HW1_Prob2

September 5, 2018

1 Read Data Set and Import Libraries

```
In [1]: from fancyimpute import KNN
        import pandas as pd
        import numpy as np
        #import matplotlib.pyplot as plt
        import os
        from sklearn import preprocessing
        from sklearn.preprocessing import Imputer
        current_path = os.getcwd()
        data_path ="DataSet"
        file_name = "data_akbilgic.xlsx"

        path = os.path.join(current_path,data_path)
        path = os.path.join(path,file_name)
        df = pd.read_excel(path)
        df
```

```
Using TensorFlow backend.
```

Out[1]:

	date	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	\
0	2009-01-05	0.035754	0.038376	-0.004679	0.002193	0.003894	0.000000	
1	2009-01-06	0.025426	0.031813	0.007787	0.008455	0.012866	0.004162	
2	2009-01-07	-0.028862	-0.026353	-0.030469	-0.017833	-0.028735	0.017293	
3	2009-01-08	-0.062208	-0.084716	0.003391	-0.011726	-0.000466	-0.040061	
4	2009-01-09	0.009860	0.009658	-0.021533	-0.019873	-0.012710	-0.004474	
5	2009-01-12	-0.029191	-0.042361	-0.022823	-0.013526	-0.005026	-0.049039	
6	2009-01-13	0.015445	-0.000272	0.001757	-0.017674	-0.006141	0.000000	
7	2009-01-14	-0.041168	-0.035552	-0.034032	-0.047383	-0.050945	0.002912	
8	2009-01-15	0.000662	-0.017268	0.001328	-0.019551	-0.014335	-0.050448	
9	2009-01-16	0.022037	0.032278	0.007533	0.006791	0.006289	0.025453	
10	2009-01-19	-0.022692	-0.044349	-0.054262	-0.011550	-0.009351	0.003239	
11	2009-01-20	-0.013709	-0.029661	0.000000	-0.017834	-0.004171	-0.023411	
12	2009-01-21	0.000865	0.001529	0.042572	0.005011	-0.007729	-0.020561	
13	2009-01-22	-0.003815	0.005043	-0.015278	-0.009841	-0.001898	0.018818	
14	2009-01-23	0.005661	-0.010008	0.005363	-0.009640	0.000074	-0.038808	
15	2009-01-26	0.046831	0.061708	0.005538	0.034787	0.037891	-0.008182	

16	2009-01-27	-0.006635	0.010949	0.010866	-0.000798	-0.003475	0.048148
17	2009-01-28	0.034567	0.035871	0.033007	0.044182	0.023748	0.005594
18	2009-01-29	-0.020528	-0.020272	-0.033681	-0.020256	-0.024774	0.017723
19	2009-01-30	-0.008777	-0.023458	-0.023053	-0.020479	-0.009713	-0.031666
20	2009-02-02	-0.025919	-0.035607	-0.000533	-0.015637	-0.017454	-0.015134
21	2009-02-03	0.015279	0.022403	0.015710	0.024040	0.021039	-0.006175
22	2009-02-04	0.018578	0.023231	-0.007518	0.026577	0.015275	0.026908
23	2009-02-05	-0.014133	-0.014571	0.016233	0.003932	0.000071	-0.011169
24	2009-02-06	0.036607	0.042759	0.026541	0.029306	0.014788	0.015846
25	2009-02-09	0.011353	0.021468	0.001484	0.004766	0.003651	-0.013411
26	2009-02-10	-0.040542	-0.043907	-0.050369	-0.035170	-0.022182	-0.002902
27	2009-02-11	-0.022106	-0.033893	0.007923	0.005434	0.005019	-0.030745
28	2009-02-12	-0.014888	-0.020825	0.001738	-0.027421	-0.007610	0.000000
29	2009-02-13	0.007027	0.009709	-0.010048	0.001322	-0.003003	0.009563
..	...	...	...	...	...	...	...
506	2011-01-12	0.003761	0.009064	0.008967	0.018160	0.006084	0.000202
507	2011-01-13	0.009547	0.016298	-0.001712	0.000895	-0.004439	0.007294
508	2011-01-14	-0.010199	-0.004700	0.007357	0.000083	-0.003625	-0.008604
509	2011-01-17	-0.015563	-0.015368	0.001375	0.000333	-0.002736	0.000364
510	2011-01-18	-0.006049	0.004580	0.000000	0.009196	0.011742	0.001534
511	2011-01-19	0.000521	-0.003209	-0.010167	-0.008532	-0.013247	0.003617
512	2011-01-20	-0.017835	-0.031652	-0.001296	-0.008292	-0.018372	-0.011412
513	2011-01-21	0.009744	-0.000511	0.002411	0.005416	0.004828	-0.015720
514	2011-01-24	-0.011067	-0.007426	0.005819	0.000757	0.008040	0.006847
515	2011-01-25	-0.000749	0.004639	0.000263	-0.001240	-0.004418	0.011467
516	2011-01-26	0.012326	0.005404	0.004212	0.009635	0.008665	-0.005992
517	2011-01-27	-0.014082	-0.014976	0.002242	0.003953	-0.000687	0.007352
518	2011-01-28	-0.028498	-0.038599	-0.018014	-0.007403	-0.014131	-0.011356
519	2011-01-31	0.001056	-0.008883	0.007633	-0.003571	-0.003150	-0.011887
520	2011-02-01	0.024116	0.035257	0.016556	0.014976	0.016057	0.003567
521	2011-02-02	0.007447	0.014239	-0.002726	-0.000084	0.007075	0.017641
522	2011-02-03	-0.024478	-0.027850	0.002351	0.001392	-0.002804	-0.002489
523	2011-02-04	0.024507	0.013959	0.002880	0.003127	0.002354	0.010695
524	2011-02-07	-0.006196	0.008553	0.006221	0.009298	0.008898	0.004591
525	2011-02-08	0.005356	0.006886	0.004176	0.005425	0.006638	0.004140
526	2011-02-09	0.004823	-0.003255	-0.002790	-0.000320	-0.006423	-0.001708
527	2011-02-10	-0.017664	-0.024921	0.000749	0.002644	-0.005351	-0.001148
528	2011-02-11	0.004782	0.006418	0.005492	0.004204	0.007101	0.011241
529	2011-02-14	-0.002498	0.000405	0.002382	0.003444	-0.000462	0.000000
530	2011-02-15	0.003606	0.000893	-0.003240	0.000461	-0.003803	0.001968
531	2011-02-16	0.008599	0.013400	0.006238	0.001925	0.007952	0.005717
532	2011-02-17	0.009310	0.015977	0.003071	-0.001186	0.000345	0.002620
533	2011-02-18	0.000191	-0.001653	0.001923	0.002872	-0.000723	0.000568
534	2011-02-21	-0.013069	-0.013706	-0.020742	-0.014239	-0.011275	0.001358
535	2011-02-22	-0.007246	-0.019442	0.000000	-0.000473	-0.002997	-0.017920

