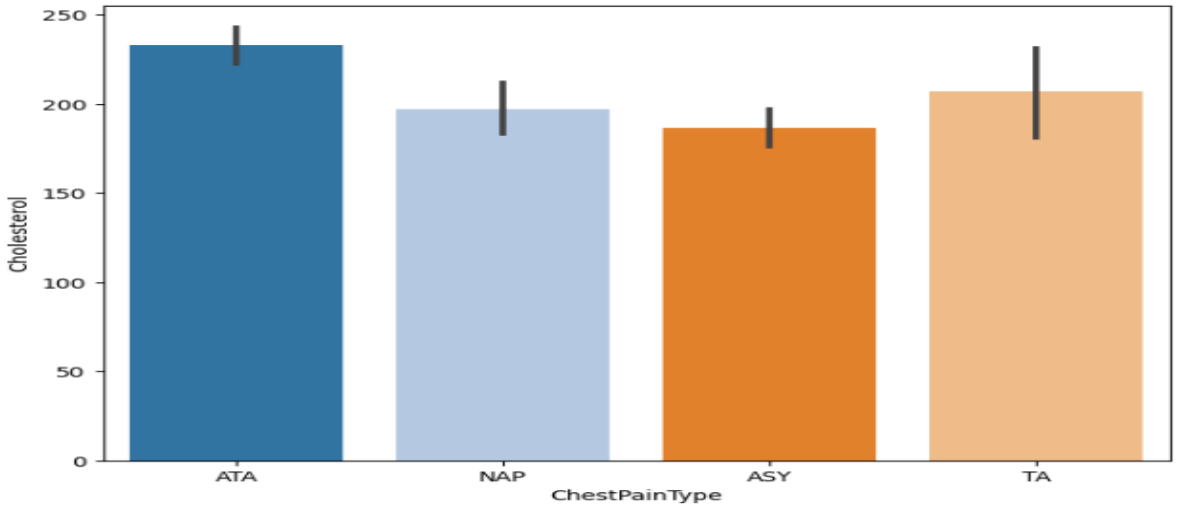
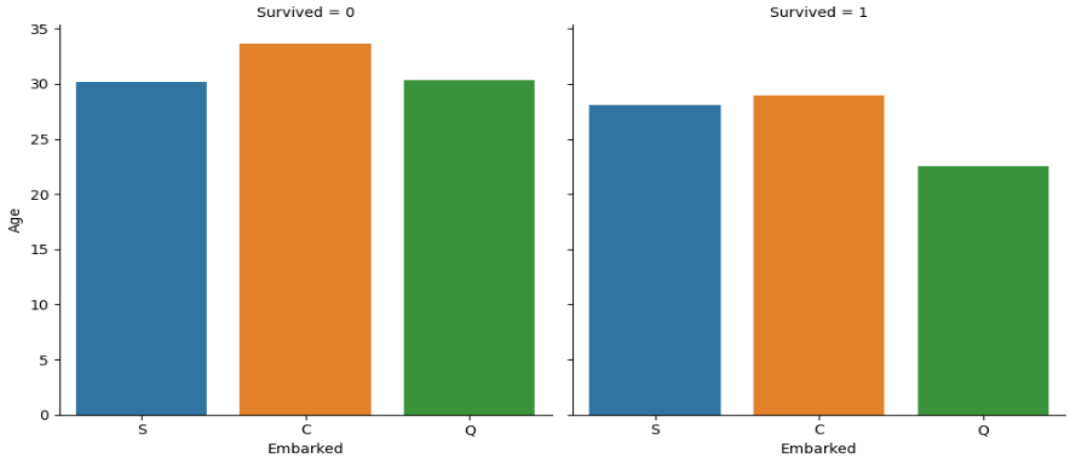
Mastering Data Visualization Techniques (Part 1)

Prepared by: Syed Afroz Ali

```
plt.figure(figsize = (8, 6))
plt.ticklabel_format(style = 'plain')
sns.barplot(x = heart["ChestPainType"], y = heart["Cholesterol"], palette = "tab20");
```

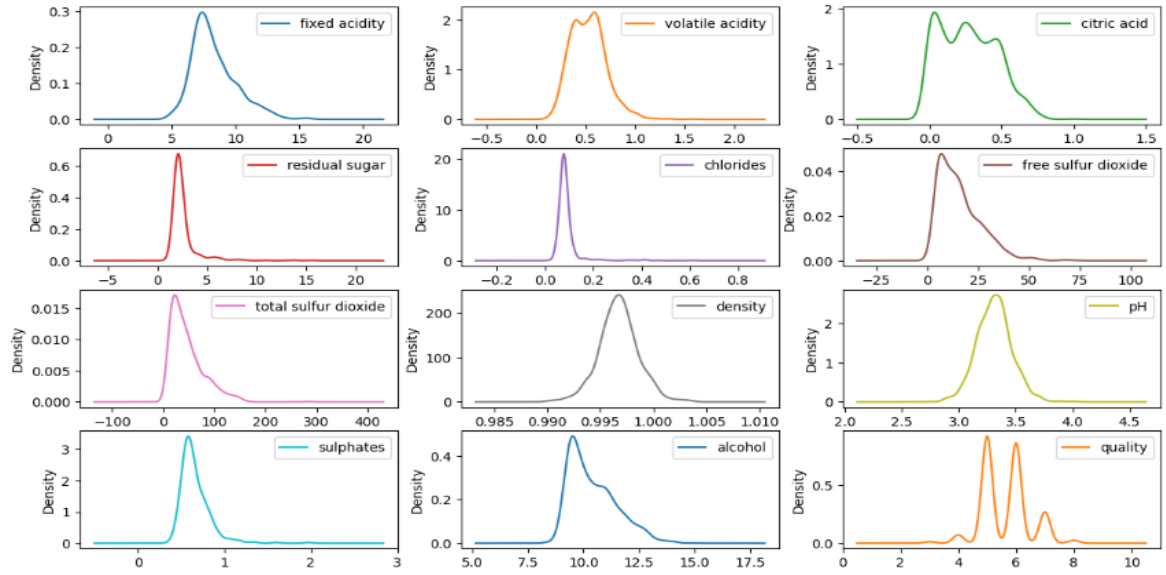



```
sns.catplot(data = titanic , x ="Embarked" , y ="Age" , col ="Survived" , kind="bar" , ci =None)
plt.show()
```

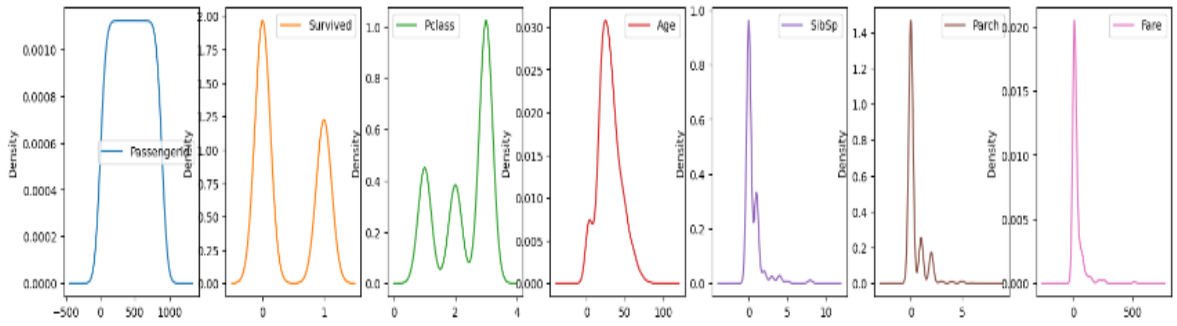



```
wine.plot(kind='density', subplots=True, layout=(4,3), sharex=False, figsize= (14,8))
```

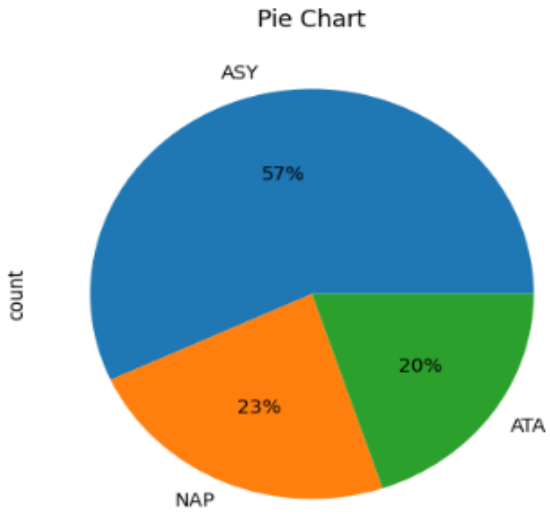
```
plt.show()
```



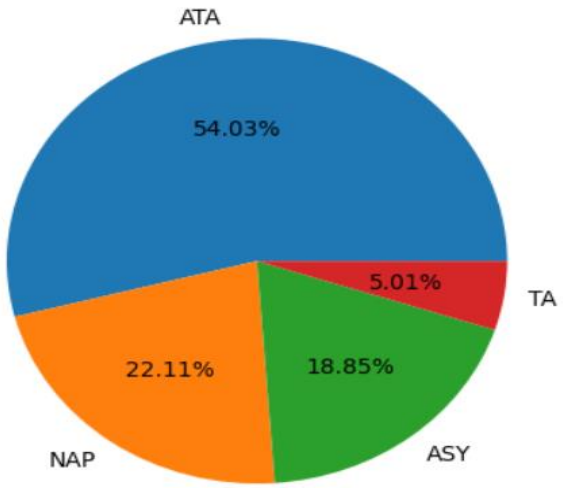

```
numeric_feature = titanic.dtypes!=object
final_numeric_feature = titanic.columns[numeric_feature].tolist()

titanic[final_numeric_feature].plot(kind='density', subplots=True, layout=(1,7), sharex=False, figsize= (20,4))
plt.show()
```



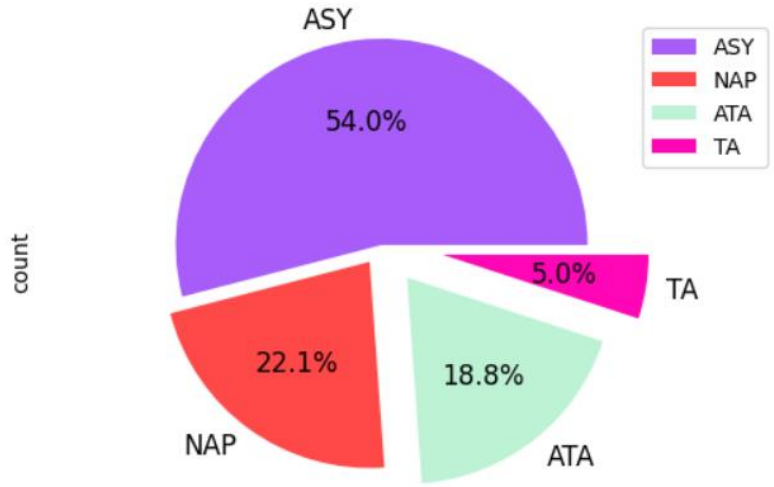

```
heart["ChestPainType"].value_counts()[:3].plot.pie(figsize = (5, 5),
                                                   autopct = '%1.0f%%')
plt.title("Pie Chart")
plt.xticks(rotation = 90)
plt.show()
```



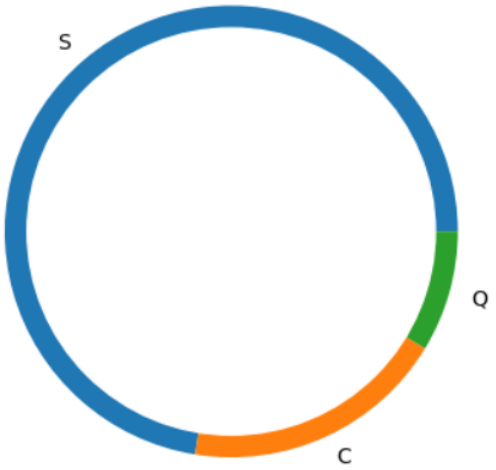

```
plt.pie(heart['ChestPainType'].value_counts(),labels=heart[
'ChestPainType'].unique(),autopct = '%1.2f%%');
```



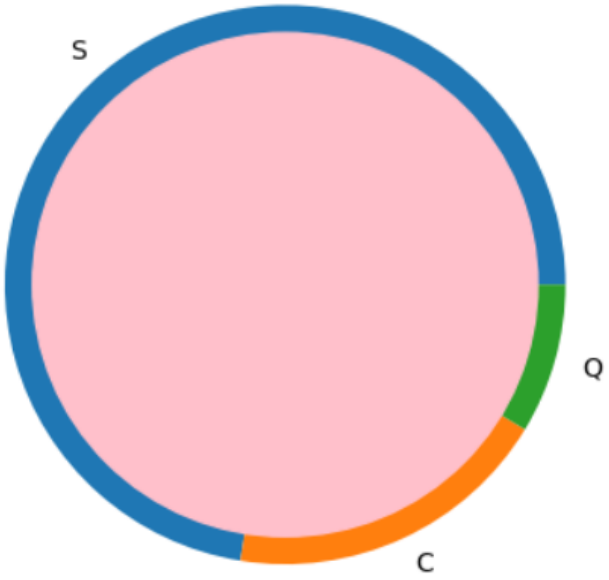

```
plt.figure(figsize = (6, 4))
counts = heart["ChestPainType"].value_counts()
explode = (0, 0.1, 0.2, 0.3)
colors = ['#A85CF9', '#FF4949', '#BDF2D5', '#FF06B7', '#4B
7BE5', '#FF5D5D', '#FAC213', '#37E2D5', '#6D8B74', '#E9D5
CA']
counts.plot(kind = 'pie', fontsize = 12, colors = colors, expl
ode = explode, autopct = '%1.1f%%')
plt.axis('equal')
plt.legend(labels = counts.index, loc = "best")
plt.show()
```



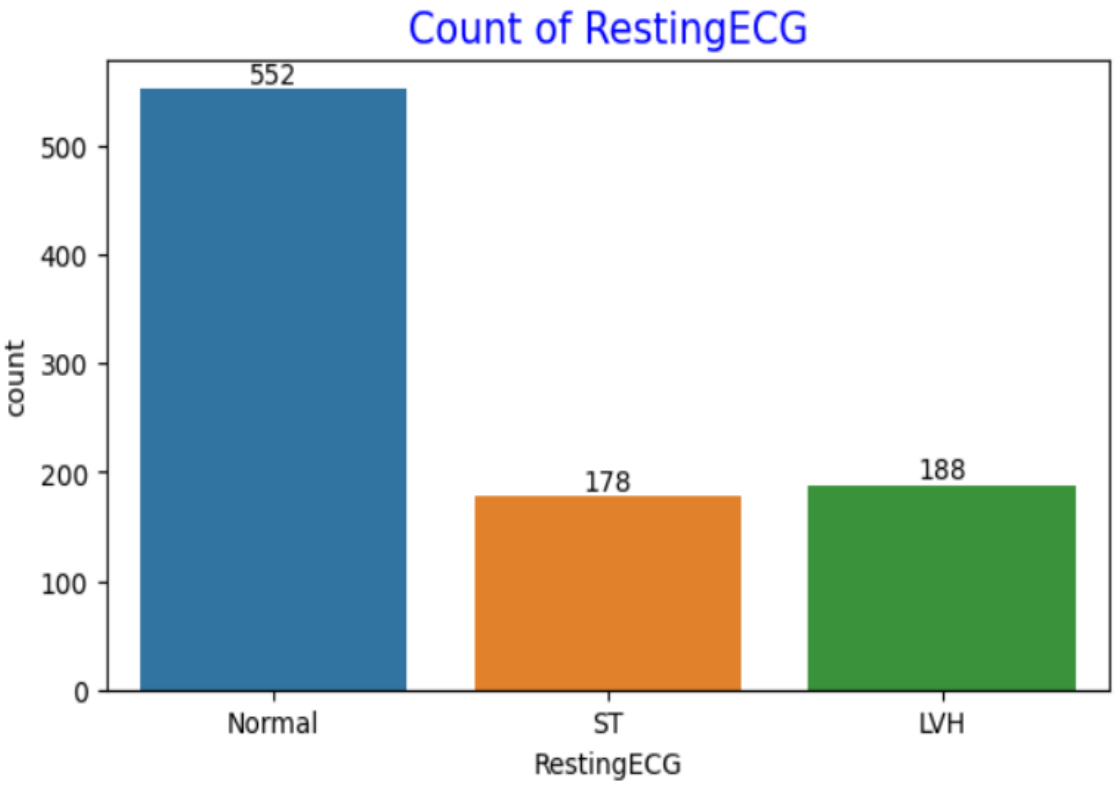

```
my_circle=plt.Circle( (0,0), 0.9, color='white')
plt.pie(titanic['Embarked'].value_counts()[:10].values, label
s = titanic['Embarked'].value_counts()[:10].index)
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```



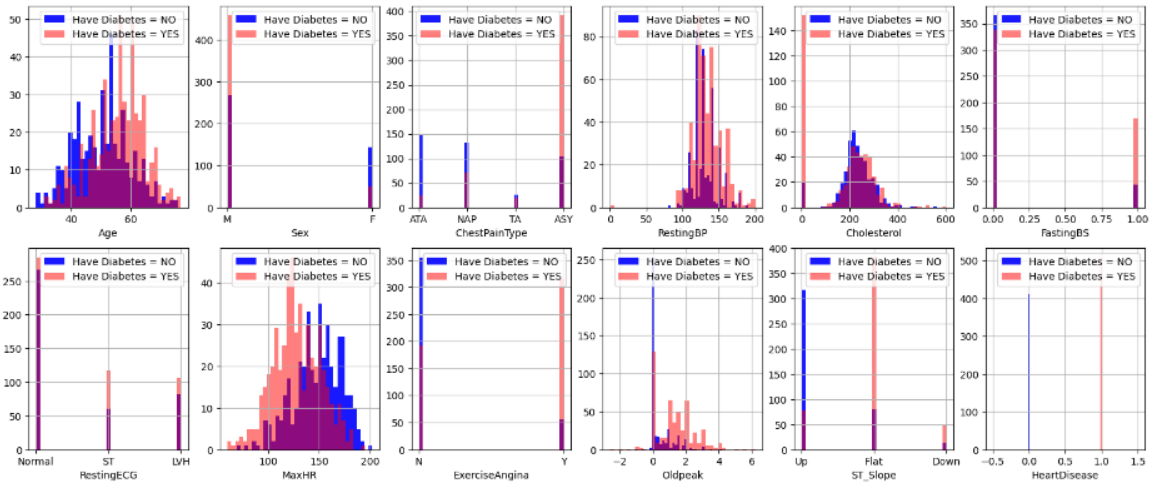

```
my_circle=plt.Circle( (0,0), 0.9, color='pink')
plt.pie(titanic['Embarked'].value_counts()[:10].values, label
s = titanic['Embarked'].value_counts()[:10].index)
p=plt.gcf()
p.gca().add_artist(my_circle)
plt.show()
```



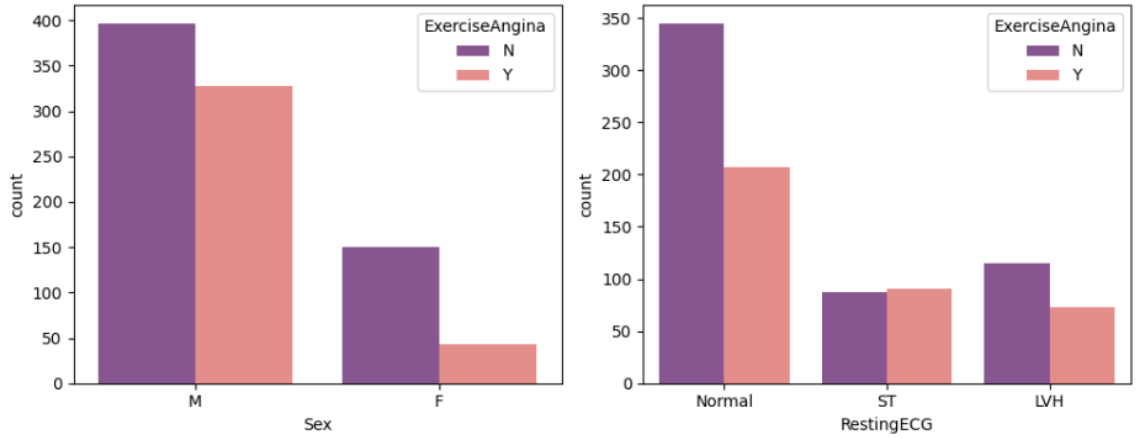

```
plt.figure(figsize = (7,4))
ax = sns.countplot(x=heart['RestingECG'])
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Count of RestingECG", fontsize = 15,color='Blue');
```



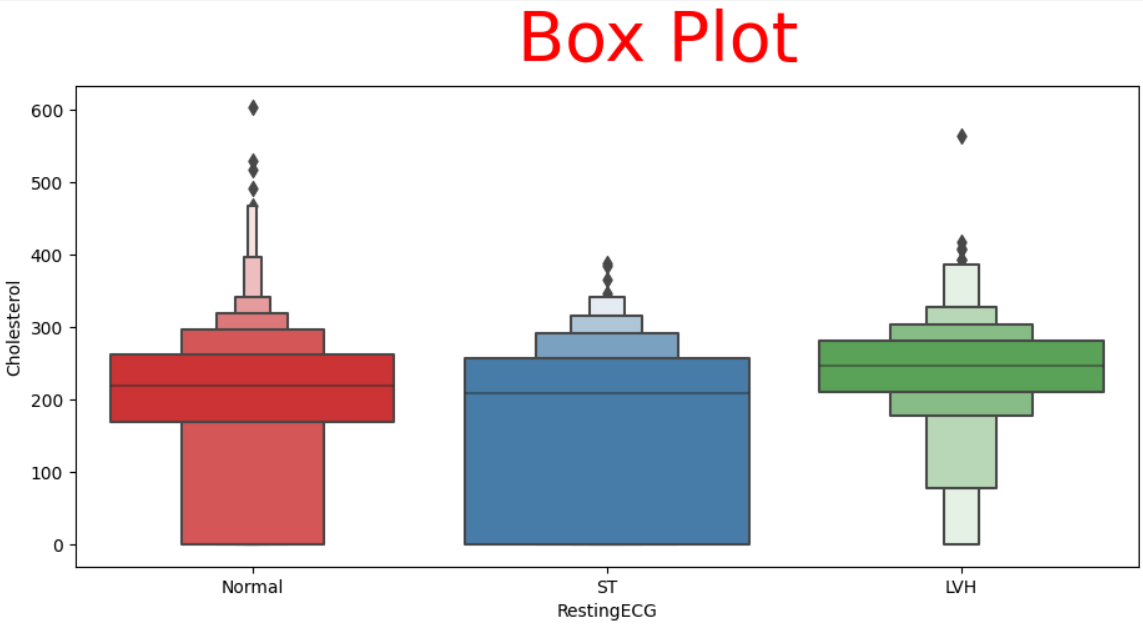
```
# Visulazing the distibution of the data for every feature
plt.figure(figsize=(20, 8))

for i, column in enumerate(heart.columns, 1):
    plt.subplot(2, 6, i)
    heart[heart["HeartDisease"] == 0][column].hist(bins=35,
color='blue', label='Have Diabetes = NO', alpha=0.9)
    heart[heart["HeartDisease"] == 1][column].hist(bins=35,
color='red', label='Have Diabetes = YES', alpha=0.5)
    plt.legend()
    plt.xlabel(column)
```



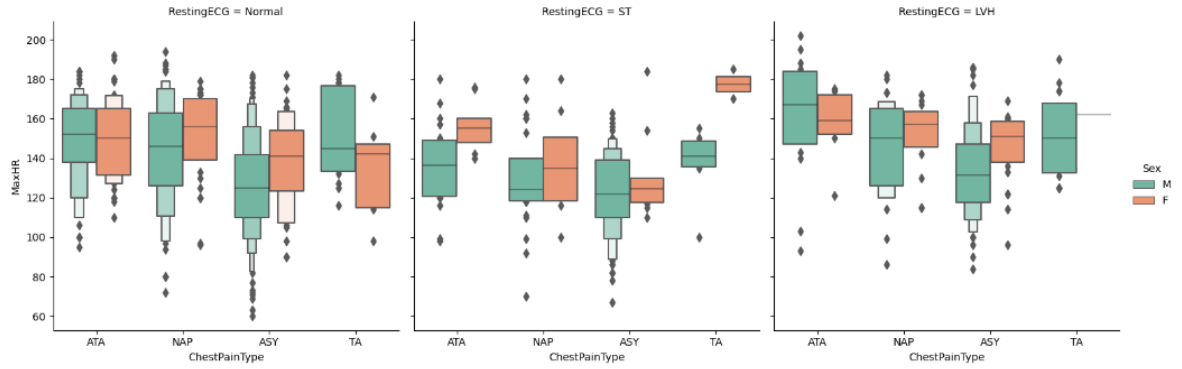

```
cat = ['Sex','RestingECG']

fig, ax = plt.subplots(1, 2, figsize = (10, 4))
for indx, (column, axes) in list(enumerate(list(zip(cat,
                                        ax.flatten())))):
    sns.countplot(ax = axes, x = heart[column], hue = heart[
'ExerciseAngina'],
                  palette = 'magma', alpha = 0.8)

else:
    [axes.set_visible(False) for axes in ax.flatten()[indx + 1:]
]
plt.tight_layout()
plt.show()
```



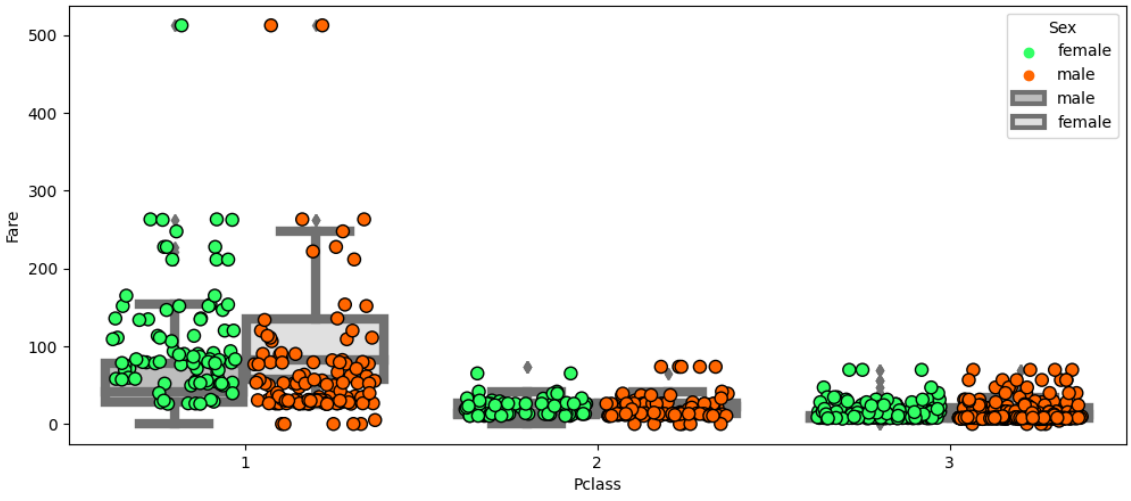

```
plt.figure(figsize=(11,5))
plt.gcf().text(.55, .95, "Box Plot", fontsize = 40, color='Red'
,ha='center', va='center')
sns.boxenplot(x=heart['RestingECG'] , y = heart['Cholester
ol'],palette="Set1")
plt.show()
```



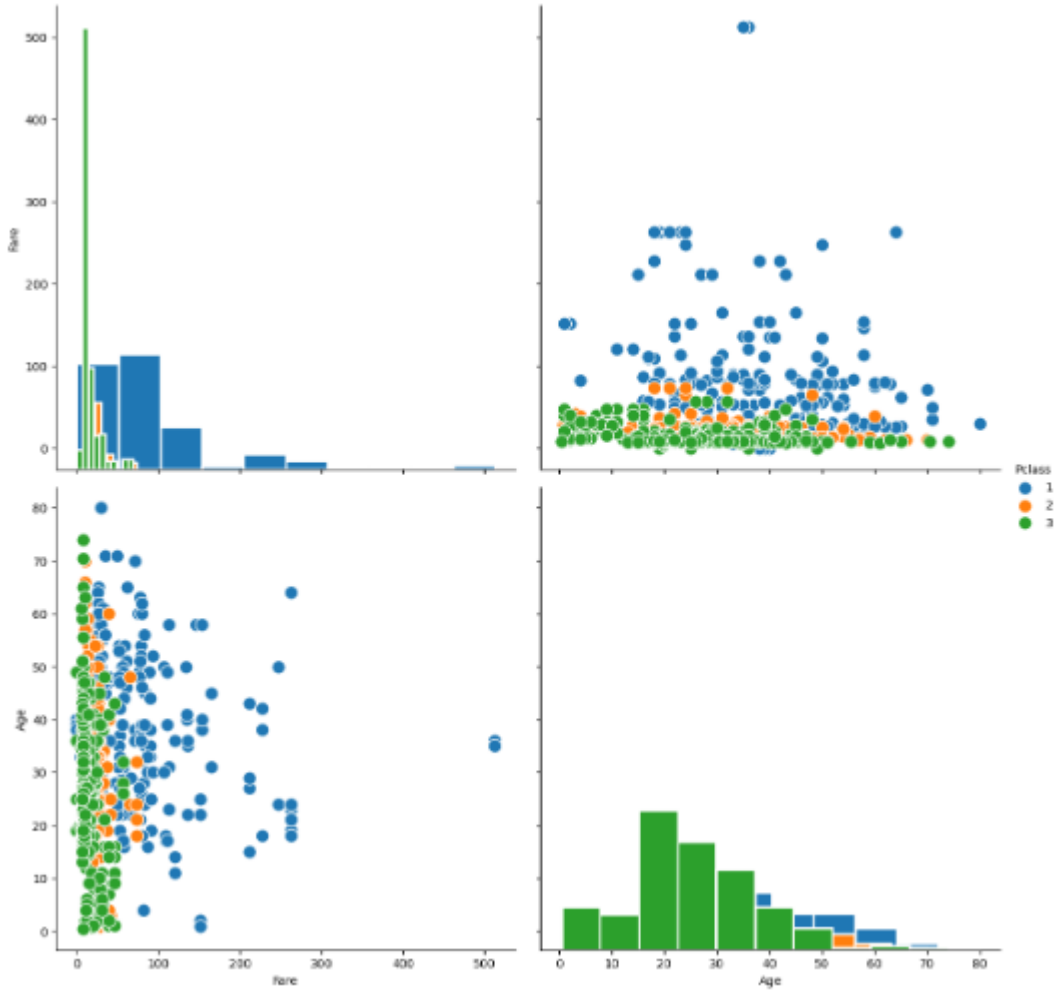

```
# Facet along the columns to show a categorical variable
using "col" parameter
plt.figure(figsize=(11,5))
sns.catplot(x="ChestPainType" , y = "MaxHR", hue= "Sex",
            col="RestingECG", kind="boxen",palette="Set2" , h
eight=5, aspect=1 ,data=heart)
plt.show();
```



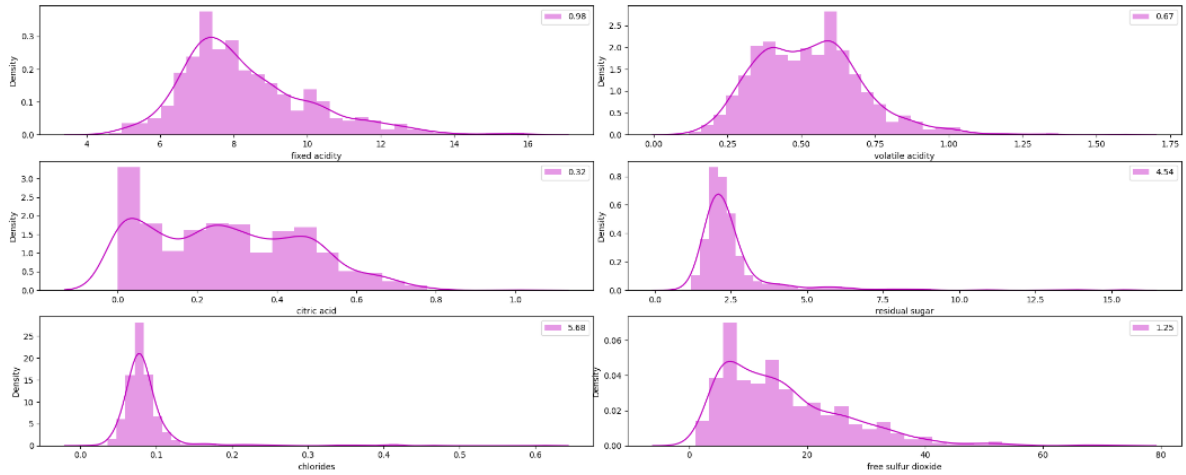

```
plt.figure(figsize=(12,5))
params = dict(data=titanic ,x = titanic.Pclass ,y = titanic.Fare ,hue=titanic.Sex,dodge=True)
sns.stripplot(**params , size=8,jitter=0.35,palette=['#33FF66','#FF6600','Blue'],edgecolor='black',linewidth=1)
sns.boxplot(**params ,palette=['#BDBDBD','#E0E0E0'],linewidth=6)
plt.show()
```



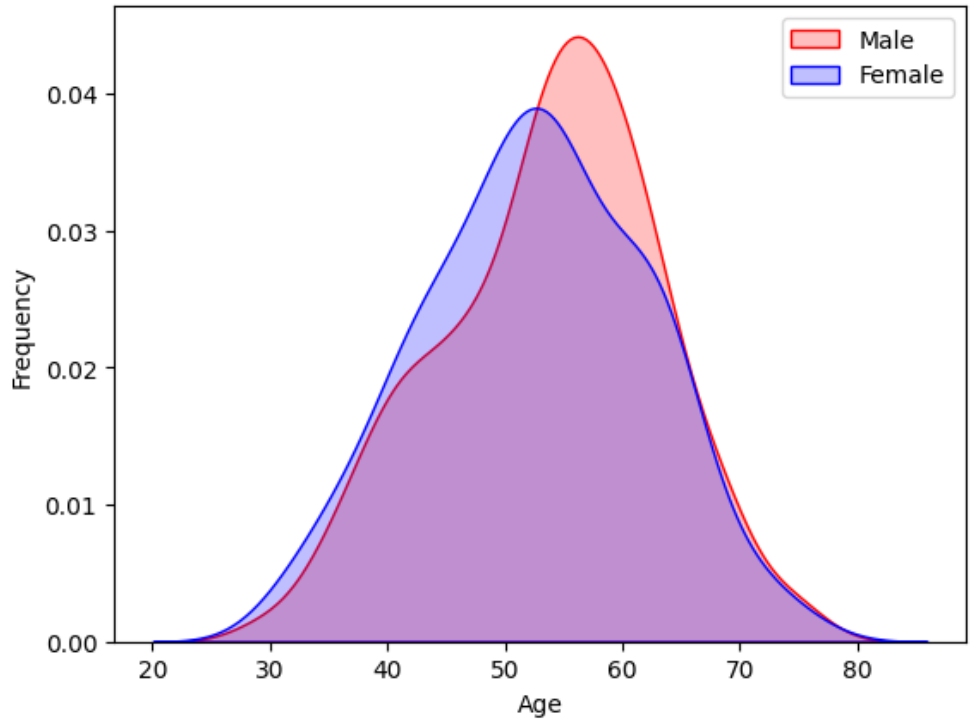

```
# Plot a subset of variables
g = sns.PairGrid(titanic, hue='Pclass' ,x_vars=["Fare" , "Age"],y_vars=["Fare" , "Age"],
                 height=6, aspect=1)
g = g.map_offdiag(plt.scatter , edgecolor="w", s=130)
g = g.map_diag(plt.hist , edgecolor ='w', linewidth=2)
g = g.add_legend()
plt.show()
```



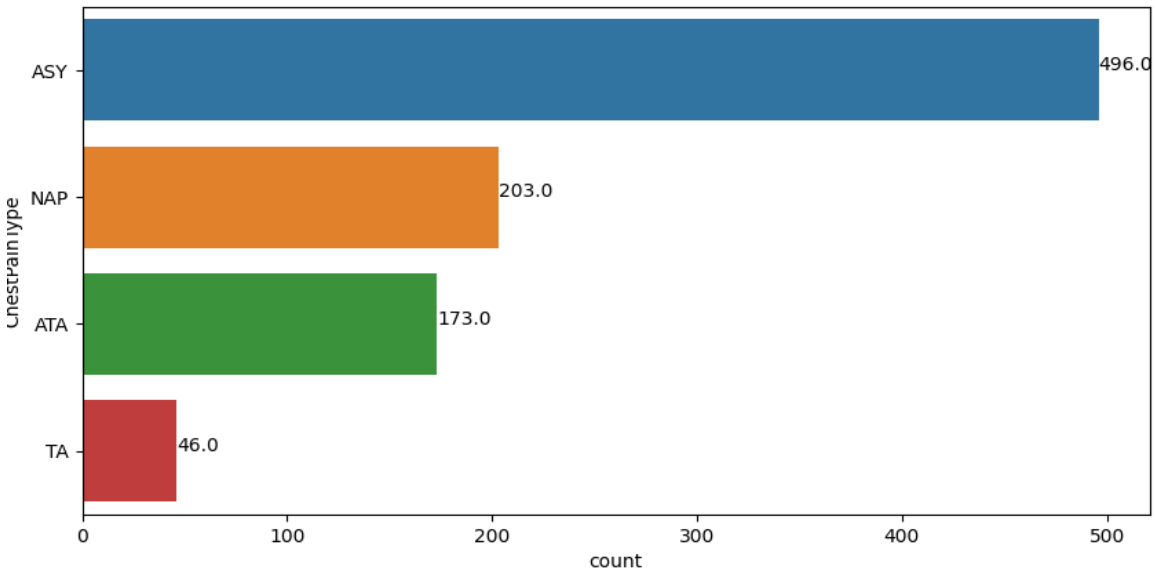

```
features_mean= list(wine.columns[:6])

num_rows, num_cols = 3,2

fig, axes = plt.subplots(num_rows, num_cols, figsize=(20, 8
))
fig.tight_layout()

for index, column in enumerate(wine[features_mean].colu
mns):
    i,j = (index // num_cols, index % num_cols)
    g = sns.distplot(wine[column], color="m", label="%.2f"%
(wine[column].skew()), ax=axes[i,j])
    g = g.legend(loc="best")
```



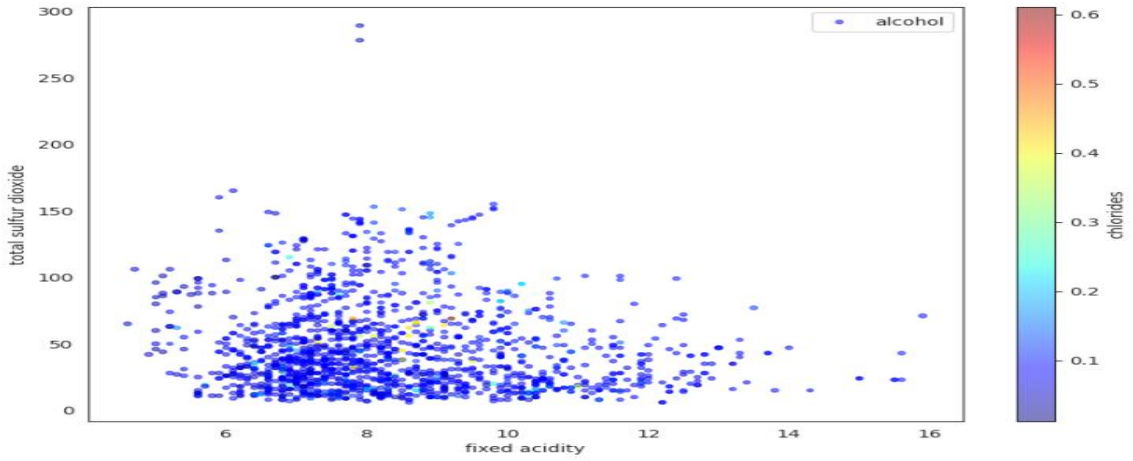

```
y = heart['Sex']

# Explore Age distibution
g = sns.kdeplot(heart["Age"][(y == 'M') & (heart["Age"].notnull())], color="Red", shade=True)
g = sns.kdeplot(heart["Age"][(y == 'F') & (heart["Age"].notnull())], ax=g, color="Blue", shade=True)
g.set_xlabel("Age")
g.set_ylabel("Frequency")
g = g.legend(["Male","Female"])
```



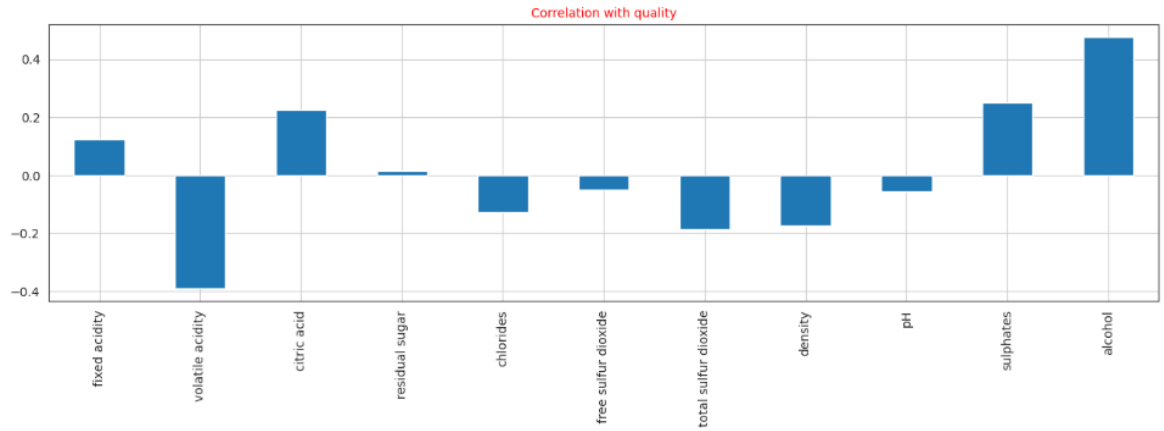
```
raw_df = heart [['Age', 'Sex', 'ChestPainType', 'RestingBP', 'Cholesterol', 'FastingBS',
       'RestingECG', 'MaxHR', 'ExerciseAngina', 'Oldpeak', 'ST_Slope',
       'HeartDisease']]
# Function to print width of barcharts on the bars
def barw(ax):
    for p in ax.patches:
        val = p.get_width() #height of the bar
        x = p.get_x()+ p.get_width() # x- position
        y = p.get_y() + p.get_height()/2 #y-position
        ax.annotate(round(val,2),(x,y))
plt.figure(figsize=(10,5))
ax0 = sns.countplot(data = heart, y ='ChestPainType', order = heart['ChestPainType'].value_counts().index)
barw(ax0)
plt.show()
```



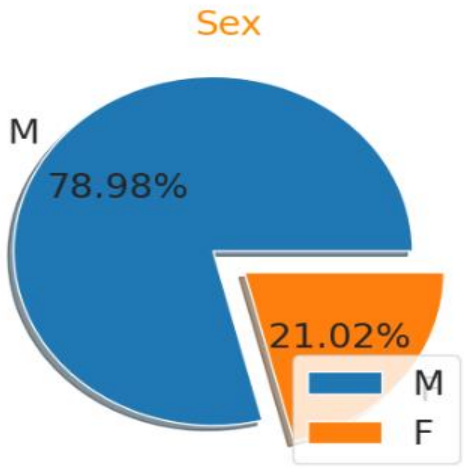

```
sns.set_style('white')
wine.plot(kind="scatter", x="fixed acidity", y="total sulfur dioxide", alpha=.5,
          s=wine["alcohol"], label="alcohol", figsize=(10,7),
          c="chlorides", cmap=plt.get_cmap("jet"), colorbar=True,
          sharex=False)
plt.legend()
plt.show()
```



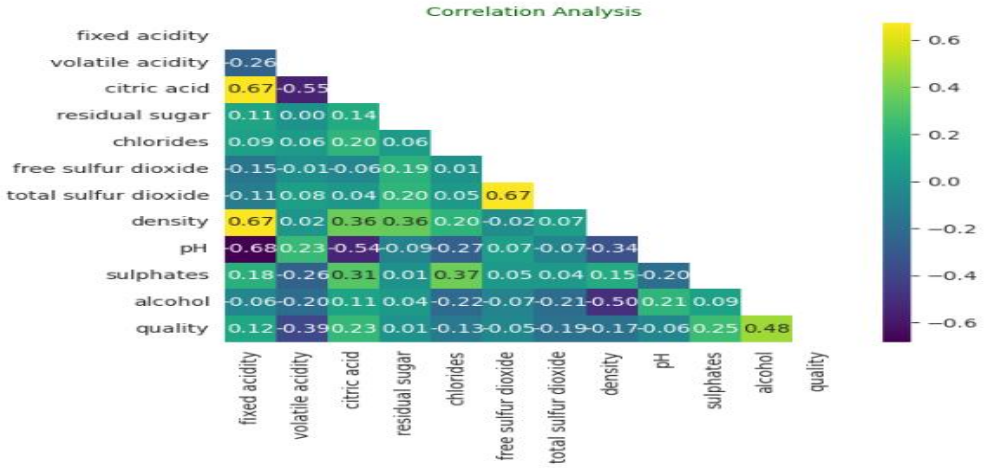

```
#Correlation with Response Variable class
X = wine.drop(['quality'], axis=1)
y = wine['quality']

X.corrwith(y).plot.bar(figsize=(16, 4), rot=90, grid=True)
plt.title('Correlation with quality',
          fontsize=30,
          color='Red',
          font='Times New Roman')
plt.show()
```



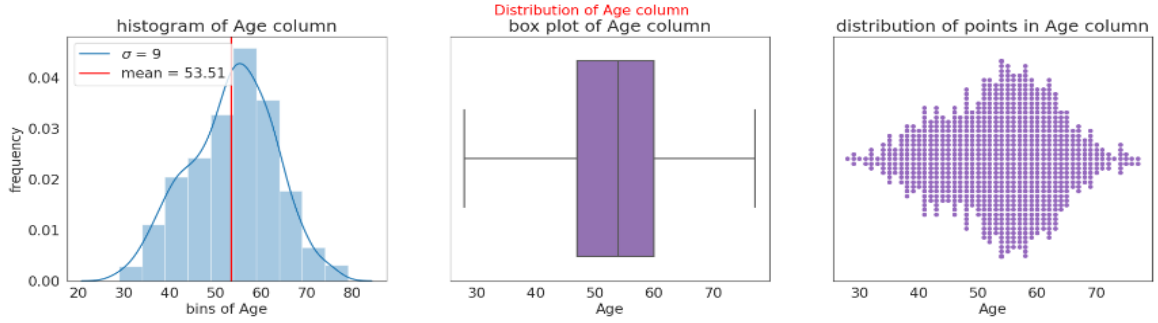

```
import matplotlib
matplotlib.rcParams.update({'font.size': 20})
ax=heart['Sex'].value_counts().plot.pie(explode=[0.1, 0.1],autopct='
%1.2f%%',shadow=True);
ax.set_title(label = "Sex", fontsize = 40,color='DarkOrange',font='Luc
ida Calligraphy');
plt.legend(labels=['M','F'])
plt.axis('off');
```



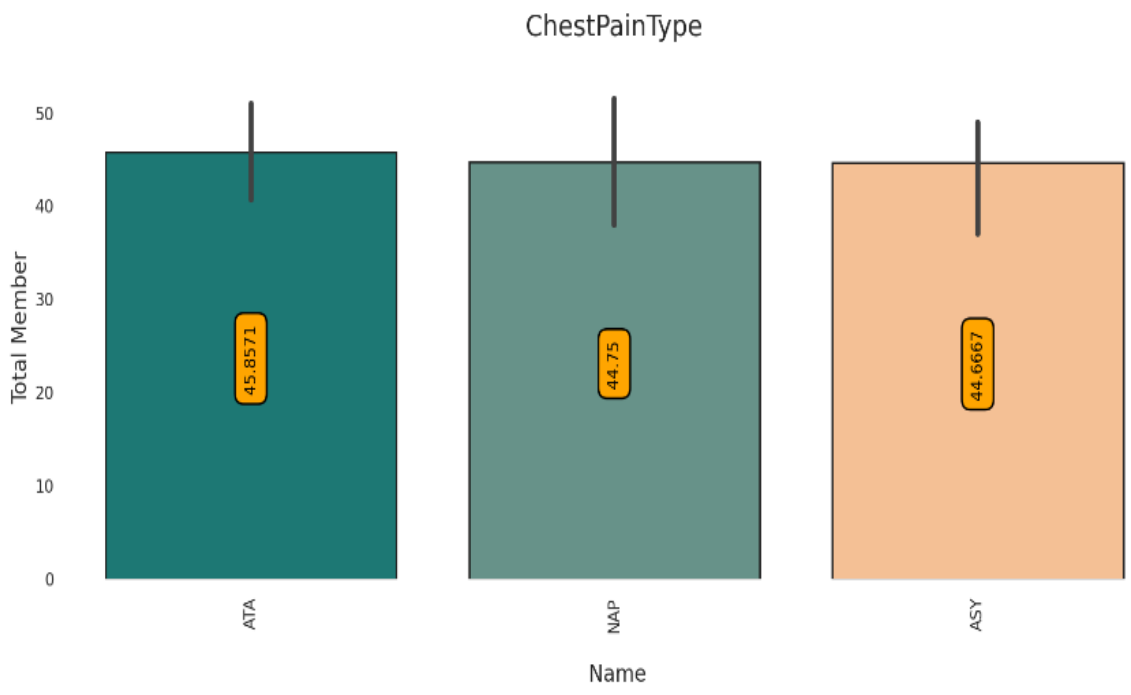

