1 - Introdução

Anaconda é uma distribuição popular de *Python* e *R* que simplifica a gestão de pacotes e ambientes virtuais. Projetada para facilitar o trabalho de cientistas de dados, desenvolvedores e pesquisadores, Anaconda oferece uma plataforma abrangente que inclui não apenas *Python* e *R*, mas também uma ampla gama de bibliotecas e ferramentas para análise de dados, aprendizado de máquina e computação científica.

Os ambientes virtuais são uma parte fundamental dessa plataforma, permitindo isolar projetos e gerenciar dependências de forma independente. Isso garante que diferentes projetos não interfiram entre si, evitando conflitos de versões de pacotes e facilitando a reprodução de resultados. Neste documento, vamos explorar como utilizar o Anaconda para criar e gerenciar ambientes virtuais, proporcionando uma base sólida para o desenvolvimento eficiente e organizado de projetos em Python e outras linguagens suportadas.

2 - O que é Conda?

Conda é uma ferramenta de gestão de pacotes e ambientes virtuais desenvolvida pela Anaconda Inc. Trata-se de uma plataforma de código aberto que facilita a instalação, atualização e gestão de pacotes para diversas linguagens de programação, como *Python*. Além de simplificar a gestão de pacotes, *Conda* permite a criação de ambientes virtuais isolados, onde cada projeto pode ter suas próprias dependências e versões de pacotes.

Uma das grandes vantagens do *Conda* é sua capacidade de resolver automaticamente as dependências entre pacotes, garantindo que todas as bibliotecas necessárias para um projeto sejam instaladas corretamente. Isso é especialmente útil em projetos complexos que dependem de múltiplas bibliotecas e ferramentas.

Conda também é multiplataforma, funcionando em sistemas operacionais como *Windows, macOS e Linux*. Isso torna a ferramenta extremamente versátil e adequada para equipes de desenvolvimento que trabalham em diferentes ambientes.

Além disso, *Conda* não se limita apenas a pacotes de *Python*. Ele pode gerenciar pacotes de outras linguagens e até mesmo bibliotecas de baixo nível, como aquelas usadas para computação científica. Isso faz do *Conda* uma solução abrangente para a gestão de pacotes e ambientes virtuais, proporcionando uma experiência de desenvolvimento mais organizada e eficiente.

3 - Como Instalar o Anaconda

Instalar o *Anaconda* é um processo simples que pode ser realizado em sistemas operacionais como *Windows, macOS e Linux*. Abaixo estão os passos detalhados para realizar a instalação:

Passo 1: Download do Instalador

- Vá até a página oficial de downloads do Anaconda:

https://www.anaconda.com/download/success

- Escolha a Versão Apropriada: Selecione a versão do Anaconda que corresponde ao seu sistema operacional (*Windows, macOS ou Linux*). Escolha entre a versão de 64 bits ou 32 bits, dependendo da arquitetura do seu sistema.

Passo 2: Executar o Instalador

No Windows:

- Execute o ficheiro *.exe* baixado.
- Siga as instruções do instalador, aceitando os termos de licença e escolhendo as opções de instalação padrão. Recomenda-se: instalar o Anaconda apenas para o usuário atual e ***adicionar o Anaconda ao PATH***, caso seja oferecida essa opção.

No macOS:

- Abra o ficheiro *.pkg* baixado.
- Siga as instruções do instalador, aceitando os termos de licença e selecionando as opções de instalação padrão.

No Linux:

- Abra um terminal e navegue até o diretório onde o ficheiro do instalador foi baixado.
- Torne o instalador executável com o comando: ***chmod +x nome_do_instalador.sh***
- Execute o instalador com o comando: ***./nome_do_instalador.sh***
- Siga as instruções no terminal, aceitando os termos de licença e selecionando as opções de instalação padrão.

Passo 3: Verificar a Instalação

Após a instalação, é importante verificar se o *Anaconda* foi instalado corretamente:

- Abra um Terminal ou Prompt de Comando.
- Digite o comando: ***conda --version***

Este comando deve retornar a versão do *Conda* instalada, confirmando que a instalação foi bem-sucedida.

Passo 4: Atualizar Conda (Opcional)

Após a instalação, é recomendável atualizar o Conda para a versão mais recente. Para isso, use o comando: ***conda update conda***

Siga as instruções para concluir a atualização.

4 - Criando um Ambiente Virtual com Conda

Criar um ambiente virtual com *Conda* é um processo simples e direto. Ambientes virtuais permitem isolar projetos uns dos outros, garantindo que cada projeto tenha suas próprias dependências e versões de pacotes. Aqui está um guia passo a passo sobre como criar e gerenciar ambientes virtuais usando *Conda*.

<u>Pelo Terminal ou Prompt de Comando</u>

Passo 1: Abrir o Terminal ou Prompt de Comando

Primeiro, abra o terminal no macOS ou Linux, ou o Prompt de Comando no Windows.

Passo 2: Atualizar o Conda (Opcional)

É uma boa prática garantir que você está usando a versão mais recente do Conda. Para atualizar o Conda, execute: ***conda update conda***

Siga as instruções para completar a atualização.

Passo 3: Criar um Novo Ambiente Virtual

Para criar um novo ambiente virtual, use o comando "*conda create*". Você precisa especificar o nome do ambiente e a versão do Python que deseja usar:

conda create --name <Nome do Ambiente> <python=Versão do Python>

Caso não seja especificado a versão do *Python*, o ambiente será criado com a versão mais recente.

Exemplo:

- Para criar um ambiente chamado *"meu_ambiente"* com *"Python 3.8"*, digite:

  conda create --name meu_ambiente python=3.8

Você pode incluir pacotes adicionais que deseja instalar no momento da criação do ambiente, separando-os com espaços.

conda create --name <Nome do Ambiente> <python=Versão do Python> <pacote1> <pacote2>

Exemplo:

- Para criar um ambiente chamado *"meu_ambiente"* com *"Python 3.8"* com os pacotes de *numpy* e *pandas*, digite:

  conda create --name meu_ambiente python=3.8 numpy pandas

Passo 4: Ativar o Ambiente Virtual

Para começar a usar o ambiente virtual recém-criado, você precisa ativá-lo. No *terminal* ou *prompt* de comando, digite:

conda activate meu_ambiente

Após ativar o ambiente, você verá o nome do ambiente entre parênteses no *prompt*, indicando que o ambiente está ativo.

Passo 5: Instalar Pacotes no Ambiente Ativo

<u>Com o ambiente ativado</u>, você pode instalar pacotes usando o comando *"conda install"*. Por exemplo, para instalar o pacote *matplotlib*, digite: ***conda install matplotlib***

Conda irá gerenciar automaticamente as dependências necessárias.

Passo 6: Listar Ambientes Disponíveis

Para ver uma lista de todos os ambientes virtuais disponíveis em seu sistema, use o comando:

conda env list

Isso exibirá todos os ambientes, juntamente com seus diretórios de instalação.

Passo 7: Desativar o Ambiente Virtual

Para desativar o ambiente atual e retornar ao ambiente base, use o comando:

conda deactivate

Passo 8: Remover um Ambiente Virtual (Opcional)

Se você não precisar mais de um ambiente virtual, pode removê-lo usando o comando:

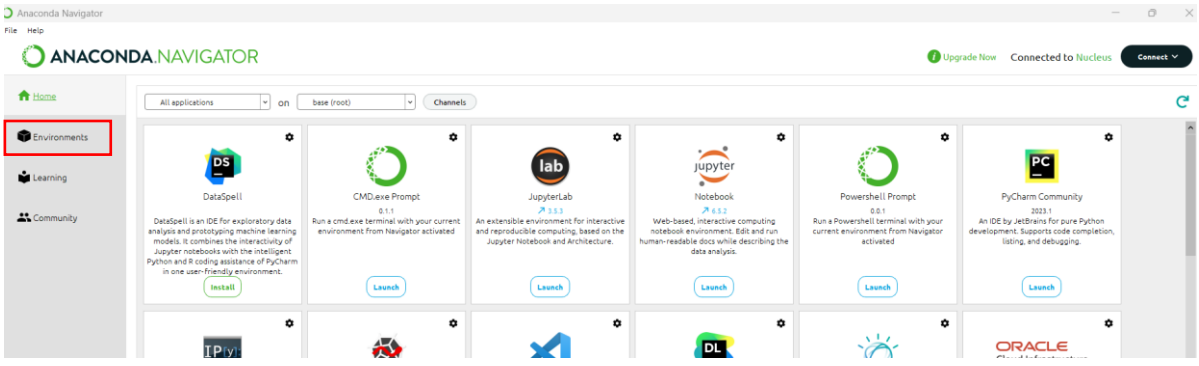
conda remove --name meu_ambiente --all

Isso apagará o ambiente *"meu_ambiente"* e todos os seus pacotes instalados.

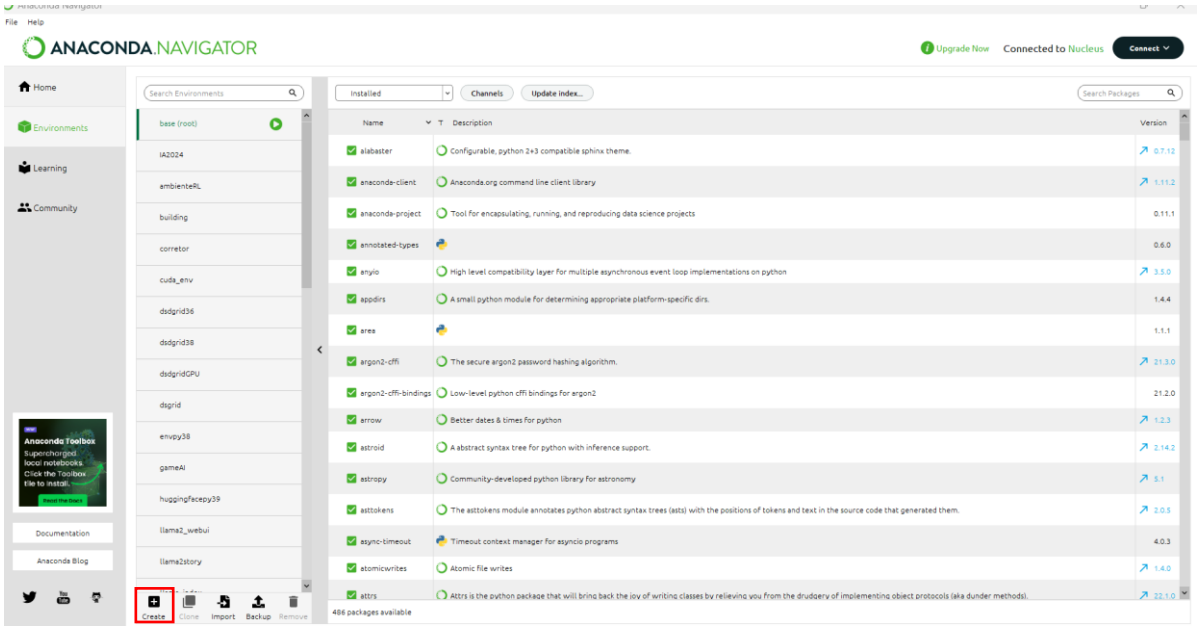
<u>Pelo Anaconda Navigator</u>

Passo 1: Criar um Novo Ambiente Virtual

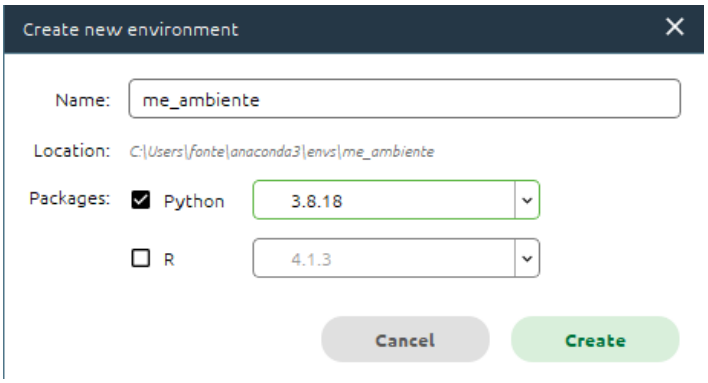
- No menu a esquerda, escolher *"Enviroments"*



- No menu inferior, escolher *"Create"*



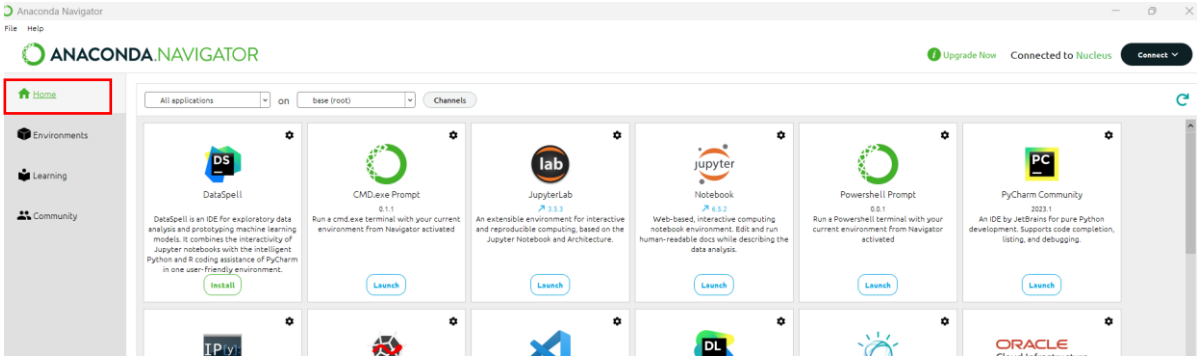
- Na caixa, definir o nome do ambiente e a versão do *Python* a ser utilizada (por exemplo, ambiente de nome me_ambiente com o python 3.8.18).



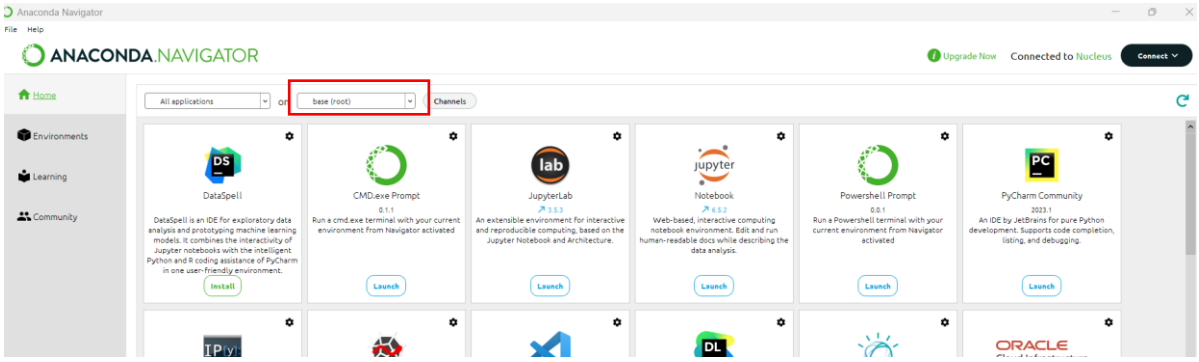
Passo 2: Ativar o Ambiente Virtual

Pode-se seguir o passo 4 do uso pelo Terminal ou:

- Escolher a opção *"Home"* no menu a esquerda



- Escolher o ambiente criado na caixa de rolagem



<u>Com o ambiente ativado</u>, você pode instalar pacotes usando o comando *"conda install"*. Por exemplo, para instalar o pacote *matplotlib*, digite: ***conda install matplotlib***

Passo 3: Instalar Pacotes no Ambiente Ativo

<u>Com o ambiente ativado</u>, você pode instalar pacotes usando o comando *"conda install"*. Por exemplo, para instalar o pacote *matplotlib*, digite: ***conda install matplotlib***

Comando para criar o ambiente virtual: conda create -n ambienteRL python=3.7 pip

Ativando o ambiente: conda activate ambienteRL

Comando para instalar o jupyter notebook: pip install jupyter

Comando para instalar a biblioteca Stable Baselines: pip install stable-baselines3

Comando para instalar o emulador do Super Mario: pip install gym-super-mario-bros

Compilador C++ do Visual Studio: https://visualstudio.microsoft.com/pt-br/downloads/

Comando para instalar a biblioteca OpenCV: pip install opencv-python

Pasta com todos os códigos python e jupyter notebooks utilizados nessa aula: https://didatica.tech/wp-content/uploads/2023/05/codigos_jupyters.zip

Documentação oficial Stable Baselines: https://stable-baselines.readthedocs.io/en/master/

Documentação Gym Super Mario: https://pypi.org/project/gym-super-mario-bros/

https://github.com/Kautenja/gym-super-mario-bros

Para criar um ambiente virtual usando o Virtualenv para isolar as dependências do Python, você pode seguir as seguintes etapas:

1. Certifique-se de ter o Virtualenv instalado. Se você não tiver, pode instalá-lo usando o pip:

   pip install virtualenv

2. Navegue até o diretório onde deseja criar o ambiente virtual. Por exemplo, para criar um ambiente chamado "meu_ambiente", você pode usar o seguinte comando:

```
cd caminho/para/meu_ambiente
```

3. Agora, crie o ambiente virtual executando o comando `virtualenv` seguido pelo nome que deseja dar ao ambiente:

```
virtualenv nome_do_ambiente
```

4. O Virtualenv criará uma estrutura de diretórios para o ambiente virtual. Para ativar o ambiente virtual, dependendo do seu sistema operacional, execute o comando apropriado:

- No Windows:

```
nome_do_ambiente\Scripts\activate
```

- No Linux ou macOS:

```
source nome_do_ambiente/bin/activate
```

5. Após ativar o ambiente virtual, seu prompt de comando deve mostrar o nome do ambiente entre parênteses, indicando que você está trabalhando dentro do ambiente virtual.

6. Agora você pode instalar pacotes Python e trabalhar dentro do ambiente virtual. As dependências instaladas no ambiente virtual não afetarão o ambiente Python global do sistema.

7. Quando terminar de usar o ambiente virtual, você pode desativá-lo executando o comando `deactivate`. Isso retornará seu prompt de comando ao ambiente Python global do sistema.

Lembre-se de que cada ambiente virtual é isolado e pode ter suas próprias dependências e versões do Python. Isso permite que você gerencie as dependências de diferentes projetos de forma independente.
