In [2]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

In [3]:
```
df = sns.load_dataset('iris')
```

In [4]:
```
df
```

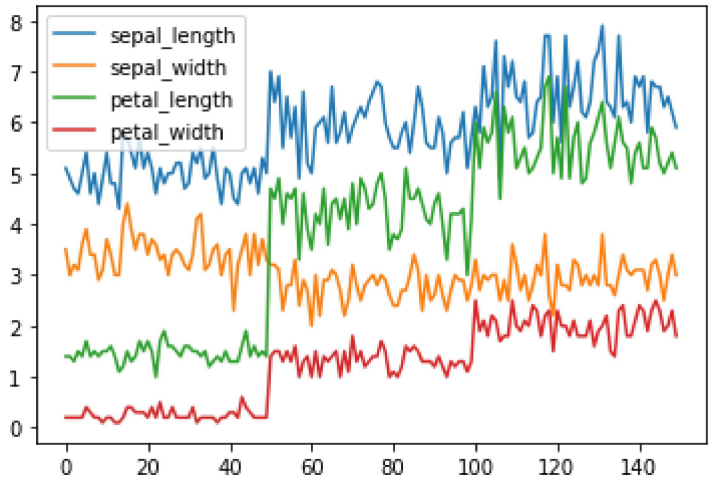
Out[4]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

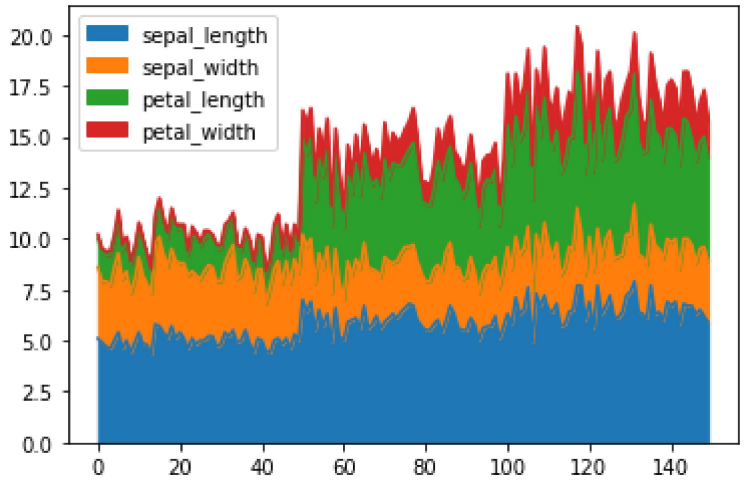
In [5]:
```
df.plot()  #describe function like for numerical columns
# by default we get a line graph
```

Out[5]:
```
<AxesSubplot:>
```



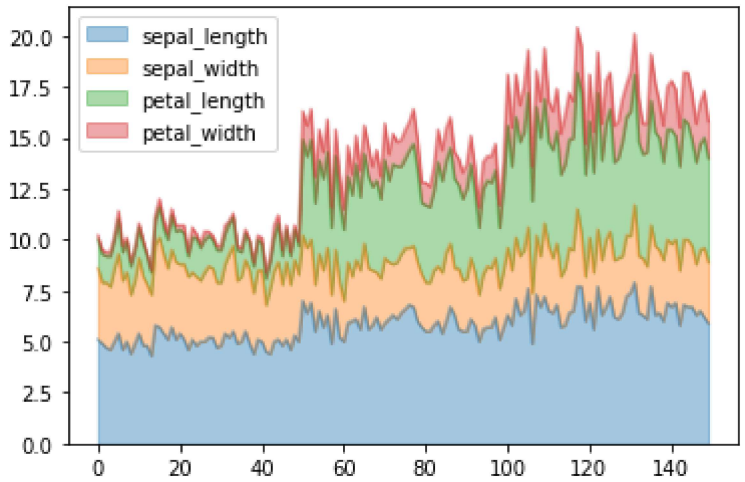
In [6]:
```
# to get different kind of graph
df.plot(kind = 'area')
```

Out[6]: <AxesSubplot:>



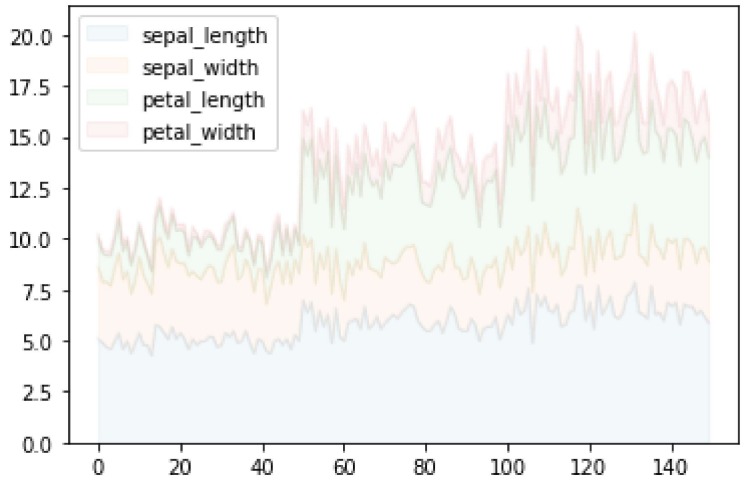
In [7]:
```
# to change the color or Intensity or darkness of color
df.plot(kind = 'area' ,alpha= 0.4)
```

Out[7]: <AxesSubplot:>



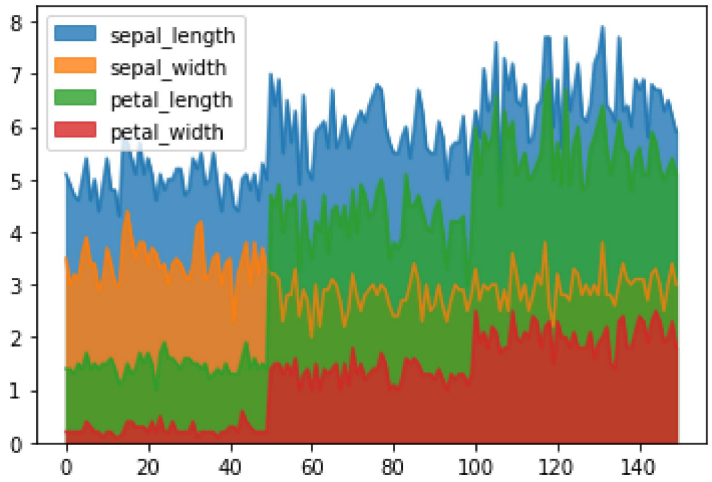
In [8]:
```
df.plot(kind = 'area' ,alpha= 0.05)
```

Out[8]: <AxesSubplot:>



In [9]:
```
# to make it not completely overlap
df.plot(kind = 'area' ,alpha= 0.8 , stacked = False)
```

Out[9]: <AxesSubplot:>



In [10]:
```
df
```

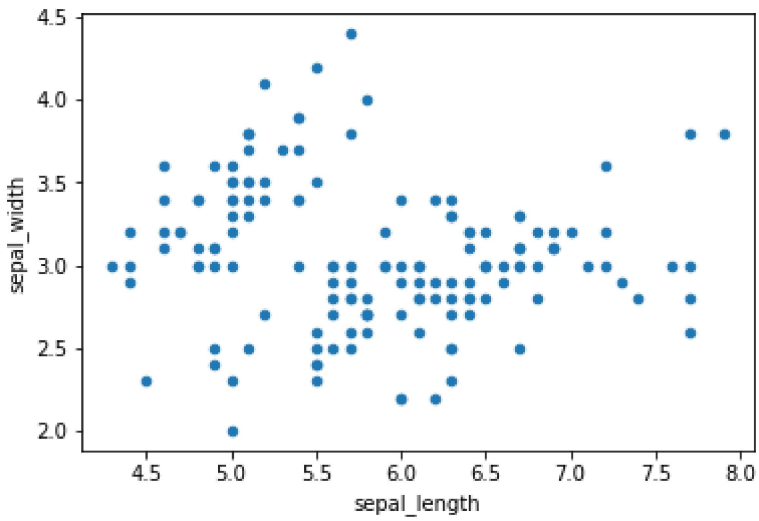
Out[10]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

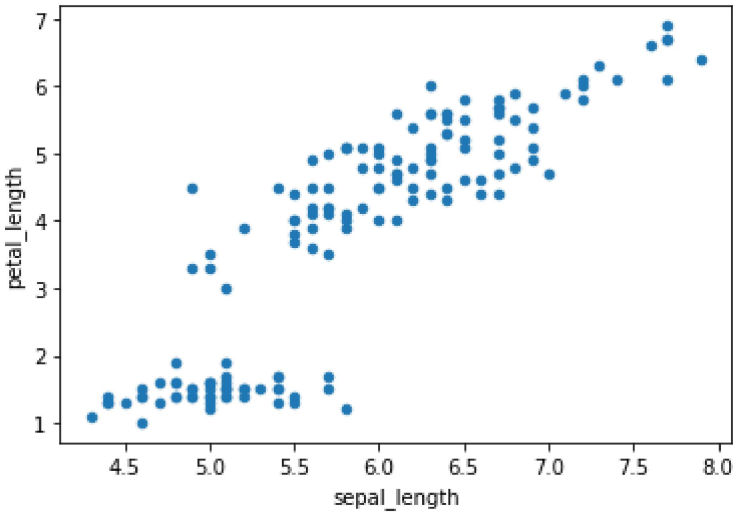
In [11]:
```
# say we want to compare some relations :
df.plot.scatter(x='sepal_length' , y ='sepal_width')
```

Out[11]:
```
<AxesSubplot:xlabel='sepal_length', ylabel='sepal_width'>
```



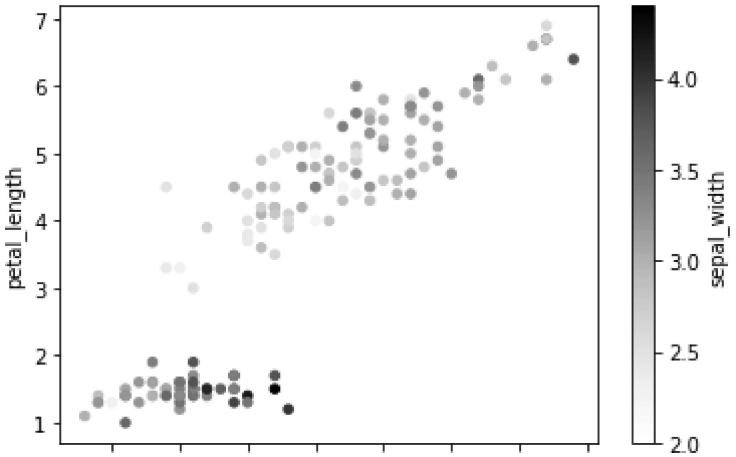
In [12]:
```
df.plot.scatter(x='sepal_length' , y ='petal_length')
```

Out[12]:
```
<AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>
```



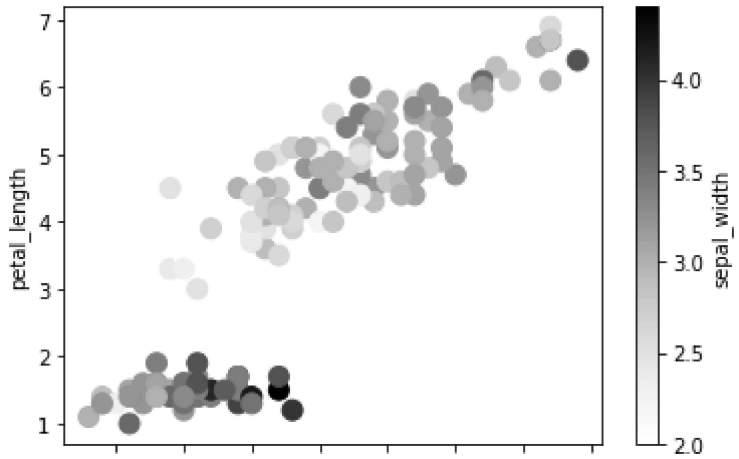
In [13]:
```
# to find relation in more depth with color or control
df.plot.scatter(x='sepal_length' , y ='petal_length' , c = 'sepal_width')


# sepal_width represent intensity
```

