03 March

April 13, 2023

```
[ ]: Que 1: Name any five plots that we can plot using the Seaborn library. Also,␣
      ↪state the uses of each plot.
```

```
[ ]: ANS -
```

```
[ ]: 1 . Line plot :
                   A line plot is the simplest plot in all plotting types, AS it is␣
      ↪the visulization od a single function
            . This plot helps us to see the relationship between x-axis , y-axis␣
      ↪and it also takes some parameters such as hue
            size,color,etc.
```

```
[ ]: 2 . count plot :
                   A count plot is used to show the counts of observations in each␣
      ↪categorical bin using bars.
            This method is  accepting the parameters x,y which takes the name of A␣
      ↪variable in data, hue it is an
            optional parameter that takes dataframe, array , or list of array␣
      ↪dataset for plotting.
```

```
[ ]: 3 .  bar plot :
                   A bar chart is a way of comparing a set of categorical data. it␣
      ↪is better to convert continuous data
            to bins before plotting the bar chart display data using several bars␣
      ↪each representing a particular category,
            This method is acepting the parameters x,y which take the name of a␣
      ↪variable in data , hue it is an optional
            parameter it helps to take column name for color encoding.
```

```
[ ]: 4 . scatter plot :
                      scatter plot is the same as line plot, in a line plot isntead␣
      ↪of points being joined by line segnment
            The points are shown individually with a dot,circle,or ant orther shape.
      ↪ The position of  each marker on the horizontal
            and vertical axis indicates value for an individual  data point. This␣
      ↪polt is used to observe relationship between variable
```

```
        This method is accepting the parameters x,y which take the name of a␣
↪variable in datahue it is an optional parameter it help
        to take columns name for color encoding, marker is a parameter that it␣
↪used to change the shape of the point.
```

[ ]:
```
5 . histogrm :
              A histogram is a graph that shows the underlying frequency␣
↪distribution of set of continuous data.
        This chart is helpful in data collection and data analysis and hence it␣
↪is distribution of the data, outlier, skew
        present in the data.
```

[ ]:
```

```

[ ]:
```

```

[ ]:
```
Que 2: Load the "fmri" dataset using the load_dataset function of seaborn. Plot␣
↪a line plot using x =
"timepoint" and y = "signal" for different events and regions.
```

[ ]:
```
ANS -
```

[1]:
```
import seaborn as sns
```

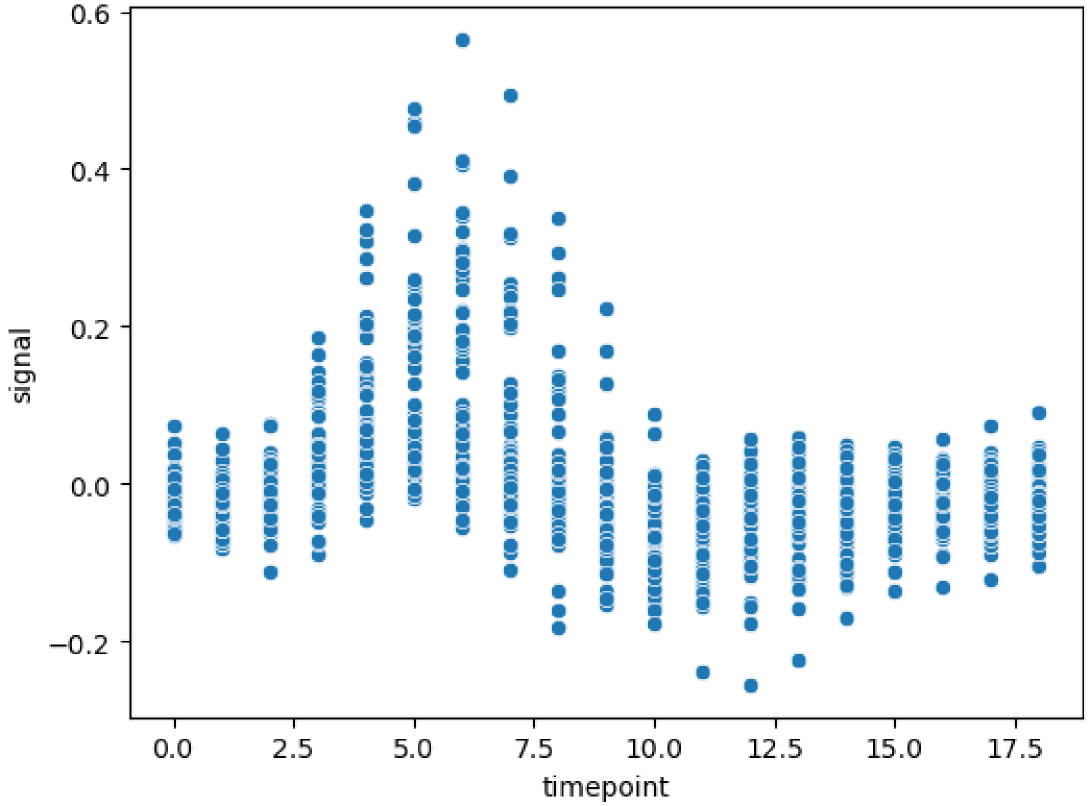
[26]:
```
fmri = sns.load_dataset('fmri')
```

