# ***Lab # 03***

***Title:*** Demonstrate how to perform exploratory data analysis on give dataset

## What is Exploratory Data Analysis?

Exploratory Data Analysis or EDA is used to take insights from the data. Data Scientists and Analysts try to find different patterns, relations, and anomalies in the data using some statistical graphs and other visualization techniques. Following things are part of EDA :

1. Get maximum insights from a data set
2. Uncover underlying structure
3. Extract important variables from the dataset
4. Detect outliers and anomalies(if any)
5. Test underlying assumptions
6. Determine the optimal factor settings

## Why EDA is Important?

The main purpose of EDA is to detect any errors, outliers as well as to understand different patterns in the data. It allows Analysts to understand the data better before making any assumptions. The outcomes of EDA helps businesses to know their customers, expand their business and take decisions accordingly.

## Data Description

In this lab, we have a bank that wants to know who amongst their existing customers is going to churn. In the below dataset, each row represents an individual customer.

Download churn modeling data: Link

The **'Exited'** column in the far right says if the customer 'Exited' (churned) or not. The remaining columns such as 'CreditScore', 'Age', 'Tenure', 'Balance', 'IsActiveMember' etc may contain patterns that can used to predict if the customer 'Exited' or not.

Let's see some standard examples of how to go about performing EDA on the Churn modeling dataset.

## Basic information about data

The df.info() function will give us the basic information about the dataset. For any data, it is good to start by knowing its information. Let's see how it works with our data.

```
Dataset.info()
```

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   RowNumber        10000 non-null  int64
 1   CustomerId       10000 non-null  int64
 2   Surname          10000 non-null  object
 3   CreditScore      10000 non-null  int64
 4   Geography        10000 non-null  object
 5   Gender           10000 non-null  object
 6   Age              10000 non-null  int64
 7   Tenure           10000 non-null  int64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64
 10  HasCrCard        10000 non-null  int64
 11  IsActiveMember   10000 non-null  int64
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)

memory usage: 1.1+ MB
```

## Unique values in the data

You can find the number of unique values in the particular column using unique() function in python.

```
#unique values
Dataset['Exited'].unique()
```

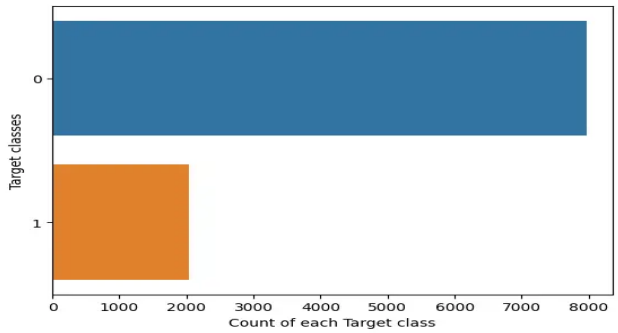
Output:

```
array([1, 0])
```

## Check frequency counts of Target

The Seaborn.countplot() method is used to display the count of categorical observations in each bin in the dataset. A count plot resembles a histogram over a categorical variable as opposed to a quantitative one

```
import seaborn as sns
from matplotlib import pyplot as plt
# Check distribution of target class
sns.countplot(y="Exited" ,data=Dataset)
plt.xlabel("Count of each Target class")
plt.ylabel("Target classes")
plt.show()
```



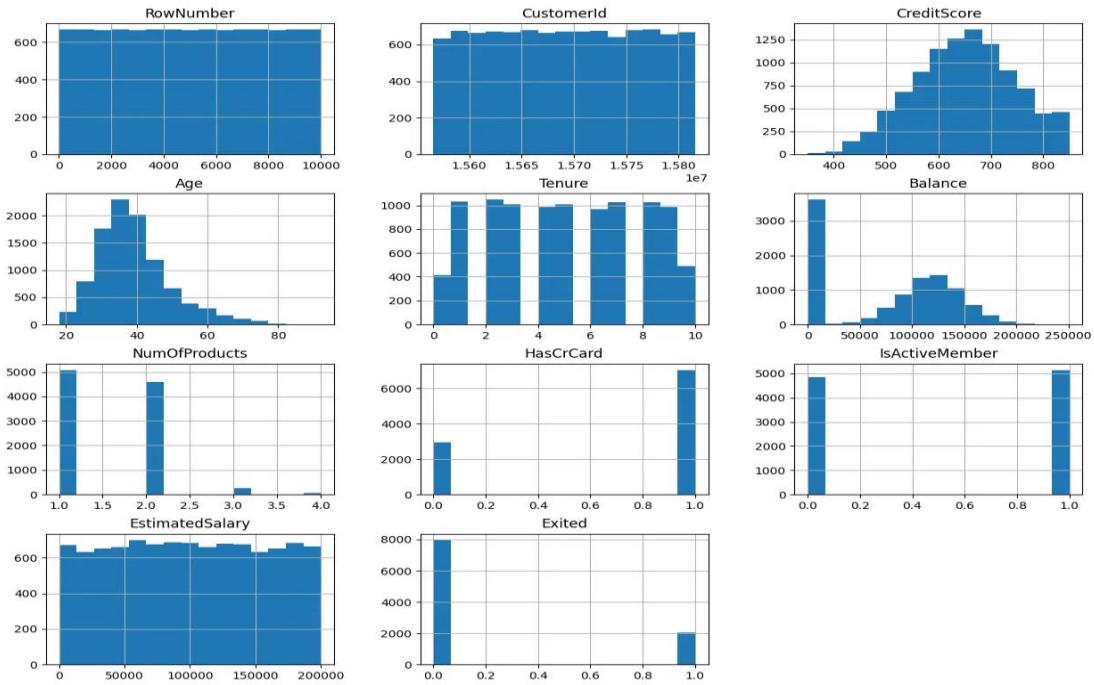

