DimensionReductionWithFAMD

November 4, 2023

[1]:
```
#package prince https://github.com/MaxHalford/prince
#FAMD Factor Analysis of Mixed Data
#For mixture of categorical and numerical data
```

[1]:
```
#Importing the necessary package
import math
import pandas as pd
import numpy as np
from prince import FAMD#Dataset preparation with mixed numerical and categorical␣
 ↪features
df = pd.read_csv('SurveyAnswers.csv')
```

[2]:
```
df=df.set_index('CDR_name')
df.drop(['commercial','opensource'],axis=1,inplace=True)
df
```

[2]:
```
         rest_api another_api  gui  api_n   gui_n aql  sql   json_xml  \
CDR_name
ehrbase         y           y    y   4.00       0   y    n          2
better          y           y    y   4.00       3   y    n          2
base24          y           y    y   4.00       4   n    n          2
cabo            y           y    y   2.91       3   n    n          2
arenaehr        y           n    y   4.00       0   y    n          2
eweave        n             n    y   0.00       3   y    y          0
ehrcare         y           y    n   3.33       0   y    n          1
clever          y           y    y   4.00       3   n    y          0
ehrdb           y           y    y   4.00       3   y    n          2
rhp             y           y    y   4.00       3   n    y          0
ehrn            y           n    y   3.00       2   y    y          0

          flat_struct  opt_wt  openehrextr_fhir  add_openehr  add_others
CDR_name
ehrbase             2       2                 1            1           1
better              2       2                 1            2           4
base24              0       1                 1            1           2
cabo                0       1                 1            0           0
arenaehr            1       2                 1            2           2
```

```
eweave                   0       0                 0            1           2
ehrcare                  1       1                 0            1           1
clever                   1       1                 1            2           3
ehrdb                    2       1                 1            1           3
rhp                      1       1                 0            2           4
ehrn                     1       1                 1            0           1
```

[3]:
```
famd = FAMD(n_components = 2, n_iter = 3, random_state = 101,engine="sklearn")
famd=famd.fit(df)
df_famd = famd.transform(df)
df_famd
```

[3]:
```
                 0         1
CDR_name
ehrbase  -1.355253 -0.847850
better   -1.320558  0.241814
base24   -0.356374  0.962018
cabo     -0.102387  0.257373
arenaehr -0.806795 -0.965018
eweave    3.262109 -0.663478
ehrcare  -0.209372 -1.446912
clever    0.249242  1.496386
ehrdb    -0.871460  0.025389
rhp       0.553719  1.558004
ehrn      0.957130 -0.617726
```

[4]:
```
famd.explained_inertia_ #variance explained
```

[4]:
```
array([0.34520031, 0.20760006])
```

[5]:
```
round(sum(famd.explained_inertia_)*100,1)
```

[5]:
```
55.3
```

[6]:
```
famd.eigenvalues_
```

[6]:
```
array([0.14130948, 0.08498213])
```

[7]:
```
import plotly.express as px
fig=px.scatter(df_famd,x=0,y=1,color=df.index)
fig.show()
```

[8]:
```
famd = FAMD(n_components = 3, n_iter = 3, random_state = 101,engine="sklearn")
famd = famd.fit(df)
df_famd = famd.transform(df)
df_famd
```

```
[8]:                  0         1         2
     CDR_name
     ehrbase  -1.355253 -0.847850 -0.114822
     better   -1.320558  0.241814 -0.978906
     base24   -0.356374  0.962018  1.140254
     cabo     -0.102387  0.257373  1.926911
     arenaehr -0.806795 -0.965018 -0.695325
     eweave    3.262109 -0.663478 -0.296131
     ehrcare  -0.209372 -1.446912  0.404369
     clever    0.249242  1.496386 -0.398770
     ehrdb    -0.871460  0.025389 -0.213718
     rhp       0.553719  1.558004 -0.740811
     ehrn      0.957130 -0.617726 -0.033051
```

```
[9]: famd.explained_inertia_ #variance explained
```

```
[9]: array([0.34520031, 0.20760006, 0.15088997])
```

```
[10]: round(sum(famd.explained_inertia_)*100,1)
```

```
[10]: 70.4
```

```
[11]: famd.eigenvalues_
```

```
[11]: array([0.14130948, 0.08498213, 0.06176757])
```

```
[12]: famd.row_coordinates(df)
```

```
[12]:                 0         1         2
     CDR_name
     ehrbase  -1.355253 -0.847850 -0.114822
     better   -1.320558  0.241814 -0.978906
     base24   -0.356374  0.962018  1.140254
     cabo     -0.102387  0.257373  1.926911
     arenaehr -0.806795 -0.965018 -0.695325
     eweave    3.262109 -0.663478 -0.296131
     ehrcare  -0.209372 -1.446912  0.404369
     clever    0.249242  1.496386 -0.398770
     ehrdb    -0.871460  0.025389 -0.213718
     rhp       0.553719  1.558004 -0.740811
     ehrn      0.957130 -0.617726 -0.033051
```

```
[13]: %matplotlib notebook
      import matplotlib.pyplot as plt

      from mpl_toolkits.mplot3d import Axes3D
from matplotlib import interactive,pyplot
from mpl_toolkits.mplot3d import Axes3D
from numpy.random import rand
from pylab import figure


m=rand(3,3) # m is an array of (x,y,z) coordinate triplets

fig = figure()
ax = fig.add_subplot(projection='3d')
listofloc=[]
mycolors=['r','r','b','b','r','k','r','g','r','g','r']
mymarkers=['s','s','o','o','s','v','s','^','s','^','s']

nloc=[0]*len(df_famd)
for i in range(len(df_famd)): #plot each point + its index as text above
    ax.scatter(df_famd.iloc[i,0],df_famd.iloc[i,1],df_famd.
 ↪iloc[i,2],color=mycolors[i],marker=mymarkers[i])
#     if [df_famd.iloc[i,0],df_famd.iloc[i,1],df_famd.iloc[i,2]] in listofloc:
#         nloc[listofloc.index([df_famd.iloc[i,0],df_famd.iloc[i,1],df_famd.
 ↪iloc[i,2]])]+=1
#         pad=0.12*(nloc[listofloc.index([df_famd.iloc[i,0],df_famd.
 ↪iloc[i,1],df_famd.iloc[i,2]])]-1)
#     else:
#         listofloc.append([df_famd.iloc[i,0],df_famd.iloc[i,1],df_famd.
 ↪iloc[i,2]])
#         nloc[listofloc.index([df_famd.iloc[i,0],df_famd.iloc[i,1],df_famd.
 ↪iloc[i,2]])]=1
#         pad=0
    pad=0
    delta0=0
    delta1=0.05
    delta2=-0.10
    ax.text(df_famd.iloc[i,0]+delta0,df_famd.iloc[i,1]+delta1,df_famd.
 ↪iloc[i,2]+pad+delta2,  '%s' % (df.index[i]), size=9, zorder=1,
    color=mycolors[i])


ax.set_xlabel(f'component 0 {round(famd.explained_inertia_[0]*100,1)}%',␣
 ↪fontsize=14)
ax.set_ylabel(f'component 1 {round(famd.explained_inertia_[1]*100,1)}%',␣
 ↪fontsize=14)
ax.set_zlabel(f'component 2 {round(famd.explained_inertia_[2]*100,1)}%',␣
 ↪fontsize=14)
ax.view_init(30., -35.)
```

```
plt.show()
# plt.savefig('DimensionsReductionWithFAMD_1000dpi.pdf',dpi=1000,format='pdf')
# plt.savefig('DimensionsReductionWithFAMD_1000dpi.tiff',dpi=1000,format='tiff')
```

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

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

[65]:

```
import plotly.express as px

import plotly.graph_objects as go
from plotly.validators.scatter.marker import SymbolValidator

raw_symbol = SymbolValidator().values

labels = {
    str(i): f"Comp {i+1} ({var:.1f}%)"
    for i, var in enumerate(famd.explained_inertia_ * 100)
}
#mycolors=['red','red','blue','blue','red','black','red','green','red','green','red']
#mycolors=[0,0,1,1,0,2,0,3,0,3,0]
mymarkers=['circle','circle','square','square','circle','triangle-down','circle','triangle-up',
#mymarkers=[' ',' ','','',' ','_',' ','-',' ','-',' ']
mycolors=df_famd.index
mysymbols=df_famd.index
#mysymbols=[0,3,9,15,18,21,24,27,33,36,39]
mycolor_discrete_map={'ehrbase':'red','better':'red','base24':'blue','cabo':
 ↪'blue','arenaehr':'red',
                    'eweave':'black','ehrcare':'red','clever':'green','ehrdb':
 ↪'red','rhp':'green','ehrn':'red'}
# mysymbol_map={'ehrbase':'circle','better':'circle','base24':'square','cabo':
 ↪'square','arenaehr':'circle',
#                     'eweave':'triangle-down','ehrcare':'circle','clever':
 ↪'triangle-up','ehrdb':'circle',
#               'rhp':'triangle-up','ehrn':'circle'}
mysymbol_map={'ehrbase':raw_symbol[0*3],'better':raw_symbol[1*3],'base24':
 ↪raw_symbol[3*3],'cabo':raw_symbol[5*3],
              'arenaehr':raw_symbol[6*3],
                    'eweave':raw_symbol[7*3],'ehrcare':raw_symbol[8*3],'clever':
 ↪raw_symbol[9*3],
              'ehrdb':raw_symbol[11*3],
              'rhp':raw_symbol[12*3],'ehrn':raw_symbol[13*3]}
```

```
symbol_sequence=[raw_symbol[25*3],raw_symbol[27*3],raw_symbol[6*3],raw_symbol[5*3],raw_symbol[3
                ␣
 ↪raw_symbol[0*3],raw_symbol[21*3],raw_symbol[9*3],raw_symbol[33*3],raw_symbol[11*3],
                raw_symbol[29*3]]

dimensions=range(3)

#size=11*[0.2]

fig = px.scatter_matrix(
    df_famd,
    labels=labels,
    dimensions=dimensions,
    color_discrete_map=mycolor_discrete_map,
    symbol_sequence=symbol_sequence,#symbol_map=mysymbol_map,
    color=mycolors,
    #size=size,
    symbol=mysymbols
)
fig.update_traces(diagonal_visible=False)
fig.show()
#fig.write_image('DimensionsReductionWithFAMD_2dprojections_1000dpi.pdf',scale=2)
fig.write_image('DimensionsReductionWithFAMD_2dprojections_1000dpi.jpeg',scale=8)
```

[ ]:

```

```