Plotting

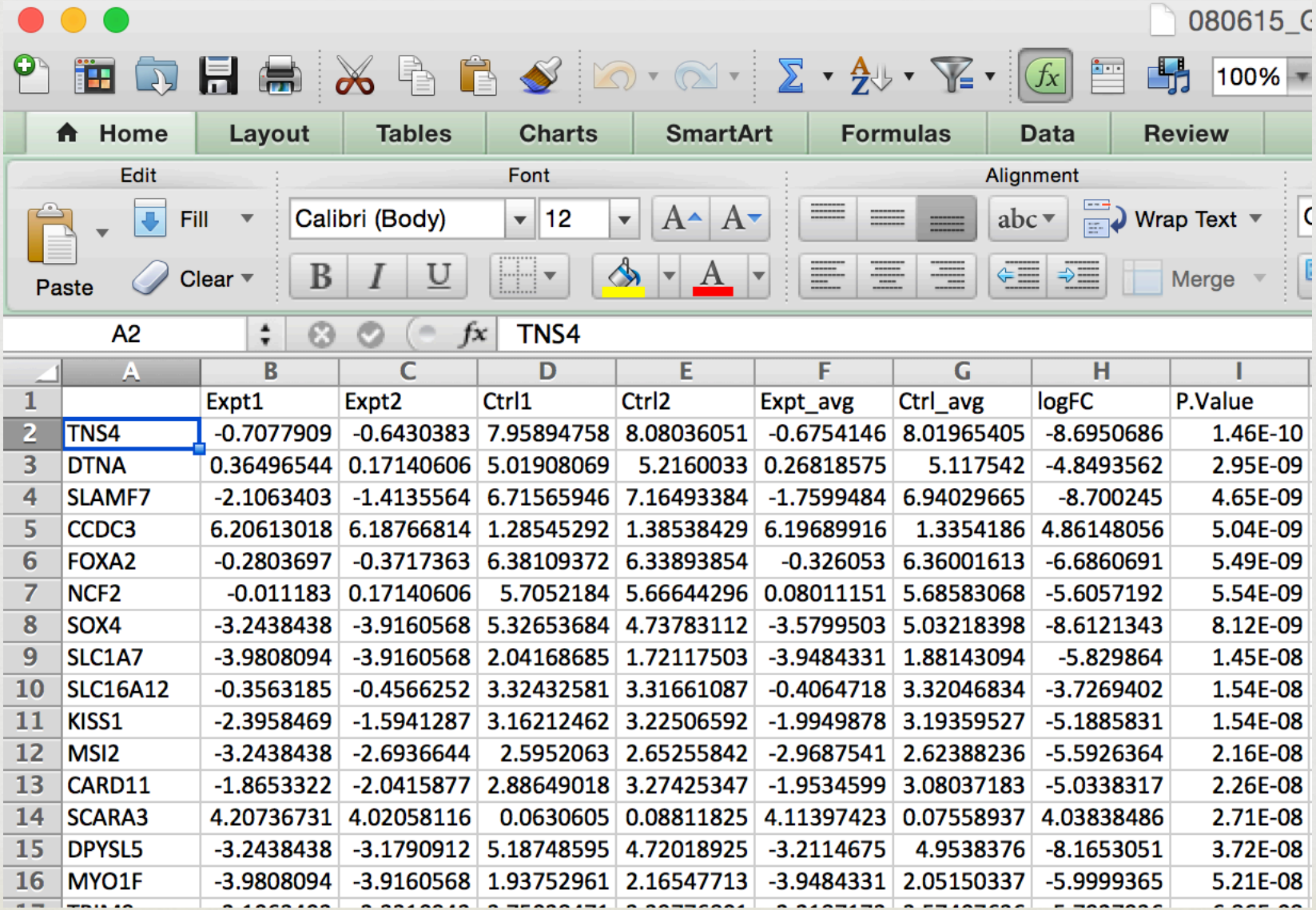
With Pandas

Agenda

- Download Gene_expression.xlsx from the website under the tutorials tab (Download the slides if you to spend less typing code today)
- Pandas Review
- Plots: scatter
- Plots: histogram
- Plots: multiple figures on a plot

- Everything in blue today, type into your terminal!

