1.Relational plots : This plot is used to understand the realtion between two variables.

2.Categorical polots : This plot deals with categrical variables and how they can be visulized.

3.Distribution plots : used for examing unvirante and bivariate distrubtions.

4.Marks plots : matrix plot is an array of scatterplots.

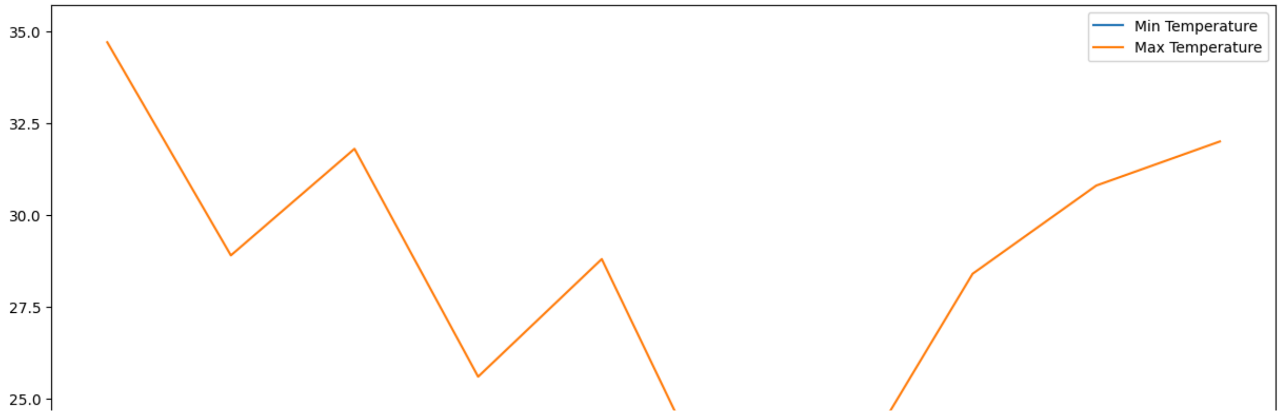
5.Regression plots:The regression plots in seaborn are primarily intended to add a visual guide.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
import seaborn as sns
%matplotlib inline
```

```
#Simple plotting with seaborn #Data
dates=['1981-01-01','1981-01-02','1982-01-03','1981-01-04','1981-01-05','1981-01-06','1
min_temperature=[20.7,17.9,18.8,14.6,15.8,15.8,15.8,17.4,21.8,20.0]
max_temperature=[34.7,28.9,31.8,25.6,28.8,21.8,22.8,28.4,30.8,32.0]
```

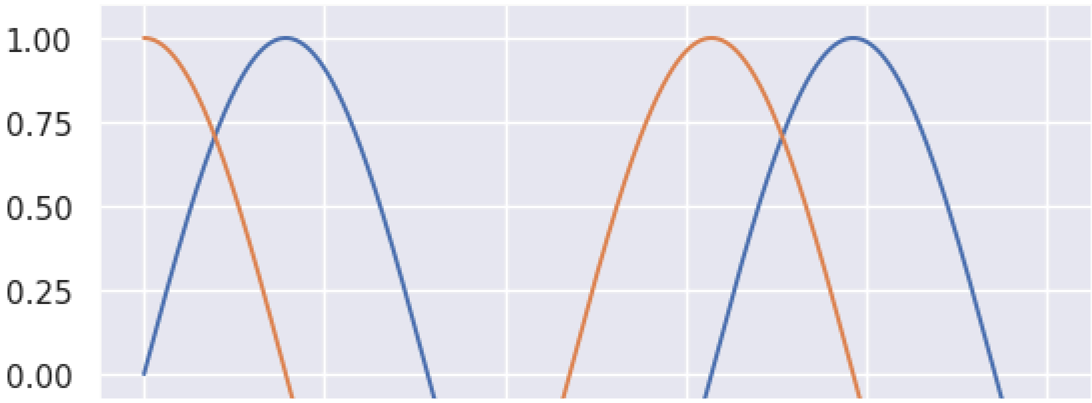
```
#plotting
fig,axes=plt.subplots(nrows=1,ncols=1,figsize=(15,10))
axes.plot(dates,min_temperature,label='Min Temperature')
axes.plot(dates,max_temperature,label='Max Temperature')
axes.legend()
```

```
<matplotlib.legend.Legend at 0x7b79a750d4b0>
```

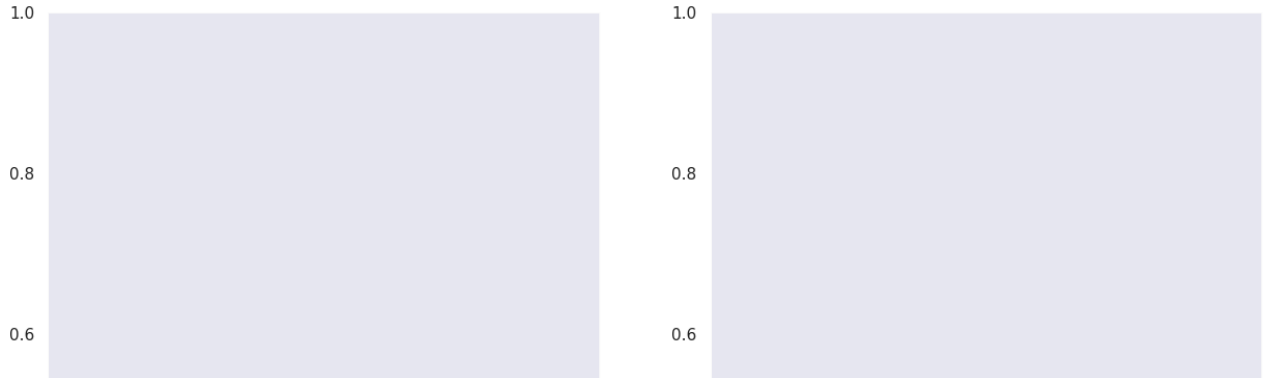


```
#seaborn style as the default matplotlib style
sns.set()
```

```
#simple sine plot
x = np.linspace(0,10,1000)
plt.plot(x,np.sin(x),x,np.cos(x));
```



```
# Relational Plots
#Line plot : it is one of the most basic plot in seaborn liberary
#This plot is mainly used to visualize the data in form of some time series, i.,e,in
sns.set(style="dark")
fig, ax=plt.subplots(ncols=2, nrows=1, figsize=(15,10))
```



