# CSE 4502/5717 Big Data Analytics
## Fall 2024 Exam 3 Helpsheet

1. **Association Rules Mining.** An **itemset** is a set of items. A $k$**-itemset** is an itemset of size $k$. A **transaction** is an itemset. A **rule** is represented as $X \to Y$ where $X \neq \emptyset, Y \neq \emptyset, X \cap Y = \emptyset$.

   We are given a database $DB$ of transactions and the number of transactions in the database is $n$. Let $I$ be the set of distinct items in the database and let $d = |I|$.

   For an itemset $X$, we define $\sigma(X)$ as the number of transactions in which $X$ occurs, i.e. $\sigma(X) = |\{T \in DB | X \subseteq T\}|$ The **support** of any rule $X \to Y$ is $\frac{\sigma(X \cup Y)}{n}$. The **confidence** of any rule $X \to Y$ is $\frac{\sigma(X \cup Y)}{\sigma(X)}$.

   Association Rules Mining is defined as follows.

   **Input:** A DB of transactions and two numbers: minSupport and minConfidence.
   **Output:** All rules $X \to Y$ whose support is $\geq$ minSupport and whose confidence is $\geq$ minConfidence.

   An itemset is **frequent** if $\sigma(X) \geq n \cdot \text{minSupport}$

   We discussed the Apriori algorithm for finding all the frequent itemsets. This algorithm is based on the a priori principle: If $X$ is not frequent then no superset of $X$ is frequent. Also, If $X$ is frequent then every subset of $X$ is also frequent.

   The pseudocode for the Apriori algorithm is given next.

   **Algorithm 1:** Apriori algorithm

   ```
   k := 1;
   Compute F_1 = {i ∈ I | σ(i) ≥ n · minSupport};
   while F_k ≠ ∅ do
       k := k + 1;
       Generate candidates C_k from F_{k-1};
       for T ∈ DB do
           for C ∈ C_k do
               if C ⊆ T then
                   σ(C) := σ(C) + 1;
       F_k := ∅;
       for C ∈ C_k do
           if σ(C) ≥ n · minSupport then
               F_k := F_k ∪ {C};
   ```

   We can use a hash tree to compute the support for each candidate itemset.

   We also presented a randomized Monte Carlo algorithm for identifying frequent itemsets. The idea was to pick a random sample, identify frequent itemsets in the sample (with a smaller support) and output these. We proved that the output of this algorithm will be correct with a high probability using the Chernoff bounds:

   If $X$ is $B(n, p)$, then the following are true:

   $$Prob.[X \geq (1+\epsilon)np] \leq \exp(-\epsilon^2 np/3)$$

   $$Prob.[X \leq (1-\epsilon)np] \leq \exp(-\epsilon^2 np/2),$$

for any $0 < \epsilon < 1$.

2. **Polynomial Arithmetic.** A degree-$n$ polynomial can be evaluated at a given point in $O(n)$ time. Lagrangian interpolation algorithm runs in $O(n^3)$ time whereas Newton's interpolation algorithm takes $O(n^2)$ time.

   Two degree-$n$ polynomials can be multiplied in $O(n \log n)$ time. A degree-$n$ polynomial can be evaluated at $n$ given arbitrary points in $O(n \log^2 n)$ time. Also, interpolation of a polynomial presented in value form at $n$ arbitrary points can be done in $O(n \log^3 n)$ time.