assignment=25

July 29, 2023

1 Q1. Load the "titanic" dataset using the load_dataset function of seaborn. Use Plotly express to plot a scatter plot for age and fare columns in the titanic dataset.

`Titanic Dataset -`

It is one of the most popular datasets used for understanding machine learning basics. It contains information of all the passengers aboard the RMS Titanic, which unfortunately was shipwrecked. This dataset can be used to predict whether a given passenger survived or not.

[ ]:
```
#importing pandas library
import pandas as pd

#loading data
titanic = pd.read_csv('...\input\train.csv')
```

Seaborn: It is a python library used to statistically visualize data. Seaborn, built over Matplotlib, provides a better interface and ease of usage. It can be installed using the following command, pip3 install seaborn

Code: Printing data head

Features: The titanic dataset has roughly the following types of features:

Categorical/Nominal: Variables that can be divided into multiple categories but having no order or priority. Eg. Embarked (C = Cherbourg; Q = Queenstown; S = Southampton) Binary: A subtype of categorical features, where the variable has only two categories. Eg: Sex (Male/Female) Ordinal: They are similar to categorical features but they have an order(i.e can be sorted). Eg. Pclass (1, 2, 3) Continuous: They can take up any value between the minimum and maximum values in a column. Eg. Age, Fare Count: They represent the count of a variable. Eg. SibSp, Parch Useless: They don't contribute to the final outcome of an ML model. Here, PassengerId, Name, Cabin and Ticket might fall into this category. Code: Graphical Analysis

import seaborn as sns

[2]:
```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

[3]:
```
sns.set_style('whitegrid')
```

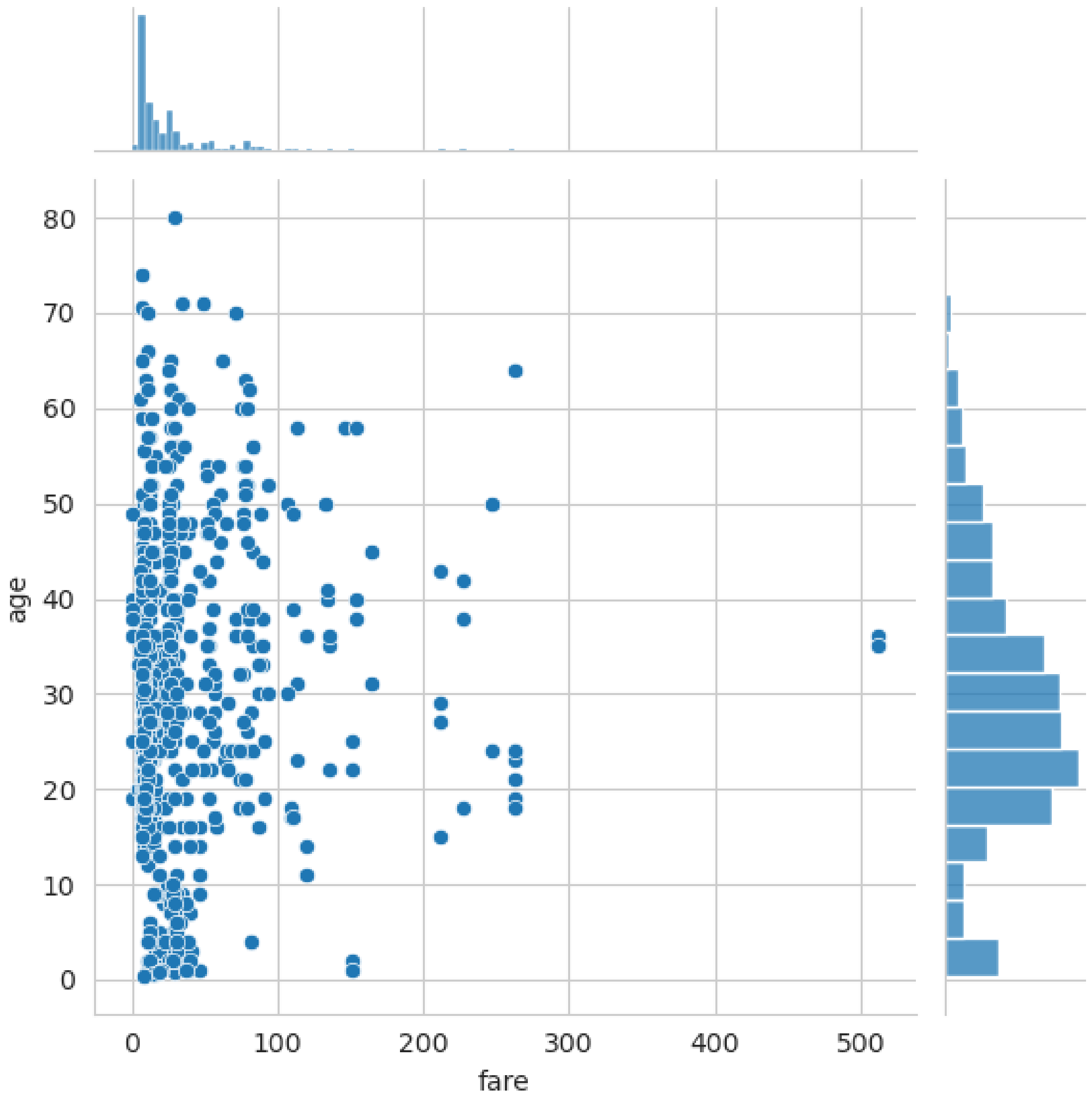
```
[4]: titanic = sns.load_dataset('titanic')
```