[27]:
```
fmri
```

[27]:
```
     subject  timepoint event    region    signal
0        s13         18  stim  parietal -0.017552
1         s5         14  stim  parietal -0.080883
2        s12         18  stim  parietal -0.081033
3        s11         18  stim  parietal -0.046134
4        s10         18  stim  parietal -0.037970
…        …          …   …       …         …
1059      s0          8   cue   frontal  0.018165
1060     s13          7   cue   frontal -0.029130
1061     s12          7   cue   frontal -0.004939
1062     s11          7   cue   frontal -0.025367
1063      s0          0   cue  parietal -0.006899

[1064 rows x 5 columns]
```

[28]:
```
sns.scatterplot(x = fmri.timepoint , y = fmri.signal)
```

[28]:
```
<AxesSubplot: xlabel='timepoint', ylabel='signal'>
```



```
[ ]:
```

```
[ ]:
```

```
[ ]: Que 3: Load the "titanic" dataset using the load_dataset function of seaborn.␣
      ↪Plot two box plots using x =
     'pclass', y = 'age' and y = 'fare'.
```

```
[ ]: ANS -
```

```
[9]: import seaborn as sns
```

```
[10]: titanic = sns.load_dataset('titanic')
```

```
[11]: titanic
```

```
[11]:      survived  pclass     sex   age  sibsp  parch     fare embarked   class  \
      0           0       3    male  22.0      1      0   7.2500        S   Third
      1           1       1  female  38.0      1      0  71.2833        C   First
      2           1       3  female  26.0      0      0   7.9250        S   Third
      3           1       1  female  35.0      1      0  53.1000        S   First
```

