Master's Programme in Data Science

Forecasting Saturation Vapour Pressure (Group 48)

Nooa Saareks, Markus Tarma, Sanni Tuomisto

December 2024

University of Helsinki
Faculty of Science

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki

Contents

1. Introduction

Predicting saturation vapor pressure using machine learning involves a series of well-defined steps. This report provides a comprehensive overview of the process, encompassing the selection, creation, optimization, and finalization of regression models for this task. The workflow begins with exploratory data analysis (EDA), which entails investigating and evaluating the dataset. Subsequently, feature engineering and selection are conducted to identify and select the most relevant features for the model. Finally, model selection is performed, followed by optimization of the chosen models to achieve optimal performance.

2. Exploratory data analysis

The first step was to perform an exploratory data analysis (EDA) to understand the dataset better. We utilized Python libraries including Pandas and NumPy for data manipulation, and Matplotlib and Seaborn for visualization.

It can be seen (Table 2.1) that the parent species toluene makes up about 67.4 percent of the train data set, with apin forming 23.1 and decane 8.3 percent, while the rest of the parent species make up non-significant portions of the train data set. Even though we don't have the labels of the test set, we can observe that the test set has very similar proportions of these parent species.

Parent species	**Count**	**Ratio (%)**
toluene	17,950	67.4
apin	6,165	23.1
decane	2,218	8.3
apin_decane	46	0.2
apin_toluene	37	0.1
apin_decane_toluene	9	0.0
decane_toluene	2	0.0

Table 2.1: Counts and ratios of parent species in train data.

We visualized feature distributions using histograms to gain an overview of the feature values. From the histograms, we can observe that the values for the number of different functional groups (hydroxyl (alkyl), aldehyde, ketone, carboxylic acid, ester, ether (alicyclic), nitrate, nitro, aromatic hydroxyl, carbonylperoxynitrate, peroxide, hydroperoxide, carbonylperoxyacid, nitroester) in the molecules are discrete and most of them have a high number of zero values. We can also see that some of the features vary in their value ranges. The number of stable conformers (NumOfConf) and molecular weight (MW) have the largest ranges, with mean values in the hundreds, while most of the other feature values range from zero to five. This suggests that some form of scaling may be necessary. Additionally, the distribution of the number of stable conformers (NumOfConf) seems to be right-skewed.

Using Pearson correlation, we examined the linear relationships between features and the target variable log_pSat_Pa. While we anticipated that this method might not capture non-linear relationships well, it provided a useful initial understanding of the linear relationships. From the heatmap (Figure 2.2) we see that the most correlated features clustered near the upper left corner, indicated by darker shades of red and blue. Notable correlations included a strong positive relationship between molecular weight (MW) and

