CS685/785 Foundation of Data Science

Tutorial on Jupyter Notebooks

Xi Li

Fall 2024

Outline

- Anaconda/miniconda Installation
- Conda Environment
- Coding in Jupyter Notebook
- Assignment/Project Codes Submission

Anaconda/miniconda Installation

- Install Anaconda/miniconda

  https://docs.anaconda.com/anaconda/install/

System requirements

On Windows, macOS, and Linux, it is best to install Anaconda for the local user, which does not require administrator permissions and is the most robust type of installation. However, with administrator permissions, you can install Anaconda system wide.

Installing on Windows

Installing on macOS

Installing on Linux

- For windows user, you can use Windows Subsystem for Linux (**WSL**)

  https://learn.microsoft.com/en-us/windows/wsl/install

Conda Environment Setup

- Check that conda is correctly installed:

```
(base) xi@CS-648BN34:~$ conda --version
conda 24.7.1
```

- Sett up an environment for the course:

```
(base) xi@CS-648BN34:~$ conda create -n cs685 python=3.8
```

- Activate and enter the environment

```
(base) xi@CS-648BN34:~$ conda activate cs685
(cs685) xi@CS-648BN34:~$
```

- Deactivate and exit the environment

```
(cs685) xi@CS-648BN34:~$ conda deactivate
(base) xi@CS-648BN34:~$
```

Coding in Jupyter Notebook

- Install Jupyter Notebook

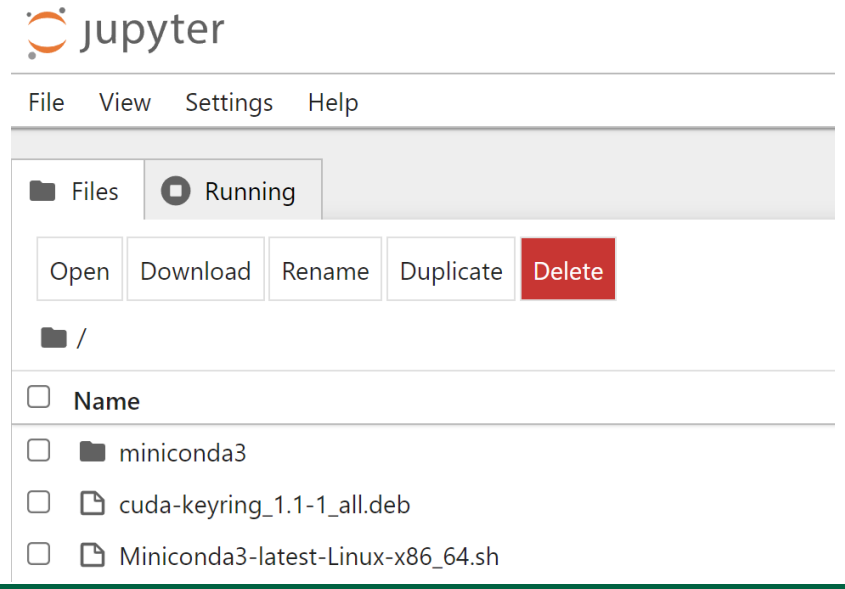
```
(cs685) xi@CS-648BN34:~$ conda install anaconda::jupyter
```

