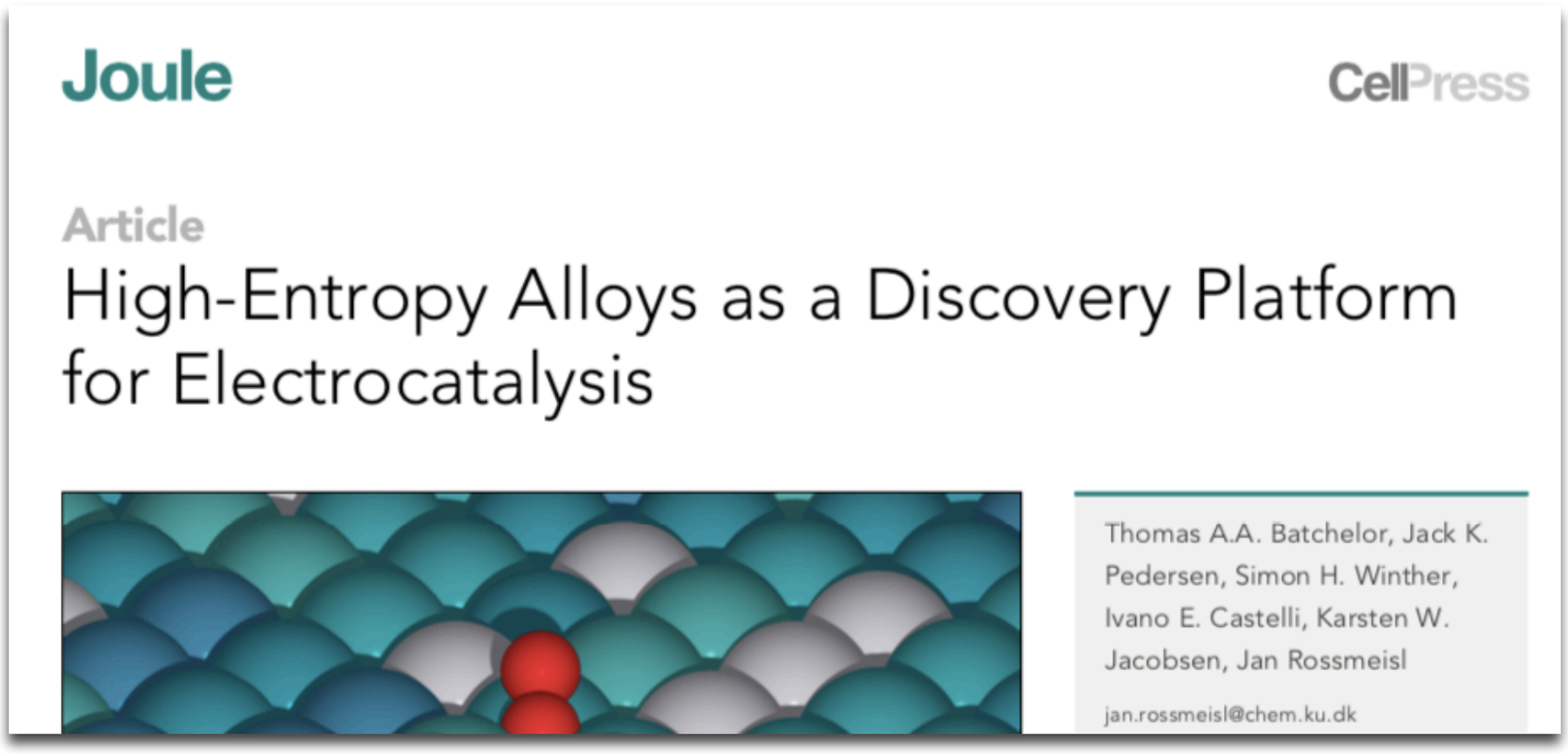
Using machine learning to explore High-Entropy Alloys

Contributed to and agreed on by Jack K. Pedersen, Hao Wan, Christian M. Clausen

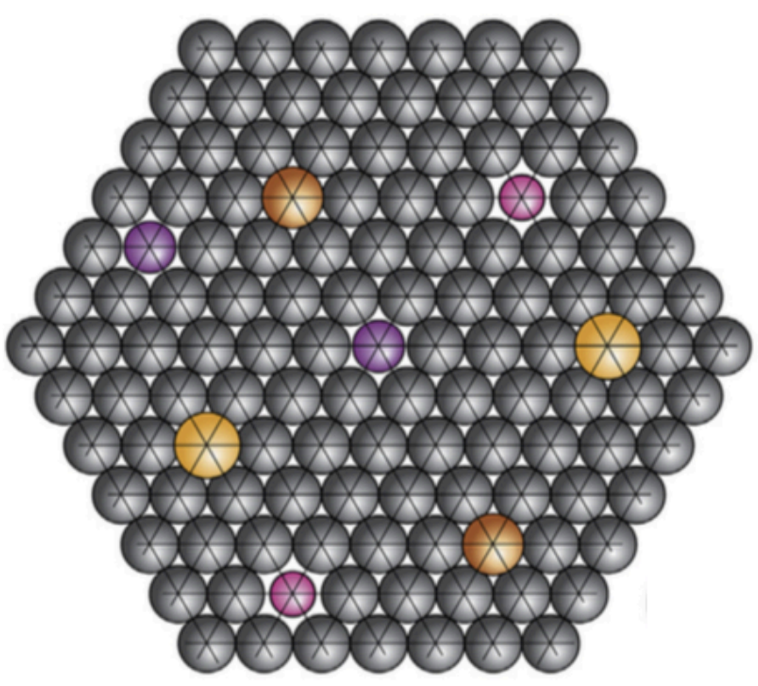
June 12th 2019

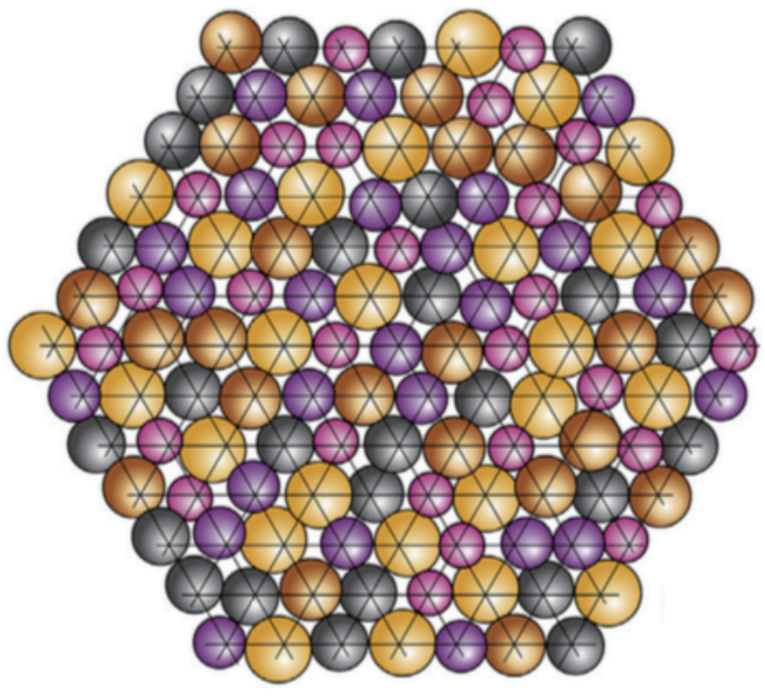


UNIVERSITY OF COPENHAGEN
DEPARTMENT OF CHEMISTRY

Complex solid state solutions / High-Entropy Alloys

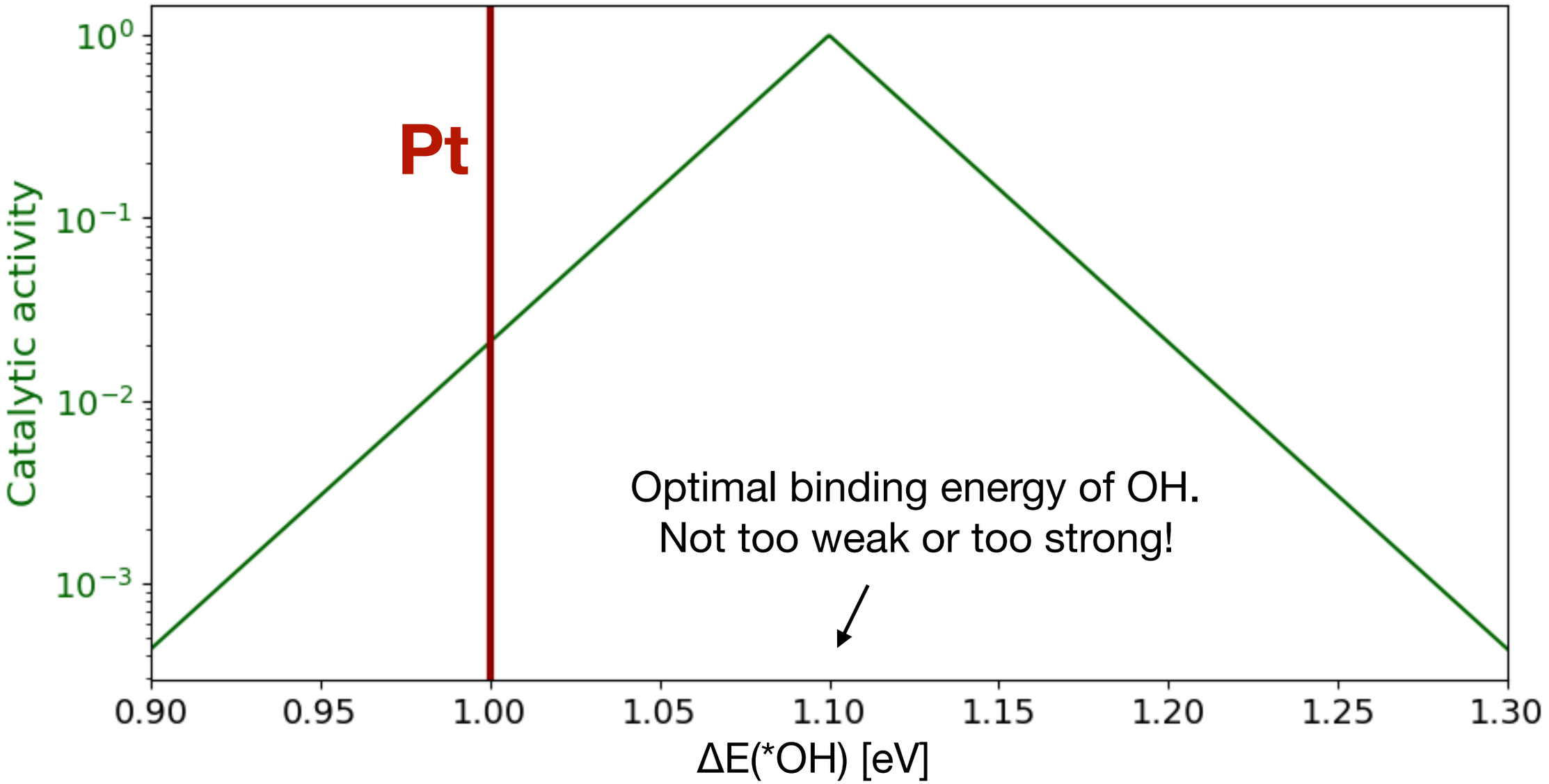
- Alloys have traditionally consisted of a single matrix element with minor fractions of other elements.
- From 2004 and onwards alloys containing equal fractions of multiple elements became a popular research topic.





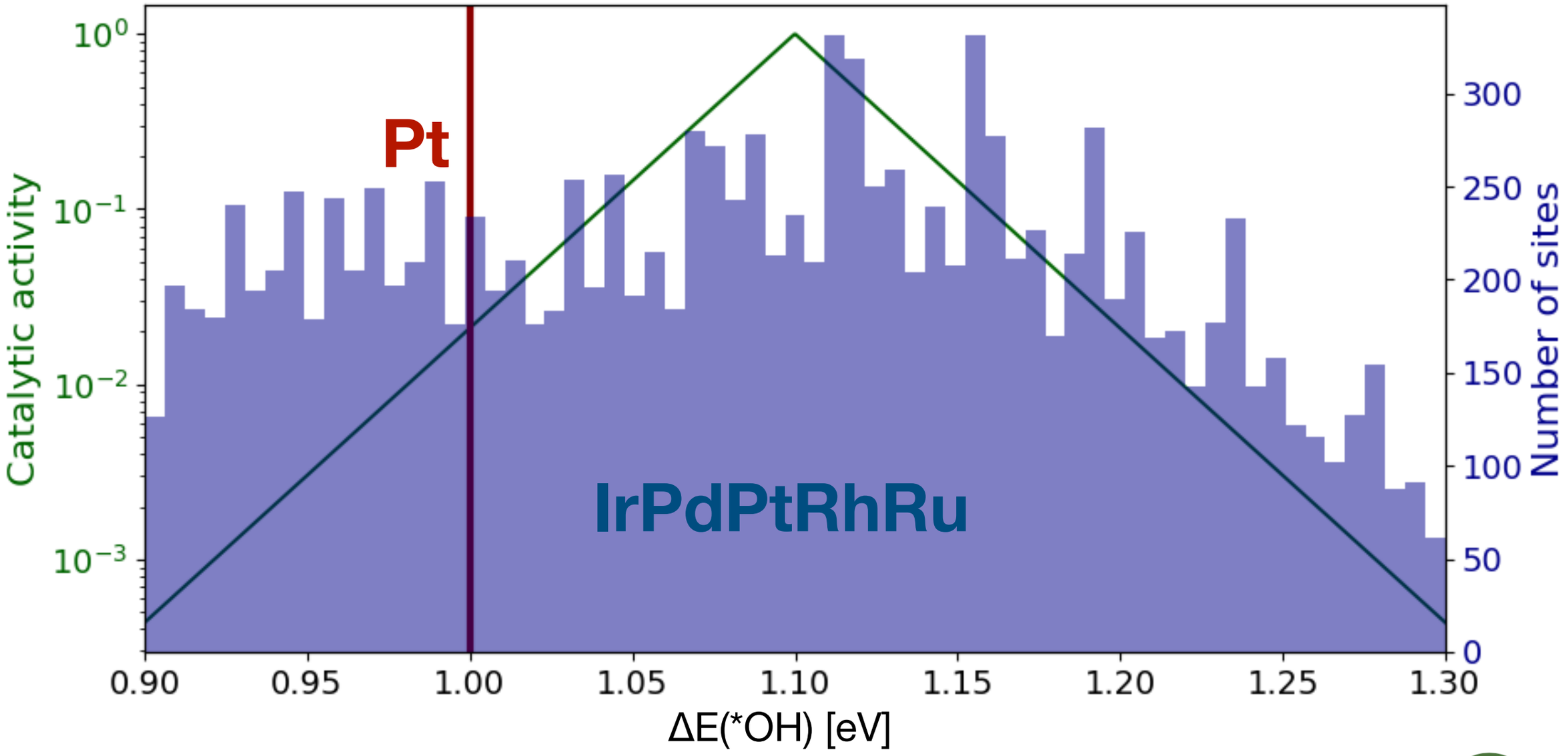
Miracle et al. Acta Materialia, 2016

Heterogenous catalysis - Oxygen Reduction Reaction



$$O_2 + 4\,H^+ + 4\,e^- \longrightarrow 2\,H_2O$$

Heterogenous catalysis - Oxygen Reduction Reaction



$$O_2 + 4\,H^+ + 4\,e^- \longrightarrow 2\,H_2O$$

UNIVERSITY OF COPENHAGEN
DEPARTMENT OF CHEMISTRY

Exploring the hyperdimensional alloy space

Calculate sample of adsorption energies

↓

Train machine learning model

↓

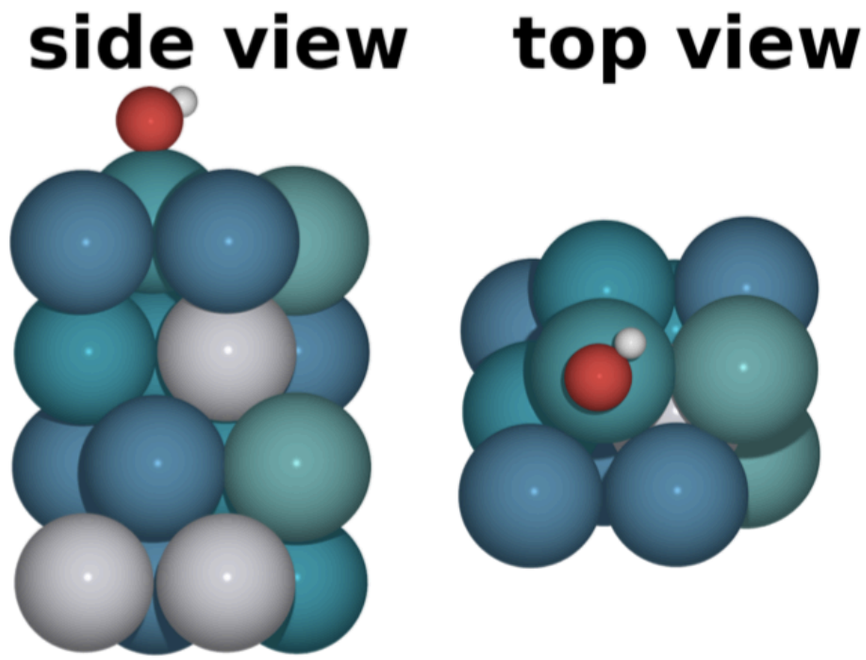
Estimate adsorption energy of all possible sites

↓

Optimize the alloy composition

Training set

- Binding energy of OH calculated with quantum mechanical methods (half an hour on 20 cores).
- 3888 metal ‘slabs’ of 2x2x4 atoms
- Pd or Pt as central atom.
- Randomized composition from uniform distribution of Ir, Pd, Pt, Rh and Ru.
- Due to limited data a dedicated test set will not be created and cross-validation results will be used for evaluation.



$$\Delta E_{bind} = E_{slab+OH} + \frac{1}{2}E_{H_2} - E_{slab} - E_{H_2O}$$

Training set

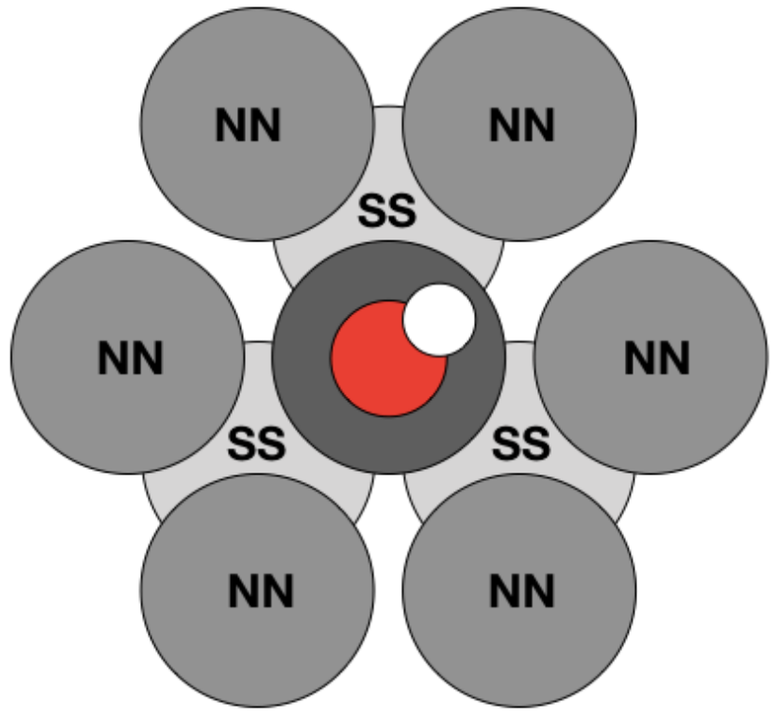
4. layer				3. layer				2. layer				Top layer				ΔE_{bind}
Pd	Pd	Pt	Ru	Pt	Rh	Ru	Pt	Ir	Pt	Rh	Rh	Ir	Ir	Pt	Rh	1.0350 eV
Ru	Ru	Rh	Pt	Pd	Pd	Pd	Pt	Pd	Rh	Ir	Pd	Ir	Ru	Pt	Pd	0.9407 eV
Rh	Pt	Pd	Pt	Ru	Rh	Ir	Ru	Rh	Pd	Pt	Ir	Ru	Ru	Pd	Ru	1.1498 eV
Ir	Ir	Rh	Pt	Rh	Ir	Pt	Pd	Rh	Pt	Ru	Ru	Ru	Rh	Pt	Pt	0.9984 eV
Ir	Ir	Pt	Rh	Pd	Pt	Rh	Ru	Rh	Rh	Pd	Ru	Pt	Rh	Pd	Ir	1.2047 eV
Pd	Ir	Pt	Pt	Pd	Ir	Pd	Ir	Pt	Pd	Pd	Pd	Ru	Pt	Pt	Ir	1.0272 eV
Rh	Ir	Pd	Ru	Pt	Pd	Pd	Pt	Ir	Pt	Ir	Ru	Rh	Pd	Pt	Pd	0.7557 eV
Ir	Pt	Pt	Ir	Ru	Pt	Ir	Pt	Ru	Pd	Rh	Ir	Pt	Pd	Pt	Ir	1.0852 eV
Ir	Ir	Rh	Pt	Pt	Pt	Pd	Rh	Ru	Rh	Pd	Ir	Rh	Ir	Pt	Pd	0.9986 eV
Ru	Pt	Ir	Rh	Rh	Pt	Ru	Ir	Rh	Ir	Ir	Ru	Pd	Rh	Pt	Rh	0.7758 eV

+ 3878 more lines

UNIVERSITY OF COPENHAGEN
DEPARTMENT OF CHEMISTRY

Categorical feature encoding *How do we turn elements into numbers?*

- OneHot scheme: [1,0,0,0,0] x 16 atoms
- Dummy scheme [0,0,0,0] x 16 atoms
- Hash scheme [index] x 16 atoms
- Zoned scheme (see figure)

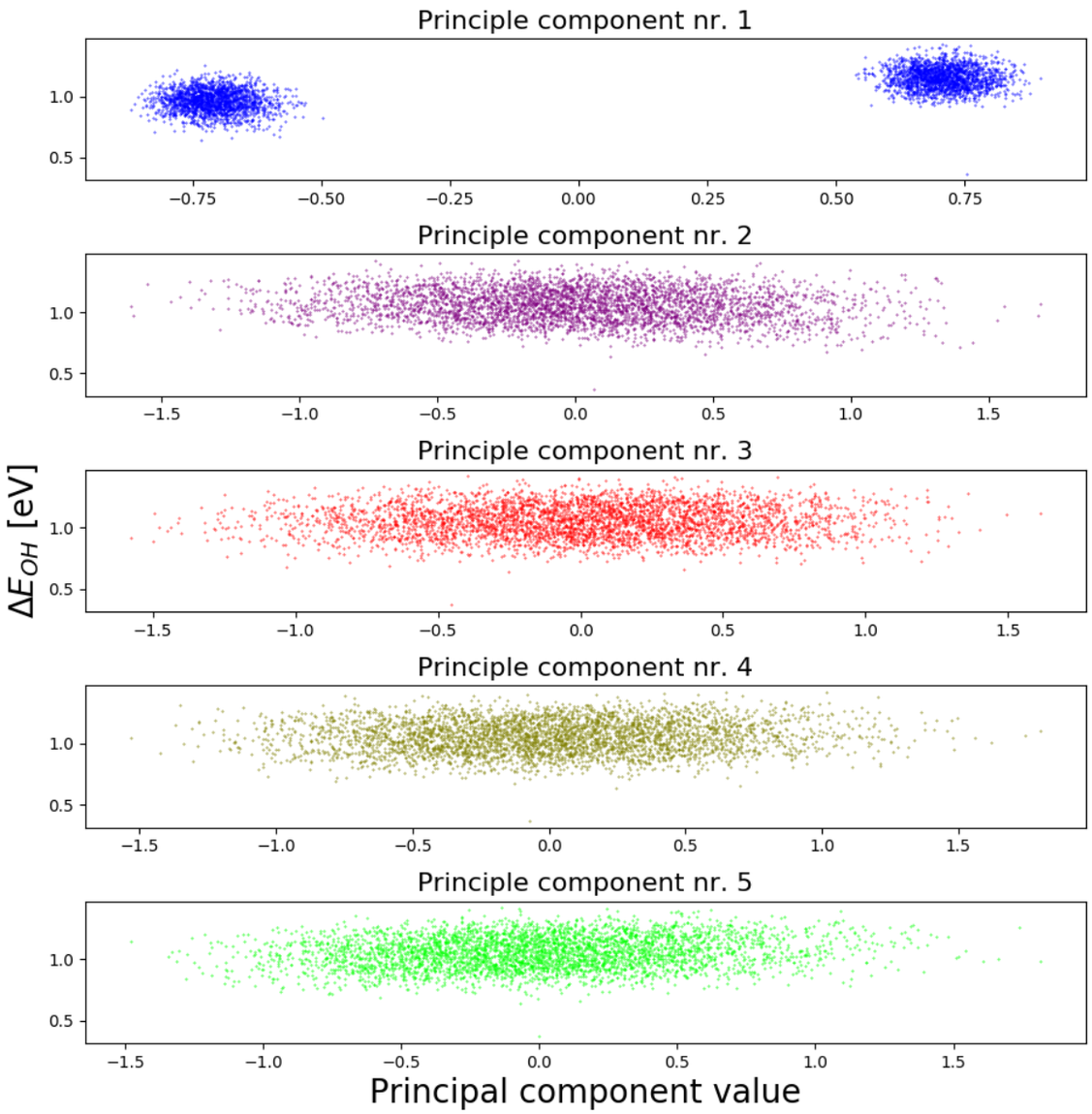


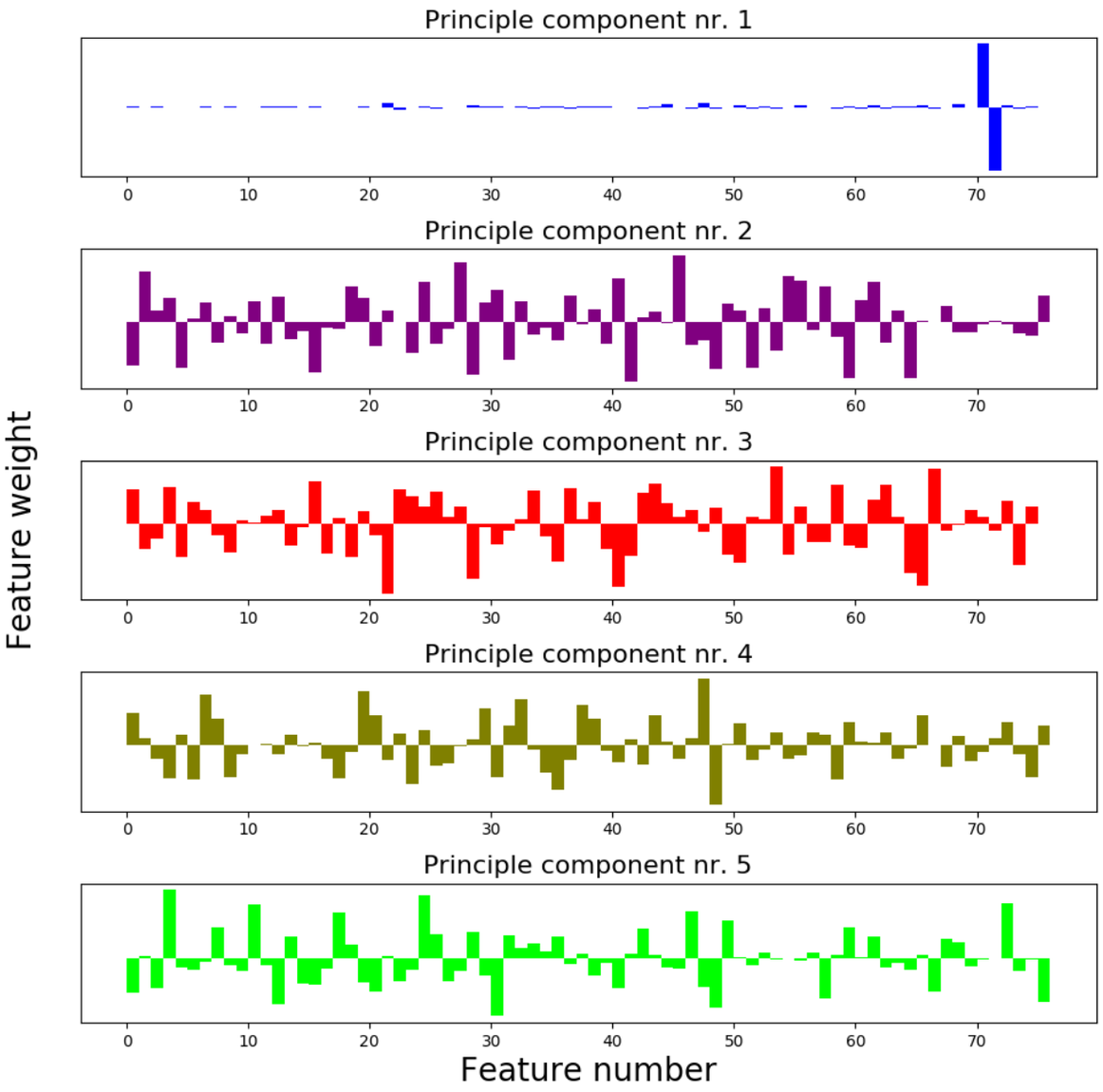
- Mean Absolute Error from 5-fold cross-validation:

MAE-table	OneHot	Dummy	Hash	Zoned
Linear Model	0.0271 eV	0.0270 eV	0.1057 eV	0.0424 eV
XGBoost	0.0256 eV	0.0306 eV	0.1060 eV	0.0399 eV

OneHot-encoding seems like the best compromise to convey most information…

Principle Component Analysis *Where is the information contained?*





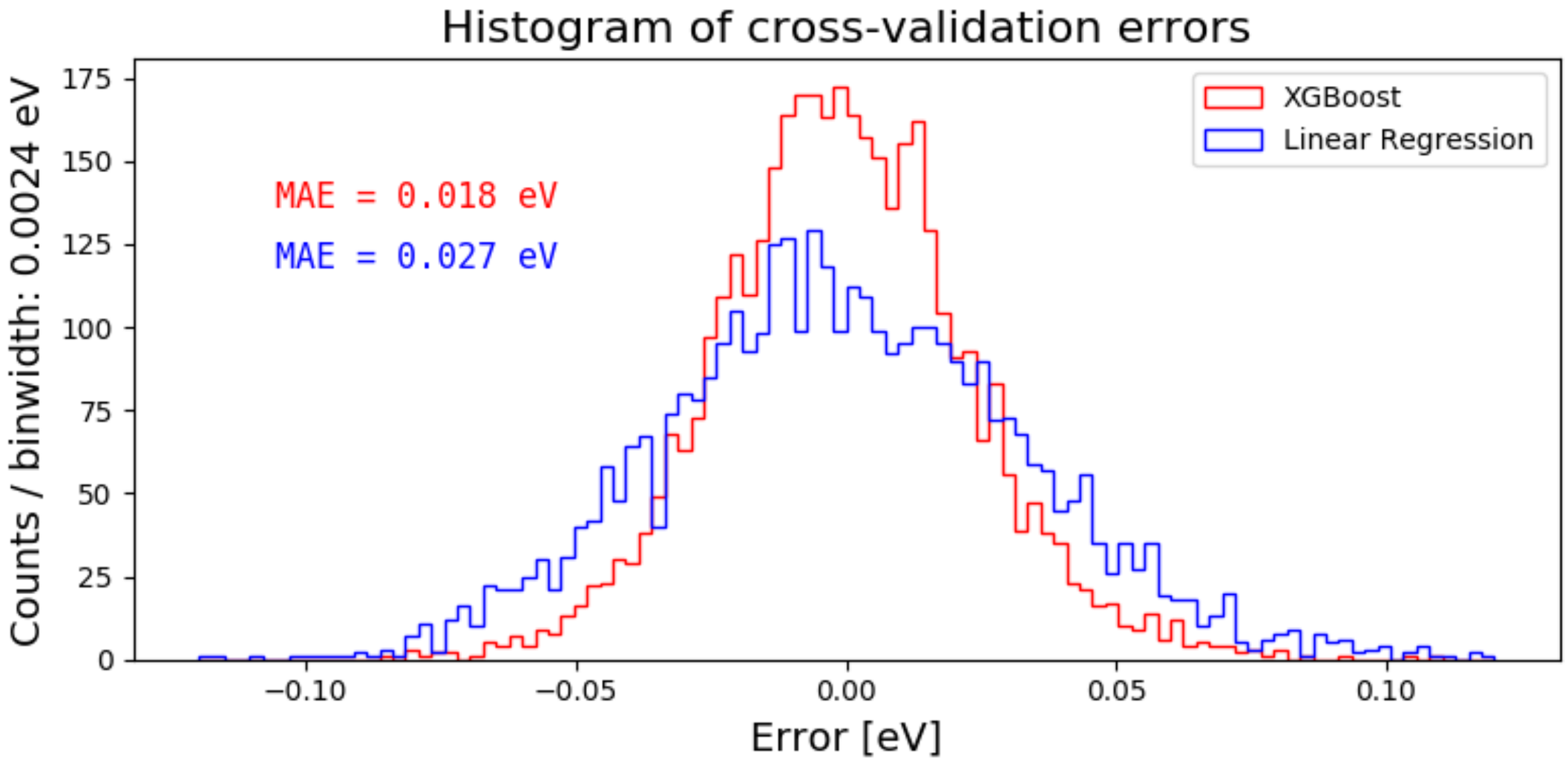
Information is apparently uniformly distributed apart from the adsorption site…

XGBoost *A decision tree model for categorical data seems logical*

- Performed with SKLearn RandomizedSearchCV for 200 random configurations.
  - ’n_estimators’ ∈ [50,500] (number of boost-iterations)
  - ’max_depth’ ∈ [1,8] (depth of tree)
  - ’learning_rate’ ∈ [0.0,0.5] (shrinkage of gradient boosting)
  - ’booster’ = ‘gbtree’ or ‘dart’ (booster-model)

