Machine Learning for Seismology Workshop

SSA 2019 Annual Meeting

Karianne Bergen (Harvard)
Qingkai Kong (UC Berkeley)
Zefeng Li (Caltech)
Youzuo Lin (LANL)
Maruti Mudunuru (LANL)
Daniel Trugman (LANL)

Workshop Outline & Schedule

12:30 – 1:15	Machine Learning Workflow – *Karianne*
1:15 – 2:00	Clustering – *Daniel*
2:00 – 2:30	Feature engineering & selection – *Maruti*
2:30 – 2:40	Break
2:40 – 4:00	Supervised learning – *Qingkai & Zefeng*
4:00 – 4:30	Introduction to deep learning – *Youzuo*

Machine Learning for Seismology

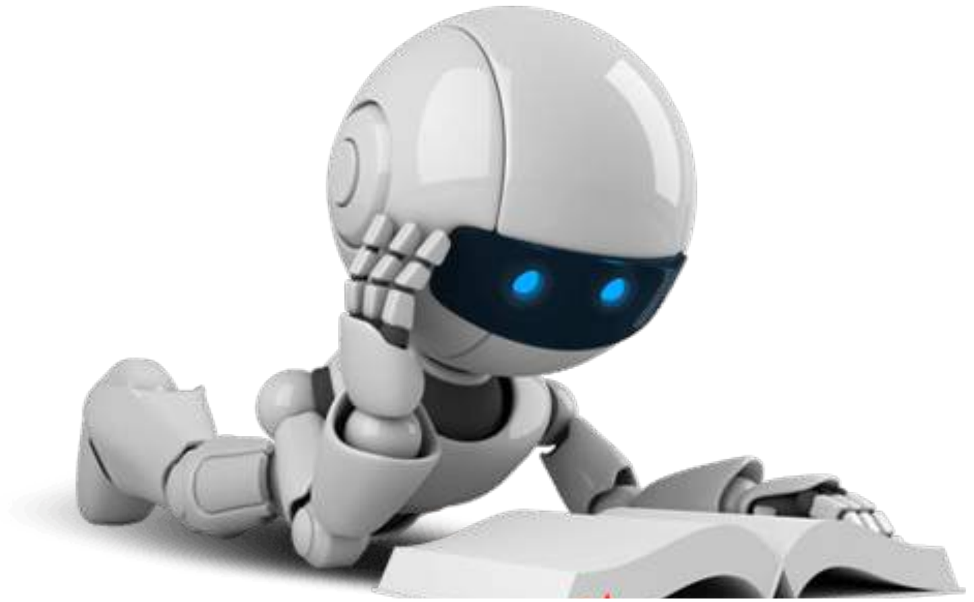
The Machine Learning Workflow

Karianne J. Bergen

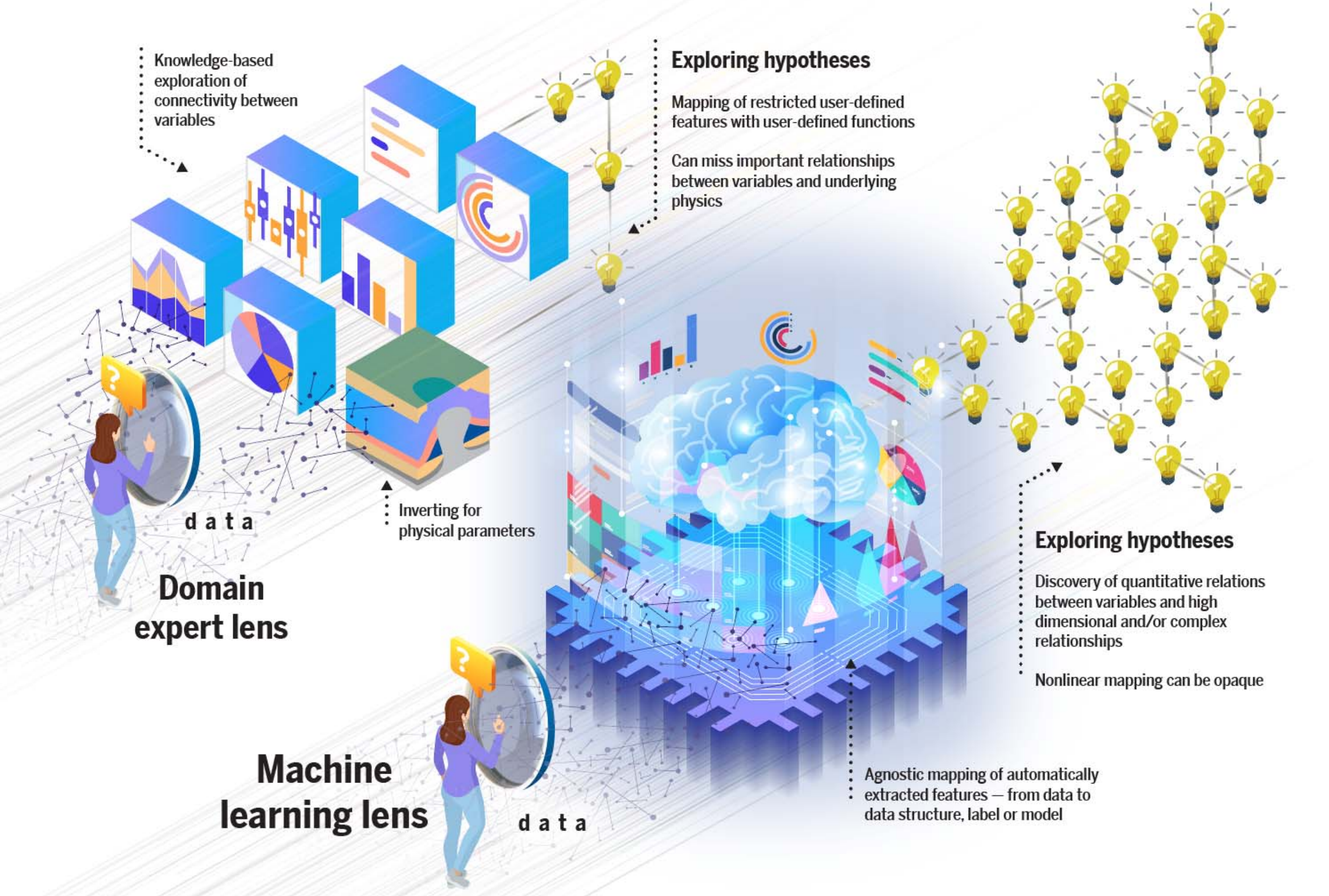
Data Science Initiative Postdoctoral Fellow
Harvard University

What is machine learning (ML)?

A form of data analysis that **automates** model building using algorithms that **improve** their performance at some task **with experience** (data).



Why machine learning?

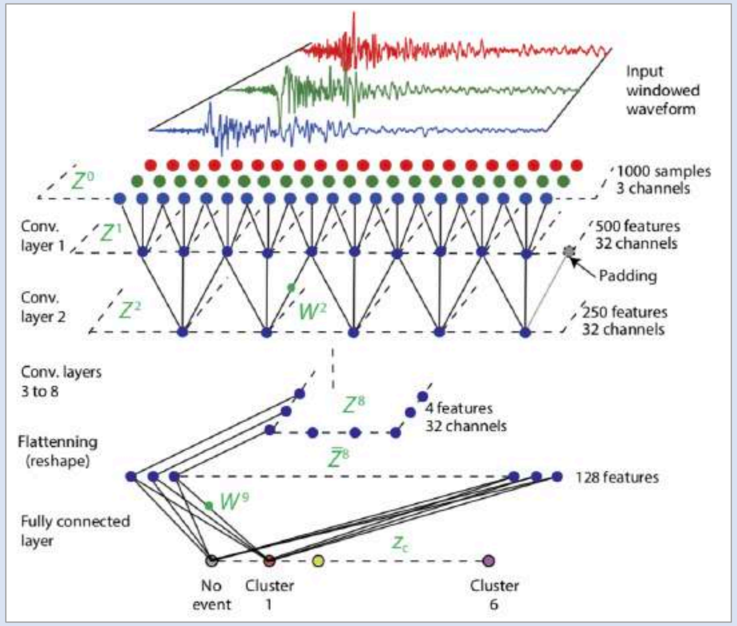


ML can identify patterns in **large, complex data sets** that may be difficult to discover with traditional approaches.

Bergen et al. (2019), Science

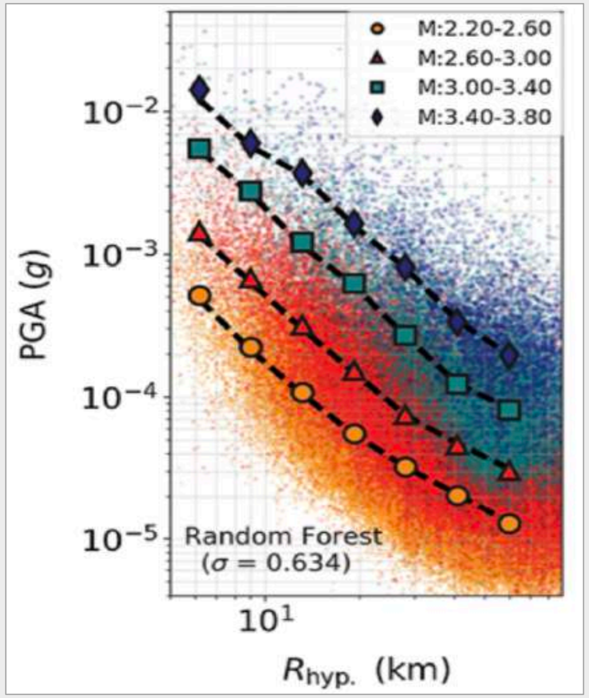
Uses of machine learning in seismology

Automation



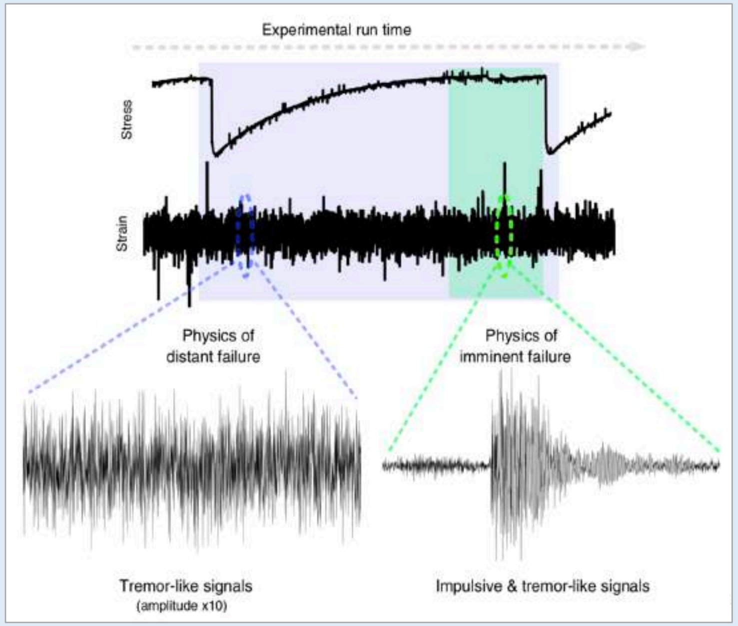
Earthquake detection
with deep neural networks
[Perol et al. (2018)]

Modeling & Inversion



Ground motion prediction
with random forests
[Trugman & Shearer (2018)]

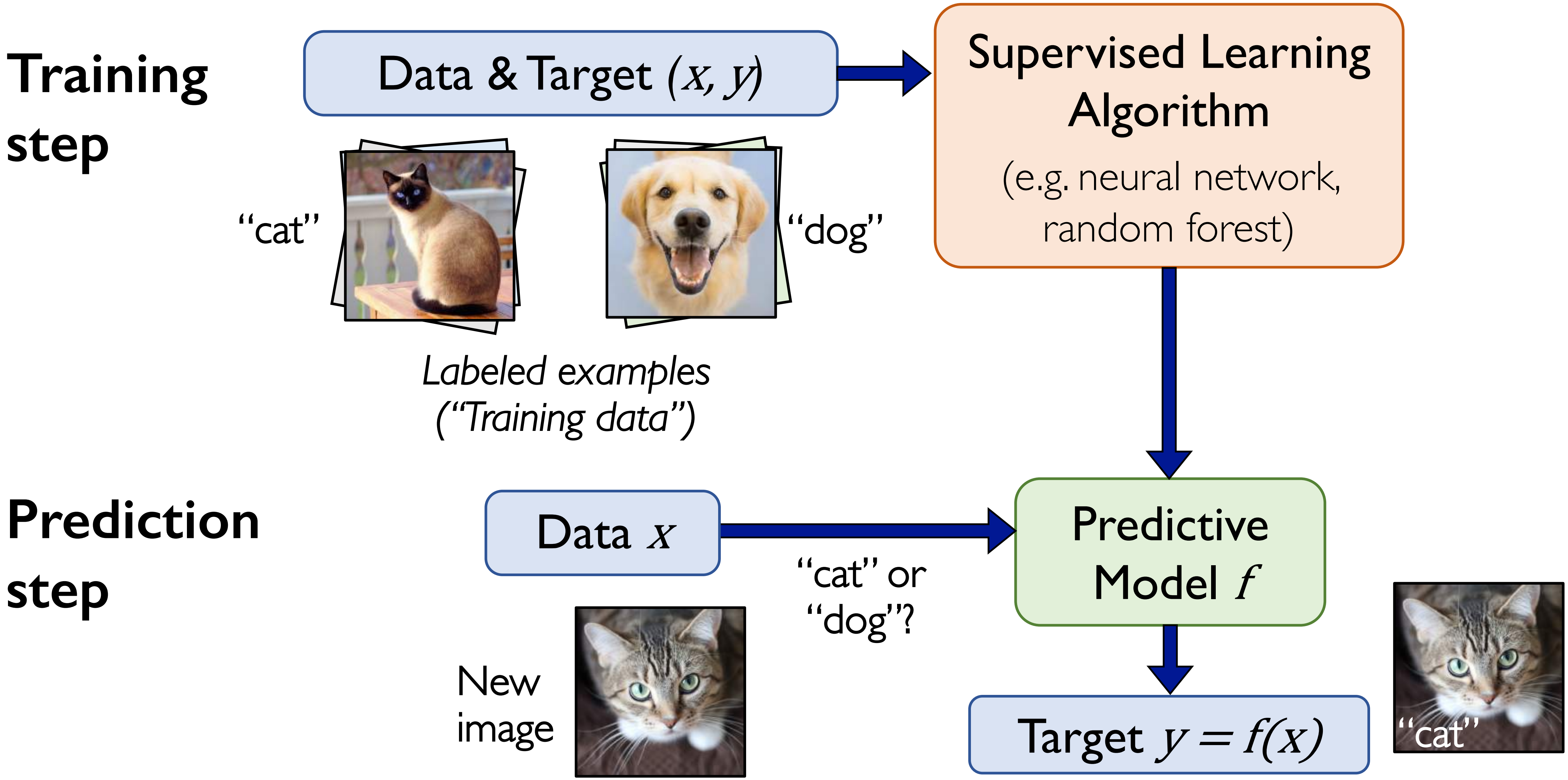
Discovery



Predicting time-to-failure in
"labquakes" with random
forests *[Rouet-Leduc et al. (2017)]*

Recent reviews: *Bergen et al. (2019), Kong et al. (2018)*

Supervised Learning: *Building models from examples*



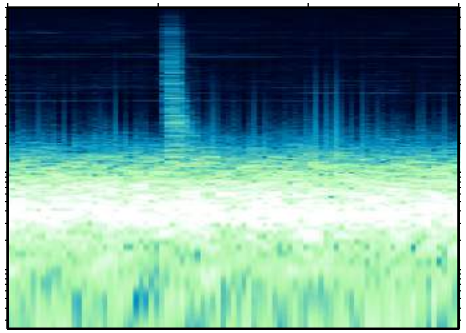
Supervised Learning: *Building models from examples*

Training step

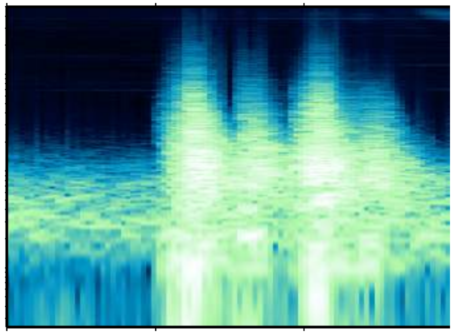
Data & Target (x, y)

Supervised Learning Algorithm

(e.g. neural network, random forest)



“Local Earthquake”



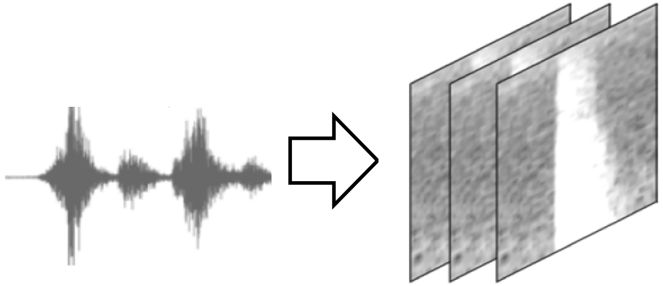
“Slow Earthquake”

Labeled examples
(“Training data”)

Prediction step

Data x

“Local” or “Slow”



Predictive Model f

Target $y = f(x)$

“Slow earthquake”

Nakano et al. (2019), SRL

Unsupervised Learning: *Finding patterns in data*

Data X → Unsupervised Learning Algorithm (e.g. PCA, k-means) → Structure in X

Images only
(no labels)

Common patterns or features in images

OR

Groups of similar images

Basic ML Workflow

- Gathering & cleaning the data
- Representing the data
- Building (training) the model
- Evaluating & improving the model
- Deploying the model

R

python™

PYTORCH

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

TensorFlow

K Keras

scikit learn

Spark MLlib

Basic ML Workflow

- **Gathering & cleaning the data**
- Representing the data
- Building (training) the model
- Evaluating & improving the model
- Deploying the model

Gathering and cleaning the data

Identifying, parsing, & compiling data (from multiple sources)

Determining relevant data attributes

Data quality: missing data & outliers

Standardization: normalization, detrending, etc.

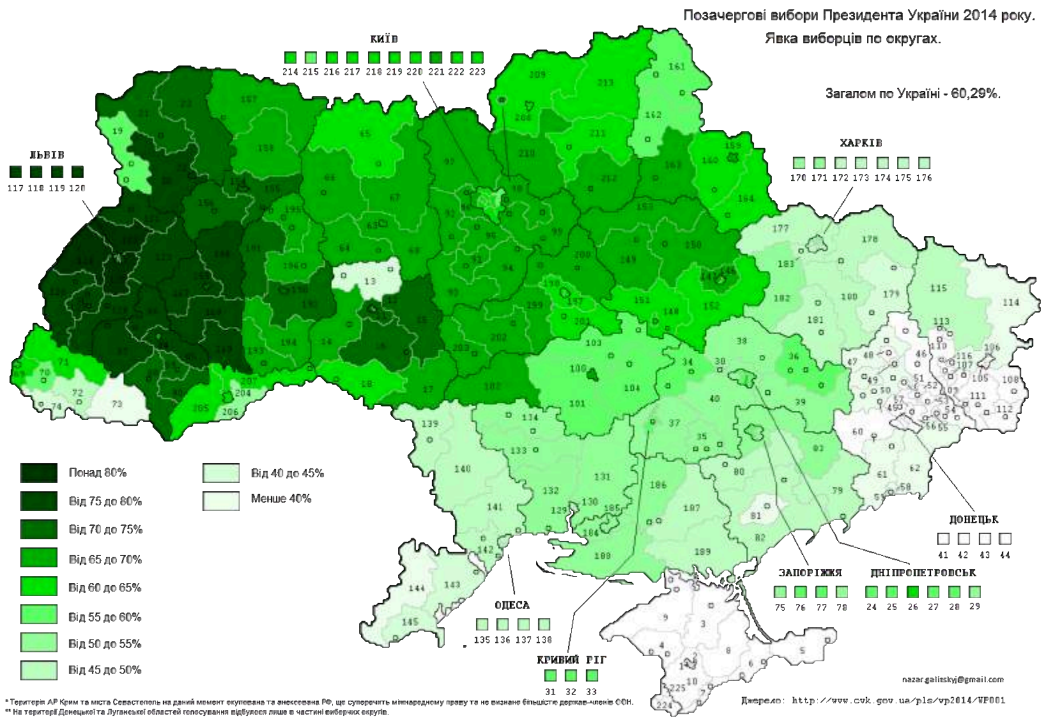
Exploratory data analysis

Split data into training & test sets *(we'll come back to this…)*

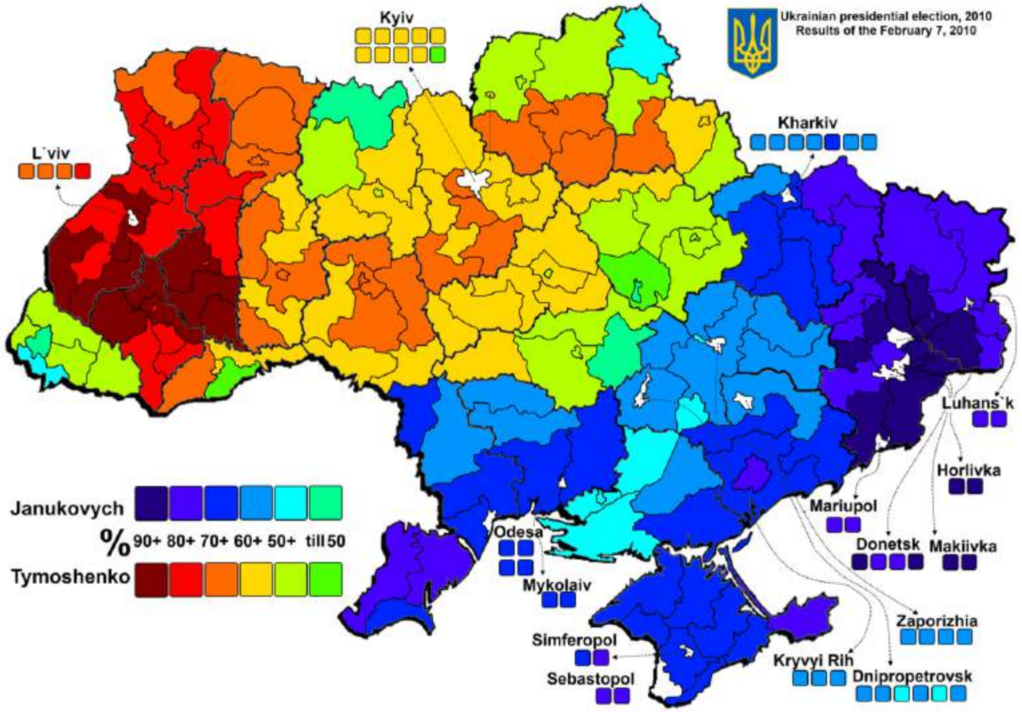
Cleaning the data: *handling missing data*

Some methods can handle missing values, others require imputation

Caution: Is the data *missing at random?*



Voter turn out,
2014 Ukrainian presidential election



Election results,
2010 Ukrainian presidential election

Data leakage in ML

Unintended inclusion of information in training data that should not be available to the predictive model.

Due to data collection, aggregation, or preparation process

1. Combining data from multiple sources
2. Creating artificial dependencies
3. Unintentionally including information from the future

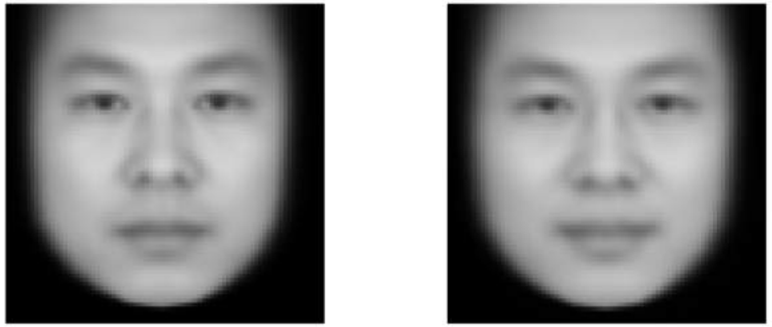
Kaufman et al. (2011) "Leakage in Data Mining: Formulation, Detection and Avoidance"

Data leakage in ML

ID photos of non-criminals are "acquired from Internet using the web spider tool."

ID photos of criminals "are [acquired from] the ministry of public security [or] city police departments in China."

"We find some discriminating structural features for predicting criminality, such as **lip curvature**…"



Wu & Zhang (2016), arXiv:1611.04135

Basic ML Workflow

- Gathering & cleaning the data
- **Representing the data**
- Building (training) the model
- Evaluating & improving the model
- Deploying the model

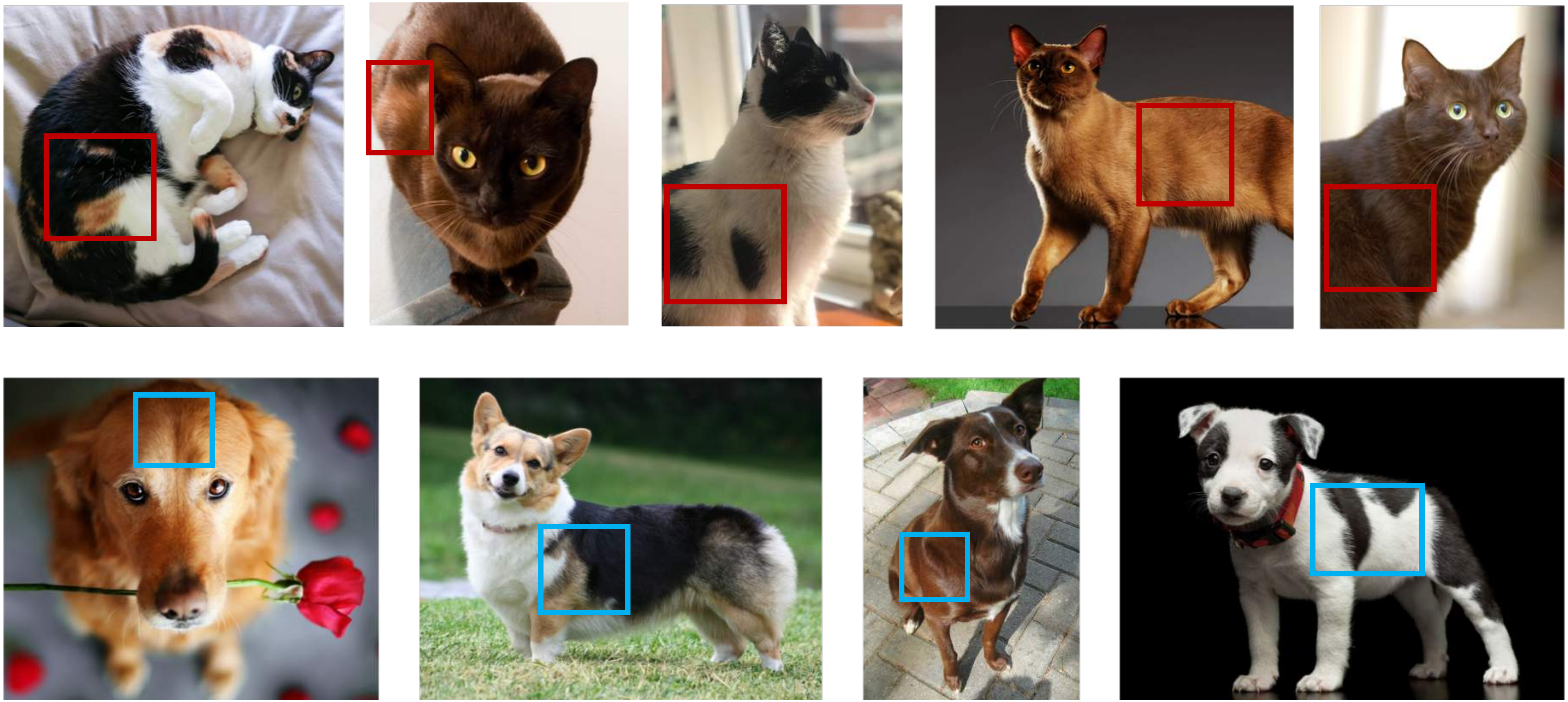
Feature extraction & feature selection

Data representation important!

> *"At the end of the day, some ML projects succeed and some fail.*
> *What makes the difference?*
> *Easily the most important factor is the features used."*
>
> *– P. Domingos (2011)*

Data representation is important!

Data representation is important!



Feature extraction & feature selection

Data representation important!

Curse of dimensionality

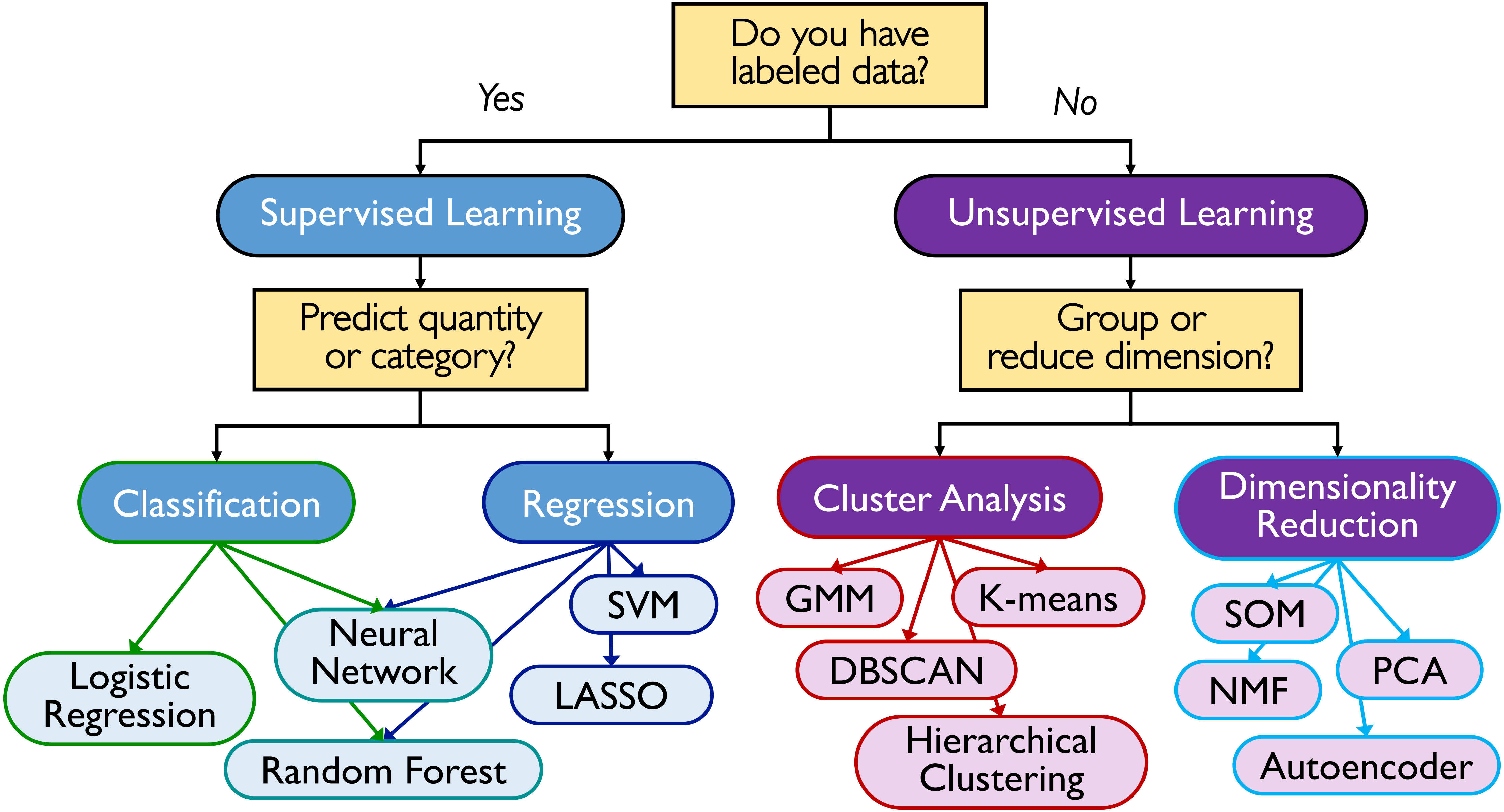
- Feature selection
- Dimensionality reduction

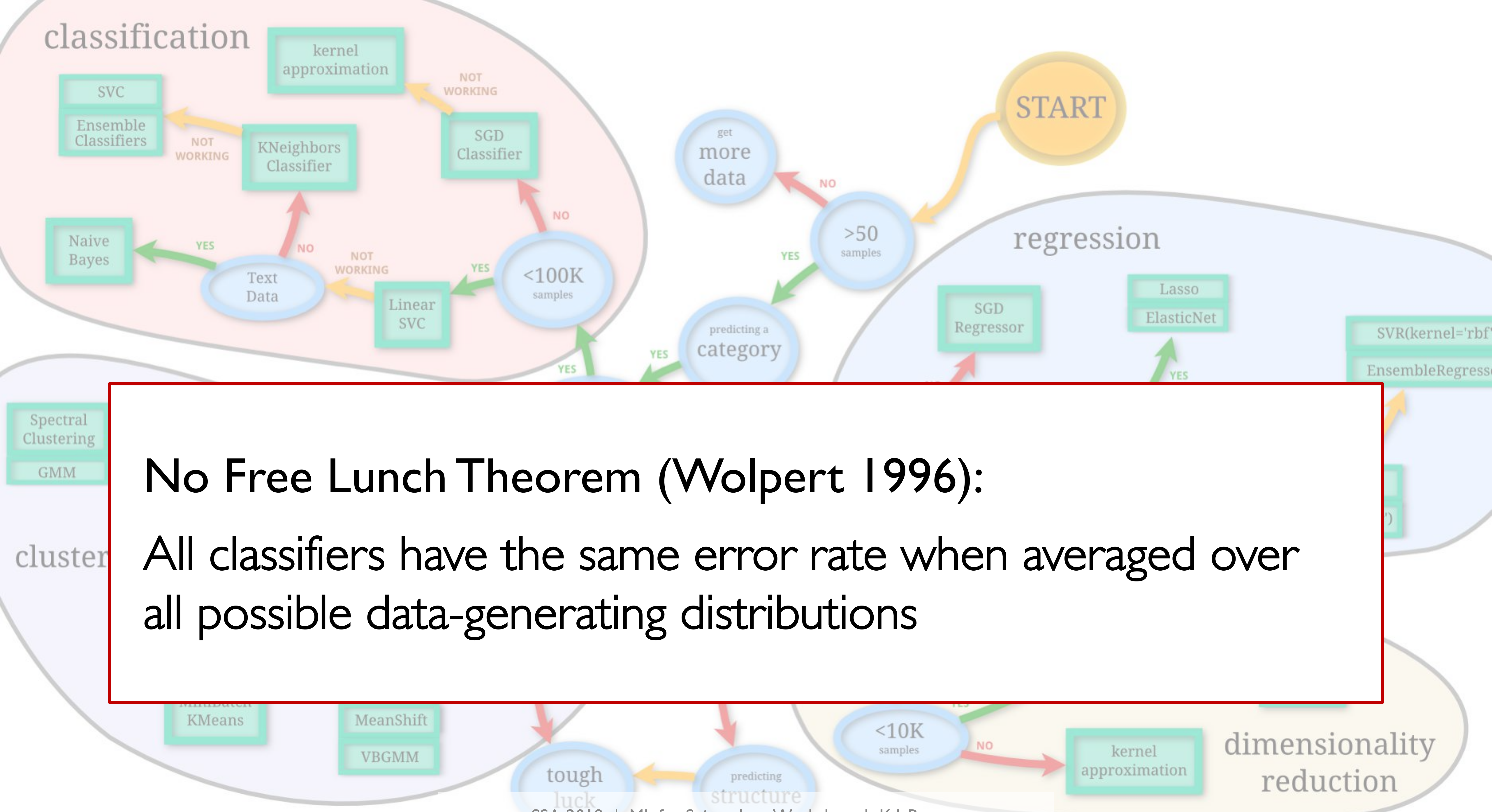
Automatic representation learning & feature extraction in deep neural networks

Basic ML Workflow

- Gathering & cleaning the data
- Representing the data
- **Building (training) the model**
- Evaluating & improving the model
- Deploying the model

Common ML Methods





No Free Lunch Theorem (Wolpert 1996):

All classifiers have the same error rate when averaged over all possible data-generating distributions

Choosing a ML model (and loss function)

Are labeled data available?

What is your prediction target?

Properties of your data?

How big is your data set?

Computational concerns?

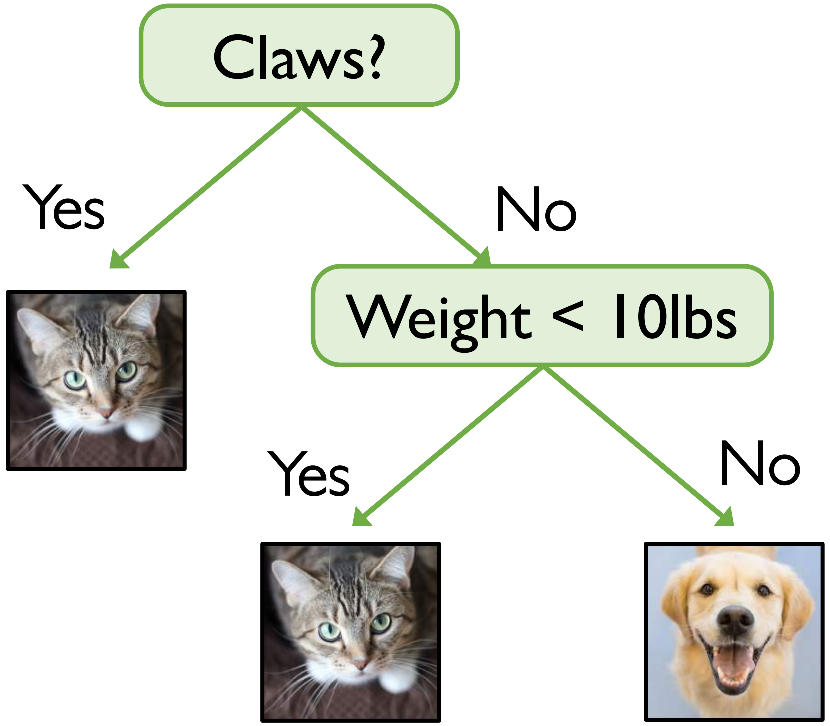
Interpretability vs. prediction?

General strategy: Start with a simple model, move toward more complex models only as necessary

Decision tree: strengths and limitations

Uses a series of conditions on input features to make predictions,
Can be represented as binary tree structure

- ✓ Regression or classification
- ✓ Non-linear model, feature interactions
- ✓ Interpretable

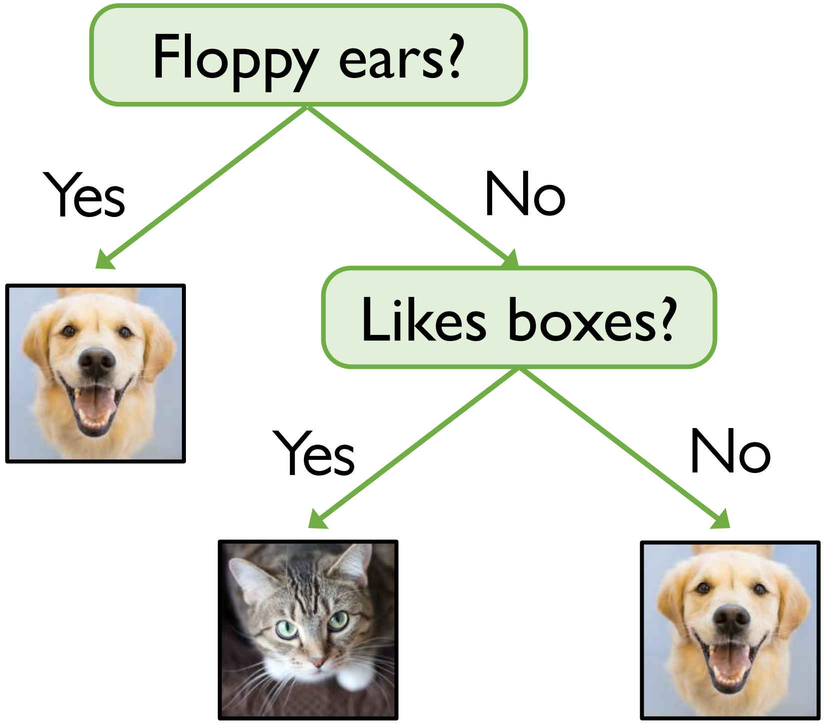


Decision tree: strengths and limitations

Uses a series of conditions on input features to make predictions,
Can be represented as binary tree structure

- ✓ Regression or classification
- ✓ Non-linear model, feature interactions
- ✓ Interpretable

- ✗ Unstable (high variance)
- ✗ Overfitting (node purity)
- ✗ Poor model for continuous attributes



General advice: Study the ML model/algorithm before you use it!

Training (“fitting”) the model

Machine learning method → a class of models (functions)

Training the model: use data to find “best” model within class

Linear regression

- Set of possible models: linear functions

$$\hat{y} = \hat{f}(x; \beta) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

- Which linear model?

  Model that minimizes the mean squared error (loss) on the training data.

$$\min_{\beta} \left\| Y - \hat{f}(X; \beta) \right\|^2$$

Basic ML Workflow

- Gathering & cleaning the data
- Representing the data
- Building (training) the model
- **Evaluating & improving the model**
- Deploying the model

Evaluating and Improving the Model

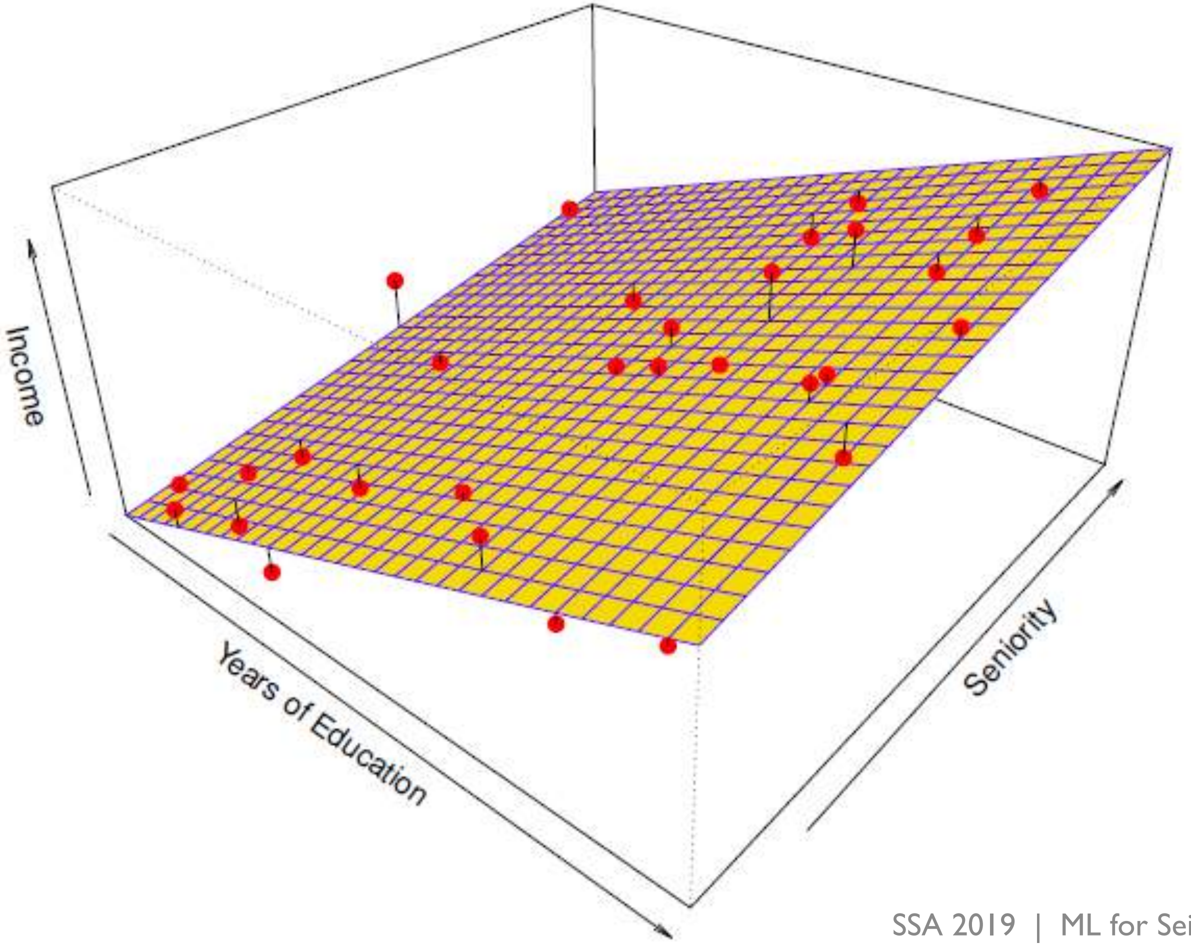
Goal: use training data and ML algorithm to learn a predictive model that *generalizes*.

(has high prediction accuracy on previously unseen data)

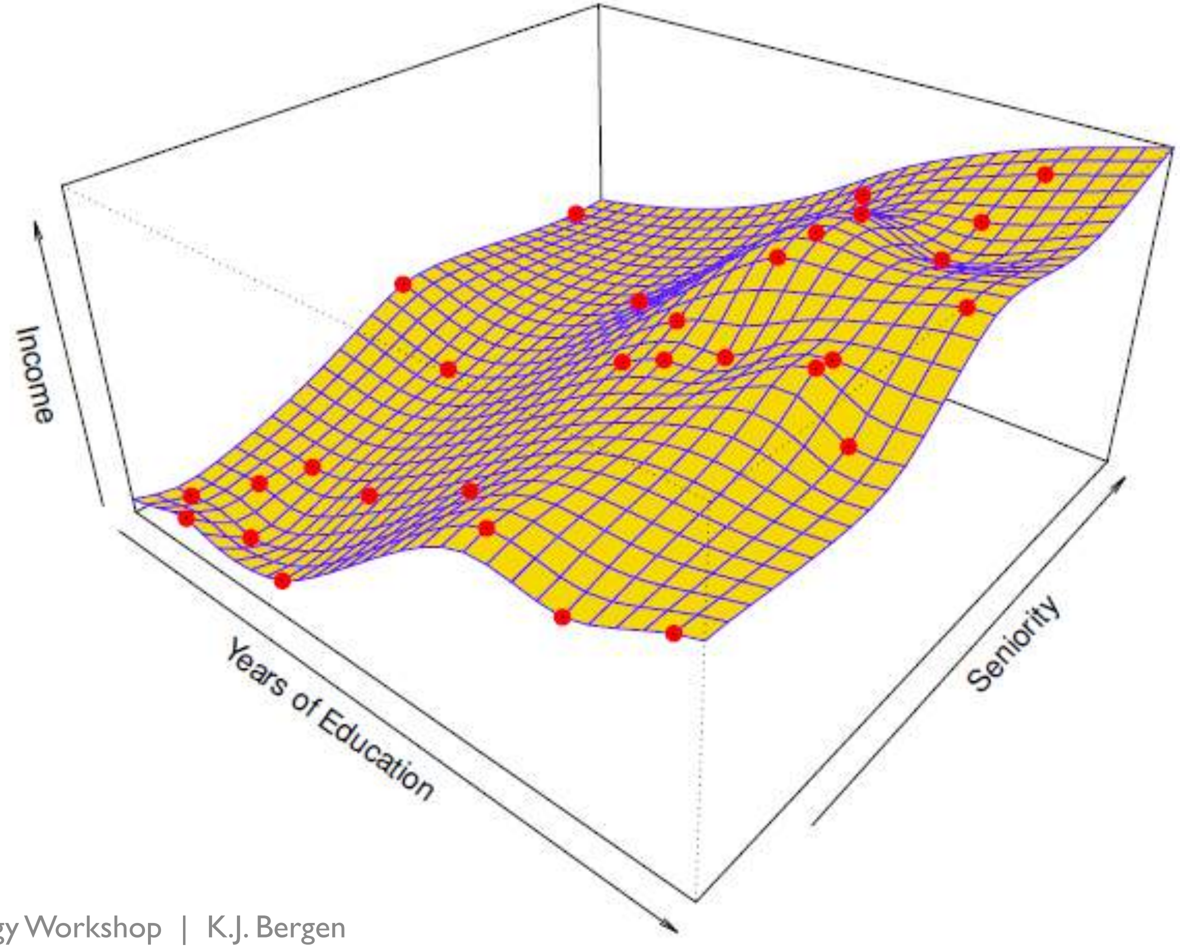
Challenge for generalization: overfitting

Figures 2.4 & 2.6, ISL (2013)

ML model fits the underlying trend.
learns the "signal."

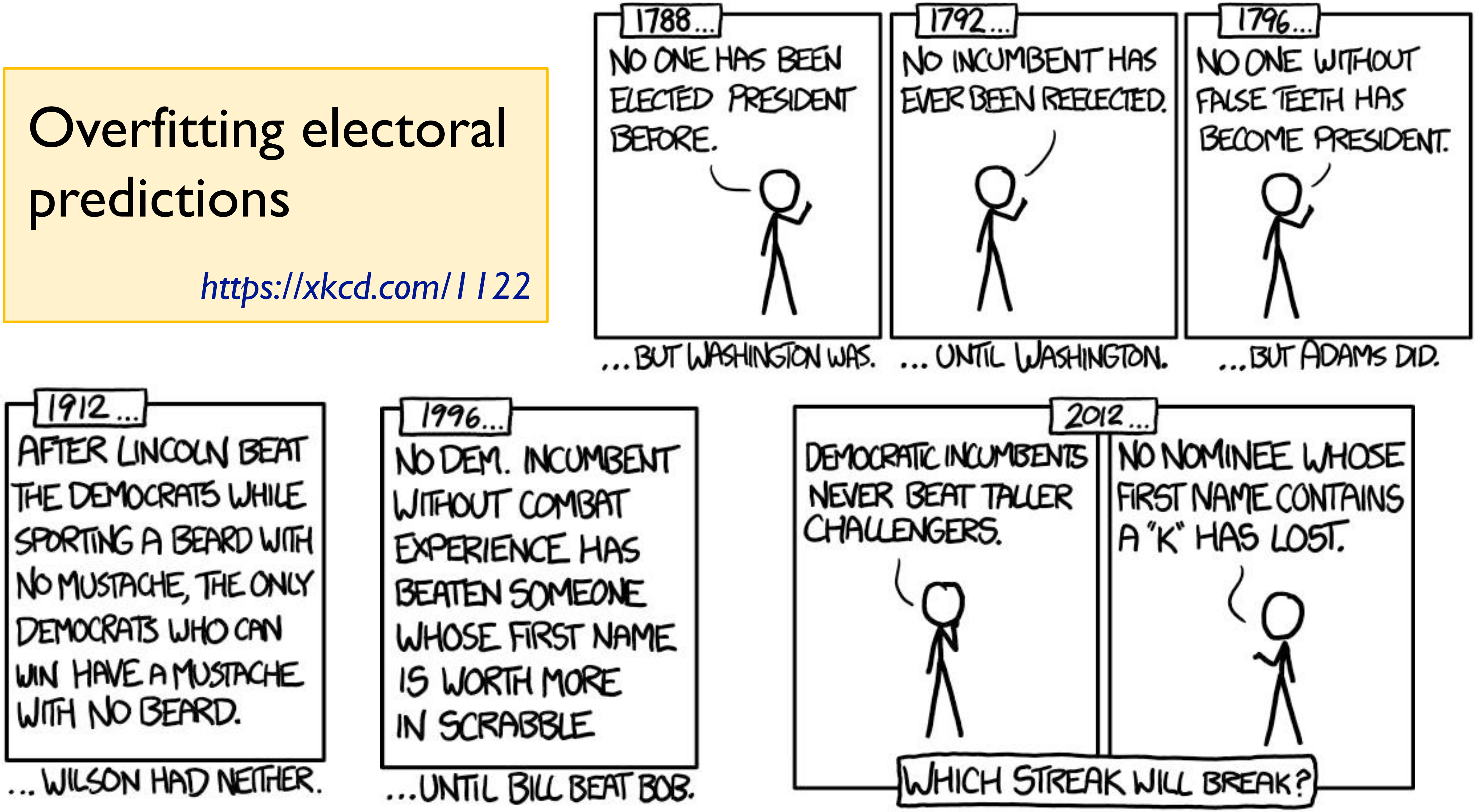


ML model overfits the training data.
learns the "noise."



Overfitting electoral predictions

https://xkcd.com/1122



Generalization error & the Bias-Variance trade-off

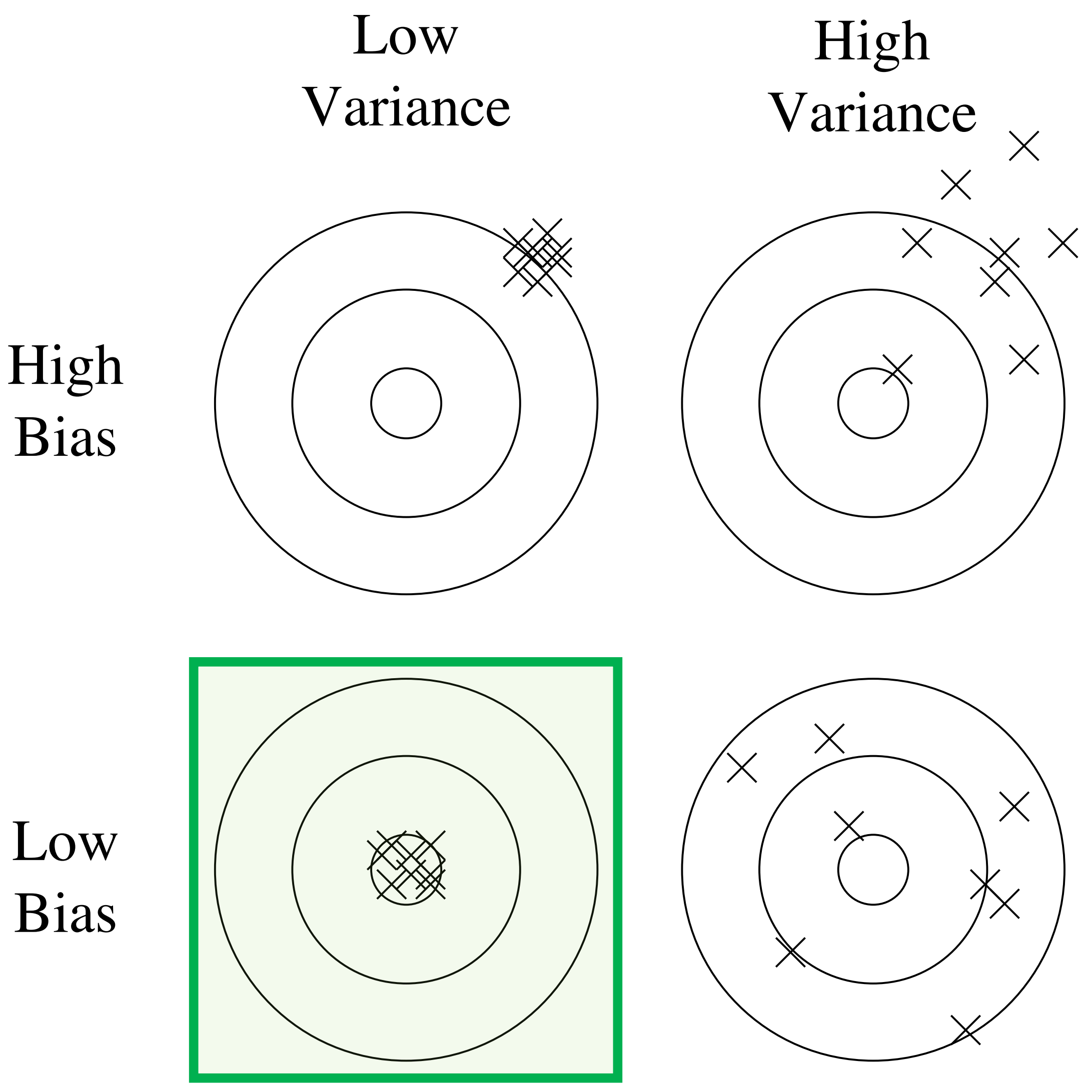
Bias

Error due to model misspecification (oversimplification)

Variance

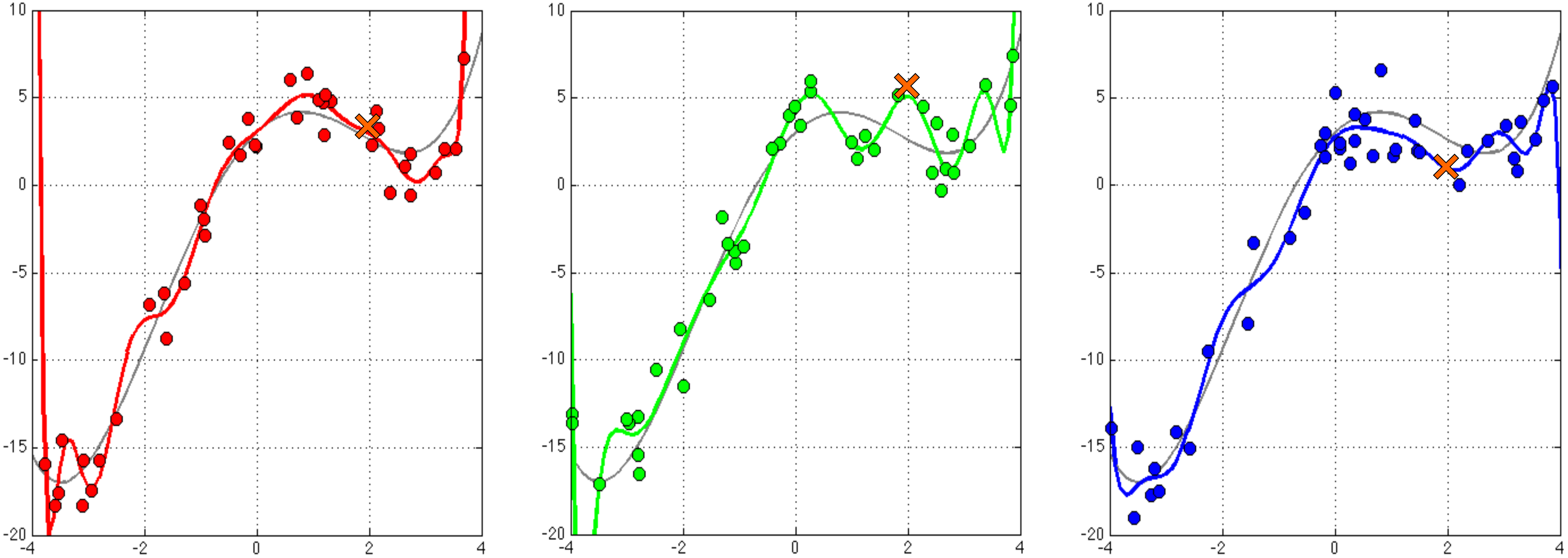
Error due to sensitivity of model to fluctuations in the training data

Irreducible error



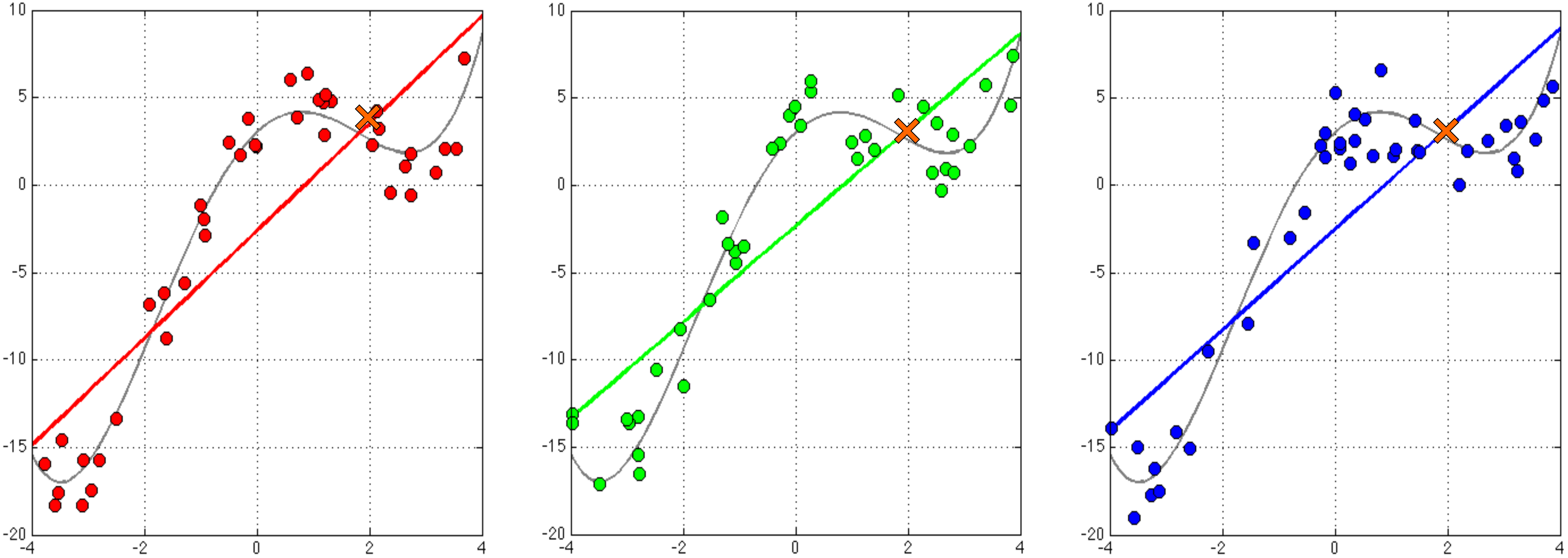
Domingos (2011)

Overfitting: High variance



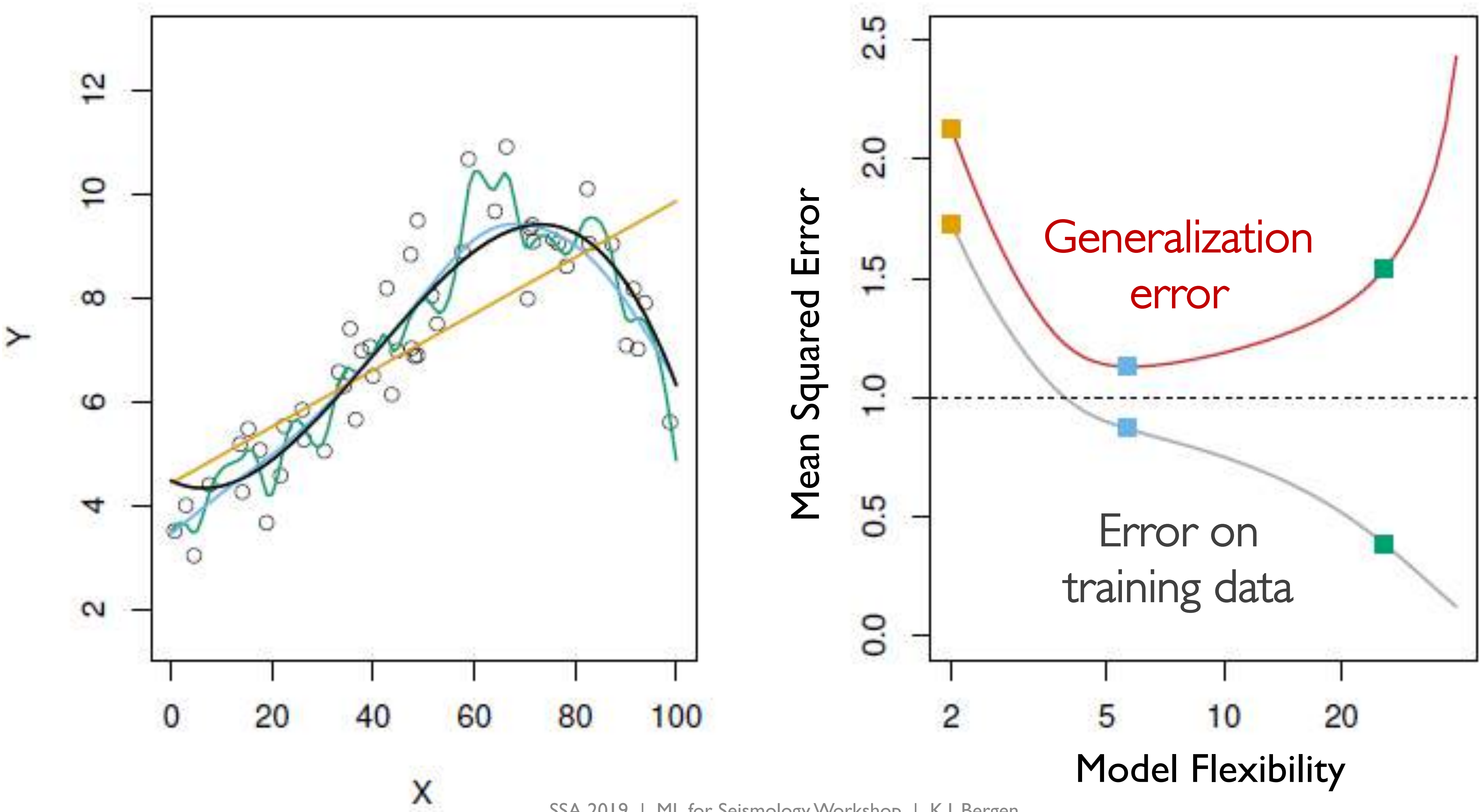
Training error < Generalization error

Underfitting: High bias (& low variance)

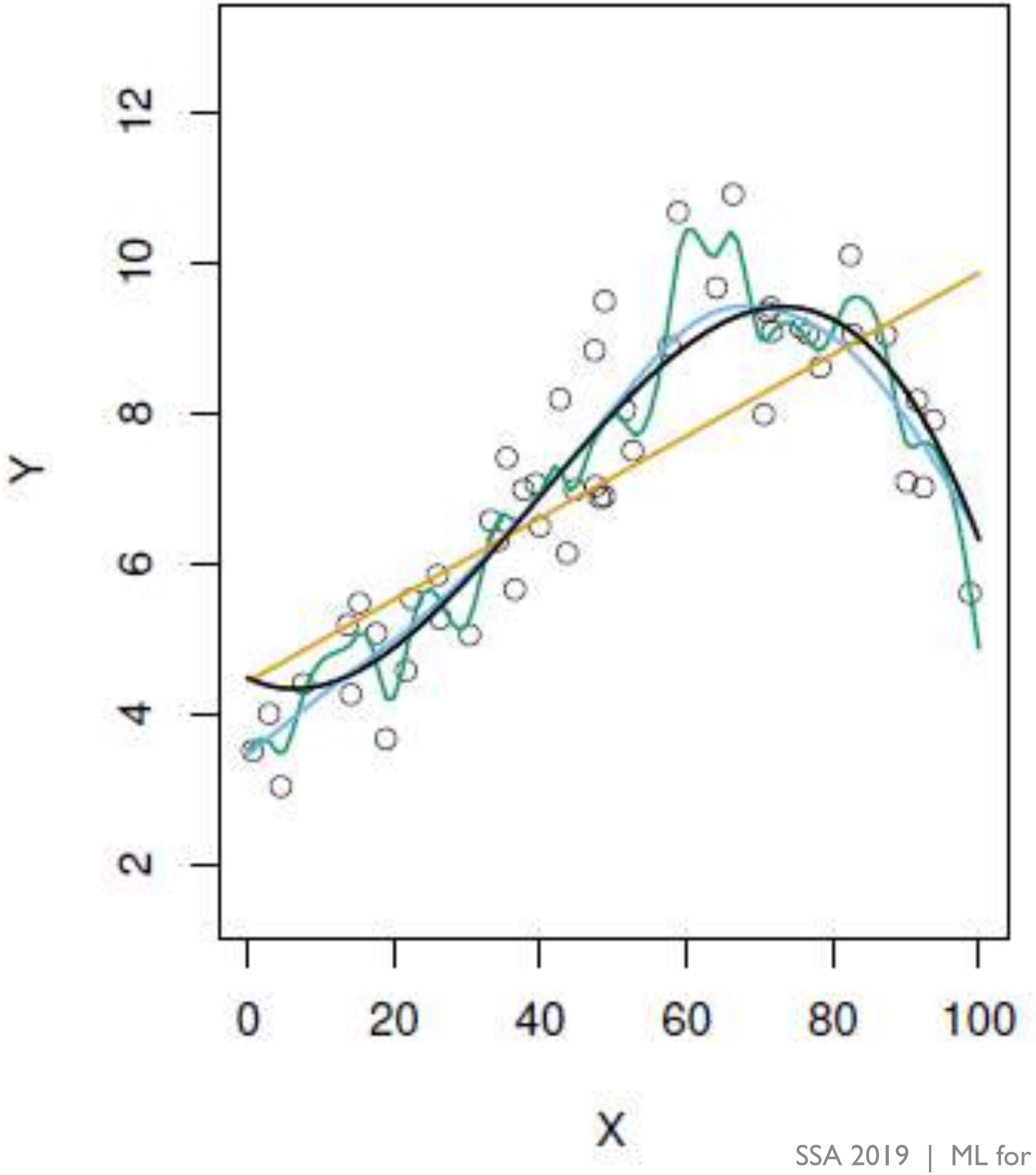


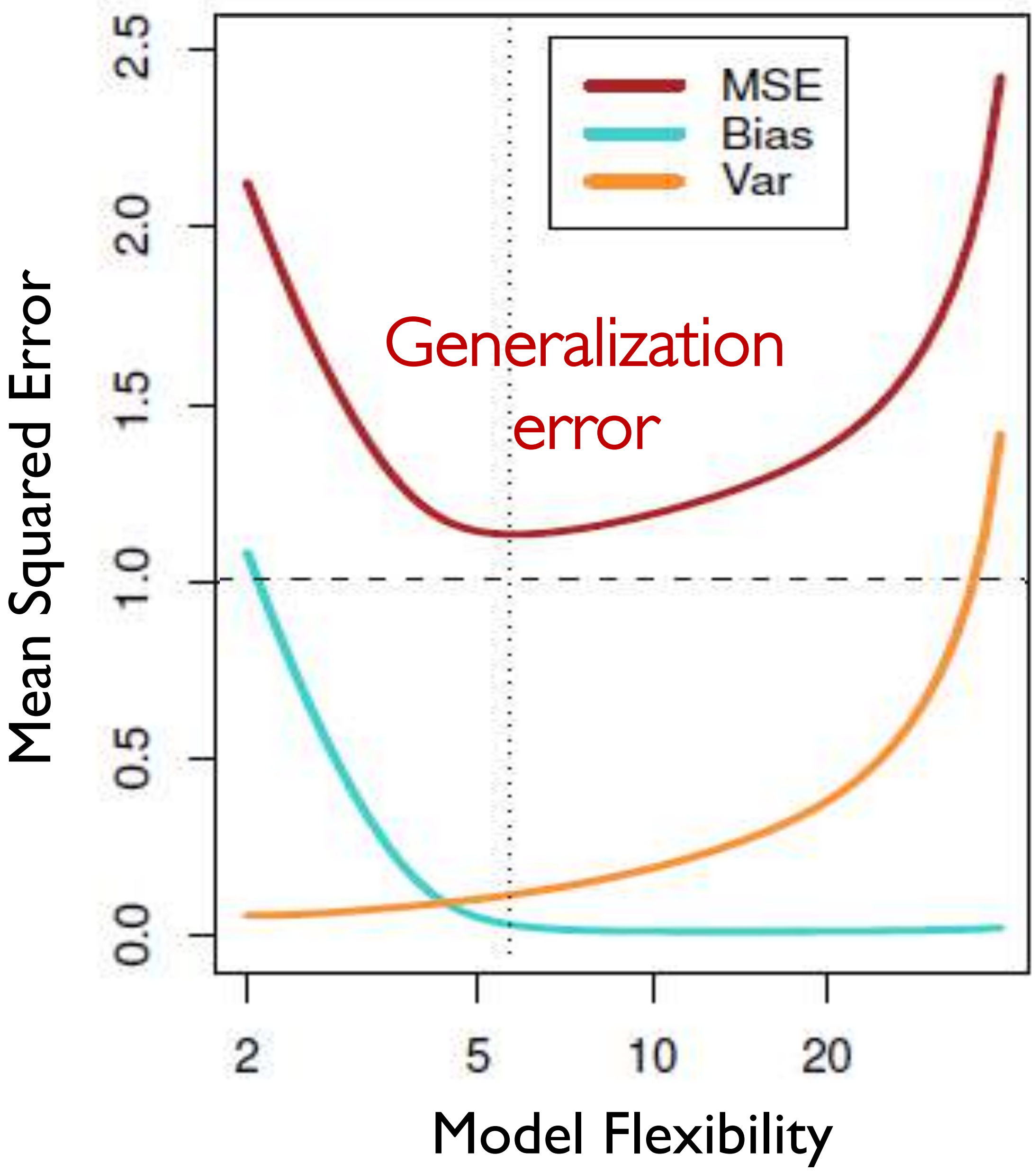
Training error ≈ Generalization error

Figures 2.9, 2.12 , ISL (2013)



Figures 2.9, 2.12 , ISL (2013)





Am I overfitting?

Model memorizes training data, doesn't learn underlying structure.

→ will not generalize to new observations

Characteristic of overfitting:

Training accuracy –	99%	vs.	94%
Test accuracy –	70%		92%

Test error > Training error (*overfitting*)

Test error ≈ Training error (*not overfitting*)

Strategies to avoid overfitting

Degree of model flexibility selected by cross-validation!

Use a simple (less flexible) model

Regularization – adding extra information/constraints

Penalties to promote sparsity or smoothness, early stopping, dropout

Ensembles – averaging multiple models

Averaging randomized decision trees → random forest model

Both approaches can reduce variance with limited effect on bias.

Evaluating and Improving the Model

Goal: use training data and ML algorithm to learn a predictive model that *generalizes*.

How do we **estimate generalization error** of a model?

Estimating generalization: Train-test split

Training Data	Test Data

Training data

Set of observations used to train the model.

Validation data

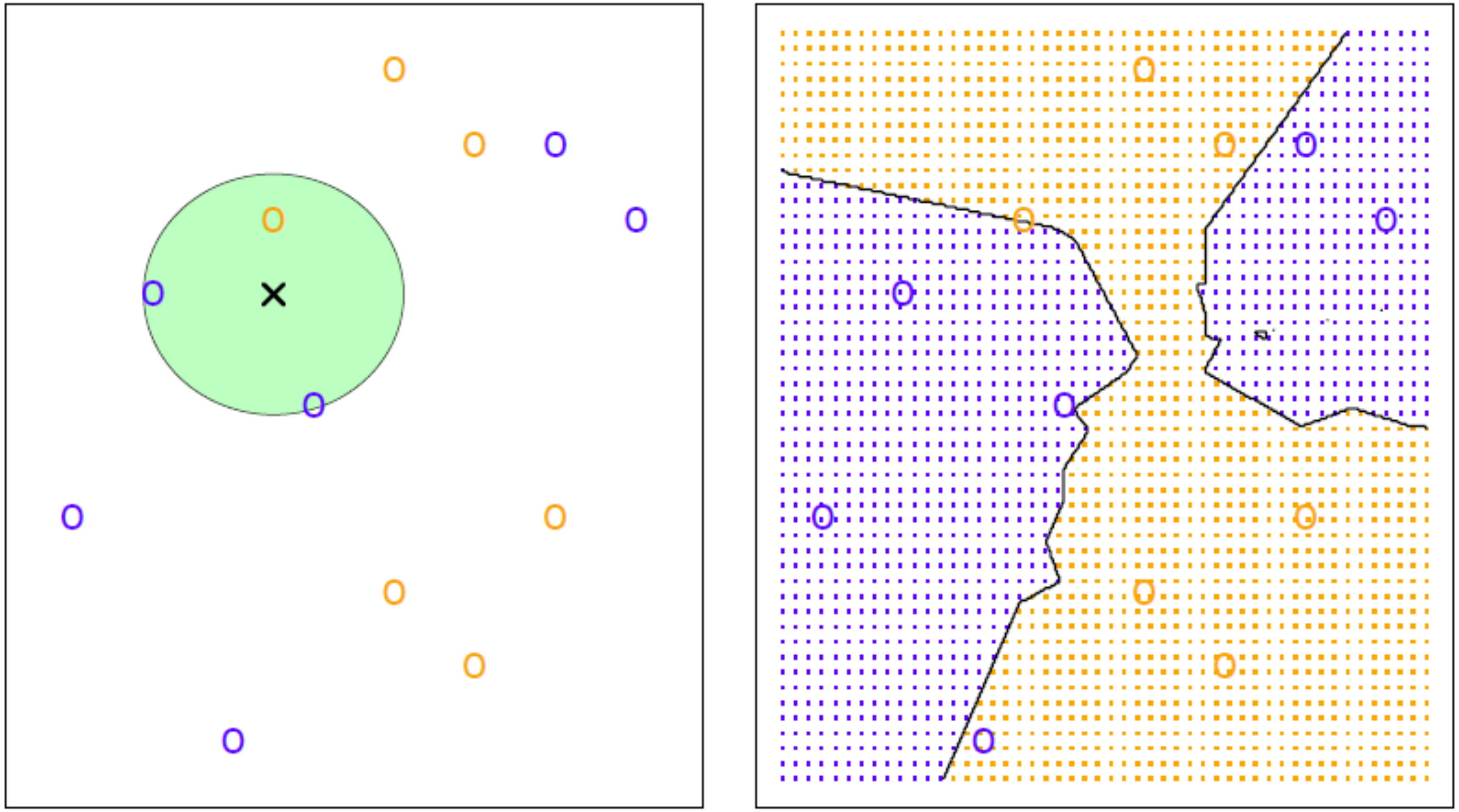
Set of observations used for evaluation during hyperparameter tuning.

Test data

Held out observations used to measure generalization performance.

*These data are **not** available to the algorithm during learning process.*

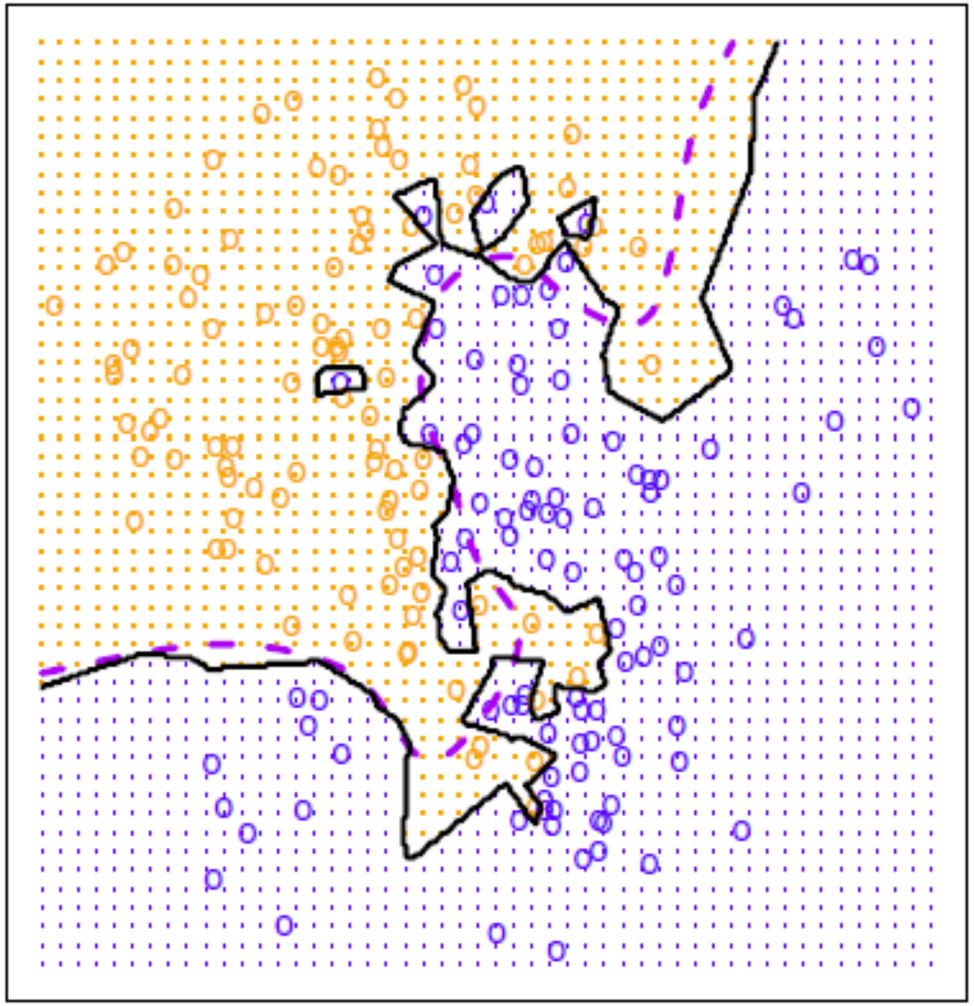
K-Nearest Neighbor classifier (KNN)



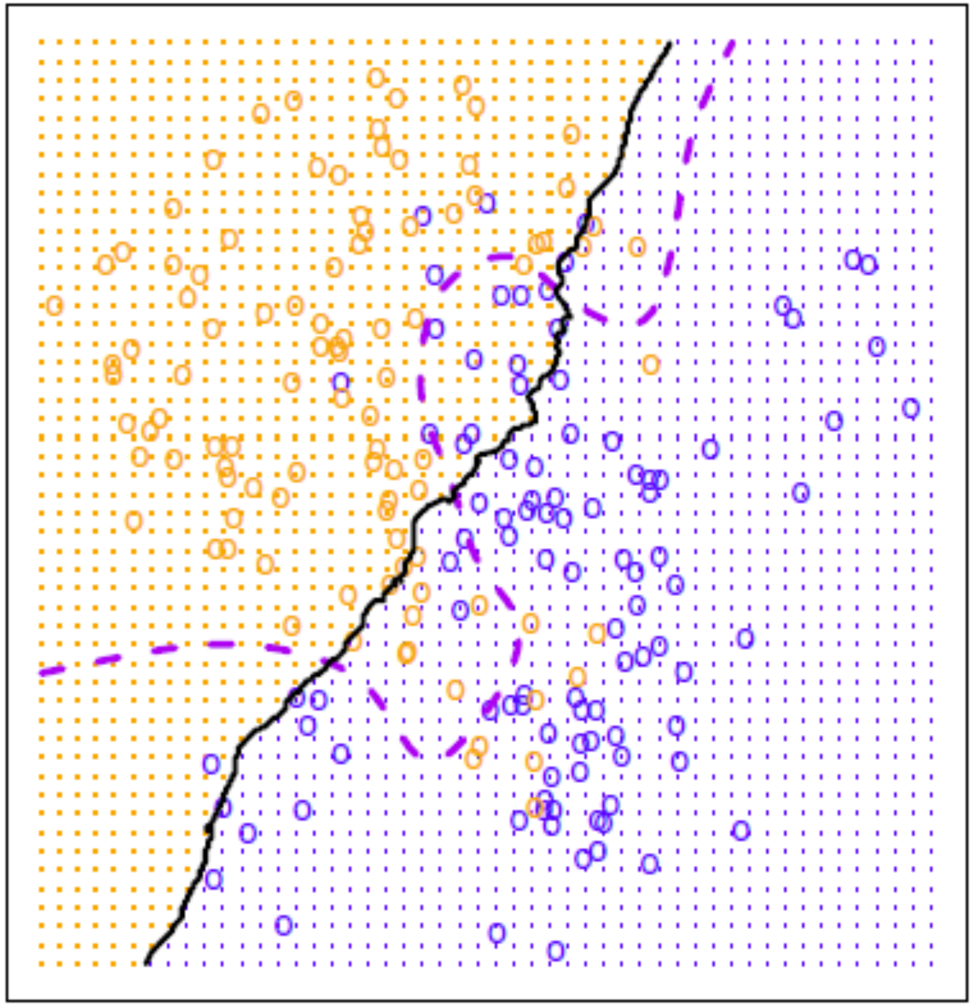
For new observation X, find K nearest observations in training data, assign X to class most common among neighbors.

Figure 2.14, ISL (2013)

KNN classifier depends on choice of hyperparameter K.



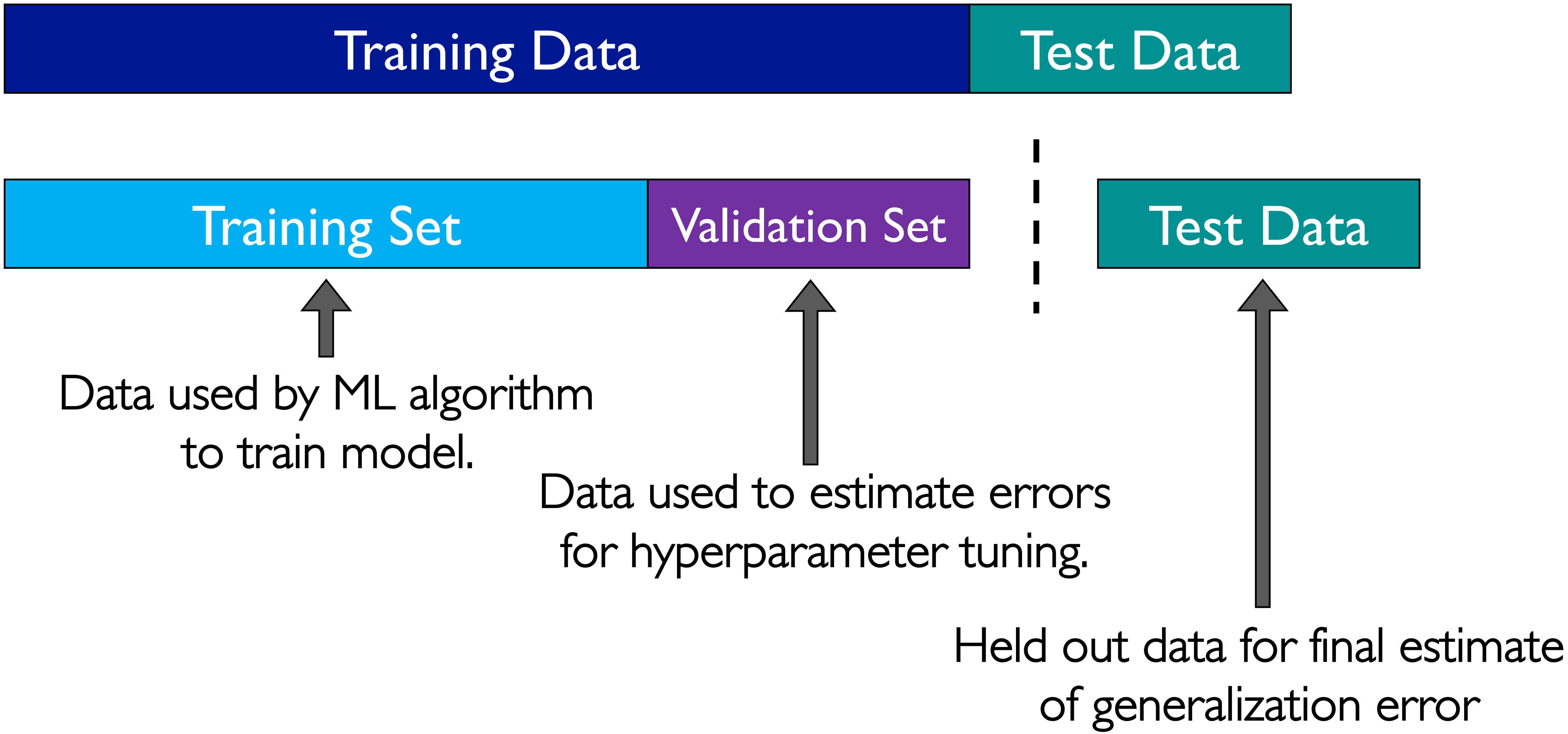
K = 1 → model overfitting



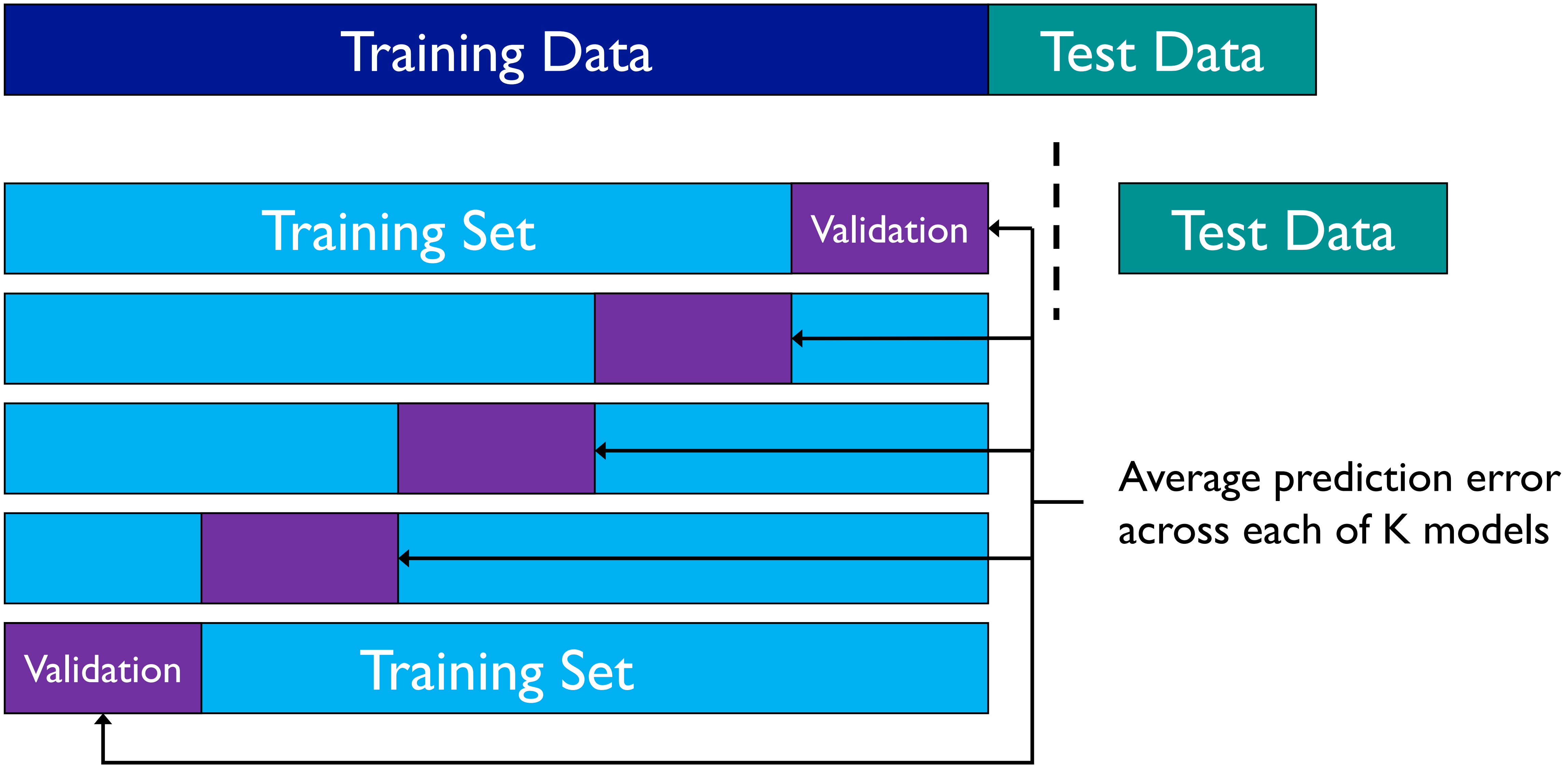
K = 100 → model underfitting

Figure 2.16, ISL (2013)

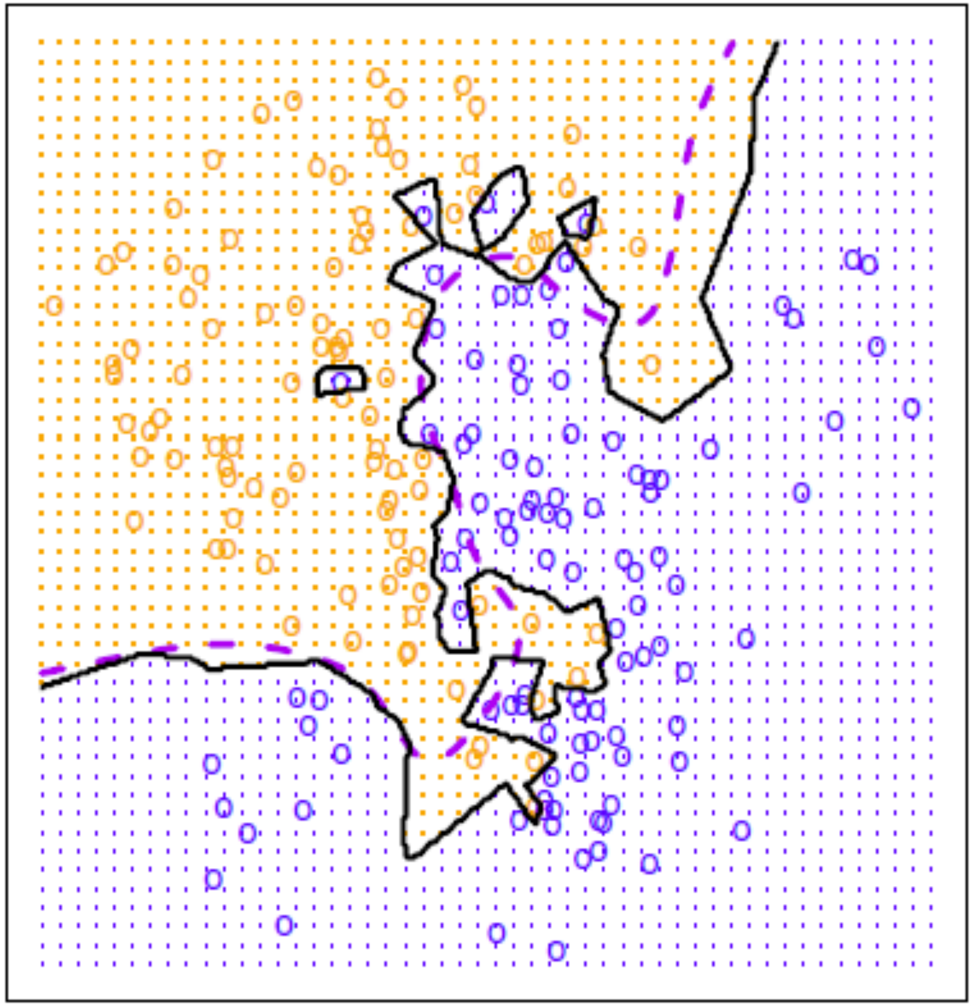
Cross-validation: *tuning model with validation data*



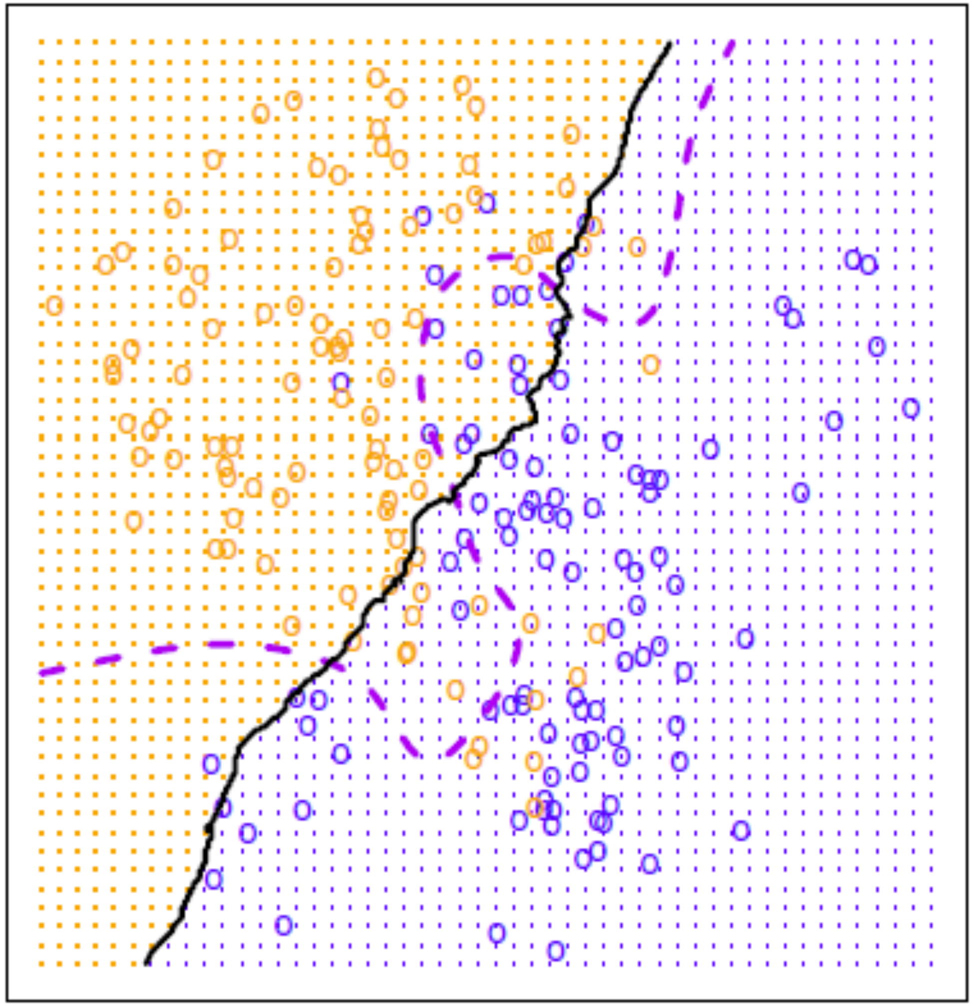
Cross-validation (K-fold)



Hyperparameter K & the bias-variance trade-off



K = 1 → model overfitting



K = 100 → model underfitting

Figure 2.16, ISL (2013)

Choice of K (KNN classifier)

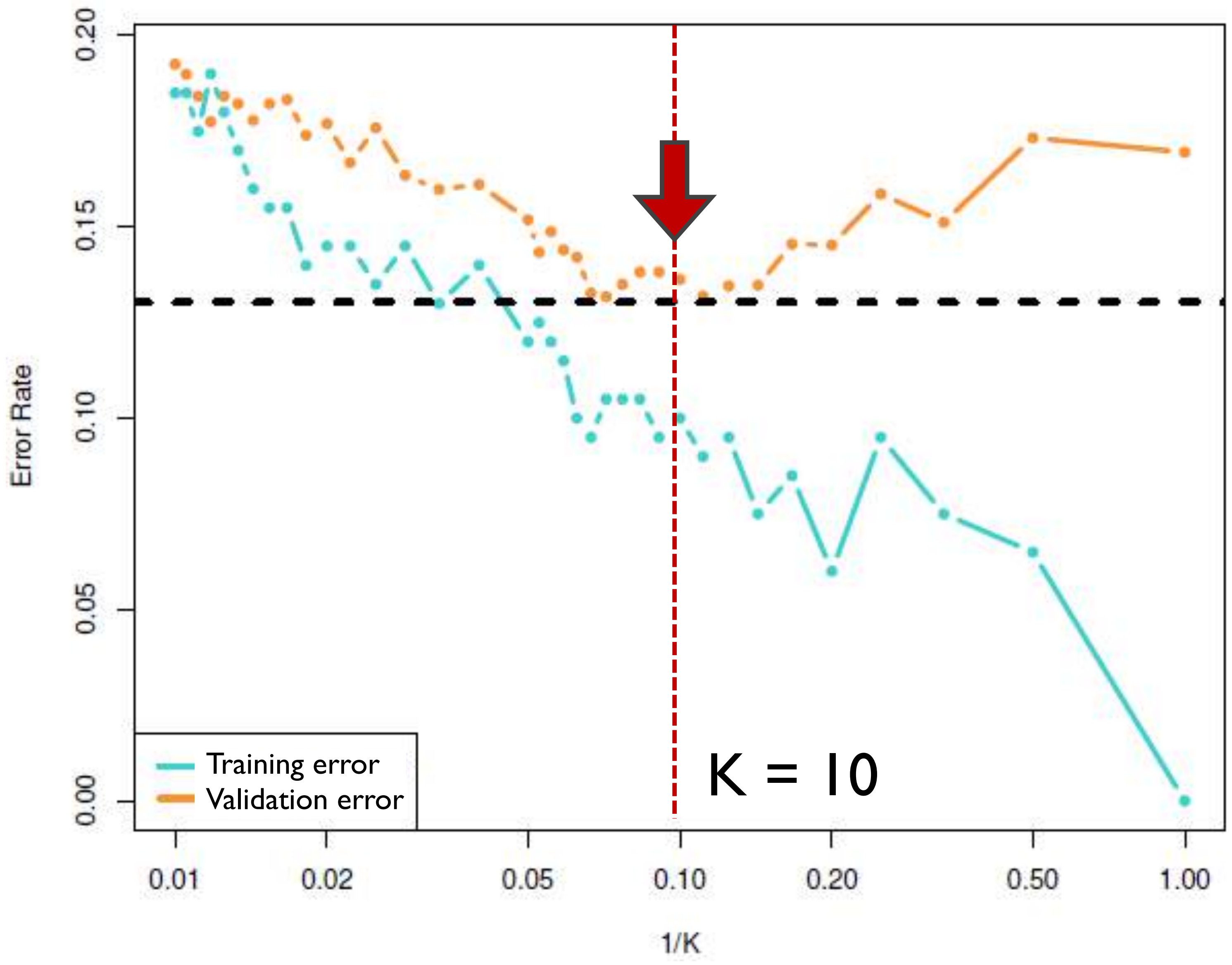


Figure 2.17, ISL (2013)

K = 10

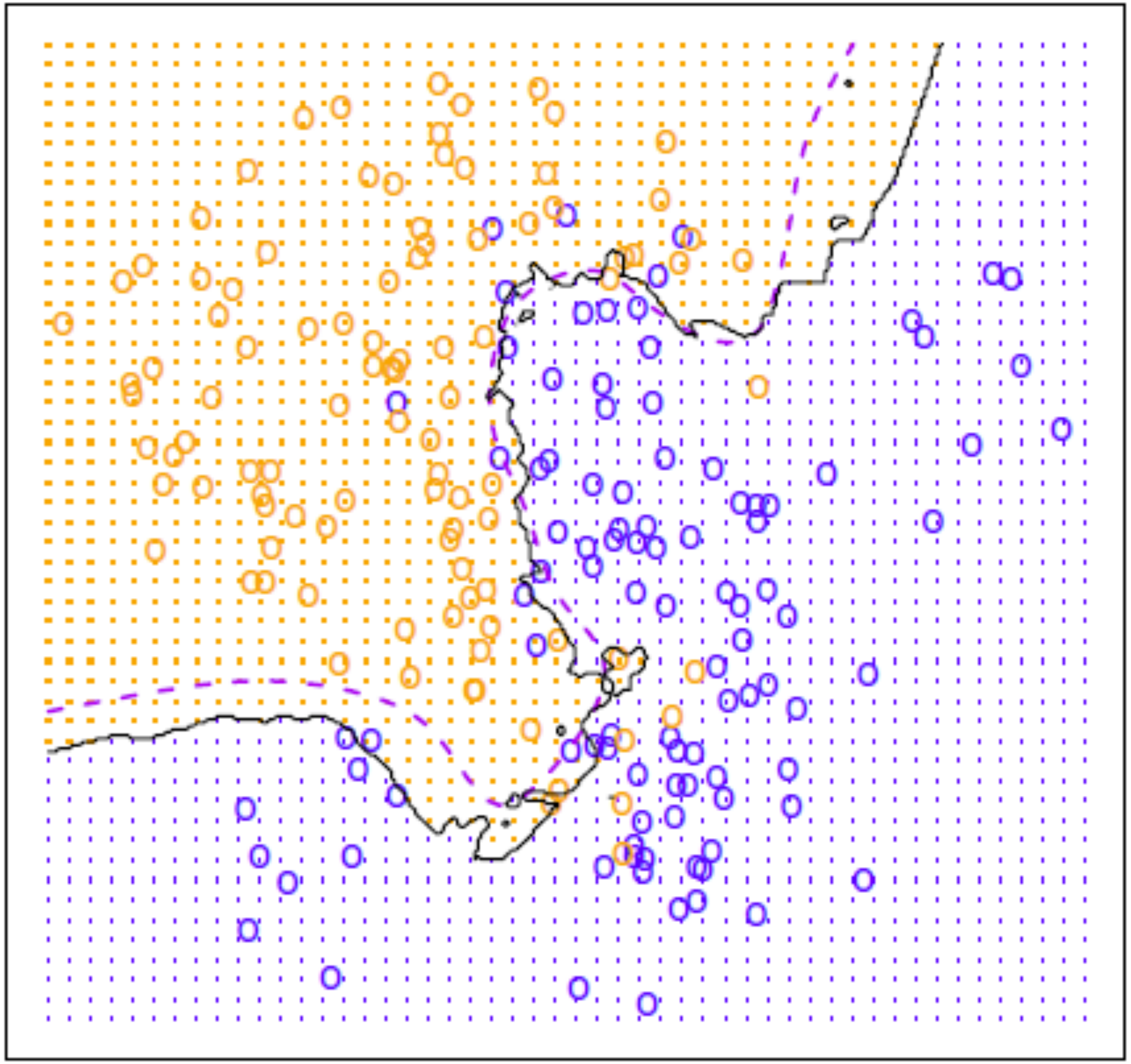
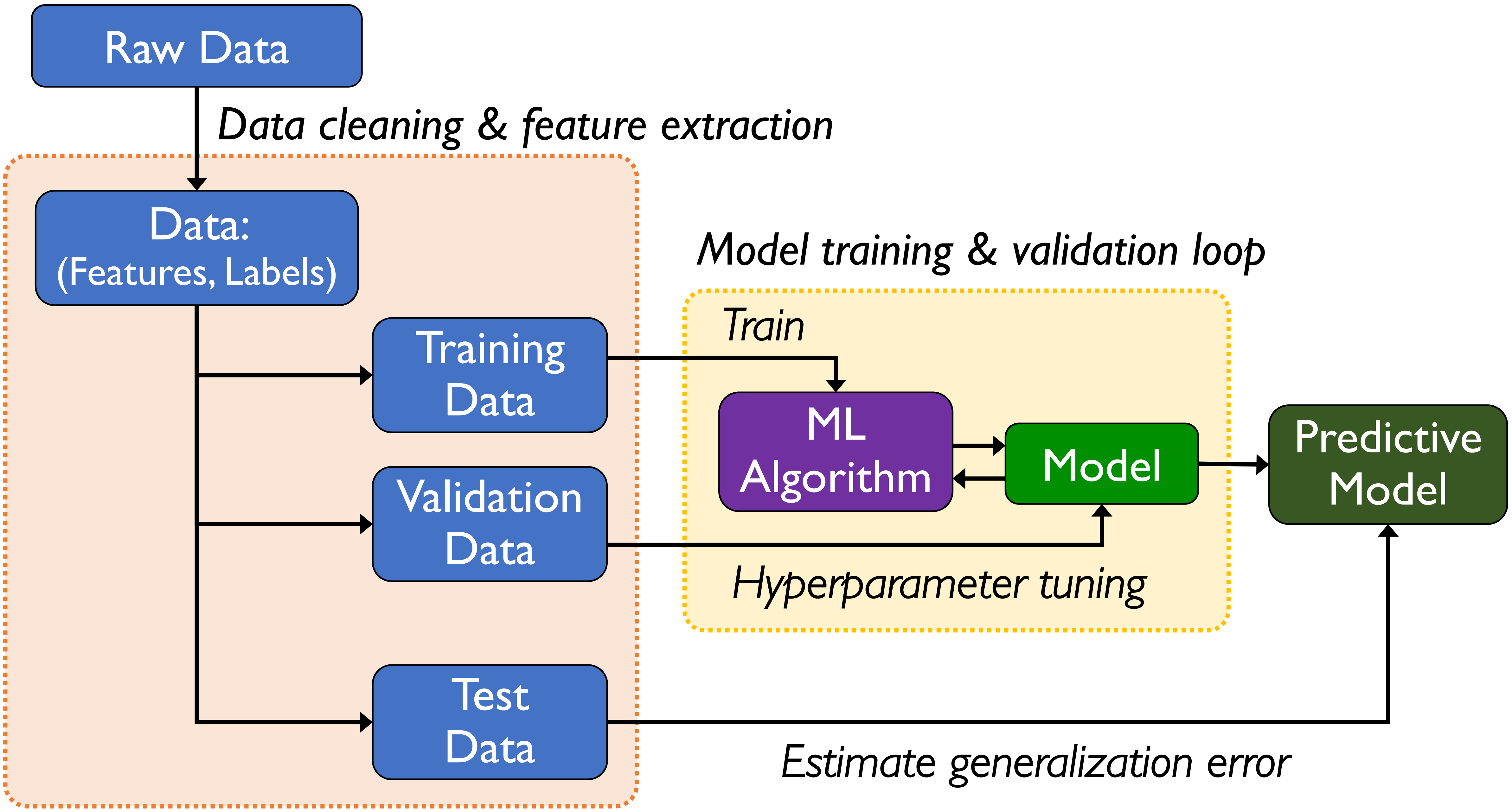


Figure 2.15, ISL (2013)


Raw Data
Data cleaning & feature extraction
Data:
(Features, Labels)
Training Data
Validation Data
Test Data
Model training & validation loop
Train
ML Algorithm
Model
Hyperparameter tuning
Predictive Model
Estimate generalization error

Basic ML Workflow

- Gathering & cleaning the data
- Representing the data
- Building (training) the model
- Evaluating & improving the model
- **Deploying the model**

Questions?

karianne_bergen@fas.harvard.edu

References and Resources

- Witten *et al.* (2013) "Introduction to Statistical Learning with Applications in R."
- Kaufman *et al.* (2011) "Leakage in Data Mining: Formulation, Detection & Avoidance."
- Domingos (2011) "A Few Useful Things to Know about Machine Learning."
- ML for seismology reviews: Bergen *et al.* (2019), *Science;* Kong *et al.* (2018), *SRL*