| **Function** | **R** | **Python** |
| --- | --- | --- |
| **Univariate Analysis – Categorical Column** | | |
| 1. Find unique values 2. Count of each category 3. Top category | Bar chart  ***ggplot(student\_score, aes(x=famsize)) + geom\_bar(stat='count', fill="steelblue", width = 0.5) + labs(x="Family Size", y="Count", title="Family Size Analysis") + theme\_minimal()***  Bar Plot when applied to the raw data without any transformation, finds the frequency of the values in the categorical column. | Count plot  ***sns.countplot(student\_score['famsize']);***  ***plt.xlabel("Family Size");***  ***plt.ylabel("Count");***  ***plt.title("Family Size Analysis");***  Count Plot counts the frequency of the values in the categorical column. |
| 1. Percentage of each category | Pareto chart  ***pareto.chart(table(student\_score$reason), main = 'Pareto Analysis-"Reasons"')*** | Pareto Chart is not in-built in Python. Percentage of the values are to be calculated using the frequency. The percentage is to be added as a separate column. A bar plot and a line plot can be plotted (as sub plots or overlay) to achieve the similar version of Pareto in R.  Alternatively, user defined functions can also be written. |
| **Univariate Analysis – Numerical Column** | | |
| 1. Density distribution | Histogram  ***hist(student\_score$age, col="steelblue", freq = FALSE)***  ***lines(density(student\_score$age), col='black',lwd=1.5)***  Plotting only the histogram does not give the density distribution. Using ggplot + geom\_hist() also gives the similar plot, with more flexibility to change the bin size, etc. | Histogram  ***sns.distplot(student\_score['age']);***  Plotting histogram using the matplotlib library (.hist) function, does not give the density distribution. |
| 1. Five point summary | Boxplot  ***ggplot(data=student\_score, aes(y=age)) + geom\_boxplot(color="steelblue") + theme\_minimal()***  Without using the plotly library, the boxplot will be static. The values will not be displayed on ‘mouse over’. | Boxplot  ***plt.boxplot(student\_score['age']);***  Without using the plotly library, the boxplot will be static. The values will not be displayed on ‘mouse over’.  Using the seaborn library, the boxplot will be horizontally oriented by default. |
| **Univariate Analysis – Text Column** | | |
| 1. Word Cloud   – text data | Word Cloud  ***wordcloud(words\_freq\_imp$words, words\_freq\_imp$Freq, min.freq = 50, colors = brewer.pal(8,"Dark2"), random.order = FALSE, random.color = TRUE)***  Initial transformation to be done on the text data and stopwords removed, before word cloud is built. | Word Cloud  ***WordCloud(stopwords=stopwords, background\_color='white').generate(str(wc['description']))***  ‘Stopwords’ are the words that do not carry any meaning. They are to be removed before word cloud is built. |
| **Bivariate Analysis** | | |
| 1. Categorical vs Numerical | Barplot  ***student\_score %>% group\_by(famsize) %>% summarise(Mean\_G1=mean(G1, na.rm=T)) %>% ggplot(aes(famsize,Mean\_G1)) + geom\_bar(stat='identity',fill='steelblue', width=0.4) + theme\_minimal()***  Data has to be transformed and then fed into the bar chart function to plot the mean of the numerical column vs the categorical column. | Barplot  ***sns.barplot(student\_score['famsize'], student\_score['G1']);***  The Bar Plot in Python automatically calculates the mean of the numerical column. It is not required to transform the data. The black line on top of each bar indicates the ‘confidence interval’.  Bar Plots can also be used to calculate the ‘sum’ or any other statistic using the ‘estimator’ parameter and the confidence interval values will change accordingly. |
