Content

1. Comparison of loading matrix in Python and R Programming

For the comparison Python and R are figure bellow.

1.1 Python

1.1.1 Code,

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

from matplotlib.pyplot import figure

cell = pd.read_csv('cell_num.csv', index_col=0)

from sklearn.preprocessing import StandardScaler

x = StandardScaler().fit_transform(cell)
x = pd.DataFrame(x, columns=cell.columns)

from sklearn.decomposition import PCA

pcm = PCA()
pca1 = pcm.fit_transform(x)

loading = pcm.components_
```

1.1.2 Explanation,

Standard Scaler

class sklearn.preprocessing.**StandardScaler**(*, *copy=True, with_mean=True, with_std=True*)

Standardize features by removing the mean and scaling to unit variance. The standard score of a sample **x** is calculated as:

z = (x - u) / s

where **u** is the mean of the training samples or zero if ***with_mean=False***, and **s** is the standard deviation of the training samples or one if ***with_std=False***.

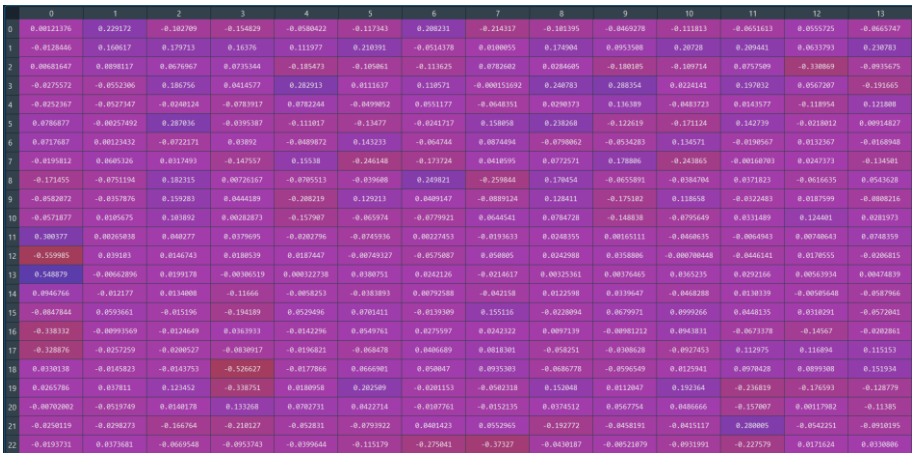
PCA

class sklearn.decomposition.**PCA**(*n_components=None, *, copy=True, whiten=False, svd_solver='auto', tol=0.0, iterated_power='auto', n_oversamples=10, power_iteration_normalizer='auto', random_state=None*)

Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space. The input data is centered but not scaled for each feature before applying the SVD. It uses the LAPACK implementation of the full SVD or a randomized truncated SVD by the method of Halko et al. 2009, depending on the shape of the input data and the number of components to extract. It can also use the ***scipy.sparse.linalg*** ARPACK implementation of the truncated SVD.

1.1.3 Result

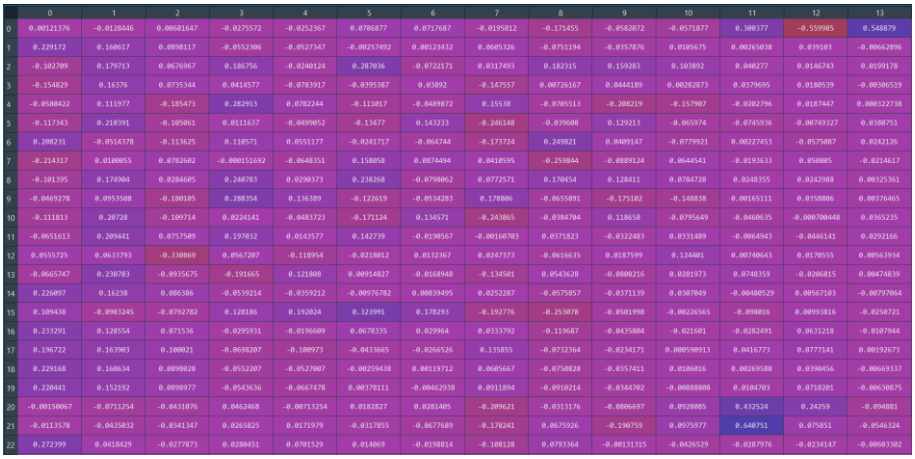
Loading



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.00121376	0.229172	-0.102709	-0.154829	-0.0580422	-0.117343	0.208231	-0.214317	-0.101395	-0.0469278	-0.111813	-0.0651613	0.0555725	-0.0665747
1	-0.0128446	0.160617	0.179713	0.16376	0.111977	0.210391	-0.0514378	0.0100055	0.174904	0.0953508	0.20728	0.209441	0.0633793	0.230783
2	0.00681647	0.0898117	0.0676967	0.0735344	-0.185473	-0.105061	-0.113625	0.0782602	0.0284605	-0.180105	-0.109714	0.0757509	-0.330869	-0.0935675
