math-eight.blogspot.com

Descriptive Statistics and Linear Regression Using 'statistics' module and 'statsmodels' module

12–16 minutes

In [2]:

```
DataSet = [13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25,
30, 33, 33, 35, 35, 35, 36, 40, 45, 46, 52, 70]

# Sum of all elements using simple built in sum function

print("Sum of all items of Data Set : " + str(sum(DataSet)))

# Getting Count of each items using counter collection

"""
Counter is an unordered collection where elements are stored
as Dict keys and their count as dict value

"""


from collections import Counter
print("Count of each items in Data Set : ")
print(Counter(DataSet))

# Use of statistics module
```

```
import statistics as st

# Mean -> Sum of all data items / total no of data items

print("Mean of Data Set : ")
print(st.mean(DataSet))

# Median -> Average of two items exist in mid of data set

print("Median of Data Set : ")
print(st.median(DataSet))

# Mode -> Item with highest frequency of appearance

print("Mode of Data Set : ")
print(st.mode(DataSet))

# Mid-range -> Average of MaxVale And MinValue item

print("Mid Range Value Of Data Set : ")
print(st.mean([max(DataSet), min(DataSet)]))

# Other Useful statistical measures

print("Quantiles Of Data Set : ")
print(st.quantiles(data = DataSet, n = 4)) # [20.0, 25.0, 35.25]
print("Std. Deviation Of Data Set : ")
print(st.stdev(DataSet))
print("Variance Of Data Set : ")
print(st.variance(DataSet))
```

Sum of all items of Data Set : 774

Count of each items in Data Set :

Counter({25: 4, 35: 3, 16: 2, 20: 2, 22: 2, 33: 2, 13: 1, 15: 1, 19: 1, 21: 1, 30: 1, 36: 1, 40: 1, 45: 1, 46: 1, 52: 1, 70: 1})

Mean of Data Set :

29.76923076923077

Median of Data Set :

25.0

Mode of Data Set :

25

Mid Range Value Of Data Set :

41.5

Quantiles Of Data Set :

[20.0, 25.0, 35.25]

Std. Deviation Of Data Set :

13.158442741624686

Variance Of Data Set :

173.14461538461538

In [4]:

```
df = pd.read_csv('HeightWeight.csv')
```

Out[5]:

	Index	Height(Inches)	Weight(Pounds)
0	1	65.78331	112.9925
1	2	71.51521	136.4873
2	3	69.39874	153.0269
3	4	68.21660	142.3354
4	5	67.78781	144.2971

'statistics' is a core Python package. We can use it but now list it. !pip show statistics WARNING: Package(s) not found: statistics

In [8]:

st.correlation(df['Height(Inches)'], df['Weight(Pounds)'])

Linear Regression¶

In [6]:

New in version 3.10
    slope, intercept = st.linear_regression(df['Height(Inches)'], df['Weight(Pounds)'])

Out[7]:

(3.0834764454029657, -82.57574306454092)

Using 'statsmodels' module

Name: statsmodels
Version: 0.13.5
Summary: Statistical computations and models for Python
Home-page: https://www.statsmodels.org/
Author:
Author-email:
License: BSD License
Location: /home/ashish/anaconda3/envs/py310/lib/python3.10/site-packages
Requires: numpy, packaging, pandas, patsy, scipy
Required-by:

In [9]:

Calculating various statistics value for a data set using statmodels , sciPy , numpy and pandas module functions
importing required modules

```
from statsmodels import stats
import statsmodels.stats.weightstats as ws
import statsmodels.stats.descriptivestats as ds
import statsmodels.stats.libqsturng as lq
```

In [10]:

```
DataSet = [13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 36, 40, 45, 46, 52, 70]
```

In [11]:

```
mean = ws.stats.gmean(DataSet)
print(mean)

# median = ws.stats.median(DataSet)
# AttributeError: module 'scipy.stats' has no attribute 'median'

desc_stats = ds.describe(DataSet)
print("desc_stats using statsmodels : ", desc_stats)
```

```
27.347117200207276
desc_stats using statsmodels :                         0
nobs              26.000000
missing            0.000000
mean              29.769231
std_err            2.580583
upper_ci          34.827080
lower_ci          24.711381
std               13.158443
iqr               14.750000
```

iqr_normal	10.934191
mad	10.213018
mad_normal	12.800120
coef_var	0.442015
range	57.000000
max	70.000000
min	13.000000
skew	1.206785
kurtosis	4.506284
jarque_bera	8.768727
jarque_bera_pval	0.012471
mode	25.000000
mode_freq	0.153846
median	25.000000
1%	13.500000
5%	15.250000
10%	16.000000
25%	20.250000
50%	25.000000
75%	35.000000
90%	45.500000
95%	50.500000
99%	65.500000

In [12]:

```
type(desc_stats) # pandas.core.frame.DataFrame
```

Out[12]:

```
pandas.core.frame.DataFrame
```

In [16]:

```
for i in ['mean', 'median', 'mode', 'std', '25%', '50%', '75%',
```

```
            'iqr', 'min', 'max']:
        print(i, desc_stats.loc[i][0])
```

```
mean 29.76923076923077
median 25.0
mode 25.0
std 13.158442741624686
25% 20.25
50% 25.0
75% 35.0
iqr 14.75
min 13.0
max 70.0
```

In [14]:

```
import pandas as pd
mean = pd.Series(DataSet).describe()
print("Mean using pandas :", mean)
```

```
Mean using pandas : count    26.000000
mean     29.769231
std      13.158443
min      13.000000
25%      20.250000
50%      25.000000
75%      35.000000
max      70.000000
dtype: float64
```

Linear Regression¶

In [19]:

```
import statsmodels.api as sm
```

```
import pandas as pd
```

In [20]:

```
df = pd.read_csv('HeightWeight.csv')
```

In [21]:

```
results = sm.OLS(df['Height(Inches)'], df['Weight(Pounds)']).fit()
```

Out[23]:

OLS Regression Results

Dep. Variable:	Height(Inches)	**R-squared (uncentered):**	0.993
Model:	OLS	**Adj. R-squared (uncentered):**	0.993
Method:	Least Squares	**F-statistic:**	3.783e+06
Date:	Mon, 15 May 2023	**Prob (F-statistic):**	0.00
Time:	13:09:01	**Log-Likelihood:**	-78144.
No. Observations:	25000	**AIC:**	1.563e+05
Df Residuals:	24999	**BIC:**	1.563e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std**	**t**	**P>|t|**	**[0.025**	**0.975]**

		err				
Weight(Pounds)	0.5313	0.000	1944.918	0.000	0.531	0.532

Omnibus:	3.114	**Durbin-Watson:**	1.974
Prob(Omnibus):	0.211	**Jarque-Bera (JB):**	3.091
Skew:	0.024	**Prob(JB):**	0.213
Kurtosis:	3.025	**Cond. No.**	1.00

Notes:

[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.

[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.