**PART A**

(PART A : TO BE REFFERED BY STUDENTS)

**EXPERIMENT NO. 7**

**A.1 AIM: -** To Implementing Predicting Energy Efficiency for Residential Buildings

**A.2 Prerequisite**

* Different programming language (Python or Java), Understanding of Machine Learning Algorithms, Machine Learning Algorithms

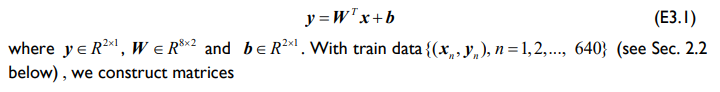
**A.3 Outcome**

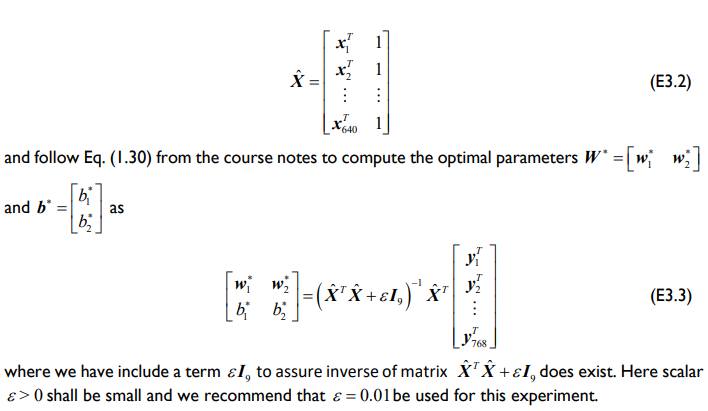
After successful completion of this experiment students will be able to Optimize the problem.

**A.4 Theory**

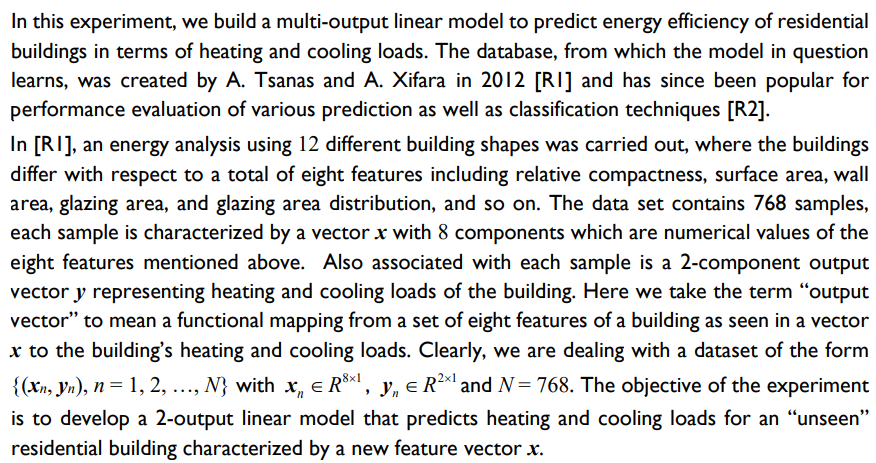
**Multi-output linear model for prediction**

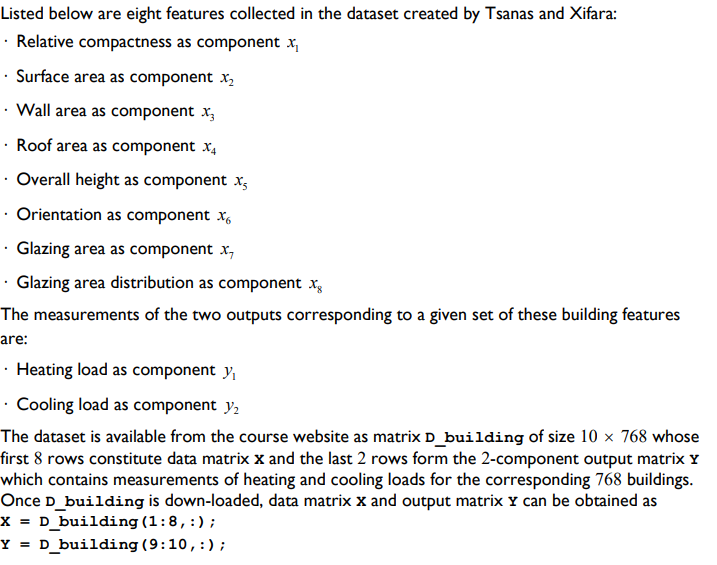
The linear model of interest is given by

