Intro to ML: Term Project Report

Group 312: Théo Friberg, Samu Syrjänen and Tatu Linnala

December 2023

1 Introduction

This report details our solution to the term project of the Introduction to Machine Learning course lectured at the University of Helsinki. This project included a Kaggle competition. We submitted our solutions to Kaggle using "Group 312" as our group name.

The task was to predict the saturation vapour pressure of different molecules based on other features. Essentially, our goal was to build a machine learning model to predict how sensitive certain substances are to evaporation. We were given two sets of data for the project. The training set was used to train and test the machine learning models. The competition data, which didn't contain the target values, was used in the competition to rank solutions submitted by competing teams. The solutions were ranked based on the R^2 score, so we focused on optimizing this.

1.1 Exploring the Data

The input data has the columns given in Table 1. We exclude the `Id` column from consideration since it is chemically irrelevant. The `parentspecies` variable is both the only one to have missing values and to be categorical. We encoded this variable using one–hot encoding. Keeping the encoded `parentspecies` variable seemed to result in slightly better–performing models. The remaining columns are – to our chemical understanding – not categorical and thus form an $\mathbb{R}^n$-space (though some, chiefly the one starting by `Num`, are discrete).

We chose to transform the target variable such that $y = \log_{10}(\text{pSat\_Pa})$. On a logarithmic scale, y is distributed in a Gaussian fashion, as seen in Figure 1. This is preferable since most machine learning methods work best with normally distributed variables. Table 2 details more statistical properties of this distribution. Interestingly, we have highly volatile compounds with a saturation pressure on the order of 10^5 and highly non-volatile compounds with saturation pressures on the order of 10^{-13}. We detected some of the data points with very high `pSat_Pa` values to be outliers. However, due to a lack of domain knowledge, we did not remove these points from the data. Besides giving us a Gaussian distribution, working with the base-10 logarithm of the values helps us deal with these extreme values. Without this transformation, the highly volatile compounds might pose leverage point issues with some machine learning methods.

The two-dimensional histogram in Figure 1 shows that there is a clear trend between the size of the molecule (here, in terms of the molecular weight, but also in terms of the number of atoms) and volatility. We can convince ourselves of this in a couple of cases by looking at the rows of data themselves as in Table 1. By inspecting sample 1 on the table, we can deduce the compound to be formaldehyde based on the values of `NumberOfAtoms`, `NumOfC` and `NumOfO`.[1] According to Wikipedia, formaldehyde has a boiling point of -19°C. This means that formaldehyde is comparatively easily evaporated, which means that it should have a high saturation vapour pressure. After inspecting the data and `pSat_Pa` statistics, we can confirm our hypothesis. Sample 2 is a heavy organic compound which we would expect to be solid at room temperature. The value of `pSat_Pa` confirms this.

[1] It has four atoms, one of which is a carbon atom and one of which is an oxygen atom. The oxygen and carbon need to be connected to each other, assuming the remaining two atoms to be hydrogens. The bond must be a double covalent bond, since if it were a single bond, there would be two unused bonds. This deduction shows it must be formaldehyde, which is confirmed by the fact that it also has one aldehyde group.

Comment	Field	Sample 1	Sample 2
Synthetic number of molecule	Id	1	166420
	MW	30.010564684	359.992467556
	NumOfAtoms	4	32
	NumOfC	1	7
	NumOfO	1	15
	NumOfN	0	2
	NumHBondDonors	0	3
	NumOfConf	1	165
	NumOfConfUsed	1	3
Categorical, missing values	parentspecies	apin_decane_toluene	toluene
	C.C..non.aromatic.	0	0
	C.C.C.O.in.non.aromatic.ring	0	0
	hydroxyl..alkyl.	0	3
	aldehyde	1	0
	ketone	0	0
	carboxylic.acid	0	0
	ester	0	0
	ether..alicyclic.	0	0
	nitrate	0	0
	nitro	0	0
	aromatic.hydroxyl	0	0
	carbonylperoxynitrate	0	2
	peroxide	0	1
	hydroperoxide	0	0
	carbonylperoxyacid	0	0
	nitroester	0	0
The target variable	pSat_Pa	641974.491	151549.268

Table 1: The columns of the chemical data.

