7/23/2022

Group 6 Assignment

Group Members
Shantanu Aggarwal
Itisha Sharma
Hansaraj Baikunth Parida
Meghshyam Bijja
Chaitanya Nevhal

Group 6 - Rotman-Jaro Advanced Data Science Program - Coding & Data Tools 2022

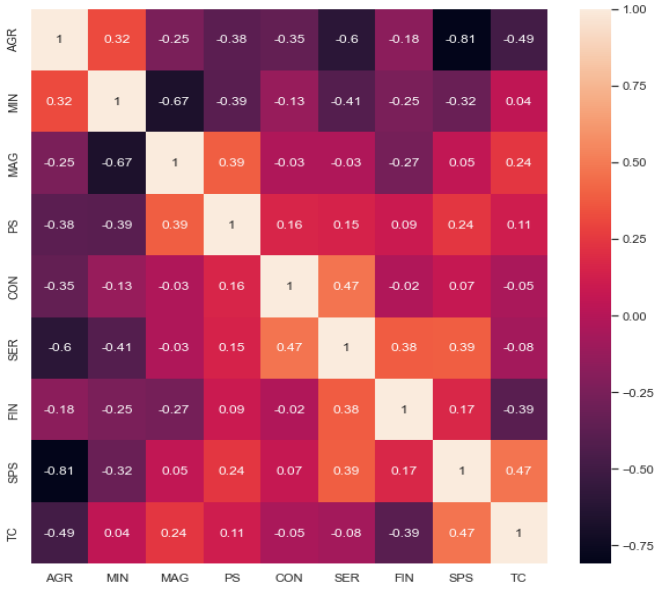
Table of contents

Group 6 - Rotman-Jaro Advanced Data Science Program - Coding & Data Tools 2022

Q 1. A. Which variables would you suggest to include (and exclude) in the cluster (or segmentation) analysis? And why?

Ans:

Calculating the correlation matrix, we get the following output:

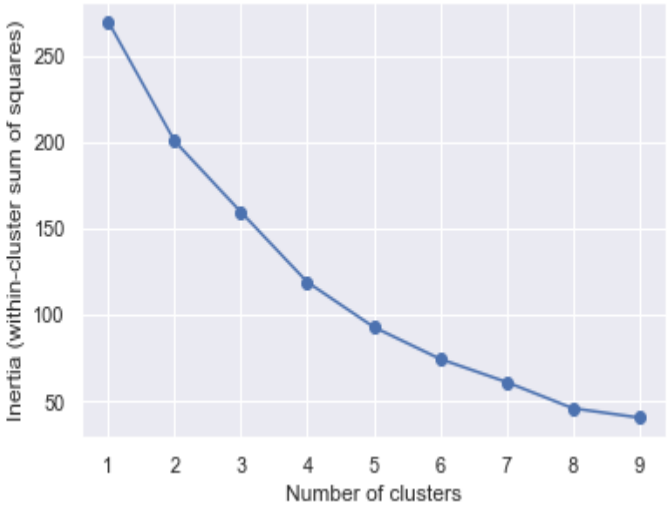


From the above correlation matrix, we can infer that Agriculture(AGR) is highly correlated to Social & personal services(SPS){**0.81**} and has High correlation with Services (SER) as well. Also, Manufacturing and Mining are Highly Correlated {**0.67**}. Since Agriculture is correlated to 2 other variables, we will not use it for our analysis. Between Manufacturing and Mining, we will be using manufacturing because mining shows higher correlation with other variables as well.

B. Based on the variables included in the analysis from step 1a, perform a clustering analysis and identify three possible clustering solutions (e.g., solution 1: 2 clusters, solution 2: 3 clusters, solution 3: 4 clusters, etc.)

Ans:

Using performance elbow method, we get the following chart:



From the graph we can conclude that there is sharp decline in inertia at 2 clusters and 4 clusters. So for our 3 solutions will be 2 clusters, 3 clusters and 4 clusters.

Alternatively, calculating Silhouette Analysis on cluster size, we observe:

For n_clusters= 2 The average silhouette_score is : 0.299
For n_clusters= 3 The average silhouette_score is : 0.294
For n_clusters= 4 The average silhouette_score is : 0.306
For n_clusters= 5 The average silhouette_score is : 0.244
For n_clusters= 6 The average silhouette_score is : 0.243

Considering Top 3 silhouette scores, our 3 solutions will be 2 clusters, 3 clusters and 4 clusters.

C. Which of the clustering solutions (which you identified in 1b), would you propose to the client as the final recommendation? And why? (2 points)

Ans: As mentioned, there is a sharp decline in inertia from 3 clusters to 4 which is accompanied by decline in marginal gain when we increase the cluster from 4 to 5 and further.
Calculating Silhouette Score for each cluster :
For n_clusters= 2 The average silhouette_score is : 0.2999719095248648
For n_clusters= 3 The average silhouette_score is : 0.29426270400290455
For n_clusters= 4 The average silhouette_score is : 0.30622327250554554

We observe that when no. of clusters is 4, we achieve maximum silhouette score
Hence, our final recommendation would be a cluster solution with 4 clusters to the client.

D. Based on your proposed clustering solution in 1c, write a comparative analysis on how countries in a segment are different from those in other segments. And what implications it has for the client? (3 points)

Ans: Below are the cluster centers obtained after running cluster analysis

Cluster Centers:
['MAG' 'PS' 'CON' 'SER' 'SPS' 'FIN' 'TC']
[0.0312 0.073 0.617 0.816 -0.028 0.353 -0.364]
[0.229 -0.217 -0.156 -0.966 -0.620 -1.285 0.425]
[-0.019 0.293 -0.752 -0.077 1.200 0.503 0.582]
[-2.106 -1.266 -1.505 -2.343 -3.070 2.159 -2.745]

For above we can conclude following differences among 4 clusters thus formed:

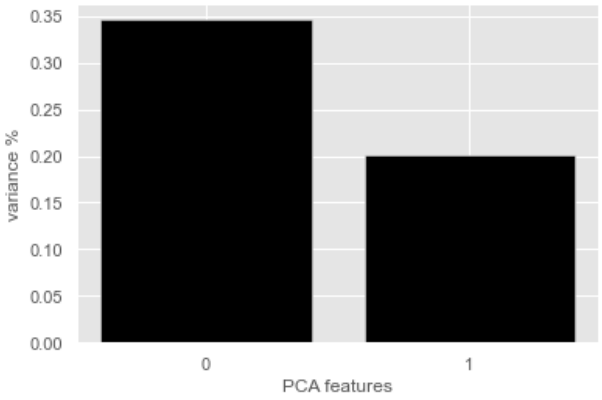
- Cluster 1 contains countries with high construction & services sector employment
- Cluster 2 contains countries with high manufacturing employment and moderately high Transport employment accompanied with low financial employment
- Cluster 3 contains countries with high social & personal service and Transport and communication employment
- Cluster 4 contains countries with very high financial employment but has very low employment in other sectors

