introduction to Matplotlib

In [1]:

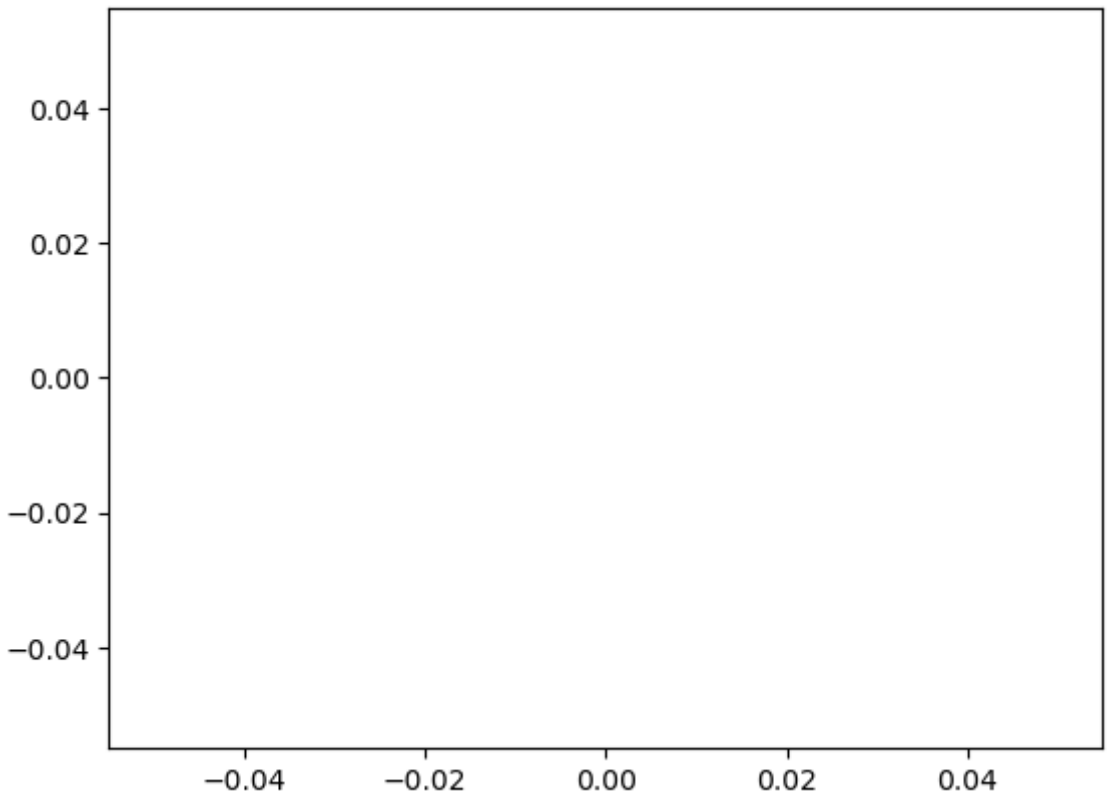
```
%matplotlib inline
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
```

In [2]:

```
plt.plot()
```

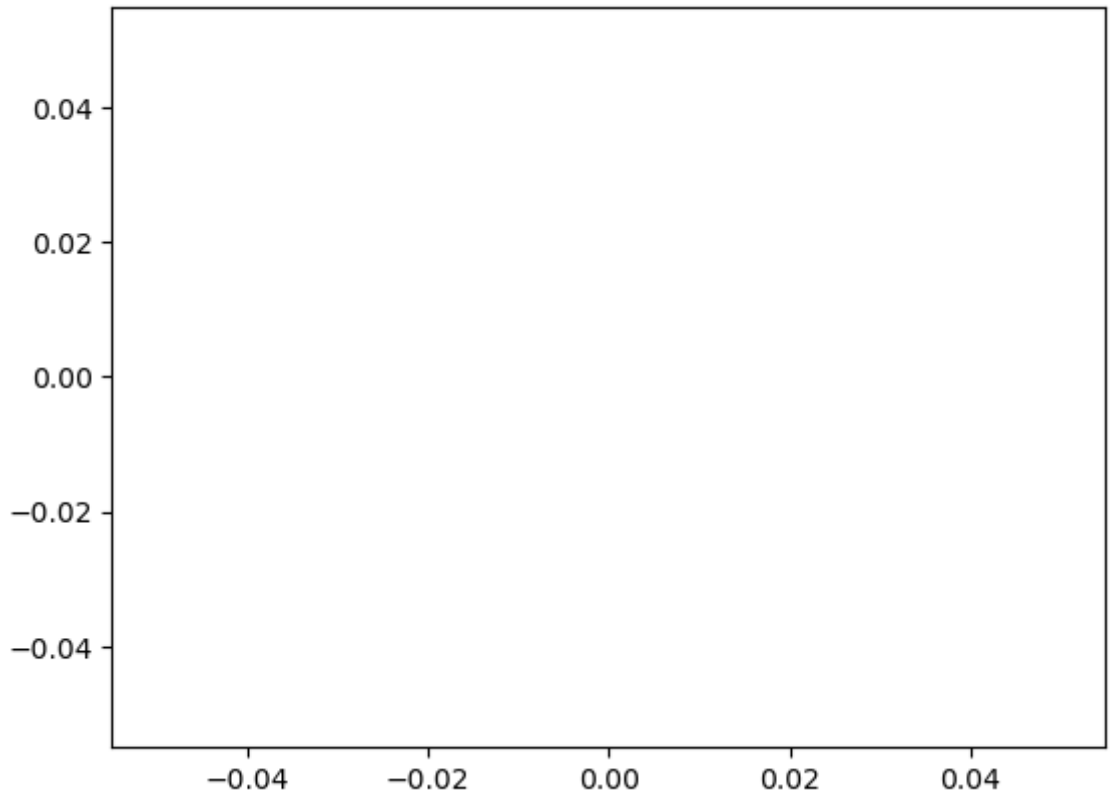
Out[2]:

```
[]
```



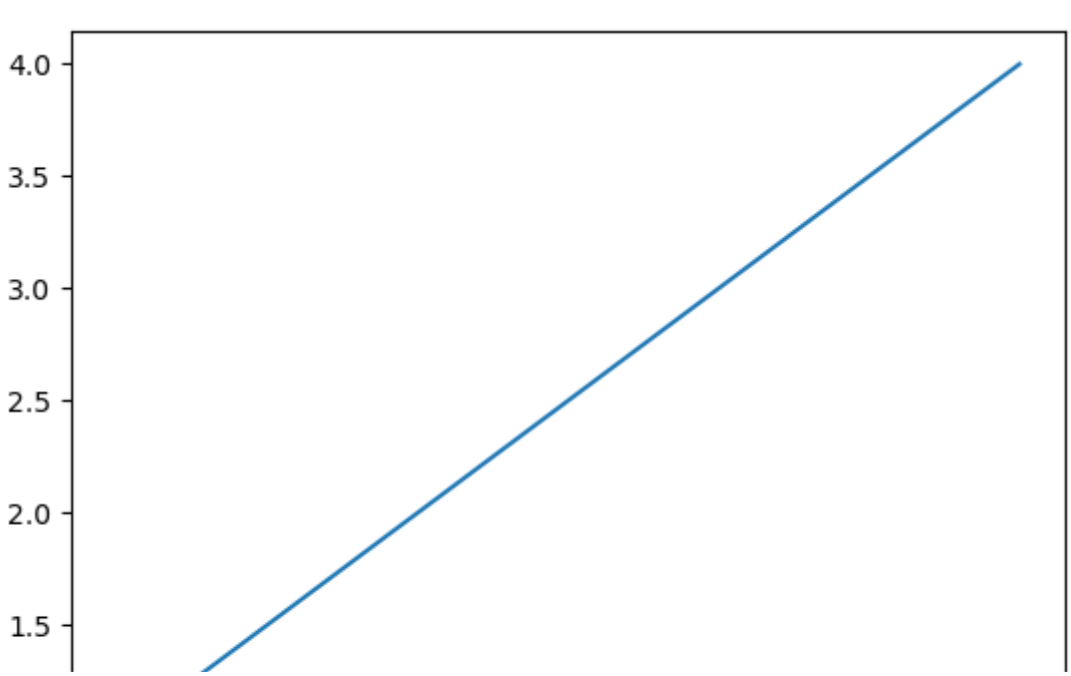
In [3]:

```
plt.plot();
```



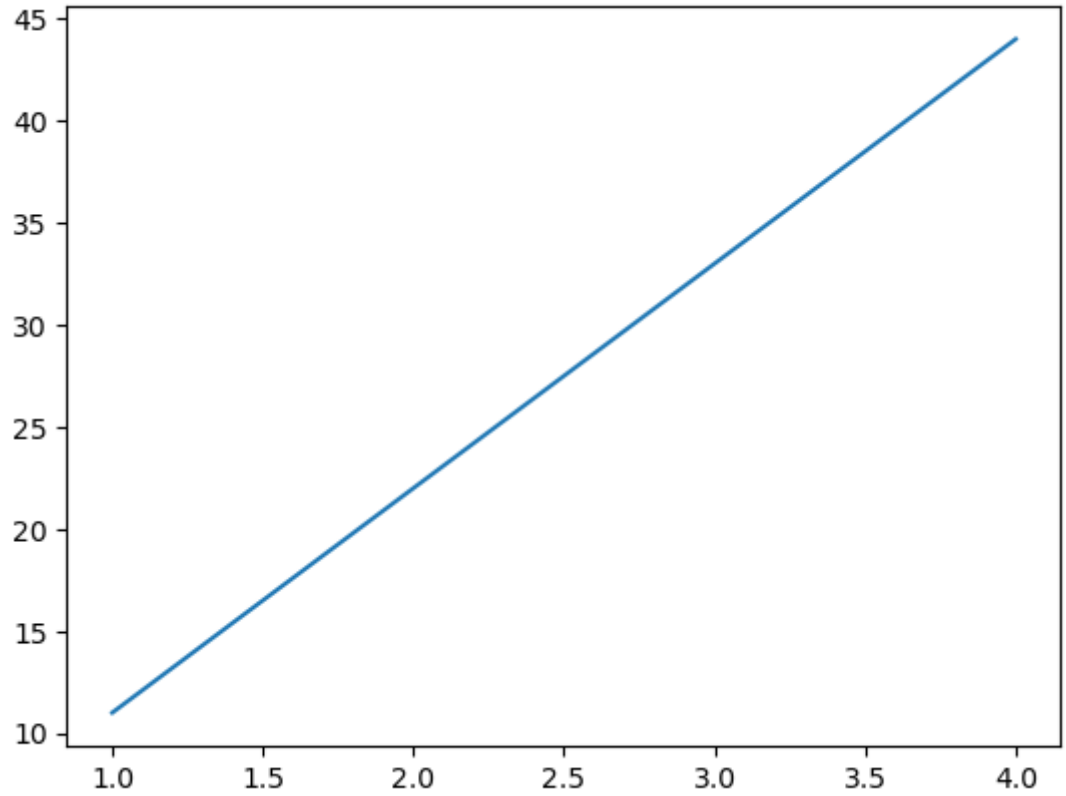
In [4]:

```
plt.plot([1, 2, 3, 4]);
```



In [5]:

```
x=[1, 2, 3, 4]
y=[11, 22, 33, 44]
plt.plot(x,y);
```



In [6]:

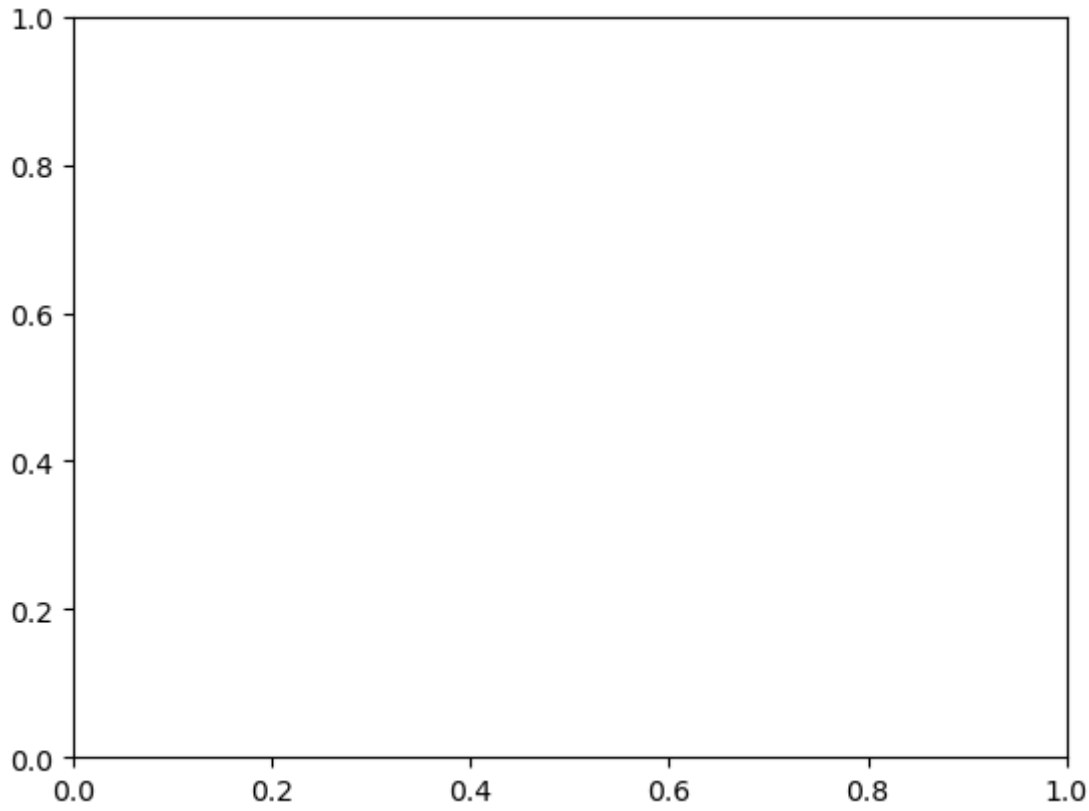
```
#in genral try use the object oriented interface over the pyplot interfase
```

In [7]:

```
# 1st method
fig = plt.figure() #creates a figure
ax=fig.add_subplot() #add some axes
plt.show
```

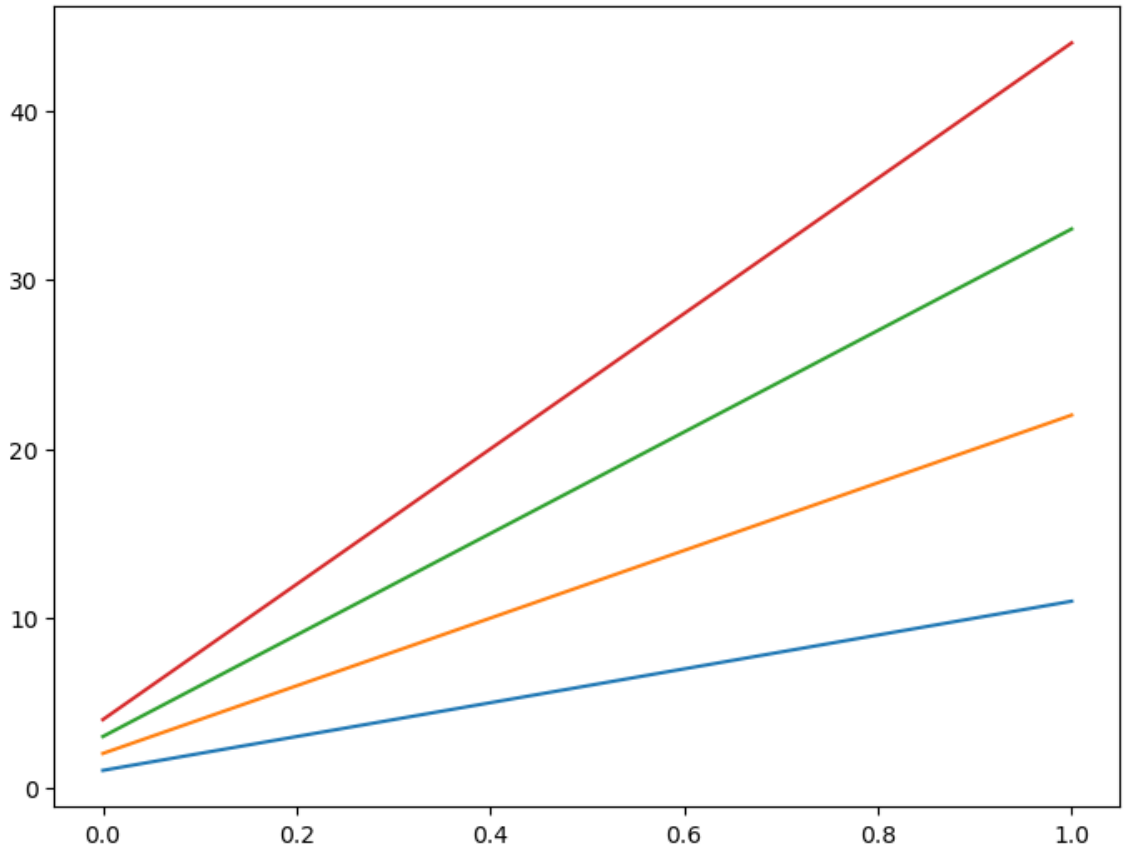
Out[7]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [8]:

```
#2nd method
fig =plt.figure()
ax = fig.add_axes([1, 1, 1, 1])
ax.plot([x, y])
plt.show()
```