```
matplotlib.rcParams.update({'font.size': 10})
corr = wine.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
plt.title('Correlation Analysis',
          fontsize=25,
          color='DarkGreen',
          font='Times New Roman')
sns.heatmap(corr,
            mask=mask,
            annot=True,
            lw=0,
            linecolor='white',
            cmap='viridis',
            fmt="0.2f")
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.show()
```



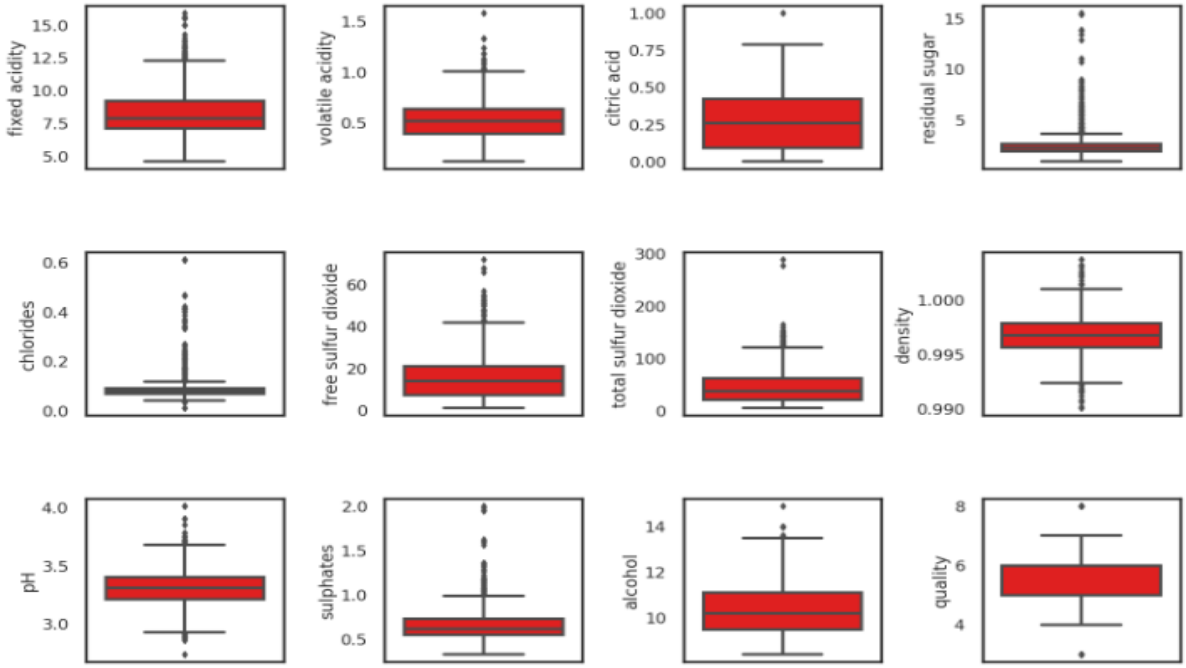
```
#set configuration for charts
plt.rcParams["figure.figsize"]=[20 , 5]
plt.rcParams["font.size"]=15
plt.rcParams["legend.fontsize"]="medium"
plt.rcParams["figure.titlesize"]="medium"

def plot_disribution(data , x ,color,bins ):
    mean = data[x].mean()
    std = data[x].std()
    info=dict(data = data , x = x , color = color)
    plt.subplot(1 , 3 , 1 , title =f"Ditstribution of {x} column")
    sns.distplot(a=data[x] , bins = bins)
    plt.xlabel(f"bins of {x}")
    plt.axvline(mean , label ="mean" , color ="red")
    plt.ylabel("frequency")
    plt.legend(["${\sigma}$ = %d"%std , f"mean = {mean:.2f}"])
    plt.title(f"histogram of {x} column")
    plt.subplot(1 , 3 , 2)
    sns.boxplot(**info)
    plt.xlabel(f"{x}")
    plt.title(f"box plot of {x} column")
    plt.subplot(1 , 3 , 3)
    sns.swarmplot(**info)
    plt.xlabel(f"{x}")
    plt.title(f"distribution of points in {x} column")
    plt.suptitle(f"Distribution of {x} column" , fontsize =15 , color="red
")
plt.show()

age_bins = np.arange(29 , 77+5 , 5)
base_color = sns.color_palette()[4]
plot_disribution(data = heart , x ="Age" , color = base_color , bins=ag
e_bins)
```



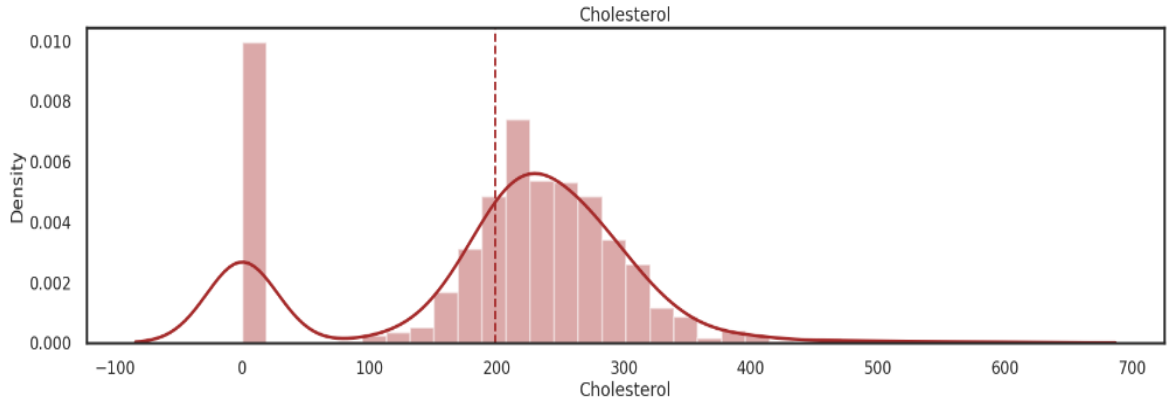
```
sns.set_style("white")
sns.set_context("poster",font_scale = .7)
palette = ["#1d7874","#679289","#f4c095","#ee2e31","#ffb563","#918450","#f85e00","#a41623","#9a031e","#d6d6d6","#ffee32","#ffd100","#333533","#202020"]
# sns.palplot(sns.color_palette(palette))
# plt.show()

plt.subplots(figsize=(20,8))
p = sns.barplot(x=heart["ChestPainType"][:14],y=heart["Age"],palette=palette, saturation=1, edgecolor = "#1c1c1c", linewidth = 2)
p.axes.set_title("\n ChestPainType \n", fontsize=25)
plt.ylabel("Total Member" , fontsize = 20)
plt.xlabel("\n Name" , fontsize = 20)
# plt.yscale("log")
plt.xticks(rotation = 90)
for container in p.containers:
    p.bar_label(container,label_type = "center",padding = 6,size = 15,color = "black",rotation = 90,
    bbox={"boxstyle": "round", "pad": 0.6, "facecolor": "orange", "edgecolor": "black", "alpha": 1})

sns.despine(left=True, bottom=True)
plt.show()
```



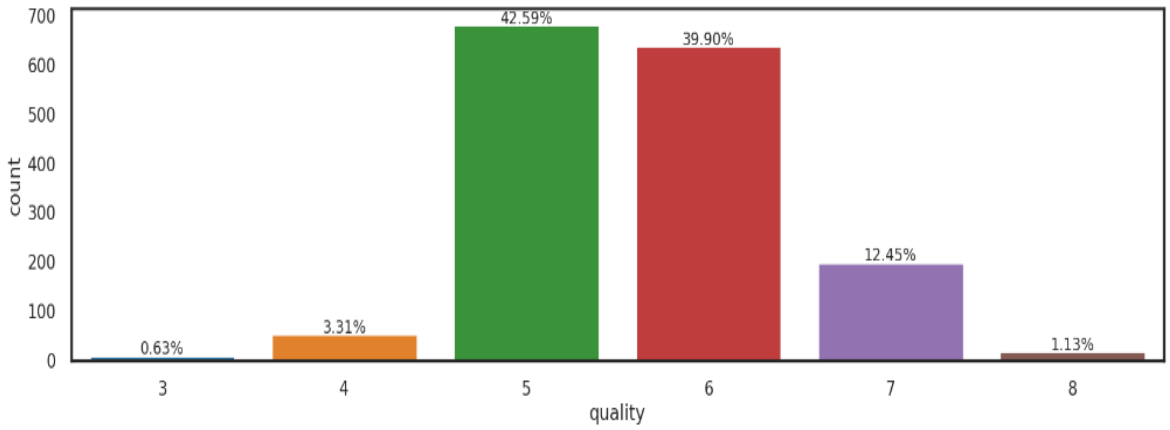
```
fig,axis=plt.subplots(ncols=4,nrows=3,figsize=(15,10))
index=0
axis=axis.flatten()

for col,values in wine.items():
    sns.boxplot(y=col,data=wine,color='r',ax=axis[index])
    index+=1
plt.tight_layout(pad=0.5,w_pad=0.7,h_pad=5.0);
```



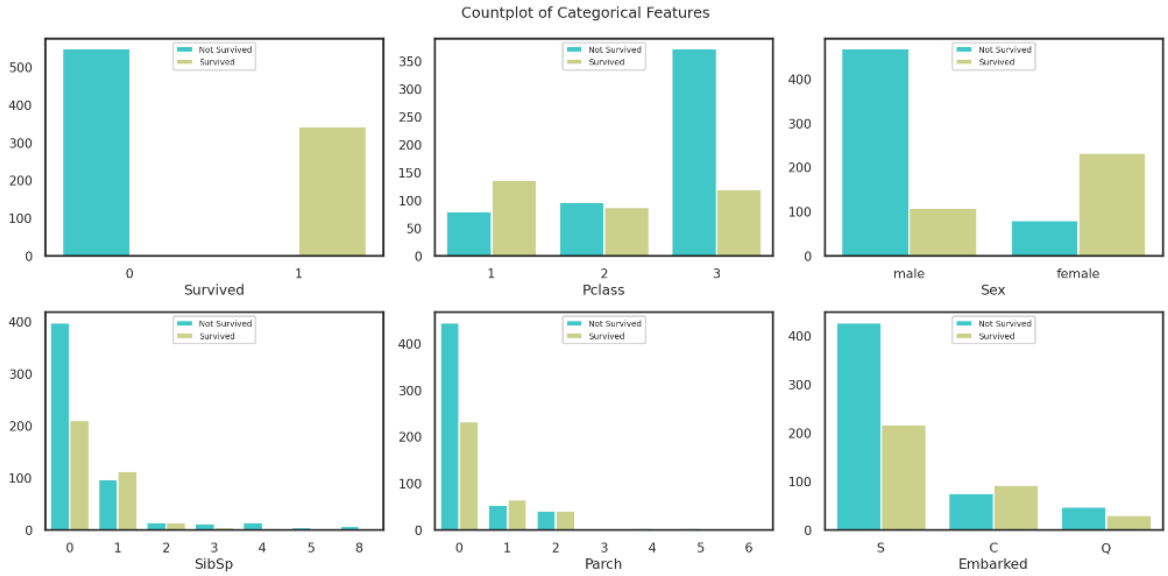

```
#checking the target variables for distribution
sns.distplot(heart['Cholesterol'],color='Brown')
plt.axvline(x=heart['Cholesterol'].mean(), color='Brown', lin
estyle='--', linewidth=2)
plt.title('Cholesterol');
```



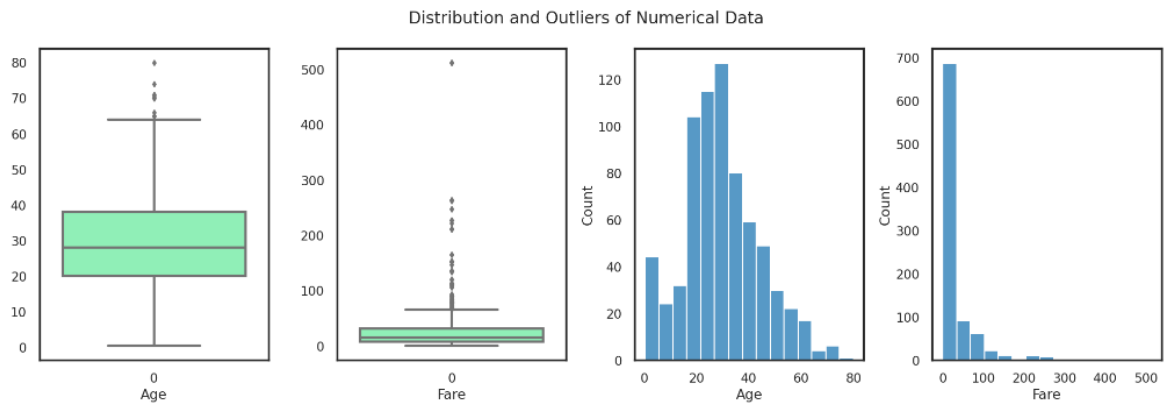
```
s = sns.countplot(x = 'quality',data = wine)
sizes=[]
for p in s.patches:
    height = p.get_height()
    sizes.append(height)
    s.text(p.get_x()+p.get_width()/2.,
           height + 3,
           '{:1.2f}%'.format(height/len(wine)*100),
           ha="center", fontsize=14)
```



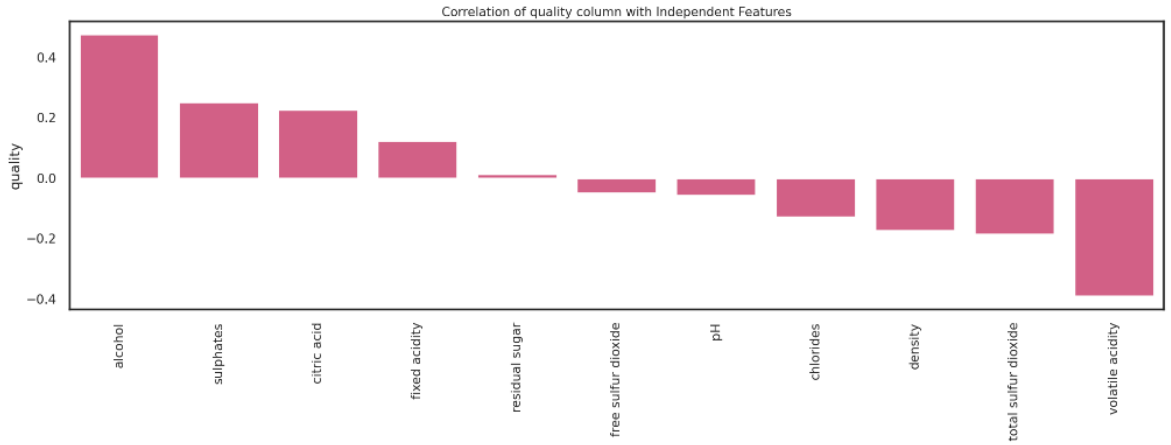

```
countfeature = ["Survived", "Pclass", "Sex", "SibSp", "Parch", "Embarked"]
countlist = list(enumerate(countfeature))

plt.figure(figsize = (20,10))
plt.suptitle("Countplot of Categorical Features", fontsize=18)
for i in countlist:
    plt.subplot(2,3,i[0]+1)
    sns.countplot(data = titanic, x = i[1], hue = "Survived", palette="rainbow")
    plt.ylabel("")
    plt.legend(['Not Survived', 'Survived'], loc='upper center', prop={'size': 10})
plt.tight_layout()
plt.show()
```



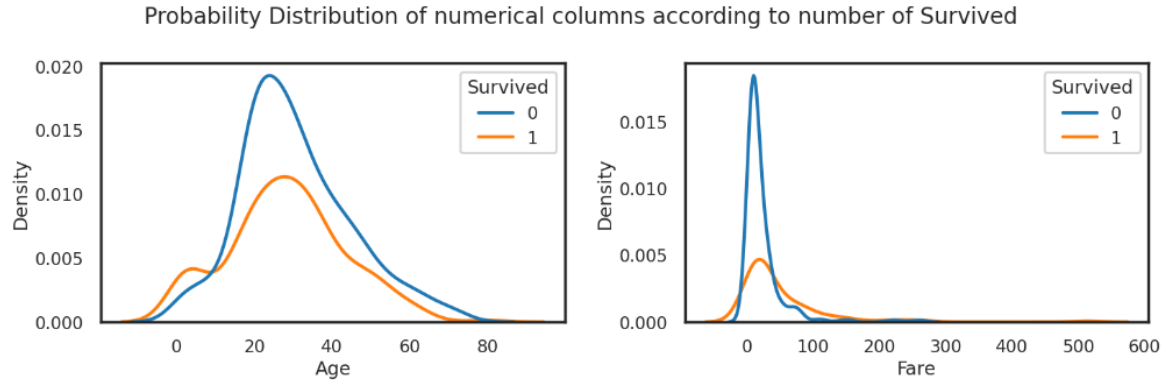

```
numfeature = ["Age", "Fare"]
enumfeat = list(enumerate(numfeature))

plt.figure(figsize=(20,7))
plt.suptitle("Distribution and Outliers of Numerical Data", fontsize=20)
for i in enumfeat:
    plt.subplot(1,4,i[0]+1)
    sns.boxplot(data = titanic[i[1]], palette="rainbow")
    plt.xlabel(str(i[1]))
for i in enumfeat:
    plt.subplot(1,4,i[0]+3)
    sns.histplot(data = titanic[i[1]], palette="rainbow", bins=15)
    plt.xlabel(str(i[1]))
plt.tight_layout()
plt.show()
```



```
plt.figure(figsize=(20,6))
plt.title("Correlation of quality column with Independent Features", f
ontsize=15)
corr = wine.corr()["quality"].sort_values(ascending=False)[1:]
sns.barplot(x=corr.index, y=corr, color=(0.90,0.30,0.50))
plt.tight_layout()
plt.xticks(rotation = 90)
plt.show()
```




```
plt.figure(figsize=(15,5))
plt.suptitle("Probability Distribution of numerical columns
according to number of Survived", fontsize = 20)
for i in enumfeat:
    plt.subplot(1,2,i[0]+1)
    sns.kdeplot(data=titanic, x=i[1], hue="Survived")
plt.tight_layout()

plt.show()
```



```
import missingno as msno
msno.matrix(titanic, color=(0.50,0.30,0.80))
plt.show()
x = titanic.isnull().sum()
for a, b in x.items():
    if b > 0:
        print(f"There are {b} missing values in column: {a}")
```

Titaic Data:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

wine Data:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5

Heart Data:

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0

Mastering Data Visualization Techniques (Part 2)

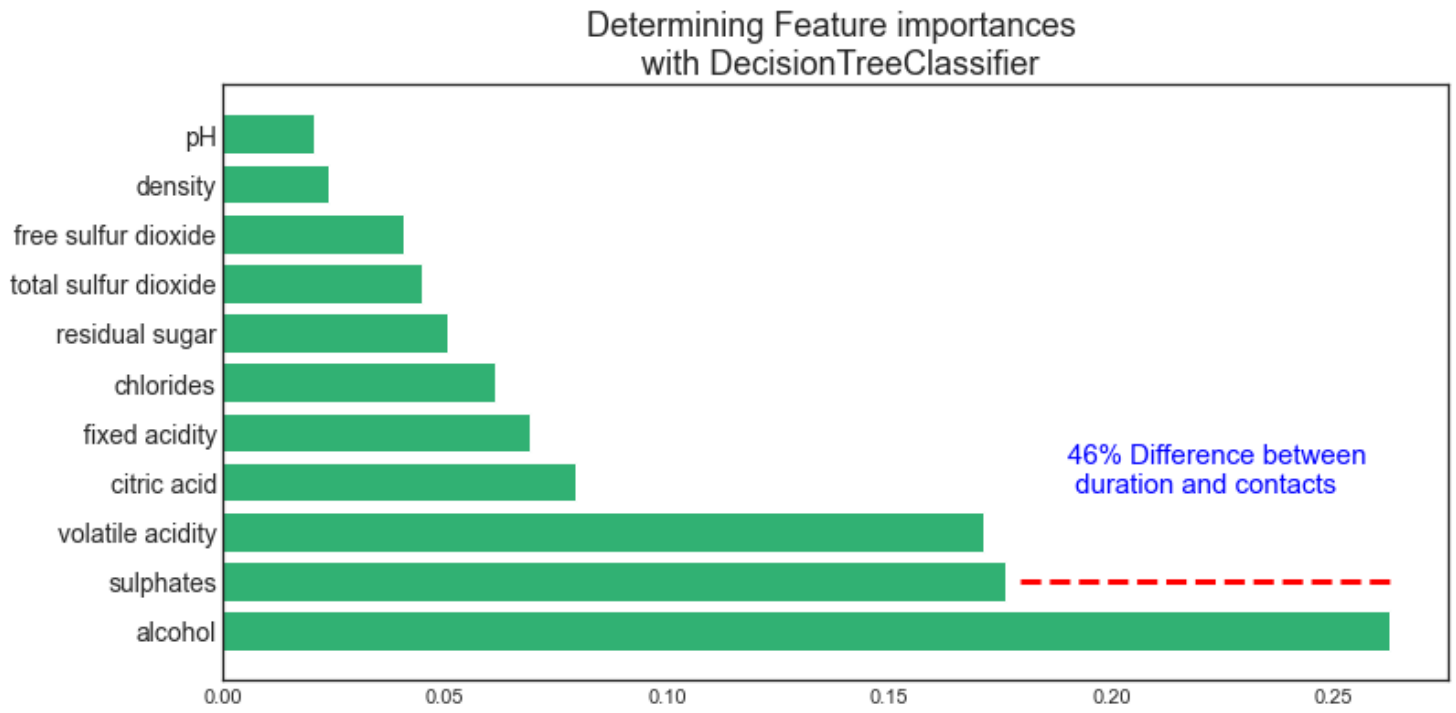
Prepared by: Syed Afroz Ali

Feature Importance Visualization

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
plt.style.use('seaborn-white')

# Create train and test splits
target_name = 'quality'
X = wine.drop('quality', axis=1)
label=wine[target_name]
```

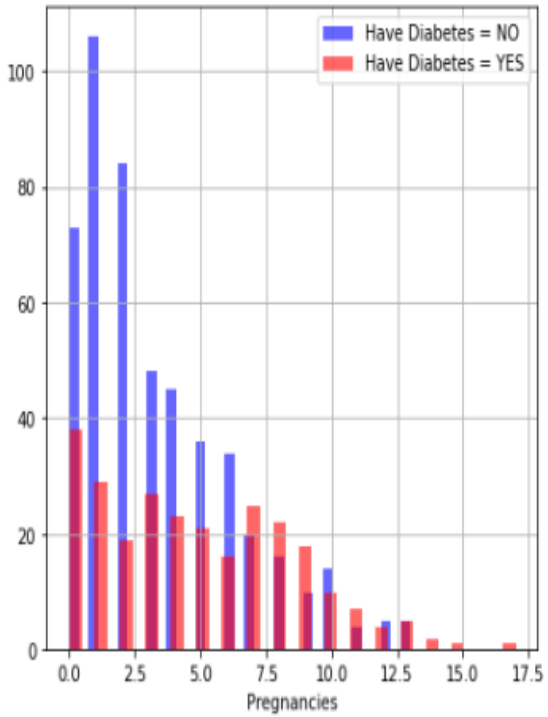
```
X_train, X_test, y_train, y_test = train_test_split(X,label,test_size=0.2,
random_state=42, stratify=label)
# Build a classification task using 3 informative features
tree = tree.DecisionTreeClassifier(
    class_weight='balanced',
    min_weight_fraction_leaf = 0.01
)
tree = tree.fit(X_train, y_train)
importances = tree.feature_importances_
feature_names = wine.drop('quality', axis=1).columns
indices = np.argsort(importances)[::-1]

# Print the feature ranking

for f in range(X_train.shape[1]):
    print("%d. feature %d (%f)" % (f + 1, indices[f],
importances[indices[f]]))
```

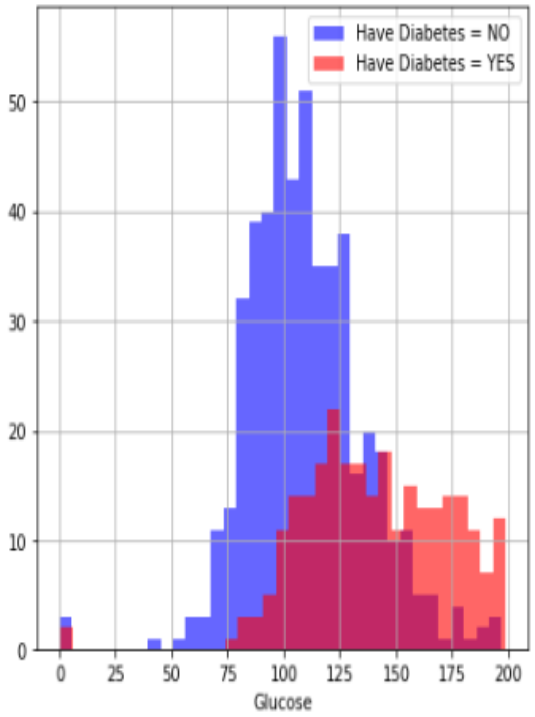
```
# Plot the feature importances of the forest
def feature_importance_graph(indices, importances, feature_names):
    plt.figure(figsize=(12,6))
    plt.title("Determining Feature importances \n with
DecisionTreeClassifier", fontsize=18)
    plt.barh(range(len(indices)), importances[indices],
color='#31B173',  align="center")
    plt.yticks(range(len(indices)), feature_names[indices],
rotation='horizontal',fontsize=14)
    plt.ylim([-1, len(indices)])
    plt.axhline(y=1.0, xmin=0.65, xmax=0.952, color='red', linewidth=3,
linestyle='--')
    plt.text(0.19, 2.8, '46% Difference between \n duration and
contacts', color='Blue', fontsize=15)

feature_importance_graph(indices, importances, feature_names)
plt.show()
```

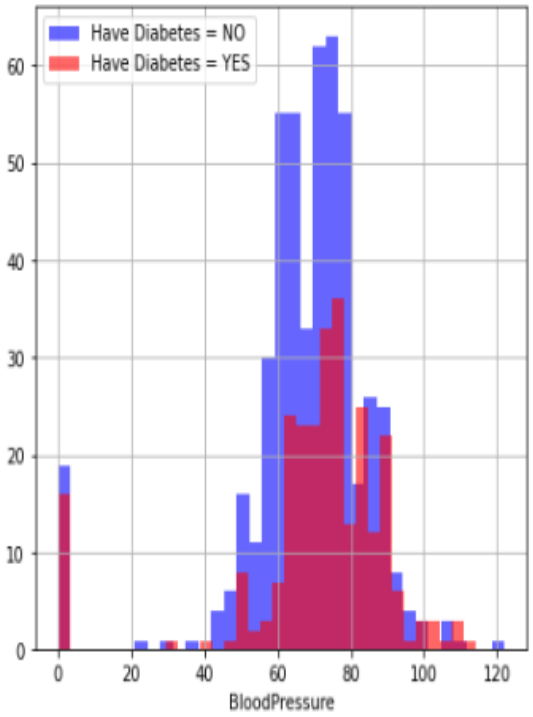



```
# Visulazing the distibution of the data for every feature
plt.figure(figsize=(20, 20))

for i, column in enumerate(df.columns, 1):
    plt.subplot(3, 3, i)
```

```
    df[df["Outcome"] == 0] [column].hist(bins=35, color='blue',
label='Have Diabetes = NO', alpha=0.6)
    df[df["Outcome"] == 1] [column].hist(bins=35, color='red',
label='Have Diabetes = YES', alpha=0.6)
    plt.legend()
    plt.xlabel(column)
```







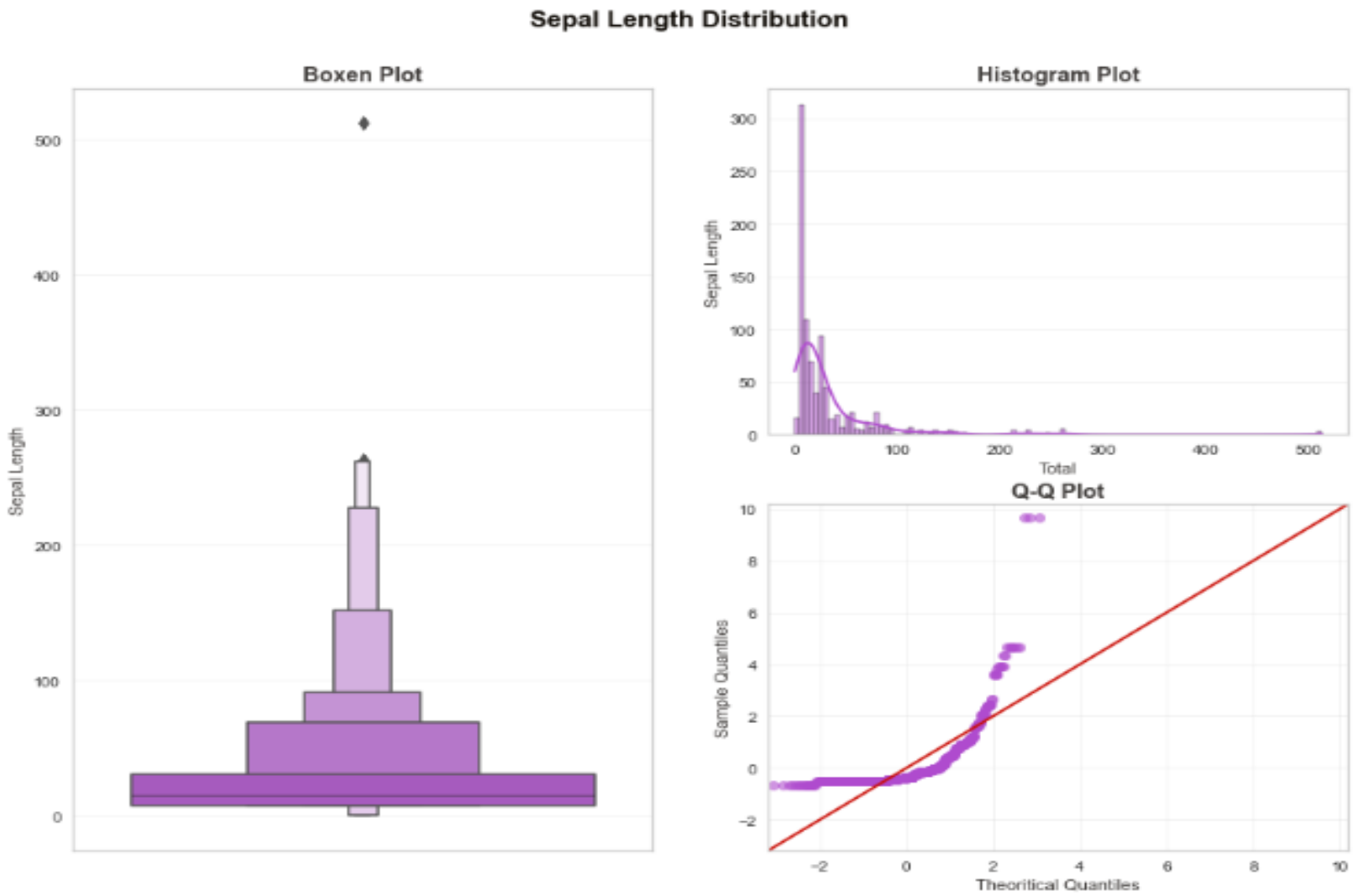
```
from yellowbrick.classifier import ConfusionMatrix
from yellowbrick.classifier import ClassPredictionError
from yellowbrick.classifier import ROCAUC
from yellowbrick.style import set_palette

from statsmodels.graphics.gofplots import qqplot

# --- Variable, Color & Plot Size ---
var = titanic['Fare']
color = color_mix[2]
fig = plt.figure(figsize = (14, 10))

# --- Skewness & Kurtosis ---
print('\033[35m\033[1m'+'.: Sepal Length Skewness & Kurtosis
:.'+'\033[0m')
print('*' * 40)
```

```
print('Skewness:'+'\033[35m\033[1m {:.3f}'.format(var.skew(axis = 0,
skipna = True)))
print('\033[0m'+'Kurtosis:'+'\033[35m\033[1m {:.3f}'.format(var.kurt(axis
= 0, skipna = True)))
print('\n')

# --- General Title ---
fig.suptitle('Sepal Length Distribution', fontweight = 'bold', fontsize =
16, fontfamily = 'sans-serif',
             color = black_grad[0])
fig.subplots_adjust(top = 0.9)

# --- Histogram ---
ax_1=fig.add_subplot(2, 2, 2)
plt.title('Histogram Plot', fontweight = 'bold', fontsize = 14, fontfamily =
'sans-serif', color = black_grad[1])
sns.histplot(data = titanic, x = var, kde = True, color = color)
```

```
plt.xlabel('Total', fontweight = 'regular', fontsize = 11, fontfamily =
'sans-serif', color = black_grad[1])
plt.ylabel('Sepal Length', fontweight = 'regular', fontsize = 11,
fontfamily = 'sans-serif', color = black_grad[1])
plt.grid(axis = 'x', alpha = 0)
plt.grid(axis = 'y', alpha = 0.2)

# --- Q-Q Plot ---
ax_2 = fig.add_subplot(2, 2, 4)
plt.title('Q-Q Plot', fontweight = 'bold', fontsize = 14, fontfamily = 'sans-
serif', color = black_grad[1])
qqplot(var, fit = True, line = '45', ax = ax_2, markerfacecolor = color,
markeredgecolor = color, alpha = 0.6)
plt.xlabel('Theoritical Quantiles', fontweight = 'regular', fontsize = 11,
fontfamily = 'sans-serif',
          color = black_grad[1])
```

```
plt.ylabel('Sample Quantiles', fontweight = 'regular', fontsize = 11,
fontfamily = 'sans-serif', color = black_grad[1])
plt.grid(axis = 'both', alpha = 0.2)

# --- Boxen Plot ---
ax_3 = fig.add_subplot(1, 2, 1)
plt.title('Boxen Plot', fontweight = 'bold', fontsize = 14, fontfamily =
'sans-serif', color = black_grad[1])
sns.boxenplot(y = var, data = titanic, color = color, linewidth = 1.5)
plt.ylabel('Sepal Length', fontweight = 'regular', fontsize = 11,
fontfamily = 'sans-serif', color = black_grad[1])
plt.grid(axis = 'y', alpha = 0.2)
plt.show();
```

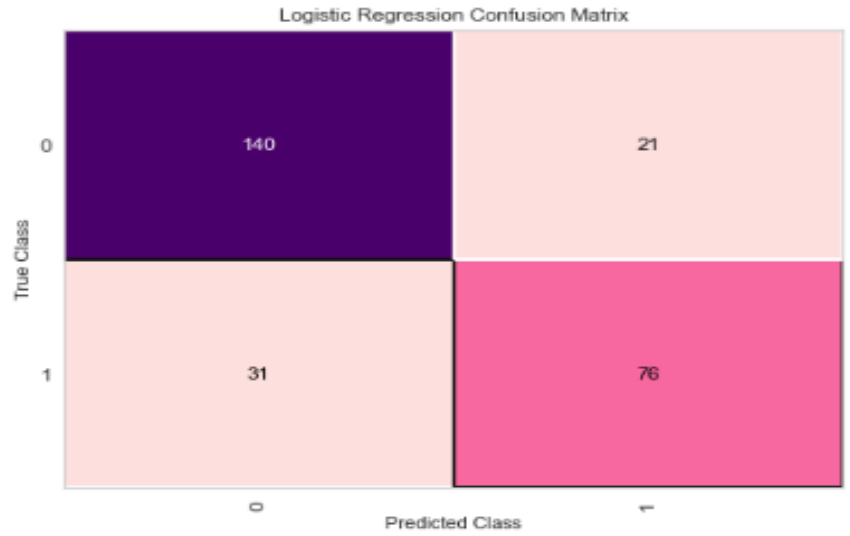
```
.: Sepal Length Skewness & Kurtosis :.
****************************************
Skewness: 4.787
Kurtosis: 33.398
```

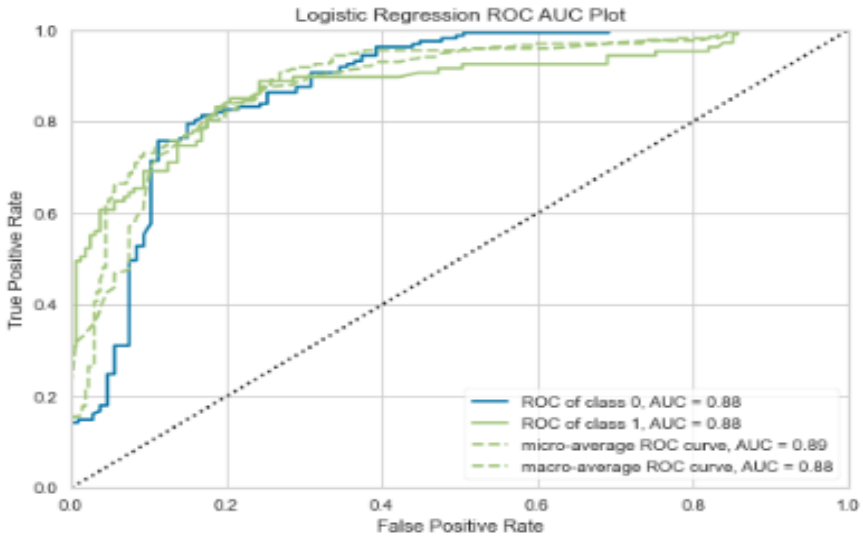


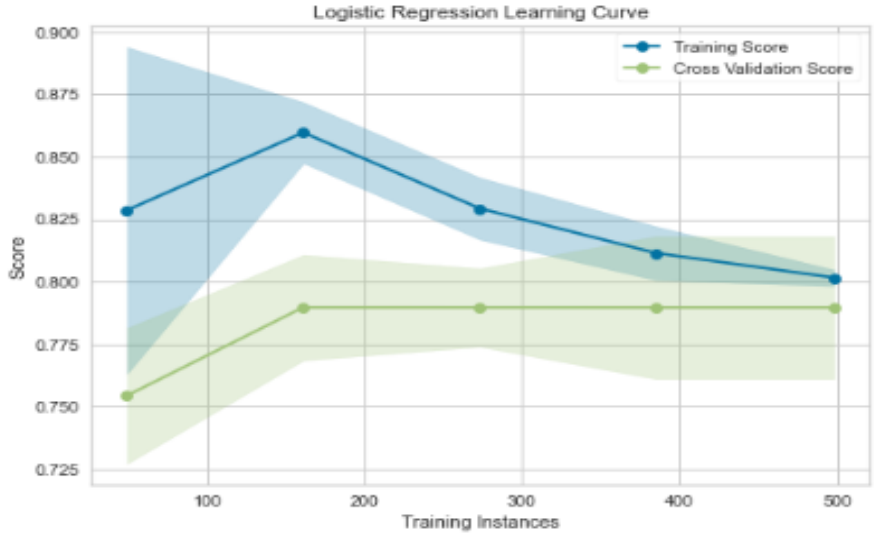
```
from yellowbrick.model_selection import LearningCurve,
FeatureImportances
from sklearn.metrics import
accuracy_score,precision_recall_curve

# --- Applying Logistic Regression ---
LRclassifier = LogisticRegression(solver='liblinear')
LRclassifier.fit(X_train, y_train)

y_pred_LR = LRclassifier.predict(X_test)

# --- LR Accuracy ---
LRAcc = accuracy_score(y_pred_LR, y_test)
print('.:. Logistic Regression Accuracy:'+'\033[35m\033[1m
{:.2f}%'.format(LRAcc*100)+' \033[0m.:.')
```

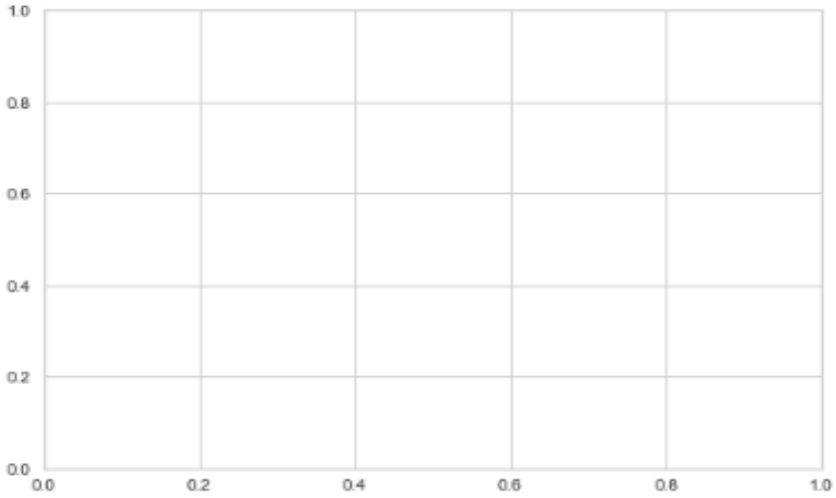
```
# --- LR Classification Report ---
print('\033[35m\033[1m\n.: Classification Report'+'\033[0m')
print('*' * 25)
print(classification_report(y_test, y_pred_LR))

# --- Performance Evaluation ---
print('\033[35m\n\033[1m'+'.: Performance
Evaluation'+'\033[0m')
print('*' * 26)
fig, ((ax1, ax2), (ax3, ax4)) = plt.subplots(2, 2, figsize = (14, 10))

# --- LR Confusion Matrix ---
logmatrix = ConfusionMatrix(LRclassifier, ax=ax1,
cmap='RdPu', title='Logistic Regression Confusion Matrix')
logmatrix.fit(X_train, y_train)
logmatrix.score(X_test, y_test)
```

```
logmatrix.finalize()

# --- LR ROC AUC ---
logrocauc = ROCAUC(LRclassifier, ax = ax2, title = 'Logistic
Regression ROC AUC Plot')
logrocauc.fit(X_train, y_train)
logrocauc.score(X_test, y_test)
logrocauc.finalize()

# --- LR Learning Curve ---
loglc = LearningCurve(LRclassifier, ax = ax3, title = 'Logistic
Regression Learning Curve')
loglc.fit(X_train, y_train)
loglc.finalize()
plt.tight_layout();
```

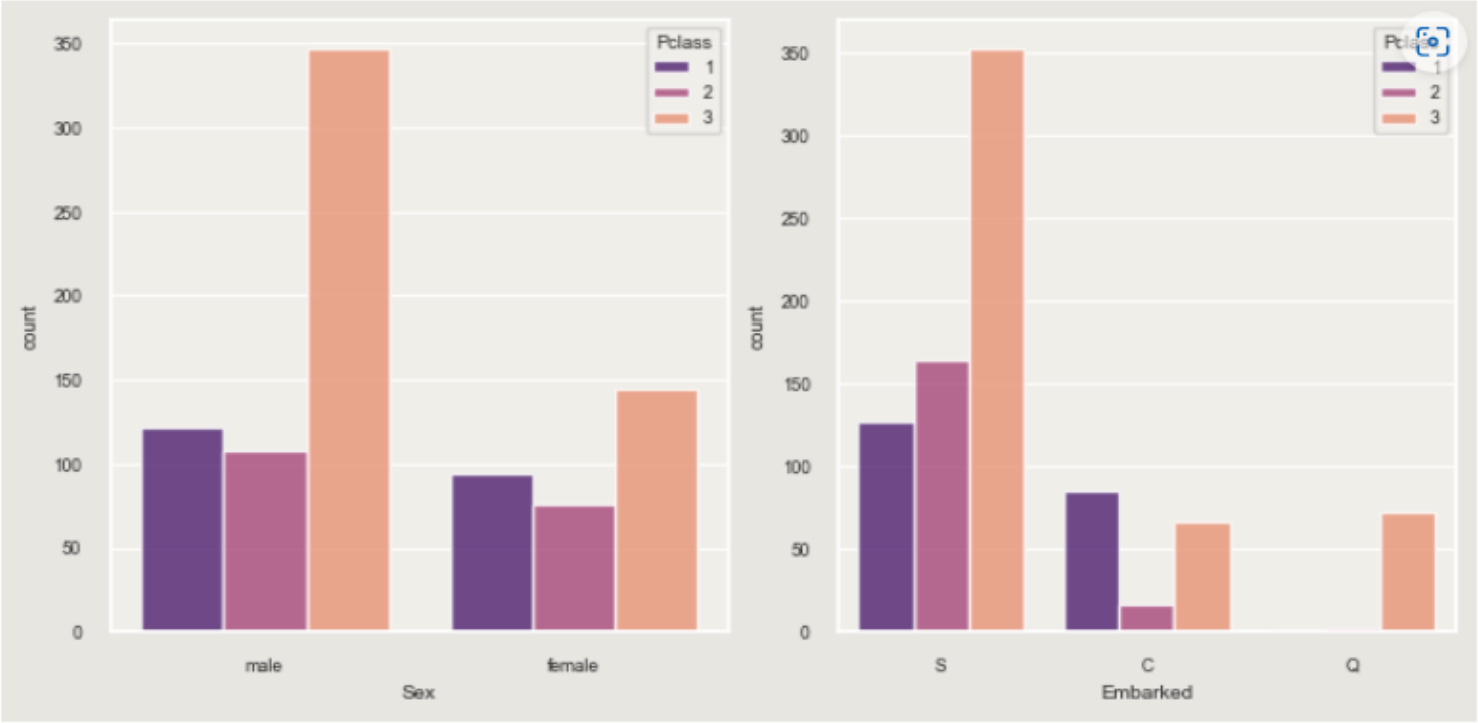

Logistic Regression Confusion Matrix
140
21
31
76
0
1
True Class
0
1
Predicted Class


Logistic Regression ROC AUC Plot
True Positive Rate
False Positive Rate
ROC of class 0, AUC = 0.88
ROC of class 1, AUC = 0.88
micro-average ROC curve, AUC = 0.89
macro-average ROC curve, AUC = 0.88


Logistic Regression Learning Curve
Training Score
Cross Validation Score
Score
Training Instances

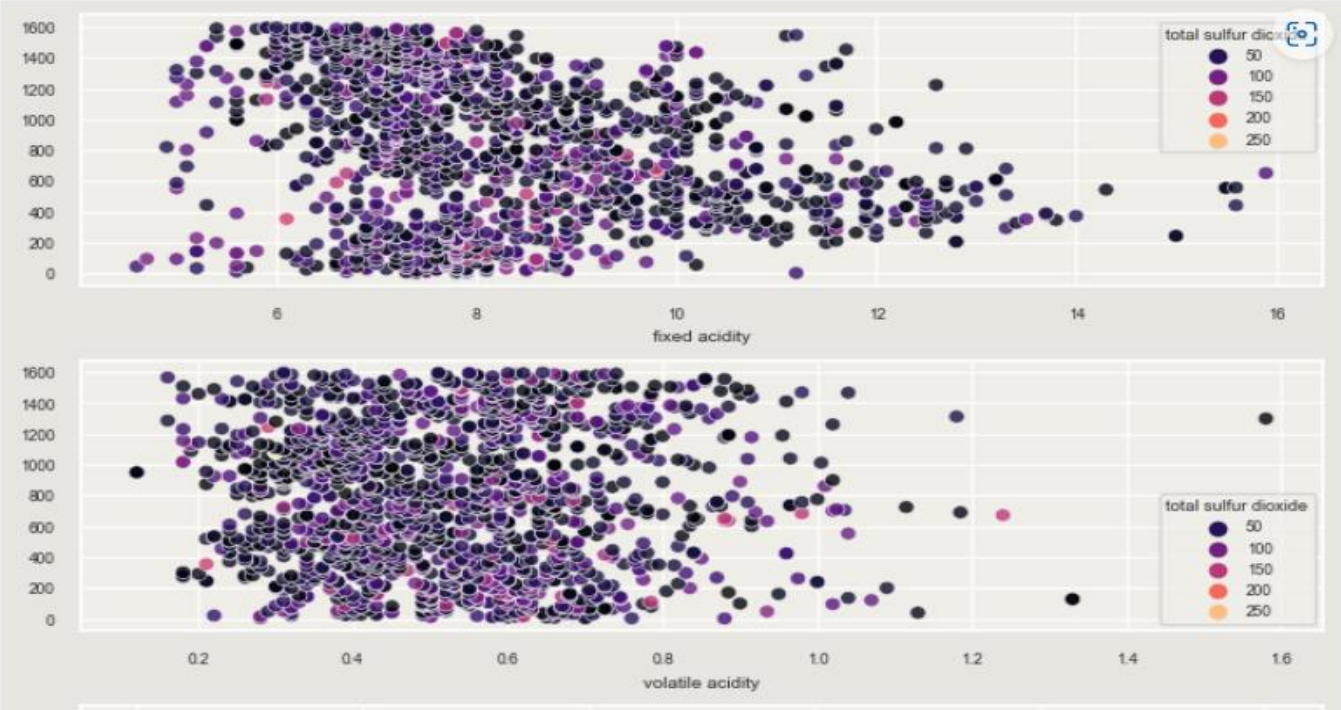


```
cat = ['Sex','Embarked']
sns.set_theme(rc = {'figure.dpi': 100, 'axes.labelsize': 7,
                    'axes.facecolor': '#f0eee9', 'grid.color': '#fffdfa',
                   'figure.facecolor': '#e8e6e1'}, font_scale = 0.55)
fig, ax = plt.subplots(5, 2, figsize = (7, 18))
for indx, (column, axes) in list(enumerate(list(zip(cat,
                                                    ax.flatten())))):

    sns.countplot(ax = axes, x = titanic[column], hue = titanic['Pclass'],
                  palette = 'magma', alpha = 0.8)

else:
    [axes.set_visible(False) for axes in ax.flatten()[indx + 1:]]
plt.tight_layout()
plt.show()
```

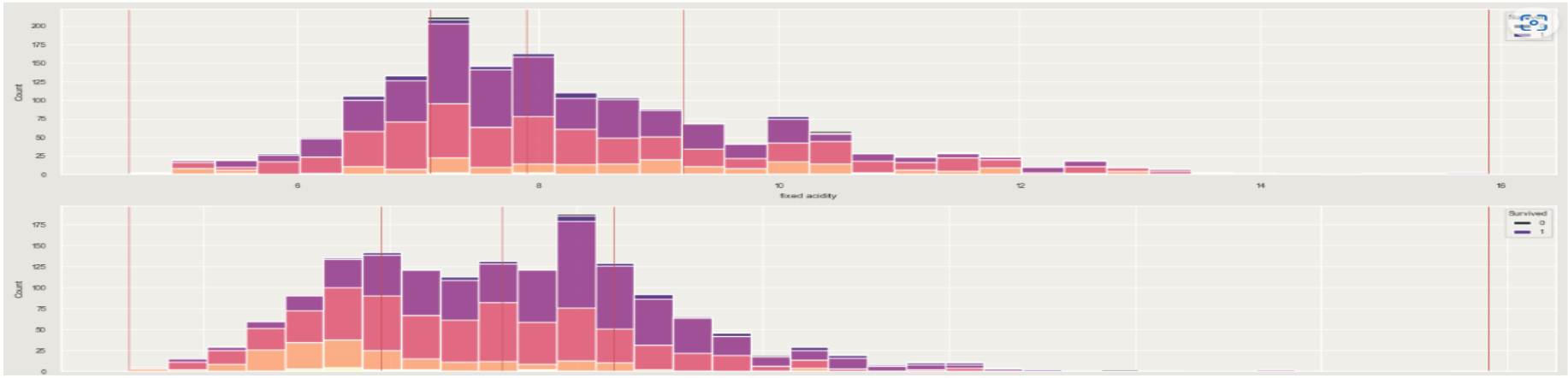



```
num = wine.select_dtypes(include="number")
fig, ax = plt.subplots(14, 1, figsize = (7, 30))
for indx, (column, axes) in list(enumerate(list(zip(num, ax.flatten())))):

    sns.scatterplot(ax = axes, y = wine[column].index, x =
wine[column],hue = wine['total sulfur dioxide'],
```

```
                palette = 'magma', alpha = 0.8)

else:
    [axes.set_visible(False) for axes in ax.flatten()[indx + 1:]]
plt.tight_layout()
plt.show()
```

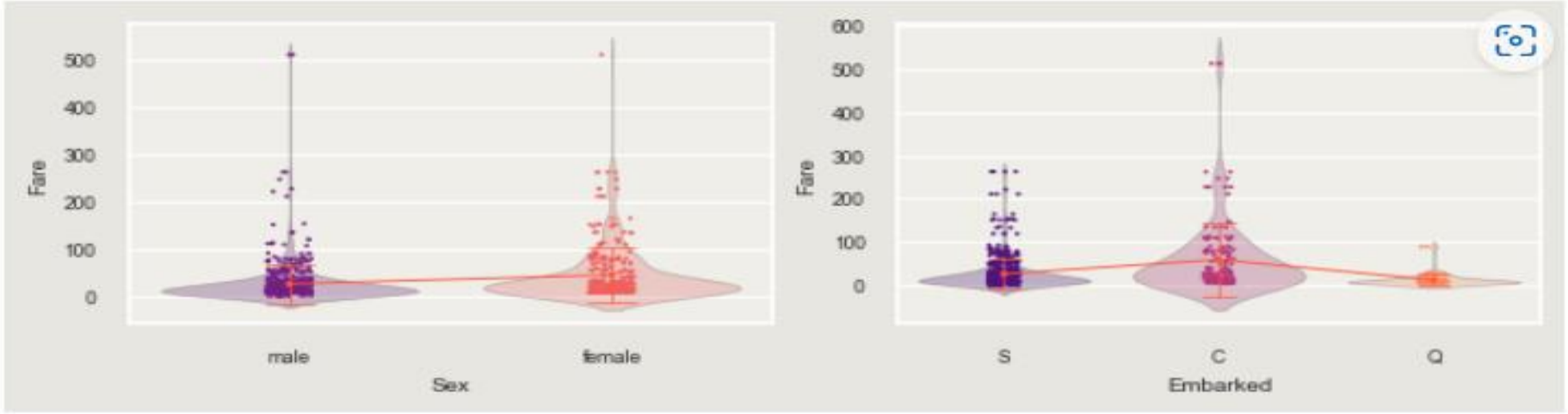


```
num = wine.select_dtypes(include="number")
fig, ax = plt.subplots(12, 1, figsize = (14, 35))
for indx, (column, axes) in list(enumerate(list(zip(num, ax.flatten())))):

    sns.histplot(ax = axes, x = wine[column],hue = wine['quality'],
                 palette = 'magma', alpha = 0.8, multiple = 'stack')
    legend = axes.get_legend() # sns.hisplot has some issues with
legend
    handles = legend.legendHandles
    legend.remove()
    axes.legend(handles, ['0', '1'], title = 'Survived', loc = 'upper right')
    Quantiles = np.quantile(wine[column], [0, 0.25, 0.50, 0.75, 1])

    for q in Quantiles: axes.axvline(x = q, linewidth = 0.5, color =
'r')
plt.tight_layout()
plt.show()
```

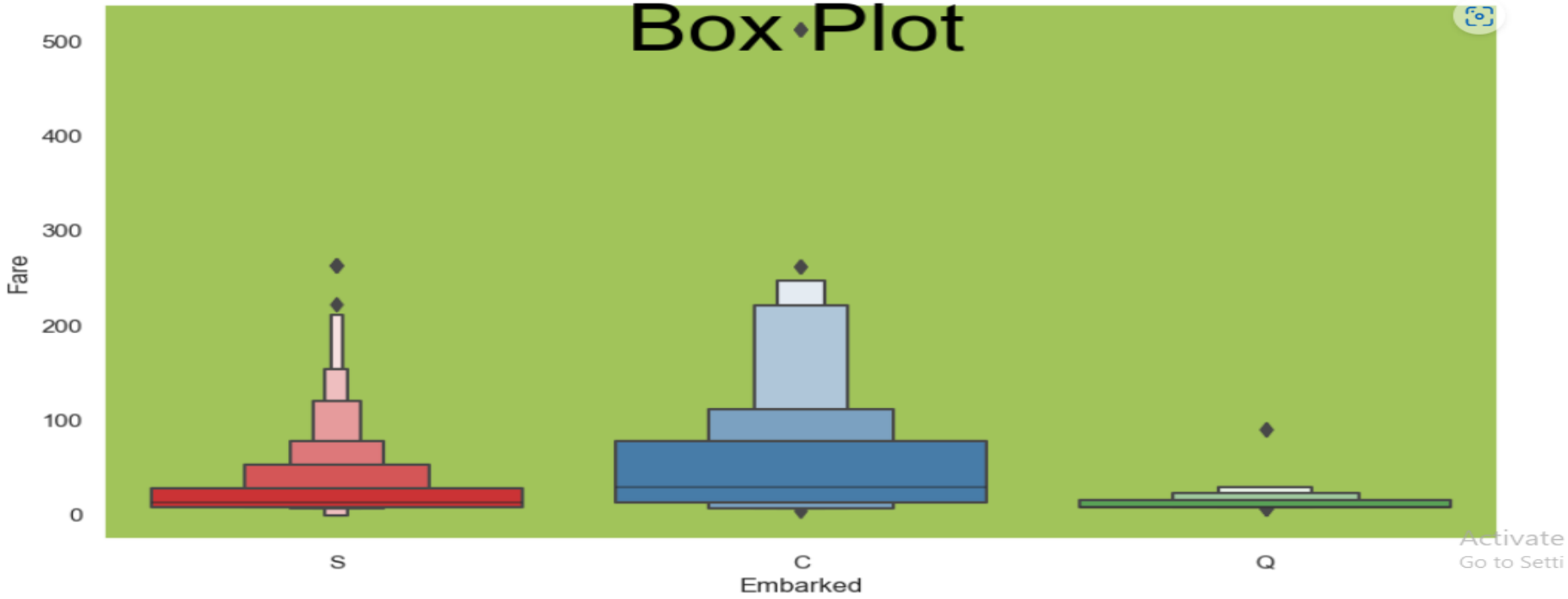



```
cat = ['Sex','Embarked']
fig, ax = plt.subplots(5, 2, figsize = (6.5, 10))
for indx, (column, axes) in list(enumerate(list(zip(cat, ax.flatten())))):
    sns.violinplot(ax = axes, x = titanic[column],
                   y = titanic['Fare'],
                   scale = 'width', linewidth = 0.5,
                   palette = 'magma', inner = None)
    plt.setp(axes.collections, alpha = 0.3)
    sns.stripplot(ax = axes, x = titanic[column],
                  y = titanic['Fare'],
                  palette = 'magma', alpha = 0.9,
                  s = 1.5, jitter = 0.07)
```

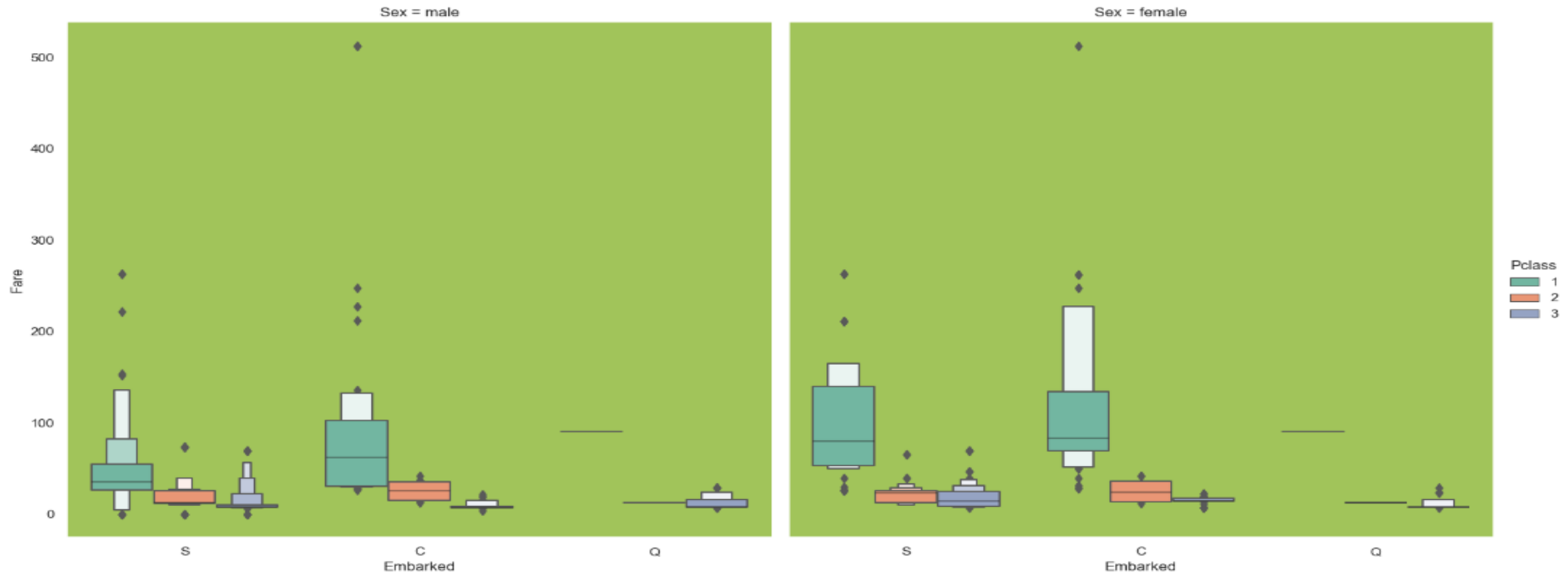
```
    sns.pointplot(ax = axes, x = titanic[column],
                  y = titanic['Fare'],
                  color = '#ff5736', scale = 0.25,
                  estimator = np.mean, ci = 'sd',
                  errwidth = 0.5, capsize = 0.15, join = True)
    plt.setp(axes.lines, zorder = 100)
    plt.setp(axes.collections, zorder = 100)
else:
    [axes.set_visible(False) for axes in ax.flatten()[indx + 1:]]
plt.tight_layout()
plt.show()
```



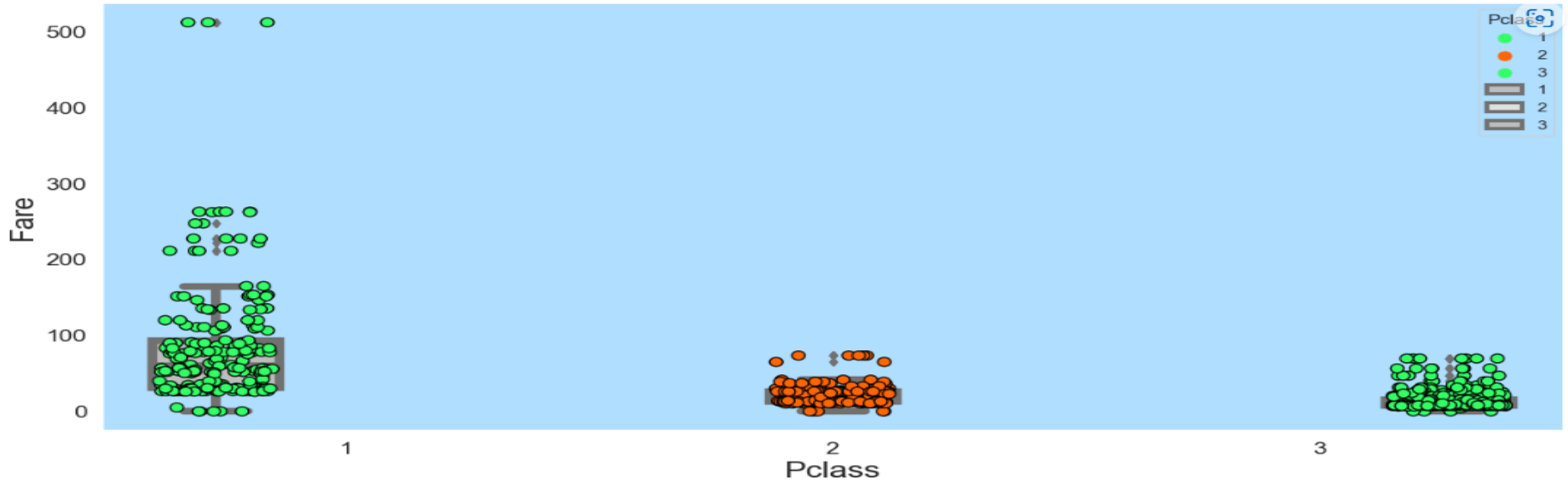
```
sns.set(rc={"axes.facecolor":"#a1c45a" , "axes.grid" : False})
plt.figure(figsize=(11,6))
plt.gcf().text(.51, .84, "Box Plot", fontsize = 40, color='Black' ,ha='center',
va='center')
sns.boxenplot(x=titanic['Embarked'] , y = titanic['Fare'],palette="Set1")
plt.show()
```



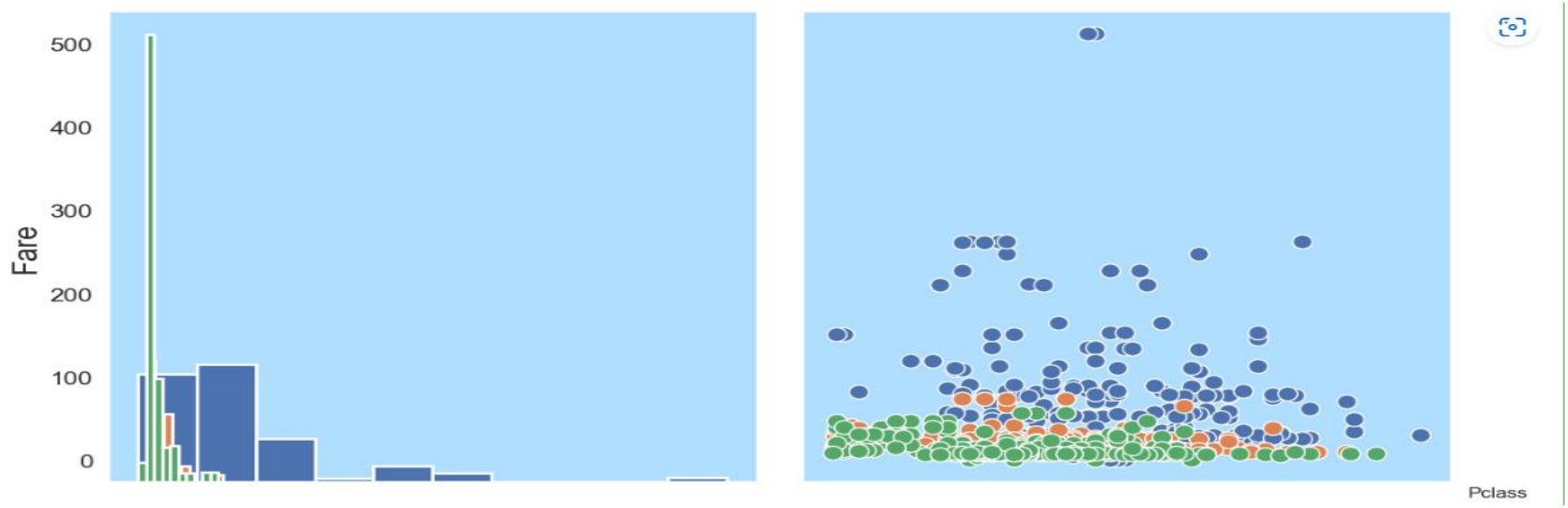
```
# Facet along the columns to show a categorical variable using "col"
plt.figure(figsize=(11,7))
sns.catplot(x="Embarked" , y = "Fare", hue= "Pclass",
            col="Sex", kind="boxen",palette="Set2" , height=8, aspect=1
,data=titanic)
plt.show();
```



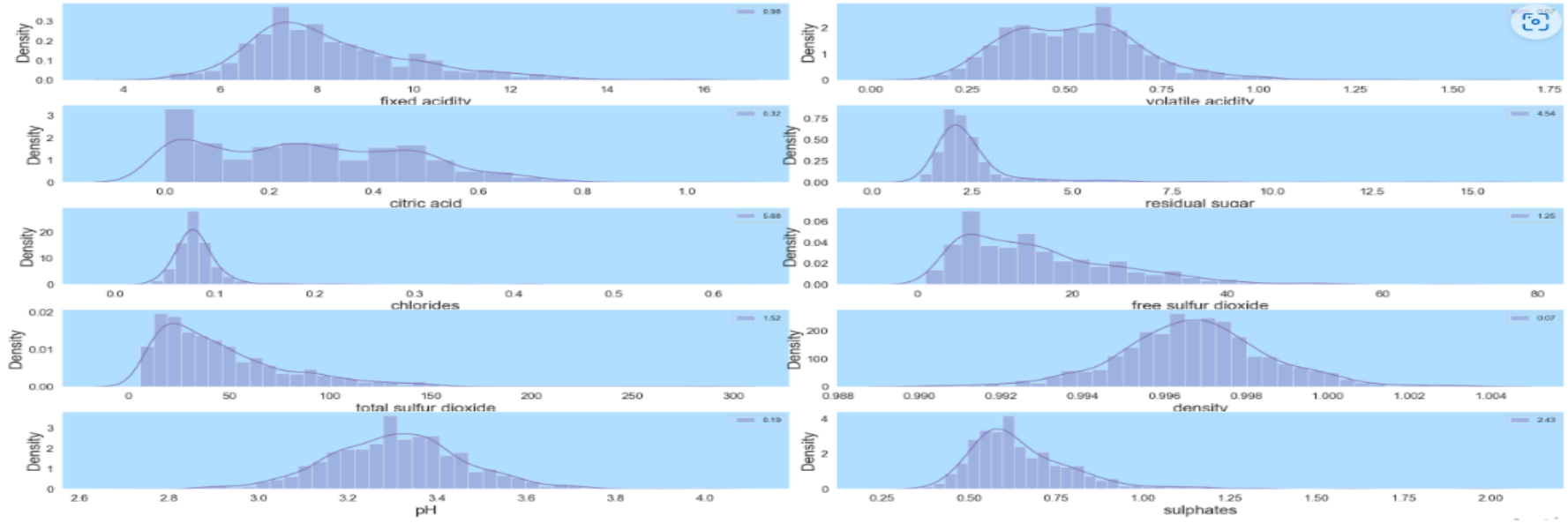
```
plt.figure(figsize=(16,7))
sns.set(rc={"axes.facecolor":"#b0deff","axes.grid":False,
            'xtick.labelsize':15,'ytick.labelsize':15,
            'axes.labelsize':20,'figure.figsize':(20.0, 9.0)})
params = dict(data=titanic ,x = titanic.Pclass ,y = titanic.Fare
,hue=titanic.Pclass,dodge=True)
sns.stripplot(**params ,
size=8,jitter=0.35,palette=['#33FF66','#FF6600'],edgecolor='black',linewidth=1)
sns.boxplot(**params ,palette=['#BDBDBD','#E0E0E0'],linewidth=6)
plt.show()
```



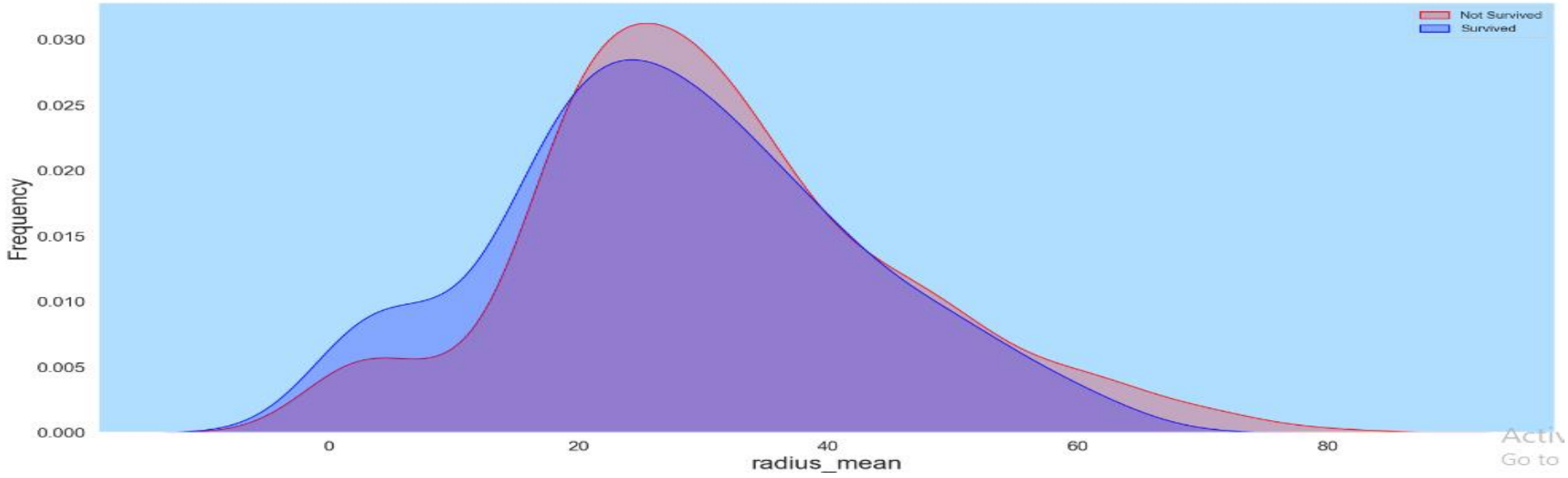
```
# Plot a subset of variables
g = sns.PairGrid(titanic, hue='Pclass' ,x_vars=["Fare" , "Age"],y_vars=["Fare" ,
"Age"],
                 height=6, aspect=1)
g = g.map_offdiag(plt.scatter , edgecolor="w", s=130)
g = g.map_diag(plt.hist , edgecolor ='w', linewidth=2)
g = g.add_legend()
plt.show()
```



```
df = pd.read_csv("winequality-red.csv")
features_mean= list(df.columns[:10])
num_rows, num_cols = 5,2
fig, axes = plt.subplots(num_rows, num_cols, figsize=(25, 12))
fig.tight_layout()
for index, column in enumerate(df[features_mean].columns):
    i,j = (index // num_cols, index % num_cols)
    g = sns.distplot(df[column], color="m", label="%.2f"%(df[column].skew()),
ax=axes[i,j])
    g = g.legend(loc="best")
```

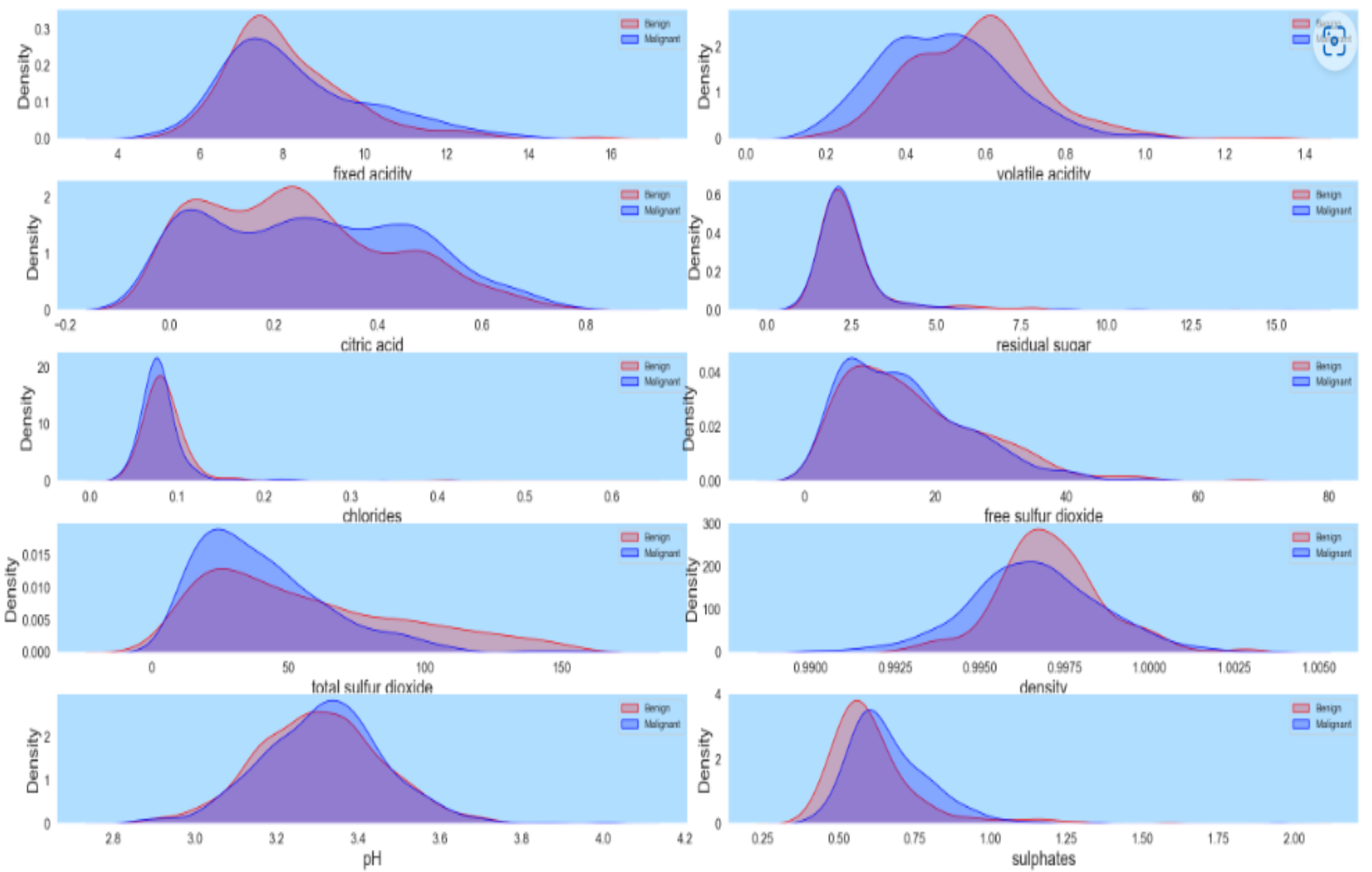


```
y = df['Sex']

# Explore Age distibution
g = sns.kdeplot(df["Age"][(y == 'male') & (df["Age"].notnull())], color="Red",
shade=True)
g = sns.kdeplot(df["Age"][(y == 'female') & (df["Age"].notnull())], ax=g,
color="Blue", shade=True)
g.set_xlabel("radius_mean")
g.set_ylabel("Frequency")
g = g.legend(["Not Survived","Survived"])
```

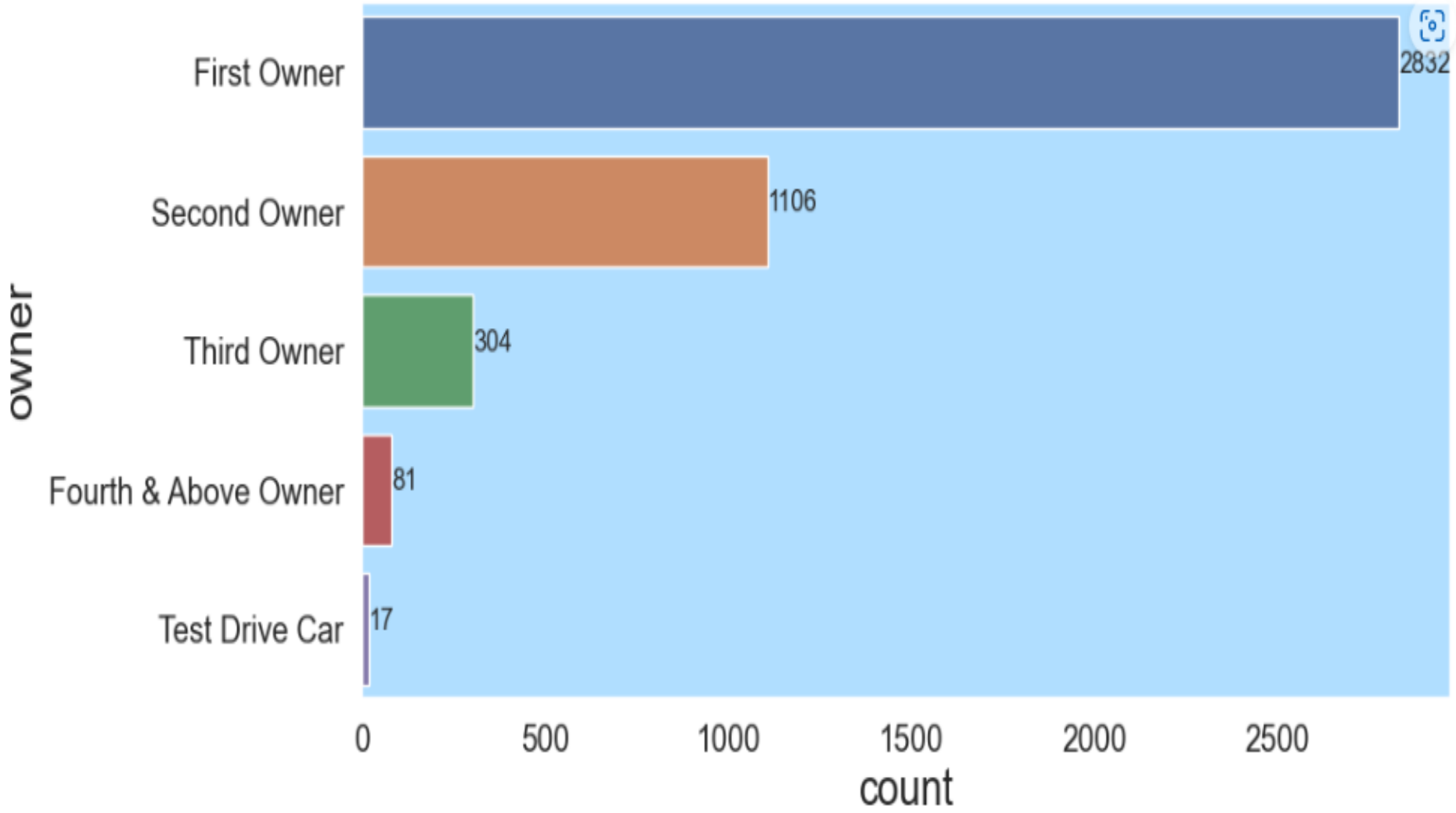


```
df = pd.read_csv("winequality-red.csv")
features_mean= list(df.columns[:10])
df_b = df[df['quality'] == 5]
df_m = df[df['quality'] == 6]
num_rows, num_cols = 5,2
fig, axes = plt.subplots(num_rows, num_cols, figsize=(25, 12))
fig.tight_layout()
for index, column in enumerate(df[features_mean].columns):
    i,j = (index // num_cols, index % num_cols)
    g = sns.kdeplot(df_b[column], color="Red", shade=True,
ax=axes[i,j])
    g = sns.kdeplot(df_m[column], ax=g, color="Blue", shade=True)
    g.set_xlabel(column)
    g = g.legend(["Benign","Malignant"])
```

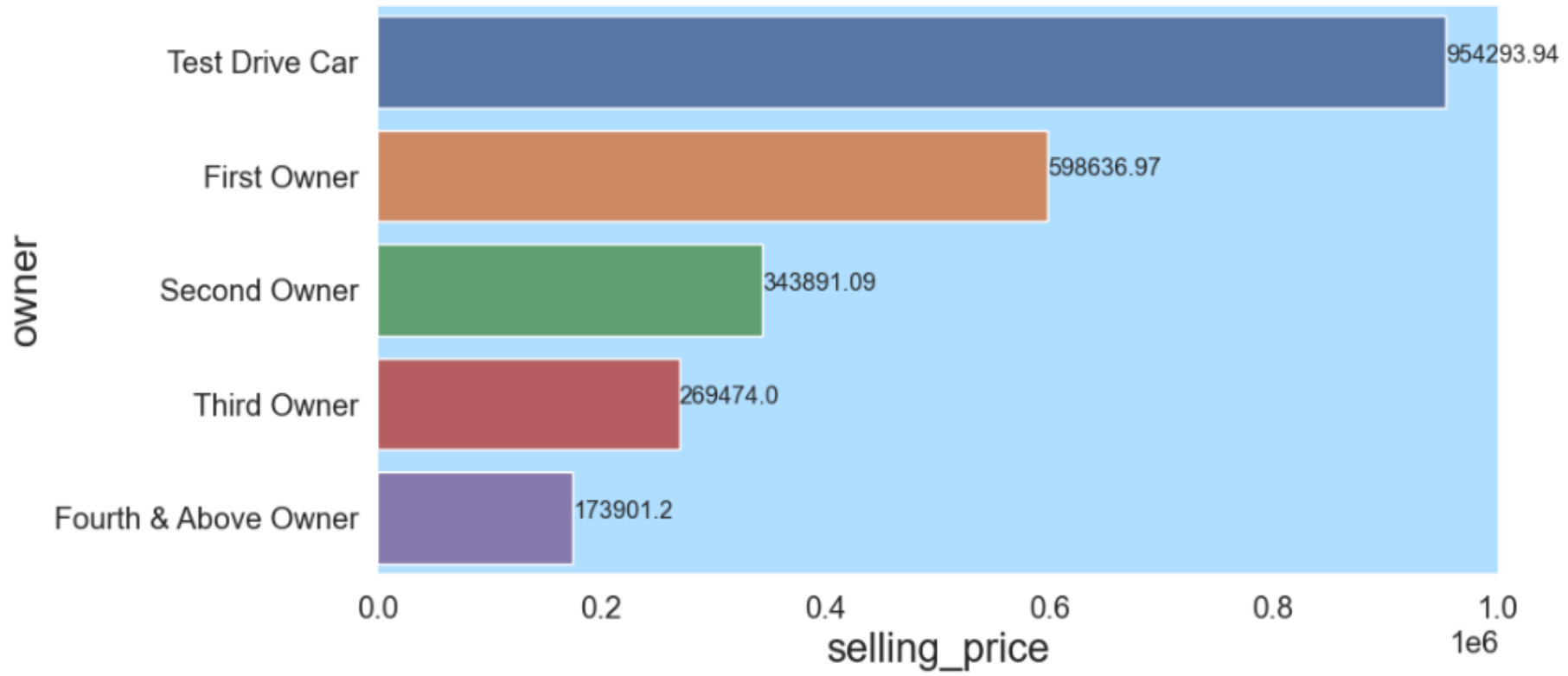
Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625



```
raw_df = raw_df [['name', 'year', 'selling_price', 'km_driven',
'fuel', 'seller_type',
       'transmission', 'owner']]
# Function to print width of barcharts on the bars
def barw(ax):
    for p in ax.patches:
        val = p.get_width() #height of the bar
        x = p.get_x()+ p.get_width() # x- position
        y = p.get_y() + p.get_height()/2 #y-position
        ax.annotate(round(val,2),(x,y))
plt.figure(figsize=(10,5))
ax0 = sns.countplot(data = raw_df, y ='owner', order =
raw_df['owner'].value_counts().index)
barw(ax0)
plt.show()
```

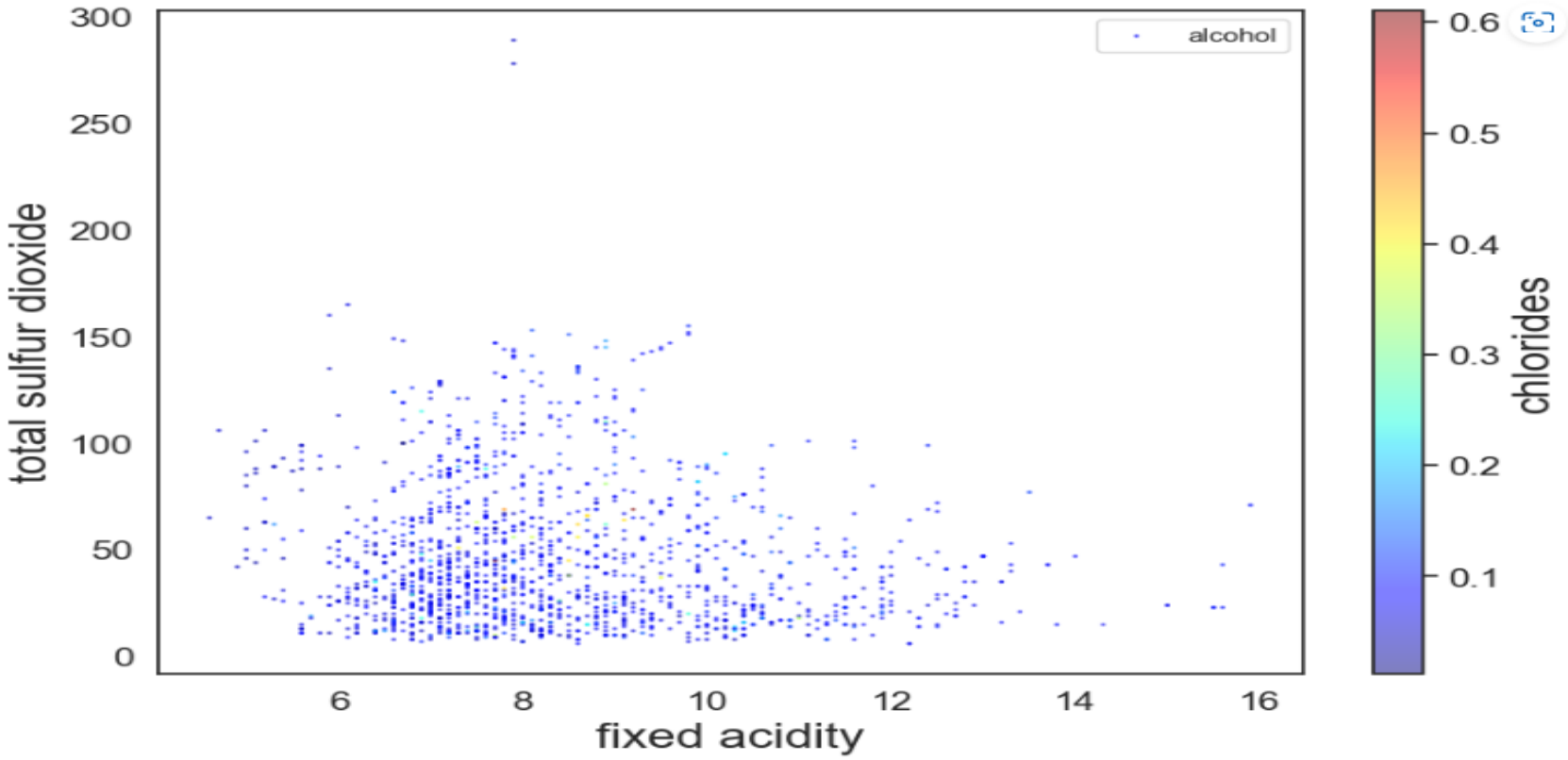

First Owner
Second Owner
Third Owner
Fourth & Above Owner
Test Drive Car
owner
2832
1106
304
81
17
0
500
1000
1500
2000
2500
count

```
raw_df = pd.read_csv('datasets_33080_1320127_CAR DETAILS FROM
CAR DEKHO.csv')
raw_df = raw_df [['name', 'year', 'selling_price', 'km_driven', 'fuel',
'seller_type',
       'transmission', 'owner']]
df_gc = raw_df.groupby('owner').mean()
df_gc.reset_index(inplace= True)
df_gc[['owner','selling_price']].sort_values('selling_price', ascending
=False)

plt.figure(figsize=(10,5))
ax1 = sns.barplot(data = raw_df, x='selling_price', y ='owner', order =
df_gc.sort_values('selling_price',ascending =False)['owner'], ci =None)
barw(ax1)
plt.show()
```

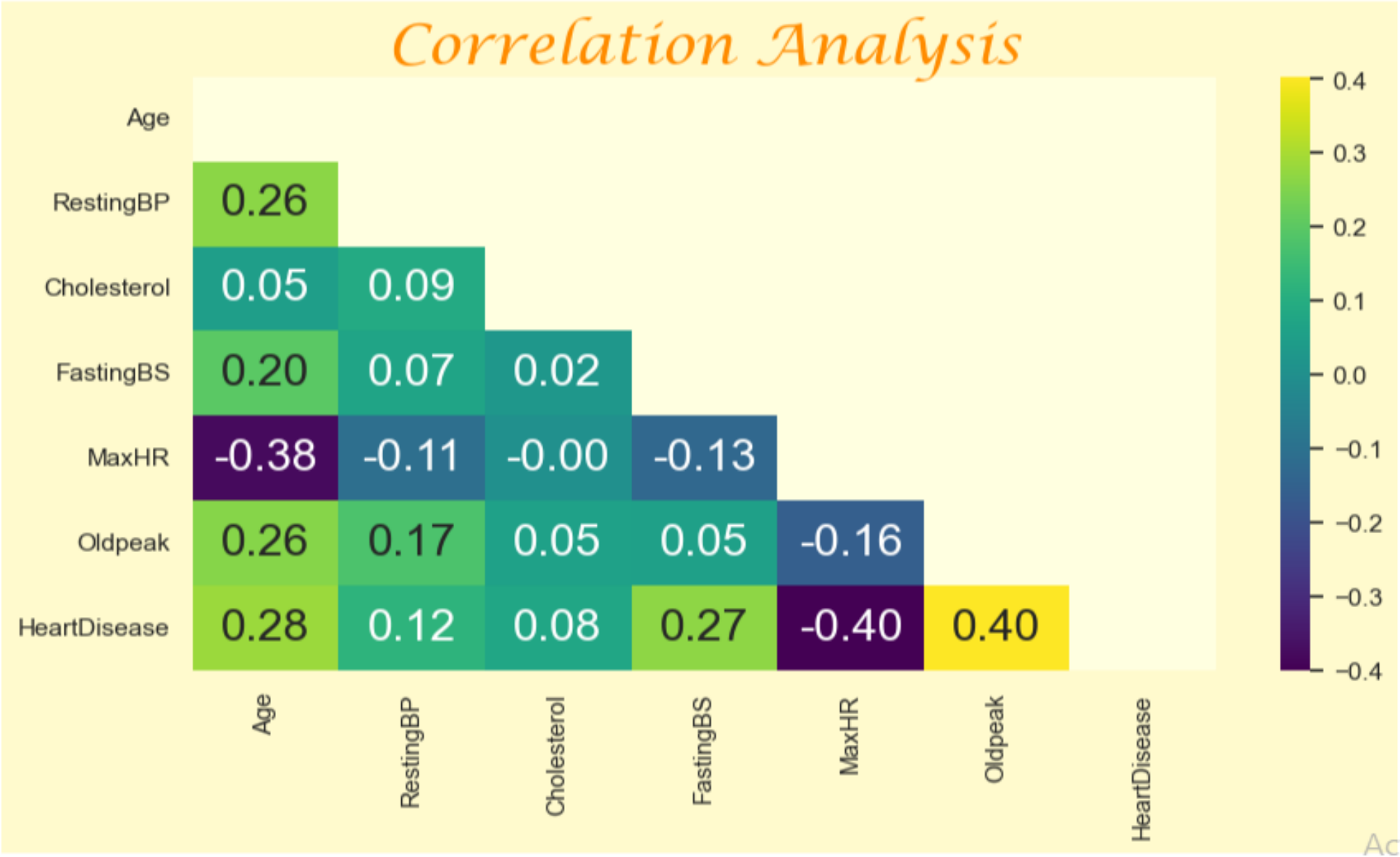


```
sns.set_style('white')
df.plot(kind="scatter", x="fixed acidity", y="total sulfur dioxide",
alpha=.5,
        s=df["alcohol"]/10, label="alcohol", figsize=(10,7),
        c="chlorides", cmap=plt.get_cmap("jet"), colorbar=True,
```

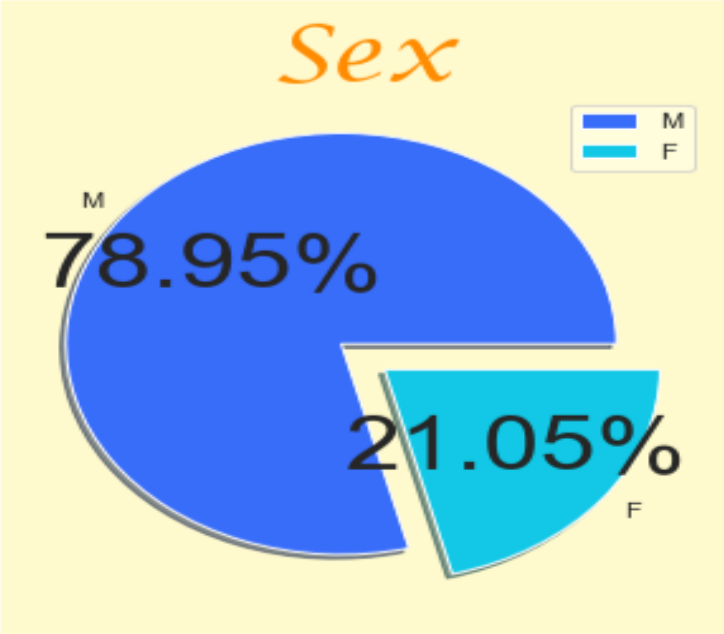
```
            sharex=False)
plt.legend()
plt.show()
```



```
matplotlib.rcParams.update({'font.size': 20})
corr = heart.corr()
mask = np.triu(np.ones_like(corr, dtype=bool))
plt.figure(dpi=100)
plt.title('Correlation Analysis',
          fontsize=25,
          color='DarkOrange',
          font='Lucida Calligraphy')
sns.heatmap(corr,
            mask=mask,
            annot=True,
            lw=0,
            linecolor='white',
            cmap='viridis',
            fmt="0.2f")
plt.xticks(rotation=90)
plt.yticks(rotation=0)
plt.show()
```


Correlation Analysis
Age
RestingBP
Cholesterol
FastingBS
MaxHR
Oldpeak
HeartDisease
0.26
0.05 0.09
0.20 0.07 0.02
-0.38 -0.11 -0.00 -0.13
0.26 0.17 0.05 0.05 -0.16
0.28 0.12 0.08 0.27 -0.40 0.40
Age
RestingBP
Cholesterol
FastingBS
MaxHR
Oldpeak
HeartDisease
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.4

