RAAIM Outputs

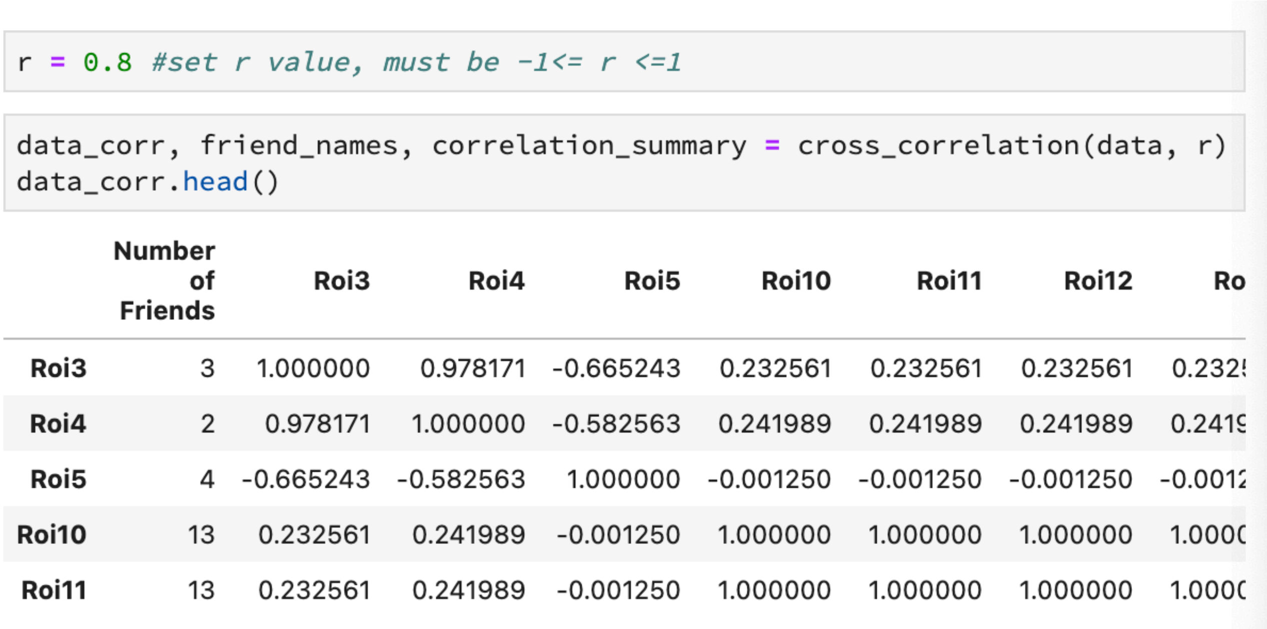
Analysis

Output 1

- `data.head()` of normalized values

Output 2: Friends CC

- Cross-Correlation Matrix of “Friends”

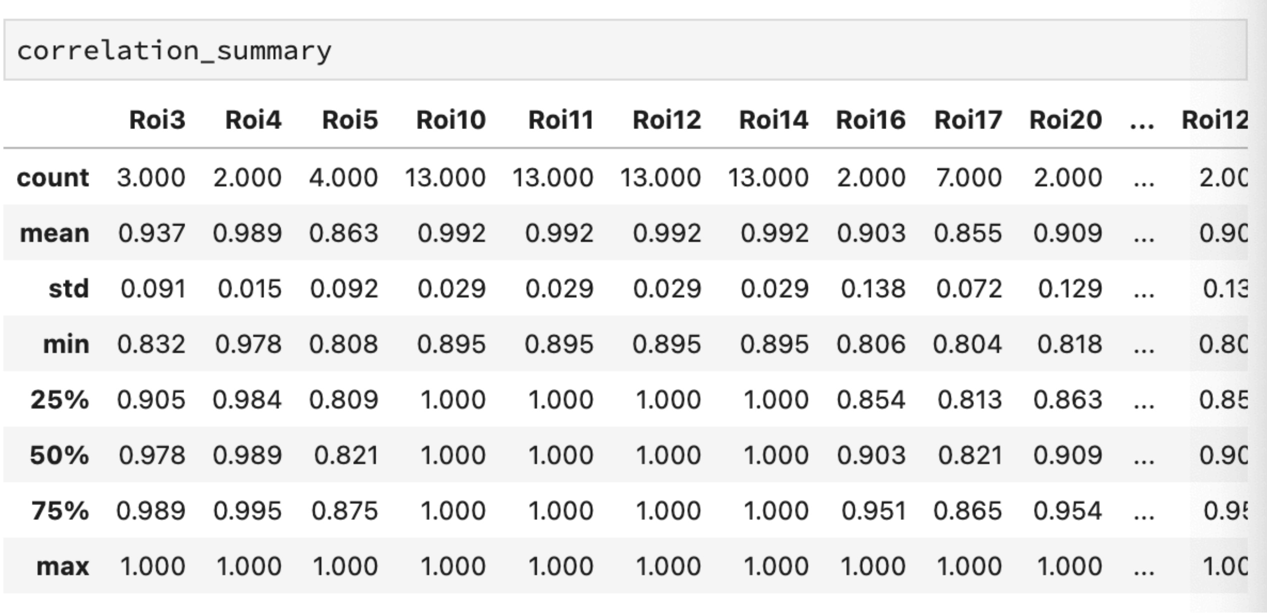


```
r = 0.8 #set r value, must be -1<= r <=1
```

```
data_corr, friend_names, correlation_summary = cross_correlation(data, r)
data_corr.head()
```

	Number of Friends	Roi3	Roi4	Roi5	Roi10	Roi11	Roi12	Ro
Roi3	3	1.000000	0.978171	-0.665243	0.232561	0.232561	0.232561	0.2325
Roi4	2	0.978171	1.000000	-0.582563	0.241989	0.241989	0.241989	0.2419
Roi5	4	-0.665243	-0.582563	1.000000	-0.001250	-0.001250	-0.001250	-0.0012
Roi10	13	0.232561	0.241989	-0.001250	1.000000	1.000000	1.000000	1.0000
Roi11	13	0.232561	0.241989	-0.001250	1.000000	1.000000	1.000000	1.0000

- Cross-Correlation Summary



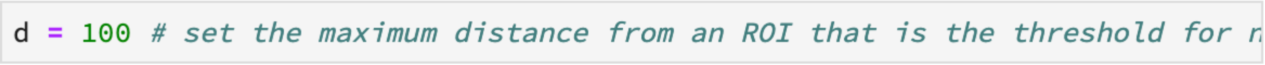
```
correlation_summary
```

	Roi3	Roi4	Roi5	Roi10	Roi11	Roi12	Roi14	Roi16	Roi17	Roi20	...	Roi12
count	3.000	2.000	4.000	13.000	13.000	13.000	13.000	2.000	7.000	2.000	...	2.00
mean	0.937	0.989	0.863	0.992	0.992	0.992	0.992	0.903	0.855	0.909	...	0.90
std	0.091	0.015	0.092	0.029	0.029	0.029	0.029	0.138	0.072	0.129	...	0.13
min	0.832	0.978	0.808	0.895	0.895	0.895	0.895	0.806	0.804	0.818	...	0.80
25%	0.905	0.984	0.809	1.000	1.000	1.000	1.000	0.854	0.813	0.863	...	0.85
50%	0.978	0.989	0.821	1.000	1.000	1.000	1.000	0.903	0.821	0.909	...	0.90
75%	0.989	0.995	0.875	1.000	1.000	1.000	1.000	0.951	0.865	0.954	...	0.95
max	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	...	1.00

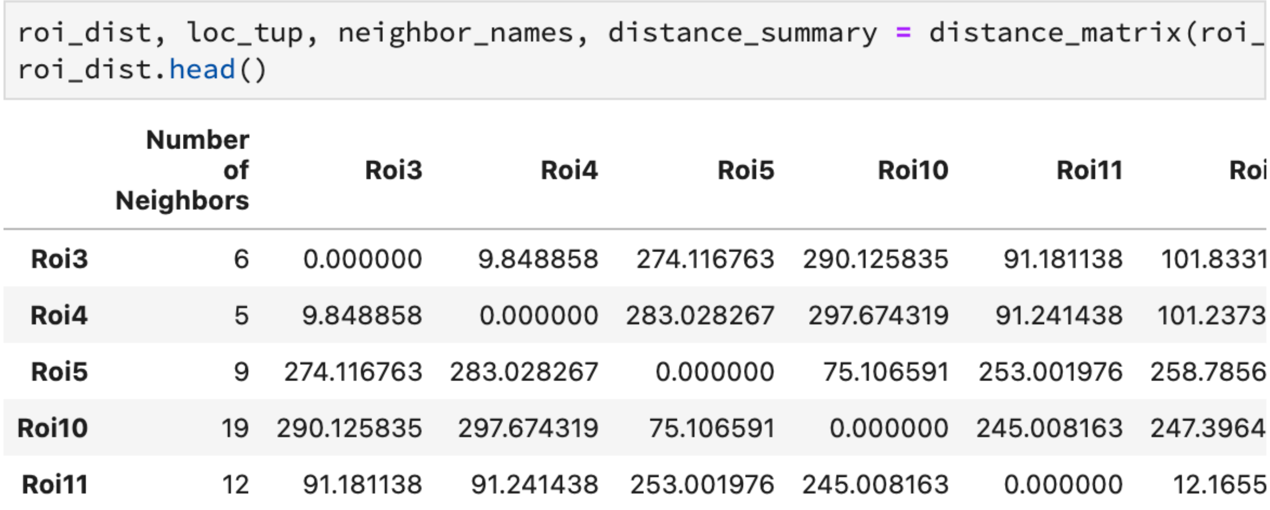
-> Export both to .csv files

Output 3: Neighbors

- Distance Matrix of “Neighbors”



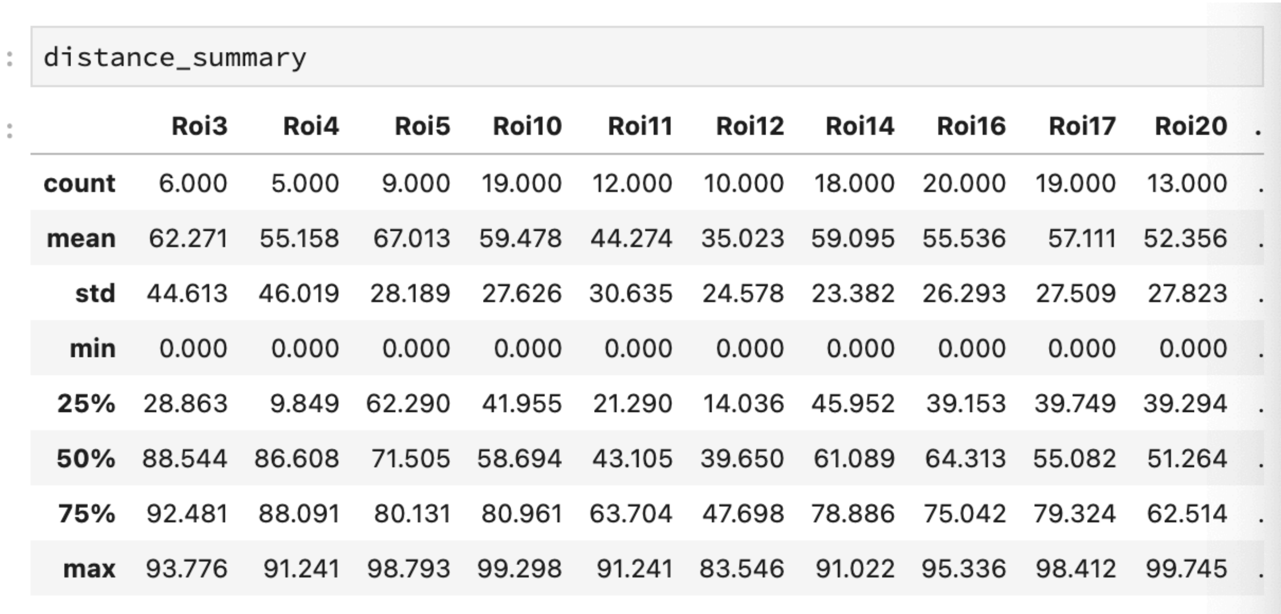
```
d = 100 # set the maximum distance from an ROI that is the threshold for r
```



```
roi_dist, loc_tup, neighbor_names, distance_summary = distance_matrix(roi_
roi_dist.head()
```

	Number of Neighbors	Roi3	Roi4	Roi5	Roi10	Roi11	Roi
Roi3	6	0.000000	9.848858	274.116763	290.125835	91.181138	101.8331
Roi4	5	9.848858	0.000000	283.028267	297.674319	91.241438	101.2373
Roi5	9	274.116763	283.028267	0.000000	75.106591	253.001976	258.7856
Roi10	19	290.125835	297.674319	75.106591	0.000000	245.008163	247.3964
Roi11	12	91.181138	91.241438	253.001976	245.008163	0.000000	12.1655

- Distance Matrix Summary



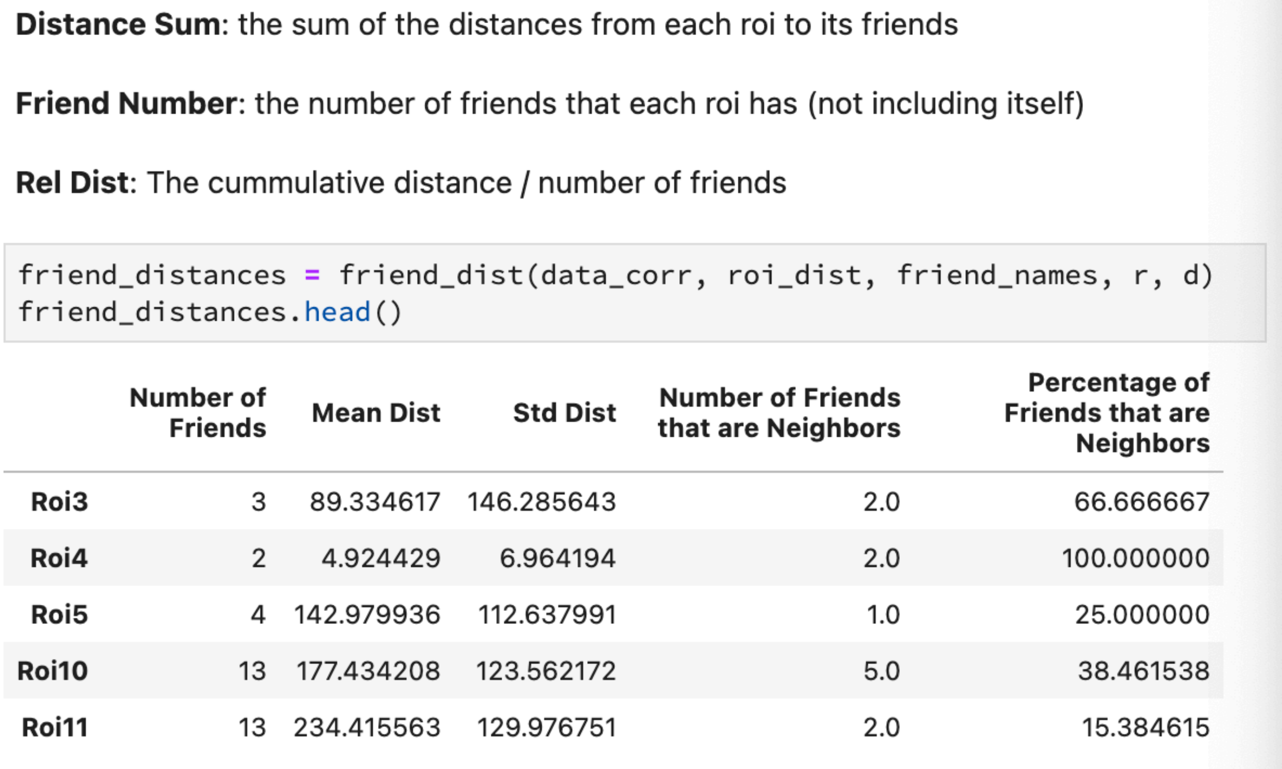
```
distance_summary
```

	Roi3	Roi4	Roi5	Roi10	Roi11	Roi12	Roi14	Roi16	Roi17	Roi20	.
count	6.000	5.000	9.000	19.000	12.000	10.000	18.000	20.000	19.000	13.000	.
mean	62.271	55.158	67.013	59.478	44.274	35.023	59.095	55.536	57.111	52.356	.
std	44.613	46.019	28.189	27.626	30.635	24.578	23.382	26.293	27.509	27.823	.
min	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	.
25%	28.863	9.849	62.290	41.955	21.290	14.036	45.952	39.153	39.749	39.294	.
50%	88.544	86.608	71.505	58.694	43.105	39.650	61.089	64.313	55.082	51.264	.
75%	92.481	88.091	80.131	80.961	63.704	47.698	78.886	75.042	79.324	62.514	.
max	93.776	91.241	98.793	99.298	91.241	83.546	91.022	95.336	98.412	99.745	.

-> Export both to .csv files

Output 4: Friends/Neighbors

- “Friends” in terms of “Neighbors”



Distance Sum: the sum of the distances from each roi to its friends

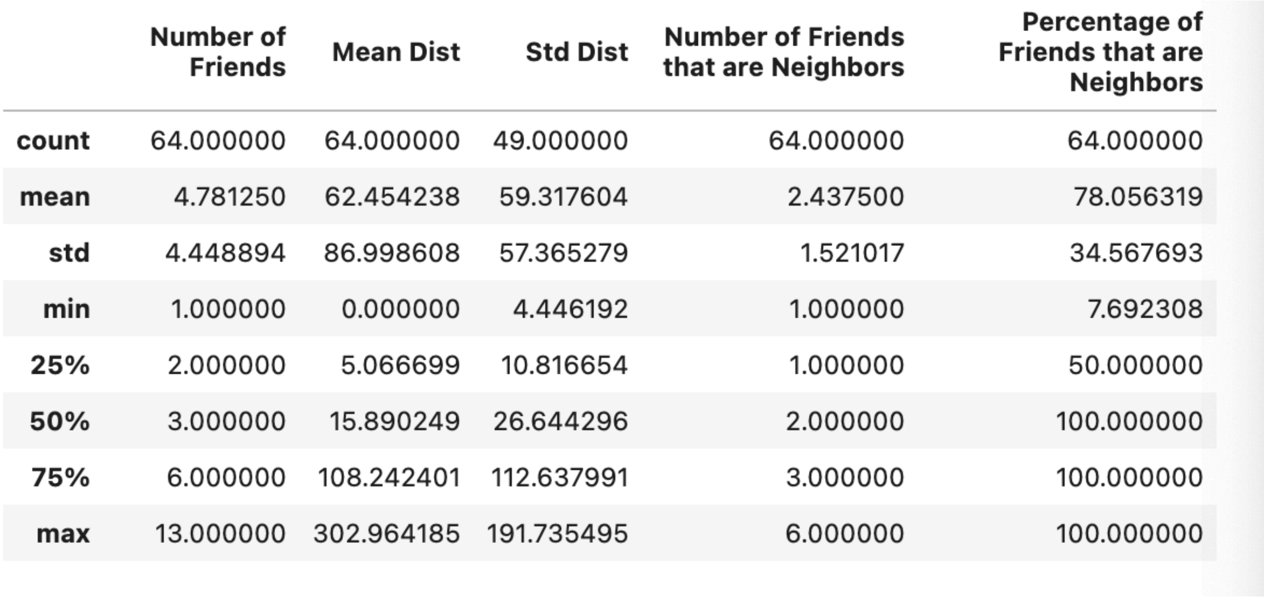
Friend Number: the number of friends that each roi has (not including itself)

Rel Dist: The cummulative distance / number of friends

```
friend_distances = friend_dist(data_corr, roi_dist, friend_names, r, d)
friend_distances.head()
```

	Number of Friends	Mean Dist	Std Dist	Number of Friends that are Neighbors	Percentage of Friends that are Neighbors
Roi3	3	89.334617	146.285643	2.0	66.666667
Roi4	2	4.924429	6.964194	2.0	100.000000
Roi5	4	142.979936	112.637991	1.0	25.000000
Roi10	13	177.434208	123.562172	5.0	38.461538
Roi11	13	234.415563	129.976751	2.0	15.384615

- “Friends” in terms of “Neighbors” Summary

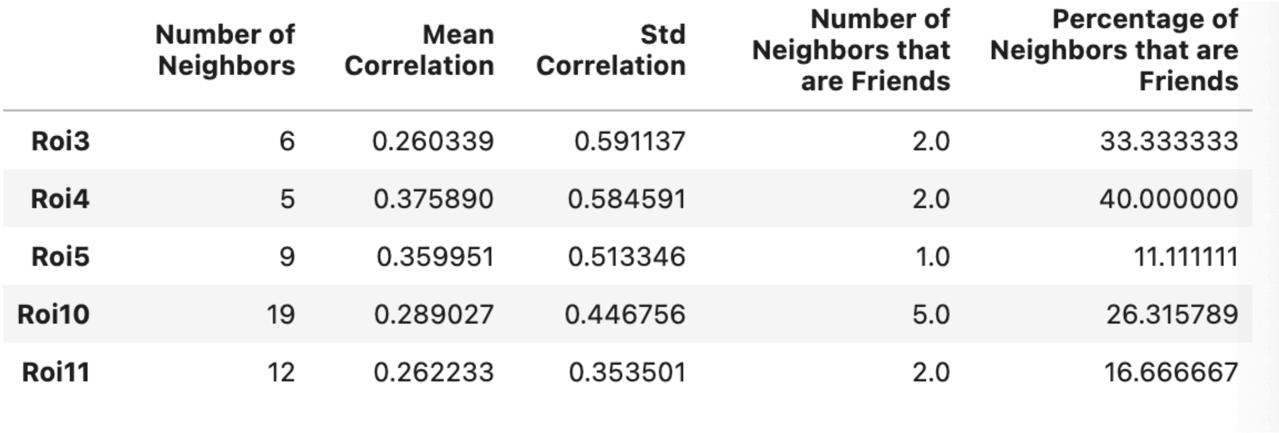


	Number of Friends	Mean Dist	Std Dist	Number of Friends that are Neighbors	Percentage of Friends that are Neighbors
count	64.000000	64.000000	49.000000	64.000000	64.000000
mean	4.781250	62.454238	59.317604	2.437500	78.056319
std	4.448894	86.998608	57.365279	1.521017	34.567693
min	1.000000	0.000000	4.446192	1.000000	7.692308
25%	2.000000	5.066699	10.816654	1.000000	50.000000
50%	3.000000	15.890249	26.644296	2.000000	100.000000
75%	6.000000	108.242401	112.637991	3.000000	100.000000
max	13.000000	302.964185	191.735495	6.000000	100.000000

-> Export both to .csv files

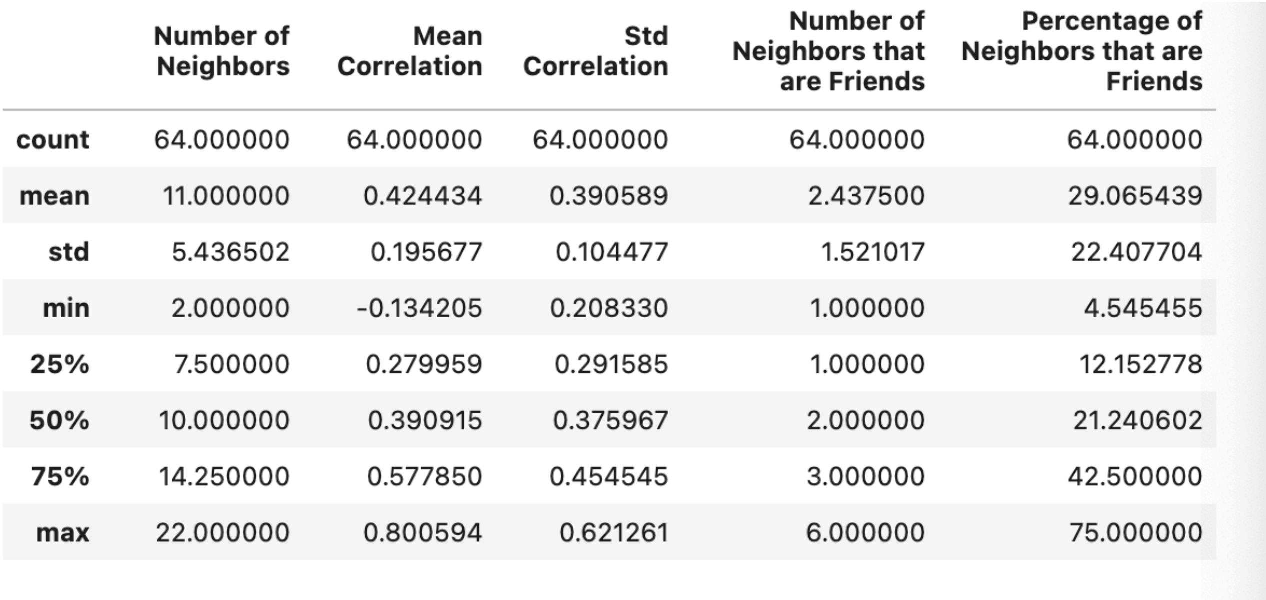
Output 5: Neighbors/Friends

- Neighbors in terms of Friends



	Number of Neighbors	Mean Correlation	Std Correlation	Number of Neighbors that are Friends	Percentage of Neighbors that are Friends
Roi3	6	0.260339	0.591137	2.0	33.333333
Roi4	5	0.375890	0.584591	2.0	40.000000
Roi5	9	0.359951	0.513346	1.0	11.111111
Roi10	19	0.289027	0.446756	5.0	26.315789
Roi11	12	0.262233	0.353501	2.0	16.666667

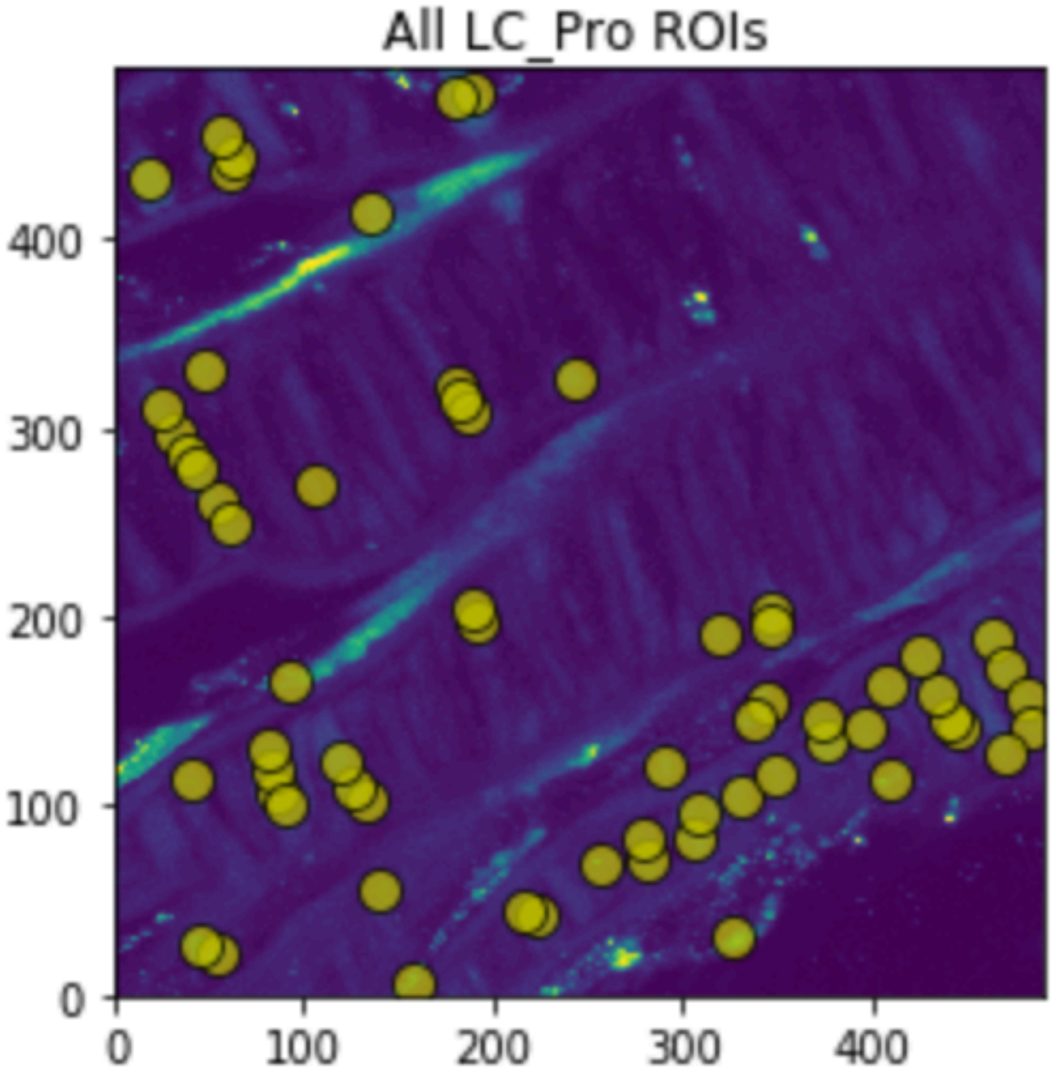
- Neighbors in terms of Friends Summary



	Number of Neighbors	Mean Correlation	Std Correlation	Number of Neighbors that are Friends	Percentage of Neighbors that are Friends
count	64.000000	64.000000	64.000000	64.000000	64.000000
mean	11.000000	0.424434	0.390589	2.437500	29.065439
std	5.436502	0.195677	0.104477	1.521017	22.407704
min	2.000000	-0.134205	0.208330	1.000000	4.545455
25%	7.500000	0.279959	0.291585	1.000000	12.152778
50%	10.000000	0.390915	0.375967	2.000000	21.240602
75%	14.250000	0.577850	0.454545	3.000000	42.500000
max	22.000000	0.800594	0.621261	6.000000	75.000000

Figures

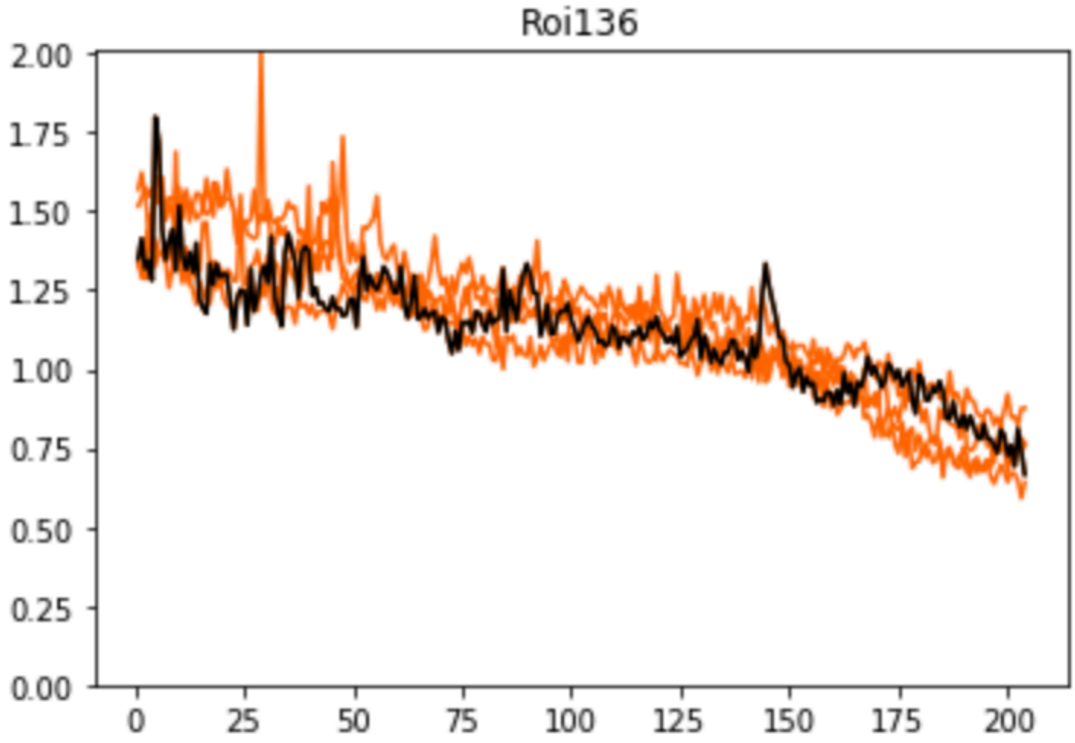
Roi overlay with name-calling

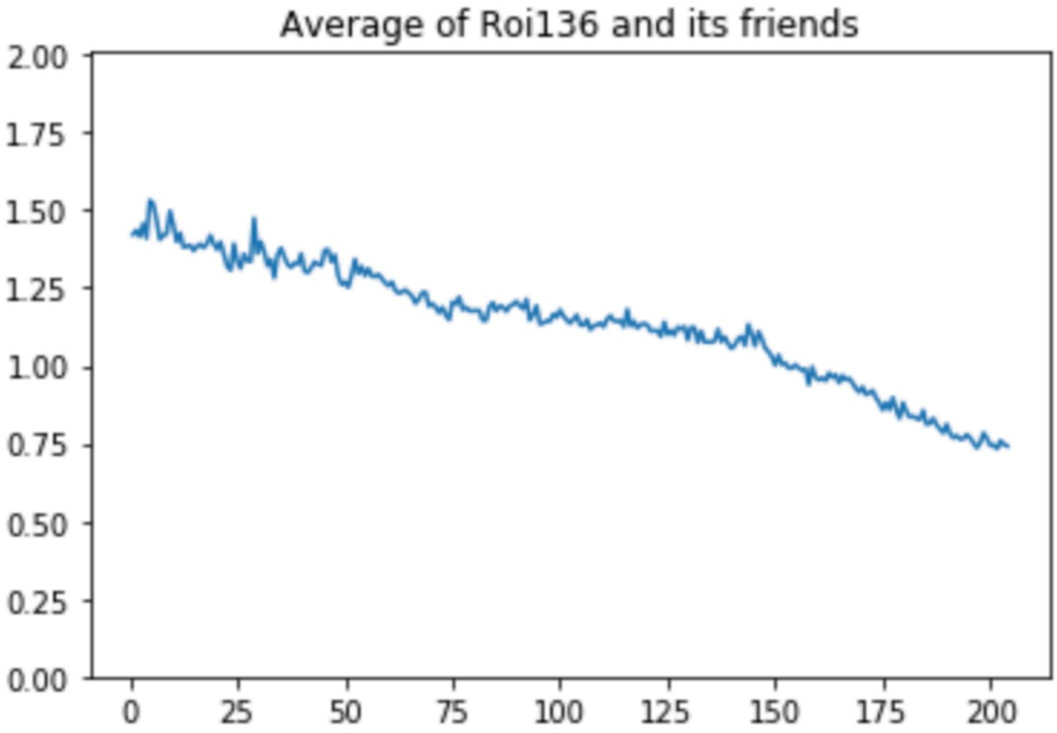


Select Roi & Friends Plot (broken in RAAIM)

Line Plots: all ROIs, one ROI (broken in RAAIM)

Stacked line plot: ROI & Friends

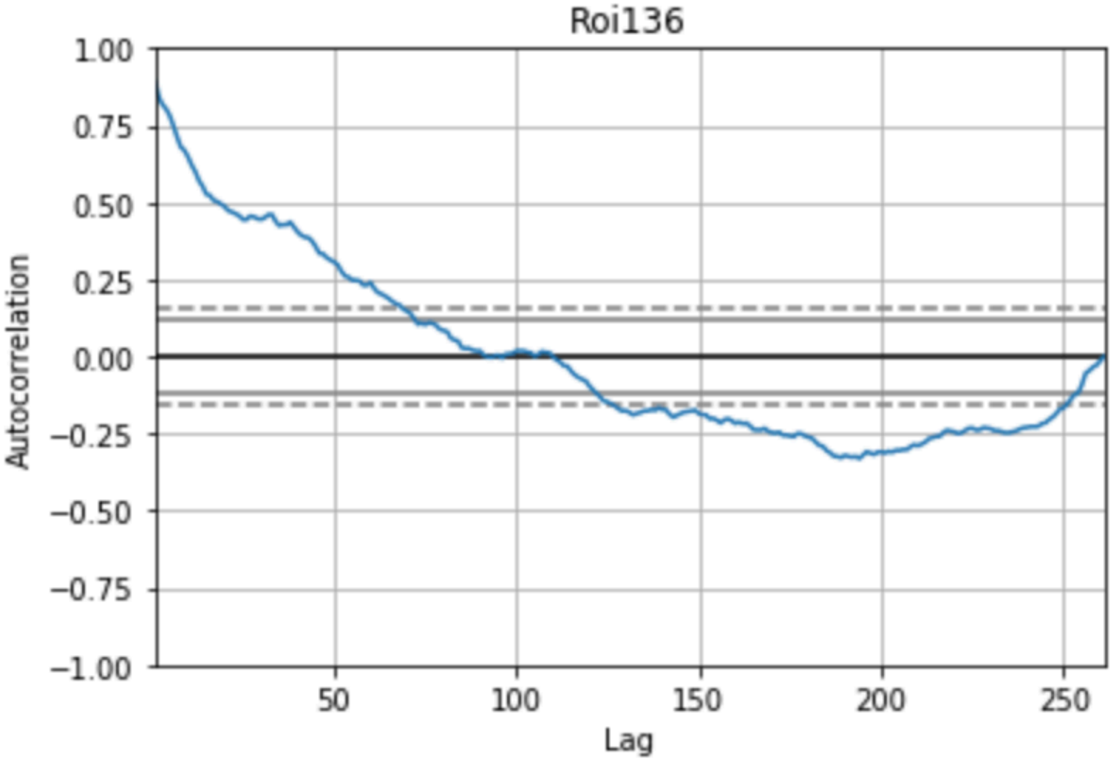




Autocorrelation Plot

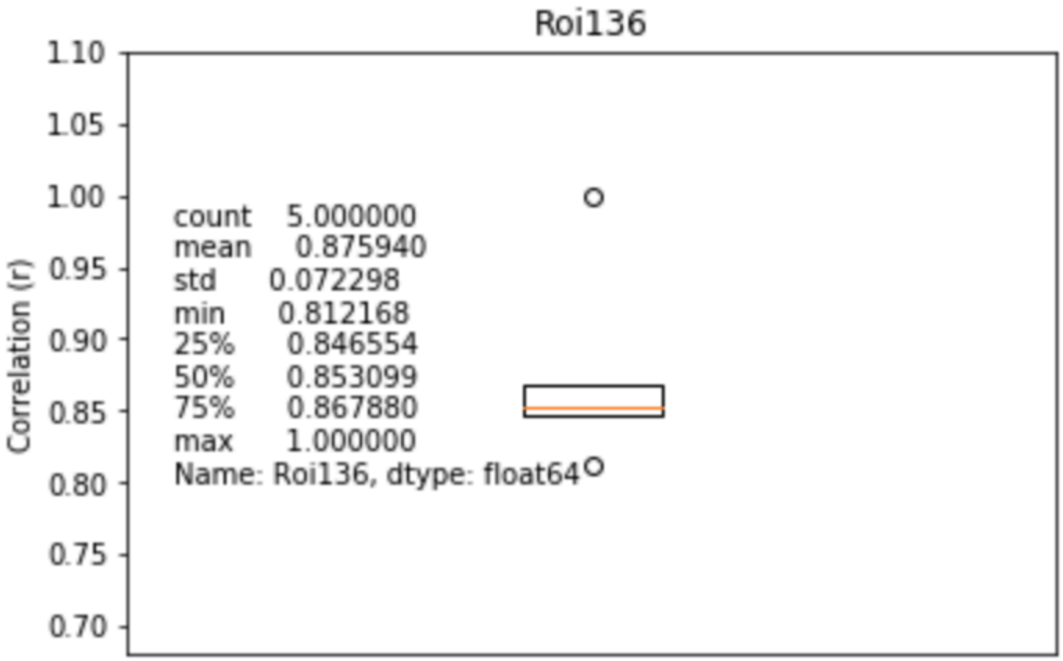
```
roi = 'Roi136'

autocorrelation_plot(data[roi])
plt.title(roi)
plt.show()
```



Box & Whisker Plots

```
boxandwhiskers(roi = 'Roi136', flavor = 'corr', data_corr
```



```
count    5.000000
mean     0.875940
std      0.072298
min      0.812168
25%      0.846554
50%      0.853099
75%      0.867880
max      1.000000
Name: Roi136, dtype: float64
```