Note Oct 4th

Hard Clustering — 1 pt → 1 cluster

Soft Clustering — helps w/ overlapping clustering clusters, to give each point a probability of being in a given cluster

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{n} \sum_{j=1}^{k} P(c_j) P(x_i \mid c_j)$$ Note: taking log-transformation doesn't change

$$\downarrow$$

$$d(\theta^*) = \sum_{i=1}^{n} \log \sum_{j=1}^{k} P(c_j) P(x_i \mid c_j)$$

$$\downarrow$$

solve: $\frac{\partial}{\partial \theta_{ML}} d(\theta) = 0$ | $\frac{d}{d\Sigma} L(\theta) = 0$

$\Rightarrow$ esimate: $\hat{\mu}_j = \frac{\sum_{i=1}^{n} P(c_j \mid x_i) x_i}{\sum_{i=1}^{n} P(c_j \mid x_i)}$ | $\hat{\Sigma}_j = \frac{\sum_{i=1}^{n} P(c_j \mid x_i)(x_i - \hat{\mu}_j)^T (x_i - \hat{\mu}_i)}{\sum_{i=1}^{n} P(c_j \mid x_i)}$ | $\hat{P}(c_j) = \frac{1}{n} \sum_{i=1}^{n} P(c_j \mid x_i)$

Bayes rule: $P(c_j \mid x_i) = \frac{P(x_i \mid c_j) \, P(c_j)}{\sum_{i=1} P(c_i) \, P(x_i \mid c_j)}$

Clustering Aggregation: Goal: ① Compare clusterings

② Combine info from multi clusterings to create a new clustering

Disagreement Distance. ↓

$$D(P, C) = \sum_{x,y} \mathbb{I}_{P,C}(x, y)$$

Note On 6th.

SVD : Singular value Decomposition:

- approx A w/ smaller B easier to store
- Dimensionality Reduction / Feature Extraction
- Anomaly Detection & Denoising

Linear Algebra review: Linear independent : Cannot represent any col as linear combo of other vectors in a set iff $A=[v_1, \cdots v_n]$

Determinant : $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ => $\det = ad - bc$

3×3 => $\det = a \cdot \det\begin{pmatrix} e & f \\ h & i \end{pmatrix} - b \cdot \det\begin{pmatrix} d & f \\ g & i \end{pmatrix} + c \cdot \det\begin{pmatrix} d & e \\ g & h \end{pmatrix}$

(Note: $m > n$ vectors in n-dim cannot be linearly independent.

Basis : of a vector space is a linearly independent subset of v that spans V.

Rank: max number of linearly independent cols of A. -> Full Rank iff $\text{rank}(A) = \min(m,n)$

Approximation: In practice datasets, there are a lot of redundant data.

$A_{(n\times m)} = U_{(n\times k)} \cdot V_{(k\times m)}$

$\cdot\ d(A, A^{(k)})$ is small

- Forbenius Distance: $\|A - B\| = \sqrt{\sum_{ij} (a_{ij} - b_{ij})^2}$

when $k < \text{rank}(A)$ $\quad A^{(k)} = \underset{\{B \mid \text{Rank}(B) = k\}}{\arg\min} \ d_F(A, B)$

SVD of Rank r $\quad A_{(n\times m)} = \underset{(n\times r)}{U}\ \underset{(r\times r)}{\Sigma}\ \underset{(r\times m)}{V^T}$

Oct 12th / 13th Note.

classification: Given a training set where data is labeled w/ special attribute called class

Techniques:
- instance based
- decision tree
- Naive Bayes
- Support vector Machine (SVM)
- Neural Networks.

kNN: used k closest records to perform classification.

→ limits: { If k is too small, then it is sensitive to noise point + over fitting.
If k is too big, then neighborhood may include points from other classes.

→ pros { simple to understand why a given unseen record was given a particular class.
Adapts to new examples.

→ cons { expensive to classify points.
kNN can be problematic in high dimensions.

Decision Tree

Algorithm: Need specify terminating conditions: → Let D be set of training rec that can reach node t.

→ Splitting based on normal attributes: - multiway split → use as many partitions as values.
- Binary split → categorized to binary.

Oct 18th / 20th

Recommender System: (user, user) similarity measure.
(item, item) similarity measure

pros → no training intuitive [Diag].
cons → users rate differently rating changing over time.

Content Based: feature to movie similarity.
Work: If we had user-to-feature + feature to movie = user to movie

$$p^{(j)} = \underset{p^{(j)}}{\operatorname{argmin}} \frac{1}{n_m(j)} \sum_{i:r(i,j)=1} (p^T q^{(i)} - r_{ij})^2 + \lambda \|p^{(j)}\|^2$$

penalty on the size of the param p.

Collaborative Feature

can we Learn P, Q. $R = PQ$ ✓ Learn using SVD as it is sparse.

$$\min_{p,q} \sum_{(i,j)\in R} (r_{ij} - p_i^T q_j)^2 + \lambda (\|p_i\|_F^2 + \|q_j\|_F^2)$$

Steps: 1. Start w/ random Q
2. Get P
3. Improve Q.
4. Repeat 2 & 3

Least Square: $\beta_{LS} = \underset{\beta}{\operatorname{argmin}} \sum_i d(h_\beta(x_i), y_i) = \underset{\beta}{\operatorname{argmin}} \|\vec{y} - \beta X\|_2^2 = \beta_{LS}$.

Maximum Likihood: Since $\varepsilon \sim N(0, \sigma^2)$ and $y = X\beta + \varepsilon$, then $y \sim N(X\beta, \sigma^2)$, $\beta_{LS} = (X^T X)^{-1} X^T y$

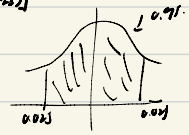
Oct 27th. Note.

Evaluate Regression Model: $y_i \sim N(5, 25)$ for $1 \le i \le 100$ and $y_i = \mu + \epsilon$ where $\epsilon \sim N(0, 25)$

Then, the LSS for μ (μ_{LS} = sample mean) $\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} y_i$.

$CL = [\hat{y} - 1.96(SE(\mu_{LS}),\ \hat{y} + 1.96(SE(\mu_{LS})]$

z-values: $\int_{-z}^{z} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} dx = 0.95$



QQ plot: need to check our variance is distributed in the same way a variable following our target distribution

Constant variance: we assume that our noise has constant variance

we can plot our fitted values against our residuals (noise estimates)