```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mlxtend.frequent_patterns import apriori
from mlxtend.frequent_patterns import association_rules
from mlxtend.preprocessing import TransactionEncoder
from sklearn.preprocessing import MinMaxScaler
import sklearn.cluster as cluster
import sklearn.metrics as metrics
from tabulate import tabulate

#Question 5

#a
df = pd.read_csv("D:/ML Assignments SEM 1/Assignment 2/Groceries.csv")
df.describe()

itemList = df.groupby(['Customer'])['Item'].apply(list).values.tolist()
iL_count = df.groupby(['Customer'])['Item'].count()

te = TransactionEncoder()
te_ary = te.fit(itemList).transform(itemList)
indicators = pd.DataFrame(te_ary, columns=te.columns_)


freqItems = apriori(indicators, min_support = (75/len(iL_count)),use_colnames=True)
k_value = len(freqItems['itemsets'][len(freqItems)-1])
print("Total number of Itemsets : ",len(freqItems))
print("Largest Value of K is : ",k_value)


#5b
associationRule = association_rules(freqItems, metric = 'confidence',min_threshold
= 0.01)
print("Number of rules : ",len(associationRule))

#5c
plt.figure(figsize=(12,8))
sc =
plt.scatter(associationRule['confidence'],associationRule['support'],s=associationR
ule['lift'],c=associationRule['lift'])
plt.title("5C")
plt.colorbar(sc,label='Lift')
plt.xlabel("Confidence")
plt.ylabel("Support")
plt.show()

#5d
associationRuleMetric =
association_rules(freqItems,metric='confidence',min_threshold=0.6)
asdf = pd.DataFrame(associationRuleMetric)
print(asdf)


#Question 6

df = pd.read_csv("D:/ML Assignments SEM 1/Assignment 2/TwoFeatures.csv")
plt.figure(figsize=(12,8))
sc=plt.scatter(df['x1'],df['x2'])
```

```
plt.xlabel("x1")
plt.xlabel("x2")
plt.title("-6a-")
plt.show()

#6b
intervalVariables=df[['x1','x2']]
scaledIntervalVariables = MinMaxScaler(feature_range =
(1,8)).fit_transform(intervalVariables)
maxNC = 8
elbowValues = np.zeros(maxNC)
TWCSS = np.zeros(maxNC)
for c in range(maxNC):
    KClusters = c+1
    kmeans=cluster.KMeans(n_clusters=KClusters).fit(scaledIntervalVariables)
    WCSS = np.zeros(KClusters)
    nC = np.zeros(KClusters)
    for i in range(len(scaledIntervalVariables)):
        k=kmeans.labels_[i]
        nC[k]+=1
        diff = scaledIntervalVariables[i]-kmeans.cluster_centers_[k]
        WCSS[k]+=diff.dot(diff)
    for k in range(KClusters):
        elbowValues[c]+=WCSS[k]/nC[k]
        TWCSS[c]+=WCSS[k]
print("Elbow Values are : ")
EV=[]
for i in range(len(elbowValues)):
    EV.append(i)
    print(elbowValues[i])


print("TWCSS values are : ")
for j in range(len(TWCSS)):
    print("\n")
    print(TWCSS[j])

#6c
plt.figure(figsize=(12,8))
sc=plt.plot([1,2,3,4,5,6,7,8],elbowValues,marker='8')
plt.title('6c')
plt.xlabel("Cluster")
plt.ylabel("Elbow value")
plt.show()


#6d
intervalVariables=df[['x1','x2']]
scaledIntervalVariables = MinMaxScaler(feature_range =
(0,10)).fit_transform(intervalVariables)
maxNC = 8
elbowValues = np.zeros(maxNC)
TWCSS = np.zeros(maxNC)
for c in range(maxNC):
    KClusters = c+1
    kmeans=cluster.KMeans(n_clusters=KClusters).fit(scaledIntervalVariables)
    WCSS = np.zeros(KClusters)
    nC = np.zeros(KClusters)
    for i in range(len(scaledIntervalVariables)):
        k=kmeans.labels_[i]
        nC[k]+=1
        diff = scaledIntervalVariables[i]-kmeans.cluster_centers_[k]
        WCSS[k]+=diff.dot(diff)
    for k in range(KClusters):
        elbowValues[c]+=WCSS[k]/nC[k]
        TWCSS[c]+=WCSS[k]

print("Elbow Values are : ")
EV=[]
for i in range(len(elbowValues)):
    EV.append(i)
    print(elbowValues[i])


print("TWCSS values are : ")
for j in range(len(TWCSS)):
    print(TWCSS[i])

#6e
plt.figure(figsize=(12,8))
plt.title('6e')
plt.xlabel("Cluster")
plt.xlabel("Elbow value")
sc=plt.plot([1,2,3,4,5,6,7,8],elbowValues,marker='o')
plt.show()
```