exercises

August 5, 2023

1 Coding exercises Submitted By Aditya Kumar (UI22CS03)

1.1 Exercise 1: Generate the data by running this cell

This will give you a list of numbers to work with in the remaining exercises.

[135]:
```
import random
import statistics
random.seed(0)
salaries = [round(random.random()*1000000, -3) for _ in range(100)]
print(salaries)
```

```
[844000.0, 758000.0, 421000.0, 259000.0, 511000.0, 405000.0, 784000.0, 303000.0,
477000.0, 583000.0, 908000.0, 505000.0, 282000.0, 756000.0, 618000.0, 251000.0,
910000.0, 983000.0, 810000.0, 902000.0, 310000.0, 730000.0, 899000.0, 684000.0,
472000.0, 101000.0, 434000.0, 611000.0, 913000.0, 967000.0, 477000.0, 865000.0,
260000.0, 805000.0, 549000.0, 14000.0, 720000.0, 399000.0, 825000.0, 668000.0,
1000.0, 494000.0, 868000.0, 244000.0, 325000.0, 870000.0, 191000.0, 568000.0,
239000.0, 968000.0, 803000.0, 448000.0, 80000.0, 320000.0, 508000.0, 933000.0,
109000.0, 551000.0, 707000.0, 547000.0, 814000.0, 540000.0, 964000.0, 603000.0,
588000.0, 445000.0, 596000.0, 385000.0, 576000.0, 290000.0, 189000.0, 187000.0,
613000.0, 657000.0, 477000.0, 90000.0, 758000.0, 877000.0, 923000.0, 842000.0,
898000.0, 923000.0, 541000.0, 391000.0, 705000.0, 276000.0, 812000.0, 849000.0,
895000.0, 590000.0, 950000.0, 580000.0, 451000.0, 660000.0, 996000.0, 917000.0,
793000.0, 82000.0, 613000.0, 486000.0]
```

1.2 Exercise 2: Calculating statistics and verifying

1.2.1 mean

[136]:
```
#Method 1 : Finding Mean by using formula
sum=0
for i in salaries:
    sum=sum+i
mean_formula = sum/len(salaries)
print('The Mean of Salaries by formula=',mean_formula)
```

```
The Mean of Salaries by formula= 585690.0
```

[137]:
```
#Method 2 : Finding Mean using Statistics inbuild library of Python
mean = statistics.mean(salaries)
print('The Mean of Salaries=',mean)
```

```
The Mean of Salaries= 585690.0
```

1.2.2 median

[138]:
```
#Method 1 : FInding Median by Formula and defination
salaries.sort()
length = len(salaries)
if length%2 ==0:
    md1 = length/2
    md2 = length/2 + 1
    median_sorted = (salaries[int(md1)]+ salaries[int(md2)])/2
else:
    md1 = length/2
    median_sorted = (salaries[int(md1)])

print('The Median of Salaries by formula =',median_sorted)
```

```
The Median of Salaries by formula = 593000.0
```

[139]:
```
#Calculation of Median using Statistics library
median = statistics.median(salaries)
print('The Median of Salaries =',median)
```

```
The Median of Salaries = 589000.0
```

1.2.3 mode

[140]:
```
#Method 1 : Calculation of MODE using Formula
countt =0
mode_formula=salaries[0]
for all in salaries:
    if salaries.count(all)>countt:
        countt=salaries.count(all)
        mode_formula= all

print('The Mode of Salaries=',mode_formula)
```

```
The Mode of Salaries= 477000.0
```

[141]:
```
#Method 2: Calculation of MODE using Statistics library
mode = statistics.mode(salaries)
print('The Mode of Salaries=',mode)
```

```
The Mode of Salaries= 477000.0
```

1.2.4 sample variance

Remember to use Bessel's correction.

[142]:
```
#Method 1:Calculation of Sample Variance using Formula
sample_var=0
for all in salaries:
    sample_var= sample_var+ ((all-mean)**2)/(len(salaries)-1)

print('The Sample Variance of Salaries by formula =',sample_var)
```

```
The Sample Variance of Salaries by formula = 70664054444.44446
```

[143]:
```
#Method 2:Calculation of Sample Variance using Statistics library
varience = statistics.variance(salaries,mean)
print('The Sample Variance of Salaries=',varience)
```

```
The Sample Variance of Salaries= 70664054444.44444
```

1.2.5 sample standard deviation

Remember to use Bessel's correction.

[144]:
```
#Method 1:Calculation of Standard Deviation using Formula
import statistics
stdev_calculated=(varience**(0.5))
print('The Sample standard deviation of Salaries=',stdev_calculated)
```

```
The Sample standard deviation of Salaries= 265827.11382484
```

[145]:
```
#Method 2:Calculation of Standard Deviation using Statistics library
stdev = statistics.stdev(salaries)
print('The Sample standard deviation of Salaries=',stdev)
```

```
The Sample standard deviation of Salaries= 265827.11382484
```

1.3 Exercise 3: Calculating more statistics

1.3.1 range

[146]:
```
#Calculation of Range using simple inbuild min & max syntax
maximum=max(salaries)
minimum=min(salaries)
ranges = maximum-minimum
print("The Range of it is",ranges)
```

```
The Range of it is 995000.0
```

1.3.2 coefficient of variation

Make sure to use the sample standard deviation.

[147]:
```
#Calculation of coefficient of variation using simple mean, stdev relation␣
 ↪formula
import statistics
std = statistics.stdev(salaries)
mean = statistics.mean(salaries)
coeff_variation = std/mean
print('The Coefficient of variation of Salaries=',coeff_variation)
```

```
The Coefficient of variation of Salaries= 0.45386998894439035
```

1.3.3 interquartile range

[148]:
```
#Calculation of Interquartile Range using Statistics library
sorted_salaries = sorted(salaries)

# Calculate the indices for Q1 and Q3
n = len(sorted_salaries)
q1_index = n // 4
q3_index = (3 * n) // 4

# Calculating Q1, Q3, and Interquartile Range
q1 = sorted_salaries[q1_index]
q3 = sorted_salaries[q3_index]
interquartilerange = q3 - q1

print("Q1:", q1)
print("Q3:", q3)
print("Interquartile Range (IQR):", interquartilerange)
```

```
Q1: 405000.0
Q3: 825000.0
Interquartile Range (IQR): 420000.0
```

1.3.4 quartile coefficent of dispersion

[149]:
```
#After finding Q1 and Q3 we can easily find its Quartile Coefficent of␣
 ↪Dispersion by below formaule
print("Quartile coefficent of dispersion= ", ((q3-q1)/(q3+q1)))
```

```
Quartile coefficent of dispersion=  0.34146341463414637
```

1.4 Exercise 4: Scaling data

1.4.1 min-max scaling

[150]:
```
import random
random.seed(0)

min_salary = min(salaries)
max_salary = max(salaries)
#Lets now scale it
scaled_salaries = [(salary - min_salary) / (max_salary - min_salary) for salary␣
 ↪in salaries]

print("Original Salaries:", salaries)
print("Scaled Salaries:", scaled_salaries)
```

```
Original Salaries: [1000.0, 14000.0, 80000.0, 82000.0, 90000.0, 101000.0,
109000.0, 187000.0, 189000.0, 191000.0, 239000.0, 244000.0, 251000.0, 259000.0,
260000.0, 276000.0, 282000.0, 290000.0, 303000.0, 310000.0, 320000.0, 325000.0,
385000.0, 391000.0, 399000.0, 405000.0, 421000.0, 434000.0, 445000.0, 448000.0,
451000.0, 472000.0, 477000.0, 477000.0, 477000.0, 486000.0, 494000.0, 505000.0,
508000.0, 511000.0, 540000.0, 541000.0, 547000.0, 549000.0, 551000.0, 568000.0,
576000.0, 580000.0, 583000.0, 588000.0, 590000.0, 596000.0, 603000.0, 611000.0,
613000.0, 613000.0, 618000.0, 657000.0, 660000.0, 668000.0, 684000.0, 705000.0,
707000.0, 720000.0, 730000.0, 756000.0, 758000.0, 758000.0, 784000.0, 793000.0,
803000.0, 805000.0, 810000.0, 812000.0, 814000.0, 825000.0, 842000.0, 844000.0,
849000.0, 865000.0, 868000.0, 870000.0, 877000.0, 895000.0, 898000.0, 899000.0,
902000.0, 908000.0, 910000.0, 913000.0, 917000.0, 923000.0, 923000.0, 933000.0,
950000.0, 964000.0, 967000.0, 968000.0, 983000.0, 996000.0]
Scaled Salaries: [0.0, 0.01306532663316583, 0.07939698492462312,
0.0814070351758794, 0.08944723618090453, 0.10050251256281408,
0.10854271356783919, 0.18693467336683417, 0.18894472361809045,
0.19095477386934673, 0.23919597989949748, 0.2442211055276382,
0.25125628140703515, 0.2592964824120603, 0.26030150753768844,
0.27638190954773867, 0.28241206030150756, 0.2904522613065327,
0.3035175879396985, 0.31055276381909547, 0.32060301507537686,
0.3256281407035176, 0.385929648241206, 0.39195979899497485, 0.4,
0.40603015075376886, 0.4221105527638191, 0.43517587939698493,
0.4462311557788945, 0.4492462311557789, 0.45226130653266333, 0.4733668341708543,
0.47839195979899496, 0.47839195979899496, 0.47839195979899496,
0.48743718592964824, 0.49547738693467336, 0.5065326633165829,
0.5095477386934674, 0.5125628140703518, 0.5417085427135678, 0.542713567839196,
0.5487437185929648, 0.5507537688442211, 0.5527638190954773, 0.5698492462311557,
0.5778894472361809, 0.5819095477386935, 0.5849246231155779, 0.5899497487437186,
0.5919597989949749, 0.5979899497487438, 0.6050251256281407, 0.6130653266331658,
0.6150753768844222, 0.6150753768844222, 0.6201005025125628, 0.6592964824120603,
0.6623115577889447, 0.6703517587939698, 0.6864321608040201, 0.7075376884422111,
0.7095477386934673, 0.7226130653266332, 0.7326633165829146, 0.7587939698492462,
```

```
0.7608040201005025, 0.7608040201005025, 0.7869346733668342, 0.7959798994974875,
0.8060301507537688, 0.8080402010050252, 0.8130653266331658, 0.8150753768844221,
0.8170854271356784, 0.828140703517588, 0.8452261306532663, 0.8472361809045226,
0.8522613065326633, 0.8683417085427135, 0.871356783919598, 0.8733668341708543,
0.8804020100502512, 0.8984924623115578, 0.9015075376884422, 0.9025125628140703,
0.9055276381909547, 0.9115577889447236, 0.91356783919598, 0.9165829145728643,
0.9206030150753769, 0.9266331658291457, 0.9266331658291457, 0.9366834170854271,
0.9537688442211055, 0.9678391959798995, 0.9708542713567839, 0.9718592964824121,
0.9869346733668342, 1.0]
```

1.4.2 standardizing

[151]:
```
#Doing standardizing of Salaries data by formula
mean= statistics.mean(salaries)
stdev = statistics.stdev(salaries)

# Perform standardization on the salaries
standardized_salaries = [(salary-mean) / stdev for salary in salaries]

print("Original Salaries:", salaries)
print("Standardized Salaries:", standardized_salaries)
```

```
Original Salaries: [1000.0, 14000.0, 80000.0, 82000.0, 90000.0, 101000.0,
109000.0, 187000.0, 189000.0, 191000.0, 239000.0, 244000.0, 251000.0, 259000.0,
260000.0, 276000.0, 282000.0, 290000.0, 303000.0, 310000.0, 320000.0, 325000.0,
385000.0, 391000.0, 399000.0, 405000.0, 421000.0, 434000.0, 445000.0, 448000.0,
451000.0, 472000.0, 477000.0, 477000.0, 477000.0, 486000.0, 494000.0, 505000.0,
508000.0, 511000.0, 540000.0, 541000.0, 547000.0, 549000.0, 551000.0, 568000.0,
576000.0, 580000.0, 583000.0, 588000.0, 590000.0, 596000.0, 603000.0, 611000.0,
613000.0, 613000.0, 618000.0, 657000.0, 660000.0, 668000.0, 684000.0, 705000.0,
707000.0, 720000.0, 730000.0, 756000.0, 758000.0, 758000.0, 784000.0, 793000.0,
803000.0, 805000.0, 810000.0, 812000.0, 814000.0, 825000.0, 842000.0, 844000.0,
849000.0, 865000.0, 868000.0, 870000.0, 877000.0, 895000.0, 898000.0, 899000.0,
902000.0, 908000.0, 910000.0, 913000.0, 917000.0, 923000.0, 923000.0, 933000.0,
950000.0, 964000.0, 967000.0, 968000.0, 983000.0, 996000.0]
Standardized Salaries: [-2.199512275430514, -2.150608309943509,
-1.9023266390094862, -1.8948029520114855, -1.8647082040194827,
-1.8233279255304788, -1.7932331775384762, -1.4998093846164489,
-1.4922856976184482, -1.4847620106204475, -1.304193522668431,
-1.285384305173429, -1.2590514006804265, -1.228956652688424,
-1.2251948091894236, -1.165005313205418, -1.142434252211416, -1.112339504219413,
-1.0634355387324086, -1.037102634239406, -0.9994841992494026,
-0.9806749817544008, -0.7549643718143799, -0.7323933108203778,
-0.7022985628283751, -0.6797275018343729, -0.6195380058503674,
-0.5706340403633628, -0.529253761874359, -0.517968231377358,
-0.5066827008803569, -0.4276839874013496, -0.40887476990634786,
-0.40887476990634786, -0.40887476990634786, -0.37501817841534474,
-0.34492343042334195, -0.3035431519343381, -0.2922576214373371,
```

```
-0.28097209094033604, -0.17187862946932592, -0.16811678597032556,
-0.14554572497632348, -0.1380220379783228, -0.1304983509803221,
-0.06654701149731616, -0.03645226350531338, -0.02140488950931198,
-0.010119359012310937, 0.008689858482690806, 0.016213545480691503,
0.038784606474693596, 0.06511751096769604, 0.09521225895969881,
0.10273594595769951, 0.10273594595769951, 0.12154516345270126,
0.26825705991371485, 0.2795425904107159, 0.3096373384027187,
0.36982683438672426, 0.4488255478657316, 0.4563492348637323, 0.5052532003507368,
0.5428716353407403, 0.6406795663147493, 0.6482032533127501, 0.6482032533127501,
0.7460111842867592, 0.7798677757777622, 0.8174862107677657, 0.8250098977657664,
0.8438191152607681, 0.8513428022587689, 0.8588664892567696, 0.9002467677457734,
0.9641981072287793, 0.9717217942267801, 0.9905310117217818, 1.0507205077057873,
1.0620060382027885, 1.0695297252007891, 1.0958626296937914, 1.1635758126757978,
1.1748613431727988, 1.178623186671799, 1.1899087171688003, 1.2124797781628023,
1.220003465160803, 1.2312889956578041, 1.2463363696538055, 1.2689074306478076,
1.2689074306478076, 1.3065258656378111, 1.370477205120817, 1.4231430141068218,
1.434428544603823, 1.4381903881028233, 1.4946180405878284, 1.543522006074833]
```

1.5 Exercise 5: Calculating covariance and correlation

1.5.1 covariance

[152]:
```
#Calculation of Covariance using Statistics library
import random
import statistics
random.seed(0)
#First I have decleared Salaries1 using random -3 function
salaries1 = [round(random.random()*1000000, -3) for _ in range(100)]
#Then I have decleared Salaries2 using random -6 function
salaries2 = [round(random.random()*1000000, -6) for _ in range(100)]
#Covarince is find btw two variables so here we go
covariance=statistics.covariance(salaries1,salaries2)

print('The covariance of two Random Generate Salaries are =',covariance)
```

```
The covariance of two Random Generate Salaries are = -8983434343.434343
```

1.5.2 Pearson correlation coefficient (ρ)

[153]:
```
#Calculation of Person's correlation Coefficient using Staatistics library
perasonscorellation = statistics.correlation(salaries1,salaries2)
print('The Pearson correlation coefficient of two Random Generate Samples is␣
 ↪=',perasonscorellation)
```

```
The Pearson correlation coefficient of two Random Generate Samples is =
-0.0677392474582281
```

[154]:
```
#Assignmnet 1 subimiited by Aditya Kumar (UI22CS03)
```