VIT-AP UNIVERSITY, ANDHRA PRADESH

Academic year: 2022-2023 **Branch/ Class:** B.Tech
Semester: Fall **Date:** 16-12-2021
Faculty Name: Dr.Aravapalli Rama Satish **School:** SCOPE

NAME:Majjiga Jaswanth REGNO:20BCD7171

****

1. Load the data from the following link:
https://archive.ics.uci.edu/ml/datasets/Facebook+Live+Sellers+in+Thailand

```
[1] import numpy as np
    import pandas as pd
```

```
myData = pd.read_csv("Live_20210128.csv")
print(myData.head())
```

```
   status_id status_type status_published  num_reactions  num_comments  \
0          1       video   4/22/2018 6:00            529           512
1          2       photo  4/21/2018 22:45            150             0
2          3       video   4/21/2018 6:17            227           236
3          4       photo   4/21/2018 2:29            111             0
4          5       photo   4/18/2018 3:22            213             0

   num_shares  num_likes  num_loves  num_wows  num_hahas  num_sads  \
0         262        432         92         3          1         1
1           0        150          0         0          0         0
2          57        204         21         1          1         0
3           0        111          0         0          0         0
4           0        204          9         0          0         0

   num_angrys  Column1  Column2  Column3  Column4
0           0      NaN      NaN      NaN      NaN
1           0      NaN      NaN      NaN      NaN
2           0      NaN      NaN      NaN      NaN
3           0      NaN      NaN      NaN      NaN
4           0      NaN      NaN      NaN      NaN
```

2. Display the number of rows and columns of loaded data set.

```
rows = len(myData.axes[0])
cols = len(myData.axes[1])
print(rows)
print(cols)
```

```
7050
16
```

3. Display summary of the loaded data set.

```
[4] myData.describe()
```

	status_id	num_reactions	num_comments	num_shares	num_likes	num_loves
count	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000	7050.000000
mean	3525.500000	230.117163	224.356028	40.022553	215.043121	12.728652
std	2035.304031	462.625309	889.636820	131.599965	449.472357	39.972930
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1763.250000	17.000000	0.000000	0.000000	17.000000	0.000000
50%	3525.500000	59.500000	4.000000	0.000000	58.000000	0.000000
75%	5287.750000	219.000000	23.000000	4.000000	184.750000	3.000000
max	7050.000000	4710.000000	20990.000000	3424.000000	4710.000000	657.000000

4. Count the number of missing values are there for each column and drop the columns which have more number of null values approximately equal to the number of rows of data.

```
[5] myData.isna()
```

	status_id	status_type	status_published	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num_sads	n
0	False	False	False	False	False	False	False	False	False	False	False	
1	False	False	False	False	False	False	False	False	False	False	False	
2	False	False	False	False	False	False	False	False	False	False	False	
3	False	False	False	False	False	False	False	False	False	False	False	
4	False	False	False	False	False	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	
7045	False	False	False	False	False	False	False	False	False	False	False	
7046	False	False	False	False	False	False	False	False	False	False	False	
7047	False	False	False	False	False	False	False	False	False	False	False	
7048	False	False	False	False	False	False	False	False	False	False	False	
7049	False	False	False	False	False	False	False	False	False	False	False	

7050 rows × 16 columns

```
myData.dropna(axis=1,how='all')
```

	status_id	status_type	status_published	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num_sads	n
0	1	video	4/22/2018 6:00	529	512	262	432	92	3	1	1	
1	2	photo	4/21/2018 22:45	150	0	0	150	0	0	0	0	
2	3	video	4/21/2018 6:17	227	236	57	204	21	1	1	0	
3	4	photo	4/21/2018 2:29	111	0	0	111	0	0	0	0	
4	5	photo	4/18/2018 3:22	213	0	0	204	9	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	
7045	7046	photo	9/24/2016 2:58	89	0	0	89	0	0	0	0	
7046	7047	photo	9/23/2016 11:19	16	0	0	14	1	0	1	0	
7047	7048	photo	9/21/2016 23:03	2	0	0	1	1	0	0	0	
7048	7049	photo	9/20/2016 0:43	351	12	22	349	2	0	0	0	
7049	7050	photo	9/10/2016 10:30	17	0	0	17	0	0	0	0	

7050 rows × 12 columns

```
myData.dropna(axis=1,inplace=True)
```

```
[46] print(myData.isnull().sum())
```

```
status_id           0
status_type         0
status_published    0
num_reactions       0
num_comments        0
num_shares          0
num_likes           0
num_loves           0
num_wows            0
num_hahas           0
num_sads            0
num_angrys          0
dtype: int64
```

5.List out number of categorical variables and numerical variables are available in the data set.

```
numerics = ['int16', 'int32', 'int64']
data = myData.select_dtypes(include=numerics)
data
```

	status_id	num_reactions	num_comments	num_shares	num_likes	num_loves	num_wows	num_hahas	num_sads	num_angrys
0	1	529	512	262	432	92	3	1	1	0
1	2	150	0	0	150	0	0	0	0	0
2	3	227	236	57	204	21	1	1	0	0
3	4	111	0	0	111	0	0	0	0	0
4	5	213	0	0	204	9	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
7045	7046	89	0	0	89	0	0	0	0	0
7046	7047	16	0	0	14	1	0	1	0	0
7047	7048	2	0	0	1	1	0	0	0	0
7048	7049	351	12	22	349	2	0	0	0	0
7049	7050	17	0	0	17	0	0	0	0	0

7050 rows × 10 columns

```
data=myData.select_dtypes(exclude=numerics)
data
```

	status_type	status_published
0	video	4/22/2018 6:00
1	photo	4/21/2018 22:45
2	video	4/21/2018 6:17
3	photo	4/21/2018 2:29
4	photo	4/18/2018 3:22
...	...	...
7045	photo	9/24/2016 2:58
7046	photo	9/23/2016 11:19
7047	photo	9/21/2016 23:03
7048	photo	9/20/2016 0:43
7049	photo	9/10/2016 10:30

7050 rows × 2 columns

6.display statistical summary of numerical attributes.

```
print(data.isnull().sum())
```

```
status_type         0
status_published    0
dtype: int64
```

```
data.describe()
```

	status_type	status_published
count	7050	7050
unique	4	6913
top	photo	3/20/2018 1:54
freq	4288	3

7.Count the number of unique values are existed in categorical attributes and drop the all columns which have number of unique values approximately equal to number of rows of

data.

Code:

```
duplicated_data = data[data.duplicated() == True]

duplicated_data
```

	status_type	status_published
339	photo	3/20/2017 21:46
387	photo	2/9/2017 3:00
494	photo	12/4/2016 9:58
626	photo	8/25/2016 4:31
797	photo	4/12/2016 8:31
...	...	...
6311	video	5/19/2018 9:14
6524	photo	1/13/2018 1:48
6574	photo	12/25/2017 7:50
6776	photo	9/16/2017 7:55
6949	video	7/25/2017 5:05

118 rows × 2 columns

8. Convert the remaining categorical variable into equivalent numerical form and perform Min-Max Scaling for all attributes.

9. Develop a python program to perform K-Means Clustering with number of clusters as K = 2, 3, 4, 5, 6 and display inter-cluster distances and sum of all inter-cluster distances. Identify at
what K value the inter-cluster distance is minimal.

```
import pandas as pd
import numpy as np
import random as rd
import matplotlib.pyplot as plt

myData = pd.read_csv("Live_20210128.csv")
print(myData.head())

diff = 1
```

```
j=0

while(diff!=0):

XD=X

i=1

for index1,row_c in Centroids.iterrows():

ED=[]

for index2,row_d in XD.iterrows():

d1=(row_c["ApplicantIncome"]-row_d["ApplicantIncome"])**2

d2=(row_c["LoanAmount"]-row_d["LoanAmount"])**2

d=np.sqrt(d1+d2)

ED.append(d)

X[i]=ED

i=i+1

C=[]

for index,row in X.iterrows():

min_dist=row[1]

pos=1

for i in range(K):

if row[i+1] < min_dist:

min_dist = row[i+1]

pos=i+1

C.append(pos)

X["Cluster"]=C

Centroids_new = X.groupby(["Cluster"]).mean()[["LoanAmount","ApplicantIncome"]]

if j == 0:

diff=1

j=j+1

else:

diff = (Centroids_new['LoanAmount'] - Centroids['LoanAmount']).sum() +
(Centroids_new['ApplicantIncome'] - Centroids['ApplicantIncome']).sum()

print(diff.sum())

Centroids = X.groupby(["Cluster"]).mean()[["LoanAmount","ApplicantIncome"]]
```

```
color=['blue','green','cyan']
for k in range(K):
data=X[X["Cluster"]==k+1]
plt.scatter(data["ApplicantIncome"],data["LoanAmount"],c=color[k])
plt.scatter(Centroids["ApplicantIncome"],Centroids["LoanAmount"],c='red')
plt.xlabel('Income')
plt.ylabel('Loan Amount (In Thousands)')
plt.show()
```