week5_hw

March 14, 2018

1 Name: Hassan Mphammad

2 ID: 2131117616

Load the the data from:
https://github.com/vega/vega-datasets/raw/gh-pages/data/cars.json
Then perform the proper steps to answer the questions in the cells

3 Report:

Variables	Description	Data type	Notes
Acceleration	tell the acceleration to each car with acceleration mean of 15.5	continues, float	_
Cylinders	describe the amount of cylinders in each car	categorical, float	_
Displacement	tells Displacement the car can make	continues, float	_
Horsepower	tells the power of each car engine in horses	continues, float	6 values are missing
Miles_per_Gallon	describe how many miles the car can travel for gallon	continues, float	8 values are missing
Name	tells the name of each car	continues, object	_
Origin	the origin of the car, where the car was made	categorical, Object	_
Weight_in_lbs	tells the weight of cars in lbs	continues, float	_
Year	tells which year the car was made in	continues, object	should be datetime

- The frequency of the 4 cylinders car are more in above average miles per gallon while the frequency of the 8 cylinders car are more in below average miles per gallon.
- Cars with 4 cylinders have more miles per gallon than cars with 6 or 8 cylinders, cars with 4 cylinders are more.

- USA is the most country that make 8 cylinders cars.
- Japan and Europe most of their cars are 4 and 6 cylinders.
- USA care more about the engine power more then the economic cars.
- Europe and Japan don't care about the power of the car engine as much of the car economic that doesn't burn to much fuel.

1- fuel in Japan and Europe are more expensive than USA.
2- people in USA love speed and racing.
clean_cars_df.Cylinders[(clean_cars_df.Miles_per_Gallon <= 15)].hist()
clean_cars_df.Origin[(clean_cars_df.Miles_per_Gallon <= 15)].value_counts().plot(kind="bar")
as the graphs shows:

- the cars that has low miles per gallon are 8 cylinder cars and made in USA and they are the lowest miles per gallon in the dataframe.

4 Analysis:

5 Data Preperation

This exercise will guide you through the typical steps that you are expected to perform when performing exploratory data analysis. Make sure you go through all the steps, and pay attention to the order. You will be expected to do this on your own in future exercise.

Note: You can add cells as needed

```
In [50]: # What columns are there?
```

```
In [6]: %matplotlib inline
        import numpy as np
        import pandas as pd
        cars_df = pd.read_json("https://github.com/vega/vega-datasets/raw/gh-pages/data/cars.js
```

```
In [17]: cars_df.head()
```

```
Out[17]:    Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
         0          12.0          8         307.0       130.0              18.0
         1          11.5          8         350.0       165.0              15.0
         2          11.0          8         318.0       150.0              18.0
         3          12.0          8         304.0       150.0              16.0
         4          10.5          8         302.0       140.0              17.0

                          Name Origin  Weight_in_lbs        Year
         0  chevrolet chevelle malibu    USA          3504  1970-01-01
         1          buick skylark 320    USA          3693  1970-01-01
         2         plymouth satellite    USA          3436  1970-01-01
         3              amc rebel sst    USA          3433  1970-01-01
         4                ford torino    USA          3449  1970-01-01
```

```
In [3]: # Which columns contain missing values? how many missing values are there

In [46]: cars_df[cars_df.isnull().any(axis=1)]
```

Out[46]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	Name	Origin	Weight_in_lbs	Year
10	17.5	4	133.0	115.0	NaN	citroen ds-21 pallas	Europe	3090	1970-01-01
11	11.5	8	350.0	165.0	NaN	chevrolet chevelle concours (sw)	USA	4142	1970-01-01
12	11.0	8	351.0	153.0	NaN	ford torino (sw)	USA	4034	1970-01-01
13	10.5	8	383.0	175.0	NaN	plymouth satellite (sw)	USA	4166	1970-01-01
14	11.0	8	360.0	175.0	NaN	amc rebel sst (sw)	USA	3850	1970-01-01
17	8.0	8	302.0	140.0	NaN	ford mustang boss 302	USA	3353	1970-01-01
38	19.0	4	98.0	NaN	25.0	ford pinto	USA	2046	1971-01-01
39	20.0	4	97.0	48.0	NaN	volkswagen super beetle 117	Europe	1978	1971-01-01
133	17.0	6	200.0	NaN	21.0	ford maverick	USA	2875	1974-01-01
337	17.3	4	85.0	NaN	40.9	renault lecar deluxe	Europe	1835	1980-01-01
343	14.3	4	140.0	NaN	23.6	ford mustang cobra	USA	2905	1980-01-01
361	15.8	4	100.0	NaN	34.5	renault 18i	Europe	2320	1982-01-01
367	15.4	4	121.0	110.0	NaN	saab 900s	Europe	2800	1982-01-01
382	20.5	4	151.0	NaN	23.0	amc concord dl	USA	3035	1982-01-01

```
In [7]: missing_cars_df = cars_df[cars_df.isnull().any(axis=1)]
        missing_cars_df.describe()    # 14 missing values
```

Out[7]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon
count	14.000000	14.000000	14.000000	8.000000	6.000000
mean	14.914286	5.571429	205.071429	135.125000	28.000000
std	3.936781	1.949923	116.157363	43.165586	7.886951
min	8.000000	4.000000	85.000000	48.000000	21.000000
25%	11.125000	4.000000	105.250000	113.750000	23.150000
50%	15.600000	4.000000	145.500000	146.500000	24.300000
75%	17.450000	8.000000	338.000000	167.500000	32.125000
max	20.500000	8.000000	383.000000	175.000000	40.900000

```
       Weight_in_lbs
count      14.000000
mean     3030.642857
std       804.556219
min      1835.000000
25%      2440.000000
50%      2970.000000
75%      3725.750000
max      4166.000000
```

```
In [323]: missing_cars_df.Acceleration.isnull().value_counts()

Out[323]: False    14
          Name: Acceleration, dtype: int64

In [324]: missing_cars_df.Cylinders.isnull().value_counts()

Out[324]: False    14
          Name: Cylinders, dtype: int64

In [325]: missing_cars_df.Displacement.isnull().value_counts()

Out[325]: False    14
          Name: Displacement, dtype: int64

In [326]: missing_cars_df.Horsepower.isnull().value_counts() # 6 missing values

Out[326]: False    8
          True     6
          Name: Horsepower, dtype: int64

In [327]: missing_cars_df.Miles_per_Gallon.isnull().value_counts() # 8 missing values

Out[327]: True     8
          False    6
          Name: Miles_per_Gallon, dtype: int64

In [328]: missing_cars_df.Weight_in_lbs.isnull().value_counts()

Out[328]: False    14
          Name: Weight_in_lbs, dtype: int64

In [329]: missing_cars_df.Year.isnull().value_counts()

Out[329]: False    14
          Name: Year, dtype: int64

In [1]: # Which rows are duplicated?

In [51]: cars_df.duplicated().value_counts()  # non are duplicated
```

```
Out[51]: False    406
         dtype: int64
```

```
In [ ]: # Fix the data to exlude missing values and duplicate rows (hint: use filteration)
```

```
In [8]: clean_cars_df = cars_df[~(cars_df.isnull().any(axis=1)) | (cars_df.duplicated())]
        clean_cars_df
```

Out[8]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	\
0	12.0	8	307.0	130.0	18.0	
1	11.5	8	350.0	165.0	15.0	
2	11.0	8	318.0	150.0	18.0	
3	12.0	8	304.0	150.0	16.0	
4	10.5	8	302.0	140.0	17.0	
5	10.0	8	429.0	198.0	15.0	
6	9.0	8	454.0	220.0	14.0	
7	8.5	8	440.0	215.0	14.0	
8	10.0	8	455.0	225.0	14.0	
9	8.5	8	390.0	190.0	15.0	
15	10.0	8	383.0	170.0	15.0	
16	8.0	8	340.0	160.0	14.0	
18	9.5	8	400.0	150.0	15.0	
19	10.0	8	455.0	225.0	14.0	
20	15.0	4	113.0	95.0	24.0	
21	15.5	6	198.0	95.0	22.0	
22	15.5	6	199.0	97.0	18.0	
23	16.0	6	200.0	85.0	21.0	
24	14.5	4	97.0	88.0	27.0	
25	20.5	4	97.0	46.0	26.0	
26	17.5	4	110.0	87.0	25.0	
27	14.5	4	107.0	90.0	24.0	
28	17.5	4	104.0	95.0	25.0	
29	12.5	4	121.0	113.0	26.0	
30	15.0	6	199.0	90.0	21.0	
31	14.0	8	360.0	215.0	10.0	
32	15.0	8	307.0	200.0	10.0	
33	13.5	8	318.0	210.0	11.0	
34	18.5	8	304.0	193.0	9.0	
35	14.5	4	97.0	88.0	27.0	
..	...	...	...	...	...	
375	19.6	4	112.0	88.0	28.0	
376	18.6	4	112.0	88.0	27.0	
377	18.0	4	112.0	88.0	34.0	
378	16.2	4	112.0	85.0	31.0	
379	16.0	4	135.0	84.0	29.0	
380	18.0	4	151.0	90.0	27.0	
381	16.4	4	140.0	92.0	24.0	
383	15.3	4	105.0	74.0	36.0	

384	18.2	4	91.0	68.0	37.0
385	17.6	4	91.0	68.0	31.0
386	14.7	4	105.0	63.0	38.0
387	17.3	4	98.0	70.0	36.0
388	14.5	4	120.0	88.0	36.0
389	14.5	4	107.0	75.0	36.0
390	16.9	4	108.0	70.0	34.0
391	15.0	4	91.0	67.0	38.0
392	15.7	4	91.0	67.0	32.0
393	16.2	4	91.0	67.0	38.0
394	16.4	6	181.0	110.0	25.0
395	17.0	6	262.0	85.0	38.0
396	14.5	4	156.0	92.0	26.0
397	14.7	6	232.0	112.0	22.0
398	13.9	4	144.0	96.0	32.0
399	13.0	4	135.0	84.0	36.0
400	17.3	4	151.0	90.0	27.0
401	15.6	4	140.0	86.0	27.0
402	24.6	4	97.0	52.0	44.0
403	11.6	4	135.0	84.0	32.0
404	18.6	4	120.0	79.0	28.0
405	19.4	4	119.0	82.0	31.0

	Name	Origin	Weight_in_lbs	Year
0	chevrolet chevelle malibu	USA	3504	1970-01-01
1	buick skylark 320	USA	3693	1970-01-01
2	plymouth satellite	USA	3436	1970-01-01
3	amc rebel sst	USA	3433	1970-01-01
4	ford torino	USA	3449	1970-01-01
5	ford galaxie 500	USA	4341	1970-01-01
6	chevrolet impala	USA	4354	1970-01-01
7	plymouth fury iii	USA	4312	1970-01-01
8	pontiac catalina	USA	4425	1970-01-01
9	amc ambassador dpl	USA	3850	1970-01-01
15	dodge challenger se	USA	3563	1970-01-01
16	plymouth 'cuda 340	USA	3609	1970-01-01
18	chevrolet monte carlo	USA	3761	1970-01-01
19	buick estate wagon (sw)	USA	3086	1970-01-01
20	toyota corona mark ii	Japan	2372	1970-01-01
21	plymouth duster	USA	2833	1970-01-01
22	amc hornet	USA	2774	1970-01-01
23	ford maverick	USA	2587	1970-01-01
24	datsun pl510	Japan	2130	1970-01-01
25	volkswagen 1131 deluxe sedan	Europe	1835	1970-01-01
26	peugeot 504	Europe	2672	1970-01-01
27	audi 100 ls	Europe	2430	1970-01-01
28	saab 99e	Europe	2375	1970-01-01
29	bmw 2002	Europe	2234	1970-01-01

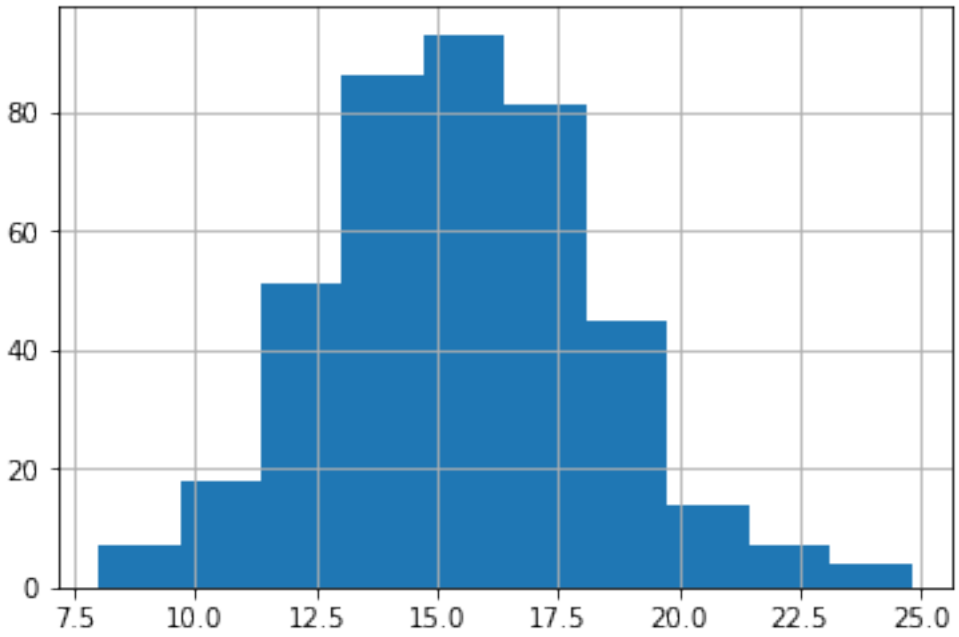
```
30                          amc gremlin     USA          2648  1970-01-01
31                            ford f250     USA          4615  1970-01-01
32                            chevy c20     USA          4376  1970-01-01
33                           dodge d200     USA          4382  1970-01-01
34                             hi 1200d     USA          4732  1970-01-01
35                         datsun pl510   Japan          2130  1971-01-01
..                                  ...     ...           ...         ...
375                  chevrolet cavalier     USA          2605  1982-01-01
376            chevrolet cavalier wagon     USA          2640  1982-01-01
377           chevrolet cavalier 2-door     USA          2395  1982-01-01
378          pontiac j2000 se hatchback     USA          2575  1982-01-01
379                       dodge aries se     USA          2525  1982-01-01
380                     pontiac phoenix     USA          2735  1982-01-01
381                ford fairmont futura     USA          2865  1982-01-01
383                volkswagen rabbit l  Europe          1980  1982-01-01
384                  mazda glc custom l   Japan          2025  1982-01-01
385                    mazda glc custom   Japan          1970  1982-01-01
386              plymouth horizon miser     USA          2125  1982-01-01
387                      mercury lynx l     USA          2125  1982-01-01
388                    nissan stanza xe   Japan          2160  1982-01-01
389               honda Accelerationord   Japan          2205  1982-01-01
390                     toyota corolla   Japan          2245  1982-01-01
391                       honda civic   Japan          1965  1982-01-01
392                  honda civic (auto)   Japan          1965  1982-01-01
393                       datsun 310 gx   Japan          1995  1982-01-01
394                buick century limited     USA          2945  1982-01-01
395  oldsmobile cutlass ciera (diesel)     USA          3015  1982-01-01
396          chrysler lebaron medallion     USA          2585  1982-01-01
397                       ford granada l     USA          2835  1982-01-01
398                     toyota celica gt   Japan          2665  1982-01-01
399                    dodge charger 2.2     USA          2370  1982-01-01
400                     chevrolet camaro     USA          2950  1982-01-01
401                      ford mustang gl     USA          2790  1982-01-01
402                            vw pickup  Europe          2130  1982-01-01
403                       dodge rampage     USA          2295  1982-01-01
404                         ford ranger     USA          2625  1982-01-01
405                          chevy s-10     USA          2720  1982-01-01

[392 rows x 9 columns]
```

6 Preliminary Data Exploration

You start by exploring single variable to gain a better understanding of it. The questions in the following cells are the typical questions that you might ask when doing exploratory data analysis.

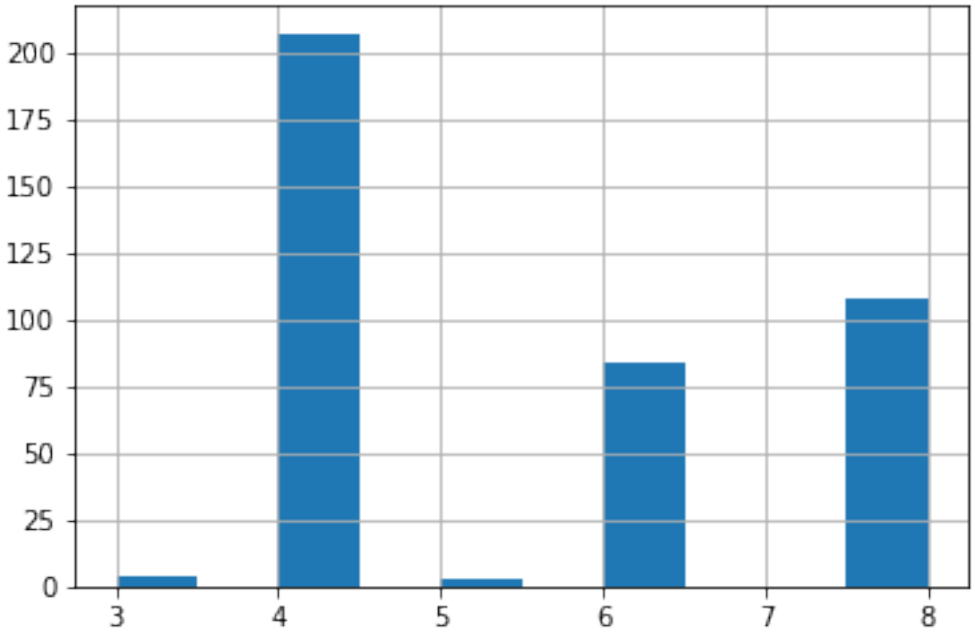
```
In [ ]: # Describe each columns, using statistical summaries and the appropriate visualization
        # What can you say about each variable? (write it in comments under each variable as y
```

```
In [66]: clean_cars_df.Acceleration.describe()
         # tell the acceleration to each car with mean accelerathion of 15.5

Out[66]: count    406.000000
         mean      15.519704
         std        2.803359
         min        8.000000
         25%       13.700000
         50%       15.500000
         75%       17.175000
         max       24.800000
         Name: Acceleration, dtype: float64

In [64]: clean_cars_df.Acceleration.hist()

Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0xa63adbf358>
```

