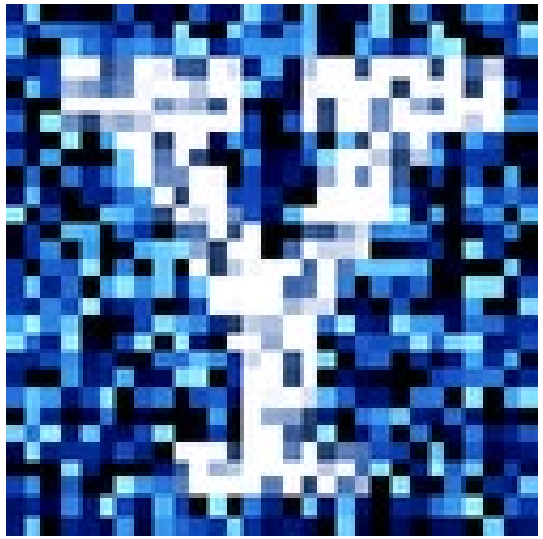
YData: Introduction to Data Science



Lecture 35: classifiers

Overview

Continuation of classification

Key concepts in classification

k nearest neighbor (kNN) classifier

Evaluating classifier performance

Overfitting

Announcements

Homework 10 has been posted

- It is due on Sunday the 24th

Practice 10 has been posted

- It is not due

Project 3 has been posted

- It is due Wednesday the 27th

Prediction: regression and clasification

We “learn” a function f

- f(**x**) ⟶ y

Input: **x** is a data vector of "features"

Output:

- Regression: output is a real number ($y \in R$)
- Classification: output is a categorical variable y_k

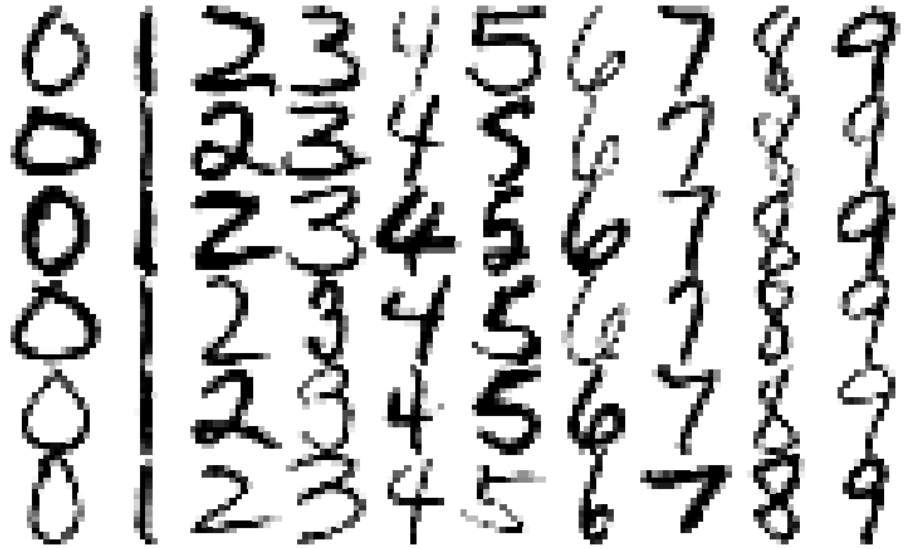
Example: salmon or sea bass?

What could be features in this task?

- Length, width, lightness, number and shape of fins, etc.

From Duda, Hart and Stork, 2001

Example: what is in this image?



What could be features in this task?

- Pixel values

Example: GPT-3 predicting/generating text

Question answering:

Are we living in a simulation?

Image generation

"Draw an astronaut riding a horse"

What could be features in this task?

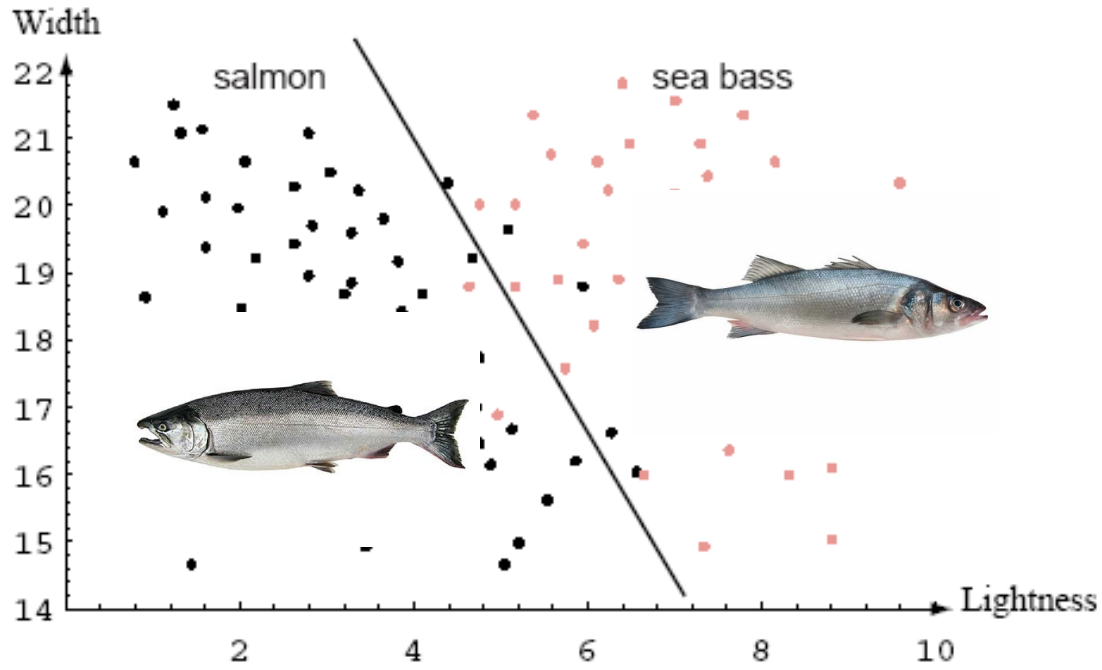
- Words

A few key concepts

A binary classifier is a function from the set of input features to {0, 1}

- E.g., f(pixel values) ⟶ salmon or bass

It is linear if we can draw a straight line (or a multi-dimensional plane) between the two predicted values



Let's explore this in Jupyter!

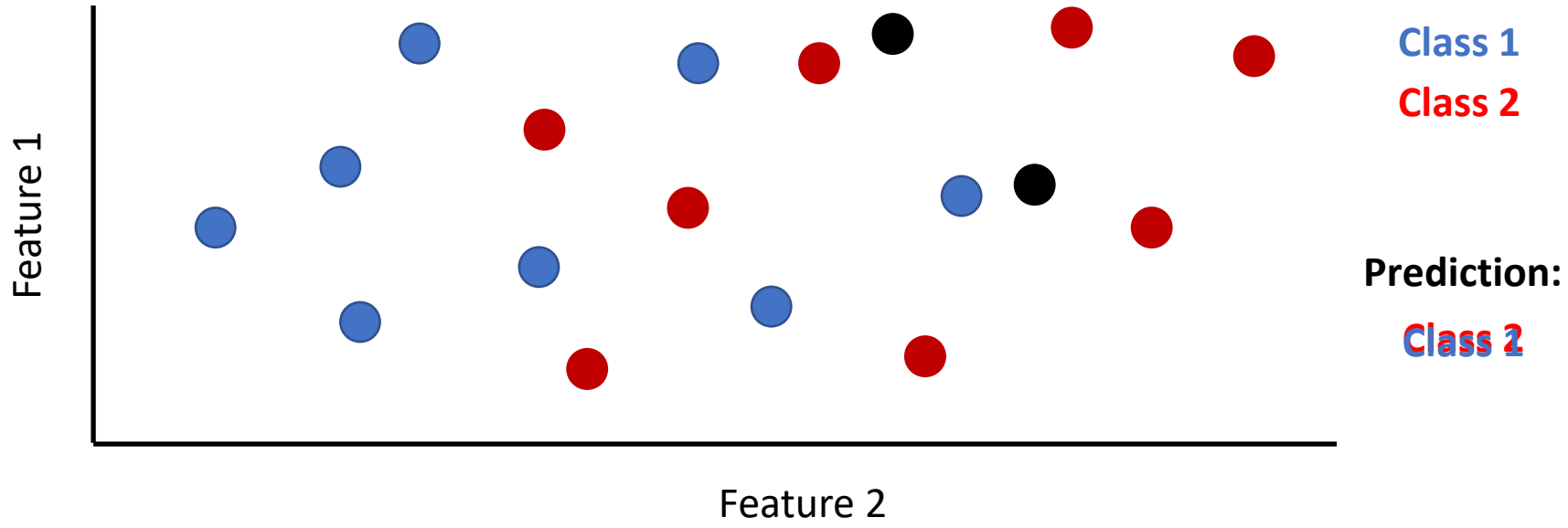
Training a classifier

Nearest Neighbor Classifier (k = 1)

Training the classifier: Store all the features with their labels

Making predictions: The label of closest training point is returned

- i.e., we find the distance between a test point and all training points, and the closest point is the prediction



Distance

Distance between two points

Two features x and y: $D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$

Three features x, y, and z: $D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2 + (z_0 - z_1)^2}$

- And so on for more features…

It's important the features are standardized

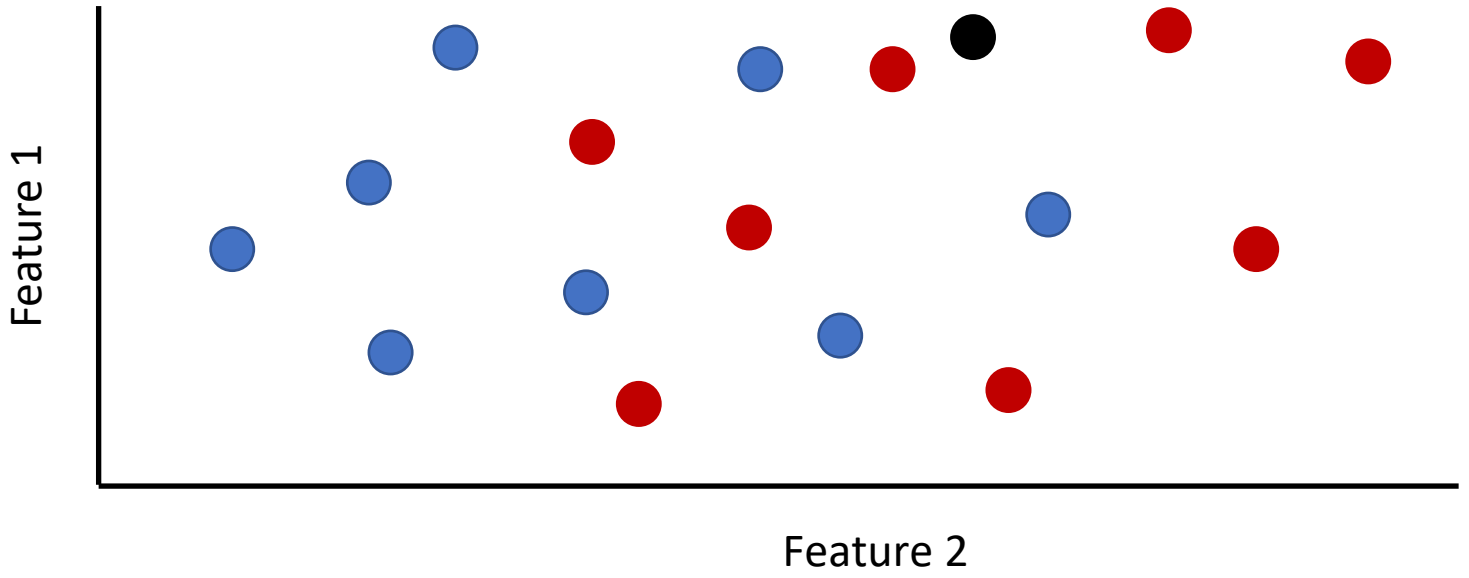
- If not, features that typically have larger values will dominate the distance measurement

Let's explore this in Jupyter!

Finding the k Nearest Neighbors (k ≥ 1)

To classify a point:

- Find its k nearest neighbors
- Take a majority vote of the k nearest neighbors to see which of the two classes appears more often
- Assign the point the class that wins the majority vote



Let's explore this in Jupyter!

Evaluation

Accuracy of a classifier

The accuracy of a classifier on a labeled data set is the proportion of examples that are labeled correctly on the *test set*

If the labeled data set is sampled at random from a population, then we can infer accuracy on that population

Let's explore this in Jupyter!

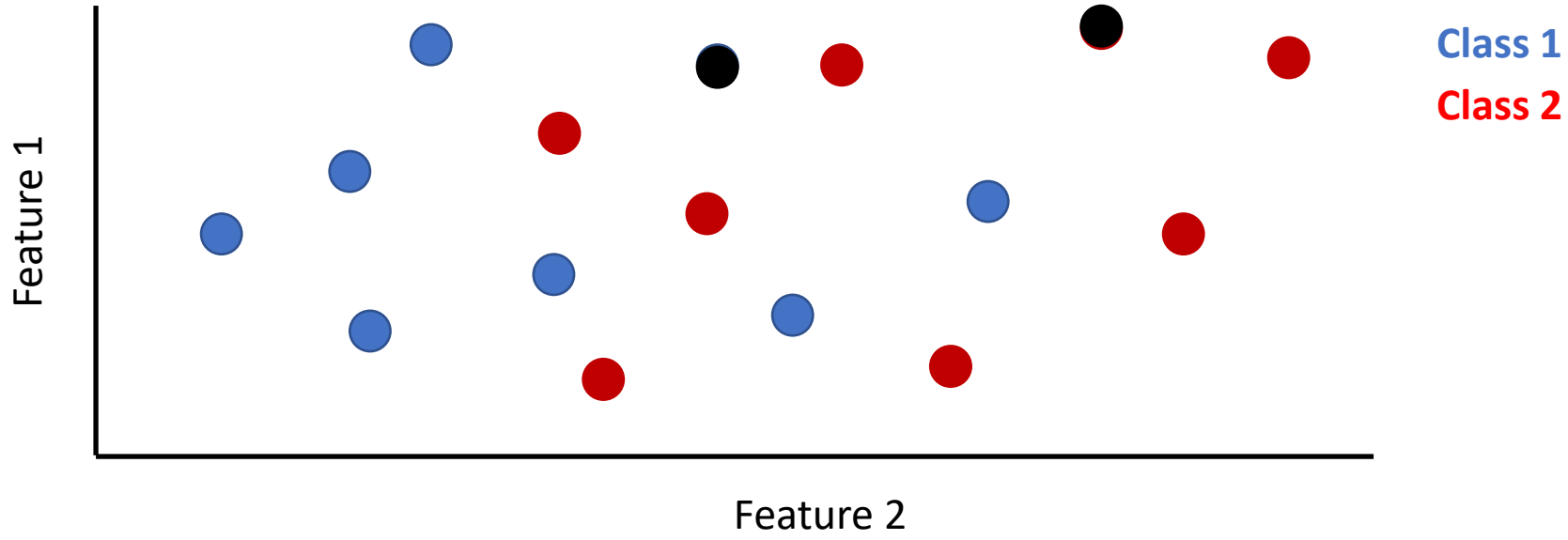
Training and test accuracy

Q: What would happen if we tested the classifier using the training data with k = 1?

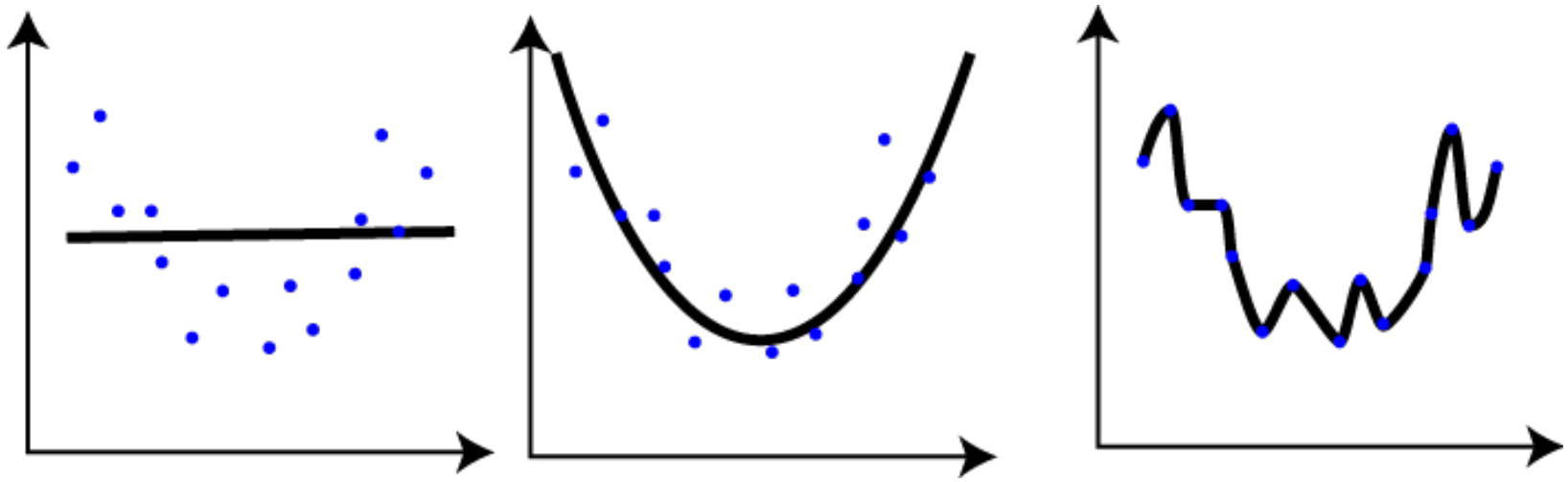
A: We would have 100% accuracy

Q: Would this indicate that the classifier is good?

- A: No!



Review: overfitting



Review: overfitting

If our classifier has over-fit to the training data then:

a) We might not have a realistic estimate of how accurate its predictions will be on new data

b) There might be a better classifier that would not over-fit to the data and thus can make better predictions

What we really want to estimate is how well the classifier will make predictions on new data, which is called the **generalization (or test) error**

Overfitting song…

Review cross-validation

Training error rate (training accuracy): model predictions are made on using the same data that the model was fit with

Test error rate (test accuracy): model predictions are made on a separate set of data

k-fold cross-validation

Are there any downsides to using half the data for training and half for testing?

Since we are only using half the data for training, potentially can build a better model if we used more of our data

- Say 90% of our data for training and 10% of testing

Problem: Then our estimate of the generalization error would be worse

Solutions?

k-fold cross-validation

k-fold cross-validation

- Split the data into k parts
- Train on k-1 of these parts and test on the left out part
- Repeat this process for all k parts
- Average the prediction accuracies to get a final estimate of the generalization error