In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [2]:
```
df=pd.read_excel('health_data_nulls.xlsx')
df1=pd.read_excel('health_data_oob_values.xlsx')
df
```

Out[2]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure
0	1	61.0	268300.0	41.0	NaN	3.0	62
1	2	55.0	122200.0	51.0	7.0	56.0	53
2	3	53.0	82100.0	37.0	0.0	55.0	42
3	4	30.0	101400.0	41.0	20.0	61.0	48
4	5	64.0	181100.0	NaN	0.0	70.0	81
...	...	...	...	...	...	...	...
995	996	50.0	141300.0	9.0	11.0	36.0	43
996	997	40.0	155700.0	0.0	NaN	3.0	29
997	998	36.0	84700.0	42.0	47.0	21.0	21
998	999	51.0	124500.0	63.0	40.0	1.0	25
999	1000	28.0	241200.0	NaN	90.0	0.0	42

1000 rows × 7 columns

In [3]:
```
df1.head(20)
```

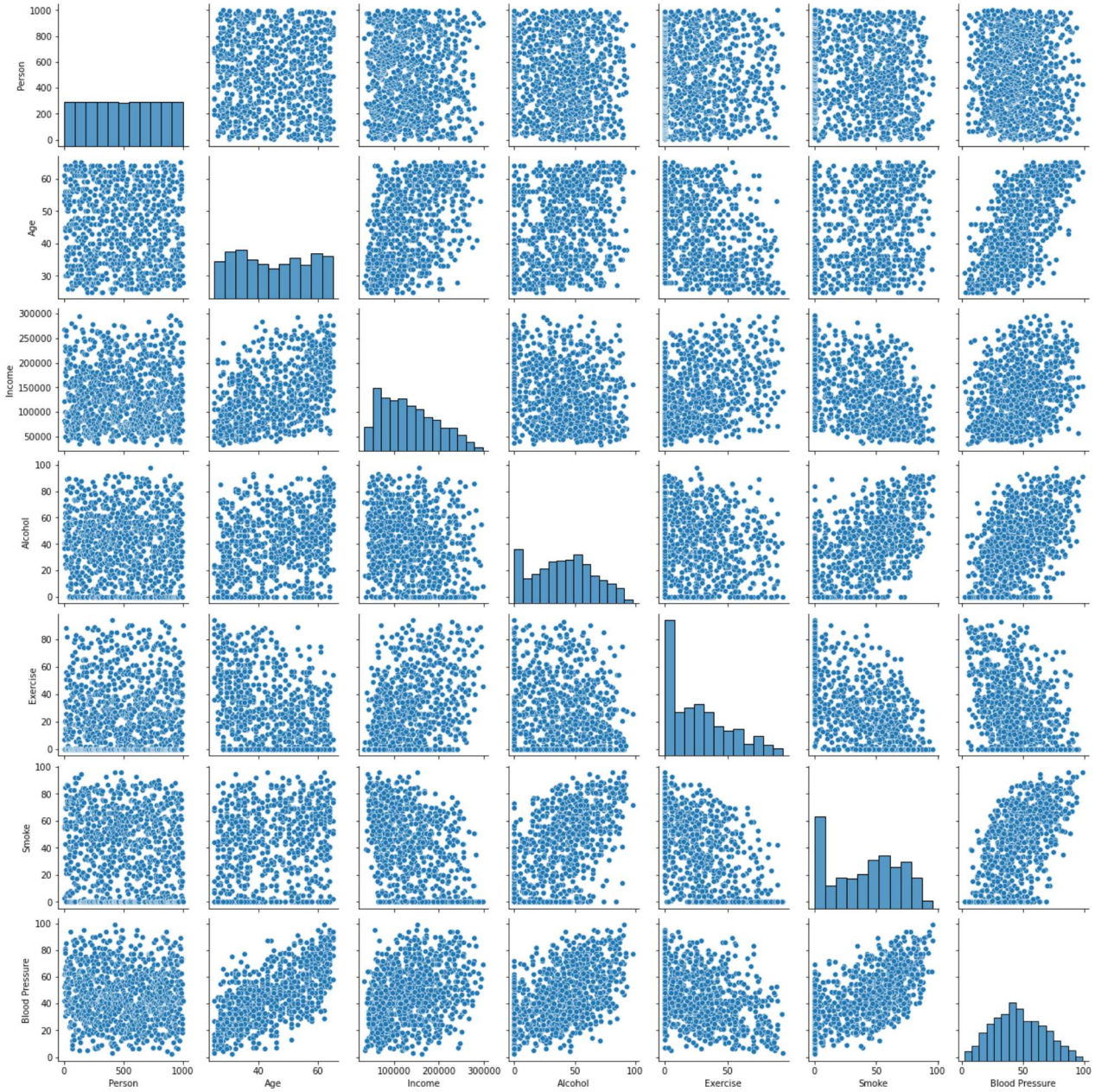
Out[3]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure
0	1	61	268300	41	999	3	62
1	2	55	122200	51	7	56	53
2	3	53	82100	37	0	55	42
3	4	30	101400	41	20	61	48
4	5	64	181100	999	0	70	81
5	6	45	156600	60	35	999	80
6	7	56	160400	55	999	59	63
7	8	999	78800	31	12	43	31
8	9	59	233500	25	15	33	66
9	10	44	50400	64	0	85	54
10	11	999	224400	69	21	55	78