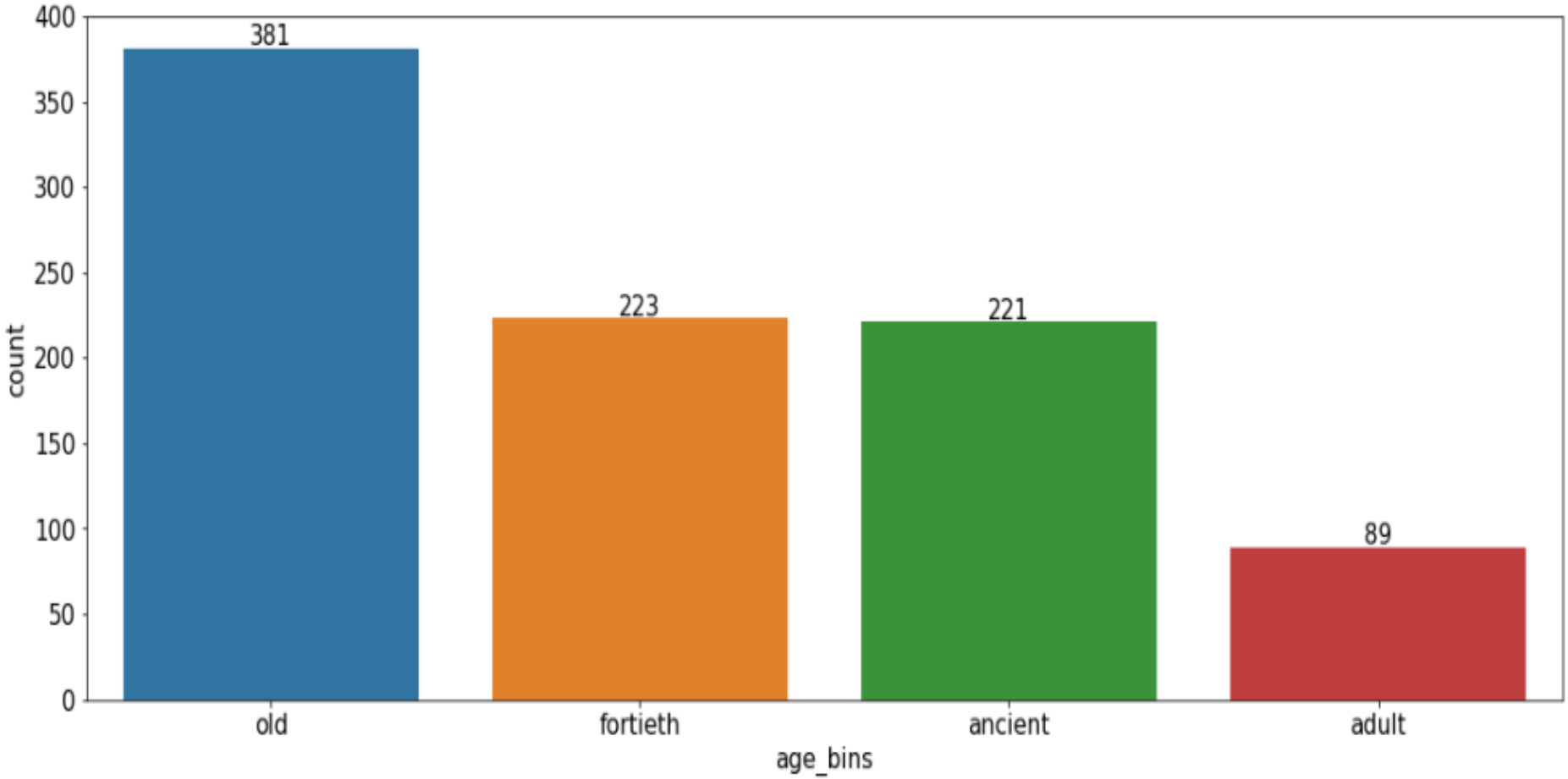
```
matplotlib.rcParams.update({'font.size': 40})
ax=heart['Sex'].value_counts().plot.pie(explode=[0.1,
0.1],autopct='%1.2f%%',shadow=True);
ax.set_title(label = "Sex", fontsize =
40,color='DarkOrange',font='Lucida Calligraphy');
plt.legend(labels=['M','F'])
plt.axis('off');
```



```
heart["age_bins"]= pd.cut(heart["Age"] , bins=[29 , 40 , 50 , 60
,  80] , labels=["adult" , "fortieth" , "old" , "ancient"] )
def count_plot(data  , x=None , y=None  , figsize =None , title
=None , color =None , prop=False , rotation_x =0 ):
    if x is None and y is None :
        raise("Expected y or x")
    if x is not None and y is not None:
        raise("Expected y or x not both")
    count_type = data[y if x is None else
x].value_counts(ascending =False)
    Sum = count_type.sum()
    type_order = count_type.index
    plt.figure(figsize=figsize if figsize is None else (12 , 7))
    if x is None:
        sns.countplot(data = data , y=y , color = color
,order=type_order)
```

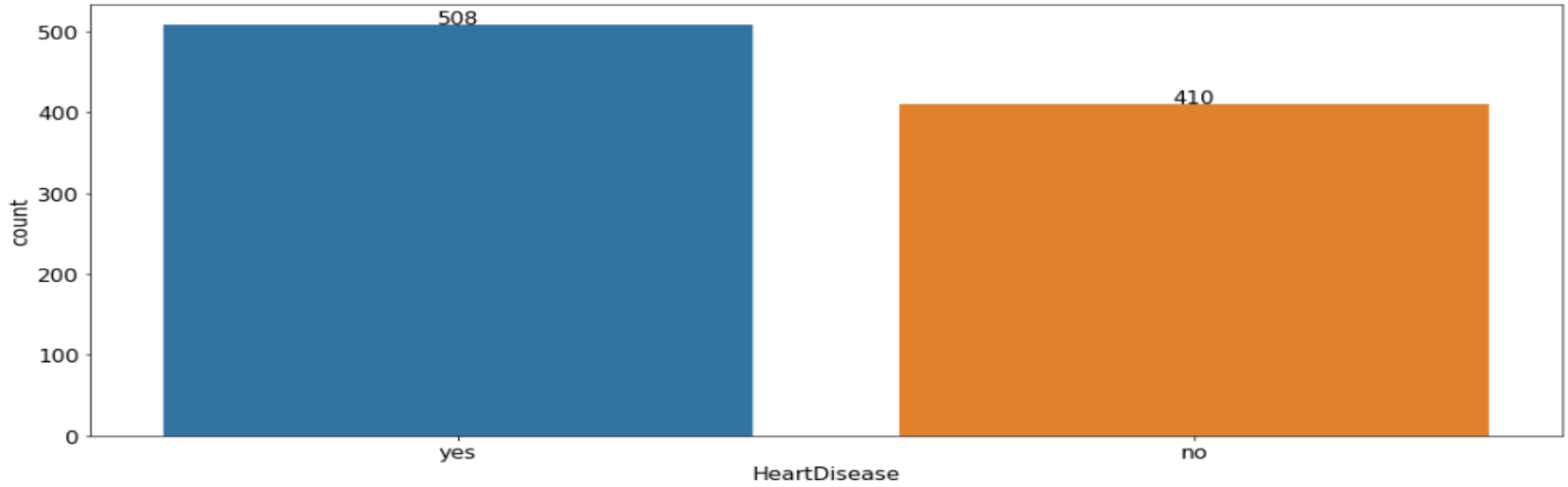
```
        if prop==True:
            for i in range(len(count_type)):
                count = count_type[i]
                pct_string ="{:0.1f}%".format(100*count/Sum)
                plt.text(count+1 , i , pct_string , va="center")
        if prop==False:
            for i in range(len(count_type)):
                count = count_type[i]
                pct_string ="{}".format(count)
                plt.text(count+1 , i , pct_string , va="center")
        plt.title(title)
        plt.show()
    if y is None :
        sns.countplot(data = data , x = x , color = color , order =
type_order)
        locs , labels =plt.xticks(rotation = rotation_x)
```

```
    if prop == True :
        for loc , label in zip(locs , labels):
            count = count_type[label.get_text()]
            pct_string ="{:0.1f}%".format(100*count/Sum)
            plt.text(loc , count+2 ,pct_string,ha ="center")
    if prop==False :
        for loc , label in zip(locs , labels):
            count = count_type[label.get_text()]
            pct_string ="{}".format(count)
            plt.text(loc , count+2 ,pct_string,ha ="center")
    plt.title(title)
    plt.show()


count_plot(data = heart , x ="age_bins")
```

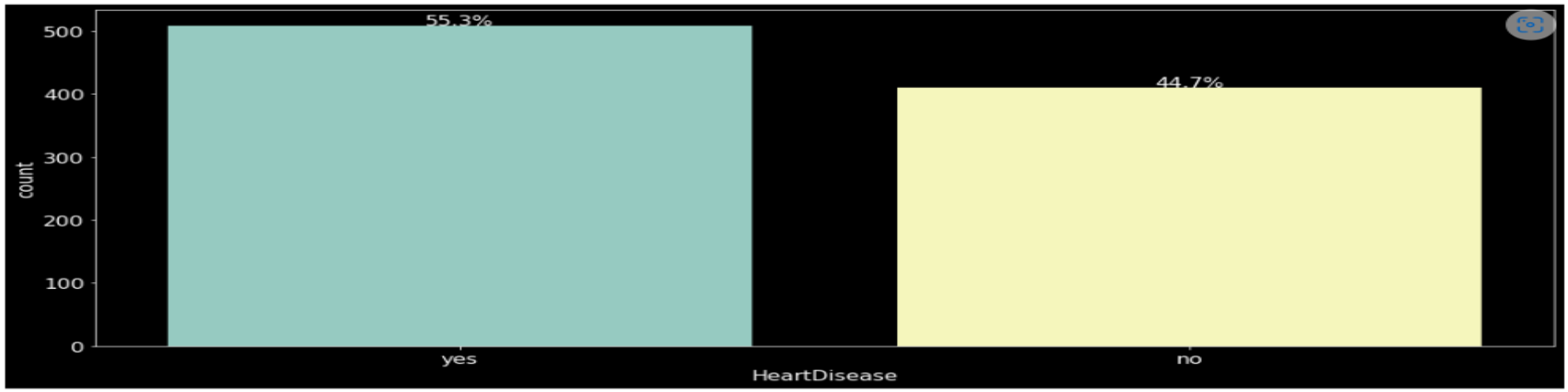



```
heart.rename(columns={"target":"have disease"} , inplace=True)
heart.replace({1:"yes" , 0:"no"} , inplace =True)
```

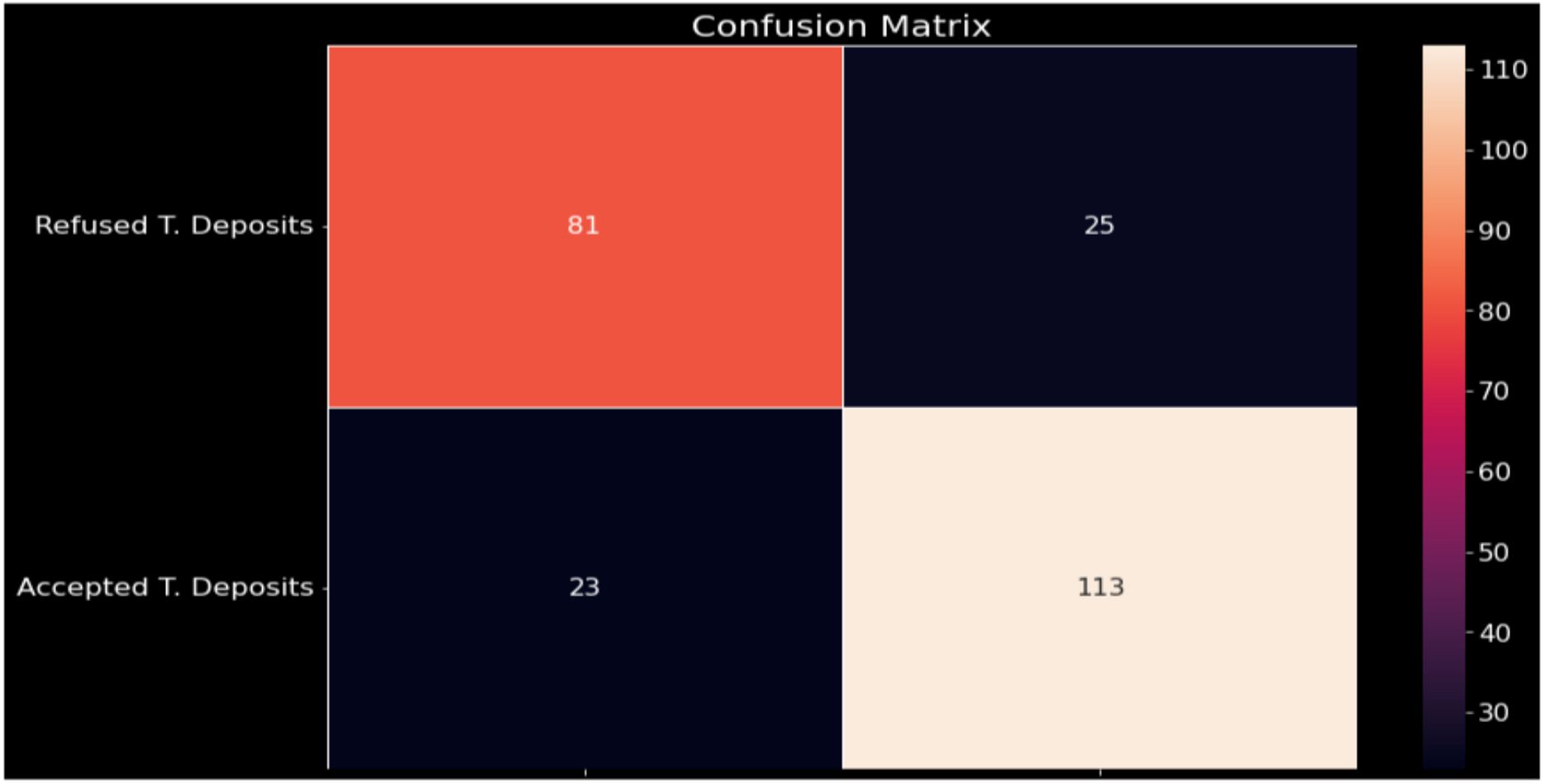
count_plot(data = heart , x ="HeartDisease")



count_plot(data = heart , x ="HeartDisease" , prop=True)



```
from sklearn.metrics import confusion_matrix
# 4697: no's, 4232: yes
conf_matrix = confusion_matrix(y_train, y_train_pred)
f, ax = plt.subplots(figsize=(12, 8))
sns.heatmap(conf_matrix, annot=True, fmt="d", linewidths=.5,
ax=ax)
plt.title("Confusion Matrix", fontsize=20)
plt.subplots_adjust(left=0.15, right=0.99, bottom=0.15,
top=0.99)
ax.set_yticks(np.arange(conf_matrix.shape[0]) + 0.5,
minor=False)
ax.set_xticklabels("")
ax.set_yticklabels(['Refused T. Deposits', 'Accepted T.
Deposits'], fontsize=16, rotation=360)
plt.show()
```




```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
lr = LogisticRegression()
```

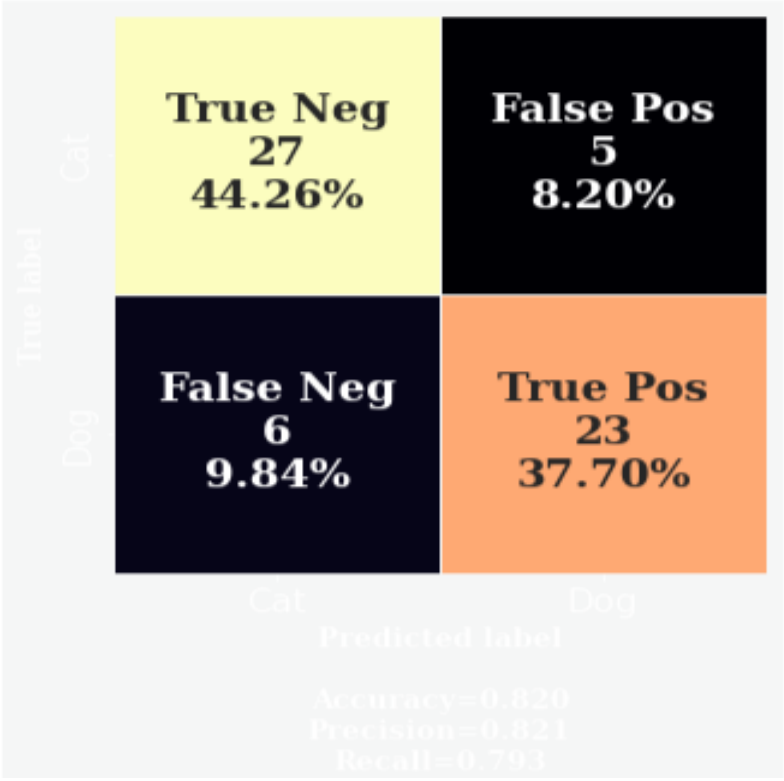
```
lr.fit(X_train,y_train)
y_pred_lr = lr.predict(X_test)
confusion_matrix(y_test,y_pred_lr)

def make_confusion_matrix(cf,
                          group_names=None,
                          categories='auto',
                          count=True,
                          percent=True,
                          cbar=True,
                          xyticks=True,
                          xyplotlabels=True,
                          sum_stats=True,
                          figsize=None,
                          cmap='Blues',
                          title=None):


    # CODE TO GENERATE TEXT INSIDE EACH SQUARE
    blanks = ['' for i in range(cf.size)]

    if group_names and len(group_names)==cf.size:
        group_labels = ["{}\n".format(value) for value in group_names]
```

```
    else:
        group_labels = blanks

    if count:
        group_counts = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        group_counts = blanks

    if percent:
        group_percentages = ["{0:.2%}".format(value) for value in cf.flatten()/np.sum(cf)]
    else:
        group_percentages = blanks

    box_labels = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in
zip(group_labels,group_counts,group_percentages)]
    box_labels = np.asarray(box_labels).reshape(cf.shape[0],cf.shape[1])


    # CODE TO GENERATE SUMMARY STATISTICS & TEXT FOR SUMMARY STATS
    if sum_stats:
        #Accuracy is sum of diagonal divided by total observations
        accuracy  = np.trace(cf) / float(np.sum(cf))
```

```
        #if it is a binary confusion matrix, show some more stats
        if len(cf)==2:
            #Metrics for Binary Confusion Matrices
            precision = cf[1,1] / sum(cf[:,1])
            recall    = cf[1,1] / sum(cf[1,:])
            f1_score  = 2*precision*recall / (precision + recall)
            stats_text = "\n\nAccuracy={:0.3f}\nPrecision={:0.3f}\nRecall={:0.3f}\nF1
Score={:0.3f}".format(
                accuracy,precision,recall,f1_score)
        else:
            stats_text = "\n\nAccuracy={:0.3f}".format(accuracy)
    else:
        stats_text = ""


    # SET FIGURE PARAMETERS ACCORDING TO OTHER ARGUMENTS
    if figsize==None:
        #Get default figure size if not set
        figsize = plt.rcParams.get('figure.figsize')

    if xyticks==False:
        #Do not show categories if xyticks is False
```

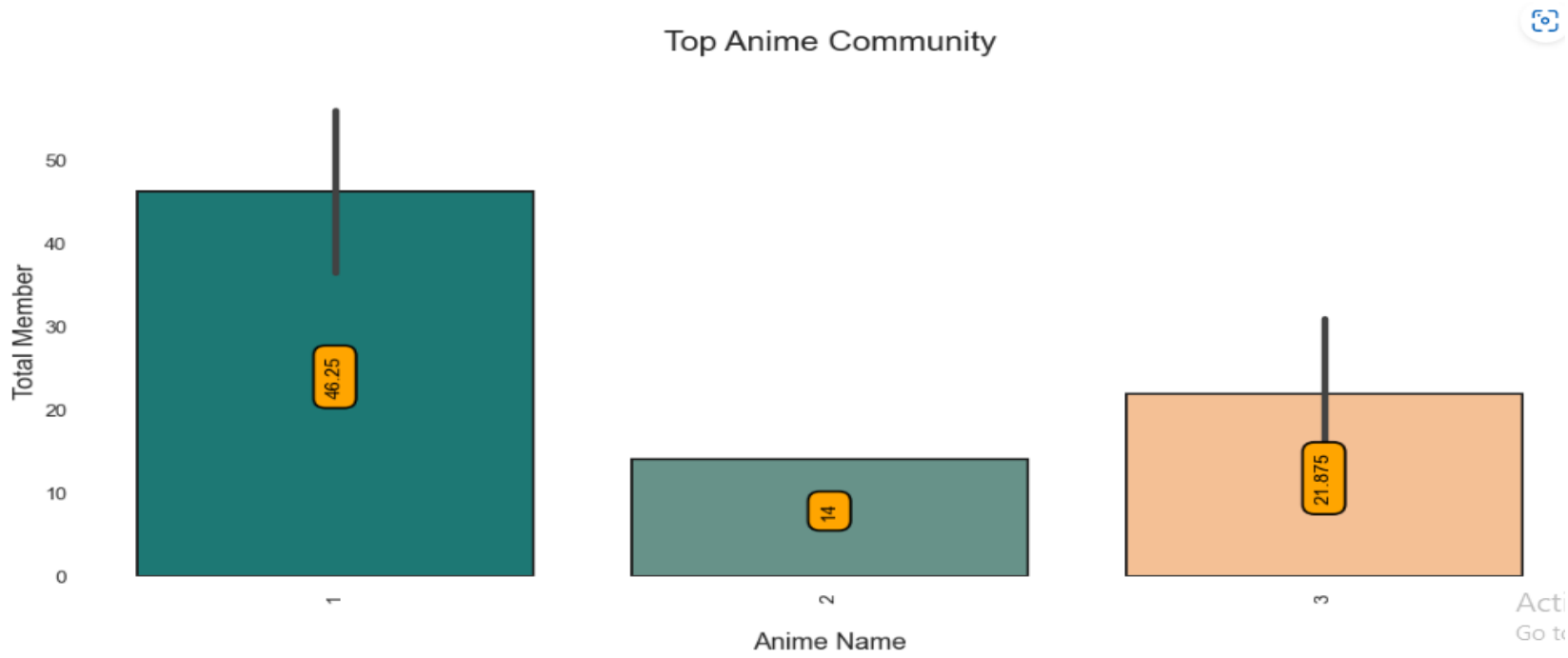
```
        categories=False

          # MAKE THE HEATMAP VISUALIZATION
    fig = plt.figure(figsize=figsize)
    fig.patch.set_facecolor('#f5f6f6')
    sns.heatmap(cf,annot=box_labels,fmt="",linewidths = 1,square = True,linecolor=
'#f5f6f6',
                cmap=cmap,cbar=cbar,annot_kws={'fontfamily':'serif','size':18,'weight':'bold'},
                xticklabels=categories,
                yticklabels=categories,)

    if xyplotlabels:
        plt.ylabel('True label', **{'fontfamily':'serif','size':12,'weight':'bold'})
        plt.xlabel('Predicted label' + stats_text,**{'fontfamily':'serif','size':12,'weight':'bold'})
    else:
        plt.xlabel(stats_text,**{'fontfamily':'serif','size':12,'weight':'bold'})



vani_cf_matrix = confusion_matrix(y_test,y_pred_lr)
my_cols = [colors[3],colors[2]]

labels = [ 'True Neg','False Pos','False Neg','True Pos']
```

```
categories = ['Cat', 'Dog']
make_confusion_matrix(vani_cf_matrix,figsize = (10,5),group_names=labels,cbar =
False,cmap = 'magma',categories=categories,
                      title = 'Vanila CNN comfusion matrix')


plt.show()
```



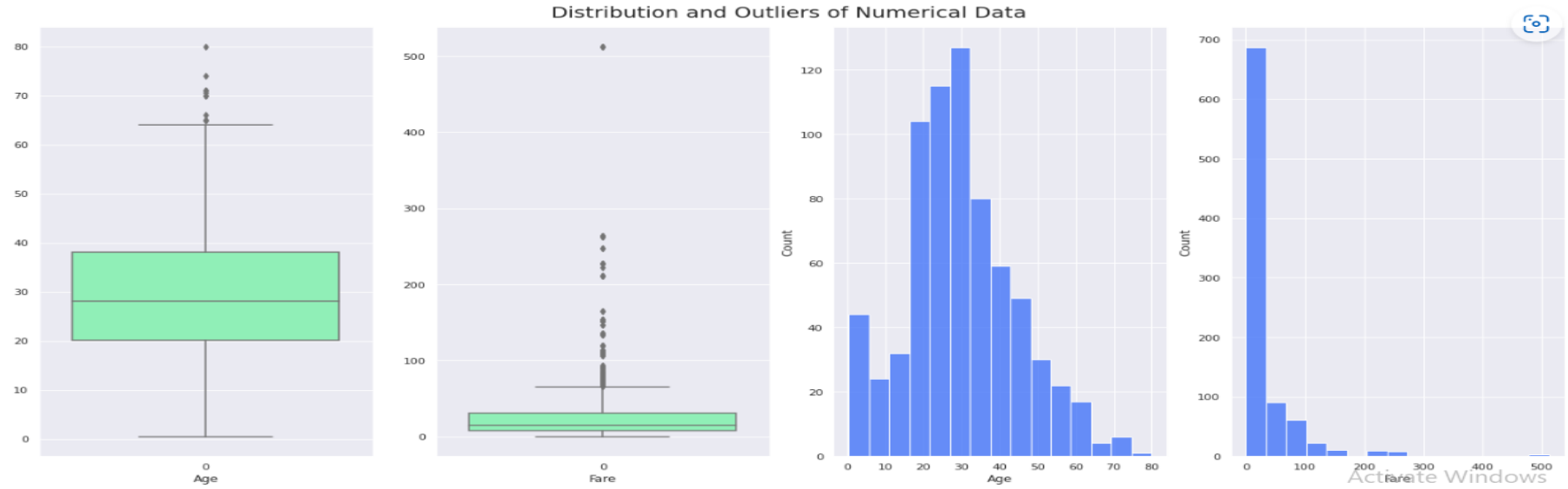
```
sns.set_style("white")
sns.set_context("poster",font_scale = .7)
palette =
["#1d7874","#679289","#f4c095","#ee2e31","#ffb563","#918450","#f85e00","#a
41623","#9a031e","#d6d6d6","#ffee32","#ffd100","#333533","#202020"]
# sns.palplot(sns.color_palette(palette))
```

```
# plt.show()

plt.subplots(figsize=(20,8))
p = sns.barplot(x=dataset["Pclass"][:14],y=dataset["Age"],palette=palette,
saturation=1, edgecolor = "#1c1c1c", linewidth = 2)
p.axes.set_title("\nTop Anime Community\n", fontsize=25)
plt.ylabel("Total Member" , fontsize = 20)
plt.xlabel("\nAnime Name" , fontsize = 20)
# plt.yscale("log")
plt.xticks(rotation = 90)
for container in p.containers:
    p.bar_label(container,label_type = "center",padding = 6,size = 15,color =
"black",rotation = 90,
    bbox={"boxstyle": "round", "pad": 0.6, "facecolor": "orange", "edgecolor":
"black", "alpha": 1})

sns.despine(left=True, bottom=True)
plt.show()
```

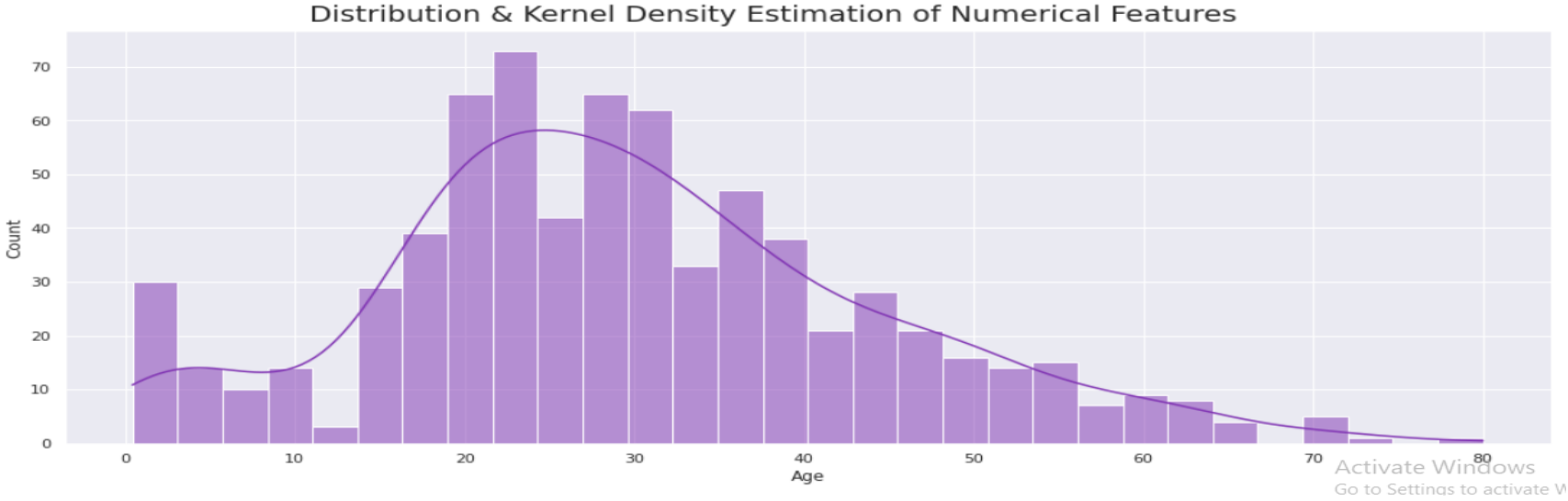



```
numfeature = ["Age", "Fare"]
enumfeat = list(enumerate(numfeature))

plt.figure(figsize=(20,9))
plt.suptitle("Distribution and Outliers of Numerical Data", fontsize=20)
for i in enumfeat:
```

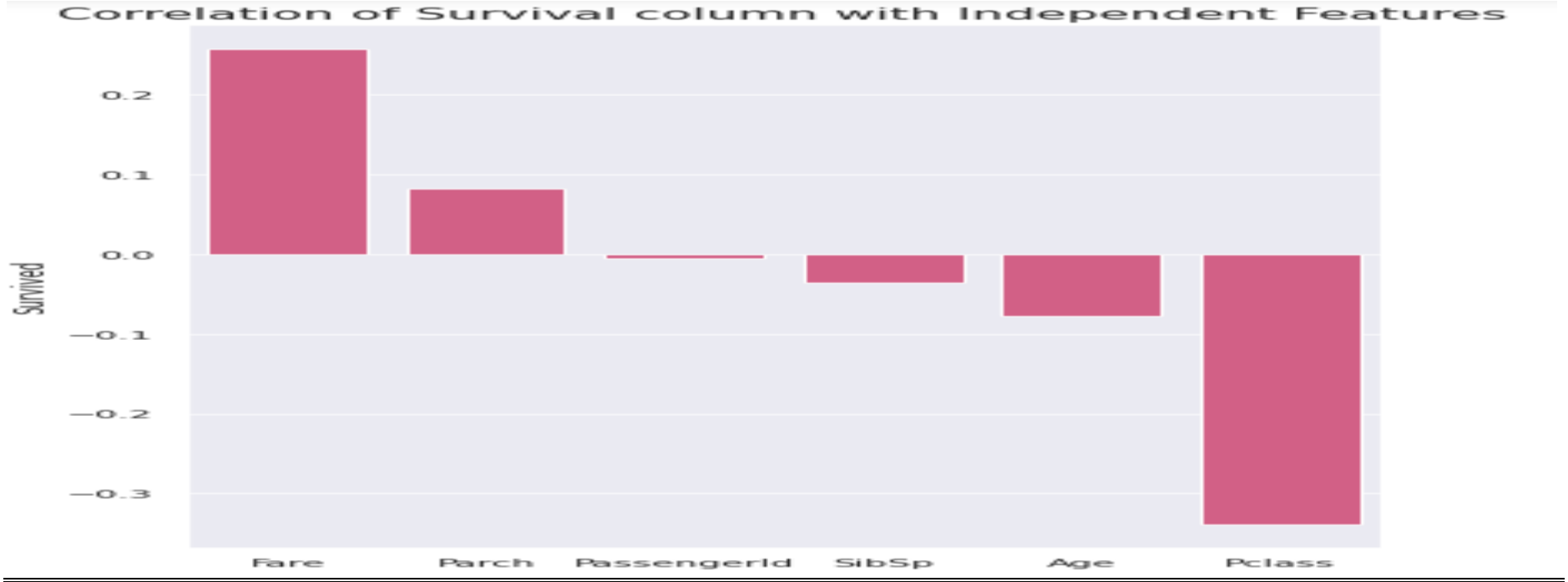
```
    plt.subplot(1,4,i[0]+1)
    sns.boxplot(data = train[i[1]], palette="rainbow")
    plt.xlabel(str(i[1]))
for i in enumfeat:
    plt.subplot(1,4,i[0]+3)
    sns.histplot(data = train[i[1]], palette="rainbow", bins=15)
    plt.xlabel(str(i[1]))
plt.tight_layout()
plt.show()
```



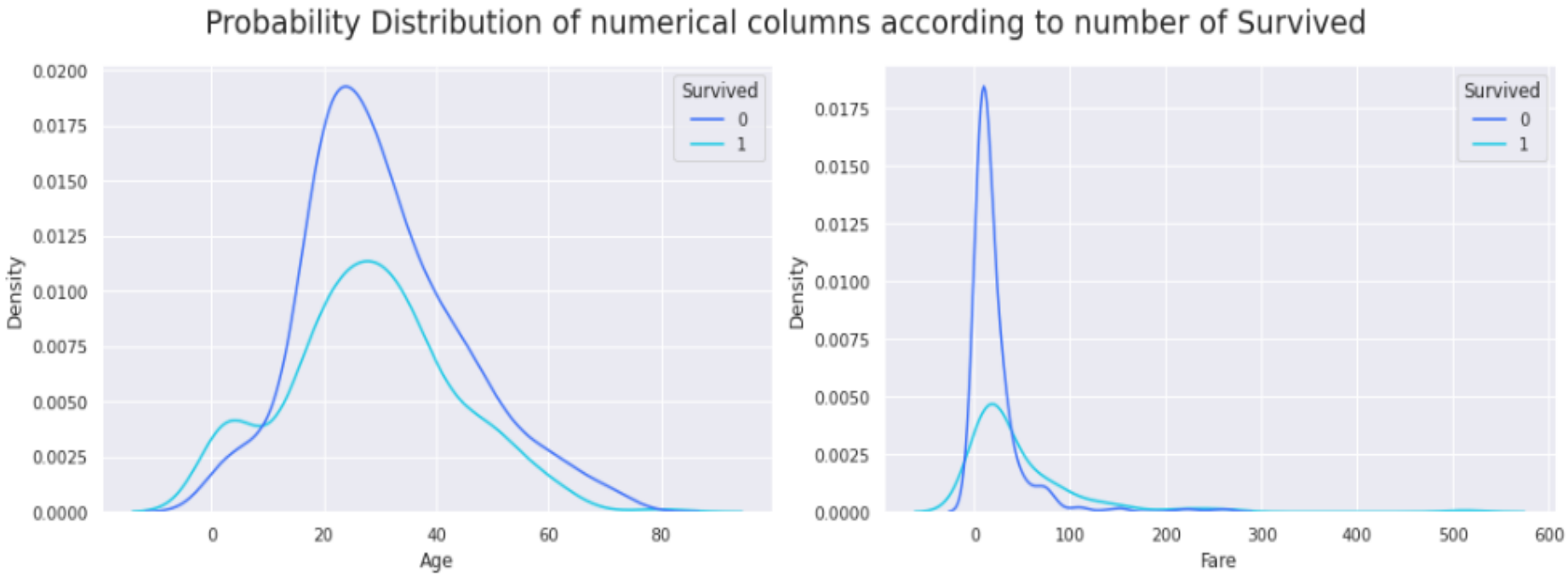
```
plt.figure(figsize=(15,12))
plt.suptitle("Distribution & Kernel Density Estimation of Numerical
Features", fontsize=20)
for i in enumfeat:
    plt.subplot(2,1,i[0]+1)
    sns.histplot(x = train[i[1]], kde=True, bins=30,
color=(0.50,0.20,0.70))
plt.tight_layout()
plt.show()
```



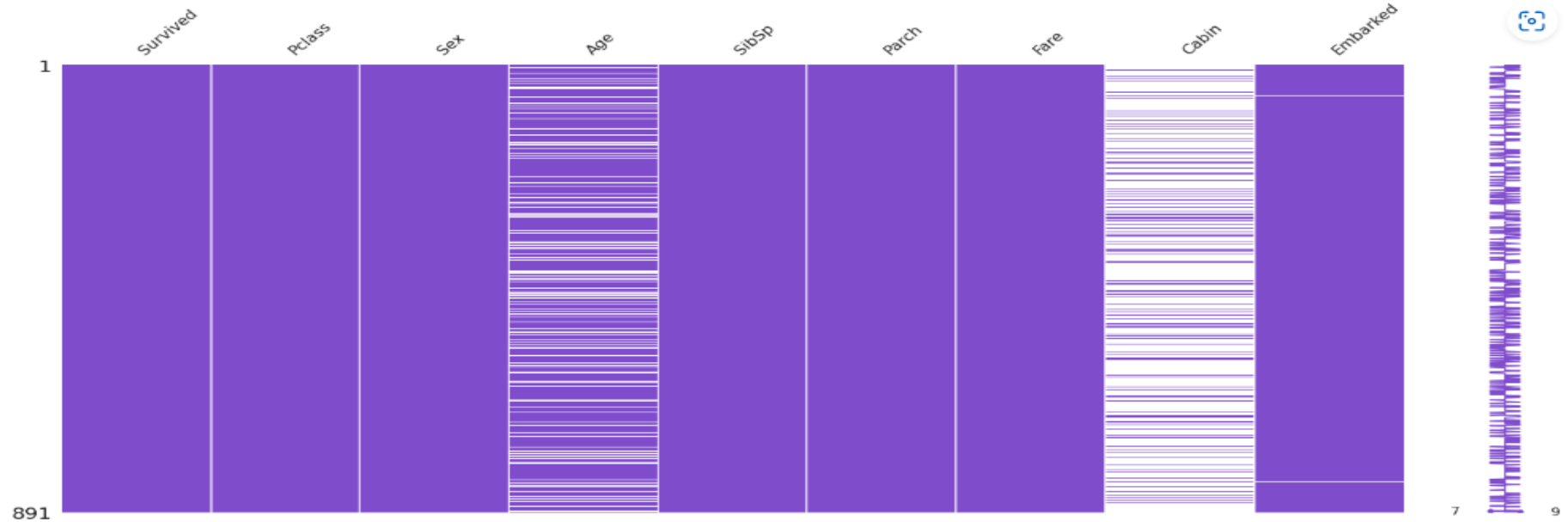
```
plt.figure(figsize=(6,8))
plt.title("Correlation of Survival column with Independent Features",
fontsize=15)
corr = train.corr()["Survived"].sort_values(ascending=False)[1:]
sns.barplot(x=corr.index, y=corr, color=(0.90,0.30,0.50))
plt.tight_layout()
plt.show()
```



```
plt.figure(figsize=(15,5))
plt.suptitle("Probability Distribution of numerical columns according to number of Survived", fontsize = 20)
for i in enumfeat:
    plt.subplot(1,2,i[0]+1)
    sns.kdeplot(data=train, x=i[1], hue="Survived")
plt.tight_layout()
plt.show()
```



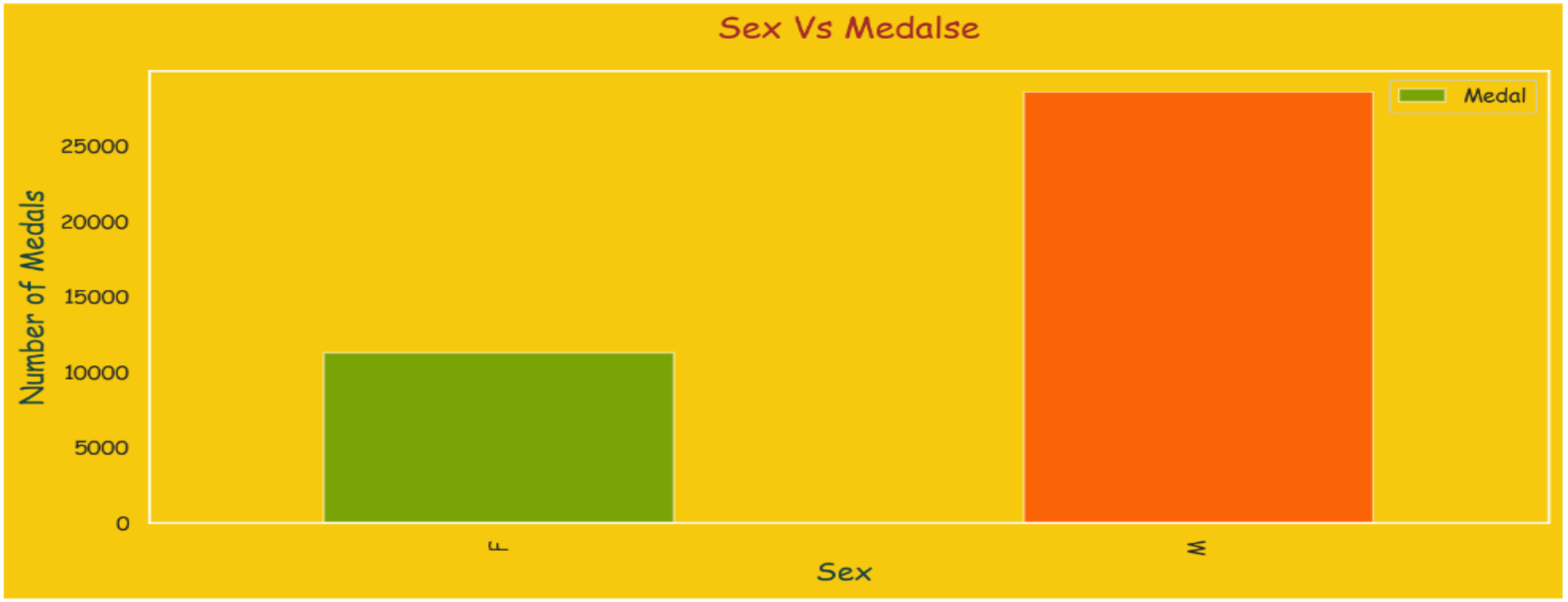
```
import missingno as msno
msno.matrix(train, color=(0.50,0.30,0.80))
plt.show()
x = train.isnull().sum()
for a, b in x.items():
    if b > 0:
        print(f"There are {b} missing values in column: {a}")
```




```
There are 177 missing values in column: Age
There are 687 missing values in column: Cabin
There are 2 missing values in column: Embarked
```

```
rc = {'figure.dpi': 150, 'axes.labelsize': 4,
                'axes.facecolor': '#F6C90E', 'grid.color': 'Red','figure.figsize':(12,5),
                'figure.facecolor': '#F6C90E'}
sns.set_theme(context='notebook',
              style='dark',
              palette='deep',
              font='Comic Sans Ms',
              font_scale=1,
              color_codes='red',
              rc=rc)

color = ['Green',"Red"]
df.groupby('Sex')['Medal'].count().sort_values(ascending=True).plot(kind="bar",
color=color,alpha=.5);
plt.title("Sex Vs Medalse",fontsize=17,color='Brown',font='Comic Sans
Ms',pad=20);
plt.xlabel("Sex ",fontsize=15,color='#1a4441',font='Comic Sans Ms')
plt.ylabel("Number of Medals",fontsize=15,color='#1a4441',font='Comic Sans
Ms');
```

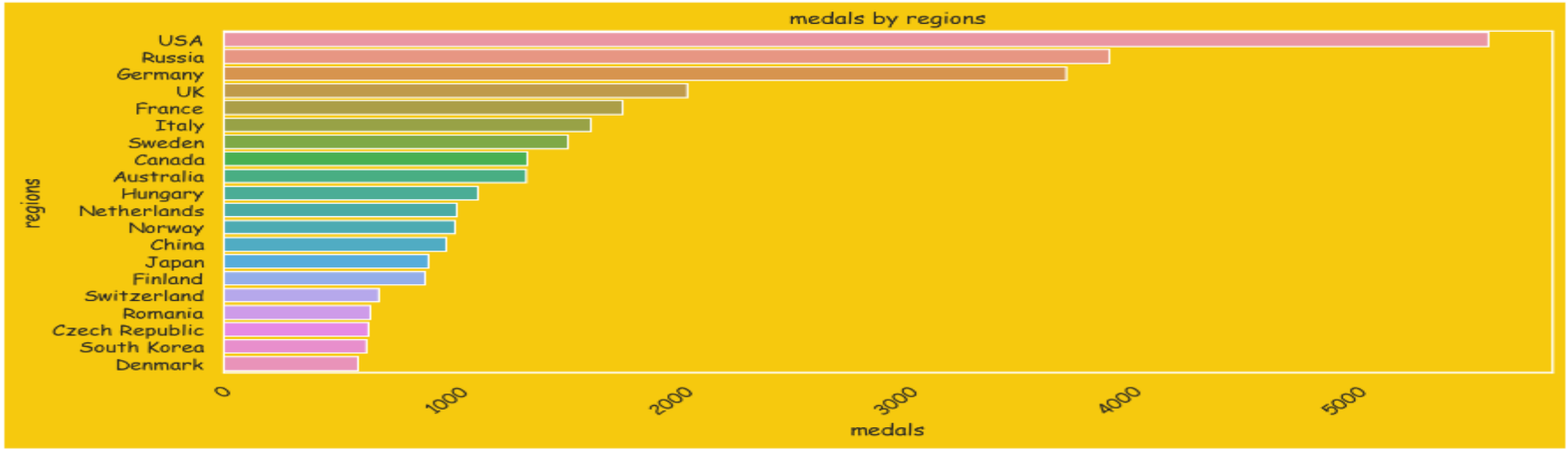
```
plt.legend(loc='best');
plt.savefig('world regions.png');
```



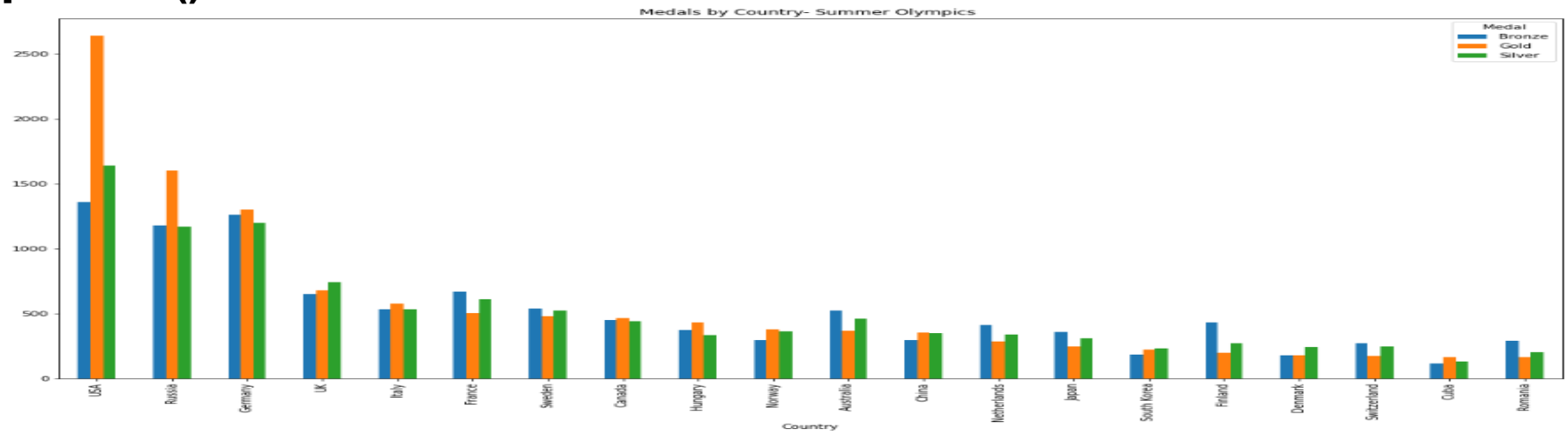

```
region_medal=df.groupby('region')['Medal'].count().nlargest(20).reset_index()
region_medal.head()
```

	region	Medal
0	USA	5637
1	Russia	3947
2	Germany	3756
3	UK	2068
4	France	1777

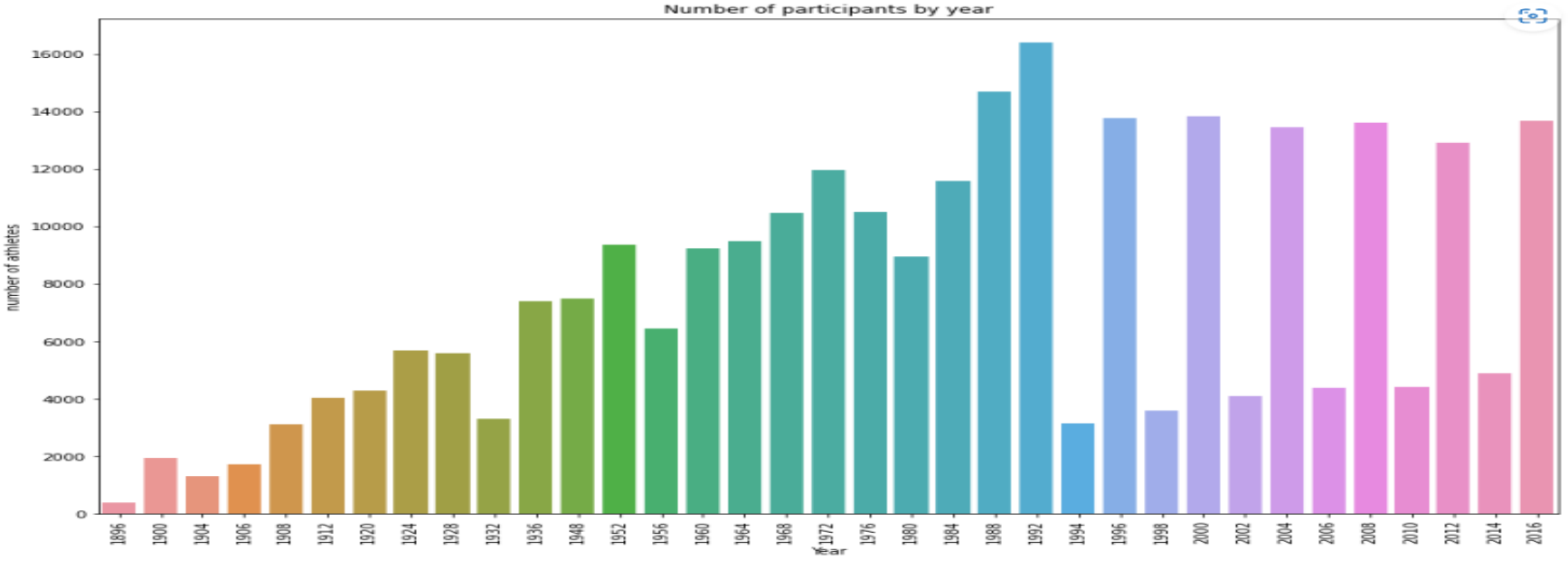
```
sns.barplot(y='region',x='Medal',data=region_medal)
plt.title('medals by regions')
plt.xlabel('medals')
plt.ylabel('regions')
plt.xticks(rotation=45)
plt.show()
```



```
summer_medals=df.groupby(['region', 'Medal']).size().reset_index()
summer_medals.columns=['region', 'Medal', 'count']
summer_medals.pivot('region', 'Medal', 'count').fillna(0)
summer_medals_20=summer_medals.pivot('region', 'Medal',
'count').fillna(0).sort_values(['Gold'], ascending=False).head(20)
summer_medals_20.plot(kind='bar')
plt.xlabel('Country')
plt.title('Medals by Country- Summer Olympics ')
fig = plt.gcf()
fig.set_size_inches(18.5, 10.5)
plt.show()
```



