**Assignment 2 Report**

Problem 1: logistic regression

A binomial **logistic regression** perceptron is used to predict the probability such that an observation falls into one of two class labels based on one or more independent variables.

ROC curve:

ROC stands for Receiver Operating Characteristic

It is a plot of the true positive rate against the false positive rate

AUC(Area under the curve):

Area under the curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one (assuming 'positive' ranks higher than 'negative')

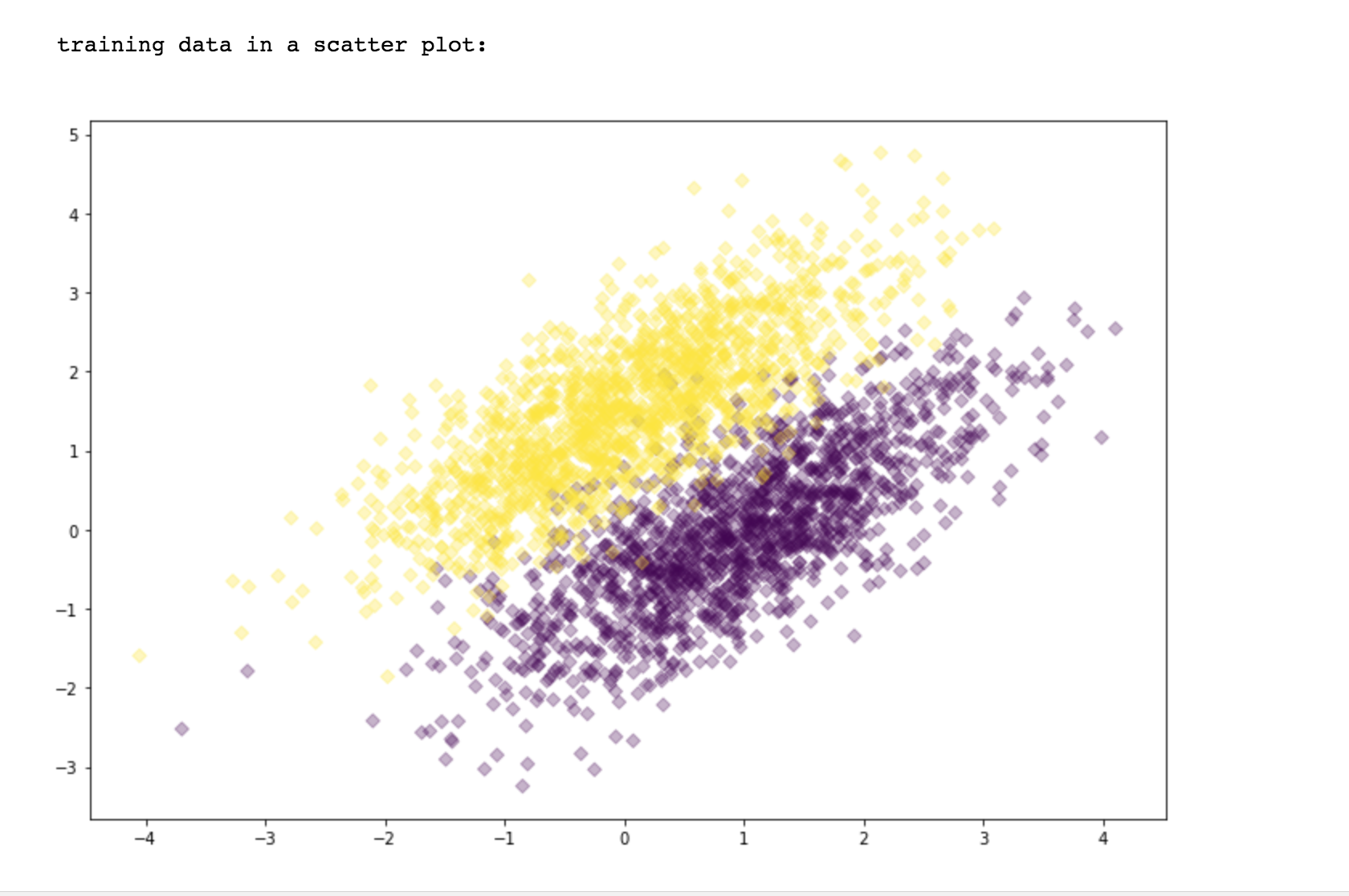
* Implementation Logic:

