Hyperiondev

TASK

Working with Datasets

Visit our website

Introduction

WELCOME TO THE WORKING WITH DATASETS TASK!

In the context of this task, when we refer to a dataset, we are referring to a collection of related data. This data can be manipulated in various ways programmatically. In this task, you will be using Pandas DataFrames to manipulate data.

JUPYTER

In this compulsory task, you will be using the Jupyter Notebook. This tool is **described as follows**: "The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualisations and narrative text. Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualisation, machine learning, and much more."

To use this tool, do the following:

1. **Install Jupyter**

    - **Option 1: Installing Jupyter with pip**

      First, ensure that you have the latest pip; older versions may have trouble with some dependencies:

      ```
      pip3 install --upgrade pip
      ```

      Then install the Jupyter Notebook using:

      ```
      pip3 install jupyter
      ```

    - **Option 2: Installing Jupyter using Anaconda**

        - Download **Anaconda**. We recommend downloading Anaconda's latest Python 3 version.

        - Install the version of Anaconda which you downloaded, following the instructions on the download page.

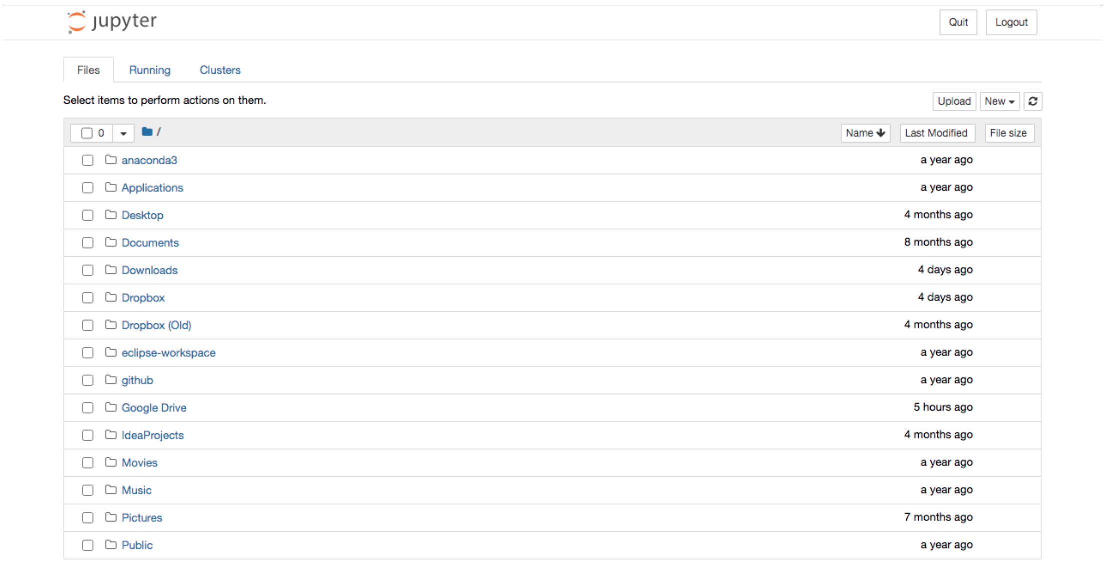
2. Run the Jupyter notebook

Once you have installed Jupyter, you can start the notebook server from the command line:

```
jupyter notebook
```

This will print some information about the notebook server in your terminal, including the URL of the web application. The notebook will then open in your browser.

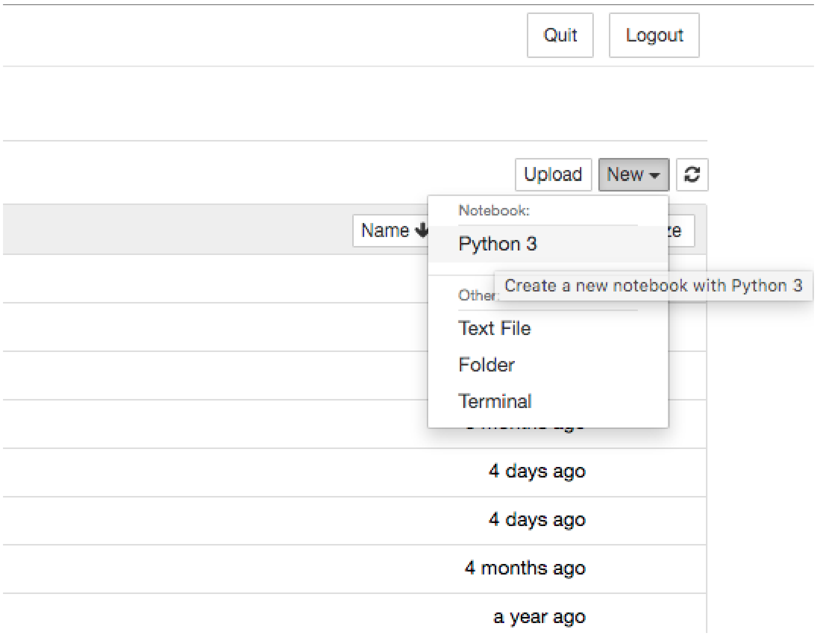
Once the notebook has opened, you should see the dashboard showing the list of notebooks, files and subdirectories in the directory you've opened. You can see an example of a Jupyter notebook below:



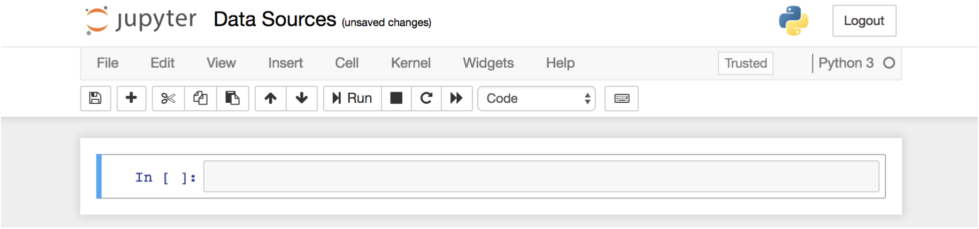
Select items to perform actions on them.

Name	Last Modified	File size
anaconda3	a year ago	
Applications	a year ago	
Desktop	4 months ago	
Documents	8 months ago	
Downloads	4 days ago	
Dropbox	4 days ago	
Dropbox (Old)	4 months ago	
eclipse-workspace	a year ago	
github	a year ago	
Google Drive	5 hours ago	
IdeaProjects	4 months ago	
Movies	a year ago	
Music	a year ago	
Pictures	7 months ago	
Public	a year ago	

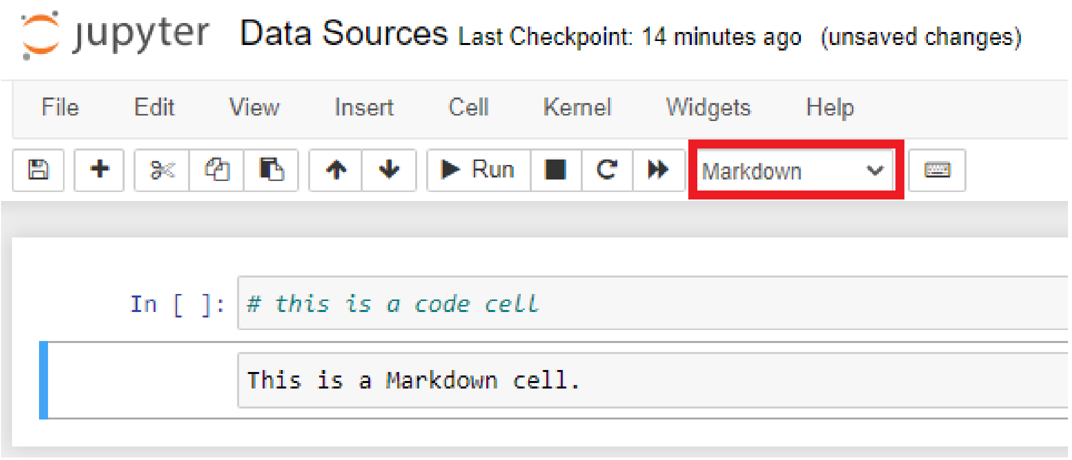
To start a new notebook, click the *New* drop-down menu and click on *Python 3*.



A new Jupyter notebook will look like the screenshot below. Make sure to change the name of the notebook to Data Sources.



In Jupyter notebooks you can specify whether a cell contains code or Markdown. Markdown is a lightweight Markup language that is used embed documentation or other textual information between code code cells.

