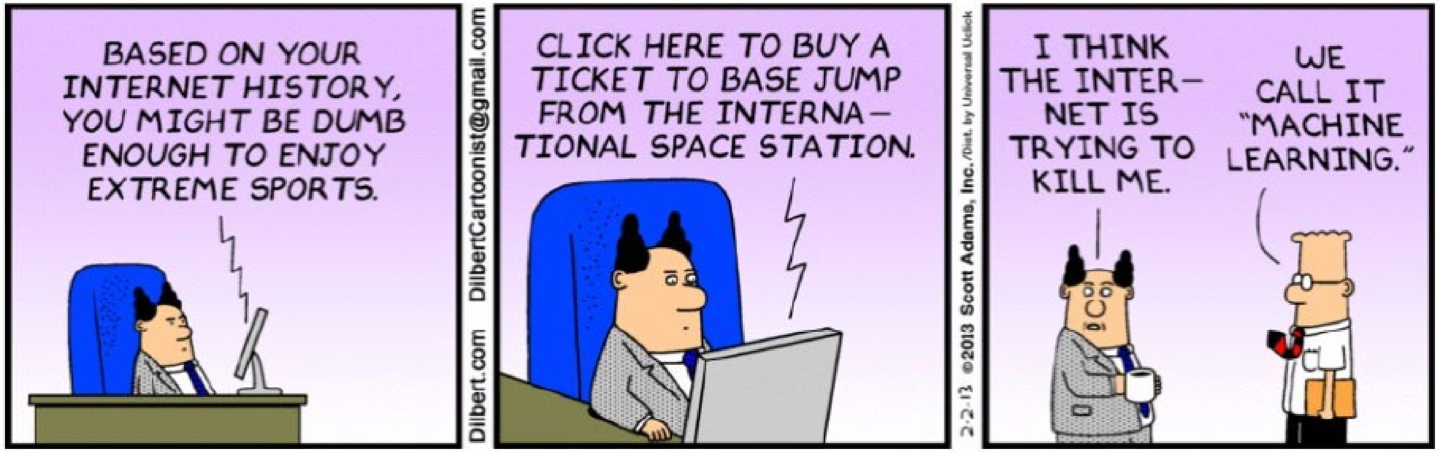
Classification

Introduction to data analysis: Lecture 10

Ori Plonsky

Spring 2023



Classification

Static/Ben-El?

Good employee/Bad employee?