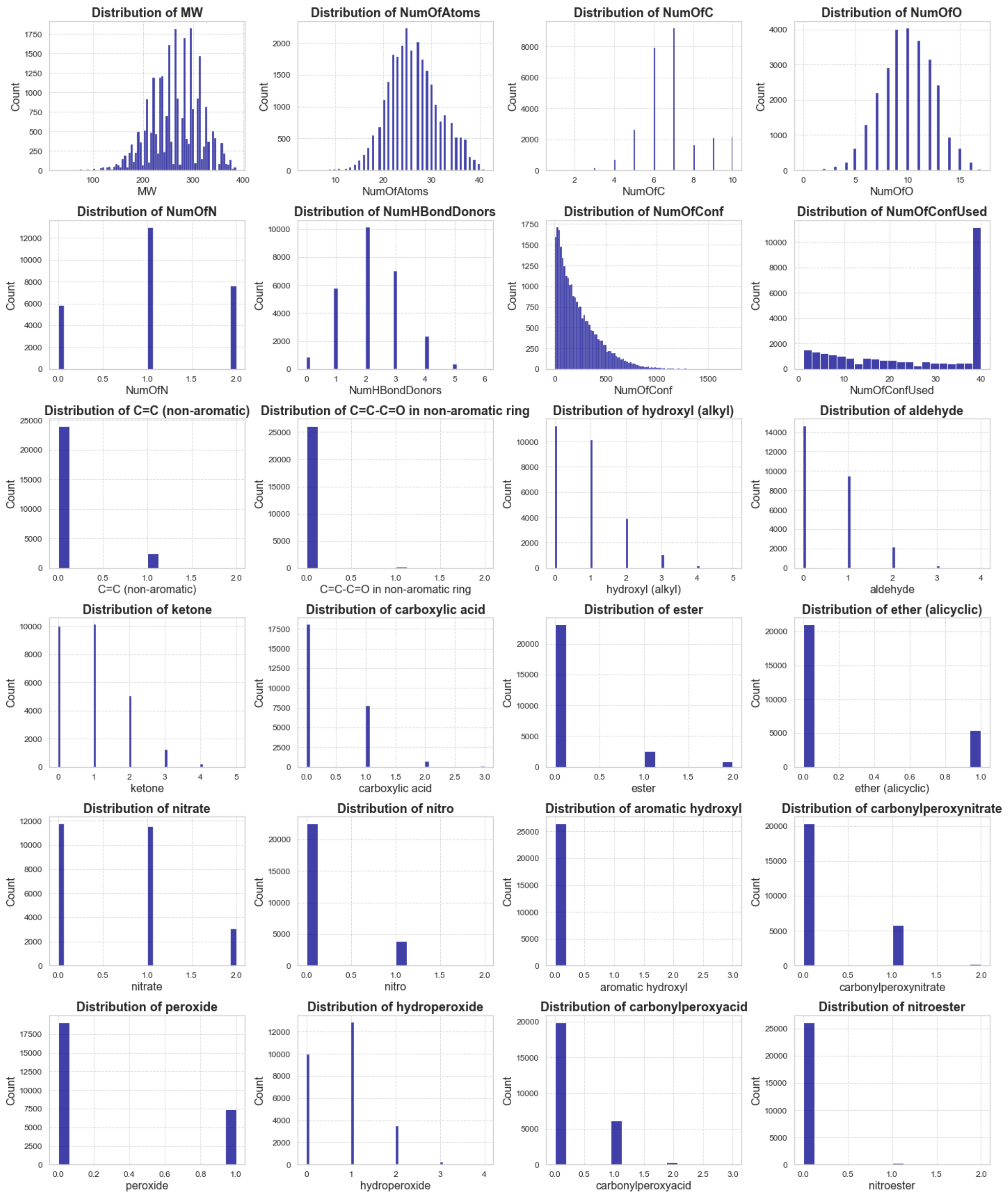
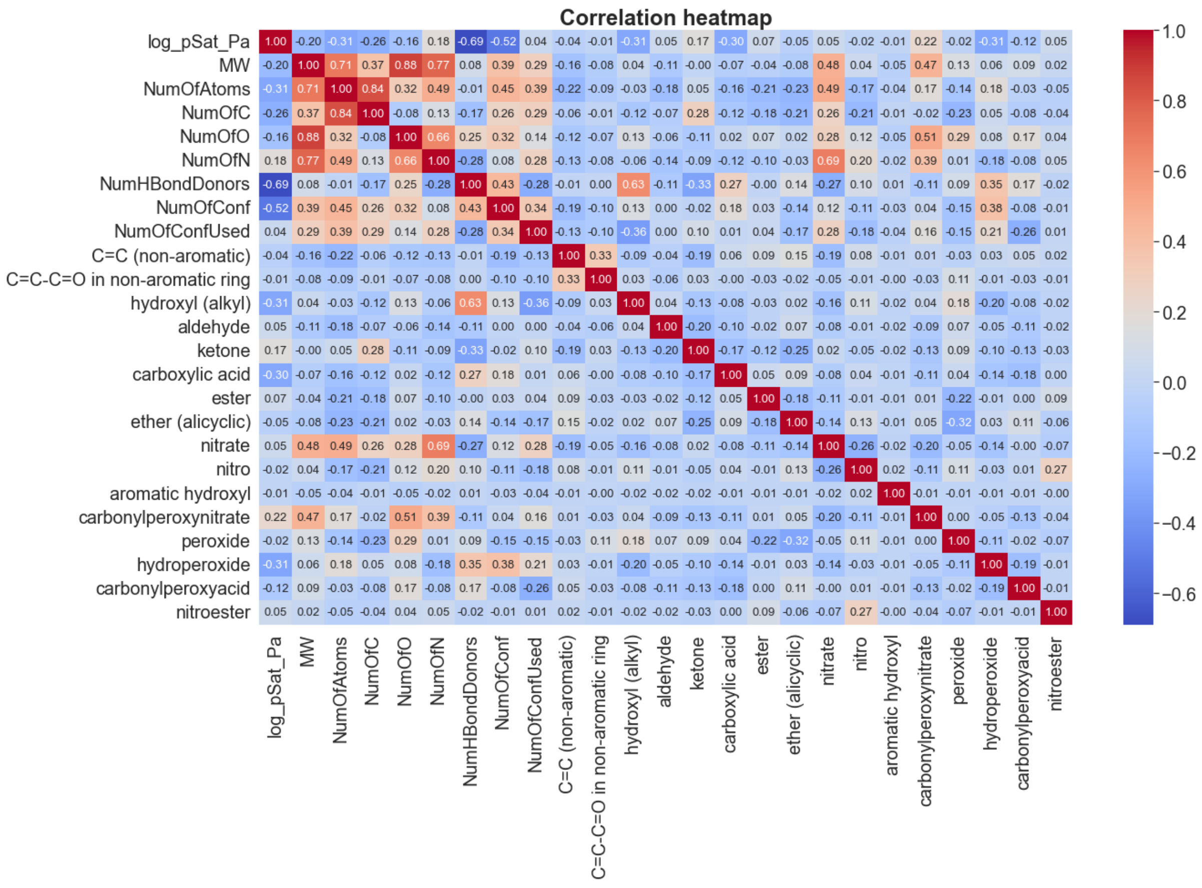


Figure 2.1: Histograms of the feature value distributions.

number of atoms, suggesting that molecules with more atoms tend to have greater mass. Additionally, the number of oxygen and nitrogen atoms demonstrated a strong positive correlation with molecular weight, while the number of carbon atoms demonstrated a strong correlation with the number of atoms in the molecule. The number of hydrogen bond donors (NumHBondDonors) and the number of stable conformers (NumOfConf) seem to have the strongest linear relationship with the target variable, both showing a negative correlation.



	log_pSat_Pa	MW	NumOfAtoms	NumOfC	NumOfO	NumOfN	NumHBondDonors	NumOfConf	NumOfConfUsed	C=C (non-aromatic)	C=C-C=O in non-aromatic ring	hydroxyl (alkyl)	aldehyde	ketone	carboxylic acid	ester	ether (alicyclic)	nitrate	nitro	aromatic hydroxyl	carbonylperoxynitrate	peroxide	hydroperoxide	carbonylperoxyacid	nitroester
log_pSat_Pa	1.00	-0.20	-0.31	-0.26	-0.16	0.18	-0.69	-0.52	0.04	-0.04	-0.01	-0.31	0.05	0.17	-0.30	0.07	-0.05	0.05	-0.02	-0.01	0.22	-0.02	-0.31	-0.12	0.05
MW	-0.20	1.00	0.71	0.37	0.88	0.77	0.08	0.39	0.29	-0.16	-0.08	0.04	-0.11	-0.00	-0.07	-0.04	-0.08	0.48	0.04	-0.05	0.47	0.13	0.06	0.09	0.02
NumOfAtoms	-0.31	0.71	1.00	0.84	0.32	0.49	-0.01	0.45	0.39	-0.22	-0.09	-0.03	-0.18	0.05	-0.16	-0.21	-0.23	0.49	-0.17	-0.04	0.17	-0.14	0.18	-0.03	-0.05
NumOfC	-0.26	0.37	0.84	1.00	-0.08	0.13	-0.17	0.26	0.29	-0.06	-0.01	-0.12	-0.07	0.28	-0.12	-0.18	-0.21	0.26	-0.21	-0.01	-0.02	-0.23	0.05	-0.08	-0.04
NumOfO	-0.16	0.88	0.32	-0.08	1.00	0.66	0.25	0.32	0.14	-0.12	-0.07	0.13	-0.06	-0.11	0.02	0.07	0.02	0.28	0.12	-0.05	0.51	0.29	0.08	0.17	0.04
NumOfN	0.18	0.77	0.49	0.13	0.66	1.00	-0.28	0.08	0.28	-0.13	-0.08	-0.06	-0.14	-0.09	-0.12	-0.10	-0.03	0.69	0.20	-0.02	0.39	0.01	-0.18	-0.08	0.05
NumHBondDonors	-0.69	0.08	-0.01	-0.17	0.25	-0.28	1.00	0.43	-0.28	-0.01	0.00	0.63	-0.11	-0.33	0.27	-0.00	0.14	-0.27	0.10	0.01	-0.11	0.09	0.35	0.17	-0.02
NumOfConf	-0.52	0.39	0.45	0.26	0.32	0.08	0.43	1.00	0.34	-0.19	-0.10	0.13	0.00	-0.02	0.18	0.03	-0.14	0.12	-0.11	-0.03	0.04	-0.15	0.38	-0.08	-0.01
NumOfConfUsed	0.04	0.29	0.39	0.29	0.14	0.28	-0.28	0.34	1.00	-0.13	-0.10	-0.36	0.00	0.10	0.01	0.04	-0.17	0.28	-0.18	-0.04	0.16	-0.15	0.21	-0.26	0.01
C=C (non-aromatic)	-0.04	-0.16	-0.22	-0.06	-0.12	-0.13	-0.01	-0.19	-0.13	1.00	0.33	-0.09	-0.04	-0.19	0.06	0.09	0.15	-0.19	0.08	-0.01	0.01	-0.03	0.03	0.05	0.02
C=C-C=O in non-aromatic ring	-0.01	-0.08	-0.09	-0.01	-0.07	-0.08	0.00	-0.10	-0.10	0.33	1.00	0.03	-0.06	0.03	-0.00	-0.03	-0.02	-0.05	-0.01	-0.00	-0.03	0.11	-0.01	-0.03	-0.01
hydroxyl (alkyl)	-0.31	0.04	-0.03	-0.12	0.13	-0.06	0.63	0.13	-0.36	-0.09	0.03	1.00	0.04	-0.13	-0.08	-0.03	0.02	-0.16	0.11	-0.02	0.04	0.18	-0.20	-0.08	-0.02
aldehyde	0.05	-0.11	-0.18	-0.07	-0.06	-0.14	-0.11	0.00	0.00	-0.04	-0.06	0.04	1.00	-0.20	-0.10	-0.02	0.07	-0.08	-0.01	-0.02	-0.09	0.07	-0.05	-0.11	-0.02
ketone	0.17	-0.00	0.05	0.28	-0.11	-0.09	-0.33	-0.02	0.10	-0.19	0.03	-0.13	-0.20	1.00	-0.17	-0.12	-0.25	0.02	-0.05	-0.02	-0.13	0.09	-0.10	-0.13	-0.03
carboxylic acid	-0.30	-0.07	-0.16	-0.12	0.02	-0.12	0.27	0.18	0.01	0.06	-0.00	-0.08	-0.10	-0.17	1.00	0.05	0.09	-0.08	0.04	-0.01	-0.11	0.04	-0.14	-0.18	0.00
ester	0.07	-0.04	-0.21	-0.18	0.07	-0.10	-0.00	0.03	0.04	0.09	-0.03	-0.03	-0.02	-0.12	0.05	1.00	-0.18	-0.11	-0.01	-0.01	0.01	-0.22	-0.01	0.00	0.09
ether (alicyclic)	-0.05	-0.08	-0.23	-0.21	0.02	-0.03	0.14	-0.14	-0.17	0.15	-0.02	0.02	0.07	-0.25	0.09	-0.18	1.00	-0.14	0.13	-0.01	0.05	-0.32	0.03	0.11	-0.06
nitrate	0.05	0.48	0.49	0.26	0.28	0.69	-0.27	0.12	0.28	-0.19	-0.05	-0.16	-0.08	0.02	-0.08	-0.11	-0.14	1.00	-0.26	-0.02	-0.20	-0.05	-0.14	-0.00	-0.07
nitro	-0.02	0.04	-0.17	-0.21	0.12	0.20	0.10	-0.11	-0.18	0.08	-0.01	0.11	-0.01	-0.05	0.04	-0.01	0.13	-0.26	1.00	0.02	-0.11	0.11	-0.03	0.01	0.27
aromatic hydroxyl	-0.01	-0.05	-0.04	-0.01	-0.05	-0.02	0.01	-0.03	-0.04	-0.01	-0.00	-0.02	-0.02	-0.02	-0.01	-0.01	-0.01	-0.02	0.02	1.00	-0.01	-0.01	-0.01	-0.01	-0.00
carbonylperoxynitrate	0.22	0.47	0.17	-0.02	0.51	0.39	-0.11	0.04	0.16	0.01	-0.03	0.04	-0.09	-0.13	-0.11	0.01	0.05	-0.20	-0.11	-0.01	1.00	0.00	-0.05	-0.13	-0.04
peroxide	-0.02	0.13	-0.14	-0.23	0.29	0.01	0.09	-0.15	-0.15	-0.03	0.11	0.18	0.07	0.09	0.04	-0.22	-0.32	-0.05	0.11	-0.01	0.00	1.00	-0.11	-0.02	-0.07
hydroperoxide	-0.31	0.06	0.18	0.05	0.08	-0.18	0.35	0.38	0.21	0.03	-0.01	-0.20	-0.05	-0.10	-0.14	-0.01	0.03	-0.14	-0.03	-0.01	-0.05	-0.11	1.00	-0.19	-0.01
carbonylperoxyacid	-0.12	0.09	-0.03	-0.08	0.17	-0.08	0.17	-0.08	-0.26	0.05	-0.03	-0.08	-0.11	-0.13	-0.18	0.00	0.11	-0.00	0.01	-0.01	-0.13	-0.02	-0.19	1.00	-0.01
nitroester	0.05	0.02	-0.05	-0.04	0.04	0.05	-0.02	-0.01	0.01	0.02	-0.01	-0.02	-0.02	-0.03	0.00	0.09	-0.06	-0.07	0.27	-0.00	-0.04	-0.07	-0.01	-0.01	1.00

Figure 2.2: Heatmap showing the pairwise Pearson correlations between features and the target variable. Stronger correlations are represented by darker shades of red (positive correlation) and blue (negative correlation).

3. Feature engineering

3.1 Clustering the data

By clustering the data into, for example, 3 clusters it can be seen in table 3.1 that toluene makes up most of the 2 clusters while the rest are mostly packed to the one left This division isn't perfect though as apin and decane do spill over to the other 2 cluster. When the test data is clustered, the result look quite similar.

Parent Species	**Cluster 0**	**Cluster 1**	**Cluster 2**
apin	68.4	11.5	20.1
apin_decane	0.0	0.0	100.0
apin_decane_toluene	0.0	0.0	100.0
apin_toluene	0.0	20.0	80.0
decane	82.7	1.0	16.3
toluene	1.7	49.8	48.5

Table 3.1: Percentage distribution of species across different clusters

The parent species is the only categorical feature in the data set. We started by dividing the data into toluene and other, which only minimally enhanced our results. We clustered the combined test and data set to do some semi-supervised learning and thus enhance the model by using the test set as much as possible. We further investigated whether or not it makes sense to divide the data set and fit separate models, or whether we should rather use PCA on the combined set or even do both. Trying to divide the data sets by cluster or by the values of parent species, we were not able to reach a cross-validated R-squared of more than 72 percent. Thus, the idea of dividing the data set by clusters was abandoned, and we decided to use one-hot encoding instead to convert categorical data into a format that is suitable for the regression model.

3.2 Encoding Categorical Feature

We used one-hot encoding to convert the categorical values of the parentspecies column (e.g., toluene, apin, decane, apin_toluene, apin_decane, apin_decane_toluene, decane_toluene) into a numerical format. This method creates a binary column for each category, assigning a value of 1 if an observation belongs to a specific parent species category and 0 otherwise. One-hot encoding handles missing values by assigning 0 to all encoded columns for observations where the parentspecies value is missing. The parentspecies column was the only one with missing values, with 210 missing entries in the training dataset. We performed encoding using scikit-learn's OneHotEncoder [1].

3.3 Scaling

During the EDA phase, we observed that some features had different value ranges. To bring the features to a similar scale before using them in the model, wie chose to apply standardization, that is, scaling the values to have a zero mean and unit variance. We implemented this using the scikit-learn's StandardScaler [2]. Standardization can be especially important before performing Principal Component Analysis (PCA), as the results of PCA can be affected by the scale of the features [2]. Features with larger variances can dominate the principal components and therefore scaling the features to have a standard deviation of one is generally beneficial. We decided to use PCA, as discussed later in this report, which made standardization a logical choice.

Machine learning models often struggle with features that have a skewed distribution [1]. As we noticed while performing EDA, the column NumOfConf had a right-skewed distribution (Figure 2.1). One way to mitigate this problem is to apply a log transformation, that is, taking the log of the skewed feature. As we can see from the Figure 3.1, the log transformation seems to have reduced the skewness of the NumOfConf distribution. Log transformation does not work in all cases and should be applied with caution. However, since it provided better results with our model, we chose to apply it.

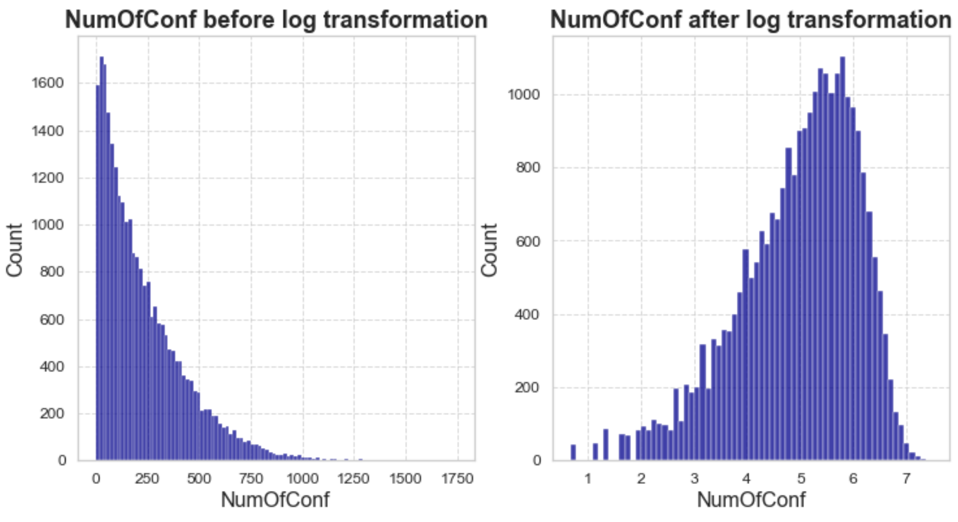


Figure 3.1: Distribution of the NumOfConf before and after applying log transformation. The left histogram shows the original distribution, while the right histogram illustrates the effect of the log transformation.

[1]https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html
[2]https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

4. Feature selection

In our first feature selection experiment, we used Random Forest Regression to identify the most important features. Random Forest builds multiple decision trees on random subsets of the data. Subsets are created with bootstrapping, where a random sample of the training data is selected with replacement. At each split in the tree, only a random subset of features is considered to find the optimal split, rather than using all features, and the final prediction is made by averaging the predictions from all trees [2].

Feature importance scores are calculated based on how frequently and effectively each feature is used to split the data in the trees. We used sklearn's RandomForestRegressor [1] to calculate feature importance scores, which uses impurity-based feature importance. This method measures how much a feature helps reduce variance when used for splitting.

The computed feature importance scores are shown in Figure 4.1. These scores were calculated using one-hot encoding, but without scaling the features. It appears that the ranking follows the cardinality of the features, with features of lower cardinality ranked lower and those of higher cardinality ranked higher. After obtaining these results, we found that this method can have bias issues with features with higher cardinality [3]. We noticed that this is acknowledged as a potential issue in scikit-learn's documentation for RandomForestRegressor upon closer review.

One-hot encoding was likely not the best option for the parentspecies column in this case, as it created binary columns for each species, resulting in low cardinality. We did not explore this method further, as selecting features based on these results did not lead to the desired improvements with the models. Instead, we were more inclined to use Principal Component Analysis (PCA) to reduce the dimensionality of the data. PCA achieves this by identifying and retaining the components that explain the most variance in the data. Further details of our work with PCA are in Chapter 5.

[1]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

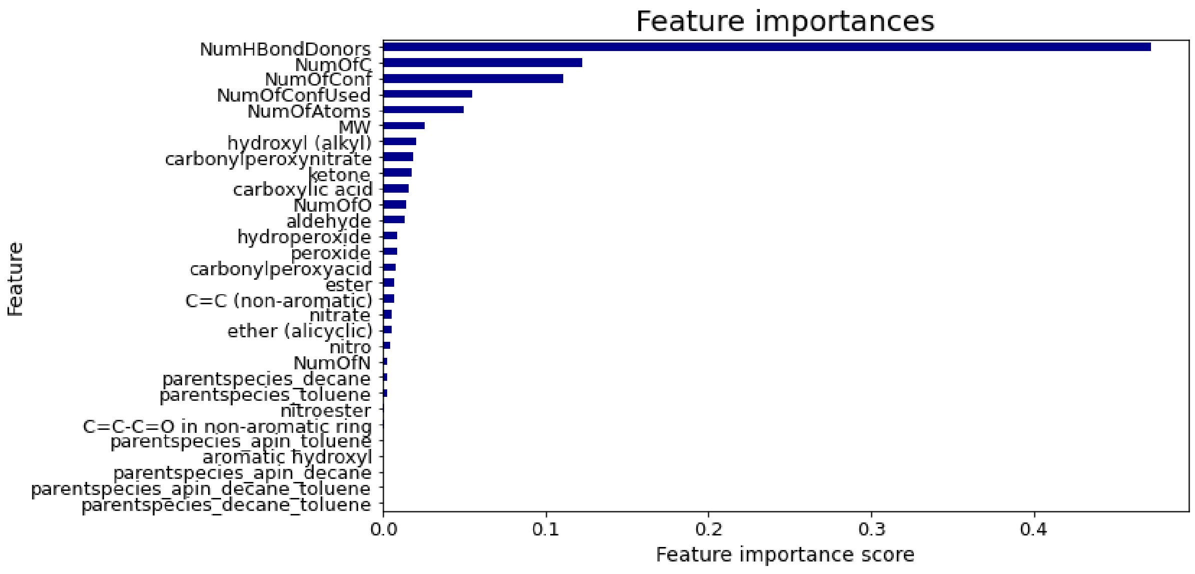


Figure 4.1: Feature importances computed using Random Forest.

5. Model selection

After completing the exploratory data analysis and pre-processing steps, we turned our focus to selecting the most suitable regression model for our project. Our approach emphasized widely recognized and commonly used models to ensure reliable comparisons.

5.1 Choosing the model

The model selection process involved evaluating several algorithms and comparing their performance metrics, as summarized in Table 5.1.

Table 5.1: Regressor Performance (sorted by R2 score)

Regressor	**MSE**	**MAE**	**R2 Score**
MLP Regressor	2.5071	1.1519	0.7443
SVR (RBF)	2.5553	1.1603	0.7370
Random Forest Regressor	2.6740	1.2024	0.7249
LightGBM	2.7021	1.2170	0.7219
Gradient Boosting Regressor	2.7027	1.2167	0.7218
Linear Regression	2.8939	1.2682	0.7022
SVR (Linear)	2.9060	1.2670	0.7009
Decision Tree Regressor	3.5253	1.4275	0.6373
AdaBoost Regressor	4.0385	1.5680	0.5806

Upon evaluating the performance of these models, the MLP regressor, Random Forest Regression and SVR with a RBF kernel emerged as the most promising candidates. After choosing these three models, we ran GridSearch and optuna searches to investigate whether these models have potential for further refinement.

5.2 Optimizing PCA and model performance

We combined our model selection with PCA to see if using some combination of dimensions from PCA and some model would deliver more promising results. In this research, we used the three models that had previously been deemed the most suitable. We used the hyperparameters that had previously been deemed optimal with the whole data set. PCA was run on the combined data set of both the test data and training data, in order to take advantage of the structure of the test data as well, even though we lacked the

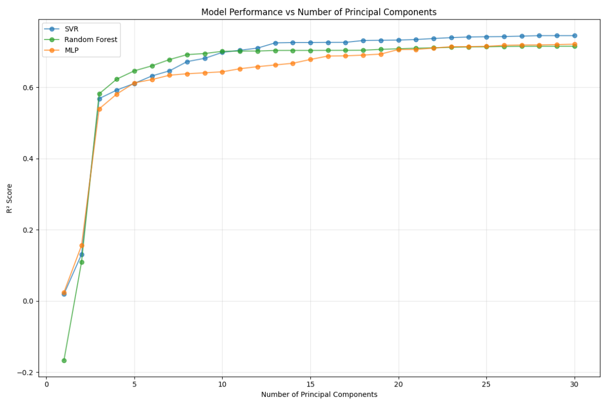


Figure 5.1: Model R^2 vs number of PCA components

labels, as a form of semi-supervised learning. The R-squared scores were determined by cross-validation. The results can be seen in figure 5.1. It can be seen that initially the RF regressor seems to perform better than SVR but as the number of PCA components is raised, the SVR is clearly the best. This confirms our previous analysis of the fact that SVR is clearly the best for this task.

In the analysis above, the same hyperparameters were used for all PCA components. We decided to inspect more closely if the performance could be further improved by finding optimal parameters for each number of principal components, the result of which can be seen in table 5.2. First of all, it can be seen that the optimal hyperparameters can differ quite much when including any additional principal components. Secondly we were able to make some improvements to the R-squared by optimizing parameters separately.

PCA Components	Best R^2	Best C	Best gamma	Best epsilon
20	0.745316	69.700568	0.01	0.52
21	0.748076	84.624383	0.01	0.50
22	0.748455	81.911551	0.01	0.65
23	0.748860	4.330488	scale	0.31
24	0.753822	91.231236	0.01	0.70
25	0.753823	94.373087	0.01	0.76
26	0.753964	75.709698	0.01	0.66
27	0.753668	93.631253	0.01	1.00
28	0.753956	76.854501	0.01	0.51
29	0.754047	93.523074	0.01	0.59
30	0.754029	92.433083	0.01	0.64

Table 5.2: Performance metrics for different PCA components for SVR

Since our original SVR fitted on the whole dataset showed some signs of overfitting with an R-squared of 77.2 per cent versus 75.3 in cross-validation, we decided to use only 24 principal components, as the validation score is virtually identical (within a 0.0001 per cent difference) and using 6 principal components less reduced the model's variance a bit. We could have also used only up to 21 principal components, as the R-squared there is within 99 per cent of the best one. Eventually we found that using 24 principal components brought slightly better results. We received an R-squared of 0.7475 as the private score and 0.7598 as the public score.

6. Self grading

We believe that the work deserves a grade of 4. The work was done in a holstic and complete way showing a good understanding of machine learning, the data and the source material.

We believe that the methods, including different machine learning techniques and feature selection techniques have been analysed sufficiently as we considered most of the relevant regression models as well as looking at using both PCA and clustering. On top of this the feature engineering and selection was done well with us considering the main issues such as missing values, scaling and log transformation. The advantages and disadvanteges were discussed for most techniques and we made well argumented decisions on them such as abandoning the use of clustering and feature selection using Random Forest.

In the end the conclusion that SVR is our best choice was presented clearly and concisely and we showed critical thinking around it. The work was done according to the schedule.

References

[1] C. Huyen. *Designing machine learning systems : an iterative process for production-ready applications / Chip Huyen.* O'Reilly Media, Incorporated, Sebastopol, first edition edition, 2022.

[2] G. G. M. James, D. Witten, T. Hastie, R. Tibshirani, and J. Taylor. *An introduction to statistical learning : with applications in Python.* Springer Texts in Statistics. Springer US, New York, NY, 2023.

[3] C. Strobl, A.-L. Boulesteix, A. Zeileis, and T. Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics*, 8(1):25, 2007.