Basic Statistics - Measures of Central Tendencies and Variance

Perform the following operations on any open source dataset (eg. data.csv)

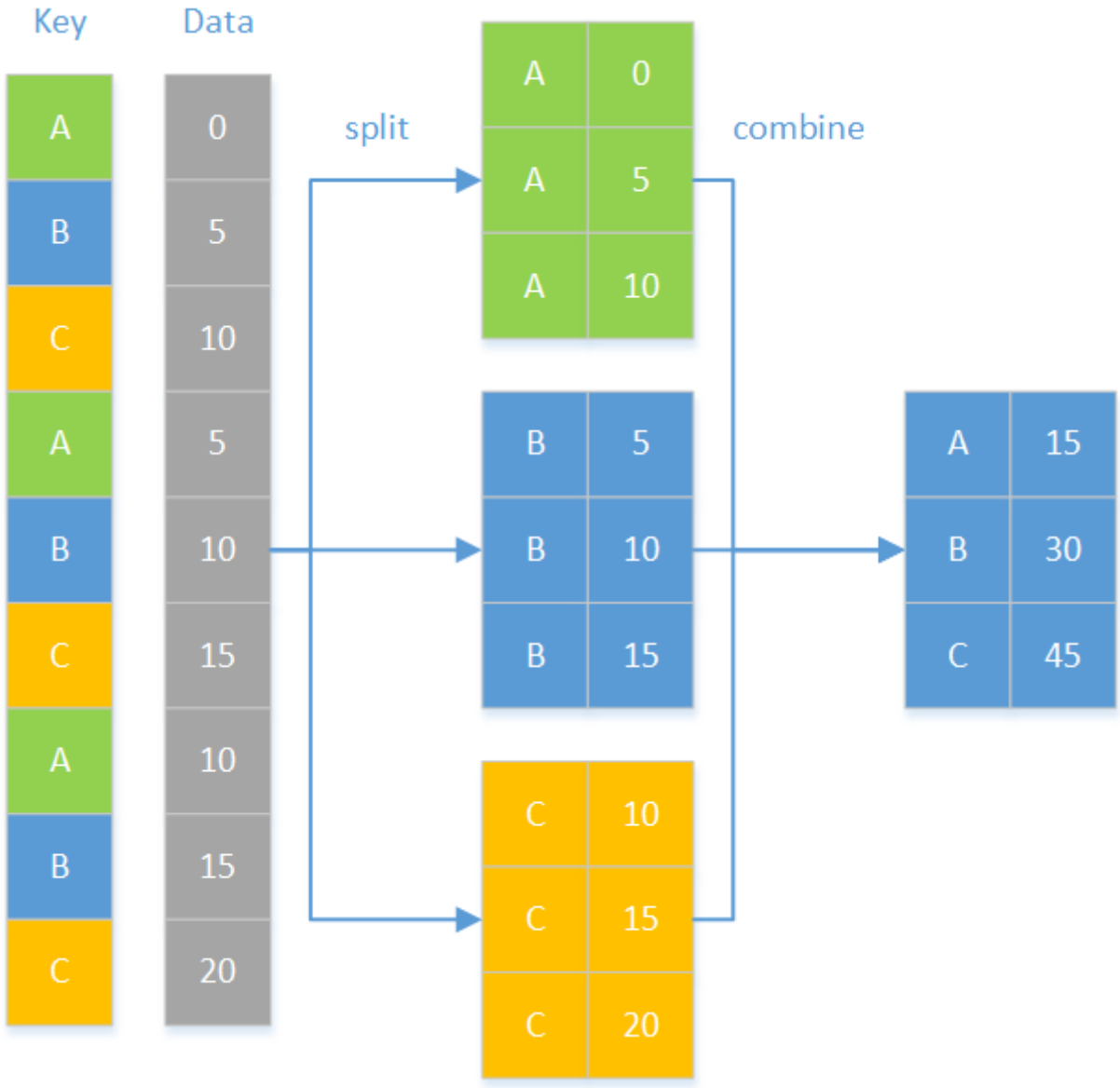
1. Provide summary statistics (mean, median, minimum, maximum, standard deviation) for a dataset (age, income etc.) with numeric variables grouped by one of the qualitative (categorical) variable. For example, if your categorical variable is age groups and quantitative variable is income, then provide summary statistics of income grouped by the age groups. Create a list that contains a numeric value for each response to the categorical variable.
    A. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of ‘Iris-setosa’, ‘Iris-versicolor’ and ‘Iris-versicolor’ of iris.csv dataset. Provide the codes with outputs and explain everything that you do in this step.

By,

Vinayak Jalan

TE B 74

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. ¶


Key
Data
A
B
C
A
B
C
A
B
C
0
5
10
5
10
15
10
15
20
split
A 0
A 5
A 10
B 5
B 10
B 15
C 10
C 15
C 20
combine
A 15
B 30
C 45

In [150]:

```
import pandas as pd
import numpy as np
student = pd.read_csv("/content/StudentsPerformance.csv")
```

In [151]:

```
student.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   gender                       1000 non-null   object 
 1   race/ethnicity               1000 non-null   object 
 2   parental level of education  1000 non-null   object 
 3   lunch                        1000 non-null   object 
 4   test_preparation_course      1000 non-null   object 
 5   math_score                   991 non-null    float64
 6   reading_score                995 non-null    float64
 7   writing_score                994 non-null    float64
dtypes: float64(3), object(5)
memory usage: 62.6+ KB
```

In [152]:

```
student.describe()
```

Out[152]:

	math_score	reading_score	writing_score
count	991.000000	995.000000	994.000000
mean	66.116044	69.223116	68.113682
std	15.217867	14.577775	15.182945
min	0.000000	17.000000	10.000000
25%	57.000000	59.000000	58.000000
50%	66.000000	70.000000	69.000000
75%	77.000000	79.000000	79.000000
max	100.000000	100.000000	100.000000

In [153]:

```
student.head()
```

Out[153]:

	gender	race/ethnicity	parental level of education	lunch	test_preparation_course	math_score	readi
0	female	group B	bachelor's degree	standard	none	72.0	
1	female	group C	some college	standard	completed	69.0	
2	female	group B	master's degree	standard	none	90.0	
3	male	group A	associate's degree	free/reduced	none	47.0	
4	male	group C	some college	standard	none	76.0	

In [154]:

```
male_female = student.groupby('gender')['gender'].count()
print(male_female)
```

```
gender
female    518
male      482
Name: gender, dtype: int64
```

In [155]:

```
student.test_preparation_course.unique()
```

Out[155]:

```
array(['none', 'completed'], dtype=object)
```

In [156]:

```
mean_math = student.groupby('gender').math_score.mean()
```

In [157]:

```
print(mean_math)
```

```
gender
female    63.654902
male      68.725572
Name: math_score, dtype: float64
```

In [158]:

```
mean_math_test_preparation = student.groupby(['gender','test_preparation_course']).math
_score.mean()
print(mean_math_test_preparation)
```

```
gender  test_preparation_course
female  completed                 67.331492
        none                      61.632219
male    completed                 72.339080
        none                      66.677524
Name: math_score, dtype: float64
```

In [159]:

```
student.math_score.unique()
```

Out[159]:

```
array([ 72.,  69.,  90.,  47.,  76.,  71.,  88.,  40.,  64.,  38.,  58.,
        nan,  78.,  50.,  18.,  46.,  54.,  66.,  65.,  44.,  74.,  73.,
        70.,  62.,  63.,  56.,  97.,  81.,  75.,  57.,  55.,  53.,  59.,
        82.,  77.,  33.,  52.,   0.,  79.,  39.,  67.,  45.,  60.,  61.,
        41.,  49.,  30.,  80.,  42.,  27.,  43.,  68.,  85.,  98.,  87.,
        51.,  99.,  84.,  91.,  83.,  89.,  22., 100.,  96.,  94.,  48.,
        35.,  34.,  86.,  92.,  37.,  28.,  24.,  26.,  95.,  36.,  29.,
        32.,  93.,  19.,  23.,   8.])
```

Group by of a Single Column and Apply the describe() Method on a Single Column

In [160]:

```
print(student.groupby('gender').math_score.describe())
```

```
        count        mean        std   min   25%   50%   75%    max
gender
female  510.0   63.654902  15.593640   0.0  54.0  65.0  74.0  100.0
male    481.0   68.725572  14.371106  27.0  59.0  69.0  79.0  100.0
```

In [161]:

```
groups = pd.cut(student['math_score'],bins=4)
groups
```

Out[161]:

```
0       (50.0, 75.0]
1       (50.0, 75.0]
2      (75.0, 100.0]
3       (25.0, 50.0]
4      (75.0, 100.0]
           ...
995    (75.0, 100.0]
996     (50.0, 75.0]
997     (50.0, 75.0]
998     (50.0, 75.0]
999    (75.0, 100.0]
Name: math_score, Length: 1000, dtype: category
Categories (4, interval[float64, right]): [(-0.1, 25.0] < (25.0, 50.0] <
(50.0, 75.0] <
                                            (75.0, 100.0]]
```

In [162]:

```
student.groupby(groups)['math_score'].count()
```

Out[162]:

```
math_score
(-0.1, 25.0]       7
(25.0, 50.0]     143
(50.0, 75.0]     567
(75.0, 100.0]    274
Name: math_score, dtype: int64
```

In [163]:

```
pd.crosstab(groups, student['gender'])
```

Out[163]:

gender	female	male
math_score		
(-0.1, 25.0]	7	0
(25.0, 50.0]	90	53
(50.0, 75.0]	301	266
(75.0, 100.0]	112	162

2. Write a Python program to display some basic statistical details like percentile, mean, standard deviation etc. of the species of ‘Iris-setosa’, ‘Iris-versicolor’ and 'Iris-versicolor’ of iris.csv dataset.

Python Descriptive Statistics – Measuring Central Tendency

In [164]:

```
import statistics as st
```

In [165]:

```
data = [1,2,3,4,5,6]
```

In [166]:

```
st.mean(data)
```

Out[166]:

```
3.5
```

In [167]:

```
st.median(data)
```

Out[167]:

```
3.5
```

In [187]:

```
#Will show error as data is having no unique modal value
st.mode(data)
```

```
---------------------------------------------------------------------------
-
StatisticsError                           Traceback (most recent call las
t)
<ipython-input-187-7adf61ce2b58> in <module>()
      1 #Will show error as data is having no unique modal value
----> 2 st.mode(data)

/usr/lib/python3.7/statistics.py in mode(data)
    504     elif table:
    505         raise StatisticsError(
--> 506                 'no unique mode; found %d equally common values' %
len(table)
    507                 )
    508     else:

StatisticsError: no unique mode; found 5 equally common values
```

In [169]:

```
data1 = [1,2,7,5,4,7,8,2,1,7]
st.mode(data1)
```

Out[169]:

```
7
```

In [170]:

```
#Variance
st.variance(data1)
```

Out[170]:

7.6

In [171]:

```
#Variance
st.variance(data1)
```

Out[171]:

7.6

In [172]:

```
import pandas as pd
df = pd.DataFrame(data1)
```

In [173]:

```
df.mean()
```

Out[173]:

```
0    4.4
dtype: float64
```

In [174]:

```
df.mode()
```

Out[174]:

	0
0	7

In [175]:

```
df.median()
```

Out[175]:

```
0    4.5
dtype: float64
```

In [176]:

```
#using California housing train csv file
df1 = pd.read_csv("/content/sample_data/california_housing_train.csv")
df1
```

Out[176]:

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	ho
0	-114.31	34.19	15.0	5612.0	1283.0	1015.0	
1	-114.47	34.40	19.0	7650.0	1901.0	1129.0	
2	-114.56	33.69	17.0	720.0	174.0	333.0	
3	-114.57	33.64	14.0	1501.0	337.0	515.0	
4	-114.57	33.57	20.0	1454.0	326.0	624.0	
...	...	...	...	...	...	...	
16995	-124.26	40.58	52.0	2217.0	394.0	907.0	
16996	-124.27	40.69	36.0	2349.0	528.0	1194.0	
16997	-124.30	41.84	17.0	2677.0	531.0	1244.0	
16998	-124.30	41.80	19.0	2672.0	552.0	1298.0	
16999	-124.35	40.54	52.0	1820.0	300.0	806.0	

17000 rows × 9 columns

In [177]:

```
df1.mean()
```

Out[177]:

```
longitude               -119.562108
latitude                  35.625225
housing_median_age        28.589353
total_rooms             2643.664412
total_bedrooms           539.410824
population              1429.573941
households               501.221941
median_income              3.883578
median_house_value    207300.912353
dtype: float64
```

In [178]:

```
df1["households"].mean()
```

Out[178]:

```
501.2219411764706
```

In [179]:

```
df1["households"].median()
```

Out[179]:

```
409.0
```

In [180]:

```
df1["households"].mode()
```

Out[180]:

```
0    306.0
1    386.0
dtype: float64
```

In [181]:

```
df1["households"].var()
```

Out[181]:

```
147856.2770525285
```

In [182]:

```
st.stdev(df1["households"])
```

Out[182]:

```
384.5208408559009
```

Descriptive Statistics on IRIS dataset

In [183]:

```
import pandas as pd
data = pd.read_csv("iris.csv")
print('Iris-setosa')
```

```
Iris-setosa
```

In [184]:

```
setosa = data['species'] == 'Iris-setosa'
print(data[setosa].describe())
```

```
       sepal_length  sepal_width  petal_length  petal_width
count           0.0          0.0           0.0          0.0
mean            NaN          NaN           NaN          NaN
std             NaN          NaN           NaN          NaN
min             NaN          NaN           NaN          NaN
25%             NaN          NaN           NaN          NaN
50%             NaN          NaN           NaN          NaN
75%             NaN          NaN           NaN          NaN
max             NaN          NaN           NaN          NaN
```

In [185]:

```
print('\nIris-versicolor')
setosa = data['species'] == 'Iris-versicolor'
print(data[setosa].describe())
```

```
Iris-versicolor
       sepal_length  sepal_width  petal_length  petal_width
count           0.0          0.0           0.0          0.0
mean            NaN          NaN           NaN          NaN
std             NaN          NaN           NaN          NaN
min             NaN          NaN           NaN          NaN
25%             NaN          NaN           NaN          NaN
50%             NaN          NaN           NaN          NaN
75%             NaN          NaN           NaN          NaN
max             NaN          NaN           NaN          NaN
```

In [186]:

```
print('\nIris-virginica')
setosa = data['species'] == 'Iris-virginica'
print(data[setosa].describe())
```

```
Iris-virginica
       sepal_length  sepal_width  petal_length  petal_width
count           0.0          0.0           0.0          0.0
mean            NaN          NaN           NaN          NaN
std             NaN          NaN           NaN          NaN
min             NaN          NaN           NaN          NaN
25%             NaN          NaN           NaN          NaN
50%             NaN          NaN           NaN          NaN
75%             NaN          NaN           NaN          NaN
max             NaN          NaN           NaN          NaN
```