2 Programming

2.1

1. Data Preprocessing

After loading the data, I drop the first two columns that represent “station” and “date”. What’s more, since there are still some NAN values in the original data, I drop all the rows that have missing columns.

2. Model Selection

I use the Linear Regression with multiple outputs to estimate the parameters, then do the prediction.

3. How to estimate the parameter W?

Since the amount of data is still relatively small, I choose the closed form solution to estimate the parameter W, that is, to add an additional column which contains 1s to the left side of the training data, then add an additional row to the top of the W matrix that we need to estimate. Finally, compute the matrix $\widehat{W} = (X^T X)^{-1} X^T Y$ by the function that I defined in the code.

4. Model Evaluation

By the train_test_split function provided by sklearn, I set up 10 distinct random seeds in each iteration to generate different set of training data. Then I estimate the parameter, and predict the y_hat_test as well as y_hat_train. After that, I separately calculate the train and test RMSE for Next_Tmax and Next_Tmin. Finally, I take the average of the two-dimensional RMSE to generate the result.

The RMSE definition:

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(y_t - \widehat{y_t})^2}$$

Where y_t is the true value and $\widehat{y_t}$ is the predicted value for one column in the output matrix.

The result dataframe is shown below:

Index	Training_RMSE	Testing_RMSE
0	1.2293749	1.247396
1	1.2205682	1.2816744
2	1.225238	1.2626857
3	1.2365733	1.2175749
4	1.2327009	1.2332352
5	1.2289127	1.2478378
6	1.2333427	1.2309739
7	1.2313633	1.2382074
8	1.2201993	1.2809426
9	1.2330323	1.2321962

In summary, the model is quite robust between the training data and testing data. The temperature difference between the prediction and actual values are small. It's about 1.2-1.3 degree of error. So the model is relatively precise to do the regression of this data.

2.2

1. Data Preprocessing

Since there is no NAN value in the dataset, so there's no need to drop any value. Then, I transfer the class variables by get_dummies in pandas, which contain the value 1, 2 and 3, into one-hot encoding. Then the Y matrix contains 3 columns of binary values. After that, I continue to do the further analysis.

2. Model Selection

I use Linear Regression for classification to estimate the parameters, and do the prediction.

3. How to estimate the parameter W?

As for the measure in 2.1, I also use the closed form solution for 2.2. However, before the standard process of calculating $\widehat{W}$, I first change the Y variable to one-hot assignment. Then the remaining process is the same as 2.1.

4. Model Evaluation

By the train_test_split function provided by sklearn, I set up 10 distinct random seeds in each iteration to generate different set of training data. Then I estimate the parameter and calculate the regression result, and I use the argmax function provided by numpy to get the classification result. Finally I compute the error rate defined on the assignment requirement. Below is the result dataframe:

Index	Training_error	Testing_error
0	0.13333333	0.13333333
1	0.15833333	0.13333333
2	0.14166667	0.13333333
3	0.16666667	0.13333333
4	0.15	0.2
5	0.14166667	0.23333333
6	0.125	0.2
7	0.15833333	0.033333333
8	0.15833333	0.26666667
9	0.15	0.16666667

In summary, since two classes of the three aren't linearly separable, we cannot use the simple linear model to perfectly classify all the iris plants. Advanced non-linear model may be used to classify the data.