	BOVESPA	EU	EM
0	0.031190	0.012698	0.028524

1	0.018920	0.011341	0.008773
2	-0.035899	-0.017073	-0.020015
3	0.028283	-0.005561	-0.019424
4	-0.009764	-0.010989	-0.007802
5	-0.053849	-0.012451	-0.022630
6	0.003572	-0.012220	-0.004827
7	-0.040302	-0.045220	-0.008677
8	0.030314	-0.012070	-0.023429
9	0.004867	0.008561	0.010917
10	-0.013151	-0.012045	-0.004029
11	-0.040899	-0.015088	-0.024107
12	0.033532	-0.003339	-0.005092
13	-0.016982	-0.006552	-0.003227
14	0.006261	-0.003620	-0.008077
15	0.009838	0.032800	0.010320
16	0.004922	-0.002642	0.006344
17	0.038725	0.029974	0.022104
18	-0.014750	-0.023109	0.000409
19	-0.008538	-0.007201	0.002243
20	-0.016289	-0.019739	-0.019091
21	0.027574	0.017862	0.012719
22	0.009565	0.018770	0.015166
23	0.024128	-0.004139	0.002073
24	0.039282	0.019127	0.032338
25	-0.015462	0.005627	0.007895
26	-0.021440	-0.024388	-0.002139
27	-0.008799	0.001097	-0.007926
28	-0.008482	-0.014092	-0.014773
29	0.028551	0.003032	0.017764
..	...	...	...
506	0.017036	0.014397	0.011872
507	-0.012813	-0.000367	-0.002109
508	0.003092	-0.000761	0.000238
509	-0.004677	-0.002147	-0.004537
510	0.004395	0.010995	0.002406
511	-0.012229	-0.010645	0.001086
512	-0.007105	-0.010088	-0.010442
513	-0.006186	0.008396	-0.006854
514	0.004244	0.004158	-0.001145
515	-0.010396	-0.004044	-0.000885
516	0.000000	0.007097	0.005019
517	-0.009623	0.001919	-0.001149
518	-0.020082	-0.010478	-0.009912
519	-0.001846	-0.002650	-0.005789
520	0.018926	0.015089	0.006224
521	-0.017230	0.001618	0.003631
522	0.001154	-0.002883	0.000476
523	-0.022662	0.002761	-0.003185

```
524  0.001424  0.008217 -0.003346
525  0.006238  0.003980 -0.004499
526 -0.023895 -0.003024 -0.014249
527  0.005590 -0.003742 -0.014760
528  0.018077  0.004727  0.003931
529  0.012123  0.000169  0.013448
530 -0.003266 -0.000550 -0.001430
531  0.018371  0.006975  0.003039
532  0.001686 -0.000581  0.001039
533  0.005628  0.000572  0.006938
534 -0.011942 -0.012615 -0.000958
535 -0.012252 -0.005465 -0.014297

[536 rows x 10 columns]
```

2 Removing Date

In [2]:
```
df_removed_date = df.drop("date",axis=1)

NUM_ROWS = df_removed_date.shape[0]
NUM_COLS = df_removed_date.shape[1]
df_removed_date
```

Out[2]:
```
          ISE     ISE.1        SP       DAX      FTSE    NIKKEI   BOVESPA  \
0    0.035754  0.038376 -0.004679  0.002193  0.003894  0.000000  0.031190
1    0.025426  0.031813  0.007787  0.008455  0.012866  0.004162  0.018920
2   -0.028862 -0.026353 -0.030469 -0.017833 -0.028735  0.017293 -0.035899
3   -0.062208 -0.084716  0.003391 -0.011726 -0.000466 -0.040061  0.028283
4    0.009860  0.009658 -0.021533 -0.019873 -0.012710 -0.004474 -0.009764
5   -0.029191 -0.042361 -0.022823 -0.013526 -0.005026 -0.049039 -0.053849
6    0.015445 -0.000272  0.001757 -0.017674 -0.006141  0.000000  0.003572
7   -0.041168 -0.035552 -0.034032 -0.047383 -0.050945  0.002912 -0.040302
8    0.000662 -0.017268  0.001328 -0.019551 -0.014335 -0.050448  0.030314
9    0.022037  0.032278  0.007533  0.006791  0.006289  0.025453  0.004867
10  -0.022692 -0.044349 -0.054262 -0.011550 -0.009351  0.003239 -0.013151
11  -0.013709 -0.029661  0.000000 -0.017834 -0.004171 -0.023411 -0.040899
12   0.000865  0.001529  0.042572  0.005011 -0.007729 -0.020561  0.033532
13  -0.003815  0.005043 -0.015278 -0.009841 -0.001898  0.018818 -0.016982
14   0.005661 -0.010008  0.005363 -0.009640  0.000074 -0.038808  0.006261
15   0.046831  0.061708  0.005538  0.034787  0.037891 -0.008182  0.009838
16  -0.006635  0.010949  0.010866 -0.000798 -0.003475  0.048148  0.004922
17   0.034567  0.035871  0.033007  0.044182  0.023748  0.005594  0.038725
18  -0.020528 -0.020272 -0.033681 -0.020256 -0.024774  0.017723 -0.014750
19  -0.008777 -0.023458 -0.023053 -0.020479 -0.009713 -0.031666 -0.008538
20  -0.025919 -0.035607 -0.000533 -0.015637 -0.017454 -0.015134 -0.016289
21   0.015279  0.022403  0.015710  0.024040  0.021039 -0.006175  0.027574
22   0.018578  0.023231 -0.007518  0.026577  0.015275  0.026908  0.009565
```

23	-0.014133	-0.014571	0.016233	0.003932	0.000071	-0.011169	0.024128
24	0.036607	0.042759	0.026541	0.029306	0.014788	0.015846	0.039282
25	0.011353	0.021468	0.001484	0.004766	0.003651	-0.013411	-0.015462
26	-0.040542	-0.043907	-0.050369	-0.035170	-0.022182	-0.002902	-0.021440
27	-0.022106	-0.033893	0.007923	0.005434	0.005019	-0.030745	-0.008799
28	-0.014888	-0.020825	0.001738	-0.027421	-0.007610	0.000000	-0.008482
29	0.007027	0.009709	-0.010048	0.001322	-0.003003	0.009563	0.028551
..	...	...	...	...	...	...	...
506	0.003761	0.009064	0.008967	0.018160	0.006084	0.000202	0.017036
507	0.009547	0.016298	-0.001712	0.000895	-0.004439	0.007294	-0.012813
508	-0.010199	-0.004700	0.007357	0.000083	-0.003625	-0.008604	0.003092
509	-0.015563	-0.015368	0.001375	0.000333	-0.002736	0.000364	-0.004677
510	-0.006049	0.004580	0.000000	0.009196	0.011742	0.001534	0.004395
511	0.000521	-0.003209	-0.010167	-0.008532	-0.013247	0.003617	-0.012229
512	-0.017835	-0.031652	-0.001296	-0.008292	-0.018372	-0.011412	-0.007105
513	0.009744	-0.000511	0.002411	0.005416	0.004828	-0.015720	-0.006186
514	-0.011067	-0.007426	0.005819	0.000757	0.008040	0.006847	0.004244
515	-0.000749	0.004639	0.000263	-0.001240	-0.004418	0.011467	-0.010396
516	0.012326	0.005404	0.004212	0.009635	0.008665	-0.005992	0.000000
517	-0.014082	-0.014976	0.002242	0.003953	-0.000687	0.007352	-0.009623
518	-0.028498	-0.038599	-0.018014	-0.007403	-0.014131	-0.011356	-0.020082
519	0.001056	-0.008883	0.007633	-0.003571	-0.003150	-0.011887	-0.001846
520	0.024116	0.035257	0.016556	0.014976	0.016057	0.003567	0.018926
521	0.007447	0.014239	-0.002726	-0.000084	0.007075	0.017641	-0.017230
522	-0.024478	-0.027850	0.002351	0.001392	-0.002804	-0.002489	0.001154
523	0.024507	0.013959	0.002880	0.003127	0.002354	0.010695	-0.022662
524	-0.006196	0.008553	0.006221	0.009298	0.008898	0.004591	0.001424
525	0.005356	0.006886	0.004176	0.005425	0.006638	0.004140	0.006238
526	0.004823	-0.003255	-0.002790	-0.000320	-0.006423	-0.001708	-0.023895
527	-0.017664	-0.024921	0.000749	0.002644	-0.005351	-0.001148	0.005590
528	0.004782	0.006418	0.005492	0.004204	0.007101	0.011241	0.018077
529	-0.002498	0.000405	0.002382	0.003444	-0.000462	0.000000	0.012123
530	0.003606	0.000893	-0.003240	0.000461	-0.003803	0.001968	-0.003266
531	0.008599	0.013400	0.006238	0.001925	0.007952	0.005717	0.018371
532	0.009310	0.015977	0.003071	-0.001186	0.000345	0.002620	0.001686
533	0.000191	-0.001653	0.001923	0.002872	-0.000723	0.000568	0.005628
534	-0.013069	-0.013706	-0.020742	-0.014239	-0.011275	0.001358	-0.011942
535	-0.007246	-0.019442	0.000000	-0.000473	-0.002997	-0.017920	-0.012252

	EU	EM
0	0.012698	0.028524
1	0.011341	0.008773
2	-0.017073	-0.020015
3	-0.005561	-0.019424
4	-0.010989	-0.007802
5	-0.012451	-0.022630
6	-0.012220	-0.004827
7	-0.045220	-0.008677

8	-0.012070	-0.023429
9	0.008561	0.010917
10	-0.012045	-0.004029
11	-0.015088	-0.024107
12	-0.003339	-0.005092
13	-0.006552	-0.003227
14	-0.003620	-0.008077
15	0.032800	0.010320
16	-0.002642	0.006344
17	0.029974	0.022104
18	-0.023109	0.000409
19	-0.007201	0.002243
20	-0.019739	-0.019091
21	0.017862	0.012719
22	0.018770	0.015166
23	-0.004139	0.002073
24	0.019127	0.032338
25	0.005627	0.007895
26	-0.024388	-0.002139
27	0.001097	-0.007926
28	-0.014092	-0.014773
29	0.003032	0.017764
..	...	...
506	0.014397	0.011872
507	-0.000367	-0.002109
508	-0.000761	0.000238
509	-0.002147	-0.004537
510	0.010995	0.002406
511	-0.010645	0.001086
512	-0.010088	-0.010442
513	0.008396	-0.006854
514	0.004158	-0.001145
515	-0.004044	-0.000885
516	0.007097	0.005019
517	0.001919	-0.001149
518	-0.010478	-0.009912
519	-0.002650	-0.005789
520	0.015089	0.006224
521	0.001618	0.003631
522	-0.002883	0.000476
523	0.002761	-0.003185
524	0.008217	-0.003346
525	0.003980	-0.004499
526	-0.003024	-0.014249
527	-0.003742	-0.014760
528	0.004727	0.003931
529	0.000169	0.013448
530	-0.000550	-0.001430

```
531  0.006975  0.003039
532 -0.000581  0.001039
533  0.000572  0.006938
534 -0.012615 -0.000958
535 -0.005465 -0.014297

[536 rows x 9 columns]
```

3 Normalizing data (min max scaling)

In [3]:
```python
# normalizing
min_max_scaler = preprocessing.MinMaxScaler()
col_names = {i-1:col for i,col in enumerate(df)}
print(col_names)
df_scaled = pd.DataFrame(min_max_scaler.fit_transform(df_removed_date.values))
df_scaled = df_scaled.rename(index =str ,columns =col_names)
df_scaled
```

```
{0: u'ISE', 1: u'ISE.1', 2: u'SP', 3: u'DAX', 4: u'FTSE', 5: u'NIKKEI', 6: u'BOVESPA', 7: u'EU
```

Out[3]:

	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	BOVESPA \
0	0.746889	0.664154	0.404333	0.489969	0.558409	0.451728	0.722875
1	0.668147	0.628741	0.505990	0.546240	0.643736	0.489000	0.618569
2	0.254242	0.314902	0.194024	0.310007	0.248066	0.606576	0.152590
3	0.000000	0.000000	0.470147	0.364884	0.516936	0.093003	0.698163
4	0.549467	0.509203	0.266894	0.291678	0.400483	0.411670	0.374747
5	0.251732	0.228529	0.256379	0.348714	0.473569	0.012617	0.000000
6	0.592052	0.455624	0.456815	0.311440	0.462955	0.451728	0.488108
7	0.160419	0.265266	0.164966	0.044461	0.036816	0.477806	0.115163
8	0.479339	0.363922	0.453323	0.294573	0.385028	0.000000	0.715423
9	0.642311	0.631251	0.503922	0.531282	0.581186	0.679646	0.499113
10	0.301278	0.217804	0.000000	0.366469	0.432426	0.480731	0.345955
11	0.369773	0.297052	0.442491	0.310002	0.481699	0.242092	0.110082
12	0.480885	0.465344	0.789654	0.515290	0.447857	0.267614	0.742782
13	0.445205	0.484303	0.317900	0.381822	0.503312	0.620231	0.313392
14	0.517455	0.403093	0.486227	0.383631	0.522072	0.104221	0.510966
15	0.831348	0.790044	0.487651	0.782860	0.881760	0.378467	0.541373
16	0.423705	0.516167	0.531103	0.463091	0.488319	0.882866	0.499581
17	0.737841	0.650637	0.711653	0.867287	0.747240	0.501819	0.786921
18	0.317779	0.347714	0.167832	0.288235	0.285741	0.610429	0.332361
19	0.407376	0.330521	0.254502	0.286232	0.428989	0.168178	0.385165
20	0.276677	0.264969	0.438146	0.329744	0.355356	0.316213	0.319278
21	0.590788	0.577969	0.570601	0.686287	0.721472	0.396437	0.692133
22	0.615935	0.582436	0.381187	0.709081	0.666650	0.692676	0.539050
23	0.366539	0.378471	0.574868	0.505591	0.522043	0.351713	0.662841
24	0.753395	0.687802	0.658923	0.733607	0.662017	0.593616	0.791662
25	0.560853	0.572923	0.454593	0.513088	0.556097	0.331643	0.326312

26	0.165188	0.220189	0.031749	0.154212	0.310388	0.425745	0.275498
27	0.305753	0.274217	0.507105	0.519090	0.569108	0.176426	0.382947
28	0.360779	0.344729	0.456661	0.223852	0.448989	0.451728	0.385641
29	0.527866	0.509479	0.360552	0.482137	0.492807	0.537359	0.700438
..	...	...	...	...	...	...	...
506	0.502964	0.505998	0.515617	0.633447	0.579234	0.453534	0.602557
507	0.547080	0.545027	0.428528	0.478302	0.479147	0.517041	0.348825
508	0.396532	0.431732	0.502489	0.471008	0.486885	0.374687	0.484027
509	0.355632	0.374174	0.453708	0.473255	0.495344	0.454985	0.417989
510	0.428170	0.481801	0.442491	0.552896	0.633052	0.465461	0.495102
511	0.478263	0.439778	0.359581	0.393587	0.395373	0.484119	0.353792
512	0.338313	0.286313	0.431925	0.395742	0.346630	0.349543	0.397348
513	0.548584	0.454336	0.462150	0.518932	0.567291	0.310966	0.405158
514	0.389914	0.417023	0.489946	0.477063	0.597843	0.513038	0.493817
515	0.468583	0.482120	0.444639	0.459114	0.479351	0.554408	0.369377
516	0.568270	0.486252	0.476839	0.556838	0.603784	0.398069	0.457744
517	0.366926	0.376286	0.460772	0.505781	0.514833	0.517564	0.375946
518	0.257014	0.248828	0.295594	0.403730	0.386965	0.350044	0.287034
519	0.482345	0.409163	0.504739	0.438167	0.491403	0.345291	0.442054
520	0.658158	0.647323	0.577499	0.604836	0.674090	0.483665	0.618624
521	0.531069	0.533918	0.420259	0.469508	0.588659	0.609692	0.311280
522	0.287664	0.306824	0.461667	0.482772	0.494700	0.429437	0.467553
523	0.661138	0.532410	0.465978	0.498359	0.543756	0.547494	0.265109
524	0.427052	0.503240	0.493220	0.553813	0.605995	0.492841	0.469848
525	0.515125	0.494245	0.476546	0.519007	0.584504	0.488798	0.510770
526	0.511064	0.439530	0.419742	0.467387	0.460276	0.436435	0.254622
527	0.339614	0.322631	0.448601	0.494016	0.470473	0.441450	0.505264
528	0.510754	0.491722	0.487279	0.508032	0.588907	0.552384	0.611408
529	0.455247	0.459275	0.461917	0.501207	0.516975	0.451728	0.560793
530	0.501788	0.461909	0.416068	0.474400	0.485201	0.469351	0.429985
531	0.539854	0.529392	0.493360	0.487558	0.597004	0.502925	0.613904
532	0.545277	0.543299	0.467534	0.459599	0.524650	0.475185	0.472073
533	0.475748	0.448175	0.458172	0.496068	0.514491	0.456817	0.505584
534	0.374650	0.383140	0.273345	0.342305	0.414130	0.463884	0.356235
535	0.419044	0.352192	0.442491	0.466011	0.492862	0.291268	0.353601

	EU	EM
0	0.530944	0.776771
1	0.519229	0.548080
2	0.273987	0.214765
3	0.373348	0.221615
4	0.326501	0.356172
5	0.313877	0.184496
6	0.315871	0.390618
7	0.031043	0.346048
8	0.317164	0.175245
9	0.495236	0.572906
10	0.317382	0.399859

11	0.291118	0.167390
12	0.392530	0.387549
13	0.364792	0.409142
14	0.390100	0.352987
15	0.704444	0.565992
16	0.398544	0.519966
17	0.680058	0.702435
18	0.221892	0.451240
19	0.359190	0.472473
20	0.250972	0.225464
21	0.575518	0.593768
22	0.583352	0.622099
23	0.385620	0.470504
24	0.586431	0.820926
25	0.469914	0.537923
26	0.210851	0.421741
27	0.430818	0.354739
28	0.299714	0.275466
29	0.447515	0.652189
..	...	...
506	0.545606	0.583968
507	0.418174	0.422086
508	0.414775	0.449259
509	0.402816	0.393981
510	0.516243	0.474364
511	0.329470	0.459079
512	0.334275	0.325608
513	0.493813	0.367157
514	0.457234	0.433254
515	0.386440	0.436256
516	0.482600	0.504614
517	0.437912	0.433206
518	0.330910	0.331746
519	0.398477	0.379485
520	0.551578	0.518570
521	0.435309	0.488552
522	0.396458	0.452020
523	0.445177	0.409632
524	0.492265	0.407771
525	0.455700	0.394418
526	0.395244	0.281535
527	0.389044	0.275618
528	0.462147	0.492021
529	0.422800	0.602213
530	0.416599	0.429946
531	0.481545	0.481694
532	0.416335	0.458533
533	0.426279	0.526836

```
534  0.312461  0.435419
535  0.374177  0.280975

[536 rows x 9 columns]
```

4 Quantization (Binning) #1

In [4]:
```
# Quantization of the dataframe
# Binning 1 Supervised
l= []
for col in pd.DataFrame(df_scaled):
    l.append(pd.cut(df_scaled[col],4,labels=False))
df_binning1 = pd.DataFrame(np.matrix(l).T)
df_binning1 =df_binning1.rename(index =str ,columns =col_names)
df_binning1
```

Out[4]:

	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	BOVESPA	EU	EM
0	2	2	1	1	2	1	2	2	3
1	2	2	2	2	2	1	2	2	2
2	1	1	0	1	0	2	0	1	0
3	0	0	1	1	2	0	2	1	0
4	2	2	1	1	1	1	1	1	1
5	1	0	1	1	1	0	0	1	0
6	2	1	1	1	1	1	1	1	1
7	0	1	0	0	0	1	0	0	1
8	1	1	1	1	1	0	2	1	0
9	2	2	2	2	2	2	1	1	2
10	1	0	0	1	1	1	1	1	1
11	1	1	1	1	1	0	0	1	0
12	1	1	3	2	1	1	2	1	1
13	1	1	1	1	2	2	1	1	1
14	2	1	1	1	2	0	2	1	1
15	3	3	1	3	3	1	2	2	2
16	1	2	2	1	1	3	1	1	2
17	2	2	2	3	2	2	3	2	2
18	1	1	0	1	1	2	1	0	1
19	1	1	1	1	1	0	1	1	1
20	1	1	1	1	1	1	1	1	0
21	2	2	2	2	2	1	2	2	2
22	2	2	1	2	2	2	2	2	2
23	1	1	2	2	2	1	2	1	1
24	3	2	2	2	2	2	3	2	3
25	2	2	1	2	2	1	1	1	2
26	0	0	0	0	1	1	1	0	1
27	1	1	2	2	2	0	1	1	1
28	1	1	1	0	1	1	1	1	1
29	2	2	1	1	1	2	2	1	2

```
..   ...    ...  ..  ...   ...     ...      ...  ..  ..
506    2      2   2    2     2       1        2   2   2
507    2      2   1    1     1       2        1   1   1
508    1      1   2    1     1       1        1   1   1
509    1      1   1    1     1       1        1   1   1
510    1      1   1    2     2       1        1   2   1
511    1      1   1    1     1       1        1   1   1
512    1      1   1    1     1       1        1   1   1
513    2      1   1    2     2       1        1   1   1
514    1      1   1    1     2       2        1   1   1
515    1      1   1    1     1       2        1   1   1
516    2      1   1    2     2       1        1   1   2
517    1      1   1    2     2       2        1   1   1
518    1      0   1    1     1       1        1   1   1
519    1      1   2    1     1       1        1   1   1
520    2      2   2    2     2       1        2   2   2
521    2      2   1    1     2       2        1   1   1
522    1      1   1    1     1       1        1   1   1
523    2      2   1    1     2       2        1   1   1
524    1      2   1    2     2       1        1   1   1
525    2      1   1    2     2       1        2   1   1
526    2      1   1    1     1       1        1   1   1
527    1      1   1    1     1       1        2   1   1
528    2      1   1    2     2       2        2   1   1
529    1      1   1    2     2       1        2   1   2
530    2      1   1    1     1       1        1   1   1
531    2      2   1    1     2       2        2   1   1
532    2      2   1    1     2       1        1   1   1
533    1      1   1    1     2       1        2   1   2
534    1      1   1    1     1       1        1   1   1
535    1      1   1    1     1       1        1   1   1

[536 rows x 9 columns]
```

5 Quantization (Binning) #2

In [5]:
```
# Quantization of the dataframe
# Binning 2 UnSupervised binning equal bin width (Quantiles)

l= []
for col in pd.DataFrame(df_scaled):
    l.append(pd.qcut(df_scaled[col],[0, .25, .5, .75, 1.],labels=False))
df_binning2 = pd.DataFrame(np.matrix(l).T)
df_binning2 =df_binning2.rename(index =str ,columns =col_names)
df_binning2
```

Out[5]:
```
     ISE  ISE.1  SP  DAX  FTSE  NIKKEI  BOVESPA  EU  EM
0      3      3   0    2     2       1        3   3   3
```

1	3	3	3	3	3	2	3	3	3
2	0	0	0	0	0	3	0	0	0
3	0	0	2	0	1	0	3	1	0
4	2	2	0	0	0	1	0	0	0
5	0	0	0	0	1	0	0	0	0
6	3	1	2	0	0	1	2	0	1
7	0	0	0	0	0	2	0	0	0
8	1	0	2	0	0	0	3	0	0
9	3	3	3	2	2	3	2	3	3
10	0	0	0	0	0	2	0	0	1
11	0	0	1	0	1	0	0	0	0
12	1	1	3	2	0	0	3	1	0
13	1	2	0	0	1	3	0	0	1
14	2	0	2	0	1	0	2	1	0
15	3	3	2	3	3	0	3	3	3
16	1	2	3	1	1	3	2	1	2
17	3	3	3	3	3	2	3	3	3
18	0	0	0	0	0	3	0	0	1
19	0	0	0	0	0	0	0	0	2
20	0	0	1	0	0	0	0	0	0
21	3	3	3	3	3	1	3	3	3
22	3	3	0	3	3	3	3	3	3
23	0	0	3	2	1	0	3	1	2
24	3	3	3	3	3	3	3	3	3
25	3	3	2	2	2	0	0	2	3
26	0	0	0	0	0	1	0	0	1
27	0	0	3	2	2	0	0	2	0
28	0	0	2	0	0	1	0	0	0
29	2	2	0	2	1	3	3	2	3
..	...	...	..	...	...	...	...	..	..
506	2	2	3	3	2	2	3	3	3
507	2	3	1	2	1	2	0	1	1
508	0	1	3	1	1	0	2	1	1
509	0	0	2	1	1	2	1	1	1
510	1	2	1	3	3	2	2	3	2
511	1	1	0	0	0	2	0	0	2
512	0	0	1	0	0	0	1	0	0
513	2	1	2	2	2	0	1	3	0
514	0	1	2	1	3	2	2	2	1
515	1	2	1	1	1	3	0	1	1
516	3	2	2	3	3	1	1	2	2
517	0	0	2	2	1	2	0	2	1
518	0	0	0	0	0	0	0	0	0
519	1	1	3	1	1	0	1	1	0
520	3	3	3	3	3	2	3	3	2
521	2	3	1	1	2	3	0	2	2
522	0	0	2	2	1	1	2	1	1
523	3	3	2	2	2	3	0	2	1

```
524    1      2   2    3     3       2        2   3   1
525    2      2   2    2     2       2        2   2   1
526    2      1   1    1     0       1        0   1   0
527    0      0   1    2     1       1        2   1   0
528    2      2   2    2     2       3        3   2   2
529    1      1   2    2     1       1        3   1   3
530    2      1   1    1     1       2        1   1   1
531    2      2   2    2     3       2        3   2   2
532    2      3   2    1     1       2        2   1   1
533    1      1   2    2     1       2        2   2   3
534    0      0   0    0     0       2        0   0   1
535    0      0   1    1     1       0        0   1   0

[536 rows x 9 columns]
```

6 Generating missing values randomly (5% to 10% per column)

```
In [17]: # Generating random numbers for adding missing values
         import random
         randnum_ls = np.array(random.sample(range(1,10+1),NUM_COLS))
         randnum_ls =randnum_ls/100.0
         num_missing_per_col = np.round(randnum_ls*NUM_ROWS).astype(np.int) # number of missin

         #Randomly selects indices to add to missing values
         def ls_index_to_change(NUM_ROWS,num_missing):
             indices_to_replace =np.array(random.sample(range(NUM_ROWS),num_missing))
             return indices_to_replace


         def add_NaN(df,num_missing_per_col):
             df_new = df.copy(deep=True)
             ls = [] # list of index list
             for num in num_missing_per_col:
                 indices = ls_index_to_change(df_new.shape[0],num) # getting the indices to ad
                 ls.append(indices)
             for i, col in enumerate(df):
                 df_new[col][ls[i]] = np.nan
             return df_new
         df_binning2_missing = add_NaN(df_binning2,num_missing_per_col)
         df_binning2_copy=df_binning2.copy(deep=True)
```

```
/Users/francisco/miniconda3/envs/python2/lib/python2.7/site-packages/ipykernel_launcher.py:20:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
```

```
In [13]: df_binning2_missing
```

Out[13]:

	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	BOVESPA	EU	EM
0	3.0	3.0	0.0	2.0	2.0	1.0	3.0	3.0	3.0
1	3.0	3.0	3.0	3.0	3.0	2.0	3.0	3.0	3.0
2	0.0	NaN	0.0	NaN	0.0	3.0	0.0	NaN	NaN
3	0.0	NaN	2.0	0.0	1.0	0.0	3.0	1.0	0.0
4	2.0	2.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
5	0.0	NaN	0.0	0.0	1.0	0.0	0.0	0.0	0.0
6	3.0	1.0	2.0	0.0	NaN	1.0	2.0	0.0	1.0
7	0.0	0.0	0.0	NaN	0.0	2.0	0.0	0.0	0.0
8	1.0	0.0	2.0	0.0	0.0	0.0	3.0	0.0	0.0
9	3.0	3.0	3.0	2.0	2.0	3.0	2.0	3.0	3.0
10	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	1.0
11	0.0	0.0	1.0	0.0	NaN	0.0	0.0	0.0	0.0
12	NaN	1.0	3.0	2.0	0.0	0.0	3.0	1.0	0.0
13	1.0	2.0	0.0	0.0	1.0	3.0	0.0	0.0	1.0
14	2.0	0.0	2.0	0.0	NaN	0.0	2.0	1.0	0.0
15	3.0	3.0	2.0	NaN	3.0	0.0	3.0	3.0	3.0
16	1.0	2.0	3.0	1.0	1.0	3.0	2.0	1.0	2.0
17	3.0	3.0	3.0	3.0	3.0	2.0	3.0	3.0	3.0
18	0.0	0.0	0.0	0.0	0.0	NaN	0.0	0.0	NaN
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0
20	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
21	3.0	3.0	3.0	3.0	NaN	1.0	3.0	3.0	NaN
22	3.0	3.0	0.0	3.0	3.0	3.0	3.0	3.0	NaN
23	0.0	0.0	3.0	2.0	1.0	0.0	3.0	1.0	2.0
24	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
25	3.0	3.0	2.0	2.0	2.0	0.0	0.0	2.0	3.0
26	0.0	0.0	NaN	0.0	0.0	1.0	0.0	0.0	1.0
27	0.0	0.0	3.0	2.0	2.0	0.0	0.0	2.0	0.0
28	0.0	0.0	2.0	0.0	0.0	1.0	0.0	0.0	0.0
29	2.0	2.0	0.0	NaN	1.0	3.0	3.0	2.0	3.0
..	...	...	...	...	...	...	...	...	...
506	2.0	2.0	3.0	3.0	2.0	2.0	3.0	3.0	3.0
507	2.0	3.0	1.0	NaN	1.0	2.0	0.0	1.0	1.0
508	0.0	1.0	3.0	NaN	1.0	0.0	2.0	1.0	1.0
509	0.0	0.0	2.0	1.0	1.0	2.0	1.0	1.0	1.0
510	1.0	2.0	1.0	3.0	3.0	2.0	2.0	3.0	2.0
511	1.0	1.0	0.0	0.0	0.0	2.0	0.0	0.0	2.0
512	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0
513	2.0	1.0	2.0	2.0	2.0	0.0	1.0	3.0	0.0
514	0.0	1.0	2.0	1.0	3.0	2.0	2.0	2.0	1.0
515	1.0	2.0	1.0	1.0	1.0	3.0	0.0	NaN	1.0
516	3.0	2.0	2.0	3.0	3.0	1.0	1.0	2.0	2.0
517	0.0	0.0	2.0	2.0	NaN	2.0	0.0	2.0	1.0
518	0.0	0.0	0.0	0.0	0.0	0.0	NaN	0.0	0.0
519	1.0	1.0	3.0	1.0	1.0	0.0	1.0	1.0	0.0
520	3.0	3.0	3.0	3.0	3.0	2.0	3.0	3.0	2.0
521	2.0	3.0	1.0	1.0	2.0	3.0	0.0	2.0	2.0

522	0.0	0.0	2.0	2.0	1.0	1.0	2.0	1.0	1.0
523	3.0	3.0	NaN	2.0	2.0	3.0	0.0	2.0	1.0
524	1.0	2.0	2.0	3.0	NaN	2.0	2.0	3.0	1.0
525	2.0	2.0	2.0	2.0	2.0	NaN	2.0	2.0	1.0
526	2.0	1.0	1.0	1.0	0.0	1.0	0.0	1.0	0.0
527	0.0	0.0	1.0	2.0	1.0	1.0	2.0	1.0	0.0
528	2.0	2.0	2.0	2.0	2.0	3.0	3.0	2.0	2.0
529	1.0	1.0	2.0	2.0	1.0	NaN	3.0	1.0	3.0
530	2.0	1.0	1.0	1.0	1.0	2.0	1.0	1.0	1.0
531	2.0	2.0	2.0	2.0	3.0	2.0	3.0	NaN	2.0
532	2.0	3.0	2.0	1.0	1.0	2.0	2.0	1.0	1.0
533	1.0	1.0	2.0	2.0	1.0	2.0	2.0	2.0	3.0
534	0.0	0.0	NaN	0.0	0.0	2.0	0.0	0.0	1.0
535	0.0	0.0	1.0	1.0	1.0	0.0	0.0	1.0	0.0

[536 rows x 9 columns]

In [7]: df_binning2_copy

Out[7]:

	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	BOVESPA	EU	EM
0	3	3	0	2	2	1	3	3	3
1	3	3	3	3	3	2	3	3	3
2	0	0	0	0	0	3	0	0	0
3	0	0	2	0	1	0	3	1	0
4	2	2	0	0	0	1	0	0	0
5	0	0	0	0	1	0	0	0	0
6	3	1	2	0	0	1	2	0	1
7	0	0	0	0	0	2	0	0	0
8	1	0	2	0	0	0	3	0	0
9	3	3	3	2	2	3	2	3	3
10	0	0	0	0	0	2	0	0	1
11	0	0	1	0	1	0	0	0	0
12	1	1	3	2	0	0	3	1	0
13	1	2	0	0	1	3	0	0	1
14	2	0	2	0	1	0	2	1	0
15	3	3	2	3	3	0	3	3	3
16	1	2	3	1	1	3	2	1	2
17	3	3	3	3	3	2	3	3	3
18	0	0	0	0	0	3	0	0	1
19	0	0	0	0	0	0	0	0	2
20	0	0	1	0	0	0	0	0	0
21	3	3	3	3	3	1	3	3	3
22	3	3	0	3	3	3	3	3	3
23	0	0	3	2	1	0	3	1	2
24	3	3	3	3	3	3	3	3	3
25	3	3	2	2	2	0	0	2	3
26	0	0	0	0	0	1	0	0	1
27	0	0	3	2	2	0	0	2	0

```
28     0    0  2   0   0     1     0  0  0
29     2    2  0   2   1     3     3  2  3
..   ...  ... ..  ... ...   ...   ... .. ..
506    2    2  3   3   2     2     3  3  3
507    2    3  1   2   1     2     0  1  1
508    0    1  3   1   1     0     2  1  1
509    0    0  2   1   1     2     1  1  1
510    1    2  1   3   3     2     2  3  2
511    1    1  0   0   0     2     0  0  2
512    0    0  1   0   0     0     1  0  0
513    2    1  2   2   2     0     1  3  0
514    0    1  2   1   3     2     2  2  1
515    1    2  1   1   1     3     0  1  1
516    3    2  2   3   3     1     1  2  2
517    0    0  2   2   1     2     0  2  1
518    0    0  0   0   0     0     0  0  0
519    1    1  3   1   1     0     1  1  0
520    3    3  3   3   3     2     3  3  2
521    2    3  1   1   2     3     0  2  2
522    0    0  2   2   1     1     2  1  1
523    3    3  2   2   2     3     0  2  1
524    1    2  2   3   3     2     2  3  1
525    2    2  2   2   2     2     2  2  1
526    2    1  1   1   0     1     0  1  0
527    0    0  1   2   1     1     2  1  0
528    2    2  2   2   2     3     3  2  2
529    1    1  2   2   1     1     3  1  3
530    2    1  1   1   1     2     1  1  1
531    2    2  2   2   3     2     3  2  2
532    2    3  2   1   1     2     2  1  1
533    1    1  2   2   1     2     2  2  3
534    0    0  0   0   0     2     0  0  1
535    0    0  1   1   1     0     0  1  0

[536 rows x 9 columns]
```

7 Imputation code (mean and median)

In [8]:
```
#Imputation

def impute_mean(df):
    for col in df:
        df_prime=df[col].dropna() # drop columns
        mean = np.mean(df_prime)
        print(mean)
        df_prime = df[col].replace(to_replace=np.nan,value=mean)
        df[col]=df_prime
```

```
        return df

    #imputation using median
    def impute_median(df):
        for col in df:
             # if not the class column
            df_prime=df[col].dropna() # drop columns
            median = np.median(df_prime)
            print(median)
            df_prime = df[col].replace(to_replace=np.nan,value=median)
        return df
```

8 Imputing mean

In [14]:
```
#mean = np.mean(df_binning2.dropna()['ISE'])
#df_binning2['ISE'].replace(np.nan,mean)

df_imputed_mean = impute_mean(df_binning2_missing)
df_imputed_mean
```

```
1.5104761904761905
1.5009708737864078
1.539553752535497
1.504149377593361
1.4897540983606556
1.4833005893909628
1.5009416195856873
1.498076923076923
1.5060240963855422
```

Out[14]:

	ISE	ISE.1	SP	DAX	FTSE	NIKKEI	BOVESPA \
0	3.000000	3.000000	0.000000	2.000000	2.000000	1.000000	3.000000
1	3.000000	3.000000	3.000000	3.000000	3.000000	2.000000	3.000000
2	0.000000	1.500971	0.000000	1.504149	0.000000	3.000000	0.000000
3	0.000000	1.500971	2.000000	0.000000	1.000000	0.000000	3.000000
4	2.000000	2.000000	0.000000	0.000000	0.000000	1.000000	0.000000
5	0.000000	1.500971	0.000000	0.000000	1.000000	0.000000	0.000000
6	3.000000	1.000000	2.000000	0.000000	1.489754	1.000000	2.000000
7	0.000000	0.000000	0.000000	1.504149	0.000000	2.000000	0.000000
8	1.000000	0.000000	2.000000	0.000000	0.000000	0.000000	3.000000
9	3.000000	3.000000	3.000000	2.000000	2.000000	3.000000	2.000000
10	0.000000	0.000000	0.000000	0.000000	0.000000	2.000000	0.000000
11	0.000000	0.000000	1.000000	0.000000	1.489754	0.000000	0.000000
12	1.510476	1.000000	3.000000	2.000000	0.000000	0.000000	3.000000
13	1.000000	2.000000	0.000000	0.000000	1.000000	3.000000	0.000000
14	2.000000	0.000000	2.000000	0.000000	1.489754	0.000000	2.000000
15	3.000000	3.000000	2.000000	1.504149	3.000000	0.000000	3.000000

16	1.000000	2.000000	3.000000	1.000000	1.000000	3.000000	2.000000
17	3.000000	3.000000	3.000000	3.000000	3.000000	2.000000	3.000000
18	0.000000	0.000000	0.000000	0.000000	0.000000	1.483301	0.000000
19	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
20	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	0.000000
21	3.000000	3.000000	3.000000	3.000000	1.489754	1.000000	3.000000
22	3.000000	3.000000	0.000000	3.000000	3.000000	3.000000	3.000000
23	0.000000	0.000000	3.000000	2.000000	1.000000	0.000000	3.000000
24	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000	3.000000
25	3.000000	3.000000	2.000000	2.000000	2.000000	0.000000	0.000000
26	0.000000	0.000000	1.539554	0.000000	0.000000	1.000000	0.000000
27	0.000000	0.000000	3.000000	2.000000	2.000000	0.000000	0.000000
28	0.000000	0.000000	2.000000	0.000000	0.000000	1.000000	0.000000
29	2.000000	2.000000	0.000000	1.504149	1.000000	3.000000	3.000000
..	...	...	...	...	...	...	...
506	2.000000	2.000000	3.000000	3.000000	2.000000	2.000000	3.000000
507	2.000000	3.000000	1.000000	1.504149	1.000000	2.000000	0.000000
508	0.000000	1.000000	3.000000	1.504149	1.000000	0.000000	2.000000
509	0.000000	0.000000	2.000000	1.000000	1.000000	2.000000	1.000000
510	1.000000	2.000000	1.000000	3.000000	3.000000	2.000000	2.000000
511	1.000000	1.000000	0.000000	0.000000	0.000000	2.000000	0.000000
512	0.000000	0.000000	1.000000	0.000000	0.000000	0.000000	1.000000
513	2.000000	1.000000	2.000000	2.000000	2.000000	0.000000	1.000000
514	0.000000	1.000000	2.000000	1.000000	3.000000	2.000000	2.000000
515	1.000000	2.000000	1.000000	1.000000	1.000000	3.000000	0.000000
516	3.000000	2.000000	2.000000	3.000000	3.000000	1.000000	1.000000
517	0.000000	0.000000	2.000000	2.000000	1.489754	2.000000	0.000000
518	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.500942
519	1.000000	1.000000	3.000000	1.000000	1.000000	0.000000	1.000000
520	3.000000	3.000000	3.000000	3.000000	3.000000	2.000000	3.000000
521	2.000000	3.000000	1.000000	1.000000	2.000000	3.000000	0.000000
522	0.000000	0.000000	2.000000	2.000000	1.000000	1.000000	2.000000
523	3.000000	3.000000	1.539554	2.000000	2.000000	3.000000	0.000000
524	1.000000	2.000000	2.000000	3.000000	1.489754	2.000000	2.000000
525	2.000000	2.000000	2.000000	2.000000	2.000000	1.483301	2.000000
526	2.000000	1.000000	1.000000	1.000000	0.000000	1.000000	0.000000
527	0.000000	0.000000	1.000000	2.000000	1.000000	1.000000	2.000000
528	2.000000	2.000000	2.000000	2.000000	2.000000	3.000000	3.000000
529	1.000000	1.000000	2.000000	2.000000	1.000000	1.483301	3.000000
530	2.000000	1.000000	1.000000	1.000000	1.000000	2.000000	1.000000
531	2.000000	2.000000	2.000000	2.000000	3.000000	2.000000	3.000000
532	2.000000	3.000000	2.000000	1.000000	1.000000	2.000000	2.000000
533	1.000000	1.000000	2.000000	2.000000	1.000000	2.000000	2.000000
534	0.000000	0.000000	1.539554	0.000000	0.000000	2.000000	0.000000
535	0.000000	0.000000	1.000000	1.000000	1.000000	0.000000	0.000000

	EU	EM
0	3.000000	3.000000

1	3.000000	3.000000
2	1.498077	1.506024
3	1.000000	0.000000
4	0.000000	0.000000
5	0.000000	0.000000
6	0.000000	1.000000
7	0.000000	0.000000
8	0.000000	0.000000
9	3.000000	3.000000
10	0.000000	1.000000
11	0.000000	0.000000
12	1.000000	0.000000
13	0.000000	1.000000
14	1.000000	0.000000
15	3.000000	3.000000
16	1.000000	2.000000
17	3.000000	3.000000
18	0.000000	1.506024
19	0.000000	2.000000
20	0.000000	0.000000
21	3.000000	1.506024
22	3.000000	1.506024
23	1.000000	2.000000
24	3.000000	3.000000
25	2.000000	3.000000
26	0.000000	1.000000
27	2.000000	0.000000
28	0.000000	0.000000
29	2.000000	3.000000
..	...	...
506	3.000000	3.000000
507	1.000000	1.000000
508	1.000000	1.000000
509	1.000000	1.000000
510	3.000000	2.000000
511	0.000000	2.000000
512	0.000000	0.000000
513	3.000000	0.000000
514	2.000000	1.000000
515	1.498077	1.000000
516	2.000000	2.000000
517	2.000000	1.000000
518	0.000000	0.000000
519	1.000000	0.000000
520	3.000000	2.000000
521	2.000000	2.000000
522	1.000000	1.000000
523	2.000000	1.000000

```
524  3.000000  1.000000
525  2.000000  1.000000
526  1.000000  0.000000
527  1.000000  0.000000
528  2.000000  2.000000
529  1.000000  3.000000
530  1.000000  1.000000
531  1.498077  2.000000
532  1.000000  1.000000
533  2.000000  3.000000
534  0.000000  1.000000
535  1.000000  0.000000

[536 rows x 9 columns]
```

9 Imputing median

In [ ]:
```
df_imputed_median = impute_median(df_binning2_missing)
df_imputed_median
```

10 Imputing KNN

In [18]:
```
def knn_impute(df,k):
    df_filled = pd.DataFrame(KNN(k).complete(df))
    return df_filled

df_knn = knn_impute(df_binning2_missing,3)
df_knn.rename(index=str,columns={0:"ISE",1:"ISE.1", 2:"SP", 3:"DAX", 4:"FTSE", 5:"NIKK
```

```
Imputing row 1/536 with 0 missing, elapsed time: 0.060
Imputing row 101/536 with 0 missing, elapsed time: 0.062
Imputing row 201/536 with 0 missing, elapsed time: 0.063
Imputing row 301/536 with 0 missing, elapsed time: 0.064
Imputing row 401/536 with 0 missing, elapsed time: 0.067
Imputing row 501/536 with 0 missing, elapsed time: 0.069
```

Out[18]:
```
        ISE  ISE.1        SP       DAX      FTSE    NIKKEI   BOVESPA   EU  \
0  3.000000    3.0  0.000000  2.000000  2.000000  1.000000  3.000000  3.0
1  3.000000    3.0  3.000000  3.000000  3.000000  2.000000  3.000000  3.0
2  0.000000    0.0  0.000000  0.000000  0.000000  3.000000  0.666667  0.0
3  0.000000    0.0  0.652174  0.000000  1.000000  0.000000  3.000000  1.0
4  2.000000    2.0  0.000000  0.000000  0.000000  1.000000  0.000000  0.0
5  0.000000    0.0  0.000000  0.000000  1.000000  0.000000  0.000000  0.0
6  3.000000    1.0  2.000000  0.000000  0.000000  1.000000  2.000000  0.0
7  0.000000    0.0  0.000000  0.000000  0.000000  2.000000  0.000000  0.0
8  0.578947    0.0  2.000000  0.000000  0.000000  0.000000  3.000000  0.0
```

9	3.000000	3.0	3.000000	2.000000	2.000000	3.000000	2.000000	3.0
10	0.000000	0.0	0.000000	0.000000	0.000000	2.000000	0.000000	0.0
11	0.000000	0.0	1.000000	0.000000	1.000000	0.000000	0.000000	0.0
12	1.000000	1.0	3.000000	2.000000	0.000000	0.000000	3.000000	1.0
13	1.000000	2.0	0.000000	0.000000	1.000000	3.000000	0.000000	0.0
14	2.000000	0.0	2.000000	0.000000	1.000000	0.000000	2.000000	1.0
15	3.000000	3.0	2.000000	3.000000	3.000000	0.000000	3.000000	3.0
16	1.000000	2.0	3.000000	1.000000	1.000000	3.000000	2.000000	1.0
17	3.000000	3.0	3.000000	3.000000	3.000000	2.000000	3.000000	3.0
18	0.000000	0.0	0.000000	0.000000	0.000000	3.000000	0.000000	0.0
19	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
20	0.000000	0.0	1.000000	0.000000	0.000000	0.000000	0.000000	0.0
21	3.000000	3.0	3.000000	3.000000	3.000000	1.000000	3.000000	3.0
22	3.000000	3.0	0.000000	3.000000	3.000000	3.000000	3.000000	3.0
23	0.000000	0.0	3.000000	2.000000	1.000000	0.000000	3.000000	1.0
24	3.000000	3.0	3.000000	3.000000	3.000000	3.000000	3.000000	3.0
25	3.000000	3.0	2.000000	2.000000	2.000000	0.000000	0.000000	2.0
26	0.000000	0.0	0.000000	0.000000	0.000000	1.000000	0.000000	0.0
27	0.000000	0.0	3.000000	2.000000	2.000000	0.733333	0.000000	2.0
28	0.000000	0.0	2.000000	0.000000	0.000000	1.000000	0.000000	0.0
29	2.000000	2.0	0.000000	2.000000	1.000000	3.000000	3.000000	2.0
..	...	...	...	...	...	...	...	...
506	2.000000	2.0	3.000000	3.000000	2.000000	2.000000	3.000000	3.0
507	2.000000	3.0	1.000000	2.000000	1.000000	2.000000	0.000000	1.0
508	0.000000	1.0	3.000000	1.000000	1.000000	0.000000	2.000000	1.0
509	0.000000	0.0	2.000000	1.000000	1.000000	2.000000	1.000000	1.0
510	1.000000	2.0	1.000000	3.000000	2.695652	2.000000	2.000000	3.0
511	0.000000	1.0	0.000000	0.000000	0.000000	2.000000	0.000000	0.0
512	0.000000	0.0	1.000000	0.000008	0.000000	0.000000	1.000000	0.0
513	2.000000	1.0	2.000000	2.000000	1.802817	0.000000	1.000000	3.0
514	0.000000	1.0	2.000000	1.000000	3.000000	2.000000	2.000000	2.0
515	1.000000	2.0	1.000000	1.000000	1.000000	3.000000	0.000000	1.0
516	3.000000	2.0	2.333333	3.000000	3.000000	1.000000	1.000000	2.0
517	0.000000	0.0	2.000000	2.000000	1.000000	2.000000	0.000000	2.0
518	0.000000	0.0	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
519	1.000000	1.0	3.000000	1.000000	1.000000	0.000000	1.533333	1.0
520	3.000000	3.0	3.000000	3.000000	3.000000	2.000000	3.000000	3.0
521	2.000000	3.0	1.000000	1.000000	2.000000	3.000000	0.000000	2.0
522	0.000000	0.0	2.000000	0.650000	1.000000	1.000000	1.950000	1.0
523	3.000000	3.0	2.000000	2.000000	2.000000	3.000000	0.000000	2.0
524	1.000000	2.0	2.000000	3.000000	3.000000	2.000000	2.000000	3.0
525	2.000000	2.0	2.000000	2.000000	2.000000	1.000014	2.000000	2.0
526	2.000000	1.0	1.000000	1.000000	0.000000	1.000000	0.000000	1.0
527	0.000000	0.0	1.000000	0.000000	1.000000	1.000000	2.000000	1.0
528	2.000000	2.0	2.000000	2.000000	2.000000	3.000000	3.000000	2.0
529	1.000000	1.0	2.000000	2.000000	1.000000	1.000000	1.680000	1.0
530	2.000000	1.0	1.000000	1.000000	1.000000	2.000000	1.000000	1.0
531	2.000000	2.0	2.000000	2.000000	3.000000	2.000000	3.000000	2.0

532	2.000000	3.0	2.000000	1.000000	1.000000	2.000000	2.000000	1.0
533	1.000000	1.0	1.363636	2.000000	1.000000	2.000000	2.000000	2.0
534	0.000000	0.0	0.000000	0.000000	0.000000	2.000000	0.000000	0.0
535	0.000000	0.0	1.000000	1.000000	1.000000	0.000000	0.000000	1.0

	EM
0	3.000000
1	3.000000
2	0.000000
3	0.000000
4	0.000000
5	0.000000
6	1.000000
7	0.000000
8	0.000000
9	2.666667
10	1.000000
11	0.000000
12	0.000000
13	1.000000
14	0.000000
15	3.000000
16	2.000000
17	3.000000
18	1.000000
19	2.000000
20	0.000000
21	3.000000
22	3.000000
23	2.000000
24	3.000000
25	3.000000
26	1.000000
27	0.000000
28	0.000000
29	3.000000
..	...
506	3.000000
507	1.000000
508	1.000000
509	1.000000
510	2.000000
511	2.000000
512	0.000000
513	1.197183
514	1.000000
515	1.000000
516	2.000000

```
517  1.000000
518  0.000000
519  0.000000
520  2.000000
521  2.000000
522  1.000000
523  1.000000
524  1.000000
525  2.000007
526  0.904762
527  0.000000
528  2.000000
529  3.000000
530  1.000000
531  2.000000
532  1.000000
533  3.000000
534  1.000000
535  1.999979

[536 rows x 9 columns]
```

11 Calculating MSE for Imputation methods

In [27]:
```python
from sklearn.metrics import mean_squared_error
def calc_MSE(df1,df2):
    return mean_squared_error(df1,df2)

mse_knn = calc_MSE(df_binning2_copy,df_knn)


mse_mean = calc_MSE(df_binning2_copy,df_imputed_mean)
mse_median = calc_MSE(df_binning2_copy,df_imputed_median)
mse_dict ={"mse_knn":[mse_knn],"mse_median":[mse_median],"mse_mean":[mse_mean]}
pd.DataFrame(mse_dict)
```

Out[27]:

	mse_knn	mse_mean	mse_median
0	0.039731	0.072492	0.072492