COMP3314 Tutorial 1

Basics for Python Programming & Assignment 1

2024-09-20

COMP3314 TAs

Agenda

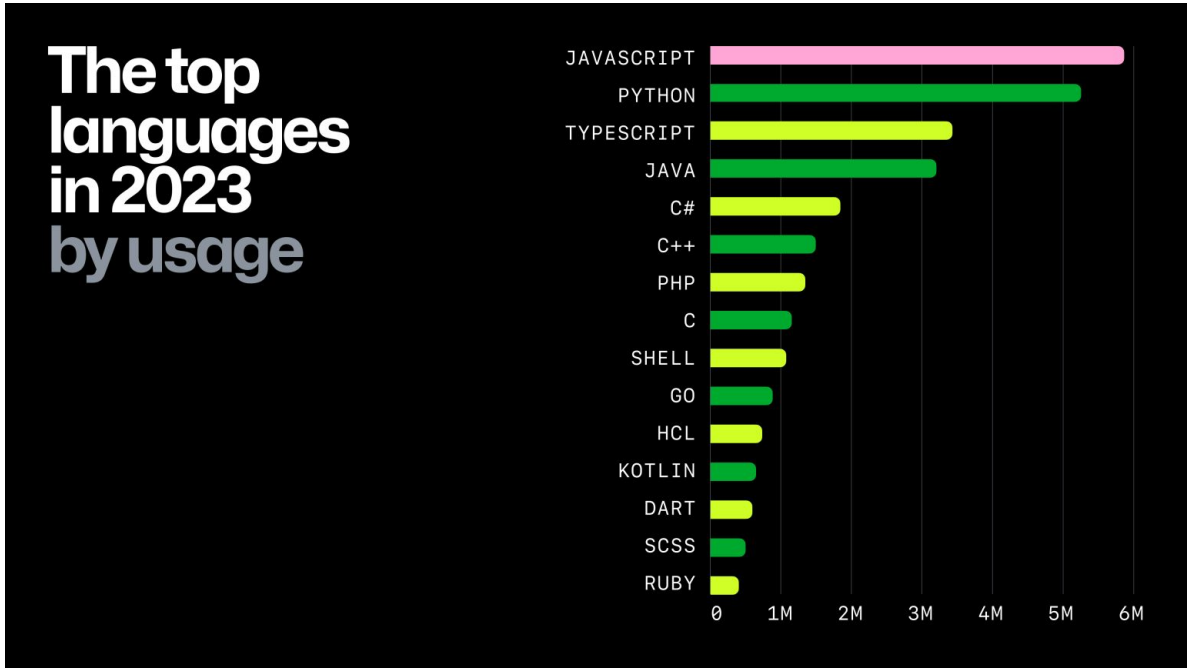
- Introduction
- Basics of Python
  - Setting Up Python Environment
  - Installing Miniconda
  - Managing Python Virtual Environments
  - Installing Libraries
- Development environment
  - Introduction to Jupyter Notebook
  - Leveraging Google Colab
  - Using Visual Studio Code for Python Development
- Overview of Assignment 1
- Summary and Q&A

Basics of Python

- What is Python?
  - High-Level, Interpreted Language: Known for simplicity and readability.
  - Created by Guido van Rossum in the late 1980s.
  - Purpose: Designed for ease of use, quick application development.
- Key Features of Python
  - Intuitive Syntax: Ideal for beginners.
  - Versatile Use: From web development to automation.
  - Open Source: With a large, supportive community.
  - Rich Libraries: For data analysis, ML, scientific computing.
  - Interpreted Nature: Facilitates quick prototyping.
- Python in Machine Learning & Data Science
  - Dominant Language: Due to simplicity and powerful libraries.
  - Strong Community Support: Resources and forums for learning.
  - Efficient for Prototyping: Quick experimentation with ML models.

One of the most popular programming languages

- Python is the 2# most popular programming language on GitHub



Source: https://github.blog/news-insights/research/the-state-of-open-source-and-ai/#the-most-popular-programming-languages

Installing Miniconda

- What is a Python virtual environment?
  - A virtual environment is a "container" for of multiple installed Python libraries and executables
  - Best practice: use separate environment for each project
- What is Miniconda?
  - A popular tool for managing Python virtual environment
  - Miniconda is the "mini" version of conda, recommended for general use
- Installing Miniconda
  - Find the proper version for your OS and follow the steps
    - https://docs.conda.io/projects/miniconda/en/latest/
  - Optional: prevent conda from activating base automatically
    - https://stackoverflow.com/a/54560785/1255535
      - conda config --set auto_activate_base false
- Live demo for installation on macOS/Linux
  - Please refer to: https://asciinema.org/a/YhEyleUmEHeKfPRKIX4nxlKuK
- Windows installation
  - Please refer to: https://www.youtube.com/watch?v=oHHbsMfyNR4

Demo recording: https://asciinema.org/a/YhEyleUmEHeKfPRKIX4nxlKuK

```
multipass shell air
```

Managing Python Virtual Environments with Conda

- Creating a new virtual environment
  - `# Create an environment called "demo"`
  - `conda create -n demo python=3.8`
- Activating and deactivating environments
  - `# Check existing environment`
  - `conda env list`
  - `# Activate "demo" environment`
  - `conda activate demo`
  - `# Check python version`
  - `python --version`
  - `# Deactivate environment`
  - `conda deactivate`

Installing Python Libraries

- Introduction to pip and conda
- Common libraries for machine learning
  - NumPy, scikit-learn, PyTorch, TensorFlow, Jupyter
- Installing libraries using pip commands
  - `# Activate your virtual environment first!`
  - `conda activate demo`
  - `# Install Python libraries`
  - `pip install numpy`
  - `pip install scikit-learn`
  - `pip install jupyter`
  - `...`

NumPy

- What is NumPy?
  - NumPy: A fundamental package for numerical computation in Python.
  - Core Feature: Multidimensional array object (ndarray).
  - Purpose: Optimized for numerical operations, linear algebra, random number capabilities.
- Key Features of NumPy
  - Efficient Array Computing: Fast, memory-efficient array processing.
  - Mathematical Functions: Comprehensive mathematical functions.
  - Interoperability: Works well with other libraries.


```
import numpy as np

# Creating a NumPy array
arr = np.array([1, 2, 3, 4, 5])

# Performing element-wise operations
squared = arr ** 2

# Computing basic statistics
mean_value = np.mean(arr)

print(f"Original Array: {arr}")
print(f"Squared Array: {squared}")
print(f"Mean Value: {mean_value}")
```

```
Original Array: [1 2 3 4 5]
Squared Array: [ 1  4  9 16 25]
Mean Value: 3.0
```

scikit-learn

- What is Scikit-Learn?
  - Scikit-Learn: A Python library for machine learning.
  - Purpose: Offer simple and efficient tools for data mining and data analysis.
- Key Features of Scikit-Learn
  - Wide Range of Algorithms: Classification, regression, clustering, etc.
  - Data Preprocessing Tools: Feature scaling, normalization, .etc.
  - Model Evaluation: Cross-validation, metrics for performance evaluation.

