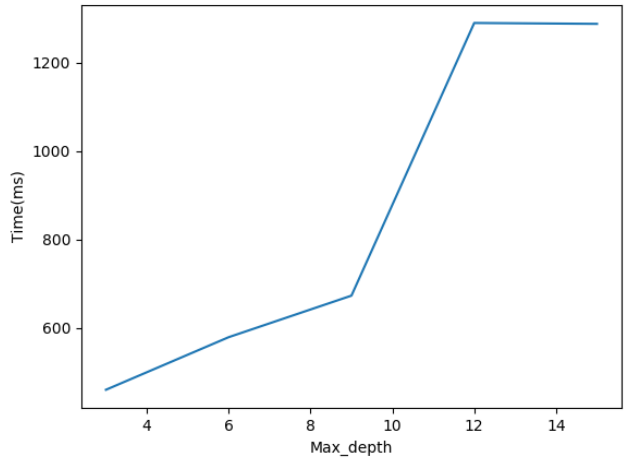
HW4

Yifu Liu

1. Decision Tree

   a. Table

Max_Depths\ Averages_errors	
3	6.36431016808
6	4.90028729205
9	4.86380517577
12	5.20330023454
15	5.38284332882



   b. The result on Kaggle has the large distance with the result on my machine. The model that I chose was when the Max_depth = 9, and the result on machine is 4.86. However, the result on Kaggle is 13.18, which obviously shows the difference.

2.

N_neighbors\ Averages_errors	
3	6.9055240562674829
5	6.7913171602689903
10	7.0013426559540761
20	7.2986417441194344

25	7.4350466725607784

b. The result on Kaggle has the large distance with the result on my machine. The model that I chose was when the N_neighbors = 5, and the result on machine is 6.79. However, the result on Kaggle is 11.58, which obviously shows the difference. Even though, the error distance between the result on Kaggle and my machine is closer than Decision Tree, but the expected distance is large than what I expected.

c. I chose when P = 1, P = 2(default) P = 100 for this question, where P = 1 is Manhattan distance and P = 2 is Euclidean distance. When P = 1, the result is 6.49(average), when P = 2, the result is 6.79(average), and when P = 100, the result is 7.20. My understanding of the reason of this question was, for instance, in this dataset, there are only three features, and if use Euclidean distance, the result is the maximum distance between any of two dimensions. It doesn't make any sense, and also we have over 50 features in our dataset, so when p = 2, there is no reason to use Minkowski for this problem.

3. Linear Model

a.Table

Alpha\Ridge and Lasso	Ridge	Lasso
1e-6	8.6689743225	8.65471810989
1e-4	8.66897431796	8.65461613361
1e-2	8.66897386415	8.64597722224
1	8.66892855182	8.53155573386
10	8.66852611112	8.234827216

b. [34 20 32 25]. Based on the code that was written, those four features given the least contributions to the results.

4. SVM

a.Table

Kernels/Degrees	
Poly, 1	6.8699886145139502
Poly, 2	7.5433651967860218
Rbf	6.986263733306191

I chose the Poly with Kernel = 1.

c. It only states the Poly kernel with Degree = 2 is not a good assumption for this data set. Bad assumption can give a large bias such as Poly kernel with Degree = 2.

5. Neural Networks

a. Table

Hidden Units	val	train
10	6.7158388729588152	7.31621661
20	7.4616212947509721	6.80122495
30	7.3665147944202047	6.82050837
40	7.1119380939373213	6.9850941

b. No. Since more hidden layers means more space to fit the dataset, so that the result should be reduced consistently. However, in this problem, the result for train error got slightly increased when hidden units move from 30 to 40, which shows it's not always reducing the error for neural networks.

6. Kaggle Competition

a. I choice to use SVM with scaled train and test set, with Kernal = "poly", degress = 1 and 3-fold validation, for the best output on Kaggle. The hyper-parameter range is(0.01, 0.1, 1, 10) and 10 is the best based on the result, and the MAE on Kaggle is 8.57.