CSL 487 Introduction to Data Science LAB

SEMESTER PROJECT

Maximum Marks: 30

Submission Due Date: 7th July, 2022

Sr.no	Name	Enrolment	Semester
1	**Ammar Anis**	**02-134192-073**	**6**
2	**Shaheer Suleman**	**02-134192-009**	**6**

Name	Designation
DR KASHIF HUSSAIN	**Course Instructor**
Ms. SALAS AKBAR	**Lab Engineer**

Contents

1. Chapter 1

1.1. Problem Statement

We decided to create a product analysis system for our project because it seemed the most relevant; we use a product analysis system in multiple instances when product hunting. be drops hipping, Amazon, Daraz, and so on Using all of this as inspiration, we decided to create a product analysis system, and then narrowed it down to something we wanted to make. an Amazon product recommendation system that would recommend products based on a large data set and provide the final output based on different preferences, giving the user a variety of recommendations to choose from, taking into account various variables such as categories, price, rating, rank, year, and much more. Because the recommendation is an example of a classification, model, we will be using K-Nearest-Neighbor or KNN algorithm to find our output.

Chapter 2

2.2. Literature Review

E-commerce implementation is a critical process for an organization's success and benefit. As a result, many researchers have studied intensive research works in the area of e-Commerce implementation from a variety of perspectives and findings. However, the aspect of Business-to-Customer (B2C) e-Commerce implementation has yet to be undertaken in a comprehensible manner in the context of a full life cycle of information system development prior to implementation, during implementation, and after implementation. Sixty-five (65) primary research studies were chosen and analysed based on the implementation phase theme, research approach, and research area. According to the findings, the majority of papers (49 percent) discussed B2C e-Commerce in the pre-implementation phase and used a quantitative approach (63 percent) as the most popular research method. Two more phases of implementation The existing literature does not adequately address the implementation phase and post-implementation phase. Furthermore, existing literature has addressed important aspects of e-Commerce implementation but has not been mapped to the appropriate implementation phases. These factors remain on the surface without a clear direction in which e-Commerce implementation is critical for organizations. Thus, based on a systematic literature review, this study concludes that more research is required to fully comprehend the complex process of e-Commerce implementation. Furthermore, research is needed to screen these factors to a specific and more accurate "map" in the process of e-Commerce implementation.

2. Chapter 3

2.1. Methodology

As a starting step, look at what you are trying to solve within a business. The first step to that is understanding the business -- what is the business dealing with, what is their input, what is the final output given by the business, and what are the other factors that lead to the final output. With this information, you get a clear understanding of the business.

Data requirements and data collection are within the second step of the methodology. Here, you define the data that you need to solve the problem that the business is facing. In this step, you must look at the format of data that needs to be collected as well as what particular data should be collected. For example, a data scientist working in a banking sector use case that predicts loan eligibility needs to collect data that includes an individual's monthly income, profession, age, and years in the country to predict if the individual is eligible to get loan. After the data has been collected, you begin understanding and preparing the data, which is the third step within the methodology. Next is generating and evaluating models is the next step in the methodology. There are two types of modeling, descriptive modeling and predictive modeling. Descriptive modeling tells you what service a particular individual might be interested in and is based on recommendation systems and clustering algorithms. Predictive modeling is where you predict a future value based on key inputs and is based on linear or logistic regression and classification algorithms.

Chapter 4

4.1. Code Snippet

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

```
df=pd.read_csv('/content/sample_data/mks of urinal for men
UK(7).csv') df.head()
```

	phrase	Search Volume	Relevancy	Seller1	Seller2	Seller3	Seller4	Seller5	Sell
0	balance board	61	NaN	3	5	>306	1	6	
1	balanceboard	52	NaN	6	1	>306	2	26	
2	balanceboards	315	NaN	2	3	>306	1	5	
3	balance board erwachsene	217	NaN	4	3	>306	1	22	
4	balance boards	194	NaN	6	14	>306	1	2	

```
df = df[['Search
Volume','Seller1','Seller2','Seller3','Seller4','Seller5','Seller6','Sell df =
df[df != '-'] df = df[df != '>306'] df = df[df != 'N/R'] df.fillna(0, inplace
= True) df
```

	Search Volume	Seller1	Seller2	Seller3	Seller4	Seller5	Seller6	Seller7	Seller8
0	61	3	5	0	1	6	2	8	11
1	52	6	1	0	2	26	8	4	10

2 3152 3 0 1 5 4 12 18

3 2174 3 0 1 22 7 6 8 **4** 1946 14 0 1 2 7 4 18

... ...
...

