## Cientista de dados:

Alguém que extrai conhecimento de dados desorganizado.

Conceitos de **geradores** e **iteradores** para trabalhar com grandes conjuntos de dados ou processos contínuos de forma eficiente. Esses conceitos permitem processar elementos um por um, sem carregar tudo na memória de uma vez, o que ajuda a economizar espaço e torna o código mais rápido em algumas situações.

Um **iterador** em Python é um objeto que representa uma sequência de valores e permite iterar sobre eles, um de cada vez. Por exemplo, quando usamos um for loop, estamos na verdade usando um iterador para processar a sequência.

Um **gerador** é uma forma mais prática de criar um iterador. Ele permite gerar valores sob demanda, usando a palavra-chave yield para retornar um valor sem encerrar a função. Quando yield é chamado, o estado da função é "pausado" e, na próxima vez que o gerador for chamado, ele continua de onde parou. Isso é muito útil para sequências grandes ou infinitas.

### Vantagens dos Geradores

1. **Eficiência de Memória**: Como os valores são gerados sob demanda, o gerador não precisa armazenar toda a sequência na memória.
2. **Leitura de Códigos Longos e Contínuos**: Geradores são ideais para trabalhar com grandes arquivos de dados ou quando a sequência de dados é muito longa.
3. **Performance em Grandes Estruturas de Dados**: Para processar elementos de grandes estruturas de dados, eles reduzem o tempo de processamento, pois geram os elementos apenas quando solicitados.
4. **Lazy Evaluation (Avaliação Preguiçosa)**: Geradores só computam o próximo valor quando necessário, diferentemente de listas, que armazenam todos os valores de uma vez.

A função enumerate em Python é usada para iterar sobre uma sequência (como uma lista, tupla, ou string) enquanto acompanha a posição de cada elemento. Essa função é especialmente útil quando você precisa tanto do **índice** quanto do **valor** de cada item em uma iteração.

### \*args: Argumentos Posicionais Variáveis

\*args permite que uma função receba um número arbitrário de argumentos posicionais. Os argumentos recebidos com \*args são armazenados como uma **tupla** dentro da função.

### \*\*kwargs: Argumentos Nomeados Variáveis

\*\*kwargs permite que uma função receba um número arbitrário de argumentos nomeados (chave-valor). \*\*kwargs transforma esses argumentos em um **dicionário**.

## Visualizando Dados:

Dois usos primários para visualização de dados: para explorar dados e para comunicar dados.

- Gráficos de Barra: boa escolha quando quer mostrar como algumas quantidades variam entre um conjunto particular de itens; para criar gráficos de histograma de valores numérico carregados, a fim de explorar visualmente como estão distribuídos.

- Gráfico de Dispersão: visualizar o relacionamento entre dois pares de conjuntos de dados.

## Álgebra Linear:

Ramo da matemática que lida com espaços vetoriais; sustenta um grande número de conceitos e técnicas de data Science.

Vetores: objetos que podem ser somados juntos (para formar novos vetores) e que podem ser multiplicados pelos escalares (exemplo, números), também para formar novos vetores; São pontos em algum espaço de dimensão finita.

Matrizes: coleção bidimensional; listas de listas, com cada lista interior possuindo o mesmo tamanho e representando uma linha da matriz.

Pode usar uma matriz para representar um conjunto de dados consistindo de múltiplos vetores, uma função linear que mapeia vetores k para vetores n, e relações binárias.

## Estatística:

Refere-se à matemática e às técnicas com as quais entendemos os dados.

Dispersão: refere-se à medida de como os nossos dados estão espalhados.

Covariância: equivalente pareado a variância, mas mede como duas variáveis variam em conjunto de suas medidas.

Paradoxo de Simpson: em que as correlações podem ser enganosas quando as variáveis de confusão são ignoradas.

## Probabilidade:

Quantificar a incerteza associada com eventos escolhidos a partir de um universo deles; podemos usar para construir e avaliar modelos.

Dependência: dizemos que dois eventos E e F são dependentes se E ocorre nos der informações sobre se F ocorre e vice-versa.

Matematicamente, dizemos que esses dois eventos são independentes se a probabilidade deles acontecerem é o produto da probabilidade de que cada um deles aconteça.

Probabilidade Condicional:

Quando dois eventos são independentes, temos: P(A,B) = P(A)\*P(B)

Se não são necessariamente independentes (e a P(B) != 0), definimos a probabilidade de A “condicionada a B”: P(A|B) = P(A,B) / P(B)

A probabilidade de A acontecer uma vez que sabemos que B: P(A,B) = P(A|B)\*P(B)

Quando são independentes: P(A|B) = P(A)

Onde obter dados:

Data.gov ; reddit; Kaggle; Amazon e Robb Seaton