```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_absolute_error, mean_squared_error
```

```
data = pd.read_csv('/content/energydata
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19735 entries, 0 to 19734
Data columns (total 29 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   date         19735 non-null  object 
 1   Appliances   19735 non-null  int64  
 2   lights       19735 non-null  int64  
 3   T1           19735 non-null  float64
 4   RH_1         19735 non-null  float64
 5   T2           19735 non-null  float64
 6   RH_2         19735 non-null  float64
 7   T3           19735 non-null  float64
 8   RH_3         19735 non-null  float64
 9   T4           19735 non-null  float64
 10  RH_4         19735 non-null  float64
 11  T5           19735 non-null  float64
 12  RH_5         19735 non-null  float64
 13  T6           19735 non-null  float64
 14  RH_6         19735 non-null  float64
 15  T7           19735 non-null  float64
 16  RH_7         19735 non-null  float64
 17  T8           19735 non-null  float64
 18  RH_8         19735 non-null  float64
 19  T9           19735 non-null  float64
 20  RH_9         19735 non-null  float64
 21  T_out        19735 non-null  float64
 22  Press_mm_hg  19735 non-null  float64
 23  RH_out       19735 non-null  float64
 24  Windspeed    19735 non-null  float64
 25  Visibility   19735 non-null  float64
 26  Tdewpoint    19735 non-null  float64
 27  rv1          19735 non-null  float64
 28  rv2          19735 non-null  float64
dtypes: float64(26), int64(2), object(1)
memory usage: 4.4+ MB
```

```
Start coding or generate with AI.
```

Question 17. From the dataset, fit a linear model on the relationship between the temperature in the living room in Celsius (x = T2) and the temperature outside the building (y = T6). What is the Root Mean Squared error in three D.P?

```
x = data[['T2']]
y = data['T6']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,

model = LinearRegression()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f'Root Mean Squared error: {rmse:.3f}')
```

```
Root Mean Squared error: 3.630
```

Question 18. Remove the following columns: [“date”, “lights”]. The target variable is “Appliances”. Use a 70-30 train-test set split with a random state of 42 (for reproducibility). Normalize the dataset using the MinMaxScaler. Run a multiple linear regression using the training set. What is the Mean Absolute Error (in three decimal places) for the training set?

```
cleaned_data = data.drop(columns=["date", "lights"])

x = cleaned_data.drop(columns=["Appliances"])
y = cleaned_data["Appliances"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
scaler = MinMaxScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)

model = LinearRegression()
model.fit(x_train_scaled, y_train)
y_train_pred = model.predict(x_train_scaled)
mae_train = mean_absolute_error(y_train, y_train_pred)
print(f"Mean Absolute Error for the training set: {mae_train:.3f}")
```

```
Mean Absolute Error for the training set: 53.742
```

Question 19. What is the Root Mean Squared Error (in three decimal places) for the training set?

```
mse_train = mean_squared_error(y_train, y_train_pred)
rmse_train = np.sqrt(mse_train)
print(f"Root Mean Squared Error for the training set: {rmse_train:.3f}")
```

```
Root Mean Squared Error for the training set: 95.216
```

Question 20. What is the Mean Absolute Error (in three decimal places) for test set?

```
y_test_pred = model.predict(x_test)
mae_test = mean_absolute_error(y_test, y_test_pred)
print(f"Mean Absolute Error for the test set: {mae_test:.3f}")
```

```
mse_test = mean_squared_error(y_test, y_pred)
rmse_test = np.sqrt(mse_test)
print("Root Mean Squared Error for the test set:", round(rmse_test, 3))
```

```
data = data.drop(columns=["date", "lights"])
x = data.drop(columns=["Appliances"])
y = data["Appliances"]

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3,
scaler = MinMaxScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
model = LinearRegression()
model.fit(x_train_scaled, y_train)
y_train_pred = model.predict(x_train_scaled)
mae_train = mean_absolute_error(y_train, y_train_pred)

print("Mean Absolute Error for the training set:", round(mae_train, 3))
```

```
Mean Absolute Error for the training set: 53.742
```

```
y_test_pred = model.predict(x_test_scaled)
mse_test = mean_squared_error(y_test, y_test_pred)
rmse_test = mse_test ** 0.5

print("Root Mean Squared Error for the test set:", round(rmse_test, 3))
```

```
Root Mean Squared Error for the test set: 93.64
```

```
lasso_model = Lasso()
lasso_model.fit(x_train, y_train)
non_zero_features = sum(lasso_model.coef_ != 0)

print("Number of features with non-zero feature weights:", non_zero_featu
```

```
Number of features with non-zero feature weights: 21
```

Question 25. What is the new RMSE with the Lasso Regression on the test set?

```
y_pred_lasso = lasso_model.predict(x_test)
mse_test_lasso = mean_squared_error(y_test, y_pred_lasso)
rmse_test_lasso = np.sqrt(mse_test_lasso)
print("Root Mean Squared Error with Lasso Regression on test set:", round
```

```
Root Mean Squared Error with Lasso Regression on test set: 93.892
```

```
Start coding or generate with AI.
```