

CSCI 5622 Machine Learning

ML EM (Expectation Maximization)

DATE	READ	DUE
Today, Nov 2	6.12 EM	Project Questions
Wed, Nov 4	<u><i>Semi-supervised Learning</i></u>	
Mon, Nov 9	SSL continued	Exprmnt 1 Write-up

www.RodneyNielsen.com/teaching/CSCI5622-F09/

Instructor: Rodney Nielsen

Assistant Professor Adjunct, CU Dept. of Computer Science

Research Assistant Professor, DU, Dept. of Electrical & Computer Engr.

Research Scientist, Boulder Language Technologies

ML Partially Observable Data

- Some feature values occasionally missing
 - Estimate missing values
 - Most common value (for that class)
 - Mean value (for that class)
 - Learn the value (for that class)
 - In DTs, assume fractional values based on proportions
- When values, θ , are never observed?
 - EM Algorithm can learn these values, provided that their probability distribution is known

ML Expectation Maximization - EM

- Uses
 - Training Bayesian Belief networks
 - Training Radial Basis Function networks
 - Foundation of a number of clustering algorithms / unsupervised learning algorithms
 - Baum-Welch Forward-Backward algorithm; Partially Observable Markov Decision Processes

- Estimating the means of K Gaussians
- $\mathbf{D} = \langle \mathbf{X}, \mathbf{Y} \rangle$ is the data; \mathbf{X} is known, \mathbf{Y} is unknown
- \mathbf{D} is generated by a set of K Normal distributions $\{\mathcal{N}(\boldsymbol{\mu}_k, \sigma^2)\}$ ($\sigma_j = \sigma_k = \sigma: 1 \leq j, k \leq K$)
 - Where $\langle \mathbf{X}, \mathbf{Y} \rangle$, a random variable, is generated by first randomly selecting a k then randomly generating an $\mathbf{x}^{(i)}$ according to $\mathcal{N}(\boldsymbol{\mu}_k, \sigma^2)$
- We want to learn $\boldsymbol{\theta} = \langle \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}_K \rangle$, which can be used to estimate \mathbf{Y}
- Note simplifying assumption of ($\sigma_j = \sigma_k = \sigma: 1 \leq j, k \leq K$)

ML Estimating the mean of 1 Gaussian

- Estimating the mean of one Gaussian is simple
 - Use the maximum likelihood estimate

$$\mu_{ML} = \operatorname{argmin}_{\mu} \sum_{i=1}^N (\mathbf{x}^{(i)} - \mu)^2 = \frac{1}{N} \sum_{i=1}^N \mathbf{x}^{(i)} = \bar{\mathbf{x}}$$

- $\mu_{ML} = \operatorname{mean}(\mathbf{X})$

ML Estimating K Gaussian Means

- How do you estimate K Gaussian means?
 - Use the maximum likelihood estimate

$$\mathbf{Z} = \left\{ \left\langle \mathbf{x}^{(i)}, y_1^{(i)}, \dots, y_K^{(i)} \right\rangle \right\}; y_k^{(i)} = \begin{cases} 1, & \text{if } \mathbf{x}^{(i)} \in \mathcal{N}(\mu_k, \sigma_k) = C_k \\ 0, & \text{otherwise} \end{cases}$$

$$\begin{aligned} \mu_{k,ML} &= \arg \min_{\mu_k} \sum_{i=1}^{N_k} \left(\mathbf{x}^{(k,i)} - \mu^{(k)} \right)^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} \mathbf{x}^{(k,i)} = \bar{\mathbf{x}}^{(k)} \\ &= \bar{\mathbf{x}} : \mathbf{x} \in \left\{ \left\langle \mathbf{x}, y_1 = 0, \dots, y_k = 1, y_{k+1} = 0, \dots \right\rangle \right\} \end{aligned}$$

- I.E., est. class population mean \cong class sample mean
- Why isn't this as trivial under our scenario?

$$\forall i, k : \mathbf{x}^{(i)} \text{ is known and } y_k^{(i)} \text{ is unknown}$$

ML Iterative Re-estimation

- Use EM since we don't know the means of the Gaussians
- Repeatedly recalculate the missing values θ
 - Based on the current hypothesis h for the Gaussian means, compute the *expectation* for the class indicator variables
 - Then based on the predicted data in each class, recalculate the mean of each Gaussian to *maximize* the likelihood of the data

ML EM Algorithm

- Generate an arbitrary hypothesis h
 - I.E., make arbitrary assignments to $\langle \mu_1, \mu_2, \dots, \mu_K \rangle$
- The **(Expectation) E-Step**: Based on the current hypothesis h , calculate the Expected values of the class indicator variables

