



Instituto Politécnico  
de Viana do Castelo

## LICENCIATURA EM ENGENHARIA INFORMÁTICA

### Inteligência Artificial

Ferramenta Python & Exploração de  
Dados usando Kaggle Notebook

Jorge Ribeiro

• [jribeiro@estg.ipvc.pt](mailto:jribeiro@estg.ipvc.pt)



1. Introdução e Objetivos
2. Linguagem Python
3. Kaggle
4. Análise e Exploração de Estatísticas
5. Bibliografia



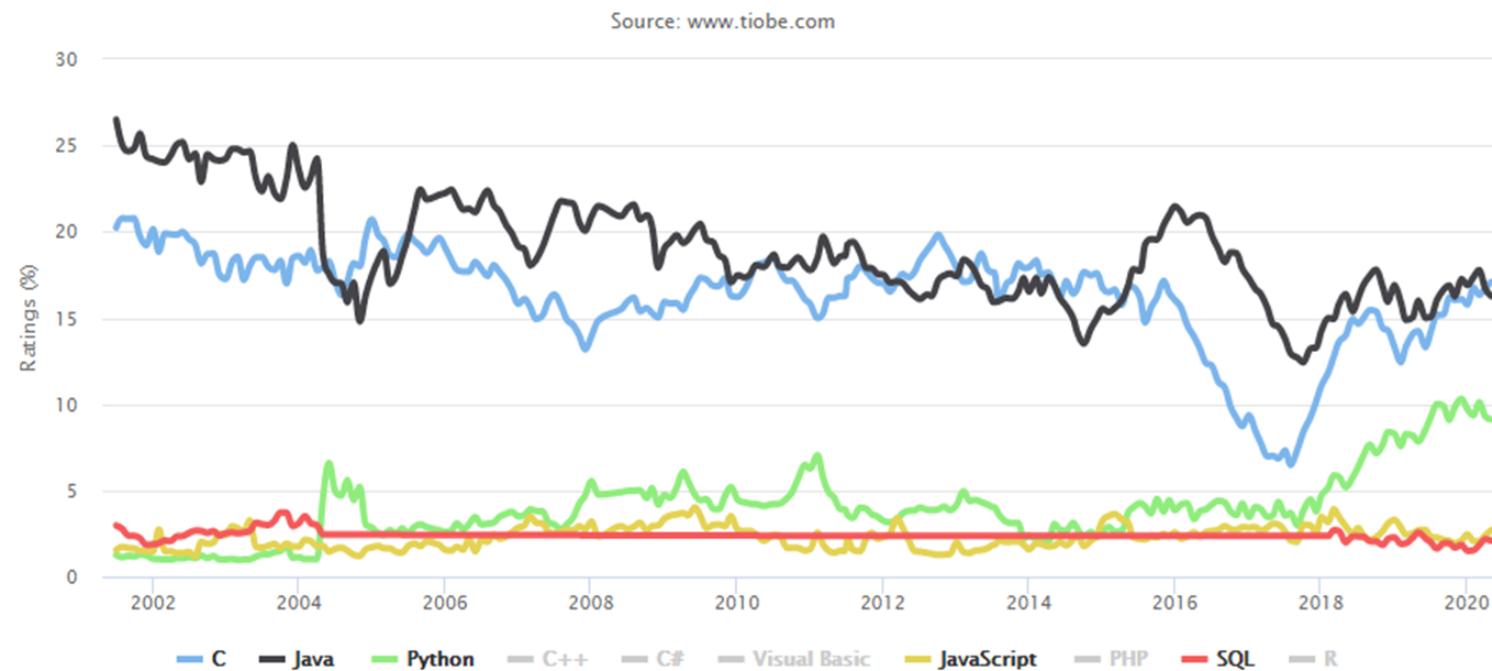
Este trabalho tem com objetivo a exploração da linguagem Python. A linguagem Python é usada para todo o tipo de software contudo é fortemente usada na área da Inteligência Artificial. É dentro desta área que irá ser feito esta apresentação. Durante o mesmo será realizado inúmeros cálculos estatísticos à nossa fonte de dados de modo a demonstrar todo o potencial desta linguagem.

Utilizar-se-á o Kaggle Kernels que é um ambiente na cloud onde se pode executar os nossos notebooks. É uma ferramenta totalmente gratuita. Unicamente precisamos de criar uma conta Kaggle para criar e editar notebooks, cada conta vem com 1 instância de GPU e até 30 horas / semana de uso, não se precisa de configurar nenhuma infraestrutura em nuvem, e tem uma comunidade incrível para partilhar o nosso trabalho.

Contudo mais à frente iremos ver com mais detalhe esta ferramenta.

# ■ Linguagem Python

Python é uma linguagem de propósito geral, ou seja, pode ser utilizada para as mais diversas aplicações. É gratuita Open Source e foi projetada tendo como um dos principais objetivos ser de fácil leitura e utilização. Definindo com termos mais técnicos, Python é interpretada, orientada a objetos, funcional, tipada, imperativa e de script.



A linguagem Python foi desenvolvida sob 19 princípios, são eles:

1. Bonito é melhor do que feio.
2. Explícito é melhor do que implícito.
3. Simples é melhor do que complexo.
4. Complexo é melhor do que complicado.
5. Horizontal é melhor do que aninhado.
6. Esparsos é melhor que denso.
7. A legibilidade conta.
8. Casos especiais não são especiais o suficiente para quebrar as regras.
9. Porém, a praticidade supera a pureza.
10. Os erros nunca devem passar silenciosamente.
11. A menos que sejam explicitamente silenciados.
12. Diante da ambiguidade, recuse a tentação de adivinhar.
13. Deve haver uma, e de preferência apenas uma, forma óbvia de se fazer algo.
14. Embora essa forma possa não ser óbvia no início, a menos que você seja holandês.
15. Agora é melhor do que nunca.
16. Mas “nunca” é melhor do que “imediatamente agora”.
17. Se a implementação é difícil de explicar, é uma má ideia.
18. Se a implementação for fácil de explicar, pode ser uma boa ideia.
19. Namespaces são uma ótima ideia – vamos fazer mais disso!

## O que é possível fazer com Python?

Agora que já vimos os benefícios da linguagem, chegou a hora de ter uma visão mais abrangente sobre o que é possível fazer com Python:

- Automatizar tarefas repetitivas, criando códigos que interagem com seu sistema operacional;
- Varrer a internet (web scraping) navegando por sites, coletando, organizando e salvando informações;
- Monitorar e minerar redes sociais, conectando-se diretamente via APIs que facilitam a extração de dados;
- Construir um site ou uma aplicação para a web;
- Construir um aplicativo mobile;
- Criar aplicações em blockchain (diversos projetos descentralizados já possuem suporte para Python);
- Criar jogos;
- Manipular grandes conjuntos de textos com as mais avançadas tecnologias (processamento de linguagem natural);
- Criar gráficos para BI (Business Intelligence);
- Criar ferramentas de Analytics para tomadas de decisão;
- Manipular dados de forma avançada, com todos os recursos que um cientista de dados poderia necessitar;
- Rodar algoritmos de machine learning, tendo acesso a tudo que há de mais avançado na área;
- Criar aplicações de inteligência artificial, utilizando deep learning, reinforcement learning, entre outros;
- Trabalhar com Big Data;
- Realizar trading automatizado em bolsa de valores;
- Fazer pesquisa científica e computação numérica, tendo bibliotecas alternativas ao software Matlab;
- Utilizar funções e módulos prontos para engenharia, geologia, climatologia, entre outras áreas;
- Programar microcontroladores e robôs.

## Interpretador Python: qual a melhor IDE

A linguagem Python, depois de instalada em seu computador, pode ser executada diretamente via terminal, sem a necessidade de nenhum software específico.

Mas para tornar o ambiente mais atrativo visualmente e facilitar a visualização do código, bem como ter à disposição recursos extras, é bastante útil escrever e compilar os códigos Python em uma IDE (IDE é uma sigla em inglês para “ambiente de desenvolvimento integrado”).

As melhores e mais populares IDEs para programar em Python são:

- Jupyter Notebook
- Pycharm
- VS Code
- Sublime Text
- Atom
- Spyder
- Vim

## Bibliotecas e Frameworks populares do Python

Confira abaixo algumas bibliotecas e frameworks muito utilizados, que vale a pena você conferir e aprender a manipular para dominar funções e recursos específicos:

- Frameworks e bibliotecas para desenvolvimento web: Django, Flask, Pyramid;
- Bibliotecas para desenvolvimento mobile: Kivy, BeeWare;
- Frameworks e bibliotecas para machine learning, inteligência artificial e ciência de dados: [Scikit-learn](#), Tensorflow, Keras, Pytorch, Pandas, Seaborn;
- Bibliotecas para ciência e computação numérica (equivalência ao Matlab): Scipy, [Numpy](#), Matplotlib;
- Bibliotecas para web scraping: Scrapy, BeautifulSoup, Requests, Urllib;
- Bibliotecas para chatbots e processamento de texto: ChatterBot, NLTK, SpaCy;
- **Bibliotecas para manipulação de imagens:** OpenCV, Scikit-image, Pillow;
- **Bibliotecas para games:** Pygame, Kivy, Panda3D, Blender.

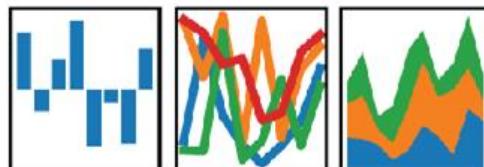
## O que é pandas?

O pandas é uma biblioteca muito usada para análise de dados em Python, que é uma das linguagens de programação mais populares da atualidade. (revisado em 2020)

Uma das principais vantagens de usar Python e o pandas é que você pode manipular uma quantidade relativamente grande de dados (arquivos de até alguns GB) com facilidade, rapidez e com infinitas possibilidades de análise, uma vez que você pode programar qualquer coisa e pegar ideias emprestadas de outras bibliotecas.

# pandas

$$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$$



	BandName	WavelengthMax	WavelengthMin
0	CoastalAerosol	450	430
1	Blue	510	450
2	Green	590	530
3	Red	670	640
4	NearInfrared	880	850
5	ShortWaveInfrared_1	1650	1570
6	ShortWaveInfrared_2	2290	2110
7	Cirrus	1380	1360

Kaggle é uma plataforma bastante conhecida que permite aos seus utilizadores participar em competições de Machine Learning, explorar e publicar conjuntos de dados e também ter acesso Notebooks. É um ótimo ecossistema para se aprender, ligar-se e colaborar com outros Cientistas de Dados para construir modelos Machine Learning.

<https://medium.com/swlh/why-kaggle-kernels-is-the-best-way-to-run-and-share-your-jupyter-notebook-easily-powerfully-and-d16666cd3f84>

Why Kaggle Kernels is the Best Way to Share Your Jupyter Notebook



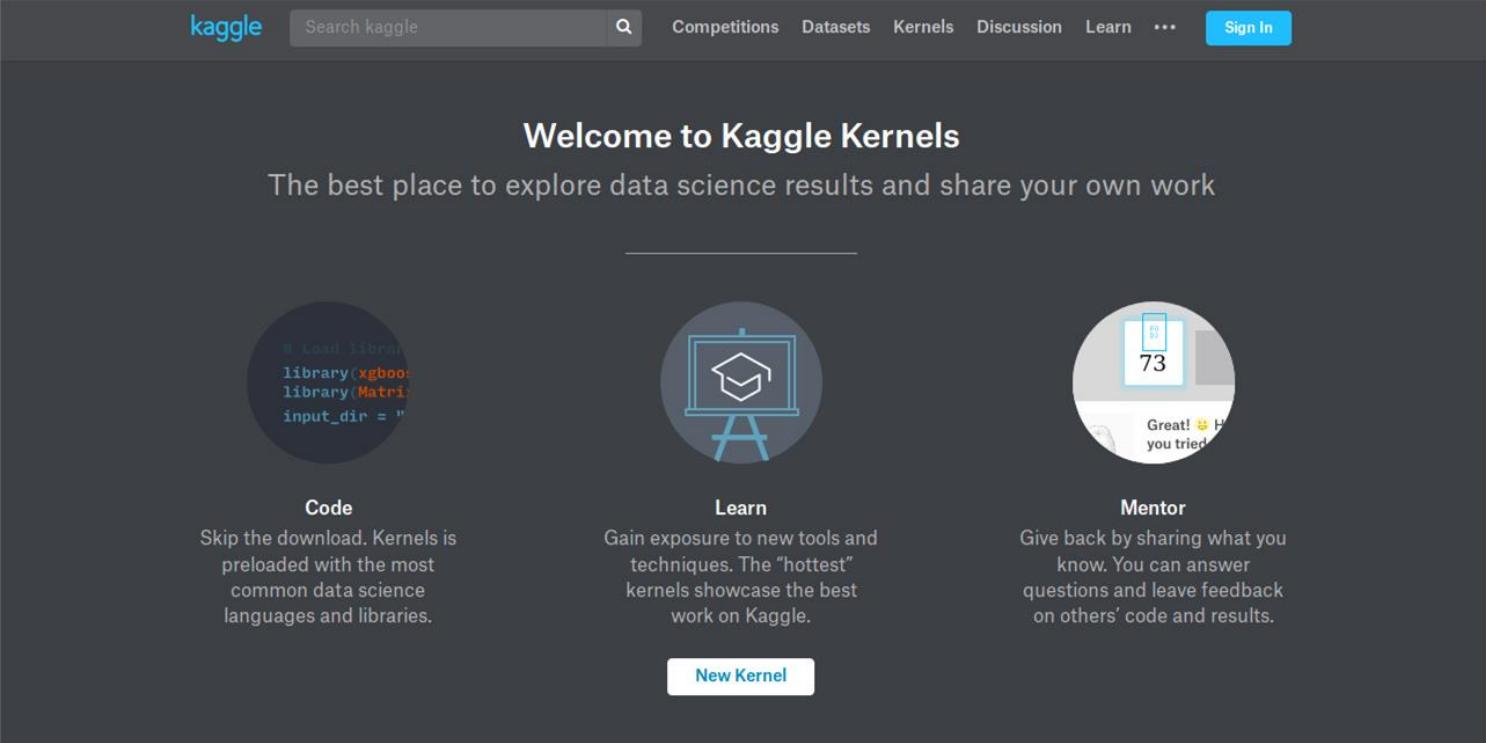
Binh Phan

Follow

May 14, 2020 · 3 min read



Os kernels são “editores de scripts on-line que permitem executar código sem instalar o R ou o Python (e os dados já estão conectados)”. – Em outras palavras, podemos continuar trabalhando no navegador e explorar, modelar, visualizar, ... seus dados sem ter que instalar nada. É assim que funciona o Kaggle



The screenshot shows the 'Welcome to Kaggle Kernels' page. At the top, there's a navigation bar with 'kaggle', a search bar, and links for 'Competitions', 'Datasets', 'Kernels', 'Discussion', 'Learn', '...', and 'Sign In'. Below the header, the text 'Welcome to Kaggle Kernels' and 'The best place to explore data science results and share your own work' is displayed. The page is divided into three main sections:

- Code**: Shows a snippet of R code: '# Load libraries', 'library(xgboost)', 'library(Matrix)', 'input\_dir = '''. It includes a circular icon with a blue gradient.
- Learn**: Shows a blue icon of a whiteboard with a house symbol. It includes a circular icon with a blue gradient.
- Mentor**: Shows a circular icon with a blue gradient containing the number '73'. Below it, the text 'Great! 🎉 Here you tried' is visible.

At the bottom of the section, there's a 'New Kernel' button.

**Exploration and analysis - Auto-MPG**

Running | Restart | Python | Python 3.6.1 | Private | Publish

Input Files

Notebook

Markdown | Code |

**# Welcome to exploration and analysis of the auto mpg data set.**

Welcome to this ipython notebook created for exploration and analysis of the Auto-MPG data-set from UCI Machine Learning Library.

The data-set is fairly standard on kaggle but can be accessed separately from the UCI Machine Learning Repository along with many other interesting data-sets. Check <http://archive.ics.uci.edu/ml/index.php> for more.

This notebook aims primarily to demonstrate use of pandas and seaborn for exploration and visualization of the data-set along with use of scikit learn library to build regression models to predict the Miles Per Gallon(MPG) using the factors provided in the data-set

**## So what is the auto-mpg data set?**

The following description can be found on the UCI Repository page for the data-set (<http://archive.ics.uci.edu/ml/datasets/Auto+MPG>)

- This dataset is a slightly modified version of the dataset provided in the StatLib library. In line with the use by Ross Quinlan (1993) in predicting the attribute "mpg", 8 of the original instances were removed because they had unknown values for the "mpg" attribute. The original dataset is available in the file "auto-mpg.data-original".

"The data concerns city-cycle fuel consumption in miles per gallon, to be predicted in terms of 3 multivalued discrete and 5 continuous attributes." (Quinlan, 1993)

But now we will treat this as a expedition to discover unknown knowledge in the unchartered lands of this dataset.

Let's first ready the equipment we will need for this expedition

```
[1]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy.stats import norm
```

**Featured Dataset**

**Breast Cancer Wisconsin (Diagnostic) Data Set**  
Predict whether the cancer is benign or malignant

UCI Machine Learning • last updated a year ago

Overview | Data | Kernels | Discussion | Activity | Download (48 KB) | New Kernel

Tags: human medicine, small, featured

Top Contributors: Manish Kumar (1st), Luis Bronch (2nd), Buddhini Wijesinghe (3rd)

Kernels: Basic Machine Lear... (71), Breast Cancer Dat... (13 replies)

Discussion: 13 replies

Select Kernel Type: Script | Notebook

Description: Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. The image is of equal width by height, and each pixel is a binary value (a value of 1 if the pixel is light, otherwise 0). This database is a collection of 569 instances which are labeled either "benign" or "malignant". The attribute names and their descriptions are as follows:

- Python, R, RMarkdown
- Describes character of the cell nuclei in 10 image samples obtained by FNA.
- Linearly inseparable
- Described in [K. P. Bennett and D. M. Margineantu, "Linear Separability and Kernel Functions," in *Linear Separability*, Kluwer, Dordrecht, 1999].
- This database is a collection of 569 instances which are labeled either "benign" or "malignant".
- The database is taken from the UCI Machine Learning Repository.
- Also can be found [https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+\(Diagnostic\)](https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)).
- Attribute Information: (1) ID number (2) Diagnosis (M= malignant, B= benign) (3) Radius (mean of distances from center to points on the perimeter) (4) texture (standard deviation of gray-scale values) (5) perimeter (sum of lengths of the boundary) (6) area (area of the image) (7) smoothness (local variation in radius lengths) (8) compactness (perimeter^2 / area - 1.0) (9) concavity (severity of concave portions of the contour) (10) concave points (number of concave portions of the contour) (11) symmetry (fractal dimension ("coastline approximation") - 1) (12) fractal dimension ("coastline approximation") - 1
- The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. For instance, field 3 is Mean Radius, field 13 is Radius SE, field 23 is Worst Radius.
- All feature values are recoded with four significant digits.
- Missing attribute values: none
- Class distribution: 357 benign, 212 malignant

Released Under CC BY-NC-SA 4.0



kaggle.com/code

Apps Trading | DEGIRO Bitcoin Exchange |... Coin Market TrainingPeaks Moodle@IPVC - 2... Trello Website list

Sign In Register

**kaggle**

Search

**Code**

Explore and run machine learning code with Kaggle Notebooks. Find help in the Documentation.

+ New Notebook



Search public notebooks Filters

Python R Beginner NLP Finance Random Forest GPU TPU Competition notebook

**Trending**

See all (257)

**Formula 1 70th Anniversary**  
Updated 21 hours ago  
Formula 1 World Championship (1950 - 2021)+1

**Soundscape Visualisations**  
Updated 7 hours ago  
BirdCLEF 2021 - Birdcall Identification

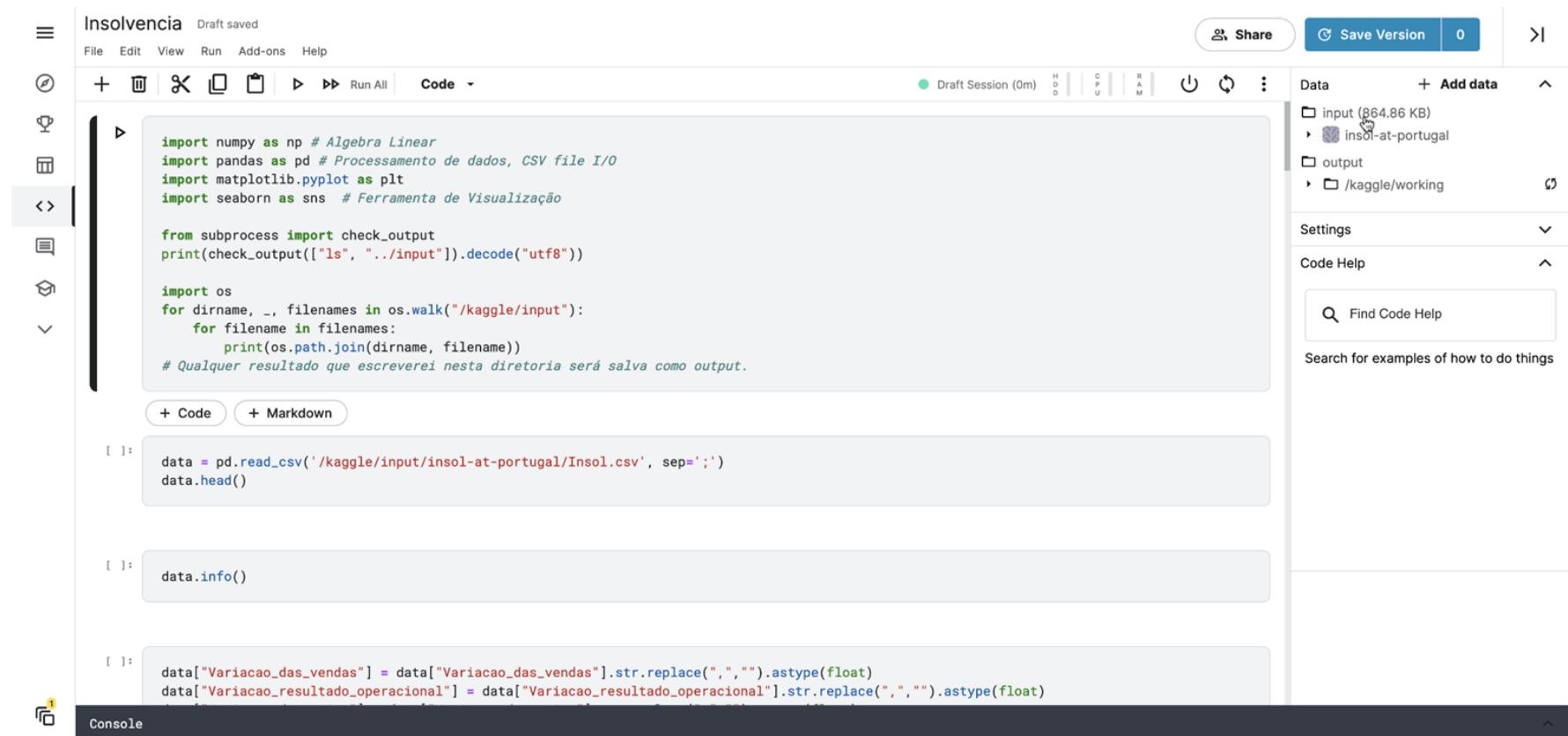
**Plotly for Beginners: [Polar Charts / Image]**  
Updated a day ago  
FIFA 19 complete player dataset+2

**TPS-April SkLearn, PyCaret for Newbies**  
Updated a day ago  
Tabular Playground Series - Apr 2021

View Active Events

<https://www.kaggle.com/code>

## O nosso Notebook



The screenshot shows a Kaggle Notebook interface with the following details:

- Title:** Insolvencia (Draft saved)
- Toolbar:** File, Edit, View, Run, Add-ons, Help.
- Code Editor:** Contains Python code for data processing:

```
import numpy as np # Algebra Linear
import pandas as pd # Processamento de dados, CSV file I/O
import matplotlib.pyplot as plt
import seaborn as sns # Ferramenta de Visualização

from subprocess import check_output
print(check_output(["ls", "../input"]).decode("utf8"))

import os
for dirname, _, filenames in os.walk("/kaggle/input"):
    for filename in filenames:
        print(os.path.join(dirname, filename))
# Qualquer resultado que escreverei nesta diretoria será salva como output.
```
- Output Cells:**
  - [ ]: data = pd.read\_csv('/kaggle/input/insol-at-portugal/Insol.csv', sep=';')
data.head()
  - [ ]: data.info()
  - [ ]: data["Variacao\_das\_vendas"] = data["Variacao\_das\_vendas"].str.replace(",","").astype(float)
data["Variacao\_resultado\_operacional"] = data["Variacao\_resultado\_operacional"].str.replace(",","").astype(float)
- Right Sidebar:**
  - Data panel: input (864.86 KB), insol-at-portugal, output, /kaggle/working.
  - Settings and Code Help sections.
  - Find Code Help and Search for examples of how to do things.
- Console Tab:** Shows a small icon with a '1'.

<https://www.kaggle.com/hldergonalves/insolvencia>

# Análise e Exploração dos Dados – Importar Bibliotecas

[1]:

```
import numpy as np # Algebra Linear
import pandas as pd # Processamento de dados, CSV file I/O
import matplotlib.pyplot as plt
import seaborn as sns # Ferramenta de Visualização

from subprocess import check_output
print(check_output(["ls", "../input"]).decode("utf8"))

import os
for dirname, _, filenames in os.walk("/kaggle/input"):
    for filename in filenames:
        print(os.path.join(dirname, filename))
# Qualquer resultado que escreverei nesta diretoria será salvo como output.
```

insol-at-portugal

/kaggle/input/insol-at-portugal/Insol.csv

## Análise e Exploração dos Dados – Importar Insol.csv

[2]:

```
data = pd.read_csv(' /kaggle/input/insol-at-portugal/Insol.csv', sep=';')
data.head()
```

[2]:

	Distrito	NACE1	NACE2	Numero_Empregados	Num_anos_disponiveis_base	Variacao_das_vendas	Variacao_resultado_operacional	Variacao_do_activo	Variacao_dos_capitais_proprios
0	1	14	141	225	9	-0,055115655	-12,18344284	-0,081616494	-0,353879041
1	1	13	139	28	10	-0,163781044	-7,685481008	-0,120589583	0,002009229
2	5	14	141	80	9	-0,193743196	-1,486740076	-0,118888347	0,055864361
3	5	14	141	175	8	-0,156786571	-1,366234018	-0,122806701	-0,109388014
4	1	13	139	140	10	-0,059913255	-0,742674271	-0,019611869	0,002116005

5 rows × 10 columns

Data + Add data ^

- input (864.86 KB)
  - insol-at-portugal
- output
  - /kaggle/working

## Informação dos Dados Importados

[ 3 ]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2738 entries, 0 to 2737
Data columns (total 33 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Distrito          2738 non-null    int64  
 1   NACE1             2738 non-null    int64  
 2   NACE2             2738 non-null    int64  
 3   Numero_Empregados 2738 non-null    int64  
 4   Num_anos_disponiveis_base 2738 non-null    int64  
 5   Variacao_das_vendas 2738 non-null    object  
 6   Variacao_resultado_operacional 2738 non-null    object  
 7   Variacao_do_ativo 2738 non-null    object  
 8   Variacao_dos_capitais_proprios 2738 non-null    object  
 9   Variacao_do_ativo_circulante 2738 non-null    object  
 10  Variacao_das_existencias 2738 non-null    object  
 11  Variacao_do_imobilizado 2738 non-null    object  
 12  Fundo_de_maneio 2738 non-null    int64  
 13  REPV              2738 non-null    object  
 14  Prazo_medio_de_recebimento 2738 non-null    object  
 15  Liquidez_geral 2738 non-null    object  
 16  Liquidez_reduzida 2738 non-null    object  
 17  Solvabilidade 2738 non-null    object  
 18  Autonomia_financeira 2738 non-null    object  
 19  Endividamento 2738 non-null    object  
 20  Estrutura_financeira 2738 non-null    object  
 21  Passivo_de_curto_prazo_a_dividir_passivo_total 2738 non-null    object  
 22  Custos_dos_encargos_financeiros_dividir_resultado_operacional 2738 non-null    object  
 23  Rendibilidade_operacional_vendas 2738 non-null    object  
 24  Rendibilidade_liquida_das_vendas 2738 non-null    object  
 25  Rendibilidade_do_ativo 2738 non-null    object  
 26  Rendibilidade_capitais_proprios 2738 non-null    object  
 27  Passivo_curto_prazo_dividir_vendas 2738 non-null    object  
 28  Peso_das_amortizacoes_dividir_vendas 2738 non-null    object  
 29  Peso_encargos_financeiros_dividir_vendas 2738 non-null    object  
 30  Produtividade_por_trabalhador 2738 non-null    object  
 31  Certificacao_legal_contas 2738 non-null    int64  
 32  Insolvencia 2738 non-null    int64  
dtypes: int64(8), object(25)
memory usage: 706.0+ KB
```

# Informação dos Dados Importados

**Distrito** – 1 – Braga; 2 – Bragança; 3 – Viana do Castelo; 4 – Vila Real e 5 – Porto

**NACE1** – 13 – Fabricação de textéis e 14 – Indústria do vestuário

**NACE2** – 131 - Preparação e fiação de fibras textéis, 132 - Tecelagem de textéis, 133 - Acabamento de textéis, 139 - Fabricação de outros textéis, 141 - Confeção de artigos de vestuário excepto artigos de peles com pelo, 142 - Fabricação de artigos de pele com pelo e 143 - Fabricação de artigos de malha

**Certificacao\_legal\_contas** – 0 – Não e 1 – Sim

**Insolvencia** – 0 – Não e 1 – Sim

# Conversão das colunas do Tipo ‘Object’ para Float para permitir realizar Cálculos Estatísticos

```
data["Variacao_das_vendas"] = data["Variacao_das_vendas"].str.replace(",","").astype(float)
data["Variacao_resultado_operacional"] = data["Variacao_resultado_operacional"].str.replace(",","").astype(float)
data["Variacao_do_ativo"] = data["Variacao_do_ativo"].str.replace(",","").astype(float)
data["Variacao_dos_capitais_proprios"] = data["Variacao_dos_capitais_proprios"].str.replace(",","").astype(float)
data["Variacao_das_existencias"] = data["Variacao_das_existencias"].str.replace(",","").astype(float)
data["Variacao_do_imobilizado"] = data["Variacao_do_imobilizado"].str.replace(",","").astype(float)
data["REPV"] = data["REPV"].str.replace(",","").astype(float)
data["Prazo_medio_de_recebimento"] = data["Prazo_medio_de_recebimento"].str.replace(",","").astype(float)
data["Liquidez_geral"] = data["Liquidez_geral"].str.replace(",","").astype(float)
data["Liquidez_reduzida"] = data["Liquidez_reduzida"].str.replace(",","").astype(float)
data["Solvabilidade"] = data["Solvabilidade"].str.replace(",","").astype(float)
data["Autonomia_financeira"] = data["Autonomia_financeira"].str.replace(",","").astype(float)
data["Endividamento"] = data["Endividamento"].str.replace(",","").astype(float)
data["Estrutura_financeira"] = data["Estrutura_financeira"].str.replace(",","").astype(float)
data["Passivo_de_curto_prazo_a_dividir_passivo_total"] = data["Passivo_de_curto_prazo_a_dividir_passivo_total"].str.replace(",","").astype(float)
data["Custos_dos_encargos_financeiros_dividir_resultado_operacional"] = data["Custos_dos_encargos_financeiros_dividir_resultado_operacional"].str.replace(",","").astype(float)
data["Rendibilidade_operacional_vendas"] = data["Rendibilidade_operacional_vendas"].str.replace(",","").astype(float)
data["Rendibilidade_liquida_das_vendas"] = data["Rendibilidade_liquida_das_vendas"].str.replace(",","").astype(float)
data["Rendibilidade_do_ativo"] = data["Rendibilidade_do_ativo"].str.replace(",","").astype(float)
data["Rendibilidade_capitais_proprios"] = data["Rendibilidade_capitais_proprios"].str.replace(",","").astype(float)
data["Passivo_curto_prazo_dividir_vendas"] = data["Passivo_curto_prazo_dividir_vendas"].str.replace(",","").astype(float)
data["Peso_das_amortizacoes_dividir_vendas"] = data["Peso_das_amortizacoes_dividir_vendas"].str.replace(",","").astype(float)
data["Peso_encargos_financeiros_dividir_vendas"] = data["Peso_encargos_financeiros_dividir_vendas"].str.replace(",","").astype(float)
data["Produtividade_por_trabalhador"] = data["Produtividade_por_trabalhador"].str.replace(",","").astype(float)
data["Variacao_do_ativo_circulante"] = data["Variacao_do_ativo_circulante"].str.replace(",","").astype(float)
```

## Conversão das colunas do Tipo ‘Object’ para Float para permitir realizar Cálculos Estatísticos

[5]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2738 entries, 0 to 2737
Data columns (total 33 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Distrito          2738 non-null    int64  
 1   NACE1             2738 non-null    int64  
 2   NACE2             2738 non-null    int64  
 3   Numero_Empregados 2738 non-null    int64  
 4   Num_anos_disponiveis_base 2738 non-null    int64  
 5   Variacao_das_vendas 2738 non-null    float64 
 6   Variacao_resultado_operacional 2738 non-null    float64 
 7   Variacao_do_ativo 2738 non-null    float64 
 8   Variacao_dos_capitais_proprios 2738 non-null    float64 
 9   Variacao_do_ativo_circulante 2738 non-null    float64 
 10  Variacao_das_existencias 2738 non-null    float64 
 11  Variacao_do_imobilizado 2738 non-null    float64 
 12  Fundo_de_maneio 2738 non-null    int64  
 13  REPV              2738 non-null    float64 
 14  Prazo_medio_de_recebimento 2738 non-null    float64 
 15  Liquidez_geral 2738 non-null    float64 
 16  Liquidez_reduzida 2738 non-null    float64 
 17  Solvabilidade 2738 non-null    float64 
 18  Autonomia_financeira 2738 non-null    float64 
 19  Endividamento 2738 non-null    float64 
 20  Estrutura_financeira 2738 non-null    float64 
 21  Passivo_de_curto_prazo_a_dividir_passivo_total 2738 non-null    float64 
 22  Custos_dos_encargos_financeiros_dividir_resultado_operacional 2738 non-null    float64 
 23  Rendibilidade_operacional_vendas 2738 non-null    float64 
 24  Rendibilidade_liquida_das_vendas 2738 non-null    float64 
 25  Rendibilidade_do_ativo 2738 non-null    float64 
 26  Rendibilidade_capitais_proprios 2738 non-null    float64 
 27  Passivo_curto_prazo_dividir_vendas 2738 non-null    float64 
 28  Peso_das_amortizacoes_dividir_vendas 2738 non-null    float64 
 29  Peso_encargos_financeiros_dividir_vendas 2738 non-null    float64 
 30  Produtividade_por_trabalhador 2738 non-null    float64 
 31  Certificacao_legal_contas 2738 non-null    int64  
 32  Insolvencia 2738 non-null    int64  
dtypes: float64(25), int64(8)
memory usage: 706.0 KB
```

## Correlação de Dados

### Coeficiente de Correlação de Pearson

Também chamado de “coeficiente de correlação produto-momento” ou simplesmente de “ $\rho$  de Pearson” mede o grau da correlação (e a direção dessa correlação — se positiva ou negativa) entre duas variáveis. Este coeficiente, normalmente representado por  $\rho$  assume apenas valores entre -1 e 1.

A formula desenvolvida por Karl Pearson, há mais de 120 anos, continua a ser a mais utilizada para o cálculo da correlação.

Calcula-se o coeficiente de correlação de Pearson pela seguinte fórmula:

$$\rho = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\left[ \sum_{i=1}^n (x_i - \bar{x})^2 \right] \left[ \sum_{i=1}^n (y_i - \bar{y})^2 \right]}}$$

A correlação de Pearson mede a associação linear entre **variáveis contínuas**. É o valor que indica o quanto a relação entre as variáveis pode ser descrita por uma reta.

## Correlação de Dados

### Interpretando o valor de $\rho$

- 0.9 a 1 positivo ou negativo indica uma correlação muito forte.
- 0.7 a 0.9 positivo ou negativo indica uma correlação forte.
- 0.5 a 0.7 positivo ou negativo indica uma correlação moderada.
- 0.3 a 0.5 positivo ou negativo indica uma correlação fraca.
- 0 a 0.3 positivo ou negativo indica uma correlação desprezível.

Existem várias possibilidades de interpretação da correlação. Pode-se considerar a inclinação da reta que representa a relação entre as variáveis, pode-se considerar as séries de valores como vetores, e o  $\rho$ , em uma interpretação geométrica, representaria o cosseno do ângulo formado entre os vetores, etc. Algumas informações interessantes podem ser encontradas no artigo “[Thirteen ways to look at the correlation coefficient](#)”.

## Correlação de Dados

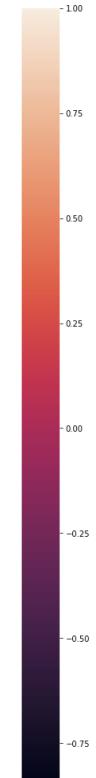
[ 6 ]:  
data.corr()

	Distrito	NACE1	NACE2	Numero_Empregados	Num_anos_disponiveis_base	Variacao_das_vendas	Variacao_resultado_operacional
Distrito	1.000000	0.037939	0.037091	0.041098	0.062187	0.019131	0.0107
NACE1	0.037939	1.000000	0.740732	-0.027026	-0.107619	0.036619	0.0016
NACE2	0.037091	0.740732	1.000000	-0.091921	-0.138888	0.057968	0.0156
Numero_Empregados	0.041098	-0.027026	-0.091921	1.000000	0.350247	-0.008386	0.0234
Num_anos_disponiveis_base	0.062187	-0.107619	-0.138888	0.350247	1.000000	-0.084069	0.0346
Variacao_das_vendas	0.019131	0.036619	0.057968	-0.008386	-0.084069	1.000000	0.2903
Variacao_resultado_operacional	0.010782	0.001686	0.015698	0.023409	0.034602	0.290335	1.0000
Variacao_do_ativo	0.047157	0.016935	0.028938	0.015393	-0.078099	0.559880	0.2023
Variacao_dos_capitais_proprios	0.010062	-0.018725	0.003964	-0.004091	0.011201	0.168846	0.2758
Variacao_do_ativo_circulante	0.021154	0.026027	0.018839	0.013656	-0.071728	0.353910	0.2203
Variacao_das_existencias	0.003055	0.016647	0.011263	-0.012314	-0.080497	0.050727	0.0276
Variacao_do_imobilizado	0.032348	-0.027162	0.001465	0.000418	-0.009735	0.084247	0.0185
Fundo_de_maneio	0.014417	-0.092560	-0.059891	0.625267	0.169693	-0.009067	0.0082
REPV	-0.012831	0.040679	0.042099	0.035598	0.019746	0.042391	-0.0242
Prazo_medio_de_recebimento	0.013277	0.101487	0.044559	0.003294	-0.011140	-0.003470	-0.0102
Liquidez_geral	0.028792	-0.045093	-0.023296	0.032797	0.113687	-0.030926	0.0008
Liquidez_reduzida	-0.043541	-0.049664	-0.035347	0.012015	0.057471	-0.031808	-0.0177
Solvabilidade	-0.041281	-0.047932	-0.035403	0.053188	0.126892	-0.016807	0.0257
Autonomia_financeira	-0.032169	-0.066863	-0.066803	0.067769	0.143857	0.022031	0.0967
Endividamento	0.031467	0.036635	0.050356	-0.085084	-0.140283	-0.021254	-0.0975
Estrutura_financeira	0.027322	-0.013030	0.005135	0.039168	0.054555	0.066010	0.1304
Passivo_de_curto_prazo_a_dividir_passivo_total	-0.027886	-0.089991	-0.083434	0.148825	0.189942	0.019554	0.0138
Custos_dos_encargos_financeiros_dividir_resultado_operacional	-0.022257	-0.003285	0.019673	0.001319	0.031020	0.202210	0.4342
Rendibilidade_operacional_vendas	-0.003200	0.015983	0.013421	0.045638	0.069670	0.112831	0.2265
Rendibilidade_liquida_das_vendas	-0.027638	0.031624	0.023319	0.052673	0.055564	0.190195	0.2707
Rendibilidade_do_ativo	-0.010382	-0.046734	-0.033311	0.041195	0.067369	0.151391	0.2852
Rendibilidade_capitais_proprios	-0.022366	0.019480	0.032749	-0.023369	0.001680	0.100414	0.2019

# Análise e Exploração de Estatísticas

## Correlação de Dados

```
f,ax = plt.subplots(figsize=(18, 18))
sns.heatmap(data.corr(), annot=True, linewidths=.5, fmt= '.1f', ax=ax)
plt.show()
```



# Variáveis a Analisar

[ 9 ]:

```
data.columns
```

```
[ 9 ]: Index(['Distrito', 'NACE1', 'NACE2', 'Numero_Empregados',
       'Num_anos_disponiveis_base', 'Variacao_das_vendas',
       'Variacao_resultado_operacional', 'Variacao_do_activo',
       'Variacao_dos_capitais_proprios', 'Variacao_do_ativo_circulante',
       'Variacao_das_existencias', 'Variacao_do_imobilizado',
       'Fundo_de_maneio', 'REPV', 'Prazo_medio_de_recebimento',
       'Liquidez_geral', 'Liquidez_reduzida', 'Solvabilidade',
       'Autonomia_financeira', 'Endividamento', 'Estrutura_financeira',
       'Passivo_de_curto_prazo_a_dividir_passivo_total',
       'Custos_dos_encargos_financeiros_dividir_resultado_operacional',
       'Rendibilidade_operacional_vendas', 'Rendibilidade_liquida_das_vendas',
       'Rendibilidade_do_ativo', 'Rendibilidade_capitais_proprios',
       'Passivo_curto_prazo_dividir_vendas',
       'Peso_das_amortizacoes_dividir_vendas',
       'Peso_encargos_financeiros_dividir_vendas',
       'Produtividade_por_trabalhador', 'Certificacao_legal_contas',
       'Insolvencia'],
      dtype='object')
```

n - número de elementos da amostra;

[16]:

```
# Número de elementos da amostra
data[data.columns[0]].count() # (slowest, but avoids counting NaN values in the first column)
```

[16... 2738

[17]:

```
# Outra forma
print(f'Número de linhas      = {len(data)}')
print(f'Número de colunas = {len(data.columns)}')
```

Número de linhas = 2738  
Número de colunas = 33

**Mínimo (min) - valor menor da amostra;**

**Máximo (max) - valor maior da amostra;**

[18]:

```
# Para encontrar o valor mais baixo vamos percorrer o nosso CSV

row_count = data.shape[0]
column_list = []
for i in range(row_count):
    item = data.at[i, data.columns.values[0]] # Primeira Coluna
    column_list.append(float(item))           #parse float e adicionar à lista

print(max(column_list))      #Print - Distrito Número Máximo
print(min(column_list))      #Print - Distrito Número Minimo
```

5.0  
1.0

## Moda - valor mais comum da amostra;

### Mode

**Mode** is the number occurring most often in the dataset.

- It is only meaningful if we have many repeated values in our dataset
- If no value is repeated, there is **no mode**
- A dataset can have **one mode, multiple modes or no mode**.

```
9]:  
      # Moda da coluna NACE2  
      data['NACE2'].mode()
```

```
9... 0    141  
      dtype: int64
```

## Média - valor esperado de cada atributo;

### Mean

Mean (usually referred to Arithmetic Mean, also called Average) is calculated as sum of all numbers in the dataset and dividing by the total number of values

### Arithmetic Mean

$$\text{Arithmetic mean} = \frac{\text{Sum of all numbers}}{\text{No. of values in the set}} \quad \text{or}$$

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$

Arithmetic mean of our data is calculated as,

```
mean = (35 + 15 + 97 + 17 + ...) / 315
```

```
]# Média do Número de Empregados
data['Número_Empregados'].mean()
```

```
... 29.096420745069395
```

**Mediana - valor central da amostra (ordenar valores, se n é ímpar a é o valor do meio, se é par, é o valor médio das 2 observações centrais);**

### Median

**Median** is the middle value, when the data is sorted in ascending order. Half of the data points are smaller and half of data points are larger than the median.

```
[1]:  
# Mediana do Fundo_de_maneio  
data['Fundo_de_maneio'].median()
```

```
[1]: 22087.5
```

1º/3º Quartil - 25% dos valores da amostra estão  
abaixo/acima deste valor;

### Interquartile Range (IQR)

Interquartile range or **IQR** is the amount spread in middle 50% of the dataset or the distance between *first Quartile* ( $Q_1$ ) and *third Quartile* ( $Q_3$ )

- First Quartile ( $Q_1$ ) = Median of data points to left of the median in ordered list (25th percentile)
- Second Quartile ( $Q_2$ ) = Median of data (50th percentile)
- Third Quartile ( $Q_3$ ) = Median of data points to right of the median in ordered list (75th percentile)
- $IQR = Q_3 - Q_1$

2]:

```
# Primeiro Quartil, Mediana, 3 Quartil
data['Num_anos_disponiveis_base'].quantile([0.25,0.5,0.75])
```

:2...

```
0.25    4.0
0.50    4.0
0.75    7.0
Name: Num_anos_disponiveis_base, dtype: float64
```

## Variância e Desvio padrão



```
# Passo 1: Calcular Média(μ)
numeroEmpregados = data['Número_Empregados']
media = numeroEmpregados.mean()
print(f'Média = {media}')

# Passo 2: calculate numerator part - sum of (x - Média)
numeroEmpregados_numerator = sum([(x - media) ** 2 for x in numeroEmpregados])

# Passo 3: Calcular a variância
numeroEmpregados_variancia = numeroEmpregados_numerator / len(numeroEmpregados)
print(f'Variância = {numeroEmpregados_variancia}')

# Passo 4: Calcular o Desvio Padrão
numeroEmpregados_desvioPadrao = math.sqrt(numeroEmpregados_variancia)
print(f'Desvio Padrão = {numeroEmpregados_desvioPadrao}')
```

Média = 29.096420745069395  
Variância = 3208.2434422656183  
Desvio Padrão = 56.641358054566616

+ Code

+ Markdown

## Variância e Desvio padrão

### Variance and Standard deviation

Standard deviation and variance measures the spread of a dataset. If the data is spread out largely, standard deviation (and variance) is greater.

In other terms,

- if more data points are closer to the mean, standard deviation is less
- if the data points are further from the mean, standard deviation is more

Formula for variance for **population** is given as,

$$\text{Variance} = \sigma^2 = \frac{\sum_{i=1}^n (x_i - \mu)^2}{n}$$

where,  $\mu$  is the **mean** of the dataset

Standard deviation is just the square root of variance

$$\text{Standard deviation} = \sigma = \sqrt{\frac{\sum_{i=1}^n (x_i - \mu)^2}{n}}$$

#### Note:

For **Sample**, we use  $n - 1$  instead of  $n$ ,  $\bar{x}$  - mean of sample

$$\text{Standard deviation} = S_{sample} = \sqrt{\frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n-1}}$$

Let's take `Insol.csv` dataset.

## Desvio Médio Absoluto

### Desvio Médio Absoluto (MAD)

Mean absolute deviation is the average distance between mean and each data point.

$$\text{Mean absolute deviation (MAD)} = \frac{\sum |x_i - \bar{x}|}{n}$$

Let's calculate mean absolute deviation for Fundo\_de\_Maneio

```
▶ fundo_de_maneio = data['Fundo_de_maneio'].mad()  
print(f'Desvio Médio Absoluto = {fundo_de_maneio}')
```

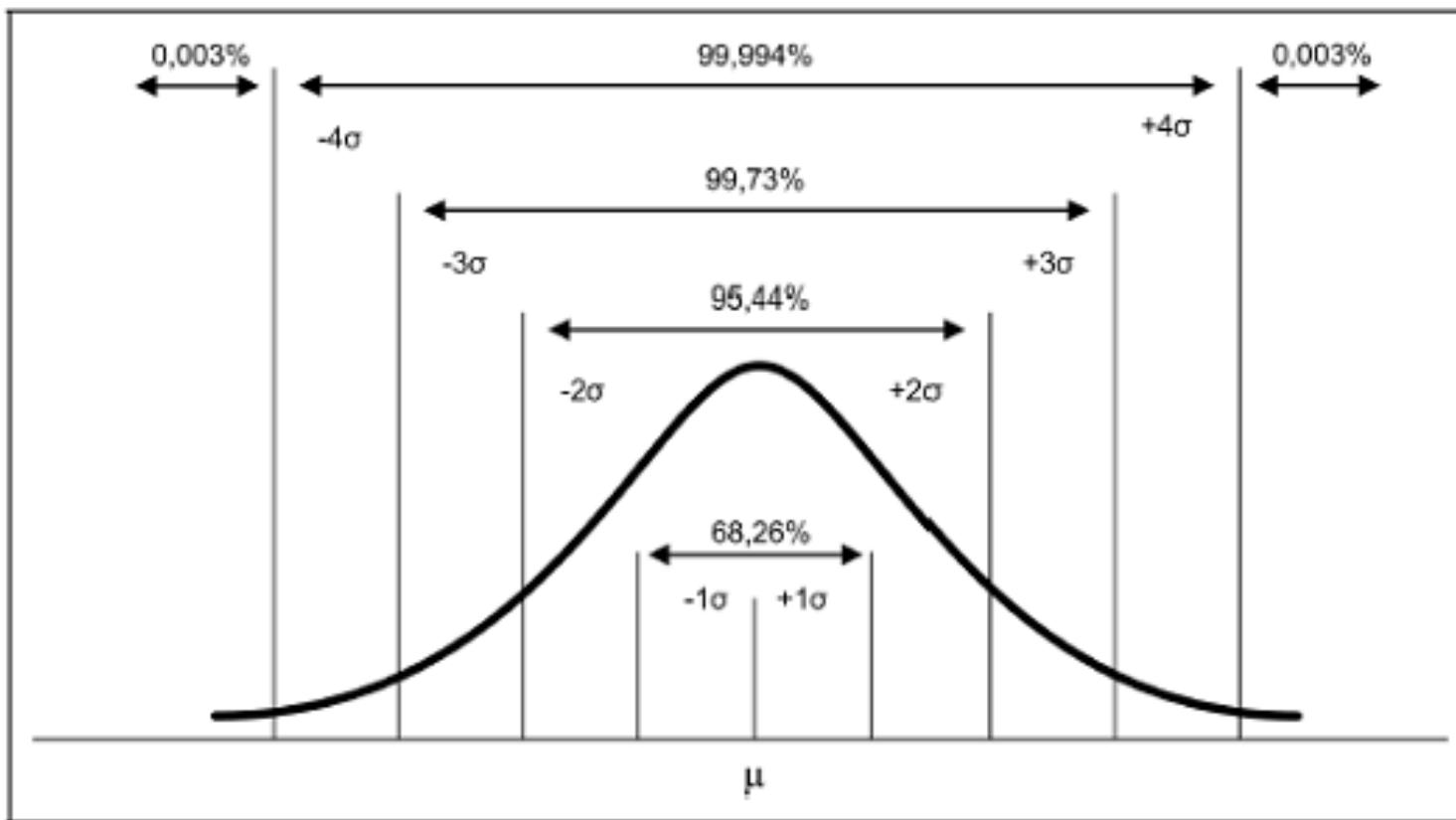
Desvio Médio Absoluto = 538645.1840978444

### Analizar a Distribuição Normal (ou gaussiana);

$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(x - \mu)^2\right\}$$

A funcionalidade para distribuição normal é implementada por argumentos que combinam as letras acima com o termo norm. Por default as funções assumem a distribuição normal padrão  $N(\mu = 0, \sigma^2 = 1)$

### Analizar a Distribuição Normal (ou gaussiana);



## Distribuição Normal

```
▶ fundo_de_maneio = data['Fundo_de_maneio']

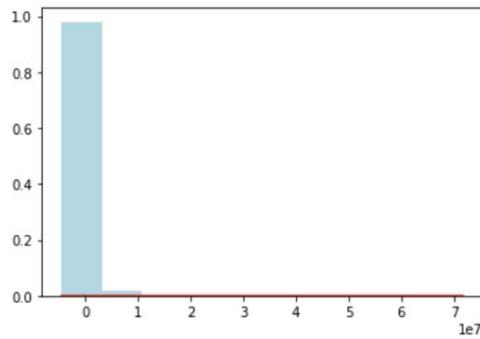
# Get mean (mu) and std (sigma)
fundo_de_maneio_media, fundo_de_maneio_std = fundo_de_maneio.mean(), fundo_de_maneio.std()

# Plot histogram (normalized) - LIGHT-BLUE
fundo_de_maneio.hist(color='lightblue', weights = np.zeros_like(fundo_de_maneio) + 1.0 / fundo_de_maneio.count())

# Plot line graph - RED
fundo_de_maneio.value_counts(sort=False, normalize=True).plot.line(color='red')

# Normal distribution for random points between 1 to 10 with mean, std.
random_data = np.arange(1, 10, 0.001)
pyplot.plot(random_data, stats.norm.pdf(random_data, fundo_de_maneio_media, fundo_de_maneio_std), color='green')
```

14... [<matplotlib.lines.Line2D at 0x7f0c37210650>]



+ Code

+ Markdown

## Bibliografia

- <https://www.kaggle.com/hldergonalves/insolvencia>
- <https://www.kaggle.com/>
- <https://medium.com/fintechexplained/everything-about-python-from-beginner-to-advance-level-227d52ef32d2>
- <https://www.udemy.com/course/python-for-statistical-analysis/>
- <https://www.coursera.org/specializations/statistics-with-python>

o teu • de partida



Instituto Politécnico  
de Viana do Castelo

[www.ipvc.pt](http://www.ipvc.pt)