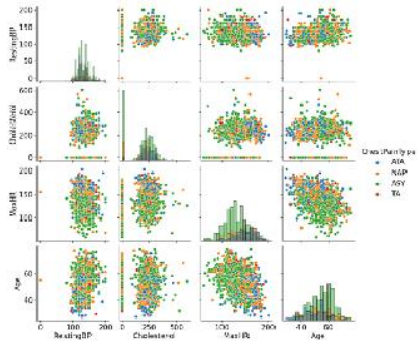
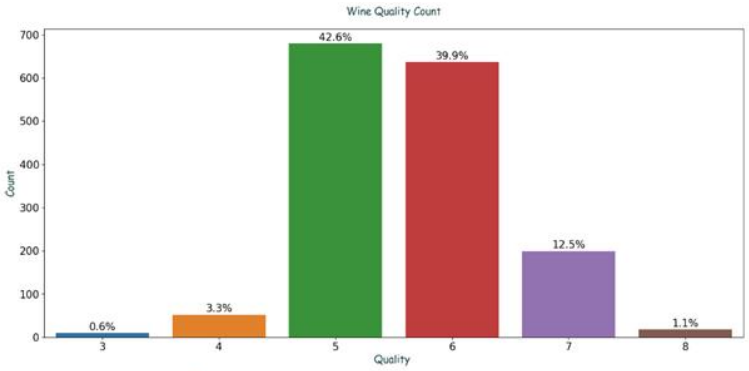
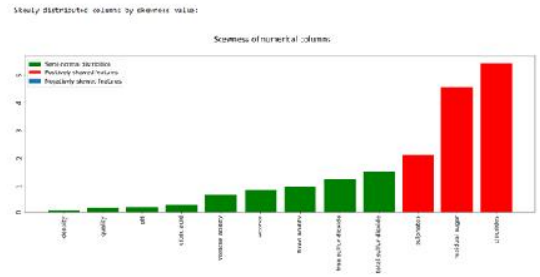
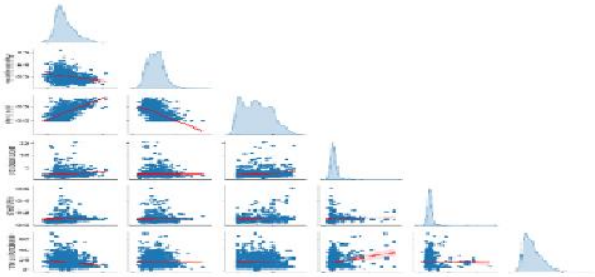
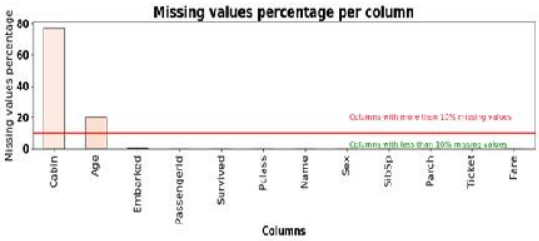
Machine Learning Visualization: Part 4





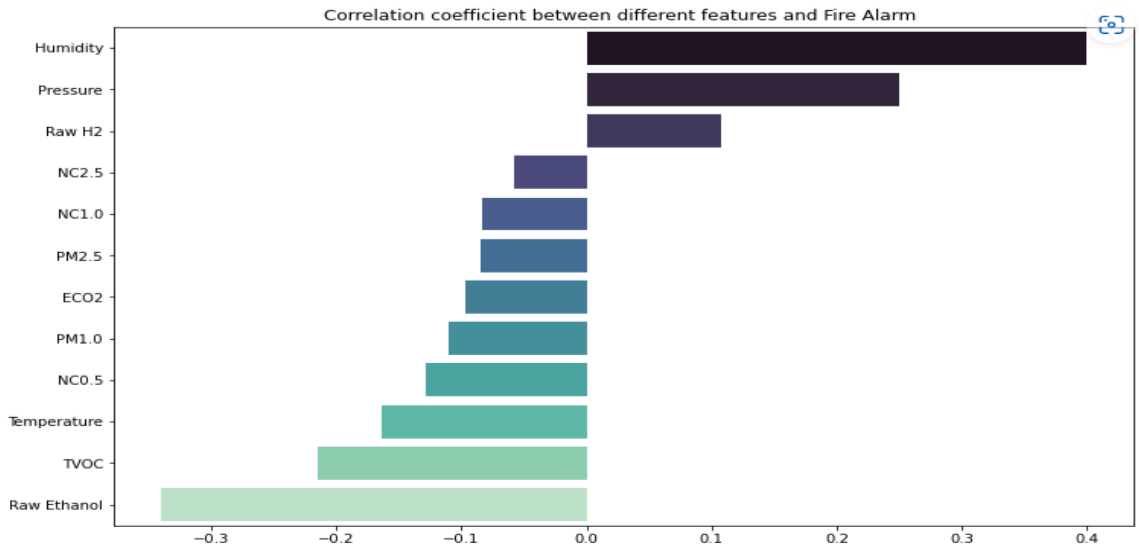




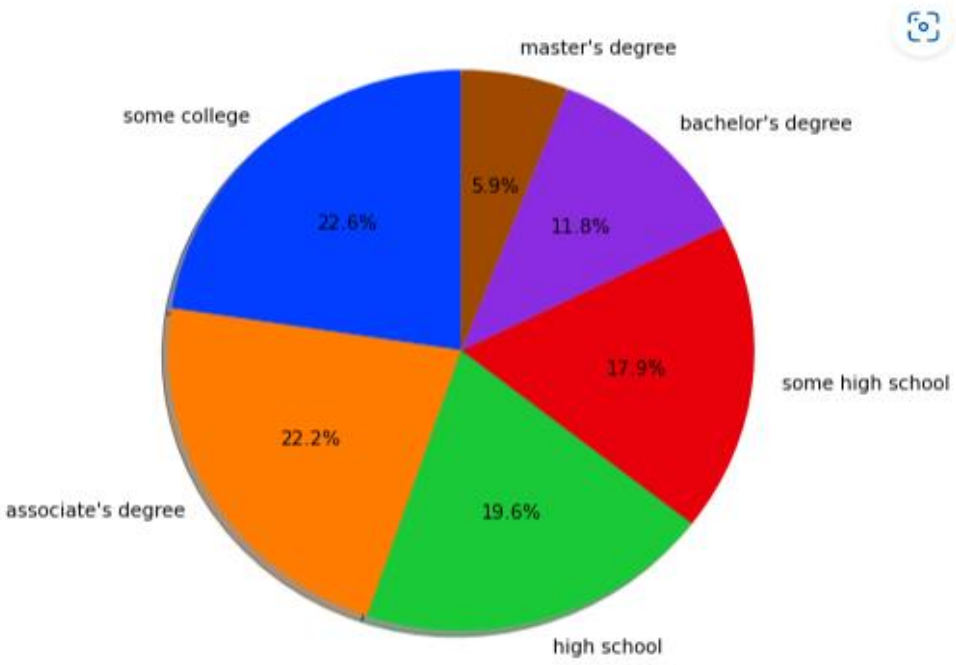



```
plt.figure(figsize=(12,8))
data_4 = data.corr()["Fire Alarm"].sort_values(ascending=False)
indices = data_4.index
labels = []
corr = []
for i in range(1, len(indices)):
    labels.append(indices[i])
    corr.append(data_4[i])
sns.barplot(x=corr, y=labels, palette='mako')
plt.title('Correlation coefficient between different features and Fire Alarm ')
plt.show()
```

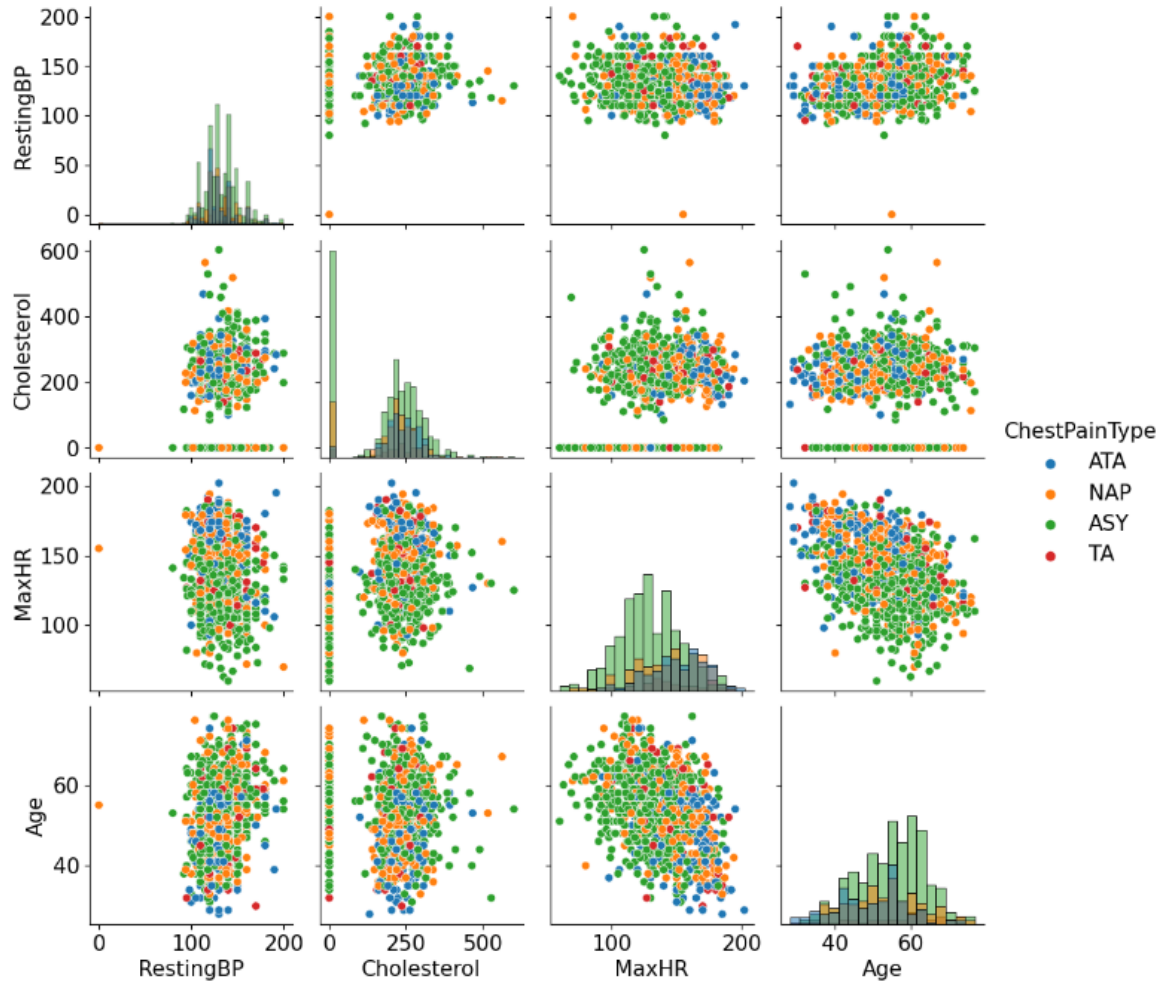
Syed Afroz Ali



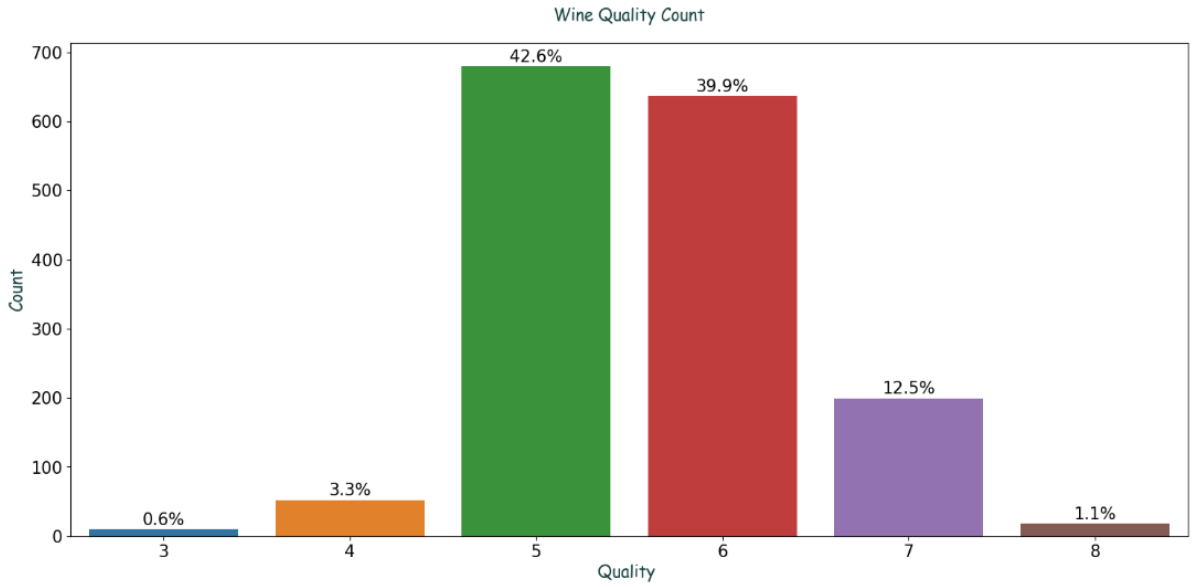

```
education=df['parental level of education'].value_counts()

sns.set_palette('bright')
plt.figure(figsize=(10,7))
labels=education.index
sizes=education.values
plt.pie(sizes,labels=labels,autopct='%1.1f%%',
        shadow=True,startangle=90)
plt.show()
```



```
import matplotlib
matplotlib.rcParams.update({'font.size': 15})
plt.figure(figsize=(18,9))
cols_out = ["RestingBP", "Cholesterol", "MaxHR", "Age",'ChestPainType']
sns.pairplot(heart[cols_out], hue="ChestPainType", diag_kind="hist",
palette="tab10") # tab10
plt.show();
```