144 160 0 0 4 0 0 0 0 0

145 230 0 0 4 0 0 0 0 0

146 60 0 0 0 0 5 0 0 0

147 379 0 0 5 0 0 0 0 0

148 111 0 6 0 0 0 0 0 0

1 [redacted]

```
df= df.astype(float)
```

```
for i in range(1, 10):   df['Relevancy'] = np.where((df['Seller'+str(i)] < 16)
& (df['Seller'+str(i)] > 0), df['R
```

```
#df['Relevancy'] = np.where((df['Seller1'] < 16) & (df['Seller1'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller2'] < 16) & (df['Seller2'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller3'] < 16) & (df['Seller3'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller4'] < 16) & (df['Seller4'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller5'] < 16) & (df['Seller5'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller6'] < 16) & (df['Seller6'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller7'] < 16) & (df['Seller7'] > 0),
df['Relevancy'] +
# df['Relevancy'] = np.where((df['Seller8'] < 16) & (df['Seller8'] > 0),
df['Relevancy'] +
   # df['Relevancy'] = np.where((df['Seller9'] < 16) & (df['Seller9'] > 0),
    df['Relevancy'] + # df['Relevancy'] = np.where((df['Seller10'] < 16) &
                    (df['Seller10'] > 0), df['Relevancy']
# df
```

Search
Seller1 Seller2 Seller3 Seller4 Seller5 Seller6 Seller7 Seller8 Volume

0	61.0	3.0	5.0	0.0	1.0	6.0	2.0	8.0	11.0
1	52.0	6.0	1.0	0.0	2.0	26.0	8.0	4.0	10.0
2	315.0	2.0	3.0	0.0	1.0	5.0	4.0	12.0	18.0
3	217.0	4.0	3.0	0.0	1.0	22.0	7.0	6.0	8.0 **4** 194.0 6.0 14.0 0.0 1.0 2.0 7.0 4.0 18.0
	...	...	...	...	...	...	...	...	...
144	160.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0
145	230.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0
146	60.0	0.0	0.0	0.0	0.0	5.0	0.0	0.0	0.0
147	379.0	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0
148	111.0	0.0	6.0	0.0	0.0	0.0	0.0	0.0	0.0

1 [redacted]

```
df.drop(df[(df['Relevancy'] < 3)].index,axis =
0,inplace=True) df
```

	Search Volume	Seller1	Seller2	Seller3	Seller4	Seller5	Seller6	Seller7	Seller8	S
0	61.0	3.0	5.0	0.0	1.0	6.0	2.0	8.0	11.0	
1	52.0	6.0	1.0	0.0	2.0	26.0	8.0	4.0	10.0	
2	315.0	2.0	3.0	0.0	1.0	5.0	4.0	12.0	18.0	
3	217.0	4.0	3.0	0.0	1.0	22.0	7.0	6.0	8.0	
4	194.0	6.0	14.0	0.0	1.0	2.0	7.0	4.0	18.0	
5	94.0	2.0	15.0	0.0	3.0	5.0	1.0	4.0	22.0	
6	340.0	1.0	14.0	0.0	4.0	5.0	3.0	2.0	20.0	
7	164.0	2.0	10.0	0.0	3.0	8.0	7.0	12.0	25.0	
8	105.0	1.0	7.0	0.0	2.0	17.0	4.0	3.0	39.0	
9	151.0	4.0	3.0	0.0	1.0	24.0	7.0	13.0	22.0	
10	158.0	22.0	21.0	0.0	7.0	12.0	2.0	1.0	34.0	
11	141.0	3.0	2.0	0.0	18.0	7.0	1.0	5.0	29.0	
12	3437.0	2.0	22.0	31.0	4.0	8.0	1.0	6.0	19.0	
13	131.0	2.0	16.0	0.0	1.0	3.0	9.0	4.0	22.0	

14	839.0	1.0	10.0	0.0	4.0	9.0	14.0	3.0	28.0
15	213.0	1.0	13.0	0.0	7.0	3.0	17.0	2.0	5.0
16	160.0	5.0	4.0	0.0	1.0	22.0	6.0	33.0	15.0
17	110.0	5.0	17.0	0.0	1.0	12.0	8.0	7.0	28.0
18	67.0	1.0	21.0	0.0	3.0	14.0	11.0	23.0	26.0
19	351.0	2.0	22.0	0.0	5.0	20.0	6.0	21.0	25.0
20	141.0	2.0	14.0	0.0	3.0	8.0	12.0	6.0	42.0
21	88.0	1.0	18.0	0.0	2.0	3.0	22.0	4.0	30.0
22	119.0	2.0	3.0	0.0	1.0	27.0	17.0	16.0	44.0
23	59.0	4.0	9.0	93.0	3.0	10.0	1.0	6.0	15.0
24	253.0	1.0	16.0	0.0	6.0	13.0	11.0	10.0	40.0
25	50.0	28.0	7.0	0.0	32.0	3.0	2.0	20.0	1.0
26	589.0	1.0	20.0	0.0	2.0	5.0	25.0	22.0	21.0
27	68.0	16.0	22.0	0.0	9.0	19.0	11.0	29.0	41.0
28	88.0	12.0	47.0	0.0	11.0	14.0	17.0	29.0	24.0
29	219.0	1.0	11.0	0.0	15.0	13.0	21.0	34.0	2.0
30	62.0	5.0	1.0	0.0	3.0	31.0	13.0	74.0	26.0
31	101.0	1.0	24.0	0.0	17.0	7.0	27.0	31.0	12.0
32	160.0	8.0	10.0	0.0	2.0	5.0	60.0	7.0	22.0
34	101.0	11.0	22.0	0.0	14.0	19.0	23.0	21.0	50.0
36	108.0	7.0	6.0	0.0	8.0	21.0	16.0	82.0	44.0
37	111.0	19.0	1.0	0.0	22.0	6.0	20.0	56.0	4.0
38	1058.0	2.0	4.0	226.0	7.0	8.0	1.0	3.0	12.0
39	118.0	12.0	11.0	0.0	54.0	1.0	15.0	64.0	9.0
40	355.0	27.0	1.0	0.0	5.0	24.0	74.0	22.0	2.0
41	76.0	2.0	6.0	213.0	4.0	17.0	7.0	9.0	12.0
45	249.0	20.0	3.0	0.0	63.0	12.0	21.0	0.0	7.0
46	77.0	0.0	25.0	0.0	0.0	10.0	9.0	0.0	0.0

```
      48    119.0       2.0       11.0        0.0       28.0       17.0       26.0       12.0        23.0
#CKWS
totalrows = 50332.0len(df) 1.0   7.0        0.0       42.0       18.0       59.0       20.0        92.0

      54    116.0      13.0        5.0        0.0       70.0       35.0       19.0       48.0         1.0
x = len(df.loc[(df['Seller1'] < 16) & (df['Seller1'] > 0)])
x = ((x/totalrows)*6268.0     100)3.0     9.0       0.0       36.0       13.0       18.0         0.0
              0.0

x2 = len70(df.loc[(df76.0['Seller2'0.0   ] < 4.016) & (df0.0['Seller2'0.0 ] > 0)])7.0   0.0
   0.0 13.0 x2 = ((x2/totalrows)*100)

x3 = len(df.loc[(df['Seller3'] < 16) & (df['Seller3'] > 0)]) x3
= ((x3/totalrows)*100)

x4 = len(df.loc[(df['Seller4'] < 16) & (df['Seller4'] > 0)]) x4
= ((x4/totalrows)*100)

x5 = len(df.loc[(df['Seller5'] < 16) & (df['Seller5'] > 0)]) x5
= ((x5/totalrows)*100)

x6 = len(df.loc[(df['Seller6'] < 16) & (df['Seller6'] > 0)]) x6
= ((x6/totalrows)*100)

x7 = len(df.loc[(df['Seller7'] < 16) & (df['Seller7'] > 0)]) x7
= ((x7/totalrows)*100)

x8 = len(df.loc[(df['Seller8'] < 16) & (df['Seller8'] > 0)]) x8
= ((x8/totalrows)*100)

x9 = len(df.loc[(df['Seller9'] < 16) & (df['Seller9'] > 0)]) x9
= ((x9/totalrows)*100)

x10 = len(df.loc[(df['Seller10'] < 16) & (df['Seller10'] > 0)]) x10
= ((x10/totalrows)*100)

df1=pd.read_csv('/content/sample_data/train_test_data.csv') data_frame
= pd.DataFrame([x, x2, x3, x4, x5, x6, x7, x8, x9, x10]) df1['CKWS'] =
data_frame
```

```
#CVS
totalCVS = df['Search Volume'].sum()

CVS = df.loc[(df['Seller1'] < 16) & (df['Seller1'] > 0)] x
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller2'] < 16) & (df['Seller2'] > 0)] x2
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller3'] < 16) & (df['Seller3'] > 0)] x3
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller4'] < 16) & (df['Seller4'] > 0)] x4
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller5'] < 16) & (df['Seller5'] > 0)] x5
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller6'] < 16) & (df['Seller6'] > 0)] x6
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller7'] < 16) & (df['Seller7'] > 0)] x7
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller8'] < 16) & (df['Seller8'] > 0)] x8
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller9'] < 16) & (df['Seller9'] > 0)] x9
= ((CVS['Search Volume'].sum()/totalCVS)*100)

CVS = df.loc[(df['Seller10'] < 16) & (df['Seller10'] > 0)] x10
= ((CVS['Search Volume'].sum()/totalCVS)*100)

data_frame = pd.DataFrame([x, x2, x3, x4, x5, x6, x7, x8, x9, x10]) df1['CSV']
= data_frame
```

```
x = df1.sum(axis = 1) df1.loc[x > 150,
'Viable'] = 'Low' df1.loc[((x <= 150) & (x
> 100)), 'Viable'] = 'Medium' df1.loc[x <=
100, 'Viable'] = 'High'
df1.to_csv('/content/sample_data/train_tes
t_data.csv') df1
/usr/local/lib/python3.7/dist-
packages/ipykernel_launcher.py:1:
FutureWarning: Dropp   """Entry point for
launching an IPython kernel.
```

Sellers	CKWS	CSV Viable

0	Seller1	82.978723	90.669603	Low
1	Seller2	70.212766	54.171764	Medium
2	Seller3	0.000000	0.000000	High
3	Seller4	74.468085	87.293043	Low
4	Seller5	65.957447	[illegible]	Medium

```
df2 = pd.read_csv('/content/sample_data/Final Output.csv')
df2
```

	Sellers	CKWS	CSV	Viable
0	Seller1	97.222222	99.572795	Low
1	Seller2	22.222222	72.683140	High
2	Seller3	38.888889	89.897568	Medium
3	Seller4	13.888889	46.128453	High
4	Seller5	66.666667	27.748920	High
...	...	...	...	...
491	Seller6	15.828292	92.487205	Medium
492	Seller7	66.842396	30.174558	High
493	Seller8	83.553085	25.134668	Medium
494	Seller9	76.824917	95.468479	Low
495	Seller10	32.644893	20.162299	High

496 rows × 4 columns

```
df2.loc[df2['Viable'] == 'High',
'Categorical_Viable'] = '0' df2.loc[df2['Viable']
== 'Medium', 'Categorical_Viable'] = '1'
df2.loc[df2['Viable'] == 'Low',
'Categorical_Viable'] = '2' df2 =
df2[['CKWS','CSV','Categorical_Viable']] df2=
df2.astype(float) df2
```

	CKWS	CSV	Categorical_Viable
0	97.222222	99.572795	2.0
1	22.222222	72.683140	0.0
2	38.888889	89.897568	1.0

3	13.888889	46.128453	0.0
4	66.666667	27.748920	0.0
...	...	...	...
491	15.828292	92.487205	1.0
492	66.842396	30.174558	0.0
493	83.553085	25.134668	1.0
494	76.824917	95.468479	2.0
495	32.644893	20.162299	0.0

496 rows × 3 columns

```
x = df2[['CKWS','CSV']]
y = df2['Categorical_Viable']
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.3)
from sklearn.metrics import
classification_report,confusion_matrix,accuracy_score
y_train.shape
```

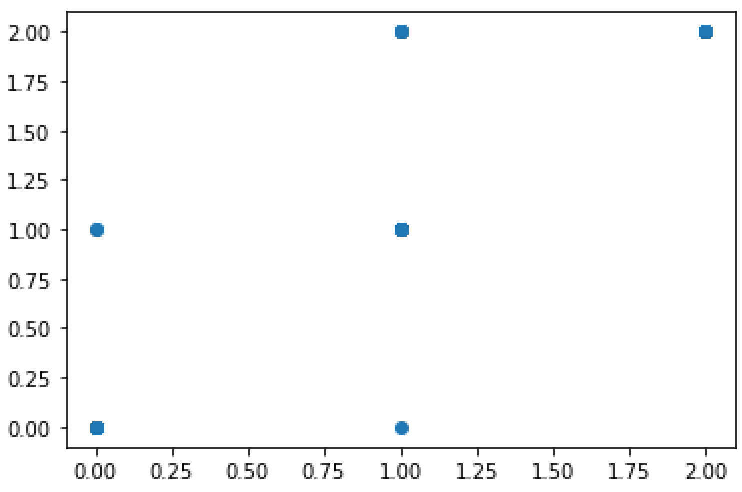
(347,)

```
#Apply KNN from sklearn.neighbors
import KNeighborsClassifier knn =
(KNeighborsClassifier(n_neighbors =
40)) knn.fit(x_train,y_train)
knn.score(x_test,y_test)
```

0.9463087248322147

```
predict =
knn.predict(x_test)
plt.scatter(predict,
y_test)
```

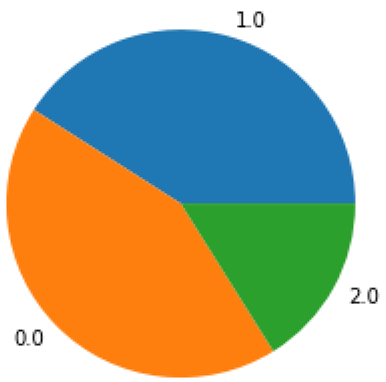
<matplotlib.collections.PathCollection at 0x7ff21e1ca190>



```
from collections
import Counter
counter_object =
Counter(y_test)
keyst =
counter_object.keys(
) valuest =
counter_object.value
s() keyst, valuest

# print(num_values)
```

```
(dict_keys([1.0, 0.0, 2.0]), dict_values([61,
                                          64, 24]))
```

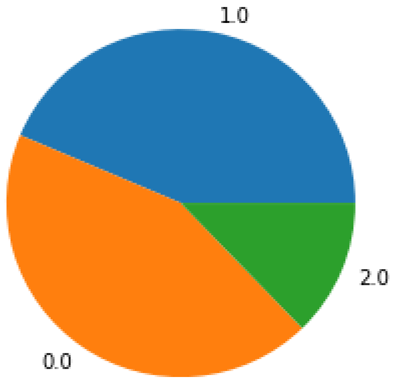
```
#KNN testing data
plt.pie( valuest,
labels=keyst )
plt.show()
```




```
from collections
import Counter
counter_object =
Counter(predict)
keysk =
counter_object.keys()
valuesk =
counter_object.values
() keysk, valuesk #
print(num_values)
```

```
(dict_keys([1.0, 0.0, 2.0]), dict_values([65,
                                          65, 19]))
```

```
#KNN predict
plt.pie( valuesk,
labels=keysk )
plt.show()
```




```
print(classification_report(y_test,predict))
print(classification_report(y_train,knn.predict(x_train)))
```

```
                    precision    recall  f1-score   support

         0.0        0.97       0.98       0.98
   64             1.0        0.91       0.97
  0.94           61          2.0        1.00
          0.79       0.88         24

    accuracy                              0.95
149     macro avg        0.96       0.91       0.93
149 weighted avg         0.95       0.95       0.95
149                  precision    recall  f1-score
support

         0.0        0.99       0.99       0.99
   169            1.0        0.93       0.99
  0.96         130           2.0        1.00
          0.81       0.90         48

    accuracy                           0.97
347      macro avg         0.97        0.93
0.95           347  weighted avg       0.97
0.97        0.97         347
```

