INF4820, Algorithms for AI and NLP:

Evaluating Classifiers
Clustering

Erik Velldal
University of Oslo

Sept. 18, 2012

Topics for today

Classification

- Recap
- Evaluating classifiers
  - Accuracy, precision, recall and F-score

Clustering

- Unsupervised machine learning for class discovery.
- Flat vs. hierarchical clustering.
- Example of flat / partional clustering: k-means clustering.

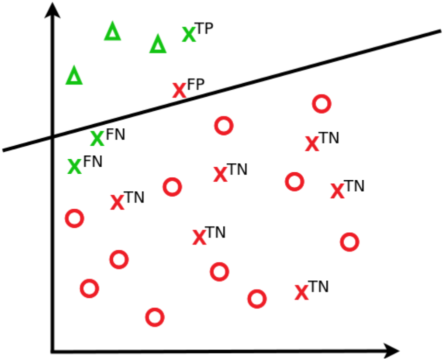
Topics we covered last week

- Supervised vs unsupervised learning.
- Vectors space classification.
- How to represent classes and class membership.
- Rocchio + kNN.
- Linear vs non-linear decision boundaries.

Testing a classifier

- We've seen how vector space classification amounts to computing the boundaries in the space that separate the class regions; *the decision boundaries*.
- To evaluate the boundary, we measure the number of correct classification predictions on unseeen test items.
  - Many ways to do this...
- Why can't we test on the training data?
- We want to test how well a model *generalizes* on a held-out test set.
- (Or, if we have little data, by n-fold cross-validation.)
- Labeled test data is sometimes refered to as the gold standard.

Example: Evaluating classifier decisions



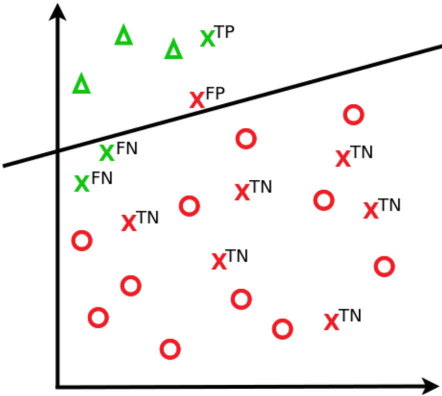
- Predictions for a given class can be wrong or correct in two ways:

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

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.
- $F\text{-}score = \frac{2 \times precision \times recall}{precision + recall}$
  - Balanced measure of precision and recall (harmonic mean).

Example: Evaluating classifier decisions



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

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

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

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

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.

Two categorization tasks in machine learning

Classification

- Supervised learning, requiring labeled training data.
- Given some training set of examples with class labels, train a classifier to predict the class labels of unseen objects.

Clustering

- Unsupervised learning from unlabeled data.
- Automatically group similar objects together.
- No pre-defined classes: we only specify the similarity measure.
- Described by ? (?) as "*the search for structure in data*".
- General objective:
  - Partition the data into subsets, so that the similarity among members of the same group is high (homogeneity) while the similarity between the groups themselves is low (heterogeneity).

Example applications of cluster analysis

- Visualization and exploratory data analysis.
- Generalization and abstraction. “Reason by analogy”.
  - Can have class-based models, even without predefined classes.
  - Helps alleviating the sparse data problem.
- Many applications within IR. Examples:
  - 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 (see e.g. `clusty.com`).
- News aggregation / topic directories.
- Social network analysis; identify sub-communities and user segments.
- Image segmentation, product recommendations, demographic analysis, . . .

Types of clustering methods

Different methods can be divided according to the *memberships* they create and the *procedure* by which the clusters are formed:

$$\text{Procedure} \begin{cases} \text{Flat} \\ \text{Hierarchical} \begin{cases} \text{Agglomerative} \\ \text{Divisive} \end{cases} \\ \text{Hybrid} \end{cases}$$

$$\text{Memberships} \begin{cases} \text{Hard} \\ \text{Soft} \\ \text{Disjunctive} \end{cases}$$

Types of clustering methods (cont'd)

Hierarchical

- Creates a tree structure of hierarchically nested clusters
- Divisive (top-down): Let all objects be members of the same cluster; then successively split the group into smaller and maximally dissimilar clusters until all objects is its own singleton cluster.
- Agglomerative (bottom-up): Let each object define its own cluster; then successively merge most similar clusters until only one remains.

Flat

- Often referred to as partitional clustering when assuming hard and disjoint clusters. (But can also be soft.)
- Tries to directly decompose the data into a set of clusters.

Flat clustering

- Given a set of objects $O = \{o_1, \ldots, o_n\}$, a hard flat clustering algorithm seeks to construct a set of clusters $C = \{c_1, \ldots, c_k\}$, where each object o_i is assigned to a single cluster c_i.
- The cardinality k ($=$ the number of clusters) must typically be manually specified as a parameter to the algorithm.
- But the most important parameter is the similarity function s.
- More formally, we want to define an assignment $\gamma : O \rightarrow C$ that optimizes some objective function $F_s(\gamma)$.
- The objective function is defined in terms of the similarity function, and generally 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 of partitionings of O.
- Naive solution: enumerate all possible assignments $\Gamma = \{\gamma_1, \dots, \gamma_m\}$ 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 $\vec{x}_j$ belongs to the cluster c_i with the nearest centroid $\vec{\mu}_i$.
- Typically assumes Euclidean distance as the similarity function s.
- The optimization problem: For each cluster, minimize the *within-cluster sum of squares*, $F_s = \mathrm{WCSS}$:

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

- WCSS also amounts to the more general measure of how well a model fits the data known as the *residual sum of squares* (RSS).
- Minimizing RSS is 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)

Algorithm

Initialize: Compute centroids for k seeds.

Iterate:

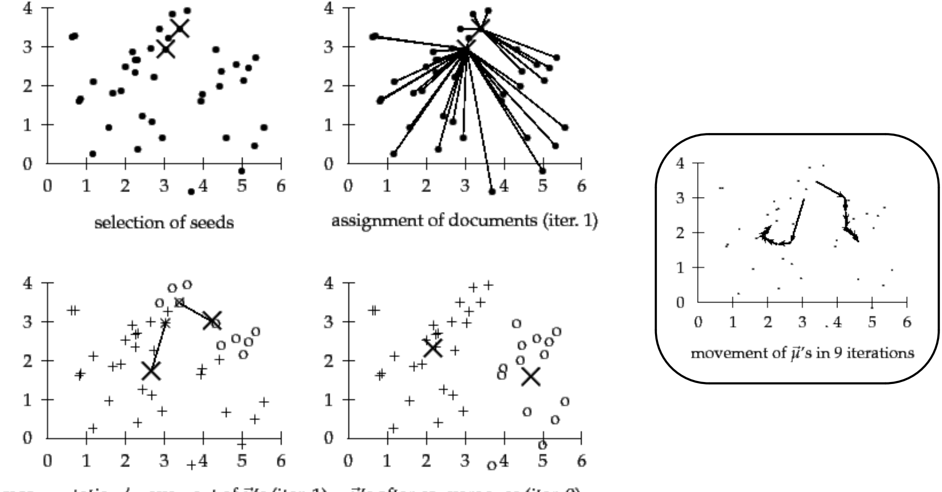
- Assign each object to the cluster with the nearest centroid.
- Compute new centroids for the clusters.

Terminate: When stopping criterion is satisfied.

Properties

- In short, we iteratively reassign memberships and recompute centroids until the configuration stabilizes.
- WCSS is monotonically decreasing (or unchanged) for each iteration.
- Guaranteed to converge but not to find the global minimum.
- The time complexity is linear, $\mathrm{O}(kn)$.

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



selection of seeds

assignment of documents (iter. 1)

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

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

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

Comments on k-Means

"Seeding"

- We initialize the algorithm by choosing random *seeds* that we use to compute the first set of centroids.
- Many possible heuristics for selecting the seeds:
  - 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;
  - compute an hierarchical clustering on a subset of the data to find k initial clusters; etc..
- The initial seeds can have a large impact on the resulting clustering (because we typically end up only finding a local minimum of the objective function).
- Outliers are troublemakers.

Comments on k-Means

Possible termination criterions

- 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.

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 the random seeds makes the clustering non-deterministic.
- The number of clusters k must be pre-specified. Often no principled means of *a priori* specifying k.
- The clustering quality often considered inferior to that of the less efficient hierarchical methods.
- Not as informative as the more stuctured clusterings produced by hierarchical methods.

Next week

- Agglomerative vs. divisive hierarchical clustering
- Reading: Chapter 17 in Manning, Raghavan & Schütze (2008), *Introduction to Information Retrieval*; `http://informationretrieval.org/` (see course web-page for the relevant sections).