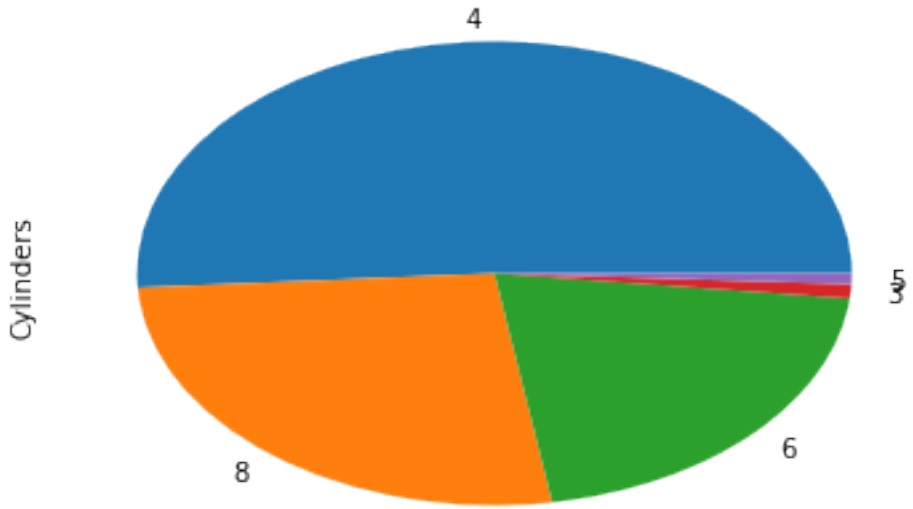



```
In [56]: clean_cars_df.Cylinders.describe()

Out[56]: count    406.000000
         mean       5.475369
         std        1.712160
         min        3.000000
         25%        4.000000
         50%        4.000000
         75%        8.000000
         max        8.000000
         Name: Cylinders, dtype: float64
```

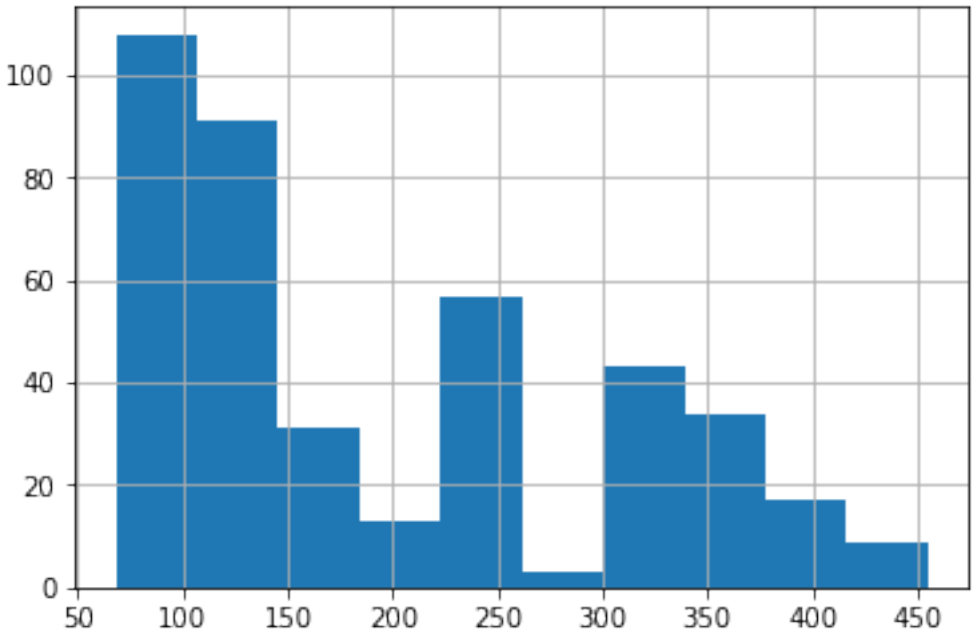
```
In [216]: clean_cars_df.Cylinders.hist()

Out[216]: <matplotlib.axes._subplots.AxesSubplot at 0xa6442e79b0>
```




```
In [242]: clean_cars_df.Cylinders.value_counts().plot(kind="pie")

Out[242]: <matplotlib.axes._subplots.AxesSubplot at 0xa6445cd518>
```



```
In [94]: clean_cars_df.Displacement.describe()

Out[94]: count    406.000000
         mean     194.779557
         std      104.922458
         min       68.000000
         25%      105.000000
         50%      151.000000
         75%      302.000000
         max      455.000000
         Name: Displacement, dtype: float64

In [95]: clean_cars_df.Displacement.hist()

Out[95]: <matplotlib.axes._subplots.AxesSubplot at 0xa645d36710>
```

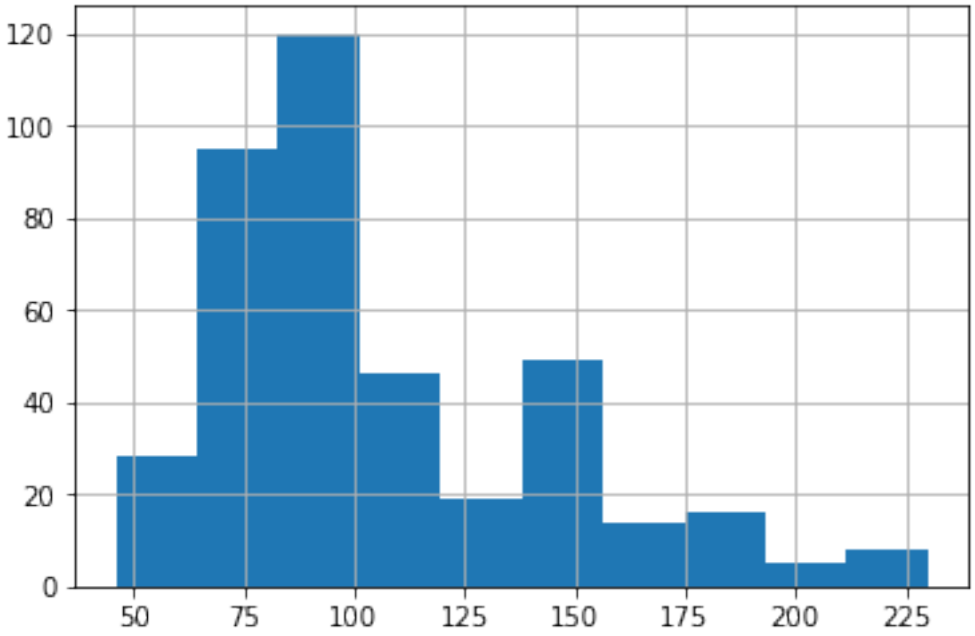



```
In [59]: clean_cars_df.Horsepower.describe()

Out[59]: count    400.000000
         mean     105.082500
         std       38.768779
         min       46.000000
         25%       75.750000
```

```
50%       95.000000
75%      130.000000
max      230.000000
Name: Horsepower, dtype: float64
```

```
In [96]: clean_cars_df.Horsepower.hist()
```

```
Out[96]: <matplotlib.axes._subplots.AxesSubplot at 0xa645e462e8>
```

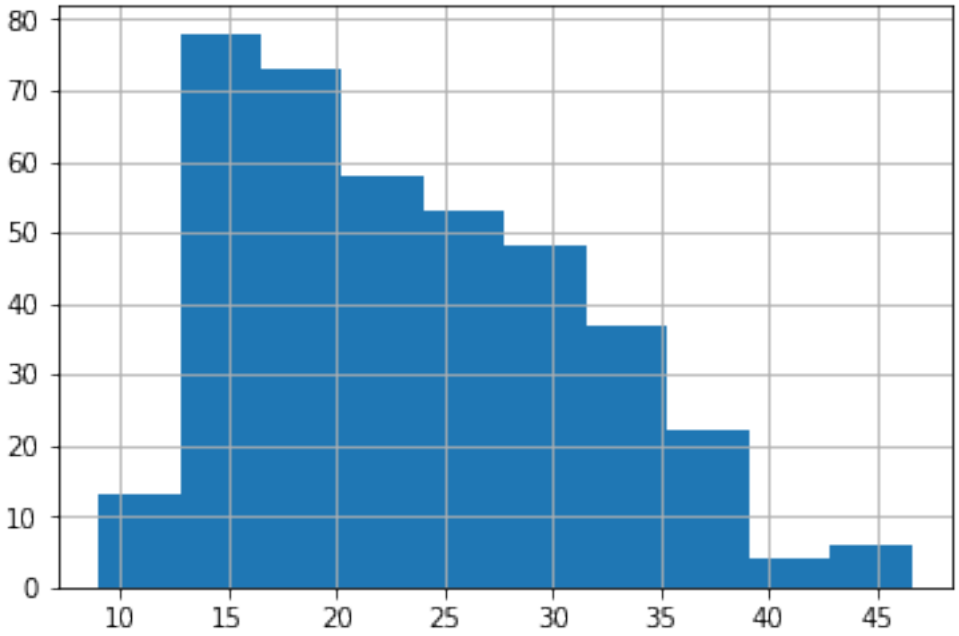



```
In [60]: clean_cars_df.Miles_per_Gallon.describe()
```

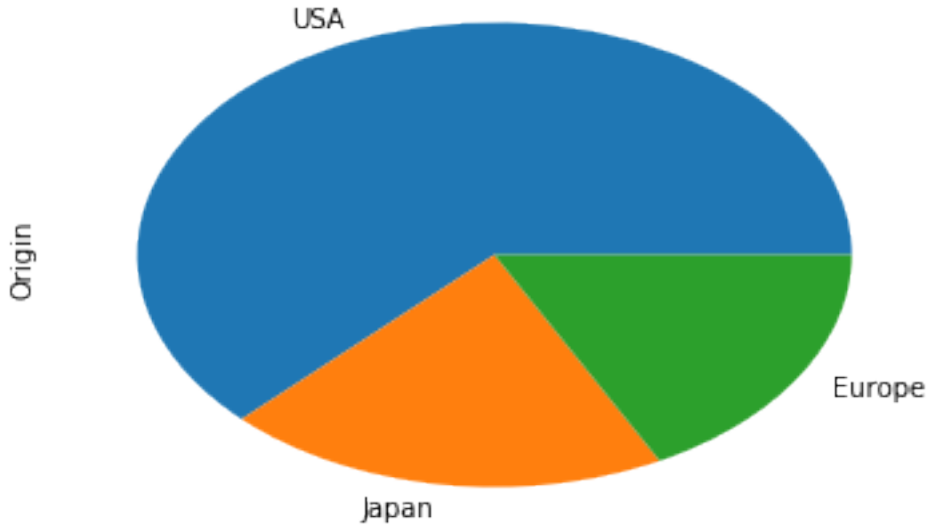
```
Out[60]: count    398.000000
         mean      23.514573
         std        7.815984
         min        9.000000
         25%       17.500000
         50%       23.000000
         75%       29.000000
         max       46.600000
         Name: Miles_per_Gallon, dtype: float64
```

```
In [271]: clean_cars_df.Miles_per_Gallon.hist()
```