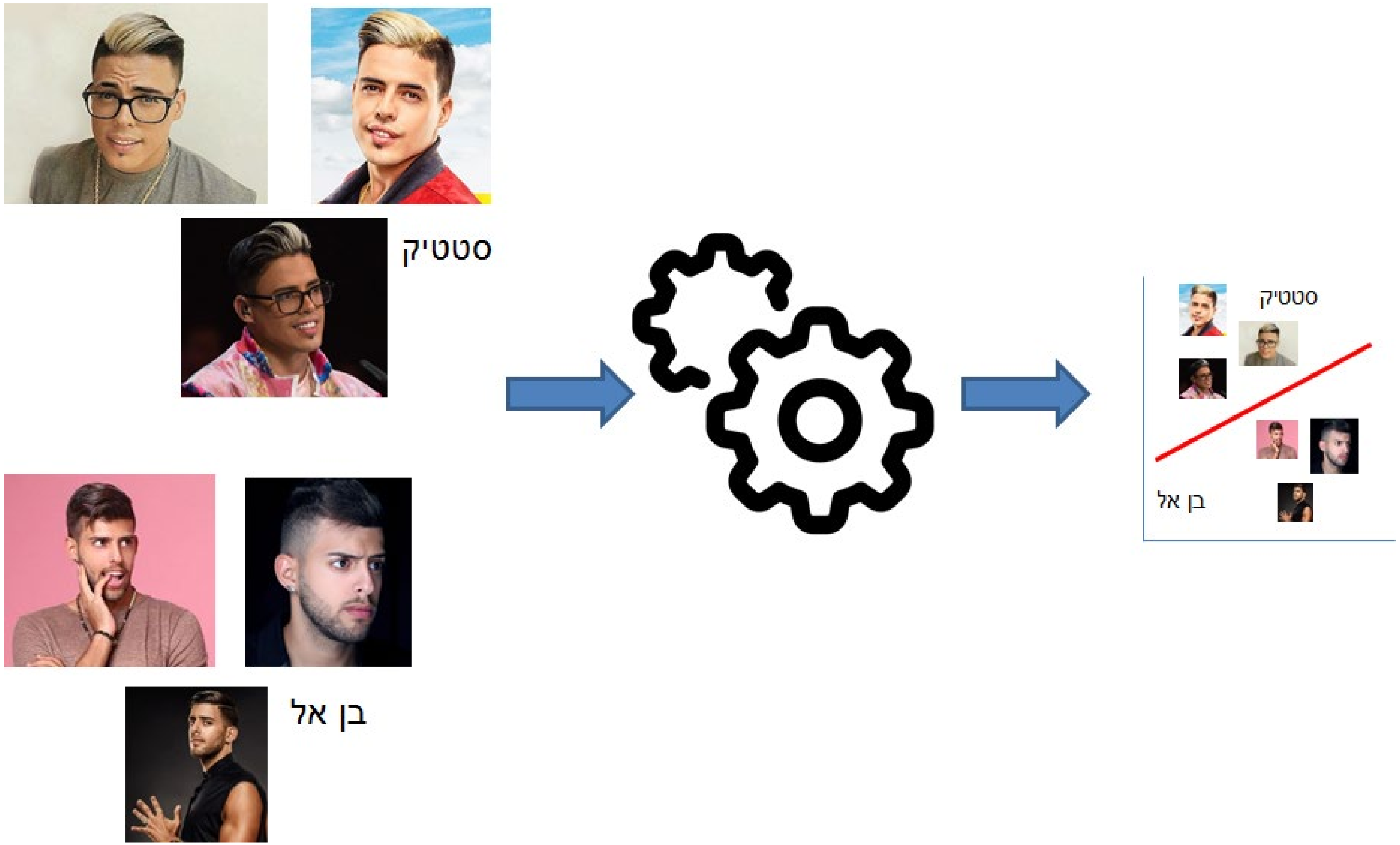
Predictions

- We want to predict an unobserved response Y
- Given a set of p observed features $X = \left(X_1, \ldots, X_p\right)$
- In supervised learning we have training data $\mathcal{T} = \left(x_i, y_i\right), i = 1, \ldots, N$
- We assume there exists a function f such that $Y = f\left(X\right) + \varepsilon$
  And try to learn f from our training data
- We approximate f with $\hat{f}$ and predict $\hat{Y} = \hat{f}\left(X_1, \ldots, X_p\right)$
  - Note that in prediction, we only really care about the value $\hat{Y}$
    - And how "close" it is to Y
  - (In inference problems, we care about the $\hat{f}$ itself.)

Classification: Supervised learning

- Input:
  a training set of N data points, each labeled with one of K different classes
- Learning:
  use the training data to learn what characterizes each class
- Evaluation:
  predict labels for a test set of data and compare the true labels (ground truth) to the ones predicted by the classifier