Number of samples	27147
μ	-3.854276
σ	2.178311
min	-13.789350
25th percentile	-5.247255
50th percentile	-3.808918
75th percentile	-2.400747
max	5.864807

Table 2: Details of the $\log_{10}$(pSat_pa).

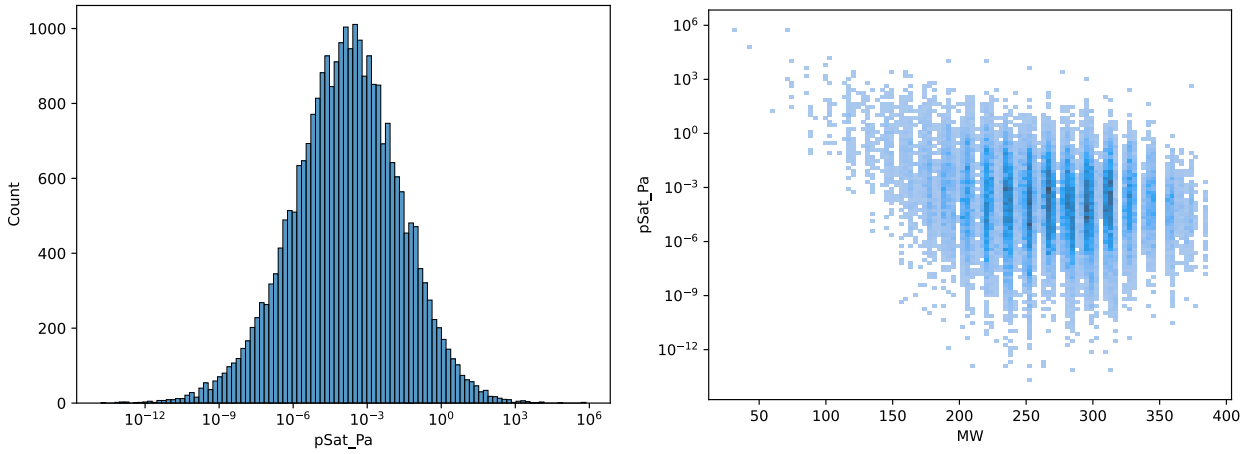

Figure 1: One and two dimensional histograms of `pSat_Pa`. Note the $\log_{10}$ `pSat_Pa` axis on both. On the right, the horizontal axis is molecular weight.

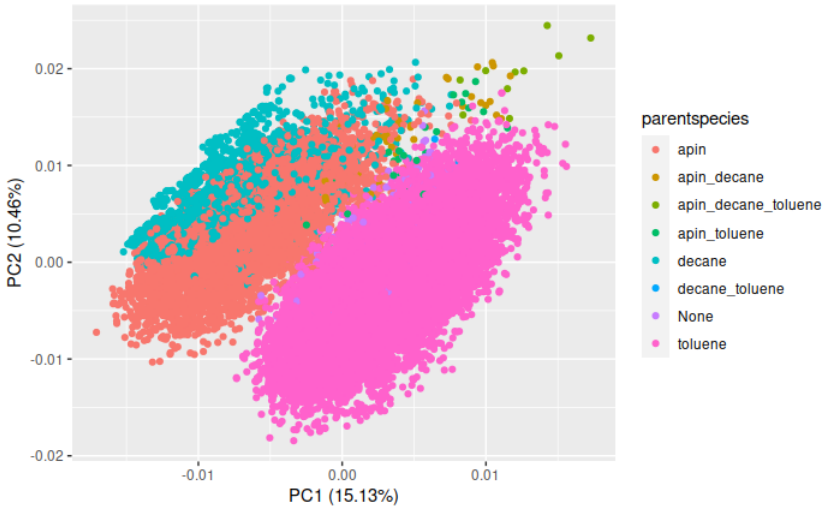

Figure 2: The first two principal components of the data.

1.2 Principal Component Analysis

We performed principal component analysis (PCA) on the data. For this purpose, we used data without the `parentspecies` variable. The first two principal components, together with the data, are plotted in Figure 2. The data points are coloured based on the `parentspecies` variable. From the figure, we can see the data points form clusters with different `parentspecies` values.

2 Methods

We tested multiple different machine learning models. We started from the simplest ones: a dummy model and linear regression models. Lasso regularization was added to the linear models. We also tested random forest and gradient boosting models.

2.1 Considered Models

Dummy Model

For comparison, we made two dummy regressors using sklearns DummyRegressor. The first dummy (dummy1) is using the target mean, and the second dummy (dummy2) is using the median. Here are the 5-fold cross-validation R^2 scores of the dummies:

```
             dummy 1 (mean)
R2 score     -0.0553571913024419

             dummy 2 (median)
R2 score     -0.05601824879336901
```

Linear Regression

Next, we try to fit a simple linear regression into the data. First, we normalize the feature values so that features with large values are initially weighted similarly to features with small values. Here is the normalization function:

```
def normalize(X, X_train):
    return (X - X_train.mean()) / X_train.std()
```

L1 α	Average R^2 under 10-fold CV	R^2 on hidden data set
0	0.650582	0.6554
0.1	0.436493	0.63789

Table 3: Performances of two quadratic models.

We fit a simple linear regressor with intercept term into the normalized data. Since the features were normalized, the coefficients should indicate the weights of the features regarding how well they explain the data. We unfortunately observed that the calculated coefficients have extremely high variance between each run. It means that they do not accurately represent the importances of the features.

Despite the varying coefficients, the 5-fold cross-validation score was consistent:

```
            Linear regression
R2 score    0.671978952994792
```

From the R^2 score we can see that the linear regressor already performed considerably better than the dummy models.

Quadratic Regression and Lasso

Two quadratic models were fit against $\log_{10}$(pSat_Pa) and the input data was normalized for mean and standard deviation. A third-degree polynomial was also tested, but it was discarded because of exploding interaction term count and poor performance under cross-validation. Models were fitted with interaction terms. Table 3 details the performances of the different models.

We were interested in seeing whether Lasso regularization could yield a model with fewer terms and still reasonable results. For $\alpha = 0.1$ and under cross-validation, we get an average R^2 of 0.436493. On the test data, this model performed at $R^2 = 0.63789$, which was less than the non-regularized model, but produced a significantly sparse twenty-term polynomial sum which (with factors rounded) looks like this:

$$
\begin{aligned}
\log_{10}(\text{pSat\_Pa}) \approx & -0.6263 \cdot \text{NumOfC} \\
& -1.2422 \cdot \text{NumHBondDonors} \\
& -0.3150 \cdot \text{NumOfConf} \\
& -0.1639 \cdot \text{carboxylic.acid} \\
& +0.1355 \cdot \text{carbonylperoxynitrate} \\
& -0.0287 \cdot \text{peroxide} \\
& -0.0092 \cdot \text{hydroperoxide} \\
& +0.0170 \cdot \text{MWNumHBondDonors} \\
& +0.0119 \cdot \text{NumOfC}^2 \\
& +0.0859 \cdot \text{NumOfO}^2 \\
& +0.0753 \cdot \text{NumHBondDonors} \cdot \text{NumOfConfUsed} \\
& +0.0124 \cdot \text{NumOfConfUsed} \cdot \text{carbonylperoxyacid} \\
& -0.0568 \cdot \text{C.C..non.aromatic.}^2 \\
& -0.0363 \cdot \text{carboxylic.acid}^2 \\
& +0.0091 \cdot \text{ester}^2 \\
& -0.0016 \cdot \text{ether..alicyclic.}^2 \\
& -0.0004 \cdot \text{aromatic.hydroxyl}^2 \\
& +0.0397 \cdot \text{carbonylperoxynitrate}^2 \\
& -0.0130 \cdot \text{carbonylperoxyacid}^2 \\
& +0.0044 \cdot \text{nitroester}^2
\end{aligned}
$$

We can see that the two highest terms are the number of hydrogen bonds and the number of carbon atoms. Both have a negative contribution to the saturation vapour pressure, which sounds chemically reasonable: the first leads to strong intermolecular bonds and the second to high molecular weight.

