**Anotações IA**

**Tipos de dados:**

* Estruturados = Dados organizados em uma tabela.
* Semi-estruturados = Dados em uma tabela mas ainda não organizados.
* Não estruturados = Arquivos não organizados (PDF, Jornal, etc).

**Data Crawler:**

* Algoritmo que busca dados na internet.

**Tipos de problemas em Data Science:**

* Classe
* Cluestering
* Regressão / Forecasting

**Data Preparation:**

**Etapas iniciais =**

* **Data cleaning**

Remoção / Processamento

1. Outliersc
2. Duplicated
3. Nulls
4. Blanks

* **Data transformation**
* **Data discretization**
* **Data scaling**
* **Data reduction**

**Quantitativa = números.**

**Ordinal = ordem, classe grupo.**

**Qualitativa = características.**

**Amostra Sistemática = Baseada em Regra.**

**Amostra Casual simples = Aleatória.**

**Tendência central (Média) =** Tudo que tende ao centro.



**Três tendências centrais =** Média, moda, mediana.

**Desvio simples =** (Cada valor)X - (media)\_X



**Amplitude =** (Num maior)N> - (Num menor)N<

**Variância = Ex2(quadrado) =** 40 / N-1 (Para pegar o 40, fazer cada X ao quadrado)

**Desvio Padrão =** Raiz quadrada da variância

**Redução de dimensionalidade**

PCA (Principal Component Analysis)

* Seleção de características x Extração de características
* Um dos principais algoritmos de aprendizagem de maquina não supervisionada
* Identifica a correlação entre variáveis, e caso haja uma forte correlação é possível reduzir a dimensionalidade
* Das ***m*** variáveis independentes, PCA extrai **p <= m** novas variáveis independentes que explica melhor a variação na base de dados, sem considerar a variável dependente
* O usuário pode escolher o numero de **p**

**LDA (Linear Discriminant Analysis)**

* Além de encontrar os componentes principais, LDA também encontra os eixos que maximizam a separação entre múltiplas classes
* É supervisionado por causa da relação com a classe
* Das **m** variáveis independentes, LDA extrai **p <= m** novas variáveis independentes que mais separam as classes de variável dependente

**Kernel PCA**

* PCA e LDA são utilizados quando os dados são linearmente separáveis
* Kernel PCA é uma versão do PCA que os dados são mapeados para uma dimensão maior usando o **kernel trick**
* Os componentes principais são extraídos dos dados com dimensionalidade

**Naive bayes**

Analisa os dados e gera uma tabela de probabilidade, estima uma probabilidade no novo registro em cada uma das classes.

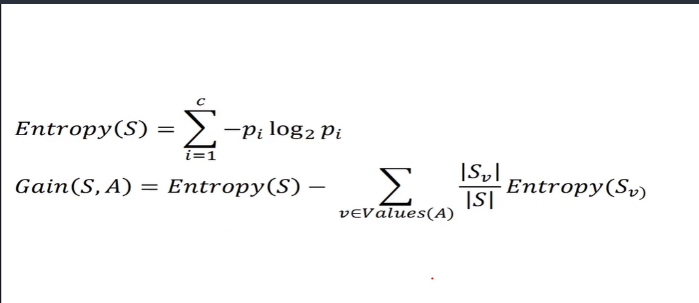
**Arvore de decisão**

Gera uma estrutura de arvore para classificar um registro.

**Cálculo:**

Cálculo entropia e ganho de informação.

Analisa atributo por atributo para verificar qual é o mais importe para colocar nos ramos superiores da arvore de decisão.



Entropia = Verifica a aleatoriedade dos dados.

Ganho de informação = Verifica o quanto de informação cada coluna tem para todas as colunas.

**Random Forest**

Combina a decisão de cada arvore de decisão.

O conceito de aprendizagem é de Ensemble learning (Aprendizagem em conjunto).

Usa média para Regressão Linear, tira uma média de todos os valores gerados.

Usa votos da maioria para classificação para dar o resultado.

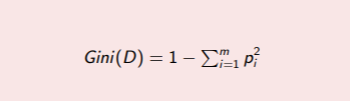
***Conceito do nome Random Fores:***

Random (Randomico).

Escolhe de forma aleatória K (Onde K é o numero de atributos) atributos para comparação da métrica de pureza / impureza.

Calculo:

***Impureza de gini*** = Mede a impureza de D, de uma partição de dados ou de um conjunto de tuplas (Linha) de treinamento.



onde Pi é a probabilidade da tupla em D pertencer a classe Ci e é estimada por |Ci,D |/|D|. A soma é computada sobre m classes. Se os dados pertencem a uma mesma classe a impureza é 1 − 1 = 0 (os dados são puros).

***Entropia*** = Minimiza a informação necessária para classificar os dados das partições resultantes e reflete a impureza ou a aleatoriedade das partições. A abordagem é minimizar o expected information requirements(requisitos de informação esperados) necessário para discriminar os dados a partir da existência da partição.

**Aprendizagem por regras.**

Gera regras, para classificação ele verifica se o dado se encaixa em cada uma das regras até que se encaixe em alguma.

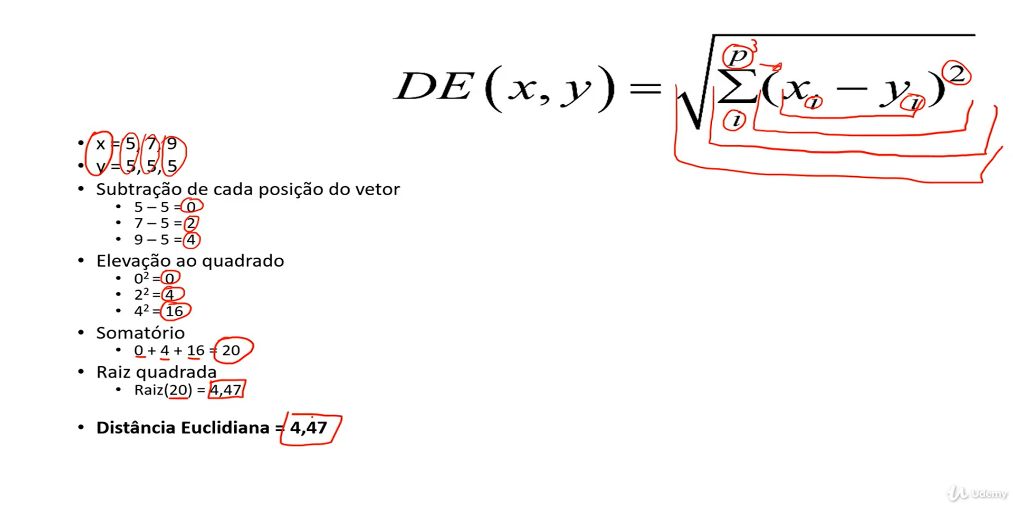
**K- Nearest Neighbour (Aprendizagem baseado por instancias KNN)**

Vizinho mais próximos, calcula a distância do novo registro com os dados já classificados, não gera modelos, armazena os dados.

Não constrói modelo, apenas armazena os dados e quando entra um novo registro faz o cálculo de distância. (LAZY = Métodos que não geram modelos).

**Cálculo:**

Distância Euclidiana.



Calcula a quão distante um registro está do outro.

***Exemplo de aplicação:***

Sistemas de recomendação (Filtragem colaborativa) como Netflix, YouTube.

**Regressão logística**

Calcula a melhor curva possível para classificação.

**SVM (Suporte vector machine)**

Para classificação de um novo registro ele verifica em qual lado do hiperplano aparece.

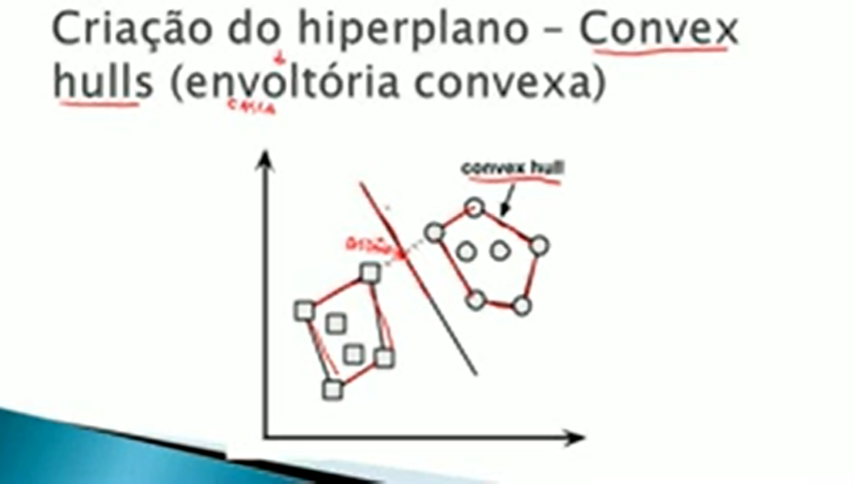
Utilizado para tarefas complexas, reconhecimento de caracteres, voz, imagens.

Aprende hiperplanos (linha) de separação com margem máxima, encontra a melhor reta possível.

Margem máxima significa a distância entre o hiperplano que é gerado e os vetores de suporte que estão próximos, quanto mais distantes dos vetores mais eficiente na generalização de novos registros.

**Calculo:**

Convex hulls (Envoltória convexa)



Acha os pontos de ligação em cada um dos vetores.

Abordagem matemática (MAIS UTILIZADA)

Formula DOT PRODUCT = W \* X + B - 0

(HIPERPLANO DE SEPARAÇÂO)

Encontra reta parcial e com base da reta gera a hiperplano com a margem máxima.

**Regressão Linear**

Objetivo é prever números, valores e não realizar uma classificação.

Utiliza o cálculo da descida do Gradiente para localizar o mínimo global para minimizar os erros.

Regressão Linear simples = Utiliza apenas um atributo e a classe.

Regressão Linear múltipla = Utiliza mais de um atributo.

**Produto Escalar**

O produto escalar é uma função binária definida entre dois vetores que fornece um número real (também chamado "escalar") como resultado.

O resultado do produto do comprimento de B pela **Projeção Escalar** de A em B