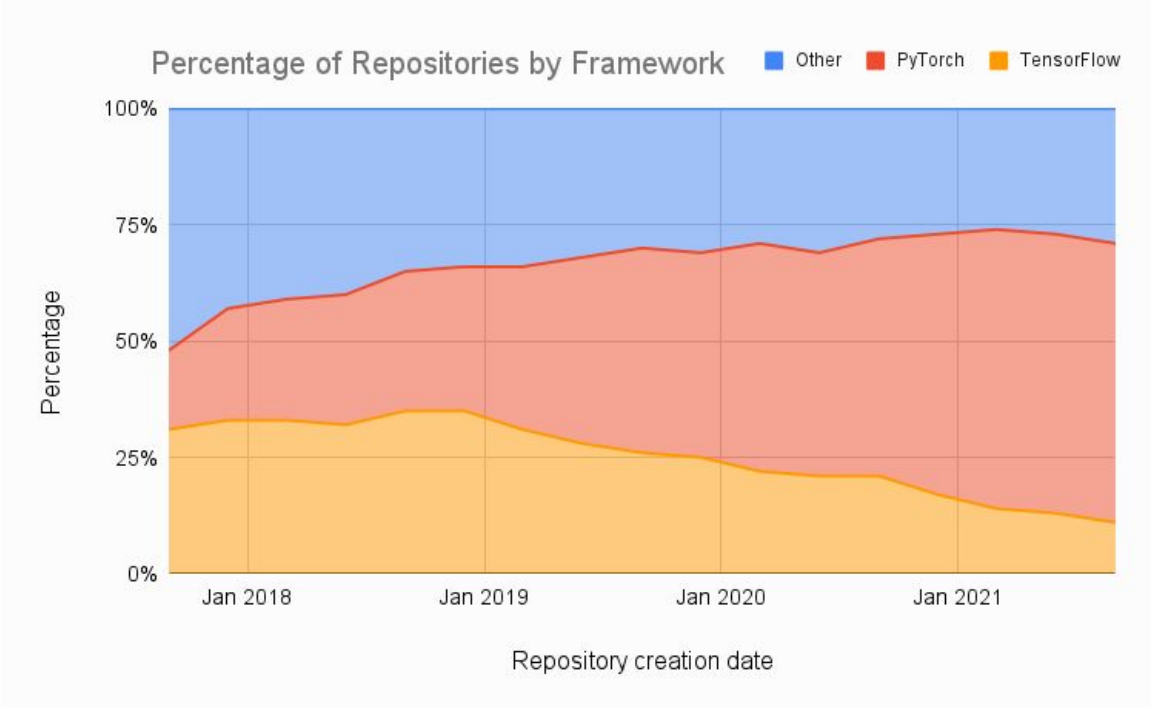
```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import
train_test_split
from sklearn.metrics import accuracy_score

# Load dataset
iris = load_iris()
X, y = iris.data, iris.target

# Split dataset
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.3)

# Train a model
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)

# Predict and evaluate
predictions = classifier.predict(X_test)
accuracy = accuracy_score(y_test, predictions)
```

PyTorch

- What is PyTorch?
  - PyTorch: An open-source machine learning library developed by Facebook's AI Research lab.
  - Purpose: Preferred for deep learning and artificial intelligence projects.
  - Features: Dynamic computational graph and tensor computation with strong GPU acceleration.
- Key Features of PyTorch
  - Dynamic Computation Graphs: Flexibility and ease in defining and modifying neural networks.
  - Tensor Library: Similar to NumPy, but with GPU support.
  - Autograd Module: Automatic differentiation for gradient calculations.


```
import torch
import torch.nn as nn
import torch.optim as optim

# Simple neural network
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc = nn.Linear(1, 1)

    def forward(self, x):
        return self.fc(x)

# Create a model, criterion and optimizer
model = Net()
criterion = nn.MSELoss()
optimizer = optim.SGD(model.parameters(), lr=0.01)

# Dummy data
inputs = torch.tensor([[1.0], [2.0], [3.0]])
targets = torch.tensor([[2.0], [4.0], [6.0]])

# Forward pass, backward pass, optimize
optimizer.zero_grad()
outputs = model(inputs)
loss = criterion(outputs, targets)
loss.backward()
optimizer.step()

print(f"Loss: {loss.item()}")
```

paperswithcode.com repository trend 2022



PyTorch is currently the leading deep learning library for research

Ref: https://www.assemblyai.com/blog/pytorch-vs-tensorflow-in-2023/

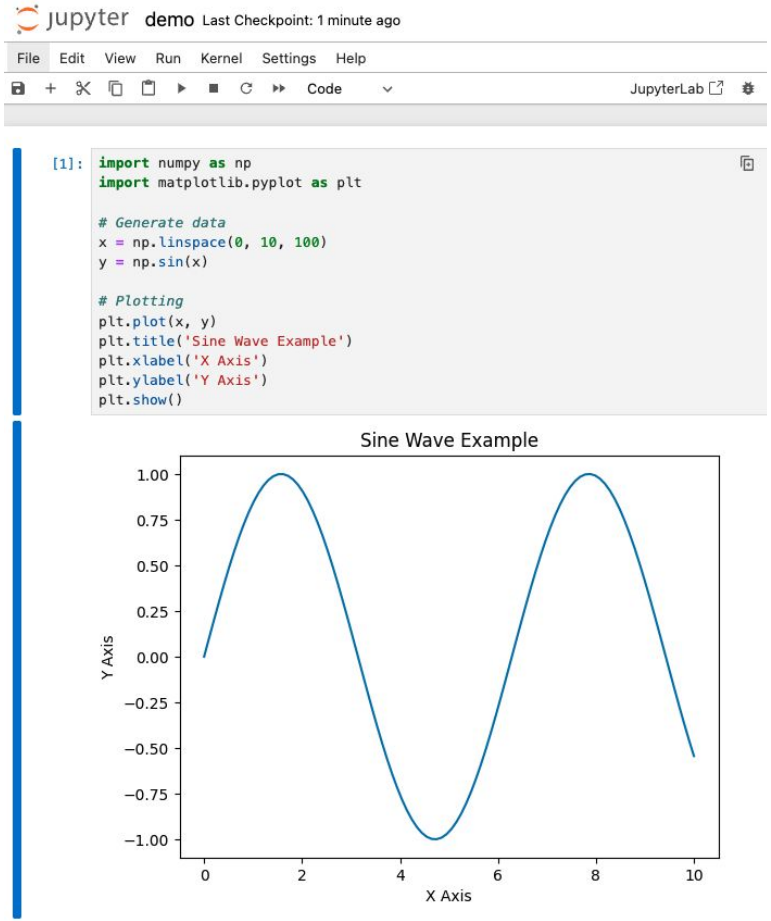
TensorFlow

- What is TensorFlow?
  - An open-source library developed by Google for numerical computation and machine learning.
  - Primarily used for deep learning applications.
- Key Features of TensorFlow
  - High-Level APIs: Simplifies tasks in machine learning.
  - Scalability: Runs on CPUs, GPUs, and TPUs.
  - Tensor Processing: Efficient handling of multi-dimensional arrays.
- When to Choose TensorFlow
  - TensorFlow is often preferred for large-scale deployments and applications requiring robust production pipelines.
  - TensorFlow, backed by Google, offers more extensive enterprise support, making it a reliable choice for businesses.

```
import tensorflow as tf

# Define a simple sequential model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(units=1, input_shape=[1])
])

# Compile the model
model.compile(optimizer='sgd',
loss='mean_squared_error')

# Example data
X = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0],
dtype=float)
y = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0],
dtype=float)

# Train the model
model.fit(X, y, epochs=50)
```

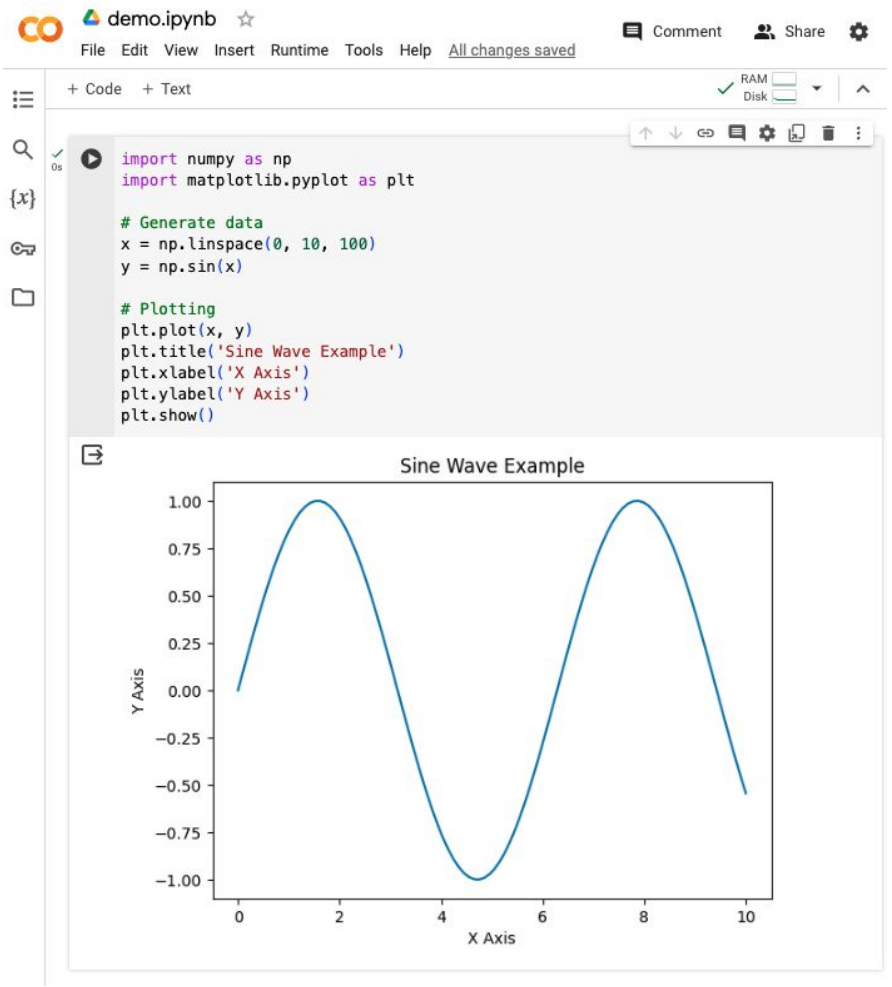
Introduction to Jupyter Notebook (Jupyter Lab)

- What is Jupyter Notebook?
  - An open-source **web** application that allows you to create and share documents containing live code, equations, visualizations, and narrative text.
- Key Features of Jupyter Notebook
  - **Interactive** Environment: Run code in real-time, view results inline.
  - Integration of Code and Rich Text: Combine code, visuals, and text in a single document.
- Installing and starting Jupyter
  - Jupyter notebook
    - `pip install jupyter`
    - `jupyter notebook`
  - Jupyter lab
    - `pip install jupyterlab`
    - `jupyter lab`



Leveraging Google Colab

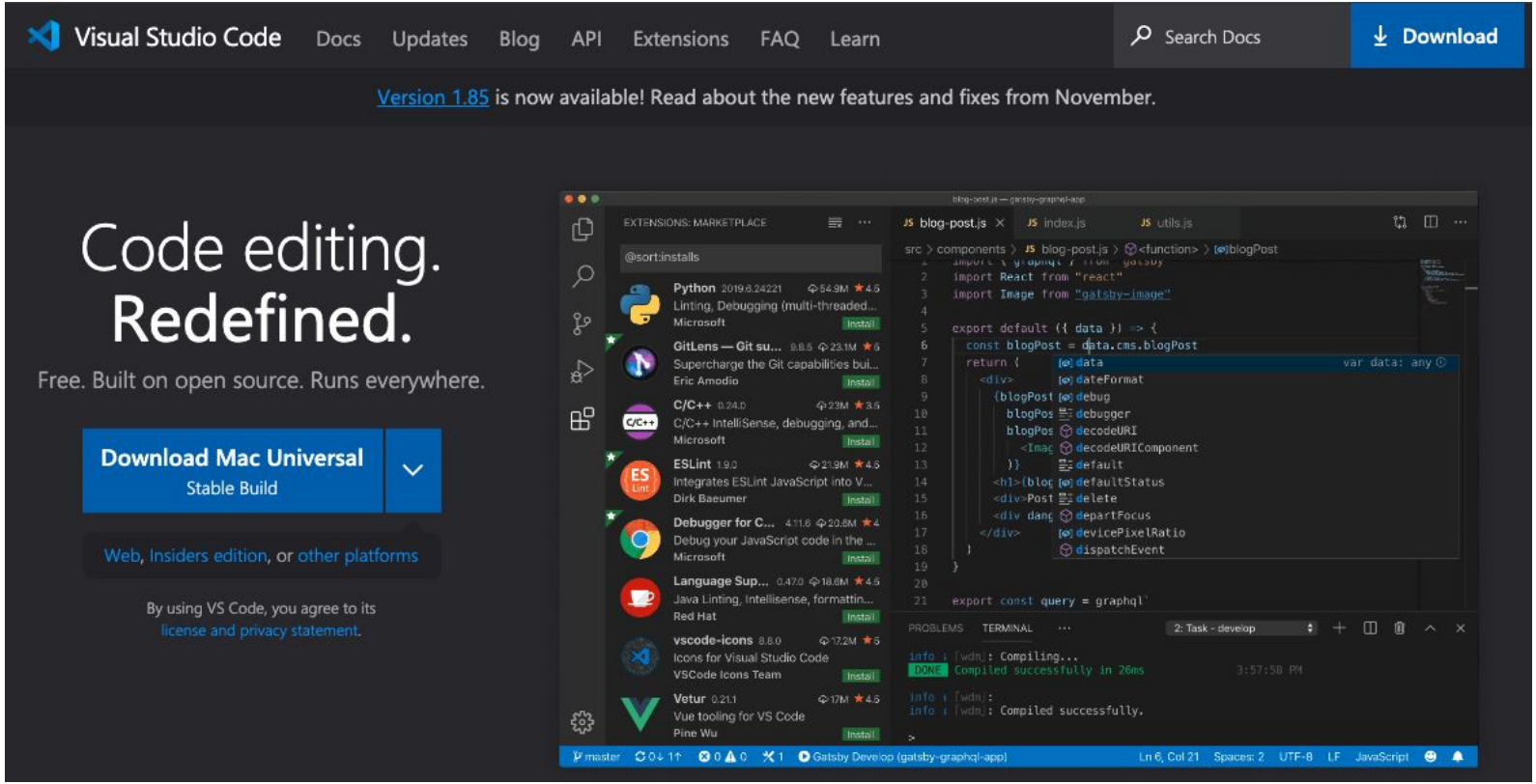
- What is Google Colab?
  - A free cloud service hosted by Google to encourage machine learning education and research.
  - https://colab.google/
- Key Features of Google Colab
  - Zero Configuration: Run Python code in the browser with no setup.
  - Free Access to GPUs: Offers free access to NVIDIA GPUs to speed up computing tasks.
  - Integration with Google Drive: Easy to save and load data from Google Drive.
  - Collaborative Environment: Supports real-time collaboration
  - Jupyter Notebook Compatibility: Based on Jupyter Notebooks, providing a familiar interface.