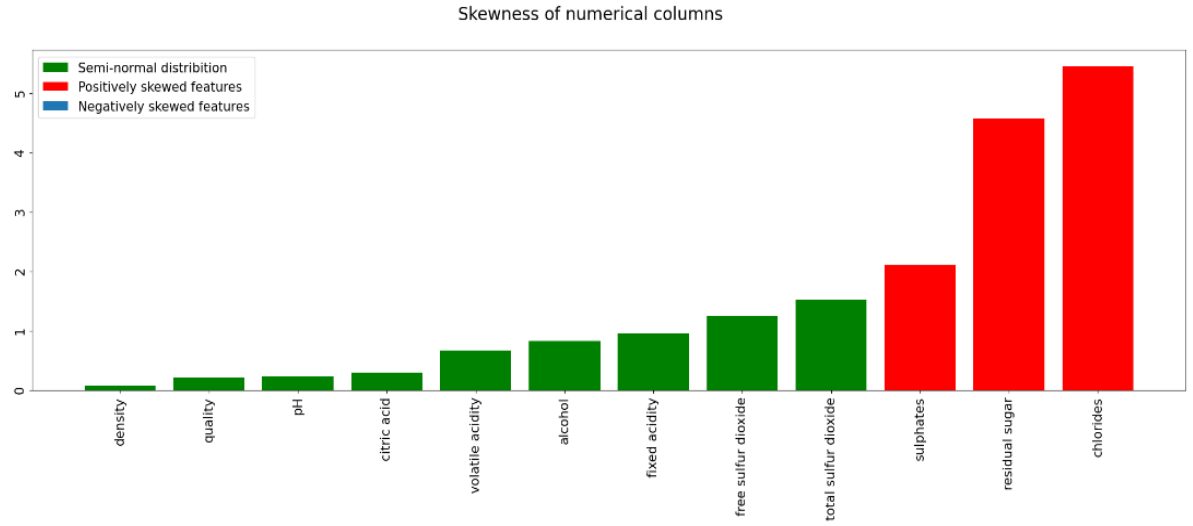



```
fig, ax = plt.subplots(figsize = (18,8))
sns.countplot(x= wine["quality"])
plt.title("Wine Quality Count",fontsize=20,color='#1a4441',font='Comic
Sans Ms',pad=20)
plt.xlabel("Quality ",fontsize=15,color='#1a4441',font='Comic Sans Ms')
plt.ylabel("Count",fontsize=15,color='#1a4441',font='Comic Sans Ms');

total = len(wine)
for p in ax.patches:
    percentage = f'{100 * p.get_height() / total:.1f}%\n'
    x = p.get_x() + p.get_width() / 2
    y = p.get_height()
    ax.annotate(percentage, (x, y), ha='center', va='center')
```



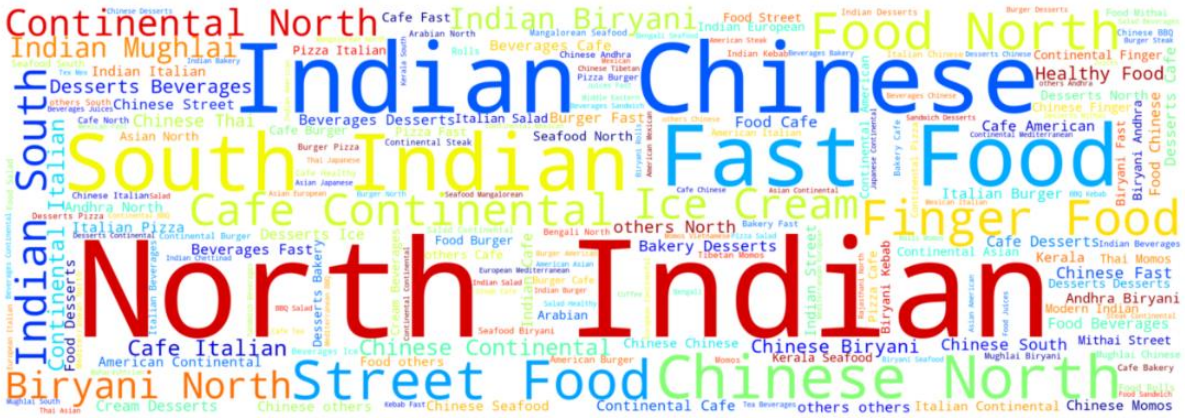

```
print("Skewly distributed columns by skewness value:\n")
skew_df = wine.skew().sort_values()

fig,ax = plt.subplots(figsize=(25,7))
ax.bar(x = skew_df[(skew_df<2)& (skew_df>-2)].index, height =
skew_df[(skew_df<2)& (skew_df>-2)], color = "g", label= "Semi-normal
distribition")
ax.bar(x = skew_df[skew_df>2].index, height = skew_df[skew_df>2], color
= "r", label = "Positively skewed features")
ax.bar(x = skew_df[skew_df<-2].index, height = skew_df[skew_df<-2], color
= "b", label = "Negatively skewed features")
ax.legend()
fig.suptitle("Skewness of numerical columns",fontsize = 20)
ax.tick_params(labelrotation=90);
```

```
Skewly distributed columns by skewness value:
```

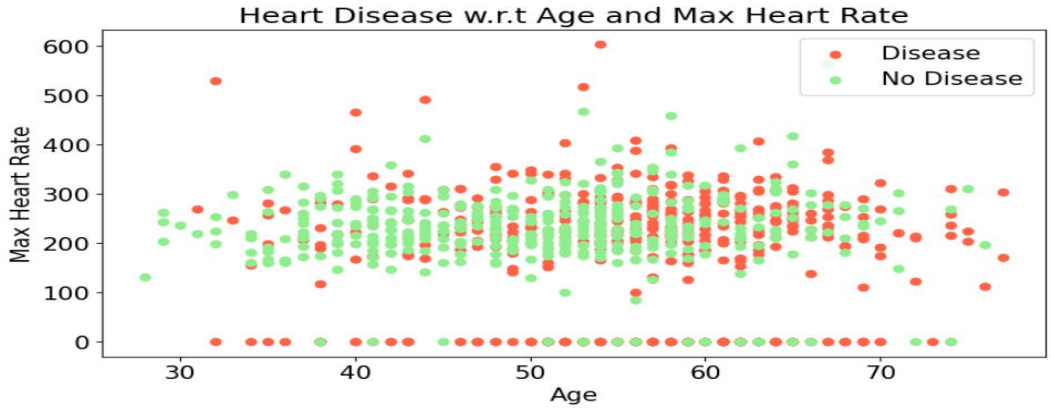


Syed Afroz Ali

```
from wordcloud import WordCloud, STOPWORDS
text = " ".join(Company for Company in df["Cuisines"])
#font = "Quicksand-Bold.ttf"
word_cloud = WordCloud(width = 2300,
                       height = 800,
                       colormap = 'jet',
                       background_color = "white").generate(text)
plt.figure(figsize = (50, 8))
plt.imshow(word_cloud, interpolation = "gaussian")
plt.axis("off")
plt.show()
```

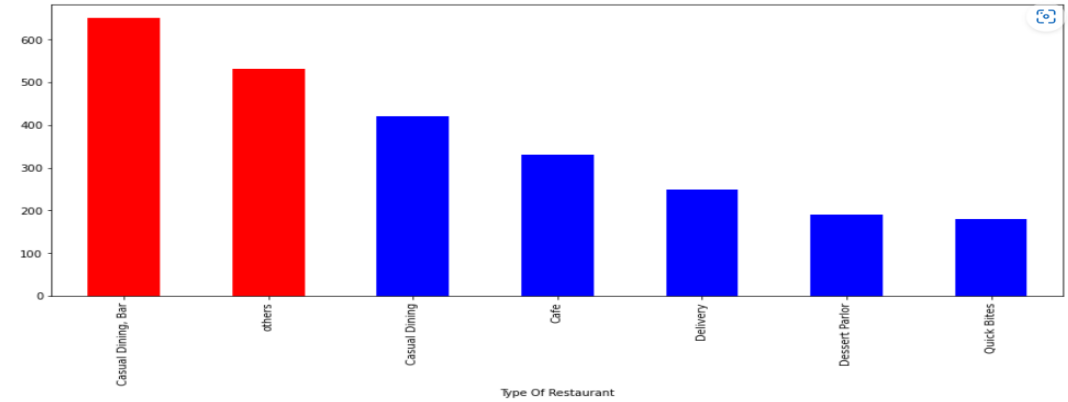



```
plt.figure(figsize=(10,5))
#plotting the values for people who have heart disease
plt.scatter(heart.Age[heart.HeartDisease==1],
            heart.Cholesterol[heart.HeartDisease==1],
            c="tomato")
#plotting the values for people who doesn't have heart disease
plt.scatter(heart.Age[heart.HeartDisease==0],
            heart.Cholesterol[heart.HeartDisease==0],
            c="lightgreen")
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");
```

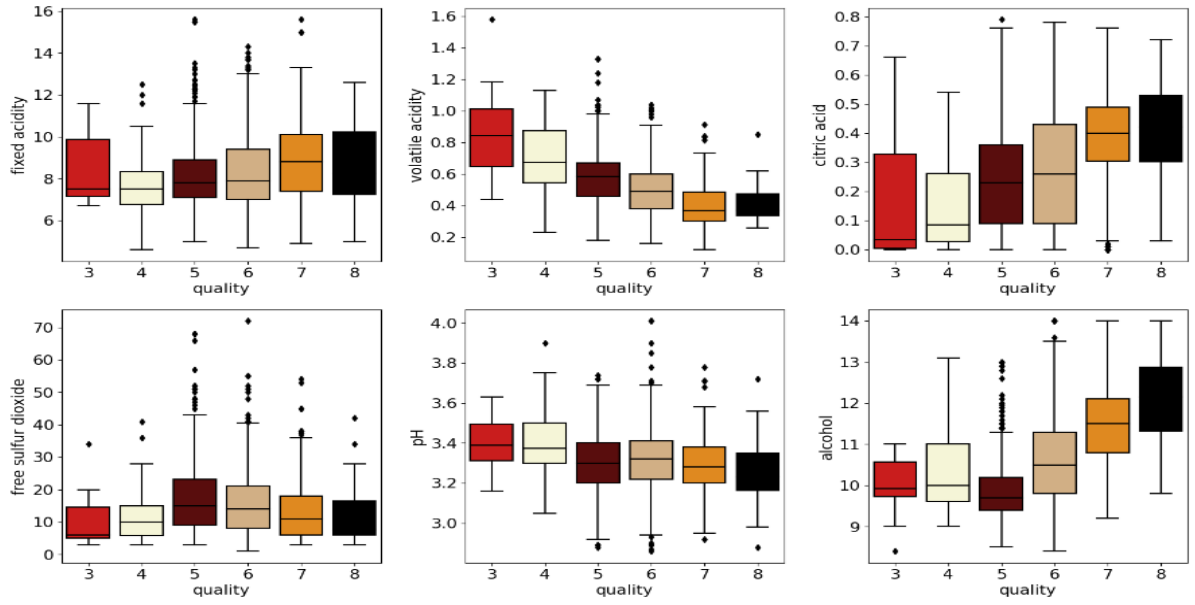


Syed Afroz Ali

```
df2=df.groupby('Type Of Restaurant')['Cost Per Head'].mean().sort_values(ascending=False)
plt.figure(figsize = (15,6))
color = ['b' if i < 500 else 'r' for i in df2]
df2.plot.bar(color=color);
```

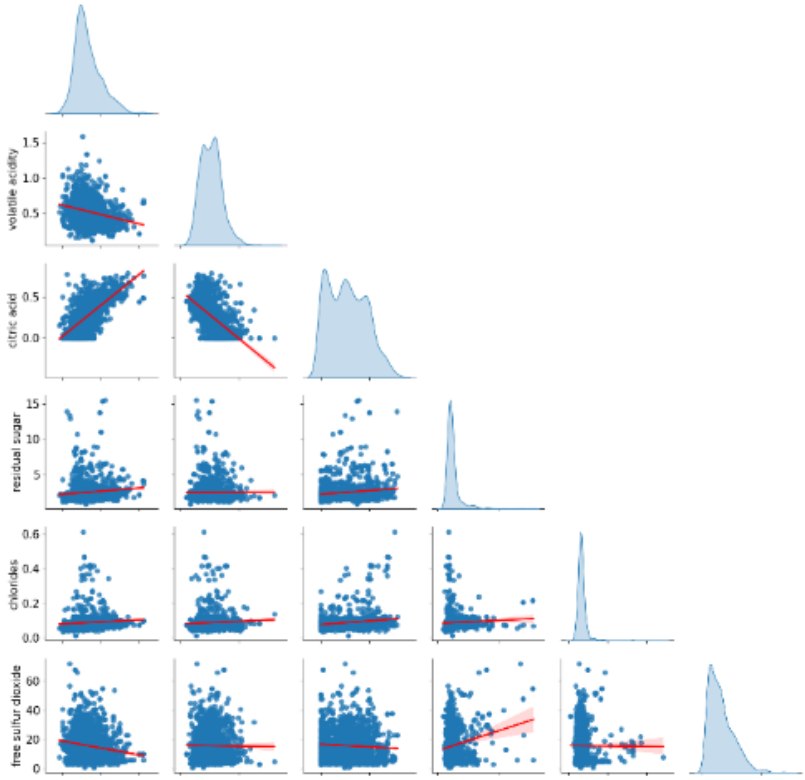



```
import math
cont_features=['fixed acidity', 'volatile acidity', 'citric acid','free sulfur dioxide','pH', 'alcohol']

y=3
x=math.ceil(len(cont_features)/y)

plt.subplots(x,y,figsize=(15,10))
for i in range(1,len(cont_features)+1) :
    plt.subplot(x,y,i)
    sns.boxplot(data=wine,y=cont_features[i-1],x='quality',palette=['#e60000','#FAFAD2','#660000','#DEB078','#FF8C00','black'])
plt.tight_layout()
plt.show()
```

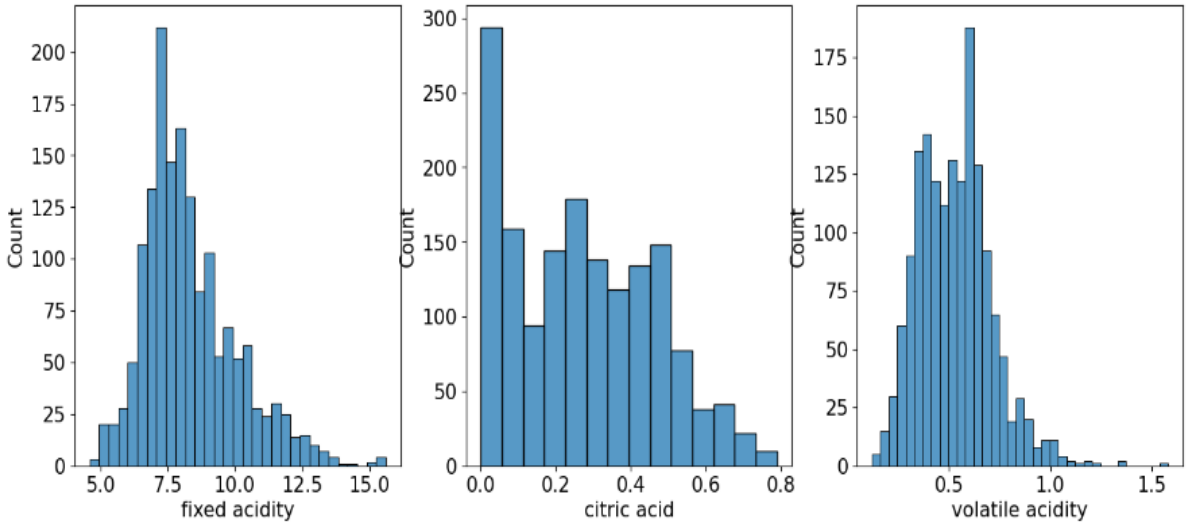


Syed Afroz Ali

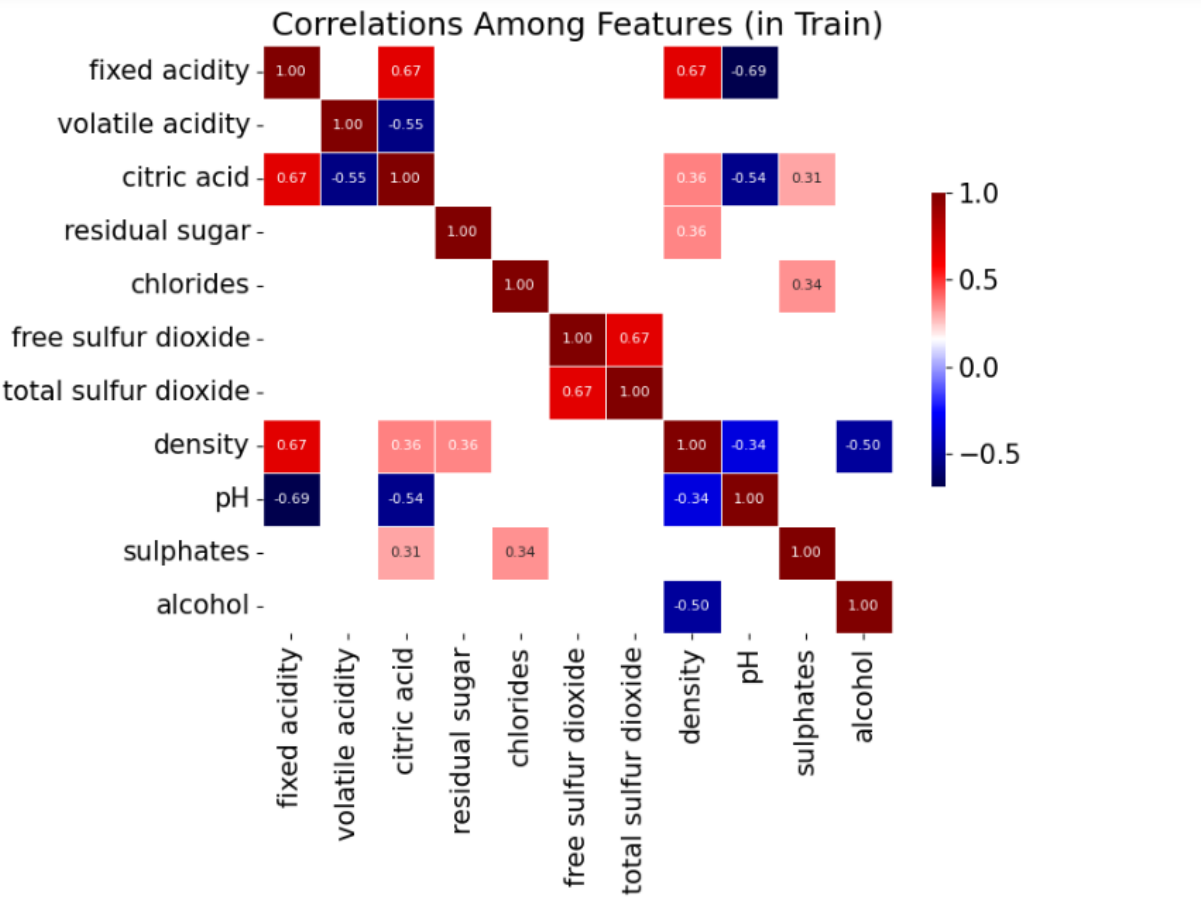
```
sns.pairplot(wine.drop(columns=['quality']),kind="reg",diag_kind='kde',plot
_kws={'line_kws':{'color':'red'}},corner=True)
plt.tight_layout()
plt.show()
```




```
features = ['fixed acidity','citric acid','volatile acidity']
fig, axs = plt.subplots(1,3, figsize=(16,6))

for f, ax in zip(features,axs.ravel()):
    sns.histplot(wine, x=f, ax=ax)
plt.show()
```

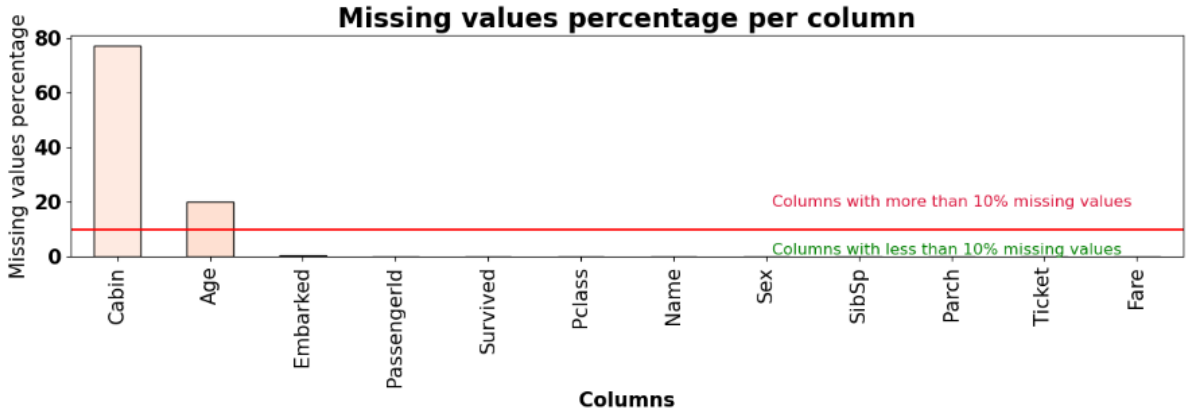


```
## Explore the correlation between all numerical features
corr_mat_train = wine.drop(columns = ['quality'], axis = 1).corr()

## Keep only correlation higher than a threshold
threshold = 0.3
corr_threshold_train = corr_mat_train[(corr_mat_train > threshold) | (corr_mat_train
< -threshold)]

plt.figure(figsize = (8, 6))
sns.heatmap(corr_threshold_train, annot = True, cmap = 'seismic', fmt = ".2f",
            linewidths = 0.5, cbar_kws={'shrink': .5},annot_kws={'size':
8}).set_title('Correlations Among Features (in Train)');
```

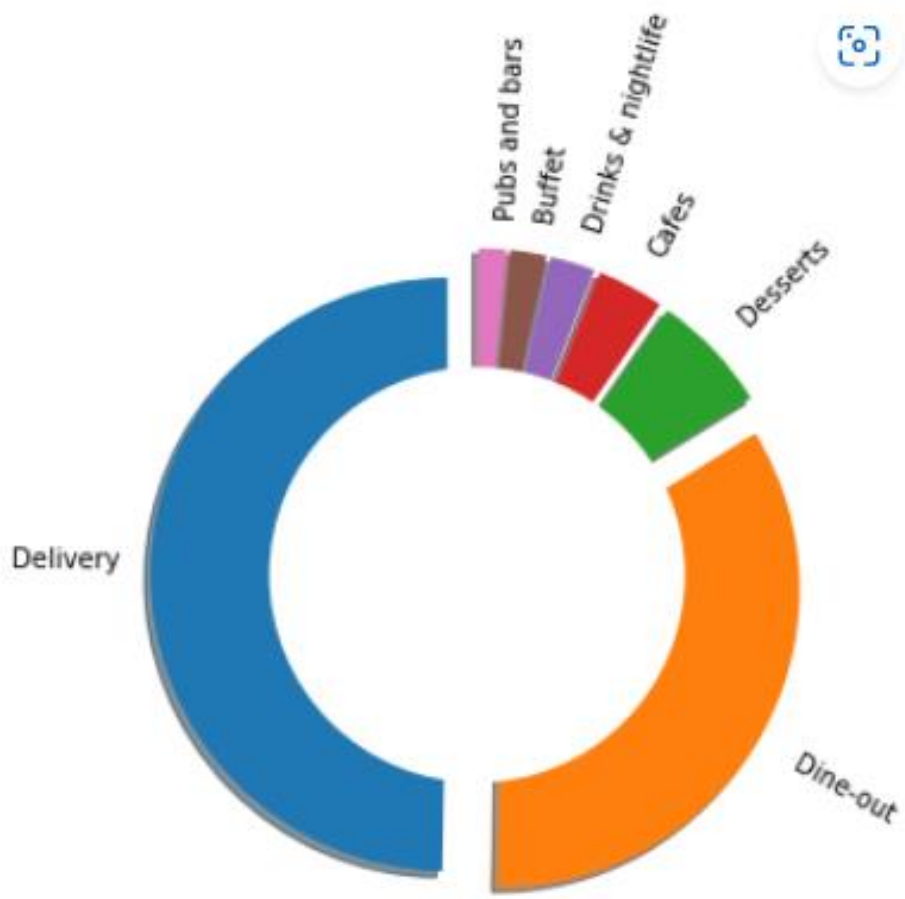



```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline

def missing_values(data, thresh = 20, color = 'black', edgecolor = 'black',
height = 3, width = 15):

    plt.figure(figsize = (width, height))
    percentage = (data.isnull().mean()) * 100
    percentage.sort_values(ascending = False).plot.bar(color = color,
edgecolor = edgecolor)
```

```
    plt.axhline(y = thresh, color = 'r', linestyle = '-')

    plt.title('Missing values percentage per column', fontsize = 20, weight =
'bold' )

    plt.text(len(data.isnull().sum()/len(data))/1.7, thresh + 12.5, f'Columns
with more than {thresh}% missing values', fontsize = 12, color = 'crimson',
         ha = 'left' ,va = 'top')
    plt.text(len(data.isnull().sum()/len(data))/1.7, thresh - 5, f'Columns with
less than {thresh}% missing values', fontsize=12, color='green',
         ha = 'left' ,va = 'top')
    plt.xlabel('Columns', size = 15, weight = 'bold')
    plt.ylabel('Missing values percentage')
    plt.yticks(weight = 'bold')

    return plt.show()

missing_values(titanic, thresh = 10, color = sns.color_palette('Reds',15))
```




```
# Pie chart
labels = df['listed_in(type)'].value_counts().index
sizes = df['listed_in(type)'].value_counts().values
# only "explode" the 2nd slice (i.e. 'Hogs')
explode = (0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1)
fig1, ax1 = plt.subplots(figsize = (8, 8))

ax1.pie(sizes, labels = labels,
        shadow = True, startangle = 90, explode = explode, rotatelabels = True)
centre_circle = plt.Circle((0, 0), 0.70,fc = 'white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)

# Equal aspect ratio ensures that pie is drawn as a circle
ax1.axis('equal')
plt.tight_layout()
plt.show()
```




```
plt.rcParams['figure.figsize'] = (18, 5)
Y = pd.crosstab(df['rate'], df['book_table'])
Y.div(Y.sum(1).astype(float), axis = 0).plot(kind = 'bar', stacked =
True,color=['red','yellow'])
plt.title('table booking vs Normal rate', fontweight = 30, fontsize = 20)
plt.legend(loc="upper right")
plt.show()
```