EDA

Univariate charts

Number: (distribution of number column)
List Plot, distplot, hist, box plot

category: [distribution of category column)
count-plot, bar chart, frequency Plot

Bivariate charts

Number v/s Number : [relative distribution in 2 column]
Scatter Plot, reg Plot

Number v/s category
Box Plot, strip, swarm, violin

category v/s category [Cross Tab.]
Grouped bar chat

Bivariate charts with third category (Hue)

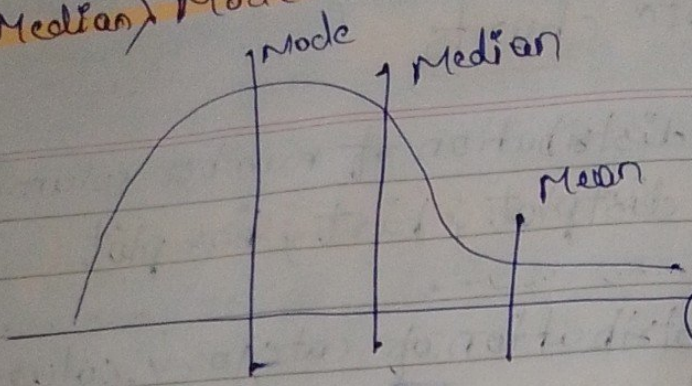
number v/s number v/s category
Scatter plot, reg plot, pair plot

category ~~number~~ v/s number v/s category
Box plot, strip, swarm, violin.

Series Data

Line plot, area plot.

Mean > Median > Mode

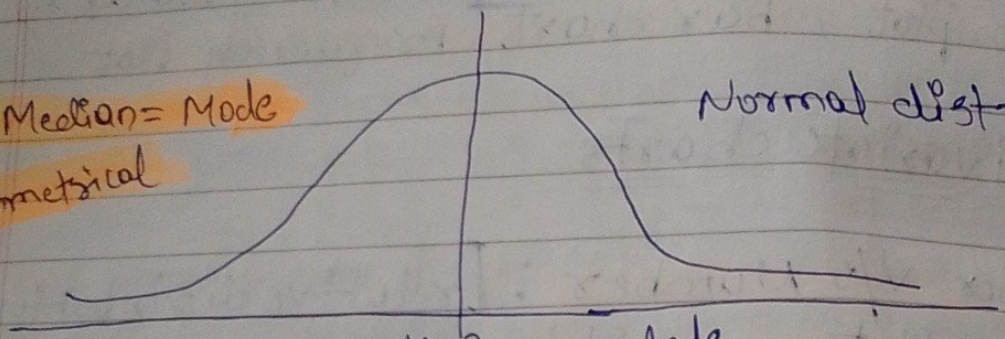


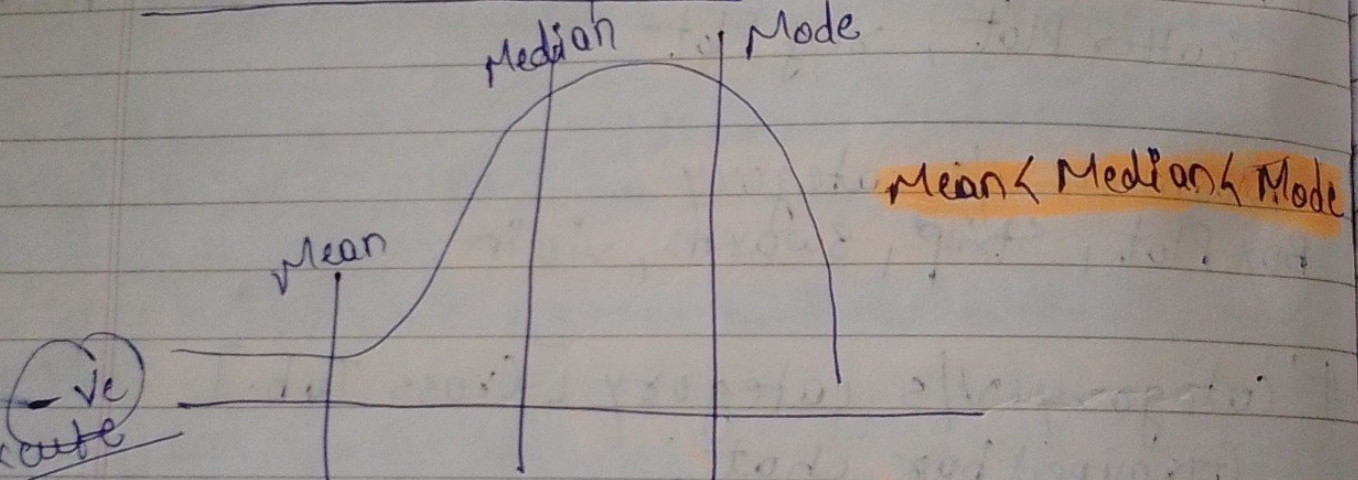
right excute

Mean = Median = Mode

Symmetrical

Normal distribution





Mean < Median < Mode

exoute

```
import scipy.stats as stats # statistical library.
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns.

test = pd.read_csv ('test_AbJT22l.csv')
train = pd.read_csv('train_vgrqXOR.csv')
sklearn.preprocessing import StandardScaler, MinMax Scaler
```

a probability to each pos...

combined = pd.concat([train, test]), ignore_index=True)
↳ दो data को combine कर दिया

combined.select_dtypes(include=np.number).columns
↳ find all numerical variables & numerical columns.

{Univariate} {Number}

num_cols = ['Item_Weight', 'Item_Visibility', 'Item_MRP', 'Item_Outlet_Sales']

```
nrows = 2      # no. of rows in the plot
ncols = 2      # no. of columns
iterator = 1   # Plot iterator

plt.figure(figsize=(10,6))
for i in num_cols:
    plt.subplot(nrows, ncols, iterator)
    sns.distplot(combined.loc[:, i])
    iterator = iterator + 1
plt.tight_layout()
plt.show()
```

bar/pai pash उपर line vala graph उसको देख k pata chalega ki graph में column कैसा kya है raha hai.

