LinkedIn LEARNING

Certificate of Completion

Congratulations, Aadarsha Chapagain

Python for Data Science Essential Training Part 1

Course completed on Mar 02, 2022 at 08:29PM UTC • 6 hours 2 min

By continuing to learn, you have expanded your perspective, sharpened your skills, and made yourself even more in demand.

Head of Content Strategy, Learning

LinkedIn Learning
1000 W Maude Ave
Sunnyvale, CA 94085

Certificate Id: AT1OiICt88Mtv8NbY1ZTprdCbQHP

Chapter2

March 1, 2022

1 Chapter 2 - Data Preparation Basics

1.1 Segment 1 - Filtering and selecting data

[1]:
```
import numpy as np
import pandas as pd

from pandas import Series, DataFrame
```

1.1.1 Selecting and retrieving data

You can write an index value in two forms. - Label index or - Integer index

[2]:
```
series_obj = Series(np.arange(8), index=['row 1', 'row 2', 'row 3', 'row 4',␣
↪'row 5', 'row 6','row 7', 'row 8'])
series_obj
```

[2]:
```
row 1    0
row 2    1
row 3    2
row 4    3
row 5    4
row 6    5
row 7    6
row 8    7
dtype: int32
```

[3]:
```
series_obj['row 7']
```

[3]:
```
6
```

[4]:
```
series_obj[[0, 7]]
```

[4]:
```
row 1    0
row 8    7
dtype: int32
```

[5]:
```
np.random.seed(25)
DF_obj = DataFrame(np.random.rand(36).reshape((6,6)),
                   index=['row 1', 'row 2', 'row 3', 'row 4','row 5','row 6'],
                   columns=['column 1','column 2','column 3','column 4','column␣
 ↪5','column 6'])
DF_obj
```

[5]:

	column 1	column 2	column 3	column 4	column 5	column 6
row 1	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
row 2	0.684969	0.437611	0.556229	0.367080	0.402366	0.113041
row 3	0.447031	0.585445	0.161985	0.520719	0.326051	0.699186
row 4	0.366395	0.836375	0.481343	0.516502	0.383048	0.997541
row 5	0.514244	0.559053	0.034450	0.719930	0.421004	0.436935
row 6	0.281701	0.900274	0.669612	0.456069	0.289804	0.525819

[6]:
```
DF_obj.loc[['row 2', 'row 5'], ['column 5', 'column 2']]
```

[6]:

	column 5	column 2
row 2	0.402366	0.437611
row 5	0.421004	0.559053

1.1.2 Data slicing

You can use slicing to select and return a slice of several values from a data set. Slicing uses index values so you can use the same square brackets when doing data slicing.

How slicing differs, however, is that with slicing you pass in two index values that are separated by a colon. The index value on the left side of the colon should be the first value you want to select. On the right side of the colon, you write the index value for the last value you want to retrieve. When you execute the code, the indexer then simply finds the first record and the last record and returns every record in between them.

[7]:
```
series_obj['row 3': 'row 7']
```

[7]:
```
row 3    2
row 4    3
row 5    4
row 6    5
row 7    6
dtype: int32
```

1.1.3 Comparing with scalars

Now we're going to talk about comparison operators and scalar values. Just in case you don't know that a scalar value is, it's basically just a single numerical value. You can use comparison operators like greater than or less than to return true/false values for all records to indicate how each element compares to a scalar value.

[8]:
```
DF_obj < .2
```

[8]:

	column 1	column 2	column 3	column 4	column 5	column 6
row 1	False	False	False	True	False	True
row 2	False	False	False	False	False	True
row 3	False	False	True	False	False	False
row 4	False	False	False	False	False	False
row 5	False	False	True	False	False	False
row 6	False	False	False	False	False	False

1.1.4 Filtering with scalars

[9]:
```
series_obj[series_obj > 6]
```

[9]:
```
row 8    7
dtype: int32
```

1.1.5 Setting values with scalars

[10]:
```
series_obj['row 1', 'row 5', 'row 8'] = 8
series_obj
```

[10]:
```
row 1    8
row 2    1
row 3    2
row 4    3
row 5    8
row 6    5
row 7    6
row 8    8
dtype: int32
```

Filtering and selecting using Pandas is one of the most fundamental things you'll do in data analysis. Make sure you know how to use indexing to select and retrieve records.

2 Chapter 2 - Data Preparation Basics

2.1 Segment 2 - Treating missing values

[11]:
```
import numpy as np
import pandas as pd

from pandas import Series, DataFrame
```

2.1.1 Figuring out what data is missing

[12]:
```
missing = np.nan
```

```
series_obj = Series(['row 1', 'row 2', missing, 'row 4', 'row 5', 'row 6',␣
 ↪missing, 'row 8'])
series_obj
```

[12]:
```
0    row 1
1    row 2
2      NaN
3    row 4
4    row 5
5    row 6
6      NaN
7    row 8
dtype: object
```

[13]:
```
series_obj.isnull()
```

[13]:
```
0    False
1    False
2     True
3    False
4    False
5    False
6     True
7    False
dtype: bool
```

2.1.2 Filling in for missing values

[14]:
```
np.random.seed(25)
DF_obj = DataFrame(np.random.rand(36).reshape(6,6))
DF_obj
```

[14]:

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	0.113041
2	0.447031	0.585445	0.161985	0.520719	0.326051	0.699186
3	0.366395	0.836375	0.481343	0.516502	0.383048	0.997541
4	0.514244	0.559053	0.034450	0.719930	0.421004	0.436935
5	0.281701	0.900274	0.669612	0.456069	0.289804	0.525819

[15]:
```
DF_obj.loc[3:5, 0] = missing
DF_obj.loc[1:4, 5] = missing
DF_obj
```

[15]:

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	NaN

```
2  0.447031  0.585445  0.161985  0.520719  0.326051       NaN
3       NaN  0.836375  0.481343  0.516502  0.383048       NaN
4       NaN  0.559053  0.034450  0.719930  0.421004       NaN
5       NaN  0.900274  0.669612  0.456069  0.289804  0.525819
```

[16]:
```
filled_DF = DF_obj.fillna(0)
filled_DF
```

[16]:
```
          0         1         2         3         4         5
0  0.870124  0.582277  0.278839  0.185911  0.411100  0.117376
1  0.684969  0.437611  0.556229  0.367080  0.402366  0.000000
2  0.447031  0.585445  0.161985  0.520719  0.326051  0.000000
3  0.000000  0.836375  0.481343  0.516502  0.383048  0.000000
4  0.000000  0.559053  0.034450  0.719930  0.421004  0.000000
5  0.000000  0.900274  0.669612  0.456069  0.289804  0.525819
```

[17]:
```
filled_DF = DF_obj.fillna({0: 0.1, 5:1.25})
filled_DF
```

[17]:
```
          0         1         2         3         4         5
0  0.870124  0.582277  0.278839  0.185911  0.411100  0.117376
1  0.684969  0.437611  0.556229  0.367080  0.402366  1.250000
2  0.447031  0.585445  0.161985  0.520719  0.326051  1.250000
3  0.100000  0.836375  0.481343  0.516502  0.383048  1.250000
4  0.100000  0.559053  0.034450  0.719930  0.421004  1.250000
5  0.100000  0.900274  0.669612  0.456069  0.289804  0.525819
```

[18]:
```
fill_DF = DF_obj.fillna(method='ffill')
fill_DF
```

[18]:
```
          0         1         2         3         4         5
0  0.870124  0.582277  0.278839  0.185911  0.411100  0.117376
1  0.684969  0.437611  0.556229  0.367080  0.402366  0.117376
2  0.447031  0.585445  0.161985  0.520719  0.326051  0.117376
3  0.447031  0.836375  0.481343  0.516502  0.383048  0.117376
4  0.447031  0.559053  0.034450  0.719930  0.421004  0.117376
5  0.447031  0.900274  0.669612  0.456069  0.289804  0.525819
```

2.1.3 Counting missing values

[19]:
```
np.random.seed(25)
DF_obj = DataFrame(np.random.rand(36).reshape(6,6))
DF_obj.loc[3:5, 0] = missing
DF_obj.loc[1:4, 5] = missing
DF_obj
```

[19]:

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	NaN
2	0.447031	0.585445	0.161985	0.520719	0.326051	NaN
3	NaN	0.836375	0.481343	0.516502	0.383048	NaN
4	NaN	0.559053	0.034450	0.719930	0.421004	NaN
5	NaN	0.900274	0.669612	0.456069	0.289804	0.525819

[20]:
```
DF_obj.isnull().sum()
```

[20]:
```
0    3
1    0
2    0
3    0
4    0
5    4
dtype: int64
```

[21]:
```
DF_no_NaN = DF_obj.dropna()
DF_no_NaN
```

[21]:

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.4111	0.117376

[22]:
```
DF_no_NaN = DF_obj.dropna(axis=1)
DF_no_NaN
```

[22]:

	1	2	3	4
0	0.582277	0.278839	0.185911	0.411100
1	0.437611	0.556229	0.367080	0.402366
2	0.585445	0.161985	0.520719	0.326051
3	0.836375	0.481343	0.516502	0.383048
4	0.559053	0.034450	0.719930	0.421004
5	0.900274	0.669612	0.456069	0.289804

3 Chapter 2 - Data Preparation Basics

3.1 Segment 3 - Removing duplicates

3.1.1 Removing duplicates

[23]:
```
DF_obj= DataFrame({'column 1':[1,1,2,2,3,3,3],
                   'column 2':['a', 'a','b', 'b', 'c', 'c', 'c'],
                   'column 3':['A', 'A', 'B', 'B', 'C', 'C', 'C']})
DF_obj
```

[23]:

	column 1	column 2	column 3
0	1	a	A

```
1          1          a          A
2          2          b          B
3          2          b          B
4          3          c          C
5          3          c          C
6          3          c          C
```

[24]:
```
DF_obj.duplicated()
```

[24]:
```
0    False
1     True
2    False
3     True
4    False
5     True
6     True
dtype: bool
```

[25]:
```
DF_obj.drop_duplicates()
```

[25]:
```
   column 1 column 2 column 3
0          1          a          A
2          2          b          B
4          3          c          C
```

3.2 Segment 4 - Concatenating and transforming data

[26]:
```
DF_obj = pd.DataFrame(np.arange(36).reshape(6,6))
DF_obj
```

[26]:
```
    0   1   2   3   4   5
0   0   1   2   3   4   5
1   6   7   8   9  10  11
2  12  13  14  15  16  17
3  18  19  20  21  22  23
4  24  25  26  27  28  29
5  30  31  32  33  34  35
```

[27]:
```
DF_obj_2 = pd.DataFrame(np.arange(15).reshape(5,3))
DF_obj_2
```

[27]:
```
    0   1   2
0   0   1   2
1   3   4   5
2   6   7   8
3   9  10  11
4  12  13  14
```

3.2.1 Concatenating data

[28]:
```
pd.concat([DF_obj, DF_obj_2], axis=1)
```

[28]:

	0	1	2	3	4	5	0	1	2
0	0	1	2	3	4	5	0.0	1.0	2.0
1	6	7	8	9	10	11	3.0	4.0	5.0
2	12	13	14	15	16	17	6.0	7.0	8.0
3	18	19	20	21	22	23	9.0	10.0	11.0
4	24	25	26	27	28	29	12.0	13.0	14.0
5	30	31	32	33	34	35	NaN	NaN	NaN

[29]:
```
pd.concat([DF_obj, DF_obj_2])
```

[29]:

	0	1	2	3	4	5
0	0	1	2	3.0	4.0	5.0
1	6	7	8	9.0	10.0	11.0
2	12	13	14	15.0	16.0	17.0
3	18	19	20	21.0	22.0	23.0
4	24	25	26	27.0	28.0	29.0
5	30	31	32	33.0	34.0	35.0
0	0	1	2	NaN	NaN	NaN
1	3	4	5	NaN	NaN	NaN
2	6	7	8	NaN	NaN	NaN
3	9	10	11	NaN	NaN	NaN
4	12	13	14	NaN	NaN	NaN

3.2.2 Transforming data

Dropping data

[30]:
```
DF_obj.drop([0, 2])
```

[30]:

	0	1	2	3	4	5
1	6	7	8	9	10	11
3	18	19	20	21	22	23
4	24	25	26	27	28	29
5	30	31	32	33	34	35

[31]:
```
DF_obj.drop([0, 2], axis=1)
```

[31]:

	1	3	4	5
0	1	3	4	5
1	7	9	10	11
2	13	15	16	17
3	19	21	22	23
4	25	27	28	29
5	31	33	34	35

3.2.3 Adding data

[32]:
```
series_obj = Series(np.arange(6))
series_obj.name = "added_variable"
series_obj
```

[32]:
```
0    0
1    1
2    2
3    3
4    4
5    5
Name: added_variable, dtype: int32
```

[33]:
```
variable_added = DataFrame.join(DF_obj, series_obj)
variable_added
```

[33]:

	0	1	2	3	4	5	added_variable
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1
2	12	13	14	15	16	17	2
3	18	19	20	21	22	23	3
4	24	25	26	27	28	29	4
5	30	31	32	33	34	35	5

[34]:
```
added_datatable = variable_added.append(variable_added, ignore_index=False)
added_datatable
```

[34]:

	0	1	2	3	4	5	added_variable
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1
2	12	13	14	15	16	17	2
3	18	19	20	21	22	23	3
4	24	25	26	27	28	29	4
5	30	31	32	33	34	35	5
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1
2	12	13	14	15	16	17	2
3	18	19	20	21	22	23	3
4	24	25	26	27	28	29	4
5	30	31	32	33	34	35	5

[35]:
```
added_datatable = variable_added.append(variable_added, ignore_index=True)
added_datatable
```

[35]:

	0	1	2	3	4	5	added_variable
0	0	1	2	3	4	5	0
1	6	7	8	9	10	11	1

```
2   12  13  14  15  16  17              2
3   18  19  20  21  22  23              3
4   24  25  26  27  28  29              4
5   30  31  32  33  34  35              5
6    0   1   2   3   4   5              0
7    6   7   8   9  10  11              1
8   12  13  14  15  16  17              2
9   18  19  20  21  22  23              3
10  24  25  26  27  28  29              4
11  30  31  32  33  34  35              5
```

3.2.4 Sorting data

[36]:
```
DF_sorted = DF_obj.sort_values(by=(5), ascending=[False])
DF_sorted
```

[36]:
```
    0   1   2   3   4   5
5  30  31  32  33  34  35
4  24  25  26  27  28  29
3  18  19  20  21  22  23
2  12  13  14  15  16  17
1   6   7   8   9  10  11
0   0   1   2   3   4   5
```

3.3 Segment 5 - Grouping and data aggregation

[37]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)

cars.columns = ['car_names', 'mpg', 'cyl', 'disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']
cars.head()
```

[37]:
```
           car_names   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  \
0          Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1     4
1      Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1     4
2         Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1     4
3     Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0     3
4  Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0     3

   carb
0     4
1     4
2     1
3     1
4     2
```

[38]:
```
cars_groups = cars.groupby(cars['cyl'])
cars_groups.mean()
```

[38]:

cyl	mpg	disp	hp	drat	wt	qsec	vs	am	gear	carb
4	26.663636	105.136364	82.636364	4.070909	2.285727	19.137273	0.909091	0.727273	4.090909	1.545455
6	19.742857	183.314286	122.285714	3.585714	3.117143	17.977143	0.571429	0.428571	3.857143	3.428571
8	15.100000	353.100000	209.214286	3.229286	3.999214	16.772143	0.000000	0.142857	3.285714	3.500000

[39]:
```
cars_groups = cars.groupby(cars['am'])
cars_groups.mean()
```

[39]:

am	mpg	cyl	disp	hp	drat	wt	qsec	vs	gear	carb
0	17.147368	6.947368	290.378947	160.263158	3.286316	3.768895	18.183158	0.368421	3.210526	2.736842
1	24.392308	5.076923	143.530769	126.846154	4.050000	2.411000	17.360000	0.538462	4.384615	2.923077

[ ]:
```

```

Chapter4

March 1, 2022

1 Chapter 4 - Practical Data Visualization

1.1 Segment 1 - Creating standard data graphics

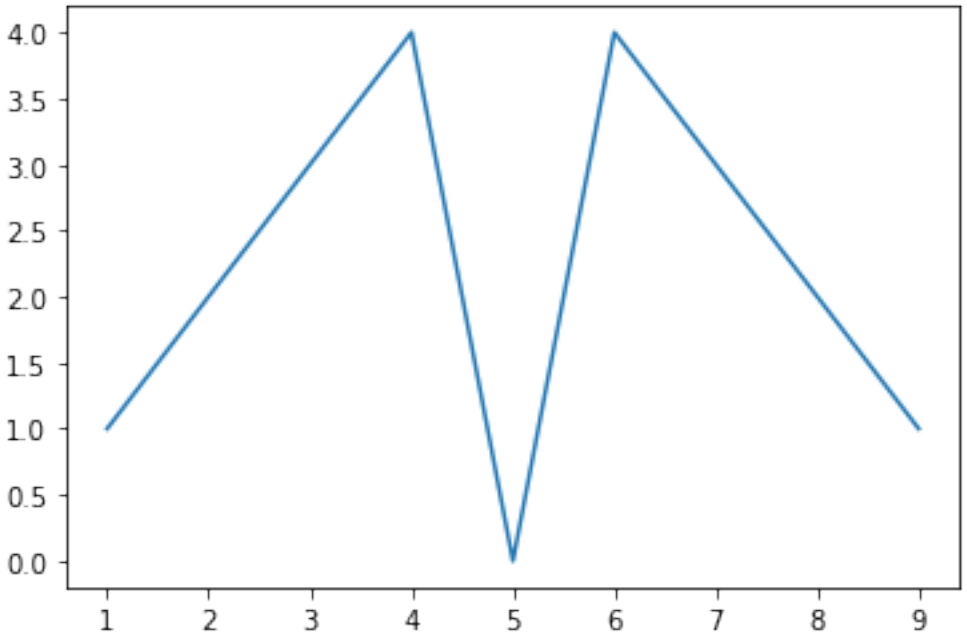
[1]:
```
import numpy as np
from numpy.random import randn
import pandas as pd
from pandas import Series, DataFrame

import matplotlib.pyplot as plt
from matplotlib import rcParams
```

1.1.1 Creating a line chart from a list object

Plotting a line chart in matplotlib

[2]:
```
x = range(1,10)
y = [1,2,3,4,0,4,3,2,1]

plt.plot(x,y)
```

[2]:
```
[<matplotlib.lines.Line2D at 0x26093e64880>]
```

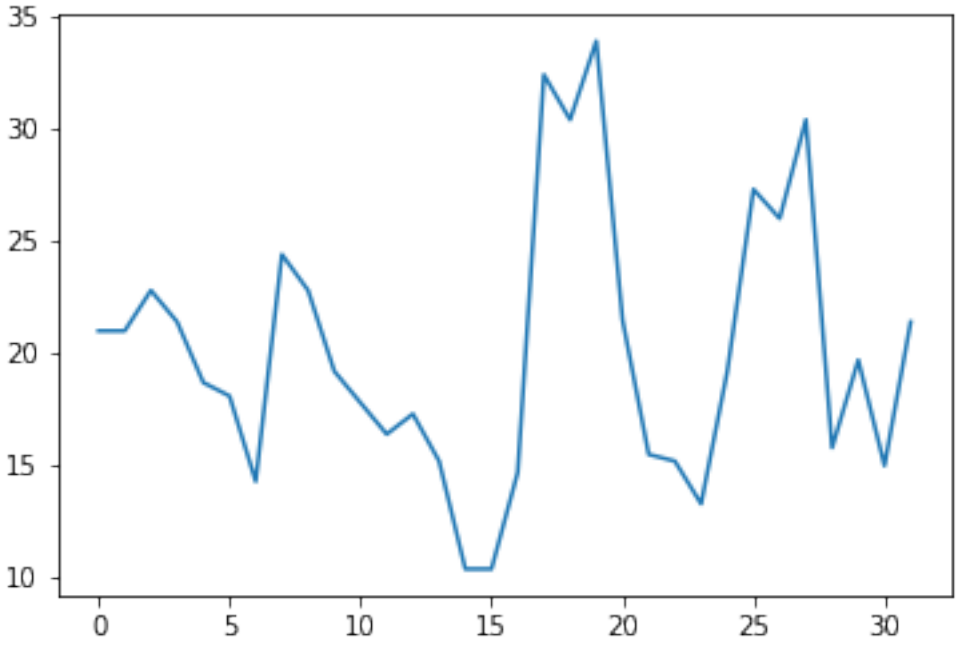


Plotting a line chart from a Pandas object

[3]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']

mpg = cars['mpg']
```

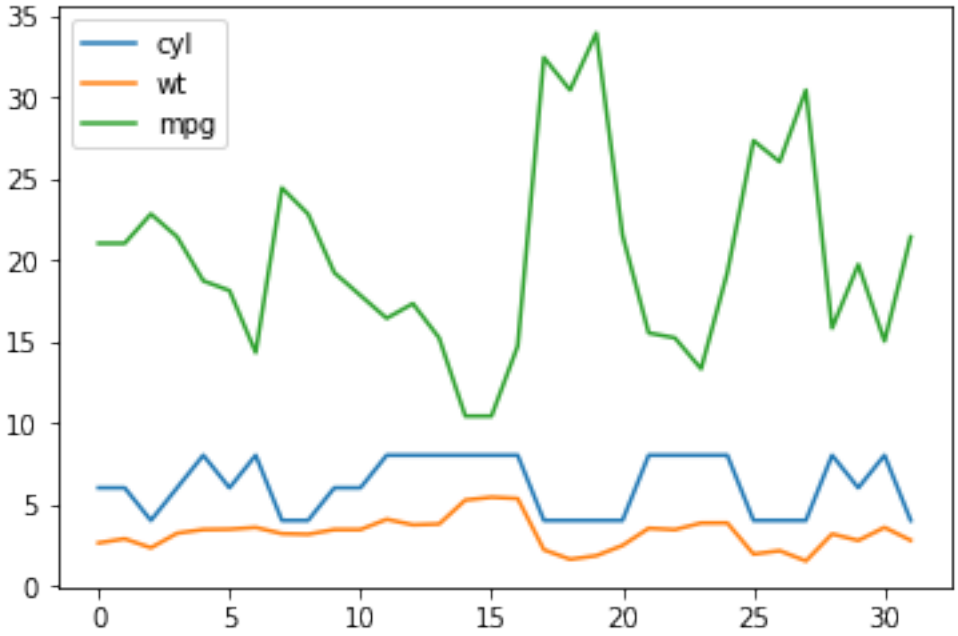
[4]:
```
mpg.plot()
```

[4]: <AxesSubplot:>



[5]:
```
df = cars[['cyl','wt','mpg']]
df.plot()
```

[5]: <AxesSubplot:>

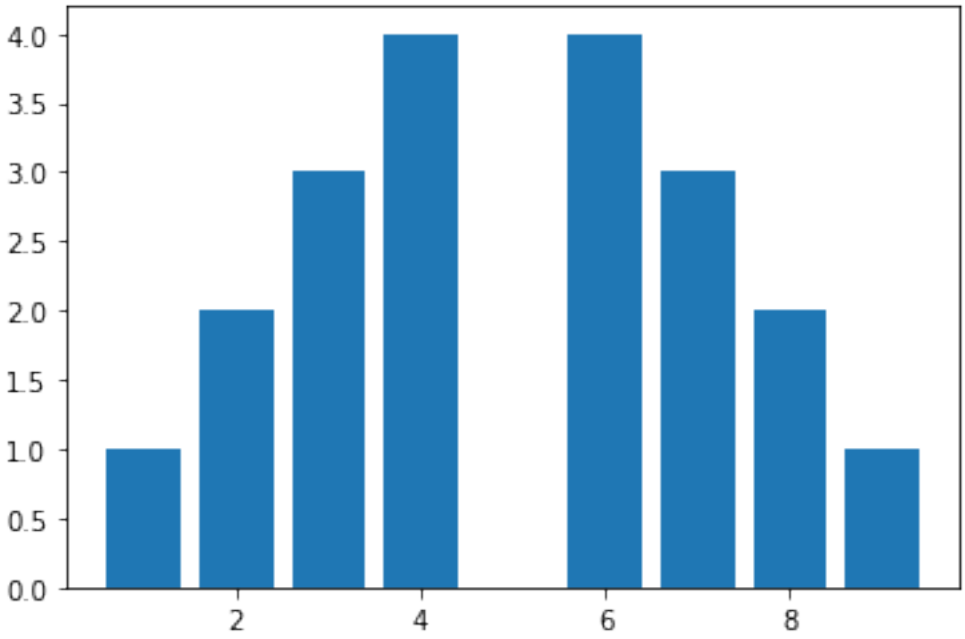


1.1.2 Creating bar charts

Creating a bar chart from a list

[6]:
```
plt.bar(x, y)
```

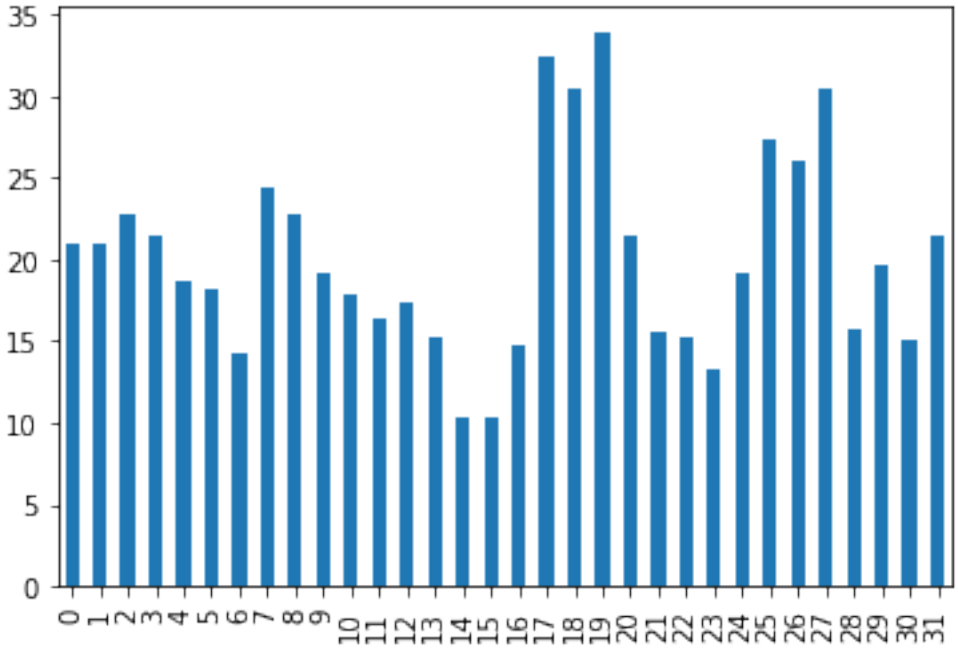
[6]: <BarContainer object of 9 artists>



Creating bar charts from Pandas objects

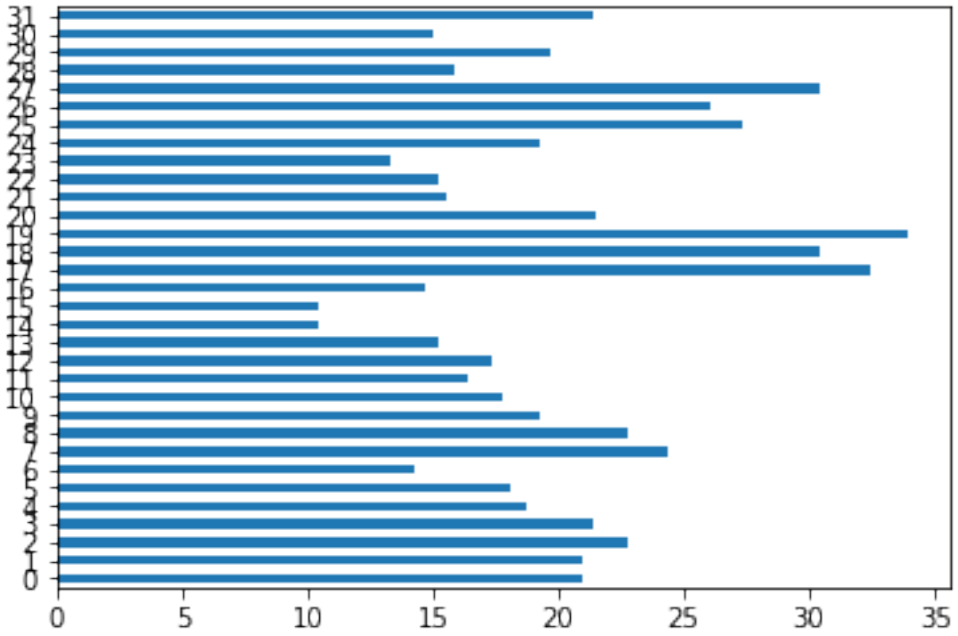
[7]:
```
mpg.plot(kind="bar")
```

[7]: <AxesSubplot:>



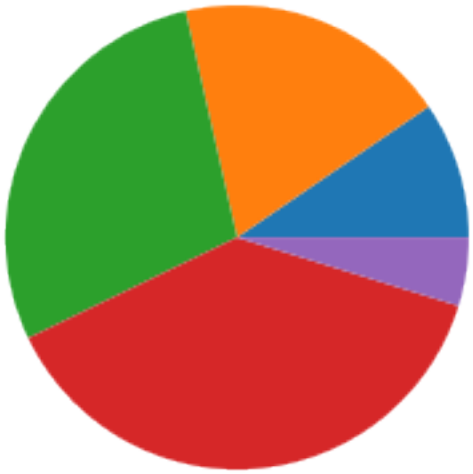
[8]:
```
mpg.plot(kind="barh")
```

[8]: <AxesSubplot:>



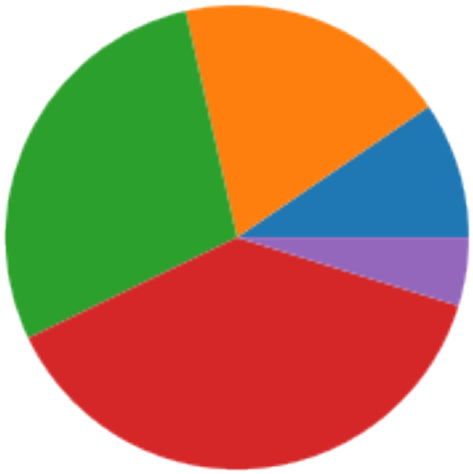
1.1.3 Creating a pie chart

[9]:
```
x = [1,2,3,4,0.5]
plt.pie(x)
plt.show()
```



1.1.4 Saving a plot

[10]:
```
plt.pie(x)
plt.savefig('pie_chart.png')
plt.show()
```



[11]:
```
%pwd
```

[11]:
```
'C:\\Users\\aadar\\Documents\\TERM2\\BDM 1034 - Application Design for Big
Data\\Week6\\Assignment'
```

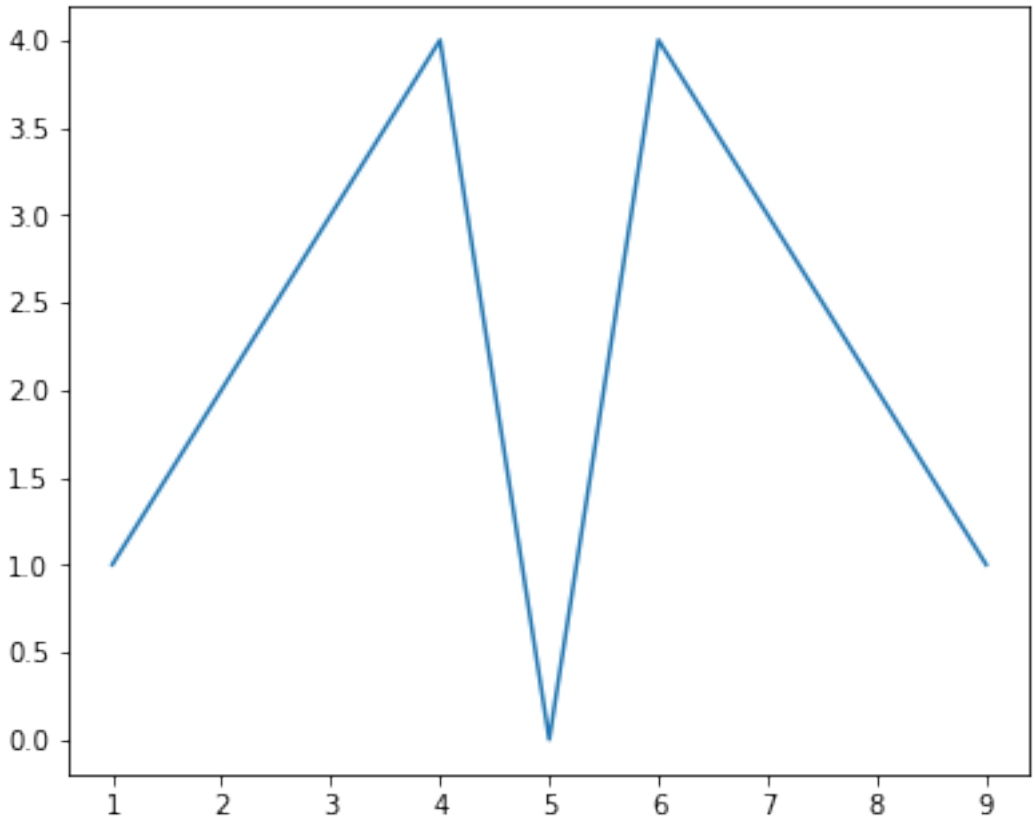
1.2 Segment 2 - Defining elements of a plot

