Kaggle Competition Project Report (Classification on Titanic Dataset)

Dawit Beshah(1144901)

Experiment one: Building Neural Net from Scratch

[Appendix I: kaggle1.pynb]

Methods: in this experiment, what I try to do is:

-Dropping some feature that I think is irrelevant by examine the training data. I dropped (ID,A1,A2).
-learning_rate=0.0000001, training_epochs = 2000, display_step =50,
-initialized the weight and the bias vector with zero (here the weight and biased vector are with 17 entries as the input vector).
-use the tensorflow's GradentDescent optimizer to minimize the training cost with ***softmax*** activation function

Result: I can't get results because mismatch inputs(17) and output(1) vector

Discussion: This problem can't be solved with single layer network.

Experiment Two: Multilayer Neural Network with different optimizers and hidden layer configuration

[Appendix II: Kaggle2.pynb]

Methods1: in this experiment, I just used trial and error method by observing the accuracy the prediction model with different optimizers and hidden layer configuration

The optimizers I applied are:

- tf.train.GradientDescentOptimizer.
- tf.train.AdagradOptimizer
- tf.train.MomentumOptimizer
- tf.train.FtrlOptimizer
- tf.train.RMSPropOptimizer
- tf.train.ProximalAdagradOptimizer

I choose these optimizers depended results from the pervious experiment result I did for assignment one.

Result : I run trials about 7-10 different hidden layer configuration for each optimizer and the best Kaggle score I got 65.4 with -tf.train.ProximalAdagradOptimizer [with learning rate =0.0001 and hidden layer =[18, 9, 2, 1].

Discussion: I found out the best way of configuring hidden network is to put almost equal number of neuron as the number of neurons in the input layers at the hidden layers that are near to the input layer and narrow or wide up in the consecutive layer to get almost approximated number of neurons as the output neurons in the last layer of hidden network.

For this case: input neurons: - 18 and output neurons :- 1 and the best hidden network configuration is [18, 9, 2, 1].

Method 2: use k-folding with k = 5 to improve the prediction model. And I run number it to get good prediction as the folding is random every time.

Result: I got around 64 Kaggle score during the best run.

Discussion: train our model with some k amount folding improve the prediction model.

Experiment Three: use different sklearn library calls as classifier

[Appendix III: Kaggle 3 & 4]

Methods: I used:

- ske.RandomForestClassifier
- ske.GradientBoostingClassifier
- ske.VotingClassifier
- logistics regression
- SGDClassifier()

Result: the best run I came up with ske.GradientBoostingClassifier (n_estimators=50) and with SGDClassifier().

Lastly: because TFlearn libray of Tensorflow isn't install in our server I can't run some of my program.