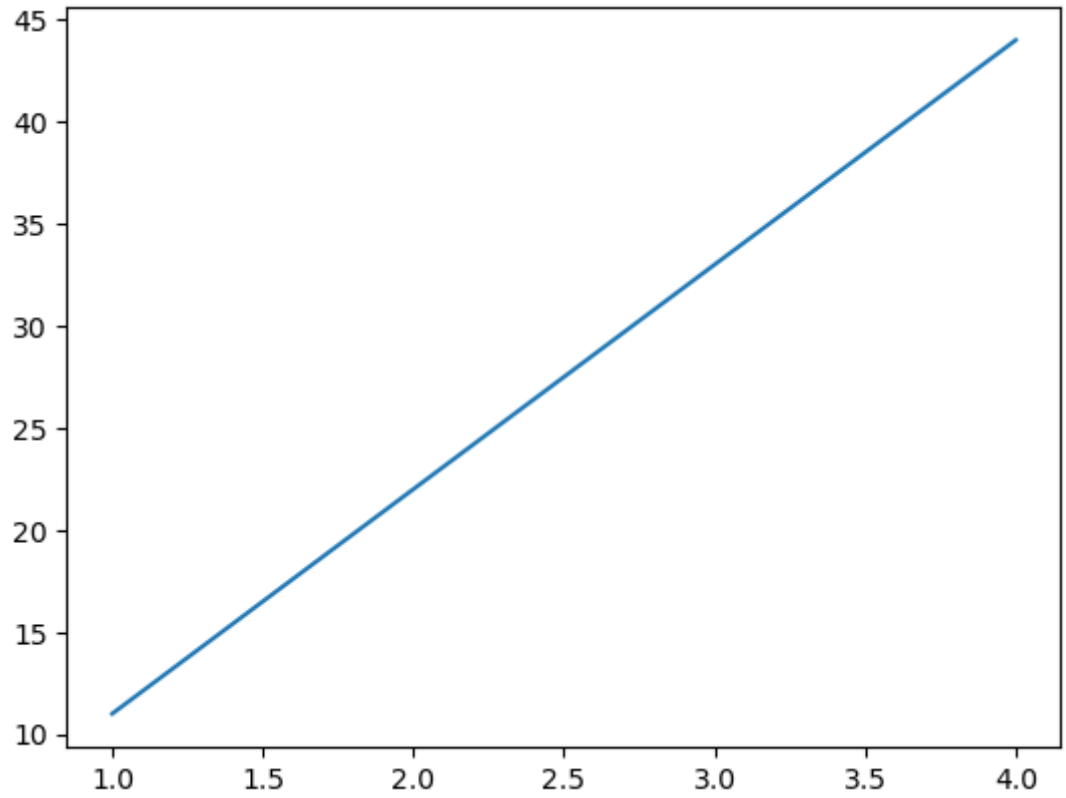


In [9]:

```
#3rd method (recommended)
fig, ax =plt.subplots()
ax.plot(x, y)
```

Out[9]:

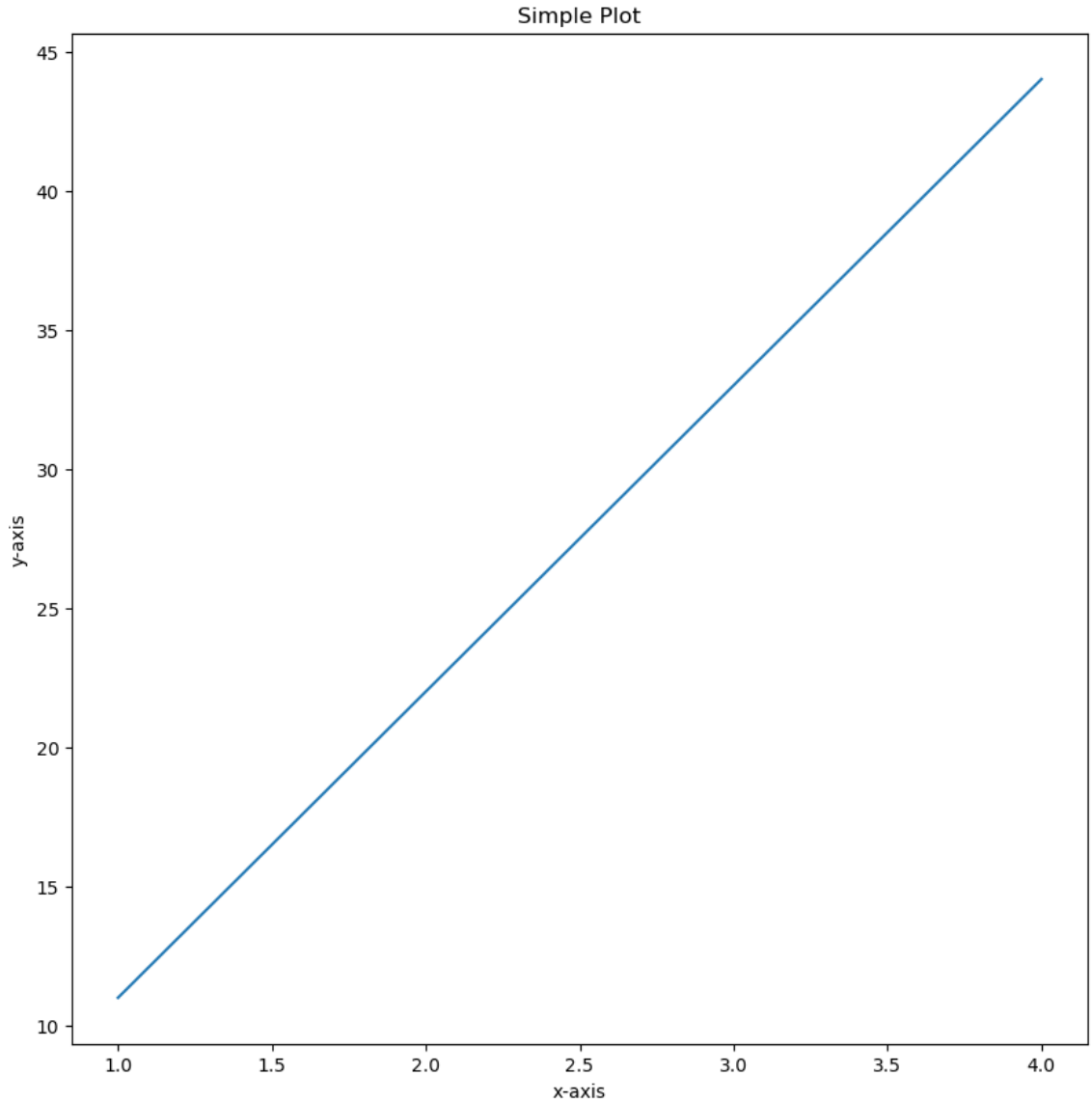
```
[<matplotlib.lines.Line2D at 0x1f9bfa53130>]
```



Matplotlib example workflow

In [10]:

```
# import matplotlib and get it ready for plotting in jupypter
%matplotlib inline
import matplotlib.pyplot as plt
# 1. prepare data
x= [1, 2, 3, 4]
y=[11, 22, 33, 44]
#2. setup plot
fig, ax =plt.subplots(figsize=(10, 10)) #figsize(width, height)
#3 ptot the data
ax.plot(x, y)
#4custamize data
ax.set(title="Simple Plot",
       xlabel="x-axis",
       ylabel="y-axis")
#5save and show
fig.savefig("sample-plot.png")
```



making figures with numpy

we want

- Line plot
- Scatter plot
- Bar plot
- Histogram
- subplots

In [11]:

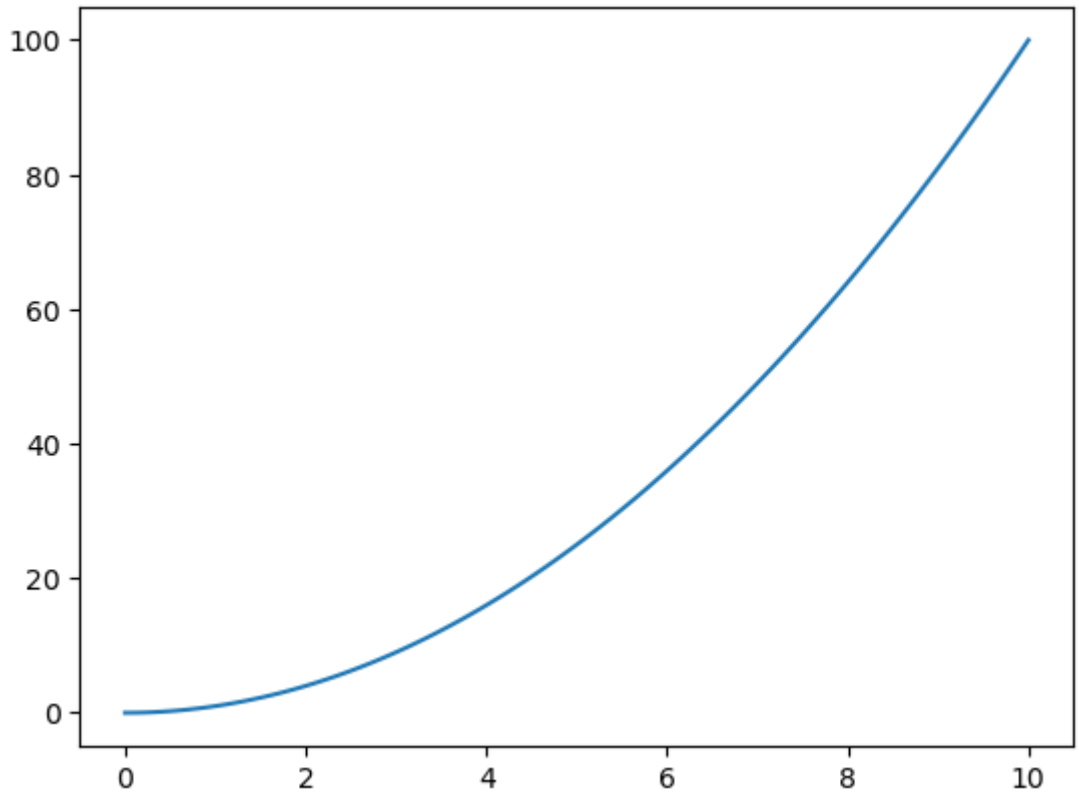
```
x=np.linspace(0, 10, 100)
x[:10]
```

Out[11]:

```
array([0.        , 0.1010101 , 0.2020202 , 0.3030303 , 0.4040404 ,
       0.50505051, 0.60606061, 0.70707071, 0.80808081, 0.90909091])
```

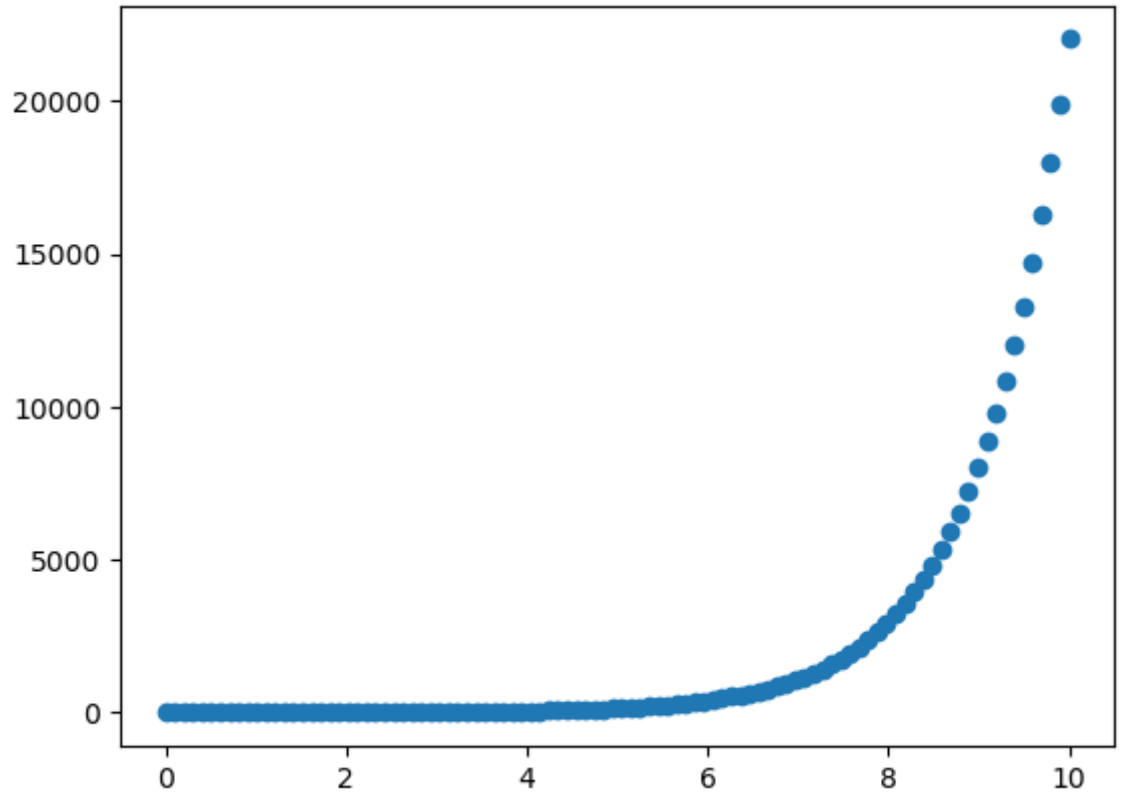
In [12]:

```
#plot the data and create a line plot
fig, ax= plt.subplots()
ax.plot(x, x**2);
```



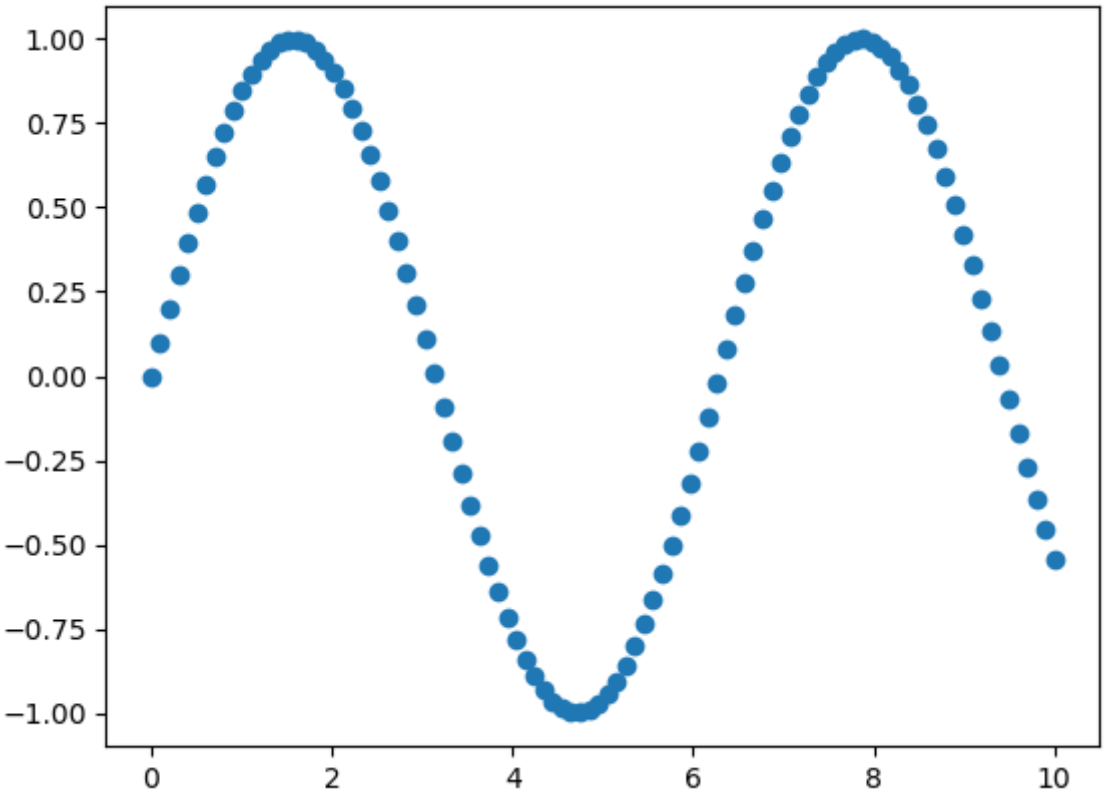
In [13]:

```
#use same data to make a scatter
fig, ax = plt.subplots()
ax.scatter(x, np.exp(x));
```



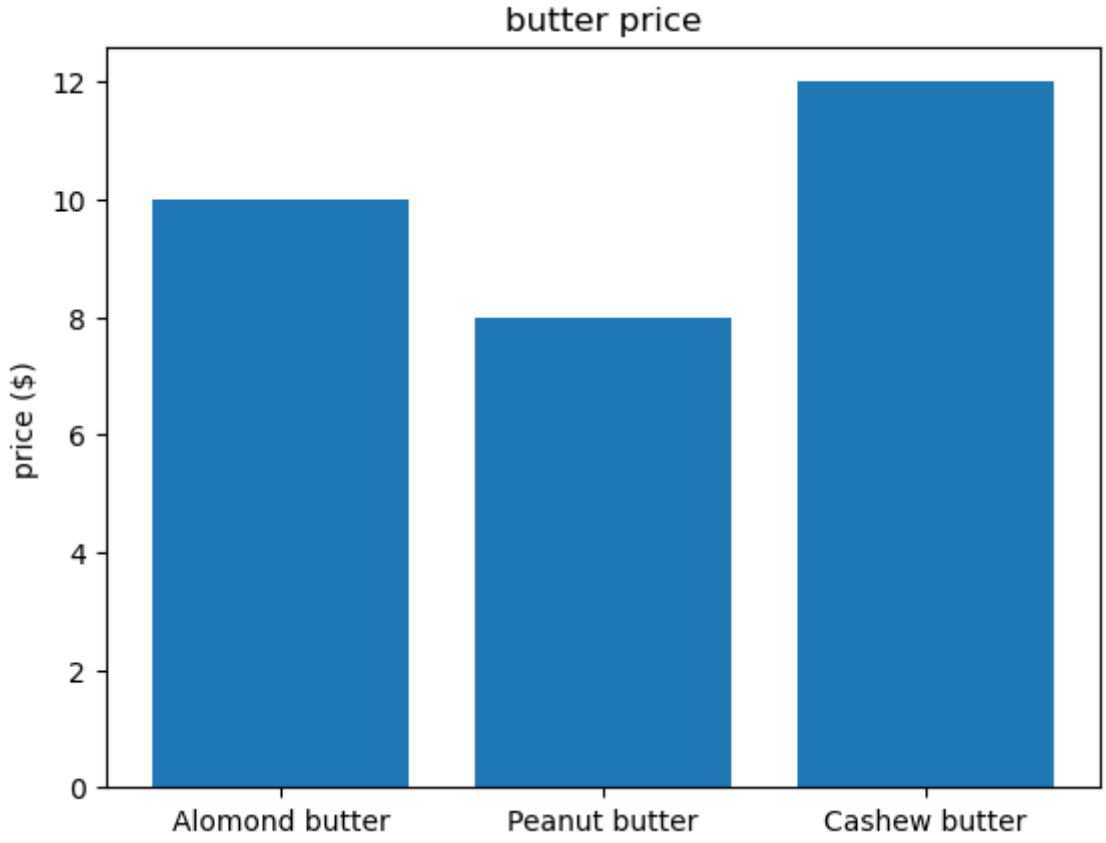
In [14]:

```
#Another scatter plot
fig, ax =plt.subplots();
ax.scatter(x, np.sin(x));
```



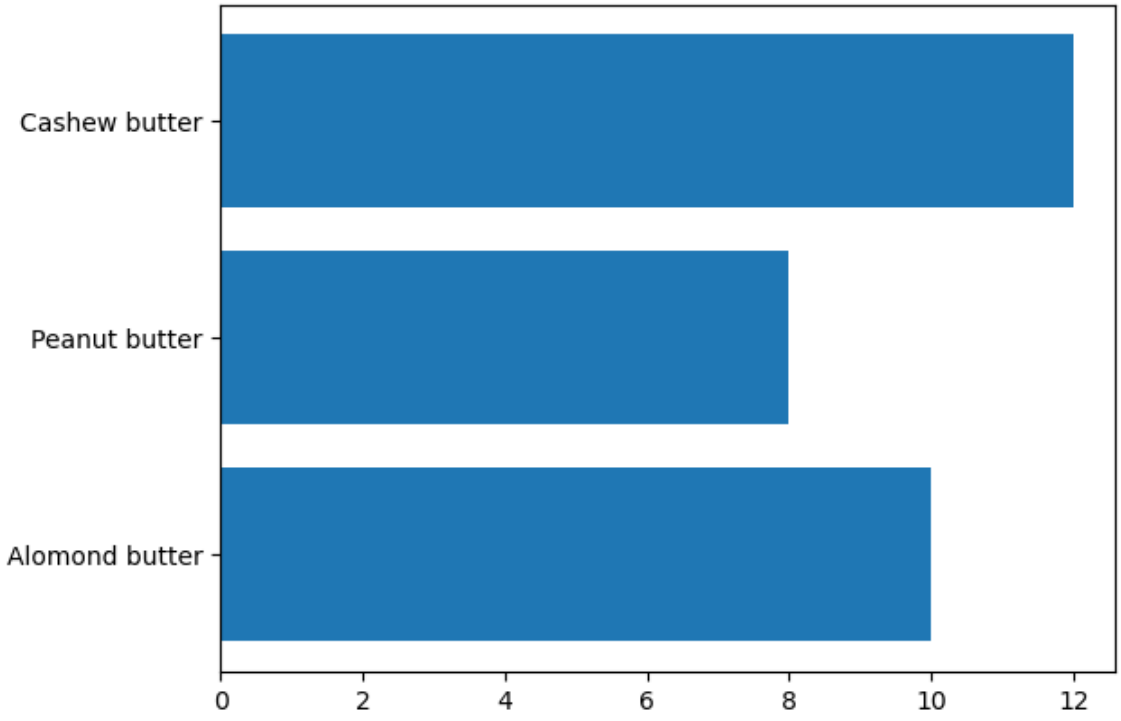
In [15]:

```
#make a plot from dictionary
nut_butter_prices={"Alomond butter": 10,
                   "Peanut butter": 8,
                   "Cashew butter": 12}
fig, ax =plt.subplots()
ax.bar(nut_butter_prices.keys(), nut_butter_prices.values())
ax.set(title="butter price",
       ylabel="price ($)");
```



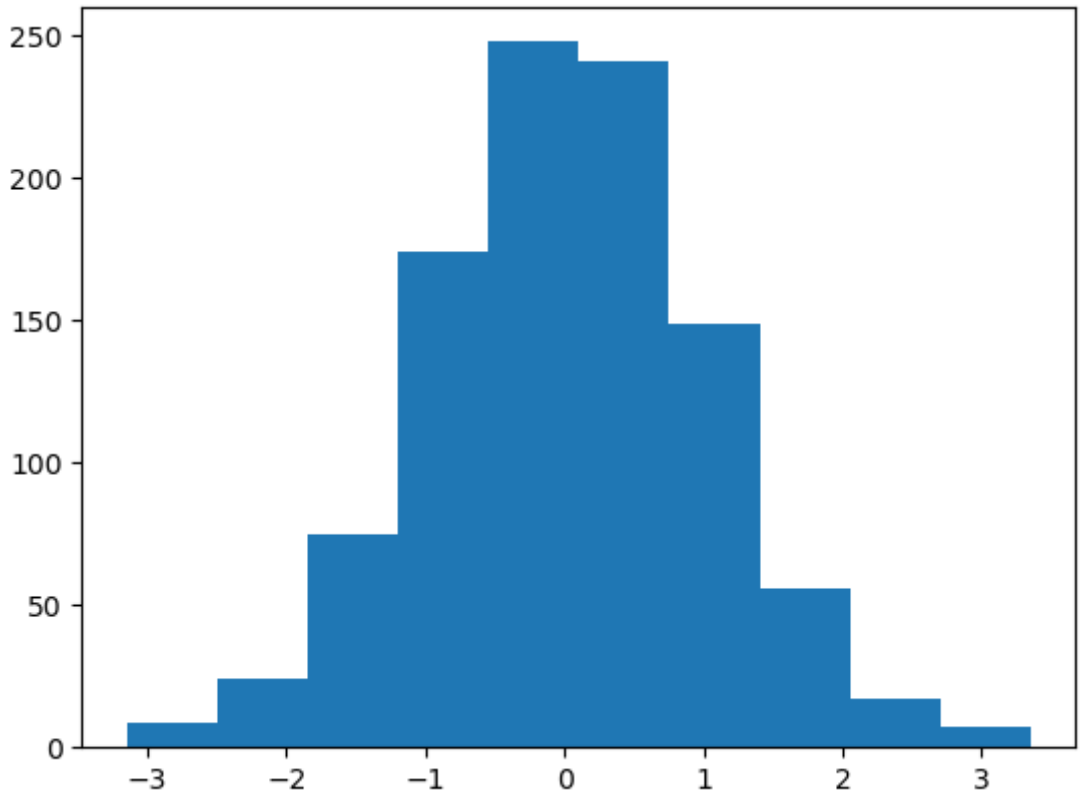
In [16]:

```
fig, ax =plt.subplots()
ax.barh(list(nut_butter_prices.keys()), list(nut_butter_prices.values()));
```



In [17]:

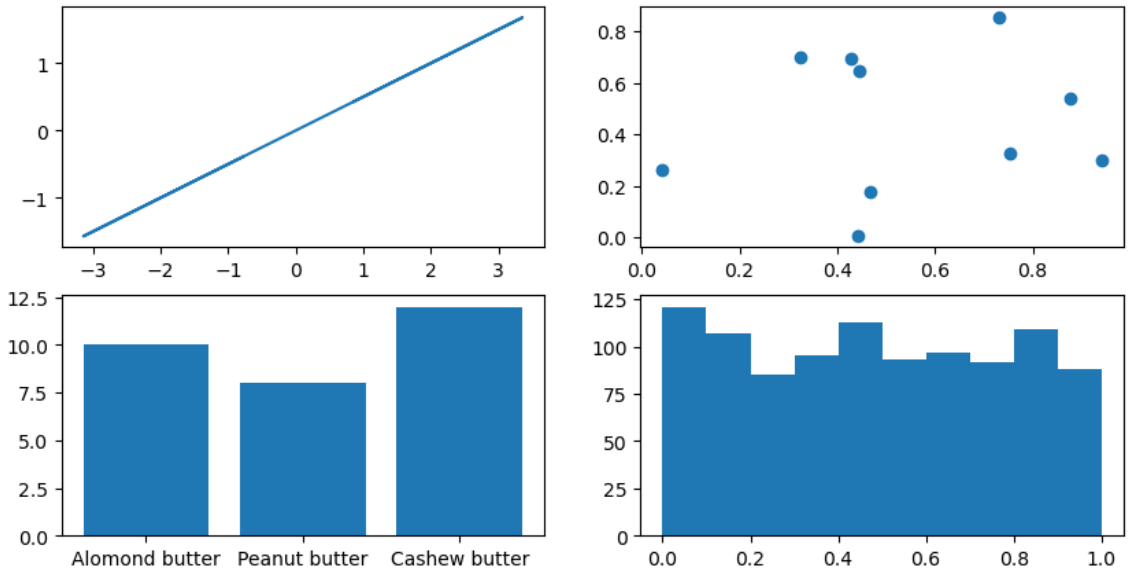
```
#make some data for histogam
x =np.random.randn(1000)
fig, ax = plt.subplots()
ax.hist(x);
```



two option for subplots

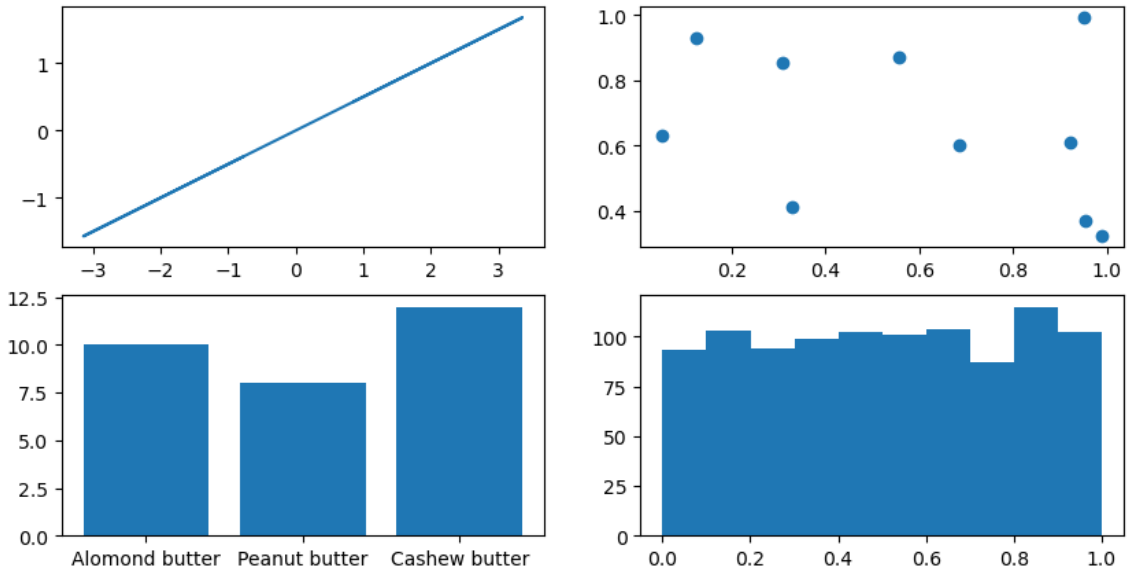
In [18]:

```
#subplots option 1
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(nrows=2,
                                             ncols=2,
                                             figsize=(10, 5))
#plot to each diffrent data
ax1.plot(x, x/2);
ax2.scatter(np.random.random(10), np.random.random(10));
ax3.bar(nut_butter_prices.keys(), nut_butter_prices.values());
ax4.hist(np.random.random(1000));
```



In [19]:

```
#subplots option two
fig, ax = plt.subplots(nrows=2,
                       ncols=2,
                       figsize=(10, 5))
#plot to each diffrent index
ax[0, 0].plot(x, x/2);
ax[0, 1].scatter(np.random.random(10), np.random.random(10));
ax[1, 0].bar(nut_butter_prices.keys(), nut_butter_prices.values());
ax[1, 1].hist(np.random.random(1000));
```



plotting from panda dataframe

In [20]:

```
import pandas as pd
```

In [21]:

```
#make dataframe
car_sales =pd.read_csv("car-sales.csv")
car_sales
```

