In [1]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```python
df=pd.read_csv(r'C:\Users\user\Downloads\4_drug200.csv')
df
```

Out[2]:

	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 [3]:

```python
df.head(10)
```

Out[3]:

	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
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

In [4]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 6 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Age          200 non-null    int64
 1   Sex          200 non-null    object
 2   BP           200 non-null    object
 3   Cholesterol  200 non-null    object
 4   Na_to_K      200 non-null    float64
 5   Drug         200 non-null    object
dtypes: float64(1), int64(1), object(4)
memory usage: 9.5+ KB
```

In [5]:

```python
df.describe()
```

Out[5]:

	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

In [6]:
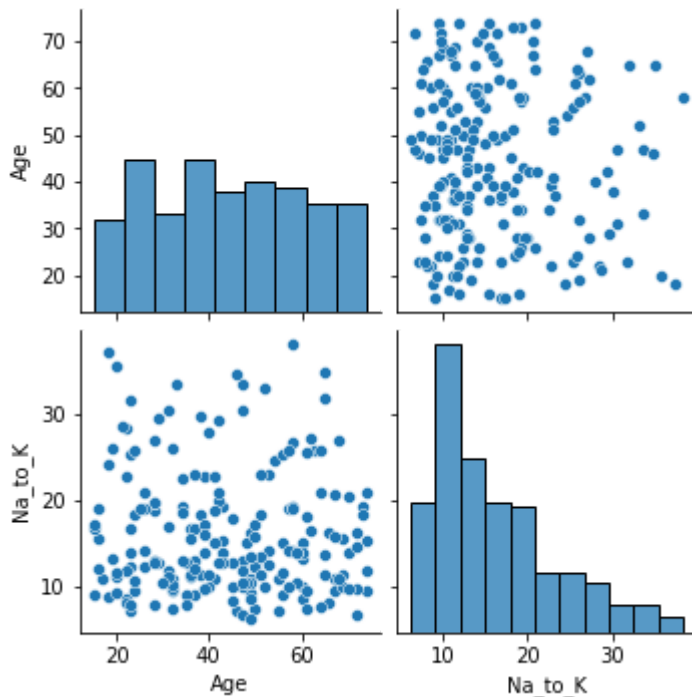
```python
df.columns
```

Out[6]:

```
Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug'], dtype='objec
t')
```

In [7]:

```python
sns.pairplot(df)
```

Out[7]:

```
<seaborn.axisgrid.PairGrid at 0x1c53a7f8fd0>
```
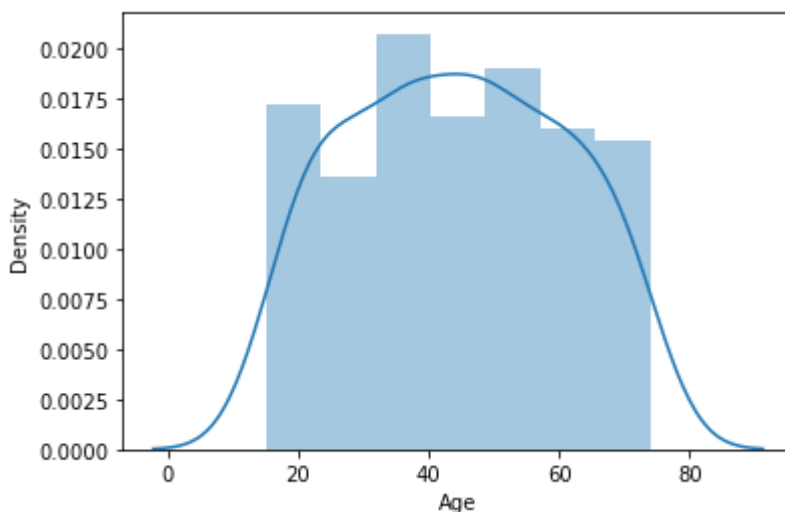

In [8]:

```python
sns.distplot(df['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 fun
ction for histograms).
  warnings.warn(msg, FutureWarning)
```
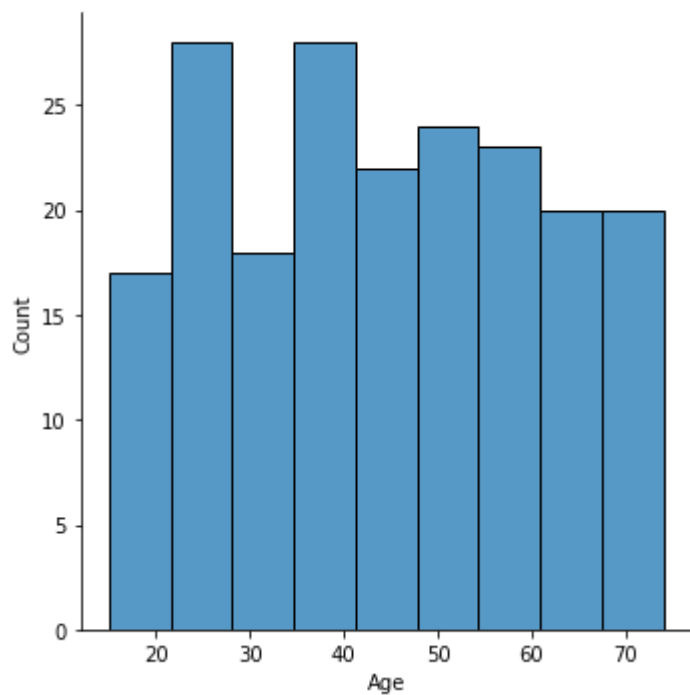
Out[8]:

```
<AxesSubplot:xlabel='Age', ylabel='Density'>
```

In [9]:

```python
sns.displot(df["Age"])
```

Out[9]:

`<seaborn.axisgrid.FacetGrid at 0x1c53afc88b0>`

In [10]:

```python
df1=df[['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug']]
```
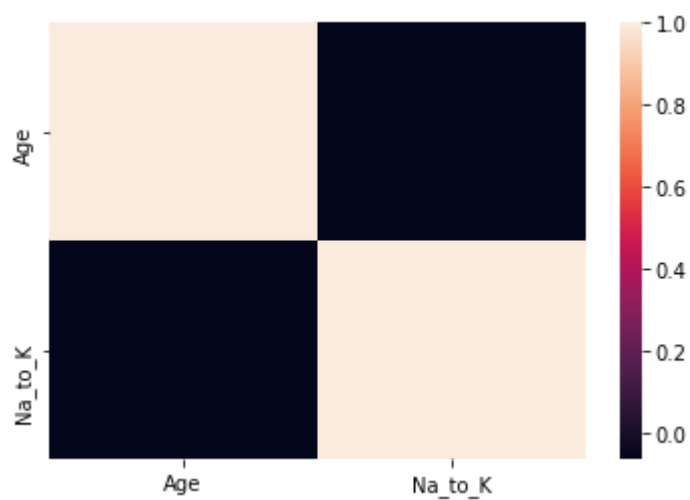
In [11]:

```python
sns.heatmap(df1.corr())
```

Out[11]:

`<AxesSubplot:>`

In [12]:

```python
x=df1[['Age', 'Na_to_K']]
y=df1[['Age']]
```

In [13]:

```python
from sklearn.model_selection import train_test_split
```

In [14]:

```python
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [15]:

```python
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)#ValueError: Input contains NaN, infinity or a value too large for
```

Out[15]:

```
LinearRegression()
```

In [16]:

```python
print(lr.intercept_)
```

```
[2.13162821e-14]
```

In [17]:

```python
coef= pd.DataFrame(lr.coef_)
coef
```

Out[17]:

	0	1
0	1.0	4.011805e-18

In [18]:

```python
print(lr.score(x_test,y_test))
```

```
1.0
```

In [19]:

```python
prediction = lr.predict(x_test)
plt.scatter(y_test,prediction)
```
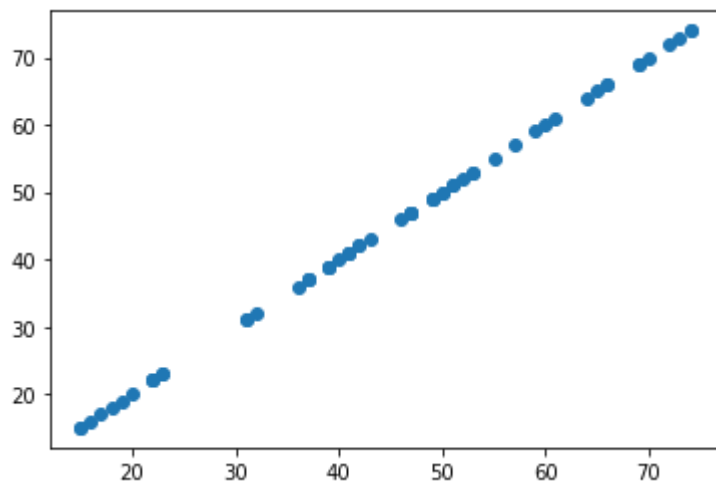
Out[19]:

```
<matplotlib.collections.PathCollection at 0x1c53cb2d220>
```


In [20]:

```python
lr.score(x_test,y_test)
```

Out[20]:

```
1.0
```

In [21]:

```python
lr.score(x_train,y_train)
```

Out[21]:

```
1.0
```

In [22]:

```python
from sklearn.linear_model import Ridge,Lasso
```

In [23]:

```python
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[23]:

```
Ridge(alpha=10)
```

In [24]:

```python
rr.score(x_test,y_test)
```

Out[24]:

```
0.99999929198331
```

In [25]:

```python
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[25]:

```
Lasso(alpha=10)
```

In [26]:

```python
la.score(x_test,y_test)
```

Out[26]:

```
0.9986074580534169
```