# Lecture #6: Atomic structures encoding

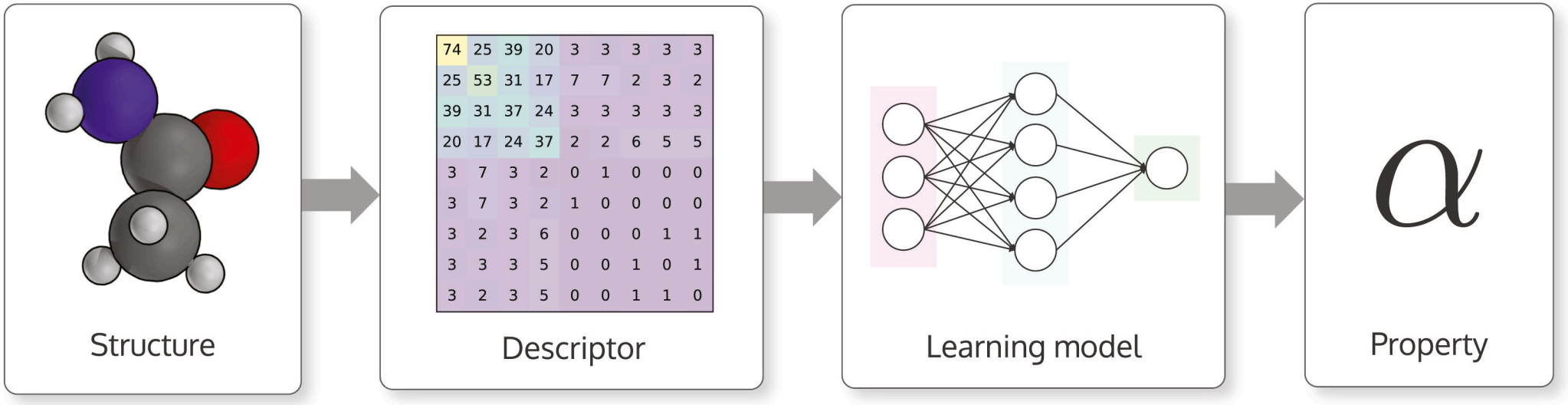
Slides by Artem Dembitskiy

## Previously on

- Supervised machine learning
- Ridge regression
- Random forest

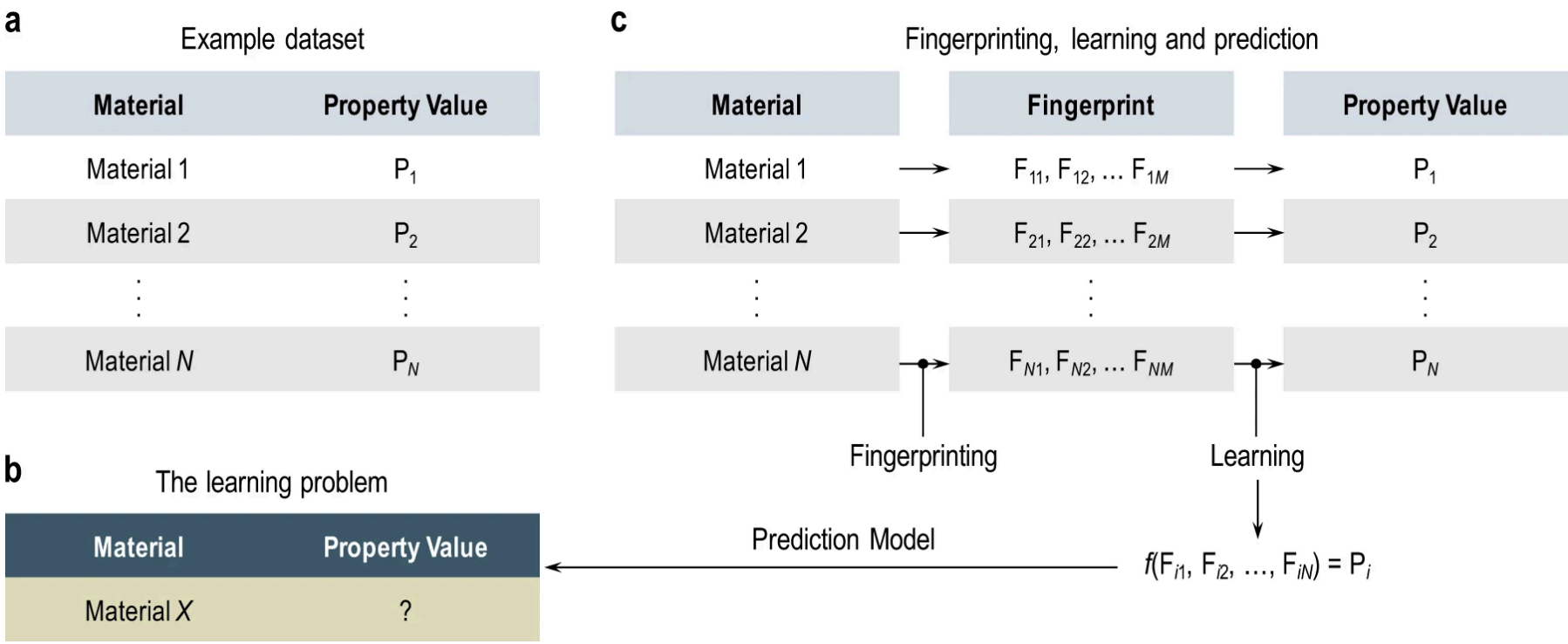
## Goals/Agenda

- Heirarchy of crystal/molecular structure descriptors
- Feature importance
- Crystal structure fingerprints


Structure
74 25 39 20 3 3 3 3 3
25 53 31 17 7 7 2 3 2
39 31 37 24 3 3 3 3 3
20 17 24 37 2 2 6 5 5
3 7 3 2 0 1 0 0 0
3 7 3 2 1 0 0 0 0
3 2 3 6 0 0 0 1 1
3 3 3 5 0 0 1 0 1
3 2 3 5 0 0 1 1 0
Descriptor
Learning model
α
Property

## Features

- In classical ML we use features
- to represent materials in a machine-readable format
- Think of it like a fingerprint



