**-Naive Bayes :**

**Teorema de Bayes:**

- Probabilidade “a Priori” = P(A)

- Probabilidade condicional = P(B|A)

- Probabilidade conjunta = P(A) P(B|A)

- Probabilidade “a Posteriori” = P(A|B)

**-SVM :**

**-SimpleKMeans :**

A idéia do algoritmo K-Means (também chamado de K-Médias) é fornecer uma classificação de informações de acordo com os próprios dados. Esta classificação, como será vista a seguir, é baseada em análise e comparações entre os valores numéricos dos dados. Desta maneira, o algoritmo automaticamente vai fornecer uma classificação automática sem a necessidade de nenhuma supervisão humana, ou seja, sem nenhuma pré-classificação existente. Por causa desta característica, o K-Means é considerado como um algoritmo de mineração de dados não supervisionado.

**PASSO 01: Fornecer valores para os centróides. Neste passo os k centróides devem receber valores iniciais. No início do algoritmo geralmente escolhe-se os k primeiros pontos da tabela. Também é importante colocar todos os pontos em um centróide qualquer para que o algoritmo possa iniciar seu processamento.**

**PASSO 02: Gerar uma matriz de distância entre cada ponto e os centróides. Neste passo, a distância entre cada ponto e os centróides é calculada. A parte mais ‘pesada’ de cálculos ocorre neste passo pois se temos N pontos e k centróides teremos que calcular  N x k distâncias neste passo.**

**PASSO 03: Colocar cada ponto nas classes de acordo com a sua distância do centróide da classe. Aqui, os pontos são classificados de acordo com sua distância dos centróides de cada classe. A classificação funciona assim: o centróide que está mais perto deste ponto vai ‘incorporá-lo’, ou seja, o ponto vai pertencer à classe representada pelo centróide que está mais perto do ponto. É importante dizer que o algoritmo termina se nenhum ponto ‘mudar’ de classe, ou seja, se nenhum ponto for ‘incorporado’ a uma classe diferente da que ele estava antes deste passo.**

**PASSO 04: Calcular os novos centróides para cada classe. Neste momento, os valores das coordenadas dos centróides são refinados. Para cada classe que possui mais de um ponto o novo valor dos centróides é calculado fazendo-se a média de cada atributo de todos os pontos que pertencem a esta classe.**

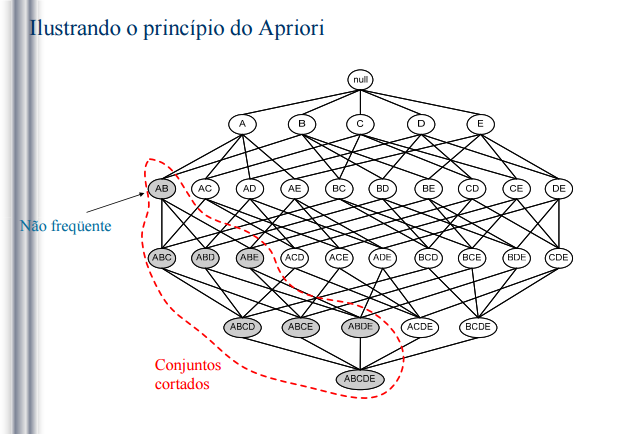
**PASSO 05: Repetir até a convergência. O algoritmo volta para o PASSO 02 repetindo iterativamente o refinamento do cálculo das coordenadas dos centróides.**

**Notem que desta maneira teremos uma classificação que coloca cada ponto em apenas uma classe. Desta maneira dizemos que este algoritmo faz uma classificação *hard* (hard clustering) uma vez que cada ponto só pode ser classificado em uma classe. Outros algoritmos trabalham com o conceito de classificação *soft* onde existe uma métrica que diz o quão ‘dentro’ de cada classe o ponto está.**

**-Apriori :**

Este algoritmo trabalha por meio de uma busca aprofundada nos dados gerando conjuntos de padrões, onde os mais frequentes são mantidos e os menos são eliminados.

O Apriori é um algoritmo de geração de regras de associação. Em mineração de dados regras de associação são usadas para descobrir elementos que ocorrem em comum dentro de um determinado conjunto 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. Cara regra gerada deve ser seu suporte e sua confiança maior ou igual ao suporte e confiança mínimo passado para o algoritmo, respectivamente.

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**.

**PASSO 1**: 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.

**PASSO 2:**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 subconjutos 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.

**PASSO 3:** 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**.

**PASSO 4:**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. Este 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.**

**-EM :**

**-Árvore de Decisão :**

Fontes :

-Naive Bayes :

<https://www.analyticsvidhya.com/blog/2015/09/naive-bayes-explained/>

<http://www.saedsayad.com/naive_bayesian.htm>

<http://machinelearningmastery.com/naive-bayes-classifier-scratch-python/>

<http://docs.orange.biolab.si/3/visual-programming/widgets/classify/naivebayes.html>

<https://www.youtube.com/watch?v=3HJVRBEMwoU>

-SVM :

<https://www.maxwell.vrac.puc-rio.br/9947/9947_6.PDF>

<http://scikit-learn.org/stable/modules/svm.html>

<http://docs.orange.biolab.si/3/visual-programming/widgets/classify/svm.html>

<http://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html>

-SimpleKMeans :

<https://imasters.com.br/artigo/4709/sql-server/data-mining-na-pratica-algoritmo-k-means/?trace=1519021197&source=single>

<https://www.google.com.br/search?q=SimpleKMeans&oq=SimpleKMeans&aqs=chrome..69i57&sourceid=chrome&ie=UTF-8>

<https://datasciencelab.wordpress.com/2013/12/12/clustering-with-k-means-in-python/>

-Apriori :

<http://www.deamo.prof.ufu.br/arquivos/Aula2.pdf>

<https://imasters.com.br/artigo/7853/sql_server/data_mining_na_pratica_regras_de_associacao/>

<http://www.lbd.dcc.ufmg.br/colecoes/eniac/2013/0038.pdf>

<https://www.profissionaisti.com.br/2013/11/data-mining-atraves-da-regra-de-associacao-apriori/> (usa um artigo como base)

<http://www.inf.ufrgs.br/~alvares/CMP259DCBD/RegrasDeAssociacao.pdf> (imagem)

-EM :

<http://www.cmi.ac.in/~madhavan/courses/datamining12/reading/em-tutorial.pdf>

<http://www.nature.com/nbt/journal/v26/n8/full/nbt1406.html>

-Árvore de Decisão :

<http://web.tecnico.ulisboa.pt/ana.freitas/bioinformatics.ath.cx/bioinformatics.ath.cx/indexf23d.html?id>

<http://conteudo.icmc.usp.br/pessoas/mello/courses/scc5879-aula05.pdf>

<http://www-usr.inf.ufsm.br/~pozzer/disciplinas/pj3d_decisionTrees.pdf>

<http://pt.wikihow.com/Criar-uma-%C3%81rvore-de-Decis%C3%A3o>

<http://docs.orange.biolab.si/2/reference/rst/Orange.classification.tree.html>