Joseph Loss

XR Trading PC2 Writeup

## **Predicting Chicago Air Quality Through Linear Regression**

Please see the *predictions.csv* file for the predicted values of each record in the “TESTING\_DATA.txt” file.

The model itself can be run from the *main.py* file. Please be sure to include the two accompanying files, *create\_new\_features.py* and *readdata.py* in the same folder because these contain classes and functions that are used by the main file.

**Feature Engineering and Variable Transformations:**

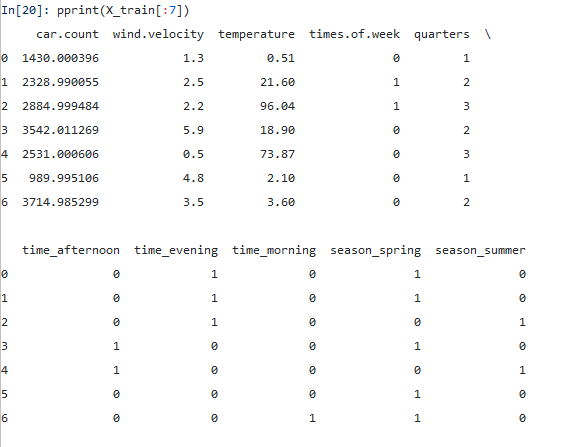
During my EDA, I plotted regressions of each variable in relation to one another. From the correlation heatmap of covariates (Figure 1), I saw that Temperature.1 and Temperature.2 were extremely highly correlated at 0.99. When this happens, we know that keeping one covariate vs. the other does not add to the predictive power of the linear regression model. As a result, I used the product of the two highly correlated variables as an independent variable (removing the individual temperatures).

I was able to extract 9 new features out of the day.index and time.of.day variables. The exact details of my methodology are explained in the comments of my source code. The short explanation is that my code determined (starting on 01/31/2010) the date for each observation in the dataset. From there, I created a binary feature called “times.of.week” that tracked whether the observation fell on a weekday or on a weekend.

Using a similar methodology, I was also able to track the quarter of the year (starting on 01/01/2010 and adjusting 31 days to the start of the “day.index” variable), as well as the season of the year, which definitely has an impact on the levels of s02 in the air.   
  
Note: the seasons were calculated using the date-ranges for Northern Hemisphere seasons. This is not only to improve the accuracy of my model’s predictions but also due to the geographical location of Chicago.

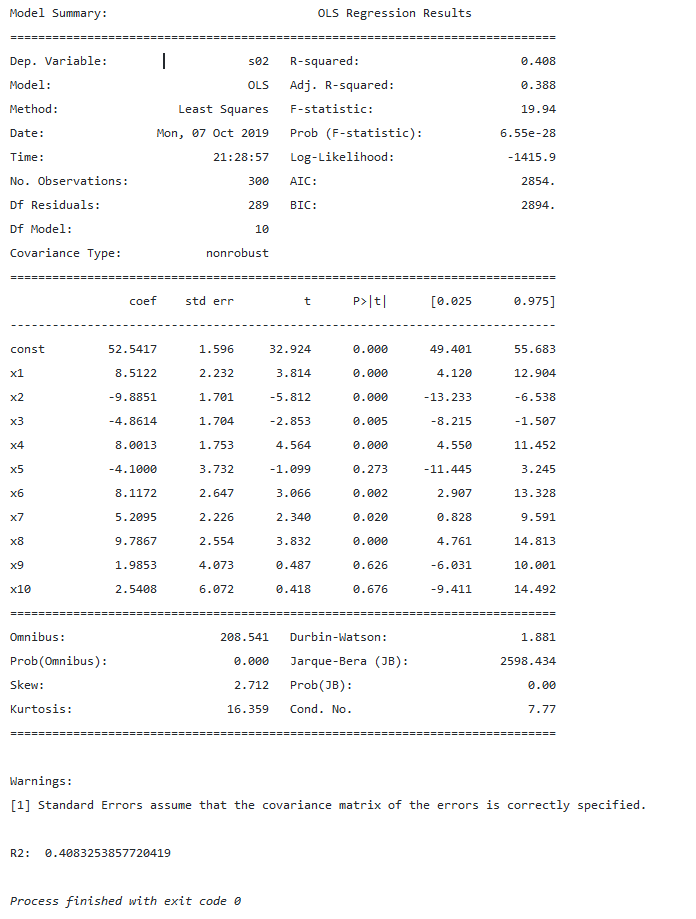
I used a similar extrapolation to compute the time of day for each observation from the time.of.day variable. The head of the dfTrain dataframe is displayed in the figure on the next page.

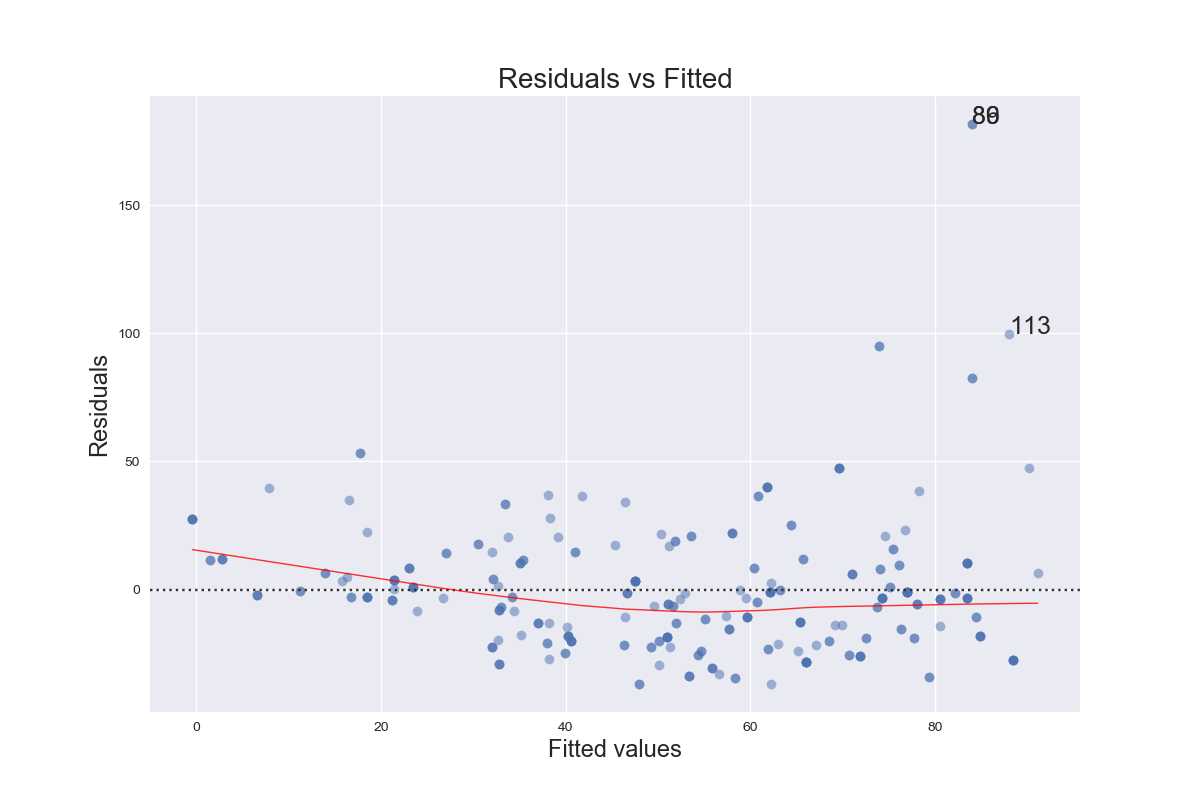
Note that the “dummy variable” columns *time\_night* and *season\_winter* are removed from the model; this is standard practice when using pd.get\_dummies.

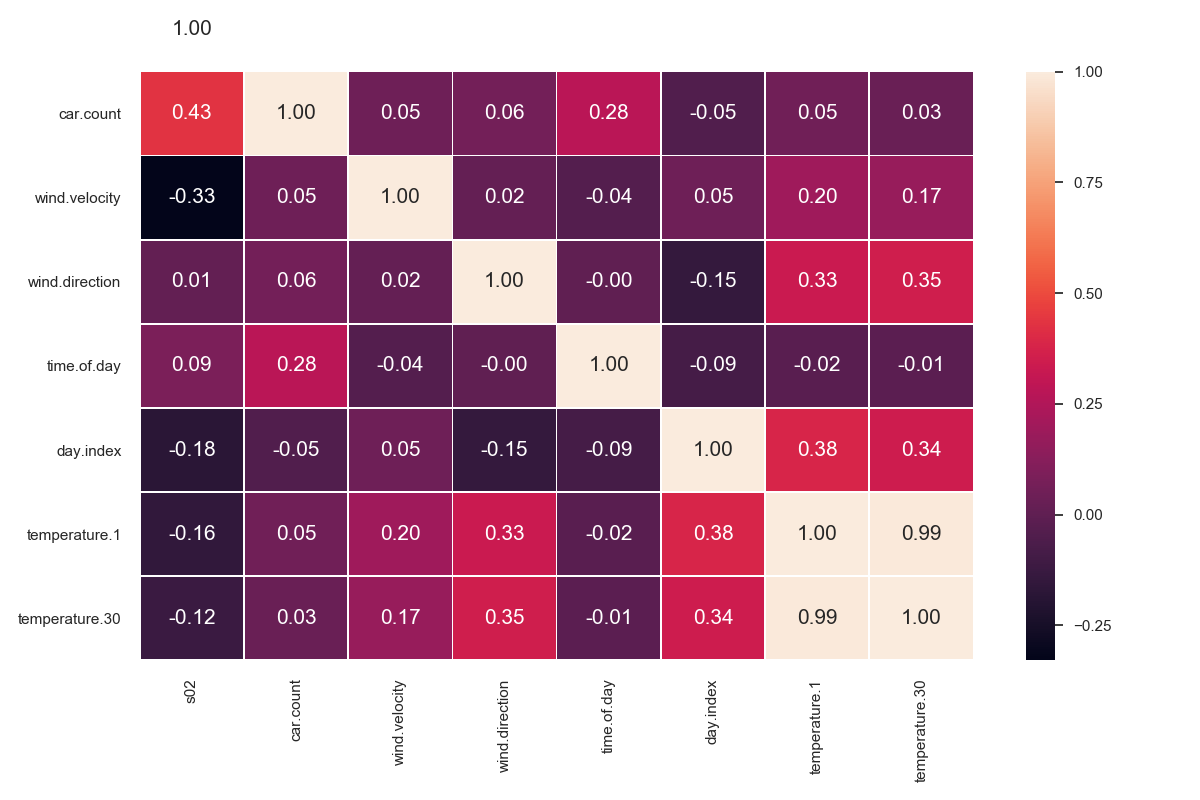
Also note: the variables are not standardized in the image. The model uses X\_train\_std and X\_test\_std when fitting and predicting the datasets.

The model and summary statistics are on the following page, along with a plot of the model residuals vs fitted values.

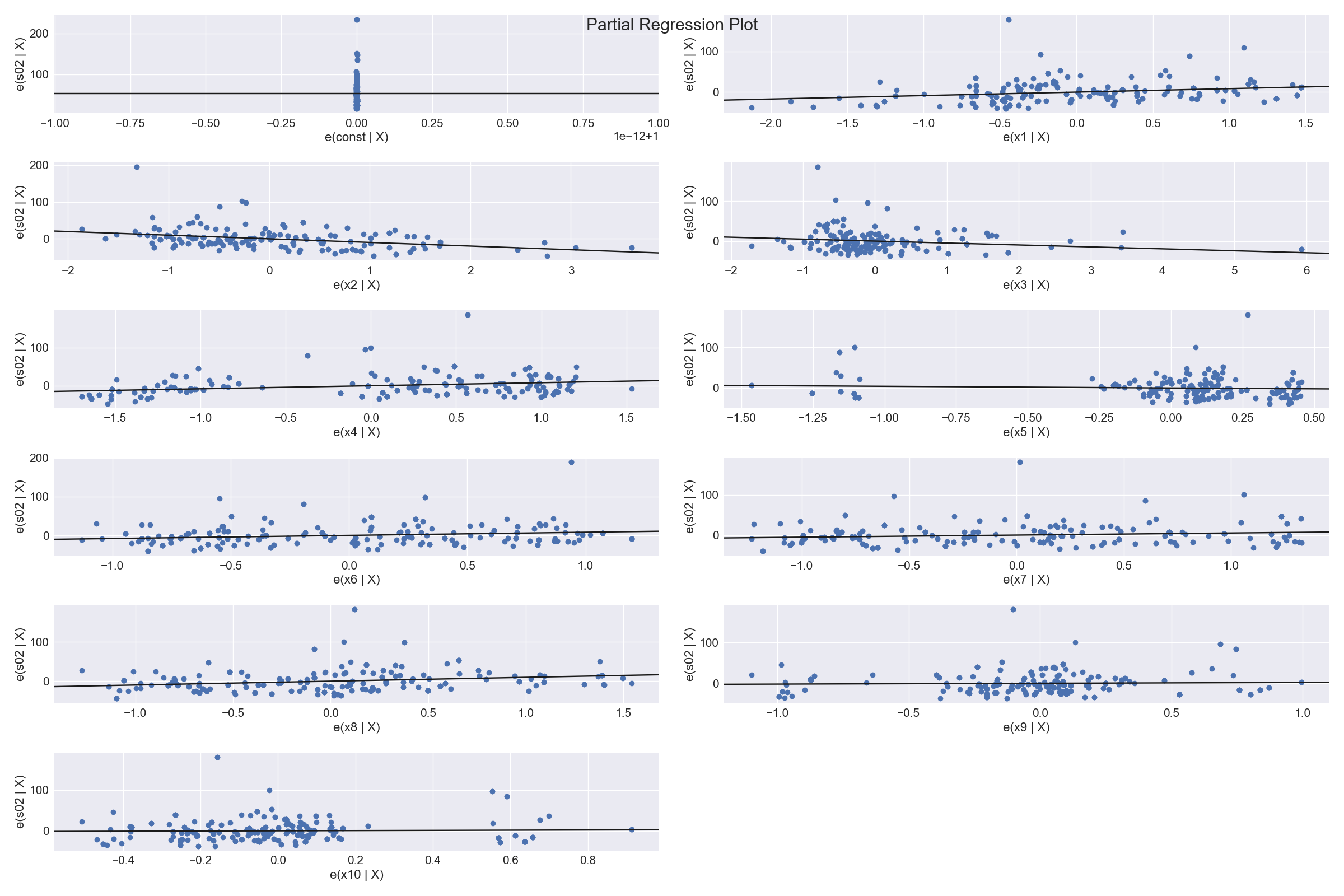
Overall, this project took about a day to complete, but it probably would have taken less time had I known scikit-learn doesn’t output the model statistics or summary information (I first built the entire model using this package and had to re-write it using the statsmodels python library).







Figure



Figure