

Computational Chemistry 2 Chapter 12 2021

Machine learning basics

There are several machine learning (ML) courses in Aalto so this lecture will not be very broad or deep.

Almost all science is fitting models to datasets. Experiments are designed to collect data from which knowledge is extracted by using accepted theories. The experimental data is fitted to theories if they exist (natural science vs. human science).

Now we can have a lot of data which is not connected to theories, like images, but there is some information in this data. How can we find information or correlations from vast data sets. Answer: Machine learning

ML is used in many fields, like pharmaceutical industry and gene studies (bioinformatics), image and speech recognition, machine translation, etc..

We meet the ML every day in applications like Apple Siri and in many net advertising sites.

Huge amount of ML methods has been collected to a python library **Scikit-learn**. This is a very convenient way to do ML computations.

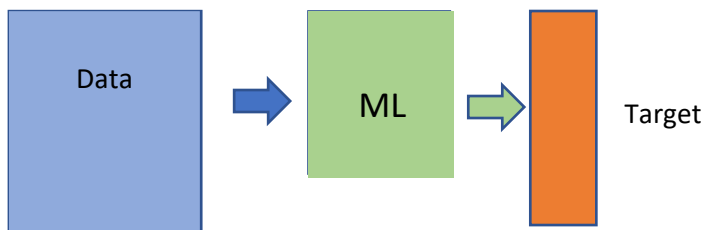
The sklearn is easy to use in python or in Jupyter notebooks

```
import sklearn
from sklearn.model_selection import KFold
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.kernel_ridge import KernelRidge
import matplotlib.pyplot as plt
```

Machine learning classes

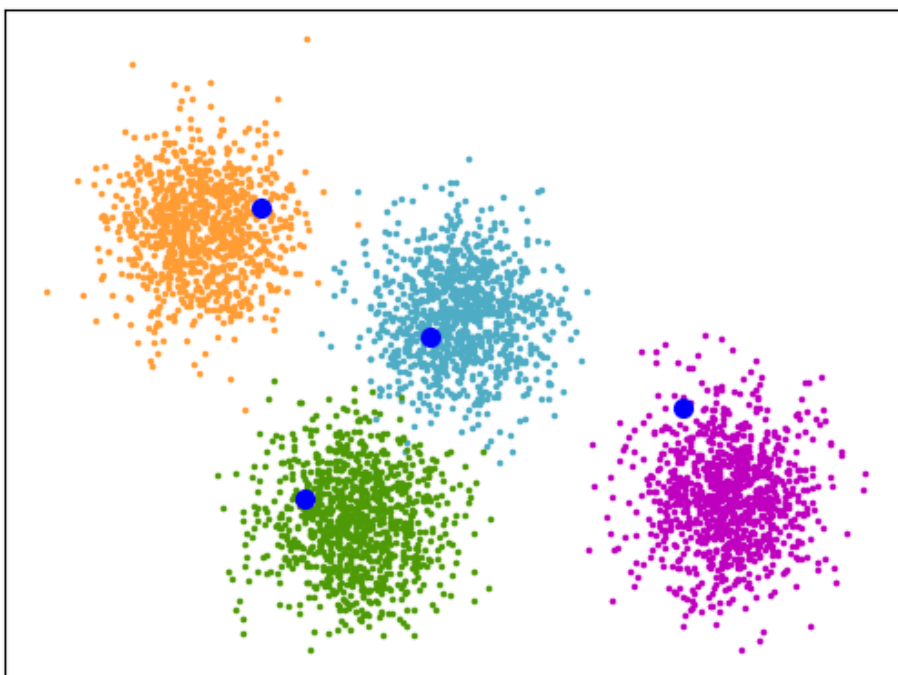
I used the <https://www.ibm.com/cloud/learn/machine-learning> web-page.

Supervised learning (SL): The aim is to learn known outputs and find good descriptors for the system. This is the most relevant ML for materials science. This is also a relatively easy ML problem.



Unsupervised learning (UL): The aim is to classify data and find patterns in it. Example: understanding hand-written text. This is more difficult and usually a lot of data is needed. Typically, the UL methods will cluster data with some similarity methods.

K-Means++ Initialization



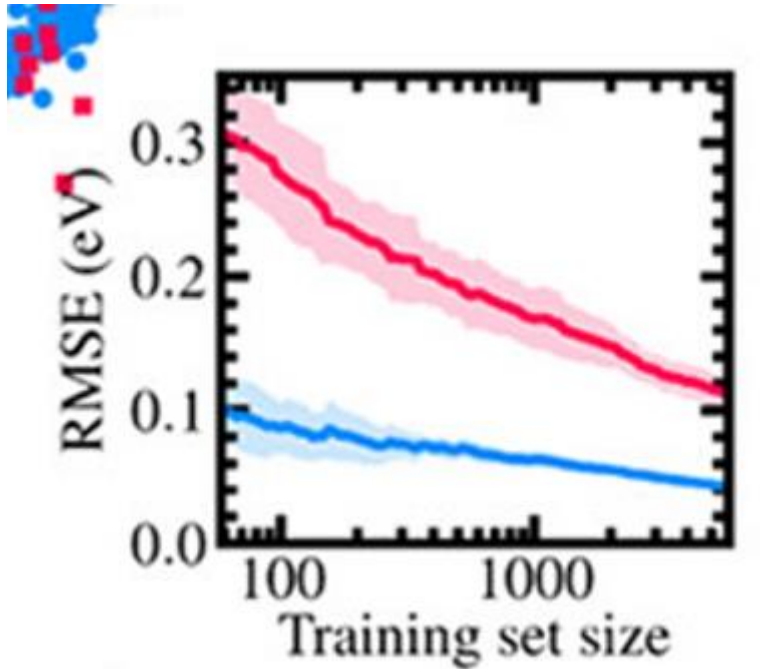
Semi-supervised learning: A mixture of the two methods above. For example, in some cases the output is known. Example: we can know some of the hand-written letters. If the data set have 100 000 examples we know 300 of them and the machine need to learn rest of them.