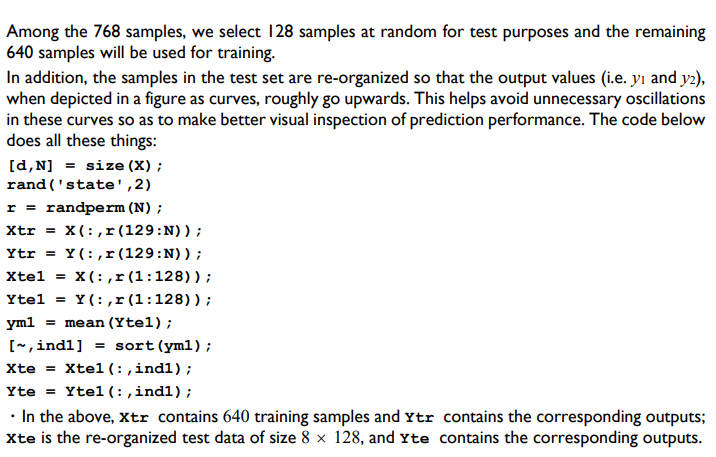


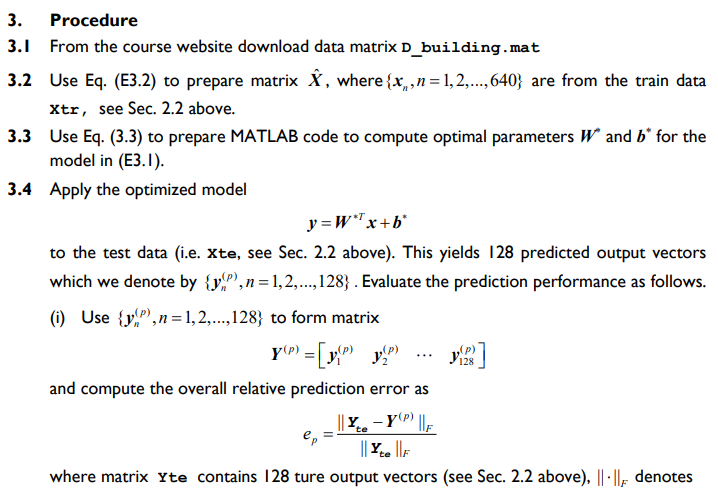


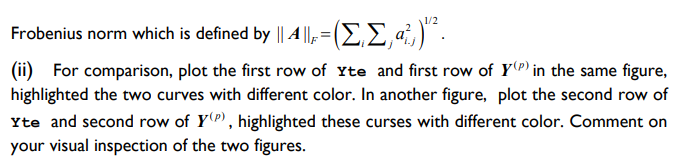
**A5. Task**











Links: <http://archive.ics.uci.edu/ml/datasets/Energy+efficiency?ref=datanews.io>

Or

<https://www.kaggle.com/datasets/elikplim/eergy-efficiency-dataset>

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case there is no Black board access available)***

|  |  |
| --- | --- |
| Roll No.C015 | Name:Prachi Dave |
| Class :B | Batch :EB1 |
| Date of Experiment: | Date of Submission |
| Grade : |  |

**B.1 Documentation written by student:**

# %% [markdown]

# To Implementing Predicting Energy Efficiency for Residential Buildings

# %%

import pandas as pd

from sklearn.multioutput import MultiOutputRegressor

from sklearn.linear\_model import Ridge, Lasso, LinearRegression

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import mean\_squared\_error, r2\_score

import numpy as np

import matplotlib.pyplot as plt

df = pd.read\_excel('ENB2012\_data.xlsx')

df

# %%

corr = df.corr()

X,y = df.iloc[:,:-2].values.T, df.iloc[:,-2:].values.T

# %%

d, N = X.shape

np.random.seed(2)

r = np.random.permutation(N)

Xtr = X[:, r[128:]]

Ytr = y[:, r[128:]]

Xtel = X[:, r[:128]]

Ytel = y[:, r[:128]]

ym1 = np.mean(Ytel, axis=1)

ind1 = np.argsort(ym1)

