Memorial University of Newfoundland
Pattern Recognition
Lecture 17, July 11, 2006

http://www.engr.mun.ca/~charlesr
Office Hours: Tuesdays & Thursdays 8:30 - 9:30 PM
EN-3026

Presentations - July 18th and 20th

- All graduate students are to give a 10-12 minute presentation + answer questions for 3-5 minutes
- All students are expected to attend
  - Fill in evaluation sheets for each presentation
- Email me (charles.robertson@verafin.com) by Thursday, July 13th, with:
  - Final project title
  - Requirements for presentation (if any)

“Chance favours the prepared mind.”
Louis Pasteur
Presentations - July 25th

- Suggestions and considerations
  - Think about what your audience already knows and probably doesn’t know
  - Include information on the pattern recognition algorithm (2-4 minutes)
  - Describe the context of your use of the algorithm (1-2 min)
  - Describe how you used the algorithm (2-3 min)
  - Discuss your experiment and results - is the algorithm actually useful? (2-3 min)

Recap

- K-Means
  - Basic algorithm review
  - Extensions - global minimization method
  - Discussion of strategies for dealing with k

- Examples
  - K-Means - basic and global minimization
  - Dendrogram
Feature Selection and Extraction

- Classifier performance depend on a combination of the number of samples, number of features, and complexity of the classifier.
- Ideally, using more features leads to better performance
  - Only true if the class-conditional densities are completely known
  - Not true if the number of training samples small relative to the number of features
- This is known as the ‘peaking phenomenon’, or equivalently ‘the curse of dimensionality’

Fig. 6. Classification error vs. the number of features using the floating search feature selection technique (see text).

Feature Selection and Extraction

In general, we would like to have a classifier or clustering algorithm to use a minimum number of dimensions:

- reduces the number of computations
- statistical estimation reliability (more samples/dimension)

We can often use prior knowledge about the physical experiment to choose the most discriminating measurements, but this is not always the case.

Feature Selection:
Given \( m \) measurements, choose \( n < m \) best as features.

Feature Extraction:
Given \( m \) measurements \( \{ x_i \} \), find \( n < m \) functions \( y_j = f_j(x_1, \ldots, x_m) \) which produce the best features.

Feature Selection

We require:

1. A criterion to establish the best features
2. An algorithm to optimize the chosen criterion

We need to evaluate a measure of the feature’s utility to determine how much it contributes to performance.

\[
\max J(x_k) = P(E|\{x_i, i \neq k\}) - P(E|\{x_i\forall i\})
\]

Can’t usually test classifier before choosing features, so we need some sub-optimal but reasonable criterion.
Feature Selection - Criterion

Interclass distance (normalized by intraclass distance)

2 classes: \[ J_i = \frac{(m_{i1} - m_{i2})^2}{s_{i1}^2 + s_{i2}^2} \]

where \( m_{i1} \) = mean of \( i \)th feature of class 1
and \( s_{i1} \) = scatter (variance) of \( i \)th feature in class 1

k classes: \[ J_i = \sum_{j=1}^{k} \sum_{l=1}^{k} \frac{(m_{ij} - m_{il})^2}{s_{ij}^2 + s_{il}^2} \]

Optimizing

Whether we use distance or some other criterion, we still have to find the best subset of measurements.

For \( m \) measurements, \( n \) features, there are \( \binom{m}{n} = \frac{m!}{n!(m-n)!} \) combinations.

Usually an exhaustive comparison is not feasible.

Need a sub-optimal strategy...
1. Feature Ranking
If $J_i$ = measure of $i^{th}$ feature’s effectiveness, then order features such that:

$$J_1 \geq J_2 \geq ... \geq J_n \geq ... \geq J_m$$

- Take first $n$ features.
- Might not comprise the best subset, since some features might be highly correlated.

2. Incrementally Best Feature
Let $y_1 = x_i$ such that $J_i$ is maximized.

Choose $y_2 = x_j$ such that $J(y_1, x_j) > J(y_1, x_k) \forall k \neq j$

Choose $y_3 = x_k$ such that ... (for $y_1$ and $y_2$)

And so on...

Note: When we start evaluating subsets rather than individual features, we might use a criterion like:

(a) $\frac{|S_B|}{|S_W|}$

(b) $\sum_{i=1}^{k} \sum_{j=1}^{k} (m_i - m_j)^T S_j^{-1} (m_i - m_j)$

(c) $I(C, Y)$
Still don’t necessarily get the best subset since it’s possible that
\[ J(y_2; y_3) > J(y_1; y_2) \]

3. Successive Addition/Deletion
Given \( k \) features \( \{y_1, \ldots, y_k\} \) with \( J_k(\{y\}) \), add \( y_{k+1} \) such that \( J(y_1, \ldots, y_{k+1}) \) is maximized.

Now find \( l, 1 \leq l \leq k \), to maximize \( J(\{y_i, 1 \leq i \leq k+1, i \neq l\}) \)
If this is greater than \( J_k \), then delete \( y_l \) and add the next incrementally best feature.
Feature Selection

In summary:
1. Evaluate feature to determine how much it contributes to performance.
   a. Interclass distance
   b. Information measures
2. Optimize chosen criterion
   a. Rank features by effectiveness and choose best
   b. Incrementally add features to set of chosen features
   c. Successively add and delete features to chosen set