3	-0.0275572	-0.0552306	0.186756	0.0414577	0.282913	0.0111637	0.110571	-0.000155692	0.240783	0.288354	0.0224141	0.197032	0.0567207	-0.191665
4	-0.0252367	-0.0527347	-0.0240124	-0.0783917	0.0782244	-0.0499052	0.0551177	-0.0648351	0.0290373	0.136389	-0.0483723	0.0143577	-0.118954	0.121808
5	0.0786877	-0.00257492	0.287036	-0.0395387	-0.111017	-0.13477	-0.0241717	0.158058	0.238268	-0.122619	-0.171124	0.142739	-0.0218012	0.00914827
6	0.0717687	0.00123432	-0.0722171	0.03892	-0.0489872	0.143233	-0.064744	0.0874494	-0.0798062	-0.0534283	0.134571	-0.0190567	0.0132367	-0.0168948
7	-0.0195812	0.0605326	0.0317493	-0.147557	0.15538	-0.246148	-0.173724	0.0410595	0.0772571	0.178806	-0.243865	-0.00160703	0.0247373	-0.134501
8	-0.171455	-0.0751194	0.182315	0.00726167	-0.0705513	-0.039608	0.249821	-0.259844	0.170454	-0.0655891	-0.0384704	0.0371823	-0.0616635	0.0543628
9	-0.0582072	-0.0357876	0.159283	0.0444189	-0.208219	0.129213	0.0409147	-0.0889124	0.128411	-0.175102	0.118658	-0.0322483	0.0187599	-0.0808216
10	-0.0571877	0.0105675	0.103892	0.00282873	-0.157907	-0.065974	-0.0779921	0.0644541	0.0784728	-0.148838	-0.0795649	0.0331489	0.124401	0.0281973
11	0.300377	0.00265038	0.040277	0.0379695	-0.0202796	-0.0745936	0.00227453	-0.0193633	0.0248355	0.00165111	-0.0460635	-0.0064943	0.00740643	0.0748359
12	-0.559985	0.039103	0.0146743	0.0180539	0.0187447	-0.00749327	-0.0575087	0.050805	0.0242988	0.0358806	-0.000700448	-0.0446141	0.0170555	-0.0206815
13	0.548879	-0.00662896	0.0199178	-0.00306519	0.00032738	0.0380751	0.0242126	-0.0214617	0.00325361	0.00376465	0.0365235	0.0292166	0.00563934	0.00474839
14	0.0946766	-0.012177	0.0134008	-0.11666	-0.0058253	-0.0383893	0.00792588	-0.042158	0.0122598	0.0339647	-0.0468288	0.0130339	-0.00505648	-0.0587966
15	-0.0847844	0.0593661	-0.015196	-0.194189	0.0529496	0.0701411	-0.0139309	0.155116	-0.0228094	0.0679971	0.0999266	0.0448135	0.0310291	-0.0572041
16	-0.138332	-0.00993569	-0.0124649	0.0363933	-0.0142296	0.0549761	0.0275597	0.0242322	0.0097139	-0.00981212	0.0943831	-0.0673378	-0.14567	-0.0202861
17	-0.328876	-0.0257259	-0.0200527	-0.0830917	-0.0196821	-0.068478	0.0406689	0.0810301	-0.058251	-0.0308628	-0.0927453	0.112975	0.116894	0.115153
18	0.0330138	-0.0145823	-0.0143753	-0.526627	-0.0177866	0.0666901	0.050047	0.0935303	-0.0686778	-0.0596549	0.0125941	0.0970428	0.0899308	0.151934
19	0.0265786	0.037811	0.123452	-0.338751	0.0180958	0.202589	-0.0201153	-0.0502318	0.152048	0.0112047	0.192364	-0.236819	-0.176593	-0.128779
20	-0.00702002	-0.0519749	0.0140178	0.133268	0.0702731	0.0422714	-0.0107761	-0.0152135	0.0374512	0.0567754	0.0486666	-0.157007	0.00117982	-0.11385
21	-0.0250119	-0.0298273	-0.166764	-0.210127	-0.052831	-0.0793922	0.0401423	0.0552965	-0.192772	-0.0458191	-0.0415117	0.280005	-0.0542251	-0.0910195
22	-0.0193731	0.0373681	-0.0669548	-0.0953743	-0.0399644	-0.115179	-0.275841	-0.37327	-0.0430187	-0.00521079	-0.0931991	-0.227579	0.0171624	0.0330806

Figure 1. result of loading matrix in Python

Transpose the loading



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	0.00121376	-0.0128446	0.00681647	-0.0275572	-0.0252367	0.0786877	0.0717687	-0.0195812	-0.171455	-0.0582072	-0.0571877	0.300377	-0.559985	0.548879
1	0.229172	0.160617	0.0898117	-0.0552306	-0.0527347	-0.00257492	0.00123432	0.0605326	-0.0751194	-0.0357876	0.0105675	0.00265038	0.039103	-0.00662896
2	-0.102709	0.179713	0.0676967	0.186756	-0.0240124	0.287036	-0.0722171	0.0317493	0.182315	0.159283	0.103892	0.040277	0.0146743	0.0199178
3	-0.154829	0.16376	0.0735344	0.0414577	-0.0783917	-0.0395387	0.03892	-0.147557	0.00726167	0.0444189	0.00282873	0.0379695	0.0180539	-0.00306519
4	-0.0580422	0.111977	-0.185473	0.282913	0.0782244	-0.111017	-0.0489872	0.15538	-0.0705513	-0.208219	-0.157907	-0.0202796	0.0187447	0.00032738
5	-0.117343	0.210391	-0.105061	0.0111637	-0.0499052	-0.13477	0.143233	-0.246148	-0.039608	0.129213	-0.065974	-0.0745936	-0.00749327	0.0380751
6	0.208231	-0.0514378	-0.113625	0.110571	0.0551177	-0.0241717	-0.064744	-0.173724	0.249821	0.0409147	-0.0779921	0.00227453	-0.0575087	0.0242126
7	-0.214317	0.0100055	0.0782602	-0.000155692	-0.0648351	0.158058	0.0874494	0.0410595	-0.259844	-0.0889124	0.0644541	-0.0193633	0.050805	-0.0214617
8	-0.101395	0.174904	0.0284605	0.240783	0.0290373	0.238268	-0.0798062	0.0772571	0.170454	0.128411	0.0784728	0.0248355	0.0242988	0.00325361
9	-0.0469278	0.0953508	-0.180105	0.288354	0.136389	-0.122619	-0.0534283	0.178806	-0.0655891	-0.175102	-0.148838	0.00165111	0.0358806	0.00376465
10	-0.111813	0.20728	-0.109714	0.0224141	-0.0483723	-0.171124	0.134571	-0.243865	-0.0384704	0.118658	-0.0795649	-0.0460635	-0.000700448	0.0365235
11	-0.0651613	0.209441	0.0757509	0.197032	0.0143577	0.142739	-0.0190567	-0.00160703	0.0371823	-0.0322483	0.0331489	-0.0064943	-0.0446141	0.0292166
12	0.0555725	0.0633793	-0.330869	0.0567207	-0.118954	-0.0218012	0.0132367	0.0247373	-0.0616635	0.0187599	0.124401	0.00740643	0.0170555	0.00563934
13	-0.0665747	0.230783	-0.0935675	-0.191665	0.121808	0.00914827	-0.0168948	-0.134501	0.0543628	-0.0808216	0.0281973	0.0748359	-0.0206815	0.00474839
14	0.226097	0.16238	0.086386	-0.0539214	-0.0359212	-0.00976782	0.00839495	0.0252287	-0.0575857	-0.0371139	0.0307049	-0.00480529	0.00567103	-0.00797064
15	0.109438	-0.0983245	-0.0792782	0.128186	0.192024	0.323991	0.178293	-0.192776	-0.253078	-0.0501998	-0.00226565	-0.098016	0.00993816	-0.0250721
16	0.233291	0.128554	0.071536	-0.0295931	-0.0196609	0.0678335	0.029964	0.0333792	-0.119687	-0.0435804	-0.021601	-0.0202491	0.0631218	-0.0107944
17	0.196722	0.163903	0.100021	-0.0698207	-0.100973	-0.0433665	-0.0266526	0.135855	-0.0732364	-0.0234171	0.000590913	0.0416773	0.0777141	0.00192673
18	0.229168	0.160634	0.0898028	-0.0552207	-0.0527007	-0.00259438	0.00119712	0.0605667	-0.0750828	-0.0357411	0.0106816	0.00269588	0.0390456	-0.00669337
19	0.220441	0.152192	0.0898977	-0.0543636	-0.0667478	0.00178111	-0.00462938	0.0911894	-0.0910214	-0.0344702	-0.00888008	0.0104703	0.0718201	-0.00630075
20	-0.00150067	-0.0711254	-0.0431076	0.0462468	-0.00713254	0.0182827	0.0281405	-0.209621	-0.0313176	-0.0806697	0.0928085	0.432524	0.24259	-0.094881
21	-0.0113578	-0.0435032	-0.0341347	0.0265825	0.0171979	-0.0317855	-0.0677689	-0.178241	0.0675926	-0.190759	0.0975977	0.640751	0.075851	-0.0546324
22	0.272399	0.0418429	-0.0277873	0.0280451	0.0701529	0.014069	-0.0198814	-0.108128	0.0793364	-0.00111315	-0.0426529	-0.0207976	-0.0234147	-0.00603302

Figure 2. result of loading matrix after Transpose

1.2 R

1.2.1 Code,

```
cell <- read.csv('cell_num.csv')[-1]

cell.pca <- prcomp(cell, center = TRUE, scale. =TRUE)
cell.pca

rotation <- cell.pca$rotation # loading matrix
```

```
cell.pca2 <- princomp(cell, cor = FALSE, scores = TRUE)

score2 <- cell.pca2$scores      # score matrix
```

1.2.2 Explanation,

Prcomp(data, Center=True, Scale.=True)

Performs a principal components analysis on the given data matrix and returns the results as an object of class prcomp.

Center=True

a logical value indicating whether the variables should be shifted to be zero centered. Alternately, a vector of length equal the number of columns of x can be supplied. The value is passed to scale.

Scale.=True

a logical value indicating whether the variables should be scaled to have unit variance before the analysis takes place. The default is FALSE for consistency with S, but in general scaling is advisable. Alternatively, a vector of length equal the number of columns of x can be supplied. The value is passed to scale.

Princomp(data, Cor = False, Score = True)

princomp performs a principal components analysis on the given numeric data matrix and returns the results as an object of class princomp.

Cor = False
a logical value indicating whether the calculation should use the correlation matrix or the covariance matrix. (The correlation matrix can only be used if there are no constant variables.)

Score = True
a logical value indicating whether the score on each principal component should be calculated.

1.2.3 Result

Loading

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
AngleCh1	0.001213758	-0.0128446123	0.0068164729	-0.0275571979	0.025236732	0.078687719	-0.071768701	0.019581209	-0.171455127
AreaCh1	0.229171873	0.1606173389	0.0898117268	-0.0552306227	0.052734677	-0.002574919	-0.001234323	-0.060532632	-0.075119429
AvgIntenCh1	-0.102708778	0.1797133188	0.0676967446	0.1867561859	0.024012447	0.287036197	0.072217121	-0.031749278	0.182315055
AvgIntenCh2	-0.154828672	0.1637601833	0.0735343990	0.0414577159	0.078391741	-0.039538733	-0.038920012	0.147557365	0.007261668
AvgIntenCh3	-0.058042158	0.1119770361	-0.1854732857	0.2829129148	-0.078224403	-0.111017286	0.048987210	-0.155380193	-0.070551270
AvgIntenCh4	-0.117343465	0.2103908646	-0.1050609766	0.0111637334	0.049905152	-0.134770023	-0.143233033	0.246148142	-0.039608007
ConvexHullAreaRatioCh1	0.208230625	-0.0514378289	-0.1136247878	0.1105707168	-0.055117714	-0.024171675	0.064744019	0.173723510	0.249821430
ConvexHullPerimRatioCh1	-0.214316603	0.0100055401	0.0782601607	-0.0001516919	0.064835078	0.158058057	-0.087449401	-0.041059550	-0.259843784
DiffIntenDensityCh1	-0.101395191	0.1749037803	0.0284605058	0.2407828341	-0.029037330	0.238268492	0.079806182	-0.077257118	0.170453834
DiffIntenDensityCh3	-0.046927826	0.0953508357	-0.1801045733	0.2883537424	-0.136388911	-0.122618727	0.053428332	-0.178806382	-0.065589084
DiffIntenDensityCh4	-0.111813390	0.2072797056	-0.1097135240	0.0224141010	0.048372282	-0.171123603	-0.134570614	0.243864692	-0.038470449
EntropyIntenCh1	-0.065161252	0.2094408354	0.0757509445	0.1970320543	-0.014357655	0.142739113	0.019056749	0.001607035	0.037182253
EntropyIntenCh3	0.055572513	0.0633792707	-0.3308687444	0.0567207332	0.118953556	-0.021801182	-0.013236685	-0.024737320	-0.061663475
EntropyIntenCh4	-0.066574672	0.2307831122	-0.0935675004	-0.1916651719	-0.121807763	0.009148271	0.016894805	0.134500743	0.054362830
EqCircDiamCh1	0.226096747	0.1623801039	0.0863859766	-0.0539213982	0.035921227	-0.009767816	-0.008394952	-0.025228726	-0.057585651
EqEllipseLWRCh1	0.109437895	-0.0983244641	-0.0792781604	0.1281857649	-0.192023624	0.323990685	-0.178293346	0.192775937	-0.253077857
EqEllipseOblateVolCh1	0.233290590	0.1285538730	0.0715360380	-0.0295931175	0.019660866	0.067833539	-0.029963951	-0.033379160	-0.119687374
EqEllipseProlateVolCh1	0.196721792	0.1639029941	0.1000210739	-0.0698207219	0.100973296	-0.043366493	0.026652589	-0.135854606	-0.073236375
EqSphereAreaCh1	0.229168267	0.1606344567	0.0898027765	-0.0552206946	0.052700671	-0.002594382	-0.001197117	-0.060566689	-0.075082836
EqSphereVolCh1	0.220440861	0.1521917806	0.0898976588	-0.0543635694	0.066747811	0.003781108	0.004629379	-0.091189389	-0.091021393

Figure 3. result of loading matrix in R

1.3 Analysis

So for loading result both in Python and R are have similar result but in the python you need to do transpose if you want have same result as in the R result.