Homework - II

Siddharth Chandrasekaran

October 22, 2017

Answer 1.a. Expression for Entropy is defined as H(D) = $-\sum_{i=1}^{K} p_i \log_2 p_i$,where p_i is a list of numbers that adds up to 1 and represents the weight of each child node that results from split in the tree. K is the number of output classes.
In this case , information gain is $H(D)$ - $(H(D_0) + H(D_1))$
[Entropy of parent - (Entropy of the two splits)]
More generally, information gain is defined as decrease in entropy after each split.

Answer 1.b. Decision tree with depths : $3, 6, 9, 12, 14$ were trained on the dataset.

Parameters	Accuracy	Error
$'maxdepth' : 3$	0.44855	0.55145
$'maxdepth' : 6$	0.4993	0.5007
$'maxdepth' : 9$	0.5182	0.4818
$'maxdepth' : 12$	0.5138	0.4862
$'maxdepth' : 14$	0.5117	0.4883

The depth of the tree increases the accuracy till the depth 9 and then accuracy starts dropping for higher depths. This is because after depth 9 over fitting starts. And for lower depths there is under fitting.
The model with the lowest estimated out of sample error was a decision tree with: $'maxdepth' : 9$. (Out of sample and test errors are defined as [1-accuracy])
The kaggle test error for this model was 0.44139 (1-0.55861) and the out of sample error was 0.4818 (1-0.5182). Kaggle test error was lower in this case.

Answer 2.a. Time taken is defined as the time taken to fit and predict.
Time taken to fit for KNN is O(1)
Time taken to predict = Time taken to compute all distances + Time taken to find K nearest neighbors = O(NF) + O(NK) = O(N *max(F,K)

Answer 2.b. KNN classifier with neighbors : $3, 5, 7, 9, 11$ were trained on the dataset.

Parameters	Accuracy	Error
$'nneighbors' : 3$	0.55205	0.44795
$'nneighbors' : 5$	0.54735	0.45265
$'nneighbors' : 7$	0.5424	0.4576
$'nneighbors' : 9$	0.5424	0.4576
$'nneighbors' : 11$	0.5352	0.4648

The model with least out of sample error was for the following parameters
$'nneighbors' : 3$. (Out of sample and test errors are defined as [1-accuracy])

The kaggle test error for this model was 0.38876(1-0.611) and the out of sample error was 0.44795(1-0.55205). Kaggle test error was lower in this case

Answer 3.a. LinearModel classifier (SGD Classifier) with regularization constants: $0.000001, 0.0001, 0.01, 1, 10$ and with losses hinge and log were trained on the dataset.

Parameters for ridge	Accuracy	Error
$'alpha' : 1e-06, 'loss' : hinge :$	0.4524	0.5476
$'alpha' : 0.0001, 'loss' : hinge$	0.41805	0.58195
$'alpha' : 0.01, 'loss' : hinge$	0.5255	0.4745
$'alpha' : 1, 'loss' : hinge$	0.4812	0.5188
$'alpha' : 10, 'loss' : hinge$	0.4439	0.5561
$'alpha' : 1e-06, 'loss' : log :$	0.41035	0.58965
$'alpha' : 0.0001, 'loss' : log$	0.41025	0.58975
$'alpha' : 0.01, 'loss' : log$	0.54165	0.45835
$'alpha' : 1, 'loss' : log$	0.4993	0.5007
$'alpha' : 10, 'loss' : log$	0.4106	0.5894

The model that worked best for was with the following parameters $'alpha' : 0.01, 'loss' :' log'$
The out of sample error in this case was 0.45835 (1-0.54165) And the kaggle test error was 0.34989(1-0.65011). Kaggle test error was lower in this case.

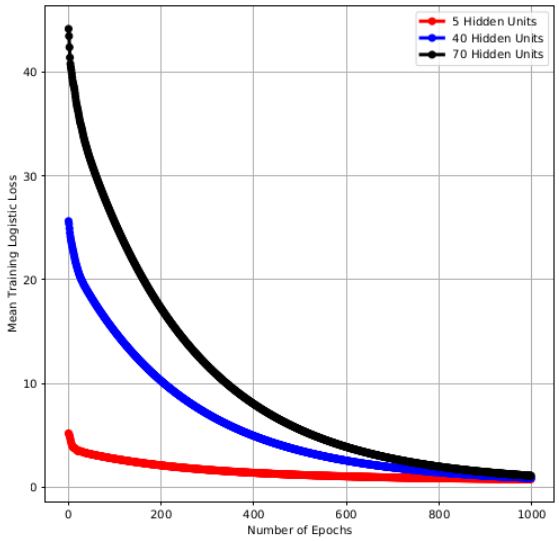
Answer 4.a. Code for this is present in *NNgradientonesample.py* and *run_me.py*.

$l = c + V@sig(b + W@x)$
$dLdf = np.array([0, 0, -1, 0]) + (np.exp(l)/np.sum(np.exp(l))).T$
$h = sig(b + W@x)$
$dLdV = dLdf.T@h.T$
$dLdb = np.multiply(sigp(b + W@x), V.T@dLdf.T)$
$dLdW = np.dot(dLdb, x.T)$
$return dLdW, dLdV, dLdb, dLdf.T$

2.Output for one data example:
Loss = 4.8145
dLdc, Autograd [[0.917135 0.045394 -0.991889 0.02936]]
dLdc, partial derivative [[0.917135 0.045394 -0.991889 0.02936]]
dLdV, Autograd [[-0.908328 0.848131 0.87481 -0.869308 -0.784553] [-0.044958 0.041979 0.043299 -0.043027 -0.038832] [0.982365 -0.917261 -0.946115 0.940164 0.848501] [-0.029078 0.027151 0.028005 -0.027829 -0.025116]]
dLdV, partial derivative [[-0.908328 0.848131 0.87481 -0.869308 -0.784553] [-0.044958 0.041979 0.043299 -0.043027 -0.038832] [0.982365 -0.917261 -0.946115 0.940164 0.848501] [-0.029078 0.027151 0.028005 -0.027829 -0.025116]]
dLdb, Autograd [[0.049305 0.189112 0.115923 -0.175011 -0.786228]]
dLdb, partial derivative [[0.049305 0.189112 0.115923 -0.175011 -0.786228]]
dLdW, Autograd [0.004256 0.016843 0.007743 -0.007793 -0.037776 0.003351 0.008865 0.003396 -0.006905 -0.024877]
dLdW, partial derivative [0.004256 0.016843 0.007743 -0.007793 -0.037776 0.003351 0.008865

0.003396 -0.006905 -0.024877]

Answer 4.b. 1. Plot: Mean Logistic Loss V/s Iteration Number of Epochs



2. Table:

Parameters	Time Taken
$'hiddenlayers' : 5$	100.4632
$'hiddenlayers' : 40$	321.1371
$'hiddenlayers' : 70$	515.0286

3. Split Table:

Parameters	Validation Error
$'hiddenlayers' : 5$	0.3995
$'hiddenlayers' : 40$	0.3475
$'hiddenlayers' : 70$	0.3385

Increase in number of hidden units increases the accuracy of the model. This is because the capacity of the model increases as the number of hidden layers increase.
The model that worked best for was with the following parameters $'hiddenlayer' : 70$
The out of sample error in this case was 0.3385 (1-0.6615) And the kaggle test error was 0.20874 (1-0.79126). Kaggle test error was lower in this case.