```
[9]: titanic.head()
```

```
[9]:    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
```

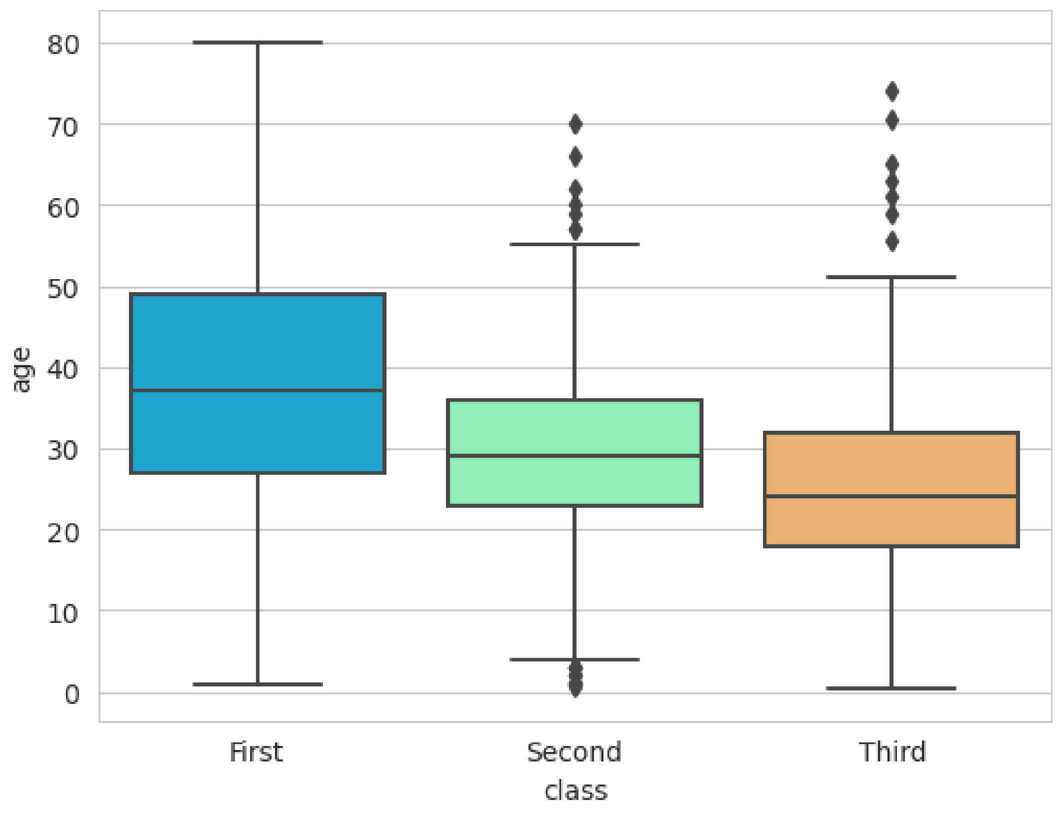
```
[10]: sns.jointplot(x='fare',y='age',data=titanic)
```

```
[10]: <seaborn.axisgrid.JointGrid at 0x7ff3fae21990>
```



```
[11]: sns.boxplot(x='class',y='age',data=titanic,palette='rainbow')
```

```
[11]: <AxesSubplot: xlabel='class', ylabel='age'>
```

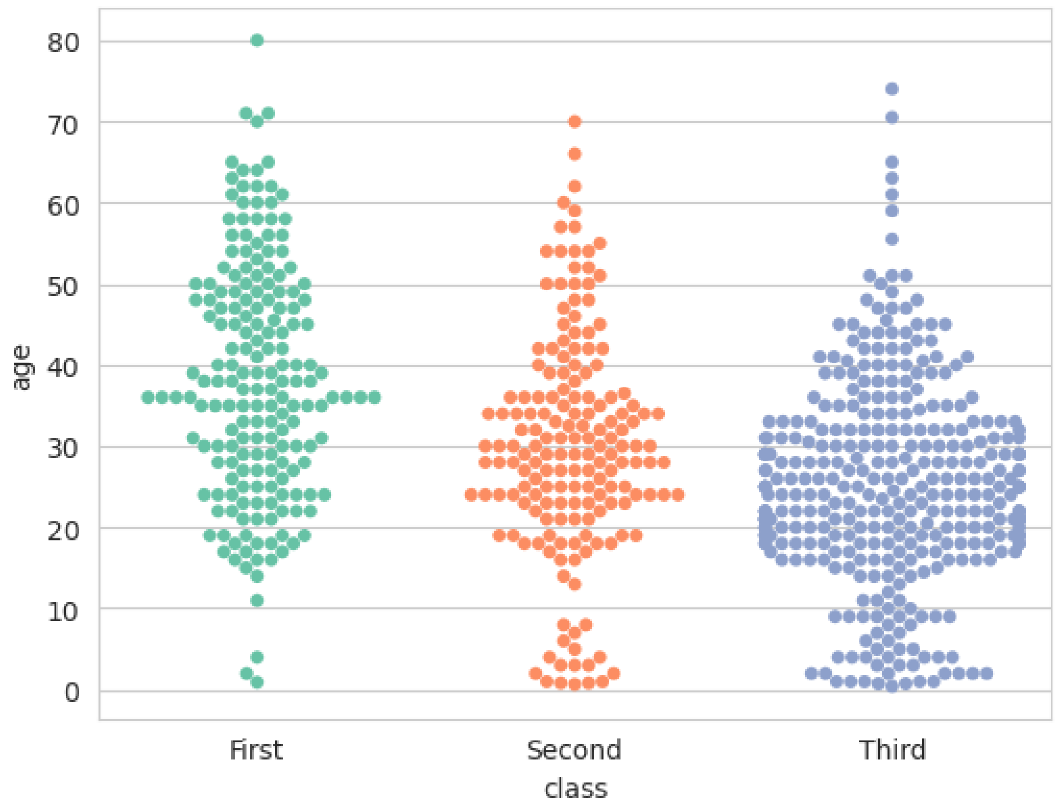


```
[12]: sns.swarmplot(x='class',y='age',data=titanic,palette='Set2')
```

```
/tmp/ipykernel_2724/3331474474.py:1: FutureWarning: Passing `palette` without
assigning `hue` is deprecated.
  sns.swarmplot(x='class',y='age',data=titanic,palette='Set2')
```

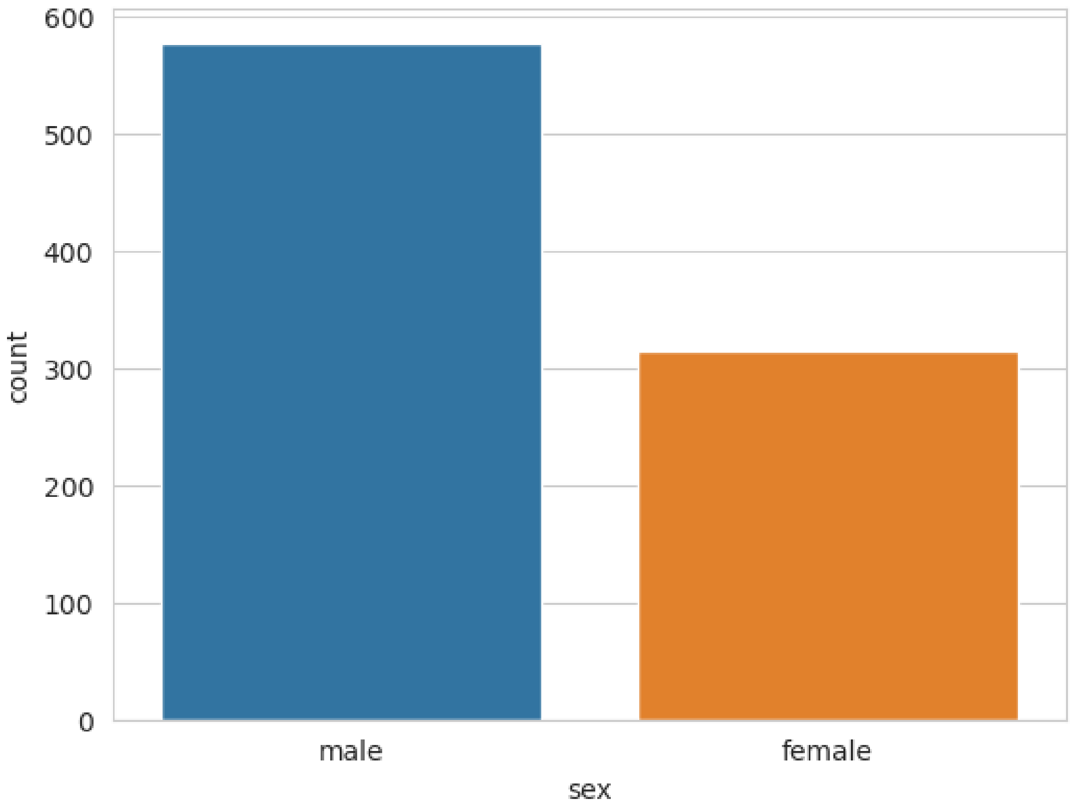
```
[12]: <AxesSubplot: xlabel='class', ylabel='age'>
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/categorical.py:3540:
UserWarning: 15.2% of the points cannot be placed; you may want to decrease the
size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



```
[13]: sns.countplot(x='sex',data=titanic)
```

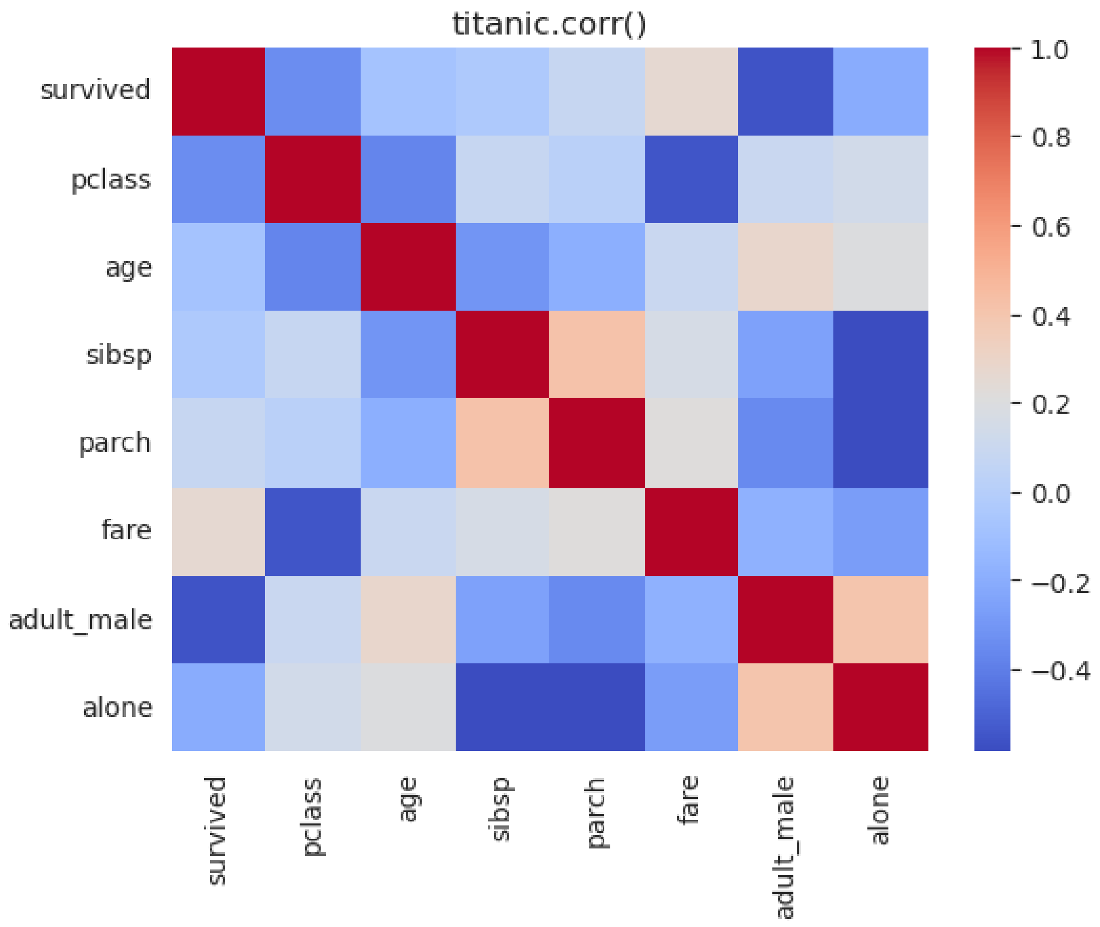
```
[13]: <AxesSubplot: xlabel='sex', ylabel='count'>
```



```
[14]: sns.heatmap(titanic.corr(),cmap='coolwarm')
      plt.title('titanic.corr()')
```

```
/tmp/ipykernel_2724/1112825032.py:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  sns.heatmap(titanic.corr(),cmap='coolwarm')
```

```
[14]: Text(0.5, 1.0, 'titanic.corr()')
```



2 Q2. Using the tips dataset in the Plotly library, plot a box plot using Plotly express.

Plotly is a library which helps create interactive, deployable and publication-ready graphs. The library allows graph building in multiple languages like Python, R, Matlab and Javascript. In this course we will learn the python implementations to building interactive plots using plotly.

Plotly library is very powerful and many features. Simple graphs, 3D plots, Dashboards, APIs, Animations etc. can be constructed using Plotly. In this course we will learn how to create simple graphs, 3D plots and some basic dashboards.

Plotly was initially written in javascript, and named plotly.js. Support for all other languages is in the form of respective APIs which process native code.

A note before we begin: Plotly is a very well documented library. All the example code (other than the exercises) you see in this course is taken from the Plotly documentation. Refer to Plotly documentation to learn more about various graphs available

Importing Plotly A simple import statement can be used to import plotly library. The version of the plotly library that is installed can be checked using ‘plotly.version’ statement

plotly.express.box(data_frame=None, x=None, y=None, color=None, facet_row=None, facet_col=None, facet_col_wrap=0, facet_row_spacing=None, facet_col_spacing=None, hover_name=None, hover_data=None, custom_data=None, animation_frame=None, animation_group=None, category_orders=None, labels=None, color_discrete_sequence=None, color_discrete_map=None, orientation=None, boxmode=None, log_x=False, log_y=False, range_x=None, range_y=None, points=None, notched=False, title=None, template=None, width=None, height=None) → plotly.graph_objects._figure.Figure In a box plot, rows of data_frame are grouped together into a box-and-whisker mark to visualize their distribution.

Each box spans from quartile 1 (Q1) to quartile 3 (Q3). The second quartile (Q2) is marked by a line inside the box. By default, the whiskers correspond to the box’ edges +/- 1.5 times the interquartile range (IQR: Q3-Q1), see “points” for other options.

Parameters data_frame (DataFrame or array-like or dict) – This argument needs to be passed for column names (and not keyword names) to be used. Array-like and dict are transformed internally to a pandas DataFrame. Optional: if missing, a DataFrame gets constructed under the hood using the other arguments.

x (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to position marks along the x axis in cartesian coordinates. Either x or y can optionally be a list of column references or array_likes, in which case the data will be treated as if it were ‘wide’ rather than ‘long’.

y (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to position marks along the y axis in cartesian coordinates. Either x or y can optionally be a list of column references or array_likes, in which case the data will be treated as if it were ‘wide’ rather than ‘long’.

color (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to assign color to marks.

facet_row (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to assign marks to facetted subplots in the vertical direction.

facet_col (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to assign marks to facetted subplots in the horizontal direction.

facet_col_wrap (int) – Maximum number of facet columns. Wraps the column variable at this width, so that the column facets span multiple rows. Ignored if 0, and forced to 0 if facet_row or a marginal is set.

facet_row_spacing (float between 0 and 1) – Spacing between facet rows, in paper units. Default is 0.03 or 0.0.7 when facet_col_wrap is used.

facet_col_spacing (float between 0 and 1) – Spacing between facet columns, in paper units Default is 0.02.

hover_name (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like appear in bold in the hover tooltip.

hover_data (str, or list of str or int, or Series or array-like, or dict) – Either a name or list of names of columns in data_frame, or pandas Series, or array_like objects or a dict with column names as keys, with values True (for default formatting) False (in order to remove this column from hover information), or a formatting string, for example ':.3f' or '|%a' or list-like data to appear in the hover tooltip or tuples with a bool or formatting string as first element, and list-like data to appear in hover as second element Values from these columns appear as extra data in the hover tooltip.

custom_data (str, or list of str or int, or Series or array-like) – Either name or list of names of columns in data_frame, or pandas Series, or array_like objects Values from these columns are extra data, to be used in widgets or Dash callbacks for example. This data is not user-visible but is included in events emitted by the figure (lasso selection etc.)

animation_frame (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to assign marks to animation frames.

animation_group (str or int or Series or array-like) – Either a name of a column in data_frame, or a pandas Series or array_like object. Values from this column or array_like are used to provide object-constancy across animation frames: rows with matching `animation_group`s will be treated as if they describe the same object in each frame.

category_orders (dict with str keys and list of str values (default {})) – By default, in Python 3.6+, the order of categorical values in axes, legends and facets depends on the order in which these values are first encountered in data_frame (and no order is guaranteed by default in Python below 3.6). This parameter is used to force a specific ordering of values per column. The keys of this dict should correspond to column names, and the values should be lists of strings corresponding to the specific display order desired.

labels (dict with str keys and str values (default {})) – By default, column names are used in the figure for axis titles, legend entries and hovers. This parameter allows this to be overridden. The keys of this dict should correspond to column names, and the values should correspond to the desired label to be displayed.

color_discrete_sequence (list of str) – Strings should define valid CSS-colors. When color is set and the values in the corresponding column are not numeric, values in that column are assigned colors by cycling through color_discrete_sequence in the order described in category_orders, unless the value of color is a key in color_discrete_map. Various useful color sequences are available in the plotly.express.colors submodules, specifically plotly.express.colors.qualitative.

color_discrete_map (dict with str keys and str values (default {})) – String values should define valid CSS-colors Used to override color_discrete_sequence to assign a specific colors to marks corresponding with specific values. Keys in color_discrete_map should be values in the column denoted by color. Alternatively, if the values of color are valid colors, the string 'identity' may be passed to cause them to be used directly.

orientation (str, one of 'h' for horizontal or 'v' for vertical.) – (default 'v' if x and y are provided and both continous or both categorical, otherwise 'v'`('h') if`x(y) is categorical and y`(x)` is continuous, otherwise 'v'`('h') if only`x(y) is provided)

boxmode (str (default 'group')) – One of 'group' or 'overlay' In 'overlay' mode, boxes are on drawn top of one another. In 'group' mode, boxes are placed beside each other.

log_x (boolean (default False)) – If True, the x-axis is log-scaled in cartesian coordinates.

log_y (boolean (default False)) – If True, the y-axis is log-scaled in cartesian coordinates.

range_x (list of two numbers) – If provided, overrides auto-scaling on the x-axis in cartesian coordinates.

range_y (list of two numbers) – If provided, overrides auto-scaling on the y-axis in cartesian coordinates.

points (str or boolean (default 'outliers')) – One of 'outliers', 'suspectedoutliers', 'all', or False. If 'outliers', only the sample points lying outside the whiskers are shown. If 'suspectedoutliers', all outlier points are shown and those less than 4*Q1-3*Q3 or greater than 4*Q3-3*Q1 are highlighted with the marker's 'outliercolor'. If 'outliers', only the sample points lying outside the whiskers are shown. If 'all', all sample points are shown. If False, no sample points are shown and the whiskers extend to the full range of the sample.

notched (boolean (default False)) – If True, boxes are drawn with notches.

title (str) – The figure title.

template (str or dict or plotly.graph_objects.layout.Template instance) – The figure template name (must be a key in plotly.io.templates) or definition.

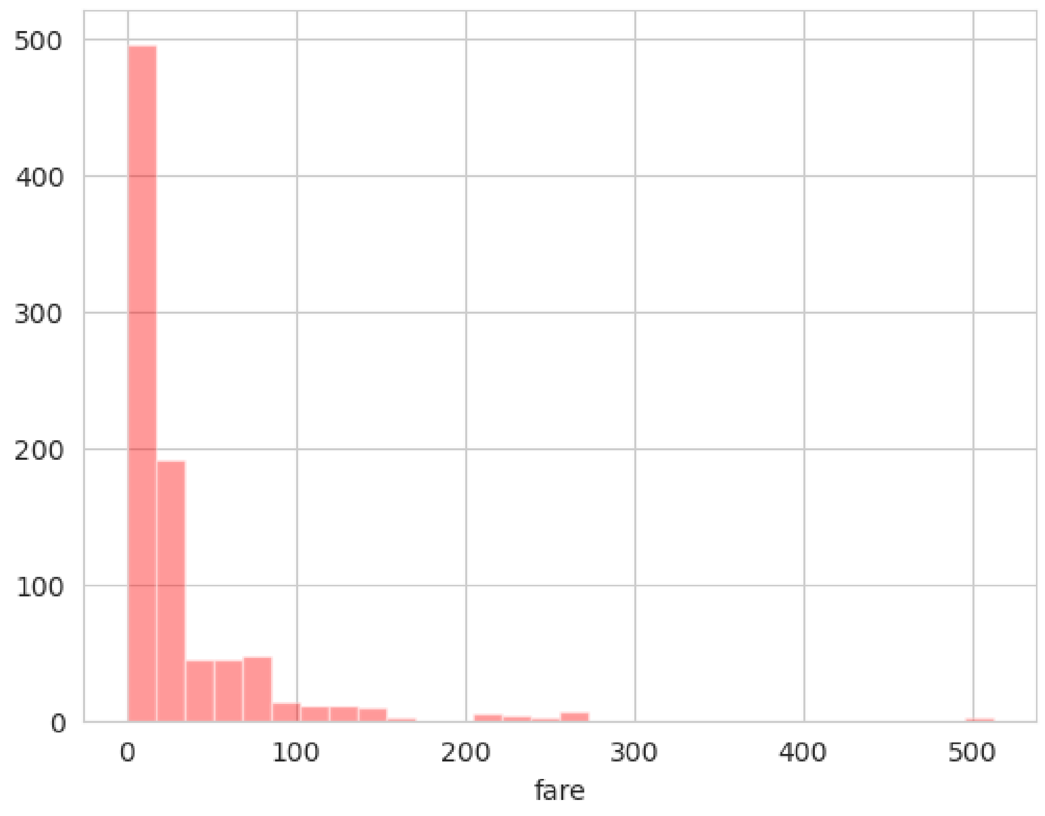
width (int (default None)) – The figure width in pixels.

height (int (default None)) – The figure height in pixels.

[15]:
```
sns.distplot(titanic['fare'],bins=30,kde=False,color='red')
```

```
/tmp/ipykernel_2724/2463549253.py:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(titanic['fare'],bins=30,kde=False,color='red')
```

[15]:
```
<AxesSubplot: xlabel='fare'>
```



3 Q3. Using the tips dataset in the Plotly library, Plot a histogram for x= "sex" and y="total_bill" column inthe tips dataset. Also, use the "smoker" column with the pattern_shape parameter and the "day"column with the color parameter.

```
[16]: import pandas as pd
```

```
[17]: orders = pd.read_table('http://bit.ly/chiporders')
```

```
[18]: user_cols = ['user_id', 'age', 'sex', 'smoker',"total_bill" "day" ]
      users = pd.read_table('http://bit.ly/movieusers', delimiter='|', header=None,␣
       ↪names=user_cols)
```

```
[19]: users.head()
```

```
[19]:    user_id  age sex      smoker total_billday
      0        1   24   M  technician        85711
      1        2   53   F       other        94043
      2        3   23   M      writer        32067
```

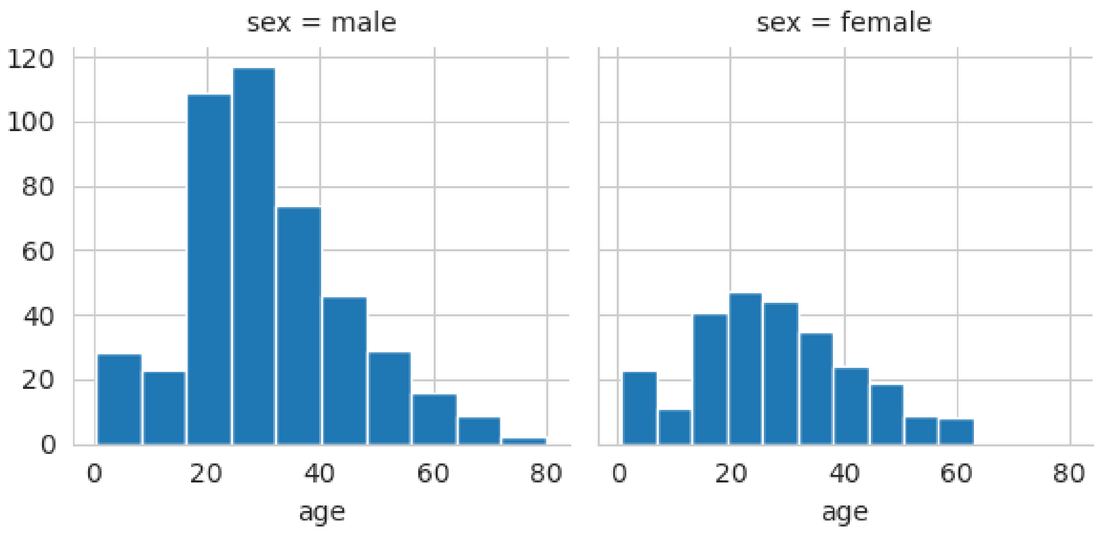
```
3          4   24   M  technician         43537
4          5   33   F       other         15213
```

g = sns.FacetGrid(data=titanic,col='sex') g.map(plt.hist,'age')

[20]:
```
g = sns.FacetGrid(data=titanic,col='sex')
g.map(plt.hist,'age')
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/axisgrid.py:745: FutureWarning:
iteritems is deprecated and will be removed in a future version. Use .items
instead.
  plot_args = [v for k, v in plot_data.iteritems()]
/opt/conda/lib/python3.10/site-packages/seaborn/axisgrid.py:745: FutureWarning:
iteritems is deprecated and will be removed in a future version. Use .items
instead.
  plot_args = [v for k, v in plot_data.iteritems()]
```

[20]:
```
<seaborn.axisgrid.FacetGrid at 0x7ff3f0c81f60>
```



4 Q4. Using the iris dataset in the Plotly library, Plot a scatter matrix plot, using the "species" column forthe color parameter.

Note: Use "sepal_length", "sepal_width", "petal_length", "petal_width" columns only with the dimensions parameter.

Scatter matrix with Plotly Express A scatterplot matrix is a matrix associated to n numerical arrays (data variables), $X_1, X_2, ..., X_n$, of the same length. The cell (i,j) of such a matrix displays the scatter plot of the variable Xi versus Xj.

Here we show the Plotly Express function px.scatter_matrix to plot the scatter matrix for the columns of the dataframe. By default, all columns are considered.

