assignment 3

September 16, 2021

##

Kunal Garg

##

101903683

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
df1 = pd.read_csv("dataset/AWCustomers.csv")
df2 = pd.read_csv("dataset/AWSales.csv")
```

[3]:
```
df = pd.merge(df1, df2, on='CustomerID')
```

[4]:
```
df.head()
```

[4]:

	CustomerID	Title	FirstName	MiddleName	LastName	Suffix	\
0	21173	NaN	Chad	C	Yuan	NaN	
1	13249	NaN	Ryan	NaN	Perry	NaN	
2	29350	NaN	Julia	NaN	Thompson	NaN	
3	13503	NaN	Theodore	NaN	Gomez	NaN	
4	22803	NaN	Marshall	J	Shan	NaN	

	AddressLine1	AddressLine2	City	StateProvinceName	…	\
0	7090 C. Mount Hood	NaN	Wollongong	New South Wales	…	
1	3651 Willow Lake Rd	NaN	Shawnee	British Columbia	…	
2	1774 Tice Valley Blvd.	NaN	West Covina	California	…	
3	2103 Baldwin Dr	NaN	Liverpool	England	…	
4	Am Gallberg 234	NaN	Werne	Nordrhein-Westfalen	…	

	Gender	MaritalStatus	HomeOwnerFlag	NumberCarsOwned	NumberChildrenAtHome	\
0	M	M	1	3	0	
1	M	M	1	2	1	
2	F	S	0	3	0	
3	M	M	1	2	1	
4	M	S	1	1	0	

```
  TotalChildren YearlyIncome LastUpdated  BikeBuyer  AvgMonthSpend
0             1        81916  2017-03-06          1          50.97
1             2        81076  2017-03-06          1          53.11
2             0        86387  2017-03-06          1          54.08
3             2        61481  2017-03-06          1          56.93
4             0        51804  2017-03-06          1          55.41

[5 rows x 26 columns]
```

[5]:
```
df.dtypes
```

[5]:
```
CustomerID               int64
Title                   object
FirstName               object
MiddleName              object
LastName                object
Suffix                  object
AddressLine1            object
AddressLine2            object
City                    object
StateProvinceName       object
CountryRegionName       object
PostalCode              object
PhoneNumber             object
BirthDate               object
Education               object
Occupation              object
Gender                  object
MaritalStatus           object
HomeOwnerFlag            int64
NumberCarsOwned          int64
NumberChildrenAtHome     int64
TotalChildren            int64
YearlyIncome             int64
LastUpdated             object
BikeBuyer                int64
AvgMonthSpend          float64
dtype: object
```

0.1 Part I

(a)

[6]:
```
df.describe()
```

[6]:
```
          CustomerID  HomeOwnerFlag  NumberCarsOwned  NumberChildrenAtHome  \
count   18361.000000   18361.000000     18361.000000          18361.000000
mean    20241.987092       0.610588         1.270301              0.338162
std      5336.332731       0.487630         0.913989              0.568957
min     11000.000000       0.000000         0.000000              0.000000
25%     15621.000000       0.000000         1.000000              0.000000
50%     20248.000000       1.000000         1.000000              0.000000
75%     24863.000000       1.000000         2.000000              1.000000
max     29483.000000       1.000000         5.000000              3.000000

        TotalChildren   YearlyIncome     BikeBuyer  AvgMonthSpend
count    18361.000000   18361.000000  18361.000000   18361.000000
mean         0.850389   72754.779642      0.551767      51.766744
std          0.927315   30686.014313      0.497326       3.437684
min          0.000000   25435.000000      0.000000      44.100000
25%          0.000000   53312.000000      0.000000      49.410000
50%          0.000000   61851.000000      1.000000      51.420000
75%          2.000000   87410.000000      1.000000      53.590000
max          3.000000  139115.000000      1.000000      65.290000
```

(b)

[7]:
```
# correlation matrix doesn't account for string based attributes
# so I selected those attributes that were relevant by analysis
selected_columns =␣
 ↪['CustomerID','HomeOwnerFlag','NumberCarsOwned','NumberChildrenAtHome','YearlyIncome','Bike
```

[8]:
```
df = df[selected_columns].copy()
```

(c)

[9]:
```
df.dtypes
```

[9]:
```
CustomerID              int64
HomeOwnerFlag           int64
NumberCarsOwned         int64
NumberChildrenAtHome    int64
YearlyIncome            int64
BikeBuyer               int64
AvgMonthSpend         float64
Education              object
Occupation             object
Gender                 object
MaritalStatus          object
dtype: object
```

0.2 Part II

(a) Handling Null values

[10]:
```
pd.isna(df).any()
```

[10]:
```
CustomerID              False
HomeOwnerFlag           False
NumberCarsOwned         False
NumberChildrenAtHome    False
YearlyIncome            False
BikeBuyer               False
AvgMonthSpend           False
Education               False
Occupation              False
Gender                  False
MaritalStatus           False
dtype: bool
```

(b) Normalization

[11]:
```
d = df.copy()
m = d['YearlyIncome'].min()
M = d['YearlyIncome'].max()
d['YearlyIncome'] = (d['YearlyIncome']-m)/(M-m)
```

[12]:
```
df['YearlyIncome'] = d['YearlyIncome']
```

[13]:
```
df['YearlyIncome']
```

[13]:
```
0        0.496842
1        0.489453
2        0.536172
3        0.317083
4        0.231958
           …
18356    0.242065
18357    0.312781
18358    0.232442
18359    0.543121
18360    0.980128
Name: YearlyIncome, Length: 18361, dtype: float64
```

(c) Discretization (Binning) on Continuous attributes or Categorical Attributes with too many different values

[14]:
```
bins = np.linspace(min(d['AvgMonthSpend']), max(d['AvgMonthSpend']),4)
labels = ['low','medium','high']
d['bins'] = pd.cut(d['AvgMonthSpend'], bins=bins, labels=labels,␣
 ↪include_lowest=True)
```

[15]:
```
d.bins
```

[15]:
```
0           low
1        medium
2        medium
3        medium
4        medium
          …
18356       low
18357       low
18358       low
18359    medium
18360    medium
Name: bins, Length: 18361, dtype: category
Categories (3, object): ['low' < 'medium' < 'high']
```

[16]:
```
df['bins'] = d['bins']
```

(d) Standardization/Normalization (Z_score)

[17]:
```
std = d['NumberCarsOwned'].std()
mean = d['NumberCarsOwned'].mean()
d['NumberCarsOwned'] = (d['NumberCarsOwned']-mean)/std
```

[18]:
```
df['NumberCarsOwned'] = d['NumberCarsOwned']
df['NumberCarsOwned']
```

[18]:
```
0        1.892473
1        0.798367
2        1.892473
3        0.798367
4       -0.295738
           …
18356   -0.295738
18357    0.798367
18358    0.798367
18359   -1.389843
18360    0.798367
Name: NumberCarsOwned, Length: 18361, dtype: float64
```

(e) Binarization (One Hot Encoding)

[19]:
```
d.NumberChildrenAtHome.value_counts()
```

[19]:
```
0    13038
1     4468
2      824
```

```
3       31
Name: NumberChildrenAtHome, dtype: int64
```

[20]:
```
children = pd.get_dummies(d.NumberChildrenAtHome, prefix='Children')
df = pd.concat([df, children], axis=1)
```

[21]:
```
df = df.drop(labels=['NumberChildrenAtHome'], axis=1)
df
```

[21]:

	CustomerID	HomeOwnerFlag	NumberCarsOwned	YearlyIncome	BikeBuyer	\
0	21173	1	1.892473	0.496842	1	
1	13249	1	0.798367	0.489453	1	
2	29350	0	1.892473	0.536172	1	
3	13503	1	0.798367	0.317083	1	
4	22803	1	-0.295738	0.231958	1	
…	…	…	…	…	…	
18356	25414	0	-0.295738	0.242065	0	
18357	11459	0	0.798367	0.312781	1	
18358	12160	0	0.798367	0.232442	1	
18359	14353	0	-1.389843	0.543121	0	
18360	16676	1	0.798367	0.980128	1	

	AvgMonthSpend	Education	Occupation	Gender	MaritalStatus	\
0	50.97	Bachelors	Clerical	M	M	
1	53.11	Partial College	Clerical	M	M	
2	54.08	Bachelors	Clerical	F	S	
3	56.93	Partial College	Skilled Manual	M	M	
4	55.41	Partial College	Skilled Manual	M	S	
…	…	…	…	…	…	
18356	48.14	Graduate Degree	Skilled Manual	F	M	
18357	50.23	Bachelors	Skilled Manual	F	S	
18358	49.56	Partial College	Skilled Manual	F	S	
18359	51.42	High School	Clerical	F	S	
18360	52.27	Graduate Degree	Professional	F	M	

	bins	Children_0	Children_1	Children_2	Children_3
0	low	1	0	0	0
1	medium	0	1	0	0
2	medium	1	0	0	0
3	medium	0	1	0	0
4	medium	1	0	0	0
…	…	…	…	…	…
18356	low	1	0	0	0
18357	low	1	0	0	0
18358	low	1	0	0	0
18359	medium	1	0	0	0
18360	medium	1	0	0	0

```
[18361 rows x 15 columns]
```

0.3 Part III

(a)

[23]:
```
def jaccard_similarity(list1, list2):
    s1 = set(list1)
    s2 = set(list2)
    return float(len(s1.intersection(s2)) / len(s1.union(s2)))
jaccard_similarity(df.YearlyIncome, df.BikeBuyer)
```

[23]:
```
0.00013024225058609013
```

[24]:
```
from scipy.spatial.distance import cosine
print(1 - cosine(df["YearlyIncome"], df["BikeBuyer"]))
```

```
0.7141098793231326
```

(b)

[25]:
```
r = np.corrcoef(df.YearlyIncome, df.BikeBuyer)
r
```

[25]:
```
array([[1.        , 0.24945696],
       [0.24945696, 1.        ]])
```