IN2110: Språkteknologiske metoder

Klyngeanalyse

Erik Velldal

Språkteknologigruppen (LTG)

12. februar, 2019

Today

- Evaluation of classifiers
- Unsupervised machine learning for class discovery: Clustering
- k-means clustering
- Recap

Recap – Classification

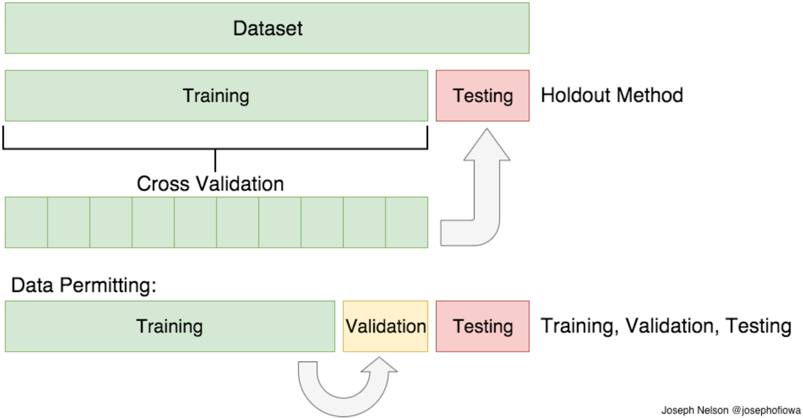
- Supervised vector space classification
- Rocchio
- kNN
- Differences?

Testing a classifier

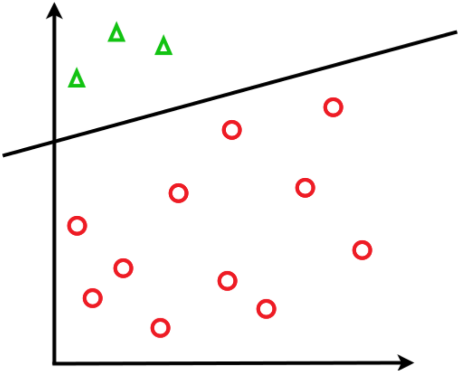
- A classification model imlicitly defines a decision boundary separating the class regions.
- To evaluate a classifier, we measure the number of correct predictions on unseen test items.
- Labeled test data is sometimes refered to as the gold standard.
- The model does not get to see the gold labels; we only use them for evaluating its predictions.

Using data splits

- While tuning our model, estimated from the training set, we repeatedly evaluate towards the development or validation data.
- Or, if we have little data, by n-fold cross-validation.
- Then we evaluate how our *final* model *generalizes* on a held-out test set.

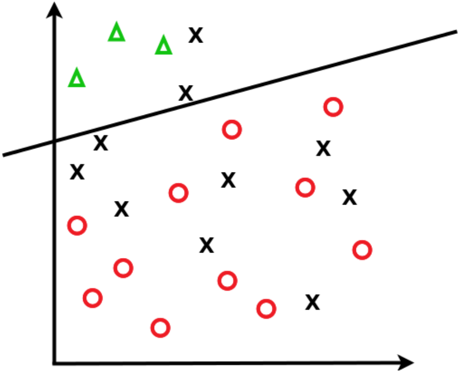


Example: Evaluating classifier decisions



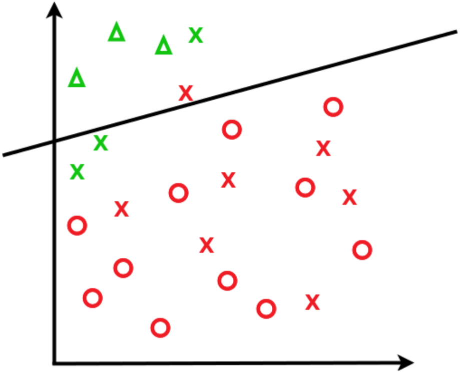
- Predictions for a given class can be wrong or correct in two ways.

Example: Evaluating classifier decisions



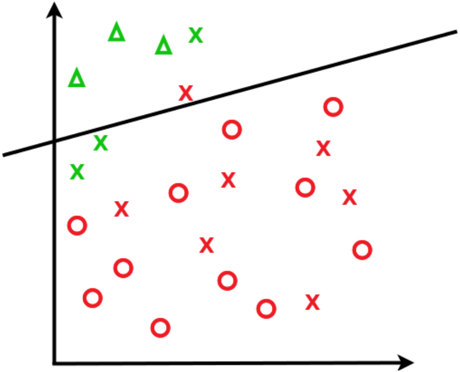
- Predictions for a given class can be wrong or correct in two ways.

Example: Evaluating classifier decisions



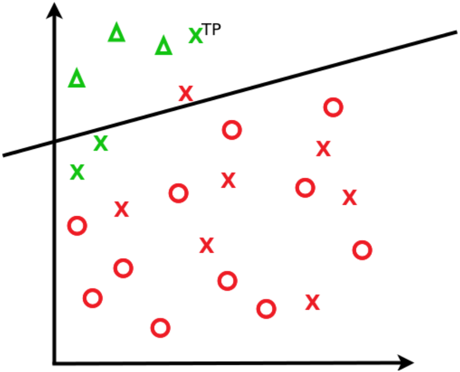
- Predictions for a given class can be wrong or correct in two ways.

Example: Evaluating classifier decisions



- Predictions for a given class can be wrong or correct in two ways.

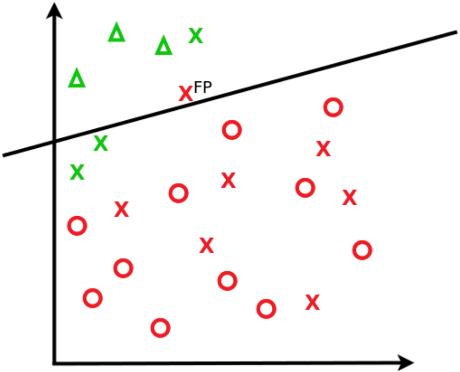
	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)



- Predictions for a given class can be wrong or correct in two ways.

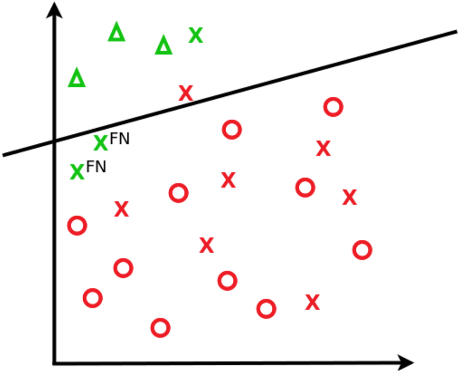
	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)

Example: Evaluating classifier decisions



- Predictions for a given class can be wrong or correct in two ways.

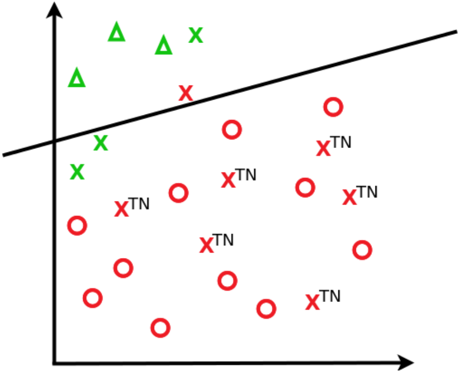
	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)



- Predictions for a given class can be wrong or correct in two ways.

	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)

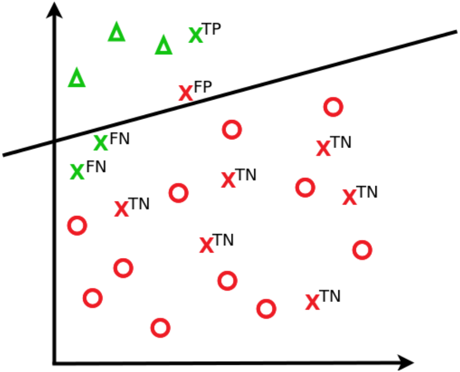
Example: Evaluating classifier decisions



- Predictions for a given class can be wrong or correct in two ways.

	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)

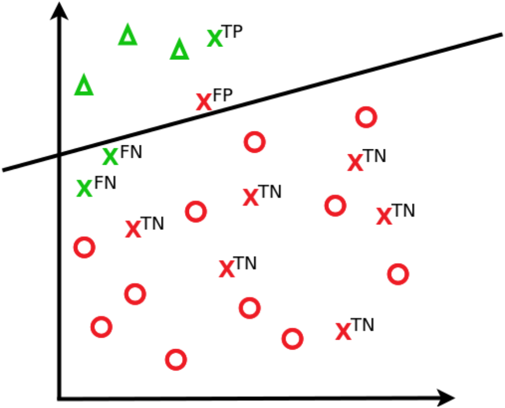
Example: Evaluating classifier decisions



- Predictions for a given class can be wrong or correct in two ways.

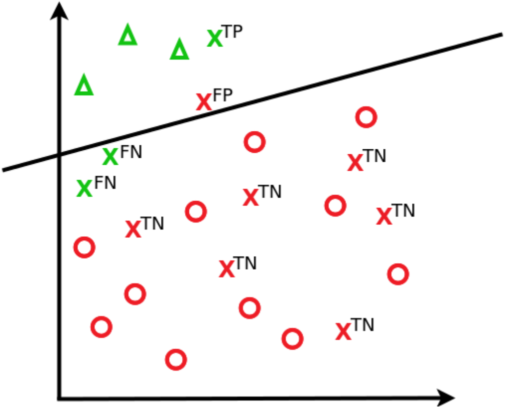
	gold = positive	gold = negative
prediction = positive	true positive (TP)	false positive (FP)
prediction = negative	false negative (FN)	true negative (TN)

Example: Evaluating classifier decisions



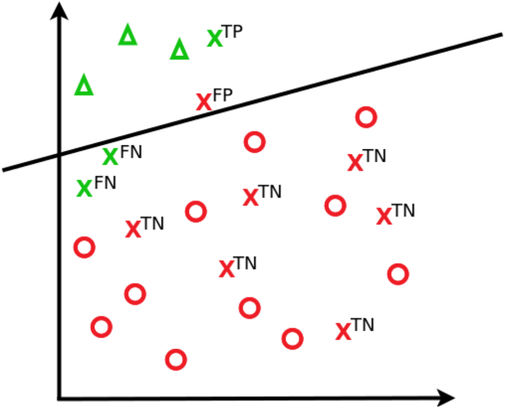
Accuracy $= \frac{TP+TN}{N}$

Example: Evaluating classifier decisions



Accuracy $= \frac{TP+TN}{N}$
$= \frac{1+6}{10} = 0.7$

Example: Evaluating classifier decisions



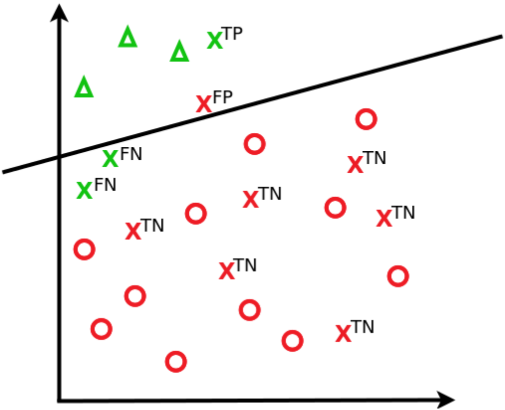
Accuracy $= \frac{TP+TN}{N}$

$= \frac{1+6}{10} = 0.7$

Precision $= \frac{TP}{TP+FP}$

Recall $= \frac{TP}{TP+FN}$

Example: Evaluating classifier decisions

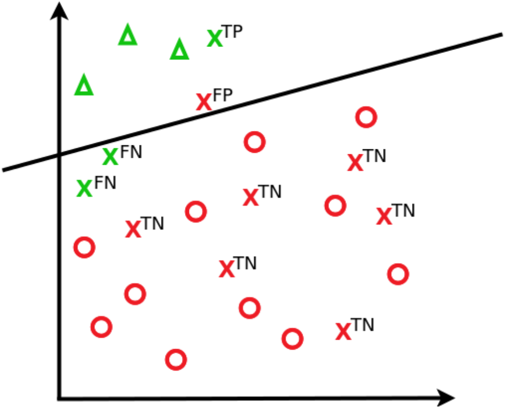


Accuracy $= \frac{TP+TN}{N}$
$= \frac{1+6}{10} = 0.7$

Precision $= \frac{TP}{TP+FP}$
$= \frac{1}{1+1} = 0.5$

Recall $= \frac{TP}{TP+FN}$
$= \frac{1}{1+2} = 0.33$

Example: Evaluating classifier decisions



Accuracy $= \frac{TP+TN}{N}$
$= \frac{1+6}{10} = 0.7$

Precision $= \frac{TP}{TP+FP}$
$= \frac{1}{1+1} = 0.5$

Recall $= \frac{TP}{TP+FN}$
$= \frac{1}{1+2} = 0.33$

F-score
$= 2 \times \frac{precision \times recall}{precision + recall} = 0.4$

Evaluation measures

- Accuracy $= \frac{TP+TN}{N} = \frac{TP+TN}{TP+TN+FP+FN}$
  - The ratio of correct predictions.
  - Not suitable for unbalanced numbers of positive / negative examples.
- Precision $= \frac{TP}{TP+FP}$
  - The number of detected class members that were correct.
- Recall $= \frac{TP}{TP+FN}$
  - The number of actual class members that were detected.
  - Trade-off: Positive predictions for all examples would give 100% recall but (typically) terrible precision.
- F-score $= 2 \times \frac{precision \times recall}{precision + recall}$
  - Balanced measure of precision and recall (harmonic mean).

Evaluating multi-class predictions

Macro-averaging

- Sum precision and recall for each class, and then compute global averages of these.
- The macro average will be highly influenced by the small classes.

Evaluating multi-class predictions

Macro-averaging

- Sum precision and recall for each class, and then compute global averages of these.
- The macro average will be highly influenced by the small classes.

Micro-averaging

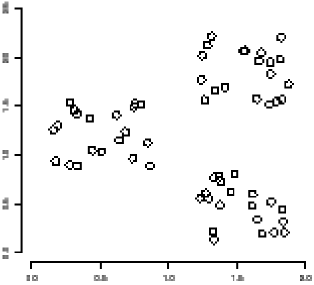
- Sum TPs, FPs, and FNs for all points/objects across all classes, and then compute global precision and recall.
- The micro average will be highly influenced by the large classes.

Classification:

(We will return to classification later in the term.)

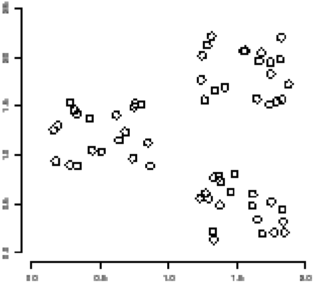
Cluster analysis (*klyngeanalyse*)

- A cluster: ‘A group of similar things or people positioned or occurring closely together.’ *Oxford Dictionaries*
- Cf. the contiguity hypothesis in classification



Cluster analysis (*klyngeanalyse*)

- A cluster: 'A group of similar things or people positioned or occurring closely together.' *Oxford Dictionaries*
- Cf. the contiguity hypothesis in classification



Clustering or cluster analysis

- Unsupervised learning from unlabeled data.
- Automatically group similar objects together into k categories.
- No pre-defined classes:
- We only specify the features and similarity measure (and k, usually).

Example applications of cluster analysis

- Clustering for understanding or knowledge acquisition: visualization and exploratory data analysis.

