CS 178 – Final Project Report

1. Neural Network

   Step 1: Select the best value for number of hidden layers (h)
   Split the data into (Xtr, Ytr) and (Xte, Yte) with a fraction of 0.75. We then trained the data with MLPRegressor from sklearn package, and calculated different mean square errors with a list of c values (i.e. refer to Table 1.1). We used ADAM as it generally works better on large data sets than LBFGS. We chose the best value of h = 4 because the predictor would be underfitting when h is smaller than 4 and overfitting when h is larger than 4.

   Step 2: Pack the 2 predictors into an ensemble predictor and train it
   We built an ensemble predictor that included MLPRegressor (with h=4 and ADAM algorithm) and GradientBoostingRegressor. We predicted the data (Xtr, Ytr) and (Xte, Yte) separately in it. The data set from X testing set was also trained for future prediction.

   Step 3: Choose the best loss function for Gradient Boosting
   We applied all 4 functions available in the predictor (i.e. refer to Table 1.2). Huber function, which is the combination of least squares regression and last absolute deviation, got the highest AUC score on validation data, which is 0.70687.

   Step 4: Choose the best number of estimators for Gradient Boosting
   Because Gradient Boosting used to robust to overfitting so a larger number of estimators usually results in better performance. Therefore, We tested the GradientBoostingRegressor (GBR) with different number of estimators to find the one with the highest AUC score (refer to Table 1.3). We chose 4000 estimators since the AUC score on validation started to decrease after 4000.

   Step 5: Conclusion
   AUC score on training data = 0.82025, AUC score on validation data = 0.72466, Kaggle score on test data = 0.73012

2. Random Forest

   Step 1: Pack the 3 predictors into an ensemble predictor and train it
   We built the "stacked" ensemble predictor that included RandomForestRegressor (assume number of trees=100), ExtraTreesRegressor, GradientBoostingRegressor from sklearn package. Because there are only 14 features in the X training set, we decided to add new features to it by multiplying the number of trees by 2. The data is then trained in the GBR. We split the data into (Xtr, Ytr) and (Xte, Yte) with a fraction of 0.75. The data sets (Xtr, Ytr) and (Xte, Yte) are predicted separately in the predictor. The data set from X testing set was also trained for future prediction.

   Step 2: Find the best number of estimators for Random Forest
   Because we needed to find the best estimator (i.e. number of tree) for the ensemble predictor, we randomly chose several estimator values and calculated the AUC scores on training data and validation data (i.e. refer to Table 2.1). We chose the best estimator to be 275 since it had the highest AUC score on validation data.

   Step 3: Choose the best number of estimators for Gradient Boosting
   Because Gradient Boosting used to robust to overfitting so a larger number of estimators usually results in better performance. Therefore, We tested the GradientBoostingRegressor (GBR) with different number of estimators to find the one with the highest AUC score (refer to Table 2.2). We chose 8000 estimators since the AUC score on validation data tended to grow very slow starting at 4000.

   Step 4: Conclusion
   AUC score on training data = 0.96943, AUC score on validation data = 0.76335, Kaggle score on test data = 0.76631

3. KNN

   Step 1: Select the best k value

Split the data into (Xtr, Ytr) and (Xte, Yte) with a fraction of 0.75. We then trained the data with KNeighborsRegressor from sklearn package, and calculated different mean square errors with a list of k values (i.e. refer to Table 3.1). We chose the best k value to be 200 as the predictor would be underfitting when K is smaller than 200 and overfitting when K is larger than 200.

Step 2: Pack the 3 predictors into an ensemble predictor and train it
we built the "stacked" ensemble predictor that included KMeans, KNeighborsRegressor, GradientBoostingRegressor from sklearn package. First, the data sets were put in KMeans for clustering with the number of clusters=8, and then trained in KNeighborsRegressor with k=200. The sets were boosted by using Gradient Boosting. The data sets (Xtr, Ytr) and (Xte, Yte) are predicted separately in the predictor. The data set from X testing set was also trained for future prediction.

Step 4: Choose the best number of estimators for Gradient Boosting
Because Gradient Boosting used to robust to overfitting so a larger number of estimators usually results in better performance. Therefore, We tested the GradientBoostingRegressor (GBR) with different number of estimators to find the one with the highest AUC score (refer to Table 3.2). We chose 16000 estimators since the AUC score on validation data tended to grow very slow starting at 8000.

Step 5: Conclusion
AUC score on training data = 0.93615, AUC score on validation data = 0.75676, Kaggle score on test data = 0.76155

4. SVM

Step 1: Select the best c value
The X training set and Y training set into (Xtr, Ytr) and (Xte, Yte) with the training fraction of 0.75. We then trained the data with SVC from sklearn package, and calculated different mean square errors with a list of c values (i.e. refer to Table 4.1). Note that the kernel was set to RBF. We chose the best c value to be 0.8 because the predictor would be underfitting when C is smaller than 0.8 and overfitting when C is larger than 0.8. We use the same predictor again to calculate different mean square errors with the c value of 0.8 and a list of gamma values (i.e. refer to Table 4.2). We chose the best gamma value to be 0.1 as the predictor would be overfitting when gamma is larger than 0.1.

Step 2: Select the best kernel for SVC
We then calculated the validation MSE to determine which kernel is better to be used in SVC with C value of 0.8 (i.e. refer to Table 4.3). It turned out that RBF generally had a lower MSE than Sigmoid, so we used RBF as our kernel.

Step 3: Pack the 3 predictors into 1
We built the "stacked" ensemble predictor that included KMeans, SVC, GradientBoostingRegressor from sklearn package are used. We used the default values 8 and 100 to be the number of clusters in KMeans and number of estimators in GradientBoostingRegressor respectively. First, the data sets were trained in KMeans, and then in SVC with C=0.8, gamma=0.1 and RBF kernel. The sets were boosted by using Gradient Boosting to have a better performance.

Step 4: Train the ensemble predictor
The data sets (Xtr, Ytr) and (Xte, Yte) are predicted separately in the predictor. The data set from X testing set was also trained for future prediction.

Step 5: Choose the best number of estimators for Gradient Boosting
Because Gradient Boosting used to robust to overfitting so a larger number of estimators usually results in better performance. Therefore, We tested the GradientBoostingRegressor (GBR) with different number of estimators to find the one with the highest AUC score (refer to Table 4.4). We chose 16000 estimators since the AUC score on validation data tended to grow very slow starting at 8000.

Step 6: Conclusion
AUC score on training data = 0.94748, AUC score on validation data = 0.75698, Kaggle score on test data = 0.76318

Data Page:

h value	MSE (Training data)	MSE (Validation data)
1	0.22478	0.27537
2	0.22480	0.22546
3	0.27368	0.22481
4	0.22548	0.22482
5	0.51241	0.51205
6	1.25718	1.25763
10	12.35137	11.92280

Table 1.1: table of MSE of training data and validation data for different h values

\	Least squares regression (LS)	Least absolute deviation (LAD)	Huber (LS & LAD)	Quantile regression
Validation AUC	0.70350	0.60408	0.70687	0.52816

Table 1.2: table of AUC score on validation data for different loss functions

Number of estimators	1000	2000	4000	8000
Validation AUC	0.72429	0.72434	0.72466	0.72390

Table 1.3: table of AUC score on validation data for different number of estimators in GBR

Estimator	AUC (Training data)	AUC (Validation data)
100	0.97684	0.69563
500	0.99353	0.69320
300	0.98511	0.69703
275	0.98133	0.69769
250	0.98089	0.69723

Table 2.1: table of AUC of training data and validation data for different estimators

Number of estimators	1000	2000	4000	8000
Validation AUC	0.75375	0.75674	0.76015	0.76335

Table 2.2: table of AUC score on validation data for different number of estimators in GBR

K value	MSE (Training data)	MSE (Validation data)
1	0.00467	0.40080
2	0.10193	0.30330
5	0.16406	0.24702
10	0.18345	0.22703
50	0.20062	0.21120
100	0.20439	0.21089
200	0.20642	0.21035
500	0.20862	0.21043
1000	0.21018	0.21114
5000	0.21910	0.21972
7500	0.22478	0.22555

Table 3.1: table of MSE of training data and validation data for different k values

Number of estimators	1000	2000	4000	8000	16000
Validation AUC	0.72377	0.73670	0.74631	0.75465	0.75676

Table 3.2: table of AUC score on validation data for different number of estimators in GBR

C value	MSE (Training data)	MSE (Validation data)
0.1	0.34627	0.32880
0.5	0.32653	0.32840
0.8	0.03373	0.32600
1	0.02373	0.33000
1.3	0.01680	0.33640
10	0.00600	0.34120
100	0.00427	0.34240

Table 4.1: table of MSE of training data and validation data for different c values

gamma value	MSE (Training data)	MSE (Validation data)
0.01	0.09413	0.32600
0.1	0.02880	0.32400
0.5	0.01200	0.32560
0.8	0.01107	0.32600
1	0.01093	0.32600
1.3	0.00533	0.32620
10	0.00387	0.32640

Table 4.2: table of MSE of training data and validation data for different gamma values

/	Sigmoid	RBF
MSE (Validation)	0.34080	0.32600

Table 4.3: table of MSE of validation data for different types of kernel.

Number of estimators	1000	2000	4000	8000	16000
Validation AUC	0.73099	0.74096	0.74850	0.75544	0.75698

Table 4.4: table of AUC score on validation data for different number of estimators in GBR