Out[13]:
```
<AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>
```



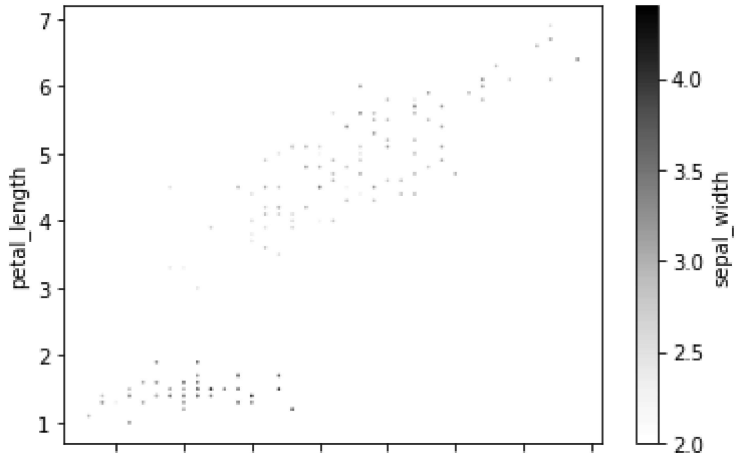
In [14]:
```
# for size of the dots
df.plot.scatter(x='sepal_length' , y ='petal_length' , c = 'sepal_width' , s =100
```

Out[14]: <AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>



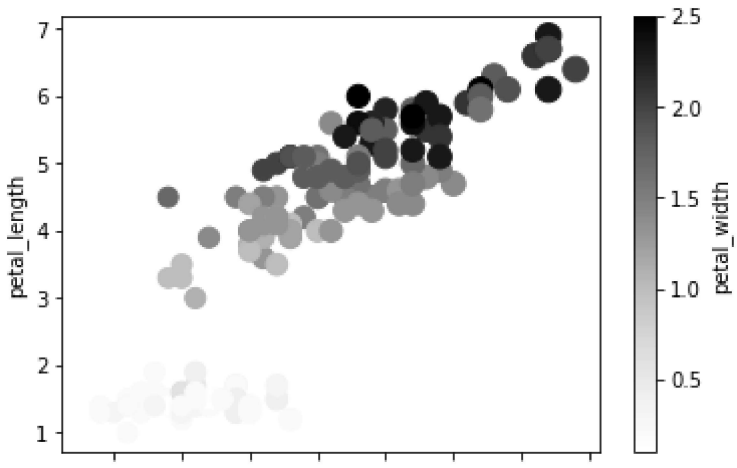
In [15]:
```
df.plot.scatter(x='sepal_length' , y ='petal_length' , c = 'sepal_width' , s = 0.
```

Out[15]: <AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>



In [16]:
```
# to make it dynamic
df.plot.scatter(x='sepal_length' , y ='petal_length' , c = 'petal_width', s = df[
```

Out[16]: <AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>



In [17]:
```
# hexbin plot here name of parameter change
df.plot.hexbin(x ='sepal_length' , y ='petal_length',gridsize=10,cmap="viridis")
```

Out[17]: <AxesSubplot:xlabel='sepal_length', ylabel='petal_length'>

In [18]:
```
# 3-D plot
```

In [19]:
```
from mpl_toolkits import mplot3d
```

In [20]:
```
# create a data
x = np.linspace(-1,6 , 30)
x
```

Out[20]:
```
array([-1.        , -0.75862069, -0.51724138, -0.27586207, -0.03448276,
        0.20689655,  0.44827586,  0.68965517,  0.93103448,  1.17241379,
        1.4137931 ,  1.65517241,  1.89655172,  2.13793103,  2.37931034,
        2.62068966,  2.86206897,  3.10344828,  3.34482759,  3.5862069 ,
        3.82758621,  4.06896552,  4.31034483,  4.55172414,  4.79310345,
        5.03448276,  5.27586207,  5.51724138,  5.75862069,  6.        ])
```

In [21]:
```
y = np.linspace(-1,6 , 30 )
y
```

Out[21]:
```
array([-1.        , -0.75862069, -0.51724138, -0.27586207, -0.03448276,
        0.20689655,  0.44827586,  0.68965517,  0.93103448,  1.17241379,
        1.4137931 ,  1.65517241,  1.89655172,  2.13793103,  2.37931034,
        2.62068966,  2.86206897,  3.10344828,  3.34482759,  3.5862069 ,
        3.82758621,  4.06896552,  4.31034483,  4.55172414,  4.79310345,
        5.03448276,  5.27586207,  5.51724138,  5.75862069,  6.        ])
```

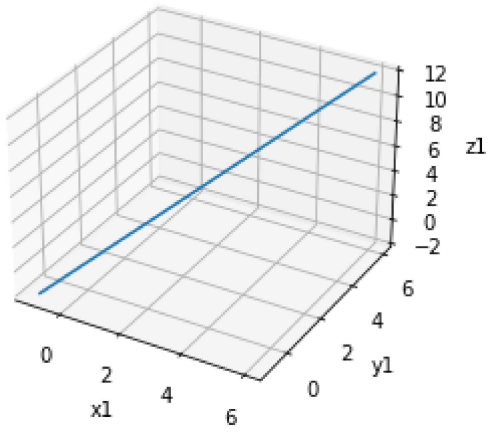
In [22]:
```
z = x+y
```

In [23]:
```
z
```

Out[23]:
```
array([-2.        , -1.51724138, -1.03448276, -0.55172414, -0.06896552,
        0.4137931 ,  0.89655172,  1.37931034,  1.86206897,  2.34482759,
        2.82758621,  3.31034483,  3.79310345,  4.27586207,  4.75862069,
        5.24137931,  5.72413793,  6.20689655,  6.68965517,  7.17241379,
        7.65517241,  8.13793103,  8.62068966,  9.10344828,  9.5862069 ,
       10.06896552, 10.55172414, 11.03448276, 11.51724138, 12.        ])
```

In [24]:
```
# first define dimn
ax = plt.axes(projection = '3d')
# now giving axis
ax.plot3D(x,y,z)
# defining axis's data labels
ax.set_xlabel('x1')
ax.set_ylabel('y1')
ax.set_zlabel('z1')
```

Out[24]: Text(0.5, 0, 'z1')



In [25]:
```
df
```

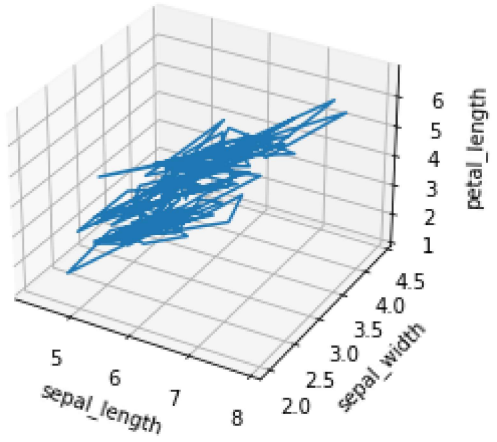
Out[25]:

	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
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

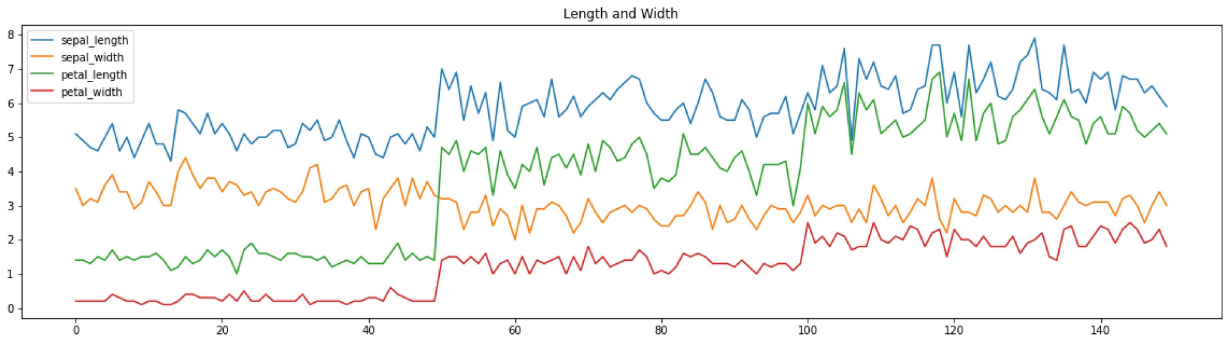
In [26]:
```
# plotting 3d graph for dataframe
ax = plt.axes(projection = '3d')
ax.plot3D(df['sepal_length'],df['sepal_width'],df['petal_length'])
ax.set_xlabel('sepal_length')
ax.set_ylabel('sepal_width')
ax.set_zlabel('petal_length')
```

Out[26]:
```
Text(0.5, 0, 'petal_length')
```



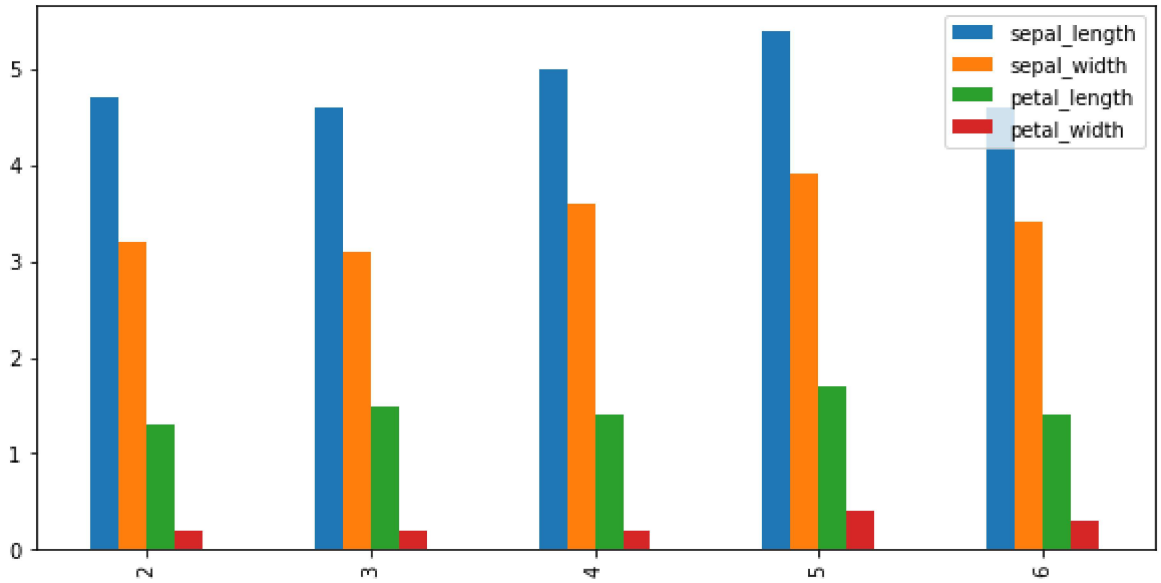
In [27]:
```
# to increase the size of the graph and for title
df.plot(figsize= (20,5) , title ='Length and Width')
```

Out[27]:
```
<AxesSubplot:title={'center':'Length and Width'}>
```



In [28]:
```
# Bar graph
df.iloc[2:7].plot(kind ='bar', figsize = (10,5))
```

Out[28]: <AxesSubplot:>



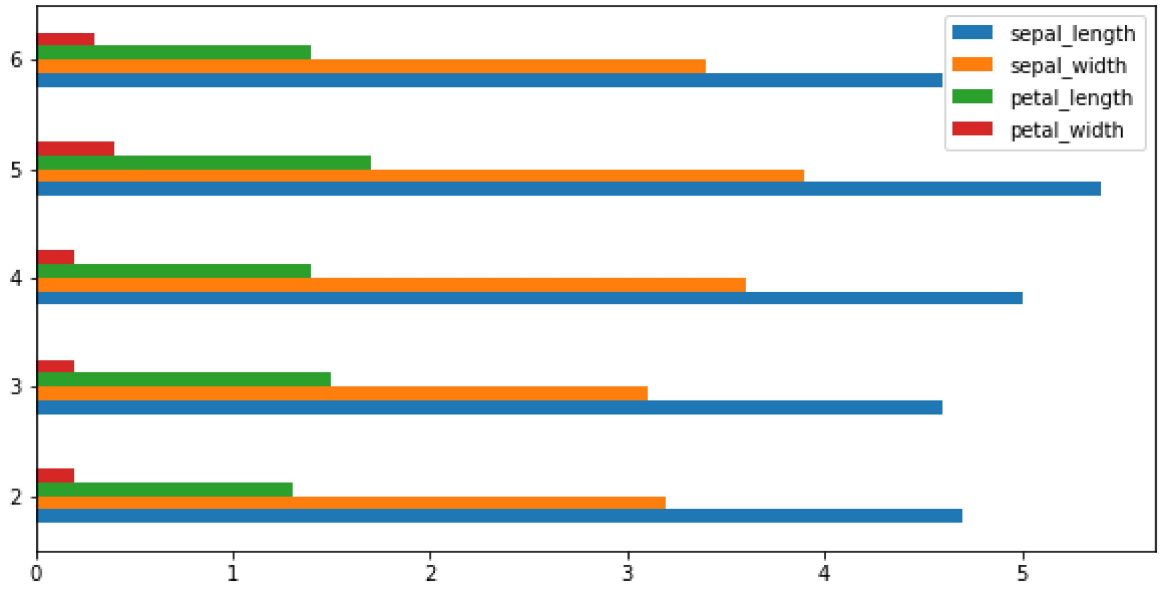
In [29]:
```
df.iloc[2:7]
```

Out[29]:

	sepal_length	sepal_width	petal_length	petal_width	species
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
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa

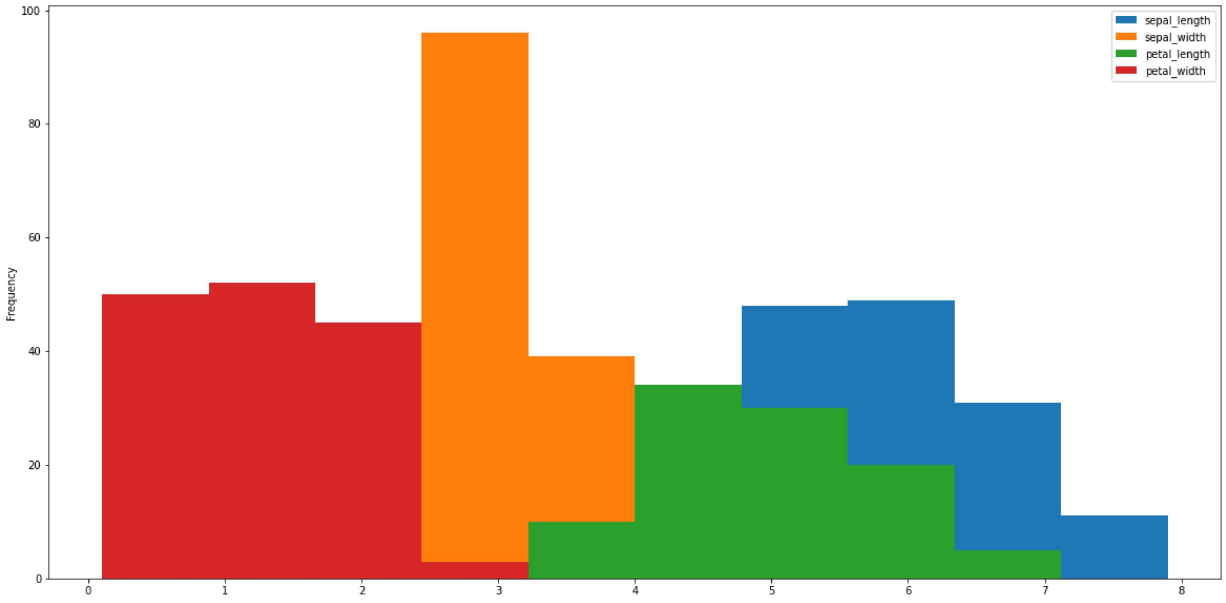
In [30]:
```
# horizontal bar
df.iloc[2:7].plot(kind ='barh', figsize = (10,5))
```

Out[30]: <AxesSubplot:>



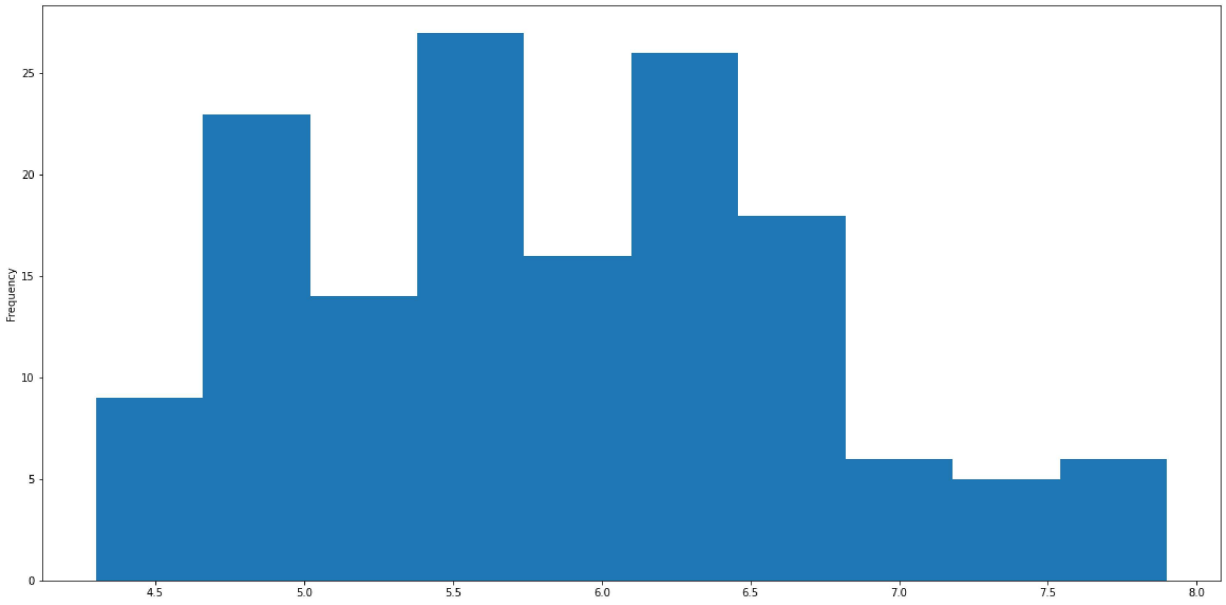
In [31]:
```
# histogram
df.plot(kind= 'hist' , figsize = (20,10))
```

Out[31]: <AxesSubplot:ylabel='Frequency'>



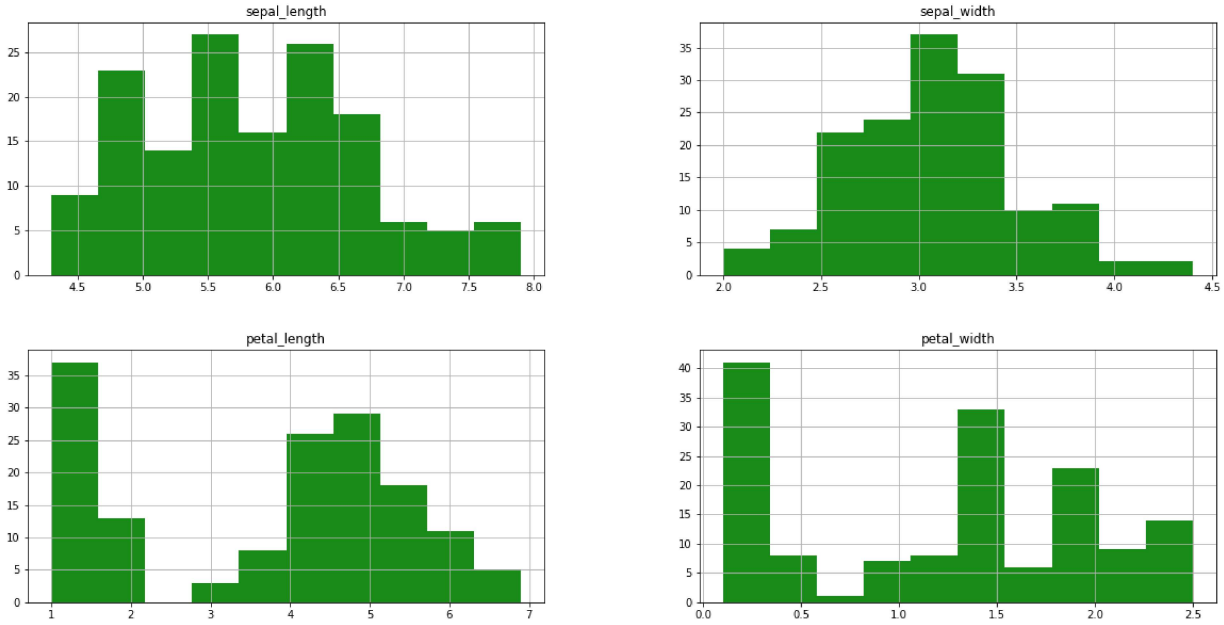
In [32]:
```
# for single column
df['sepal_length'].plot(kind = 'hist' , figsize = (20,10))
```

Out[32]: <AxesSubplot:ylabel='Frequency'>



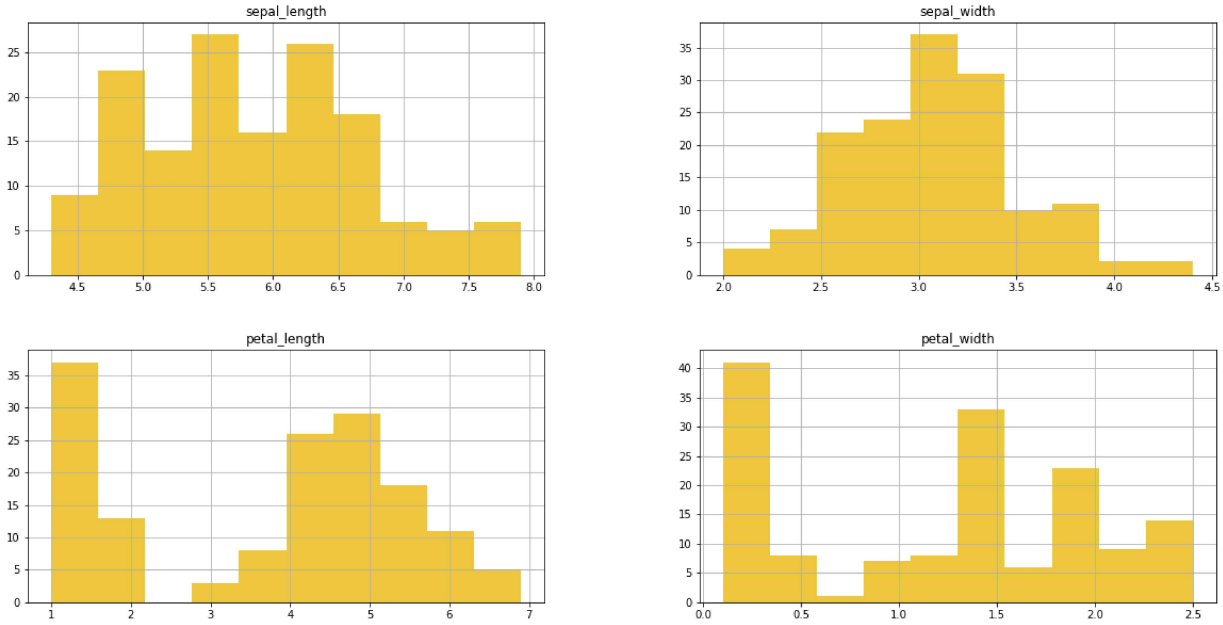
In [33]:
```
# hist function
# for color
df.hist(figsize=(20,10) , color = 'g', alpha= .9)
```

Out[33]:
```
array([[<AxesSubplot:title={'center':'sepal_length'}>,
        <AxesSubplot:title={'center':'sepal_width'}>],
       [<AxesSubplot:title={'center':'petal_length'}>,
        <AxesSubplot:title={'center':'petal_width'}>]], dtype=object)
```



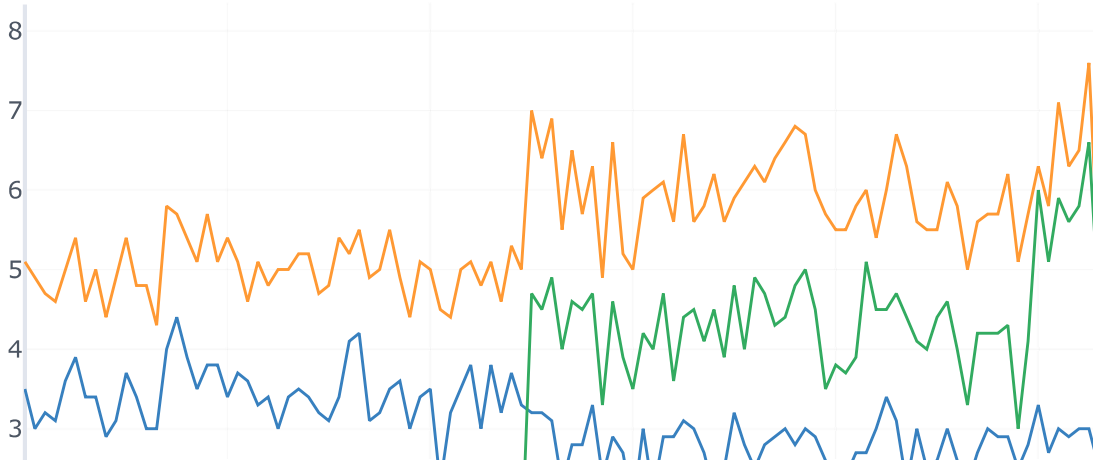
In [34]:
```
# changing color with hexadecimal code
df.hist(figsize=(20,10) , color = '#EDC128', alpha= .9)
```

Out[34]:
```
array([[<AxesSubplot:title={'center':'sepal_length'}>,
        <AxesSubplot:title={'center':'sepal_width'}>],
       [<AxesSubplot:title={'center':'petal_length'}>,
        <AxesSubplot:title={'center':'petal_width'}>]], dtype=object)
```

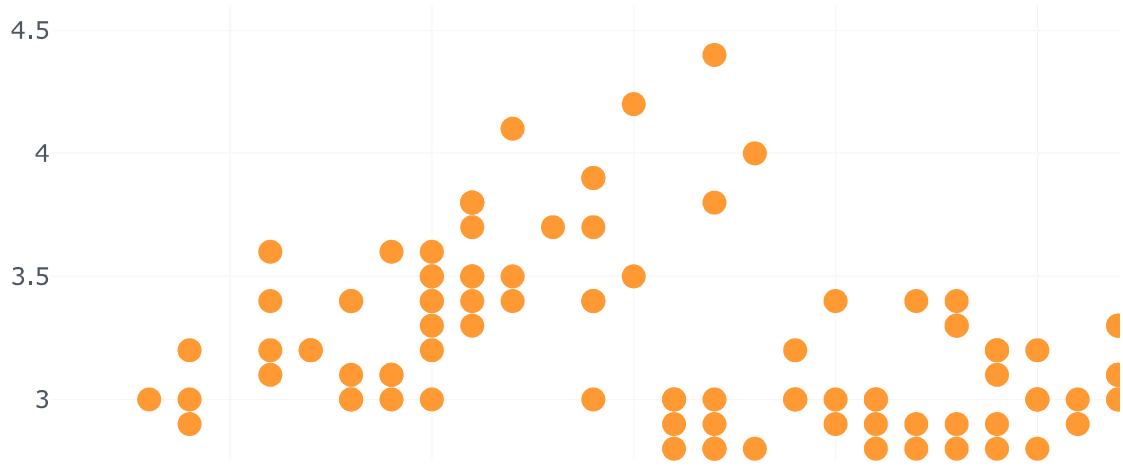


In [35]:
```
import cufflinks as cf
cf.go_offline()
```

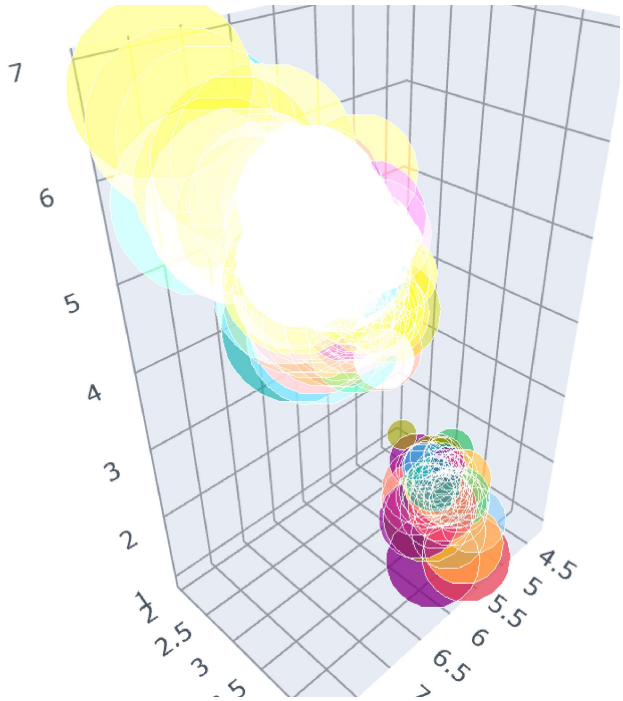
In [36]:
```
df.iplot()
```



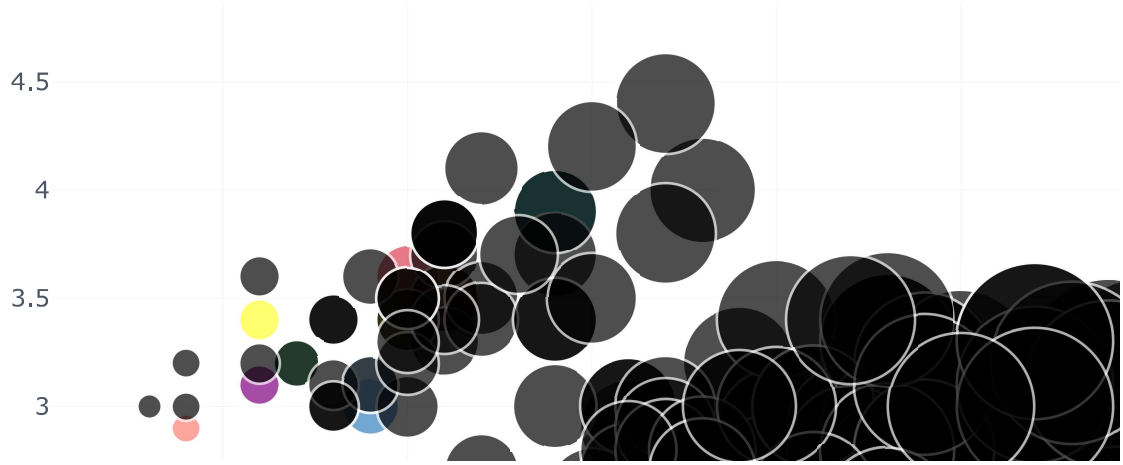
In [37]:
```
df.iplot(x = 'sepal_length' , y = 'sepal_width',kind = 'scatter' , mode = 'marker
# mode is for dots
```



In [38]:
```
df.iplot(x ='sepal_length' ,y ='sepal_width' ,z ='petal_length'  ,size = 'sepal_l
```



In [39]:
```
df.iplot(x ='sepal_length' ,y ='sepal_width' ,size = 'sepal_length' ,kind = 'bubb
```



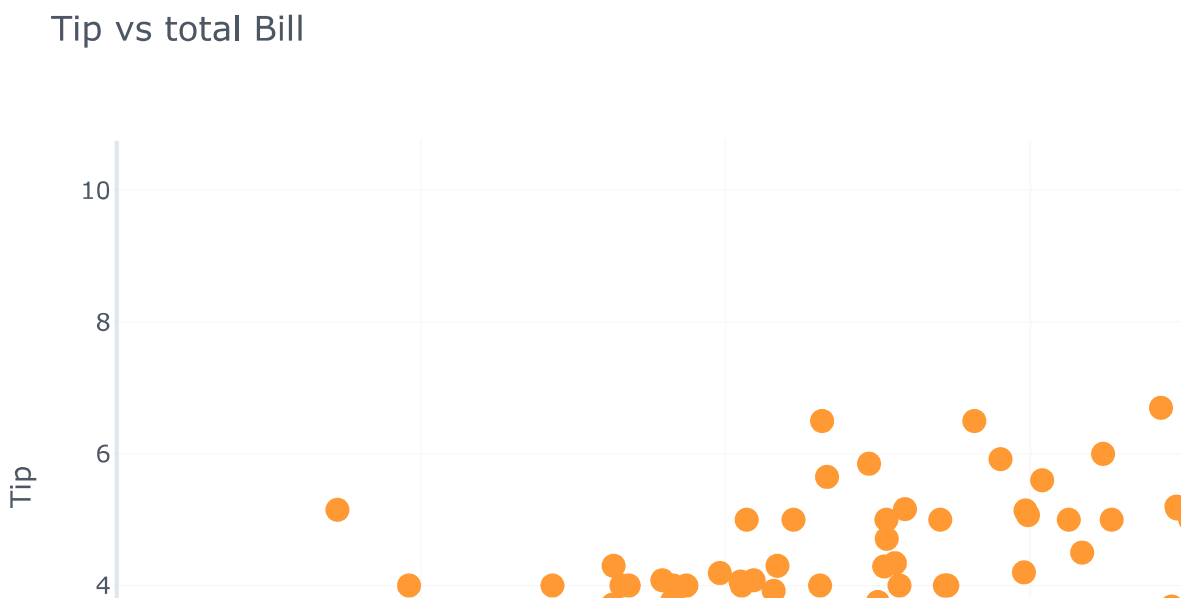
In [40]:
```
df1 = sns.load_dataset('tips')
```

In [41]:
```
df1
```

Out[41]:

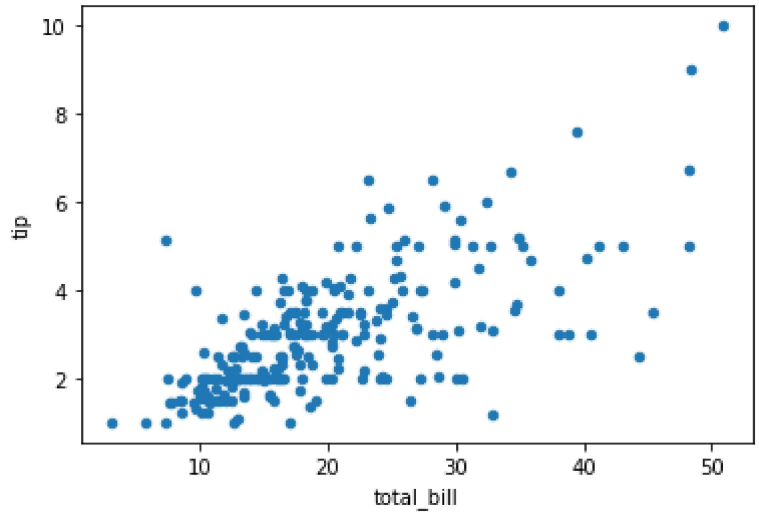
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4
...	...	...	...	...	...	...	...
239	29.03	5.92	Male	No	Sat	Dinner	3
240	27.18	2.00	Female	Yes	Sat	Dinner	2
241	22.67	2.00	Male	Yes	Sat	Dinner	2
242	17.82	1.75	Male	No	Sat	Dinner	2
243	18.78	3.00	Female	No	Thur	Dinner	2

