3.2 Report

Explain briefly the knowledge supporting your implementation and your design step by step. Explicitly comment on the role of any arguments you have added to your functions.

The functions "l2\_rls\_train" and "l2\_rls\_predict" seems to be based on Ridge regression, which is a linear regression. And I use L2 regularization as requested to prevent overfitting of the model. The L2 regularization term adds a penalty to the loss function that is proportional to the squared size of the model parameters. By including this penalty term in the loss function, the model is encouraged to choose smaller parameter values, which helps prevent overfitting and improves generalization performance.

These functions take the input data and labels as numpy arrays, along with the regularization parameter "lambe" in the "l2\_rls\_train" function and return the trained model parameters or predicted output values as numpy arrays.

# assign lamda

lamda = lambe \* np.identity(X.shape[1] + 1)

For this code in the function “l2\_rls\_train”, He creates a diagonal matrix with rams on the diagonal and adds a row and a column of zeros to make the size of the matrix match the augmented data.

# Compute the coefficient vector.

if lambe == 0:

w = np.linalg.pinv(X\_tilde) @ y

else:

w = np.linalg.inv(np.transpose(X\_tilde) @ X\_tilde + np.dot(lamda, np.identity(X.shape[1] + 1))) @ (np.transpose(X\_tilde) @ y)

For this code in the function “l2\_rls\_train”, if lambe == 0, the model reverts to ordinary least squares regression. Overwise, the model parameters are obtained by solving a modified set of linear equations including an L2 regularization term. The "w" represents the trained model parameters and return to this function.

predicted\_y = np.dot(X\_tilde, w)

For this code in the function “l2\_rls\_predict”, the “predicted\_y”represents the output values for the input data using the trained machine learning model's "w".

4.2 Report

Explain the classification steps and report your chosen hyper-parameter and results on the test set. Did you notice any common features among the easiest and most difficult subjects to classify? Describe your observations and analyse your results.

The step for this section is below:

1. The data set was divided into a training set and a test set. The training set contains 5 randomly selected images from each subject, while the test set contains the rest of the images.
2. For the choice of hyperparameters, we used a k-fold cross-validation method with 5 folds. We tested different lambda values ranging from 0 to 100 in steps of 10. The best lambe value was chosen based on the average classification error rate for the 5-fold.
3. We train a model on the training set using selected lambe values and then test the model on the test set.
4. A 40 x 40 classification error matrix (confusion matrix) was constructed for the test sample. The confusion matrix shows the number of correct and incorrect predictions for each subject.

5.2 Report­­­­

Report the MAPE and make some observations regarding the results of the face completion model. How well has your model performed? Offer one suggestion for how it can be improved.

6.3 Report

Analyse the impact that changing the learning rate has on the cost function and obtained testing accuracies over each iteration in experiment 6.2. Drawing from what you observed in your experiments, what are the consequences of setting the learning rate and iteration number too high or too low?

7.3 Report

Explain in the report your experiment design, comparative result analysis and interpretation of obtained results. Try to be thorough in your analysis.

\*\* Remember that all graphs should have axis labels and a title. \*\*

7.3 Report

Explain in your report the following:

(1) Your implementation of `hinge\_gd\_train`. If you analytically derived the loss function, please include it here.

(2) Your experiment design, comparative result analysis and interpretation of obtained results.

\*\* Remember that all graphs should have axis labels and a title. \*\*