[12]:
```
%matplotlib inline
rcParams['figure.figsize']= 5,4
```

1.2.1 Defining axes, ticks, and grids

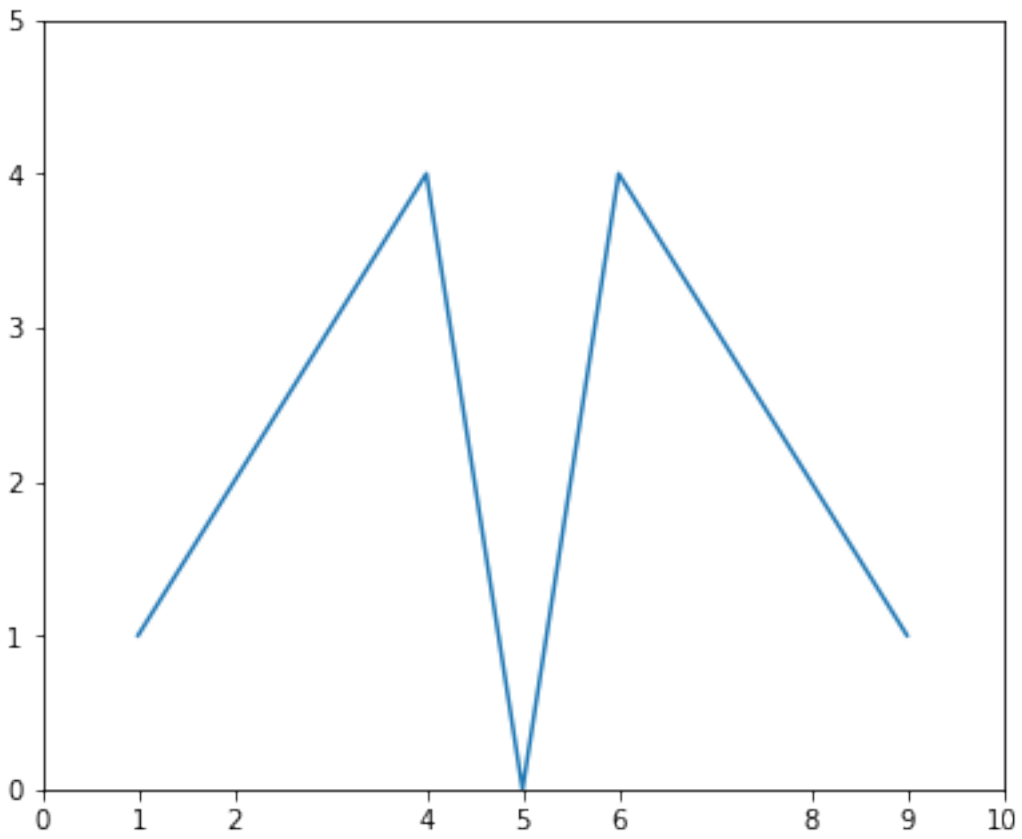
[13]:
```
x = range(1,10)
y = [1,2,3,4,0,4,3,2,1]

fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

ax.plot(x,y)
```

[13]:
```
[<matplotlib.lines.Line2D at 0x26094a6d3a0>]
```



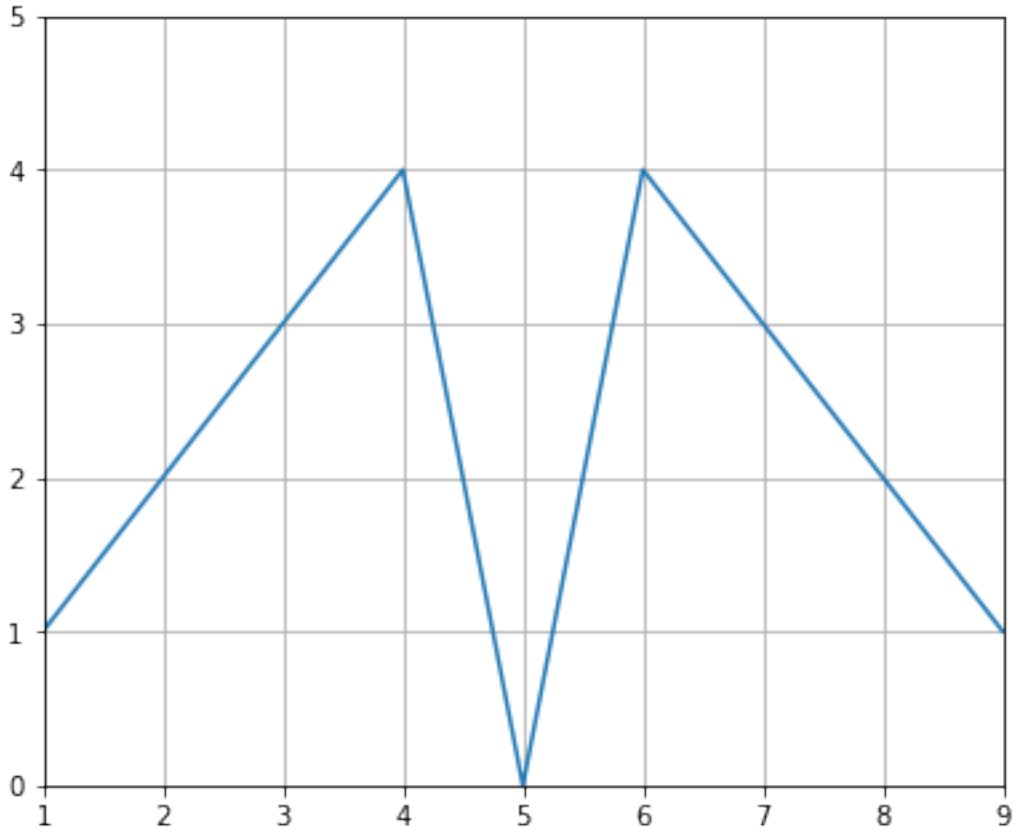
[14]:
```
fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

ax.set_xlim([1,9])
ax.set_ylim([0,5])

ax.set_xticks([0,1,2,4,5,6,8,9,10])
ax.set_yticks([0,1,2,3,4,5])

ax.plot(x,y)
```

[14]: [<matplotlib.lines.Line2D at 0x26094ae4ee0>]



[15]:
```
fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

ax.set_xlim([1,9])
ax.set_ylim([0,5])

ax.grid()
ax.plot(x,y)
```

[15]: [<matplotlib.lines.Line2D at 0x26095af8880>]

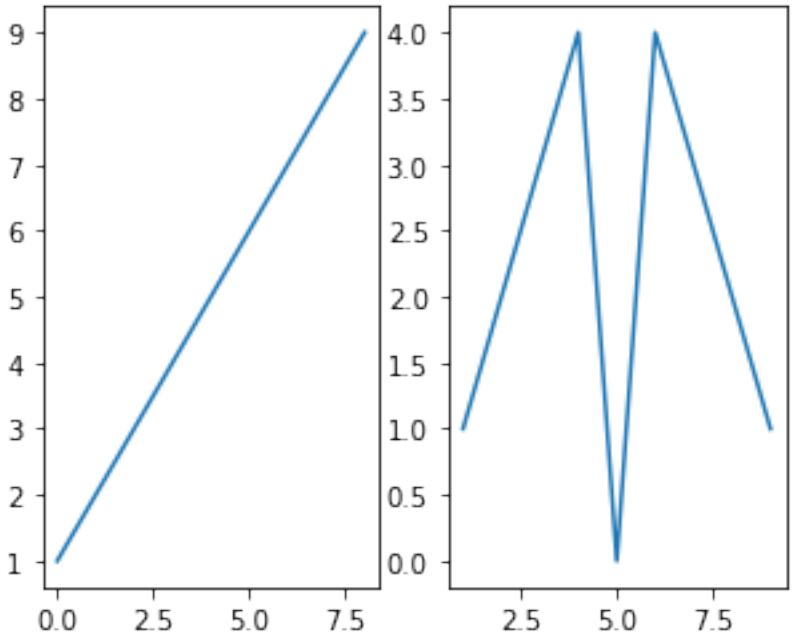


1.2.2 Generating multiple plots in one figure with subplots

[16]:
```
fig = plt.figure()
fig, (ax1, ax2) = plt.subplots(1,2)

ax1.plot(x)
ax2.plot(x,y)
```

[16]: [<matplotlib.lines.Line2D at 0x26095b96280>]

<Figure size 360x288 with 0 Axes>

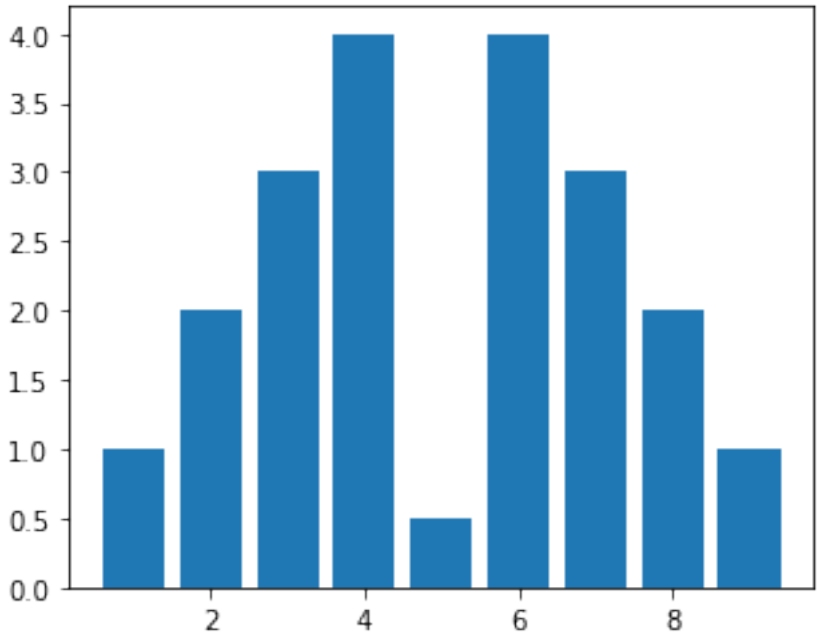


1.3 Segment 3 - Plot formatting

1.3.1 Defining plot color

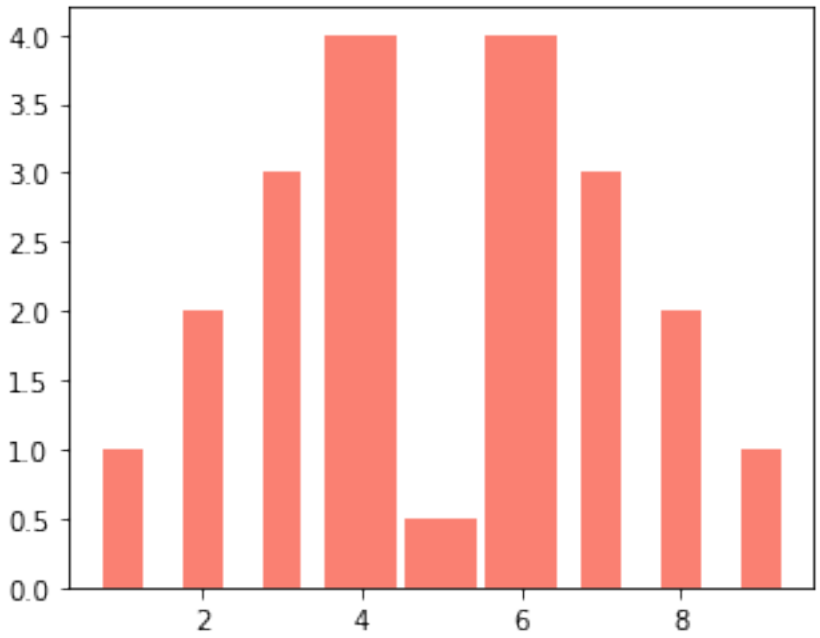
[17]:
```
x = range(1,10)
y = [1,2,3,4,0.5,4,3,2,1]

plt.bar(x,y)
```

[17]: <BarContainer object of 9 artists>



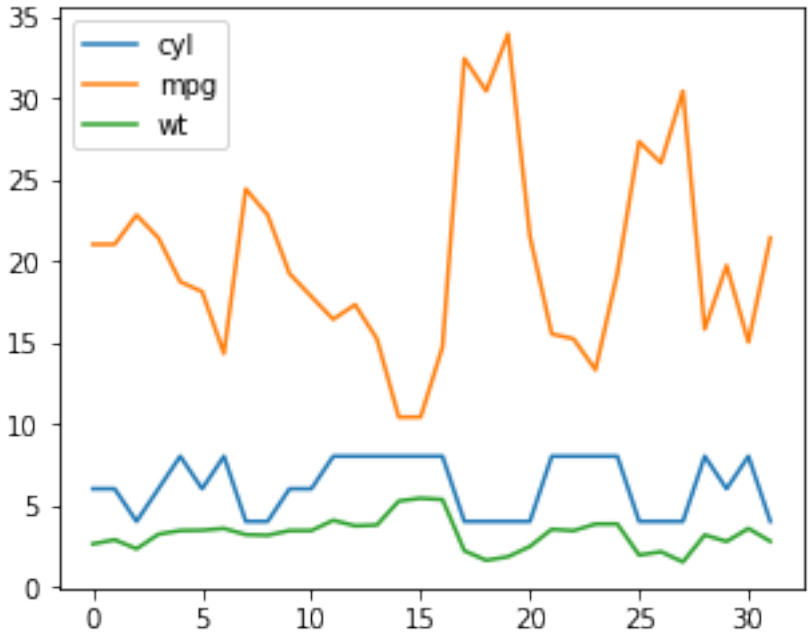
[18]:
```
wide = [.5,.5,.5,.9,.9,.9,.5,.5,.5]
color = ['salmon']
plt.bar(x, y, width=wide, color=color, align='center')
```

[18]: <BarContainer object of 9 artists>



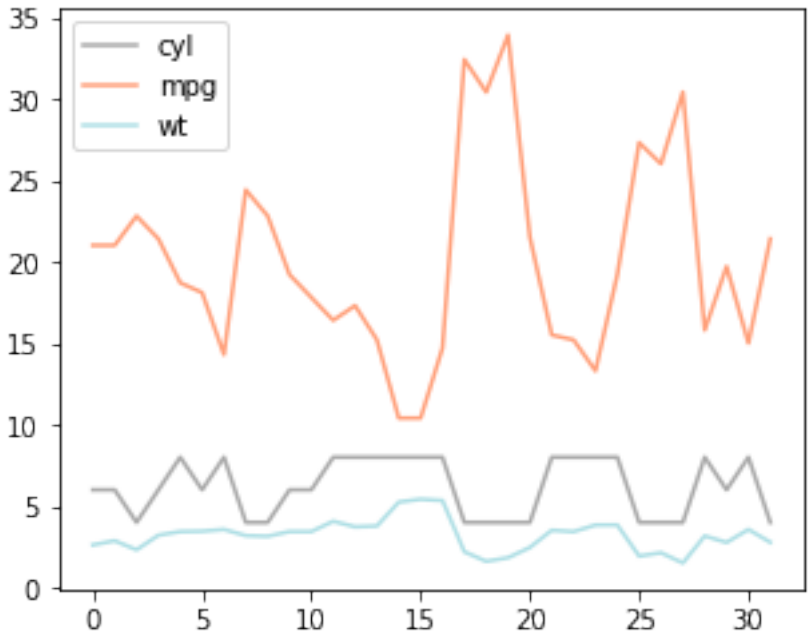
[19]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']

df = cars[['cyl', 'mpg', 'wt']]
df.plot()
```

[19]: <AxesSubplot:>

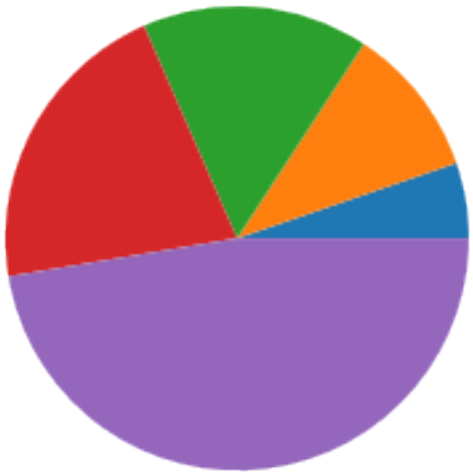


[20]:
```
color_theme = ['darkgray', 'lightsalmon', 'powderblue']
df.plot(color=color_theme)
```

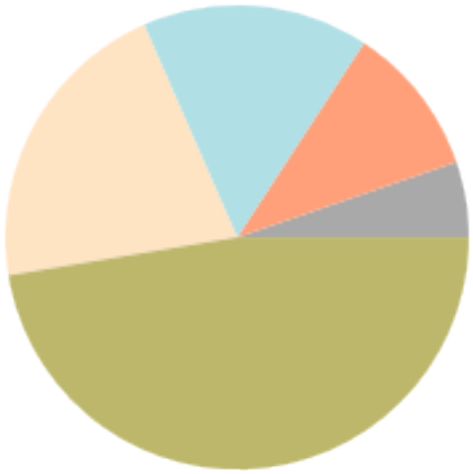
[20]: <AxesSubplot:>



[21]:
```
z = [1,2,3,4,9]
plt.pie(z)
plt.show()
```



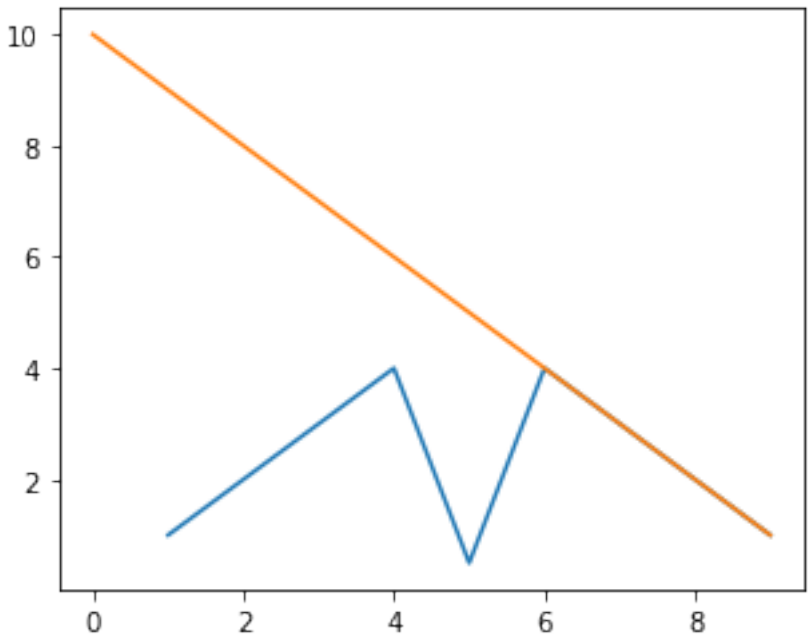
[22]:
```
color_theme = ['#A9A9A9', '#FFA07A', '#B0E0E6', '#FFE4C4', '#BDB76B']
plt.pie(z, colors=color_theme)
plt.show()
```



1.3.2 Customizing line styles

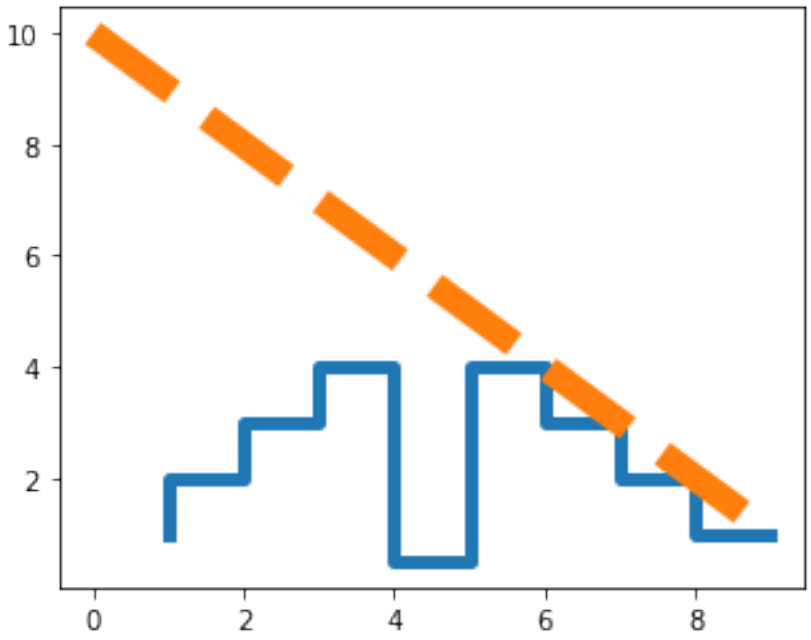
[23]:
```
x1 = range(0,10)
y1 = [10,9,8,7,6,5,4,3,2,1]

plt.plot(x, y)
plt.plot(x1, y1)
```

[23]:
```
[<matplotlib.lines.Line2D at 0x26095db2400>]
```



[24]:
```
plt.plot(x, y, ds='steps', lw=5)
plt.plot(x1, y1, ls='--', lw=10)
```

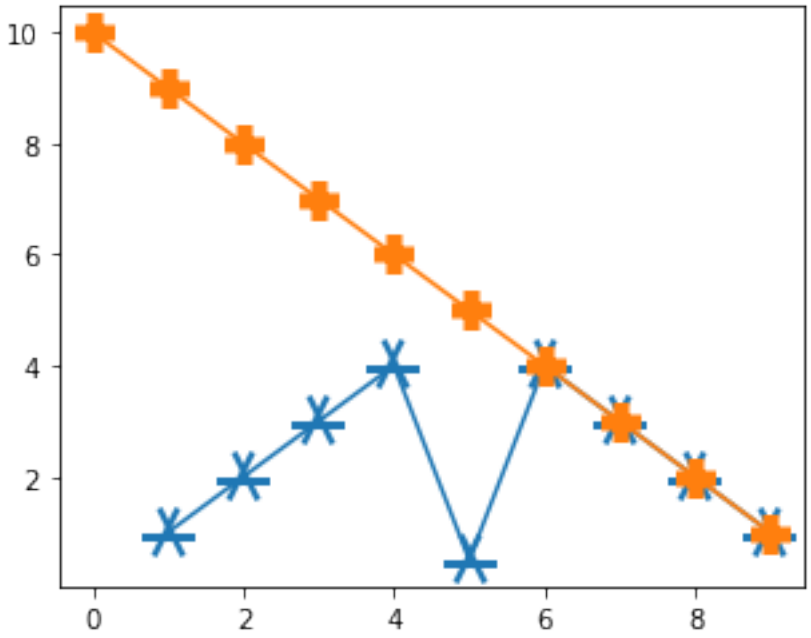
[24]: [<matplotlib.lines.Line2D at 0x26095db2fa0>]



1.3.3 Setting plot markers

[25]:
```
plt.plot(x, y, marker='1', mew=20)
plt.plot(x1, y1, marker='+', mew=15)
```

[25]:
```
[<matplotlib.lines.Line2D at 0x26095e5a460>]
```



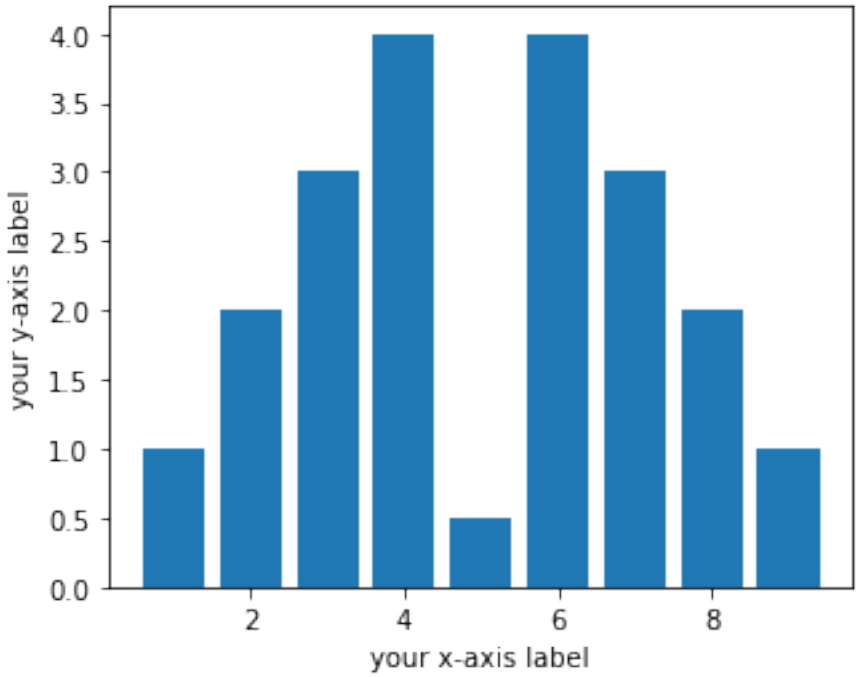
1.4 Segment 4 - Creating labels and annotations

1.4.1 Labeling plot features

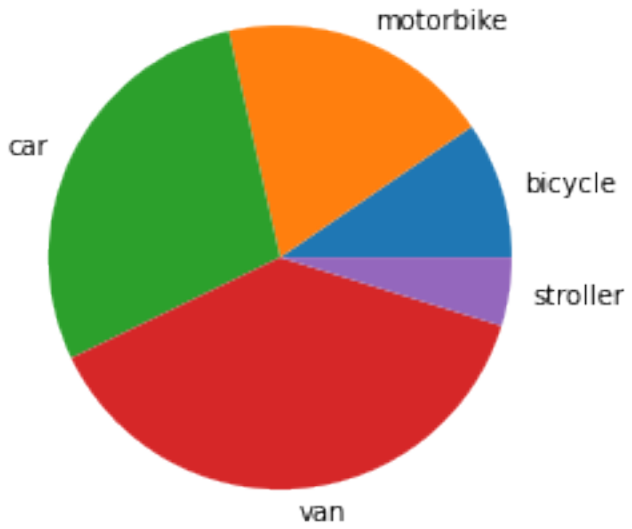
The functional method

[26]:
```
x = range(1,10)
y = [1,2,3,4,.5,4,3,2,1]
plt.bar(x,y)

plt.xlabel('your x-axis label')
plt.ylabel('your y-axis label')
```

[26]:
```
Text(0, 0.5, 'your y-axis label')
```



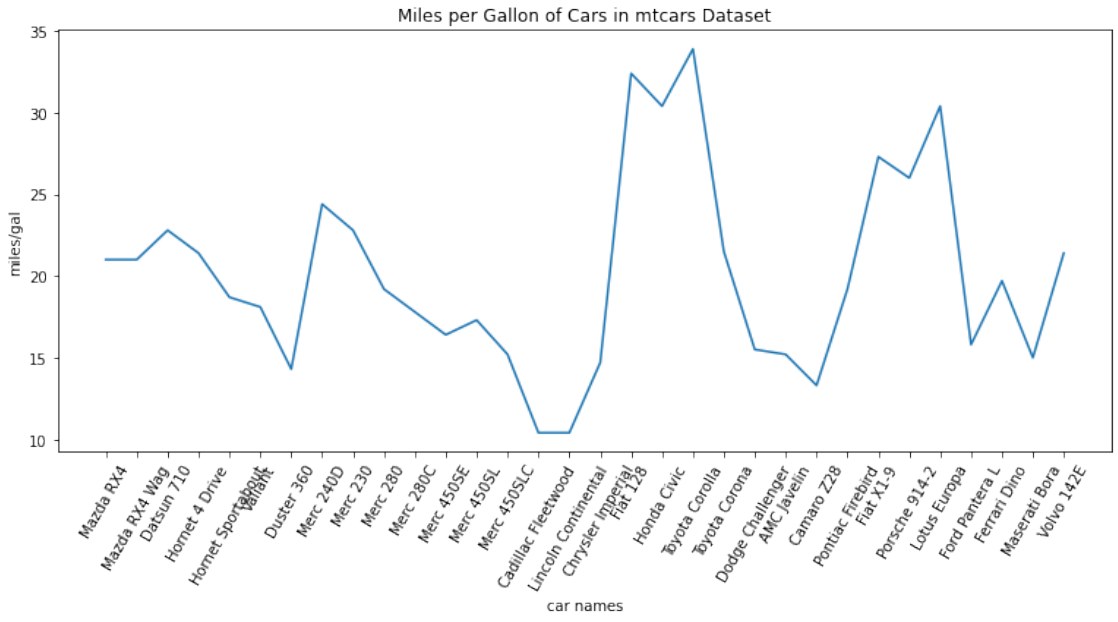
[27]:
```
z = [1,2,3,4,.5]
veh_type = ['bicycle', 'motorbike', 'car', 'van', 'stroller']

plt.pie(z, labels=veh_type)
plt.show()
```



The object-oriented method

[37]:
```
rcParams['figure.figsize']= 10,4
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']

mpg = cars.mpg

fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

mpg.plot()

ax.set_xticks(range(32))

ax.set_xticklabels(cars.car_names, rotation=60, fontsize='medium')
ax.set_title('Miles per Gallon of Cars in mtcars Dataset')

ax.set_xlabel('car names')
ax.set_ylabel('miles/gal')
```

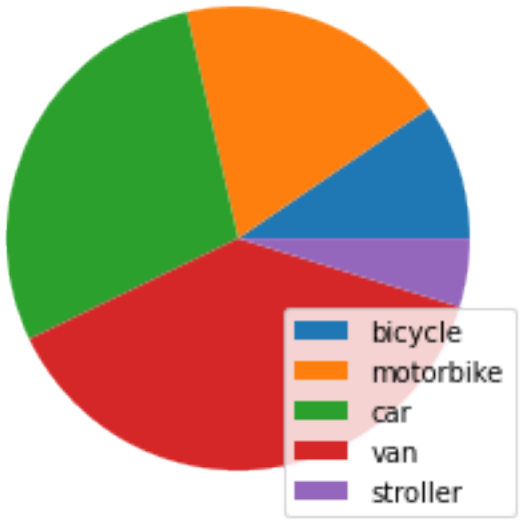
[37]:
```
Text(0, 0.5, 'miles/gal')
```



1.4.2 Adding a legend to your plot

The functional method

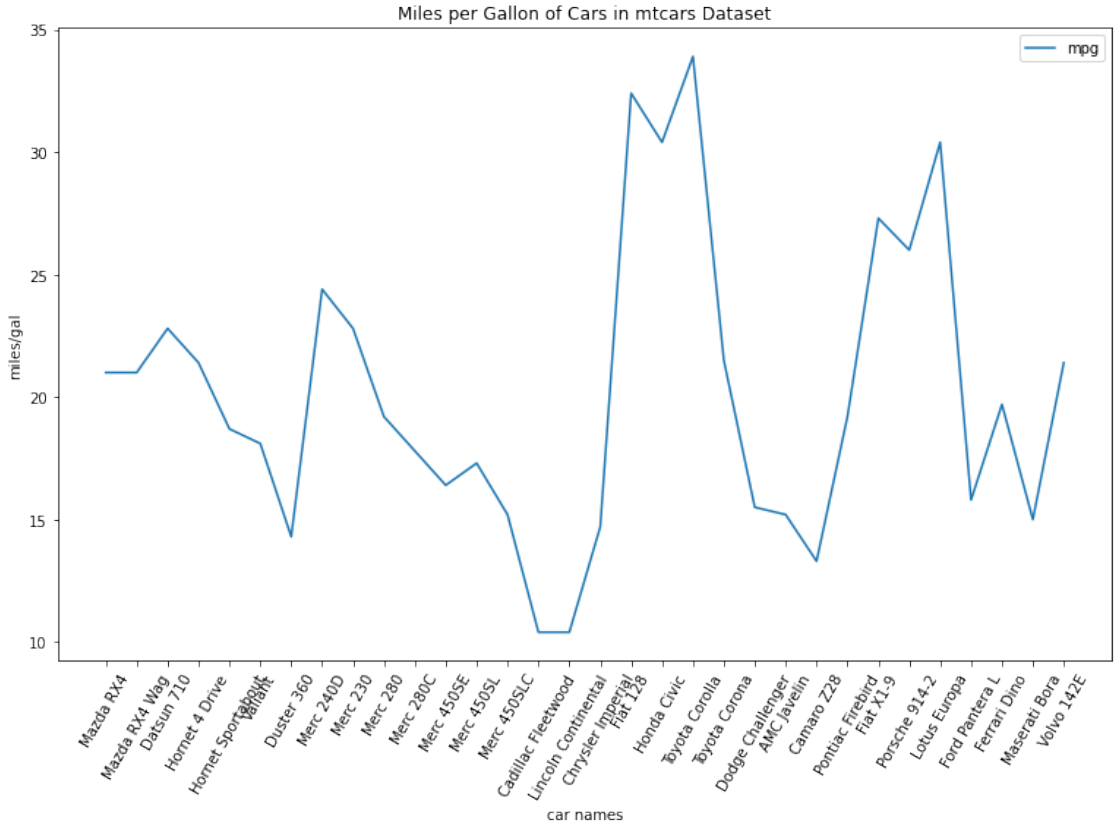
[29]:
```
plt.pie(z)
plt.legend(veh_type, loc='best')
plt.show()
```



The object-oriented method

[38]:
```
rcParams['figure.figsize']= 10,6
fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

