Principles of Machine Learning(CS4011)
Programming Assignment#1B

S Ram Ananth
ME15B153

September 12, 2017

1 Logistic Regression

1.1 Goal

The goal of this experiment is to perform 2-class Logistic Regression on DS2 using only forest and mountain classes. Report per-class precision, recall and f-measure on the test data. Now perform $L1$-regularized Logistic Regression on the same dataset and report similar performance results using $l1$ logreg code provided by Boyds Group.(http://www.stanford.edu/~boyd/l1_logreg/)

1.2 Approach

The dataset DS2 was prepared from the set of images provided for each class. This is was done by splitting each image into red,blue and green channels after which for each channel,a frequency histogram containing 32 bins was obtained. All these vectors were concatenated to give a $96D$ vector on which logistic regression was performed. The given problem is a classification problem as the dependant variable is categorical (either forest or mountain). Here logistic regression is preferred to linear regression because linear regression may predict values outside range of dependant variable and linear regression performs poorly in cases of class imbalance.Logistic regression solves this by using a trying to maximise a log-likelihood function instead of trying to reduce the ordinary least squares error function.

1.3 Result

L2 regularised logistic regression is performed for 2 classes, mountain(labelled as -1) and forest (labelled as 1)

Measure	Class mountain	Class Forest
Accuracy	0.75	0.75
Precision	0.7083	0.8125
Recall	0.85	0.65
F-Measure	0.7727	0.722

L1-regularised logistic regression is performed for 2 classes, mountain(labelled as -1) and forest (labelled as 1)

Measure	Class mountain	Class Forest
Accuracy	0.975	0.975
Precision	1.0	0.9523
Recall	0.95	1.0
F-Measure	0.9743	0.9756

2 Backpropogation

2.1 Goal

Implement original backpropagation algorithm. Use DS2 with all 4 classes for training your neural network. Report per-class precision, recall and F-measure on the test data used in previous question.

Now consider the following alternate error function for training neural networks.

$$R(\theta) = \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1}^{K}(y_{ik} - f_k(x_i))^2 + \gamma(\sum_{k}\sum_{m}\beta_{km}^2 + \sum_{m}\sum_{l}\alpha_{ml}^2)$$

where N is the number of training instances, K is the number of output features, f_k(x) is the predicted output vector, y is the original output vector, a and b are the weights and g is a regularization parameter. Derive the gradient descent update rule for this definition of R. Now train your neural network with this new error function. Report per-class precision, recall and F-measure on the same test data for various values of g from 10^{-2} to10^2 in multiples of 10 and repeat the experiment and report the results.

2.2 Approach

The weights are learnt for a neural network arel learnt through gradient descent. However even for a really shallow neural network computing the gradients of output functions with respect to weights normally is really tricky. This is done using very efficiently using the backpropogation algorithm where we work backwards from output.

Using the softmax function as output activation,sigmoid as activation function for other layers and cross entropy as the loss function,

$$\nabla_{a_{L+1}} L = \frac{\partial L}{\partial f_l}\frac{\partial f_l}{\partial a_{L+1}}$$

where l is correct label and f is predicted value

$$= -\frac{1}{f_l}(softmax^{'}(a_{L+1}))$$

$$= f(x) - e(l)$$

in vectorised form where e(l) is k dimensional vector whose l^{th} value is 1 and rest are 0.

Further using chain rule we to find gradient of loss function with respect to weight

$$\nabla_{h_i} L = (W_{i+1})^T \nabla_{a_{i+1}} L$$

$$\nabla_{a_i} L = \nabla_{h_i} L \odot g^{'}(a)$$

where $g^{'}$ is derivative of sigmoid function.

$$\frac{\partial L}{\partial W_{kij}} = \frac{\partial L}{\partial a_{ki}} h_{ki,,j}$$

$$\nabla_{a_k} L = \nabla_{b_k} L$$

For the alternate error function only $\nabla_{a_{L+1}} L$ changes to $(y - f(x))softmax^{'}(a_{L+1})$ and $2\gamma W$ term is added to $\frac{\partial L}{\partial W}$ term in gradient descent update rule.

2.3 Result

The following metrics were obtained for case with cross entropy as loss function.

Measure	Class mountain	Class forest	Class insidecity	Class Coast
Accuracy	0.45	0.45	0.45	0.45
Precision	0.3125	0.625	0.388	0.409
Recall	0.25	0.75	0.35	0.45
F-Measure	0.2777	0.6818	0.368	0.428

Alternate-error function best fit was obtained for $\gamma = 10$

Measure	Class mountain	Class forest	Class insidecity	Class Coast
Accuracy	0.5	0.5	0.5	0.5
Precision	0.458	0.66	0.55	0.35
Recall	0.55	0.6	0.5	0.35
F-Measure	0.5	0.63	0.526	0.35

The best fit was found for $\gamma = 10$. Regularistion helps overcome overfitting and weights observed after regularisation are maintained small in comparison to one without regularisation