There are several ML methods, like neural networks, decision trees, regression algorithms. We came to them a bit later.

Data quality is an important aspect. Is the data balanced, free of major errors, is there duplicated data, are there outliers, what is the "noise level" in the data, etc. These are usually difficult questions and hard to answer before the analysis. One still need to make sure that the data quality is as good as possible.

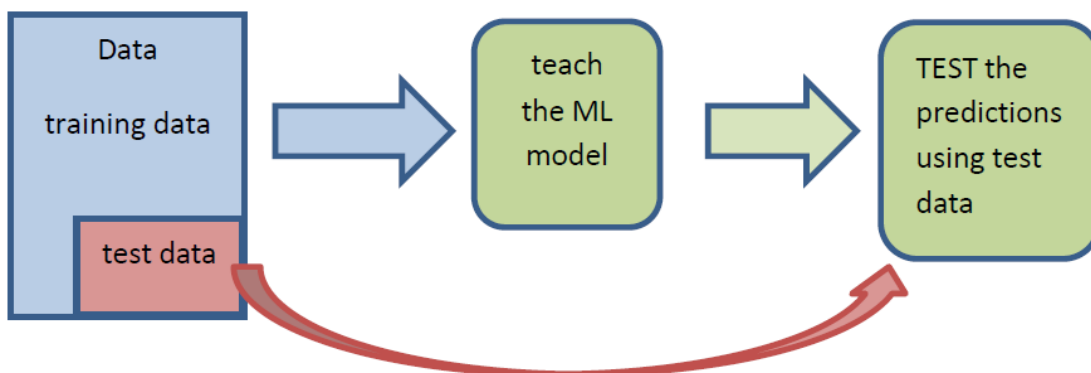
How large the data set should be? As large as possible, but in materials science the data set are usually not very large. Data size of below 100 is challenging since all ML methods rely on statistic. 1000 is OK and larger set are even better.

The quality of the data set can be tested by enlarging the data. Below blue is the training error and red is the test error. (The analysis is based in 10-fold cross validation, sorry of the low quality figure)

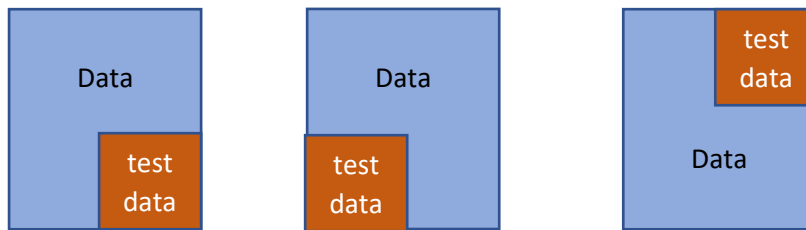


Validation

One of the main topic in ML is the method validation. To that end the original data set is divided to training and test set. The training set is used to teach the ML methods and the data in the test set is NOT use in the training. The test set size is typically 2-5 % of the data. The test set is chosen randomly to form the data. This procedure can be repeated on many data divisions.

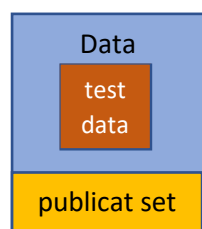


Cross validation: One can make the training/test data partitioning several times. This approach produces several ML models and test and in this way quality of the ML models can be tested better than on single data partitioning.



Each data set will give different fit the model. With cross validation we can get statistic of the fit.

One can also leave some data out of the cross validation data and use that as second level test set or **publication set**. The publication set is never used in training.



(This wiki page is very good:
[https://en.wikipedia.org/wiki/Cross-validation_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)))

ML methods parameter optimising

All ML methods contain parameter and they need to be optimized to ensure that the ML methods is working optimally. In sklearn the default parameters are quite good (if the data set is reasonably large). The teaching is simple if one is using GridSearchCV methods. There are more sophisticated methods. (for all this see the Sklearn manual, <https://scikit-learn.org/>).

```
model = RandomForestRegressor()
parameters = {"n_estimators": range(20, 80, 10), "min_samples_split": [2,3]} # for RF
clf = GridSearchCV(model, parameters, cv = 10, verbose=2)
output : score, parameters
```

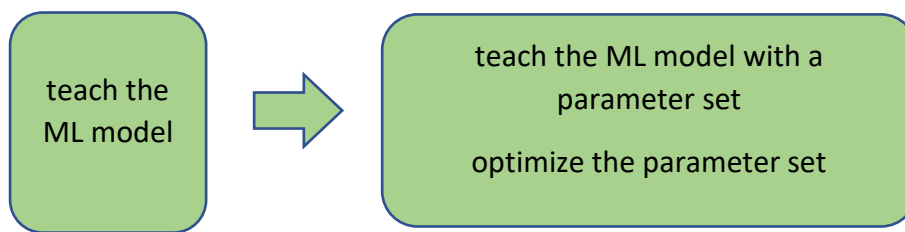
```
-0.4466248626907848 {'min_samples_split': 3, 'n_estimators': 30}
-0.45742261440125276 {'min_samples_split': 3, 'n_estimators': 50}
-0.4587929994651951 {'min_samples_split': 3, 'n_estimators': 60}
-0.4597841296896924 {'min_samples_split': 3, 'n_estimators': 40}
-0.4648287622992993 {'min_samples_split': 2, 'n_estimators': 40}
-0.4660148003169513 {'min_samples_split': 3, 'n_estimators': 70}
-0.4662565949899477 {'min_samples_split': 2, 'n_estimators': 60}
-0.4711060835686439 {'min_samples_split': 2, 'n_estimators': 50}
```

or

```
model = GradientBoostingRegressor()
```

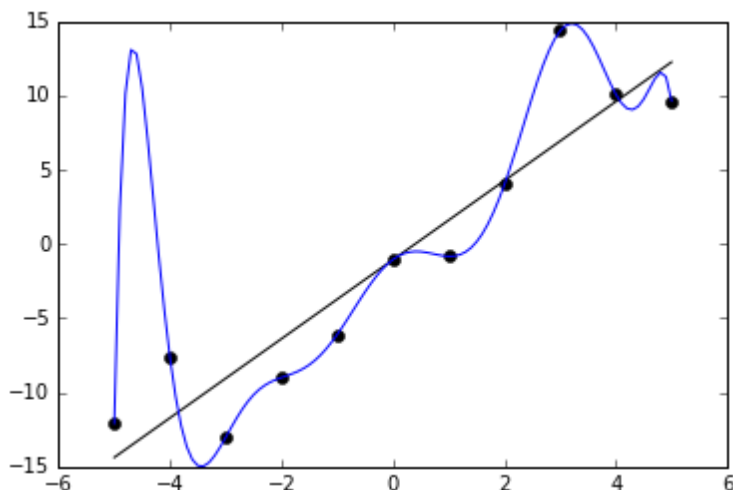
```
parameters = {'learning_rate': np.arange(0.05, 0.3, 0.05), "loss": ['ls', 'huber'],
              "n_estimators": range(20, 80, 10), 'subsample': [1.0, 0.9]}
```

```
clf = GridSearchCV(model, parameters, cv = 10, verbose=2)
```

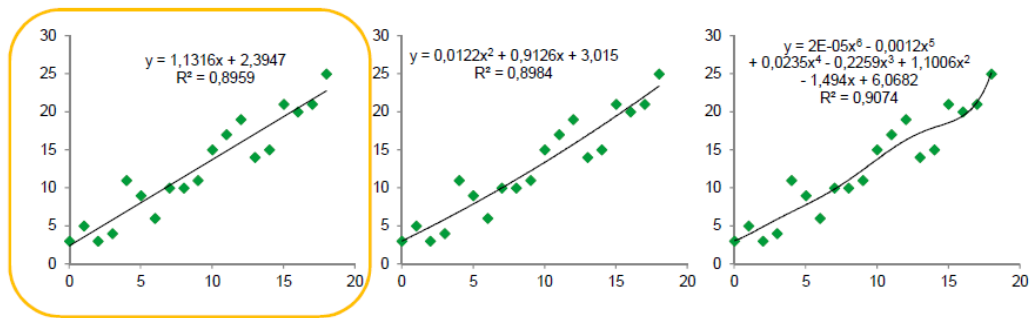


Overfitting

In every complex model there is a risk of overfitting. This is easy to demonstrate with polynomial fitting. If a N-order polynome is fitted to N data points it will fit perfectly to the points but in between the data can be very bad. If we have test set of points we can easily see the overfitting.



Another example:



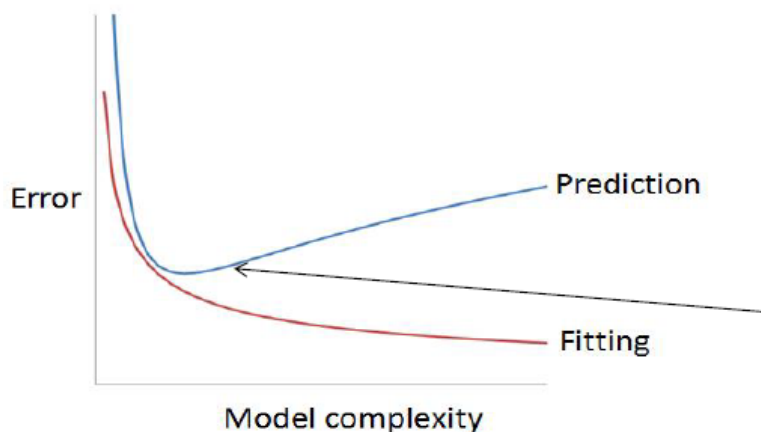
Split No.	1st order	2nd order	6th order
1	2.138	2.033	1.758
2	2.555	2.565	7.503
3	2.674	2.617	2.756
4	1.721	1.868	140.031
5	2.863	3.031	3.180
Mean validation error	2.39	2.42	31.05

However, expected error of our best model is not 2.39.