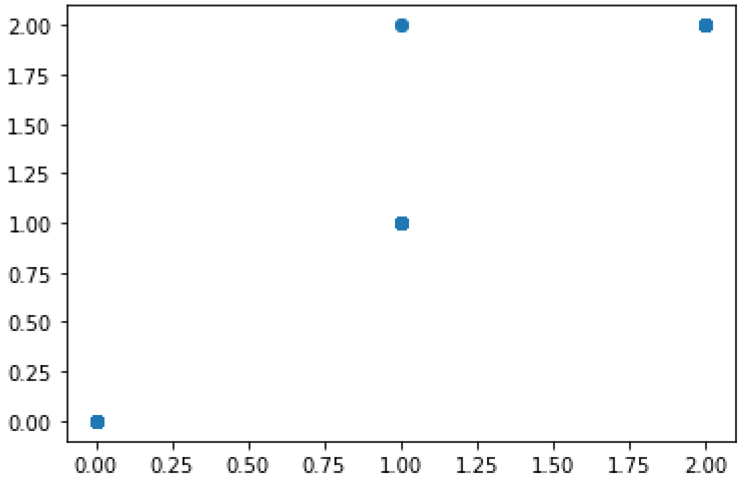
```
#Apply Logistic Regression from
sklearn.linear_model import
```

```
LogisticRegression lr =
LogisticRegression()
lr.fit(x_train,y_train)
```

```
LogisticRegression()
```

```
predict1 =
lr.predict(x_test)
plt.scatter(predict1
, y_test)
```

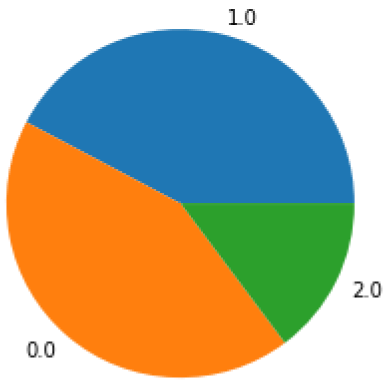
```
<matplotlib.collections.PathCollection at 0x7ff21e322790>
```




```
from collections
import Counter
counter_object =
Counter(predict1)
keysl =
counter_object.keys()
valuesl =
counter_object.values(
) keysl, valuesl

# print(num_values)
```

```
(dict_keys([1.0, 0.0, 2.0]), dict_values([63, 64, 22]))
```

```
#Logistic Regression
Predict plt.pie(
valuesl,
labels=keysl )
plt.show()
```




```
print(classification_report(y_test,predict1))
print(classification_report(y_train,lr.predict(x_train)))
```

```
                   precision    recall  f1-score   support

         0.0       1.00      1.00      1.00
  64           1.0       0.97      1.00
  0.98           61          2.0       1.00
         0.92       0.96        24

    accuracy                           0.99
149     macro avg       0.99      0.97      0.98
149 weighted avg       0.99      0.99      0.99
149                precision    recall  f1-score
support

         0.0       1.00      0.99      1.00
  169           1.0       0.99      1.00
  1.00         130          2.0       1.00
         1.00       1.00        48

    accuracy                         1.00
347     macro avg         1.00       1.00
1.00         347  weighted avg       1.00
1.00       1.00        347
```

```
#Apply Naive Bayes

from sklearn.naive_bayes import
MultinomialNB classifier =
MultinomialNB().fit(x_train,y_t
rain)
```

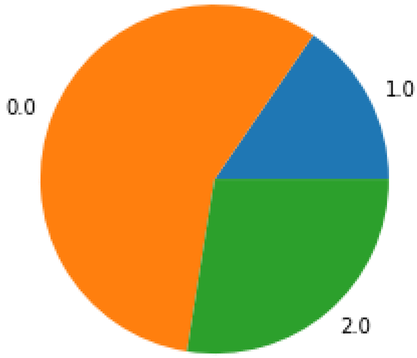
```
predict2 = classifier.predict(x_test)
print(classification_report(y_test,predict2))
```

```
                 precision    recall  f1-score

     support          0.0       0.48      0.64

                                0.55        64



            1.0       0.30      0.11      0.17
        61          2.0       0.07      0.12
                   0.09        24

        accuracy                          0.34
     149     macro  avg         0.29      0.29
     0.27         149  weighted  avg      0.34
     0.34      0.32       149
```

```
from collections
import Counter
counter_object =
Counter(predict2)
keysn =
counter_object.keys()
valuesn =
counter_object.values(
) keysn, valuesn

# print(num_values)
```

```
(dict_keys([1.0, 0.0, 2.0]), dict_values([23,
                                   85, 41]))
```

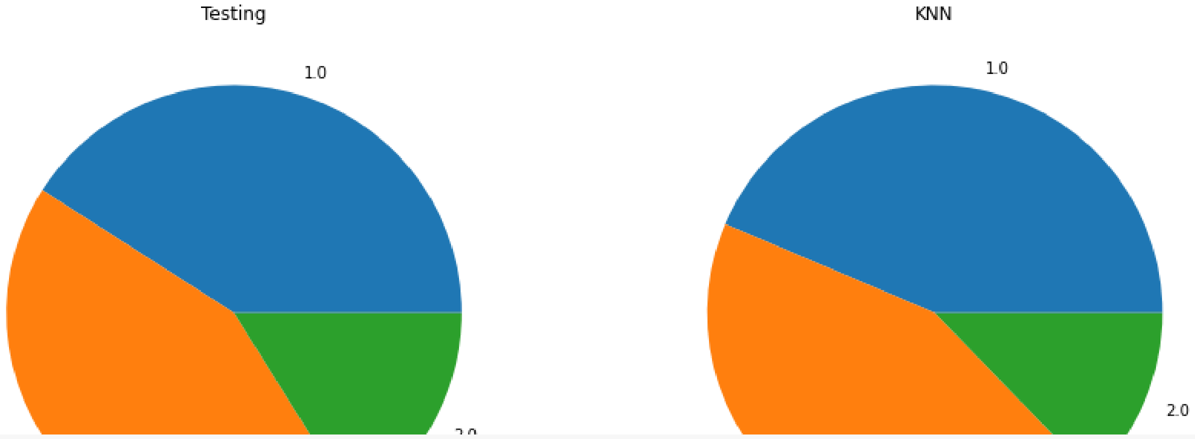
```
#Naive bayes Predict
plt.pie( valuesn,
labels=keysn )
plt.show()
```




```
plt.pie( valuest,
labels=keyst )
plt.show()
print('Testing')
plt.pie( valuesk,
labels=keysk )
plt.show()
print('KNN')
plt.pie( valuesl,
labels=keysl )
plt.show()
print('Logistic')
plt.pie( valuesn,
labels=keysn )
plt.show()
print('Naive Bayes')
```

```
# plt.figure(figsize=(20, 20))

figure, axis = plt.subplots(2, 2,figsize=(15,15))

# For Sine Function
axis[0, 0].pie( valuest, labels=keyst ) axis[0,
0].set_title("Testing")

# For Cosine Function
axis[0, 1].pie( valuesk, labels=keysk ) axis[0,
1].set_title("KNN")

# For Tangent Function
axis[1, 0].pie( valuesl, labels=keysl ) axis[1,
0].set_title("Logistic")

# For Tanh Function
axis[1, 1].pie( valuesn, labels=keysn )
axis[1, 1].set_title("Naive bayes") plt.show
```

