Programming Assignment #1
IITM-CS4011 : Principles of Machine Learning

EE15B025
Ganga Meghanath

September 12, 2017

1 Logistic Regression

Logistic Regression is performed on forest and mountain classes in DS2. The normalised and appropriately arranged train and test datasets are stored in Dataset folder.

The results obtained are as follows :

(per class)	Class1 : Forest	Class2 : Mountain
Precision	0.909090909091	1.0
Recall	1.0	0.9
F-Measure	0.952380952381	0.947368421053

$$Accuracy = 0.95$$

Correct Labels : [-1. 1. -1. 1. -1. 1. 1. -1. -1. -1. 1. 1. -1. -1. -1. -1. 1. 1. 1. 1. 1. 1. -1. 1. -1. -1. 1. -1. 1. 1. 1. 1. -1. -1. -1. -1. 1. 1. -1. -1.]

Predicted Labels : [-1. 1. -1. -1. -1. 1. 1. -1. -1. -1. 1. 1. -1. -1. -1. -1. 1. 1. 1. 1. 1. -1. -1. 1. -1. -1. 1. -1. 1. 1. 1. 1. -1. -1. -1. -1. 1. 1. -1. -1.]

1.1 L1 Logistic Regression

On performing L1 regularised Logistic Regression (by Boyds Group (http: //www.stanford.edu/~boyd/l1_l
on the same dataset ,we observe the following results,

For $\lambda = 0.01$:

Precision for class Forest : 1.0 and Mountain : 0.952380952381

Recall for class Forest : 0.95 and Mountain : 1.0

F-measure for class Forest : 0.974358974359 and Mountain : 0.975609756098

$$Accuracy : 0.975$$

Correct Labels : [-1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]

Predicted Labels : [-1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. -1. 1. -1. -1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]

Train :

```
ganga@ganga-GL502VMK:~/Documents/Academics/5th Semester/CS4011 Machine Learning/Assignments/l1_logreg-0.8.2-i686-pc-linux-gnu$ ./l1_logreg_train -s Train_features Train_labels 0.01 model

Reading data...

Problem summary:
    [feature matrix]        dense matrix of (518 examples x 96 features)
    [standardization]       yes
    [lambda_max]            0.235783
    [lambda]                0.01        (ratio = 0.0424)

Running solver...
    NT iter          gap        primal obj
         0       5.8802e-01    6.9016e-01
         1       2.5855e-01    4.1536e-01
         2       1.9368e-01    3.8217e-01
         3       1.3354e-01    3.5961e-01
         4       8.7592e-02    3.4795e-01
         5       5.1793e-02    3.4131e-01
         6       2.8098e-02    3.3697e-01
         7       1.4894e-02    3.3443e-01
         8       7.8812e-03    3.3296e-01
         9       4.2447e-03    3.3222e-01
        10       2.2974e-03    3.3181e-01
        11       1.2490e-03    3.3159e-01
        12       6.7853e-04    3.3147e-01
        13       3.6810e-04    3.3141e-01
        14       1.9910e-04    3.3137e-01
        15       1.0744e-04    3.3136e-01
        16       5.7880e-05    3.3135e-01
        17       3.1164e-05    3.3134e-01
        18       1.6759e-05    3.3134e-01
        19       9.0107e-06    3.3134e-01
        20       4.8462e-06    3.3133e-01
        21       2.6067e-06    3.3133e-01
        22       1.4040e-06    3.3133e-01
        23       7.7995e-07    3.3133e-01
        24       4.3741e-07    3.3133e-01
        25       2.4451e-07    3.3133e-01
        26       1.3690e-07    3.3133e-01
        27       7.6631e-08    3.3133e-01
        28       4.2914e-08    3.3133e-01
        29       2.4038e-08    3.3133e-01
        30       1.3469e-08    3.3133e-01
        31       7.5495e-09    3.3133e-01
Solution found.

    [NT  iterations]            31

Timing:
    [read data]                0.02 (sec)
    [solve problem]            0.04 (sec)
    [write solution]              0 (sec)
    [total time]               0.06 (sec)
```

Test :

```
ganga@ganga-GL502VMK:~/Documents/Academics/5th Semester/CS4011 Machine Learning/Assignments/l1_logreg-0.8.2-i686-pc-linux-gnu$ ./l1_logreg_classify -t Test_labels model Test_features results

Reading data...

Problem summary:
    [feature matrix]        dense matrix of (40 examples x 96 features)
    [mode]                  classify and test

Running classifier...

Classification result:
    [right predic. count]   39
    [wrong predic. count]   1
    [test error]            1 / 40 = 0.025

Timing:
    [read data]                  0 (sec)
    [solve problem]              0 (sec)
    [write solution]             0 (sec)
    [total time]                 0 (sec)
```

Prediction :

```
ganga@ganga-GL502VMK:~/Documents/Academics/5th Semester/CS4011 Machine Learning/Assignments/EE15B025_PA1b/Code/q7/l1_logreg-0.8.2-i686-pc-linux-gnu$ ./l1_logreg_classify model Test_features test_predicti
on

Reading data...

Problem summary:
    [feature matrix]        dense matrix of (40 examples x 96 features)
    [mode]                  classify only

Running classifier...

Classification result:
    [positive class count]  21
    [negative class count]  19

Timing:
    [read data]                  0 (sec)
    [solve problem]              0 (sec)
    [write solution]             0 (sec)
    [total time]                 0 (sec)
```

1.2 L2 Logistic Regression

$\lambda = 0.01$

Precision for class Forest : 0.9 and Mountain : 0.9

Recall for class Forest : 0.9 and Mountain : 0.9
F-measure for class Forest : 0.9 and Mountain : 0.9

$$Accuracy = 0.9$$

Correct Labels : [-1. 1. -1. 1. -1. 1. 1. -1. -1. -1. 1. 1. -1. -1. -1. -1. 1. 1. 1. 1. 1. 1. -1. 1. -1. -1. 1. -1. 1. 1. 1. 1. -1. -1. -1. -1. 1. 1. -1. -1.]
Predicted Labels : [-1. 1. -1. 1. -1. 1. 1. 1. -1. -1. 1. 1. 1. -1. -1. -1. 1. 1. 1. 1. 1. -1. -1. 1. -1. -1. -1. -1. 1. 1. 1. 1. -1. -1. -1. -1. 1. 1. -1. -1.]

Inference :

For the considered dataset, L1 regularised Logistic Regression is found to give more accuracy (for λ= 0.01).

2 Backpropagation

Backpropagation is a method used in artificial neural networks to calculate the error contribution of each neuron after a batch of data (in image recognition, multiple images) is processed. This is used by an enveloping optimization algorithm to adjust the weight of each neuron, completing the learning process for that case.

In the below results, we have implemented a 3 layered neural network having an input layer, output layer and a hidden layer. The activation function made use of is sigmoid and softmax has been used as the output activation function inorder to obtain a probabilistic estimation of the prediction of the class. The output layer has 4 neurons which give the probability of the input image belonging to the 4 classes. The labels used for training and testing are one-hot encoded. The training and test data have been normalised before use. Depending on the kind of loss function we use for learning, the results obtained are portrayed below :

Note : Regularisation has been implemented in both cases inorder to minimise overfitting. Random samples have been chosen to create batches during training.

2.1 Cross Entropy

2.1.1 Loss Function :

$$L(\theta) = -\sum_{i=1}^{m}\sum_{j=1}^{K} y_{ij} log(\hat{y_{ij}}) + \lambda(\sum_{i=1}^{m}\sum_{j=1}^{K^1}(W_{ij}^1)^2 + \sum_{i=1}^{m}\sum_{j=1}^{K^2}(W_{ij}^2)^2)$$

Forward Propagation :

$$a^{(1)} \quad = \quad X; add(x_0)$$

$$z^{(1)} = a^{(1)} * (W^{(1)})^T$$

$$a^{(2)} = g(z^{(2)}); add(a_0^{(2)})$$

$$z^{(3)} = a^{(2)} * (W^{(2)})^T$$

$$\hat{y} = f(z^{(3)})$$

where,
$g(x) = sigmoid(x) = \dfrac{1}{1+e^{-x}}$
$f(x_i) = softmax(x_i) = \dfrac{e^{x_i}}{\displaystyle\sum_{j=1}^{K} e^{x_j}}$

Backpropagation

Gradient Calculation :

$$\delta^{(3)} = \hat{y} - y$$

$$\delta^{(2)} = (\delta^{(3)})^T * a^{(1)} \odot g^{'}(z^{(2)})$$

$$\Delta^{(1)} = \frac{(\delta^{(2)})^T * a^{(1)}}{m} + \lambda W^{(1)}$$

$$\Delta^{(2)} = \frac{(\delta^{(3)})^T * a^{(2)}}{m} + \lambda W^{(1)}$$

Update Rule :

$$W^{(1)} = W^{(1)} - \alpha \Delta^{(1)}$$

$$W^{(2)} = W^{(2)} - \alpha \Delta^{(2)}$$

Results Obtained

Testing1 : 1000 hidden units, $\alpha = 0.1$

$$\text{Running for lambda} = 0.01 \text{ Accuracy} : 0.5625$$

$$\text{Running for lambda} = 0.1 \text{ Accuracy} : 0.575$$

$$\text{Running for lambda} = 1 \text{ Accuracy} : 0.5625$$

$$\text{Running for lambda} = 10 \text{ Accuracy} : 0.55$$

$$\text{Running for lambda} = 100 \text{ Accuracy} : 0.25$$

Testing2 : Hidden Layer Size = 1000 and $\alpha = 0.1$

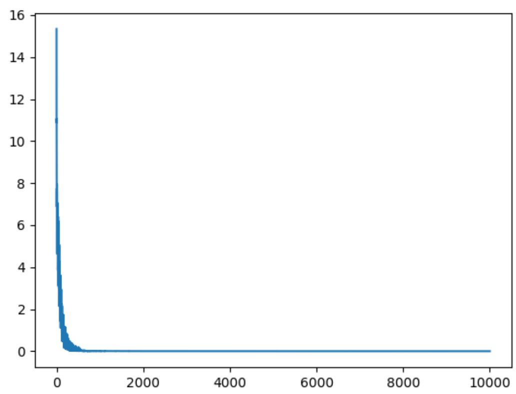
- Running for lambda = 0

$$\text{Accuracy} : 0.55$$

$$\text{Precision} : \text{Mountain} = 0.45, \text{Forest} = 0.727272727273, \text{Coast} = 0.47619047619, \text{Insidecity} = 0.529411764706$$

$$\text{F-measure} : \text{Mountain} = 0.45, \text{Forest} = 0.761904761905, \text{Coast} = 0.487804878049, \text{Insidecity} = 0.486486486486$$

$$\text{Recall} : \text{Mountain} = 0.45, \text{Forest} = 0.8, \text{Coast} = 0.5, \text{Insidecity} = 0.45$$



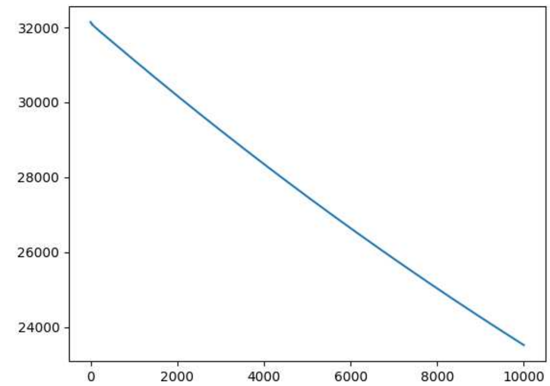
- Running for lambda = 0.01

$$\text{Accuracy} : 0.4875$$

$$\text{Precision} : \text{Mountain} = 0.5, \text{Forest} = 0.541666666667, \text{Coast} = 0.434782608696, \text{Insidecity} = 0.444444444444$$

F-measure : Mountain = 0.545454545455, Forest = 0.590909090909, Coast = 0.46511627907, Insidecity = 0.275862068966

