Nama : Ananda Sheva Hidayat
NPM : 2217051096
Kelas : B

Ulangan Akhir Semester / Praktikum Pemrograman Interpreter

Ketentuan Laporan

1.) Jelaskan fungsi yang dibuat untuk kasus diatas

Program di atas membuat fungsi untuk menghitung statistik sederhana (rata-rata, median, dan modus) dari dua kolom dalam sebuah DataFrame

2.) Jelaskan class dan method yang dibuat untuk kasus diatas

1. CustomStatistics (Class) Deskripsi: Kelas ini bertanggung jawab menyediakan fungsi penghitungan statistik pada suatu DataFrame.

Method :

calculate_mean(self, nama_kolom): Menghitung rata-rata dari kolom dengan nama tertentu. Menggunakan fungsi np.mean dari NumPy untuk perhitungan yang lebih efisien.

calculate_median(self, nama_kolom): Menghitung median dari kolom dengan nama tertentu. Menggunakan fungsi np.median dari NumPy.

calculate_mode(self, nama_kolom): Menghitung modus dari kolom dengan nama tertentu. Menggunakan fungsi np.argmax dan np.bincount dari NumPy.

2. TestCustomStatistics (Class Pengujian) Deskripsi: Class pengujian unit menggunakan modul unittest untuk memastikan implementasi yang benar pada CustomStatistics.

Metode Pengujian:

setUp(self): Method yang dipanggil sebelum setiap pengujian untuk menyiapkan DataFrame contoh dan instance CustomStatistics.

3.) Jelaskan pengujian menggunakan error handling untuk kasus diatas

Dengan melakukan pengujian menggunakan error handling seperti di atas, kita dapat memverifikasi bahwa kelas CustomStatistics tidak hanya memberikan hasil yang benar dalam kondisi normal tetapi juga dapat menangani kondisi kesalahan dengan benar

1. test_invalid_input(self) :Pengujian ini memastikan bahwa kelas CustomStatistics dapat menangani input yang tidak valid dengan memunculkan ValueError jika input yang diberikan bukan merupakan objek pandas DataFrame.

2. test_zero_division_error(self) : Pengujian ini memastikan bahwa kelas CustomStatistics dapat menangani kesalahan pembagian nol dengan memunculkan ZeroDivisionError jika ditemui kondisi pembagian nol.

3. test_index_error(self) : Pengujian ini memastikan bahwa kelas CustomStatistics dapat menangani kesalahan indeks dengan memunculkan IndexError jika ditemui kondisi indeks yang tidak valid.

4.) Jelaskan pengujian menggunakan unit testing untuk kasus diatas

Pengujian menggunakan unit testing dilakukan dalam kelas TestCustomStatistics pada file unit_test.py. Dalam pengujian ini, beberapa metode pengujian dibuat untuk menguji fungsionalitas dari metode-metode yang ada dalam kelas CustomStatistics. Berikut adalah penjelasan untuk setiap metode pengujian unit :

1. test_mean_calculation(self) : Method ini menguji apakah perhitungan rata-rata pada kolom tertentu berfungsi dengan benar.

2. test_median_calculation(self) : Method ini menguji apakah perhitungan median pada kolom tertentu berfungsi dengan benar.

3. test_mode_calculation(self) : Method ini menguji apakah perhitungan modus pada kolom tertentu berfungsi dengan benar.

4. test_invalid_input(self) : Method ini menguji apakah kelas CustomStatistics dapat menangani input yang tidak valid dengan memunculkan ValueError jika input yang diberikan bukan merupakan objek pandas DataFrame.

5. test_zero_division_error(self) : Method ini menguji apakah Class CustomStatistics dapat menangani kesalahan pembagian nol dengan memunculkan ZeroDivisionError jika ditemui kondisi pembagian nol.

6. test_index_error(self) : Method ini menguji apakah kelas CustomStatistics dapat menangani kesalahan indeks dengan memunculkan IndexError jika ditemui kondisi indeks yang tidak valid.

5.) Jelaskan penggunaan modules, packages dan library untuk kasus diatas

Pada program diatas, konsep modul, paket (package), dan pustaka (library) digunakan untuk memisahkan fungsionalitas program menjadi unit-unit yang terkelompok. Berikut penjelasan untuk penggunaan modul, paket, dan pustaka dalam kasus tersebut:

<u>Modules:</u>

stats.py : Modul ini berisi kelas CustomStatistics yang menyediakan fungsi-fungsi untuk menghitung statistik dari DataFrame.
Penggunaan : Modul ini diimpor dalam main.py dan unit_test.py untuk menggunakan perintah yang ada.

main.py : Program utama yang menggunakan perintah dari stats.py. Menyiapkan DataFrame contoh dan menghitung statistik menggunakan CustomStatistics.
Penggunaan : Mengimpor statistics untuk menggunakan kelas CustomStatistics.

unit_test.py : Modul ini berisi kelas TestCustomStatistics yang menggunakan modul unittest untuk melakukan pengujian unit pada CustomStatistics.
Penggunaan : Mengimpor statistics dan unittest untuk melakukan pengujian unit.

Packages:

tests : Paket ini berisi modul test_statistics.py yang digunakan untuk melakukan pengujian unit pada kelas CustomStatistics.