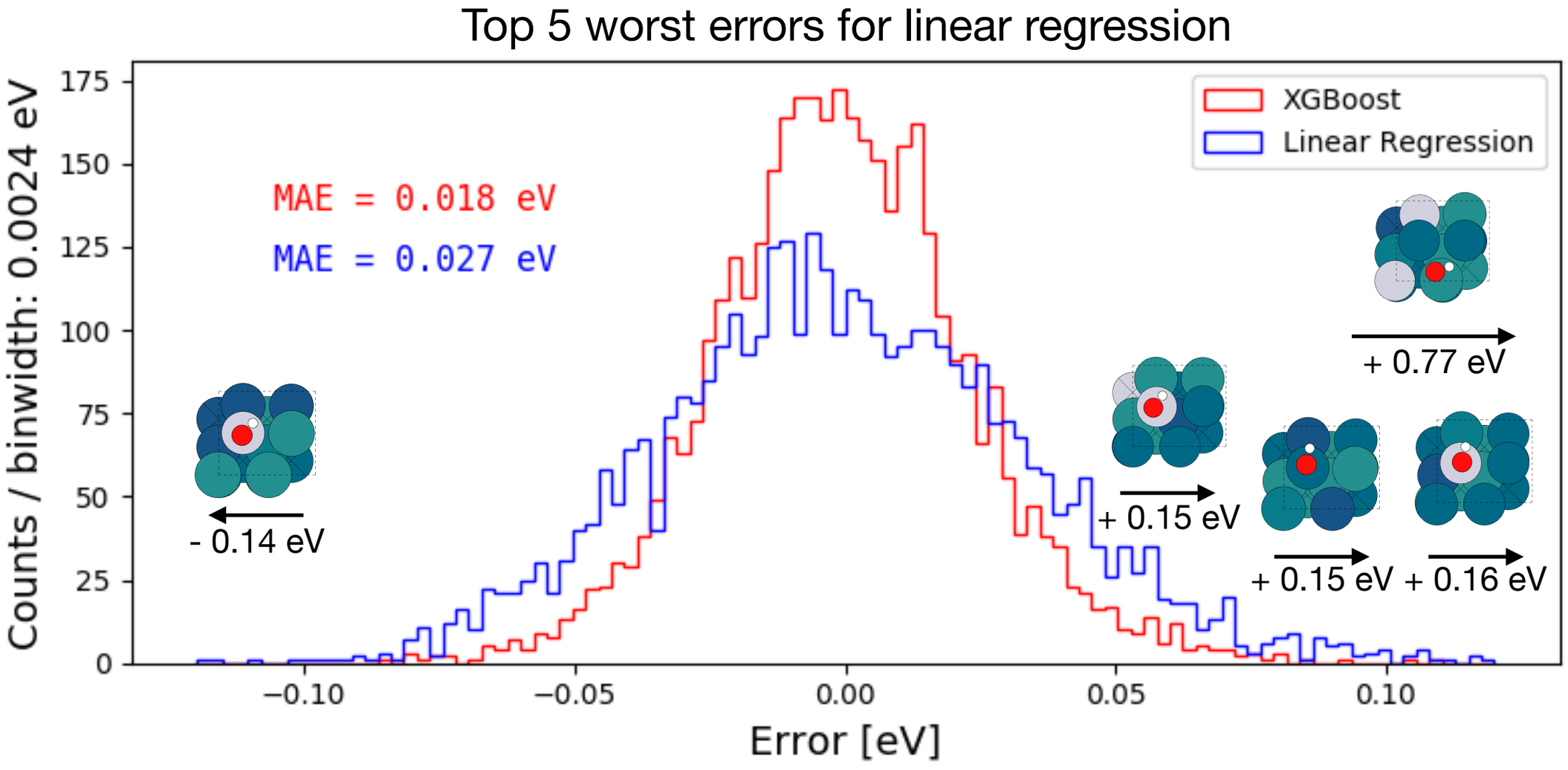
```
Rank 1 model
Mean fit time: 6.296 (std: 0.081) seconds
Mean validation score: 0.018 (std: 0.000)
Parameters: {'n_estimators': 456, 'max_depth': 3, 'learning_rate': 0.24924924924924924, 'booster': 'gbtree'}

Rank 2 model
Mean fit time: 6.745 (std: 0.042) seconds
Mean validation score: 0.019 (std: 0.000)
Parameters: {'n_estimators': 486, 'max_depth': 3, 'learning_rate': 0.2082082082082082, 'booster': 'gbtree'}

Rank 3 model
Mean fit time: 6.520 (std: 0.070) seconds
Mean validation score: 0.019 (std: 0.000)
Parameters: {'n_estimators': 482, 'max_depth': 3, 'learning_rate': 0.1906906906906907, 'booster': 'gbtree'}
```

Linear regression vs. XGBoost *What causes errors in the predictions?*



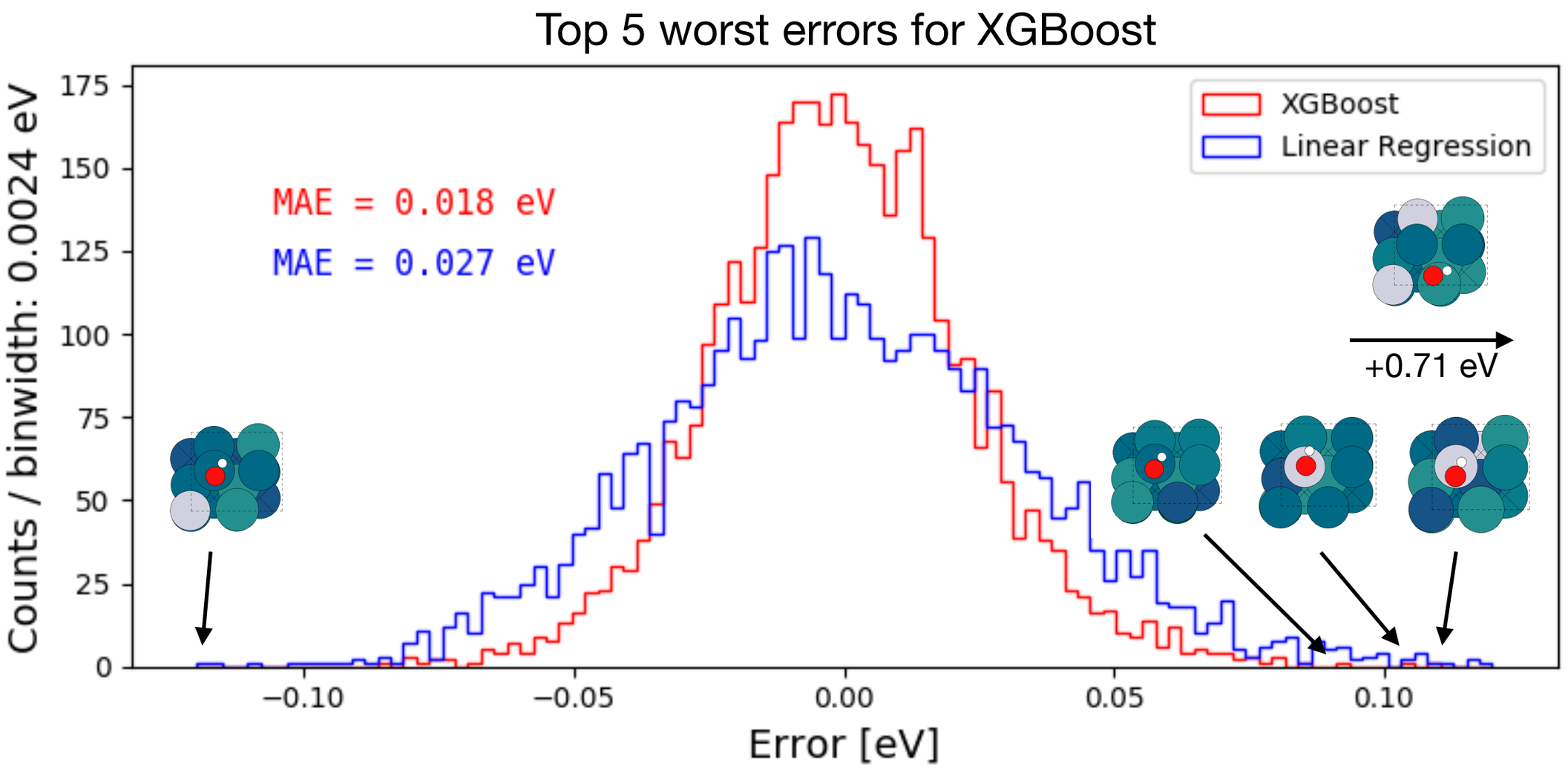
Going from linear regression to XGBoost reduces the error by a factor of 2/3…

