IN2110: Språkteknologiske metoder

Klassifikasjon

Erik Velldal

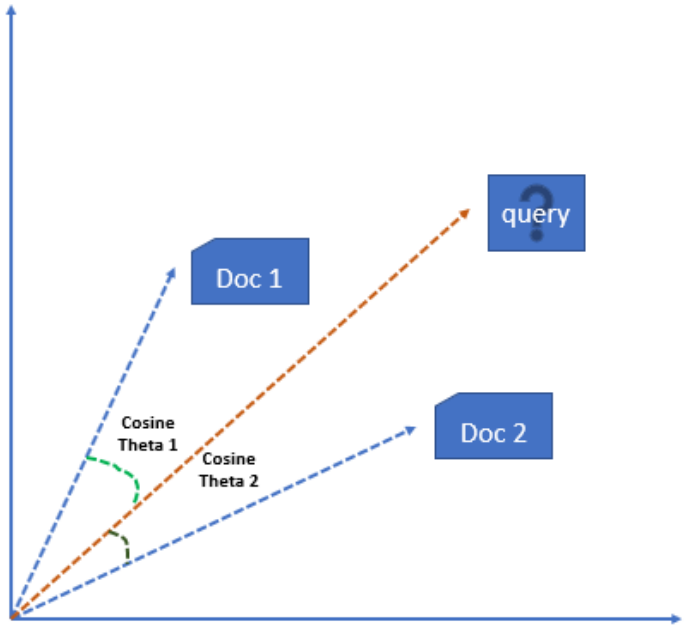
Språkteknologigruppen (LTG)

04. februar, 2025



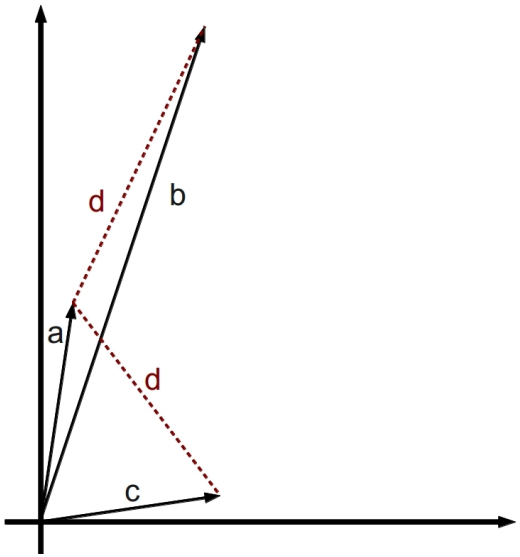
Recap: Vector space models

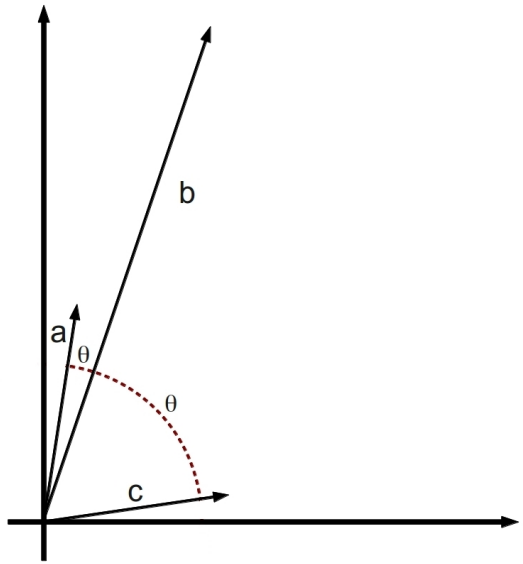
- General approach to representing data in a geometrical model.
- Example use case: vector spaces for IR.
- Documents represented as points/vectors in a feature space, $\boldsymbol{x} = \langle x_1, \ldots, x_n \rangle \in \mathbb{R}^n$.

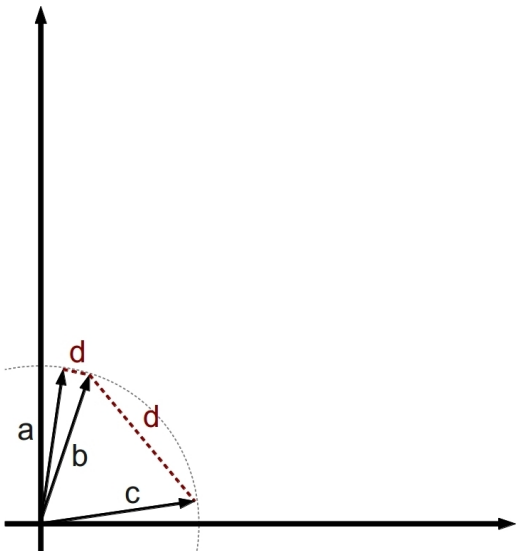


- Bag-of-words:
  - Documents represented by their unordered collection of word types.
  - Each dimension in the space corresponds to a word in the vocabulary.
- Similarity modeled by distance of documents (and queries) in the space.

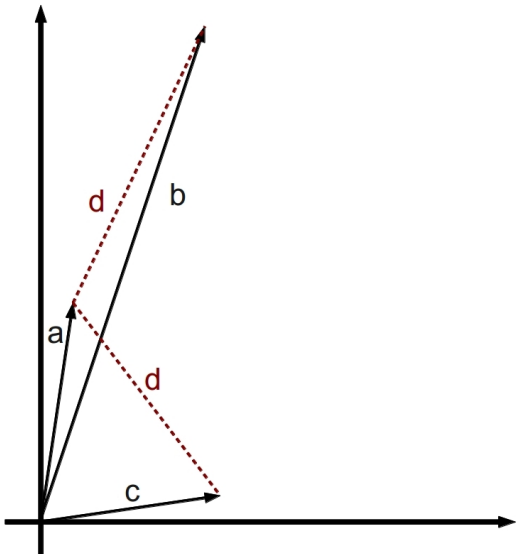
Recap: measuring similarity

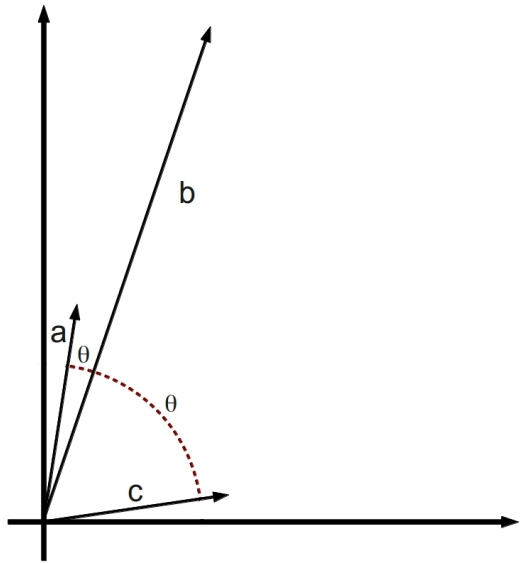


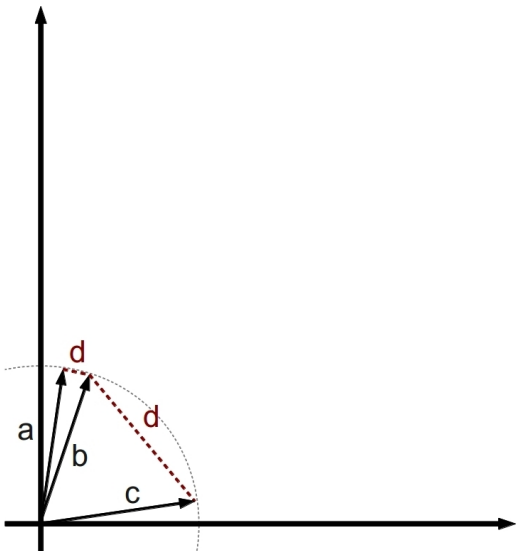




Recap: measuring similarity







- Euclidean distance between points.
- Cosine similarity of vector angles.
- Raw counts often weighted by TF-IDF.
- Can reduce length bias by normalization.

Next: Two categorization tasks in machine learning



Classification

- Supervised learning, requiring labeled training data
- Train a classifier to automatically assign *new* instances to *predefined* classes, given a set of training examples
- We look at two examples this week
  - Rocchio classification
  - k-Nearest Neighbors classification (kNN)
  - And return to cover others in later lectures, e.g. logistic regression
- We cover evaluation next week

Next: Two categorization tasks in machine learning



Classification

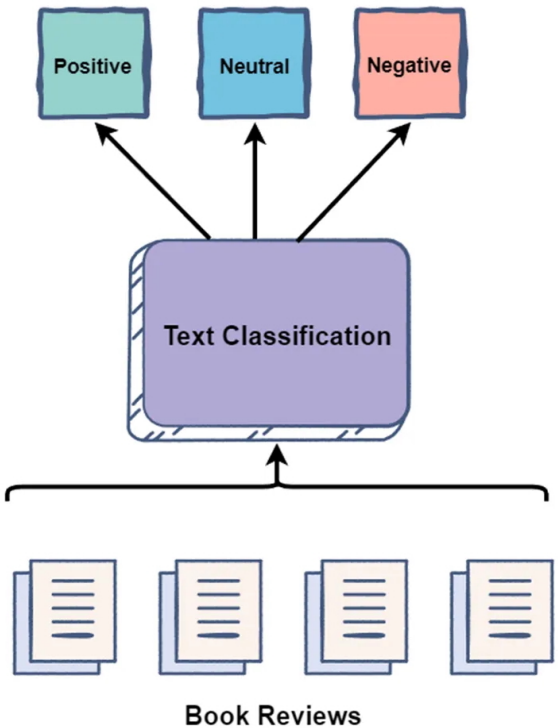
- Supervised learning, requiring labeled training data
- Train a classifier to automatically assign *new* instances to *predefined* classes, given a set of training examples
- We look at two examples this week
  - Rocchio classification
  - k-Nearest Neighbors classification (kNN)
  - And return to cover others in later lectures, e.g. logistic regression
- We cover evaluation next week

Clustering

- Unsupervised learning from unlabeled data
- Automatically group similar objects together.
- No predefined classes or structure
- The topic for next week

What is a classification?

- A domain of objects/observations to classify
- Labels: a finite set of named classes
- Classifier: maps each object to a unique label



Classification in NLP

Text classification

- Topic classification
- Authorship attribution
- Spam detection
- Sentiment analysis
  - Positive/negative/neutral
- Language identification
- Abusive language detection
- Etc.

Other classification tasks in NLP

- Tagging
- Named-entity recognition
- Event detection
- Language models
  - Predict the next word in a sentence
- Etc.

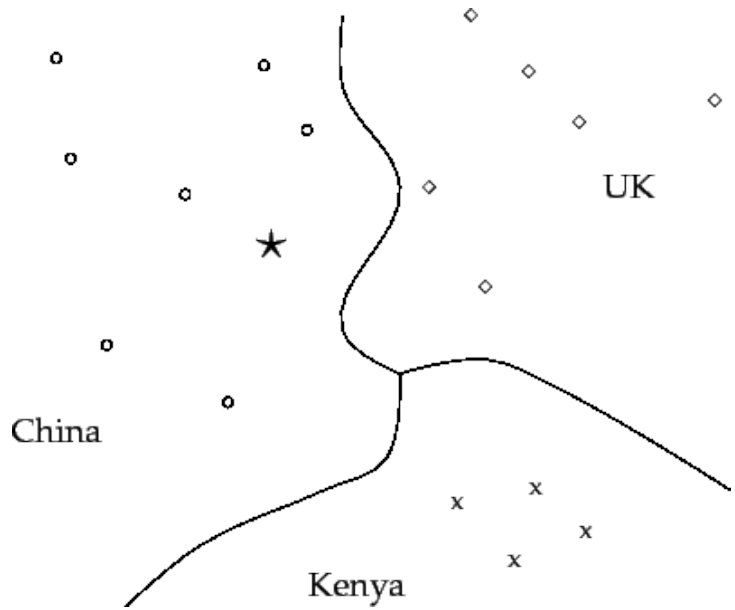
- Topic classification of documents with kNN is the topic of oblig 1a.

Three types of classification

- Binary: two classes
- Multi-class ("one-of"): three or more classes, mutually exclusive
- Some learning algorithms are by nature binary (e.g., the perceptron, logistic regression) and have to be adapted to multiclass classification
- Both Rocchio and kNN are multiclass by nature
- Can also classify an object with respect to several binary classes to achieve multi-label classification ("any-of")

Classes and classification

- In our vector space model, objects are represented as points, so classes will correspond to collections of points; regions.
- Vector space classification is based on the contiguity hypothesis:



- Objects in the same class form a contiguous region, and regions of different classes do not overlap.
- Classification amounts to computing the boundaries in the space that separate the classes; the decision boundaries.

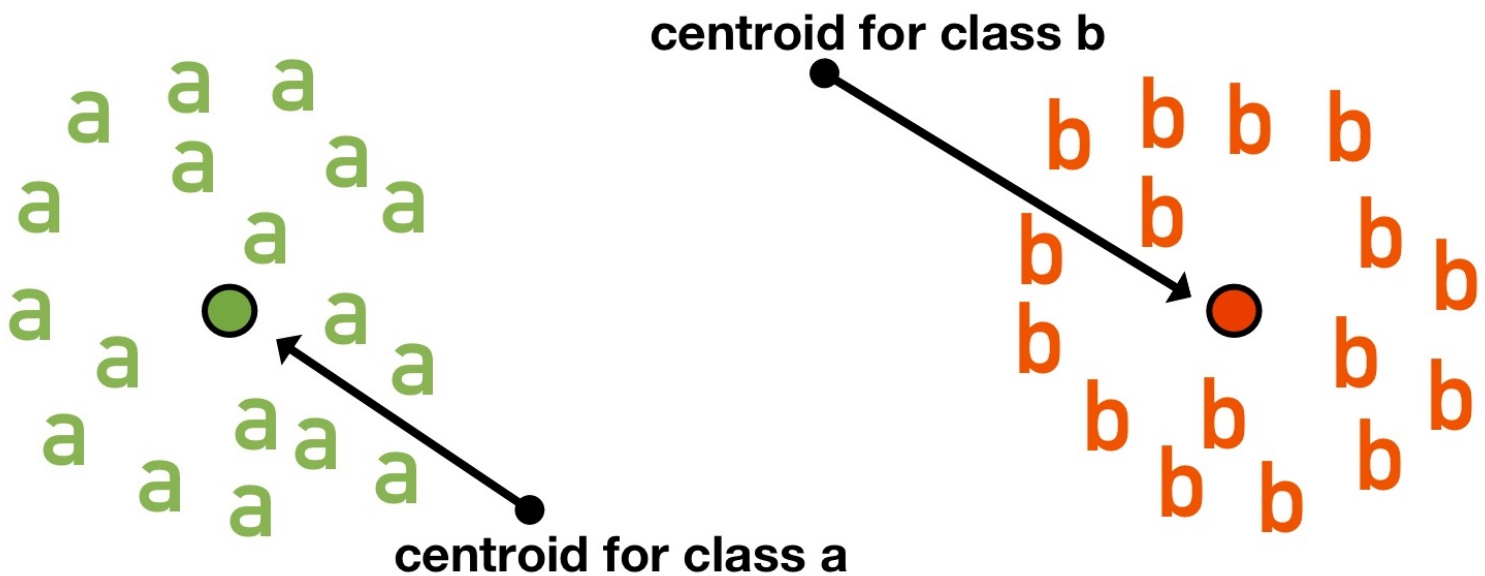
Rocchio classification (1:2)



Rocchio classification (1:2)

- Each class C_i is represented by its centroid,
- or 'center of gravity'.
- Computed as the average of the vectors $\boldsymbol{x}_j$ of its members;

$$\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x}_j \in C_i} \boldsymbol{x}_j$$