- Launch Jupyter Notebook

```
(cs685) xi@CS-648BN34:~$ jupyter notebook
```

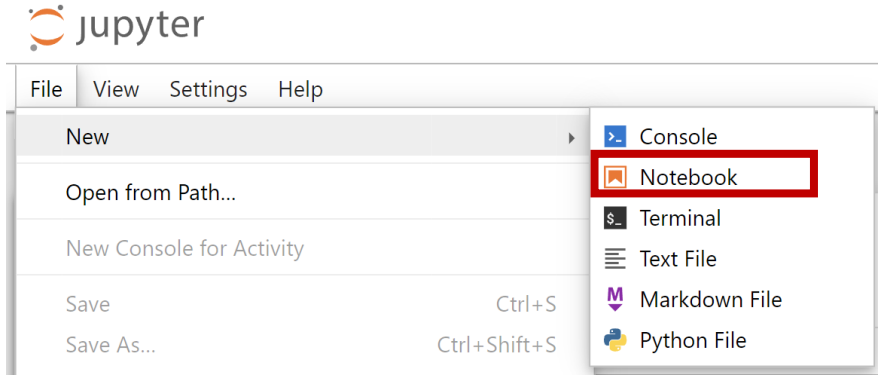
- Then go to http://localhost:8888/

  in your web browser

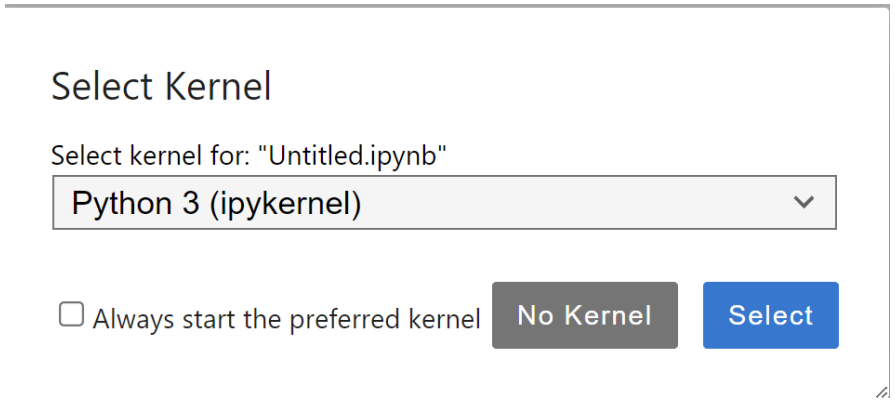


Coding in Jupyter Notebook

- Create a notebook

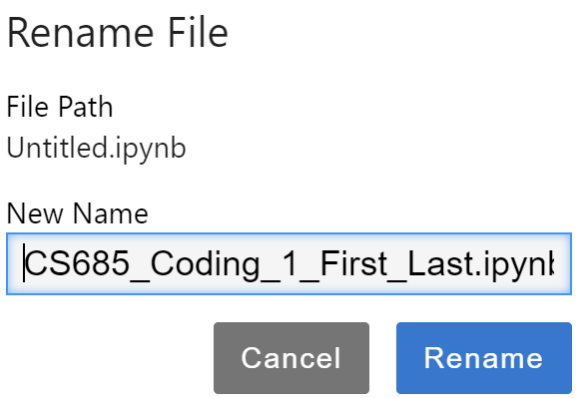


- Select kernel



Coding in Jupyter Notebook

- Rename the notebook: CS685_Coding_X_FirstName_LastName.ipynb



Coding in Jupyter Notebook

- It contains multiple cells. We can write and run python code in each cell.

[156]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

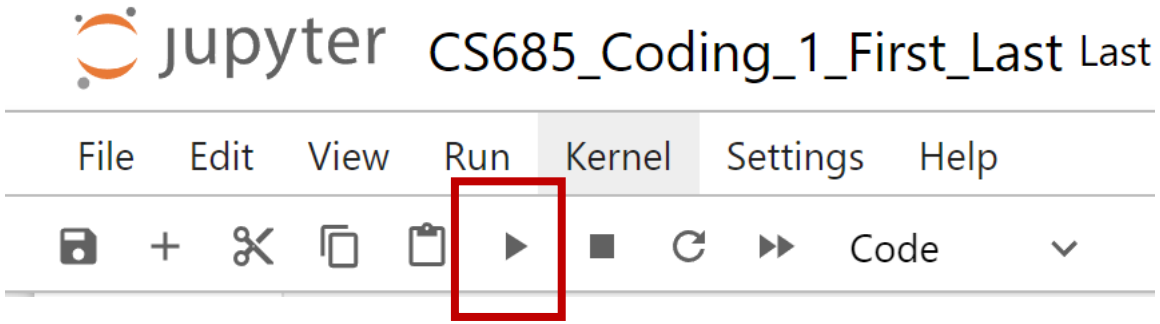
[157]:
```
# Load the dataset
iris = load_iris()
X = iris.data  # features
y = iris.target  # target values

# Only use two features
X = X[:, [0, -1]]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Coding in Jupyter Notebook

- Run the code in a cell by hitting “shift” + “enter” or clicking



Coding in Jupyter Notebook

- Run the code in a cell

[156]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

[157]:
```
# Load the dataset
iris = load_iris()
X = iris.data  # features
y = iris.target  # target values

# Only use two features
X = X[:, [0, -1]]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Coding in Jupyter Notebook

- Execution order of cells matters

[156]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

[157]:
```
# Load the dataset
iris = load_iris()
X = iris.data  # features
y = iris.target  # target values

# Only use two features
X = X[:, [0, -1]]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Coding in Jupyter Notebook

- Execution order of cells matters

[2]:
```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

[1]:
```
# Load the dataset
iris = load_iris()
X = iris.data  # features
y = iris.target  # target values

# Only use two features
X = X[:, [0, -1]]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 2
      1 # Load the dataset
----> 2 iris = load_iris()
      3 X = iris.data  # features
      4 y = iris.target  # target values

NameError: name 'load_iris' is not defined
```

Coding in Jupyter Notebook

- Global variables are shared among cells

```
In [1]: x = 1 + 1
        print x

        2

In [2]: y = 2 * x
        print y

        4
```

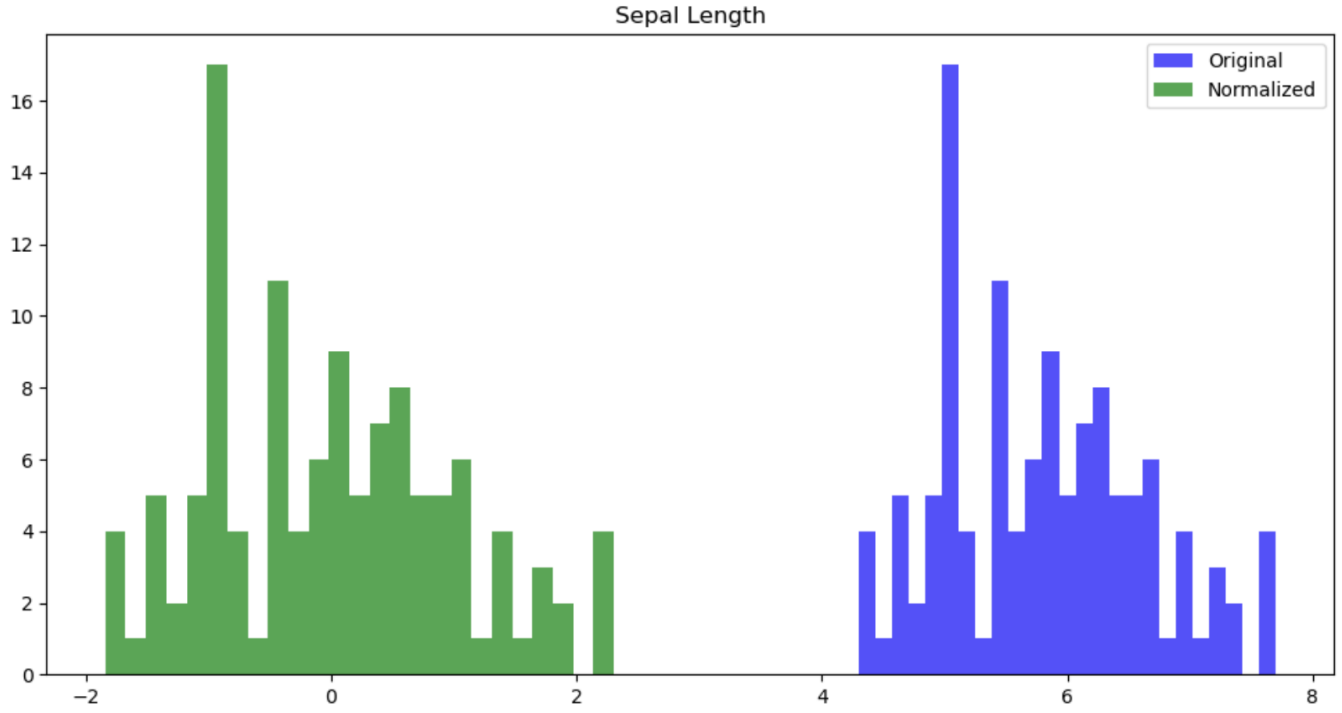
Coding in Jupyter Notebook

- We can get immediate result of a cell.

```python
import matplotlib.pyplot as plt

# Plotting the histograms and scatter plots
plt.figure(figsize=(12, 6))

# Histograms for original and normalized Sepal Length
plt.hist(X_train[:, 0], bins=25, alpha=0.7, label='Original', color='blue')
plt.hist(X_train_norm[:, 0], bins=25, alpha=0.7, label='Normalized', color='green')
plt.title('Sepal Length')
plt.legend()
plt.show()
```



Assignment/Project Codes Submission

- Save and export as html file