Shorabh Dhandharia
A20361514

HW3 – Discriminative Learning

Problem Statement

In this assignment, I have implemented different algorithms for discriminative learning. For logistic regression and multilayer perceptron, I have used "iris" and "mnist" dataset. For cross validation, I have used previous assignment's technique of 10-fold cross validation. I have used the Python's inbuilt function to compute and confusion matrix and calculated precision, recall, F-measure and accuracy manually. I have implemented the algorithm using stochastic gradient descent.

Proposed Solution

1. *2-Class Discrimination with Logistic Regression*
   Implement logistic regression for 2 class dataset and use non-linear combination of inputs to increase the capacity of classifier and evaluate the performance.
2. *K-Class Discrimination with Logistic Regression*
   Implement logistic regression for K class dataset and evaluate the performance.
3. *Two Layer Feedforward MLP for single output*
   Derive the backpropagation update equations for the weights of the outputs by minimizing the error function which is sum of mean squares. Compare results with the maximum likelihood estimation.
4. *Two Layer Feedforward MLP with 3-Class*
   Implement a two layer MLP for 3 class classification. Evaluate the performance of the algorithm as function of number of hidden units and as function of learning rate. Use momentum in the implementation and compare it with scikit-learn implementation.

Implementation Details

For Logistic Regression I am using the iris dataset which consists of 4 features – length and width of sepals and petals. The algorithm uses these features to classify to which of the three species – Iris Setosa, Iris virginica and Iris versicolor. I have also used the mnist dataset for classifying 2 and k classes.

For evaluation, I have also used non-linear features by transferring to higher dimension.

For multilayer perceptron, I am using iris and mnist dataset. The mnist dataset is a large dataset around 70000 samples of handwritten digits. Out of which I am extracting datasets for 2 and 3 classes.

All the algorithms were implemented using stochastic gradient descent with learning rate 0.0001 and iterations 100.

The Implementation of the code is almost similar in all the three algorithms as below:

- Reading the entire dataset into Z matrix
- Reshuffling the rows to perform cross validation correctly
- Split the matrix into X and Y
- Strip the data to get float values
- Binarize the Y data
- Apply Cross Validation
    - Divide the Data into Training and Test
    - Calculate indicator function of each class in Training data
    - Estimate the model parameters
    - Compute the discriminant function
    - Classify the examples and compute different measures

For Neural Network, the algorithm is:

- Follow above steps till cross validation
- Guess the no. of units, w and v
- Calculate Z using X and initial guess of w
- Calculate yhat using Z and initial guess of v
- Compute gradient for v and w
- Update w and v value

Results and Discussions

Below Tables Are Measures Calculated without Cross Validation –

Without cross validation, Precision, Recall and Accuracy comes 1.0 for 2-class but it comes around 0.85 for k-class since the dataset is very small. But as dataset increases, we can see the image dataset gives around 0.99 accuracy.

After transferring to higher dimension of degree 2 and 3 the results give almost 100% accuracy for 2-class discrimination.

Logistic Function:

2-Class	k-Class
"iris" Dataset Confusion Matrix: [[50 0] [0 50]] Precision is: [1.0, 1.0] Recall is: [1.0, 1.0] F-Measure is: [1.0, 1.0] Accuracy is: 1.0	"iris" Dataset Confusion Matrix: [[50 0 0] [1 34 15] [0 11 39]] Precision is: [1.0, 0.68, 0.78] Recall is: [0.98, 0.75, 0.72] F-Measure is: [0.99, 0.71, 0.75]

	Accuracy is: 0.82
"MNIST" Dataset Confusion Matrix: [[6873 30] [28 7849]] Precision is: [0.99, 0.99] Recall is: [0.99, 0.99] F-Measure is: [0.99, 0.99] Accuracy is: 0.9960	"MNIST" Dataset Confusion Matrix: [[6592 175 136] [152 7249 476] [190 312 6488]] Precision is: [0.95, 0.92, 0.93] Recall is: [0.95, 0.93, 0.91] F-Measure is: [0.95, 0.93, 0.92] Accuracy is: 0.93

Multilayer Perceptron:

In case of Multilayer Perceptron, using iris dataset I got 0.94 of accuracy whereas for MNIST dataset I got around 0.89 of accuracy. Even using the scikit-learn implementation I am getting almost same accuracy of 0.95 and 0.89.

"iris" Dataset Confusion Matrix: [[50 0 0] [0 45 5] [2 1 47]] Precision is: [1.0, 1.0, 0.71] Recall is: [0.96, 0.98, 0.90] F-Measure is: [0.98, 0.93, 0.92] Accuracy is: 0.94	"MNIST" Dataset Confusion Matrix: [[6206 418 279] [289 7168 420] [335 469 6186]] Precision is: [0.89, 0.90, 0.88] Recall is: [0.90, 0.88, 0.89] F-Measure is: [0.90, 0.89, 0.89] Accuracy is: 0.89

MNIST dataset contains 14780 samples for 2 classes and 21770 samples for 3 classes. It contains around 784 features.

Below Tables Are Measures Calculated with Cross Validation –

After applying cross validation, the size of test data comes down to 10 and hence the algorithm almost 100% of the time classifies correctly for 2 class. And hence the Precision, Recall and Accuracy almost comes to 1.0 in case of k-class it gives an accuracy of around 0.87

Logistic regression:

2-Class	k-Class
Confusion Matrix: [[7 0] [0 3]] Precision is: [1.0, 1.0] Recall is: [1.0, 1.0] F-Measure is: [1.0, 1.0] Accuracy is: 1.0	Confusion Matrix: [[9 0 0] [0 3 1] [0 1 1]] Precision is: [1.0, 0.75, 0.5] Recall is: [1.0, 0.75, 0.5] F-Measure is: [1.0, 0.75, 0.5] Accuracy is: 0.866666666667

Multilayer Perceptron:

In case of Multilayer Perceptron, using iris dataset I got 0.87 of accuracy. Using the scikit-learn implementation I am getting accuracy of 0.89.

Confusion Matrix: [[5 0 0] [0 3 0] [0 2 5]] Precision is: [1.0, 1.0, 0.71] Recall is: [1.0, 0.6, 1] F-Measure is: [1.0, 0.75, 0.83] Accuracy is: 0.86

In all the above experiments, I tested the implementation by changing the no. of iterations and learning rate. I am getting all these values for learning rate of 0.0001 as mentioned above but when I increase the learning rate, I do not get the desired output which results in “not a number” error.

Note:

The derivation document is placed along with this report in the same folder.

All the codes have been executed and evaluated for the datasets mentioned above.
To run the code successfully you need to change the file name and location in all the codes, references are provided below.
Logistic2Class.py:6, LogisticKClass.py:6, NeuralNetwork.py:6,