**EDA Notes:**

4 steps of solving data analytics problem:

1) business understanding.

2) data understanding.

3) data collection.

4) data preparation.(taking unclean data and clean data)

Steps involved in EDA:

1) Data sourcing

2) Data cleaning

3) Univariate and Bivariate analysis with visualization. // categorial and numerical data analysis.

4) Derived Metrics.

DATA ANALYTICS/SCIENCE PROCESS:

1) data collection (where we get data from the database, web scripting, directly from the client)

initially, data is raw data.

iterative approach:(take data clean it)

{

2) then data gets processed.

3) clean dataset (extracting relevant info from data, some techniques that we follow in cleaning) Clean data is input for our EDA process.

4) Exploratory Data Analysis: during eda, we also find some irrelevant info and we remove that. After EDA, we get insights into the problem statement and the possibility of the problem occurring.

}

Note: In some scenarios, we get directly from cleaning to Models, EDA is not necessary.

5) Model & algorithms: write an algorithm or sol for the problem.

6) Final Data Product: after the algorithm or model design we finally create a product.

7) Visualize Report: Using Power bu, tableau, or Python we visualize reports.

8) Make decisions: Based on our report we finally get to a decision.

WHAT IS EDA?

EDA is an approach to analyze the datasets to summarize their main characteristics in the form of visual methods. A first step in the data analysis process. It also helps us to find errors, discover data, map out data structure, and find anomalies.

**1- VISUALIZATION:**

A presentation of the data in graphical or visual form to understand the data more clearly.

ADVANTAGES:

1)easily analyze the data and summarize.

2)help to get meaningful insights from data.

3)help to find the trend or pattern of the data.

STEPS INVOLVED IN EDA:

1) Data sourcing: data collection from multiple sources.

Two kinds of data:

1) public: (publicly available data for research).

2)private: (private which is not accessible or we need access permission.)

**2) Data Cleaning:**

You get rid of any additional information that isn't required.

ADVANTAGES:

accuracy of the model.

STEPS OF DATA CLEANING:

1)handle missing values.

2)standardization of the data.

3)outlier treatment.

4)handle invalid values.

1. **Handle missing values:**

Delete Rows/Columns.

Rows can be deleted if it has an insignificant number of missing values.

Columns can be deleted if it has more than 75% of the missing value.

2)Replacing with mean/median/mode.

This method can be used on independent variables when it has numerical variables. On categorical features, we apply the median more to fill in missing values.

3)Algorithm imputation:

Some ML algorithms don't care about the missing values they handle missing values in the datasets.

4)Predicting the missing values:

dataset with no missing value becomes the training set and the dataset with a missing value becomes the test set and the missing value is treated as the target variable.

**PYTHON CODE FOR FINDING MISSING VALUES:**

------------------------------------------------------------------------------------------------------------------------------------------

import pandas as pd

improt numpy as np

import matplotlib.pyplot as plt

df = pd.read\_csv('/content/Churn\_Modeling.csv')

df.info()

ANOTHER WAY OF FINDING VALUES:

printf(df.isnull()lsum()) //isnull() is one function to determine the null values

DELETING THE COLUMNS WITH MISSING VALUES:

updated df = df.dropna(axis=1)

updated\_df.info()

PROBLEM WITH THIS METHOD:

The problem is losing this data could lose important info about age and gender since they have around < (1 & 3)%

This method can only be use if there are so many null values

DELETING THE ROWS WITH MISSING VALUES:

updated df = df.dropna(axis=0) //axis=0 is for row

updated\_df.info()

This method is more accurate because the columns contains more valuable information than we expected.

In ML models, this can be a technique and can be used but there are some better options too.(such as imputation).

Hit & Trial is one way to determine which is best method acc to the data we are given with.

IMPUTATION: [In this case we are filling data with a certain number.]

1) Filling the missing data with mean or median if its a numerical variable.

2) Filling the missing data with mode if its a categorical value.

3) Filling the numerical value with 0 or -9999, or some other num that will not occour in data.

4) Filling the categorical value with a new type for the missing values.

Detemining mean & median values:

df['Age'].mean();

df['Age'].median();

#filna: fills the null records

#dropna: drops the null records

updated\_df = df

updated\_df['Age']=updated\_df['Age'].filna(df['Age'.mean()])

updated\_df.info()

updated\_df['Age']=updated\_df['Age'].filna(df['Age'.mean()])

upon treating with median it will still clears the null value problem. Use median when u have more outliers and mean for less.

