EDA UNIVARIATE ANALYSIS:

Submission: G.V.Shriya, 122010303037, vgottumu@gitam.in

Reading the data as a dataframe into the 'df' variable

```
[ ]  df = pd.read_csv("/content/drive/MyDrive/ipl_2022_dataset.csv")
```

The variable df holds the dataframe containing the ipl 2022 data

```
[ ]  df.sample(5)
```

	Unnamed: 0	Player	Base Price	TYPE	COST IN ₹ (CR.)	Cost IN $ (000)	2021 Squad	Team
94	94	Ramesh Kumar	20 Lakh	BATTER	0.20	26.0	NaN	Kolkata Knight Riders
626	626	Hardik Tamore	20 Lakh	WICKETKEEPER	NaN	NaN	NaN	Unsold
62	62	Lalit Yadav	20 Lakh	ALL-ROUNDER	0.65	84.5	DC	Delhi Capitals
591	591	Dev Lakra	20 Lakh	ALL-ROUNDER	NaN	NaN	NaN	Unsold
224	224	Umran Malik	Retained	BOWLER	4.00	520.0	SRH	Sunrisers Hyderabad

By seeing a random 5 rows of data, we notice that there are NaN values.

So we remove those rows of data using dropna() function and store the new data in df1:

```
df1 = df.dropna()
```

Then we perform some statistical summary on the df1's TYPE, team and Cost in Dollars columns or features:

```
[ ] print(df1['TYPE'].min())
    print(df1['TYPE'].value_counts().min())

    ALL-ROUNDER
    21

[ ] print(df1['TYPE'].max())
    print(df1['TYPE'].value_counts().max())

    WICKETKEEPER
    54

[ ] print(df1['Team'].min())
    print(df1['Team'].value_counts().min())

    Chennai Super Kings
    14

[ ] print(df1['Team'].max())
    print(df1['Team'].value_counts().max())

    Sunrisers Hyderabad
    19
```

We infer that TYPE All-Rounder is having least frequency with a value of 21 and TYPE WICKETKEEPER is having the most frequency with a value of 54.

We infer that Team Chennai Super Kings is having least frequency with a value of 14 and Team Sunrisers Hyderabad is having the most frequency with a value of 19.

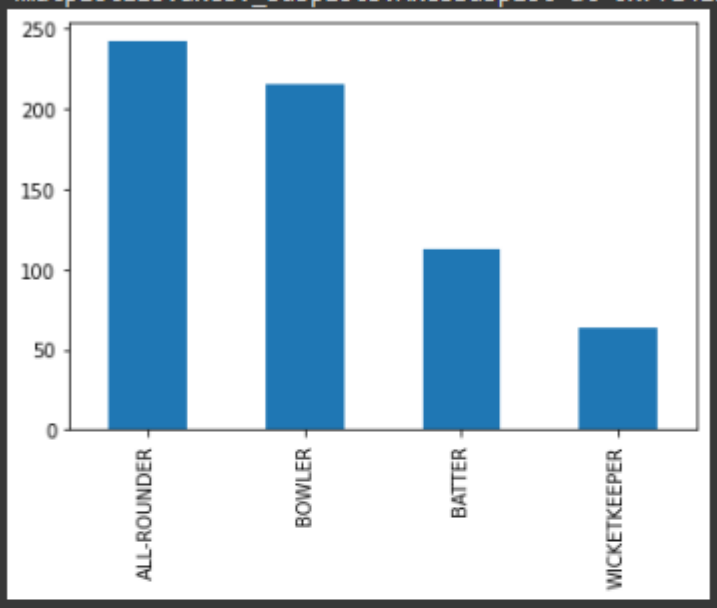
```
[ ] df1['Cost IN $ (000)'].min()

    26.0

[ ] df1['Cost IN $ (000)'].max()

    2210.0

[ ] df1['Cost IN $ (000)'].mean()

    650.2452830188679

    df1['Cost IN $ (000)'].median()

    520.0

[ ] df1['Cost IN $ (000)'].mode()

    0    26.0
    dtype: float64
```

Statistical summary with min, max, mean, median and mode is performed on the Cost in Dollars column.

Bar Graph

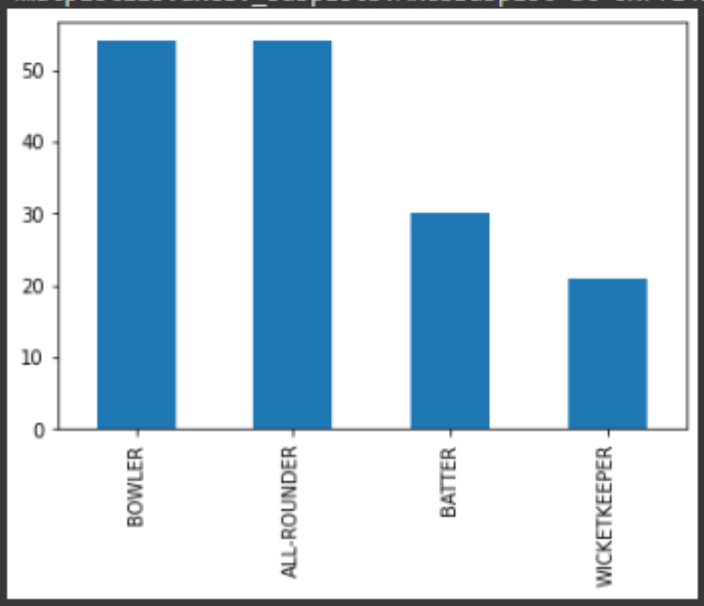
```
[ ]  df['TYPE'].value_counts().plot(kind = 'bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f142217e580>
```




```
[ ]  df1['TYPE'].value_counts().plot(kind = 'bar')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421e9f730>
```

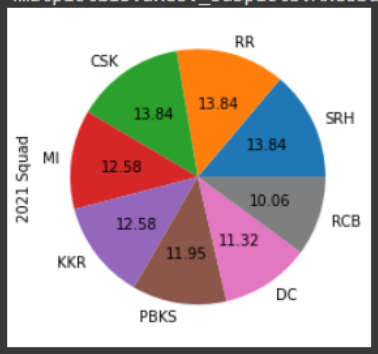


Bar graph is made for the TYPE feature, with both df and df1, and we can see how the NaN values presence influences the graph. Without eliminating NaN values, ALL-ROUNDER column seems higher than the BOWLER column but the reverse happens after removing the NaN values. In both the cases WICKETKEEPER is the least frequent category.

Pie Chart

```
[ ] df['2021 Squad'].value_counts().plot(kind = 'pie', autopct = "%.2f")
```

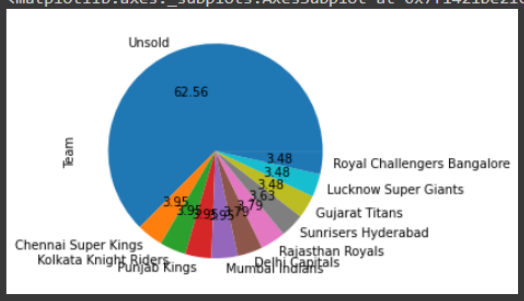
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421f44070>
```



A pie chart visualization of the 2021 Squad column's percentage distribution shows that CSK, RR and SRH have same percentage representations and the highest entries comparatively.

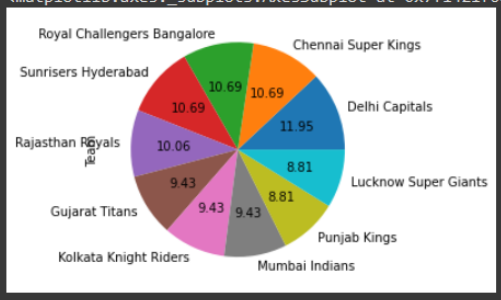
```
[ ] df['Team'].value_counts().plot(kind = 'pie', autopct = "%.2f")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421be21c0>
```




```
[ ] df1['Team'].value_counts().plot(kind = 'pie', autopct = "%.2f")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421f6e9d0>
```

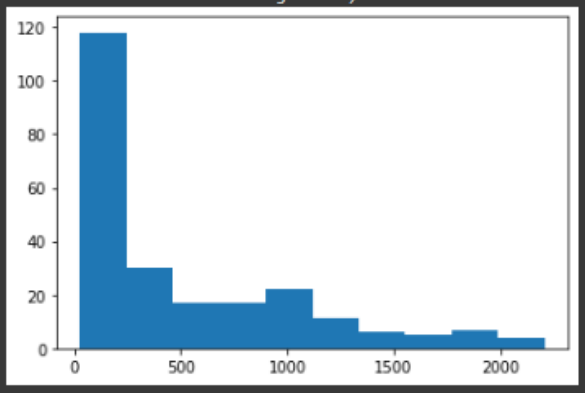


Here there are two pie charts, one for each of df and df1. We see that NaN values included dataframe df has more percentage of Unsold category whereas in df1 theres no Unsold category representation and Delhi Capitals had the highest percentage.

Histogram

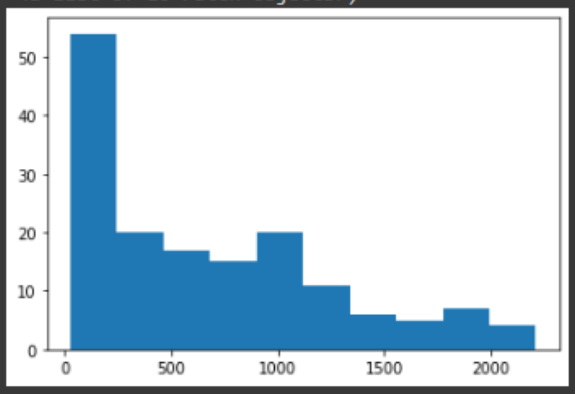
```
[ ] plt.hist(df['Cost IN $ (000)'])
```

```
(array([118.,  30.,  17.,  17.,  22.,  11.,   6.,   5.,   7.,   4.]),
 array([  26. ,  244.4,  462.8,  681.2,  899.6, 1118. , 1336.4, 1554.8,
        1773.2, 1991.6, 2210. ]),
 <a list of 10 Patch objects>)
```




```
plt.hist(df1['Cost IN $ (000)'])
```

```
(array([54., 20., 17., 15., 20., 11.,  6.,  5.,  7.,  4.]),
 array([  26. ,  244.4,  462.8,  681.2,  899.6, 1118. , 1336.4, 1554.8,
        1773.2, 1991.6, 2210. ]),
 <a list of 10 Patch objects>)
```

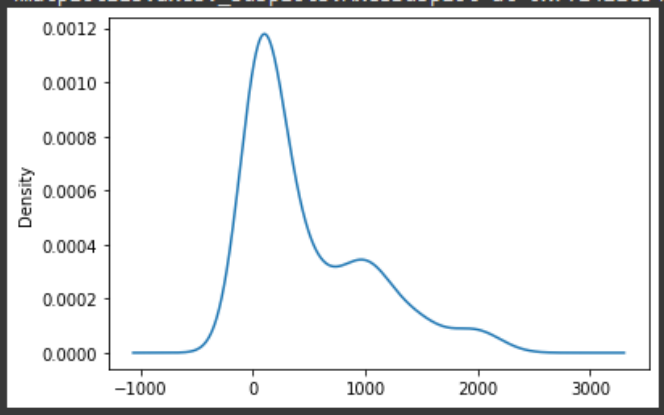


The two histograms show the frequency distribution of the Cost in Dollars column, with each visualization for one of df and df1. We see that NaN values included dataframe df has lesser granularity than df1. In both the cases, the graph is skewed towards the left with a distinct spike in 0-500 range.

Density Plot

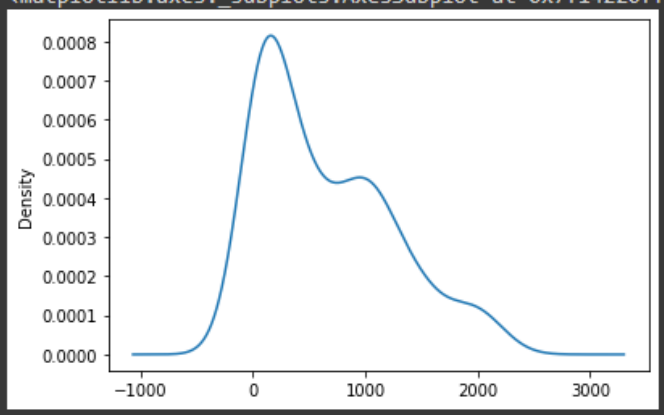
```
[ ]  df['Cost IN $ (000)'].plot(kind='density')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421cb4850>
```




```
[ ]  df1['Cost IN $ (000)'].plot(kind='density')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f14220ff1c0>
```

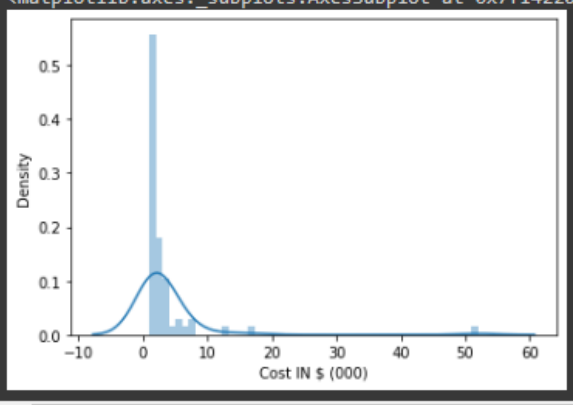


The two density plots show the distribution of the Cost in Dollars column, with each visualization for one of df and df1. It uses a kernel density estimate to show the probability density function of the variable Cost in Dollars. In both the cases, there is a distinct spike near the value of 0. But we see that NaN values included dataframe df has a higher spike with a density of 0.0012 than in df1 where the spike is having a density of 0.0008.

DistPlot

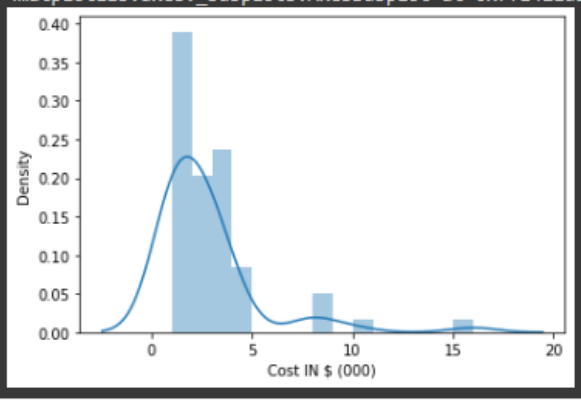
```
[ ]  sns.distplot(df['Cost IN $ (000)'].value_counts())
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f1422058280>
```




```
[ ]  sns.distplot(df1['Cost IN $ (000)'].value_counts())
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421d1ce80>
```

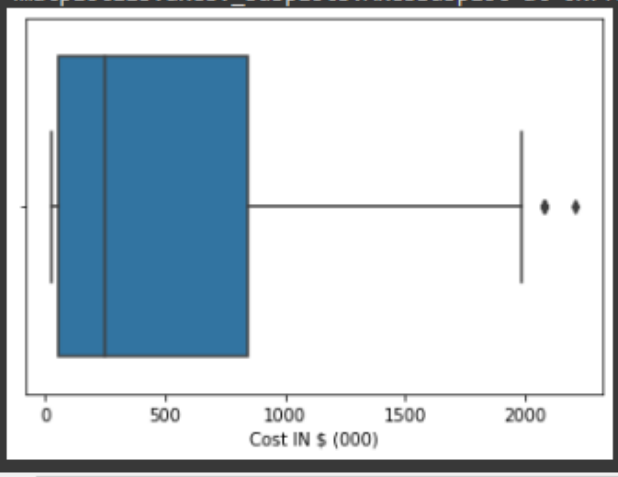


The two distplots show the variation in the data distribution of the Cost in Dollars column, with each visualization for one of df and df1. We see that NaN values included dataframe df has lesser granularity than df1. In both the cases, a distinct spike is noticed in 0-5 dollar region.

BoxPlot

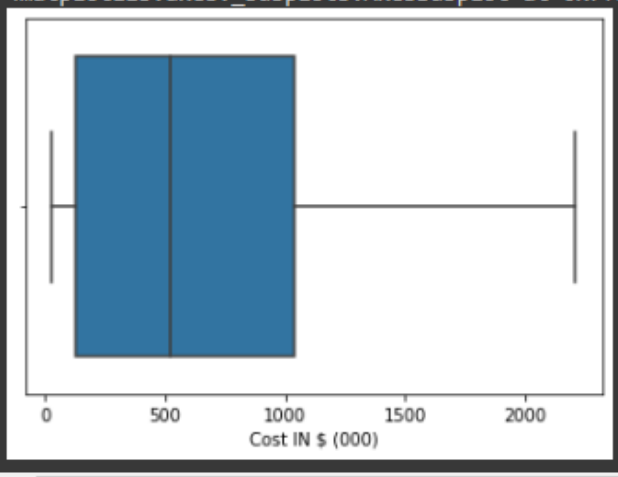
```
[ ]  sns.boxplot(df['Cost IN $ (000)'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:3
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421df1dc0>
```




```
[ ]  sns.boxplot(df1['Cost IN $ (000)'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:3
  warnings.warn(
<matplotlib.axes._subplots.AxesSubplot at 0x7f1421dc72b0>
```



The median is skewed to the left and there are outliers in the visualization with df dataframe. The interquartile range lies in less than 1000 dollar range. But in the df1 dataframe, it crosses the 1000 dollar mark. The interquartile range lies closer to the minimum whisker in df than in df1.