Avinash Navlani
April 12th, 2019

PYTHON +1

Introduction to Factor Analysis in Python

In this tutorial, you'll learn the basics of factor analysis and how to implement it in python.

Factor Analysis (FA) is an exploratory data analysis method used to search influential underlying factors or latent variables from a set of observed variables. It helps in data interpretations by reducing the number of variables. It extracts maximum common variance from all variables and puts them into a common score.

Factor analysis is widely utilized in market research, advertising, psychology, finance, and operation research. Market researchers use factor analysis to identify price-sensitive customers, identify brand features that influence consumer choice, and helps in understanding channel selection criteria for the distribution channel.

In this tutorial, you are going to cover the following topics:

- Factor Analysis
- Types of Factor Analysis
- Determine Number of Factors
- Factor Analysis Vs. Principle Component Analysis
- Factor Analysis in python
- Adequacy Test
- Interpreting the results

- Conclusion

Factor Analysis

Factor analysis is a linear statistical model. It is used to explain the variance among the observed variable and condense a set of the observed variable into the unobserved variable called factors. Observed variables are modeled as a linear combination of factors and error terms (Source). Factor or latent variable is associated with multiple observed variables, who have common patterns of responses. Each factor explains a particular amount of variance in the observed variables. It helps in data interpretations by reducing the number of variables.

Factor analysis is a method for investigating whether a number of variables of interest X1, X2,....., Xl, are linearly related to a smaller number of unobservable factors F1, F2,....., Fk.

Source: This image is recreated from an image that I found in factor analysis notes. The image gives a full view of factor analysis.

Assumptions:

1. There are no outliers in data.
2. Sample size should be greater than the factor.
3. There should not be perfect multicollinearity.
4. There should not be homoscedasticity between the variables.

Types of Factor Analysis

- Exploratory Factor Analysis: It is the most popular factor analysis approach among social and management researchers. Its basic assumption is that any observed variable is directly associated with any factor.
- Confirmatory Factor Analysis (CFA): Its basic assumption is that each factor is associated with a particular set of observed variables. CFA confirms what is expected on the basic.

How does factor analysis work?

conclude the survey. This conversion of the observed variables to unobserved variables can be achieved in two steps:

- **Factor Extraction:** In this step, the number of factors and approach for extraction selected using variance partitioning methods such as principal components analysis and common factor analysis.
- **Factor Rotation:** In this step, rotation tries to convert factors into uncorrelated factors — the main goal of this step to improve the overall interpretability. There are lots of rotation methods that are available such as: Varimax rotation method, Quartimax rotation method, and Promax rotation method.

Terminology

What is a factor?

A factor is a latent variable which describes the association among the number of observed variables. The maximum number of factors are equal to a number of observed variables. Every factor explains a certain variance in observed variables. The factors with the lowest amount of variance were dropped. Factors are also known as latent variables or hidden variables or unobserved variables or Hypothetical variables.

What are the factor loadings?

The factor loading is a matrix which shows the relationship of each variable to the underlying factor. It shows the correlation coefficient for observed variable and factor. It shows the variance explained by the observed variables.

What is Eigenvalues?

Eigenvalues represent variance explained each factor from the total variance. It is also known as characteristic roots.

What are Communalities?

What is Factor Rotation?

Rotation is a tool for better interpretation of factor analysis. Rotation can be orthogonal or oblique. It re-distributed the commonalities with a clear pattern of loadings.

Choosing the Number of Factors

Kaiser criterion is an analytical approach, which is based on the more significant proportion of variance explained by factor will be selected. The eigenvalue is a good criterion for determining the number of factors. Generally, an eigenvalue greater than 1 will be considered as selection criteria for the feature.

The graphical approach is based on the visual representation of factors' eigenvalues also called scree plot. This scree plot helps us to determine the number of factors where the curve makes an elbow.

Source

Factor Analysis Vs. Principle Component Analysis

- PCA components explain the maximum amount of variance while factor analysis explains the covariance in data.
- PCA components are fully orthogonal to each other whereas factor analysis does not require factors to be orthogonal.
- PCA component is a linear combination of the observed variable while in FA, the observed variables are linear combinations of the unobserved variable or factor.
- PCA components are uninterpretable. In FA, underlying factors are labelable and interpretable.
- PCA is a kind of dimensionality reduction method whereas factor analysis is the latent variable method.
- PCA is a type of factor analysis. PCA is observational whereas FA is a modeling technique.

Factor Analysis in python using factor_analyzer package

Import Required Libraries

```
# Import required libraries
import pandas as pd
from sklearn.datasets import load_iris
from factor_analyzer import FactorAnalyzer
import matplotlib.pyplot as plt
```

Loading Data

Let's perform factor analysis on BFI (dataset based on personality assessment project), which were collected using a 6 point response scale: 1 Very Inaccurate, 2 Moderately Inaccurate, 3 Slightly Inaccurate 4 Slightly Accurate, 5 Moderately Accurate, and 6 Very Accurate. You can also download this dataset from the following the link: https://vincentarelbundock.github.io/Rdatasets/datasets.html

```
df= pd.read_csv("bfi.csv")
```

Preprocess Data

```
df.columns
```

```
Index(['A1', 'A2', 'A3', 'A4', 'A5', 'C1', 'C2', 'C3', 'C4', 'C5', 'E1', 'E2',
       'E3', 'E4', 'E5', 'N1', 'N2', 'N3', 'N4', 'N5', 'O1', 'O2', 'O3', 'O4',
       'O5', 'gender', 'education', 'age'],
      dtype='object')
```

```
# Dropping unnecessary columns
df.drop(['gender', 'education', 'age'],axis=1,inplace=True)
```

```
# Dropping missing values rows
df.dropna(inplace=True)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 2436 entries, 0 to 2799
Data columns (total 25 columns):
A1    2436 non-null float64
A2    2436 non-null float64
A3    2436 non-null float64
A4    2436 non-null float64
A5    2436 non-null float64
C1    2436 non-null float64
C2    2436 non-null float64
C3    2436 non-null float64
C4    2436 non-null float64
C5    2436 non-null float64
E1    2436 non-null float64
E2    2436 non-null float64
E3    2436 non-null float64
E4    2436 non-null float64
E5    2436 non-null float64
N1    2436 non-null float64
N2    2436 non-null float64
N3    2436 non-null float64
N4    2436 non-null float64
N5    2436 non-null float64
O1    2436 non-null float64
O2    2436 non-null int64
O3    2436 non-null float64
O4    2436 non-null float64
O5    2436 non-null float64
dtypes: float64(24), int64(1)
memory usage: 494.8 KB
```

```
df.head()
```

A4	A5	C1	C2	C3	C4	C5	...	N1	N2	N3	N4	N5	O1

4.0	4.0	2.0	3.0	3.0	4.0	4.0	...	3.0	4.0	2.0	2.0	3.0	3.0
2.0	5.0	5.0	4.0	4.0	3.0	4.0	...	3.0	3.0	3.0	5.0	5.0	4.0
4.0	4.0	4.0	5.0	4.0	2.0	5.0	...	4.0	5.0	4.0	2.0	3.0	4.0
5.0	5.0	4.0	4.0	3.0	5.0	5.0	...	2.0	5.0	2.0	4.0	1.0	3.0
4.0	5.0	4.0	4.0	5.0	3.0	2.0	...	2.0	3.0	4.0	4.0	3.0	3.0

5 rows × 25 columns

Adequacy Test

Before you perform factor analysis, you need to evaluate the “factorability” of our dataset. Factorability means "can we found the factors in the dataset?". There are two methods to check the factorability or sampling adequacy:

- Bartlett’s Test
- Kaiser-Meyer-Olkin Test

Bartlett’s test of sphericity checks whether or not the observed variables intercorrelate at all using the observed correlation matrix against the identity matrix. If the test found statistically insignificant, you should not employ a factor analysis.

```
from factor_analyzer.factor_analyzer import calculate_bartlett_sphericity
chi_square_value,p_value=calculate_bartlett_sphericity(df)
chi_square_value, p_value
```

```
(18146.065577234807, 0.0)
```

In this Bartlett ’s test, the p-value is 0. The test was statistically significant, indicating that the observed correlation matrix is not an identity matrix.

estimates the proportion of variance among all the observed variable. Lower proportion id more suitable for factor analysis. KMO values range between 0 and 1. Value of KMO less than 0.6 is considered inadequate.

```
from factor_analyzer.factor_analyzer import calculate_kmo
kmo_all,kmo_model=calculate_kmo(df)
```

```
kmo_model
```

```
0.8486452309468382
```

The overall KMO for our data is 0.84, which is excellent. This value indicates that you can proceed with your planned factor analysis.

Choosing the Number of Factors

For choosing the number of factors, you can use the Kaiser criterion and scree plot. Both are based on eigenvalues.

```
# Create factor analysis object and perform factor analysis
fa = FactorAnalyzer()
fa.analyze(df, 25, rotation=None)
# Check Eigenvalues
ev, v = fa.get_eigenvalues()
ev
```

	Original_Eigenvalues
0	5.134311
1	2.751887
2	2.142702
3	1.852328

4	1.548163
5	1.073582
6	0.839539
7	0.799206
8	0.718989
9	0.688089
10	0.676373
11	0.651800
12	0.623253
13	0.596563
14	0.563091
15	0.543305
16	0.514518
17	0.494503
18	0.482640
19	0.448921
20	0.423366
21	0.400671
22	0.387804
23	0.381857
24	0.262539

```
# Create scree plot using matplotlib
plt.scatter(range(1,df.shape[1]+1),ev)
plt.plot(range(1,df.shape[1]+1),ev)
plt.title('Scree Plot')
plt.xlabel('Factors')
plt.ylabel('Eigenvalue')
plt.grid()
plt.show()
```

The scree plot method draws a straight line for each factor and its eigenvalues. Number eigenvalues greater than one considered as the number of factors.

Here, you can see only for 6-factors eigenvalues are greater than one. It means we need to choose only 6 factors (or unobserved variables).

Performing Factor Analysis

```
# Create factor analysis object and perform factor analysis
fa = FactorAnalyzer()
fa.analyze(df, 6, rotation="varimax")


fa.loadings
```

	Factor1	Factor2	Factor3	Factor4	Factor5	Factor6
A1	0.040783	0.095220	0.048734	-0.113057	-0.530987	0.161216
A2	0.235538	0.033131	0.133714	0.063734	0.661141	-0.006244
A3	0.343008	-0.009621	0.121353	0.033990	0.605933	0.160106
A4	0.219717	-0.081518	0.235140	-0.125338	0.404594	0.086356
A5	0.414458	-0.149616	0.106382	0.030977	0.469698	0.236519
C1	0.077248	-0.004358	0.554582	0.190124	0.007511	0.095035

C2	0.038370	0.068330	0.674545	0.087593	0.057055	0.152775
C3	0.031867	-0.039994	0.551164	-0.011338	0.101282	0.008996
C4	-0.066241	0.216283	-0.638475	-0.143846	-0.102617	0.318359
C5	-0.180812	0.284187	-0.544838	0.025837	-0.059955	0.132423
E1	-0.590451	0.022280	0.053915	-0.071205	-0.130851	0.156583
E2	-0.684578	0.233624	-0.088497	-0.045561	-0.116716	0.115065
E3	0.556774	-0.000895	0.103390	0.241180	0.179396	0.267291
E4	0.658395	-0.136788	0.113798	-0.107808	0.241143	0.158513
E5	0.507535	0.034490	0.309813	0.200821	0.078804	0.008747
N1	0.068011	0.805806	-0.051264	-0.074977	-0.174849	-0.096266
N2	0.022958	0.789832	-0.037477	0.006726	-0.141134	-0.139823
N3	-0.065687	0.725081	-0.059039	-0.010664	-0.019184	0.062495
N4	-0.345072	0.578319	-0.162174	0.062916	0.000403	0.147551
N5	-0.161675	0.523097	-0.025305	-0.161892	0.090125	0.120049
O1	0.225339	-0.020004	0.133201	0.479477	0.005178	0.218690
O2	-0.001982	0.156230	-0.086047	-0.496640	0.043989	0.134693
O3	0.325954	0.011851	0.093880	0.566128	0.076642	0.210777
O4	-0.177746	0.207281	-0.005671	0.349227	0.133656	0.178068
O5	-0.014221	0.063234	-0.047059	-0.576743	-0.057561	0.135936

- Factor 1 has high factor loadings for E1,E2,E3,E4, and E5 (Extraversion)
- Factor 2 has high factor loadings for N1,N2,N3,N4, and N5 (Neuroticism)

- Factor 4 has high factor loadings for O1,O2,O3,O4, and O5 (Opennness)
- Factor 5 has high factor loadings for A1,A2,A3,A4, and A5 (Agreeableness)
- Factor 6 has none of the high loagings for any variable and is not easily interpretable. Its good if we take only five factors.

Let's perform factor analysis for 5 factors.

```
# Create factor analysis object and perform factor analysis using 5 factors
fa = FactorAnalyzer()
fa.analyze(df, 5, rotation="varimax")
fa.loadings
```

	Factor1	Factor2	Factor3	Factor4	Factor5
A1	0.040465	0.111126	0.022798	-0.077931	-0.428166
A2	0.213716	0.029588	0.139037	0.062139	0.626946
A3	0.317848	0.009357	0.109331	0.056196	0.650743
A4	0.204566	-0.066476	0.230584	-0.112700	0.435624
A5	0.393034	-0.122113	0.087869	0.066708	0.537087
C1	0.070184	0.010416	0.545824	0.209584	0.038878
C2	0.033270	0.089574	0.648731	0.115434	0.102782
C3	0.023907	-0.030855	0.557036	-0.005183	0.111578
C4	-0.064984	0.240410	-0.633806	-0.107535	-0.037498
C5	-0.176395	0.290318	-0.562467	0.036822	-0.047525
E1	-0.574835	0.042819	0.033144	-0.058795	-0.104813
E2	-0.678731	0.244743	-0.102483	-0.042010	-0.112517

E3	0.536816	0.024180	0.083010	0.280877	0.257906
E4	0.646833	-0.115614	0.102023	-0.073422	0.306101
E5	0.504069	0.036145	0.312899	0.213739	0.090354
N1	0.078923	0.786807	-0.045997	-0.084704	-0.216363
N2	0.027301	0.754109	-0.030568	-0.010304	-0.193744
N3	-0.061430	0.731721	-0.067084	-0.004217	-0.027712
N4	-0.345388	0.590602	-0.178902	0.075225	0.005886
N5	-0.161291	0.537858	-0.037309	-0.149769	0.100931
O1	0.213005	-0.002224	0.115080	0.504907	0.061550
O2	0.004560	0.175788	-0.099729	-0.468925	0.081809
O3	0.310956	0.026736	0.076873	0.596007	0.126889
O4	-0.191196	0.220582	-0.021906	0.369012	0.155475
O5	-0.005347	0.085401	-0.062730	-0.533778	-0.010384

```
# Get variance of each factors
fa.get_factor_variance()
```

	Factor1	Factor2	Factor3	Factor4	Factor5
SS Loadings	2.473090	2.709633	2.041106	1.522153	1.844498
Proportion Var	0.098924	0.108385	0.081644	0.060886	0.073780
Cumulative Var	0.098924	0.207309	0.288953	0.349839	0.423619

Total 42% cumulative Variance explained by the 5 factors.

Pros and Cons of Factor Analysis

related variables, which help the market researchers to compress the market situations and find the hidden relationship among consumer taste, preference, and cultural influence. Also, It helps in improve questionnaire in for future surveys. Factors make for more natural data interpretation.

Results of factor analysis are controversial. Its interpretations can be debatable because more than one interpretation can be made of the same data factors. After factor identification and naming of factors requires domain knowledge.

Conclusion

Congratulations, you have made it to the end of this tutorial!

In this tutorial, you have learned what factor analysis is. The different types of factor analysis, how does factor analysis work, basic factor analysis terminology, choosing the number of factors, comparison of principal component analysis and factor analysis, implementation in python using python FactorAnalyzer package, and pros and cons of factor analysis.

I look forward to hearing any feedback or questions. you can ask the question by leaving a comment and I will try my best to answer it.

If you would like to learn more about factors in Python, take DataCamp's Unsupervised Learning in Python course.