\theta)$ ignored

$$\Rightarrow N = \ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\right) + \frac{(X-\mu)^2}{2\sigma^2} \quad \text{(by gaussian formula)}$$

$\Rightarrow$ find minimun of $N \Rightarrow N = -(X-\mu)^2$

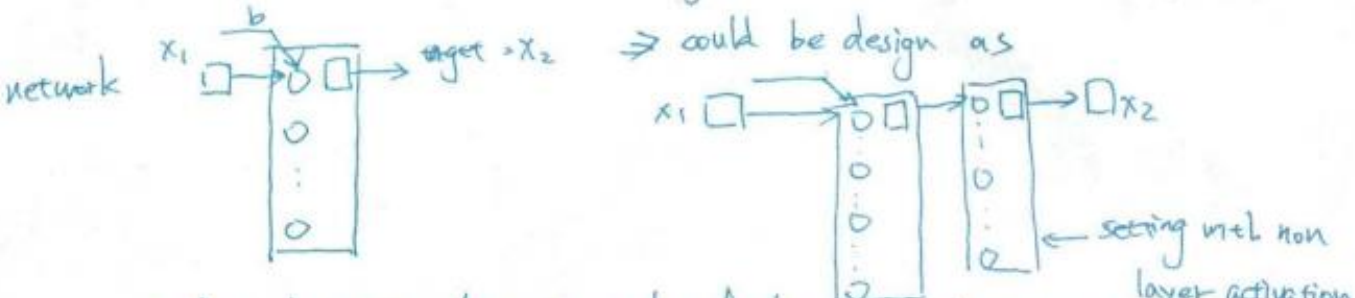
$$\Rightarrow \hat{\theta}_{NN} = \underset{\theta}{\operatorname{argmin}} -\sum_{i=1}^{N} (X-\mu)^2$$

$$\Rightarrow \hat{\theta}_{NN} = \underset{\theta}{\operatorname{argmin}} -\frac{1}{N}\sum_{i=1}^{N} (d_i - f(x_i,\theta))^2$$

where means MSE

Train the data find MSE be converged when difference < 0.1

network X_1 b target $= X_2$ $\Rightarrow$ could be design as



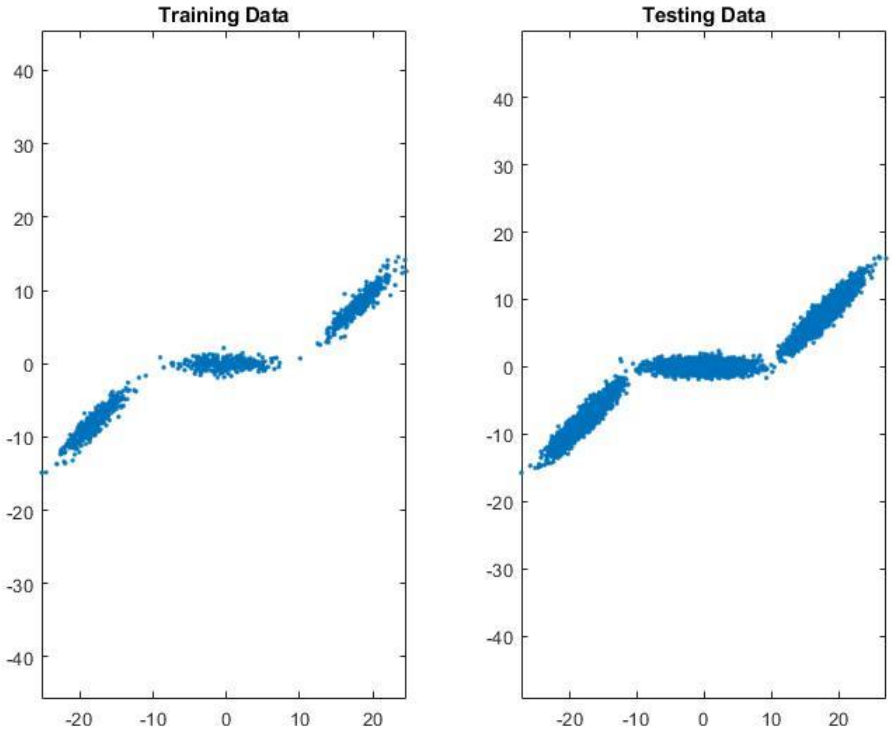
Select between logistic and softplus when determine the final model for testing 10000 data.

Theory and mathematical calculation:

Training and experiment:

EECE5644
ID 001459022
Yu Shun Lin Exam 2

The original data of training and testing data



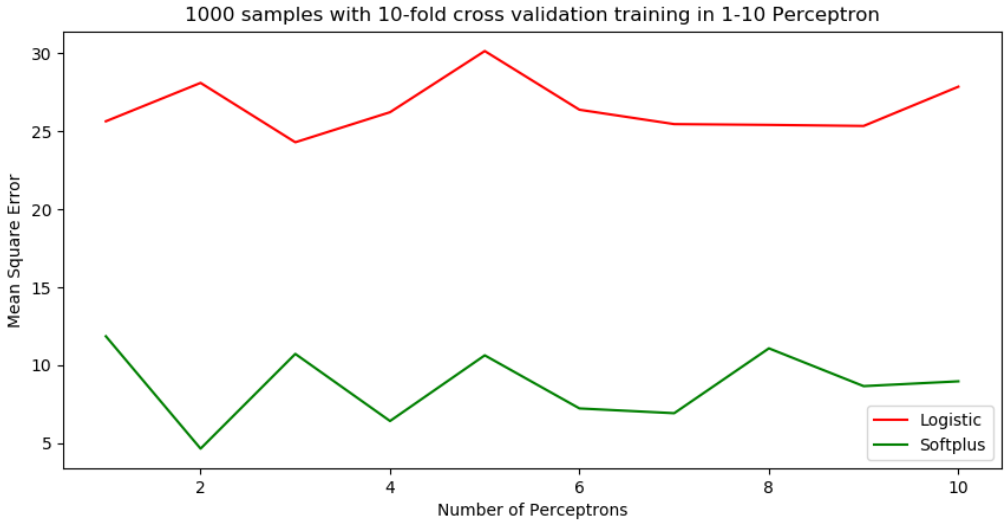
The plots show the original data of training and testing which are 1000 and 10000 samples
Training is base on finding the mean square error of inputs x1. when find a newMSE, compare the newMSE to the old MSE value, if the difference is smaller than a value (define as epsilon), determine as converged.

With the method finding the best argmin value in the mathematical part show as the best model

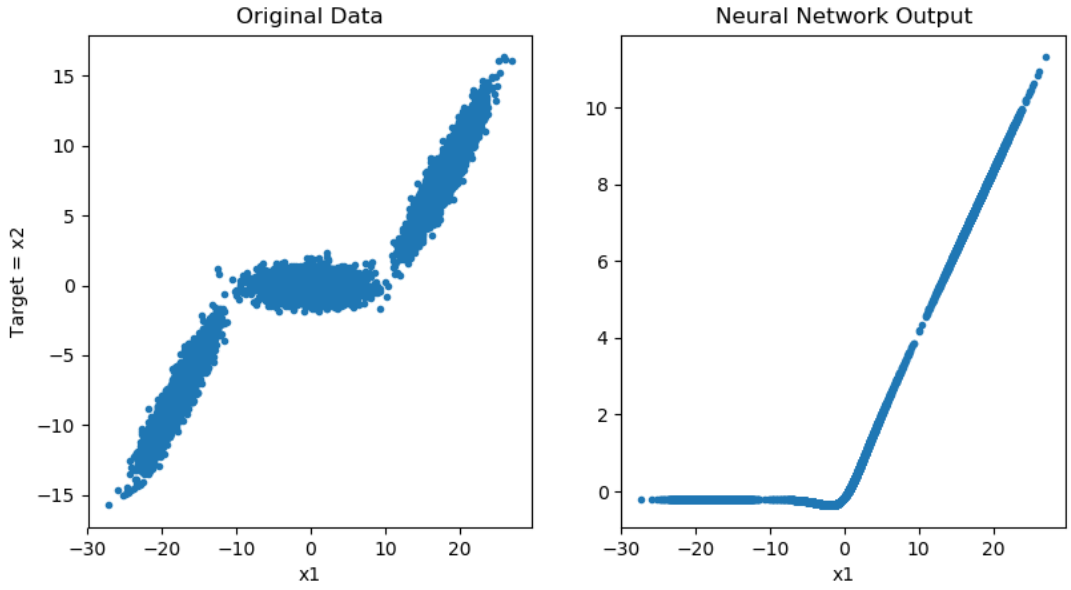
Separate the training data as 10-Fold to train and validate the model.
Then apply this model to 10000 testing data

Tried the following epoch to observe that is the final prediction is getting like the original testing data
epochs = 1
epochs = 30
epochs = 50
epochs = 100

EECE5644
ID 001459022
Yu Shun Lin Exam 2

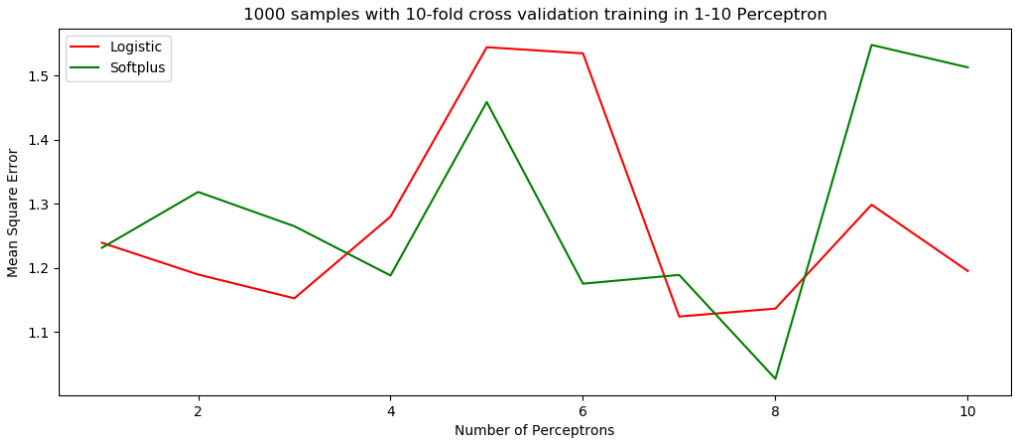


This result shows the NN training only with epochs = 1, both logistic and softplus have large MSE
The result could be bad as below

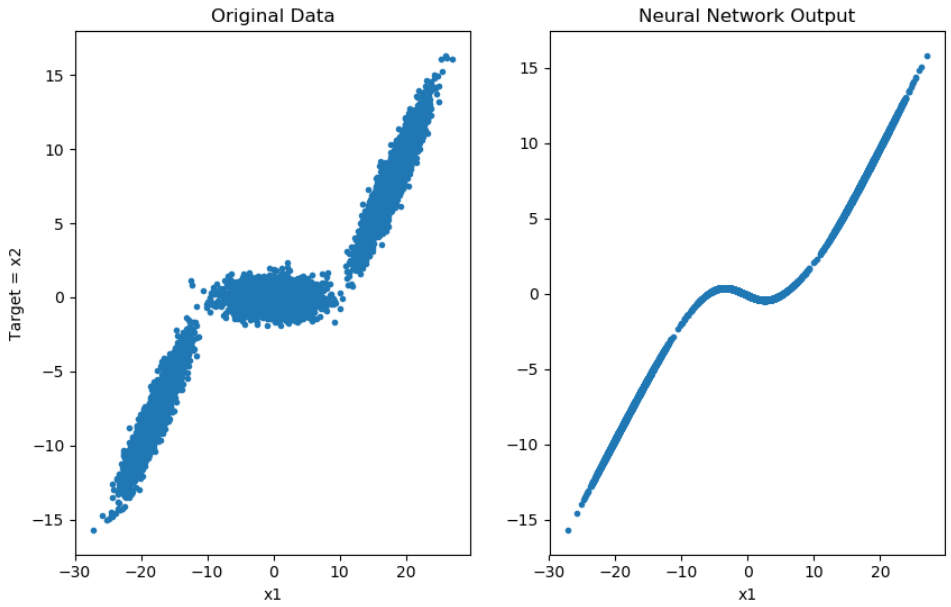


Now we increase the number of epochs to 50 see that if it is better

EECE5644
ID 001459022
Yu Shun Lin Exam 2



With 50 Epochs, the MSE become smaller. Thus, we might have better estimation



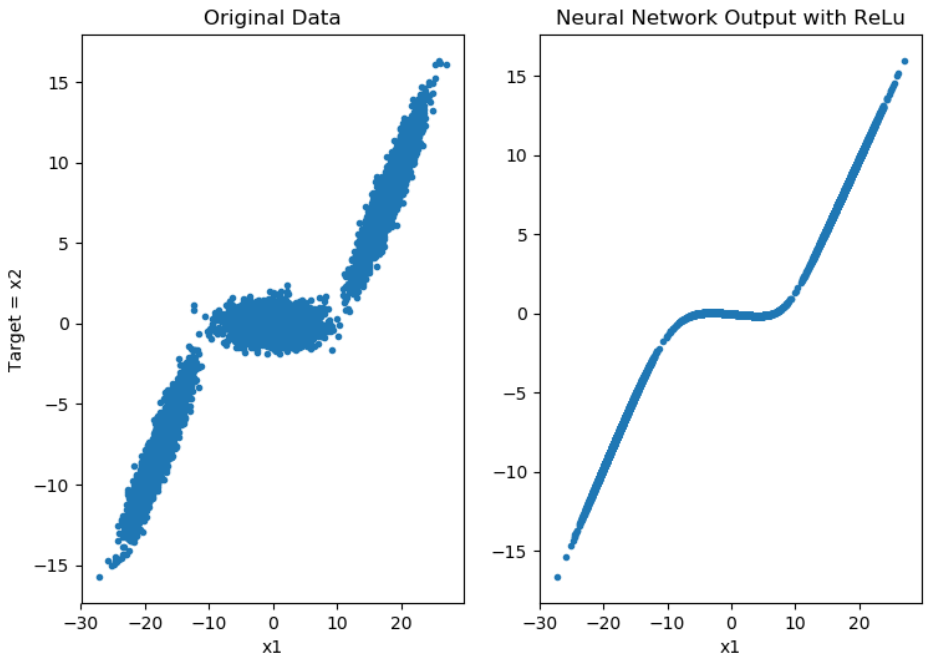

```
0.9630711078643799
Score in each train_validation determination 0.9630711078643799
MSE on Test Data: 0.6719991505146027
Running Time =  4749.9818935
Press any key to continue . . .
```

These plot shows better than the former one. But the middle data show two curves (not sharp turn) because the data distribution of Target (x2) is around 0 with Gaussian distribution. The MSE performance = 0.6720

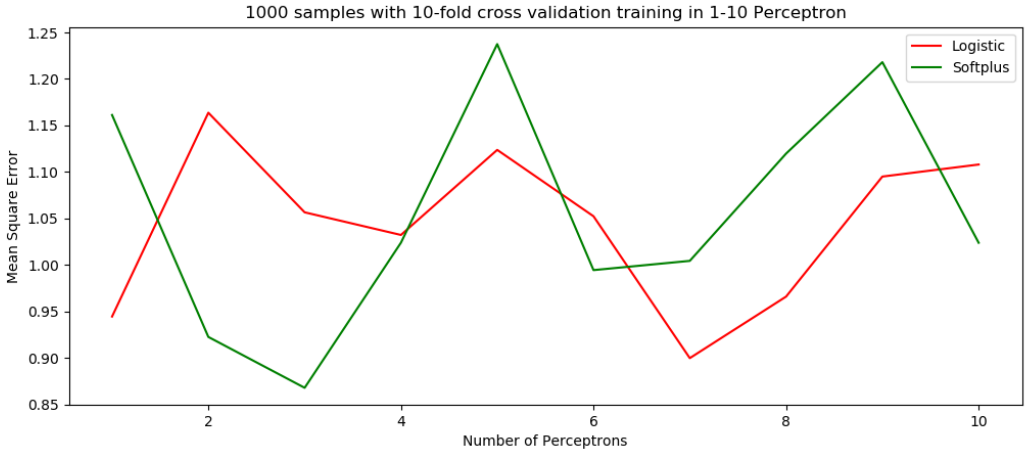
The Neural Network training is supposed to be like linear combination of two activation function. The expect result will be like SmoothReLu.

Thus, try the relu function in Keras to show the case

EECE5644
ID 001459022
Yu Shun Lin Exam 2

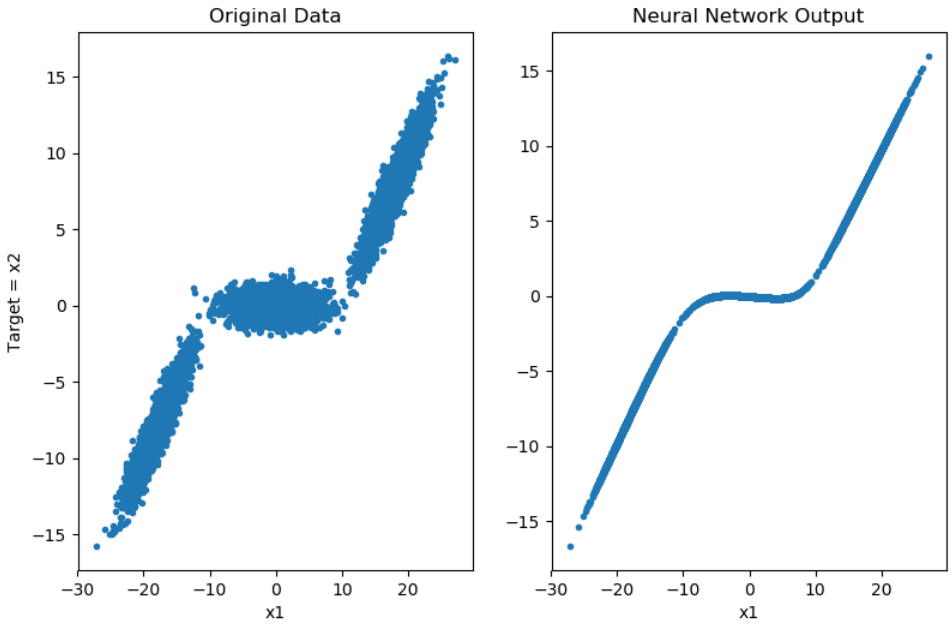


ReLu shows the expected result of Neural Network training result
Now we try to increase the epochs = 100 to see if it is getting similar to ReLu to check the neural network is create correctly



The plot shows the logistic and sofplus activation function's MSE performance in each perceptron

EECE5644
ID 001459022
Yu Shun Lin Exam 2



The 100 epochs result seems better than the 50-epochs' one. This indicate the neural network is created correctly.

Here the performance MSE = 0.6016

```
0.7053458261489868
Score in each train_validation determination 0.7053458261489868
MSE on Test Data: 0.600523956155777
10000/10000 [==============================] - 0s 28us/step
10000/10000 [==============================] - 0s 28us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 26us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 28us/step
10000/10000 [==============================] - 0s 26us/step
10000/10000 [==============================] - 0s 29us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 30us/step
10000/10000 [==============================] - 0s 28us/step
10000/10000 [==============================] - 0s 26us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 27us/step
10000/10000 [==============================] - 0s 27us/step
MSE on Test Data in Smooth ReLu: 0.6015588176727295
Running Time =  6372.7752212000005
```

Conclusion of Question 2:

The more training epochs, the more like the desire result. Also, the Mean Square Error will become smaller. The final 100 epochs training is similar to the ReLu activation function which is desire that the neural network being.

EECE5644
ID 001459022
Yu Shun Lin Exam 2

Reference:

https://keras.io/backend/
https://github.com/keras-team/keras/tree/master/examples
https://www.tensorflow.org/
https://www.tensorflow.org/tutorials/keras/classification
https://scikit-learn.org/stable/model_selection.html#model-selection
https://matplotlib.org/3.1.1/api/_as_gen/matplotlib.pyplot.html
https://www.mathworks.com/help/thingspeak/create-and-train-a-feedforward-neural-network.html
https://www.mathworks.com/help/deeplearning/ref/network.html
https://www.mathworks.com/help/deeplearning/ref/mse.html
https://www.mathworks.com/help/deeplearning/ug/create-and-train-custom-neural-network-architectures.html
https://www.mathworks.com/help/deeplearning/ref/train.html#namevaluepairarguments

Code Resource:

https://github.com/MakiseYuki/EECE5644-Machine-Learning/tree/master/Exam%202