# Programming Part Report

# (a)

Classification: Logistic regression(linear) and decision tree(nonlinear) have similar accuracy, which are about 0.888 and about 0.91, while random forest can predict the result with 0.97 accuracy, which is higher than previous two. In my opinion, random forest can fit this task better because random forest ensembles lots of decision trees' results to predict the final results with more confidence and why it outperforms linear model may because the distribution of the data is too complex such that a simple linear model is hard to training within the similar training time; and hence it can perform better than logistic regression and decision tree.

Regression: Linear regression(linear)'s closed form solution has about 22.337 mse, decision tree(nonlinear) has about 17.39 mse, and random forest has about 10.46 mse. Random forest's performence is the best, decision tree the second and linear regression the last. In my opinion, it's because the distribution of the data is much more complex such that linear model can not fit this task well, and random forest outperforms decision tree is because random forest ensembles lots of sub decision trees' prediction to get a much more powerful final prediction.

Table 1: Experiment Parameters and Results

| model | linear | nonlinear | random forest |
|---|---|---|---|
| learning rates | 0.5 | X | X |
| iterations | 2000 | X | X |
| max depth | X | 5 | 5 |
| n_estimators | X | X | 100 |
| n_size | X | X | 2/3 |
| accuracy | 0.888 | 0.911 | 0.977 |
| MSE | 55.334 | 17.395 | 10.460 |

Note that n_estimators is the number of trees in random forest and n_size is the fraction of the data used in training trees divided by the overall data used in random forest and using $norm_X = (X - \min(X))/(\max(X) - \min(X))$ to normalize data.

# (b)

According to the table2, I find the results produced with normalized data are better than the ones with standardized data in general. In data, different variables often have different units and different domain range, to eliminate these impact, we use normalization and standardization. Sepcifically, in normalization, we map the data's domain image into $[0, 1]$, while in standardization, we transform the data into standard normal distribution.

In classification, because normalization just scaled the data propotionally, it doesn't change the distribution of the data, while the standardization changes the data's distribution into normal distribution even though the data's distribution is not a normal distribution, which may affect the classification significantly. However, compared to normalization, standardization is affected by the outliers more slightly(and this may be the cause that in logistic regression, standardization performs better than normalization; logistic regression may be affected by outliers more than other nonlinear models), so it may have less chance to overfitting due to few noisy data.

In Regression, normalization simply rescales the data and may be significantly impacted by some outliers, while in standardization, the significance of outliers will be reduced because the difference of data is measured by standard deviation. But also, if the data's distribution isn't similar with normal distribution, the benefits of standardization may be diminished and in such case, normalization may be more appropriate because it doesn't expect the distribution of given data belong to any distribution.

Table 2: Comparison with Normalization and Standardization

| model | linear | nonlinear | random forest |
|---|---|---|---|
| Acc with Normalization | 0.888 | 0.911 | 0.977 |
| Acc with Standardization | 0.911 | 0.866 | 0.866 |
| MSE with Normalization | 22.337 | 17.395 | 10.460 |
| MSE with Standardization | 25.755 | 28.650 | 13.805 |

Note that the MSE of linear model is calculated from closed form solution and the hyperparameters and normalization method are the same as table1.

# (c)

In logistic regression, as the learning rates and iterations become larger, the accuracy becomes higher; however, if learning rate is too large then weights may be affected too much such that they are optimized oscillatorily and hence slower the optimization process.

In linear regression, because there is a huge gap between training results and closed form solution, a large learning rates and more iterations can help to converge faster and more.

Table 3: Different Learning Rates and Iterations

| learning rates | iterations | logistic regression Acc | linear regression MSE |
|---|---|---|---|
| 0.01 | 1000 | 0.622 | 66.251 |
| 0.01 | 2000 | 0.622 | 63.732 |
| 0.01 | 10000 | 0.8 | 62.347 |
| 0.1 | 1000 | 0.8 | 62.347 |
| 0.1 | 2000 | 0.866 | 60.930 |
| 0.1 | 10000 | 0.888 | 55.335 |
| 0.5 | 1000 | 0.888 | 58.159 |
| 0.5 | 2000 | 0.888 | 55.334 |
| 0.5 | 10000 | 0.977 | 43.348 |
| 0.5 | 100000 | 1.0 | 31.471 |

# (d)

First, as max depth and n_estimators becomes larger, the model will become more complexity and therefore become time-consuming to training. However, if max dpeth become too large, the ability of generalization may be diminished and the potential for overfitting will increase. As the test 2. to the test 8., the max depth grows from 5 to 10 but the accuracy for classification is decreased. As for n_estimators, big n_estimators can bring in better generalization and decrease the potential for overfitting in my experiment.

Second, as sampled data size increases, the accuracy will increase and MSE will decrease. It's because the input data for each trees in random forest become bigger and hence can better fit the both tasks. This hyperparameter doesn't affect the complexity of the model but strengthen the ability of generalizations and may decrease the potential for overfitting because the correlation between each feature can be better formulated with more data.

Based on above, I first choose smaller max depth and n_estimators in order to fast check whether the performence is good enough. Then if I find the model is underfitting now, I will try to increase these two parameters carefully because it will become more difficult to training when increasing them and it's possible to overfit the input data if max depth becomes too large. As for sampled data size, I try to choose a moderate size because small size may underfit while large size may diminish the benefits of ensembling.

Table 4: Different Learning Rates and Iterations

| Test | max depth | n_estimators | sampled data size | random forest Acc | random forest MSE |
|---|---|---|---|---|---|
| 1. | 5 | 100 | 1/2 | 0.955 | 11.052 |
| 2. | 5 | 100 | 2/3 | 0.977 | 10.460 |
| 3. | 5 | 200 | 1/2 | 1.0 | 11.320 |
| 4. | 5 | 200 | 2/3 | 0.977 | 10.094 |
| 5. | 10 | 100 | 1/2 | 0.955 | 10.024 |
| 6. | 10 | 100 | 2/3 | 0.955 | 9.286 |
| 7. | 10 | 200 | 1/2 | 0.977 | 10.216 |
| 8. | 10 | 200 | 2/3 | 0.977 | 9.103 |

Note that n_estimators is the number of trees in random forest and sampled data size is the fraction of the data used in training trees divided by the overall data used in random forest.

# (e)

The strengths of linear models are that it's the simplest among these models, so the complexity of linear model is the smallest. The weakness of linear model is that it can't fit well if the distribution of the data is too complex for linear model to predict. The strengths of nonlinear model and random forest are they can deal with the complex situation where the distribution of the data is very complex. Moreover, random forest often outperforms nonlinear models because the trick of ensembling can merge lots of different results predicted by many decision trees in it. However, compared with nonlinear model, the weakness of random forest is the complexity of it is too large even if nonlinear model is more complex than linear model too; and both them are easy to overfitting than linear model. Hence, if the distribution of the data is very simple, I prefer linear model to the others because it is easy-training and it probably has better ability to generalization in this situation. If the distribution of the data is complex and I need to efficiently get a good enough solution, then nonlinear model is my first choice. If time isn't a problem and the distribution of the data is complex, I will choose random forest. Addtionally, if I want a human-explanable model, then decision tree(nonlinear model) is the best one for me, linear model the second because I can learn something from the different weights to different features, random forest the last because I think it's difficult to argue why these decision trees choose these thresholds and others choose thresholds differently. Based on the above discussion, the following table is a simple summary.

Table 5: Pros and Cons between Models

| Models | linear | nonlinear | random forest |
|---|---|---|---|
| Complexity | small | medium | large |
| Interpretability | second | best | probably the last |
| Performance | | Depends on situation | |