Data Exploration

1. Types of Raw Data Resources:
Record, Graph and Network, Ordered, Spatial, image and multimedia
2. Types of Attributes:
Nominal, Ordinal, Interval, Ratio
3. Proximity:
Similarity: [0,1]
Dissimilarity: [0,inf)
4. Proximity for Binary

- A contingency table for binary data

		Object j		
		1	0	sum
Object i	1	q	r	$q+r$
	0	s	t	$s+t$
	sum	$q+s$	$r+t$	p

q: number of attributes both i and j have 1.
s: number of attributes i has 0 and j has 1.
r: number of attributes i has 1 and j has 0.
t: number of attributes both i and j have 0.

- Distance measure for asymmetric binary variables (e.g., event or not):

$$d(i,j) = \frac{r+s}{q+r+s}$$

- Distance measure for symmetric binary variables (e.g., binary gender):

$$d(i,j) = \frac{r+s}{q+r+s+t}$$

- Jaccard coefficient (*similarity* measure for *asymmetric* binary variables):

$$sim_{Jaccard}(i,j) = \frac{q}{q+r+s}$$

- Simple Matching (similarity measure for symmetric binary variables):

$$SMC(i,j) = \frac{q+t}{q+r+s+t}$$

5. Proximity for Nominal
Method1: Simple Matching SMC(i,j)=#attributes match/#all attributes
Method2: Convert to binary
6. Distance on numerical:
Minkowski Distance:

$$d(x,y) = \sqrt[h]{|x_1 - y_1|^h + |x_2 - y_2|^h + \cdots + |x_n - y_n|^h}$$

- $x = (x_1, x_2, \ldots, x_n)$ and $y = (y_1, y_2, \ldots, y_n)$ are two p-dimensional data objects, and h is the order
- The distance so defined is also called ℓ_h-norm

Properties

- d(x, y) > 0 if x ≠ y, and d(x, x) = 0 (Positive definiteness)
- d(x, y) = d(y, x) (Symmetry)
- d(x, y) ≤ d(x, u) + d(u, y) (Triangle Inequality)

H=1: Manhattan distance
H=2: Euclidean distance
H=inf: Supremum distance
7. Similarity on numerical:
Cosine similarity:
cos(x, y) = (x • y) /(||x|| ||y||)
8. Correlation:
Limited to linear relationship
Corr = covariance(x,y)/sd(x)sd(y)

$$\text{covariance}(\mathbf{x},\mathbf{y}) = s_{xy} = \frac{1}{n-1}\sum_{k=1}^{n}(x_k - \overline{x})(y_k - \overline{y})$$

$$\text{standard\_deviation}(\mathbf{x}) = s_x = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \overline{x})^2}$$

$$\text{standard\_deviation}(\mathbf{y}) = s_y = \sqrt{\frac{1}{n-1}\sum_{k=1}^{n}(y_k - \overline{y})^2}$$

-1: perfect negative relationship
+1: perfect positive relationship
0: no linear relationship

Property	Cosine	Correlation	Euclidean Distance
Invariant to scaling (multiplication)	Yes	Yes	No
Invariant to translation (addition)	No	Yes	Yes

9. Proximity for ordinal -> numerical:

$$z_i = \frac{r_i - 1}{M_i - 1}\, r_i \in \{1, \cdots, M_i\}$$

10. Proximity on mixed type

$$d(x,y) = \frac{\sum_{i=1}^{n} \delta_{x,y}^{(i)} \cdot d_{x,y}^{(i)}}{\sum_{i=1}^{n} \delta_{x,y}^{(i)}},$$ ($\delta_{x,y}^{(i)}$ is the weight of each attribute i)

If *Attribute* i is binary or nominal:
$d_{x,y}^{(i)} = 0$ if $x_i = y_i$, or $d_{x,y}^{(i)} = 1$ otherwise
If *Attribute* i is numeric: use the normalized distance (normalized to [0,1])
If *Attribute* i is ordinal, transfer it to numerical attribute as introduced in the previous slide.

11. Skewed Data
Right(positive) skew: mode<median<mean
Left(negative) skew: mean<median<mode
Empirical relation among the three m's:

$$mean - mode = 3 \times (mean - median)$$

Data Preprocessing

1. Measure for data quality:
Accuracy, Completeness, Consistency, Timeliness, Believability, Interpretability
2. Major tasks:
Data cleaning, integration, reduction, transformation and discretization
3. Data cleaning:
Incomplete, noisy, inconsistent, intentional
4. Handle missing data:
Ignore tuples, Fill manually, Fill with a global constant, the attribute mean, the attribute mean for all samples belonging to the same class, inference-based such as Bayesian formula or decision tree
5. Data integration:
Entity identification problem: Identify real world entities from multiple data sources
6. Handle redundant data:
Due to object identification or derivable data,
detected by correlation analysis and covariance analysis
7. Correlation Analysis (Nominal Data)
Χ² (chi-square) test

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

The larger the Χ² value, the more likely the variables are related
The cells that contribute the most to the Χ² value are those whose actual count is very different from the expected count
8. Correlation Analysis (Numerical Data)
Correlation coefficient (also called Pearson's product moment coefficient)

$$r_{A,B} = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{(n-1)\sigma_A \sigma_B} = \frac{\sum_{i=1}^{n}(a_i b_i) - n\bar{A}\bar{B}}{(n-1)\sigma_A \sigma_B}$$

where n is the number of tuples, $\bar{A}$ and $\bar{B}$ are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\Sigma(a_i b_i)$ is the sum of the AB cross-product.

If $r_{A,B} > 0$, A and B are positively correlated (A's values increase as B's). The higher, the stronger correlation.
$r_{A,B} = 0$: independent; $r_{AB} < 0$: negatively correlated
Correlation Calculation:

$$a'_k = (a_k - mean(A)) / std(A)$$

$$b'_k = (b_k - mean(B)) / std(B)$$

$$correlation(A,B) = A' \bullet B'$$

9. Covariance (Numerical Data)
Covariance is similar to correlation

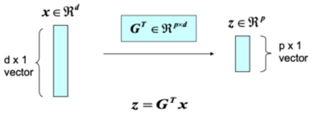
$$Cov(A,B) = E((A - \bar{A})(B - \bar{B})) = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{n}$$

Correlation coefficient: $r_{A,B} = \frac{Cov(A,B)}{\sigma_A \sigma_B}$

Positive covariance: If $Cov_{A,B} > 0$, then A and B both tend to be larger than their expected values.
Negative covariance: If $Cov_{A,B} < 0$ then if A is larger than its expected value, B is likely to be smaller than its expected value.
Independence: $Cov_{A,B} = 0$ but the converse is not true:

- Some pairs of random variables may have a covariance of 0 but are not independent. Only under some additional assumptions (e.g., the data follow multivariate normal distributions) does a covariance of 0 imply independence

Simplified: $Cov(A,B) = E(A \cdot B) - \bar{A}\bar{B}$
11. Data Reduction
Dimensionality Reduction:
When dim increase, data is increasingly sparse, density and distance less meaningful, possible combinations of subspaces grow exponentially
12. Principal Component Analysis



$$z = G^T x$$

G is a d x p linear transformation matrix
Given: a data set $\{x_1, x_2, \ldots, x_N\}$, where $x_i \in \Re^d$
Goal: find a vector g that transforms each data point x_i to a scalar z_i such that var[z] is maximized

$$\max_g g^T S g \quad \text{s.t.} \quad g^T g = 1$$

$$L = g^T S g - \lambda(g^T g - 1)$$

$$\frac{\partial L}{\partial g} = 2Sg - 2\lambda g = 0$$

$$\Rightarrow Sg = \lambda g$$ (eigenvalue equation)

The first principal component g is the eigenvector that corresponds to the largest eigenvalue of **S**

In general, $\text{var}[z_k] = g_k^T S g_k = \lambda_k$
the kth principal component corresponds to the eigenvector of the kth largest eigenvalue of S
13. Attribute Creation
Attribute extraction, Mapping data to new space, Attribute construction
14. Numerosity Reduction
Parametric method: Assume model
Linear regression, multiple regression, log-linear model
Non-parametric: Do not assume model
histograms, clustering, sampling
15. Regression Analysis:
Dependent (response, measurement) vs independent (explanatory, predictors)
Least Squares Method
16. Histogram Analysis:
Divide data into buckets and store average (sum) for each bucket
Partitioning rules: Equal-width: equal bucket range / Equal-frequency (or equal-depth)
17. Clustering:
Partition data set into clusters based on similarity, and store cluster representation (e.g., centroid and diameter) only
18. Sampling:
obtaining a small sample s to represent the whole data set N
Simple random sampling

- There is an equal probability of selecting any particular item

Sampling without replacement

- Once an object is selected, it is removed from the population

Sampling with replacement

- A selected object is not removed from the population

Stratified sampling:

- Partition the data set, and draw samples from each partition (proportionally, i.e., approximately the same percentage of the data)
- Used in conjunction with skewed data

19. Data Compression
Original Data ->lossless<- Compressed data ->lossy->original data approximated
20. Data Transformation
Normalization
Min-max normalization: to [new_min$_A$, new_max$_A$]

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,000 is mapped to $\frac{73{,}600 - 12{,}000}{98{,}000 - 12{,}000}(1.0 - 0) + 0 = 0.716$

Z-score normalization (μ: mean, σ: standard deviation):

$$v' = \frac{v - \mu_A}{\sigma_A}$$

- Ex. Let μ = 54,000, σ = 16,000. Then $\frac{73{,}600 - 54{,}000}{16{,}000} = 1.225$

Normalization by decimal scaling

$$v' = \frac{v}{10^j}$$ Where j is the smallest integer such that Max(|v'|) < 1

21. Data Discretization Methods
Binning

- Top-down split, unsupervised

Histogram analysis

- Top-down split, unsupervised

Clustering analysis (unsupervised, top-down split or bottom-up merge)
Decision-tree analysis (supervised, top-down split)
Correlation (e.g., χ^2) analysis (unsupervised, bottom-up merge)
22. Binning Methods for Data Smoothing
Equal-width: max-min/#bin (When you need consistent interval sizes.)
Equal frequency: # inside bin the same (When balanced bin sizes are important)
Clustering-based:Use cluster algo to form bin (when the data has natural groupings)

Frequent Pattern Mining

1. Relative support of itemset: fraction of transaction containing the itemset
2. Relative support of rule:

$$\text{Supp}(X \rightarrow Y) = \frac{\text{Number of transactions containing both X and Y}}{\text{Total Number of Transactions}}$$

Confidence of rule:

$$\text{Conf}(X \rightarrow Y) = \frac{\text{Number of transactions containing both X and Y}}{\text{Number of Transactions containing X}}$$

Strong rule: with relative support and confidence higher than their threshold
3. Association Rule Mining Task: All strong rules
Step1: Frequent Itemset Generation
Step2: Rule generation
4. Closed itemset: none of its immediate supersets has the same support as the itemset
Maximal itemset: none of its immediate supersets is frequent
5. Apriori Principle:

$$\forall X, Y : (X \subseteq Y) \Rightarrow supp(X) \geq supp(Y)$$

Method:

- Let k=1
- Generate frequent itemsets of length 1
- Repeat until no new frequent itemsets are identified
  - Generate length (k+1) candidate itemsets from length k frequent itemsets
  - **Prune** candidate itemsets containing subsets of length k that are **infrequent**
  - Count the support of each candidate by scanning the transaction table.
  - Eliminate candidates that are infrequent, leaving only those that are frequent

Limitation: BFS, generate huge candidates, repetitive scan of supports
6. Rule generation

Given a frequent itemset L, find all non-empty subsets f ⊂ L such that f → L – f satisfies the minimum confidence requirement

If |L| = k, then there are 2^k – 2 candidate association rules (ignoring L → ∅ and ∅ → L)

Relation:

suppose {A,B,C,D} is a frequent 4-itemset, so:

conf(ABC → D) ≥ conf(AB → CD) ≥ conf(A → BCD)

7. Correlations (Lift)

$\text{Lift}(X \to Y) = \frac{p(X \cup Y)}{p(X)p(Y)}$

Lift can be smaller or larger than 1, so we need to contrast two lifts: $\text{Lift}(X \to Y)$ v.s. $\text{Lift}(X \to \neg Y)$ to see how strong X → Y is.

8. FP growth

DFS, avoid generate huge candidate, avoid repetitive scan

Construct FP tree:

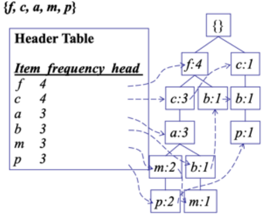
1. Scan DB once, find frequent 1-itemset (single item pattern)
2. Sort frequent items in frequency descending order, f-list

TID	*Items bought*	*(ordered) frequent items*
100	{f, a, c, d, g, i, m, p}	{f, c, a, m, p}
200	{a, b, c, f, l, m, o}	{f, c, a, b, m}
300	{b, f, h, j, o, w}	{f, b}
400	{b, c, k, s, p}	{c, b, p}
500	{a, f, c, e, l, p, m, n}	{f, c, a, m, p}

min_support = 3

3. Insert transactions into the FP-Tree

F-list = f-c-a-b-m-p



Advantages of the Pattern Growth Approach

- Divide-and-conquer:
  - Decompose both the mining task and DB according to the frequent patterns obtained so far
  - Lead to focused search of smaller databases
- Other factors
  - No candidate generation, no candidate test
  - Compressed database: FP-tree structure
  - No repeated scan of entire database
  - Basic ops: counting local freq items and building sub FP-tree, no pattern search and matching
- A good open-source implementation and refinement of FPGrowth

Benefits of the FP-tree Structure

- Completeness
  - Preserve complete information for frequent pattern mining
  - Never break a long pattern of any transaction
- Compactness
  - Reduce irrelevant info—infrequent items are gone
  - Items in frequency descending order: the more frequently occurring, the more likely to be shared
  - Never be larger than the original database (not count node-links and the *count* field)

9. Handling binary attributes:

Asymmetric can be converted to item occurrence

Symmetric can be convert to asymmetric then occurrence

10. Handling categorical attributes:

Some attributes have many values->aggregate over low-support values

Distribution highly skewed->discard the highly frequent items

11. Discretization-based: bin continuous variable in rule left part

Interval too wide:

May merge several disparate patterns

May lose some interesting patters

Interval too narrow:

Pattern is broken up into smaller patterns

Some windows my not meet support

Discretization: all possible intervals

Number of interval boundaries = k

Total number of Adjacent intervals:

$C_2^k = k(k-1)/2$

- Execution time
  - If the range is partitioned into k-1 intervals, there are O(k^2) new items
  - If an interval [a,b) is frequent, then all intervals that contains [a,b) must also be frequent
    - E.g.: if {Age ∈[21,25), Chat Online=Yes} is frequent, then {Age ∈[10,50), Chat Online=Yes} is also frequent
  - Improve efficiency:
    - Use maximum support to avoid intervals that are too wide

12. Statistics-based: rule right part use statistics

How to determine whether an association rule interesting?

- Compare the statistics for segment of population covered by the rule vs segment of population not covered by the rule:

  A ⇒ B: μ versus Ā ⇒ B: μ′

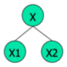
- Statistical hypothesis testing:
  - Null hypothesis: H0: μ′ = μ + Δ
  - Alternative hypothesis: H1: μ′ > μ + Δ
  - Z-test: Z has zero mean and variance 1 under null hypothesis

$$Z = \frac{\mu' - \mu - \Delta}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

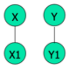
13. Multi-level association rules

How do support and confidence vary as we traverse the concept hierarchy?

- If X is the parent item for both X1 and X2, then supp(X) ≥ supp(X1) + supp(X2)



- If supp(X1 ∪ Y1) ≥ minsup,
  and X is parent of X1, Y is parent of Y1
  then supp(X ∪ Y1) ≥ minsup, supp(X1 ∪ Y) ≥ minsup
  supp(X ∪ Y) ≥ minsup



- If conf(X1 ⇒ Y1) ≥ minconf,
  then conf(X1 ⇒ Y) ≥ minconf

Because $\frac{supp(X1,Y)}{supp(X1)} \geq \frac{supp(X1,Y1)}{supp(X1)}$

14. Constraint-Based Frequent Pattern Mining

-Prune the pattern space:

-Anti-monotonic: If constraint c is violated, its further

-Monotonic: If c is satisfied, no need to check c again for its further mining.

-Succinct: we can explicitly and precisely determine if any itemset satisfies the constraint by examining if it contains some specific items.

-Convertible: c is not monotonic nor anti-monotonic nor succinct, but it can be converted into it if items in the transaction can be properly ordered

-Strongly convertible constraint: if both the following are satisfied:

The constraint is convertible anti-monotone w.r.t. item value descending order;

The constraint is convertible monotone w.r.t. item value ascending order

Classification

1. supervised learning: Regression vs Classification

2. Decision tree

The idea of Generate_decision_tree(D)

- If D contains records that belong to the same class, return leaf node with this class.
- Otherwise,
  - If no unused attribute, return leaf node with majority class.
  - Otherwise, **use an attribute to split the data** into smaller subsets: $D_1, D_2, \ldots D_k$. Call Generate_decision_tree(D_i) for each i.

Generate_decision_tree(D) is a recursive function

Advantages:

- Relatively inexpensive to construct
- Extremely fast at classifying unknown records
- Easy to interpret for small-sized trees
- Can easily handle redundant or irrelevant attributes

Disadvantages:

- Each decision boundary involves only a single attribute
- Interacting attributes (that can distinguish between classes together but not individually) may be passed
- The output is deterministic

3. Measure of node impurity

Maximum of $1 - 1/c$ when records are equally distributed among all classes

-Gini Index Minimum of 0 when all records belong to one class

$$Gini(D) = 1 - \sum_{i=0}^{c-1} {p_i}^2$$

D is the data of the current node; $\boldsymbol{p_i}$ is the probability of class $\boldsymbol{i}$ at current node; $\boldsymbol{c}$ is the total number of classes

$$Entropy(D) = -\sum_{i=0}^{c-1} p_i \log_2 p_i$$

$\boldsymbol{p_i}$ is the probability of class $\boldsymbol{i}$ at the current node, and $\boldsymbol{c}$ is the total number of classes

- Maximum of $\log_2 c$ when records are equally distributed among all classes

-Entropy
- Minimum of 0 when all records belong to one class

$$Error(D) = 1 - \max_{i \in \{0,\ldots,c-1\}} p_i$$

Maximum of $1 - 1/c$ when records are equally distributed among all classes

Minimum of 0 when all records belong to one class, implying the most interesting situation

-Classfication Error

4. Finding the best to split

Step 1: Compute impurity measure before splitting

Step 2: Compute impurity measure after splitting by an attribute:

- Compute impurity measure of each child node
- Weighted sum of all the child nodes' impurities.

Step 3: Compute the impurity change before and after splitting for this candidate attribute, and select the one with largest reduction.

5. Gain Ratio

$$Gain\ Ratio = \frac{Information\ Gain}{Split\ Info} \quad ,Split\ Info = -\sum_{j=1}^{k} \frac{|D_j|}{|D|} log_2 \frac{|D_j|}{|D|}$$

$|D|$: sample size of parent node ; $|D_j|$: sample size of child node j;

k: number of partitions.

- Adjusts Information Gain by the entropy of the partitioning ($Split\ Info$).
  - Higher entropy partitioning (large number of small partitions) is penalized!

6. Bayes Classifier

$$P(Y|X) = \frac{P(X,Y)}{P(X)} = \frac{P(X|Y)P(Y)}{P(X)}$$

Output

Input features

Issue with Naïve. Bayes:

If one of the conditional probabilities is zero, then the entire expression becomes zero

9. Estimate probabilities for continuous attribute

Discretization: Partition the range into bins:

- Replace continuous value with bin value
  - Attribute changed from continuous to ordinal

Probability density estimation:

- Assume attribute follows a normal distribution
- Use data to estimate parameters of distribution (e.g., mean and standard deviation)
- Once probability distribution is known, use it to estimate the conditional probability P(X_i|Y)

- Normal distribution:

$$P(X_i | Y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(X_i - \mu_{ij})^2}{2\sigma_{ij}^2}}$$

- One for each (X_i,Y_j) pair

10. Methods for estimating a classifier's accuracy:

Holdout, random subsampling, cross-validation, bootstrap

11.

Confusion Matrix:

Actual class\Predicted class	C	¬ C
C	**True Positives (TP)**	**False Negatives (FN)**
¬ C	**False Positives (FP)**	**True Negatives (TN)**

12. Evaluation metric:

Accuracy = (TP + TN)/All Error rate = (FP + FN)/All

(Class imbalance problem)

Sensitivity = TP/P Specificity = TN/N

Precision: exactness – what % of tuples that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

Recall: completeness – what % of positive tuples did the classifier label as positive?

$$recall = \frac{TP}{TP + FN}$$

Perfect score is 1.0

Inverse relationship between precision & recall

***F* measure (*F₁* or *F*-score)**: harmonic mean of precision and recall,

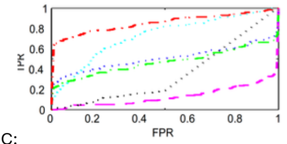
$$F = \frac{2 \times precision \times recall}{precision + recall}$$

***F*$_\beta$:** weighted measure of precision and recall

- assigns ß times as much weight to recall as to precision

$$F_\beta = \frac{(1+\beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

- **TPR = TP/P; FPR = FP/N**



ROC:

Issues Affecting Model Selection

- **Accuracy**
  - classifier accuracy: predicting class label
- **Speed**
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)
- **Robustness**: handling noise and missing values
- **Scalability**: efficiency in disk-resident databases
- **Interpretability**
  - understanding and insight provided by the model

13. Ensemble method:

Ensemble Methods work better than a single base classifier if:

All base classifiers are independent of each other

All base classifiers perform better than random guessing (error rate < 0.5 for binary classification)

14. Bagging:

Bootstrap sampling: sampling with replacement

Build classifier on each bootstrap sample

Add final classification/ majority vote

15. Boosting

An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records

Initially, all N records are assigned equal weights (for being selected for training)

Unlike bagging, weights may change at the end of each boosting round, wrong classification increase weight, vice versa

16. Random Forest

Construct an ensemble of decision trees by manipulating training set as well as features

Use bootstrap sample to train every decision tree (similar to Bagging)

Use the following tree induction algorithm:

At each node use a random selection of attributes as candidates and split by the best attribute among them

Repeat this procedure until all leaves are pure (unpruned tree)

17. Ensemble methods try to reduce the variance of complex models (with low bias) by aggregating responses of multiple base classifiers

Typical methods for imbalance data in 2-class classification:

- **Oversampling**: re-sampling of data from positive class
- **Under-sampling**: randomly eliminate tuples from negative class
- **Threshold-moving**: moves the decision threshold, t, so that the rare class tuples are easier to classify, and hence, less chance of costly false negative errors
- **Ensemble techniques**: Ensemble multiple classifiers introduced above