We would need a third set that contains examples that were not in original input set.

The third set is called the *test set* or sometimes *the publication set*.

The best model is not the model that fit best to the data but that have the best predictive power.



- The best model is characterized by minimal prediction error

The example to order-N polynome is trivial but the overfitting is a real problem in **every ML model**. Naturally the model need to be good enough, so one can also underfit the problem. To find good balance a lot of testing is needed.

Machine learning methods

There are several ML methods. Many of them have been implemented to sklearn python package.

Methods for labeled data (we know the data objects, like Pt(111) surface or PtAg mixture. This sounds trivial but if we have pictures and we need to know what is in them (a cat or a

car or a human) the situation is more difficult. The labeled data is considered expensive since it need humans to make the labelling.)

- **Regression algorithms:** Linear and logistic regression are examples of regression algorithms used to understand relationships in data. Linear regression is familiar to all scientists. More sophisticated regression algorithm called a support vector machine.

- **Decision trees:** Decision trees use classified data to make recommendations based on a set of decision rules. For example, a decision tree that recommends betting on a particular horse to win, place, or show could use data about the horse (e.g., age, winning percentage, pedigree) and apply rules to those factors to recommend an action or decision.

We have used a lot the RandomForest method

- **Instance-based algorithms:** A good example of an instance-based algorithm is K-Nearest Neighbor or k-nn. It uses classification to estimate how likely a data point is to be a member of one group or another based on its proximity to other data points.

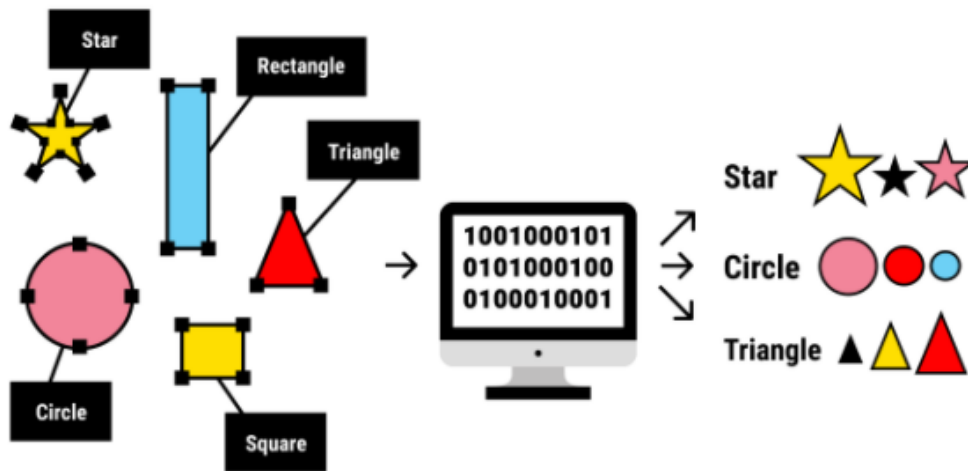
Methods for unlabeled data (opposite the labeled data, we do not need to know the object. Very often the ML task is to identify them.)

- **Clustering algorithms:** Think of clusters as groups. Clustering focuses on identifying groups of similar records and labeling the records according to the group to which they belong. This is done without prior knowledge about the groups and their characteristics. Types of clustering algorithms include the K-means, TwoStep, and Kohonen clustering.

- **Association algorithms:** Association algorithms find patterns and relationships in data and identify frequent 'if-then' relationships called *association rules*. These are similar to the rules used in data mining.

- **Neural networks:** A neural network is an algorithm that defines a layered network of calculations featuring an input layer, where data is ingested; at least one hidden layer, where calculations are performed make different conclusions about input; and an output layer, where each conclusion is assigned a probability. A deep neural network defines a network with multiple hidden layers, each of which successively refines the results of the previous layer.

Often the labeled data is needed (or it is very useful) for teaching the unlabeled algorithms.

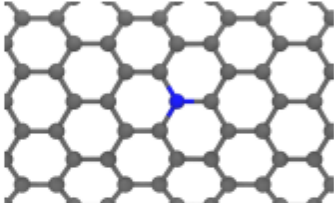
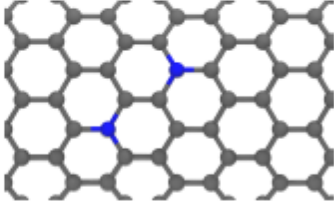
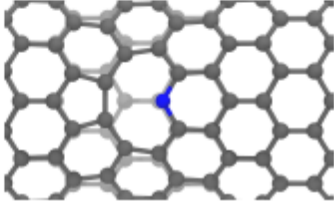


Descriptors

Descriptors are very important in materials science. We should know the geometry and other properties of the material or molecules but how we will tell that to a machine. The descriptors can be almost anything.

