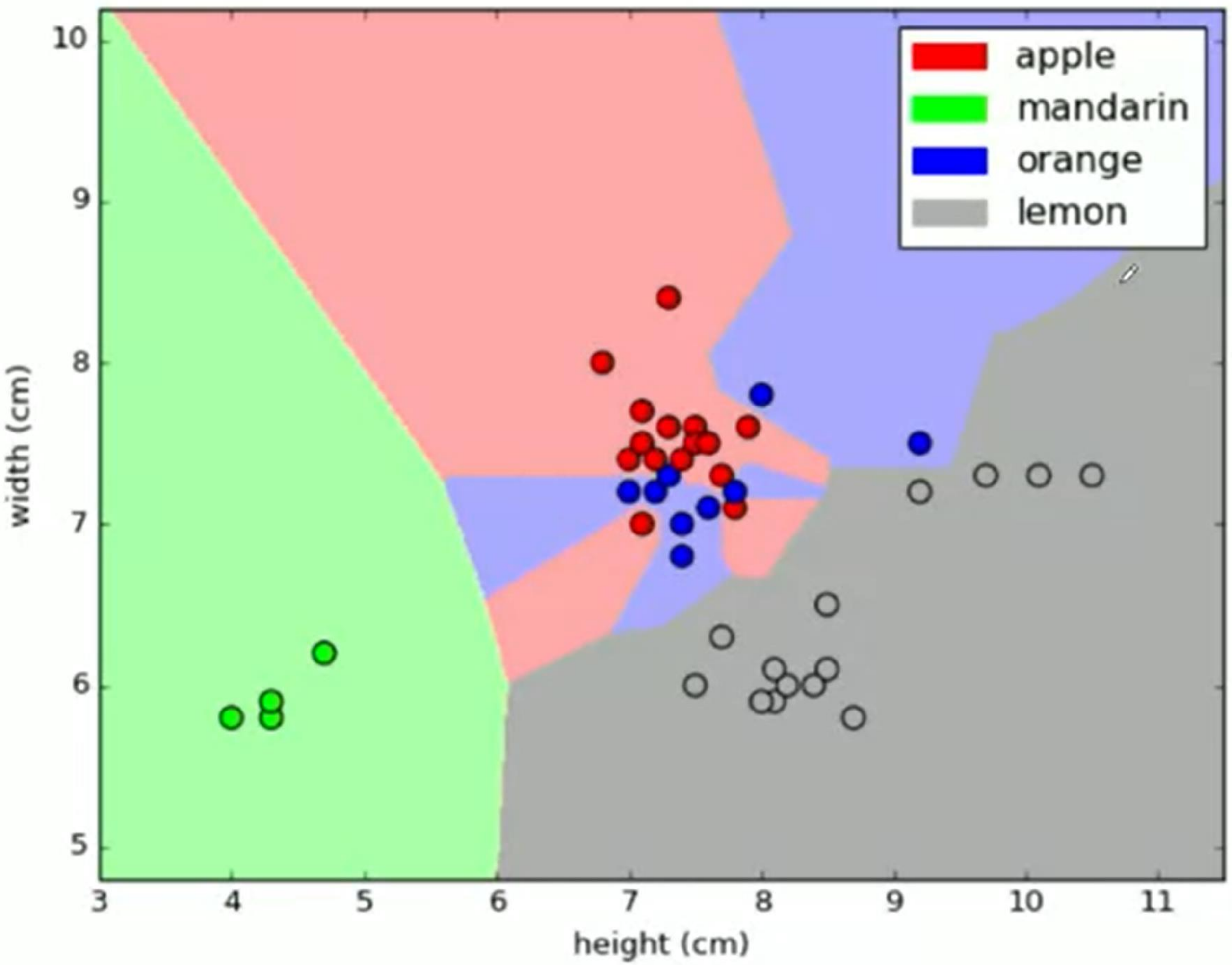
The k-Nearest Neighbor (k-NN) Classifier Algorithm

Given a training set X_train with labels y_train, and given a new instance x_test to be classified:

1. Find the most similar instances (let's call them X_NN) to x_test that are in X_train.
2. Get the labels y_NN for the instances in X_NN
3. Predict the label for x_test by combining the labels y_NN
   e.g. simple majority vote

A visual explanation of k-NN classifiers

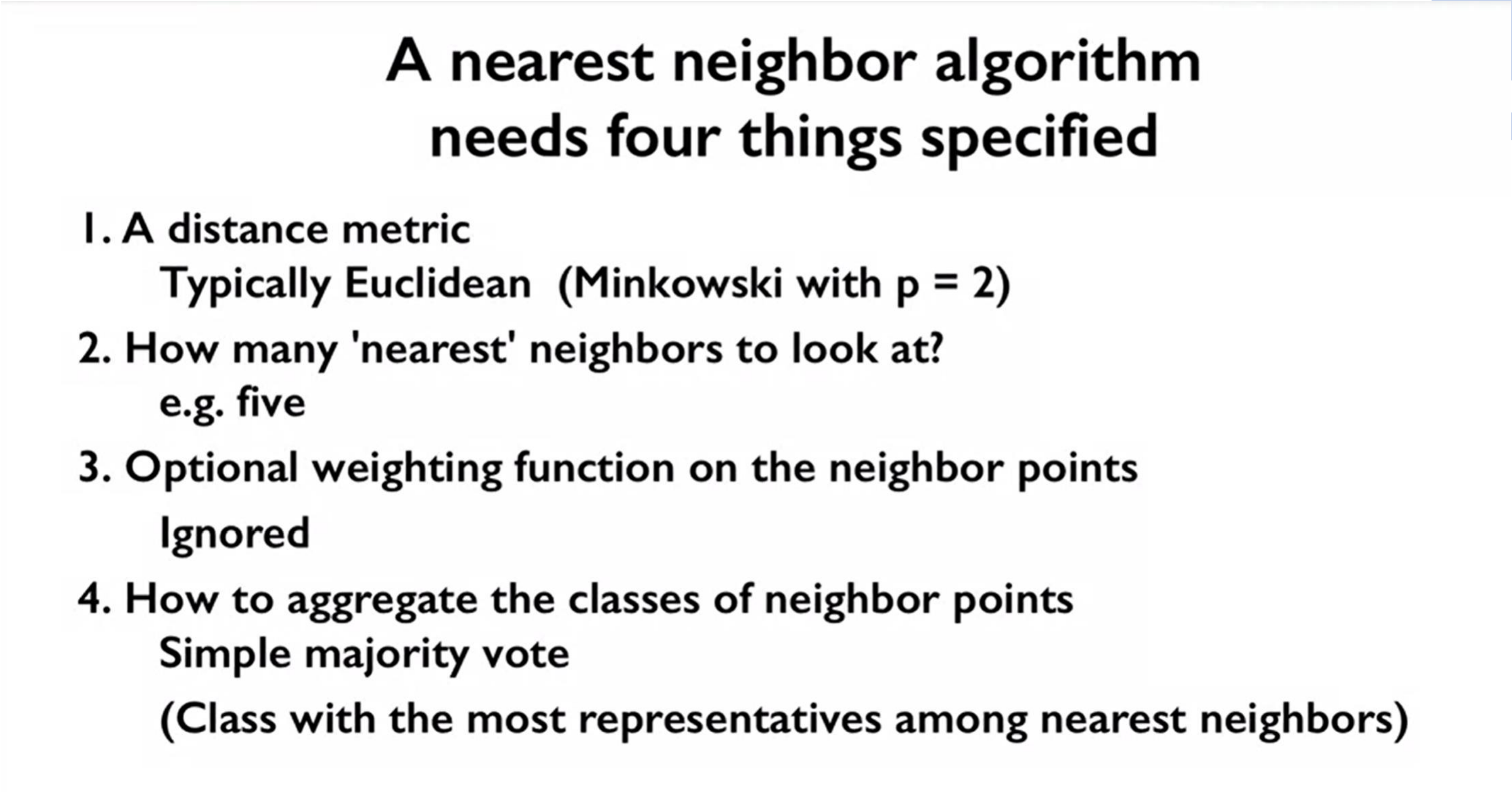


Fruit dataset
Decision boundaries
with k = 1

- Query point

A nearest neighbor algorithm needs four things specified

1. A distance metric
2. How many 'nearest' neighbors to look at?
3. Optional weighting function on the neighbor points
4. Method for aggregating the classes of neighbor points



A nearest neighbor algorithm needs four things specified

1. A distance metric

   Typically Euclidean (Minkowski with p = 2)

2. How many 'nearest' neighbors to look at?

   e.g. five

3. Optional weighting function on the neighbor points

   Ignored

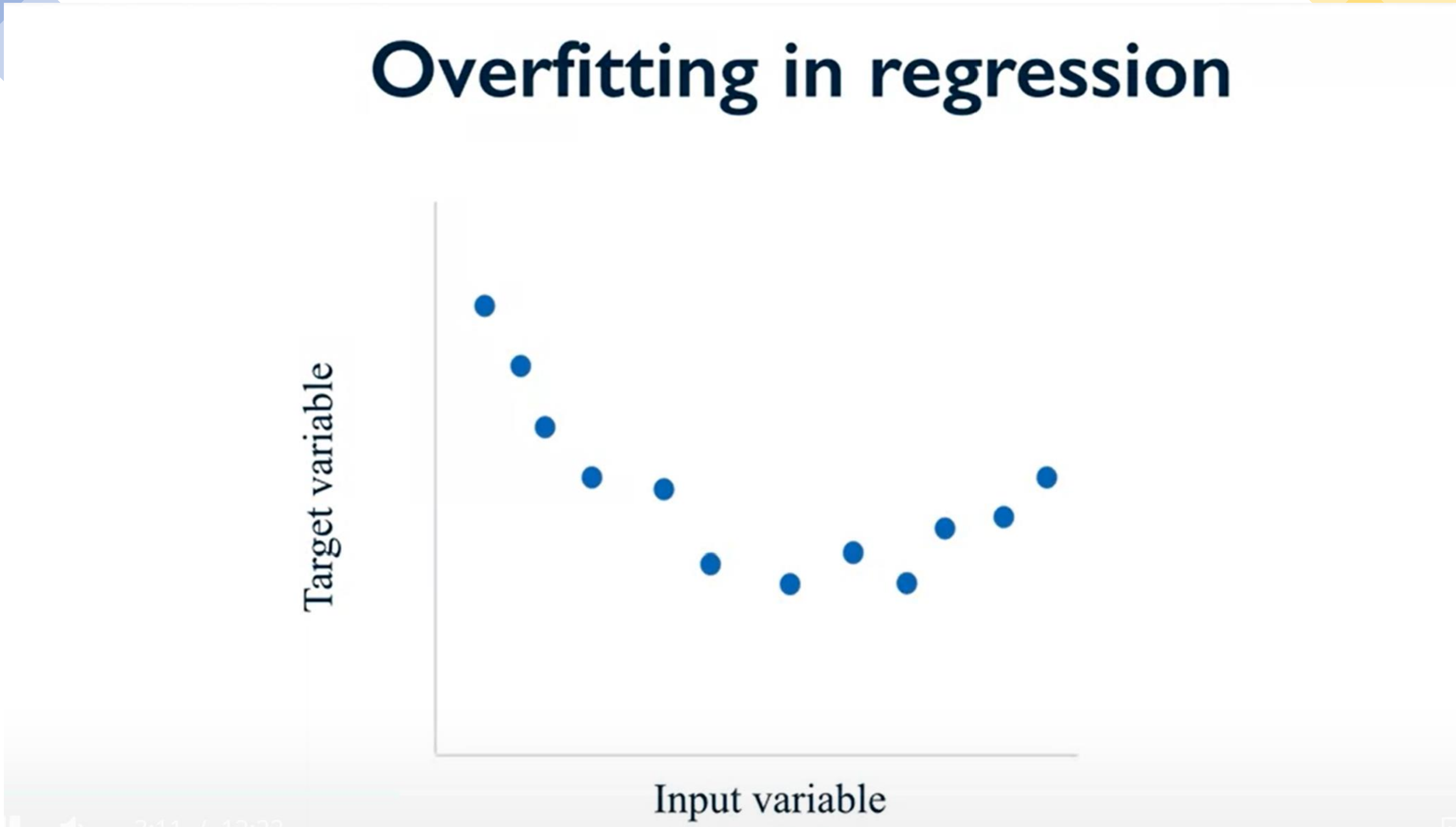
4. How to aggregate the classes of neighbor points

   Simple majority vote

   (Class with the most representatives among nearest neighbors)

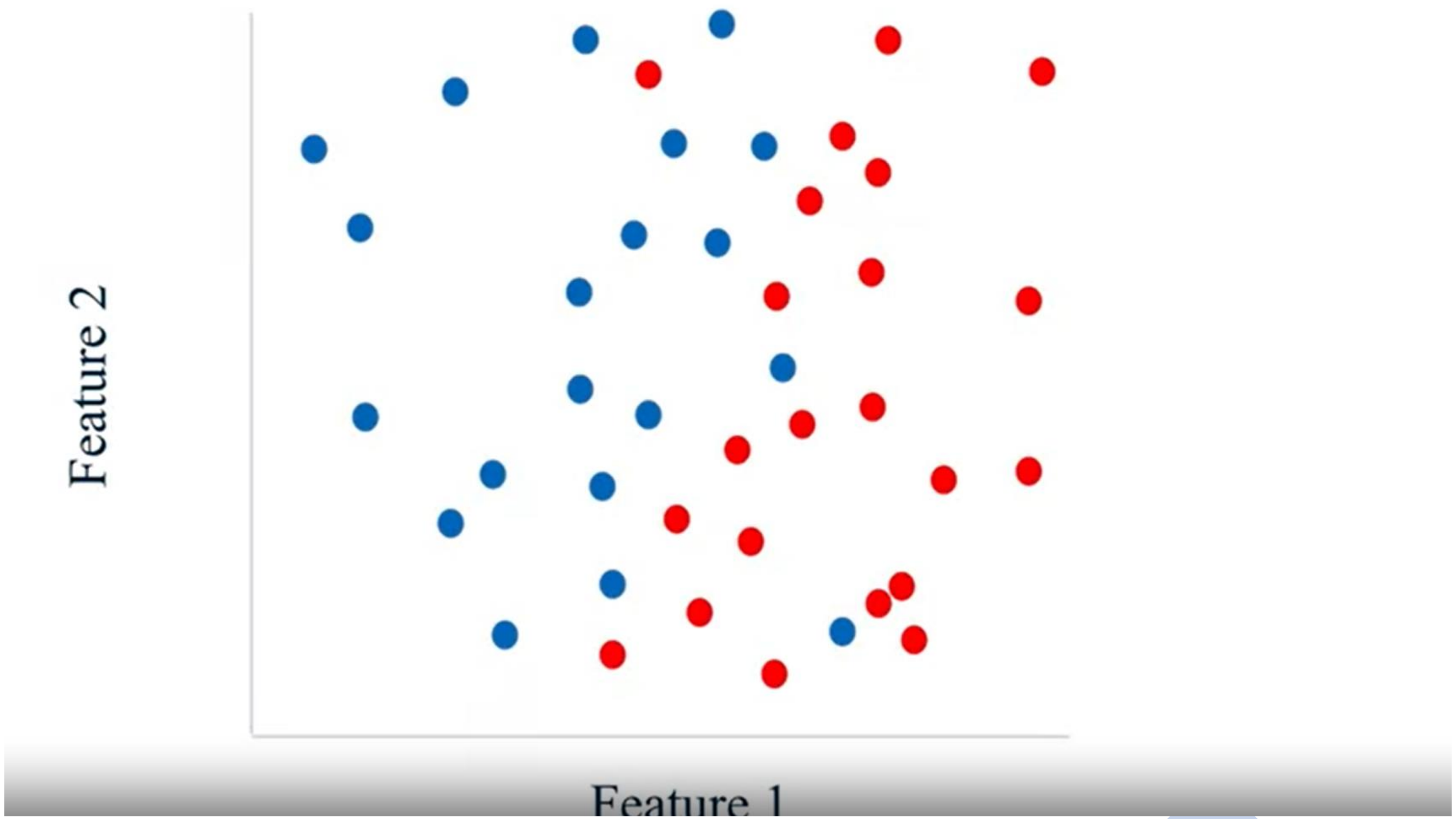
Generalization, Overfitting, and Underfitting

- Generalization ability refers to an algorithm's ability to give accurate predictions for new, previously unseen data.
- Assumptions:
  - *Future unseen data (test set) will have the same properties as the current training sets.*
  - *Thus, models that are accurate on the training set are expected to be accurate on the test set.*
  - *But that may not happen if the trained model is tuned too specifically to the training set.*
- Models that are too complex for the amount of training data available are said to overfit and are not likely to generalize well to new examples.
- Models that are too simple, that don't even do well on the training data, are said to underfit and also not likely to generalize well.

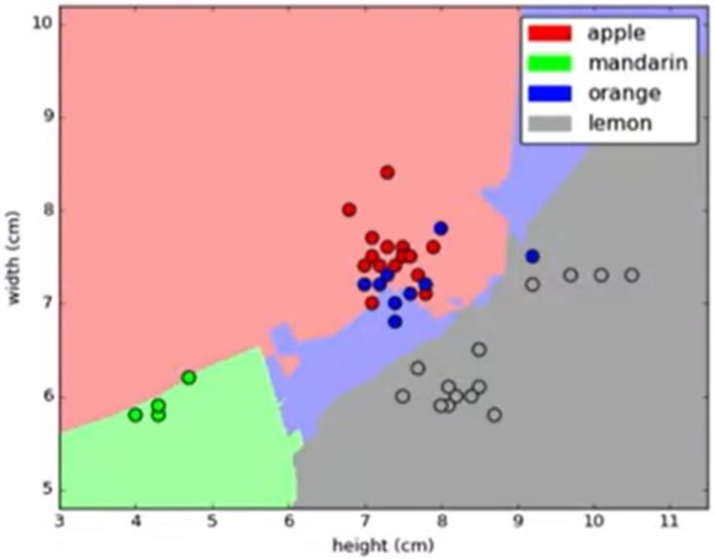


Overfitting in regression

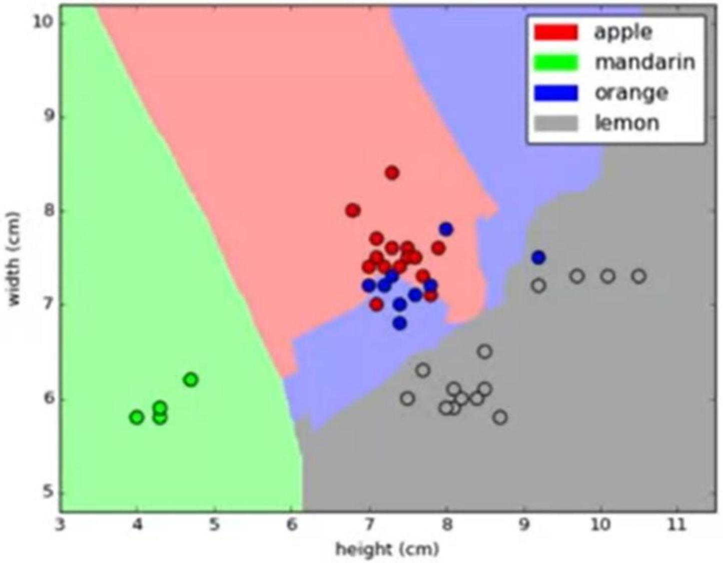
Overfitting in classification



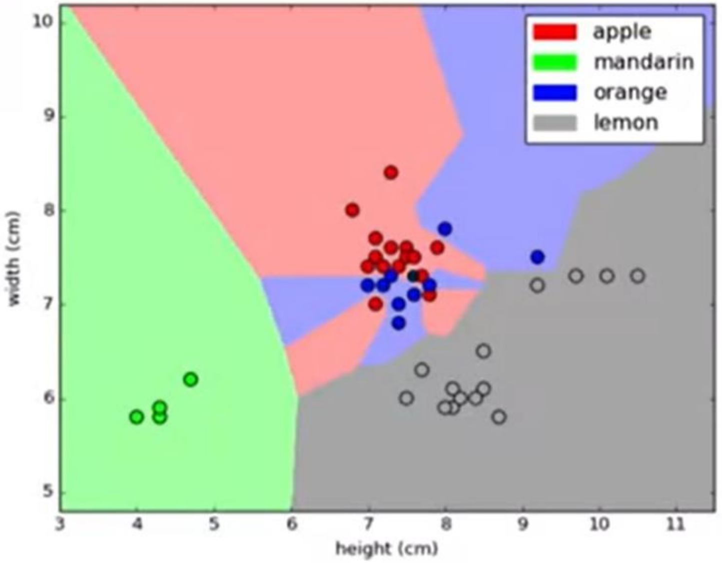
Overfitting with k-NN classifiers



K=10

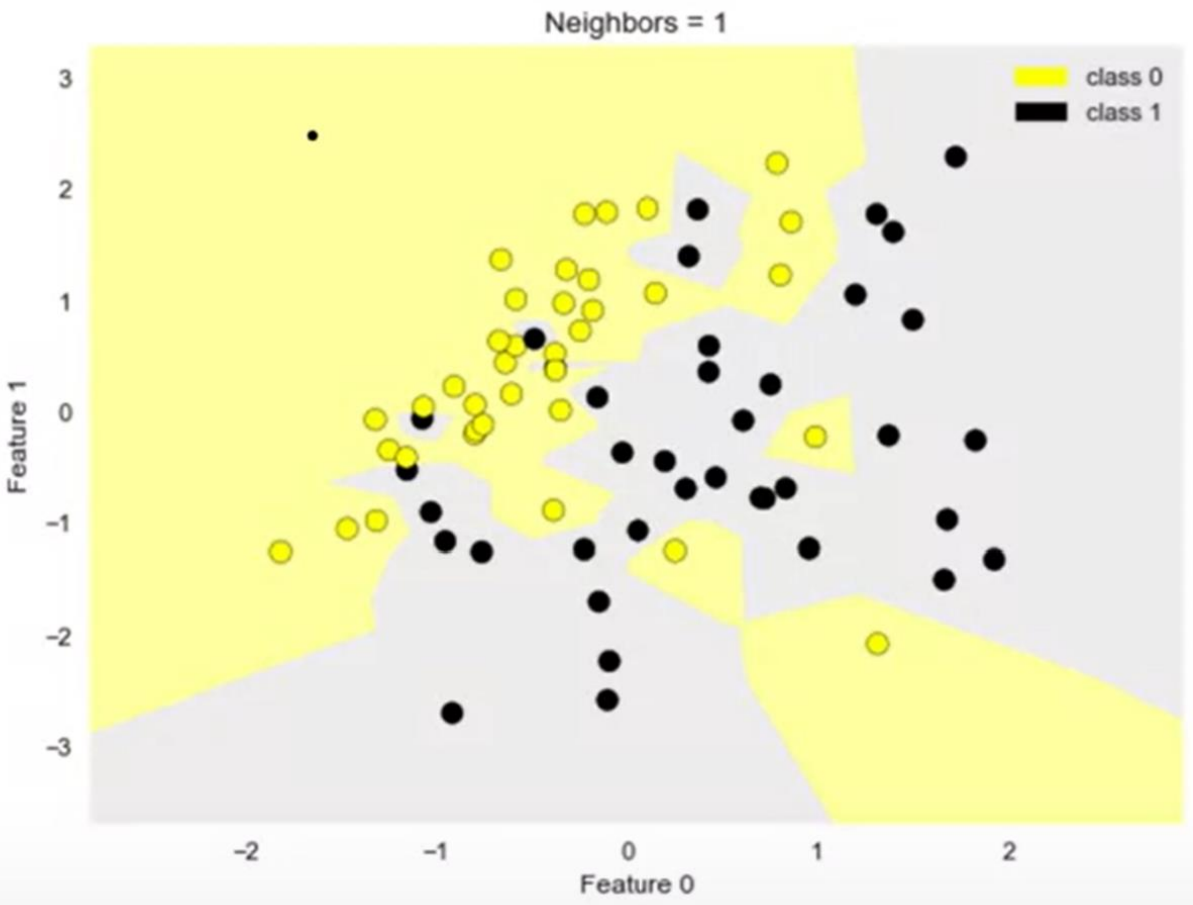


K=5

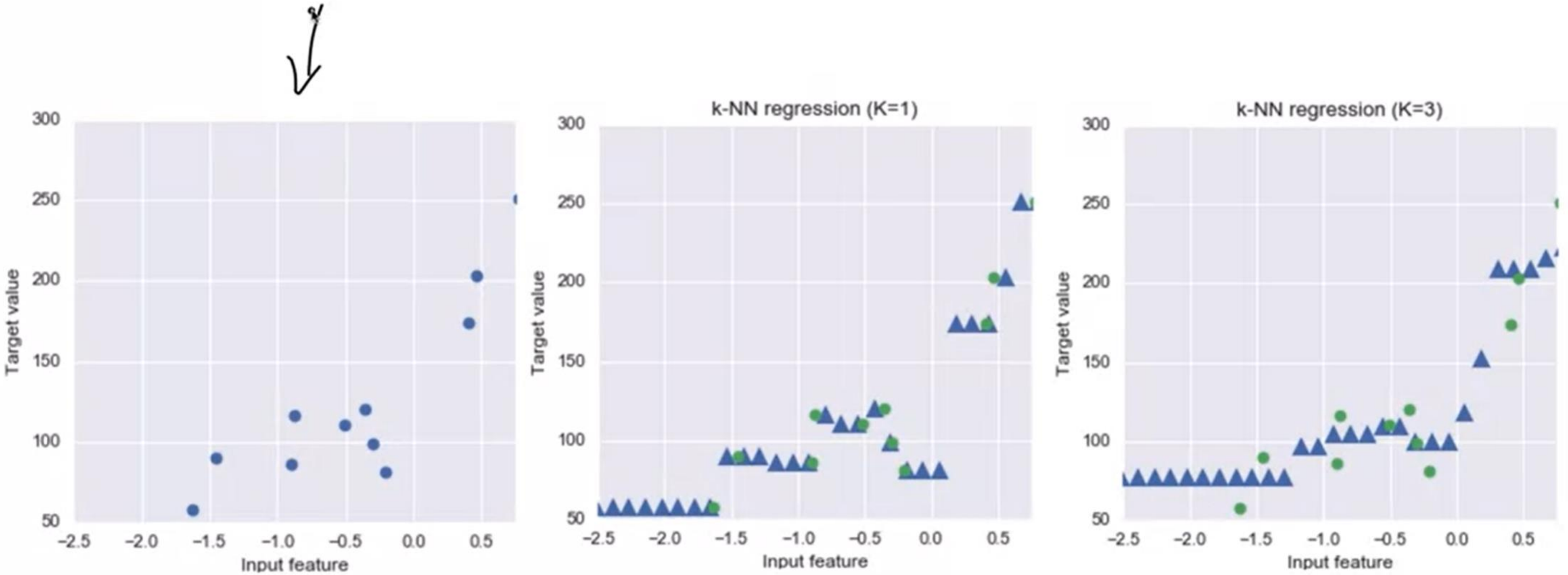


K=1

Nearest neighbors classification (k=1)



k-Nearest neighbors regression



The R^2 ("r-squared") regression score

- Measures how well a prediction model for regression fits the given data.
- The score is between 0 and 1:
  - *A value of 0 corresponds to a constant model that predicts the mean value of all training target values.*
  - *A value of 1 corresponds to perfect prediction*
- Also known as "coefficient of determination"

KNeighborsClassifier and KNeighborsRegressor: important parameters

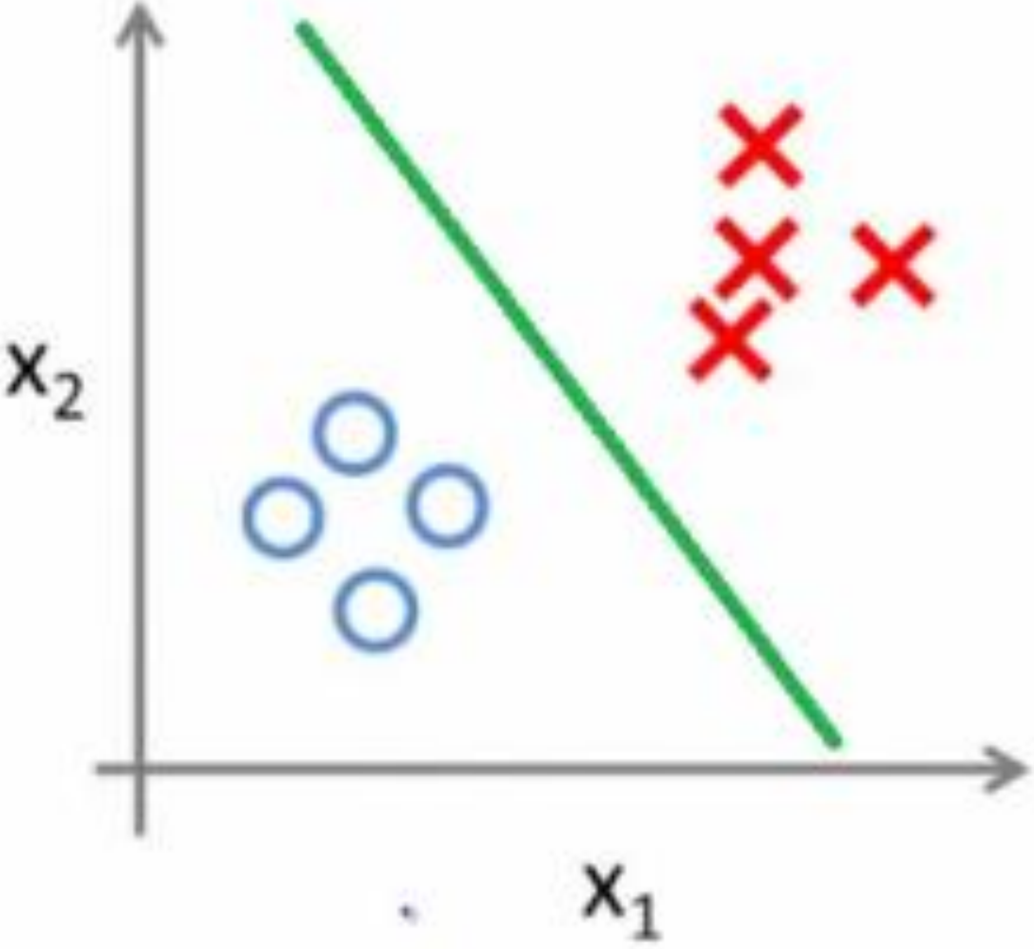
__Model complexity__

- *n_neighbors :* number of nearest neighbors (k) to consider
  - *Default = 5*

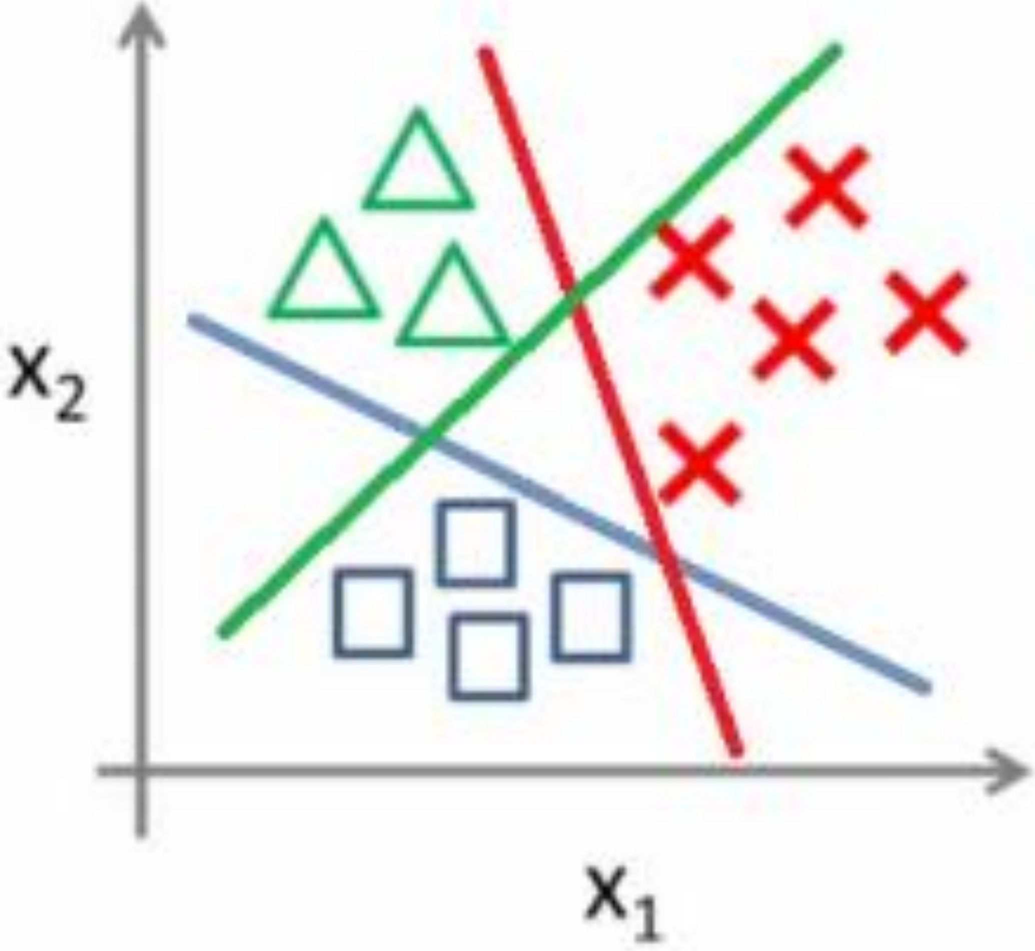
__Model fitting__

- *metric: distance function between data points*
  - *Default: Minkowski distance with power parameter p = 2 (Euclidean)*

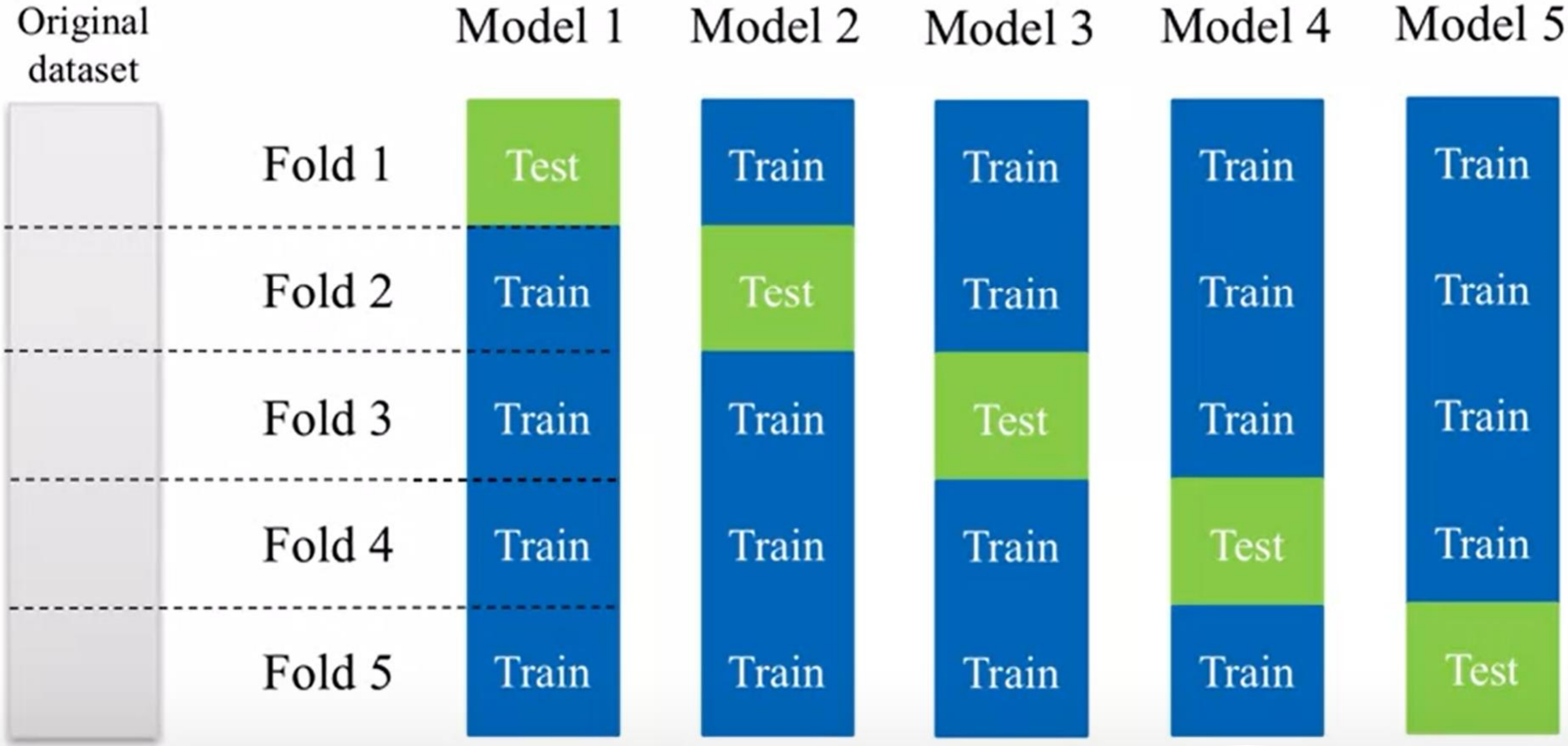
Binary classification:



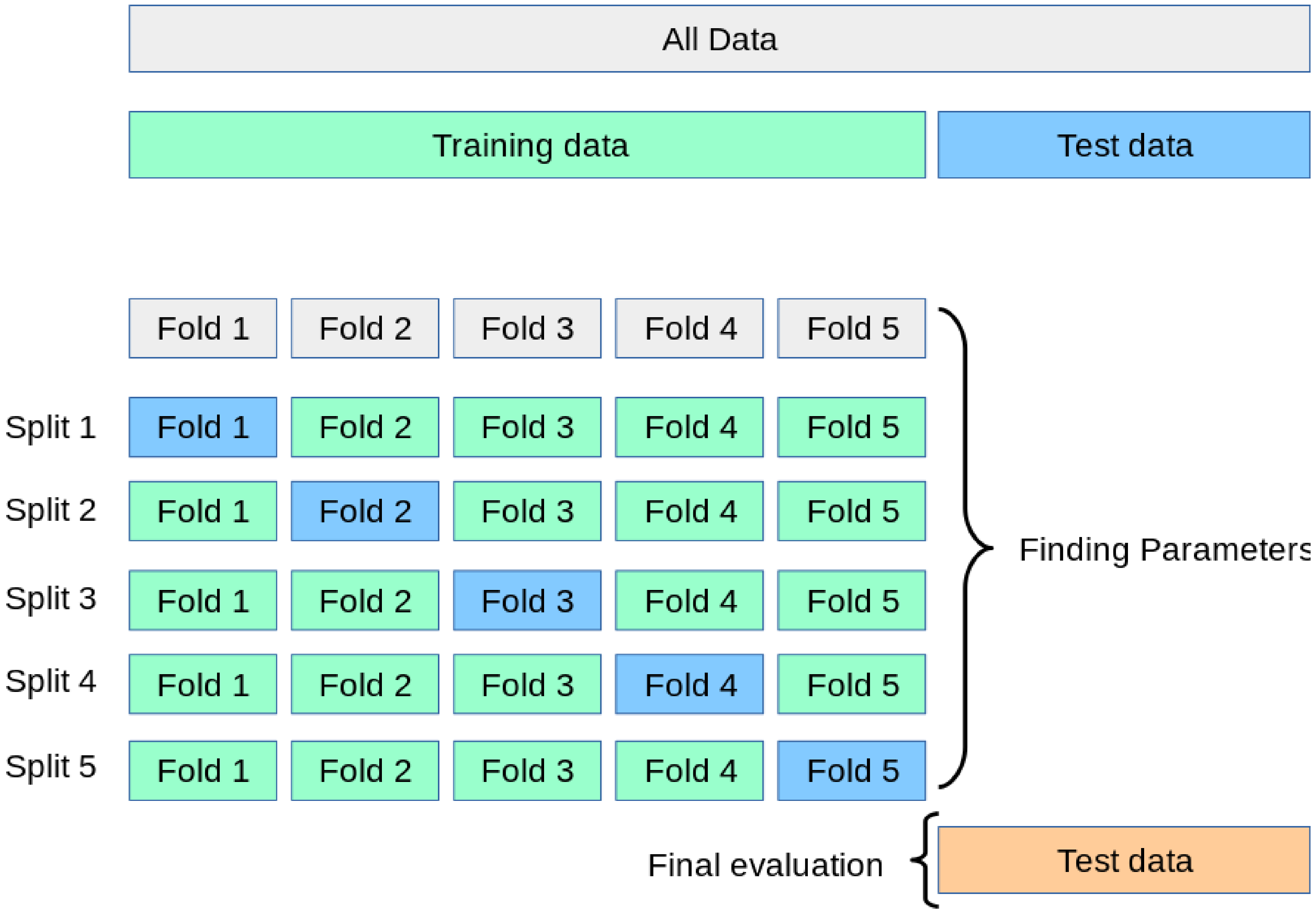
Multi-class classification:



Cross-validation Example (5-fold)



Original dataset		Model 1	Model 2	Model 3	Model 4	Model 5
	Fold 1	Test	Train	Train	Train	Train
	Fold 2	Train	Test	Train	Train	Train
	Fold 3	Train	Train	Test	Train	Train
	Fold 4	Train	Train	Train	Test	Train
	Fold 5	Train	Train	Train	Train	Test



All Data

Training data | Test data

Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 1: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 2: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 3: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 4: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Split 5: Fold 1 | Fold 2 | Fold 3 | Fold 4 | Fold 5

Finding Parameters

Final evaluation: Test data

Feature Normalization

- Normalization is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. It is also known as **Min-Max scaling**

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Standardization

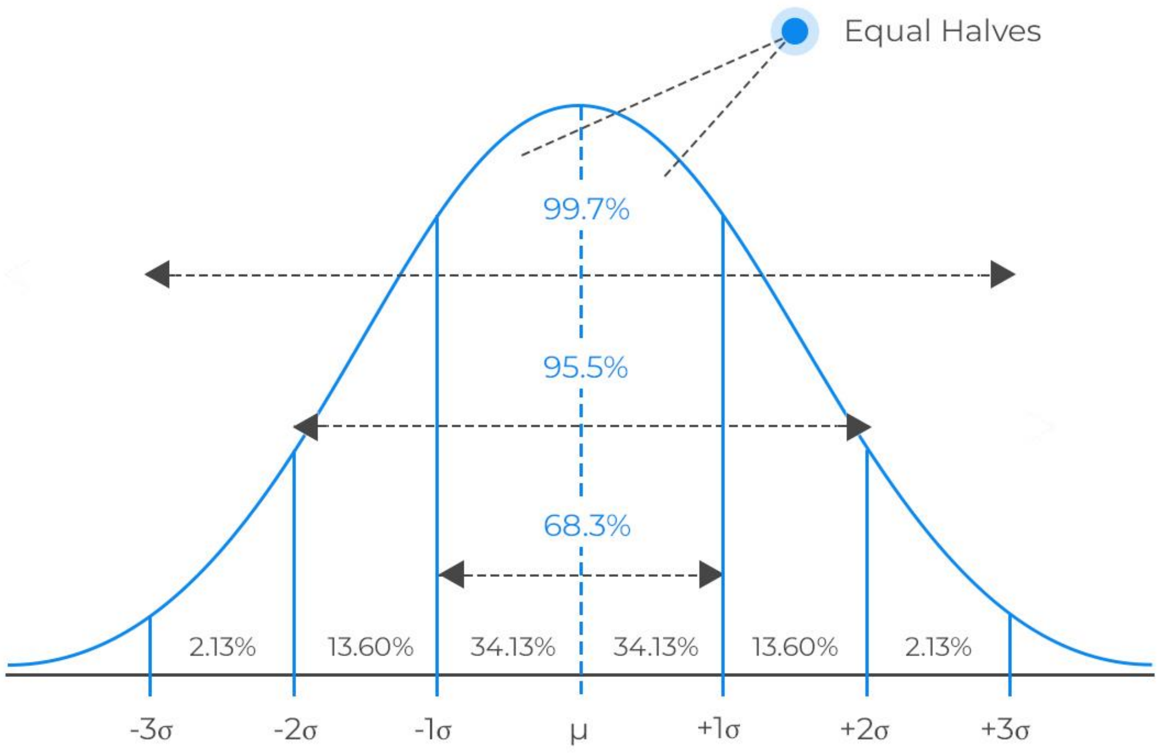
- **Standardization is another scaling technique where the values are centered around the mean with a unit standard deviation. This means that the mean of the attribute becomes zero and the resultant distribution has a unit standard deviation/ SD of 1.**

$$X' = \frac{X - \mu}{\sigma}$$

The Big Question – Normalize or Standardize?

- **Normalization** is good to use when you know that the distribution of your data does not follow a Gaussian distribution. This can be useful in algorithms that do not assume any distribution of the data like K-Nearest Neighbors and Neural Networks.
- **Standardization**, on the other hand, can be helpful in cases where the data follows a Gaussian distribution. However, this does not have to be necessarily true. Also, unlike normalization, standardization does not have a bounding range. So, even if you have outliers in your data, they will not be affected by standardization.

Shape of the normal distribution



No. of standard deviations from the mean

Need for Feature Normalization/standardization

- Some algorithms require that all features are on the same scale
- Faster convergence, ‘fair’ or uniform influence on the weights.