Supervised learning - training



Supervised learning - prediction

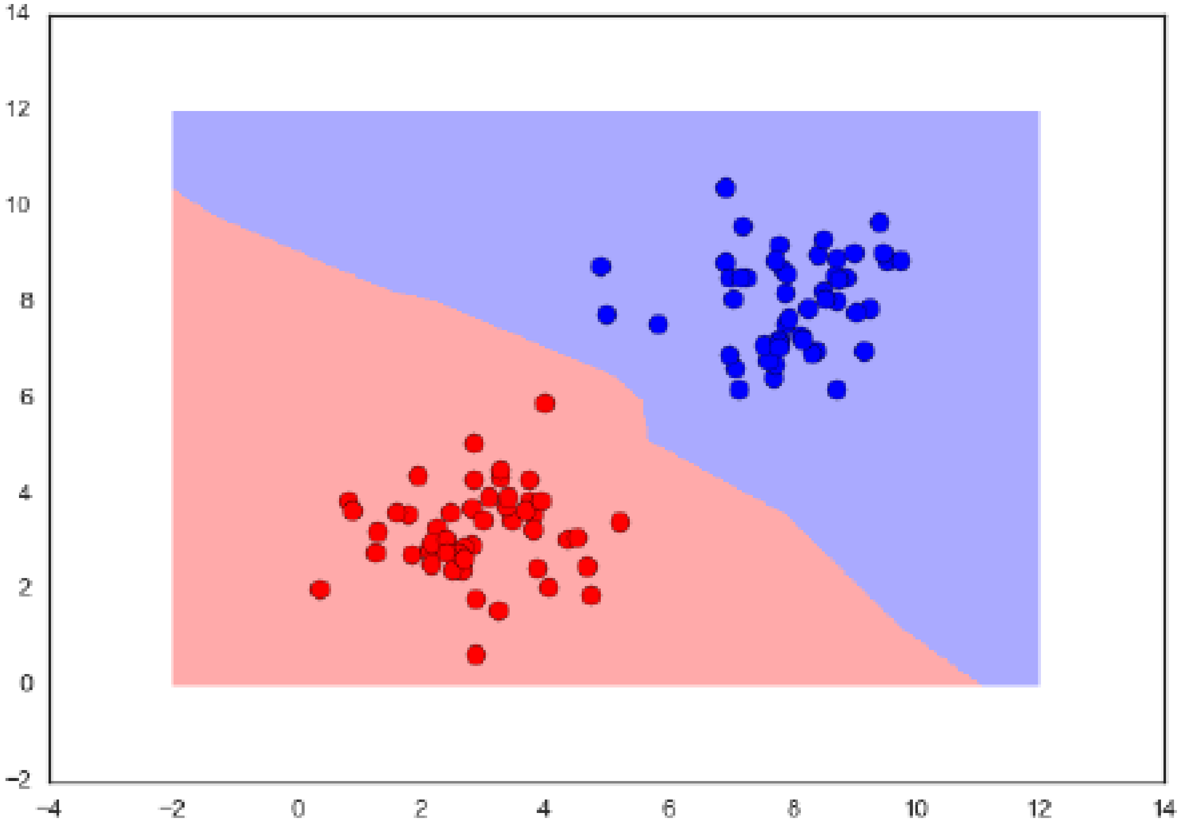
סטטיק

k-Nearest Neighbors (kNN)

- The k-Nearest Neighbor (kNN) model is an intuitive way to predict the class of a response variable
- To predict a response for a set of observed predictor values, we use the majority class of other observations most similar to it
  - Its nearest neighbors

1-NN classification

- Predict class of new data point according to the closest data point in the training set



Distance metrics

- L1 (Manhattan) distance: $d_1(I_1, I_2) = \sum_p \left|I_1^p - I_2^p\right|$

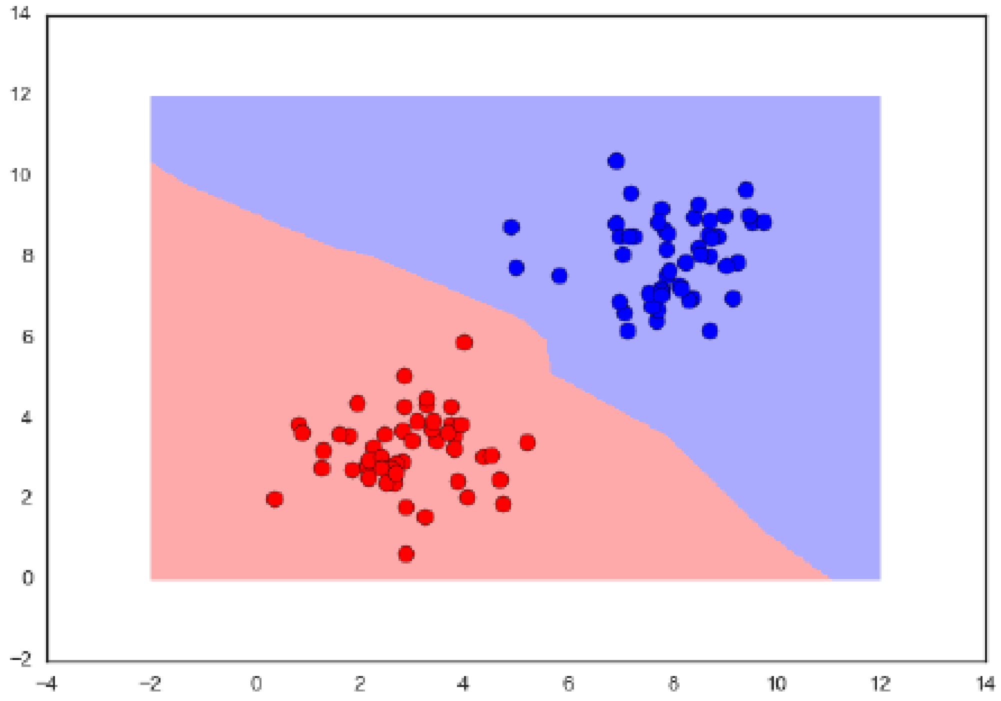
- L2 (Euclidean) distance: $d_2(I_1, I_2) = \sqrt{\sum_p \left(I_1^p - I_2^p\right)^2}$

*Default sklearn implementation is L2 distance

(notebook)

Decision boundary

- A change in input attributes might change the classifier's prediction
- Inputs that are very "close" but result in different predicted labels are on either side of a **decision boundary**
    - To visualize it, we can plot predictions for a range of possible inputs



Scale problems

- If different variables have different scales, their impact on the prediction differs
- Solution: transform the variables such that they have a common scale
- Many scaling methods
  - Standardizing/normalizing (using z-scores): $x_{normalized} = \dfrac{x - \bar{X}}{S_x}$
  - Range normalization: $x_{range-scaled} = \dfrac{x - x_{\min}}{x_{\max} - x_{\min}}$

Evaluation

- Train on training data, test on test data

- Training data: train classifier
- Test data: measure performance

- (for all algorithms, not just for kNN)

Evaluation

- We want to know how accurate our classifier is

Evaluation

- We want to know how accurate our classifier is
- Accuracy of *k*NN on train data?

Evaluation

- We want to know how accurate our classifier is
- Accuracy of *k*NN on train data?
- We want to know how accurate our classifier will be on new data

Evaluation

- We want to know how accurate our classifier is
- Accuracy of *k*NN on train data?
- We want to know how accurate our classifier will be on new data

Test data:

- We “pretend” we do not have some of the data

- If new data will come from the same distribution as the test data, the accuracy on the test data will be similar to accuracy on new data

Evaluation

- We want to know how accurate our classifier is
- Accuracy of *k*NN on train data?
- We want to know how accurate our classifier will be on <u>new</u> data

Test data:

- We "pretend" we do not have some of the data
  - We therefore cannot use it <u>at any point</u> during training!

- If new data will come from the same distribution as the test data, the accuracy on the test data will be similar to accuracy on new data

Evaluation

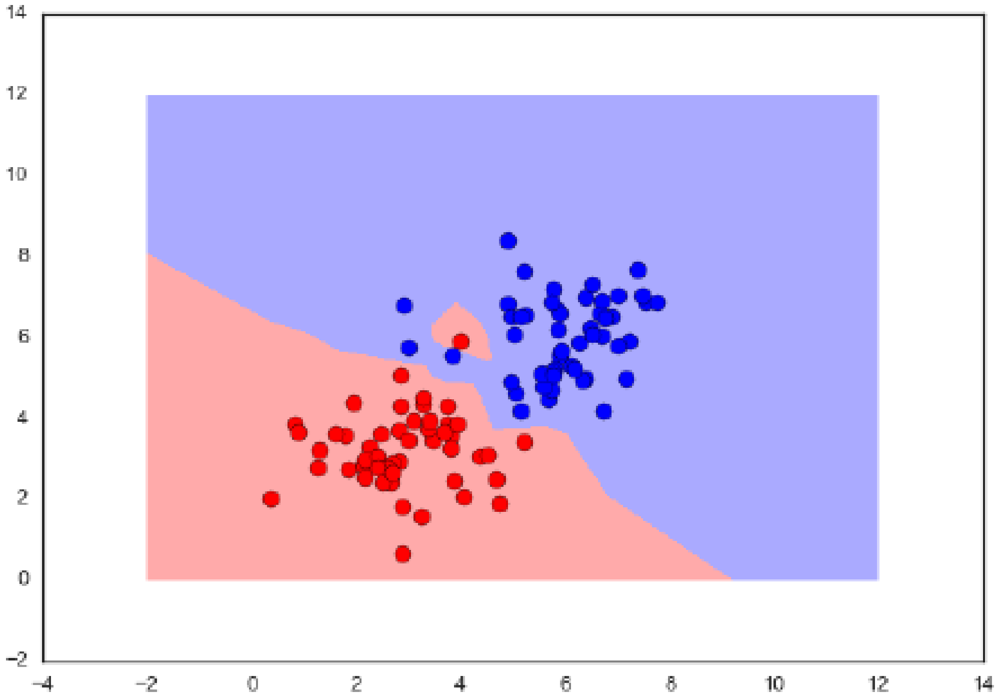
- We want to know how accurate our classifier is
- Accuracy of *k*NN on train data?
- We want to know how accurate our classifier will be on new data

Test data:

- We "pretend" we do not have some of the data
  - We therefore cannot use it at any point during training!
  - **Beware data leakage**: never use for training any information you would not have had you did not actually have access to the test data (e.g., scale using only train data values)
- If new data will come from the same distribution as the test data, the accuracy on the test data will be similar to accuracy on new data

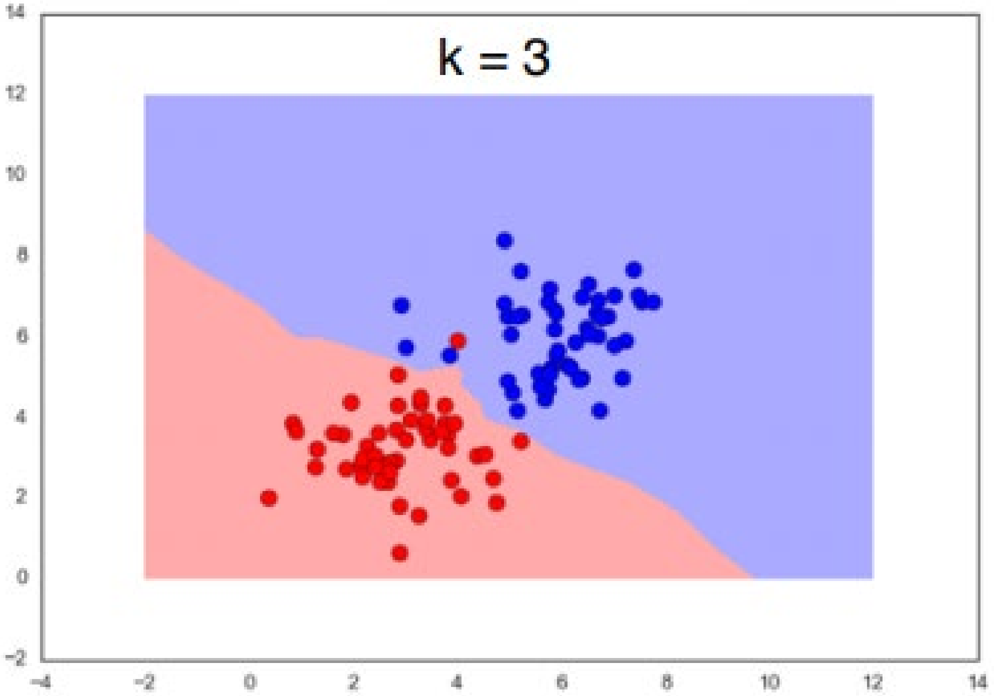
(notebook)

1-NN: Problem?

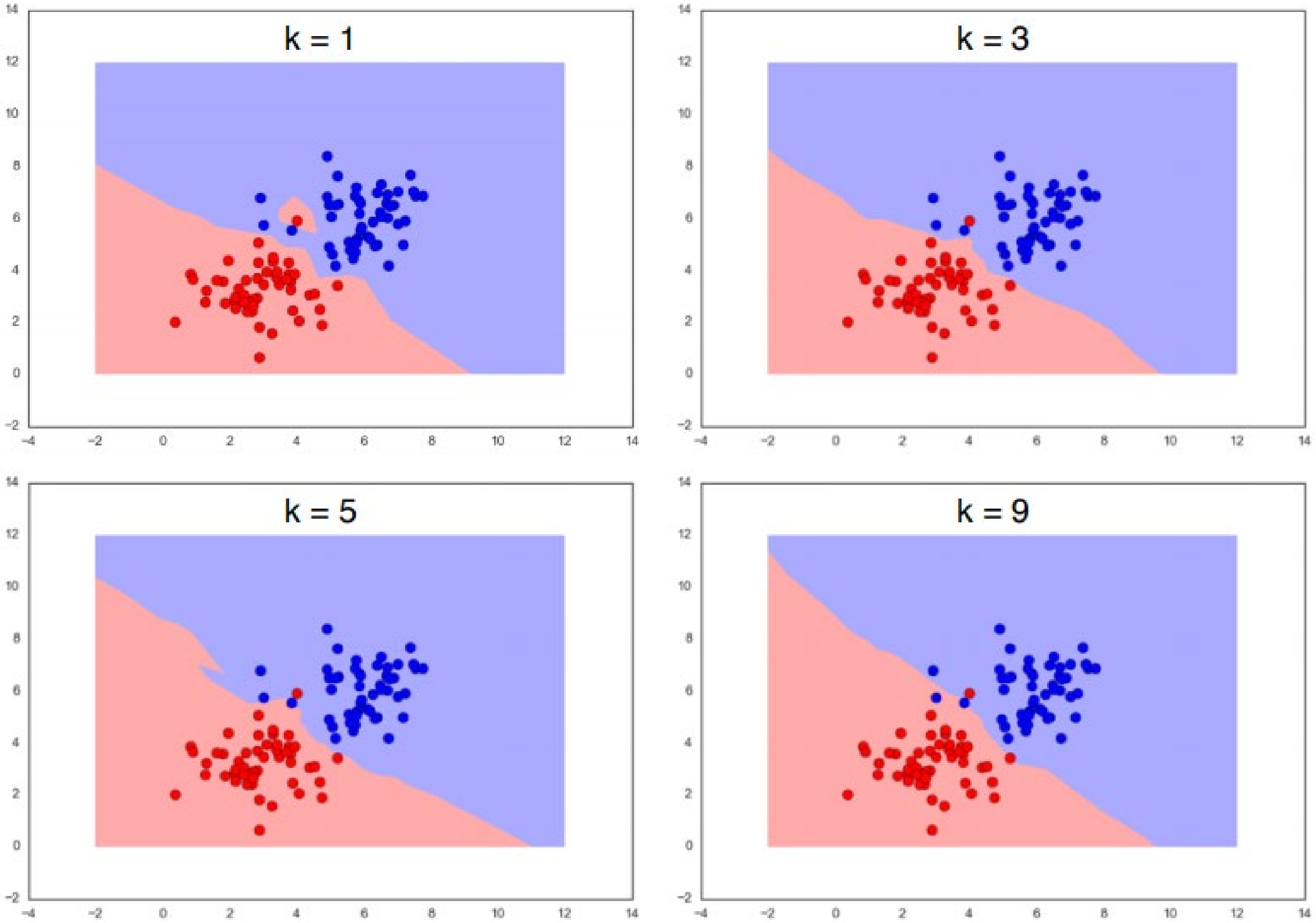


kNN classification

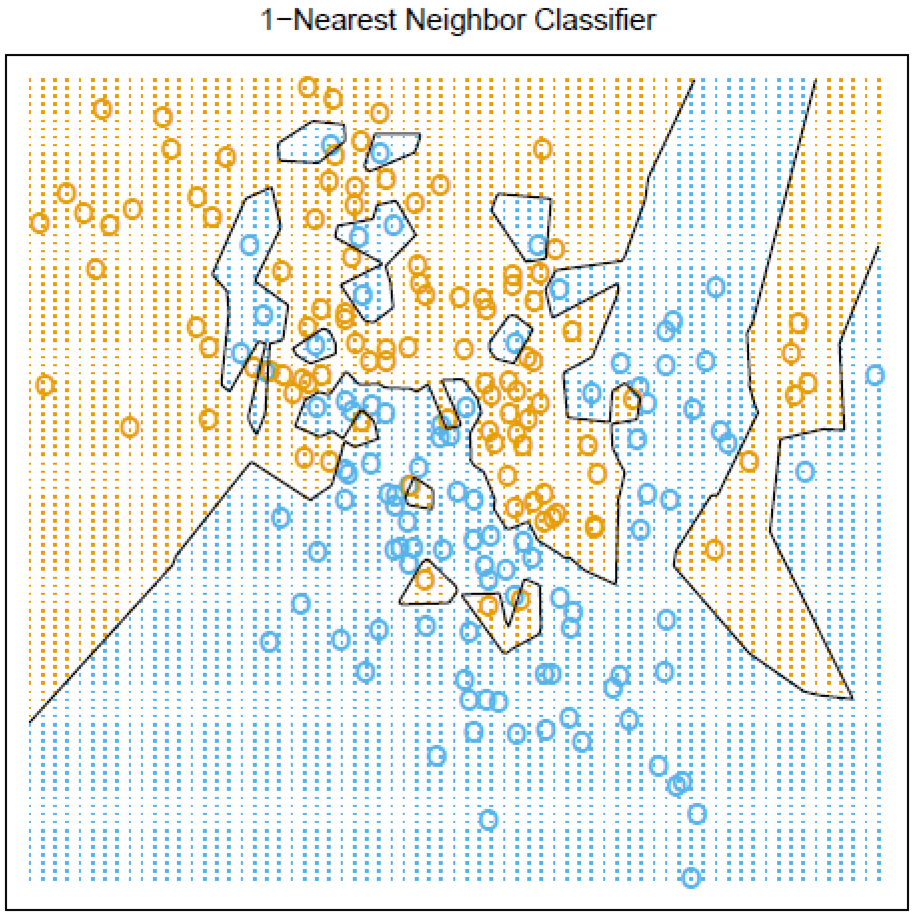
- Predict class of new data point by majority vote of *k* nearest neighbors in training set

