GE 461: Introduction to Data Science
Homework Supervised Learning

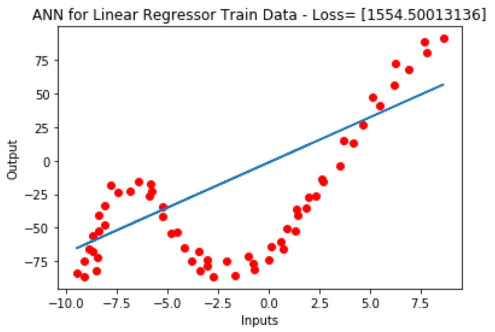
Batıhan Akça - 21502824

PART 1:

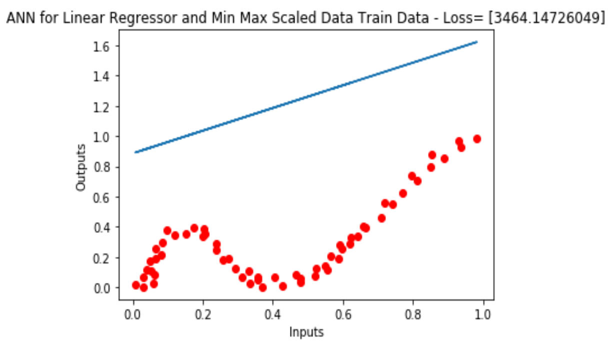
(a) With such methods in CNN like pooling layers, overfitting which is an important problem for deep learning studies can be reduced.
(b) Very deep CNNs start memorizing data so that their models give very low losses for train data but cannot be successful for other data, works for specifically for train data. Regularization helps for this problem. For more complexity stronger regularization can be used.
(c)

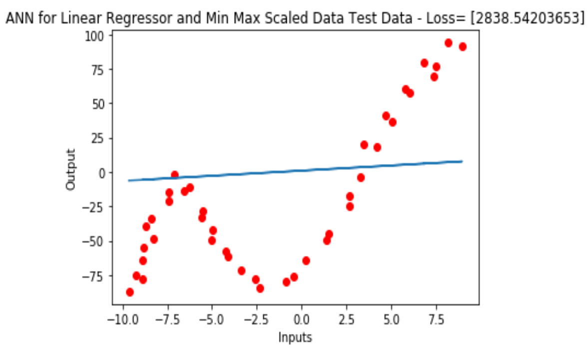
PART 2:

ANN For Linear Regressor No Hidden Layer



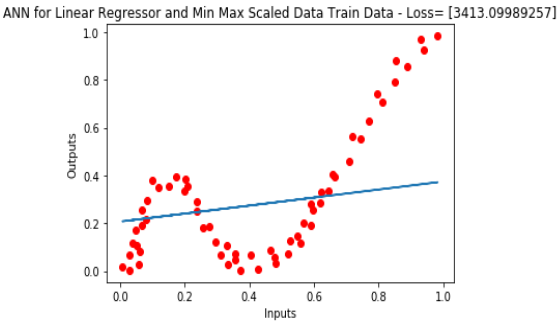
Learning rate: 0.001
Range of initial weights: [0,1]
Number of epochs: 1
When to stop: Selected
Is normalization used: No
Training loss (averaged over training instances): 1554
Test loss (averaged over test instances): 1255

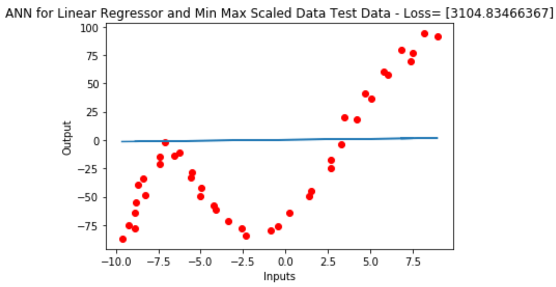




Learning rate: 0.001
Range of initial weights: [0,1]
Number of epochs: 1
When to stop: Selected
Is normalization used: Yes
Training loss (averaged over training instances):
Test loss (averaged over test instances):

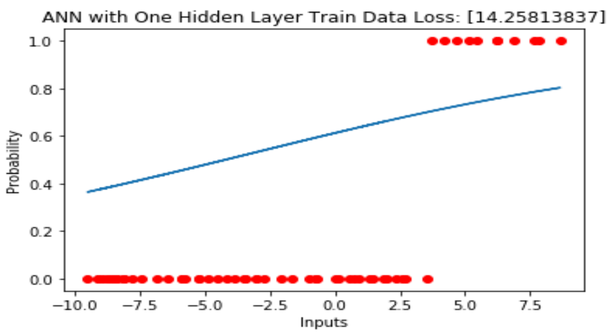
Learning rate: 0.001
Range of initial weights: [0,1]
Number of epochs: 10
When to stop: Selected
Is normalization used: No
Training loss (averaged over training instances): 1326
Test loss (averaged over test instances): 1225





