Multiple Regression

# example code

|  |
| --- |
| #import list  import pandas as pd  import numpy as np  from sklearn.linear\_model import LinearRegression  from sklearn.metrics import mean\_squared\_error  from sklearn import preprocessing  from sklearn.model\_selection import train\_test\_split  from sklearn.metrics import r2\_score  import matplotlib.pyplot as plt  # Load data\_set  data = pd.read\_csv("../[01]data\_set/imports-85.data",na\_values = '?')  # Set columns  data.columns = ['symboling','normalized-losses','make','fuel-type','aspiration','num-of-doors','body-style','drive-wheels','engine-location','wheel-base','length','width','height','curb-weight','engine-type','num-of-cylinders','engine-size','fuel-system','bore','stroke','compression-ratio','horsepower','peak-rpm','city-mpg','highway-mpg','price']  # option 0 : delete all nan values row.  '''data = data.dropna(axis=0)'''  # option 1 : replace all nan values with mean values of columns  df = data[["make","engine-size","highway-mpg","city-mpg","price"]]  df = df.groupby("make").transform(lambda x: x.fillna(x.mean()))  # In this sample code use option 1  # Set X,Y data\_set  X = df[["engine-size","highway-mpg","city-mpg"]]  y = df["price"]  # def funtion : draw Learning\_curves in this Data set.  def plot\_learning\_curves(model, X, y, X\_train, X\_val, y\_train, y\_val):      train\_errors, val\_errors = [], []      for m in range (1, len(X\_train)):          model.fit(X\_train[:m], y\_train[:m])          # Train model with x\_train[0~m],y\_train[0~m]          y\_train\_predict = model.predict(X\_train[:m])          # Check model with x\_train[0~m]          y\_val\_predict = model.predict(X\_val)          # Check model with x\_val[0~m]          train\_errors.append(mean\_squared\_error(y\_train[:m], y\_train\_predict))          val\_errors.append(mean\_squared\_error(y\_val, y\_val\_predict))          # add train and val error in list.      # Draw learning curves.      plt.plot(np.sqrt(train\_errors), "r-+", linewidth = 2, label = "train set")      plt.plot(np.sqrt(val\_errors), "b-", linewidth = 3, label = "validation set")      plt.xlabel("size of train set")      plt.ylabel("RMSE")      plt.legend()  # New linear model  model = LinearRegression()  # Split Dataset into Trainset and val\_set  X\_train, X\_val, y\_train, y\_val = train\_test\_split(X,y, test\_size=0.2,random\_state= 0)  # Draw learning curves  plot\_learning\_curves(model,X,y,X\_train, X\_val, y\_train, y\_val)  # New linear model  model\_1 = LinearRegression()  # Train model  model\_1.fit(X\_train,y\_train)  # Make y\_pred list with x\_val list  y\_pred = model\_1.predict(X\_val)  # Check Suitable of y\_pred and y\_val  score = r2\_score(y\_val,y\_pred)  print(score) |

# testing result

|  |
| --- |
|  |