Notebook

March 26, 2021

[1]:
```python
dataset_name = 'hou'
```

[2]:
```python
%reload_ext autoreload
%autoreload 2
default_figsize=(14,12)
```

[3]:
```python
import datasets
import numpy as np
import pandas as pd
import seaborn as sn
import matplotlib.pyplot as plt
import matplotlib
matplotlib.rcParams['figure.figsize'] = (14, 12)


dataset_module = datasets.datasets_by_name_all[dataset_name]
x,y,metadata = dataset_module.load(dropna=True,verbose=True)
y = datasets.map_y_em(y,dataset_name)

# generate dataframe with both x and y
xy = pd.concat([x,y],axis=1)
xy.describe()
```

```
Warning loading data from Hou2016_VPHAS-SDSS-IPHAS-2MASS.csv:
Dropped 27 rows with missing values.
Rows (original):   1034
Rows (after drop): 1007
```

[3]:

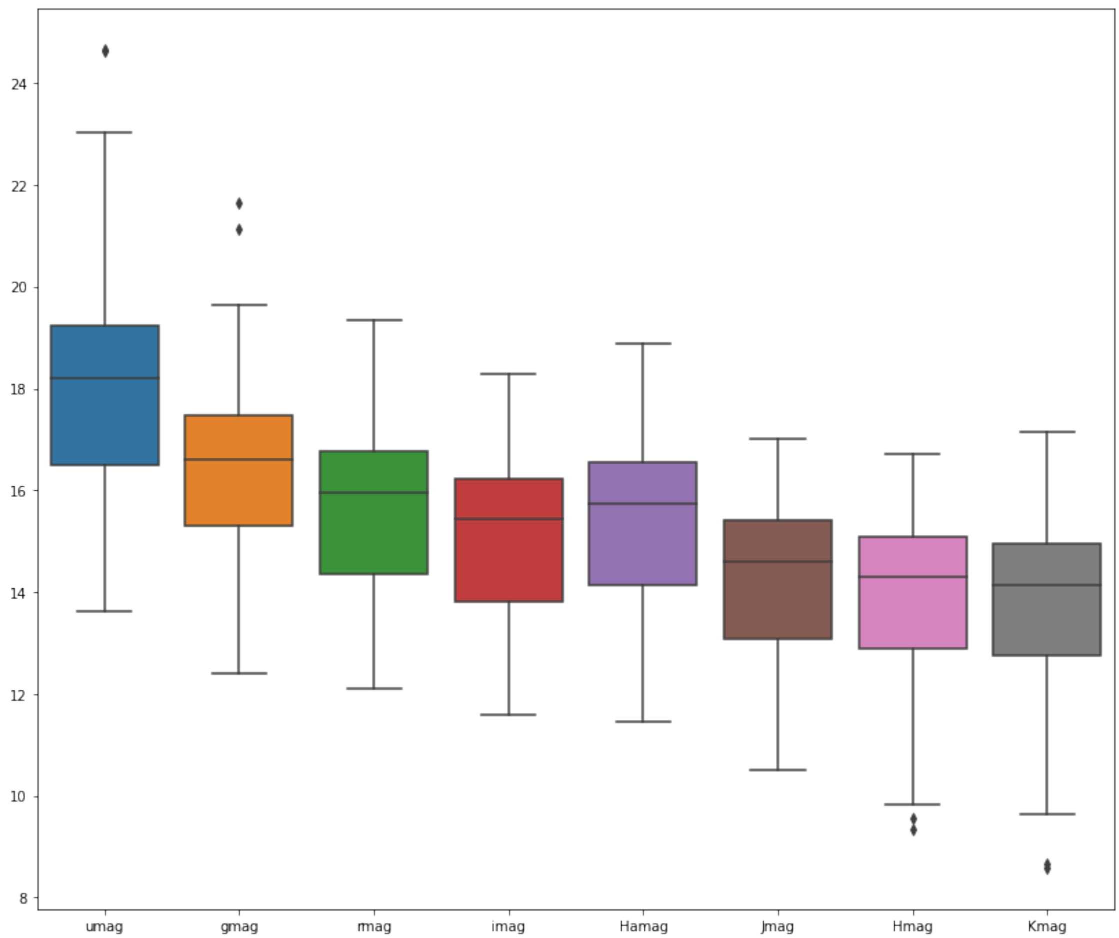
	umag	gmag	rmag	imag	Hamag	\
count	1007.000000	1007.000000	1007.000000	1007.000000	1007.000000	
mean	17.947877	16.366036	15.557746	15.048451	15.347805	
std	1.660195	1.368795	1.418495	1.370818	1.440670	
min	13.616000	12.398000	12.100000	11.590000	11.450000	
25%	16.505000	15.296000	14.365000	13.825000	14.125000	
50%	18.217000	16.618000	15.950000	15.430000	15.750000	
75%	19.226000	17.470500	16.755000	16.225000	16.560000	
max	24.651000	21.633000	19.330000	18.290000	18.890000	

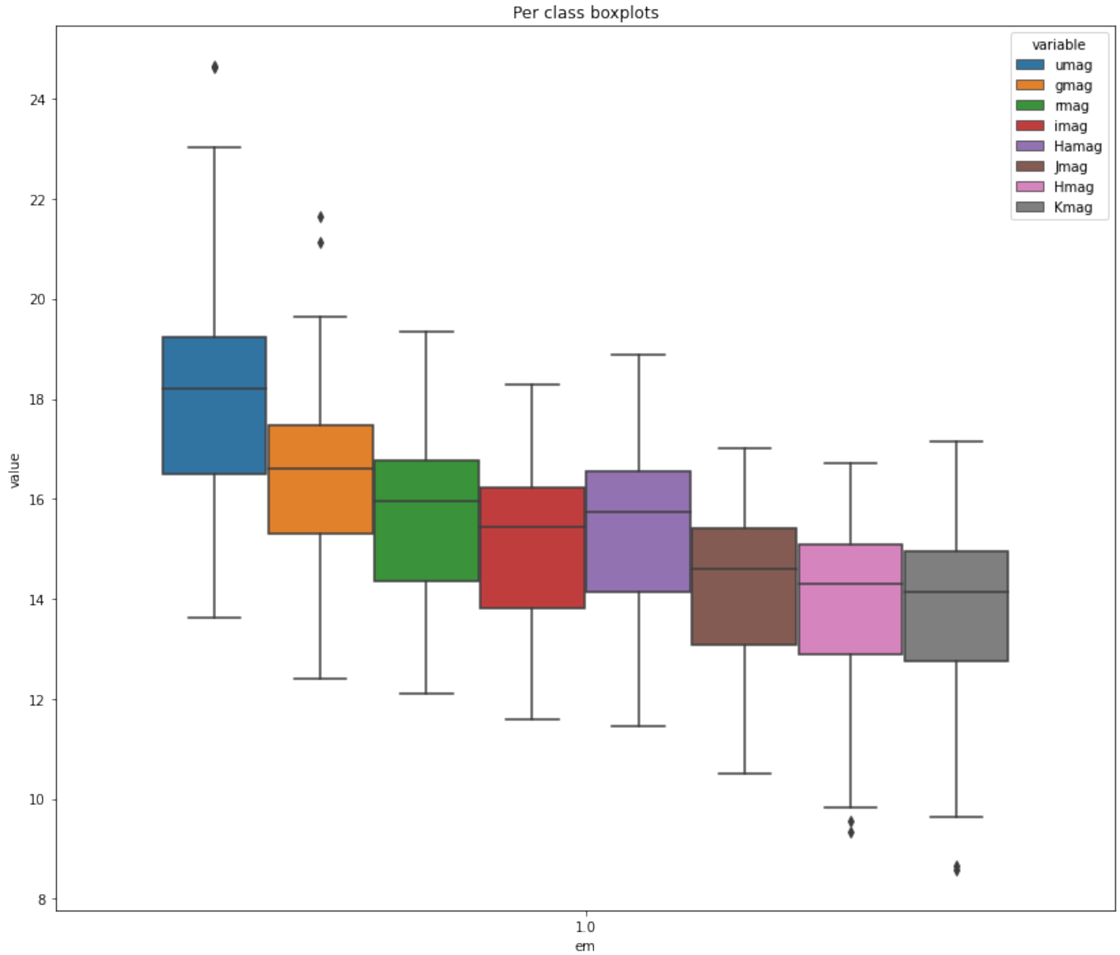
	Jmag	Hmag	Kmag	em
count	1007.000000	1007.000000	1007.000000	1007.0
mean	14.248893	13.983537	13.843248	1.0
std	1.329480	1.331519	1.341729	0.0
min	10.501000	9.331000	8.578000	1.0
25%	13.083000	12.900500	12.767000	1.0
50%	14.586000	14.294000	14.133000	1.0
75%	15.405500	15.085000	14.954000	1.0
max	17.013000	16.700000	17.150000	1.0

1 Variable visualization

[4]:
```
sn.boxplot(data=x)

plt.figure()
xy_long = pd.melt(xy, id_vars='em')
sn.boxplot(x='em', y='value', hue='variable', data=xy_long)
plt.title("Per class boxplots")
```

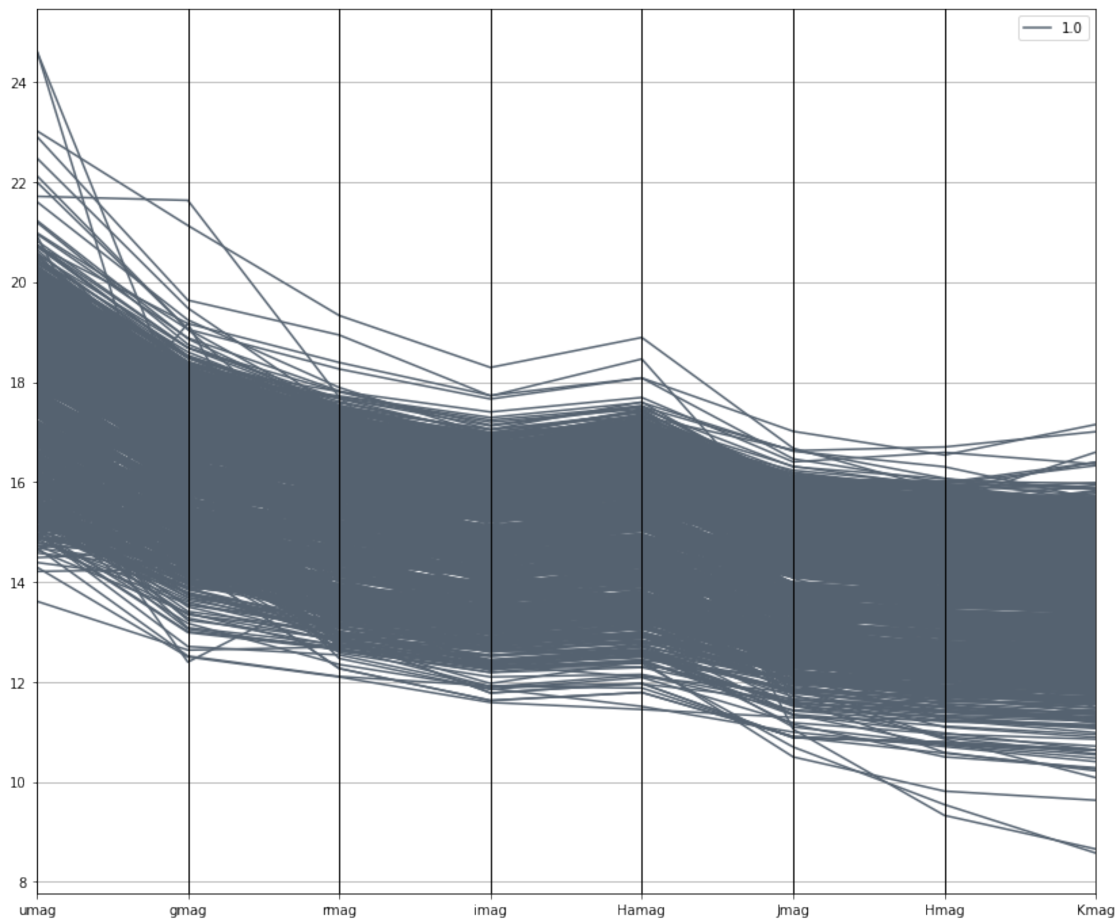
[4]:
```
Text(0.5, 1.0, 'Per class boxplots')
```


24
22
20
18
16
14
12
10
8
umag
gmag
rmag
imag
Hamag
Jmag
Hmag
Kmag



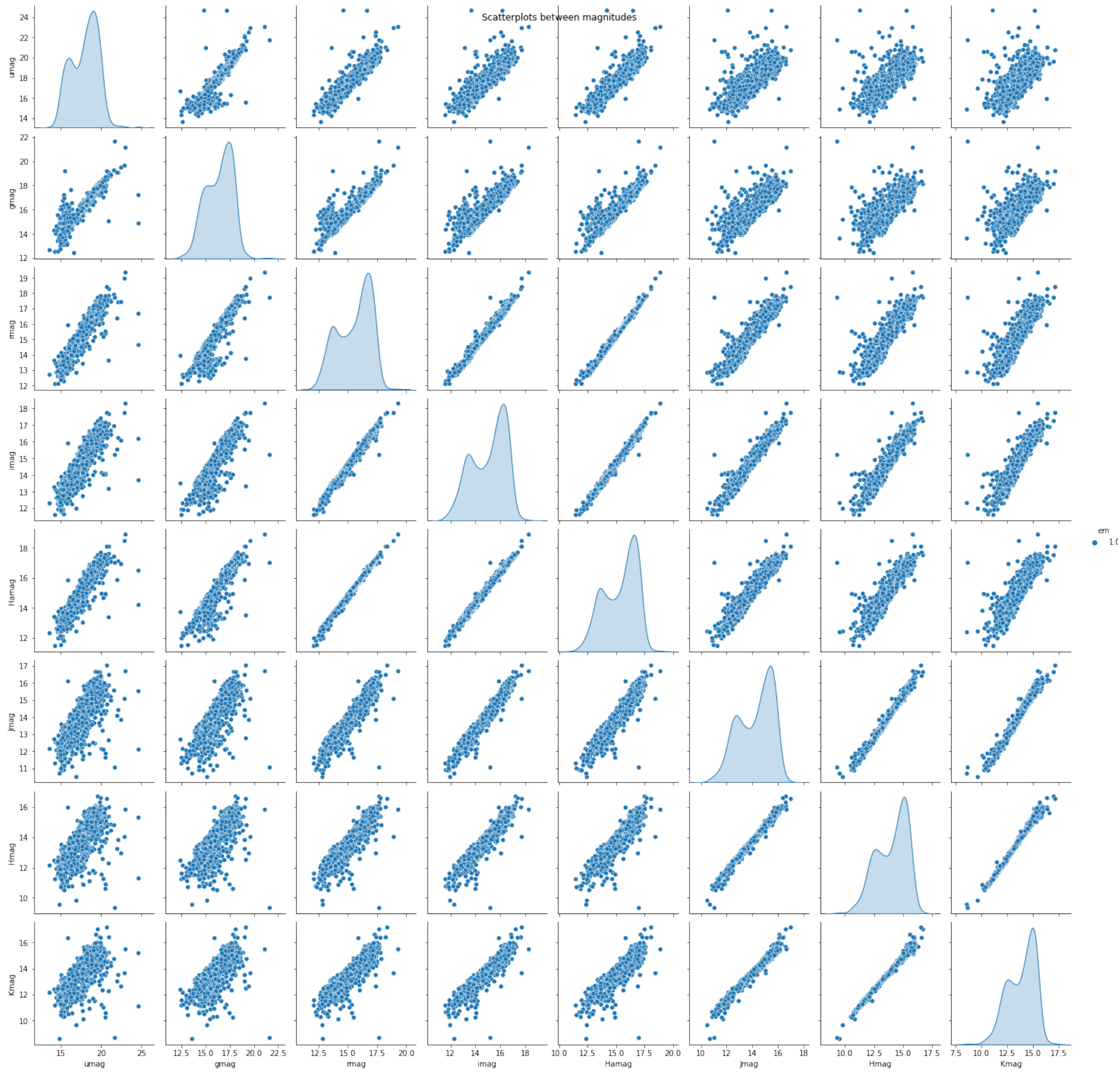

```
[5]: pd.plotting.parallel_coordinates(xy,"em",color=('#556270','#C7F464'))
```

```
[5]: <AxesSubplot:>
```



[6]:
```
sn.heatmap(xy.corr().abs())
plt.title("Correlation between magnitudes")
plt.show()

sn.pairplot(xy,hue="em")
plt.suptitle("Scatterplots between magnitudes")
# axes=pd.plotting.scatter_matrix(x,c=y["em"],alpha=0.
 ↪9,grid=False,figsize=(14,12))
```

[6]: Text(0.5, 0.98, 'Scatterplots between magnitudes')



2 Outlier detection via confidence interval

[7]:
```
from scipy import stats
m = len(x.columns) # number of columns = number of hypothesis
confidence= 0.99
adjusted_confidence = 1- (1-confidence)/m  # bonferroni-adjusted confidence
max_zscore = stats.norm.ppf(adjusted_confidence)
print(f"Confidence  (desired): {confidence}")
print(f"Confidence (adjusted): {adjusted_confidence}")
print(f"Z-score    (adjusted): {max_zscore}")

indices = (np.abs(stats.zscore(x-x.mean())) > max_zscore).any(axis=1)
outliers_x = x[indices]
```

```
if dataset_name != "all_em":
    outliers_metadata = metadata[indices]
    outliers_x = pd.concat([outliers_x,outliers_metadata],axis=1)
outliers_x
```

```
Confidence  (desired): 0.99
Confidence (adjusted): 0.99875
Z-score    (adjusted): 3.023341439739154
```

[7]:

```
       umag    gmag   rmag   imag  Hamag    Jmag    Hmag    Kmag  \
94   23.028  21.130  19.33  18.29  18.89  16.676  15.830  15.471
132  24.635  17.203  16.66  16.17  16.48  15.515  15.300  15.175
622  16.941  15.160  12.83  11.97  12.42  10.501   9.816   9.634
629  24.651  14.845  14.63  13.68  14.19  12.102  11.286  11.082
662  14.853  13.601  12.86  12.31  12.37  10.700   9.547   8.578
683  21.713  21.633  17.70  15.20  17.00  11.054   9.331   8.658

                       ID Fe_type  …    _RA2000      w1  e_umag  k_err  \
94   J053411.98+290903.2      NaN  …  83.549950  15.449   0.300  0.146
132  J052530.75+293821.3      NaN  …  81.378157  15.104   1.578  0.118
622  J062939.48+005504.4      NaN  …  97.414520   9.503   0.009  0.026
629  J055222.83+204152.3      NaN  …  88.095161  10.898  40.354  0.044
662  J055054.77+201447.6      NaN  …  87.728220   7.401   0.004  0.020
683  J064108.31+102408.1      NaN  … 100.284660   8.268   0.133  0.023

          k e_Kmag  h_err  e_Hamag  e_Jmag  Halpha_type
94   15.471  0.146  0.120     0.04   0.112           II
132  15.175  0.118  0.105     0.01   0.061           II
622   9.634  0.026  0.026     0.00   0.024           VI
629  11.082  0.044  0.030     0.00   0.026           VI
662   8.578  0.020  0.029     0.00   0.021            V
683   8.658  0.023  0.024     0.01   0.022           II

[6 rows x 28 columns]
```

3 Outlier detection via IQR

[8]:

```
iqr_factor=1.5
q25,q75=x.quantile(0.25),x.quantile(0.75)
iqr=q75-q25
min_values = q25-iqr_factor*iqr
max_values = q75+iqr_factor*iqr
# ou
indices = (np.logical_or(x<min_values,x>max_values)).any(axis=1)
outliers_x = x[indices]
if dataset_name != "all_em":
```

```
    outliers_metadata = metadata[indices]
    outliers_x = pd.concat([outliers_x,outliers_metadata],axis=1)
outliers_x
```

[8]:

```
        umag    gmag   rmag   imag  Hamag    Jmag    Hmag    Kmag  \
94   23.028  21.130  19.33  18.29  18.89  16.676  15.830  15.471
132  24.635  17.203  16.66  16.17  16.48  15.515  15.300  15.175
629  24.651  14.845  14.63  13.68  14.19  12.102  11.286  11.082
662  14.853  13.601  12.86  12.31  12.37  10.700   9.547   8.578
683  21.713  21.633  17.70  15.20  17.00  11.054   9.331   8.658

                    ID Fe_type  …     _RA2000      w1  e_umag  k_err  \
94   J053411.98+290903.2     NaN  …   83.549950  15.449   0.300  0.146
132  J052530.75+293821.3     NaN  …   81.378157  15.104   1.578  0.118
629  J055222.83+204152.3     NaN  …   88.095161  10.898  40.354  0.044
662  J055054.77+201447.6     NaN  …   87.728220   7.401   0.004  0.020
683  J064108.31+102408.1     NaN  …  100.284660   8.268   0.133  0.023

          k e_Kmag  h_err  e_Hamag  e_Jmag  Halpha_type
94   15.471  0.146  0.120     0.04   0.112           II
132  15.175  0.118  0.105     0.01   0.061           II
629  11.082  0.044  0.030     0.00   0.026           VI
662   8.578  0.020  0.029     0.00   0.021            V
683   8.658  0.023  0.024     0.01   0.022           II

[5 rows x 28 columns]
```

4 Analysis of q-features (q_3) (all magnitudes)

[9]:

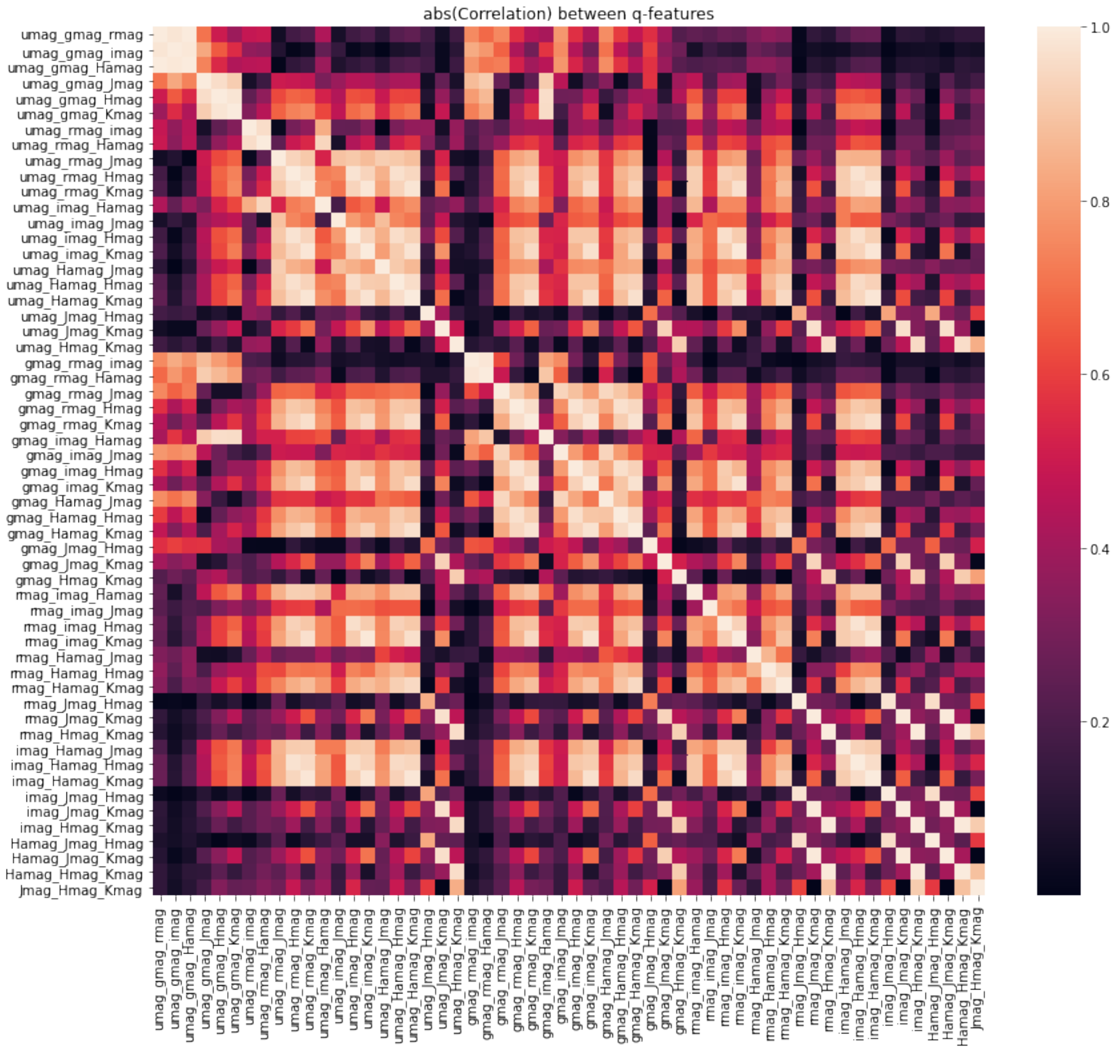
```
x_np=x.to_numpy()
import qfeatures
coefficients = dataset_module.coefficients
systems = dataset_module.systems
coefficients_np = np.array([coefficients[k] for k in x.columns])
systems = [systems[k] for k in x.columns]
q=qfeatures.calculate(x_np,coefficients_np,x.columns,systems,combination_size=3)
m = q.magnitudes

q_df = pd.DataFrame(m, columns = q.column_names)
q_df.describe()
```

[9]:

```
       umag_gmag_rmag  umag_gmag_imag  umag_gmag_Hamag  umag_gmag_Jmag  \
count     1007.000000     1007.000000      1007.000000     1007.000000
mean         1.200440        0.741977         1.063210       -1.623278
std          0.823664        0.872624         0.843513        1.436653
min         -6.198398       -7.366959        -6.524593      -15.935431
```

```
25%        1.178662        0.741553         1.024215         -2.055917
50%        1.369199        0.963304         1.265187         -1.360764
75%        1.489747        1.051404         1.369208         -0.828021
max        9.704550        9.063398         9.472379          5.653403

       umag_gmag_Hmag  umag_gmag_Kmag  umag_rmag_imag  umag_rmag_Hamag  \
count     1007.000000     1007.000000     1007.000000      1007.000000
mean        -4.063644       -7.404559        1.770638         2.186077
std          2.257856        3.456072        0.516441         0.622566
min        -29.070391      -46.138137       -0.026327        -0.079757
25%         -4.958109       -8.856859        1.531942         1.868495
50%         -3.670370       -6.819732        1.732561         2.143327
75%         -2.723609       -5.253641        1.956249         2.447827
max          4.414391        4.119078        8.865444         9.593336

       umag_rmag_Jmag  umag_rmag_Hmag  …  imag_Hamag_Jmag  imag_Hamag_Hmag  \
count     1007.000000     1007.000000  …      1007.000000      1007.000000
mean        -1.391000       -4.728029  …         0.356941         0.975940
std          1.015766        2.267895  …         0.134663         0.385626
min        -15.186556      -33.829435  …        -0.094083        -0.067130
25%         -1.894056       -5.835587  …         0.273833         0.739326
50%         -1.311667       -4.451217  …         0.339347         0.922130
75%         -0.854889       -3.377304  …         0.421736         1.161500
max          4.667222        1.825435  …         1.751083         5.368848

       imag_Hamag_Kmag  imag_Jmag_Hmag  imag_Jmag_Kmag  imag_Hmag_Kmag  \
count      1007.000000     1007.000000     1007.000000     1007.000000
mean          1.814892        0.228467       -0.512821        0.491838
std           0.720126        0.231544        0.514158        0.610817
min          -0.639804       -0.871457       -5.255294       -2.465464
25%           1.356389        0.099630       -0.738353        0.206157
50%           1.719804        0.231152       -0.444235        0.462778
75%           2.188350        0.353446       -0.242647        0.749595
max           9.922418        1.406913        2.277294        5.374222

       Hamag_Jmag_Hmag  Hamag_Jmag_Kmag  Hamag_Hmag_Kmag  Jmag_Hmag_Kmag
count      1007.000000      1007.000000      1007.000000     1007.000000
mean          0.279772        -0.783490         0.594054        0.146156
std           0.341645         0.754679         0.813179        0.185327
min          -1.889261        -8.177190        -3.191216       -0.707641
25%           0.092609        -1.085876         0.228765        0.047830
50%           0.280348        -0.676980         0.558020        0.132556
75%           0.472978        -0.394304         0.937892        0.221882
max           1.874522         3.231967         7.073667        1.382222

[8 rows x 56 columns]
```

[10]:
```
sn.heatmap(q_df.corr().abs())
plt.title("abs(Correlation) between q-features")
plt.show()
```



5 Analysis of q-features (q_4) (calculated by system to avoid combinatory explosion)

[11]:
```
x_np=x.to_numpy()
import qfeatures
coefficients = dataset_module.coefficients
systems = dataset_module.systems
coefficients_np = np.array([coefficients[k] for k in x.columns])
systems = [systems[k] for k in x.columns]
```

```
q= qfeatures.calculate(x_np,coefficients_np,x.
 ↪columns,systems,combination_size=4,by_system=True)

m = q.magnitudes

q_df = pd.DataFrame(m, columns = q.column_names)
q_df.describe()
```

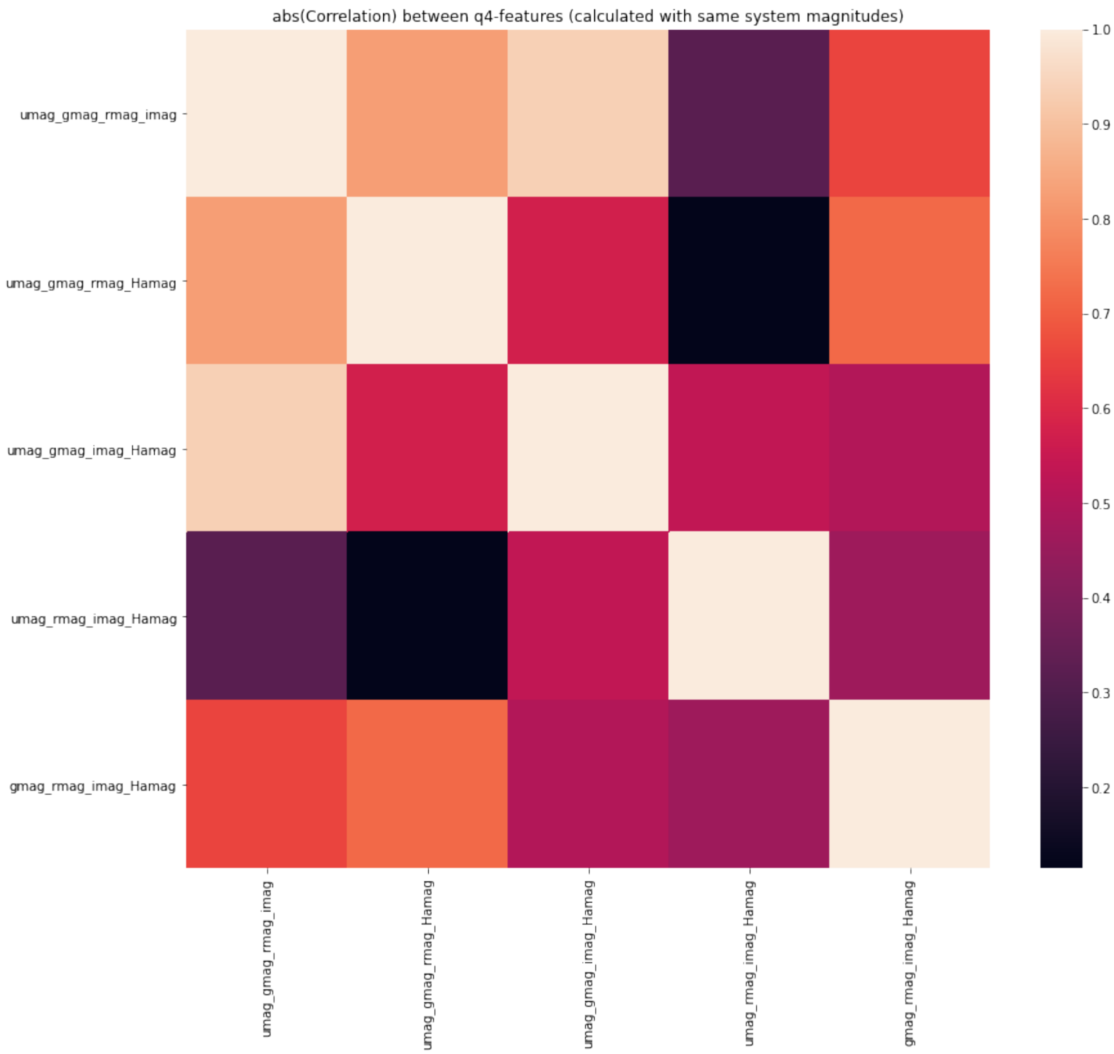
[11]:

	umag_gmag_rmag_imag	umag_gmag_rmag_Hamag	umag_gmag_imag_Hamag
count	1007.000000	1007.000000	1007.000000
mean	0.656622	0.235753	0.823012
std	0.629001	0.950934	0.606004
min	-4.461667	-5.723059	-4.482791
25%	0.576500	-0.009324	0.754326
50%	0.768667	0.491529	0.875488
75%	0.883667	0.738000	0.994628
max	8.080167	6.984824	8.513209

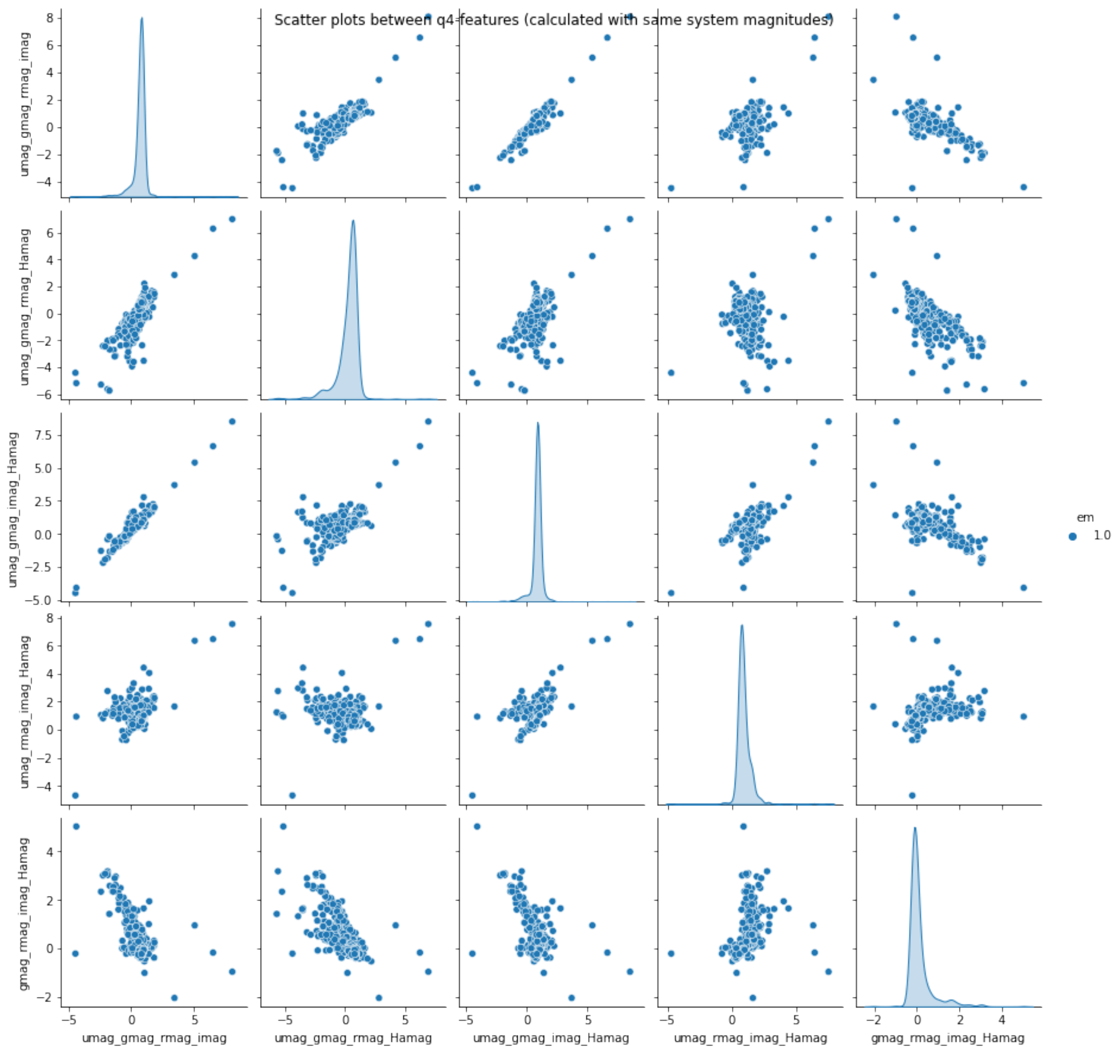
	umag_rmag_imag_Hamag	gmag_rmag_imag_Hamag
count	1007.000000	1007.000000
mean	0.942091	0.119078
std	0.590319	0.577309
min	-4.693977	-2.038512
25%	0.662663	-0.162698
50%	0.839070	-0.064628
75%	1.095535	0.129860
max	7.554023	5.026581

[12]:

```
sn.heatmap(q_df.corr().abs())
_=plt.title("abs(Correlation) between q4-features (calculated with same system␣
 ↪magnitudes)")
```



[13]:
```
q_dfy=pd.concat([q_df,y],axis=1)
sn.pairplot(q_dfy,hue="em")
_=plt.suptitle("Scatter plots between q4-features (calculated with same system␣
↪magnitudes)")
```


Scatter plots between q4-features (calculated with same system magnitudes)
umag_gmag_rmag_imag
umag_gmag_rmag_Hamag
umag_gmag_imag_Hamag
umag_rmag_imag_Hamag
gmag_rmag_imag_Hamag
em
1.0