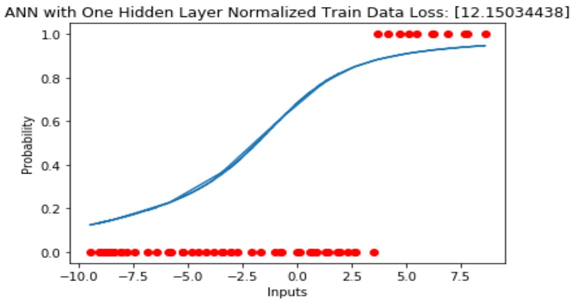
Learning rate: 0.001
Range of initial weights: [0,1]
Number of epochs: 10
When to stop: Selected
Is normalization used: Yes
Training loss (averaged over training instances): 3413
Test loss (averaged over test instances): 3104

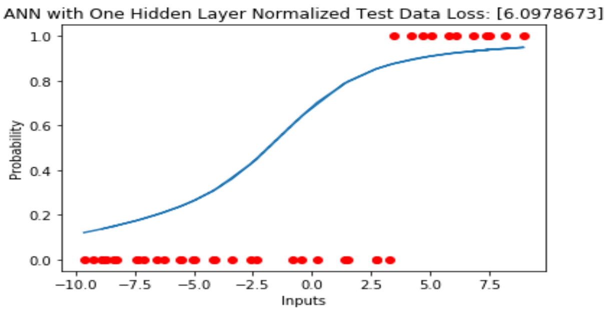
ANN With One Hidden Layer



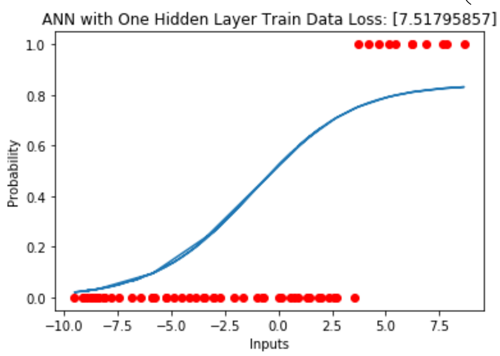
ANN used (specify the number of hidden units): 2

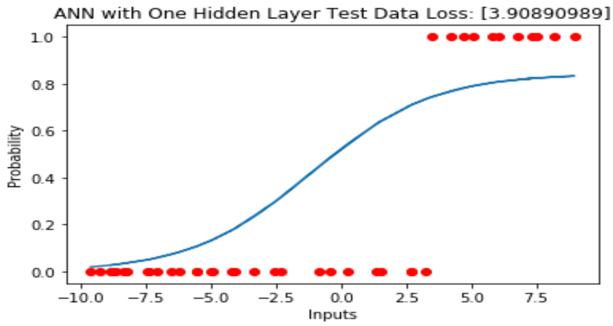
Learning rate: 0.0001
Range of initial weights: [0,1], 0.5 for hidden to outputs
Number of epochs: 10
Is normalization used: No
Training loss (averaged over training instances): 14.25
Test loss (averaged over test instances): 8.34



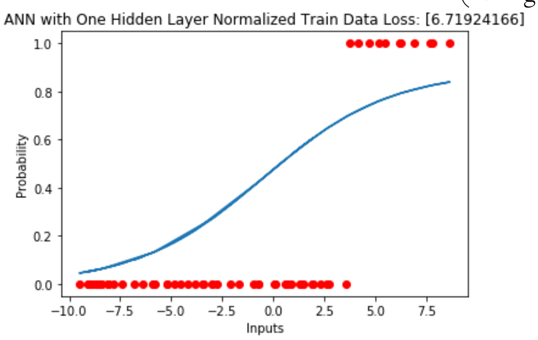


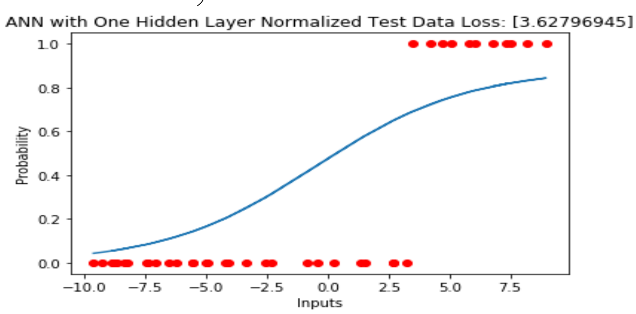
ANN used (specify the number of hidden units): 2
Learning rate: 0.0001
Range of initial weights: [0,1], 0.5 for hidden to outputs
Number of epochs: 10
Is normalization used: Yes
Training loss (averaged over training instances): 12.150
Test loss (averaged over test instances): 6.097





ANN used (specify the number of hidden units): 2
Learning rate: 0.0001
Range of initial weights: [0,1], 0.5 for hidden to outputs
Number of epochs: 100
Is normalization used: No
Training loss (averaged over training instances): 7.517
Test loss (averaged over test instances): 3.908





ANN used (specify the number of hidden units): 2
Learning rate: 0.0001
Range of initial weights: [0,1], 0.5 for hidden to outputs
Number of epochs: 100
Is normalization used: Yes
Training loss (averaged over training instances): 6.719
Test loss (averaged over test instances): 3.627

Most of the time (it only works for very low epoch numbers (<5) but low epochs cannot learn weights properly) for learning rate >0.1, weights go infinity, so the model does not work. Therefore, once it observed learning rate is used 0.01 at maximum.

Initialization is not important for linear regressor without hidden layer because its cost function is convex. It means that, since the local minimum is also the global minimum, with enough number of epochs it converges to the minimum in the end for every initialized weight. However, for the ANN with hidden layers it is not the case. Since we used squared error loss function with sigmoid activation function, we lost convexity. It did not perform for the homework but it the cross-entropy function were used as the loss function, ANNs with hidden layers would give better results without initialization consideration.

Max-min scale normalization significantly helped the ANN with hidden layers, however, for the linear regressor it decreased its performance for low numbered epochs and learned slower than the non-normalized one.

Results from Linear Regressor without Hidden Layer

Epochs	Learning Rate	Train Loss	Test Loss	Norm-Train Loss	Norm-Test Loss
1	0.1	3.02E+26	3.59E+26	3397	2787
1	1	1.19E+134	2.26E+134	3392	2886
10	0.01	1201	1382	3406	2892
10	0.1	1.37E+235	1.16E+235	3404	2855
100	0.01	1.20E+03	1.35E+03	3405	2853
100	0.001	1.19E+03	1.37E+03	3415	3101
1000	0.001	1.19E+03	1.38E+03	3405	2855
1000	0.0001	1.19E+03	1.37E+03	3408	2926

Result from ANN with One Hidden Layer

Epochs	Learning Rate	Hidden Units	Train Loss	Test Loss	Norm-Train Loss	Norm-Test Loss	Train Stdev	Test Stdev	Norm-Train Stdev	Norm-Test Stdev
10	0.01	2	5.37E+00	3.91E+00	6.04518903	4.69354003	0.0938902	0.10186	0.111679119	0.126904456
100	0.001	2	5.16E+00	3.86E+00	5.15845279	3.72426458	0.0942528	0.10341	0.097221589	0.10248196
10	0.01	4	5.68008405	4.266751	5.1544722	3.81109132	0.1033186	0.11418	0.094268032	0.102162392
100	0.001	4	5.37696113	3.987669	5.46958896	4.12004927	0.098591	0.107468	0.102136296	0.112577106
10	0.01	8	5.24E+00	3.73E+00	5.49425767	4.37615355	0.0965441	0.101038	0.114485037	0.129022733
100	0.001	8	5.13E+00	3.78E+00	5.26550376	3.91030107	0.0943394	0.101998	0.09717086	0.105829948
10	0.01	16	5.19E+00	4.02E+00	5.07765725	3.61509824	0.1040106	0.115494	0.093541478	0.097933027
100	0.001	16	4.96E+00	3.67E+00	4.93945495	3.64718899	0.095903	0.10233	0.095146426	0.101376631
10	0.01	32	5.06381083	4.021177	5.32801828	4.69127226	0.1058589	0.117396	0.13035338	0.148982085
100	0.001	32	4.75903881	3.344142	4.87661187	3.57406832	0.0933974	0.095331	0.095031987	0.100033002

Results have shown that increased complexity decreases losses inevitably, however, after a point it loses its generalizability. For our example, our dataset had dominance of "0" labeled samples and there were few with "1" (11 ones /39 zeros), with the increasing complexity prediction curve start losing higher probabilities for label 1 in other words it started giving at most 0.6 probability output for the top highest input samples with a lesser loss in overall. But also without a significant complexity, it is not possible to see a sigmoid functions like curve as an output.