**Association Rule**: In short, **support** of {A, B, C}->{D} refers to transactions containing A, B, C, and D, while **confidence** of it is support/(transactions containing A, B, C). **Apriori** (1) Join: Generate candidate itemsets of length from frequent itemsets of length . (2) Prune: Remove candidates that have infrequent subsets. (3) Count: Remove candidates whose support is less than support. **FP-Tree**: From FP-Tree to transactions: bottom-up deduction; From FP-Tree to patterns: for every frequent item, compute the conditional(elements above, not including itself), then generalize to all the combs of the conditional elements. **Sequential K-means Clustering** Acquire the next example x, If is closest to x: (1) Increment (2)Replace by **Forgetful** Update Func: Retaining: **Hierarchical Clustering Methods**: **(a)Agglomerative(bottom-up)**: Keep finding the nearest clusters. (1)Single Linkage (nearest neighbor); (2)Complete Linkage(the distance between the most distant members); (3) Group Average Clustering(average of the distances between all pairs of records); (4) Centroid Clustering(the mean vectors of the two clusters); (5) Median Clustering. **(b) Divisive:(top-down)** (1) polythetic(group average linkage): Assign everything to B initially. For every element, check it is distance to A and B, then compute D(p,B)-D(p, A), select the max positive value. (2) monothetic: Chi-Square: , for every attribute, compute the sum of chi-square(). Choose the attribute with the max sum. **Subspace Clustering**: Density-Based: In dim-1, find all dense units, then use Apriori. Find subspaces-look like: {x}; Identify clusters: look like {x1},{x1y2}; Entropy-Based: Compute the entropy for {x}, {y}, {z} Entropy<=T. Then use Apriori; KL-transform-a) Compute Covariance Matrix: Suppose data is M(N\*d: N is the number of samples, while d is the number of features.) Normalize data matrix: The covariance matrix is b) Compute PCA: We would have d eigenvalues, and their corresponding eigenvectors. We could arrange them in descending order: ; c) Select the last k dimensions. We now have a matrix For a new sample, we could compute to get a vector of dimension k. Different from PCA, which selects the first k dimensions. PCA captures the information of high variance, while the transformation-based clustering focus on information of low variance. **Decision Tree**: ID3 : Information=Entropy= Gain=. Info(T, Attribute)=weighted sum of Info(T, Attribute=a); C4.5: Gain=Gain of ID3/SplitInfo, splitInfo=entropy of the split;(Why 4.5? In ID3, there is a higher tendency to choose an attribute containing more values (e.g., attribute identifier and attribute HKID). Thus, splitInfo in C4.5 is used to penalize an attribute containing more values. If this value is larger, the penalty is larger. CART: Gini= used as info **Bayes Belief Network**: put variables pointing to y as the conditions, while using Bayes rules for variables pointed to from y. When computing P(z1,z2|x), using P(z1,z2|y)\*P(y|x), then suppose conditional independence. Disadvantages: (1) The Bayesian Belief network classifier requires a predefined knowledge about the network. (2) The Bayesian Belief Network classifier cannot work directly when the network contains cycles. **SVM** Original Problem: Transformed problem: (1) why transform? We want to transform the objective function from a non-linear form to a quadratic form. Then, the problem becomes a form of quadratic programming which has many existing efficient techniques for that. (2) # constraints = #points **LSTM** (1)Forget Gate: (2)Input Gate: (3)Input Activation: (4)Cell State: (5)Output Gate: (6)Hidden State: **GRU**: (1) Reset Gate: (2) Update Gate: (3) Candidate Activation: (4) Hidden State: The advantages of the GRU model: (1) The training time is shorter. (2) It requires fewer data points to capture the properties of the data. The disadvantages of the GRU model: (1) It has a lower power of capturing the properties in the data. Thus, the result generated by GRU is usually less accurate than the traditional LSTM model. (2) It could not "remember" or "memorize" longer sequences compared with the traditional LSTM model. **Data Warehouse** Gain: compute for all the nodes below. Greedy Algorithm: Choose node with max gain. Other concepts: minimum overall access cost-the cost of obtaining the sql from other sqls. **Data Streaming** -deficient synopsis: 1, There is no false negative(Frequent classified as infrequent); 2, The difference between the estimated frequency and the true frequency is at most ; 3, All items whose true frequencies less than (s-)N are classified as infrequent items in the algorithm output **Sticky Sampling Algorithm** t=. Data 1-2t, r(sampling rate)=1; 2t+1-4t: r=2. Algorithm: (1) When data e arrives, if e exists in S, increment f in (e, f), else if e does not exist in S, add entry (e, 1) with prob. 1/r. (2) Just after r changes, for each entry (e, f), repeatedly toss a coin with P(head) = 1/r until the outcome of the coin toss is head. If the result is tail, decrement f in (e, f), and totally delete e if f=0(won't continue tossing the coin even if it is not head) (3) Get a list of items where . Has -deficient synopsis. Memory Consumption: At most **Lossy Counting Alg** width , b=. (1) When data e arrives, If e exists in D, Increment f in (e, f, ); else if e does not exist in D, add entry (e, 1, ) (2) Whenever it reaches the bucket boundary(reads in w elements), delete all elements whose . (3) Get a list of items where . Has -deficient synopsis. Memory Consumption: At most **Space-Saving** , (1) When data e arrives, If e exists in D, Increment f in (e, f, ) else If e does not exist in D, (If the size of D = M: ; Remove all entries e where ). Add entry (e, 1, ) , Get a list of items where **HITS** adjacency matrix: A(i, j), i points to j. Iteration: , **Pagerank** Stochastic Matrix: if webpage has outgoing links, then if there is a link from to ; otherwise, ; Update Func: . Initial of HITS and PageRank: 1. spider trap: a group of one or more pages that have no links out of the group will eventually accumulate all the importance of the web. **GNN** (1) Node-level task: To predict whether a node is in one group; (2) Edge-level task: To predict whether two nodes are connected with strong strength; (3) Graph-level task: To predict whether a whole graph belongs to one category. Five Major GNN models: (1)Message Passing Neural Network(MPNN); (2) Graph Convolution Network (GCN); (3)Graph Attention Network (GAN); (4) GraphSAGE (Sample and aggregate); (5) Recurrent Graph Neural Network(RecGNN) Notes: In a very large graph, GraphSAGE could perform well. **EM Algorithm**: Find maximum likelihood estimates for models with latent variables. Steps: (1) E-step: Calculate the expected value of the latent variables given the current parameters. (2) M-step: Maximize the likelihood function to update the parameters. **DBSCAN**: Density-based clustering to identify clusters and noise. Key points: (1) eps: Maximum distance between two samples to be considered neighbors. (2) minPts: Minimum number of points to form a dense region. **Alg Process**: (1) For each data point p Perform a range query from p with radius ; N(p) ← the result of the range query. If |N(p)|>=Minpts, Mark p as a core point. (2) For each core point p, Generate a cluster C for p. (3) While there exist two clusters C1 and C2 such that there exist p1 in C1 and p2 in C2, merge these two clusters. (4) For each point p where |N(p)| < Minpts, if N(p) contains a core point q, Assign p to the cluster q belongs to.