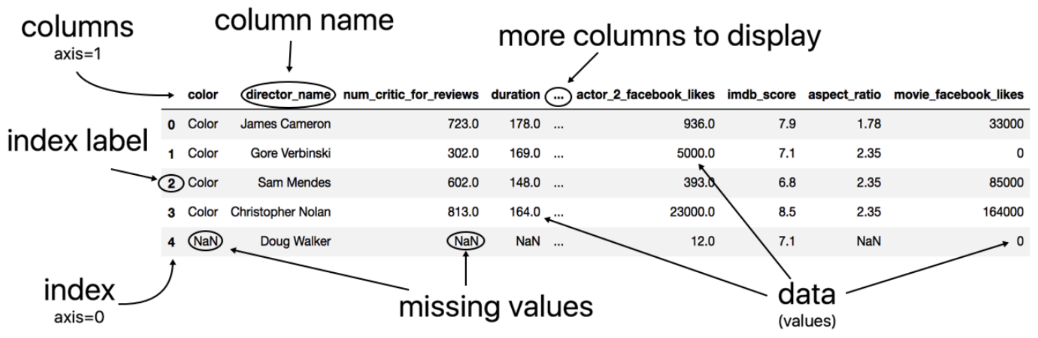

jupyter
Data Sources Last Checkpoint: 14 minutes ago (unsaved changes)
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Run
Markdown

```
In [ ]: # this is a code cell
```

```
This is a Markdown cell.
```

WHAT IS A PANDAS DATAFRAME?

The **pandas' library documentation** defines a DataFrame as a “two-dimensional, size-mutable, with labelled rows and columns”.



	color	director_name	num_critic_for_reviews	duration	...	actor_2_facebook_likes	imdb_score	aspect_ratio	movie_facebook_likes
0	Color	James Cameron	723.0	178.0	...	936.0	7.9	1.78	33000
1	Color	Gore Verbinski	302.0	169.0	...	5000.0	7.1	2.35	0
2	Color	Sam Mendes	602.0	148.0	...	393.0	6.8	2.35	85000
3	Color	Christopher Nolan	813.0	164.0	...	23000.0	8.5	2.35	164000
4	NaN	Doug Walker	NaN	NaN	...	12.0	7.1	NaN	0

Anatomy of a DataFrame

Image source: (Petrou, 2017)

In simple terms, think of a DataFrame as a table of data with the following characteristics (Lynn, 2018):

- “There can be multiple rows and columns in the data.
- Each row represents a sample of data,
- Each column contains a different variable that describes the samples (rows).

- The data in every column is usually the same type of data – e.g. numbers, strings, dates.
- Usually, unlike an excel data set, DataFrames avoid having missing values, and there are no gaps and empty values between rows or columns.”

You can read data from a .csv file into a DataFrame using the **`read_csv()`** function as shown below:

```
pd.read_csv('credit.csv', delimiter = ',')
```

However, there are also other functions that can be used to read data from different sources into DataFrames. For example, **`read_excel()`** can be used to read data from a spreadsheet file into a DataFrame, and **`read_sql()`** can be used to load data from a SQL database. Sometimes it is easier to extract data from other sources into a .csv file and read it into a DataFrame.

SELECTING COLUMNS IN PANDAS

There are many ways to specify columns in Pandas. The simplest way is to use dictionary notation for specific columns. In essence, Pandas Dataframes can be thought of as dictionaries: the key is the column name and the value is the corresponding column values.

```
import pandas as pd

import seaborn as sns


iris_df = sns.load_dataset('iris')

print(iris_df.columns)   #  ['sepal_length',  'sepal_width',  'petal_length',
'petal_width', 'species']

just_the_species = iris_df['species']
```

To select multiple columns, you simply need to specify a list of strings with each column name:

```
sepal_and_petal_info = iris_df[['sepal_length', 'sepal_width', 'petal_length',
'petal_width']]
```

You can also choose specific values to be included in your search (i.e. omit certain rows from the results).

```
small_sepal_length = iris_df[iris_df['sepal_length'] < 4.2]
```

In essence, we are filtering the dataset for all entries where the sepal_length is less than 4.2.

BUILT-IN DATAFRAME METHODS

When finding insights into your data, it is often useful to be able to use some kind of method to process your data. For example, finding the mean or total of a column. These are common statistical methods that are needed for any type of data analysis.

This is a list of common in-build methods in Pandas for such things:

- **mean()**: Computes the mean for each column
- **min()**: Computes the minimum for each column
- **max()**: Computes the maximum for each column
- **std()**: Computes the standard deviation for each column
- **var()**: Computes the variance for each column
- **nunique()**: Computes the number of unique values in each column

GROUPING IN PANDAS

Data analysis can sometimes get a bit complicated, and some more advanced functionality is needed. Let's say you want to average the insurance charges of all people between the ages of 30 and 35. This can be done quite easily using:

```
insurance_df = pd.read_csv("insurance.csv")


below_35 = insurance_df[insurance_df['age'] < 35]

between_30_and_35 = below_35[below_35['age'] > 30]


print(between_30_and_35['charges'].mean())
```

Now let's say you want to average the insurance charges of every person in each age group. This can still be done with the syntax you know, but it will take a lot of lines of code: this is bad, because we hate writing too many lines of code! Thankfully, Pandas provides us with something that allows us to do this with one line of code:

```
print(insurance_df.groupby('age')['charges'].mean())
```

This `groupby()` method tells the aggregation to work separately on each unique group specified.

Extra resource

*For more information about working with Jupyter, please consult the first chapter ("**IPython: Beyond Normal Python**") in the book entitled, "**Python Data Science Handbook**" by Jake VanderPlas.*

Instructions

- Follow the instructions in this task to install Jupyter Notebook.
- In your command line interface, change directory (`cd`) to the current folder.
- Open Jupyter Notebook by typing: `jupyter notebook`
- Within this task folder, you will find Jupyter notebook examples. You can open and explore them by going to Jupyter's home screen and double-clicking on the notebook.
    - **Compulsory task 1:** Dataset Examples.ipynb
    - **Compulsory task 2:** Report_Example.ipynb

Compulsory Task 1

Open the **Datasets Compulsory task.ipynb** file and complete the following tasks in the notebook. Save your notebook to your task folder for submission.

1. Write the code that performs the action described in the following statements.
   a. Select the 'Limit' and 'Rating' columns of the first five observations
   b. Select the first five observations with 4 cards
   c. Sort the observations by 'Education'. Show users with a high education value first.
2. Write a short explanation in the form of a comment for the following lines of code.
   a. `df.iloc[:,:]`
   b. `df.iloc[5:,5:]`
   c. `df.iloc[:,0]`
   d. `df.iloc[9,:]`

Compulsory Task 2

Open and run the example file for this task in VS Code before attempting this task. Follow these steps:

- Open the **Report.ipynb** in this folder.
- Create a DataFrame that contains the data in **balance.txt**.
- Write the code needed to produce a report that provides the following information:
  - Compare the average income based on ethnicity.
  - On average, do married or single people have a higher balance?
  - What is the highest income in our dataset?
  - What is the lowest income in our dataset?

- How many cards do we have recorded in our dataset? (Hint: use `sum()`)
- How many females do we have information for vs how many males? (Hint: use `count()`. For a list of all methods for computation of descriptive stats, explore the **pandas documentation**.)

Completed the task(s)?

Ask an expert to review your work!

Review work

Rate us

Share your thoughts

HyperionDev strives to provide internationally-excellent course content that helps you achieve your learning outcomes.

Think that the content of this task, or this course as a whole, can be improved? Do you think we've done a good job?

Click here to share your thoughts anonymously.

REFERENCES

Lynn, S. (2018). The Pandas DataFrame – loading, editing, and viewing data in Python. Retrieved from Shane Lynn: Pandas Tutorials: **https://www.shanelynn.ie/using-pandas-dataframe-creating-editing-viewing-data-in-python/**

Jupyter Team. (2015). Running the Notebook — Jupyter Documentation 4.1.1 alpha documentation. Retrieved 18 August 2020, from **https://test-jupyter.readthedocs.io/en/latest/running.html**

Petrou, T. (2017, October 27). Dissecting the anatomy of a DataFrame. (Packt>) Retrieved from Pandas Cookbook: Recipes for Scientific Computing, Time Series Analysis and Data Visualization using Python: **https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781784393878/1/ch01lvl1sec12/dissecting-the-anatomy-of-a-dataframe**