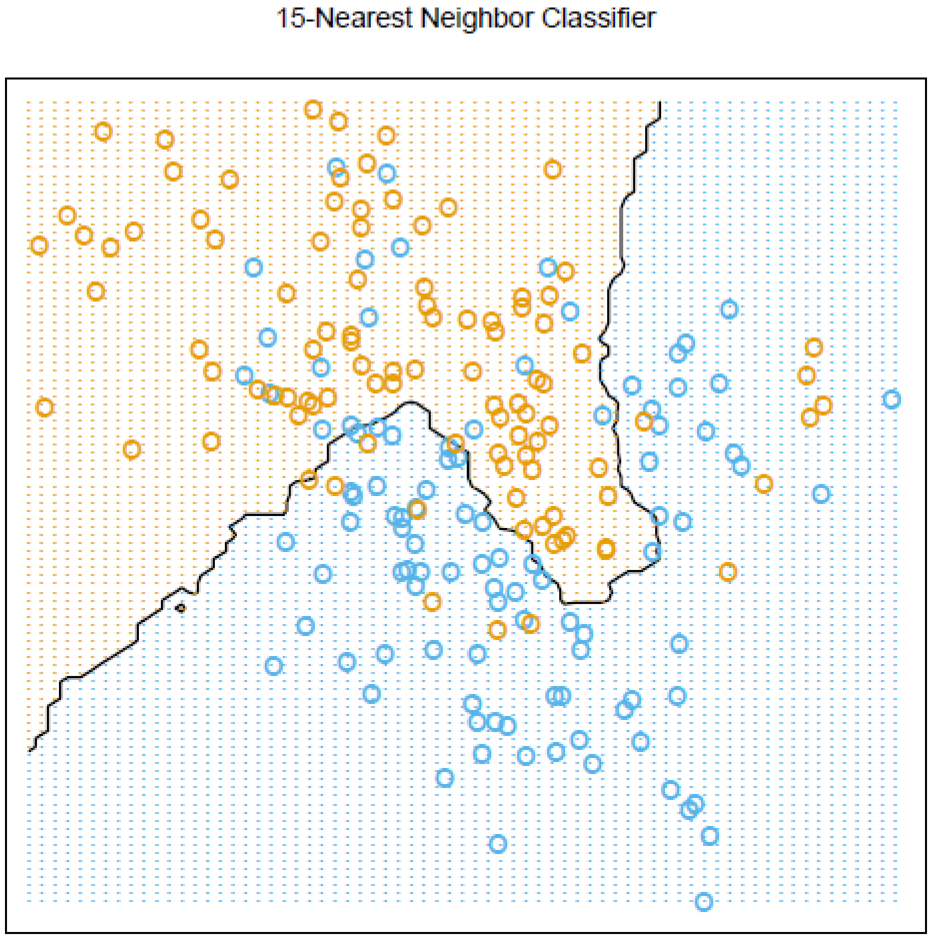


Impact of *k*



Impact of *k*





(notebook)

Source: Hastie et al., 2009. The elements of statistical learning 2nd ed.