```
#Loading data with seaborn
df=sns.load_dataset("tips")
print(df.head)
```

```
<bound method NDFrame.head of      total_bill   tip     sex smoker   day    t
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

[244 rows x 7 columns]>
```

```
#Lineplot
sns.lineplot(x="total_bill", y="tip", hue="size", style="time", data=df, ax=ax[0])
ax[0].set_title('Line Plot')
```

```
Text(0.5, 1.0, 'Line Plot')
```

```
#Scatterplot
sns.scatterplot(x="total_bill", y="tip", style="time", data=df, ax=ax[1])
ax[1].set_title('Scatter Plot')
```

```
Text(0.5, 1.0, 'Scatter Plot')
```

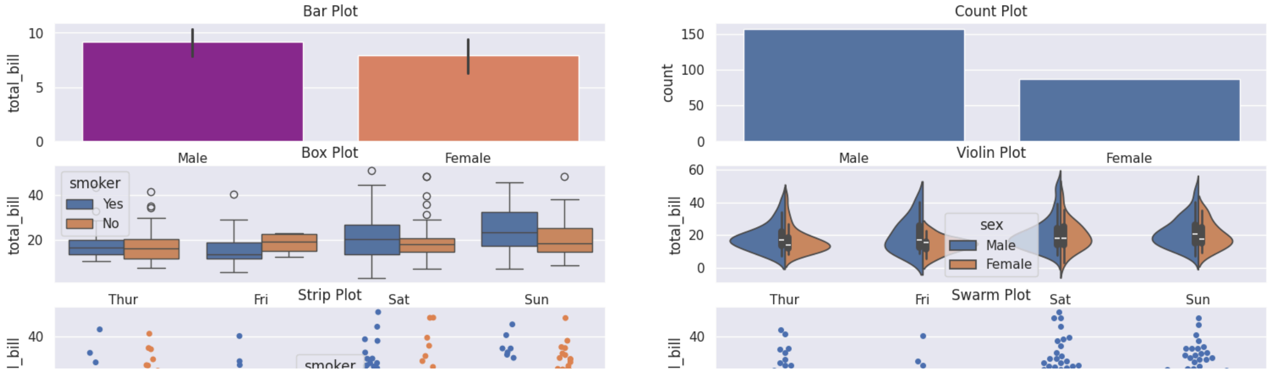
```
# Define the figure object
fig = plt.gcf()
```

```
<Figure size 640x480 with 0 Axes>
```

```
# Saving Plot
fig.savefig('Scatter_plot1.png')
print('Plot Saved')
```

```
Plot Saved
```

```
#Categorical Plot
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.pyplot import figure
import seaborn as sns


sns.set_style('darkgrid')
fig,ax=plt.subplots(nrows=5,ncols=2)
fig.set_size_inches(18.5,10.5)
df=sns.load_dataset('tips')
sns.barplot(x='sex',y='total_bill',data=df,palette='plasma',estimator=np.std,ax=ax[0,0]
sns.countplot(x='sex',data=df,ax=ax[0,1]).set_title('Count Plot')
sns.boxplot(x='day',y='total_bill',data=df,hue='smoker',ax=ax[1,0]).set_title('Box Plot
sns.violinplot(x='day',y='total_bill',data=df,hue='sex',split =True,ax=ax[1,1]).set_tit
sns.stripplot(x='day',y='total_bill',data=df, jitter=True, hue='smoker', dodge=True, ax
sns.swarmplot(x='day',y='total_bill',data=df,ax=ax[2,1]).set_title('Swarm Plot')
sns.violinplot(x='day',y='total_bill',data=df,ax=ax[3,0])
sns.swarmplot(x='day',y='total_bill',data=df,color='black',ax=ax[3,0]).set_title('Combi
<ipython-input-14-83dbc4e47270>:5: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed i

  sns.barplot(x='sex',y='total_bill',data=df,palette='plasma',estimator=np.st
Text(0.5, 1.0, 'Combined Plot')
```



```
#Density plot
sns.scatterplot(x='day',y='total_bill',data=df,color='black',ax=ax[3,1])
```

```
<Axes: xlabel='day', ylabel='total_bill'>
```

```
#boxplot
sns.boxenplot(x="day",y="total_bill",color="b",scale="linear",data=df,ax=ax[4,0])
```

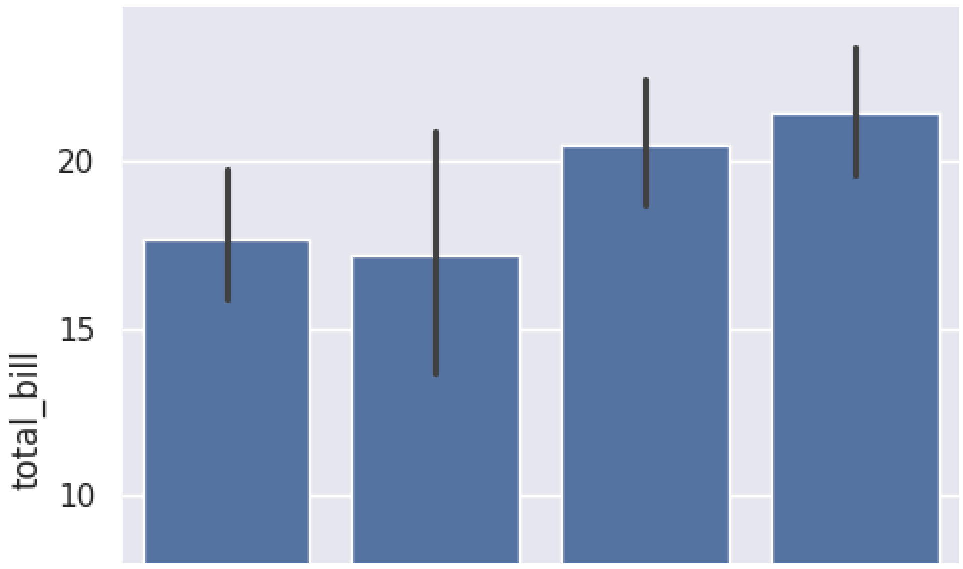
```
<ipython-input-16-2d719fdc8517>:2: FutureWarning:

The `scale` parameter has been renamed to `width_method` and will be removed
  sns.boxenplot(x="day",y="total_bill",color="b",scale="linear",data=df,ax=ax
<Axes: xlabel='day', ylabel='total_bill'>
```

```
#Ridgeplot
sns.pointplot(x="day",y="total_bill",color="b",hue="sex",data=df,ax=ax[4,1])
```

```
<ipython-input-17-fc4e90deedd0>:2: FutureWarning:

Setting a gradient palette using color= is deprecated and will be removed in

  sns.pointplot(x="day",y="total_bill",color="b",hue="sex",data=df,ax=ax[4,1]
<Axes: xlabel='day', ylabel='total_bill'>
```

```
#catplot
sns.catplot(x='day',y='total_bill',data=df,kind='bar')
```

```
<seaborn.axisgrid.FacetGrid at 0x7b797109f8e0>
```



Distribution plots:In seaborn is used for examining univariate and bivariate distributions

Four main types of plots

1)Joinplot,

2)distplot,

3)pairplot,

4)rugplot.

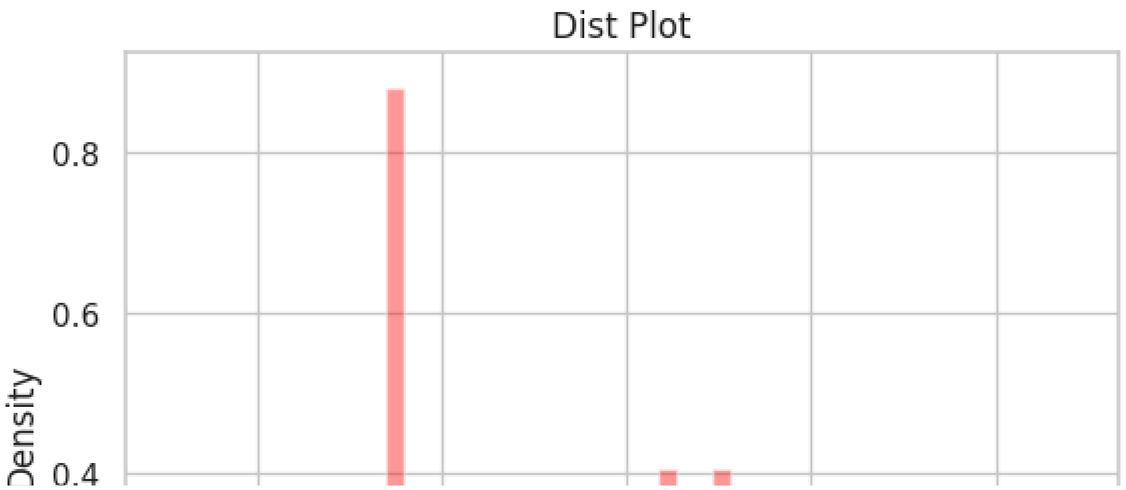
```
sns.set_style('whitegrid')
```

```
#Data- 'iris'
df=sns.load_dataset('iris')
print(df.head())
```

```
   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
```

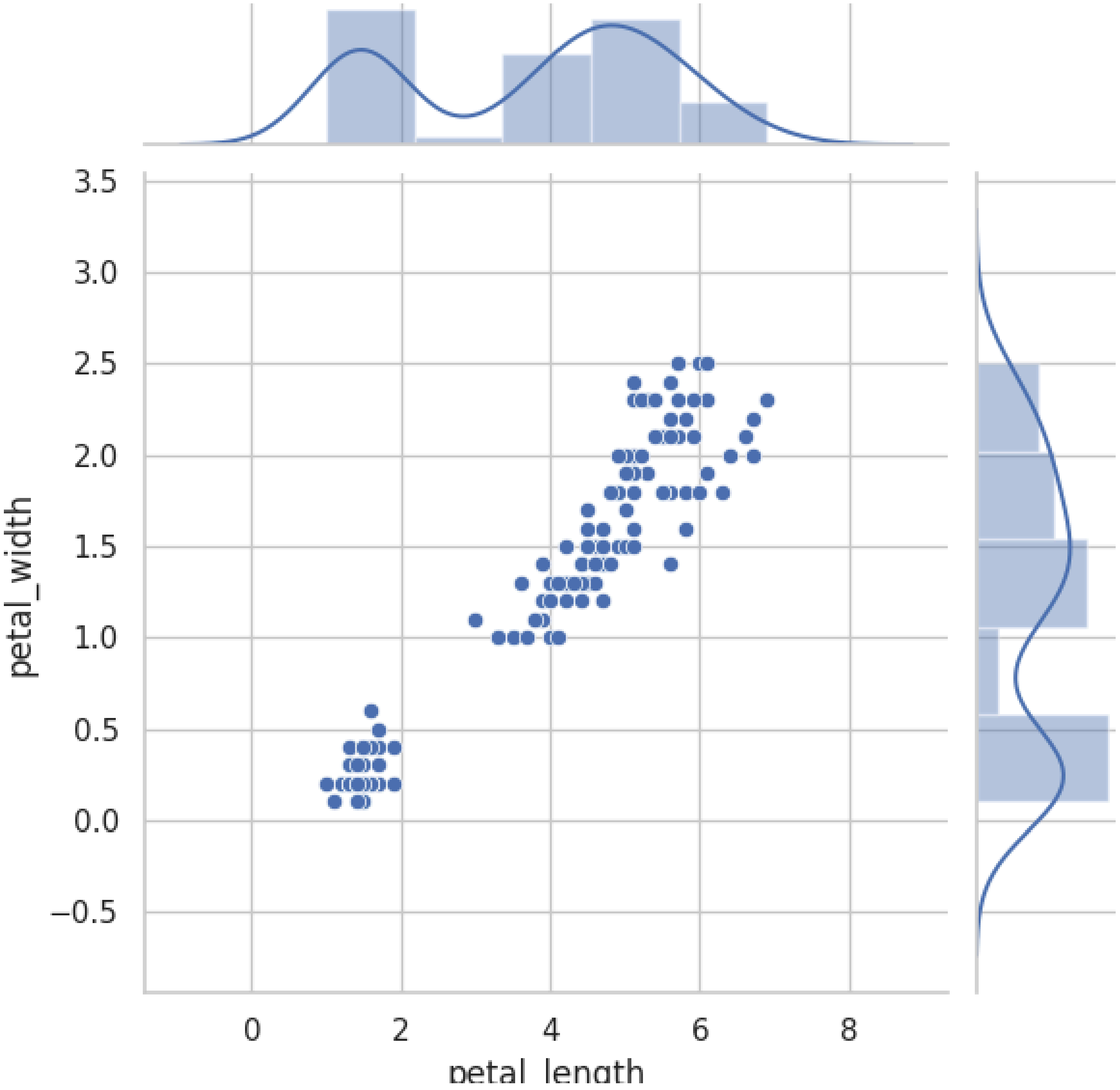
```
#Distplot
sns.distplot(df['petal_length'],kde=True,color='red',
             bins=30).set_title('Dist Plot')
```