Forward & Backward Filling- imputation:

Forward filling is when we fill the null values by their previously filled values.

df1['Age'] = df1['Age'].ffill();

df1['Age'] = df1['Age'].bfill();

age value would get updated in this way.

------------------------------------------------------------------------------------------------------------------------------------------

TIME-SERIES FORCASTING ALGORITHMS are also there for filling missing values but they are advanced, so we can actually ignore them and focus more on ML model. In TS we basically do predictions and fill the null values and this process is done recursively.

**2)FEATURE SCALING**:

Rescale the values in the features. In feature scaling we convert the scale of different measurement into a single scale. It standardize the whole dataset in one range. It happens when we have independant var or features that are different in terms of ranges or unites of the feature.

e.g:

weight W\_ES

85 (highest) 85/85=1

80 80/85=0.95

75 75/85=0.88

70 (lowest) 70/85=0.82

- it has same importance same for 70 which was lowest but they are now not in tens but in (0-1) which are most important and useful when u build your model. It is more imp in model buidling part however, in eda it is most imp.

MOST IMPORTANCE FEATURE SCALING TECHNIQUES ARE:

1) STANDARDIZATION: Z= (X-mean) / SIGMA.

2) NORMALIZATION: X(norm)= (X-Xmin) / (Xmax - Xmin)

- normal distributed data standarization is more imp, for opposite normalization is helpful.

**Python code for feture scaling:**

------------------------------------------------------------------------------------------------------------------------------------------

pip install -U scikit-learn //lib for feature scaling

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

df = pd.read\_csv('/content/Churn\_Modelling.csv')

df.info()

getting the .csv file data

from sklearn.processing import StandardScaler

from sklearn.processing import MinMaxScaler

df.head()

df.describe().round(2)

Normalization for data:

new\_df = pd.DataFrame(df,columns = ['Age','Tenure'])

new\_df['Age']=new\_df['Age'].filna(new\_df['Age'].mean())

scaler = MinMaxScaler() //Initiating the MinMaxScaler() func

normalized\_df = scaler.fit\_transform(new\_df)

print(normalizef\_df)

STANDARIZATION:

scaler = StandardScaler() //Initiating the StandardScaler() func

standardized\_df = scaler.fit\_transform(new\_df)

print(standardized\_df)

------------------------------------------------------------------------------------------------------------------------------------------

**3)OUTLIER TREATMENT:**

outliers are most extreme values of data.

Spot outliers:

1)Boxplot.

2)Histogram.

3)Scatter plot

4)Z-score

5)Inter quartile range

Handle Outlier using follwoing methods:

1)Remove the outlier.

2)Replace outlier with suitable values by using following methods:

I) Quantile

II) Inter quartile range

3) Use that ML model what are not sensitive to outliers.

4)Like: KNN, Decision Tree, SVM, Naive Bayes, Ensemble method.

**OUTLIER PYTHON CODE:**

**1) 3-SIGMA TECHINQUE:**

------------------------------------------------------------------------------------------------------------------------------------------

import numpy as np

import matplotlib.pyplot as plt

import statistics

import pandas as pd

from google.colab import drive

drive.mount('/content/drive')

data = pd.read\_csv('/content/drive/MyDrive/AI, Data Science & Analytics/raw\_sales.csv')

data.head(5)

//Function for outlier detection:

def find\_anomalies(data):

#define a list to accumlate anomalies

anomalies = []

# Set upper and lower limit to 3 standard deviation

random\_data\_std = statistics.stdev(data)

random\_data\_mean = statistics.mean(data)

# 3-standard deviation

anomaly\_cut\_off = random\_data\_std \* 3

lower\_limit = random\_data\_mean - anomaly\_cut\_off

upper\_limit = random\_data\_mean + anomaly\_cut\_off

# Generate outliers

for outlier in data:

if outlier > upper\_limit or outlier < lower\_limit:

anomalies.append(outlier)

return anomalies

data.price

- outilers logic defined above is anything > mean+3(sigma) or < mean+3(sigma) are outliers.

list\_1 = find\_anomalies(data['price'])

len(list\_1)

len(data)

data.price.skew() //to see the distribution of data, it is around 4 in this case which means data is not standard distributed

import seaborn as sns

sns.kdeplot(data.price)

data['price\_transformed'] = np.log(data.price)

data.price\_transformed.skew() //this time it gets around 0.9% just bcuz we applied logarithmic distribution to data.

------------------------------------------------------------------------------------------------------------------------------------------

**2) BOX PLOT:** //better as compare to other visual analytics. (such as scatter plots)

PYTHON CODE:

------------------------------------------------------------------------------------------------------------------------------------------

list1 = [43, 54, 56, 61, 62, 66, 68, 69, 69, 70, 71, 72, 77, 78, 79, 85, 87, 88, 89, 93, 95, 96, 98, 99, 99]

len(list1)

25

max(list1)

99

min(list1)

43

import statistics

statistics.mean(list1)

76.96

sorted(list1)

[43,54,56,61,62,66,68,69,69,70,71,72,77,78,79,85,87,88,89,93,95,96,98,99,99]

To find the 90th percentile for these (ordered) scores, start by multiplying 90 percent times the total number of scores, which gives 90% ∗ 25 = 0.90 ∗ 25 = 22.5 (the index). Rounding up to the nearest whole number, you get 23.

list2 = sorted(list1)

Hence, 98 is the 90th percentile for this dataset

Now say you want to find the 20th percentile. Start by taking 0.20 x 25 = 5 (the index); this is a whole number, which tells you the 20th percentile is the average of the 5th and 6th values in the ordered data set (62 and 66).

so, 20th percentile is 62+66/2 = 64

The median (the 50th percentile) for the test scores is the 13th score: 77.

------------------------------------------------------------------------------------------------------------------------------------------

**4) HANDLE INVALID VALUE:**

1-Encode Unicode property.

2-Covert incorrect data types. (correct the incorrect data types to the correct data types)

3-Correct values that go beyond range. (if some values are beyond logical range)

4-Correct wrong structure. (values that don't follow a defined structure can be removed)

DIFFERENT TYPES OF DATA:

QUALTITATIVE QUANTATIVE

nominal ordinal Discrete Continuous

(categorical) (order) (finite, whole) (infinite)

7 TYPES OF ANALYSIS:

- based on type of data we decide which analysis we use.

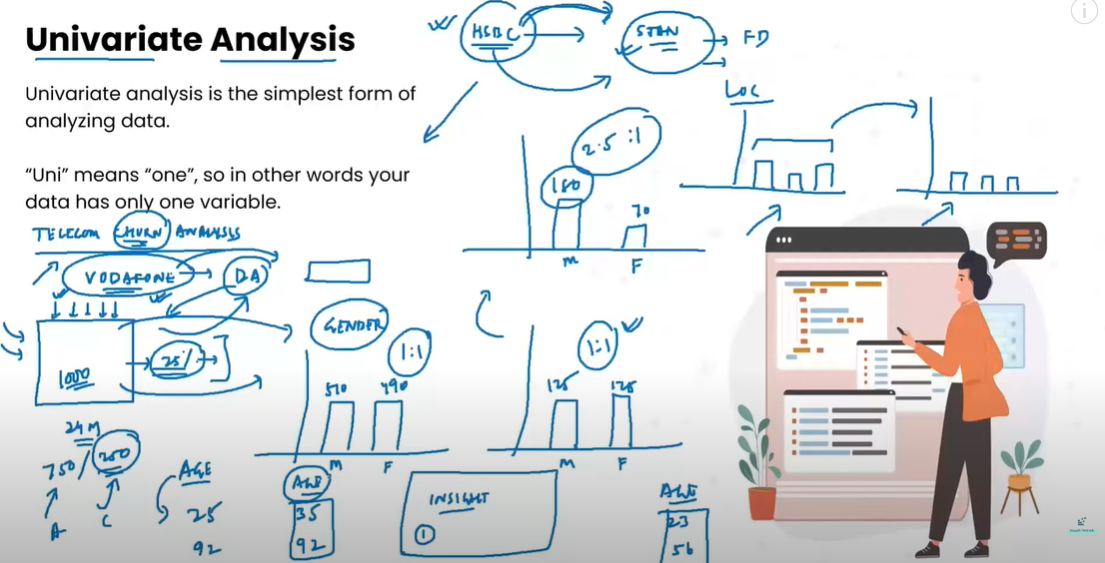
Univariate Analysis:

- deals with only one variable.If u are with numerical data we get to know about the statistical data and how the spread of data is.For categorical we get to know the distribution of the our data.

EXAMPLE:

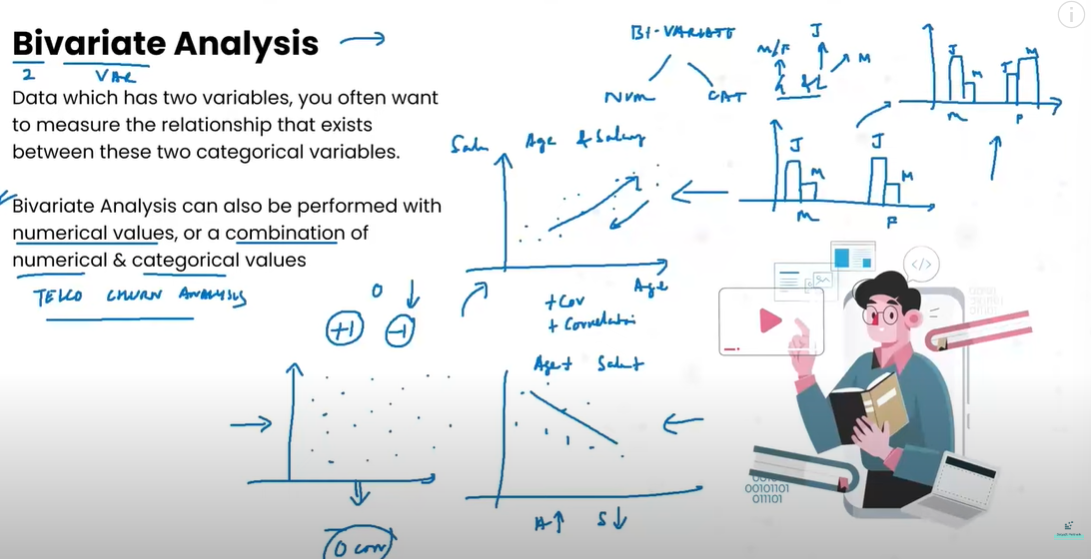
we have 1000 people data and 25% are churn (people who change their jobs). 1000 people 25% means 750 active and 250 churn.

we will analyse the columns one by one.Lets say we take gender column first we will see the gender for all customers analyze it and then for churn 25% people and compare them to each other to get insights.



Bivariate Analysis:

2 variables data falls into bivariate. It can be done for both numerical and categorical data. It could be a combination of numerical and categorical values.

EXAMPLE: 

Multivariate Analysis:

Data which has more than two variables we usually perform eda to measure relationship that exists between these features.

Numerical Analysis:

When dealing with one variable we might be interested in knowing their statistical information. For multiple variables we might be interested in knowing their correlation with each other. For plotting scatter plots for each variable we use corr() correlation func which comprises of heatmap which creates graph of 100 by 100 var in both x and y axis so, in this way multiple variables can be deal. Heatmap will give us an idea of which variable has which +ve or -ve relation with each other.

Derived Metrics:

Creates a new variable from the existing variable to get a insightful info from the data.

Eg: making new column by multiplying 2 columns

Techniques for derived metrics:

Feature binning, Feature encoding, From domain knowledge, Calculated from data.

Feature Binning:

It converts or transform continuous/numeric variable to categorical variable. It also helps to identify missing values or outliers.

Types of binning:

Surpervised Unsupervised

It transforms continuous/numerical It transforms continuous/numerical

variable into categorical value variable into categorical value

without taking dependent variable without taking dependent variable

* Equal width binning - Entropy based binning
* Equal frequency binning

Only required when we are preparing data for EDA.

Feature Encoding:

A technique to transform categorical data into numerical data. Since computers don’t understand the categorical data.

Types of Feature Encoding:

Label encoding: by assigning numerical values to each of the categorical data. Priority of numbering can be a issue.

One-Hot encoding: by assigning 1 and 0 to represents presents and absents. When independent variables are nominal.

Target Encoding: we create the average of the dependent variable for each category and replace the category.

Hash Encoder: it represents categorical independent variables using the new dimensions.