Machine learning: lecture 17

Tommi S. Jaakkola
MIT CSAIL
tommi@csail.mit.edu

Topics

- Clustering cont'd
  - semi-supervised clustering
  - clustering by dynamics
- Structured probability models
  - hidden Markov models

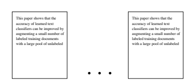
Overview of clustering methods

- Flat clustering methods
  - e.g., mixture models, k-means clustering
- Hierarchical clustering methods:
  - Top-down (splitting)
    - e.g., hierarchical mixture models
  - Bottom-up (merging)
    - e.g., hierarchical agglomerative clustering
- Spectral clustering
- Semi-supervised clustering
- Clustering by dynamics

  Etc.

Semi-supervised clustering

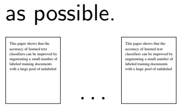
- Let's assume we have some additional *relevance* information about the examples and we'd like clusters to preserve this information as much as possible.



For example, by merging together documents we do not wish to loose information about the words they contain (word distributions).

Semi-supervised clustering

- Let's assume we have some additional *relevance* information about the examples and we'd like clusters to preserve this information as much as possible.



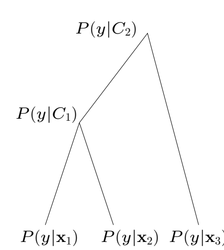
For example, by merging together documents we do not wish to loose information about the words they contain (word distributions).

$\mathbf{x}_i$	Training example (e.g., a text document)
y	Relevance variable (e.g., a word)
$P(y\|\mathbf{x}_i)$	Relevance information (e.g., word distribution)

Semi-supervised clustering cont'd

- We cluster documents $\{\mathbf{x}_i\}$ on the basis of their word distributions $\{P(y|\mathbf{x}_i)\}$
- The word distribution for a cluster is the average word distribution of documents in the cluster

$$P(y|C) = \frac{1}{|C|}\sum_{i \in C} P(y|\mathbf{x}_i)$$



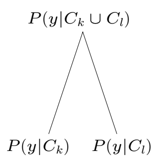
- When merging two clusters we need to take into account their sizes: for example, if $C_2 = C_1 \cup \mathbf{x}_3$ then

$$P(y|C_2) = \frac{1}{2+1}\big(2 \cdot P(y|C_1) + 1 \cdot P(y|\mathbf{x}_3)\big)$$

Semi-supervised clustering cont'd

- We still need to specify a distance metric to determine which clusters to merge and in what order.
- The distance should reflect how much relevance information we loose by merging the clusters

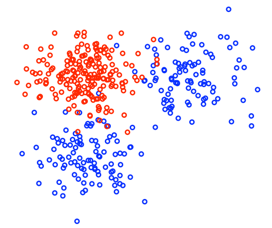
$$
\begin{aligned}
d(C_k, C_l) &= \big(|C_k| + |C_l|\big) \cdot I(y; \text{cluster identity}) \\
&= |C_k| \sum_y P(y|C_k) \log \frac{P(y|C_k)}{P(y|C_k \cup C_k)} \\
&\quad + |C_l| \sum_y P(y|C_l) \log \frac{P(y|C_l)}{P(y|C_k \cup C_k)}
\end{aligned}
$$



Semi-supervised clustering: example

- Suppose we have a set of labeled examples $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ and we take the label as the relevance variable:

$$
P(y|\mathbf{x}_i) = \begin{cases} 1, & \text{if } y = y_i \\ 0, & \text{otherwise} \end{cases}
$$



Clustering by dynamics

- We may wish to cluster time course signals not by direct comparison but in terms of dynamics that governs the signals
  - system behavior monitoring (anomaly detection)
  - biosequencies, processes

  etc.

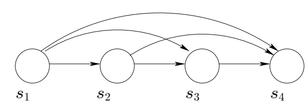
```
1.  0010011001000101000001000011101101010100
2.  0101111110100110101000001000000101011001
3.  1101011000000110110010001101111101011101
4.  11010101111010111101111101101101101000101
```

- We will use *Markov models* to capture the dynamics. The distance metric for clustering is based on similarity of models.

Modeling time course signals

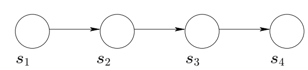
- Full probability model

$$
P(s_1, \ldots, s_t) = P(s_1)P(s_2|s_1)P(s_3|s_2, s_1)P(s_4|s_3, s_2, s_1)\cdots
$$



- First order Markov model

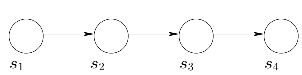
$$
P(s_1, \ldots, s_t) = P(s_1)P(s_2|s_1)P(s_3|s_2)P(s_4|s_3)\cdots
$$



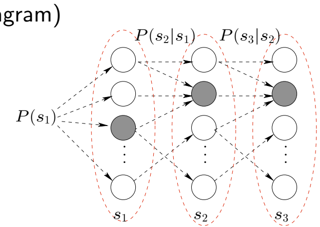
Discrete Markov models

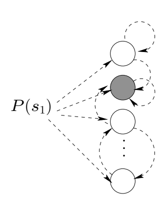
- Representation in terms of variables and dependencies (a graphical model):

$$
P(s_1, \ldots, s_t) = P(s_1)P(s_2|s_1)P(s_3|s_2)P(s_4|s_3)\cdots
$$



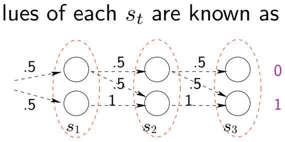
- Representation in terms of state transitions (transition diagram)





Discrete Markov models: properties

- The values of each s_t are known as *states*



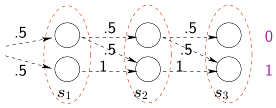
$P(\cdot)$: 0.5
0.5

$P_1(\cdot|\cdot)$: 0.5 0.5
0.0 1.0

- When successive state transitions are governed by the same (one-step) transition probability matrix $P_1(s_t|s_{t-1})$, the Markov model is *homogeneous*

Discrete Markov models: properties

- The values of each s_t are known as *states*



$P(\cdot)$: 0.5, 0.5

$P_1(\cdot|\cdot)$:

0.5	0.5
0.0	1.0

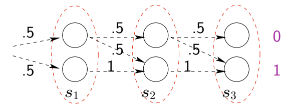
- When successive state transitions are governed by the same (one-step) transition probability matrix $P_1(s_t|s_{t-1})$, the Markov model is *homogeneous*
- Example: a language model

  This $\rightarrow$ is $\rightarrow$ a $\rightarrow$ boring $\rightarrow \ldots$

  Is a homogeneous Markov model appropriate in this case?

Discrete Markov models: properties

- The values of each s_t are known as *states*



$P(\cdot)$: 0.5, 0.5

$P_1(\cdot|\cdot)$:

0.5	0.5
0.0	1.0

- When successive state transitions are governed by the same (one-step) transition probability matrix $P_1(s_t|s_{t-1})$, the Markov model is *homogeneous*
- If after k transitions we can get from any state i to any other state j, the markov chain is *ergodic*.

  More precisely, the multi-step transition probabilities must satisfy $P_k(s|r) > 0$ for all r, s, and some fixed k

Discrete Markov models: ML estimation

$S^{(1)}$: 0010011001000101000001000011101101010100
$S^{(2)}$: 0101111110100110101000001000000101011001

$$l(S^{(1)}, S^{(2)}) = \sum_{i=1,2}\left[\log P(s_1^{(i)}) + \sum_{t=2}^{40} \log P_1(s_t^{(i)}|s_{t-1}^{(i)})\right]$$

- ML estimates of the parameters (initial state and transition probabilities) are based on simple counts:

$$\begin{aligned}\hat{n}(s) &= \#\text{ of times } s_1 = s\\ \hat{n}(r,s) &= \#\text{ number of times } r \rightarrow s\\ \hat{P}(s) &= \frac{\hat{n}(s)}{\sum_{s'}\hat{n}(s')}\\ \hat{P}_1(s|r) &= \frac{\hat{n}(r,s)}{\sum_{s'}\hat{n}(r,s')}\end{aligned}$$

Simple clustering example cont'd

- Four binary sequences of length 40:

  1. 0010011001000101000001000011101101010100
  2. 0101111110100110101000001000000101011001
  3. 1101011000000110110010001101111101011101
  4. 1101010111101011110111101101101101000101

- We can estimate a Markov model based on any subset of the sequences
- We still need to derive the clustering metric based on the resulting transitition probabilities (dynamics)

Clustering metric

- To determine a distance between two arbitrary sequences

$$S^{(1)} = \{s_1^{(1)}, \ldots, s_{n_1}^{(1)}\} \text{ and } S^{(2)} = \{s_1^{(2)}, \ldots, s_{n_2}^{(2)}\},$$

  we measure how well a Markov model would capture the sequences separately or jointly.

$$\begin{aligned}l(S^{(1)}) &= \log P(S^{(1)}|\hat{\theta}_1)\\ l(S^{(2)}) &= \log P(S^{(2)}|\hat{\theta}_2)\\ l(S^{(1)}, S^{(2)}) &= \log P(S^{(2)}|\hat{\theta}) + \log P(S^{(2)}|\hat{\theta})\end{aligned}$$

  where the parameters are ML estimates. Our distance is now defined as

$$d_M(S^{(1)}, S^{(2)}) = l(S^{(1)}) + l(S^{(2)}) - l(S^{(1)}, S^{(2)})$$

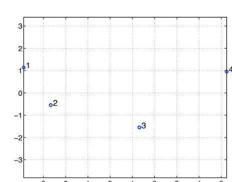
Simple example cont'd

- Four binary sequences of length 40:

  1. 0010011001000101000001000011101101010100
  2. 0101111110100110101000001000000101011001
  3. 1101011000000110110010001101111101011101
  4. 1101010111101011110111101101101101000101

- The resulting pairwise distance matrix:

	1	2	3	4
1	0	0.4155	2.8010	5.8505
2	0.4155	0	1.6849	4.1799
3	2.8010	1.6849	0	1.7682
4	5.8505	4.1799	1.7682	0



(distances need to be recomputed after merging)

Clustering metrics, equivalence

- We have seen two different ways of inducing a metric, one based on information theory, the other through defining a model. These are closely related ideas.

Clustering metrics, equivalence

- We have seen two different ways of inducing a metric, one based on information theory, the other through defining a model. These are closely related ideas.

  Suppose we transform each observed binary sequence 0010011001000... into a bag of pairs {00, 01, 10, ...}.

  Let's use a variable y to refer to the possible pairs (4 of them). For example, $y = 1$ would mean 00.

Clustering metrics, equivalence

- We have seen two different ways of inducing a metric, one based on information theory, the other through defining a model. These are closely related ideas.

  Suppose we transform each observed binary sequence 0010011001000... into a bag of pairs {00, 01, 10, ...}.

  Let's use a variable y to refer to the possible pairs (4 of them). For example, $y = 1$ would mean 00.

  We model each sequence with a distribution over pairs $P(y|S^{(1)})$ whose ML estimate is obtained by counting

$$\hat{P}(y|S^{(1)}) = \frac{n(y)}{n}$$

Clustering metrics, equivalence cont'd

The maximum value of the log-likelihood is given by

$$\begin{aligned} l(S^{(1)}) &= \sum_t \log \hat{P}(y_t|S^{(1)}) = \sum_y n(y) \log \hat{P}(y|S^{(1)}) \\ &= n \sum_y \frac{n(y)}{n} \log \hat{P}(y|S^{(1)}) \\ &= n \sum_y \hat{P}(y|S^{(1)}) \log \hat{P}(y|S^{(1)}) \end{aligned}$$

Similarly, if we merge two sequences

$$\begin{aligned} l(S^{(1)}, S^{(2)}) &= n \sum_y \hat{P}(y|S^{(1)}) \log \hat{P}(y|S^{(1)} \cup S^{(2)}) \\ &\quad + n \sum_y \hat{P}(y|S^{(2)}) \log \hat{P}(y|S^{(1)} \cup S^{(2)}) \end{aligned}$$

Clustering metrics, equivalence cont'd

- The model based metric is now given by

$$\begin{aligned} d_M(S^{(1)}, S^{(2)}) &= l(S^{(1)}) + l(S^{(2)}) - l(S^{(1)}, S^{(2)}) \\ &= n \sum_y \hat{P}(y|S^{(1)}) \log \frac{\hat{P}(y|S^{(1)})}{\hat{P}(y|S^{(1)} \cup S^{(2)})} \\ &\quad + n \sum_y \hat{P}(y|S^{(2)}) \log \frac{\hat{P}(y|S^{(2)})}{\hat{P}(y|S^{(1)} \cup S^{(2)})} \\ &= n \cdot I(y; \text{seq identity}) \end{aligned}$$

Topics

- Clustering cont'd
  - semi-supervised clustering
  - clustering by dynamics
- Structured probability models
  - hidden Markov models

Beyond Markov models

- How can we model
  1. `0101010101010101010101010101010101010101...`

Beyond Markov models

- How can we model
  1. `0101010101010101010101010101010101010101...`
  2. `0010010010010010010010010010010010010010...`

Beyond Markov models

- How can we model
  1. `0101010101010101010101010101010101010101...`
  2. `0010010010010010010010010010010010010010...`
  3. `0100100010000101001000100001010010001000...`

Beyond Markov models

- How can we model
  1. `0101010101010101010101010101010101010101...`
  2. `0010010010010010010010010010010010010010...`
  3. `0100100010000101001000100001010010001000...`
- What about