ML-Notes

Index—

1. Types of variables
2. Data cleaning
3. Missing value

. Backward filling, forward filling, mode filling

. Scikit learn

1. One hot encoding

. Label encoding, Ordinal encoding

1. Outlier

. Remove outlier-iqr, z-score

1. Feature scaling

. Standardisation

. Normalisation

1. Duplicate Data

. Handling and Removing

. Replace Data and Change Data Type

1. Function transformer

. Backward elimination

. Forward elimination

1. Train test split
2. Regression analysis

. simple linear

. multiple linear

. polynomial

. Cost function

. mean square error

. mean absolute error

. root mean squared error

. Regularization

. lasso

. ridge

. Logistic Regression

. Multiclass Classification

. Confusion Matrix

1. Imbalanced Data
2. Naïve Bayes

ML

1. SUPERVISED
2. UNSUPERVISED
3. REINFORCEMENT

SUPERVISED

1. REGRESSION ANALYSIS
2. CLASSIFICATION ANALYSIS

**data types:**

1. numerical (nos.-int, float)-discrete(repeats), continuous (multiple nos. within a particular range)

2. categorical(string)-ordinal (order followed), nominal (not inter-related)

3. dates and times

4. mixed (numerical + categorical)-e.g. User id

**Data cleaning:** process of preparing data for analysis/ml/dl by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated or improperly formatted.

* Handling missing data
* Outlier detection and handling
* Data scaling and transformation
* Encoding categorical variables
* Handling duplicates
* Dealing with inconsistent data

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

Supervised learning problems :-

1. Regression-predict a no from infinitely many possible outcomes.
2. Classification-predicts category(need not be nos.)-small no. of possible outcomes.

Unsupervised learning-not telling the algo before hand but just giving data and telling to find a pattern from it :-

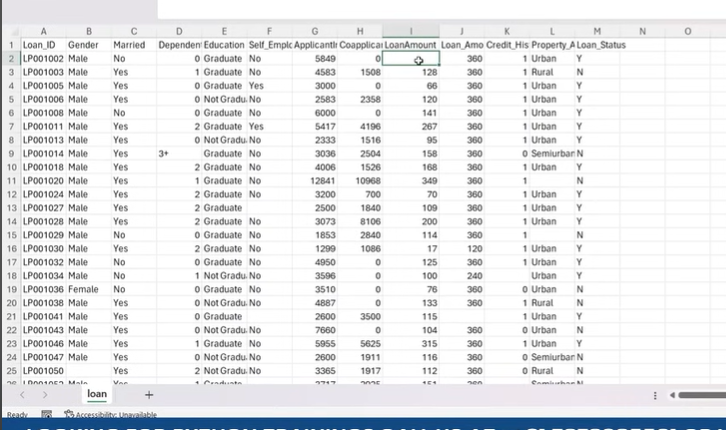
1. Clustering- group similar data points together into clusters
2. Anomaly detection- find unusual data points
3. Dimensionality reduction- compress data using fewer nos.

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

**Missing data**- ml doesn’t work on missing values as math formulas are used

Eg. Blank\*2=??

Identify: use pandas



import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

dataset=pd.read\_csv(r“D:\data set\video\loan.csv”)

dataset.head(10) #first 10 rows-NaN represents null value

dataset.shape #gives total rows and columns

dataset.isnull() #null values are represented as true

dataset.isnull().sum() #each column total null values are displayed

dataset.isnull().sum().sum()

#to understand the pattern of null values, we calc percentage null values

(dataset.isnull().sum()/dataset.shape[0])\*100 #column percentage

(dataset.isnull().sum().sum()/dataset.shape[0]\*dataset.shape[1])\*100 #total percentage

**#not null-**

dataset.notnull().sum()

**#graph representation-**

sns.heatmap(dataset.isnull())

plt.show() #white lines=null values

**#if more than 50% missing value then don’t use that dataset-check-**

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

dataset=pd.read\_csv(“loan.csv”)

dataset.head(4)

dataset.shape

dataset.isnull().sum()

sns.heatmap(dataset.isnull())

plt.show()

**#deleting missing value-**

#deleting column

dataset.drop(columns=[“credit\_history”,”…“], inplace=True)

#deleting row

dataset.dropna(inplace=True) #deletes all missing values and all rows as well

#handling missing values

#not so correct method-

dataset.isnull().sum()

dataset.fillna(10).head(10)

dataset.fillna(10)

**#more appropriate method-**

**#Data-num/categorical ie string**

**#Backward filling/forward filling/mode filling**

dataset.info()

dataset.fillna(method=”bfill”) #backward filling

dataset.fillna(method=”ffill”, axis=1) #forward filling, axis=1=>column wise filling, axis=0=>row wise

#forward column value gets copied in nan state of next column-ffill, axis=1

#column value gets copied to previous column-bfill, axis=1

#similarly for row

**#mode filling-**

dataset[“Gender”].mode()[0]

dataset[“Gender”].fillna(dataset[“Gender”].mode()[0] inplace=True) #particular column

#loop-

dataset.select\_dtypes(include=”object”).isnull().sum()

for i in dataset.select\_dtypes(include=”object”).columns:

dataset[i].fillna(dataset[i].mode()[0],inplace=True)

**Missing Value-Scikit Learn**

import pandas as pd

import seaborn as sns

import matplolib.pyplot as plt

dataset=pd.read\_csv(“loan.csv”)

dataset.head(3)

dataset.isnull().sum()

dataset.info()

dataset.select\_dtypes(include=”float64”).columns #gives column names in which missing value is present=\*\*

from sklearn.impute import SimpleImputer

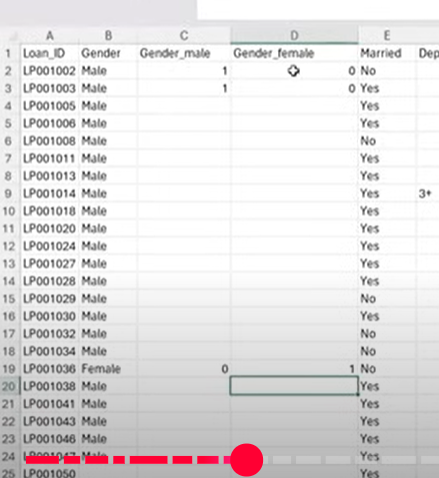
si=SimpleImputer(strategy=”mean”)

ar=si.fit\_transform(dataset[[\*\*]]) #gives array

new\_dataset=pd.DataFrame(ar,columns=dataset.select\_dtypes(include=”float64”).columns)

new\_dataset.isnull().sum() #this should give 0

**One hot encoding and dummy variables-** encoding=Data conversion;categorical to numerical-limited data is present then one hot

****

import pandas as pd

dataset=pd.read\_csv(“loan.csv”)

dataset.head(3)

dataset.isnull().sum()

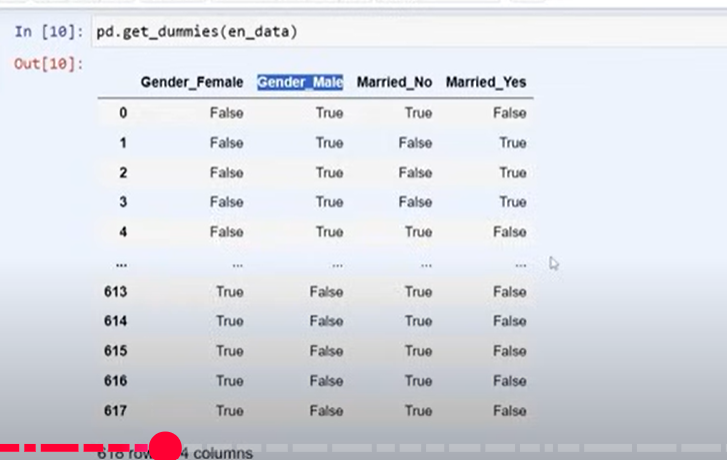
dataset[“Gender”].fillna(dataset[“Gender”].mode()[0],inplace=True)

dataset[“Married”].fillna(dataset[“Married”].mode()[0],inplcae=True)

en\_data=dataset[[“Gender”,Married”]] #seperates gender and married column

pd.get\_dummies(en\_data)

#result-



pd.get\_dummies(en\_data).info() #gives type as Boolean, so, we have to again convert it to num. **so better use scikit learn-**

**#using scikit learn**

from sklearn.preprocessing import OneHotEncoder

ohe=OneHotEncoder()

arr=ohe.fit\_transform(en\_data).toarray() #gives sparse matrix-matrix filled with 0 and 1

pd.DataFrame(arr,columns=["Gender\_Female","Gender\_Male","Married\_No","Married\_Yes"]) #encoded data

#after this data is exceeded

#so use- ohe=OneHotEncoder(drop=’first’)

#this deletes “Gender\_Female” and “Married\_No” column

#so syntax-

from sklearn.preprocessing import OneHotEncoder

ohe=OneHotEncoder(drop='first')

arr=ohe.fit\_transform(en\_data).toarray()

pd.DataFrame(arr,columns=["Gender\_Male","Married\_Yes"]) #encoded data

Categorical data-1. Ordinal data 2. Nominal data

**Ordinal Encoding-**ordinal data-some relation/order between data

**Label Encoding-**nominal data-no interconnection between data—

import pandas as pd

df=pd.DataFrame({"name":["cow","cat","dog","black"]})

df

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

df["en\_name"]=le.fit\_transform(df["name"])   #fit-trains model;transform-ensure consistency in data preprocessing between training and testing datasets

df

dataset=pd.read\_csv("loan.csv")

dataset.head(3)

la=LabelEncoder()

la.fit(dataset["Property\_Area"])

la.transform(dataset["Property\_Area"])

dataset["Property\_Area"].unique()

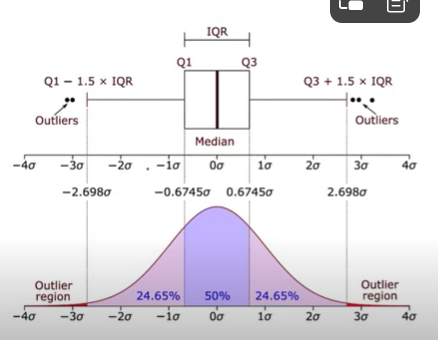
#to replace in orginal dataset

dataset["Property\_Area"]=la.transform(dataset["Property\_Area"])

**Ordinal Encoding**—(preferred)-scikit learn, map function

(vs code)

**Outlier**-when data goes beyond the given range of data-eg. 5,6,7,8,100



**#outlier detection-**

import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

dataset=pd.read\_csv("loan.csv")

dataset.head(3)

dataset.info()

dataset.describe()

#see outlier using box plot

plt.figure(figsize=(15,5))

sns.boxplot(x="CoapplicantIncome",data=dataset)

plt.show()

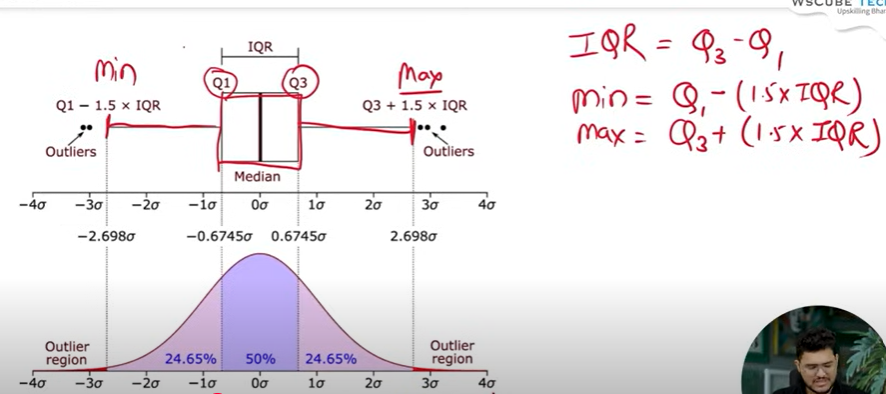
sns.boxplot(x="ApplicantIncome",data=dataset)

plt.show()

sns.displot(dataset["ApplicantIncome"]) #distribution plot

plt.show()

**#outlier removal-**



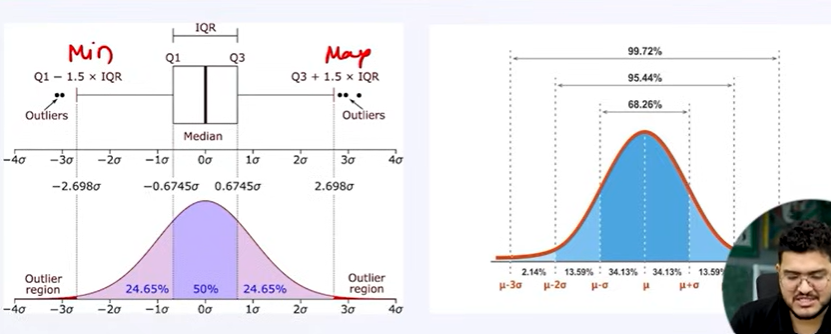
#outlier removal-iqr(Interquartile Range=q3-q1) technique

#Q1 (First Quartile): 25th percentile (middle of the lower half)

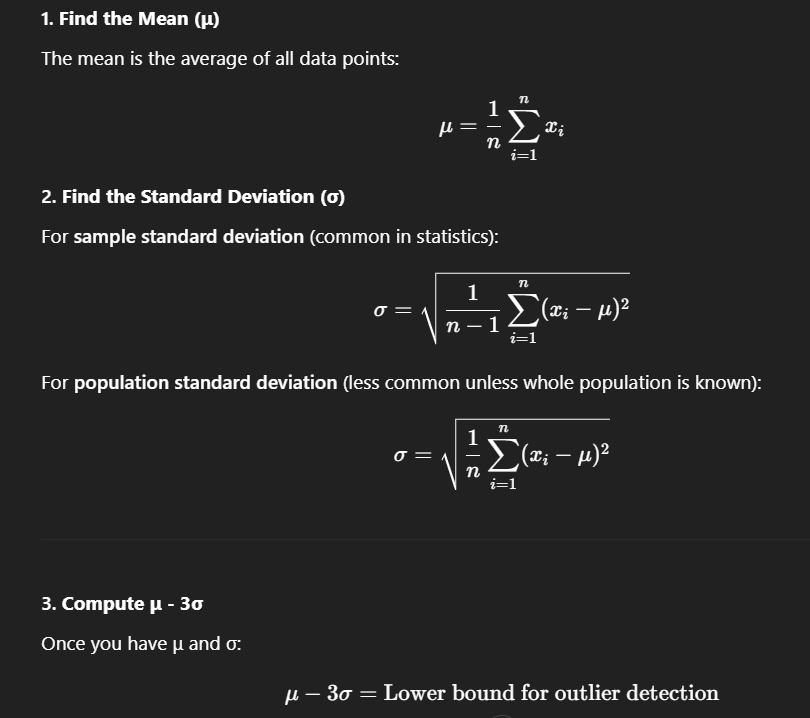
#Q3 (Third Quartile): 75th percentile (middle of the upper half)

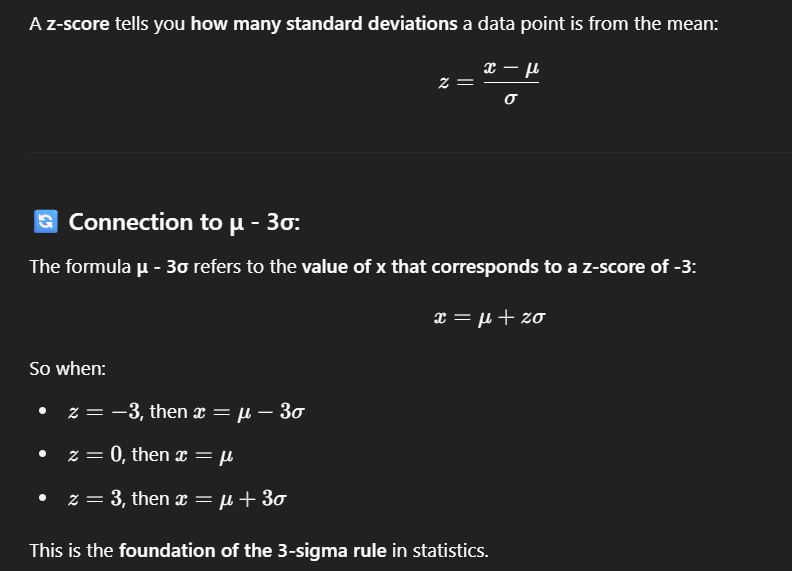
(vs code)

#using z-score



Data after 3rd sd(μ ± 3σ) is considered as outlier





So, any data out of range of z=-3 to +3 is treated as outlier

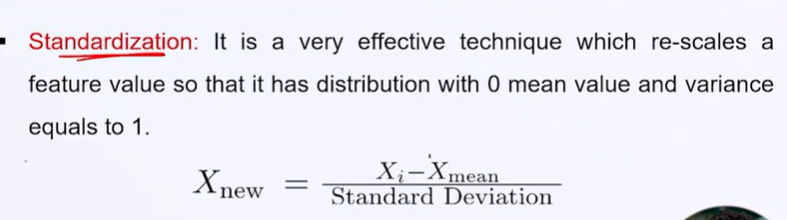
(vs code)

**Feature Scaling Techniques-**

#in this, outliers are not removed but their magnitude reduces so doesn’t affect much

#1. Standardisation--

#reduces magnitude of data;nature of data remains same even after scaling

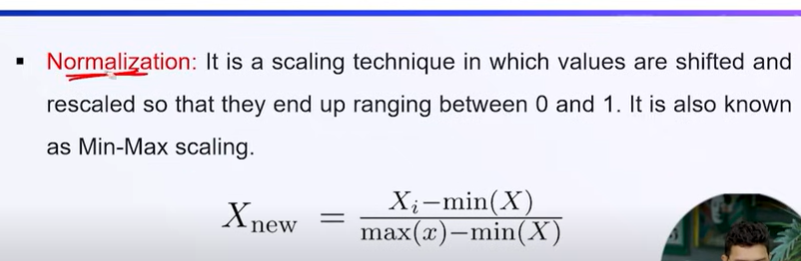


Mean(Xnew)=0 and Variance(Xnew)=1

#2. Normalisation=min/max technique--

#data reduces acc to min and max value of data; nature remains same

#data gets reduced and lies within range 0 to 1



**Handling and Removing Duplicate Data-(vs code)**

**Replace Data and Change Data Types-**

#data of mixed data type is present

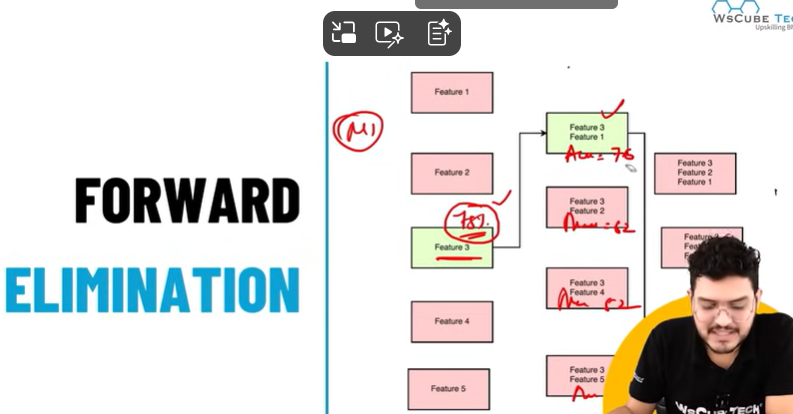
**Function Transformer-**

#non-normal distribution data to normal distribution data

**Feature selection technique**

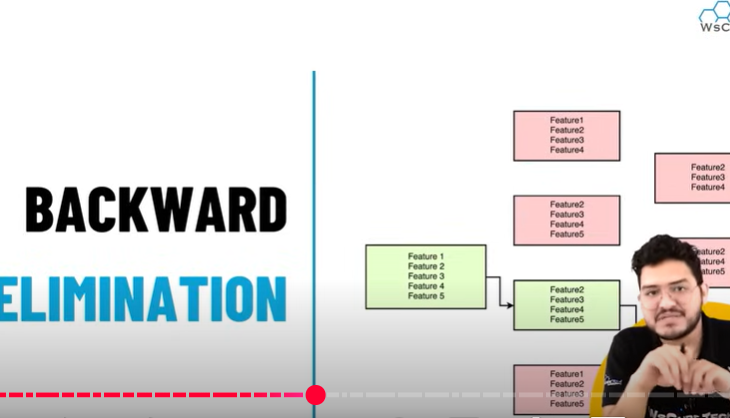
A feature is an attribute that has an impact on a problem or is useful for the problem, and choosing the important features for the model is known as feature selection.(having domain knowledge is better)

1. Forward elimination-



1 with highest accuracy-> 2 ke groups-> highest among all groups and original-> 3 ke groups…

1. Backward elimination-



Group of 5-> remove one(4)->remove one(3)…always choose feature with highest accuracy

**Training and Testing - for unsupervised learning**

If accuracy good after testing then deploy

Build model=train

Check accuracy that how well model will work with new dataset=test

**SUPERVISED LEARNING**

**1.Regression Analysis 2.Classification Analysis**

Regression-

Use when outcome is continuous in nature

#1. linear algo-input feature and output are in linear relation

#->simple linear, multi-linear, lasso, ridge

#2. non-linear algo

#->polynomial, decision tree, random forest, support vector, k-nearest neighbour algo

**simple linear regression**

input(independent variable) has only one data

eg. cgpa and package-predict package based on cgpa

eq. y=mx+c

      y=dependent variable

      x=independent variable

      m=slope/gradient/coefficient

         m=(x2-x1)/(y2-y1)

         m=-ve, angle>90

         m=+ve, angle<90

         m=0, angle=90

      c=intercept

         c=+ve..line cuts +ve y axis

         c=-ve..line cuts -ve y axis

**Multiple Linear Regression**

#this is extension of simple linear regression as it takes more than one predictor variable to predict the response variable

#multiple inputs, single output

#eg. x1,x2,y

#   find relation between x1 and y and x2 and y

#   if these are individually linear(y increases as x increases) then ok-simple linear apply

#   if no linear relation then do feature selection

#equation- y=m1x1+m2x2+m3x3+...+c = eq of a plane

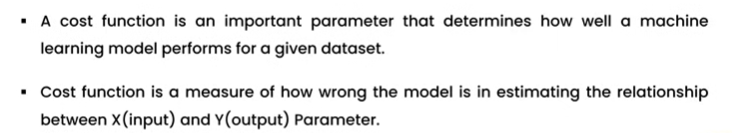
#eg. based on age and experience, we have to predict salary

**Polynomial Regression**

#Y = b0 + b1x1 + b2x1^2 + b3x1^3 +...bnx1^n

#seperate dependent and independent values

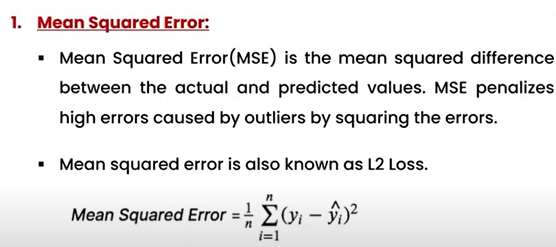
**Cost Function**

****

1. Regression cost fun- mean square error(MSE), root mean square error(RMSE), mean absolute error(MAE), R^2 Accuracy
2. Classification-

Binary classification cost function

Multi-class classification cost function- binary class entropy cost function, log loss function



Y=original value, Y^ is predicted, n=no. of rows

MSE is use to predict best fit line

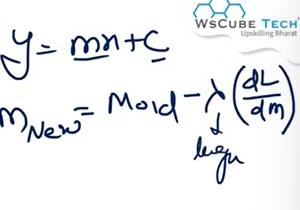
1.randomly create pred line

2.cal error btw line and data points

3.change line, bring closer to data points

4.repeat until you get min error

(some more cal related to diff…)-gradient decent



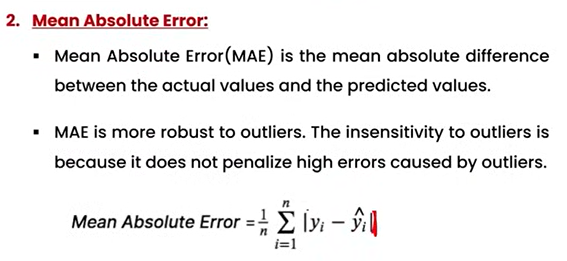
Lambda=learning rate

Problems-

MSE is robust with outlier-line gets shifted from original position to more towards outlier

So wrong prediction is obtained

Error is in sq



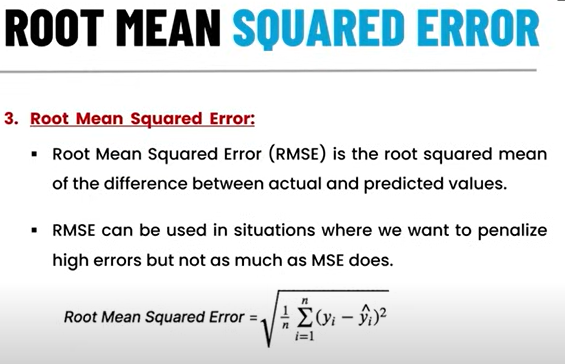
Mod is used

So, graph is not differentiable

Error is obtained in original format and not in square=no unit changes

Behaves well even with outliers-doesn’t shift much towards outlier

Since not differentiable(because of sharp curve of mod); differentiable only at surrounding points, so, not so perfect prediction



Finding best fit line(best value of m and c)-

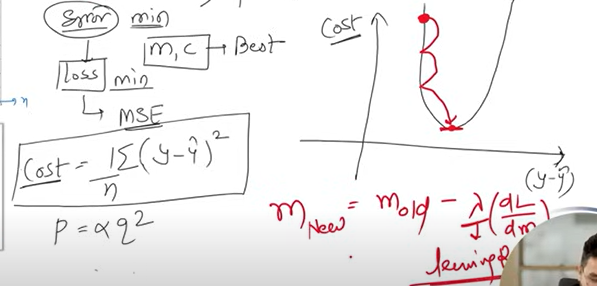
Error=diff btw original data point and predicted data point

Minimize error->more accurate line

Loss eq-

Mean square error loss(MSE)

Use gradient decent to achieve min point of parabola of cost



Lambda=learning rate

M=angle, c=intercept

L=y-mx-c

**Regularization-**form of regression that contains/regularizes or shrinks the coefficient estimates towards zero. This technique discourages learning a more complex or flexible model, so as to avoid the risk of overfitting. Reduces computational power.

**--Lasso Regularization(L1)**

. Used in **feature selection** using a shrinkage method also referred to as the penalized regression method.

. Lasso regression magnitude of coefficients can be **exactly zero.**

. Cost fun=loss+lambda\*sum(w)

Loss=sum of squared residual

Lambda=penalty

W=slope of curve

**--Ridge Regularization(L2)**

. is an extension to linear regression that introduces a regularization term to reduce model complexity and help prevent overfitting.

. Ridge regression is working value/magnitude of coefficients is **almost equal to zero.**

. Cost fun=loss+lambda\*sum(w^2)

**Classification Analysis**

* used to identify the category of new observations on the basis of training data.
* Programs learns from given dataset or observations and then classifies new observations into a no. of classes or groups
* Classes=targets/labels/categories

1. **Binary Classifier:** output is binary. Eg. True/false, yes/no
2. **Multi-class Classifier:** problem has more than 2 outcomes. Eg. Types of music

**Algorithms:**

Linear: logistic, support vector machines(svm)

Non-linear: k-nearest neighbours, svm(with kernel), naïve bayes, decision tree, random forest

**Evaluation**

Log loss/ cross entropy loss

Confusion matrix

Auc-roc curve

**Logistic Regression**

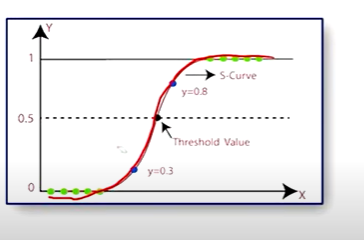
Data should be **linearly separable**

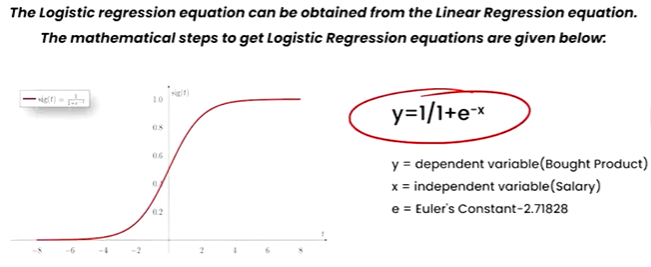
Outcome must be categorical or discrete value

1 binomial- 2 outcomes-eg. 0/1

2 multinomial-3 or more-eg. Cat/dog/sheep

3 ordinal-data classified in order-eg. s/m/l/xl

Sigmoid prediction 

 x=m1x1+m2x2+…+b = linear regression

If data is not linearly separable then use **polynomial feature**

**Multiclass Classification-**

Ovr method-converts data into one hot encoding-ie multi class gets converted to binary

Multinomial method

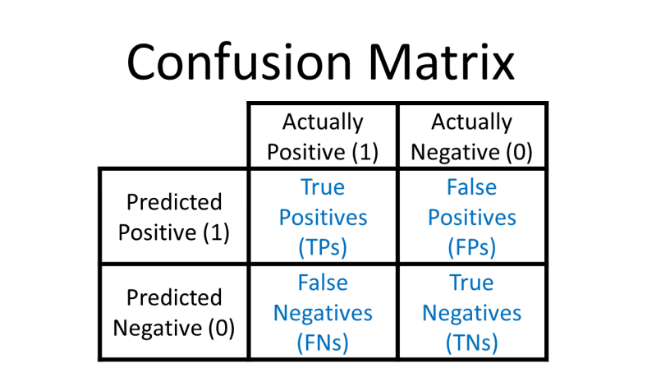
**Confusion/Error Matrix**

Simple and useful tool for understanding the performance of a classification model.

Helps you evaluate how well your model is doing in categorizing things correctly

Matrix consists of predictions result in a summarized form, which has no. of correct predictions and incorrect predictions.

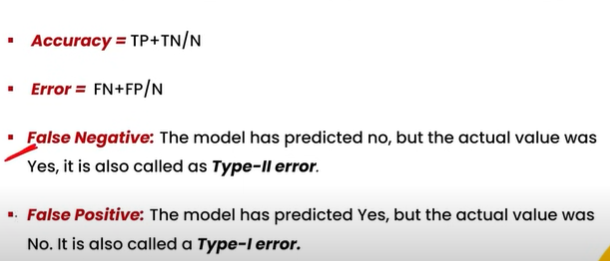
Creates matrix between y original and y predicted of testing data, so that we can know the difference.

TN=true negative 

FN=false negative

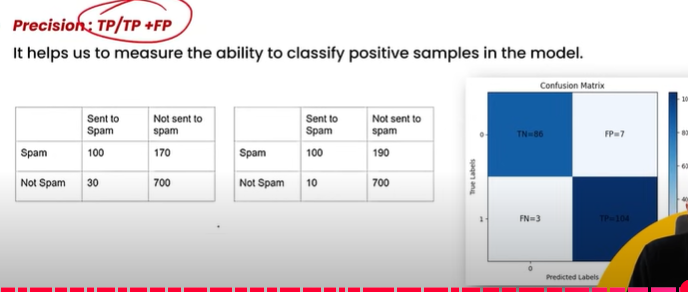
TP=true positive

FP=false positive



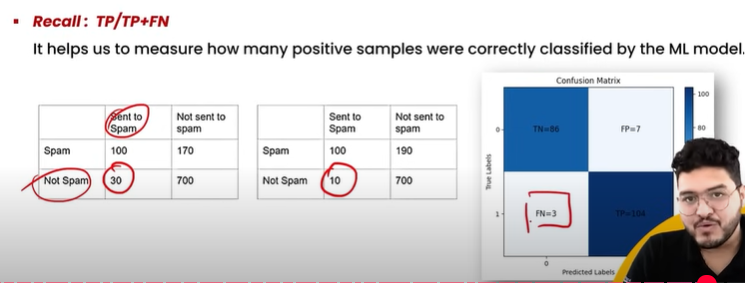
N=TP+TN+FP+FN

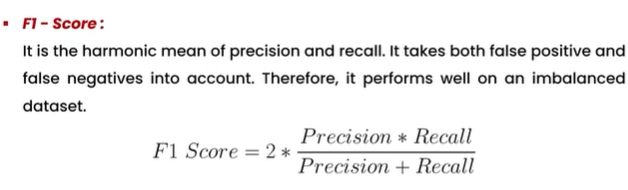
All 3 are ok-acceptable except FN

Precision: 

More precisio..less fn

Recall inversely proportional to fn





Precision, recall, f1 score high preferred

**IMBALANCED DATASET**

-a category occurs many times- dataset becomes biased

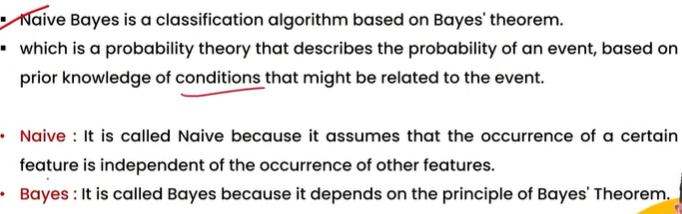
Fix:

-random under sampling-will reduce the majority of class so that it will have same no of as the minority.

-random over sampling-will increase the size of minority is inactive class to the class of majority class is active.

**NAÏVE BAYES ALGO**

-works on conditional prob



Prob=no. of favourable outcomes/total no. of outcomes- lies between 0 and 1

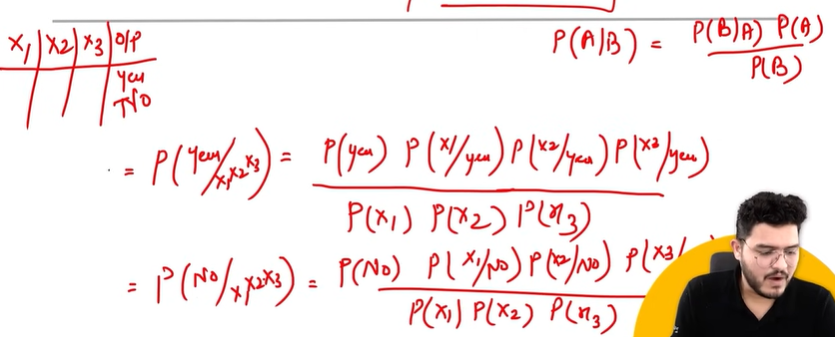
Event-possible outcomes

1. Dependent/Conditional prob
2. Independent prob

P(R or B) = P(R)\*P(B/R) = P(B or R); or=intersection

P(B or R)=P(B)\*P(R/B)

BAYES TH-P(A/B)=(P(B/A)\*P(A))/P(B)- prob of occurrence of A when event B has occurred.



Naïve bayes-

1. Gaussian-features that are continuous and have normal distribution
2. Multinomial-discrete data like text data, wher each feature represents the frequency of a term
3. Bernoulli-data represents binary features-one hot encoding