How flexible should my algorithms be?

Lecture 08

…and launch of the Kaggle competition!

Quiz

Review of Supervised Learning

Algorithm development and application pipeline

Algorithm Development

Application

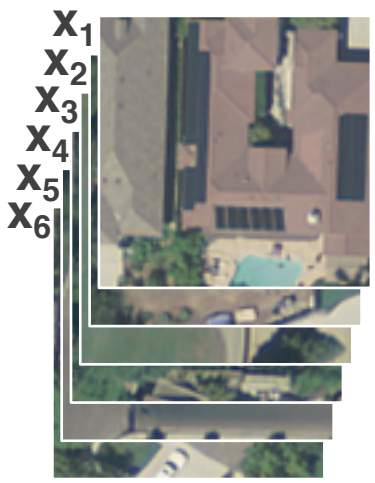
(iterate as needed)

1. **Load training data**
2. **Preprocess & extract features**
3. **Choose learning model**
4. **Evaluate performance**
5. **Make predictions on unseen data**

y

1
1
0
0
1
0

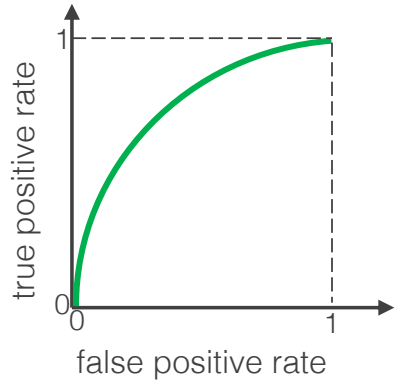
X



X'

x_1	0.38	0.39	0.85	0.78
x_2	0.81	0.91	0.97	0.53
x_3	0.65	0.59	0.91	0.11
x_4	0.94	0.05	0.40	0.26
x_5	0.27	0.19	0.03	0.64
x_6	0.02	0.98	0.36	0.11

Fisher's linear discriminant
perceptron
logistic regression
decision trees
random forests
support vector machine
k nearest neighbors
neural networks