Decision Trees

For tree-based models, we first attempted to fit a simple decision tree regressor into the data using different max_depth values. The cross-validated R^2 scores are shown in the figure 3. The best R^2 score was achieved with a max_depth of 8:

```
            Decision tree regresson
R2 score    0.638297
```

Random Forest and Gradient Boosting

We used the `RandomForestRegressor` from the `sklearn` library and `XGBRegressor` from the `xgboost` library. We computed the "out–of–the-box" R^2 scores, *i.e.* the scores without feature selection or hyperparameter optimization, for the random forest and gradient boosting models. We did this to evaluate what kind of performance we can expect from these models. Using 5–fold cross–validation, random forest gave us an $R^2 = 0.67$ gradient boosting $R^2 = 0.70$.

2.2 Model Tuning

After the initial testing, we chose the most promising models for further refining. The ones chosen were the random forest and gradient boosting models since these provided the best "out-of-the-box" performance. Linear models were not considered further since the choice of the polynomial degree would be difficult considering the number of features in the data. To improve model performance, we performed feature

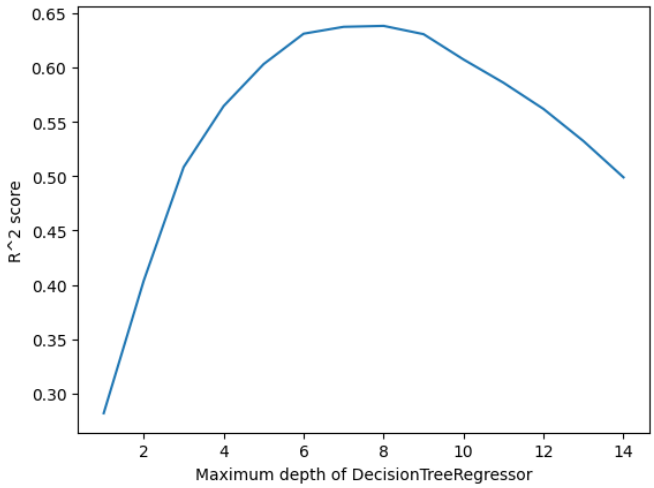

Figure 3: The R^2 scores of decision tree regressor using different max_depth values.

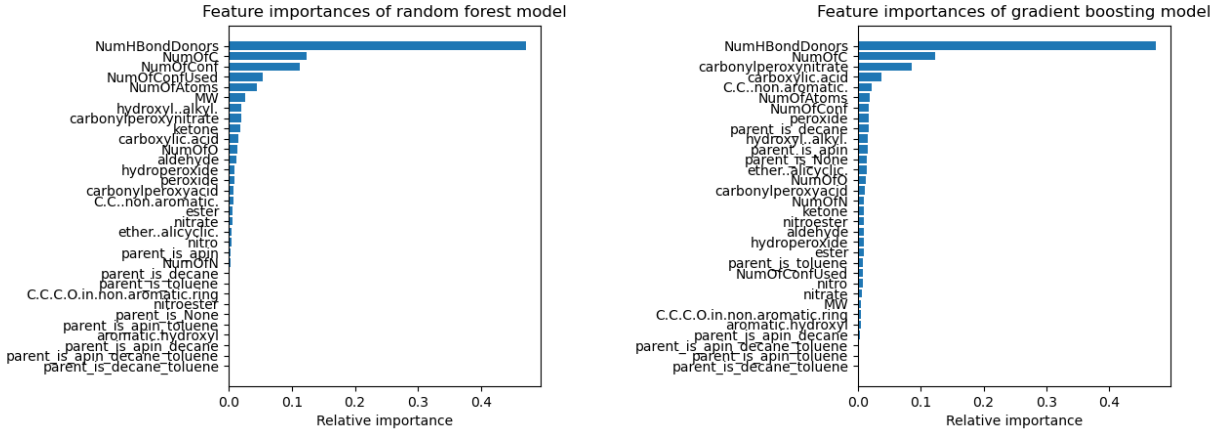

Figure 4: The feature importances of the random forest (left) and gradient boosting (right) models.

selection and hyperparameter optimization. Feature selection was performed first and hyperparameter optimization later using only the selected features.