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure
11	12	42	175000	22	26	21	43
12	13	63	255900	46	32	24	68
13	14	30	70300	53	999	79	56
14	15	52	229500	74	56	69	85
15	16	54	188600	62	51	5	39
16	17	42	265000	41	88	0	47
17	18	36	65400	41	14	59	34
18	19	56	81200	64	0	71	53
19	20	33	67200	46	9	54	33

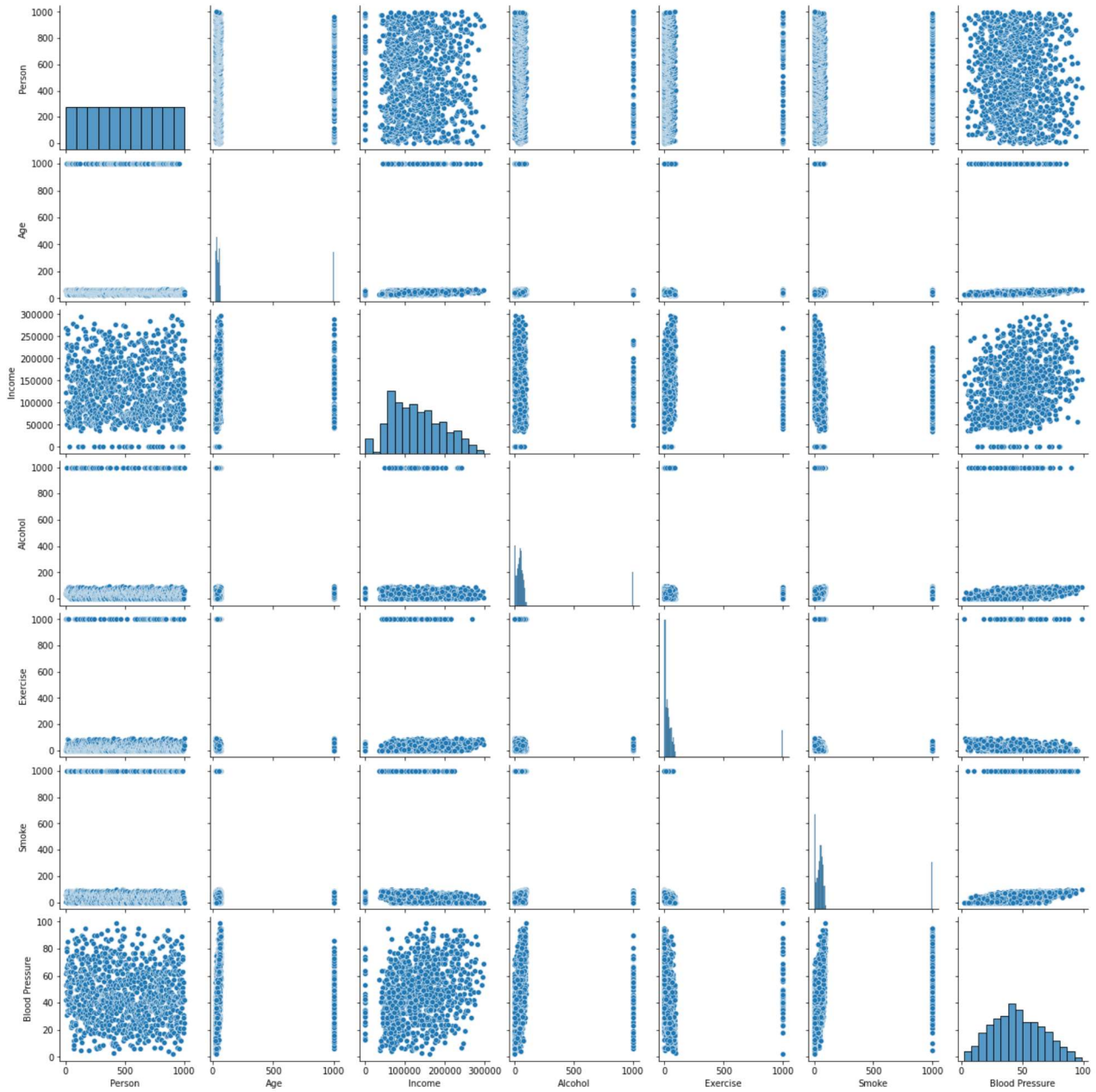
In [4]:
```
sns.pairplot(df)
```

Out[4]:
```
<seaborn.axisgrid.PairGrid at 0x19292de59d0>
```



In [5]:
```
sns.pairplot(df1)
```

Out[5]: <seaborn.axisgrid.PairGrid at 0x19299bcda00>



In [6]:
```
df1.Alcohol[df1.Alcohol==999]=np.nan
df1.head(15)
```

Out[6]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure
0	1	61	268300	41.0	999	3	62
1	2	55	122200	51.0	7	56	53
2	3	53	82100	37.0	0	55	42
3	4	30	101400	41.0	20	61	48
4	5	64	181100	NaN	0	70	81
5	6	45	156600	60.0	35	999	80
6	7	56	160400	55.0	999	59	63
7	8	999	78800	31.0	12	43	31

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure
8	9	59	233500	25.0	15	33	66