```
# Value counts
print(Dataset['Exited'].value_counts())
OutPut:
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```

So, the number of people churned is about one-fifth of the total number. Though it's quite high, there is a large class imbalance.

## Check distribution of every feature

```
# Check the distribution of all the features
df.hist(figsize=(15,12),bins = 15)
plt.title("Features Distribution")
plt.show()
```



Looking at the charts, it's quite clear which values are discrete (NumofProducts, HasCreditCard, etc) and which ones are continuous. You can also see the shapes of the distributions.

## Check how different numerical features are related to target class

How to check if a numeric variable can be helpful in predicting a target class?

**A numeric column will be more useful predictor, if there is a significant difference in mean of the target for the various values of the predictor.**

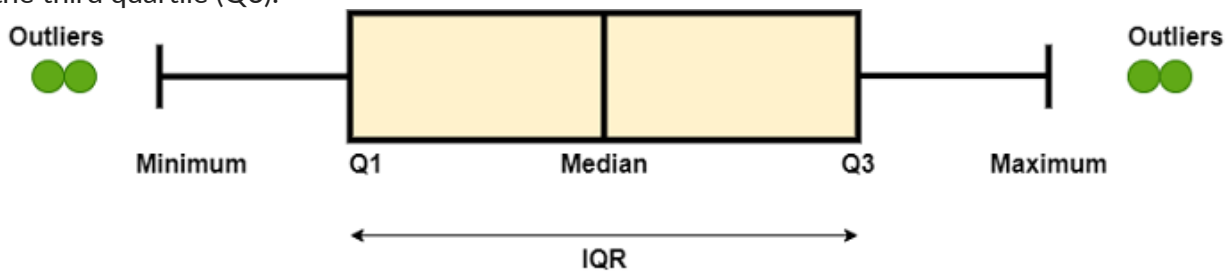
The opposite it not necessarily true, that is, it does not mean it is not useful if such difference exists. There are be patterns that are not see, that can help improve the model predictions even when no visible mean difference is observed.

### Box plot

A Box plot is a way to visualize the distribution of the data by using a box and some vertical lines. It is known as the whisker plot. The data can be distributed between five key ranges, which are as follows:

1. **Minimum**: Q1-1.5*IQR
2. **1st quartile** (Q1): 25th percentile
3. **Median**:50th percentile
4. **3rd quartile**(Q3):75th percentile
5. **Maximum**: Q3+1.5*IQR

Here IQR represents the InterQuartile Range which starts from the first quartile (Q1) and ends at the third quartile (Q3).




Take a particular numeric predictor. Then, make a box plot for both the values of the target. A useful predictor will typically have a different position of the boxes and / or the size of the boxes will be different.

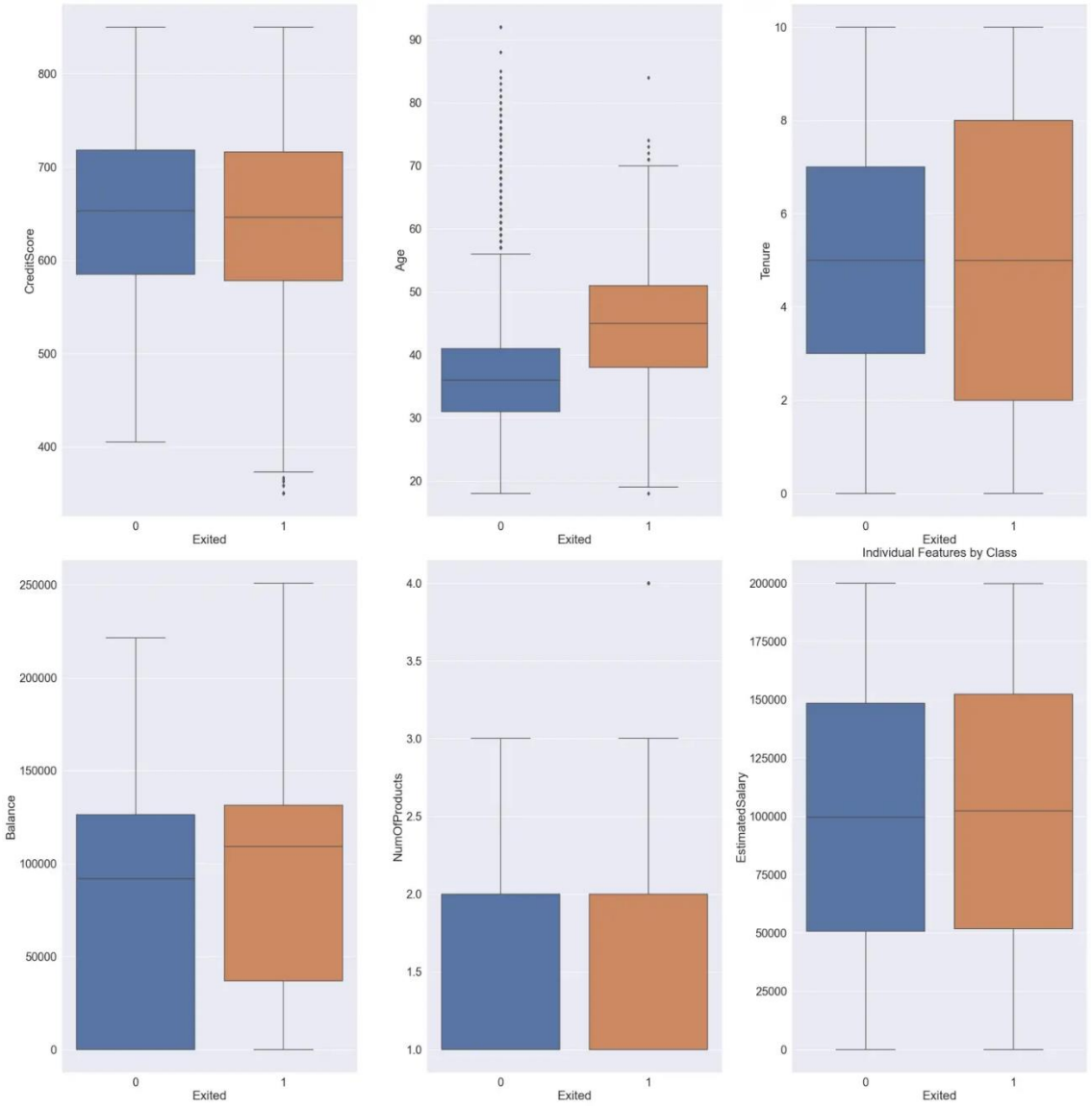
Increase font size globally.

```
sns.set(font_scale=2)
```

Plot boxplots one each value of the target, for each variable.

```
# Check the distribution of y variable corresponding to every x variable
```

```
sns.boxplot(x = Dataset["Exited"], y = Dataset["CreditScore"])
plt.tight_layout()
plt.title("Individual Features by Class")
plt.show()
```



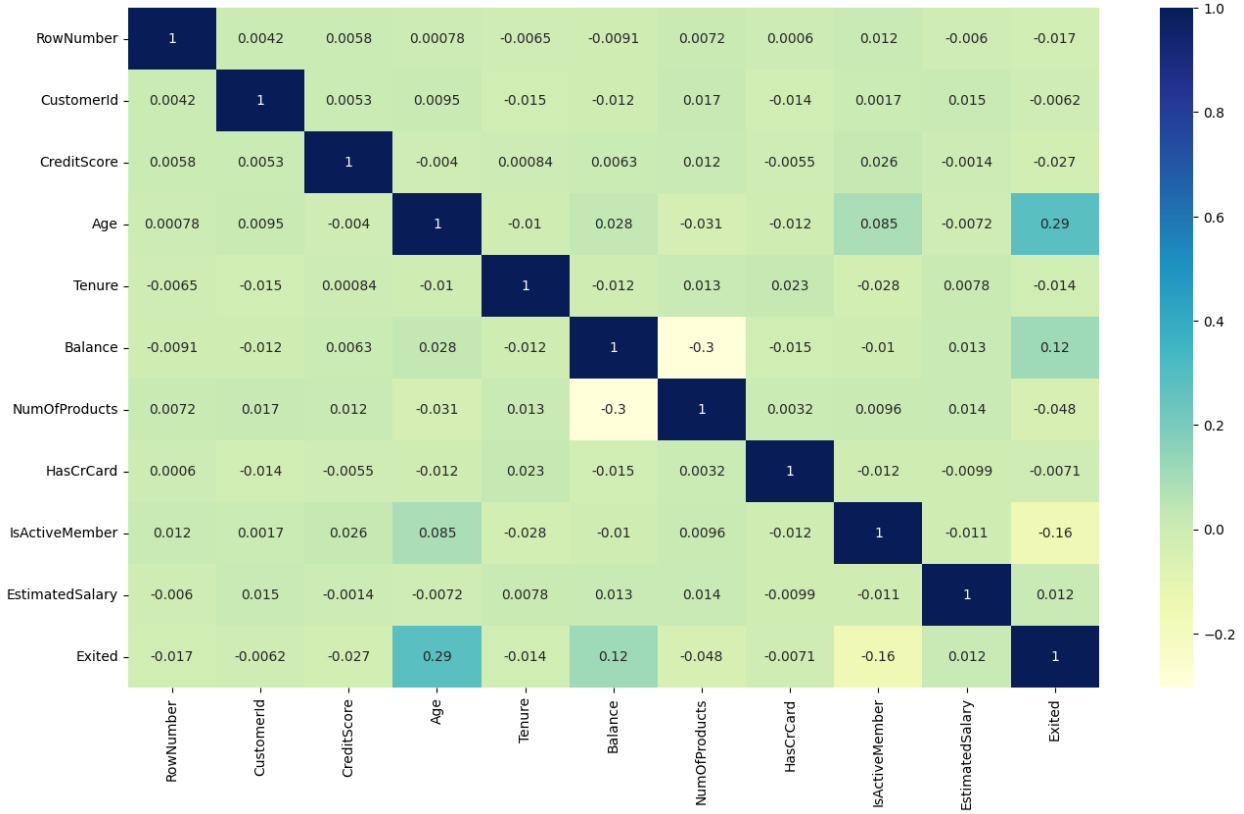
From the above plots, the position / size of the boxes are different for: Credit Score, Age, Tenure, Balance, Estimared Salary (to a certain extent). So, these variables will probably be helpful in predicting the 'Exited' column.

## Correlation heatmap

A correlation heatmap is a heatmap that shows a 2D correlation matrix between two discrete dimensions, using colored cells to represent data from usually a monochromatic scale. The values of the first dimension appear as the rows of the table while of the second dimension as a column. The color of the cell is proportional to the number of measurements that match the dimensional value. This makes correlation heatmaps ideal for data analysis since it makes patterns easily readable and highlights the differences and variation in the same data. A correlation heatmap, like a regular heatmap, is assisted by a colorbar making data easily readable and comprehensible Each square shows the correlation between the variables on each axis. Correlation ranges from -1 to +1. Values closer to zero means there is no linear trend between the two variables. The close to 1 the correlation is the more positively correlated they are; that is as one increases so does the other and the closer to 1 the stronger this relationship is. A correlation closer to -1 is similar, but instead of both increasing one variable will decrease as the other increases. The diagonals are all 1 because those squares are correlating each variable to itself (so it's a perfect correlation). For the rest the larger the number and darker the color the higher the correlation between the two variables. The plot is also symmetrical about the diagonal since the same two variables are being paired together in those squares

**Syntax:** heatmap(data, vmin, vmax, center, *cmap,.....)*

```
plt.figure(figsize = (16,9))
dataplot=sns.heatmap(Dataset.corr(),cmap="YlGnBu", annot=True)
```




## Comparing distributions with pair plot

A pairplot is a type of data visualization provided by the Seaborn library that allows you to create a matrix of scatterplots and histograms for a dataset. It helps you visualize the relationships between pairs of variables in your data and understand the distribution of each individual variable.

***Syntax:*** *seaborn.pairplot( data, **kwargs )*

***Parameter:***

***data:*** *Tidy (long-form) dataframe where each column is a variable and each row is an observation.*

***hue:*** *Variable in "data" to map plot aspects to different colors.*

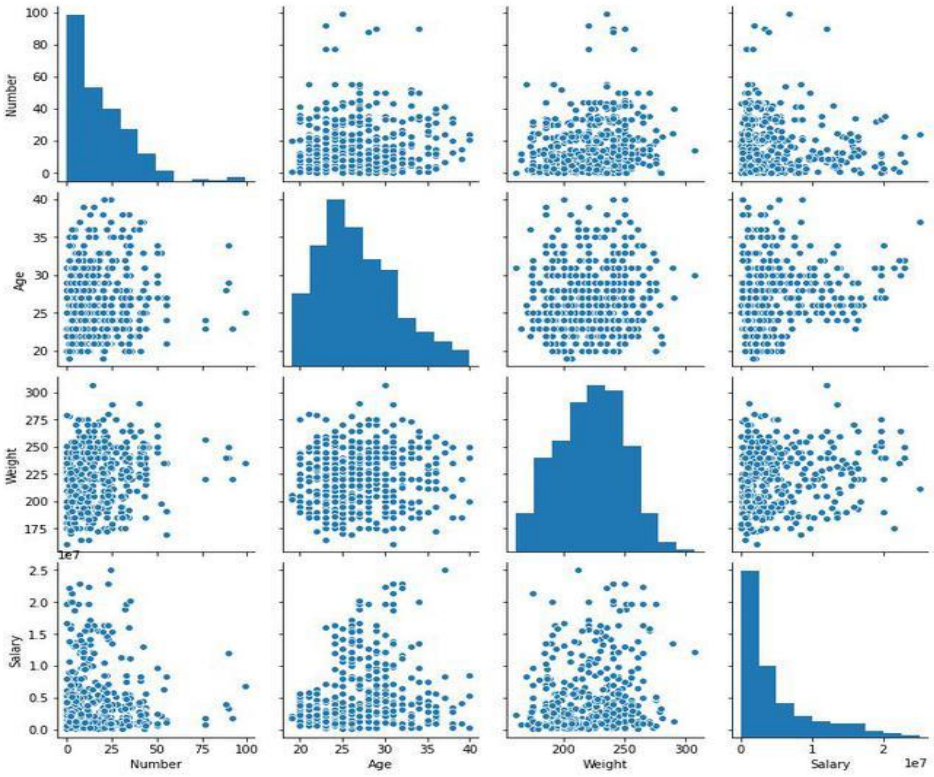
***palette:*** *dict or seaborn color palette*

***{x, y}_vars:*** *lists of variable names, optional*

***dropna:*** *boolean, optional*

```
# load the csv
data = pandas.read_csv("nba.csv")
```

```
# pairplot
seaborn.pairplot(data)
```




## Lab Task #01:

Select a dataset from a reputable source or dataset repository (e.g., UCI Machine Learning Repository or Kaggle) with a meaningful and interesting context (e.g., housing prices, customer reviews, or health data). Load the selected dataset. Begin by providing a brief overview of the dataset, including its size (number of rows and columns) and the types of variables. Calculate and display summary statistics for key variables in the dataset, such as mean, median, standard deviation, minimum, and maximum values. Create visualizations, such as histograms, box plots, or scatter plots, to explore the distribution and relationships of variables. Discuss any initial observations or insights gained from this exploratory analysis. Are there any outliers or interesting patterns that stand out?

## Lab Task #02

Select a subset of variables: Choose a specific subset of variables from the dataset that you find particularly interesting or relevant to your analysis. For example, if you're working with a housing dataset, you might focus on variables like "square footage" and "price." Create advanced visualizations: Utilize advanced visualization techniques to gain deeper insights into the selected variables. Consider creating visualizations such as:

- Heatmaps: Generate a heatmap to visualize correlations between variables within the chosen subset.
- Violin Plots: Create violin plots to show the distribution of data and the probability density at different values.
- Pairwise Scatterplots: Generate scatterplots for pairwise combinations of variables to explore multivariate relationships.
- Time Series Plots: If applicable, create time series plots to analyze trends and patterns over time.

Interpret the advanced visualizations: Analyze the advanced visualizations and provide interpretations of the patterns and insights they reveal. Discuss any notable relationships, trends, or anomalies you observe within the selected variables