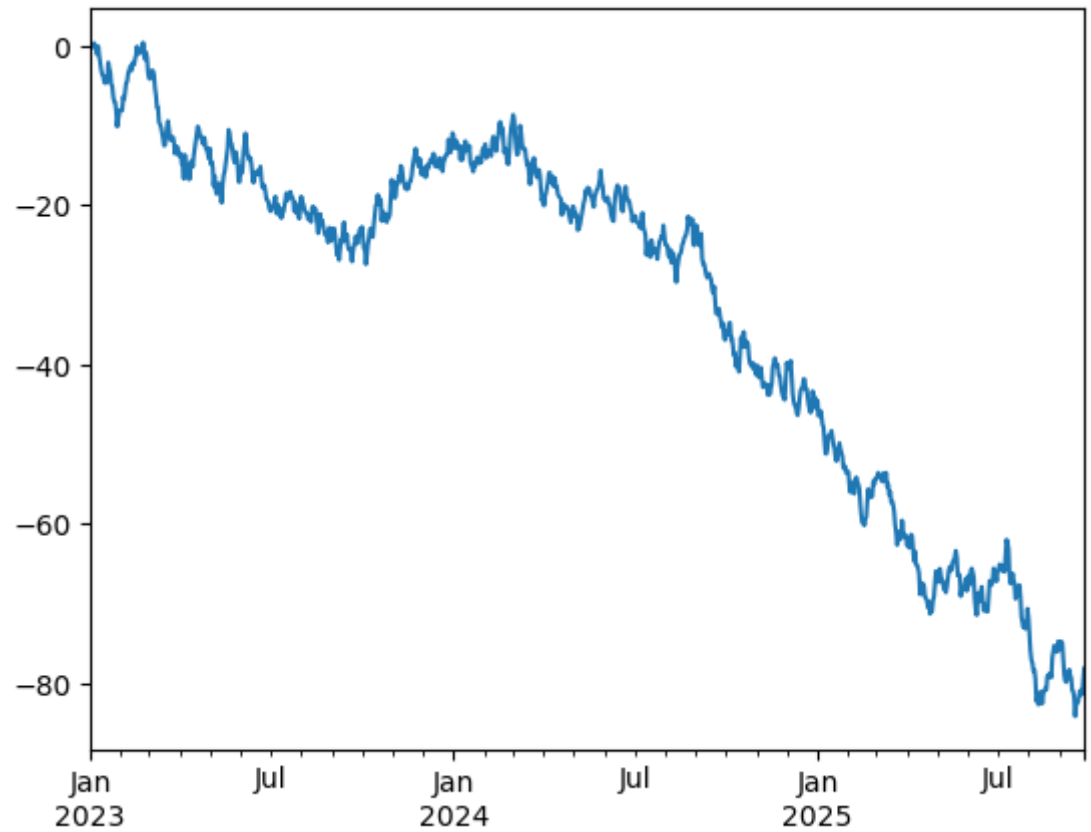
Out[21]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [22]:

```
ts = pd.Series(np.random.randn(1000), index=pd.date_range("1/1/2023", periods=1000))
ts = ts.cumsum() #it add data next to other

ts.plot();
```



In [23]:

```
car_sales
```

Out[23]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	$4,000.00
1	Honda	Red	87899	4	$5,000.00
2	Toyota	Blue	32549	3	$7,000.00
3	BMW	Black	11179	5	$22,000.00
4	Nissan	White	213095	4	$3,500.00
5	Toyota	Green	99213	4	$4,500.00
6	Honda	Blue	45698	4	$7,500.00
7	Honda	Blue	54738	4	$7,000.00
8	Toyota	White	60000	4	$6,250.00
9	Nissan	White	31600	4	$9,700.00

In [24]:

```
car_sales["Price"] = car_sales["Price"].str.replace('[\$\,\.]', '')
car_sales
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_16504\2276578600.py:1: FutureW
arning: The default value of regex will change from True to False in a fut
ure version.
  car_sales["Price"] = car_sales["Price"].str.replace('[\$\,\.]', '')
```

Out[24]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	400000
1	Honda	Red	87899	4	500000
2	Toyota	Blue	32549	3	700000
3	BMW	Black	11179	5	2200000
4	Nissan	White	213095	4	350000
5	Toyota	Green	99213	4	450000
6	Honda	Blue	45698	4	750000
7	Honda	Blue	54738	4	700000
8	Toyota	White	60000	4	625000
9	Nissan	White	31600	4	970000

In [25]:

```
type(car_sales["Price"][0])
```

Out[25]:

```
str
```

In [26]:

```
#remove last two zero
car_sales["Price"] = car_sales["Price"].str[:-2]
car_sales
```

Out[26]:

	Make	Colour	Odometer (KM)	Doors	Price
0	Toyota	White	150043	4	4000
1	Honda	Red	87899	4	5000
2	Toyota	Blue	32549	3	7000
3	BMW	Black	11179	5	22000
4	Nissan	White	213095	4	3500
5	Toyota	Green	99213	4	4500
6	Honda	Blue	45698	4	7500
7	Honda	Blue	54738	4	7000
8	Toyota	White	60000	4	6250
9	Nissan	White	31600	4	9700

In [27]:

```
car_sales["Sales Date"] = pd.date_range("1/1/2020", periods=len(car_sales))
car_sales
```

Out[27]:

	Make	Colour	Odometer (KM)	Doors	Price	Sales Date
0	Toyota	White	150043	4	4000	2020-01-01
1	Honda	Red	87899	4	5000	2020-01-02
2	Toyota	Blue	32549	3	7000	2020-01-03
3	BMW	Black	11179	5	22000	2020-01-04
4	Nissan	White	213095	4	3500	2020-01-05
5	Toyota	Green	99213	4	4500	2020-01-06
6	Honda	Blue	45698	4	7500	2020-01-07
7	Honda	Blue	54738	4	7000	2020-01-08
8	Toyota	White	60000	4	6250	2020-01-09
9	Nissan	White	31600	4	9700	2020-01-10

In [28]:

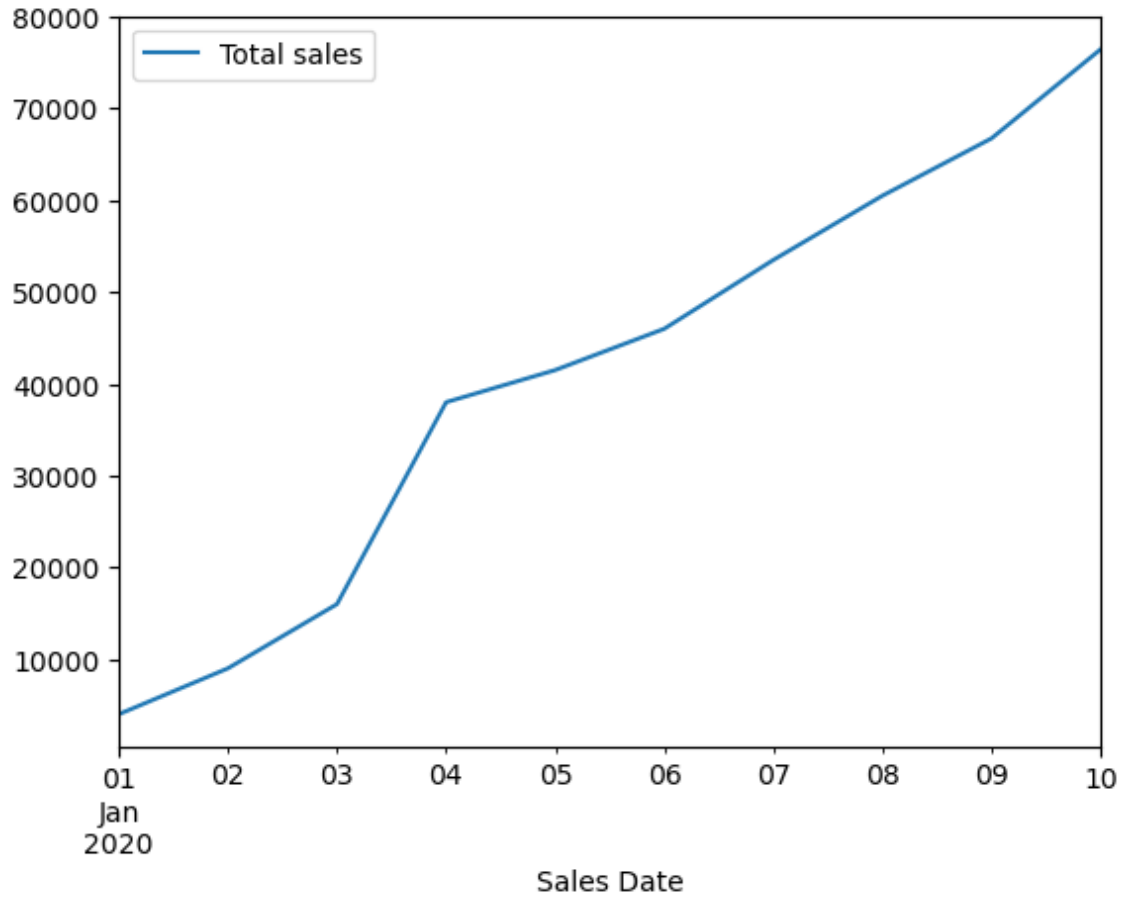
```
car_sales["Total sales"] = car_sales["Price"].astype(int).cumsum()
car_sales
```

Out[28]:

	Make	Colour	Odometer (KM)	Doors	Price	Sales Date	Total sales
0	Toyota	White	150043	4	4000	2020-01-01	4000
1	Honda	Red	87899	4	5000	2020-01-02	9000
2	Toyota	Blue	32549	3	7000	2020-01-03	16000
3	BMW	Black	11179	5	22000	2020-01-04	38000
4	Nissan	White	213095	4	3500	2020-01-05	41500
5	Toyota	Green	99213	4	4500	2020-01-06	46000
6	Honda	Blue	45698	4	7500	2020-01-07	53500
7	Honda	Blue	54738	4	7000	2020-01-08	60500
8	Toyota	White	60000	4	6250	2020-01-09	66750
9	Nissan	White	31600	4	9700	2020-01-10	76450

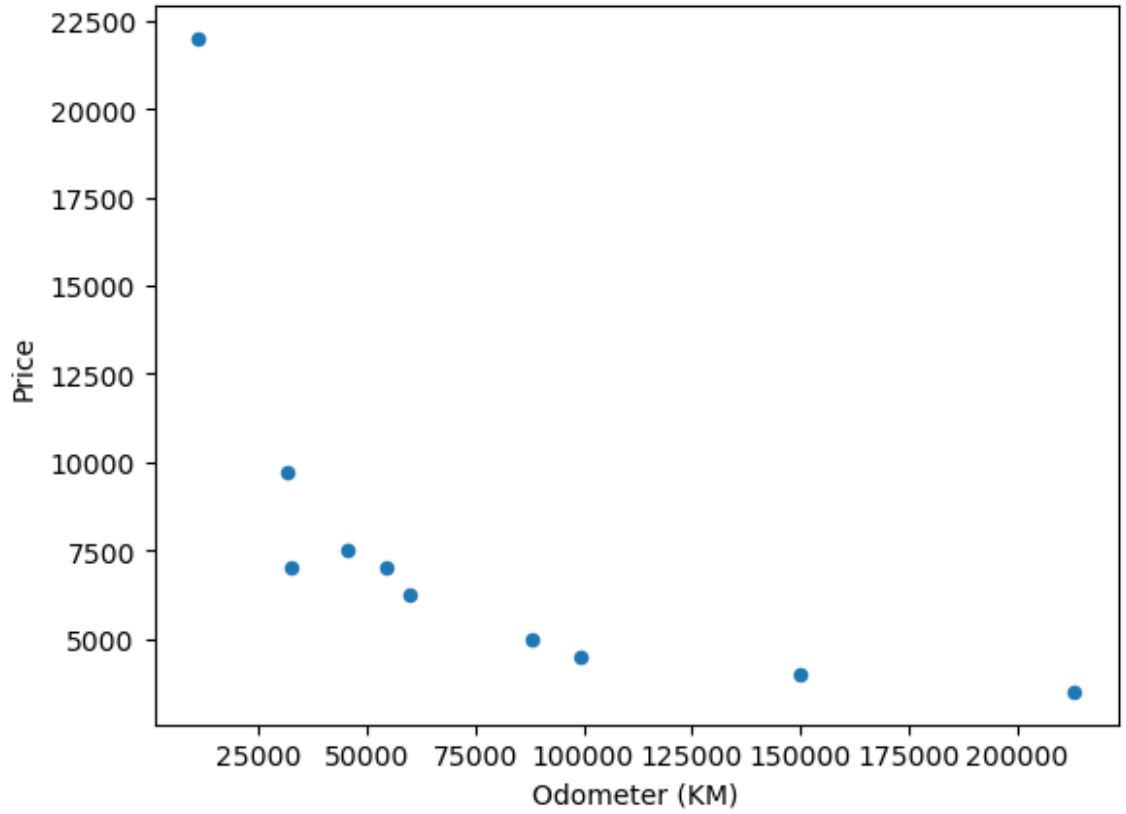
In [29]:

```
#lets plot total sales
car_sales.plot(x="Sales Date", y="Total sales");
```



In [30]:

```
#change price into int
car_sales["Price"] =car_sales["Price"].astype(int)
car_sales.plot(x="Odometer (KM)", y="Price", kind="scatter");
```



In [31]:

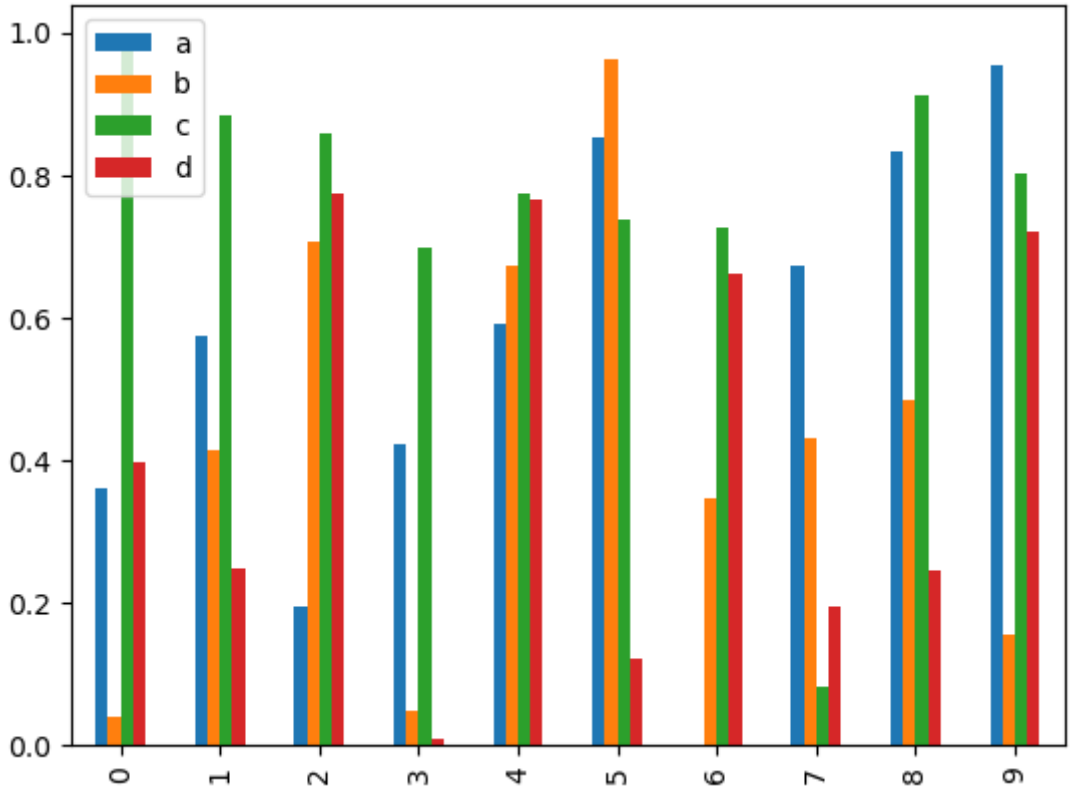
```
#bar graph
x=np.random.rand(10, 4)
x
#turn into data frame
df=pd.DataFrame(x, columns=['a', 'b', 'c', 'd'])
df
```

Out[31]:

	a	b	c	d
0	0.362398	0.041412	0.989799	0.398084
1	0.573974	0.414272	0.885453	0.248564
2	0.193885	0.706265	0.858015	0.774717
3	0.424230	0.047917	0.698737	0.008659
4	0.590290	0.673334	0.774345	0.767166
5	0.853222	0.962687	0.738990	0.122743
6	0.001790	0.346463	0.726064	0.663162
7	0.672246	0.431655	0.082708	0.194112
8	0.832900	0.485995	0.913110	0.244491
9	0.954108	0.156227	0.801385	0.720911