Libraries:

pandas : Library ini digunakan untuk manipulasi dan analisis data dalam bentuk DataFrame.
Penggunaan : Digunakan dalam main.py untuk membuat DataFrame dan memanipulasinya.

numpy : Library ini digunakan untuk operasi matematika yang efisien, seperti perhitungan mean, median, dan modus.
Penggunaan : Dalam stats.py, fungsi-fungsi dari numpy digunakan untuk perhitungan statistik.

unittest : Library ini digunakan untuk melakukan pengujian unit.
Penggunaan : Dalam unit_test.py, modul ini digunakan untuk membuat kelas pengujian dari CustomStatistics

6.) Jelaskan motivasi anda terkait pembelajaran pemrograman dikemudian hari (contoh proyek, pekerjaan, dll)

Motivasi saya terkait pembelajaran pemrograman kedepannya adalah bahwa saya paham dan mampu menguasai berbagai bahasa pemrograman dan skill khususnya yang mengarah pada Web Developer. Saat ini saya sedang mendalami HTML dan CSS dan pernah membuat projek sederhana yaitu portfolio web. Saya harap lebih bisa mendalami bahasa pemrograman lain seperti JavaScript, Python dan berbagai library yang digunakan agar lebih paham dan matang dalam keikutsertaan proyek proyek yang akan dibuat selanjutnya. Dan harapan saya kedepannya, saya bisa bekerja di bidang yang sesuai keinginan dan menjadi front end developer

Program Akhir

```python
#main.py
from stats import CustomStatistics
import pandas as pd

#Menyiapkan DataFrame
data = {'Fee': [20000, 25000, 30000, 22000, 26000],
        'Total': [18000, 22000, 27000, 19800,
df000]}pd.DataFrame(data)

#Membuat instance method dari CustomStatistics
custom_stats = CustomStatistics(df)

#Menghitung mean, median, dan modus untuk 'Fee'
mean_fee = custom_stats.calculate_mean('Fee')
median_fee = custom_stats.calculate_median('Fee')
mode_fee = custom_stats.calculate_mode('Fee')

#Menghitung mean, median, dan modus untuk 'Total'
mean_total = custom_stats.calculate_mean('Total')
median_total = custom_stats.calculate_median('Total')
mode_total = custom_stats.calculate_mode('Total')

#Menampilkan hasil
print("Statistik untuk Fee")
print("Median : ", median_fee)
print("Modus : ", mode_fee)
print("Mean : ", mean_fee)

print("\nStatistik untuk Total")
print("Median : ", median_total)
print("Modus : ", mode_total)
print("Mean : ", mean_total)
```

```python
#stats.py
import pandas as pd
import numpy as np

class CustomStatistics:
    def __init__(self, data):
        #Memastikan input data merupakan pandas DataFrame
        if not isinstance(data, pd.DataFrame):
            raise ValueError("Data input harus berupa pandas Data Frame.")
        self.data = data

    def calculate_mean(self, nama_kolom):
        try:
            #Menghitung rata-rata menggunakan fungsi mean dari NumPy
            data_kolom = self.data[nama_kolom]
            mean = np.mean(data_kolom)
            return mean
        except Exception as e:
            #Menangani exception jika terjadi kesalahan
            raise Exception(f"Error menghitung rata-rata untuk kolom '{nama_kolom}':
{e}")
    def calculate_median(self, nama_kolom):
        try:
            #Menghitung median menggunakan fungsi median dari NumPy
            data_kolom = self.data[nama_kolom]
            median = np.median(data_kolom)
            return median
        except Exception as e:
            #Menangani exception jika terjadi kesalahan
            raise Exception(f"Error menghitung median untuk kolom '{nama_kolom}': {e}")

    def calculate_mode(self, nama_kolom):
        try:
            #Menghitung modus menggunakan fungsi argmax dan bincount dari NumPy
            data_kolom = self.data[nama_kolom]
            mode = float(np.argmax(np.bincount(data_kolom)))
            return mode
        except Exception as e:
            #Menangani exception jika terjadi kesalahan
            raise Exception(f"Error menghitung modus untuk kolom '{nama_kolom}': {e}")
```

```python
#unit_test.py
import unittest
import pandas as pd
from stats import CustomStatistics

class TestCustomStatistics(unittest.TestCase):
    def setUp(self):
        # Menyiapkan DataFrame contoh untuk pengujian
        data = {'Fee': [20000, 25000, 30000, 22000, 26000],
                'Total': [18000, 22000, 27000, 19800, 25000]}
        self.df = pd.DataFrame(data)
        self.custom_stats = CustomStatistics(self.df)

    def test_mean_calculation(self):
        mean_fee = self.custom_stats.calculate_mean('Fee')
        mean_total = self.custom_stats.calculate_mean('Total')

        self.assertEqual(mean_fee, 24600.0)
        self.assertEqual(mean_total, 22360.0)

    def test_median_calculation(self):
        median_fee = self.custom_stats.calculate_median('Fee')
        median_total = self.custom_stats.calculate_median('Total')

        self.assertEqual(median_fee, 25000.0)
        self.assertEqual(median_total, 22000.0)

    def test_mode_calculation(self):
        mode_fee = self.custom_stats.calculate_mode('Fee')
        mode_total = self.custom_stats.calculate_mode('Total')

        self.assertEqual(mode_fee, 20000)
        self.assertEqual(mode_total, 18000)

    def test_invalid_input(self):
        with self.assertRaises(ValueError):
            invalid_stats = CustomStatistics("bukan_dataframe")

    def test_zero_division_error(self):
        with self.assertRaises(ZeroDivisionError):
            mean_invalid = self.custom_stats.calculate_mean('KolomTidakValid')

    def test_index_error(self):
        with self.assertRaises(IndexError):
            median_invalid =
self.custom_stats.calculate_median('KolomTidakValid')
if __name__ == '__main__':
    unittest.main()
```

Output

```
C:\Users\User\AppData\Local\Microsoft\W
Statistik untuk Fee
Median :  25000.0
Modus :  20000.0
Mean :  24600.0

Statistik untuk Total
Median :  22000.0
Modus :  18000.0
Mean :  22360.0

Process finished with exit code 0
```