9	10	44	50400	64.0	0	85	54
10	11	999	224400	69.0	21	55	78
11	12	42	175000	22.0	26	21	43
12	13	63	255900	46.0	32	24	68
13	14	30	70300	53.0	999	79	56
14	15	52	229500	74.0	56	69	85

In [7]:
```
df['Age'].isnull().sum()
```

Out[7]: 104

In [8]:
```
df['Income'].isnull().sum()
```

Out[8]: 31

In [9]:
```
df.shape
```

Out[9]: (1000, 7)

In [10]:
```
df['Age'].mean()
```

Out[10]: 44.825892857142854

In [11]:
```
df['AgeImpMean']=df['Age'].fillna(df['Age'].mean())
df.head(15)
```

Out[11]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean
0	1	61.0	268300.0	41.0	NaN	3.0	62	61.000000
1	2	55.0	122200.0	51.0	7.0	56.0	53	55.000000
2	3	53.0	82100.0	37.0	0.0	55.0	42	53.000000
3	4	30.0	101400.0	41.0	20.0	61.0	48	30.000000
4	5	64.0	181100.0	NaN	0.0	70.0	81	64.000000
5	6	45.0	156600.0	60.0	35.0	NaN	80	45.000000
6	7	56.0	160400.0	55.0	NaN	59.0	63	56.000000
7	8	NaN	78800.0	31.0	12.0	43.0	31	44.825893
8	9	59.0	233500.0	25.0	15.0	33.0	66	59.000000
9	10	44.0	50400.0	64.0	0.0	85.0	54	44.000000

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean
10	11	NaN	224400.0	69.0	21.0	55.0	78	44.825893
11	12	42.0	175000.0	22.0	26.0	21.0	43	42.000000
12	13	63.0	255900.0	46.0	32.0	24.0	68	63.000000
13	14	30.0	70300.0	53.0	NaN	79.0	56	30.000000
14	15	52.0	229500.0	74.0	56.0	69.0	85	52.000000

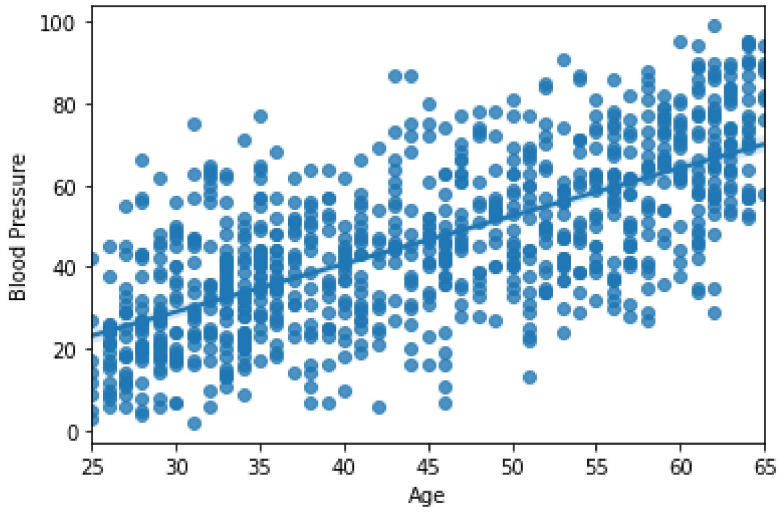
In [12]:
```
df['AgeImpMed']=df['Age'].fillna(df['Age'].median())
df['AgeImpMode']=df['Age'].fillna(df['Age'].mode())
df.head(15)
```

Out[12]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean	AgeImpMed	AgeImpMo
0	1	61.0	268300.0	41.0	NaN	3.0	62	61.000000	61.0	6
1	2	55.0	122200.0	51.0	7.0	56.0	53	55.000000	55.0	5
2	3	53.0	82100.0	37.0	0.0	55.0	42	53.000000	53.0	5
3	4	30.0	101400.0	41.0	20.0	61.0	48	30.000000	30.0	3
4	5	64.0	181100.0	NaN	0.0	70.0	81	64.000000	64.0	6
5	6	45.0	156600.0	60.0	35.0	NaN	80	45.000000	45.0	4
6	7	56.0	160400.0	55.0	NaN	59.0	63	56.000000	56.0	5
7	8	NaN	78800.0	31.0	12.0	43.0	31	44.825893	45.0	N
8	9	59.0	233500.0	25.0	15.0	33.0	66	59.000000	59.0	5
9	10	44.0	50400.0	64.0	0.0	85.0	54	44.000000	44.0	4
10	11	NaN	224400.0	69.0	21.0	55.0	78	44.825893	45.0	N
11	12	42.0	175000.0	22.0	26.0	21.0	43	42.000000	42.0	4
12	13	63.0	255900.0	46.0	32.0	24.0	68	63.000000	63.0	6
13	14	30.0	70300.0	53.0	NaN	79.0	56	30.000000	30.0	3
14	15	52.0	229500.0	74.0	56.0	69.0	85	52.000000	52.0	5

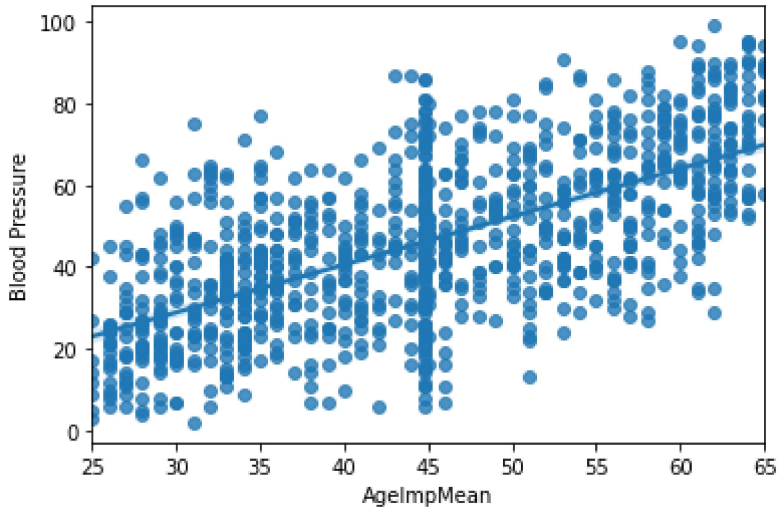
In [13]:
```
sns.regplot(x='Age',y='Blood Pressure',data=df)
```

Out[13]:
```
<AxesSubplot:xlabel='Age', ylabel='Blood Pressure'>
```



In [14]:
```
sns.regplot(x='AgeImpMean',y='Blood Pressure',data=df)
```

Out[14]: <AxesSubplot:xlabel='AgeImpMean', ylabel='Blood Pressure'>



In [16]:
```
df['AgeNA']=df['Age'].isna()
df.head(15)
```

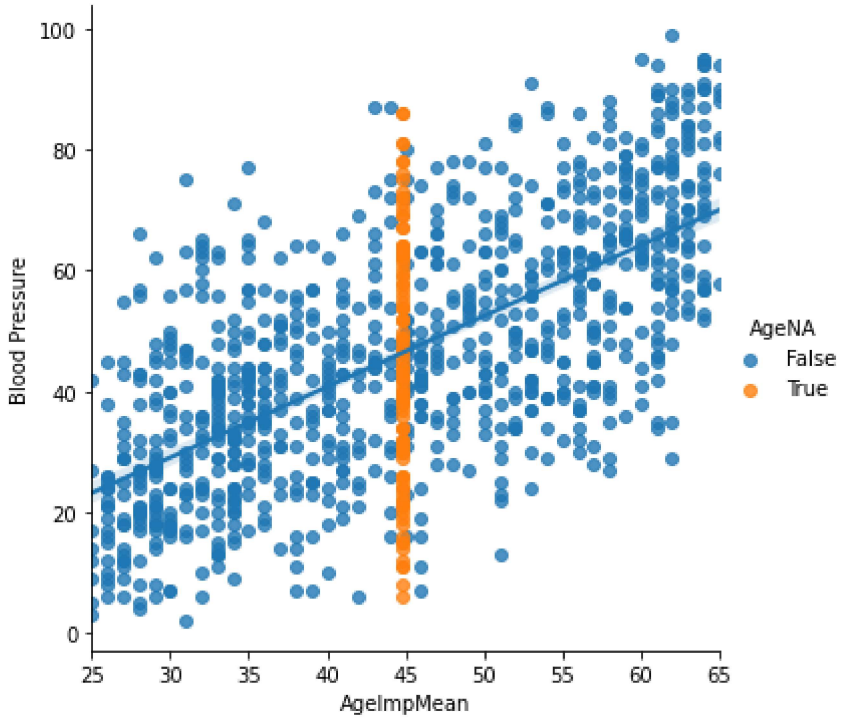
Out[16]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean	AgeImpMed	AgeImpMo
0	1	61.0	268300.0	41.0	NaN	3.0	62	61.000000	61.0	6
1	2	55.0	122200.0	51.0	7.0	56.0	53	55.000000	55.0	5
2	3	53.0	82100.0	37.0	0.0	55.0	42	53.000000	53.0	5
3	4	30.0	101400.0	41.0	20.0	61.0	48	30.000000	30.0	3
4	5	64.0	181100.0	NaN	0.0	70.0	81	64.000000	64.0	6
5	6	45.0	156600.0	60.0	35.0	NaN	80	45.000000	45.0	4
6	7	56.0	160400.0	55.0	NaN	59.0	63	56.000000	56.0	5
7	8	NaN	78800.0	31.0	12.0	43.0	31	44.825893	45.0	N

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean	AgeImpMed	AgeImpMo
8	9	59.0	233500.0	25.0	15.0	33.0	66	59.000000	59.0	5
9	10	44.0	50400.0	64.0	0.0	85.0	54	44.000000	44.0	4
10	11	NaN	224400.0	69.0	21.0	55.0	78	44.825893	45.0	N
11	12	42.0	175000.0	22.0	26.0	21.0	43	42.000000	42.0	4
12	13	63.0	255900.0	46.0	32.0	24.0	68	63.000000	63.0	6
13	14	30.0	70300.0	53.0	NaN	79.0	56	30.000000	30.0	3
14	15	52.0	229500.0	74.0	56.0	69.0	85	52.000000	52.0	5

In [17]:
```
sns.lmplot(x='AgeImpMean',y='Blood Pressure',data=df, hue="AgeNA")
```

Out[17]: <seaborn.axisgrid.FacetGrid at 0x1929d2e9160>



In [18]:
```
from scipy import stats
```

In [19]:
```
corr,_ =stats.pearsonr(df['AgeImpMean'],df['Blood Pressure'])
corr
```

Out[19]: 0.6449459380613165

In [20]:
```
import statsmodels.api as sm
```

In [21]:
```
correlation = df['AgeImpMean'].corr(df['Blood Pressure'])
correlation
```

Out[21]: 0.6449459380613166

In [22]:
```
correlation = df['Age'].corr(df['Income'])
correlation
```

Out[22]: 0.4989889602448255

In [24]:
```
correlation = df['Age'].corr(df['Blood Pressure'])
correlation
```

Out[24]: 0.6785621974236399

In [23]:
```
corr,_ =stats.pearsonr(df['Age'],df['Blood Pressure'])
corr
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-23-612ab9ead134> in <module>
----> 1 corr,_ =stats.pearsonr(df['Age'],df['Blood Pressure'])
      2 corr

~\anaconda3\lib\site-packages\scipy\stats\stats.py in pearsonr(x, y)
   3933     # scipy.linalg.norm(xm) does not overflow if xm is, for example,
   3934     # [-5e210, 5e210, 3e200, -3e200]
-> 3935     normxm = linalg.norm(xm)
   3936     normym = linalg.norm(ym)
   3937

~\anaconda3\lib\site-packages\scipy\linalg\misc.py in norm(a, ord, axis, keepdims, check
_finite)
    138     # Differs from numpy only in non-finite handling and the use of blas.
    139     if check_finite:
--> 140         a = np.asarray_chkfinite(a)
    141     else:
    142         a = np.asarray(a)

~\anaconda3\lib\site-packages\numpy\lib\function_base.py in asarray_chkfinite(a, dtype,
order)
    486     a = asarray(a, dtype=dtype, order=order)
    487     if a.dtype.char in typecodes['AllFloat'] and not np.isfinite(a).all():
--> 488         raise ValueError(
    489             "array must not contain infs or NaNs")
    490     return a

ValueError: array must not contain infs or NaNs
```

In [24]:
```
df.dropna()
```

Out[24]:

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean	AgeImpMed	AgeImpMo
1	2	55.0	122200.0	51.0	7.0	56.0	53	55.0	55.0	5

	Person	Age	Income	Alcohol	Exercise	Smoke	Blood Pressure	AgeImpMean	AgeImpMed	AgeImpMo
2	3	53.0	82100.0	37.0	0.0	55.0	42	53.0	53.0	5
3	4	30.0	101400.0	41.0	20.0	61.0	48	30.0	30.0	3
8	9	59.0	233500.0	25.0	15.0	33.0	66	59.0	59.0	5
9	10	44.0	50400.0	64.0	0.0	85.0	54	44.0	44.0	4
...	...	...	...	...	...	...	...	...	...	
991	992	44.0	138300.0	23.0	27.0	12.0	31	44.0	44.0	4
994	995	30.0	63000.0	34.0	28.0	33.0	18	30.0	30.0	3
995	996	50.0	141300.0	9.0	11.0	36.0	43	50.0	50.0	5
997	998	36.0	84700.0	42.0	47.0	21.0	21	36.0	36.0	3
998	999	51.0	124500.0	63.0	40.0	1.0	25	51.0	51.0	5

638 rows × 11 columns

In [25]:
```
from statsmodels.formula.api import ols #or glm for logistic
```

In [28]:
```
df2=df
```

In [29]:
```
df2['BP']=df['Blood Pressure']
```

In [30]:
```
model_lm=ols(formula='BP~Age', data=df2).fit()
print(model_lm.params)
```

```
Intercept   -6.063997
Age          1.170684
dtype: float64
```

In [31]:
```
model_lm=ols(formula='Blood Pressure~Age', data=df).fit()
print(model_lm.params)
```

```
Traceback (most recent call last):

  File "C:\Users\Top\anaconda3\lib\site-packages\IPython\core\interactiveshell.py", line
3437, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)

  File "<ipython-input-31-d980f41bc4fe>", line 1, in <module>
    model_lm=ols(formula='Blood Pressure~Age', data=df).fit()

  File "C:\Users\Top\anaconda3\lib\site-packages\statsmodels\base\model.py", line 169,
in from_formula
    tmp = handle_formula_data(data, None, formula, depth=eval_env,

  File "C:\Users\Top\anaconda3\lib\site-packages\statsmodels\formula\formulatools.py", l
```

```
ine 63, in handle_formula_data
    result = dmatrices(formula, Y, depth, return_type='dataframe',

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\highlevel.py", line 309, in dmat
rices
    (lhs, rhs) = _do_highlevel_design(formula_like, data, eval_env,

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\highlevel.py", line 164, in _do_
highlevel_design
    design_infos = _try_incr_builders(formula_like, data_iter_maker, eval_env,

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\highlevel.py", line 66, in _try_
incr_builders
    return design_matrix_builders([formula_like.lhs_termlist,

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\build.py", line 689, in design_m
atrix_builders
    factor_states = _factors_memorize(all_factors, data_iter_maker, eval_env)

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\build.py", line 354, in _factors
_memorize
    which_pass = factor.memorize_passes_needed(state, eval_env)

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\eval.py", line 474, in memorize_
passes_needed
    subset_names = [name for name in ast_names(self.code)

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\eval.py", line 474, in <listcomp
>
    subset_names = [name for name in ast_names(self.code)

  File "C:\Users\Top\anaconda3\lib\site-packages\patsy\eval.py", line 105, in ast_names
    for node in ast.walk(ast.parse(code)):

  File "C:\Users\Top\anaconda3\lib\ast.py", line 47, in parse
    return compile(source, filename, mode, flags,

  File "<unknown>", line 1
    Blood Pressure
          ^
SyntaxError: invalid syntax
```

In [32]:
```
model_lm=ols(formula='Age~BP', data=df2).fit()
print(model_lm.params)
```

```
Intercept    26.571017
BP            0.393314
dtype: float64
```

In [35]:
```
df2['Predicted']=df2['BP']*0.393314+26.571017
```

In [36]:
```
df2['Predicted'].head()
```

Out[36]:
```
0    50.956485
1    47.416659
2    43.090205
3    45.450089
4    58.429451
Name: Predicted, dtype: float64
```

In [ ]:

```
df['AgeImpMed']=df['Age'].fillna(df['Age'].median())
df['AgeImpMode']=df['Age'].fillna(df['Age'].mode())
df.head(15)
#how do we modify this to use the regression line?
```