```
In [51]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         from sklearn import datasets,linear_model
         from sklearn.metrics import mean_squared_error
         import seaborn as sns
```

```
In [52]: dia=pd.read_csv('diabetes.csv')
         dia
```

Out[52]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

```
In [53]: dia.head()
```

Out[53]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

```
In [54]: dia.tail()
```

Out[54]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

```
In [55]: dia.describe()
```

Out[55]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	C
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1
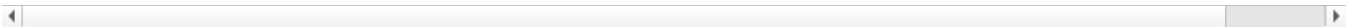
```
In [56]: dia.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [57]: `dia.keys()`

Out[57]:
```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

In [58]: `sns.histplot(dia,x='BMI')`

```
C:\Users\ajaykanth\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```
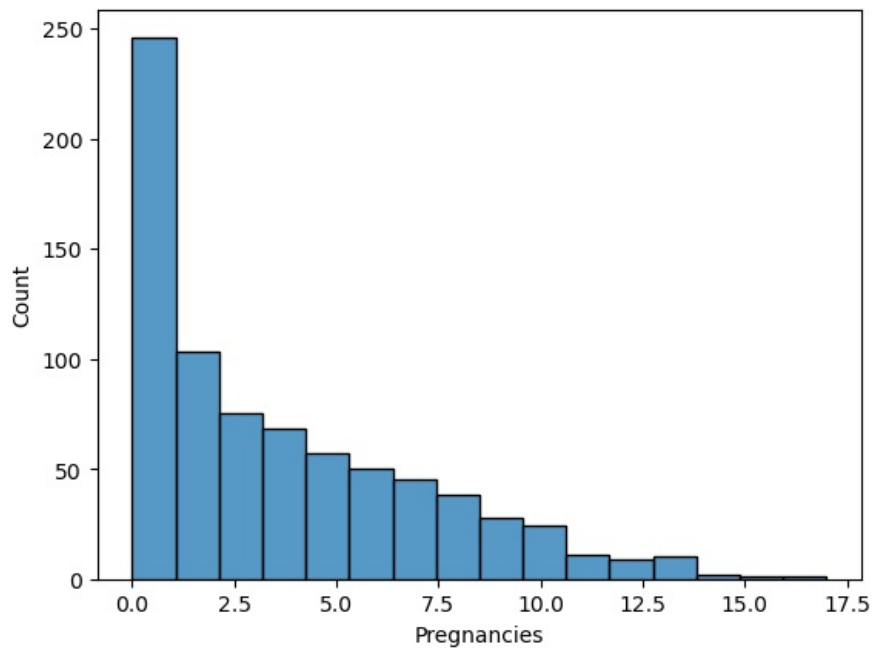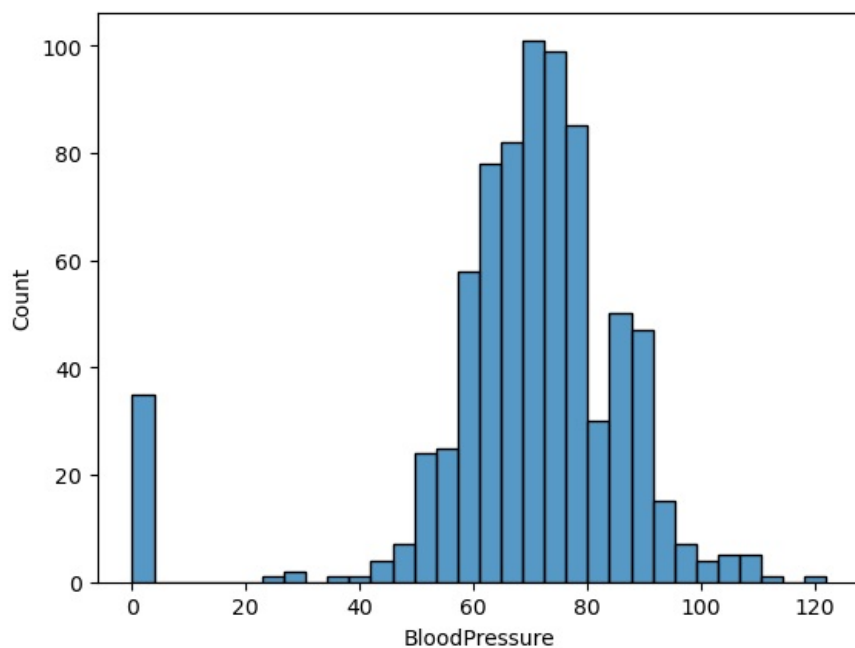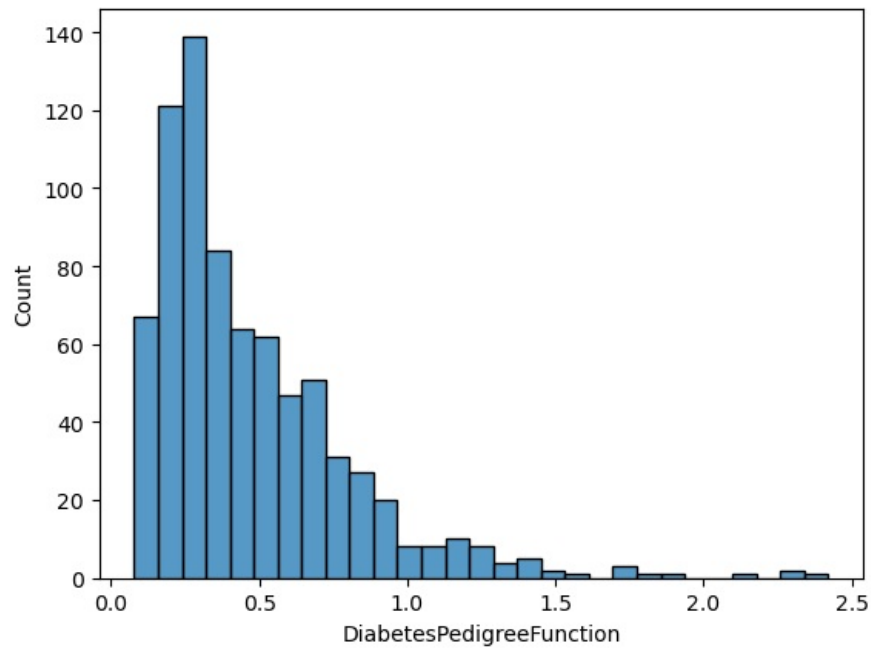
Out[58]: `<Axes: xlabel='BMI', ylabel='Count'>`
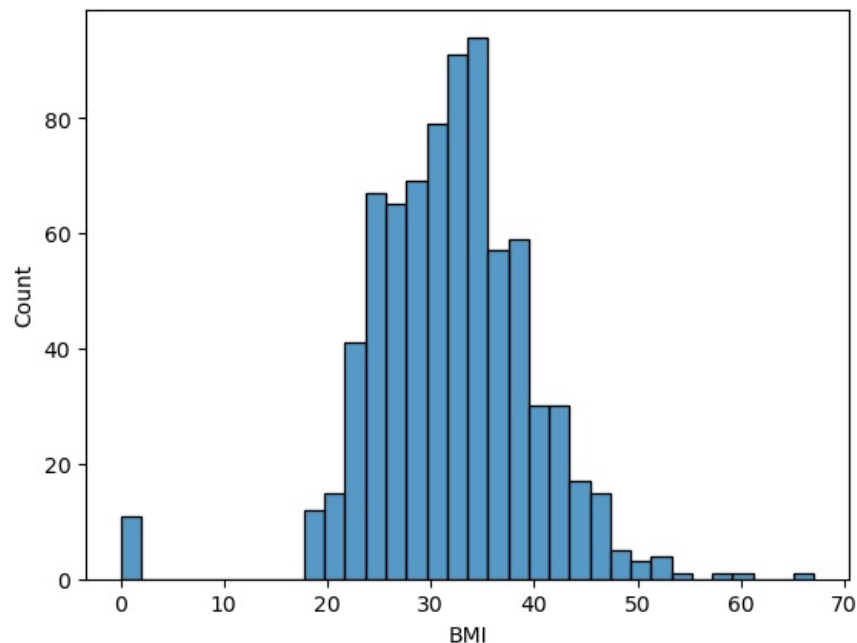
In [59]: `sns.histplot(dia,x='Pregnancies')`

```
C:\Users\ajaykanth\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```
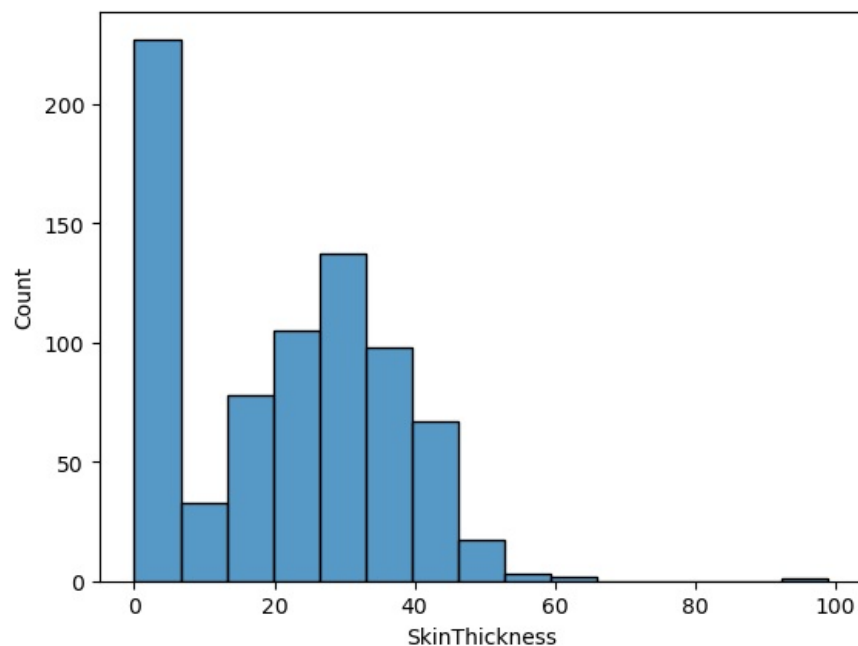
Out[59]: `<Axes: xlabel='Pregnancies', ylabel='Count'>`

In [60]: `sns.histplot(dia,x='BloodPressure')`

C:\Users\ajaykanth\anaconda3\Lib\site-packages\seaborn_oldcore.py:1119: FutureWarning: use_inf_as_na option is deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
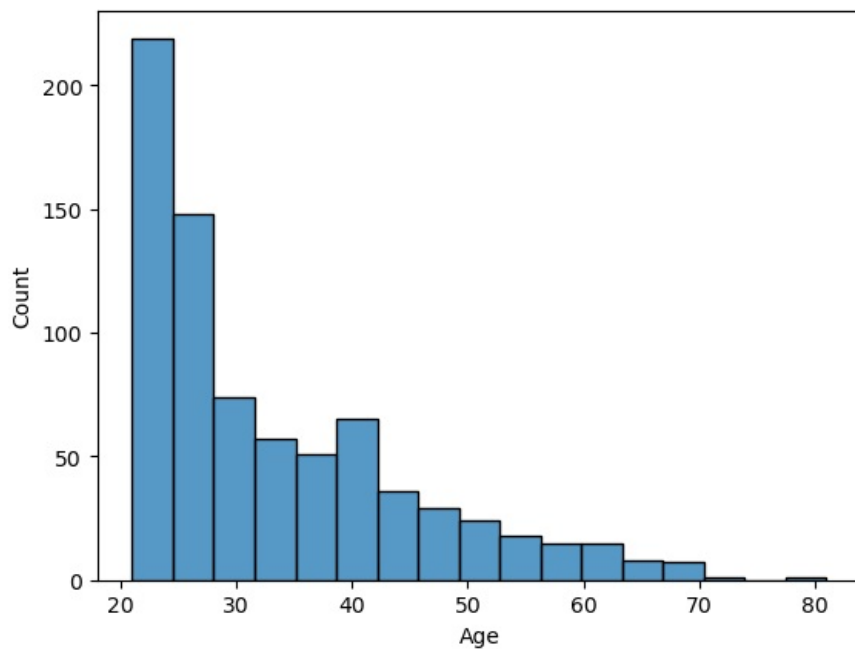 with pd.option_context('mode.use_inf_as_na', True):

Out[60]: `<Axes: xlabel='BloodPressure', ylabel='Count'>`

In [61]: `sns.histplot(dia,x='DiabetesPedigreeFunction')`

Out[61]: <Axes: xlabel='DiabetesPedigreeFunction', ylabel='Count'>

In [62]:
```
sns.histplot(dia,x='SkinThickness')
```

Out[62]: <Axes: xlabel='SkinThickness', ylabel='Count'>

In [63]:
```
sns.histplot(dia,x='Age')
```

Out[63]: <Axes: xlabel='Age', ylabel='Count'>

`sns.histplot(dia,x='Insulin')`

```
C:\Users\ajaykanth\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```
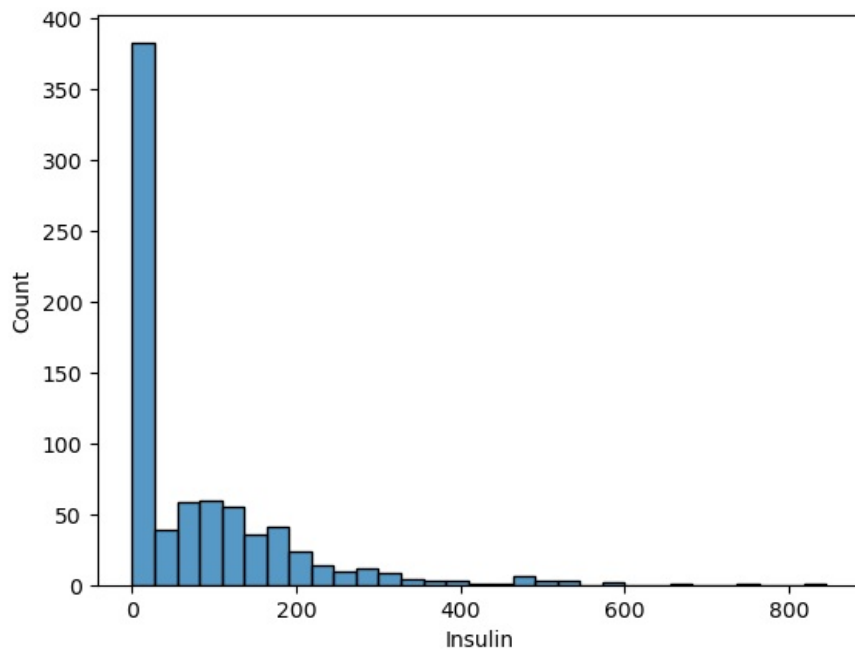
Out[64]: <Axes: xlabel='Insulin', ylabel='Count'>

`sns.histplot(dia,x='SkinThickness')`

```
C:\Users\ajaykanth\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_as_na option is
deprecated and will be removed in a future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```
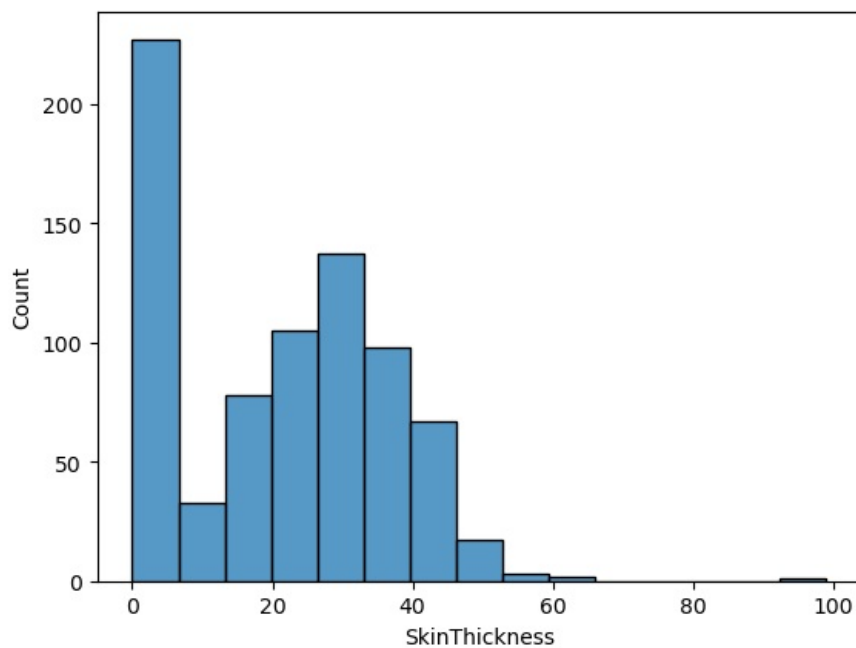
Out[65]: <Axes: xlabel='SkinThickness', ylabel='Count'>

```
In [66]: dia_X=dia
         dia_X
```

Out[66]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
...	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	0.171	63	0
764	2	122	70	27	0	36.8	0.340	27	0
765	5	121	72	23	112	26.2	0.245	30	0
766	1	126	60	0	0	30.1	0.349	47	1
767	1	93	70	31	0	30.4	0.315	23	0

768 rows × 9 columns

```
In [67]: dia_X_train=dia_X[:-30]
         dia_X_test=dia_X[-30:]
         dia_y_train=dia_X[:-30]
         dia_y_test=dia_X[-30:]
```

```
In [68]: model=linear_model.LinearRegression()
         model.fit(dia_X_train,dia_y_train)
```

Out[68]: ▼ LinearRegression

LinearRegression()

```
In [69]: dia_y_predicted=model.predict(dia_X_test)
         print('mean square error :',mean_squared_error(dia_y_predicted,dia_y_test))
```
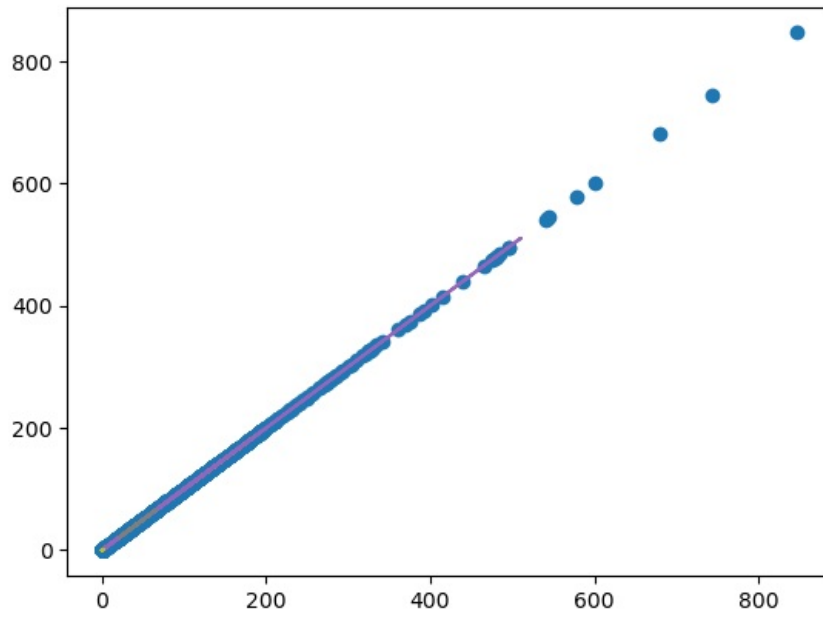
mean square error : 3.7073180087291624e-28

```
In [70]: model.coef_
         model.intercept_
```

Out[70]: array([-1.77635684e-15, -9.94759830e-14, -5.68434189e-14, 7.10542736e-15,
               -1.27897692e-13, 1.42108547e-14, -2.77555756e-16, -2.84217094e-14,
               -3.88578059e-15])

```
In [71]: plt.scatter(dia_X_train,dia_y_train)
```

```
plt.plot(dia_y_predicted,dia_X_test)
plt.show()
```