Project Report 3

Name : Dhivakar.R

Course : AI and ML (Aug 2020)

Exploratory Factor Analysis

Problem Statement :

Factor analysis is a useful technique to find latent factors that can potentially describe multiple attributes, which is sometimes very useful for dimensionality reduction. Use the Airline Passenger Satisfaction dataset to perform factor analysis. (Use only the columns that represent the ratings given by the passengers, only 14 columns). Choose the best features possible that helps in dimensionality reduction, without much loss in information.

Prerequisites

What things you need to install the software and how to install them:

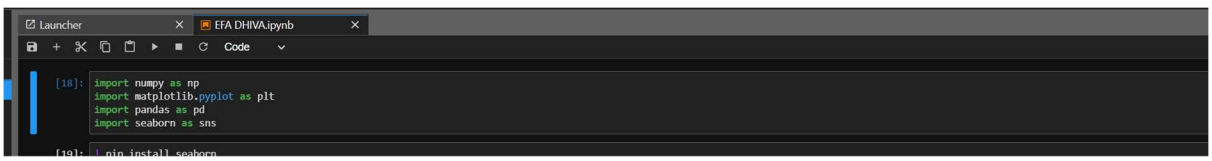
Python 3.6 This setup requires that your machine has latest version of python. The following url https://www.python.org/downloads/ can be referred to download python. Once you have python downloaded and installed, you will need to setup PATH variables (if you want to run python program directly, detail instructions are below in how to run software section). To do that check this: https://www.pythoncentral.io/add-python-to-path-python-is-not-recognized-asan-internal-or-externalcommand/ . Setting up PATH variable is optional as you can also run program without it and more instruction are given below on this topic. Second and easier option is to download anaconda and use its anaconda prompt to run the commands.

To install anaconda check this url https://www.anaconda.com/download/ You will also need to download and install below 3 packages after you install either python or anaconda from the steps above Sklearn (scikit-learn) numpy scipy .

Dataset used:

Dataset Link: https://www.kaggle.com/teejmahal20/airline-passenger-satisfaction

Importing the libraries and loading dataset.



```
[19]: ! pip install seaborn

Requirement already satisfied: seaborn in c:\users\dhiva\anaconda3\lib\site-packages (0.9.0)
Requirement already satisfied: pandas>=0.15.2 in c:\users\dhiva\anaconda3\lib\site-packages (from seaborn) (0.25.1)
Requirement already satisfied: matplotlib>=1.4.3 in c:\users\dhiva\anaconda3\lib\site-packages (from seaborn) (3.1.1)
Requirement already satisfied: scipy>=0.14.0 in c:\users\dhiva\anaconda3\lib\site-packages (from seaborn) (1.3.1)
Requirement already satisfied: numpy>=1.9.3 in c:\users\dhiva\anaconda3\lib\site-packages (from seaborn) (1.16.5)
Requirement already satisfied: python-dateutil>=2.6.1 in c:\users\dhiva\anaconda3\lib\site-packages (from pandas>=0.15.2->seaborn) (2.8.0)
Requirement already satisfied: pytz>=2017.2 in c:\users\dhiva\anaconda3\lib\site-packages (from pandas>=0.15.2->seaborn) (2019.3)
Requirement already satisfied: cycler>=0.10 in c:\users\dhiva\anaconda3\lib\site-packages (from matplotlib>=1.4.3->seaborn) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\dhiva\anaconda3\lib\site-packages (from matplotlib>=1.4.3->seaborn) (1.1.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in c:\users\dhiva\anaconda3\lib\site-packages (from matplotlib>=1.4.3->seaborn) (2.4.2)
Requirement already satisfied: six>=1.5 in c:\users\dhiva\anaconda3\lib\site-packages (from python-dateutil>=2.6.1->pandas>=0.15.2->seaborn) (1.15.0)
Requirement already satisfied: setuptools in c:\users\dhiva\anaconda3\lib\site-packages (from kiwisolver>=1.0.1->matplotlib>=1.4.3->seaborn) (41.4.0)

[20]: train = pd.read_csv('train.csv')
      test = pd.read_csv('test.csv')

[21]: test.shape

[21]: (25976, 25)
```

Zero Centering the data

```
[22]: X_train = train.iloc[:, 8:-3]
      X_test = test.iloc[:, 8:-3]
      print(X_train.shape)
      print(X_test.shape)
      X_train.head()

      (103904, 14)
      (25976, 14)
```

[22]:

	Inflight wifi service	Departure/Arrival time convenient	Ease of Online booking	Gate location	Food and drink	Online boarding	Seat comfort	Inflight entertainment	On-board service	Leg room service	Baggage handling	Checkin service	Inflight service	Cleanliness
0	3	4	3	1	5	3	5	5	4	3	4	4	5	5
1	3	2	3	3	1	3	1	1	1	5	3	1	4	1
2	2	2	2	2	5	5	5	5	4	3	4	4	4	5
3	2	5	5	5	2	2	2	2	2	5	3	1	4	2
4	3	3	3	3	4	5	5	3	3	4	4	3	3	3

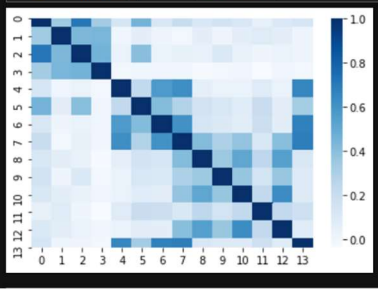
```
[23]: train_target = train.iloc[:, -1:]
      print(train_target.shape)
      np.unique(train_target.values, return_counts = True)

      (103904, 1)
[23]: (array(['neutral or dissatisfied', 'satisfied'], dtype=object),
       array([58879, 45025], dtype=int64))
```

```
[24]: # Zero centering the data
      X = X_train.values
      X_mean = np.mean(X, axis = 0)
      X_n = X - np.matrix(X_mean)
      X_n = X_n.T
      print(X_n.shape)

      (14, 103904)

[25]: c1 = np.cov(X_n)
      c2 = np.corrcoef(X_n)
      ax = sns.heatmap(c2, cmap = "Blues")
```




```
[26]: eig_val, eig_vec = np.linalg.eig(c1)
      eig_sorted = np.sort(eig_val)[::-1]
      arg_sort = np.argsort(eig_val)[::-1]

[27]: print(eig_val)

      [6.52815927 4.47255915 3.43215579 1.98504987 1.61162741 1.18057993
       1.02652179 0.87877104 0.32742411 0.75170941 0.57857082 0.47242866
       0.51571787 0.51339277]

[28]: eig_vec_ls = []
      eig_val_ls = []

[29]: imp_vec = arg_sort[:3]
      for i in imp_vec:
          eig_vec_ls.append(eig_vec[:, i])
          eig_val_ls.append(eig_val[i])
      print(eig_vec_ls)
      print(eig_val_ls)

      [array([0.27033179, 0.15491373, 0.21598175, 0.09124768, 0.32248111,
              0.31012556, 0.35572388, 0.42488029, 0.24281505, 0.20346816,
              0.20154101, 0.16263595, 0.20267719, 0.36360393]), array([ 0.39021937,  0.49052653,  0.51498574,  0.37850549, -0.20260897,
               0.09688795, -0.1994409 , -0.22052189, -0.06650134, -0.02366476,
              -0.04491473, -0.04552343, -0.04930291, -0.2197468 ]), array([ 0.05306717, -0.00824977,  0.07508455,  0.06404967,  0.31940211,
               0.14576517,  0.27020783,  0.0037756 , -0.4376789 , -0.36397705,
              -0.42392401, -0.16410797, -0.43174682,  0.27201769])]
      [6.52815927359577, 4.472559148142908, 3.4321557866762897]
```

[30]:
```
# Estimate V
eig_val_arr = np.array(eig_val_ls)
lambda_1 = np.diag(eig_val_arr)
print(lambda_1)
eig_vec_mat = np.matrix(eig_vec_ls).T
v = eig_vec_mat@np.sqrt(lambda_1)
print(v)
```

```
[[6.52815927 0.         0.        ]
 [0.         4.47255915 0.        ]
 [0.         0.         3.43215579]]
[[ 0.69070483  0.82525254  0.09831266]
 [ 0.39580866  1.03738639 -0.01528359]
 [ 0.55183905  1.08911378  0.13910223]
 [ 0.23314023  0.80047953  0.11865892]
 [ 0.8239477  -0.42848608  0.59172685]
 [ 0.79237896  0.20490276  0.27004569]
 [ 0.90888385 -0.42178611  0.50058914]
 [ 1.08558032 -0.46636909  0.00699471]
 [ 0.62039882 -0.14063987 -0.81084737]
 [ 0.5198665  -0.05004723 -0.67430674]
 [ 0.51494257 -0.09498759 -0.78536494]
 [ 0.41553912 -0.09627489 -0.30402772]
 [ 0.51784555 -0.1042679  -0.79985753]
 [ 0.9290176  -0.46472989  0.50394211]]
```

[31]:
```
# compute std
var_ls = []
X_var = np.var(X_n, axis = 1)
X_var = np.ravel(X_var)
print(X_var.shape)
print(X_var)
for i in range(v.shape[0]):
    s = np.sum(np.square(np.ravel(v[i, :])))
    sig_2 = X_var[i] - s
    var_ls.append(sig_2)
var_ls = np.array(var_ls)
s = np.diag(var_ls)
print(s)
```

```
(14,)
[1.76311414 2.32583197 1.95698483 1.63229974 1.76764022 1.82115689
 1.73997514 1.77684714 1.65984098 1.73079886 1.39451944 1.60121119
 1.38217027 1.72204345]
[[0.59533385 0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         1.09276337 0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.44694024 0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.92309795 0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.55500941 0.
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         1.07838266
  0.         0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.48541226 0.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.38081347 0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.59769326 0.         0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         1.00334337 0.         0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.50353286 0.
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         1.32683672
  0.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.46336238 0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         0.         0.         0.         0.         0.
  0.         0.38903823]]
```

Dimensionality reduction transformation

[32]:
```
# Dimensionality reduction transformation
c1_inv = np.linalg.inv(c1)
w = v.T@c1_inv
print(w.shape)
print(w)
```

```
(3, 14)
[[ 0.10580392  0.06063097  0.08453211  0.03571301  0.1262144   0.12137862
   0.13922513  0.16629195  0.09503427  0.07963447  0.07888021  0.06365334
   0.0793249   0.14230927]
 [ 0.18451462  0.2319447   0.2435102   0.17897573 -0.09580334  0.04581331
  -0.09430532 -0.10427343 -0.03144506 -0.01118984 -0.02123786 -0.02152568
  -0.0233128  -0.10390693]
 [ 0.02864458 -0.00445306  0.04052911  0.03457271  0.17240676  0.07868107
   0.14585269  0.00203799 -0.23625016 -0.1964674  -0.22882555 -0.08858214
  -0.23304814  0.14682961]]
```

[33]:
```
z = w@X_n
z1 = z.T
print(z1)
```

```
[[ 1.17813457 -0.77690998  0.17824378]
 [-1.64286334  0.89139133 -0.56877708]
 [ 1.06568195 -1.37490904  0.5429592 ]
 ...
 [-0.10159264 -1.73737597 -0.01210009]
 [-1.87354186 -0.06824035 -1.3963161 ]
 [-2.16462     0.64493744 -0.26353567]]
```