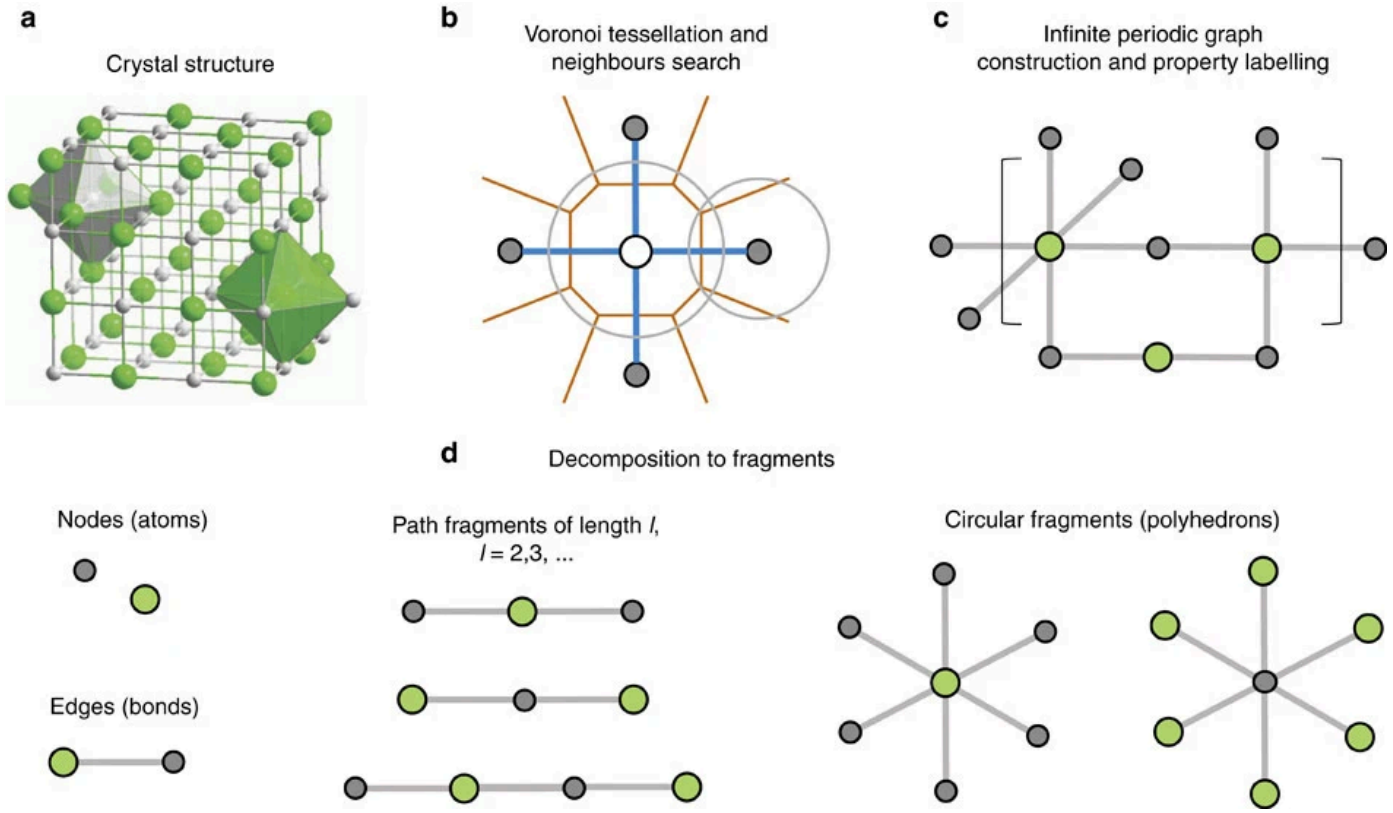
The better you find this representation for a particular problem, the better your model fits the data.

# Geometrical and compositional encoding of atomic structure

## Heirarchy of features

Depending on the resolution we have:

- Local descriptors
  - site
- Fragment descriptors
  - bond
  - polyhedron
- Global descriptors
  - chemical family
  - structural type
  - density




Universal fragment descriptors for predicting properties of inorganic crystals

## Geometrical (Structural) features

- Atomic packing (e.g. volume per atom, density)
- Voronoi polyhedra features
    - area, volume, face distance, solid angle
- bond distance/angle

## Can be calculated for sublattices

## Min/max/mean statistics

Image source

### Voronoi polehdra features can be collected by pymatgen's methodology

```
from pymatgen.analysis.local_env import VoronoiNN

features = VoronoiNN().get_voronoi_polyhedra(structure, site_id)
```

## Compositional (elemental/atomic) features

Aggregation over

- Atomic numbers
- Valence electrons
- Covalent radii
- Atomic fraction of each element in a composition
- Stoichiometry related
- Electron affinity
- Ionic radii
- Oxidation states
- Electronegativity
- ...

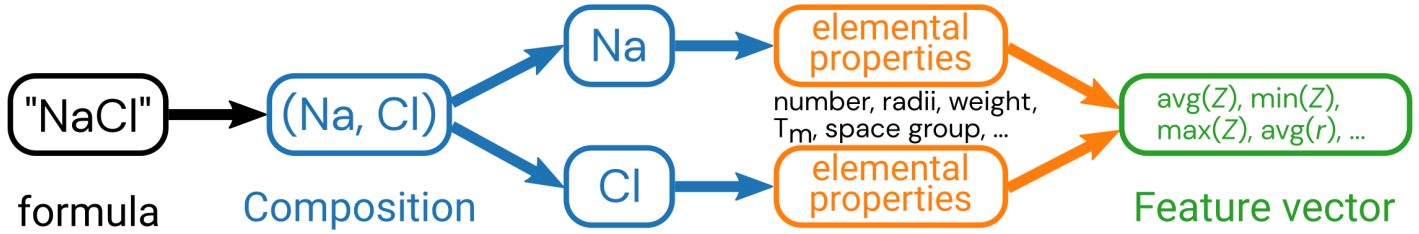
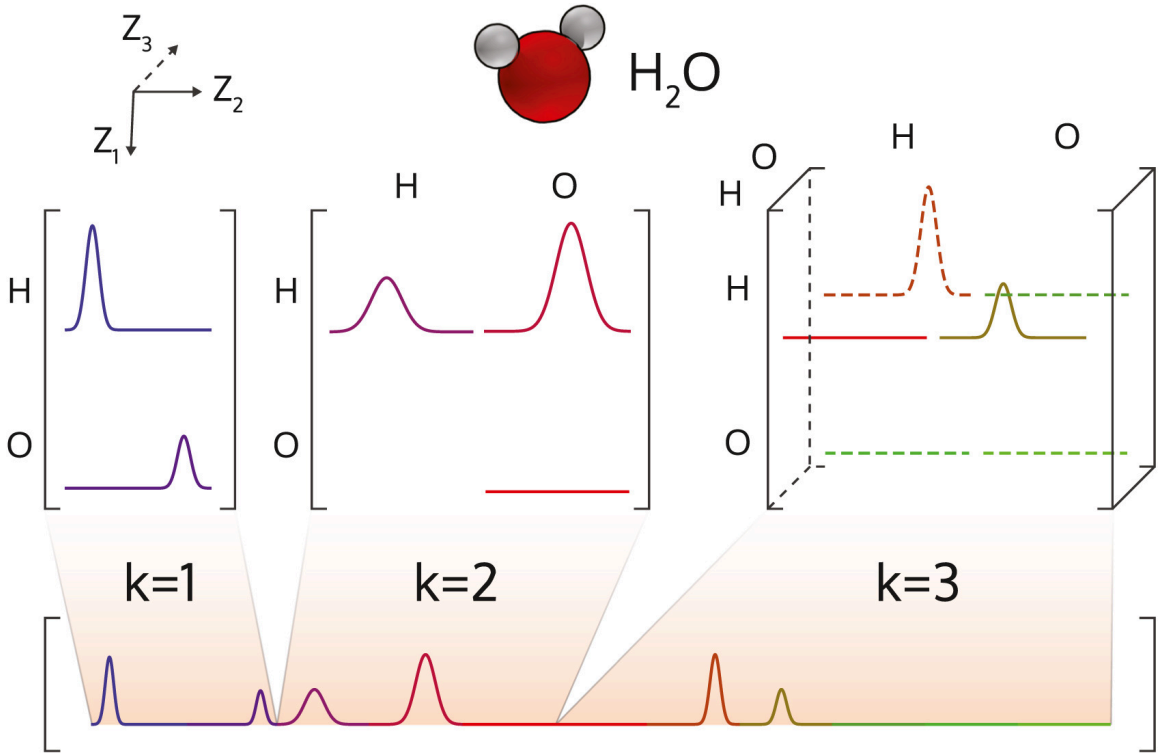