---

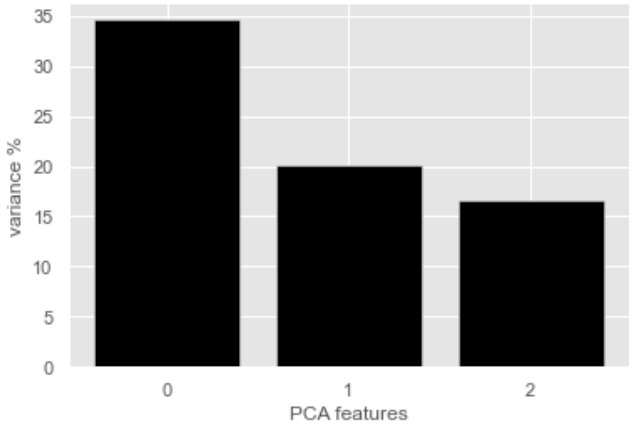
Q2. Using the same dataset (EUEmployment.csv), please run a principal component analysis (PCA) and extract two and three PCA components. Interpret and provide clear explanations of the results. Which of the solution would you propose to the client?

Ans: Before PCA we have normalized the data.
Running PCA for 2 dimensions:

% Variance explained:



Percent variance explained by PCA feature0 is **34.58%**
Percent variance explained by PCA_ feature1 is **20.11%**
Running PCA with **2 components** results in loss of **45.31%** of original variance in the dataset. Such a solution is not usable as we are losing nearly 50 percent information present with us.



Running PCA for 3 dimensions:

% Variance explained: Percent variance explained by PCA feature0 is **34.58%**
Percent variance explained by PCA_ feature1 is **20.11%**
Percent variance explained by PCA_ feature1 is **16.62%**

Running PCA with **3 components** results in loss of **28.7%** of original variance in the dataset. Here, information loss is significantly reduced, which is a much better solution than in the previous scenario.

Since we are gaining 17% information by adding 1 dimension, we would propose PCA with 3 dimensions to the client.

Q3. Consider a scenario of running Clustering and PCA analysis on a dataset which includes variables *weight* (in kg) with range 45-110 and *height* (in inches) with range 55 to 65. The range of *weight* is much wider than that of *height.* Does it create any challenges for Clustering and/or PCA analysis? If yes, what are those challenges and how would you overcome them? If no, why not?

Ans: Yes, this would a challenge for both clustering and PCA.

Clustering:

Clustering relies on distance between data-points. In these scenarios, where range of 1 variable is much greater than other, clusters formation relies heavily on the variable with higher range giving a skewed cluster. Applying methodologies like k-means will get affected heavily as the cluster formation will be completely dependent on variable with higher variance scale.

PCA:

PCA relies on maximizing variance along. if variance along one feature is very large than PCA will result in extremely favoring that dimension.

Solution:
Normalizing data before applying clustering and PCA analysis results in overcoming this issue. As data is now at same scale both distance calculation and variance will be independent of the range on features

---

Q4.
A. Would you recommend quantitative or qualitative research? Why? (5 points)

Since no research has been done on this topic, therefore it is suggested to do qualitative research. It would help us finding the relevant variables, organizing, and quantifying the unstructured content. It would also help us in deep diving the perceptions and thought processes of the children while taking the online classes. This method would not only help us to know the 'what' people think but also 'why' they think in that manner.

B. If you believe qualitative research is a better choice, which specific methodology (e.g., focus group, 1:1 interview, etc.) would you recommend to meet the objectives mentioned above? (7 points)

We would opt for 1:1 interview method. This method would help us to get precise and unbiased information from the students. In 1:1 interview, the students will not be under any influence and hence, we will get genuine responses. Since there would not be any time bounds in the 1:1 session, therefore, we can make ensure active participation of the each child. We can even match the body language with the responses. We are not preferring focus groups interviews as they are facilitator driven and may lead to biased information as there are chances that few children might share their views and other might not. Other methods like observation, case study etc are not cost effective and are time consuming.

C. For the chosen methodology in 'b', please write an "Introduction" (Max 300-400 words) which you need to provide to participants to help them understand the nature of your discussion with them (3 points)

As we all know that the Covid-19 had hit us hard and we all suddenly had to adapt ourselves to various technological means of imparting knowledge to our students. This sudden change has made us realize that it is time to upgrade our academic and operation policy. This update of the policies is necessary to be abreast with the technological advances and new methods of knowledge sharing.
Who else can be better than our own students to help us in this major transformation. Therefore, we are planning to conduct one on one interactions between the counsellor/ researcher and you to gain insights. The inputs will help us in providing a conducive and productive learning experience for you. We would like to understand your perceptions towards online classes, learn about your daily routine and approach towards the virtual set-up. The interactions would last for not more than 20 minutes and the schedule would be shared with you shortly. The conversation between you and the counsellor will remain confidential. This discussion will lay foundation for the future process of

effective exchange of expertise between the knowledge experts and you. The entire team of Dehradun Public School is thankful for your participation in its journey of transfiguration.

Q 5. Based on the above description and the story mentioned in the article,

A. What are YOUR VIEWS/ FINAL set of arguments (~500 words) on fair research and data collection practices, ethics in business, sharing and using customer data for profit, and privacy in today's world? You can write a balanced set of arguments OR arguments favouring / opposing to what Target did. (5 points)

In today's world the customers are aware that they are under surveillance even if they are poorly informed about the various types of data being collected about them. Therefore, the daughter, in this case study was aware that her personal information is being collected by the Target but was oblivious to the way it could be used by them. Here, Target was unethical in collecting and analyzing the data from other sources outside themselves. As a fair research and data collection practice, Target should have made it clear to the users about the kind of information being collected by them and possible consequences and usage of data. On the other hand, it was also the responsibility of the customer to be conscious about the kind of information he/ she is sharing in public.

We understand the fact that a marketing strategy's success is measured by consumer response, and organizations can define the market and identify business threats and opportunities by using the insights from consumer data. In this ever-changing world of data driven technology, the risks and potential harms of using consumer data often remains unrecognized. The knowledge about different data collection methods and personalization techniques are scarce, making the consumers more vulnerable and unable to control the use of their data.

Some customers do get attracted by the benefits of personalized targeting but on the contrary in some cases it might lead to the decrease in interest and involvement of the customer. This can be coined as personalization – privacy paradox.

The laws like, General Data Protection Regulation (GDPR) in Europe, can only help us to ensure the minimum regulation to ensure public order but the ethical behavior is based on voluntary action of the organizations. Since organizations are a part of society and there is a thin line between privacy and personalization, therefore, the organizations need to maintain them at their individual levels. The organizations must restrict themselves from selling their customer data to other parties. It is also suggested that the companies are adhering to the notion of not targeting minors directly by any means for promotions.

