

Problem statement

predicting the house price in USA.To create a model to help him estimate of what the house would sell for.

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

```
In [2]: df=pd.read_csv("drug")
```

To display top 10 rows

```
In [3]: 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

Data Cleaning And Pre-Processing

In [4]: `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]: `# Display the statistical summary`
`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]: # To display the col headings  
df.columns
```

```
Out[6]: Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug'], dtype='object')
```

```
In [7]: cols=df.dropna(axis=1)
```

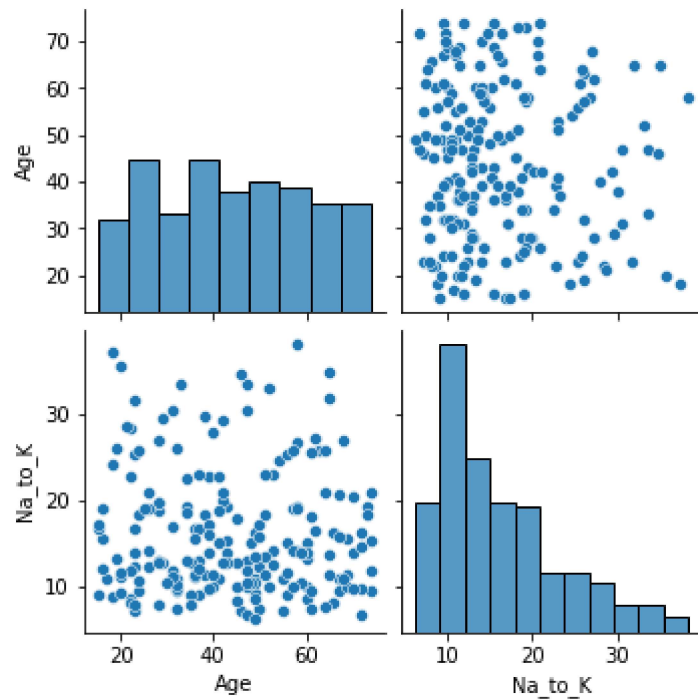
```
In [8]: cols.columns
```

```
Out[8]: Index(['Age', 'Sex', 'BP', 'Cholesterol', 'Na_to_K', 'Drug'], dtype='object')
```

EDA and Visualization

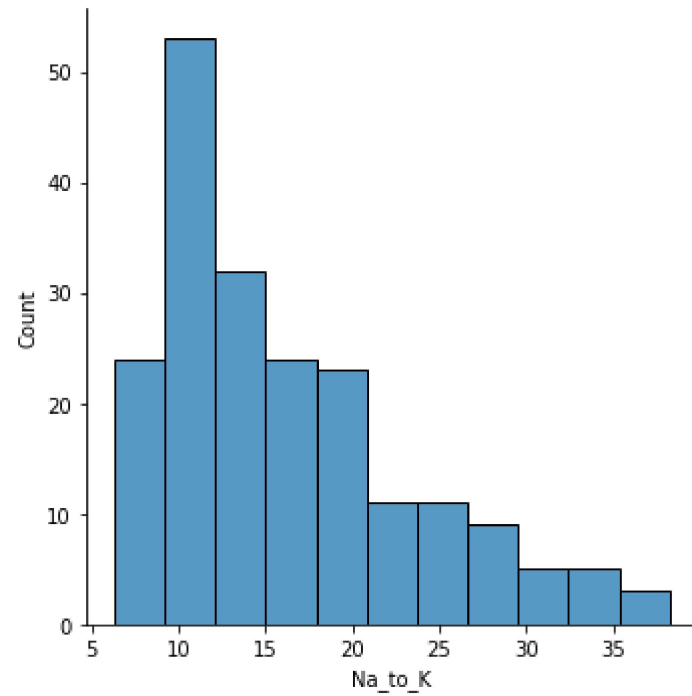
```
In [9]: sns.pairplot(cols)
```

```
Out[9]: <seaborn.axisgrid.PairGrid at 0x2986017d310>
```



```
In [11]: sns.displot(df['Na_to_K'])
```

```
Out[11]: <seaborn.axisgrid.FacetGrid at 0x29863ef0f70>
```

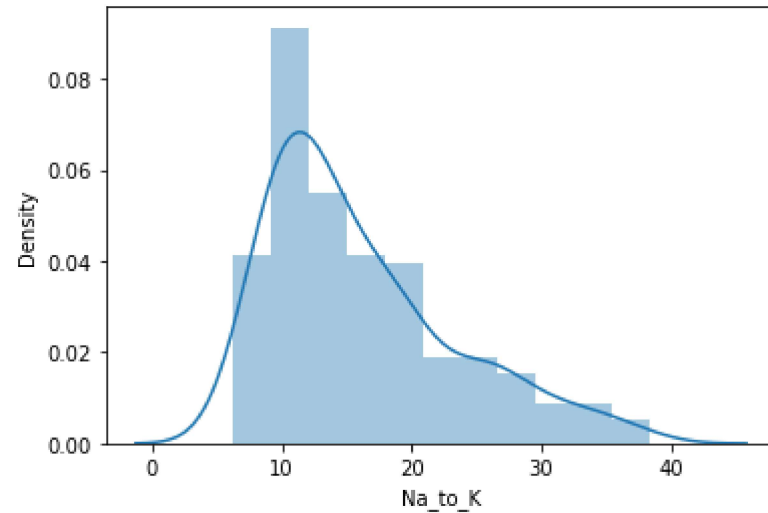


```
In [12]: # We use displot in older version we get distplot use displot
sns.distplot(df['Na_to_K'])
```

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[12]: <AxesSubplot:xlabel='Na_to_K', ylabel='Density'>
```



```
In [20]: df1=cols[['Age', 'Na_to_K']]
df1
```

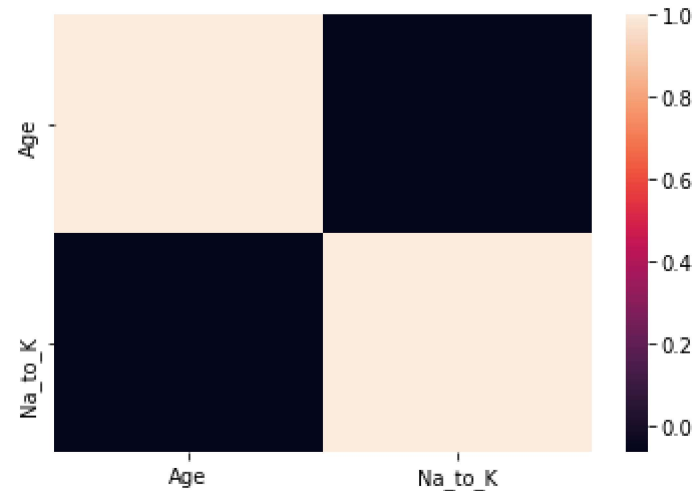
Out[20]:

	Age	Na_to_K
0	23	25.355
1	47	13.093
2	47	10.114
3	28	7.798
4	61	18.043
...
195	56	11.567
196	16	12.006
197	52	9.894
198	23	14.020
199	40	11.349

200 rows × 2 columns

```
In [21]: sns.heatmap(df1.corr())
```

```
Out[21]: <AxesSubplot:>
```



To train the model - MODEL BUILD

Going to train linear regression model; We split our data into 2 variables x and y where x is independent var(input) and y is dependent on x(output), we could ignore address col as it is not required for our model

```
In [23]: x=df1[['Age', 'Na_to_K']]  
y=df1[['Na_to_K']]
```

To split the dataset into test data

```
In [24]: # importing lib for splitting test data  
from sklearn.model_selection import train_test_split
```

```
In [25]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

```
In [26]: from sklearn.linear_model import LinearRegression
```

```
lr=LinearRegression()  
lr.fit(x_train,y_train)
```

```
Out[26]: LinearRegression()
```

```
In [27]: print(lr.intercept_)
```

```
[7.10542736e-15]
```

```
In [28]: print(lr.score(x_test,y_test))
```

```
1.0
```

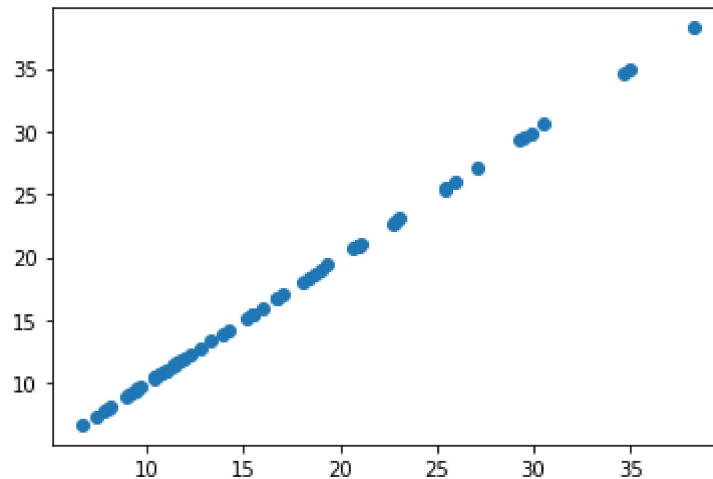
```
In [29]: coeff=pd.DataFrame(lr.coef_)  
coeff
```

```
Out[29]:
```

	0	1
0	0.0	1.0

```
In [30]: pred = lr.predict(x_test)  
plt.scatter(y_test,pred)
```

```
Out[30]: <matplotlib.collections.PathCollection at 0x298666be5e0>
```



In []: