IML 2023: Predicting Saturation Vapour Pressure From Molecular Properties

Helmi Karesti, Janne Penttala, Maija Säisä

2023-12-10

Introduction

The objective of our project was to build a machine learning model to predict saturation vapour pressure based on some interpretable features. We were given data sets to help us build our model. We started by getting familiar with the data through data exploration and preprocessing. After we had verified that our data was clean, we used it to select our model. Finally, after model selection we applied a few different methods to try and optimize the performance and accuracy of our model. The prediction accuracy was scored based on R2 value. This report contains a more in-depth explanation of our methods regarding the steps of building our model.

Data analysis

Preprocessing

As a first step to preprocessing we made sure that our data is valid. In other words, we wanted to ensure there are no missing nor mismatched values for any of the variables. This was quite straightforward as Kaggle provided us variable-specific graphs for each variable, meaning we could verify the validity of both the training and test data with the information from Kaggle. We noted that all the data points in both data sets were valid, and thus, the variables did not have missing values.

After validating the data, there were a few minor changes we decided to make to further prepare our data. First of all we decided to remove the 'Id' column from both data sets, as this would not be needed in predicting our target variable. Another thing we did was to apply one-hot encoding for a variable called 'parentspecies', as this was the only categorical variable. This encoding transformed the variable 'parentspecies' into eight binary columns (for example, category_apin, category_apin_decane etc.) for the train set and seven binary columns for the test set. Therefore, after applying one-hot encoding, we needed to create an extra column for the test set to match our training set.

Lastly we wanted to ensure the data consistency of both training and test data. To do this, we simply compared the two data sets to each other and made sure that both of them have the same columns in the same order. The number of columns in the preprocessed data sets were 32.

Data exploration

The data used in the project is based on GeckoQ dataset. Our first step with the project was to get familiar with this dataset and get a preliminary understanding of the features it has. Our plan was to first use material that was already provided by the project descriptions and then continue with some more data exploration by handling the data ourselves. We first got familiar with the features and their descriptions. Understanding the features perfectly is not necessarily important in this context, however it is useful to have a general understanding of how they might affect our target variable.

We noticed that the range of values of the variables varied notably. Because we planned to test several models, some of which would perform better with scaled variables, we decided to apply the StandardScaler() on the data sets.

To get a better understanding of our data, we created three plots for each of the explainable variables. For each explainable variable, we plotted i) the explainable variable against our target variable as a scatter plot, ii) the count of observations per explainable variable value as a histogram, and iii) the distribution of explainable variable values as a boxplot. We did this with and without scaling, and concluded that the shape of the graphs, naturally, remains the same, and the only difference is the value of the explainable variables, that is, the x axis. The three graphs for one of the explainable variables are shown in Figure 1.

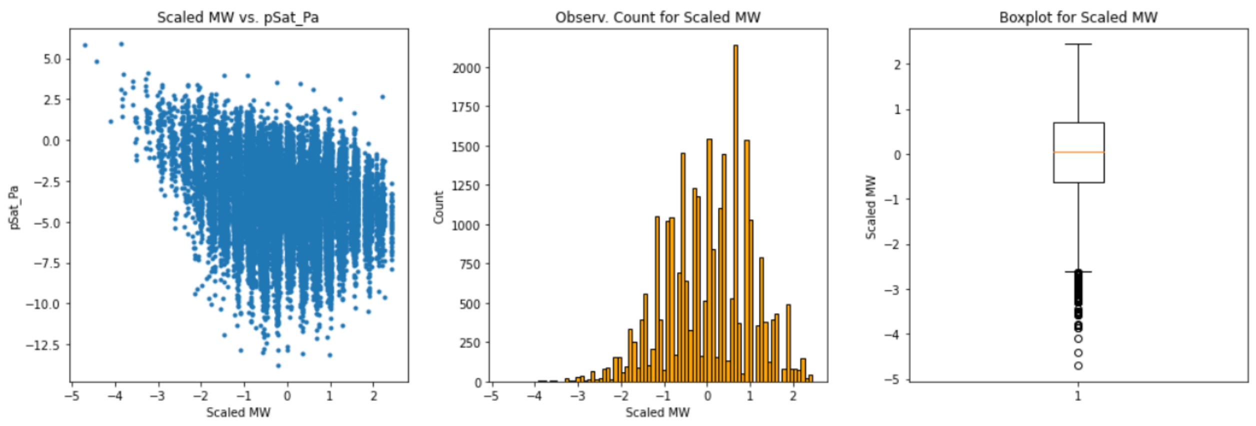

Figure 1: The three graphs for the (scaled) explainable variable 'MW'. Equal plots created for each of the explainable variables.

Analyzing the first graph, that is, the scatter plot, helped us to understand the relationship of each variable and the target variable better. The scatter plots demonstrated well that there was only one continuous variable while the rest of the variables were discrete (including the one-hot encoded categorical variable). That is, the 'MW' or molecule's molecular weight is the only continuous variable. ANY CONCLUSIONS WE DID BECAUSE OF THAT? NO?

Also, many of the variables had in maximum three different values, so the values were quite concentrated on specific points. Later, this information was used in feature selection, testing the effect of dropping the features with only a few different values. However, feature selection based on these observations did not improve the results.

Furthermore, we could observe a somewhat linear relationship for some of the variables, for example, 'MW', 'NumHBondDonors', 'NumOfAtoms', and 'NumOfO' (see Figure 2). Therefore, also including the linear regression model and testing including only these type of features was supported. However, linear regression model or feature selection based on these observations did not improve the results.

Analyzing the second graph, that is, the histogram, helped us TO BE CONTINUED.

In addition, we compared the distributions of variable values for both train and test data. DOES THIS REFER TO BOXPLOT OR SOMETHING ELSE? This would turn out useful in selecting the features for our model.

Methods

Model selection

For selecting our model, we first compared different models and their results in predicting the target variable. We evaluated the performance of the models based on their R2 score. The models we considered were Dummy

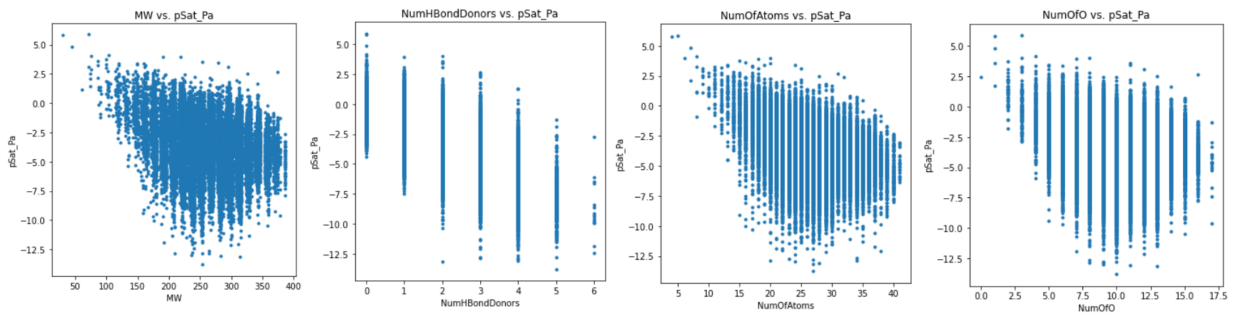

Figure 2: Somewhat linear relationship observed in some of the scatter plots.

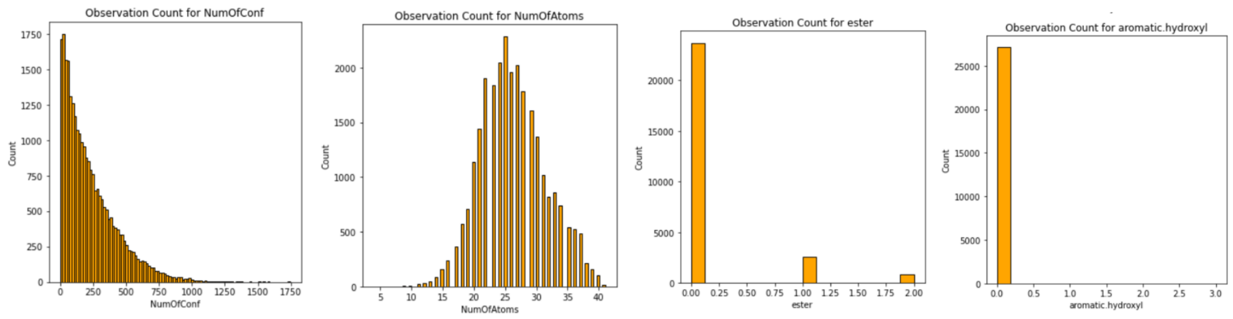

Figure 3: Various distributions for the count of observation values.

Model, OLS Linear Regression, Random Forest, Support Vector Regressor, Lasso Regression and eXtreme Gradient Boosting. After testing these six different models with and without scaling, the best results were obtained by Support Vector Regression model with scaling.

We also applied cross-validation in order to further verify our choice of model. We did this by using k-fold cross-validation (with k set to 5). The performance was once again evaluated by the models R2 score. As a result, Support Vector Regression model had once again the best performance.

Optimizing the model

To optimize our model to perform better, we tried i.a. removing outliers, tuning hyperparameters and selecting less features. Even if we obtained the best initial results with SVR, we decided to try and optimize the performance with other models as well. These models included Random Forest and XGB. We used randomized search to explore on different combinations of hyperparameters in order to improve the performance. The three models (SVR, Random Forest & XGB) delivered somewhat similar results, R2 score locally varying from 0.74 to 0.76 (when using 25% test data split). The effect of using the best hyperparameters vs. default values when defining the models turned out to be minimal.

We also tried different combinations of the features to see if we could improve the performance. We ended up dropping several features that were produced by one-hot encoding of *parentspecies* variable, as they seemed to contain little information with both train and test data. These variables were *category_None*, *category_apin_decane*, *category_apin_decane_toluene*, *category_apin_toluene* and *category_decane_toluene*. We also removed *C.C.C.O.in.non.aromatic.ring*, *aromatic.hydroxyl* and *nitroester* variables as they too contained only little information. These feature selection methods resulted in a small improvement in the performance of the models.

The evaluation of the optimization process for the Kaggle competition turned out to be challenging, as the R2 score in Kaggle did not correlate with local results. In the end we obtained the best results with SVR using standard scaler, default hyperparameters and the feature selection as described above.

Feature selection

Results

Discussion