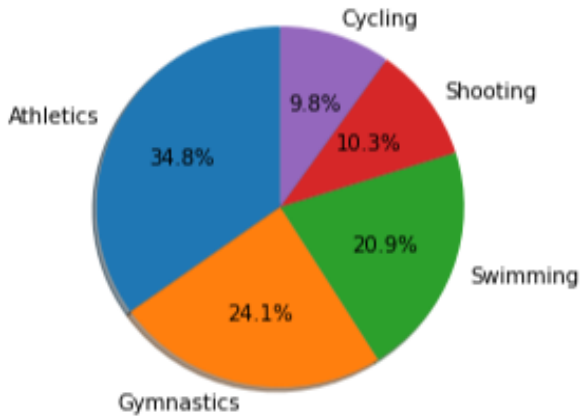
```
year=df['Year'].value_counts()

plt.figure(figsize=(15,10))
sns.barplot(x=year.index, y=year.values)
plt.xticks(rotation=90)
plt.xlabel("Year")
plt.ylabel("number of athletes")
plt.title("Number of participants by year")
plt.show()
```



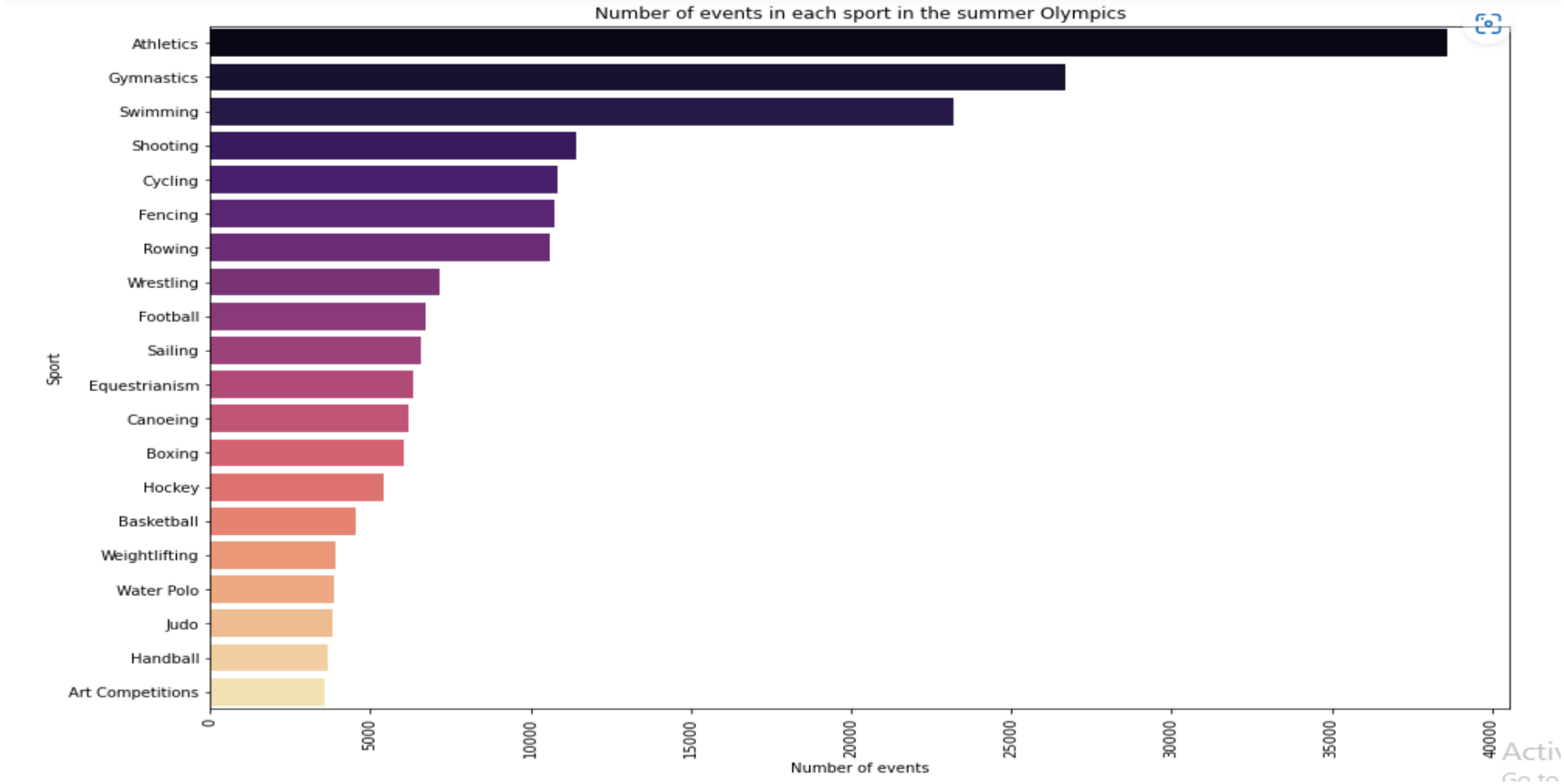
```
sport=df['Sport'].value_counts()[:5]
print(sport)
```

```
Athletics     38624
Gymnastics    26707
Swimming      23195
Shooting      11448
Cycling       10859
Name: Sport, dtype: int64
```

```
labels=sport.index
sizes=sport.values
plt.pie(sizes,labels=labels,autopct='%1.1f%%',
        shadow=True,startangle=90)
plt.show()
```



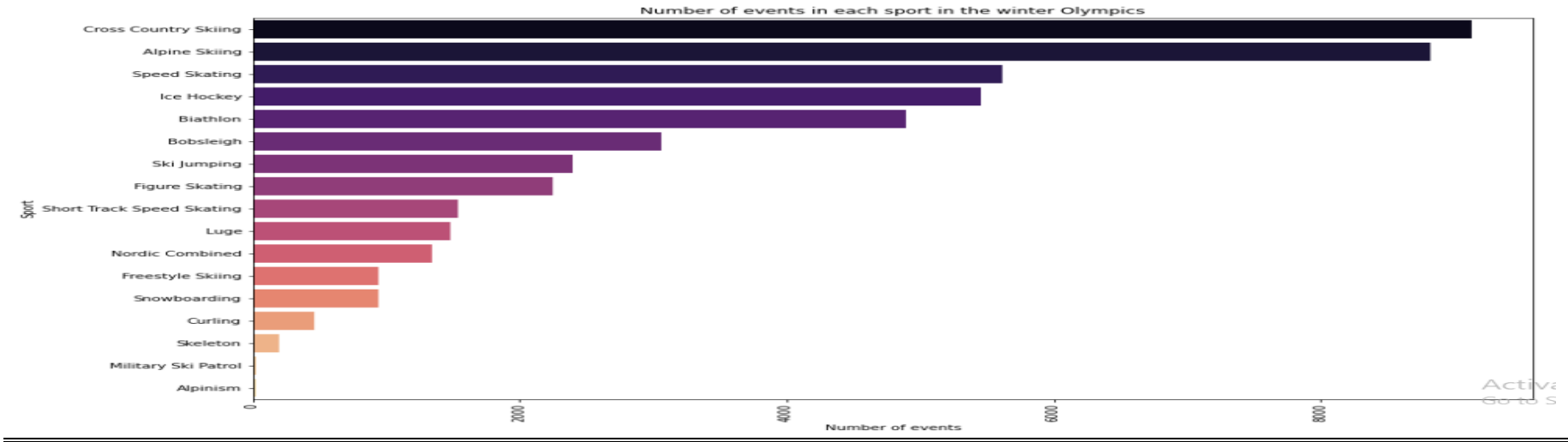
```
sport_summer=df[df['Season']=='Summer']['Sport'].value_counts
().sort_values(ascending=False).head(20)
sport_summer

plt.figure(figsize=(15,10))
sns.barplot(y=sport_summer.index, x=sport_summer.values,
palette='magma')
plt.xlabel('Number of events')
plt.ylabel('Sport')
plt.xticks(rotation=90)
plt.title("Number of events in each sport in the summer
Olympics")
plt.show()
```




```
sport_winter=df[df['Season']=='Winter']['Sport'].value_counts().sort_values(ascending=False)
```

```
Cross Country Skiing    9133
Alpine Skiing           8829
Speed Skating           5613
Ice Hockey              5456
Biathlon                4893
```

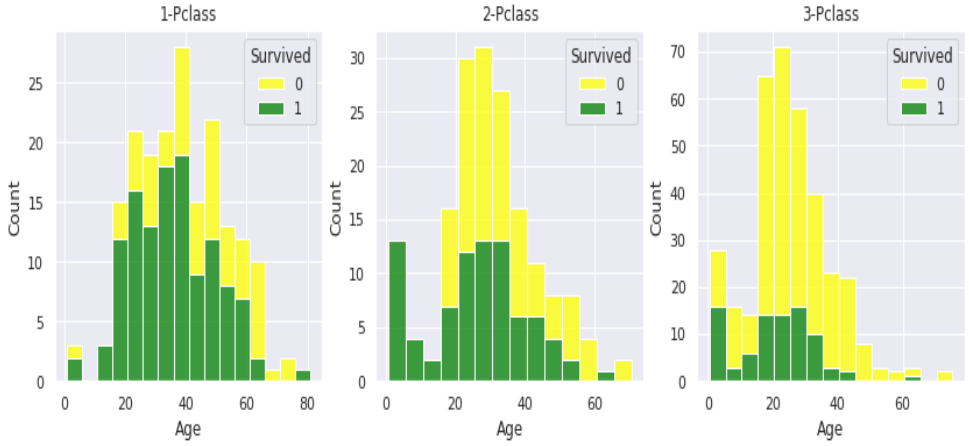
```
plt.figure(figsize=(15,10))
sns.barplot(y=sport_winter.head(20).index, x=sport_winter.head(20).values,
palette='magma')
plt.xlabel('Number of events')
plt.ylabel('Sport')
plt.xticks(rotation=90)
plt.title("Number of events in each sport in the winter Olympics")
plt.show()
```



Mastering Data Visualization Techniques (Part 3)

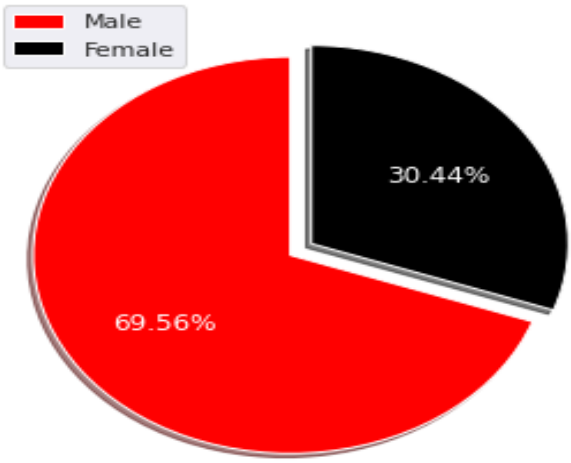
Prepared by: Syed Afroz Ali

```
plot , ax = plt.subplots(1 , 3 , figsize=(14,4))
sns.histplot(data = train_data.loc[train_data["Pclass"]==1] , x = "Age" , hue = "Survived",binwidth=5,ax = ax[0],palette = sns.color_palette(["yellow" , "green"]),multiple = "stack").set_title("1-Pclass")
sns.histplot(data = train_data.loc[train_data["Pclass"]==2] , x = "Age" , hue = "Survived",binwidth=5,ax = ax[1],palette = sns.color_palette(["yellow" , "green"]),multiple = "stack").set_title("2-Pclass")
sns.histplot(data = train_data.loc[train_data["Pclass"]==3] , x = "Age" , hue = "Survived",binwidth=5,ax = ax[2],palette = sns.color_palette(["yellow" , "green"]),multiple = "stack").set_title("3-Pclass")
plt.show()
```



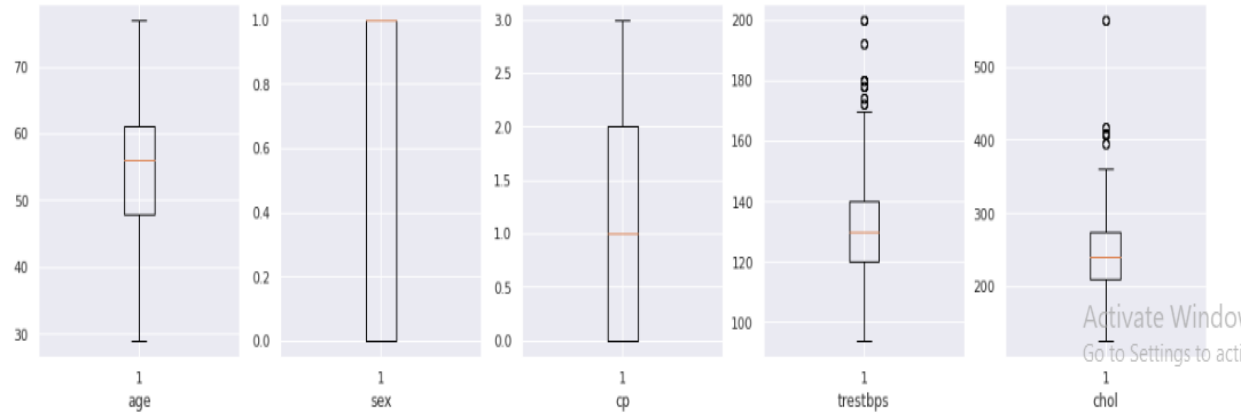
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
sex = ["Male", "Female"]
values = data["sex"].value_counts()
color = ["#FF0000", "#000000"]

plt.figure(figsize = (5, 7))
plt.pie(values, labels = sex, colors = color, explode = (0.1, 0),
textprops = {"color":"w"}, autopct = "%.2f%%", shadow = Tru
e, startangle = 90)

plt.legend();
```

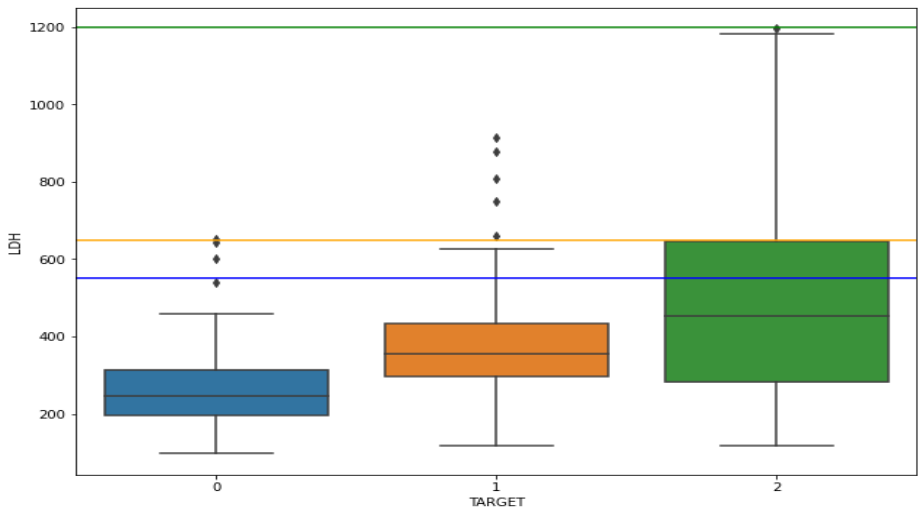



```
# Plotting Outliers
col = 1
plt.figure(figsize = (20, 15))
for i in data.columns:
    if col < 14:
        plt.subplot(3, 5, col)
        plt.boxplot(data[i])
        plt.xlabel(i)
    col = col + 1
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

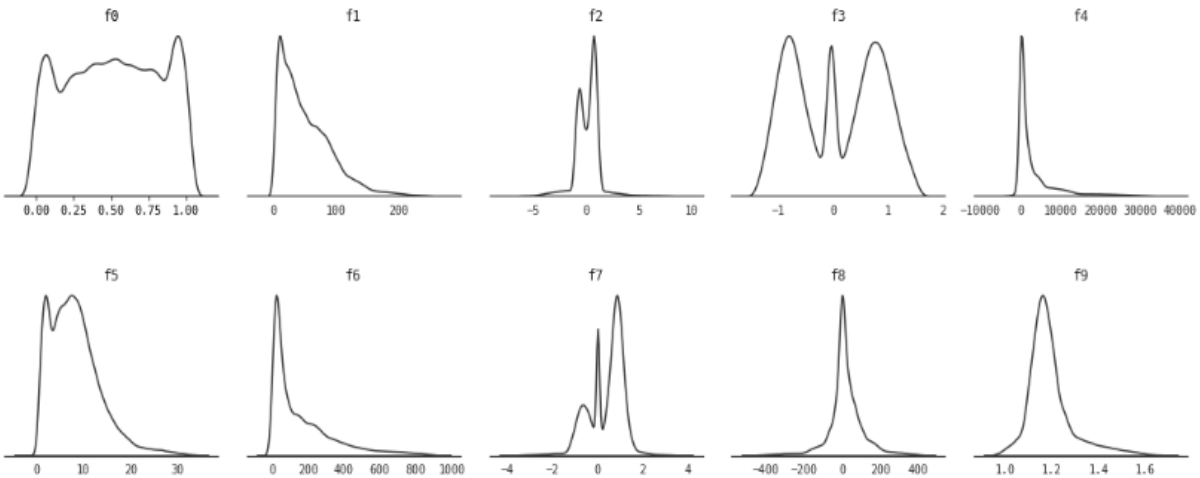



```
fig = plt.figure( figsize=(8, 6))
ax = fig.add_axes([0,0,1,1])
sns.boxplot(ax=ax, data=df, x='TARGET', y='LDH')#,fli
erprops=dict(marker='o', markersize=6),fliersize=2)

ax.axhline(y=550,color='b')
ax.axhline(y=650,color='orange')
ax.axhline(y=1200,color='g')
```



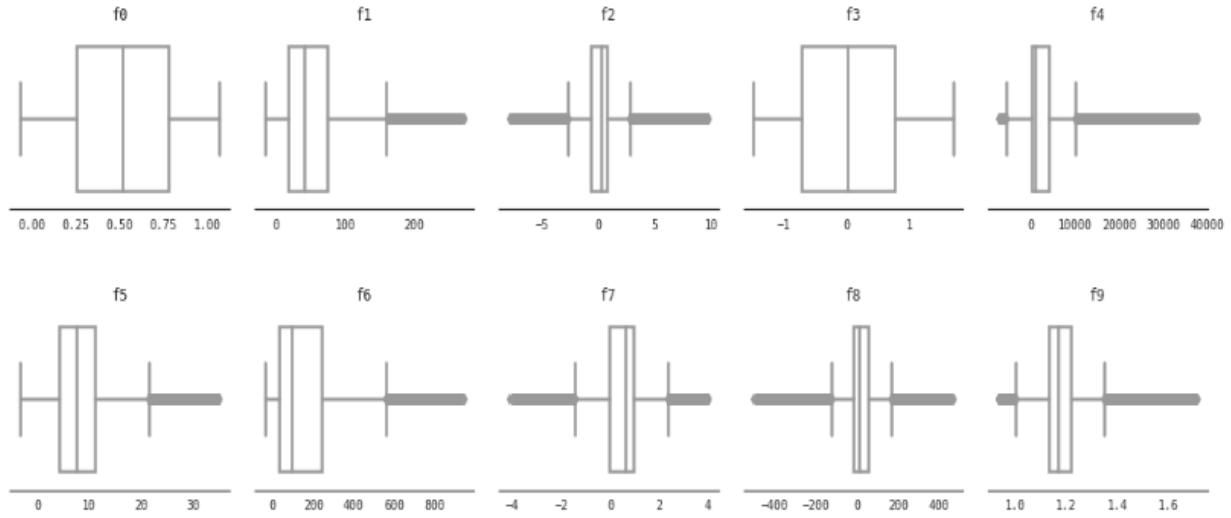
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig = plt.figure(figsize = (15, 60))
for i in range(len(train.columns.tolist()[:100])):
    plt.subplot(20,5,i+1)
    sns.set_style("white")
    plt.title(train.columns.tolist()[:100][i], size = 12, fontname
= 'monospace')
    a = sns.kdeplot(train[train.columns.tolist()[:100][i]], shade
= True, alpha = 0.9, linewidth = 1.5, facecolor=(1, 1, 1, 0), ed
gecolor=".2")
    plt.ylabel('')
    plt.xlabel('')
    plt.xticks(fontname = 'monospace')
    plt.yticks([])
    for j in ['right', 'left', 'top']:
        a.spines[j].set_visible(False)
        a.spines['bottom'].set_linewidth(1.2)

fig.tight_layout(h_pad = 3)
plt.show()
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

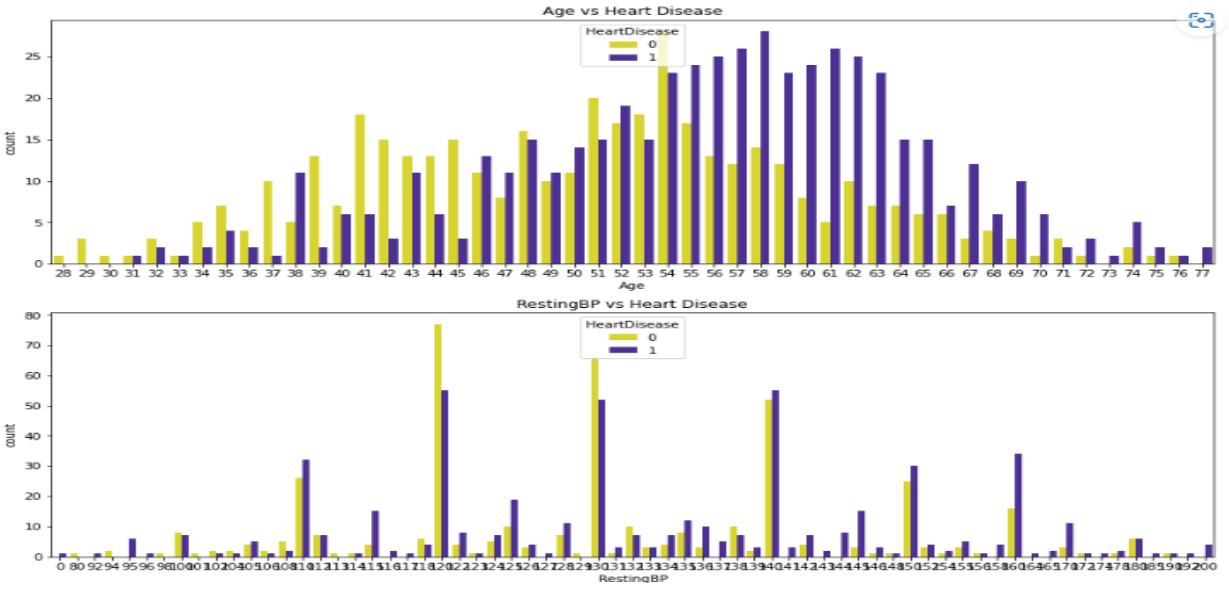
```
fig = plt.figure(figsize = (15, 60))
for i in range(len(train.columns.tolist()[:100])):
    plt.subplot(20,5,i+1)
    sns.set_style("white")
    plt.title(train.columns.tolist()[:100][i], size = 12, fontname
= 'monospace')
    a = sns.boxplot(train[train.columns.tolist()[:100][i]], linewi
dth = 2.5,color = 'white')
    plt.ylabel('')
    plt.xlabel('')
    plt.xticks(fontname = 'monospace')
    plt.yticks([])
    for j in ['right', 'left', 'top']:
        a.spines[j].set_visible(False)
        a.spines['bottom'].set_linewidth(1.2)

fig.tight_layout(h_pad = 3)
plt.show()
```




```
fig, ax = plt.subplots(nrows = 5,ncols = 1,figsize = (15,30))
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
colors = ['#F3ED13','#451FA4']
for i in range(len(numerical_features)):
    plt.subplot(5,1,i+1)
    sns.countplot(numerical_features[i],data = data,hue = "HeartDisease",palette = colors)
    title = numerical_features[i] + ' vs Heart Disease'
    plt.title(title);
```



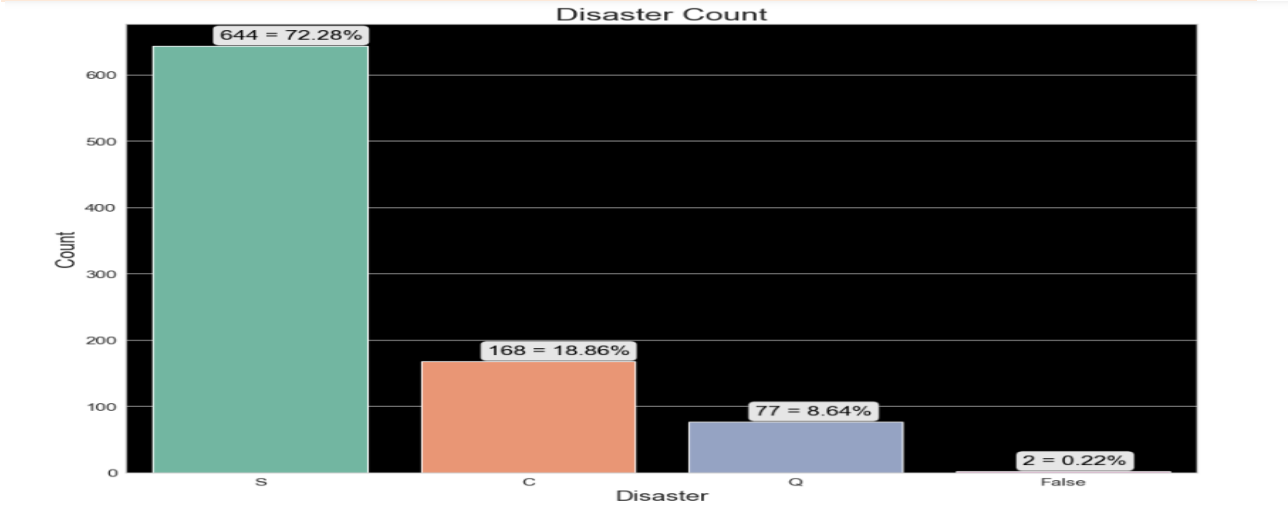
.

```
train.iloc[:, :-1].describe().T.sort_values(by='std' , ascending = False)\
                .style.background_gradient(cmap='GnBu')\
                .bar(subset=["max"], color='#BB0000')\
                .bar(subset=["mean",], color='green')
```

	count	mean	std	min	25%	50%	75%	max
FoodCourt	8510.00	458.08	1611.49	0.00	0.00	0.00	76.00	29813.00
VRDeck	8505.00	304.85	1145.72	0.00	0.00	0.00	46.00	24133.00
Spa	8510.00	311.14	1136.71	0.00	0.00	0.00	59.00	22408.00
RoomService	8512.00	224.69	666.72	0.00	0.00	0.00	47.00	14327.00
ShoppingMall	8485.00	173.73	604.70	0.00	0.00	0.00	27.00	23492.00
Age	8514.00	28.83	14.49	0.00	19.00	27.00	38.00	79.00

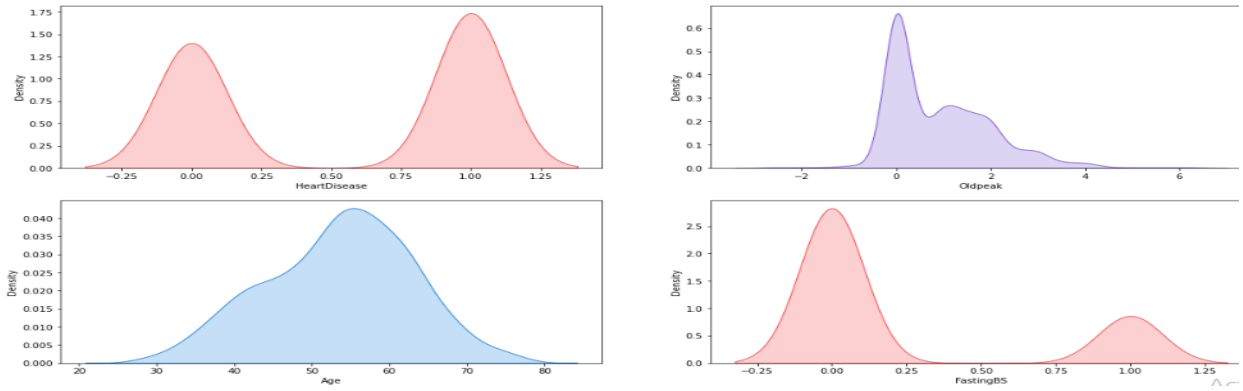
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
plt.figure(figsize = (15, 12))
ax = plt.axes()
ax.set_facecolor('black')
ax = sns.countplot(x = 'Embarked', data = titanic, palette = [
custom_colors[2], custom_colors[1]], edgecolor = 'white', lin
ewidth = 1.2)
plt.title('Disaster Count', fontsize = 25)
plt.xlabel('Disaster', fontsize = 20)
plt.ylabel('Count', fontsize = 20)
ax.xaxis.set_tick_params(labelsize = 15)
ax.yaxis.set_tick_params(labelsize = 15)
bbox_args = dict(boxstyle = 'round', fc = '0.9')
for p in ax.patches:
        ax.annotate('{:.0f} = {:.2f}%'.format(p.get_height(), (p.get
_height() / len(titanic['Embarked'])) * 100), (p.get_x() + 0.25, p
.get_height() + 10),
                    color = 'black',
                    bbox = bbox_args,
                    fontsize = 18)
plt.show()
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

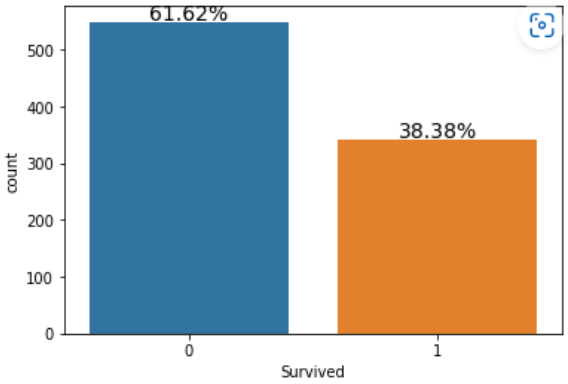
```
#Plotting the distributions of the numerical variables
color_plot = ['#de972c','#74c91e','#1681de','#e069f5','#f54545','#f0ea46','#7950cc']

fig,ax = plt.subplots(4,2,figsize=(20,20))
sns.kdeplot(df['HeartDisease'],color=np.random.choice(color_plot), ax=ax[0][0], shade=True)
sns.kdeplot(df['Oldpeak'],color=np.random.choice(color_plot), ax=ax[0][1], shade=True)
sns.kdeplot(df['Age'],color=np.random.choice(color_plot), ax=ax[1][0], shade=True)
sns.kdeplot(df['FastingBS'],color=np.random.choice(color_plot), ax=ax[1][1], shade=True)
sns.kdeplot(df['RestingBP'],color=np.random.choice(color_plot), ax=ax[2][0],shade=True)
sns.kdeplot(df['Cholesterol'],color=np.random.choice(color_plot), ax=ax[2][1], shade=True)
sns.kdeplot(df['MaxHR'],color=np.random.choice(color_plot), ax=ax[3][0],shade=True)
fig.delaxes(ax[3][1])
```



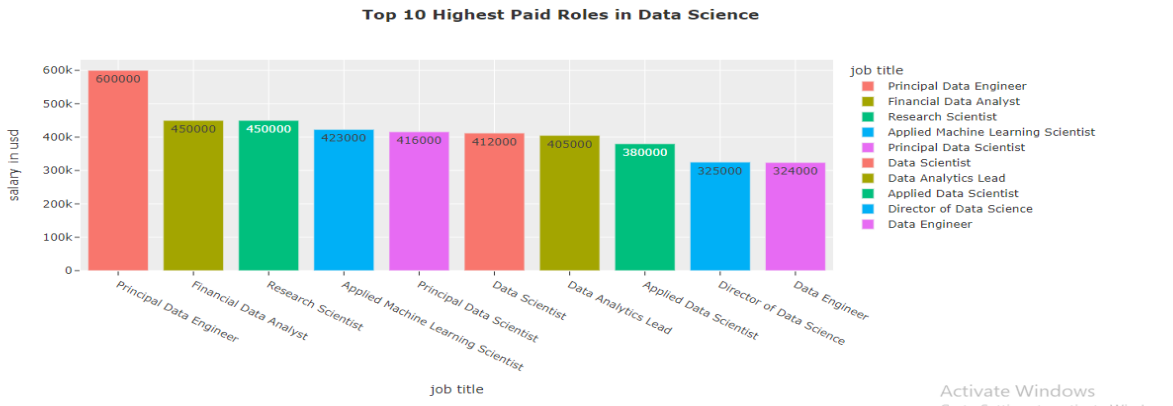

```
s = sns.countplot(x = 'Survived',data = train)
sizes=[]
for p in s.patches:
    height = p.get_height()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
    sizes.append(height)
    s.text(p.get_x()+p.get_width()/2.,
           height + 3,
           '{:1.2f}%'.format(height/len(train)*100),
           ha="center", fontsize=14)
```



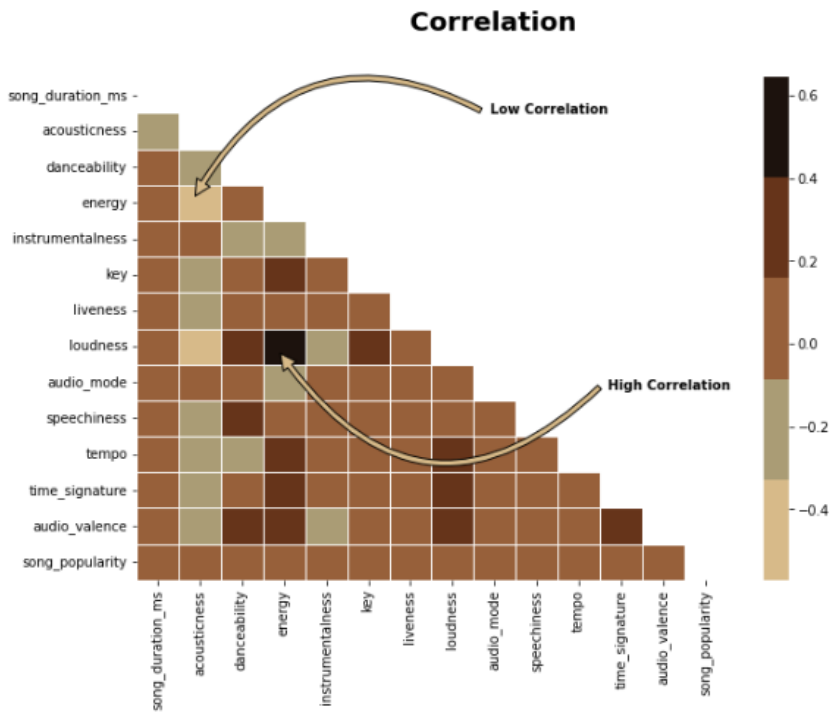

```
z=df['job_title'].value_counts().head(10)
fig=px.bar(z,x=z.index,y=z.values,color=z.index,text=z.values,labels={'index':'job title','y':'count','text':'count'},template='seaborn',title='<b> Top 10 Popular Roles in Data Sceince')
fig.show()
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
hm= df.drop('id', axis =1)
mask = np.zeros_like(hm.corr(), dtype=np.bool)
mask[np.triu_indices_from(mask)]= True

plt.suptitle('Correlation', size = 20, weight='bold')

ax = sns.heatmap(hm.corr(), linewidths = 0.9, linecolor = 'whi
te', cbar = True,mask=mask, cmap=heatmap)

ax.annotate('Low Correlation',
                fontsize=10,fontweight='bold',
                xy=(1.3, 3.5), xycoords='data',
                xytext=(0.6, 0.95), textcoords='axes fraction',
                arrowprops=dict(
                facecolor=heatmap[0], shrink=0.025,
                connectionstyle='arc3, rad=0.50'),
                horizontalalignment='left', verticalalignment='to
p'
)
ax.annotate('High Correlation',
                fontsize=10,fontweight='bold',
                xy=(3.3, 7.5), xycoords='data',
                xytext=(0.8, 0.4), textcoords='axes fraction',
                arrowprops=dict(
                facecolor=heatmap[0], shrink=0.025,
                connectionstyle='arc3, rad=-0.6'),
                horizontalalignment='left', verticalalignment='to
p'
)
plt.show()
```

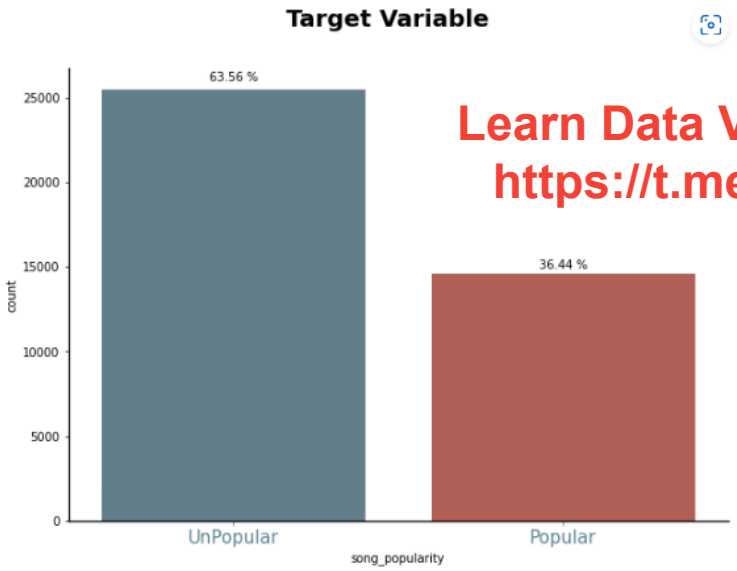
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/




```
plt.suptitle('Target Variable', size = 20, weight='bold')

song_popularity = df['song_popularity'].map({0:'UnPopular', 1:
'Popular'})

a = sns.countplot(data = df, x =song_popularity,palette=them
e)
plt.tick_params(axis="x", colors=theme[0],labelsize=15)

for p in a.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

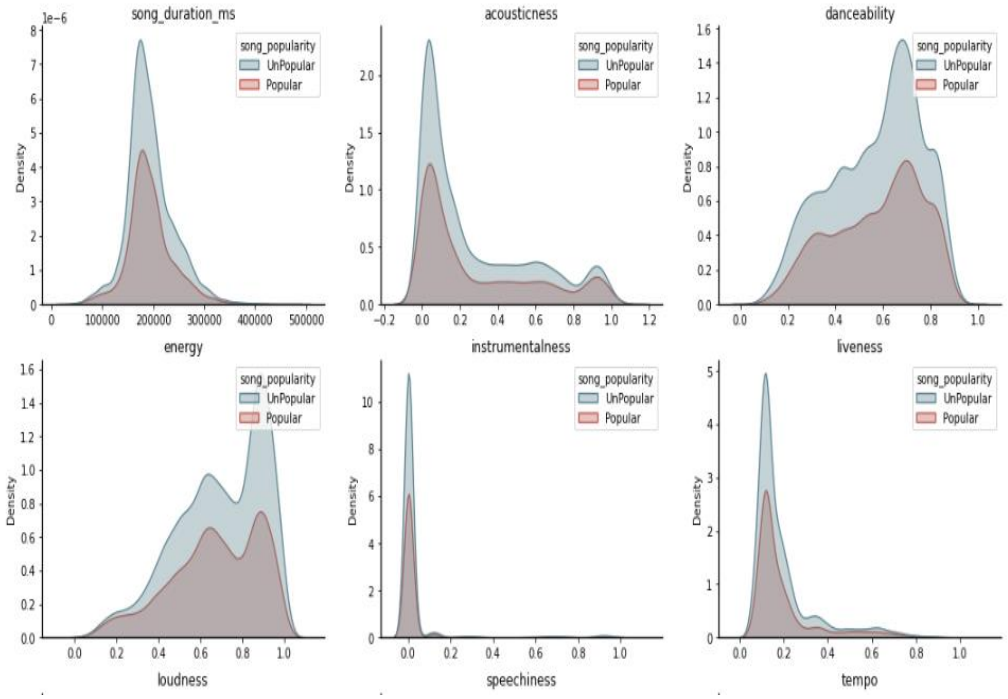
```
    a.annotate(f'{height/df.shape[0]*100} %', (x + width/2, y + h
eight*1.02), ha='center')

plt.show()
```




```
cont = ['song_duration_ms', 'acousticness', 'danceabi
lity', 'energy',
       'instrumentalness', 'liveness', 'loudness',
       'speechiness', 'tempo', 'audio_valence']
cat = [ 'key', 'audio_mode', 'time_signature']

a = 4  # number of rows
b = 3  # number of columns
c = 1  # initialize plot counter

plt.figure(figsize= (18,18))
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

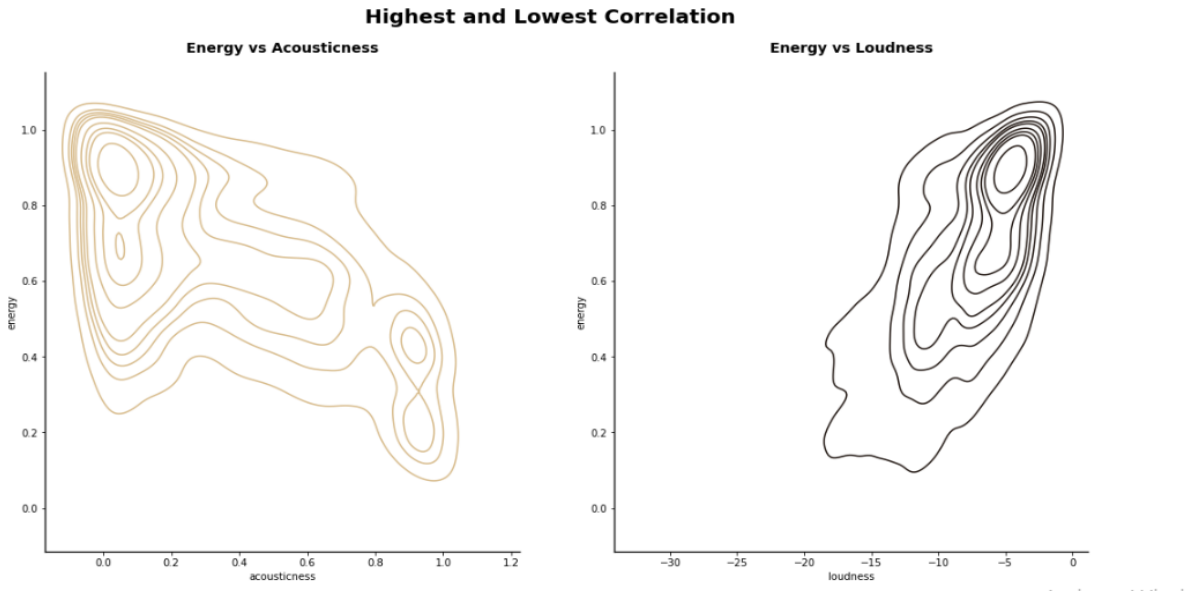
```
for i in cont:
        plt.suptitle('Distribution of Features', size = 20, weight='bold')
        plt.subplot(a, b, c)
        A=sns.kdeplot(data= df, x=i,hue=song_popularity,palette=theme[:-2], linewidth = 1.3,shade=True, alpha=0.35)
        plt.title(i)
        plt.xlabel(" ")
        c = c + 1
```




```
#plotting
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(18, 9))
fig.suptitle(' Highest and Lowest Correlation ', size = 20, weight='bold')
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
axs = [ax1, ax2]

#kdeplot
sns.kdeplot(data=df, y='energy', x='acousticness', ax
=ax1, color=heatmap[0])
ax1.set_title('Energy vs Acousticness', size = 14, wei
ght='bold', pad=20)

#kdeplot
sns.kdeplot(data=df, y='energy', x='loudness', ax=ax2
, color=heatmap[4])
ax2.set_title('Energy vs Loudness', size = 14, weight=
'bold', pad=20);
```

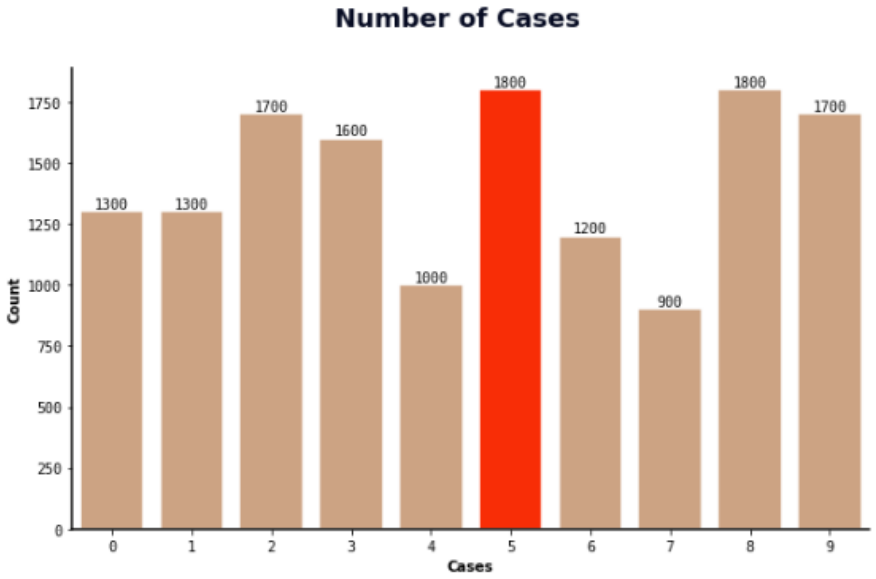


Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
colors = ["#e9d9c8","#cca383","#070c23","#f82d06",
"#e8c195","#cd7551","#a49995","#a3a49c","#6c7470
"]
sns.palplot(sns.color_palette(colors))


#plot
A = sns.countplot(train_df['case_num'],
            color=colors[1],
            edgecolor='white',
            linewidth=1.5,
            saturation=1.5)

#Patch
patch_h = []
for patch in A.patches:
    reading = patch.get_height()
    patch_h.append(reading)

idx_tallest = np.argmax(patch_h)
A.patches[idx_tallest].set_facecolor(colors[3])

#Lables
plt.ylabel('Count', weight='semibold', fontname = 'Georgia')
plt.xlabel('Cases', weight='semibold', fontname = 'Georgia')
plt.suptitle('Number of Cases', fontname = 'Georgia', weight=
'bold', size = 18, color = colors[2])
A.bar_label(A.containers[0], label_type='edge')

plt.show()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625

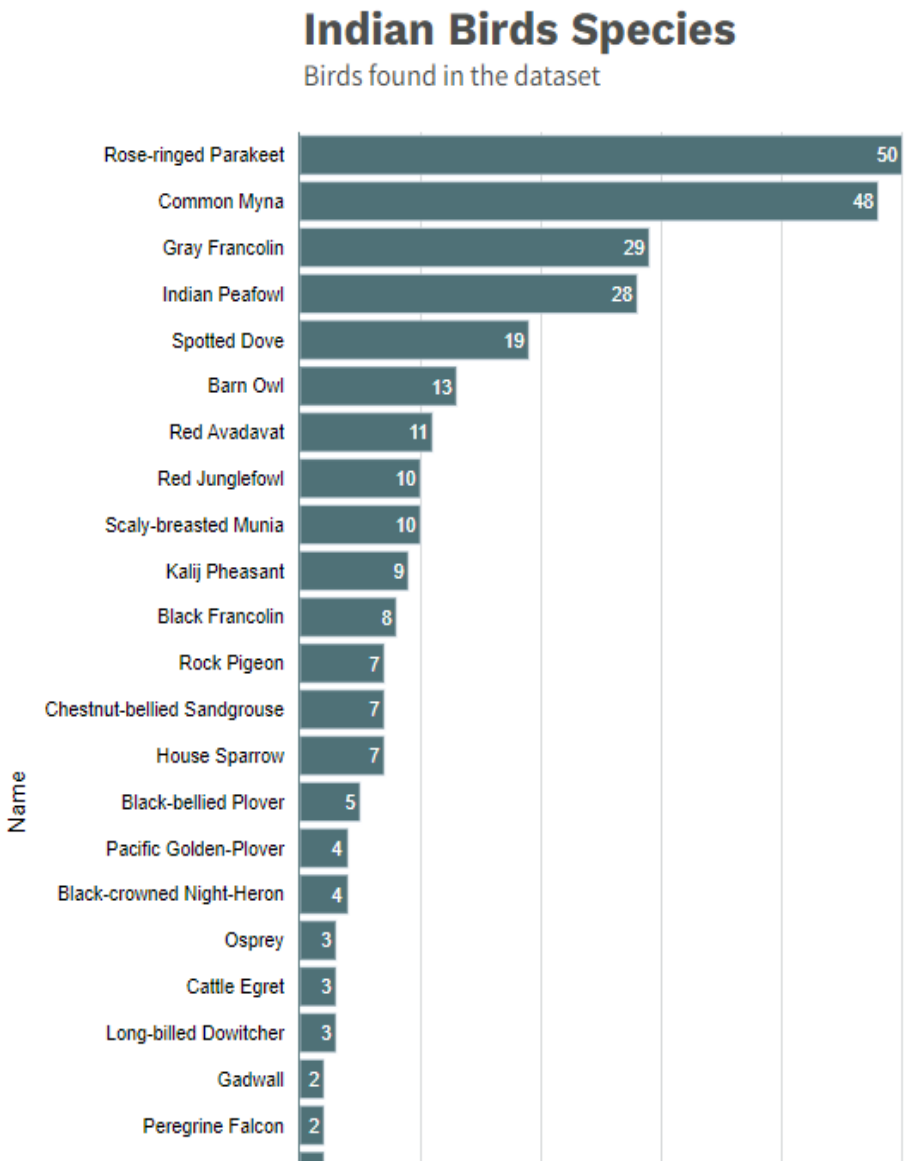



```
Datas = india_df["common_name"].value_counts().reset_inde
x().sort_values(by='common_name')

# Creating the bar chart
trace = go.Bar(
    y = Datas["index"],
    x = Datas["common_name"],
    orientation = "h",
    marker_color= "#4F7177",
    text = Datas["common_name"],
)

layout = dict(
    width = 600,
    height= 1000,
    plot_bgcolor = "#FFFFFF",
    font=dict(family='Arial',
              size=12,
              color='black'),
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
   margin = dict(
      l=0,
      r=0,
      b=100,
      t=100,
      pad=0
   ),
   xaxis = dict(showline=True, linewidth=1.45, linecolor="#4
F7177",gridcolor='#D5D7D8',
               #griddash='dot',
               title_text='Counts'),
   yaxis = dict(showline=True, linewidth=1.45, linecolor="#4F
7177",ticksuffix = "  ",title_text='Name'),
   bargap = 0.15,hoverlabel_bgcolor="#4F7177",hovermode="
x"
)

fig = go.Figure(data = trace, layout = layout)
fig.layout.xaxis.fixedrange = True
fig.layout.yaxis.fixedrange = True

#text
texter("Indian Birds Species",0.000,1.10,28,"Work Sans")
texter("Birds found in the dataset",0.000,1.06,18,"Source Sa
ns Pro")
texter("heyRobin!",1.00,-0.06,16,"Playfair Display")

fig.show()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

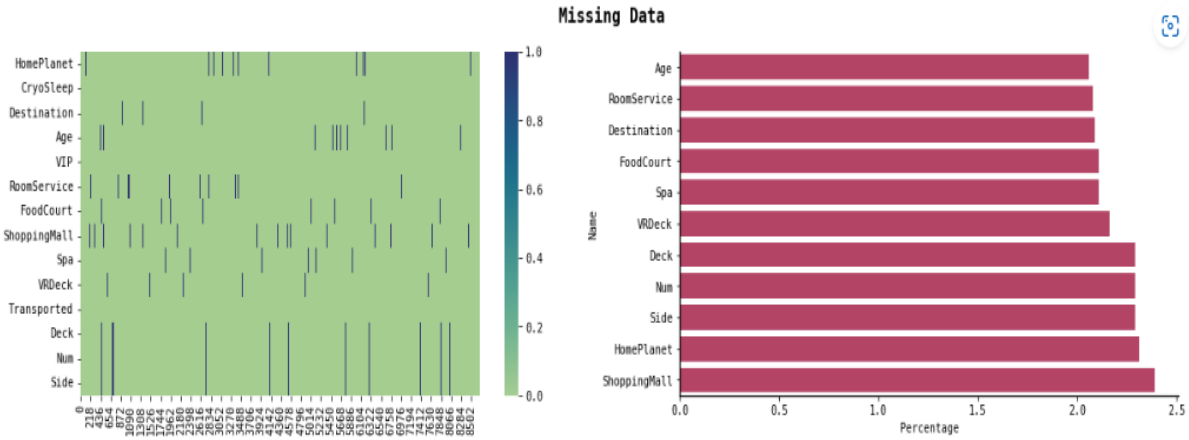


Missing Values:

```
fig, axes = plt.subplots(1,2, figsize=(20,5))
fig.suptitle('Missing Data', size = 15, weight='bold')

#first plot
sns.heatmap(train.isna().transpose(),
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
        cmap="crest",ax=axes[0])

#missing data
missing = round(train.isna().sum()/train.shape[0]* 100
,2)
missing = missing[missing>0].sort_values().to_frame(
)
missing.columns = ['Percentage']
missing.index.names = ['Name']
missing = missing.reset_index()
sns.barplot(data = missing, y= 'Name', x = 'Percentag
e',ax=axes[1],color=pal[0])
plt.show()
```



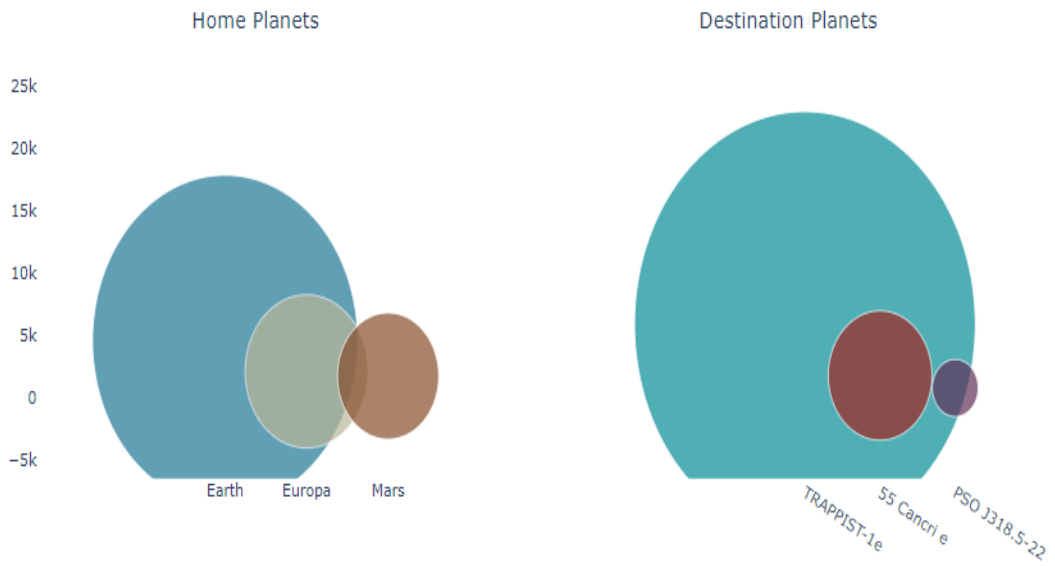

```
from plotly.subplots import make_subplots
import plotly.graph_objects as go
#Data
cnt_srshp =train['HomePlanet'].value_counts()
cnt_srsdes =train['Destination'].value_counts()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig = make_subplots(rows=2, cols=2, shared_yaxes=True,
    subplot_titles=("Home Planets","Destination Planets","VIP
","CryoSleep"))

#figure1
fig.add_trace(go.Scatter(
    x=cnt_srshp.index,
    y=cnt_srshp.values,
    mode='markers',
    marker=dict(
        sizemode = 'diameter',
        sizeref = 20,
        size = cnt_srshp.values,
        color = ['#1D7595','#B9B596','#864D29'])), 1, 1)
#figure2
fig.add_trace(go.Scatter(
    x=cnt_srsdes.index,
    y=cnt_srsdes.values,
    mode='markers',
    marker=dict(
        sizemode = 'diameter',
        sizeref = 20,
        size = cnt_srsdes.values,
        color = ['#048B95','#A1231F','#602F58'])),
            1, 2)

#figure3
fig.add_trace(go.Histogram(x=train["VIP"],
                           marker=dict(color=pal)),
              row=2, col=1)
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

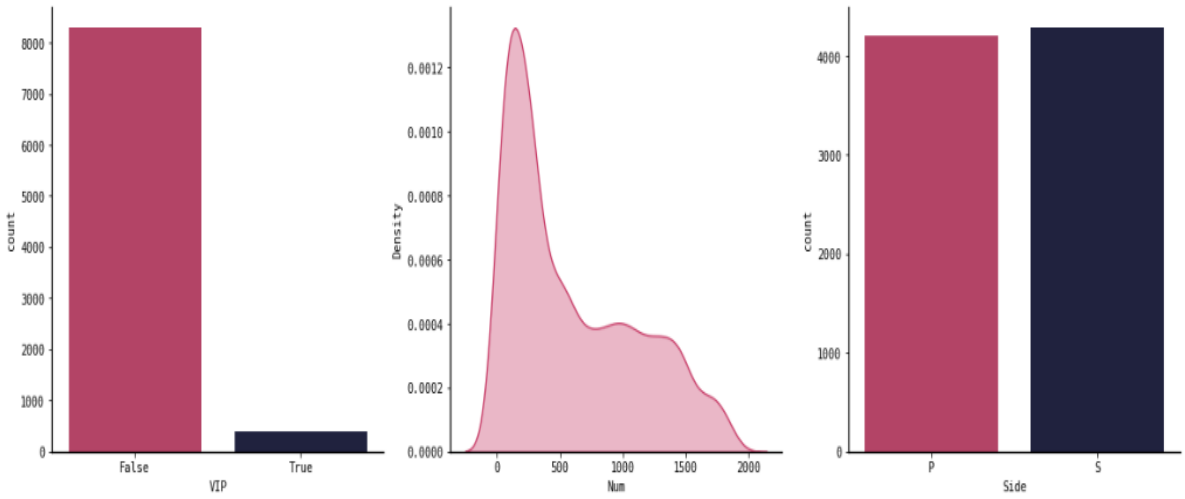
```
#figure4
fig.add_trace(go.Histogram(x=train["CryoSleep"],
                           marker=dict(color=pal)),
              row=2, col=2)

fig.update_layout(height=1000,width=1000, coloraxis=dict(c
olorscale='Bluered_r'), showlegend=False,
    title_x=0.9,
    titlefont=dict(size = 2, color='black', family='Space Mono'),
    plot_bgcolor='rgba(0,0,0,0)'
)
fig.show()
```




```
fig, axes = plt.subplots(1,3, figsize=(20,6))
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
sns.countplot(train["VIP"], ax=axes[0],palette =pal)
sns.kdeplot(train["Num"],linewidth = 1.3,shade=True,
alpha=0.35, ax=axes[1],color=pal[0])
sns.countplot(train["Side"], ax=axes[2],palette =pal)

plt.show()
```



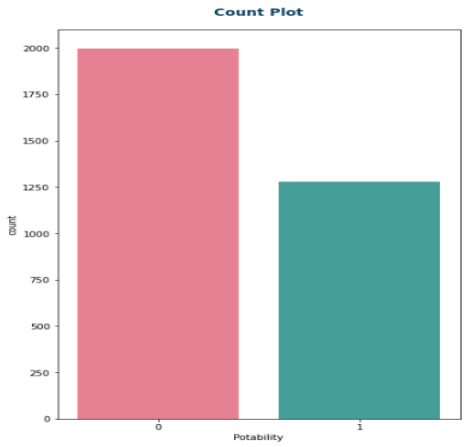

```
import matplotlib as mlb
import matplotlib.image as mpimg
from matplotlib.offsetbox import AnnotationBbox, OffsetImage

#plotting
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 11))
fig.suptitle(' Potablity of Water Quality ', size = 26, color = th
eme[3], weight='bold')
axs = [ax1, ax2]

#Count-Plot
sns.countplot(water_df['Potability'], ax=ax1, palette='husl')
```

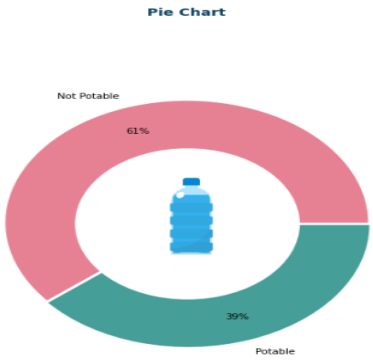
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
ax1.set_title('Count Plot', size = 14, color = theme[3], weight
='bold', pad=20)

#Data-2
names = ["Not Potable", "Potable"]
values = water_df['Potability'].value_counts()
colors = ["#E68193","#459E97"]
explode = (0.01, 0.01)

#Doughnut-chart
ax2.pie(x= values,labels =names, colors=colors,autopct='%1.
0f%%', pctdistance=0.8,explode=explode)

#draw-circle
centre_circle = plt.Circle((0,0),0.62,fc='white')
ax2.add_artist(centre_circle)
ax2.axis('equal')

ax2.set_title('Pie Chart', size = 14, color = theme[3], weight='
bold', pad=20)

#Image

path = mpimg.imread('../input/water/water bottle.png')
imagebox = OffsetImage(path , zoom=0.3)
xy = (0.5, 0.7)
ab = AnnotationBbox(imagebox, xy, frameon=False, pad=1, x
ybox=(0.02, 0.05))
ax2.add_artist(ab)
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
plt.subplots_adjust(left=None, bottom=None, right=None, top
=0.8, wspace=0.4, hspace=None);
```





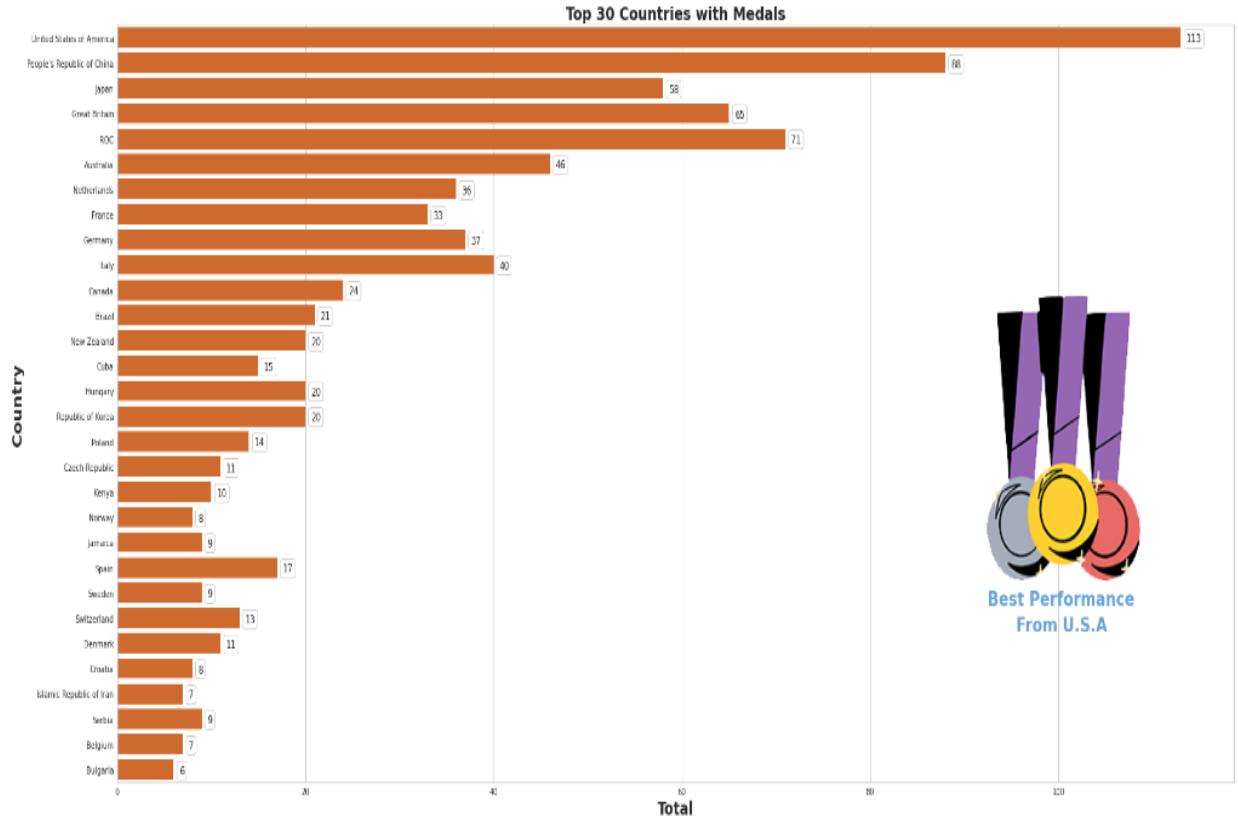

```
#Figure with Image
import matplotlib as mlb
import matplotlib.image as mpimg
from matplotlib.offsetbox import AnnotationBbox, Off
setImage

plt.figure(figsize=(27,15));
ax = sns.barplot(y ='Country',
                 x='Total',
                 data=medals[:30],
                 color=olympics_col[3],
                 zorder=2,
                 linewidth=0,
                 orient='h',
                 saturation=1,
                 alpha=1,
                 )
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
#Lables
ax.set_xlabel("Total",fontsize=20, weight='bold')
ax.set_ylabel("Country",fontsize=20, weight='bold')
ax.tick_params(labelsize=10, width=0.5, length=1.5)
plt.title("Top 30 Countries with Medals",size=20,weig
ht='bold')

#Patches
for a in ax.patches:
    value = f'{a.get_width():.0f}'
    x = a.get_x() + a.get_width() + 0.60
    y = a.get_y() + a.get_height() / 1.8
    ax.text(x, y, value, ha='left', va='center', fontsize=12,
            bbox=dict(facecolor='none', edgecolor='black', box
style='round', linewidth=0.2))

#image
path = mpimg.imread('../input/font-worksans/medal-cr
op.png')
imagebox = OffsetImage(path , zoom=1.6)
xy = (0.5, 0.7)
ab = AnnotationBbox(imagebox, xy, frameon=False, p
ad=1, xybox=(100.5, 16))

ax.add_artist(ab)
ax.text(x = 92.5, y = 22.5, s = 'Best Performance', fon
tsize=22, weight = 'bold',color=olympics_col[1])
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
ax.text(x = 95.5, y = 23.5, s = 'From U.S.A', fontsize=22, weight = 'bold',color=olympics_col[1]);
```

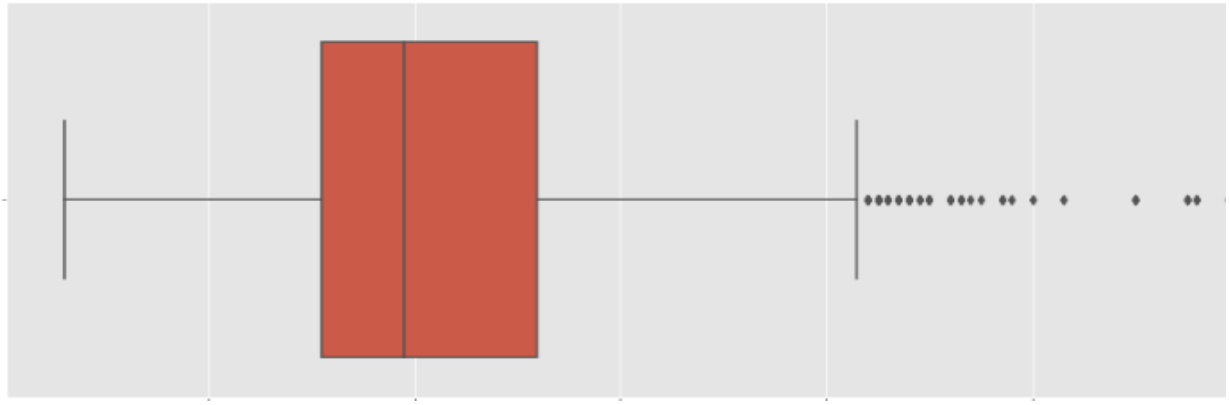



```
for col in numeric_features[1:]:
    fig = plt.figure(figsize=(9, 6))
    ax = fig.gca()
    feature = data[col]
    feature.hist(bins=50, ax = ax)
    ax.axvline(feature.mean(), color='magenta', linestyle='dashed', linewidth=2)
    ax.axvline(feature.median(), color='cyan', linestyle='dashed', linewidth=2)
    ax.set_title(col)
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

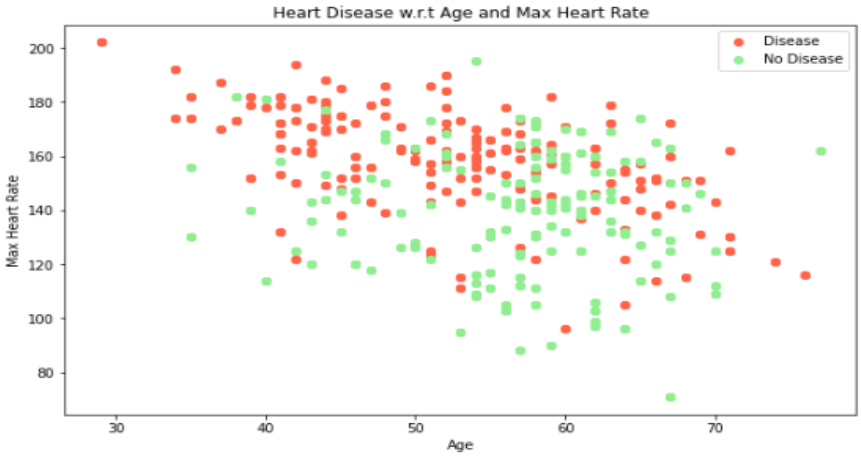
```
plt.show()
```


```
data1=mydata[['Parameter 1']]
for i in data1.columns:
    plt.figure(figsize=(15,6))
    sns.boxplot(data1[i])
    plt.xticks(rotation=90)
    plt.show()
```



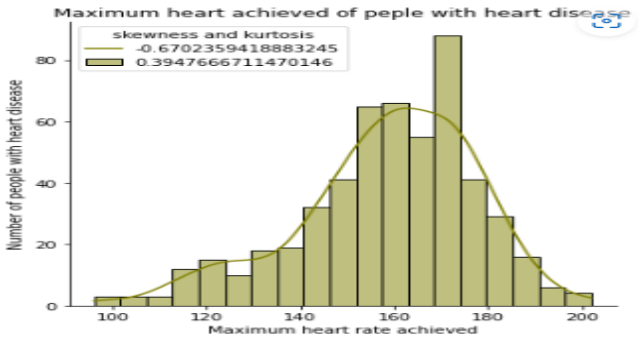
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
# Creating a figure
plt.figure(figsize=(10,6))

#plotting the values for people who have heart disease
plt.scatter(df.age[df.target==1],
            df.thalach[df.target==1],
            c="tomato")

#plotting the values for people who doesn't have heart disea
se
plt.scatter(df.age[df.target==0],
            df.thalach[df.target==0],
            c="lightgreen")

# Addind info
plt.title("Heart Disease w.r.t Age and Max Heart Rate")
plt.xlabel("Age")
plt.legend(["Disease", "No Disease"])
plt.ylabel("Max Heart Rate");
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

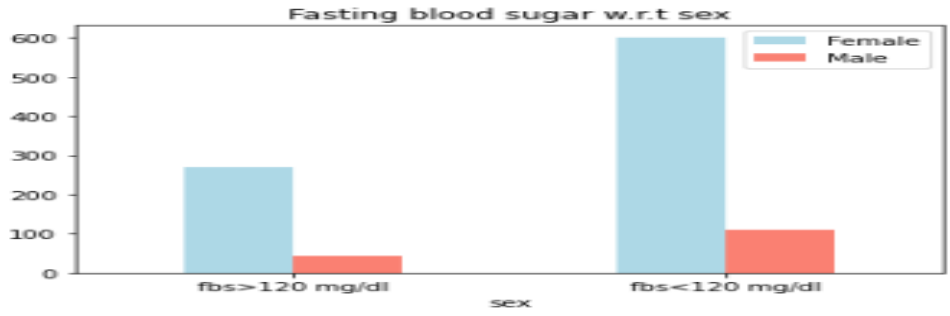
```
sns.displot(x = df.thalach[df.target==1], data = df, kde = True, color= 'olive')#
skewness=str(df.thalach[df.target==1].skew())
kurtosis=str(df.thalach[df.target==1].kurt())
plt.legend([skewness,kurtosis],title=("skewness and kurtosis"))
plt.title("Maximum heart achieved of peple with heart disease")
plt.xlabel("Maximum heart rate achieved")
plt.ylabel("Number of people with heart disease");
```




```
pd.crosstab(df.sex, df.fbs)
```

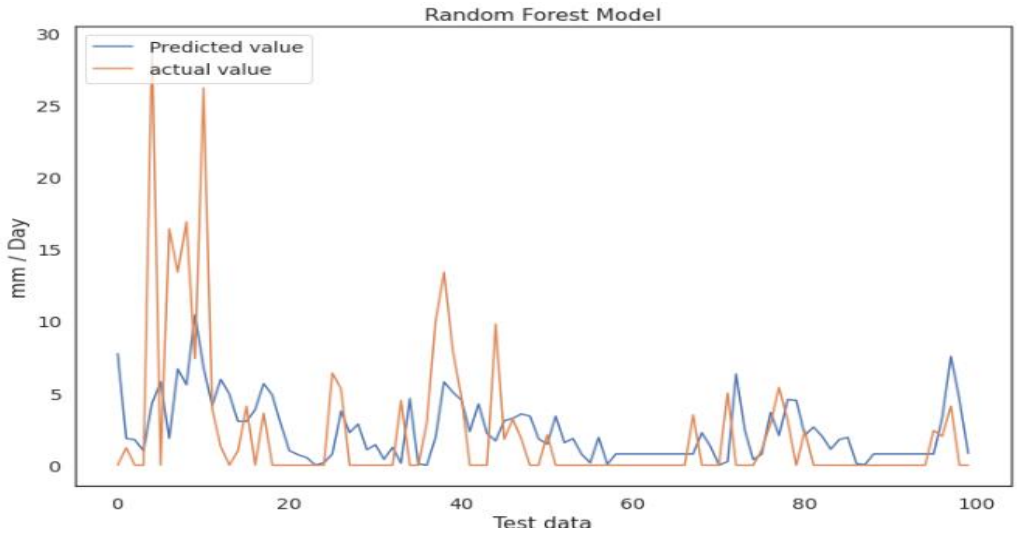
fbs	0	1
sex		
0	270	42
1	602	111

```
fig = pd.crosstab(df.sex, df.fbs).plot(kind = 'bar', color = ['lightblue', 'salmon'])
plt.title("Fasting blood sugar w.r.t sex")
fig.set_xticklabels(labels=['fbs>120 mg/dl', 'fbs<120 mg/dl'], rotation=0)
plt.legend(['Female', 'Male']);
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

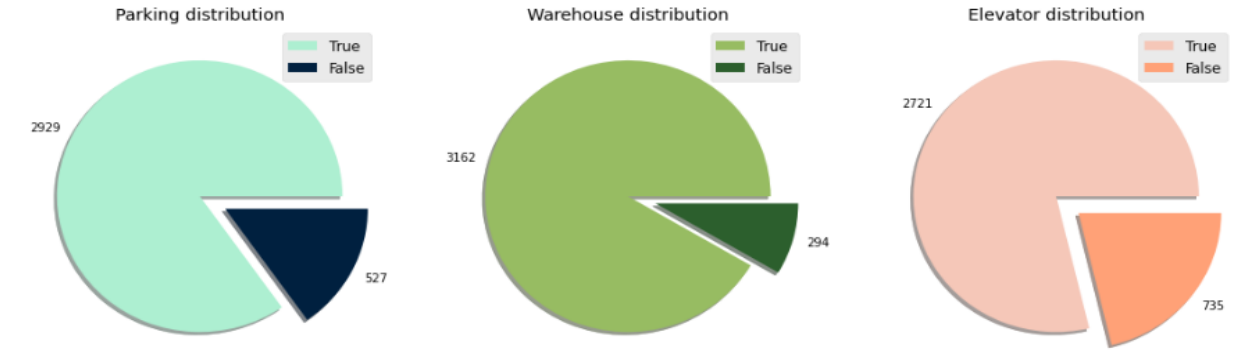
```
pred = rf_model.predict(x_test)
plt.rcParams['figure.figsize'] = (12,8)
plt.plot(pred, label='Predicted value')
plt.plot(y_test, label='actual value')
plt.legend(loc ="upper left")
plt.title('Random Forest Model')
plt.xlabel('Test data')
plt.ylabel('mm / Day')
plt.show()
```




```
fig, ax = plt.subplots(ncols=3, figsize=(18,6))

colors = [['#ADEFD1FF', '#00203FFF'], ['#97BC62FF', '
#2C5F2D'], ['#F5C7B8FF', '#FFA177FF']]
explode = [0, 0.2]
columns = ['Parking', 'Warehouse', 'Elevator']
for i in range(3):
        data = df[columns[i]].value_counts()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

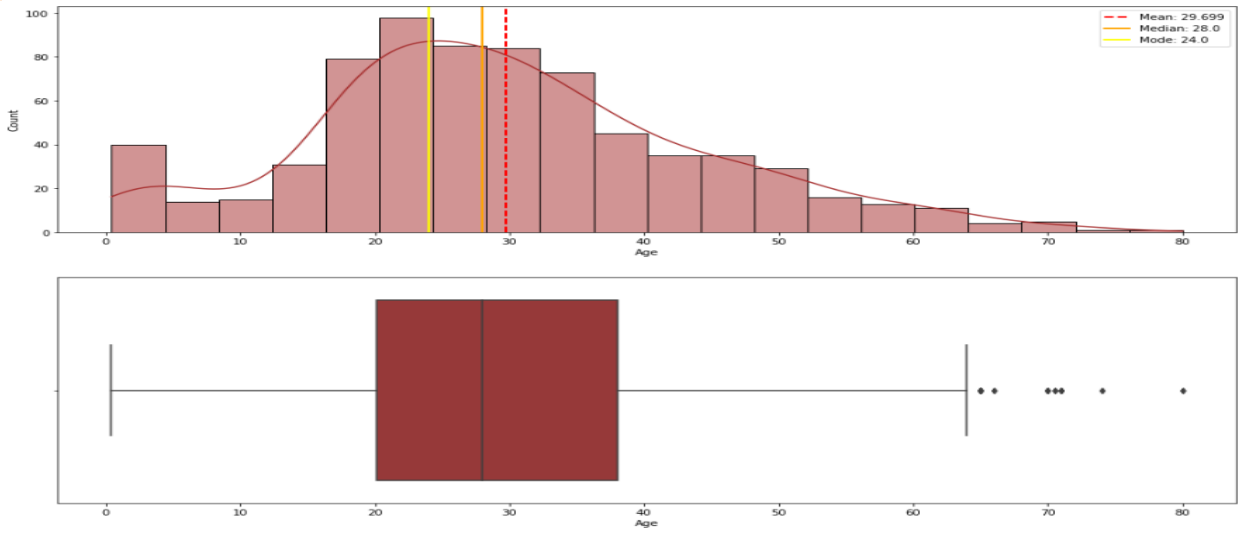
```
        ax[i].pie(data, labels=data.values, explode=expl
ode, colors=colors[i], shadow=True)
        ax[i].legend(labels=data.index, fontsize='large')
        ax[i].set_title('{} distribution'.format(columns[i]))
```



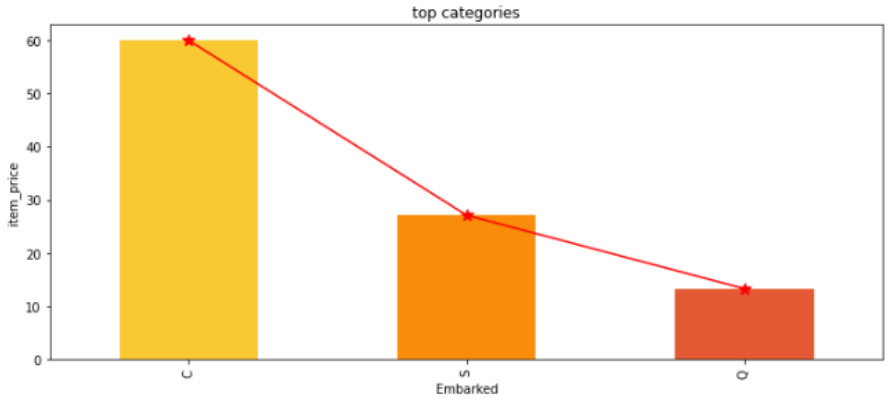
```
def plot_hist(feature):
    fig, ax = plt.subplots(2, 1, figsize=(17, 12))

    sns.histplot(data = titanic[feature], kde = True, ax =
ax[0],color="Brown")

    ax[0].axvline(x = titanic[feature].mean(), color = 'r',
linestyle = '--', linewidth = 2, label = 'Mean:
{}'.format(round(titanic[feature].mean(), 3)))
    ax[0].axvline(x = titanic[feature].median(), color =
'orange', linewidth = 2, label = 'Median:
{}'.format(round(titanic[feature].median(), 3)))
    ax[0].axvline(x = statistics.mode(titanic[feature]), color =
'yellow', linewidth = 2, label = 'Mode:
{}'.format(statistics.mode(titanic[feature])))
    ax[0].legend()

    sns.boxplot(x = titanic[feature], ax = ax[1],color="Brown")
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
    plt.show()
plot_hist('Age')
```

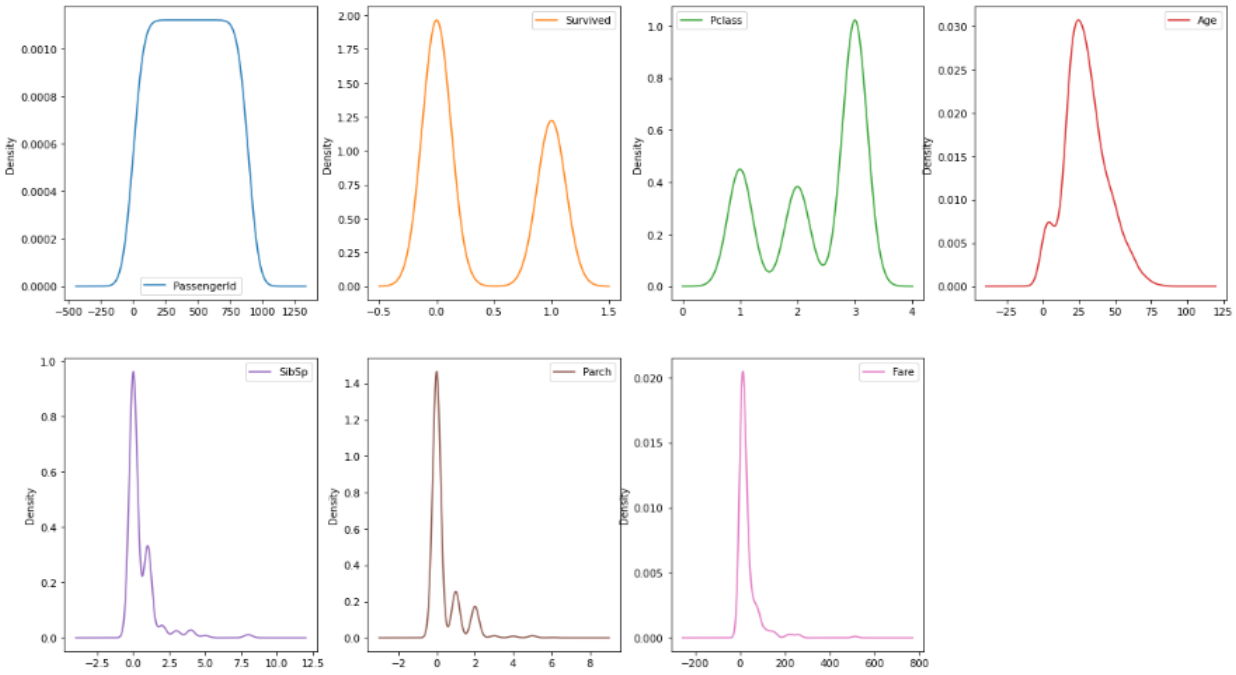



```
plt.figure(figsize=(12,5))
plt.title('top categories')
plt.ylabel('item_price')
titanic.groupby('Embarked')['Fare'].mean().sort_values(ascending=False)[0:15].plot(kind='line', marker='*', color='red', ms=10)
titanic.groupby('Embarked')['Fare'].mean().sort_values(ascending=False)[0:15].plot(kind='bar',color=sns.color_palette("inferno_r", 7))
plt.show()
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
numeric_feature = titanic.dtypes!=object
final_numeric_feature =
titanic.columns[numeric_feature].tolist()

titanic[final_numeric_feature].plot(kind='density',
subplots=True, layout=(2,4), sharex=False, figsize= (20,12))
plt.show()
```




```
df.describe().round(2).T.sort_values(by='std' , ascending = False)\
                .style.background_gradient(cmap='GnBu')\
                .bar(subset=["max"], color='#BB0000')\
                .bar(subset=["min",], color='green')\
                .bar(subset=["mean",], color='Orange')\
                .bar(subset=['std'], color='pink')\
                .bar(subset=['50%'], color='magenta')
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

	count	mean	std	min	25%	50%	75%	max
motor_speed	1330816.000000	2202.080000	1859.660000	-275.550000	317.110000	1999.980000	3760.640000	6000.020000
i_q	1330816.000000	37.410000	92.180000	-293.430000	1.100000	15.770000	100.610000	301.710000
torque	1330816.000000	31.110000	77.140000	-246.470000	-0.140000	10.860000	91.600000	261.010000
i_d	1330816.000000	-68.720000	64.930000	-278.000000	-115.410000	-51.090000	-2.980000	0.050000
u_d	1330816.000000	-25.130000	63.090000	-131.530000	-78.690000	-7.430000	1.470000	131.470000
u_q	1330816.000000	54.280000	44.170000	-25.290000	12.070000	48.940000	90.030000	133.040000
stator_winding	1330816.000000	66.340000	28.670000	18.590000	42.790000	65.110000	88.140000	141.360000
profile_id	1330816.000000	40.790000	25.050000	2.000000	17.000000	43.000000	65.000000	81.000000
stator_tooth	1330816.000000	56.880000	22.950000	18.130000	38.420000	56.040000	75.590000	111.950000
coolant	1330816.000000	36.230000	21.790000	10.620000	18.700000	26.900000	49.860000	101.600000
stator_yoke	1330816.000000	48.190000	19.990000	18.080000	31.990000	45.630000	61.460000	101.150000
pm	1330816.000000	58.510000	19.000000	20.860000	43.150000	60.270000	72.010000	113.610000
ambient	1330816.000000	24.570000	1.930000	8.780000	23.180000	24.800000	26.220000	30.710000


```
def highlight_min(s, props=''):
    return np.where(s == np.nanmin(s.values), props, '')
titanic.describe().style.apply(highlight_min, props='color:yellow;background-color:Grey', axis=0)
```

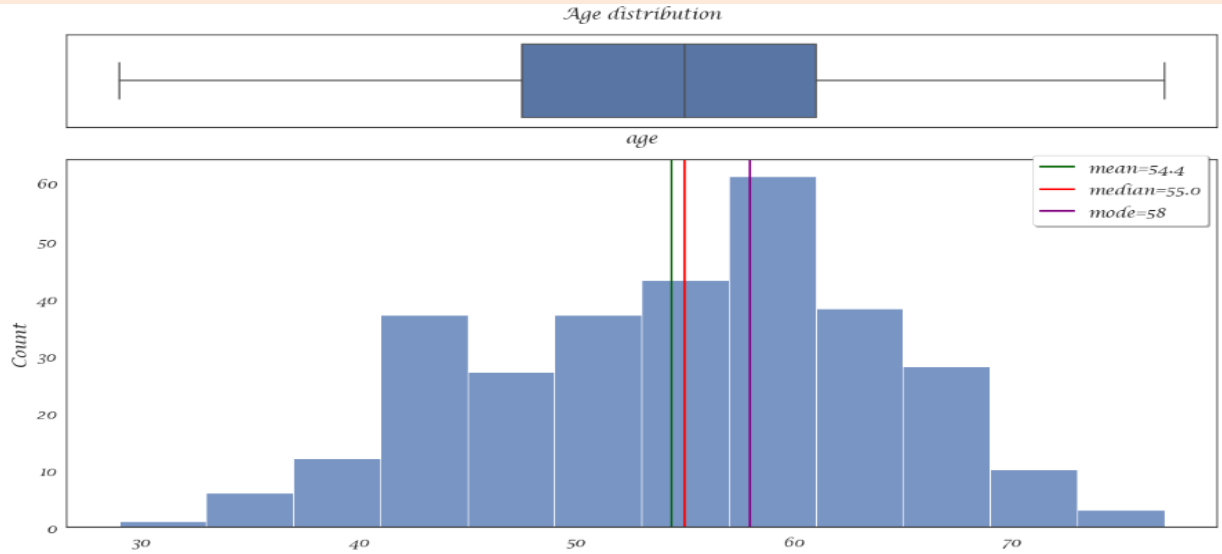
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
titanic[titanic["Age"] >= 50].describe().style.background_gradient(cmap='RdPu')
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000	74.000000
mean	445.675676	0.364865	1.554054	57.540541	0.270270	0.283784	46.364415
std	242.330133	0.484678	0.742854	6.767042	0.504698	0.692826	49.480722
min	7.000000	0.000000	1.000000	50.000000	0.000000	0.000000	6.237500
25%	250.750000	0.000000	1.000000	52.000000	0.000000	0.000000	12.643750
50%	483.500000	0.000000	1.000000	56.000000	0.000000	0.000000	27.631250
75%	631.750000	1.000000	2.000000	61.750000	0.000000	0.000000	68.744800
max	880.000000	1.000000	3.000000	80.000000	2.000000	4.000000	263.000000

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig, ax = plt.subplots(2, 1, sharex=True,
figsize=(17,10),gridspec_kw={"height_ratios": (.2, .8)})
ax[0].set_title('Age distribution',fontsize=18,pad=20)
sns.boxplot(x='age', data=heart, ax=ax[0])
ax[0].set(yticks=[])
sns.histplot(x='age', data=heart, ax=ax[1])
ax[1].set_xlabel(col, fontsize=16)
plt.axvline(heart['age'].mean(), color='darkgreen',
linewidth=2.2, label='mean=' +
str(np.round(heart['age'].mean(),1)))
plt.axvline(heart['age'].median(), color='red', linewidth=2.2,
label='median='+ str(np.round(heart['age'].median(),1)))
plt.axvline(heart['age'].mode()[0], color='purple',
linewidth=2.2, label='mode='+ str(heart['age'].mode()[0]))
plt.legend(bbox_to_anchor=(1, 1.03), ncol=1, fontsize=17,
fancybox=True, shadow=True, frameon=True)
plt.tight_layout()
plt.show()
```




```
plt.rcParams['font.family'] = 'Lucida Calligraphy'
```

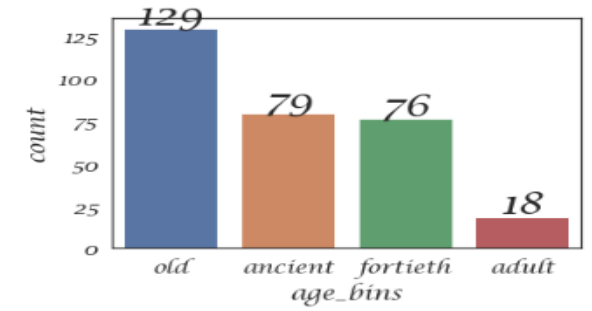
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
plt.rcParams['font.size'] = 30

heart["age_bins"]= pd.cut(heart["age"] , bins=[29 , 40 , 50 , 60 ,  80] , labels=["adult" , "fortieth" , "old" , "ancient"] )

def count_plot(data  , x=None , y=None  , figsize =None , title =None , color =None , prop=False , rotation_x =0 ):
    if x is None and y is None :
        raise("Expected y or x")
    if x is not None and y is not None:
        raise("Expected y or x not both")
    count_type = data[y if x is None else x].value_counts(ascending =False)
    Sum = count_type.sum()
    type_order = count_type.index
    plt.figure(figsize=figsize if figsize is None else (12 , 7))
    if x is None:
        sns.countplot(data = data , y=y , color = color ,order=type_order)
        if prop==True:
            for i in range(len(count_type)):
                count = count_type[i]
                pct_string ="{:0.1f}%".format(100*count/Sum)
                plt.text(count+1 , i , pct_string , va="center")
        if prop==False:
            for i in range(len(count_type)):
                count = count_type[i]
                pct_string ="{}".format(count)
                plt.text(count+1 , i , pct_string , va="center")
        plt.title(title)
        plt.show()
```

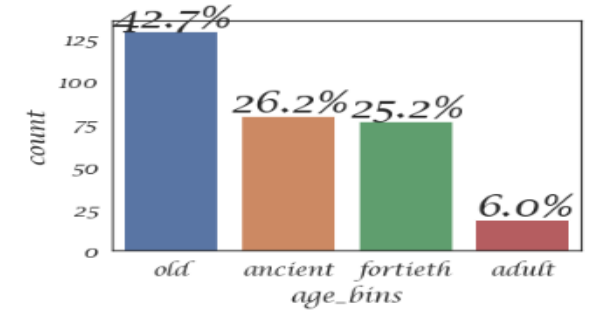
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
    if y is None :
        sns.countplot(data = data , x = x , color = color , order =
type_order)
        locs , labels =plt.xticks(rotation = rotation_x)
        if prop == True :
            for loc , label in zip(locs , labels):
                count = count_type[label.get_text()]
                pct_string ="{:0.1f}%".format(100*count/Sum)
                plt.text(loc , count+2 ,pct_string,ha ="center")
        if prop==False :
            for loc , label in zip(locs , labels):
                count = count_type[label.get_text()]
                pct_string ="{}".format(count)
                plt.text(loc , count+2 ,pct_string,ha ="center")
        plt.title(title)
        plt.show()
```

```
1 count_plot(data = heart , x ="age_bins")
```

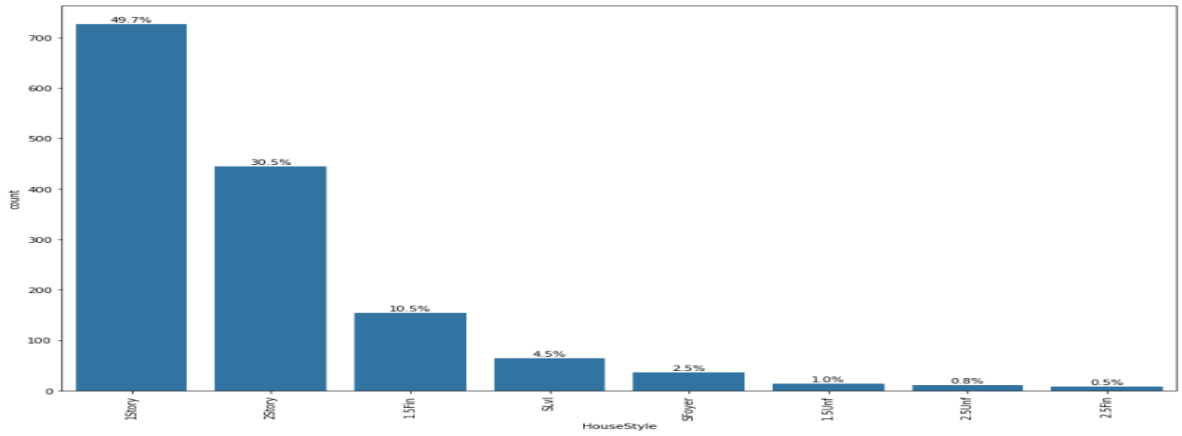



```
1 count_plot(data = heart ,prop=True, x ="age_bins")
```



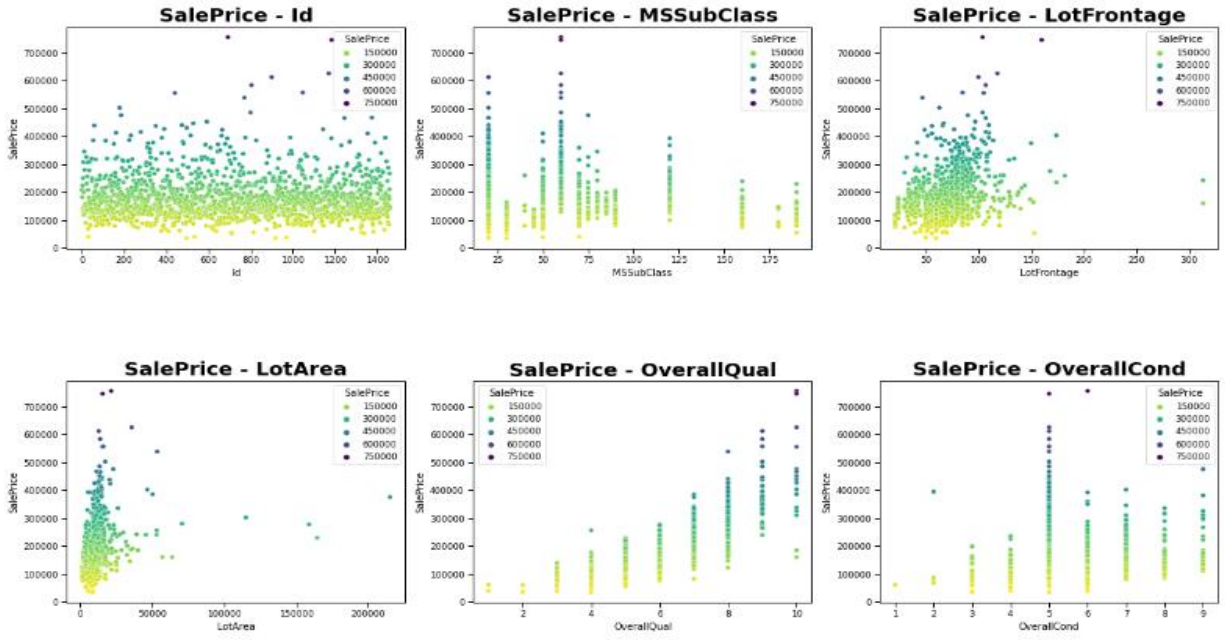
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70239014/

```
# Barchart sorted by frequency
base_color = sns.color_palette()[0]
cat_order = train_eda[col_name].value_counts().index
plt.figure(figsize=(15,10))
plt.xticks(rotation = 90)
sns.countplot(data = train_eda, x = col_name, order = cat_order, color =
base_color);

# add annotations
n_points = train_eda.shape[0]
cat_counts = train_eda[col_name].value_counts()
locs, labels = plt.xticks() # get the current tick locations and labels

# loop through each pair of locations and labels
for loc, label in zip(locs, labels):

    # get the text property for the label to get the correct count
    count = cat_counts[label.get_text()]
    pct_string = '{:0.1f}%'.format(100*count/n_points)

    # print the annotation just below the top of the bar
    plt.text(loc, count+4, pct_string, ha = 'center', color = 'black')
```



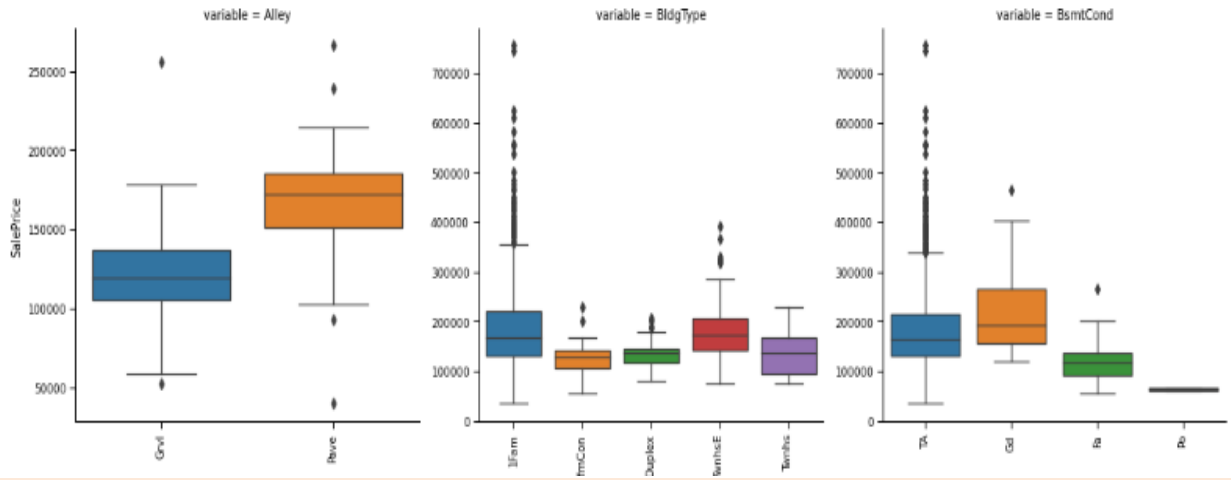
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
train = pd.read_csv('train_housing.csv')

#Visualising numerical predictor variables with Target Variables
train_num = train.select_dtypes(include=['int64','float64'])
fig,axs= plt.subplots(12,3,figsize=(20,80))
#adjust horizontal space between plots
fig.subplots_adjust(hspace=0.6)
for i,ax in zip(train_num.columns,axs.flatten()):
    sns.scatterplot(x=i, y='SalePrice',
hue='SalePrice',data=train_num,ax=ax,palette='viridis_r')
    plt.xlabel(i,fontsize=12)
    plt.ylabel('SalePrice',fontsize=12)
    #ax.set_yticks(np.arange(0,900001,100000))
    ax.set_title('SalePrice'+' -
'+str(i),fontweight='bold',size=20)
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
train = pd.read_csv('train_housing.csv')
categorical = train.select_dtypes(include=['object'])
##Visualising Categorical predictor variables with Target Variables
def facetgrid_boxplot(x, y, **kwargs):
    sns.boxplot(x=x, y=y)
    x=plt.xticks(rotation=90)

f = pd.melt(train, id_vars=['SalePrice'],
value_vars=sorted(train[categorical.columns]))
g = sns.FacetGrid(f, col="variable", col_wrap=3,
sharex=False, sharey=False, size=5)
g = g.map(facetgrid_boxplot, "value", "SalePrice")
```

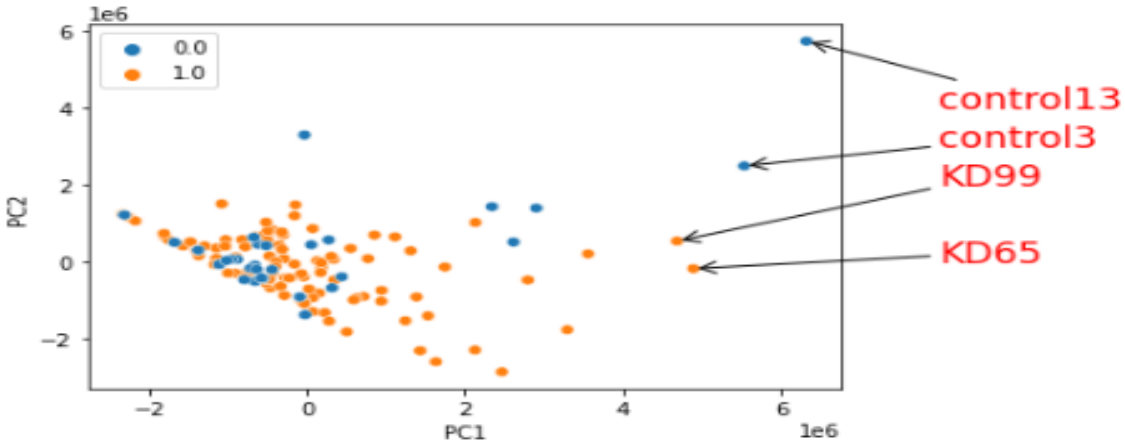



```
import matplotlib.pyplot as plt
import seaborn as sns

sns.scatterplot(x=df.iloc[:,0], y=df.iloc[:,1], hue=y)
plt.annotate("KD65", (df.iloc[64,0], df.iloc[64,1]), (8*1e6, 1),
arrowprops=dict(arrowstyle="->"), fontsize="xx-large",c='red'
)
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

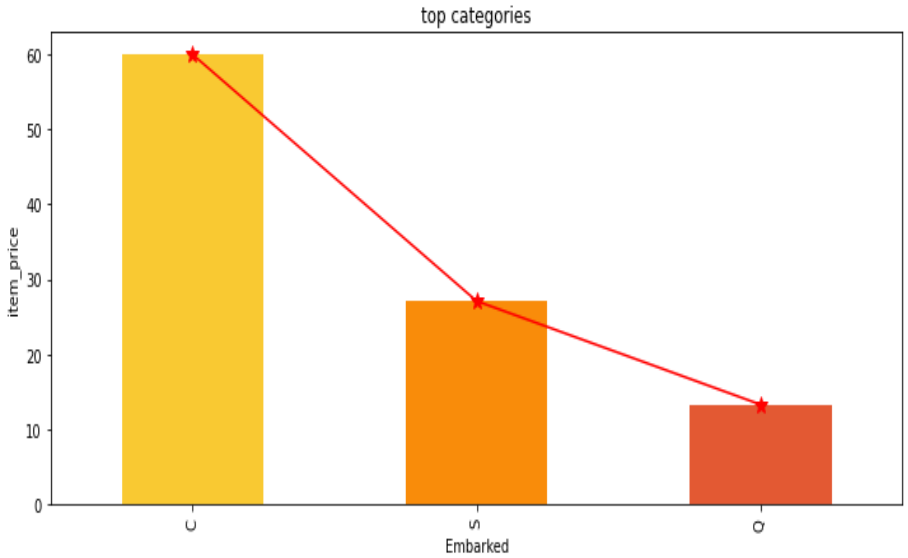
```
plt.annotate("KD99", (df.iloc[98,0], df.iloc[98,1]), (8*1e6, 2*1e6), arrowprops=dict(arrowstyle="->"), fontsize="xx-large",c='red')
plt.annotate("control3", (df.iloc[107,0], df.iloc[107,1]), (8*1e6, 3*1e6), arrowprops=dict(arrowstyle="->"), fontsize="xx-large",c='red')
plt.annotate("control13", (df.iloc[117,0], df.iloc[117,1]), (8*1e6, 4*1e6), arrowprops=dict(arrowstyle="->"), fontsize="xx-large",c='red')
```

```
Text(8000000.0, 4000000.0, 'control13')
```

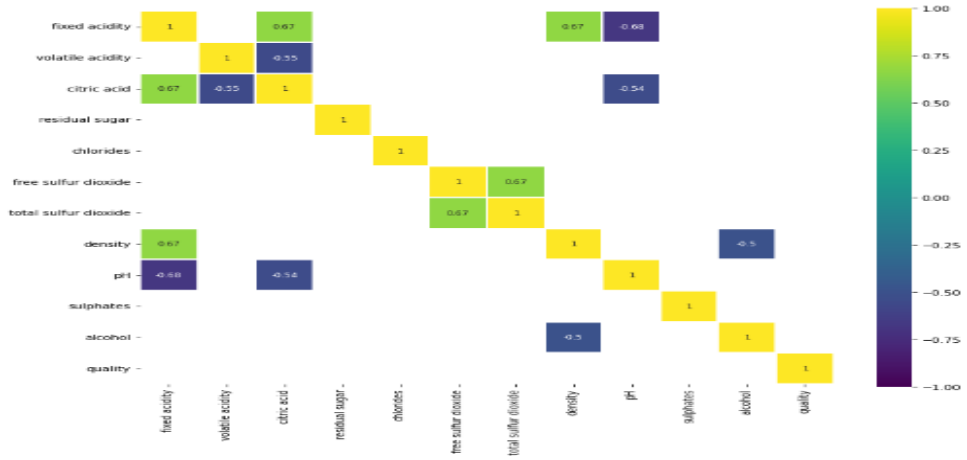



```
plt.figure(figsize=(12,5))
plt.title('top categories')
plt.ylabel('item_price')
titanic.groupby('Embarked')['Fare'].mean().sort_values(ascending=False)[0:15].plot(kind='line', marker='*', color='red', ms=10)
titanic.groupby('Embarked')['Fare'].mean().sort_values(ascending=False)[0:15].plot(kind='bar',color=sns.color_palette("inferno_r", 7))
plt.show()
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/



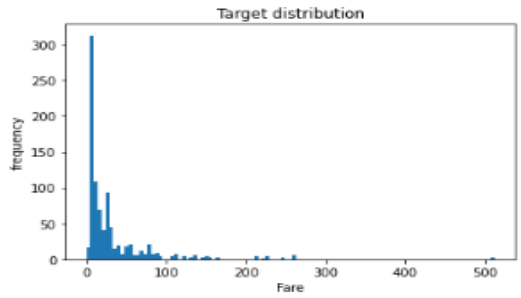

```
corr = wine.corr() # We already examined SalePrice correlations
plt.figure(figsize=(12, 10))

sns.heatmap(corr[(corr >= 0.5) | (corr <= -0.4)],
cmap='viridis', vmax=1.0, vmin=-1.0,
linewidths=0.1,annot=True, annot_kws={"size": 8},
square=True);
```

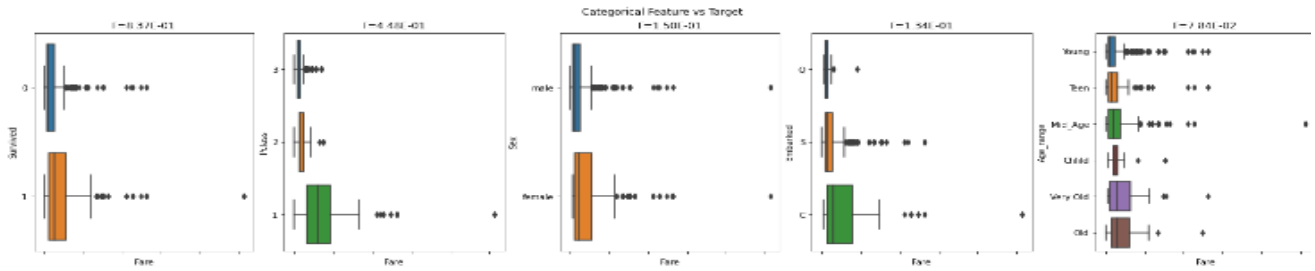


Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
import dabl
dabl.plot(titanic, 'Fare');
```

```
Target looks like regression
```





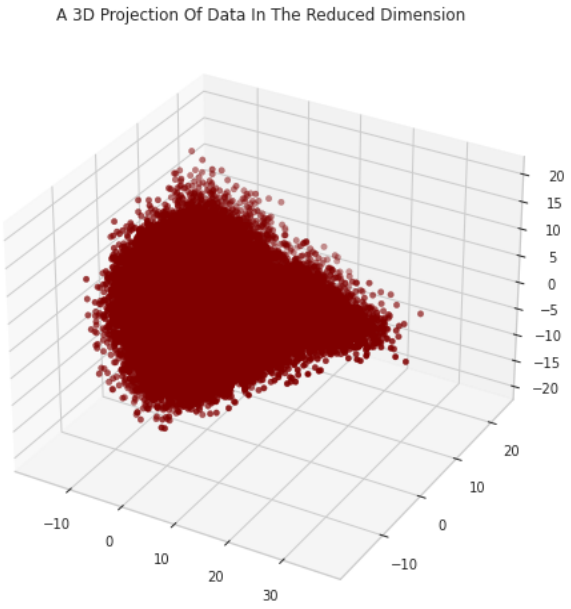
Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

Mastering Data Visualization Techniques (Part 4)

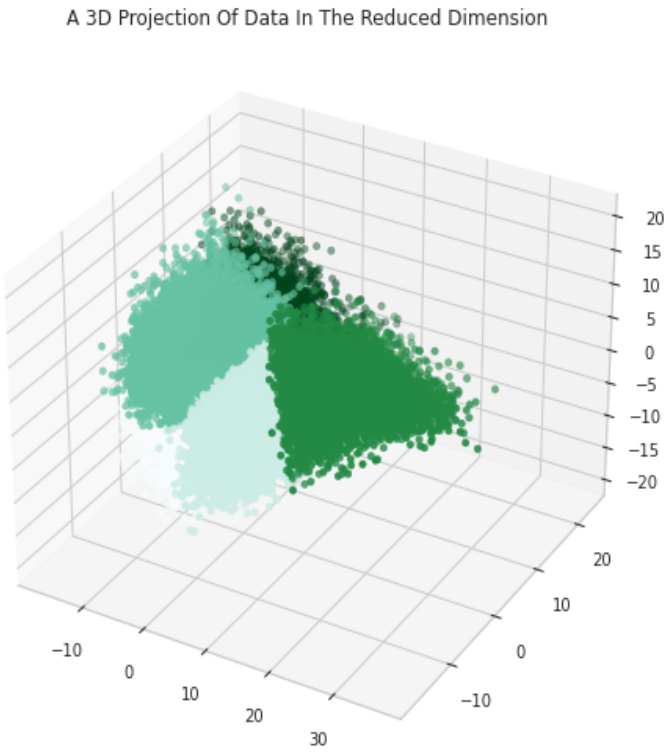
Prepared by: Syed Afroz Ali

```
X = dataset.copy()
from sklearn.decomposition import PCA
pca = PCA(n_components=3)
pca.fit(X)
PCA_ds = pd.DataFrame(pca.transform(X), columns=(["col1","col2", "col3"]))

# A 3D Projection Of Data In The Reduced Dimension
x =PCA_ds["col1"]
y =PCA_ds["col2"]
z =PCA_ds["col3"]
#To plot
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x, y, z, c="maroon", marker="o" )
ax.set_title("A 3D Projection Of Data In The Reduced Dimension")
plt.show()
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
# A 3D Projection Of Data In The Reduced Dimension
x =PCA_ds["col1"]
y =PCA_ds["col2"]
z =PCA_ds["col3"]

#To plot
fig = plt.figure(figsize=(10,8))
ax = fig.add_subplot(111, projection="3d")
ax.scatter(x, y, z, c=labels, marker="o", cmap="BuGn")
ax.set_title("A 3D Projection Of Data In The Reduced Dimension")
plt.show()
```

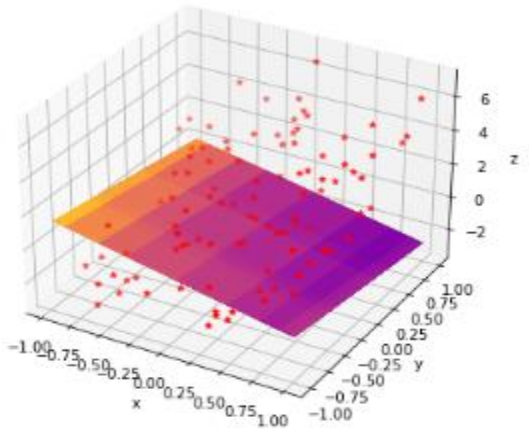



```
for i in range(0, 10):
    fig = plt.figure(figsize=(8, 6))
    ax = plt.axes(projection="3d")

    ax.scatter(x, y, z, marker='*', color='red')

    X, Y = np.meshgrid(x, y)
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

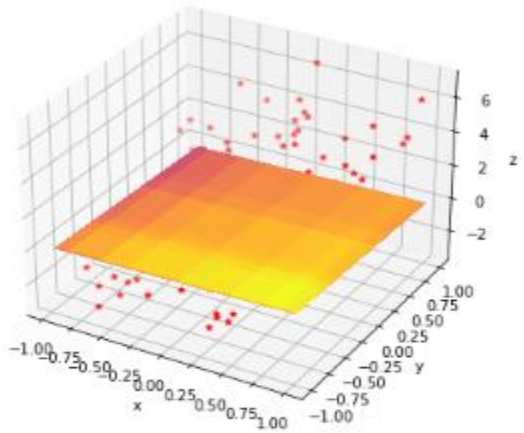
```
Z = theta_0[i]*X + theta_1[i]*Y + theta_2[i]
ax.plot_surface(X, Y, Z, cmap='plasma')

ax.set_xlabel("x")
ax.set_ylabel("y")
ax.set_zlabel("z")
ax.set_title("Thetas: {},{},{}".format(theta_0[i], theta_1[i], theta_2[i]))
plt.show()
print(40*"=")
```

```
Thetas: -1.4035055897613047,-0.763083683174633,-0.4754543787231061
```



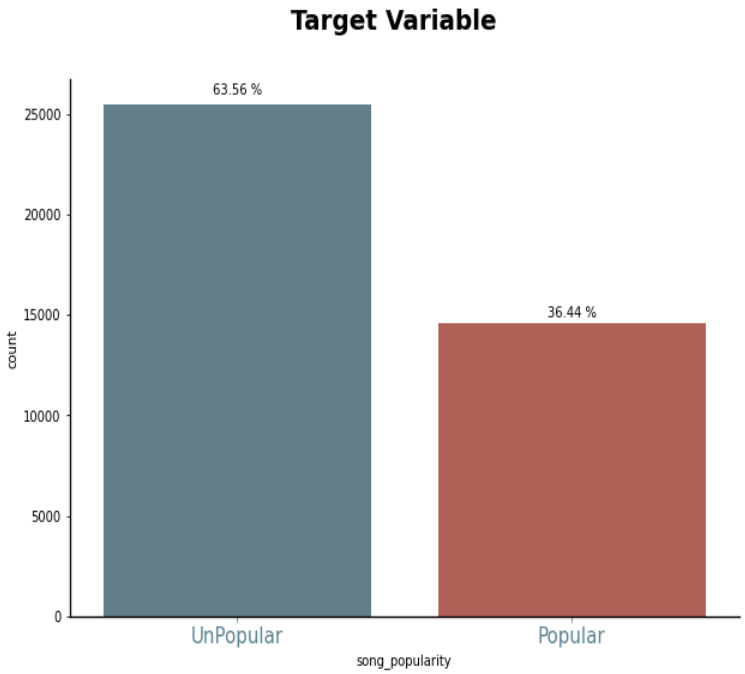

```
========================================
```

```
Thetas: 0.12226339350069884,-0.22698056361690117,0.006858537493075969
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

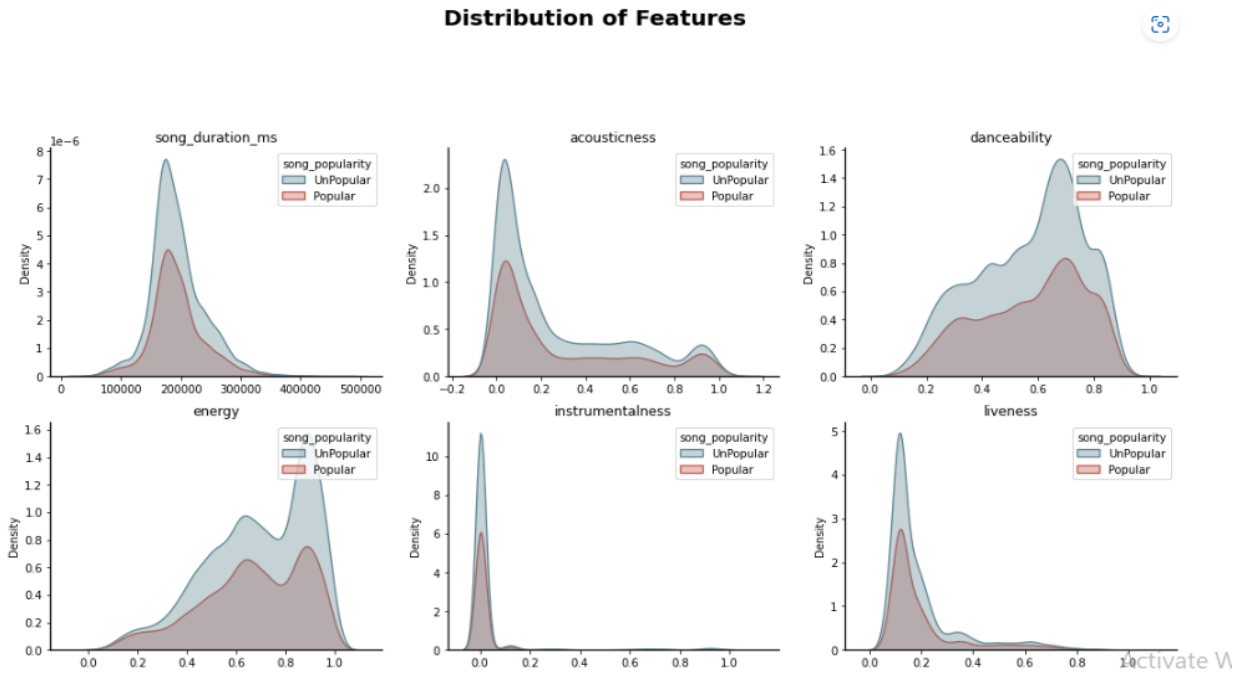
```
plt.suptitle('Target Variable', size = 20, weight='bold')

song_popularity = df['song_popularity'].map({0:'UnPopular', 1:'Popular'})

a = sns.countplot(data = df, x =song_popularity,palette=theme)
plt.tick_params(axis="x", colors=theme[0],labelsize=15)

for p in a.patches:
    width = p.get_width()
    height = p.get_height()
    x, y = p.get_xy()
    a.annotate(f'{height/df.shape[0]*100} %', (x + width/2, y + height*1.02), ha='center')

plt.show()
```




```
cont = ['song_duration_ms', 'acousticness', 'danceability', 'energy',
       'instrumentalness', 'liveness', 'loudness',
       'speechiness', 'tempo', 'audio_valence']
cat = [ 'key', 'audio_mode', 'time_signature']
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

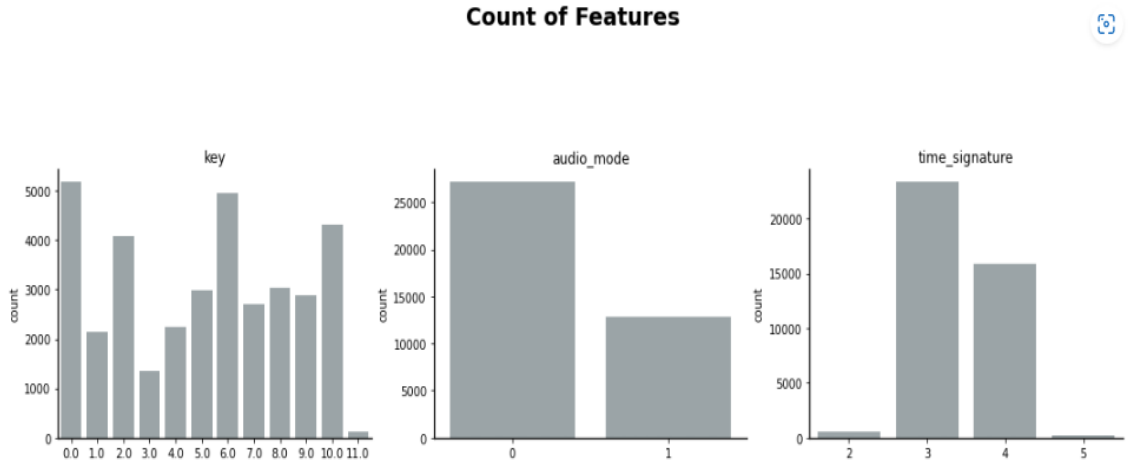
```
a = 4  # number of rows
b = 3  # number of columns
c = 1  # initialize plot counter

plt.figure(figsize= (18,18))

for i in cont:
        plt.suptitle('Distribution of Features', size = 20, weight='bold')
        plt.subplot(a, b, c)
        A=sns.kdeplot(data= df, x=i,hue=song_popularity,palette=theme[:-
2], linewidth = 1.3,shade=True, alpha=0.35)
        plt.title(i)
        plt.xlabel(" ")
        c = c + 1
```




```
a = 4  # number of rows
b = 3  # number of columns
c = 1  # initialize plot counter

plt.figure(figsize= (18,18))
```

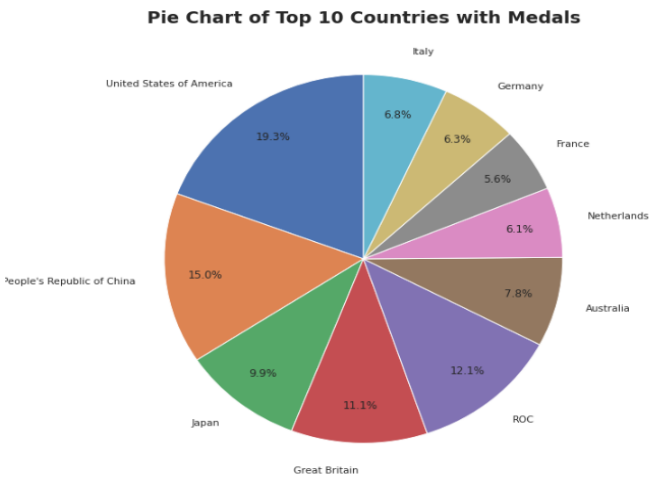

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
for i in cat:

        plt.suptitle('Count of Features', size = 20, weight='bold')

        plt.subplot(a, b, c)
        A=sns.countplot(df[i],color=theme[3], alpha=0.5)
        plt.title(i)
        plt.xlabel(" ")
        plt.tick_params(axis="x", colors='black',labelsize=10)
        c = c + 1
```

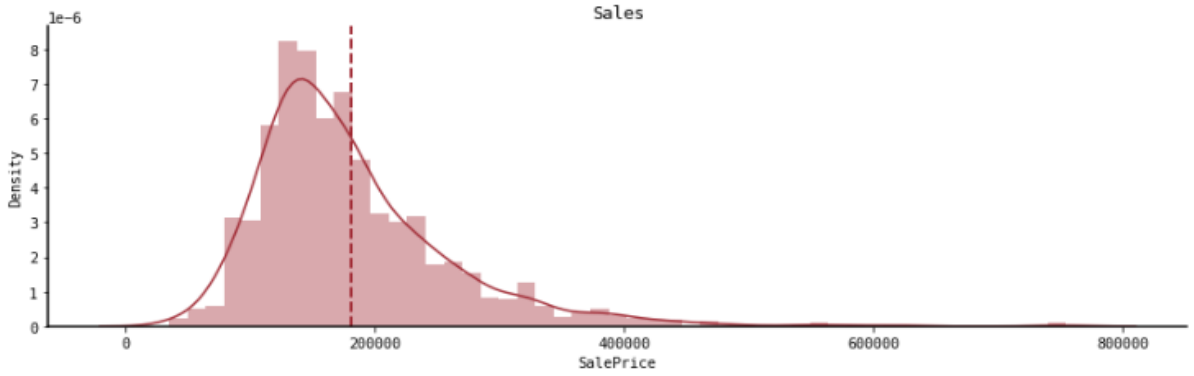



```
figure = plt.figure(figsize=(30,10))

A = plt.pie(medals['Total'][:10],
        labels=medals['Country'][:10],
        startangle=90,
        labeldistance=1.15,
        pctdistance=0.8,
        autopct='%1.1f%%')
plt.title("Pie Chart of Top 10 Countries with Medals",size=20,weight='bold')
plt.show();
```

Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/



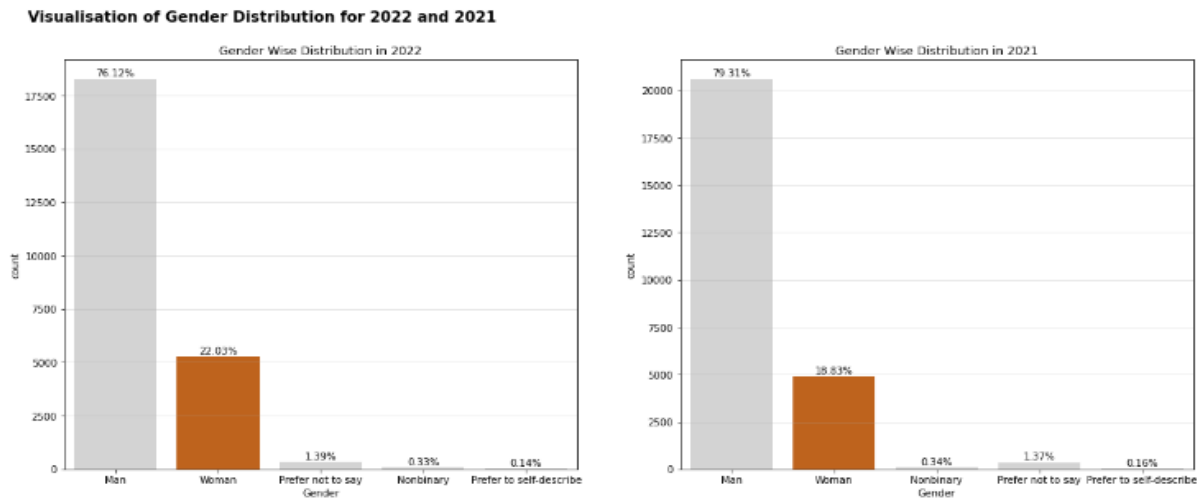

```
#checking the target variables for distribution
sns.distplot(house['SalePrice'],color=colors[7])
plt.axvline(x=house['SalePrice'].mean(), color=colors[7], linestyle='--', li
newidth=2)
plt.title('Sales');
```




```
l = df_current['Q3'].value_counts(normalize=True).mul(100).tolist()[1]-df
_old['Q2'].value_counts(normalize=True).mul(100).values.tolist()[1]

print(5*'\n',"\033[1;32m Increase in Woman is only\033[1;32m",round(l,
2),'%\033[1;32m Over Last Year\033[1;32m',5*'\n')
```

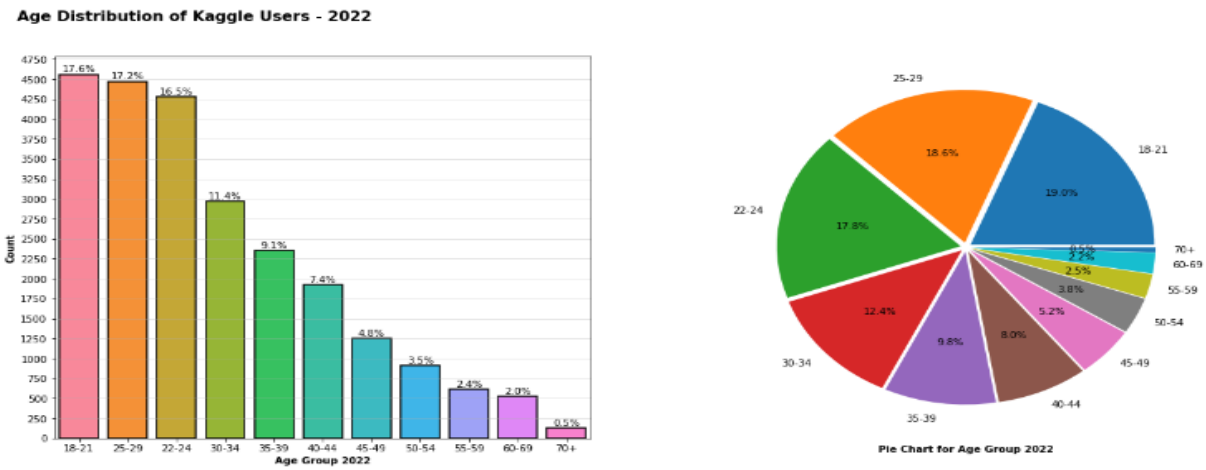
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig, ax = plt.subplots(1, 2, figsize=(20,8))
fig.text(0.1, 0.95, "Visualisation of Gender Distribution for 2022 and 20
21", fontsize=15, fontweight='bold')

sns.countplot(x='Q3', data=df_current,palette="Dark2", ax=ax[0]); #Cur
rent Year
sns.countplot(x='Q2', data=df_old,palette="Dark2",ax=ax[1]); #Last Yea
r

for i, ax in enumerate(ax.flatten()):
    ax.grid(axis='y', linestyle='-', alpha=0.4)
    if i==0:t=shape;year = 2022
    else:t=shape_21;year =2021
    for p in ax.patches:
        percentage = f'{100 * p.get_height() / t:.2f}%\n'
        ax.annotate(percentage, (p.get_x() + p.get_width() / 2,p.get_height
()), ha='center', va='center')
        ax.set_xlabel('Gender');ax.set_title("Gender Wise Distribution in "+
str(year))
        if not(0.5 <= p.get_x() < 1.5):
            p.set_facecolor('lightgrey')

plt.show()
```



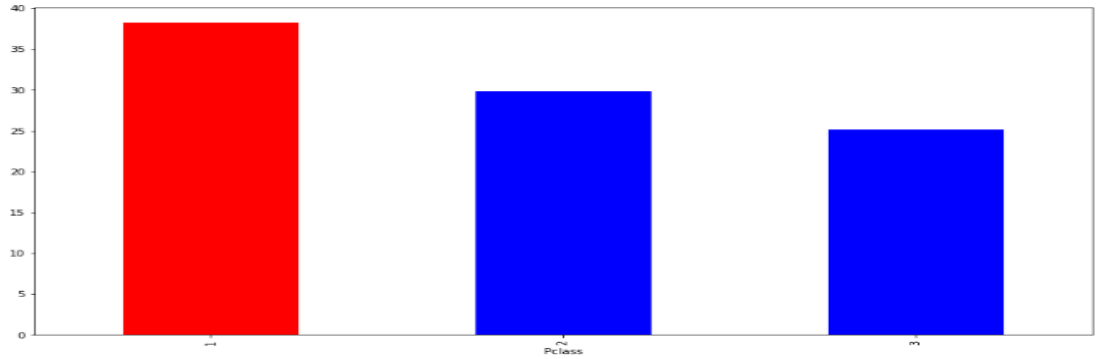
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig, ax = plt.subplots(1,2, figsize=(20,8))

fig.text(0.1, 0.95, "Age Distribution of Kaggle Users - 2022", fontsize=15, fontweight='bold')

sns.barplot(x=df_current['Q2'].value_counts().index, y=df_current['Q2'].value_counts().values, ax=ax[0],
            edgecolor='black', linewidth=1.5, saturation=1.5)
ax[0].yaxis.set_major_locator(MaxNLocator(nbins=20));ax[0].grid(axis='y', linestyle='-', alpha=0.4)
ax[0].set_ylabel('Count', weight='semibold')
ax[0].set_xlabel('Age Group 2022', weight='semibold')
ax[1].set_xlabel('Pie Chart for Age Group 2022', weight='semibold')
for p in ax[0].patches:
        percentage = f'{100 * p.get_height() / t:.1f}%\n'
        ax[0].annotate(percentage, (p.get_x() + p.get_width() / 2,p.get_height()), ha='center', va='center')

ax[1].pie(df_current['Q2'].value_counts(), labels = df_current['Q2'].value_counts().index,  autopct='%1.1f%%',
        explode=[0.03 for i in df_current['Q2'].value_counts().index])

plt.show()
```

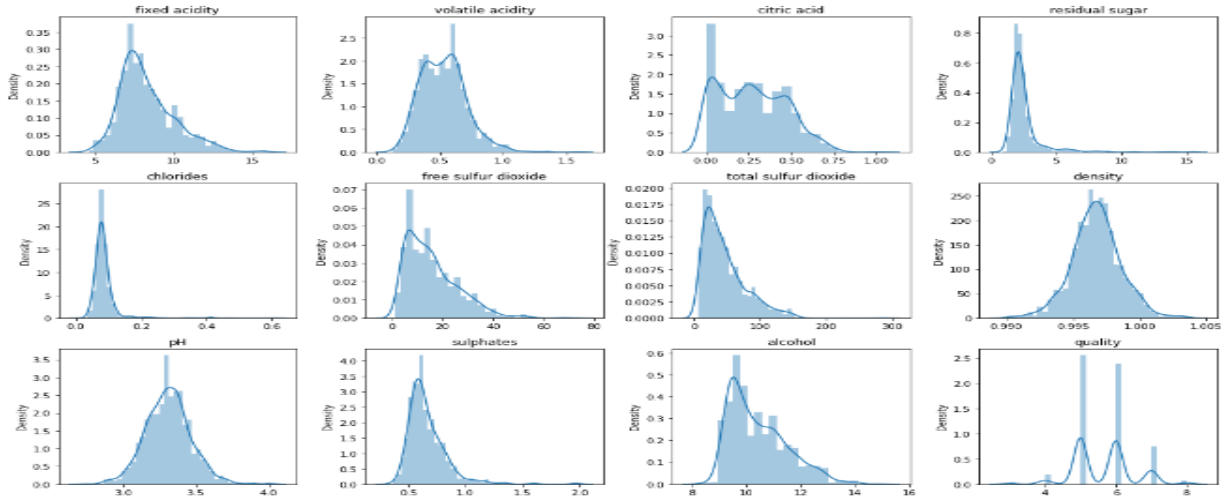


Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
df2=titanic.groupby('Pclass')['Age'].mean().sort_values(ascending=False)
plt.figure(figsize = (15,8))
color = ['b' if i < 30 else 'r' for i in df2]
df2.plot.bar(color=color);
```



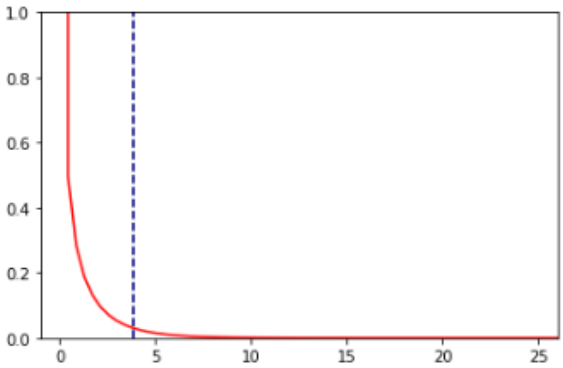

```
col=['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality']

fig = plt.figure(figsize=(15,10))

for i in range(len(col)):
    plt.subplot(3,4,i+1)
    plt.title(col[i])
    sns.distplot(df,x=df[col[i]])
plt.tight_layout()
plt.show()
```



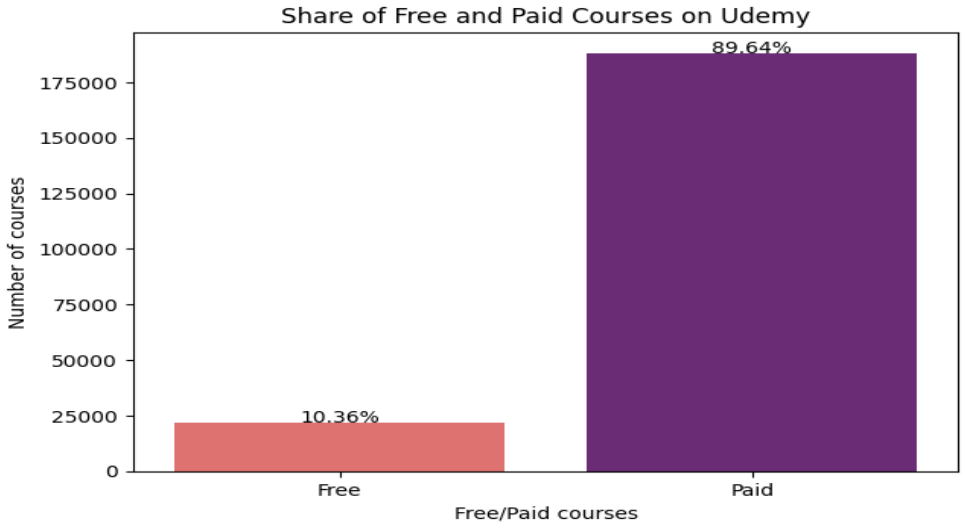
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
fig, ax = plt.subplots(1, 1)

plt.xlim(-1,26)
plt.ylim(0,1)
x = np.linspace(f.ppf(0.0000000001, dfn, dfd),f.ppf(0.9999999999, dfn, d
fd), 100)
ax.plot(x, f.pdf(x, dfn, dfd), 'r-')
ax.axvline(f.ppf(0.95, dfn, dfd), ls = "--", color = "navy")
print('upper 5%:', f.ppf(0.95, dfn, dfd))
```

```
upper 5%: 3.8426563592313365
```



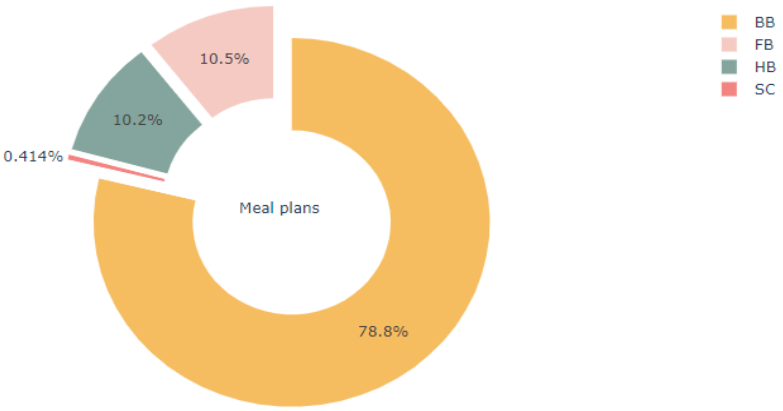

```
# Free or Paid Courses - Countplot
fig, ax = plt.subplots(figsize=(7,5), dpi=100)
ax = sns.countplot(data=courses, x='is_paid', palette='magma_r')
ax.set_xticklabels(labels=['Free', 'Paid'])
ax.set_xlabel("Free/Paid courses")
ax.set_ylabel("Number of courses")
ax.set_title("Share of Free and Paid Courses on Udemy")
percentage = round(courses['is_paid'].value_counts() * 100 /len(course
s), 2)
patches = ax.patches
for i in range(len(patches)):
  x = patches[i].get_x() + patches[i].get_width()/2
  y = patches[i].get_height()+.05
  ax.annotate('{:.2f}%'.format(percentage[i]), (x, y), ha='center')
```




```
#Creating a stripplot to visualize differences in data distribution between hotels

features = ['lead_time', 'stays_in_weekend_nights', 'stays_in_week_nights', 'adults', 'children', 'babies', 'previous_cancellations', 'previous_bookings_not_canceled', 'booking_changes', 'adr', 'days_in_waiting_list']

n = 1

sns.set_style('darkgrid')
sns.set(font_scale = 1.2)
plt.figure(figsize = (14, 18))

for feature in features:
    plt.subplot(4,3,n)
    sns.stripplot(x = df['hotel'], y = df[feature], palette = 'summer').set(xlabel = None, ylabel = None)
    plt.title(f'{feature} strip plot')
    n = n + 1
    plt.tight_layout()
```

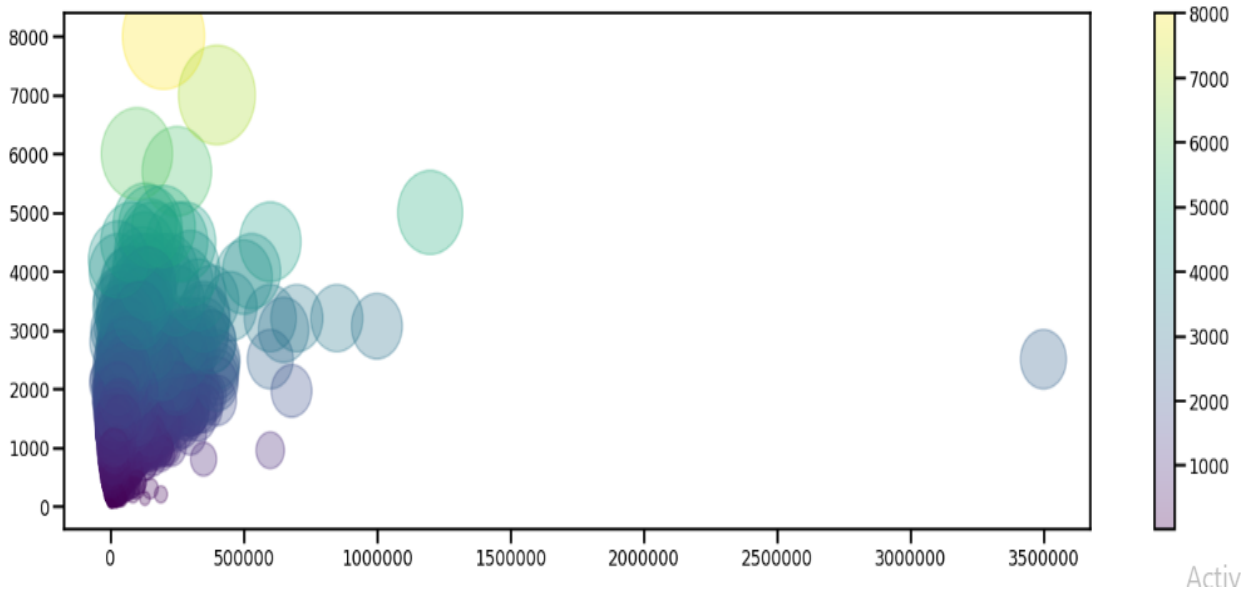
Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/


```
import plotly.graph_objects as go

labels = confirmed_bookings['meal'].unique()
values = confirmed_bookings['meal'].value_counts()
palette = ["#f6bd60", "#f5cac3", "#84a59d", "#f28482"]

fig = go.Figure(data=[go.Pie(labels = labels,
                             values = values,
                             hole=.5,
                             title = 'Meal plans',
                             legendgroup = True,
                             pull = [0.1, 0.1, 0.1, 0.1]
                            )
                      ]
                )

fig.update_traces(marker = dict(colors = palette));
```



Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

```
x = rent_df["Rent"]
y = rent_df["Size"]
colors = rent_df["Size"]
sizes = rent_df["Size"]

plt.figure(figsize = (25, 8))
plt.ticklabel_format(style = 'plain')
plt.scatter(x, y, c = colors, s = sizes, alpha = 0.3, cmap = 'viridis')
plt.colorbar();
```

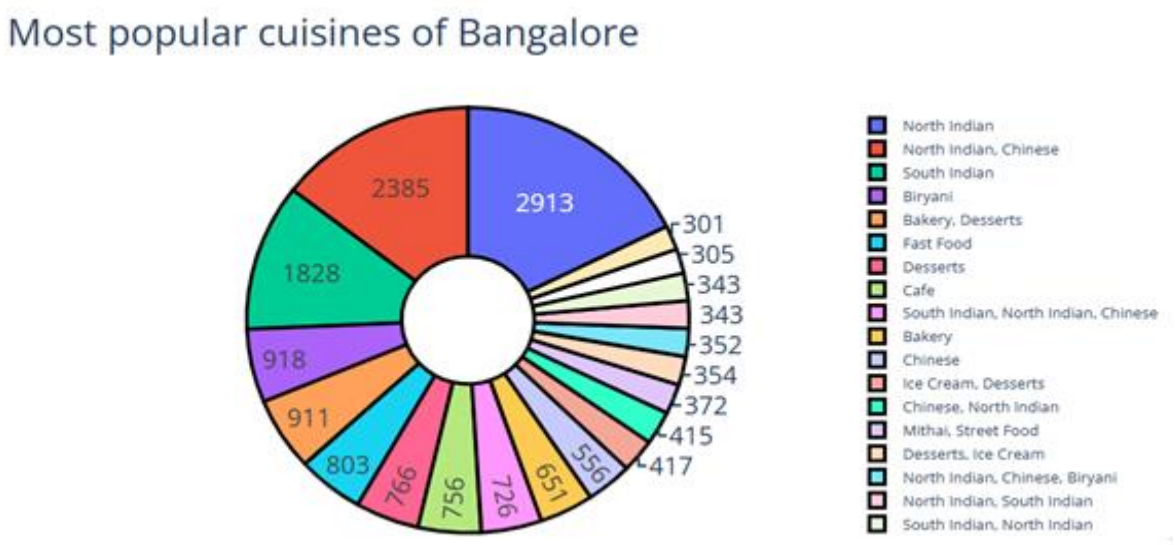


Syed Afroz Ali
Follow me on LinkedIn for more: https://www.linkedin.com/in/syed-afroz-70939914/

Mastering Data Visualization Techniques (Part 5)