Image source

## Atomic structure fingerprints

(not taught in the course)

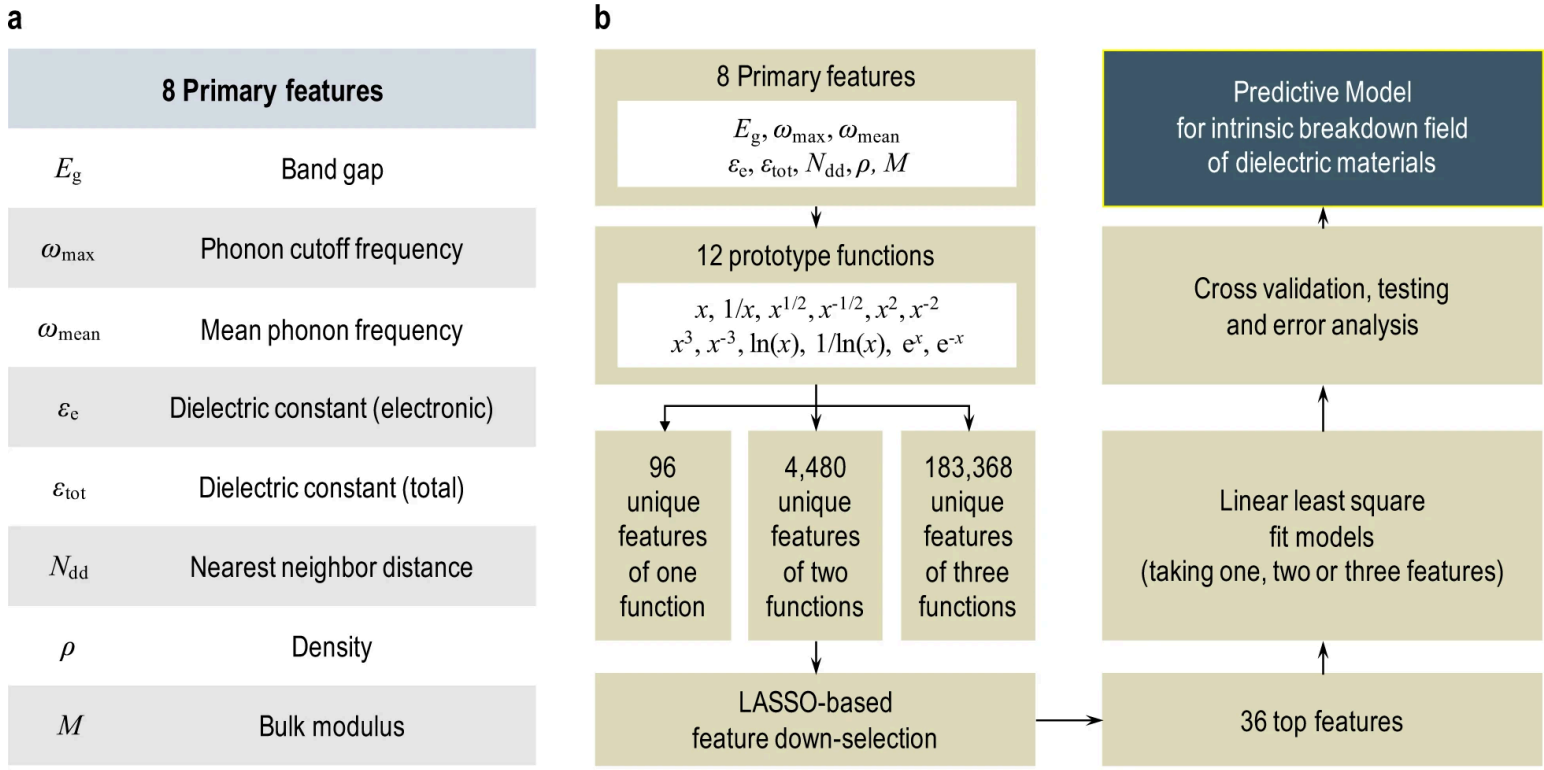
- Atom-centered Symmetry Functions
- Smooth overlap of atomic positions
- Many-body tensor representations
- Pair distribution function
- X-ray diffraction pattern

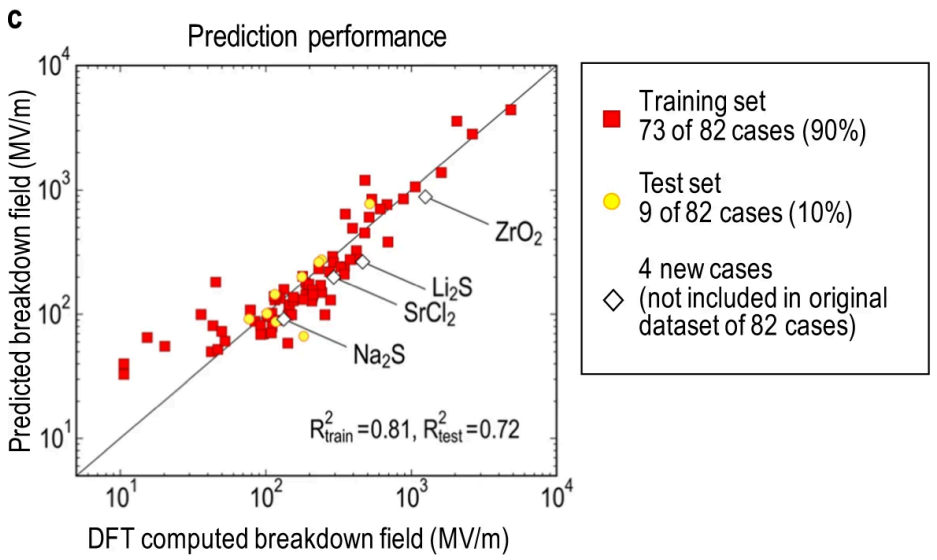


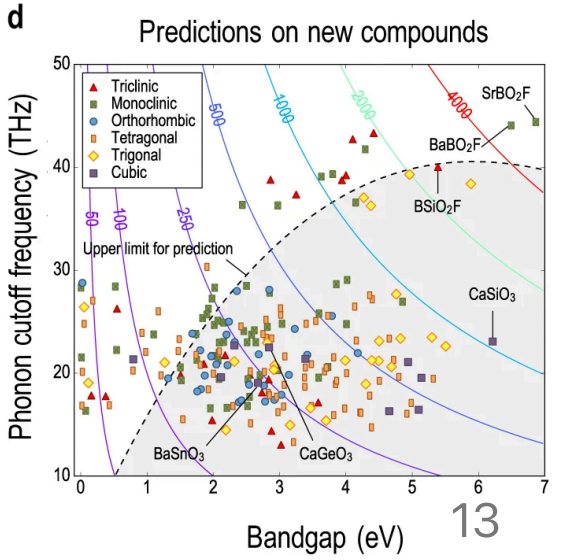
## Feature engineering

- Primary descriptors are used to design more complex features
- e.g. by applying set of mathematical operators

See SISSO paper




Machine learning in materials informatics: recent applications and prospects

## Pros

- Better performance compared to primary descriptors

## Cons

- Increased computational complexity
- Garbage feature needs to be filtered
- Lack of interpretability

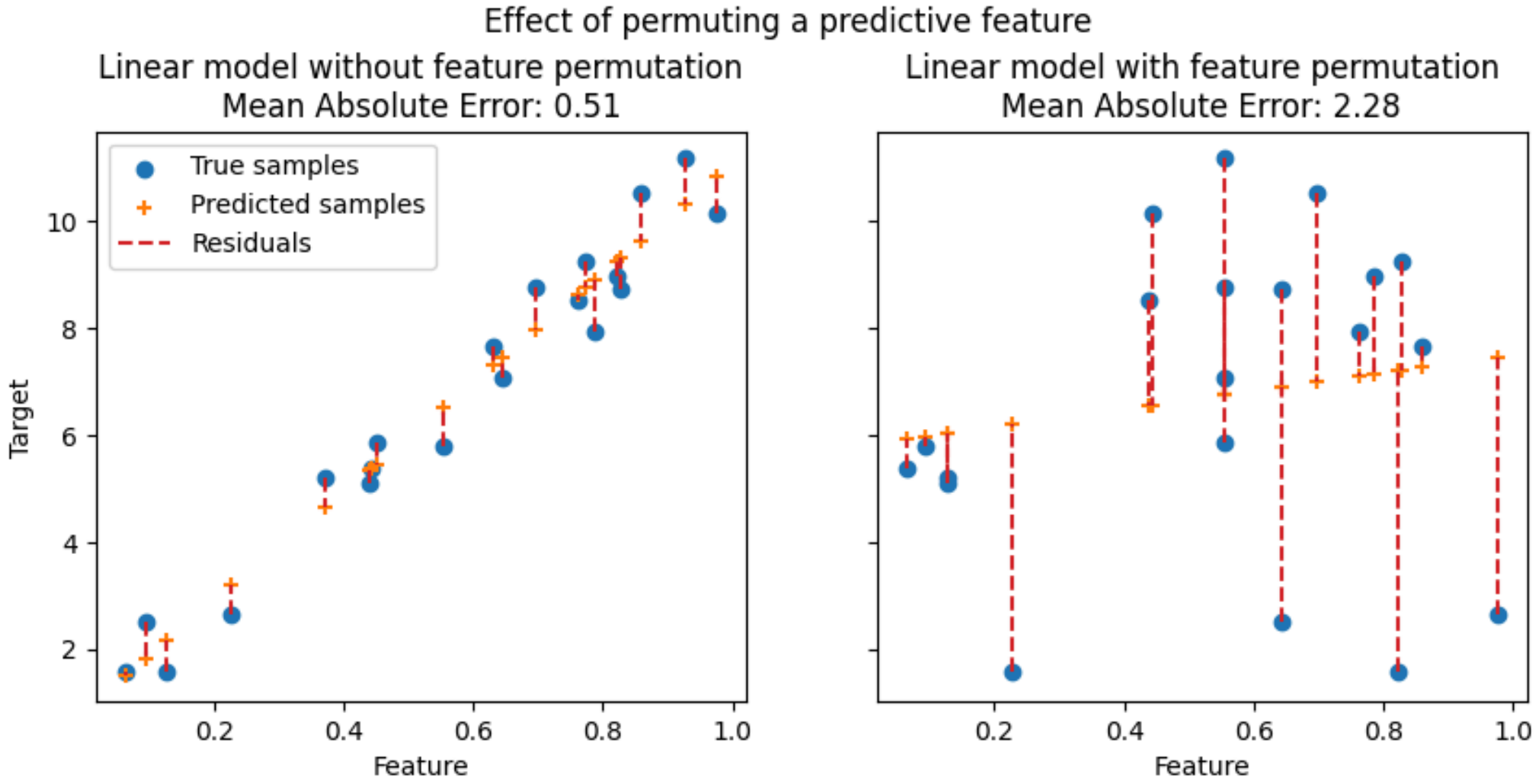
## Permutation feature importance

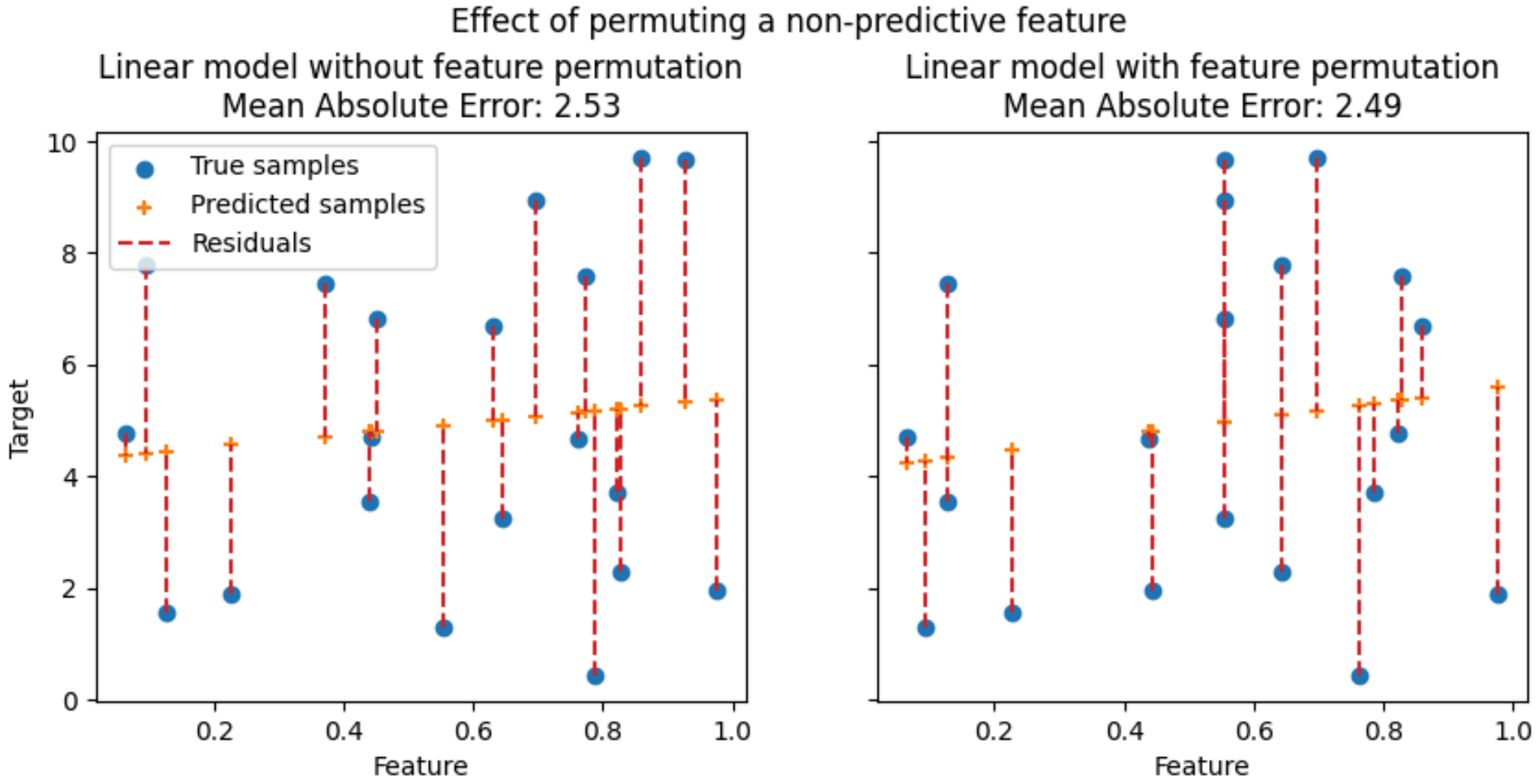
### We want to know which features most influence model prediction

Given dataset $\{X, y\}$

- Fit the model
- Get scores
- Randomly shuffle one of the feature vectors $x_i$
- Refit model
- Get scores
- The higher degradation of the model performance the more important the feature

Permutation importance in sklearn


Effect of permuting a predictive feature
Linear model without feature permutation
Mean Absolute Error: 0.51
Linear model with feature permutation
Mean Absolute Error: 2.28
True samples
Predicted samples
Residuals
Target
Feature
Feature
2
4
6
8
10
0.2
0.4
0.6
0.8
1.0


Effect of permuting a non-predictive feature
Linear model without feature permutation
Mean Absolute Error: 2.53
Linear model with feature permutation
Mean Absolute Error: 2.49
True samples
Predicted samples
Residuals
Target
Feature
Feature
0
2
4
6
8
10
0.2
0.4
0.6
0.8
1.0

## Pros

- Works for any model
- Yields variance of the feature importance

## Cons

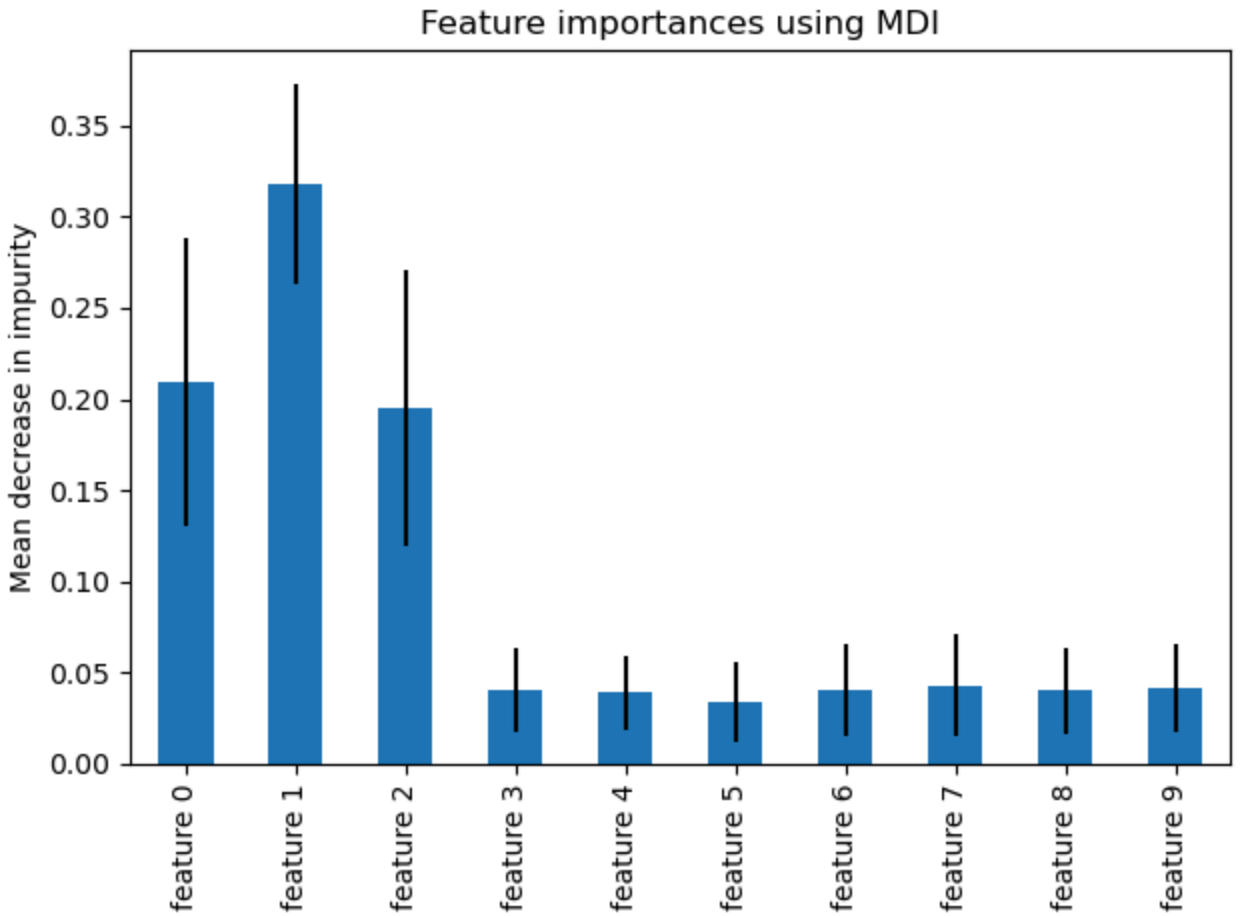
- Computational complexity
  - Requires refitting the model for each feature
- Wrong output for correlated features

## Intuition behind the impurity-based feature importance

Decision tree:

- Selects the best feature for splitting at each node
- Best split is measured by gain in purity (Gini impurity or Entropy)
- The higher gain the higher importance of the feature

In the case of Random Forest the feature importances are "computed as the mean and standard deviation of accumulation of the impurity decrease within each tree."




Feature importances with a forest of trees in sklearn

### Pros

- You have this information after fitting the model
- Yields variance of the feature importance

### Cons

- Wrong output for high-cardinality features

## Feature selection

- Drop features with low variance
- Backward/Forward feature selection
- LASSO
- Tree-based feature selection

## Why?

- Remove redundant features
- Develop robust model
- The less features the less compute is needed
- Feature collection = time = money
    - save your resources

## Forward feature selection algorithm

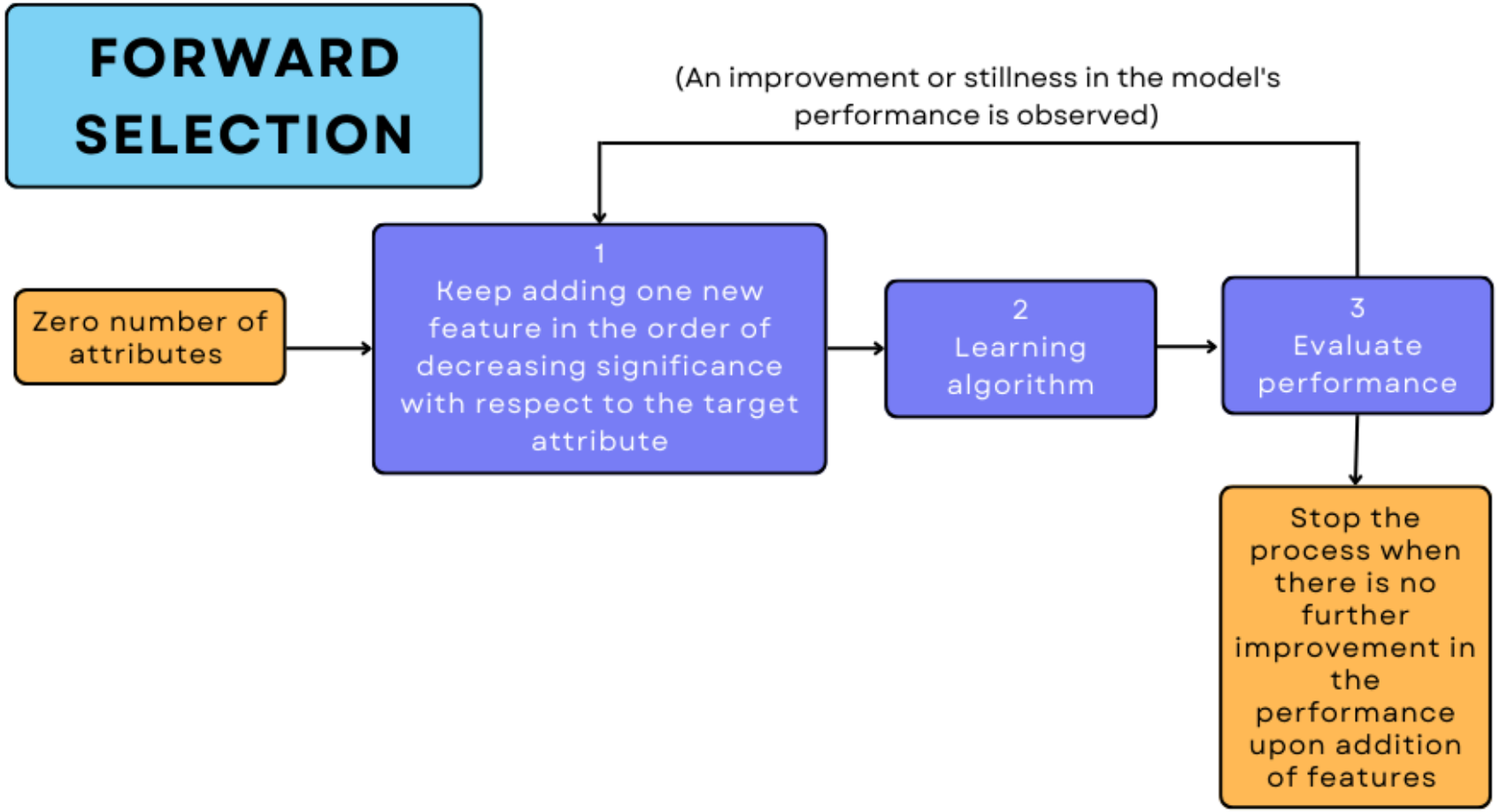



Image source

## How to collect features?

Design your own Featurizer

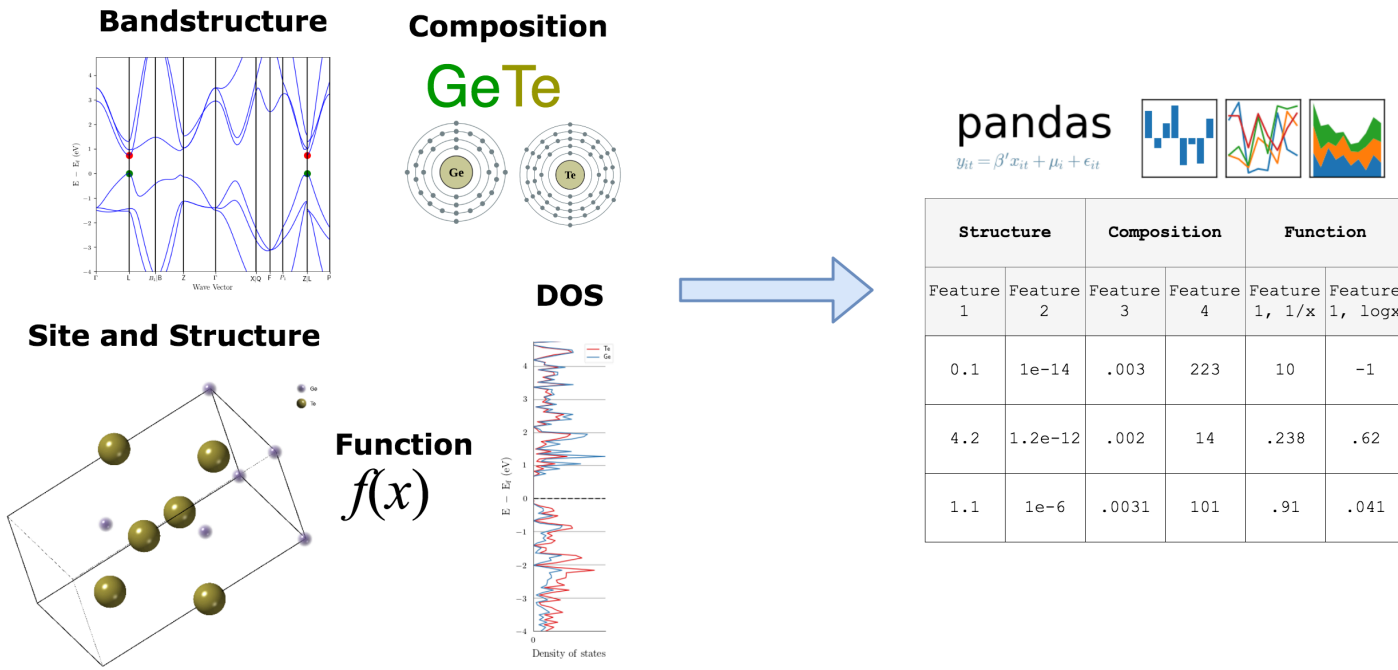
- You are an expert in your field
- Consider essential descriptors of your target
- Use pymatgen/ase/etc.

Ready-to-use Featurizers

- Matminer Python Library
- Dscribe Python library

Gross level features

- Generate gross level features non-specific to your task
- Select the best candidates




| Structure | | Composition | | Function | |
|---|---|---|---|---|---|
| Feature 1 | Feature 2 | Feature 3 | Feature 4 | Feature 1, 1/x | Feature 1, logx |
| 0.1 | 1e-14 | .003 | 223 | 10 | -1 |
| 4.2 | 1.2e-12 | .002 | 14 | .238 | .62 |
| 1.1 | 1e-6 | .0031 | 101 | .91 | .041 |

**The most important properties of an ideal descriptor:**

- Invariant with respect translation of the coordinate system
- Invariant with respect to rotation of the coordinate system
- Invariant with respect to permutation of atomic indices: changing the enumeration of atoms does not affect the target
- Unique: single way to construct a descriptor and the descriptor itself corresponds to a single property
- Continuous: small changes in the atomic structure -> small changes in the descriptor
- Compact
- Computationally cheap

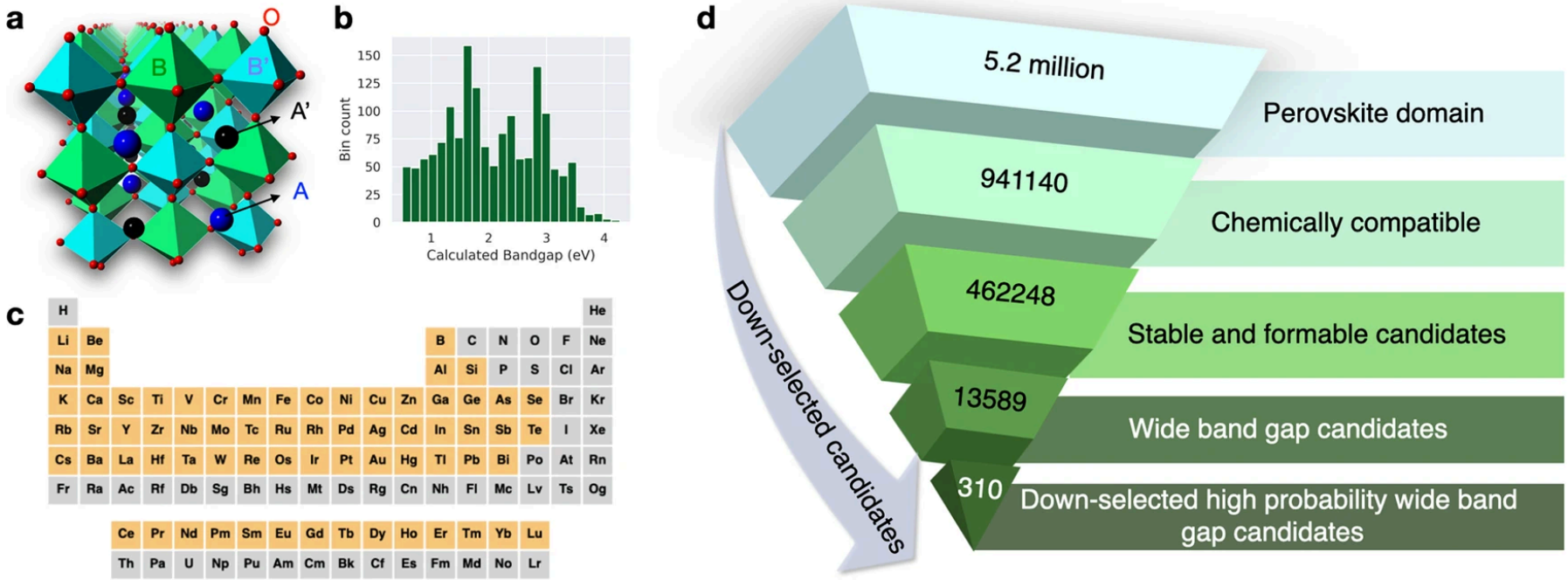
From DScribe: Library of descriptors for machine learning in materials science by Lauri Himanen at al. (Computer Physics Communications, 2020)

## Which features to consider?

- What accuracy do I need?
- At which scale your property is defined?
  - Site (i.e. defect formation energy)
  - Bond (i.e. migration barrier)
  - Structure (i.e. hardness)
- Size of the chemical space?
  - Several elements or the whole periodic table
- Diversity of crystal structures?
  - Fixed structural type? Not fixed?
- Do we know (expect) any relashionship?
- How many samples do I have?
  - ~100? ~100,000?

## You can find the right feature design by answering these questions

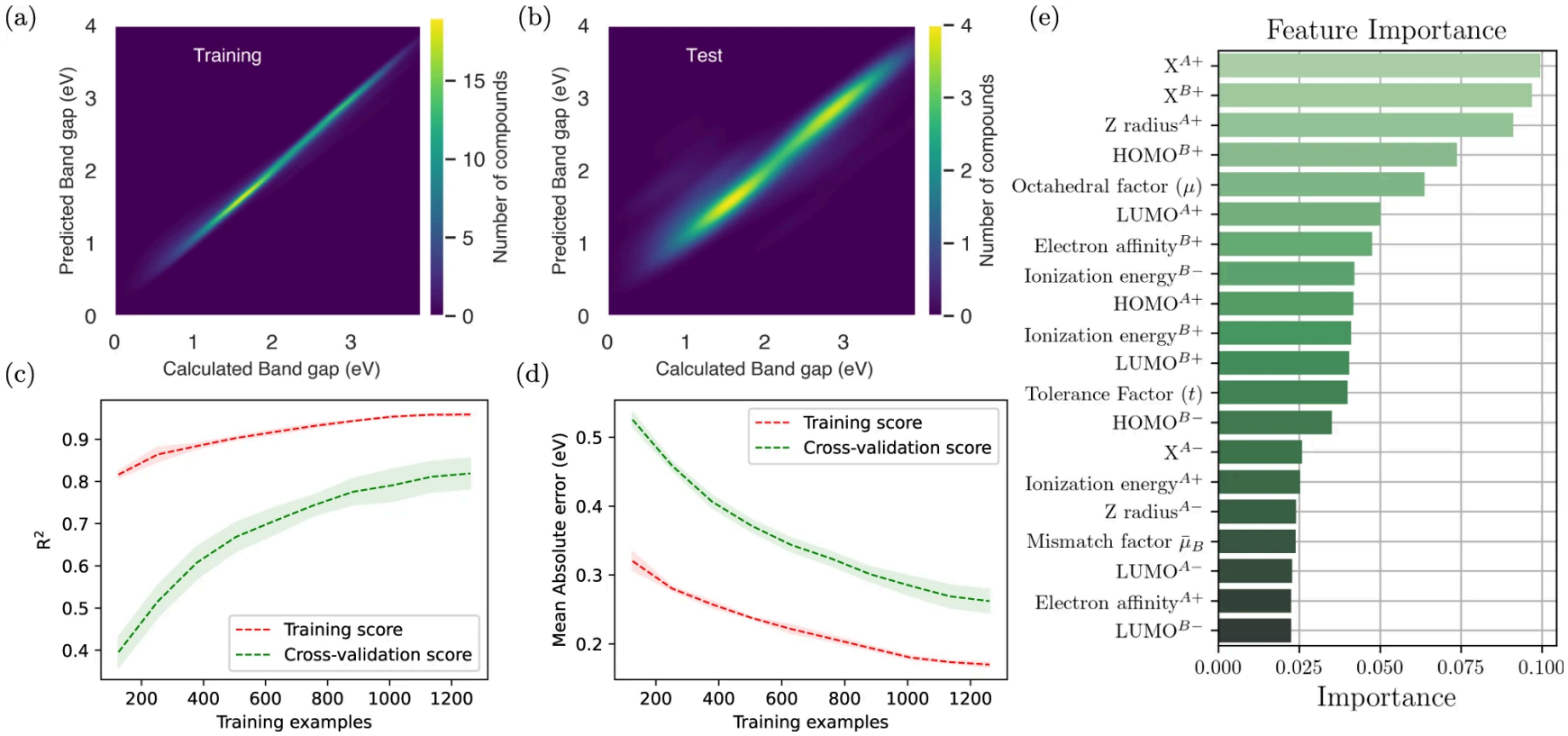
# Example


a
O
B
B'
A'
A
b
Bin count
150
125
100
75
50
25
0
1
2
3
4
Calculated Bandgap (eV)
c
H
He
Li Be
B C N O F Ne
Na Mg
Al Si P S Cl Ar
K Ca Sc Ti V Cr Mn Fe Co Ni Cu Zn Ga Ge As Se Br Kr
Rb Sr Y Zr Nb Mo Tc Ru Rh Pd Ag Cd In Sn Sb Te I Xe
Cs Ba La Hf Ta W Re Os Ir Pt Au Hg Tl Pb Bi Po At Rn
Fr Ra Ac Rf Db Sg Bh Hs Mt Ds Rg Cn Nh Fl Mc Lv Ts Og
Ce Pr Nd Pm Sm Eu Gd Tb Dy Ho Er Tm Yb Lu
Th Pa U Np Pu Am Cm Bk Cf Es Fm Md No Lr
d
5.2 million
Perovskite domain
941140
Chemically compatible
462248
Stable and formable candidates
13589
Wide band gap candidates
310
Down-selected high probability wide band gap candidates
Down-selected candidates

# Considered features

| Abbreviation | Feature |
|---|---|
| *Elemental* | |
| HOMO | Highest Occupied Molecular Orbital (eV) |
| LUMO | Lowest Unoccupied Molecular Orbital (eV) |
| IE | Ionization energy (kJ/mol) |
| X | Pauling Electronegativity |
| Z radius | Zunger's Pseudopotential radius (a.u.) |
| EA | Electron affinity (kJ/mol) |
| *Geometric* | |
| $t$ | Tolerance factor |
| $\mu$ | Octahedral factor |
| $\bar{\mu}$ | Mismatch factor |

## Features for Eg prediction




Band gap predictions of double perovskite oxides using machine learning

# Thank you for your attention!