Python para Data Science

Módulo Básico

paanalytics.net @paanalytics pedro.alves@paanalytics.net

Aula 1 - Conceitos básicos

Ementa do Curso

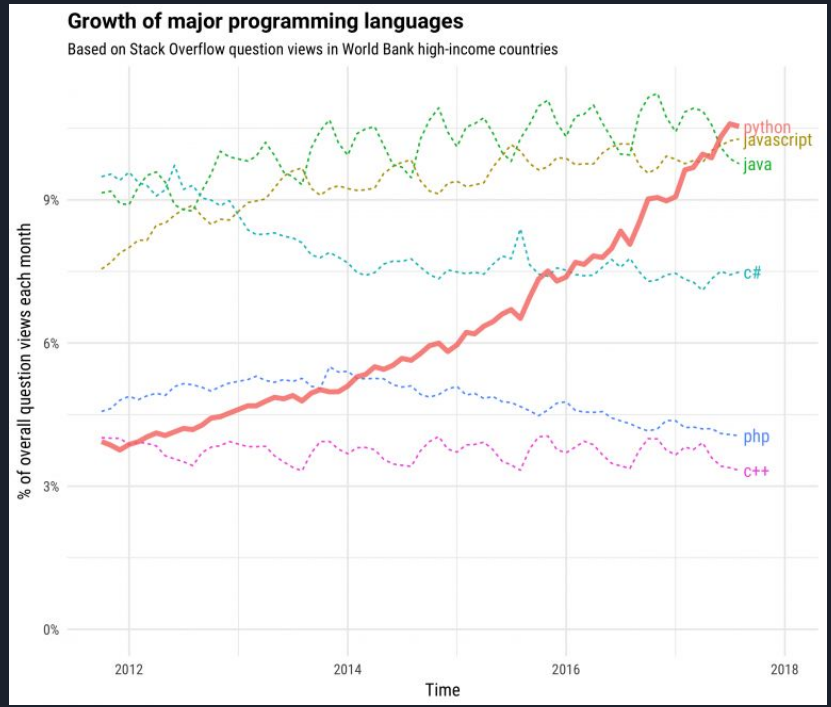
- **Aula 1 - Conceitos básicos**
  - **Python no mercado de trabalho**
  - **O que é uma IDE e como instalar uma IDE para usar Python (Jupyter Notebook)**
  - **Conceitos básicos Jupyter Notebook**
  - **Tipos de variáveis no Python**
  - **Funções básicas e métodos em Python**
  - **Listas**
  - **Dicionários**

Crescimento do Python ao longo dos anos

Python é uma linguagem que tem se tornado extremamente popular pela simplicidade e versatilidade.

Foi muito adotada para trabalhos de data science devido à grande disponibilidade de bibliotecas com essa finalidade.

O gráfico ao lado exemplifica o crescimento do Python. A referência usada é a % de perguntas no Stack Overflow (site usado para resolução de problemas relacionados à programação).

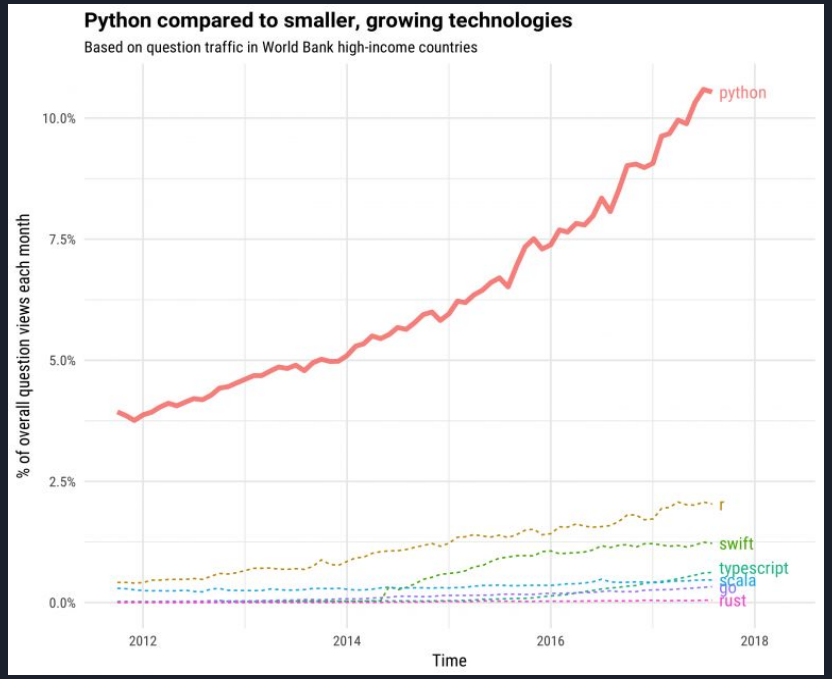


Crescimento do Python ao longo dos anos

Quando comparamos o Python com linguagens de programação mais novas e em desenvolvimento a diferença é ainda maior.

Quando falamos de ciência de dados, o R é a segunda linguagem mais usada, também extremamente útil para o assunto, porém o principal diferencial do Python é sua aplicabilidade tanto para uso em dados, quanto para usos gerais de programação

Isso o torna extremamente poderoso para desenvolver soluções mais completas.



18%

das vagas relacionadas a programação e análise de dados pedem conhecimento em Python de acordo com estudo feito em 2019*

fonte:https://www.digitalhouse.com/br/blog/por-que-aprender-python

Alta demanda e pouca oferta

Python ainda é um conhecimento pouco encontrado pelos recrutadores e muito procurado.

Salário médio de um desenvolvedor Python

R$ 7,1 mil

Fonte:https://blog.geekhunter.com.br/salario-de-programador-cargos-em-alta-2020/

Bora para o mão na massa?

Não se esqueça de avaliar o curso ★★★★★

paanalytics.net @paanalytics pedro.alves@paanalytics.net

Instalação do Python na máquina

Instalação do Python via Anaconda (plataforma mais popular de Data Science no mundo)

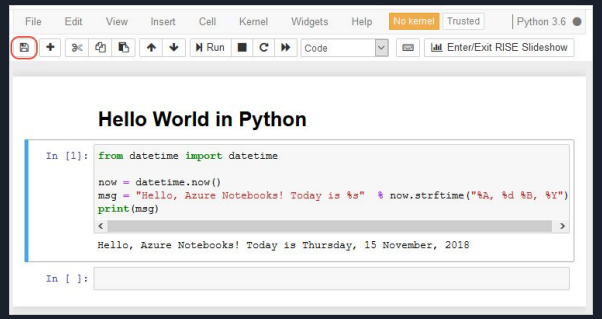
https://www.anaconda.com

Jupyter Notebook

Ferramenta IDE usada para escrever código em Python e documentar código.

Gera arquivos IPythonNotebook (.ipynb) que é um arquivo específico para abrir no jupyter notebook.

Também pode ser usado para gerar scripts Python (.py)



Conceitos básicos - Jupyter Notebook

Tipos de células:

- Markdown → Usada para documentar e escrever textos
- Code → Usada para códigos
    - O símbolo # dentro de uma célula de código irá transformar a linha em texto para documentação

Comandos básicos:

- Shift + Enter → Rodar célula e ir para a próxima
- Ctrl + Enter → Rodar célula

Conceitos básicos - Python

Python → Linguagem de tipagem forte → Diferencia letras maiúsculas de minúsculas

Variável: Usada para armazenar informações que usaremos com recorrência

Tipos de variáveis mais comuns no Python:

- String → Texto (Definida sempre com um texto entre aspas simples ou duplas)
- Int → Número Inteiro
- Float → Número Decimal (Casa decimal sempre delimitada com ponto e não vírgula)
- Bool → Booleano (Verdadeiro ou Falso)
- Lista
- Dicionário

Conceitos básicos - Operações com variáveis

- Operadores:
  - + → Soma (2+3 = 5)
  - - → Subtração (3 - 1 = 2)
  - * → Multiplicação (2 * 2 = 4)
  - / → Divisão (2 / 2 = 1)
  - ^ → Potência (2 ^3 = 8)
- Operações com texto:
  - + → Soma ('Oi' + '!' = Oi!)
  - * → Multiplicação ('Oi' * 2 = OiOi)

- Comparação:
  - == → Igual
  - != → Diferente
  - > → Maior
  - >= → Maior ou igual
  - < → Menor
  - <= → Menor ou igual

Conceitos básicos - Funções básicas e métodos

- Funções básicas:
  - print() → Imprime o texto ou variável entre parêntese na tela
  - import → Carrega bibliotecas Python
  - !pip install → instala bibliotecas Python
- Métodos - funções implícitas dentro de cada tipo de variável, acessadas usando ponto após a variável (ex: string.replace → Substitui uma parte do texto por outra)
- Instruções sobre uso de funções e métodos → Shift + Tab no Jupyter Notebook

Listas

Usadas para armazenar vários valores dentro de uma mesma variável

Identificadas por colchetes (Ex: lista = [1, 2, 3, ‘olá’, 0.5])

Operações com listas:

```
lista 1 + lista 2 = Lista contendo todos os elementos da lista 1 e lista 2

lista 1 * 2 = Lista com informações da lista 1 repetidos 2 vezes
```

Listas - Principais métodos

- lista.append(n) → Adiciona o elemento n à lista
- lista.remove(n) → Remove o elemento n da lista
- lista.sort() → Ordena a lista
- lista.count(n) → Conta quantos elementos n na lista
- lista.len() → Conta quantos elementos na lista
- lista.index(n) → Retorna qual a posição do elemento n na lista

Dicionários

dict = { ‘Segunda’ : 1, ‘Terça’:2, ‘Quarta’:3 , ‘Quinta’: 4 , ‘Sexta’: 5}

Dicionários são usados para armazenar pares

chave-valor, para posterior uso ou comparação

Identificados por chaves

Muito usados para realizar de-para de chaves-valor.

Principais Métodos:

dict.keys() → Retorna as chaves armazenadas no dicionário

dict.values() → Retorna os valores armazenados no dicionário

dict.get(n) → Retorna o valor associado à chave n no dicionário

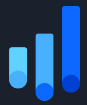
dict.pop(n) → Remove a chave n e o valor associado a ela no dicionário

Ementa do Curso

- Aula 1 - Conceitos básicos
  - Python no mercado de trabalho
  - O que é uma IDE e como instalar uma IDE para usar Python (Jupyter Notebook)
  - Conceitos básicos Jupyter Notebook
  - Tipos de variáveis no Python
  - Funções básicas e métodos em Python
  - Listas
  - Dicionários

Hora de Praticar Exercícios - Aula 1

Não se esqueça de avaliar o curso


paanalytics.net


@paanalytics


pedro.alves@paanalytics.net

Aula 2 - Manipulação de Dados

Ementa do Curso

- **Aula 2 - Manipulação de Dados**
  - **Numpy, Pandas e conceito de Dataframe**
  - **Importação/Exportação de Dados**
  - **Métodos básicos para Dataframes**
  - **Segmentação de Dataframes (Slicing)**
  - **Cruzamento de Bases (Merge)**

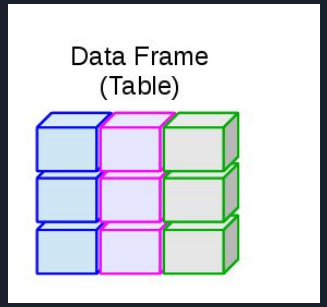
Numpy

Biblioteca usada para executar cálculos matemáticos em grandes volumes de dados

Sua principal vantagem em relação à funções matemáticas do Python é poder trabalhar com vetores

Desta forma é possível executar cálculos matemáticos com vários valores em sequência (como por exemplo uma coluna de uma tabela)

Pandas



Biblioteca usada para ingestão,manipulação e exportação de **Dataframes**

Dataframes são estruturas de linhas (índices ou index) e colunas em formato de tabela.

As linhas são chamadas de índices (index)

Importação/Exportação de Dados (Pandas)

Métodos usado para importar e exportar dados

Sintaxe:

```
pd.read_xxxx(string com caminho completo do arquivo)
```

Importação de Dados (exemplo pd.read_csv, pd.read_excel)

```
pd.to_xxxx(string com caminho completo do arquivo)
```

Exportação de Dados (exemplo pd.to_csv, pd.to_excel)

Métodos básicos para Dataframes

Métodos mais usados para Dataframes:

`df.head(n)` → visualizar as primeiras n linhas do Dataframe

`df.count()` → retorna a contagem de registros não vazios em cada coluna do dataframe

`df.describe()` → Retorna estatísticas descritivas do Dataframe (apenas colunas numéricas)

`df.sum()` → Retorna a soma dos valores em cada coluna do Dataframe

`df.mean()` → Retorna a média dos valores em cada coluna do Dataframe

Segmentação de Dataframes (Slicing)

Métodos usados para segmentação de Dataframes:

Loc = Usado para segmentação usando rótulos de índices e colunas → `df.loc[linhas,colunas]`

Iloc = Usado para segmentação usando números referentes à índices e colunas → `df.iloc[linhas,colunas]`

Segmentação por índices → `df[índices]`

Segmentação por colunas → `df['Coluna'] ou df[['Coluna1', 'Coluna2', 'Coluna3']]`

Cruzamento de Bases (Merge)

Método usado para combinar bases de dados usando uma ou mais chaves de combinação (Semelhante ao PROCV/VLOOKUP no Excel)

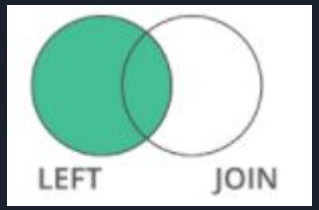
Sintaxe:

```
pd.merge(base1, base2, on=[[ 'chave1', 'chave2' ]], how= 'left'/'right'/'inner'/'outer')

dataframes combinados

chaves para combinação (caso seja apenas uma, não é necessário colchetes)

como a combinação será feita
```

Cruzamento de Bases (Merge)
Métodos de Combinação

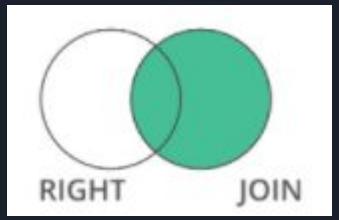
Left:

Traz todas as linhas e colunas do primeiro DataFrame + registros encontrados do segundo Dataframe



Right:

Traz todas as linhas e colunas do segundo DataFrame + registros encontrados do primeiro Dataframe

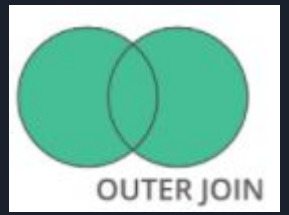


Inner:

Traz apenas as linhas onde o cruzamento de dados foi bem sucedido

Outer:

Traz todos os registros de ambos os dataframes (dados não encontrados ficam vazios)



Ementa do Curso

- **Aula 2 - Manipulação de Dados**
  - **Instalação de bibliotecas - Numpy e Pandas e conceito de Dataframe**
  - **Importação/Exportação de Dados**
  - **Métodos básicos para Dataframes**
  - **Segmentação de Dataframes (Slicing)**
  - **Cruzamento de Bases (Merge)**

Hora de Praticar Exercícios - Aula 2

Não se esqueça de avaliar o curso

paanalytics.net

@paanalytics

pedro.alves@paanalytics.net

Aula 3 - Funções Definidas pelo Usuário, Loop e Operadores Condicionais

Ementa do Curso

- **Aula 3 - Funções Definidas pelo Usuário , Loop e Operadores Condicionais**
  - **Como definir uma função**
  - **Função Lambda**
  - **Loop**
  - **Operadores Condicionais**

Funções Definidas pelo Usuário

Objetivo: Simplificar operações recorrentes para poderem ser chamadas apenas passando o nome da função e o argumento

Sintaxe:

`def funcao(argumentos)` ou `lambda x:`

Obs:

- É importante prever se é necessário definir um valor padrão para os argumentos

Loop

Objetivo: Executar operações repetidamente de acordo com uma regra definida pelo usuário

Exemplo: Executar a mesma operação em todos os valores ao longo de uma coluna

Sintaxe:

```
for i in seq::

while condição::
```

Operadores Condicionais

Objetivo: Executar operações de acordo com condições impostas pelo usuário (Semelhante à função SE no Excel)

Sintaxe:

```
If condição:

Elif condição:

Else:
```

Ementa do Curso

- **Aula 3 - Funções Definidas pelo Usuário , Loop e Operadores Condicionais**
  - **Como definir uma função**
  - **Função Lambda**
  - **Loop**
  - **Operadores Condicionais**

Hora de Praticar Exercícios - Aula 3

Não se esqueça de avaliar o curso

paanalytics.net @paanalytics pedro.alves@paanalytics.net

Aula 4 - Visualização de Dados

Ementa do Curso

- **Aula 4 - Visualização de Dados**
  - **Bibliotecas mais usadas**
  - **Começando com Matplotlib**
  - **Dica - Seaborn Heatmap**

Bibliotecas mais usadas - Matplotlib

Biblioteca mais comum para visualização de dados em Python

Possui diversas variações de visualizações e é usada como base para diversas outras bibliotecas disponíveis

Como carregar a biblioteca:

```
import matplotlib.pyplot as plt
```

Bibliotecas mais usadas - Seaborn

Biblioteca construída baseada em Matplotlib.

Possui variações mais elaboradas das visualizações e detalhes estéticos mais elaborados

Como carregar a biblioteca:

```
import seaborn as sns
```

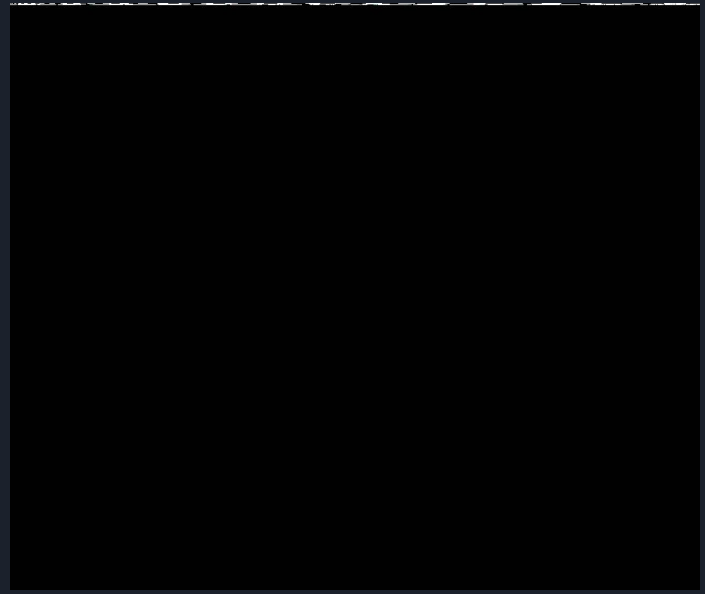
Bibliotecas mais usadas - Bokeh

Biblioteca usada para construir gráficos com botões interativos, muito usada para construção de dashboards.

Como carregar a biblioteca:

```
import bokeh
```

No caso do Bokeh, a biblioteca não vem como padrão instalada no Python. Então pode ser preciso instalar antes de carregar usando o comando `!pip install bokeh`



Matplotlib

Estrutura do Matplotlib

1. Geração de uma tela vazia (gráfico em branco)
2. Formatação da tela (eixos, título, etc)
3. Preenchimento da tela com dados - Aqui escolhemos qual o tipo de gráfico iremos usar
4. Formatação dos dados (legenda, etc)

Ementa do Curso

- **Aula 4 - Visualização de Dados**
  - **Bibliotecas mais usadas**
  - **Começando com Matplotlib**
  - **Dica - Seaborn Heatmap**

Hora de Praticar Exercícios - Aula 4

Não se esqueça de avaliar o curso

Projeto Final - Análise Exploratória de Dados com Python

O que é Análise Exploratória de Dados (AED)?

Em estatística, a análise exploratória de dados (AED) é uma abordagem à análise de conjuntos de dados de modo a resumir suas características principais, frequentemente com métodos visuais.

Qual o objetivo da análise exploratória?

Estudar a base de dados que se está trabalhando e entender padrões e pontos de atenção que possam impactar em análises posteriores

É através da AED que o Cientista de Dados entende os dados que estão disponíveis e começa a extrair insights e hipóteses que podem levar à resolução do problema

Quais as etapas da análise exploratória?

- *Tipos de Variáveis*
- *Estatística Descritiva e Tabelas de Frequência*
- *Detecção de dados nulos*
- *Detecção de Outliers*
- *Exploração Visual dos Dados*

Tipos de Variáveis

Ao começar um trabalho com uma base de dados qualquer o ideal é entender quais são os tipos de variáveis que temos.

Variáveis diferentes irão requerer tratamentos diferentes e entendimentos diferentes

Principais tipos:

- Variáveis Quantitativas: Que possuem um valor numérico associado
  - Discretas: Representam um conjunto finito (Ex: N° de filhos de um casal)
  - Contínuas: Representam um conjunto infinito de números (Ex: Peso de uma pessoa)
- Variáveis Qualitativas: Representadas por uma qualidade ou atributo
  - Ordinais: Apresentam uma ordem definida (Ex: Meses do Ano)
  - Nominais: Não apresentam ordem (Ex: Sexo ou cor dos olhos)

Estatística Descritiva e Tabelas de Frequência

Ao começar a explorar os dados geralmente usamos estatística descritiva para entender alguns aspectos das variáveis

Para variáveis quantitativas usamos estatística descritiva (média, moda, mediana, máximo, mínimo, etc)

Para variáveis qualitativas usamos tabelas de frequência

Detecção de Dados Nulos

Porque precisamos detectar e tratar dados nulos?

A maioria dos modelos de machine learning não trabalha com dados nulos, é importante que estes sejam tratados se a ideia do problema é resolver com modelagem.

Em problemas de análises e estudos, entender os dados nulos é importante para saber se sua base pode ser impactada pela quantidade de dados nulos (eles fazem diferença frente ao todo?)

Estratégia para tratar dados nulos:

- Exclusão de dados nulos (DropNa)
- Preenchimento de dados nulos (FillNa)

Detecção de Outliers

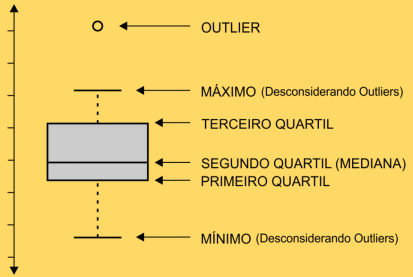
Um outlier é um valor que foge da normalidade e que pode (e provavelmente irá) causar anomalias nos resultados obtidos por meio de algoritmos e sistemas de análise.

Entender os outliers é fundamental em uma análise de dados por pelo menos dois aspectos:

1. os outliers podem viesar negativamente todo o resultado de uma análise;
2. o comportamento dos outliers pode ser justamente o que está sendo procurado.

Como Detectar um Outlier:

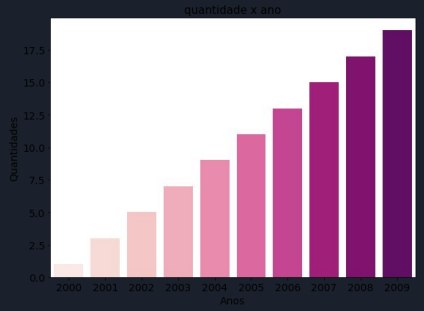
Método 1: Visualmente através de um Boxplot

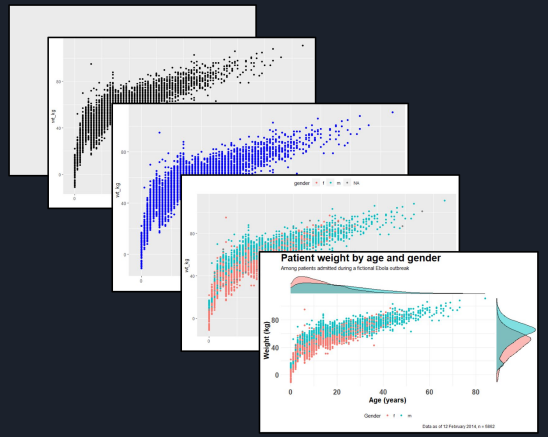


Método 2: Calculando os limites superior e inferior da sua amostra e comparando com o ponto testado

Exploração Visual dos Dados

Construção de visualizações que nos permitam identificar anomalias, padrões procurados e insights sobre os dados analisados





Parabéns!
Você chegou ao fim do módulo básico

Agora é com você
Continue Praticando

“Sucesso é o acúmulo de pequenos esforços, repetidos dia e noite”

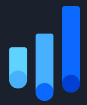
Robert Collie

paanalytics.net @paanalytics pedro.alves@paanalytics.net

Muito Obrigado pelo seu apoio!

Para dúvidas, sugestões e feedback, entre em contato comigo pelos canais abaixo

Não se esqueça de avaliar o curso


paanalytics.net


@paanalytics


pedro.alves@paanalytics.net