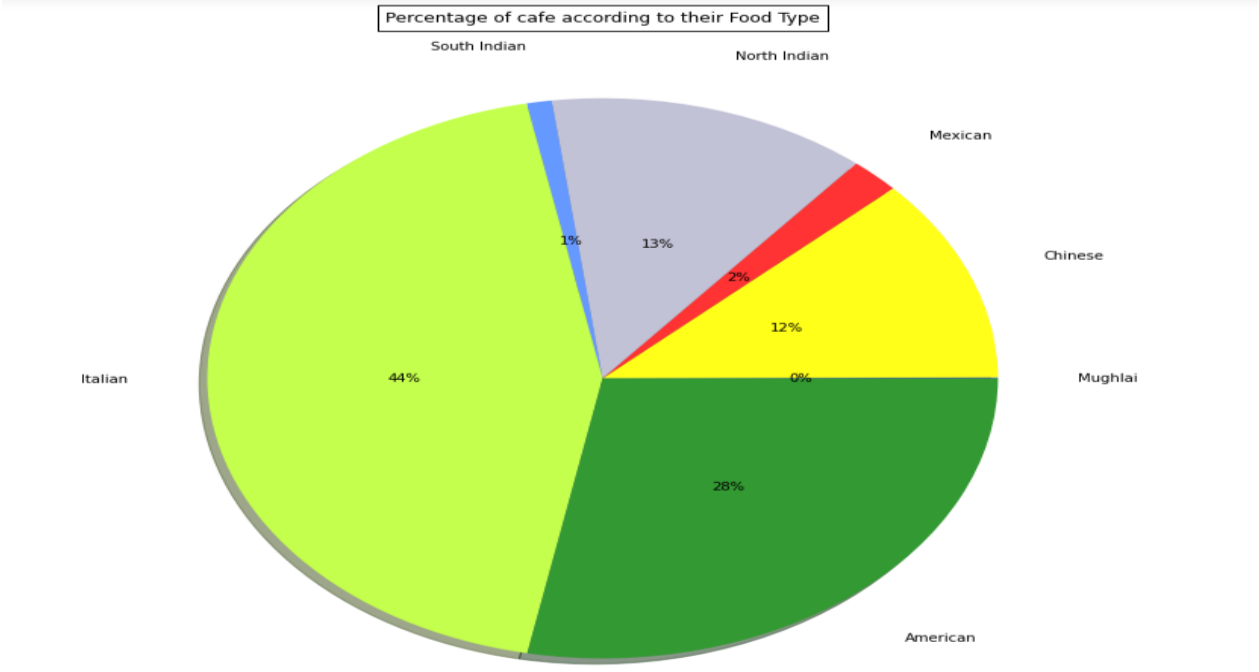
Prepared by: Syed Afroz Ali

```
import plotly.graph_objs as go
values = data['cuisines'].value_counts()[:20]
labels=values.index
text=values.index
fig =
go.Figure(data=[go.Pie(values=values,labels=labels,hole=.3)]
)
fig.update_traces(hoverinfo='label+percent', textinfo='value',
textfont_size=20,
                  marker=dict(line=dict(color='#000000', width=3)))
fig.update_layout(title="Most popular cuisines of Bangalore
",
                  titlefont={'size': 30},
                  )
fig.show()
```



Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

```
MughlaiFoodcafe = data[data['cuisines'].str.contains('Mughlai',
case=False, regex=True,na=False)]
MughlaiFoodcafe.head()

#pie chart showing % of various Food serving Type cafe
slices=[MughlaiFoodcafe.shape[0],
        ChineseFoodcafe.shape[0],
        MexicanFoodcafe.shape[0],
        NorthIndianFoodcafe.shape[0],
        SouthIndianFoodcafe.shape[0],
        ItalianFoodcafe.shape[0],
        AmericanFoodcafe.shape[0]]
labels=['Mughlai','Chinese','Mexican','North Indian','South
Indian','Italian','American']
colors = ['#3333cc','#ffff1a','#ff3333','#c2c2d6','#6699ff','#c4ff4d','#339933']
plt.pie(slices,colors=colors, labels=labels, autopct='%1.0f%%',
pctdistance=.5, labeldistance=1.2,shadow=True)
fig = plt.gcf()
plt.title("Percentage of cafe according to their Food Type",
bbox={'facecolor':'1', 'pad':5})
fig.set_size_inches(12,12)
plt.show()
```



Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

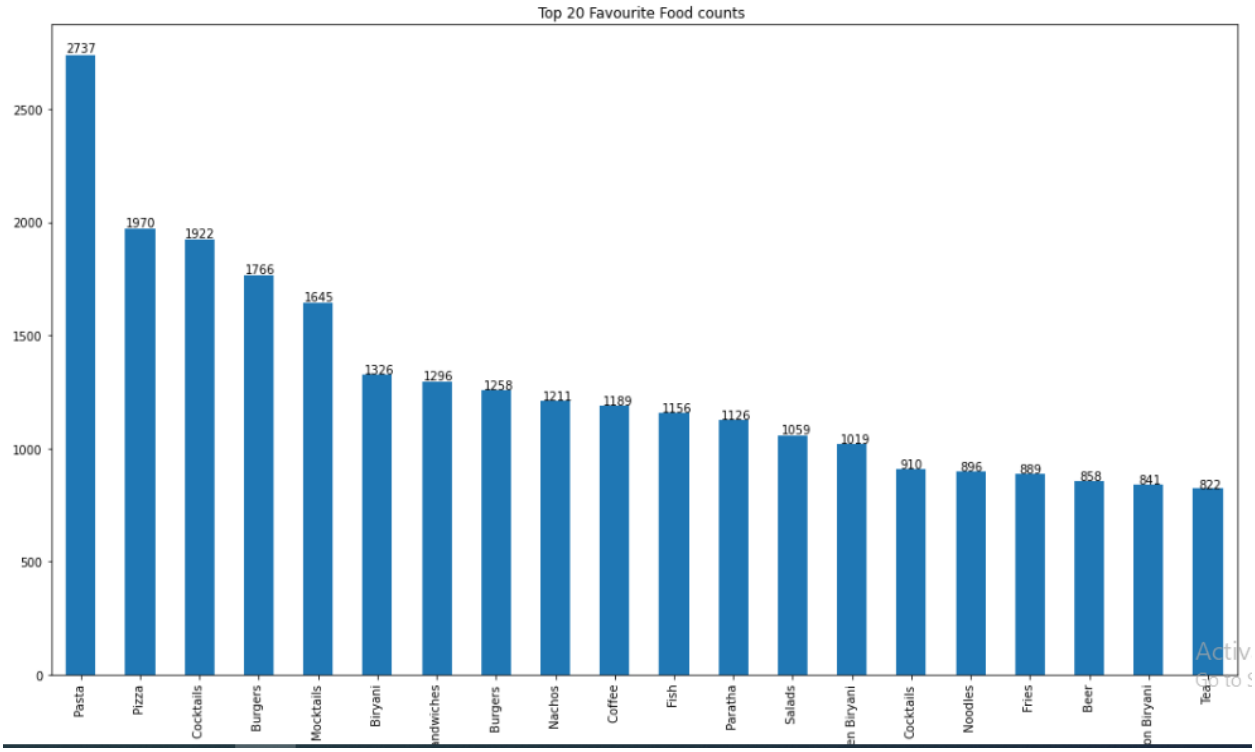
```
# Most Liked Dishes in Bangalore
import re
data=data[data['dish_liked'].notnull()]
data.index=range(data.shape[0])
likes=[]
for i in range(data.shape[0]):
    splited_array=re.split(',',data['dish_liked'][i])
    for item in splited_array:
        likes.append(item)

print("Count of Most liked dishes of Bangalore")
favourite_food = pd.Series(likes).value_counts()
favourite_food.head(20)
```

```
Count of Most liked dishes of Bangalore
 Pasta                2737
 Pizza                1970
 Cocktails            1922
 Burgers              1766
 Mocktails            1645
 Biryani              1326
 Sandwiches           1296
Burgers               1258
 Nachos               1211
 Coffee               1189
 Fish                 1156
 Paratha              1126
 Salads               1059
 Chicken Biryani      1019
Cocktails              910
 Noodles               896
 Fries                 889
 Beer                  858
 Mutton Biryani        841
 Tea                   822
```

```
ax = favourite_food.nlargest(n=20,
keep='first').plot(kind='bar',figsize=(15,15),title = 'Top 20
Favourite Food counts ')
for p in ax.patches:

   ax.annotate(str(p.get_height()), (p.get_x() * 1.005,
p.get_height() * 1.005))
```

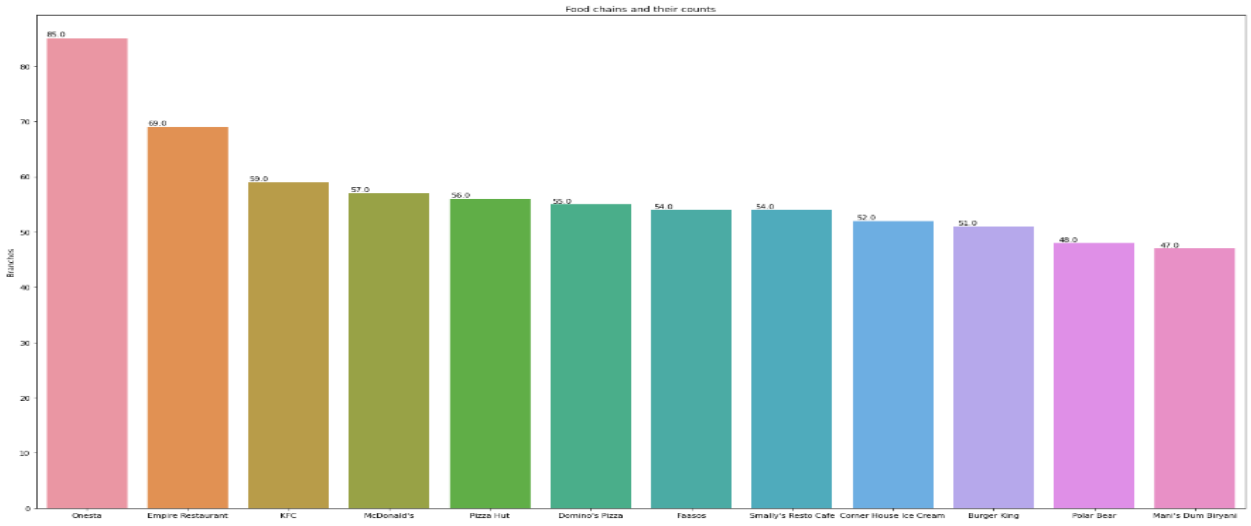
Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625




```
#Analysis of biggest food chains
branches =
data.groupby(['name']).size().to_frame('count').reset_index().
sort_values(['count'],ascending=False)
ax = sns.barplot(x='name', y='count', data=branches[:12])
plt.xlabel('')
plt.ylabel('Branches')
plt.title('Food chains and their counts')
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() * 1.005,
p.get_height() * 1.005))

fig = plt.gcf()
fig.set_size_inches(25,15)
```

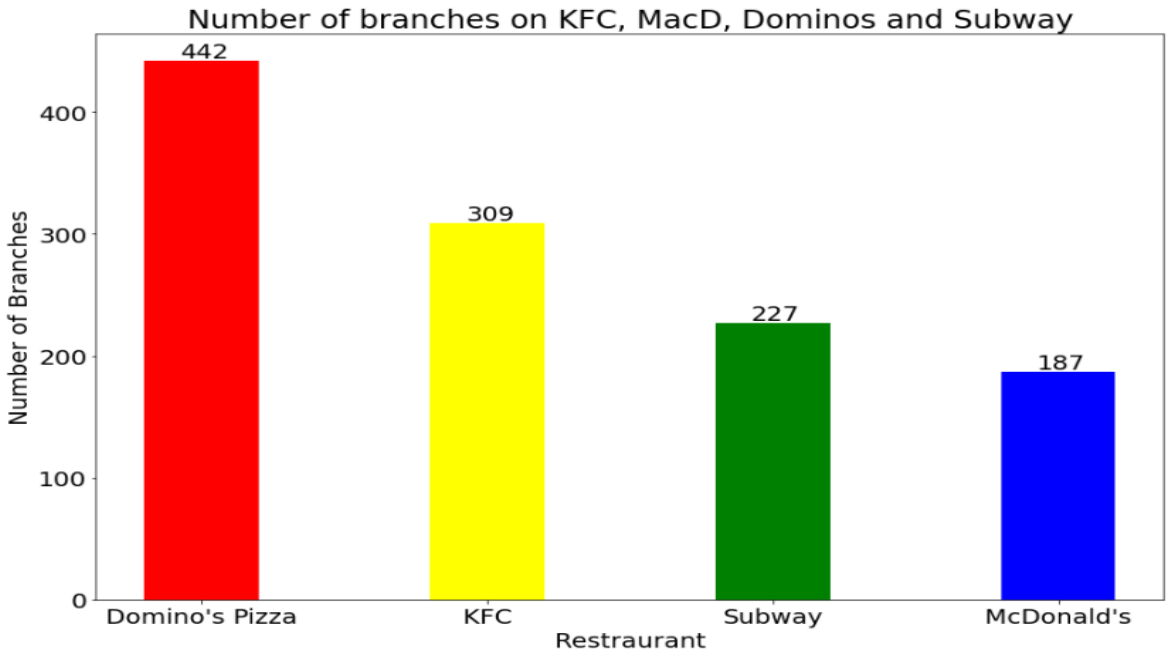
Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu



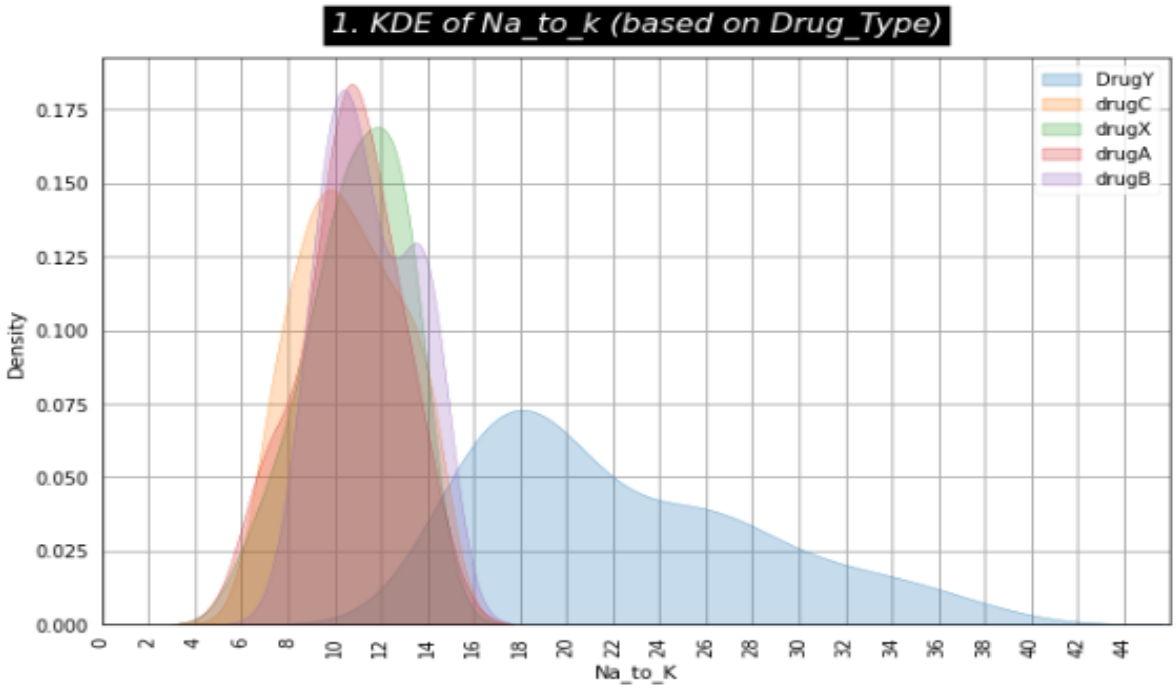

```
data = df.groupby('name').size()[["Domino's Pizza", "KFC",
"McDonald's",'Subway']].sort_values(ascending = False)

x    = data.index
y    = data.values

plt.figure(figsize = (15,10))

color = ['red','yellow','green','blue']
ax = plt.bar(x,y,width = 0.4,color = color)
for i in ax:
    x_ = i.xy[0] + i.get_width() / 2
    y_ = i.get_height()
    txt = str(y_)
    plt.annotate(
        text       = txt,
        xy         = (x_,y_),
        xytext     = (-17,2.9),
        textcoords = 'offset points'
    )
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

```
plt.xlabel('Restraurant')
plt.ylabel('Number of Branches')
plt.title('Number of branches on KFC, MacD, Dominos and Subway')
plt.show()
```

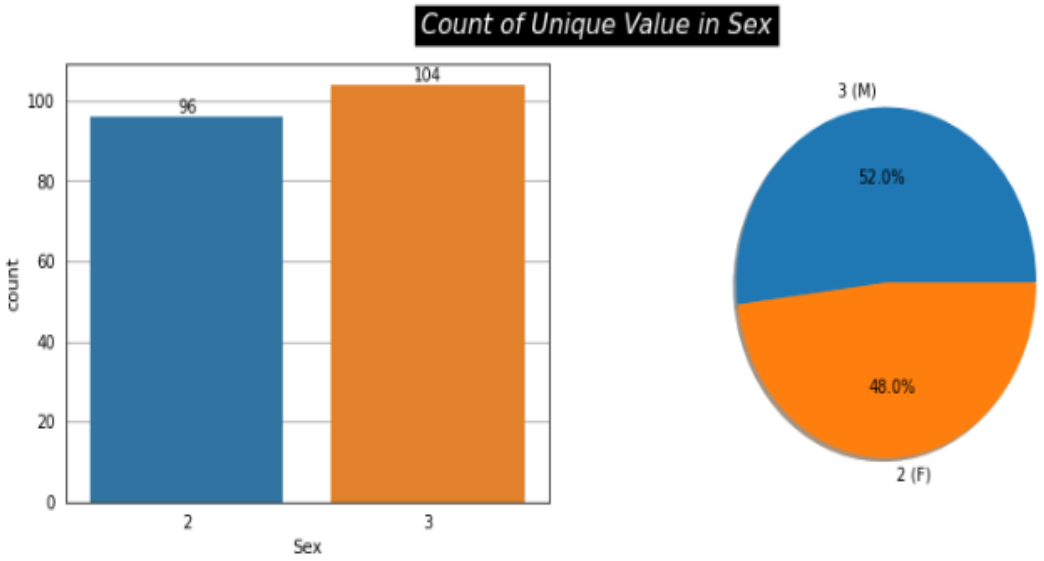


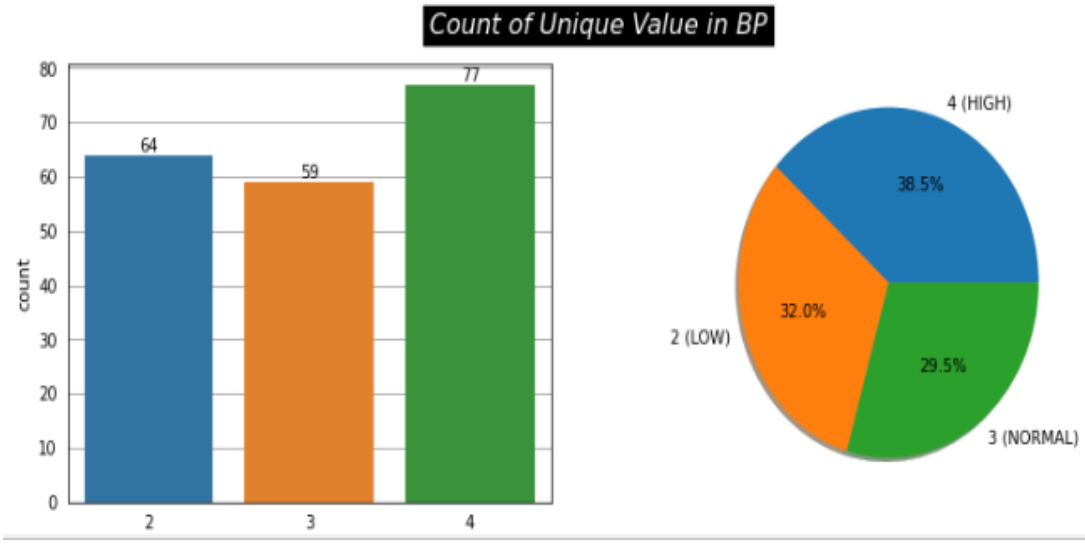

```
plt.style.use('seaborn-notebook')
for i, label in enumerate(df.Drug_Type.unique().tolist()):
    sns.kdeplot(df2.loc[df2['Drug_Type'] == i+1, 'Na_to_K'], label=label, shade=True)

plt.title('1. KDE of Na_to_k (based on Drug_Type)', fontdict=font, pad=15)
plt.xticks(np.arange(0,46,2), rotation=90)
plt.xlim([0,46])
plt.legend()
plt.show()
```




```
# draw countplot and pie plot of categorical data
for col in categorical:
    fig, axes = plt.subplots(1,2,figsize=(10,4))
    # count of col (countplot)
    sns.countplot(data=df2, x=col, ax=axes[0])
    for container in axes[0].containers:
        axes[0].bar_label(container)
    # count of col (pie chart)
    slices = df2[col].value_counts().values
    activities = [f"{i} ({var})" for i, var in
zip(df2[col].value_counts().index,
df[col].value_counts().index)]
    axes[1].pie(slices, labels=activities, shadow=True,
autopct='%1.1f%%')
    plt.suptitle(f'Count of Unique Value in {col}', y=1.09,
**font)
    plt.show()
```

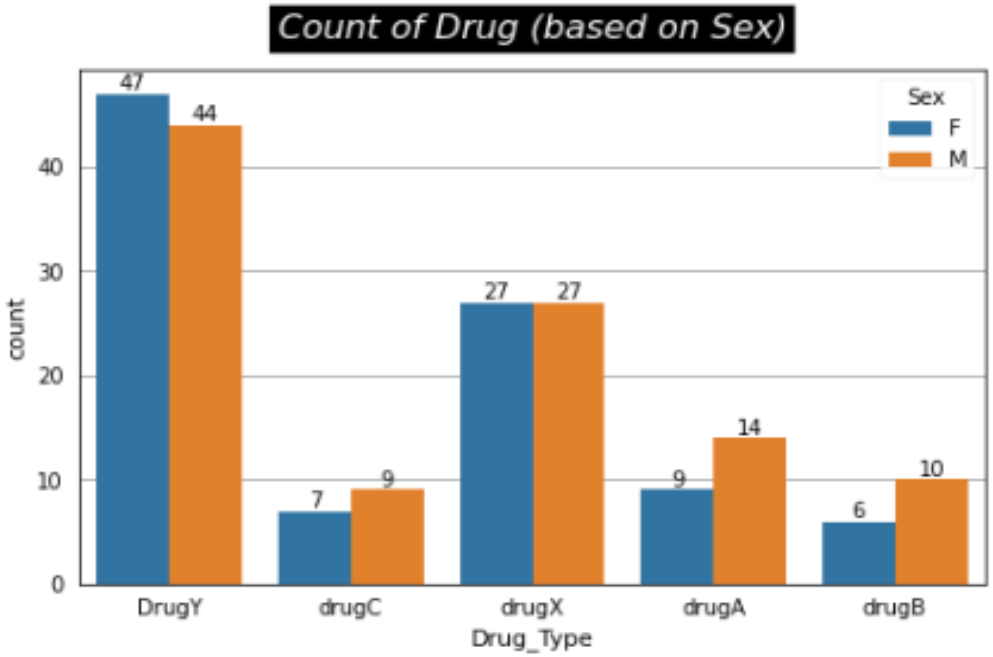
Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu






```
for col in ['Sex','BP','Cholesterol']:
    ax = sns.countplot(data=df, x='Drug_Type', hue=col)
    for container in ax.containers:
        ax.bar_label(container)
    plt.title(f'Count of Drug (based on {col})', fontdict=font,
pad=15)
    plt.show()
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

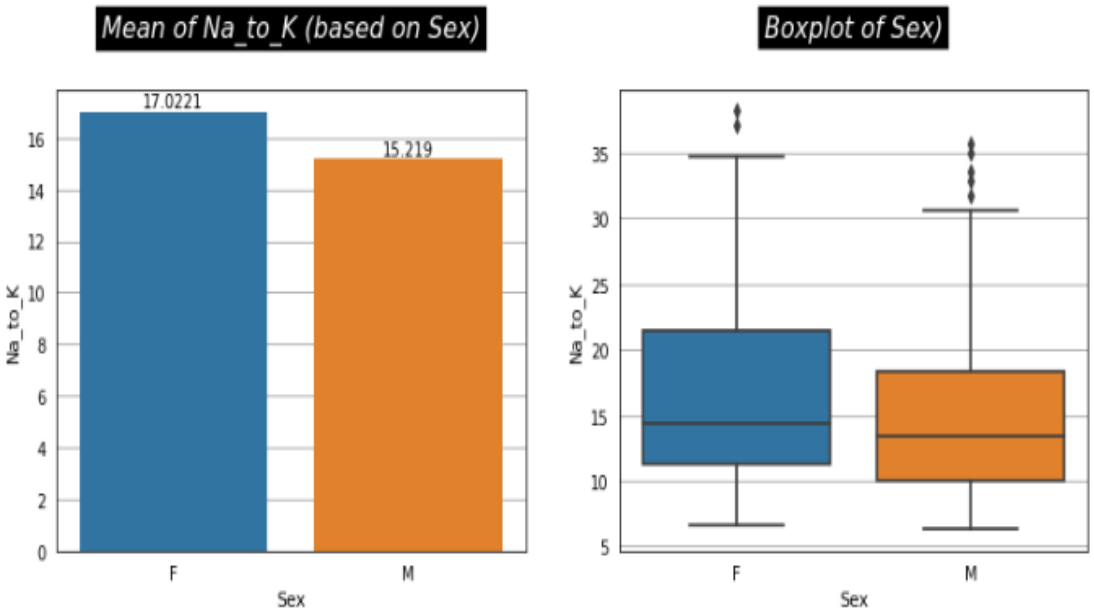



```
for col in ['Sex', 'BP', 'Cholesterol']:
    fig , ax= plt.subplots(1,2, figsize=(10,4))
    gp =
df.groupby([col])['Na_to_K'].mean().to_frame().reset_index()
    sns.barplot(data=gp, x=col, y='Na_to_K', ax=ax[0])
    for container in ax[0].containers:
        ax[0].bar_label(container)
    ax[0].set_title(f'Mean of Na_to_K (based on {col})', y=1.09,
**font)
    sns.boxplot(data=df, x=col, y='Na_to_K', ax=ax[1])
    ax[1].set_title(f'Boxplot of {col})', y=1.09, **font)
```

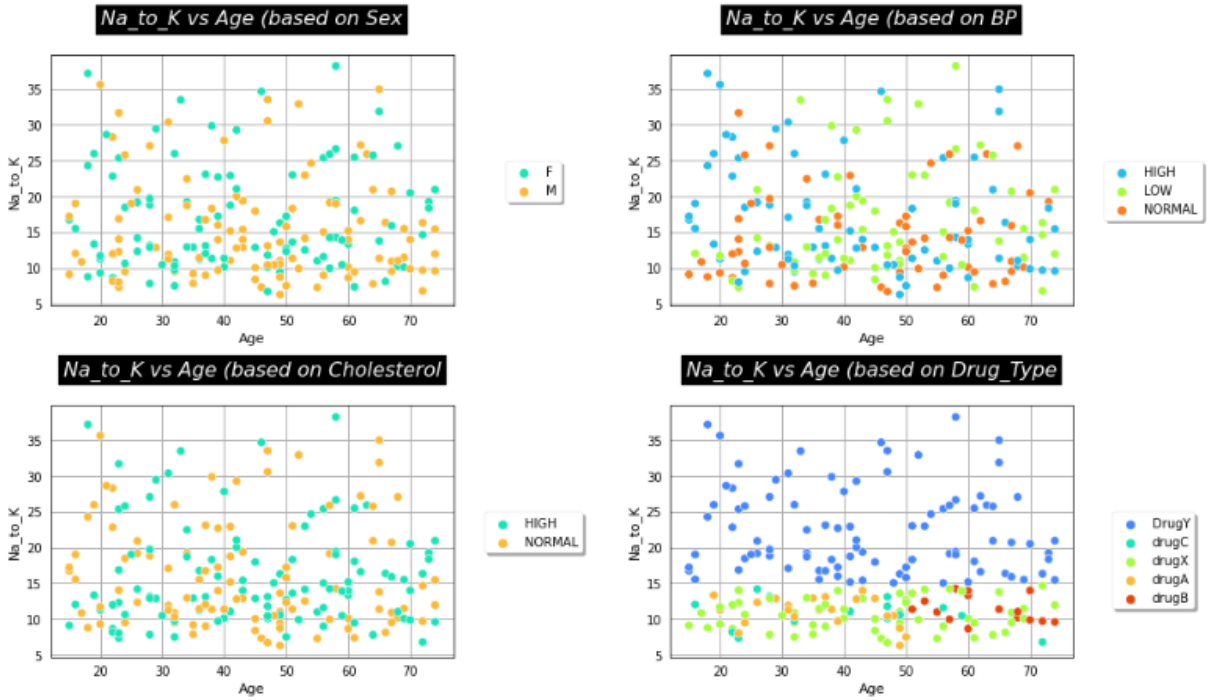
Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

```
    plt.show()
```



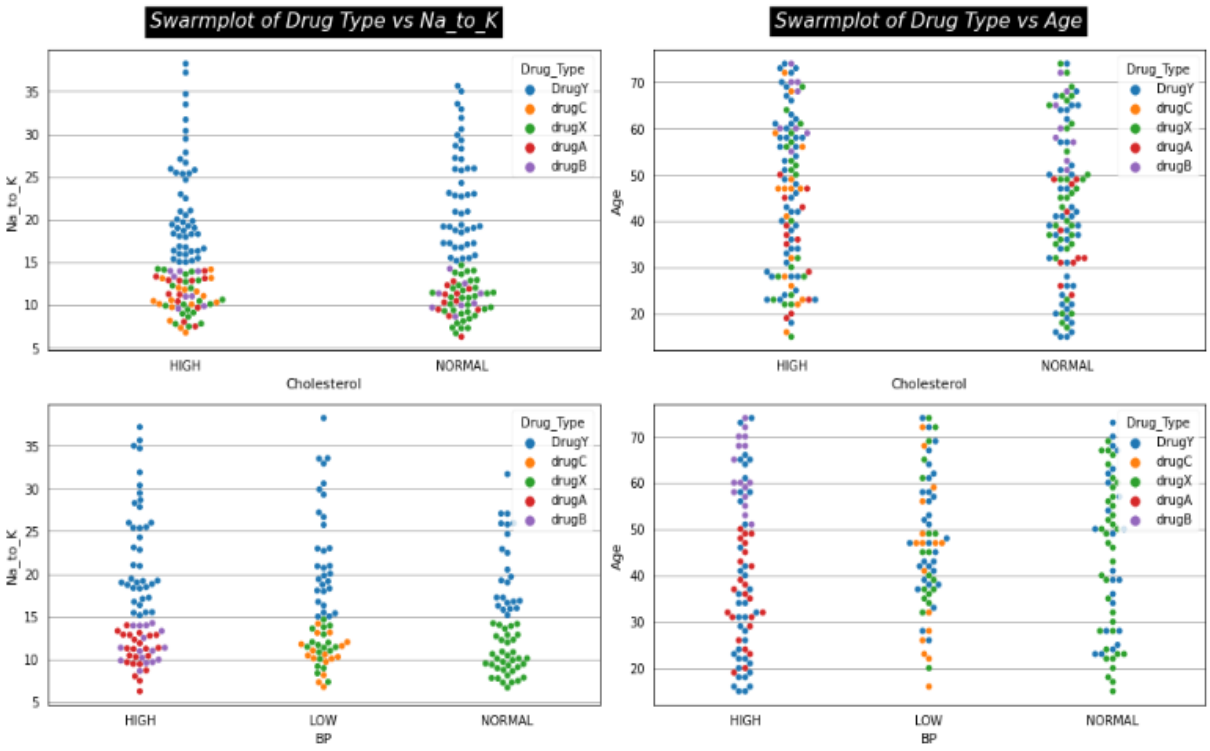

```
fig, ax = plt.subplots(2,2,figsize=(14,8))
for i, col in enumerate(['Sex', 'BP', 'Cholesterol',
'Drug_Type']):
    sns.scatterplot(data=df, x='Age', y='Na_to_K', hue=col,
ax=ax[i//2, i%2], palette='turbo')
    ax[i//2, i%2].set_title(f'Na_to_K vs Age (based on {col}',
y=1.09, **font)
    ax[i//2, i%2].legend(loc='upper center',
bbox_to_anchor=(1.2, 0.6),
        fancybox=True, shadow=True)

fig.tight_layout()
plt.show()
```




```
fig, ax = plt.subplots(3,2,figsize=(14,12))
sns.swarmplot(data=df, x='Cholesterol', y='Na_to_K',
hue='Drug_Type', ax=ax[0,0])
sns.swarmplot(data=df, x='Cholesterol', y='Age',
hue='Drug_Type', ax=ax[0,1])
sns.swarmplot(data=df, x='BP', y='Na_to_K',
hue='Drug_Type', ax=ax[1,0])
sns.swarmplot(data=df, x='BP', y='Age', hue='Drug_Type',
ax=ax[1,1])
sns.swarmplot(data=df, x='Sex', y='Na_to_K',
hue='Drug_Type', ax=ax[2,0])
sns.swarmplot(data=df, x='Sex', y='Age', hue='Drug_Type',
ax=ax[2,1])
ax[0,0].set_title('Swarmplot of Drug Type vs Na_to_K',y=1.05,
**font)
ax[0,1].set_title('Swarmplot of Drug Type vs Age',y=1.05,
**font)
plt.tight_layout()
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

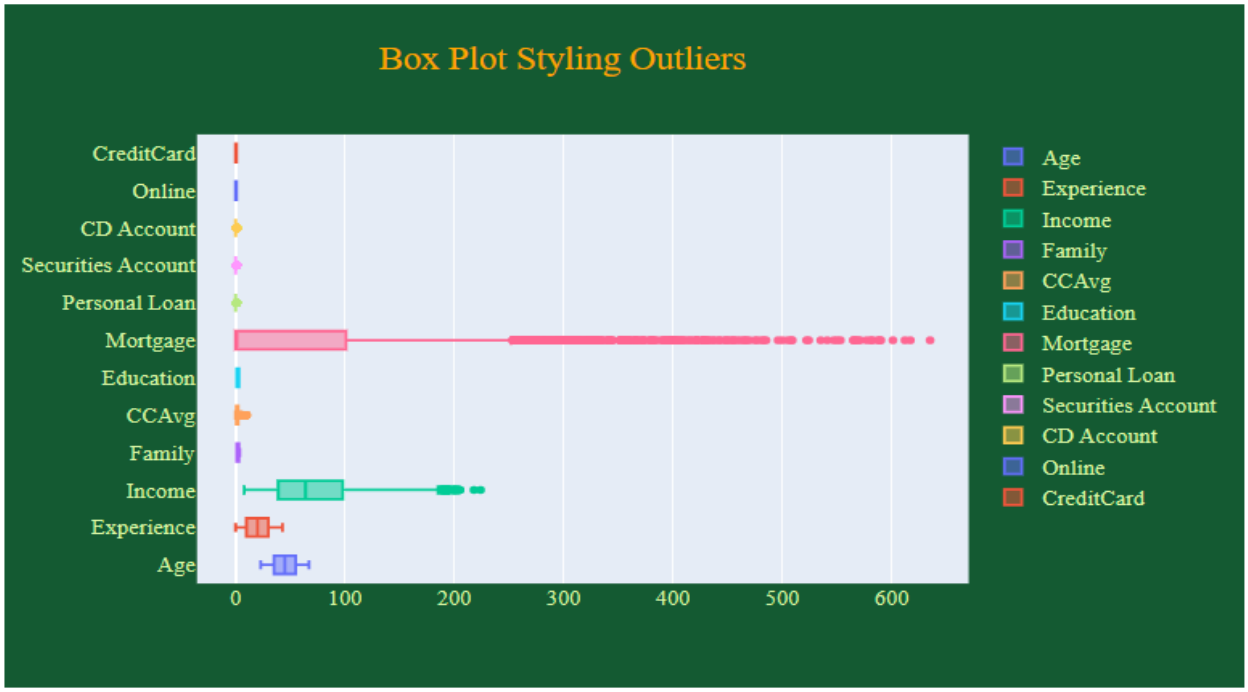
Learn Data Visualization With Python
https://t.me/AIMLDeepThaught/625




```
import itertools
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
from plotly.subplots import make_subplots
# for solve problem of show plotly plots
from plotly.offline import init_notebook_mode
init_notebook_mode(connected=True)
plt.style.use('_mpl-gallery')
FONT = {'fontsize':20, 'fontstyle':'normal', 'fontfamily':'Times New Roman', 'backgroundcolor':'#145A32', 'color':'orange'} # for plot title

fig = go.Figure()
for col in df:
    fig.add_trace(go.Box(x=df[col], name=col))
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

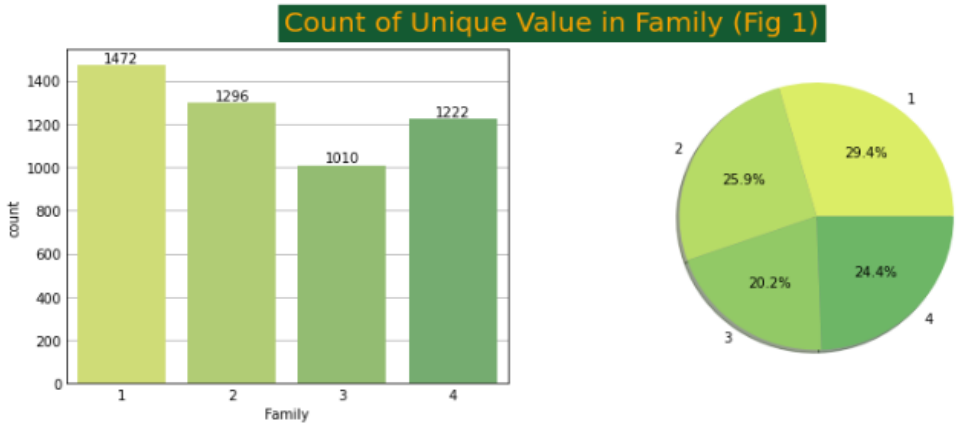
```
fig.update_layout(
    title_text="Box Plot Styling Outliers",
    title_font=dict(color='orange', family='newtimeroman',
size=25),
    title_x=0.45,
    paper_bgcolor='#145A32',
    # plot_bgcolor='#DAF7A6',
    font=dict(color='#DAF7A6', family='newtimeroman',
size=16),
    )
fig.show()
```

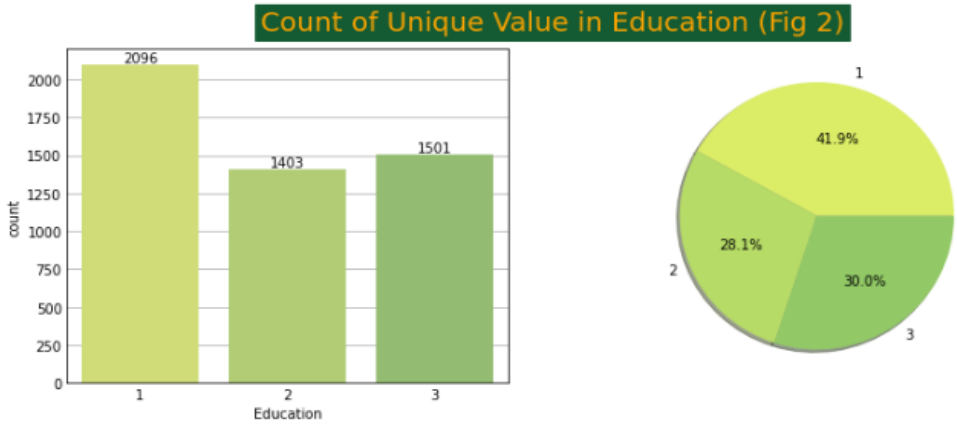



```
# univariate analysis of categorical data:
sns.set_palette("summer_r")
for i, col in enumerate(discrete_cols1):

    fig, axes = plt.subplots(1,2,figsize=(10,4))
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

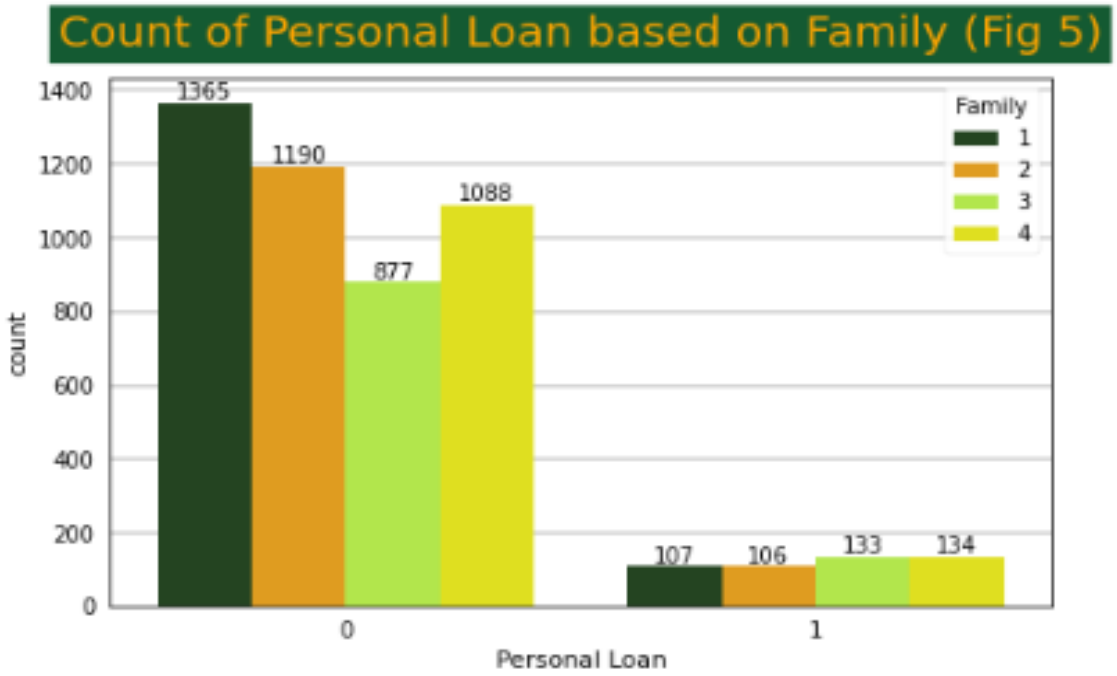
```
    # count of col (countplot)
    sns.countplot(data=df, x=col, ax=axes[0])
    for container in axes[0].containers:
        axes[0].bar_label(container)
    # count of col (pie chart)
    slices = df[col].value_counts().sort_index().values
    activities = [var for var in
df[col].value_counts().sort_index().index]
    axes[1].pie(slices, labels=activities, shadow=True,
autopct='%1.1f%%')
    plt.suptitle(f'Count of Unique Value in {col} (Fig {i+1})',
y=1.09, **FONT)
    plt.show()
```





Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

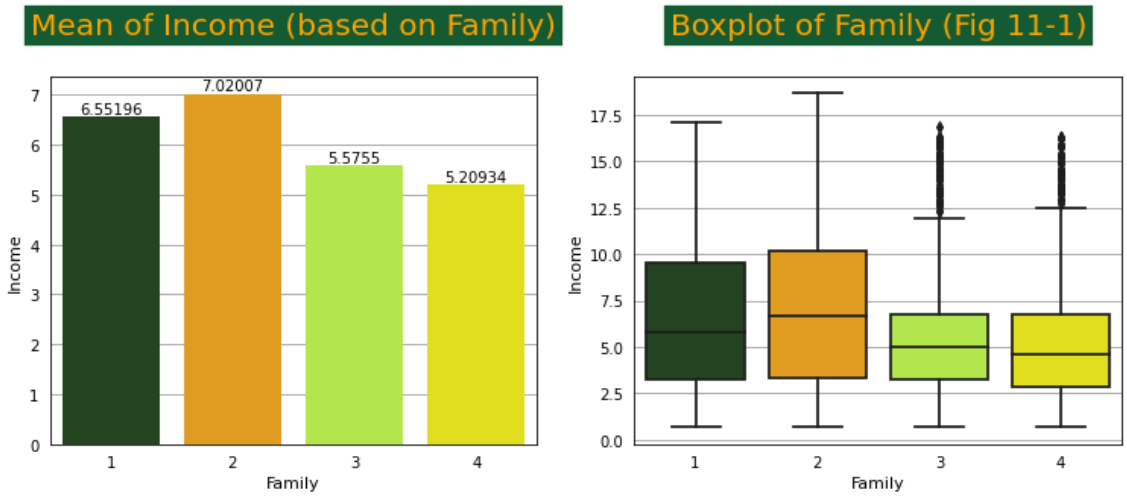
```
sns.set_palette(['#1f4a1b','orange','#bbff33','yellow'])
discrete_cols2 = ['Family', 'Education', 'Securities Account',
'CD Account', 'Online', 'CreditCard']
for i, col in enumerate(discrete_cols2):
    ax = sns.countplot(data=df, x='Personal Loan', hue=col)
    for container in ax.containers:
        ax.bar_label(container)
    plt.title(f'Count of Personal Loan based on {col} (Fig {i+5})',
fontdict=FONT, pad=15)
    plt.show()
```




```
for i, col in enumerate(['Income', 'CCAvg','Mortgage']):
    print('='*30, f"Mean of {col} in each categorical feature",
'='*30)
    for j, cat in enumerate(discrete_cols2):
        fig , ax= plt.subplots(1,2, figsize=(10,4))
        gp =
df.groupby([cat])[col].mean().to_frame().reset_index()
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

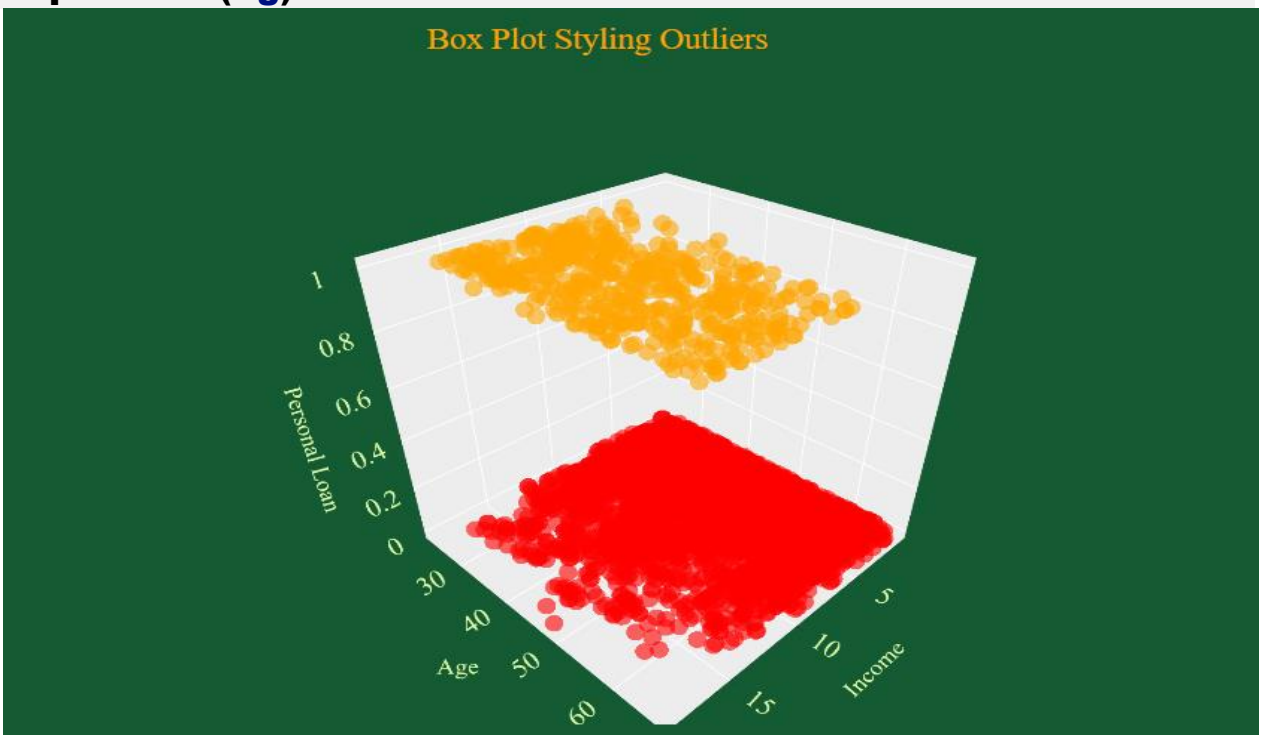
```
        sns.barplot(data=gp, x=cat, y=col, ax=ax[0])
        for container in ax[0].containers:
            ax[0].bar_label(container)
        ax[0].set_title(f'Mean of {col} (based on {cat})', y=1.09,
**FONT)

        sns.boxplot(data=df, x=cat, y=col, ax=ax[1])
        ax[1].set_title(f'Boxplot of {cat} (Fig {i+11}-{j+1})',
y=1.09, **FONT)

        plt.show()
```




```
continuous_cols = ['Age','Experience','CCAvg','Mortgage']

for i, col in enumerate(continuous_cols):
    fig = px.scatter_3d(
        data_frame= df,
        x=df.Income,
        y=df[col],
        z=df['Personal Loan'],
        color=df['Personal Loan'].astype(str),
        color_discrete_map={'1':'orange', '0':'red'},
        template='ggplot2',
```

Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu

```
        hover_name='Age',
        # hover_data=
        opacity=0.6,
        # symbol='Transmission',
        # symbol_map=
        # log_x=True,
        # log_z=True,
        height=700,
        title=f'3D scatter of features based on Personal Loan (Fig
{i+1})')
    fig.update_layout(
        title_text="Box Plot Styling Outliers",
        title_font=dict(color='orange', family='newtimeroman',
size=25),
        title_x=0.45,
        paper_bgcolor='#145A32',
        # plot_bgcolor='#DAF7A6',
        font=dict(color='#DAF7A6', family='newtimeroman', size=16),
    )
    pio.show(fig)
```



Syed Afroz Ali (Kaggle Grandmaster)
Follow me on LinkedIn for more: https://lnkd.in/g3W7w_Xu