Machine Learning

- Can be roughly divided into:
  - Supervised Learning: Trying to learn in order to predict an class or a value
  - Unsupervised Learning: Trying to group similar examples together or to find interesting patterns in the data

Supervised Learning

- Algorithms (small example set)
  - Decision Tree Learning
  - Rule Induction
  - Neural Networks
  - SVM
  - ...

Unsupervised Learning

- Algorithms (small example set)
  - Clustering
    - K-Means, Spectral Clustering, ...
  - Local Pattern Mining
    - Item set mining, sub-sequence mining, subgraph mining
    - Association Rules
  - ...

Foundations of AI
17. Machine Learning Revisted
Supervised and Unsupervised Learning

Wolfram Burgard, Bernhard Nebel, and Andreas Karwath
Supervised Learning: Rule Induction

• Method 1:
  – Learn decision tree, convert to rules

• Method 2:
  – Sequential covering algorithm:
    • Learn one rule with high accuracy, any coverage
    • Remove positive examples covered by this rule
    • Repeat

Sequential Covering Algorithm
Sequential-Covering(Target_attribute, Attributes, Examples, Threshold)
Output: Set of Rules

• Learned_rules ← { }
• Rule ← Learn-one-rule(Target_attribute, Attributes, Examples)
• While Performance(Rule, Examples) > Threshold, do
  • Learned_rules ← Learned_rules ∪ {Rule}
  • Examples ← Examples / {examples correctly classified by Rule}
  • Rule ← Learn-one-rule(Target_attribute, Attributes, Examples)
• Learned_rules ← sort Learned_rules according to Performance over Examples
• return Learned_rules

EnjoySports

<table>
<thead>
<tr>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>sunny</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>cold</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>change</td>
<td>no</td>
</tr>
<tr>
<td>sunny</td>
<td>sunny</td>
<td>high</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
</tbody>
</table>

Learn-One-Rule

IF Wind=weak THEN PlayTennis=yes
IF Wind=strong THEN PlayTennis=yes
IF Humidity=normal THEN PlayTennis=yes
IF Humidity=high THEN PlayTennis=no
...
Learn One Rule

General-to-Specific Search:
• Consider the most general rule (hypothesis) which matches every instances in the training set.
• Repeat
  – Add the attribute that most improves rule performance measured over the training set.
• Until the hypothesis reaches an acceptable level of performance.

General-to-Specific Beam Search (CN2):
• Rather than considering a single candidate at each search step, keep track of the k best candidates.

Learn One Rule

While Pos, do
Learn a NewRule
- NewRule := most general rule possible
- NewRuleNeg := Neg
- while NewRuleNeg, do
  1. Candidate_literals := generate candidates
  2. Best_literal := argmax L ∈ Candidate_literals Performance(SpecializeRule(NewRule, L))
  3. add Best_literal to NewRule preconditions
  4. NewRuleNeg := subset of NewRuleNeg that satisfies NewRule preconditions
- Learned_rules := Learned_rules + NewRule
- Pos := Pos \{members of Pos covered by NewRule
Return Learned_rules

Subtleties: Learn One Rule

• Easily generalizes to multi-valued target functions
• Choose evaluation function to guide search:
  – Entropy (i.e., information gain)
  – Sample accuracy: \( \frac{n_c}{n} \)
  – m-estimate \( \frac{n_c + mp}{n + m} \)

  • Where \( n_c \) correct rule predictions (support )
  • and \( n \) all predictions (coverage)

Variants of Rule Learning Programs

• Sequential or simultaneous covering of data?
• General to specific, or specific to general?
• Generate-and-test, or example-driven?
• Whether and how to post-prune?
• What statistical evaluation function?
• How to combine predictions for multiple classes?
Ripper

- A state of the art rule-learner (Cohen)
- Key idea:
  - apply reduced error pruning on rule set (IREP)
    - rule IF $c_1$ and $c_2$ and ... and $c_n$ THEN class
    - post prune by consider deleting “$c_i$ and ... and $c_n$”
  - once all rules have been learned optimize rule set $R_1$, ...
    - $R_k$
    - try to improve rules $R_i$ by
      - growing and pruning
      - deleting
- Standard approach by now

Unsupervised Methods: Clustering

<table>
<thead>
<tr>
<th>Sky</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>Water</th>
<th>Forecast</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>warm</td>
<td>normal</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>sunny</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>same</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>cold</td>
<td>high</td>
<td>strong</td>
<td>warm</td>
<td>change</td>
<td>no</td>
</tr>
<tr>
<td>sunny</td>
<td>sunny</td>
<td>high</td>
<td>strong</td>
<td>cool</td>
<td>change</td>
<td>yes</td>
</tr>
</tbody>
</table>

Clustering (1)

- Common technique for statistical data analysis (machine learning, data mining, pattern recognition, ...)
- Classification of a data set into subsets (clusters)
- Ideally, data in each subset have a similar characteristics (proximity according to distance function)

Clustering (2)

- Needed: distance (similarity / dissimilarity) function, e.g., Euclidian distance
- Clustering quality
  - Inter-clusters distance maximized
  - Intra-clusters distance minimized
- The quality depends on
  - Clustering algorithm
  - Distance function
  - The application (data)
Types of Clustering

- **Hierarchical Clustering**
  - Agglomerative Clustering (bottom up)
  - Divisive Clustering (top-down)

- **Partitional Clustering**
  - K-Means Clustering (hard & soft)
  - Gaussian Mixture Models (EM-based)

K-Means Clustering

- Partitions the data into $k$ clusters ($k$ is to be specified by the user)
- Find $k$ reference vectors $m_j, j=1,...,k$ which best explain the data $X$
- Assign data vectors to nearest (most similar) reference $m_i$

$$
\left\|\mathbf{x}^t - m_i\right\| = \min_j \left\|\mathbf{x}^t - m_j\right\|
$$

$\mathbf{x}^t$ is an $r$-dimensional data vector in a real-valued space. $m_i$ is a reference vector (center of cluster = mean).

Reconstruction Error (K-Means as Compression Alg.)

- The total reconstruction error is defined as

$$
E\left(\left\{m_i, j=1,\ldots,k\right\} | \mathbf{X}\right) = \sum_t \sum_i b_i^t \left\|\mathbf{x}^t - m_i\right\|^2
$$

with

$$
b_i^t = \begin{cases} 
1 & \text{if } \left\|\mathbf{x}^t - m_i\right\| = \min_j \left\|\mathbf{x}^t - m_j\right\| \\
0 & \text{otherwise}
\end{cases}
$$

- Find reference vectors which minimize the error
- Taking its derivative with respect to $m_i$ and setting it to 0 leads to

$$
m_i = \frac{\sum_i b_i^t \mathbf{x}^t}{\sum_i b_i^t}
$$

K-Means Algorithm

- Initialize $m_i, i=1,\ldots,k$, for example, to $k$ random $\mathbf{x}^t$
- Repeat
  - For all $\mathbf{x}^t \in \mathcal{X}$
    $$
    b_i^t \leftarrow \begin{cases} 
    1 & \text{if } \left\|\mathbf{x}^t - m_i\right\| = \min_j \left\|\mathbf{x}^t - m_j\right\| \\
    0 & \text{otherwise}
    \end{cases}
    $$
  - For all $m_i, i=1,\ldots,k$
    $$
    m_i \leftarrow \frac{\sum_t b_i^t \mathbf{x}^t}{\sum_t b_i^t}
    $$
- Until $m_i$ converge
- Recompute the cluster centers $m_i$, using current cluster membership
- Assign each $\mathbf{x}^t$ to the closest cluster
K-Means Example

Strength of K-Means

- Easy to understand and to implement
- Efficient $O(nkt)$
  
  $n = \#\text{iterations}, k = \#\text{clusters}, t = \#\text{data points}$
- Converges to a local optimum (global optimum is hard to find)
- Most popular clustering algorithm

Weaknesses of K-Means

- User needs to specify $\#\text{clusters} (k)$
- Sensitive to initialization (strategy: use different seeds)
- Sensitive to outliers since all data points contribute equally to the mean (strategy: try to eliminate outliers)

An example

(A). Random selection of $k$ centers

Iteration 1: (B). Cluster assignment

(C). Re-compute centroids
An example (cont ...)

Iteration 2: (D). Cluster assignment

(E). Re-Compute centroids

Iteration 3: (F). Cluster assignment

(G). Re-Compute centroids

Weaknesses of k-means: Problems with outliers

(A): Undesirable clusters

(B): Ideal clusters

Soft Assignments

• So far, each data point was assigned to exactly one cluster
• A variant called soft k-means allows for making fuzzy assignments
• Data points are assigned to clusters with certain probabilities

Soft K-Means Clustering

• Each data point is given a soft assignment to all means
  \[ c_{tk} = \frac{\exp(-\beta \|x^t - m_k\|^2)}{\sum_i \exp(-\beta \|x^t - m_i\|^2)}, \quad \sum_k c_{tk} = 1 \]
• \( \beta \) is a “stiffness” parameter and plays a crucial role
• Means are updated
  \[ m_k = \frac{\sum_t c_{tk} x^t}{\sum_t c_{tk}} \]
• Repeat assignment and update step until assignments do not change anymore
Soft K-Means Clustering

- Points between clusters get assigned to both of them
- Points near the cluster boundaries play a partial role in several clusters
- Additional parameter $\beta$
- Clusters with varying shapes can be treated in a probabilistic framework (mixtures of Gaussians)

After Clustering

- Dimensionality reduction methods find correlations between features and group features
- Clustering methods find similarities between instances and group instances
- Allows knowledge extraction through number of clusters, prior probabilities, cluster parameters, i.e., center, range of features.
  Example: CRM, customer segmentation

Clustering as Preprocessing

- Estimated group labels $h_j$ (soft) or $b_j$ (hard) may be seen as the dimensions of a new $k$ dimensional space, where we can then learn our discriminant or regressor.
- Local representation (only one $b_j$ is 1, all others are 0; only few $h_j$ are nonzero) vs Distributed representation (After PCA; all $z_j$ are nonzero)

Summary

- K-Means is the most popular clustering algorithm
- It is efficient and easy to implement
- Converges to a local optimum
- A variant of hard k-means exists allowing soft assignments
- Soft k-means corresponds to the EM algorithm which is a general optimization procedure