| 1. Numerical vs Numerical | Scatter plot  ***ggplot(student\_score, aes(x=G3, y=absences)) + geom\_point(color='steelblue', size=1) + geom\_smooth(method="lm") + theme\_minimal()***  The regression line is achieved using the geom\_smooth. Changing the value of the ‘method’ parameter changes the type of fit. | Scatter plot  ***sns.lmplot(x='G3', y='absences', data=student\_score);***  The scatterplot gives only the scatter plot. The lmplot gives the scatter plot along with the regression line fit. |
| 1. Date vs Numerical Column   – Trending data – Timeline vs numerical data | Line Chart  ***student\_score %>% ggplot(aes(Dt,absences)) + geom\_line(color=’steelblue’) + theme\_classic()***  Like the other charts, categorical or numerical variables can be included to group the y-values. | Line Chart  ***sns.lineplot(student\_score['D'], student\_score['absences']);***  Like the other charts, categorical or numerical variables can be included (as hue) to group the y-values. |
| **Multivariate Analysis** | | |
| 1. Categorical vs Categorical vs Numerical | Grouped Bar plot  ***student\_score %>% group\_by(health, famsize) %>% summarise(Mean\_G1=mean(G1, na.rm=T)) %>% ggplot(aes(health,Mean\_G1, fill=famsize)) + geom\_bar(stat='identity', width=0.4, position="dodge") + theme\_minimal()***  The value of the ‘position’ parameter determines whether the plot is a grouped bar plot or a stacked bar plot or a 100% stacked bar plot. | Grouped Bar Plot  ***sns.barplot(student\_score['health'], student\_score['G1'], hue=student\_score['famsize']);***  The barplot function with the hue parameter outputs a grouped barplot. |
| 1. Categorical vs Numerical vs Numerical | Multiple Scatter Plot  ***ggplot(student\_score, aes(x=absences, y=G3, color=famsize)) + geom\_point(size=1) + geom\_smooth(method="lm", se=F) + theme\_minimal()***  The color paremeter can be substituted the value of a categorical column. Data is automatically grouped by the categorical column. | Multiple Scatter Plot  ***sns.lmplot(x='absences', y='G3',data=student\_score,hue='famsize');***  ‘Hue’ parameter is used for the purpose of grouping the ‘y – axis’ values. This parameter can take categorical or numerical values. It can take more thn one values too. |
| 1. Numerical vs Numerical vs Numerical | Scatter Plot  ***ggplot(student\_score, aes(x=absences, y=G3, color=age)) + geom\_point(size=1) + theme\_minimal()***  The same ‘color’ parameter as seen above can be used to substitue numerical vales. | Scatter Plot  ***sns.scatterplot(x='absences', y='G3',data=student\_score,hue='age');***  ‘Hue’ parameter takes numerical values in this case, to group the values of ‘y-axis’. |
| 1. Numerical vs Numerical vs Numerical | Heatmap  ***ggplot(student\_score,aes(absences, G3, fill= age)) + geom\_tile() + theme\_minimal()***  Heatmap can also be produced using the ‘heatmap’ function. Though the chart shows the visual co-relation of the three numerical variables, it does not give the numerical value of co-relation. | Heatmap  ***sns.heatmap(student\_score[['absences','G3','age']].corr(), annot=True);***  Heatmap is used to find corelation between two or more numeric columns. |
| 1. Numerical vs Numerical vs Numerical vs Categorical | Scatter Plot  ***ggplot(student\_score, aes(x=absences, y=G3, size=age, color=famsize)) + geom\_point() + theme\_minimal()***  When the grouping of ‘y-axis’ to be done with more than one column, then we can use both ‘color’ and ‘size’ parameter. Though not mandatory, as a best practise, the ‘color’ is used with categorical values and ‘size’ parameter is used with numerical values. | Scatter Plot  ***sns.scatterplot(x='absences', y='G3',data=student\_score,hue='age', style= 'famsize');***  Though ‘hue’ parameter can take more than one values, ‘style’ is also one more way to represent grouped values. |
| 1. Faceting | Faceting is done when y-values are to be grouped by more than two variables. More than one faceting variable can be used. Different chart types can also be produced. | The ‘col’ parameter is used for faceting in python. This is another way to group y-values, other than ‘hue’ and ‘style’. Different chart types can also be produced. |