Descriptive Statistical:

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.

```
flight_data.describe()
```

Python

	YEAR	QUARTER	MONTH	DAY_OF_MONTH	DAY_OF_WEEK	FL_NUM	ORIGIN_AIRPORT_ID	DEST_AIRPORT_ID	CRS_DEP_TIME	DEP_TIME	...	CRS_ARR_TIME	ARR_TIME	ARR_DELAY	ARR_DEL15	CANCELLED	DIVERTED	CRS_ELAPSED_TIME	ACTUAL_ELAPSED_TIME	DISTANCE	Unnamed: 25
count	11231.0	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11231.000000	11124.000000	...	11231.000000	11116.000000	11043.000000	11043.000000	11231.000000	11231.000000	11231.000000	11043.000000	11231.000000	0.0
mean	2016.0	2.544475	6.628973	15.790758	3.990199	1334.325617	12334.516695	12302.274508	1320.798326	1327.189410	...	1537.312795	1523.978499	-2.573123	0.124513	0.010150	0.006589	190.052124	179.661233	1161.031965	NaN
std	0.0	1.090701	3.356678	8.782056	1.995257	811.875227	1595.026510	1601.988550	490.737845	500.306462	...	502.512494	512.536541	39.232521	0.330181	0.100241	0.080908	78.186317	77.940399	643.663379	NaN
min	2016.0	1.000000	1.000000	1.000000	1.000000	7.000000	10397.000000	10397.000000	10.000000	1.000000	...	2.000000	1.000000	-67.000000	0.000000	0.000000	0.000000	93.000000	75.000000	559.000000	NaN
25%	2016.0	2.000000	4.000000	8.000000	2.000000	624.000000	10397.000000	10397.000000	905.000000	905.000000	...	1130.000000	1135.000000	-19.000000	0.000000	0.000000	0.000000	127.000000	117.000000	594.000000	NaN
50%	2016.0	3.000000	7.000000	16.000000	4.000000	1267.000000	12478.000000	12478.000000	1320.000000	1324.000000	...	1559.000000	1547.000000	-10.000000	0.000000	0.000000	0.000000	159.000000	149.000000	907.000000	NaN
75%	2016.0	3.000000	9.000000	23.000000	6.000000	2032.000000	13487.000000	13487.000000	1735.000000	1739.000000	...	1952.000000	1945.000000	1.000000	0.000000	0.000000	0.000000	255.000000	236.000000	1927.000000	NaN
max	2016.0	4.000000	12.000000	31.000000	7.000000	2853.000000	14747.000000	14747.000000	2359.000000	2400.000000	...	2359.000000	2400.000000	615.000000	1.000000	1.000000	1.000000	397.000000	428.000000	2422.000000	NaN

rows × 22 columns

Visual Analysis

Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions.

Univariate Analysis

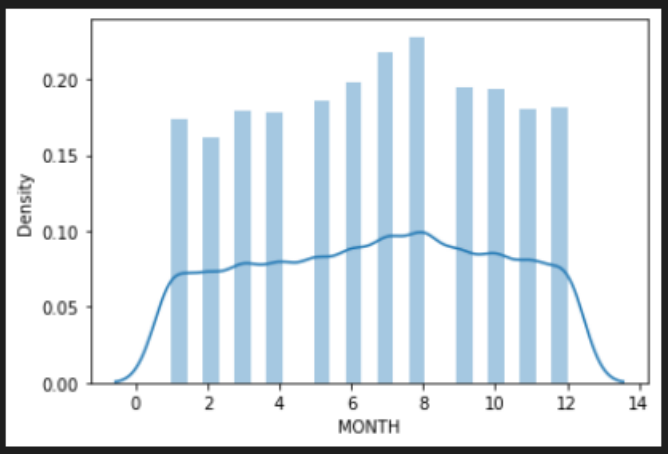
In simple words, univariate analysis is understanding the data with a single feature. Here we have displayed two different graphs such as distplot and countplot.

The Seaborn package provides a wonderful function distplot. With the help of distplot, we can find the distribution of the feature. To make multiple graphs in a single plot, we use subplot.

```
sns.distplot(flight_data.MONTH)
```

```
C:\Users\Saumya\Anaconda3\lib\site-packages\seaborn\distributions.py:2557:
figure-level function with similar flexibility) or `histplot` (an axes-lev
  warnings.warn(msg, FutureWarning)

<AxesSubplot:xlabel='MONTH', ylabel='Density'>
```



- In our dataset we have some categorical features. With the count plot function, we are going to count the unique category in those features. We have created a dummy data frame with categorical features. With for loop and subplot we have plotted this below graph.
- From the plot we came to know, Applicants income is skewed towards left side, where as credit history is categorical with 1.0 and 0.0

Countplot:-

A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic API and options are identical to those for barplot() , so you can compare counts across nested variables.

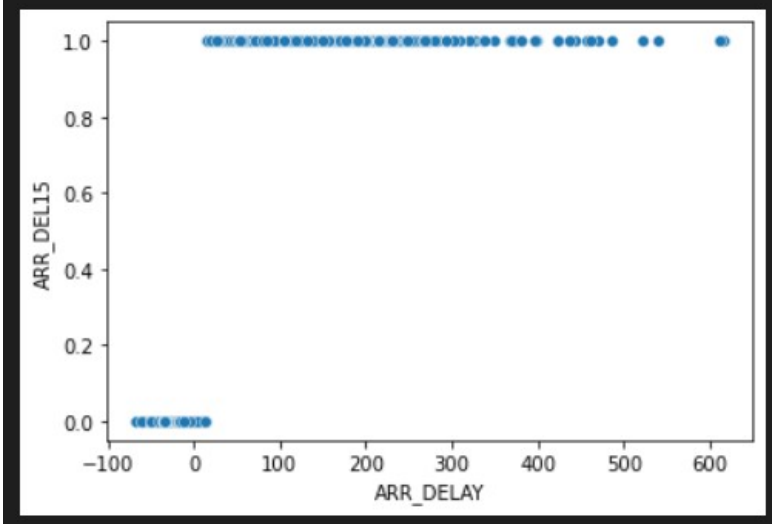
From the graph we can infer that , gender and education is a categorical variables with 2 categories , from gender column we can infer that 0-

category is having more weightage than category-1,while education with 0,it means no education is a underclass when compared with category -1, which means educated .

Bivariate Analysis

```
sns.scatterplot(x='ARR_DELAY',y='ARR_DEL15',data=flight_data)
```

```
<AxesSubplot:xlabel='ARR_DELAY', ylabel='ARR_DEL15'>
```

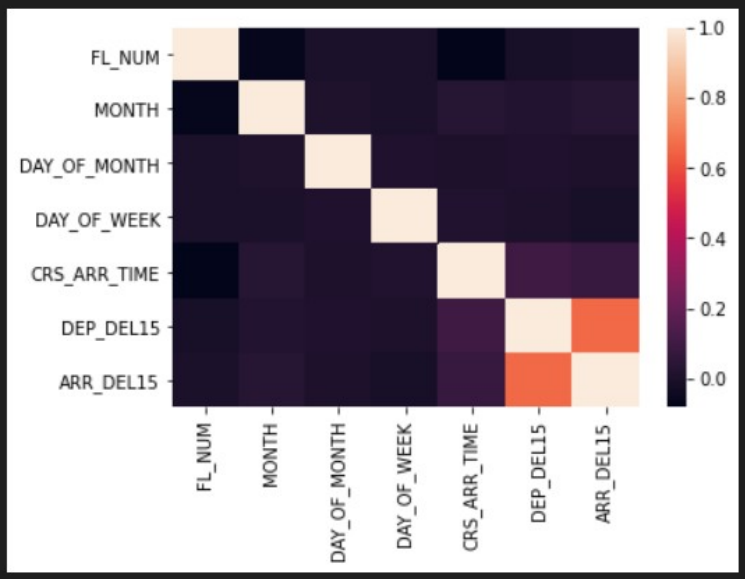


Multivariate Analysis

In simple words, multivariate analysis is to find the relation between multiple features. Here we have used a swarm plot from the seaborn package.

```
[32]  sns.heatmap(dataset.corr())
```

```
<AxesSubplot:>
```



From the above graph we are plotting the relationship all the features.

Splitting data into dependent and independent variables

```
dataset = pd.get_dummies(dataset, columns=['ORIGIN', 'DEST'])
dataset.head()
```

```
x = dataset.iloc[:, 0:8].values
y = dataset.iloc[:, 8:9].values
```

Splitting data into train and test

Now let’s split the Dataset into train and test sets

Changes: first split the dataset into x and y and then split the data set

Here x and y variables are created. On x variable, df is passed with dropping the target variable. And on y target variable is passed. For splitting training and testing data we are using the train_test_split() function from sklearn. As parameters, we are passing x, y, test_size, random_state.

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=0)
```

```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(dataset.drop('ARR_DEL15', axis=1), df['ARR_DEL15'], test_size=0.2, random_state=0)
```

```
x_test.shape
```

```
(2247, 16)
```

```
x_train.shape
```

```
(8984, 16)
```

```
y_test.shape
```

```
(2247, 1)
```

+ Code + Markdown

```
y_train.shape
```

```
(8984, 1)
```

Scaling the data

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)
```

```
sns.catplot(x="ARR_DEL15",y="ARR_DELAY",kind='bar',data=flight_data)
```

```
<seaborn.axisgrid.FacetGrid at 0x22716099eb0>
```