Visual Studio Code

Visual Studio Code

- Introduction to Visual Studio Code (VSCode)
  - Visual Studio Code (VSCode): A powerful, open-source code editor developed by Microsoft.
  - Offers comprehensive coding functionality for various programming languages, including Python.
- Key Features of VSCode
  - Extensibility: Vast array of extensions available for different programming languages and tools.
  - Integrated Terminal: Built-in terminal for running shell commands and scripts.
  - Version Control Integration: Seamless Git integration for version control.
  - Debugging Tools: Advanced debugging features built into the editor.
  - Customization: Highly customizable interface, key bindings, and settings.
- Why VSCode for Python and Machine Learning
  - Python Support: Excellent support for Python through extensions.
  - Community and Ecosystem: Large community, regularly updated with new features and improvements.
  - Code and Data Visualization: Offers features like code refactoring and easy visualization of data and computational graphs.



https://code.visualstudio.com/

Assignment 1 Overview

Requirements of Assignment 1

- Where to download the assignment
  - Course Moodle Assignment 1.
  - 4 files with questions and some template code.
- What to submit
  - Completed python notebook with executed outputs.
- Name of the submission
  - Use your uid, xxxx.zip. For example: 3009666.zip.
- Where to submit
  - Course Moodle Assignment 1.
- When to submit
  - Due day is Sunday 23:59 two weeks later
- Plagiarism policy
  - Do not copy. Both the student who copies and the student who offers his/her work for copying will be penalized.

Assignment 1 Overview

- Q1: Written questions (50 points)
  - Write your answer in the given cell using using Markdown grammar and Latex math equations.
- Q2: Perceptron boolean operators (15 points)
  - The example code for “NOT Operator” is provided, do not change it.
  - Implement other operators referring to the example code
- Q3: Digits classification (15 points)
  - We provide the template code as well.
  - Implement the required classification methods following the sample code.
- Q4: Decision boundary (20 points)
  - The supporting codes would guide you:
    - Train a perceptron on given dataset
    - Visualize the dataset and decision regions
    - Draw a random decision boundary
  - What you need to do
    - Write code to compute the actual decision boundary for the trained perceptron.
    - Implement a Adaline model and draw the decision boundary

Live demo and explanation of the Jupyter files.

Guidelines for Assignment Submission with Jupyter

- Step 1: Download the assignment files (e.g., .ipynb files) from Moodle
- Step 2:
  - Option 1: Open the ipynb file with **jupyter notebook**
  - Option 2: Use **Google Colab** and upload the ipynb file
- Step 4: Answer the questions in this file
- Step 5: Execute all the code blocks to print the results
- Step 6: Save and download this **executed** .ipynb file
- Step 7: Complete all your questions, put your answer files in one .zip,
- Step 8: Rename it using your uid, like 3009666000.zip
- Step 9: Submit the .zip on Moodle

Remember to save the executed notebook file!