Ambientes execução Python

- Google Colab “free”
  - Limitações:
    - 12 hr execução contínua
    - 100 GB HD local
    - 12,7 GB RAM
    - Uma sessão ativa por conta
    - 10 GB uploads
    - download de grandes conjuntos de dados ou modelos pode ser restrito.
  - Vantagens:
    - Acesso livre, mas limitado a GPU e TPU
    - Pré-configurado com várias bibliotecas
    - Compartilhamento
    - Integrado Google Drive
    - Controle de versões

Anaconda (miniconda, conda)

- Gerenciador de pacotes Conda
- Instalação fácil ~1 GB
- Bibliotecas pré-instaladas
- Permite que você crie ambientes isolados
- Disponível para Windows, macOS e Linux
- Integração com o Jupyter Notebook
- inclui o Spyder, um poderoso software de desenvolvimento científico em Python
- otimizado para desempenho, especialmente para tarefas com uso intensivo de dados

- Vários GB de armazenamento
- O Conda geralmente é bom no gerenciamento de dependências, mas ainda pode haver conflitos, especialmente em projetos complexos ou ao misturar pacotes Conda e pip.
- Muitos pacotes pré-instalados podem gerar sobrecarga
- Às vezes, os pacotes no Anaconda podem não ser as versões mais recentes, pois são selecionados para estabilidade e compatibilidade

PyCharm da Jet Brains

- interface de usuário amigável e intuitiva
- autocompletar inteligente e sugestões de código
- ferramentas de depuração poderosas
- suporta gerenciamento de dependências através de ferramentas como *pip*, *conda* e *virtualenv*
- integração com sistemas de controle de versão como Git, Mercurial e SVN
- ferramentas de refatoração
- ferramentas para escrever e executar testes unitários, testes de integração e testes de desempenho
- integrado com ferramentas de integração contínua e entrega contínua (Jenkins, Travis CI e GitHub Actions).
- suporte para gerar documentação e comentários de código
- ampla gama de plugins e extensões
- suporta também JavaScript, TypeScript, SQL, e HTML/CSS
- exigente em termos de recursos do sistema
- versão profissional paga
- desempenho diminui em projetos grandes
-

VSCode M$

- leve e rápido
- vasta coleção de extensões
- integração com Git
- ferramentas de depuração robustas
- terminal integrado que suporta várias shells, como Bash, PowerShell e Command Prompt,
- Com extensões: autocompletar inteligente e sugestões de código
- Suporte a várias linguagens
- gratuito e de código aberto (VSCodium)
-

- integrado com ferramentas de integração contínua e entrega contínua (Jenkins, Travis CI e GitHub Actions)
- vasta gama de extensões de terceiros
- configuração inicial pode ser complexa
- funcionalidades avançadas para Python dependem de extensões
- muitas extensões pode aumentar o uso de recursos do sistema
- Suporte a Depuração Limitado
- configuração de ambientes virtuais pode ser um pouco mais manual

Criação de ambiente virtual do Python no VSCode (venv)

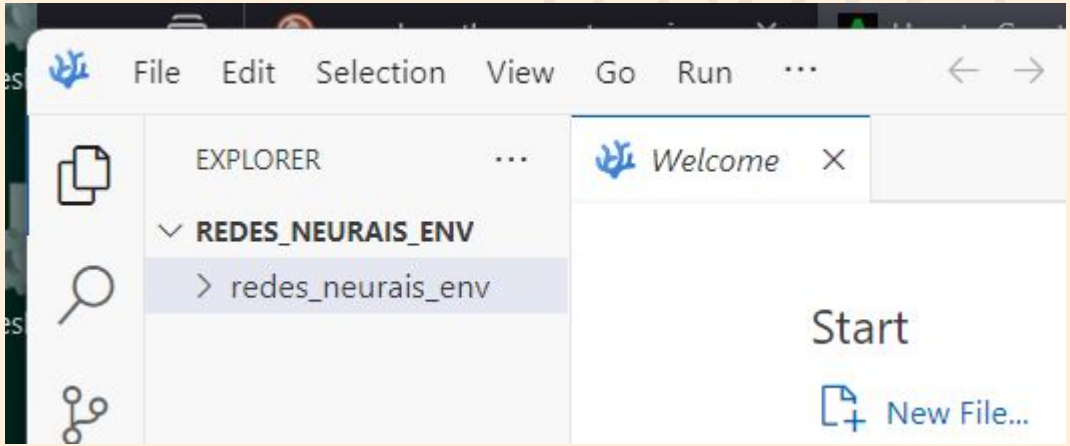
- Quando criamos qualquer projeto (ou projeto ativo), um ambiente virtual nos ajuda a colocar todas as dependências em um único lugar para evitar conflitos com nosso ambiente Python global ou com outros projetos.
- Ele encapsula todas as dependências dos projetos no ambiente virtual, fornecendo um espaço limpo e isolado para os requisitos do projeto.

- Primeiro, crie sua pasta (ou pasta do projeto) e abra-a no VScode.
- No nosso caso, o nome da pasta é "RNA" (de Redes Neurais Artificiais)
- Abra o terminal Vscode (menu View / Terminal) e escreva o comando abaixo para criar um ambiente virtual Python:
- > python -m venv RNA_env

Criação de ambiente virtual do Python no VSCode (venv)

- Esse comando cria uma pasta (ou diretório) que contém o interpretador Python juntamente com sua biblioteca padrão e pacotes ou módulos adicionais.
- Para ativar seu ambiente virtual, escreva o comando abaixo de acordo com seu sistema:
- Linux:
  - (base) source RNA_env/scripts/activate
- Windows:

  > RNA_env/scripts/activate



OPS!!

Corrigindo (no Windows) ...

- execução de scripts foi desabilitada neste sistema
- No terminal VSCode:

..\RNA_env> Get-ExecutionPolicy

- *Restricted*
- Corrigindo…

Set-ExecutionPolicy -Scope CurrentUser

Forneça valores para os seguintes parâmetros:

ExecutionPolicy: Bypass

> RNA_env/scripts/activate

- (RNA_env) caminho\RNA_env>

Ambos: Windows / Linux

- Agora que nosso ambiente virtual está ativado, você pode instalar qualquer pacote ou biblioteca Python dentro dele.
- Para esta demonstração, instalaremos duas bibliotecas Python externas: NumPy e Pandas.
- (RNA_env) caminho\RNA_env> pip install numpy
- (RNA_env) caminho\RNA_env> pip install pandas

Para desativar o ambiente

- Antes de desativar

> pip list

- Mostra os pacotes instalados no ambiente e suas dependências

- Para desativar seu ambiente virtual, você pode executar o seguinte comando:

> deactivate

- Mas não é obrigatório ao terminar uma sessão de uso
- Na próxima sessão de uso, basta repetir as instruções do slide 6

Fonte: https://allinpython.com/how-to-create-python-virtual-environment-in-vscode/