We did recently a study of HER (hydrogen evolution reaction) on N doped carbon nanotubes taking into account several defects. Overall, there was 8 different defects and several hydrogen configurations. Totally we did ca. 7000 DFT calculations. The output was the hydrogen binding energy. (Kronberg, Lappalainen, Laasonen, JPCC, 125, 15918 (2021)). In this project we used the Random Forest method and a very new Shapley analysis of the data.

Table 1: Specification and illustration of the studied model NCNT systems. The abbreviations V_1 , V_2 and SW denote a single vacancy, double vacancy and Stone–Wales rotation, respectively. For the Stone–Wales defect two distinct nitrogen positions resembling indole (N_{1a}) and indolizine (N_{1b}) structures were considered.

N moiety	Defect	(n, m)	Image
Graphitic, N_1	None	$(14, 0)$ $(8, 8)$	
Graphitic, N_2	None	$(14, 0)$ $(8, 8)$	
Pyridinic, N_1	V_1	$(14, 0)$ $(8, 8)$	

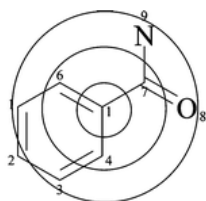
This project had rather complex descriptors. This example is not the easiest one, but it illustrates that the very different descriptors can be used.

Table 2: Mathematical formulation and explanation of all 25 input features used in the RF models. Note that all features except those inferring to adsorption-induced changes are calculated on the reference configurations, i.e. the NCNT structures before adsorption (NCNT + (n - 1)H). A glossary of auxiliary variables used in defining each feature is presented in Table S1 in the Supporting Information.

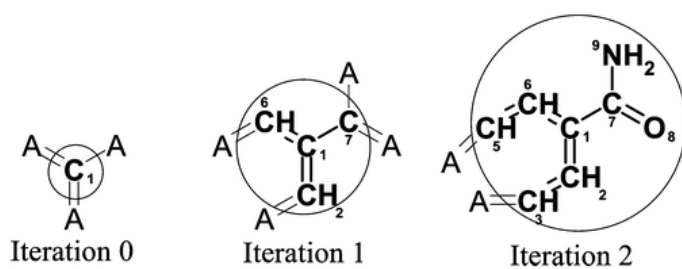
Feature	Definition	Unit	Description
x_k	N_k/N_{NCNT}	at%	Atomic concentration of $k = \text{N}, \text{V}$ or H^a
$\min d_k$	$\min_k \sqrt{r^2 \theta_k^2 + z_k^2}$	Å	Shortest curvilinear distance along NCNT surface between S and $k = \text{N}$ or H^b
$\min l$	$\min_k \mathbf{R}_S - \mathbf{R}_k $	Å	Shortest S to $k \in \text{NN}$ bond length ^c
$\max l$	$\max_k \mathbf{R}_S - \mathbf{R}_k $	Å	Longest S to $k \in \text{NN}$ bond length
RMSD	$\sqrt{\langle (\Delta \mathbf{R}_k)^2 \rangle}$	Å	Adsorption-induced root-mean-squared displacement of atomic positions
RmaxSD	$\sqrt{\max_k (\Delta \mathbf{R}_k)^2}$	Å	Adsorption-induced root-maximum-squared displacement of atomic position
χ	$\arctan\left(\frac{\sqrt{3}m}{2n+m}\right)$	rad	Chiral angle of (n, m) NCNT
$\min \varphi_k$	$\min_{j \neq i} \arccos(\hat{\mathbf{u}}_{ik} \cdot \hat{\mathbf{u}}_{jk})$	rad	Smallest angle where $k = \text{S}$ or nearest N defines the vertex and $i, j \in \text{NN}$ of k
$\max \varphi_k$	$\max_{j \neq i} \arccos(\hat{\mathbf{u}}_{ik} \cdot \hat{\mathbf{u}}_{jk})$	rad	Largest angle where $k = \text{S}$ or nearest N defines the vertex and $i, j \in \text{NN}$ of k
α_k	$\arccos(\hat{\mathbf{z}} \cdot \hat{\mathbf{v}}_k)$	rad	Angular displacement of S with respect to the nearest $k = \text{N}$ or H
$\overline{\text{CN}}_k$	$\sum_i \frac{\text{CN}_i}{\text{CN}_{i,\max}}$	–	Generalized coordination number of $k = \text{S}$ or nearest N with $i \in \text{NN}$
$\Delta \overline{\text{CN}}_k$		–	Adsorption-induced change in $\overline{\text{CN}}_k$
Z		–	Atomic number of the adsorption site
M	$2S + 1$	–	Spin multiplicity of the system
q	$n_{\text{val}} - (n_{\uparrow} + n_{\downarrow})$	e	Residual charge on the adsorption site (iterative Hirshfeld partitioning)
μ	$n_{\uparrow} - n_{\downarrow}$	e	Spin polarization on the adsorption site (iterative Hirshfeld partitioning)
E_g	$E_{\text{LUMO}} - E_{\text{HOMO}}$	eV	Energy gap, for open-shell systems the SOMO-LUMO gap

^aN, V, H = Nitrogen, vacancy, hydrogen; ^bS = Adsorption site; ^cNN = Nearest neighbor sites;

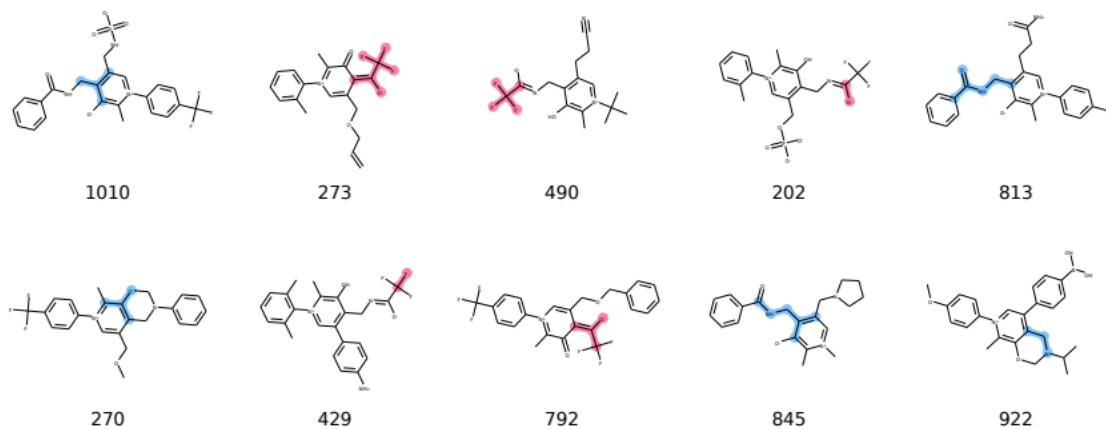
One interesting descriptor is Extended-connectivity fingerprint (ECFP). It is a systematic tool that list atoms environment in molecules. (Ref: Rogers and Hahn, *J. Chem. Inf. Model.* 2010, 50, 5, 742–754). The 0 level is the atom itself, the level 1 contains the atoms neighbors and so on.



Considering atom 1 in benzoic acid amide



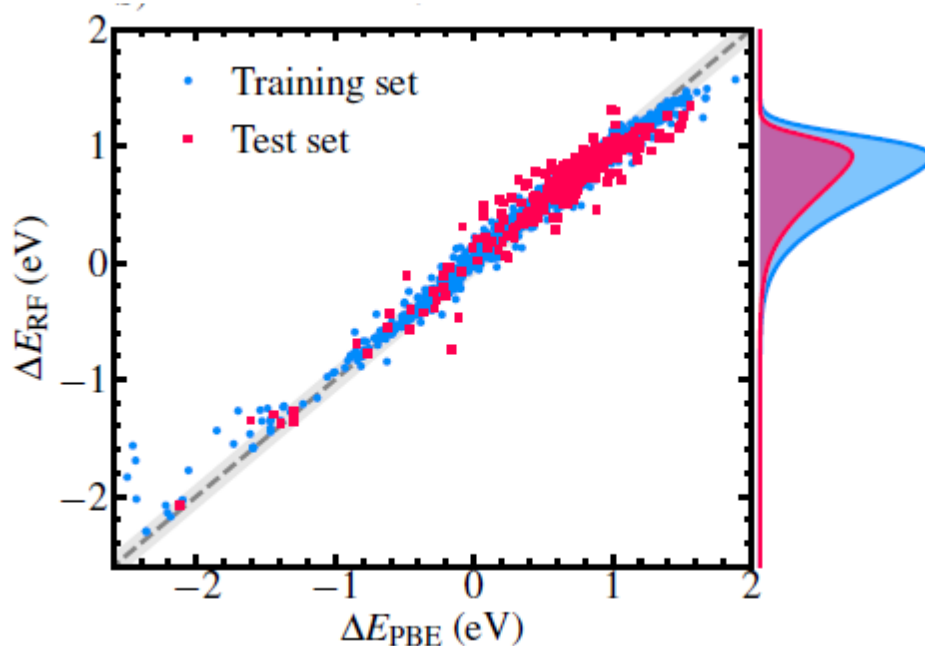
Next one can list all the different ECFP's of all the studied molecules. There are quite a few of them but surprisingly few. We did a project in which there was 7000 different molecules and we found 1025 ECFP4's



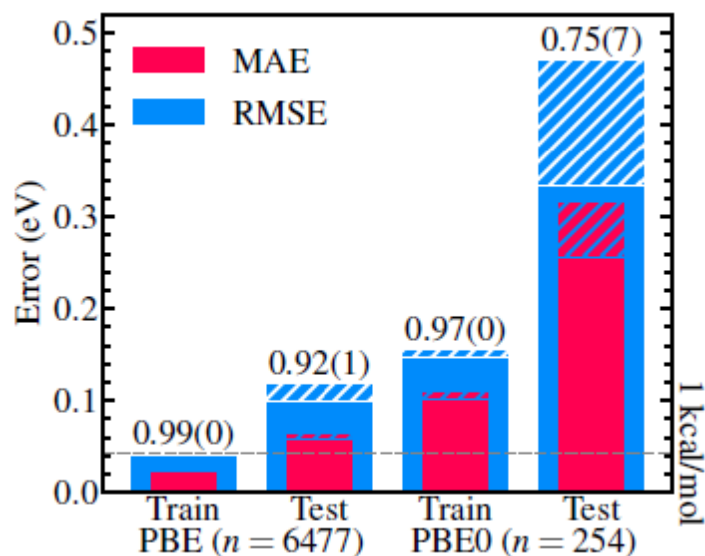
In this case the descriptor was the presence of the ECFP4. Mathematically a vector of 0's and 1's with length 1025.

Results

Naturally the RF model learned the data well. The parity plot compares the computed (DFT) values to the ML predictions.



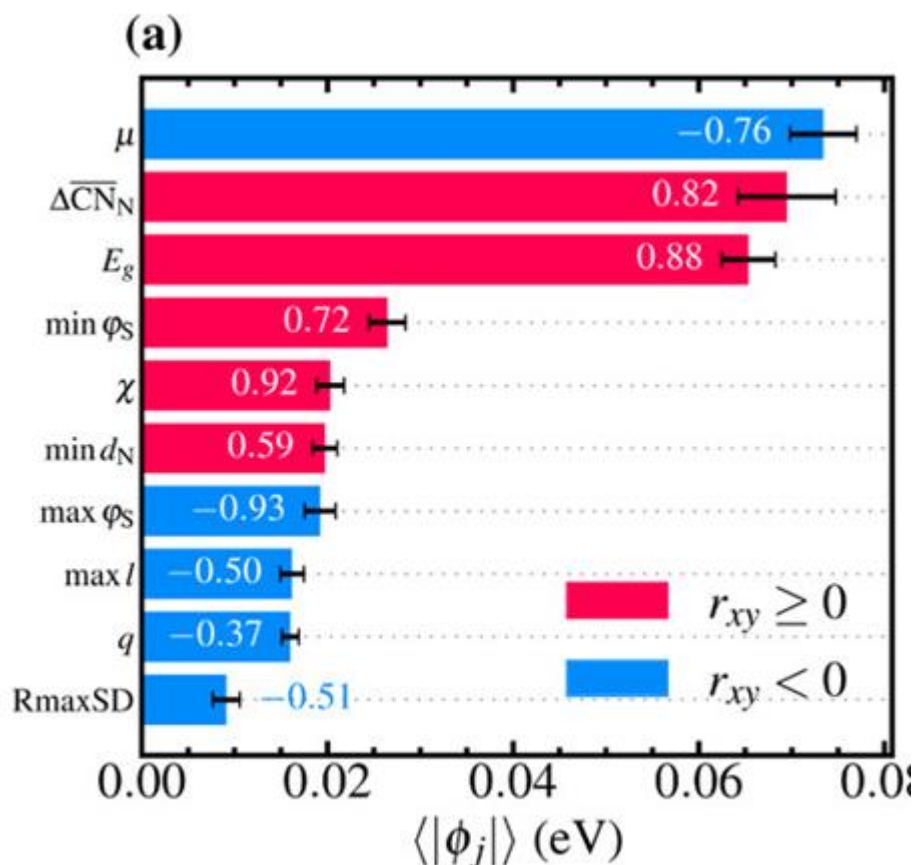
As one can see, where there is a lot of data the learning is good and at the very negative values the scattering is larger. The accuracy of the trained data is below kcal/mol, which is better than the DFT accuracy. One can also see the effect of the size of the sample. We did some PBE0 calculations. Here the data set is much smaller and the learning errors are



larger.

Figure 3: Unbiased generalization performance of the RF models based on 10_5-fold nested CV on the GGA and hybrid HF/DFT datasets. The solid bars denote a lower bound of the respective averaged errors while the hatched parts indicate the variability as twice the standard deviation across the outer CV folds. The average coefficients of determination with standard deviations are annotated above the bars. The limit of chemical accuracy is marked for reference by the dashed line.

The next deep question is how the descriptors contribute to the output. This is usually addressed on rather superficial way. Typically, the methods like RF will return the weight of the descriptors. This is useful if some of the descriptors have low weights. Then one can reduce the descriptors and still get quite good predictions with less descriptors.



Explainable AI, the Shapley analysis

Rather recently a very interesting Shapley additive explanation (SHAP) methods has been introduced. It will approximate the model output with additive functions ϕ , Shapley values. The ML predicted value f can be written as

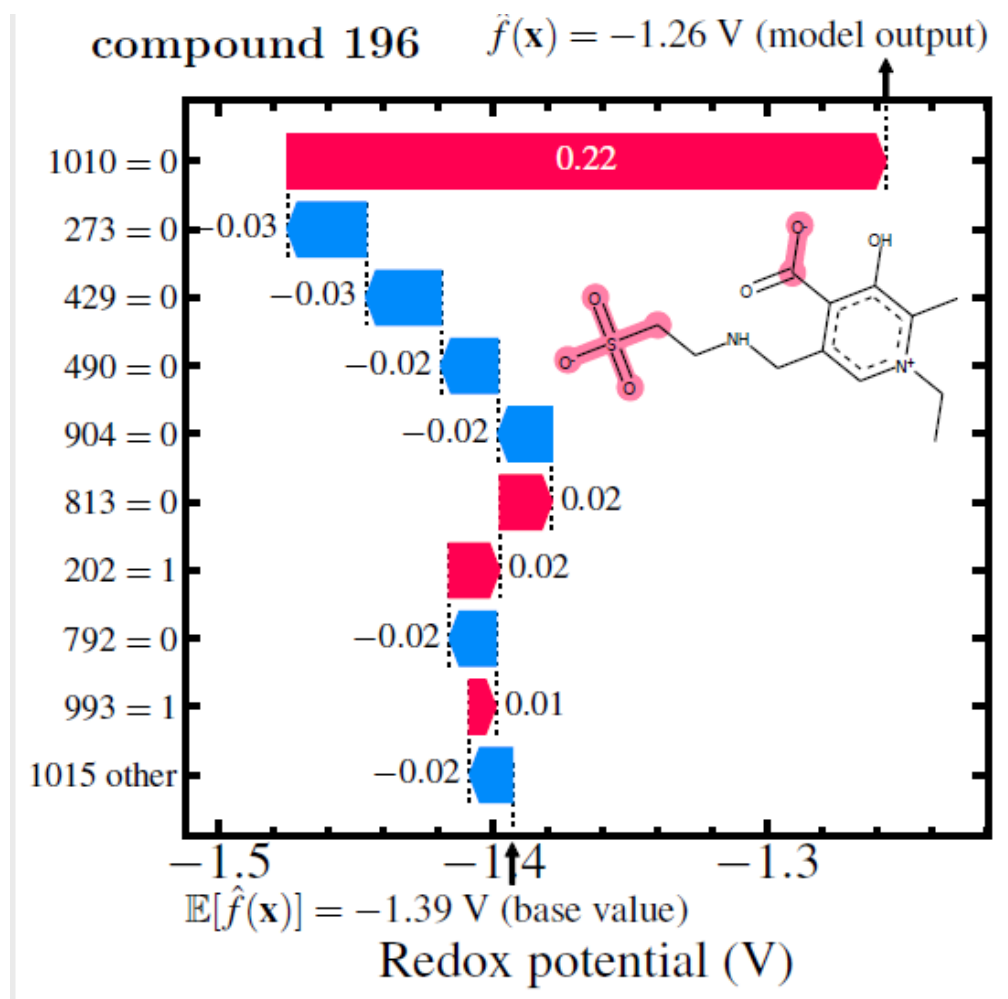
$$f(x) = \langle f \rangle + \sum_j \phi_j(f, x)$$

where $\langle \rangle$ is the average of f and x are the descriptors. Even this looks very simple the computation of the Shapley values is complicated. The brake-through publication is from 2017

(Lundberg, S. M.; Lee, S.-I. A unified approach to interpreting model predictions. Adv. Neural Inf. Process Syst. 2017; pp 4765–4774.)

The SHAP analysis gives much more information of the ML procedure. We can analyse the individual descriptor contribution to the output. If we have chemically meaningful descriptors we can learn a lot more form the results. Below is an example of the molecules redox potential prediction. The numbers refer to the ECFP4 features in the molecules (0 means that they are not present and 1 that they are). Note that the

1015 lowest weight descriptors have very small contribution and the descriptor 1010 has very large contribution.



The SHAP analysis has results to a new subfield of ML, the explainable artificial intelligence (XAI). There are several problems where it is very useful to understand where the ML predictions come from. Clearly materials development projects belong to this class.

Where we can get the data for ML projects

In chemical and material science problems we have some large experimental databases (DB), like the crystal structure DB's but for many properties we do not have large DB's. Individual values can be found from the literature but if we need thousands of numbers large scale DFT computations are a promising approach. The experimental data from various sources can contain errors whereas if the DFT computations are done systematically the data is of good quality. Of course, the DFT is not perfect but for ML we need trends and large data sets. This is the reason why most chemistry and materials science ML projects are based on DFT calculations.

Because the DFT results are so useful (for ML) there are also DB's for the DFT results, like NOMAD. A good review of the Databases is *Himanen et al. Adv. Sci.* **2019**, *6*, 1900808, DOI: 10.1002/adv.201900808

NOMAD: Provides storage for full input and output files of all important computational materials science codes, with multiple big-data services built on top. Contains over 50 236 539 total energy calculations.

Warning the databases are not always easy to use and the data quality can be quite poor. We did a M.Sc. study of chemical reactions using DFT DB's and the results were not very good. We are in the beginning of the DFT DB's and the rules what one needs to store to the DB's does not exist. It also seems that the data in the DB's are not checked very carefully. I hope that the quality DB's will improve in the future. Naturally this criticism does not apply to all databases.

Data quality

Remember: GARBAGE IN GARBAGE OUT

Data quality is essential to ML. Wherever you get the data one should be skeptical of its quality.

Are there some chemical bonds the data should fill?
In large databases are there duplicated data.
When the ML parity plot is done are there some outliers in the data. They can be due to the poor ML model OR from poor input

data. When doing 1000's of DFT calculations, are all the results converged? If using external DB's how you know the data quality.

Predictability

The predictability is one of the hardest questions in ML. We can easily analyse the predictability of the data set we have but what happen if we go outside the data set. If the new molecules (or materials) are similar we can expect reasonable predictions. But what is "similar"?

The larger and more diverge the learning DB is the more we can predict. We need tool to analyse the divergence of the DB's then we can have some information what can be predicted.