MA_Assignment_1

September 22, 2023

1 Assignment 1: Ford Ka STP Analysis

Group Members: Jing Du, Yiran Li, Chenhai Mu, Yirou Xie

[1]:
```
!pip install factor_analyzer
```

```
Collecting factor_analyzer
  Downloading factor_analyzer-0.5.0.tar.gz (42 kB)
                             42.5/42.5 kB
1.3 MB/s eta 0:00:00
  Installing build dependencies … done
  Getting requirements to build wheel … done
  Preparing metadata (pyproject.toml) … done
Requirement already satisfied: numpy in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from factor_analyzer)
(1.24.2)
Requirement already satisfied: pandas in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from factor_analyzer)
(1.5.3)
Collecting pre-commit
  Downloading pre_commit-3.4.0-py2.py3-none-any.whl (203 kB)
                             203.7/203.7
kB 5.2 MB/s eta 0:00:00a 0:00:01
Requirement already satisfied: scikit-learn in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from factor_analyzer)
(1.3.0)
Requirement already satisfied: scipy in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from factor_analyzer)
(1.11.1)
Requirement already satisfied: python-dateutil>=2.8.1 in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from
pandas->factor_analyzer) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from
pandas->factor_analyzer) (2022.7)
Collecting cfgv>=2.0.0
  Downloading cfgv-3.4.0-py2.py3-none-any.whl (7.2 kB)
Requirement already satisfied: pyyaml>=5.1 in
```

```
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from pre-
commit->factor_analyzer) (6.0.1)
Collecting nodeenv>=0.11.1
  Downloading nodeenv-1.8.0-py2.py3-none-any.whl (22 kB)
Collecting virtualenv>=20.10.0
  Downloading virtualenv-20.24.5-py3-none-any.whl (3.7 MB)
                             3.7/3.7 MB
11.3 MB/s eta 0:00:0000:0100:01
Collecting identify>=1.0.0
  Downloading identify-2.5.29-py2.py3-none-any.whl (98 kB)
                             98.9/98.9 kB
3.3 MB/s eta 0:00:00
Requirement already satisfied: joblib>=1.1.1 in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from scikit-
learn->factor_analyzer) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from scikit-
learn->factor_analyzer) (3.2.0)
Requirement already satisfied: setuptools in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from
nodeenv>=0.11.1->pre-commit->factor_analyzer) (65.6.3)
Requirement already satisfied: six>=1.5 in
/Users/cassiedu/miniconda3/lib/python3.10/site-packages (from python-
dateutil>=2.8.1->pandas->factor_analyzer) (1.12.0)
Collecting platformdirs<4,>=3.9.1
  Downloading platformdirs-3.10.0-py3-none-any.whl (17 kB)
Collecting distlib<1,>=0.3.7
  Downloading distlib-0.3.7-py2.py3-none-any.whl (468 kB)
                             468.9/468.9
kB 8.9 MB/s eta 0:00:0000:01
Collecting filelock<4,>=3.12.2
  Downloading filelock-3.12.4-py3-none-any.whl (11 kB)
Building wheels for collected packages: factor_analyzer
  Building wheel for factor_analyzer (pyproject.toml) … done
  Created wheel for factor_analyzer:
filename=factor_analyzer-0.5.0-py2.py3-none-any.whl size=42487
sha256=6c9e1473f7a7b745b742f22bc05c0754a38ffc14799a0100a8595fe54bc8220f
  Stored in directory: /Users/cassiedu/Library/Caches/pip/wheels/74/a2/6c/26fb1a
ddf1ce6c60a8cef8397f2999f0a1e6e2fcddc8abf33e
Successfully built factor_analyzer
Installing collected packages: distlib, platformdirs, nodeenv, identify,
filelock, cfgv, virtualenv, pre-commit, factor_analyzer
  Attempting uninstall: platformdirs
    Found existing installation: platformdirs 2.5.2
    Uninstalling platformdirs-2.5.2:
      Successfully uninstalled platformdirs-2.5.2
Successfully installed cfgv-3.4.0 distlib-0.3.7 factor_analyzer-0.5.0
```

```
filelock-3.12.4 identify-2.5.29 nodeenv-1.8.0 platformdirs-3.10.0 pre-
commit-3.4.0 virtualenv-20.24.5
```

[3]:
```
import pandas as pd
import numpy as np
from google.colab import files
from scipy.stats import chi2_contingency
import statsmodels.api as sm
import matplotlib.pyplot as plt
import factor_analyzer
from scipy.cluster import hierarchy
from sklearn import cluster
from sklearn.cluster import KMeans
from yellowbrick.cluster import KElbowVisualizer
```

2 Question 4: Demographic Segments

[4]:
```
# Get Data
uploaded = files.upload()
demo = pd.read_excel('Ford Ka Data.xlsx', sheet_name = 'Demographic Data',␣
 ↪skiprows = 6)
```

```
<IPython.core.display.HTML object>
```

```
Saving Ford Ka Data.xlsx to Ford Ka Data.xlsx
```

[114]:
```
# Data Preprocessing
demo.columns = demo.columns.str.replace(' ', '_')
demo_filtered = demo[demo['Preference_Group'] != 3]
```

[115]:
```
# Check Grouping variable
demo_filtered.value_counts(demo['Preference_Group'])
```

[115]:
```
Preference_Group
1    116
2     72
dtype: int64
```

[116]:
```
demo_x = demo_filtered.iloc[:, 2:]
print(demo_x.corr())
```

```
                     Gender       Age  Marital_Status  Number_of_Children  \
Gender             1.000000 -0.023979       -0.060910            0.108601
Age               -0.023979  1.000000        0.091010            0.048387
Marital_Status    -0.060910  0.091010        1.000000           -0.033339
Number_of_Children 0.108601  0.048387       -0.033339            1.000000
1st_Time_Purchase -0.137010  0.087820        0.062207           -0.036906
```

```
Age_Category       -0.035172  0.964922        0.071580            0.064532
Children_Category   0.117438 -0.013771       -0.066475            0.959283
Income_Category    -0.101870  0.126536       -0.017825            0.056422

                    1st_Time_Purchase  Age_Category  Children_Category  \
Gender                      -0.137010     -0.035172           0.117438
Age                          0.087820      0.964922          -0.013771
Marital_Status               0.062207      0.071580          -0.066475
Number_of_Children          -0.036906      0.064532           0.959283
1st_Time_Purchase            1.000000      0.059244          -0.051832
Age_Category                 0.059244      1.000000           0.005682
Children_Category           -0.051832      0.005682           1.000000
Income_Category              0.099700      0.141322           0.069530

                    Income_Category
Gender                    -0.101870
Age                        0.126536
Marital_Status            -0.017825
Number_of_Children         0.056422
1st_Time_Purchase          0.099700
Age_Category               0.141322
Children_Category          0.069530
Income_Category            1.000000
```

Cross Tabulation

[117]:
```
# Cross tabulation and chi2 significance testing
result = {}
for c in demo_filtered.columns[2:]:
  ct = pd.crosstab(index = demo_filtered['Preference_Group'], columns =␣
 ↪demo_filtered[c])
  print(ct)
  chi2, p, dof, expected = chi2_contingency(ct)
  print(f'p-value: {p}')
  print(' ')
  result[c] = p
```

```
Gender              1   2
Preference_Group
1                  54  62
2                  36  36
p-value: 0.7566209707272762

Age                20  21  22  23  24  26  27  28  29  30  …  48  49  50  51  \
Preference_Group                                           …
1                   3   1   2   2   2   5   5   2   6   5  …   3   1   1   2
2                   2   0   1   0   0   5   3   3   2   3  …   3   3   1   1
```

```
Age                52  54  55  56  57  58
Preference_Group
1                   2   1   2   1   0   0
2                   0   1   1   1   1   2

[2 rows x 37 columns]
p-value: 0.6953643703705099

Marital_Status     1   2   3
Preference_Group
1                 66  14  36
2                 34   6  32
p-value: 0.16753348988110406

Number_of_Children   0   1   2  3  4
Preference_Group
1                    62  29  15  8  2
2                    45  12   7  8  0
p-value: 0.3292833389612073

1st_Time_Purchase   1    2
Preference_Group
1                  13  103
2                   8   64
p-value: 1.0

Age_Category       1   2   3   4   5   6
Preference_Group
1                 10  18  23  11  36  18
2                  3  13  12  11  15  18
p-value: 0.2397592491624098

Children_Category   0   1   2
Preference_Group
1                  62  29  25
2                  45  12  15
p-value: 0.3561240358951042

Income_Category    1   2   3   4   5   6
Preference_Group
1                 11  19  18  19  28  21
2                  5  15  16  16  12   8
p-value: 0.37851010071730096
```

[146]:
```
# Summary Table of p-values
results = pd.DataFrame.from_dict(result.items())
```

```
results = results.transpose()
results.columns = results.iloc[0]
results = results.drop(labels = 0)
results
```

[146]:
```
0    Gender       Age Marital_Status Number_of_Children 1st_Time_Purchase  \
1  0.756621  0.695364       0.167533           0.329283                1.0

0 Age_Category Children_Category Income_Category
1     0.239759          0.356124         0.37851
```

3 Question 5: Attitudinal analysis

[8]:
```
# Get Data
uploaded = files.upload()
attitude = pd.read_excel('Ford Ka Data.xlsx', sheet_name = 'Psychographic␣
 ↪Data', skiprows = 6)
```

```
<IPython.core.display.HTML object>
```

```
Saving Ford Ka Data.xlsx to Ford Ka Data (1).xlsx
```

[158]:
```
# Data Preprocessing
attitude.columns = attitude.columns.str.replace(' ', '_')
df_merge = pd.merge(left = demo, right = attitude, left_on =␣
 ↪'Respondent_Number', right_on = 'Respondent_Number')
df_filtered = df_merge[df_merge.Preference_Group != 3]
```

Unrotated Factor Analysis

[159]:
```
# independent variables
reg_x = df_filtered.iloc[:, -62:]
```

[169]:
```
# Unrotated Factor Analysis PCA
attitude_pca = factor_analyzer.FactorAnalyzer(n_factors=62, rotation=None,␣
 ↪method='principal').fit(reg_x)
```

[170]:
```
# Loadings
def get_loadings_communalities(pca,round_dig=2,index_names=None):
  '''Returns a DataFrame containings the loadings'''
  df = pd.DataFrame(
      pca.loadings_,
      index=index_names if index_names else [f'q{i}' for i in range(1,1+pca.
 ↪loadings_.shape[0])],
      columns=[f'RC{i}' for i in range(1,1+pca.loadings_.shape[1])] if pca.
 ↪rotation else [f'PC{i}' for i in range(1,1+pca.loadings_.shape[1])]
      )
```

```
  if pca.rotation:
    df['communalities']=pca.get_communalities()
  df=df.round(3)
  return  df

get_loadings_communalities(attitude_pca)
```

[170]:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	\
q1	-0.523	-0.241	0.556	0.105	-0.039	0.017	-0.104	0.064	-0.122	0.057	
q2	-0.834	0.193	0.288	-0.017	0.054	0.027	0.018	0.010	-0.002	0.096	
q3	0.165	0.701	-0.106	0.034	0.084	0.009	-0.027	0.021	0.140	0.068	
q4	-0.006	0.658	-0.545	-0.094	0.017	0.045	-0.023	0.077	0.062	0.009	
q5	-0.138	0.836	0.280	0.022	-0.046	0.065	0.036	-0.002	0.065	0.018	
..	…	…	…	…	…	…	…	…	…	…	
q58	0.185	-0.167	0.599	0.015	0.070	0.016	0.121	0.234	-0.072	-0.290	
q59	0.203	-0.216	0.627	-0.078	-0.045	0.066	0.152	0.017	0.031	0.156	
q60	-0.238	0.030	-0.670	0.058	-0.008	-0.020	0.027	-0.038	-0.007	-0.135	
q61	-0.384	0.180	-0.573	0.095	0.083	0.120	-0.088	0.162	-0.159	-0.024	
q62	-0.278	0.141	-0.535	-0.157	0.067	0.099	0.055	0.043	0.138	0.259	

	…	PC53	PC54	PC55	PC56	PC57	PC58	PC59	PC60	PC61	PC62
q1	…	0.045	-0.070	-0.124	0.028	0.001	-0.051	0.006	0.013	-0.002	-0.013
q2	…	0.018	-0.078	0.036	-0.123	-0.010	0.182	0.019	0.012	-0.009	-0.009
q3	…	-0.051	0.043	0.015	0.002	-0.036	-0.007	-0.016	-0.015	0.027	-0.003
q4	…	0.020	0.045	-0.104	-0.002	0.044	0.106	0.027	0.037	-0.009	0.019
q5	…	0.085	0.018	0.016	0.189	-0.078	0.062	0.016	0.039	-0.056	0.006
..	…	…	…	…	…	…	…	…	…	…	
q58	…	0.013	-0.071	-0.009	0.036	0.029	-0.002	-0.001	0.014	0.002	-0.011
q59	…	0.014	0.035	0.037	0.006	0.018	0.034	-0.027	-0.022	0.011	-0.016
q60	…	0.001	-0.014	0.025	-0.020	0.020	0.009	-0.016	0.002	-0.035	0.005
q61	…	0.021	0.010	0.002	0.048	-0.043	-0.009	-0.015	-0.002	-0.019	0.009
q62	…	0.020	-0.013	-0.002	-0.006	0.047	-0.052	0.008	-0.010	-0.014	0.024

[62 rows x 62 columns]

[176]:
```
# Summary Data of PCA
def get_summary(pca,round_dig=2):
  ''' Print a summary of the PCA fit '''
  return pd.DataFrame(
      [pca.get_factor_variance()[0],
       pca.get_factor_variance()[1],
       pca.get_factor_variance()[2]],
       columns=['PC{}'.format(i) for i in
                range(1,1+len(pca.get_factor_variance()[0]))],
      index=['Sum of Squares Loadings','Proportion of Variance Explained',
             'Cumulative Proportion']
           ).round(round_dig)
```

```
summary = get_summary(attitude_pca)
summary
```

[176]:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	\
Sum of Squares Loadings	14.99	12.18	5.68	1.56	1.36	1.25	1.23	
Proportion of Variance Explained	0.24	0.20	0.09	0.03	0.02	0.02	0.02	
Cumulative Proportion	0.24	0.44	0.53	0.56	0.58	0.60	0.62	

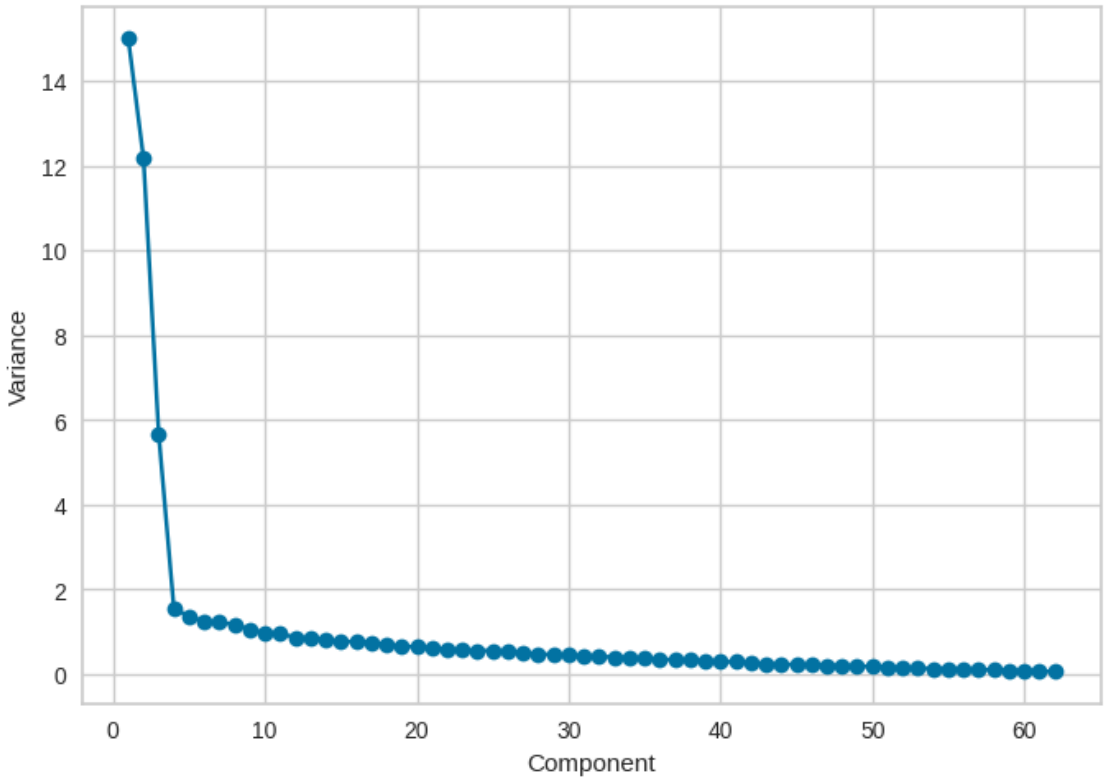
	PC8	PC9	PC10	…	PC53	PC54	PC55	\
Sum of Squares Loadings	1.17	1.07	0.98	…	0.15	0.13	0.13	
Proportion of Variance Explained	0.02	0.02	0.02	…	0.00	0.00	0.00	
Cumulative Proportion	0.64	0.65	0.67	…	0.99	0.99	0.99	

	PC56	PC57	PC58	PC59	PC60	PC61	PC62
Sum of Squares Loadings	0.12	0.11	0.10	0.09	0.09	0.08	0.06
Proportion of Variance Explained	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Cumulative Proportion	0.99	0.99	0.99	1.00	1.00	1.00	1.00

[3 rows x 62 columns]

[177]:
```
# Elbow plot
plt.plot(1+np.arange(len(attitude_pca.get_factor_variance()[0])),
         attitude_pca.get_factor_variance()[0],'o-')
plt.xlabel('Component')
plt.ylabel('Variance')
```

[177]: Text(0, 0.5, 'Variance')



Varimax Rotated PCA

[223]:
```
# Varimax Rotated PCA
attitude_pca_rotated = factor_analyzer.FactorAnalyzer(n_factors=3,␣
 ↪rotation='varimax', method='principal').fit(reg_x)
```

[255]:
```
# Loadings
loadings = get_loadings_communalities(attitude_pca_rotated)
loadings[loadings['communalities'] >= 0.6]
```

[255]:

	RC1	RC2	RC3	communalities
q1	-0.602	-0.282	0.445	0.640
q2	-0.883	0.138	0.128	0.815
q4	0.056	0.668	-0.530	0.730
q5	-0.238	0.820	0.257	0.796
q14	0.867	-0.249	0.207	0.856
q15	0.555	-0.505	0.212	0.607
q17	0.251	-0.863	0.174	0.838
q20	0.851	-0.136	-0.165	0.769
q21	0.574	-0.492	0.202	0.612
q23	-0.873	0.135	0.174	0.811
q24	0.330	0.642	0.414	0.693

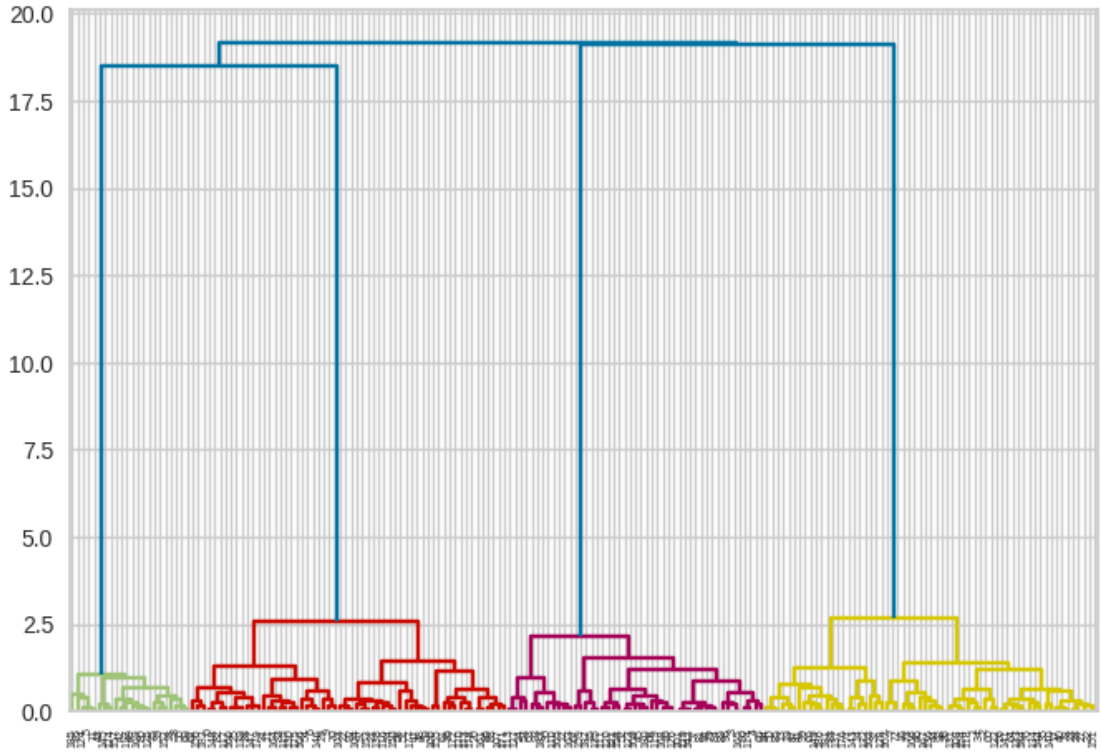
```
q25 -0.523  0.586 -0.227          0.669
q28 -0.493  0.577 -0.161          0.602
q31  0.667  0.489  0.426          0.866
q37  0.292  0.537  0.542          0.668
q41 -0.693 -0.468 -0.405          0.864
q42 -0.198 -0.526 -0.567          0.638
q44 -0.625 -0.640  0.036          0.802
q46 -0.759 -0.240  0.071          0.638
q51  0.760  0.141  0.133          0.615
q52  0.728  0.008  0.477          0.758
q53  0.733 -0.030  0.460          0.750
q55  0.770  0.127  0.020          0.609
```

[225]:
```
# Scores
reg_x_scores = attitude_pca_rotated.transform(reg_x)
df_scores = pd.DataFrame(reg_x_scores,columns=['RC1','RC2', 'RC3'])
df_scores.head(5)
```

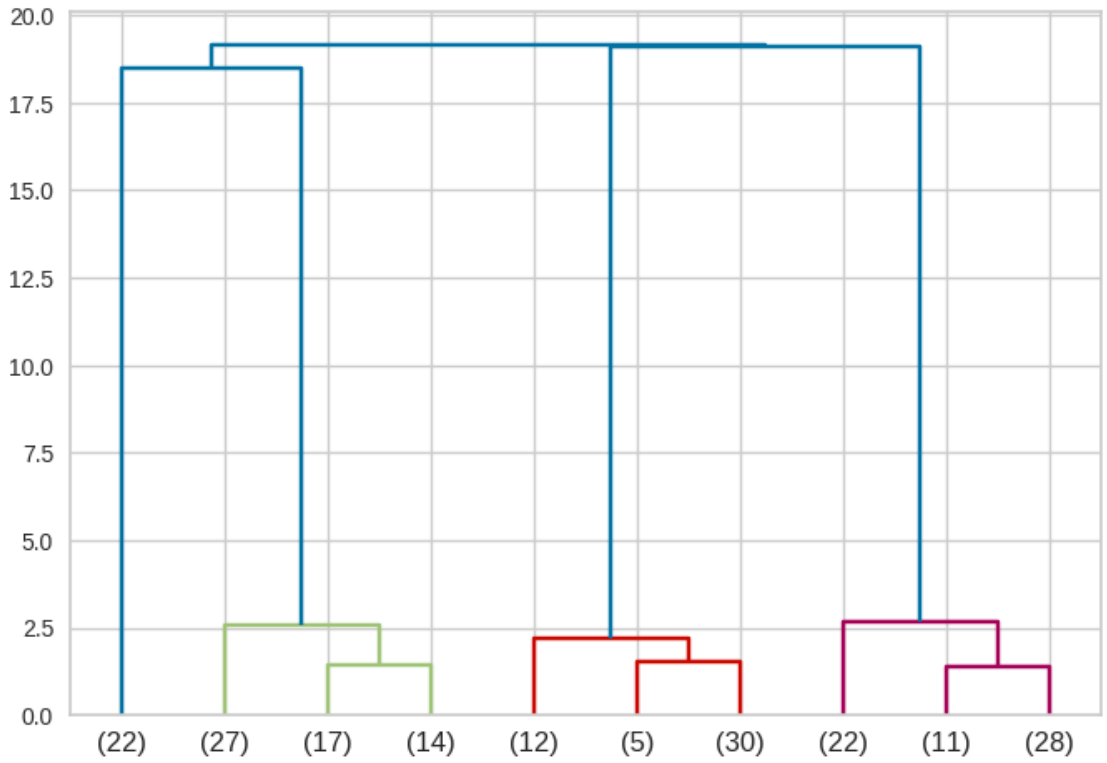
[225]:
```
        RC1       RC2       RC3
0  0.798030 -1.164379 -0.674693
1  0.358847  1.616252  0.224352
2  0.605838  1.417868 -0.078381
3 -1.634792 -0.160789 -0.224294
4 -1.930989 -0.333771 -0.087445
```

Hierarchy Clustering

[226]:
```
# Hierarchy Clustering
np.random.seed(1200)
linkages = hierarchy.linkage(df_scores,method='ward')
hierarchy.dendrogram(linkages)
plt.show()
```



[227]:
```
hierarchy.dendrogram(linkages,orientation='top',
                     truncate_mode='lastp',p=10)
plt.show()
```



[228]:
```
# Cluster values
def check_clusters(data,labels):
  print(list(zip(*np.unique(labels,return_counts=True))))
  return pd.pivot_table(data,index=labels)
```

[239]:
```
# For loop running 2-5 clusters with hierarchy clustering
for i in range(2,6):
  print(' ')
  print(f'hierarchy clustering with {i} clusters:')
  labels_hc = hierarchy.fcluster(linkages,t=i,criterion='maxclust')
  check = check_clusters(df_scores, labels_hc)
  print(check)
```

```
hierarchy clustering with 2 clusters:
[(1, 80), (2, 108)]
        RC1       RC2       RC3
1  0.776572 -0.828568  0.167326
2 -0.575239  0.613754 -0.123945

hierarchy clustering with 3 clusters:
[(1, 80), (2, 47), (3, 61)]
        RC1       RC2       RC3
```

```
1  0.776572 -0.828568  0.167326
2 -1.652590 -0.402407 -0.183701
3  0.254852  1.396698 -0.077903

hierarchy clustering with 4 clusters:
[(1, 22), (2, 58), (3, 47), (4, 61)]
        RC1       RC2       RC3
1  0.475090 -0.625366  2.515196
2  0.890928 -0.905644 -0.723246
3 -1.652590 -0.402407 -0.183701
4  0.254852  1.396698 -0.077903

hierarchy clustering with 5 clusters:
[(1, 22), (2, 58), (3, 47), (4, 22), (5, 39)]
        RC1       RC2       RC3
1  0.475090 -0.625366  2.515196
2  0.890928 -0.905644 -0.723246
3 -1.652590 -0.402407 -0.183701
4  0.344706  1.441591 -0.384377
5  0.204165  1.371373  0.094979
```

Kmeans Clustering

[238]:
```
# For loop running 2-5 clusters with kmeans
for i in range(2,6):
  print(' ')
  centroids_km, labels_km, inertia_km = cluster.
  ↪k_means(df_scores,n_clusters=i,random_state=1200)
  check = check_clusters(df_scores,labels_km)
  print(f'kmeans with {i} clusters:')
  print(check)
```

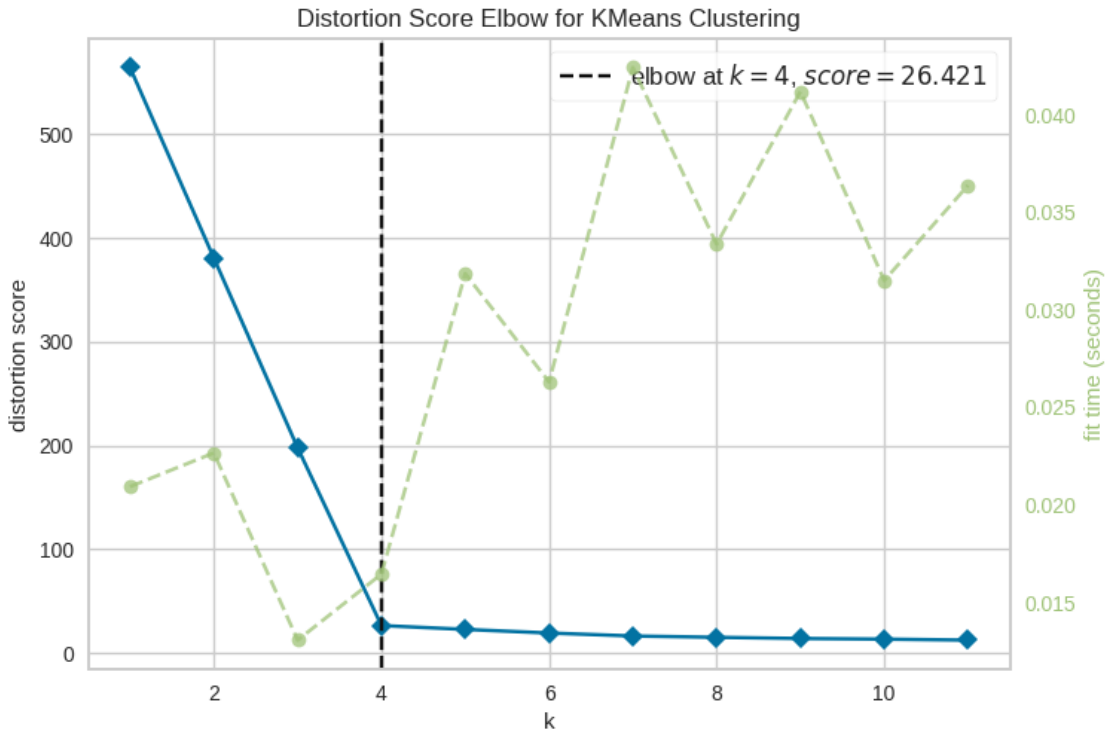
```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(

[(0, 108), (1, 80)]
kmeans with 2 clusters:
        RC1       RC2       RC3
0 -0.575239  0.613754 -0.123945
1  0.776572 -0.828568  0.167326


/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(

[(0, 61), (1, 80), (2, 47)]
kmeans with 3 clusters:
        RC1       RC2       RC3
0  0.254852  1.396698 -0.077903
1  0.776572 -0.828568  0.167326
2 -1.652590 -0.402407 -0.183701

[(0, 61), (1, 58), (2, 47), (3, 22)]
kmeans with 4 clusters:
        RC1       RC2       RC3
0  0.254852  1.396698 -0.077903
1  0.890928 -0.905644 -0.723246
2 -1.652590 -0.402407 -0.183701
3  0.475090 -0.625366  2.515196


/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(

[(0, 47), (1, 16), (2, 58), (3, 22), (4, 45)]
kmeans with 5 clusters:
        RC1       RC2       RC3
0 -1.652590 -0.402407 -0.183701
1  0.348398  1.421494 -0.479260
2  0.890928 -0.905644 -0.723246
3  0.475090 -0.625366  2.515196
4  0.221591  1.387881  0.064801
```

[231]:
```
# Elbow plot
np.random.seed(1200)
model = KMeans()
visualizer = KElbowVisualizer(model, k=(1,12)).fit(df_scores)
visualizer.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```



[231]:
```
<Axes: title={'center': 'Distortion Score Elbow for KMeans Clustering'},
xlabel='k', ylabel='distortion score'>
```

[232]:
```
# Determine 4 clusters
centroids_km, labels_km, inertia_km = cluster.
  ↪k_means(df_scores,n_clusters=4,random_state=1200)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870:
FutureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

[233]:
```
# Check clusters
check_clusters(df_scores, labels_km)
```

```
[(0, 61), (1, 58), (2, 47), (3, 22)]
```

[233]:

	RC1	RC2	RC3
0	0.254852	1.396698	-0.077903
1	0.890928	-0.905644	-0.723246
2	-1.652590	-0.402407	-0.183701
3	0.475090	-0.625366	2.515196

4 Question 6: Relating clusters to demographic

[234]:
```
for c in demo_filtered.columns[1:]:
  ct = pd.crosstab(index = labels_km, columns = demo_filtered[c],␣
 ↪rownames=['Cluster'])
  print(ct)
  chi2, p, dof, expected = chi2_contingency(ct)
  print(f'p-value: {p}')
  print('expected frequency:')
  print(f'{expected}')
  print(' ')
```

```
Preference_Group   1   2
Cluster
0                 29  32
1                 35  23
2                 34  13
3                 18   4
p-value: 0.010495519888543622
expected frequency:
[[37.63829787 23.36170213]
 [35.78723404 22.21276596]
 [29.         18.        ]
 [13.57446809  8.42553191]]

Gender    1   2
Cluster
0        23  38
1        33  25
2        23  24
3        11  11
p-value: 0.2139968987231126
expected frequency:
[[29.20212766 31.79787234]
 [27.76595745 30.23404255]
 [22.5        24.5       ]
 [10.53191489 11.46808511]]

Age      20  21  22  23  24  26  27  28  29  30  …  48  49  50  51  52  54  \
Cluster                                          …
0         2   0   1   1   0   4   2   0   1   4  …   2   2   1   0   0   1
1         3   0   1   1   1   4   1   1   2   2  …   1   1   1   1   1   0
2         0   0   0   0   1   2   3   2   4   2  …   3   1   0   1   1   1
3         0   1   1   0   0   0   2   2   1   0  …   0   0   0   1   0   0

Age      55  56  57  58
Cluster
```

```
0         2   0   1   0
1         1   1   0   1
2         0   0   0   1
3         0   1   0   0

[4 rows x 37 columns]
p-value: 0.7482273186237629
expected frequency:
[[1.62234043 0.32446809 0.97340426 0.64893617 0.64893617 3.24468085
  2.59574468 1.62234043 2.59574468 2.59574468 1.29787234 3.24468085
  0.97340426 3.24468085 1.62234043 1.29787234 1.62234043 0.64893617
  1.94680851 3.89361702 3.24468085 2.2712766  4.21808511 2.92021277
  0.97340426 0.97340426 0.97340426 1.94680851 1.29787234 0.64893617
  0.97340426 0.64893617 0.64893617 0.97340426 0.64893617 0.32446809
  0.64893617]
 [1.54255319 0.30851064 0.92553191 0.61702128 0.61702128 3.08510638
  2.46808511 1.54255319 2.46808511 2.46808511 1.23404255 3.08510638
  0.92553191 3.08510638 1.54255319 1.23404255 1.54255319 0.61702128
  1.85106383 3.70212766 3.08510638 2.15957447 4.0106383  2.77659574
  0.92553191 0.92553191 0.92553191 1.85106383 1.23404255 0.61702128
  0.92553191 0.61702128 0.61702128 0.92553191 0.61702128 0.30851064
  0.61702128]
 [1.25       0.25       0.75       0.5        0.5        2.5
  2.         1.25       2.         2.         1.         2.5
  0.75       2.5        1.25       1.         1.25       0.5
  1.5        3.         2.5        1.75       3.25       2.25
  0.75       0.75       0.75       1.5        1.         0.5
  0.75       0.5        0.5        0.75       0.5        0.25
  0.5       ]
 [0.58510638 0.11702128 0.35106383 0.23404255 0.23404255 1.17021277
  0.93617021 0.58510638 0.93617021 0.93617021 0.46808511 1.17021277
  0.35106383 1.17021277 0.58510638 0.46808511 0.58510638 0.23404255
  0.70212766 1.40425532 1.17021277 0.81914894 1.5212766  1.05319149
  0.35106383 0.35106383 0.35106383 0.70212766 0.46808511 0.23404255
  0.35106383 0.23404255 0.23404255 0.35106383 0.23404255 0.11702128
  0.23404255]]

Marital_Status   1   2   3
Cluster
0               32   5  24
1               27  10  21
2               30   3  14
3               11   2   9
p-value: 0.4306947026762594
expected frequency:
[[32.44680851  6.4893617  22.06382979]
 [30.85106383  6.17021277 20.9787234 ]
 [25.          5.         17.        ]
```

```
 [11.70212766  2.34042553  7.95744681]]

Number_of_Children   0   1   2  3  4
Cluster
0                   34   8  12  6  1
1                   34  13   4  6  1
2                   28  13   4  2  0
3                   11   7   2  2  0
p-value: 0.4718154195044676
expected frequency:
[[34.71808511 13.30319149  7.13829787  5.19148936  0.64893617]
 [33.0106383  12.64893617  6.78723404  4.93617021  0.61702128]
 [26.75       10.25        5.5         4.          0.5       ]
 [12.5212766   4.79787234  2.57446809  1.87234043  0.23404255]]

1st_Time_Purchase  1   2
Cluster
0                  6  55
1                  8  50
2                  3  44
3                  4  18
p-value: 0.44241922211065987
expected frequency:
[[ 6.81382979 54.18617021]
 [ 6.4787234  51.5212766 ]
 [ 5.25       41.75      ]
 [ 2.45744681 19.54255319]]

Age_Category  1   2   3  4   5   6
Cluster
0             4   7  13  3  22  12
1             6   8  11  9  13  11
2             1  11   7  9   9  10
3             2   5   4  1   7   3
p-value: 0.3370190209824496
expected frequency:
[[ 4.21808511 10.05851064 11.35638298  7.13829787 16.54787234 11.68085106]
 [ 4.0106383   9.56382979 10.79787234  6.78723404 15.73404255 11.10638298]
 [ 3.25        7.75        8.75        5.5        12.75        9.        ]
 [ 1.5212766   3.62765957  4.09574468  2.57446809  5.96808511  4.21276596]]

Children_Category   0   1   2
Cluster
0                  34   8  19
1                  34  13  11
2                  28  13   6
3                  11   7   4
p-value: 0.1852492532447845
```

```
expected frequency:
[[34.71808511 13.30319149 12.9787234 ]
 [33.0106383  12.64893617 12.34042553]
 [26.75       10.25       10.        ]
 [12.5212766   4.79787234  4.68085106]]

Income_Category  1   2   3   4   5   6
Cluster
0                4  14  16  12  10   5
1                5   6   9  12  14  12
2                3   9   7   8  12   8
3                4   5   2   3   4   4
p-value: 0.46016254688533265
expected frequency:
[[ 5.19148936 11.03191489 11.03191489 11.35638298 12.9787234   9.40957447]
 [ 4.93617021 10.4893617  10.4893617  10.79787234 12.34042553  8.94680851]
 [ 4.          8.5         8.5         8.75       10.          7.25      ]
 [ 1.87234043  3.9787234   3.9787234   4.09574468  4.68085106  3.39361702]]
```

[ ]: