Algoritmo Apriori

Comumentemente utilizado para extração de dados, este algoritmo utiliza uma função hash sob uma árvore para coletar informações de um banco de dados. Este algoritmo recebe como parâmetro um conjunto de transações **T**, o valor percentual **S** como o suporte e um valor percentual C para a confiança. O algoritmo gera um conjunto de regras no formato A => B **[suporte, confiança]**, onde o conjunto A é chamado de antecedente da regra e o conjunto B é chamado de conseqüente. O Algoritmo parte do princípio de que se um item é frequente, então todos os seus subconjuntos também são frequentes(PICHILIANI, 2008).

O algoritmo APRIORI é dividido em duas partes. Na primeira parte são selecionados todos os subconjuntos de **T** que podem ser utilizados em alguma regra, ou seja, que contenham o suporte acima do suporte mínimo **S**. A segunda parte do algoritmo faz a geração das regras a partir dos subconjuntos gerados na primeira parte, sendo que estas regras devem ter uma confiança maior que a confiança mínima **C** (PICHILIANI, 2008).

Pichiliani (2008) define 4 passos para obter a primeira parte do algoritmo:

- Inicializações: neste passo são iniciadas as estruturas de memória que conterão os dados trabalhados no algoritmo. É gerado uma estrutura que conterá os conjuntos com 1 elemento, isto é todos os elementos da base **T** são colocados neste conjunto. Chamaremos este conjunto de **L1**, pois ele contém apenas os conjuntos com um elemento, ou seja, todos os conjuntos unitários com os produtos. Inicia-se também um variável chamada **K** com o valor 2, pois ela será o indexador da estrutura que armazena os conjuntos;
- Gerar os subconjuntos: neste passo é necessário gerar todos os subconjuntos possíveis de **Lk-1** que tenham o tamanho **K**, pois é a partir destes subconjuntos de **Lk-1** que o algoritmo vai determinar quais podem ou não ser utilizados. Chamaremos todos os subconjuntos com tamanho **K** que podem ser gerados a partir de **Lk-1** de **Ck**. Esta geração de subconjuntos é muito importante e, muitas vezes, é neste passo que o algoritmo demora;
- Filtrar os subconjuntos: neste passo vamos analisar cada elemento do conjunto **Ck**, que foi gerado no passo anterior. Calcula-se qual o suporte de cada elemento de **Ck** e selecionam-se apenas os elementos de **Ck** cujo suporte seja maior do que o suporte mínimo especificado como parâmetro e chamado de **S**.Todos os elementos de **Ck** cujo suporte seja maior do que o suporte mínimo são jogados dentro do conjunto **Lk;**
- Seleção do Conjunto: todos os elementos de **Ck** cujo suporte seja maior do que o suporte mínimo especificado são armazenados em uma estrutura indexada por **K** chamada **Uk**. Em seguida, o algoritmo incrementa o contador **K** e verifica se existe algum elemento em **Lk-1**. Se existir, o algoritmo volta para o **PASSO 2**. Caso contrário, o algoritmo termina retornado a união de todos os elementos dos conjuntos **Uk**. É importante lembrar os elementos de **Uk** podem ser conjuntos de elementos como, por exemplo, o conjunto {A1, A4}, utilizado como antecedente da regra apresentada no começo deste artigo.

A segunda parte do algoritmo gera as regras a partir de todos os elementos retornados pelo primeiro passo. Esta parte do algoritmo é bem simples: basta verificar para cada um dos elementos retornados pela primeira parte do algoritmo qual é o suporte deste conjunto em relação aos elementos de todos os conjuntos **Lk** (PICHILIANI, 2008).

Referências

Pichiliani, Mauro. Data Mining na Prática: Regras de Associação. Disponível em: <http://imasters.com.br/artigo/7853/sql-server/data-mining-na-pratica-regras-de-associacao?trace=1519021197&source=single>. Acessado em 12/10/2016.