NATIONAL INSTITUTE OF TECHNOLOGY

ANDHRA PRADESH

DEPARTMENT OF

ELECTRICAL AND ELECTRONICS ENGINEERING

SHORT-TERM ELECTRICITY LOAD DEMAND PREDICTION

Guide:

Dr. Sri Phani Krishna

Team:

A. Parameswarudu (520106)

B. Bhoomika (520117)

CH. Rupesh (520118)

G. Uday Chaitanya (520131)

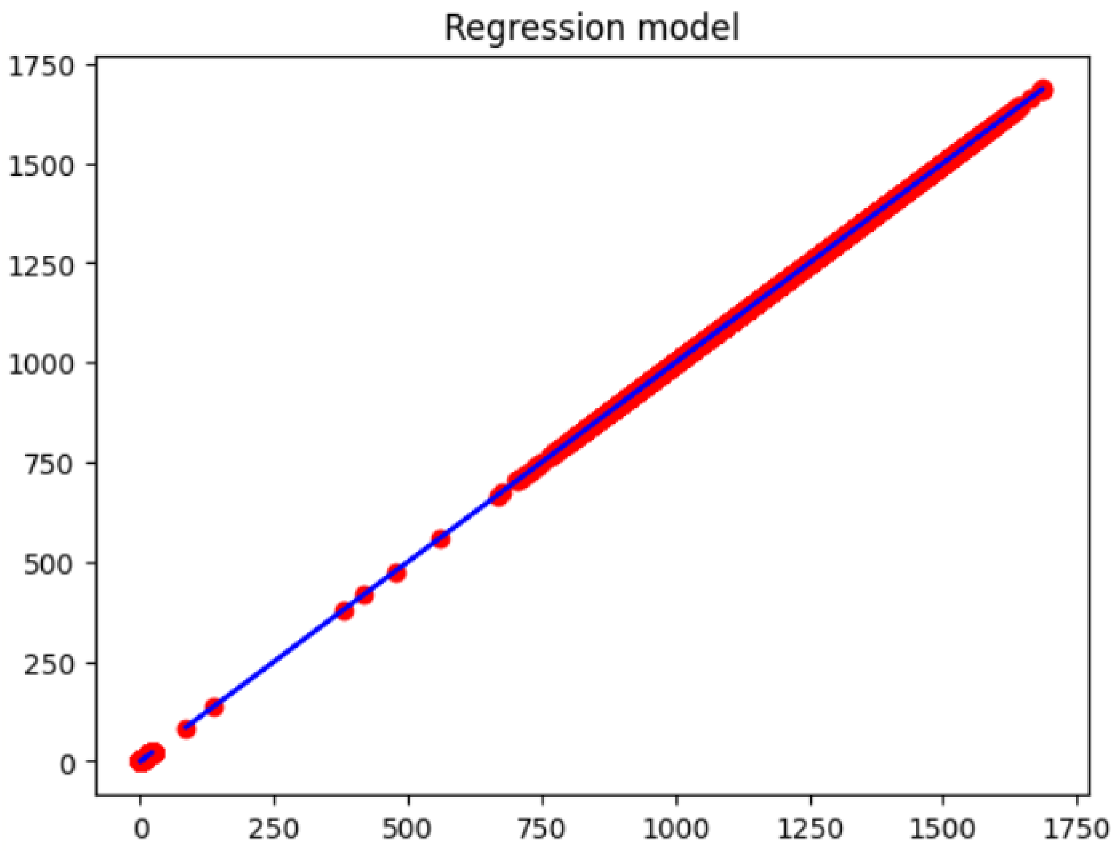
K. Manjusha (520148)

Problem statement:

As energy demand is increasing globally, the energy management system is becoming more important. Electricity consumption is one among thing which is rapidly increasing. As for of Resource planning like accurate predictions of electricity consumption are necessary which involves determining the amount of electricity that will be needed in a given period of time this resource planning helps and utilises to ensure that they have enough electricity generation capacity to meet the demand. And next by electricity consumption prediction one can be able to reduce the operating costs by energy providers so that they can optimise their generation production, etc and It is used to balance the load on power grid so that power outages, risk of blackouts can be reduced and mainly by predicting electricity consumption it helps to notify the people in advance and help to conserve the energy during those periods thus helps in saving of money on electricity bills and reduce their overall electricity consumption everyday

DecisionTreeRegressor:

```
from sklearn.tree import DecisionTreeRegressor     #importing the regressor
regressor = DecisionTreeRegressor(random_state=0)  #creating a regressor object for the decision tree
X= df_train.iloc[:,1:]
y = df_train.iloc[:,1:]
X = X.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-3",'holiday','Holiday_ID'],axis=1) #droping the features that have high VIF values (>5)
y =y.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-3",'holiday','Holiday_ID'],axis=1)
regressor.fit(X,y)
```



MLP regression:

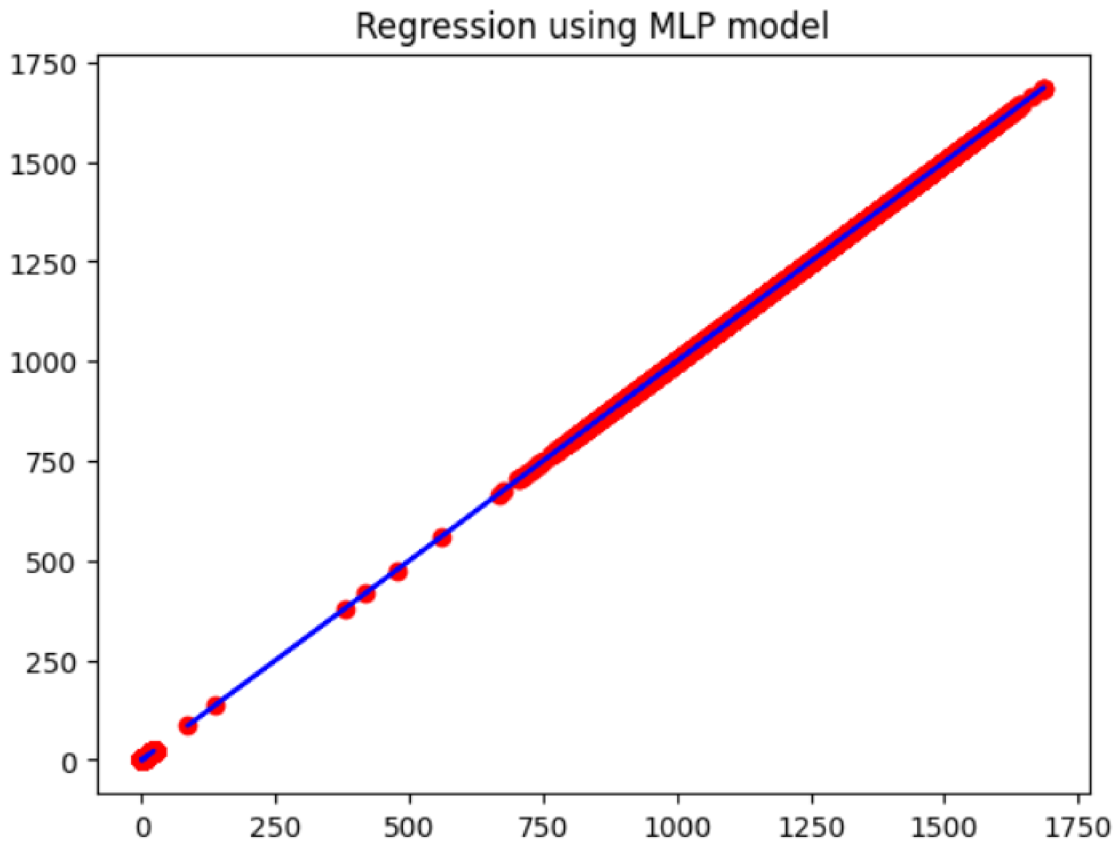
```
#buliding ML model using MLPRegressor            used data points (2570
3)

from sklearn.neural_network import MLPRegressor

X= df_train.iloc[:,1:]
y = df_train.iloc[:,1:]

X = X.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-
3",'holiday','Holiday_ID'],axis=1) #droping the features that have high
 VIF values (>5)
y =y.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-
3",'holiday','Holiday_ID'],axis=1)

mlp = MLPRegressor(hidden_layer_sizes=(7),
                    random_state=4,
                    verbose= True,
                    activation= 'relu',
                    learning_rate_init=0.01)
mlp.fit(X,y)
```



Testing the testing set with model developed using ""MLPRegressor""

```
test = df_test
X_new = test.iloc[:,1:]
X_new = X_new.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-
3",'holiday','Holiday_ID'],axis =1)
y_pred = mlp.predict(X_new)
y_pred
y = test.iloc[:,1:]
y = y.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-
3",'holiday','Holiday_ID'],axis =1)

from sklearn.metrics import r2_score
r2_score(y_true = y, y_pred = y_pred)
```

output:

```
0.9689968905543799 (r2_score)
```

Testing the model built using ""DessionTreeRegression"" on testdata set

```
test = df_test
X_new = test.iloc[:,1:]
X_new = X_new.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-3",'holiday','Holiday_ID'],axis =1)
y_pred = regressor.predict(X_new)
y_pred
y = test.iloc[:,1:]
y = y.drop(["MA_X-4", "T2M_toc", "week_X-4", "week_X-3",'holiday','Holiday_ID'],axis =1)

from sklearn.metrics import r2_score
r2_score(y_true = y, y_pred = y_pred)
```

output:

```
0.9419405215422714 (r2_score)
```