Example applications of cluster analysis

- Clustering for understanding or knowledge acquisition: visualization and exploratory data analysis.
- Many applications within IR, e.g.:
  - Speed up search: First retrieve the most relevant cluster, then retrieve documents from within the cluster.
  - Presenting the search results: Instead of ranked lists, organize the results as clusters.

Example applications of cluster analysis

- Clustering for understanding or knowledge acquisition: visualization and exploratory data analysis.
- Many applications within IR, e.g.:
  - Speed up search: First retrieve the most relevant cluster, then retrieve documents from within the cluster.
  - Presenting the search results: Instead of ranked lists, organize the results as clusters.
- Dimensionality reduction: class-based features.

Example applications of cluster analysis

- Clustering for understanding or knowledge acquisition: visualization and exploratory data analysis.
- Many applications within IR, e.g.:
  - Speed up search: First retrieve the most relevant cluster, then retrieve documents from within the cluster.
  - Presenting the search results: Instead of ranked lists, organize the results as clusters.
- Dimensionality reduction: class-based features.
- Social network analysis; identify sub-communities and user segments.
- Product recommendations, demographic analysis, news aggregation, ...

Main types of clustering methods

Hierarchical

- Creates a tree structure of hierarchically nested clusters.

Flat

- Tries to directly decompose the data into a set of clusters.
- What we will focus on.

Flat clustering

- Given a set of objects $O = \{o_1, \ldots, o_n\}$, construct a set of clusters $C = \{c_1, \ldots, c_k\}$, where each object o_i is assigned to a cluster c_j.
- $=$ a partition.
- Parameters:
  - The cardinality k (the number of clusters).
  - The similarity function s.

Flat clustering

- Given a set of objects $O = \{o_1, \ldots, o_n\}$, construct a set of clusters $C = \{c_1, \ldots, c_k\}$, where each object o_i is assigned to a cluster c_j.
- $=$ a partition.
- Parameters:
  - The cardinality k (the number of clusters).
  - The similarity function s.
- Formally defined as an optimization problem:
- We want to find an assignment $\gamma : O \rightarrow C$ that optimizes some objective function $F_s(\gamma)$.
- In general terms, we want to optimize for:
  - High intra-cluster similarity
  - Low inter-cluster similarity

Flat clustering (cont'd)

Optimization problems are search problems:

- There's a finite number of possible partitionings of O.
- Naive solution: enumerate all possible assignments $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ and choose the best one,

$$\hat{\gamma} = \underset{\gamma \in \Gamma}{\arg\min}\, F_s(\gamma)$$

Flat clustering (cont'd)

Optimization problems are search problems:

- There's a finite number of possible partitionings of O.
- Naive solution: enumerate all possible assignments $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ and choose the best one,

$$\hat{\gamma} = \arg\min_{\gamma \in \Gamma} F_s(\gamma)$$

- Problem: Exponentially many possible partitions.
- Approximate the solution by iteratively improving on an initial (possibly random) partition until some stopping criterion is met.

k-means

- Unsupervised variant of the Rocchio classifier.
- Goal: Partition the n observed objects into k clusters C so that each point $\boldsymbol{x}_j$ belongs to the cluster c_i with the nearest centroid $\boldsymbol{\mu}_i$.
- Typically assumes Euclidean distance as the similarity function s.

k-means

- Unsupervised variant of the Rocchio classifier.
- Goal: Partition the n observed objects into k clusters C so that each point $\boldsymbol{x}_j$ belongs to the cluster c_i with the nearest centroid $\boldsymbol{\mu}_i$.
- Typically assumes Euclidean distance as the similarity function s.
- The optimization problem: For each cluster, minimize the *within-cluster sum of squares*, $F_s = \mathrm{WCSS}$:

$$\mathrm{WCSS} = \sum_{c_i \in C} \sum_{\boldsymbol{x}_j \in c_i} \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_i \right\|^2$$

- Equivalent to minimizing the average squared distance between objects and their cluster centroids (since n is fixed) – a measure of how well each centroid represents the members assigned to the cluster.

k-means (cont'd)

- Goal: Partition the n observed objects into k clusters C so that each point $\boldsymbol{x}_j$ belongs to the cluster c_i with the nearest centroid $\boldsymbol{\mu}_i$.

Algorithm

Initialize: Randomly select k centroid seeds.

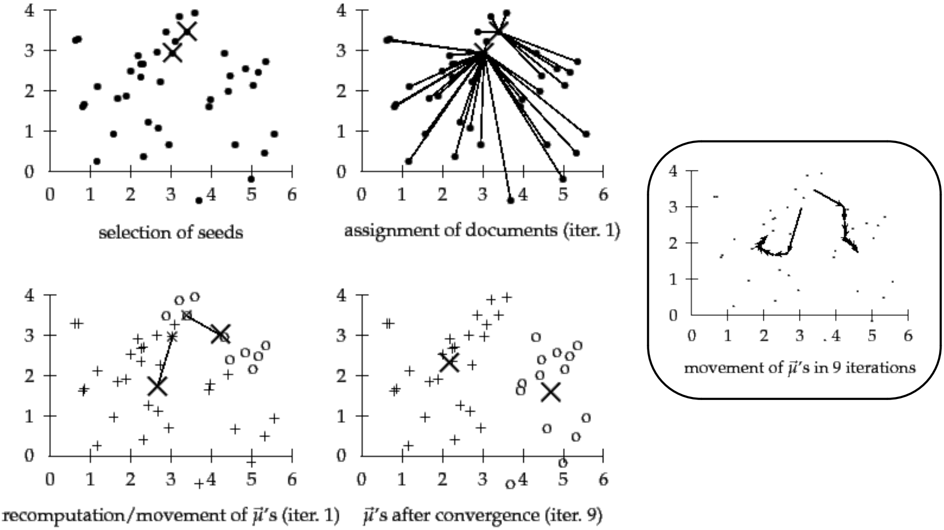
Iterate:

- Assign each object to the cluster with the nearest centroid.
- Compute new centroids for the clusters.

Terminate: When stopping criterion is satisfied.

- In short, we iteratively reassign memberships and recompute centroids until the configuration stabilizes.

k-means example for $k = 2$ in R^2 (Manning, Raghavan & Schütze 2008)



selection of seeds

assignment of documents (iter. 1)

movement of $\vec{\mu}$'s in 9 iterations

recomputation/movement of $\vec{\mu}$'s (iter. 1)

$\vec{\mu}$'s after convergence (iter. 9)

Comments on k-means

Possible termination criteria

- Fixed number of iterations
- Clusters or centroids are unchanged between iterations.
- Threshold on the decrease of the objective function (absolute or relative to previous iteration)

Comments on k-means

Possible termination criteria

- Fixed number of iterations
- Clusters or centroids are unchanged between iterations.
- Threshold on the decrease of the objective function (absolute or relative to previous iteration)

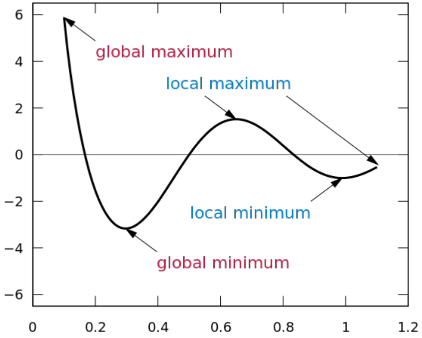
Some close relatives of k-means

- k-medoids: Like k-means but uses medoids instead of centroids to represent the cluster centers.
- Fuzzy c-means (FCM): Like k-means but assigns soft memberships in $[0, 1]$, where membership is a function of the centroid distance.
  - The computations of both WCSS and centroids are weighted by the membership function.

Properties of k-means

- The time complexity is linear, $\mathrm{O}(kn)$.
- WCSS is monotonically decreasing (or unchanged) for each iteration.

- Guaranteed to converge but not to find the global minimum.
- Possible solution: multiple random initializations.
- (k-means is non-deterministic)



Comments on k-means

'Seeding'

- We initialize the algorithm by choosing random seeds that we use to compute the first set of centroids, e.g:
  - pick k random objects from the collection;
  - pick k random points in the space;
  - pick k sets of m random points and compute centroids for each set; etc.
- The seeds can have a large impact on the resulting clustering.
- Outliers are troublemakers.

Flat Clustering: The good and the bad

Pros

- Conceptually simple, and easy to implement.
- Efficient. Typically linear in the number of objects.

Cons

- The dependence on random seeds as in k-means makes the clustering non-deterministic.
- The number of clusters k must be pre-specified. Often no principled means of *a priori* specifying k.
- Not as informative as the more structured clusterings produced by hierarchical methods.

Connecting the dots

- Focus of the last two lectures: Rocchio / nearest centroid classification, kNN classification, and k-means clustering.
- Note how k-means clustering can be thought of as performing Rocchio classification in each iteration.

Connecting the dots

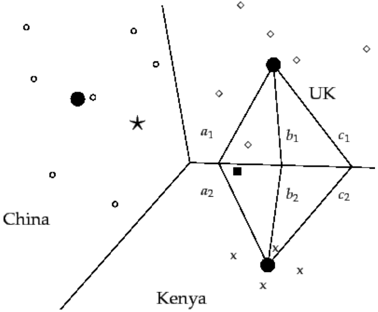
- Focus of the last two lectures: Rocchio / nearest centroid classification, kNN classification, and k-means clustering.
- Note how k-means clustering can be thought of as performing Rocchio classification in each iteration.
- Moreover, Rocchio can be thought of as a 1 Nearest Neighbor classifier with respect to the centroids.

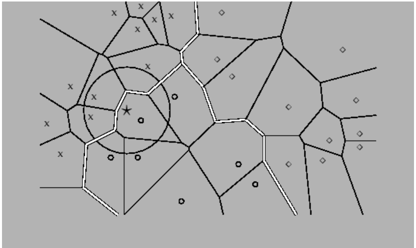
Connecting the dots

- Focus of the last two lectures: Rocchio / nearest centroid classification, kNN classification, and k-means clustering.
- Note how k-means clustering can be thought of as performing Rocchio classification in each iteration.
- Moreover, 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 both Rocchio and k-means, we're partitioning the observations according to the Voronoi diagram generated by the centroids.





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

Tying up a loose end

BoW representations of text

- So far we've been assuming BoW features for representing documents.
- Often also be used for representing other units of texts, like sentences.
- Many sentence-classification tasks in NLP.
- Example: polarity classification (part of sentiment analysis).

I was impressed, this was not bad!

$$\Rrightarrow$$

{was, was, !, not, I, impressed, bad, this }

- What is missing with a BoW representation?

Dealing with compositionality

I was impressed, this was not bad!

$\neq$

I was not impressed, this was bad!

- Will have the same BoW representation! :(

Dealing with compositionality

I was impressed, this was not bad!
$\neq$
I was not impressed, this was bad!

- Will have the same BoW representation! :(
- A simplistic but much-used approximation to capture ordering constraints: n-grams (typically bigrams and trigrams).
- Ordered sub-sequences of n words.

{was, was, !, not, I, impressed, bad, this }

vs.

{'I was', 'was impressed' ... 'was not', 'not bad', 'bad, !' }

No information sharing

- No information sharing between features.
- All features are equally distinct.

- The pizza was great
- The margeritha was awesome
- The dog was sick

- Would be nice if our BoW representations knew that *pizza* and *margeritha* are similar to each other (but not to *dog*).
- We've discussed one possible approach in this lecture. . . What?
- Will return to this issue in a few weeks.

Next lecture

- Focus on *words* rather than *documents*.
- Distributional models of word meaning (lexical semantics).
- Example tasks for evaluating word vectors
- Lecturers:
  - Eivind Alexander Bergem
  - Samia Touileb