{Category}

combined.select_dtypes(include='object').columns

cat_cols = ['Item_Fat_Content', 'Item_Type', 'Outlet_Identifier', 'Outlet_Size', 'Outlet_Location_Type', 'Outlet_Type']

matical func that des...
Rand ...fnhlo.

```
nrows = 3
ncols = 2
repeater=1
plt.figure(figsize=(10,8))
for i in cat_cols:
    plt.subplot(nrows, ncols, repeater)
    combined.loc[:,i].value_counts().plot(kind='bar')
    repeater+=1
plt.tight_layout()
plt.show()
```

Bi-variate

Number v/s Number

```
nrows=2
ncols=2
iterator=1
plt.figure(figsize=(12,8))
for i in num_cols:
    plt.subplot(nrows, ncols, iterator)
    plt.scatter(combined.loc[:,i],
                combined.Item_Outlet_Sales)
    plt.title(i)
    iterator+=1
plt.tight_layout()
plt.show().
```

```
nrows=3
ncols=2
rep=1
plt.figure(figsize=(12,10))
for i in cat_cols:
    plt.subplot(nrows, ncols, rep)
    sns.boxplot(data=combined, x=i, y='Item_Outlet_Sales')
    plt.xticks(rotation=90)
    rep+=1
plt.tight_layout()
plt.show()
```

{Category v/s Number}

{Category v/s Category}

atical func^n that ...
and

```
# fix the item fat content
combined.Item_Fat_Content.replace(to_replace=['Low fat','LF',
                                  'reg'],
                                  value=['Low Fat','Low Fat','Regular'],
                                  inplace=True)
```

change the name, names change hoge.

```
# average sales
print("Mean Sales:", combined.Item_Outlet_Sales.mean())
```

Mean

```
# what is the 95% values of sales.
print("95% sales values:", combined.Item_Outlet_Sales.quantile
                                                    (0.95))
```

quantile

```
# Identify the product that sells most.
combined.Item_MRP.~~median~~mode()[0]
```

Mode

```
# Find the middle most element observation for the MRP
combined.Item_MRP.median()
```

Median

```
# Compare the Mean and Trimmed Mean of sales
print('mean:', combined.Item_Outlet_Sales.mean())
```

mean

```
trimmed_mean = stats.trim_mean(train.Item_Outlet_Sales,
                               proportiontocut=0.01)
print("Trimmed Mean", trimmed_mean)
```

```
mean : 2181.2889135703 2
Trimmed Mean : 2138.3762
```

Trimmed mean

what % Diary Items are in the data

```
print(combined.loc[combined.Item_Type=='Diary'].shape
      [0]/combined.shape[0])
```

0.07997747.

which product as the highest variation in sales

Standard deviation

```
combined.groupby("Item_Type")["Item_Outlet_Sales"].
                                              std().
```

what is the middle 50% of Item MRP

```
Q3 = combined.Item_MRP.quantile(0.75)
Q1 = combined.Item_MRP.quantile(0.25)
print('IQR Range:', Q3-Q1)
```

IQR

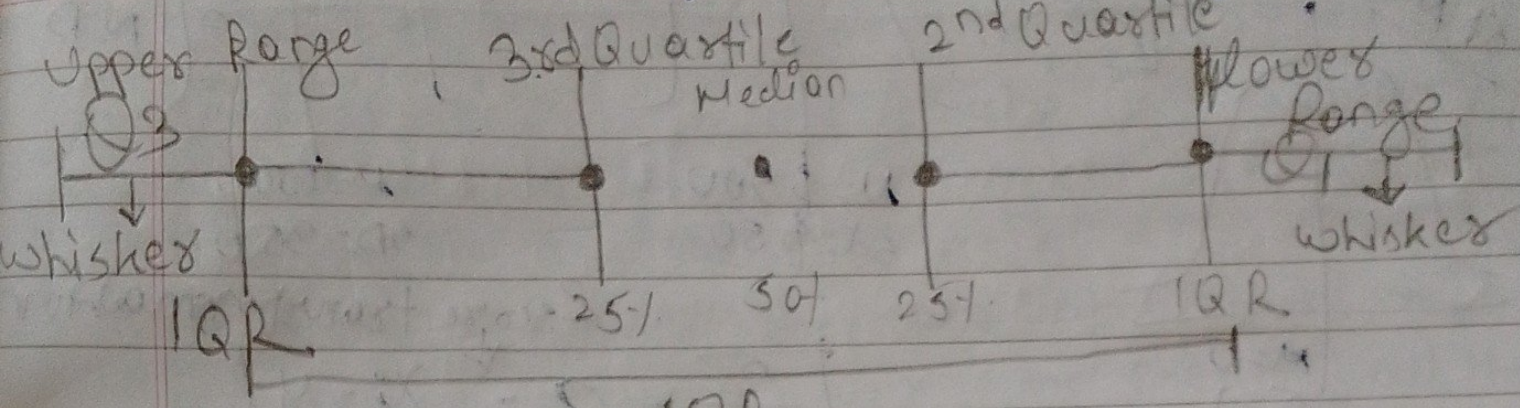
find the % of variation in item weight for each product

```
combined.groupby("Item_Type")["Item_Weight"]
for i in combined.Item_Type.unique():
    mean = combined.loc[combined.Item_Type==i,
           "Item_Weight"].mean()
    std = combined.loc[combined.Item_Type==i,"Item_Weight"].
                                              std()
    print("CV for",i, std/mean)
```

print

hogaya.

IQR = Q3 − Q1 (upper quartile Q3 − lower quartile Q1)



✱ data wo column kabhi include nhi kiya jata where SD standard deviation = 0 ✱

$$CV = \frac{\sigma}{\mu}$$

coefficient of variation

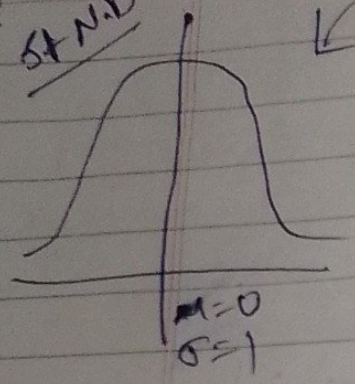
जिसका CV कम होगा उसको अच्छा player माना जाएगा, क्यूंकि वो stable जादा

ex. 2 player का

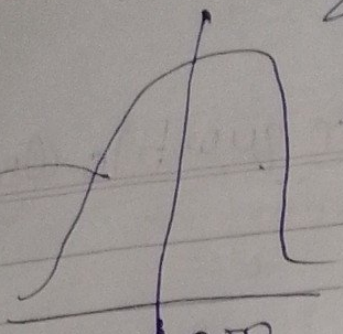
1	$\frac{1-1}{5-1} = 0$
2	$\frac{2-1}{5-1} = \frac{1}{4} = 0.25$
3	$\frac{3-1}{5-1} = \frac{2}{4} = 0.5$
4	$\frac{4-1}{5-1} = \frac{3}{4}$
5	$\frac{5-1}{5-1} = 1$ Min = 1, Max = 5

$$\text{Min Max scaler} = \frac{X_i - \text{Min}(X_i)}{\text{Max}(X_i) - \text{Min}(X_i)}$$

thematical func^n that
Random variable.

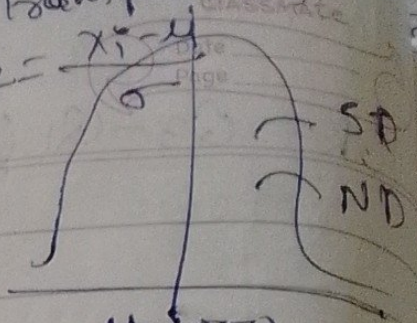


Standard Scaler



Zscore Transformation

$\frac{\sigma}{\mu}$ $z = \frac{x_i - \mu}{\sigma}$



Zscore transformation

$$\frac{\sigma}{\mu} = z = \frac{x_i - \mu}{\sigma}$$

Standard Scaler

Two Scaling techniques
1. Standard Scaler
2. Min/Max Scaler

$$\text{Min/Max Scaler} = \frac{x_i - \text{Min}(x_i)}{\text{Max}(x_i) - \text{Min}(x_i)}$$

1 $\frac{1-1}{5-1} = 0$

2 $\frac{2-1}{5-1} = 1/4 = 0.25$

3 $\frac{3-1}{5-1} = 2/4 = 0.50$

4

5

Min/Max Scaler वहाँ apply नहीं होगा जहाँ Outliers present हो।

SCALING

(b) Standard Scaler is one of the popular techniques used for scaling in the Data Science Fraternity It involves calculating the mean and std of all the numerical columns and then using Z score to convert the data into Standard Normal Distribution where the Mean of all the columns is equal to 0 and SD is 1 standard Deviation is 1

(c) Normalization

(b) It is another technique used to scale the data in such a way where the min of each column is equal to 0 and the max of each column is equal to 1.

(b) Since, it takes the range in denominator thus, it is very much influenced by the presence of the outliers in the data. Therefore Min Max scaler is not a very effective technique in scaling the data.

Note(1) Standard Scaler is quite versatile in nature as it can handle the presence of Outliers well

Note(2) However, in the presence of outliers in data one must apply ROBUST SCALER to scale the data.

tical func^n that describes the p...
ndard variabl...

① from sklearn.preprocessing import StandardScaler, MinMaxScaler

② sc = StandardScaler() # Machine Instance

Scale the Data
sc = (Xi - mean) / sd ✓ FORMULA

✓ # .fit() will only learn the mean & std.

✓ # .fit_transform() will not only learn the mean & std but also convert/transform

scaled = sc.fit_transform(pd.DataFrame(combined.item...
(=error it is 1D convert it into 2D (Dataframe))

Note: SC takes data in 2D format and hence,
I need to supply the Data Frame.

Apply the standard scaler on the whole Data

print(pd.DataFrame(scaled, columns=["ZScore"]).mean()
print(pd.DataFrame(scaled, columns=["ZScore"]).std()

ZScore -1.391374e-16
dtype Float64
ZScore 1.000035
dtype Float64

10^{-16} = Near = 0

```
#Apply Standard Scaler on the whole Data
sc = StandardScaler()
scaled = sc.fit_transform(combined.loc[:, num_cols])

pd.DataFrame(scaled, columns = num_cols).describe()

    #Min Max Scaler()
nmax = MinMaxScaler()
pd.DataFrame(mmax.fit_transform(combined.loc[:,
                num_cols]), columns =num_cols).min()
```

output

```
Item_Weight          0.0
Item_Visibility      0.0
Item_MRP             0.0
Item_Outlet_Sales    0.0
dtype: float64
```

```
def minmax(df, xi):
    return ((df[xi]-min(df[xi]))/(max(df[xi])-
            min(df[xi])))
minmax(combined, "Item_MRP").describe()
```

output

```
count    14204.000000
mean         0.465686
std          0.268529
min          0.000000
25%          0.266244
50%          0.470958
75%          0.656055
max          1.000000
```

atical func^n that describes the p... random variable.

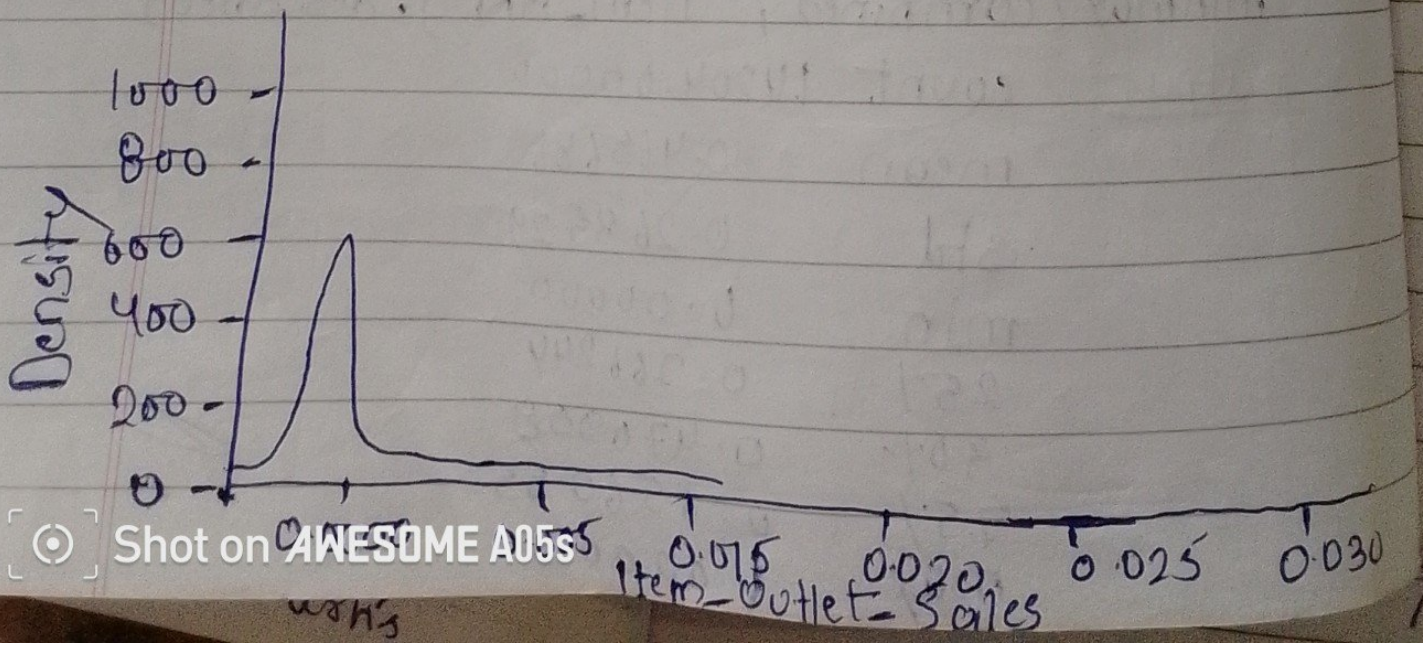
TRANSFORMATION

Why Transform?

- The Puppose of Transformation is to reduce the SKEWNESS in the data
- Most of the statistical Models have an assomption about the data that it should be Normally Distributed / Gaussian Curve
- We can apply some of the transformation techniques to reduce the skewness in the data.
- Those Techniques are as follows.
  - Log Transformation
  - SQRT Transformation
  - Cube ROOT
  - Reciprocal
  - Box-Cox
  - Yeo-Johnson.

APPLY RECIPROCAL TRANSFORMATION

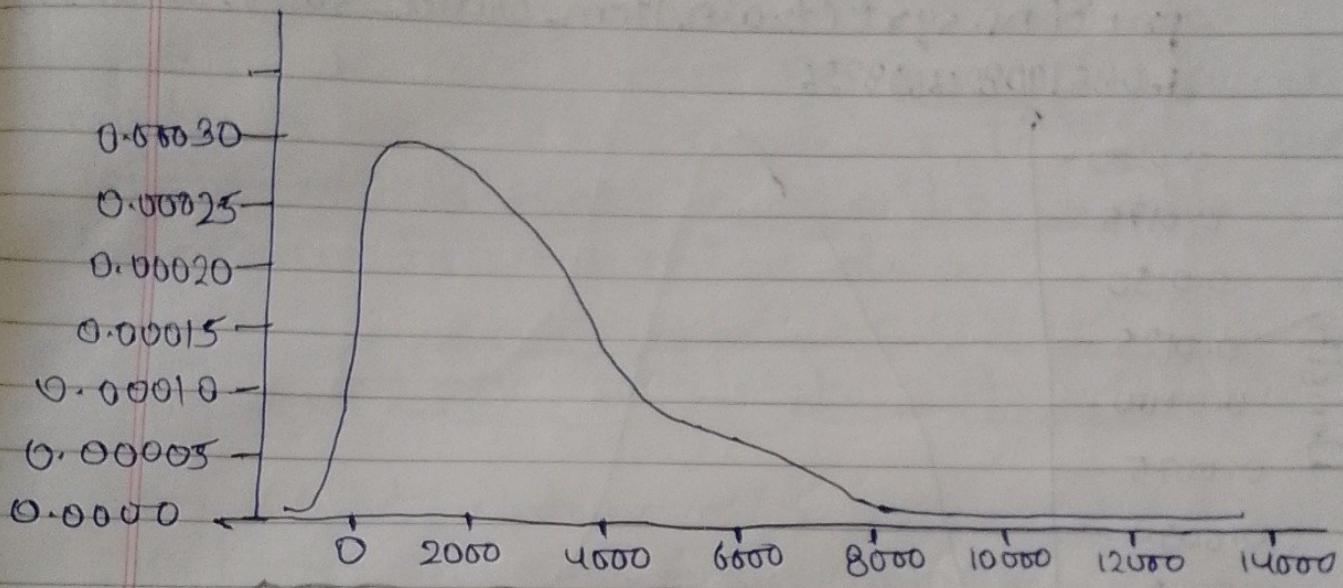
```
sns.distplot(np.reciprocal(train.Item_Outlet_Sales))
print(np.reciprocal(train.Item_Outlet_Sales).skew())
```



#WITHOUT TRANSFORMATION

```
sns.distplot(train.Item_Outlet_Sales)
print(train.Item_Outlet_Sales.skew())
```

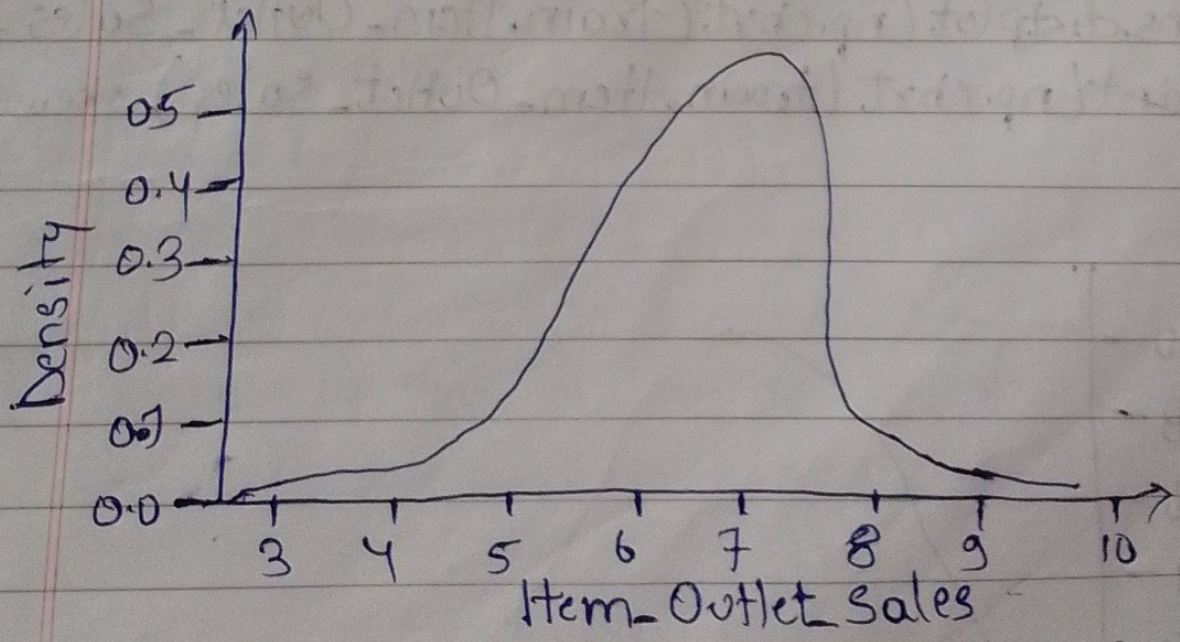
1.1775306028542796



#APPLY LOG TRANSFORMATION

```
sns.distplot(np.log(train.Item_Outlet_Sales))
print(np.log(train.Item_Outlet_Sales).skew())
```

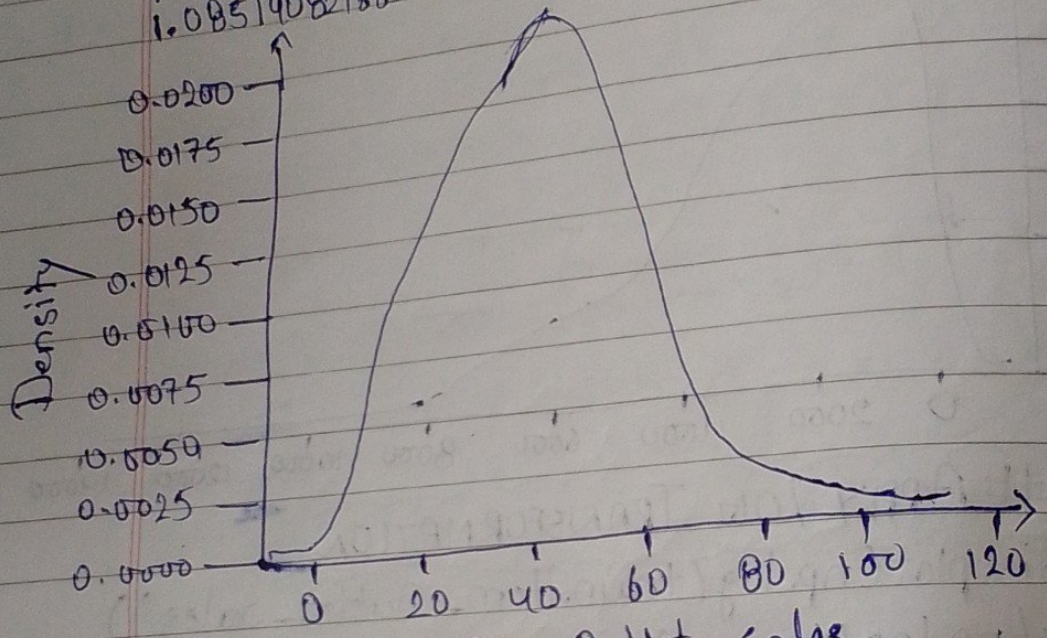
-0.8877533432094305



APPLY SQRT TRANSFORMATION

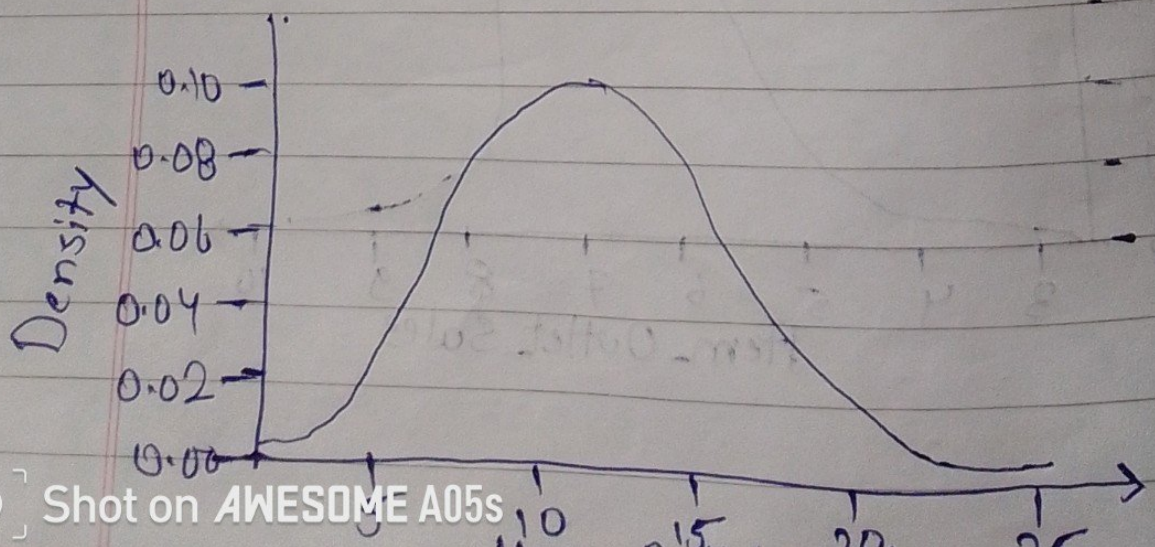
```
sns.distplot(np.sqrt(train.Item_Outlet_Sales))
print(np.sqrt(train.Item_Outlet_Sales).skew())
```

1.0851408218698326



APPLY CUBEROOT TRANSFORMATION

```
sns.distplot(np.cbrt(train.Item_Outlet_Sales))
print(np.cbrt(train.Item_Outlet_Sales).skew())
```



a mathematical function that describes the output of a

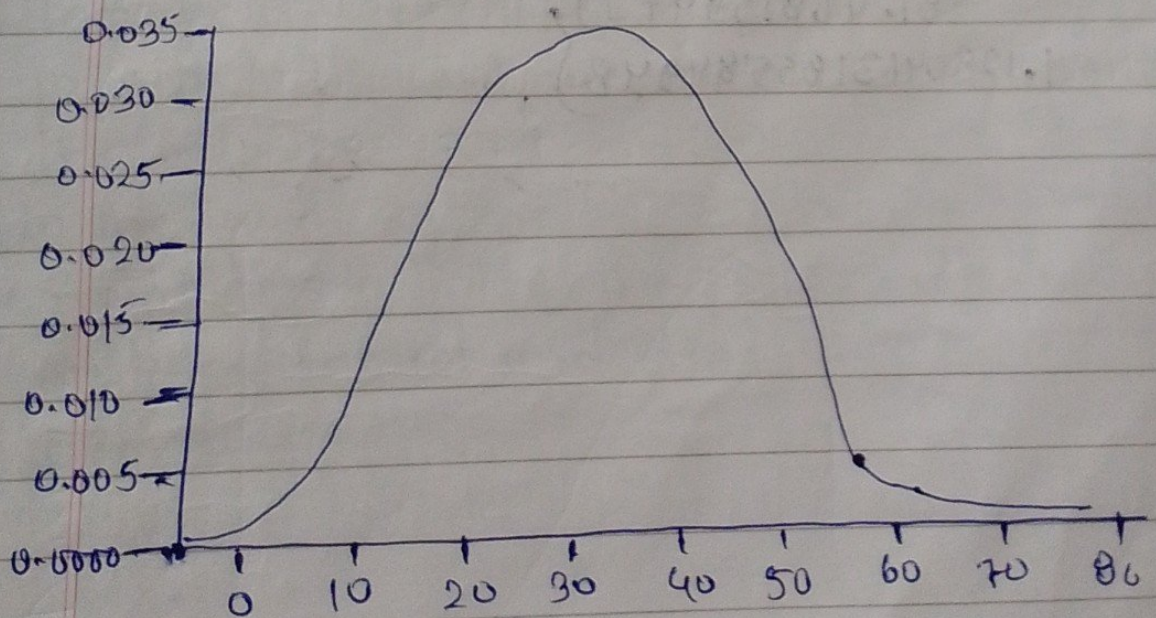
BOX COX TRANSFORMATION & YEO-JOHNSON TRANSFORMATION

① Box-Cox Transformation works on transforming the Positive Values (greater than 0) only as the function entails usage of Log Transformation as per the formula if the function encounters 0 in the data.

② Where as the Yeo Johnson Function can work on any number be it 0, -ve or +ve number. In totality, the Yeo Johnson is more flexible and a versatile transformation bet on the data.

```
# use scipy.stats . Box-Cox
sns.distplot(stats.boxcox(train.Item_Outlet_Sales)[0])
print(stats.skew(stats.boxcox(train.Item_Outlet_Sales)[0]))
```

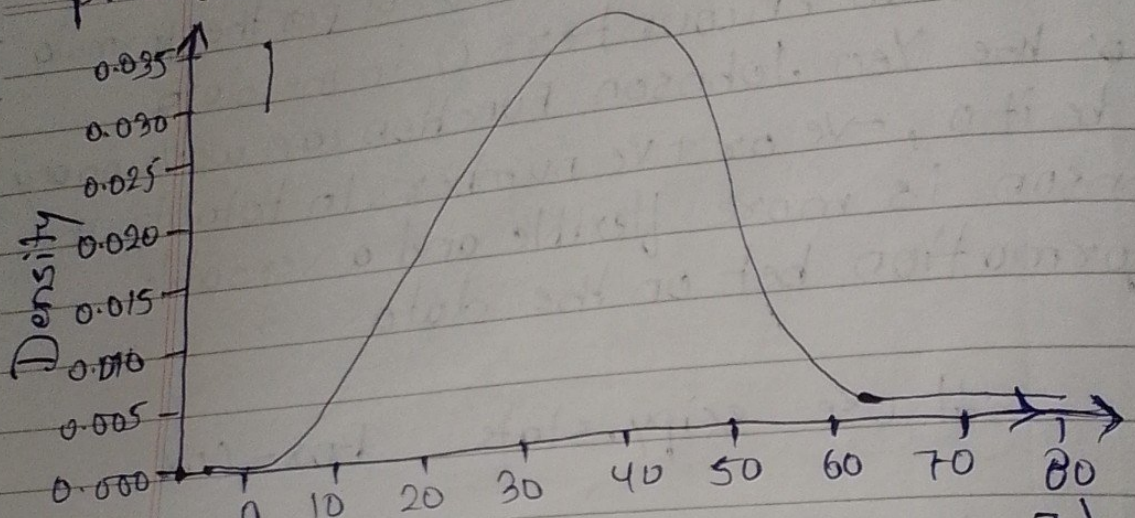
-0.07496422376727 6



a mathematical [illegible]
now)

Use Yeo-Johnson

```
sns.distplot(stats.yeojohnson(train.Item_Outlet_Sales)[0])
print(stats.skew(stats.yeojohnson(train.Item_Outlet_
                                  Sales)[0]))
```




```
stats.yeojohnson([23,54,98,0,-134])
```

output

```
(array([ 3.07413761, 80.56195565, 157.18079893,
        -81.46815947]),
 1.1280463185584948)
```

MISSING VALUES

- Rule says if 95 % or more data is missing in the column, then it is best to drop that column

Notes: This is a theoretical rule. The point is to find the pattern in the data in such a way that you can handle most of the missing values in the data.

1. Now, basis discussion, we see that prima facie we can consider the mode of the categorical variable & Median in Numerical variable to impute the
2. However, this approach might not work because we have not explore the pattern from the rest of the columns/ feature
3. Therefore, we need to do the pattern exploration on priority to find out the closest possible pattern for the missing value and then impute it accordingly
4. In the event of the pattern not being available or unable to mine the pattern, it's always good to build the model and deal with the missing values accordingly.

|| FEATURE ENGINEERING ||

① Bin the Item Identifier and Item Type

Note: The purpose of creating new features and binning is to make sure that the machine is able to capture the pattern well. This concept is known as Generalization.

combined.head(2)

	Item_Identifier	Item_Weight	Item_Fat_Content	Item_Visibility
0	FDA15	9.30	Low_Fat	0.016047
1	DRC01	5.92	Regular	0.019278

Item_Type	Item_MRP	OutletIdentifier	Outlet_Establishment_Year
Dairy	249.8092	OUT049	1999
Soft Drinks	48.2692	OUT018	2009

Outlet_Size	Outlet_Location_Type	OutletType	Item_Outlet_Sales
Medium	Tier1	Supermarket Type1	3735.1380
Medium	Tier3	Supermarket Type2	443.4228

combined.Item_Type.nunique() = 16 categories

#Extract the first two letters from Item ID

```
ids=[]

for i in combined.Item_Identifier:
    ids.append(i[:2])

combined["Item_IDS"] = pd.Series(ids)
```

Steps that I follow

1. Univariate
2. Bivariate
3. Feature Engineering
4. Missing values
5. Outlier Analysis and Removal = This is the step where I split the data in train and test back again
6. Scaling & Transformation
7. Categorical Encoding

Why I split the Data in train and test: Before Outliers ??

- You remove the Outliers from training data only and not from test
- Train: that dataset on which the model will be trained
- Test: is that dataset on which the model will predict.

Outlier Analysis

Here at this stage, we will split the data in Train & Test

* The No of columns / Features in the Test set always n-1 where n is the total no. of columns in the Train set.

* Train set is the data from where the model will learn the pattern because it has predictors and the target variable

* Test is the data where we will do the prediction and check the performance of the model.

#Splitting the Data back in train and test

```
train.shape, test.shape
((8523,12), (5681,12))
```

original data split in train & test

```
combined.shape
(14204,13)
```

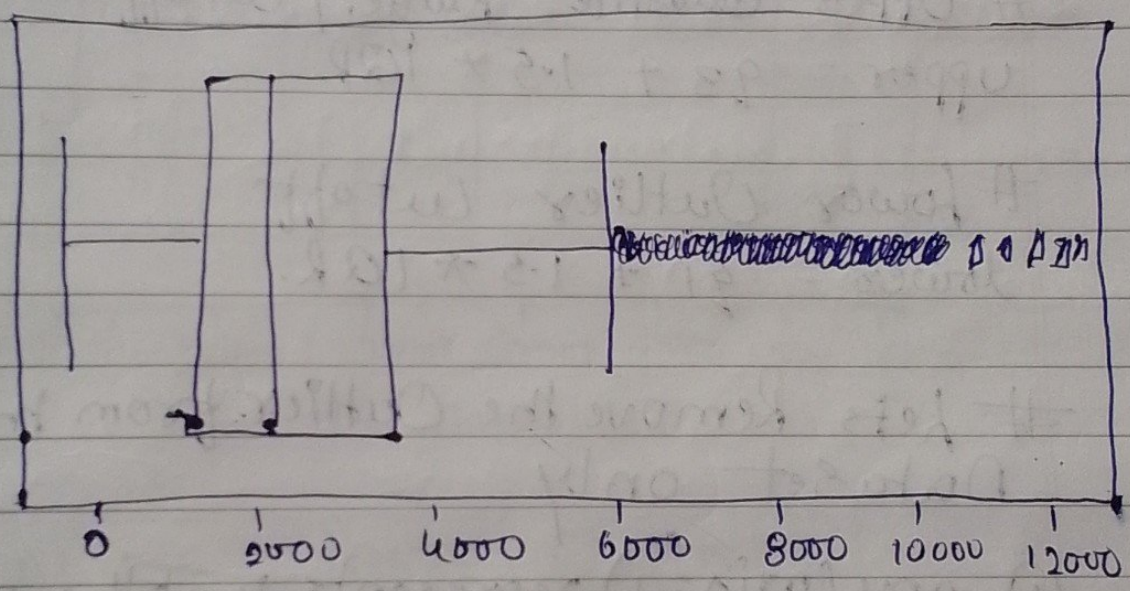
```
newtrain = combined.loc[0:train.shape[0]-1]
newtest = combined.loc[train.shape[0]:,]
newtrain.shape , newtest.shape
(8523,13)          (5681,13)
```

Rule of Identifying the Outliers:

1. Any value greater than $(Q_3 + 1.5 * IQR)$ is known as Outlier. This is called Upper Outlier
2. Similarly, Any value less than $(Q_1 - 1.5 * IQR)$ is known as Lower Outlier

See the Outlier

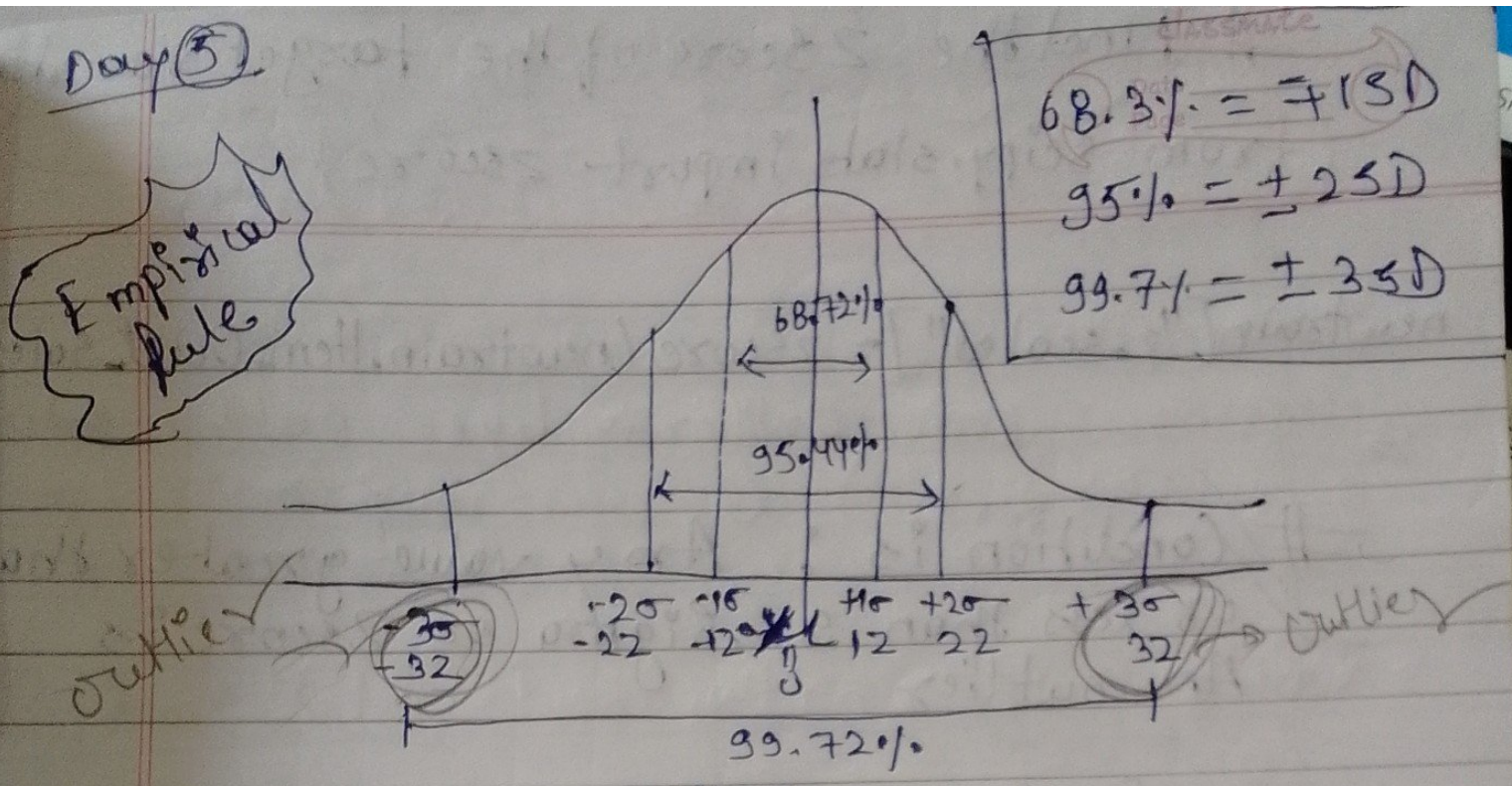
```
sns.boxplot(data = newtrain, x = "Item_Outlet_Sales")
```




```
Q3 = newtrain.Item_Outlet_Sales.quantile(0.75)
Q1 = newtrain.Item_Outlet_Sales.quantile(0.25)
IQR = Q3 - Q1
Q3 + 1.5 * IQR
6501.8699 — whisker value
```

Any value greater than this are outliers.

Day (5)



Normal → (Zscore) → St. Normal

$$Zscore = \frac{xi - \mu}{\sigma}$$

Outliers using Zscore

(*) Generally the Z score is applied on the Target Variable

Empirical Rule

(*) 68.3% Data lies within +/- 1 S.D.
(*) 95.44% Data lies within +/- 2 S.D.
(*) 99.7% Data lies within +/- 3 S.D.

ENCODING

Category → Numeric — binary (0, 1)

* ONE HOT ENCODING / N-1 ENCODING
* LABEL ENCODING / ORDINAL ENCODING
* FREQUENCY ENCODING
* TARGET ENCODING

Big Mart Sales

Item ID	Outletsize
FD	Small
NC	Medium
DR	High
FD	
DR	

DR → NO ORDINAL

High → ORDINAL

House Price / No Broker

#B	Area	BSMT	DCC	HB(x)
1	2072	Y	30km	
2	1100	N	1km	
3	5000	Y	0.5M	
5	3000	N	2kms	
Rk	1080	N	}	
		N		

N → one Hot Encoding ≈ pd.get_dummies

Sex	M	F
M	1	0
F	0	1
M	1	0
F	0	1
M	1	0
F	0	1

density function (PDF)

SEX	M
M	1
F	0
M	1
F	0
M	1
F	0

no difference from previous table

No

Met

Si

Ry

One Hot Encoding Handles Multicornality

ORDINAL LABEL ENCODING

```
dict_x = {'Small': 1, 'Medium': 2, 'Large': 3}
data['outlet_size'].map(dict_x)
```

1
2
3

Sklearn – labeler
→ Alphabetical sequence

mean
train

output

Nomical	FREQUENCY ENCODING
Metformin	1500
Sitaguptin	5000
Rybelsus	3500

इसमें हर item को उसकी frequency दी जाती है and इसका disadvantage यह है कि अगर किसी item की frequency same होगी तो fail हो जाएगा।

TARGET ENCODING

```
mean_sales = train.groupby("Outlet_Identifier")["Item_Outlet_Sales"].
                mean().to_dict()
```

OUTPUT

```
{'OUT010': 839.351
 'OUT013': 2298.995
 'OUT017': 2340.675
 'OUT018': 1995.498
 'OUT019': 340.3297
}
```

3

Target Encoding में हमने outlet_Identifier की encoding को उसके basis of Item_Outlet_Sales and target encoding में encoding की Target variable के according.

ONE- HOT ENCODING

```
dummytrain = pd.get_dummies(wt_outliers, drop_first = True)

dummytest = pd.get_dummies(newtest, drop_first = True)

dummytrain.shape , dummytest.shape
```

NAME: Rishi Tanwar STD.: SEC.: ROLL NO.: SUB.:

S. No.	Date	Title	Page No.	Teacher's Sign / Remarks

Standard Deviation is a measure which shows how much variation (such as spread, dispersion, spread) from the mean exists. The standard deviation indicates a "typical" deviation from the mean.

Standard Deviation the most widely used measure of dispersion, is based on all values. Therefore a change in given one value affects the value of standard deviation. It is independent of origin but not of scale. It is also useful in certain advanced statistical problems.

Skewness is a measurement of the distortio of symmetrical or asymmetrical distribut in a dataset skewness is demonstrated on a bell curve.

Standard Normal Deviation = A standard Normal deviate is a normally distributed deviate. It is a realization of a standard normal random variable, defined as a random variable with expected value 0 and variance 1.

S. No. | Date | Title | Page No. | Teacher's Sign / Remarks

Standard Deviation is a measure which shows how much variation (such as spread, dispersion, spread) from the mean exists. The standard deviation indicates a "typical" deviation from the mean.

Standard Deviation the most widely used measure of dispersion, is based on all values. Therefore a change in given one value affects the value of standard deviation. It is independent of origin but not of scale. It is also useful in certain advanced statistical problems.

Skewness is a measurement of the distortion of symmetrical or asymmetrical distribution in a dataset skewness is demonstrated on a bell curve.

Standard Normal Deviation = A standard Normal deviate is a normally distributed deviate. It is a realization of a standard normal random variable, defined as a random variable with expected value 0 and variance 1.

It is a mathematical func^n that describ... continuous Random...

pmf

EDA is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.

Mean = the average of the given set of values. It denotes the equal distribution of values for a given data set.
The mean (average) of a dataset is found by adding all numbers in the data set and then dividing by number of values in the set

Median = Median is the middle value when a data set is ordered from least to greatest

Mode = Mode is the number that occurs most often in a data set

Variance = is the measure of how notably a collection of data is spread out. If all the data values are identical, then it indicates the variance is zero. All non zero variance is zero. All non zero variances are considered to be positive. A little variances represents that the data points are close to the mean and to each other, whereas if the data points are highly spread out from the mean and from one another indicates the high variance. In short, the variance is defined as the average of the squared distance from each point to the mean.

EDA is used by data scientists to investigate data sets and summarize their main characteristics, often employing data visualization methods.

Mean = the average of the given set of values. It denotes the equal distribution of values for a given data set.
The mean (average) of a dataset is found by adding all numbers in the data set and then dividing by number of values in the set

Median = Median is the middle value when a data set is ordered from least to greatest

Mode = Mode is the number that occurs most often in a data set

Variance = is the measure of how notably a collection of data is spread out. If all the data values are identical, then it indicates the variance is zero. All non zero variances is zero. All non zero variances are considered to be positive. A little variances represents that the data points are close to the mean or to each other, whereas if the data points are highly spread out from the mean and from one another indicates the high variance. In short, the variance is defined as the average of the squared distance from each point to the mean.