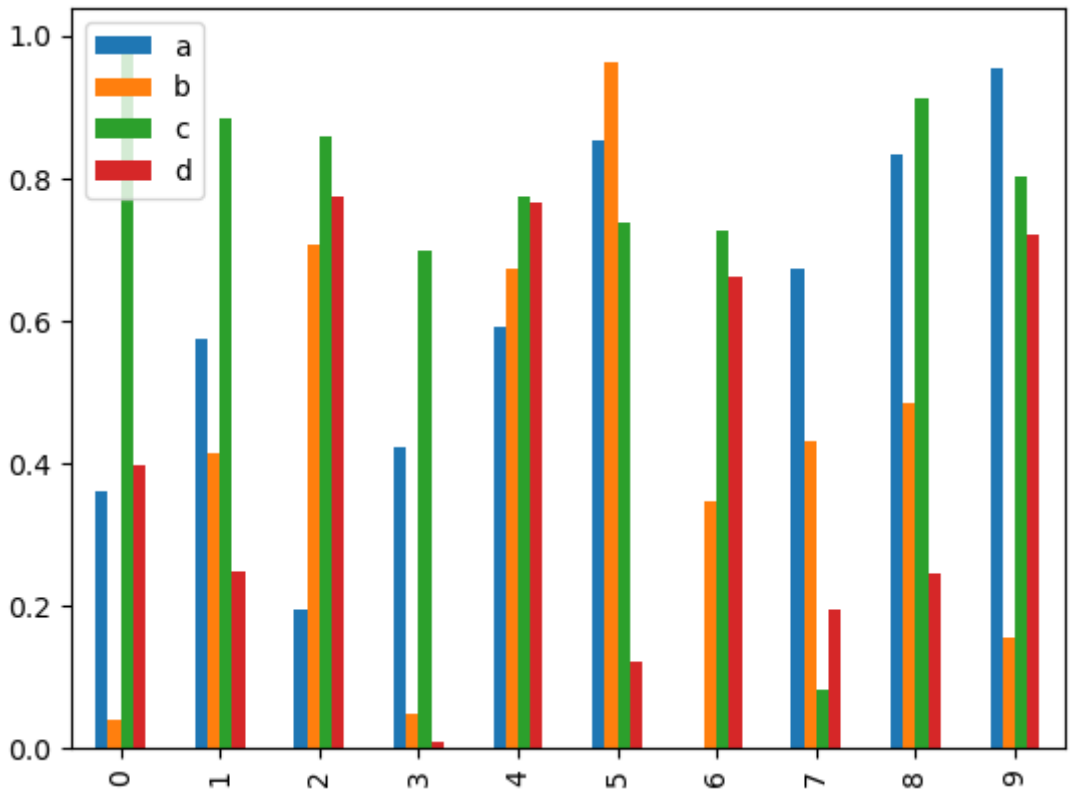
In [32]:

```
df.plot.bar();
```



In [33]:

```
df.plot(kind="bar");
```



In [34]:

```
car_sales
```

Out[34]:

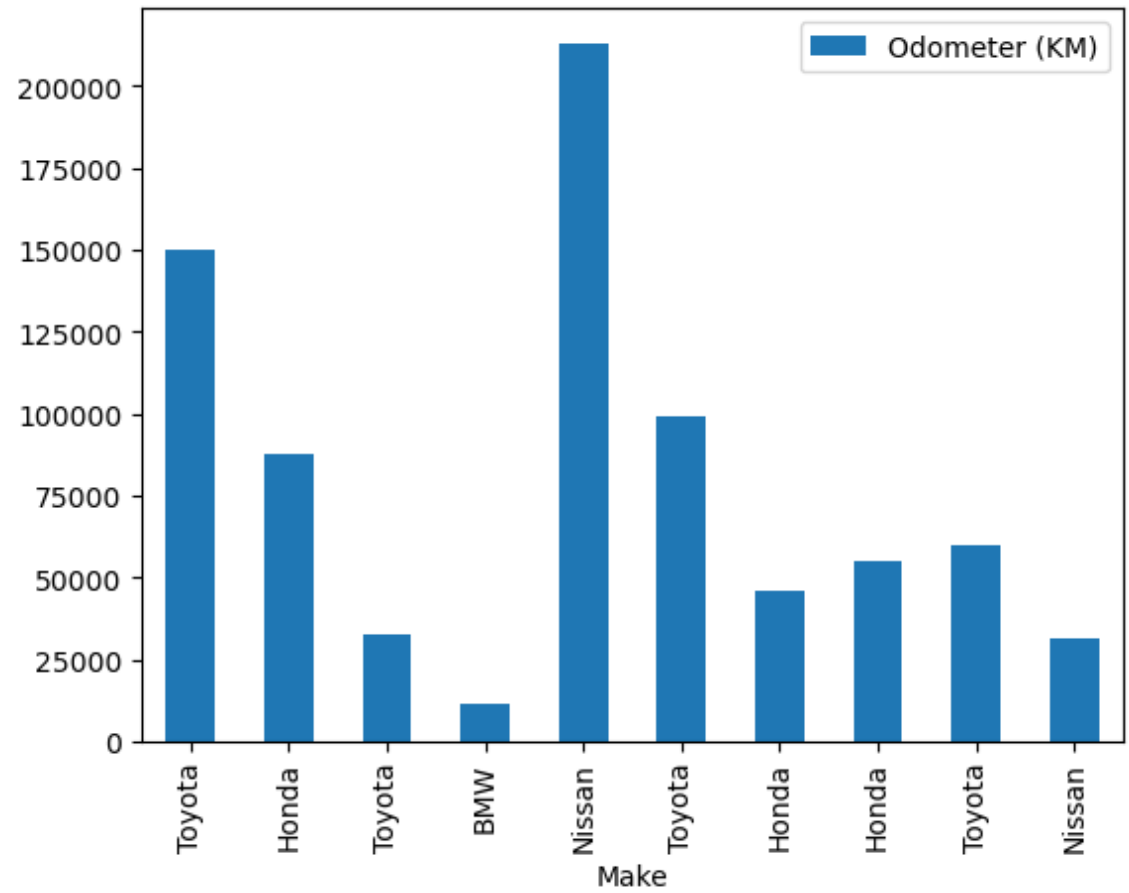
	Make	Colour	Odometer (KM)	Doors	Price	Sales Date	Total sales
0	Toyota	White	150043	4	4000	2020-01-01	4000
1	Honda	Red	87899	4	5000	2020-01-02	9000
2	Toyota	Blue	32549	3	7000	2020-01-03	16000
3	BMW	Black	11179	5	22000	2020-01-04	38000
4	Nissan	White	213095	4	3500	2020-01-05	41500
5	Toyota	Green	99213	4	4500	2020-01-06	46000
6	Honda	Blue	45698	4	7500	2020-01-07	53500
7	Honda	Blue	54738	4	7000	2020-01-08	60500
8	Toyota	White	60000	4	6250	2020-01-09	66750
9	Nissan	White	31600	4	9700	2020-01-10	76450

In [35]:

```
car_sales.plot(x="Make", y="Odometer (KM)", kind="bar")
```

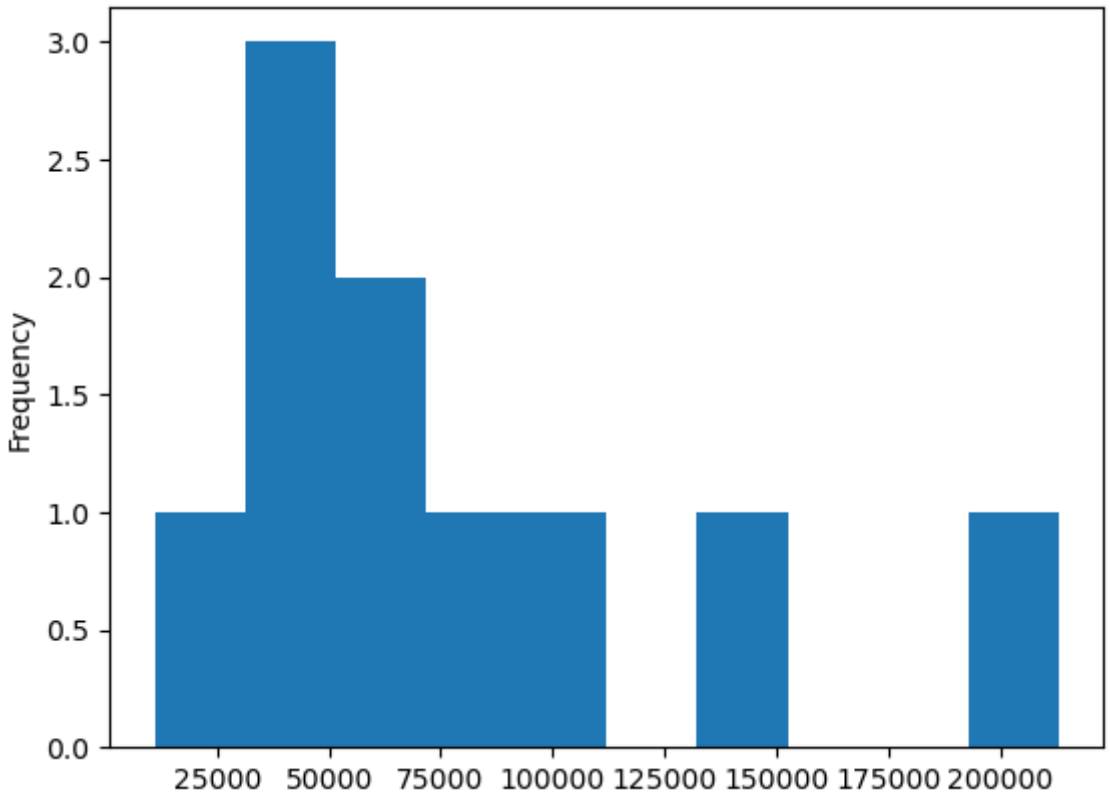
Out[35]:

```
<AxesSubplot: xlabel='Make'>
```



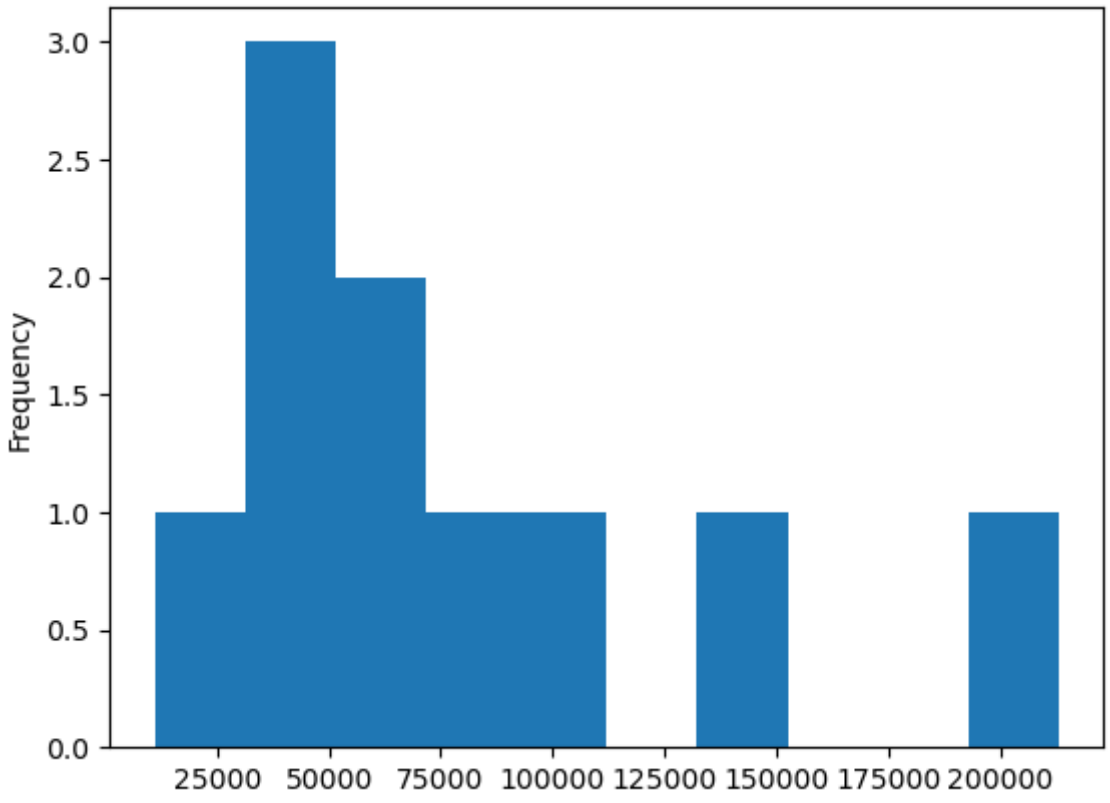
In [36]:

```
#Hitogram
car_sales["Odometer (KM)"].plot.hist();
```



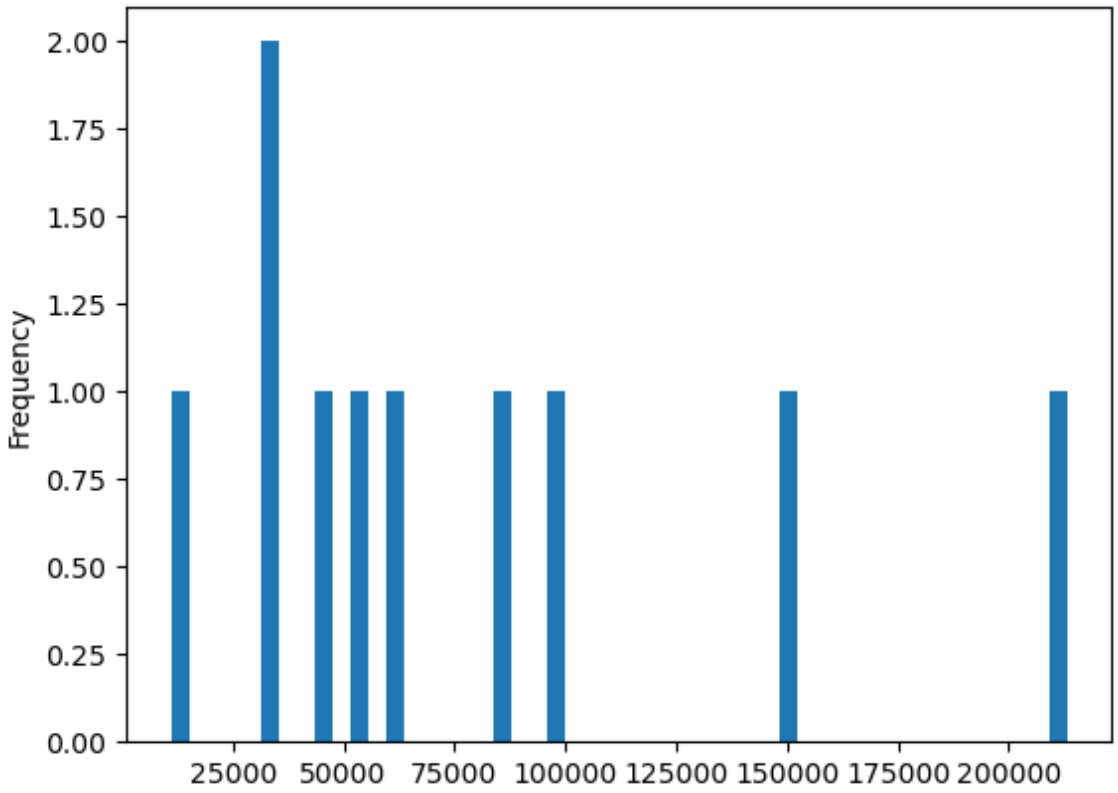
In [37]:

```
car_sales["Odometer (KM)"].plot(kind="hist");
```



In [38]:

```
car_sales["Odometer (KM)"].plot.hist(bins=50);# bins is size sample
```



In [39]:

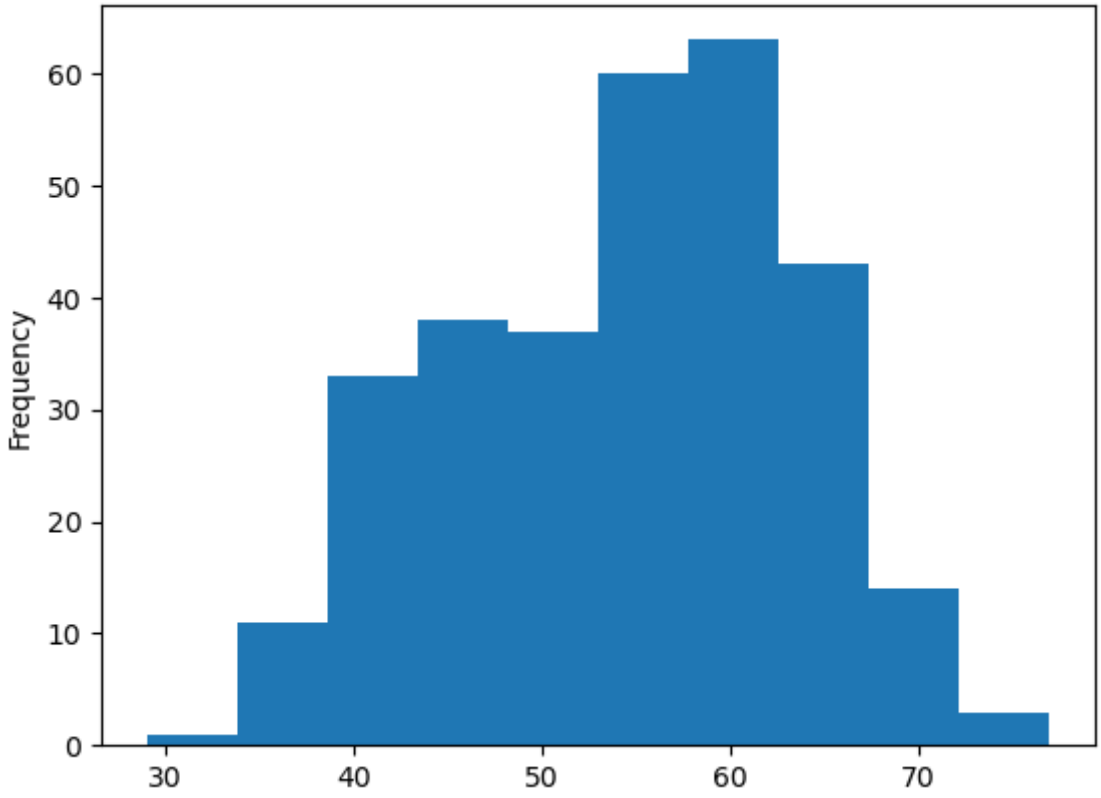
```
#Let's try on another
heart_disease = pd.read_csv("heart-disease.csv")
heart_disease.head()
```

Out[39]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	targ
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	

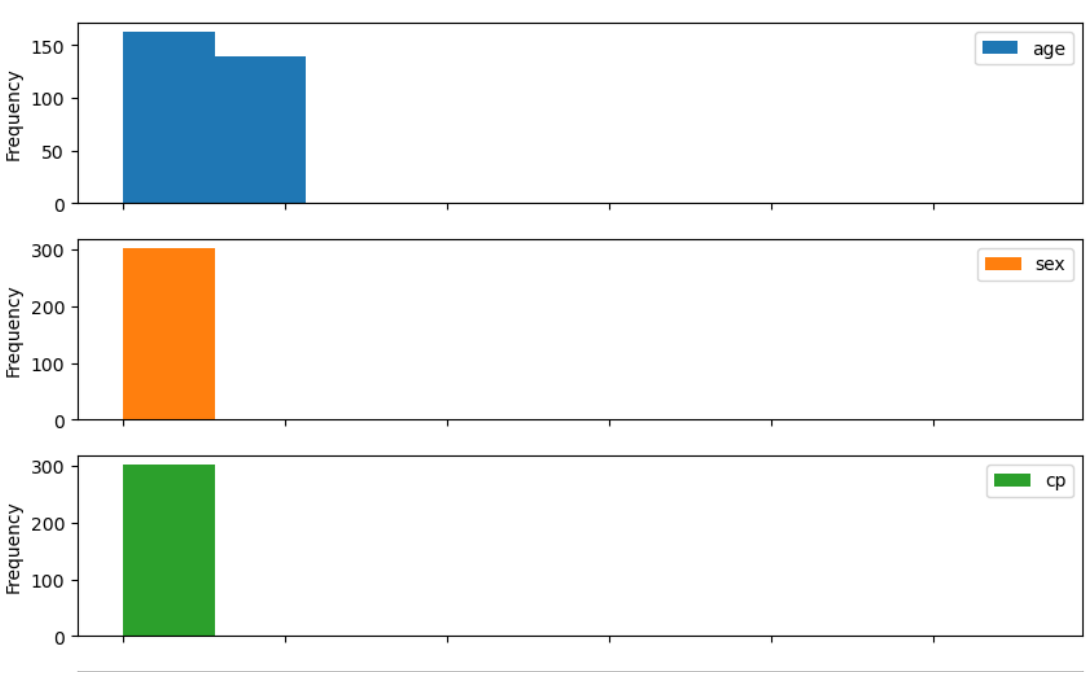
In [40]:

```
#create a histogram
heart_disease["age"].plot.hist(bins=10);
```



In [41]:

```
heart_disease.plot.hist(figsize=(10, 30), subplots=True);
```



which one should you use (py plot vs matplotlib 00)

- when ploting quickely use pyplot method

- for advance use OO method

In [42]:

```
heart_disease
```

Out[42]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	ta
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	

303 rows × 14 columns

In [43]:

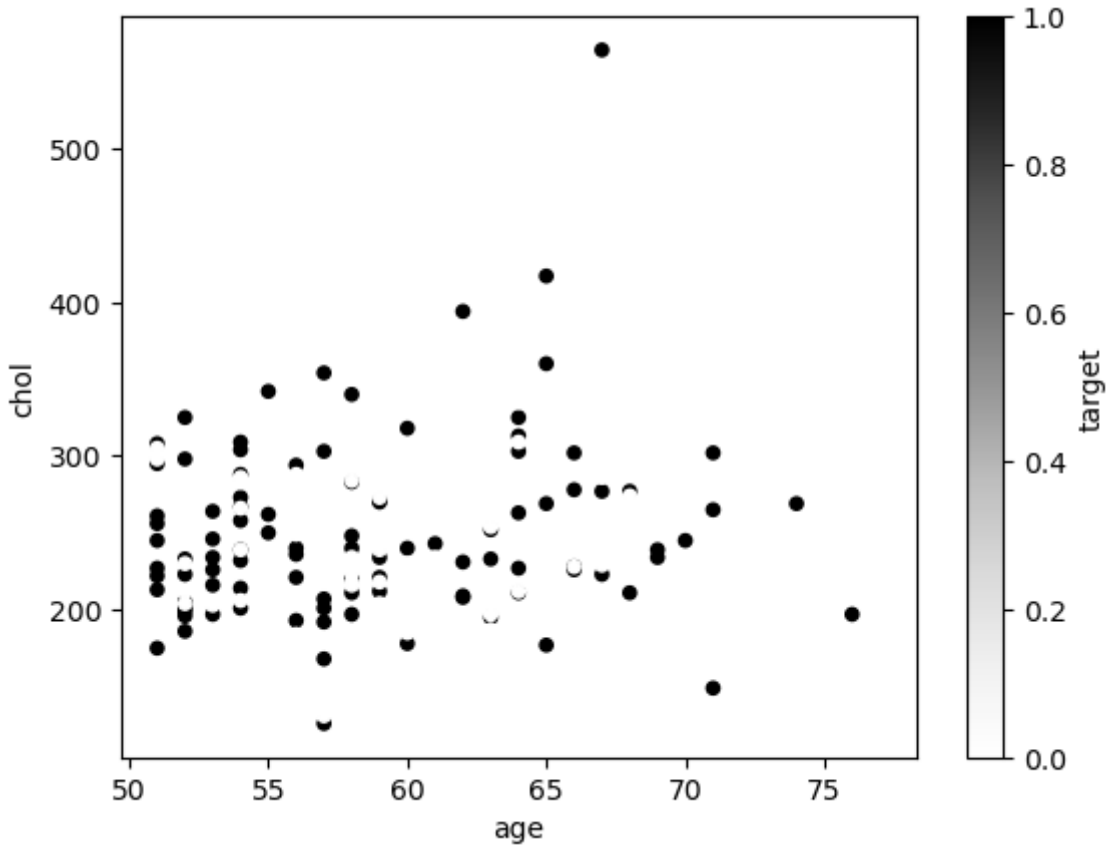
```
over_50 = heart_disease[heart_disease["age"] >50]
over_50.head()
```

Out[43]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	targ
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	

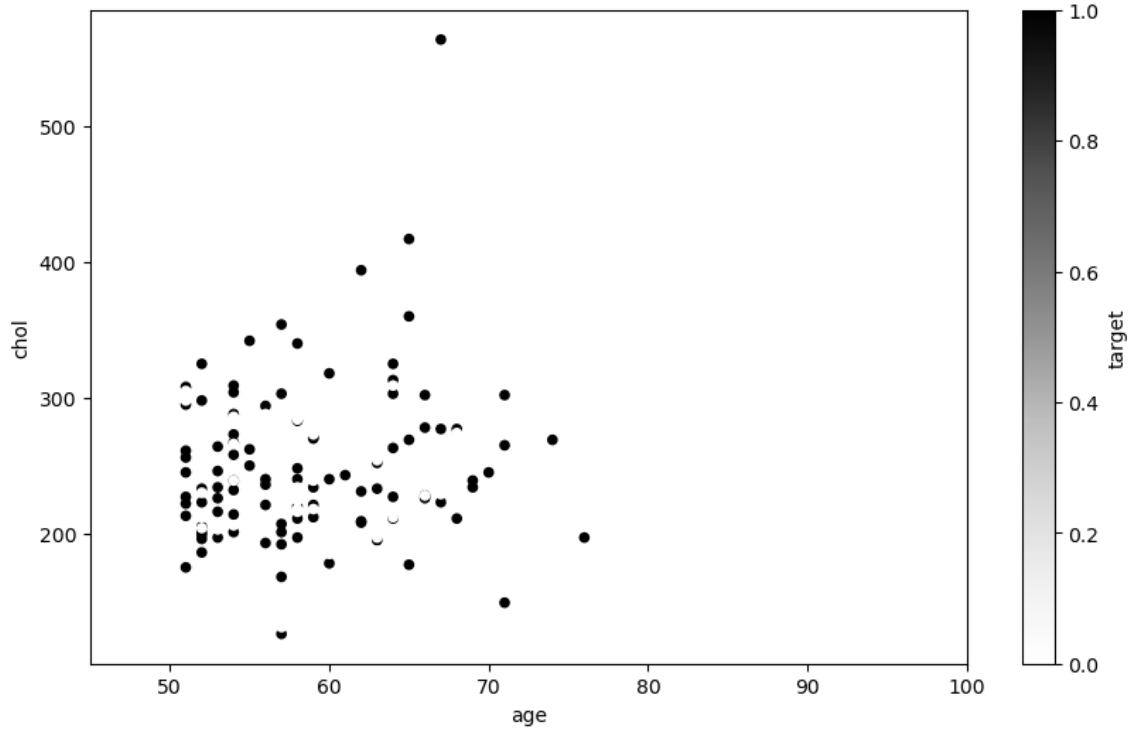
In [44]:

```
#pyplot method
over_50.plot(kind='scatter',
             x='age',
             y='chol',
             c='target');
```



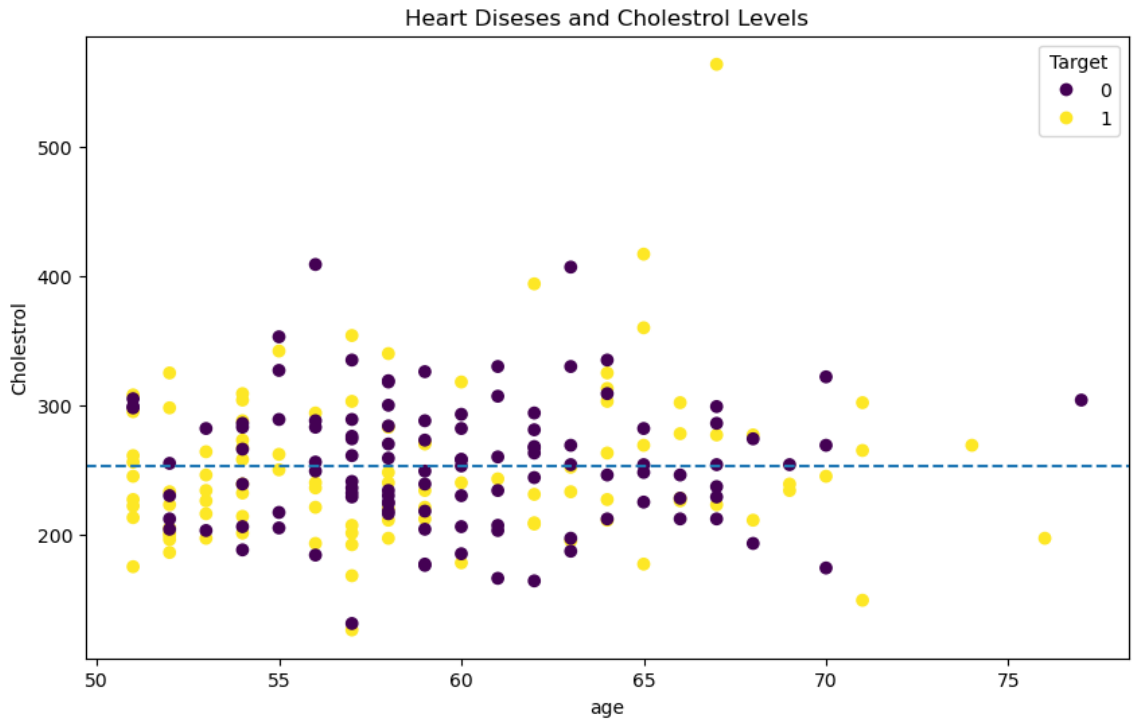
In [45]:

```
#OO method mixted with pyplot
fix, ax = plt.subplots(figsize=(10, 6))
over_50.plot(kind='scatter',
             x='age',
             y='chol',
             c='target',
             ax=ax);
ax.set_xlim([45, 100]);
```



In [54]:

```
#oo method from sracth
fig, ax = plt.subplots(figsize=(10, 6))
#plot the data
scatter= ax.scatter(x=over_50["age"],
                    y=over_50["chol"],
                    c=over_50["target"]);
#custmise the plot
ax.set(title="Heart Diseses and Cholestrol Levels",
       xlabel="age",
       ylabel="Cholestrol");
#add a legend
ax.legend(*scatter.legend_elements(), title="Target");
#add a horizontol line
ax.axhline(over_50["chol"].mean(),
           linestyle='--');
```



In [56]:

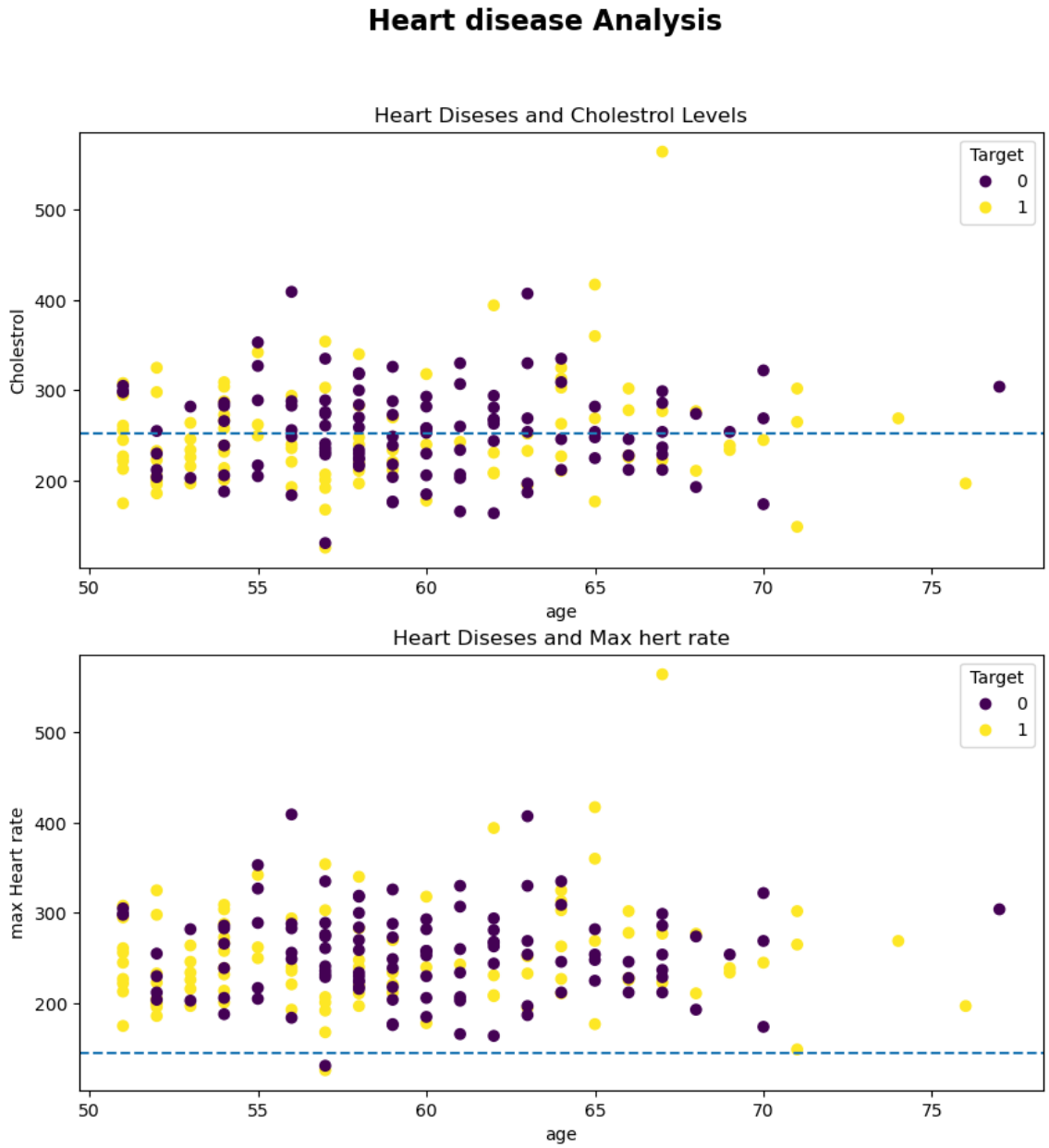
```
over_50.head()
```

Out[56]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	targ
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	
5	57	1	0	140	192	0	1	148	0	0.4	1	0	1	
6	56	0	1	140	294	0	0	153	0	1.3	1	0	2	

In [65]:

```
#subplots of chol age thalach
fig, (ax0, ax1) = plt.subplots(nrows=2,
                               ncols=1,
                               figsize=(10,10))
#add data to ax0
scatter= ax0.scatter(x=over_50["age"],
                     y=over_50["chol"],
                     c=over_50["target"]);
#custmise the plot
ax0.set(title="Heart Diseses and Cholestrol Levels",
      xlabel="age",
      ylabel="Cholestrol");
#add a legend
ax0.legend(*scatter.legend_elements(), title="Target");
#add a horizontol line
ax0.axhline(over_50["chol"].mean(),
          linestyle='--');
#add data to ax1
scatter= ax1.scatter(x=over_50["age"],
                     y=over_50["chol"],
                     c=over_50["target"]);
#custmise the plot
ax1.set(title="Heart Diseses and Max hert rate",
      xlabel="age",
      ylabel="max Heart rate");
#add a legend
ax1.legend(*scatter.legend_elements(), title="Target");
#add a horizontol line
ax1.axhline(y=over_50["thalach"].mean(),
          linestyle='--');
#add a titlte to fig
fig.suptitle("Heart disease Analysis", fontsize=16, fontweight="bold");
```



Customising Matplotlib plots and getting stylish

In [67]:

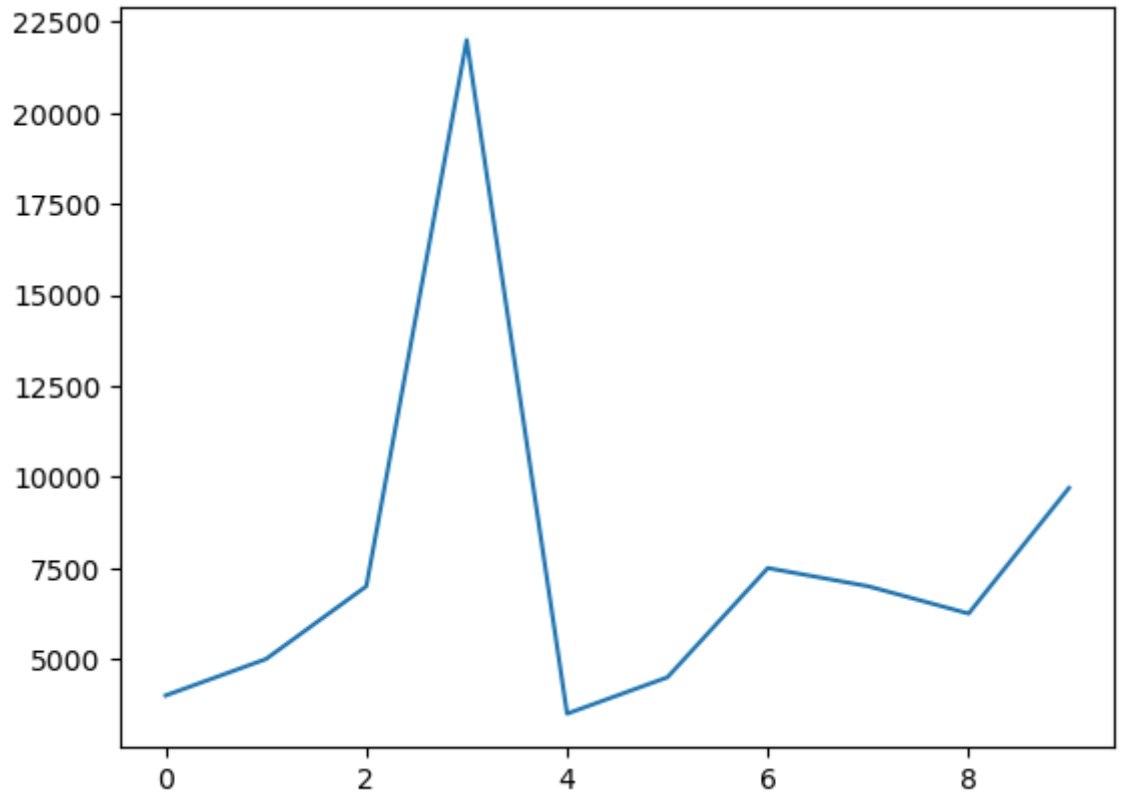
```
#see diffrent style
plt.style.available
```

Out[67]:

```
['Solarize_Light2',
 '_classic_test_patch',
 '_mpl-gallery',
 '_mpl-gallery-nogrid',
 'bmh',
 'classic',
 'dark_background',
 'fast',
 'fivethirtyeight',
 'ggplot',
 'grayscale',
 'seaborn-v0_8',
 'seaborn-v0_8-bright',
 'seaborn-v0_8-colorblind',
 'seaborn-v0_8-dark',
 'seaborn-v0_8-dark-palette',
 'seaborn-v0_8-darkgrid',
 'seaborn-v0_8-deep',
 'seaborn-v0_8-muted',
 'seaborn-v0_8-notebook',
 'seaborn-v0_8-paper',
 'seaborn-v0_8-pastel',
 'seaborn-v0_8-poster',
 'seaborn-v0_8-talk',
 'seaborn-v0_8-ticks',
 'seaborn-v0_8-white',
 'seaborn-v0_8-whitegrid',
 'tableau-colorblind10']
```

In [71]:

```
car_sales["Price"].plot();
```



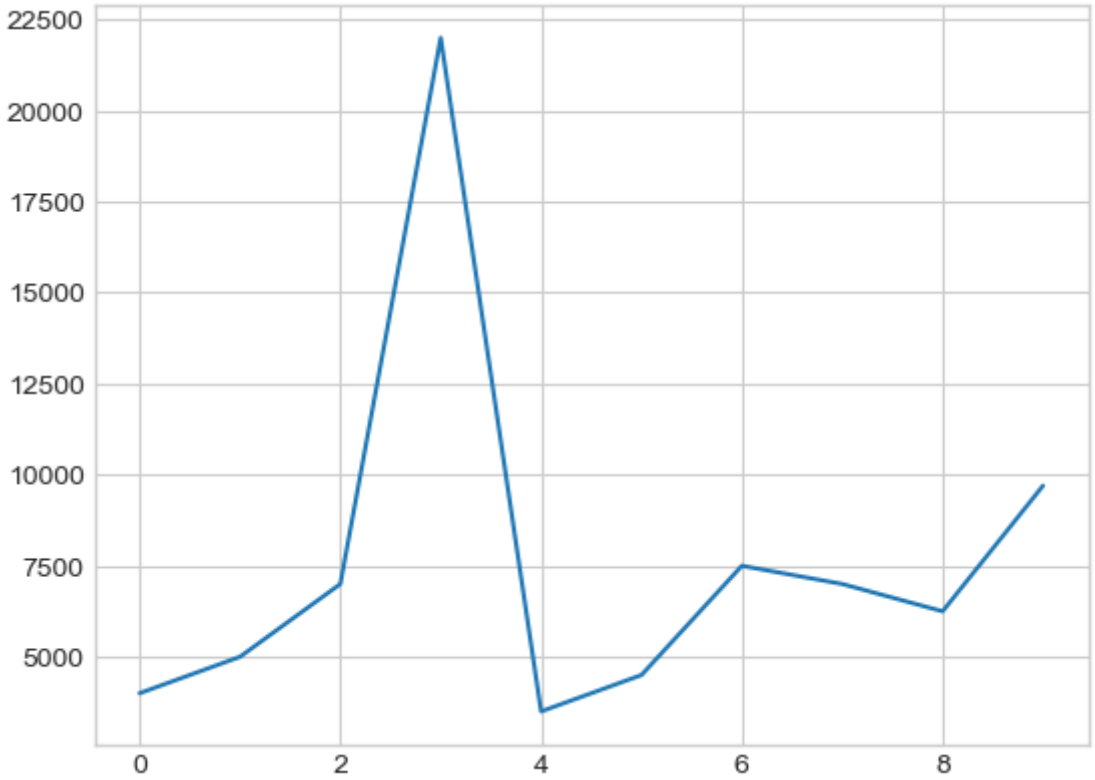
In [72]:

```
plt.style.use('seaborn-whitegrid')
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_16504\2414357448.py:1: Matplot
libDeprecationWarning: The seaborn styles shipped by Matplotlib are deprec
ated since 3.6, as they no longer correspond to the styles shipped by seab
orn. However, they will remain available as 'seaborn-v0_8-<style>'. Altern
atively, directly use the seaborn API instead.
  plt.style.use('seaborn-whitegrid')
```

In [73]:

```
car_sales["Price"].plot();
```



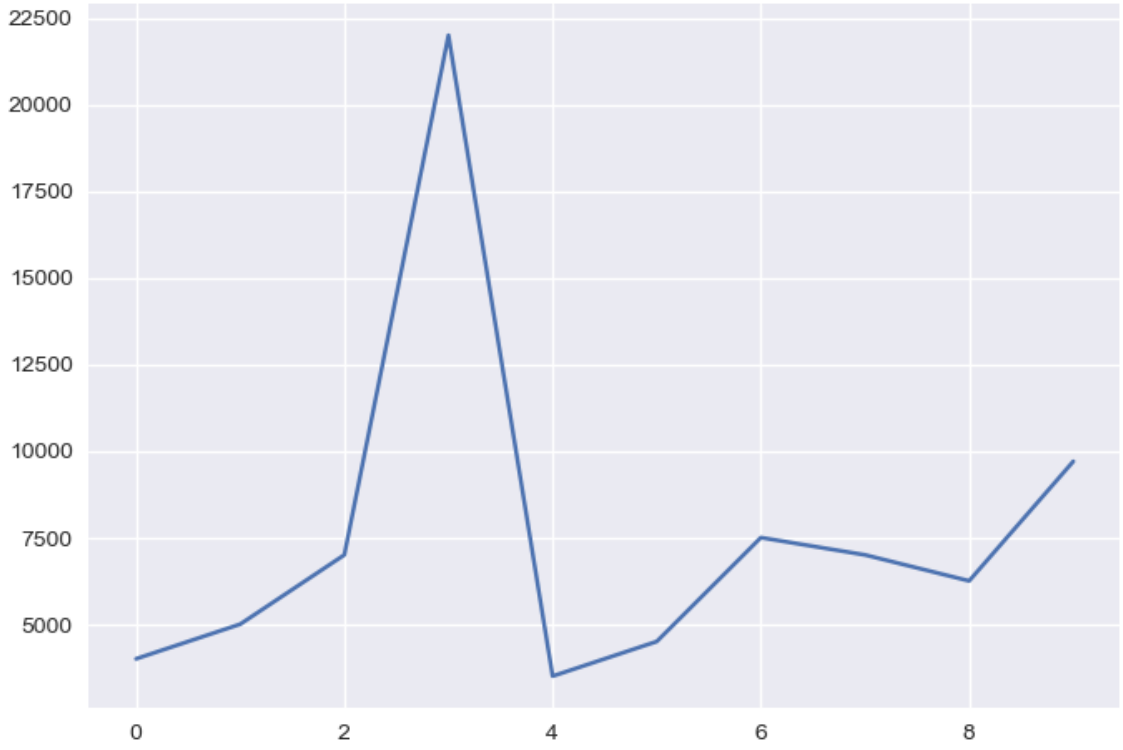
In [74]:

```
plt.style.use('seaborn')
```

```
C:\Users\alokr\AppData\Local\Temp\ipykernel_16504\240305066.py:1: Matplotl
ibDeprecationWarning: The seaborn styles shipped by Matplotlib are depreca
ted since 3.6, as they no longer correspond to the styles shipped by seabo
rn. However, they will remain available as 'seaborn-v0_8-<style>'. Alterna
tively, directly use the seaborn API instead.
  plt.style.use('seaborn')
```

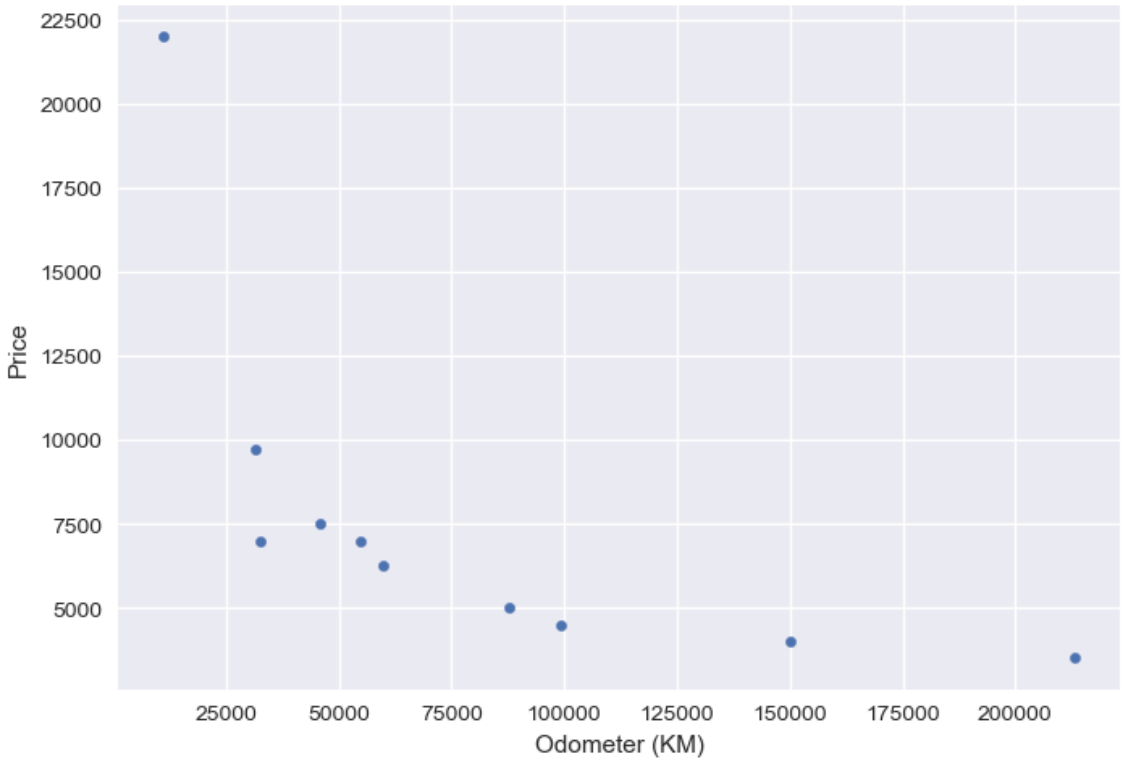
In [75]:

```
car_sales["Price"].plot();
```



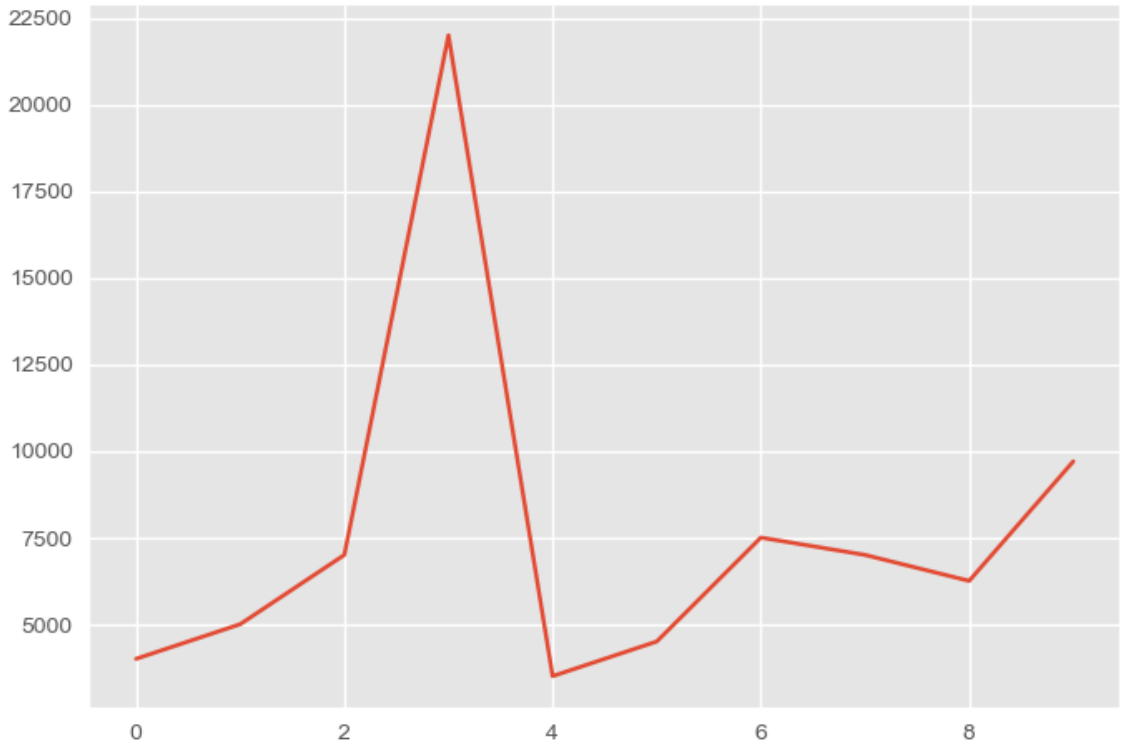
In [76]:

```
car_sales.plot(x="Odometer (KM)", y="Price", kind="scatter");
```



In [77]:

```
plt.style.use('ggplot')
car_sales["Price"].plot();
```



In [81]:

```
#create some data
x= np.random.randn(10, 4)
x
```

Out[81]:

```
array([[ 1.12391938, -0.93327806, -1.70881754, -0.26147393],
       [-0.05770042, -1.07791012,  1.61277206,  1.15115298],
       [ 0.19976328,  1.51065785,  2.32752331, -0.18051452],
       [-0.91776801, -0.76344029, -0.46107546,  0.30155895],
       [ 0.17914824, -0.62835752, -0.54401529,  0.67517527],
       [-0.005235  , -2.31139737, -0.83476789,  1.67483434],
       [-1.01079745, -0.77108814, -1.18840391, -0.49484966],
       [ 0.07877555, -1.47507239,  0.50620294, -0.78832966],
       [ 0.82473068,  0.90201369, -0.12642325,  1.66519463],
       [-0.33611143,  1.1773238 , -1.48528246, -1.43581443]])
```

In [83]:

```
df = pd.DataFrame(x, columns=['a', 'b', 'c', 'd'])
df
```

Out[83]:

	a	b	c	d
0	1.123919	-0.933278	-1.708818	-0.261474
1	-0.057700	-1.077910	1.612772	1.151153
2	0.199763	1.510658	2.327523	-0.180515
3	-0.917768	-0.763440	-0.461075	0.301559
4	0.179148	-0.628358	-0.544015	0.675175
5	-0.005235	-2.311397	-0.834768	1.674834
6	-1.010797	-0.771088	-1.188404	-0.494850
7	0.078776	-1.475072	0.506203	-0.788330
8	0.824731	0.902014	-0.126423	1.665195
9	-0.336111	1.177324	-1.485282	-1.435814

In [86]:

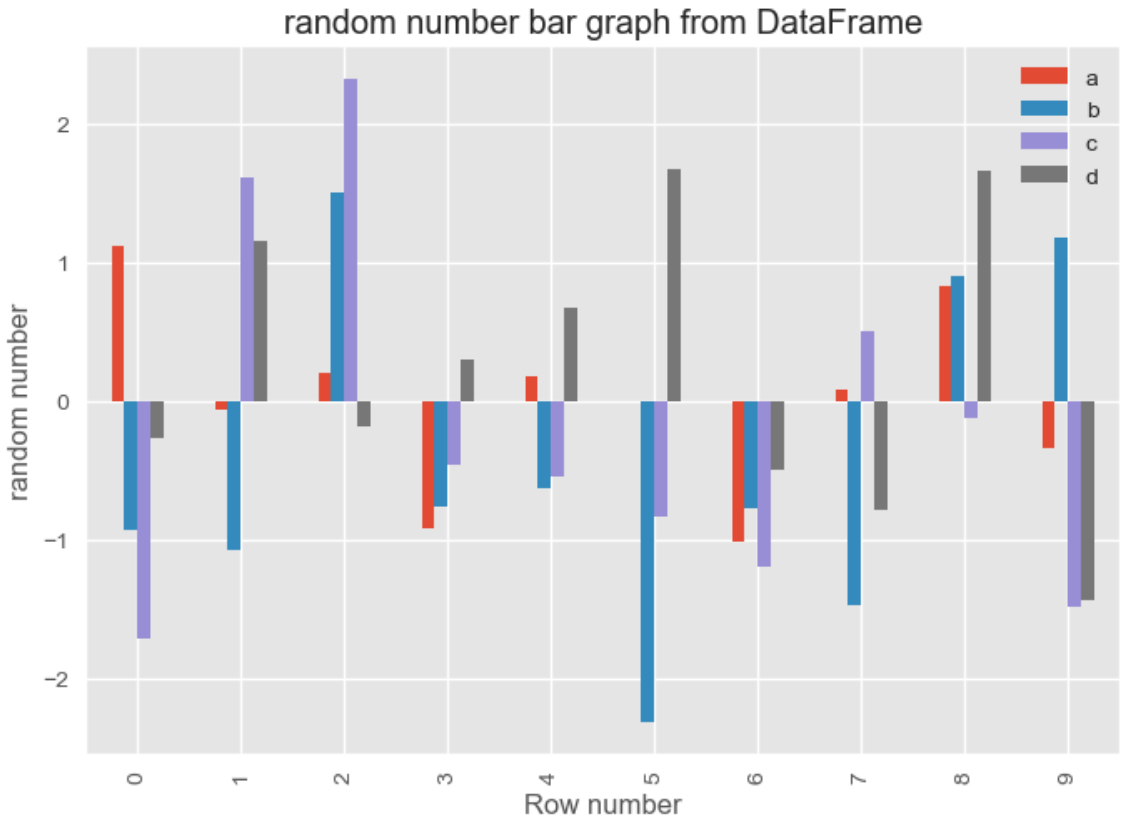
```
ax= df.plot(kind='bar')
type(ax)
```

Out[86]:

```
matplotlib.axes._subplots.AxesSubplot
```

In [87]:

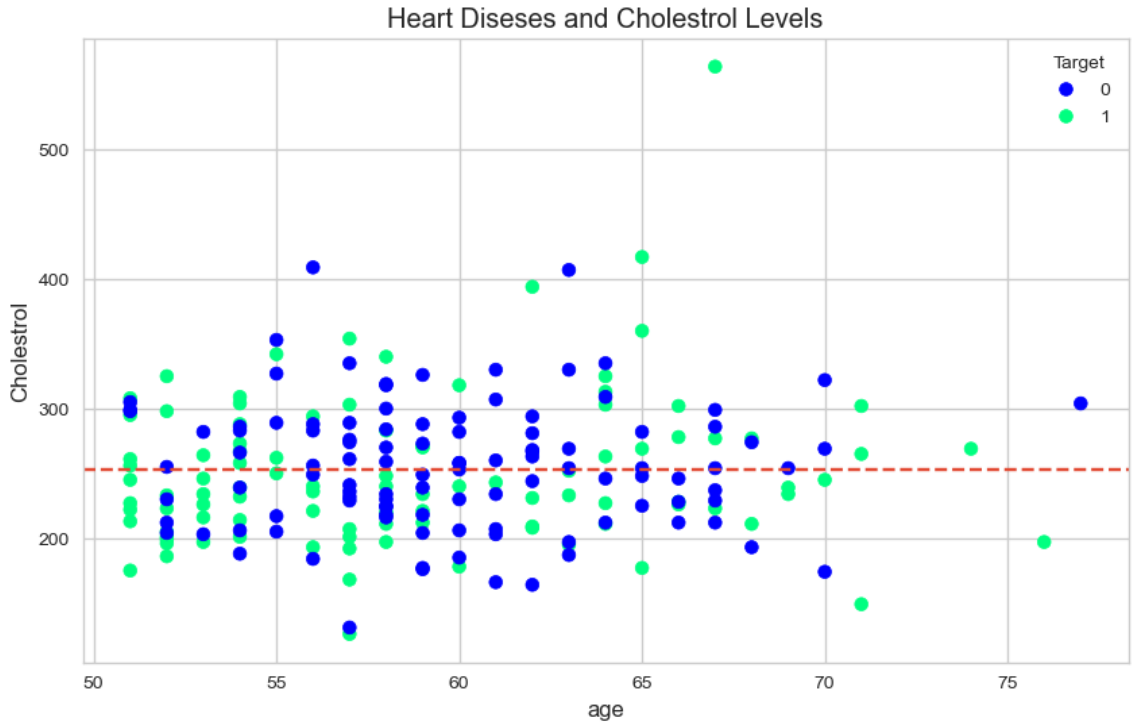
```
#customise our plot with the set() method
ax= df.plot(kind='bar')
#add some lebels
ax.set(title="random number bar graph from DataFrame",
       xlabel="Row number",
       ylabel="random number")
#Make the legend visible
ax.legend().set_visible(True)
```



In [89]:

```
#set the style
plt.style.use('seaborn-whitegrid')
#oo method from sracth
fig, ax = plt.subplots(figsize=(10, 6))
#plot the data
scatter= ax.scatter(x=over_50["age"],
                    y=over_50["chol"],
                    c=over_50["target"],
                    cmap="winter"); #this change the color schme
#custmise the plot
ax.set(title="Heart Diseses and Cholestrol Levels",
       xlabel="age",
       ylabel="Cholestrol");
#add a legend
ax.legend(*scatter.legend_elements(), title="Target");
#add a horizontol line
ax.axhline(over_50["chol"].mean(),
           linestyle='--');
```

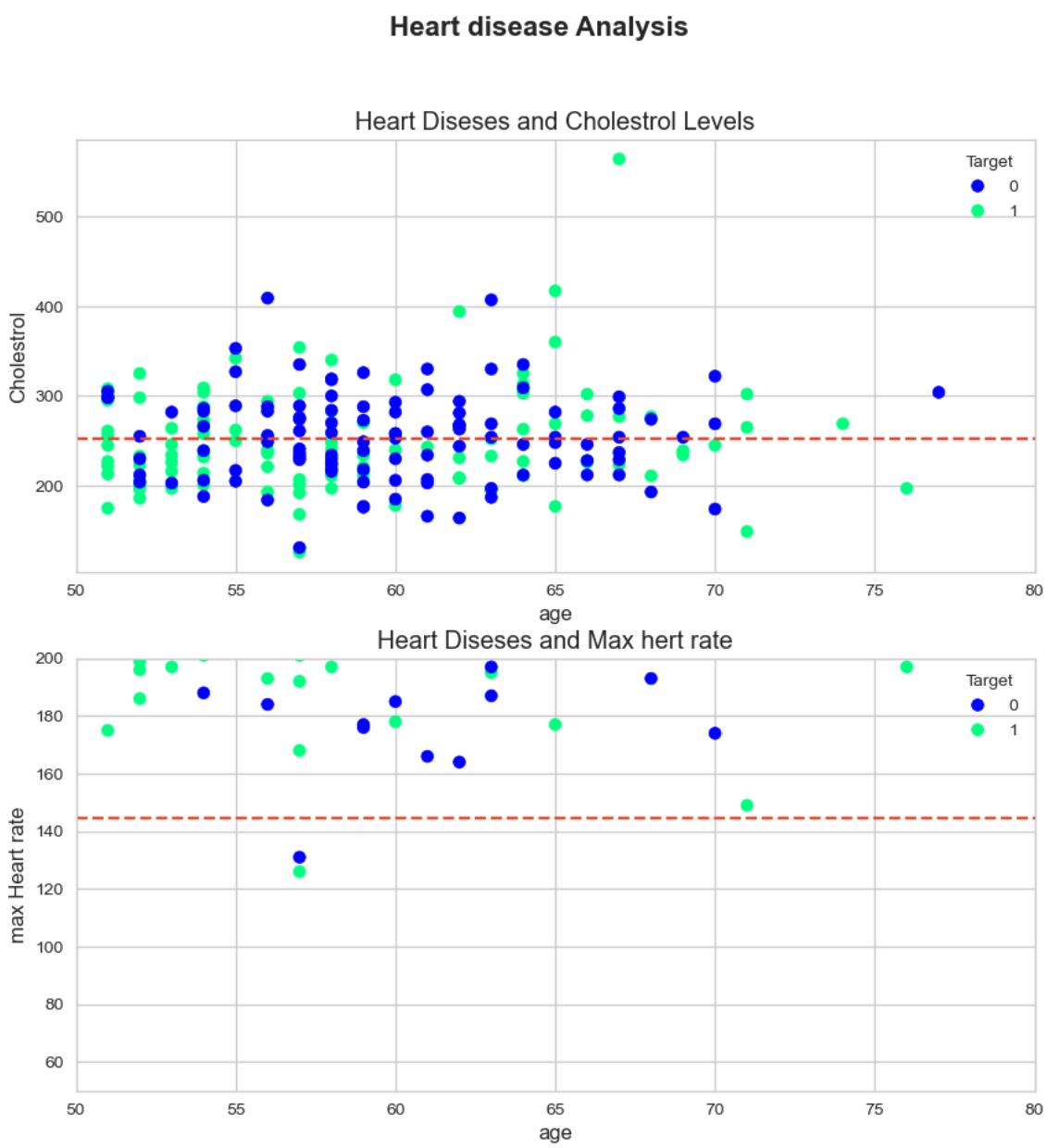
```
C:\Users\alokr\AppData\Local\Temp\ipykernel_16504\1599980799.py:2: Matplot
libDeprecationWarning: The seaborn styles shipped by Matplotlib are deprec
ated since 3.6, as they no longer correspond to the styles shipped by seab
orn. However, they will remain available as 'seaborn-v0_8-<style>'. Altern
atively, directly use the seaborn API instead.
  plt.style.use('seaborn-whitegrid')
```



In [98]:

```
#customising the y and x axis limititatons

#subplots of chol age thalach
fig, (ax0, ax1) = plt.subplots(nrows=2,
                               ncols=1,
                               figsize=(10,10))
#add data to ax0
scatter= ax0.scatter(x=over_50["age"],
                     y=over_50["chol"],
                     c=over_50["target"],
                      cmap="winter");
#custmise the plot
ax0.set(title="Heart Diseses and Cholestrol Levels",
      xlabel="age",
      ylabel="Cholestrol");
#change the x axis limits
ax0.set_xlim([50, 80])
#add a legend
ax0.legend(*scatter.legend_elements(), title="Target");
#add a horizontol line
ax0.axhline(over_50["chol"].mean(),
          linestyle='--');
#add data to ax1
scatter= ax1.scatter(x=over_50["age"],
                     y=over_50["chol"],
                     c=over_50["target"],
                      cmap="winter");
#custmise the plot
ax1.set(title="Heart Diseses and Max hert rate",
      xlabel="age",
      ylabel="max Heart rate");
#add a legend
ax1.legend(*scatter.legend_elements(), title="Target");
#change the x  and axis limits
ax1.set_xlim([50, 80])
ax1.set_ylim([50, 200])
#add a horizontol line
ax1.axhline(y=over_50["thalach"].mean(),
          linestyle='--');
#add a titlte to fig
fig.suptitle("Heart disease Analysis", fontsize=16, fontweight="bold");
```


Heart disease Analysis
Heart Diseses and Cholestrol Levels
Target
0
1
Cholestrol
500
400
300
200
50
55
60
65
70
75
80
age
Heart Diseses and Max hert rate
Target
0
1
max Heart rate
200
180
160
140
120
100
80
60
50
55
60
65
70
75
80
age

In [ ]: