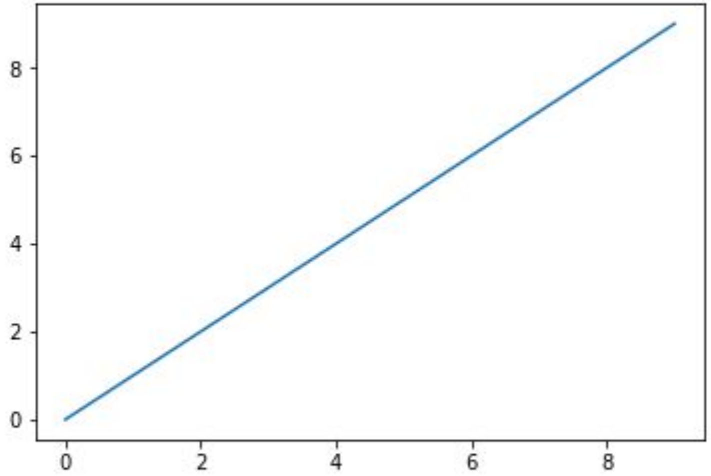
Visualization

Libraries for visualization

- matplotlib - base for other libraries
- seaborn - easier than matplotlib, better quality, suitable for work with Dataframes
- plotly - interactive presentations
- bokeh - haven't used it, probably interactive too

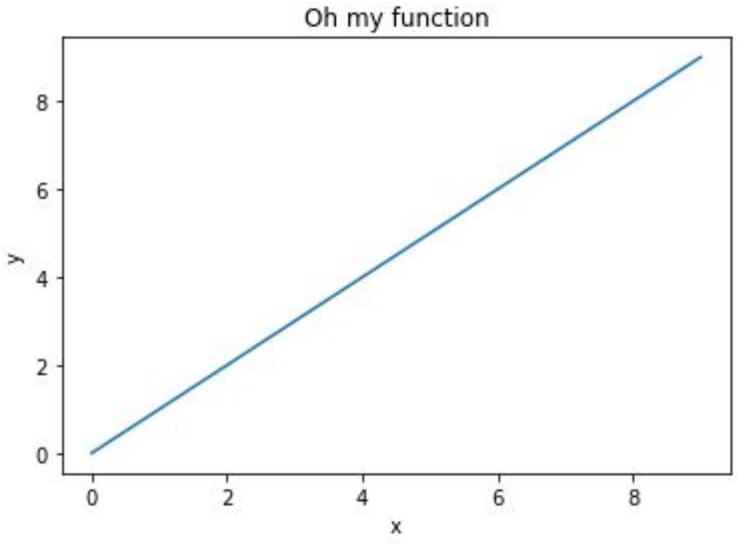
Matplotlib

```
import matplotlib.pyplot as plt



# Prepare data
xs = range(10)
ys = range(10)

plt.plot(xs, ys)
# Add plt.show() if you are working not in jupyter
```



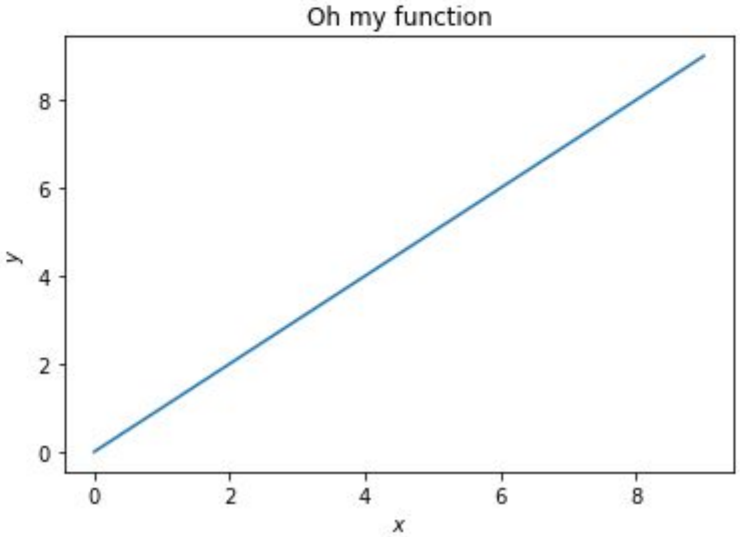
Customization

```
plt.plot(xs, ys)

# Customization
plt.title('Oh my function')
plt.xlabel('x')
plt.ylabel('y')
# Add plt.show() if you are work
```

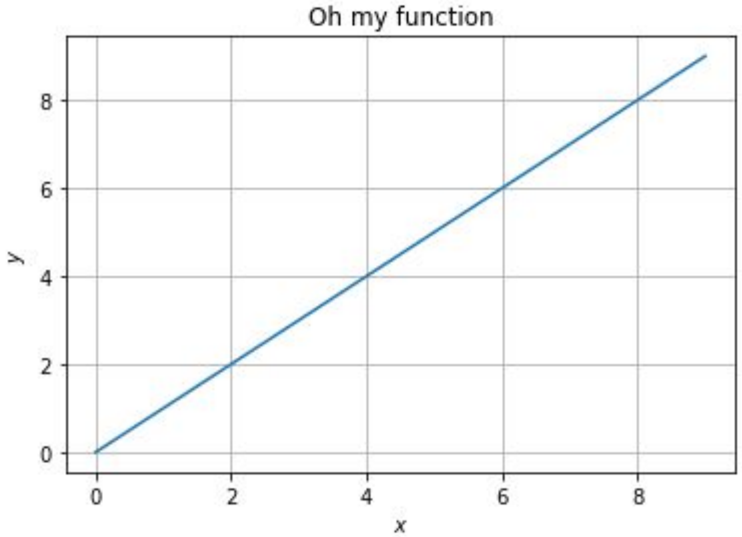


LaTeX

```
plt.plot(xs, ys)

# Customization
plt.title('Oh my function')
plt.xlabel('$x$')
plt.ylabel('$y$')
# Add plt.show() if you are wor
```



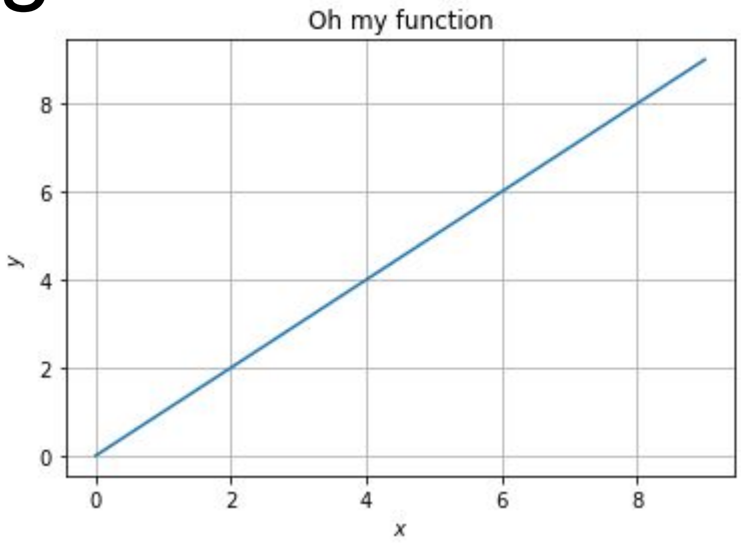
Grid



```
plt.plot(xs, ys)

# Customization
plt.title('Oh my function')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.grid(True)
# Add plt.show() if you are working not in jupyter
```

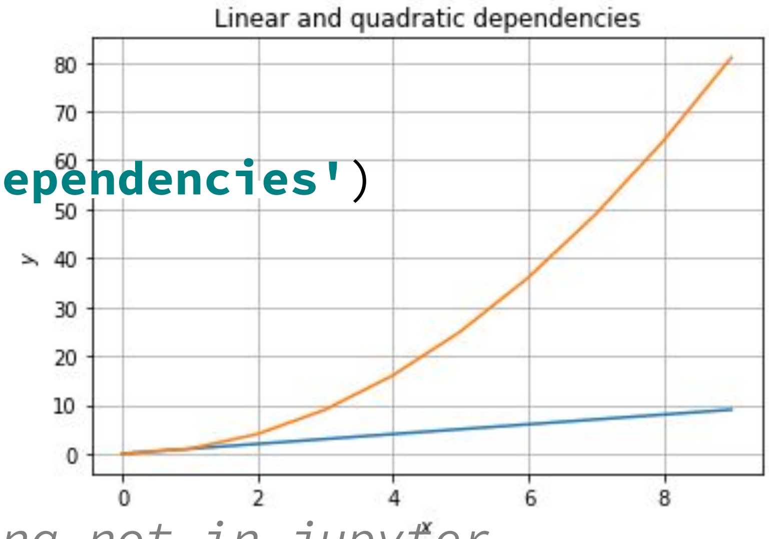
Save image

```
plt.plot(xs, ys)

# Customization
plt.title('Oh my function')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.grid(True)

plt.savefig('path_to_save.png')
# Add plt.show() if you are working not in jupyter
```



