1 Neural Networks

Code: https://github.ubc.ca/cpsc340-2017S/sopida_zhenxil_a5/tree/master/code/main.py

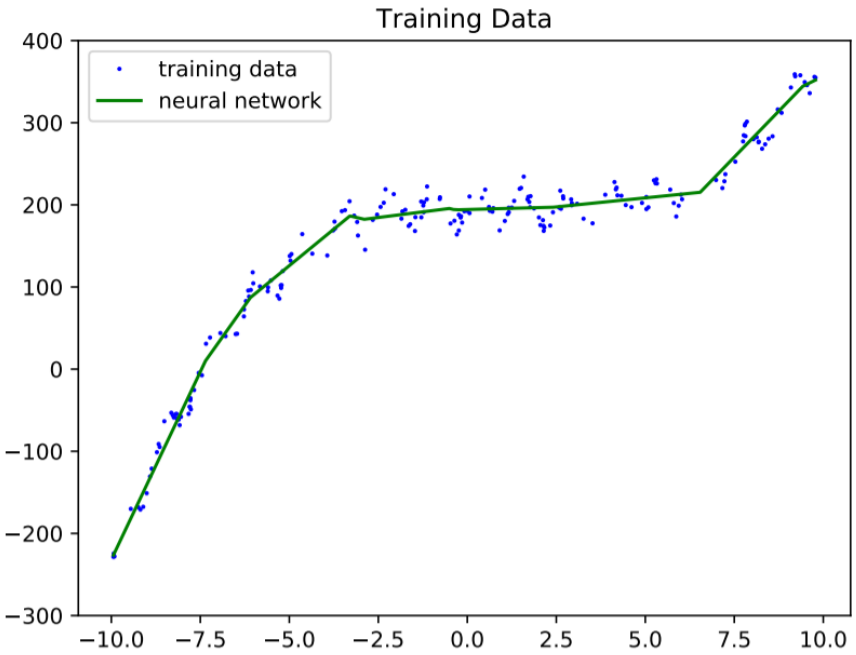

Fig: https://github.ubc.ca/cpsc340-2017S/sopida_zhenxil_a5/tree/master/figs/basisData.pdf

Training error = 223.20
Test error = 257.73

Changes made:

1. Set solver to lbfgs
2. Adding 10 hidden layers
3. Set alpha to 0.5

The solver is set to lbfgs since our data set is small. Hidden layers are added for deep learning which lowers the training error. If the depth is too high, the training error will not be a good approximation of the test error. Hence, alpha used in L2-regularization is set to reduce overfitting of the model.

2 Neural Networks with Your Own Data

Code: https://github.ubc.ca/cpsc340-2017S/sopida_zhenxil_a5/tree/master/code/main.py

We used the Boston house-prices dataset from sklearn.datasets for testing. There are 506 samples with dimensionality of 13. The data was split into 50% training and 50% validation.

The parameter alpha from the Neural Network model is the L2-regularization parameter. We tried different values for alpha:

When alpha=0.5:

Validation error = 31.99740017272013

When alpha=0.01:

Validation error = 30.895361524058437

When alpha=0.1:

Validation error = 30.484973713642347

hidden_layer_sizes controls the size of hidden layers of Neural Network, with alpha = 0.1:

hidden_layer_sizes=(1):

Validation error = 96.43627395257292

hidden_layer_sizes=(10):

Validation error = 24.416529842971684

hidden_layer_sizes=(100):

Validation error = 27.256572247246364

As we can see, there is not much effect on validation error by changing different values for the alpha. After adding 1 hidden layer with alpha = 0.1, the validation increases from 30.5 to 96.4. After 10 hidden layers, the validation error improves to 24.4. However, adding any more hidden layers doesn't improve the validation error further.

The validation error is relatively small. However, this dataset is about housing price and we need the smallest possible error to predict the future housing price in Boston. Therefore, Neural Network is not a good model in this case.

3 Recommender Systems

3.1 Implementation

Code: https://github.ubc.ca/cpsc340-2017S/sopida_zhenxil_a5/tree/master/code/recommender.ipynb

3.2 Comparing methods

Method	Training error	Validation error
1 Global average	1.118	1.124
2 Per-user average	0.912	0.924
3 Per-movie average	0.790	0.996
4 Average of per-user and per-movie averages	0.747	0.852
5 Linear regression with movie features	1.082	1.088
6 Per-user linear regression on genre	0.715	1.040
7 SVD with per-user and per-movie averages	0.581	0.822
8 SVD with proper handling of missing features	0.669	0.789
9 Gradient descent plus per-user movie features	0.550	0.767

Methods 8-9 are better than the rest because they both handle missing entries in the objective function. Method 9 is more overfitted than method 8 because method 9 includes linear regression which method 8 doesn't. Method 7 is overfitting because validation error is significantly larger than the training error plus it naively treats missing entries as 0. Methods 3 and 5 have higher errors because these methods predict the same result for each movie regardless of the user. Method 2 has higher errors because it predicts the same result for each user regardless of the movie. Method 1 is averaging the rating across all users and movies without taking the user- and movie-specifics into account. Hence, method 1-5 are underfitting (higher training error).

3.3 Ridge regression

It makes sense to use ridge regression for method 6 because the model is overfitting (training error is not a good approximation of validation error). Thus, method 6 needs regularization term to reduce the overfitting. On the other hand, method 5 is not overfitting (validation error is approximately the same as the training error), so it doesn't need to be regularized.

3.4 Stochastic gradient

It makes sense to use stochastic gradient for method 8 and 9 when the data size is large. This will improve the computation time since we don't have to calculate all the gradients per each iteration. However, stochastic gradient has erratic behavior which is sensitive to step size.

3.5 Validation

If we split the data by movies or users, some of the users or movies might be in our training set but not in our validation set and vice versa. For example, some users might have not rated a lot of movies and we are splitting the data by users. Hence, our sample will not be IID which will result in poorer training and validation errors

3.6 Next steps

We can do L2-regularization to reduce overfitting. The squared error loss is sensitive to outlier, so we can use absolute loss which is less sensitive to outlier. We can use validation set to again avoid overfitting our model. We could collect more features for the movie rather than just having genre. For instance, we could have movie's director, main actor/actress, etc. Also, we can collect the rating of movie based on age groups.