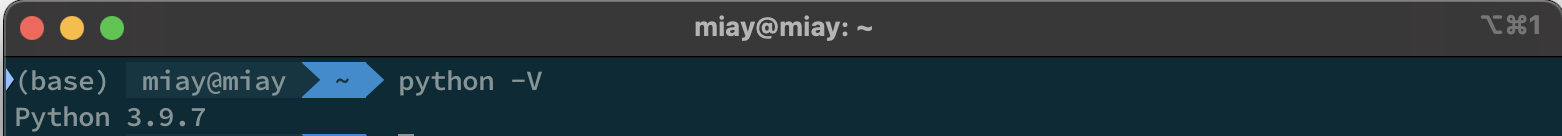
# Python installieren

Wir installieren Python nicht von python.org, da wir damit allein nicht alle notwendigen Tools haben, um später bestimmte Libraries wie numpy auf allen Betriebssystemen zu installieren. Stattdessen verwenden wir eine Python Distribution namens Anaconda, die die nötigen Tools mitliefert. Anaconda gibt es in zwei Varianten: mit vorinstallierten Libraries und ohne. Da wir Libraries immer abhängig vom Projekt installieren, wählen wir Miniconda, die Variante ohne weitere Libraries.

1. Downloade die für dich passende Version von Miniconda: <https://docs.conda.io/en/latest/miniconda.html>
2. Folge der Installationsanleitung:   
   <https://conda.io/projects/conda/en/latest/user-guide/install/index.html>  
   Die default-Einstellungen unter Windows (siehe Screenshot) können beibehalten werden. Unter Windows wird Miniconda normalerweise nicht global installiert, sondern steht nur über eine besondere Kommandozeile namens Anaconda Prompt zur Verfügung. Programme wie Visual Studio Code können die Miniconda-Installation aber automatisch erkennen und stellen alle virtuellen Umgebungen, die mit Miniconda erzeugt werden, als Python Interpreter zur Verfügung.  
   Graphical user interface, text, application, email

   Description automatically generated
3. Öffne entweder die Anaconda Prompt (Windows) oder dein Standard-Terminal. Wenn die Installation erfolgreich war, solltest du jetzt am Anfang der Zeile vor dem Prompt den Text (base)sehen. Die aktuell aktivierte Python-Version kannst du mit python -V ausgeben lassen.   
   

# Virtuelle Umgebungen

Anaconda stellt ein Packet- und Umgebungsmanager namens conda bereit. Damit können wir neue virtuelle Umgebungen für Python anlegen. Jede Umgebung hat ihre eigene Python-Version und eigene Libraries.

Um eine neue Umgebung für Data Science anzulegen:

1. Öffne die Anaconda Prompt (Windows) oder ein Terminal, indem conda zur Verfügung steht ((base)-Text am Anfang).
2. Nutze den conda create Befehl zum Anlegen einer neuen Umgebung. Mit -n kannst du den Namen der Umgebung festlegen und mit python=3.x kannst du die Python-Version festlegen, die in der neuen Umgebung installiert werden soll.

|  |
| --- |
| $ conda create -n py311 python=3.11 |

Ein Bild, das Text, Screenshot, Schrift enthält.

Automatisch generierte Beschreibung

1. Bestätige die Installationmity.conda create wird standardmäßig eine Reihe von Paketen installieren, mit der jede neue Umgebung ausgestattet wird.   
   Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

   Automatisch generierte Beschreibung
2. Aktiviere die neue Umgebung mit conda activate**.** Nach erfolgreicher Aktivierung ändert sich der Name der Umgebung im Terminal (im Beispiel zu (py310)).

|  |
| --- |
| $ conda activate <env-name> |

Ein Bild, das Text, Screenshot, Schrift, Software enthält.

Automatisch generierte Beschreibung

1. Du kannst alle Pakete einer Umgebung aktualisieren, indem du conda update --all verwendest. Bestätige die Installation wie gewohnt mit y.

# Pakete installieren

Pakete bzw. Libraries können entweder mit Hilfe von conda oder pip installiert werden. Grundsätzlich empfehlen wir pip, da damit auch Pakete installiert werden können, die es nicht in Anaconda gibt. Alle Pakete werden immer in die aktuell ausgewählte Umgebung installiert.

Vorbereitung:

1. Öffne die Anaconda Prompt (Windows) oder ein Terminal, indem conda zur Verfügung steht ((base)-Text am Anfang).
2. Optional: Aktiviere die gewünschte virtuelle Umgebung, in der du das Paket installieren möchtest:

|  |
| --- |
| $ conda activate <env-name> |

**conda**

1. Führe conda install auf der Konsole aus, um ein Paket zu installieren. conda erkennt alle dependencies (Abhängigkeiten) des zu installierenden Paketes und installiert diese ebenfalls. Die Installation muss mit y bestätigt werden. Du kannst mehrere Pakete mit Leerzeichen getrennt angeben.

|  |
| --- |
| $ conda install <package1> <package2> … |

**Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung**

1. Alternativ können Pakete auch direkt bei der Erzeugung einer neuen Umgebung mit angegeben werden:

|  |
| --- |
| $ conda create -n <name> python=3.x <package1> <package2> … |

1. Führe conda update auf der Konsole aus, um ein Paket zu aktualisieren.

|  |
| --- |
| $ conda update <package1> <package2> … |

**pip**

1. Führe pip install auf der Konsole aus, um ein Paket zu installieren. pip erkennt alle dependencies (Abhängigkeiten) des zu installierenden Paketes und installiert diese ebenfalls. Du kannst mehrere Pakete mit Leerzeichen getrennt angeben.

|  |
| --- |
| $ pip install <package1> <package2> … |

1. Mit dem Zusatz --upgrade kannst du mit pip install auch Pakete aktualisieren:

|  |
| --- |
| $ pip install --upgrade <package1> <package2> … |

# Jupyter Notebooks starten

Um Notebooks im Format .ipynb öffnen zu können, muss die Erweiterung jupyter installiert werden. Damit steht der Befehl jupyter notebook zur Verfügung. Eine Erweiterung ist jupyterlab, welches das reine Jupyter Notebook noch um weitere Features ergänzt, darunter einen Dateibrowser, mehrere Notebooks im Tab sowie ein integriertes Inhaltsverzeichnis. Mit dem Befehl jupyter lab kann Jupyterlab gestartet werden. Wichtig ist bei allen beiden Befehlen, dass der Server immer in dem Verzeichnis startet, in dem man sich zum Zeitpunkt des Befehlaufrufs in der Konsole befand.

1. Öffne die Anaconda Prompt (Windows) oder ein Terminal, indem conda zur Verfügung steht ((base)-Text am Anfang).
2. Optional: Aktiviere die gewünschte virtuelle Umgebung, in der du Jupyter starten möchtest:

|  |
| --- |
| $ conda activate <env-name> |

1. Installiere jupyter und, wenn gewünscht, jupyterlab:

|  |
| --- |
| $ pip install jupyter jupyterlab |

1. Optional: Sofern du nicht in deiner (base)-Umgebung bist, musst du die virtuelle Umgebung noch als Kernel registrieren:

|  |
| --- |
| (venv) $ python -m ipykernel install -–user --name <name> --display-name <display-name> |

Als Name <name> kannst du eine Kurzform verwenden, unter der der Kernel intern angesprochen werden kann. Der Anzeigename <display-name> ist der Name, unter dem der Kernel dann im Notebook ausgewählt werden kann.

1. Wechsle mit dem cd-Befehl in das Verzeichnis, in dem du ein Notebook starten bzw. öffnen möchtest.
2. Starte jupyter notebook oder jupyter lab:

|  |
| --- |
| $ jupyter notebook |

|  |
| --- |
| $ jupyter lab |

Daraufhin öffnet sich dein Standardbrowser und ruft die Seite localhost:8888 auf.

1. Wähle deinen Kernel aus. Wenn du ein Notebook zum ersten Mal startest oder öffnest, musst du den Kernel auswählen, mit dem das Notebook ausgeführt werden soll. Dazu kannst du aus der Liste deinen gewünschten Kernel auswählen.

Graphical user interface, text, email

Description automatically generated

Graphical user interface, text, application, email, website

Description automatically generated