

Lecture #5: Classical ML for materials science pt.1

Course roadmap

1. What is materials informatics?

- Trial-and-error paradigm
- Python crash course

3. Data in materials science

- The Materials Project
- Phase diagrams

2. Python for atomistic modeling

- ASE/Pymatgen
- NN list

4. EDA

- Why is visualization important?
- E_g vs. Electronegativity

Next: ML for materials science



Goals

- Overview of materials science problems solved with ML
- Overview of selected regression models
- Learn how to train and evaluate the model

Lecture overview

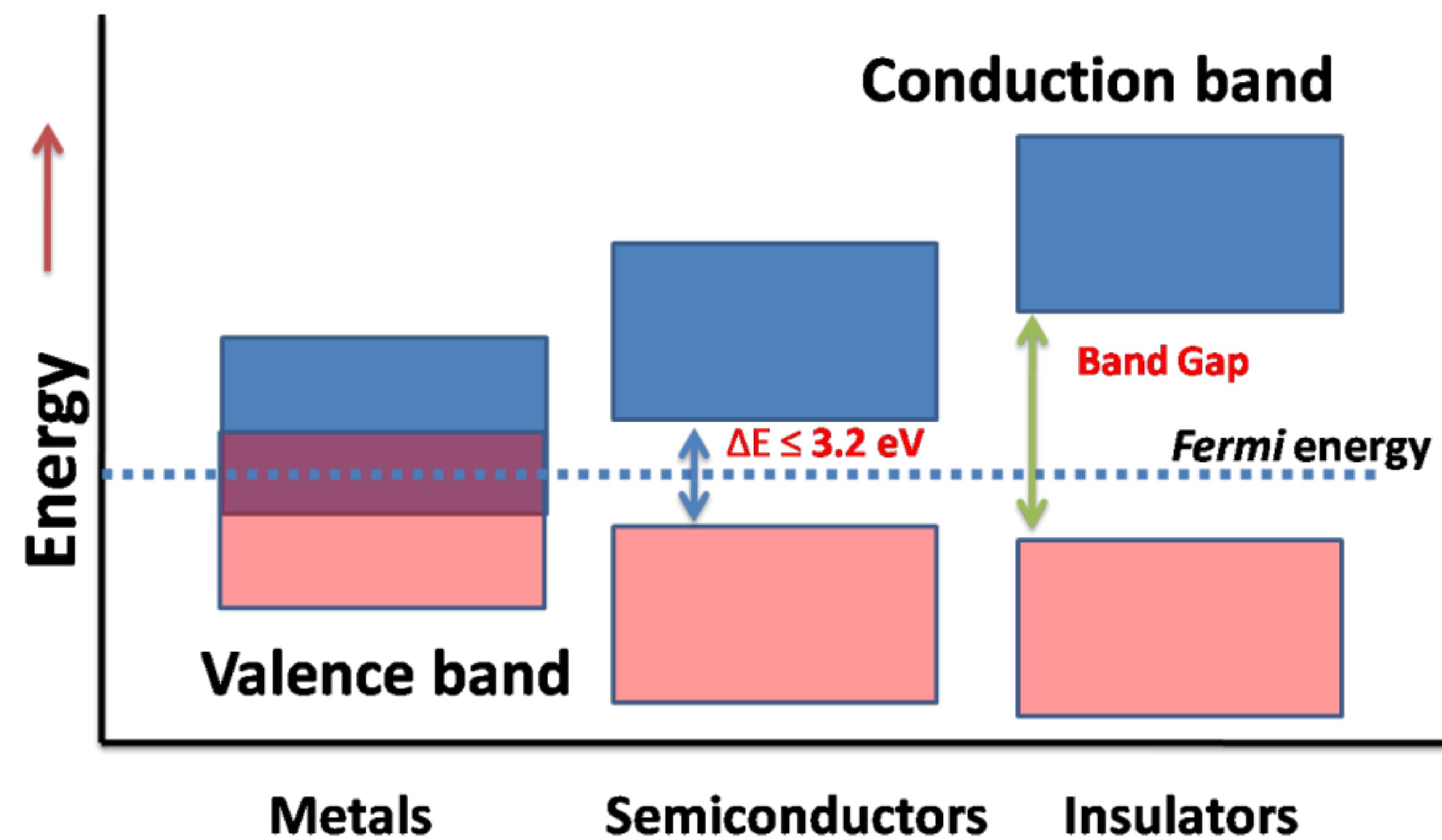
- ML tasks
- Why and when use ML?
- Property and descriptor
- Linear regression
- Cost function
- Ridge regression
- Gradient descent
- Decision tree and Random Forest
- Model evaluation

Why use ML?

- "Machine learning algorithms aim to optimize the performance of a certain task by using examples and/or past experience."
- "Study of statistical algorithms that can learn from data and generalize to unseen data, and thus perform tasks without explicit instructions." (wiki)
- Machine learning accelerates materials design by efficiently learning the structure-property relationship from data to predict unknown properties with minimal experimental or computational effort.

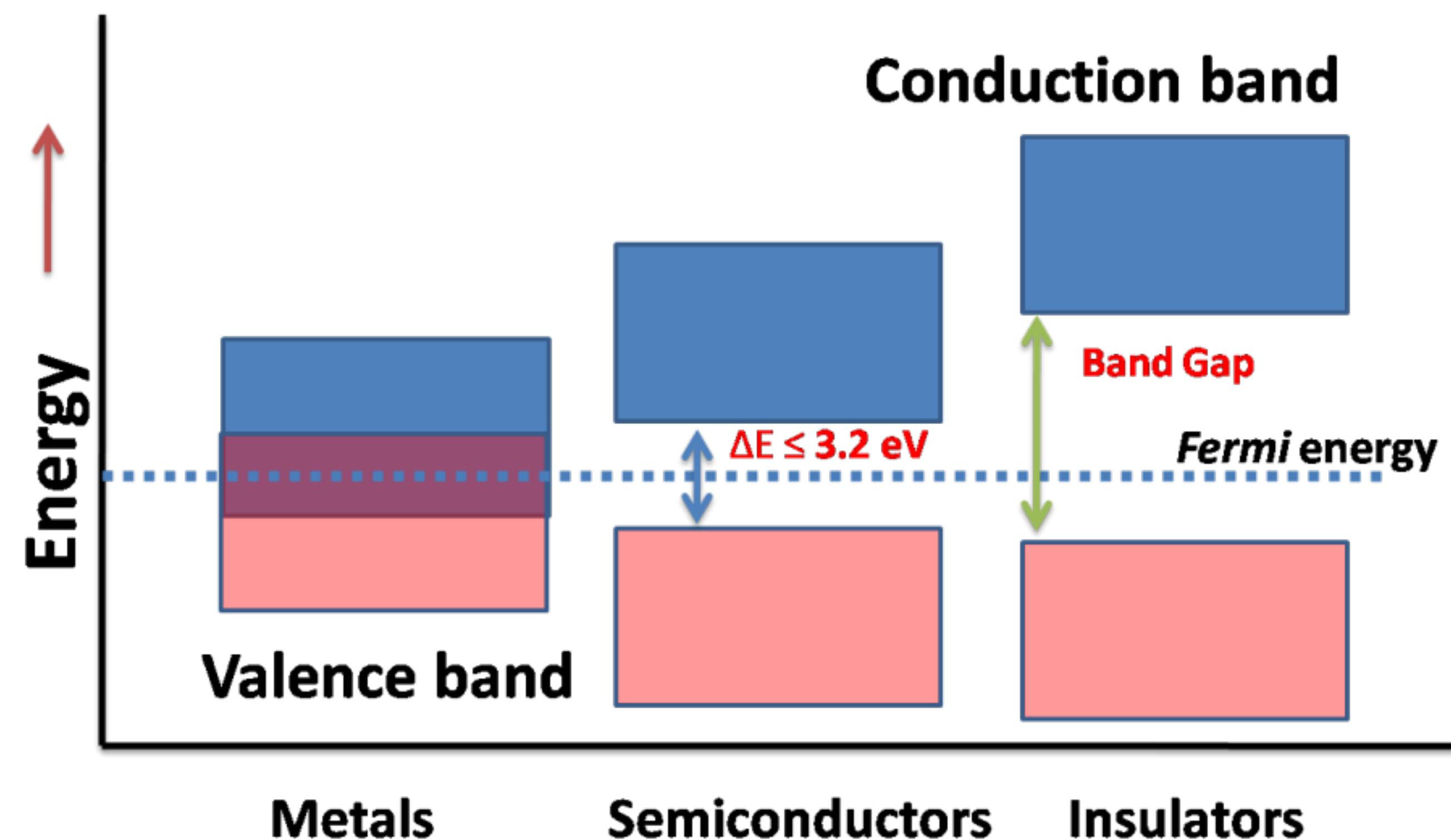
Why know band gap of a material?

- Will it conduct electricity?
- How will it interact with light?
- Can it be used for a transistor, an LED, or a solar cell?



How to know a band gap?

- Experiment (UV-Vis absorption) ~1 week
 - synthesis
 - several measurements
 - equipment required
- DFT HSE ~ 1-10 days
 - crystal structure required
 - HPC required
 - computationally expensive



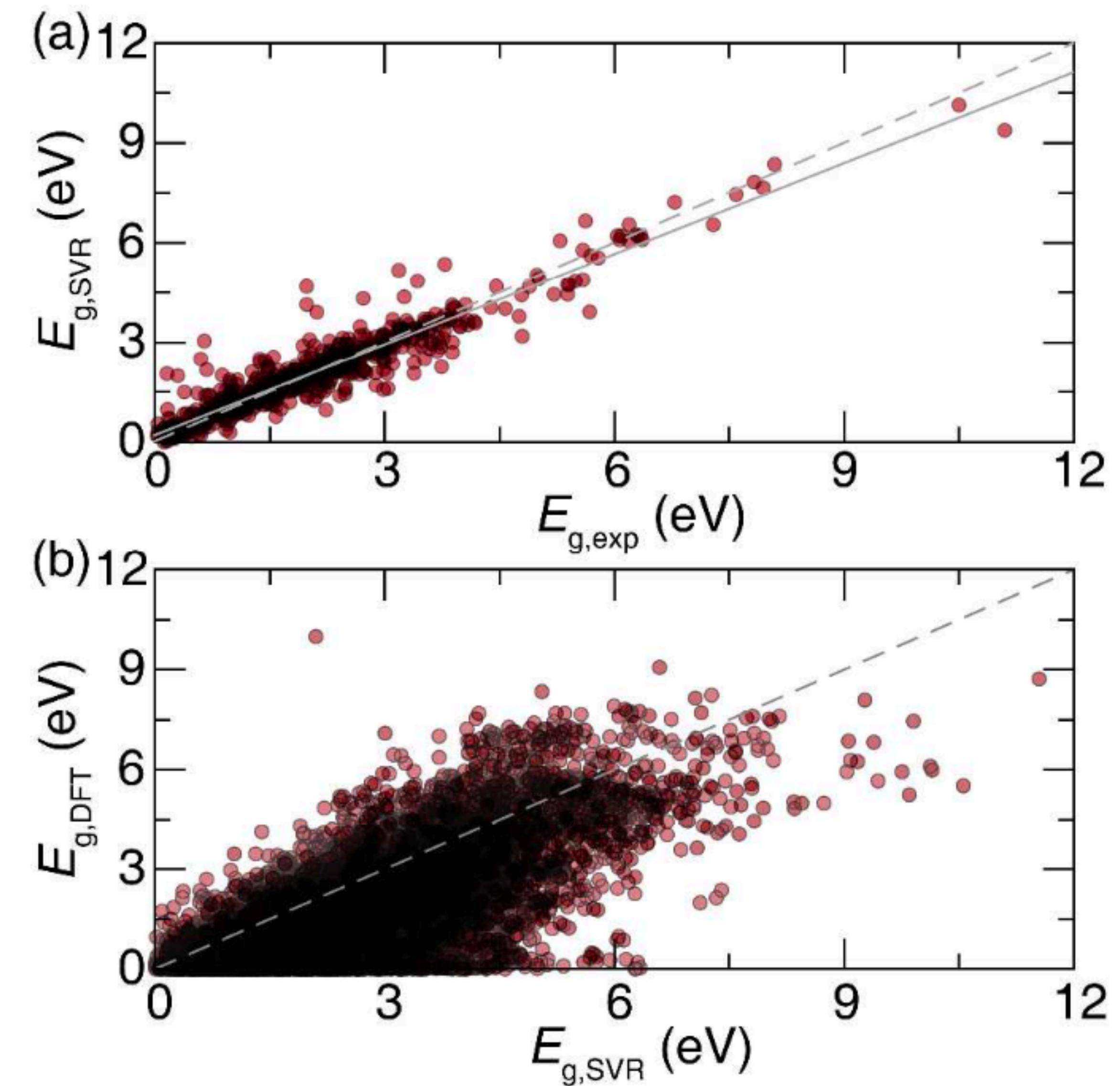
What if we have 10,000 candidates?

- How many years would it take to find the best material?
- Experiment
 - One researcher
 - $10,000 \times 7$ days -> **200 years** without parallelization
- Computation:
 - HPC with 1,600 CPU cores, 32 cores per structure -> 50 structures in parallel
 - $10,000 \times 7 / 50$ -> **4 years**

Surrogate model

- Takes < 1 second to predict the property
 - Trained on existing data
 - Chemical composition only
 - Simple laptop is sufficient

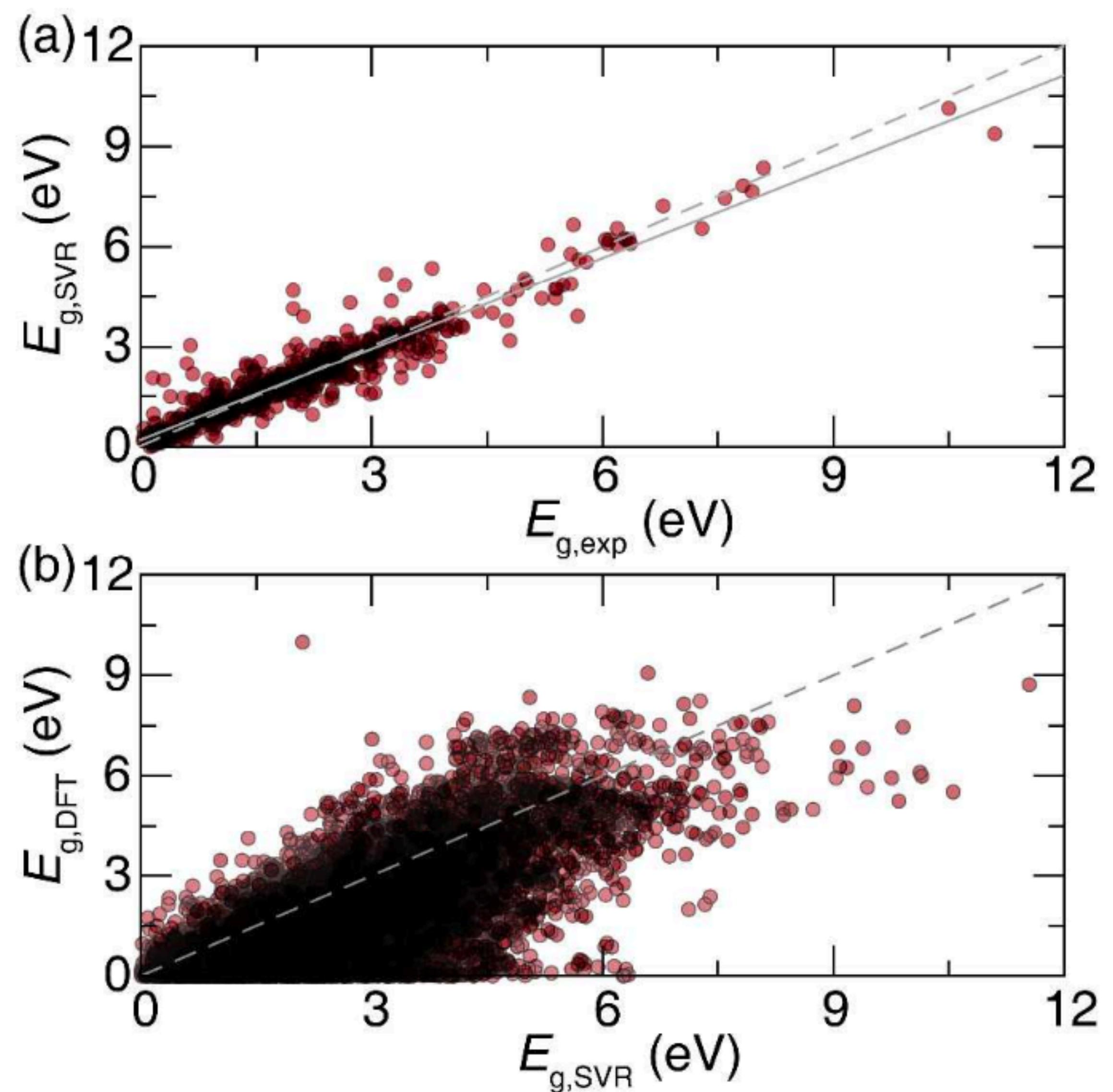
SVR = Support Vector Regression



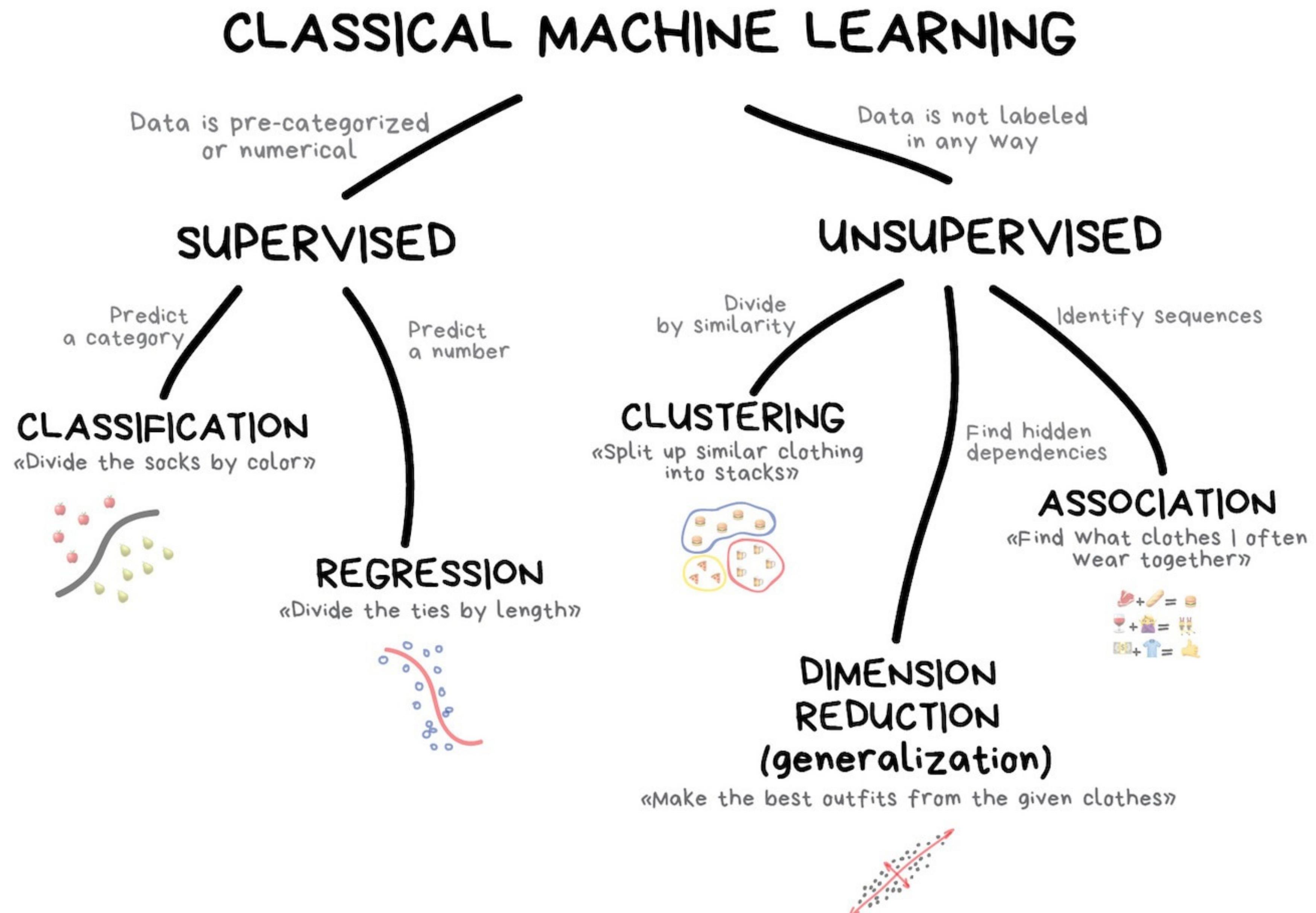
Limitations

- High quality data is required to fit the model ([Garbage in, garbage out](#))
- Accuracy is good for screening stages (RMSE = 0.45 eV)
 - but poor for a more specific tasks

There is a trade off between speed and accuracy



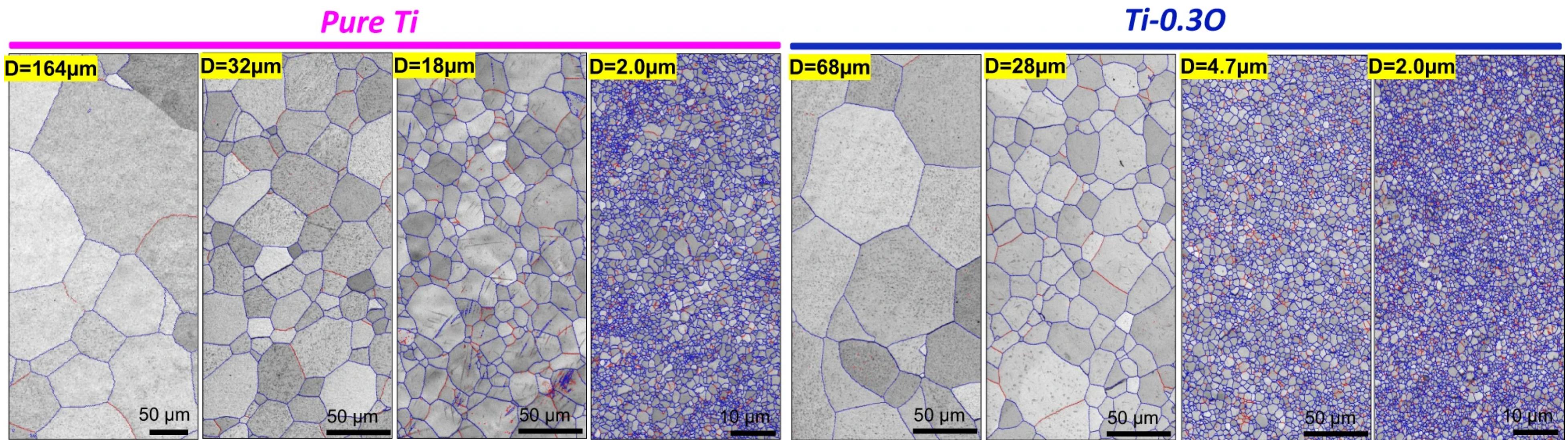
(Classical) ML tasks



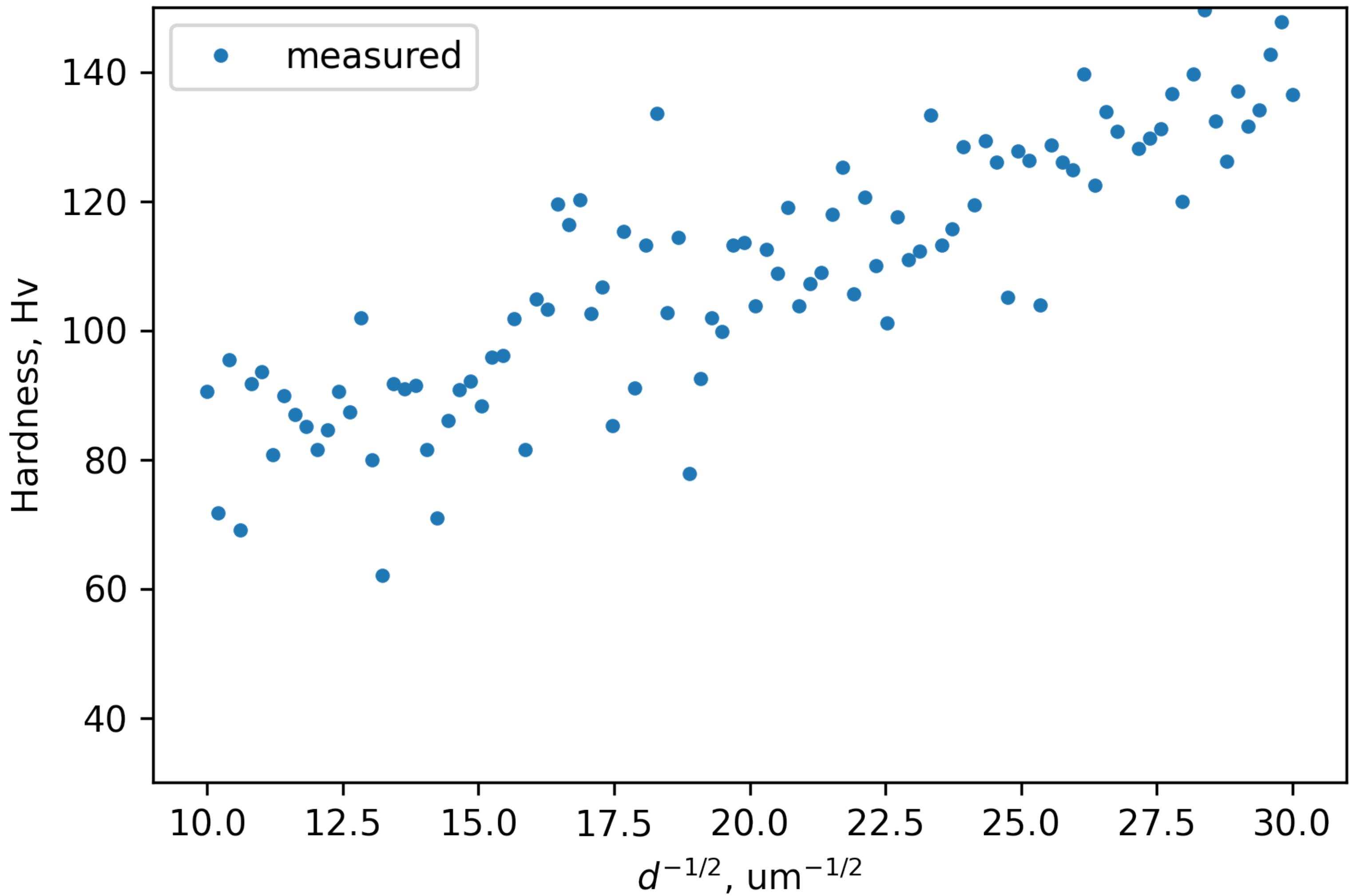
Supervised learning

"Tries to find the unknown function that connects known inputs to unknown outputs"

Grain size of alloys

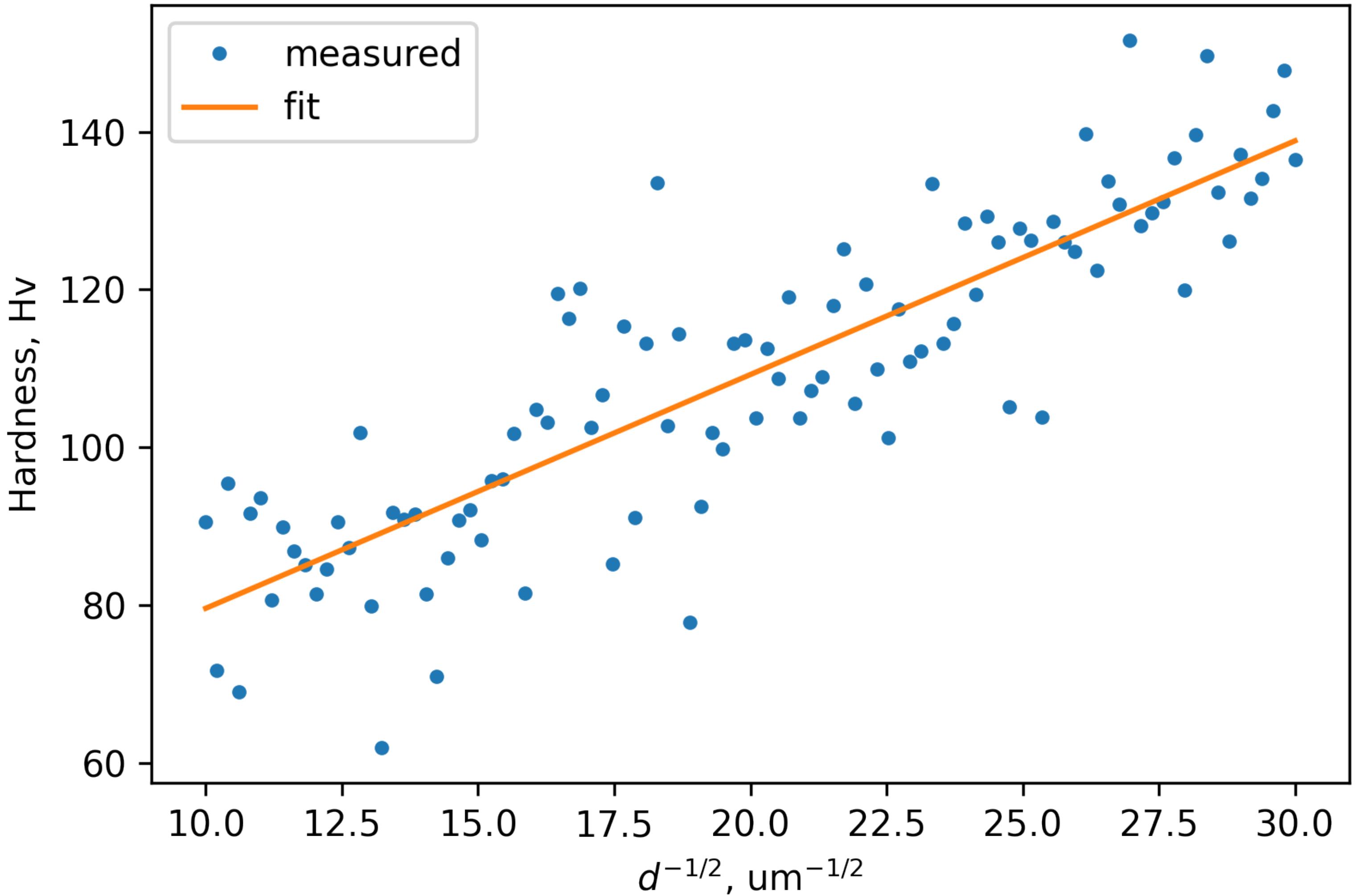


- Input: grain size
- Output: hardness



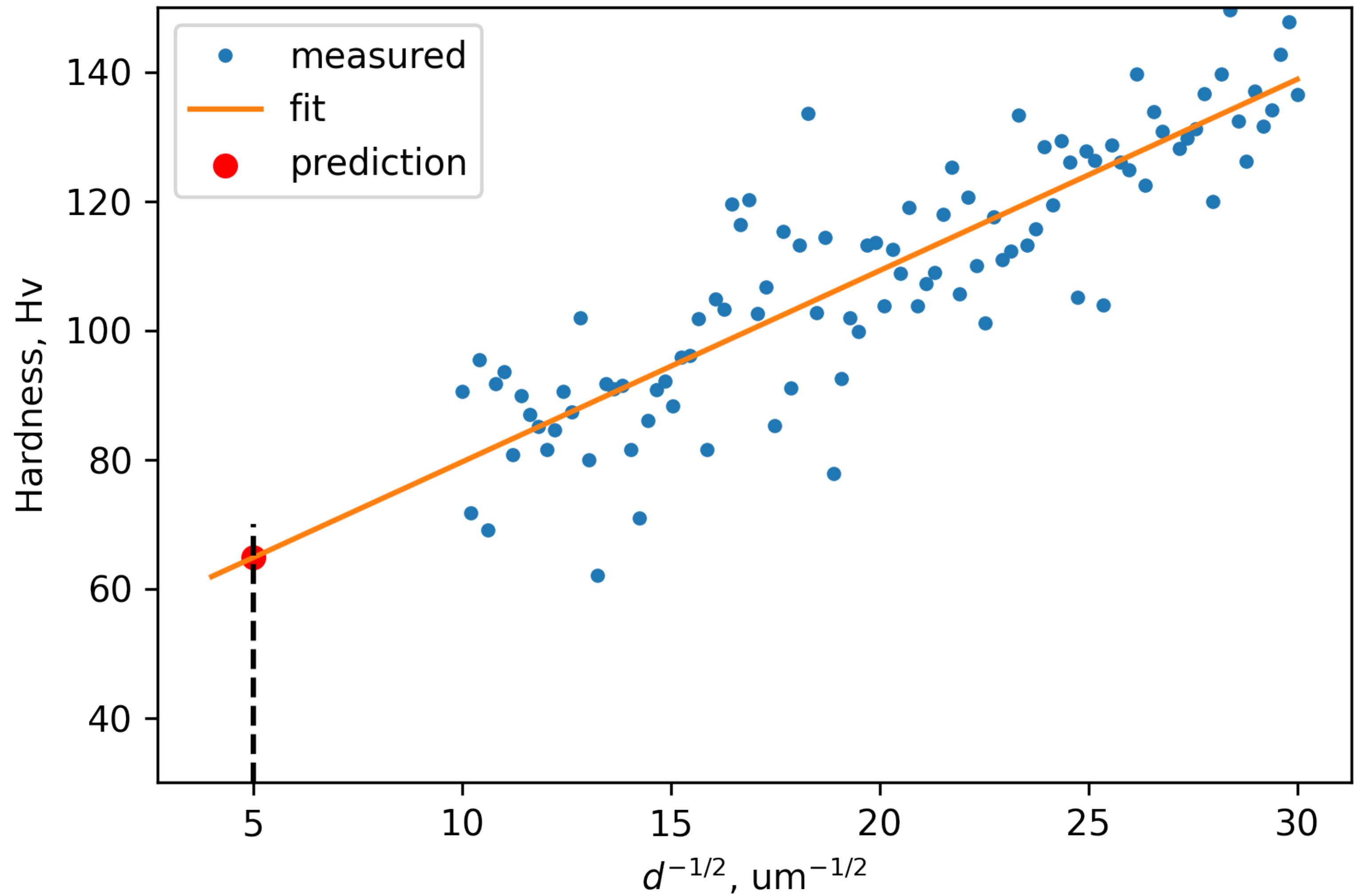
Model fit

- Input (features):
grain size (known)
- Output (label):
hardness (known)



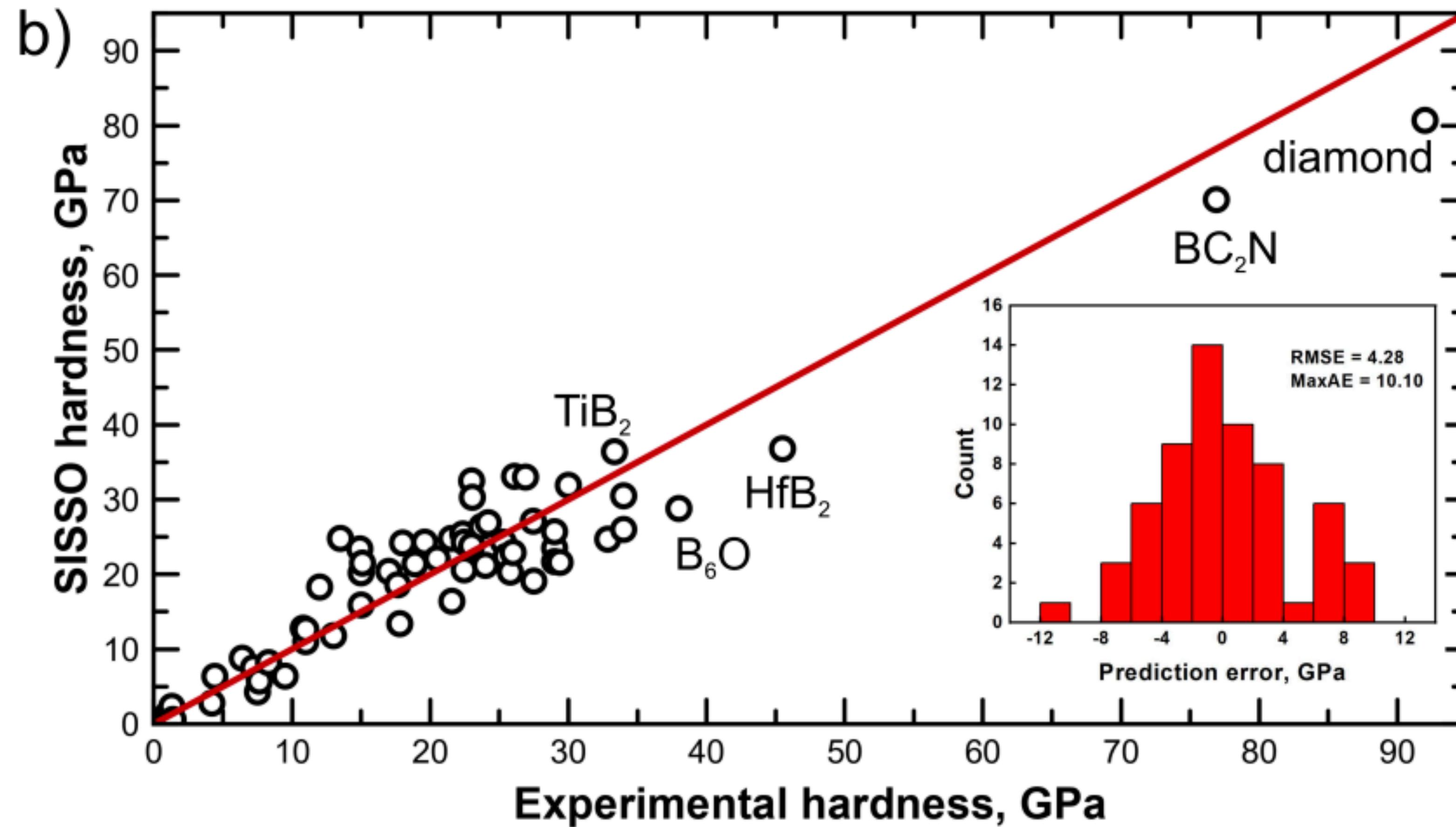
Model prediction

- Hardness($d^{-1/2} = 5 \text{ um}^{-1/2}$)?
 - Input: grain size (known)
 - Output: hardness (unknown)