<function matplotlib.pyplot.show>



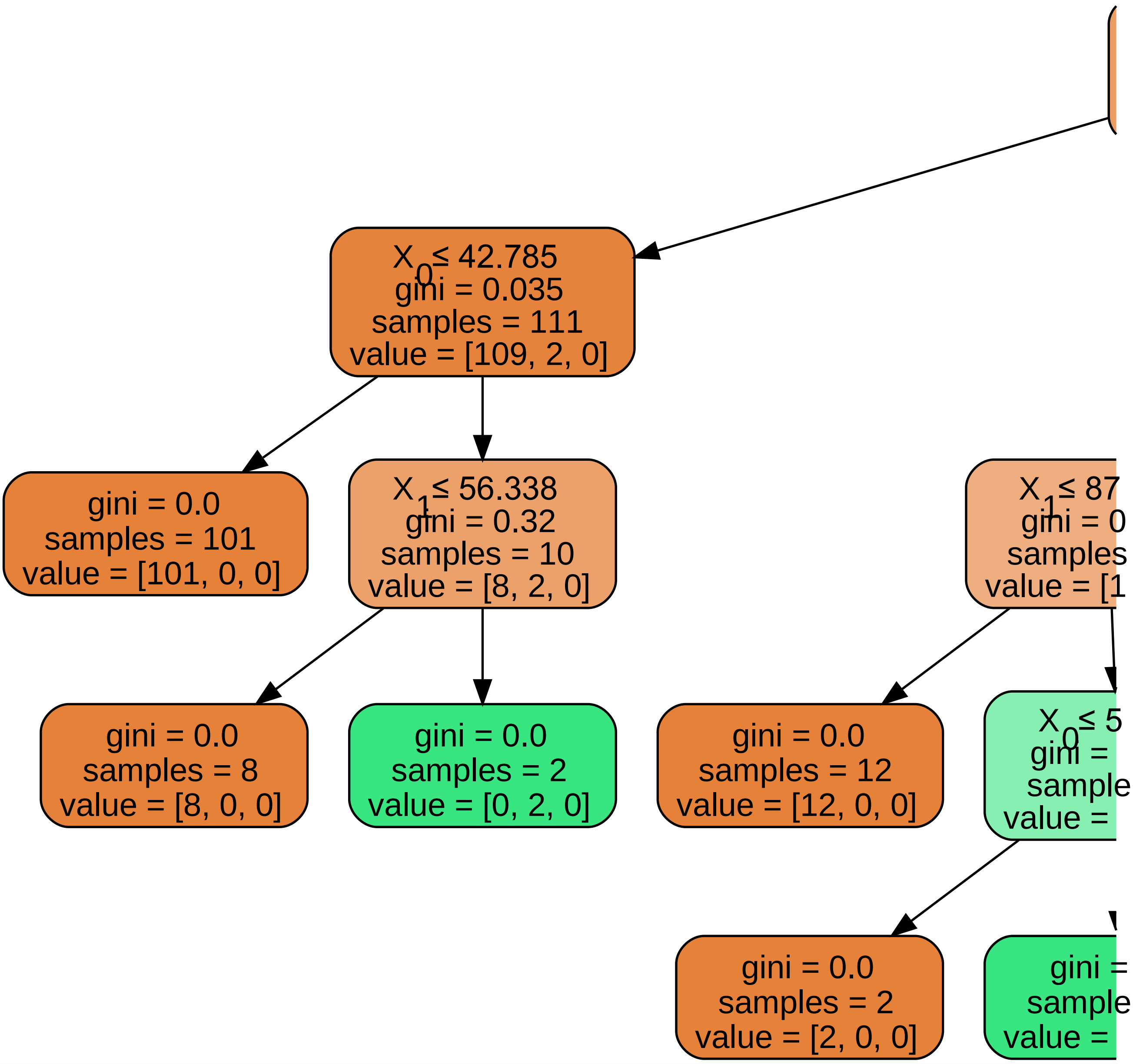

```
#training file data

from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(x_train, y_train)
```

```
#predicting data
predict3=clf.predict(x_test)
```

```
#accuracy of decision tree
clf.score(x_test,y_test), clf.score(x_train,y_train)
```

(0.9463087248322147, 1.0)

```
#data visualization
import graphviz
dot_data = tree.export_graphviz(clf, out_file=None,
                     filled=True, rounded=True,
                     special_characters=True)
graph = graphviz.Source(dot_data)
graph.view()
graph
```




```
#KNN accuracy: 97%
#Logistic accuracy: 100%
#Naive Bayes accuracy: 34%
#decisiontree accuracy: 95%

print(knn.predict([[90,80]]))
print(lr.predict([[90,80]]))
print(classifier.predict([[90,80]]))
print(clf.predict([[90,80]]))
```

```
[2.]
[2.]
[0.]
[2.]
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not
  "X does not have valid feature names, but"
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not
  "X does not have valid feature names, but"
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not
  "X does not have valid feature names, but"
/usr/local/lib/python3.7/dist-packages/sklearn/base.py:451: UserWarning: X does not
  "X does not have valid feature names, but"
```

0s completed at 9:48 PM

Chapter 5

Conclusion ……………………….

E-commerce remains one of the business methods that can benefit if done correctly; even if the stock market and commodities fell, E-Commerce was able to survive and receive high transaction volumes. In the course of our business in Malaysia, e-commerce presents a tremendous opportunity. Furthermore, it is to introduce new techniques and styles in a transaction. It is actually much better to use the extensive E-Commerce in the Internet world to bring the goodness of the individual or the state.

E-commerce has undeniably grown in importance in our society. Companies that take E-Commerce seriously and devote sufficient resources to its development will be the most successful in the future. E-commerce is a whole-business endeavour, not just an IT issue. Companies that use it as an excuse to completely redesign their business processes stand to benefit the most. Furthermore, E-Commerce is a useful technology that provides consumers with access to businesses and companies all over the world.