

KNS INSTITUTE OF TECHNOLOGY

HEGDE NAGAR, TIRUMENAHALLI, KOGILU ROAD, BENGALURU – 64



DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

DATA SCIENCE AND ITS APPLICATIONS LABORATORY MANUAL (21AD62)

Academic Year 2023- 2024

Prepared by:

Mr. Dawa Chyophel Lepcha

Assistant Professor, AIML
KNSIT, Bengaluru

Reviewed by:

Dr. Aijaz Ali Khan

Head of the Department, AIML
KNSIT, Bengaluru

Dr. Surendra Babu Nallagurla

Assistant Professor, AIML
KNSIT, Bengaluru

Name of the Student: _____

University Serial Number: _____

Semester: _____ **Batch:** _____

LAB CODE: 21AD62

List of Experiments	Date	Remark																		
Cycle I																				
<p align="center">Module 1</p> <ol style="list-style-type: none"> Installation of Python/R language, Visual Studio code editors can be demonstrated along with Kaggle data set usage. Write programs in Python/R and Execute them in either Visual Studio Code or PyCharm Community Edition or any other suitable environment. A study was conducted to understand the effect of number of hours the students spent studying on their performance in the final exams. Write a code to plot line chart with number of hours spent studying on x-axis and score in final exam on y-axis. Use a red '*' as the point character, label the axes and give the plot a title. <table border="1"> <tr> <td>Number of hrs spent studying (x)</td><td>10</td><td>9</td><td>2</td><td>15</td><td>10</td><td>16</td><td>11</td><td>16</td></tr> <tr> <td>Score in the final exam (0 - 100) (y)</td><td>95</td><td>80</td><td>10</td><td>50</td><td>45</td><td>98</td><td>38</td><td>93</td></tr> </table> <ol style="list-style-type: none"> For the given dataset mtcars.csv (www.kaggle.com/ruiromanini/mtcars), plot a histogram to check the frequency distribution of the variable 'mpg' (Miles per gallon) <p align="center">Module 2</p> <ol style="list-style-type: none"> Consider the books dataset BL-Flickr-Images-Book.csv from Kaggle (https://www.kaggle.com/adeyoyintemidayo/publication-of-books) which contains information about books. Write a program to demonstrate the following. <ul style="list-style-type: none"> • Import the data into a DataFrame • Find and drop the columns which are irrelevant for the book information. • Change the Index of the DataFrame • Tidy up fields in the data such as date of publication with the help of simple regular expression. • Combine str methods with NumPy to clean columns <p align="center">Module 3</p> <ol style="list-style-type: none"> Train a regularized logistic regression classifier on the iris dataset (https://archive.ics.uci.edu/ml/machine-learning-databases/iris/ or the inbuilt iris dataset) using sklearn. Train the model with the following hyperparameter C = 1e4 and report the best classification accuracy. Train an SVM classifier on the iris dataset using sklearn. Try different kernels and the associated hyperparameters. Train model with the following set of 	Number of hrs spent studying (x)	10	9	2	15	10	16	11	16	Score in the final exam (0 - 100) (y)	95	80	10	50	45	98	38	93		
Number of hrs spent studying (x)	10	9	2	15	10	16	11	16												
Score in the final exam (0 - 100) (y)	95	80	10	50	45	98	38	93												

hyperparameters RBFkernel, gamma=0.5, one-vs-rest classifier, no-feature-normalization. Also try C=0.01,1,10C=0.01,1,10. For the above set of hyperparameters, find the best classification accuracy along with total number of support vectors on the test data

Cycle II

Module 4

1. Consider the following dataset. Write a program to demonstrate the working of the decision tree based ID3 algorithm.

Price	Maintenance	Capacity	Airbag	Profitable
Low	Low	2	No	Yes
Low	Med	4	Yes	Yes
Low	Low	4	No	Yes
Low	Med	4	No	No
Low	High	4	No	No
Med	Med	4	No	No
Med	Med	4	Yes	Yes
Med	High	2	Yes	No
Med	High	5	No	Yes
High	Med	4	Yes	Yes
high	Med	2	Yes	Yes
High	High	2	Yes	No
high	High	5	yes	Yes

2. Consider the dataset spiral.txt (<https://bit.ly/2Lm75Ly>). The first two columns in the dataset corresponds to the co-ordinates of each data point. The third column corresponds to the actual cluster label. Compute the rand index for the following methods:

- K – means Clustering
- Single – link Hierarchical Clustering
- Complete link hierarchical clustering.
- Also visualize the dataset and which algorithm will be able to recover the true clusters.
-

Module 5

1. Mini Project – Simple web scrapping in social media

DATA SCIENCE AND ITS APPLICATIONS

SUB CODE: 21AD62

LABORATORY EXPERIMENTS

Module 1

1. Installation of Python/R language, Visual Studio code editors can be demonstrated along with Kaggle data set usage.
2. Write programs in Python/R and Execute them in either Visual Studio Code or PyCharm Community Edition or any other suitable environment.
3. A study was conducted to understand the effect of number of hours the students spent studying on their performance in the final exams. Write a code to plot line chart with number of hours spent studying on x-axis and score in final exam on y-axis. Use a red '*' as the point character, label the axes and give the plot a title.

Number of hrs spent studying (x)	10	9	2	15	10	16	11	16
Score in the final exam (0 - 100) (y)	95	80	10	50	45	98	38	93

4. For the given dataset mtcars.csv (www.kaggle.com/ruiromanini/mtcars), plot a histogram to check the frequency distribution of the variable 'mpg' (Miles per gallon)

Module 2

1. Consider the books dataset BL-Flickr-Images-Book.csv from Kaggle (<https://www.kaggle.com/adeyoyintemidayo/publication-of-books>) which contains information about books. Write a program to demonstrate the following.
 - Import the data into a DataFrame
 - Find and drop the columns which are irrelevant for the book information.
 - Change the Index of the DataFrame
 - Tidy up fields in the data such as date of publication with the help of simple regular expression.
 - Combine str methods with NumPy to clean columns

Module 3

1. Train a regularized logistic regression classifier on the iris dataset (<https://archive.ics.uci.edu/ml/machine-learning-databases/iris/> or the inbuilt iris dataset) using sklearn. Train the model with the following hyperparameter $C = 1e4$ and report the best classification accuracy.
2. Train an SVM classifier on the iris dataset using sklearn. Try different kernels and the associated hyperparameters. Train model with the following set of hyperparameters RBFkernel, $\gamma=0.5$, one-vs-rest classifier, no-feature-normalization. Also try $C=0.01, 1, 10$ and $C=0.01, 1, 10$. For the above set of hyperparameters, find the best classification accuracy along with total number of support vectors on the test data

Module 4

3. Consider the following dataset. Write a program to demonstrate the working of the decision tree based ID3 algorithm.

Price	Maintenance	Capacity	Airbag	Profitable
Low	Low	2	No	Yes
Low	Med	4	Yes	Yes
Low	Low	4	No	Yes
Low	Med	4	No	No
Low	High	4	No	No
Med	Med	4	No	No
Med	Med	4	Yes	Yes
Med	High	2	Yes	No
Med	High	5	No	Yes
High	Med	4	Yes	Yes
high	Med	2	Yes	Yes
High	High	2	Yes	No
high	High	5	yes	Yes

4. Consider the dataset spiral.txt (<https://bit.ly/2Lm75Ly>). The first two columns in the dataset corresponds to the co-ordinates of each data point. The third column corresponds to the actual cluster label. Compute the rand index for the following methods:
- K – means Clustering
 - Single – link Hierarchical Clustering
 - Complete link hierarchical clustering.
 - Also visualize the dataset and which algorithm will be able to recover the true clusters.

Module 5

1. Mini Project – Simple web scrapping in social media

Assessment Details (both CIE and SEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/ course if the student secures not less than 35% (18 Marks out of 50) in the semester-end examination (SEE), and a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together.

Continuous Internal Evaluation:

Three Unit Tests each of **20 Marks (duration 01 hour)**

1. First test at the end of 5th week of the semester
2. Second test at the end of the 10th week of the semester
3. Third test at the end of the 15th week of the semester

Two assignments each of **10 Marks**

4. First assignment at the end of 4th week of the semester
5. Second assignment at the end of 9th week of the semester

Practical Sessions need to be assessed by appropriate rubrics and viva-voce method. This will contribute to **20 marks**.

- Rubrics for each Experiment taken average for all Lab components – 15 Marks.
- Viva-Voce– 5 Marks (more emphasized on demonstration topics)

The sum of three tests, two assignments, and practical sessions will be out of 100 marks and will be **scaled down to 50 marks** (to have a less stressed CIE, the portion of the syllabus should not be common /repeated for any of the methods of the CIE. Each method of CIE should have a different syllabus portion of the course).

CIE methods /question paper has to be designed to attain the different levels of Bloom's taxonomy as per the outcome defined for the course.

Semester End Examination:

Theory SEE will be conducted by University as per the scheduled timetable, with common question papers for the subject (**duration 03 hours**)

1. The question paper will have ten questions. Each question is set for 20 marks.
2. There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), **should have a mix of topics** under that module.
3. The students have to answer 5 full questions, selecting one full question from each module
4. Marks scored shall be proportionally reduced to 50 marks

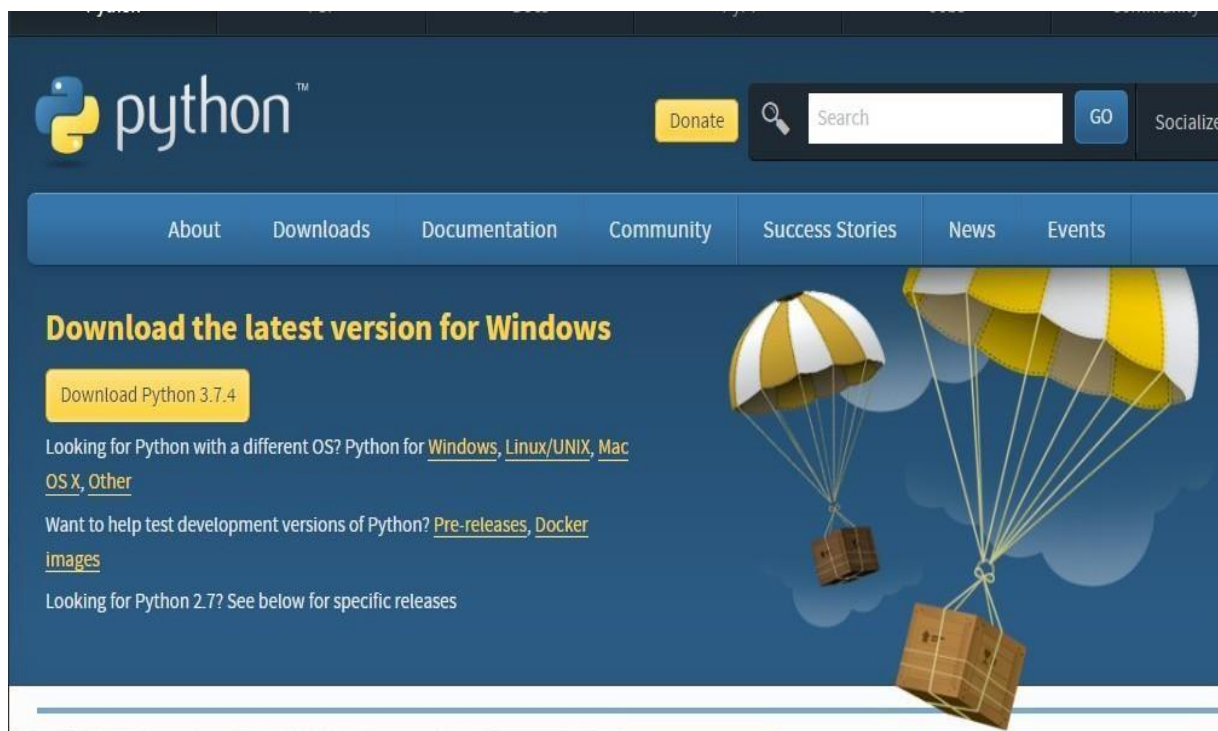
Introduction to Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting language to connect existing components together. Python is simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Python Installation

Download Python Interpreter

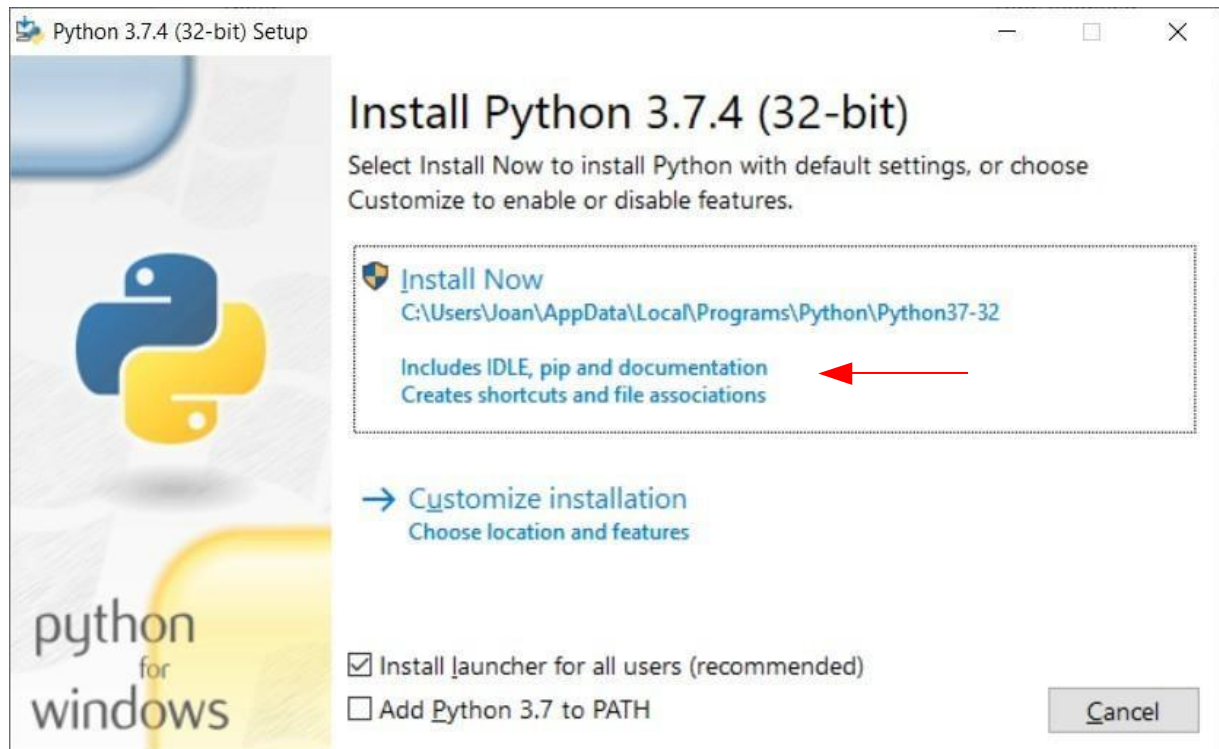
Go to the [Python downloads page](#) and select the version for your operating system (Windows, Mac, Linux)



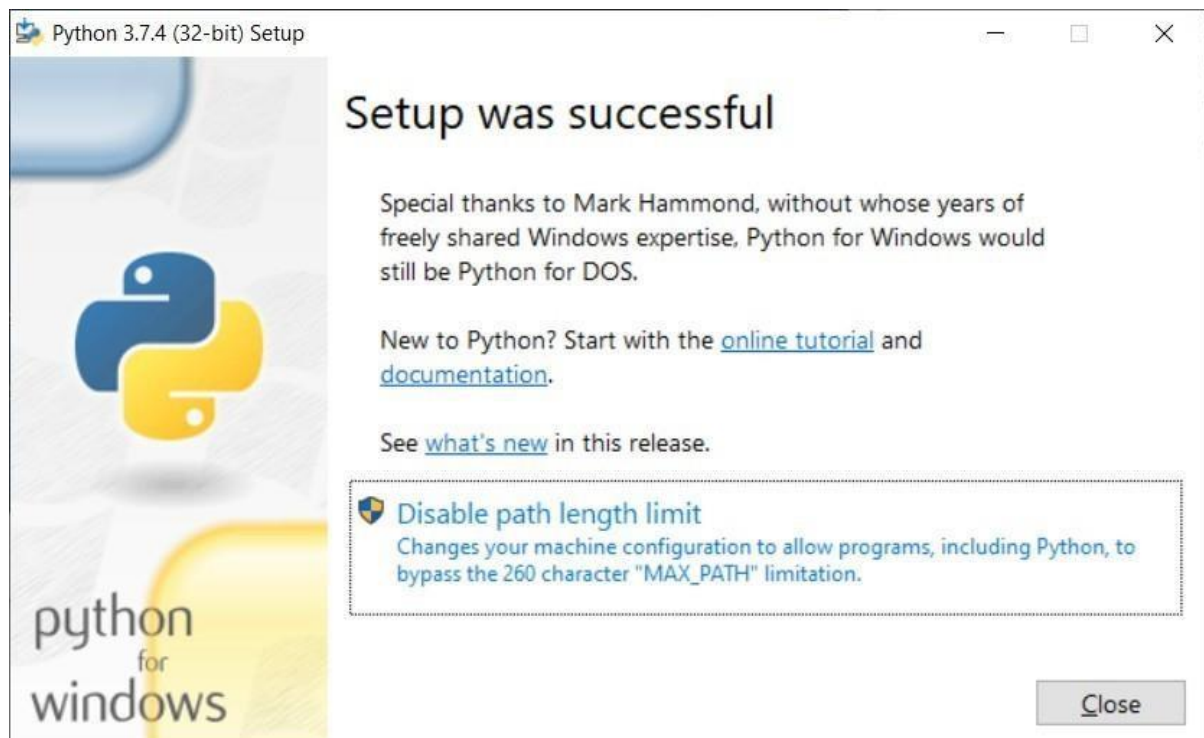
Install Python Interpreter

- Select the downloaded file to start the installation.
- **Important!** Remember the directory where Python is installed

- 'Install Now' picks a default directory
- 'Customize installation' lets you specify the directory location



If installation is successful, you should see this message

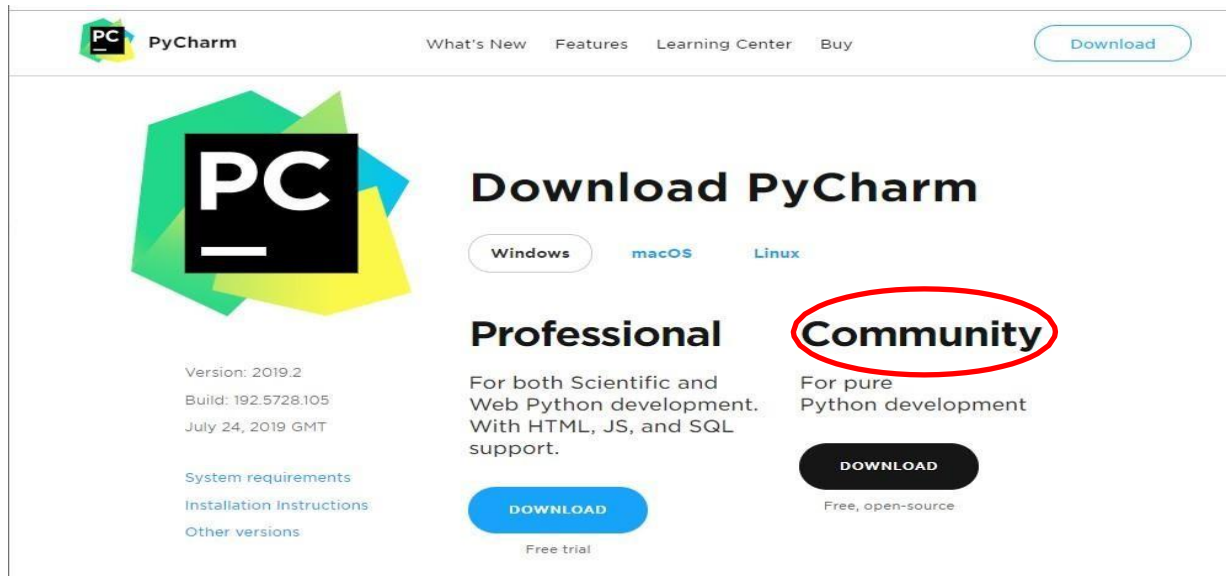


Introduction to PyCharm

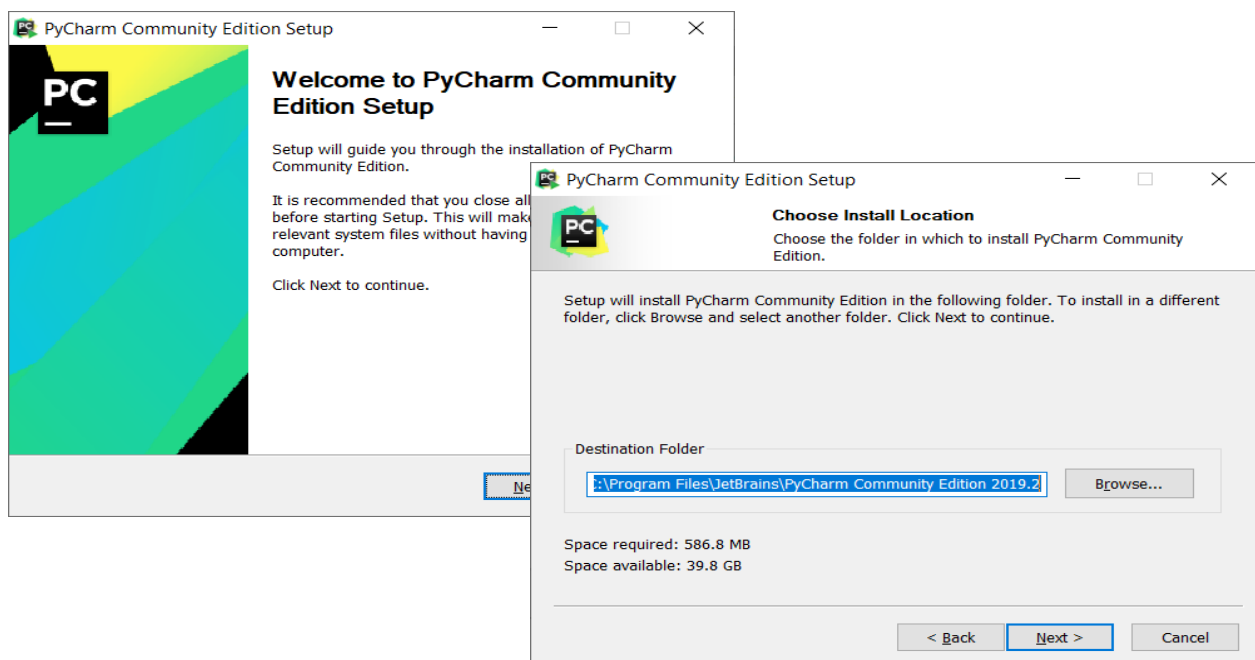
PyCharm is an integrated development environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

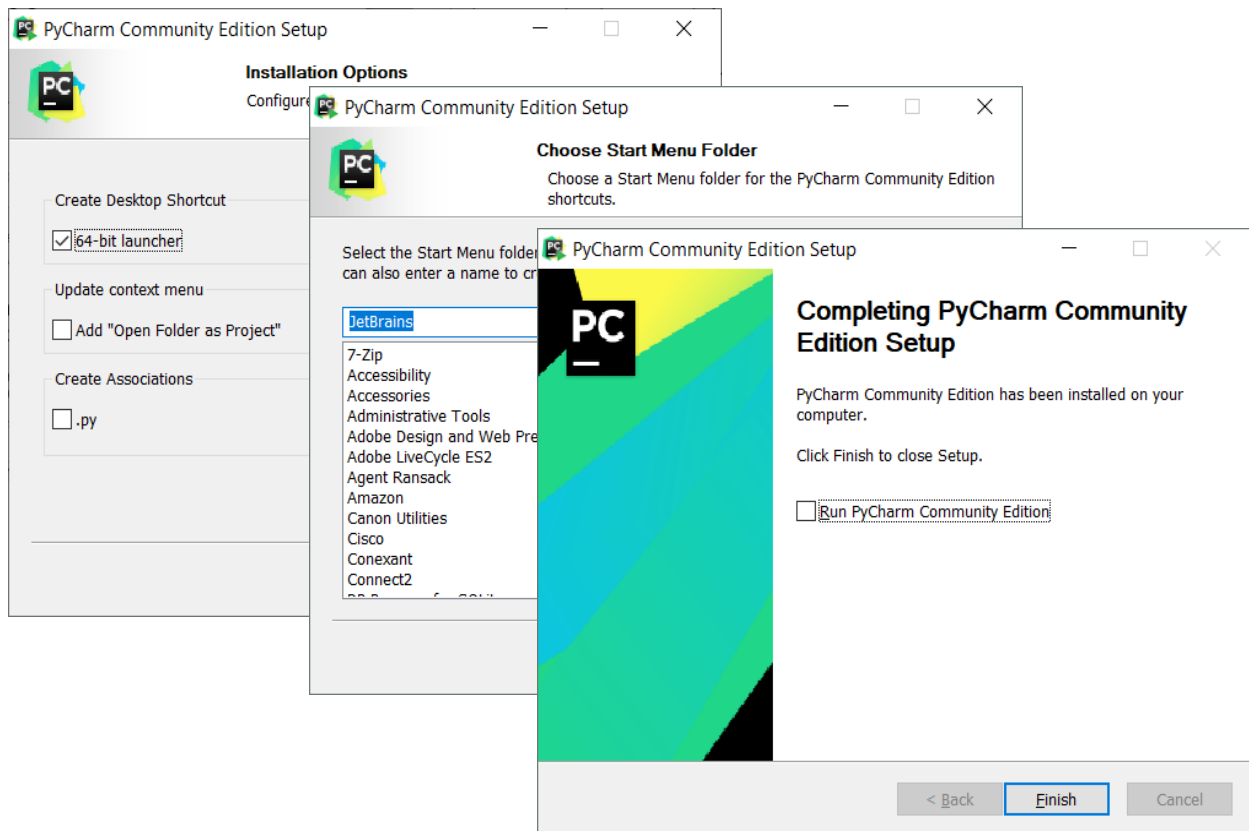
PyCharm IDE Installation

Go to [PyCharm download page](#) and select the FREECommunity Edition



Run the installation program

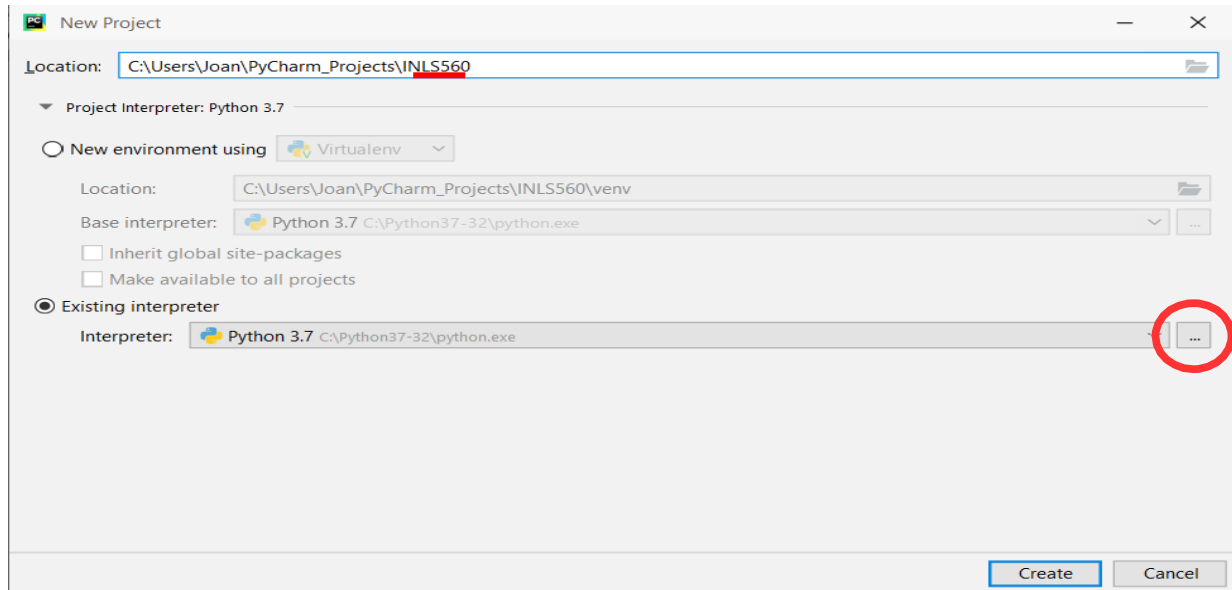




Using PyCharm: Create a project

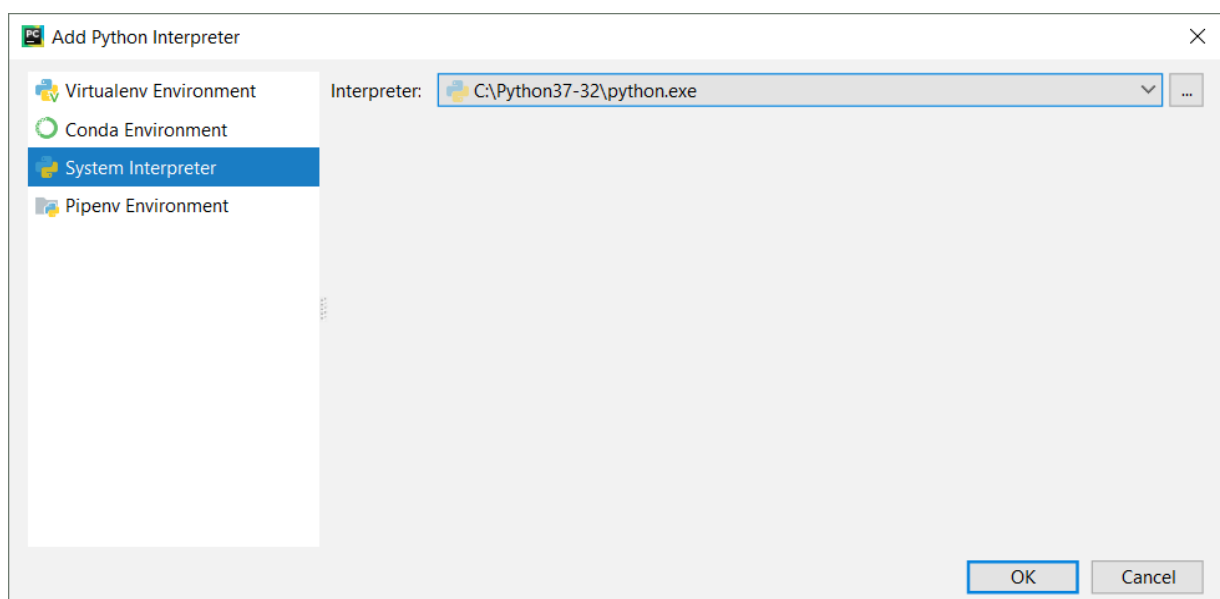


- First, using File Explorer (Windows), or Finder (Mac), create a directory for your projects, e.g., *PyCharm_Projects*
- Name your project, e.g., *INLS560*
- Specify the Python Interpreter you will use for your projects



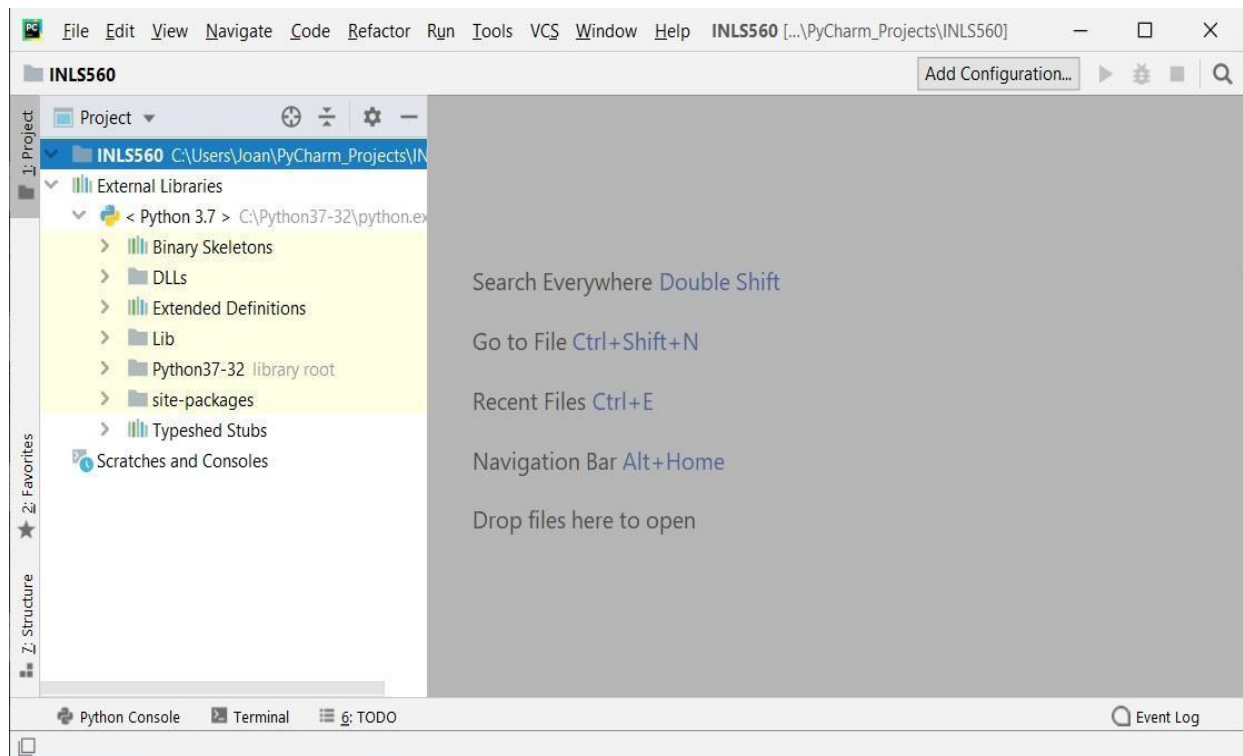
Specify the Python interpreter to use

- Select System Interpreter
- Ensure that the Interpreter field refers to the Python interpreter that you just installed. Click OK.

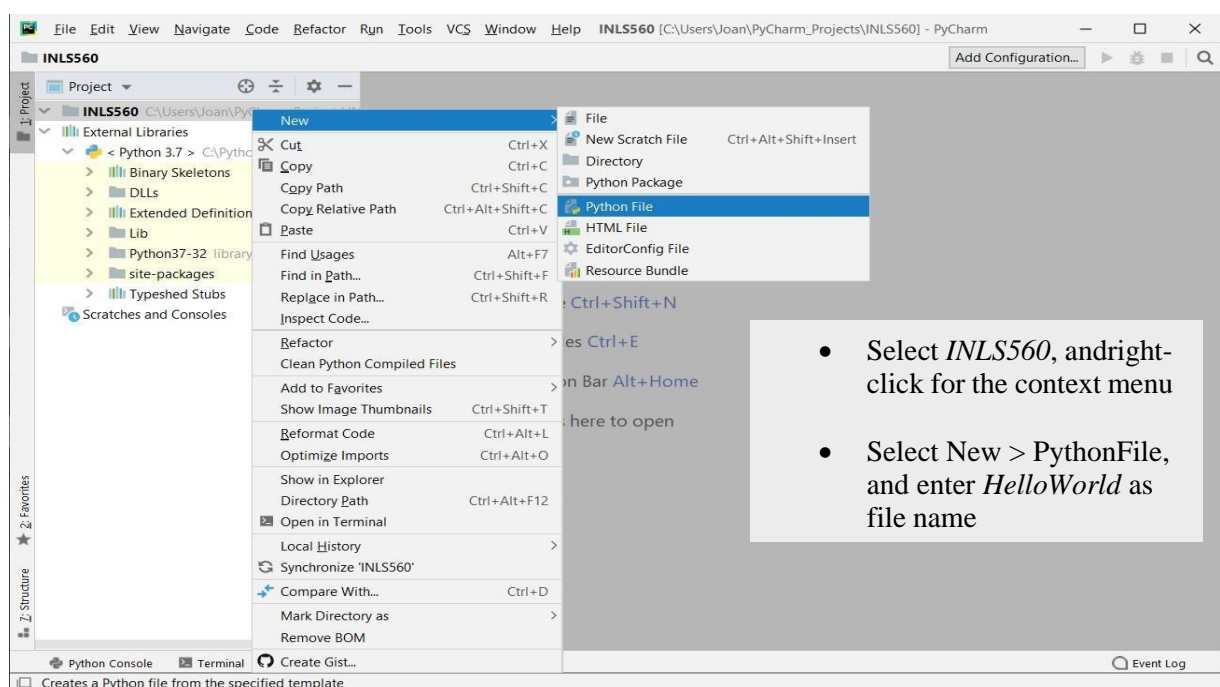


Initial project structure for *INLS560*


- Folder for your project
- Folder with External Libraries

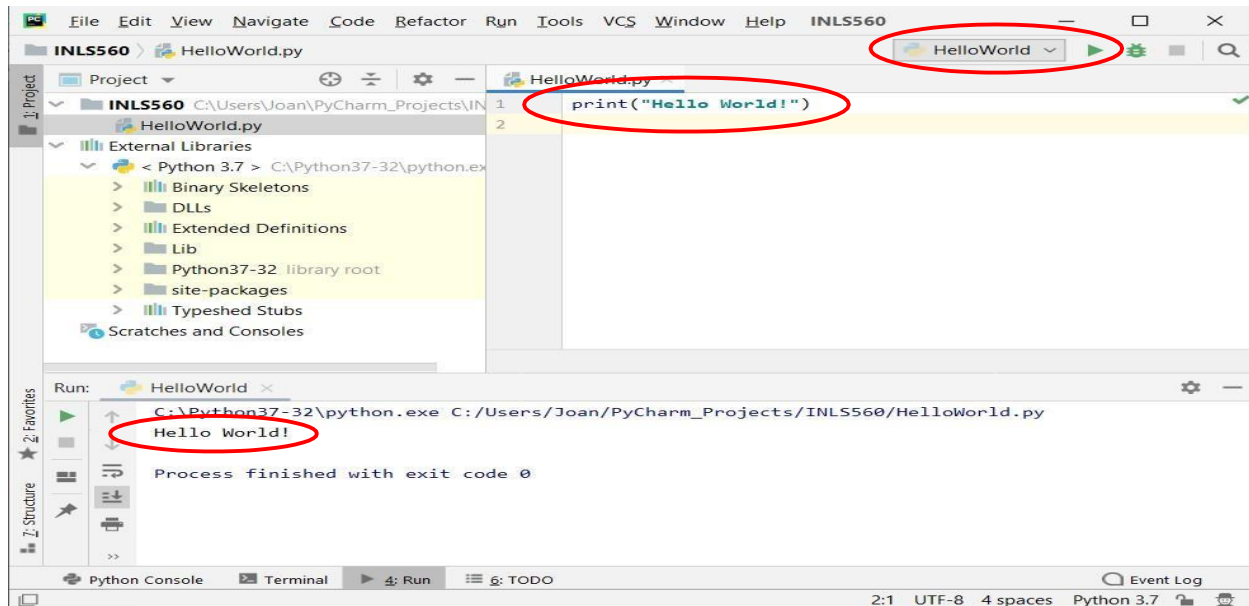


Create a Python Program

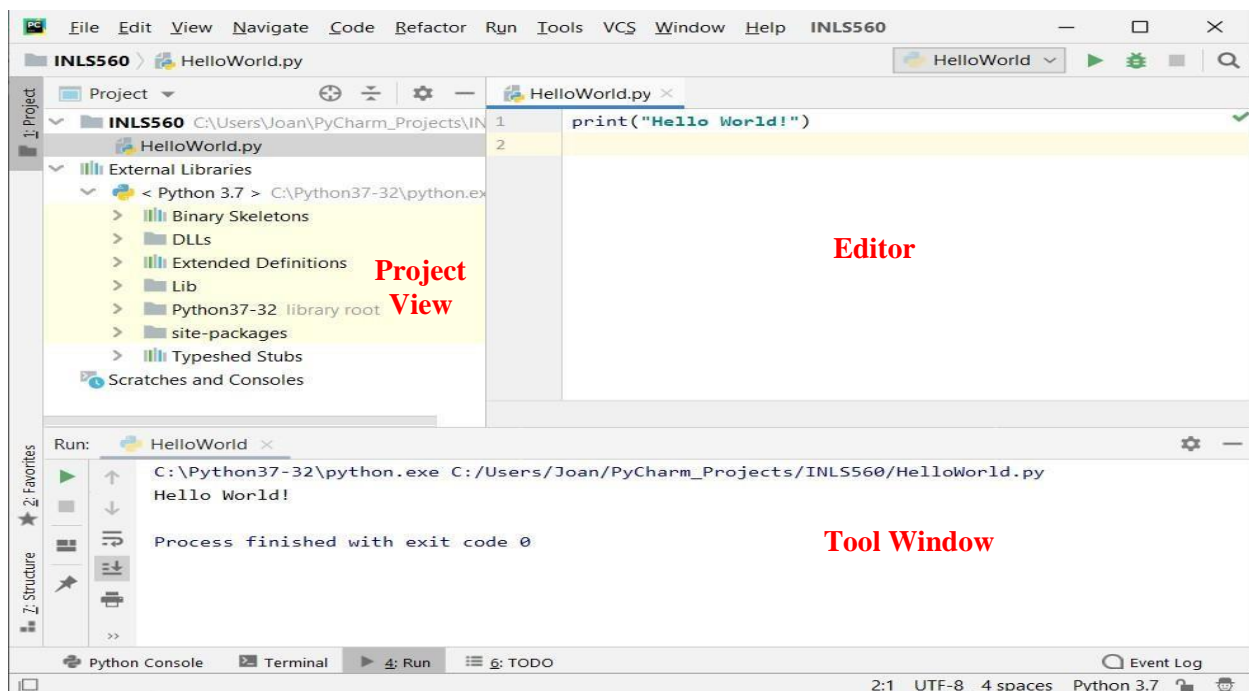


Create and run your program

- Enter `print("Hello World!")` in the Editor
- Select **HelloWorld.py** and select Run from context menu; or, select Run icon 
- Output is displayed in the Run Window in the bottom pane



Default Window Layout



EXPERIMENT -1

Aim: (3) A study was conducted to understand the effect of number of hours the students spent studying on their performance in the final exams. Write a code to plot line chart with number of hours spent studying on x-axis and score in final exam on y-axis. Use a red '*' as the point character, label the axes and give the plot a title.

Students performance in the final exams

Number of hrs spent studying (x)	10	9	2	15	10	16	11	16
Score in the final exam (0 - 100) (y)	95	80	10	50	45	98	38	93

```
import matplotlib.pyplot as plt

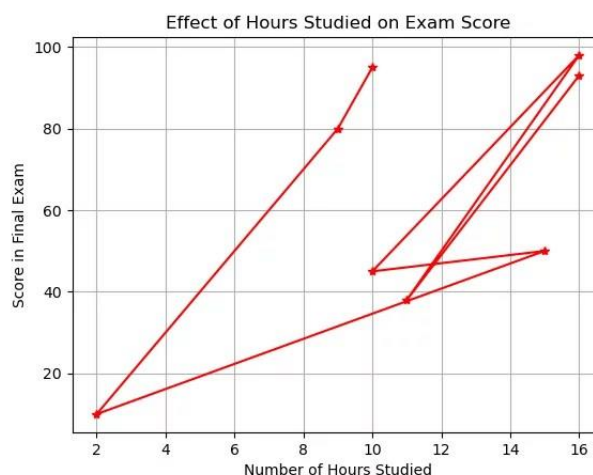
hours = [10,9,2,15,10,16,11,16]
score = [95,80,10,50,45,98,38,93]

# Plotting the line chart
plt.plot(hours, score, marker='*', color='red', linestyle='-')

# Adding labels and title
plt.xlabel('Number of Hours Studied')
plt.ylabel('Score in Final Exam')
plt.title('Effect of Hours Studied on Exam Score')

# Displaying the plot
plt.grid(True)
plt.show()
```

Output



The program above demonstrates a clear trend: generally, the more hours students study, the better they perform on the final exam. However, there are some cases where this relationship isn't quite as straightforward, yielding slightly different outcomes.

Aim: (4) For the given dataset mtcars.csv (www.kaggle.com/ruiromanini/mtcars), plot a histogram to check the frequency distribution of the variable 'mpg' (Miles per gallon)

Histogram to check the frequency distribution

```
import pandas as pd
import matplotlib.pyplot as plt

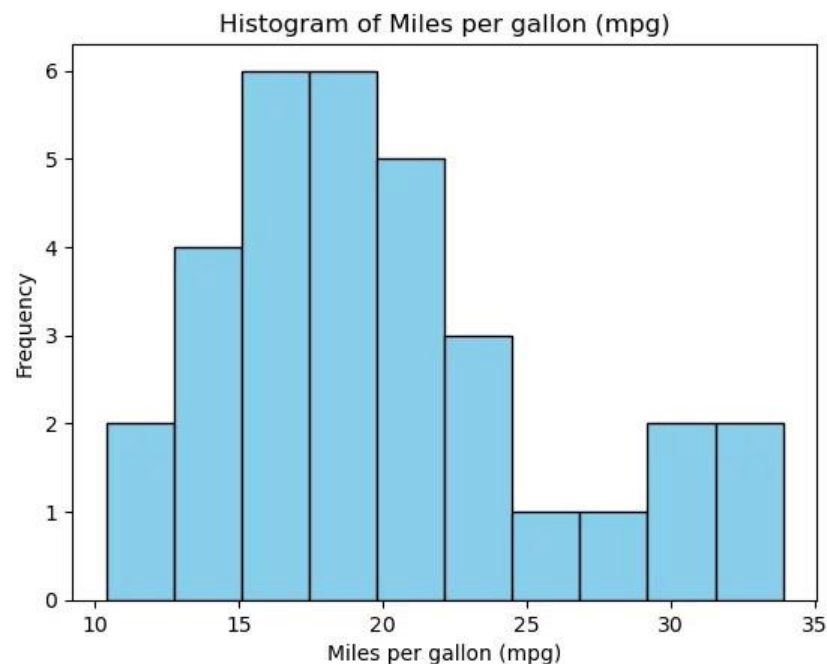
# Load the dataset
mtcars = pd.read_csv('mtcars.csv') # Replace 'path_to_your_mtcars.csv' with the actual path to your mtcars.csv file

# Plotting the histogram
plt.hist(mtcars['mpg'], bins=10, color='skyblue', edgecolor='black')

# Adding labels and title
plt.xlabel('Miles per gallon (mpg)')
plt.ylabel('Frequency')
plt.title('Histogram of Miles per gallon (mpg)')

# Displaying the plot
plt.show()
```

Output



EXPERIMENT -2

Aim: Consider the books dataset BL-Flickr-Images-Book.csv from Kaggle (<https://www.kaggle.com/adeyoyintemidayo/publication-of-books>) which contains information about books. Write a program to demonstrate the following.

- Import the data into a DataFrame
- Find and drop the columns which are irrelevant for the book information.
- Change the Index of the DataFrame
- Tidy up fields in the data such as date of publication with the help of simple regular expression.
- Combine str methods with NumPy to clean columns

Kaggle Book Data set

```
import pandas as pd
import numpy as np

# Import the data into a DataFrame
df = pd.read_csv('BL-Flickr-Images-Book.csv')

# Display the first few rows of the DataFrame
print("Original DataFrame:")
print(df.head())

# Find and drop the columns which are irrelevant for the book information
irrelevant_columns = ['Edition Statement', 'Corporate Author', 'Corporate Contributors', 'Former owner', 'Engraver',
'Contributors', 'Issuance type', 'Shelfmarks']
df.drop(columns=irrelevant_columns, inplace=True)

# Change the Index of the DataFrame
df.set_index('Identifier', inplace=True)

# Tidy up fields in the data such as date of publication with the help of simple regular expression
df['Date of Publication'] = df['Date of Publication'].str.extract(r'^(\d{4})', expand=False)

# Combine str methods with NumPy to clean columns
df['Place of Publication'] = np.where(df['Place of Publication'].str.contains('London'), 'London', df['Place of
Publication'].str.replace('-', ' '))
```



```
# Display the cleaned DataFrame
print("\nCleaned DataFrame:")
print(df.head())
```

Output

Original DataFrame:

	Identifier	Edition Statement	Place of Publication \
0	206	NaN	London
1	216	NaN	London; Virtue & Yorston
2	218	NaN	London
3	472	NaN	London
4	480	A new edition, revised, etc.	London

	Date of Publication	Publisher \
0	1879 [1878]	S. Tinsley & Co.
1	1868	Virtue & Co.
2	1869	Bradbury, Evans & Co.
3	1851	James Darling
4	1857	Wertheim & Macintosh

	Title	Author \
0	Walter Forbes. [A novel.]	By A. A. A. A.
1	All for Greed. [A novel. The dedication signed...	A., A. A.
2	Love the Avenger. By the author of "All for Gr...	A., A. A.
3	Welsh Sketches, chiefly ecclesiastical, to the...	A., E. S.
4	[The World in which I live, and my place in it...	A., E. S.

	Contributors	Corporate Author \
0	FORBES, Walter.	NaN
1	BLAZE DE BURY, Marie Pauline Rose - Baroness	NaN
2	BLAZE DE BURY, Marie Pauline Rose - Baroness	NaN
3	Appleyard, Ernest Silvanus.	NaN
4	BROOME, John Henry.	NaN

	Corporate Contributors	Former owner	Engraver	Issuance type \
0	NaN	NaN	NaN	monographic
1	NaN	NaN	NaN	monographic

2	NaN	NaN	NaN	monographic
3	NaN	NaN	NaN	monographic
4	NaN	NaN	NaN	monographic

Flickr URL \

- 0 <http://www.flickr.com/photos/britishlibrary/ta...>
- 1 <http://www.flickr.com/photos/britishlibrary/ta...>
- 2 <http://www.flickr.com/photos/britishlibrary/ta...>
- 3 <http://www.flickr.com/photos/britishlibrary/ta...>
- 4 <http://www.flickr.com/photos/britishlibrary/ta...>

Shelfmarks

- 0 British Library HMNTS 12641.b.30.
- 1 British Library HMNTS 12626.cc.2.
- 2 British Library HMNTS 12625.dd.1.
- 3 British Library HMNTS 10369.bbb.15.
- 4 British Library HMNTS 9007.d.28.

Cleaned DataFrame:

Identifier	Place of Publication	Date of Publication	Publisher \
206	London	1879	S. Tinsley & Co.
216	London	1868	Virtue & Co.
218	London	1869	Bradbury, Evans & Co.
472	London	1851	James Darling
480	London	1857	Wertheim & Macintosh

Title Author \

Identifier

- 206 Walter Forbes. [A novel.] By A. A. A. A.
- 216 All for Greed. [A novel. The dedication signed... A., A. A.
- 218 Love the Avenger. By the author of "All for Gr... A., A. A.
- 472 Welsh Sketches, chiefly ecclesiastical, to the... A., E. S.
- 480 [The World in which I live, and my place in it... A., E. S.

Flickr URL

Identifier

206 <http://www.flickr.com/photos/britishlibrary/ta...>
216 <http://www.flickr.com/photos/britishlibrary/ta...>
218 <http://www.flickr.com/photos/britishlibrary/ta...>
472 <http://www.flickr.com/photos/britishlibrary/ta...>
480 <http://www.flickr.com/photos/britishlibrary/ta...>

EXPERIMENT -3

Aim: (1) Train a regularized logistic regression classifier on the iris dataset (<https://archive.ics.uci.edu/ml/machine-learning-databases/iris/> or the inbuilt iris dataset) using sklearn. Train the model with the following hyperparameter $C = 1e4$ and report the best classification accuracy.

Logistic Regression

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline

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

# 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)

# Create a pipeline with StandardScaler and LogisticRegression with regularization
pipeline = make_pipeline(StandardScaler(), LogisticRegression(C=1e4, max_iter=1000))

# Train the model
pipeline.fit(X_train, y_train)

# Calculate the accuracy on the testing set
accuracy = pipeline.score(X_test, y_test)
print("Classification accuracy:", accuracy)
```

Output

Classification accuracy: 1.0

Aim: (2) Train an SVM classifier on the iris dataset using sklearn. Try different kernels and the associated hyperparameters. Train model with the following set of hyperparameters RBFkernel, gamma=0.5, one-vs-rest classifier, no-feature-normalization. Also try $C=0.01, 1, 10$ and $C=0.01, 1, 10$. For the above set of hyperparameters, find the best classification accuracy along with total number of support vectors on the test data.

SVM classifier

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
```

```
# 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)

# Set of hyperparameters to try
hyperparameters = [
    {'kernel': 'rbf', 'gamma': 0.5, 'C': 0.01},
    {'kernel': 'rbf', 'gamma': 0.5, 'C': 1},
    {'kernel': 'rbf', 'gamma': 0.5, 'C': 10}
]

best_accuracy = 0
best_model = None
best_support_vectors = None

# Train SVM models with different hyperparameters and find the best accuracy
for params in hyperparameters:
    model = SVC(kernel=params['kernel'], gamma=params['gamma'], C=params['C'], decision_function_shape='ovr')
    model.fit(X_train, y_train)
    accuracy = model.score(X_test, y_test)
    support_vectors = model.n_support_.sum()
    print(f"For hyperparameters: {params}, Accuracy: {accuracy}, Total Support Vectors: {support_vectors}")
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model = model
        best_support_vectors = support_vectors

print("\nBest accuracy:", best_accuracy)
print("Total support vectors on test data:", best_support_vectors)
```

Output

```
For hyperparameters: {'kernel': 'rbf', 'gamma': 0.5, 'C': 0.01}, Accuracy: 0.3, Total Support Vectors: 120
For hyperparameters: {'kernel': 'rbf', 'gamma': 0.5, 'C': 1}, Accuracy: 1.0, Total Support Vectors: 39
For hyperparameters: {'kernel': 'rbf', 'gamma': 0.5, 'C': 10}, Accuracy: 1.0, Total Support Vectors: 31
```

```
Best accuracy: 1.0
Total support vectors on test data: 39
```

EXPERIMENT - 4

Aim: (1) Consider the following dataset. Write a program to demonstrate the working of the decision tree based ID3 algorithm.

Decision Tree based ID3 algorithm

Price	Maintenance	Capacity	Airbag	Profitable
Low	Low	2	No	Yes
Low	Med	4	Yes	Yes
Low	Low	4	No	Yes
Low	Med	4	No	No
Low	High	4	No	No
Med	Med	4	No	No
Med	Med	4	Yes	Yes
Med	High	2	Yes	No
Med	High	5	No	Yes
High	Med	4	Yes	Yes
high	Med	2	Yes	Yes
High	High	2	Yes	No
high	High	5	yes	Yes

```
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import pandas as pd
from io import StringIO
from IPython.display import Image
import pydotplus

# Define the dataset
data = {
    'Price': ['Low', 'Low', 'Low', 'Low', 'Low', 'Med', 'Med', 'Med', 'Med', 'High', 'High', 'High', 'High'],
    'Maintenance': ['Low', 'Med', 'Low', 'Med', 'High', 'Med', 'Med', 'High', 'High', 'Med', 'Med', 'High', 'High'],
    'Capacity': ['2', '4', '4', '4', '4', '4', '4', '2', '5', '4', '2', '2', '5'],
    'Airbag': ['No', 'Yes', 'No', 'No', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes'],
    'Profitable': [1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1]
}

df = pd.DataFrame(data)

# Convert categorical variables into numerical ones
df = pd.get_dummies(df, columns=['Price', 'Maintenance', 'Airbag'])
```

```
# Separate features and target variable
X = df.drop('Profitable', axis=1)
y = df['Profitable']

# 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)

# Create a decision tree classifier
clf = DecisionTreeClassifier(criterion='entropy')

# Train the classifier on the training data
clf.fit(X_train, y_train)

# Predict on the testing data
y_pred = clf.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

# Visualize the decision tree
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data, filled=True, rounded=True, special_characters=True,
                feature_names=X.columns)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())

Image(graph.create_png())
```

Output

Accuracy: 0.6666666666666666

Aim: (2) Consider the dataset spiral.txt (<https://bit.ly/2Lm75Ly>). The first two columns in the dataset corresponds to the co-ordinates of each data point. The third column corresponds to the actual cluster label. Compute the rand index for the following methods:

- K – means Clustering
- Single – link Hierarchical Clustering
- Complete link hierarchical clustering.
- Also visualize the dataset and which algorithm will be able to recover the true clusters.

Clustering

```
import numpy as np
from sklearn.cluster import KMeans, AgglomerativeClustering
from sklearn.metrics import adjusted_rand_score

import matplotlib.pyplot as plt

# Load the dataset
data = np.loadtxt("Spiral.txt", delimiter=",", skiprows=1)
X = data[:, :2] # Features
y_true = data[:, 2] # Actual cluster labels

# Visualize the dataset
plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=y_true, cmap='viridis')
plt.title("True Clusters")
plt.xlabel('X1')
plt.ylabel('X2')
plt.show()

# K-means clustering
# kmeans = KMeans(n_clusters=3, random_state=42)
kmeans = KMeans(n_clusters=3, random_state=42, n_init=10)
kmeans_clusters = kmeans.fit_predict(X)

# Single-link Hierarchical Clustering
single_link = AgglomerativeClustering(n_clusters=3, linkage='single')
single_link_clusters = single_link.fit_predict(X)

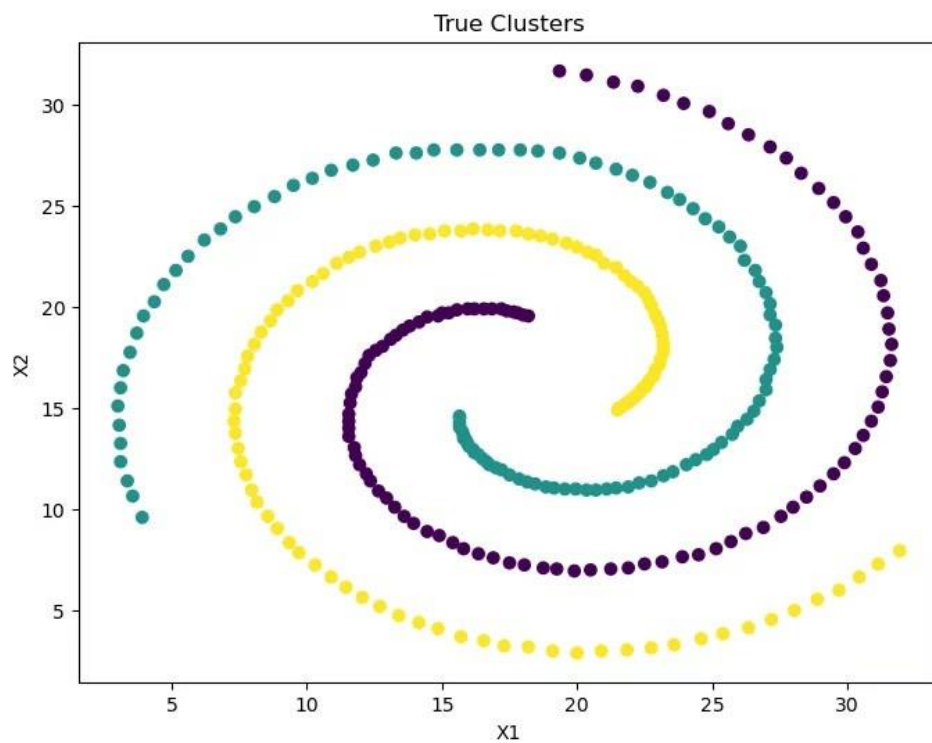
# Complete-link Hierarchical Clustering
complete_link = AgglomerativeClustering(n_clusters=3, linkage='complete')
complete_link_clusters = complete_link.fit_predict(X)

# Compute the Rand Index
rand_index_kmeans = adjusted_rand_score(y_true, kmeans_clusters)
rand_index_single_link = adjusted_rand_score(y_true, single_link_clusters)
rand_index_complete_link = adjusted_rand_score(y_true, complete_link_clusters)

print("Rand Index for K-means Clustering:", rand_index_kmeans)
print("Rand Index for Single-link Hierarchical Clustering:", rand_index_single_link)
print("Rand Index for Complete-link Hierarchical Clustering:", rand_index_complete_link)

# This code will compute the Rand Index for each clustering method and provide a visualization of the true clusters.
# The Rand Index ranges from 0 to 1, where 1 indicates perfect clustering agreement with the true clusters.
# The method with a higher Rand Index is better at recovering the true clusters.
```


Output



Rand Index for K-means Clustering: -0.0060252369726964605
Rand Index for Single-link Hierarchical Clustering: 1.0
Rand Index for Complete-link Hierarchical Clustering: 0.001841037699419282

EXPERIMENT -5

Aim: Mini Project – Simple web scrapping in social media

Mini Project

```
import requests
from bs4 import BeautifulSoup

# URL of the Instagram profile you want to scrape
url = 'https://www.instagram.com/openai/'

# Send a GET request to the URL
response = requests.get(url)

print(response.status_code)

# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the HTML content of the page
    soup = BeautifulSoup(response.text, 'html.parser')

    # Find all post elements
    posts = soup.find_all('div', class_='v1Nh3')

    # Extract data from each post
    for post in posts:
        print("Hi")
        # Extract post link
        post_link = post.find('a')['href']

        # Extract post image URL
        image_url = post.find('img')['src']

        print(f"Post Link: {post_link}")
        print(f"Image URL: {image_url}")
        print("-----")
else:
    print("Failed to retrieve data from Instagram")
```

Output

200