1. Explain briefly the knowledge supported your implementation and your design step by step.

For the function l2_rls_train, it will train a linear model. The knowledge I use is the formula of calculating the weight vector of l2 regularised least squares. The weight w is given by w = X.T @ X + H * I) ** (-1) @ X.T @ Y, where X is the data with 1s as the first column, Y is the label of X, H is the hyperparameter and I is the identity matrix. I first add 1s as the first column of the data. After that I create the identity matrix and check if the hyperparameter is 0. If it is, I will use pseudo inverse to calculate the weight vector and if not, I will use the normal way which is the formula above.

For the function l2_rls_predict, the knowledge I use is the predict function y = W.T @ X, where X is the data with 1s as the first column and W as the weight matrix. I first add 1s as the first column of the data. After that I use the formula above to create the output.

2. Explain the classification steps. Does changing the class labels impact the model performance? Explain why it does/doesn't impact. What training accuracies do you obtain with your linear classifier? Analyse the reason.

There are few steps in classification. Firstly, I split the data into training and testing set. Then I extract the data with class 1 and 30 from both sets. Thirdly, I set the label of the training and testing set to what the experiment required. Fourthly, I use the training set and label to train the model and get the weight vector. Lastly, I use the testing set to test the accuracy of the model.

Changing the class labels do not impact the model performance. The 3 experiments give similar result. The feature which affects the model performance is the hyperparameter. It is because the model will minimize the distance between the hyperparameter and the data.

I obtain 100% training accuracies for 3 experiments. It is because the model is train with the training data and accuracies, and the weight is based on the training data set and label. So if we apply training data to the model for testing, the weight will fit the training data totalling so the output will have 100% accuracy.

3. Analyse results of face completion model, how well your model performed, how it can be improved?

The result of the face completion model on left face testing set looks similar to the right face of the testing set. It has a mean absolute percentage error of 20.78 % (round off to 2 decimals place). To improve the model, I should find a better hyperparameter. To do this, I can ty different parameter and calculate the mean absolute percentage error. After that, I can find the optimal hyperparameter for the face completion model.

4. How did you choose the learning rate and iteration number, explain your results.

For the learning rates as 10**(-3) and iteration number as 200, I got a decreasing testing error as the number of iteration increases. The testing error decrease from 0.5 to 0. For learning rates as 10**(-2) and iteration number as 200, I got a stable testing error of 0.5 against all number of iterations. It is because 10**(-2) is too large to be a learning rate. So

the weight will be too large and affect the accuracy. A large learning rate may also result in inadvertently increase of training error. While a small learning rate will result in slower training and forever stuck with high training error. In conclusion, a too large or too small learning rate will result in lower accuracy.