In [42]:
```
df1.iplot(x = 'total_bill' , y = 'tip' , kind = 'scatter', mode = 'markers', xTit
```



In [43]:
```
df1.plot(x = 'total_bill' , y = 'tip' , kind = 'scatter')
```

Out[43]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```

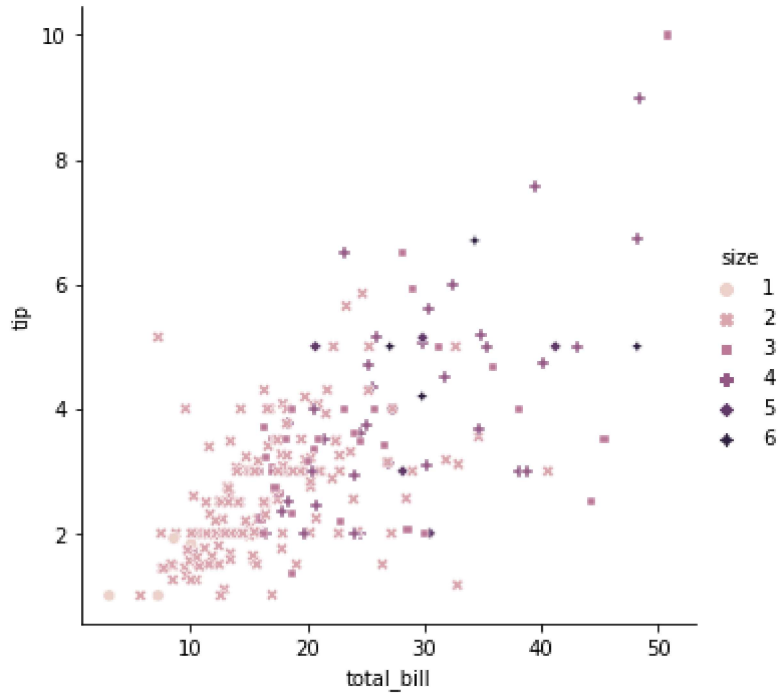


In [44]:
```
# using seaborn for plots:
```

In [45]:
```
# and using style , hue for grouping
```

In [46]:
```
sns.relplot(x = 'total_bill', y = 'tip' , data = df1 , hue = 'size',style = 'size
```

Out[46]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51b075b20>
```

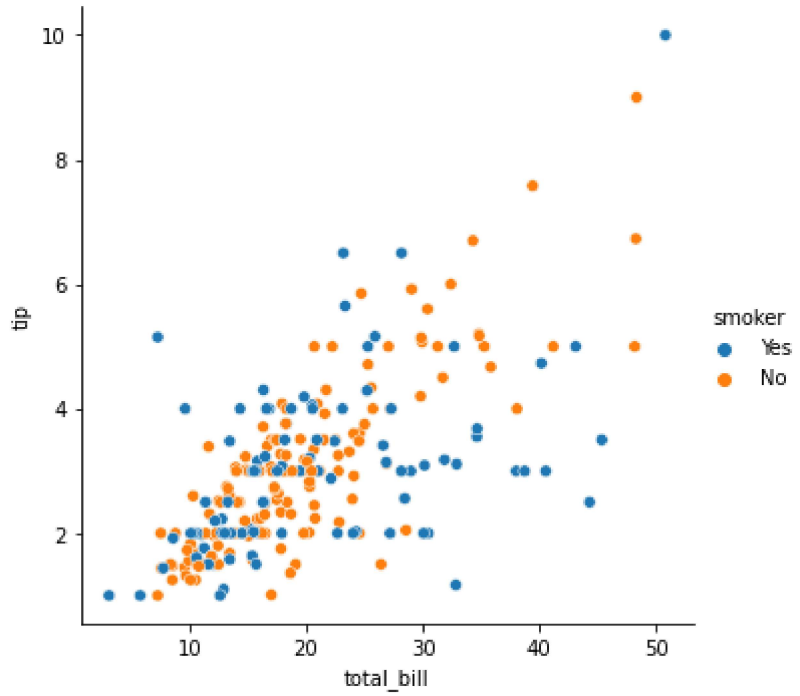


In [47]:
```
df1['smoker'].value_counts()
```

Out[47]:
```
No     151
Yes     93
Name: smoker, dtype: int64
```

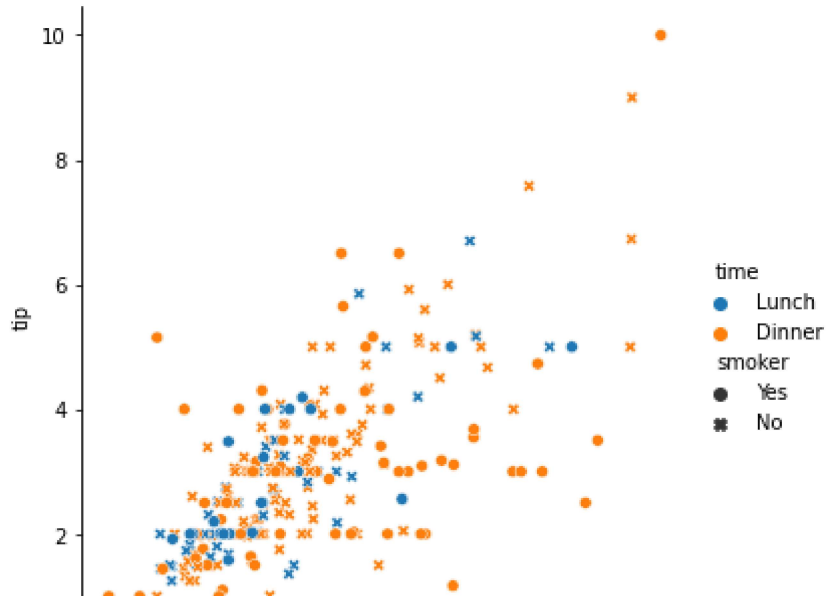
In [48]:
```
sns.relplot(x = 'total_bill', y = 'tip' , data = df1 , hue = 'smoker')
```

Out[48]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51b053df0>
```



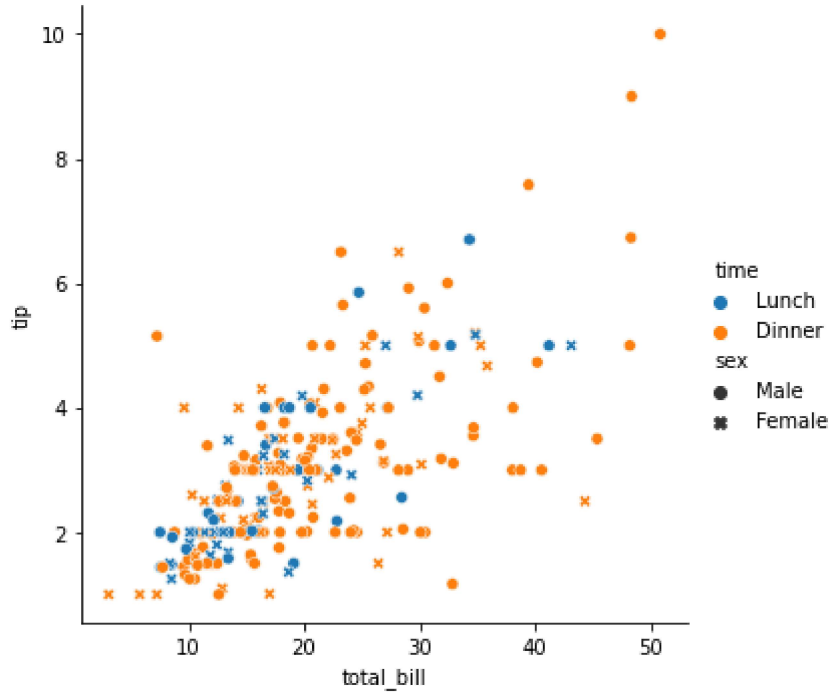
In [49]:
```
sns.relplot(x = 'total_bill', y = 'tip' , data = df1 , style = 'smoker', hue = 't
```

Out[49]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51b17b550>
```



In [50]:
```
sns.relplot(x = 'total_bill', y = 'tip' , data = df1 , style = 'sex', hue = 'time
```

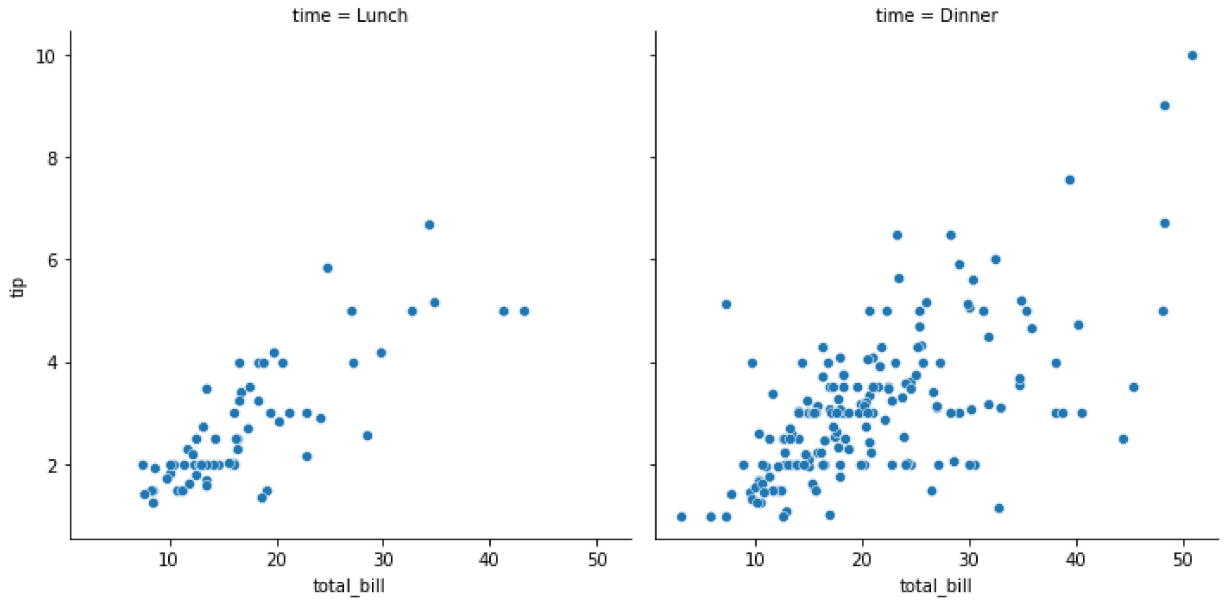
Out[50]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51b12e4c0>
```



In [51]:
```
# to seperate data
```

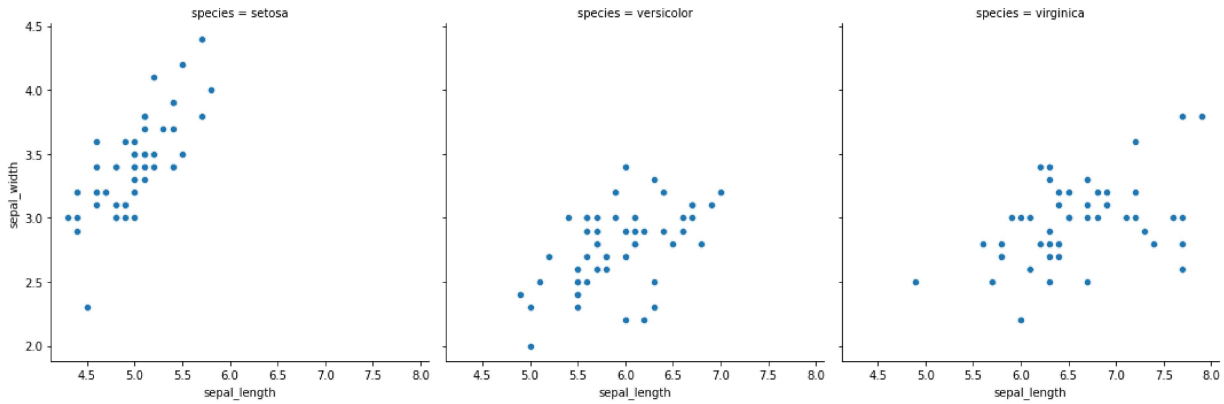
In [52]:
```
sns.relplot(x = 'total_bill' ,y = 'tip' , data = df1 , col ='time' )
```

Out[52]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51b141430>
```



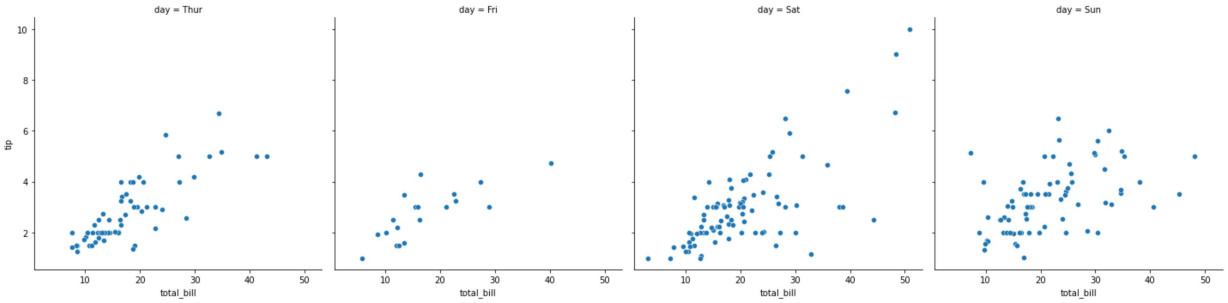
In [53]:
```
sns.relplot(x = 'sepal_length' ,y = 'sepal_width' , data = df , col ='species' )
```

Out[53]: <seaborn.axisgrid.FacetGrid at 0x1a51b2e81c0>



In [54]:
```
sns.relplot(x = 'total_bill' ,y = 'tip' , data = df1 , col ='day' )
```

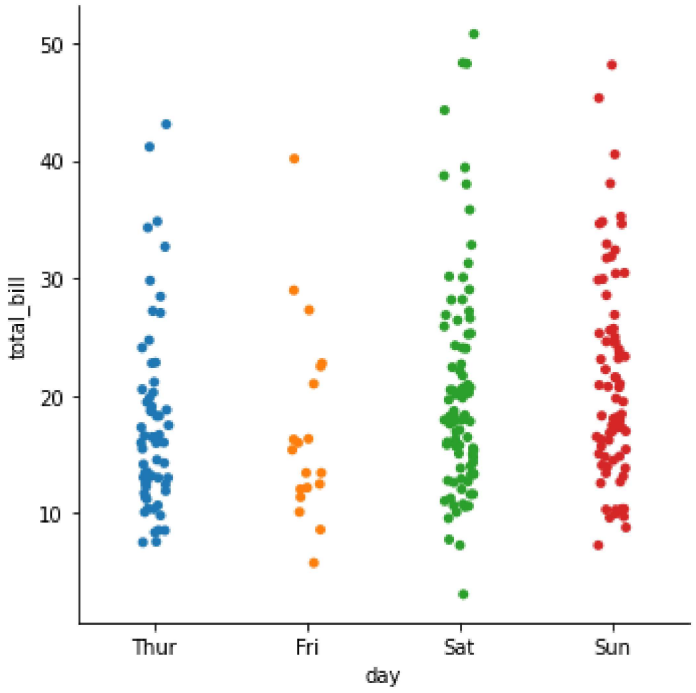
Out[54]: <seaborn.axisgrid.FacetGrid at 0x1a51b1dba60>



In [55]:
```
# catplot for categorical plot
```

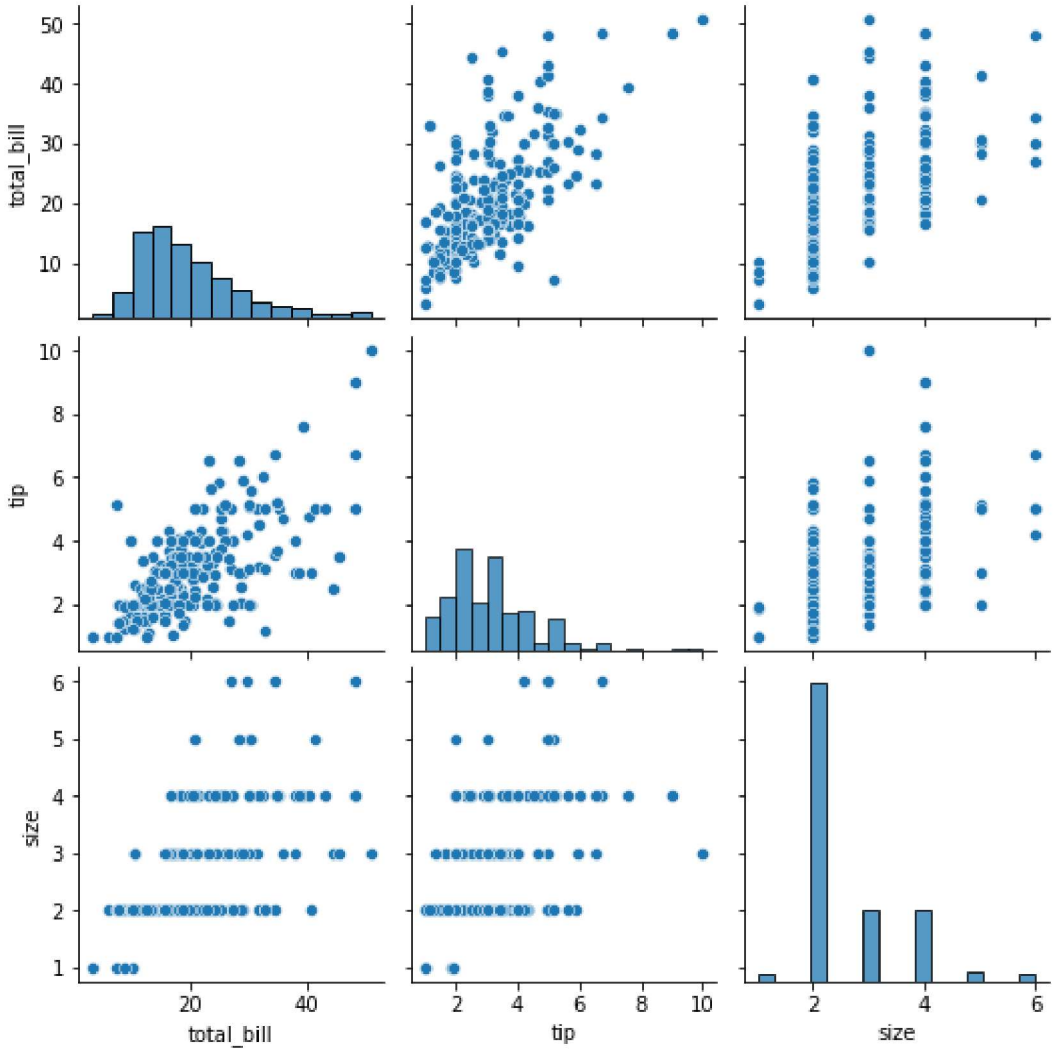
In [56]:
```
sns.catplot( x = 'day' , y = 'total_bill' , data = df1)
```

Out[56]:
```
<seaborn.axisgrid.FacetGrid at 0x1a51baa7f70>
```



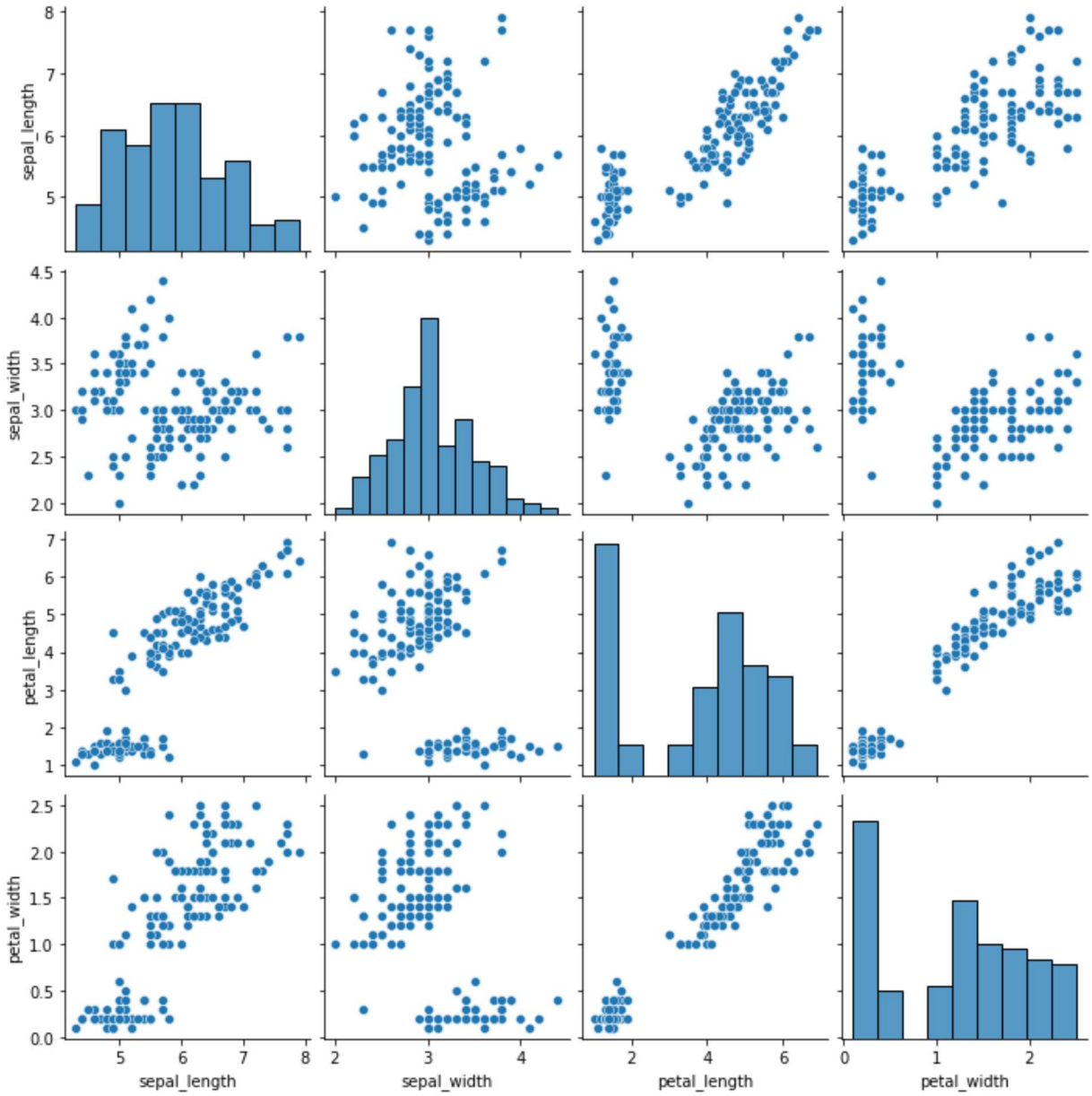
In [57]:
```
# pair plot for comparison of all  numerical data in df1
sns.pairplot(df1)
```

Out[57]: <seaborn.axisgrid.PairGrid at 0x1a51bd540d0>

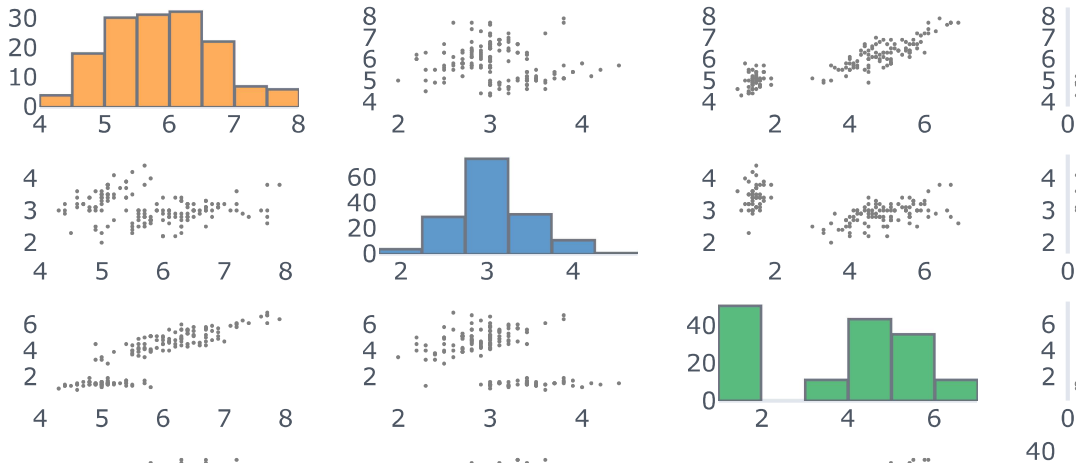


In [58]:
```
sns.pairplot(df)
```

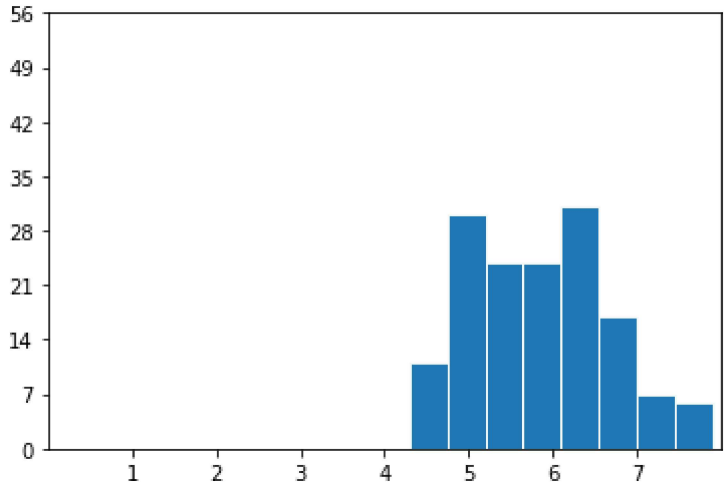
Out[58]:
```
<seaborn.axisgrid.PairGrid at 0x1a51bd4c310>
```



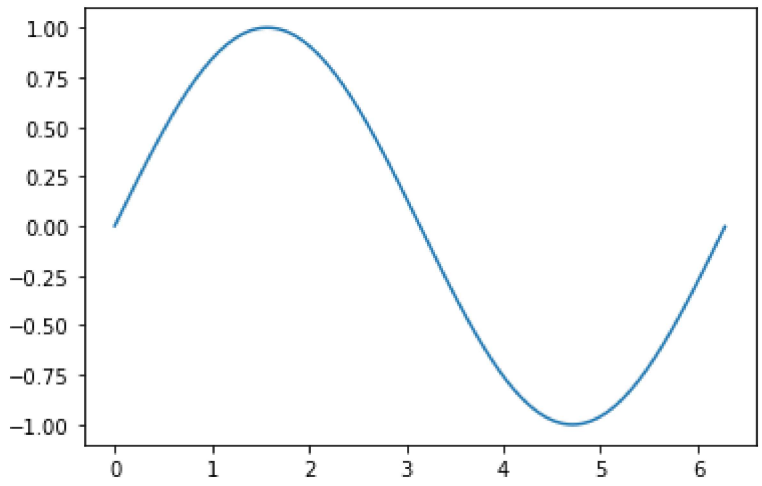
In [59]:
```
# in plotly cuuflinks if we want same thing
df.scatter_matrix()
```



In [60]:
```
import matplotlib.pyplot as plt
import numpy as np


