CSE 5334 Final Preparation

* Information Retrival Systems
  - Word: a unit separated by whitespace or punctuation
  - Term: a meaningful unit
  - Token: a unit after tokenization
  - Type: a disjoinset of tokens from documents

Term = Type ⊂ Word
Token

* Normalization
* Tokenization
  - numbers
  - stopwords
  - Type construction: ① Lemmatization ② Stemming
    ex. Porter, Lovins, Paice

A few rules of Porter Stemmer

Rule			Example		
sses	→	ss	caresses	→	caress
ies	→	i	ponies	→	poni
ss	→	ss	caress	→	caress
s	→		cats	→	cat

* Ranked Retrieval
  ① Jaccard Coefficient: from non-empty sets A, B

$$Jaccard(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

due to the Set feature, 1) X considers the freq of terms
2) X weights the important words.
3) X implies the length of documents

② tf-matching-score
  - Term Frequency tf = # of terms in documents
    - log weighted: $\begin{cases} 1 + \log_{10}(tf) & : tf > 0 \\ 0 & : \text{otherwise} \end{cases}$

⇒ $\sum 1 + \log_{10} tf$

③ TF-IDF Matrix & Cosine Similarity
  - Inversed Document Frequency $idf_t = \log_{10} \frac{N}{df_t}$
    - N: # of documents
    - df_t: # of documents which contains term

* $tf\text{-}idf = (1 + \log tf) \cdot \log \frac{N}{df_t}$

  - TF-IDF Matrix

term \ doc	d_1	d_2 ⋯
t_1	w_{11}	w_{21}
t_2	w_{12}	w_{22}

⇒ $d_1 = [w_{11}, w_{12}]$
$d_2 = [w_{21}, w_{22}]$

  - Cosine Similarity: $\frac{\vec{q} \cdot \vec{d}}{\|\vec{q}\| \times \|\vec{d}\|}$

ex. $l_{nc}.l_{tn}$ → l_{nc} for doc. l_{tn} for query

Table 5. Components of tf-idf

Term Frequency tf		Document Frequency df		Normalization	
n (natural)	$tf_{t,d}$	n (no)	1	n (none)	1
l (logarithmic)	$1 + \log(tf_{t,d})$	t (idf)	$\log \frac{N}{df_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2 + w_2^2 + \cdots + w_M^2}}$
a (augmented)	$0.5 + 0.5 \times \frac{tf_{t,d}}{\max_{t \in d} tf_{t,d}}$	p (prob idf)	$max(0, \log \frac{N - df_t}{df_t})$	u (pivoted unique)	$\frac{1}{u}$
b (boolean)	1, if $tf_{t,d} > 0$, otherwise 0			b (byte size)	$\frac{1}{CharLength^{\alpha}}, \alpha < 1$
L (log average)	$\frac{1 + \log tf_{t,d}}{1 + \log(ave_{t \in d} tf_{t,d})}$				

* Data Mining
  - 4Vs: Volume, Velocity, Variety, Veracity (Veracity ↳ quality)
  - Structed / Semi-Structed / Unstructed data
  - Nominal — eye color, student ID
    Ordinal — student grades, satisfaction level, height
    Interval — Celsius & Fahrenheit Temperature, dates on calendar
    Ratio — kelvin Temperature

* Nom. ⊂ Ord. ⊂ Int ⊂ Rat.

* Curse of dimensionality
  ↳ the circumstance that the complexity and sparsity of the data increase as the dim. of the data increases.

* Data Preprocessing
  - aggregation, sampling, dimensionality reduction, feature subset selection, feature creation, discretization and binarization, attribute transformation
    - random sampling
    - w/o replacement sampling
    - w/ replacement sampling
    - Stratified sampling

* Similarity and Distance

Attribute	Similarity	Dissimilarity
Nominal	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$
Ordinal	$s = 1 - \frac{\lvert p-q \rvert}{n-1}$ $(s = 1 - d)$ (where n is the number of values, mapping the values to integers 0 to n-1)	$d = \frac{\lvert p-q \rvert}{n-1}$
Interval or Ratio	$s = -d, s = \frac{1}{1+d}$	$d = \lvert p - q \rvert$

  - Distance

$$dist = \left(\sum_{i=1}^{n} |p_i - q_i|^r \right)^{1/r} \begin{cases} r = 1, & \text{Manhattan distance, L1 norm} \\ r = 2, & \text{Euclidean distance, L2 norm} \\ r = \infty, & \text{Supremum (Chebyshev) distance, } L_{max} \text{ norm, } L_{\infty} \text{ norm} \end{cases}$$

  - Similarity

$$M_{pq} = \begin{cases} M_{01} & p = 0, q = 1 \\ M_{10} & p = 1, q = 0 \\ M_{11} & p = 1, q = 1 \\ M_{00} & p = 0, q = 0 \end{cases}$$

(The number of attributes for each case p and q)

	SMC	Jaccard Coefficient
	$\frac{M_{11} + M_{00}}{M_{11} + M_{00} + M_{10} + M_{01}}$	$\frac{M_{11}}{M_{11} + M_{10} + M_{01}}$
ex. $p = 1000000000, q = 0000001001$	$\frac{0+7}{0+7+1+2} = \frac{7}{10}$	$\frac{0}{0+1+2} = 0$

  - if M_{00} has an important meaning → SMC ☺
  - if M_{00} is meaningless → Jaccard ☺

$$\text{Pearson Correlation Coefficient} = \frac{\text{covariance}_{x,y}}{\sigma_x \sigma_y} = \frac{\sum_{i=1}^{n}(p_i - \bar{p})(q_i - \bar{q})}{\sqrt{\sum_{i=1}^{n}(p_i - \bar{p})^2} \sqrt{\sum_{i=1}^{n}(q_i - \bar{q})^2}}$$

↳ $\bar{p} = \frac{1}{n}\sum p_i$ $\bar{q} = \frac{1}{n}\sum q_i$

  - Overall Similarity (w/ many types of attributes)

1. Define an indicator variable σ_k for the k-th attribute as follows:

$$\sigma_k = \begin{cases} 0 & \text{(if the } k^{th} \text{ attribute is a binary asymmetric attribute AND both objects have a 0 value or (if one of the object has a missing value for the } k^{th} \text{ attribute)} \\ 1 & \text{otherwise} \end{cases}$$

2. With the similarity of the k^{th} attribute s_k, the overall similarity between two objects is calculated as:

$$s(p,q) = \frac{\sum_{k=1}^{n} s_k \cdot \sigma_k}{\sum_{k=1}^{n} \sigma_k}$$

* Decision Tree
  - Hunt's

```
Start from a tree with a single root node containing all the training data.

Recursive:
  1. Check if the node is homogeneous (pure).
     - If true, make the node a leaf node and label it with the class. END the branch.
     - If not, continue to the next step.
  2. Check if the node is empty.
     - If true, make the node a leaf node. END the branch.
     - If not, continue to the next step.
  3. Check whether the node has conflicting data, a same label with different values.
     - If true, mark the node as a leaf node. END the branch.
     - If not, continue to the next step.
  4. Split the node into child nodes based on the attribute.
     - During the split, the algorithm calculates the impurity of the child nodes
       using the 1) Gini index, 2) entropy, or 3) misclassification error.
     - After splitting, the Gain is recalculated to renew the tree state.
       choose an attribute which maximizes a gain
Terminate:
  1. If all nodes become leaf nodes during the recursive process.
  2. If the split does not show certain improvement than beforehanded-set threshold, regarding the impuri
```

  - Impurity Metric
    ① Misclassification Error: $1 - \max(p_i)$
    ↳ the most correct value

ex. 3:3:3 → $\frac{3}{9}$ ⇒ M.E. = $\frac{6}{9}$
2:3:5 → $\frac{5}{10}$ ⇒ M.E. = $\frac{5}{10}$

② $\text{Gini} = 1 - \sum_{i=1}^{c} p_i^2$ $\begin{cases} c = \text{the number of classes} \\ p_i = \text{the probability of selecting an item of class } i \text{ in the node} \end{cases}$

cf. minimum = 0 maximum = $1 - \frac{1}{c}$

$$\text{Gain} = \text{Gini(parent)} - \sum_{\text{child} \in \text{children}} \frac{\text{the number of data in the child}}{\text{the number of data in the parent}} \times \text{Gini(chil}$$

1) $\text{Entropy}(t) = -\sum_j P(j|t)\log_2 P(j|t)$ — Minimum : 0 — Maximum : $\log_2 C$

Information Gain

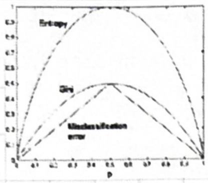
$$\text{Gain}_{\text{split}} = \text{Entropy(parent)} - \sum_{\text{child} \in \text{children}} \frac{\text{the number of data in the child}}{\text{the number of data in the parent}} \times \text{Entropy(child)}$$

Split Info

$$\text{Split Info} = -\sum_{\text{child} \in \text{children}} \frac{\text{the number of data in the child}}{\text{the number of data in the parent}} \times \log_2 \frac{\text{the number of data in the child}}{\text{the number of data in the parent}}$$

Gain Ratio

$$\text{Gain Ratio} = \frac{\text{Gain}_{\text{split}}}{\text{Split Info}}$$



- Entropy is able to represent the maximum impurity and its derivatives along with prob. among metrics

- Pre-pruning : ① Stop when the tree reaches a certain depth
  ② Stop when the # of instances in a node is less than a certain threshold
  ③ Stop if all instances in a node belong to the same class.
  ④ Stop if the split does not improve impurity like Gini, Information Gain
- Post-pruning : 1) Pessimistic Error
  2) Optimistic Error
  3) Reduced Error
  └ With validation data, prune or not.

optimistic Training Error (before splitting) = $\frac{10}{30}$

Pessimistic Error (before splitting) = $\frac{10 + (0.5 \times 1)}{30}$ (# of Nodes)

optimistic Training Error (after splitting) = $\frac{4+3+1+1}{30} = \frac{9}{30}$

Pessimistic Error (after splitting) = $\frac{9 + (0.5 \times 4)}{30}$

* Naive Bayes Classifier

Naive Bayes Classifier : $P(C \mid A_1, A_2, \ldots, A_n)$

$$\begin{cases} P(C \mid A_1, A_2, \ldots, A_n) = \frac{P(A_1, A_2, \ldots, A_n \mid C)P(C)}{P(A_1, A_2, \ldots, A_n)} \\ P(A_1, A_2, \ldots, A_n \mid C) = P(A_1 \mid C)P(A_2 \mid C)\ldots P(A_n \mid C) \end{cases}$$

→ X = (Refund=No, Status=Married, Income=120K)

$P(X \mid \text{Class=No}) = P(\text{Refund=No} \mid \text{Class=No}) \times P(\text{Status=Married} \mid \text{Class=No}) \times P(\text{Income=120K} \mid \text{Class=No}) = \frac{4}{7} \cdot \frac{4}{7} \cdot 0.0072 = 0.0024$

$P(X \mid \text{Class=Yes}) = P(\text{Refund=No} \mid \text{Class=Yes}) \times P(\text{Status=Married} \mid \text{Class=Yes}) \times P(\text{Income=120K} \mid \text{Class=Yes}) = 1 \cdot 0 \cdot (1.2 \times 10^{-9}) = 0$

$P(X \mid \text{Class=No}) \cdot P(\text{Class=No}) = 0.0024 \cdot 0.7 = 0.0017$

$P(X \mid \text{Class=Yes}) \cdot P(\text{Class=Yes}) = 0 \cdot 0.3 = 0$

$$P(C_j) = \frac{\text{Number of data in class } C_j}{\text{Total number of data}}$$

e.g. $P(\text{No}) = \frac{7}{10} = 0.7, P(\text{Yes}) = \frac{3}{10} = 0.3$

Discrete Attributes: $P(A_i \mid C_k) = \frac{\text{Number of instance having attribute } A_i}{\text{Number of Class } C_k}$

e.g. $P(\text{Status=Married} \mid \text{No}) = \frac{4}{7}$

$P(\text{Refund=Yes} \mid \text{Yes}) = 0$

Continuous Attributes: $P(A_i \mid C_k) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$ → $\sigma = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu)^2$

e.g. $P(\text{Income=120} \mid \text{No}) = \frac{1}{\sqrt{2\pi(54.54)}} e^{-\frac{(120-110)^2}{2(2975)}}$ = 0.0072

- Prevent P=0 (only for Discrete Attributes)

Laplace $P(A_i \mid C) = \frac{\text{Number of instance having attribute } A_i + 1}{\text{Number of Class } C_k + N}$ N: # of classes

m-estimate $P(A_i \mid C) = \frac{\text{Number of instance having attribute } A_i + m \cdot P(A_i)}{\text{Number of Class } C_k + m}$ m: Smoothing Parameters

* KNN

- to avoid ties — 1) odd K value
  2) weight factor $w = \frac{1}{\text{distance}^2}$
- when k↓ → sensitive to noise
  k↑ → acc. decrease.

* PEBLS

· $\text{Distance}(V_1, V_2) = \sum_{i \in C} \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|$ ⇒

class: Yes ¦ No

$\text{Distance(Single, Married)} = \left|\frac{2}{4} - \frac{0}{4}\right| + \left|\frac{2}{4} - \frac{4}{4}\right| = 1$

$\text{Distance(Single, Divorced)} = \left|\frac{2}{4} - \frac{1}{2}\right| + \left|\frac{2}{4} - \frac{1}{2}\right| = 0$

$\text{Distance(Married, Divorced)} = \left|\frac{0}{4} - \frac{1}{2}\right| + \left|\frac{4}{4} - \frac{1}{2}\right| = 1$

$\text{Distance(Refund=Yes, Refund=No)} = \left|\frac{0}{3} - \frac{3}{7}\right| + \left|\frac{3}{3} - \frac{4}{7}\right| = \frac{6}{7}$

· Weighting Method

$$\delta(X, Y) = W_X W_Y \sum_{i=1}^{d} \text{Distance}(X_i, Y_i)^2$$

Where $W_X = \frac{\text{The number of times that X is used for prediction}}{\text{The number of times that X is used to predict correctly}}$

* SVM

*decision boundary : $\vec{w} \cdot \vec{x} + b = 0$ [illegible] ±1

*support vector : a point which on ±1

Class 1 : 1, class 2 : -1 로 mapping

- when x_i, y_i are given, Find $y_i(w^T x_i + b) \geq 1$
- non-linearly separable data : slack variable
  non-linear decision boundary : Kernel trick

*Margin : $\frac{2}{\|\vec{w}\|}$

$L(w) = \frac{\|w\|^2}{2}$ minimize

* Evaluation

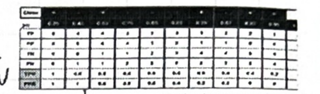
- Underfitting : train & test Err.↑ ⊕ a model is too simple
  Overfitting : train ↓ test ↑ (Err.) ⊕ a model is too complex
- Occam's Razor $\text{MDL}: L(M, D) = L(M) + L(D \mid M)$ { $L(M)$ is the length of the model; $L(D|M)$ is the length of the data given the model

- Missing Values
  Train:
  1) when computing Impurity
     - exclude in counting child nodes
     - include in counting parent node
  2) when distributing instances
     - (3T, 7F)
       (0T, 3F) (2T, 4F) ⇒ (3/9 T, 3F) (6/9 T, 4F)

  Test - 3) From trained tree, follow the most # of value in missing attribute

* Metrics

- Confusion Matrix: Actual / Pred. — TP | FN / FP | TN ; precision ↔ recall ; $f: \frac{2rp}{r+p}$
- ROC Curve



$TPR = \frac{TP}{TP+FN}$ $FPR = \frac{FP}{FP+TN}$

draw the ROC curve by moving $\frac{1}{\text{\# of positive labeled samples}}$ or $\frac{1}{\text{\# of negative labeled samples}}$ to the bottom or left fo[r] threshold respectively on the each point of the ROC curve from the (1, 1) data point with the lowest thresh[old]

- Confidence Interval
  Model 1 in dataset A (size n_1, error e_1)
  " 2 " B (" n_2, " e_2) ⇒ $d = e_1 - e_2$, $\sigma^2 = \sum \frac{e_i(1-e_i)}{n_i}$, $d_t = d \pm Z_{\alpha/2} \times \sigma$

* Sampling & Validation Techniques

① Holdout ② Random Subsampling (K Holdout) ③ Cross Valid. └ k=N : Leave-one-out
④ Stratified Sampling ⑤ Bootstrap

* Clustering

- Goal : Minimize Intra-Cluster distance
  Maximize Inter-Cluster distance
- Partitional Clustering
  $P = K! / K^K$
  - Bisecting K-means : Divide the ↑SSE cluster into 2
  - $SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$

- Hierarchical Clustering — Distance 관점에서
  · MIN : can handle various shape, sensitive to noise, outliers
  · MAX : less sensitive to noise, outliers, able to break large clusters.
  · Agglomerative — 합칠 때는 table의 가장 작은 값, Table 갱신할 때 min 또는 max.
  · Divisive — Build MST → ↑Dist. 부터 잘라 cluster 분리

* Association Rule Mining

Rule: {Milk, Diaper} → {Beer}

$$\begin{cases} \text{Support } s = \frac{\sigma(\{\text{Milk, Diaper, Beer}\})}{\text{Total number of transaction}} \\ \text{Confidence } c = \frac{\sigma(\{\text{Milk, Diaper, Beer}\})}{\sigma(\{\text{Milk, Diaper}\})} \end{cases}$$

- # of Possible Association Rule : $3^n - 2^{n+1} + 1$
- # of " Itemsets : 2^n
- Apriori Algorithm
  1) Frequent Itemset Generation ← (Apriori Principle)

① 1-itemsets

item	support
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

2-itemsets

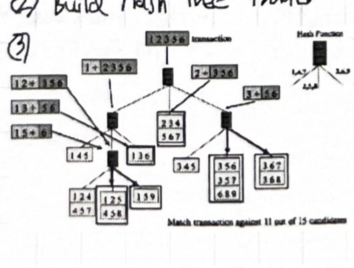
item	support
Bread, Milk	3
Bread, Beer	2
Bread, Diaper	3
Milk, Beer	2
Milk, Diaper	3
Beer, Diaper	3

3-itemsets

item	support
Bread, Milk, Diaper	3
Bread, Diaper, Beer	3
Milk, Diaper, Beer	3

② Build Hash Tree from ①

③



2) Rule Generation

$\text{Confidence}(A, B, C \to D) \geq \text{Confidence}(A, B \to C\ldots)$
$\geq \text{Confidence}(A \to B, C\ldots)$