Feature Importance and Feature Selection

Feature importances of the random forest and gradient boosting model were obtained using the corresponding `feature_importances_` methods. The feature importances are displayed in Figure 4. The most important feature with both models is, by a large margin, `NumHBondDonors`. The second most important feature is also the same with both models. The order of the other features varies between the models.

We performed feature selection using the forward subset selection technique. We started with a model that contained only the most important feature. We then added more features, one by one, in order of importance. We then picked the set of features that resulted in the best-performing model. We chose this relatively simple method because it's easy to implement and not too heavy computationally since the number of fitted models remains reasonable. We used the R^2 score as the performance metric, which we aimed to maximize. Five–fold cross–validation was used to compute the scores.

There were some differences in which features were kept with each model. With the random forest model, only the `parent_is_decane_tolune` feature was dropped. With the gradient boosting model, a total of seven

features were dropped, most of which were encoder variables for `parentspecies`.

With feature selection, we obtained an R^2 score of 0.68 for the random forest model and 0.70 for the gradient boosting model. In the case of the random forest model, the score improved slightly in comparison to the "out-of-the-box" score. On the other, the gradient boosting model did not seem to benefit from feature selection.

Hyperparameter Optimization

We used Optuna to optimize the hyperparameters. The R^2 score, which was maximized, was used as the objective. In the case of the random forest model, we saw that deviating some specific hyperparameters from their default values immediately resulted in poor model performance. We excluded these hyperparameters from the optimization and kept the default values. We did not use cross–validation in the optimization algorithm to keep the run–times reasonable.

With both models, optimizing the hyperparameters improved the R^2 scores slightly. With the random forest model, we saw an improvement from 0.68 to 0.72, and with the gradient boosting model, from 0.70 to 0.74.

3 Results & Conclusions

We submitted predictions to the Kaggle competition. The three most interesting submissions were made using the Lasso regularized linear model and the optimized random forest and gradient boosting models. With the gradient boosting model, we obtained an R^2 score of 0.6620. This was our best score. We obtained 0.6617 and 0.6379 with the random forest and quadratic models, respectively. All the models performed quite well. It was surprising how well the linear model performed compared to the other two models.

In general, we feel that we achieved respectable results on the real data, especially considering our lack of previous experience. We limited ourselves to methods shown on the course, as we had no previous experience regarding the topics of the course. These methods performed generally well.

There are a few things we feel we could have done better or explored further. We only did principal component analysis late in the process, and doing it earlier would have opened avenues that we did not explore. The PCA shows multiple clear clusters based on the different parent species, which we were hasty to dismiss for some models due to missing values. It would have been interesting to train simple regressors on the individual clusters or somehow better utilize this structure of our data. We limited ourselves to forward subset selection for our dimensionality reduction method. It would have been interesting to try other approaches as well. We also did not run our hyperparameter selection algorithm under cross–validation due to computational concerns. Using cross–validation in this step as well would have resulted in models that generalize better to the competition data.

4 Self-grading Report

4.1 Grade for the Deliverables: 4

The section 1 expresses a good understanding of the topic with multiple graphs and tables. The selected models are valid for the task and they were analysed well. Multiple different models were considered and tested to see which performs well. That process and the different models are described extensively enough in this report. The report clearly shows our results and conclusions regarding the different models, and is overall polished and "camera-ready". The project has been done independently from other groups and it shows some creativity in the steps taken to understand the data, evaluate models, and find the best combination of features and hyperparameters. The project was finished withing the given time frame and instructions were followed.

4.2 Grade for the Group as a Whole: 5

Discussions between the group members regarding the project were insightful and progress-oriented. Effort was made to ask and help if something was not clear or someone needed help. There was little irrelevant

chatter. At the beginning of the project, the group discussed the goals of the members regarding the project and found an agreement on how to carry it out. Despite the lack of time in each member's schedule, an effort was made by all the group members to contribute to the project. The learning outcomes were positively affected by the group sessions on campus.