Xte = Xtel[:, ind1]

Yte = Ytel[:, ind1]

Xtr\_tilde = np.vstack([Xtr, np.ones((1, Xtr.shape[1]))])

W = np.linalg.lstsq(Xtr\_tilde.T, Ytr.T, rcond=None)[0]

Xte\_tilde = np.vstack([Xte, np.ones((1, Xte.shape[1]))])

Yp = np.dot(W.T, Xte\_tilde)

error = np.linalg.norm(Yte - Yp, 'fro') / np.linalg.norm(Yte, 'fro')

plt.figure(figsize=(10, 8))

plt.subplot(2, 1, 1)

plt.plot(Yte[0, :], 'b', linewidth=2)

plt.plot(Yp[0, :],'r--', linewidth=2)

plt.title('Heating Load Prediction')

plt.legend(['True Heating Load', 'Predicted Heating Load'])

plt.figure(figsize=(10, 8))

plt.subplot(2, 1, 2)

plt.plot(Yte[1, :], 'g', linewidth=2)

plt.plot(Yp[1, :],'m--', linewidth=2)

plt.title('Cooling Load Prediction')

plt.legend(['True Cooling Load', 'Predicted Cooling Load'])

# %%

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X.T,y.T,random\_state = 42, test\_size = 0.2)

regr = MultiOutputRegressor(Ridge(random\_state=123)).fit(X\_train, y\_train)

linear = MultiOutputRegressor(LinearRegression()).fit(X\_train, y\_train)

lasso = MultiOutputRegressor(Lasso(alpha=4)).fit(X\_train, y\_train)

models = [regr, linear, lasso]

# %%

preds = []

names = { 1: "Ridge",

2: "Linear Regression",

3: "Lasso"

}

for i,model in enumerate(models):

    preds = model.predict(X\_test)

    print(f"The MSE for {names[i+1]} is: {mean\_squared\_error(y\_test,preds)}")

    print(f"The R2 score for {names[i+1]} is: {r2\_score(y\_test,preds)}")

    print("-"\*20)

    print()

y\_pred = linear.predict(X\_test)

# %%

plt.figure(figsize=(10, 8))

plt.subplot(2, 1, 1)

plt.plot(y\_test[0, :], 'b', linewidth=2)

plt.plot(y\_pred[0, :],'r--', linewidth=2)

plt.title('Heating Load Prediction')

plt.legend(['True Heating Load', 'Predicted Heating Load'])

plt.figure(figsize=(10, 8))

plt.subplot(2, 1, 2)

plt.plot(y\_test[1, :], 'g', linewidth=2)

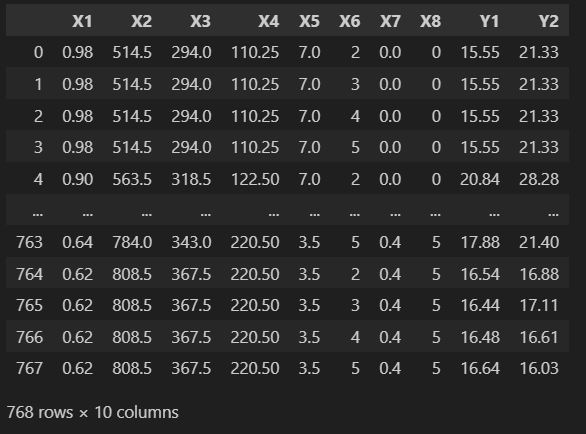
plt.plot(y\_pred[1, :],'m--', linewidth=2)

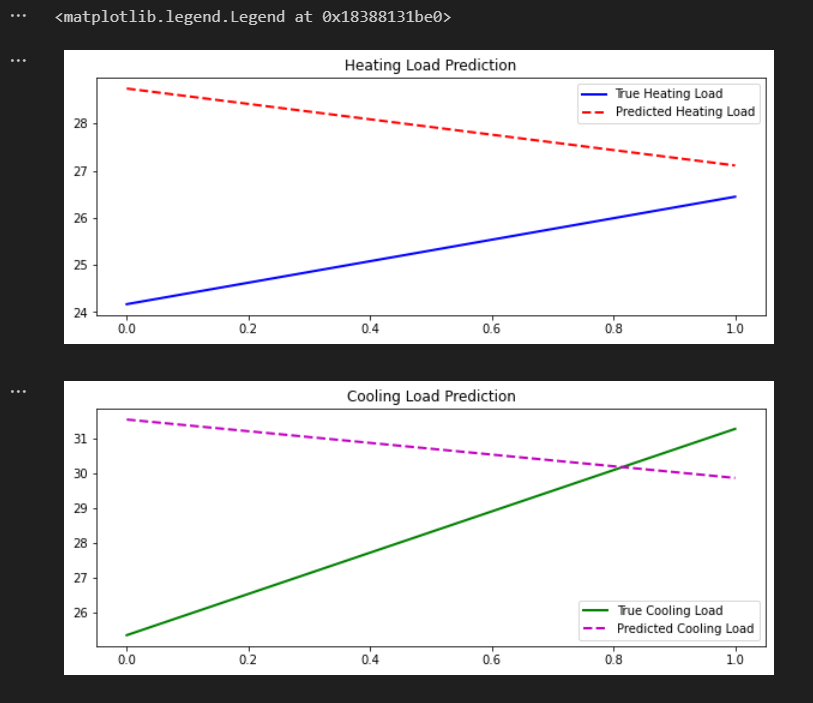
plt.title('Cooling Load Prediction')

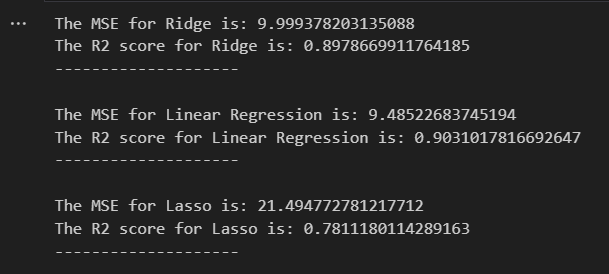
plt.legend(['True Cooling Load', 'Predicted Cooling Load'])

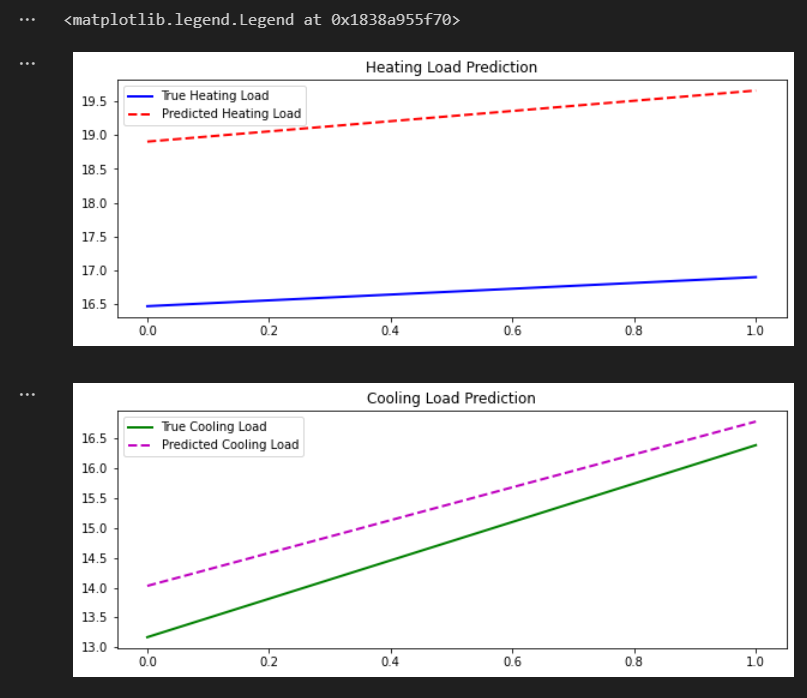
# %%

**B.2 Observations and learning:**

******

******





**B.3 Conclusion:**

* Ridge regression achieved the lowest Mean Squared Error (MSE) of 9.999 and a relatively high R2 score of 0.898, indicating good performance in balancing bias and variance.
* Linear regression performed slightly better than Ridge with a lower MSE of 9.485 and a higher R2 score of 0.903.
* Lasso regression showed poorer performance with a higher MSE of 21.495 and a lower R2 score of 0.781, indicating potential overfitting or inadequate feature selection.