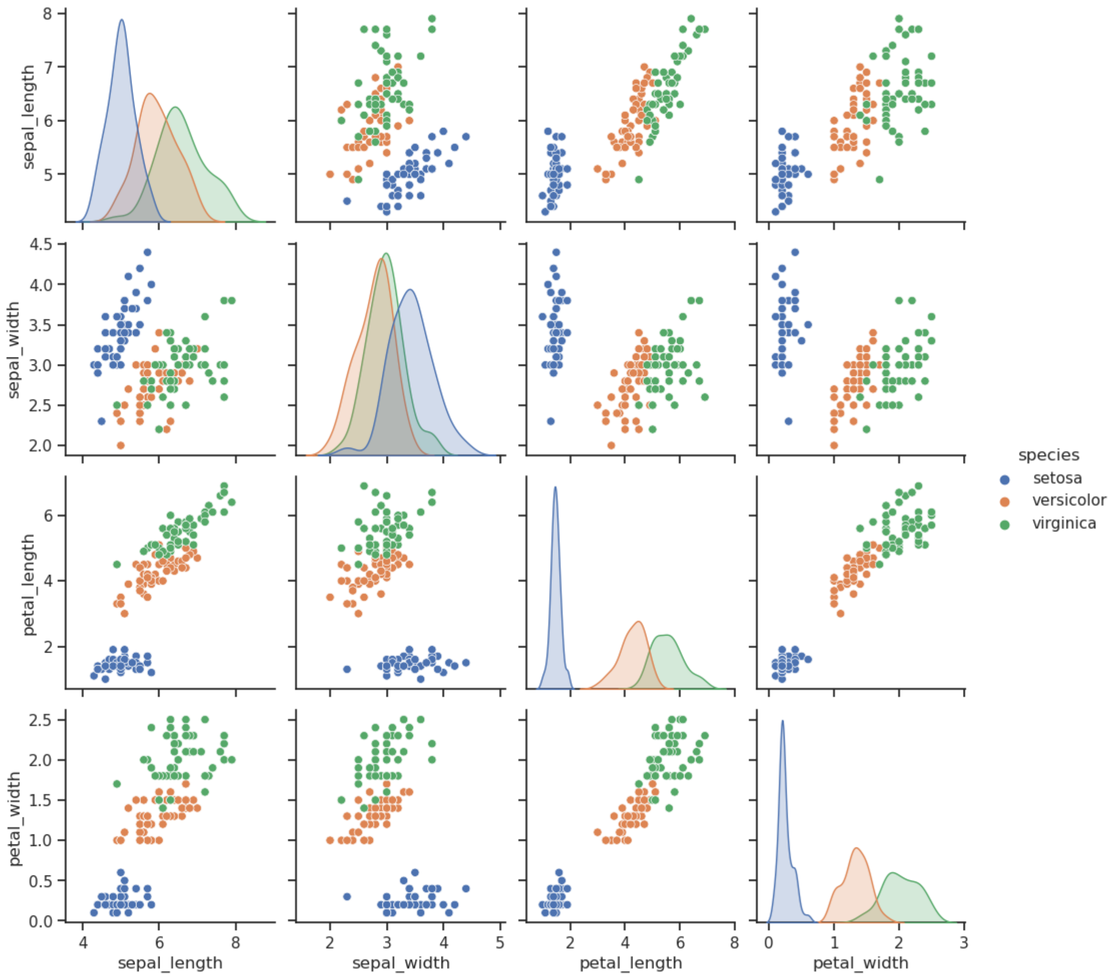
Plotly Express is the easy-to-use, high-level interface to Plotly, which operates on a variety of types of data and produces easy-to-style figures.

Plotly Express is the easy-to-use, high-level interface to Plotly, which operates on a variety of types of data and produces easy-to-style figures.

Like the 2D scatter plot px.scatter, the 3D function px.scatter_3d plots individual data in three-dimensional space

[33]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Load the Iris dataset from Seaborn
iris = sns.load_dataset('iris')

# Data preprocessing
# Separate features and target variable
X = iris.drop('species', axis=1)
y = iris['species']

# Data processing
# Perform any necessary data processing steps here

# Data analysis
# Calculate summary statistics
summary_stats = X.describe()
print("Summary Statistics:")
print(summary_stats)

# Plot the graph using Seaborn and Matplotlib
sns.set(style="ticks")
sns.pairplot(iris, hue="species")
plt.show()
```

```
Summary Statistics:
       sepal_length  sepal_width  petal_length  petal_width
count    150.000000   150.000000    150.000000   150.000000
mean       5.843333     3.057333      3.758000     1.199333
std        0.828066     0.435866      1.765298     0.762238
min        4.300000     2.000000      1.000000     0.100000
25%        5.100000     2.800000      1.600000     0.300000
50%        5.800000     3.000000      4.350000     1.300000
75%        6.400000     3.300000      5.100000     1.800000
max        7.900000     4.400000      6.900000     2.500000
```



`Conclusion:`

In conclusion, this article has demonstrated the process of plotting graphs for the Iris dataset using Seaborn and Matplotlib. By utilizing Seaborn's pairplot function, we were able to visualize the relationships between various features and the Iris flower species.

Through data preprocessing and analysis, we gained valuable insights into the dataset. The combination of Seaborn and Matplotlib provided us with powerful tools for creating visually appealing and informative graphs.

5 Q5. What is Distplot? Using Plotly express, plot a distplot.

Distplots The distplot figure factory displays a combination of statistical representations of numerical data, such as histogram, kernel density estimation or normal curve, and rug plot.

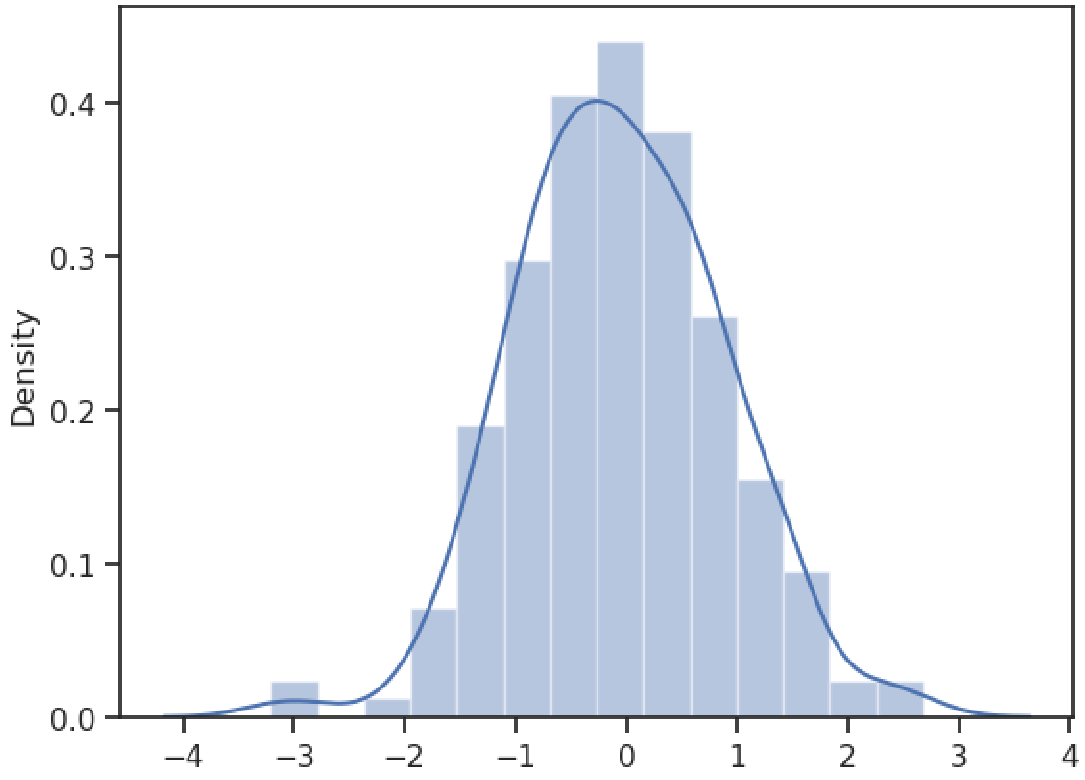
The distplot can be composed of all or any combination of the following 3 components −

histogram curve: (a) kernel density estimation or (b) normal curve, and rug plot The figure_factory module has create_distplot() function which needs a mandatory parameter called hist_data.

Following code creates a basic distplot consisting of a histogram, a kde plot and a rug plot.

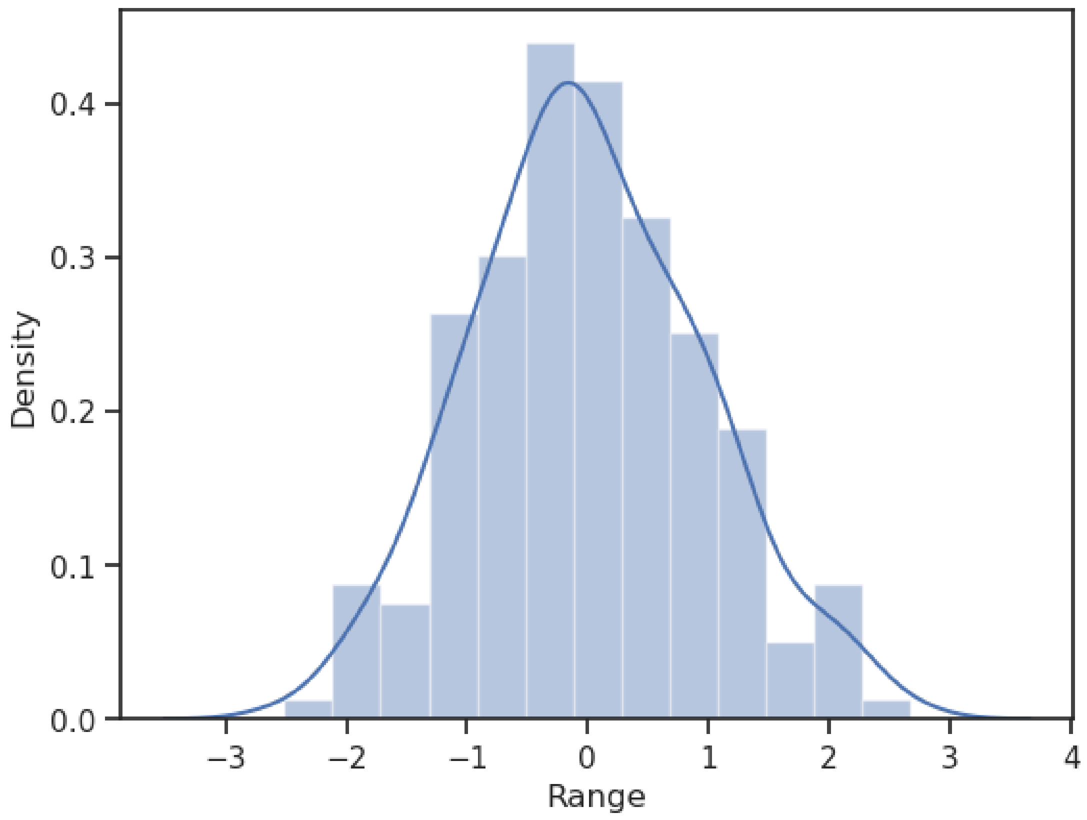
[37]:
```
import numpy as np
import seaborn as sn
import matplotlib.pyplot as plt

data = np.random.randn(200)
res = sn.distplot(data)
plt.show()
```

```
/tmp/ipykernel_5148/3383672016.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  res = sn.distplot(data)
```



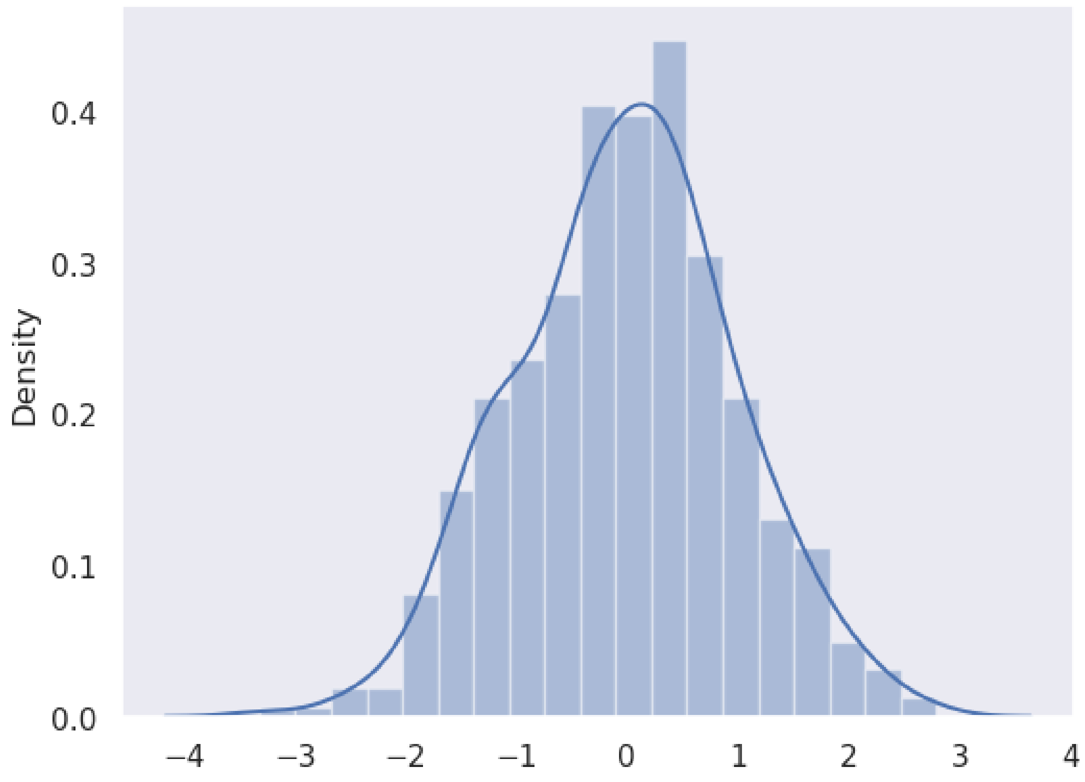
```
[40]: import numpy as np
      import seaborn as sn
      import matplotlib.pyplot as plt

      data = np.random.randn(200)
      res = pd.Series(data,name="Range")
      plot = sn.distplot(res)
      plt.show()
```