```
<ipython-input-21-a5c29d4f21af>:2: 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(df['petal_length'],kde=True,color='red',
Text(0.5, 1.0, 'Dist Plot')
```



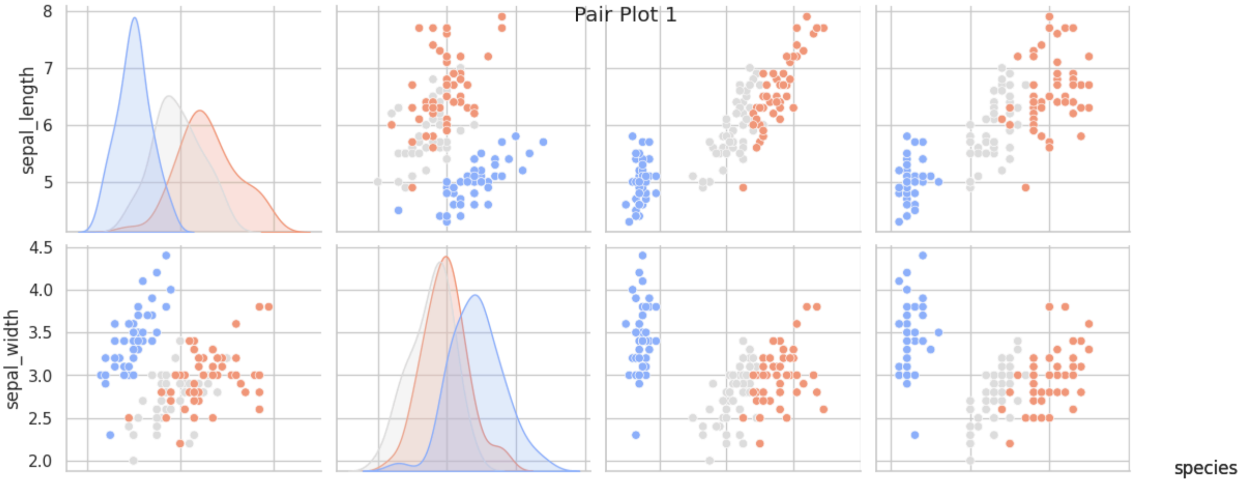
```
#Joinplot
jointgrid=sns.JointGrid(x='petal_length',y='petal_width',data=df)
jointgrid.plot_joint(sns.scatterplot)
jointgrid.plot_marginals(sns.distplot)
g=sns.jointplot(x='petal_length',y='petal_width',data=df,kind='hex')
g.fig.suptitle('Joint Plot')
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/axisgrid.py:1886: 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

  func(self.x, **orient_kw_x, **kwargs)
/usr/local/lib/python3.10/dist-packages/seaborn/axisgrid.py:1892: 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

  func(self.y, **orient_kw_y, **kwargs)
Text(0.5, 0.98, 'Joint Plot')
```

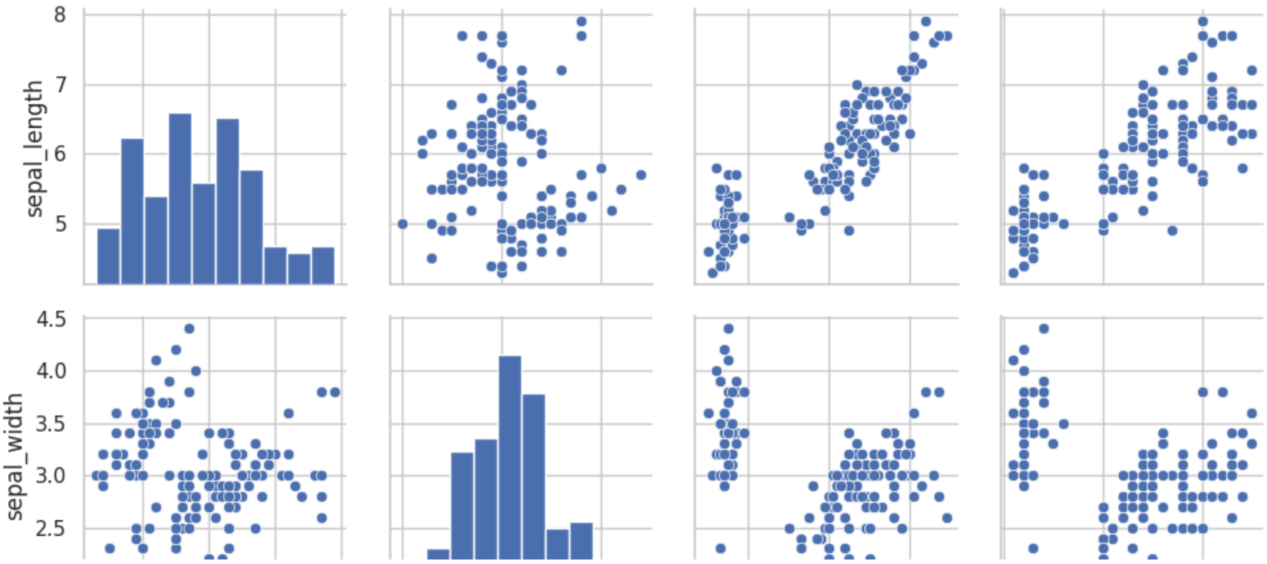