RNA seq data



080615_C

100%

Home | Layout | Tables | Charts | SmartArt | Formulas | Data | Review

Edit | Font | Alignment

Paste | Fill | Clear | Calibri (Body) | 12 | Wrap Text | Merge

A2 | TNS4

	A	B	C	D	E	F	G	H	I
1		Expt1	Expt2	Ctrl1	Ctrl2	Expt_avg	Ctrl_avg	logFC	P.Value
2	TNS4	-0.7077909	-0.6430383	7.95894758	8.08036051	-0.6754146	8.01965405	-8.6950686	1.46E-10
3	DTNA	0.36496544	0.17140606	5.01908069	5.2160033	0.26818575	5.117542	-4.8493562	2.95E-09
4	SLAMF7	-2.1063403	-1.4135564	6.71565946	7.16493384	-1.7599484	6.94029665	-8.700245	4.65E-09
5	CCDC3	6.20613018	6.18766814	1.28545292	1.38538429	6.19689916	1.3354186	4.86148056	5.04E-09
6	FOXA2	-0.2803697	-0.3717363	6.38109372	6.33893854	-0.326053	6.36001613	-6.6860691	5.49E-09
7	NCF2	-0.011183	0.17140606	5.7052184	5.66644296	0.08011151	5.68583068	-5.6057192	5.54E-09
8	SOX4	-3.2438438	-3.9160568	5.32653684	4.73783112	-3.5799503	5.03218398	-8.6121343	8.12E-09
9	SLC1A7	-3.9808094	-3.9160568	2.04168685	1.72117503	-3.9484331	1.88143094	-5.829864	1.45E-08
10	SLC16A12	-0.3563185	-0.4566252	3.32432581	3.31661087	-0.4064718	3.32046834	-3.7269402	1.54E-08
11	KISS1	-2.3958469	-1.5941287	3.16212462	3.22506592	-1.9949878	3.19359527	-5.1885831	1.54E-08
12	MSI2	-3.2438438	-2.6936644	2.5952063	2.65255842	-2.9687541	2.62388236	-5.5926364	2.16E-08
13	CARD11	-1.8653322	-2.0415877	2.88649018	3.27425347	-1.9534599	3.08037183	-5.0338317	2.26E-08
14	SCARA3	4.20736731	4.02058116	0.0630605	0.08811825	4.11397423	0.07558937	4.03838486	2.71E-08
15	DPYSL5	-3.2438438	-3.1790912	5.18748595	4.72018925	-3.2114675	4.9538376	-8.1653051	3.72E-08
16	MYO1F	-3.9808094	-3.9160568	1.93752961	2.16547713	-3.9484331	2.05150337	-5.9999365	5.21E-08

Begin plotting script

- Create a script (and also have an interactive session running):

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

Review on Pandas

- example = pd.read_excel('<path>/080615_Gene_expression.xlsx',index_col=0)
- example.columns
- example.index
- example.values
- Indexing:
  - Indexing looks like: dataframe[columns][rows] or dataframe.ix[rows, columns]
- Obtain the first 5 rows of example.

```
example.ix[0:5, :]
example[:][0:5]
example[0:5]
```

- Obtain the first column, then obtain the first four columns:

```
example['Expt1']

example[['Expt1', 'Expt2', 'Ctrl1', 'Ctrl2']]
```

Review on Pandas

- Obtain the first five rows and the first four columns:

```
example.ix[0:5, ['Expt1', 'Expt2', 'Ctrl1', 'Ctrl2']]
```

- Example of boolean series:

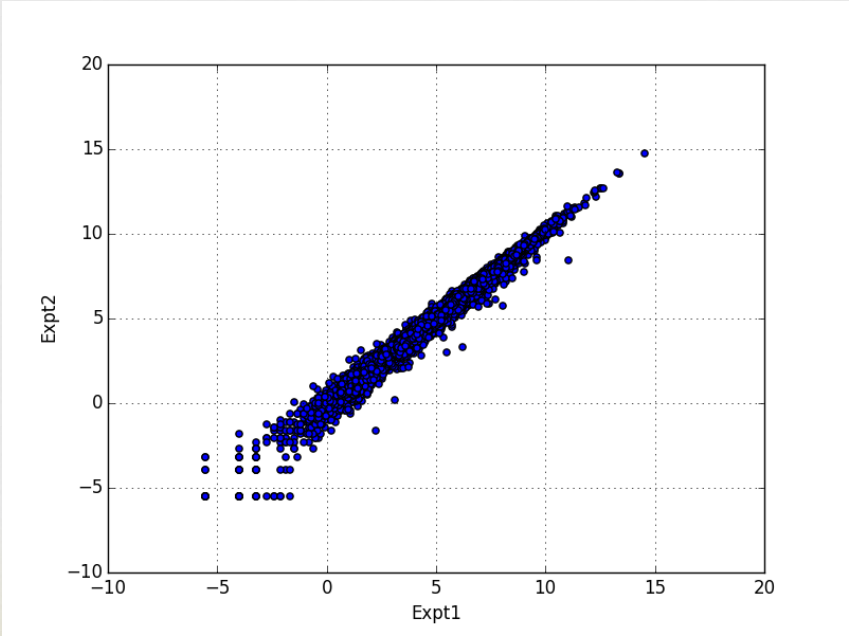
```
example['P.Value']<0.05
```

- Use boolean indexing to save rows that have a p.value < 0.05 as a new dataframe:

```
Newframe = example[example['P.Value']<0.05]
```

Basic plots using pandas

- Pandas plotting is good for large datasets!!!!
- Scatter plot:
- dataframe.plot(kind = , x = , y =)
- example.plot(kind = 'scatter', x = 'Expt1', y = 'Expt2')
- plt.show()



Closing the figure

- To close plot: click x button (this must be done every time there is an open figure)
- Save figure:
- example.plot(kind = 'scatter', x = 'Expt1', y = 'Expt2')
- plt.savefig('<path>/example_scatter1.png')
- The plotting figure still exists in memory after you save so if you want to make another figure you must clear the first.
- Clear figure:
- plt.clf()

Plot types

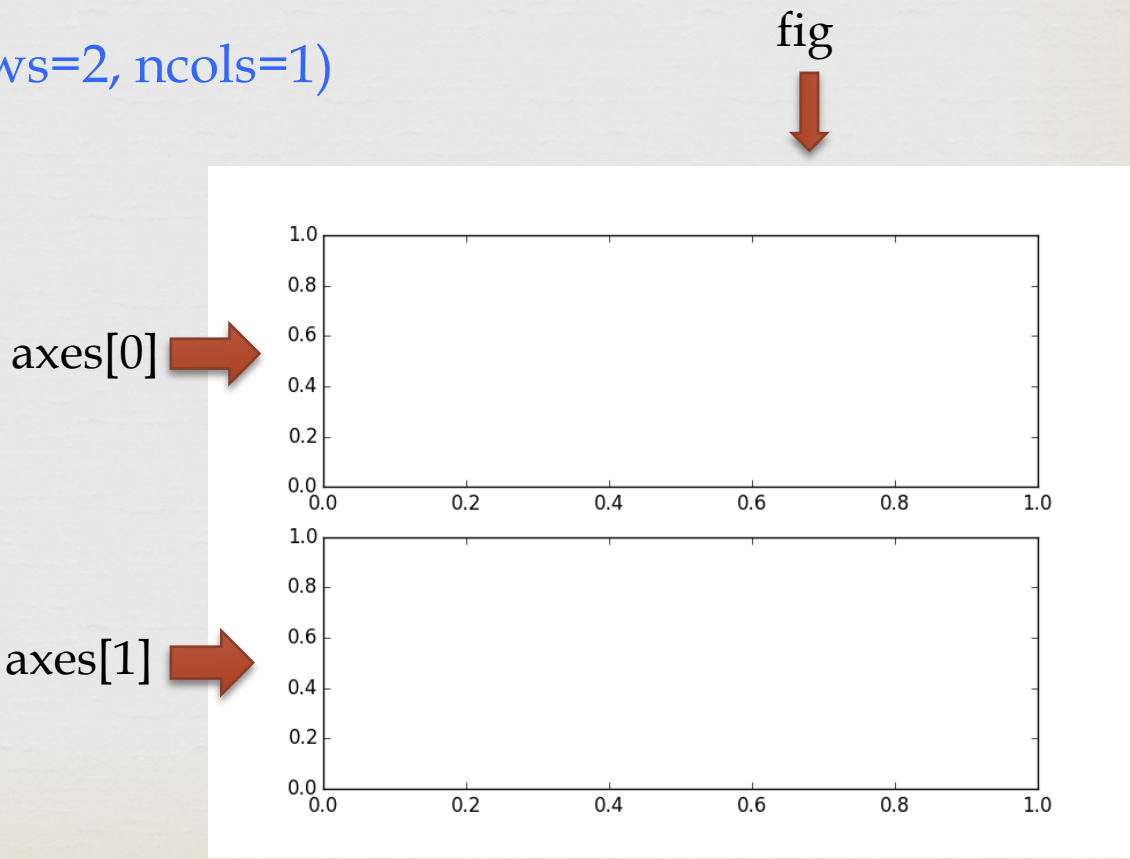
- Dataframe.plot(kind =)
- Kind keyword argument accepts:
  - 'line' as default
  - *'bar'* or *'barh'* for bar plots
  - *'hist'* for histogram
  - *'box'* for boxplot
  - *'kde'* or 'density' for density plots
  - *'area'* for area plots
  - *'scatter'* for scatter plots
  - *'hexbin'* for hexagonal bin plots
  - *'pie'* for pie plots
- Also can use these graph types:
  - *DataFrame.hist()*, and *DataFrame.boxplot()*

Histogram

- Make a histogram of all the columns of your dataframe
  - Use *DataFrame.hist()*, remember to replace *DataFrame* with the name you have called your dataframe
- Make a histogram of 1 column of your dataframe:
  - Use *DataFrame[column].hist()*

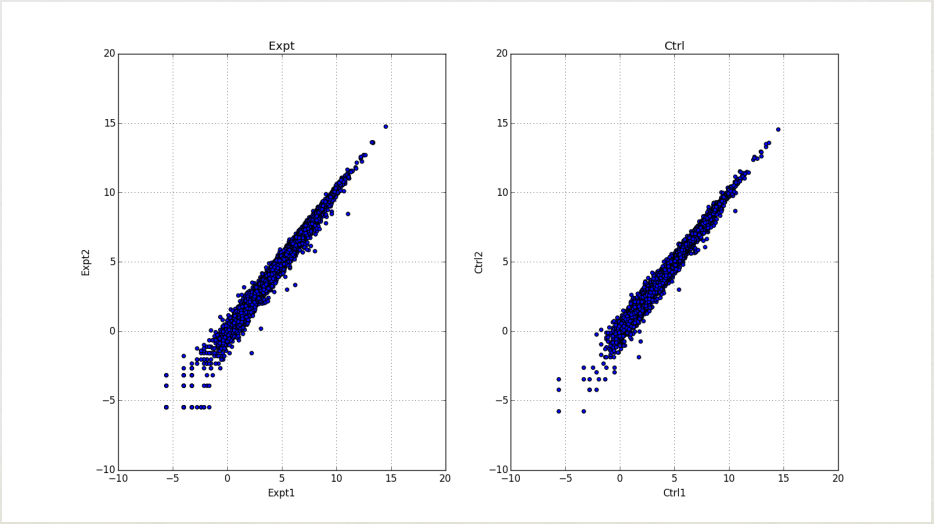
Working with multiple figures and axis

- Figure is the plotting area, axes are a way to access subplots
- fig, axes = plt.subplots(nrows=2, ncols=1)
- plt.show()



Filling each subplot (1Xn or nX1 dimension plot)

- Notice I switched the subplot dimensions:
- fig, axes = plt.subplots(nrows=1, ncols=2)
- example.plot(ax=axes[0],kind='scatter',x = 'Expt1', y = 'Expt2')
- axes[0].set_title('Expt')
- example.plot(ax=axes[1],kind='scatter',x = 'Ctrl1', y = 'Ctrl2')
- axes[1].set_title('Ctrl')
- plt.show()

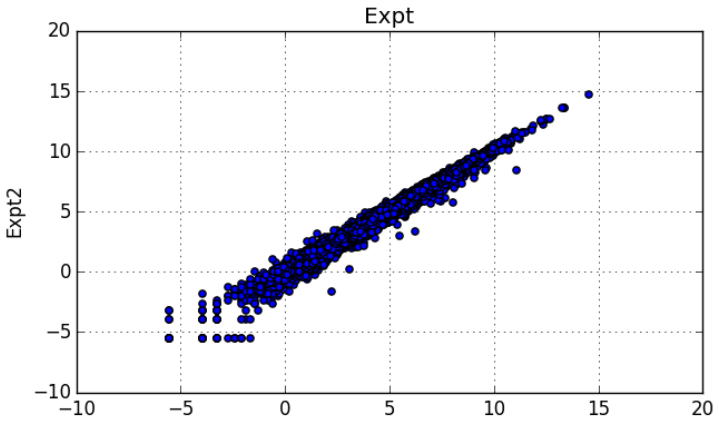


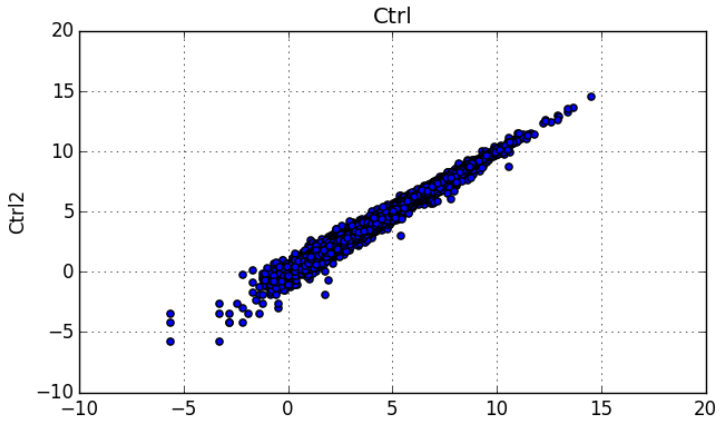
Filling each subplot (nXn dimension plot)

- fig, axes = plt.subplots(nrows=2, ncols=2)
- example.plot(ax=axes[0,0],kind='scatter',x = 'Expt1', y = 'Expt2')
- axes[0,0].set_title('Expt')
- example.plot(ax=axes[0,1],kind='scatter',x = 'Ctrl1', y = 'Ctrl2')
- axes[0,1].set_title('Ctrl')
- example['P.Value'].hist(ax=axes[1,0])
- axes[1,0].set_title('P.Value')
- example['Ctrl_avg'].plot(ax=axes[1,1], kind = 'kde')
- plt.show()

How to index multiple subplots

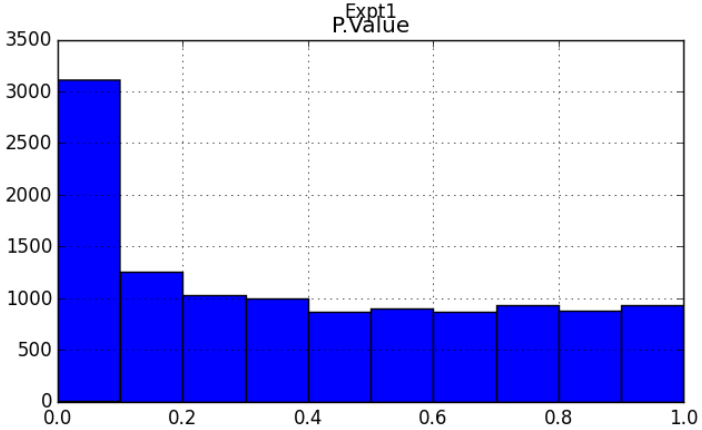
axes[0,0]

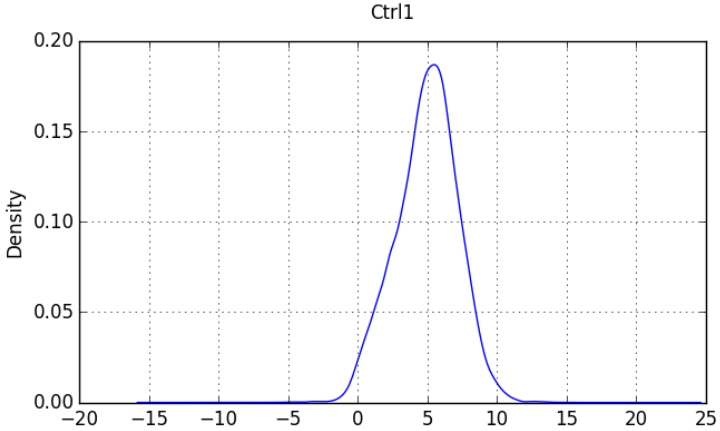




axes[0,1]

axes[1,0]

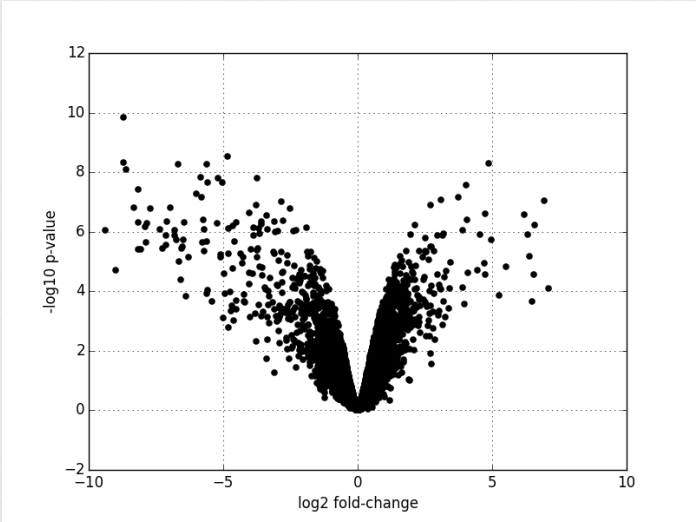




axes[1,1]

Homework

- Create a volcano plot: A volcano plot is a graph that shows effect size from a study. It is a type of scatter plot where the $-\log_{10}$ p-value is plotted on the y axis and the fold change (or in our case $\log_2$ fold-change) is on the x-axis.



Homework

- 1. Calculate the –log10 of the p value in the example dataframe (I will give you this answer):

```
data = -(np.log10(example['P.Value']))
```

- 2. Save algorithm from answer 1 as a dataframe: In order to save this you will need 2 lines of code:
  - Df = pd.DataFrame(data)
  - Fill in the code for this line with whatever you want to name the data:
  - Df.columns = [<your answer>]

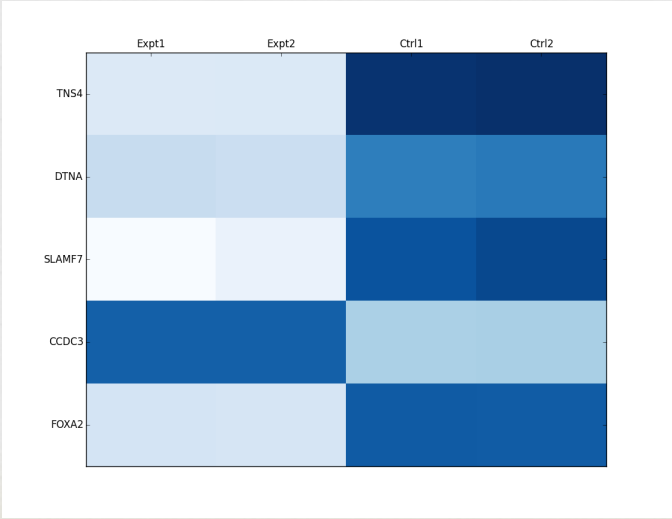
- 3. Concatenate your –log10 dataframe to your example dataframe. hint: you can use
  - newexample = pd.concat([example, Df], axis = 1)
- 4. Create scatter plot

Additional fun with plots

- Keria is going to include two more plot types on the website that you can play with:

codingclubnyumc.wordpress.com/coding-snippets/

heatmap



Distribution/scatter