In [1]:
```
#SWETHA JENIFER S-225229142-04/01/23
```

In [2]:
```
#STEP 1:
#PREPARED DATASET PIZZA AND PIZZA_TEST
```

In [3]:
```
#STEP 2:
#importing csv file
import pandas as pd
dat=pd.read_csv("pizza.csv")
```

In [4]:
```
#printing using head()
dat.head()
```

Out[4]:

	age	weight	likepizza
0	50	65	0
1	20	55	1
2	15	40	1
3	70	65	0
4	30	70	1

In [5]:
```
#printing shape
dat.shape
```

Out[5]:
```
(6, 3)
```

In [6]:
```
#printing columns
dat.columns
```

Out[6]:
```
Index(['age', 'weight', 'likepizza'], dtype='object')
```

In [7]:
```
#printing info
dat.info()
```

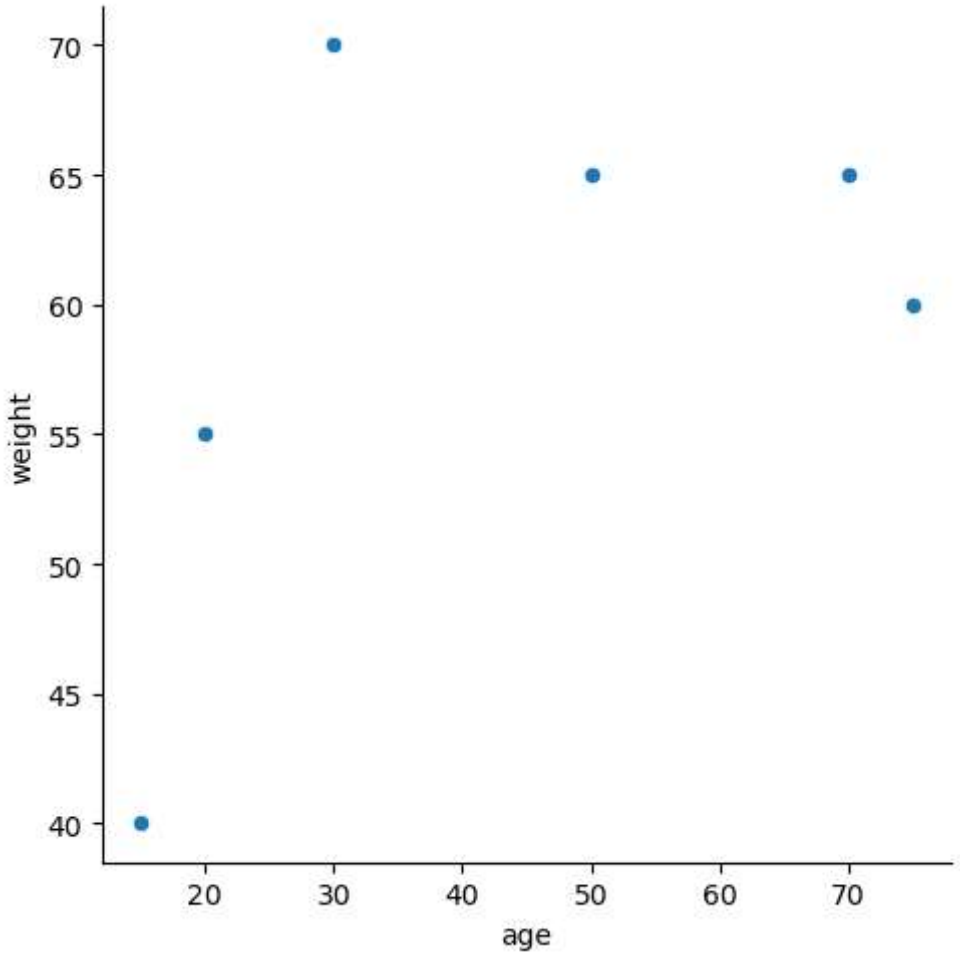
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        6 non-null      int64
 1   weight     6 non-null      int64
 2   likepizza  6 non-null      int64
dtypes: int64(3)
memory usage: 272.0 bytes
```

In [8]:
```
dat.info
```

Out[8]:
```
<bound method DataFrame.info of    age  weight  likepizza
0   50      65          0
1   20      55          1
2   15      40          1
3   70      65          0
4   30      70          1
5   75      60          0>
```

In [9]:
```
#importing packages
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.neighbors import KNeighborsClassifier as kn
```

In [10]:
```
#STEP 3:
#visualize relationships
sns.relplot(x='age',y='weight',data=dat,kind='scatter');
```



In [11]:
```
#STEP 4:
#prepare x matric and y vector
Fix = ["age","weight"]
x = dat[Fix]
```

In [12]:
```
#STEP 5:
#printing x
x
```

Out[12]:

	age	weight
0	50	65
1	20	55
2	15	40
3	70	65
4	30	70
5	75	60

In [13]:
```
y = dat.likepizza
```

In [14]:
```
#printing y
y
```

Out[14]:
```
0    0
1    1
2    1
3    0
4    1
5    0
Name: likepizza, dtype: int64
```

In [15]:
```
#printing type of x
type(x)
```

Out[15]:
```
pandas.core.frame.DataFrame
```

In [16]:
```
#printing type of y
type(y)
```

Out[16]:
```
pandas.core.series.Series
```

In [17]:
```
#STEP 6:
#MODEL BUILDING
piz_eat=kn(n_neighbors=2)
```

In [18]:
```
piz_eat.fit(x,y)
```

Out[18]:
```
KNeighborsClassifier(n_neighbors=2)
```

In [ ]:
```
#STEP 7:
#predicting if person will like pizza or not
piz_eat.predict(x)
```

In [ ]:
```
#will a person who is 25 years with weight 50 kgs like pizza or not?
pre = [[25,50]]
print(piz_eat.predict(pre))
```

In [ ]:
```
#will a person who is 60 years with weight 60 kgs like pizza or not?
pre =[[60,60]]
print(piz_eat.predict(pre))
```

In [19]:
```
#STEP 8:
#changing n_neighbors to 3
piz_eat1=kn(n_neighbors=3)
piz_eat1.fit(x,y)
```

Out[19]:
```
KNeighborsClassifier(n_neighbors=3)
```

In [ ]:
```
#will a person who is 25 years with weight 50 kgs like pizza or not?
pre=[[25,50]]
print(piz_eat1.predict(pre))
```

In [ ]:
```
#will a person who is 60 years with weight 60 kgs like pizza or not?
pre = [[60,60]]
print(piz_eat1.predict(pre))
```

In [22]:
```
#STEP 9:
#Predict on entire dataset
y_pred=piz_eat.predict
(x)
```

Out[22]:

	age	weight
0	50	65
1	20	55
2	15	40
3	70	65
4	30	70
5	75	60

In [20]:
```
#STEP 10:
#Accuracy funtion:
def accuracy(actual,pred):
    return sum(actual == pred) / float(actual.shape[0])
```

In [23]:
```
#STEP 11:
#calling accuracy funtion
accuracy(y,y_pred)
```

Out[23]:
```
0.0
```

In [24]:
```
#STEP 12:
#importing csv file
pizza1=pd.read_csv("pizza_test.csv")
```

In [25]:
```
#printing using head()
pizza1.head()
```

Out[25]:

	age	weight	likepizza
0	48	68	1
1	35	45	1
2	15	40	0
3	55	65	0

In [26]:
```
#printing shape
pizza1.shape
```

Out[26]:
```
(4, 3)
```

In [27]:
```
#printing columns
pizza1.columns
```

Out[27]:
```
Index(['age', 'weight', 'likepizza'], dtype='object')
```

In [28]:
```
#printing information
pizza1.info
```

Out[28]:
```
<bound method DataFrame.info of    age  weight  likepizza
0   48      68          1
1   35      45          1
2   15      40          0
3   55      65          0>
```

In [29]:
```
pizza1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4 entries, 0 to 3
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        4 non-null      int64
 1   weight     4 non-null      int64
 2   likepizza  4 non-null      int64
dtypes: int64(3)
memory usage: 224.0 bytes
```

In [30]:
```
Fix = ["age","weight"]
x = pizza1[Fix]
x
```

Out[30]:

	age	weight
0	48	68
1	35	45
2	15	40
3	55	65

In [31]:
```
y = pizza1.likepizza
y
```

Out[31]:
```
0    1
1    1
2    0
3    0
Name: likepizza, dtype: int64
```

In [32]:
```
pizza_eat=kn(n_neighbors=2)
```

In [33]:
```
pizza_eat.fit(x,y)
```

Out[33]:
```
KNeighborsClassifier(n_neighbors=2)
```

In [ ]:
```
pizza_eat.predict(x)
```

In [35]:
```
def accuracy(actual, pred):
    return sum(actual == pred) / (float(actual.shape[0]))
```

In [36]:
```
y_pred = pizza_eat.predict(x)
```

```
C:\Users\sweth\anaconda3\lib\site-packages\sklearn\neighbors\_classification.p
y:228: FutureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis
`), the default behavior of `mode` typically preserves the axis it acts along.
In SciPy 1.11.0, this behavior will change: the default value of `keepdims` wil
l become False, the `axis` over which the statistic is taken will be eliminate
d, and the value None will no longer be accepted. Set `keepdims` to True or Fal
se to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

In [37]:
```
accuracy(y,y_pred)
```

Out[37]:
```
0.5
```

In [ ]:
```
#STEP 13:
#finding best value for k
scores = []
for k in range(1,4):
    best=kn(n_neighbors=k)
    best.fit(x,y)
    y_pred=best.predict(x)
    acc = accuracy(y,y_pred)
    scores.append((k, acc))
scores
```

In [38]:
```
#STEP 14:
#importing accuracy score
#calling accuracy_score()
from sklearn.metrics import accuracy_score
accuracy_score(y,y_pred)
```

Out[38]:
```
0.5
```

In [ ]:
```

```