In this case study, it was unclear if Target broke any laws, but it is evident that Target had a strong data analysis team whose predictions were cent percent correct and hence they were able to target the right set of customers at the right time with the right set of items required. It would be unerring to say that the intentions of Target were not wrong (ethical) but it led to disparate impact which was unlawful.

B. **Also, would you accept a job offer from Target to work on assignments similar to what Pole did? (5 points)**

As a team, we are impressed with how Pole performed the data analysis and came out with such precise personalization of ads for the customers.

Since, Target has improvised itself in the way it marketed its promotions later in the case study, therefore we would accept the offer. We would prefer to work on similar assignments but in different fields benefitting the security of our nation e.g. identifying the suspicious activities, traffic controls etc which would fall under the purview of the central government.

It would be a cherry on cake if the government also comes up with concrete set of policies and laws to regulate the collection and use of the private data.

Assignment-q1

July 23, 2022

[ ]:
```
# loading packages

import os

import pandas as pd
import numpy as np

# plotting packages
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as clrs
import seaborn as sns
sns.set()
# Kmeans algorithm from scikit-learn
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_samples, silhouette_score
```

0.1 Load raw data

[ ]:
```
# load raw data
DATA_FOLDER = './'
raw = pd.read_csv(os.path.join(DATA_FOLDER, 'EUEmployment.csv'))
# check the raw data
columns=[str(i).replace('(','').replace(')','') for i in raw.loc[0,:].values]
columns[1]='Group'
raw.columns=columns
Data=raw.loc[1:]
Data.head(31)
for col in Data.columns:
    if col not in ['Country','Group']:
        Data[col] = Data[col].astype(float)
```

```
C:\Users\shantanu.aggarwal\AppData\Local\Temp\ipykernel_18732\3721780520.py:12:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Data[col] = Data[col].astype(float)
```

[ ]:
```
Data.shape
```

[ ]:
```
(30, 11)
```

[ ]:
```
import numpy as np
p=np.arange(.1,1,.1)
Data.describe(p)
```

[ ]:

	AGR	MIN	MAG	PS	CON	SER
count	30.000000	30.000000	30.000000	30.0000	30.000000	30.000000
mean	12.186667	3.446667	20.286667	0.8000	7.530000	15.636667
std	12.306901	8.865732	9.456789	0.6209	2.733086	5.160158
min	0.000000	0.000000	0.000000	0.0000	0.600000	3.300000
10%	2.600000	0.000000	6.120000	0.0000	5.140000	9.250000
20%	3.280000	0.100000	18.020000	0.0000	6.300000	10.280000
30%	5.070000	0.200000	19.140000	0.5700	6.400000	13.650000
40%	5.720000	0.300000	19.480000	0.7000	6.760000	14.500000
50%	8.450000	0.500000	20.300000	0.8000	7.050000	16.800000
60%	10.900000	0.600000	21.180000	0.9000	8.140000	17.680000
70%	13.590000	0.760000	23.750000	1.0300	8.680000	18.680000
80%	18.600000	1.320000	25.220000	1.2000	9.240000	20.120000
90%	22.340000	5.450000	29.420000	1.5500	9.910000	21.240000
max	55.500000	37.300000	38.700000	2.2000	16.900000	24.500000

	FIN	SPS	TC
count	30.00000	30.000000	30.000000
mean	6.65000	26.993333	6.453333
std	3.98668	8.732063	1.233368
min	0.00000	0.000000	3.000000
10%	1.23000	18.720000	4.980000
20%	2.24000	21.140000	5.540000
30%	4.39000	23.240000	5.940000
40%	6.14000	25.140000	6.400000
50%	7.15000	27.000000	6.750000
60%	8.48000	28.400000	6.800000
70%	8.82000	31.420000	6.930000
80%	9.72000	34.460000	7.260000
90%	10.87000	37.580000	7.830000
max	15.30000	41.600000	8.800000

[ ]:
```
Data['Country'].nunique()
Data['Country'].unique()
```

```
[ ]: array(['Belgium', 'Denmark', 'France', 'Germany', 'Greece', 'Ireland',
            'Italy', 'Luxembourg', 'Netherlands', 'Portugal', 'Spain', 'UK',
            'Austria', 'Finland', 'Iceland', 'Norway', 'Sweden', 'Switzerland',
            'Albania', 'Bulgaria', 'Czech/Slovakia', 'Hungary', 'Poland',
            'Romania', 'USSRF', 'Yugoslavia', 'Cyprus', 'Gibraltar', 'Malta',
            'Turkey'], dtype=object)
```

```
[ ]: Data['Group'].nunique()
     Data['Group'].unique()
```

```
[ ]: array(['EU', 'EFTA', 'Eastern', 'Other'], dtype=object)
```

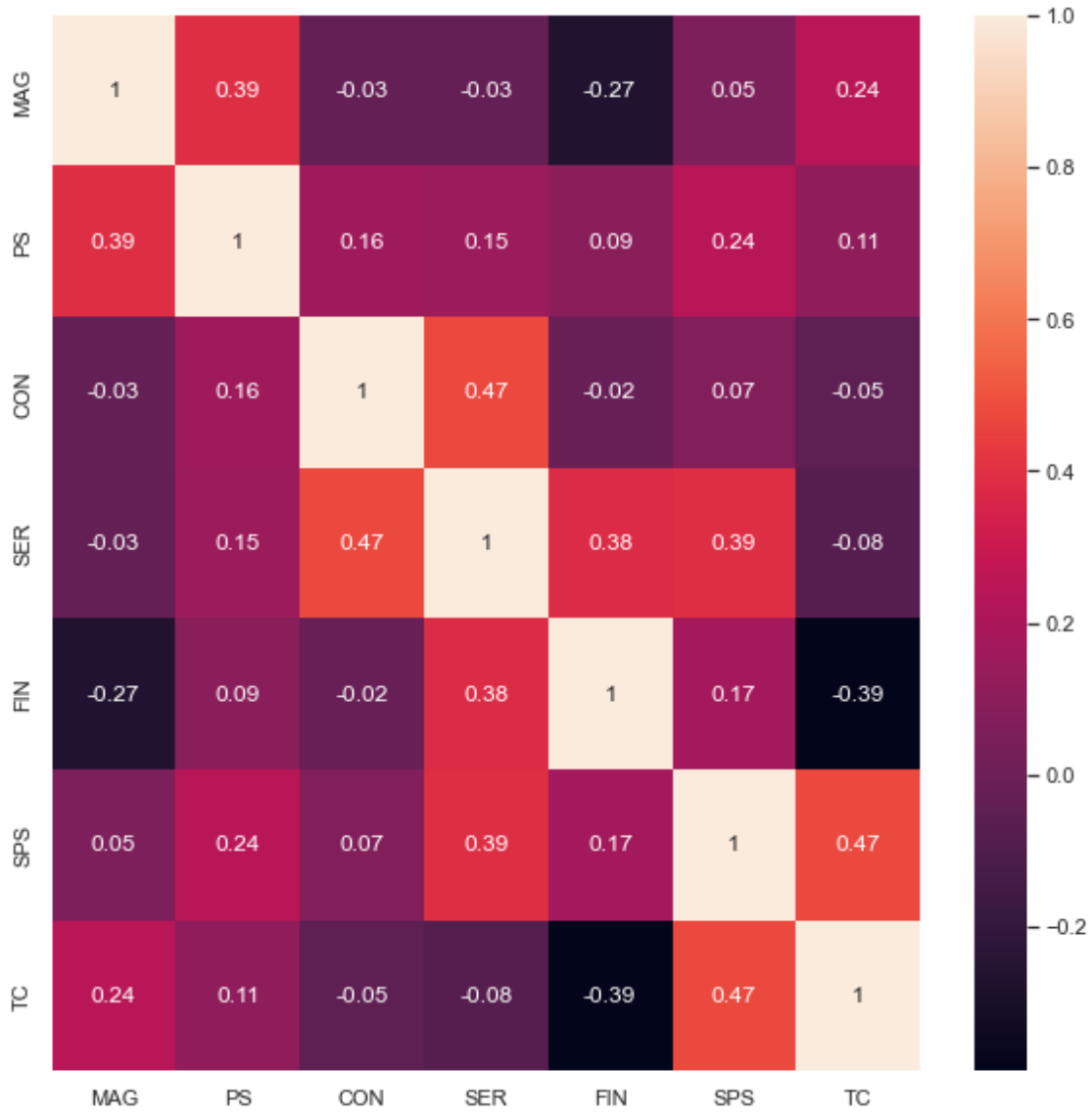
```
[ ]: Data.columns
```

```
[ ]: Index(['Country', 'Group', 'AGR', 'MIN', 'MAG', 'PS', 'CON', 'SER', 'FIN',
           'SPS', 'TC'],
          dtype='object')
```

```
[ ]: correlation_matrix = Data[['MAG', 'PS', 'CON', 'SER', 'FIN','SPS', 'TC']].
      ↪corr().round(2)
     # annot = True to print the values inside the square
     fig, ax = plt.subplots(figsize=(10,10))
     sns.heatmap(data=correlation_matrix, annot=True, ax = ax)

     print(correlation_matrix)
```

```
      MAG    PS   CON   SER   FIN   SPS    TC
MAG  1.00  0.39 -0.03 -0.03 -0.27  0.05  0.24
PS   0.39  1.00  0.16  0.15  0.09  0.24  0.11
CON -0.03  0.16  1.00  0.47 -0.02  0.07 -0.05
SER -0.03  0.15  0.47  1.00  0.38  0.39 -0.08
FIN -0.27  0.09 -0.02  0.38  1.00  0.17 -0.39
SPS  0.05  0.24  0.07  0.39  0.17  1.00  0.47
TC   0.24  0.11 -0.05 -0.08 -0.39  0.47  1.00
```



[ ]:
```
Data2 = Data.drop(columns=['Country','Group','AGR',"MIN"]).copy()

Data2=(Data2-Data2.mean())/Data2.std()
```

[ ]:
```
Data2
```

[ ]:

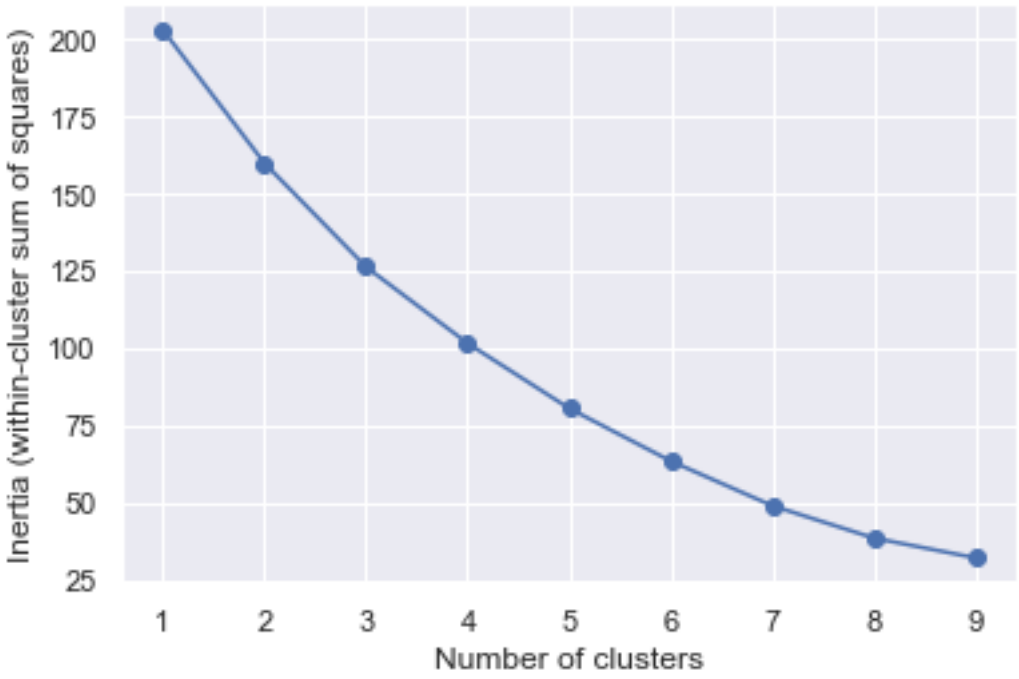
	MAG	PS	CON	SER	FIN	SPS	TC
1	0.054282	0.000000	-0.450041	0.244825	0.514212	1.134516	0.281073
2	0.011984	-0.161056	-0.413452	-0.220277	0.614546	1.065804	0.443231
3	-0.009164	0.161056	-0.157331	0.206066	0.890465	0.699338	-0.043242
4	0.477259	0.322113	0.684208	0.302962	0.739964	0.161092	-0.691873
5	-0.114909	0.322113	-0.267097	0.496755	-0.338628	-0.823784	0.362152

```
6  -0.051462  0.644226 -0.157331  0.419238  0.438962 -0.171017 -0.529715
7   0.170601 -1.288452  0.574442  1.155649 -0.514212  0.115284 -0.935109
8  -0.072611 -0.161056  0.867152  1.078132  0.514212  0.298517  0.281073
9  -0.114909 -0.161056 -2.535595  0.554893  1.216551  1.294845  0.281073
10  0.350366 -0.161056  0.245144  0.806823 -0.087792 -0.274086 -1.340503
11  0.086005 -0.322113  0.720797  0.864961 -0.188126 -0.033593 -0.529715
12  0.107154  0.644226 -0.193920  0.884340  1.442303  0.161092  0.037837
13  0.699321  0.644226  0.354910  0.671168  0.012542 -0.422962 -0.043242
14 -0.104334  0.644226 -0.267097 -0.200898  0.489129  0.710790  0.848625
15 -0.167781  0.161056  0.903740 -0.220277  0.338628  0.424489  0.199994
16 -0.601332  0.483169 -0.376863  0.380479  0.238293  1.203228  1.335098
17 -0.136057  0.000000 -0.413452 -0.278415  0.689797  1.432269  0.605389
18  0.466684 -1.288452  0.611031  0.942478  1.015883 -0.445866 -0.205400
19 -2.145196 -1.288452 -1.511112 -2.390754  2.169725 -3.091289 -2.799922
20  1.555849 -1.288452 -0.303686 -1.208619 -1.291802 -0.697811  0.848625
21 -2.145196 -1.288452  0.318321 -1.053585 -1.266718 -0.468770  0.362152
22 -2.145196 -1.288452 -0.413452 -0.452829 -1.668054  0.035120  1.902650
23  0.403238  0.161056 -0.450041 -1.034206 -1.341969 -0.285538 -1.016188
24  1.862507  1.932677 -0.632984 -1.693101 -1.517553 -1.339126  0.281073
25  0.900235 -1.288452  0.976918 -1.499308 -1.517553 -0.159565  1.578335
26  1.947102  2.254790  0.208555 -0.355932 -0.890465 -0.903948  1.091862
27 -0.136057 -0.483169  0.574442  1.562614  0.012542 -0.663455 -0.367557
28 -1.426136  1.932677  3.428359  1.717648  1.040966  0.802407 -1.178346
29  0.805065  1.127395 -1.072048 -1.053585 -0.689797  1.672762  0.605389
30 -0.527311 -0.966339 -0.852516 -0.627242 -1.066050 -1.430743 -1.664819
```

[ ]:
```
# https://stackoverflow.com/questions/41540751/
  ↪sklearn-kmeans-equivalent-of-elbow-method

Ks = range(1, 10)
inertia = [KMeans(i).fit(Data2).inertia_ for i in Ks]

fig = plt.figure()
plt.plot(Ks, inertia, '-bo')
plt.xlabel('Number of clusters')
plt.ylabel('Inertia (within-cluster sum of squares)')
plt.show()
```



[ ]:
```
# Silhouette Analysis
range_n_clusters=[2,3,4,5,6]
for n_clusters in range_n_clusters:
    clusterer=KMeans(n_clusters=n_clusters, random_state=1)
    cluster_labels=clusterer.fit_predict(Data2)
    silhouette_avg=silhouette_score(Data2,cluster_labels)
    print("For n_clusters=", n_clusters,
          "The average silhouette_score is :", silhouette_avg)
```

```
For n_clusters= 2 The average silhouette_score is : 0.2999719095248648
For n_clusters= 3 The average silhouette_score is : 0.29426270400290455
For n_clusters= 4 The average silhouette_score is : 0.30622327250554554
For n_clusters= 5 The average silhouette_score is : 0.24474089837355237
For n_clusters= 6 The average silhouette_score is : 0.24365281455452953
```

[ ]:
```
k =2
kmeans = KMeans(n_clusters=k, random_state=1,n_init=20)
kmeans.fit(Data2)
y2= kmeans.labels_
```

[ ]:
```
label2 = pd.DataFrame({'Country':Data['Country'],'Label':y2})
label2.groupby('Label').count()
```

[ ]:
```
       Country
Label
```

```
0          22
1           8
```

[ ]:
```
k =3
kmeans = KMeans(n_clusters=k, random_state=1,n_init=50)
kmeans.fit(Data2)
y3= kmeans.labels_
```

[ ]:
```
label3 = pd.DataFrame({'Country':Data['Country'],'Label':y3})
label3.groupby('Label').count()
```

[ ]:
```
       Country
Label
0           20
1            9
2            1
```

[ ]:
```
k =4
kmeans = KMeans(n_clusters=k, random_state=1,n_init=50)
kmeans.fit(Data2)
y4=kmeans.labels_
```

[ ]:
```
label4 = pd.DataFrame({'Country':Data['Country'],'Label':y4})
label4.groupby('Label').count()
```

[ ]:
```
       Country
Label
0            8
1           13
2            1
3            8
```

[ ]:
```
cluster_centers  = pd.DataFrame(kmeans.cluster_centers_,columns=['MAG', 'PS',␣
 ↪'CON','SER', 'FIN','SPS',␣
 ↪'TC'],index=['center1','center2','center3','center4'])
cluster_centers
```

[ ]:
```
              MAG        PS       CON       SER       FIN       SPS        TC
center1  0.231403 -0.221453 -0.143611 -0.990603 -1.320021 -0.656298  0.422961
center2  0.029879  0.074334  0.641991  0.821730  0.340557 -0.067068 -0.380031
center3 -2.145196 -1.288452 -1.511112 -2.390754  2.169725 -3.091289 -2.799922
center4 -0.011808  0.261717 -0.710735 -0.045864  0.495400  1.151694  0.544580
```

[ ]:
```
result = pd.DataFrame({'Country':Data['Country'], 'Label':y4})
with pd.option_context('display.max_rows', None, 'display.max_columns', 3):
    print(result.sort_values('Label'))
```

```
          Country  Label
```

30	Turkey	0
26	Yugoslavia	0
25	USSRF	0
24	Romania	0
23	Poland	0
22	Hungary	0
21	Czech/Slovakia	0
20	Bulgaria	0
28	Gibraltar	1
27	Cyprus	1
18	Switzerland	1
13	Austria	1
15	Iceland	1
8	Luxembourg	1
4	Germany	1
11	Spain	1
10	Portugal	1
5	Greece	1
12	UK	1
7	Italy	1
6	Ireland	1
19	Albania	2
2	Denmark	3
3	France	3
14	Finland	3
17	Sweden	3
16	Norway	3
29	Malta	3
9	Netherlands	3
1	Belgium	3

Assignment-q2

July 23, 2022

[ ]:
```
import os

import pandas as pd
import numpy as np

# plotting packages
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
import matplotlib.cm as cm
import matplotlib.colors as clrs
import seaborn as sns
sns.set()
```

[ ]:
```
# load raw data
DATA_FOLDER = './'
raw = pd.read_csv(os.path.join(DATA_FOLDER, 'EUEmployment.csv'))
# check the raw data
columns=[str(i).replace('(','').replace(')','') for i in raw.loc[0,:].values]
columns[1]='Group'
raw.columns=columns
Data=raw.loc[1:]
Data.head(31)
for col in Data.columns:
    if col not in ['Country','Group']:
        Data[col] = Data[col].astype(float)
```

```
C:\Users\shantanu.aggarwal\AppData\Local\Temp\ipykernel_42264\3721780520.py:12:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  Data[col] = Data[col].astype(float)
```

[ ]:
```
from sklearn.preprocessing import StandardScaler
X = Data.drop(columns=['Country','Group']).copy()
```

```
Scaler = StandardScaler()
Scaler.fit(X)
X = Scaler.transform(X)
```

[ ]:
```
from sklearn.decomposition import PCA
PCA_0 = PCA(n_components = .80)
PCA_0.fit(X)
print(PCA_0.n_components_)
```

```
4
```

[ ]:
```
from sklearn.decomposition import PCA
PCA_1 = PCA(n_components = 2)
PCA_1.fit(X)
print(PCA_1.n_components_)
Data1_Scaled_PCA_1 = PCA_1.transform(X)
Data1_Scaled_PCA_1
```

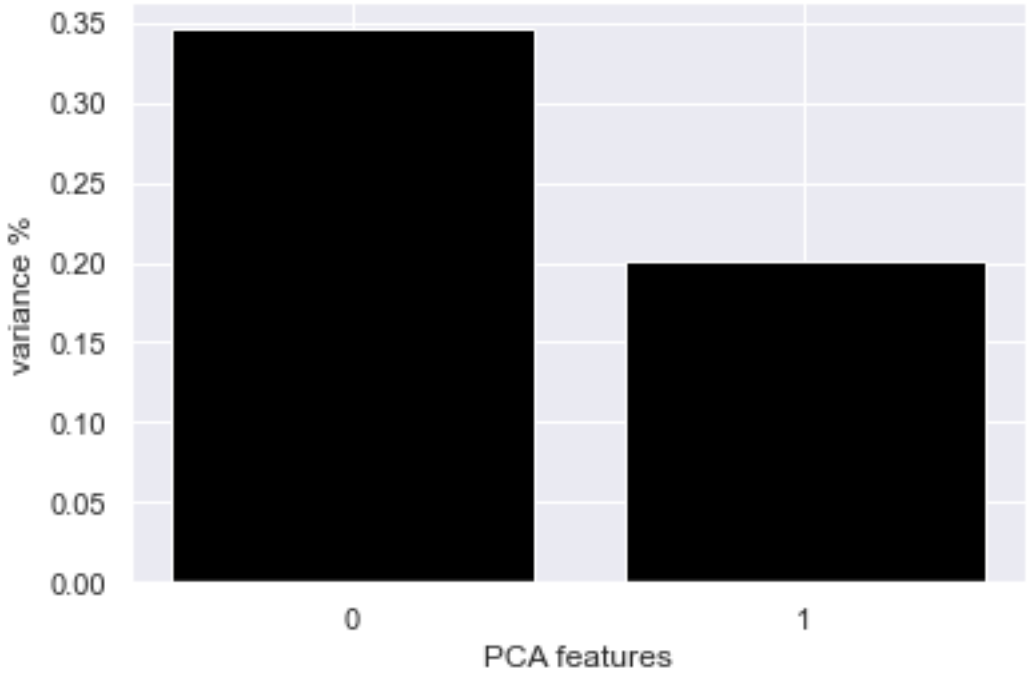
```
2
```

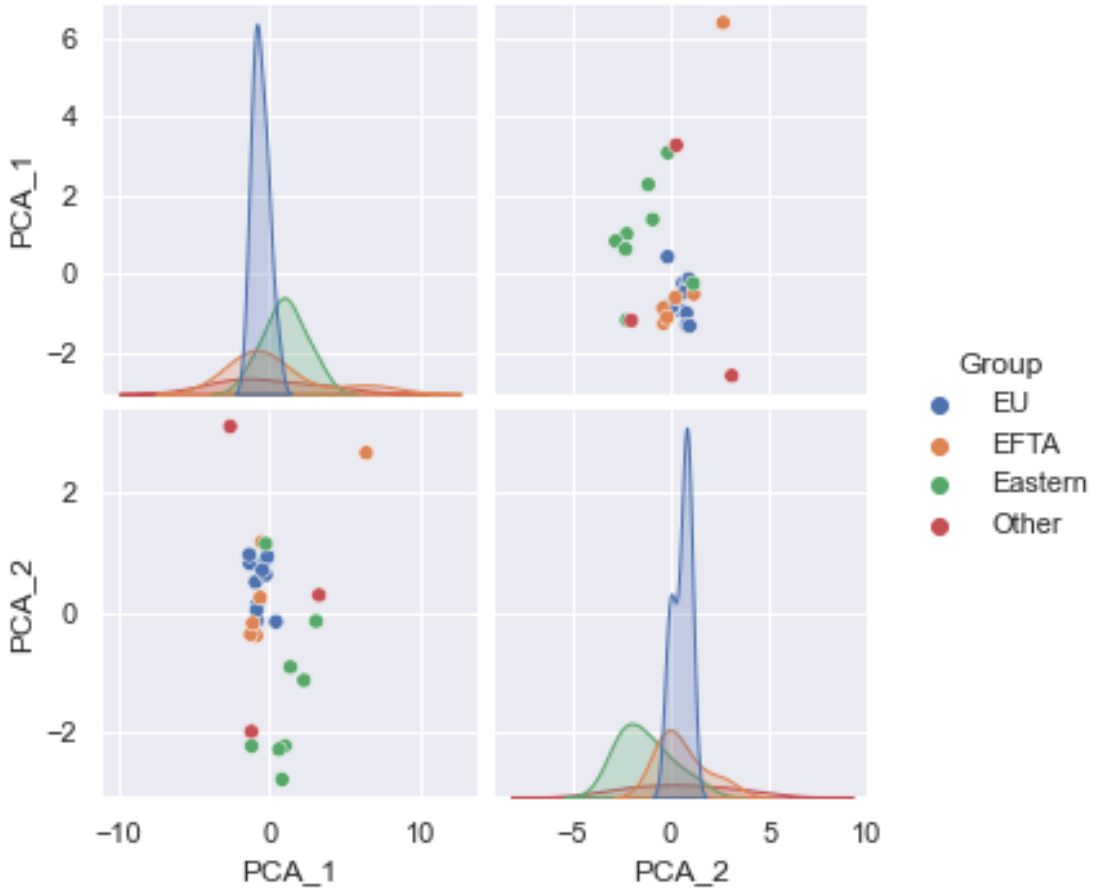
[ ]:
```
array([[-1.17332334,  0.02678337],
       [-0.83347266, -0.14000159],
       [-0.94318182,  0.50394053],
       [-1.01738187,  0.80267228],
       [ 0.41750822, -0.16021515],
       [-0.24930547,  0.62439489],
       [-0.25399875,  0.87821364],
       [-1.32066895,  0.8191242 ],
       [-0.82957684,  0.12255771],
       [-0.14941099,  0.92862056],
       [-0.4893054 ,  0.705066  ],
       [-1.34409044,  0.96388741],
       [-0.86883573,  0.03180945],
       [-0.89011638, -0.39236382],
       [-0.61927763,  0.24585954],
       [-1.2726561 , -0.37128778],
       [-1.12192327, -0.18495997],
       [-0.53117097,  1.17161652],
       [ 6.37783606,  2.66634826],
       [ 1.00104979, -2.23714447],
       [ 3.06822139, -0.15125386],
       [ 2.25710988, -1.14019185],
       [ 1.3672116 , -0.91809817],
       [ 0.81724921, -2.79913559],
       [ 0.61173546, -2.29545854],
       [-1.19413134, -2.23982382],
       [-0.26237655,  1.14374154],
       [-2.60947123,  3.10696675],
```

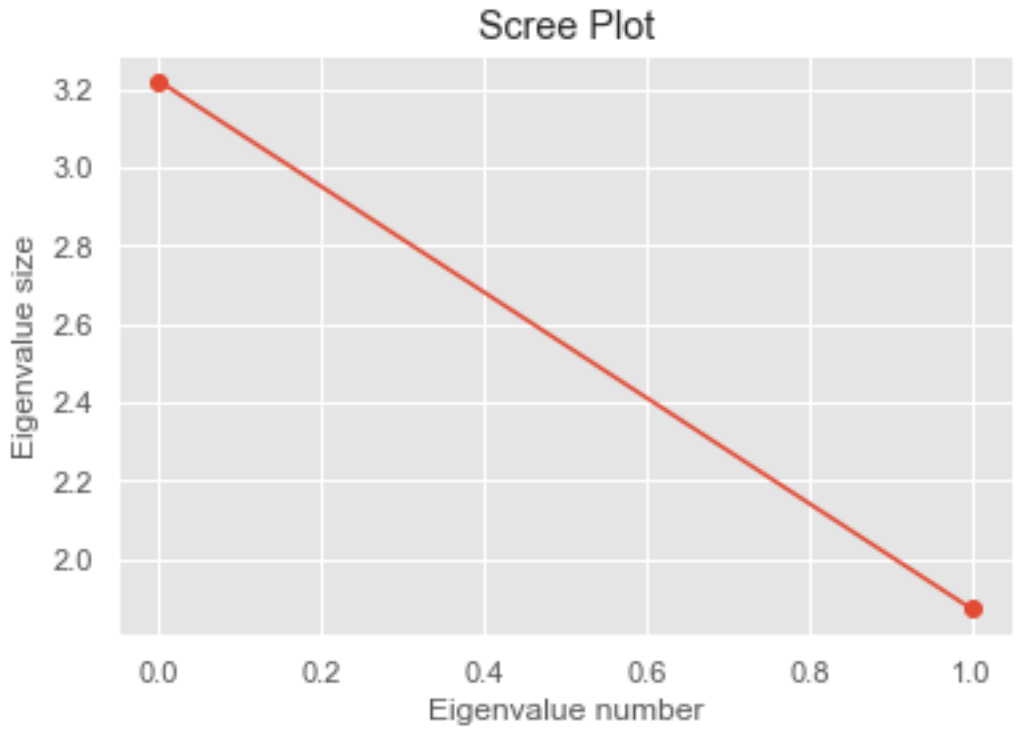
```
       [-1.20403396, -1.99812294],
       [ 3.25978809,  0.28645491]])
```

[ ]:
```
Data1_Scaled_PCA_1.shape
# Plot the variances
features = range(PCA_1.n_components_)
plt.bar(features, PCA_1.explained_variance_ratio_, color='black')
plt.xlabel('PCA features')
plt.ylabel('variance %')
plt.xticks(features)
# Save components to a DataFrame
PCA_components = pd.DataFrame(Data1_Scaled_PCA_1, columns=['PCA_1','PCA_2'])
PCA_components_1 = pd.concat([PCA_components,Data['Group']],axis=1)
PCA_components_1.head()
colors = {'EU':'red', 'EFTA':'green', 'Eastern':'blue', 'Other':'yellow'}

sns.pairplot(PCA_components_1, vars = PCA_components_1.columns[:-1],␣
 ↪hue='Group')
plt.show()
# Plot the variances
plt.style.use("ggplot")
plt.plot(PCA_1.explained_variance_, marker='o')
plt.xlabel("Eigenvalue number")
plt.ylabel("Eigenvalue size")
plt.title("Scree Plot")
plt.show()
```




PCA_1
PCA_2
6
4
2
0
−2
−10
0
10
−5
5
Group
EU
EFTA
Eastern
Other



[ ]:
```
PCA_1.explained_variance_ratio_
```

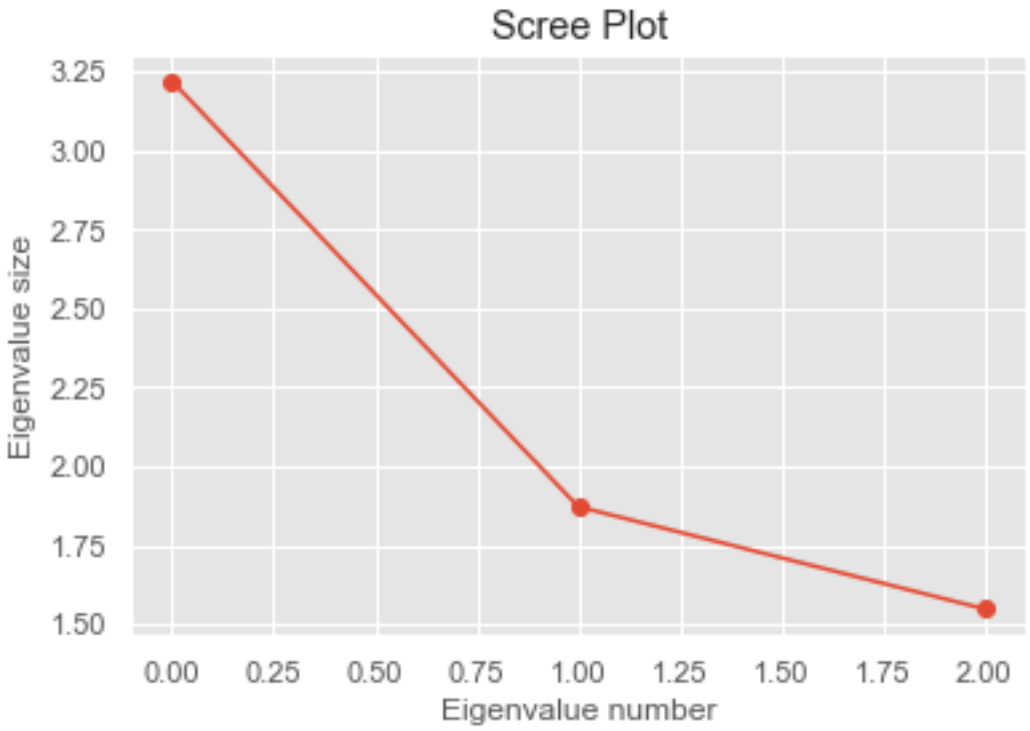
[ ]:
```
array([0.34580644, 0.20102636])
```

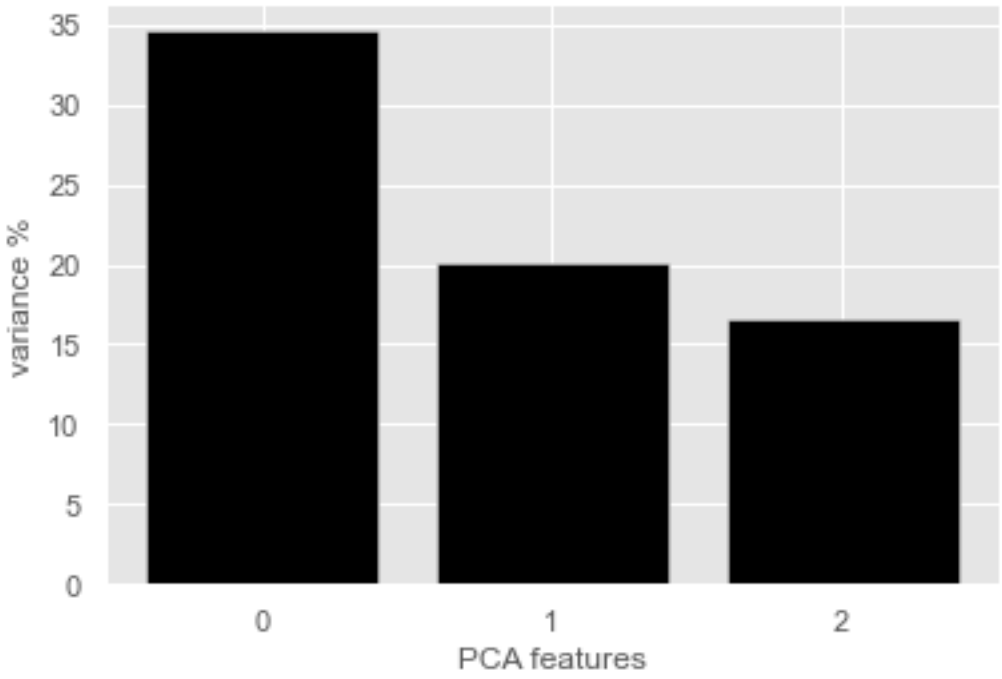
[ ]:
```
PCA_2 = PCA(n_components = 3)
PCA_2.fit(X)
Data1_Scaled_PCA_2 = PCA_2.transform(X)
```

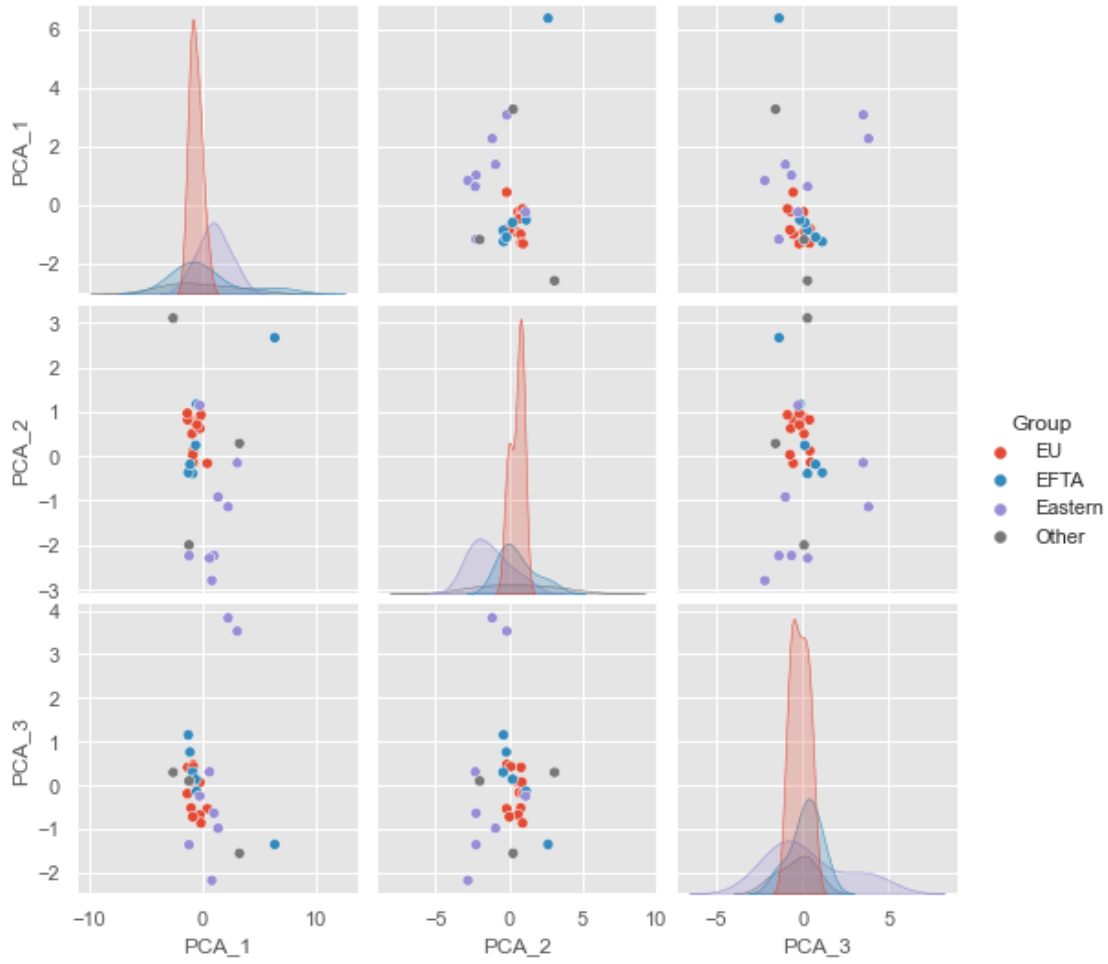
[ ]:
```
# Plot the variances
plt.style.use("ggplot")
plt.plot(PCA_2.explained_variance_, marker='o')
plt.xlabel("Eigenvalue number")
plt.ylabel("Eigenvalue size")
plt.title("Scree Plot")
plt.show()

# Plot the variances Ratio
features = range(PCA_2.n_components_)
plt.bar(features, (PCA_2.explained_variance_ratio_)*100, color='black')
plt.xlabel('PCA features')
plt.ylabel('variance %')
plt.xticks(features)
plt.show()
```

```
PCA_components = pd.DataFrame(Data1_Scaled_PCA_2, columns=['PCA_1','PCA_2',␣
 ↪'PCA_3'])
PCA_components_1 = pd.concat([PCA_components,Data['Group']],axis=1)
PCA_components_1.head()
sns.pairplot(PCA_components_1, vars = PCA_components_1.columns[:-1],␣
 ↪hue='Group')
plt.show()
```




35
30
25
20
15
10
5
0
variance %
0
1
2
PCA features



[ ]:
```
sum(PCA_2.explained_variance_ratio_)
```

[ ]: 0.7130794878724256

[ ]: