

# Problem Statement

## Linear Regression

### Import Libraries

```
In [1]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [6]: a=pd.read_csv("drug.csv")
a
```

Out[6]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY
...	...	...	...	...	...	...
195	56	F	LOW	HIGH	11.567	drugC
196	16	M	LOW	HIGH	12.006	drugC
197	52	M	NORMAL	HIGH	9.894	drugX
198	23	M	NORMAL	NORMAL	14.020	drugX
199	40	F	LOW	NORMAL	11.349	drugX

200 rows × 6 columns

### To display top 10 rows

```
In [7]: c=a.head(15)
c
```

Out[7]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
5	22	F	NORMAL	HIGH	8.607	drugX
6	49	F	NORMAL	HIGH	16.275	drugY
7	41	M	LOW	HIGH	11.037	drugC
8	60	M	NORMAL	HIGH	15.171	drugY
9	43	M	LOW	NORMAL	19.368	drugY
10	47	F	LOW	HIGH	11.767	drugC
11	34	F	HIGH	NORMAL	19.199	drugY
12	43	M	LOW	HIGH	15.376	drugY
13	74	F	LOW	HIGH	20.942	drugY
14	50	F	NORMAL	HIGH	12.703	drugX

## To find Missing values

In [8]:

c.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15 entries, 0 to 14
Data columns (total 6 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Age         15 non-null    int64
1   Sex         15 non-null    object
2   BP          15 non-null    object
3   Cholesterol 15 non-null    object
4   Na_to_K     15 non-null    float64
5   Drug        15 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 848.0+ bytes
```

## To display summary of statistics

In [9]:

a.describe()

Out[9]:

	Age	Na_to_K
count	200.000000	200.000000
mean	44.315000	16.084485
std	16.544315	7.223956
min	15.000000	6.269000
25%	31.000000	10.445500
50%	45.000000	13.936500
75%	58.000000	19.380000
max	74.000000	38.247000

# To display column heading

```
In [10]: a.columns
```

Out[10]: Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na\_to\_K', 'Drug'], dtype='object')

# Pairplot

```
In [11]: s=a.dropna(axis=1)
s
```

Out[11]:

	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	drugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	drugY
...	...	...	...	...	...	...
195	56	F	LOW	HIGH	11.567	drugC
196	16	M	LOW	HIGH	12.006	drugC
197	52	M	NORMAL	HIGH	9.894	drugX
198	23	M	NORMAL	NORMAL	14.020	drugX
199	40	F	LOW	NORMAL	11.349	drugX

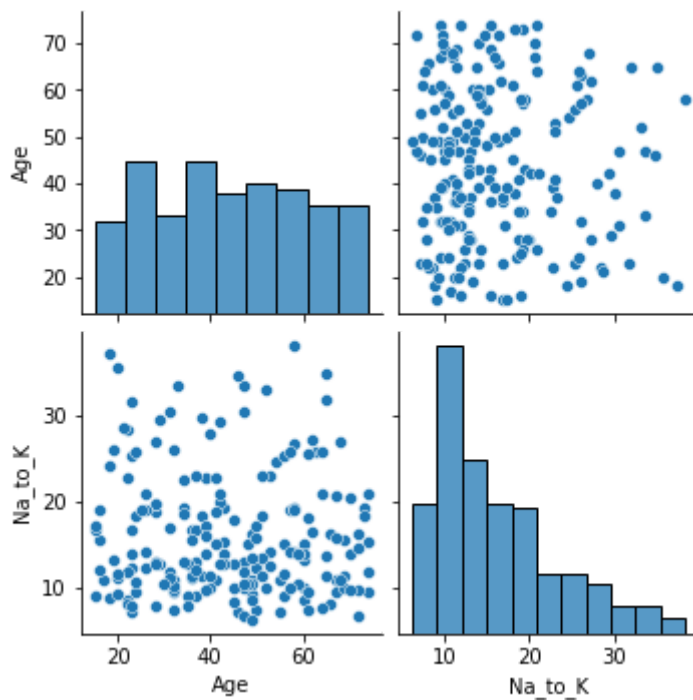
200 rows × 6 columns

```
In [12]: s.columns
```

Out[12]: Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na\_to\_K', 'Drug'], dtype='object')

```
In [13]: sns.pairplot(a)
```

Out[13]: <seaborn.axisgrid.PairGrid at 0x2515fb33fa0>

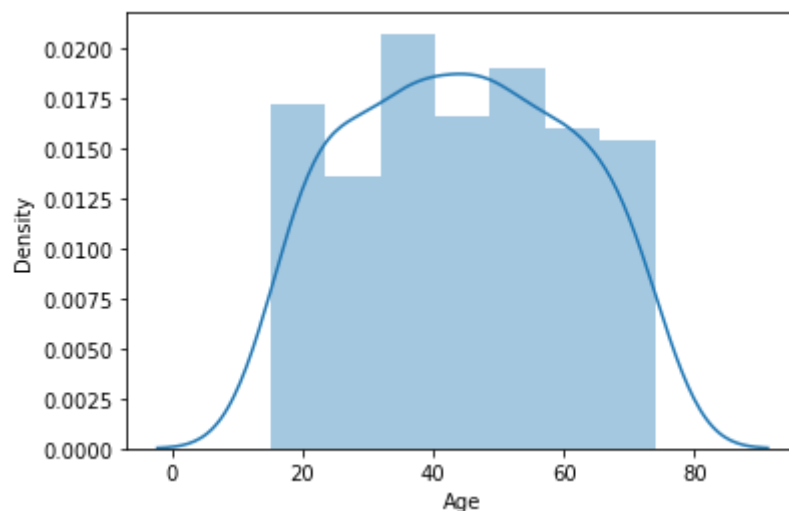


## Distribution Plot

```
In [14]: sns.distplot(a['Age'])
```

C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).  
warnings.warn(msg, FutureWarning)

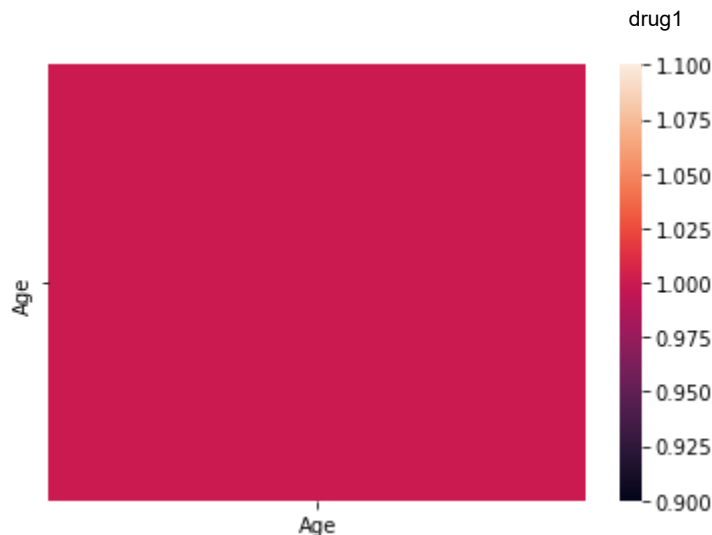
```
Out[14]: <AxesSubplot:xlabel='Age', ylabel='Density'>
```



## Correlation

```
In [15]: b=s[['Age']]
sns.heatmap(b.corr())
```

```
Out[15]: <AxesSubplot:>
```



## Train the model - Model Building

```
In [17]: g=s[['Age']]
         h=s['Age']
```

## To split dataset into training and test

```
In [18]: from sklearn.model_selection import train_test_split
         g_train,g_test,h_train,h_test=train_test_split(g,h,test_size=0.6)
```

## To run the model

```
In [19]: from sklearn.linear_model import LinearRegression
```

```
In [20]: lr=LinearRegression()
         lr.fit(g_train,h_train)
```

Out[20]: LinearRegression()

```
In [21]: print(lr.intercept_)
```

7.105427357601002e-15

## Coefficient

```
In [22]: coeff=pd.DataFrame(lr.coef_,g.columns,columns=['Co-effecient'])
         coeff
```

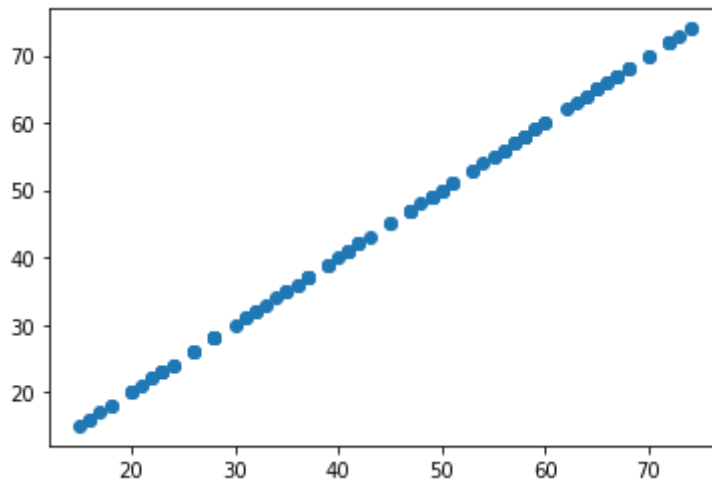
```
Out[22]:
```

	Co-effecient
Age	1.0

## Best Fit line

```
In [23]: prediction=lr.predict(g_test)
plt.scatter(h_test,prediction)
```

```
Out[23]: <matplotlib.collections.PathCollection at 0x25161e70070>
```



## To find score

```
In [24]: print(lr.score(g_test,h_test))
```

1.0

```
In [ ]:
```