```
from sklearn.neighbors import KNeighbors classifier.
 model = kneighbors classifier (n_neighbors = 2)
   model.fit (x, y)
    (model.predict (x))
     data = [25, 50]
     model.predict ([data])
     data = [60, 60]
     model.predict ([data])
     models 3 = KNeighbors classifiers (n_neighbors = 3)
      models3 .fit (x, y)
      models3 = KNeighbors classifier (n_neighbors = 3)
     models3 .fit (x, y)
     (model 3 .predict (x))
    data 2 = [25, 50]
     model.3 .predict ([data 2])
    data 3 = [60, 60]
    model 3 .predict ([data 3])
    y_pred = model 3 .predict (x)
    y_pred
def .accuracy .(actual, pred):
  return sum (actual = = pred) / float (actual. shap[0])
print ("Accuracy:", accuracy (y, y_pred))
  pizz = pd.read_csv ("pizza_test.csv")
        pizz.head()      | df = pd.read_csv ("pizza-test.csv")
        pizz.tail()      | df
        pizz.shape ()    | df.info()
```

- Will a person who is 25 years with weight 50 kgs like Pizza or not? – The answer should be 1 (ie., YES)
- Will a person who is 60 years with weight 60 kgs like Pizza or not? – The answer should be 0 (ie., NO)

Step.8 [Change n_neighbors = 3]. Now, create new model, perform fit and predict steps. Check results for the above 2 queries. Are they same?

Step9. [Predict on entire dataset]. Now, perform prediction on entire X matrix and store result as y_pred.

Step10. [Accuracy function]. Create a ***function accuracy()*** and returns accuracy. The accuracy function can be defined as follows:

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

Step11. [Find accuracy]. Call accuracy() with **y and y_pred** as parameters and print accuracy score. Are you getting score as **1.0** ?

KNN predicted all samples in X correctly. Because, you created KNN model with those values of X and y. But, it may not be the same if you use different samples for testing.

Step12. [Prediction on Test Set]

Using Pandas, import **"pizza_test.csv"** file and print properties such as head(), shape, columns and info.

Using KNN model with n_neighbors=2, that you created previously, perform prediction on X values from pizza_test dataframe. Call accuracy function and print accuracy score. Are you getting a score of **0.5**? That is, our model has predicted 2 samples correctly and two wrongly, out of 4 samples in the test set.

Step13. [Find best value for k]. If you want to improve the accuracy of your model, then you should use the best value k for the nearest neighbors.

Now, let us rerun our KNN for various values of k (k = 1 to 3) and find accuracy scores. Then, we will be able to select the best k for which accuracy score is the highest.

```
scores = list()

for k in range(1, 4):
   create knn model
   perform fit on X and y
   predict test set X
   find accuracy on y values of test set and predicted values
   store k and accuracy as a tuple in scores list
```

When you print scores list, you will get the following output.

```
[(1, 0.5), (2, 0.5), (3, 0.5)]
```

Step14. [accuracy_score function]. Call accuracy_score() function with y_test and y_pred values. You can import as *"from sklearn.metrics import accuracy_score"*.

CONCLUSION

Our accuracy values remain same for all values of k. Because, our dataset is so small. When working on larger datasets, your will find accuracy improving on various values of k.

```
res = df. Columns
Point (res)
Sns.relplot (x="age", y="weight", hue="like pizza",
                        data=pizz)
sns.relplot (x="age", y= "weight", data=pizz,
                        kind="scatter")
    rk= pizz.dropna(axis=0)
     rk.
    y= pizz. like pizza
    daubt= ["age", "weight"]
     x=pizz [datt]
      X
      Y.
      X. dtypes
      X. dtypes.
from sklearn. neighbors. import kneighbors Classifier.
   piz= KNeighbors Classi~~fiear~~(n-neighbors=2)
   piz s/. fit (X,Y)
  (pix1. predict(X))
   def accuracy(actual, pred):
    return sum (actual==pred / float (actual.
                                                shape[0])
```

NOTES

```
y-pred = piz-predict (x)
accuracy ( y, y-pred)

scores = [ ]
for k in range (1,H) :
  best = kneighborclassifier ( n-neighbors = k)
  best . fit ( x, y)
   best fit predict ( y)
   y-predict = best . predict ( x)
    acc = accuracy ( y , y-predict)
   scores. append ((k, acc))
 scores.
from sklearn. matrices import accuracy_score.
  accuracy - score ( y, y-pred)
```

```
import pandas .as pd.
 import csv.
  piz = pd. read- csv ("pizza.csv")
   piz = head ()
   piz .tail ()
    piz . Shape
   df=pd.read_csv("pizza.csv")
   df
    df. info()
     res = df. columns
     print(res)

 import pandas as pd
 import matplotlib. pyplot asplt.
  import seaborm as 'sns.
   import numpy as np.
sns. relplot ( x= "age ", y= "weight", hue = "like pizza",
                                  data=piz)
    y= piz. like pizza
   dat = ['age', 'weight']
     x = piz [dat]
      x
      y
      x. dtypes
      y. dtypes
pip install . sklearn.
```
Scanned with CamScanner

Lab2. Pizza Liking Prediction using kNN

Objectives
In this lab, you will build a k-Nearest-Neighbour model to predict whether a person will like pizza or not based on his age and weight.

Learning Outcomes
After completing this lab, you will

- Identify features (aka variables – dependent and independent) and create dataset
- Import dataset and understand properties such as size, data types and others
- Use sklearn and instantiate kNN model
- Perform fit and predict operations
- Compute accuracy
- Select the best value for K

Business Use Case
You are a Data Scientist. A local Pizza Restaurant in your city has approached you with the details of its customers such as age and weight and whether they liked their Pizza or not. The restaurant wanted you to help them to develop a system that will predict whether a customer will like their Pizzas given their age and weight. The restaurant has given you the following dataset for assessing your predictive analytics skills.

Step1. [Prepare your dataset]
Using MS Excel, create the following CSV file and save it as, **"pizza.csv"**

age	weight	likePizza
50	65	0
20	55	1
15	40	1
70	65	0
30	70	1
75	60	0

Create another CSV, as below, but with age and weight columns only and save it as, **"pizza_test.csv"**.

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

Step2. [Import dataset]. Using Pandas, import **"pizza.csv"** file and print properties such as head(), shape, columns and info.

Step3. [Visualize Relationships]. Plot relplot between "age" and "weight", with hue as "likePizza"

Step4. [Prepare X matrix and y vector]. Extract "age" and "weight" columns and store into new dataframe **X**. Similarly, extract "likePizza" column and store into **y**.

Step5. [Examine X and y]. Print X, y, type of X and type of y.

Step6. [Model building]. Create **KNeighborsClassifier(n_neighbors=2)** from **sklearn** and perform **fit** on X and y.

Step7. [Model testing]. Using your KNN model, predict if a person will like Pizza or not.