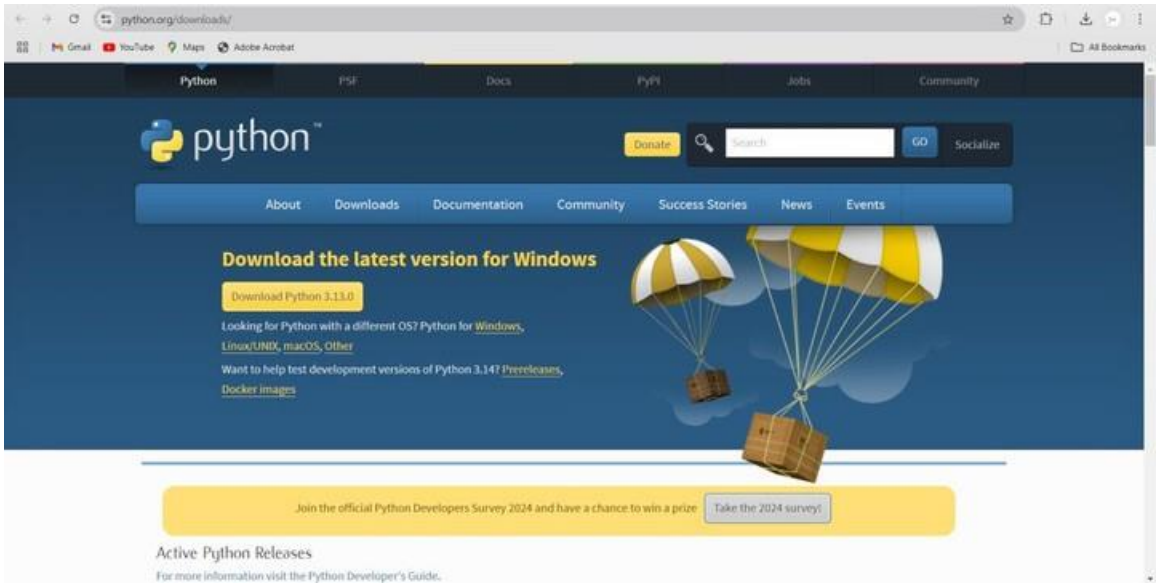
Dimensionality reduction , Hierarchical clustering in machine learning

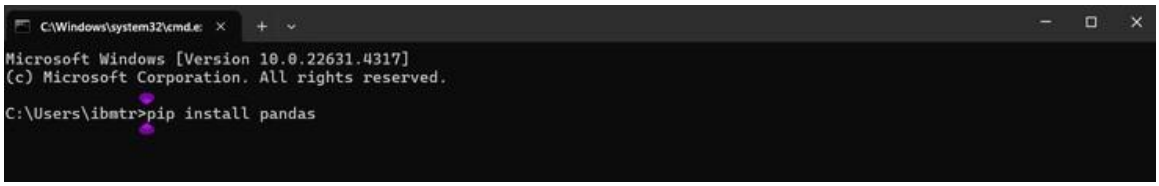
Prerequisites

1. Install Python: Make sure you have Python installed. You can download it from Python's official website (https://www.python.org/downloads/).

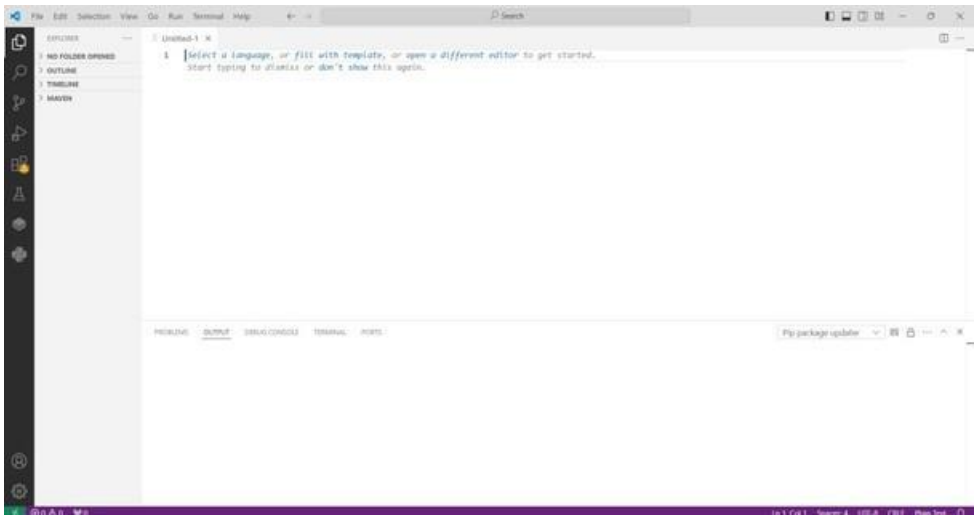


2. Install Required Libraries: You will need the following libraries: 'pandas', 'numpy', and 'matplotlib'. You can install them using pip.

```
pip install pandas numpy matplotlib
```



3. Set Up Your IDE: You can use any Python IDE or text editor (like Jupyter Notebook, VS Code, or PyCharm).



Step 1: Gather Data

For demonstration, let's create a sample dataset in CSV format. Save the following data in a file named 'business_data.csv'.

```
CustomerID,Name,Email,JoinDate,AmountSpent
1,John Doe,john@example.com,2024-01-15,150.00
2,Jane Smith,jane@example.com,2024-02-20,200.00
3,Bob Johnson,,2024-03-05,150.00
4,Mary Johnson,mary@example.com,2024-02-30,300.00
5,Tom Brown,tom@example.com,2024-03-15,400.00
6,Emily Davis,emily@example.com,2024-01-25,
1,John Doe,john@example.com,2024-01-15,150.00
```

Step 2: Load the Data

Use Pandas to load the dataset and inspect its contents.

```
# Load a sample dataset (Iris dataset)

data = load_iris()

df = pd.DataFrame(data.data,

columns=data.feature_names)

print(df.head())
```

```
PS C:\Users\ibmtr> & C:/Users/ibmtr/anaconda3/python.exe c:/Users/ibmtr/Downloads/sample.py
   CustomerID          Name              Email    JoinDate  AmountSpent
0           1      John Doe  john@example.com  2024-01-15        150.0
1           2    Jane Smith  jane@example.com  2024-02-20        200.0
2           3   Bob Johnson              NaN  2024-03-05        150.0
3           4  Mary Johnson  mary@example.com  2024-02-30        300.0
4           5     Tom Brown   tom@example.com  2024-03-15        400.0
PS C:\Users\ibmtr>
```

Step 3: Dimensionality Reduction Techniques

Dimensionality reduction helps in reducing the number of features while retaining essential patterns.

a. Principal Component Analysis (PCA)

```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)

df_pca = pca.fit_transform(df)

print(df_pca[:5])
```

Sample Output:

```
[[-2.68412563 0.31939725]

[-2.71414169 -0.17700123]

[-2.88899057 -0.14494943]

[-2.74534286 -0.31829898]

[-2.72871654 0.32675451]]
```

b. t-Distributed Stochastic Neighbor Embedding (t-SNE)

Sample Code:

```
from sklearn.manifold import TSNE
tsne = TSNE(n_components=2, random_state=42)
df_tsne = tsne.fit_transform(df)
print(df_tsne[:5])
```

Sample Output:

```
[[ 1.2379045 12.769159 ]
 [ 8.755232  7.7505245]
 [ 9.419792  8.941869 ]
 [ 9.378086  7.217551 ]
 [ 2.849782  6.5989175]]
```

Step 4: Hierarchical Clustering

Sample Code:

```
from scipy.cluster.hierarchy import dendrogram, linkage
import matplotlib.pyplot as plt
linked = linkage(df, method='ward')
plt.figure(figsize=(10, 7))
dendrogram(linked, truncate_mode='lastp')
```

```
plt.title("Hierarchical Clustering Dendrogram")
plt.show()
```

Expected Output: A dendrogram plot will display showing hierarchical relationships between data points.

Step 5: Evaluation and Visualization

Sample Code:

```
from sklearn.metrics import silhouette_score
from sklearn.cluster import AgglomerativeClustering
cluster = AgglomerativeClustering(n_clusters=3)
labels = cluster.fit_predict(df)
score = silhouette_score(df, labels)
print("Silhouette Score:", score)
```

Sample Output:

```
Silhouette Score: 0.554323
```

This score evaluates clustering quality, where higher values indicate better-defined clusters.