import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import cross\_val\_score, GridSearchCV, train\_test\_split

from sklearn.linear\_model import Ridge

from sklearn.metrics import mean\_squared\_error

from sklearn.svm import SVR

from sklearn.svm import NuSVR

##############################################################################

data = pd.read\_csv("data.csv")

ftest= data.iloc[:,0].values

ftrain= data.iloc[:,1].values

Xtest = data.iloc[:, 2:20].values

Xtrain = data.iloc[:, 21:39].values

ytest = data.iloc[:, 40].values

ytrain = data.iloc[:, 41].values

print(ftest)

print(ftrain)

scaler = StandardScaler()

Xtrain = np.column\_stack((Xtrain, ftrain))

Xtest = np.column\_stack((Xtest, ftest))

X\_train, X\_val, y\_train, y\_val = train\_test\_split(Xtrain, ytrain, test\_size=0.2)

plt.scatter(ftest,ftrain)

###########################################################################

regressor = SVR(kernel='rbf')

regressor.fit(Xtrain,ytrain)

y\_pred = regressor.predict(Xtest)

lamda = np.logspace(-2, 1, 100)

mse\_values = []

ridge = Ridge()

for i in lamda:

    ridge.alpha = i

    ridge.fit(X\_train, y\_train)

    y\_pred\_val = ridge.predict(X\_val)

    val\_mse = mean\_squared\_error(y\_val, y\_pred\_val)

    mse\_values.append(val\_mse)

optimal\_alpha = lamda[np.argmin(mse\_values)]

final\_ridge\_model=Ridge(alpha=optimal\_alpha)

final\_ridge\_model.fit(Xtrain, ytrain)

y\_pred\_test = final\_ridge\_model.predict(Xtest)

test\_mse = mean\_squared\_error(ytest, y\_pred\_test)

print(f'Optimal Alpha: {optimal\_alpha}')

print(f'Optimal Validation MSE: {min(mse\_values)}')

print(f'Test Mean Squared Error: {test\_mse}')

####################################################################################################

plt.figure(figsize=(10, 5))

plt.subplot(1, 3, 1)

plt.plot(lamda, mse\_values, '--', color='blue')

plt.xlabel('Alpha (Lambda)')

plt.ylabel(' MSE')

plt.xscale('log')

plt.title(' MSE v Alpha')

plt.subplot(1, 3, 2)

plt.scatter(ytest, y\_pred\_test, color='blue', label='Predicted')

plt.plot([min(ytest), max(ytest)], [min(ytest), max(ytest)], linestyle='--', color='red', label='Actual')

plt.xlabel('Actual Data')

plt.ylabel('Predicted Data')

plt.title('Actual vs Predicted')

plt.subplot(1, 3, 3)

Xtrain.size

plt.scatter(Xtrain, ytest, color = 'red')

plt.plot(Xtest, y\_pred, color = 'green')

plt.tight\_layout()

plt.show()

A screenshot of a computer

Description automatically generated

import numpy as np

import pandas as pd

from sklearn.metrics import mean\_squared\_error

from sklearn.preprocessing import StandardScaler

from sklearn.model\_selection import cross\_val\_score, GridSearchCV, train\_test\_split

from sklearn.linear\_model import Ridge

from sklearn.svm import SVR

from sklearn.svm import NuSVR

import matplotlib.pyplot as plt

import seaborn as sns

data = pd.read\_csv("data.csv")

ftest= data.iloc[:,0]

ftrain= data.iloc[:,1]

Xtest = data.iloc[:, 2:20].values

Xtrain = data.iloc[:, 21:39].values

ytest = data.iloc[:, 40].values

ytrain = data.iloc[:, 41].values

X\_train, X\_val, y\_train, y\_val = train\_test\_split(Xtrain, ytrain, test\_size=0.2)

SVM\_regrssion = SVR()

SVM\_regrssion.fit(Xtrain,ytrain)

yhat = SVM\_regrssion.predict(Xtest)

plt.subplot(1,2,1)

sns.scatterplot(x=ytest,y=yhat,alpha=.6,color = 'green')

sns.lineplot(x=ytest,y=ytest,color = 'red')

test(Xtest,ytest)

A screenshot of a computer

Description automatically generated