Linear regression vs. XGBoost *What causes errors in the predictions?*



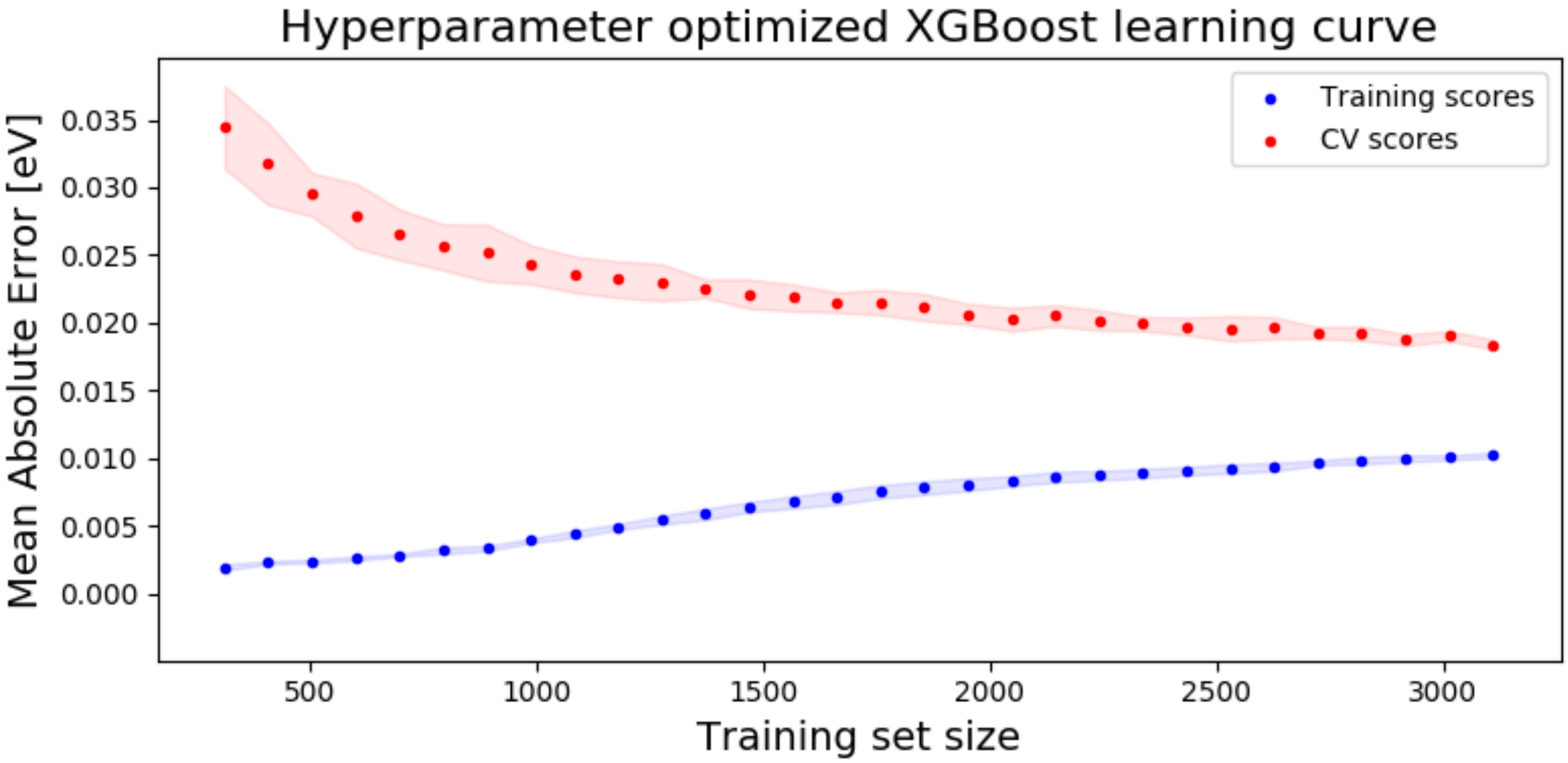
Apart from an obvious outlier there seems to be no clear pattern to the worst errors…