# make data
np.random.seed(1)
x = 4 + np.random.normal(0, 1.5, 200)

# plot:
fig, ax = plt.subplots()

ax.hist(df['sepal_length'], bins=8, linewidth=0.9, edgecolor="white")

ax.set(xlim=(0, 8), xticks=np.arange(1, 8),
       ylim=(0, 56), yticks=np.linspace(0, 56, 9))

plt.show()
```

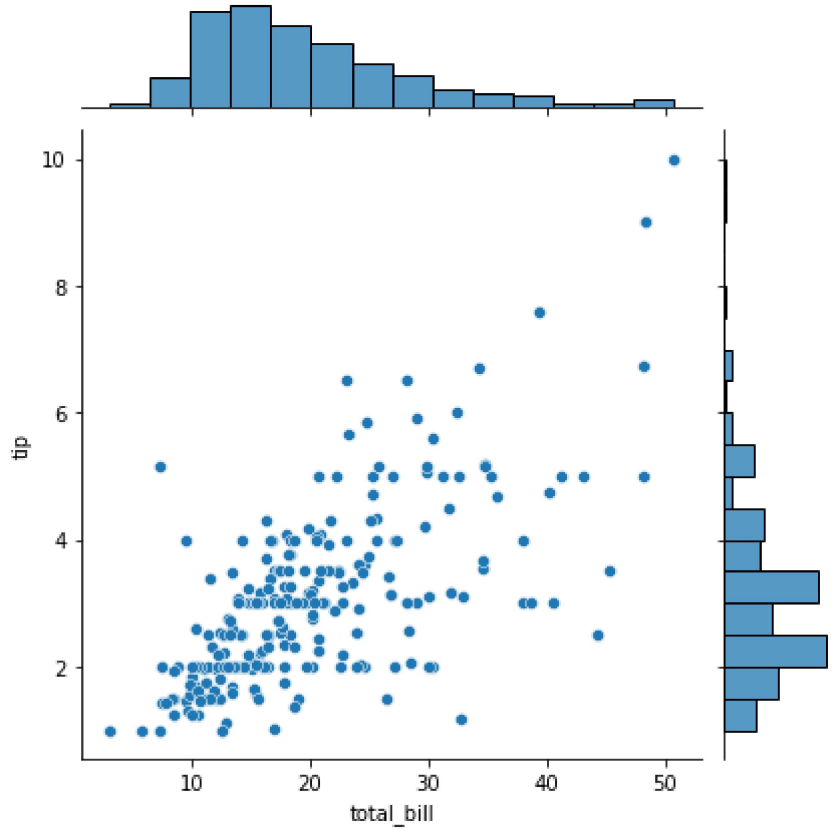


In [61]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.animation as animation

fig, ax = plt.subplots()

x = np.arange(0, 2*np.pi, 0.01)
line, = ax.plot(x, np.sin(x))


def animate(i):
    line.set_ydata(np.sin(x + i / 50))  # update the data.
    return line,


ani = animation.FuncAnimation(
    fig, animate, interval=20, blit=True, save_count=50)

# To save the animation, use e.g.
#
# ani.save("movie.mp4")
#
# or
#
# writer = animation.FFMpegWriter(
#     fps=15, metadata=dict(artist='Me'), bitrate=1800)
# ani.save("movie.mp4", writer=writer)

plt.show()
```



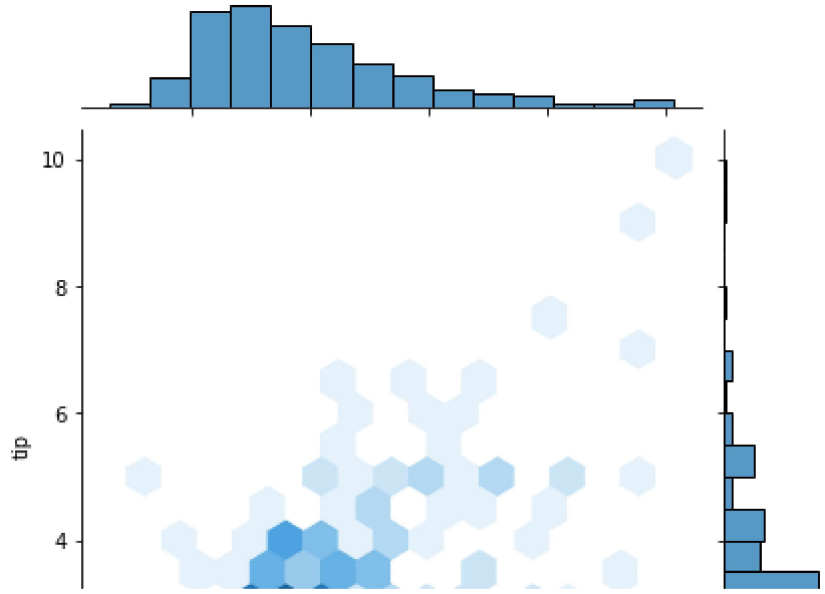
In [62]:
```
# combining two diff kind of plots
sns.jointplot(x =df1.total_bill , y = df1.tip )
```

Out[62]: <seaborn.axisgrid.JointGrid at 0x1a51e1f9ee0>



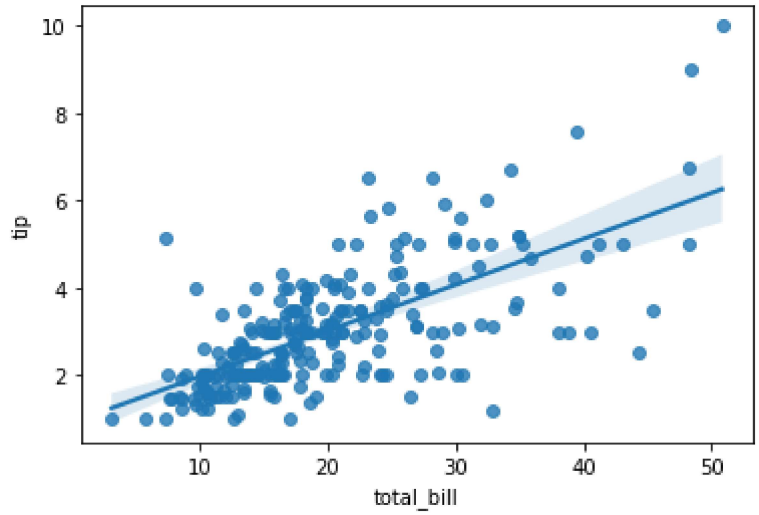
In [63]:
```
# changing kind
sns.jointplot(x =df1.total_bill , y = df1.tip,kind = 'hex' )
```

Out[63]:
```
<seaborn.axisgrid.JointGrid at 0x1a51e1d69d0>
```

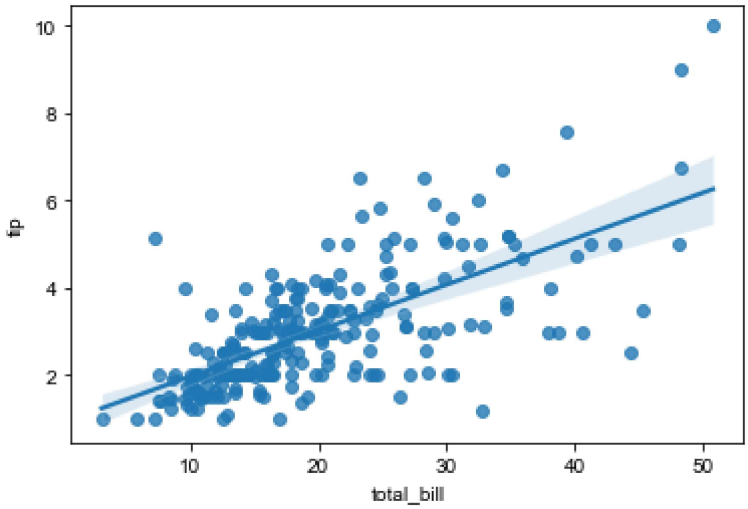


In [64]:
```
# regplot give regression data


sns.regplot(x = df1.total_bill , y = df1.tip)
```

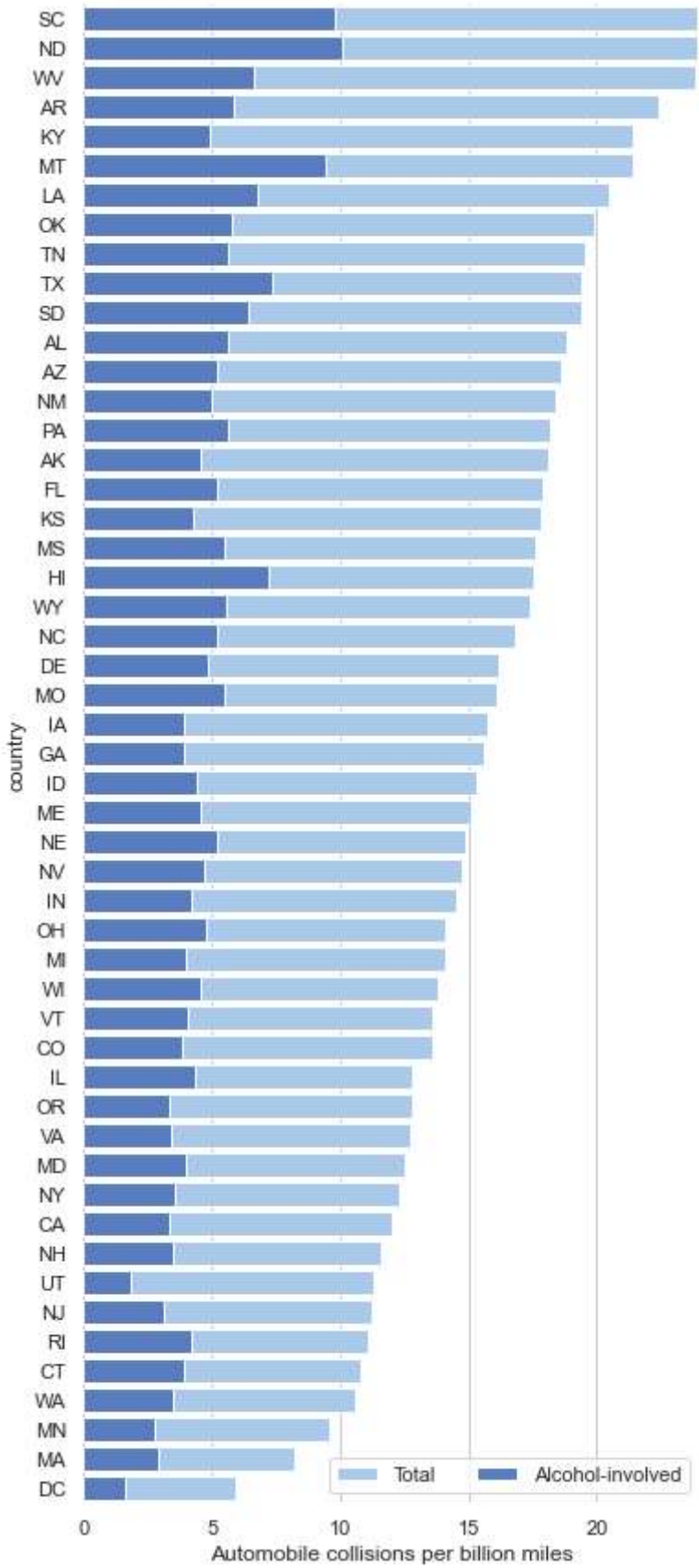
Out[64]:
```
<AxesSubplot:xlabel='total_bill', ylabel='tip'>
```



In [65]:
```
sns.regplot(x = df1.total_bill , y = df1.tip)
sns.set(rc={'figure.figsize':(30,20)})
```



In [66]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.set_theme(style="whitegrid")

# Initialize the matplotlib figure
f, ax = plt.subplots(figsize=(6, 15))

# Load the example car crash dataset
crashes = sns.load_dataset("car_crashes").sort_values("total", ascending=False)

# Plot the total crashes
sns.set_color_codes("pastel")
sns.barplot(x="total", y="abbrev", data=crashes,
            label="Total", color="b")

# Plot the crashes where alcohol was involved
sns.set_color_codes("muted")
sns.barplot(x="alcohol", y="abbrev", data=crashes,
            label="Alcohol-involved", color="b")

# Add a legend and informative axis label
ax.legend(ncol=2, loc="lower right", frameon=True)
ax.set(xlim=(0, 24), ylabel="country",xlabel="Automobile collisions per billion m
sns.despine(left=True, bottom=True)
```


SC
ND
WV
AR
KY
MT
LA
OK
TN
TX
SD
AL
AZ
NM
PA
AK
FL
KS
MS
HI
WY
NC
DE
MO
IA
GA
ID
ME
NE
NV
IN
OH
MI
WI
VT
CO
IL
OR
VA
MD
NY
CA
NH
UT
NJ
RI
CT
WA
MN
MA
DC
country
Total
Alcohol-involved
0
5
10
15
20
Automobile collisions per billion miles

In [67]:
```
sns.load_dataset("car_crashes")
```

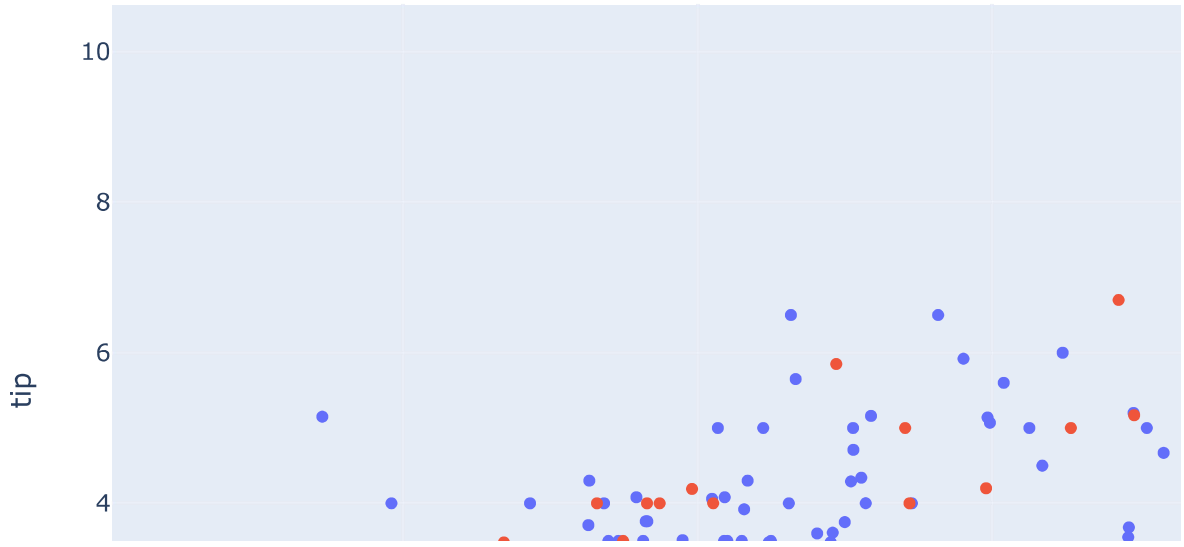
Out[67]:

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
0	18.8	7.332	5.640	18.048	15.040	784.55	145.08	AL
1	18.1	7.421	4.525	16.290	17.014	1053.48	133.93	AK
2	18.6	6.510	5.208	15.624	17.856	899.47	110.35	AZ
3	22.4	4.032	5.824	21.056	21.280	827.34	142.39	AR
4	12.0	4.200	3.360	10.920	10.680	878.41	165.63	CA
5	13.6	5.032	3.808	10.744	12.920	835.50	139.91	CO
6	10.8	4.968	3.888	9.396	8.856	1068.73	167.02	CT
7	16.2	6.156	4.860	14.094	16.038	1137.87	151.48	DE
8	5.9	2.006	1.593	5.900	5.900	1273.89	136.05	DC
9	17.9	3.759	5.191	16.468	16.826	1160.13	144.18	FL
10	15.6	2.964	3.900	14.820	14.508	913.15	142.80	GA
11	17.5	9.450	7.175	14.350	15.225	861.18	120.92	HI
12	15.3	5.508	4.437	13.005	14.994	641.96	82.75	ID
13	12.8	4.608	4.352	12.032	12.288	803.11	139.15	IL
14	14.5	3.625	4.205	13.775	13.775	710.46	108.92	IN
15	15.7	2.669	3.925	15.229	13.659	649.06	114.47	IA
16	17.8	4.806	4.272	13.706	15.130	780.45	133.80	KS
17	21.4	4.066	4.922	16.692	16.264	872.51	137.13	KY
18	20.5	7.175	6.765	14.965	20.090	1281.55	194.78	LA
19	15.1	5.738	4.530	13.137	12.684	661.88	96.57	ME
20	12.5	4.250	4.000	8.875	12.375	1048.78	192.70	MD
21	8.2	1.886	2.870	7.134	6.560	1011.14	135.63	MA
22	14.1	3.384	3.948	13.395	10.857	1110.61	152.26	MI
23	9.6	2.208	2.784	8.448	8.448	777.18	133.35	MN
24	17.6	2.640	5.456	1.760	17.600	896.07	155.77	MS
25	16.1	6.923	5.474	14.812	13.524	790.32	144.45	MO
26	21.4	8.346	9.416	17.976	18.190	816.21	85.15	MT
27	14.9	1.937	5.215	13.857	13.410	732.28	114.82	NE
28	14.7	5.439	4.704	13.965	14.553	1029.87	138.71	NV
29	11.6	4.060	3.480	10.092	9.628	746.54	120.21	NH
30	11.2	1.792	3.136	9.632	8.736	1301.52	159.85	NJ
31	18.4	3.496	4.968	12.328	18.032	869.85	120.75	NM
32	12.3	3.936	3.567	10.824	9.840	1234.31	150.01	NY
33	16.8	6.552	5.208	15.792	13.608	708.24	127.82	NC

	total	speeding	alcohol	not_distracted	no_previous	ins_premium	ins_losses	abbrev
34	23.9	5.497	10.038	23.661	20.554	688.75	109.72	ND
35	14.1	3.948	4.794	13.959	11.562	697.73	133.52	OH
36	19.9	6.368	5.771	18.308	18.706	881.51	178.86	OK
37	12.8	4.224	3.328	8.576	11.520	804.71	104.61	OR
38	18.2	9.100	5.642	17.472	16.016	905.99	153.86	PA
39	11.1	3.774	4.218	10.212	8.769	1148.99	148.58	RI
40	23.9	9.082	9.799	22.944	19.359	858.97	116.29	SC
41	19.4	6.014	6.402	19.012	16.684	669.31	96.87	SD
42	19.5	4.095	5.655	15.990	15.795	767.91	155.57	TN
43	19.4	7.760	7.372	17.654	16.878	1004.75	156.83	TX
44	11.3	4.859	1.808	9.944	10.848	809.38	109.48	UT
45	13.6	4.080	4.080	13.056	12.920	716.20	109.61	VT
46	12.7	2.413	3.429	11.049	11.176	768.95	153.72	VA
47	10.6	4.452	3.498	8.692	9.116	890.03	111.62	WA
48	23.8	8.092	6.664	23.086	20.706	992.61	152.56	WV
49	13.8	4.968	4.554	5.382	11.592	670.31	106.62	WI
50	17.4	7.308	5.568	14.094	15.660	791.14	122.04	WY

In [68]:
```
# fromm plotly exprss documentation
```

In [69]:
```
import plotly.express as px

fig = px.scatter(df1, x="total_bill", y="tip", color="time")
fig.show()
```



In [70]:
```
import plotly.express as px

fig = px.bar(df1, x="sex", y="total_bill", color="smoker", barmode="group")
fig.show()
```