hw1-akshajkammari

June 17, 2024

Setup

[1]:
```
pip install pandas
```

```
Requirement already satisfied: pandas in
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages
(2.2.2)
Requirement already satisfied: numpy>=1.26.0 in
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages
(from pandas) (2.0.0)
Requirement already satisfied: python-dateutil>=2.8.2 in
/Users/akammari/Library/Python/3.12/lib/python/site-packages (from pandas)
(2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages
(from pandas) (2024.1)
Requirement already satisfied: tzdata>=2022.7 in
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages
(from pandas) (2024.1)
Requirement already satisfied: six>=1.5 in
/Users/akammari/Library/Python/3.12/lib/python/site-packages (from python-
dateutil>=2.8.2->pandas) (1.16.0)

[notice] A new release of pip is
available: 23.2.1 -> 24.0
[notice] To update, run:
pip3 install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

[2]:
```
pip install numpy
```

```
Requirement already satisfied: numpy in
/Library/Frameworks/Python.framework/Versions/3.12/lib/python3.12/site-packages
(2.0.0)

[notice] A new release of pip is
available: 23.2.1 -> 24.0
[notice] To update, run:
```

```
pip3 install --upgrade pip
Note: you may need to restart the kernel to use updated packages.
```

[3]:
```
import pandas as pd
import numpy as np
```

Theory Questions

1. Explain the difference between a NumPy array and a Python list, and write down examples of defining a NumPy array and a Python list.

A NumPy array is a N-dimensional array which takes the form of rows and columns. It supports math operations on these arrays and is faster than Python lists because of its fixed type and continuous memory allocation. A list is a collection which is ordered and changeable. It allows duplicate members and can hold a mixture of data types.

Numpy Array example:

[4]:
```
numpy_array = np.array([1, 2, 3, 4, 5])
numpy_array
```

[4]:
```
array([1, 2, 3, 4, 5])
```

Python List example:

[5]:
```
python_list = [1, 2, 3, 4, 5]
python_list
```

[5]:
```
[1, 2, 3, 4, 5]
```

2. Explain the difference between a DataFrame and a Series in Pandas, and write down examples of defining a DataFrame and a Series in Pandas.

A DataFrame is a 2-dimensional labeled data structure with columns having the ability to store different types, similar to a table in a database or an Excel spreadsheet. A Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). It is similar to a column in a DataFrame.

Pandas DataFrame example:

[6]:
```
data = {'column1': [1, 2, 3, 4], 'column2': [5, 6, 7, 8]}
dataframe = pd.DataFrame(data)
dataframe
```

[6]:

	column1	column2
0	1	5
1	2	6
2	3	7
3	4	8

Pandas Series example:

[7]:
```
series = pd.Series([1, 2, 3, 4, 5])
series
```

[7]:
```
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

Data Manipulation with Pandas

1. Load the dataset into a Pandas DataFrame and add appropriate column names: ['sepal length', 'sepal width', 'petal length', 'petal width', 'class'].

[8]:
```
#loading the dataset from the local file "iris.data"
file_path = 'iris.data'
column_names = ['sepal_length', 'sepal_width', 'petal_length', 'petal_width',␣
 ↪'class']
df = pd.read_csv(file_path, names=column_names)

df.head()
```

[8]:

	sepal_length	sepal_width	petal_length	petal_width	class
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

2. Display the first 10 rows of the DataFrame.

[9]:
```
df.head(10)
```

[9]:

	sepal_length	sepal_width	petal_length	petal_width	class
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
5	5.4	3.9	1.7	0.4	Iris-setosa
6	4.6	3.4	1.4	0.3	Iris-setosa
7	5.0	3.4	1.5	0.2	Iris-setosa
8	4.4	2.9	1.4	0.2	Iris-setosa
9	4.9	3.1	1.5	0.1	Iris-setosa

3. Calculate the mean, median, and standard deviation of the sepal length for each class.

Mean Sepal Length per Class:

[10]:
```
mean_sepal_length = df.groupby('class')['sepal_length'].mean()
mean_sepal_length
```

[10]:
```
class
Iris-setosa        5.006
Iris-versicolor    5.936
Iris-virginica     6.588
Name: sepal_length, dtype: float64
```

Median Sepal Length per Class:

[11]:
```
median_sepal_length = df.groupby('class')['sepal_length'].median()
median_sepal_length
```

[11]:
```
class
Iris-setosa        5.0
Iris-versicolor    5.9
Iris-virginica     6.5
Name: sepal_length, dtype: float64
```

Standard Deviation Sepal Length per Class:

[12]:
```
std_sepal_length = df.groupby('class')['sepal_length'].std()
std_sepal_length
```

[12]:
```
class
Iris-setosa        0.352490
Iris-versicolor    0.516171
Iris-virginica     0.635880
Name: sepal_length, dtype: float64
```

4. Filter the DataFrame to include only rows where the petal length is greater than 1.5 and display the first 5 rows of the filtered DataFrame.

[13]:
```
filtered_df = df[df['petal_length'] > 1.5]
filtered_df.head()
```

[13]:

	sepal_length	sepal_width	petal_length	petal_width	class
5	5.4	3.9	1.7	0.4	Iris-setosa
11	4.8	3.4	1.6	0.2	Iris-setosa
18	5.7	3.8	1.7	0.3	Iris-setosa
20	5.4	3.4	1.7	0.2	Iris-setosa
23	5.1	3.3	1.7	0.5	Iris-setosa