Description:

This program uses ridge regression, lasso regression, SGD regression and multi-layer perceptron to predict the prices of electronic products. It contains three python files, preicePred.py, models_function.py and cross_validation_function.py. It also contains a csv file, DEPPData.csv, which includes data for 15000 electronic products. This program will find the best features and best hyperparameter for different. Also, it will compare the models to find out the best model by accuracy.

Specification:

1. pricePred.py:

It contains the main() function of the program. You can run this program by "python pricePred.py" or "python3 pricePred.py". However, this program will cost large portion of time to run because of the MLP model. If you want to run the program, I comment out the code about the grid search function of the MLP model by default because it can save you hours of time. I already run the grid search cross validation for all models and save the best hyperparameters. However, if you want to run the grid search, you can uncomment the code. You can see my comments in codes to learn how to uncomment that out. It will save your time.

a. processing_data():

   This data processing function, which can take the data which contains about 15000 data of electronic products. It will drop the useless data and split the dataset into training set and test set. Then, return the datasets

b. features_select():

   This function will use the forward selecting method to select the best features. This will return the accuracy for every features or features' combinations.

c. selected_features_helper():

   This helper function will use ridge regression to calculate the accuracy for specific dataset. It is called insider of features_select() function.

d. main():

   This is the main function for the program. It will run data processing, k fold cross validation, grid search validation and plot the graph. Also, it will print out the time cost for each section. If you run this program by command line. the program will be passus when the plotted graph pops up. So, you need to close the graph to continue the program. If you run this in "Spyder", it will not be passus.

2. models_funcrion.py:

This file contains all models and model function which can be used to predict the prices

with the best hyperparameters. It also contains a plot function, which can plot the accuracy vs number of epochs.

a. multi_layer_perceptron_sk():
   Using sklearn to build a multi-layer perceptron. It takes training data, test data and number of epochs to predict prices and accuracy. Then return the predict prices and accuracy.
b. lasso_sk():
   Using sklearn to build a lasso regression model. It takes training data, test data and number of epochs to predict prices and accuracy. Then return the predict prices and accuracy.
c. ridge_sk():
   Using sklearn to build a ridge regression model. It takes training data, test data and number of epochs to predict prices and accuracy. Then return the predict prices and accuracy.
d. sgdregressor_sk():
   Using sklearn to build an SGD regression model. It takes training data, test data and number of epochs to predict prices and accuracy. Then return the predict prices and accuracy.
e. draw_acc_epoch():
   Using matplotlib.pyplot to draw the graph about the relation between accuracy and number of epochs. It takes training data, test data and a name of model to draw graph. The name of model have to be one of the ['sgd','mlp','ridge','lasso'].

3. cross_validation_function.py:

This file contains two functions that run k fold cross validation and grid search cross validation for all models.

a. k_fold_validation():
   Using sklearn to make a k fold cross validation function for all models. The regressor in this function is initialized with the best hyperparameters by default. You also can try to delete the arguments in regressor initializer to get the accuracy for each model before grid search. This function will print out the time cost for each model. And return the accuracy.
b. grid_search_sk():
   This function takes training data and test data to run the grid search cross validation for all models. The time cost for MLP models would be hours. So, if you want to run this function, you can try to comment out the code for MLP section. It will return best accuracy for each model with the best hyperparameters.