```
#Pair plot
g=sns.pairplot(df,hue="species",palette='coolwarm')
g.fig.suptitle("Pair Plot 1")
g.add_legend()
```

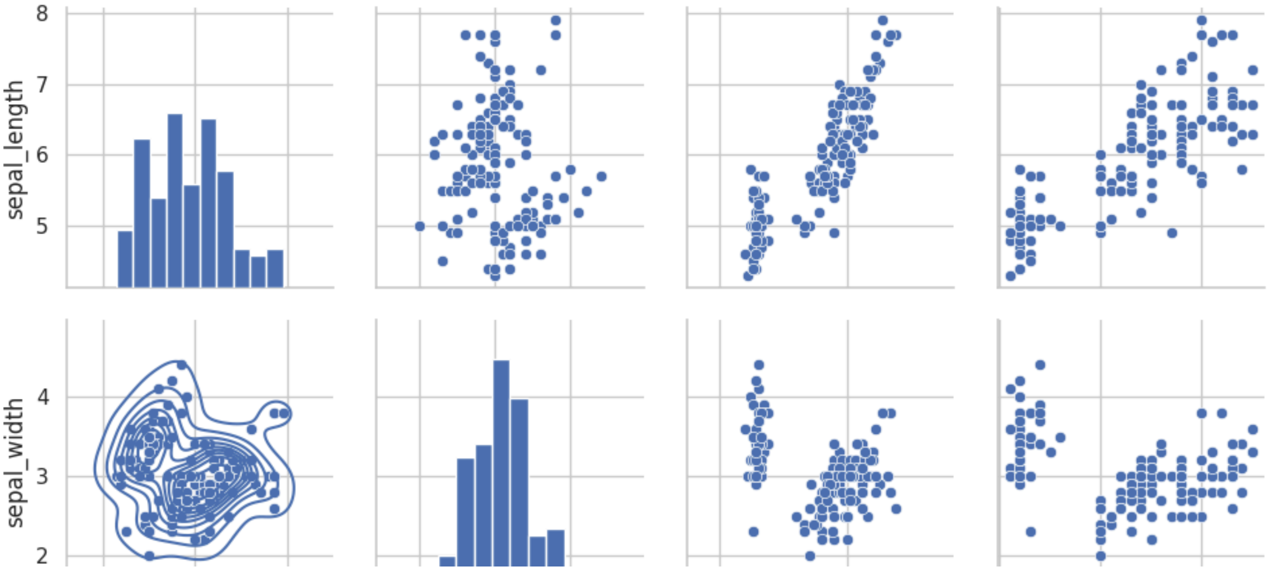
```
<seaborn.axisgrid.PairGrid at 0x7b796a3271f0>
```



```
#Pair Grid
pairgrid=sns.PairGrid(data=df)
pairgrid=pairgrid.map_offdiag(sns.scatterplot)
pairgrid=pairgrid.map_diag(plt.hist)
```

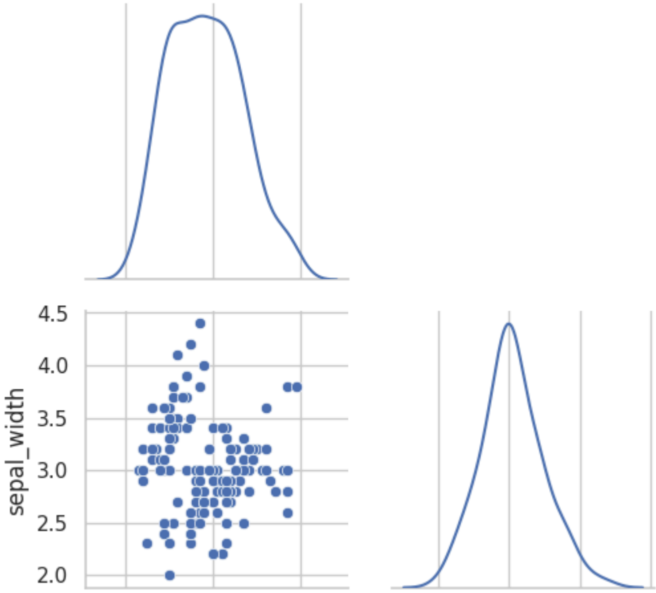


```
#Different kinds
pairgrid=sns.PairGrid(data=df)
pairgrid=pairgrid.map_offdiag(sns.scatterplot)
pairgrid=pairgrid.map_diag(plt.hist)
pairgrid=pairgrid.map_lower(sns.kdeplot)
```

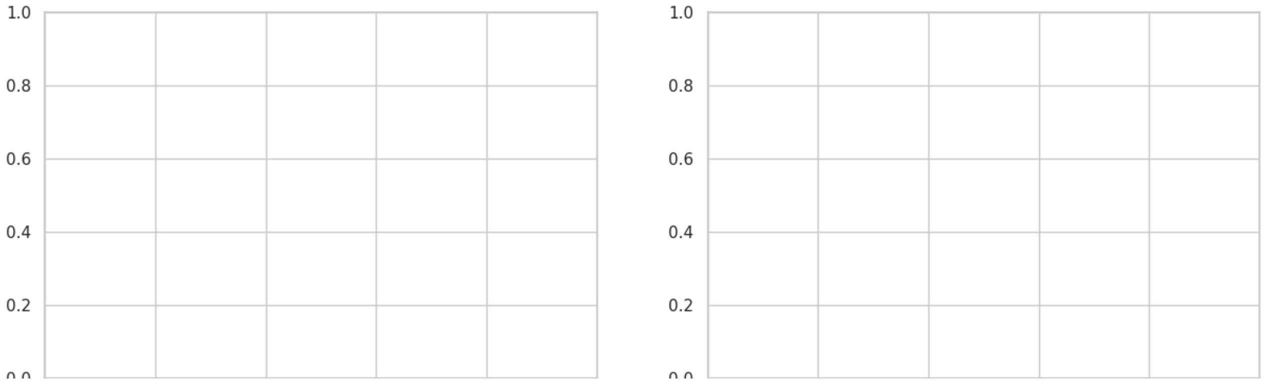


```
#Avoid Redundancy
g=sns.PairGrid(df,diag_sharey=False,corner=True)
g.map_lower(sns.scatterplot)
g.map_diag(sns.kdeplot)
```

```
<seaborn.axisgrid.PairGrid at 0x7b7968b3b9d0>
```



```
#Matrix Plot
fig, ax=plt.subplots(nrows=2,ncols=2, figsize=(15,10))
```



```
#Data
df1=sns.load_dataset('flights')
df2=sns.load_dataset('iris')
df11=pd.pivot_table(values='passengers',index='month',columns='year',data=df1)


#Calculates correlations between columns in the dataframe
df1_numeric = df1.select_dtypes(include=[np.number])
dfc1 = df1_numeric.corr()
df1['month'] = pd.to_numeric(df1['month'], errors='coerce')
dfc1 = df1.corr()


#Heatmaps-matrix plot
sns.heatmap(df11,cmap='Y1GnBu',linecolor='r',linewudth=0.5,annot=True,fmt='d',square=Tru
            ax=ax[0,0]).set_title('Heat Map Flights')
sns.heatmap(dfc2,cmap='coolwarm',linecolor='r',linewudth=1,annot=True,fmt='d',square=Tru
            ax=ax[0,1]).set_title('Heat Map Iris')
```

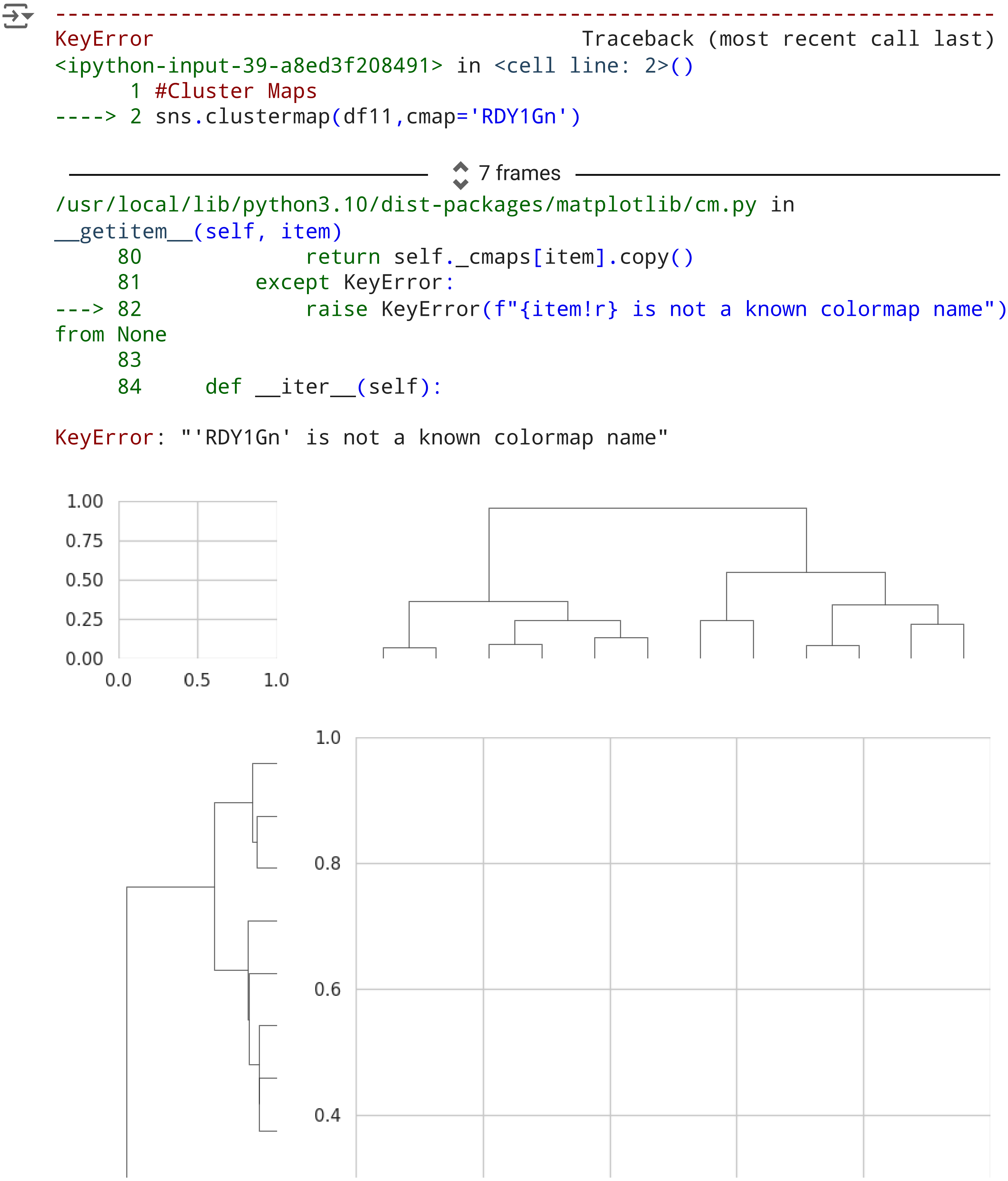
```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-35-9472c08496cd> in <cell line: 2>()
      1 #Heatmaps-matrix plot
----> 2
sns.heatmap(df11,cmap='Y1GnBu',linecolor='r',linewudth=0.5,annot=True,fmt='d'
      3               ax=ax[0,0]).set_title('Heat Map Flights')
      4
sns.heatmap(dfc2,cmap='coolwarm',linecolor='r',linewudth=1,annot=True,fmt='d'
      5               ax=ax[0,1]).set_title('Heat Map Iris')

                                  4 frames
/usr/local/lib/python3.10/dist-packages/matplotlib/cm.py in
__getitem__(self, item)
     80               return self._cmaps[item].copy()
     81           except KeyError:
---> 82               raise KeyError(f"{item!r} is not a known colormap name")
from None
     83
     84     def __iter__(self):
```

```
#Lower traingle
mask1=np.trii(dfc2)
sns.heatmap(dfc2,annot=True, mask=mask1,ax=ax[0,1],cmap='coolwarm').set_title('Heat Map
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-37-a0d86dc34c08> in <cell line: 2>()
      1 #Lower traingle
----> 2 mask1=np.trii(dfc2)
      3 sns.heatmap(dfc2,annot=True,
mask=mask1,ax=ax[0,1],cmap='coolwarm').set_title('Heat Map Lower Triangle')

/usr/local/lib/python3.10/dist-packages/numpy/__init__.py in
__getattr__(attr)
    326             raise RuntimeError("Tester was removed in NumPy 1.25.")
    327
--> 328         raise AttributeError("module {!r} has no attribute "
    329                              "{!r}".format(__name__, attr))
    330

AttributeError: module 'numpy' has no attribute 'trii'
```

```
#Upper Triangle
mask2=np.triu(dfc2)
sns.heatmap(dfc2,annot=True, mask=mask2,ax=ax[1,1],cmap='Y1GnBu').set_title('Heat Map L
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-38-82fef8a45007> in <cell line: 2>()
      1 #Upper Triangle
----> 2 mask2=np.triu(dfc2)
      3 sns.heatmap(dfc2,annot=True,
mask=mask2,ax=ax[1,1],cmap='Y1GnBu').set_title('Heat Map Lower Triangle')

NameError: name 'dfc2' is not defined
```

```
#Cluster Maps
sns.clustermap(df11,cmap='RDY1Gn')
```



```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-39-a8ed3f208491> in <cell line: 2>()
      1 #Cluster Maps
----> 2 sns.clustermap(df11,cmap='RDY1Gn')
```

7 frames

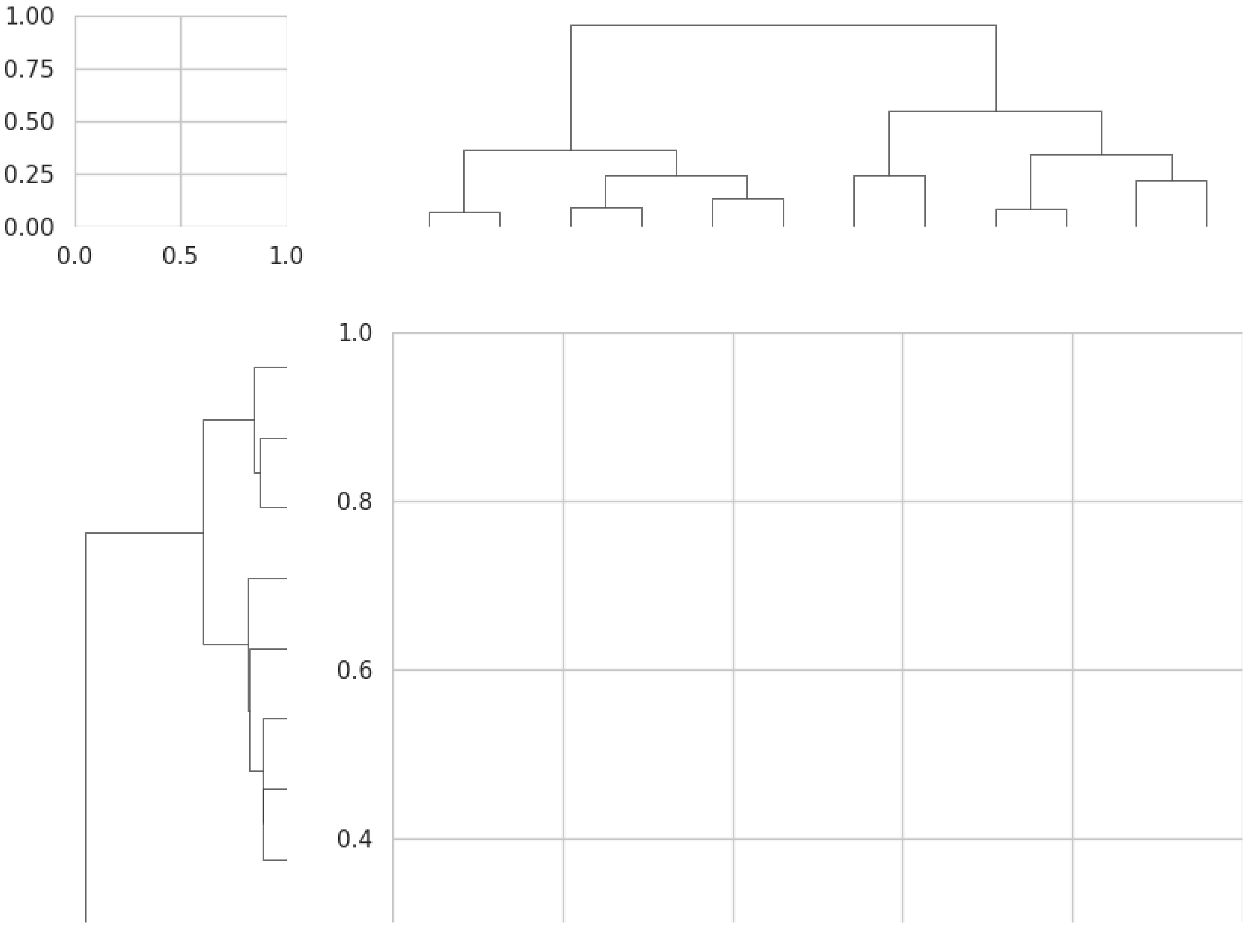
```
/usr/local/lib/python3.10/dist-packages/matplotlib/cm.py in
__getitem__(self, item)
     80             return self._cmaps[item].copy()
     81         except KeyError:
---> 82             raise KeyError(f"{item!r} is not a known colormap name")
from None
     83
     84     def __iter__(self):

KeyError: "'RDY1Gn' is not a known colormap name"
```

```
#Standard_scale=1
sns.clustermap(df11,cmap='RdY1Gn')
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-40-b8d3a58a2b3c> in <cell line: 2>()
      1 #Standard_scale=1
----> 2 sns.clustermap(df11,cmap='RdY1Gn')
```

7 frames

```
/usr/local/lib/python3.10/dist-packages/matplotlib/cm.py in
__getitem__(self, item)
     80             return self._cmaps[item].copy()
     81         except KeyError:
---> 82             raise KeyError(f"{item!r} is not a known colormap name")
from None
     83 
     84     def __iter__(self):

KeyError: "'RdY1Gn' is not a known colormap name"
```



Exercise (Titanic Dataset)

1).plot histogram for the of every class of passengers who survied.

2).plot barplot for the cabin vs fare.

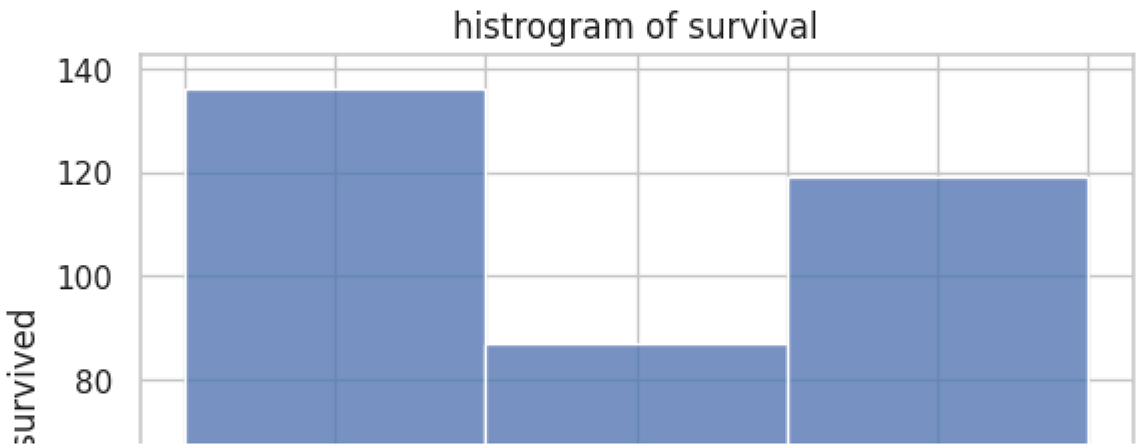
3). plot appropriate graph for Embarker city vs Survived

4).plot graph betwen all the features in agraph

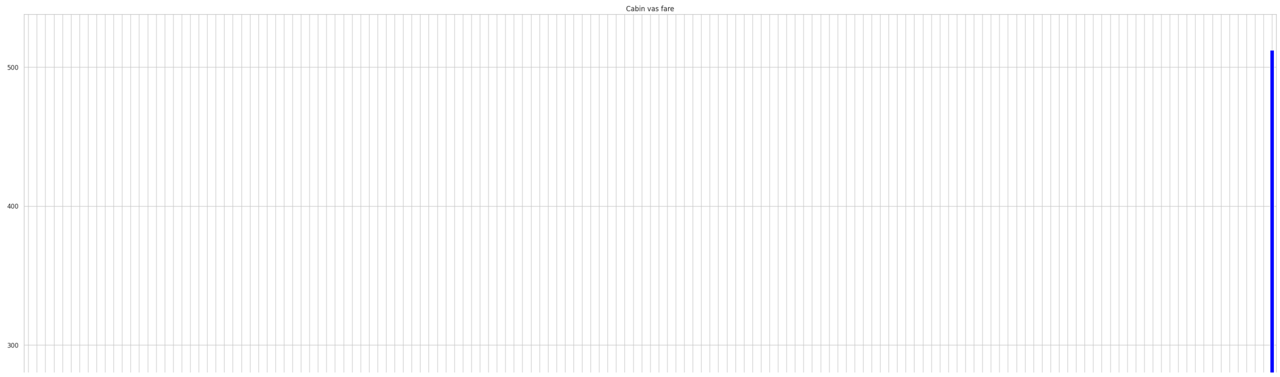
5).plot heatmap and infer two highyly co-related with survived