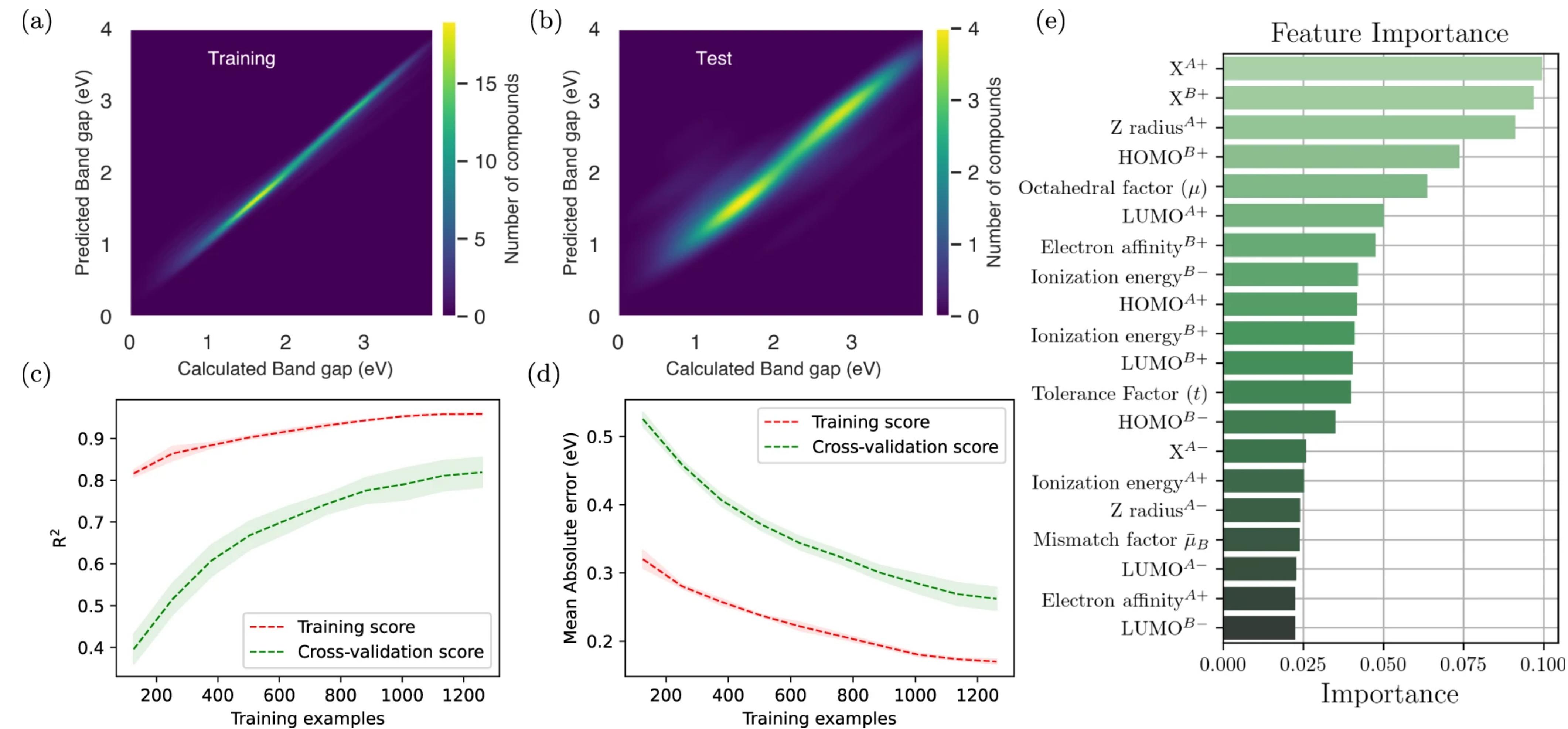


Supervised ML for materials science (examples)

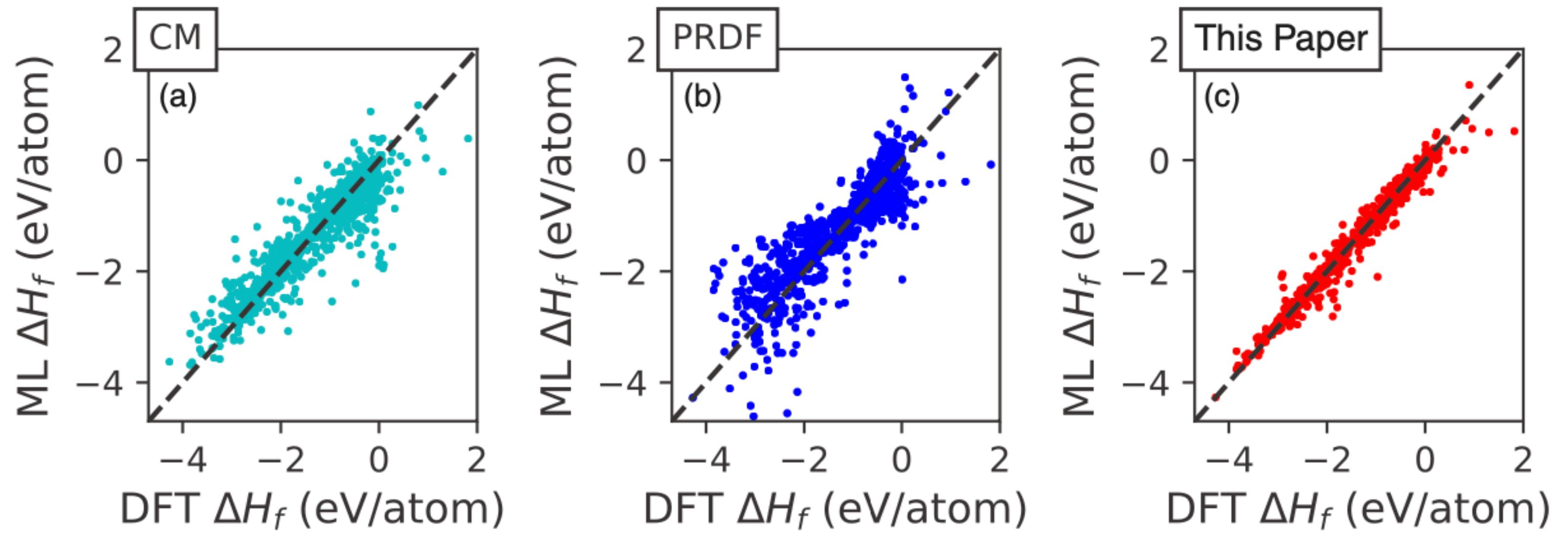
Hardness



Band gap



Formation energy



R: 0.923
 MAE: 0.25 eV/atom
 RMSE: 0.38 eV/atom
 Max Error: 2.03 eV/atom

R: 0.858
 MAE: 0.37 eV/atom
 RMSE: 0.50 eV/atom
 Max Error: 2.21 eV/atom

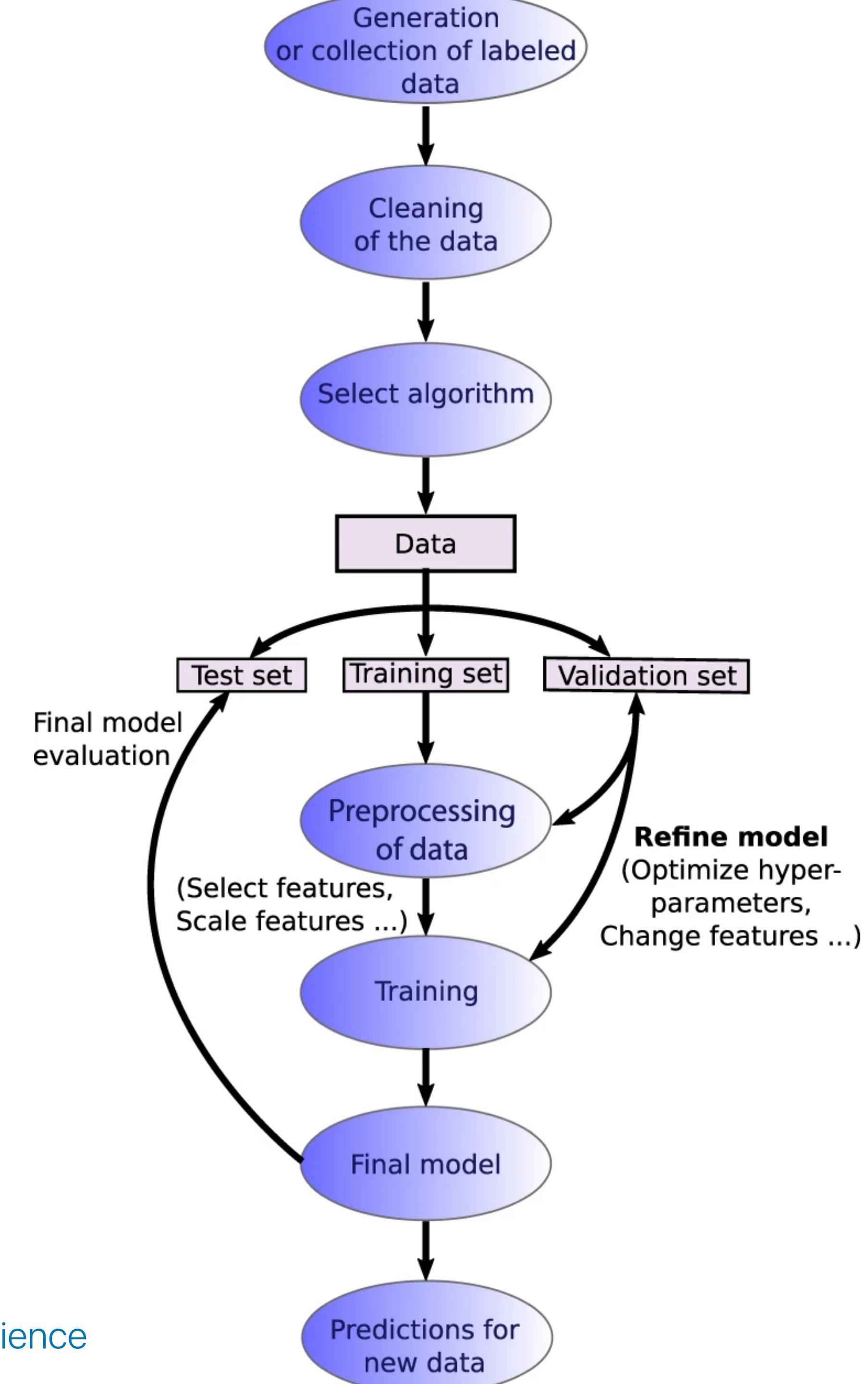
R: 0.988
 MAE: 0.09 eV/atom
 RMSE: 0.15 eV/atom
 Max Error: 1.30 eV/atom

Self-supervised learning

- is about "finding patterns in unlabeled data"
- (not taught on course)



Basic supervised learning workflow



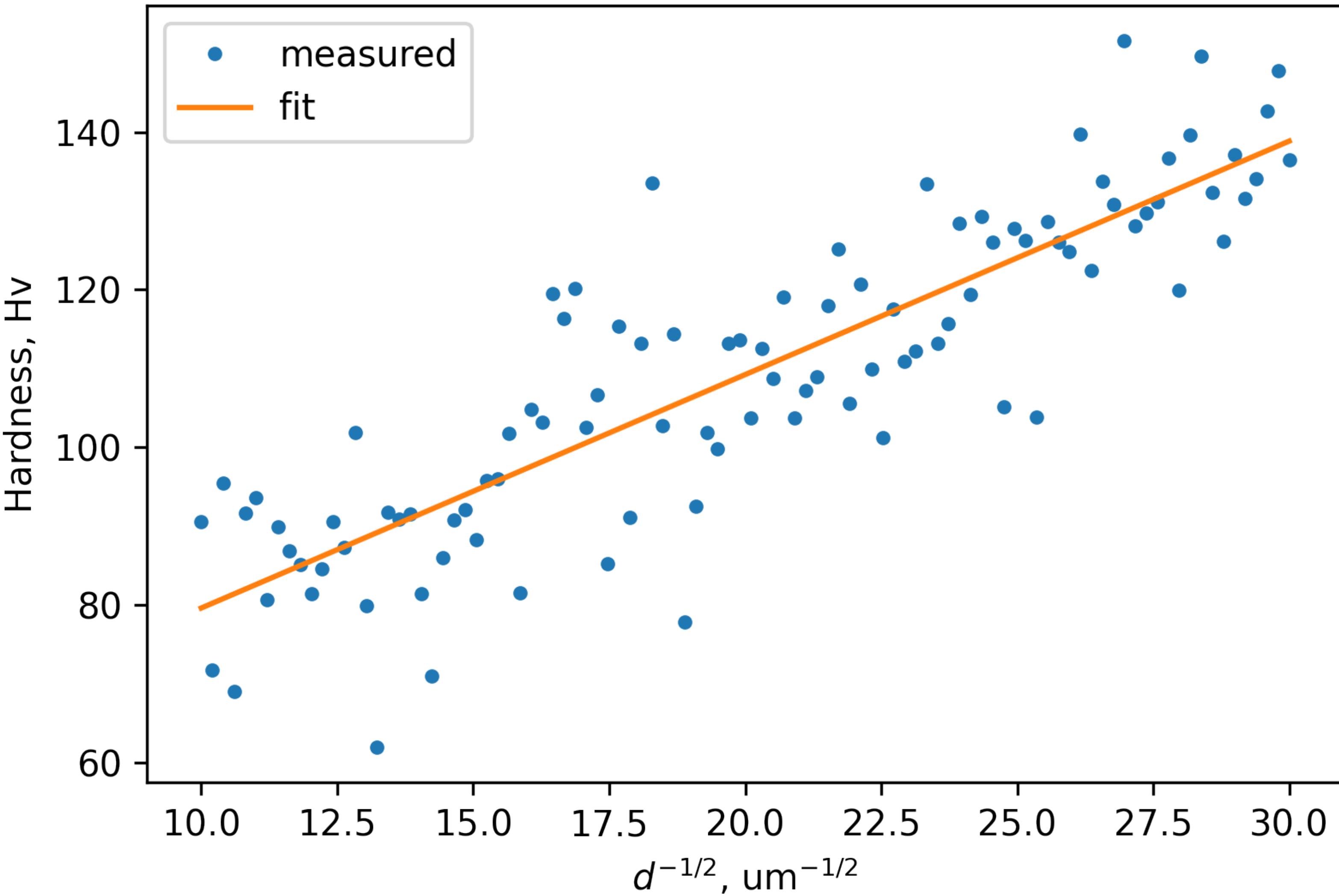
Attribution

- Parts of the next slides are adapted from the Machine Learning (MA060018) course given by Prof. Evgeny Burnaev at Skolkovo Institute of Science and Technology
- Consider enrolling in this course if you are interested in fundamentals of ML

Materials:

- https://github.com/dzisandy/Machine-Learning/blob/master/Lecture%20Slides/Lecture_2_Regression_handout.pdf
- https://github.com/dzisandy/Machine-Learning/blob/master/Lecture%20Slides/Lecture_5_Decision_Trees_handout.pdf

(Multiple)
Linear
regression



Data samples: $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_i, y_i), \dots, (\mathbf{x}_n, y_n)\}$

y_i - response variable (regressand), also called label/target/property

$\mathbf{x}_i = (x_i^1, \dots, x_i^d)$ - $1 \times d$ vector of input features (regressors), also called features/descriptors

Assume that relationship between y and \mathbf{x} is linear:

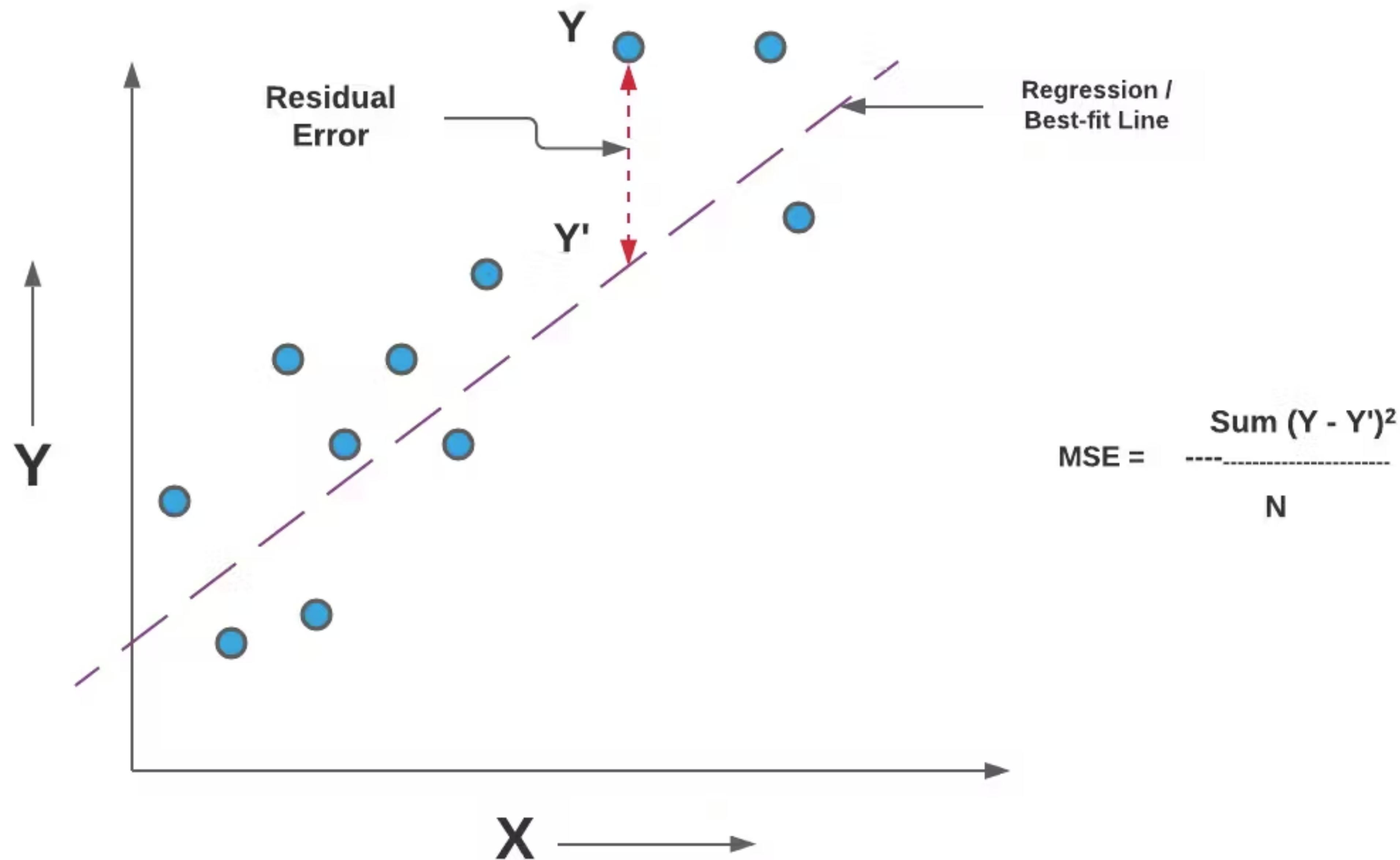
$$\hat{f} = b + w_0x_0 + w_1x_1 + \dots w_dx_d$$

$$\hat{f} = \mathbf{w}\mathbf{x}^T + b$$

\mathbf{w} and b are unknown parameters of the model

Task:

Find parameters of \hat{f} that will minimize the error = $\frac{1}{n} \sum_i^n ||y_i - \hat{f}(\mathbf{x}_i)||^2$



Loss function

$$J(\mathbf{w}, b) = \frac{1}{n} \sum_i^n (\mathbf{w} \mathbf{x}_i^T + b - y_i)^2$$

Optimization problem:

$$J(\mathbf{w}, b) \rightarrow \min_{\mathbf{w}, b}$$

Matrix form

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1d} & 1 \\ x_{21} & x_{22} & \dots & x_{2d} & 1 \\ \dots & & & & \\ x_{n1} & x_{n2} & \dots & x_{nd} & 1 \end{bmatrix} \quad W = \begin{bmatrix} w_1 \\ \dots \\ w_d \\ b \end{bmatrix} \quad Y = \begin{bmatrix} y_1 \\ \dots \\ y_n \end{bmatrix}$$

$$J(W) = \frac{1}{n} || XW - Y ||^2 \rightarrow \min_W$$

Solution

$$J(W) = \frac{1}{n} ||XW - Y||^2 \rightarrow \min_W$$

$$\nabla J(W) = 0$$

$$X^T X W = X^T Y$$

$$W = (X^T X)^{-1} X^T Y, \text{ if } X^T X \text{ is invertible}$$

$$W = (X^T X)^{-1} X^T Y, \text{ if } X^T X \text{ is invertible}$$

- Will fail if there are linearly dependent \mathbf{x}_i in X
- Computationally expensive if $d > > 1, n > > 1$
- No regularization
 - model is not robust to noise and outliers
 - model can overfit

Generalization of a model

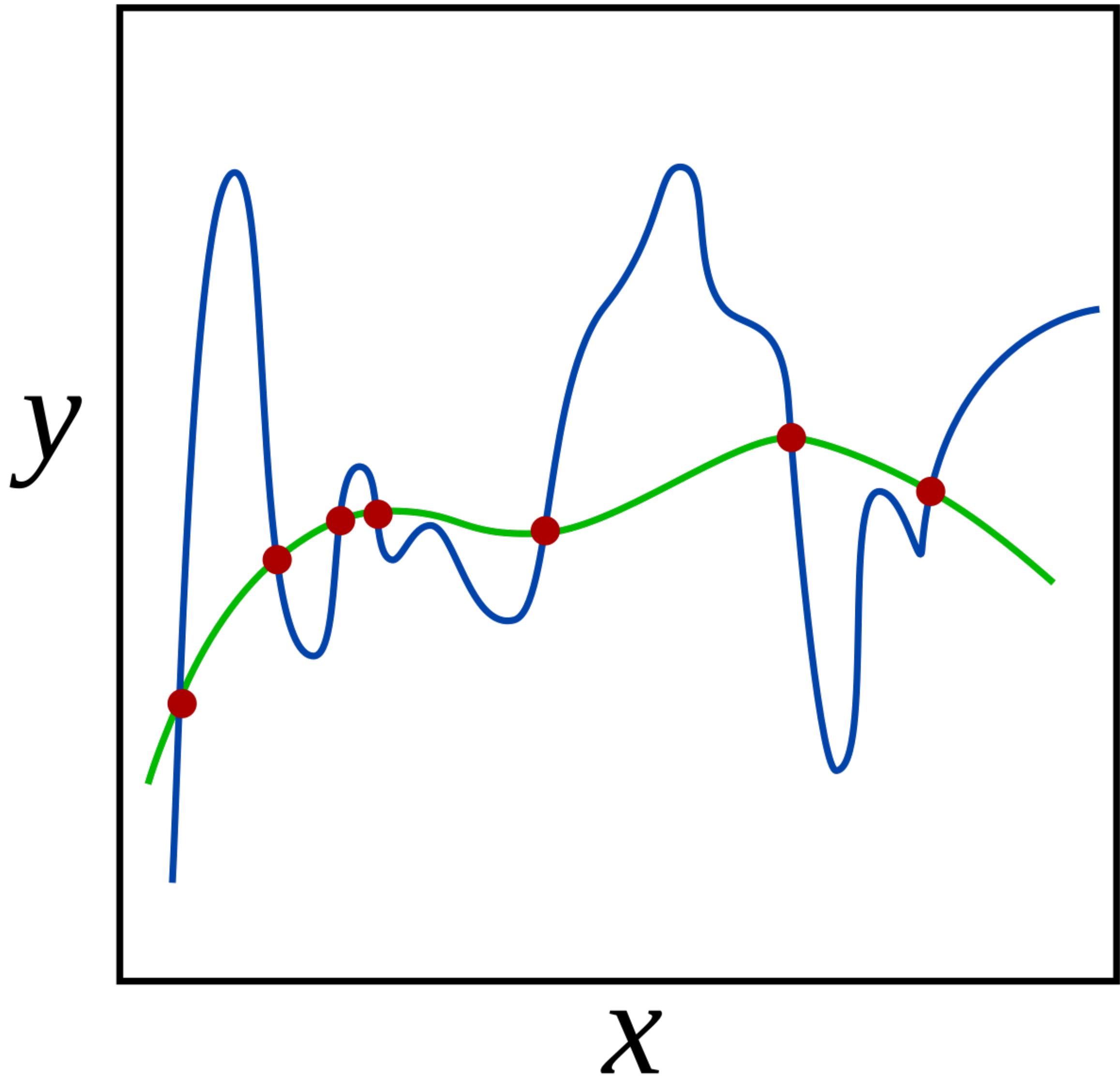
See this [blog](#)



Regularization

Which model may generalize better
to more points drawn from the
underlying unknown distribution?

See this [wiki](#) page



Ridge regression

$$J(\mathbf{w}, b) = \frac{1}{n} \sum_i^n (\mathbf{w}\mathbf{x}_i^T + b - y_i)^2 + \lambda \|\mathbf{w}\|^2$$

$$J(\mathbf{w}, b) \rightarrow \min_{\mathbf{w}, b}$$

Solution

$\mathbf{W} = (\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^T \mathbf{Y}$, where $(\mathbf{X}^T \mathbf{X} + \lambda \mathbf{I})$ is always invertible

- we penalize large weights to correct overfitting
- model is more robust to noise and outliers
- the problem of near-singular $(\mathbf{X}^T \mathbf{X})^{-1}$ is solved
- penalty defines the model's bias-variance tradeoff

Iterative approach to find the best fit parameters

Instead of solving a problem in
one step

$$\theta = (X^T X)^{-1} X^T Y$$

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$

Cost Function

$$J(\Theta_0, \Theta_1) = \frac{1}{2m} \sum_{i=1}^m [h_\Theta(x_i) - y_i]^2$$

↑ True Value
 ↑ Predicted Value

We solve it iteratively**Gradient descent steps**

- 1) (Randomly) initialize weights
- 2) Compute the gradient of the cost function with respect to each parameter (partial derivatives)
- 3) Update the weights of the model by taking steps in the opposite direction of the model
- 4) Repeat steps 2 and 3 iteratively to get the best parameter for the defined model

Gradient Descent

$$\Theta_j = \Theta_j - \alpha \frac{\partial}{\partial \Theta_j} J(\Theta_0, \Theta_1)$$

↑ Learning Rate

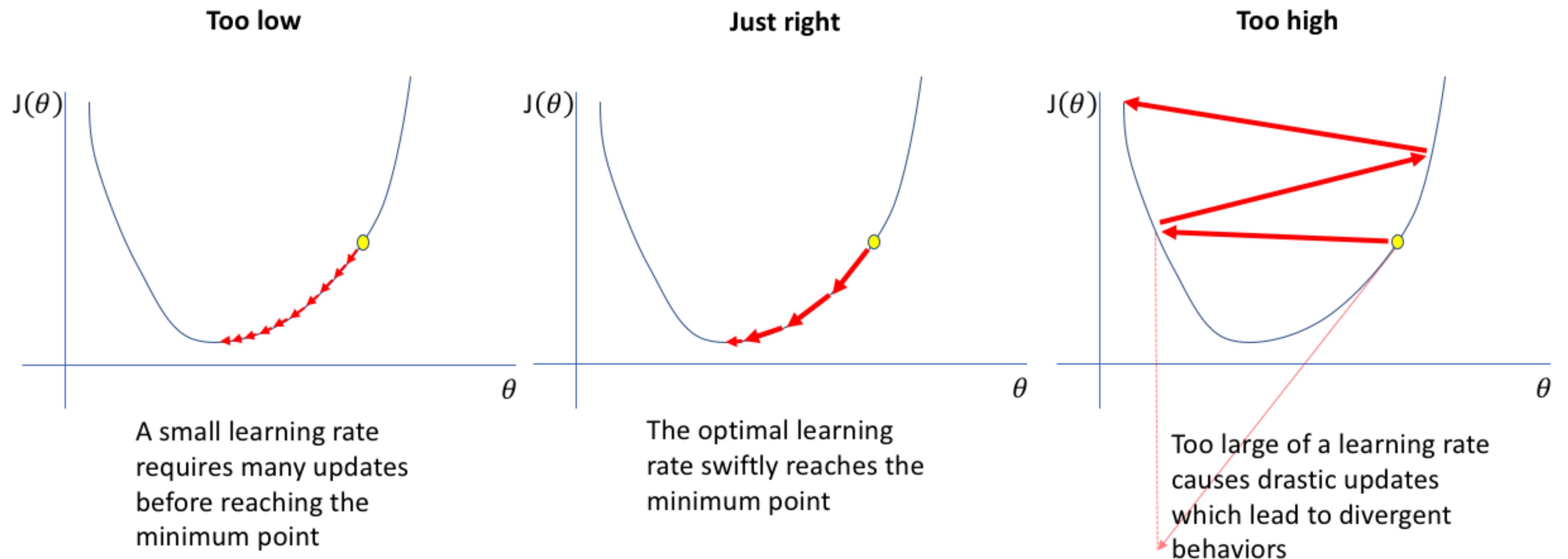
Now,

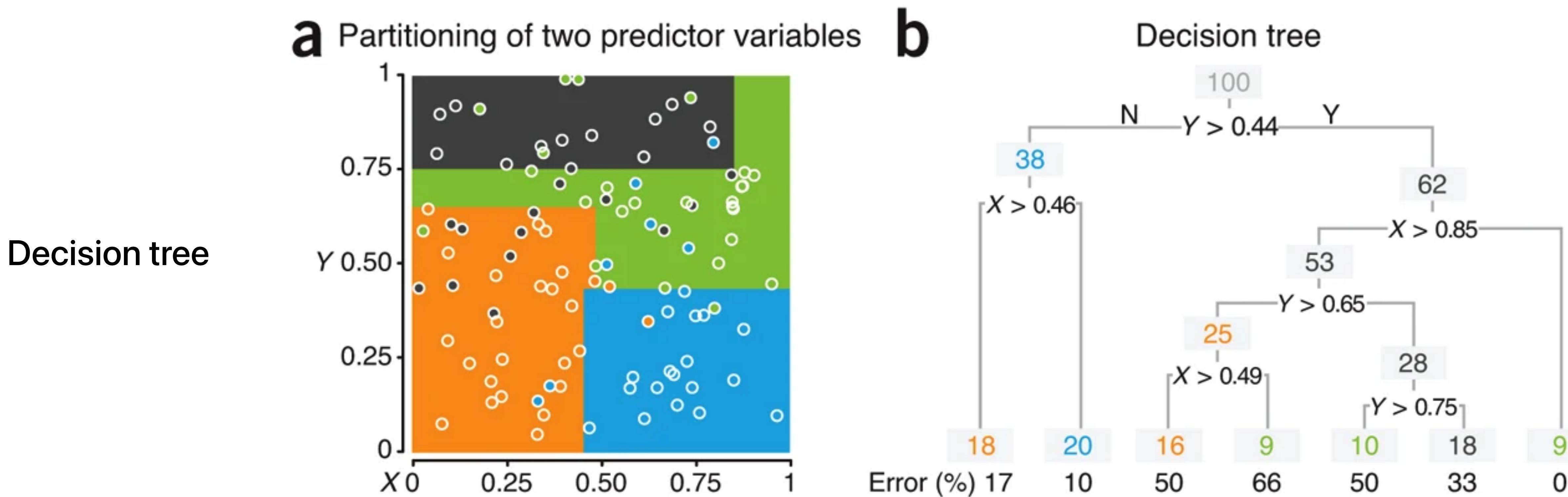
$$\begin{aligned}
 \frac{\partial}{\partial \Theta} J_\Theta &= \frac{\partial}{\partial \Theta} \frac{1}{2m} \sum_{i=1}^m [h_\Theta(x_i) - y_i]^2 \\
 &= \frac{1}{m} \sum_{i=1}^m (h_\Theta(x_i) - y) \frac{\partial}{\partial \Theta_j} (\Theta x_i - y) \\
 &= \frac{1}{m} (h_\Theta(x_i) - y) x_i
 \end{aligned}$$

Therefore,

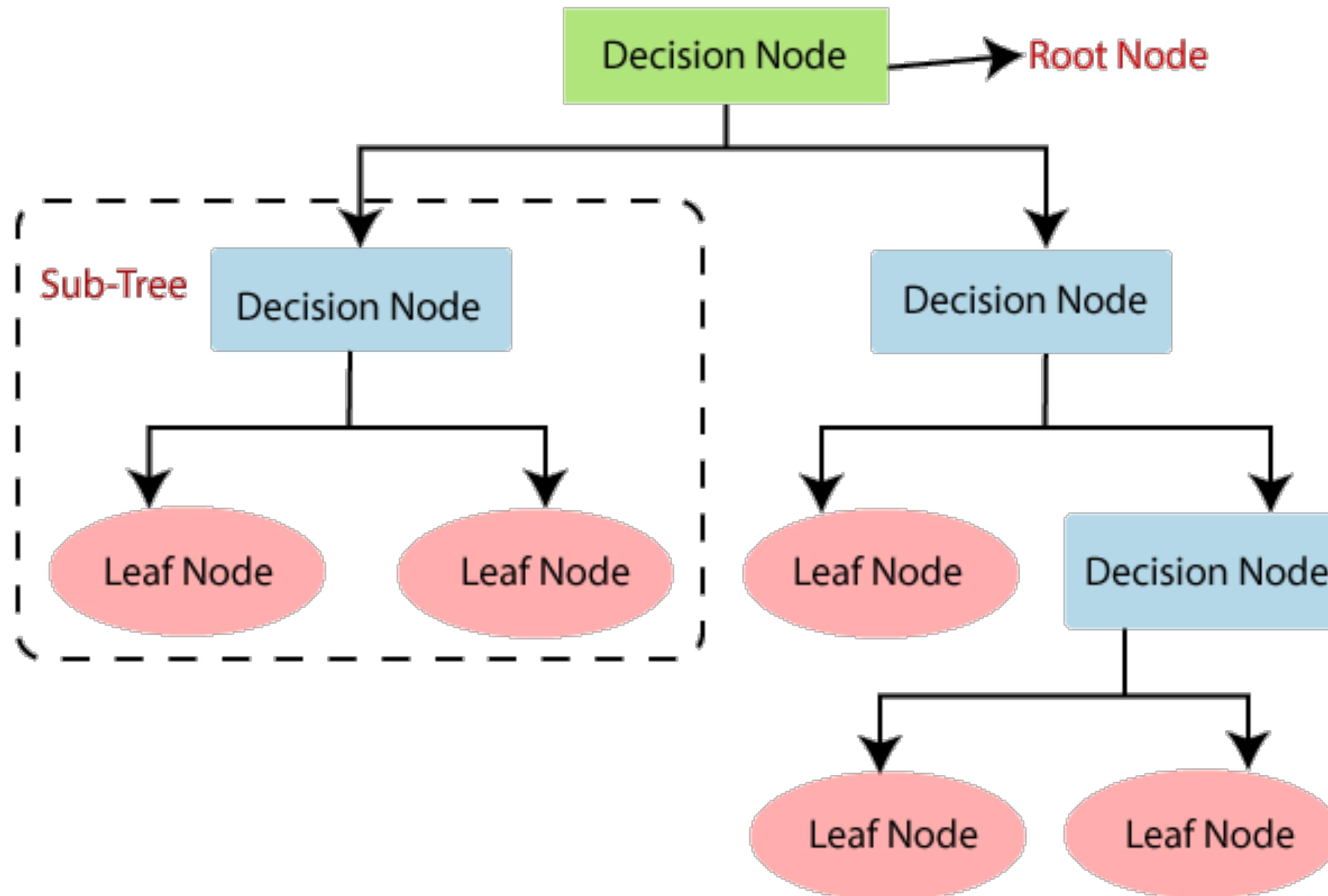
$$\Theta_j := \Theta_j - \frac{\alpha}{m} \sum_{i=1}^m [(h_\Theta(x_i) - y) x_i]$$

**Learning rate
matters**





Attributes



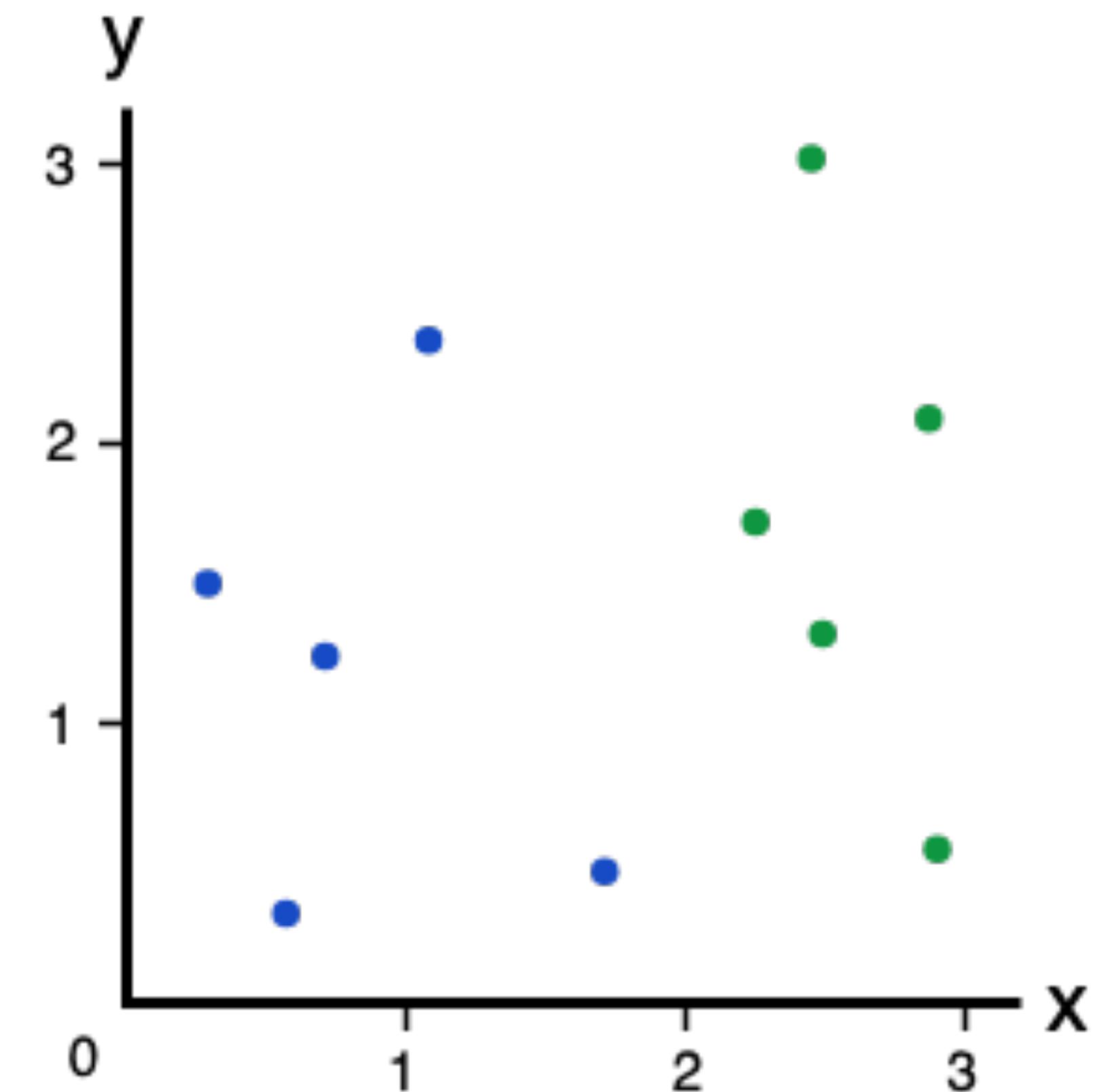
How to select a split?

- Gini

$I_G = 1 - \sum p_i^2$, where p_i - proportion of the samples made up class i for a given node

- Entropy

$E = - \sum p_i \log_2 p_i$ for all p_i ! = 0 if all samples at the node are from the same class $I_H = 0$



Before split: 5 blue samples and 5 green samples

$$E = - \sum p_i \log_2 p_i = - (0.5 \log_2 0.5 + 0.5 \log_2 0.5) = 1$$

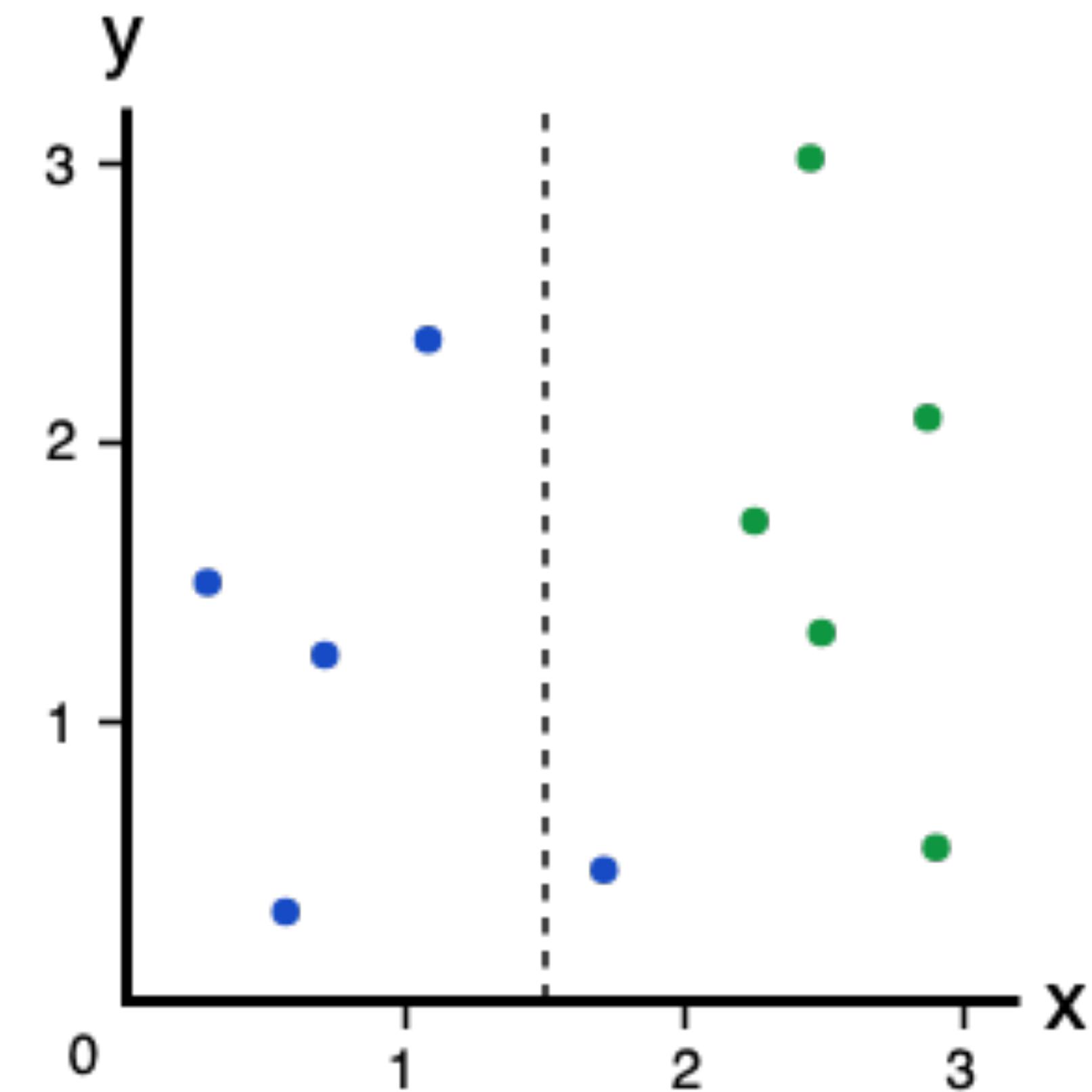
After split at $X = 1.5$

Left branch: $E_{left} = - \log_2 1 = 0$ (all samples are blue)

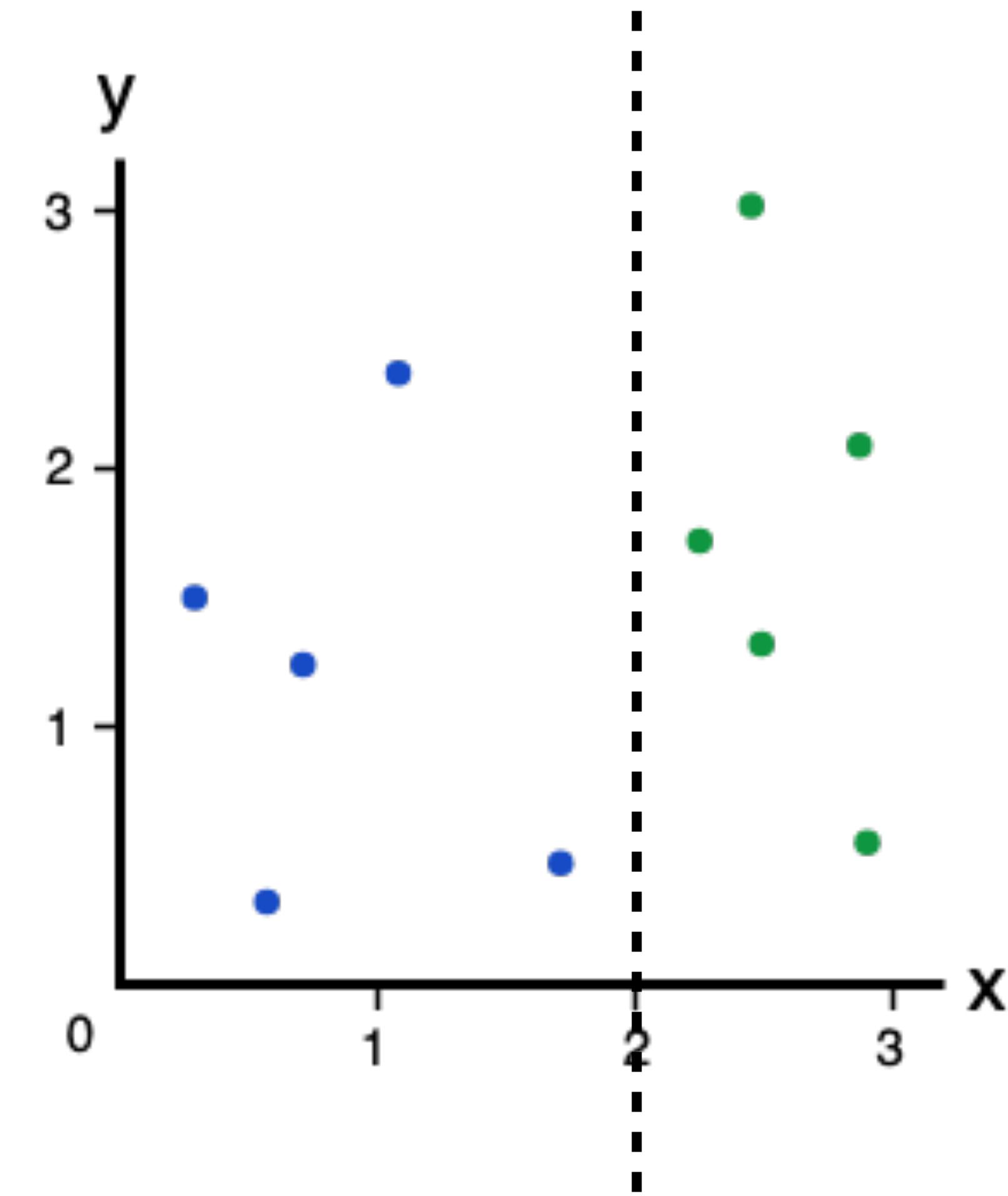
$$\text{Right branch: } E_{right} = \frac{1}{6} \log_2 \frac{1}{6} + \frac{5}{6} \log_2 \frac{5}{6} = 0.65$$

$$E_{split} = 0.4 * 0 + 0.6 * 0.65 = 0.39$$

$$\text{Information gain} = 1 - 0.39 = 0.61$$

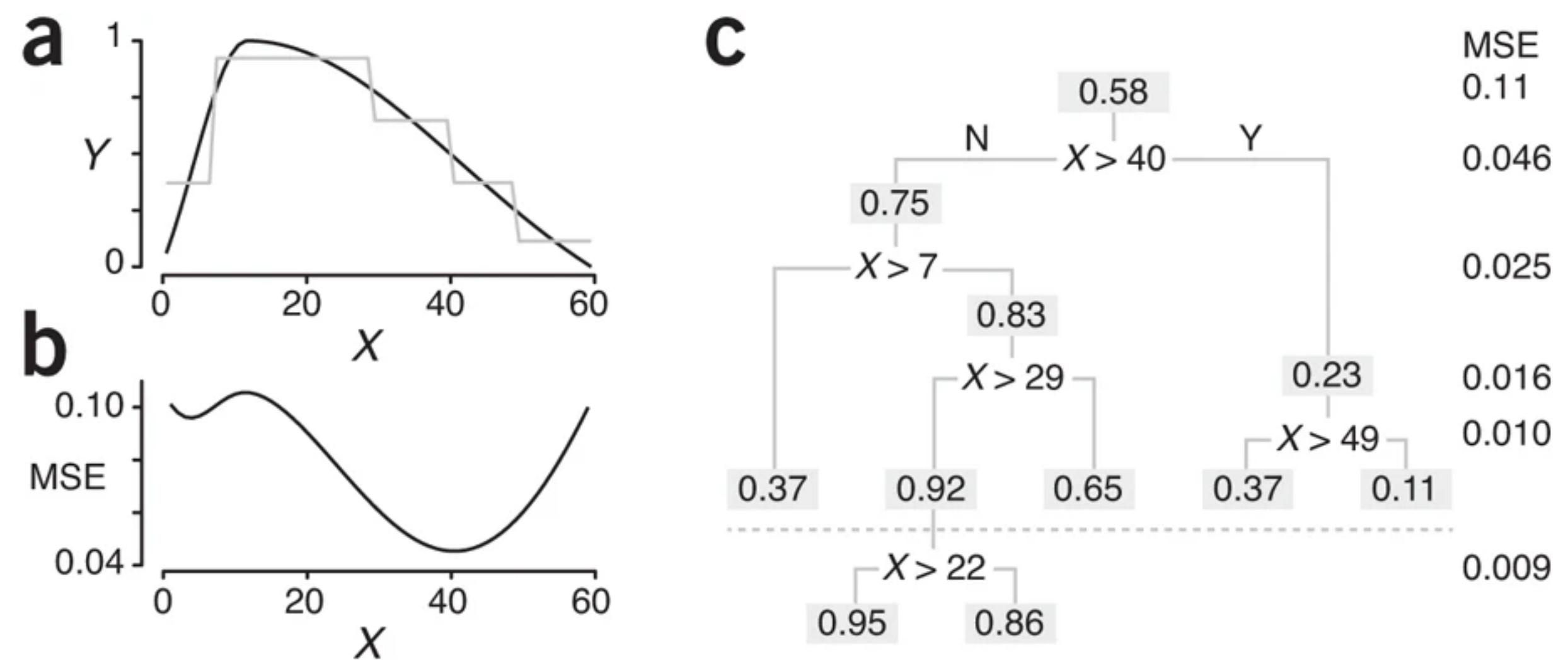


What will be the information gain for split at $X = 2$?



Regression with decision trees

- Piecewise constant approximation



Pros

- Easy to interpret
- Categorical values can be used
- Solves non-linear problems

Cons

- Easy to overfit
- Computationally expensive

Random forest (Ensemble of weak learners)

Given n samples $\{X_i, y\}$

For b in $1..B$:

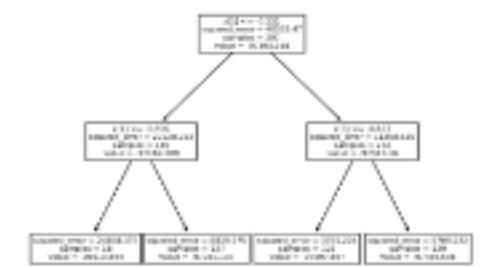
- Bootstrap (duplicates are possible) n samples from the dataset
- Fit h_b decision tree on a randomly sampled set of features
- Predict unknown label as a mean prediction of B models

$$\hat{y} = \frac{\sum h_b(x'_b)}{B} - \text{prediction}$$

Random Sampling with Replacement

$$\mathcal{X} \in \mathbb{R}^{250 \times 100} \quad \left\{ \begin{array}{l} \mathcal{X}_1 \in \mathbb{R}^{60 \times 100} \\ \mathcal{X}_2 \in \mathbb{R}^{60 \times 100} \\ \vdots \\ \mathcal{X}_n \in \mathbb{R}^{60 \times 100} \end{array} \right.$$

Train



Aggregate

Pros

- Less prone to overfit
- Can compute feature importance (next lecture)

Cons

- High computational cost

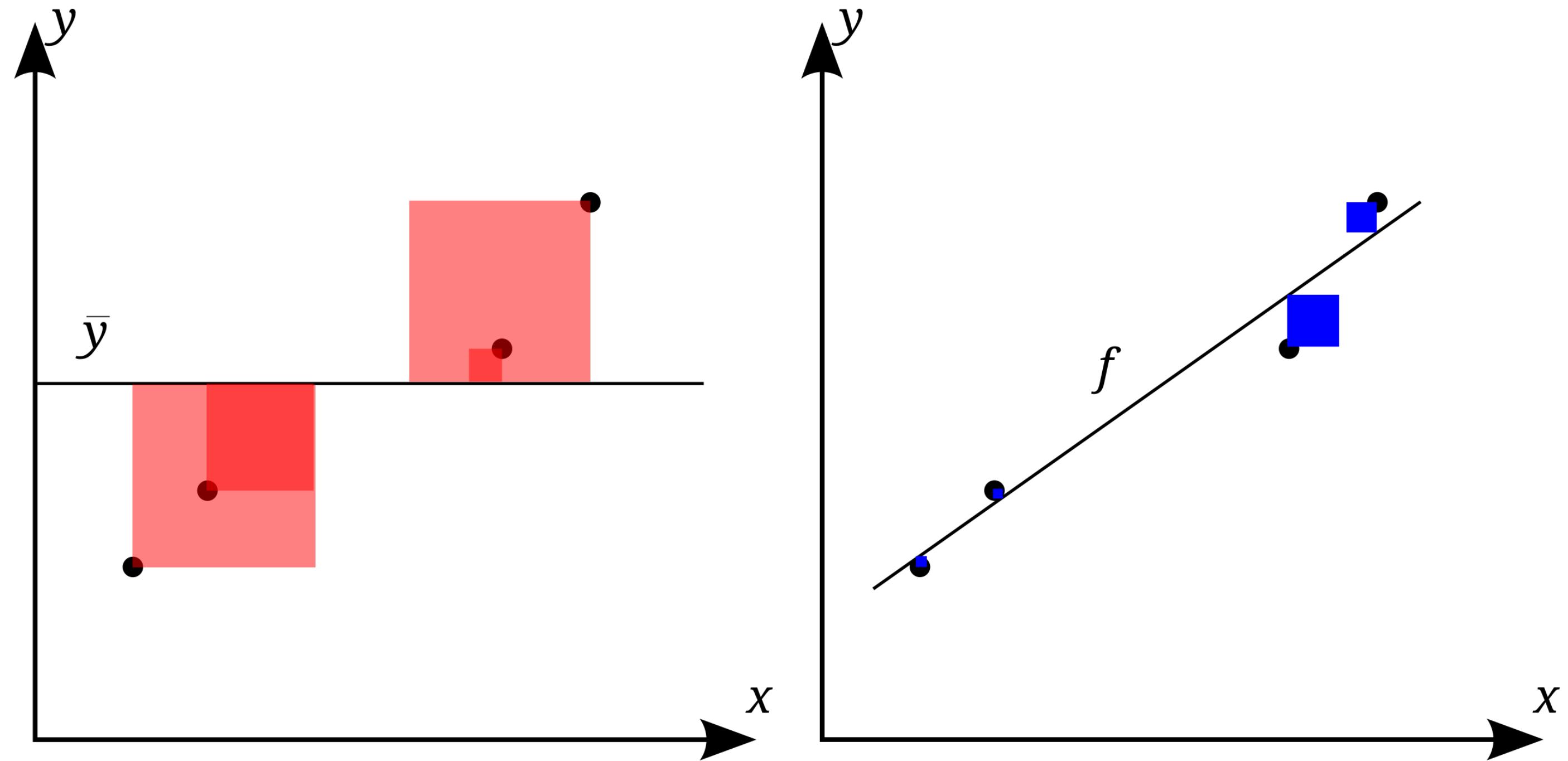
Regression model evaluation (metrics)

- Mean Absolute Error - MAE
- Root Mean Squared Error - RMSE
- Coefficient of determination - R² (R-squared)
- R_p
- ...

R² score

...is a statistical measure of how well the regression predictions approximate the real data points. An **R²** of 1 indicates that the regression predictions perfectly fit the data.

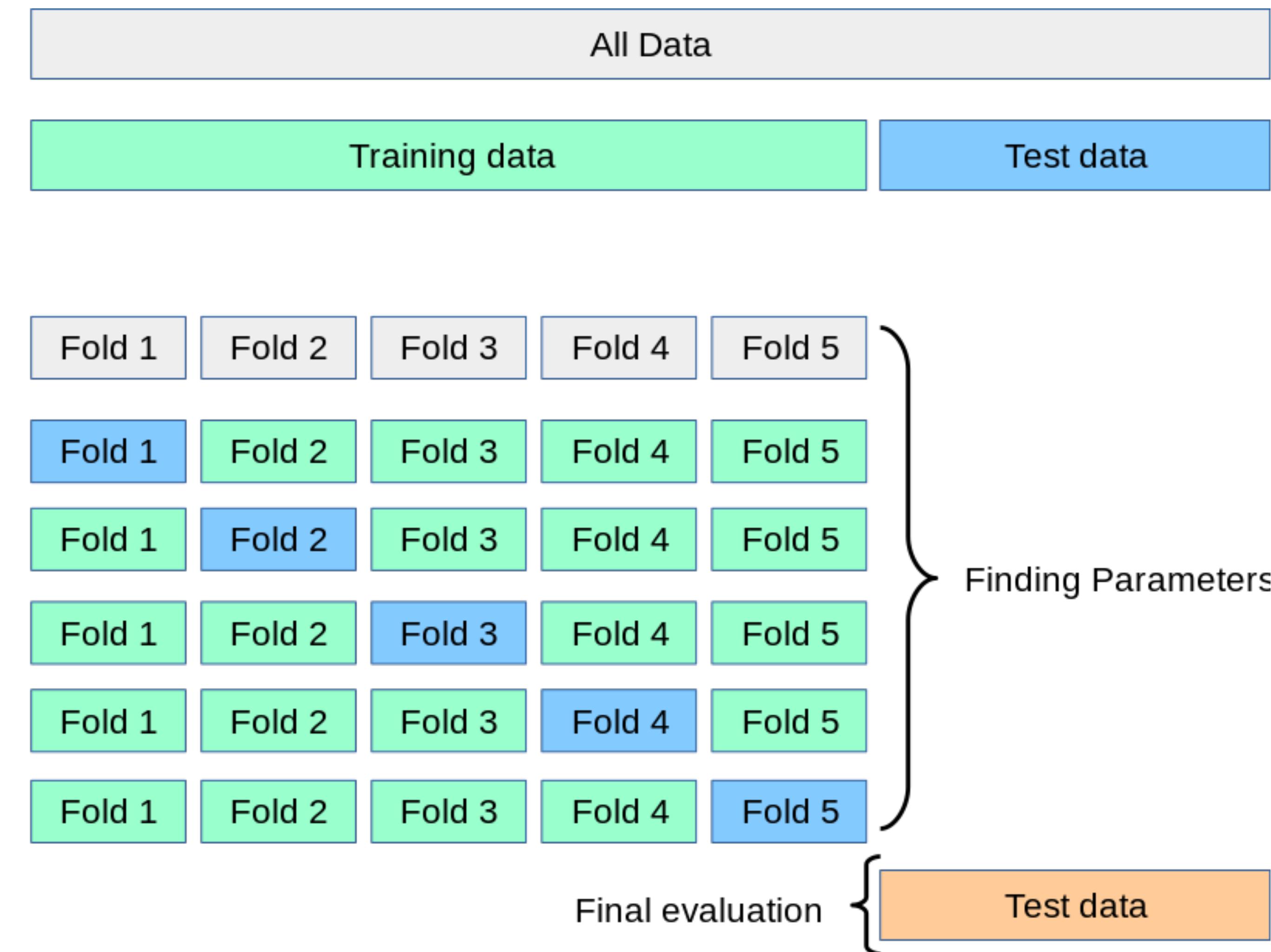
wiki



$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$$

Training/validation/test splits

- Training
 - Fit the model
- Validation
 - Tune hyperparameters
- Test
 - Get final scores (evaluate the model)



Hyperparameters?

- During the training (learning) process we find the best **parameters** (weights) of our model
- In addition we have **hyperparameters** which are usually fixed before the actual training process begins

Example

- Number of trees in random forest, or depth of a decision tree
- λ in Ridge Regression
- They control the learning process itself rather than being learned from the data.

These parameters should be tuned to get better performance of the model

When use ML?

Case #1

- I have 10 crystal structures for which I need to precisely calculate the band gap value
- The density functional theory (DFT) calculation takes about 2 days per structure

Should I use ML in this case?

Case #2

- I have 10,000 crystal structures for which I need to calculate the band gap value
- The DFT calculation takes about 10 minutes per structure
- I developed a featurization scheme that calculates features for a crystal structure
 - for further fitting of the model to predict the band gap
 - The featurization takes about 9 minutes per structure
 - I expect that regression model will possess a high accuracy (say, MAE = 0.1 eV)

Should I use ML in this case?

Case #3

- I have 10,000 crystal structures for which I need to calculate the band gap value
- The DFT calculation takes about 10 minutes per structure
- I developed a featurization scheme that calculates features for a crystal structure
 - for further fitting of the model to predict the band gap
 - The featurization takes about 0.1 second per structure
 - I expect that regression model will possess a moderate accuracy (say, MAE = 0.5 eV)

Should I use ML in this case?

About your final projects

Final project (FP) should reflect the ILOs of the course

- Apply python libraries and data science tools to solve materials science problems
- Collect, generate and analyse materials science datasets, including identification of structure-property relationships
- Critically evaluate materials informatics literature

FP description

- Write a 3 page article style report including
 - Introduction
 - Methods
 - Results
 - Discussion
 - Conclusion
 - Bibliography

Prepare a 5 minutes oral presentation

Possible topics

- Dataset collection and analysis (EDA)
- Prediction of some property of materials
 - band gap, hardness, HOMO/LUMO, formation energy, etc.
 - Screening of materials with superior properties (e.g. ionic conductors)
- MLMD study of some process
 - diffusion/adsorption + validation of the interatomic potential used
- Comparative analysis of models (aka benchmark)