Introduction to Machine Learning and Toolkit Exercises

What is Machine Learning?

This	is
Machine learning allows computers to learn and infer from data.	robot.png

Learning Objectives

- Demonstrate supervised learning algorithms
- Explain key concepts like under- and over-fitting, regularization, and cross-validation
- Classify the type of problem to be solved, choose the right algorithm, tune parameters, and validate a model
- Apply Intel® Extension for Scikit-learn* to leverage underlying compute capabilities of hardware

Overview of Course:

Topics include:

- Introduction and exploratory analysis (Week 1)
- Supervised machine learning (Weeks 2 – 10)
- Unsupervised machine learning (Weeks 11 – 12)

Prerequisites:

- Python* programming
- Calculus
- Linear algebra
- Statistics

Lab Preparation:

- pip install -r ../requirements.txt

Our Toolset: Intel® oneAPI AI Analytics Toolkit (AI Kit)

- Intel® Extension for Scikit-learn*

Intel® oneAPI Toolkits Installation

The following documents provide detailed instructions on how to get and install Intel® oneAPI packages using different installer modes and package managers:

- Intel® oneAPI Toolkits Installation Guide for Linux* OS
- Intel® oneAPI Toolkits Installation Guide for Windows*
- Intel® oneAPI Toolkits Installation Guide for macOS*

Introduction

We will be using the iris data set for this tutorial. This is a well-known data set containing iris species and sepal and petal measurements. The data we will use are in a file called `Iris_Data.csv` found in the data directory.

```
from __future__ import print_function
import os
data_path = ['data']
print (data_path)
```

```
    ['data']
```

scikit-learn*

Frameworks provide structure that Data Scientists use to build code. Frameworks are more than just libraries, because in addition to callable code, frameworks influence how code is written.

A main virtue of using an optimized framework is that code runs faster. Code that runs faster is just generally more convenient but when we begin looking at applied data science and AI models, we can see more material benefits. Here you will see how optimization, particularly hyperparameter optimization can benefit more than just speed.

These exercises will demonstrate how to apply **the Intel® Extension for Scikit-learn*,** a seamless way to speed up your Scikit-learn application. The acceleration is achieved through the use of the Intel® oneAPI Data Analytics Library (oneDAL). Patching is the term used to extend scikit-learn with Intel optimizations and makes it a well-suited machine learning framework for dealing with real-life problems.

To get optimized versions of many Scikit-learn algorithms using a patch() approach consisting of adding these lines of code Prior to importing sklearn:

- **from sklearnex import patch_sklearn**
- **patch_sklearn()**

Question 1

Load the data from the file (data/Iris_Data.csv) using the techniques learned today. Examine it.

Determine the following:

- The number of data points (rows). (*Hint:* check out the dataframe `.shape` attribute.)
- The column names. (*Hint:* check out the dataframe `.columns` attribute.)
- The data types for each column. (*Hint:* check out the dataframe `.dtypes` attribute.)

```
pip install numpy
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: numpy in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/lib/py

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: /srv/jupyter/python-venv/bin/python3 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
pip install scipy-stack
```

```
Requirement already satisfied: rfc3339-validator in /srv/jupyter/python-venv/lib/python3.
Requirement already satisfied: rfc3986-validator>=0.1.1 in /srv/jupyter/python-venv/lib/p
Requirement already satisfied: ptyprocess>=0.5 in /srv/jupyter/python-venv/lib/python3.11
Requirement already satisfied: charset-normalizer<4,>=2 in /srv/jupyter/python-venv/lib/p
Requirement already satisfied: urllib3<3,>=1.21.1 in /srv/jupyter/python-venv/lib/python3
Requirement already satisfied: certifi>=2017.4.17 in /srv/jupyter/python-venv/lib/python3
Requirement already satisfied: argon2-cffi-bindings in /srv/jupyter/python-venv/lib/pytho
Requirement already satisfied: executing>=1.2.0 in /srv/jupyter/python-venv/lib/python3.1
Requirement already satisfied: asttokens>=2.1.0 in /srv/jupyter/python-venv/lib/python3.1
Requirement already satisfied: pure-eval in /srv/jupyter/python-venv/lib/python3.11/site-
Requirement already satisfied: fqdn in /srv/jupyter/python-venv/lib/python3.11/site-packa
Requirement already satisfied: isoduration in /srv/jupyter/python-venv/lib/python3.11/sit
Requirement already satisfied: jsonpointer>1.13 in /srv/jupyter/python-venv/lib/python3.1
Requirement already satisfied: uri-template in /srv/jupyter/python-venv/lib/python3.11/si
Requirement already satisfied: webcolors>=1.11 in /srv/jupyter/python-venv/lib/python3.11
Requirement already satisfied: cffi>=1.0.1 in /srv/jupyter/python-venv/lib/python3.11/sit
Requirement already satisfied: pycparser in /srv/jupyter/python-venv/lib/python3.11/site-
Requirement already satisfied: arrow>=0.15.0 in /srv/jupyter/python-venv/lib/python3.11/s

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: /srv/jupyter/python-venv/bin/python3 -m pip install --upgrade pi
Note: you may need to restart the kernel to use updated packages.
```

```
pip install pandas
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/lib/p
Requirement already satisfied: numpy<2,>=1.23.2 in /home/ua431795ad72bf966ec9b6a21a4251ca/.l
Requirement already satisfied: python-dateutil>=2.8.2 in /srv/jupyter/python-venv/lib/python
Requirement already satisfied: pytz>=2020.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: tzdata>=2022.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.loc
Requirement already satisfied: six>=1.5 in /srv/jupyter/python-venv/lib/python3.11/site-pack

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: /srv/jupyter/python-venv/bin/python3 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
import os
```

```
from __future__ import print_function
import os
data_path = ['data']
print (data_path)
```

```
['data']
```

```
import numpy as np
import pandas as pd

filepath = os.sep.join(data_path + ['Iris_Data.csv'])
print(filepath)
data = pd.read_csv(filepath)
data.head()
```

```
data/Iris_Data.csv
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
# Number of rows
print(data.shape[0])

# Column names
print(data.columns.tolist())

# Data types
print(data.dtypes)
```

```
150
['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']
sepal_length    float64
sepal_width     float64
petal_length    float64
petal_width     float64
species          object
dtype: object
```

Question 2

Examine the species names and note that they all begin with 'Iris-'. Remove this portion of the name so the species name is shorter.

Hint: there are multiple ways to do this, but you could use either the string processing methods or the apply method.

```
# The str method maps the following function to each entry as a string
data['species'] = data.species.str.replace('Iris-', '')
# alternatively
# data['species'] = data.species.apply(lambda r: r.replace('Iris-', ''))

data.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Question 3

Determine the following:

- The number of each species present. (*Hint:* check out the series `.value_counts` method.)
- The mean, median, and quantiles and ranges (max-min) for each petal and sepal measurement.

Hint: for the last question, the `.describe` method does have median, but it's not called median. It's the *50%* quantile. `.describe` does not have range though, and in order to get the range, you will need to create a new entry in the `.describe` table, which is `max - min`.

```
print(data['species'].value_counts())
```

```
species
setosa        50
versicolor    50
virginica     50
Name: count, dtype: int64
```

```
def describe_iris(df):
    df1 = df.describe(exclude=[object])
    df1.loc["range"] = df1.loc['max'] - df1.loc['min']
    return df1
```

```
print(describe_iris(data))
```

```
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.054000      3.758667     1.198667
std        0.828066     0.433594      1.764420     0.763161
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
```

```
50%      5.800000    3.000000     4.350000    1.300000
75%      6.400000    3.300000     5.100000    1.800000
max      7.900000    4.400000     6.900000    2.500000
range    3.600000    2.400000     5.900000    2.400000
```

```
data.describe(include = 'all')
```

	sepal_length	sepal_width	petal_length	petal_width	species
count	150.000000	150.000000	150.000000	150.000000	150
unique	NaN	NaN	NaN	NaN	3
top	NaN	NaN	NaN	NaN	setosa
freq	NaN	NaN	NaN	NaN	50
mean	5.843333	3.054000	3.758667	1.198667	NaN
std	0.828066	0.433594	1.764420	0.763161	NaN
min	4.300000	2.000000	1.000000	0.100000	NaN
25%	5.100000	2.800000	1.600000	0.300000	NaN
50%	5.800000	3.000000	4.350000	1.300000	NaN
75%	6.400000	3.300000	5.100000	1.800000	NaN
max	7.900000	4.400000	6.900000	2.500000	NaN

Start coding or generate with AI.

Question 4

Calculate the following **for each species** in a separate dataframe:

- The mean of each measurement (sepal_length, sepal_width, petal_length, and petal_width).
- The median of each of these measurements.

Hint: you may want to use Pandas `groupby` method to group by species before calculating the statistic.

If you finish both of these, try calculating both statistics (mean and median) in a single table (i.e. with a single groupby call). See the section of the Pandas documentation on applying multiple functions at once for a hint.

```
# The mean calculation
data.groupby('species').mean()
```

	sepal_length	sepal_width	petal_length	petal_width
species				
setosa	5.006	3.418	1.464	0.244
versicolor	5.936	2.770	4.260	1.326
virginica	6.588	2.974	5.552	2.026

```
# The median calculation
data.groupby('species').median()
```

	sepal_length	sepal_width	petal_length	petal_width
species				
setosa	5.0	3.4	1.50	0.2
versicolor	5.9	2.8	4.35	1.3
virginica	6.5	3.0	5.55	2.0

```
# applying multiple functions at once - 2 methods

data.groupby('species').agg(['mean', 'median'])  # passing a list of recognized strings
data.groupby('species').agg([np.mean, np.median])  # passing a list of explicit aggregation func
```

```
/tmp/ipykernel_592685/1749018383.py:4: FutureWarning: The provided callable <function mean a
  data.groupby('species').agg([np.mean, np.median])  # passing a list of explicit aggregatic
/tmp/ipykernel_592685/1749018383.py:4: FutureWarning: The provided callable <function median
  data.groupby('species').agg([np.mean, np.median])  # passing a list of explicit aggregatic
/tmp/ipykernel_592685/1749018383.py:4: FutureWarning: The provided callable <function mean a
  data.groupby('species').agg([np.mean, np.median])  # passing a list of explicit aggregatic
```

	sepal_length		sepal_width		petal_length		petal_width	
	mean	median	mean	median	mean	median	mean	median
species								
setosa	5.006	5.0	3.418	3.4	1.464	1.50	0.244	0.2
versicolor	5.936	5.9	2.770	2.8	4.260	4.35	1.326	1.3
virginica	6.588	6.5	2.974	3.0	5.552	5.55	2.026	2.0

```
# If certain fields need to be aggregated differently, we can do:
from pprint import pprint

agg_dict = {field: ['mean', 'median'] for field in data.columns if field != 'species'}
agg_dict['petal_length'] = 'max'
pprint(agg_dict)
data.groupby('species').agg(agg_dict)
```

```
{'petal_length': 'max',
 'petal_width': ['mean', 'median'],
 'sepal_length': ['mean', 'median'],
 'sepal_width': ['mean', 'median']}
```

	sepal_length		sepal_width		petal_length	petal_width	
	mean	median	mean	median	max	mean	median
species							
setosa	5.006	5.0	3.418	3.4	1.9	0.244	0.2
versicolor	5.936	5.9	2.770	2.8	5.1	1.326	1.3
virginica	6.588	6.5	2.974	3.0	6.9	2.026	2.0

Question 5

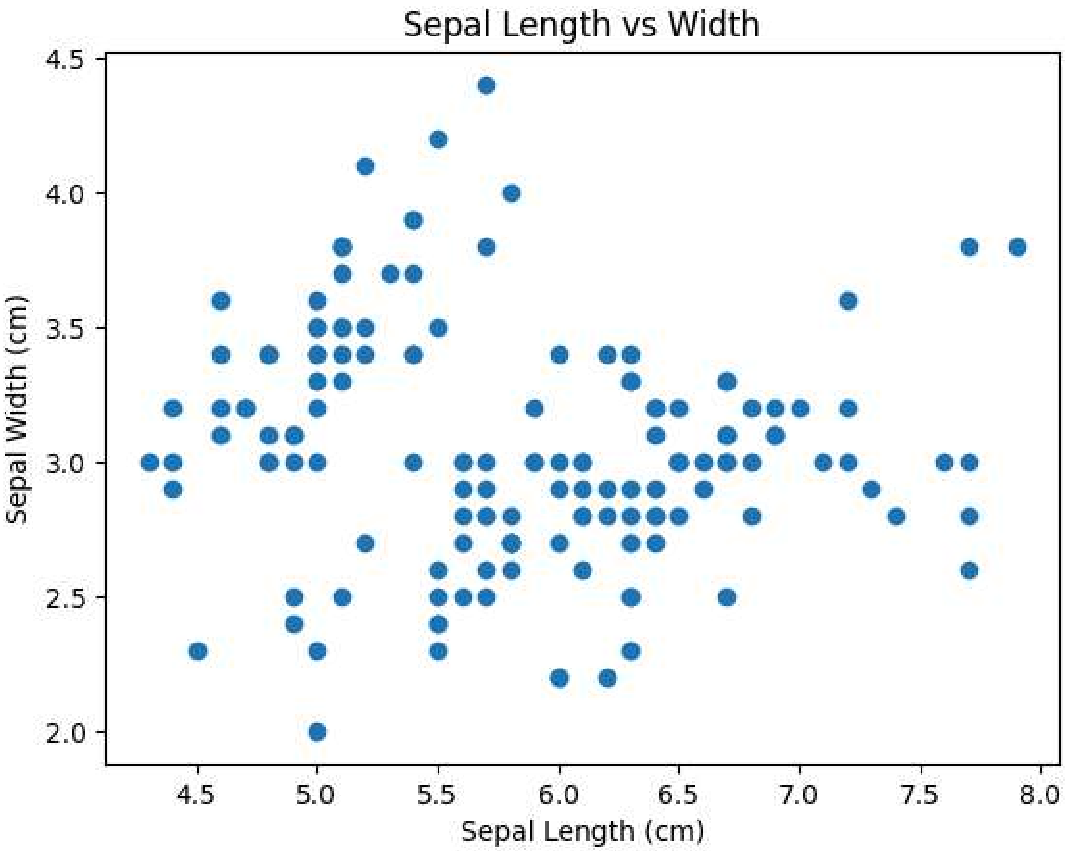
Make a scatter plot of `sepal_length` vs `sepal_width` using Matplotlib. Label the axes and give the plot a title.

```
import matplotlib.pyplot as plt
%matplotlib inline
```

```
# A simple scatter plot with Matplotlib
ax = plt.axes()

ax.scatter(data.sepal_length, data.sepal_width)

# Label the axes
ax.set(xlabel='Sepal Length (cm)',
       ylabel='Sepal Width (cm)',
       title='Sepal Length vs Width');
```

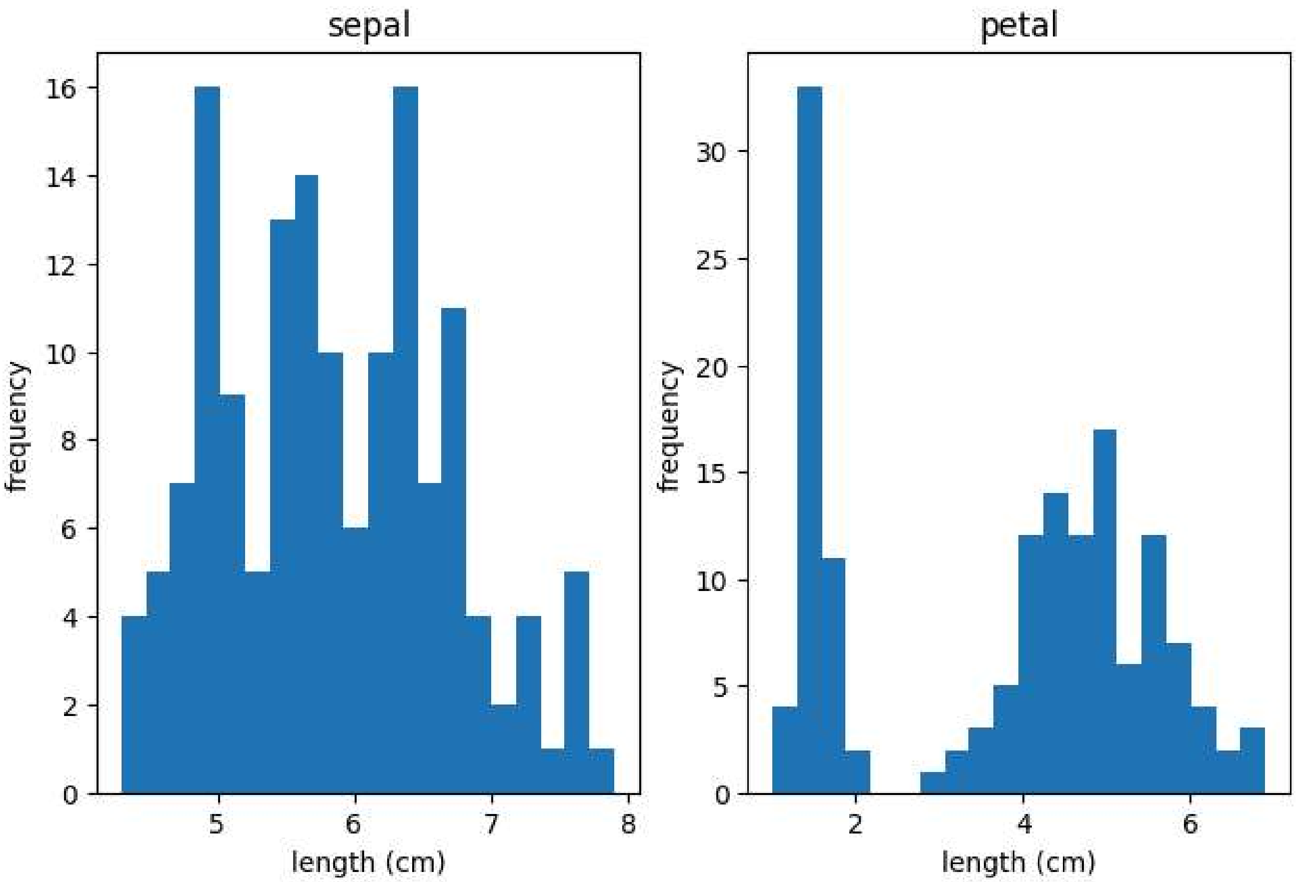


Question 6

Make a histogram of any one of the four features. Label axes and title it as appropriate.

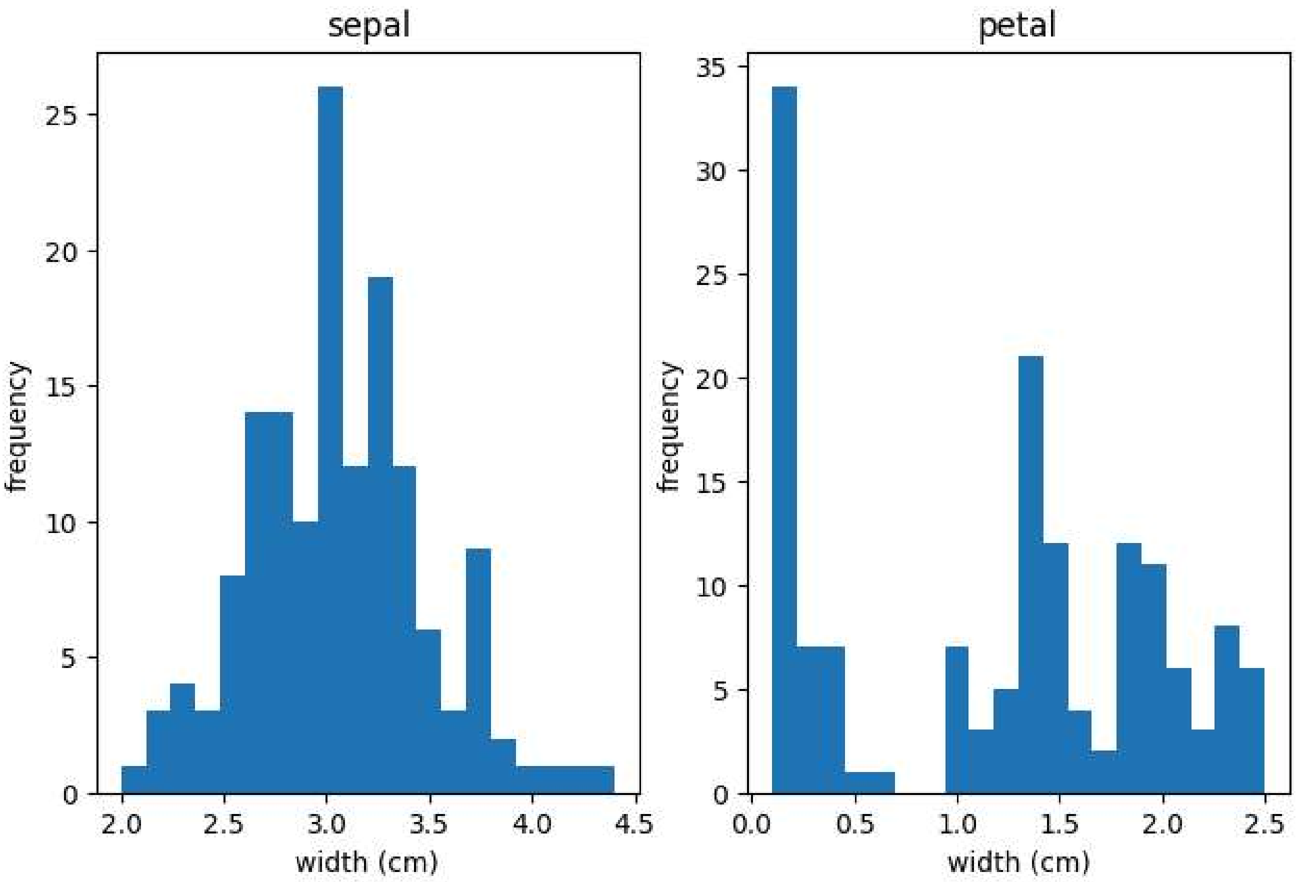
```
fig, axs = plt.subplots(1, 2, figsize=(8, 5))
axs[0].hist(data['sepal_length'], bins=20)
axs[0].set_title('sepal')
axs[1].hist(data['petal_length'], bins=20)
axs[1].set_title('petal')

for ax in axs.flat:
    ax.set(xlabel='length (cm)', ylabel='frequency')
```



```
fig, axs = plt.subplots(1, 2, figsize=(8, 5))
axs[0].hist(data['sepal_width'], bins=20)
axs[0].set_title('sepal')
axs[1].hist(data['petal_width'], bins=20)
axs[1].set_title('petal')

for ax in axs.flat:
    ax.set(xlabel='width (cm)', ylabel='frequency')
```



Question 7

Now create a single plot with histograms for each feature (`petal_width`, `petal_length`, `sepal_width`, `sepal_length`) overlayed. If you have time, next try to create four individual histogram plots in a single figure, where each plot contains one feature.

For some hints on how to do this with Pandas plotting methods, check out the visualization guide for Pandas.

```
pip install seaborn
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: seaborn in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/lib/
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /home/ua431795ad72bf966ec9b6a21a4251c
Requirement already satisfied: pandas>=1.2 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /home/ua431795ad72bf966ec9b6a21a42
Requirement already satisfied: contourpy>=1.0.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.l
Requirement already satisfied: cycler>=0.10 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: fonttools>=4.22.0 in /home/ua431795ad72bf966ec9b6a21a4251ca/.
Requirement already satisfied: kiwisolver>=1.3.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.
Requirement already satisfied: packaging>=20.0 in /srv/jupyter/python-venv/lib/python3.11/si
Requirement already satisfied: pillow>=8 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/li
Requirement already satisfied: pyparsing>=2.3.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.l
Requirement already satisfied: python-dateutil>=2.7 in /srv/jupyter/python-venv/lib/python3.
Requirement already satisfied: pytz>=2020.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: tzdata>=2022.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.loc
Requirement already satisfied: six>=1.5 in /srv/jupyter/python-venv/lib/python3.11/site-pack

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: /srv/jupyter/python-venv/bin/python3 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
pip install seaborn[stats]
```

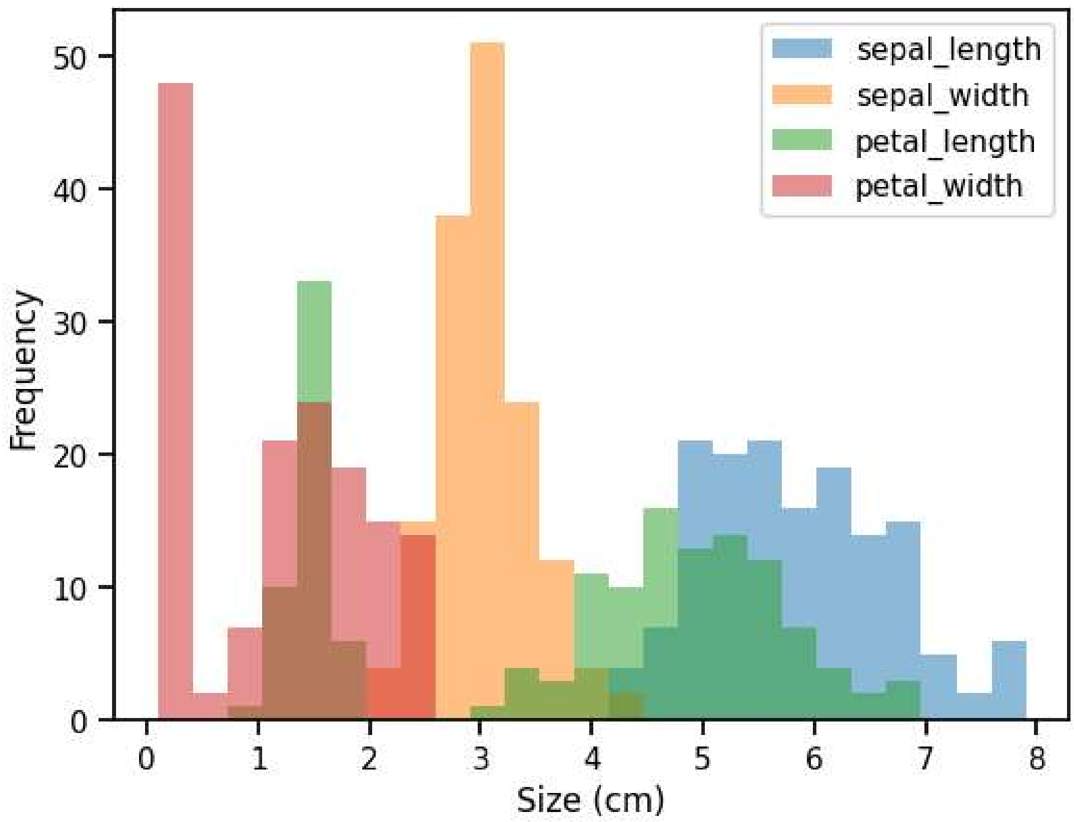
```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: seaborn[stats] in /home/ua431795ad72bf966ec9b6a21a4251ca/.loc
Requirement already satisfied: numpy!=1.24.0,>=1.20 in /home/ua431795ad72bf966ec9b6a21a4251c
Requirement already satisfied: pandas>=1.2 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/
Requirement already satisfied: matplotlib!=3.6.1,>=3.4 in /home/ua431795ad72bf966ec9b6a21a42
Requirement already satisfied: scipy>=1.7 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/l
Requirement already satisfied: statsmodels>=0.12 in /home/ua431795ad72bf966ec9b6a21a4251ca/.
Requirement already satisfied: contourpy>=1.0.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.l
Requirement already satisfied: cycler>=0.10 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: fonttools>=4.22.0 in /home/ua431795ad72bf966ec9b6a21a4251ca/.
Requirement already satisfied: kiwisolver>=1.3.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.
Requirement already satisfied: packaging>=20.0 in /srv/jupyter/python-venv/lib/python3.11/si
Requirement already satisfied: pillow>=8 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local/li
Requirement already satisfied: pyparsing>=2.3.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.l
Requirement already satisfied: python-dateutil>=2.7 in /srv/jupyter/python-venv/lib/python3.
Requirement already satisfied: pytz>=2020.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: tzdata>=2022.1 in /home/ua431795ad72bf966ec9b6a21a4251ca/.loc
Requirement already satisfied: patsy>=0.5.4 in /home/ua431795ad72bf966ec9b6a21a4251ca/.local
Requirement already satisfied: six in /srv/jupyter/python-venv/lib/python3.11/site-packages

[notice] A new release of pip is available: 23.2.1 -> 23.3.2
[notice] To update, run: /srv/jupyter/python-venv/bin/python3 -m pip install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

```
import seaborn as sns

sns.set_context('notebook')

# This uses the `.plot.hist` method
ax = data.plot.hist(bins=25, alpha=0.5)
ax.set_xlabel('Size (cm)');
```



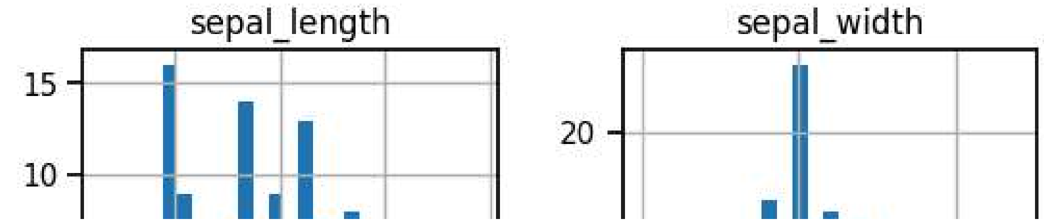
```
# To create four separate plots, use Pandas `.hist` method
axList = data.hist(bins=25)

# Add some x- and y- labels to first column and last row
for ax in axList.flatten():
    if ax.is_last_row():
        ax.set_xlabel('Size (cm)')

    if ax.is_first_col():
        ax.set_ylabel('Frequency')
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[25], line 6
      4 # Add some x- and y- labels to first column and last row
      5 for ax in axList.flatten():
----> 6     if ax.is_last_row():
      7         ax.set_xlabel('Size (cm)')
      9     if ax.is_first_col():

AttributeError: 'Axes' object has no attribute 'is_last_row'
```



Question 8

Using Pandas, make a boxplot of each petal and sepal measurement. Here is the documentation for [Pandas boxplot method](https://pandas.pydata.org/).

```
data.boxplot(column=['sepal_length', 'sepal_width', 'petal_length', 'petal_width'])
```

Question 9

Now make a single boxplot where the features are separated in the x-axis and species are colored with different hues.