Hands-on Activity 6.1 | Introduction to Data Analysis and Tools

Name: Delloson, Angelo Dan Renz G.

Section: CPE22S2

Performed on: 06/20/2024

Submitted on: MM/DD/YYYY

Submitted to: Engr. Roman M. Richard

6.1 Intended Learning Outcome

1. Use pandas and numpy data analysis tools.
2. Demonstrate how to analyze data using numpy and pandas

6.2 Resources:

Personal Computer

Jupyter Notebook

Internet Connection

6.3 Supplementary Activities:

Exercise 1

Run the given code below for exercises 1 and 2, perform the given tasks without using any Python module

```
import random
random.seed(0)
salaries = [round(random.random()*1000000, -3) for _ in range(100)]
```

Using the data generated above, calculate the following statistics without importing anything from the statistics module in the standard library (https://docs.python.org/3/library/statistics.html) and then confirm your results match up to those that are obtained when using the statistics module (where possible):

Mean

Median

Mode (hint: check out the Counter in the collections module of the standard library at https://docs.python.org/3/library/collections.html#collections.Counter)

Sample variance

Sample standard deviation

```
# Write a comment per statistical function
from statistics import median
from math import isnan
from itertools import filterfalse

def cal_mean(data):
    return sum(data) / len(data)
mean = cal_mean(salaries)

def cal_median(data):
    sorted_data = sorted(data)
    n = len(sorted_data)
    midpoint = n // 2

    if n% 2 == 1:
        return sorted_data[midpoint]
    else:
        return (sorted_data[midpoint - 1] + sorted_data[midpoint]) / 2
median = cal_median
```

```
from collections import Counter

def cal_mode(data):
    frequency = Counter(data)
    mode_data = frequency.most_common(1)
    return mode_data [0][0] if mode_data else None

mode = cal_mode(salaries)


def cal_sampVar(data):
    mean = cal_mean(data)
    squared_diff = [(x - mean) ** 2 for x in data]
    return sum(squared_diff) / (len(data) -1)

sampleVariance = cal_sampVar(salaries)



def cal_stdDev(data):
    variance = cal_sampVar(data)
    return variance ** 0.5

sample_stdDev = cal_stdDev(salaries)


print("Median: ", mean)
print("Mean: ", median)
print("Mode: ", mode)
print("Sample Varieance: ", sampleVariance)
print("Sample Standard Deviation: ", sample_stdDev)
```

```
Median:  585690.0
Mean:  <function cal_median at 0x7808c4202e60>
Mode:  477000.0
Sample Varieance:  70664054444.44444
Sample Standard Deviation:  265827.11382484
```

```
import pandas as pd

data = pd.DataFrame(salaries)
data.describe()
```

	0
count	100.000000
mean	585690.000000
std	265827.113825
min	1000.000000
25%	403500.000000
50%	589000.000000
75%	816750.000000
max	996000.000000

Exercise 2

Using the same data, calculate the following statistics using the functions in the statistics module where appropriate:

Range

Coefficient of variation Interquartile range

Quartile coefficient of dispersion

```
# Write a comment per statistical function
range = max(salaries) - min(salaries) # range function

mean = cal_mean(salaries) # COV function
standardDev = cal_stdDev(salaries)
COV = (standardDev / mean) * 100

def cal_iqr(data): # interquartile range function
    sorted_data = sorted(data)
    q1 = cal_median(sorted_data[:len(sorted_data) // 2])
    q3 = cal_median(sorted_data[(len(sorted_data) + 1) // 2:])
    return q3 - q1

iqr = cal_iqr(salaries)

def cal_qd(data): # quartile dispersion function
    sorted_data = sorted(data)
    q1 = cal_median(sorted_data[:len(sorted_data) // 2])
    q3 = cal_median(sorted_data[(len(sorted_data) + 1) // 2:])
    return (q3 - q1) / (q3 + q1)

qd = cal_qd(salaries)

# output
print(f"Range:", range)
print(f"Coefficient of Variation:%", COV)
print(f"Interquartile Range:", iqr)
print(f"Quartile Coefficient of Dispersion:", qd)
```

```
Range: 995000.0
Coefficient of Variation:% 45.38699889443903
Interquartile Range: 417500.0
Quartile Coefficient of Dispersion: 0.3417928776094965
```

Exercise 3: Pandas for Data Analysis

Load the diabetes.csv file. Convert the diabetes.csv into dataframe

Perform the following tasks in the diabetes dataframe:

1. Identify the column names
2. Identify the data types of the data
3. Display the total number of records
4. Display the first 20 records
5. Display the last 20 records
6. Change the Outcome column to Diagnosis
7. Create a new column Classification that display "Diabetes" if the value of outcome is 1 , otherwise "No Diabetes"
8. Create a new dataframe "withDiabetes" that gathers data with diabetes
9. Create a new dataframe "noDiabetes" thats gathers data with no diabetes
10. Create a new dataframe "Pedia" that gathers data with age 0 to 19
11. Create a new dataframe "Adult" that gathers data with age greater than 19
12. Use numpy to get the average age and glucose value.
13. Use numpy to get the median age and glucose value.
14. Use numpy to get the middle values of glucose and age.
15. Use numpy to get the standard deviation of the skinthickness.

```
#Indicate which item you're answering with a comment

filepath = '/content/diabetes.csv'
data = pd.read_csv(filepath)
data
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigree
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

Next steps: Generate code with data | View recommended plots

```
# Identify the column names
column_names = list(data.columns)

print(column_names)
```

```
['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']
```

```
#Identify the data types of the data
dataTypes = data.dtypes

print(dataTypes)
```

```
Pregnancies                   int64
Glucose                       int64
BloodPressure                 int64
SkinThickness                 int64
Insulin                       int64
BMI                         float64
DiabetesPedigreeFunction    float64
Age                           int64
Outcome                       int64
dtype: object
```

```
# Display the total number of records
noRecords = len(data)

print(noRecords)
```

```
768
```

```
# Display the first 20 records
first20Records = data.head(20)

print(first20Records)
```

```
    Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0             6      148             72             35        0  33.6
1             1       85             66             29        0  26.6
2             8      183             64              0        0  23.3
3             1       89             66             23       94  28.1
4             0      137             40             35      168  43.1
5             5      116             74              0        0  25.6
6             3       78             50             32       88  31.0
7            10      115              0              0        0  35.3
8             2      197             70             45      543  30.5
9             8      125             96              0        0   0.0
10            4      110             92              0        0  37.6
11           10      168             74              0        0  38.0
12           10      139             80              0        0  27.1
13            1      189             60             23      846  30.1
```

```
     14            5      166             72             19      175  25.8
     15            7      100              0              0        0  30.0
     16            0      118             84             47      230  45.8
     17            7      107             74              0        0  29.6
     18            1      103             30             38       83  43.3
     19            1      115             70             30       96  34.6

         DiabetesPedigreeFunction  Age  Outcome
     0                      0.627   50        1
     1                      0.351   31        0
     2                      0.672   32        1
     3                      0.167   21        0
     4                      2.288   33        1
     5                      0.201   30        0
     6                      0.248   26        1
     7                      0.134   29        0
     8                      0.158   53        1
     9                      0.232   54        1
     10                     0.191   30        0
     11                     0.537   34        1
     12                     1.441   57        0
     13                     0.398   59        1
     14                     0.587   51        1
     15                     0.484   32        1
     16                     0.551   31        1
     17                     0.254   31        1
     18                     0.183   33        0
     19                     0.529   32        1
```

```
# Display the last 20 records
last20Records = data.tail(20)

print(last20Records)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
     748            3      187             70             22      200  36.4
     749            6      162             62              0        0  24.3
     750            4      136             70              0        0  31.2
     751            1      121             78             39       74  39.0
     752            3      108             62             24        0  26.0
     753            0      181             88             44      510  43.3
     754            8      154             78             32        0  32.4
     755            1      128             88             39      110  36.5
     756            7      137             90             41        0  32.0
     757            0      123             72              0        0  36.3
     758            1      106             76              0        0  37.5
     759            6      190             92              0        0  35.5
     760            2       88             58             26       16  28.4
     761            9      170             74             31        0  44.0
     762            9       89             62              0        0  22.5
     763           10      101             76             48      180  32.9
     764            2      122             70             27        0  36.8
     765            5      121             72             23      112  26.2
     766            1      126             60              0        0  30.1
     767            1       93             70             31        0  30.4

         DiabetesPedigreeFunction  Age  Outcome
     748                     0.408   36        1
     749                     0.178   50        1
     750                     1.182   22        1
     751                     0.261   28        0
     752                     0.223   25        0
     753                     0.222   26        1
     754                     0.443   45        1
     755                     1.057   37        1
     756                     0.391   39        0
     757                     0.258   52        1
     758                     0.197   26        0
     759                     0.278   66        1
     760                     0.766   22        0
     761                     0.403   43        1
     762                     0.142   33        0
     763                     0.171   63        0
     764                     0.340   27        0
     765                     0.245   30        0
     766                     0.349   47        1
     767                     0.315   23        0
```

```
# Change the Outcome column to Diagnosis
data.rename(columns={'Outcome': 'Diagnosis'}, inplace=True)

print(data)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Diagnosis
0                       0.627   50          1
1                       0.351   31          0
2                       0.672   32          1
3                       0.167   21          0
4                       2.288   33          1
..                        ...  ...        ...
763                     0.171   63          0
764                     0.340   27          0
765                     0.245   30          0
766                     0.349   47          1
767                     0.315   23          0

[768 rows x 9 columns]
```

```
# Create a new column Classification that display "Diabetes" if the value of outcome is 1 , otherwise "No Diabetes"
data['Classification'] = data['Diagnosis'].apply(lambda x: 'Diabetes' if x == 1 else 'No Diabetes')

print(data)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Diagnosis Classification
0                       0.627   50          1       Diabetes
1                       0.351   31          0    No Diabetes
2                       0.672   32          1       Diabetes
3                       0.167   21          0    No Diabetes
4                       2.288   33          1       Diabetes
..                        ...  ...        ...            ...
763                     0.171   63          0    No Diabetes
764                     0.340   27          0    No Diabetes
765                     0.245   30          0    No Diabetes
766                     0.349   47          1       Diabetes
767                     0.315   23          0    No Diabetes

[768 rows x 10 columns]
```

```
# Create a new dataframe "withDiabetes" that gathers data with diabetes
withDiabetes = data[data['Diagnosis'] == 1]

print(withDiabetes)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
2              8      183             64              0        0  23.3
4              0      137             40             35      168  43.1
6              3       78             50             32       88  31.0
8              2      197             70             45      543  30.5
..           ...      ...            ...            ...      ...   ...
755            1      128             88             39      110  36.5
757            0      123             72              0        0  36.3
759            6      190             92              0        0  35.5
```

```
761            9      170             74             31        0  44.0
766            1      126             60              0        0  30.1

     DiabetesPedigreeFunction  Age  Diagnosis Classification
0                       0.627   50          1       Diabetes
2                       0.672   32          1       Diabetes
4                       2.288   33          1       Diabetes
6                       0.248   26          1       Diabetes
8                       0.158   53          1       Diabetes
..                        ...  ...        ...            ...
755                     1.057   37          1       Diabetes
757                     0.258   52          1       Diabetes
759                     0.278   66          1       Diabetes
761                     0.403   43          1       Diabetes
766                     0.349   47          1       Diabetes

[268 rows x 10 columns]
```

```
# Create a new dataframe "noDiabetes" thats gathers data with no diabetes
noDiabetes = data[data['Diagnosis'] == 0]

print(noDiabetes)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
1              1       85             66             29        0  26.6
3              1       89             66             23       94  28.1
5              5      116             74              0        0  25.6
7             10      115              0              0        0  35.3
10             4      110             92              0        0  37.6
..           ...      ...            ...            ...      ...   ...
762            9       89             62              0        0  22.5
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Diagnosis Classification
1                       0.351   31          0    No Diabetes
3                       0.167   21          0    No Diabetes
5                       0.201   30          0    No Diabetes
7                       0.134   29          0    No Diabetes
10                      0.191   30          0    No Diabetes
..                        ...  ...        ...            ...
762                     0.142   33          0    No Diabetes
763                     0.171   63          0    No Diabetes
764                     0.340   27          0    No Diabetes
765                     0.245   30          0    No Diabetes
767                     0.315   23          0    No Diabetes

[500 rows x 10 columns]
```

```
# Create a new dataframe "Pedia" that gathers data with age 0 to 19
Pedia = data[(data['Age'] >= 0) & (data['Age'] <= 19)]

print(Pedia)
```

```
Empty DataFrame
Columns: [Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, Diagnosis, Classification]
Index: []
```

```
# Create a new dataframe "Adult" that gathers data with age greater than 19
Adult = data[data['Age'] > 19]

print(Adult)
```

```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35        0  33.6
1              1       85             66             29        0  26.6
2              8      183             64              0        0  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           ...      ...            ...            ...      ...   ...
763           10      101             76             48      180  32.9
764            2      122             70             27        0  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0        0  30.1
767            1       93             70             31        0  30.4

     DiabetesPedigreeFunction  Age  Diagnosis Classification
0                       0.627   50          1       Diabetes
```

```
1                     0.351   31          0    No Diabetes
2                     0.672   32          1       Diabetes
3                     0.167   21          0    No Diabetes
4                     2.288   33          1       Diabetes
..                      ...  ...        ...            ...
763                   0.171   63          0    No Diabetes
764                   0.340   27          0    No Diabetes
765                   0.245   30          0    No Diabetes
766                   0.349   47          1       Diabetes
767                   0.315   23          0    No Diabetes

[768 rows x 10 columns]
```

```
# Use numpy to get the average age and glucose value.
import numpy as np

aveAge = np.mean(data['Age'])
aveGlucose = np.mean(data['Glucose'])

print("Average Age:", aveAge)
print("Average Glucose:", aveGlucose)
```

```
Average Age: 33.240885416666664
Average Glucose: 120.89453125
```

```
# Use numpy to get the median age and glucose value.
medAge = np.median(data['Age'])
medGlucose = np.median(data['Glucose'])

print("Median Age:", medAge)
print("Median Glucose:", medGlucose)
```

```
Median Age: 29.0
Median Glucose: 117.0
```

```
# Use numpy to get the middle values of glucose and age.
midGlucose = np.median(data['Glucose'])
midAge = np.median(data['Age'])

print("Middle Glucose:", midGlucose)
print("Middle Age:", midAge)
```

```
Middle Glucose: 117.0
Middle Age: 29.0
```

```
# Use numpy to get the standard deviation of the skinthickness
stdSkinThickness = np.std(data['SkinThickness'])

print("Standard Deviation of Skin Thickness:", stdSkinThickness)
```

```
Standard Deviation of Skin Thickness: 15.941828626496939
```

Conclusion

This lesson shows us how to use data analysis tools like Pandas and Numpy. It helps us do the job. We learn to handle large sets of data easily with these tools. This makes analyzing and understanding data much simpler.