All these code can be easily run if you can create a Conda environment with the following packages: Numpy, Pandas, mathplotlib, tensorflow, scipy and scikit-learn, joblib and Jupyternotebook.

__Logistic Regression Models__
There are two logistic regression codes
01_Logistic_Regression_With_ValSplit_Final.py **and**
01_Logistic_Regression_Without_ValSplit_Final.py

These code can be used to train logistic regression model. User need to provide the three parameters lr, epochs and lamb in the main functions (Line number 208-210).

The code will provide training and validation accuracy, total correct and wrong predictions, misclassification rates, plot of the losses during training. 01_Logistic_Regression_Without_ValSplit_Final.py will not provide validation accuracy details as there is no train-validation data split.

In addition, three Jupyternotebook were added as appendix files:

Appendix00_01_Logistic_Regression_With_ValSplit_Final.ipynb
Appendix00_01_Logistic_Regression_Without_ValSplit_Final.ipynb
Appendix00_LR_HyperParameter_Search_WithValSplit.ipynb

The first two are jupyternotebook of the logistic regression code given above. The third one shows the hyperparameter searches that I have performed. You can simply run the notebook if you want to reproduce the results.

__CNN Models:__

The best model was uploaded as ***02_CNN_Model4.ipynb***. The user needs to just run the code. It will produce output including model summary, training and validation losses and accuracy and their plots. Other metrics such as confusion metrics and classification reports can be obtained by simply running the notebook. Also, it generates the test predictions on the test data.

In addition, I have also uploaded other CNN models with prefix Appendix01, in case if someone wants to run it.
Appendix01_CNN_Model1_withValSet_EarlyStopping.ipynb
Appendix01_CNN_Model2-withValSet_Dropout-BatchNorm_EarlyStopping.ipynb
Appendix01_CNN_Model3-Dropout-BatchNorm-Noearly-Stopping-withoutVal.ipynb

__Random Forest Model__:

Use the jupternotebook ***02_Random_Forest_Final-Pickle.ipynb*** for running the code. The code will perform randomized grid search for the Random Forest Classifier. Due to memory issues in the laptop, I have ran the grid search another hpc cluster. For users, who want to rerun the code for grid search need to remove the commented out lines.

However, I have provided the best parameters from grid search as rf_random_best_params.pkl file. You do not need to rerun the code. The model will train on the best params stored in the pickle file. The code will also retrain the entire dataset with the best parameter values and this model is used for test predictions.

I have also uploaded another jupyternotebook Appendix02_Random_Forest-Train_Val_Split.ipynb. This notebook contains code for randomized grid search with smaller numbers of parameter spaces. This code was used for initial RF test predictions in the Kaggle. However, the main code ***02_Random_Forest_Final-Pickle.ipynb*** has the best parameters.

The user needs to simply run the code and outputs will generated.