CoCalc (Collaborative Calculation; free plan): https://cocalc.com/

1. Software compatibility
    a. with GitHub: incompatibile (couldn't run notebooks directly from GitHub, or import a single notebook from GitHub)
    b. Other sources
2. User friendliness:
    a. Configurations / traffic: 1-core shared CPU
        i. 1000MB of shared RAM
        ii. 3000MB of disk space (per project).
        iii. Sessions will shut down after 30 minutes of inactivity, though they can run for up to 24 hours.
    b. Packages (https://cocalc.com/doc/software.html)
        i. additional packages: by request (not available for free plan)
        ii. pre-installed for Python: https://cocalc.com/doc/software-python.html
    c. Real-time collaboration: Yes
    d. Shareable: Yes
    e. Keep as a private file: Yes
    f. Internet access: not available for a free plan
    g. Shortcuts: 95% similar to Jupyter
    h. **Version Control**: provides version control, "time travel", with excellent functionality
3. **Supporting languages (JupyterNotebook)**: using Linux terminal, CoCalc supports softwares including Bash, C++, C++ 17, GP, Haskell, JavaScript, Octave, Prolog, Python 2&3, R, Sagemath, Singular, Sparql
4. Customizability (&ease) of the configuration file:
    a. .ipynb
    b. .txt
    c. .html
    d. .md
    e. .rst
    f. .tex
    g. .pdf
    h. .sagews
    i. .asciidoc
5. Speed of deployment (descriptions from the official website): "It takes less time to run a Jupyter Notebook than grabbing a cup of coffee!" (Zero setup, web-browser based)
6. Configurations

a. Python Environments

   i. already installed: https://cocalc.com/doc/software-python.html

   ii. request to install (only available for paid users): https://doc.cocalc.com/howto/install-python-lib.html ### ML-predict-drugclass-master It is impossible to run the provided JupyterNotebook under the free plan, since half of the libraries are not installed. Installing those libraries requires membership and installation requests.

b. ML-predict-drugclass-master

   i. Able to import:

      1. **import os**
      2. **import random**
      3. **import time**
      4. **import warnings**
      5. **import numpy as np**
      6. **import pandas as pd**
      7. **import matplotlib.pyplot as plt**
      8. **from pprint import** pprint
      9. **from collections import** Counter
      10. **from concurrent import** futures
      11. **from IPython.core.display import** Image
      12. **from sklearn import** preprocessing
      13. **from sklearn.ensemble import** RandomForestClassifier
      14. **from sklearn.model_selection import** train_test_split
      15. **from sklearn import** metrics
      16. **from sklearn.metrics import** balanced_accuracy_score
      17. **from sklearn.metrics import** plot_confusion_matrix

   ii. Need to request to install: *rdkit, py3Dmol, mordred*

      1. **import py3Dmol**
      2. **from mordred import** Calculator, descriptors
      3. **from rdkit import** DataStructs
      4. **from rdkit import** Chem
      5. **from rdkit.Chem import** AllChem, Draw, rdDepictor, PandasTools
      6. **from rdkit.Chem.Draw import** IPythonConsole
      7. **from rdkit.Chem.Draw.MolDrawing import** MolDrawing, DrawingOptions

c. ML-TC1-master (Due to the limited capacity of RAM (1000 MB) and high requirements of CNN, the kernel crashed while running cell 19.)

        i. All libraries are able to be imported

7. Publications related to CoCalc
    a. The CoCalc Wiki Pages (GitHub): https://github.com/sagemathinc/cocalc/wiki
    b. Official
        i. User Manual: https://doc.cocalc.com/contents.html
        ii. Instructor guide: https://doc.cocalc.com/teaching-instructors.html
        iii. Student guide: https://doc.cocalc.com/teaching-students.html
8. How effective is it to use these cloud deployments for running a workshop/teaching? https://doc.cocalc.com/teaching-instructors.html
    a. **Managed platform:** no time is wasted setting up, maintaining, backing up and securing a flaky platform. Instead, you can sleep while CoCalc ensures everything runs smoothly 24/7.
    b. **Focus on helping students:** Jump right into a student's file and assist via a chat on the side.
    c. **Proven track record:** CoCalc has been used for teaching since 2013 in hundreds of courses by tens of thousands of students.
    d. **Open source:** CoCalc itself and every software it offers is open-source. Say good-bye to vendor lock-in, hidden licensing fees, and proprietary black boxes.
    e. **Student work is isolated:** Each student does course work in a separate project.
9. **Summary**
    a. Pros & Cons¶
        i. Pros
            1. 95% similarity to JupyterNotebook (easy to use)
            2. provides version control, "time travel", with excellent functionality
            3. supports more than 10 softwares and configurations files
            4. easy to deploy (quick to set up)
        ii. Cons
            1. incompatibility with GitHub
            2. limited shared RAM (1000MB) and disk space (3000MB)
            3. installing additional packages requires membership and request submissions
    b. Overall, I do not think CoCalc is an efficient and effective tool for teaching a Machine Learning workshop with following reasons:
        i. Installing packages requires membership and is time-consuming.
        ii. CoCalc has relatively smaller shared RAM and Storage, running complex models (such as Convolutional neural network) will results in a killed kernel.
    c. Nevertheless, teaching more basic knowledge of Python is practical, with following available packages:

i. Python: https://cocalc.com/doc/software-python.html
ii. R: https://cocalc.com/doc/software-r.html