```
Start coding or generate with AI.
```

support vector machine - used for classification and regression problem

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

```
df=pd.read_csv('survey lung cancer.csv')
```

```
df
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	C
0	M	69	1	2	2	1	1	2	1	2	2	
1	M	74	2	1	1	1	2	2	2	1	1	
2	F	59	1	1	1	2	1	2	1	2	1	
3	M	63	2	2	2	1	1	1	1	1	2	
4	F	63	1	2	1	1	1	1	1	2	1	
...	...	...	...	...	...	...	...	...	...	...	...	...
304	F	56	1	1	1	2	2	2	1	1	2	
305	M	70	2	1	1	1	1	2	2	2	2	
306	M	58	2	1	1	1	1	1	2	2	2	
307	M	67	2	1	2	1	1	2	2	1	2	
308	M	62	1	1	1	2	1	2	2	2	2	

309 rows × 16 columns

Next steps: Generate code with df | New interactive sheet

INSPECTION

```
df.head()
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COU
0	M	69	1	2	2	1	1	2	1	2	2	
1	M	74	2	1	1	1	2	2	2	1	1	
2	F	59	1	1	1	2	1	2	1	2	1	
3	M	63	2	2	2	1	1	1	1	1	2	
4	F	63	1	2	1	1	1	1	1	2	1	

Next steps: Generate code with df | New interactive sheet

```
df.tail()
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	C
304	F	56	1	1	1	2	2	2	1	1	2	
305	M	70	2	1	1	1	1	2	2	2	2	
306	M	58	2	1	1	1	1	1	2	2	2	
307	M	67	2	1	2	1	1	2	2	1	2	
308	M	62	1	1	1	2	1	2	2	2	2	

Start coding or generate with AI.

```
df.sample()
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	C
60	M	70	1	2	1	2	2	2	2	2	2	

```
df.describe()
```

	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZI
count	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.000000	309.0000
mean	62.673139	1.563107	1.569579	1.498382	1.501618	1.504854	1.673139	1.556634	1.5566
std	8.210301	0.496806	0.495938	0.500808	0.500808	0.500787	0.469827	0.497588	0.4975
min	21.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000
25%	57.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.0000
50%	62.000000	2.000000	2.000000	1.000000	2.000000	2.000000	2.000000	2.000000	2.0000
75%	69.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.0000
max	87.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.000000	2.0000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 309 entries, 0 to 308
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   GENDER                 309 non-null    object
 1   AGE                    309 non-null    int64
 2   SMOKING                309 non-null    int64
 3   YELLOW_FINGERS         309 non-null    int64
 4   ANXIETY                309 non-null    int64
 5   PEER_PRESSURE          309 non-null    int64
 6   CHRONIC DISEASE        309 non-null    int64
 7   FATIGUE                309 non-null    int64
 8   ALLERGY                309 non-null    int64
 9   WHEEZING               309 non-null    int64
 10  ALCOHOL CONSUMING      309 non-null    int64
 11  COUGHING               309 non-null    int64
 12  SHORTNESS OF BREATH    309 non-null    int64
 13  SWALLOWING DIFFICULTY  309 non-null    int64
 14  CHEST PAIN             309 non-null    int64
 15  LUNG_CANCER            309 non-null    object
dtypes: int64(14), object(2)
```

```
memory usage: 38.8+ KB
```

```
df.isnull().sum()
```

	0
GENDER	0
AGE	0
SMOKING	0
YELLOW_FINGERS	0
ANXIETY	0
PEER_PRESSURE	0
CHRONIC DISEASE	0
FATIGUE	0
ALLERGY	0
WHEEZING	0
ALCOHOL CONSUMING	0
COUGHING	0
SHORTNESS OF BREATH	0
SWALLOWING DIFFICULTY	0
CHEST PAIN	0
LUNG_CANCER	0

dtype: int64

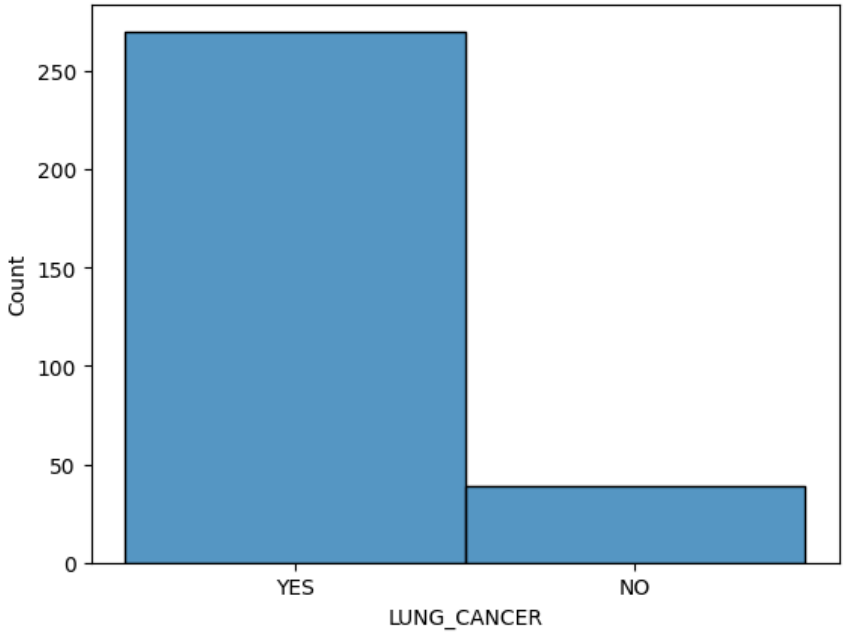
```
df.nunique()
```

	0
GENDER	2
AGE	39
SMOKING	2
YELLOW_FINGERS	2
ANXIETY	2
PEER_PRESSURE	2
CHRONIC DISEASE	2
FATIGUE	2
ALLERGY	2
WHEEZING	2
ALCOHOL CONSUMING	2
COUGHING	2
SHORTNESS OF BREATH	2
SWALLOWING DIFFICULTY	2
CHEST PAIN	2
LUNG_CANCER	2

dtype: int64

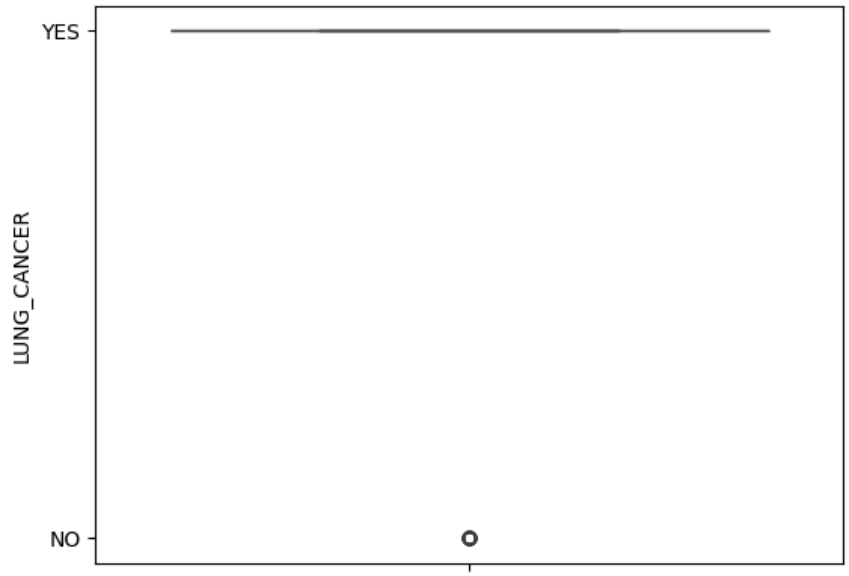
```
sns.histplot(df['LUNG_CANCER'])
```

```
<Axes: xlabel='LUNG_CANCER', ylabel='Count'>
```



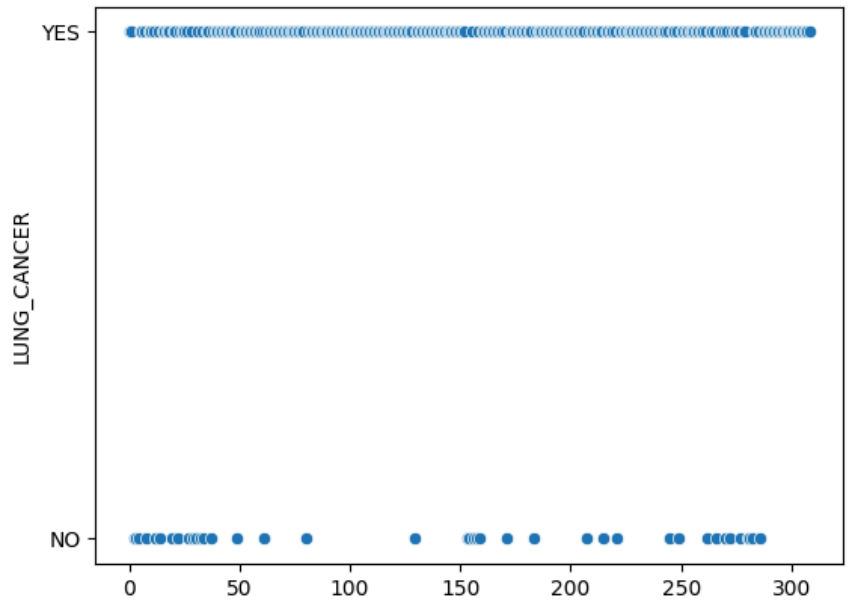

```
sns.boxplot(df['LUNG_CANCER'])
```

```
<Axes: ylabel='LUNG_CANCER'>
```




```
sns.scatterplot(df['LUNG_CANCER'])
```

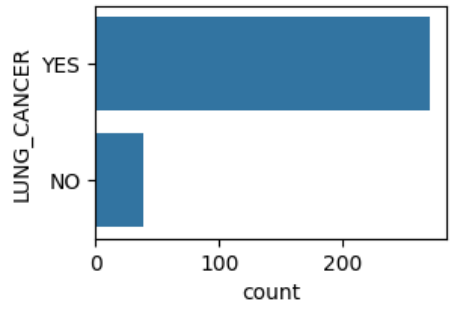
```
<Axes: ylabel='LUNG_CANCER'>
```



UNIVARIENT ANALYSIS

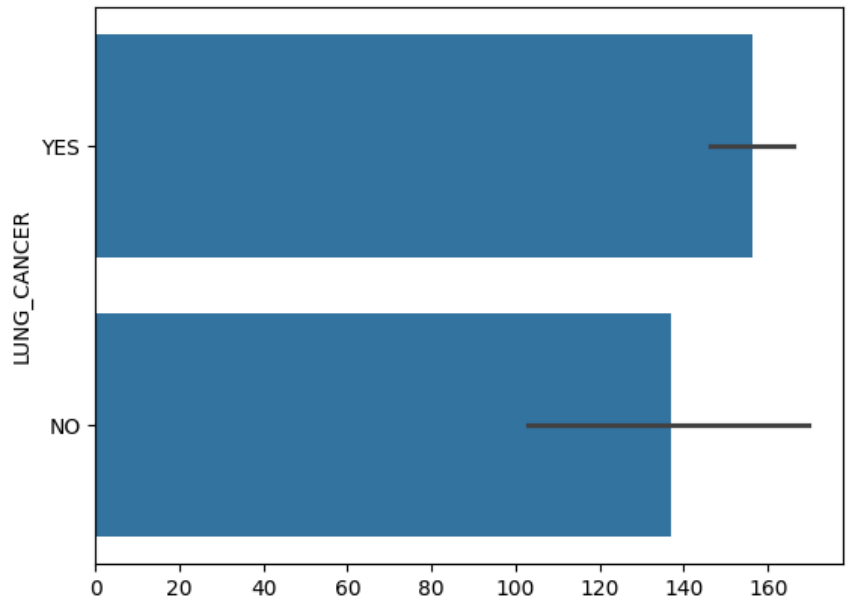
```
plt.figure(figsize=(3,2))
sns.countplot(df['LUNG_CANCER'])
```

```
<Axes: xlabel='count', ylabel='LUNG_CANCER'>
```



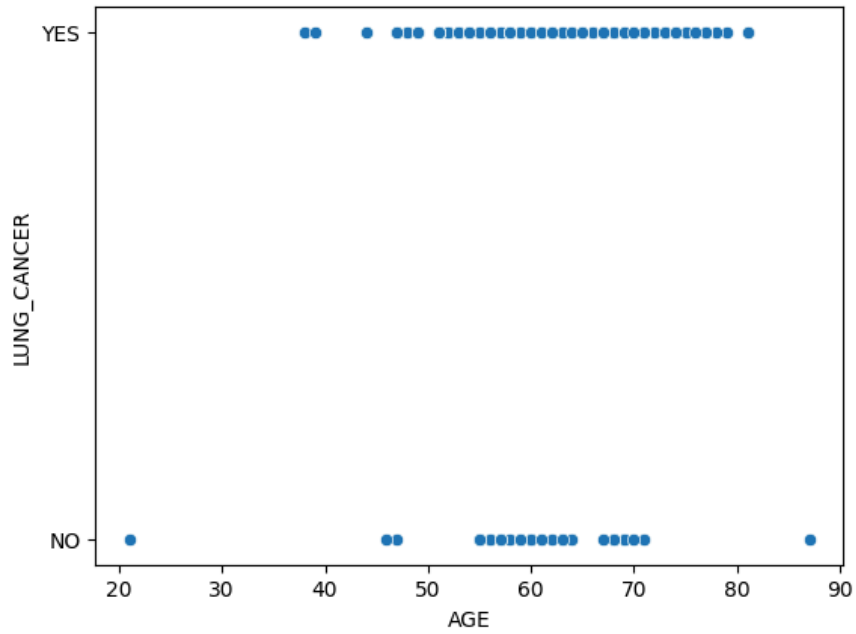

```
sns.barplot(df['LUNG_CANCER'])
```

```
<Axes: ylabel='LUNG_CANCER'>
```




```
sns.scatterplot(x='AGE',y='LUNG_CANCER',data=df)
```

```
<Axes: xlabel='AGE', ylabel='LUNG_CANCER'>
```



ENCODING

```
df.columns
```

```
Index(['GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
       'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
       'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
       'SWALLOWING DIFFICULTY', 'CHEST PAIN', 'LUNG_CANCER'],
      dtype='object')
```

```
df.head()
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COU
0	M	69	1	2	2	1	1	2	1	2	2	
1	M	74	2	1	1	1	2	2	2	1	1	
2	F	59	1	1	1	2	1	2	1	2	1	
3	M	63	2	2	2	1	1	1	1	1	2	
4	F	63	1	2	1	1	1	1	1	2	1	

Next steps: Generate code with df | New interactive sheet

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
df['GENDER']=le.fit_transform(df['GENDER'])
df['LUNG_CANCER']=le.fit_transform(df['LUNG_CANCER'])
```

```
df
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	C
0	1	69	1	2	2	1	1	2	1	2	2	
1	1	74	2	1	1	1	2	2	2	1	1	
2	0	59	1	1	1	2	1	2	1	2	1	
3	1	63	2	2	2	1	1	1	1	1	2	
4	0	63	1	2	1	1	1	1	1	2	1	
...	...	...	...	...	...	...	...	...	...	...	...	...
304	0	56	1	1	1	2	2	2	1	1	2	
305	1	70	2	1	1	1	1	2	2	2	2	
306	1	58	2	1	1	1	1	1	2	2	2	
307	1	67	2	1	2	1	1	2	2	1	2	
308	1	62	1	1	1	2	1	2	2	2	2	

309 rows × 16 columns

Next steps: Generate code with df | New interactive sheet

BREAK X AND Y

```
x=df.drop('LUNG_CANCER',axis=1)
y=df['LUNG_CANCER']
```

TRAIN TEST SPLIT

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

STANDARD SCALER

```
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.fit_transform(x_test)
```

MODEL

```
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
```

```
▾ SVC
SVC()
```

SCORE

```
model.score(x_train,y_train)*100,model.score(x_test,y_test)*100
```

```
(94.73684210526315, 93.54838709677419)
```

CONFUSION MATRIX

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,model.predict(x_test))
```

```
array([[ 1,  1],
       [ 3, 57]])
```

```
y_pred=model.predict(x_test)
cm=confusion_matrix(y_test,y_pred)
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,model.predict(x_test)))
```

```
              precision    recall  f1-score   support

           0       0.25      0.50      0.33         2
           1       0.98      0.95      0.97        60

    accuracy                           0.94        62
   macro avg       0.62      0.72      0.65        62
weighted avg       0.96      0.94      0.95        62
```

accuracy

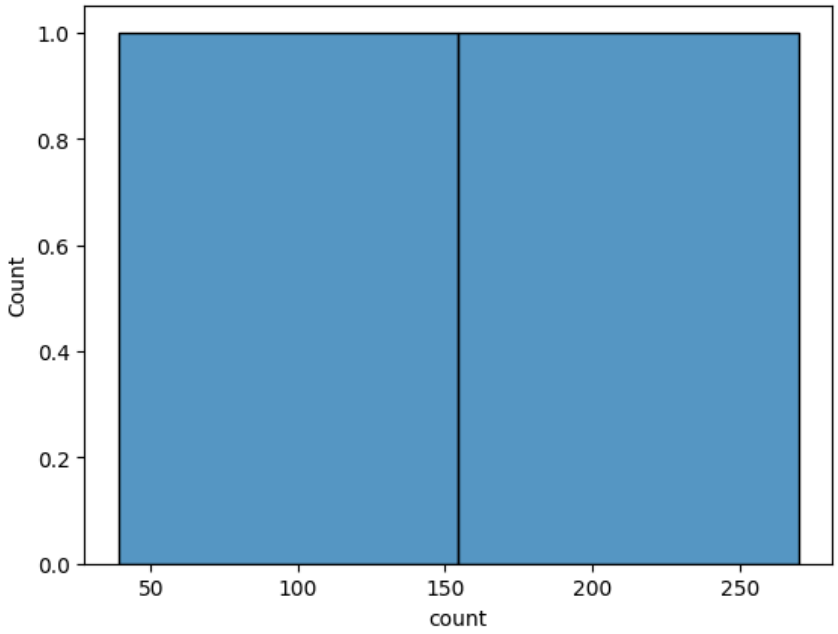
```
from sklearn.metrics import accuracy_score,classification_report
print("Accuracy:",accuracy_score(y_test,y_pred))
print ("\nclassification_report:\n",classification_report(y_test,y_pred))
#
```

```
Accuracy: 0.9354838709677419

classification_report:
               precision    recall  f1-score   support

           0       0.25      0.50      0.33         2
           1       0.98      0.95      0.97        60

    accuracy                           0.94        62
   macro avg       0.62      0.72      0.65        62
weighted avg       0.96      0.94      0.95        62
```

BIVARIENT

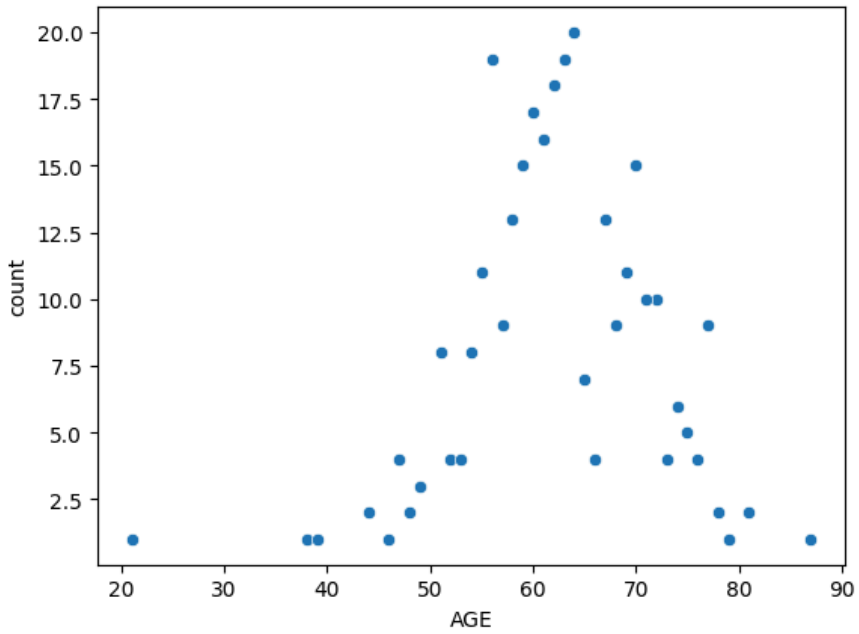
```
sns.histplot(df['LUNG_CANCER'].value_counts())
```

```
<Axes: xlabel='count', ylabel='Count'>
```



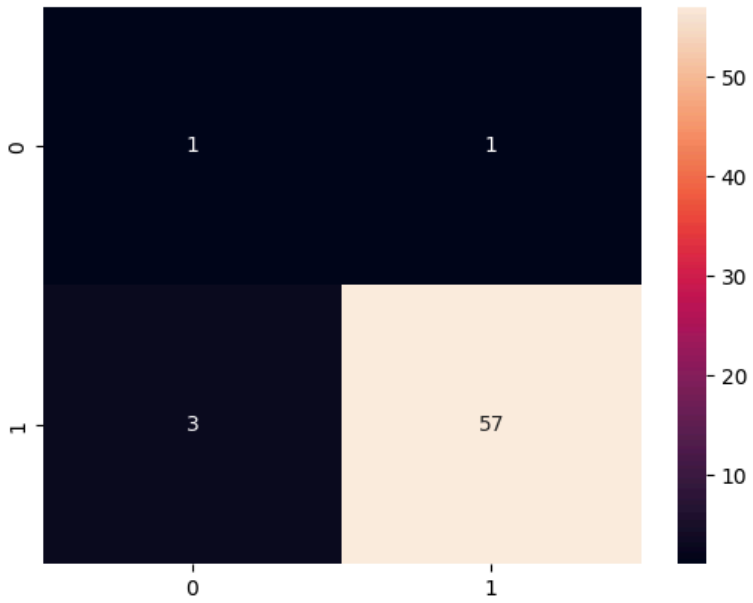

```
sns.scatterplot(df['AGE'].value_counts())
```

```
<Axes: xlabel='AGE', ylabel='count'>
```




```
sns.heatmap(cm,annot=True,fmt='g')
```

```
<Axes: >
```




```
sns.df([x='LUNG_CANCER',y='AGE'])
```

```
  File "/tmp/ipython-input-1099646399.py", line 1
    sns.df([x='LUNG_CANCER',y='AGE'])
           ^
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?
```

Next steps: Explain error

```
df.corr()*100
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATI
GENDER	100.000000	2.130644	3.627685	-21.295946	-15.212660	-27.556432	-20.460564	-8.356
AGE	2.130644	100.000000	-8.447456	0.520487	5.317036	1.868514	-1.264213	1.261
SMOKING	3.627685	-8.447456	100.000000	-1.458487	16.026698	-4.282232	-14.152231	-2.957
YELLOW_FINGERS	-21.295946	0.520487	-1.458487	100.000000	56.582929	32.308324	4.112218	-11.805
ANXIETY	-15.212660	5.317036	16.026698	56.582929	100.000000	21.684122	-0.967782	-18.853
PEER_PRESSURE	-27.556432	1.868514	-4.282232	32.308324	21.684122	100.000000	4.851481	7.814
CHRONIC								