

Descriptive Statistics - Measures of Central Tendency and variability Perform the following operations on any open source dataset (e.g., data.csv)

- 1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.

```
import pandas as pd
import numpy as np

df = pd.read_csv("hr.csv")
df
```

🔗

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Educatic
0	41	Yes	Travel_Rarely	1102	Sales		1
1	49	No	Travel_Frequently	279	Research & Development		8
2	37	Yes	Travel_Rarely	1373	Research & Development		2
3	33	No	Travel_Frequently	1392	Research & Development		3
4	27	No	Travel_Rarely	591	Research & Development		2
...	...	...	...	...	...		...
			Travel_Frequently	884	Research & Development		23
1466	39	No	Travel_Rarely	613	Research & Development		6
1467	27	No	Travel_Rarely	155	Research & Development		4
1468	49	No	Travel_Frequently	1023	Sales		2
1469	34	No	Travel_Rarely	628	Research & Development		8

1470 rows × 35 columns

```
df.columns

Index(['Age', 'Attrition', 'BusinessTravel', 'DailyRate', 'Department',
      'DistanceFromHome', 'Education', 'EducationField', 'EmployeeCount',
      'EmployeeNumber', 'EnvironmentSatisfaction', 'Gender', 'HourlyRate',
      'JobInvolvement', 'JobLevel', 'JobRole', 'JobSatisfaction',
      'MaritalStatus', 'MonthlyIncome', 'MonthlyRate', 'NumCompaniesWorked',
      'Over18', 'OverTime', 'PercentSalaryHike', 'PerformanceRating',
      'RelationshipSatisfaction', 'StandardHours', 'StockOptionLevel',
      'TotalWorkingYears', 'TrainingTimesLastYear', 'WorkLifeBalance',
      'YearsAtCompany', 'YearsInCurrentRole', 'YearsSinceLastPromotion',
      'YearsWithCurrManager'],
      dtype='object')

df['MonthlyIncome'].mean()

6502.931292517007

df['Age'].mean()

36.923809523809524

df.loc[:, 'MonthlyIncome'].mean()

6502.931292517007
```

```

df.loc[:, 'Age'].mean()

36.923809523809524

df['MonthlyIncome'].median()

4919.0

df['MonthlyIncome'].mode()

0    2342
Name: MonthlyIncome, dtype: int64

df['Age'].median()

36.0

df['Age'].mode()

0    35
Name: Age, dtype: int64

array1 = np.array(df['MonthlyIncome'])
array2 = np.array(df['Age'])

print("monthly income", array1)

monthly income [5993 5130 2090 ... 6142 5390 4404]

print("Age", array2)



Saved successfully! ×


print("Maximum in monthly income", array1.max())

Maximum in monthly income 19999

print("Maximum in monthly income", max(array1))

Maximum in monthly income 19999

print("Maximum in age : ", array2.max())

Maximum in age : 60

print("Maximum in monthly income", array1.max())
print("Maximum in age : ", array2.max())
print("Minimum in Monthly income : ", array1.min())
print("Minimum in Age : ", array2.min())

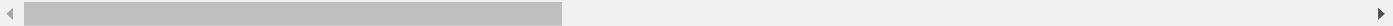
Maximum in monthly income 19999
Maximum in age : 60
Minimum in Monthly income : 1009
Minimum in Age : 18

df['BusinessTravel'].replace({'Travel_Rarely':0, 'Travel_Frequently':1}, inplace=True)
df['Department'].replace({'Sales':0, 'Research & Development':1}, inplace=True)
df['Attrition'].replace({'Yes':1, 'No':0}, inplace=True)
df['EducationField'].replace({'Life Sciences':0, 'Medical':1, 'Other':2}, inplace=True)
df

```

	Age	Attrition	BusinessTravel	DailyRate	Department	DistanceFromHome	Education	EducationField	EmployeeCount	EmployeeNumber
0	41	1	0	1102	0	1	2	0	1	1
1	49	0	1	279	1	8	1	0	1	2
2	37	1	0	1373	1	2	2	2	1	4
3	33	0	1	1392	1	3	4	0	1	5
4	27	0	0	591	1	2	1	0	1	7
...	...	...	...	...	...	...	...	...	...	...
1465	36	0	1	884	1	23	2	0	1	2061
1467	27	0	0	155	1	4	3	0	1	2064
1468	49	0	1	1023	0	2	3	0	1	2065
1469	34	0	0	628	1	8	3	0	1	2068

1470 rows × 35 columns



Saved successfully! ×

[Colab paid products](#) - [Cancel contracts here](#)