Computing a centroid; example

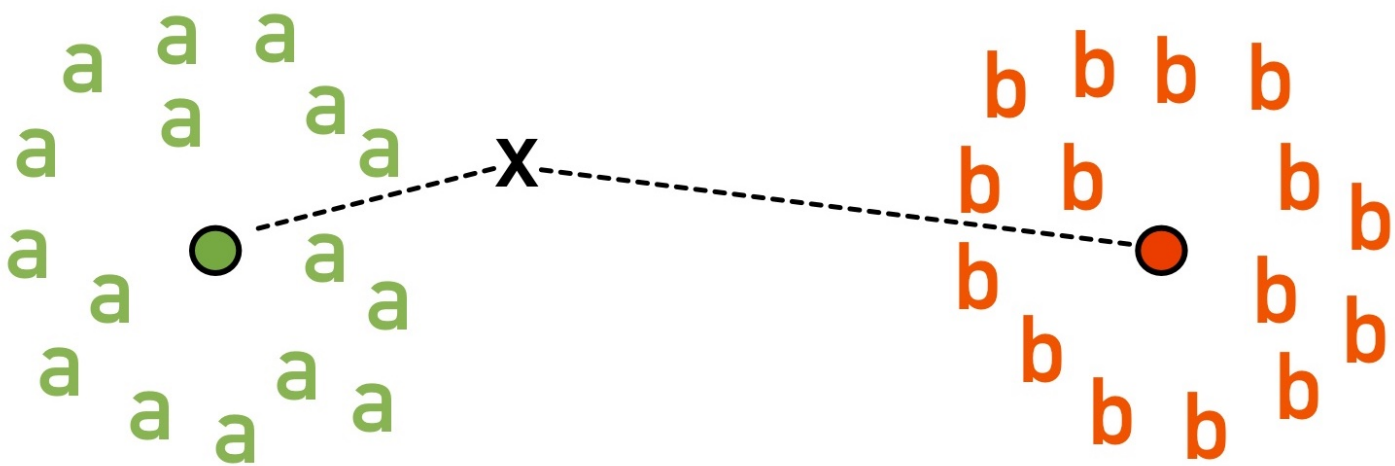
- General formula: $\boldsymbol{\mu}_i = \frac{1}{|C_i|} \sum_{\boldsymbol{x}_j \in C_i} \boldsymbol{x}_j$

- Given a class label '*music*' with 3 training documents represented by n-dimensional vectors $C_{music} = \{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}\}$:
  $\boldsymbol{x_1} = \langle x_{11}, x_{12}, \ldots, x_{1n} \rangle$
  $\boldsymbol{x_2} = \langle x_{21}, x_{22}, \ldots, x_{2n} \rangle$
  $\boldsymbol{x_3} = \langle x_{31}, x_{32}, \ldots, x_{3n} \rangle$

- The centroid would be computed as:
  $\boldsymbol{\mu}_{music} = \langle \frac{x_{11}+x_{21}+x_{31}}{3}, \frac{x_{12}+x_{22}+x_{32}}{3}, \ldots, \frac{x_{1n}+x_{2n}+x_{3n}}{3} \rangle$

Rocchio classification (2:2)

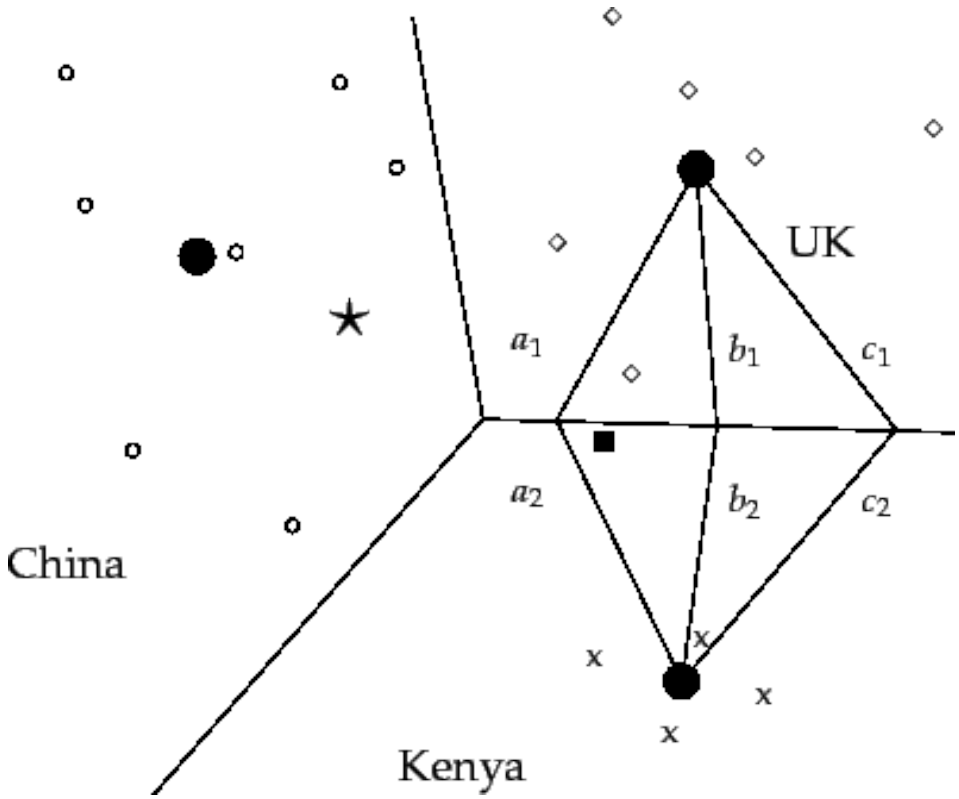
- The Rocchio decision rule:
- To classify a new object o_j (represented by a feature vector $\boldsymbol{x_j}$);
  - determine which centroid $\boldsymbol{\mu}_i$ that $\boldsymbol{x_j}$ is closest to,
  - and assign it the corresponding class label i.



- AKA nearest centroid classifier or nearest prototype classifier.

The decision boundary of the Rocchio classifier

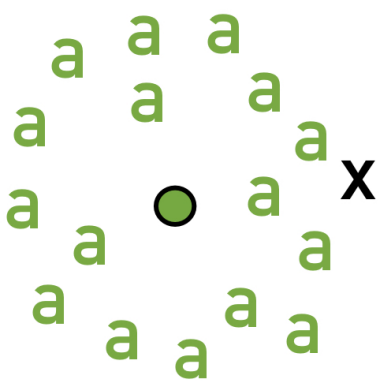
- Defines the boundary between two classes by the set of points equidistant from the centroids.
- In two dimensions, this set of points corresponds to a *line*.
- Correspods to a plane in 3D or a hyperplane in higher dimensions.
- Boundaries *implied* by the decision rule; not explicitly computed.

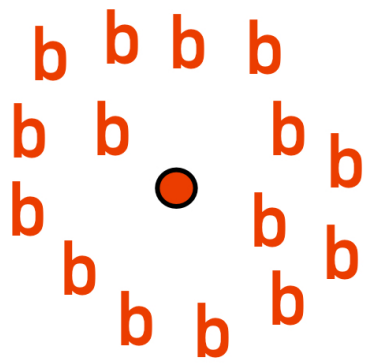


Problems with the Rocchio classifier

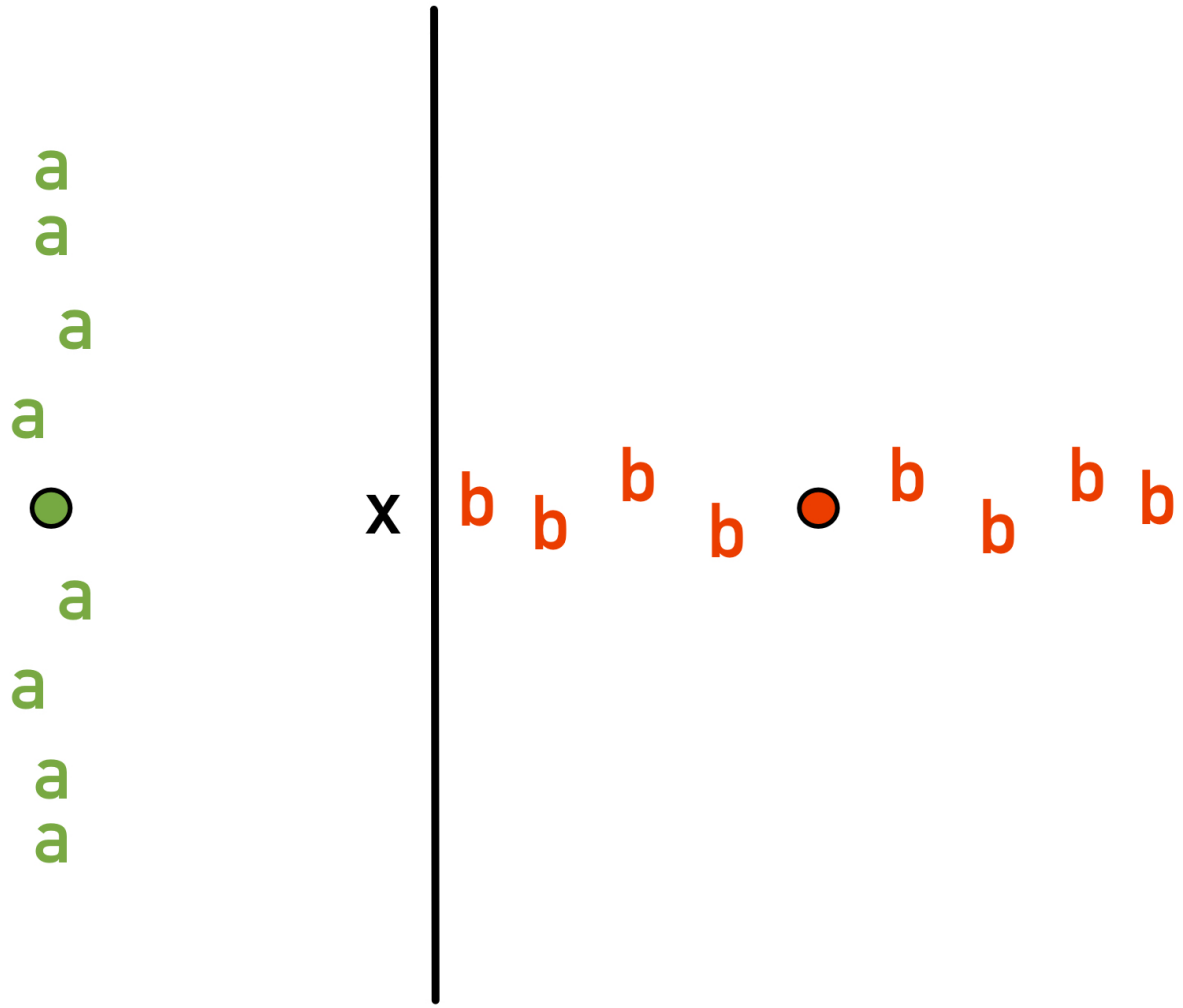
- The classification decision ignores the distribution of members locally within a class, only based on the centroid distance.
- Implicitly assumes that classes are spheres with similar radii.
- Does not work well for classes than cannot be accurately represented by a single prototype or center (e.g. disconnected or elongated regions).
- Because the Rocchio classifier defines a linear decision boundary, it is only suitable for problems involving *linearly separable* classes.

Ideal

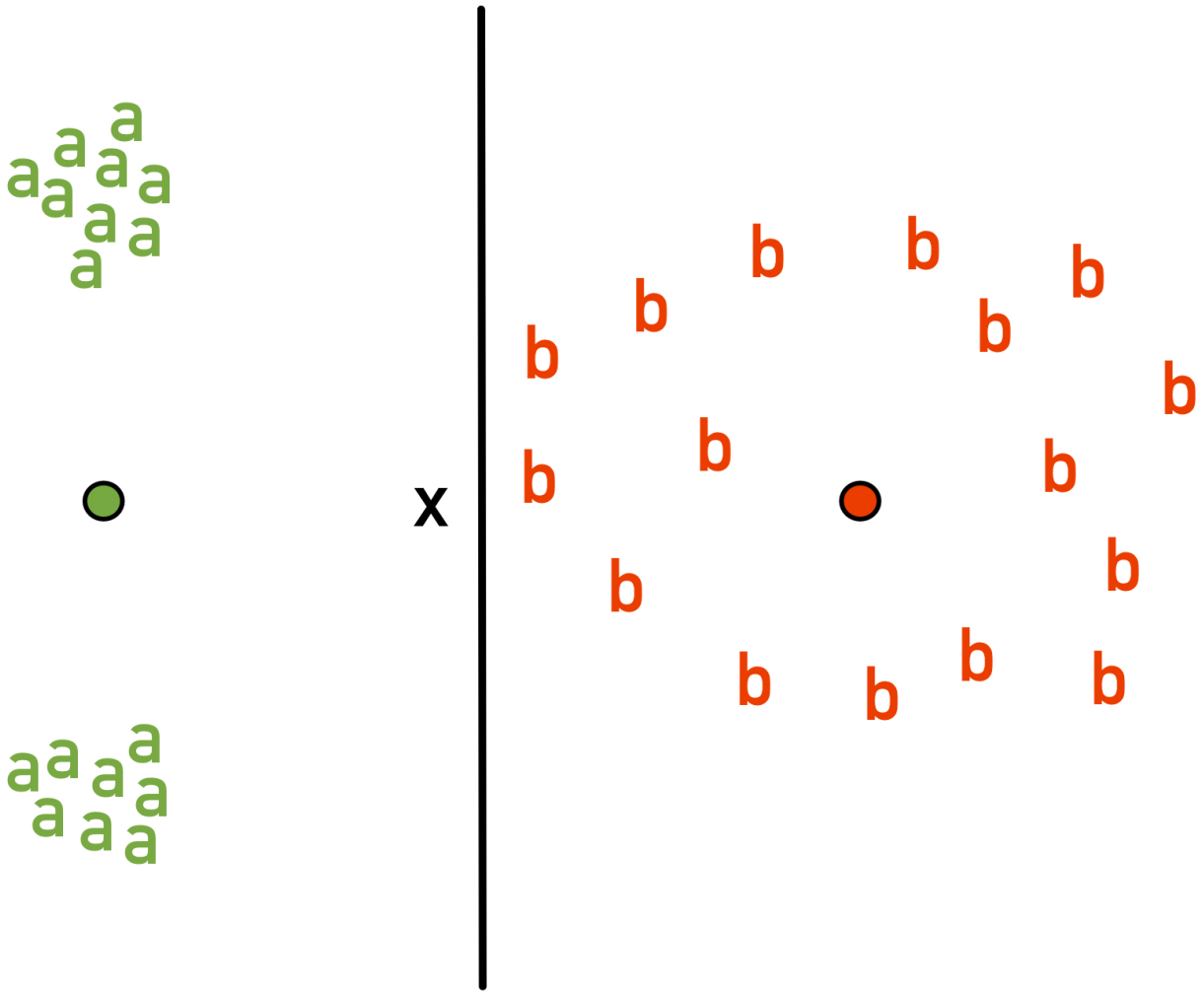




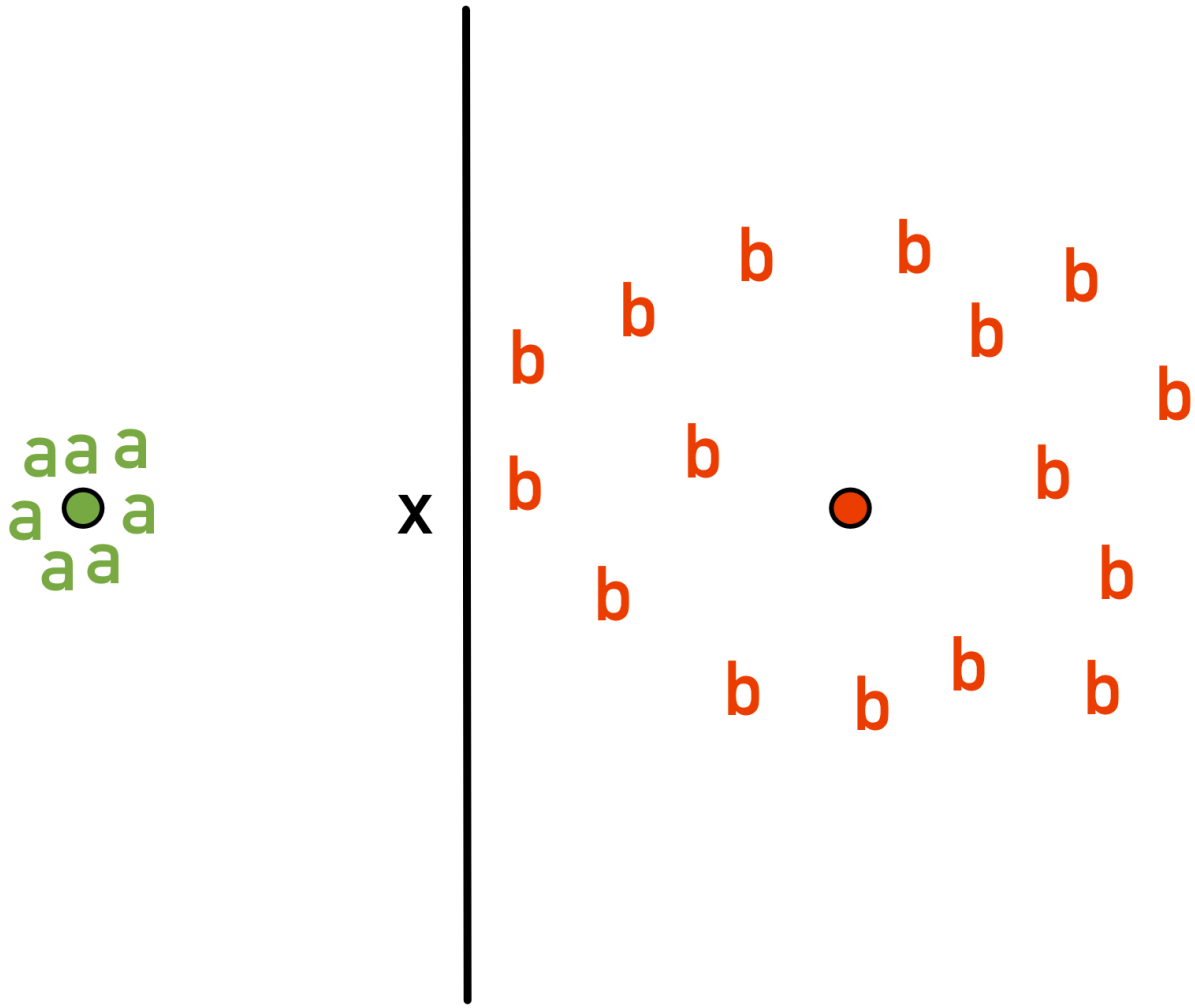
Problematic: Elongated regions



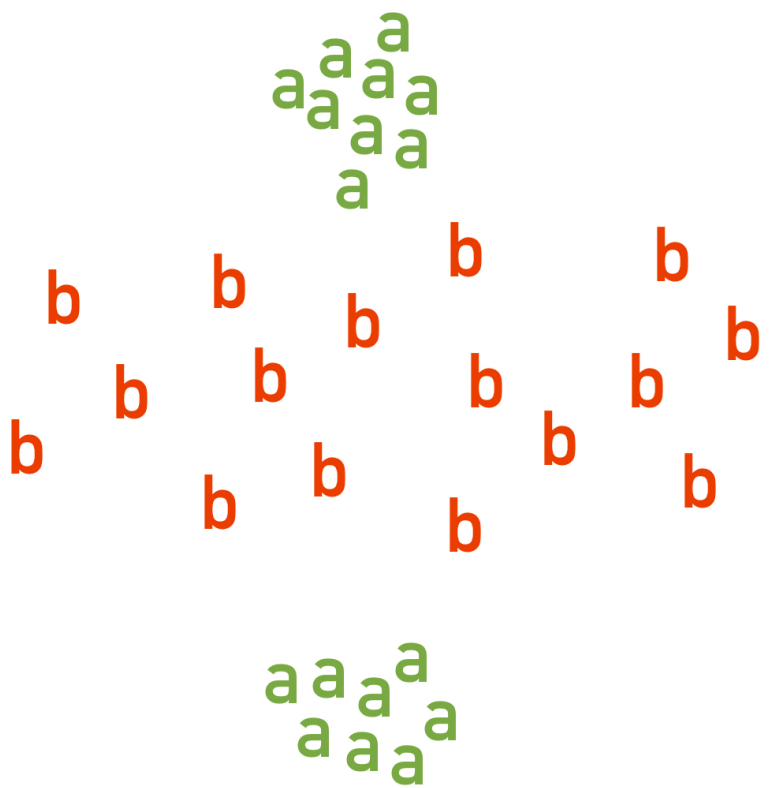
Problematic: Non-contiguous regions



Problematic: Different sizes



Problematic: Nonlinear boundary

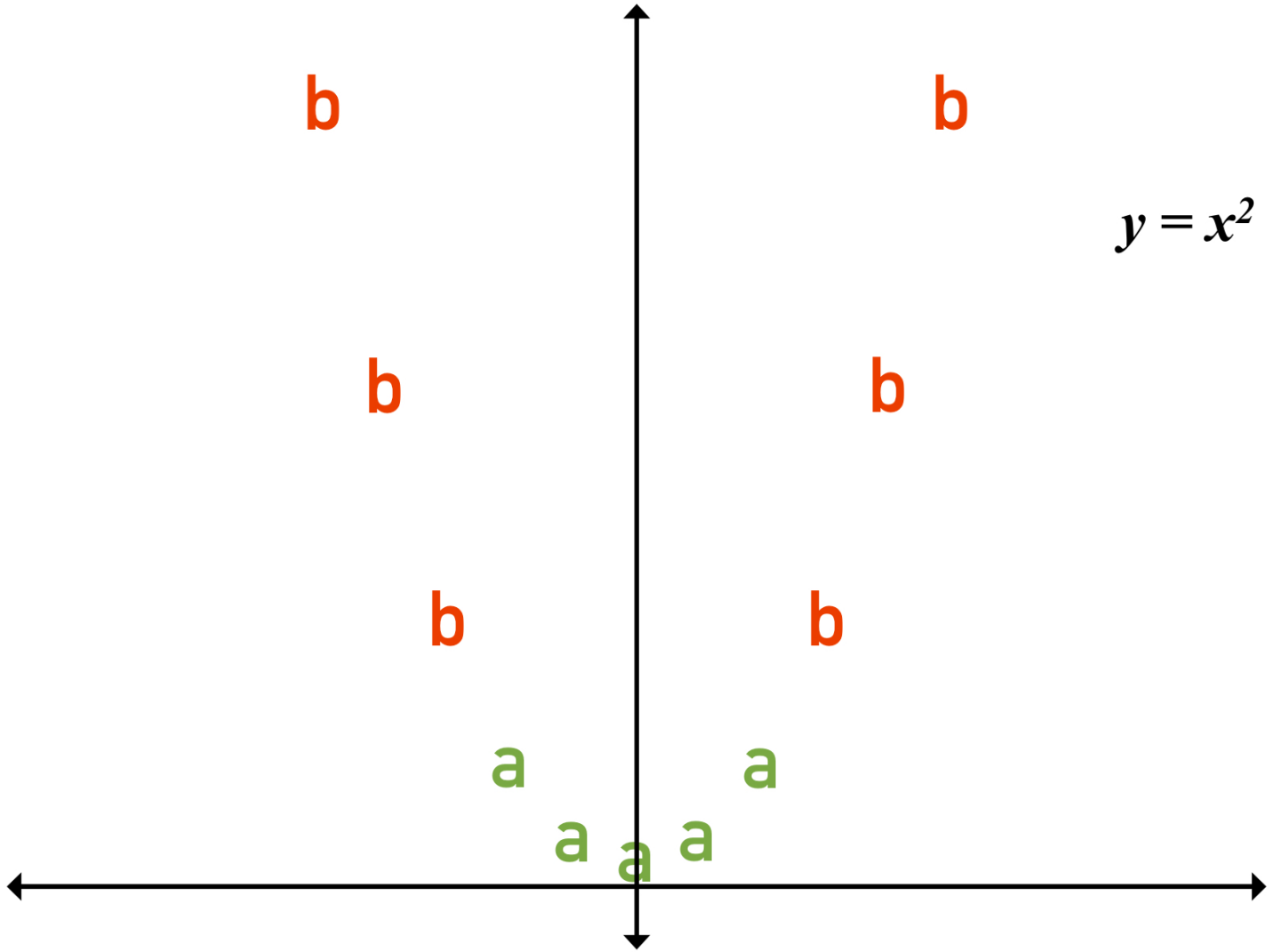


A side-note on non-linearity



- Before we turn to talk about non-linear classifiers, note that:
  Classes that are not linearly separable in a given feature space. . .

A side-note on non-linearity



- . . . may become linearly separable when the features are mapped to a higher-dimensional space (this is the basis for so-called kernel methods).

kNN-classification

- k Nearest Neighbor classification.
- An example of a memory-based, non-linear classifier.

Decision rule

- For $k = 1$: Assign each object to the class of its closest neighbor.
- For $k > 1$: Assign each object to the majority class among its k closest neighbors.

- The parameter k must be specified in advance.

kNN-classification

- k Nearest Neighbor classification.
- An example of a memory-based, non-linear classifier.

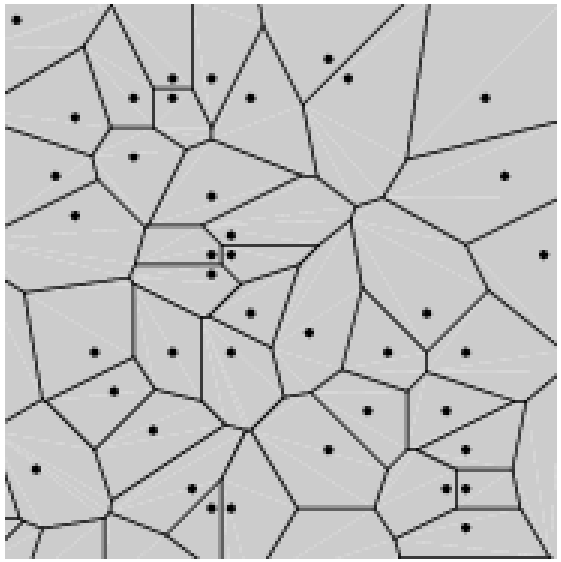
Decision rule

- For $k = 1$: Assign each object to the class of its closest neighbor.
- For $k > 1$: Assign each object to the majority class among its k closest neighbors.

- The parameter k must be specified in advance.
- Rationale: given *the contiguity hypothesis*, we expect a test object o_i to have the same label as the training objects in the local region of $\boldsymbol{x_i}$.
- Unlike Rocchio, the kNN decision boundary is determined locally.
  - The decision boundary defined by the Voronoi tessellation.

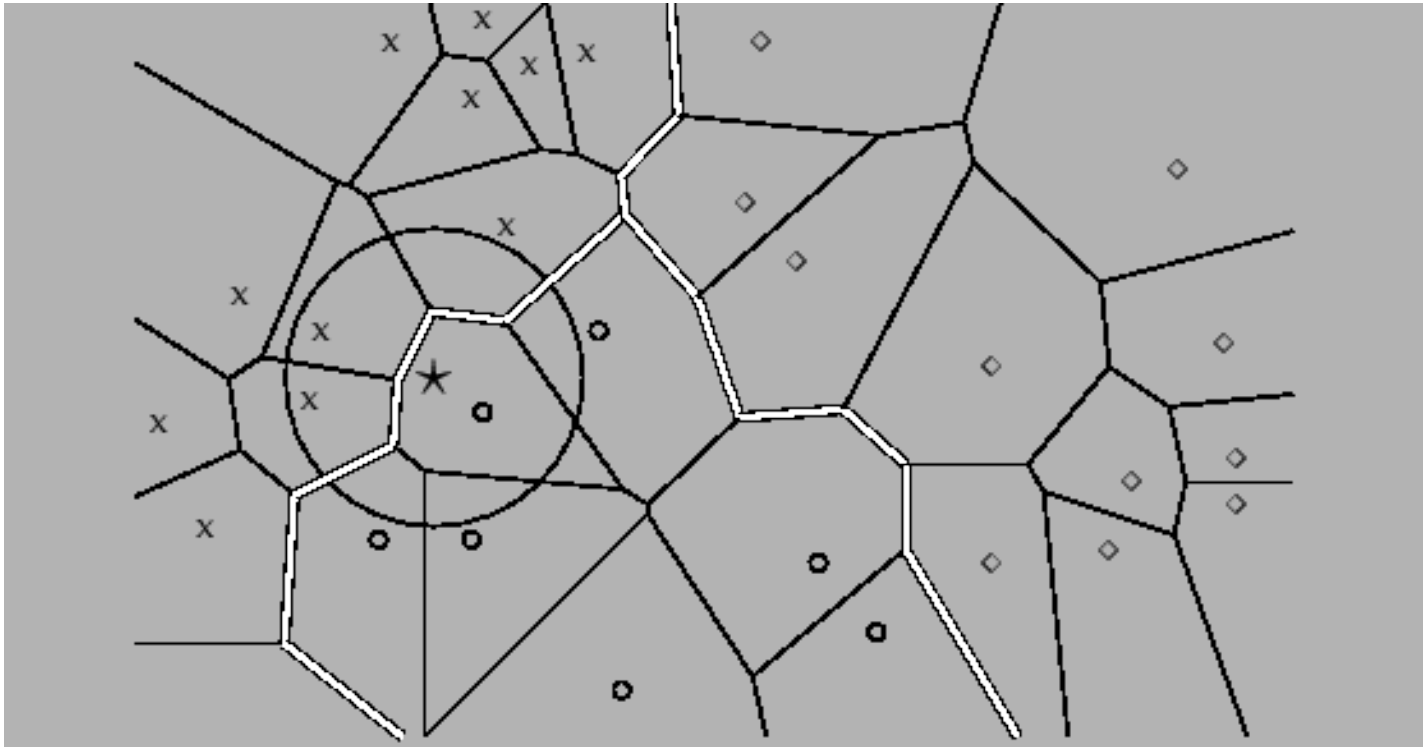
Voronoi tessellation

- Assuming $k = 1$: For a given set of objects in the space, let each object define a cell consisting of all points that are closer to that object than to other objects.
- Results in a set of convex polygons; so-called Voronoi cells.
- Decomposing a space into such cells gives us the so-called Voronoi tessellation.



- In the general case of $k \geq 1$, the Voronoi cells are given by the regions in the space for which the set of k nearest neighbors is the same.

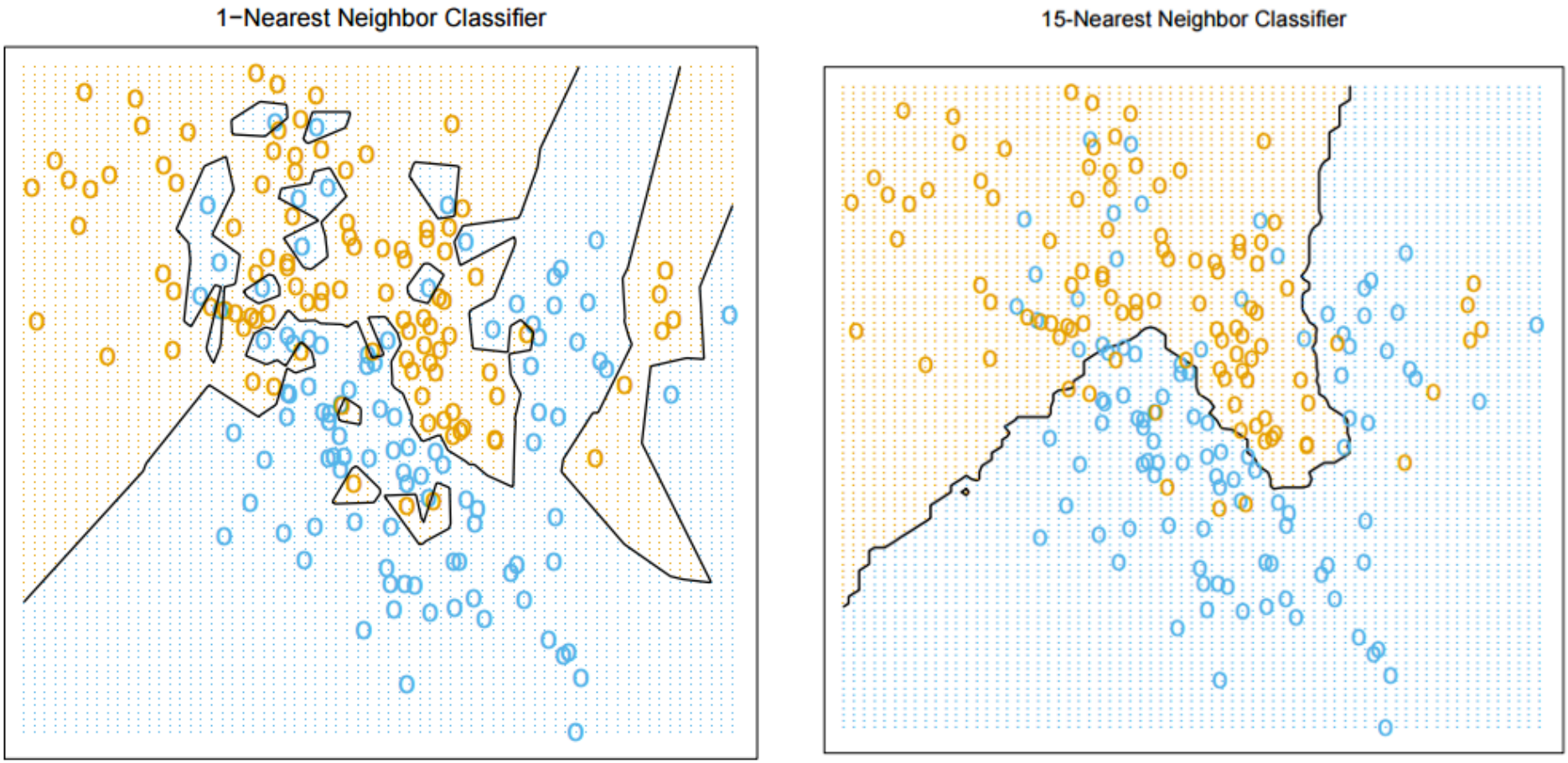
Voronoi tessellation for 1NN



► **Figure 14.3** Voronoi tessellation and decision boundaries (double lines) in 1NN classification. The three classes are: X, circle and diamond.

Decision boundary for 1NN: defined along the regions of Voronoi cells for the objects in each class. Shows the non-linearity of kNN.

The effect of K

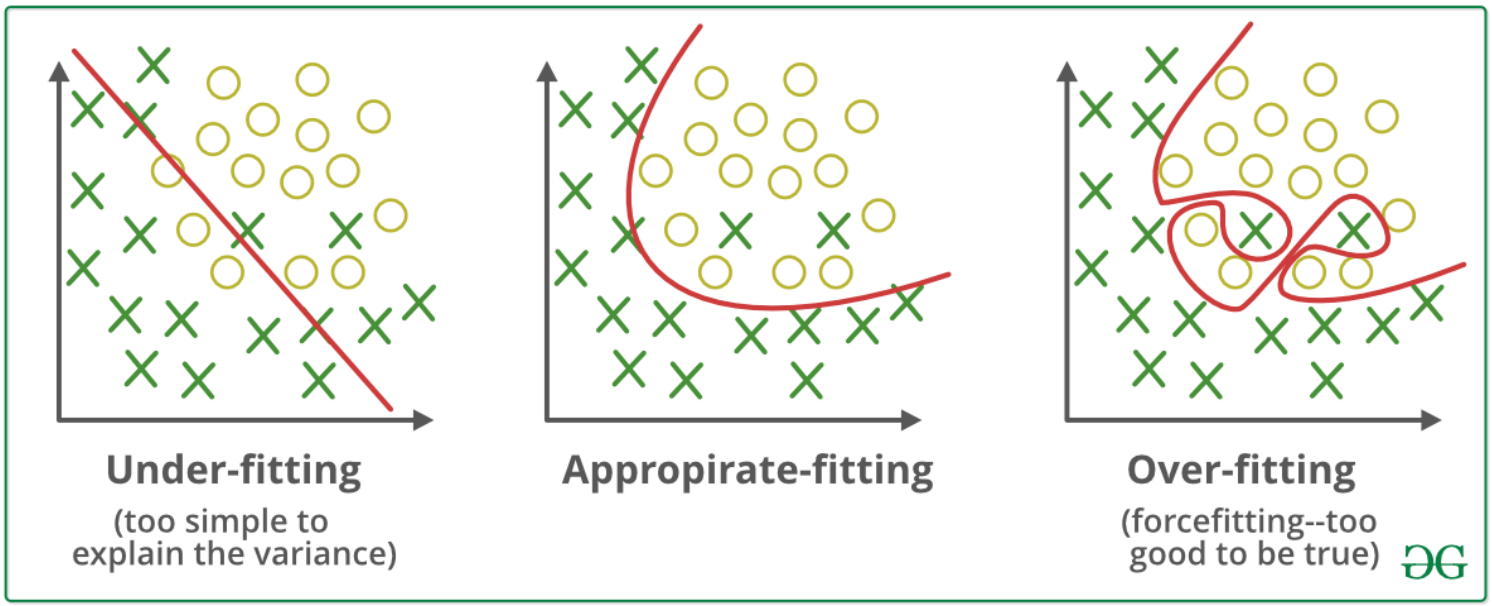


Figures from Elements of Statistical Learning

- What would happen if $K = N$?

Model fitting

- Overfitting: the model fits the training data closely but generalizes poorly, i.e. has many test errors.
- Underfitting: the model makes many errors even for the training data, i.e. it is unable to describe the data.



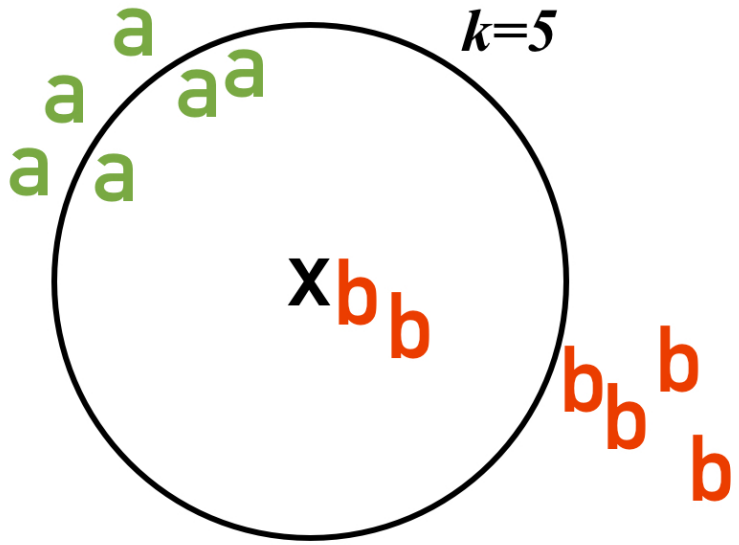
Bias–variance tradeoff

- The bias–variance tradeoff; the balancing of two sources of errors:
- Bias: the impact of assumptions in the model. Oversimplified models have high bias, leading to underfitting.
- Variance: sensitivity to changes in training data. Highly complex models may have high variance leading to overfitting.
- Example for kNN:
  - k=1: very high variance, very low bias.
  - k=N: very low variance, very high bias.

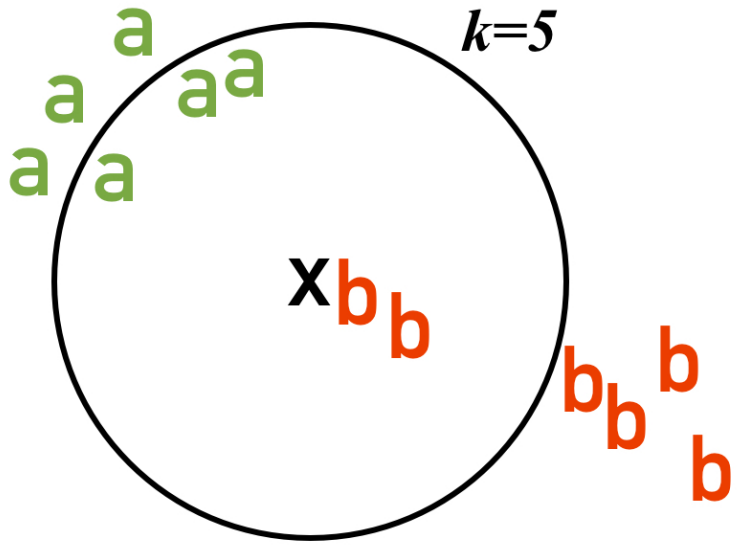
‘Softened’ kNN-classification

A probabilistic version

- The probability of membership in a class c given by the proportion of the k nearest neighbors in c.



'Softened' kNN-classification



A probabilistic version

- The probability of membership in a class c given by the proportion of the k nearest neighbors in c.

Distance weighted votes

- The score for a given class c_i can be computed as

$$\text{score}(c_i, o_j) = \sum_{\boldsymbol{x_n} \in \text{knn}(\boldsymbol{x}_j)} \text{I}(c_i, \boldsymbol{x}_n)\, \text{sim}(\boldsymbol{x_n}, \boldsymbol{x_j})$$

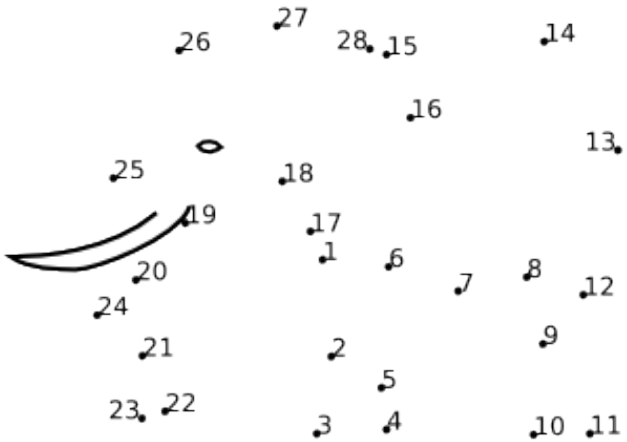
  where $\text{knn}(\boldsymbol{x}_j)$ is the set of k nearest neighbors of $\boldsymbol{x}_j$, sim is the similarity measure, and $\text{I}(c_i, \boldsymbol{x}_n)$ is 1 if $\boldsymbol{x}_n \in c_i$ and 0 otherwise.

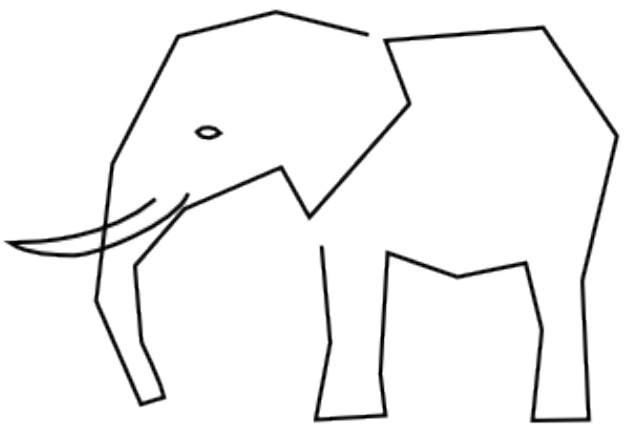
- Can give more accurate results, and also help resolve ties.

Peculiarities of kNN

- Not really any *learning* or estimation going on at all;
- simply memorizes all training examples.
- Generally with ML; the more training data the better.
- But for kNN, large training sets come with an efficiency penalty.
- Test time is linear in the size of the training set,
- but independent of the number of classes.
- A potential advantage for problems with many classes.
- Notice the similarity to the problem of ad hoc retrieval (e.g., returning relevant documents for a given query);
  - Both are instances of finding nearest neighbors.

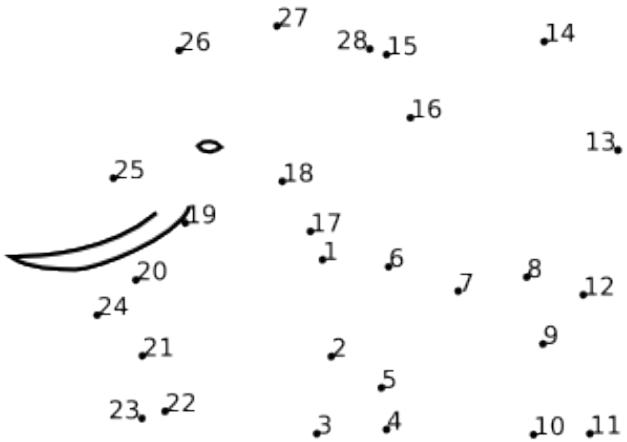
Connecting the dots...

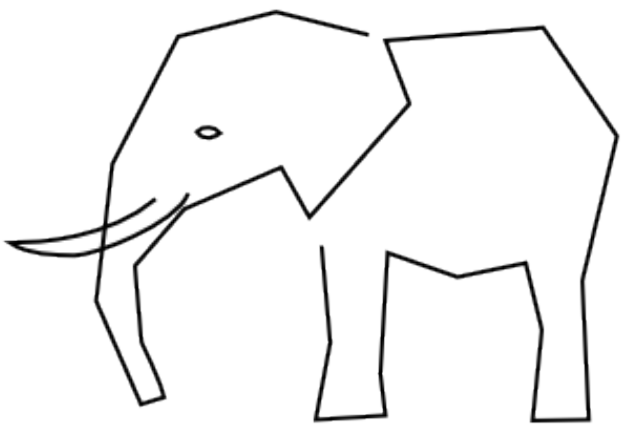




- Rocchio can be thought of as a 1 Nearest Neighbor classifier with respect to the centroids.

Connecting the dots...

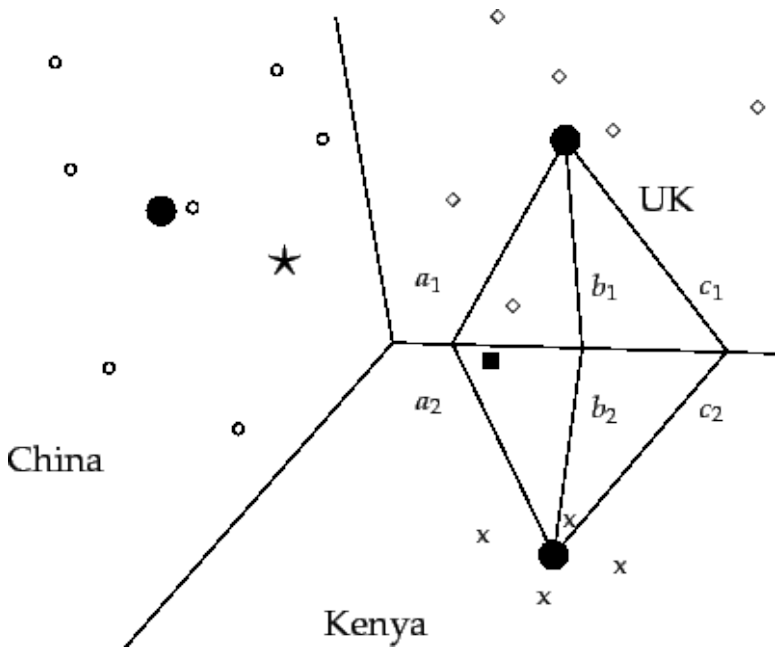


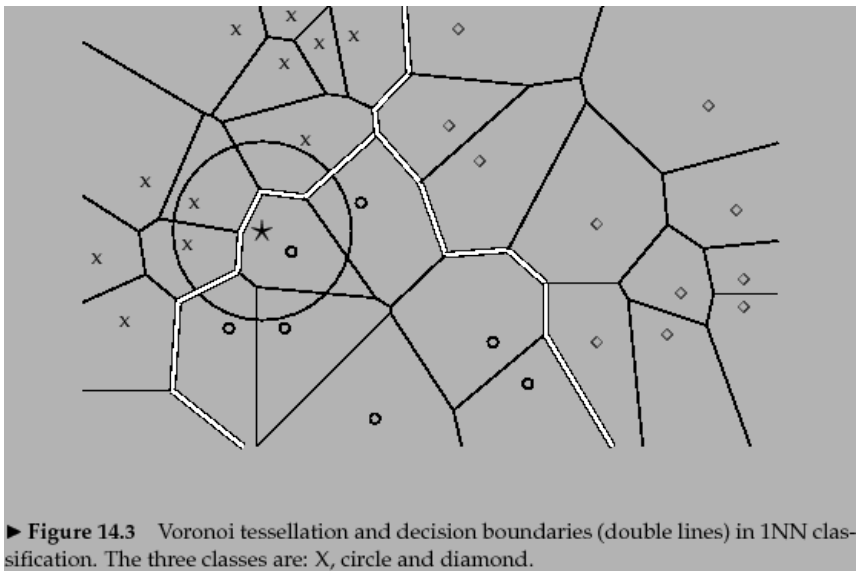


- Rocchio can be thought of as a 1 Nearest Neighbor classifier with respect to the centroids.
- How can this be? Isn't kNN non-linear and Rocchio linear?

Connecting the dots

- Recall that the kNN decision boundary is locally linear for each cell in the Voronoi diagram.
- For Rocchio, we only use one example per class, i.e. the centroid, yielding a single Voronoi cell for each entire class.





► **Figure 14.3** Voronoi tessellation and decision boundaries (double lines) in 1NN classification. The three classes are: X, circle and diamond.

A practical note on feature extraction



- Before training and applying a classifier we first have to create the feature vectors to represent our data.
- Sometimes referred to as vectorization.
- The feature types (e.g. the BoW vocabulary) needs to be defined relative to the training set (including parameters like frequency cut-offs, idf-weights, etc).
- When vectorizing test data we must use the same features as in training.
- Vectorization therefore often done in two passes: first defining the feature set based on the training data, then creating the feature vectors. (*Fit* and *transform* in scikit-learn terminology)

Next lecture

- Evaluating classifiers
- Unsupervised machine learning for class discovery: Clustering
- k-means clustering.
- Reading: *Manning, Raghavan & Schütze (2008)*, section 16, 16.1, 16.2, and 16.4 up until 16.4.1.