Recall : Mountain = 0.6, Forest = 0.65, Coast = 0.5, Insidecity = 0.2



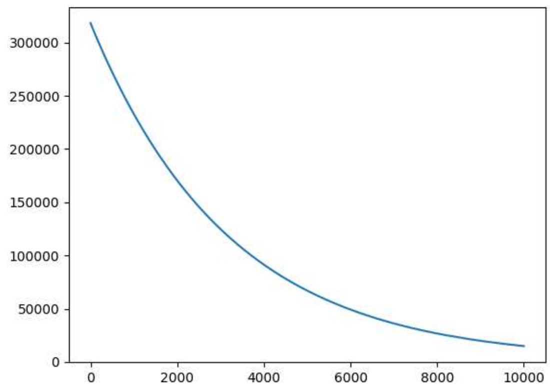
- Running for lambda = 0.1

Accuracy : 0.7125

Precision : Mountain = 0.53125, Forest = 1.0, Coast = 0.75, Insidecity = 0.818181818182

F-measure : Mountain = 0.653846153846, Forest = 0.787878787879, Coast = 0.818181818182, Insidecity = 0.58064516129

Recall : Mountain = 0.85, Forest = 0.65, Coast = 0.9, Insidecity = 0.45



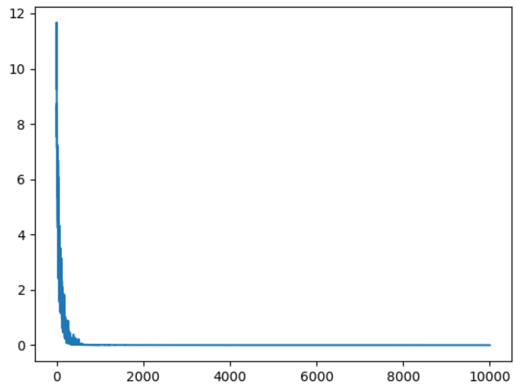
- Running for lambda = 1

$$\text{Accuracy}: \ 0.6125$$

$$\text{Precision}: \ \text{Mountain} = 0.666666666667, \text{Forest} = 0.75, \text{Coast} = 0.5, \text{Insidecity} = 0.636363636364$$

$$\text{F-measure}: \ \text{Mountain} = 0.682926829268, \text{Forest} = 0.666666666667, \text{Coast} = 0.615384615385, \text{Insidecity} = 0.451612903226$$

$$\text{Recall}: \ \text{Mountain} = 0.7, \text{Forest} = 0.6, \text{Coast} = 0.8, \text{Insidecity} = 0.35$$



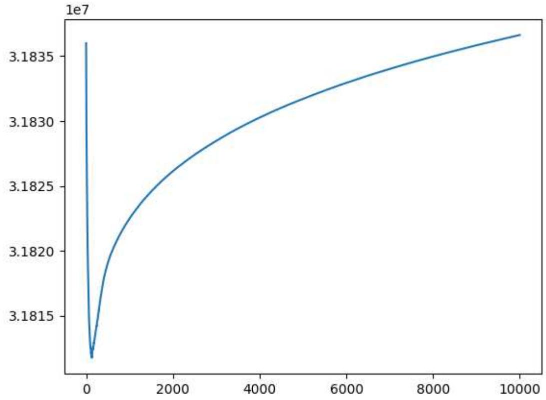
- Running for lambda = 10

$$\text{Accuracy}: \ 0.5$$

$$\text{Precision}: \ \text{Mountain} = 0.333333333333, \text{Forest} = 0.652173913043, \text{Coast} = 0.5, \text{Insidecity} = 0.466666666667$$

$$\text{F-measure}: \ \text{Mountain} = 0.315789473684, \text{Forest} = 0.697674418605, \text{Coast} = 0.545454545455, \text{Insidecity} = 0.4$$

$$\text{Recall}: \ \text{Mountain} = 0.3, \text{Forest} = 0.75, \text{Coast} = 0.6, \text{Insidecity} = 0.35$$



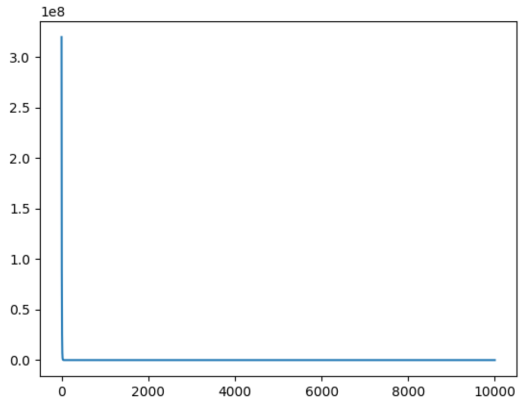
- Running for lambda = 100

$$\text{Accuracy : } 0.25$$

$$\text{Precision : Mountain} = 0.0, \text{ Forest} = 0.0, \text{ Coast} = 0.0, \text{ Insidecity} = 0.25$$

$$\text{F-measure : Mountain} = 0.0, \text{ Forest} = 0.0, \text{ Coast} = 0.0, \text{ Insidecity} = 0.4$$

$$\text{Recall : Mountain} = 0.0, \text{ Forest} = 0.0, \text{ Coast} = 0.0, \text{ Insidecity} = 1.0$$



Maximum Accuracy Obtained = 71.25% for $\lambda = 0.1$

2.1.2 Loss Function :

$$L(\theta) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{K}(y_{ij} - f_j(x_i))^2 + \lambda(\sum_{i=1}^{m}\sum_{j=1}^{K^1}(W_{ij}^1)^2 + \sum_{i=1}^{m}\sum_{j=1}^{K^2}(W_{ij}^2)^2)$$

Forward Propagation :

$$\begin{aligned}
a^{(1)} &= X; add(x_0) \\
z^{(1)} &= a^{(1)} * (W^{(1)})^T \\
a^{(2)} &= g(z^{(2)}); add(a_0^{(2)}) \\
z^{(3)} &= a^{(2)} * (W^{(2)})^T \\
\hat{y} &= f(z^{(3)})
\end{aligned}$$

where,
$g(x) = sigmoid(x) = \dfrac{1}{1+e^{-x}}$
$f(x_i) = softmax(x_i) = \dfrac{e^{x_i}}{\displaystyle\sum_{j=1}^{K} e^{x_j}}$

Backpropagation

Gradient Calculation :

$$\begin{aligned}
\delta^{(3)} &= (\hat{y} - y) \odot \hat{y} \odot (1 - \hat{y}) \\
\delta^{(2)} &= (\delta^{(3)})^T * a^{(1)} \odot g^{'}(z^{(2)}) \\
\Delta^{(1)} &= \frac{(\delta^{(2)})^T * a^{(1)}}{m} + \lambda W^{(1)} \\
\Delta^{(2)} &= \tfrac{(\delta^{(3)})^T * a^{(2)}}{m} \quad + \lambda W^{(1)}
\end{aligned}$$

Update Rule :

$$\begin{aligned}
W^{(1)} &= W^{(1)} - \alpha \Delta^{(1)} \\
W^{(2)} &= W^{(2)} - \alpha \Delta^{(2)}
\end{aligned}$$

Results Obtained

Testing1 : Hidden layer units = 45

- These are for $\alpha = 0.1$

$$\text{Running for lambda} = 0 \text{ Accuracy} : 0.525$$

$$\text{Running for lambda} = 0 \text{ Accuracy} : 0.6$$

$$\text{Running for lambda} = 0.01 \text{ Accuracy} : 0.5625$$

$$\text{Running for lambda} = 0.1 \text{ Accuracy} : 0.675$$

$$\text{Running for lambda} = 1 \text{ Accuracy} : 0.55$$

$$\text{Running for lambda} = 10 \text{ Accuracy} : 0.5625$$

$$\text{Running for lambda} = 100 \text{ Accuracy} : 0.25$$

- These are for $\alpha = 0.01$

$$\text{Running for lambda} = 0.01 \text{ Accuracy} : 0.6125$$

$$\text{Running for lambda} = 0.1 \text{ Accuracy} : 0.6$$

$$\text{Running for lambda} = 1 \text{ Accuracy} : 0.5125$$

$$\text{Running for lambda} = 10 \text{ Accuracy} : 0.55$$

$$\text{Running for lambda} = 100 \text{ Accuracy} : 0.25$$

$$\text{Running for lambda} = 0 \text{ Accuracy} : 0.5375$$

Testing2 : Hidden Layer Size = 45 and $\alpha = 0.1$

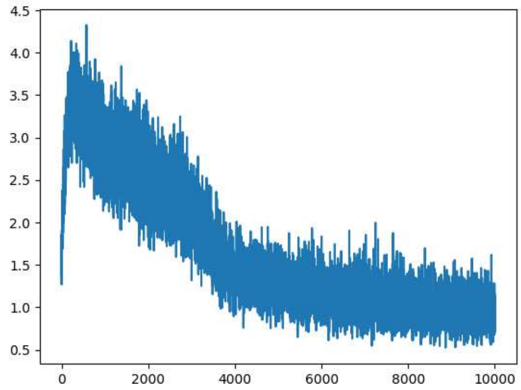
- Running for lambda = 0

$$\text{Accuracy} : 0.55$$

Precision : Mountain = 0.526315789474, Forest = 0.777777777778, Coast = 0.52380952381, Insidecity = 0.409090909091

F-measure : Mountain = 0.512820512821, Forest = 0.736842105263, Coast = 0.536585365854, Insidecity = 0.428571428571

Recall : Mountain = 0.5, Forest = 0.7, Coast = 0.55, Insidecity = 0.45



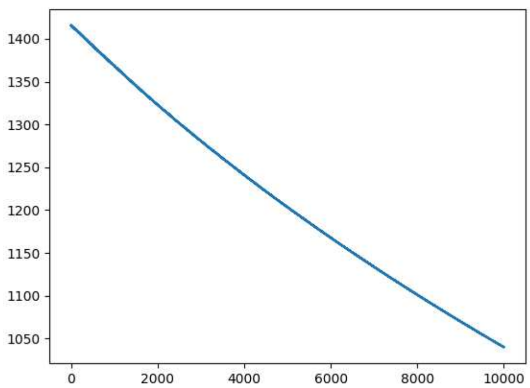
- Running for lambda = 0.01

Accuracy : 0.575

Precision : Mountain = 0.625, Forest = 0.619047619048, Coast = 0.5, Insidecity = 0.571428571429

F-measure : Mountain = 0.555555555556, Forest = 0.634146341463, Coast = 0.52380952381, Insidecity = 0.585365853659

Recall : Mountain = 0.5, Forest = 0.65, Coast = 0.55, Insidecity = 0.6



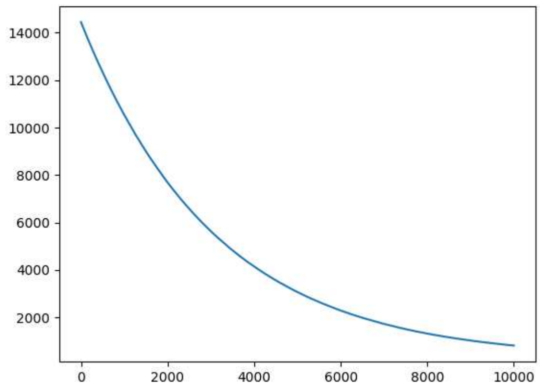
- Running for lambda = 0.1

Accuracy : 0.6375

Precision : Mountain = 0.458333333333, Forest = 0.833333333333, Coast = 0.625, Insidecity = 0.714285714286

F-measure : Mountain = 0.5, Forest = 0.789473684211, Coast = 0.681818181818, Insidecity = 0.588235294118

Recall : Mountain = 0.55, Forest = 0.75, Coast = 0.75, Insidecity = 0.5



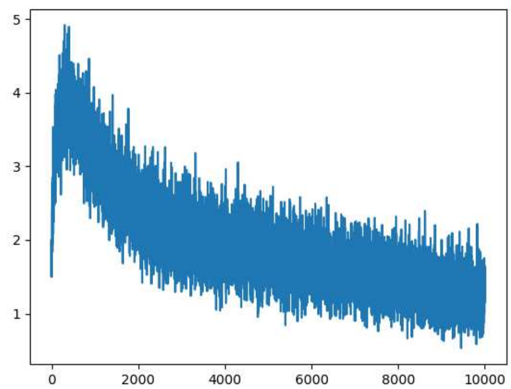
- Running for lambda = 1

Accuracy : 0.5625

Precision : Mountain = 0.583333333333, Forest = 0.727272727273, Coast = 0.521739130435, Insidecity = 0.434782608696

F-measure : Mountain = 0.4375, Forest = 0.761904761905, Coast = 0.558139534884, Insidecity = 0.46511627907

Recall : Mountain = 0.35, Forest = 0.8, Coast = 0.6, Insidecity = 0.5



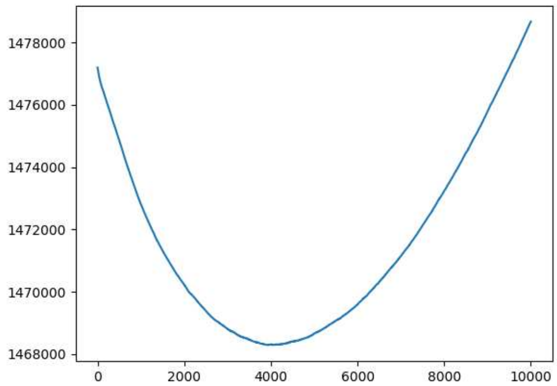
- Running for lambda = 10

$$\text{Accuracy}: 0.65$$

$$\text{Precision}: \text{Mountain} = 0.647058823529, \text{Forest} = 0.727272727273, \text{Coast} = 0.652173913043, \text{Insidecity} = 0.555555555556$$

$$\text{F-measure}: \text{Mountain} = 0.594594594595, \text{Forest} = 0.761904761905, \text{Coast} = 0.697674418605, \text{Insidecity} = 0.526315789474$$

$$\text{Recall}: \text{Mountain} = 0.55, \text{Forest} = 0.8, \text{Coast} = 0.75, \text{Insidecity} = 0.5$$



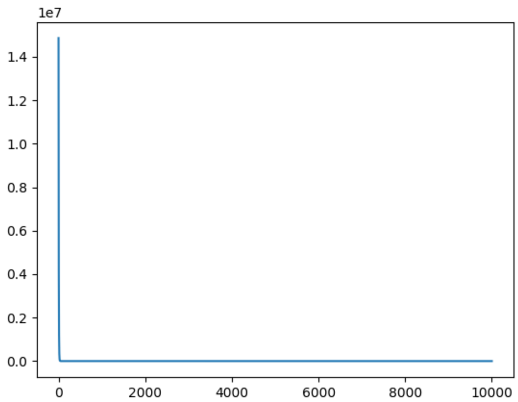
- Running for lambda = 100

$$\text{Accuracy}: 0.25$$

$$\text{Precision}: \text{Mountain} = 0.25, \text{Forest} = 0.0, \text{Coast} = 0.0, \text{Insidecity} = 0.0$$

$$\text{F-measure}: \text{Mountain} = 0.4, \text{Forest} = 0.0, \text{Coast} = 0.0, \text{Insidecity} = 0.0$$

$$\text{Recall}: \text{Mountain} = 1.0, \text{Forest} = 0.0, \text{Coast} = 0.0, \text{Insidecity} = 0.0$$



Maximum Accuracy Obtained = 65% for $\lambda = 10$

Inference

One of the major issues with artificial neural networks is that the models can become quite complicated and are more prone to overfitting. Hence there is a need for regularising the weights. There is always a direct trade-off between overfitting and model complexity. As we can see from the data obtained above, accuracy can decrease and increase depending on the value of the regularisation parameter λ. Hence we vary the value λ inorder to obtain the best fit possible suvh that we obtain minimum test error.