Names: Andrew Nguyen & Aaron Fredericks
NetIDs: axn210071 & ajf220004

CS4372 - Assignment Result Analysis (README)

Github Repository:

https://github.com/AndrewN20/CS4372.501Assignment1LinearRegressionAnalysis/tree/main

About the Data

Our dataset is a collection of different datapoints of the features that affect the electrical power output of a powerplant at full workload from 2006-2011. There are four features on this dataset, which all are continuous data types. Our target is also a continuous data type. For the size of the dataset, there are 9568 instances, which is a good amount of data despite the limited amount of features.
Link to dataset: https://archive.ics.uci.edu/dataset/294/combined+cycle+power+plant

How to Run

Our files would have to be run in Google Colab or Jupyter Notebook, as it is a .ipynb file. Our notebook is very simple to run, simply just copy the code from the Python notebook provided in our submission, and either press enter+shift to run a code block one at a time, or select the Runtime tab and select the Run All option in Google Colab or Jupyter Notebook.
Make sure you install all of the following libraries using pip (or Anaconda if you want) and follow the instructions on installing them on the respective website:

- `Ucimlrepo (https://archive.ics.uci.edu/dataset/294/combined+cycle+power+plant)`
- `Matplotlib (https://matplotlib.org/stable/install/index.html)`
- `Numpy (https://numpy.org/install/)`
- `Seaborn (https://seaborn.pydata.org/installing.html)`
- `Pandas (https://pandas.pydata.org/docs/getting_started/install.html)`
- `Sklearn (https://scikit-learn.org/stable/install.html)`
- `Statsmodels (https://www.statsmodels.org/stable/install.html)`

Preprocessing

Before you visualize the data, you will need to preprocess the data. This is imperative so that we can make sure the data is formatted correctly and won't have to deal with any incompatible data. When preprocessing, there were no null or missing values in our data, as stated in the dataset background. But just to make sure, we checked to see if there were any leftover null values, which there were none.

Plots and Correlation Matrix (Histogram/Heatmap/Pairplot/Scatterplot)

We started visualization with a histogram of all the features in the dataset. From the **Temperature** histogram, we can see that it is roughly bell-shaped but skewed slightly right and that most of the data is between 5°C and 30°C. This makes sense because higher temperatures reduce energy output. The **Exhaust Vacuum** histogram shows a bimodal distribution with two peaks around 40 and 65, which suggests there may be two operating regimes in the power plant. The **Ambient Pressure** histogram shows a normal distribution centered around 1010–1015. The **Relative Humidity** histogram is skewed left, but most values lie between 60% and 90%.

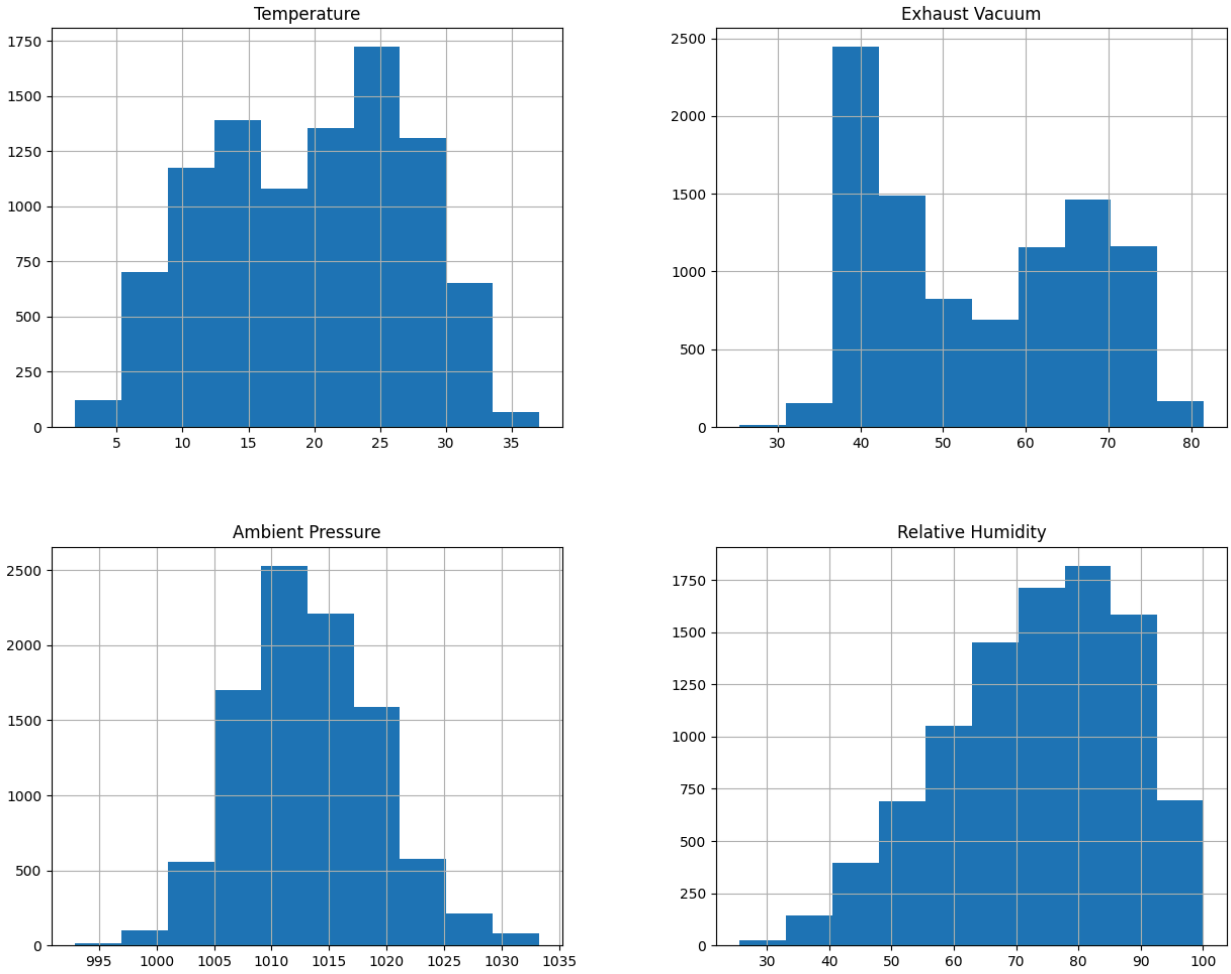


Fig. 1 A histogram of all the features

Next we did a heatmap to see if there was any correlation between the features. Temperature and Exhaust Vacuum have a strong positive correlation, which is accurate because higher ambient temperatures reduce the efficiency of the condenser in the power plant where the steam is cooled, which increases the exhaust vacuum pressure. Temperature and Ambient

Pressure have a negative correlation, this follows the temperature/pressure relationship that states that temperature is proportional to pressure. Temperature and Relative Humidity have a negative correlation because at high temperatures, air can hold much more water vapor, but relative humidity measures moisture relative to the maximum the air can hold, so as temperature increases, relative humidity decreases. Exhaust Vacuum and Ambient Pressure have a negative correlation because lower ambient pressure reduces the density of air used in the condenser, which worsens cooling, which leads to high vacuum pressure. Exhaust Vacuum and Relative Humidity have a weak negative correlation, which tells us that the high humidity corresponds to slightly lower exhaust vacuum, suggesting more efficient cooling.

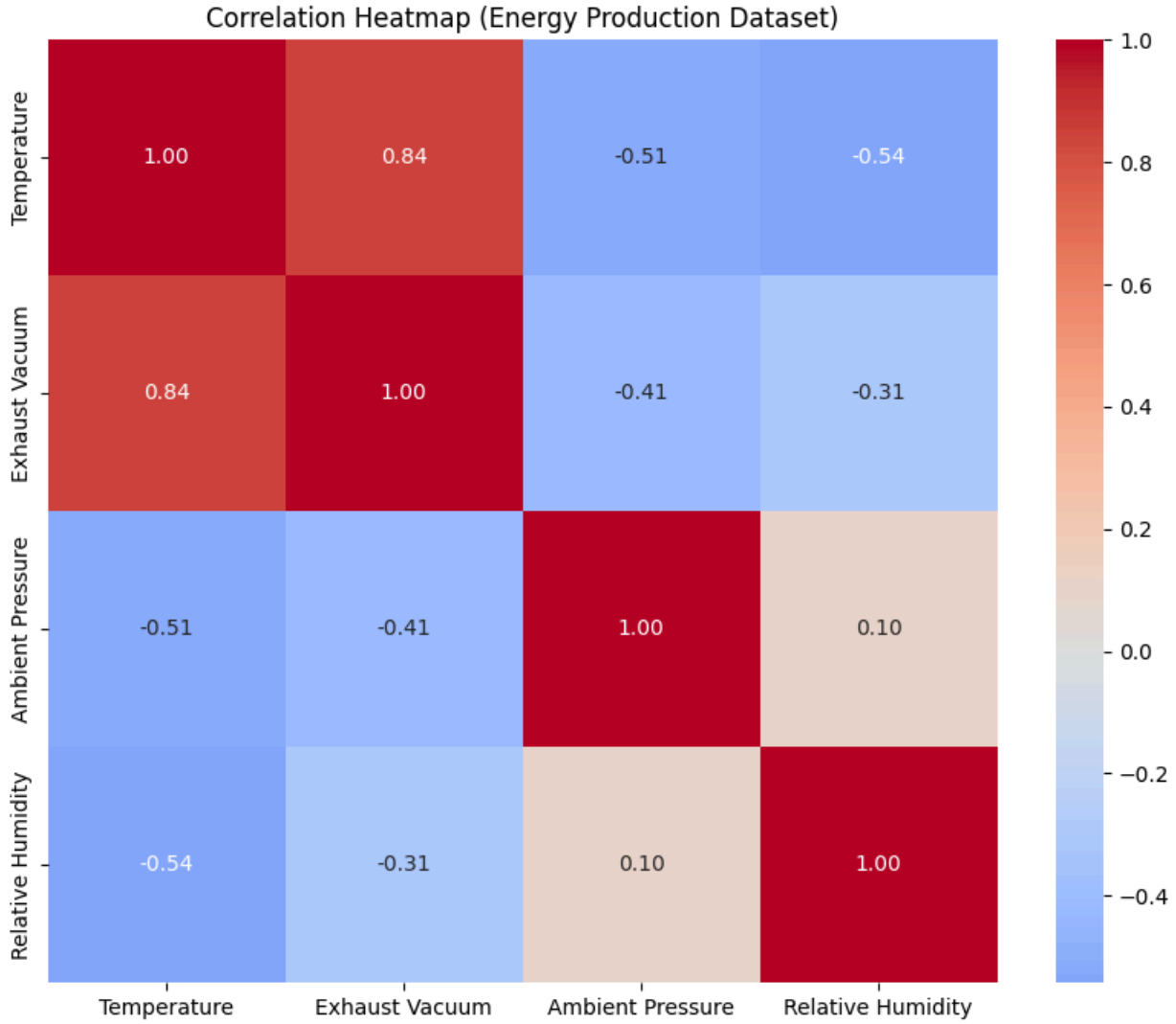


Fig. 2 A heatmap of all the features to find correlations between features

After spotting the correlations between all the features, we did a pairplot of them. **Temperature vs Exhaust Vacuum** shows a very clear positive linear relationship, as temperature rises, exhaust vacuum also rises. **Temperature vs Ambient Pressure** shows a negative trend, but the scatter is more spread out; higher temperatures are associated with lower ambient pressure.

Temperature vs Relative Humidity has a strong negative slope; higher temperatures mean lower relative humidity. **Exhaust Vacuum vs Ambient Pressure** has a negative relationship; higher vacuum values are seen when the pressure is lower. **Exhaust Vacuum vs Relative Humidity** has a very weak negative trend(there is no strong linear trend), as humidity decreases, vacuum increases slightly. **Ambient Pressure vs Relative Humidity** has no clear relationship; this makes sense, as these two are nearly independent atmospheric measures.

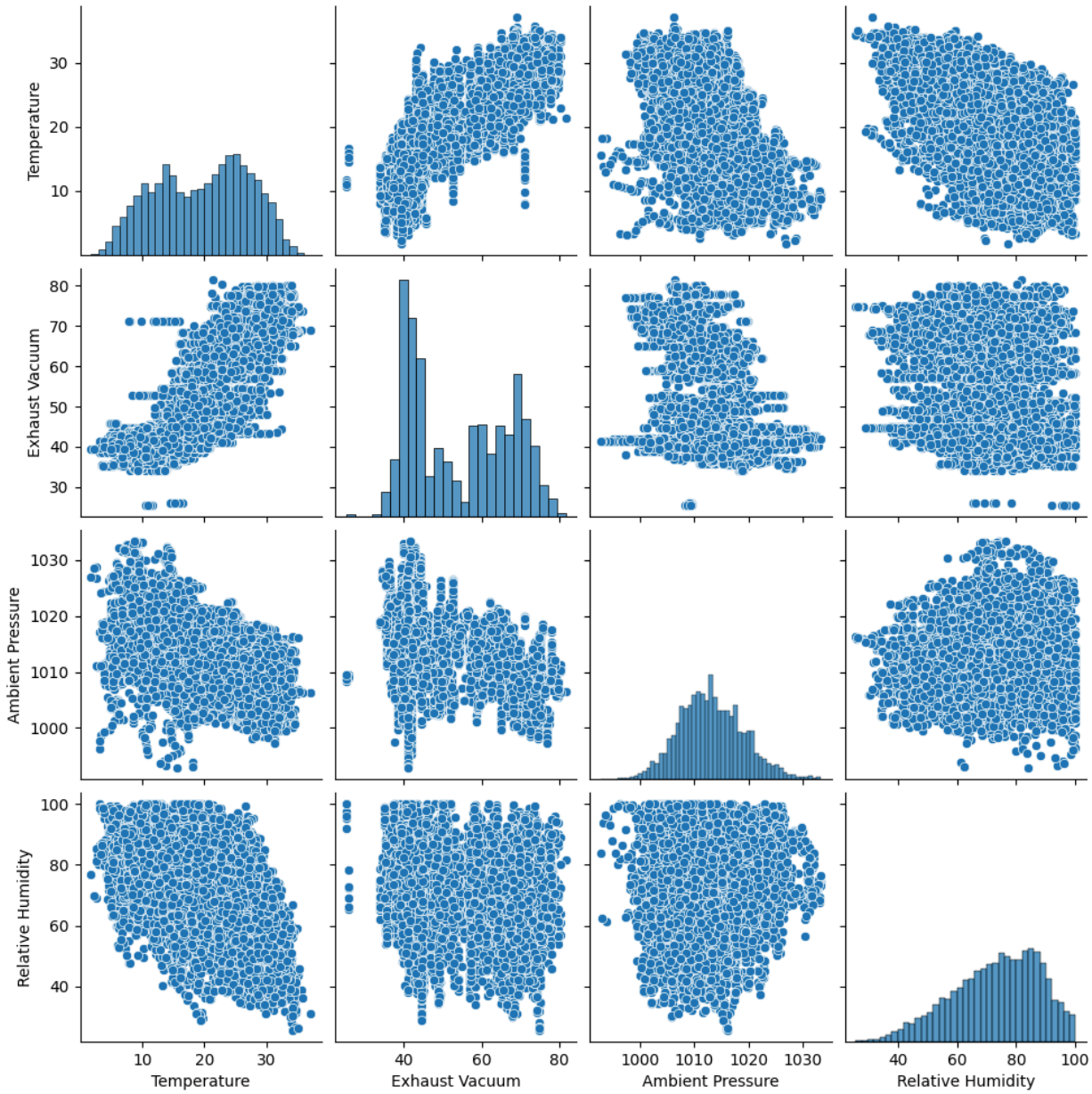
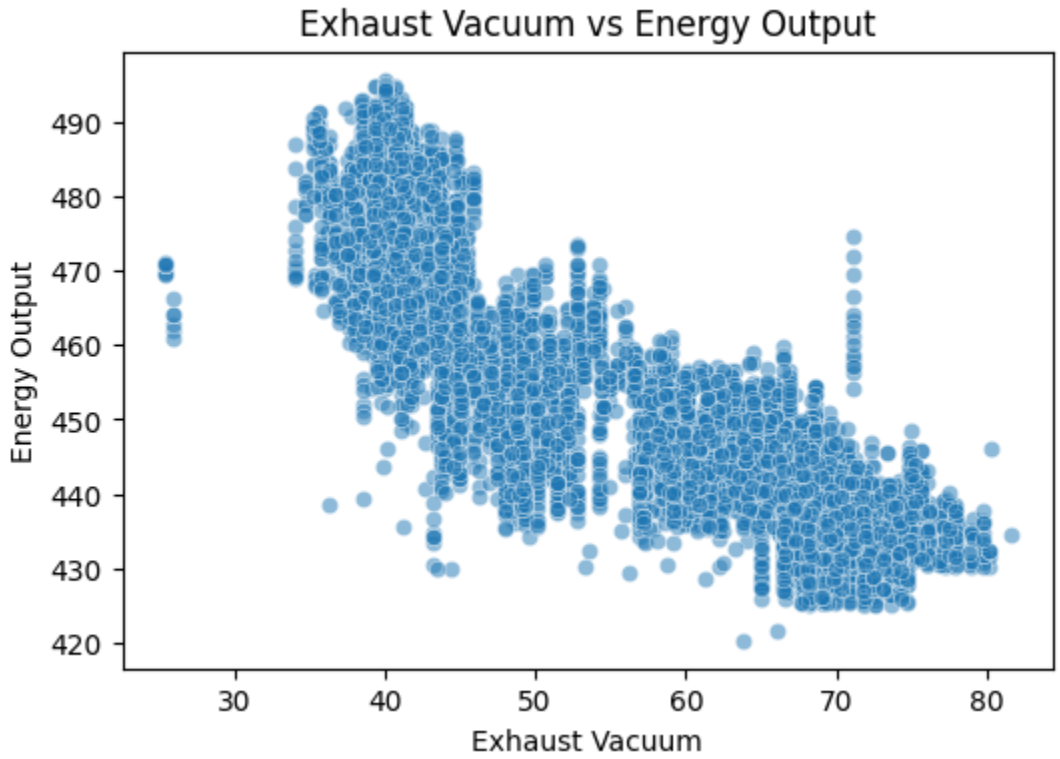
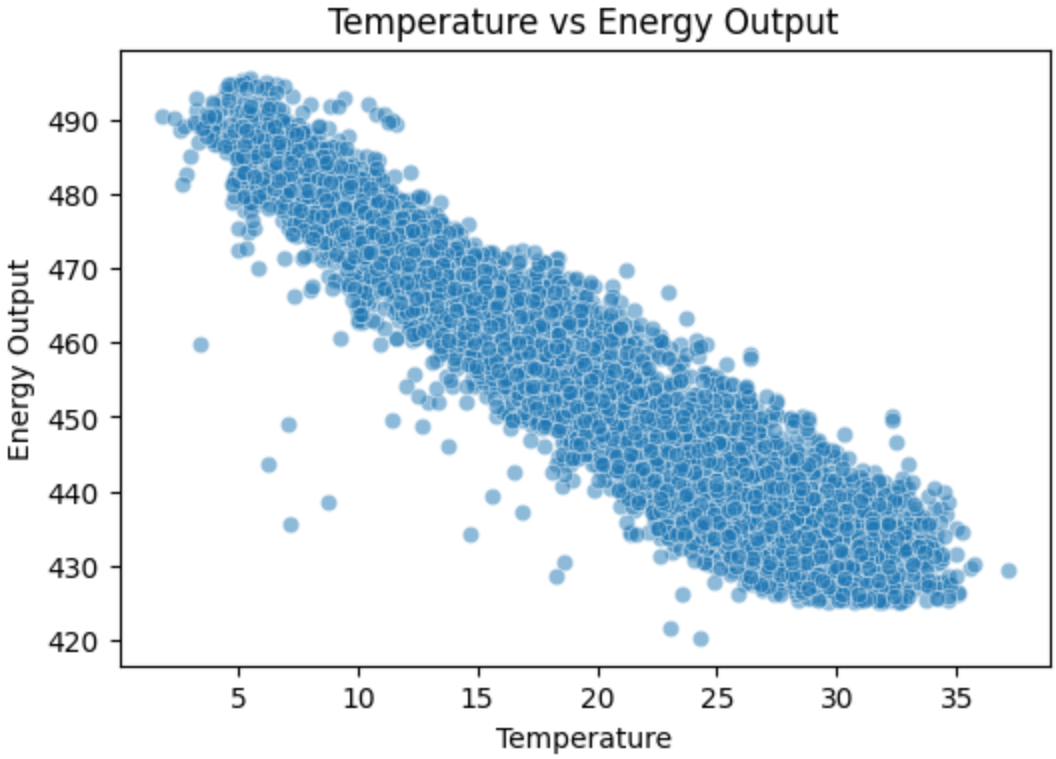


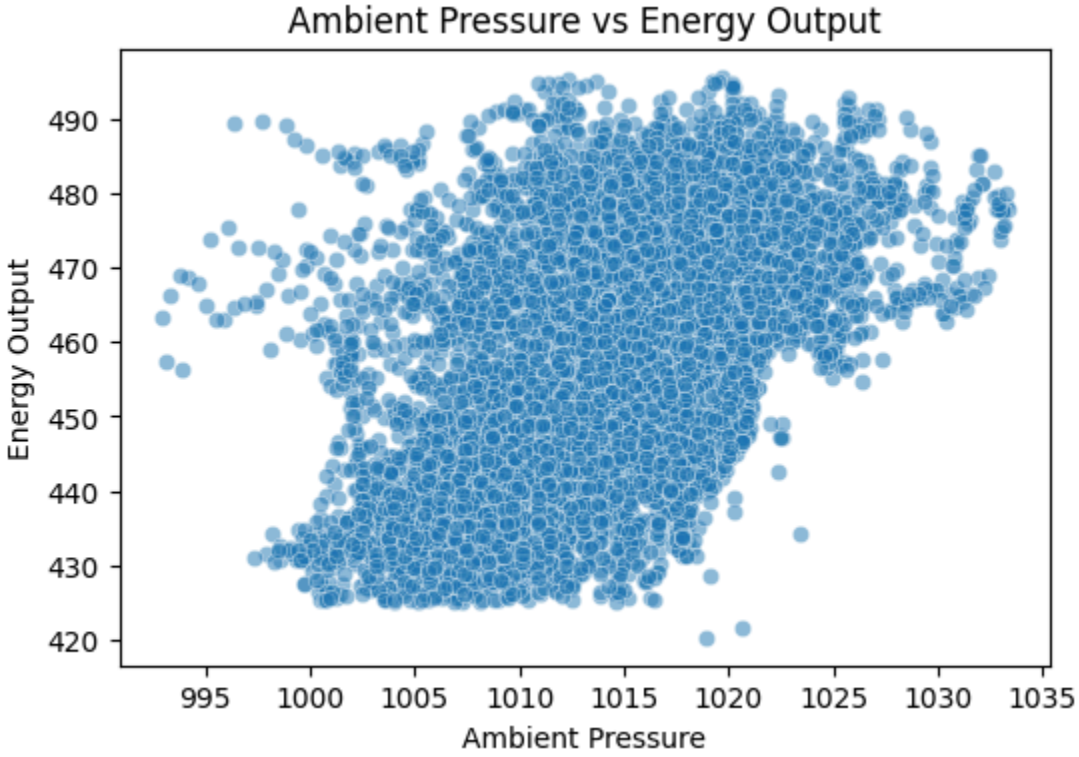
Fig. 3 A Pairplot of all the features

We also did individual scatter plots between each of the features and the target. Looking at the scatterplot shape, we noticed that Exhaust Vacuum and Temperature had more concentrated

negative correlation to temperature. Ambient Pressure and Relative Humidity's scatter plots did not concentrate properly, has more outliers than the other features, and looks like there is little to no correlation between these features and the target. We can also confirm that via the boxplot, as those were the only two that had outliers outside the quartiles. Based on this analysis, we decided to drop Ambient Pressure and Relative Humidity as our features.




Temperature vs Energy Output
Energy Output
490
480
470
460
450
440
430
420
5
10
15
20
25
30
35
Temperature


Ambient Pressure vs Energy Output
Energy Output
490
480
470
460
450
440
430
420
995
1000
1005
1010
1015
1020
1025
1030
1035
Ambient Pressure

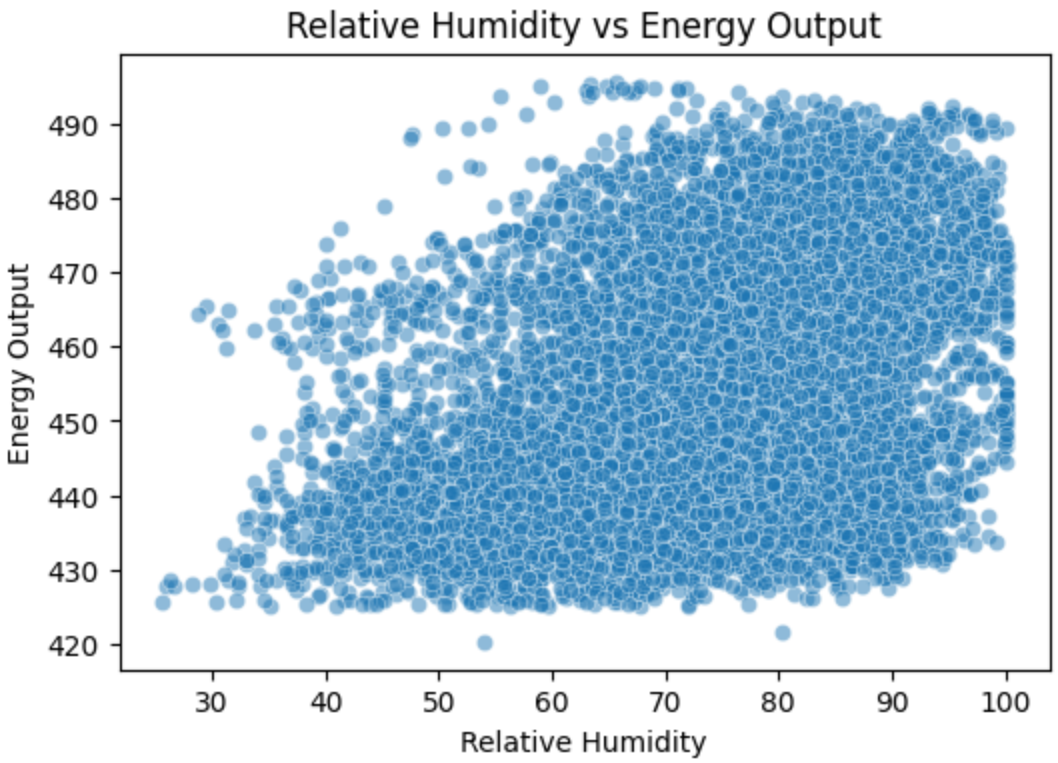


Fig. 4 Several Scatterplots that depict each feature against the target

Splitting the Data

When splitting the data, we separated the feature data and target values into training and testing data. In splitting the data to testing and training data, we set test size to 0.2 so that 80% is training data and 20% is testing data. We also tinkered with the random state hyperparameter by setting it to 5 so that we can shuffle it a bit and prevent overfitting.

SGDRegression Model

In SGDRegression, we tuned different hyperparameters and the values of each parameter. We tuned one parameter at a time so that we can clearly see what is being affected. We also started out with setting up some basics hyperparameters

Test 1:

Parameters: {alpha = 1, eta0 = 0.01, max_iter = 10000000, tol = 1, learning_rate = "invscaling"}

Results:

- `Coefficients: [-6.18075266, -4.98156895]`
- `Intercept: [454.52153408]`
- `MSE: 57.30578714487981`
- `MAE: 6.190732336943089`
- `EV: 0.8014517087397812`
- `R2: 0.8010611187617149`

Test 2:
Parameters: {alpha = 1, eta0 = 0.01, max_iter = 10000000, tol = 1, learning_rate = "optimal"}
Results:

- *Coefficients: [-6.0462007 , -4.86384923]*
- *Intercept: [454.42905877]*
- *MSE: 59.96761265114434*
- *MAE: 6.344727753825876*
- *EV: 0.7920290513382657*
- *R2: 0.7918205059955205*

Test 3:
Parameters: {alphaVar = 1, eta0 = 0.01, max_iter = 10000000, tol = 1, learning_rate = "adaptive"}
Results:

- *Coefficients: [-6.04679398, -4.86329431]*
- *Intercept: [454.4139437]*
- *MSE: 59.95882899216666*
- *MAE: 6.343385026625134*
- *EV: 0.7920346148920505*
- *R2: 0.7918509987498696*

Test 4:
Parameters: {alphaVar = 1, eta0 = 0.01, max_iter = 10000000, tol = 1, learning_rate = "constant"}
Results:

- *Coefficients: [-5.92743952, -4.94541041]*
- *Intercept: [453.9135106]*
- *MSE: 60.59750417653036*
- *MAE: 6.345476479342803*
- *EV: 0.7898871393004743*
- *R2: 0.7896338173274988*

Test 5:
Parameters: {alphaVar = 2, eta0 = 1, max_iter = 10000000, tol = 1, learning_rate = "invscaling"}
Results:

- *Coefficients: [-6.21248944, -4.92826114]*
- *Intercept: [454.47170134]*
- *MSE: 57.41394137540735*
- *MAE: 6.197146433310928*

- *EV: 0.800969175132698*
- *R2: 0.8006856578755754*

Test 6:
Parameters: {alphaVar = 1, eta0 = 1, max_iter = 10000000, tol = 1e-05, learning_rate = "invscaling"}
Results:

- *Coefficients: [-6.5788024 , -5.72591726]*
- *Intercept: [455.30873437]*
- *MSE: 48.0325738127707*
- *MAE: 5.629084863105986*
- *EV: 0.8375531918534876*
- *R2: 0.8332533767811314*

Test 7:
Parameters: {alpha= 1, eta0= 1, max_iter= 10000000, tol= 2, learning_rate = "invscaling"}
Results:

- *Coefficients: [-4.94312802, -4.33872421]*
- *Intercept: [454.86810044]*
- *MSE: 81.18657207334977*
- *MAE: 7.512956889653914*
- *EV: 0.7198494883491018*
- *R2: 0.7181582066221267*

Test 8:
Parameters: {alpha= 1, eta0= 1, max_iter= 10000000, tol = 1e-05, learning_rate= 'invscaling', penalty= 'l1'}
Results:

- *Coefficients: [-12.20161612, -2.53335575]*
- *Intercept: [454.00796517]*
- *MSE: 27.918441618619344*
- *MAE: 4.267357302785256*
- *EV: 0.903231294780945*
- *R2: 0.9030802329355883*

Test 9:
Parameters: {alpha= 1, eta0= 1, max_iter= 10000000, tol = 1e-05, learning_rate= 'invscaling', penalty= 'elasticnet'}
Results:

- *Coefficients: [-5.95651207, -5.19170283]*
- *Intercept: [454.57326806]*
- *MSE: 57.95335459255975*
- *MAE: 6.219255401164644*
- *EV: 0.7993337668749222*
- *R2: 0.7988130675615441*

Test 10:
Parameters: {alpha= 1, eta0= 1, max_iter= 100000000, tol = 1e-05, learning_rate= 'invscaling', penalty= 'l1'}
Results:

- *Coefficients: [-13.03027565, -3.19957889]*
- *Intercept: [453.64241343]*
- *MSE: 24.190499344646994*
- *MAE: 3.922665822000313*
- *EV: 0.9172175271593134*
- *R2: 0.916021904313908*

OLS Model

We also tested using the OLS model provided by StatsModel. Here are the parameters we used:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.915
Model:                            OLS   Adj. R-squared:                  0.915
Method:                 Least Squares   F-statistic:                 4.107e+04
Date:                Sun, 21 Sep 2025   Prob (F-statistic):               0.00
Time:                        16:21:19   Log-Likelihood:                -23161.
No. Observations:                7654   AIC:                         4.633e+04
Df Residuals:                    7651   BIC:                         4.635e+04
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        454.3978      0.057   7968.260      0.000     454.286     454.510
0            -12.7217      0.106   -119.923      0.000     -12.930     -12.514
1             -4.0637      0.106    -38.211      0.000      -4.272      -3.855
==============================================================================
Omnibus:                      525.346   Durbin-Watson:                   1.940
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             2003.219
Skew:                          -0.255   Prob(JB):                         0.00
Kurtosis:                       5.454   Cond. No.                         3.44
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

- *MSE: 23.212254682307425*
- *MAE: 3.8402925569931483*
- *EV: 0.919511040350015*
- *R2: 0.919417912089024*
- *RSME: 4.81790978353761*

Conclusions

Based on both the two regressors models we made, we can see that our predicted values match our test values for the target closely. For the SGDRegressor, we see how by adding the “l1” penalty to our dataset decreases our MSE and MAE, which is a good thing as having a MSE and MAE close to zero means that it is a near or exact match. Also adding that penalty increases our EV and R2 close to 1, as it means our coefficients and intercept is a close match to our actual target. In the OLS model, it also shows similar results for our MSE, MAE, EV, R2. Comparing the coefficients and intercept of the highest R2 SGDRegression model and the coefficients and intercept of the OLS model, they are very similar in value. In the OLS model, the standard error is small, the t-value is high, and the P-value is surprisingly zero, meaning these coefficients have little error and are significant. R2 and R2-adjusted are the same and are similar to the R2 in the SGDRegression model. And finally the F-statistic indicated that this model very much fits the dataset and explains the dataset very well. Overall, I can definitely say that linear regression surprisingly matched our dataset, which is a rare thing to experience as real-life data is not linear/simple.