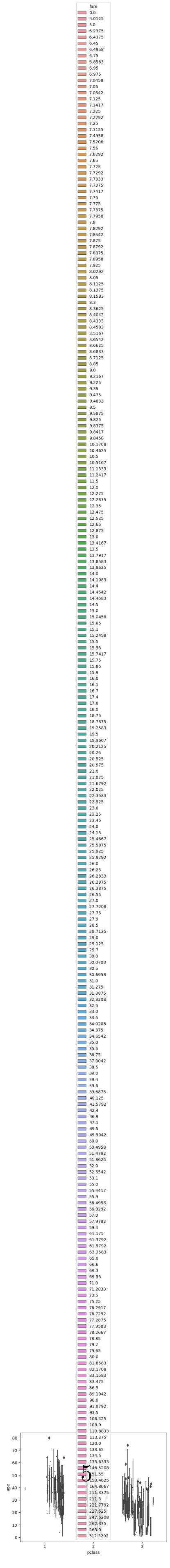
```
4           0       3    male  35.0      0      0   8.0500        S   Third
..        …     …     …   …    …    …      …      …     …
886         0       2    male  27.0      0      0  13.0000        S  Second
887         1       1  female  19.0      0      0  30.0000        S   First
888         0       3  female   NaN      1      2  23.4500        S   Third
889         1       1    male  26.0      0      0  30.0000        C   First
890         0       3    male  32.0      0      0   7.7500        Q   Third

        who adult_male deck  embark_town alive  alone
0       man       True  NaN  Southampton    no  False
1     woman      False    C    Cherbourg   yes  False
2     woman      False  NaN  Southampton   yes   True
3     woman      False    C  Southampton   yes  False
4       man       True  NaN  Southampton    no   True
..      …          …   …          …   …    …
886     man       True  NaN  Southampton    no   True
887   woman      False    B  Southampton   yes   True
888   woman      False  NaN  Southampton    no  False
889     man       True    C    Cherbourg   yes   True
890     man       True  NaN   Queenstown    no   True

[891 rows x 15 columns]
```

[47]:

```
sns.boxplot(data = titanic , x = "pclass" , y = "age" , hue = "fare")
```

[47]:

```
<AxesSubplot: xlabel='pclass', ylabel='age'>
```


fare
0.0
4.0125
5.0
6.2375
6.4375
6.45
6.4958
6.75
6.8583
6.95
6.975
7.0458
7.05
7.0542
7.125
7.1417
7.225
7.2292
7.25
7.3125
7.4958
7.5208
7.55
7.6292
7.65
7.725
7.7292
7.7333
7.7375
7.7417
7.75
7.775
7.7875
7.7958
7.8
7.8292
7.8542
7.875
7.8792
7.8875
7.8958
7.925
8.0292
8.05
8.1125
8.1375
8.1583
8.3
8.3625
8.4042
8.4333
8.4583
8.5167
8.6542
8.6625
8.6833
8.7125
8.85
9.0
9.2167
9.225
9.35
9.475
9.4833
9.5
9.5875
9.825
9.8375
9.8417
9.8458
10.1708
10.4625
10.5
10.5167
11.1333
11.2417
11.5
12.0
12.275
12.2875
12.35
12.475
12.525
12.65
12.875
13.0
13.4167
13.5
13.7917
13.8583
13.8625
14.0
14.1083
14.4
14.4542
14.4583
14.5
15.0
15.0458
15.05
15.1
15.2458
15.5
15.55
15.7417
15.75
15.85
15.9
16.0
16.1
16.7
17.4
17.8
18.0
18.75
18.7875
19.2583
19.5
19.9667
20.2125
20.25
20.525
20.575
21.0
21.075
21.6792
22.025
22.3583
22.525
23.0
23.25
23.45
24.0
24.15
25.4667
25.5875
25.925
25.9292
26.0
26.25
26.2833
26.2875
26.3875
26.55
27.0
27.7208
27.75
27.9
28.5
28.7125
29.0
29.125
29.7
30.0
30.0708
30.5
30.6958
31.0
31.275
31.3875
32.3208
32.5
33.0
33.5
34.0208
34.375
34.6542
35.0
35.5
36.75
37.0042
38.5
39.0
39.4
39.6
39.6875
40.125
41.5792
42.4
46.9
47.1
49.5
49.5042
50.0
50.4958
51.4792
51.8625
52.0
52.5542
53.1
55.0
55.4417
55.9
56.4958
56.9292
57.0
57.9792
59.4
61.175
61.3792
61.9792
63.3583
65.0
66.6
69.3
69.55
71.0
71.2833
73.5
75.25
76.2917
76.7292
77.2875
77.9583
78.2667
78.85
79.2
79.65
80.0
81.8583
82.1708
83.1583
83.475
86.5
89.1042
90.0
91.0792
93.5
106.425
108.9
110.8833
113.275
120.0
133.65
134.5
135.6333
146.5208
151.55
153.4625
164.8667
211.3375
211.5
221.7792
227.525
247.5208
262.375
263.0
512.3292
80
70
60
50
40
30
20
10
0
age
1
2
3
pclass

```
[ ]:
```

```
[ ]:
```

```
[ ]: Que 4: Use the "diamonds" dataset from seaborn to plot a histogram for the␣
      ↪'price' column. Use the hue
     parameter for the 'cut' column of the diamonds dataset.
```

```
[ ]: ANS -
```

```
[12]: import seaborn as sns
```

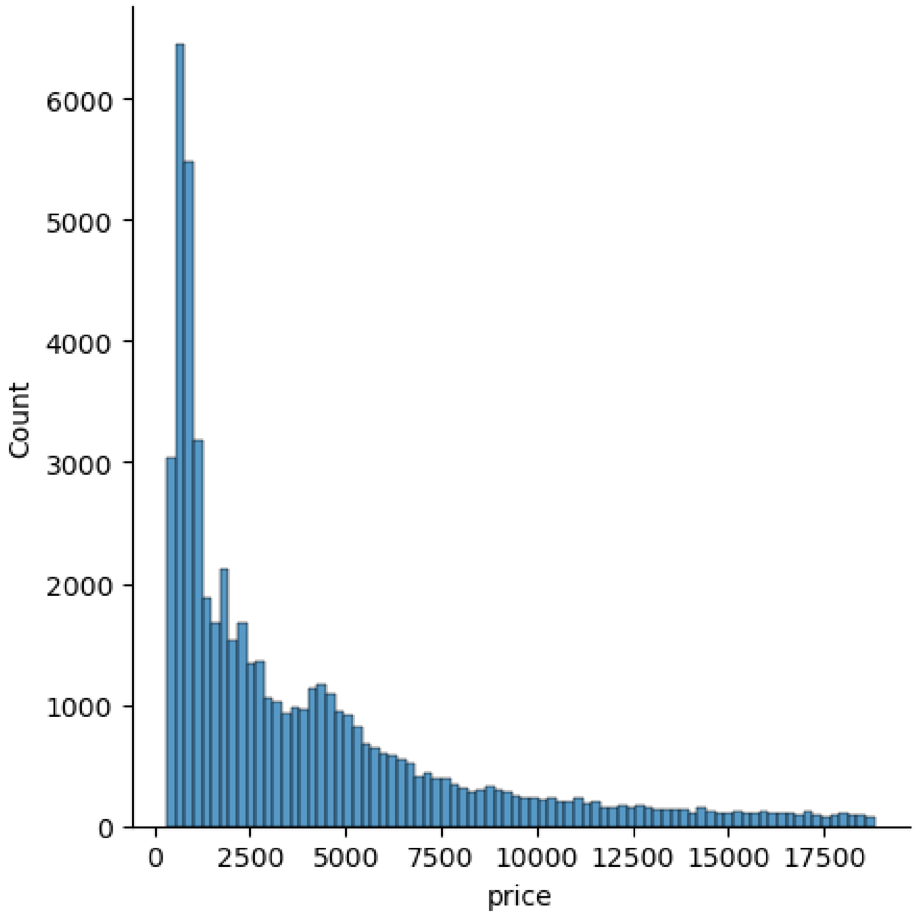
```
[13]: diamonds = sns.load_dataset('diamonds')
```

```
[14]: diamonds
```

```
[14]:        carat        cut color clarity  depth  table  price     x     y     z
      0       0.23      Ideal     E     SI2   61.5   55.0    326  3.95  3.98  2.43
      1       0.21    Premium     E     SI1   59.8   61.0    326  3.89  3.84  2.31
      2       0.23       Good     E     VS1   56.9   65.0    327  4.05  4.07  2.31
      3       0.29    Premium     I     VS2   62.4   58.0    334  4.20  4.23  2.63
      4       0.31       Good     J     SI2   63.3   58.0    335  4.34  4.35  2.75
      …       …        …   …     …    …     …     …   …   …   …
      53935   0.72      Ideal     D     SI1   60.8   57.0   2757  5.75  5.76  3.50
      53936   0.72       Good     D     SI1   63.1   55.0   2757  5.69  5.75  3.61
      53937   0.70  Very Good     D     SI1   62.8   60.0   2757  5.66  5.68  3.56
      53938   0.86    Premium     H     SI2   61.0   58.0   2757  6.15  6.12  3.74
      53939   0.75      Ideal     D     SI2   62.2   55.0   2757  5.83  5.87  3.64

      [53940 rows x 10 columns]
```

```
[41]: sns.displot(diamonds['price'])
```

```
[41]: <seaborn.axisgrid.FacetGrid at 0x7fde3a9efdf0>
```



[43]:

```
diamonds.head()
```

[43]:

```
   carat      cut color clarity  depth  table  price     x     y     z
0   0.23    Ideal     E     SI2   61.5   55.0    326  3.95  3.98  2.43
1   0.21  Premium     E     SI1   59.8   61.0    326  3.89  3.84  2.31
2   0.23     Good     E     VS1   56.9   65.0    327  4.05  4.07  2.31
3   0.29  Premium     I     VS2   62.4   58.0    334  4.20  4.23  2.63
4   0.31     Good     J     SI2   63.3   58.0    335  4.34  4.35  2.75
```

[44]:

```
diamonds['cut'].value_counts()
```

[44]:

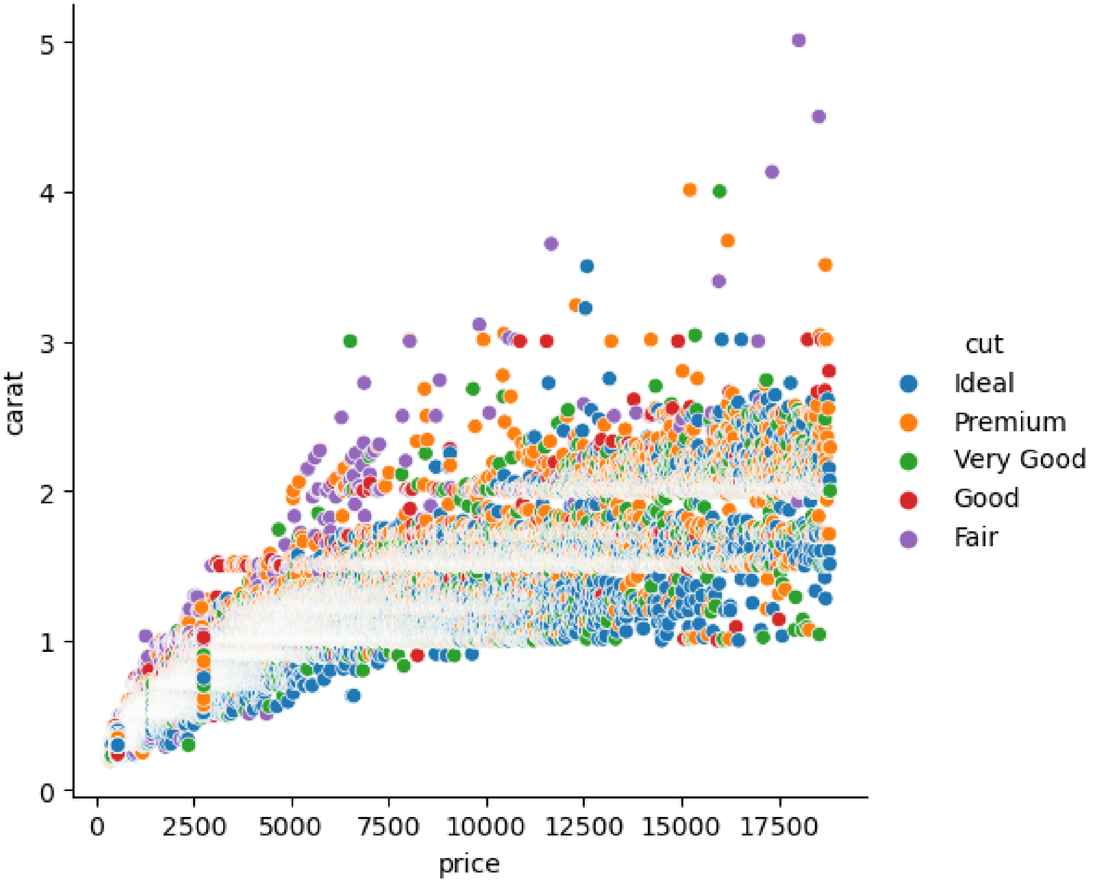
```
Ideal        21551
Premium      13791
Very Good    12082
Good          4906
Fair          1610
Name: cut, dtype: int64
```

[46]:

```
sns.relplot(x = diamonds.price , y = diamonds.carat , data = diamonds ,hue =␣
  ↪'cut')
```

[46]:

```
<seaborn.axisgrid.FacetGrid at 0x7fde3aa0d2d0>
```



[ ]:

```
```

[ ]:

```
```

[ ]:

```
```

[ ]:

```
```

[ ]:

```
```

[ ]:

```
Que 5: Use the "iris" dataset from seaborn to plot a pair plot. Use the hue␣
  ↪parameter for the "species" column
of the iris dataset.
```

```
[ ]: ANS -
```

```
[19]: import seaborn as sns
```

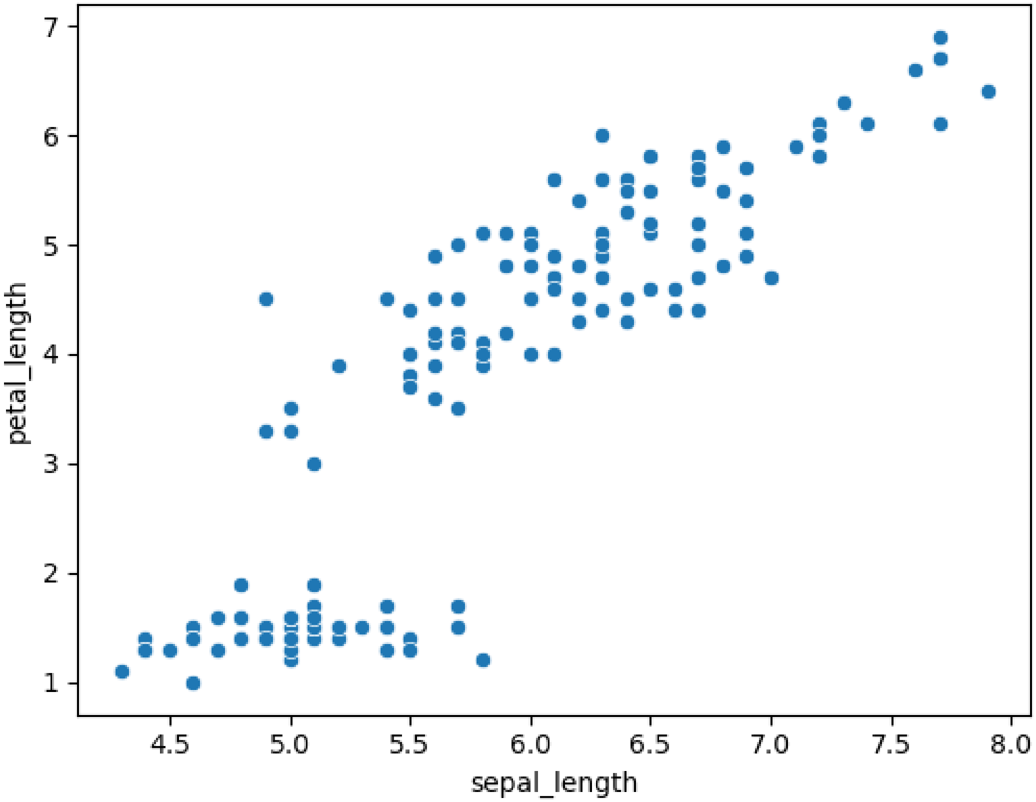
```
[20]: iris = sns.load_dataset('iris')
```

```
[21]: iris
```

```
[21]:      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 x 5 columns]
```

```
[32]: sns.scatterplot(x = iris.sepal_length, y = iris.petal_length)
```

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



[33]:
```
iris.head()
```

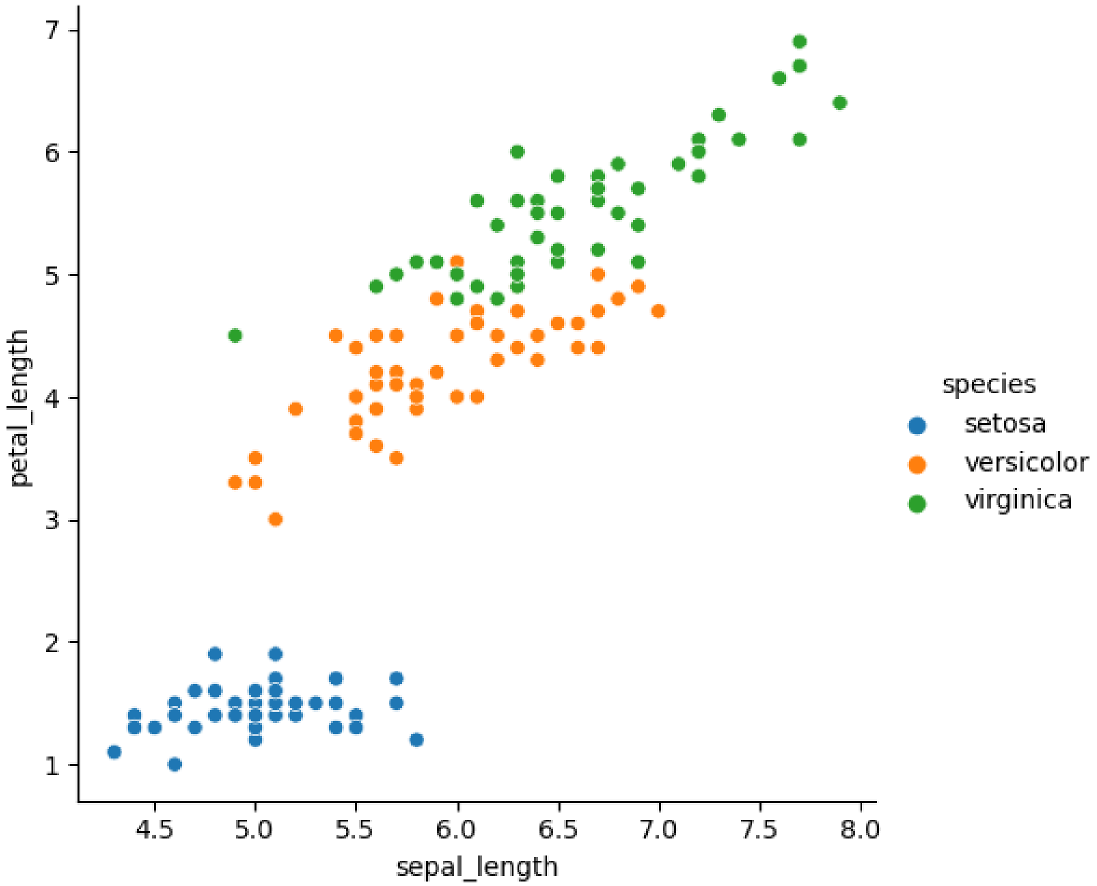
[33]:
```
   sepal_length  sepal_width  petal_length  petal_width species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```

[34]:
```
iris['species'].value_counts()
```

[34]:
```
setosa        50
versicolor    50
virginica     50
Name: species, dtype: int64
```

[35]:
```
sns.relplot(x = iris.sepal_length , y = iris.petal_length , data = iris ,hue =␣
 ↪'species')
```

[35]:
```
<seaborn.axisgrid.FacetGrid at 0x7fde3adf5d20>
```



[ ]:

[ ]:

[ ]:

[ ]:
```
Que 6: Use the "flights" dataset from seaborn to plot a heatmap.
```

[ ]:
```
ANS -
```

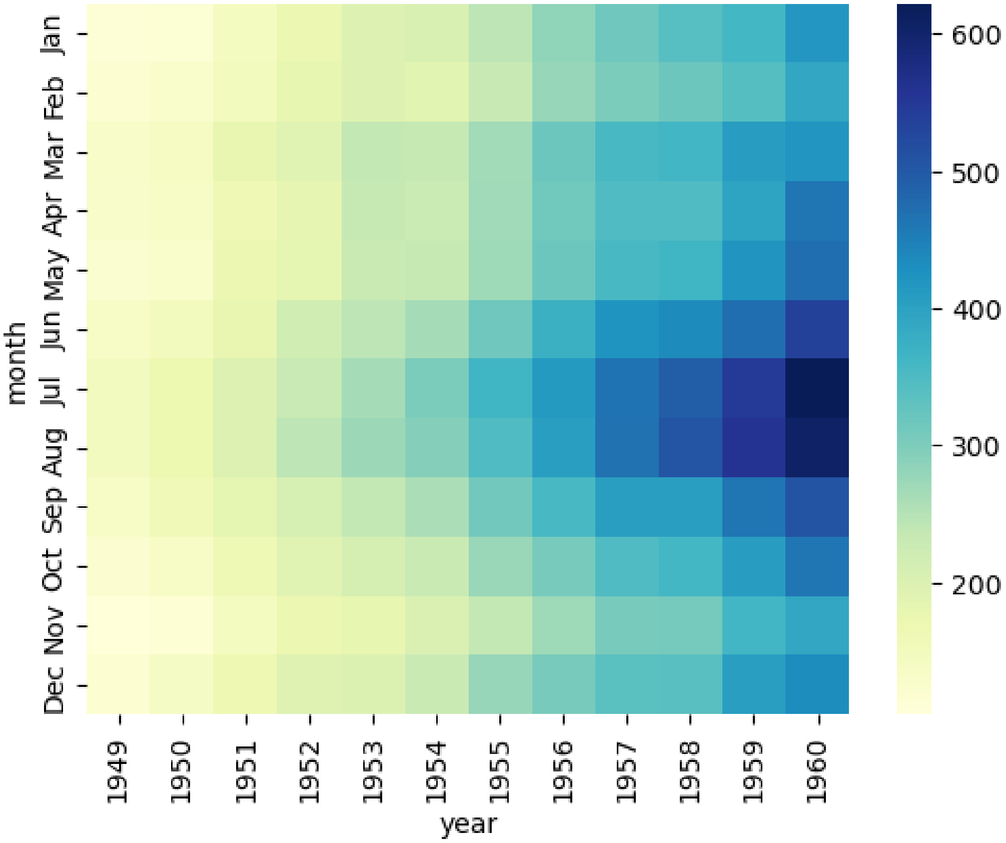
[53]:
```
import seaborn as sns
```

[54]:
```
flights_data = sns.load_dataset("flights")
flights_data = flights_data.pivot("month", "year", "passengers")
sns.heatmap(flights_data, cmap="YlGnBu")
```

```
/tmp/ipykernel_103/2858154181.py:2: FutureWarning: In a future version of pandas
all arguments of DataFrame.pivot will be keyword-only.
  flights_data = flights_data.pivot("month", "year", "passengers")
```

[54]:

```
<AxesSubplot: xlabel='year', ylabel='month'>
```



month
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Oct
Nov
Dec
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
year
600
500
400
300
200

[52]:

```
```

[ ]:

```
```