Feature Engineering

Missing Value Handling:

→ To check missing values in a dataframe, we can use df.isnull().sum() which will give us the number of missing values in each column.

→ Also we can plot sns.heatmap(df.isnull) to visualize our missing values in the dataframe.

We can handling missing values by deleting rows and columns but we have to be thinking logically that, In a row, not all the columns will have nan values. If there is some rows where maximum colums have nan values, we can delete those rows. Same goes in terms of columns. But Before deleting, we have to be very careful about how much crucial the column is for the data.

Also without deleting anything, we can manage the missing values by imposing some techniques.

① Mean value impulation :

If your column is normally distributed, you can fill your nan values with column.mean() values.

② Median value imputation:

If your column data is not normally distributed rather it is skewed in that case, you can replace your nan values with column.median() value.
So, when you have outliers in dataset, using median will be more appropriate.

③ Mode value imputation:

If your column data is categorical data not numerical, in that case you can replace your nan values with column.mode() value.

Handling Imbalanced Dataset:

In our classification supervised Learning, suppose a scenario where

Input feature 1	Input feature 2	Output Feature
3	4	Yes
9	6	Yes
2	5	Yes
. . .	. . .	. . .
8	7	Yes
10	9	No

Yes / no
ratio = 9:1

90% value in output column is Yes and 10% is No. This called an imbalanced dataset. With this dataset, the model you will be trained, will be a dumb model which will predict 'Yes' maximum time.

To solve this imbalanced issue, we use two methods →

1) Upsampling 2) Down sampling

In upsampling we try to add some more data in the smaller ratio side to make ratio balanced.

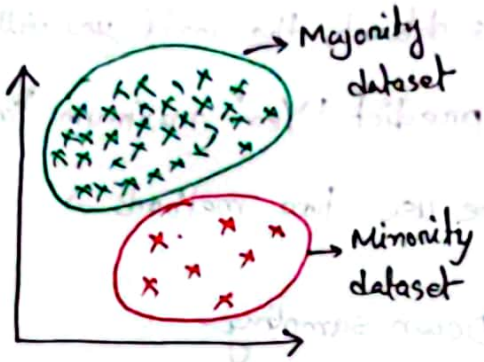
In down sampling we try to reduce the data in the larger ratio so that the ratio can be balanced.

The code for this has been uploaded to github

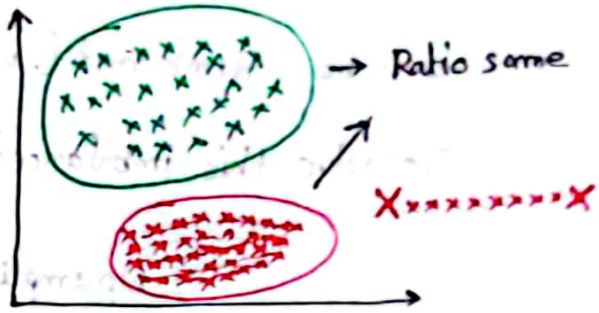
SMOTE: (Synthetic Minority Oversampling Technique)

It is an another upsampling technique to balance imbalanced dataset. We can do this by a SMOTE algorithm which can be found in Sklearn.

SMOTE Upsampling technique:



Before SMOTE



After SMOTE

What SMOTE algorithm does is, it takes two points and started putting more points between the straight lines of the two points. Doing this iteratively it upscale the minority elements and make a balance in the dataset.

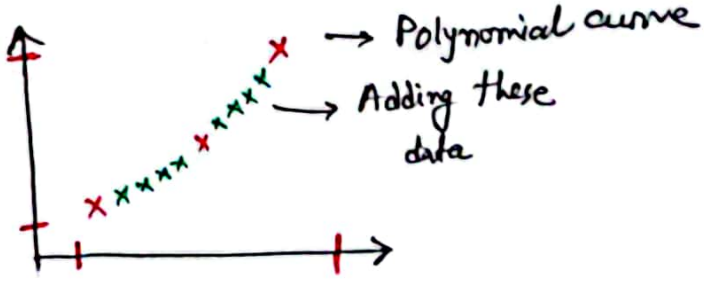
The coding part is uploaded in github.

Data Interpolation:

It is the process of estimating unknown values within a dataset based on the known values.

Suppose you have a dataset,

Before interpolation



After Interpolation

There can be different interpolation techniques like

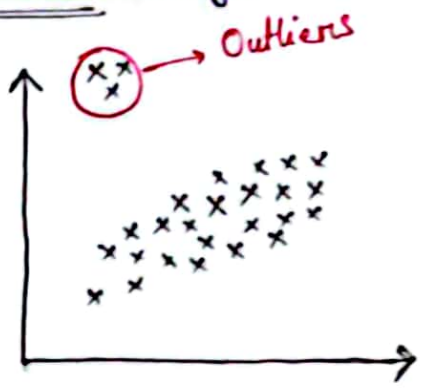
→ Linear Interpolation

→ Cubic Interpolation

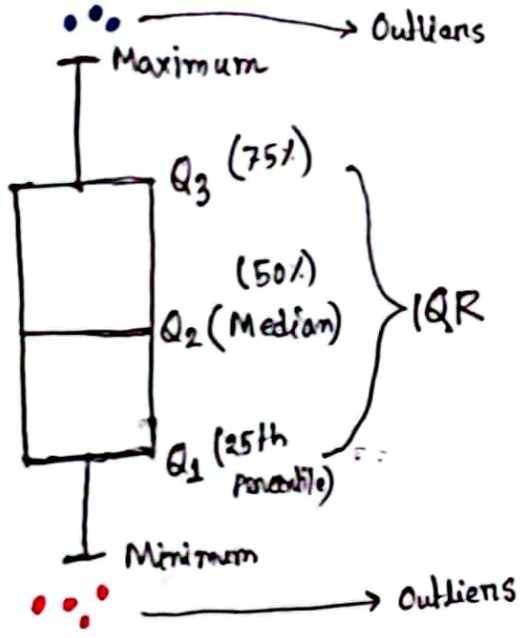
→ Polinomial Interpolation

The coding part of using this interpolation technique is uploaded to github.

Handling Outliers: (Finding)



We can easily detect outliers by plotting boxplot. In the boxplot we have



Data points that will be < Minimum and > Maximum can be considered outliers.