```
/tmp/ipykernel_5148/3223083677.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  plot = sn.distplot(res)
```



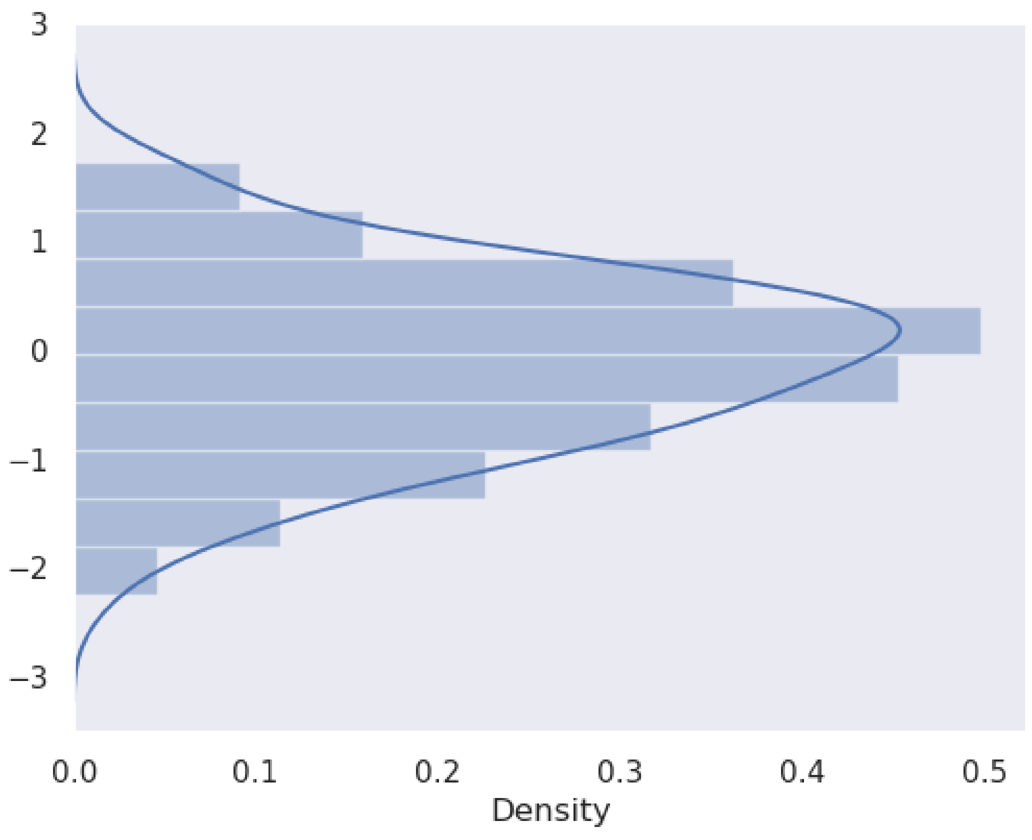
```
[41]: import numpy as np
      import seaborn as sn
      import matplotlib.pyplot as plt
      sn.set(style='dark',)
      data = np.random.randn(500)

      plot = sn.distplot(data)

      plt.show()
```

```
/tmp/ipykernel_5148/4122712530.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  plot = sn.distplot(data)
```



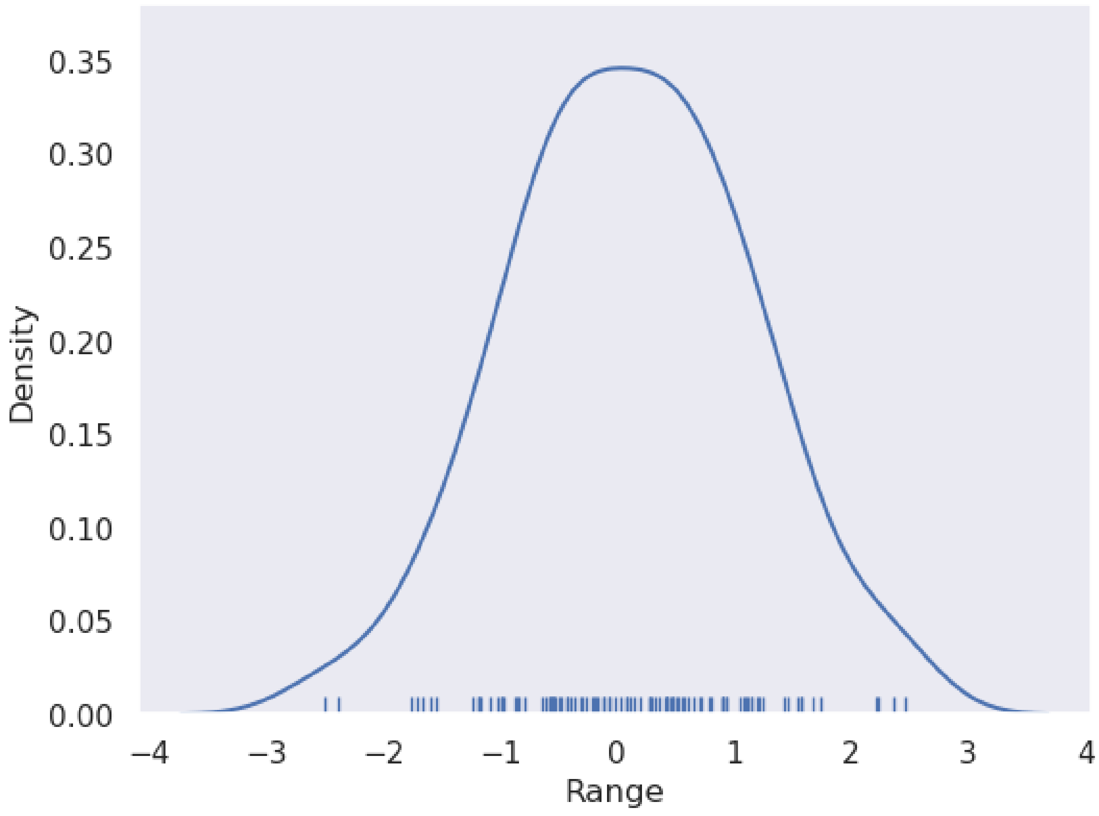
```
[42]: import numpy as np
      import seaborn as sn
      import matplotlib.pyplot as plt

      data = np.random.randn(100)

      plot = sn.distplot(data,vertical=True)

      plt.show()
```

```
/tmp/ipykernel_5148/3207764075.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  plot = sn.distplot(data,vertical=True)
```



[43]:
```
import numpy as np
import seaborn as sn
import matplotlib.pyplot as plt

data = np.random.randn(100)
res = pd.Series(data,name="Range")
plot = sn.distplot(res,rug=True,hist=False)
plt.show()
```

```
/tmp/ipykernel_5148/3714464782.py:7: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `kdeplot` (an axes-level function for kernel density
plots).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  plot = sn.distplot(res,rug=True,hist=False)
```



```
[ ]:
```