Approach To Problem:

Each section corresponds to a file. The findings in one section led to the approach taken in the next section which is reflected in the code.

The file utils.py contains helper functions I wrote which I have used throughout my notebooks.

1. Data Exploration -
    a. First, I determined the correlation between different predictors. The reason for this is to see if there exists data that can be removed from the dataset to provide a more relevant set of data for the model to learn from. This can be done by plotting the correlation matrix for the dataset and inspecting the column or row of critical_temp (our target variable). I chose the threshold above which a predictors correlation would be considered as significant to be 0.4. Upon doing this, it can be observed that only 38 predictors out of the 81 chemical properties have a strong correlation with the critical_temp, therefore, I later performed dimensionality reduction on the data. This was done through PCA and enabled me to reduce noise present in the dataset.

    b. Next, I used pandas' describe function to inspect the ranges of the data for each predictor and target. Since the ranges varied significantly across different predictors, I determined that normalization of the data was required.

    c. Finally, I inspected the predictors that are significantly correlated with critical_temp to by looking at their outliers through a series of boxplots which can be seen in the figure below. This was done to determine if the predictors contain outliers potentially skewing the results of our model. I observed that there where a significant number of predictors with outliers. Therefore, when choosing my model, I aimed to choose one that was robust towards outliers.

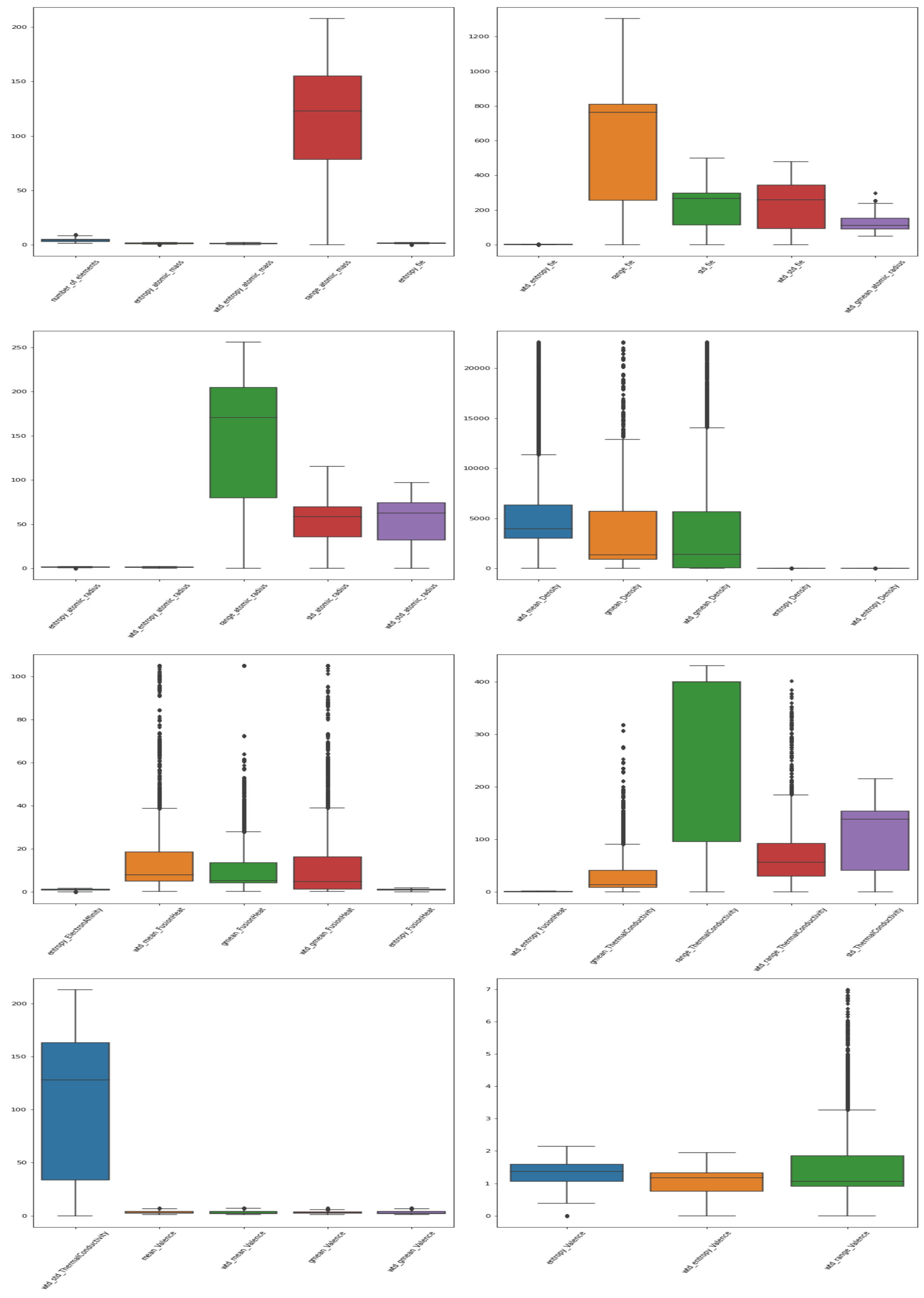

Figure 1: Boxplots of predictors with a correlation of greater than 0.4 with target

2. Model Selection –
    a. We will look at the performance of the following models in order to determine which model to use –

        i. Linear Regression – This is a simple model. It is difficult to model a nonlinear or complex data through this model. It is greatly affected by outliers.

        ii. Support Vector Regression – This model is robust when dealing with outliers in the data, however, is significantly affected by noise.

        iii. Lasso Regression – This model improves performance and reduces the effect of overfitting by performing regularization, however, selected features by the model can be affected by bias and the selected features may not comprise of all relevant predictors.

        iv. Random Forest Regression – This model is good at learning nonlinear relationships in the data and can handle higher dimensional data well. However, this model is prone to overfitting and can be adversely affected by noise.

    b. First, I split the data into training and validation sets. This is to evaluate the performance of the model trained on data it has not seen before.
    c. After this, I normalized the data using the StandardScalar class from sklearn. Next, I performed dimensionality reduction through principal component analysis (PCA) using a class found in sklearn. Both were done to decrease the chance of overfitting, reduce noise in the dataset and train our model on predictors that have a significant correlation with the target.

d. Following this, I fit each model mentioned above to the training data, evaluated the models on the validation data and recorded their root mean square error value for the purposes of determining the best model. The results are as follows:
   i. Linear Regressor RMSE = 22.2516
   Linear regression performed the worst. This could occur because the data we are trying to model is nonlinear and the model could be overfitting.

   ii. SVR RMSE = 17.4962
   Support vector regression performed a lot better than linear and lasso regression since it is resilient to outliers, however, it is possible that there is still noise affecting the model.

   iii. Lasso Regressor RMSE = 22.247
   Lasso regression performed slightly better than linear regression. This could be since the model is more resilient towards overfitting, however, is susceptible to leaving out key features as only 1 feature is selected from a group of correlated features.

   iv. Random Forest Regressor RMSE = 10.3225
   Random forest regression seemed to perform the best as it is good at modelling nonlinear relationships.

3. Model Tuning –
   a. In the previous section, I determined what model to use – Random Forest Regression. In this section, I focused on determining the best values for two hyperparameters for the model – n_estimators (no. of decision trees to be made in the model) and max_depth (the maximum depth of each tree in the forest).

b. As done before, I split the data into training and validation sets. Following this I normalized the data and performed dimensionality reduction on it.
c. After this, for different values of the hyperparameters mentioned above, I recorded the r^2 score. I then chose the values for n_estimators and max_depth by picking the lowest values of the hyperparameters for which the performance was closest to the maximum performance. This was done to choose the best performance without unnecessarily increasing the size of the model and thereby decreasing the speed and increasing the resource usage. The following are the results –
    i. No. of estimators = 80
    ii. Max depth = 20

4. Model Testing –
    a. As done before, I split my data into training and validation sets. I then normalized the data and performed dimensionality reduction on it. Following this, I created my model and fit it to my data using the hyperparameters mentioned above. The results were as follows –

       Random Forest Regressor RMSE = 10.3301

    b. I then normalized my test data and performed dimensionality reduction on it to maintain uniformity with my training and validation data. Following this, I predicted the critical_temp for the data present in test.csv. The results of this can be seen in the predicted_critical_temp column of results.csv.