mpg.plot()

ax.set_xticks(range(32))

ax.set_xticklabels(cars.car_names, rotation=60, fontsize='medium')
ax.set_title('Miles per Gallon of Cars in mtcars Dataset')

ax.set_xlabel('car names')
ax.set_ylabel('miles/gal')

ax.legend(loc='best')
```

[38]:
```
<matplotlib.legend.Legend at 0x260963bdf40>
```



1.4.3 Annotating your plot

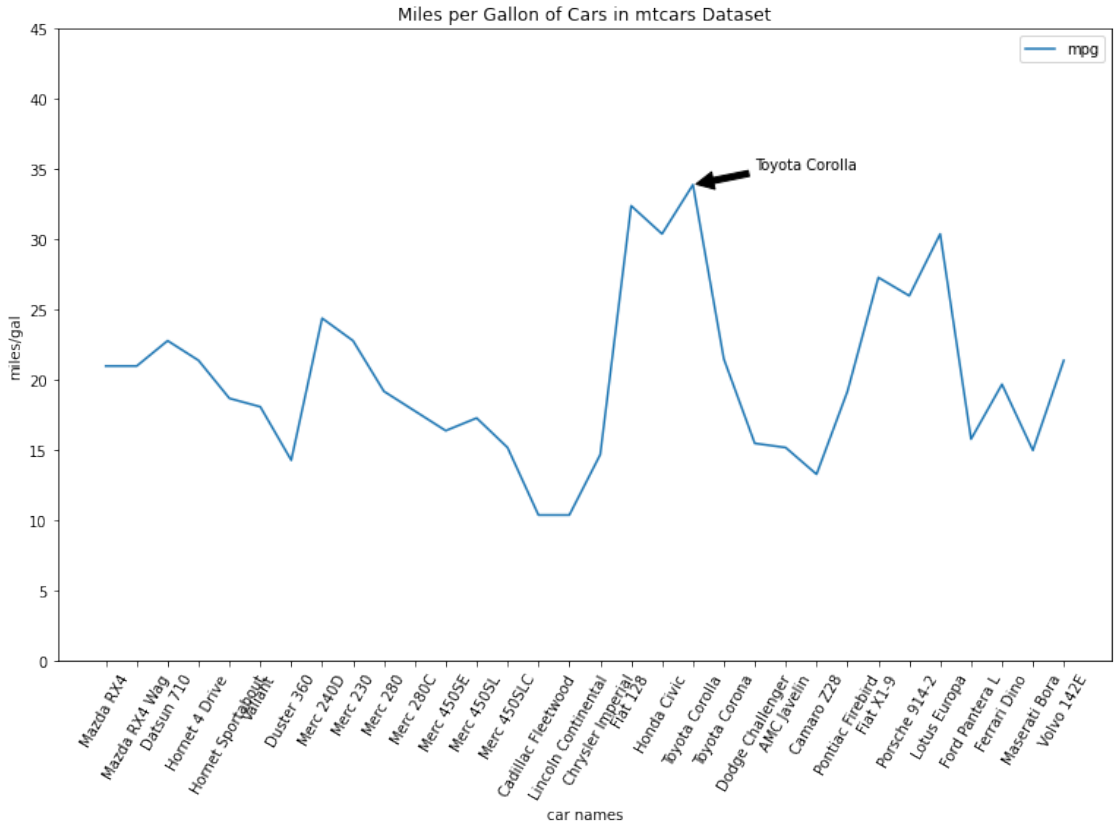
[31]:
```
mpg.max()
```

[31]: 33.9

[39]:
```
rcParams['figure.figsize']= 10,6
fig = plt.figure()
ax = fig.add_axes([.1,.1,1,1])

mpg.plot()

ax.set_xticks(range(32))

ax.set_xticklabels(cars.car_names, rotation=60, fontsize='medium')
ax.set_title('Miles per Gallon of Cars in mtcars Dataset')

ax.set_xlabel('car names')
ax.set_ylabel('miles/gal')

ax.legend(loc='best')
```

```
ax.set_ylim([0,45])

ax.annotate('Toyota Corolla', xy= (19,33.9), xytext=(21,35),
            arrowprops=dict(facecolor='black', shrink=0.05))
```

[39]:
```
Text(21, 35, 'Toyota Corolla')
```



1.4.4 The simplest time series plot

[43]:
```
rcParams['figure.figsize'] = 5, 4
address = './Data/Superstore-Sales.csv'

df = pd.read_csv(address, index_col='Order Date', encoding='cp1252',␣
 ↪parse_dates=True)
df.head()
```

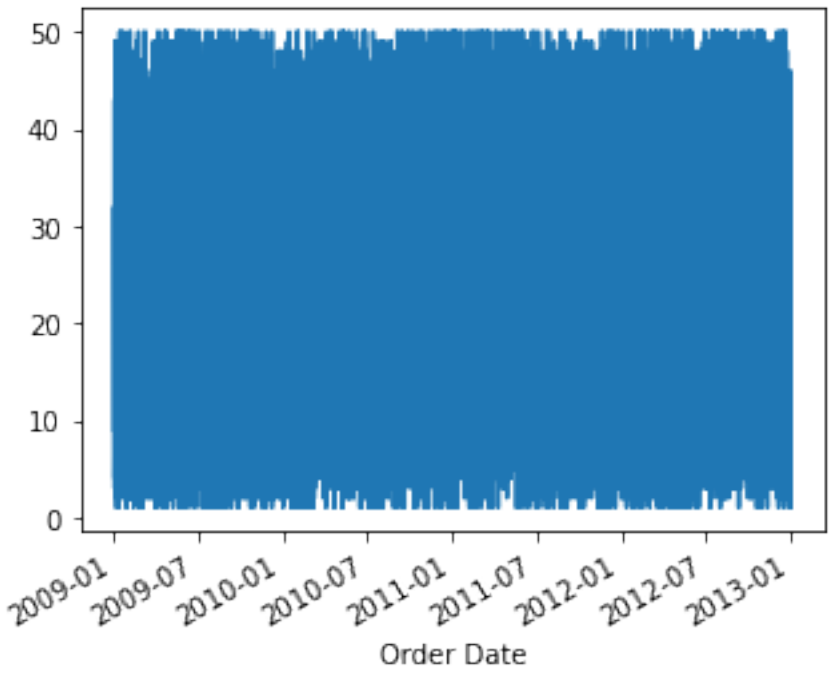
[43]:

Order Date	Row ID	Order ID	Order Priority	Order Quantity	Sales
2010-10-13	1	3	Low	6	261.5400
2012-10-01	49	293	High	49	10123.0200

```
2012-10-01      50       293           High              27    244.5700
2011-07-10      80       483           High              30   4965.7595
2010-08-28      85       515  Not Specified              19    394.2700

             Discount       Ship Mode   Profit  Unit Price  Shipping Cost  \
Order Date
2010-10-13       0.04     Regular Air  -213.25       38.94          35.00
2012-10-01       0.07  Delivery Truck   457.81      208.16          68.02
2012-10-01       0.01     Regular Air    46.71        8.69           2.99
2011-07-10       0.08     Regular Air  1198.97      195.99           3.99
2010-08-28       0.08     Regular Air    30.94       21.78           5.94

                  Customer Name Province   Region Customer Segment  \
Order Date
2010-10-13  Muhammed MacIntyre  Nunavut  Nunavut   Small Business
2012-10-01        Barry French  Nunavut  Nunavut         Consumer
2012-10-01        Barry French  Nunavut  Nunavut         Consumer
2011-07-10       Clay Rozendal  Nunavut  Nunavut        Corporate
2010-08-28      Carlos Soltero  Nunavut  Nunavut         Consumer

            Product Category            Product Sub-Category  \
Order Date
2010-10-13  Office Supplies          Storage & Organization
2012-10-01  Office Supplies                      Appliances
2012-10-01  Office Supplies  Binders and Binder Accessories
2011-07-10       Technology    Telephones and Communication
2010-08-28  Office Supplies                      Appliances

                                                 Product Name  \
Order Date
2010-10-13   Eldon Base for stackable storage shelf, platinum
2012-10-01  1.7 Cubic Foot Compact "Cube" Office Refrigera…
2012-10-01   Cardinal Slant-D® Ring Binder, Heavy Gauge Vinyl
2011-07-10                                              R380
2010-08-28                          Holmes HEPA Air Purifier

            Product Container  Product Base Margin   Ship Date
Order Date
2010-10-13        Large Box                 0.80  10/20/2010
2012-10-01       Jumbo Drum                 0.58   10/2/2012
2012-10-01        Small Box                 0.39   10/3/2012
2011-07-10        Small Box                 0.58   7/12/2011
2010-08-28       Medium Box                 0.50   8/30/2010
```

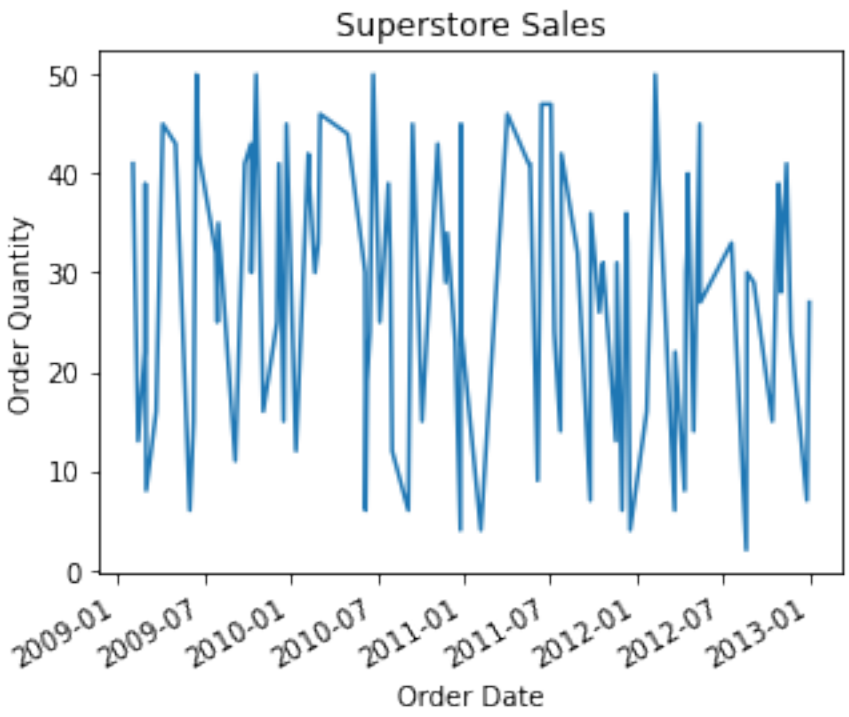
[44]:
```
df['Order Quantity'].plot()
```

[44]:
```
<AxesSubplot:xlabel='Order Date'>
```



[45]:
```
df2 = df.sample(n=100, random_state=25, axis=0)

plt.xlabel('Order Date')
plt.ylabel('Order Quantity')
plt.title('Superstore Sales')

df2['Order Quantity'].plot()
```

[45]:
```
<AxesSubplot:title={'center':'Superstore Sales'}, xlabel='Order Date',
ylabel='Order Quantity'>
```



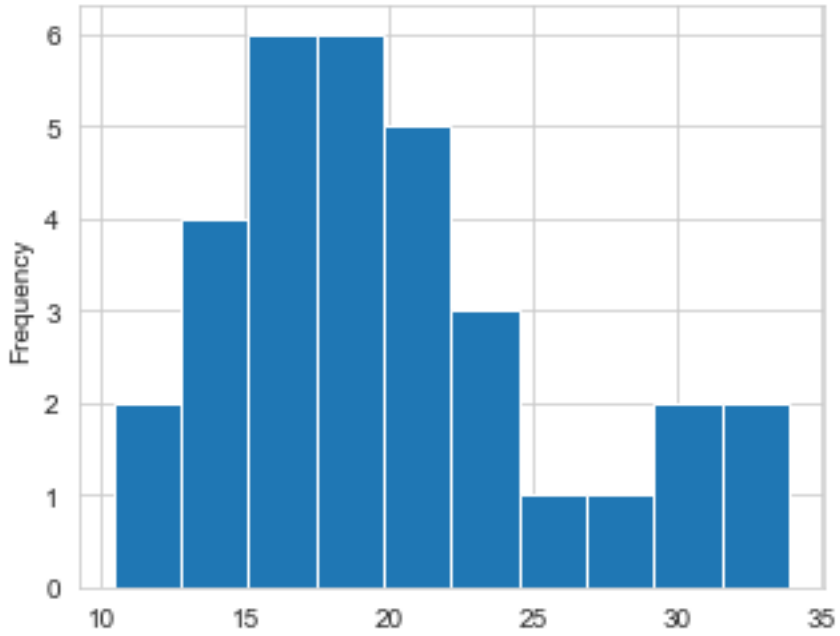
1.5 Segment 6 - Creating statistical data graphics|

[46]:
```
%matplotlib inline
rcParams['figure.figsize'] = 5, 4
import seaborn as sb
sb.set_style('whitegrid')
```

1.5.1 Eyeballing dataset distributions with histograms

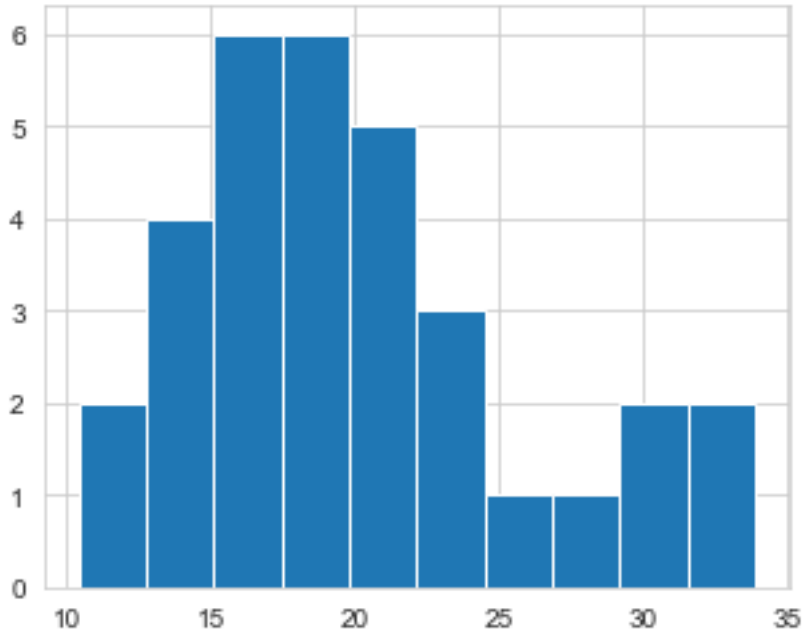
[48]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)

cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']

cars.index = cars.car_names

mpg = cars['mpg']

mpg.plot(kind='hist')
```

[48]:
```
<AxesSubplot:ylabel='Frequency'>
```



[49]:
```
plt.hist(mpg)
plt.plot()
```

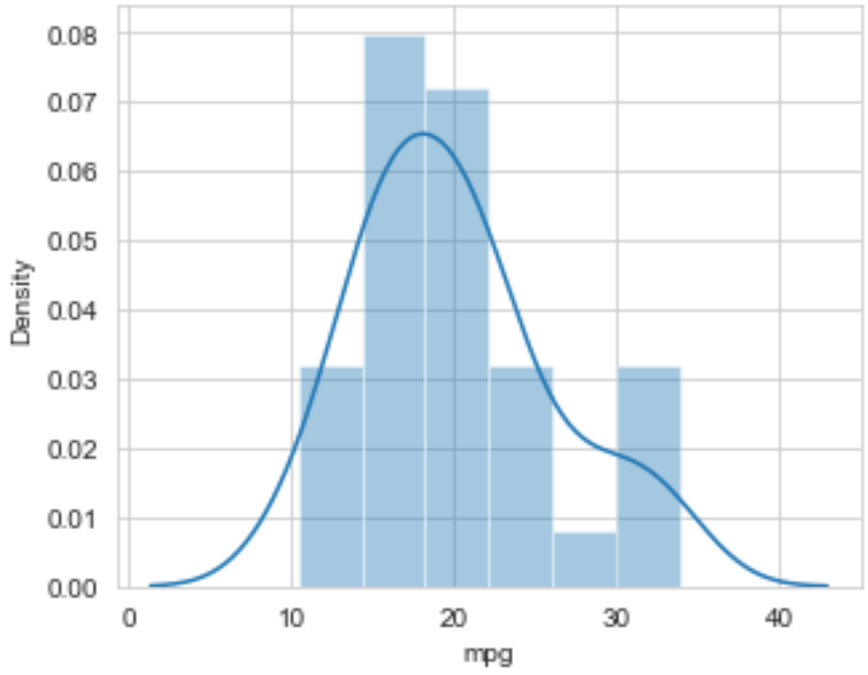
[49]: []



[50]:
```
sb.distplot(mpg)
```

```
E:\Anaconda\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:
`distplot` is a deprecated function and will be removed in a future version.
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

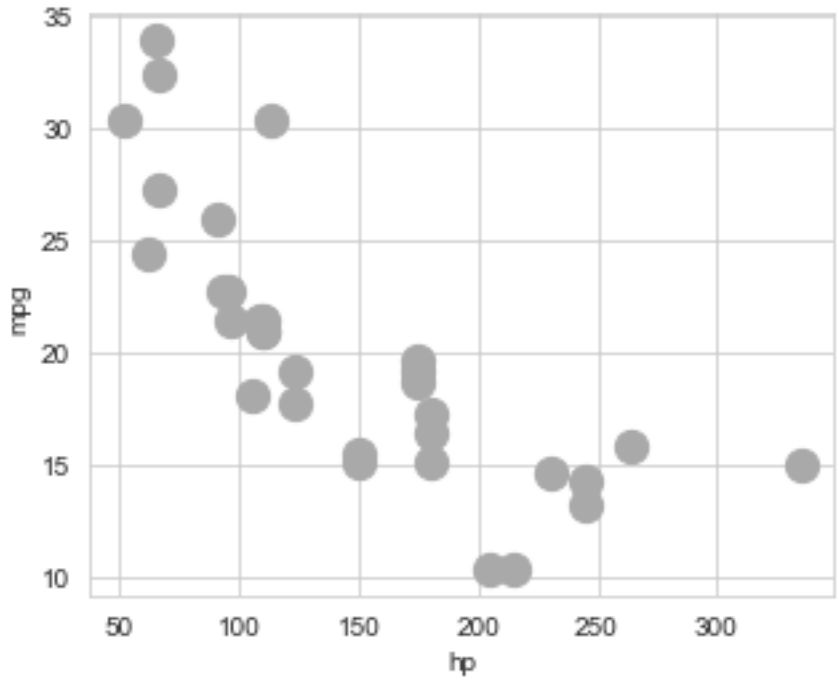
[50]:
```
<AxesSubplot:xlabel='mpg', ylabel='Density'>
```



1.5.2 Seeing scatterplots in action

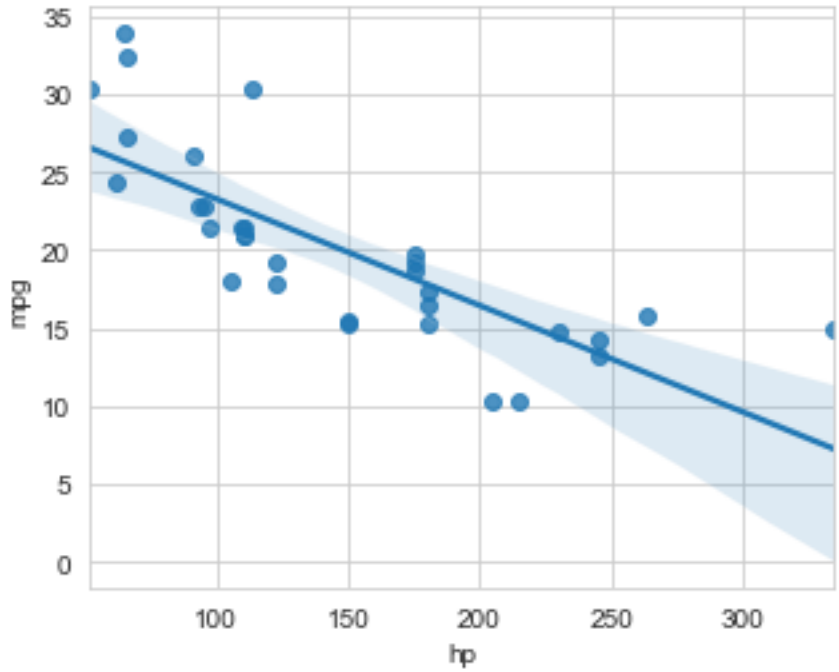
[52]:
```
cars.plot(kind='scatter', x='hp', y='mpg', c=['darkgray'], s=150)
```

[52]:
```
<AxesSubplot:xlabel='hp', ylabel='mpg'>
```



[53]:
```
sb.regplot(x='hp', y='mpg', data=cars, scatter=True)
```

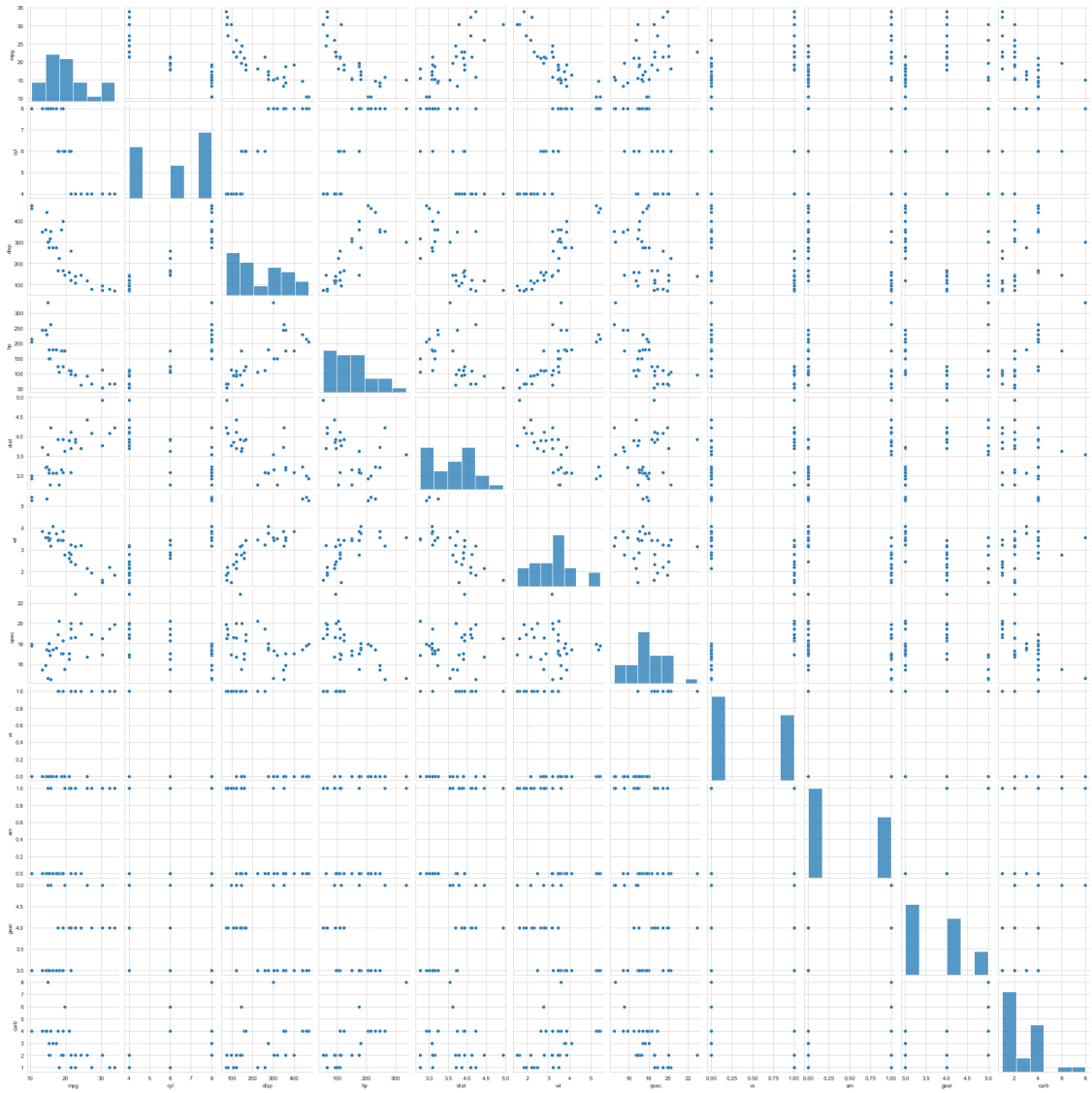
[53]:
```
<AxesSubplot:xlabel='hp', ylabel='mpg'>
```



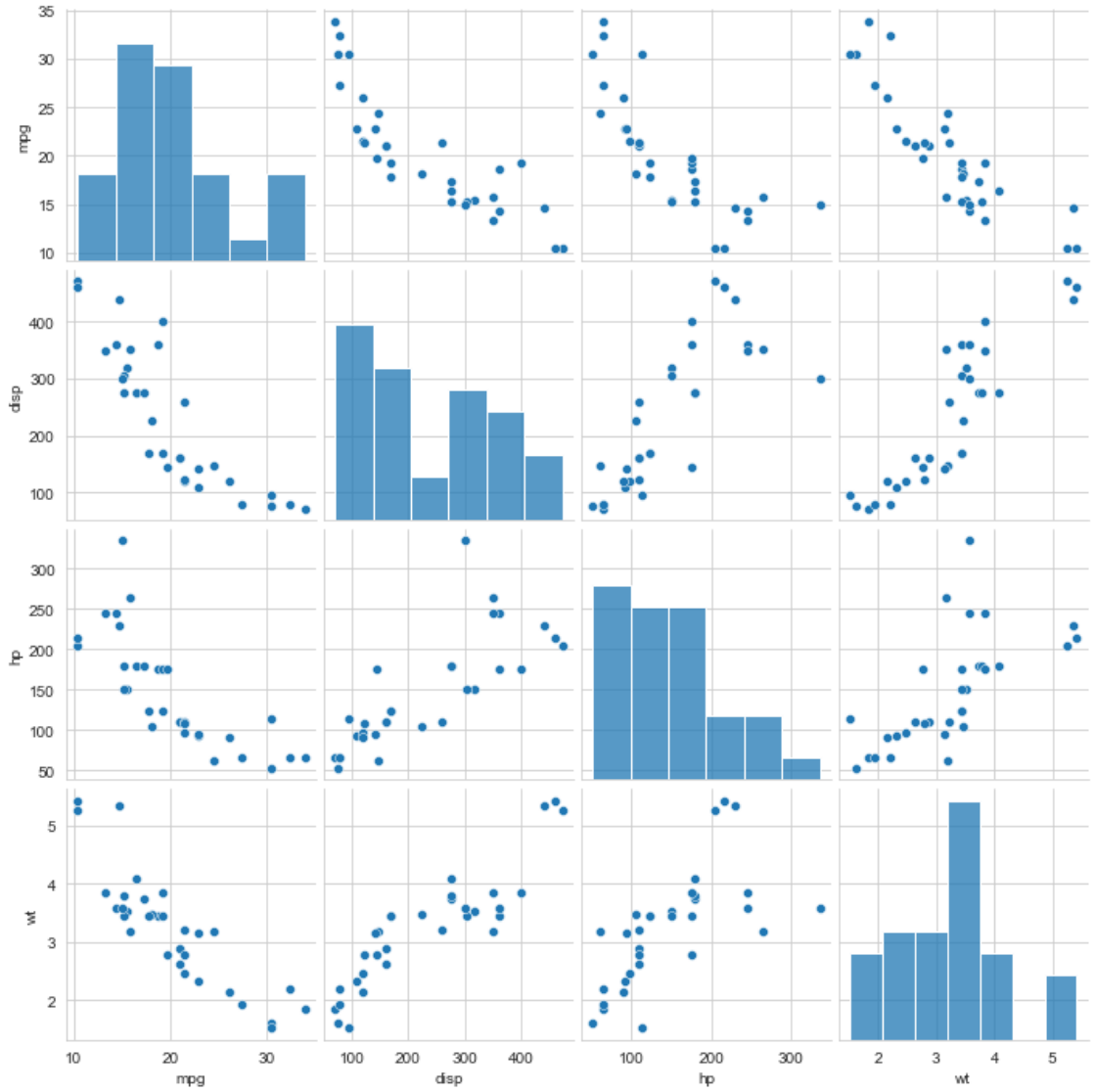
1.5.3 Generating a scatter plot matrix

[54]:
```
sb.pairplot(cars)
```

[54]:
```
<seaborn.axisgrid.PairGrid at 0x2609bee8d30>
```



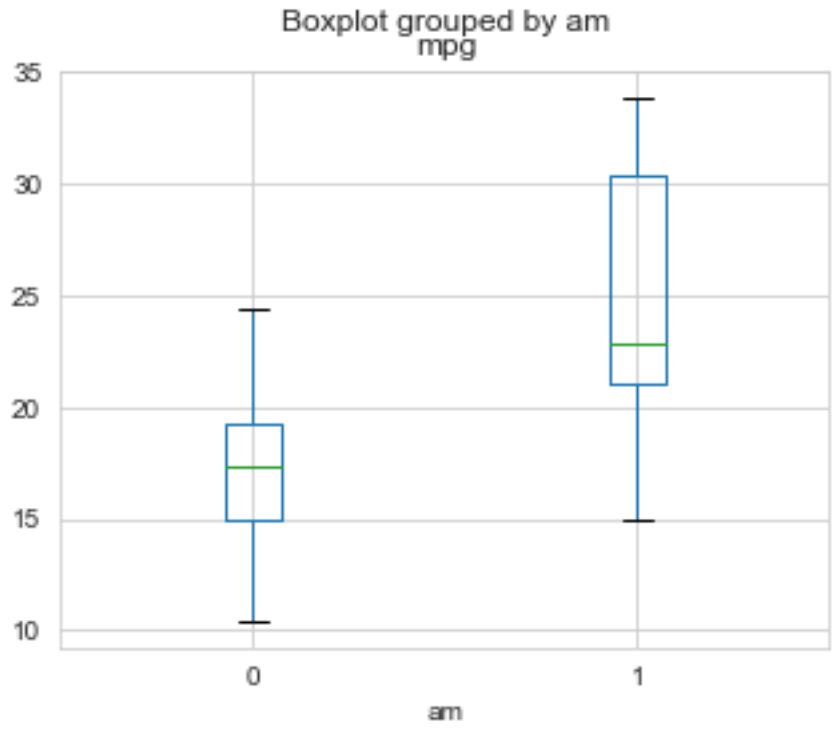
[56]:
```
cars_subset = cars[['mpg', 'disp', 'hp', 'wt']]
sb.pairplot(cars_subset)
plt.show()
```

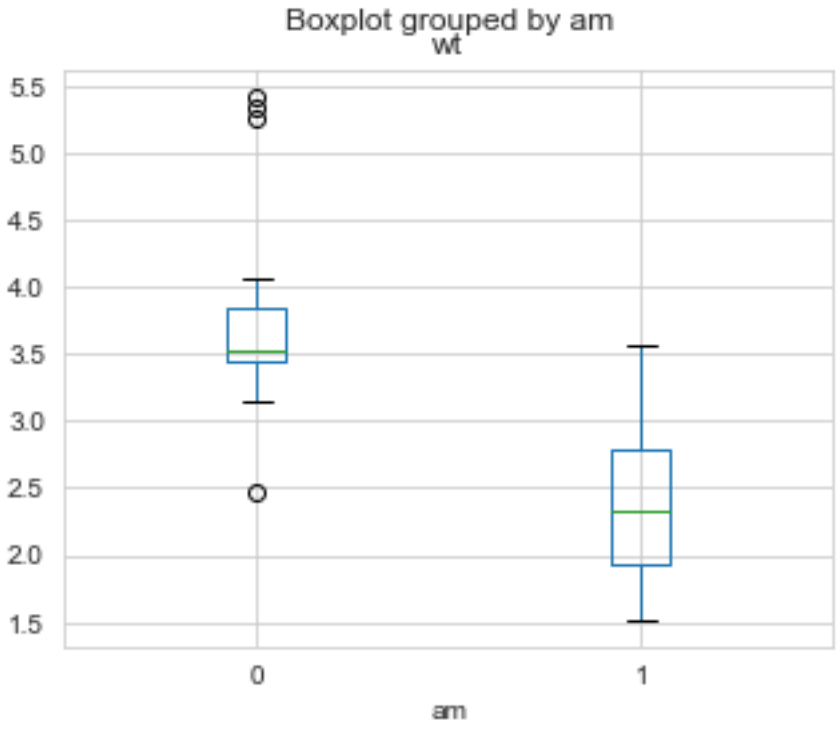


1.5.4 Building boxplots

[57]:
```
cars.boxplot(column='mpg', by='am')
cars.boxplot(column='wt', by='am')
```

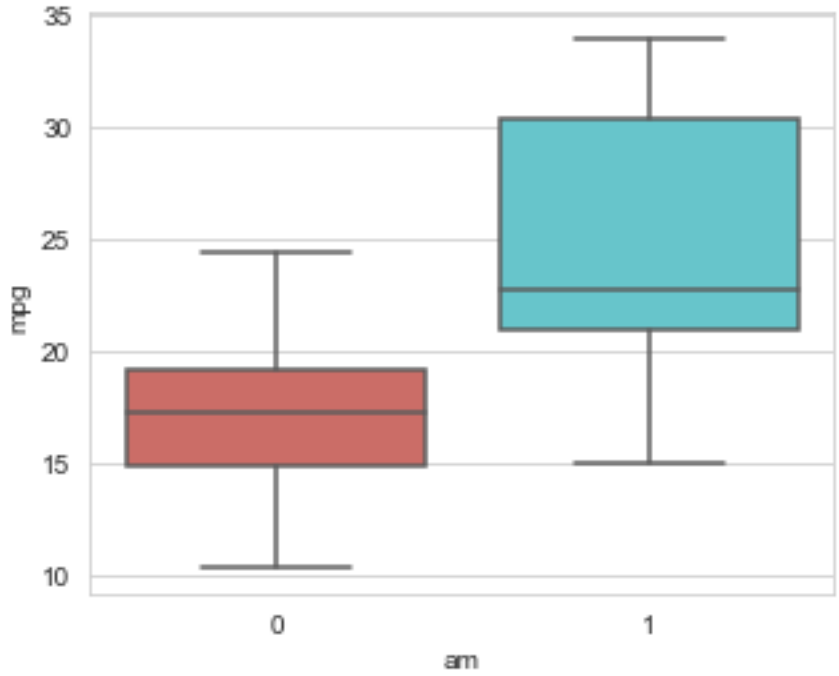
[57]:
```
<AxesSubplot:title={'center':'wt'}, xlabel='am'>
```


Boxplot grouped by am
mpg
35
30
25
20
15
10
0
1
am


Boxplot grouped by am
wt
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
0
1
am

[58]:
```
sb.boxplot(x='am', y='mpg', data=cars, palette='hls')
```

[58]: <AxesSubplot:xlabel='am', ylabel='mpg'>



[ ]:
```

```

chapter5

March 2, 2022

1 Chapter 5 - Basic Math and Statistics

1.1 Segment 1 - Using NumPy to perform arithmetic operations on data

[1]:
```
import numpy as np
from numpy.random import randn
```

[2]:
```
np.set_printoptions(precision=2)
```

1.2 Creating arrays

1.2.1 Creating arrays using a list

[3]:
```
a = np.array([1,2,3,4,5,6])
a
```

[3]:
```
array([1, 2, 3, 4, 5, 6])
```

[4]:
```
b = np.array([[10,20,20],[40,50,60]])
b
```

[4]:
```
array([[10, 20, 20],
       [40, 50, 60]])
```

1.2.2 Creating arrays via assignment

[5]:
```
np.random.seed(25)
c = 36*np.random.randn(6)
c
```

[5]:
```
array([  8.22,  36.97, -30.23, -21.28, -34.45,  -8.  ])
```

[6]:
```
d = np.arange(1, 35)
d
```

[6]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34])
```

1.3 Performing arthimetic on arrays

[7]:
```
a*10
```

[7]:
```
array([10, 20, 30, 40, 50, 60])
```

[8]:
```
c + a
```

[8]:
```
array([  9.22,  38.97, -27.23, -17.28, -29.45,  -2.  ])
```

[9]:
```
c-a
```

[9]:
```
array([  7.22,  34.97, -33.23, -25.28, -39.45, -14.  ])
```

[10]:
```
c*a
```

[10]:
```
array([   8.22,   73.94,  -90.68,  -85.13, -172.24,  -48.02])
```

[11]:
```
c/a
```

[11]:
```
array([  8.22,  18.48, -10.08,  -5.32,  -6.89,  -1.33])
```

1.3.1 Multiplying matrices and basic linear algebra

[12]:
```
aa = np.array([[2.,4.,6.],[1.,3.,5.],[10.,20.,30.]])
aa
```

[12]:
```
array([[ 2.,  4.,  6.],
       [ 1.,  3.,  5.],
       [10., 20., 30.]])
```

[13]:
```
bb = np.array([[0.,1.,2.],[3.,4.,5.],[6.,7.,8.]])
bb
```

[13]:
```
array([[0., 1., 2.],
       [3., 4., 5.],
       [6., 7., 8.]])
```

[14]:
```
aa*bb
```

[14]:
```
array([[  0.,   4.,  12.],
       [  3.,  12.,  25.],
       [ 60., 140., 240.]])
```

[15]:
```
np.dot(aa,bb)
```

[15]:
```
array([[ 48.,  60.,  72.],
       [ 39.,  48.,  57.],
       [240., 300., 360.]])
```

1.4 Segment 2 - Multiplying matrices and basic linear algebra

1.5 Multiplying matrices and basic linear algebra

[16]:
```
aa = np.array([[2.,4.,6.],[1.,3.,5.],[10.,20.,30.]])
aa
```

[16]:
```
array([[ 2.,  4.,  6.],
       [ 1.,  3.,  5.],
       [10., 20., 30.]])
```

[17]:
```
bb = np.array([[0.,1.,2.],[3.,4.,5.],[6.,7.,8.]])
bb
```

[17]:
```
array([[0., 1., 2.],
       [3., 4., 5.],
       [6., 7., 8.]])
```

[18]:
```
aa*bb
```

[18]:
```
array([[  0.,   4.,  12.],
       [  3.,  12.,  25.],
       [ 60., 140., 240.]])
```

[19]:
```
np.dot(aa,bb)
```

[19]:
```
array([[ 48.,  60.,  72.],
       [ 39.,  48.,  57.],
       [240., 300., 360.]])
```

1.6 Segment 3 - Generating summary statistics using pandas and scipy

[22]:
```
import numpy as np
import pandas as pd
from pandas import Series, DataFrame

import scipy
from scipy import stats
```

[23]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns =␣
 ↪['car_names','mpg','cyl','disp','hp','drat','wt','qsec','vs','am','gear','carb']

cars.head()
```

[23]:
```
           car_names   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  \
0          Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1     4
1      Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1     4
2         Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1     4
3     Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0     3
4  Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0     3

   carb
0     4
1     4
2     1
3     1
4     2
```

[24]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns =␣
 ↪['car_names','mpg','cyl','disp','hp','drat','wt','qsec','vs','am','gear','carb']

cars.head()
```

[24]:
```
           car_names   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  \
0          Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1     4
1      Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1     4
2         Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1     4
3     Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0     3
4  Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0     3

   carb
0     4
1     4
2     1
3     1
4     2
```

1.6.1 Looking at summary statistics that decribe a variable's numeric values

[26]:
```
cars.sum()
```

[26]:
```
car_names    Mazda RX4Mazda RX4 WagDatsun 710Hornet 4 Drive…
mpg                                                    642.9
cyl                                                      198
disp                                                  7383.1
hp                                                      4694
drat                                                  115.09
wt                                                   102.952
```

```
qsec                                                  571.16
vs                                                        14
am                                                        13
gear                                                     118
carb                                                      90
dtype: object
```

[27]:
```
cars.sum(axis=1)
```

[27]:
```
0     328.980
1     329.795
2     259.580
3     426.135
4     590.310
5     385.540
6     656.920
7     270.980
8     299.570
9     350.460
10    349.660
11    510.740
12    511.500
13    509.850
14    728.560
15    726.644
16    725.695
17    213.850
18    195.165
19    206.955
20    273.775
21    519.650
22    506.085
23    646.280
24    631.175
25    208.215
26    272.570
27    273.683
28    670.690
29    379.590
30    694.710
31    288.890
dtype: float64
```

[28]:
```
cars.median()
```

[28]:
```
mpg      19.200
cyl       6.000
```

```
disp    196.300
hp      123.000
drat      3.695
wt        3.325
qsec     17.710
vs        0.000
am        0.000
gear      4.000
carb      2.000
dtype: float64
```

[29]:
```
cars.mean()
```

[29]:
```
mpg      20.090625
cyl       6.187500
disp    230.721875
hp      146.687500
drat      3.596563
wt        3.217250
qsec     17.848750
vs        0.437500
am        0.406250
gear      3.687500
carb      2.812500
dtype: float64
```

[30]:
```
cars.max()
```

[30]:
```
car_names    Volvo 142E
mpg                33.9
cyl                   8
disp              472.0
hp                  335
drat               4.93
wt                5.424
qsec               22.9
vs                    1
am                    1
gear                  5
carb                  8
dtype: object
```

[31]:
```
mpg = cars.mpg
mpg.idxmax()
```

[31]:
```
19
```

1.6.2 Looking at summary statistics that describe variable distribution

[33]:
```
cars.std()
```

[33]:
```
mpg       6.026948
cyl       1.785922
disp    123.938694
hp       68.562868
drat      0.534679
wt        0.978457
qsec      1.786943
vs        0.504016
am        0.498991
gear      0.737804
carb      1.615200
dtype: float64
```

[34]:
```
cars.var()
```

[34]:
```
mpg        36.324103
cyl         3.189516
disp    15360.799829
hp       4700.866935
drat        0.285881
wt          0.957379
qsec        3.193166
vs          0.254032
am          0.248992
gear        0.544355
carb        2.608871
dtype: float64
```

[35]:
```
gear = cars.gear
gear.value_counts()
```

[35]:
```
3    15
4    12
5     5
Name: gear, dtype: int64
```

[36]:
```
cars.describe()
```

[36]:
```
              mpg        cyl        disp          hp       drat         wt  \
count  32.000000  32.000000   32.000000   32.000000  32.000000  32.000000
mean   20.090625   6.187500  230.721875  146.687500   3.596563   3.217250
std     6.026948   1.785922  123.938694   68.562868   0.534679   0.978457
min    10.400000   4.000000   71.100000   52.000000   2.760000   1.513000
25%    15.425000   4.000000  120.825000   96.500000   3.080000   2.581250
```

```
50%    19.200000   6.000000  196.300000  123.000000   3.695000   3.325000
75%    22.800000   8.000000  326.000000  180.000000   3.920000   3.610000
max    33.900000   8.000000  472.000000  335.000000   4.930000   5.424000

            qsec         vs         am       gear     carb
count  32.000000  32.000000  32.000000  32.000000  32.0000
mean   17.848750   0.437500   0.406250   3.687500   2.8125
std     1.786943   0.504016   0.498991   0.737804   1.6152
min    14.500000   0.000000   0.000000   3.000000   1.0000
25%    16.892500   0.000000   0.000000   3.000000   2.0000
50%    17.710000   0.000000   0.000000   4.000000   2.0000
75%    18.900000   1.000000   1.000000   4.000000   4.0000
max    22.900000   1.000000   1.000000   5.000000   8.0000
```

1.7 Segment 4 - Summarizing categorical data using pandas

1.7.1 The basics

[37]:
```
address = './Data/mtcars.csv'
cars = pd.read_csv(address)

cars.columns =␣
 ↪['car_names','mpg','cyl','disp','hp','drat','wt','qsec','vs','am','gear','carb']
cars.index = cars.car_names
cars.head(15)
```

[37]:

	car_names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
car_names												
Mazda RX4	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	Datsun 710	22.8	4	108.0	93	3.85	2.320					
Hornet 4 Drive	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215					
Hornet Sportabout	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440					
Valiant	Valiant	18.1	6	225.0	105	2.76	3.460					
Duster 360	Duster 360	14.3	8	360.0	245	3.21	3.570					
Merc 240D	Merc 240D	24.4	4	146.7	62	3.69	3.190					
Merc 230	Merc 230	22.8	4	140.8	95	3.92	3.150					
Merc 280	Merc 280	19.2	6	167.6	123	3.92	3.440					
Merc 280C	Merc 280C	17.8	6	167.6	123	3.92	3.440					
Merc 450SE	Merc 450SE	16.4	8	275.8	180	3.07	4.070					
Merc 450SL	Merc 450SL	17.3	8	275.8	180	3.07	3.730					
Merc 450SLC	Merc 450SLC	15.2	8	275.8	180	3.07	3.780					
Cadillac Fleetwood	Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250					

```
Datsun 710          18.61   1   1     4     1
Hornet 4 Drive      19.44   1   0     3     1
Hornet Sportabout   17.02   0   0     3     2
Valiant             20.22   1   0     3     1
Duster 360          15.84   0   0     3     4
Merc 240D           20.00   1   0     4     2
Merc 230            22.90   1   0     4     2
Merc 280            18.30   1   0     4     4
Merc 280C           18.90   1   0     4     4
Merc 450SE          17.40   0   0     3     3
Merc 450SL          17.60   0   0     3     3
Merc 450SLC         18.00   0   0     3     3
Cadillac Fleetwood  17.98   0   0     3     4
```

[38]:
```
carb = cars.carb
carb.value_counts()
```

[38]:
```
2    10
4    10
1     7
3     3
6     1
8     1
Name: carb, dtype: int64
```

[41]:
```
cars_cat = cars[['cyl','vs','am','gear','carb']]
cars_cat.head()
```

[41]:
```
                  cyl  vs  am  gear  carb
car_names
Mazda RX4           6   0   1     4     4
Mazda RX4 Wag       6   0   1     4     4
Datsun 710          4   1   1     4     1
Hornet 4 Drive      6   1   0     3     1
Hornet Sportabout   8   0   0     3     2
```

[114]:
```
gears_group = cars_cat.groupby('gear')
gears_group.describe()
```

[114]:
```
       vs                                        am       … gear      carb       \
    count mean std min 25% 50% 75% max count mean  …  75% max count mean
cyl                                                      …
4    11.0  0.9 0.3 0.0 1.0 1.0 1.0 1.0  11.0  0.7  …  4.0 5.0  11.0  1.5
6     7.0  0.6 0.5 0.0 0.0 1.0 1.0 1.0   7.0  0.4  …  4.0 5.0   7.0  3.4
8    14.0  0.0 0.0 0.0 0.0 0.0 0.0 0.0  14.0  0.1  …  3.0 5.0  14.0  3.5
```

```
    std min 25% 50% 75% max
cyl
4   0.5 1.0 1.0 2.0 2.0 2.0
6   1.8 1.0 2.5 4.0 4.0 6.0
8   1.6 2.0 2.2 3.5 4.0 8.0

[3 rows x 32 columns]
```

1.7.2 Transforming variables to categorical data type

[43]:
```
cars['group'] = pd.Series(cars.gear, dtype="category")
```

[44]:
```
cars['group'].dtypes
```

[44]:
```
CategoricalDtype(categories=[3, 4, 5], ordered=False)
```

[45]:
```
cars['group'].value_counts()
```

[45]:
```
3    15
4    12
5     5
Name: group, dtype: int64
```

1.7.3 Describing categorical data with crosstabs

[115]:
```
pd.crosstab(cars['vs'], cars['cyl'])
```

[115]:
```
cyl   4  6   8
vs
0     1  3  14
1    10  4   0
```

1.8 Segment 5 - Starting with parametric methods in pandas and scipy

[47]:
```
import matplotlib.pyplot as plt
import seaborn as sb
from pylab import rcParams

import scipy
from scipy.stats.stats import pearsonr
```

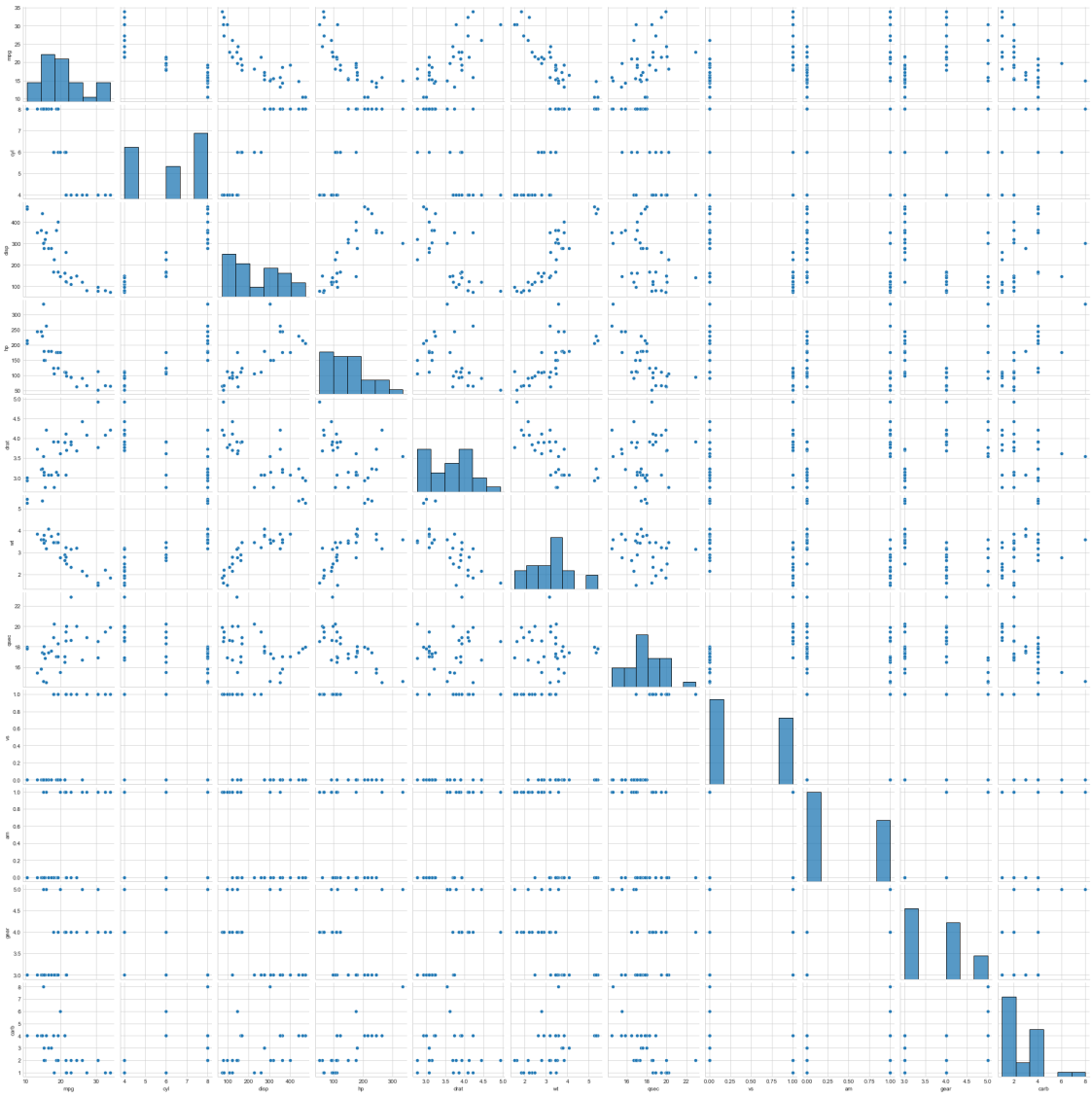
[48]:
```
%matplotlib inline
rcParams['figure.figsize'] = 8,4
plt.style.use('seaborn-whitegrid')
```

1.8.1 The Pearson Correlation

[50]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns =␣
 ↪['car_names','mpg','cyl','disp','hp','drat','wt','qsec','vs','am','gear','carb']
```

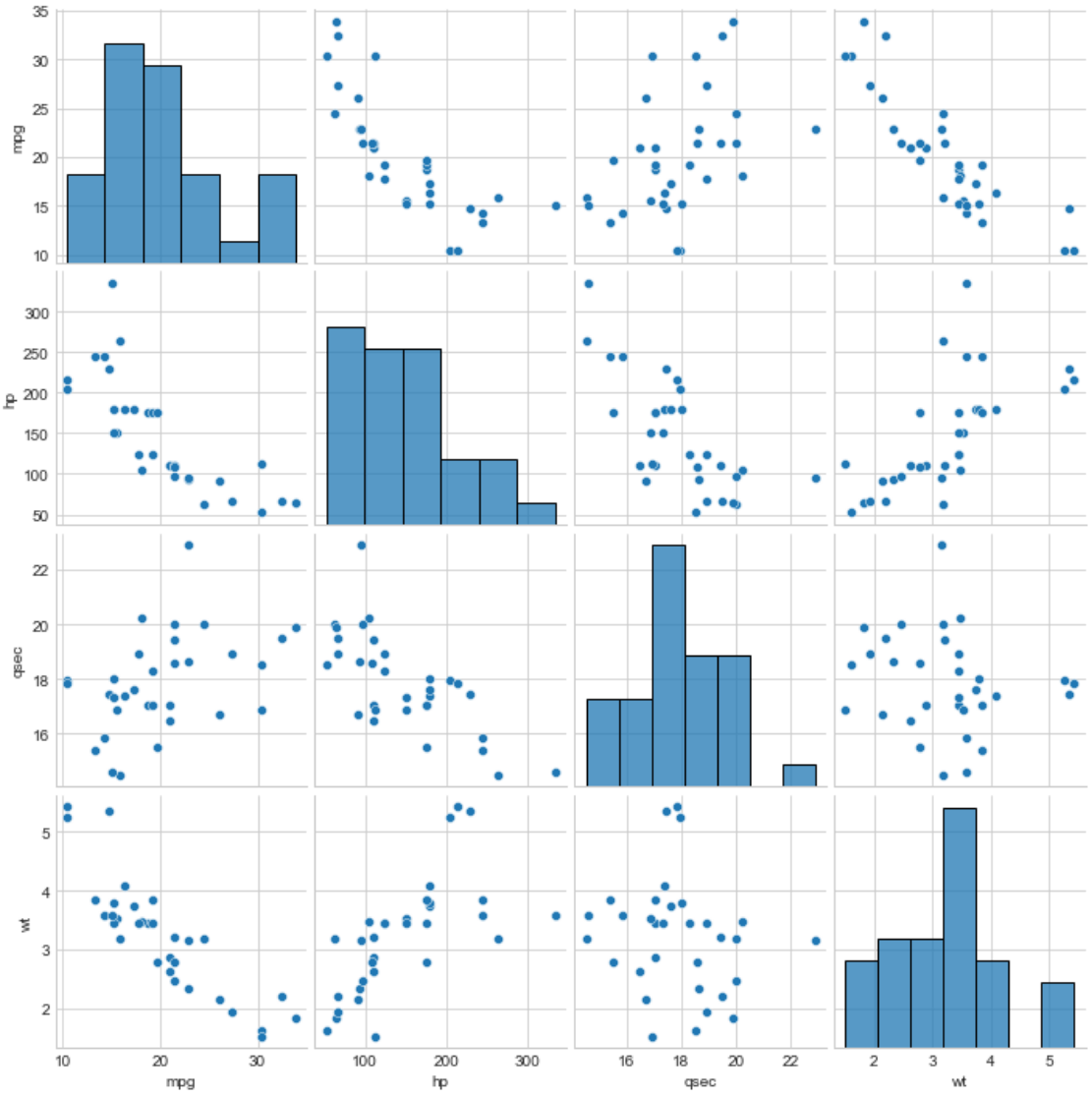
[51]:
```
sb.pairplot(cars)
```

[51]: <seaborn.axisgrid.PairGrid at 0x1f9674c6610>



[52]:
```
X = cars[['mpg', 'hp', 'qsec', 'wt']]
sb.pairplot(X)
```

[52]:
```
<seaborn.axisgrid.PairGrid at 0x1f96bebfb80>
```



1.8.2 Using scipy to calculate the Pearson correlation coefficient

[53]:
```
mpg = cars['mpg']
hp = cars['hp']
qsec = cars['qsec']
wt = cars['wt']

pearsonr_coefficient, p_value = pearsonr(mpg, hp)
```

```
print('PeasonR Correlation Coefficient %0.3f'% (pearsonr_coefficient))
```

```
PeasonR Correlation Coefficient -0.776
```

[54]:
```
pearsonr_coefficient, p_value = pearsonr(mpg, qsec)
print('PeasonR Correlation Coefficient %0.3f'% (pearsonr_coefficient))
```

```
PeasonR Correlation Coefficient 0.419
```

[55]:
```
pearsonr_coefficient, p_value = pearsonr(mpg, wt)
print('PeasonR Correlation Coefficient %0.3f'% (pearsonr_coefficient))
```

```
PeasonR Correlation Coefficient -0.868
```

1.8.3 Using pandas to calculate the Pearson correlation coefficient

[57]:
```
corr = X.corr()
corr
```

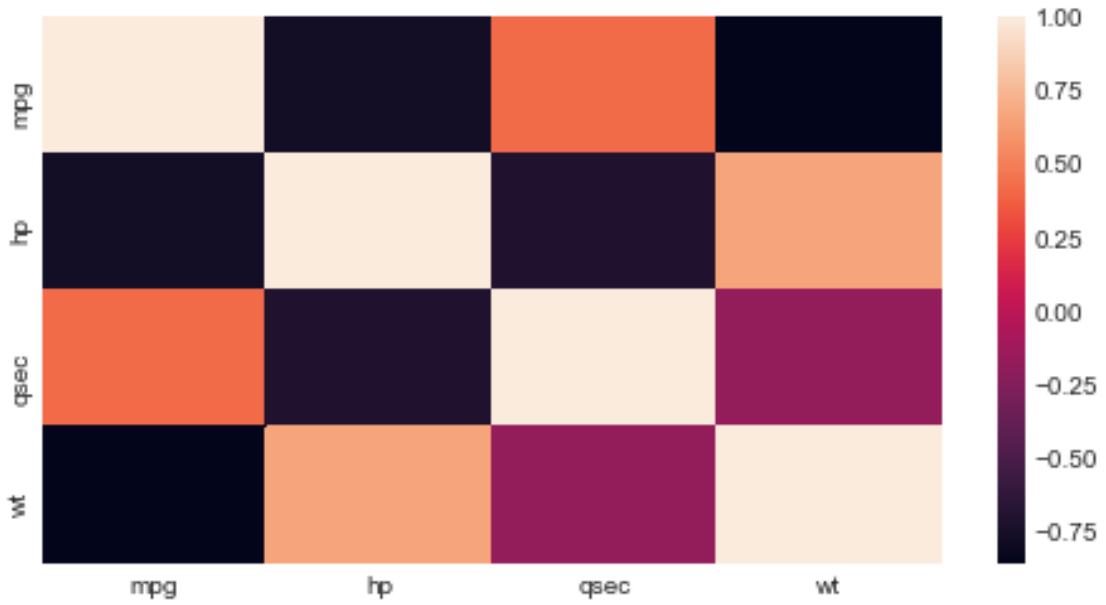
[57]:

	mpg	hp	qsec	wt
mpg	1.000000	-0.776168	0.418684	-0.867659
hp	-0.776168	1.000000	-0.708223	0.658748
qsec	0.418684	-0.708223	1.000000	-0.174716
wt	-0.867659	0.658748	-0.174716	1.000000

1.8.4 Using Seaborn to visualize the Pearson correlation coefficient

[58]:
```
sb.heatmap(corr, xticklabels=corr.columns.values, yticklabels= corr.columns.
↪values)
```

[58]:
```
<AxesSubplot:>
```



1.9 Segment 6 - Delving into non-parametric methods using pandas and scipy

[59]:
```
import scipy
from scipy.stats import spearmanr
```

[60]:
```
%matplotlib inline
rcParams['figure.figsize'] = 14, 7
plt.style.use('seaborn-whitegrid')
```

1.9.1 The Spearman Rank Correlation

[62]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']
```

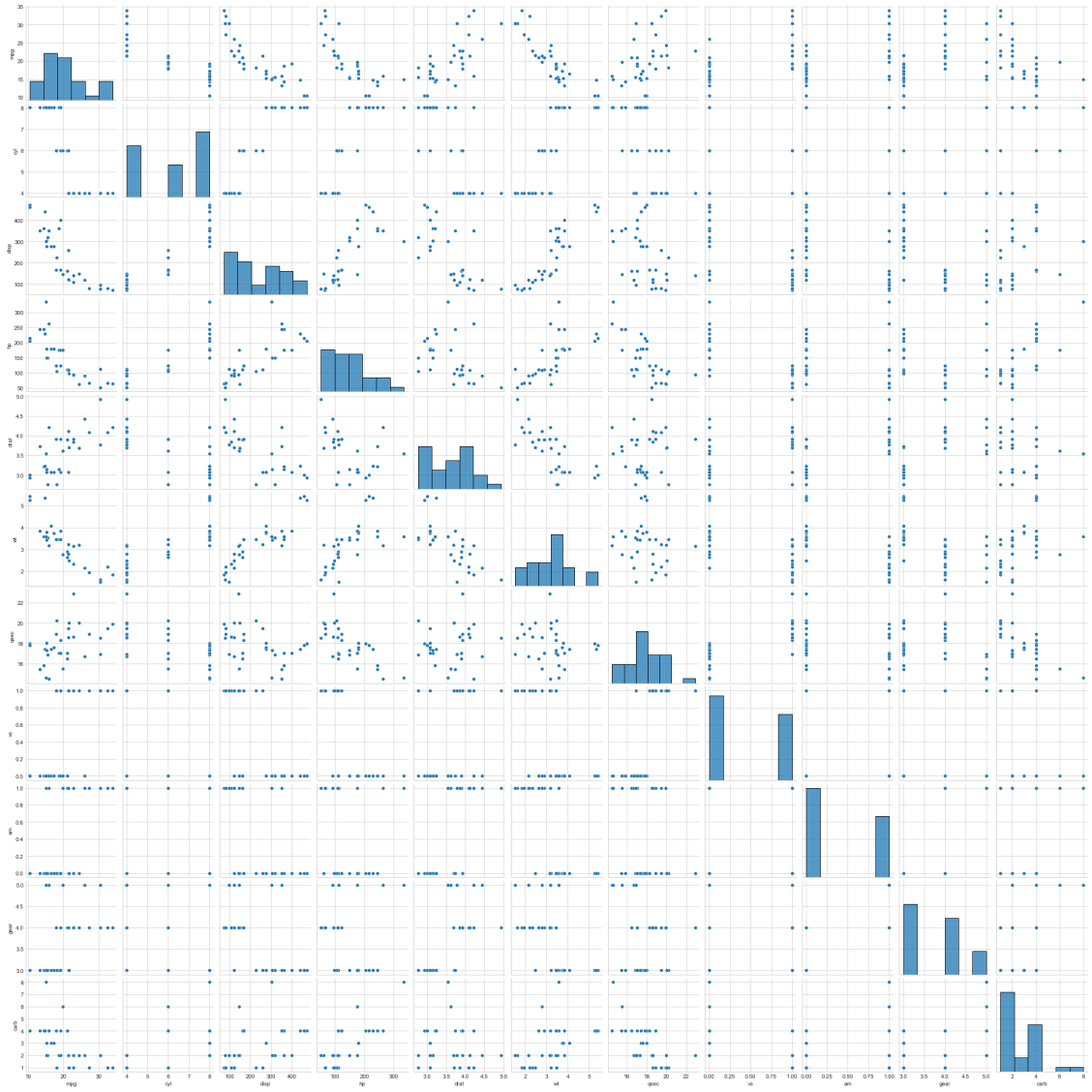
[63]:
```
cars.head()
```

[63]:

	car_names	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2

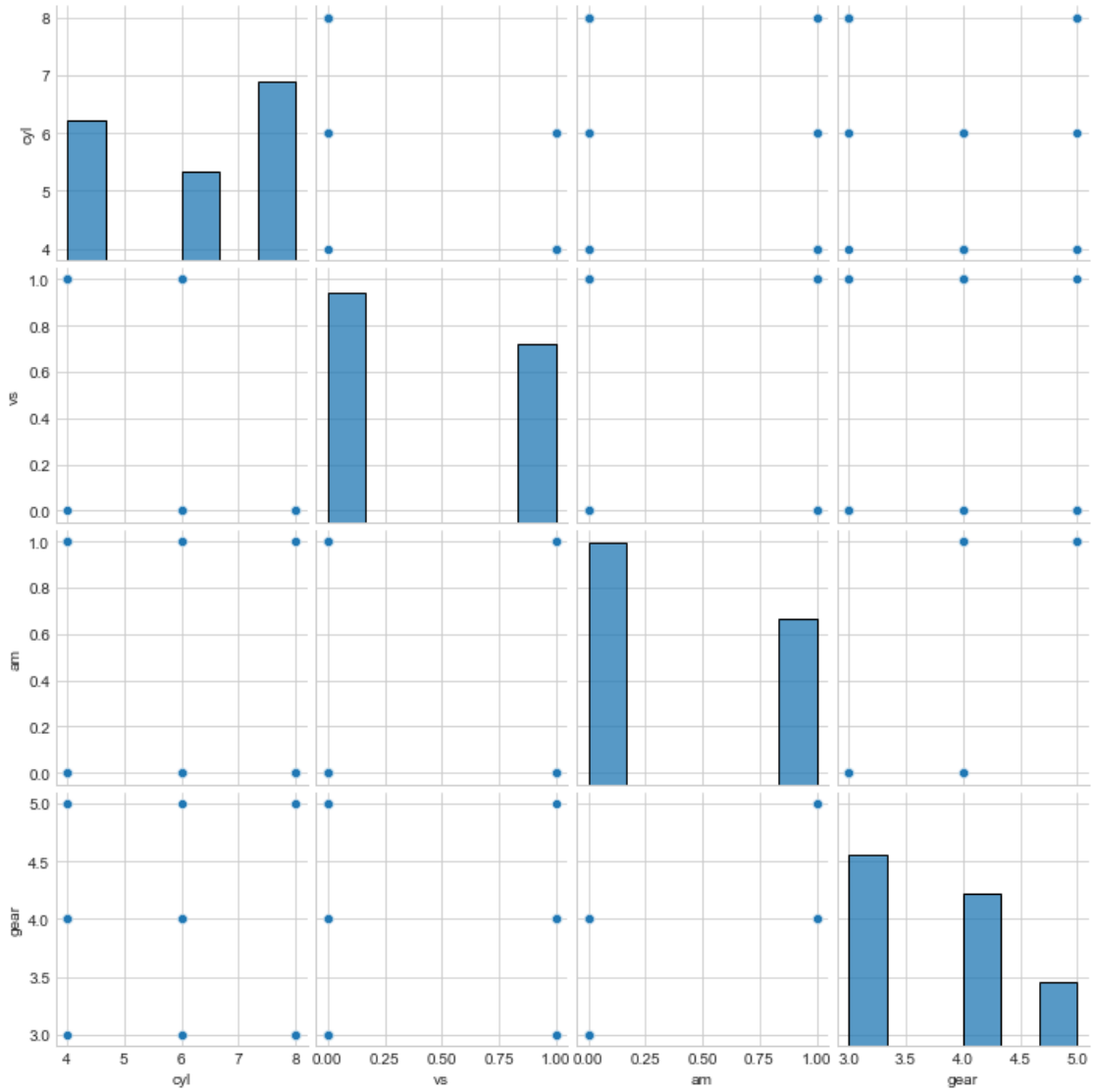
[64]:
```
sb.pairplot(cars)
```

[64]:
```
<seaborn.axisgrid.PairGrid at 0x1f96f2d5dc0>
```



[65]:
```
X = cars[['cyl', 'vs', 'am', 'gear']]
sb.pairplot(X)
```

[65]: <seaborn.axisgrid.PairGrid at 0x1f9747721c0>



[68]:
```
cyl = cars['cyl']
vs = cars['vs']
am = cars['am']
gear = cars['gear']

spearmanr_coefficient, p_value = spearmanr(cyl, vs)
print('Spearman Rank Correlation Coefficient %0.3f' % (spearmanr_coefficient))
```

```
Spearman Rank Correlation Coefficient -0.814
```

[69]:
```
spearmanr_coefficient, p_value = spearmanr(cyl, am)
print('Spearman Rank Correlation Coefficient %0.3f' % (spearmanr_coefficient))
```

```
Spearman Rank Correlation Coefficient -0.522
```

[70]:
```
spearmanr_coefficient, p_value = spearmanr(cyl, gear)
print('Spearman Rank Correlation Coefficient %0.3f' % (spearmanr_coefficient))
```

```
Spearman Rank Correlation Coefficient -0.564
```

1.9.2 Chi-square test for independence

[72]:
```
table = pd.crosstab(cyl, am)

from scipy.stats import chi2_contingency
chi2, p, dof, expected = chi2_contingency(table.values)
print ('Chi-square statistic %0.3f p_value %0.3f' % (chi2, p))
```

```
Chi-square statistic 8.741 p_value 0.013
```

[73]:
```
table = pd.crosstab(cyl, vs)

from scipy.stats import chi2_contingency
chi2, p, dof, expected = chi2_contingency(table.values)
print ('Chi-square statistic %0.3f p_value %0.3f' % (chi2, p))
```

```
Chi-square statistic 21.340 p_value 0.000
```

[74]:
```
table = pd.crosstab(cyl, gear)

from scipy.stats import chi2_contingency
chi2, p, dof, expected = chi2_contingency(table.values)
print ('Chi-square statistic %0.3f p_value %0.3f' % (chi2, p))
```

```
Chi-square statistic 18.036 p_value 0.001
```

1.10 Segment 7 - Transforming dataset distributions

[76]:
```
import sklearn
from sklearn import preprocessing
from sklearn.preprocessing import scale
```

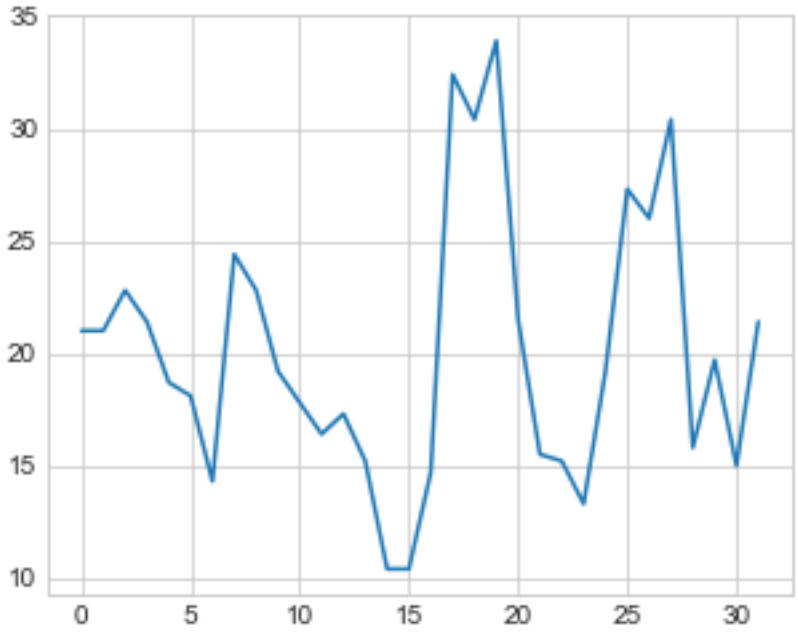
[77]:
```
%matplotlib inline
rcParams['figure.figsize'] = 5, 4
sb.set_style('whitegrid')
```

1.10.1 Normalizing and transforming features with MinMaxScalar() and fit_transform()

[78]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
↪'vs', 'am', 'gear', 'carb']
```

[79]:
```
mpg = cars.mpg
plt.plot(mpg)
```

[79]:
```
[<matplotlib.lines.Line2D at 0x1f976aac730>]
```



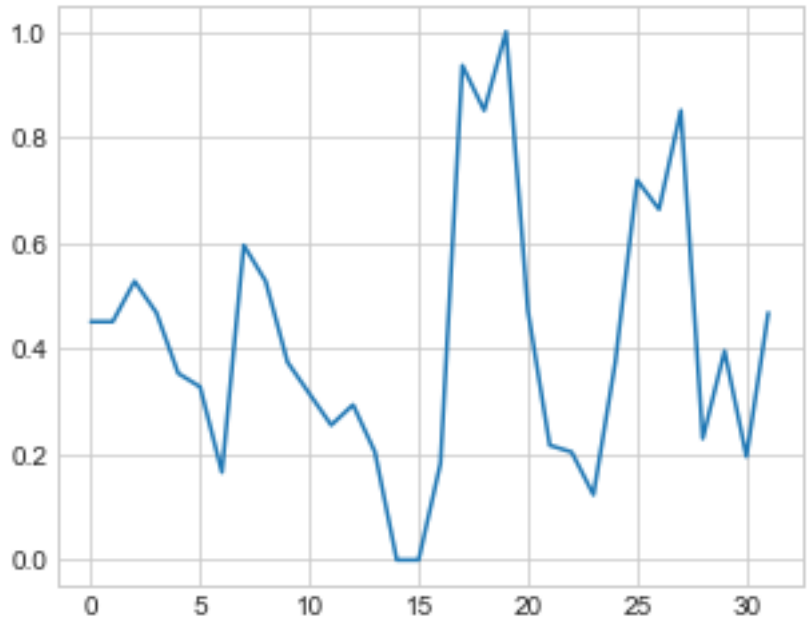
[80]:
```
cars[['mpg']].describe()
```

[80]:

	mpg
count	32.000000
mean	20.090625
std	6.026948
min	10.400000
25%	15.425000
50%	19.200000
75%	22.800000
max	33.900000

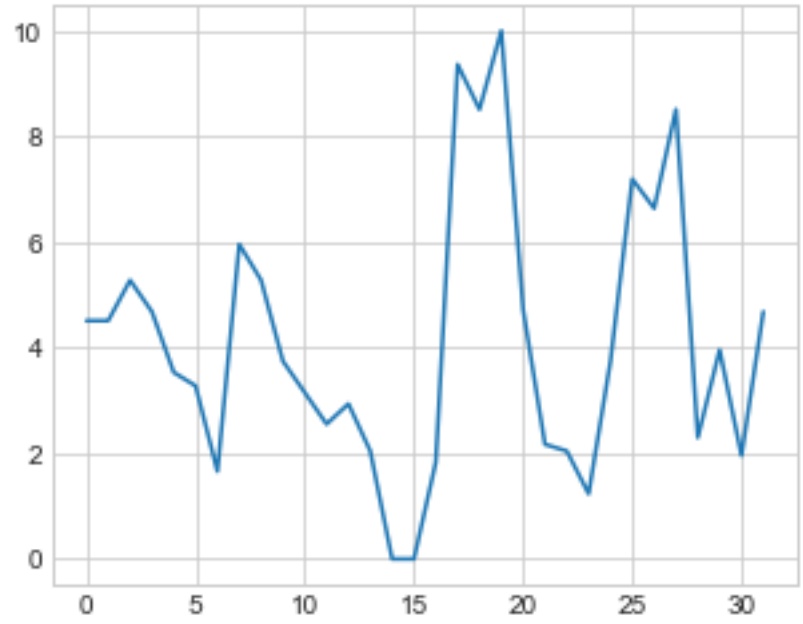
[81]:
```
mpg_matrix = mpg.values.reshape(-1,1)

scaled = preprocessing.MinMaxScaler()

scaled_mpg = scaled.fit_transform(mpg_matrix)
plt.plot(scaled_mpg)
```

[81]:
```
[<matplotlib.lines.Line2D at 0x1f976af6430>]
```



[82]:
```
scaled = preprocessing.MinMaxScaler(feature_range=(0,10))

scaled_mpg = scaled.fit_transform(mpg_matrix)
plt.plot(scaled_mpg)
```

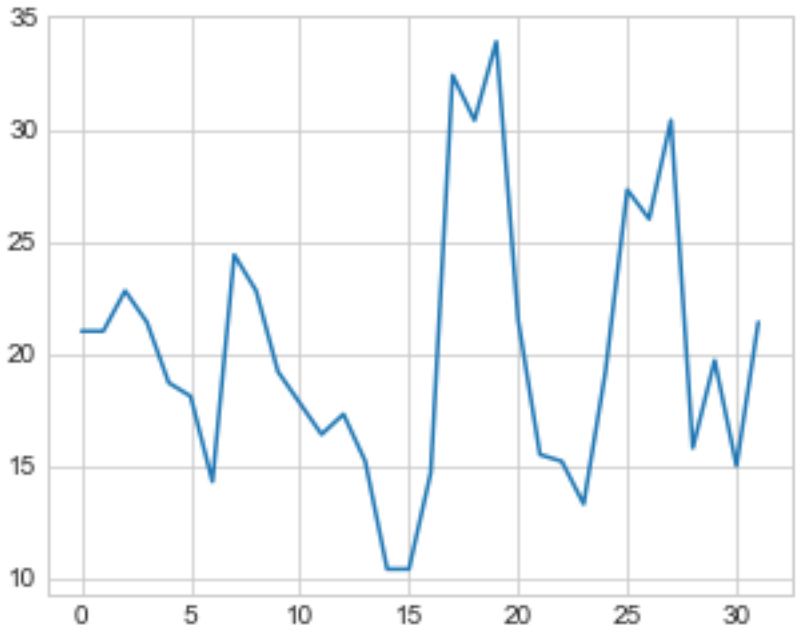
[82]: [<matplotlib.lines.Line2D at 0x1f976b2bcd0>]



1.10.2 Using scale() to scale your features

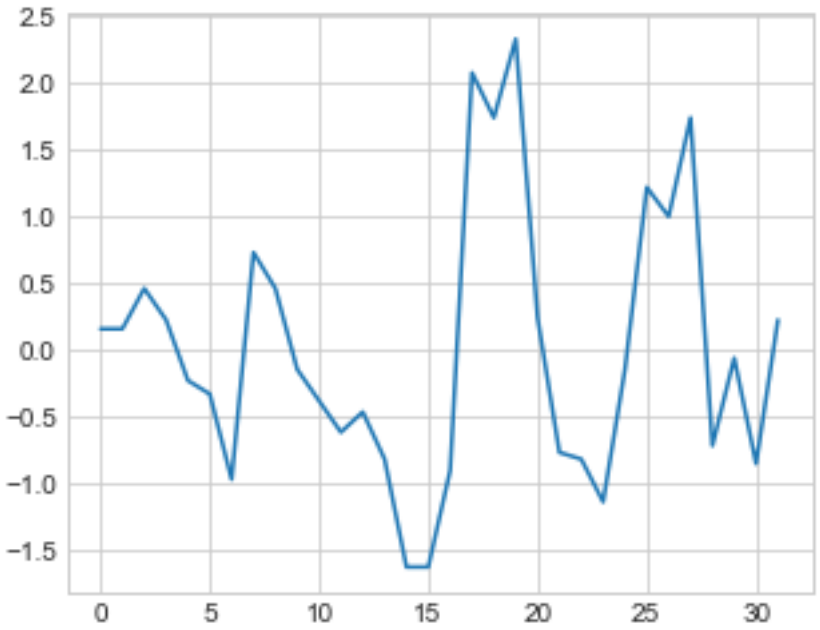
[83]:
```
standardized_mpg = scale(mpg, axis=0, with_mean=False, with_std=False)
plt.plot(standardized_mpg)
```

[83]:
```
[<matplotlib.lines.Line2D at 0x1f977b50820>]
```



[84]:
```
standardized_mpg = scale(mpg)
plt.plot(standardized_mpg)
```

[84]:
```
[<matplotlib.lines.Line2D at 0x1f977ba1ee0>]
```



[85]:
```
address = './Data/iris.data.csv'
df = pd.read_csv(filepath_or_buffer=address, header=None, sep=',')

df.columns=['Sepal Length','Sepal Width','Petal Length','Petal Width',␣
 ↪'Species']
```

[86]:
```
X = df.iloc[:,0:4].values
y = df.iloc[:,4].values
df[:5]
```

[86]:

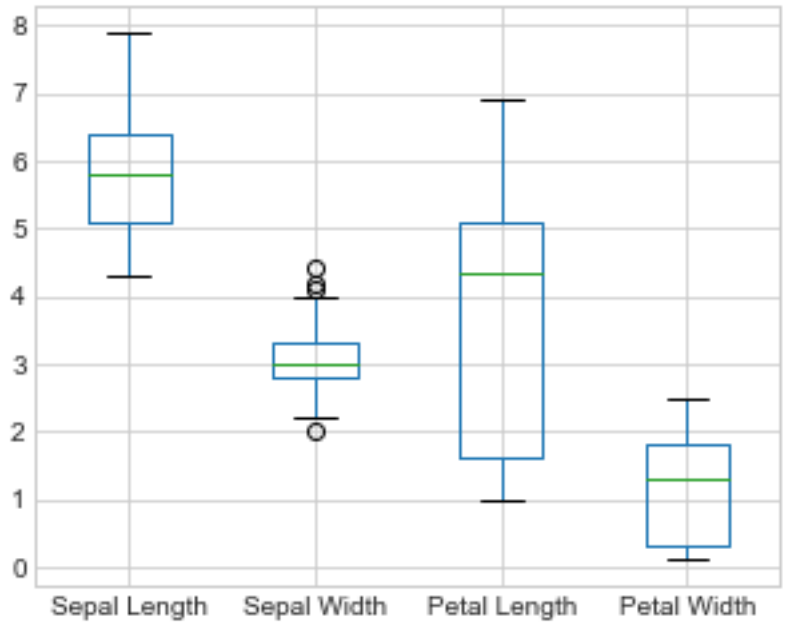
	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

1.11 Segment 8 - Extreme value analysis using univariate methods

1.11.1 Identifying outliers from Tukey boxplots

[88]:
```
df.boxplot(return_type='dict')
plt.plot()
```

[88]: []



[89]:
```
Sepal_Width = X[:,1]
iris_outliers = (Sepal_Width > 4)
df[iris_outliers]
```

[89]:

	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
15	5.7	4.4	1.5	0.4	setosa
32	5.2	4.1	1.5	0.1	setosa
33	5.5	4.2	1.4	0.2	setosa

[90]:
```
Sepal_Width = X[:,1]
iris_outliers = (Sepal_Width < 2.05)
df[iris_outliers]
```

[90]:

	Sepal Length	Sepal Width	Petal Length	Petal Width	Species
60	5.0	2.0	3.5	1.0	versicolor

1.11.2 Applying Tukey outlier labeling

[91]:
```
pd.options.display.float_format = '{:.1f}'.format
X_df = pd.DataFrame(X)
print(X_df.describe())
```

```
          0     1     2     3
count 150.0 150.0 150.0 150.0
mean    5.8   3.1   3.8   1.2
std     0.8   0.4   1.8   0.8
min     4.3   2.0   1.0   0.1
```

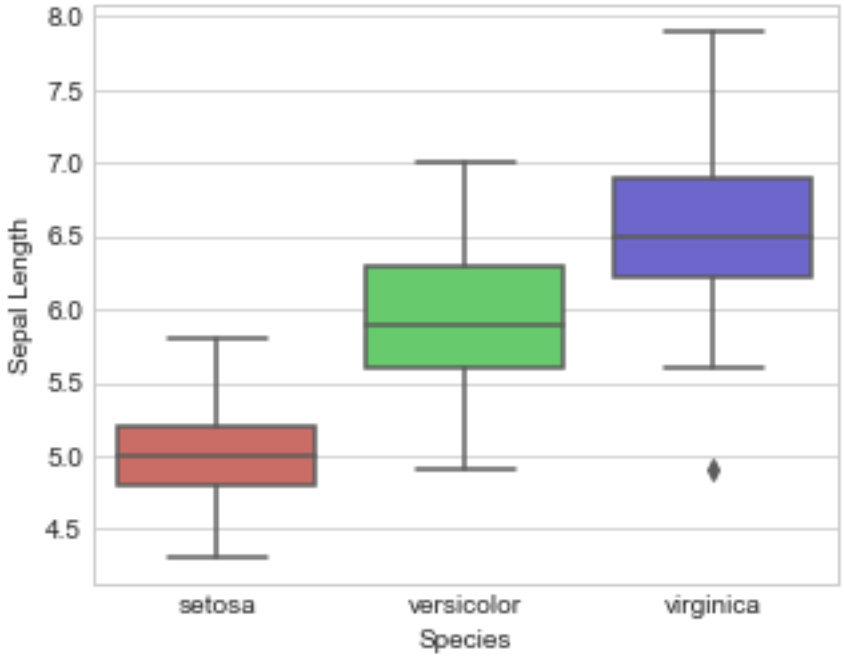
```
25%     5.1   2.8   1.6   0.3
50%     5.8   3.0   4.3   1.3
75%     6.4   3.3   5.1   1.8
max     7.9   4.4   6.9   2.5
```

1.12 Segment 9 - Multivariate analysis for outlier detection

[94]:
```
df = pd.read_csv(filepath_or_buffer='./Data/iris.data.csv', header=None,␣
↪sep=',')

df.columns=['Sepal Length','Sepal Width','Petal Length','Petal Width',␣
↪'Species']
```

[95]:
```
data = df.iloc[:,0:4].values
target = df.iloc[:,4].values
df[:5]
sb.boxplot(x='Species', y='Sepal Length', data=df, palette='hls')
```

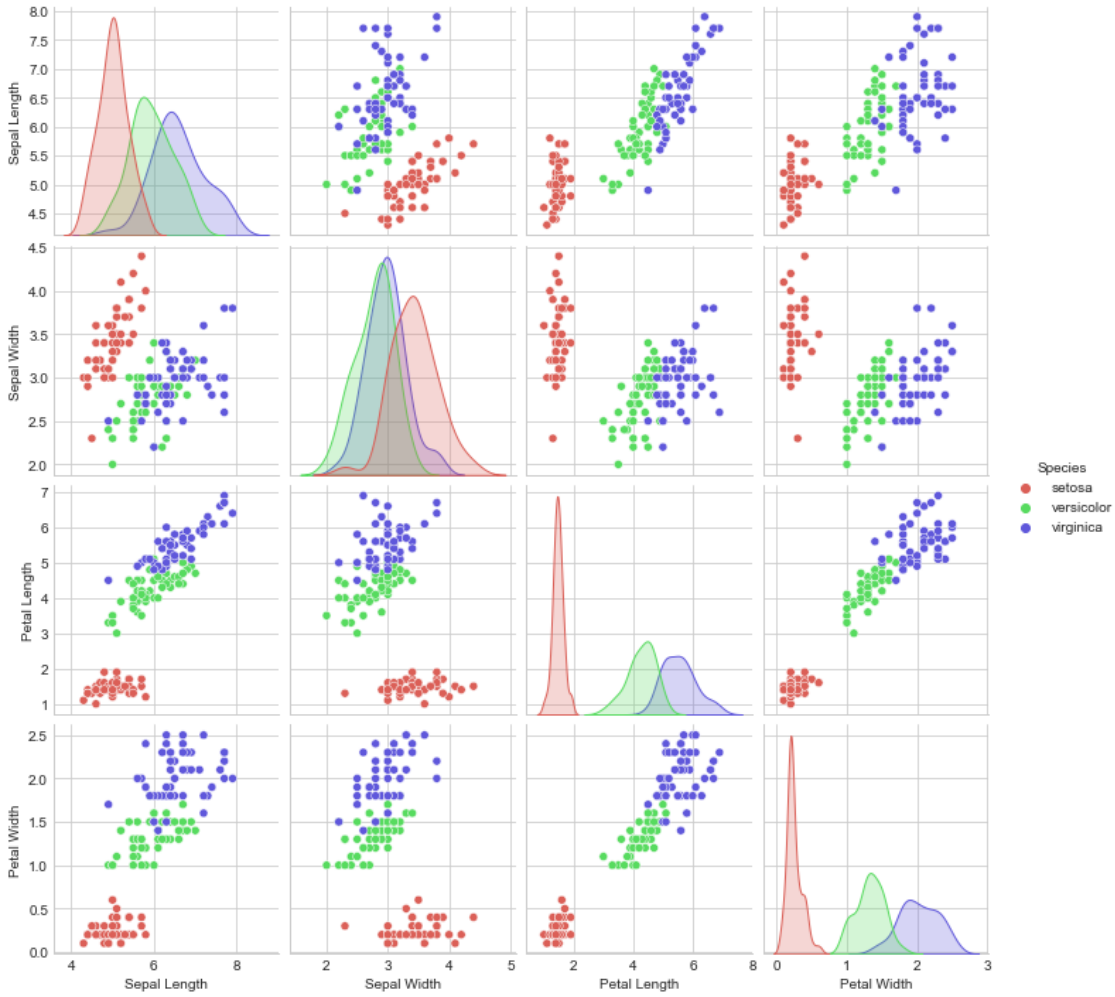
[95]: <AxesSubplot:xlabel='Species', ylabel='Sepal Length'>



1.12.1 Looking at the scatterplot matrix

[97]:
```
sb.pairplot(df, hue='Species', palette='hls')
```

[97]: <seaborn.axisgrid.PairGrid at 0x1f977d736d0>



[99]:
```
xx=np.array([[7.,9.],[5.,12.]])
yy=np.array([[2.,8.],[7.,4.]])
np.dot(xx,yy)
```

[99]:
```
array([[77., 92.],
       [94., 88.]])
```

[102]:
```
xx=np.array([[1.,2.,3.],[4.,5.,6.]])
yy=np.array([[10.,11.],[20.,21.],[30.,31.]])
np.dot(xx,yy)
```

[102]:
```
array([[140., 146.],
       [320., 335.]])
```

[103]:
```
a=np.array([1,8,2,6,3,8,5,5,5,5])
b=np.array([17,16,20,18,22,15,21,15,17,22])
```

```
(a+b)/10
```

[103]:
```
array([1.8, 2.4, 2.2, 2.4, 2.5, 2.3, 2.6, 2. , 2.2, 2.7])
```

[106]:
```
a=np.array([10, 15, 20])
b=([5, 7, 9])
(a-b)*7
```

[106]:
```
array([35, 56, 77])
```

[109]:
```
Q1= 1.714
Q3=1.936
iqr =Q3-Q1
1.75*(iqr)
```

[109]:
```
0.38849999999999996
```

[110]:
```
1.714 -0.38
```

[110]:
```
1.334
```

[111]:
```
1.936+0.39
```

[111]:
```
2.326
```

[ ]:
```

```

Chapter6

March 2, 2022

1 Chapter 6 - Data Sourcing via Web

1.1 Part 1 - Objects in BeautifulSoup

[1]:
```
import sys
print(sys.version)
```

```
3.8.8 (default, Apr 13 2021, 15:08:03) [MSC v.1916 64 bit (AMD64)]
```

[2]:
```
from bs4 import BeautifulSoup
```

1.1.1 BeautifulSoup objects

[3]:
```
our_html_document = '''
<html><head><title>IoT Articles</title></head>
<body>
<p class='title'><b>2018 Trends: Best New IoT Device Ideas for Data Scientists␣
 ↪and Engineers</b></p>

<p class='description'>It's almost 2018 and IoT is on the cusp of an explosive␣
 ↪expansion. In this article, I offer you a listing of new IoT device ideas␣
 ↪that you can use...
<br>
<br>
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this␣
 ↪article, I offer you a listing of new IoT device ideas that you can use to␣
 ↪get practice in designing your first IoT applications.
<h1>Looking Back at My Coolest IoT Find in 2017</h1>
Before going into detail about best new IoT device ideas, here's the backstory.␣
 ↪<span style="text-decoration: underline;"><strong><a href="http://bit.ly/
 ↪LPlNDJj">Last month Ericsson Digital invited me</a></strong></span> to tour␣
 ↪the Ericsson Studio in Kista, Sweden. Up until that visit, <a href="http://
 ↪www.data-mania.com/blog/m2m-vs-iot/">IoT</a> had been largely theoretical to␣
 ↪me. Of course, I know the usual mumbo-jumbo about wearables and␣
 ↪IoT-connected fitness trackers. That stuff is all well and good, but it's␣
 ↪somewhat old hat - plus I am not sure we are really benefiting so much from␣
 ↪those, so I'm not that impressed.
```

```
It wasn't until I got to the Ericsson Studio that I became extremely impressed␣
 ↪by how far IoT has really come. Relying on the promise of the 5g network␣
 ↪expansion, IoT-powered smart devices are on the cusp of an explosive growth␣
 ↪in adoption. It was Ericsson's Smart Car that sent me reeling:<a href="bit.
 ↪ly/LPlNDJj"><img class="aligncenter size-full wp-image-3802" src="http://www.
 ↪data-mania.com/blog/wp-content/uploads/2017/12/new-IoT-device-ideas.jpg"␣
 ↪alt="Get your new iot device ideas here" width="1024" height="683" /></a>

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform␣
 ↪that manages services for the Smart Cars to which it's connected. The Volvo␣
 ↪pictured above acts as a drop-off location for groceries that have been␣
 ↪ordered by its owner.

To understand how it works, imagine you're pulling your normal 9-to-5 and you␣
 ↪know you need to grab some groceries on your way home. Well, since you're␣
 ↪smart you've used Ericsson IoT platform to connect your car to the local␣
 ↪grocery delivery service (<a href="http://mat.se/">Mat.se</a>), so all you␣
 ↪need to do is open the Mat.se app and make your usual order. Mat.se␣
 ↪automatically handles the payment, grocery selection, delivery, and delivery␣
 ↪scheduling. Since your car is IoT-enabled, Mat.se issues its trusted␣
 ↪delivery agent a 1-time token to use for opening your car in order to place␣
 ↪your groceries in your car for you at 4:40 pm (just before you get off from␣
 ↪work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio,␣
 ↪make sure to go <span style="text-decoration: underline;"><strong><a␣
 ↪href="http://bit.ly/LPlNDJj">watch the videos on this page</a></strong></
 ↪span>.
<h1>Future Trends for IoT in 2018</h1>
New IoT device ideas won't do you much good unless you at least know the basic␣
 ↪technology trends that are set to impact IoT over the next year(s). These␣
 ↪include:
<ol>
        <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are␣
 ↪embedded within IoT devices spin off machine-generated data like it's going␣
 ↪out of style. For IoT to function, the platform must be solidly engineered␣
 ↪to handle big data. Be assured, that requires some serious data engineering.
 ↪</li>
        <li><strong>Machine Learning</strong> Data Science: While a lot of IoT␣
 ↪devices are still operated according to rules-based decision criteria, the␣
 ↪age of artificial intelligence is upon us. IoT will increasingly depend on␣
 ↪machine learning algorithms to control device operations so that devices are␣
 ↪able to autonomously respond to a complex set of overlapping stimuli.</li>
        <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT␣
 ↪networks must be secure. Blockchain technology is primed to meet the␣
 ↪security demands that come along with building and expanding the IoT.</li>
```

```
</ol>
<h1>Best New IoT Device Ideas</h1>
This listing of new IoT device ideas has been sub-divided according to the main␣
 ↪technology upon which the IoT devices are built. Below I'm providing a list␣
 ↪of new IoT device ideas, but for detailed instructions on how to build these␣
 ↪IoT applications, I recommend the <a href="https://click.linksynergy.com/
 ↪deeplink?id=*JDLXjeE*wk&amp;mid=39197&amp;murl=https%3A%2F%2Fwww.udemy.
 ↪com%2Ftopic%2Finternet-of-things%2F%3Fsort%3Dhighest-rated">IoT courses on␣
 ↪Udemy</a> (ß Please note: if you purchase a Udemy course through this link,␣
 ↪I may receive a small commission), or courses that are available at <a␣
 ↪href="http://www.skyfilabs.com/iot-online-courses">SkyFi</a> and <a␣
 ↪href="https://www.coursera.org/specializations/iot">Coursera</a>.
<h2>Raspberry Pi IoT Ideas</h2>
Using Raspberry Pi as open-source hardware, you can build IoT applications that␣
 ↪offer any one of the following benefits:
<ol>
        <li>Enable built-in sensing to build a weather station that measures␣
 ↪ambient temperature and humidity</li>
        <li>Build a system that detects discrepancies in electrical readings␣
 ↪to identify electricity theft</li>
        <li>Use IoT to build a Servo that is controlled by motion detection␣
 ↪readings</li>
        <li>Build a smart control switch that operates devices based on␣
 ↪external stimuli. Use this for home automation.</li>
        <li>Build a music playing application that enables music for each room␣
 ↪in your house</li>
        <li>Implement biometrics on IoT-connected devices</li>
</ol>
<h2>Arduino IoT Ideas</h2>
There are a number of new IoT device ideas that deploy Arduino as a␣
 ↪microcontroller. These include:
<ol>
        <li>Integrate Arduino with Android to build a remote-control RGB LED␣
 ↪device.</li>
        <li>Connect PIR sensors across the IoT to implement a smart building.</
 ↪li>
        <li>Build a temperature and sunlight sensor system to remotely monitor␣
 ↪and control the conditions of your garden.</li>
        <li>Deploy Arduino and IoT to automate your neighborhood streetlights.
 ↪</li>
        <li>Build a smart irrigation system based on IoT-connected temperature␣
 ↪and moisture sensors built-in to your agricultural plants.</li>
</ol>
```

```
[caption id="attachment_3807" align="aligncenter" width="300"]<a href="bit.ly/
 ↪LPlNDJj"><img class="wp-image-3807 size-medium" src="http://www.data-mania.
 ↪com/blog/wp-content/uploads/2017/12/IMG_3058-300x295.jpg" alt="" width="300"␣
 ↪height="295" /></a> An IoT Chatbot Tree at the Ericsson Studio[/caption]
<h2>Wireless (GSM) IoT Ideas</h2>
Several new IoT device ideas are developed around the GSM wireless network.␣
 ↪Those are:
<ol>
        <li>Monitor soil moisture to automate agricultural irrigation cycles.</
 ↪li>
        <li>Automate and control the conditions of a greenhouse.</li>
        <li>Enable bio-metrics to build a smart security system for your home␣
 ↪or office building</li>
        <li>Build an autonomously operating fitness application that␣
 ↪automatically makes recommendations based on motion detection and heart rate␣
 ↪sensors that are embedded on wearable fitness trackers.</li>
        <li>Build a healthcare monitoring system that tracks, informs, and␣
 ↪automatically alerts healthcare providers based on sensor readings that␣
 ↪describe a patients vital statistics (like temperature, pulse, blood␣
 ↪pressure, etc).</li>
</ol>
<h2>IoT Automation Ideas</h2>
Almost all new IoT device ideas offer automation benefits, but to outline a few␣
 ↪more ideas:
<ol>
        <li>Build an IoT device that automatically locates and reports the␣
 ↪closest nearby parking spot.</li>
        <li>Build a motion detection system that automatically issues emails␣
 ↪or sms messages to alert home owners of a likely home invasion.</li>
        <li>Use temperature sensors connected across the IoT to automatically␣
 ↪alert you if your home windows or doors have been left open.</li>
        <li>Use bio-metric sensors to build a smart system that automate␣
 ↪security for your home or office building</li>
</ol>
To learn more about IoT and what's happening on the leading edge, be sure to␣
 ↪pop over to Ericsson's Studio Tour recap and <span style="text-decoration:␣
 ↪underline;"><strong><a href="http://bit.ly/LPlNDJj">watch these videos</a></
 ↪strong></span>.

<em>(I captured some of this content on behalf of DevMode Strategies during an␣
 ↪invite-only tour of the Ericsson Studio in Kista. Rest assure, the text␣
 ↪and opinions are my own</em>)
<p class='description'>...</p>
'''
```

[4]:
```
our_soup_object = BeautifulSoup(our_html_document, 'html.parser')
print(our_soup_object)
```

```
<html><head><title>IoT Articles</title></head>
<body>
<p class="title"><b>2018 Trends: Best New IoT Device Ideas for Data Scientists
and Engineers</b></p>
<p class="description">It's almost 2018 and IoT is on the cusp of an explosive
expansion. In this article, I offer you a listing of new IoT device ideas that
you can use…
<br/>
<br/>
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use to get
practice in designing your first IoT applications.
<h1>Looking Back at My Coolest IoT Find in 2017</h1>
Before going into detail about best new IoT device ideas, here's the backstory.
<span style="text-decoration: underline;"><strong><a
href="http://bit.ly/LPlNDJj">Last month Ericsson Digital invited
me</a></strong></span> to tour the Ericsson Studio in Kista, Sweden. Up until
that visit, <a href="http://www.data-mania.com/blog/m2m-vs-iot/">IoT</a> had
been largely theoretical to me. Of course, I know the usual mumbo-jumbo about
wearables and IoT-connected fitness trackers. That stuff is all well and good,
but it's somewhat old hat - plus I am not sure we are really benefiting so much
from those, so I'm not that impressed.

It wasn't until I got to the Ericsson Studio that I became extremely impressed
by how far IoT has really come. Relying on the promise of the 5g network
expansion, IoT-powered smart devices are on the cusp of an explosive growth in
adoption. It was Ericsson's Smart Car that sent me reeling:<a
href="bit.ly/LPlNDJj"><img alt="Get your new iot device ideas here"
class="aligncenter size-full wp-image-3802" height="683" src="http://www.data-
mania.com/blog/wp-content/uploads/2017/12/new-IoT-device-ideas.jpg"
width="1024"/></a>

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform
that manages services for the Smart Cars to which it's connected. The Volvo
pictured above acts as a drop-off location for groceries that have been ordered
by its owner.

To understand how it works, imagine you're pulling your normal 9-to-5 and you
know you need to grab some groceries on your way home. Well, since you're smart
you've used Ericsson IoT platform to connect your car to the local grocery
delivery service (<a href="http://mat.se/">Mat.se</a>), so all you need to do is
open the Mat.se app and make your usual order. Mat.se automatically handles the
payment, grocery selection, delivery, and delivery scheduling. Since your car is
IoT-enabled, Mat.se issues its trusted delivery agent a 1-time token to use for
```

opening your car in order to place your groceries in your car for you at 4:40 pm (just before you get off from work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio, make sure to go watch the videos on this page.
<h1>Future Trends for IoT in 2018</h1>
New IoT device ideas won't do you much good unless you at least know the basic technology trends that are set to impact IoT over the next year(s). These include:

Big Data & Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it's going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.
Machine Learning Data Science: While a lot of IoT devices are still operated according to rules-based decision criteria, the age of artificial intelligence is upon us. IoT will increasingly depend on machine learning algorithms to control device operations so that devices are able to autonomously respond to a complex set of overlapping stimuli.
Blockchain-Enabled Security: Above all else, IoT networks must be secure. Blockchain technology is primed to meet the security demands that come along with building and expanding the IoT.

<h1>Best New IoT Device Ideas</h1>
This listing of new IoT device ideas has been sub-divided according to the main technology upon which the IoT devices are built. Below I'm providing a list of new IoT device ideas, but for detailed instructions on how to build these IoT applications, I recommend the IoT courses on Udemy (ß Please note: if you purchase a Udemy course through this link, I may receive a small commission), or courses that are available at SkyFi and Coursera.
<h2>Raspberry Pi IoT Ideas</h2>
Using Raspberry Pi as open-source hardware, you can build IoT applications that offer any one of the following benefits:

Enable built-in sensing to build a weather station that measures ambient temperature and humidity
Build a system that detects discrepancies in electrical readings to identify electricity theft
Use IoT to build a Servo that is controlled by motion detection readings
Build a smart control switch that operates devices based on external stimuli. Use this for home automation.
Build a music playing application that enables music for each room in your

```
house</li>
<li>Implement biometrics on IoT-connected devices</li>
</ol>
<h2>Arduino IoT Ideas</h2>
There are a number of new IoT device ideas that deploy Arduino as a
microcontroller. These include:
<ol>
<li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>
<li>Connect PIR sensors across the IoT to implement a smart building.</li>
<li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>
<li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>
<li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>
</ol>
[caption id="attachment_3807" align="aligncenter" width="300"]<a
href="bit.ly/LPlNDJj"><img alt="" class="wp-image-3807 size-medium" height="295"
src="http://www.data-mania.com/blog/wp-
content/uploads/2017/12/IMG_3058-300x295.jpg" width="300"/></a> An IoT Chatbot
Tree at the Ericsson Studio[/caption]
<h2>Wireless (GSM) IoT Ideas</h2>
Several new IoT device ideas are developed around the GSM wireless network.
Those are:
<ol>
<li>Monitor soil moisture to automate agricultural irrigation cycles.</li>
<li>Automate and control the conditions of a greenhouse.</li>
<li>Enable bio-metrics to build a smart security system for your home or office
building</li>
<li>Build an autonomously operating fitness application that automatically makes
recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>
<li>Build a healthcare monitoring system that tracks, informs, and automatically
alerts healthcare providers based on sensor readings that describe a patients
vital statistics (like temperature, pulse, blood pressure, etc).</li>
</ol>
<h2>IoT Automation Ideas</h2>
Almost all new IoT device ideas offer automation benefits, but to outline a few
more ideas:
<ol>
<li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>
<li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>
<li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>
<li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>
```

```
</ol>
To learn more about IoT and what's happening on the leading edge, be sure to pop
over to Ericsson's Studio Tour recap and <span style="text-decoration:
underline;"><strong><a href="http://bit.ly/LPlNDJj">watch these
videos</a></strong></span>.

<em>(I captured some of this content on behalf of DevMode Strategies during an
invite-only tour of the Ericsson Studio in Kista. Rest assure, the text
and opinions are my own</em>)
<p class="description">…</p>
</p></body></html>
```

[11]:
```
print(our_soup_object.prettify()[0:300])
```

```
<html>
 <head>
  <title>
   IoT Articles
  </title>
 </head>
 <body>
  <p class="title">
   <b>
    2018 Trends: Best New IoT Device Ideas for Data Scientists and Engineers
   </b>
  </p>
  <p class="description">
   It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article,
```

1.1.2 Tag objects

Tag names

[5]:
```
soup_object = BeautifulSoup('<h1 attribute_1 = "Heading Level 1"">Future Trends␣
↪for IoT in 2018</h1>', "lxml")

tag = soup_object.h1
type(tag)
```

[5]:
```
bs4.element.Tag
```

[6]:
```
print(tag)
```

```
<h1 attribute_1="Heading Level 1">Future Trends for IoT in 2018</h1>
```

[7]:
```
tag.name
```

[7]:
```
'h1'
```

[8]:
```
tag.name = 'heading 1'
tag
```

[8]:
```
<heading 1 attribute_1="Heading Level 1">Future Trends for IoT in 2018</heading
1>
```

[9]:
```
tag.name
```

[9]:
```
'heading 1'
```

Tag attributes

[10]:
```
soup_object = BeautifulSoup('<h1 attribute_1 = "Heading Level 1"">Future Trends␣
↪for IoT in 2018</h1>', "lxml")
tag = soup_object.h1
tag
```

[10]:
```
<h1 attribute_1="Heading Level 1">Future Trends for IoT in 2018</h1>
```

[11]:
```
tag['attribute_1']
```

[11]:
```
'Heading Level 1'
```

[12]:
```
tag.attrs
```

[12]:
```
{'attribute_1': 'Heading Level 1'}
```

[13]:
```
tag['attribute_2'] = 'Heading Level 1*'
tag.attrs
```

[13]:
```
{'attribute_1': 'Heading Level 1', 'attribute_2': 'Heading Level 1*'}
```

[14]:
```
tag
```

[14]:
```
<h1 attribute_1="Heading Level 1" attribute_2="Heading Level 1*">Future Trends
for IoT in 2018</h1>
```

[15]:
```
del tag['attribute_2']
tag
```

[15]:
```
<h1 attribute_1="Heading Level 1">Future Trends for IoT in 2018</h1>
```

[16]:
```
del tag['attribute_1']
tag.attrs
```

[16]:
```
{}
```

Navigating a parse tree using tags

[17]:
```
# First we will recreate our original parse tree.
our_html_document = '''
<html><head><title>IoT Articles</title></head>
<body>
<p class='title'><b>2018 Trends: Best New IoT Device Ideas for Data Scientists␣
 ↪and Engineers</b></p>

<p class='description'>It's almost 2018 and IoT is on the cusp of an explosive␣
 ↪expansion. In this article, I offer you a listing of new IoT device ideas␣
 ↪that you can use...
<br>
<br>
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this␣
 ↪article, I offer you a listing of new IoT device ideas that you can use to␣
 ↪get practice in designing your first IoT applications.
<h1>Looking Back at My Coolest IoT Find in 2017</h1>
Before going into detail about best new IoT device ideas, here's the backstory.␣
 ↪<span style="text-decoration: underline;"><strong><a href="http://bit.ly/
 ↪LPlNDJj">Last month Ericsson Digital invited me</a></strong></span> to tour␣
 ↪the Ericsson Studio in Kista, Sweden. Up until that visit, <a href="http://
 ↪www.data-mania.com/blog/m2m-vs-iot/">IoT</a> had been largely theoretical to␣
 ↪me. Of course, I know the usual mumbo-jumbo about wearables and␣
 ↪IoT-connected fitness trackers. That stuff is all well and good, but it's␣
 ↪somewhat old hat - plus I am not sure we are really benefiting so much from␣
 ↪those, so I'm not that impressed.

It wasn't until I got to the Ericsson Studio that I became extremely impressed␣
 ↪by how far IoT has really come. Relying on the promise of the 5g network␣
 ↪expansion, IoT-powered smart devices are on the cusp of an explosive growth␣
 ↪in adoption. It was Ericsson's Smart Car that sent me reeling:<a href="bit.
 ↪ly/LPlNDJj"><img class="aligncenter size-full wp-image-3802" src="http://www.
 ↪data-mania.com/blog/wp-content/uploads/2017/12/new-IoT-device-ideas.jpg"␣
 ↪alt="Get your new iot device ideas here" width="1024" height="683" /></a>

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform␣
 ↪that manages services for the Smart Cars to which it's connected. The Volvo␣
 ↪pictured above acts as a drop-off location for groceries that have been␣
 ↪ordered by its owner.
```

```
To understand how it works, imagine you're pulling your normal 9-to-5 and you␣
 ↪know you need to grab some groceries on your way home. Well, since you're␣
 ↪smart you've used Ericsson IoT platform to connect your car to the local␣
 ↪grocery delivery service (<a href="http://mat.se/">Mat.se</a>), so all you␣
 ↪need to do is open the Mat.se app and make your usual order. Mat.se␣
 ↪automatically handles the payment, grocery selection, delivery, and delivery␣
 ↪scheduling. Since your car is IoT-enabled, Mat.se issues its trusted␣
 ↪delivery agent a 1-time token to use for opening your car in order to place␣
 ↪your groceries in your car for you at 4:40 pm (just before you get off from␣
 ↪work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio,␣
 ↪make sure to go <span style="text-decoration: underline;"><strong><a␣
 ↪href="http://bit.ly/LPlNDJj">watch the videos on this page</a></strong></
 ↪span>.
<h1>Future Trends for IoT in 2018</h1>
New IoT device ideas won't do you much good unless you at least know the basic␣
 ↪technology trends that are set to impact IoT over the next year(s). These␣
 ↪include:
<ol>
        <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are␣
 ↪embedded within IoT devices spin off machine-generated data like it's going␣
 ↪out of style. For IoT to function, the platform must be solidly engineered␣
 ↪to handle big data. Be assured, that requires some serious data engineering.
 ↪</li>
        <li><strong>Machine Learning</strong> Data Science: While a lot of IoT␣
 ↪devices are still operated according to rules-based decision criteria, the␣
 ↪age of artificial intelligence is upon us. IoT will increasingly depend on␣
 ↪machine learning algorithms to control device operations so that devices are␣
 ↪able to autonomously respond to a complex set of overlapping stimuli.</li>
        <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT␣
 ↪networks must be secure. Blockchain technology is primed to meet the␣
 ↪security demands that come along with building and expanding the IoT.</li>
</ol>
<h1>Best New IoT Device Ideas</h1>
This listing of new IoT device ideas has been sub-divided according to the main␣
 ↪technology upon which the IoT devices are built. Below I'm providing a list␣
 ↪of new IoT device ideas, but for detailed instructions on how to build these␣
 ↪IoT applications, I recommend the <a href="https://click.linksynergy.com/
 ↪deeplink?id=*JDLXjeE*wk&amp;mid=39197&amp;murl=https%3A%2F%2Fwww.udemy.
 ↪com%2Ftopic%2Finternet-of-things%2F%3Fsort%3Dhighest-rated">IoT courses on␣
 ↪Udemy</a> (ß Please note: if you purchase a Udemy course through this link,␣
 ↪I may receive a small commission), or courses that are available at <a␣
 ↪href="http://www.skyfilabs.com/iot-online-courses">SkyFi</a> and <a␣
 ↪href="https://www.coursera.org/specializations/iot">Coursera</a>.
<h2>Raspberry Pi IoT Ideas</h2>
```

```
Using Raspberry Pi as open-source hardware, you can build IoT applications that␣
 ↪offer any one of the following benefits:
<ol>
        <li>Enable built-in sensing to build a weather station that measures␣
 ↪ambient temperature and humidity</li>
        <li>Build a system that detects discrepancies in electrical readings␣
 ↪to identify electricity theft</li>
        <li>Use IoT to build a Servo that is controlled by motion detection␣
 ↪readings</li>
        <li>Build a smart control switch that operates devices based on␣
 ↪external stimuli. Use this for home automation.</li>
        <li>Build a music playing application that enables music for each room␣
 ↪in your house</li>
        <li>Implement biometrics on IoT-connected devices</li>
</ol>
<h2>Arduino IoT Ideas</h2>
There are a number of new IoT device ideas that deploy Arduino as a␣
 ↪microcontroller. These include:
<ol>
        <li>Integrate Arduino with Android to build a remote-control RGB LED␣
 ↪device.</li>
        <li>Connect PIR sensors across the IoT to implement a smart building.</
 ↪li>
        <li>Build a temperature and sunlight sensor system to remotely monitor␣
 ↪and control the conditions of your garden.</li>
        <li>Deploy Arduino and IoT to automate your neighborhood streetlights.
 ↪</li>
        <li>Build a smart irrigation system based on IoT-connected temperature␣
 ↪and moisture sensors built-in to your agricultural plants.</li>
</ol>
[caption id="attachment_3807" align="aligncenter" width="300"]<a href="bit.ly/
 ↪LPlNDJj"><img class="wp-image-3807 size-medium" src="http://www.data-mania.
 ↪com/blog/wp-content/uploads/2017/12/IMG_3058-300x295.jpg" alt="" width="300"␣
 ↪height="295" /></a> An IoT Chatbot Tree at the Ericsson Studio[/caption]
<h2>Wireless (GSM) IoT Ideas</h2>
Several new IoT device ideas are developed around the GSM wireless network.␣
 ↪Those are:
<ol>
        <li>Monitor soil moisture to automate agricultural irrigation cycles.</
 ↪li>
        <li>Automate and control the conditions of a greenhouse.</li>
        <li>Enable bio-metrics to build a smart security system for your home␣
 ↪or office building</li>
        <li>Build an autonomously operating fitness application that␣
 ↪automatically makes recommendations based on motion detection and heart rate␣
 ↪sensors that are embedded on wearable fitness trackers.</li>
```

```
        <li>Build a healthcare monitoring system that tracks, informs, and␣
 ↪automatically alerts healthcare providers based on sensor readings that␣
 ↪describe a patients vital statistics (like temperature, pulse, blood␣
 ↪pressure, etc).</li>
</ol>
<h2>IoT Automation Ideas</h2>
Almost all new IoT device ideas offer automation benefits, but to outline a few␣
 ↪more ideas:
<ol>
        <li>Build an IoT device that automatically locates and reports the␣
 ↪closest nearby parking spot.</li>
        <li>Build a motion detection system that automatically issues emails␣
 ↪or sms messages to alert home owners of a likely home invasion.</li>
        <li>Use temperature sensors connected across the IoT to automatically␣
 ↪alert you if your home windows or doors have been left open.</li>
        <li>Use bio-metric sensors to build a smart system that automate␣
 ↪security for your home or office building</li>
</ol>
To learn more about IoT and what's happening on the leading edge, be sure to␣
 ↪pop over to Ericsson's Studio Tour recap and <span style="text-decoration:␣
 ↪underline;"><strong><a href="http://bit.ly/LPlNDJj">watch these videos</a></
 ↪strong></span>.

<em>(I captured some of this content on behalf of DevMode Strategies during an␣
 ↪invite-only tour of the Ericsson Studio in Kista. Rest assure, the text␣
 ↪and opinions are my own</em>)
<p class='description'>...</p>
'''
our_soup_object = BeautifulSoup(our_html_document, 'html.parser')
```

[22]:
```
our_soup_object.head
```

[22]:
```
<head><title>IoT Articles</title></head>
```

[21]:
```
our_soup_object.title
```

[21]:
```
<title>IoT Articles</title>
```

[20]:
```
our_soup_object.body.b
```

[20]:
```
<b>2018 Trends: Best New IoT Device Ideas for Data Scientists and Engineers</b>
```

[23]:
```
our_soup_object.body
```

[23]:
```
<body>
<p class="title"><b>2018 Trends: Best New IoT Device Ideas for Data Scientists
and Engineers</b></p>
```

<p class="description">It’s almost 2018 and IoT is on the cusp of an explosive expansion. In this article, I offer you a listing of new IoT device ideas that you can use…

It’s almost 2018 and IoT is on the cusp of an explosive expansion. In this article, I offer you a listing of new IoT device ideas that you can use to get practice in designing your first IoT applications.
<h1>Looking Back at My Coolest IoT Find in 2017</h1>
Before going into detail about best new IoT device ideas, here’s the backstory. Last month Ericsson Digital invited me to tour the Ericsson Studio in Kista, Sweden. Up until that visit, IoT had been largely theoretical to me. Of course, I know the usual mumbo-jumbo about wearables and IoT-connected fitness trackers. That stuff is all well and good, but it’s somewhat old hat - plus I am not sure we are really benefiting so much from those, so I’m not that impressed.

It wasn’t until I got to the Ericsson Studio that I became extremely impressed by how far IoT has really come. Relying on the promise of the 5g network expansion, IoT-powered smart devices are on the cusp of an explosive growth in adoption. It was Ericsson’s Smart Car that sent me reeling:

This car is connected to Ericsson’s Connected Vehicle Cloud, an IoT platform that manages services for the Smart Cars to which it’s connected. The Volvo pictured above acts as a drop-off location for groceries that have been ordered by its owner.

To understand how it works, imagine you’re pulling your normal 9-to-5 and you know you need to grab some groceries on your way home. Well, since you’re smart you’ve used Ericsson IoT platform to connect your car to the local grocery delivery service (Mat.se), so all you need to do is open the Mat.se app and make your usual order. Mat.se automatically handles the payment, grocery selection, delivery, and delivery scheduling. Since your car is IoT-enabled, Mat.se issues its trusted delivery agent a 1-time token to use for opening your car in order to place your groceries in your car for you at 4:40 pm (just before you get off from work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio, make sure to go watch the videos on this page.
<h1>Future Trends for IoT in 2018</h1>

New IoT device ideas won’t do you much good unless you at least know the basic technology trends that are set to impact IoT over the next year(s). These include:

Big Data & Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it’s going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.
Machine Learning Data Science: While a lot of IoT devices are still operated according to rules-based decision criteria, the age of artificial intelligence is upon us. IoT will increasingly depend on machine learning algorithms to control device operations so that devices are able to autonomously respond to a complex set of overlapping stimuli.
Blockchain-Enabled Security: Above all else, IoT networks must be secure. Blockchain technology is primed to meet the security demands that come along with building and expanding the IoT.

<h1>Best New IoT Device Ideas</h1>
This listing of new IoT device ideas has been sub-divided according to the main technology upon which the IoT devices are built. Below I’m providing a list of new IoT device ideas, but for detailed instructions on how to build these IoT applications, I recommend the IoT courses on Udemy (ß Please note: if you purchase a Udemy course through this link, I may receive a small commission), or courses that are available at SkyFi and Coursera.
<h2>Raspberry Pi IoT Ideas</h2>
Using Raspberry Pi as open-source hardware, you can build IoT applications that offer any one of the following benefits:

Enable built-in sensing to build a weather station that measures ambient temperature and humidity
Build a system that detects discrepancies in electrical readings to identify electricity theft
Use IoT to build a Servo that is controlled by motion detection readings
Build a smart control switch that operates devices based on external stimuli. Use this for home automation.
Build a music playing application that enables music for each room in your house
Implement biometrics on IoT-connected devices

<h2>Arduino IoT Ideas</h2>
There are a number of new IoT device ideas that deploy Arduino as a microcontroller. These include:

Integrate Arduino with Android to build a remote-control RGB LED device.
Connect PIR sensors across the IoT to implement a smart building.
Build a temperature and sunlight sensor system to remotely monitor and control the conditions of your garden.
Deploy Arduino and IoT to automate your neighborhood streetlights.
Build a smart irrigation system based on IoT-connected temperature and moisture sensors built-in to your agricultural plants.

[caption id="attachment_3807" align="aligncenter" width="300"] An IoT Chatbot Tree at the Ericsson Studio[/caption]
<h2>Wireless (GSM) IoT Ideas</h2>
Several new IoT device ideas are developed around the GSM wireless network. Those are:

Monitor soil moisture to automate agricultural irrigation cycles.
Automate and control the conditions of a greenhouse.
Enable bio-metrics to build a smart security system for your home or office building
Build an autonomously operating fitness application that automatically makes recommendations based on motion detection and heart rate sensors that are embedded on wearable fitness trackers.
Build a healthcare monitoring system that tracks, informs, and automatically alerts healthcare providers based on sensor readings that describe a patients vital statistics (like temperature, pulse, blood pressure, etc).

<h2>IoT Automation Ideas</h2>
Almost all new IoT device ideas offer automation benefits, but to outline a few more ideas:

Build an IoT device that automatically locates and reports the closest nearby parking spot.
Build a motion detection system that automatically issues emails or sms messages to alert home owners of a likely home invasion.
Use temperature sensors connected across the IoT to automatically alert you if your home windows or doors have been left open.
Use bio-metric sensors to build a smart system that automate security for your home or office building

To learn more about IoT and what’s happening on the leading edge, be sure to pop over to Ericsson’s Studio Tour recap and watch these videos.

```
<em>(I captured some of this content on behalf of DevMode Strategies during an
invite-only tour of the Ericsson Studio in Kista. Rest assure, the text
and opinions are my own</em>)
<p class="description">…</p>
</p></body>
```

[24]:
```
our_soup_object.li
```

[24]:
```
<li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it's going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>
```

[26]:
```
our_soup_object.a
```

[26]:
```
<a href="http://bit.ly/LPlNDJj">Last month Ericsson Digital invited me</a>
```

1.2 Part 2 - NavigatableString Objects

[27]:
```
soup_object = BeautifulSoup('<h1 attribute_1 = "Heading Level 1"">Future Trends␣
↪in IoT in 2018</h1>', "lxml")

tag = soup_object.h1

type(tag)
```

[27]:
```
bs4.element.Tag
```

[28]:
```
tag.name
```

[28]:
```
'h1'
```

[29]:
```
tag.string
```

[29]:
```
'Future Trends in IoT in 2018'
```

[30]:
```
type(tag.string)
```

[30]:
```
bs4.element.NavigableString
```

[31]:
```
our_navigatable_string = tag.string
our_navigatable_string
```

[31]:
```
'Future Trends in IoT in 2018'
```

[32]:
```
our_navigatable_string.replace_with('NaN')
tag.string
```

[32]:
```
'NaN'
```

Utilizing NavigatableString objects

[34]:
```
our_html_document = '''
<html><head><title>IoT Articles</title></head>
<body>
<p class='title'><b>2018 Trends: Best New IoT Device Ideas for Data Scientists␣
 ↪and Engineers</b></p>

<p class='description'>It's almost 2018 and IoT is on the cusp of an explosive␣
 ↪expansion. In this article, I offer you a listing of new IoT device ideas␣
 ↪that you can use...
<br>
<br>
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this␣
 ↪article, I offer you a listing of new IoT device ideas that you can use to␣
 ↪get practice in designing your first IoT applications.
<h1>Looking Back at My Coolest IoT Find in 2017</h1>
Before going into detail about best new IoT device ideas, here's the backstory.␣
 ↪<span style="text-decoration: underline;"><strong><a href="http://bit.ly/
 ↪LPlNDJj">Last month Ericsson Digital invited me</a></strong></span> to tour␣
 ↪the Ericsson Studio in Kista, Sweden. Up until that visit, <a href="http://
 ↪www.data-mania.com/blog/m2m-vs-iot/">IoT</a> had been largely theoretical to␣
 ↪me. Of course, I know the usual mumbo-jumbo about wearables and␣
 ↪IoT-connected fitness trackers. That stuff is all well and good, but it's␣
 ↪somewhat old hat - plus I am not sure we are really benefiting so much from␣
 ↪those, so I'm not that impressed.

It wasn't until I got to the Ericsson Studio that I became extremely impressed␣
 ↪by how far IoT has really come. Relying on the promise of the 5g network␣
 ↪expansion, IoT-powered smart devices are on the cusp of an explosive growth␣
 ↪in adoption. It was Ericsson's Smart Car that sent me reeling:<a href="bit.
 ↪ly/LPlNDJj"><img class="aligncenter size-full wp-image-3802" src="http://www.
 ↪data-mania.com/blog/wp-content/uploads/2017/12/new-IoT-device-ideas.jpg"␣
 ↪alt="Get your new iot device ideas here" width="1024" height="683" /></a>

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform␣
 ↪that manages services for the Smart Cars to which it's connected. The Volvo␣
 ↪pictured above acts as a drop-off location for groceries that have been␣
 ↪ordered by its owner.
```

```
To understand how it works, imagine you're pulling your normal 9-to-5 and you␣
 ↪know you need to grab some groceries on your way home. Well, since you're␣
 ↪smart you've used Ericsson IoT platform to connect your car to the local␣
 ↪grocery delivery service (<a href="http://mat.se/">Mat.se</a>), so all you␣
 ↪need to do is open the Mat.se app and make your usual order. Mat.se␣
 ↪automatically handles the payment, grocery selection, delivery, and delivery␣
 ↪scheduling. Since your car is IoT-enabled, Mat.se issues its trusted␣
 ↪delivery agent a 1-time token to use for opening your car in order to place␣
 ↪your groceries in your car for you at 4:40 pm (just before you get off from␣
 ↪work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio,␣
 ↪make sure to go <span style="text-decoration: underline;"><strong><a␣
 ↪href="http://bit.ly/LPlNDJj">watch the videos on this page</a></strong></
 ↪span>.
<h1>Future Trends for IoT in 2018</h1>
New IoT device ideas won't do you much good unless you at least know the basic␣
 ↪technology trends that are set to impact IoT over the next year(s). These␣
 ↪include:
<ol>
        <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are␣
 ↪embedded within IoT devices spin off machine-generated data like it's going␣
 ↪out of style. For IoT to function, the platform must be solidly engineered␣
 ↪to handle big data. Be assured, that requires some serious data engineering.
 ↪</li>
        <li><strong>Machine Learning</strong> Data Science: While a lot of IoT␣
 ↪devices are still operated according to rules-based decision criteria, the␣
 ↪age of artificial intelligence is upon us. IoT will increasingly depend on␣
 ↪machine learning algorithms to control device operations so that devices are␣
 ↪able to autonomously respond to a complex set of overlapping stimuli.</li>
        <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT␣
 ↪networks must be secure. Blockchain technology is primed to meet the␣
 ↪security demands that come along with building and expanding the IoT.</li>
</ol>
<h1>Best New IoT Device Ideas</h1>
This listing of new IoT device ideas has been sub-divided according to the main␣
 ↪technology upon which the IoT devices are built. Below I'm providing a list␣
 ↪of new IoT device ideas, but for detailed instructions on how to build these␣
 ↪IoT applications, I recommend the <a href="https://click.linksynergy.com/
 ↪deeplink?id=*JDLXjeE*wk&amp;mid=39197&amp;murl=https%3A%2F%2Fwww.udemy.
 ↪com%2Ftopic%2Finternet-of-things%2F%3Fsort%3Dhighest-rated">IoT courses on␣
 ↪Udemy</a> (ß Please note: if you purchase a Udemy course through this link,␣
 ↪I may receive a small commission), or courses that are available at <a␣
 ↪href="http://www.skyfilabs.com/iot-online-courses">SkyFi</a> and <a␣
 ↪href="https://www.coursera.org/specializations/iot">Coursera</a>.
<h2>Raspberry Pi IoT Ideas</h2>
```

```
Using Raspberry Pi as open-source hardware, you can build IoT applications that␣
 ↪offer any one of the following benefits:
<ol>
        <li>Enable built-in sensing to build a weather station that measures␣
 ↪ambient temperature and humidity</li>
        <li>Build a system that detects discrepancies in electrical readings␣
 ↪to identify electricity theft</li>
        <li>Use IoT to build a Servo that is controlled by motion detection␣
 ↪readings</li>
        <li>Build a smart control switch that operates devices based on␣
 ↪external stimuli. Use this for home automation.</li>
        <li>Build a music playing application that enables music for each room␣
 ↪in your house</li>
        <li>Implement biometrics on IoT-connected devices</li>
</ol>
<h2>Arduino IoT Ideas</h2>
There are a number of new IoT device ideas that deploy Arduino as a␣
 ↪microcontroller. These include:
<ol>
        <li>Integrate Arduino with Android to build a remote-control RGB LED␣
 ↪device.</li>
        <li>Connect PIR sensors across the IoT to implement a smart building.</
 ↪li>
        <li>Build a temperature and sunlight sensor system to remotely monitor␣
 ↪and control the conditions of your garden.</li>
        <li>Deploy Arduino and IoT to automate your neighborhood streetlights.
 ↪</li>
        <li>Build a smart irrigation system based on IoT-connected temperature␣
 ↪and moisture sensors built-in to your agricultural plants.</li>
</ol>
[caption id="attachment_3807" align="aligncenter" width="300"]<a href="bit.ly/
 ↪LPlNDJj"><img class="wp-image-3807 size-medium" src="http://www.data-mania.
 ↪com/blog/wp-content/uploads/2017/12/IMG_3058-300x295.jpg" alt="" width="300"␣
 ↪height="295" /></a> An IoT Chatbot Tree at the Ericsson Studio[/caption]
<h2>Wireless (GSM) IoT Ideas</h2>
Several new IoT device ideas are developed around the GSM wireless network.␣
 ↪Those are:
<ol>
        <li>Monitor soil moisture to automate agricultural irrigation cycles.</
 ↪li>
        <li>Automate and control the conditions of a greenhouse.</li>
        <li>Enable bio-metrics to build a smart security system for your home␣
 ↪or office building</li>
        <li>Build an autonomously operating fitness application that␣
 ↪automatically makes recommendations based on motion detection and heart rate␣
 ↪sensors that are embedded on wearable fitness trackers.</li>
```

```
        <li>Build a healthcare monitoring system that tracks, informs, and␣
 ↪automatically alerts healthcare providers based on sensor readings that␣
 ↪describe a patients vital statistics (like temperature, pulse, blood␣
 ↪pressure, etc).</li>
</ol>
<h2>IoT Automation Ideas</h2>
Almost all new IoT device ideas offer automation benefits, but to outline a few␣
 ↪more ideas:
<ol>
        <li>Build an IoT device that automatically locates and reports the␣
 ↪closest nearby parking spot.</li>
        <li>Build a motion detection system that automatically issues emails␣
 ↪or sms messages to alert home owners of a likely home invasion.</li>
        <li>Use temperature sensors connected across the IoT to automatically␣
 ↪alert you if your home windows or doors have been left open.</li>
        <li>Use bio-metric sensors to build a smart system that automate␣
 ↪security for your home or office building</li>
</ol>
To learn more about IoT and what's happening on the leading edge, be sure to␣
 ↪pop over to Ericsson's Studio Tour recap and <span style="text-decoration:␣
 ↪underline;"><strong><a href="http://bit.ly/LPlNDJj">watch these videos</a></
 ↪strong></span>.

<em>(I captured some of this content on behalf of DevMode Strategies during an␣
 ↪invite-only tour of the Ericsson Studio in Kista. Rest assure, the text␣
 ↪and opinions are my own</em>)
<p class='description'>...</p>
'''
our_soup_object = BeautifulSoup(our_html_document, 'html.parser')
```

[35]:
```
for string in our_soup_object.stripped_strings:
    print(repr(string))
```

```
'IoT Articles'
'2018 Trends: Best New IoT Device Ideas for Data Scientists and Engineers'
'It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use…'
'It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use to get
practice in designing your first IoT applications.'
'Looking Back at My Coolest IoT Find in 2017'
'Before going into detail about best new IoT device ideas, here's the
backstory.'
'Last month Ericsson Digital invited me'
'to tour the Ericsson Studio in Kista, Sweden. Up until that visit,'
'IoT'
'had been largely theoretical to me. Of course, I know the usual mumbo-jumbo
```

about wearables and IoT-connected fitness trackers. That stuff is all well and good, but it’s somewhat old hat - plus I am not sure we are really benefiting so much from those, so I’m not that impressed.\n\nIt wasn’t until I got to the Ericsson Studio that I became extremely impressed by how far IoT has really come. Relying on the promise of the 5g network expansion, IoT-powered smart devices are on the cusp of an explosive growth in adoption. It was Ericsson’s Smart Car that sent me reeling:'
'This car is connected to Ericsson’s Connected Vehicle Cloud, an IoT platform that manages services for the Smart Cars to which it’s connected. The Volvo pictured above acts as a drop-off location for groceries that have been ordered by its owner.\n\nTo understand how it works, imagine you’re pulling your normal 9-to-5 and you know you need to grab some groceries on your way home. Well, since you’re smart you’ve used Ericsson IoT platform to connect your car to the local grocery delivery service ('
'Mat.se'
'), so all you need to do is open the Mat.se app and make your usual order. Mat.se automatically handles the payment, grocery selection, delivery, and delivery scheduling. Since your car is IoT-enabled, Mat.se issues its trusted delivery agent a 1-time token to use for opening your car in order to place your groceries in your car for you at 4:40 pm (just before you get off from work).\n\nTo watch some of the amazing IoT device demos I witnessed at Ericsson Studio, make sure to go'
'watch the videos on this page'
'.'
'Future Trends for IoT in 2018'
'New IoT device ideas won’t do you much good unless you at least know the basic technology trends that are set to impact IoT over the next year(s). These include:'
'Big Data'
'& Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it’s going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.'
'Machine Learning'
'Data Science: While a lot of IoT devices are still operated according to rules-based decision criteria, the age of artificial intelligence is upon us. IoT will increasingly depend on machine learning algorithms to control device operations so that devices are able to autonomously respond to a complex set of overlapping stimuli.'
'Blockchain'
'-Enabled Security: Above all else, IoT networks must be secure. Blockchain technology is primed to meet the security demands that come along with building and expanding the IoT.'
'Best New IoT Device Ideas'
'This listing of new IoT device ideas has been sub-divided according to the main technology upon which the IoT devices are built. Below I’m providing a list of new IoT device ideas, but for detailed instructions on how to build these IoT applications, I recommend the'

'IoT courses on Udemy'
'(ß Please note: if you purchase a Udemy course through this link, I may receive a small commission), or courses that are available at'
'SkyFi'
'and'
'Coursera'
'.'
'Raspberry Pi IoT Ideas'
'Using Raspberry Pi as open-source hardware, you can build IoT applications that offer any one of the following benefits:'
'Enable built-in sensing to build a weather station that measures ambient temperature and humidity'
'Build a system that detects discrepancies in electrical readings to identify electricity theft'
'Use IoT to build a Servo that is controlled by motion detection readings'
'Build a smart control switch that operates devices based on external stimuli. Use this for home automation.'
'Build a music playing application that enables music for each room in your house'
'Implement biometrics on IoT-connected devices'
'Arduino IoT Ideas'
'There are a number of new IoT device ideas that deploy Arduino as a microcontroller. These include:'
'Integrate Arduino with Android to build a remote-control RGB LED device.'
'Connect PIR sensors across the IoT to implement a smart building.'
'Build a temperature and sunlight sensor system to remotely monitor and control the conditions of your garden.'
'Deploy Arduino and IoT to automate your neighborhood streetlights.'
'Build a smart irrigation system based on IoT-connected temperature and moisture sensors built-in to your agricultural plants.'
'[caption id="attachment_3807" align="aligncenter" width="300"]'
'An IoT Chatbot Tree at the Ericsson Studio[/caption]'
'Wireless (GSM) IoT Ideas'
'Several new IoT device ideas are developed around the GSM wireless network. Those are:'
'Monitor soil moisture to automate agricultural irrigation cycles.'
'Automate and control the conditions of a greenhouse.'
'Enable bio-metrics to build a smart security system for your home or office building'
'Build an autonomously operating fitness application that automatically makes recommendations based on motion detection and heart rate sensors that are embedded on wearable fitness trackers.'
'Build a healthcare monitoring system that tracks, informs, and automatically alerts healthcare providers based on sensor readings that describe a patients vital statistics (like temperature, pulse, blood pressure, etc).'
'IoT Automation Ideas'
'Almost all new IoT device ideas offer automation benefits, but to outline a few more ideas:'

```
'Build an IoT device that automatically locates and reports the closest nearby
parking spot.'
'Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.'
'Use temperature sensors connected across the IoT to automatically alert you if
your home windows or doors have been left open.'
'Use bio-metric sensors to build a smart system that automate security for your
home or office building'
'To learn more about IoT and what's happening on the leading edge, be sure to
pop over to Ericsson's Studio Tour recap and'
'watch these videos'
'.'
'(I captured some of this content on behalf of DevMode Strategies during an
invite-only tour of the Ericsson Studio in Kista. Rest assure, the text
and\xa0opinions are my own'
')'
'…'
```

[36]:
```
first_link= our_soup_object.a
print(first_link)
```

```
<a href="http://bit.ly/LPlNDJj">Last month Ericsson Digital invited me</a>
```

[37]:
```
first_link.parent
```

[37]:
```
<strong><a href="http://bit.ly/LPlNDJj">Last month Ericsson Digital invited
me</a></strong>
```

[38]:
```
first_link.string
```

[38]:
```
'Last month Ericsson Digital invited me'
```

[39]:
```
first_link.string.parent
```

[39]:
```
<a href="http://bit.ly/LPlNDJj">Last month Ericsson Digital invited me</a>
```

1.3 Segment 3 - Data parsing

[41]:
```
import urllib
import urllib.request
with urllib.request.urlopen('https://raw.githubusercontent.com/BigDataGal/
 ↪Data-Mania-Demos/master/IoT-2018.html') as response:
    html = response.read()
```

[42]:
```
soup = BeautifulSoup(html, "lxml")
type(soup)
```

[42]:
```
bs4.BeautifulSoup
```

1.3.1 Parsing your data

[44]:
```
print(soup.prettify()[0:100])
```

```
<html>
 <head>
  <title>
   IoT Articles
  </title>
 </head>
 <body>
  <p class="title">
   <b>
```

1.3.2 Getting data from a parse tree

[46]:
```
text_only = soup.get_text()
print(text_only)
```

```
IoT Articles

2018 Trends: Best New IoT Device Ideas for Data Scientists and Engineers
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use…


It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use to get
practice in designing your first IoT applications.
Looking Back at My Coolest IoT Find in 2017
Before going into detail about best new IoT device ideas, here's the backstory.
Last month Ericsson Digital invited me to tour the Ericsson Studio in Kista,
Sweden. Up until that visit, IoT had been largely theoretical to me. Of course,
I know the usual mumbo-jumbo about wearables and IoT-connected fitness trackers.
That stuff is all well and good, but it's somewhat old hat - plus I am not sure
we are really benefiting so much from those, so I'm not that impressed.

It wasn't until I got to the Ericsson Studio that I became extremely impressed
by how far IoT has really come. Relying on the promise of the 5g network
expansion, IoT-powered smart devices are on the cusp of an explosive growth in
adoption. It was Ericsson's Smart Car that sent me reeling:

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform
that manages services for the Smart Cars to which it's connected. The Volvo
pictured above acts as a drop-off location for groceries that have been ordered
by its owner.

To understand how it works, imagine you're pulling your normal 9-to-5 and you
```

know you need to grab some groceries on your way home. Well, since you're smart you've used Ericsson IoT platform to connect your car to the local grocery delivery service (Mat.se), so all you need to do is open the Mat.se app and make your usual order. Mat.se automatically handles the payment, grocery selection, delivery, and delivery scheduling. Since your car is IoT-enabled, Mat.se issues its trusted delivery agent a 1-time token to use for opening your car in order to place your groceries in your car for you at 4:40 pm (just before you get off from work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio, make sure to go watch the videos on this page.

Future Trends for IoT in 2018

New IoT device ideas won't do you much good unless you at least know the basic technology trends that are set to impact IoT over the next year(s). These include:

Big Data & Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it's going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.

Machine Learning Data Science: While a lot of IoT devices are still operated according to rules-based decision criteria, the age of artificial intelligence is upon us. IoT will increasingly depend on machine learning algorithms to control device operations so that devices are able to autonomously respond to a complex set of overlapping stimuli.

Blockchain-Enabled Security: Above all else, IoT networks must be secure. Blockchain technology is primed to meet the security demands that come along with building and expanding the IoT.

Best New IoT Device Ideas

This listing of new IoT device ideas has been sub-divided according to the main technology upon which the IoT devices are built. Below I'm providing a list of new IoT device ideas, but for detailed instructions on how to build these IoT applications, I recommend the IoT courses on Udemy (ß Please note: if you purchase a Udemy course through this link, I may receive a small commission), or courses that are available at SkyFi and Coursera.

Raspberry Pi IoT Ideas

Using Raspberry Pi as open-source hardware, you can build IoT applications that offer any one of the following benefits:

Enable built-in sensing to build a weather station that measures ambient temperature and humidity

Build a system that detects discrepancies in electrical readings to identify electricity theft

Use IoT to build a Servo that is controlled by motion detection readings

Build a smart control switch that operates devices based on external stimuli. Use this for home automation.

Build a music playing application that enables music for each room in your house

Implement biometrics on IoT-connected devices

Arduino IoT Ideas
There are a number of new IoT device ideas that deploy Arduino as a microcontroller. These include:

Integrate Arduino with Android to build a remote-control RGB LED device.
Connect PIR sensors across the IoT to implement a smart building.
Build a temperature and sunlight sensor system to remotely monitor and control the conditions of your garden.
Deploy Arduino and IoT to automate your neighborhood streetlights.
Build a smart irrigation system based on IoT-connected temperature and moisture sensors built-in to your agricultural plants.

[caption id="attachment_3807" align="aligncenter" width="300"] An IoT Chatbot Tree at the Ericsson Studio[/caption]
Wireless (GSM) IoT Ideas
Several new IoT device ideas are developed around the GSM wireless network. Those are:

Monitor soil moisture to automate agricultural irrigation cycles.
Automate and control the conditions of a greenhouse.
Enable bio-metrics to build a smart security system for your home or office building
Build an autonomously operating fitness application that automatically makes recommendations based on motion detection and heart rate sensors that are embedded on wearable fitness trackers.
Build a healthcare monitoring system that tracks, informs, and automatically alerts healthcare providers based on sensor readings that describe a patients vital statistics (like temperature, pulse, blood pressure, etc).

IoT Automation Ideas
Almost all new IoT device ideas offer automation benefits, but to outline a few more ideas:

Build an IoT device that automatically locates and reports the closest nearby parking spot.
Build a motion detection system that automatically issues emails or sms messages to alert home owners of a likely home invasion.
Use temperature sensors connected across the IoT to automatically alert you if your home windows or doors have been left open.
Use bio-metric sensors to build a smart system that automate security for your home or office building

To learn more about IoT and what's happening on the leading edge, be sure to pop over to Ericsson's Studio Tour recap and watch these videos.

(I captured some of this content on behalf of DevMode Strategies during an

invite-only tour of the Ericsson Studio in Kista. Rest assure, the text
and opinions are my own)
…

1.3.3 Searching and retrieving data from a parse tree

Retrieving tags by filtering with name arguments

[48]:
```
soup.find_all("li")
```

[48]:
```
[<li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it's going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>,
 <li><strong>Machine Learning</strong> Data Science: While a lot of IoT devices
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>,
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>,
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>,
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>,
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>,
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>,
 <li>Build a music playing application that enables music for each room in your
house</li>,
 <li>Implement biometrics on IoT-connected devices</li>,
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>,
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>,
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>,
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>,
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>,
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>,
 <li>Automate and control the conditions of a greenhouse.</li>,
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>,
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>,
 <li>Build a healthcare monitoring system that tracks, informs, and
```

```
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure,
etc).</li>,
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>,
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>,
 <li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>,
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>]
```

Retrieving tags by filtering with keyword arguments

[50]:
```
soup.find_all(id="link 7")
```

[50]:
```
[<a class="preview" href="http://www.skyfilabs.com/iot-online-courses" id="link
7">SkyFi</a>]
```

Retrieving tags by filtering with string arguments

[51]:
```
soup.find_all('ol')
```

[51]:
```
[<ol>
 <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it's going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>
 <li><strong>Machine Learning</strong> Data Science: While a lot of IoT devices
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>
 </ol>,
 <ol>
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>
 <li>Build a music playing application that enables music for each room in your
house</li>
 <li>Implement biometrics on IoT-connected devices</li>
```

```
 </ol>,
 <ol>
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>
 </ol>,
 <ol>
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>
 <li>Automate and control the conditions of a greenhouse.</li>
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>
 <li>Build a healthcare monitoring system that tracks, informs, and
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure, etc).</li>
 </ol>,
 <ol>
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>
 <li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>
 </ol>]
```

Retrieving tags by filtering with list objects

[52]:
```
soup.find_all(['ol', 'b'])
```

[52]:
```
[<b>2018 Trends: Best New IoT Device Ideas for Data Scientists and
Engineers</b>,
 <ol>
 <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it's going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>
 <li><strong>Machine Learning</strong> Data Science: While a lot of IoT devices
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
```

```
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>
 </ol>,
 <ol>
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>
 <li>Build a music playing application that enables music for each room in your
house</li>
 <li>Implement biometrics on IoT-connected devices</li>
 </ol>,
 <ol>
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>
 </ol>,
 <ol>
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>
 <li>Automate and control the conditions of a greenhouse.</li>
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>
 <li>Build a healthcare monitoring system that tracks, informs, and
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure, etc).</li>
 </ol>,
 <ol>
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>
 <li>Use temperature sensors connected across the IoT to automatically alert you
```

```
if your home windows or doors have been left open.</li>
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>
 </ol>]
```

Retrieving tags by filtering with regular expressions

[54]:
```
import re
t = re.compile("t")
for tag in soup.find_all(t):
    print(tag.name)
```

```
html
title
strong
strong
strong
strong
strong
strong
```

[55]:
```
with urllib.request.urlopen('https://raw.githubusercontent.com/BigDataGal/
↪Data-Mania-Demos/master/IoT-2018.html') as response:
    html = response.read()
```

[56]:
```
soup = BeautifulSoup(html, "lxml")
type(soup)
```

[56]:
```
bs4.BeautifulSoup
```

1.3.4 Parsing your data

[57]:
```
print(soup.prettify()[0:100])
```

```
<html>
 <head>
  <title>
   IoT Articles
  </title>
 </head>
 <body>
  <p class="title">
   <b>
```

1.3.5 Getting data from a parse tree

[58]:
```
text_only = soup.get_text()
print(text_only)
```

```
IoT Articles

2018 Trends: Best New IoT Device Ideas for Data Scientists and Engineers
It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use…


It's almost 2018 and IoT is on the cusp of an explosive expansion. In this
article, I offer you a listing of new IoT device ideas that you can use to get
practice in designing your first IoT applications.
Looking Back at My Coolest IoT Find in 2017
Before going into detail about best new IoT device ideas, here's the backstory.
Last month Ericsson Digital invited me to tour the Ericsson Studio in Kista,
Sweden. Up until that visit, IoT had been largely theoretical to me. Of course,
I know the usual mumbo-jumbo about wearables and IoT-connected fitness trackers.
That stuff is all well and good, but it's somewhat old hat - plus I am not sure
we are really benefiting so much from those, so I'm not that impressed.

It wasn't until I got to the Ericsson Studio that I became extremely impressed
by how far IoT has really come. Relying on the promise of the 5g network
expansion, IoT-powered smart devices are on the cusp of an explosive growth in
adoption. It was Ericsson's Smart Car that sent me reeling:

This car is connected to Ericsson's Connected Vehicle Cloud, an IoT platform
that manages services for the Smart Cars to which it's connected. The Volvo
pictured above acts as a drop-off location for groceries that have been ordered
by its owner.

To understand how it works, imagine you're pulling your normal 9-to-5 and you
know you need to grab some groceries on your way home. Well, since you're smart
you've used Ericsson IoT platform to connect your car to the local grocery
delivery service (Mat.se), so all you need to do is open the Mat.se app and make
your usual order. Mat.se automatically handles the payment, grocery selection,
delivery, and delivery scheduling. Since your car is IoT-enabled, Mat.se issues
its trusted delivery agent a 1-time token to use for opening your car in order
to place your groceries in your car for you at 4:40 pm (just before you get off
from work).

To watch some of the amazing IoT device demos I witnessed at Ericsson Studio,
make sure to go watch the videos on this page.
Future Trends for IoT in 2018
New IoT device ideas won't do you much good unless you at least know the basic
technology trends that are set to impact IoT over the next year(s). These
```

include:

Big Data & Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it's going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.
Machine Learning Data Science: While a lot of IoT devices are still operated according to rules-based decision criteria, the age of artificial intelligence is upon us. IoT will increasingly depend on machine learning algorithms to control device operations so that devices are able to autonomously respond to a complex set of overlapping stimuli.
Blockchain-Enabled Security: Above all else, IoT networks must be secure. Blockchain technology is primed to meet the security demands that come along with building and expanding the IoT.

Best New IoT Device Ideas
This listing of new IoT device ideas has been sub-divided according to the main technology upon which the IoT devices are built. Below I'm providing a list of new IoT device ideas, but for detailed instructions on how to build these IoT applications, I recommend the IoT courses on Udemy (ß Please note: if you purchase a Udemy course through this link, I may receive a small commission), or courses that are available at SkyFi and Coursera.
Raspberry Pi IoT Ideas
Using Raspberry Pi as open-source hardware, you can build IoT applications that offer any one of the following benefits:

Enable built-in sensing to build a weather station that measures ambient temperature and humidity
Build a system that detects discrepancies in electrical readings to identify electricity theft
Use IoT to build a Servo that is controlled by motion detection readings
Build a smart control switch that operates devices based on external stimuli. Use this for home automation.
Build a music playing application that enables music for each room in your house
Implement biometrics on IoT-connected devices

Arduino IoT Ideas
There are a number of new IoT device ideas that deploy Arduino as a microcontroller. These include:

Integrate Arduino with Android to build a remote-control RGB LED device.
Connect PIR sensors across the IoT to implement a smart building.
Build a temperature and sunlight sensor system to remotely monitor and control the conditions of your garden.
Deploy Arduino and IoT to automate your neighborhood streetlights.
Build a smart irrigation system based on IoT-connected temperature and moisture sensors built-in to your agricultural plants.

[caption id="attachment_3807" align="aligncenter" width="300"] An IoT Chatbot Tree at the Ericsson Studio[/caption]
Wireless (GSM) IoT Ideas
Several new IoT device ideas are developed around the GSM wireless network. Those are:

Monitor soil moisture to automate agricultural irrigation cycles.
Automate and control the conditions of a greenhouse.
Enable bio-metrics to build a smart security system for your home or office building
Build an autonomously operating fitness application that automatically makes recommendations based on motion detection and heart rate sensors that are embedded on wearable fitness trackers.
Build a healthcare monitoring system that tracks, informs, and automatically alerts healthcare providers based on sensor readings that describe a patients vital statistics (like temperature, pulse, blood pressure, etc).

IoT Automation Ideas
Almost all new IoT device ideas offer automation benefits, but to outline a few more ideas:

Build an IoT device that automatically locates and reports the closest nearby parking spot.
Build a motion detection system that automatically issues emails or sms messages to alert home owners of a likely home invasion.
Use temperature sensors connected across the IoT to automatically alert you if your home windows or doors have been left open.
Use bio-metric sensors to build a smart system that automate security for your home or office building

To learn more about IoT and what's happening on the leading edge, be sure to pop over to Ericsson's Studio Tour recap and watch these videos.

(I captured some of this content on behalf of DevMode Strategies during an invite-only tour of the Ericsson Studio in Kista. Rest assure, the text and opinions are my own)
…

1.3.6 Searching and retrieving data from a parse tree

Retrieving tags by filtering with name arguments

[60]:
```
soup.find_all("li")
```

[60]: [
Big Data & Data Engineering: Sensors that are embedded within IoT devices spin off machine-generated data like it's going out of style. For IoT to function, the platform must be solidly engineered to handle big data. Be assured, that requires some serious data engineering.,
Machine Learning Data Science: While a lot of IoT devices

```
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>,
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>,
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>,
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>,
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>,
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>,
 <li>Build a music playing application that enables music for each room in your
house</li>,
 <li>Implement biometrics on IoT-connected devices</li>,
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>,
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>,
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>,
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>,
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>,
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>,
 <li>Automate and control the conditions of a greenhouse.</li>,
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>,
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>,
 <li>Build a healthcare monitoring system that tracks, informs, and
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure,
etc).</li>,
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>,
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>,
 <li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>,
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>]
```

Retrieving tags by filtering with keyword arguments

[62]:
```
soup.find_all(id="link 7")
```

[62]:
```
[<a class="preview" href="http://www.skyfilabs.com/iot-online-courses" id="link
7">SkyFi</a>]
```

Retrieving tags by filtering with string arguments

[64]:
```
soup.find_all('ol')
```

[64]:
```
[<ol>
 <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it's going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>
 <li><strong>Machine Learning</strong> Data Science: While a lot of IoT devices
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>
 </ol>,
 <ol>
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>
 <li>Build a music playing application that enables music for each room in your
house</li>
 <li>Implement biometrics on IoT-connected devices</li>
 </ol>,
 <ol>
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>
 </ol>,
 <ol>
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>
```

```
 <li>Automate and control the conditions of a greenhouse.</li>
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>
 <li>Build a healthcare monitoring system that tracks, informs, and
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure, etc).</li>
 </ol>,
 <ol>
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>
 <li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>
 </ol>]
```

Retrieving tags by filtering with list objects

[65]:
```
soup.find_all(['ol', 'b'])
```

[65]:
```
[<b>2018 Trends: Best New IoT Device Ideas for Data Scientists and
Engineers</b>,
 <ol>
 <li><strong>Big Data</strong> &amp;amp; Data Engineering: Sensors that are embedded
within IoT devices spin off machine-generated data like it’s going out of style.
For IoT to function, the platform must be solidly engineered to handle big data.
Be assured, that requires some serious data engineering.</li>
 <li><strong>Machine Learning</strong> Data Science: While a lot of IoT devices
are still operated according to rules-based decision criteria, the age of
artificial intelligence is upon us. IoT will increasingly depend on machine
learning algorithms to control device operations so that devices are able to
autonomously respond to a complex set of overlapping stimuli.</li>
 <li><strong>Blockchain</strong>-Enabled Security: Above all else, IoT networks
must be secure. Blockchain technology is primed to meet the security demands
that come along with building and expanding the IoT.</li>
 </ol>,
 <ol>
 <li>Enable built-in sensing to build a weather station that measures ambient
temperature and humidity</li>
 <li>Build a system that detects discrepancies in electrical readings to
identify electricity theft</li>
 <li>Use IoT to build a Servo that is controlled by motion detection
readings</li>
```

```
 <li>Build a smart control switch that operates devices based on external
stimuli. Use this for home automation.</li>
 <li>Build a music playing application that enables music for each room in your
house</li>
 <li>Implement biometrics on IoT-connected devices</li>
 </ol>,
 <ol>
 <li>Integrate Arduino with Android to build a remote-control RGB LED
device.</li>
 <li>Connect PIR sensors across the IoT to implement a smart building.</li>
 <li>Build a temperature and sunlight sensor system to remotely monitor and
control the conditions of your garden.</li>
 <li>Deploy Arduino and IoT to automate your neighborhood streetlights.</li>
 <li>Build a smart irrigation system based on IoT-connected temperature and
moisture sensors built-in to your agricultural plants.</li>
 </ol>,
 <ol>
 <li>Monitor soil moisture to automate agricultural irrigation cycles.</li>
 <li>Automate and control the conditions of a greenhouse.</li>
 <li>Enable bio-metrics to build a smart security system for your home or office
building</li>
 <li>Build an autonomously operating fitness application that automatically
makes recommendations based on motion detection and heart rate sensors that are
embedded on wearable fitness trackers.</li>
 <li>Build a healthcare monitoring system that tracks, informs, and
automatically alerts healthcare providers based on sensor readings that describe
a patients vital statistics (like temperature, pulse, blood pressure, etc).</li>
 </ol>,
 <ol>
 <li>Build an IoT device that automatically locates and reports the closest
nearby parking spot.</li>
 <li>Build a motion detection system that automatically issues emails or sms
messages to alert home owners of a likely home invasion.</li>
 <li>Use temperature sensors connected across the IoT to automatically alert you
if your home windows or doors have been left open.</li>
 <li>Use bio-metric sensors to build a smart system that automate security for
your home or office building</li>
 </ol>]
```

[66]:
```
##### Retrieving tags by filtering with regular expressions
```

[67]:
```
t = re.compile("t")
for tag in soup.find_all(t):
    print(tag.name)
```

```
html
title
strong
```

```
strong
strong
strong
strong
strong
```

Retrieving tags by filtering with a Boolean value

[69]:
```
for tag in soup.find_all(True):
    print(tag.name)
```

```
html
head
title
body
p
b
p
br
br
h1
span
strong
a
a
a
img
a
span
strong
a
h1
ol
li
strong
li
strong
li
strong
h1
a
a
a
h2
ol
li
li
li
li
```

```
li
li
h2
ol
li
li
li
li
li
a
img
h2
ol
li
li
li
li
li
h2
ol
li
li
li
li
span
strong
a
em
p
```

Retrieving weblinks by filtering with string objects

[70]:
```
for link in soup.find_all('a'):
    print(link.get('href'))
```

```
http://bit.ly/LPlNDJj
http://www.data-mania.com/blog/m2m-vs-iot/
bit.ly/LPlNDJj
http://mat.se/
http://bit.ly/LPlNDJj
https://click.linksynergy.com/deeplink?id=*JDLXjeE*wk&mid=39197&murl=https%3A%2F
%2Fwww.udemy.com%2Ftopic%2Finternet-of-things%2F%3Fsort%3Dhighest-rated
http://www.skyfilabs.com/iot-online-courses
https://www.coursera.org/specializations/iot
bit.ly/LPlNDJj
http://bit.ly/LPlNDJj
```

Retrieving strings by filtering with regular expressions

[72]:
```
soup.find_all(string=re.compile("data"))
```

[72]:
```
[' & Data Engineering: Sensors that are embedded within IoT devices spin off
machine-generated data like it's going out of style. For IoT to function, the
platform must be solidly engineered to handle big data. Be assured, that
requires some serious data engineering.']
```

1.4 Segment 4 - Web scraping

[73]:
```
from IPython.display import HTML
```

[74]:
```
r = urllib.request.urlopen('https://analytics.usa.gov/').read()
soup = BeautifulSoup(r, "lxml")
type(soup)
```

[74]:
```
bs4.BeautifulSoup
```

[75]:
```
print(soup.prettify()[:100])
```

```
<!DOCTYPE html>
<html lang="en">
 <!-- Initalize title and data source variables -->
 <head>
  <!--
```

[76]:
```
for link in soup.find_all('a'):
    print(link.get('href'))
```

```
/
#explanation
https://analytics.usa.gov/data/
https://open.gsa.gov/api/dap/
data/
#top-pages-realtime
#top-pages-7-days
#top-pages-30-days
https://analytics.usa.gov/data/live/all-pages-realtime.csv
https://analytics.usa.gov/data/live/all-domains-30-days.csv
https://www.digitalgov.gov/services/dap/
https://www.digitalgov.gov/services/dap/common-questions-about-dap-faq/#part-4
https://support.google.com/analytics/answer/2763052?hl=en
https://analytics.usa.gov/data/live/second-level-domains.csv
https://analytics.usa.gov/data/live/sites.csv
mailto:dap@gsa.gov
https://analytics.usa.gov/data/
https://open.gsa.gov/api/dap/
mailto:dap@gsa.gov
```

```
https://github.com/GSA/analytics.usa.gov/issues
https://github.com/GSA/analytics.usa.gov
https://github.com/18F/analytics-reporter
http://www.gsa.gov/
https://www.digital.gov/guides/dap/
https://cloud.gov/
```

[77]:
```
print(soup.get_text())
```

```
analytics.usa.gov | The US government's web traffic.
```

```
analytics.usa.gov
```

About this site
Data | API

Select an agency

All Participating Websites
Agency for International Development
Department of Agriculture
Department of Commerce
Department of Defense
Department of Education
Department of Energy
Department of Health and Human Services
Department of Homeland Security
Department of Housing and Urban Development
Department of Justice
Department of Labor
Department of State
Department of Transportation
Department of Veterans Affairs
Department of the Interior
Department of the Treasury
Environmental Protection Agency
Executive Office of the President
General Services Administration
National Aeronautics and Space Administration
National Archives and Records Administration
National Science Foundation
Nuclear Regulatory Commission
Office of Personnel Management
Postal Service
Small Business Administration
Social Security Administration

…

people on government websites now

Visits Today
Eastern Time

Visits in the Past 90 Days

There were … visits over the past 90 days.

Devices

Based on rough network segmentation data, we estimate that less than 5% of all traffic across all agencies comes from US federal government networks.

Much more detailed data is available in downloadable CSV and JSON. This includes data on combined browser and OS usage.

Browsers

Internet Explorer

Operating Systems

Windows

Visitor Locations Right Now

Cities

Countries

United States & Territories

International

Top Pages

Now
7 Days
30 Days

People on a single, specific page now. We only count pages with at least 10 people on the page.

Download the full dataset.

Visits over the last week to domains, including traffic to all pages within that domain.

Visits over the last month to domains, including traffic to all pages within that domain. We only count pages with at least 1,000 visits in the last month.

Download the full dataset.

Top Downloads

Total file downloads yesterday on government domains.

About this Site

These data provide a window into how people are interacting with the government online.

The data come from a unified Google Analytics account for U.S. federal government agencies known as the Digital Analytics Program.

This program helps government agencies understand how people find, access, and use government services online. The program does not track individuals,

and anonymizes the IP addresses of visitors.

Not every government website is represented in these data.

Currently, the Digital Analytics Program collects web traffic from around 400 executive branch government domains,
across about 5,700 total websites,
including every cabinet department.
We continue to pursue and add more sites frequently; to add your site, email the Digital Analytics Program.

Download the data

You can download the data here. Available in JSON and CSV format.
Additionally, you can access data via our API project (currently in Beta).

A note on sampling

Due to varying Google Analytics API sampling thresholds and the sheer volume of data in this project, some non-realtime reports may be subject to sampling.
The data are intended to represent trends and numbers may not be precise.

Have a question or problem?

Get in touch.

Suggest a feature or report an issue

View our code on GitHub

View our code for the data on GitHub

```
Analytics.usa.gov is a project of GSA's Digital Analytics Program.
This website is hosted on cloud.gov.
```

[78]:
```
print(soup.prettify()[0:1000])
```

```
<!DOCTYPE html>
<html lang="en">
 <!-- Initalize title and data source variables -->
 <head>
  <!--

    Hi! Welcome to our source code.

    This dashboard uses data from the Digital Analytics Program, a US
    government team inside the General Services Administration.


    For a detailed tech breakdown of how 18F and friends built this site:

    https://18f.gsa.gov/2015/03/19/how-we-built-analytics-usa-gov/


    This is a fully open source project, and your contributions are welcome.

    Frontend static site: https://github.com/18F/analytics.usa.gov
    Backend data reporting: https://github.com/18F/analytics-reporter

    -->
  <meta charset="utf-8"/>
  <meta content="IE=Edge" http-equiv="X-UA-Compatible"/>
  <meta content="NjbZn6hQe7OwV-nTsa6nLmtrOUcSGPRyFjxm5zkmCcg" name="google-site-
verification"/>
  <link href="/css/vendor/css/uswds.v0.9.6.css" rel="stylesheet"/>
  <link href="/css/public_analytics.css" rel="stylesheet"/>
  <link href="/images/analytics-favicon.ico" rel="ic
```

[79]:
```
for link in soup.findAll('a', attrs={'href': re.compile("^http")}):
    print(link)
type(link)
```

```
<a href="https://analytics.usa.gov/data/">Data</a>
<a href="https://open.gsa.gov/api/dap/" rel="noopener" target="_blank">API</a>
<a href="https://analytics.usa.gov/data/live/all-pages-realtime.csv">Download
the full dataset.</a>
<a href="https://analytics.usa.gov/data/live/all-domains-30-days.csv">Download
the full dataset.</a>
<a class="external-link" href="https://www.digitalgov.gov/services/dap/">Digital
Analytics Program</a>
<a class="external-link" href="https://www.digitalgov.gov/services/dap/common-
questions-about-dap-faq/#part-4">does not track individuals</a>
<a class="external-link"
href="https://support.google.com/analytics/answer/2763052?hl=en">anonymizes the
IP addresses</a>
<a class="external-link" href="https://analytics.usa.gov/data/live/second-level-
domains.csv">400 executive branch government domains</a>
<a class="external-link"
href="https://analytics.usa.gov/data/live/sites.csv">about 5,700 total
websites</a>
<a href="https://analytics.usa.gov/data/">download the data here.</a>
<a href="https://open.gsa.gov/api/dap/" rel="noopener" target="_blank"> API
project</a>
<a class="usa-button usa-button-secondary-inverse"
href="https://github.com/GSA/analytics.usa.gov/issues">
<img alt="Github Icon" class="github-icon" src="/images/github-logo-white.svg"/>
                  Suggest a feature or report an issue
            </a>
<a href="https://github.com/GSA/analytics.usa.gov">
<img alt="Github Icon" class="github-icon" src="/images/github-logo.svg"/>
              View our code on GitHub</a>
<a href="https://github.com/18F/analytics-reporter">
<img alt="Github Icon" class="github-icon" src="/images/github-logo.svg"/>
              View our code for the data on GitHub</a>
<a href="http://www.gsa.gov/">
<img alt="GSA" src="/images/gsa-logo.svg"/>
</a>
<a href="https://www.digital.gov/guides/dap/">Digital Analytics Program</a>
<a href="https://cloud.gov/">cloud.gov</a>
```

[79]:
```
bs4.element.Tag
```

[80]:
```
file = open("parsed_data.txt", "w")
for link in soup.findAll('a', attrs={'href': re.compile("^http")}):
    soup_link = str(link)
    print(soup_link)
```

```
    file.write(soup_link)
file.flush()
file.close()
```

```
<a href="https://analytics.usa.gov/data/">Data</a>
<a href="https://open.gsa.gov/api/dap/" rel="noopener" target="_blank">API</a>
<a href="https://analytics.usa.gov/data/live/all-pages-realtime.csv">Download
the full dataset.</a>
<a href="https://analytics.usa.gov/data/live/all-domains-30-days.csv">Download
the full dataset.</a>
<a class="external-link" href="https://www.digitalgov.gov/services/dap/">Digital
Analytics Program</a>
<a class="external-link" href="https://www.digitalgov.gov/services/dap/common-
questions-about-dap-faq/#part-4">does not track individuals</a>
<a class="external-link"
href="https://support.google.com/analytics/answer/2763052?hl=en">anonymizes the
IP addresses</a>
<a class="external-link" href="https://analytics.usa.gov/data/live/second-level-
domains.csv">400 executive branch government domains</a>
<a class="external-link"
href="https://analytics.usa.gov/data/live/sites.csv">about 5,700 total
websites</a>
<a href="https://analytics.usa.gov/data/">download the data here.</a>
<a href="https://open.gsa.gov/api/dap/" rel="noopener" target="_blank"> API
project</a>
<a class="usa-button usa-button-secondary-inverse"
href="https://github.com/GSA/analytics.usa.gov/issues">
<img alt="Github Icon" class="github-icon" src="/images/github-logo-white.svg"/>
                  Suggest a feature or report an issue
            </a>
<a href="https://github.com/GSA/analytics.usa.gov">
<img alt="Github Icon" class="github-icon" src="/images/github-logo.svg"/>
              View our code on GitHub</a>
<a href="https://github.com/18F/analytics-reporter">
<img alt="Github Icon" class="github-icon" src="/images/github-logo.svg"/>
              View our code for the data on GitHub</a>
<a href="http://www.gsa.gov/">
<img alt="GSA" src="/images/gsa-logo.svg"/>
</a>
<a href="https://www.digital.gov/guides/dap/">Digital Analytics Program</a>
<a href="https://cloud.gov/">cloud.gov</a>
```

[81]:
```
%pwd
```

[81]:
```
'C:\\Users\\aadar\\Documents\\TERM2\\BDM 1034 - Application Design for Big
Data\\Week6\\Assignment'
```

1.5 Segment 5 - Introduction to NLP

[83]:
```
text = "On Wednesday, the Association for Computing Machinery, the world's␣
↪largest society of computing professionals, announced that Hinton, LeCun and␣
↪Bengio had won this year's Turing Award for their work on neural networks.␣
↪The Turing Award, which was introduced in 1966, is often called the Nobel␣
↪Prize of computing, and it includes a $1 million prize, which the three␣
↪scientists will share."
```

[85]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\aadar\AppData\Roaming\nltk_data…
[nltk_data]   Package punkt is already up-to-date!
```

[85]:
```
True
```

Sentence Tokenizer

[87]:
```
from nltk.tokenize import sent_tokenize
sent_tk = sent_tokenize(text)
print("Sentence tokenizing the text: \n")
print(sent_tk)
```

```
Sentence tokenizing the text:

['On Wednesday, the Association for Computing Machinery, the world's largest
society of computing professionals, announced that Hinton, LeCun and Bengio had
won this year's Turing Award for their work on neural networks.', 'The Turing
Award, which was introduced in 1966, is often called the Nobel Prize of
computing, and it includes a $1 million prize, which the three scientists will
share.']
```

1.5.1 Word Tokenizer

[89]:
```
from nltk.tokenize import word_tokenize
word_tk = word_tokenize(text)
print("Word tokenizing the text: \n")
print(word_tk)
```

```
Word tokenizing the text:

['On', 'Wednesday', ',', 'the', 'Association', 'for', 'Computing', 'Machinery',
',', 'the', 'world', ''', 's', 'largest', 'society', 'of', 'computing',
'professionals', ',', 'announced', 'that', 'Hinton', ',', 'LeCun', 'and',
'Bengio', 'had', 'won', 'this', 'year', ''', 's', 'Turing', 'Award', 'for',
'their', 'work', 'on', 'neural', 'networks', '.', 'The', 'Turing', 'Award', ',',
'which', 'was', 'introduced', 'in', '1966', ',', 'is', 'often', 'called', 'the',
'Nobel', 'Prize', 'of', 'computing', ',', 'and', 'it', 'includes', 'a', '$',
```

```
'1', 'million', 'prize', ',', 'which', 'the', 'three', 'scientists', 'will',
'share', '.']
```

1.5.2 Removing stop words

[91]:
```
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\aadar\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[91]:
```
True
```

[92]:
```
from nltk.corpus import stopwords

sw = set(stopwords.words("english"))
print("Stop words in English language are: \n")
print(sw)
```

```
Stop words in English language are:

{'were', 'once', 'did', 'can', 't', 'having', 'own', 'hadn', 'just', "aren't",
'in', 'am', "mustn't", 'himself', 'have', 'wouldn', 'won', 'for', "don't", 've',
'but', 'my', 'm', "mightn't", 'here', "should've", 'shouldn', "won't", 'been',
"you'd", 'an', 'now', 'needn', 'below', 'out', 'yours', 'by', 'herself', 'more',
'aren', 'about', 'll', 'should', 'which', 'doesn', 'your', 'both', 'how',
'yourself', 'her', 'we', 'they', 'this', 'and', "doesn't", 'mustn', 'most',
"needn't", 'our', 'haven', 'same', "hasn't", 'being', 'his', 'of', 'are', 'a',
'she', 'me', 'from', "couldn't", "you'll", 'him', 'while', 'y', 'so', 'had',
"isn't", 'ours', 'it', 're', 'shan', 'into', 'these', 'weren', "you're",
'because', 'over', 'or', 'o', 'he', 'theirs', 'on', 'up', "haven't", 'with',
'be', 'ma', 'some', 'only', 'when', 'ain', "hadn't", 'does', 'mightn', 'again',
'what', 'yourselves', 'above', 'very', 'itself', 'no', 'off', "weren't", 'at',
'if', 's', 'do', "it's", 'i', "wouldn't", 'down', 'doing', 'there', 'their',
'the', 'd', 'further', 'through', 'is', 'other', 'ourselves', 'them', 'after',
'to', 'than', 'any', 'until', 'you', 'don', 'that', 'few', 'during', 'where',
"that'll", 'has', 'themselves', 'why', 'all', 'was', 'between', 'didn',
"didn't", 'hers', 'its', 'who', 'before', 'under', 'each', 'such', 'as',
'those', 'too', 'wasn', "wasn't", 'hasn', "shan't", "she's", 'against', 'isn',
"shouldn't", 'myself', "you've", 'nor', 'not', 'then', 'whom', 'will', 'couldn'}
```

[93]:
```
filtered_words = [w for w in word_tk if not w in sw]

print("The text after removing stop words \n")
print(filtered_words)
```

```
The text after removing stop words

['On', 'Wednesday', ',', 'Association', 'Computing', 'Machinery', ',', 'world',
```

```
'’', 'largest', 'society', 'computing', 'professionals', ',', 'announced',
'Hinton', ',', 'LeCun', 'Bengio', 'year', '’', 'Turing', 'Award', 'work',
'neural', 'networks', '.', 'The', 'Turing', 'Award', ',', 'introduced', '1966',
',', 'often', 'called', 'Nobel', 'Prize', 'computing', ',', 'includes', '$',
'1', 'million', 'prize', ',', 'three', 'scientists', 'share', '.']
```

Stemming

[95]:
```
from nltk.stem import PorterStemmer
from nltk.tokenize import sent_tokenize, word_tokenize

port_stem = PorterStemmer()
```

[96]:
```
stemmed_words = []

for w in filtered_words:
    stemmed_words.append(port_stem.stem(w))

print("Filtered Sentence: \n", filtered_words, "\n")
print("Stemmed Sentence: \n", stemmed_words)
```

```
Filtered Sentence:
 ['On', 'Wednesday', ',', 'Association', 'Computing', 'Machinery', ',', 'world',
'’', 'largest', 'society', 'computing', 'professionals', ',', 'announced',
'Hinton', ',', 'LeCun', 'Bengio', 'year', '’', 'Turing', 'Award', 'work',
'neural', 'networks', '.', 'The', 'Turing', 'Award', ',', 'introduced', '1966',
',', 'often', 'called', 'Nobel', 'Prize', 'computing', ',', 'includes', '$',
'1', 'million', 'prize', ',', 'three', 'scientists', 'share', '.']

Stemmed Sentence:
 ['on', 'wednesday', ',', 'associ', 'comput', 'machineri', ',', 'world', '’',
'largest', 'societi', 'comput', 'profession', ',', 'announc', 'hinton', ',',
'lecun', 'bengio', 'year', '’', 'ture', 'award', 'work', 'neural', 'network',
'.', 'the', 'ture', 'award', ',', 'introduc', '1966', ',', 'often', 'call',
'nobel', 'prize', 'comput', ',', 'includ', '$', '1', 'million', 'prize', ',',
'three', 'scientist', 'share', '.']
```

2 Lemmatizing

[98]:
```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\aadar\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[98]: True

[99]:
```
from nltk.stem.wordnet import WordNetLemmatizer

lem = WordNetLemmatizer()

from nltk.stem.porter import PorterStemmer
stem = PorterStemmer()

lemm_words = []

for i in range(len(filtered_words)):
    lemm_words.append(lem.lemmatize(filtered_words[i]))

print(lemm_words)
```

```
['On', 'Wednesday', ',', 'Association', 'Computing', 'Machinery', ',', 'world',
'’', 'largest', 'society', 'computing', 'professional', ',', 'announced',
'Hinton', ',', 'LeCun', 'Bengio', 'year', '’', 'Turing', 'Award', 'work',
'neural', 'network', '.', 'The', 'Turing', 'Award', ',', 'introduced', '1966',
',', 'often', 'called', 'Nobel', 'Prize', 'computing', ',', 'includes', '$',
'1', 'million', 'prize', ',', 'three', 'scientist', 'share', '.']
```

Parts of Speech Tagging

[101]:
```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\aadar\AppData\Roaming\nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

[101]:
```
True
```

[102]:
```
from nltk import pos_tag
pos_tagged_words = pos_tag(word_tk)

print(pos_tagged_words)
```

```
[('On', 'IN'), ('Wednesday', 'NNP'), (',', ','), ('the', 'DT'), ('Association',
'NNP'), ('for', 'IN'), ('Computing', 'VBG'), ('Machinery', 'NNP'), (',', ','),
('the', 'DT'), ('world', 'NN'), ('’', 'NNP'), ('s', 'RB'), ('largest', 'JJS'),
('society', 'NN'), ('of', 'IN'), ('computing', 'VBG'), ('professionals', 'NNS'),
(',', ','), ('announced', 'VBD'), ('that', 'IN'), ('Hinton', 'NNP'), (',', ','),
('LeCun', 'NNP'), ('and', 'CC'), ('Bengio', 'NNP'), ('had', 'VBD'), ('won',
'VBN'), ('this', 'DT'), ('year', 'NN'), ('’', 'VBZ'), ('s', 'JJ'), ('Turing',
'NNP'), ('Award', 'NNP'), ('for', 'IN'), ('their', 'PRP$'), ('work', 'NN'),
('on', 'IN'), ('neural', 'JJ'), ('networks', 'NNS'), ('.', '.'), ('The', 'DT'),
('Turing', 'NNP'), ('Award', 'NNP'), (',', ','), ('which', 'WDT'), ('was',
'VBD'), ('introduced', 'VBN'), ('in', 'IN'), ('1966', 'CD'), (',', ','), ('is',
'VBZ'), ('often', 'RB'), ('called', 'VBN'), ('the', 'DT'), ('Nobel', 'NNP'),
```

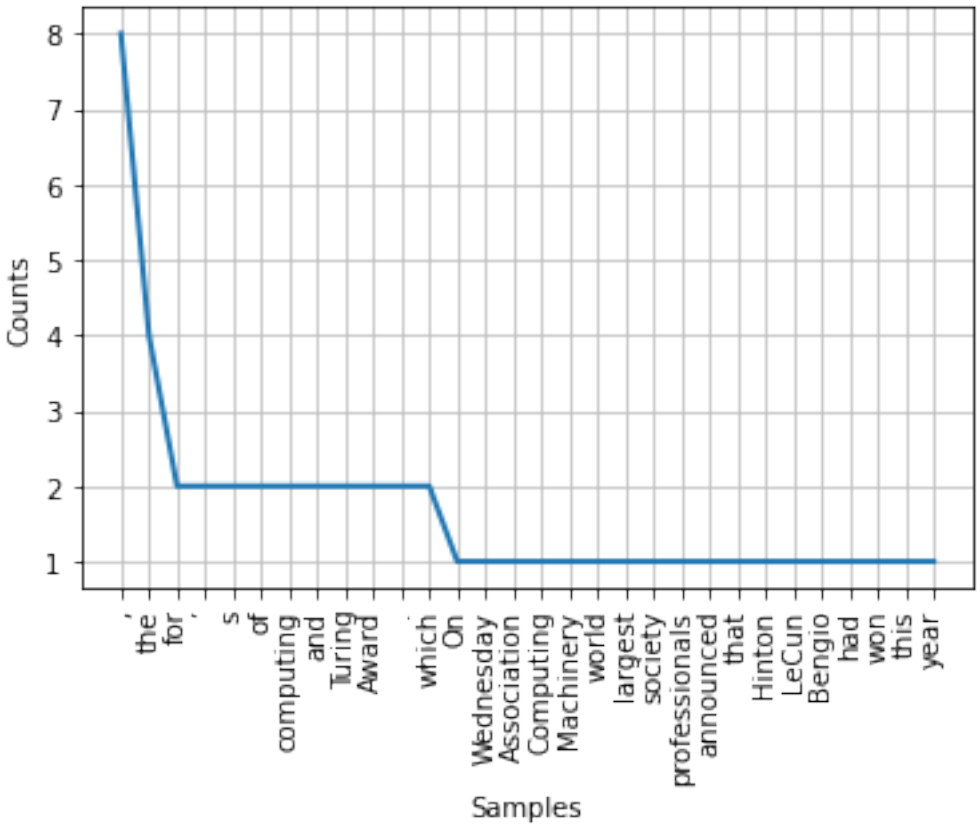
```
('Prize', 'NNP'), ('of', 'IN'), ('computing', 'NN'), (',', ','), ('and', 'CC'),
('it', 'PRP'), ('includes', 'VBZ'), ('a', 'DT'), ('$', '$'), ('1', 'CD'),
('million', 'CD'), ('prize', 'NN'), (',', ','), ('which', 'WDT'), ('the', 'DT'),
('three', 'CD'), ('scientists', 'NNS'), ('will', 'MD'), ('share', 'NN'), ('.',
'.')]
```

Frequency Distribution Plots

[104]:
```
from nltk.probability import FreqDist
fd = FreqDist(word_tk)
print(fd)
```

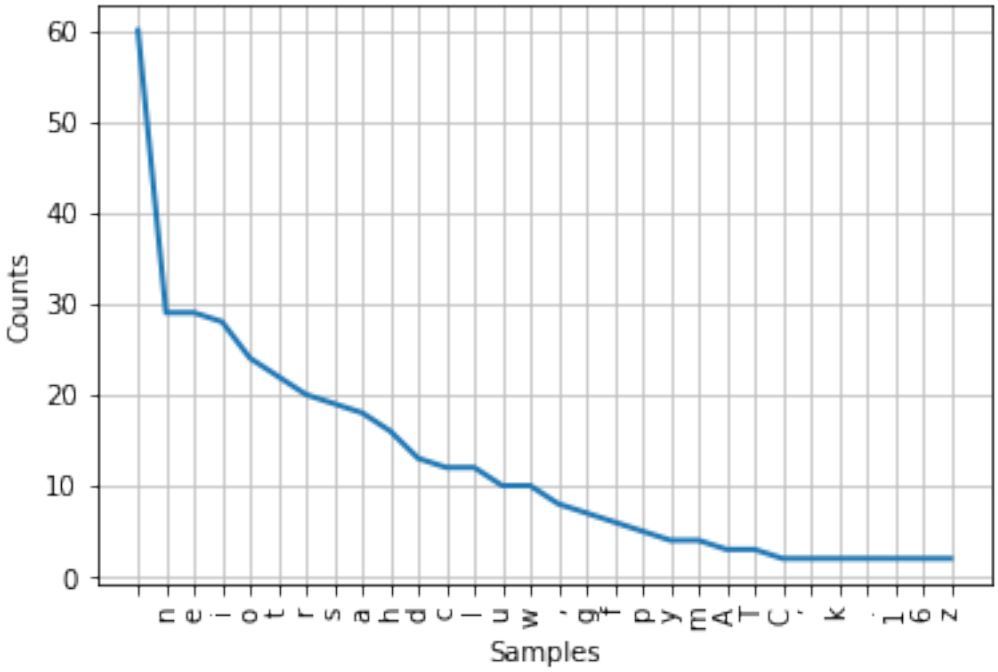
```
<FreqDist with 56 samples and 76 outcomes>
```

[105]:
```
import matplotlib.pyplot as plt
fd.plot(30, cumulative=False)
plt.show()
```



[106]:
```
fd_alpha = FreqDist(text)
print(fd_alpha)
fd_alpha.plot(30, cumulative=False)
```

```
<FreqDist with 41 samples and 387 outcomes>
```



[106]: <AxesSubplot:xlabel='Samples', ylabel='Counts'>

[ ]:

Chapter7

March 2, 2022

1 Chapter 7 - Collaborative Analytics with Plotly

1.1 Segment 1 - Creating basic charts

1.1.1 Setting up to use Plotly within Jupyter

[8]:
```
! pip install Plotly
```

```
Requirement already satisfied: Plotly in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (5.6.0)
Requirement already satisfied: six in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from Plotly)
(1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from Plotly)
(8.0.1)
```

[12]:
```
! pip install cufflinks
!pip install chart_studio
```

```
Requirement already satisfied: cufflinks in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (0.17.3)
Requirement already satisfied: pandas>=0.19.2 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(1.4.1)
Requirement already satisfied: numpy>=1.9.2 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(1.22.2)
Requirement already satisfied: six>=1.9.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(1.16.0)
Requirement already satisfied: colorlover>=0.2.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(0.3.0)
Requirement already satisfied: plotly>=4.1.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(5.6.0)
Requirement already satisfied: ipywidgets>=7.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
```

```
(7.6.5)
Requirement already satisfied: setuptools>=34.4.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(58.0.4)
Requirement already satisfied: ipython>=5.3.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from cufflinks)
(8.1.0)
Requirement already satisfied: colorama in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.4.4)
Requirement already satisfied: backcall in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.2.0)
Requirement already satisfied: pickleshare in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.7.5)
Requirement already satisfied: pygments>=2.4.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (2.11.2)
Requirement already satisfied: stack-data in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.2.0)
Requirement already satisfied: matplotlib-inline in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.1.3)
Requirement already satisfied: jedi>=0.16 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (0.18.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (3.0.28)
Requirement already satisfied: traitlets>=5 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (5.1.1)
Requirement already satisfied: decorator in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipython>=5.3.0->cufflinks) (5.1.1)
Requirement already satisfied: ipython-genutils~=0.2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipywidgets>=7.0.0->cufflinks) (0.2.0)
Requirement already satisfied: nbformat>=4.2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipywidgets>=7.0.0->cufflinks) (5.1.3)
Requirement already satisfied: ipykernel>=4.5.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipywidgets>=7.0.0->cufflinks) (6.9.1)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
```

```
ipywidgets>=7.0.0->cufflinks) (1.0.2)
Requirement already satisfied: widgetsnbextension~=3.5.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipywidgets>=7.0.0->cufflinks) (3.5.2)
Requirement already satisfied: nest-asyncio in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (1.5.4)
Requirement already satisfied: tornado<7.0,>=4.2 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (6.1)
Requirement already satisfied: debugpy<2.0,>=1.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (1.5.1)
Requirement already satisfied: jupyter-client<8.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (7.1.2)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
jedi>=0.16->ipython>=5.3.0->cufflinks) (0.8.3)
Requirement already satisfied: entrypoints in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jupyter-
client<8.0->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (0.4)
Requirement already satisfied: jupyter-core>=4.6.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jupyter-
client<8.0->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (4.9.2)
Requirement already satisfied: pyzmq>=13 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jupyter-
client<8.0->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (22.3.0)
Requirement already satisfied: python-dateutil>=2.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jupyter-
client<8.0->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (2.8.2)
Requirement already satisfied: pywin32>=1.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jupyter-
core>=4.6.0->jupyter-client<8.0->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks)
(303)
Requirement already satisfied: jsonschema!=2.5.0,>=2.4 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
nbformat>=4.2.0->ipywidgets>=7.0.0->cufflinks) (4.4.0)
Requirement already satisfied: pyrsistent!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets>=7.0.0->cufflinks) (0.18.1)
Requirement already satisfied: attrs>=17.4.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets>=7.0.0->cufflinks) (21.4.0)
Requirement already satisfied: importlib-resources>=1.4.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets>=7.0.0->cufflinks) (5.4.0)
Requirement already satisfied: zipp>=3.1.0 in
```

```
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from importlib-reso
urces>=1.4.0->jsonschema!=2.5.0,>=2.4->nbformat>=4.2.0->ipywidgets>=7.0.0->cuffl
inks) (3.7.0)
Requirement already satisfied: pytz>=2020.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
pandas>=0.19.2->cufflinks) (2021.3)
Requirement already satisfied: tenacity>=6.2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
plotly>=4.1.1->cufflinks) (8.0.1)
Requirement already satisfied: wcwidth in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from prompt-
toolkit!=3.0.0,!=3.0.1,<3.1.0,>=2.0.0->ipython>=5.3.0->cufflinks) (0.2.5)
Requirement already satisfied: notebook>=4.4.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (6.4.8)
Requirement already satisfied: terminado>=0.8.3 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(0.13.1)
Requirement already satisfied: argon2-cffi in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(21.3.0)
Requirement already satisfied: prometheus-client in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(0.13.1)
Requirement already satisfied: nbconvert in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(6.4.2)
Requirement already satisfied: Send2Trash>=1.8.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(1.8.0)
Requirement already satisfied: jinja2 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(3.0.3)
Requirement already satisfied: pywinpty>=1.1.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from terminado>=0.8
.3->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(2.0.3)
Requirement already satisfied: argon2-cffi-bindings in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from argon2-cffi->n
otebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(21.2.0)
Requirement already satisfied: cffi>=1.0.1 in
```

```
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from argon2-cffi-bi
ndings->argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0
.0->cufflinks) (1.15.0)
Requirement already satisfied: pycparser in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbextens
ion~=3.5.0->ipywidgets>=7.0.0->cufflinks) (2.21)
Requirement already satisfied: MarkupSafe>=2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from jinja2->notebo
ok>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (2.1.0)
Requirement already satisfied: defusedxml in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (0.7.1)
Requirement already satisfied: bleach in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (4.1.0)
Requirement already satisfied: nbclient<0.6.0,>=0.5.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (0.5.11)
Requirement already satisfied: jupyterlab-pygments in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (0.1.2)
Requirement already satisfied: mistune<2,>=0.8.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (0.8.4)
Requirement already satisfied: pandocfilters>=1.4.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (1.5.0)
Requirement already satisfied: testpath in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from nbconvert->not
ebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks) (0.6.0)
Requirement already satisfied: packaging in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from bleach->nbconv
ert->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(21.3)
Requirement already satisfied: webencodings in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from bleach->nbconv
ert->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->cufflinks)
(0.5.1)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from packaging->ble
ach->nbconvert->notebook>=4.4.1->widgetsnbextension~=3.5.0->ipywidgets>=7.0.0->c
ufflinks) (3.0.7)
Requirement already satisfied: asttokens in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from stack-
data->ipython>=5.3.0->cufflinks) (2.0.5)
Requirement already satisfied: executing in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from stack-
```

```
data->ipython>=5.3.0->cufflinks) (0.8.3)
Requirement already satisfied: pure-eval in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from stack-
data->ipython>=5.3.0->cufflinks) (0.2.2)
Requirement already satisfied: chart_studio in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (1.1.0)
Requirement already satisfied: plotly in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from chart_studio)
(5.6.0)
Requirement already satisfied: retrying>=1.3.3 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from chart_studio)
(1.3.3)
Requirement already satisfied: requests in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from chart_studio)
(2.27.1)
Requirement already satisfied: six in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from chart_studio)
(1.16.0)
Requirement already satisfied: tenacity>=6.2.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
plotly->chart_studio) (8.0.1)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
requests->chart_studio) (1.26.8)
Requirement already satisfied: charset-normalizer~=2.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
requests->chart_studio) (2.0.12)
Requirement already satisfied: certifi>=2017.4.17 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
requests->chart_studio) (2021.10.8)
Requirement already satisfied: idna<4,>=2.5 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from
requests->chart_studio) (3.3)
```

[15]:
```
import numpy as np
import pandas as pd

import cufflinks as cf

# import plotly.plotly as py
import chart_studio.plotly as py
# import plotly.tools as tls
import chart_studio.tools as tls
import plotly.graph_objs as go
```

[16]:
```
tls.set_credentials_file(username='aadarsha', api_key='0qf5rfEp17pSmDg2fekH')
```

1.1.2 Creating line charts

A very basic line chart

[17]:
```
a = np.linspace(start=0, stop=36, num=36)

np.random.seed(25)

b = np.random.uniform(low=0.0, high=1.0, size=36)

trace = go.Scatter(x=a, y=b)

data = [trace]

py.iplot(data, filename='basic-line-chart')
```

[17]:
```
<IPython.lib.display.IFrame at 0x205ccf32220>
```

A line chart from a pandas dataframe

[18]:
```
address = './Data/mtcars.csv'

cars = pd.read_csv(address)
cars.columns = ['car_names','mpg','cyl','disp', 'hp', 'drat', 'wt', 'qsec',␣
 ↪'vs', 'am', 'gear', 'carb']

df = cars[['cyl', 'wt','mpg']]

layout = dict(title='Chart from Pandas DataFrame', xaxis= dict(title='x-axis'),␣
 ↪yaxis= dict(title='y-axis'))

df.iplot(filename='cf-simple-line-chart', layout=layout)
```

[18]:
```
<IPython.lib.display.IFrame at 0x205ccf32ac0>
```

1.1.3 Creating bar charts

[19]:
```
data =[go.Bar(x=[1,2,3,4,5,6,7,8,9,10], y=[1,2,3,4,0.5,4,3,2,1])]
print(data)
```

```
[Bar({
    'x': [1, 2, 3, 4, 5, 6, 7, 8, 9, 10], 'y': [1, 2, 3, 4, 0.5, 4, 3, 2, 1]
})]
```

[22]:
```
layout = dict(title='Simple Bar Chart',
              xaxis= dict(title='x-axis'), yaxis=dict(title='y-axis'))
py.iplot(data, filename='basic-bar-chart',layout=layout)
```

[22]:
```
<IPython.lib.display.IFrame at 0x205ccf050d0>
```

1.1.4 Creating pie charts

[23]:
```
fig = {'data':[{'labels': ['bicycle', 'motorbike','car','van', 'stroller'],
                'values': [1, 2, 3, 4, 0.5],'type': 'pie'}],
       'layout': {'title': 'Simple Pie Chart'}}
py.iplot(fig)
```

[23]:
```
<IPython.lib.display.IFrame at 0x205cc2708b0>
```

1.2 Segment 2 - Creating statistical charts

Setting up to use Plotly within Jupyter

[28]:
```
!pip install sklearn
import sklearn
from sklearn.preprocessing import StandardScaler
```

```
Requirement already satisfied: sklearn in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (0.0)
Requirement already satisfied: scikit-learn in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from sklearn)
(1.0.2)
Requirement already satisfied: joblib>=0.11 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from scikit-
learn->sklearn) (1.1.0)
Requirement already satisfied: numpy>=1.14.6 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from scikit-
learn->sklearn) (1.22.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from scikit-
learn->sklearn) (3.1.0)
Requirement already satisfied: scipy>=1.1.0 in
e:\anaconda\envs\applicationdesignbdm1034\lib\site-packages (from scikit-
learn->sklearn) (1.8.0)
```

1.2.1 Creating histograms

Make a histogram from a pandas Series object

[29]:
```
mpg = cars.mpg

mpg.iplot(kind='histogram', filename='simple-histogram-chart')
```

[29]:
```
<IPython.lib.display.IFrame at 0x205dd1c7be0>
```

[30]:
```
cars_subset = cars[['mpg', 'disp', 'hp']]

cars_data_std = StandardScaler().fit_transform(cars_subset)

cars_select = pd.DataFrame(cars_data_std)
cars_select.columns = ['mpg', 'disp', 'hp']
```

```
cars_select.iplot(kind='histogram', filename= 'multiple-histogram-chart')
```

[30]: <IPython.lib.display.IFrame at 0x205dd27a1c0>

[31]:
```
cars_select.iplot(kind='histogram', subplots=True, filename=␣
↪'subplot-histograms')
```

[31]: <IPython.lib.display.IFrame at 0x205cce73eb0>

[32]:
```
cars_select.iplot(kind='histogram', subplots=True, shape=(3,1), filename=␣
↪'subplot-histograms')
```

[32]: <IPython.lib.display.IFrame at 0x205dd19ee20>

[33]:
```
cars_select.iplot(kind='histogram', subplots=True, shape=(1,3), filename=␣
↪'subplot-histograms')
```

[33]: <IPython.lib.display.IFrame at 0x205dd8e4460>

1.2.2 Creating box plots

[35]:
```
cars_select.iplot(kind='box', filename= 'box-plots')
```

[35]: <IPython.lib.display.IFrame at 0x205dd94fbb0>

1.2.3 Creating scatter plots

[37]:
```
fig = {'data':[{'x': cars_select.mpg, 'y':cars_select.disp, 'mode':'markers',␣
↪'name':'mpg'},
             {'x': cars_select.hp, 'y':cars_select.disp, 'mode':'markers',␣
↪'name':'hp'}],
       'layout': {'xaxis':{'title':''}, 'yaxis':{'title':'Standardized␣
↪Displacement'}}}
py.iplot(fig, filename= 'grouped-scatter-plot')
```

[37]: <IPython.lib.display.IFrame at 0x205dd26b640>

[ ]: