

Data Mining

Practical Session #1

Fall Semester 2024-2025
Master in Data Science and Advanced Analytics

Pedro Ferreira – pnferreira@novaims.unl.pt
Farina Pontejos – fpontejos@novaims.unl.pt

Please download the lab materials in the
meantime

Practical Session Preparation

Download Anaconda Navigator

- <https://www.anaconda.com/download>
- Documentation: <https://docs.anaconda.com/free/anaconda/install/>

Download notebook and other materials

- Moodle / Practical Sessions / Lab 01
- <https://github.com/fpontejos/Data-Mining-24-25>

Optional: Download GitHub Desktop

- <https://desktop.github.com/>
- Sign up for GitHub account: <https://github.com/signup>

About Us

Pedro Ferreira

Academic Background:

- BSc in Physics (2022)
- MSc in Data Science & Advanced Analytics (2022 – Oct 2024 – NOVA IMS)

Professional Experience:

- Data Science & Analysis in the Banking Sector

Research Experience:

- Machine Learning for Astrophysics
- Supervised Machine Learning for Football Performance Evaluation

About Us

Farina Pontejos

Academic Background:

- BSc in Mining Engineering (2011)
- MSc in Data Science & Advanced Analytics (2021-2023 – NOVA IMS)
- PhD in Information Management (since 2023 [Ongoing] – NOVA IMS)

Professional Experience:

- Mining Engineer; Web Developer
- IT Consulting (Systems Analysis; ERP; Data Science)

Research Experience:

- Natural Language Processing; Large Language Models; Software Engineering Applications

Resources

- Bibliography
- Data Mining Github repo:
 - <https://github.com/fpontejos/Data-Mining-24-25>
- Class slides and Jupyter Notebooks
- Google, Stack Overflow, documentations, Github and YouTube

Resources

Recommended DataCamp courses

Use the ***invite link*** (on Moodle) to get 6 months free access to all DataCamp features.
Make sure to ***use your university email address*** to sign up.

Introduction to Python

<https://app.datacamp.com/learn/courses/intro-to-python-for-data-science>

Intermediate Python

<https://app.datacamp.com/learn/courses/intermediate-python>

Our working environment

- We will be using Anaconda: Currently one of the most popular Python distributions.
- Sets up a data science oriented working environment in Python
- It installs a set of libraries (for now, think of libraries as programming tools – like a toolbox in a woodshop)
- But it can be used for many different purposes (all it takes is installing the necessary libraries)

Anaconda

www.anaconda.org

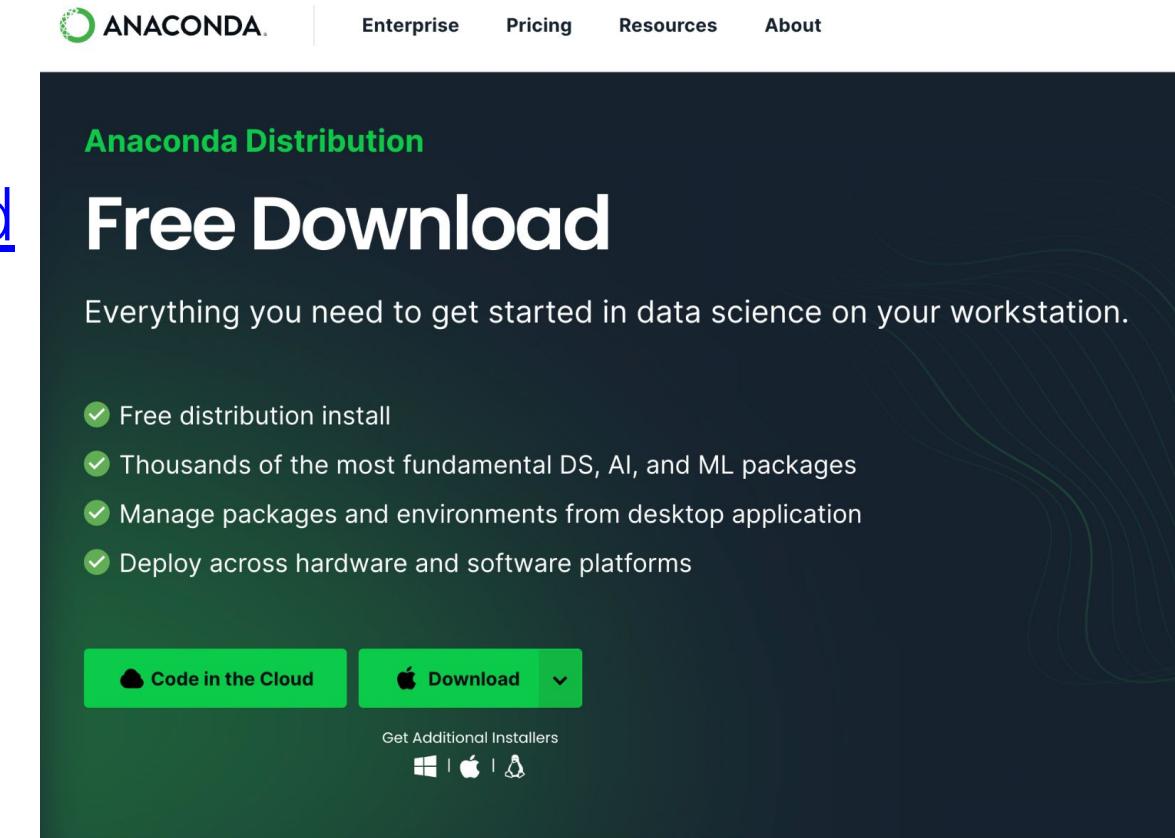
- one of the most popular Python distributions for Data Science
- manages your packages and environments.
- reduce future issues dealing with the various libraries you will be using.
- comes with most of the main libraries for data manipulation
 - Pandas
 - Numpy
 - Matplotlib
 - Scipy
 - ...
- easy to use and install



In the meantime: Download + Install Anaconda Navigator

Download:

<https://www.anaconda.com/download>



Install Anaconda Navigator

Please read the documentation applicable to your system:

<https://docs.anaconda.com/free/anaconda/install/>



When [installing Anaconda](#), you have the option to “Add Anaconda to my PATH environment variable.” *This is not recommended because it appends Anaconda to PATH.* When the installer appends to PATH, it does not call the activation scripts.

Virtual Environments

<https://docs.conda.io/projects/conda/en/latest/user-guide/concepts/environments.html>

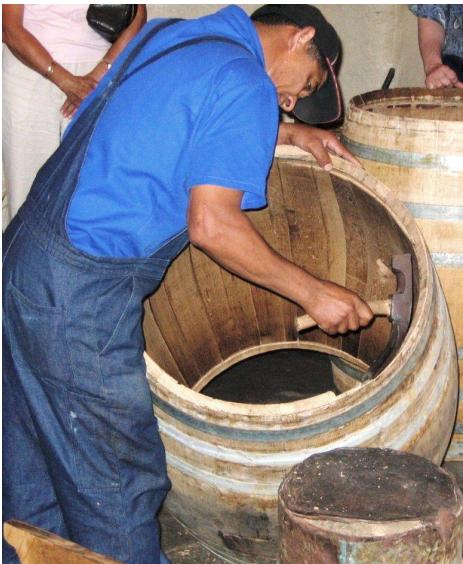
<https://docs.conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html>

- Isolated spaces that contain per-project dependencies (specific collection of installed conda packages)
- Using conda to manage environments:
 - Create, export, list, remove, and update environments
 - Switching or moving between environments (conda activate)
 - You can also share an environment file
- You can also use pip to manage environment

In other words

you may need to use different “versions” of the same types of tools, or entirely different “environments”

Project #1



Cooper (making a barrel)

Project #2



Formwork (setting a structure)

Project #3



Luthier (making a guitar)

In other words

Suppose you will build a Caravel (a Portuguese ship from the 15th century).

However, you are also a Cooper!

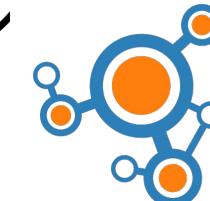
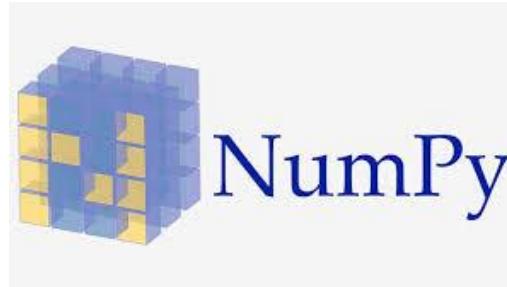
To build the Caravel the way they did back then, you will need a specific set of tools, much more rudimentary than the ones you will have at your own workshop.



In other words

- You will need to get them first (i.e., “download” them)
- However, you should not mix these tools with the ones you already have!
 - They are not appropriate to build barrels, and the ones you already have are not appropriate to build Caravels
 - They will create clutter in your workshop (unnecessarily keeping unused packages)
 - You will have duplicate tools with different (version conflicts)
 - Other carpenters may want to build their own replica of your project (reproducibility)
 - If you are working with other carpenters, they will need to use the same types of tools you are using (collaboration)
- These tools (i.e., libraries) and their versions should be specified in the project’s requirements (including the Python version)!

Python Packages



Git and GitHub

<https://guides.github.com/activities/hello-world/>

<https://docs.github.com/en/github/getting-started-with-github>

- What is GitHub?
 - . Code hosting platform for version control and collaboration
- What is Git?
 - . At the heart of GitHub is an open source version control system (VCS) called Git. Git is responsible for everything GitHub-related that happens locally on your computer.
- Why Git and GitHub?
 - . **Optional:** You can use Git and GitHub for collaborating and version control in your projects.
 - . Also we have a GitHub repository with all the practical class contents:
 - . <https://github.com/fpontejos/Data-Mining-24-25>

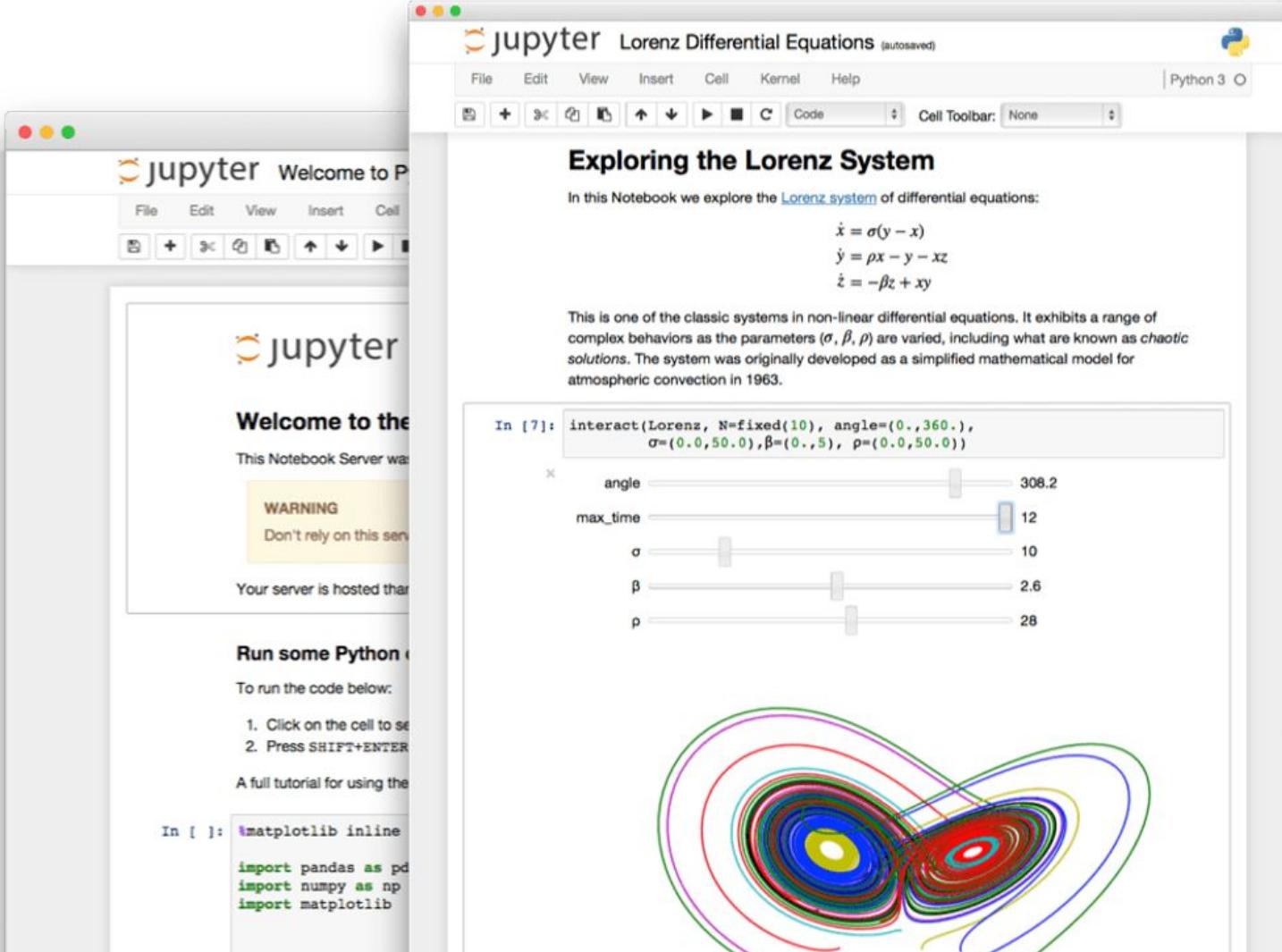
Main ways to access Python

- Python Shell and IPython
 - An interactive environment for writing and running code
- Jupyter Notebooks
 - A notebook that weaves code, data, prose, equations, analysis, and visualization
 - A tool for prototyping new code and analysis
 - A method for creating a reproducible workflow for scientific research
- IDE (Integrated Development Environment):
 - Software that helps you build code

Jupyter notebooks

<https://jupyter.org/>

We will be using Jupyter notebooks for our practical sessions.



The screenshot shows a Jupyter Notebook window titled "jupyter Lorenz Differential Equations (autosaved)". The toolbar includes File, Edit, View, Insert, Cell, Kernel, Help, and a Python 3 O button. Below the toolbar, there are buttons for cell operations like Run, Cell, Cell Toolbar, and Cell Type. The main content area is titled "Exploring the Lorenz System" and describes the Lorenz system of differential equations:

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ , β , ρ) are varied, including what are known as chaotic solutions. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

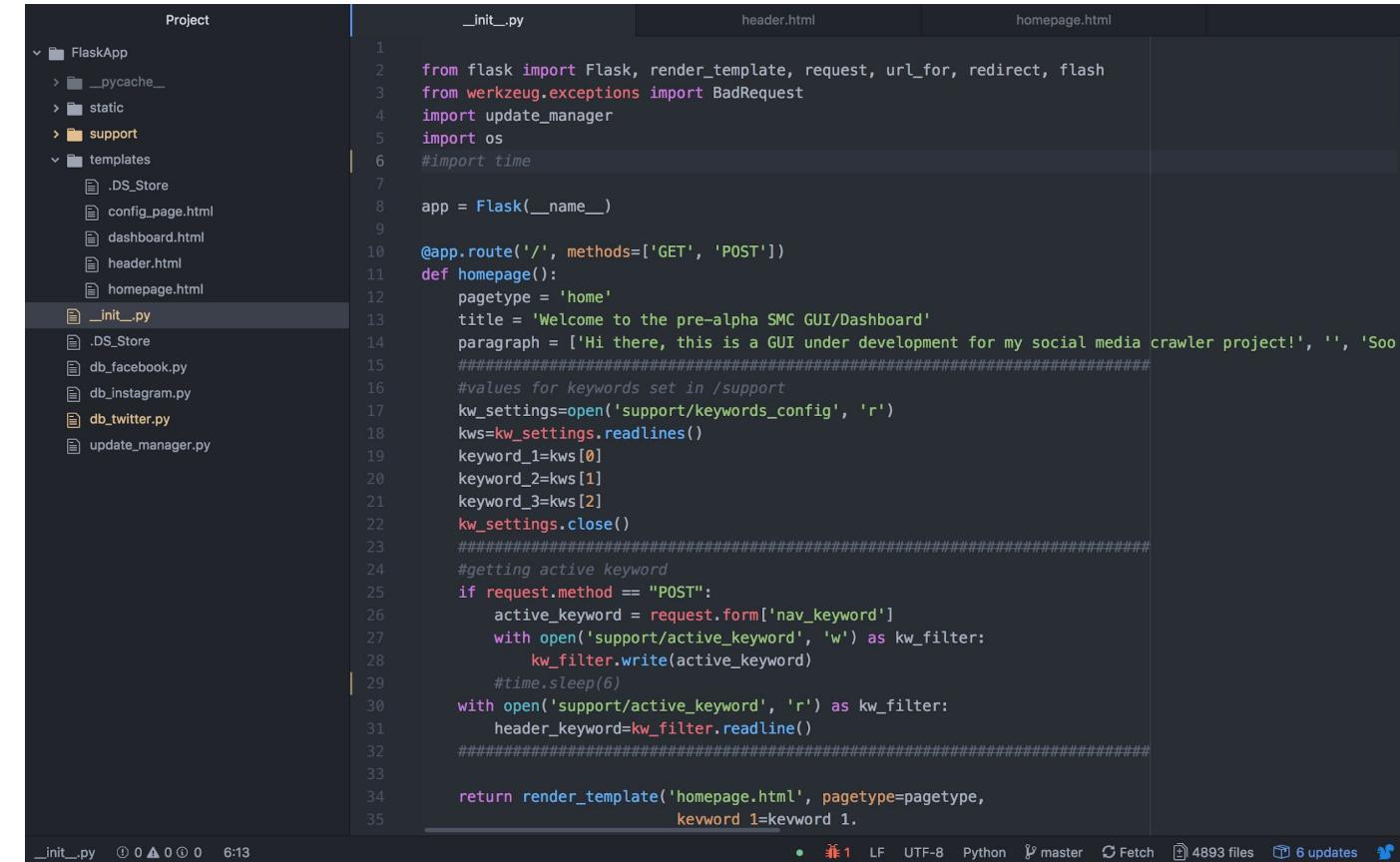
In [7]:

```
interact(Lorenz, N=fixed(10), angle=(0.,360.),
          sigma=(0.0,50.0),beta=(0.,5.), rho=(0.0,50.0))
```

Below the code, there are five sliders for parameters: angle (308.2), max_time (12), sigma (10), beta (2.6), and rho (28). At the bottom, a 3D plot visualizes the Lorenz attractor, showing a complex, butterfly-shaped trajectory in red, blue, and green.

Text Editors

- Another method to write python scripts is using text editors
- Some popular text editors:
 - Vim (Linux terminal text editor)
 - Atom (popular open source editor)
 - Sublime Text (popular proprietary text editor)
 - Notepad ++ (Windows only)
- Usually highly customizable



The screenshot shows the Atom Text Editor interface. On the left, there's a sidebar titled "Project" displaying a file tree for a "FlaskApp" project. The tree includes directories like __pycache__, static, support, and templates, along with files such as .DS_Store, config_page.html, dashboard.html, header.html, homepage.html, and __init__.py. The main editor area has three tabs: __init__.py, header.html, and homepage.html. The __init__.py tab is active and shows Python code for a Flask application. The code imports Flask, werkzeug.exceptions, update_manager, os, and time. It defines a Flask app object and a homepage() function that sets pagetype to 'home', title to 'Welcome to the pre-alpha SMC GUI/Dashboard', and paragraph to a welcome message. It then reads keyword settings from a file, processes them, and writes the active keyword to another file. Finally, it renders the homepage.html template. The status bar at the bottom shows the file name, line count (0), character count (0), word count (0), and the current time (6:13). The bottom right corner of the status bar also shows repository statistics: 4893 files, 6 updates, and a GitHub icon.

```

Project
FlaskApp
  __pycache__
  static
  support
  templates
    .DS_Store
    config_page.html
    dashboard.html
    header.html
    homepage.html
  __init__.py
  .DS_Store
  db_facebook.py
  db_instagram.py
  db_twitter.py
  update_manager.py

__init__.py
from flask import Flask, render_template, request, url_for, redirect, flash
from werkzeug.exceptions import BadRequest
import update_manager
import os
# import time

app = Flask(__name__)

@app.route('/', methods=['GET', 'POST'])
def homepage():
    pagetype = 'home'
    title = 'Welcome to the pre-alpha SMC GUI/Dashboard'
    paragraph = ['Hi there, this is a GUI under development for my social media crawler project!', '',
    ##### values for keywords set in /support
    kw_settings=open('support/keywords_config', 'r')
    kws=kw_settings.readlines()
    keyword_1=kws[0]
    keyword_2=kws[1]
    keyword_3=kws[2]
    kw_settings.close()
    #####
    #getting active keyword
    if request.method == "POST":
        active_keyword = request.form['nav_keyword']
        with open('support/active_keyword', 'w') as kw_filter:
            kw_filter.write(active_keyword)
        #time.sleep(6)
        with open('support/active_keyword', 'r') as kw_filter:
            header_keyword=kw_filter.readline()
    #####
    return render_template('homepage.html', pagetype=pagetype,
                           keyword_1=keyword_1.

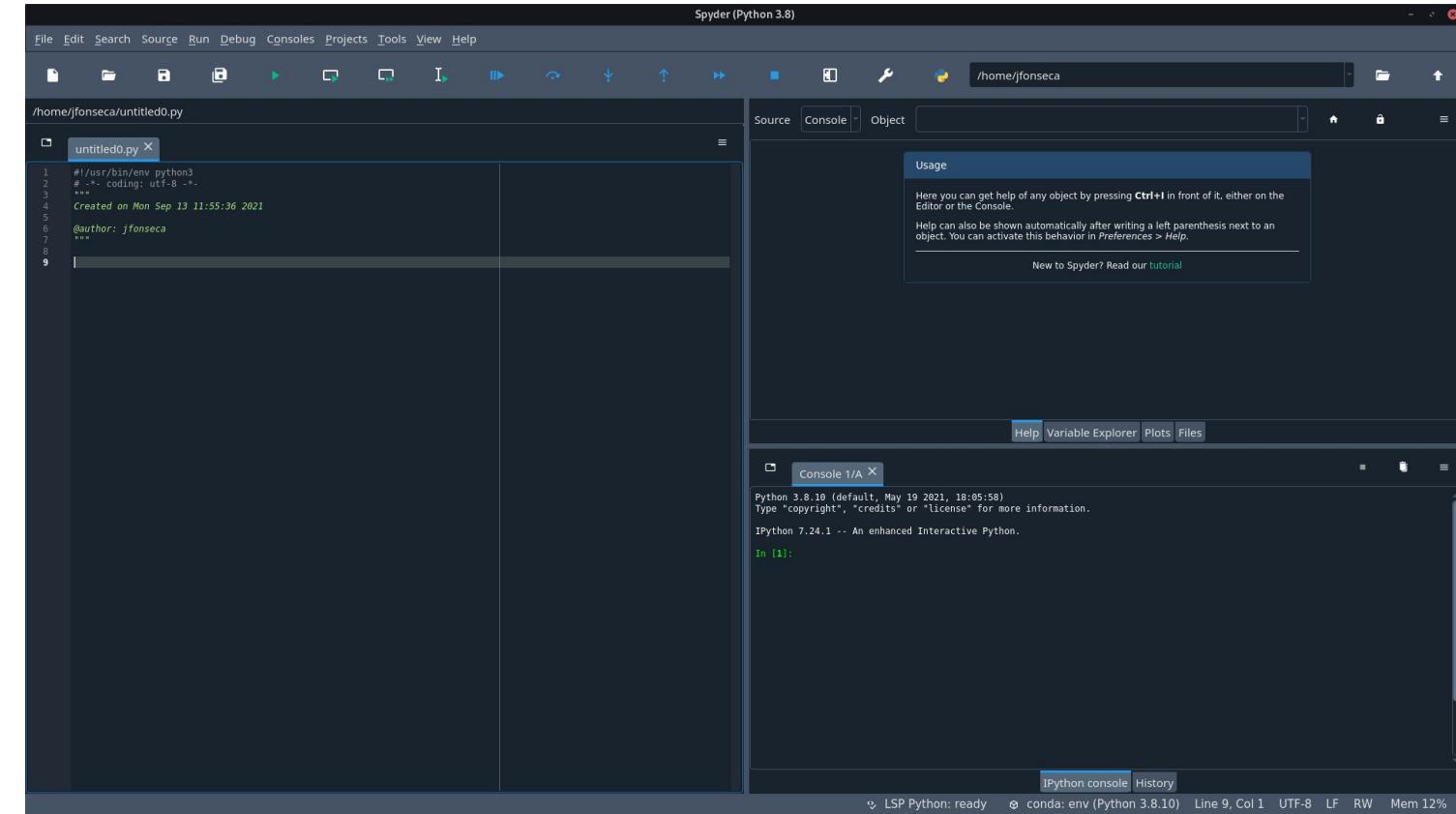
```

Atom Text Editor

Integrated Development Environment (IDE)

- Popular IDE's:
 - Spyder
 - PyCharm
 - VSCode
 - Rodeo
- Anaconda comes with Spyder and VSCode

Usage of IDE and/or Text editor (and which ones to use) comes down to personal preference



Data Mining Project

Data Mining Project

- Form groups on Moodle (**up to 4 students**)
 - Project guidelines will be released on Moodle
 - Anonymized data from a real-world scenario
-
- Your goal is to develop a **Customer Segmentation** in such a way that it will be possible for the Marketing Department to better understand all the different Customers' Profiles.

Everything is on Moodle

202425 - Data Mining - S1

Course Settings Participants Grades Reports More ▾

General

 Anúncios

 Project Materials

 Hidden from students

 Project Groups

 Hidden from students

You should be able to see this now :)

Please read the guidelines

2024-2025 / 2º ciclo / Pós-Graduações | Postgrau
/ Project Materials

Project Materials

Folder Settings More ▾

Edit



DM2425_ABCDEats_DATASET.csv



DM2425_ProjectGuidelines.pdf

ABCDEats Inc.

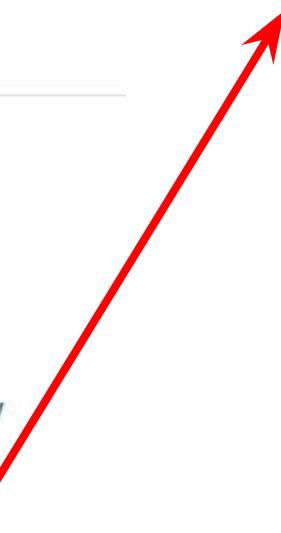
Data Mining Project Guidelines

version: v6.2024.09.09

Fall Semester 2024-2025

Master in Data Science and Advanced Analytics

NOVA Information Management School



Project Part 1: EDA

Project Evaluation

The project will be evaluated based on three components. All components are mandatory.

Part 1: Exploratory Data Analysis (20%)

Expected Outcomes

- Conduct an in-depth exploration of the dataset. Summarise key statistics for the data, and discuss their possible implications.
- Identify any trends, patterns, or anomalies within the dataset. Explore relationships between features.
- Create new features that may help enhance your analysis.
- Use visualisations to effectively communicate your findings.

Submission deadline

- 04 November 2024 21h00
- 10% penalty for each day of delay

Deliverables

- Written report, maximum of 5 pages.
- Python code, in a Jupyter notebook with cells already executed.
- See Project Deliverables section for details

Project Part 1: EDA

Academic paper example:

M. Maphosa, W. Doorsamy and B. S. Paul, "Student Performance Patterns in Engineering at the University of Johannesburg: An Exploratory Data Analysis," in *IEEE Access*, vol. 11, pp. 48977-48987, 2023, doi: 10.1109/ACCESS.2023.3277225.

<https://ieeexplore.ieee.org/abstract/document/10128127>

Project Part 2: Final Report

Part 2: Final Report (70%)

Expected Outcomes

- Preprocess the data. Invest time into evaluating your preprocessing pipeline, explaining the choices you made and the advantages and disadvantages of different decisions.
- Justify the clustering approach. Determine and provide a rationale for the clustering solution, including the number of clusters, that you decide to use.
- Explain the clusters in your final segmentation. Analyse and describe the characteristics of each group, taking into account the perspectives you used. Create profiles that highlight the distinguishing features of each cluster.
- Suggest business applications. Based on your insights, define general marketing approaches for each cluster.

Submission deadline

- 03 January 2025 21h00
- 10% penalty for each day of delay

Deliverables

- Written report, maximum of 10 pages.
- Python code, in a Jupyter notebook with cells already executed.
- See Project Deliverables section for details

Project Part 3: Discussion

Part 3: Project Discussion (10%)

After submitting the final reports, each group will be scheduled for a 15-20 minute discussion with one of the instructors. During this discussion, students will answer questions about their analysis and explain their methodologies.

Each student will receive an individual grade based on their contribution to the discussion. This aims to assess each student's level of engagement with, and comprehension of the delivered work.

Target schedule: 21-25 January 2025. **This is not yet final** and will be confirmed later on Moodle.

Project Bonus (Optional)

Optional: Opportunity for Bonus Points (max 20%)

Note: *This does not replace the requirement to deliver the other parts of the project.*

As an optional part of the project, you may develop an interactive application that allows ABCDE to explore and interact with the EDA and customer segmentation analysis performed previously. The application should be user-friendly, visually intuitive, and allow the user to gain insights from the customer segmentation results.

This is intentionally open-ended; some suggestions are provided below, but the kinds of visualisations or features to include is limited only by your imagination and skill.

The only restriction is that it must be developed using Python.

Possible application features:

- **Cluster Exploration:** The application must provide an interactive interface where users can explore the different customer segments. Each cluster should be clearly defined, with key demographic, behavioral, and preference-based characteristics displayed.
- **Visualisation Tools:** Implement visual tools such as charts, graphs, or scatter plots that help users visualize the clustering results. Consider using heatmaps, pie charts, or bar graphs to represent segment data.
- **Filter Functionality:** Allow users to filter clusters based on various attributes (e.g., age, location, purchase history) and explore how these filters affect the composition of the clusters.
- **Cluster Comparison:** Enable users to compare different clusters side-by-side in terms of their key characteristics, providing detailed insights into their similarities and differences.
- **Personalized Insights:** Include an option where users can input specific customer attributes (e.g., age, region, payment method) and get insights into which cluster the customer is most likely to belong to.

Submission deadline

- 07 January 2025 21h00
- **Please inform me by email if you intend to deliver this optional part.**

Let's get started!

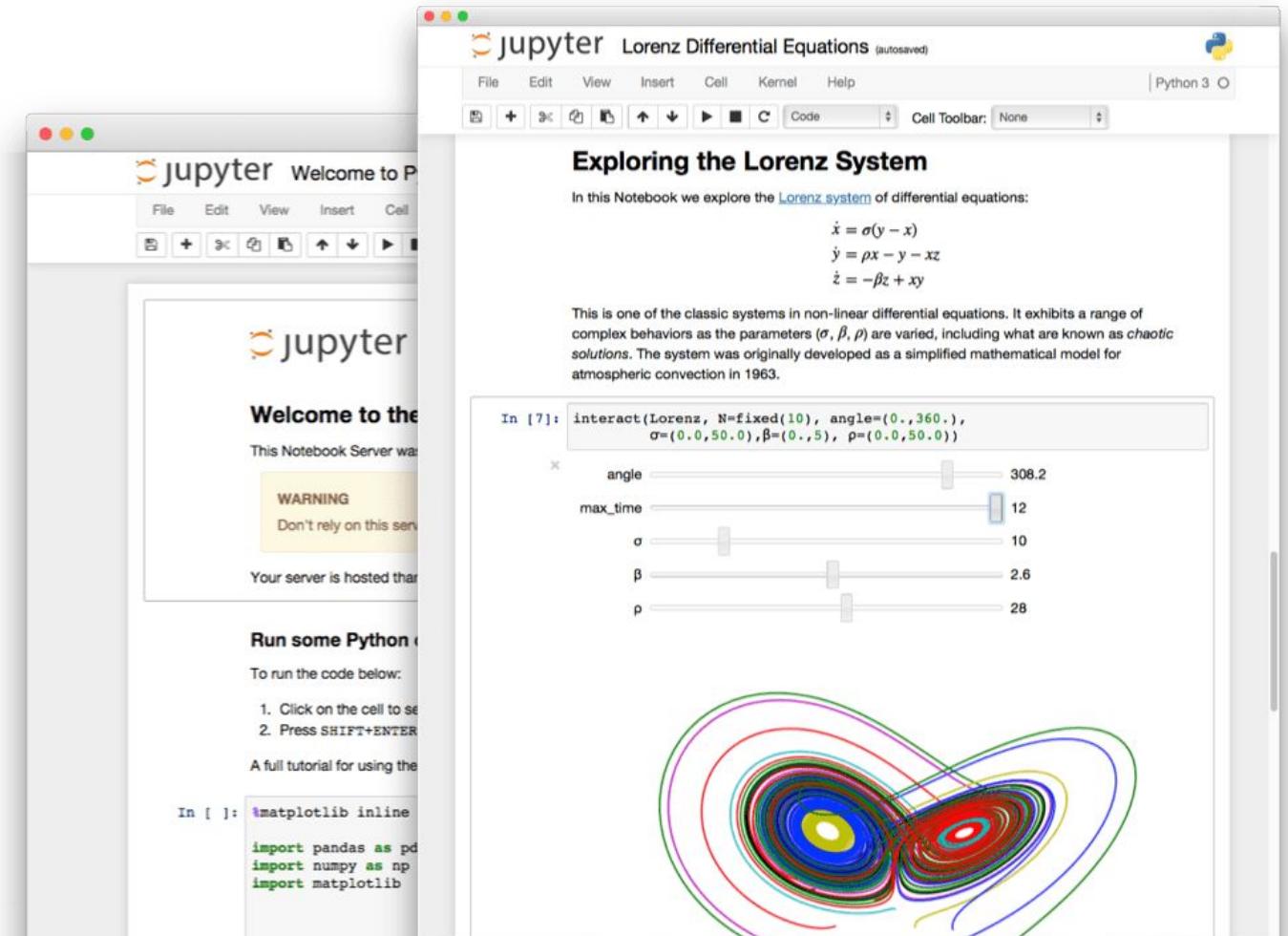
Next: Setting up our tools

The Jupyter Notebook

<http://jupyter.org/>

Let's try it out!

- Install and Open Anaconda Navigator
- Start Jupyter Notebook



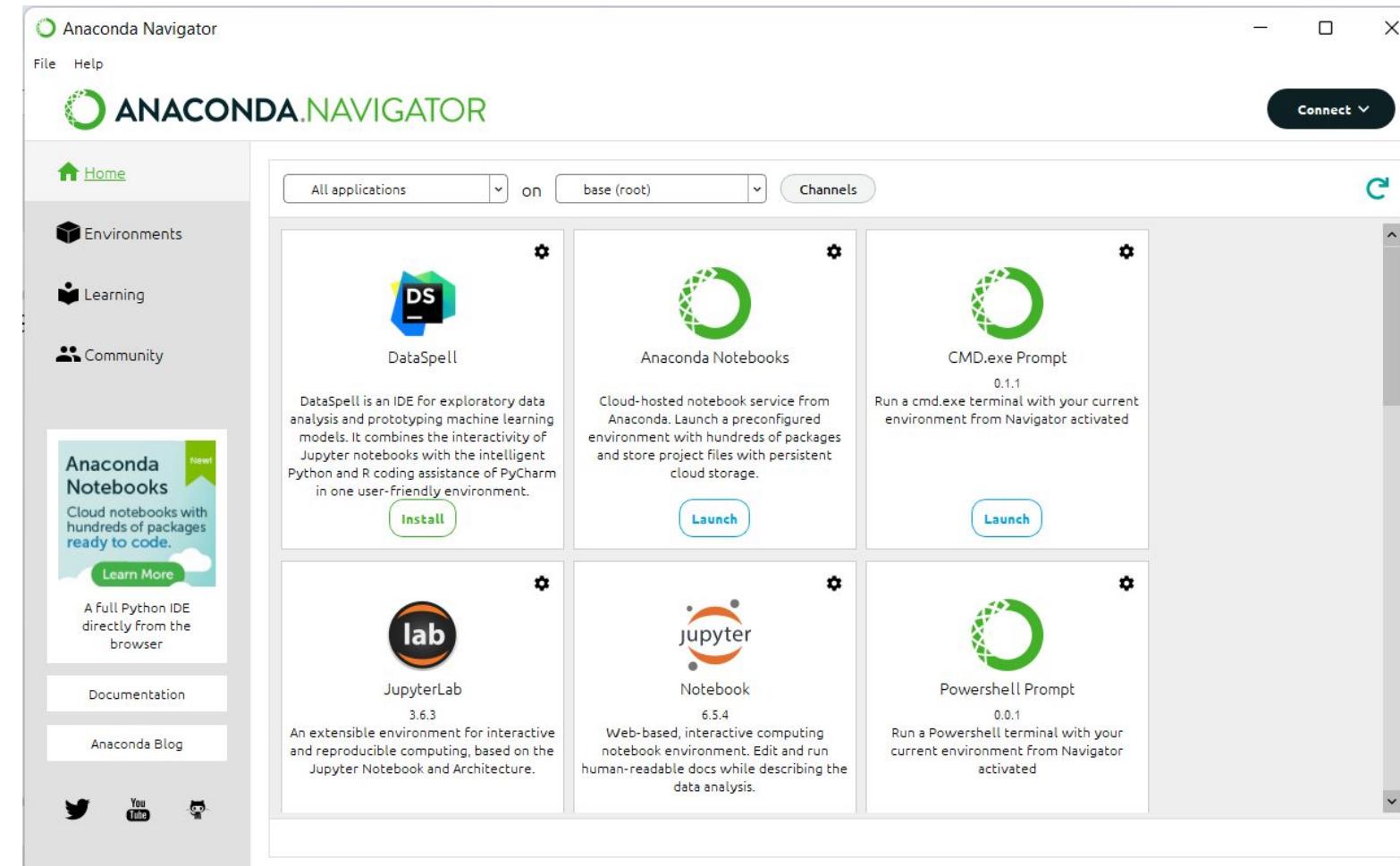
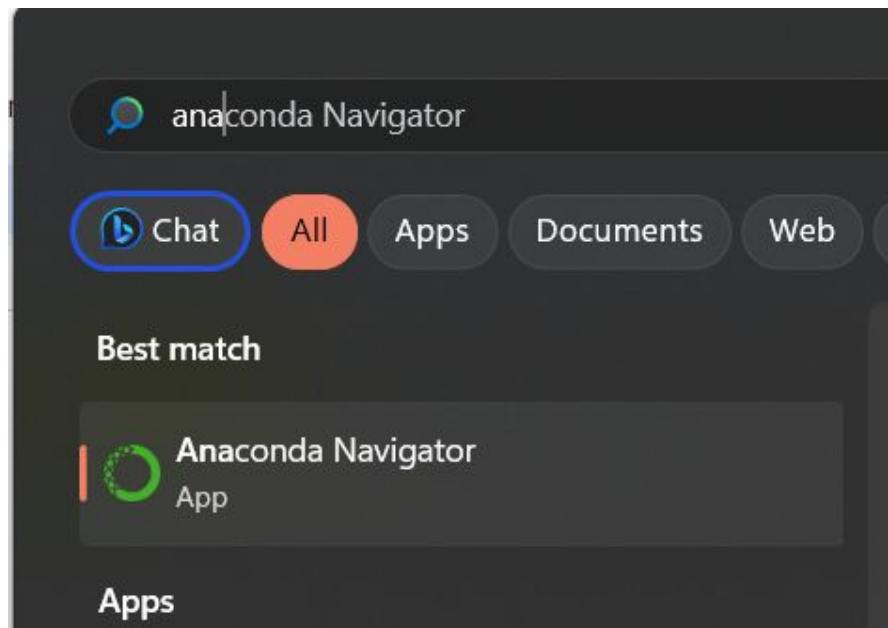
Setting up our tools

Two options:

1. **With GUI (Anaconda Navigator)**
2. Command line (miniconda) (skip to slide 64)

Setting up our tools

Load Anaconda Navigator



Create a new environment: DM2425

The screenshot shows the Anaconda Navigator interface. On the left, there's a sidebar with links like Home, Environments (which is highlighted with a red box), Learning, Community, and Anaconda Toolbox (with a 'Read the Docs' button). Below the sidebar are buttons for Documentation, Anaconda Blog, and social media links (Twitter, YouTube, GitHub). In the center, there's a search bar and a tree view showing 'base (root)' with a play button. At the bottom, there are buttons for Create, Clone, Import, Backup, and Remove. A large red box highlights the 'Create' dialog window in the center-right. The dialog has fields for Name (set to 'DM2425'), Location (empty), Packages (Python 3.12.4 selected, R 4.3.1 available), and buttons for Cancel and Create. The entire dialog is also highlighted with a red border. Red numbers #1, #2, and #3 are overlaid on the interface: #1 is on the 'base (root)' item, #2 is on the 'Create' button in the sidebar, and #3 is on the Python version dropdown in the dialog.

#1

#2

#3

ANACONDA.NAVIGATOR

Search Environments

base (root)

Home

Environments

Learning

Community

Anaconda Toolbox

Supercharged local notebooks. Click the Toolbox tile to install.

Read the Docs

Documentation

Anaconda Blog

Create

Clone

Import

Backup

Remove

Create new environment

Name: **DM2425**

Location:

Packages: Python **3.12.4** R **4.3.1**

Cancel

Create

<input checked="" type="checkbox"/> aext-share-notebook	The aext-share-notebook component of anaconda-toolbox	4.0.15
<input checked="" type="checkbox"/> aext-share-notebook-server	Anaconda extensions share notebook server	4.0.15

522 packages available

ica e Gestão de Informação
niversidade Nova de Lisboa

Create a new environment: DM2425

The screenshot shows the Anaconda Navigator application interface. On the left is a sidebar with links: Home, Environments (which is selected and highlighted in green), Learning, Community, Anaconda Toolbox (with a 'Read the Docs' button), Documentation, and Anaconda Blog. At the bottom are social media links for Twitter, YouTube, and GitHub.

The main area displays the 'Environments' view. A search bar at the top left says 'Search Environments'. Below it is a dropdown menu set to 'Installed', followed by 'Channels' and 'Update index...'. A 'Search Packages' bar with a magnifying glass icon is on the right.

The environment tree on the left shows 'base (root)' and 'DM2425', which is highlighted with a red box. To the right of the tree is a play button icon.

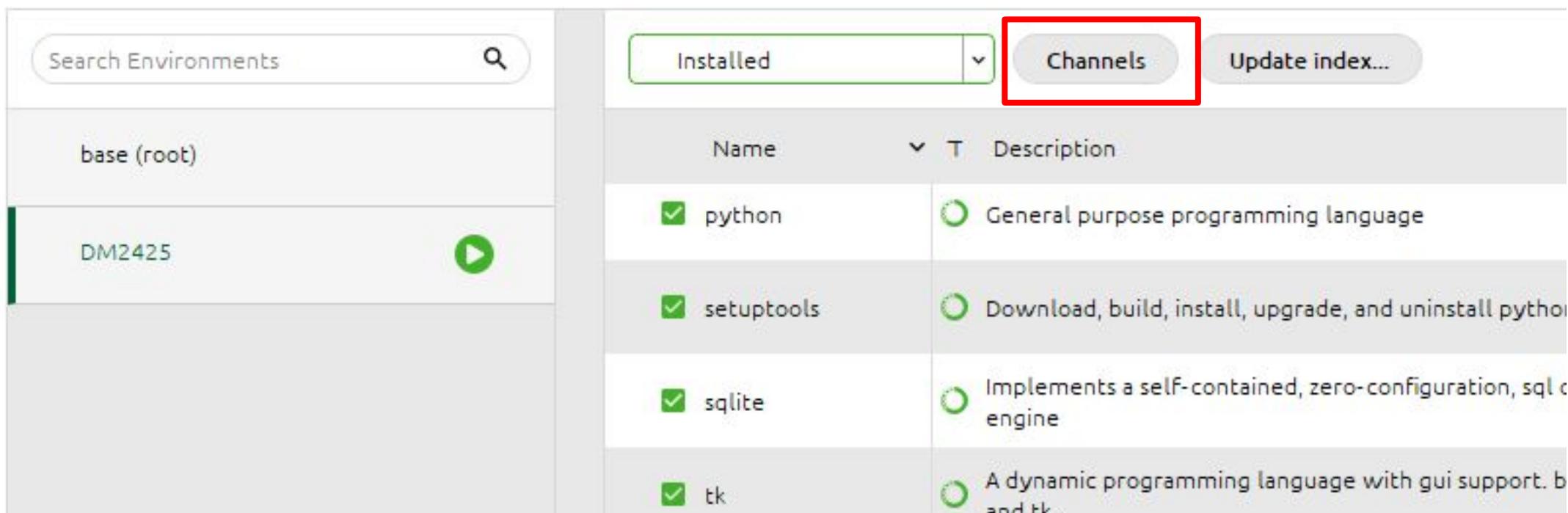
A large table on the right lists installed packages. The columns are 'Name', 'Description', and 'Version'. The packages listed are:

Name	Description	Version
python	General purpose programming language	3.12.4
setuptools	Download, build, install, upgrade, and uninstall python packages	72.1.0
sqlite	Implements a self-contained, zero-configuration, sql database engine	3.45.3
tk	A dynamic programming language with gui support. bundles tcl and tk.	8.6.14
tzdata	The time zone database (called tz, tzdb or zoneinfo)	2024a
vc	A meta-package to impose mutual exclusivity among software built with different vs versions	14.40
vs2015_runtime	Msvc runtimes associated with cl.exe version 19.29.30154 (vs 2019 update 11)	14.40.33
wheel	A built-package format for python.	0.43.0
xz	Data compression software with high compression ratio	5.4.6
zlib	Massively spiffy yet delicately unobtrusive compression library	1.2.13

At the bottom of the table, it says '16 packages available'.

At the very bottom of the interface are five buttons: Create, Clone, Import, Backup, and Remove.

Add conda channel: conda-forge



The screenshot shows the Conda interface for managing environments. On the left, there's a sidebar with a search bar for environments and two environment entries: "base (root)" and "DM2425". The "DM2425" entry has a green play button icon next to it. The main area is a table showing installed packages. The top navigation bar includes buttons for "Search Environments" (with a magnifying glass icon), "Installed" (which is currently selected and highlighted with a green border), "Channels" (which is highlighted with a red box), and "Update index...". The table has columns for Name, T (Type), and Description. The packages listed are:

Name	Type	Description
python	General purpose programming language	General purpose programming language
setuptools	Download, build, install, upgrade, and uninstall python packages	Download, build, install, upgrade, and uninstall python packages
sqlite	Implements a self-contained, zero-configuration, sql database engine	Implements a self-contained, zero-configuration, sql database engine
tk	A dynamic programming language with gui support. binds tk	A dynamic programming language with gui support. binds tk

Add conda channel: conda-forge

The screenshot shows a software interface for managing conda channels. At the top, there is a search bar with the text "pandas" and a clear button (X). Below the search bar, there are buttons for "All", "Channels", and "Update index...". A red box highlights the "Add..." button, which is located next to the search bar.

On the left, there is a list of channels with checkboxes:

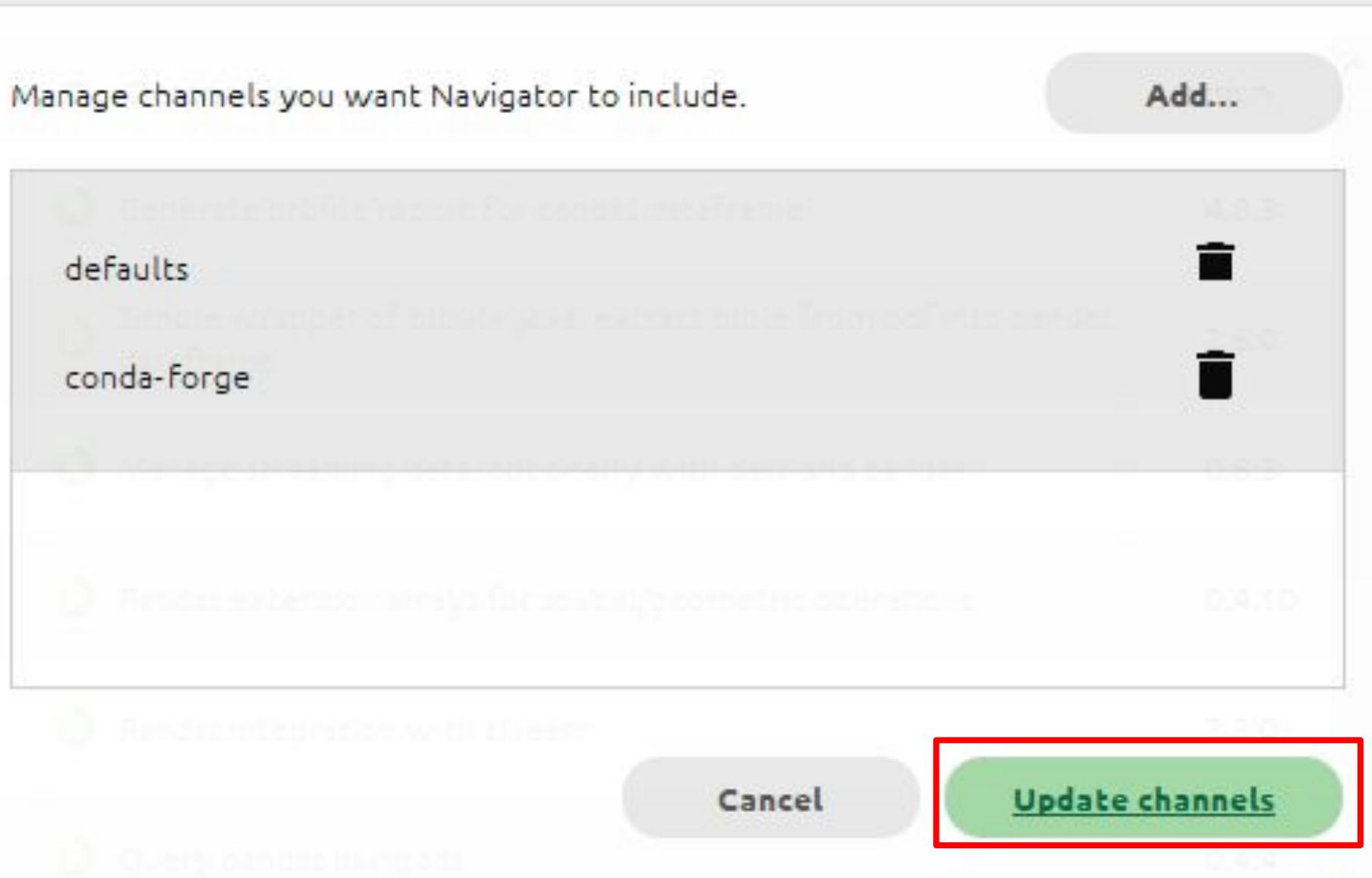
- ydata-profiling
- tabula-py
- streamz
- spatialpandas
- sklearn-pandas
- qpd

The main area displays a message: "Manage channels you want Navigator to include." Below this message, there is a list of channels:

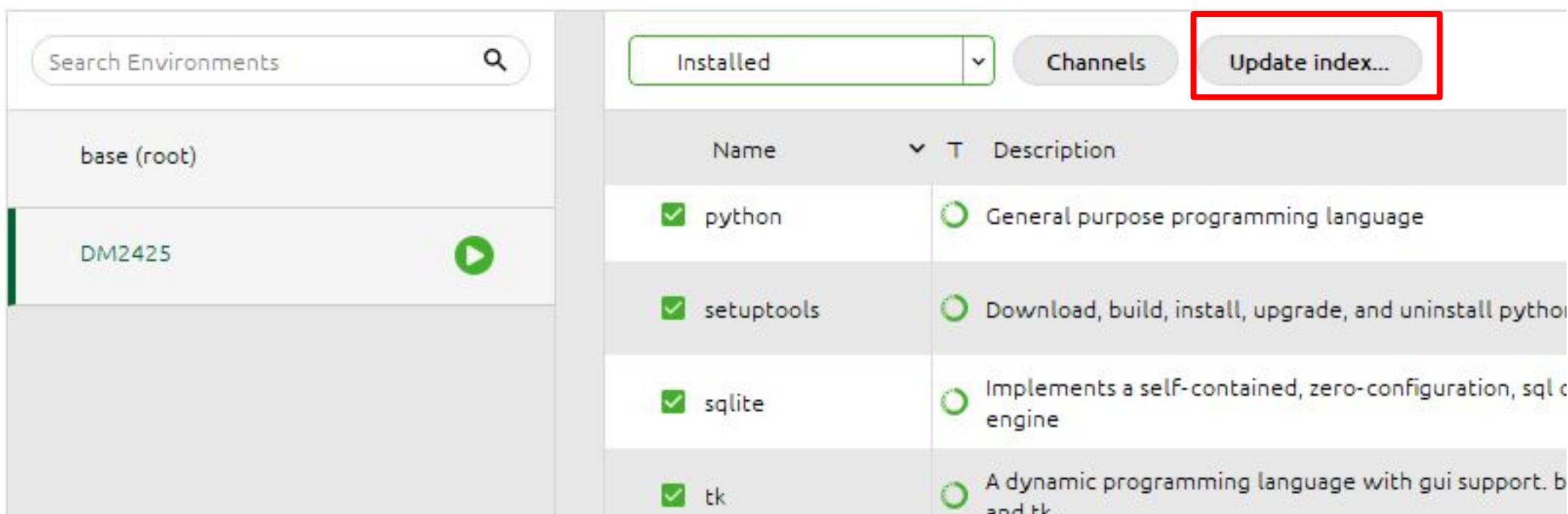
- defaults
- pandas
- pandas-dev
- pandas-datareader
- pandas-gui
- pandas-profiling
- pandas-tseries
- pandas-libs
- pandas-ml
- pandas-plotting
- pandas-timeindex
- pandas-tseries
- pandas-libs
- pandas-ml
- pandas-plotting
- pandas-timeindex

At the bottom right, there are "Cancel" and "Update channels" buttons. The "Update channels" button is highlighted with a green background.

Add conda channel: conda-forge



Add conda channel: conda-forge



The screenshot shows the Conda interface for managing Python environments. On the left, there's a sidebar with a search bar for environments and two environment entries: "base (root)" and "DM2425". The "DM2425" entry has a green play button icon next to it. The main area is titled "Installed" and contains a table of packages. The table has columns for Name, T (Type), and Description. The packages listed are python, setuptools, sqlite, and tk, all of which are checked (indicated by a green checkmark). The "Update index..." button is located at the top right of the main area, and it is highlighted with a red rectangular box.

Name	T	Description
python	✓	General purpose programming language
setuptools	✓	Download, build, install, upgrade, and uninstall python packages
sqlite	✓	Implements a self-contained, zero-configuration, sql database engine
tk	✓	A dynamic programming language with gui support. based on C and tk

Install libraries

File Help

ANACONDA.NAVIGATOR Connect ▾

Home Environments Learning Community

Search Environments base (root) DM2425

make sure to show “All”

Installed

- ✓ Installed
- Not installed
- Updatable
- Selected
- All

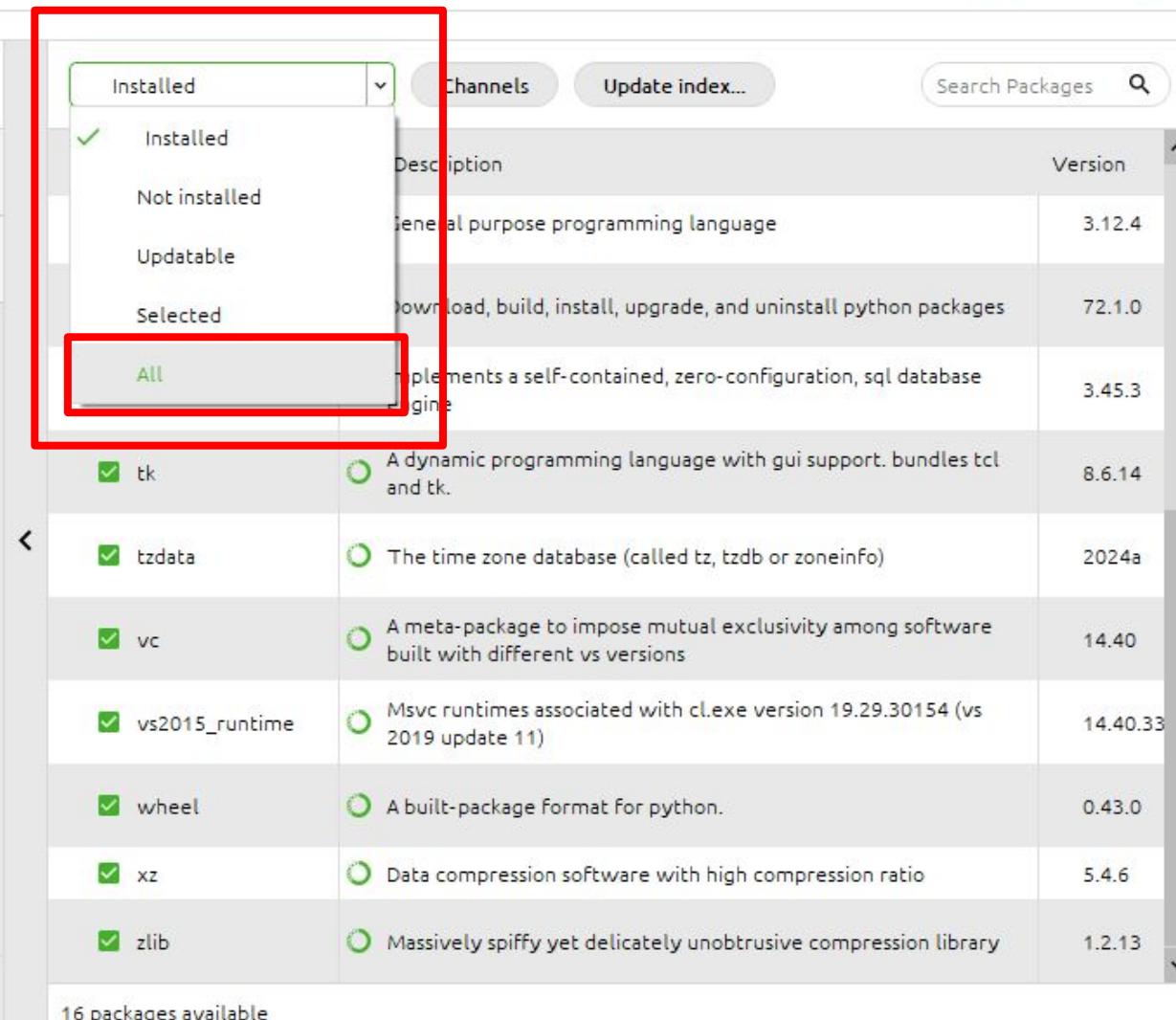
Description	Version
General purpose programming language	3.12.4
Download, build, install, upgrade, and uninstall python packages	72.1.0
Implements a self-contained, zero-configuration, sql database engine	3.45.3
A dynamic programming language with gui support. bundles tcl and tk.	8.6.14
The time zone database (called tz, tzdb or zoneinfo)	2024a
A meta-package to impose mutual exclusivity among software built with different vs versions	14.40
Msvc runtimes associated with cl.exe version 19.29.30154 (vs 2019 update 11)	14.40.33
A built-package format for python.	0.43.0
Data compression software with high compression ratio	5.4.6
Massively spiffy yet delicately unobtrusive compression library	1.2.13

16 packages available

Create Clone Import Backup Remove

Anaconda Toolbox Supercharged local notebooks. Click the Toolbox tile to Install. Read the Docs

Documentation Anaconda Blog



Install libraries

search for the libraries we need

Anaconda Navigator

Upgrade Now Connect

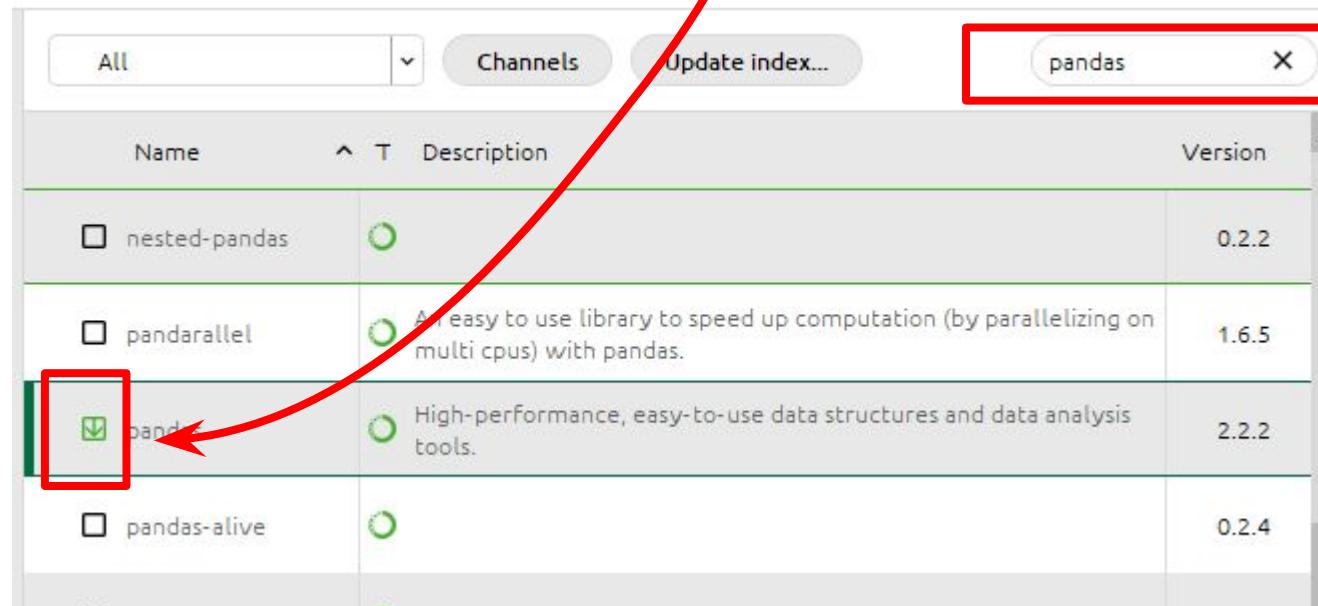
All Channels Update index... X

Name	Description	Version
ipywidgets	Jupyter interactive widgets	8.1.0
jupyter	Jupyter metapackage. install all the jupyter components in one go.	1.0.0
jupyter-archive		3.4.0
jupyter-black		0.3.4
jupyter-book		0.8.2
jupyter-cache		0.6.1

Install libraries

search for the libraries we need

- search package name, then select the check box on the left of the name



- after selecting, you can search the next library name, then select that, and so on

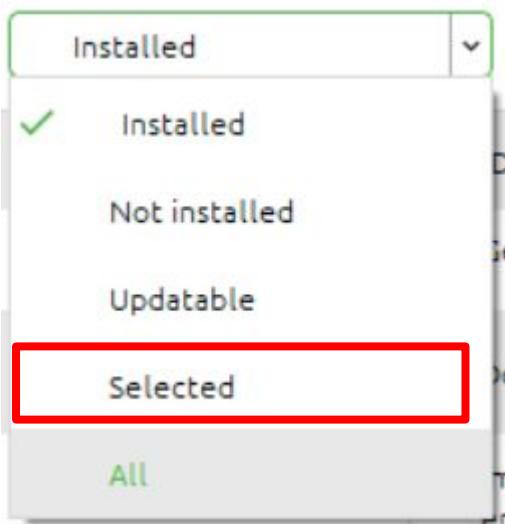
1. jupyter
2. scikit-learn
3. scikit-image
4. numpy
5. pandas
6. ipywidgets
7. matplotlib
8. seaborn
9. minisom
10. ydata-profiling

Install libraries

after searching and selecting everything on the list,

you can clear the search bar by clicking the ‘x’

then filter the “Selected” option to see if you got everything



The main interface shows the following table of packages:

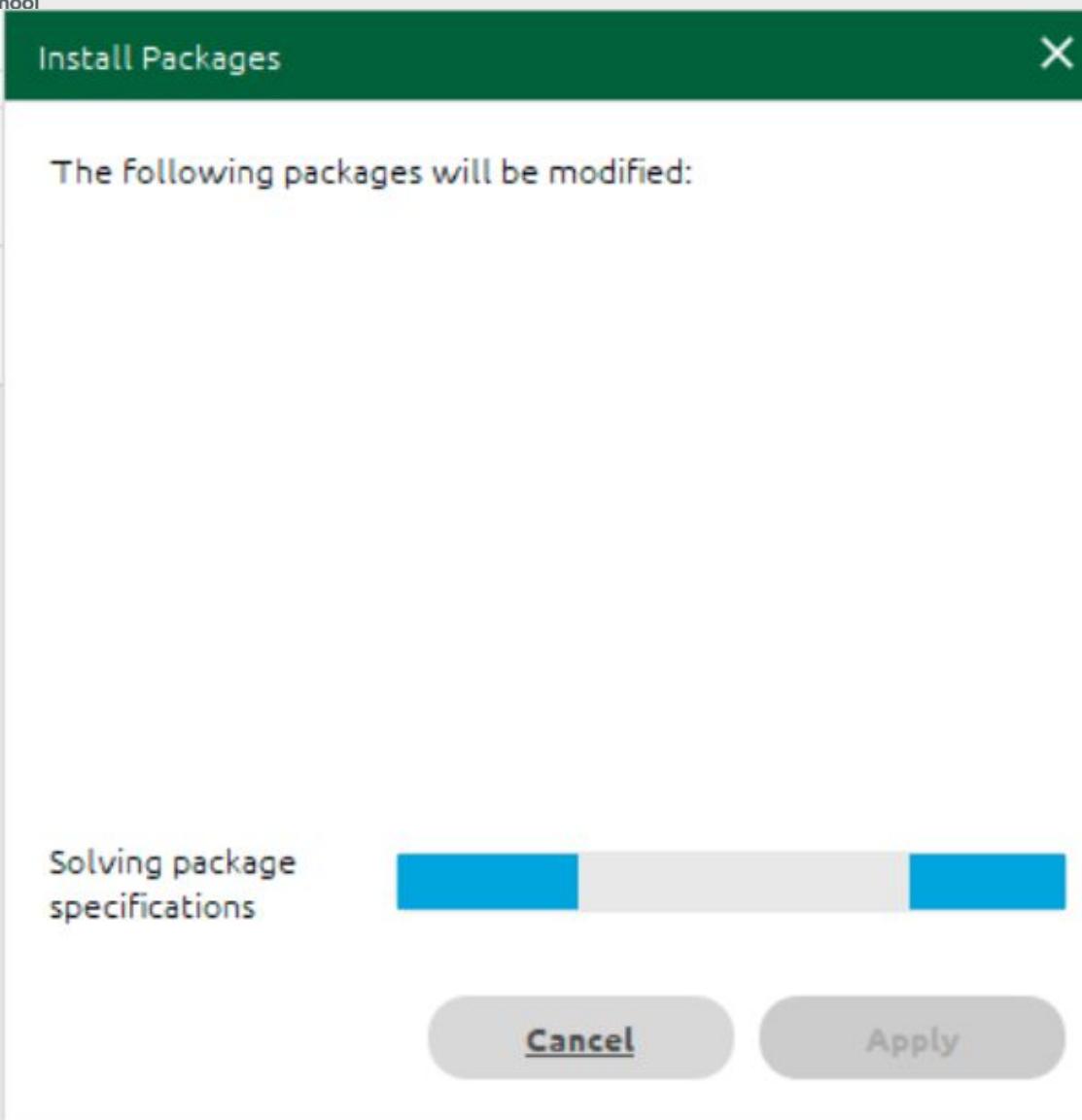
Name	Description	Version
git	Distributed version control system	2.42.0
scikit-learn	A set of python modules for machine learning and data mining	1.3.0
seaborn	Statistical data visualization	0.9.0

At the bottom, it displays: 10 packages available 10 packages selected.

A red arrow points from the "Selected" sidebar to the green "Apply" button.

then click Apply

Install libraries



15 packages will be installed

	Name	Unlink	Link	Channel	Action
1	*threadpoolctl	-	3.5.0	pkgs/main	Install
2	*tbb	-	2021.8.0	pkgs/main	Install
3	*scipy	-	1.13.1	pkgs/main	Install
4	*pybind11-abi	-	5	pkgs/main	Install
5	*numpy-base	-	1.26.4	pkgs/main	Install
6	*numpy	-	1.26.4	pkgs/main	Install

* indicates the package is a dependency of a selected package

[Cancel](#) [Apply](#)

Go back to home

Anaconda Navigator

File Help

ANACONDA.NAVIGATOR Connect ▾

Home Environments Learning Community

Installed applications DM2425 Channels

Anaconda Cloud Notebooks
Cloud-hosted notebook service from Anaconda. Launch a preconfigured environment with hundreds of packages and store project files with persistent cloud storage.

JupyterLab
4.2.5 An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

jupyter Notebook
7.2.2 Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch **Launch** **Launch**

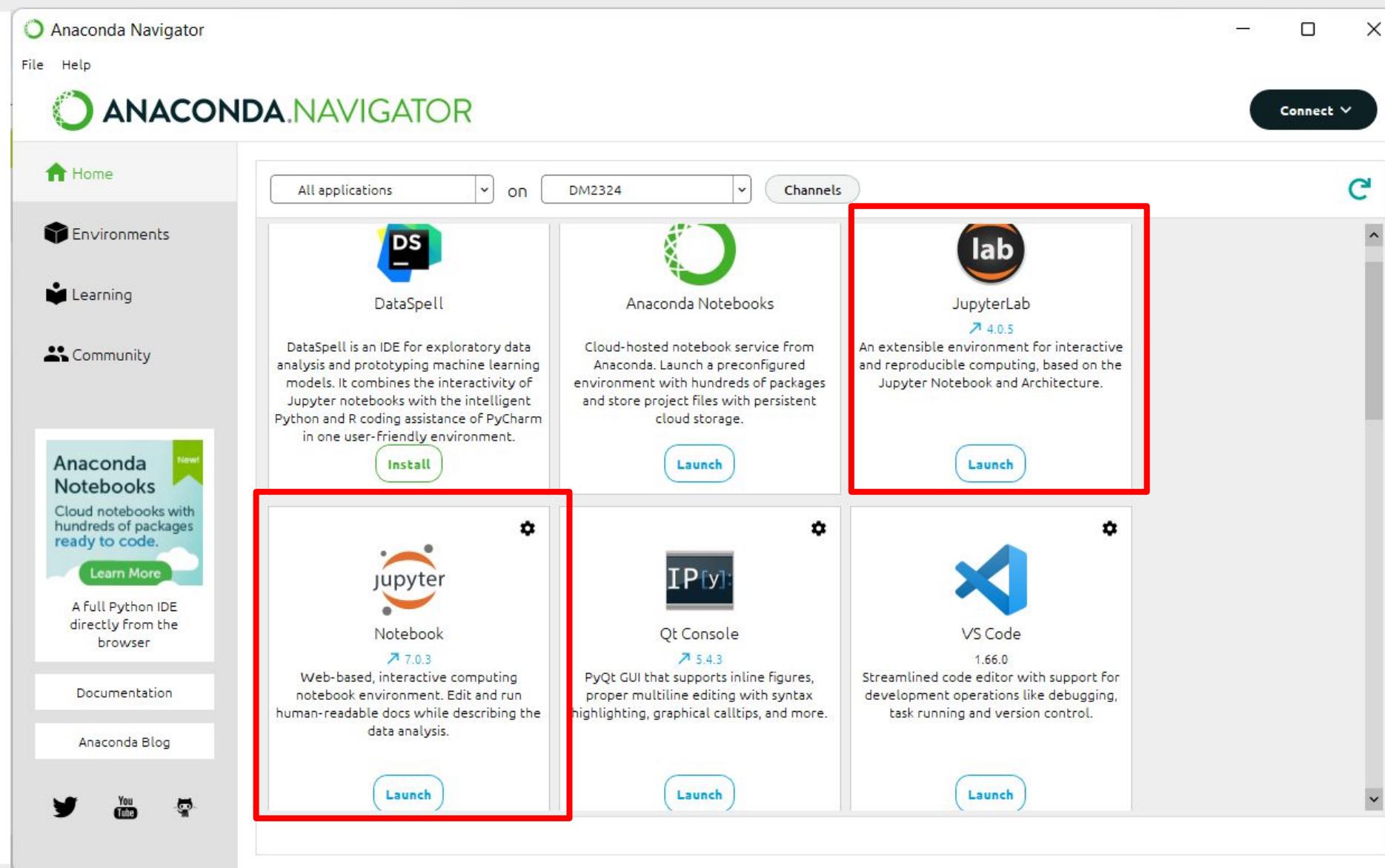
Anaconda Toolbox
Supercharged local notebooks.
Click the Toolbox

Test Jupyter Notebook

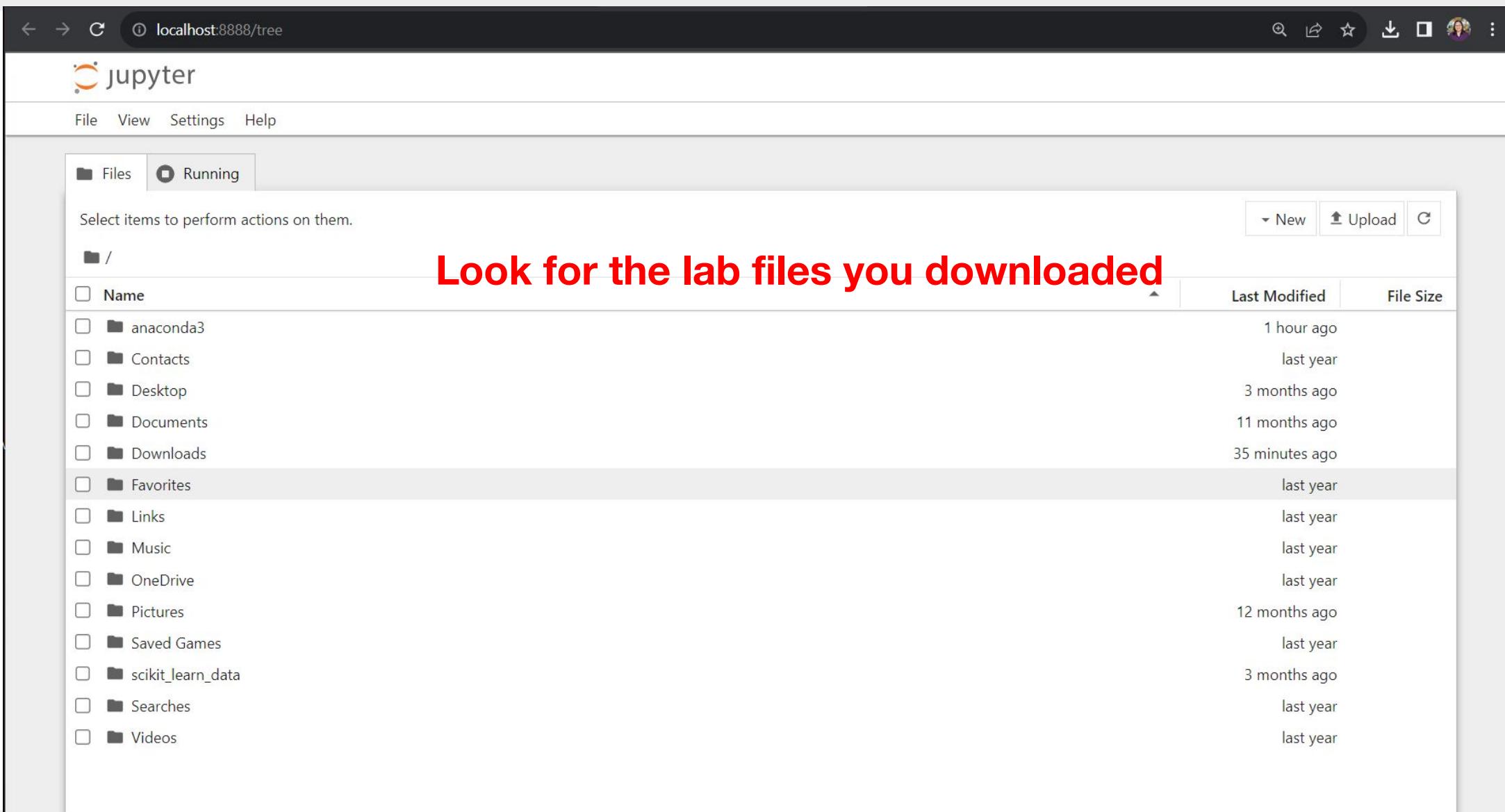
Test Jupyter notebook

Close all Terminal windows / Anaconda Navigator / Anaconda Prompt

Test loading Jupyter notebook



Test loading Jupyter notebook



The screenshot shows a Jupyter Notebook interface running locally. The title bar indicates the URL is `localhost:8888/tree`. The main area is titled "jupyter" and contains a file browser. The "Files" tab is selected, showing a list of standard system directories like anaconda3, Desktop, Downloads, Favorites, Links, Music, OneDrive, Pictures, Saved Games, scikit_learn_data, Searches, and Videos. A red banner with the text "Look for the lab files you downloaded" is overlaid in the center. The top right corner of the browser window shows user profile icons.

Name	Last Modified	File Size
anaconda3	1 hour ago	
Contacts	last year	
Desktop	3 months ago	
Documents	11 months ago	
Downloads	35 minutes ago	
Favorites	last year	
Links	last year	
Music	last year	
OneDrive	last year	
Pictures	12 months ago	
Saved Games	last year	
scikit_learn_data	3 months ago	
Searches	last year	
Videos	last year	

Load the lab01_setup.ipynb notebook file

The screenshot shows a Jupyter Notebook interface with the title "jupyter lab01_setup Last Checkpoint: 1 hour ago". The notebook contains a single section titled "Test if packages were installed correctly". Below the section title, there is a series of code cells. The first cell imports numpy and pandas. The second cell imports datasets from sklearn. The third cell imports matplotlib.pyplot and seaborn, and sets %matplotlib inline. It also configures the figure format to 'retina'. The fourth cell sets the seaborn style. The fifth cell prints the matplotlib version. The sixth cell ignores warnings. The seventh cell sets the logging level for matplotlib to WARNING.

```
[ ]: import numpy as np
[ ]: import pandas as pd

[ ]: from sklearn import datasets

[ ]: import matplotlib.pyplot as plt
[ ]: import seaborn as sns

%matplotlib inline

# for better resolution plots
%config InlineBackend.figure_format = 'retina'

# Setting seaborn style
sns.set()

[ ]: from matplotlib import __version__ as mplver
[ ]: print(mplver)

[ ]: import warnings
[ ]: warnings.filterwarnings('ignore')

[ ]: import logging
[ ]: logging.getLogger('matplotlib').setLevel(logging.WARNING)
```

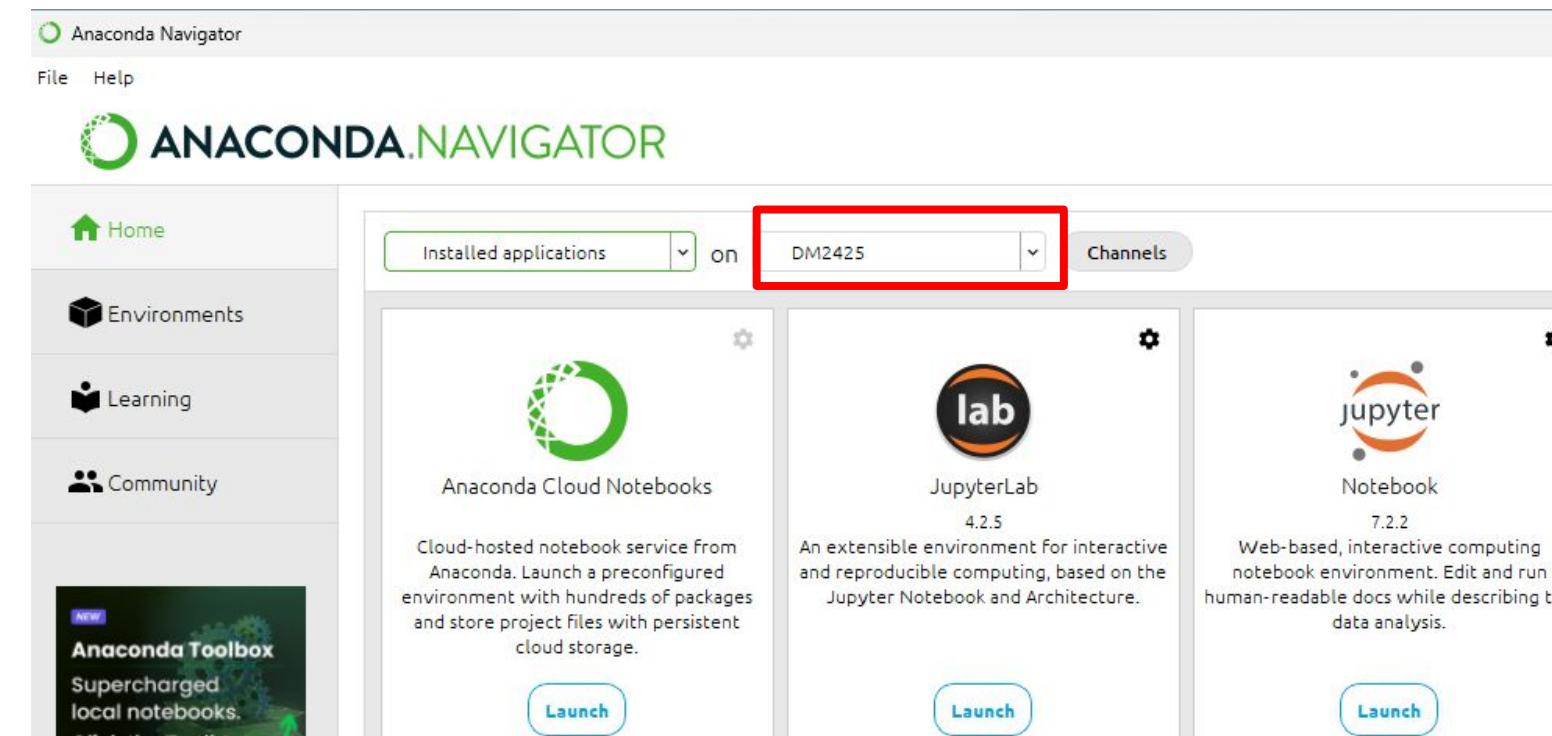
Don't worry about understanding the code at this point

Right now we just want to make sure that all the packages we need are installed and work correctly

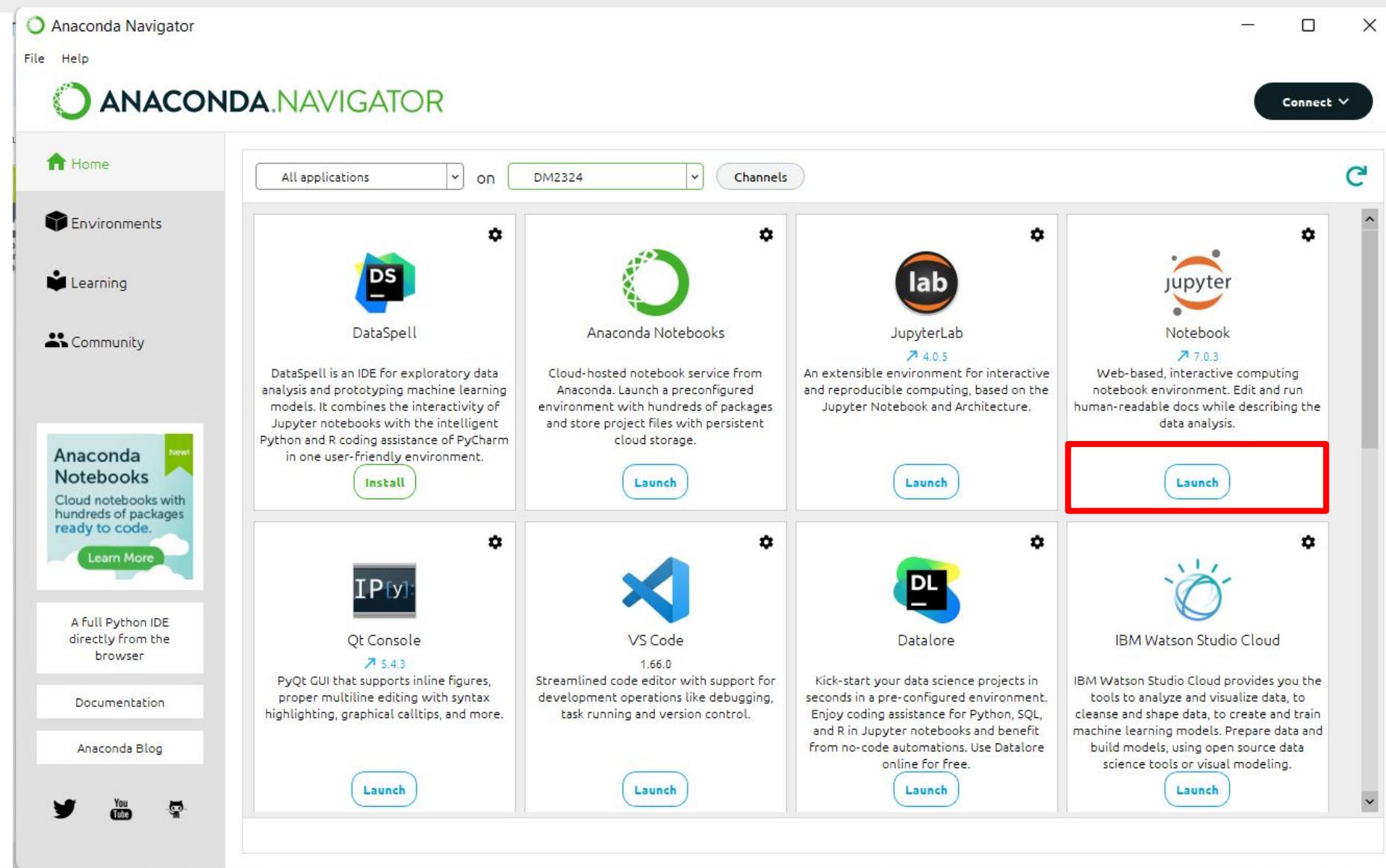
Test Jupyter notebook

Open Anaconda Navigator

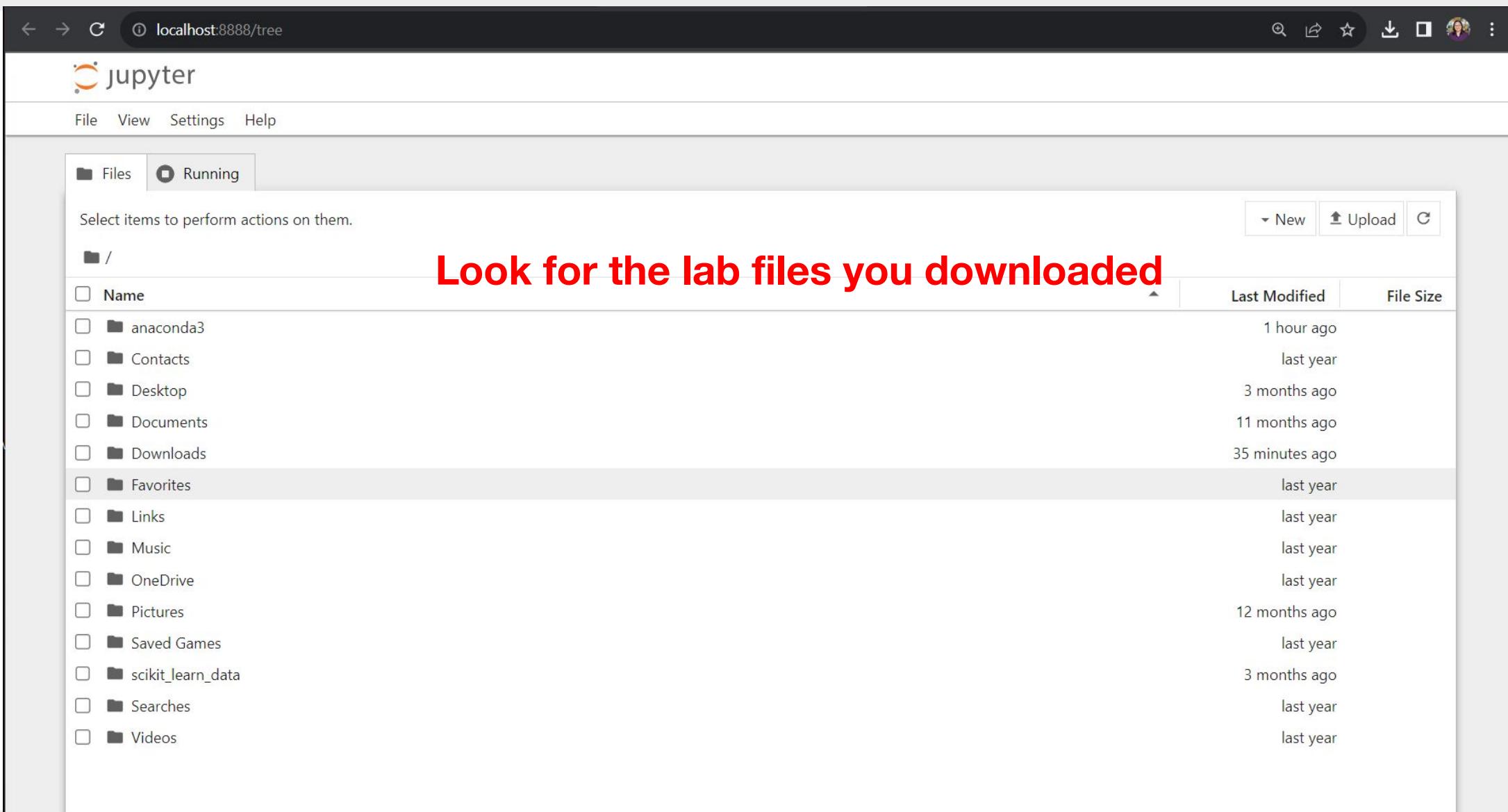
Make sure you select the environment we created



Test Jupyter notebook



Test Jupyter notebook



The screenshot shows a Jupyter Notebook interface running on a local server at `localhost:8888/tree`. The title bar includes standard browser controls like back, forward, and search, along with user profile and settings icons.

The main area displays a file tree under the heading "jupyter". A sidebar on the left lists common locations: anaconda3, Contacts, Desktop, Documents, Downloads, Favorites, Links, Music, OneDrive, Pictures, Saved Games, scikit_learn_data, Searches, and Videos. The "Running" tab is selected in the top navigation bar.

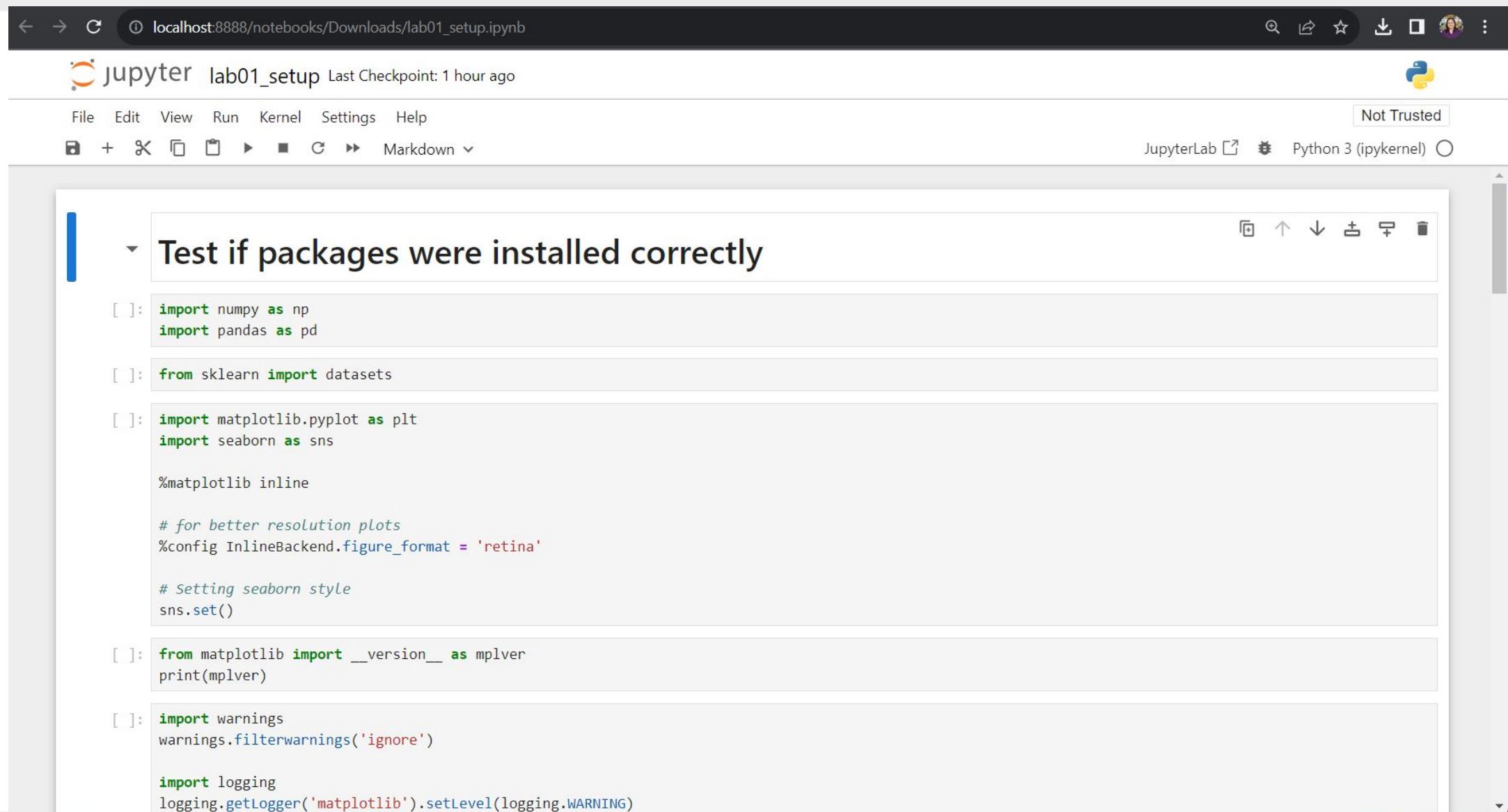
A large red text overlay in the center of the page reads "Look for the lab files you downloaded".

Name	Last Modified	File Size
anaconda3	1 hour ago	
Contacts	last year	
Desktop	3 months ago	
Documents	11 months ago	
Downloads	35 minutes ago	
Favorites	last year	
Links	last year	
Music	last year	
OneDrive	last year	
Pictures	12 months ago	
Saved Games	last year	
scikit_learn_data	3 months ago	
Searches	last year	
Videos	last year	

Don't worry about understanding the code at this point

Right now we just want to make sure that all the packages we need are installed and work correctly

Load the notebook file



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

- Title Bar:** localhost:8888/notebooks/Downloads/lab01_setup.ipynb
- Header:** jupyter lab01_setup Last Checkpoint: 1 hour ago, Python 3 (ipykernel)
- Toolbar:** File, Edit, View, Run, Kernel, Settings, Help, Not Trusted, JupyterLab, Python 3 (ipykernel)
- Cells:** The notebook contains several code cells:
 - []: `import numpy as np`
`import pandas as pd`
 - []: `from sklearn import datasets`
 - []: `import matplotlib.pyplot as plt`
`import seaborn as sns`

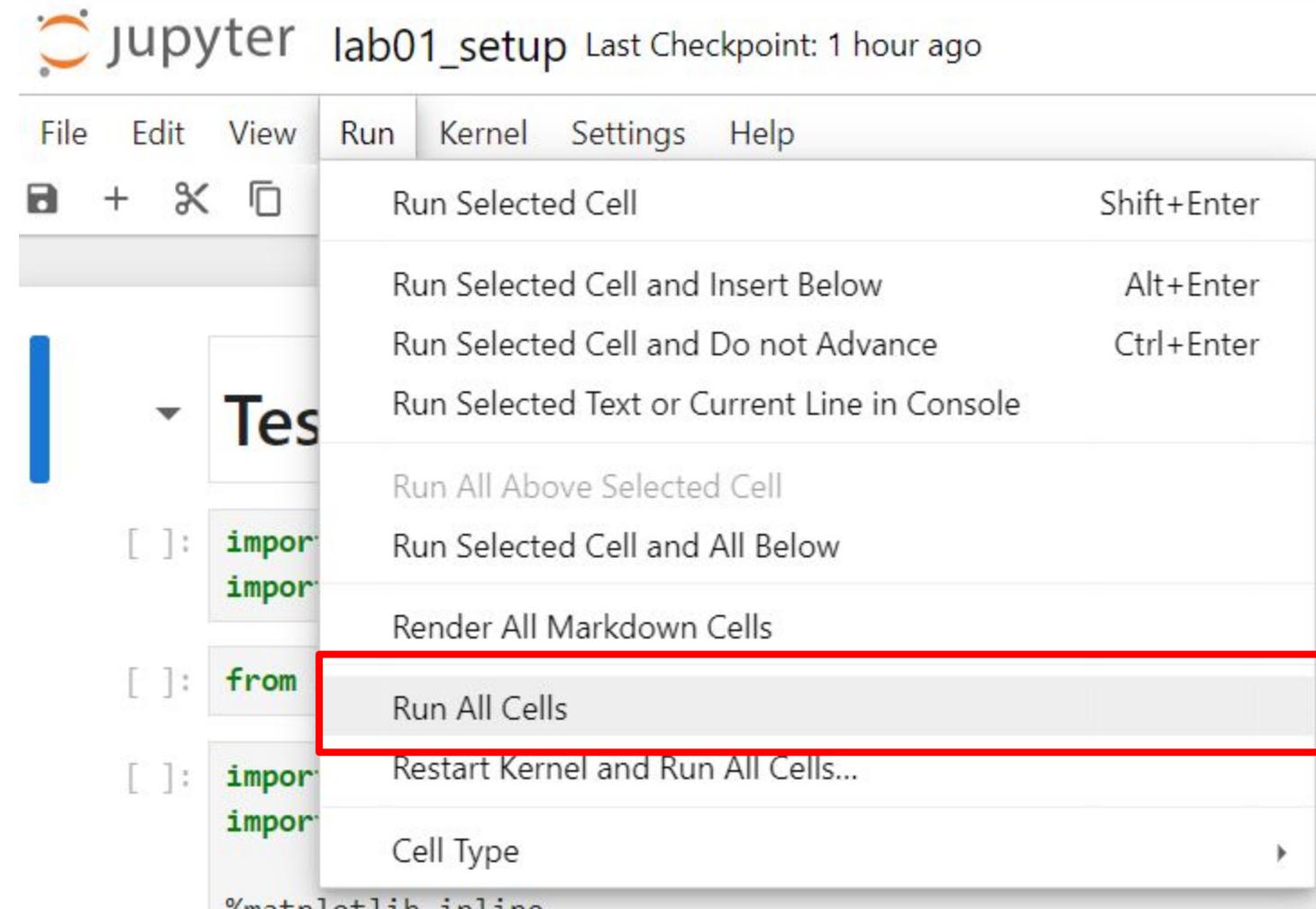
`%matplotlib inline`

`# for better resolution plots`
`%config InlineBackend.figure_format = 'retina'`

`# Setting seaborn style`
`sns.set()`
 - []: `from matplotlib import __version__ as mplver`
`print(mplver)`
 - []: `import warnings`
`warnings.filterwarnings('ignore')`

`import logging`
`logging.getLogger('matplotlib').setLevel(logging.WARNING)`

Run all cells



Does anyone still have any errors in the
notebook?

Installing using command line

You can also use miniconda
instead of Anaconda Navigator

<https://docs.anaconda.com/free/anaconda/getting-started/distro-or-miniconda/>

Installing using the command line

Install miniconda

<https://docs.conda.io/projects/miniconda/en/latest/#quick-command-line-install>

Create environment using the command line

Windows: Open Anaconda Prompt (miniconda3)

Linux/Mac: Open Terminal

```
cd Downloads
```

```
conda env create -f dm2425_env.yml
```

```
conda activate DM2425
```

Test Jupyter notebook + install other packages

Windows: Open Anaconda Prompt (miniconda3)

Linux/Mac: Open Terminal

```
conda activate DM2425
jupyter notebook
```

Follow the instructions in previous slides for testing Jupyter notebook and installing additional packages

Let's get started! (for real)

Next:
Jupyter notebook
Distance Matrix

Questions?

Morada: Campus de Campolide, 1070-312 Lisboa, Portugal

Tel: +351 213 828 610 | Fax: +351 213 828 611

Acreditações e Certificações da NOVA IMS



UNIGIS



Computing
Accreditation
Commission



Cofinanciado por