Linear regression vs. XGBoost *What causes errors in the predictions?*



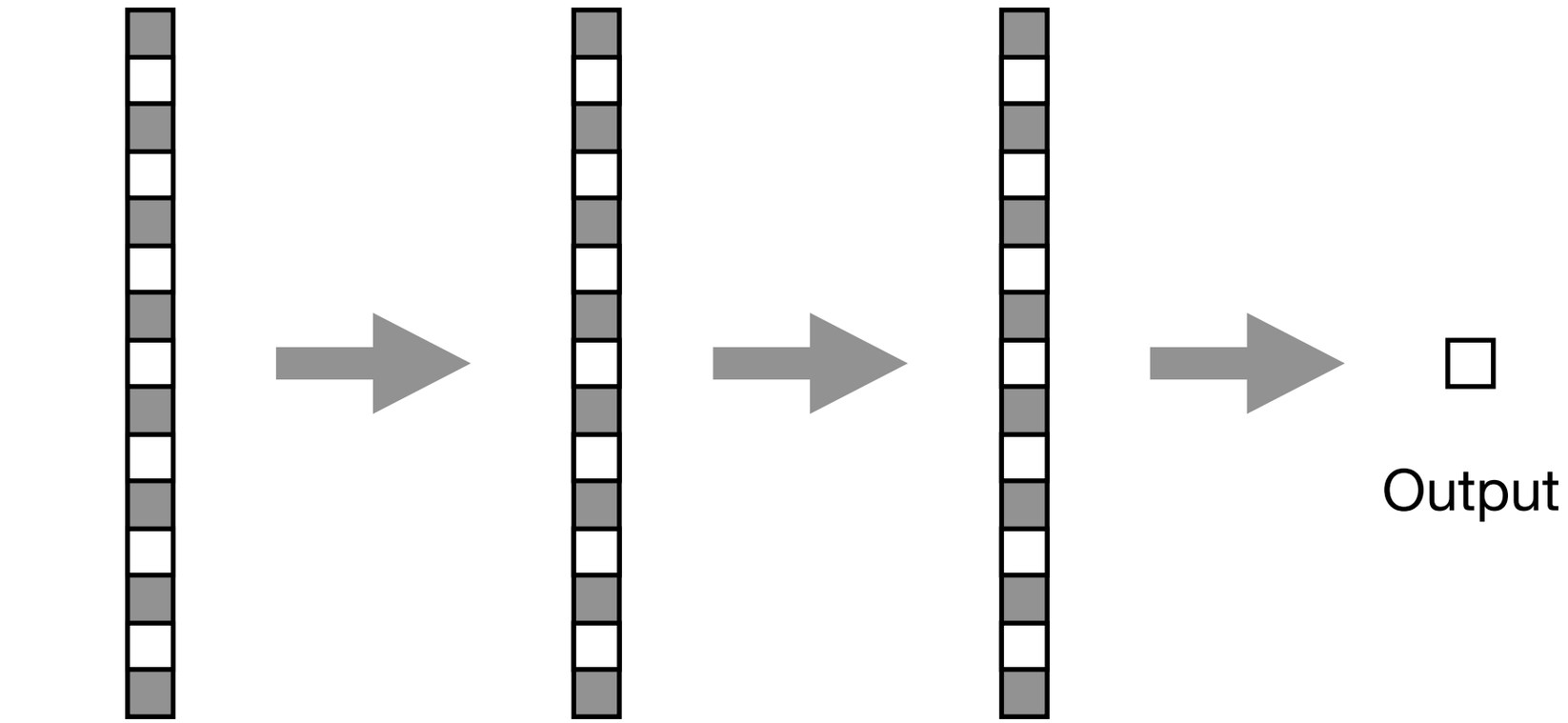
Apart from an obvious outlier there seems to be no clear pattern to the worst errors…

Data set size dependency *How many simulations do we need?*



More quantum mechanical calculations would only lead to marginally better predictions…

Neural network in PyTorch *Trying out a more complex model.*



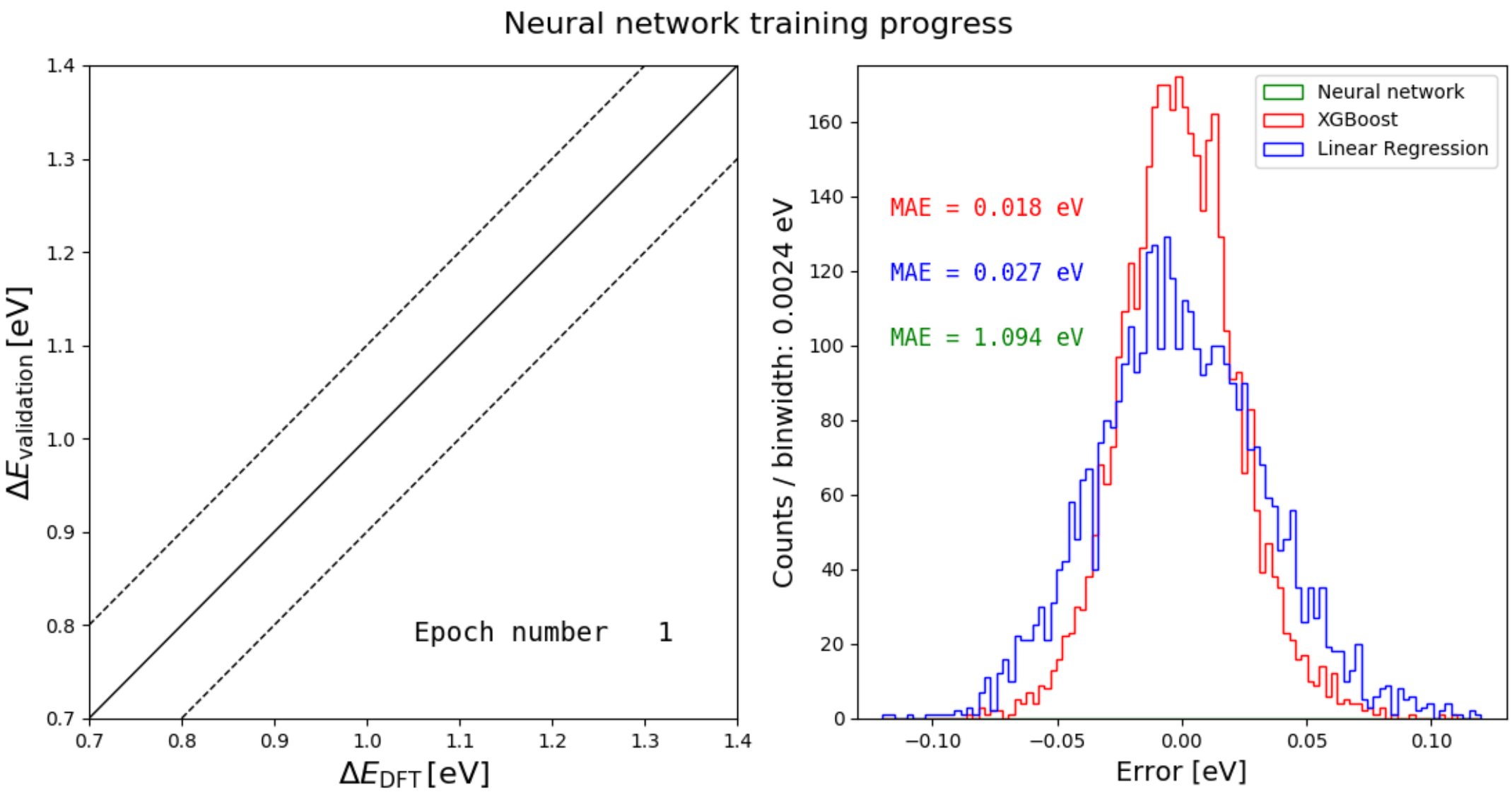
Input
77 nodes
$y = xA^T + b$

Hidden
50 nodes
$y = xA^T + b$

Hidden
50 nodes
ReLU

Optimizer: Adam algorithm **Loss function:** Mean Squared Error

Neural network in PyTorch *Trying out a more complex model.*



The neural network only performs as well as the linear regression possibly because of its simplicity…

Summing up and moving forward

Conclusions from this project:

- There will be a loss of information by truncating the input to the composition of the nearest environment of adsorption.
- XGBoost (tree-based regressor) outperforms linear regression and a simple neural network.

Impact of this project:

- Fairly accurate energy prediction of all possible adsorption sites.
- Enables estimate of optimum alloy composition.