Exp No. 4 Study of different visualization techniques for the given data in Python using Seaborn, Matplotlib, and Pandas.

Roll no:- 22354 Name: Malhar Choure

Performed Date: 19/03/22

Submission Date: 30/03/22

Seaborn

Seaborn is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis. Seaborn works easily with dataframes and the Pandas library. The graphs created can also be customized easily.

Different categories of plot in Seaborn

Plots are basically used for visualizing the relationship between variables. Those variables can be either be completely numerical or a category like a group, class or division. Seaborn divides plot into the below categories –

1. Relational plots: This plot is used to understand the relation between two variables.
1. Categorical plots: This plot deals with categorical variables and how they can be visualized.
1. Distribution plots: This plot is used for examining univariate and bivariate distributions
1. Regression plots: The regression plots in seaborn are primarily intended to add a visual guide that helps to emphasize patterns in a dataset during exploratory data analyses.
1. Matrix plots: A matrix plot is an array of scatterplots.
1. Multi-plot grids: It is an useful approach is to draw multiple instances of the same plot on different subsets of the dataset.

Installing Seaborn

```
import seaborn as sns
import pandas
import matplotlib.pyplot as plt
```

Seaborn has 18 in-built datasets, that can be found using the following command.

```
sns.get_dataset_names()
```

```
['anagrams',
 'anscombe',
```

```
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
 'fmri',
 'gammas',
 'geyser',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'taxis',
 'tips',
 'titanic']
```

We will be using the Titanic dataset for this tutorial.

```
df = sns.load_dataset('titanic')
print(df)
```

```
     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]
```

```
df.head()
```

```
   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

     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
```

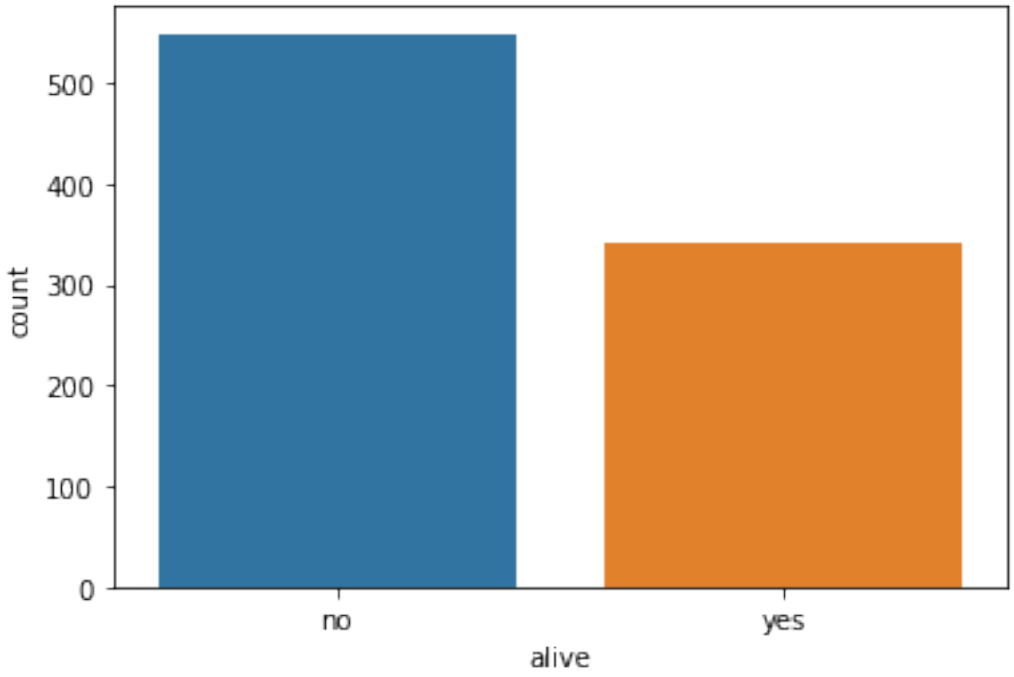
Different types of graphs

1. Count plot

A count plot is helpful when dealing with categorical values. It is used to plot the frequency of the different categories. The column alive contains categorical data in the titanic data, i.e., no and yes.

```
sns.countplot(x='alive',data=df)
```

```
<AxesSubplot:xlabel='alive', ylabel='count'>
```



data - The dataframe.

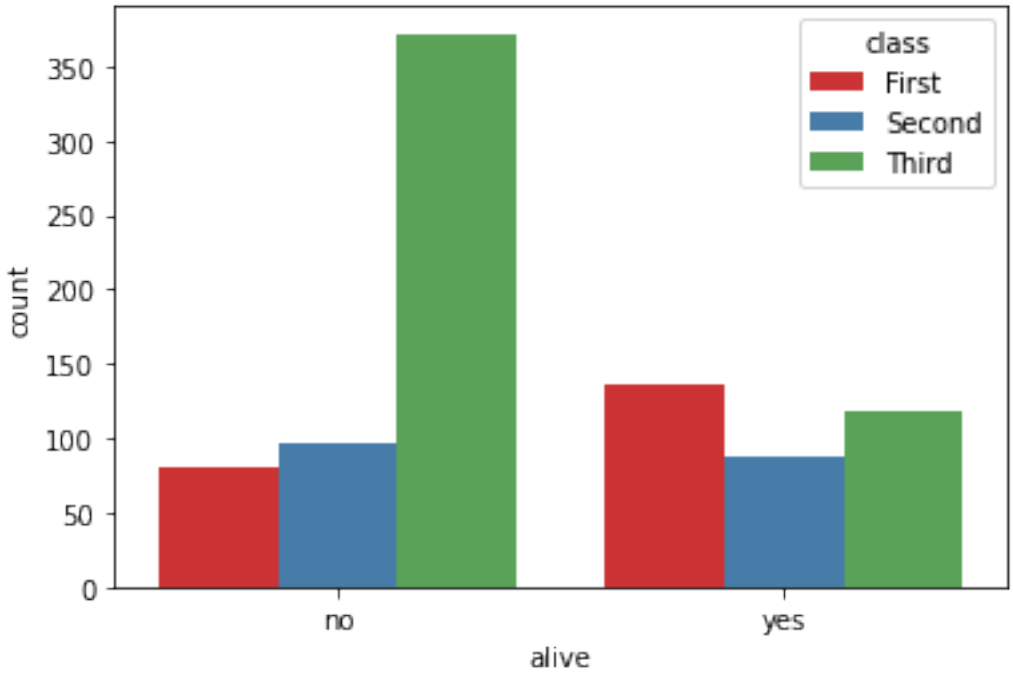
x - The name of the column.

We can observe from the graph that the number of passengers alive is significantly less than the number of passengers dead.

We can further break up the bars in the count plot based on another categorical variable. The color palette of the plot can also be customized.

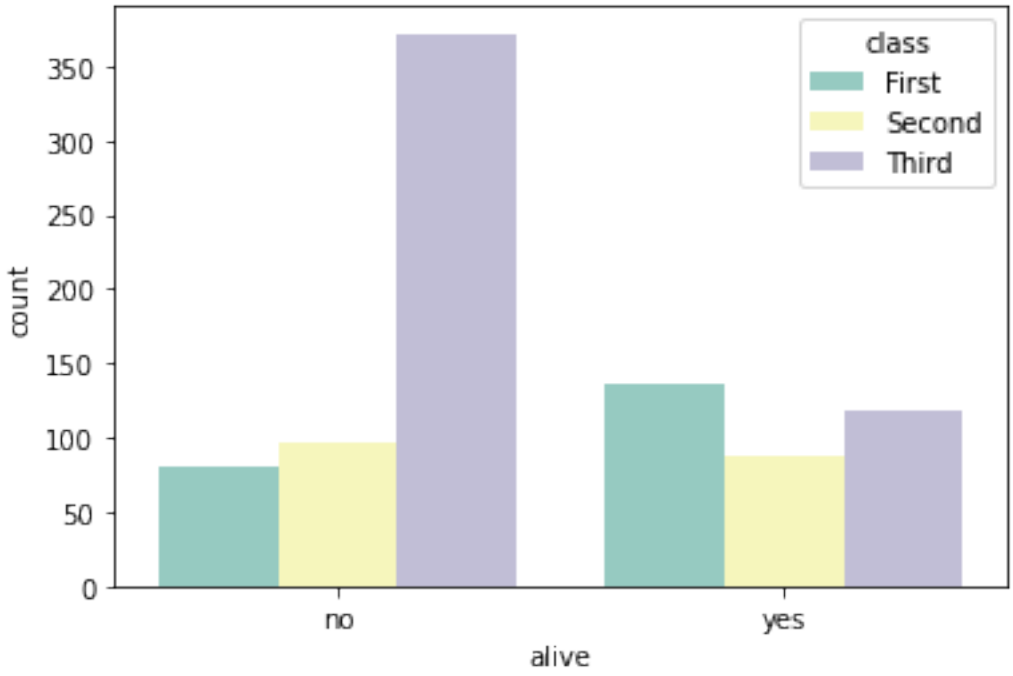
```
sns.countplot(x='alive', hue = 'class', data = df, palette = 'Set1')
```

```
<AxesSubplot:xlabel='alive', ylabel='count'>
```




```
sns.countplot(x='alive', hue = 'class', data = df, palette = 'Set3')
```

```
<AxesSubplot:xlabel='alive', ylabel='count'>
```

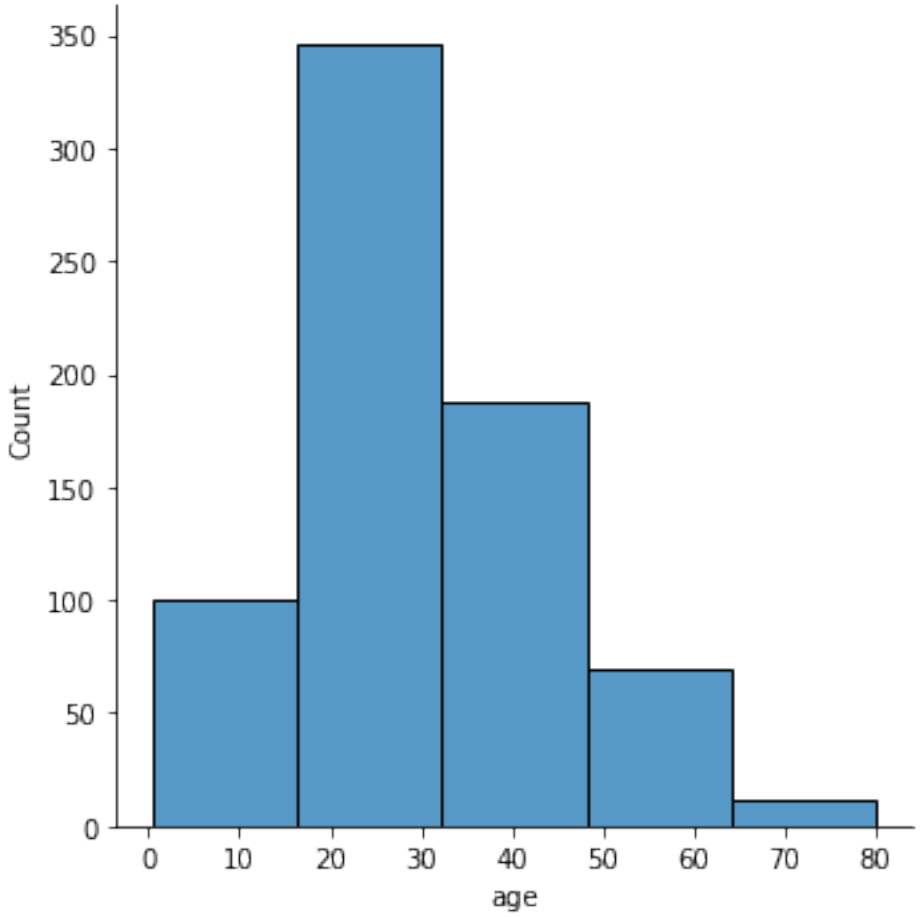


2. Distribution plot

A Distribution plot is used to plot the distribution of continuous data.

```
sns.displot(x = 'age', bins=5, data =df)
```

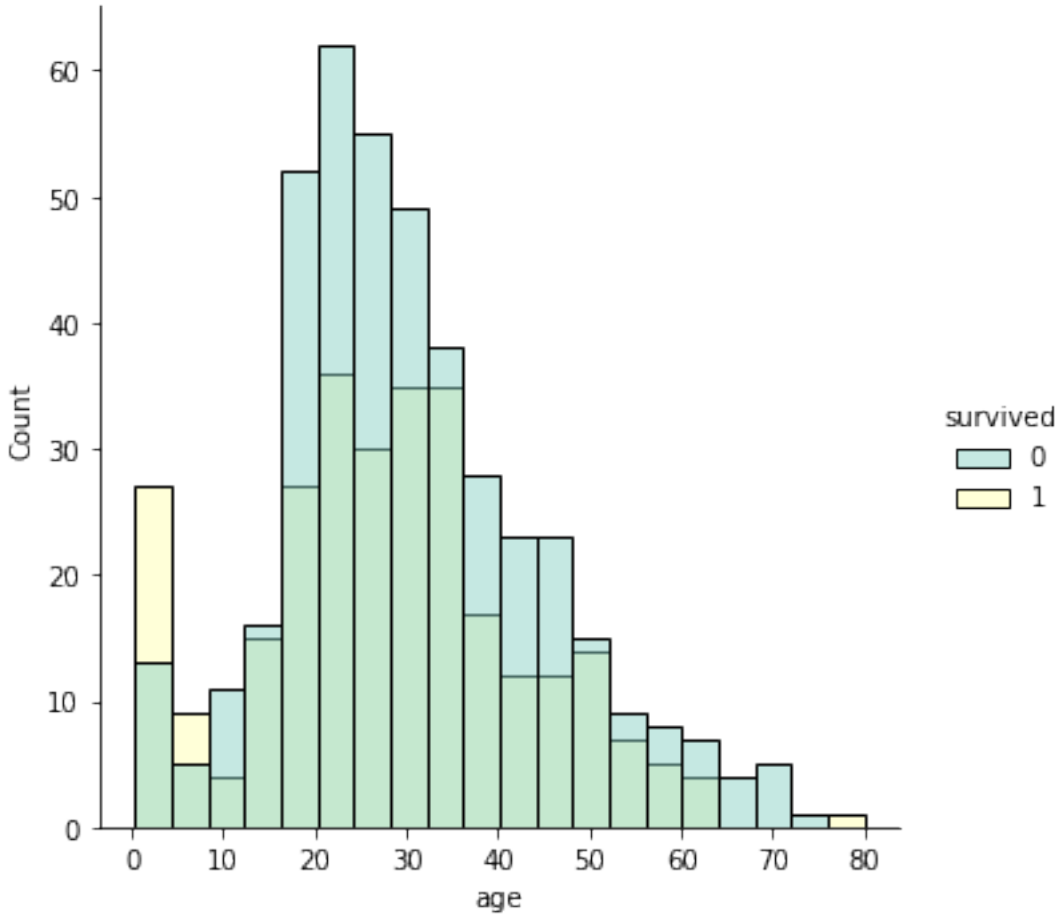
```
<seaborn.axisgrid.FacetGrid at 0x7f8840a8fd30>
```



The plot above tells us that most people onboard the titanic were in their mid-twenties.

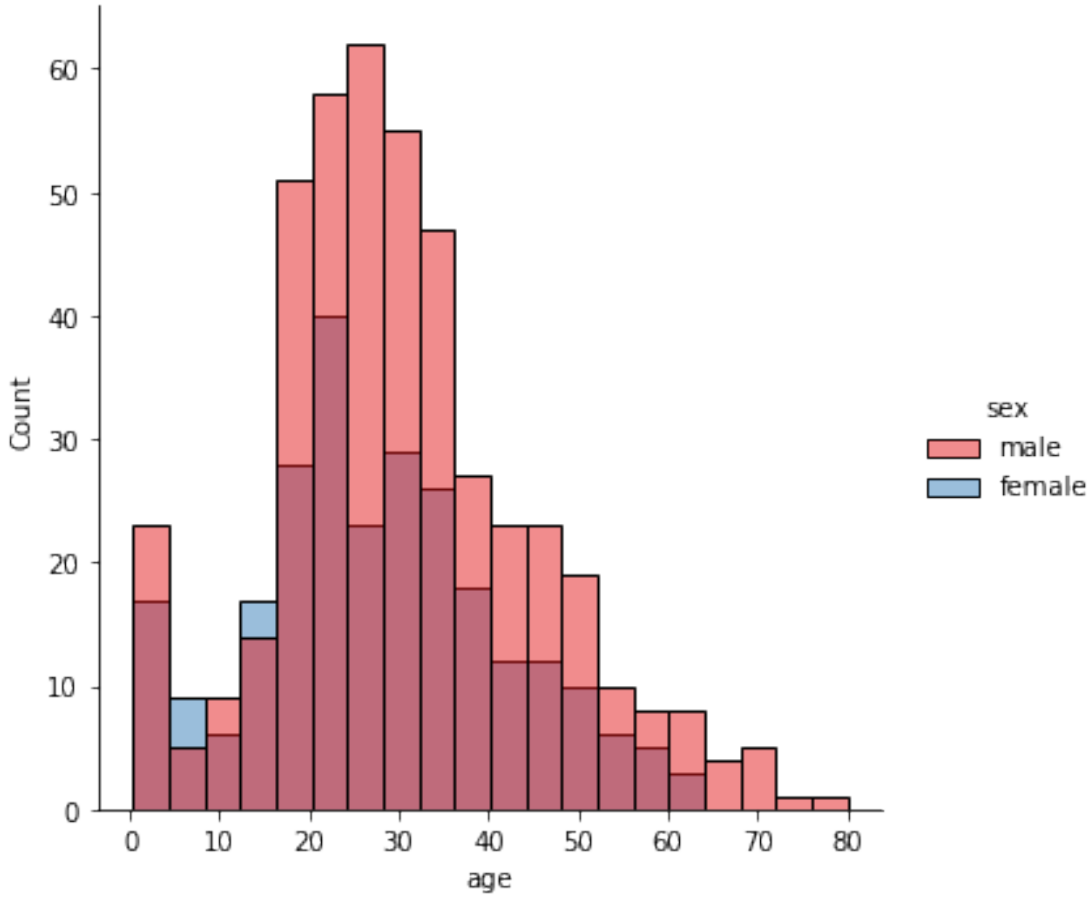
```
sns.displot(x ='age', hue = df['survived'] , palette = 'Set3',
data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f8830541550>
```




```
sns.displot(x ='age', hue = df['sex'] , palette = 'Set1', data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f8850607cd0>
```



You can also pass hue and palette as parameters to customize the graph.

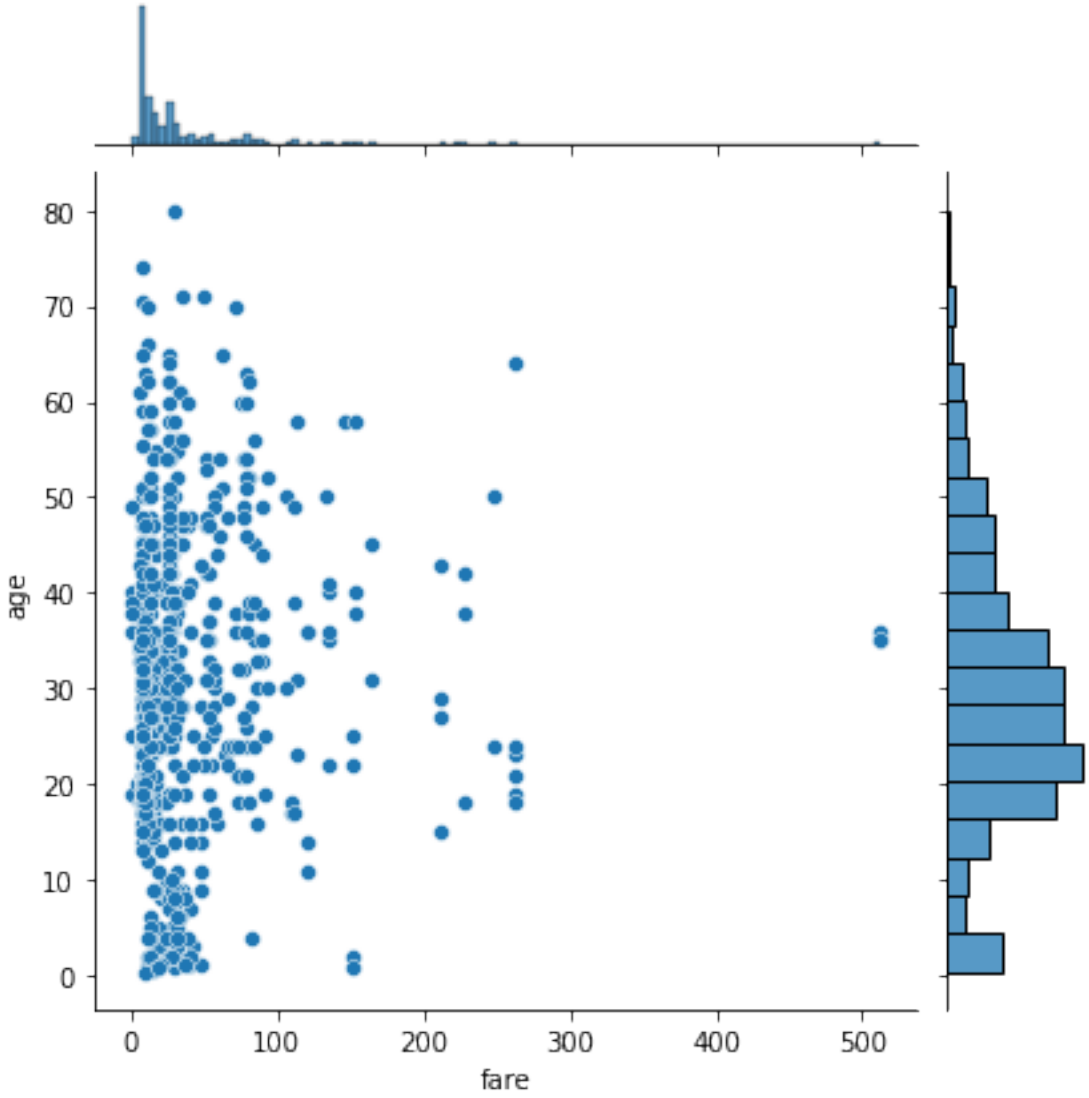
Most of the surviving passengers were in their high-twenties.

3. Joint plot

A Joint Plot is also used to plot the correlation between data.

```
sns.jointplot(x='fare' , y ='age', data = df)
```

```
<seaborn.axisgrid.JointGrid at 0x7f8850638a90>
```

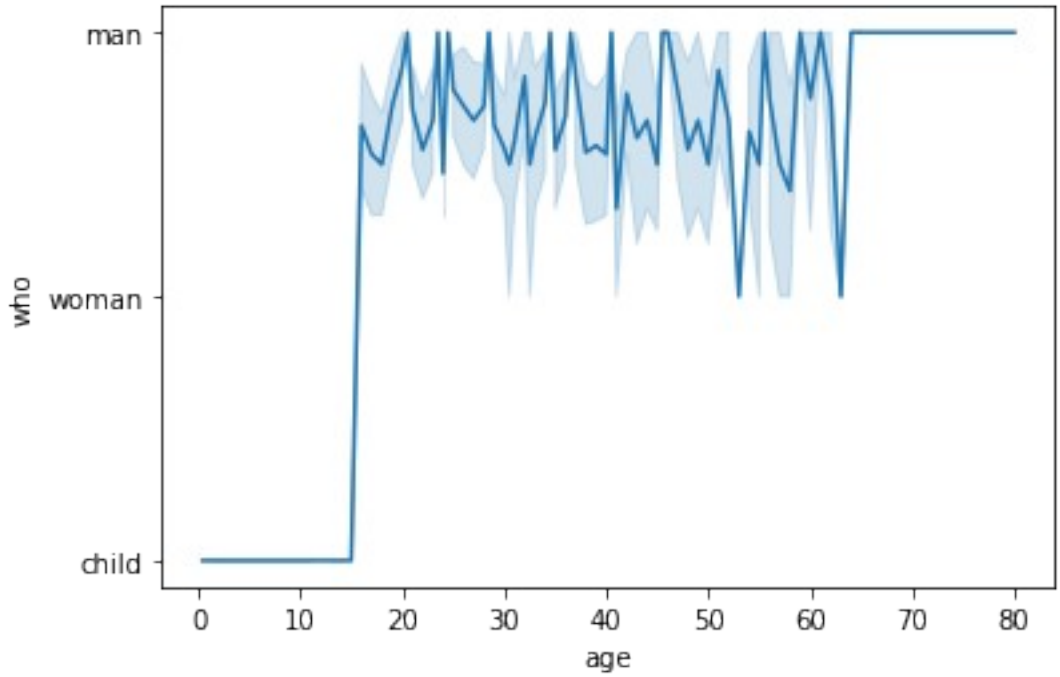


4. Line plot

The line plot is one of the most basic plot in seaborn library. This plot is mainly used to visualize the data in form of some time series, i.e. in continuous manner.

```
sns.lineplot(x="age",y="who",data=df)
```

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

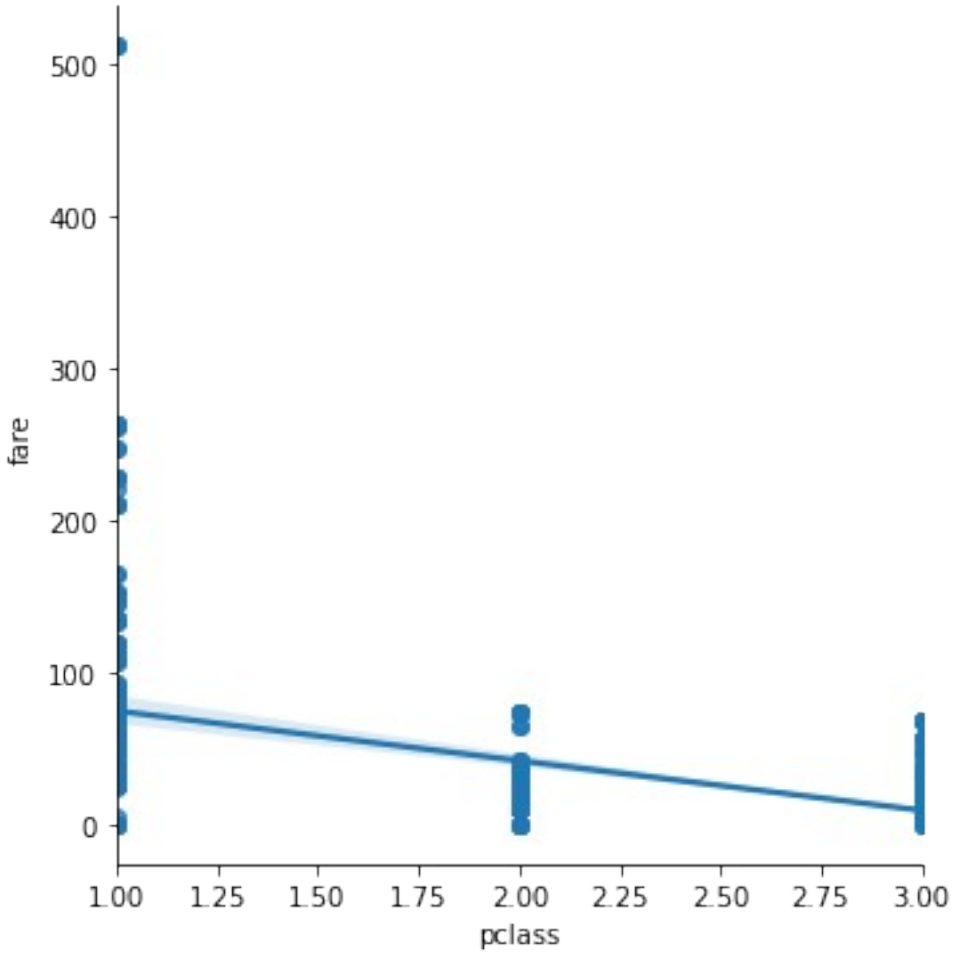


5. Lmplot

The lmplot is another most basic plot. It shows a line representing a linear regression model along with data points on the 2D-space and x and y can be set as the horizontal and vertical labels respectively.

```
sns.lmplot(x="pclass", y="fare", data=df)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f8830655610>
```



Other Graphs

Boxplot:

A boxplot is a standardized way of displaying the distribution of data based on a five number summary. It can tell you about your outliers and what their values are. It can also tell you if your data is symmetrical, how tightly your data is grouped, and if and how your data is skewed. For some distributions/datasets, you will find that you need more information than the measures of central tendency (median, mean, and mode).

Boxplots are based on a five number summary ("minimum", first quartile (Q1), median, third quartile (Q3), and "maximum").

median (Q2/50th Percentile): the middle value of the dataset.

first quartile (Q1/25th Percentile): the middle number between the smallest number (not the "minimum") and the median of the dataset.

third quartile (Q3/75th Percentile): the middle value between the median and the highest value (not the "maximum") of the dataset.

interquartile range (IQR): 25th to the 75th percentile.

“maximum”: Q3 + 1.5*IQR

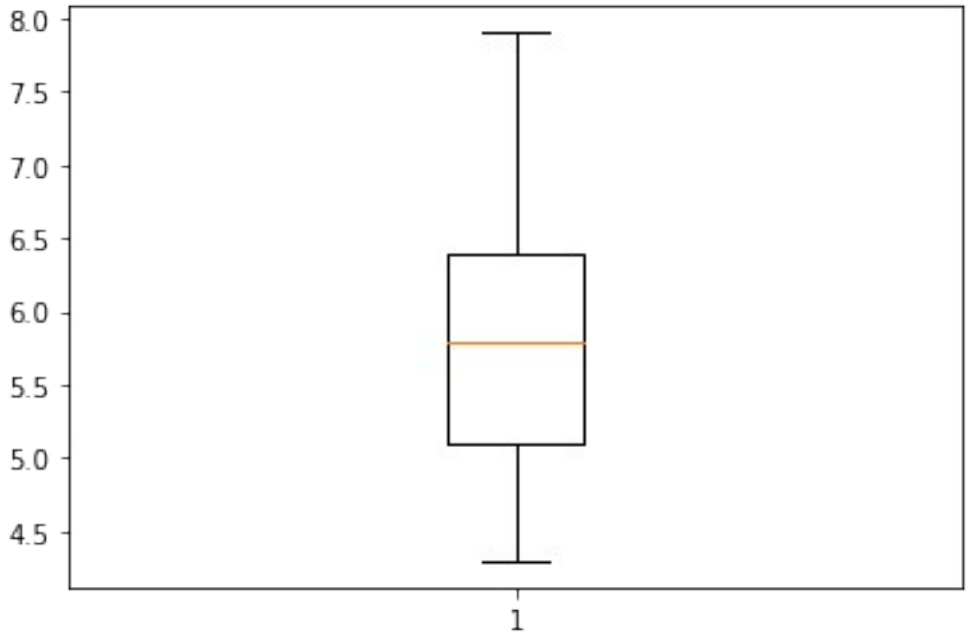
“minimum”: Q1 -1.5*IQR

```
df = sns.load_dataset('iris')
print(df,"\n")
print(df.head(),"\n")
print(df.tail(),"\n")
print(df.isnull().values.any(),"\n")
```

```
     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]

   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

     sepal_length  sepal_width  petal_length  petal_width    species
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

False
```

```
sepal_length = df["sepal_length"]
sepal_width  = df['sepal_width']
petal_length   = df['petal_length']
petal_width  = df['petal_width']
```

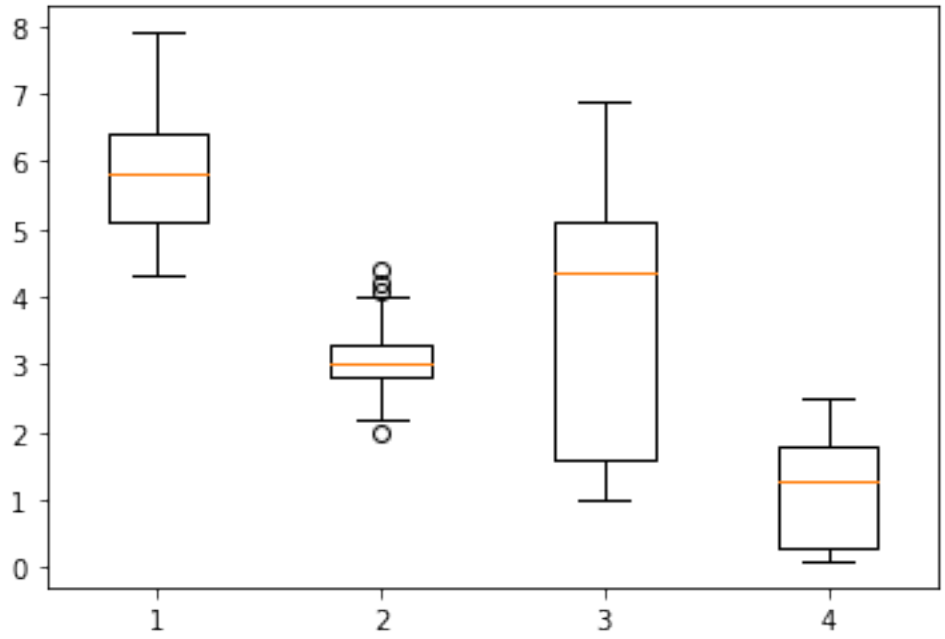
```
fig, ax = plt.subplots()
ax.boxplot(sepal_length)
plt.show()
```



After creating the plot, we can see some of the summary statistics for our data. The Box Plot shows the median of the dataset (the horizontal line in the middle), as well as the interquartile ranges (the ends of the boxes) and the minimum and maximum values of the chosen dataset feature.

We can also plot multiple columns on one figure, simply by providing more columns. This again, can be done on either the plt instance, the fig object or the ax object.

```
columns=[sepal_length,sepal_width,petal_length,petal_width]
fig, ax = plt.subplots()
ax.boxplot(columns)
plt.show()
```



Let's take the first box plot i.e, box plot of the figure and understand these statistical things:

Bottom horizontal line of blue box plot is minimum value. First horizontal line of rectangle shape of box plot is First quartile or 25%. Second horizontal line of rectangle shape of box plot is Second quartile or 50% or median. Third horizontal line of rectangle shape of box plot is third quartile or 75%. Top horizontal line of rectangle shape of box plot is maximum value. Small oval shape of second box plot is outlier data or erroneous data.

Heatmap:

Heatmap is defined as a graphical representation of data using colors to visualize the value of the matrix. It can be used to visualize confusion matrices, and correlation. In this, to represent more common values or higher activities brighter colors basically reddish colors are used and to represent less common or activity values, darker colors are preferred. Heatmap is also defined by the name of the shading matrix. Heatmaps in Seaborn can be plotted by using the seaborn.heatmap() function.

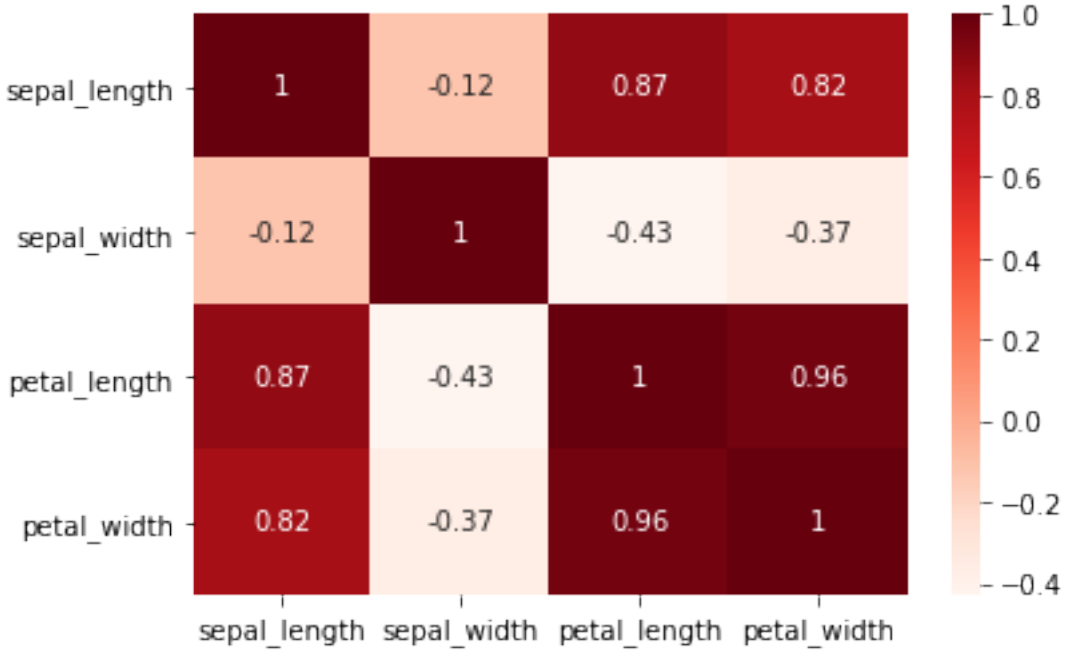
```
corr = df.corr()
sns.heatmap(corr)
```

```
<AxesSubplot:>
```

We can customize the color scheme, the minimum/maximum values, and annotations.

```
sns.heatmap(corr, cmap='Reds',annot = True)
```

```
<AxesSubplot:>
```



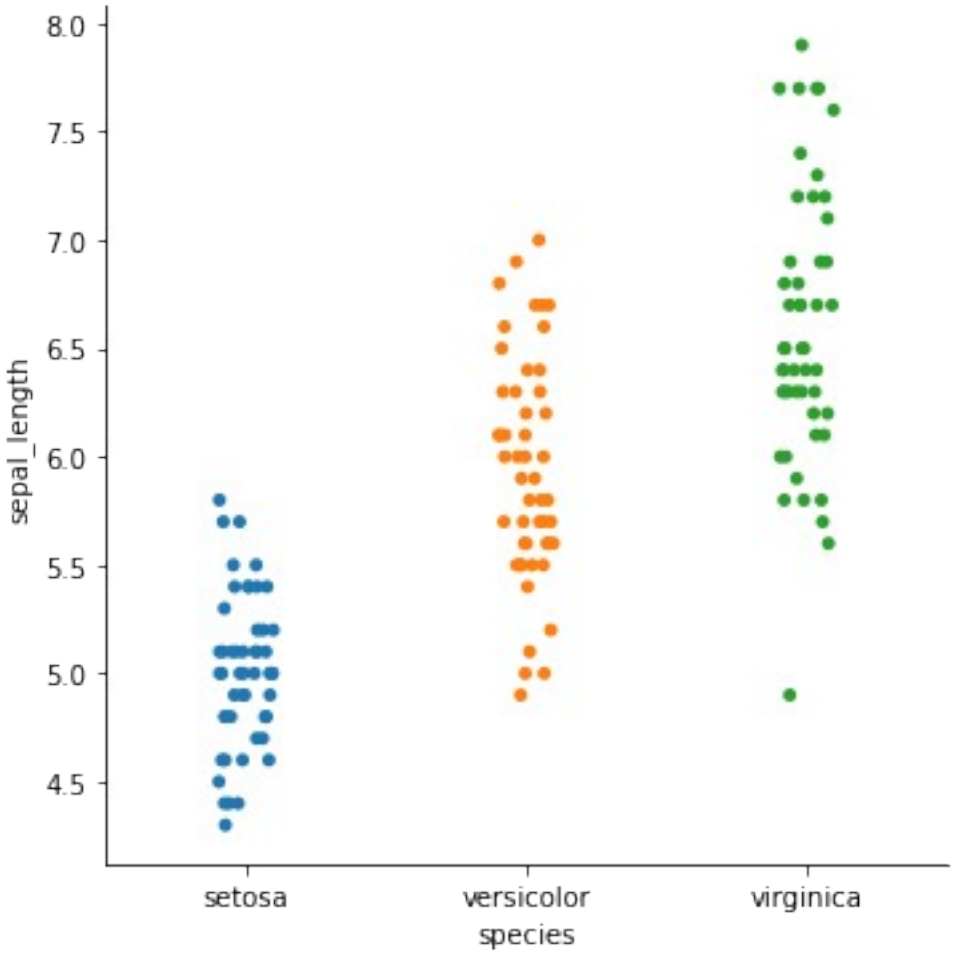
Faceting

Facet plots, where one subsets the data based on a categorical variable and makes a series of similar plots with the same scale.

We can make facet plots in Python in multiple ways. In this post, we will see an example of making simple facet plots using Seaborn's Catplot() method. It is mostly used to visualize when there is a numerical variable and a corresponding categorical variable.

```
# load data
irisdf = sns.load_dataset('iris')

# create catplot facetplot object
seaborn_facetgrid_object = sns.catplot(
    x='species',
    y='sepal_length',
    data=irisdf
)
# show plot
seaborn_facetgrid_object
```

```
<seaborn.axisgrid.FacetGrid at 0x7f8823b834c0>
```

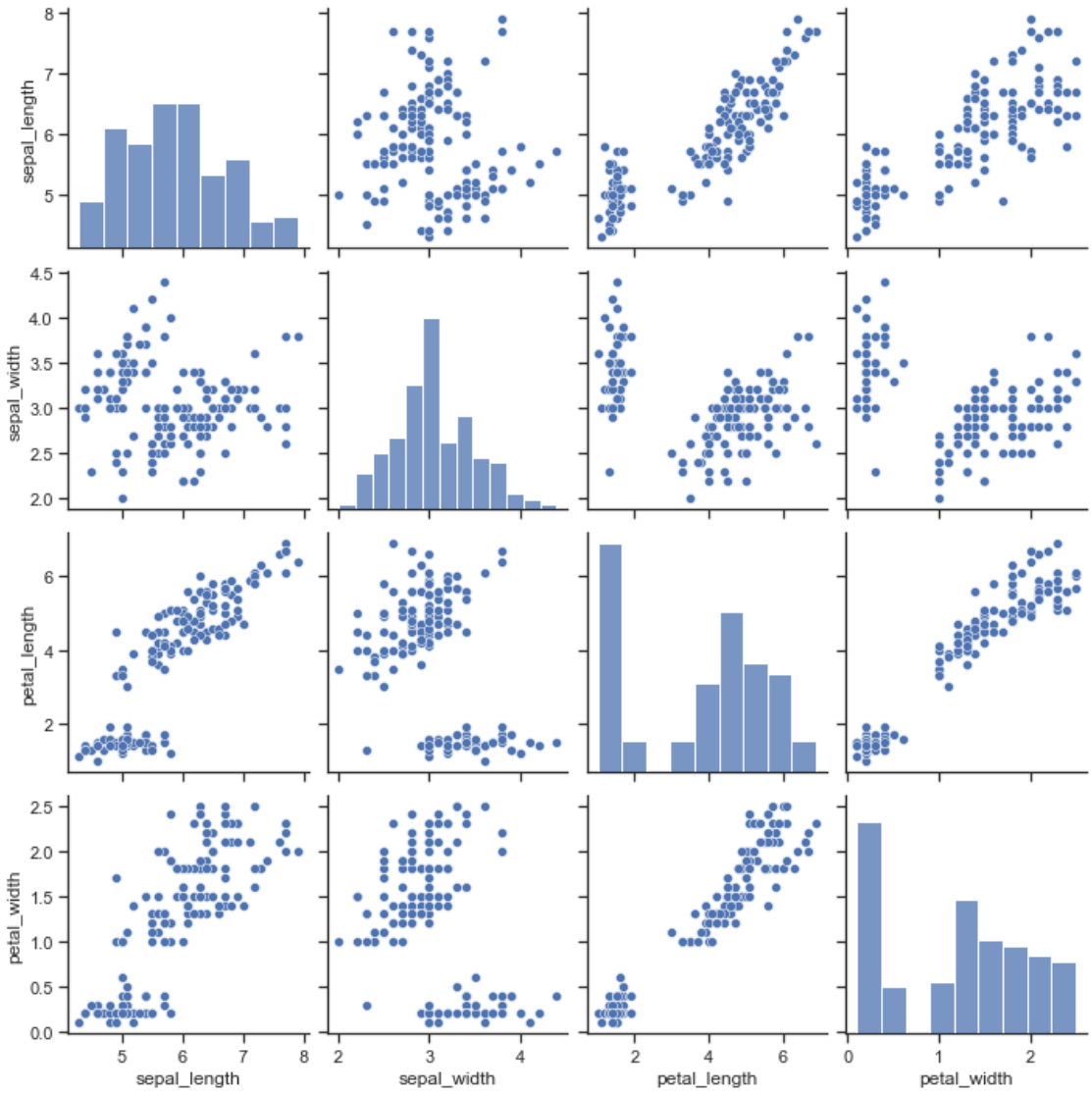


Pairplot

A pairplot is a pairwise relationships in a dataset. The pairplot function creates a grid of Axes such that each variable in data will be shared in the y-axis across a single row and in the x-axis across a single column.

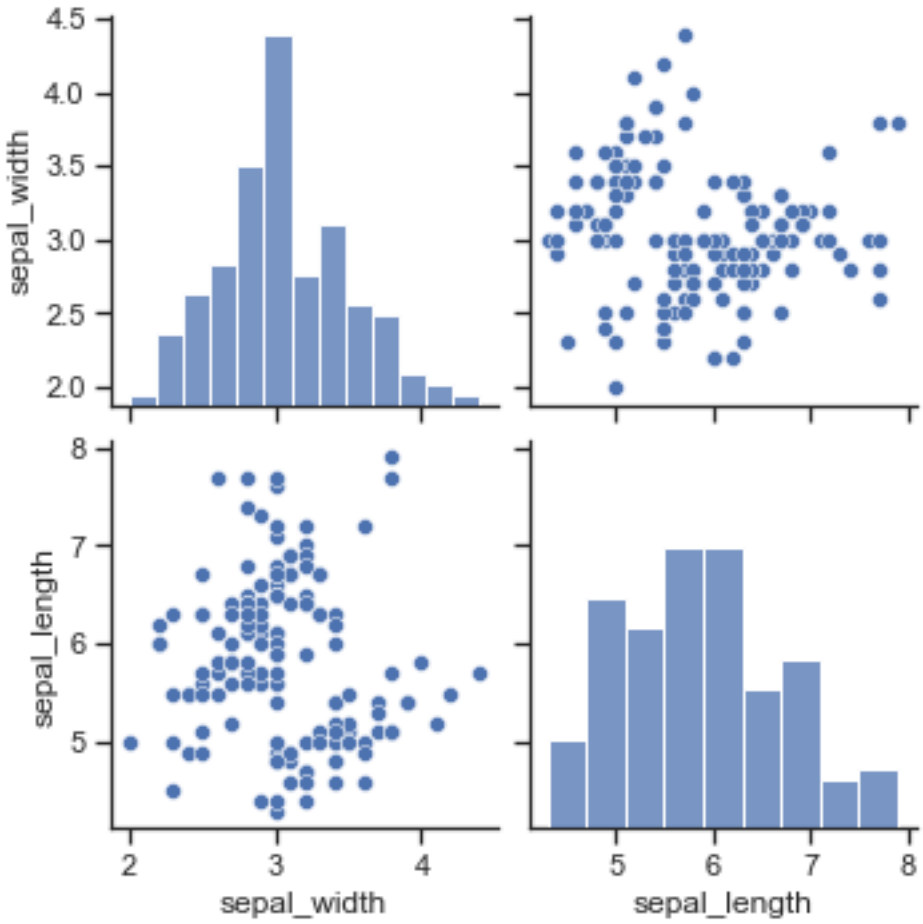
The data set has 4 measurements: sepal width, sepal length, petal_length and petal_width. The data contains measurements of different flowers.

```
sns.set(style="ticks", color_codes=True)
g = sns.pairplot(irisdf)
plt.show()
```



If you prefer a smaller plot, use less variables. For instance, if you only want sepal_width and sepal_length, tha would create a 2x2 plot.

```
g = sns.pairplot(irisdf, vars=["sepal_width", "sepal_length"])
```


4.5
4.0
3.5
3.0
2.5
2.0
sepal_width
8
7
6
5
sepal_length
2
3
4
sepal_width
5
6
7
8
sepal_length