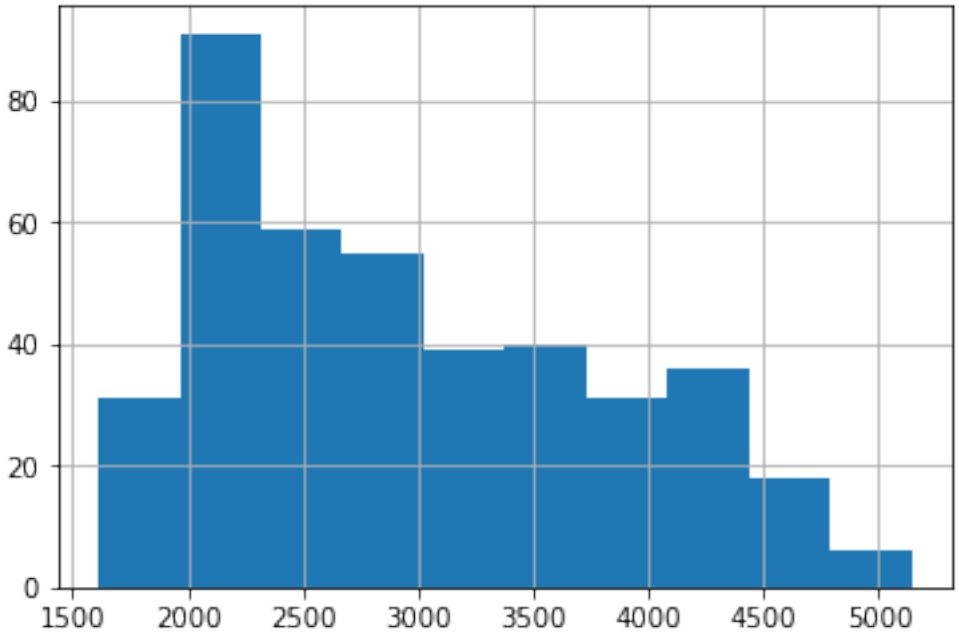
```
Out[271]: <matplotlib.axes._subplots.AxesSubplot at 0xa6494e8828>
```



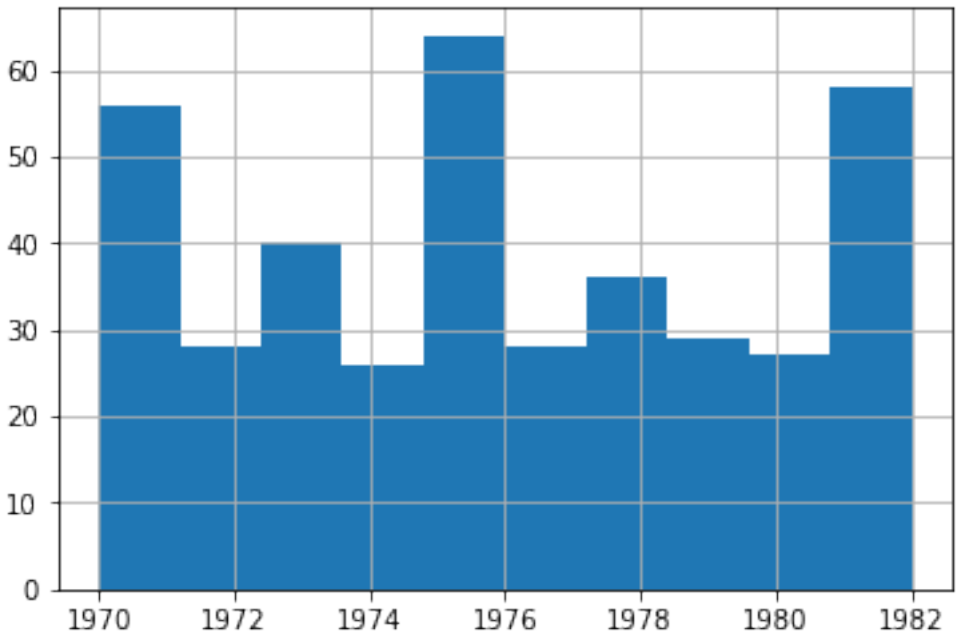

```
In [61]: clean_cars_df.Name.describe()

Out[61]: count            406
         unique           311
         top       ford pinto
         freq                6
         Name: Name, dtype: object

In [62]: clean_cars_df.Origin.describe()

Out[62]: count     406
         unique      3
         top       USA
         freq      254
         Name: Origin, dtype: object

In [285]: clean_cars_df.Origin.value_counts().plot(kind="pie")

Out[285]: <matplotlib.axes._subplots.AxesSubplot at 0xa6496f46a0>
```




```
In [63]: clean_cars_df.Weight_in_lbs.describe()

Out[63]: count     406.000000
         mean     2979.413793
         std       847.004328
         min      1613.000000
         25%      2226.500000
         50%      2822.500000
         75%      3618.250000
         max      5140.000000
         Name: Weight_in_lbs, dtype: float64

In [109]: clean_cars_df.Weight_in_lbs.hist()

Out[109]: <matplotlib.axes._subplots.AxesSubplot at 0xa647023be0>
```




```
In [64]: clean_cars_df.Year.describe()

Out[64]: count            406
         unique            12
         top       1982-01-01
         freq              61
         Name: Year, dtype: object

In [268]: cars_df_date = pd.to_datetime(clean_cars_df.Year)
          cars_df_date.hist()

Out[268]: <matplotlib.axes._subplots.AxesSubplot at 0xa6491ddef0>
```




```
In [6]: # What are the top 5 cars that have the highest weight?

In [132]: clean_cars_df.Weight_in_lbs.sort_values(ascending=False).head()

Out[132]: 51     5140
          110    4997
          49     4955
          97     4952
          102    4951
          Name: Weight_in_lbs, dtype: int64

In [7]: # Find the top 5 cars with the most cylinders, but the least horse power

In [75]: clean_cars_df[["Cylinders", "Horsepower"]].sort_values(by=["Cylinders", "Horsepower"]

Out[75]:      Cylinders  Horsepower
         307          8        90.0
         372          8       105.0
         172          8       110.0
         229          8       110.0
         256          8       110.0

In [8]: # List all the cars that have ABOVE average miles_per_galons which are made in the USA
        # then calculate and plot the frequencies for the Cylinder column

In [9]: above_average = clean_cars_df[(clean_cars_df.Miles_per_Gallon > clean_cars_df.Miles_per
        above_average
```

Out[9]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	\
36	15.5	4	140.0	90.0	28.0	
63	20.5	4	91.0	70.0	26.0	
65	17.0	4	97.5	80.0	25.0	
90	15.0	4	98.0	80.0	28.0	
137	16.5	4	122.0	80.0	26.0	
139	17.0	4	140.0	75.0	25.0	
153	14.5	4	90.0	75.0	28.0	
191	14.9	4	140.0	92.0	25.0	
192	17.7	4	98.0	79.0	26.0	
200	17.6	6	200.0	81.0	24.0	
202	22.2	4	85.0	52.0	29.0	
203	22.1	4	98.0	60.0	24.5	
213	13.6	4	140.0	72.0	26.5	
224	14.8	4	111.0	80.0	30.0	
226	15.5	4	122.0	96.0	25.5	
241	16.0	4	151.0	88.0	24.5	
243	15.8	4	140.0	89.0	25.5	
244	17.0	4	98.0	63.0	30.5	
245	15.9	4	98.0	83.0	33.5	
252	14.4	4	98.0	66.0	36.1	
262	15.4	4	140.0	88.0	25.1	
273	16.5	4	98.0	68.0	30.0	
276	14.5	4	105.0	75.0	30.9	
279	17.6	4	151.0	85.0	23.8	
302	14.4	4	98.0	80.0	35.7	
303	15.0	4	121.0	80.0	27.4	
307	22.2	8	260.0	90.0	23.9	
308	13.2	4	105.0	70.0	34.2	
309	14.9	4	105.0	70.0	34.5	
312	16.0	4	151.0	90.0	28.4	
..	...	...	...	...	...	
322	20.1	4	151.0	90.0	24.3	
330	14.4	4	156.0	105.0	27.9	
345	15.7	4	135.0	84.0	27.2	
346	16.4	4	151.0	84.0	26.6	
347	14.4	4	156.0	92.0	25.8	
348	12.6	6	173.0	110.0	23.5	
349	12.9	4	135.0	84.0	30.0	
351	16.4	4	86.0	64.0	39.0	
357	14.9	4	105.0	63.0	34.7	
358	16.2	4	98.0	65.0	34.4	
359	20.7	4	98.0	65.0	29.9	
372	19.0	8	350.0	105.0	26.6	
375	19.6	4	112.0	88.0	28.0	
376	18.6	4	112.0	88.0	27.0	
377	18.0	4	112.0	88.0	34.0	
378	16.2	4	112.0	85.0	31.0	

379	16.0	4	135.0	84.0	29.0
380	18.0	4	151.0	90.0	27.0
381	16.4	4	140.0	92.0	24.0
386	14.7	4	105.0	63.0	38.0
387	17.3	4	98.0	70.0	36.0
394	16.4	6	181.0	110.0	25.0
395	17.0	6	262.0	85.0	38.0
396	14.5	4	156.0	92.0	26.0
399	13.0	4	135.0	84.0	36.0
400	17.3	4	151.0	90.0	27.0
401	15.6	4	140.0	86.0	27.0
403	11.6	4	135.0	84.0	32.0
404	18.6	4	120.0	79.0	28.0
405	19.4	4	119.0	82.0	31.0

	Name	Origin	Weight_in_lbs	Year
36	chevrolet vega 2300	USA	2264	1971-01-01
63	plymouth cricket	USA	1955	1971-01-01
65	dodge colt hardtop	USA	2126	1972-01-01
90	dodge colt (sw)	USA	2164	1972-01-01
137	ford pinto	USA	2451	1974-01-01
139	chevrolet vega	USA	2542	1974-01-01
153	dodge colt	USA	2125	1974-01-01
191	capri ii	USA	2572	1976-01-01
192	dodge colt	USA	2255	1976-01-01
200	ford maverick	USA	3012	1976-01-01
202	chevrolet chevette	USA	2035	1976-01-01
203	chevrolet woody	USA	2164	1976-01-01
213	ford pinto	USA	2565	1976-01-01
224	buick opel isuzu deluxe	USA	2155	1977-01-01
226	plymouth arrow gs	USA	2300	1977-01-01
241	pontiac sunbird coupe	USA	2740	1977-01-01
243	ford mustang ii 2+2	USA	2755	1977-01-01
244	chevrolet chevette	USA	2051	1977-01-01
245	dodge colt m/m	USA	2075	1977-01-01
252	ford fiesta	USA	1800	1978-01-01
262	ford fairmont (man)	USA	2720	1978-01-01
273	chevrolet chevette	USA	2155	1978-01-01
276	dodge omni	USA	2230	1978-01-01
279	oldsmobile starfire sx	USA	2855	1978-01-01
302	dodge colt hatchback custom	USA	1915	1979-01-01
303	amc spirit dl	USA	2670	1979-01-01
307	oldsmobile cutlass salon brougham	USA	3420	1979-01-01
308	plymouth horizon	USA	2200	1979-01-01
309	plymouth horizon tc3	USA	2150	1979-01-01
312	buick skylark limited	USA	2670	1979-01-01
..	...	...	...	...
322	amc concord	USA	3003	1980-01-01

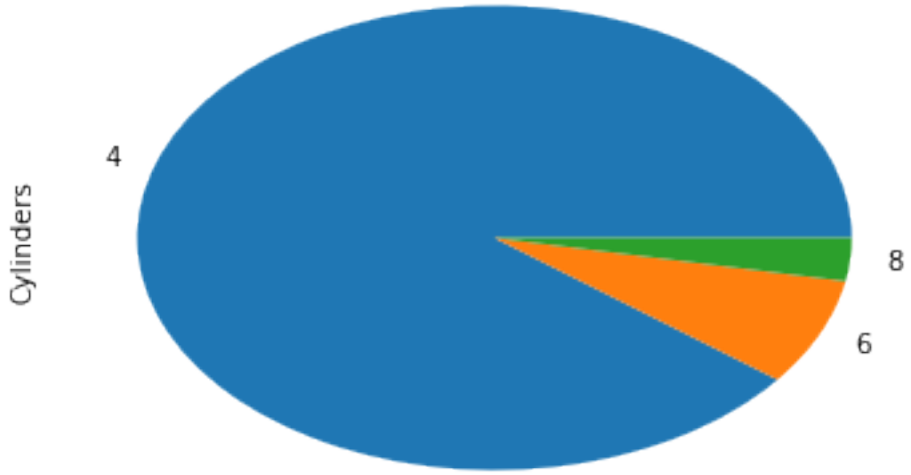
```
330                         dodge colt    USA          2800  1980-01-01
345                    plymouth reliant    USA          2490  1982-01-01
346                       buick skylark    USA          2635  1982-01-01
347              dodge aries wagon (sw)    USA          2620  1982-01-01
348                  chevrolet citation    USA          2725  1982-01-01
349                    plymouth reliant    USA          2385  1982-01-01
351                      plymouth champ    USA          1875  1982-01-01
357                  plymouth horizon 4    USA          2215  1982-01-01
358                      ford escort 4w    USA          2045  1982-01-01
359                      ford escort 2h    USA          2380  1982-01-01
372               oldsmobile cutlass ls    USA          3725  1982-01-01
375                  chevrolet cavalier    USA          2605  1982-01-01
376            chevrolet cavalier wagon    USA          2640  1982-01-01
377           chevrolet cavalier 2-door    USA          2395  1982-01-01
378          pontiac j2000 se hatchback    USA          2575  1982-01-01
379                      dodge aries se    USA          2525  1982-01-01
380                     pontiac phoenix    USA          2735  1982-01-01
381                ford fairmont futura    USA          2865  1982-01-01
386              plymouth horizon miser    USA          2125  1982-01-01
387                      mercury lynx l    USA          2125  1982-01-01
394               buick century limited    USA          2945  1982-01-01
395  oldsmobile cutlass ciera (diesel)    USA          3015  1982-01-01
396          chrysler lebaron medallion    USA          2585  1982-01-01
399                    dodge charger 2.2    USA          2370  1982-01-01
400                     chevrolet camaro    USA          2950  1982-01-01
401                     ford mustang gl    USA          2790  1982-01-01
403                       dodge rampage    USA          2295  1982-01-01
404                         ford ranger    USA          2625  1982-01-01
405                          chevy s-10    USA          2720  1982-01-01

[66 rows x 9 columns]
```

```
In [235]: above_average.Cylinders.value_counts()
```

```
Out[235]: 4    60
          6     5
          8     2
          Name: Cylinders, dtype: int64
```

```
In [236]: above_average.Cylinders.value_counts().plot(kind="pie")
```

```
Out[236]: <matplotlib.axes._subplots.AxesSubplot at 0xa64453c4e0>
```



In [8]:
```
# List all the cars that have BELOW average miles_per_galons which are made in the USA
# then calculate plot the frequencies for the Cylinder column
```

In [239]:
```
below_average = clean_cars_df[(clean_cars_df.Miles_per_Gallon < clean_cars_df.Miles_
                & (clean_cars_df.Origin =="USA")]
below_average
```

Out[239]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	\
0	12.0	8	307.0	130.0	18.0	
1	11.5	8	350.0	165.0	15.0	
2	11.0	8	318.0	150.0	18.0	
3	12.0	8	304.0	150.0	16.0	
4	10.5	8	302.0	140.0	17.0	
5	10.0	8	429.0	198.0	15.0	
6	9.0	8	454.0	220.0	14.0	
7	8.5	8	440.0	215.0	14.0	
8	10.0	8	455.0	225.0	14.0	
9	8.5	8	390.0	190.0	15.0	
15	10.0	8	383.0	170.0	15.0	
16	8.0	8	340.0	160.0	14.0	
18	9.5	8	400.0	150.0	15.0	
19	10.0	8	455.0	225.0	14.0	
21	15.5	6	198.0	95.0	22.0	
22	15.5	6	199.0	97.0	18.0	
23	16.0	6	200.0	85.0	21.0	
30	15.0	6	199.0	90.0	21.0	
31	14.0	8	360.0	215.0	10.0	

32	15.0	8	307.0	200.0	10.0
33	13.5	8	318.0	210.0	11.0
34	18.5	8	304.0	193.0	9.0
40	13.0	6	232.0	100.0	19.0
41	15.5	6	225.0	105.0	16.0
42	15.5	6	250.0	100.0	17.0
43	15.5	6	250.0	88.0	19.0
44	15.5	6	232.0	100.0	18.0
45	12.0	8	350.0	165.0	14.0
46	11.5	8	400.0	175.0	14.0
47	13.5	8	351.0	153.0	14.0
..	...	...	...	...	...
265	15.8	6	231.0	105.0	20.6
266	16.7	6	200.0	85.0	20.8
267	18.7	6	225.0	110.0	18.6
268	15.1	6	258.0	120.0	18.1
269	13.2	8	305.0	145.0	19.2
270	13.4	6	231.0	165.0	17.7
271	11.2	8	302.0	139.0	18.1
272	13.7	8	318.0	140.0	17.5
278	16.7	4	156.0	105.0	23.2
287	15.4	6	231.0	115.0	21.5
288	18.2	6	200.0	85.0	19.8
289	17.3	4	140.0	88.0	22.3
290	18.2	6	232.0	90.0	20.2
291	16.6	6	225.0	110.0	20.6
292	15.4	8	305.0	130.0	17.0
293	13.4	8	302.0	129.0	17.6
294	13.2	8	351.0	138.0	16.5
295	15.2	8	318.0	135.0	18.2
296	14.9	8	350.0	155.0	16.9
297	14.3	8	351.0	142.0	15.5
298	15.0	8	267.0	125.0	19.2
299	13.0	8	360.0	150.0	18.5
305	17.4	8	350.0	125.0	23.0
323	18.7	6	225.0	90.0	19.1
348	12.6	6	173.0	110.0	23.5
371	15.8	6	231.0	110.0	22.4
373	17.1	6	200.0	88.0	20.2
374	16.6	6	225.0	85.0	17.6
382	20.5	4	151.0	NaN	23.0
397	14.7	6	232.0	112.0	22.0

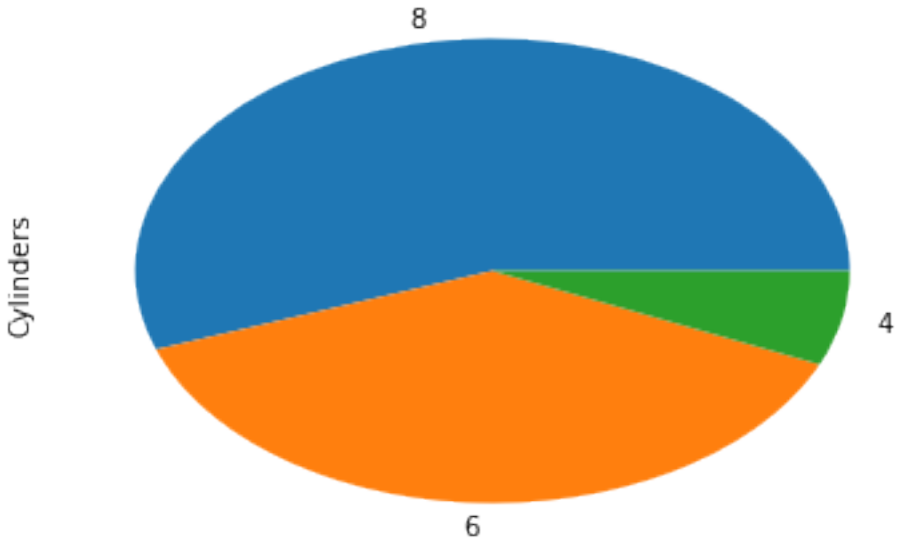
	Name	Origin	Weight_in_lbs	Year
0	chevrolet chevelle malibu	USA	3504	1970-01-01
1	buick skylark 320	USA	3693	1970-01-01
2	plymouth satellite	USA	3436	1970-01-01
3	amc rebel sst	USA	3433	1970-01-01

4	ford torino	USA	3449	1970-01-01
5	ford galaxie 500	USA	4341	1970-01-01
6	chevrolet impala	USA	4354	1970-01-01
7	plymouth fury iii	USA	4312	1970-01-01
8	pontiac catalina	USA	4425	1970-01-01
9	amc ambassador dpl	USA	3850	1970-01-01
15	dodge challenger se	USA	3563	1970-01-01
16	plymouth 'cuda 340	USA	3609	1970-01-01
18	chevrolet monte carlo	USA	3761	1970-01-01
19	buick estate wagon (sw)	USA	3086	1970-01-01
21	plymouth duster	USA	2833	1970-01-01
22	amc hornet	USA	2774	1970-01-01
23	ford maverick	USA	2587	1970-01-01
30	amc gremlin	USA	2648	1970-01-01
31	ford f250	USA	4615	1970-01-01
32	chevy c20	USA	4376	1970-01-01
33	dodge d200	USA	4382	1970-01-01
34	hi 1200d	USA	4732	1970-01-01
40	amc gremlin	USA	2634	1971-01-01
41	plymouth satellite custom	USA	3439	1971-01-01
42	chevrolet chevelle malibu	USA	3329	1971-01-01
43	ford torino 500	USA	3302	1971-01-01
44	amc matador	USA	3288	1971-01-01
45	chevrolet impala	USA	4209	1971-01-01
46	pontiac catalina brougham	USA	4464	1971-01-01
47	ford galaxie 500	USA	4154	1971-01-01
..	...	...	...	...
265	buick century special	USA	3380	1978-01-01
266	mercury zephyr	USA	3070	1978-01-01
267	dodge aspen	USA	3620	1978-01-01
268	amc concord d/l	USA	3410	1978-01-01
269	chevrolet monte carlo landau	USA	3425	1978-01-01
270	buick regal sport coupe (turbo)	USA	3445	1978-01-01
271	ford futura	USA	3205	1978-01-01
272	dodge magnum xe	USA	4080	1978-01-01
278	plymouth sapporo	USA	2745	1978-01-01
287	pontiac lemans v6	USA	3245	1979-01-01
288	mercury zephyr 6	USA	2990	1979-01-01
289	ford fairmont 4	USA	2890	1979-01-01
290	amc concord dl 6	USA	3265	1979-01-01
291	dodge aspen 6	USA	3360	1979-01-01
292	chevrolet caprice classic	USA	3840	1979-01-01
293	ford ltd landau	USA	3725	1979-01-01
294	mercury grand marquis	USA	3955	1979-01-01
295	dodge st. regis	USA	3830	1979-01-01
296	buick estate wagon (sw)	USA	4360	1979-01-01
297	ford country squire (sw)	USA	4054	1979-01-01
298	chevrolet malibu classic (sw)	USA	3605	1979-01-01

```
299  chrysler lebaron town @ country (sw)    USA          3940  1979-01-01
305                      cadillac eldorado    USA          3900  1979-01-01
323                            dodge aspen    USA          3381  1980-01-01
348                    chevrolet citation     USA          2725  1982-01-01
371                         buick century     USA          3415  1982-01-01
373                       ford granada gl     USA          3060  1982-01-01
374                  chrysler lebaron salon    USA          3465  1982-01-01
382                           amc concord dl    USA          3035  1982-01-01
397                           ford granada l    USA          2835  1982-01-01

[182 rows x 9 columns]
```

In [241]:
```
below_average.Cylinders.value_counts().plot(kind="pie")
```

Out[241]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa64455f7b8>
```



In [10]:
```
# Compare the frequencies and plots for the number of cylinders between
# above and below average miles per gallon cars, what do you notice?
# are there any interesting observations or findings? let us know what they are
```

In [288]:
```
above_average.Cylinders.value_counts()
```

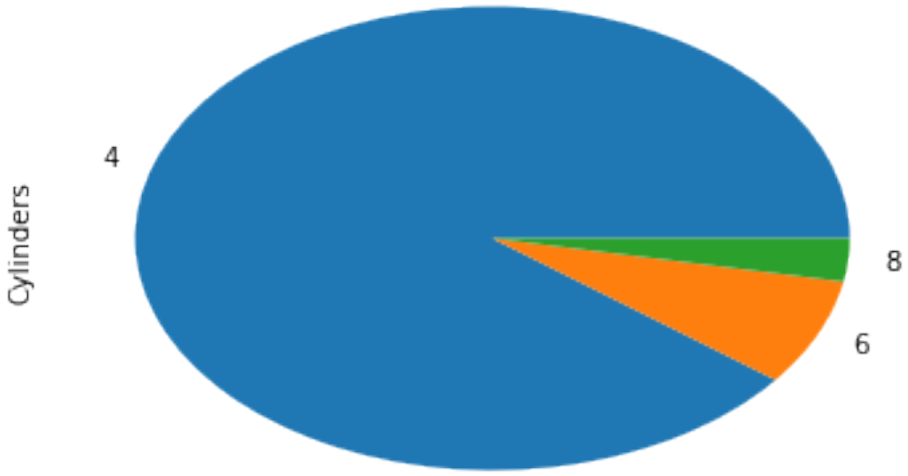
Out[288]:
```
4    60
6     5
8     2
Name: Cylinders, dtype: int64
```

In [289]:
```
below_average.Cylinders.value_counts()
```

```
Out[289]: 8    101
          6     69
          4     12
          Name: Cylinders, dtype: int64
```

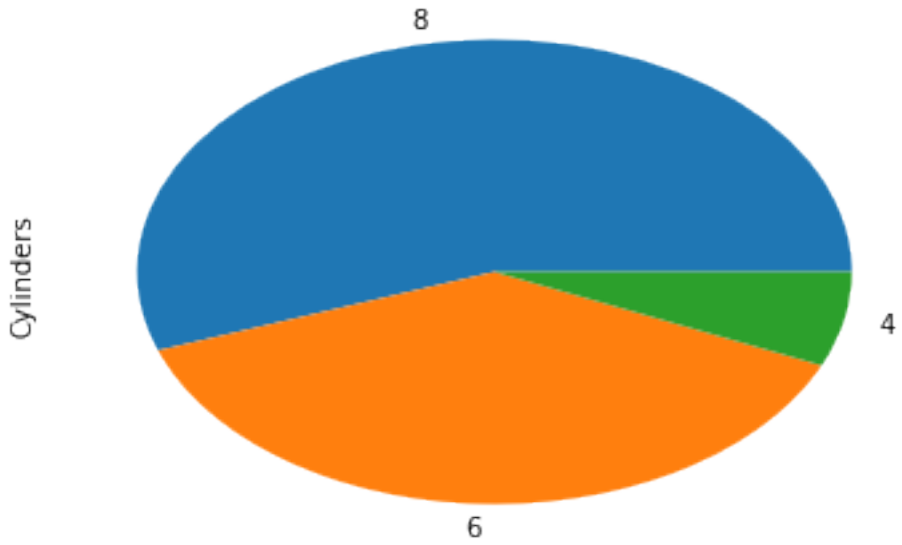
```
In [286]: above_average.Cylinders.value_counts().plot(kind="pie")
```

```
Out[286]: <matplotlib.axes._subplots.AxesSubplot at 0xa649789b00>
```




```
In [287]: below_average.Cylinders.value_counts().plot(kind="pie")
```

```
Out[287]: <matplotlib.axes._subplots.AxesSubplot at 0xa64980f400>
```




```
In [ ]: # the frequency of the 4 cylinders car are more in above average miles per gallon whil

        # cars with 4 cylinders have more miles per gallon than cars with 6 or 8 cylinders, ca
```

```
In [387]: clean_cars_df[(clean_cars_df.Origin == "USA") & (clean_cars_df.Cylinders == 8)].desc
```

Out[387]:

	Acceleration	Cylinders	Displacement	Horsepower	Miles_per_Gallon	\
count	103.000000	103.0	103.000000	103.000000	103.000000	
mean	12.955340	8.0	345.009709	158.300971	14.963107	
std	2.224759	0.0	46.776376	28.453552	2.836284	
min	8.000000	8.0	260.000000	90.000000	9.000000	
25%	11.500000	8.0	305.000000	140.000000	13.000000	
50%	13.000000	8.0	350.000000	150.000000	14.000000	
75%	14.000000	8.0	360.000000	175.000000	16.000000	
max	22.200000	8.0	455.000000	230.000000	26.600000	

	Weight_in_lbs
count	103.000000
mean	4114.718447
std	448.833159
min	3086.000000
25%	3799.000000
50%	4140.000000
75%	4403.500000
max	5140.000000

```
In [389]: clean_cars_df[(clean_cars_df.Origin == "USA") & (clean_cars_df.Cylinders == 6)].desc
```

```
Out[389]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
          count     73.000000       73.0     73.000000   73.000000         73.000000
          mean      16.467123        6.0    226.643836   99.671233         19.645205
          std        1.908974        0.0     24.677571   12.934724          3.394646
          min       11.300000        6.0    155.000000   72.000000         15.000000
          25%       15.500000        6.0    225.000000   90.000000         18.000000
          50%       16.400000        6.0    231.000000  100.000000         19.000000
          75%       17.700000        6.0    250.000000  107.000000         20.800000
          max       21.000000        6.0    262.000000  165.000000         38.000000

                 Weight_in_lbs
          count      73.000000
          mean     3218.547945
          std       332.879735
          min      2472.000000
          25%      2984.000000
          50%      3245.000000
          75%      3439.000000
          max      3907.000000

In [390]: clean_cars_df[(clean_cars_df.Origin == "USA") & (clean_cars_df.Cylinders == 4)].desc

Out[390]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
          count     69.000000       69.0     69.000000   69.000000         69.000000
          mean      16.465217        4.0    124.050725   80.956522         28.013043
          std        2.180989        0.0     21.608166   10.540447          4.566596
          min       11.600000        4.0     85.000000   52.000000         19.000000
          25%       14.900000        4.0    105.000000   72.000000         25.000000
          50%       16.200000        4.0    122.000000   84.000000         27.200000
          75%       17.700000        4.0    140.000000   88.000000         30.900000
          max       22.200000        4.0    156.000000  105.000000         39.000000

                 Weight_in_lbs
          count      69.000000
          mean     2427.391304
          std       289.974351
          min      1800.000000
          25%      2164.000000
          50%      2408.000000
          75%      2640.000000
          max      3003.000000

In [381]: clean_cars_df[(clean_cars_df.Origin == "Japan") & (clean_cars_df.Cylinders == 8)].des

Out[381]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
          count           0.0        0.0           0.0         0.0               0.0
          mean            NaN        NaN           NaN         NaN               NaN
          std             NaN        NaN           NaN         NaN               NaN
          min             NaN        NaN           NaN         NaN               NaN
```

```
25%             NaN        NaN           NaN         NaN               NaN
50%             NaN        NaN           NaN         NaN               NaN
75%             NaN        NaN           NaN         NaN               NaN
max             NaN        NaN           NaN         NaN               NaN

       Weight_in_lbs
count            0.0
mean             NaN
std              NaN
min              NaN
25%              NaN
50%              NaN
75%              NaN
max              NaN
```

In [391]:
```
clean_cars_df[(clean_cars_df.Origin == "Japan") & (clean_cars_df.Cylinders == 6)].des
```

Out[391]:
```
       Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
count      6.000000        6.0      6.000000    6.000000          6.000000
mean      13.550000        6.0    156.666667  115.833333         23.883333
std        1.434922        0.0      9.852242   12.106472          4.951936
min       11.400000        6.0    146.000000   97.000000         19.000000
25%       12.825000        6.0    148.500000  110.000000         20.500000
50%       13.650000        6.0    156.000000  118.000000         23.100000
75%       14.325000        6.0    165.000000  121.500000         25.100000
max       15.500000        6.0    168.000000  132.000000         32.700000

       Weight_in_lbs
count        6.00000
mean      2882.00000
std         56.26722
min       2807.00000
25%       2836.25000
50%       2905.00000
75%       2925.00000
max       2930.00000
```

In [392]:
```
clean_cars_df[(clean_cars_df.Origin == "Japan") & (clean_cars_df.Cylinders == 4)].des
```

Out[392]:
```
       Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
count     69.000000       69.0     69.000000   69.000000         69.000000
mean      16.569565        4.0     99.768116   75.579710         31.595652
std        1.718182        0.0     16.929912   13.982558          5.435787
min       13.500000        4.0     71.000000   52.000000         20.000000
25%       15.000000        4.0     86.000000   65.000000         27.500000
50%       16.500000        4.0     97.000000   70.000000         32.000000
75%       17.900000        4.0    113.000000   90.000000         35.000000
max       21.000000        4.0    144.000000  100.000000         46.600000
```

```
       Weight_in_lbs
count      69.000000
mean     2153.492754
std       264.306786
min      1613.000000
25%      1975.000000
50%      2130.000000
75%      2300.000000
max      2711.000000
```

```
In [384]: clean_cars_df[(clean_cars_df.Origin == "Europe") & (clean_cars_df.Cylinders == 8)].de
```

```
Out[384]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
count             0.0        0.0           0.0         0.0               0.0
mean              NaN        NaN           NaN         NaN               NaN
std               NaN        NaN           NaN         NaN               NaN
min               NaN        NaN           NaN         NaN               NaN
25%               NaN        NaN           NaN         NaN               NaN
50%               NaN        NaN           NaN         NaN               NaN
75%               NaN        NaN           NaN         NaN               NaN
max               NaN        NaN           NaN         NaN               NaN

       Weight_in_lbs
count            0.0
mean             NaN
std              NaN
min              NaN
25%              NaN
50%              NaN
75%              NaN
max              NaN
```

```
In [386]: clean_cars_df[(clean_cars_df.Origin == "Europe") & (clean_cars_df.Cylinders == 6)].de
```

```
Out[386]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
count      4.000000        4.0      4.000000    4.000000          4.000000
mean      16.425000        6.0    159.750000  113.500000         20.100000
std        2.485122        0.0     10.111874   25.566906          7.074367
min       13.600000        6.0    145.000000   76.000000         16.200000
25%       15.250000        6.0    158.500000  109.000000         16.425000
50%       16.250000        6.0    163.000000  122.500000         16.750000
75%       17.425000        6.0    164.250000  127.000000         20.425000
max       19.600000        6.0    168.000000  133.000000         30.700000

       Weight_in_lbs
count       4.000000
mean     3382.500000
std       316.478014
min      3140.000000
```

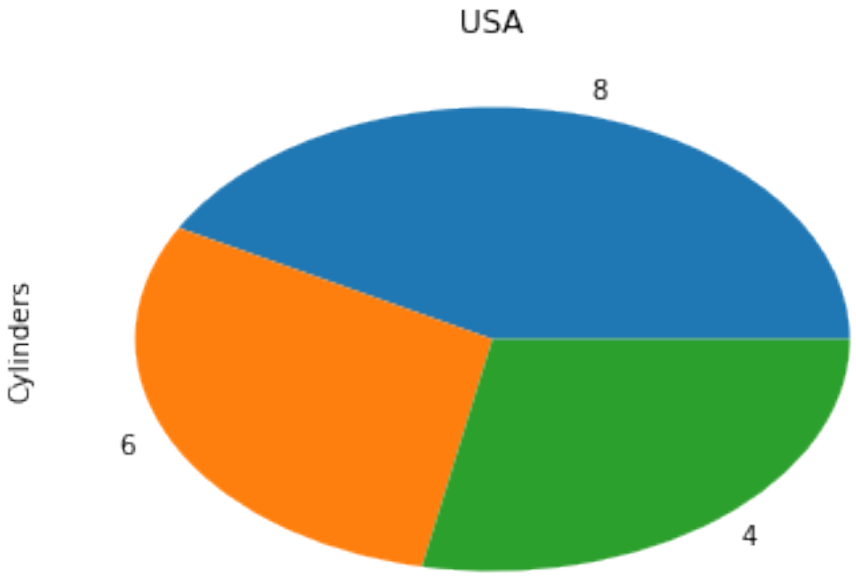
```
25%      3155.000000
50%      3285.000000
75%      3512.500000
max      3820.000000
```

```
In [394]: clean_cars_df[(clean_cars_df.Origin == "Europe") & (clean_cars_df.Cylinders == 4)].de
```

```
Out[394]:        Acceleration  Cylinders  Displacement  Horsepower  Miles_per_Gallon  \
count     61.000000       61.0     61.000000   61.000000         61.000000
mean      16.727869        4.0    104.606557   78.311475         28.106557
std        3.157274        0.0     16.653607   18.218618          6.291075
min       12.200000        4.0     68.000000   46.000000         18.000000
25%       14.500000        4.0     90.000000   69.000000         24.000000
50%       15.500000        4.0     98.000000   76.000000         27.000000
75%       18.600000        4.0    120.000000   88.000000         30.000000
max       24.800000        4.0    146.000000  115.000000         44.300000

       Weight_in_lbs
count      61.000000
mean     2338.295082
std       410.770428
min      1825.000000
25%      2000.000000
50%      2219.000000
75%      2600.000000
max      3270.000000
```

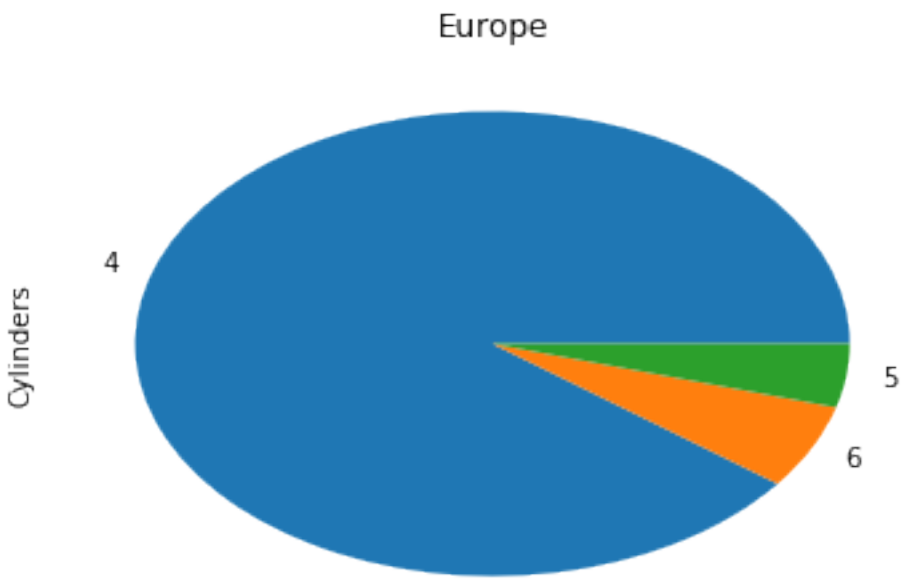
```
In [401]: clean_cars_df.Cylinders[(clean_cars_df.Origin == "USA")].value_counts().plot(kind="pi
```

```
Out[401]: <matplotlib.axes._subplots.AxesSubplot at 0xa64af5c0f0>
```



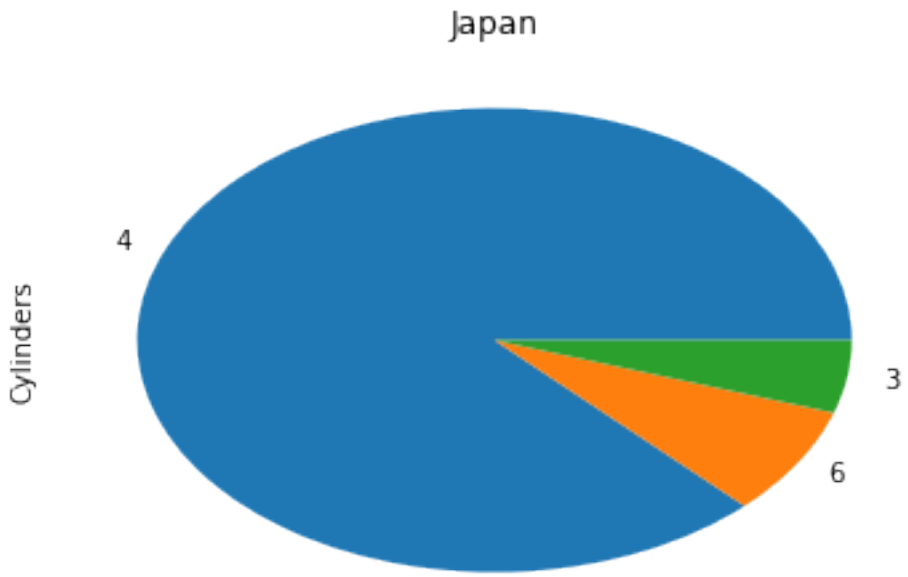
```
In [399]: clean_cars_df.Cylinders[(clean_cars_df.Origin == "Europe")].value_counts().plot(kind=
```

```
Out[399]: <matplotlib.axes._subplots.AxesSubplot at 0xa64a9b77b8>
```



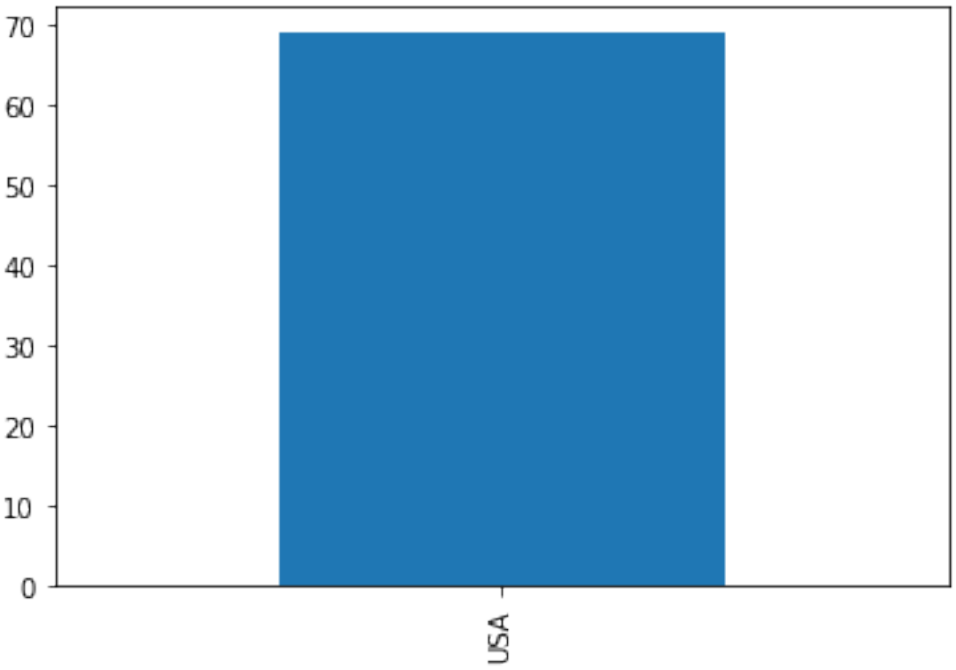

```
In [400]: clean_cars_df.Cylinders[(clean_cars_df.Origin == "Japan")].value_counts().plot(kind='
```

```
Out[400]: <matplotlib.axes._subplots.AxesSubplot at 0xa64a9c5f60>
```

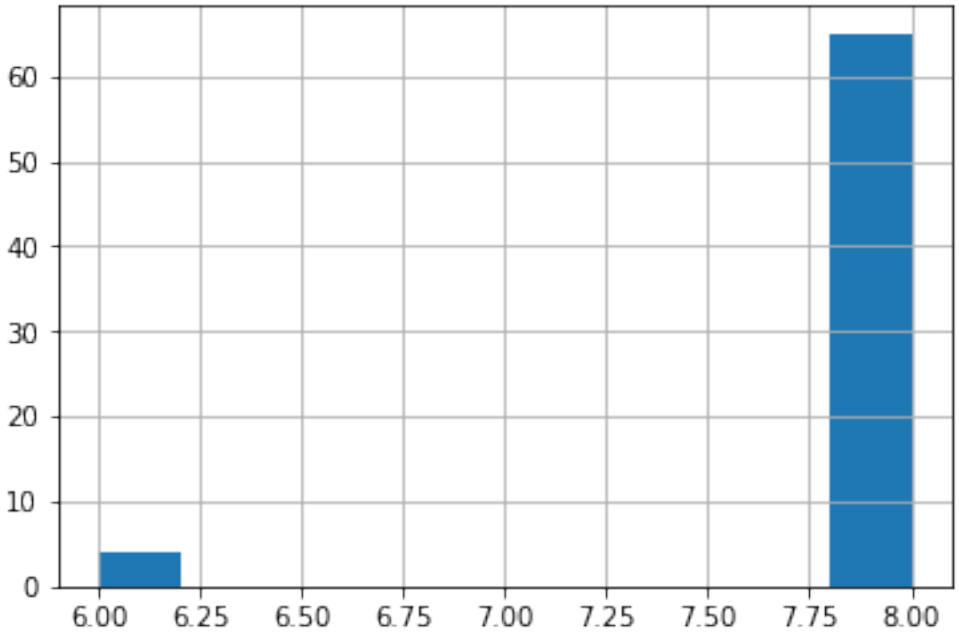



```
In [415]: clean_cars_df.Origin[(clean_cars_df.Miles_per_Gallon <= 15)].value_counts().plot(kind
```

```
Out[415]: <matplotlib.axes._subplots.AxesSubplot at 0xa64d9397b8>
```



```
In [416]: clean_cars_df.Cylinders[(clean_cars_df.Miles_per_Gallon <= 15)].hist()

Out[416]: <matplotlib.axes._subplots.AxesSubplot at 0xa64da93fd0>
```



7 The Report

Your report should be written as markdown at the top of the notebook. To know more about markdown, read the following reference.

Perform the following tasks to prepare your report:

1. Split the notebook into 2 sections, Report and Analysis
    - Report will be at the top, and will contain the findings that you discover from your analysis and be completly written in markdown. You get the facts and figures from the analysis section. While the report is at the top, it is the last part your write.
    - The analysis will be everything you did in the above section. You copy the findings in your analysis and include interesting facts into the report.
2. Include the following sections in your report:
    - Variables: Include a table listing the names of the variables, a description of what the variable is, the data type, and a notes section about the missing values and problems in the distributions.
    - Summary of Findings: Based on the analysis perform, pick the interesting findings and list them as bullet points.
    - For each finding, list the plots/evidence from your analysis that supports the finding

8 Creating a pdf report and presentation

Here you will learn how to create a pdf report and presentation slides from your notebook.

8.1 Creating HTML slides

Type the following command in CMD or Terminal:

```
jupyter nbconvert week5_hw.ipynb --to slides --reveal-prefix '//cdn.jsdelivr.net/npm/reveal.js
```

In the same directory as your notebook file, look for **week5_hw.slides.html** and open it to view the presentation slides.

Note: Replace week5_hw.ipynb with the name of any other notebook file you want to convert into slides

8.2 Create PDF document

1. Download and install pandoc
2. run the following command in the command line, in the same directory as your notebook:

```
jupyter nbconvert --to pdf week3_lab.ipynb
```

3. Look for week3_lab.pdf, open it to examine the output, then **send it to your instructor instead of the notebook**.

Note: week3_lab.ipynb is the name of the notebook file, in the future replace it with the same of the file you want to convert.

In [ ]: