CS412 HW3

Wangfei Wang

1 Neural Network with Backpropagation

Separate submission on GradeScope.

2 Neural Network Testing

2.1 Transfusion Dataset (New Dataset)

70% of the data set was separated out to serve as training set. Number of hidden layer = 1 and number of nodes per layer = 10 minimizes the CV error (see **Table 1**). Hidden layer = 2 and number nodes per layer = 2 minimized the CV error when the whole data set was considered (error table not shown here).

# hidden layers	# nodes per layer		
	2	5	10
1	0.388 ±0.412	0.3457 ±0.390	0.252 ±0.00942
2	0.350 ±0.386	0.312 ±0.294	0.258 ±0.0337
5	0.252 ±0.00942	0.252 ±0.0193	0.366 ±0.377

Table 1: **Cross Validation Errors for Neural Network Models with Different Parameters (Transfusion data).** Mean CV errors ± 1.96 × SD are shown.

2.2 Digit Dataset with Two Features (Old Dataset)

1. Number of hidden layers 1, 2, 5, 10 and the number of nodes per layer to be all of 2, 5, 10, 50, 100 were used as parameters for neural network models. Cross Validation errors for each case are given in **Table 2**.

2. Short answers:

a) How does runtime scale with the number of layers and the number of nodes per layer? Do each have a similar effect?

The runtime generally increases with the increased complexity of the model. When the number of nodes per layer is small, the number of hidden layers does not influence the runtime as much. The run time increases with the increase of number of nodes per layer.

The number of nodes per layer influences the runtime more significantly than the number

# hidden layers	# nodes per layer				
	2	5	10	50	100
1	0.185 ±0.295	0.179 ±0.261	0.209 ±0.297	0.134 ±0.310	0.0709 ±0.263
2	0.111 ±0.237	0.162 ±0.279	0.042 ±0.186	0.00323 ±0.0190	0.0699 ±0.256
5	0.143 ±0.278	0.0355 ±0.0718	0.0162 ±0.0432	0.00645 ±0.0253	0.00323 ±0.0190
10	0.0872 ±0.235	0.0258 ±0.0479	0.0163 ±0.0434	0.00645 ±0.0253	0.00645 ±0.0253

Table 2: **Cross Validation Errors for Neural Network Models with Different Parameters.** Mean CV errors ± 1.96 × SD are shown.

# hidden layers	# nodes per layer				
	2	5	10	50	100
1	11321	18855	27597	37911	53265
2	9698	17822	31133	43625	55871
5	9275	21799	33944	44357	56706
10	9459	20884	31125	41019	51283

Table 3: **Runtime (ms) for Neural Network Models with Different Parameters.** Time in ms. Numbers are rounded to integers. Run time was calculated for total time for each cross validation.

of hidden layers.

b) What is the number of layers and nodes per layer that gives an optimum result? The NN model with hidden layer = 2 and number of nodes per layer = 50 gives the smallest cross validation error upper bound. Hidden layer = 5 and number of nodes per layer = 100 also gives the smallest cross validation error upper bound. Both of them generate the same result.

c) For your optimum model, try increasing and decreasing the learning rate from the default (0.001). Discuss the tradeoff here between runtime and accuracy?

$$w_{new} = w_{current} - \eta \times gradient$$

In this particular training data set, the learning rate = 0.01 minimizes the CV error (95% CI upper bound). Runtime is shorter as η increases from 0.001 to 100. When η is very big, for example, when $\eta = 1, 10, 100$, the runtime is very small because gradient descent is taking bigger steps. But in trade off, the errors are high, showing that gradient descent can overshoot and miss the minimum.

In the other extreme case, when η is very small, i.e., smaller than 0.0001 in this case, the runtime is also smaller than needed for $\eta = 0.001$ and the CV errors are bigger than $\eta = 0.001$ case. This may be caused by the fact that the η is too small and gradient descent takes too many little steps to find its minimum, but it may hit the maximum iteration for the model and stop before the model finds its minimum.

d) Is the neural network finding the same solution every time? Why or why not? Does

Learning rate	CV error	runtime
0.00001	0.539 ±0.327	2860
0.0001	0.502 ±0.400	6905
0.001	0.107 ±0.193	12340
0.01	0.0730 ±0.310	3594
0.1	0.0969 ±0.196	1560
1	0.314 ±0.158	337
10	0.330 ±0.199	365
100	0.465 ±0.328	226

Table 4: **Learning rate affecting CV error and run time.**

this have an impact on the expected fit?

The results always vary a little bit but it does not impact the expected fit. Because although the training set is fixed, the initial weights of the neural network are set to a random value in a small range and are not exactly the same. But the small change in the initial weights won't impact the expected fit.

e) Graduate Student Question: Experiment with early stopping on neural networks that have a large number of internal nodes. Does this cause the model to under or over fit the data? Why?

Number of hidden layer = 5, nodes per layer = 900, activation = "relu", epsilon = 0.001, max_iter = 10000, alpha = 0, solver = "adam" were used to explore the early_stopping feature. When early_stopping is set to false, the training is terminated when validation score is not improving. But when it is set to true, it automatically sets aside 10% of training data as validation and terminate training when validation score is not improving by at least tol for n_iter_no_change consecutive epochs (based on the documentation from MLPClassifier).

The model may not be trained to be fitting the data well when early_stopping is set to be True. So the model can suffer underfitting when early_stopping = True. The decision boundary is shown in **Figure 3.0**.

	early_stopping = False	early_stopping = True
Training Error	0.00321	0.167
Test Error	0.00961	0.141

Table 5

3. Plot the decision region for the optimal neural network above in the 2D space Label this figure 3.1.

See **Figure 3.1** below.

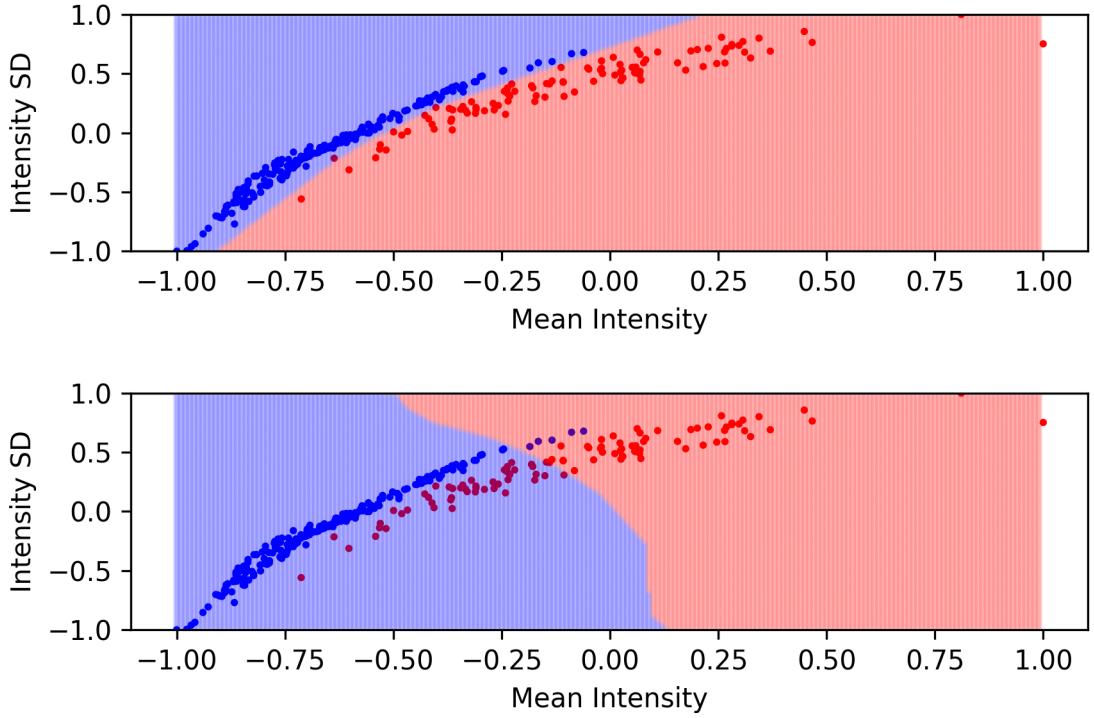

Figure 3.0. Early_stopping = False (top) vs. Early_stopping = True (bottom).

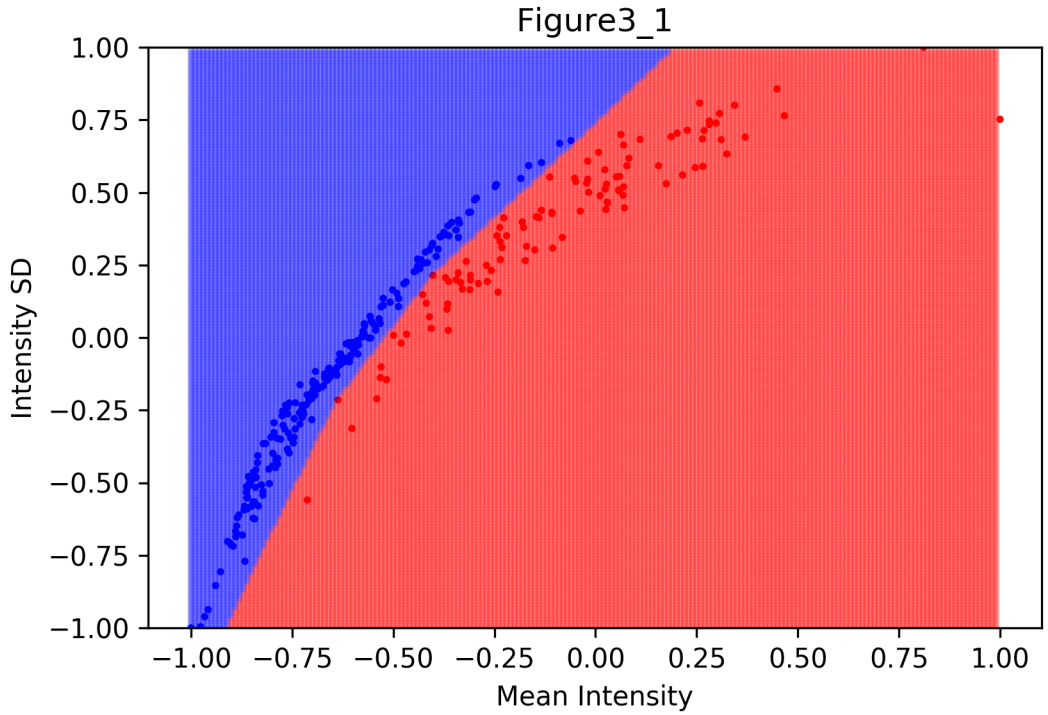

Figure 3.1. Decision region for 2D space with optimal neural network model.

3 Extra Credit

The transfusion data were used as test set for this section. Fold of CV was set to 30. The number of number of hidden layers in 1, 2, 5 and the number of nodes per layer in 2, 5, 10 were explored. **Table 6** shows the comparison of mean of CV errors in neural network and bagged neural network.

Number of hidden layers	Number of nodes per layer					
	2		5		10	
	NN	Bagged NN	NN	Bagged NN	NN	Bagged NN
1	0.348 ±0.375	0.201 ±0.103	0.294 ±0.291	0.255 ±0.332	0.252 ±0.00942	0.241 ±0.312
2	0.452 ±0.476	0.278 ±0.371	0.350 ±0.387	0.222 ±0.226	0.455 ±0.466	0.270 ±0.408
5	0.364 ±0.369	0.260 ±0.326	0.370 ±0.376	0.295 ±0.401	3.639 ±0.384	0.322 ±0.467

Table 6: **Cross Validation Errors for Bagged Neural Network Models with Different Parameters (Transfusion data).** Mean CV errors ± 1.96 × SD are shown. CV fold = 30.

a). Does the optimum number of bagged neural networks change with the complexity of those networks?

The optimum model of NN now changed to number of hidden layer = 1 and number of nodes per layer = 2 instead of number of hidden layer = 1 and number of nodes per layer = 10.

b). Do the more complex models provide better solutions when applied in a bagging ensemble model?

Bagging improved the models by reducing the CV error and variances. The more complex models usually tend to have overfitting issue, and variances can be very high. In these cases, bagging is extremely helpful for reducing variances.

c). What about neural networks makes them react well to bagging?

Because neural network output can change dramatically when the training data is slightly changed. The variances are usually high for neural networks. Also when the NN model is complex and have the tendency to overfit, bagging can reduce variance and alleviate overfitting of the model.

d). Would you choose a bagging model as your "optimum" neural network model?

Bagging could be an "optimal" neural network model, but there are also other options for ensemble methods, such as boosting and stacking. I think it depends on the data set to decide which one is better. For example,bagging is great for decreasing variance when a model is overfit. However, boosting is much more likely to be a better pick of the two methods and is also great for decreasing bias in an underfit model.