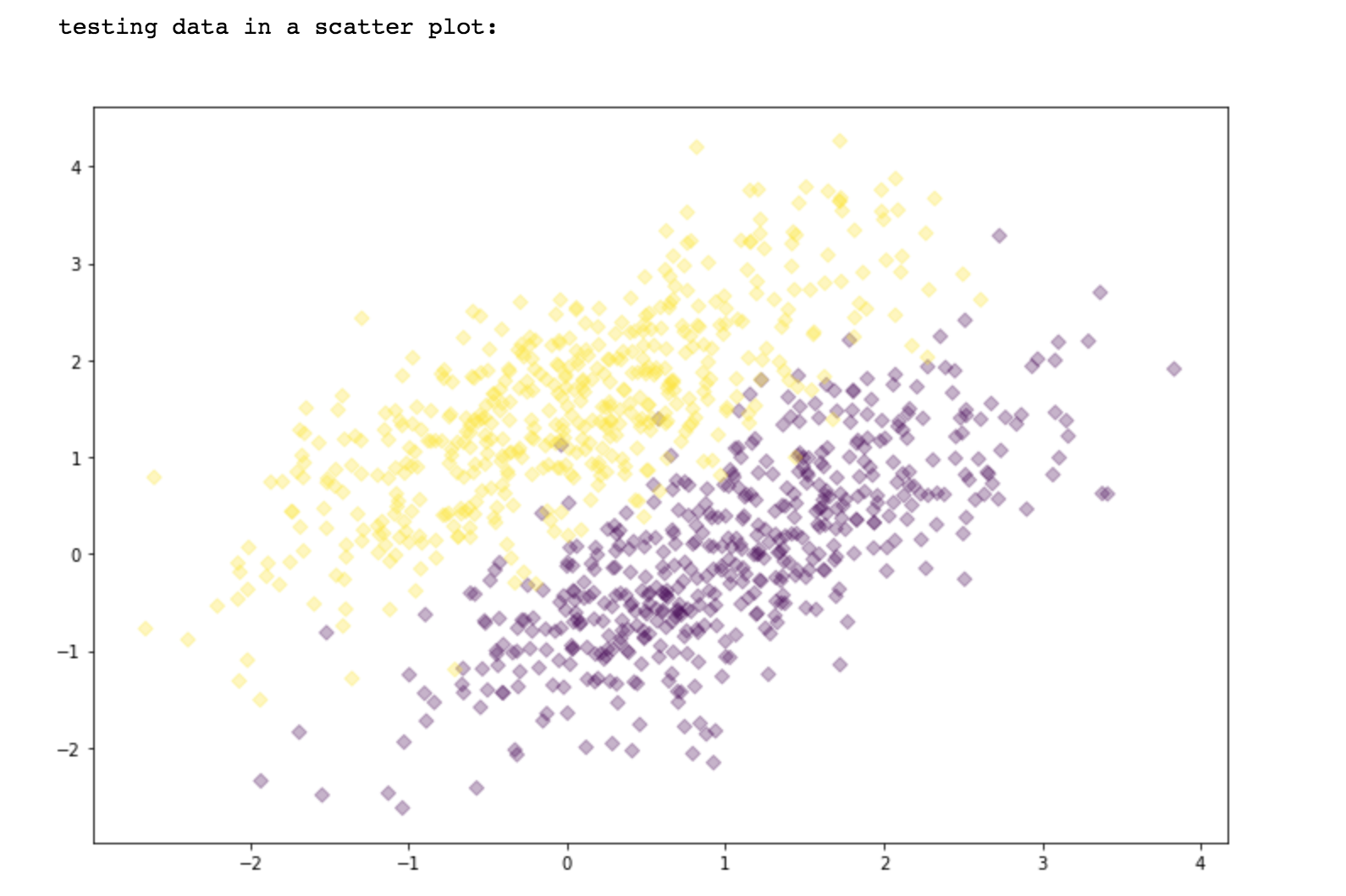
1. Using np.random.multivariate\_normal( ) function we obtain the data points for both the training and testing dataset.
2. Then stack all the generated points in vertical columns using vstack.
3. Preparing the intercept and slopes satisfy the property that the average predicted probability equals the observed prevalence of Y=1 in the dataset used to fit the model, thus we made an intercept in the data.
4. The logistic regression perceptron sets the weights randomly and then uses gradient descent to increase or decrease the weights depending upon the prediction.
5. We use the sigmoid function for the activation:
6. Cross entropy used as a Cost function:

Cost= −(ylog(p)+(1−y)log(1−p))−(ylog(p)+(1−y)log(1−p))

1. The predict() function uses the sigmoid function to obtain the weights for the testing data.
2. The plot\_ROC uses the sklearn to plot ROC and then calculate the AUC.

OUTPUT:





Cost Function:

0.634524531569

0.150308519043

0.125685280469

0.116700317831

0.112138736907

0.109446457726

0.107712575326

0.10653033404

.

.

.

0.102960436471

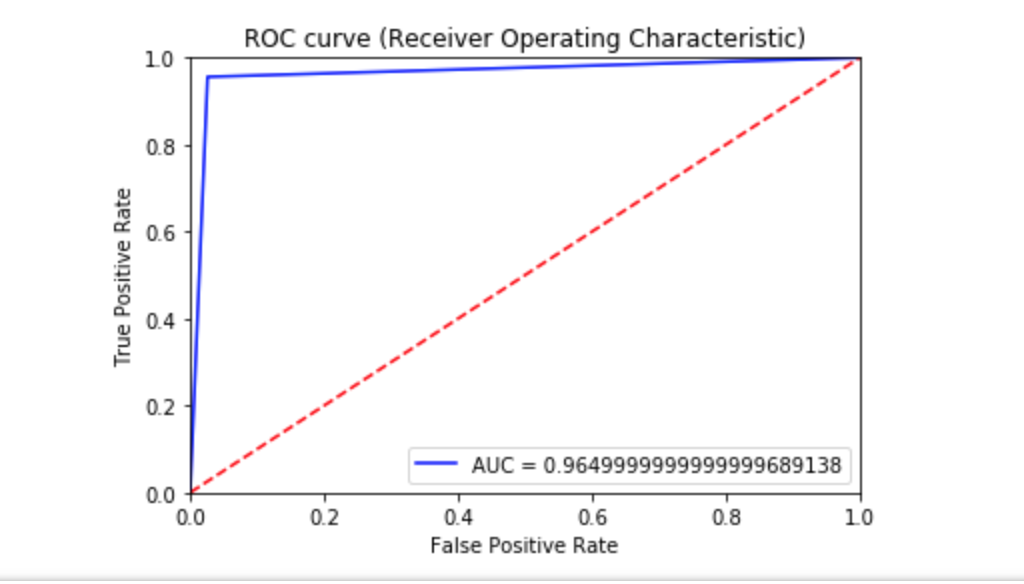
0.102947551917

0.102936759711

0.10292770596

Training Accuracy: 96.03333333333333

Testing Accuracy: 96.5



The output displays the training plotted points of both distributions and then the testing data points in a scatter plot.

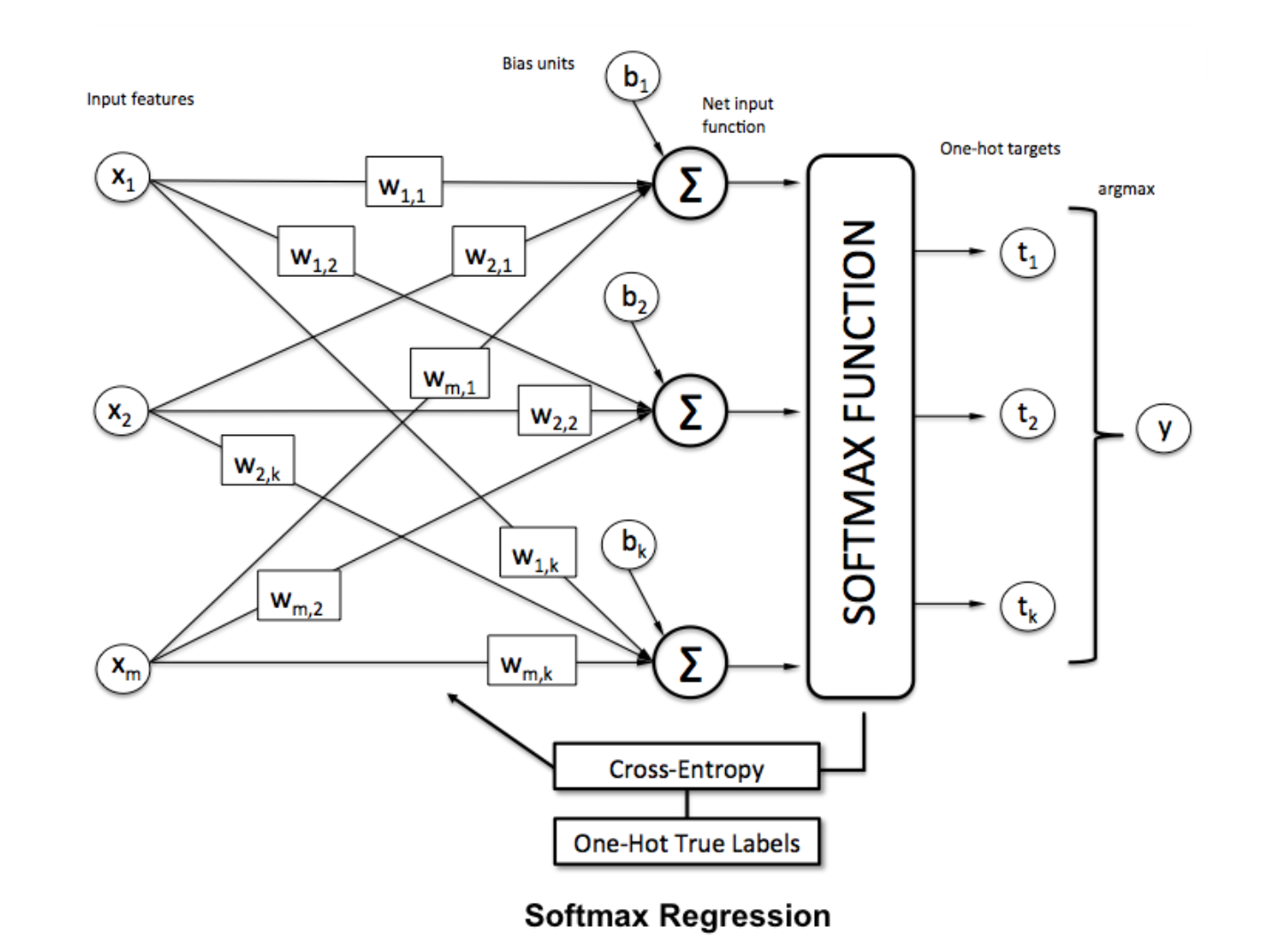
The cost value is displayed for every 100th record from all the 3000 data points in the training data.

The program then displays the calculated the training accuracy and testing accuracy and lots the ROC curve using sklearn and matplotlib.

Problem 2:

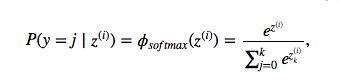
The problem 2 uses mxnet package to download the MNIST dataset.

Since the problem is a multiclass logistic regression the program uses softmax function as an activation function.

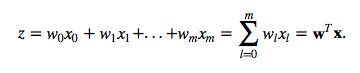


The softmax function computes the probability that this training sample x(i) belongs to class *j* given the weight and net input z(i).

The sigmoid function is replaced by the softmax function, which takes form:



Where the net input z is calculated by,



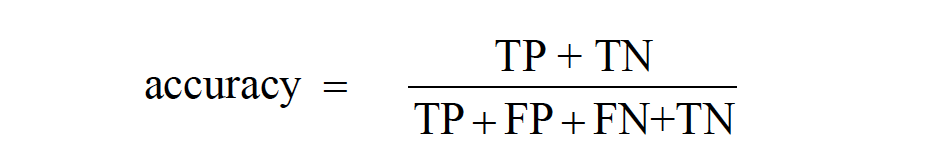
The program uses the cross-entropy cost function to calculate the error.

Cost= −(ylog(p)+(1−y)log(1−p))−(ylog(p)+(1−y)log(1−p))

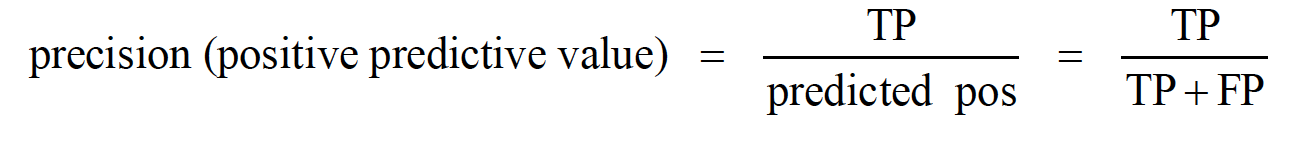
Then the program runs multiple epochs to form/learn a model on the training data. The epochs should be 10000 based on the question. But the algorithm takes a long time to run program, so for convenience’s sake we put epoch between 10-20.

The confusion matrix for all the matrix is calculated and then True positive, False positive, false negative are calculated. By using following formulae we get:

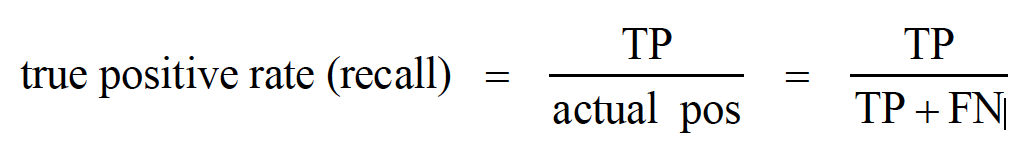
1.Accuracy:



2. precision:



3.Recall:



OUTPUT:

accuracy: 0.9014

Epoch 0. Loss: 0.408897111877 Train\_acc 0.899517 Test\_acc 0.9014

Epoch 1. Loss: 0.408897111964 Train\_acc 0.899517 Test\_acc 0.9014

Epoch 2. Loss: 0.408897111201 Train\_acc 0.899517 Test\_acc 0.9014

testing\_confusion\_matrix:

[[ 948 0 3 3 1 8 12 2 3 0]

[ 0 1092 7 4 0 5 3 2 22 0]

[ 9 9 881 33 6 4 20 12 47 11]

[ 1 1 10 947 2 11 0 12 22 4]

[ 3 3 6 5 880 2 21 4 10 48]

[ 13 1 4 81 3 698 17 9 59 7]

[ 6 1 6 2 11 14 907 2 7 2]

[ 1 6 21 21 7 1 0 928 4 39]

[ 9 7 12 54 6 16 11 11 838 10]

[ 7 6 0 20 26 6 2 31 16 895]]

sanity check: 10000

True\_Positive: [ 948 1092 881 947 880 698 907 928 838 895]

False\_Positive: [ 49 34 69 223 62 67 86 85 190 121]

False\_Negative: [ 34 45 153 65 104 196 53 102 138 116]

[ 0.96537678 0.96042216 0.85203095 0.93577075 0.89430894 0.78076063

0.94479167 0.90097087 0.85860656 0.88526212]

Final Recall:

0.897830142846

[ 0.95085256 0.96980462 0.92736842 0.80940171 0.93418259 0.9124183

0.91339376 0.91609082 0.8151751 0.88090551]

Final Precision:

0.902959338186

Above output shows that the program learns the model 3 times. Each time calculates the loss, training accuracy and testing accuracy.

The confusion matrix is calculated and displayed. Then a sanity check is performed to prove that the model learnt is right by printing the total of the matrix and comparing it with number of test cases.(sum is printed:10000)

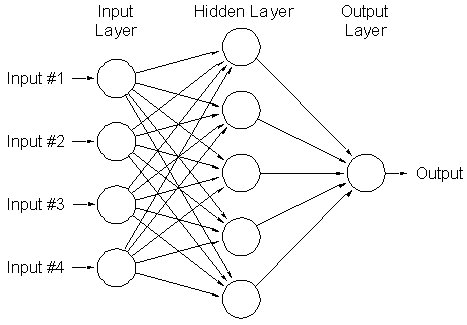
The the True positive, false positive, false\_negative are calculated for each class.

The recall and precision are calculated the last.

Problem 3:

The Artificial Neural Network is formed by stacking the perceptrons together to form a network.

Eg.



The ANN consists of input layer, hidden layer and an output layer.

Program logic:

The program has 100 hidden units and just one layer of hidden units. Each neuron receives the input weight from all the neurons in the input layer and a bias. Then each hidden unit add their own weight to the weights received from the input layer and forwards the weights to the output layer. The output layer then produces the class as an output. Then this prediction is compared to the actual class and based on the fact if the prediction is correct or wrong the all the weights to each layer are adjusted(back propagation.)

The ANN runs 10000 times based on the size of dataset for testing. Thus, the number of epochs should be 10000. But, 10000 epochs will run the program for too long. Thus, for the convenience’s sake we put the epochs=5.

OUTPUT:

Reading data

data read and cleaned

Epoch: 0

Epoch 0

Training Accuracy: 93.355

Epoch 0 : Testing Accuracy: 95.2 %

Epoch: 1

Epoch 1

Training Accuracy: 96.5066666667

Epoch 1 : Testing Accuracy: 95.96 %

Epoch: 2

Epoch 2

Training Accuracy: 97.2966666667

Epoch 2 : Testing Accuracy: 96.39 %

Epoch: 3

Epoch 3

Training Accuracy: 97.7866666667

Epoch 3 : Testing Accuracy: 96.75 %

Epoch: 4

Epoch 4

Training Accuracy: 98.075

Epoch 4 : Testing Accuracy: 96.94 %

Testing confuion matrix

[[ 970 1 0 1 0 1 2 3 0 2]

[ 0 1124 0 6 0 1 2 0 2 0]

[ 6 1 991 6 3 2 1 8 12 2]

[ 0 0 3 985 0 6 0 3 10 3]

[ 1 0 1 1 949 0 5 1 1 23]

[ 6 1 0 10 1 847 12 2 5 8]

[ 8 3 1 0 1 9 930 0 5 1]

[ 1 6 13 0 4 0 0 983 6 15]

[ 6 1 0 1 7 4 3 5 940 7]

[ 4 3 0 4 6 2 1 5 9 975]]

true positive:

[ 970 1124 991 985 949 847 930 983 940 975]

False Positive:

[32 16 18 29 22 25 26 27 50 61]

False Negative:

[ 980 1135 1032 1010 982 892 958 1028 974 1009]

True Negative:

[8988, 8849, 8950, 8961, 8996, 9083, 9016, 8945, 8976, 8930]

label precision recall

0.000000 0.968 0.990

1.000000 0.986 0.990

2.000000 0.982 0.960

3.000000 0.971 0.975

4.000000 0.977 0.966

5.000000 0.971 0.950

6.000000 0.973 0.971

7.000000 0.973 0.956

8.000000 0.949 0.965

9.000000 0.941 0.966

precision total: 0.969294826929

precision total: 0.968996360907

accuracy: 0.9694

The output prints the training accuracy and testing accuracy for each epoch

The prints the confusion matrix and print true positive, false negative, true negative, false positives based on that.

Then a table for each class and precision and recall is printed.

The precision total and accuracy is printed.

Comparing the outputs of problem 2 and problem 3:

1. The accuracy of ANN even at the 1st epoch of the implementation is more than that of the softmax perceptron. Even the increase in the accuracy at each iteration is more in ANN than in softmax perceptron.
2. Also the total precision in softmax regression is less than the precision of ANN.

References:

[1] <https://www.pugetsystems.com/labs/hpc/Machine-Learning-and-Data-Science-Multinomial-Multiclass-Logistic-Regression-1007/>

[2] <http://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html#cross-entropy>

[3] <https://www.kaggle.com/grfiv4/plot-a-confusion-matrix>

[4] <https://www.kdnuggets.com/2016/07/softmax-regression-related-logistic-regression.html>

[5] <https://en.wikipedia.org/wiki/Receiver_operating_characteristic#Area_under_the_curve>

[6] <https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/>