**PART A**

**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

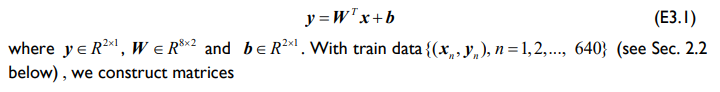
**A.3 Outcome**

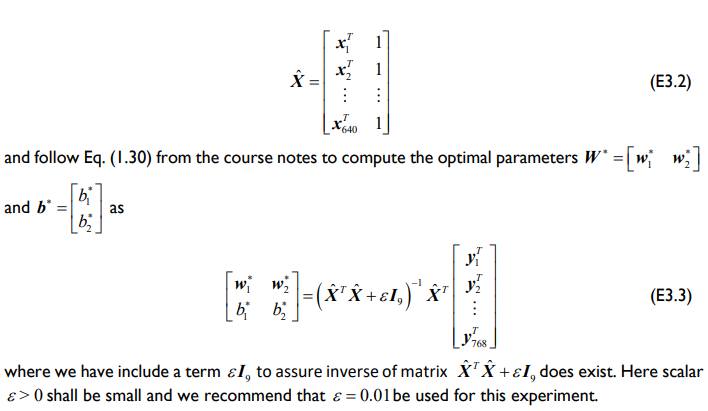
After successful completion of this experiment students will be able to Optimize the problem.

**A.4 Theory**

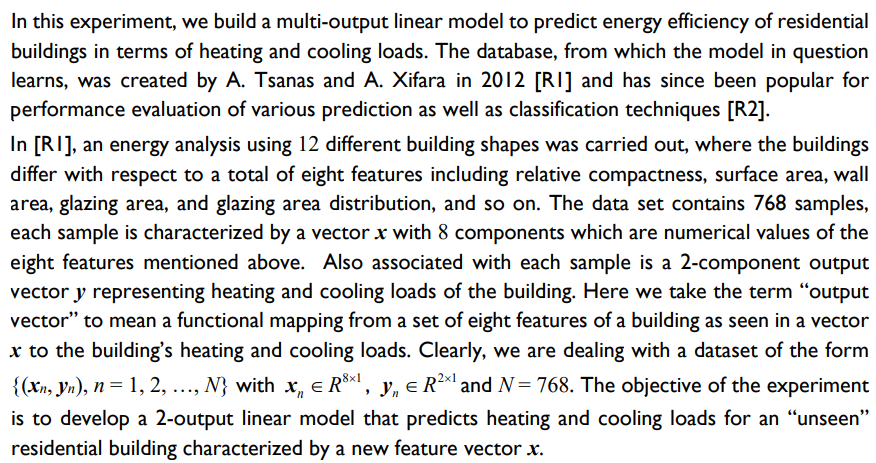
**Multi-output linear model for prediction**

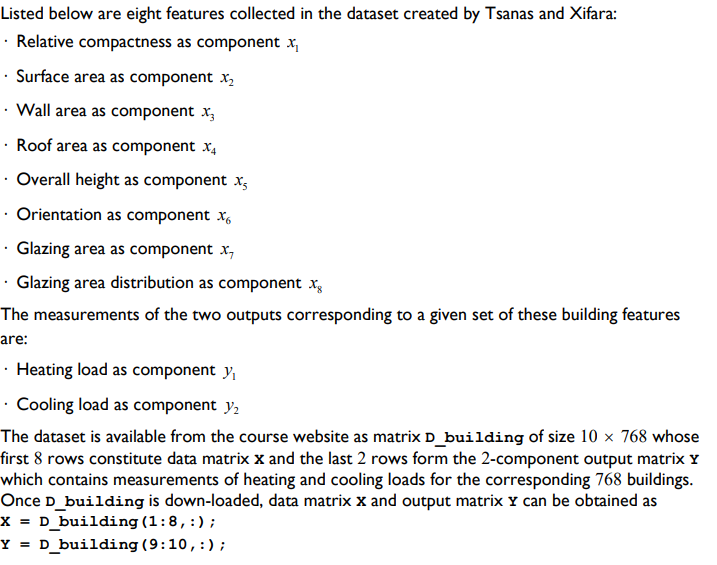
The linear model of interest is given by

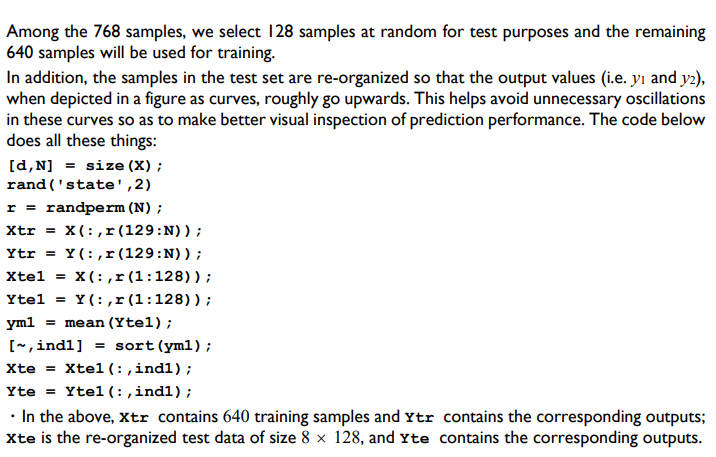


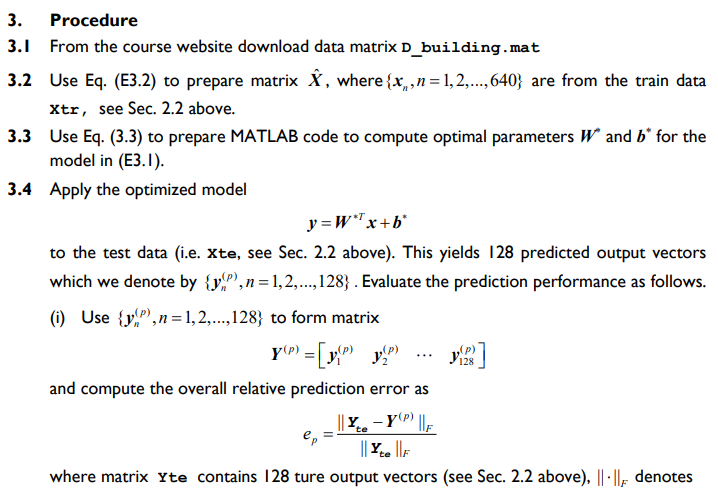


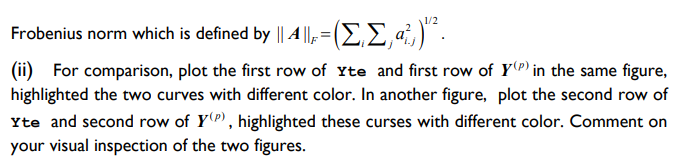
**A5. Task**











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

|  |  |
| --- | --- |
| Roll No. C009 | Name: Samarth Borade |
| Class : BTI SEM 10 | Batch : EB1 |
| Date of Experiment: 08/03/24 | Date of Submission |
| Grade : |  |

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

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

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

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

df = pd.read\_csv('exp7\_data.csv')

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

Y = df.iloc[:, -2:].values

# Manually split the data into training and test sets

np.random.seed(2)

r = np.random.permutation(len(X))

X\_train, X\_test = X[r[128:]], X[r[:128]]

Y\_train, Y\_test = Y[r[128:]], Y[r[:128]]

# Initialize linear regression models

linear\_model = LinearRegression()

lasso\_model = Lasso(*alpha*=0.1)

ridge\_model = Ridge(*alpha*=0.1)

linear\_model.fit(X\_train, Y\_train)

lasso\_model.fit(X\_train, Y\_train)

ridge\_model.fit(X\_train, Y\_train)

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

lasso\_pred = lasso\_model.predict(X\_test)

ridge\_pred = ridge\_model.predict(X\_test)

linear\_mse = mean\_squared\_error(Y\_test, linear\_pred)

lasso\_mse = mean\_squared\_error(Y\_test, lasso\_pred)

ridge\_mse = mean\_squared\_error(Y\_test, ridge\_pred)

error = np.linalg.norm(Y\_test - linear\_pred) / np.linalg.norm(Y\_test)

linear\_mae = mean\_absolute\_error(Y\_test, linear\_pred)

lasso\_mae = mean\_absolute\_error(Y\_test, lasso\_pred)

ridge\_mae = mean\_absolute\_error(Y\_test, ridge\_pred)

error2 = np.linalg.norm(Y\_test - lasso\_pred) / np.linalg.norm(Y\_test)

linear\_r2 = r2\_score(Y\_test, linear\_pred)

lasso\_r2 = r2\_score(Y\_test, lasso\_pred)

ridge\_r2 = r2\_score(Y\_test, ridge\_pred)

error3 = np.linalg.norm(Y\_test - ridge\_pred) / np.linalg.norm(Y\_test)

# Print results

print("Linear Regression:")

print("MSE:", linear\_mse)

print("MAE:", linear\_mae)

print("R2 Score:", linear\_r2)

print("Overall prediction error:", error)

print()

print("Lasso Regression:")

print("MSE:", lasso\_mse)

print("MAE:", lasso\_mae)

print("R2 Score:", lasso\_r2)

print("Overall prediction error:", error2)

print()

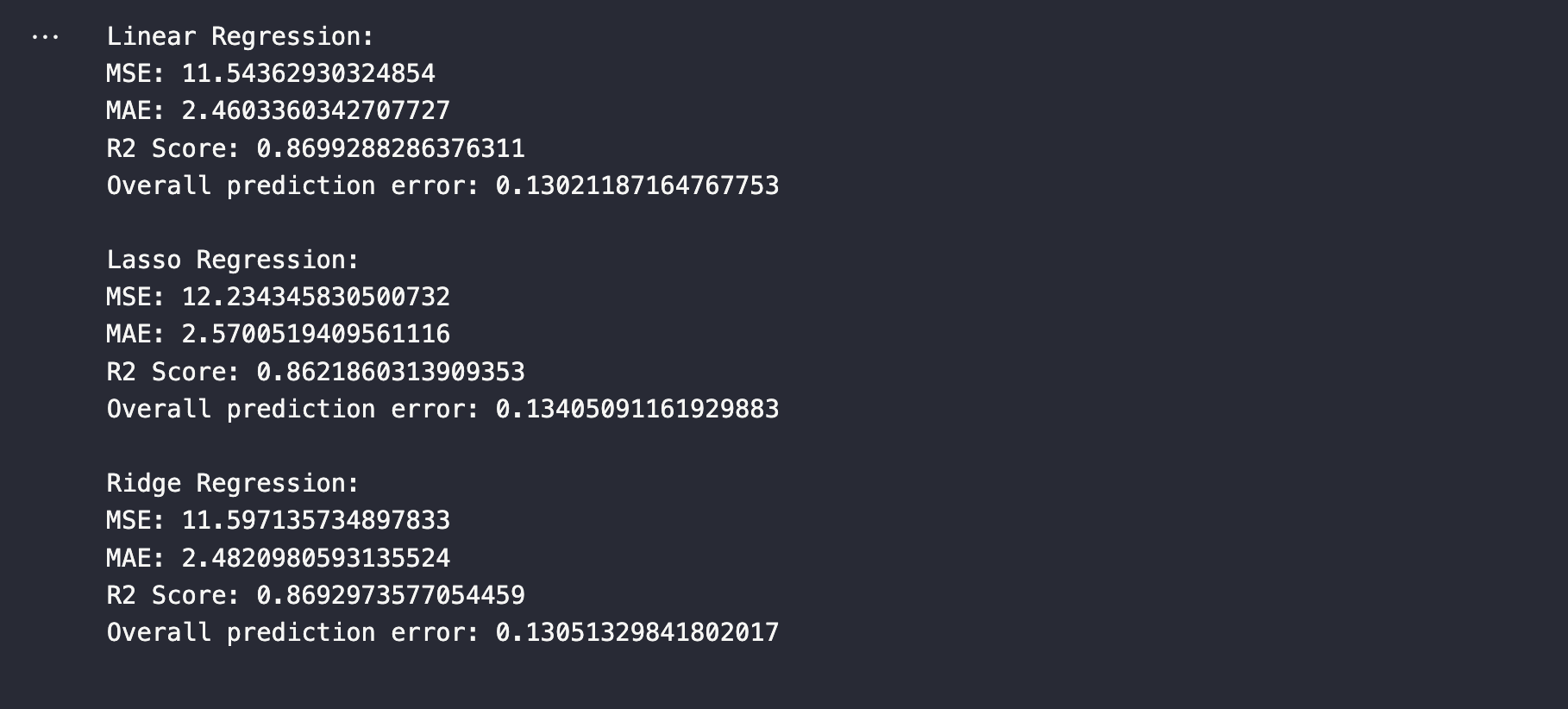
print("Ridge Regression:")

print("MSE:", ridge\_mse)

print("MAE:", ridge\_mae)

print("R2 Score:", ridge\_r2)

print("Overall prediction error:", error3)

****

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

plt.subplot(2, 1, 1)

plt.plot(Y\_test[:, 0], 'b', *linewidth*=2, *label*='Actual')

plt.plot(linear\_pred[:, 0], 'r--', *linewidth*=2, *label*='Predicted')

plt.title('Heating Load Prediction')

plt.legend()

plt.subplot(2, 1, 2)

plt.plot(Y\_test[:, 1], 'g', *linewidth*=2, *label*='Actual')

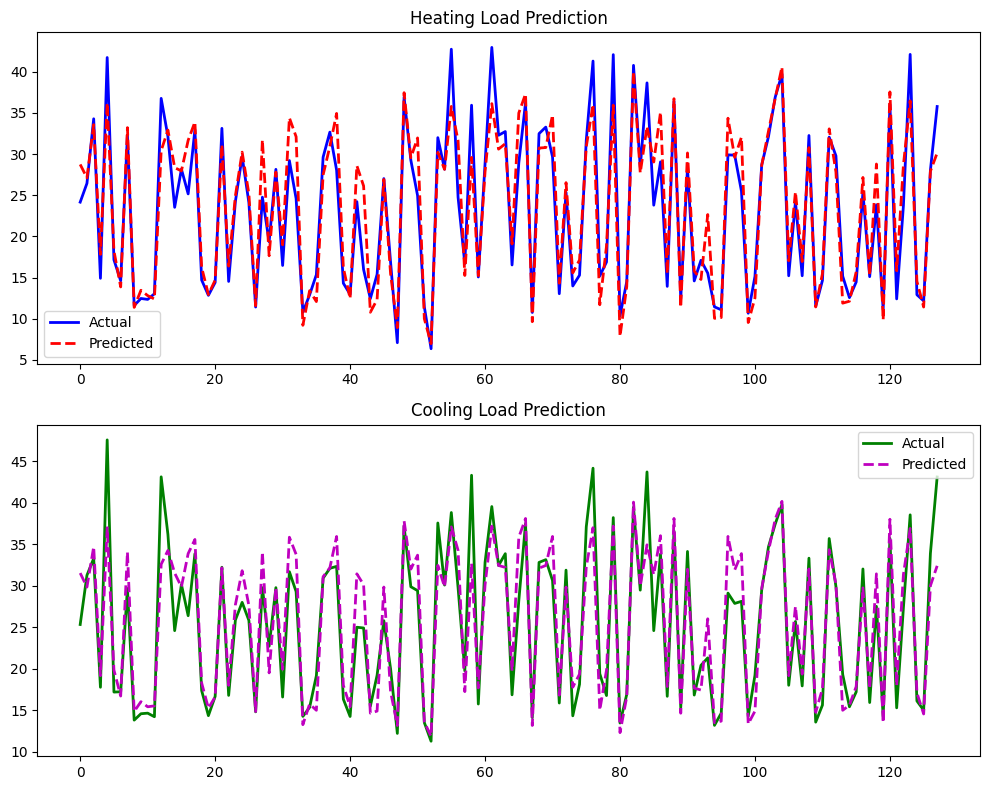
plt.plot(linear\_pred[:, 1], 'm--', *linewidth*=2, *label*='Predicted')

plt.title('Cooling Load Prediction')

plt.legend()

plt.tight\_layout()

plt.show()



**B.2 Observations and learning:**

I've worked on predicting heating and cooling loads for residential buildings using linear regression models. The data was manually split into training and test sets, a common practice for moderate-sized datasets. My model was trained using linear regression, and I calculated the relative prediction error using the **Frobenius norm**. Additionally, I employed a multi-output linear regression model using scikit-learn's **MultiOutputRegressor**, which simplifies the task and potentially improves predictions, especially with correlated output variables like heating and cooling loads. Both models were evaluated by visualizing the comparison between actual and predicted values for both heating and cooling loads. This allowed for a better understanding of model performance.

**B.3 Conclusion:**

Overall, my approach demonstrated effective methods for predicting heating and cooling loads. By employing a multi-output linear regression model, I could handle multi-output regression tasks more efficiently, potentially leading to better predictions. Additionally, visualizing the comparison between actual and predicted values provided valuable insights into the performance of the models.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*