Several plots

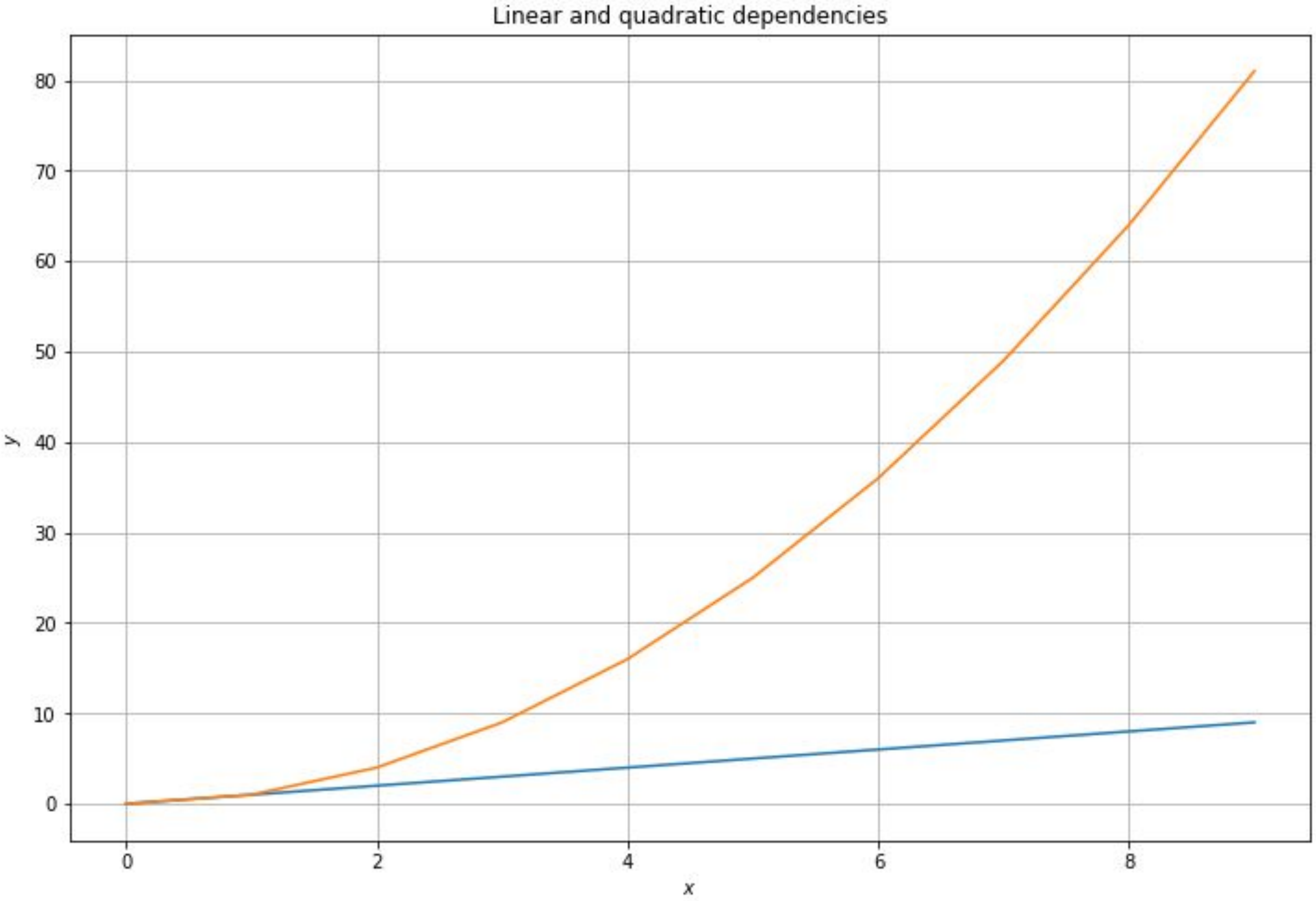
```
ys2 = [y ** 2 for y in ys]

plt.plot(xs, ys)
plt.plot(xs, ys2)

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.grid(True)

plt.savefig('path_to_save.png')
# Add plt.show() if you are working not in jupyter
```



More customization

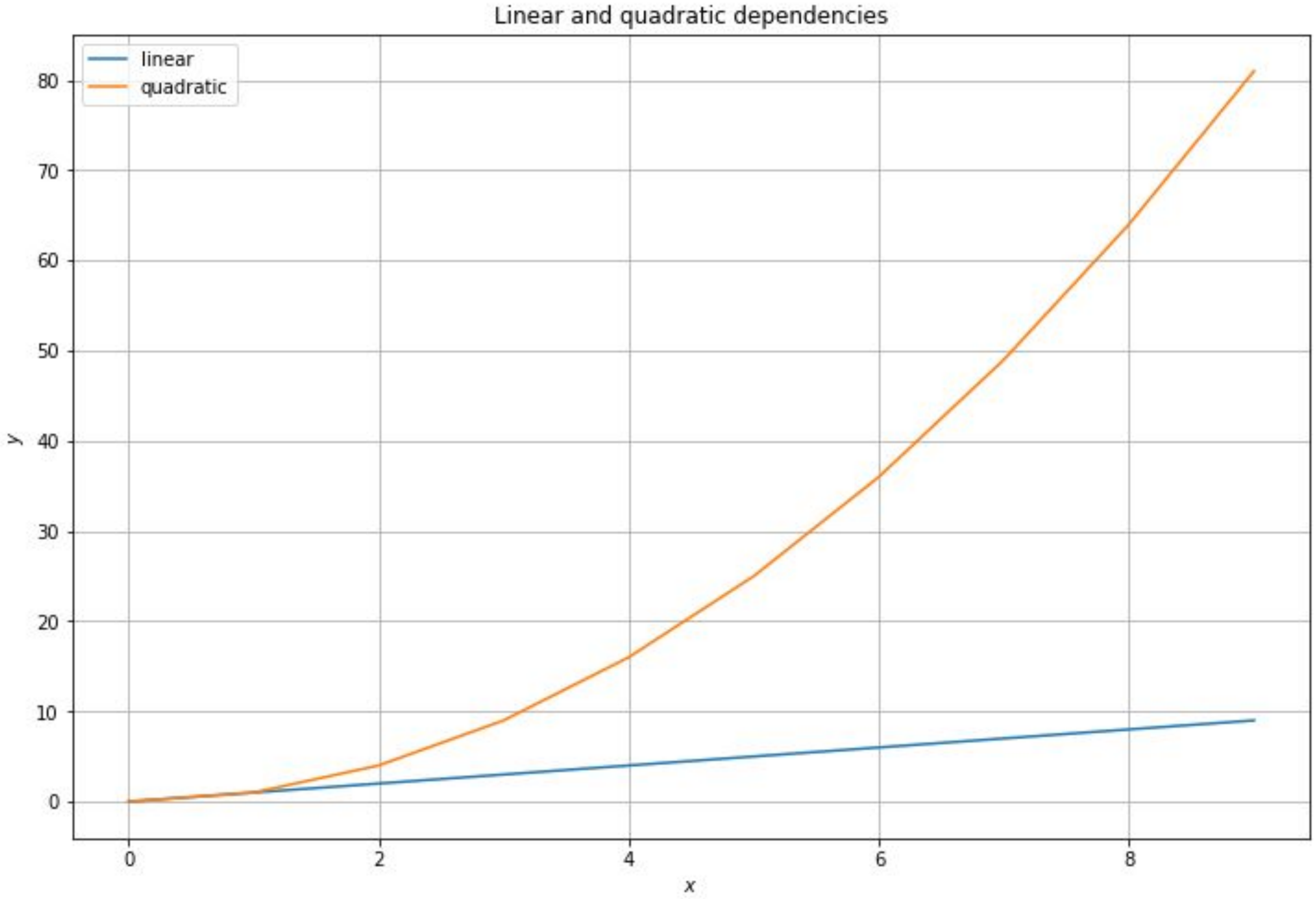
Image size

```
plt.figure(figsize=(12, 8))

plt.plot(xs, ys)
plt.plot(xs, ys2)

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.grid(True)

plt.savefig('path_to_save.png')
# Add plt.show() if you are working not in jupyter
```


Linear and quadratic dependencies
y
0
10
20
30
40
50
60
70
80
0
2
4
6
8
x

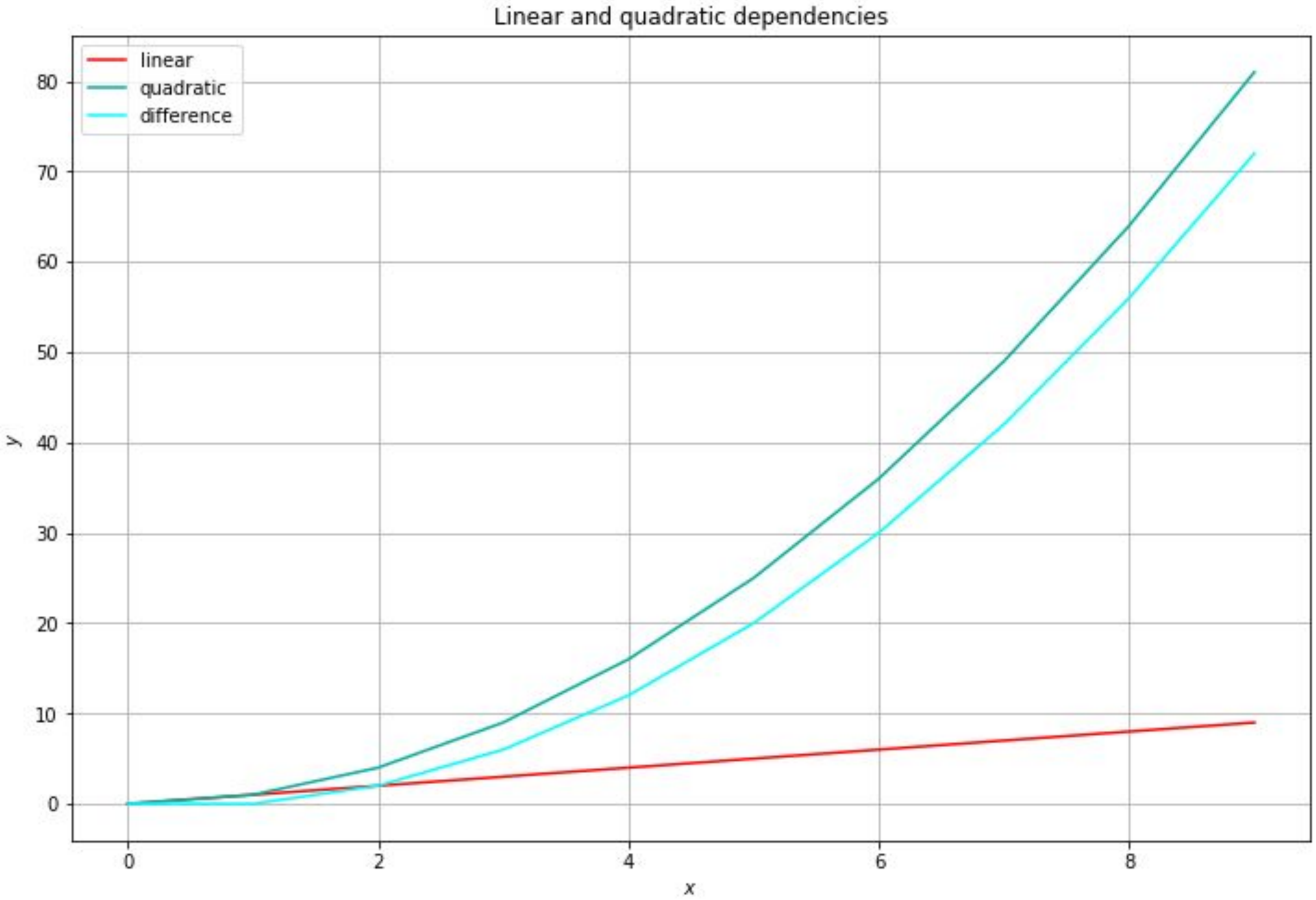
Legend

```python
plt.figure(figsize=(12, 8))

plt.plot(xs, ys, label='linear')
plt.plot(xs, ys2, label='quadratic')

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.legend()
plt.grid(True)

plt.savefig('path_to_save.png')
# Add plt.show() if you are working not in jupyter
```


Linear and quadratic dependencies
linear
quadratic
80
70
60
50
40
30
20
10
0
y
0
2
4
6
8
x

Colors

```
ys3 = [yq - yl for yq, yl in zip(ys2, ys)]

plt.figure(figsize=(12, 8))

plt.plot(xs, ys, color='r', label='linear')
plt.plot(xs, ys2, color='#00AE9F', label='quadratic')
plt.plot(xs, ys3, color='cyan', label='difference')

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.legend()
plt.grid(True)
```


Linear and quadratic dependencies
linear
quadratic
difference
80
70
60
50
40
30
20
10
0
y
0
2
4
6
8
x

Options to save image

```
plt.figure(figsize=(12, 8))

plt.plot(xs, ys, color='r', label='linear')
plt.plot(xs, ys2, color='#00AE9F', label='quadratic')
plt.plot(xs, ys3, color='cyan', label='difference')

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.legend()
plt.grid(True)

plt.savefig('path_to_save.png', bb_inches='tight',
            format='png')
```

Why is it so fucking long?!

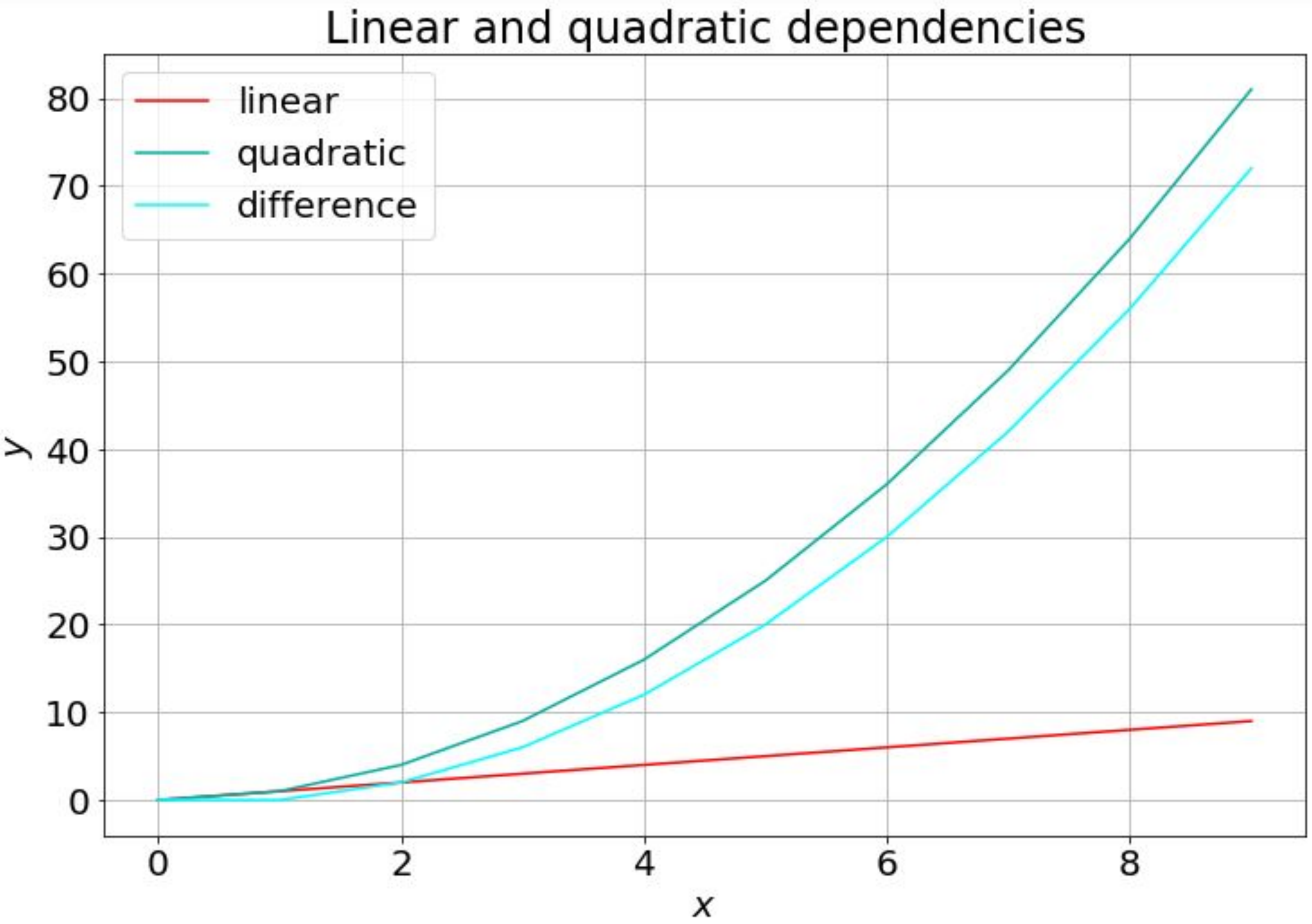
Basic library - have really big number of different parameters, but most powerful

There is a way to specify default parameters for your script - look at plt.rcParams and in plt documentation in your free time for the full list of options

```
plt.rcParams['figure.figsize']
```

Colors

```python
plt.rcParams['figure.figsize'] = [12, 8]
plt.rcParams['font.size'] = 20
plt.rcParams['savefig.bbox'] = 'tight'


plt.plot(xs, ys, color='r', label='linear')
plt.plot(xs, ys2, color='#00AE9F', label='quadratic')
plt.plot(xs, ys3, color='cyan', label='difference')

plt.title('Linear and quadratic dependencies')
plt.xlabel('$x$')
plt.ylabel('$y$')
plt.legend()
plt.grid(True)
```

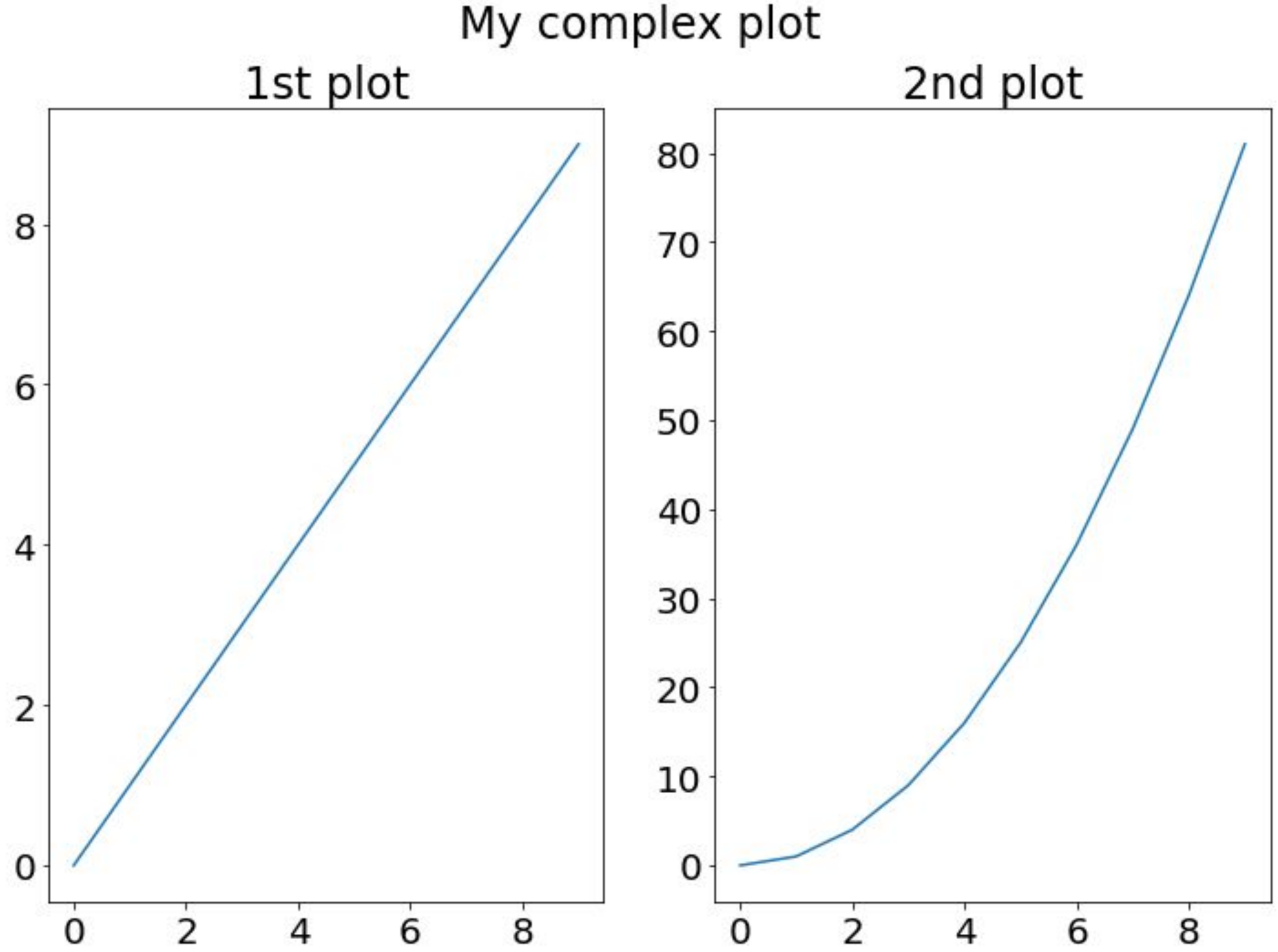

Linear and quadratic dependencies
linear
quadratic
difference
y
x
0
10
20
30
40
50
60
70
80
0
2
4
6
8

Several plots in one image

Pretty useful to have graphs near each other in one image. How to do it?
Subplots!

Colors

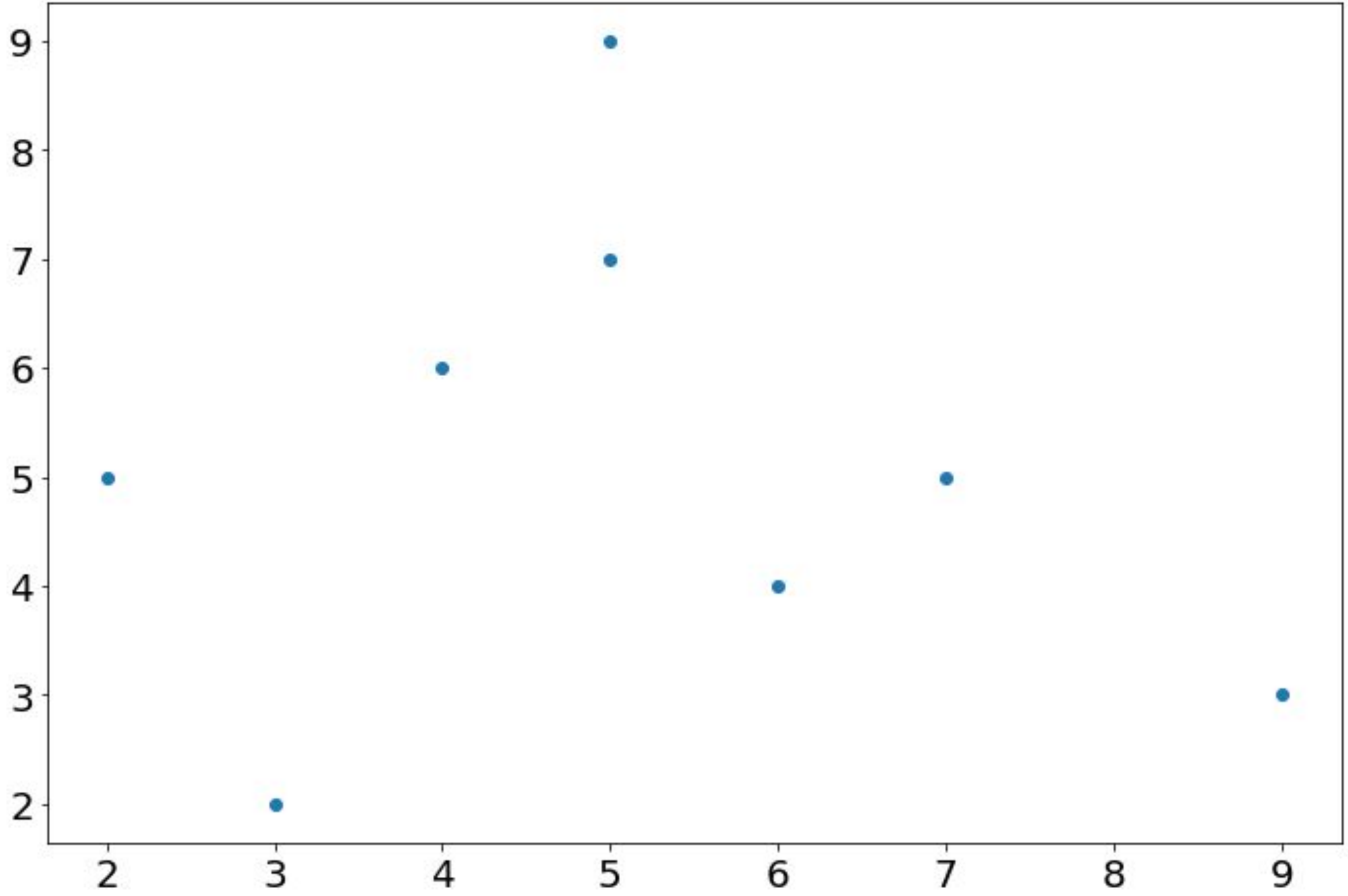
```
# axes is an iterable with small plot on big image
figure, axes = plt.subplots(nrows=1, ncols=2)

# Next you could use each axes as plt
axes[0].plot(xs, ys)
axes[1].plot(xs, ys2)

# Customization functions are slightly different -
set_attribute
axes[0].set_title('1st plot')
axes[1].set_title('2nd plot')

# Title for the whole picture
plt.suptitle('My complex plot')
```


My complex plot
1st plot
0
2
4
6
8
0
2
4
6
8
2nd plot
0
10
20
30
40
50
60
70
80
0
2
4
6
8

Other types of plots

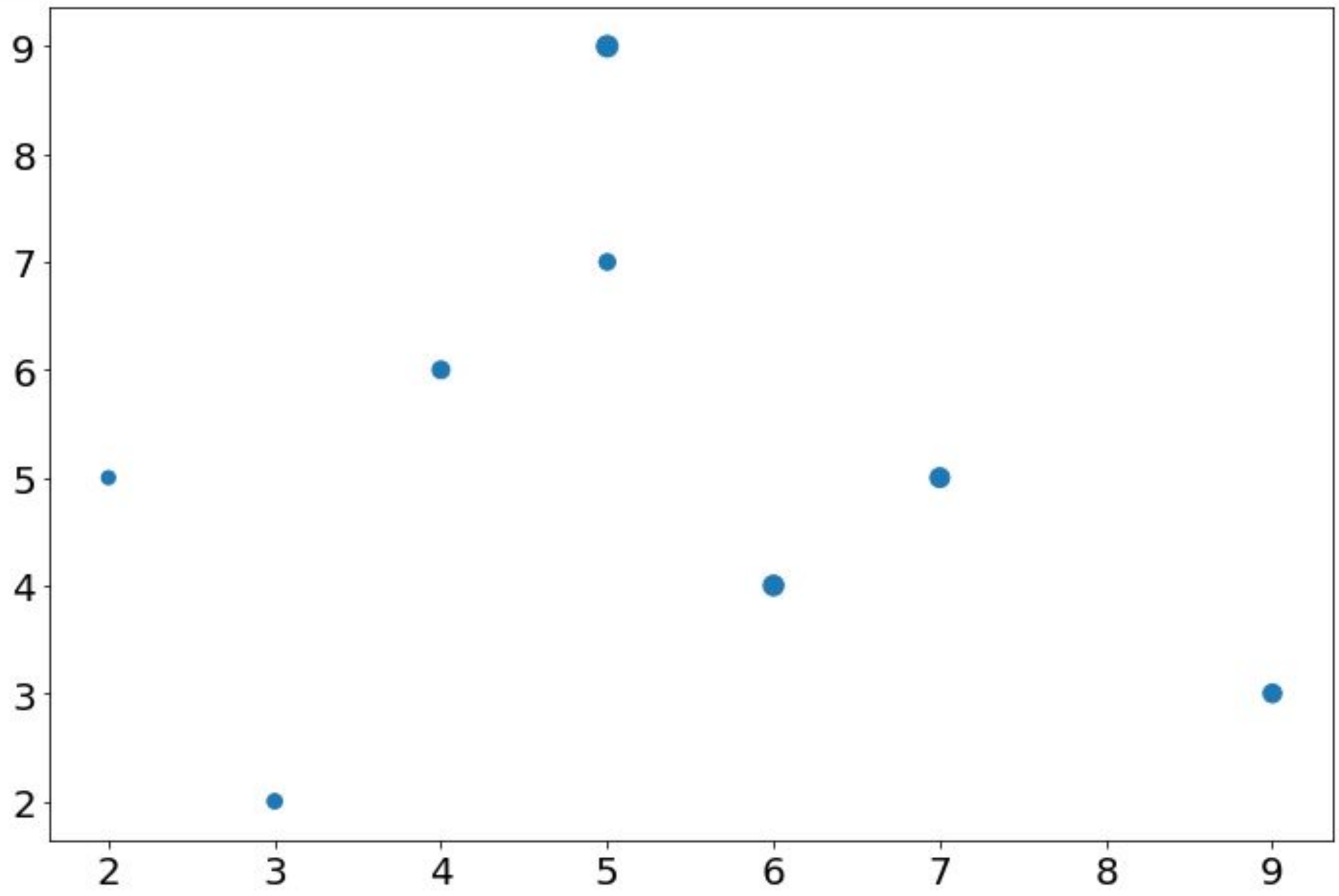
Scatterplot

```
# Prepare data
xs = [2, 3, 5, 4, 9, 7, 6, 5]
ys = [5, 2, 7, 6, 3, 5, 4, 9]

# Scatter plot
plt.scatter(xs, ys)
```


9
8
7
6
5
4
3
2
2
3
4
5
6
7
8
9

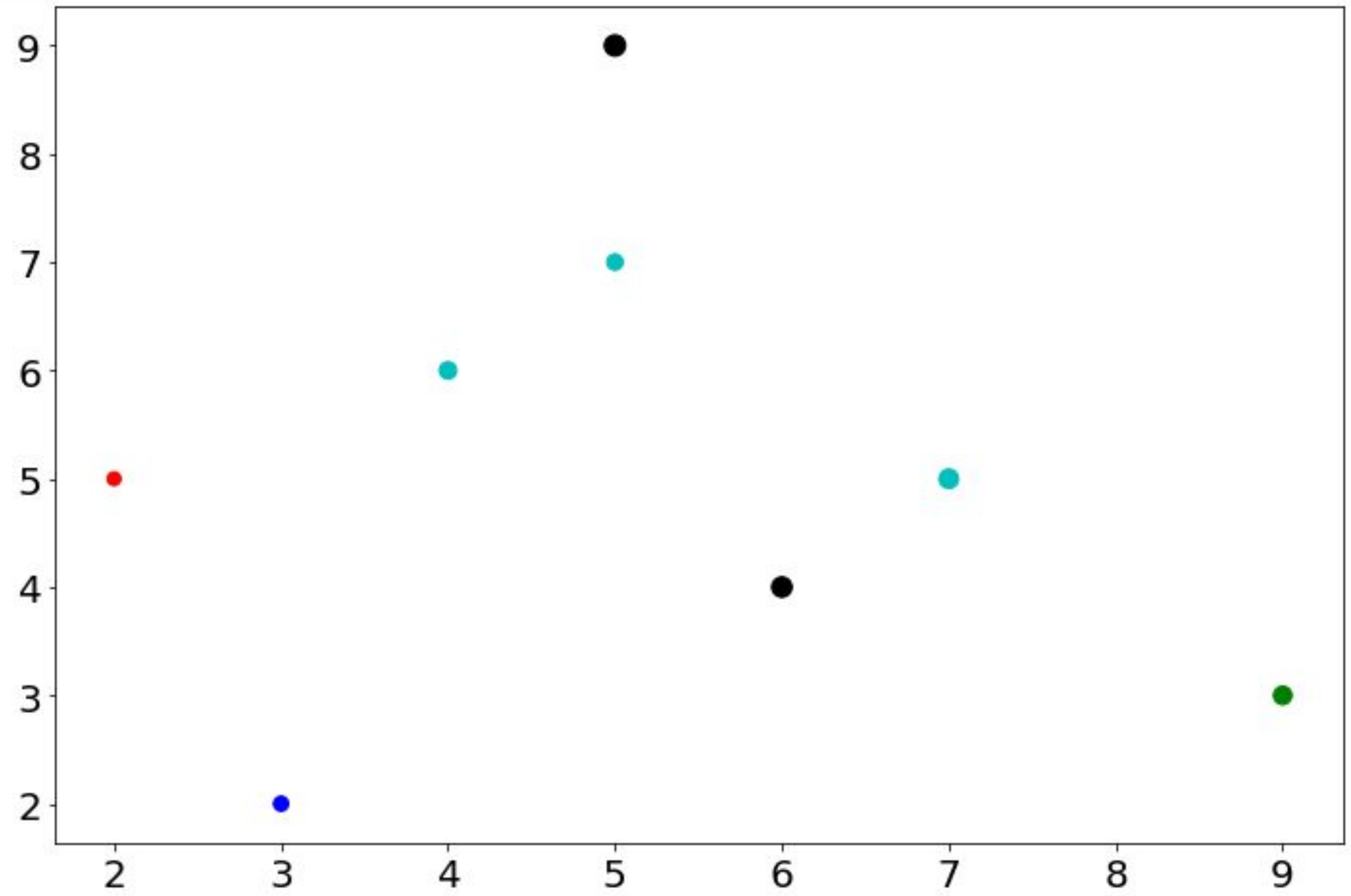
Little customization of scatterplot

```
# Create sizes for points in scatterplot
sizes = range(50, 150, 10)

plt.scatter(xs, ys, s=sizes)
```


9
8
7
6
5
4
3
2
2
3
4
5
6
7
8
9

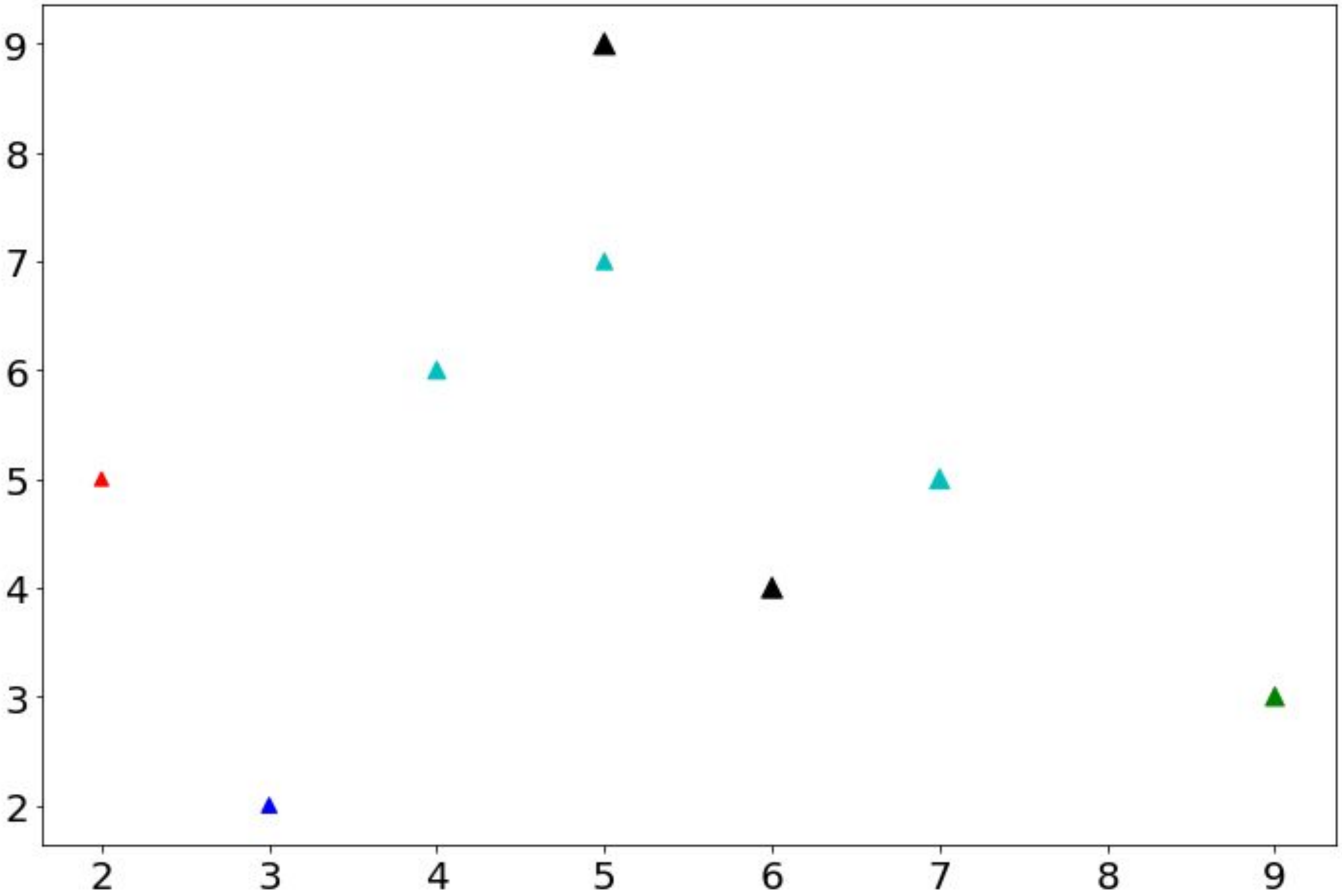
Colors

```
# Colors for scatterplot points
colors = ['r', 'b', 'c', 'c', 'g', 'c', 'k', 'k',
          'k', 'b']

plt.scatter(xs, ys, s=sizes, c=colors)
```


9
8
7
6
5
4
3
2
2
3
4
5
6
7
8
9

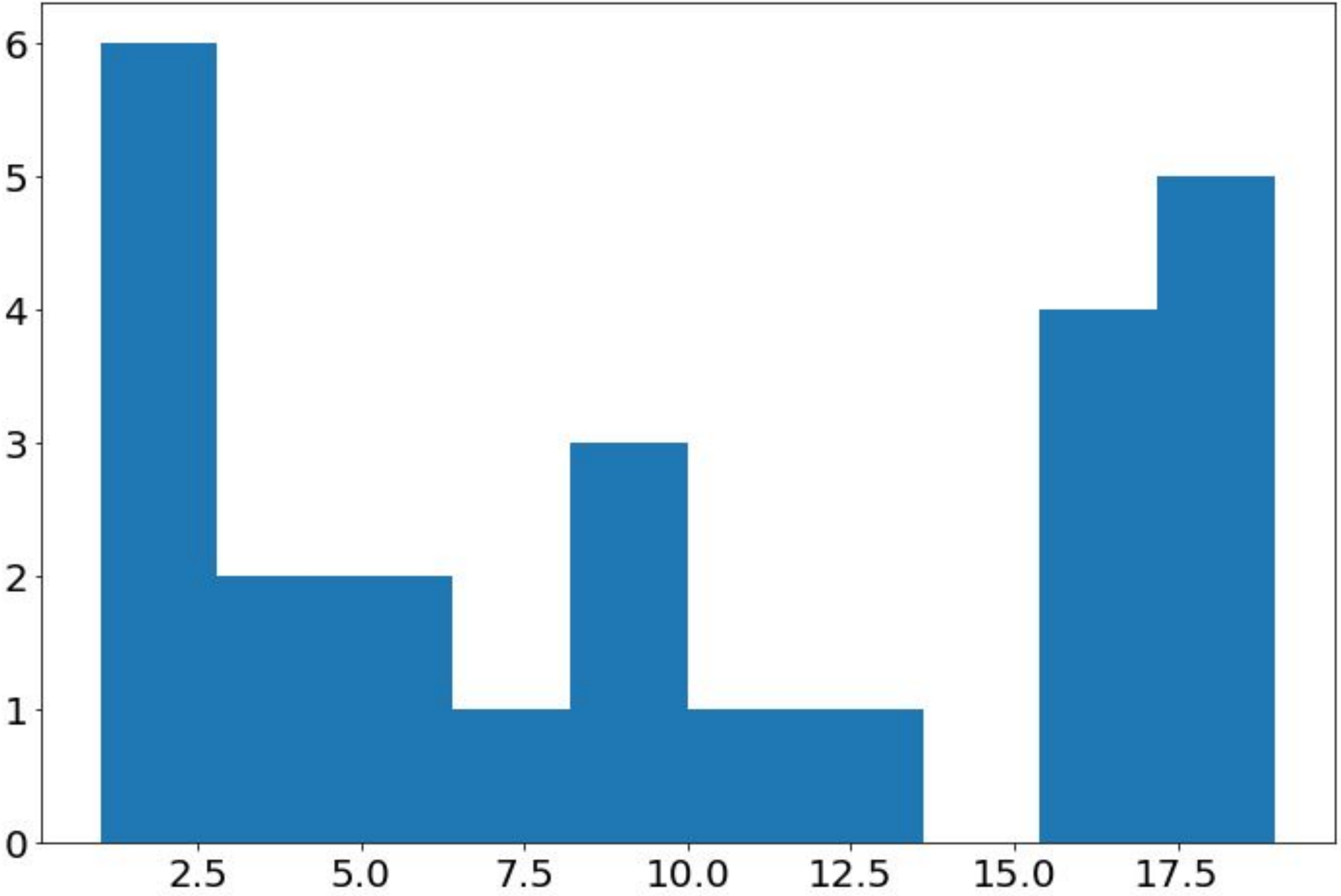
Marker shape

```
plt.scatter(xs, ys, s=sizes, c=colors, marker='^')
```


9
8
7
6
5
4
3
2
2
3
4
5
6
7
8
9

Histogram

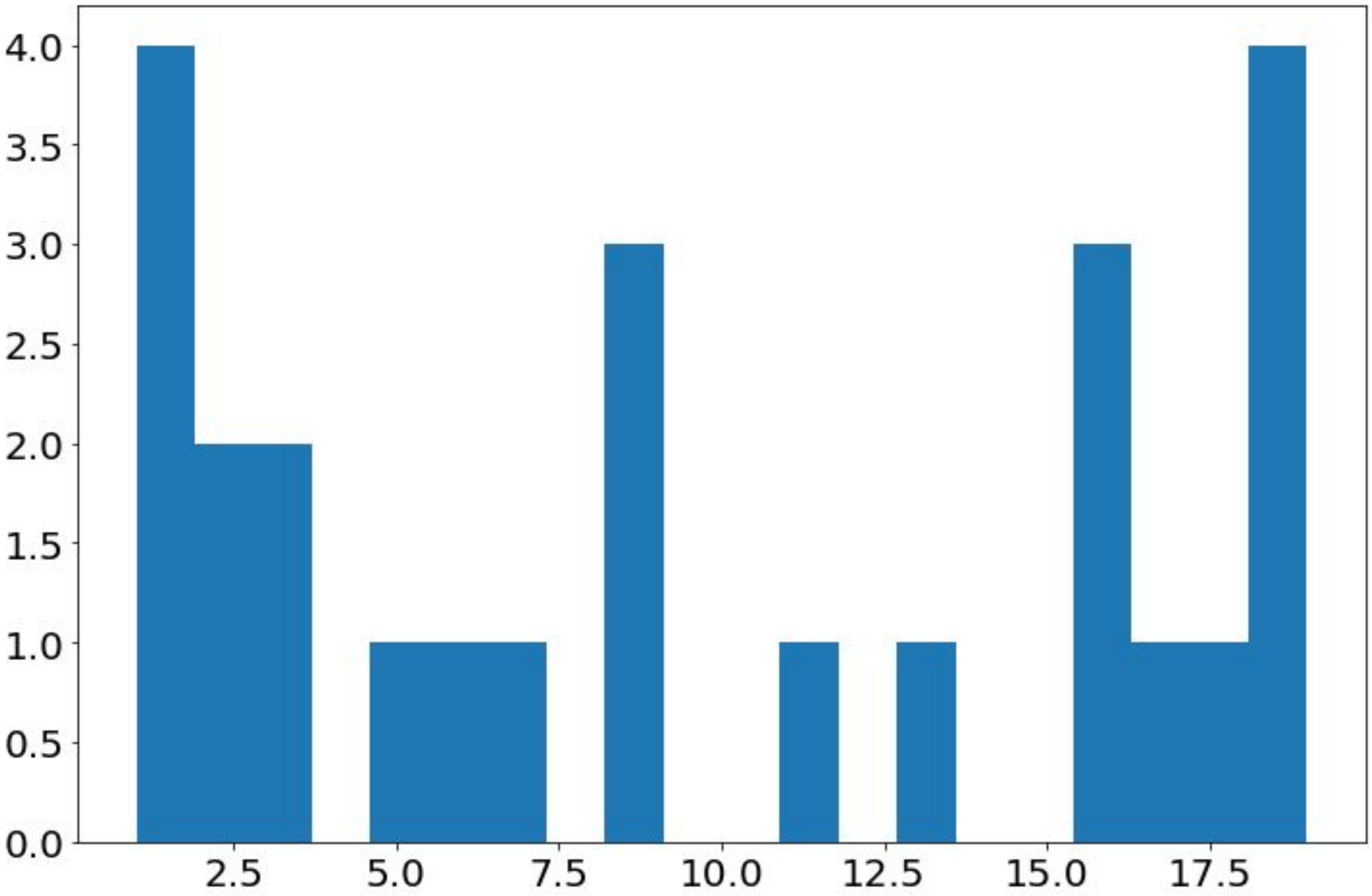
```
xs = [1, 2, 3, 1, 1, 2, 3, 1, 5, 6, 7, 9, 9, 9, 11,
      13, 16, 16, 16, 17, 18, 19, 19, 19, 19]
plt.hist(xs)
```


6
5
4
3
2
1
0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

Customizations of histogram

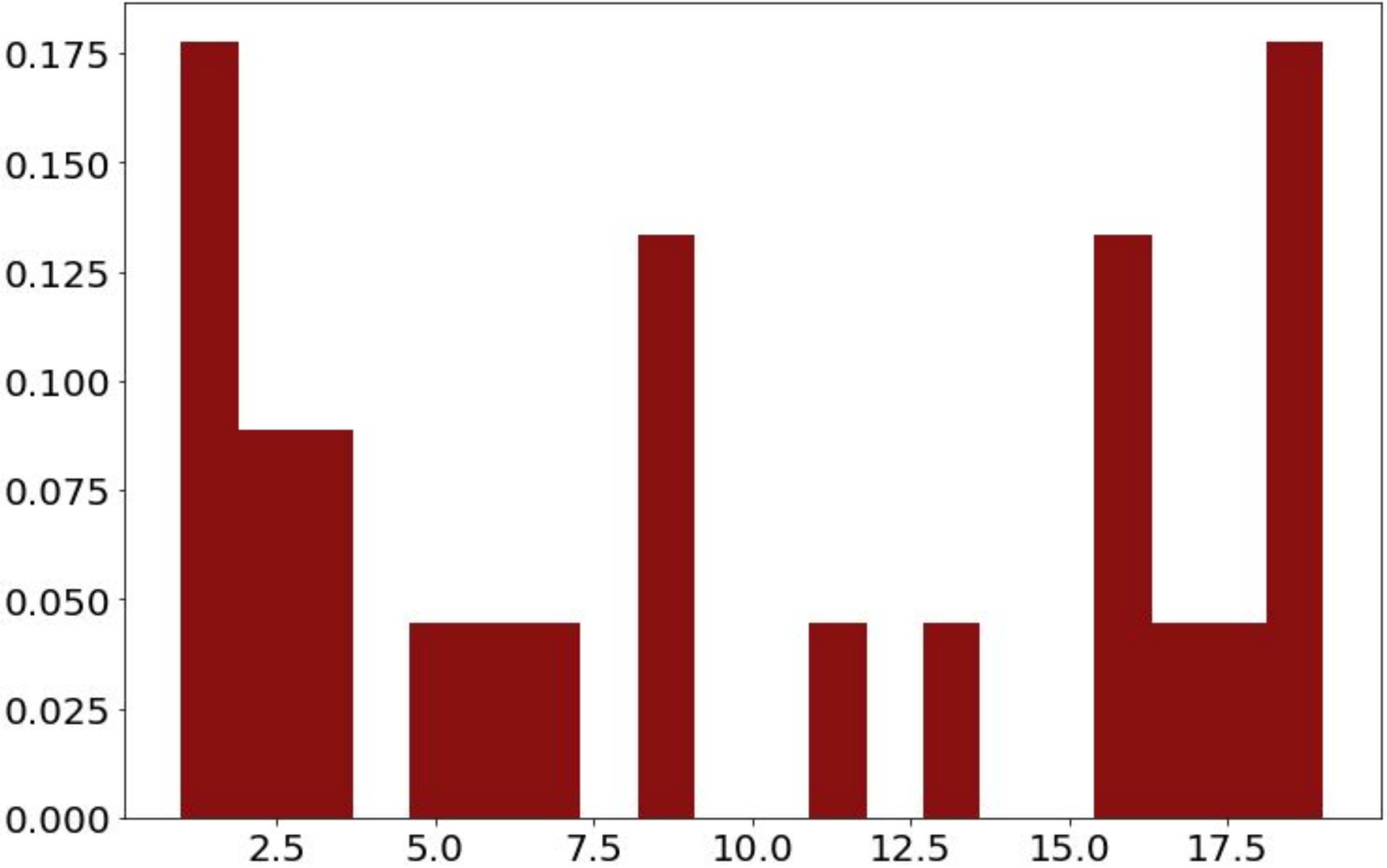
Number of bins

```
plt.hist(xs, bins=20)
```


0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

Normalization

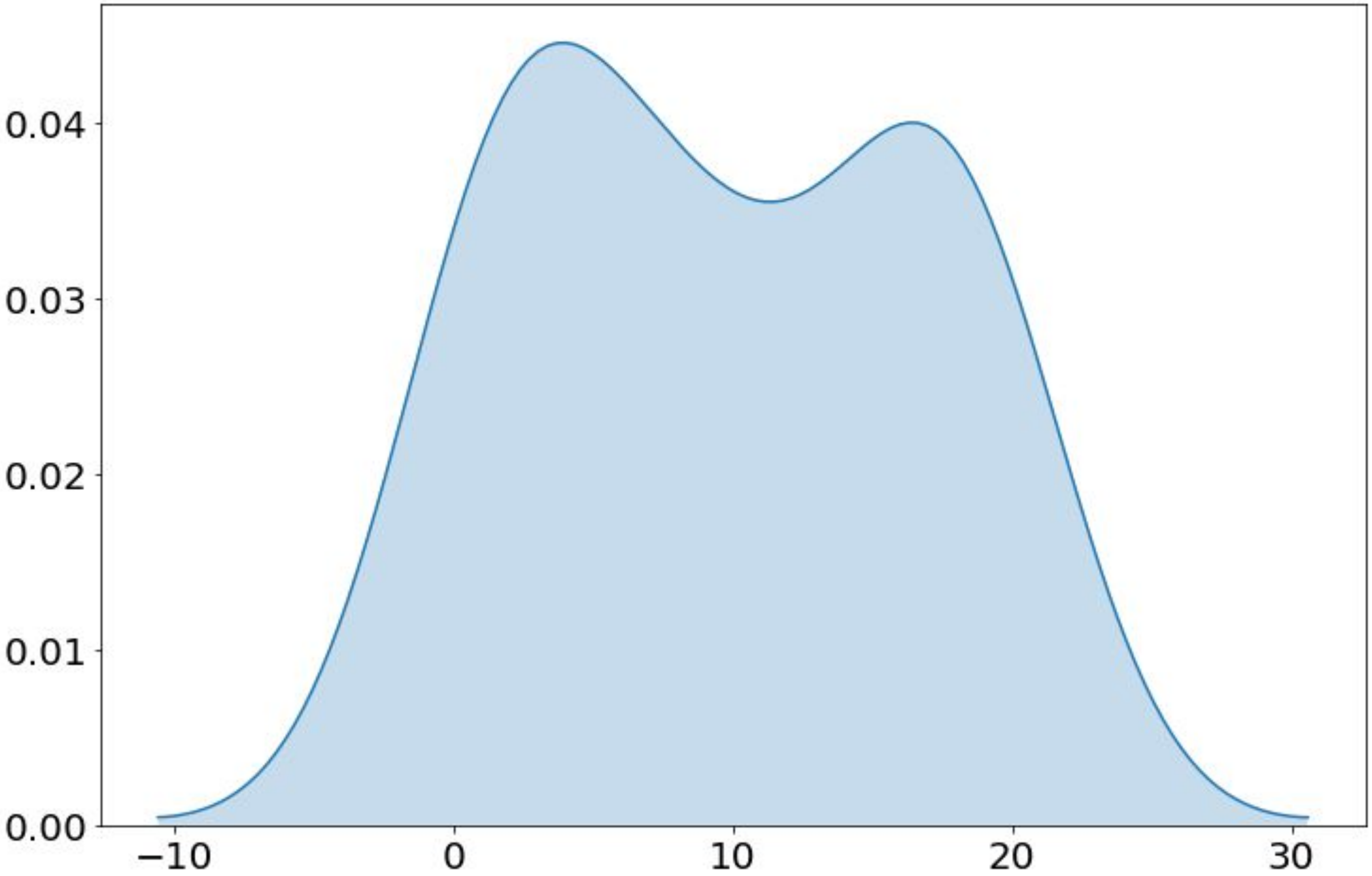
```
plt.hist(xs, bins=20, color='#891010', density=True)
```


0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
2.5
5.0
7.5
10.0
12.5
15.0
17.5

Well histograms in plt looks pretty awful, thus let's move to seaborn ...
SEABORN!!!!!

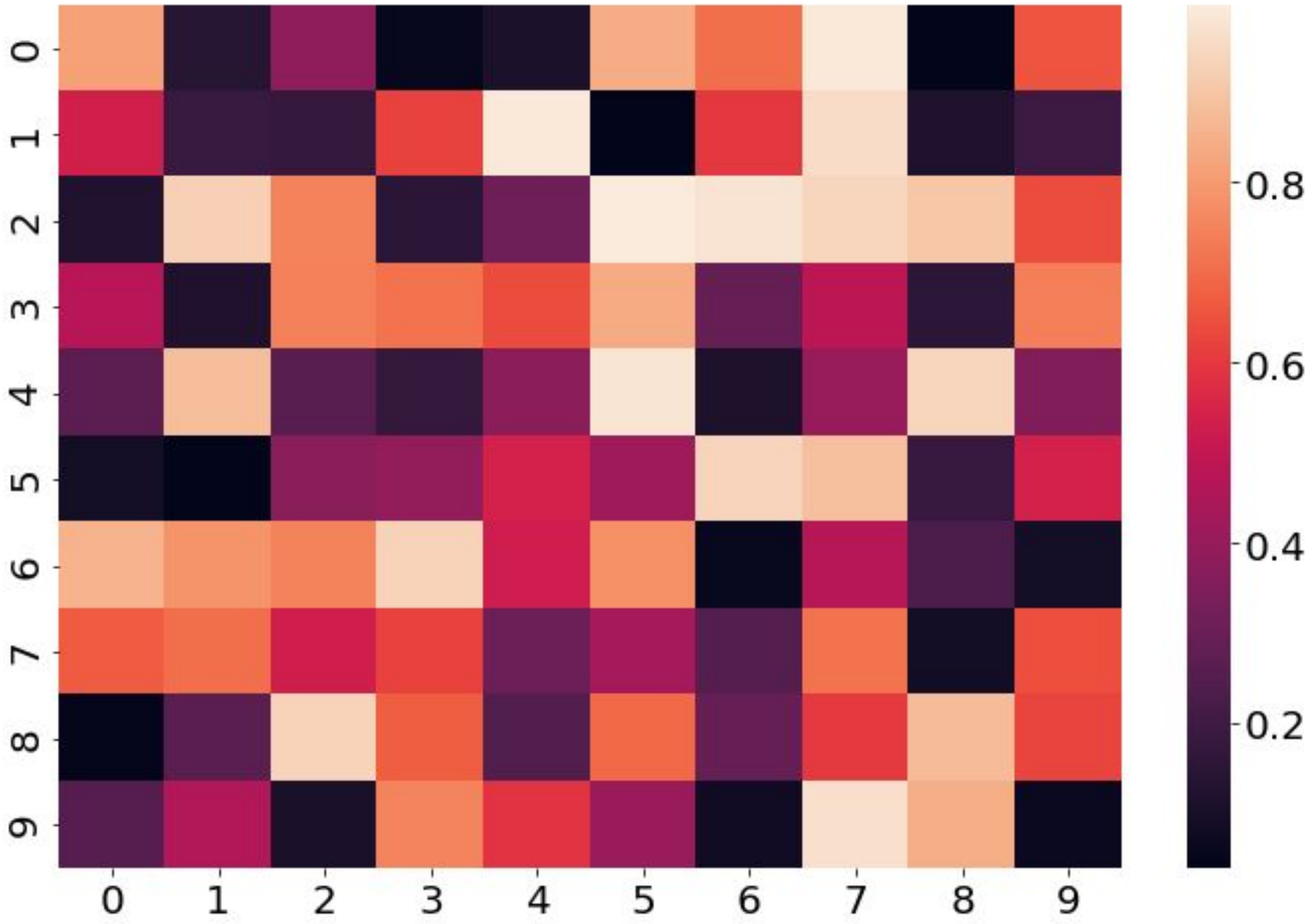
Kernel Density Estimation plot

```
import seaborn as sns


sns.kdeplot(xs, shade=True)
```


0.04
0.03
0.02
0.01
0.00
−10
0
10
20
30

Heatmap

```
# I've predefined data as a 2d matrix

sns.heatmap(data)
```

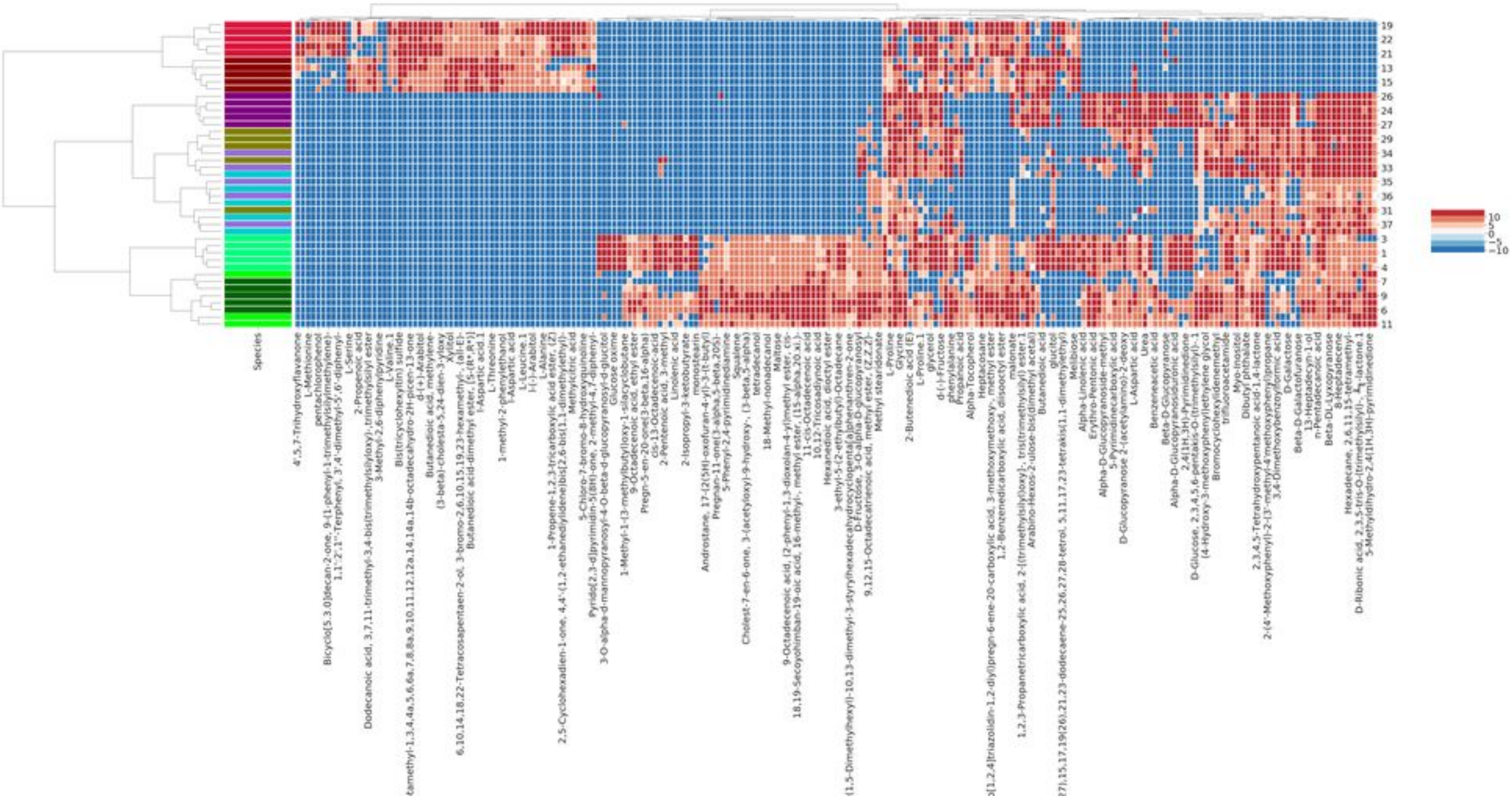

0
1
2
3
4
5
6
7
8
9
0
1
2
3
4
5
6
7
8
9
0.8
0.6
0.4
0.2

And a customized clustermap (heatmap + clustering) for dessert

```
import pandas as pd
from matplotlib.colors import ListedColormap


# Read data
matrix = pd.read_csv('path_to.csv')

# Log transformation
log_data =  np.log(
    matrix.select_dtypes(include=[np.number]) +
    0.00001)
log_data['Species'] = matrix.loc[:, 'Species']
```

```
# Color of cells
palette = ListedColormap(sns.color_palette("RdBu_r", 7))

# Prepare colored index for heatmap
colors = '#00FF7F', '#006400', '#00FF00', '#8B0000',
 '#DC143C', '#800080', '#808000', '#9370DB', '#00CED1'

correspondance = dict(
zip(matrix.loc[:, 'Species'].unique(), colors))
row_colors = matrix.loc[:, 'Species'].map(correspondance)
```

```
a = sns.clustermap(log_data.drop('Species', axis=1),
figsize=(50, 12), row_colors=row_colors, robust=True,
linewidths=0.7, annot_kws={'name': 'Arial'}, cmap=palette,
method='ward')

# Rotate row labels
plt.setp(a.ax_heatmap.yaxis.get_majorticklabels(),
rotation=0)


# Place for legend
a.cax.set_position((0.93, 0.3, 0.03, 0.1))


# Font settings
sns.set(font='sans-serif', font_scale=1)
```


Species
19
22
21
13
15
26
24
27
29
34
33
35
36
31
37
3
1
4
7
9
6
11
10
5
0
−5
−10
4',5,7-Trihydroxyflavanone
L-Methionine
pentachlorophenol
Bicyclo[5.3.0]decan-2-one, 9-(1-phenyl(-1-trimethylsilylmethylene)-
1,1':2',1''-Terphenyl, 3',4'-dimethyl-5',6'-diphenyl-
L-Serine
2-Propenoic acid
Dodecanoic acid, 3,7,11-trimethyl-3,4-bis(trimethylsilyloxy)-, trimethylsilyl ester
3-Methyl-2,6-diphenylpyridine
L-Valine.1
tamethyl-1,3,4,4a,5,6,6a,7,8,8a,9,10,11,12,12a,14,14a,14b-octadecahydro-2H-picen-13-one
Bis(tricyclohexyltin) sulfide
d-(+)-Arabitol
Butanedioic acid, methylene-
(3-beta)-cholesta-5,24-dien-3-yloxy
6,10,14,18,22-Tetracosapentaen-2-ol, 3-bromo-2,6,10,15,19,23-hexamethyl-, (all-E)-
Xylitol
Butanedioic acid-dimethyl ester, [S-(R*,R*)]
l-Aspartic acid.1
L-Threonine
1-methyl-2-phenylethanol
l-Aspartic acid
L-Leucine.1
l-(-)-Arabitol
L-Alanine
1-Propene-1,2,3-tricarboxylic acid ester, (Z)
2,5-Cyclohexadien-1-one, 4,4'-(1,2-ethanediylidene)bis[2,6-bis(1,1-dimethylethyl)-
Methylcitric acid
5-Chloro-7-bromo-8-hydroxyquinoline
Pyrido[2,3-d]pyrimidin-5(8H)-one, 2-methyl-4,7-diphenyl-
3-O-alpha-d-mannopyranosyl-4-O-beta-d-glucopyranosyl-d-glucitol
Glucose oxime
1-Methyl-1-(3-methylbutyloxy-1-silacyclobutane
9-Octadecenoic acid, ethyl ester
Pregn-5-en-20-one(3-beta,16-alpha)
cis-13-Octadecenoic acid
2-Pentenoic acid, 3-methyl
Linoleic acid
2-Isopropyl-3-ketobutyrate
monostearin
Androstane, 17-(2(5H)-oxofuran-4-yl)-3-(t-butyl)
Pregnan-11-one(3-alpha,5-beta,20S)-
5-Phenyl-2,4-pyrimidinediamine
Squalene
Cholest-7-en-6-one, 3-(acetyloxy)-9-hydroxy-, (3-beta,5-alpha)
tetradecanol
18-Methyl-nonadecanol
Maltose
9-Octadecenoic acid, (2-phenyl-1,3-dioxolan-4-yl)methyl ester, cis-
18,19-Secoyohimban-19-oic acid, 16-methyl-, methyl ester, (15-alpha,20-xi)-
11-cis-Octadecenoic acid
10,12-Tricosadiynoic acid
Hexanedioic acid, dioctyl ester
3-ethyl-5-(2-ethylbutyl)-Octadecane
(1,5-Dimethylhexyl)-10,13-dimethyl-3-styrylhexadecahydrocyclopenta[a]phenanthren-2-one
D-Fructose, 3-O-alpha-D-glucopyranosyl
9,12,15-Octadecatrienoic acid, methyl ester, (Z,Z,Z)-
Methyl stearidonate
L-Proline
Glycine
2-Butenedioic acid (E)
L-Proline.1
glycerol
d-(-)-Fructose
phenylalanine
Propanoic acid
Alpha-Tocopherol
Heptacosane
[1,2,4]triazolidin-1,2-diyl)pregn-6-ene-20-carboxylic acid, 3-methoxymethoxy-, methyl ester
1,2-Benzenedicarboxylic acid, diisooctyl ester
malonate
1,2,3-Propanetricarboxylic acid, 2-[(trimethylsilyl)oxy]-, tris(trimethylsilyl) ester.1
Arabino-Hexos-2-ulose-bis(dimethyl acetal)
Butanedioic acid
glucitol
27),15,17,19(26),21,23-dodecaene-25,26,27,28-tetrol, 5,11,17,23-tetrakis(1,1-dimethylethyl)-
Melibiose
Alpha-Linolenic acid
Erythro-Pentonic acid
Alpha-D-Glucopyranoside-methyl
5-Pyrimidinecarboxylic acid
D-Glucopyranose 2-(acetylamino)-2-deoxy
L-Aspartic acid
Urea
Benzeneacetic acid
Beta-D-Glucopyranose
Alpha-D-Glucopyranosiduronic acid
2,4(1H,3H)-Pyrimidinedione
D-Glucose, 2,3,4,5,6-pentakis-O-(trimethylsilyl)-.1
(4-Hydroxy-3-methoxyphenyl)ethylene glycol
Bromocyclohexylidenemethyl
trifluoroacetamide
Myo-Inositol
Dibutyl phthalate
2,3,4,5-Tetrahydroxypentanoic acid-1,4-lactone
2-(4'-Methoxyphenyl)-2-(3'-methyl-4'methoxyphenyl)propane
3,4-Dimethoxybenzoylformic acid
D-Galactose
Beta-D-Galactofuranose
13-Heptadecyn-1-ol
n-Pentadecanoic acid
Beta-DL-Lyxopyranose
8-Heptadecene
Hexadecane, 2,6,11,15-tetramethyl-
D-Ribonic acid, 2,3,5-tris-O-(trimethylsilyl)-, γ-lactone.1
5-Methyldihydro-2,4(1H,3H)-pyrimidinedione