CS 584 – Machine Learning

Assignment 3

Report

1) Logistic Regression:

Data sets used: Iris

Image datasets : Digits and Image segmentation data set(from UCI link provided below)

a) Logistic Regression for 2 class classification:

Activation function used: Sigmoid

Results:

Performed 10 fold cross validation and below are the results :

Confusion matrix(First 5)

```
Predicted  0  1
Truth
0          5  0
1          0  5
Predicted  0  1
Truth
0          4  0
1          0  6
Predicted  0  1
Truth
0          5  0
1          0  5
Predicted  0  1
Truth
0          6  0
1          0  4
Predicted  0  1
Truth
0          3  0
1          0  7

################################
Average of the model parameters
################################
Error rate: 0.0
```

```
Accuracy: 1.0
Precision: [ 1.  1.]
Recall: [ 1.  1.]
F-measure: [ 1.  1.]
```

Since the classes 1 and 3 of iris dataset are linearly separable we got 100% accuracy.

The above setup is run with the below configurations:

Method to calculate parameters: Stochastic gradient descent
Number of iterations: 500
Learning rate: 0.001
Initial thetas:0.01

b) <u>Logistic regression of 2-class classification using non-linear combinations:</u>

Non-linear combinations are obtained by using degree 2 polynomial on the features.

Below are the results:

Confusion matrix (First 3)

```
Predicted  0  1
Truth
0          5  0
1          0  5
Predicted  0  1
Truth
0          4  0
1          0  6
Predicted  0  1
Truth
0          5  0
1          0  5
Predicted  0  1
Truth
0          6  0
1          0  4

################################
Average of the model parameters
################################
Error rate: 0.0
Accuracy: 1.0
Precision: [ 1.  1.]
Recall: [ 1.  1.]
F-measure: [ 1.  1.]
```

Experiment is run with the same configuration as the above.

c) Logistic Regression for k-class classification:

Logistic regression is implemented for k-class discrimination using softmax as the activation function and the labels for the data points are obtained by using the below formula.

$Y^{\wedge} = \text{argmax}_j(\text{softmax}_j)$

d) Results:

Confusion matrix(First 3):

```
Predicted  0  1  2
Truth
0          6  0  0
1          0  4  0
2          0  0  5
Predicted  0  1  2
Truth
0          6  0  0
1          0  4  0
2          0  0  5
Predicted  0  1  2
Truth
0          6  0  0
1          0  6  0
2          0  0  3
Predicted  0  1  2
Truth
0          3  0  0
1          0  3  1
2          0  0  8

################################
Average of the model parameters
################################
Error rate: 0.04
Accuracy: 0.96
Precision: [ 1.          0.88238095  1.        ]
Recall: [ 1.          1.          0.88376984]
F-measure: [ 1.          0.93444333  0.93436185]
```

The above setup is run with the below configurations:

Method to calculate parameters: Stochastic gradient descent
Number of iterations: 500
Learning rate: 0.001

Initial thetas:0.01 (for all the classes)

From the above results we can conclude that the classifier is doing good.

2) Multilayer perceptron

b) 2-layer perceptron is built and following are the results:

The experiment is run with 4 hidden layers which is an optimum number to start with, random integers close to 0 using normal distribution.

Stochastic gradient descent is used to update the parameters w and v. Learning rate is chosen to be 0.1 and the number of times the training procedure repeated to ensure minimal error is 500.

Also in order to prevent the weights from varying much momentum is implemented to the input weights.

Results:

Confusion matrix

```
Predicted   0  1   2
Truth
0          12  0   0
1           0  8   0
2           0  0  10
Predicted  0  1   2
Truth
0          9  0   0
1          0  9   1
2          0  0  11
Predicted  0  1   2
Truth
0          8  0   0
1          0  9   2
2          0  0  11
Predicted  0   1  2
Truth
0          8   0  0
1          0  12  1
2          0   0  9
Predicted   0  1  2
Truth
0          13  0  0
1           0  6  2
2           0  0  9
```

```
################################
Average of the model parameters
################################
Error rate: 0.04
Accuracy: 0.96
Precision: [ 1.          0.87825175  1.        ]
Recall: [ 1.          1.          0.89620047]
F-measure: [ 1.          0.93290226  0.94411137]
```

Below are the results by running the experiment with various settings. Only a few parameters are considered and the behaviors of the model to various parameter inputs are tabulated below:

No. Hidden units	Learning Rate		Initial Parameters		Accuracy			
			Uniform	Normal	L1(v)I1	L1(v)I2	L2(v)I1	L2(v)I2
4	0.1	0.05	Range(0.0,0.5)	Range(0.0,0.5)	0.926	0.96	0.95	0.88
6	0.1	0.05	Range(0.0,0.5)	Range(0.0,0.5)	0.95	0.94	0.86	0.90
10	0.1	0.05	Range(0.0,0.5)	Range(0.0,0.5)	0.89	0.94	0.95	0.9
50	0.1	0.05	Range(0.0,0.5)	Range(0.0,0.5)	0.92	0.91	0.92	0.906

All the experiments are run using a fixed number of training repetitions and the accuracies are obtained. Though the accuracies don't change much but the performance is a bit low when considering a very less learning rate.

Also, accuracies change if a set of random initial parameters are generated and by increasing the number of iterations.

Image data set:

The afore-mentioned dataset is obtained from UCI repository and can be found at the below link:
https://archive.ics.uci.edu/ml/datasets/Image+Segmentation

The data is normalized since the data is not uniform throughout.

The results are obtained by iterating the training process for at least 500 times. In order to prevent over-fitting, the training process shall be stopped even before the convergence or by randomly muting the hidden layer's outputs(by setting 0s to the outputs)

Following are the results obtained from the image data:

```
Predicted  0  1  2  3  4  5  6
Truth
0          4  0  0  0  0  0  0
1          0  8  0  0  0  0  0
2          0  0  6  0  6  0  0
3          0  0  0  2  2  0  0
```

```
4           0   0   0   0   3   0    0
5           0   0   0   0   0   1    0
6           0   0   0   0   0   0   10
Predicted   0   1   2   3   4   5   6
Truth
0           6   0   0   0   0   0   0
1           0   4   0   0   0   0   0
2           0   0   2   0   1   0   0
3           0   0   0   5   1   1   0
4           0   0   0   1   8   0   0
5           0   0   0   0   0   9   0
6           0   0   0   0   0   0   4
Predicted   0   1   2   3    4   5   6
Truth
0           5   0   0   0    0   0   0
1           0   4   0   0    0   0   0
2           0   0   3   0    1   0   0
3           0   0   0   8    0   1   0
4           0   0   0   0   10   0   0
5           0   0   0   0    0   7   0
6           0   0   0   0    0   0   3
Predicted   0   1   2   3   4   5   6
Truth
0           9   0   0   0   0   0   0
1           0   4   0   0   0   0   0
2           0   0   6   1   0   0   0
3           0   0   0   5   0   0   0
4           0   0   2   0   1   0   0
5           0   0   0   0   0   8   0
6           0   0   0   0   0   0   6
Predicted   0    1   2   3   4   5   6
Truth
0           6    0   0   0   0   0   0
1           0   10   0   0   0   0   0
2           0    0   3   0   0   0   1
3           0    0   0   5   0   0   0
4           0    0   0   0   5   0   0
5           0    0   0   0   0   5   0
6           0    0   0   0   0   0   7
################################
Average of the model parameters
################################
Error rate: 0.0857142857143
Accuracy: 0.914285714286
Precision: [ 38.8  42.4  21.2  30.6  49.8  47.2  43.4]
Recall: [ 1.          1.          0.95        0.93333333  0.79636364
0.955       0.975     ]
F-measure: [ 1.9345863   1.91929752  1.7245738   1.74711019  1.37029953
1.67951257
  1.86631477]
```

Predicted	0	1	2	3	4	5	6	7	8	9
Truth										
0	11	0	0	0	0	0	0	0	0	0
1	0	20	0	0	0	0	0	0	0	0
2	0	0	18	0	0	0	0	0	0	0
3	0	0	0	16	0	0	0	0	0	0
4	0	0	0	0	26	0	0	0	0	0
5	0	0	0	0	0	27	0	0	0	0
6	0	0	0	0	0	0	15	0	0	0
7	0	0	0	0	0	0	0	11	0	0
8	0	0	0	0	0	0	0	0	19	0
9	0	0	0	0	0	0	0	0	0	17

Predicted	0	1	2	3	4	5	6	7	8	9
Truth										
0	19	0	0	0	0	0	0	0	0	0
1	0	20	0	0	0	0	0	0	0	0
2	0	0	20	0	0	0	0	0	0	0
3	0	0	0	13	0	0	0	0	0	0
4	0	0	0	0	14	0	0	0	0	0
5	0	0	0	0	0	17	0	0	0	0
6	0	0	0	0	0	0	16	0	0	0
7	0	0	0	0	0	0	0	17	0	0
8	0	0	0	0	0	0	0	0	19	0
9	0	0	0	0	0	0	0	0	0	25

Predicted	0	1	2	3	4	5	6	7	8	9
Truth										
0	17	0	0	0	0	0	0	0	0	0
1	0	21	0	0	0	0	0	0	0	0
2	0	0	13	0	0	0	0	0	0	0
3	0	0	0	19	0	0	0	0	0	0
4	0	0	0	0	15	0	0	0	0	0
5	0	0	0	0	0	15	0	0	0	0
6	0	0	0	0	0	0	22	0	0	0
7	0	0	0	0	0	0	0	21	0	0
8	0	0	0	0	0	0	0	0	13	0
9	0	0	0	0	0	0	0	0	0	24

Predicted	0	1	2	3	4	5	6	7	8	9
Truth										
0	17	0	0	0	0	0	0	0	0	0
1	0	18	0	0	0	0	0	0	0	0
2	0	0	26	0	0	0	0	0	0	0
3	0	0	0	29	0	0	0	0	0	0
4	0	0	0	0	12	0	0	0	0	0

```
5            0    0    0    0    0   13    0    0    0    0
6            0    0    0    0    0    0   17    0    0    0
7            0    0    0    0    0    0    0   15    0    0
8            0    0    0    0    0    0    0    0   16    0
9            0    0    0    0    0    0    0    0    0   17
Predicted    0    1    2    3    4    5    6    7    8    9
Truth
0           19    0    0    0    0    0    0    0    0    0
1            0   18    0    0    0    0    0    0    0    0
2            0    0   13    0    0    0    0    0    0    0
3            0    0    0   16    0    0    0    0    0    0
4            0    0    0    0   18    0    0    0    0    0
5            0    0    0    0    0   15    0    0    0    0
6            0    0    0    0    0    0   19    0    0    0
7            0    0    0    0    0    0    0   16    0    0
8            0    0    0    0    0    0    0    0   26    0
9            0    0    0    0    0    0    0    0    0   20
```

```
################################
Average of the model parameters
################################
Error rate: 0.0
Accuracy: 1.0
Precision: [ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
Recall: [ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
F-measure: [ 1.  1.  1.  1.  1.  1.  1.  1.  1.  1.]
```

2 a) Solution below:

Back propagation equations for single output MLP

Given:-

Error function = $E(V, \{w_j\}) = \frac{1}{2}\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2$

and $\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$ and hidden layer and output layer uses sigmoid function as activator.

Solution:-

We know, $\hat{y}$ = sigmoid $(V^T z)$ and

d/dx (sigmoid (x)) = sigmoid (x) $(1 - \text{sigmoid}(x))$

Our objective is to find the back propagation eqns.

ie $\frac{\partial E}{\partial V}$ and $\frac{\partial E}{\partial w}$

$\frac{\partial E}{\partial V} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V}$ (By chain rule). $\rightarrow$ ①

$\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_j}$ (By chain rule) $\rightarrow$ ②

Solve ①.

$\frac{\partial E}{\partial V} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial V}$, $\frac{\partial E}{\partial \hat{y}} = 2 \cdot \frac{1}{2}\sum_{i=1}^{m} y^{(i)} - \hat{y}^{(i)}(-1)$

and $\frac{\partial \hat{y}}{\partial V} = \hat{y}^{(i)}(1 - \hat{y}^{(i)}) z$.

So. $\frac{\partial E}{\partial V} = -\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) \cdot \hat{y}^{(i)}(1 - \hat{y}^{(i)}) z$.

By applying Gradient descent, we have

$$V \leftarrow V + \eta \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) \hat{y}^{(i)}(1 - \hat{y}^{(i)}) z^{(i)}$$

Illy, $\frac{\partial E}{\partial w_j} = \frac{\partial E}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z_j} \cdot \frac{\partial z_j}{\partial w_j}$

$\frac{\partial E}{\partial \hat{y}} = \cancel{\alpha} \frac{1}{\cancel{2}} \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})(-1)$

$\frac{\partial E}{\partial z_j} = \hat{y}(1-\hat{y}) v$

$\frac{\partial E}{\partial w_j} = z_j (1-z_j) x.$

So, $\frac{\partial E}{\partial w_j} = -\sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)}) (\hat{y}^{(i)}(1-\hat{y}^{(i)})) v_j \cdot z_j(1-z_j) x^{(i)}$

By Gradient descent, we have

$$w_j \leftarrow w_j + \eta \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})\ \hat{y}^{(i)}(1-\hat{y}^{(i)}) \cdot v_j\ z_j\ (1-z_j)\ x^{(i)}$$