Lab1

In [2]:
```
#Import dataset fev

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

```
C:\Users\admin\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarn
ing: A NumPy version >=1.16.5 and <1.23.0 is required for this version of
SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [3]:
```
df = pd.read_csv('fev.csv')
df.head()
```

Out[3]:

	Age	FEV	Height	Male	Smoke
0	9	1.708	57.0	0	0
1	8	1.724	67.5	0	0
2	7	1.720	54.5	0	0
3	9	1.558	53.0	1	0
4	9	1.895	57.0	1	0

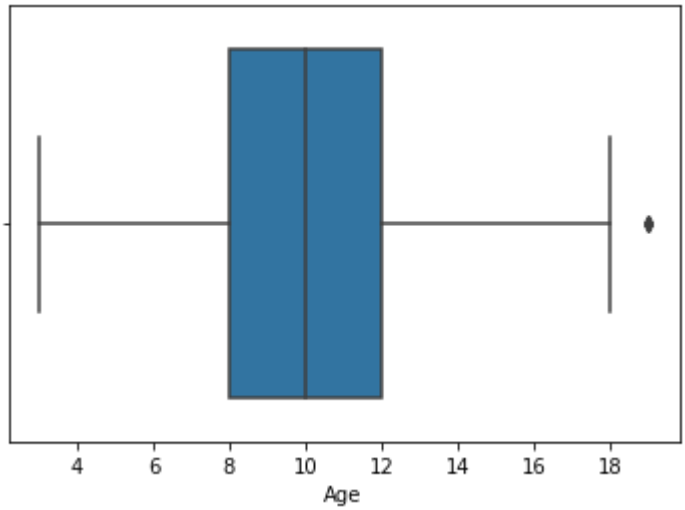
In [4]:
```
#Checking the missing values
df.isnull().sum()
```

Out[4]:
```
Age       0
FEV       0
Height    0
Male      0
Smoke     0
dtype: int64
```

In [5]:
```
# Checking for the outliers
sns.boxplot(df['Age'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

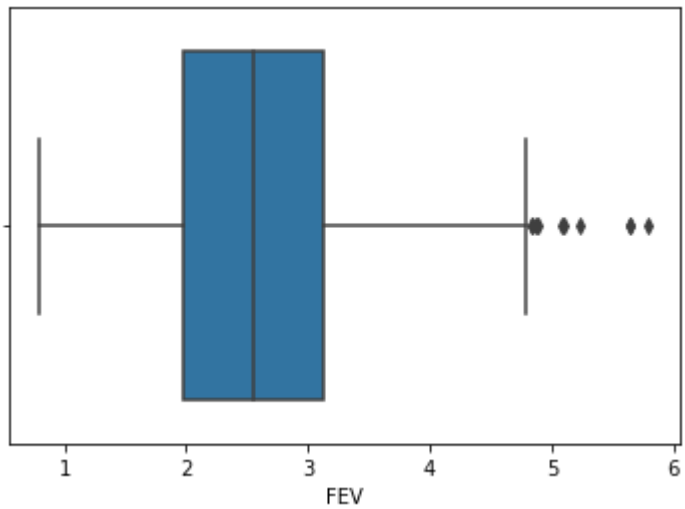
Out[5]: <AxesSubplot:xlabel='Age'>



In [6]:
```
sns.boxplot(df['FEV'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

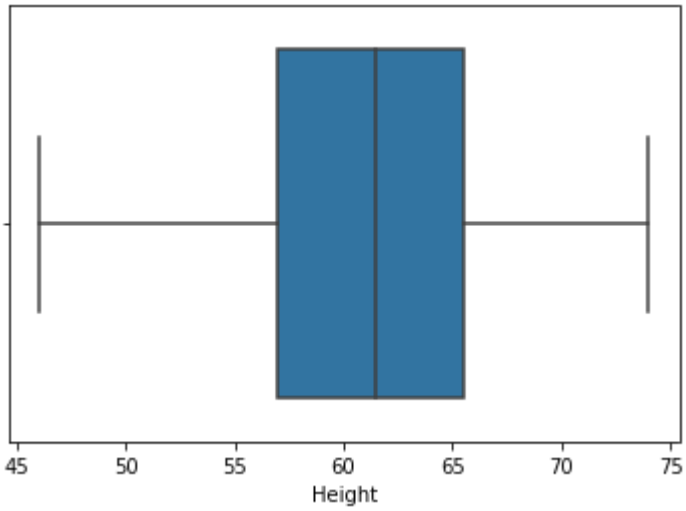
Out[6]: <AxesSubplot:xlabel='FEV'>



In [7]:
```
1 sns.boxplot(df['Height'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

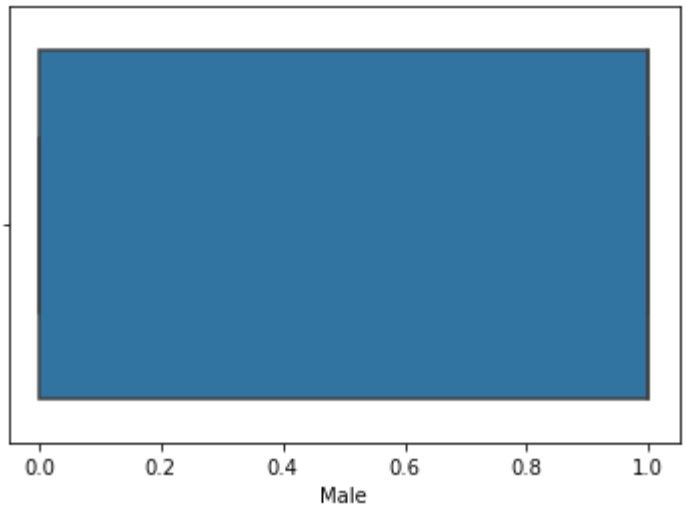
Out[7]: <AxesSubplot:xlabel='Height'>



In [8]:
```
1 sns.boxplot(df['Male'])
```

```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

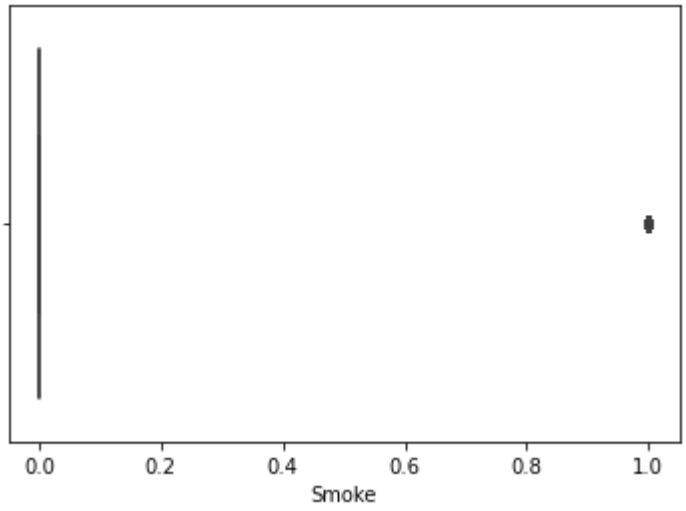
Out[8]: <AxesSubplot:xlabel='Male'>



In [9]:
```
1 sns.boxplot(df['Smoke'])
```

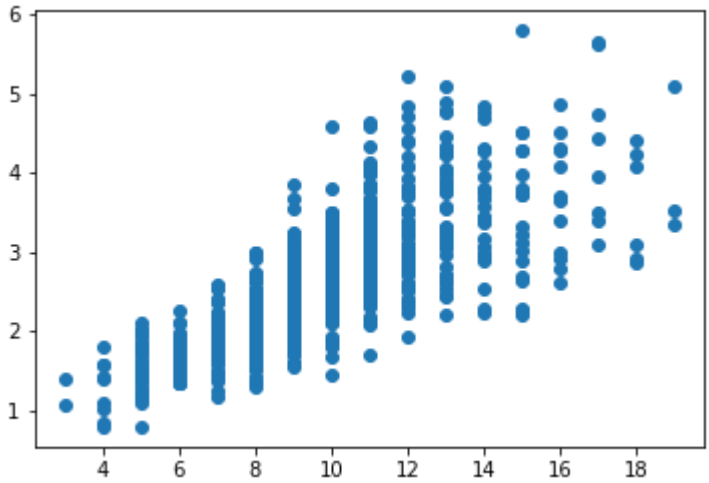
```
C:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: Futu
reWarning: Pass the following variable as a keyword arg: x. From version
0.12, the only valid positional argument will be `data`, and passing other
arguments without an explicit keyword will result in an error or misinterp
retation.
  warnings.warn(
```

Out[9]:
```
<AxesSubplot:xlabel='Smoke'>
```



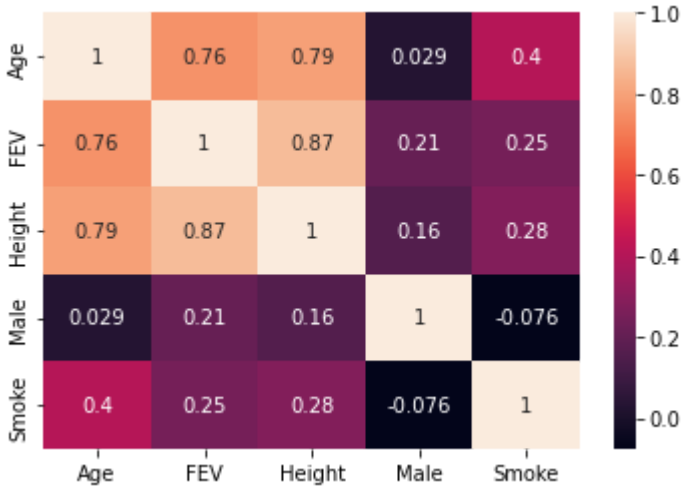
In [10]:
```
1 # Draw scatter diagram for FEV and Age
2
3 plt.scatter(df['Age'], df['FEV'])
```

Out[10]:
```
<matplotlib.collections.PathCollection at 0x25e2ae47d60>
```



In [11]:
```
1 sns.heatmap(df.corr(), annot = True)
```

Out[11]: <AxesSubplot:>



In [12]:
```
1 from sklearn.model_selection import train_test_split
```

In [13]:
```
1 from sklearn.linear_model import LinearRegression
```

In [14]:
```
1 x = df.iloc[:,0:1].values
2 y = df.iloc[:,1]
3 x.shape
```

Out[14]: (654, 1)

In [15]:
```
1 dv = 'FEV'
2 idv = ['age','height','male','smoke']
3 X = df.drop('FEV',axis=1)
4 Y = df['FEV']
```

In [16]:
```
1 from sklearn.metrics import mean_squared_error,r2_score,mean_absolute_e
2 X_train, X_test, Y_train, Y_test = train_test_split(X,Y,test_size =0.2,
```

In [17]:
```
1 model=LinearRegression()
2 model.fit(X_train,Y_train)
```

Out[17]: LinearRegression()

In [18]:
```
1 Y_pred = model.predict(X_test)
```

In [19]:

```
R_square = r2_score(Y_test,Y_pred)
R_square

percentage_explained_variance = R_square*100
print("percentage of EXplained variance :{:.2f}%".format(percentage_exp

mae =mean_absolute_error(Y_test,Y_pred)
print(mae)

mse =mean_squared_error(Y_test ,Y_pred)
print(mse)
```

```
percentage of EXplained variance :80.56%
0.2977478213196183
0.15898271151783314
```

Lab 2

In [1]:
```
#Import dataset fev

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

```
C:\Users\admin\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarn
ing: A NumPy version >=1.16.5 and <1.23.0 is required for this version of
SciPy (detected version 1.23.5
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:
```
df = pd.read_csv('House.csv')
df.head()
```

Out[2]:

	Unnamed: 0	Class	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V1
0	1	republican	n	y	n	y	y	y	n	n	n	y	NaN	y	y	
1	2	republican	n	y	n	y	y	y	n	n	n	n	n	y	y	
2	3	democrat	NaN	y	y	NaN	y	y	n	n	n	n	y	n	y	
3	4	democrat	n	y	y	n	NaN	y	n	n	n	n	y	n	y	
4	5	democrat	y	y	y	n	y	y	n	n	n	n	y	NaN	y	

In [3]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 435 entries, 0 to 434
Data columns (total 18 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   Unnamed: 0  435 non-null    int64 
 1   Class       435 non-null    object
 2   V1          423 non-null    object
 3   V2          387 non-null    object
 4   V3          424 non-null    object
 5   V4          424 non-null    object
 6   V5          420 non-null    object
 7   V6          424 non-null    object
 8   V7          421 non-null    object
 9   V8          420 non-null    object
 10  V9          413 non-null    object
 11  V10         428 non-null    object
 12  V11         414 non-null    object
 13  V12         404 non-null    object
 14  V13         410 non-null    object
 15  V14         418 non-null    object
 16  V15         407 non-null    object
 17  V16         331 non-null    object
dtypes: int64(1), object(17)
memory usage: 61.3+ KB
```

In [4]:
```
1 df['Class'].nunique()
```

Out[4]:
```
2
```

In [5]:
```
1 df['Class'].replace({'republican' : 1, 'democrat' : 0}, inplace = True)
```

In [6]:
```
1 df.isnull().sum()
```

Out[6]:
```
Unnamed: 0      0
Class           0
V1             12
V2             48
V3             11
V4             11
V5             15
V6             11
V7             14
V8             15
V9             22
V10             7
V11            21
V12            31
V13            25
V14            17
V15            28
V16           104
dtype: int64
```

In [7]:
```
print(df['V1'].mode())
df['V1'] = df['V1'].fillna('n')
```

```
0    n
dtype: object
```

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

Out[8]:
```
Unnamed: 0      0
Class           0
V1              0
V2             48
V3             11
V4             11
V5             15
V6             11
V7             14
V8             15
V9             22
V10             7
V11            21
V12            31
V13            25
V14            17
V15            28
V16           104
dtype: int64
```

In [9]:
```
print(df['V2'].mode())
df['V2'] = df['V2'].fillna('y')
```

```
0    y
dtype: object
```

In [10]:
```
df.isnull().sum()
```

Out[10]:
```
Unnamed: 0      0
Class           0
V1              0
V2              0
V3             11
V4             11
V5             15
V6             11
V7             14
V8             15
V9             22
V10             7
V11            21
V12            31
V13            25
V14            17
V15            28
V16           104
dtype: int64
```

In [11]:

```
print(df['V3'].mode())
print(df['V4'].mode())
print(df['V5'].mode())
print(df['V6'].mode())

print(df['V7'].mode())
print(df['V8'].mode())
print(df['V9'].mode())
print(df['V10'].mode())

print(df['V11'].mode())
print(df['V12'].mode())
print(df['V13'].mode())
print(df['V14'].mode())

print(df['V15'].mode())
print(df['V16'].mode())

```

```
0    y
dtype: object
0    n
dtype: object
0    y
dtype: object
0    y
dtype: object
0    y
dtype: object
0    y
dtype: object
0    y
dtype: object
0    y
dtype: object
0    n
dtype: object
0    n
dtype: object
0    y
dtype: object
0    y
dtype: object
0    n
dtype: object
0    y
dtype: object
```

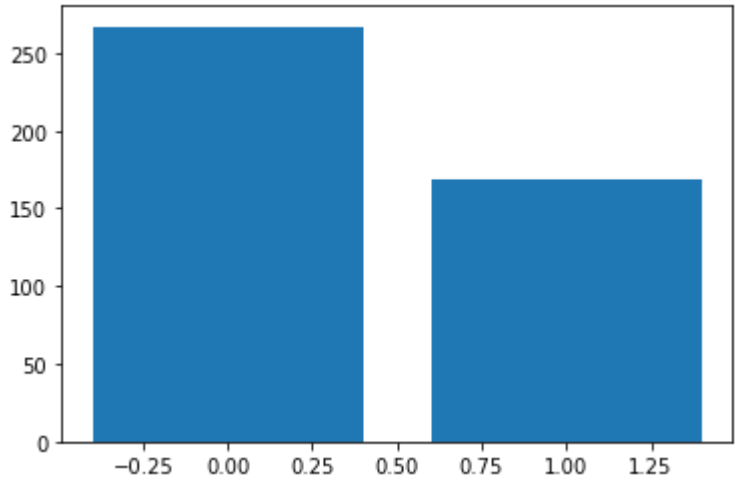
In [12]:
```
df['V3'] = df['V3'].fillna('y')
df['V4'] = df['V4'].fillna('n')
df['V5'] = df['V5'].fillna('y')
df['V6'] = df['V6'].fillna('y')

df['V7'] = df['V7'].fillna('y')
df['V8'] = df['V8'].fillna('y')
df['V9'] = df['V9'].fillna('y')
df['V10'] = df['V10'].fillna('y')

df['V11'] = df['V11'].fillna('n')
df['V12'] = df['V12'].fillna('n')
df['V13'] = df['V13'].fillna('y')
df['V14'] = df['V14'].fillna('y')

df['V15'] = df['V15'].fillna('n')
df['V16'] = df['V16'].fillna('y')
```

In [13]:
```
df.isnull().sum()
```

Out[13]:
```
Unnamed: 0    0
Class         0
V1            0
V2            0
V3            0
V4            0
V5            0
V6            0
V7            0
V8            0
V9            0
V10           0
V11           0
V12           0
V13           0
V14           0
V15           0
V16           0
dtype: int64
```

In [14]:
```
class_count= df['Class'].value_counts()
plt.bar(class_count.index,class_count.values)
```

Out[14]:
```
<BarContainer object of 2 artists>
```



In [19]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import MultinomialNB

Label_encoders={}
for column in ['V1','V2','V3','V4','V5','V6','V7','V8','V9','V10','V11'
    le = LabelEncoder()
    df[column] = le.fit_transform(df[column])
    Label_encoders[column] = le
```

In [20]:
```
X = df.drop('Class',axis=1)
Y = df['Class']
```

In [22]:
```
from sklearn.model_selection import train_test_split
```

In [23]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0
model = MultinomialNB()
model.fit(X_train, Y_train)
```

Out[23]:
```
MultinomialNB()
```

In [24]:
```
from sklearn.metrics import classification_report, accuracy_score
```

In [26]:
```
pred = model.predict(X_test)
pred
```

Out[26]:
```
array([1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1,
       0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1],
      dtype=int64)
```

In [27]:
```
from sklearn.metrics import confusion_matrix
c = confusion_matrix(Y_test, pred)
```

In [28]:
```
c
```

Out[28]:
```
array([[46,  7],
       [ 2, 32]], dtype=int64)
```

In [29]:
```
acc = accuracy_score(Y_test,pred)
acc
```

Out[29]:
```
0.896551724137931
```

In [31]:
```
print('Accuracy score: ', acc*100, '%')
```

```
Accuracy score:  89.65517241379311 %
```