```
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
df=pd.read_csv('/content/titanic.csv')
```

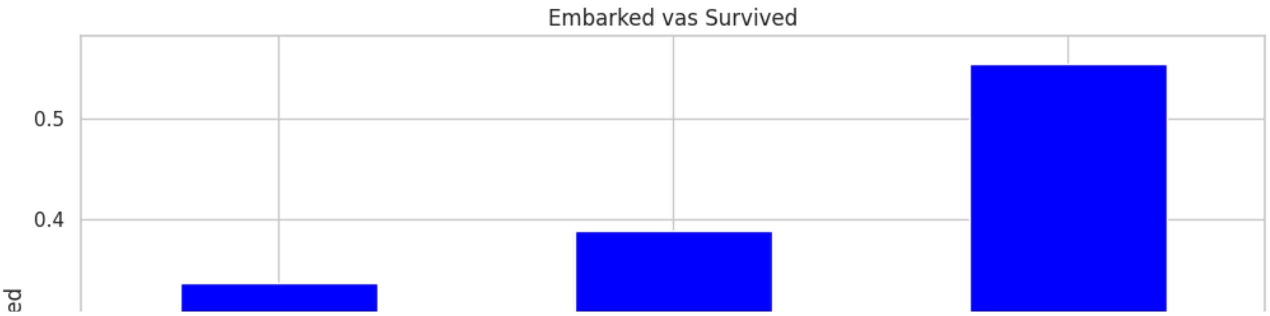
```
survived_passengers=df[df['Survived']==1]
sns.histplot(data=survived_passengers,x='Pclass',bins=range(1,5),discrete=True)
plt.title("histrogram of survival")
plt.xlabel("Passenger class")
plt.ylabel("no of survived")
plt.show()
```



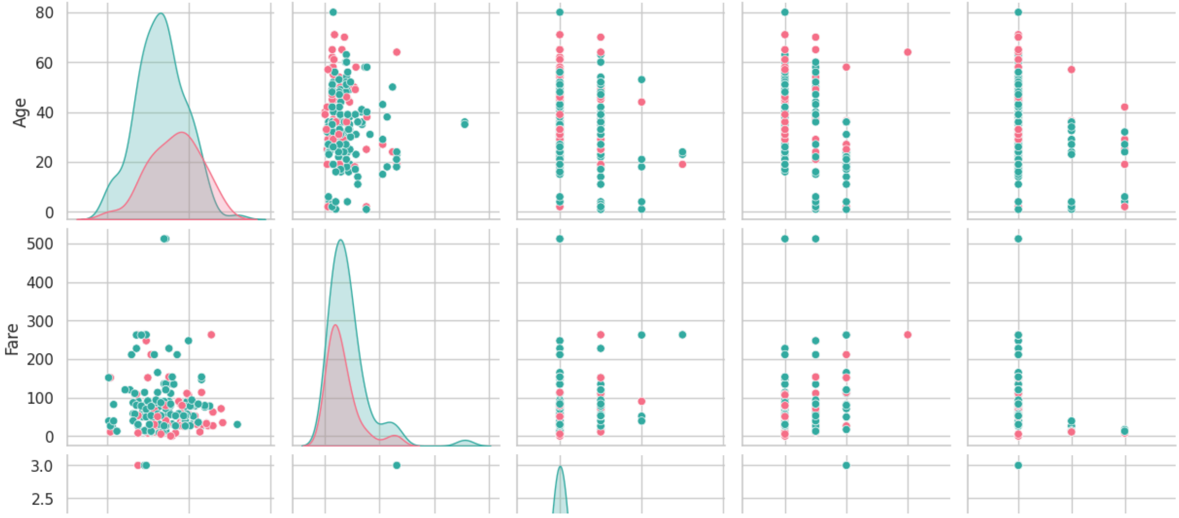
```
plt.figure(figsize=(40,24))
df.groupby('Cabin')['Fare'].mean().sort_values().plot(kind='bar',color='blue')
plt.title("Cabin vas fare")
plt.xlabel("Cabin")
plt.ylabel("Fare")
plt.show()
```



```
plt.figure(figsize=(12,6))
df.groupby('Embarked')['Survived'].mean().sort_values().plot(kind='bar',color='blue')
plt.title("Embarked vas Survived")
plt.xlabel("Embarked")
plt.ylabel("Survived")
plt.show()
```



```
titanic_df = df.dropna()
numerical_features = ['Age', 'Fare', 'SibSp', 'Parch']# Select only categorical feature
categorical_features = ['Survived', 'Pclass', 'Sex', 'Embarked']
selected_features = numerical_features + categorical_features
sns.pairplot(titanic_df[selected_features], hue='Survived', palette='husl')
plt.show()
```


Age
80
60
40
20
0
Fare
500
400
300
200
100
0
3.0
2.5