$\mathbf{x}_{new}$ $\hat{y}_{new}$

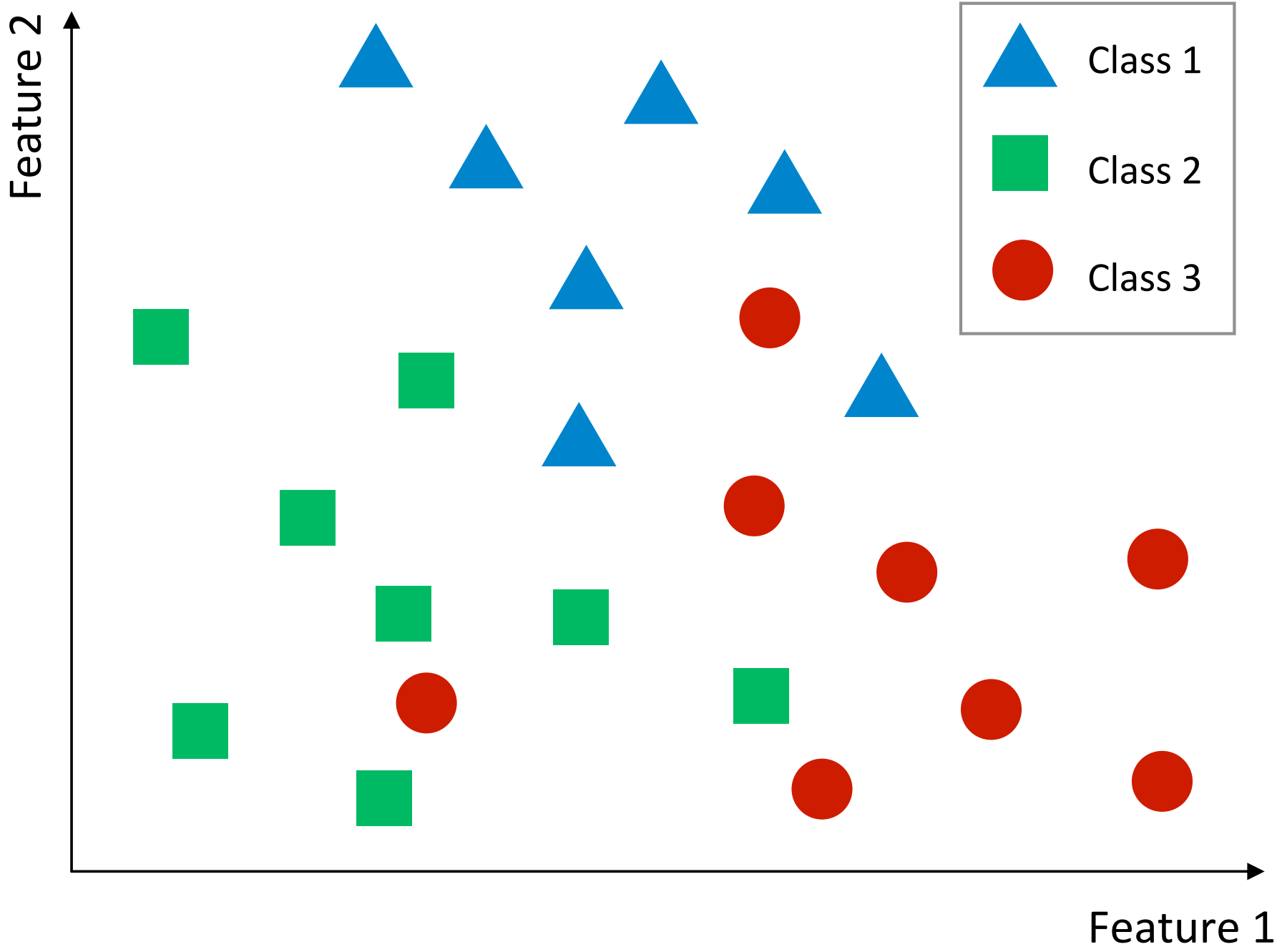
→ 1

→ 0

K-Nearest Neighbors

Classification and Regression

K Nearest Neighbor Classifier

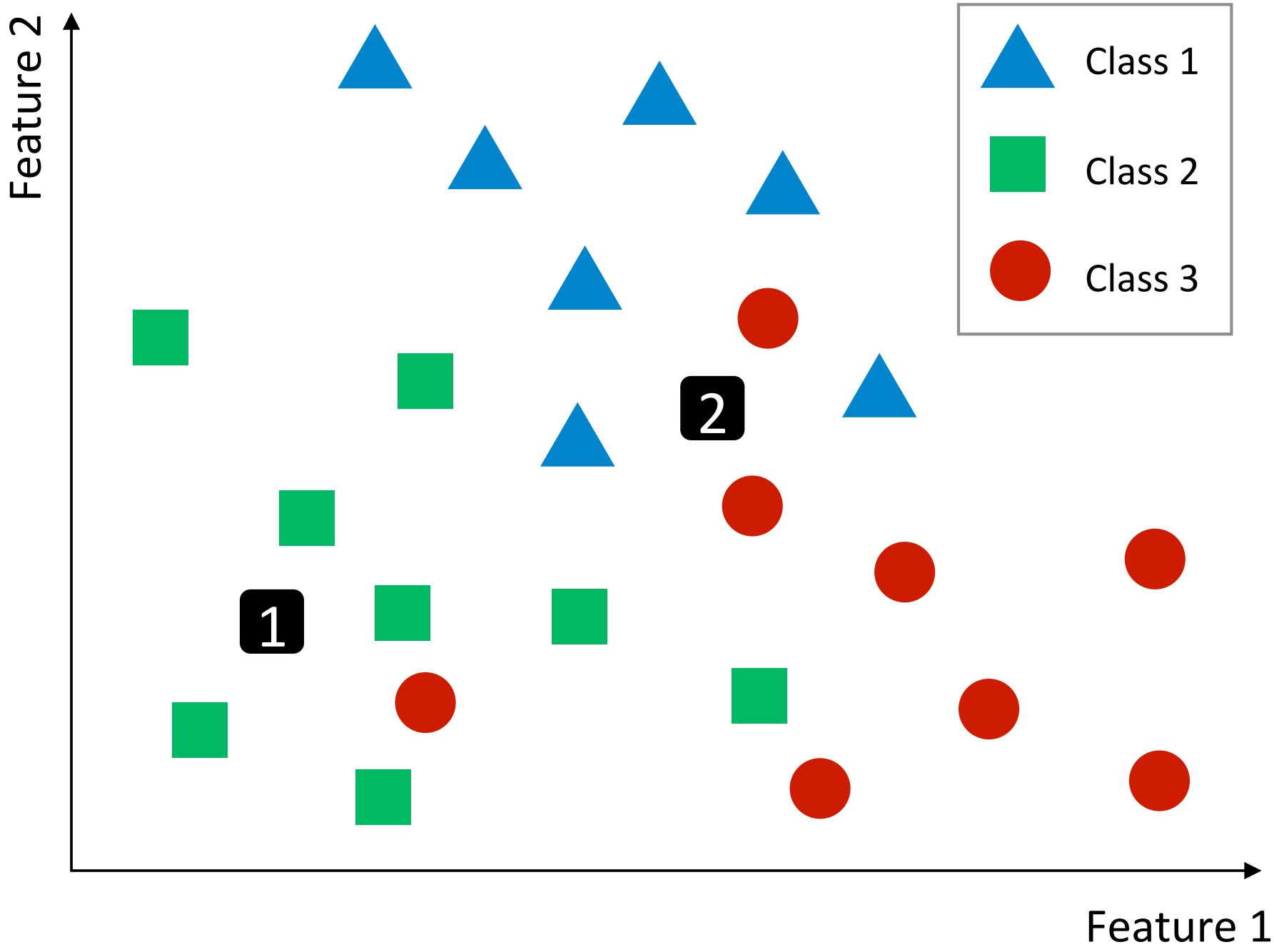


Step 1:
Training

Every new data point is a model parameter

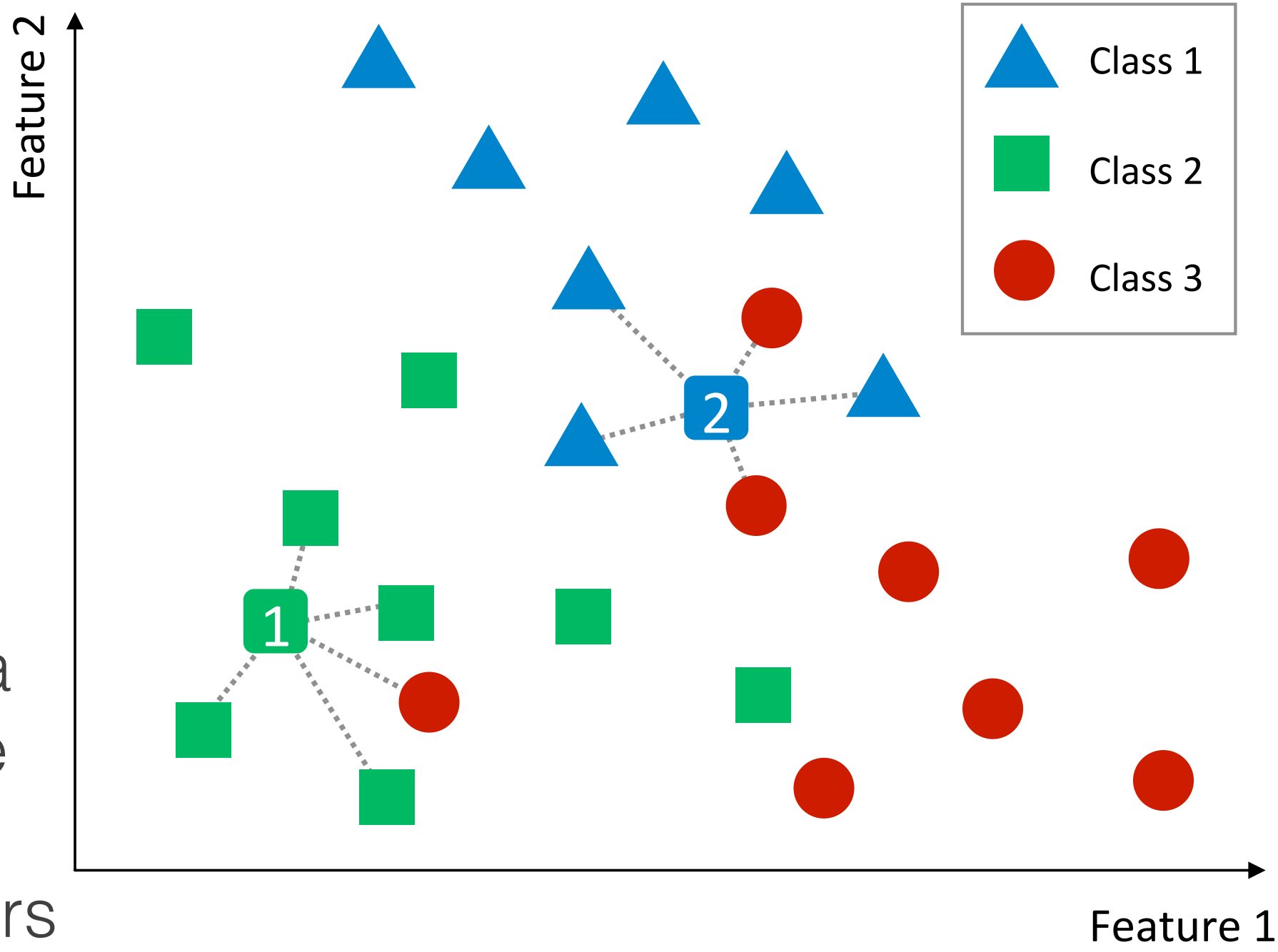
K Nearest Neighbor Classifier

Step 2:
Place new (unseen) examples in the feature space

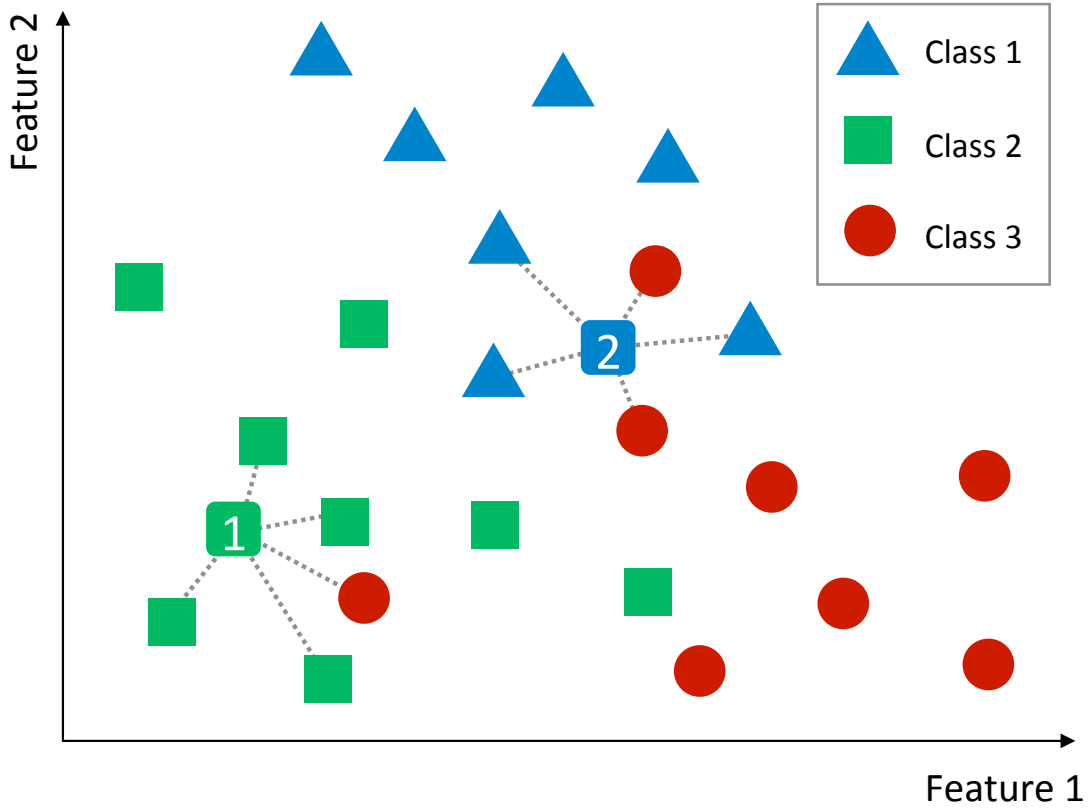


K Nearest Neighbor Classifier

Step 3:
Classify the data by assigning the class of the k nearest neighbors



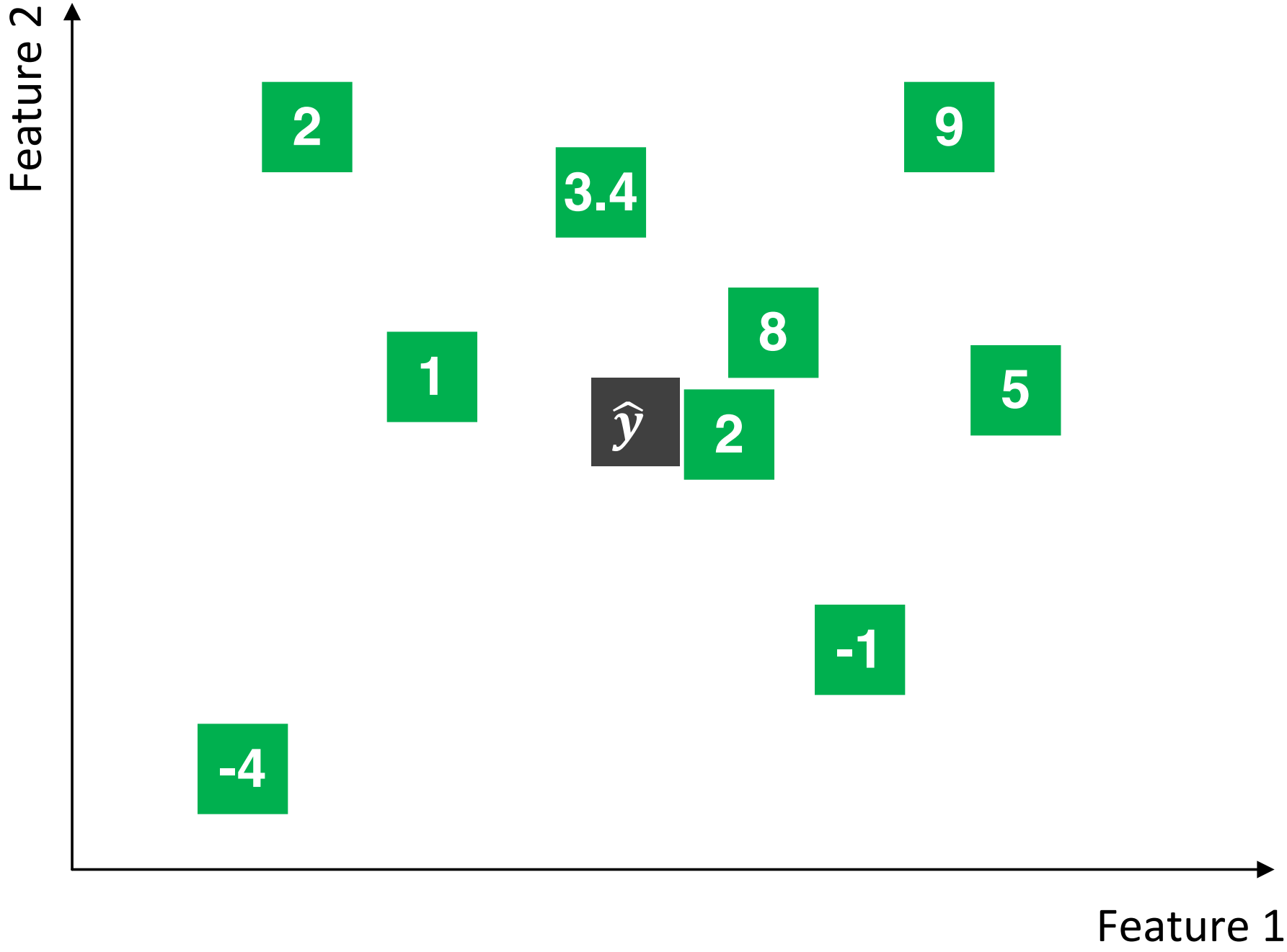
K Nearest Neighbor Classifier



Score vs Decision :
For 5-NN, the confidence score that a sample belongs to a class could be: {0,1/5,2/5,3/5,4/5,1}

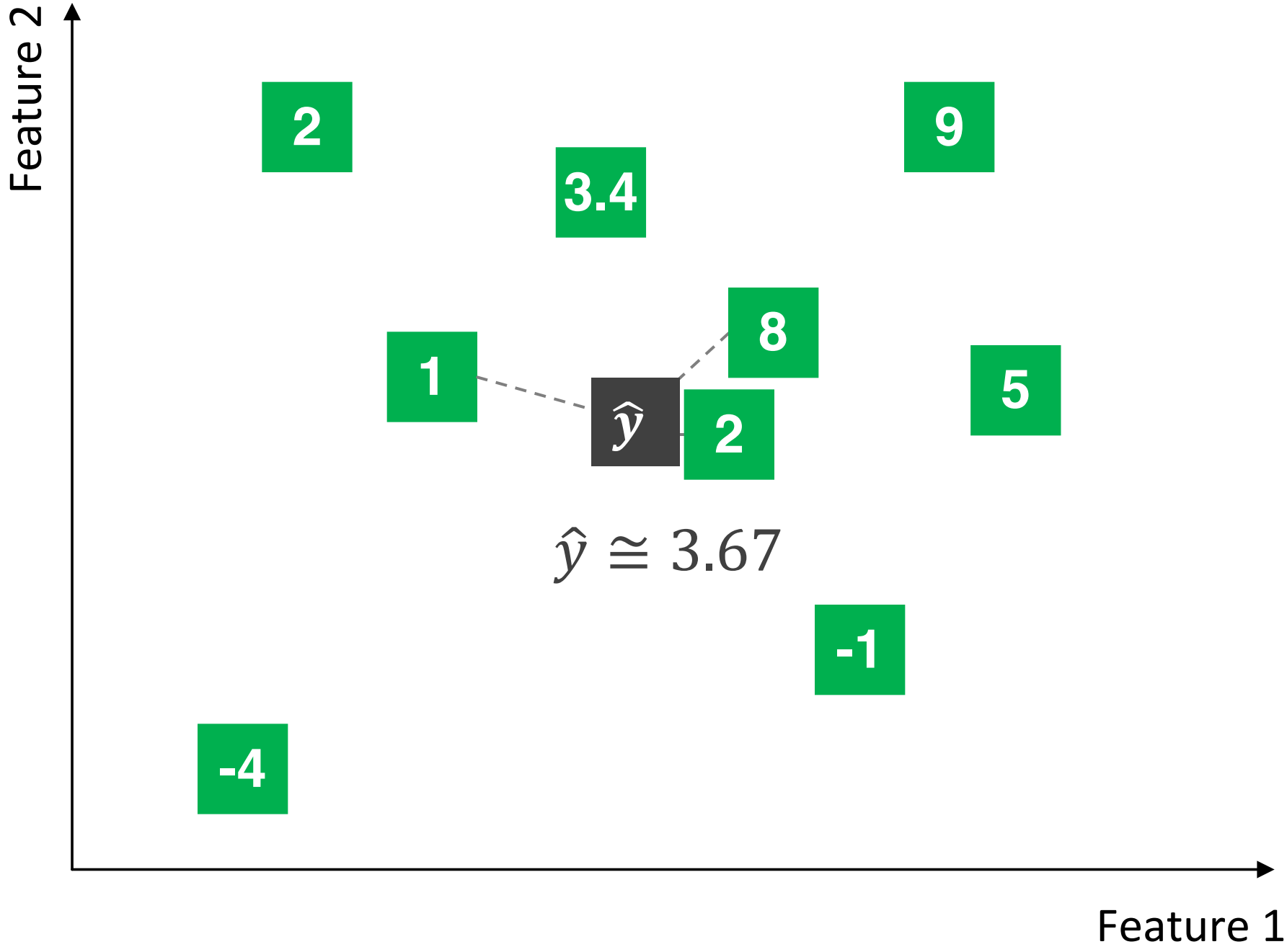
Decision Rule:
If the confidence score for a class > threshold, predict that class

K Nearest Neighbor Regression



K Nearest Neighbor Regression

$$\hat{y} = \frac{1}{k} \sum_{y_i \in \{\text{k nearest}\}} y_i$$



KNN Pros and Cons

Pros

- Simple to implement and interpret
- Minimal training time
- Naturally handles multiclass data

Cons

- Computational expensive to find nearest neighbors
- Requires all of the training data to be stored in the model
- Suffers if classes are imbalanced
- Performance may suffer in high dimensions

How flexible should my model be?

the bias-variance tradeoff and learning to generalize

bias
consistently incorrect prediction

error from poor model assumptions
(high bias results in underfit)

variance
inconsistent prediction

error from sensitivity to small changes in the training data
(high variance results in overfit)

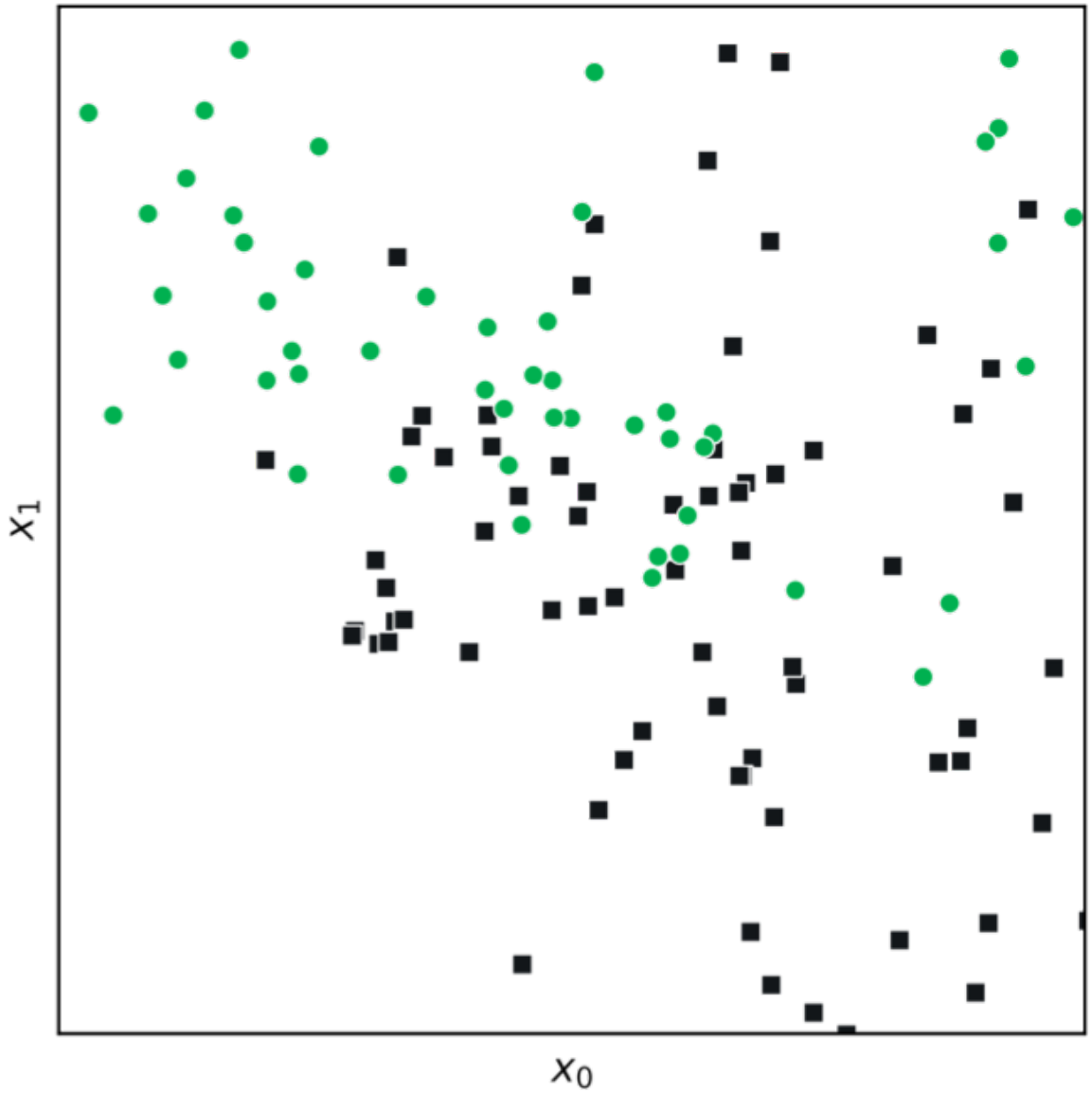
noise
lower bound on generalization error

irreducible error inherent to the problem
(e.g. you cannot predict the outcome of a flip of a fair coin any more than 50% of the time)

Bias-Variance Tradeoff

$$\text{generalization error} = \text{bias}^2 + \text{variance} + \text{noise}$$

Classification feature space



What's the best we can do for binary classification?

If we know the probability distribution of the data

The Bayes decision rule

Bayes' Rule

$$P(C|\boldsymbol{X}) = \frac{P(\boldsymbol{X}|C)P(C)}{P(\boldsymbol{X})}$$

Likelihood Prior

Posterior

Evidence

$\boldsymbol{X}$ Features

C Class label

i.e. $C \in \{c_0, c_1\}$ for the binary case

Bayes' Decision Rule:

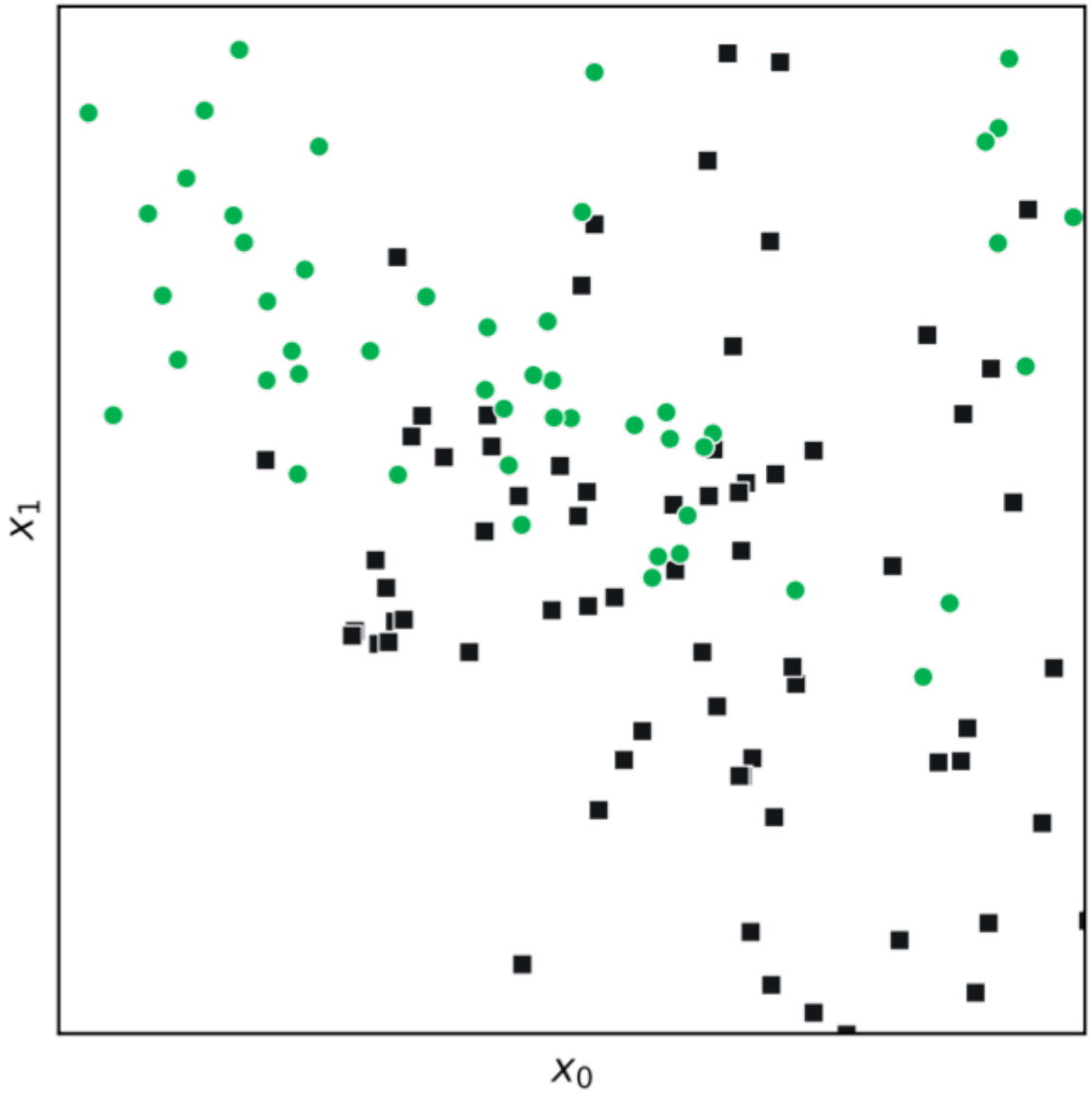
choose the most probable class given the data

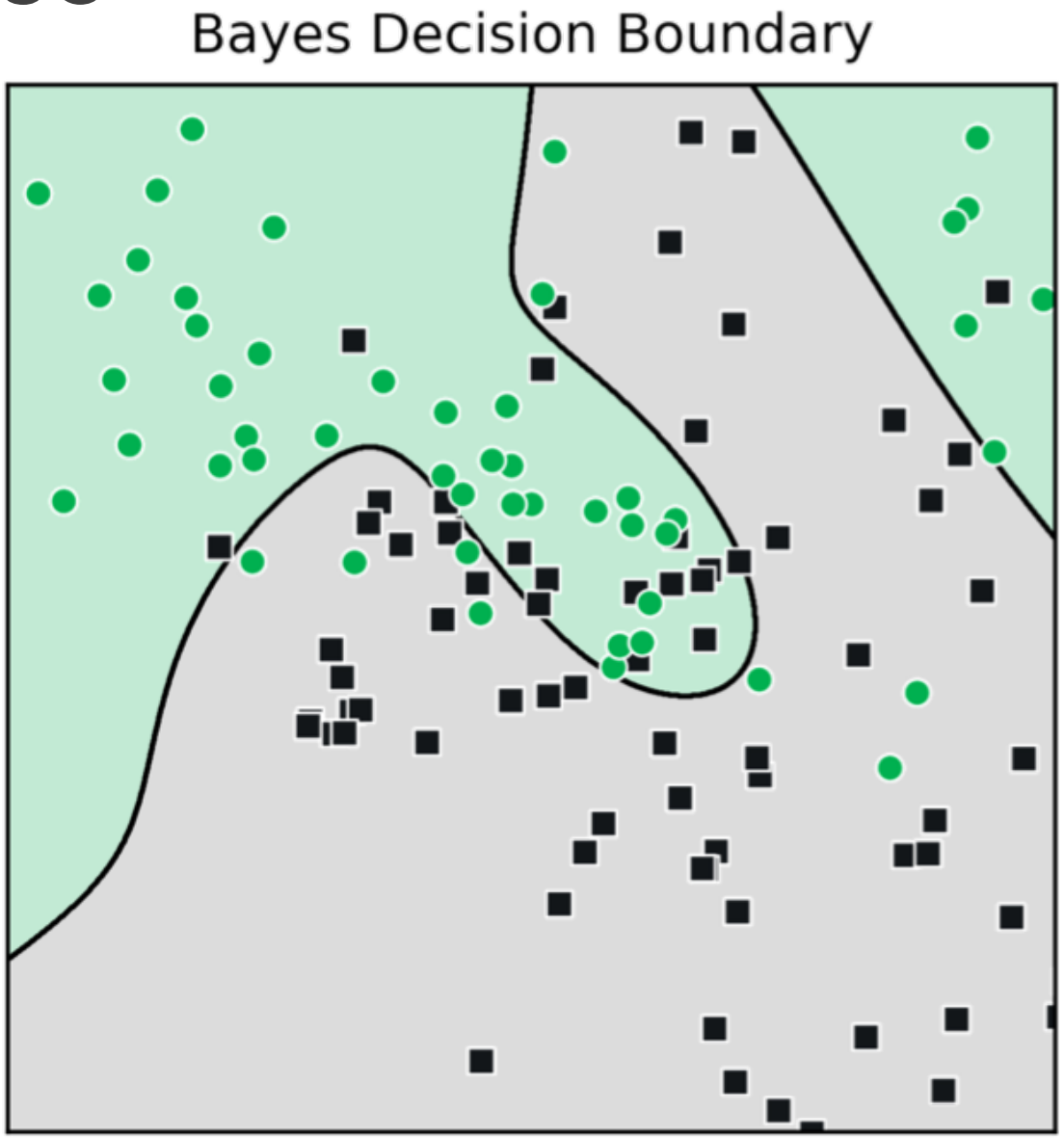
If $P(C_i = c_1|\boldsymbol{X}_i) > P(C_i = c_0|\boldsymbol{X}_i)$ then $\hat{y} = c_1$

otherwise $\hat{y} = c_0$

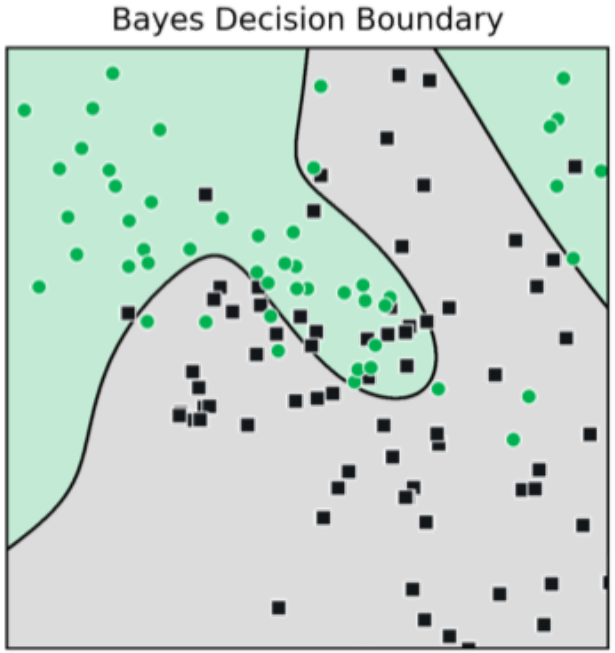
- If the distributions are correct, this decision rule is **optimal**
- Rarely do we have enough information to use this in practice

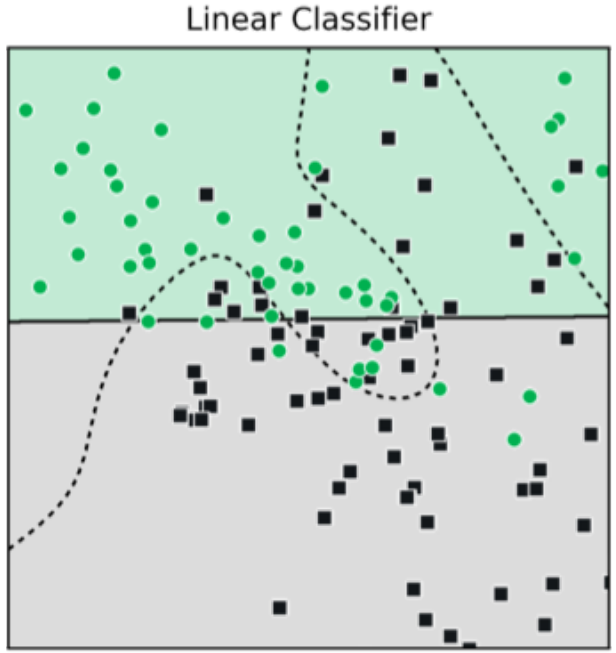
Classification feature space

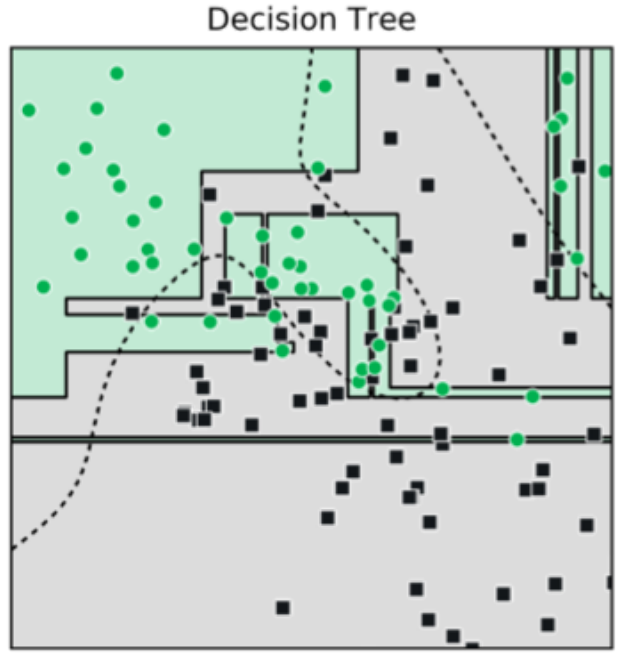


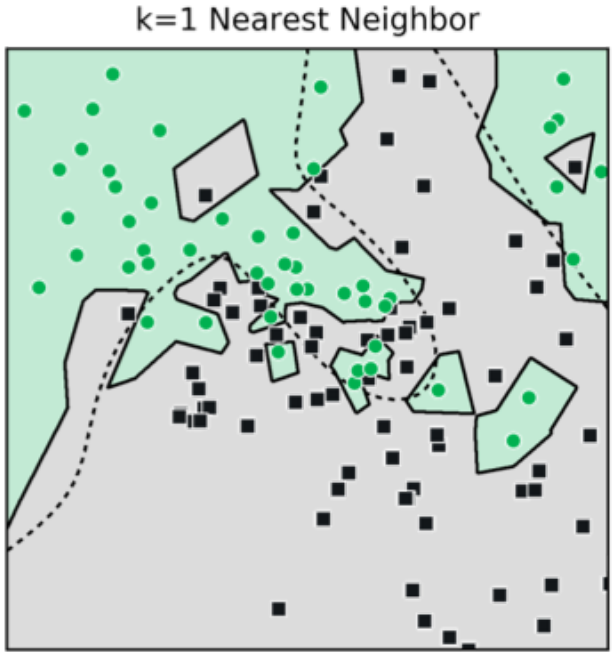


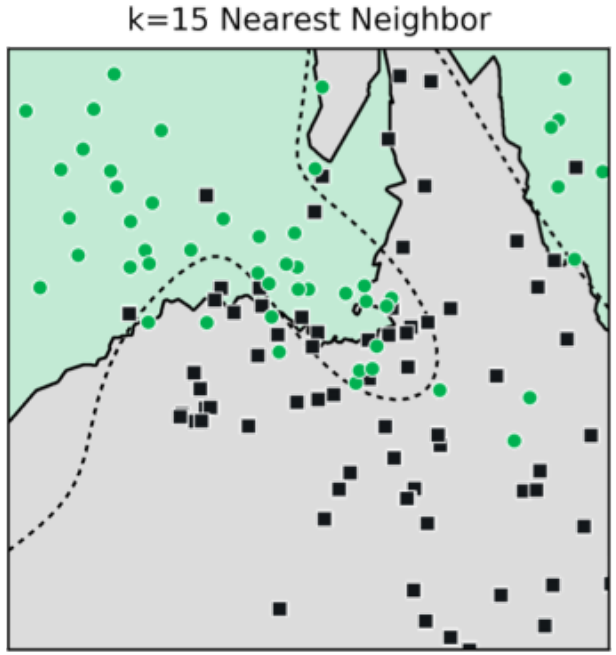
Decision Boundary Examples

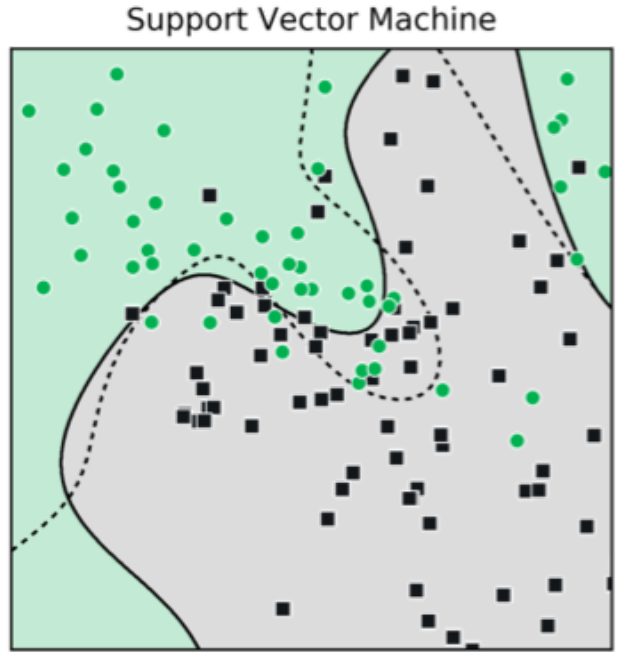




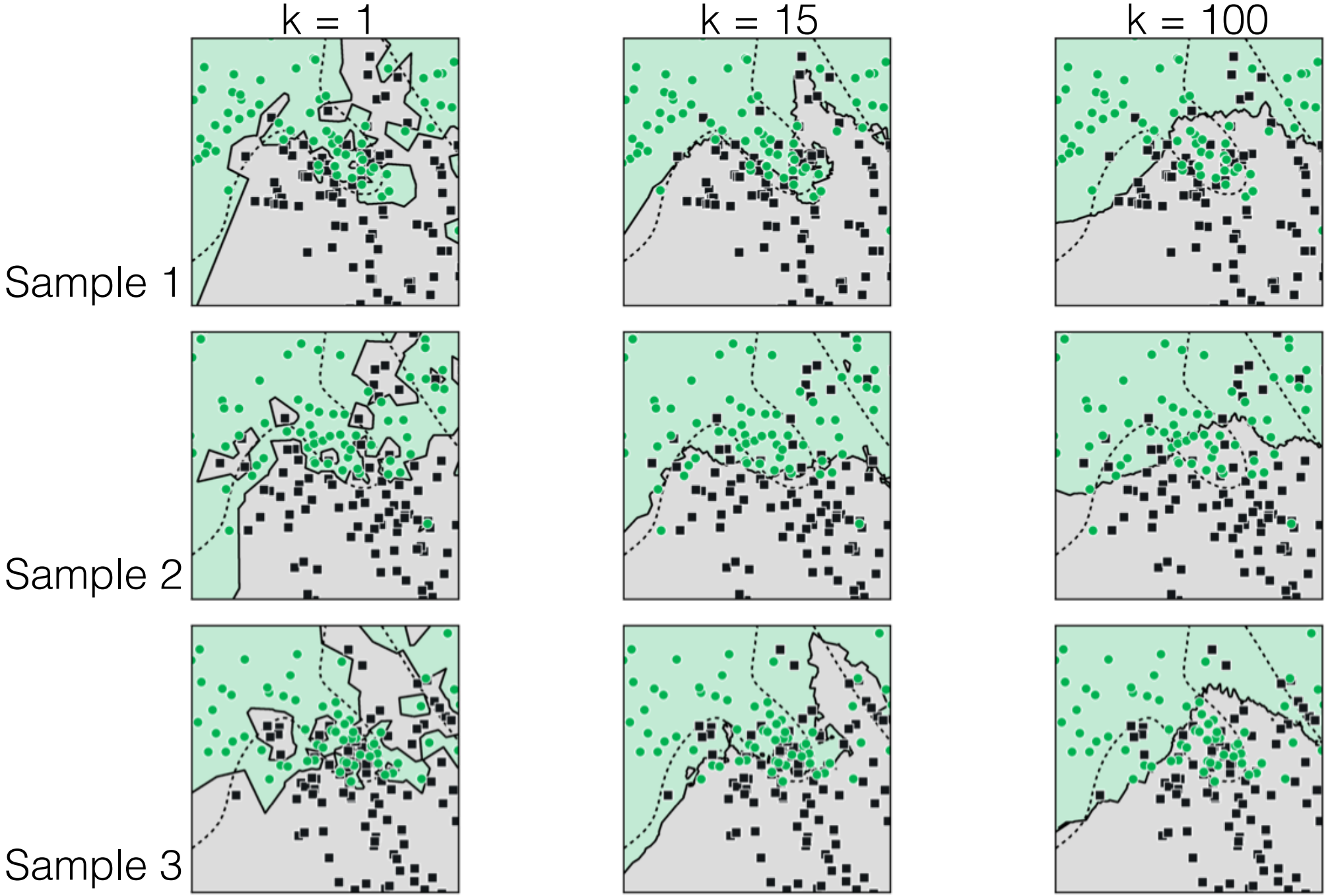




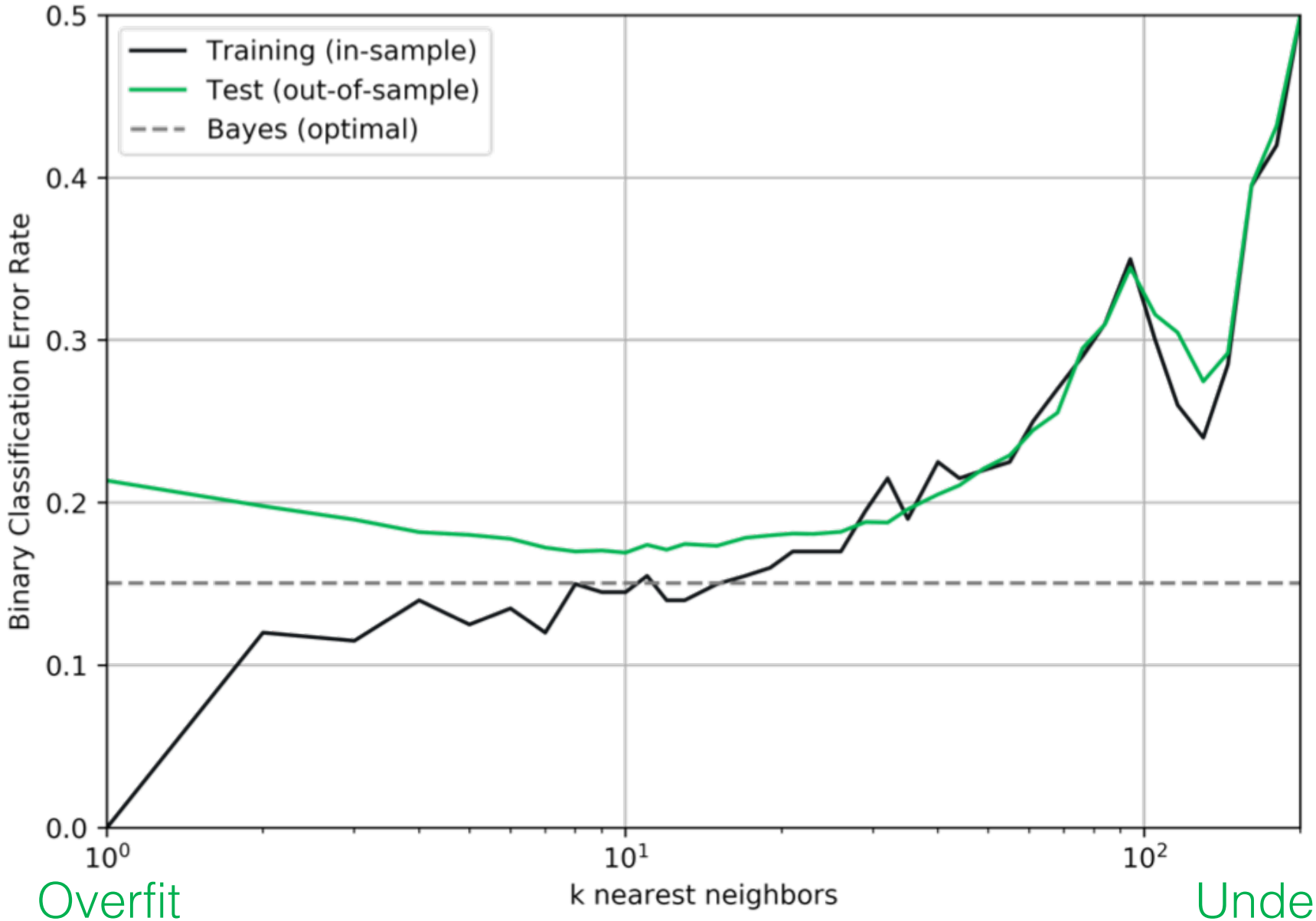




Bias Variance Tradeoff



Bias Variance Tradeoff



Overfit
High Variance
Low Bias

more flexible

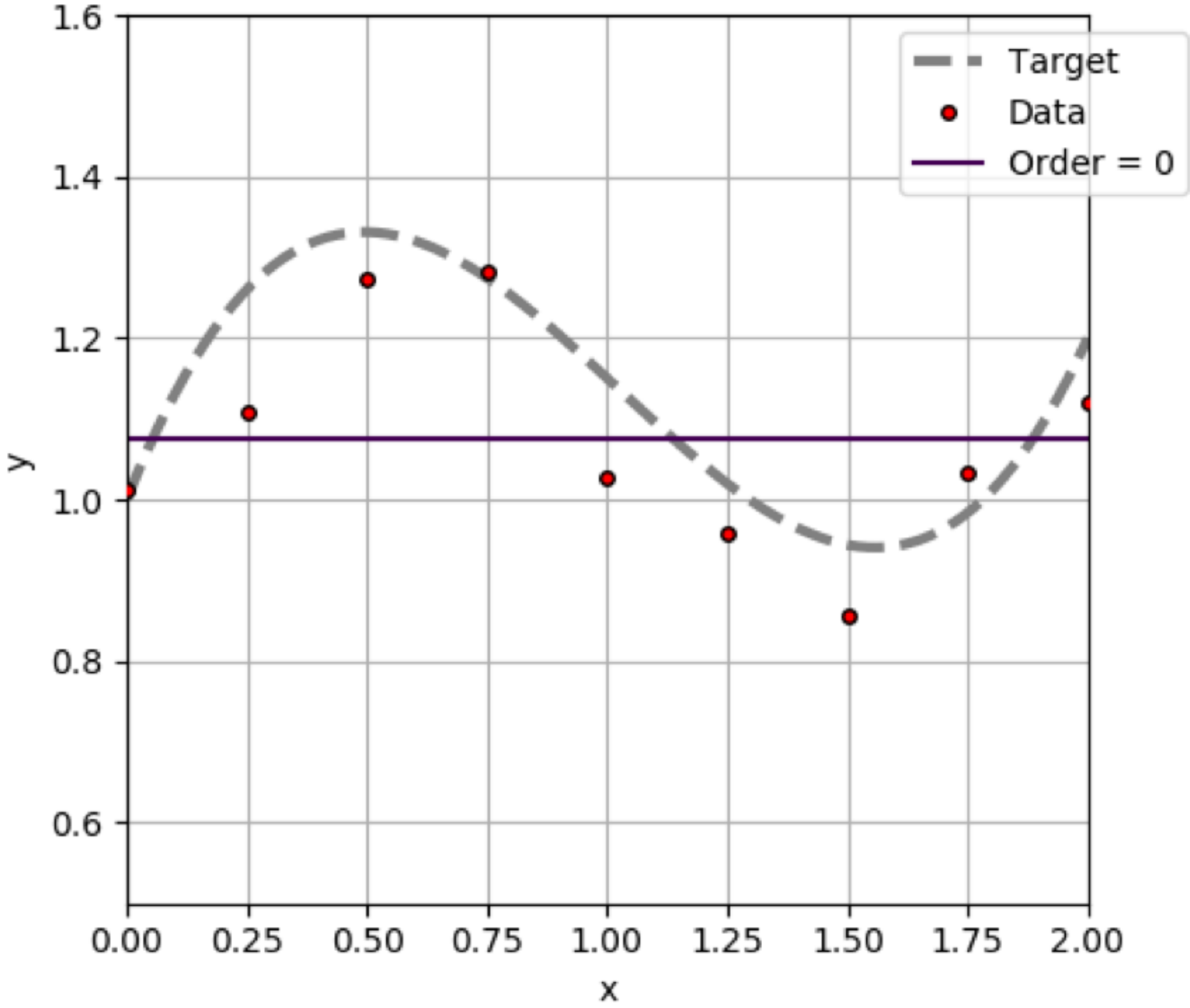
Underfit
Low Variance
High Bias

This tradeoff is equally challenging for regression

Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

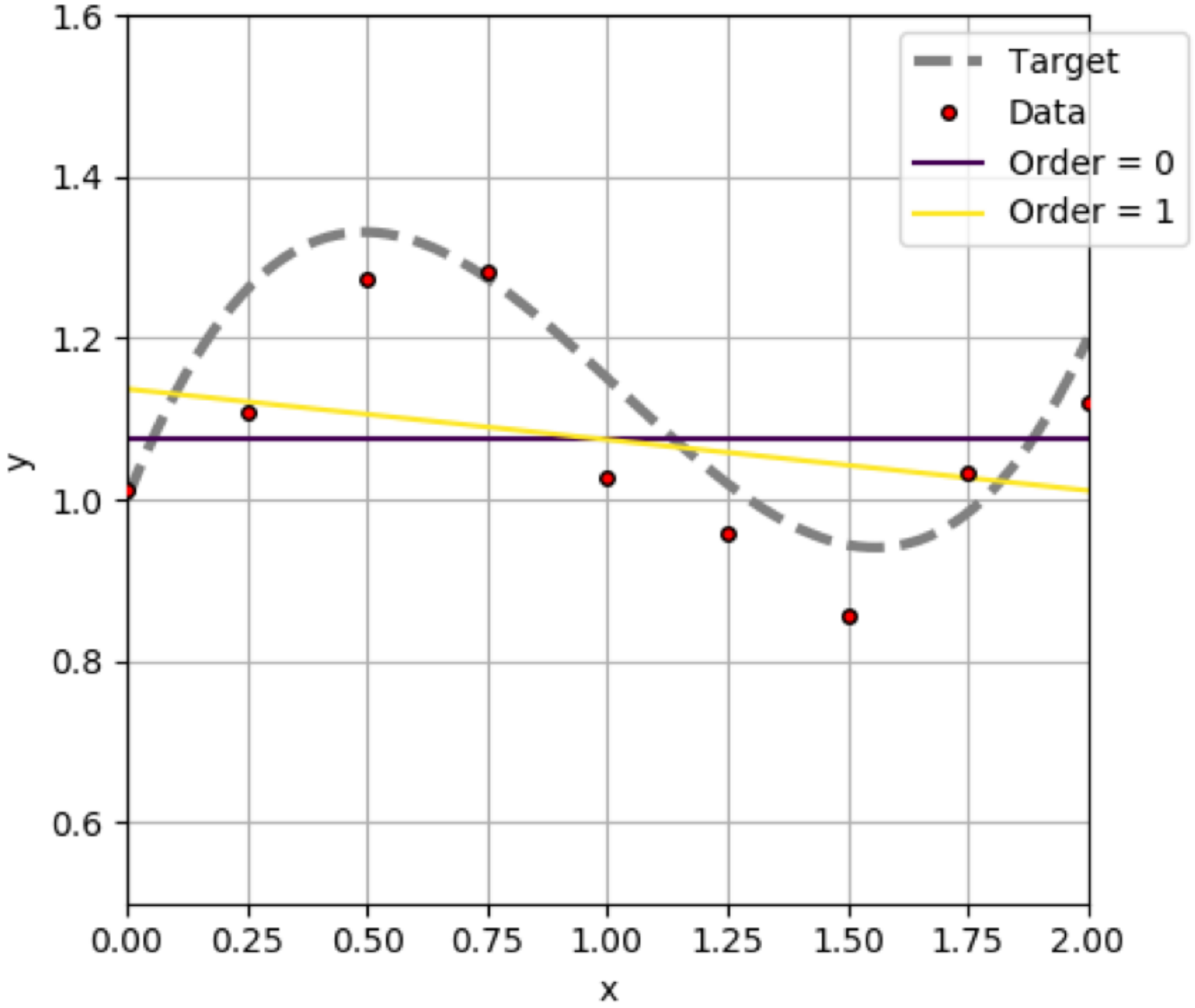
N is the model order



Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

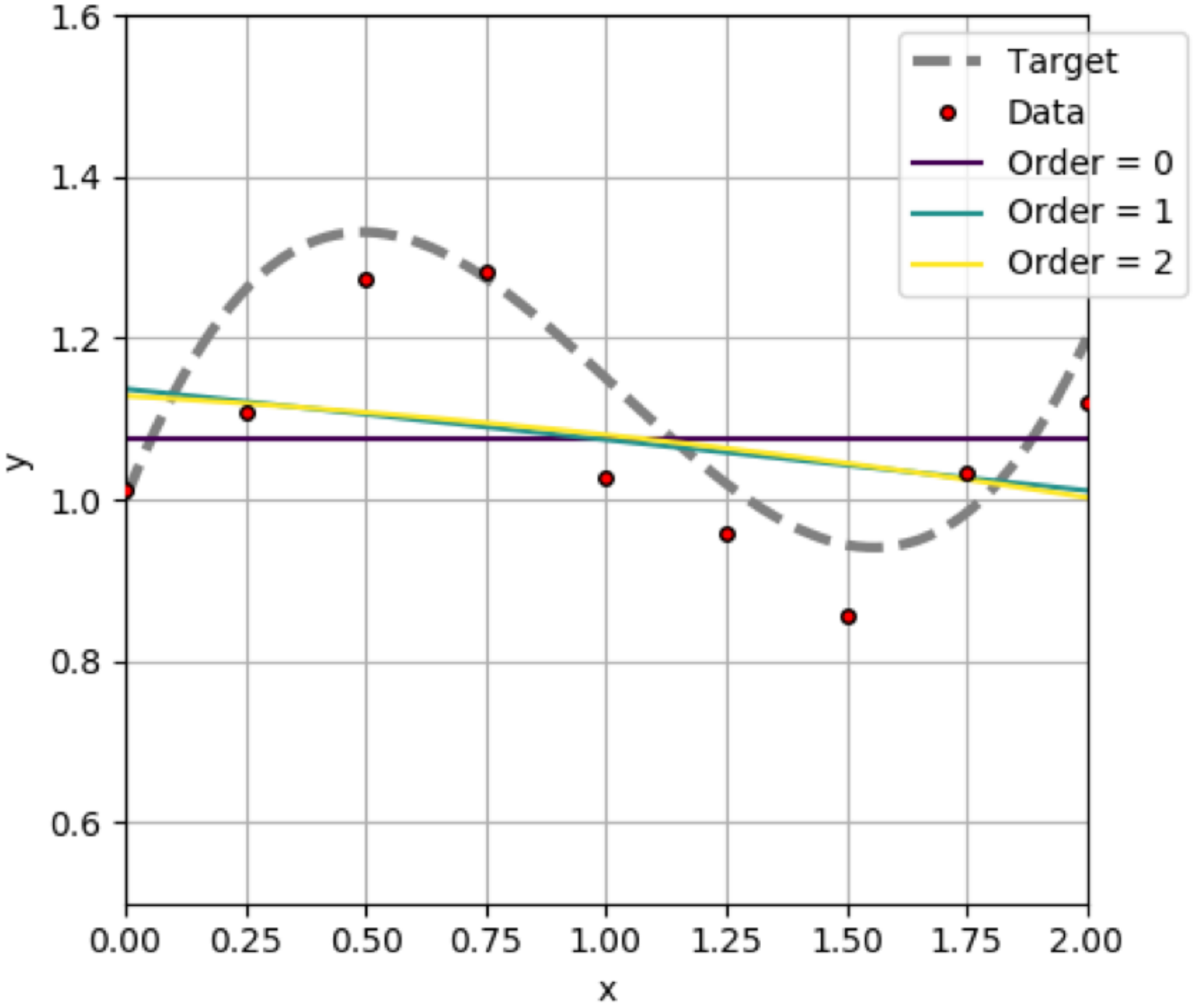
N is the model order



Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

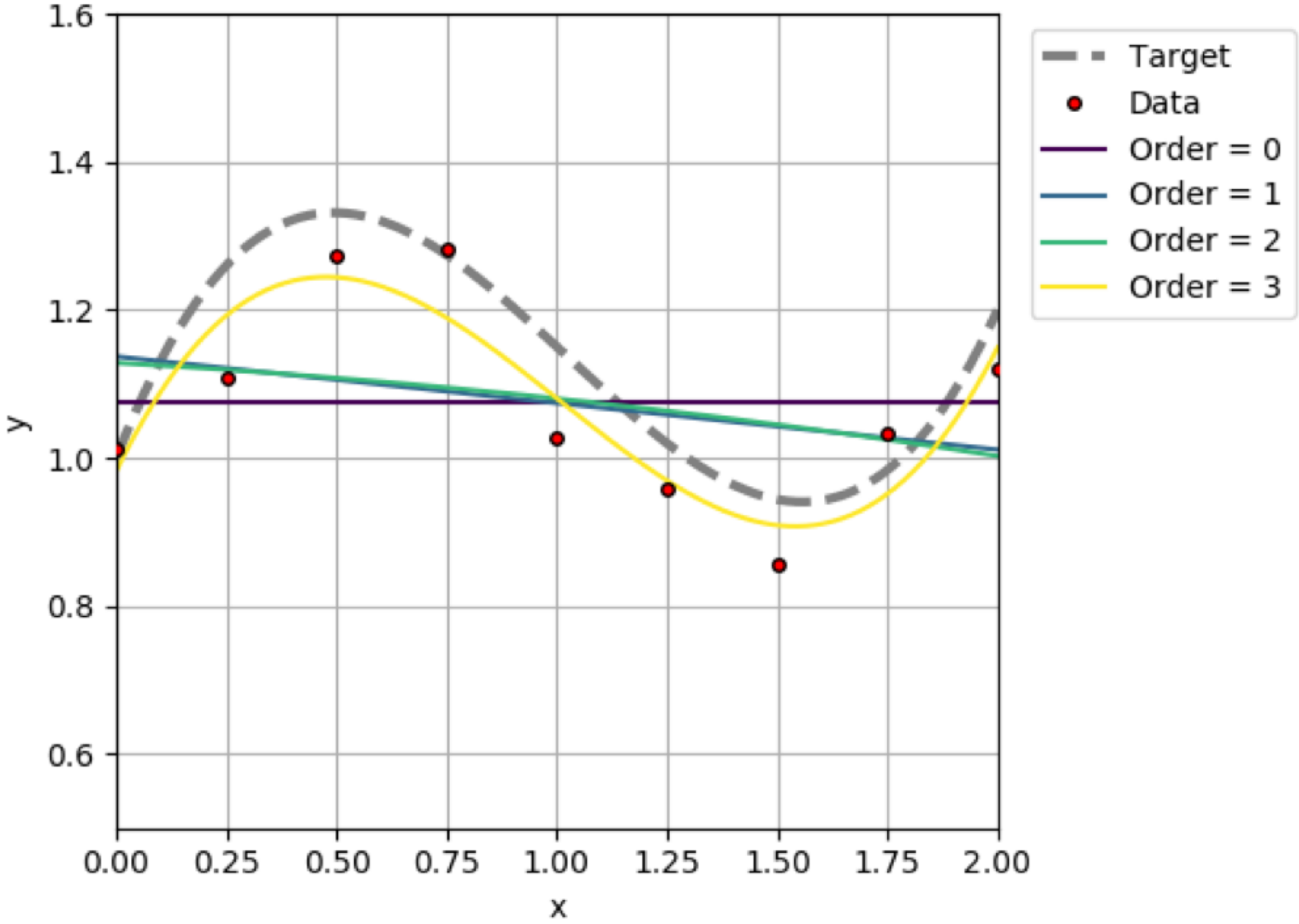
N is the model order



Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

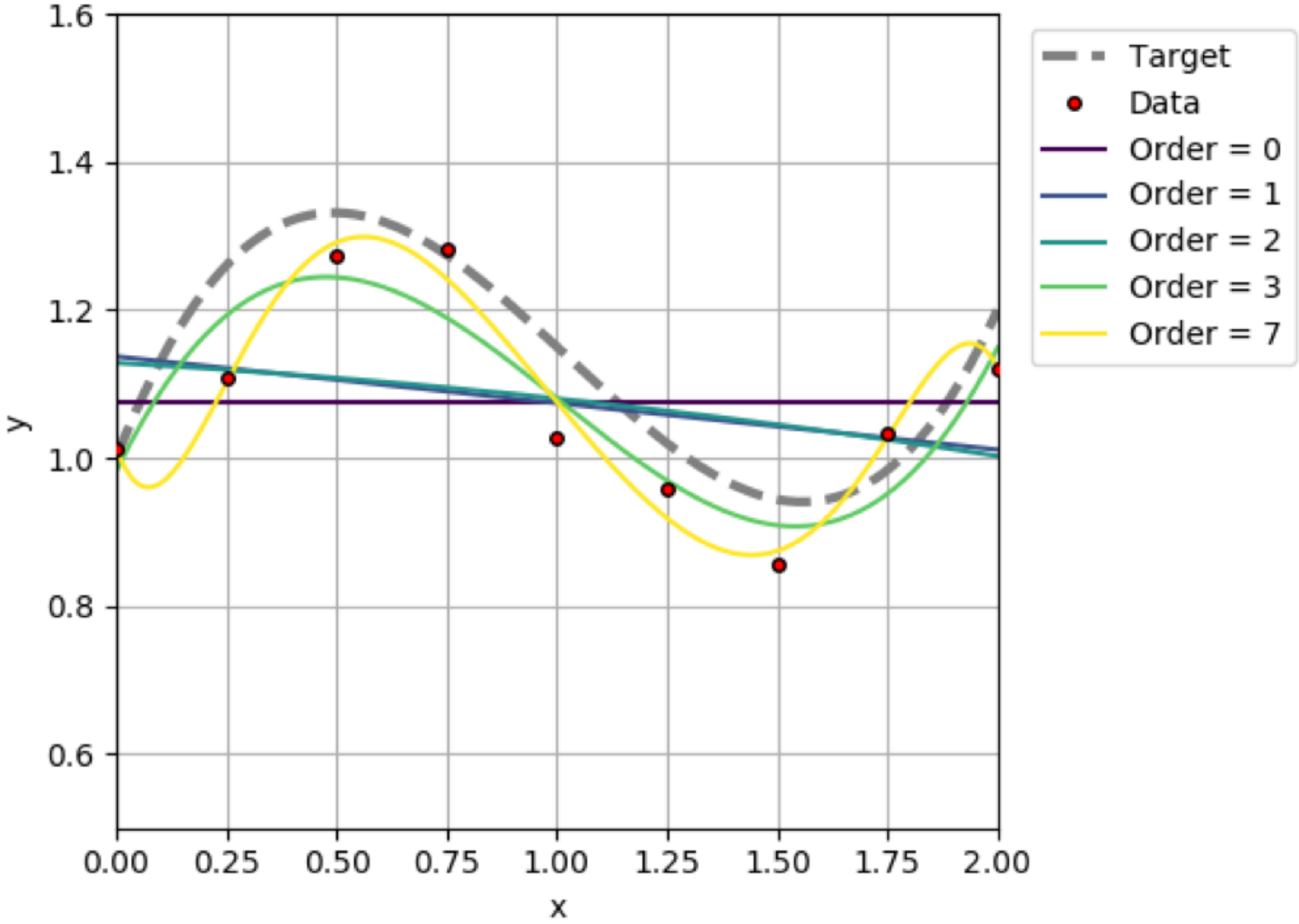
N is the model order



Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

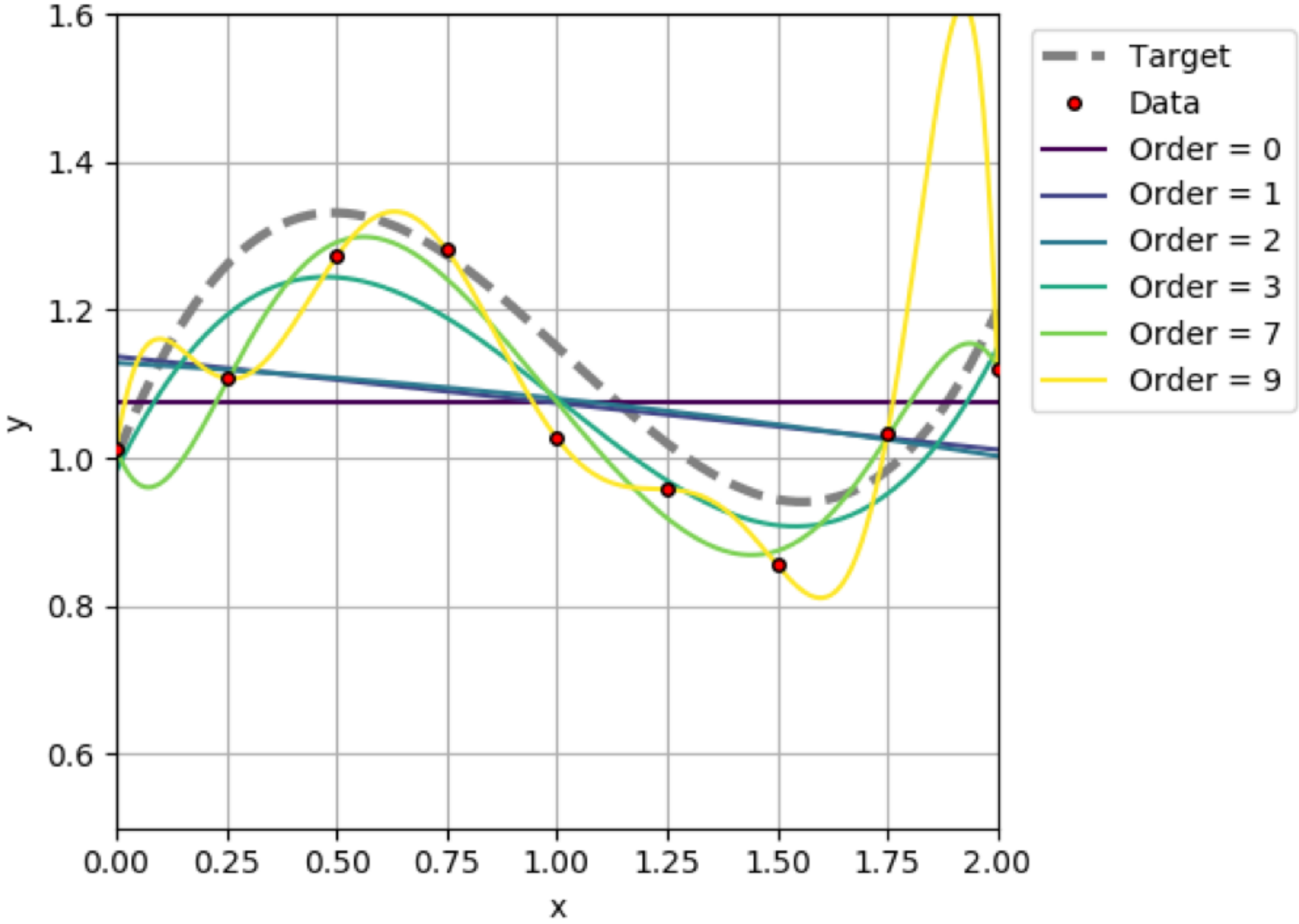
N is the model order



Linear Regression

$$\hat{y}_i = \sum_{j=0}^{N} a_j x_i^j$$

N is the model order



Problem

Too much flexibility leads to **overfit**

Too little flexibility leads to **underfit**

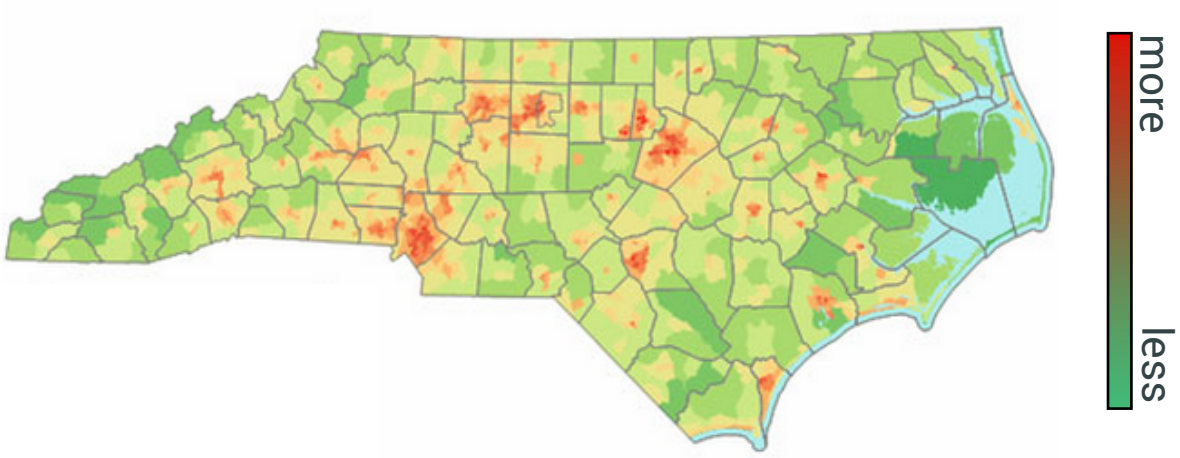
Over/underfit **hurts generalization** performance

Solutions

1. Add more data for training
2. Constrain model flexibility through **regularization**

Kaggle Competition

Goal: estimate solar array locations, power capacity, and energy generation



Sample graph from U.S. Census Bureau showing 2000 Population

Public estimates of distributed solar are generally limited to state or national scales (U.S. EIA)

High resolution estimates of solar photovoltaic (PV) array locations and energy generation are:

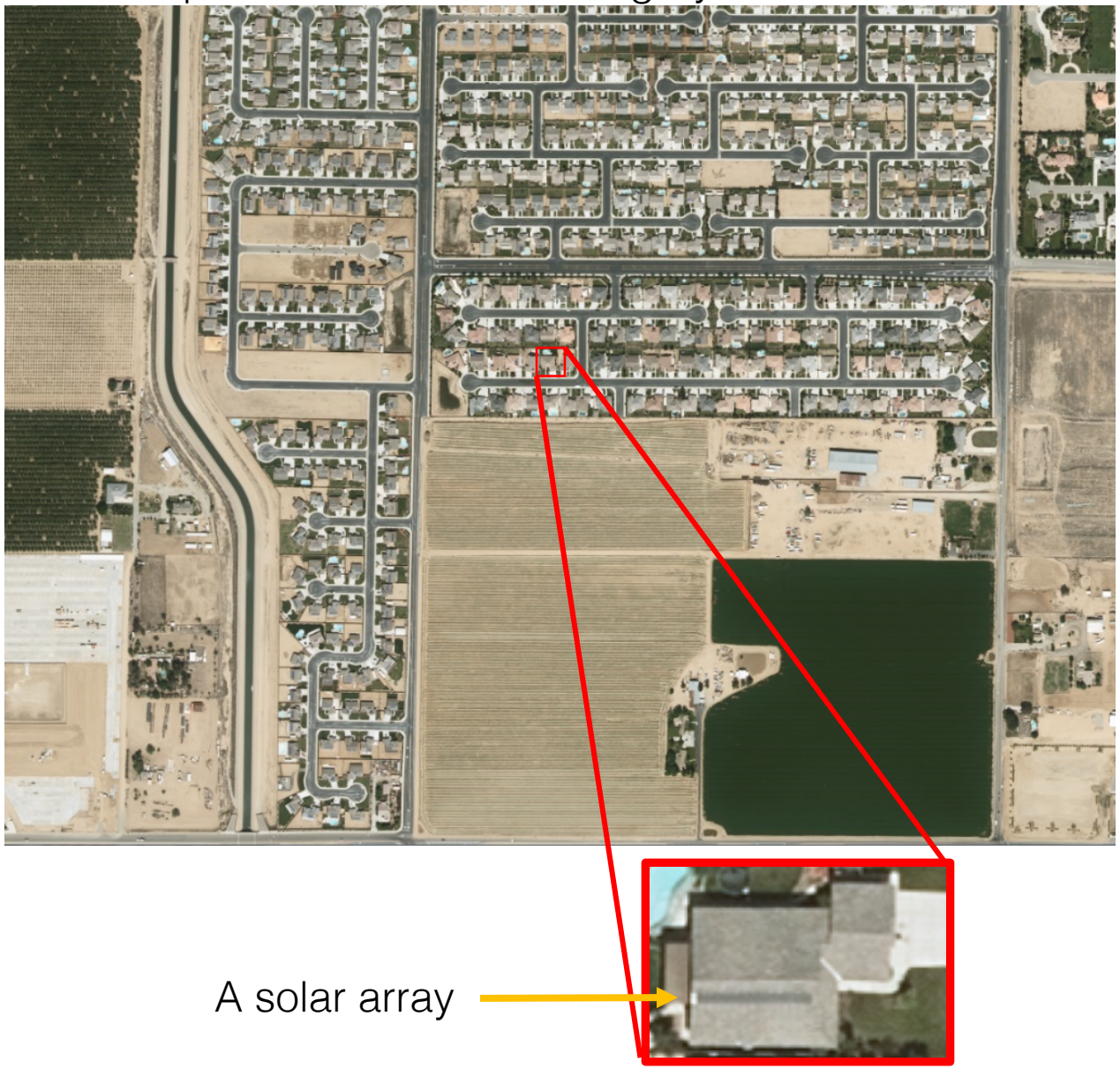
1. Difficult to obtain through surveys, public utility commissions, or utility NDAs
2. Useful for local and national decision-makers (for planning and operations)

A machine learning solution

Use computer algorithms to automatically identify panels in satellite imagery

Benefits: cheap, fast, scalable, and high resolution

Example of color satellite imagery

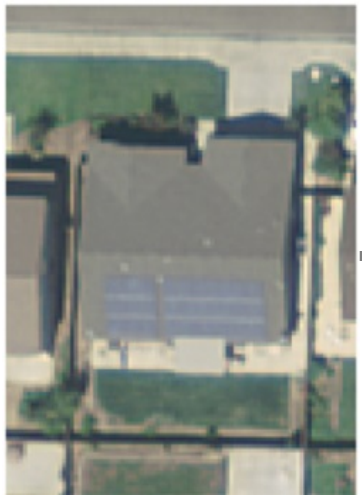


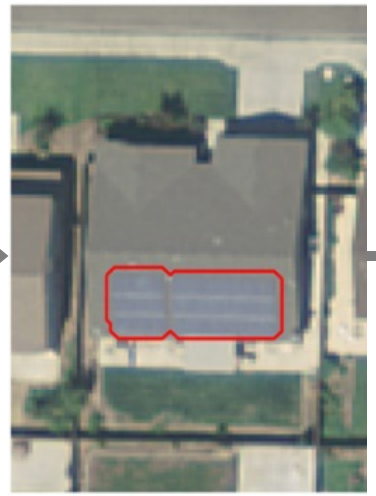
Estimating Energy from Distributed Solar Using Satellite Imagery

Process

Input Satellite Imagery → Solar Array Detection → Power Capacity Estimation → Generated Energy Estimation ← Regional insolation data → High-resolution estimates of solar energy

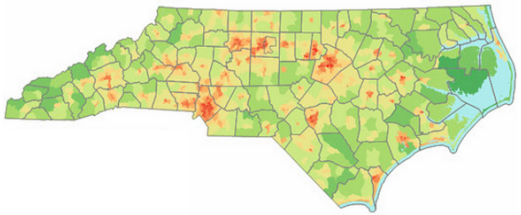
Example





Malof et al. 2017

4,750 Watts → 6.94 MWh →

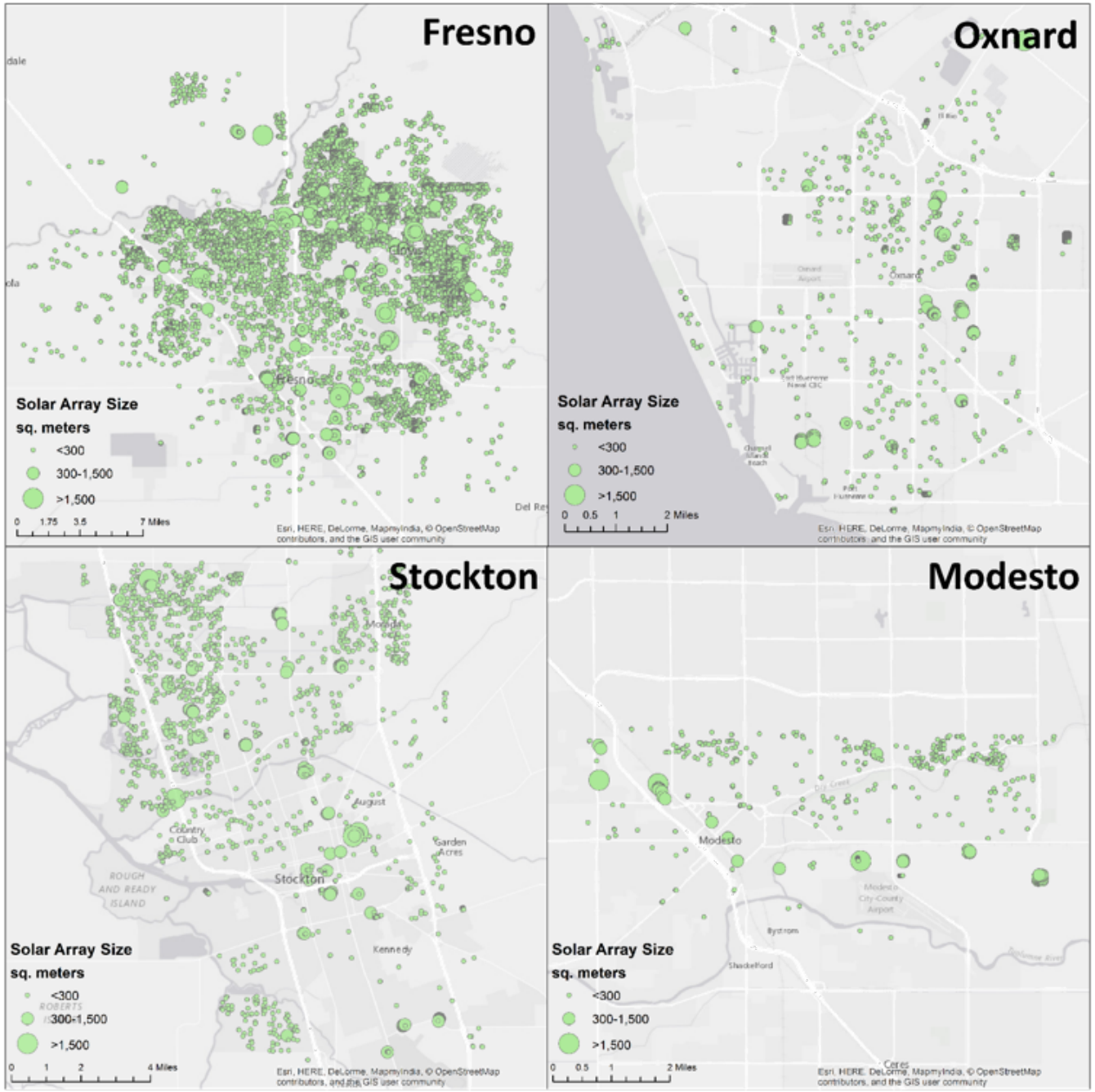


Sample graph from U.S. Census Bureau showing 2000 Population

Solar PV Dataset

19,000+ manually annotated solar arrays from 4 CA cities

Include precise polygonal outlines for each array



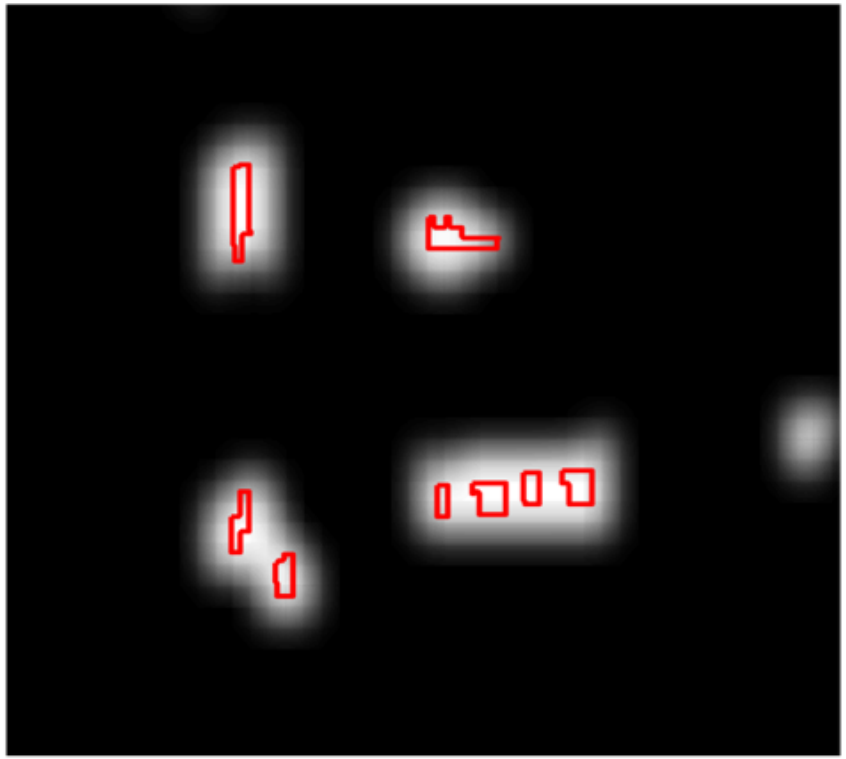
Bradbury, K., Saboo, R., Johnson, T., Malof, J., Zhang, W., Devarajan, A., Collins, L., Newell, R. (2016) "Distributed Solar Photovoltaic Array Location and Extent Dataset for Remote Sensing Object Identification." Scientific Data, 3. doi:10.1038/sdata.2016.106.

Available for download at:
https://doi.org/10.6084/m9.figshare.c.3255643.v2

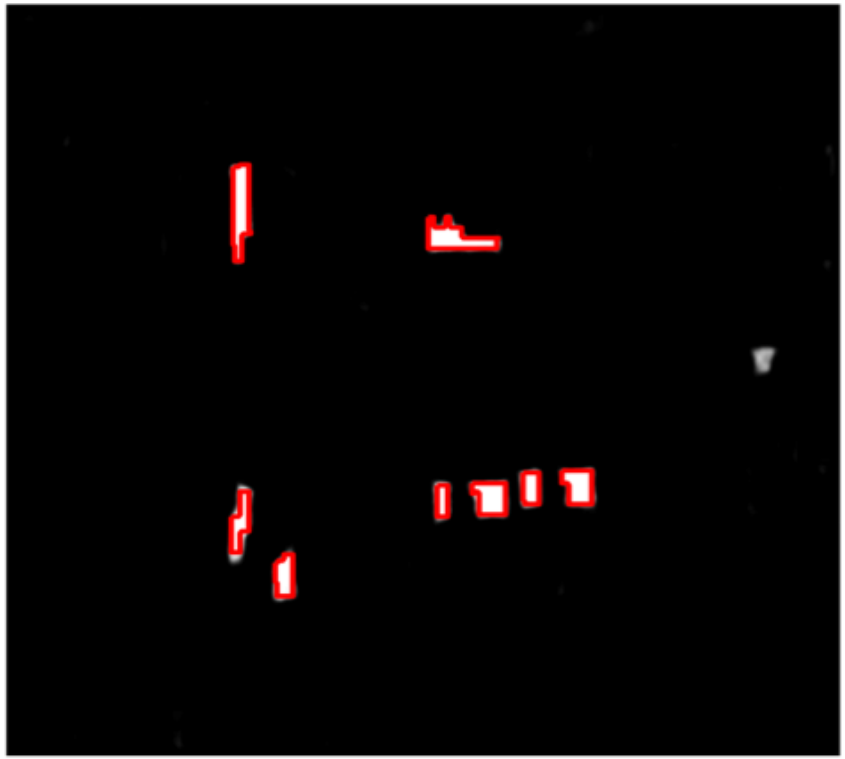
Illustrative segmentation results

Original Image

VGG CNN



SegNet



Camilo, J., L. Collins, K. Bradbury, J. Malof. “Application of a semantic segmentation convolutional neural network for accurate automatic detection and mapping of solar photovoltaic arrays in aerial imagery.” Presented at the *IEEE Applied Imagery Pattern Recognition Workshop* in Washington, D.C., 2017.

Your objective for Kaggle: binary classification

Data

Is there a solar array present in the image?

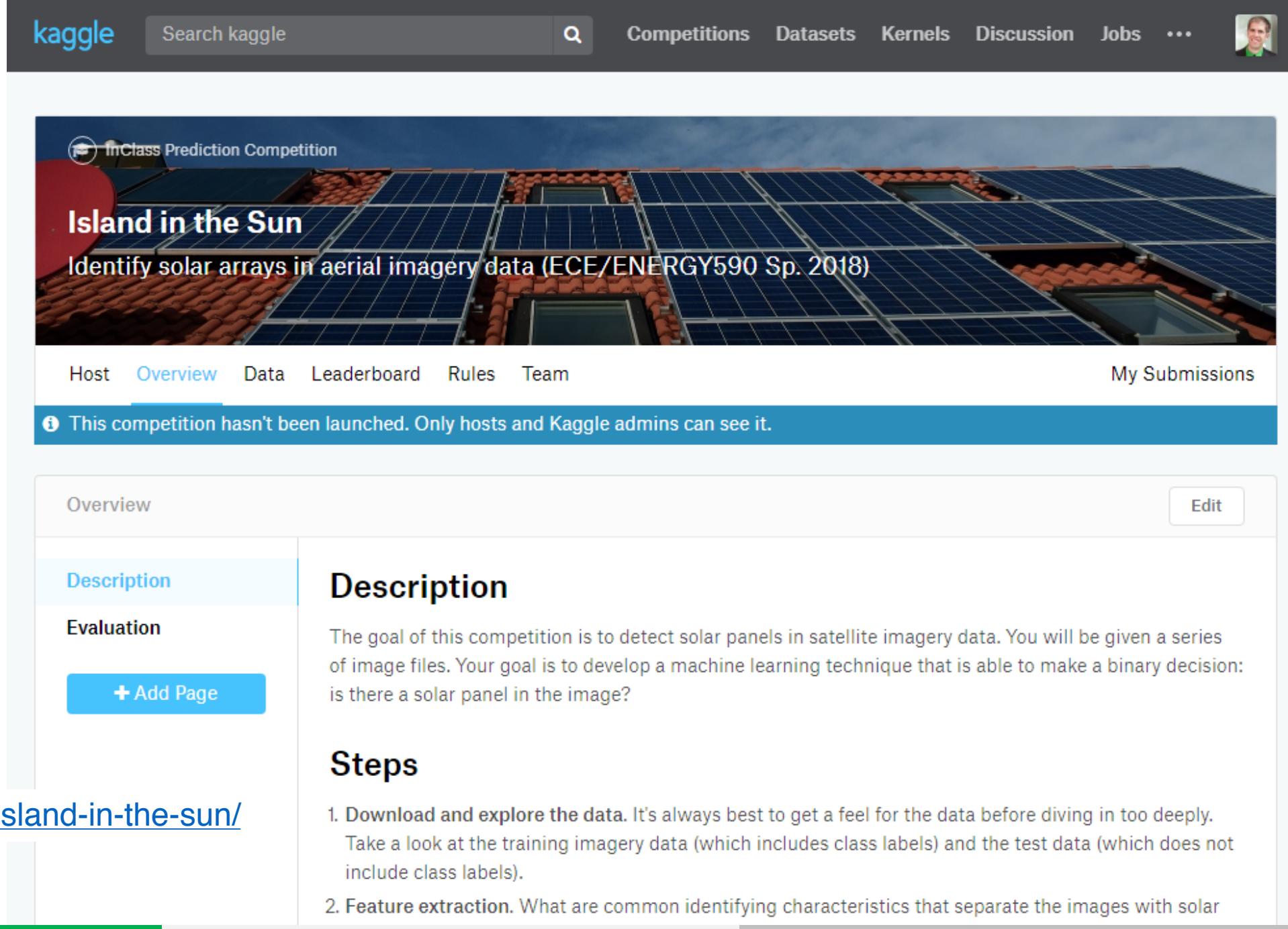
No

Yes

Logistics

- You will work in teams of 3 or 4
- The competition begins TODAY and concludes on February 28
- Prizes:
  - First place: 3 bonus points on your Kaggle report
  - Second place: 1 bonus point on your Kaggle report

- Website: https://www.kaggle.com/c/island-in-the-sun/
- You can make up to 3 submissions per day
- Sample code is provided to get you started



https://www.kaggle.com/c/island-in-the-sun/

Teams

Ke,Yujing	1
Bader,Hayden P.	1
Shen,Yangdi	1
Fang,Qian	2
Dai,Shuyang	2
Ringel,Morgan Jill	2
Li,Yifan	3
Inkawhich,Nathan Albert	3
Sapozhnikov,Lisa	3
Li,Yue	4
Lesi,Vuk	4
Bognar,Mitchell Jacob	4
Wan,Hui	5
Mondal,Sudipta	5
Farley,Virginia Meux	5
Zhang,Jingwen	6
Xie,Zhiyao	6
Shankar,Varun Jagannathan	6
Zhang,Rongsheng	7
Kaur,Gagandeep (Gagan)	7
Zhou,Jin	7
Chen,Shiwen	7