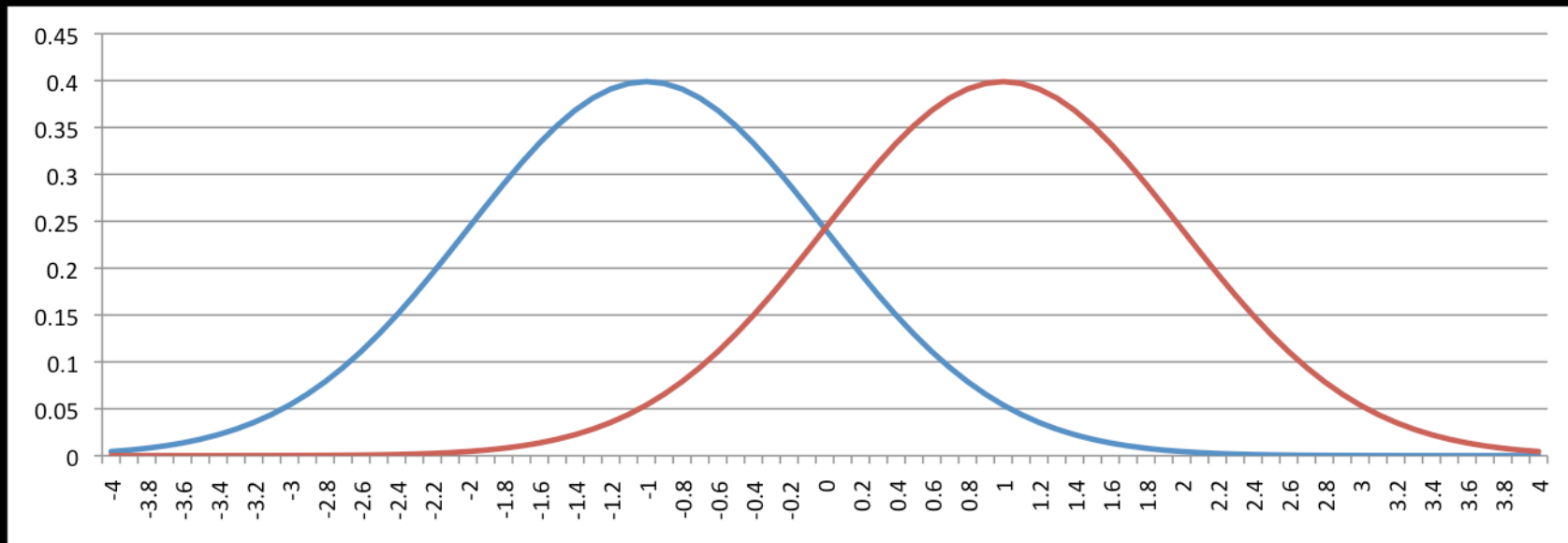
$$\mathbf{Y} = \{\mathbf{y}^{(i)}\} = \{\langle y_1^{(i)}, y_2^{(i)}, \dots, y_K^{(i)} \rangle\}$$

- The **(Maximization) M-Step**: Based on the expected value of the data, $\{\langle \mathbf{x}^{(i)}, \mathbf{y}^{(i)} \rangle\}$, compute h that Maximizes the likelihood of the data

$$h_{ML} = \langle \mu_1, \mu_2, \dots, \mu_K \rangle_{ML}$$

ML Ex: One Attribute & Two Classes

- Each of two classes is governed by its own Normal distribution $\mathcal{N}(\mu_k, \sigma^2)$



ML Ex: One Attribute & Two Classes

- Each of two classes is governed by its own Normal distribution $\mathcal{N}(\mu_k, \sigma^2)$
- The full data is $\{<x^{(i)}, y_1^{(i)}, y_2^{(i)}>\}: 1 \leq i \leq N$
- The observable data is $\{x^{(i)}\}$
- $\{\mu_1, \mu_2\}$ and the $\{y^{(i)}\}$ are unknown
- (Expectation) E-Step: Compute the expected value of $\{y^{(i)}\}$ given the current estimate for θ
- (Maximization) M-Step: Compute a new hypothesis for θ , maximizing the data's likelihood

ML Ex: One Attribute & Two Classes

- (Expectation) E-Step:

$$\begin{aligned} E[y_1^{(i)}] &= \frac{p(x = x^{(i)} | \mu = \mu_1)}{\sum_{k=1}^2 p(x = x^{(i)} | \mu = \mu_k)} \\ &= \frac{\exp\left(-\frac{1}{2\sigma^2} (x^{(i)} - \mu_1)^2\right)}{\exp\left(-\frac{1}{2\sigma^2} (x^{(i)} - \mu_1)^2\right) + \exp\left(-\frac{1}{2\sigma^2} (x^{(i)} - \mu_2)^2\right)} \end{aligned}$$

ML Ex: One Attribute & Two Classes

- (Maximization) M-Step:

$$\mu_{1,ML} = \frac{\sum_{i=1}^N E[y_1^{(i)}] x^{(i)}}{\sum_{i=1}^N E[y_1^{(i)}]}$$

ML EM Performance

- On each iteration, EM increases the likelihood $P(\langle \mathbf{X}, \mathbf{Y} \rangle | h)$, until it reaches a *local* maximum

ML Example

- *example*

ML Questions?

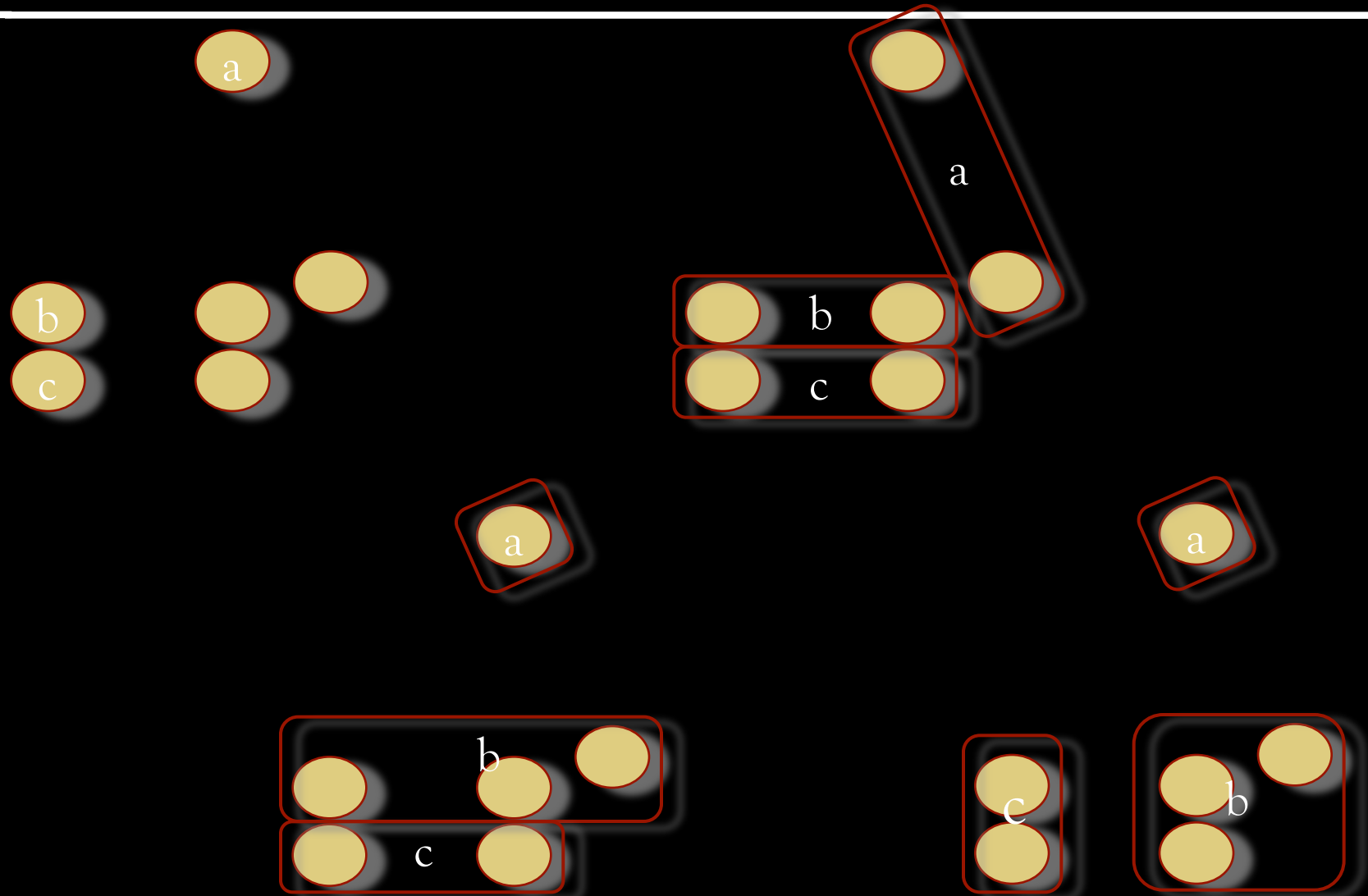
- Questions???

ML Complexity

- k -means is $O(kdN)$
- Until stopping criteria
 - Assign instances to the cluster whose centroid is “nearest”
 - Recalculate centroid = mean of cluster instances

ML

k-means Clustering

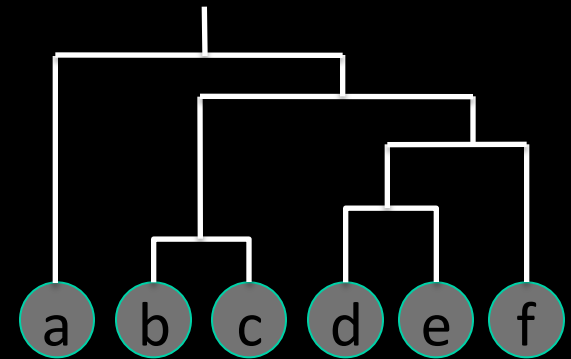


- Assign instances to all clusters with some probability
- Expectation Maximization

ML Types of Clustering

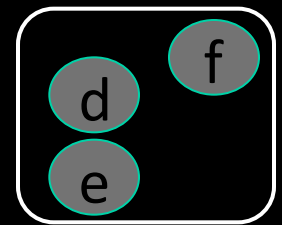
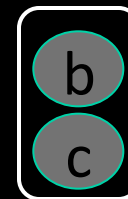
- Hierarchical clustering

- Relation between clusters is expressed in the clustering and represents similarity
- Tree where each node represents a cluster and the children represent subclasses of the parent



- Flat or Non-hierarchical clustering

- Typically pre-specified number of clusters
- Relation between clusters is unknown
- Typically iterative algorithms
 - Start with one set and iteratively improve clusters



ML Hierarchical vs. Non-hierarchical

- Hierarchical clustering
 - Good for in-depth analysis
 - More informative than flat clustering
 - Best algorithm is problem-dependent
 - Computationally intensive vs. most flat clustering
- Non-hierarchical clustering
 - K-means is generally a good algorithm (or EM also)
 - More efficient

ML Types of Clustering

- Hard clustering
 - Each instance is placed in exactly one cluster
 - Hierarchical clustering is hard clustering
 - Drawback: consider clustering words by part-of-speech
- Soft clustering
 - Instances are placed in multiple clusters, typically all clusters probabilistically
 - Usually soft clustering indicates uncertainty
 - In disjunctive clustering, exs belong to multiple clusters
 - Flat clustering can be either hard or soft clustering

ML Non-Hierarchical Clustering

- Typically initialize the k clusters based on a different random seed for each cluster
- Then iteratively improve on this solution by moving instances from one cluster to another
- Hierarchical Clustering required just one pass

ML Questions?

- Questions???

ML Evaluating Clustering Results

- Goals:
 - High intra-cluster similarity (cluster purity)
 - Low inter-cluster similarity (cluster uniqueness)
- However, the clustering quality is dependent
 - Not only on the metrics
 - But also the application
- Therefore, evaluation in context is better
 - But is rarely done

ML Clustering Metrics

- Purity
 - Measures the level of intra-cluster similarity
- Normalized Mutual Information
 - Provides an information theoretic measure
- Rand Index
 - Combines both intra- and inter-cluster assessment
- *F*-measure
 - Also factors in both intra- and inter-cluster quality

ML Notation

- $|\theta|$ = the size of the specified set
- ω_k = the set of instances in the k^{th} cluster
- $\Omega = \{\omega_k\}$ = the set of K clusters produced
- \mathcal{C}_h = the set of instances in class h
- $\mathcal{C} = \{c_h\}$ = the set of H classes

- Assign clusters to the most frequently occurring class in that cluster
- Purity is the accuracy based on these assignments; $0 \leq Purity \leq 1.0$

$$Purity = \frac{1}{N} \sum_{k=1}^K \max_h |\omega_k \cap \mathbf{c}_h|$$

- If $K=N$ (ie, each ex is a cluster), then $Purity=1.0$
- If $K=1$, then $Purity =$ proportion majority class

ML Mutual Information

- Mutual Information, $I(p; q)$, is a measure of
 - how much information you gain
 - about one probability p
 - given another probability q
- In the case of clustering, it is a measure of
 - the information gained
 - about each instance's actual class
 - based on knowing its cluster

ML Mutual Information

$$P(\hat{\theta}_\alpha) \equiv P(\langle \mathbf{x}, y \rangle^{(i)} \in \theta_\alpha)$$

$$I(\Omega; \mathbf{C}) = \sum_{k=1}^K \sum_{h=1}^H P(\hat{\omega}_k, \hat{\mathbf{c}}_h) \log \frac{P(\hat{\omega}_k, \hat{\mathbf{c}}_h)}{P(\hat{\omega}_k)P(\hat{\mathbf{c}}_h)}$$

- $I(\Omega; \mathbf{C})$ ranges from

– 0.0:

- When knowing the cluster gives no additional information over the original distribution

$$P(\hat{\omega}_k, \hat{\mathbf{c}}_h) = P(\hat{\omega}_k)P(\hat{\mathbf{c}}_h)$$

– $\log(K)=\log(H)$:

- When knowing the cluster uniquely identifies the class and all clusters are the same size

ML Mutual Information

$$I(\Omega; \mathcal{C}) = \sum_{k=1}^K \sum_{h=1}^H P(\hat{\omega}_k, \hat{\mathcal{C}}_h) \log \frac{P(\hat{\omega}_k, \hat{\mathcal{C}}_h)}{P(\hat{\omega}_k)P(\hat{\mathcal{C}}_h)}$$

- Like *Purity*, $I(\Omega; \mathcal{C})$ increases w the no. clusters

- If $K=1$:

- All instances are in one class
- Knowing the cluster provides you with no new info

$$P(\hat{\omega}_k, \hat{\mathcal{C}}_h) = P(\hat{\omega}_k)P(\hat{\mathcal{C}}_h)$$

- $I(\Omega; \mathcal{C}) = 0.0$

MI Mutual Information

- Like *Purity*, $I(\Omega; \mathcal{C})$ increases w the no. clusters
 - If $K=H$:
 - Each class is its own cluster
 - Knowing the cluster tells you the class

$$\begin{aligned} I(\Omega; \mathcal{C}) &= \sum_{k=1}^K \sum_{h=1}^H P(\hat{\omega}_k, \hat{\mathbf{c}}_h) \log \frac{P(\hat{\omega}_k, \hat{\mathbf{c}}_h)}{P(\hat{\omega}_k)P(\hat{\mathbf{c}}_h)} \\ &= \sum_{k=1}^K \sum_{h=1}^H \frac{|\omega_k \cap \mathbf{c}_h|}{N} \log \frac{N|\omega_k \cap \mathbf{c}_h|}{|\omega_k||\mathbf{c}_h|} \\ &= \sum_{h=1}^H \frac{|\mathbf{c}_h|}{N} \log \frac{N|\omega_k|}{|\omega_k||\mathbf{c}_h|} = \sum_{h=1}^H P(\mathbf{c}_h) \log \frac{1}{P(\mathbf{c}_h)} \end{aligned}$$

- $\max(I(\Omega; \mathcal{C})) = \log(H)$

ML Mutual Information

- Like *Purity*, $I(\Omega; \mathcal{C})$ increases w the no. clusters
 - If $K=N$:
 - Each instance is in its own class
 - Knowing the cluster tells you the class

$$\begin{aligned} I(\Omega; \mathcal{C}) &= \sum_{k=1}^K \sum_{h=1}^H \frac{|\omega_k \cap \mathcal{C}_h|}{N} \log \frac{N |\omega_k \cap \mathcal{C}_h|}{|\omega_k| |\mathcal{C}_h|} \\ &= \sum_{h=1}^H \frac{|\mathcal{C}_h|}{N} \log \frac{N}{|\mathcal{C}_h|} \\ &= \sum_{h=1}^H P(\mathcal{C}_h) \log \frac{1}{P(\mathcal{C}_h)} \end{aligned}$$

- $I(\Omega; \mathcal{C}) = 1.0$

ML Normalized Mutual Information

- So how do we use to create a measure that rewards inter-cluster uniqueness while still rewarding intra-cluster similarity (*Purity*)?
- Normalize the Mutual Information

$$\begin{aligned}\max[I(\Omega; \mathbf{C})] &= \sum_{h=1}^H P(\mathbf{c}_h) \log \frac{1}{P(\mathbf{c}_h)} \\ &= - \sum_{h=1}^H P(\mathbf{c}_h) \log P(\mathbf{c}_h) \\ &= H(\mathbf{C})\end{aligned}$$

ML Normalized Mutual Information

$$NMI(\Omega, \mathcal{C}) = \frac{I(\Omega; \mathcal{C})}{(H(\Omega) + H(\mathcal{C})) / 2}$$

$$H(\Omega) = - \sum_{k=1}^K P(\omega_k) \log P(\omega_k)$$

if $P(\omega_k) = P(\omega_h) \forall k, h$

$$- \sum_{k=1}^K P(\omega_k) \log P(\omega_k) = -\log P(\omega_k) = \log K$$

$\forall K > H : \log K > \log H \rightarrow H(\Omega(K)) > H(\Omega(H))$

$\rightarrow NMI(\Omega(K), \mathcal{C}) < NMI(\Omega(H), \mathcal{C})$

- All else being equal, a smaller K is better

Rand Index (RI)

- Consider all pairs of instances in the sample
 - There are $N(N-1)/2$ pairs
- Rand Index (RI) = accuracy of these pairings
 - A correct classification decision for similar instances is to place them in the same cluster
 - A correct classification decision for dissimilar instances is to place them in different clusters

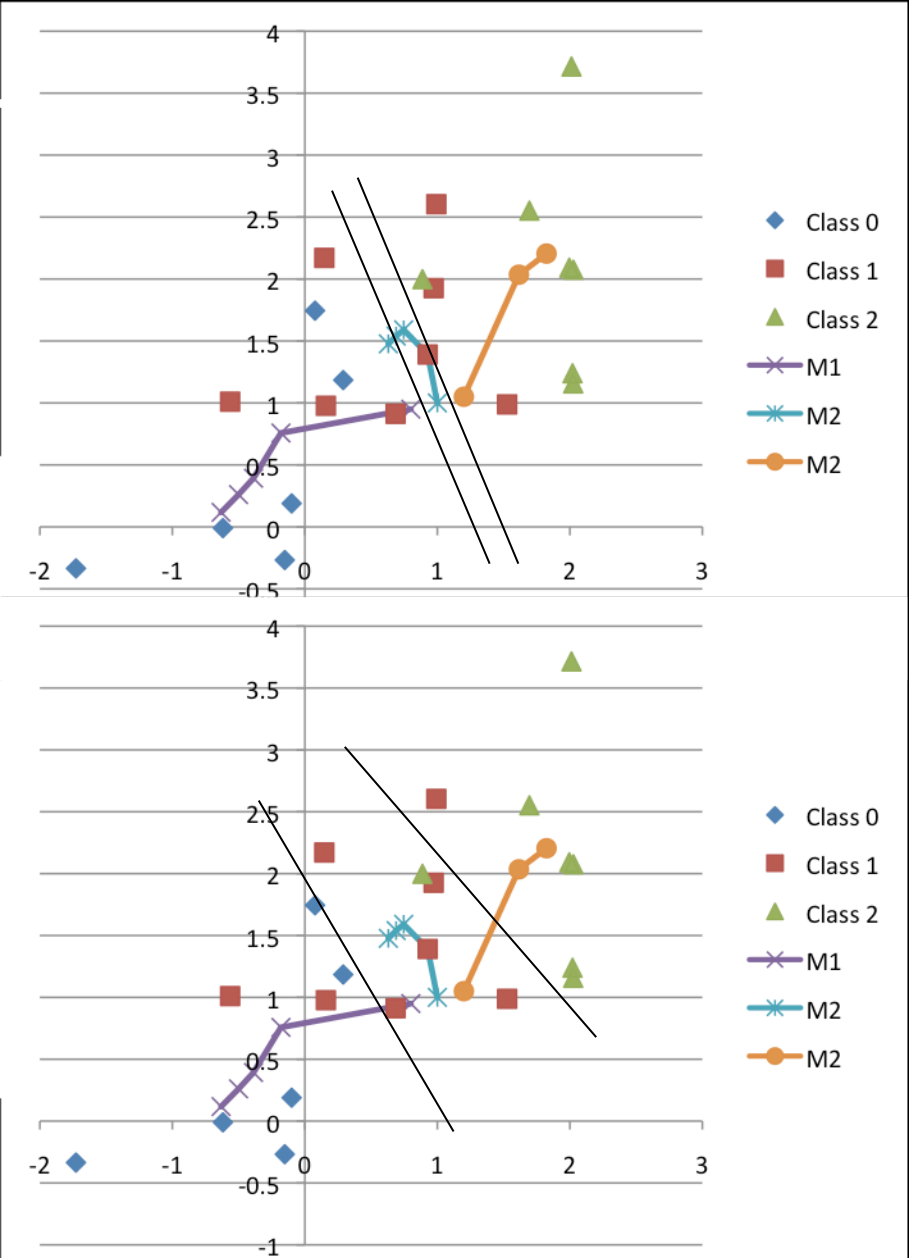
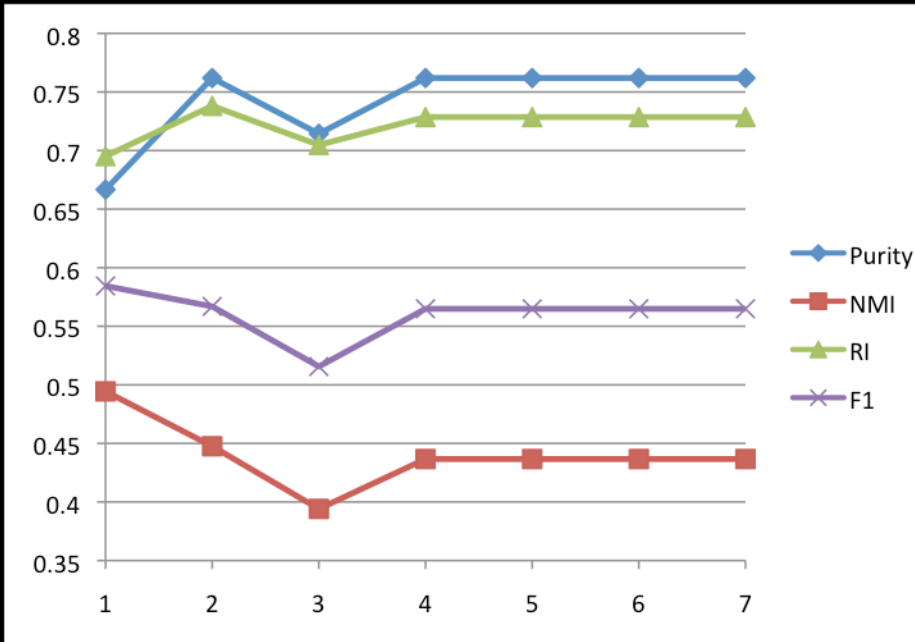
ML Rand Index (RI)

- Count True and False Positives and Negatives
- TP = # similar pairs in the same cluster
- FP = # dissimilar pairs in the same cluster
- TN = # dissimilar pairs in separate clusters
- FN = # similar pairs in separate clusters
- $RI = (TP + TN) / (TP + FP + TN + FN)$
 $= (TP + TN) / (N(N-1)/2)$

- RI weights positives and negatives equally
 - For some applications this is not appropriate
- *F*-measure allows weighting
- $P = TP / (TP + FP) = TP / \# \text{ labeled positive}$
- $R = TP / (TP + FN) = TP / \# \text{ actual positives}$
- $F = (1 + \beta^2)PR / (\beta^2P + R)$

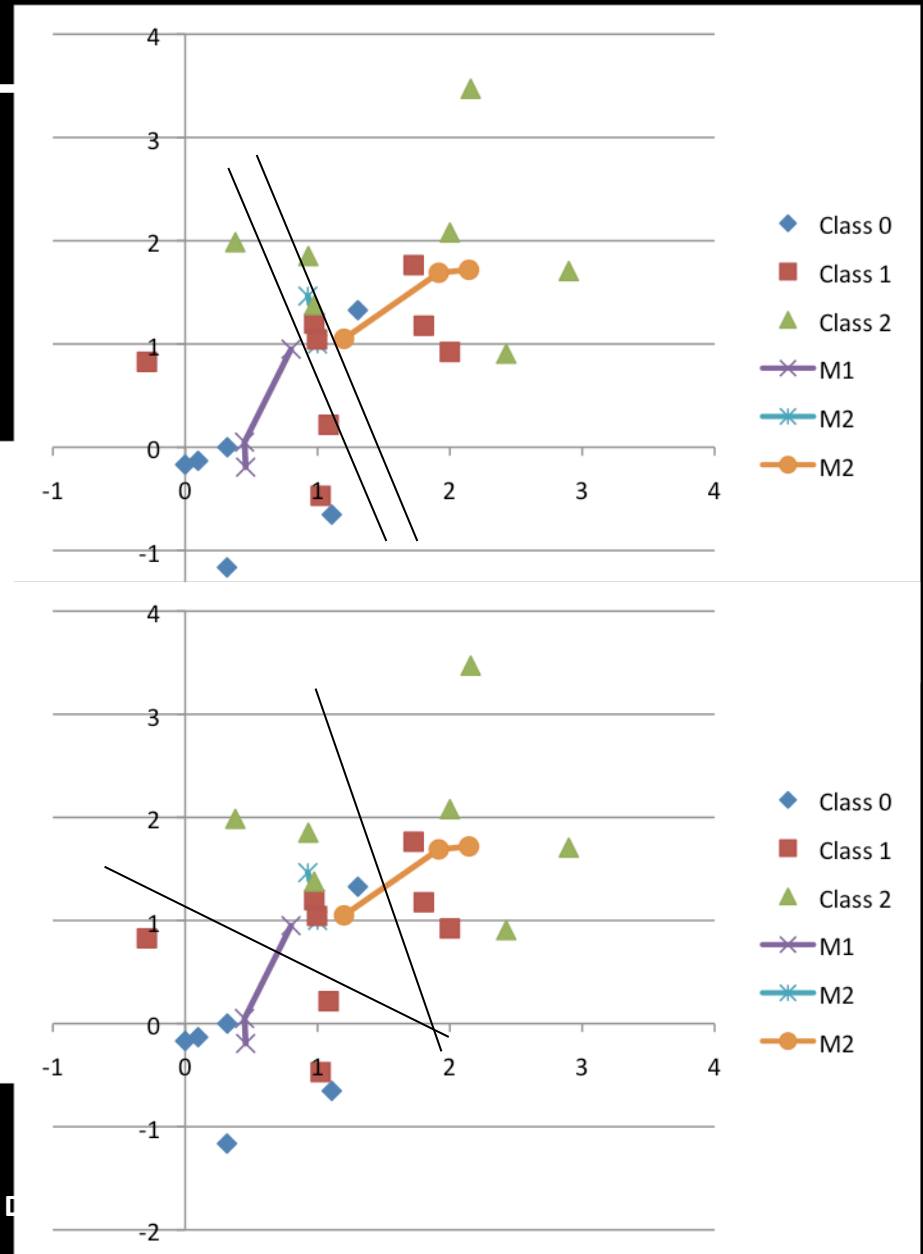
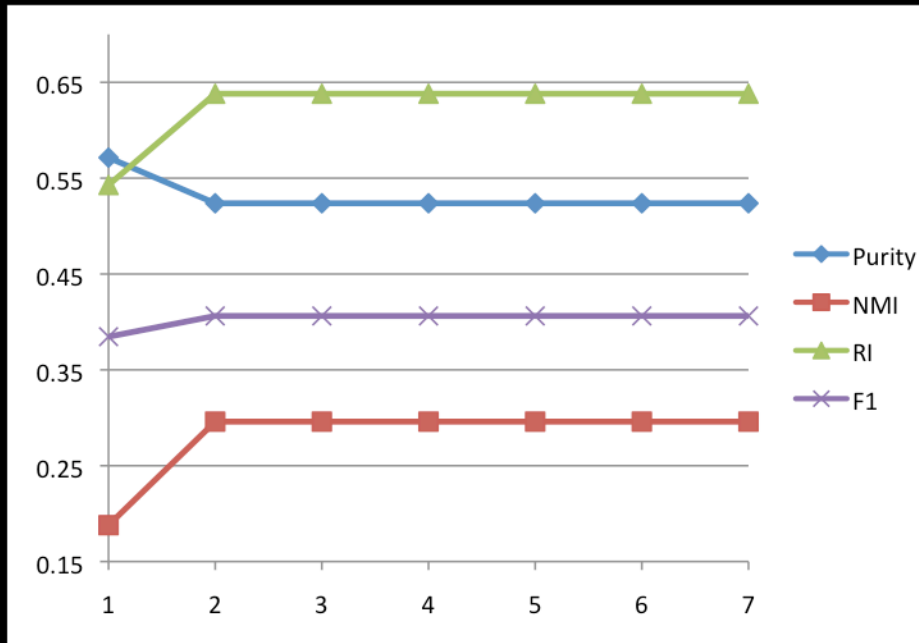
Metrics Example

ML



Metrics Example

ML



ML Questions?

- Questions???

ML Presentations

- Wed, Nov 4: Hansu Gu
- Volunteer
 - Mon, Nov 9:
 - Wed, Nov 11:

ML Projects

- Office hours: will stay until all questions are addressed
- Or make an appointment