DIMENSION REDUCTION

Project Report

DIMENSION REDUCTION (PCA):

Introduction:

Dimension reduction is a method in machine learning aimed at reducing the number of features or variables in a dataset while preserving its essential information. This technique is employed to simplify the dataset, making it more manageable and efficient for subsequent analyses. In this context, dimension reduction can enhance the computational efficiency and interpretability of the data, ultimately facilitating a more streamlined and effective modeling process. The dataset under consideration for this dimension reduction task is derived from the PCA+India+Data_Census.xlsx file.

Data Description:

Name	Description
State	State Code
District	District Code
Name	Name
TRU1	Area Name
No_HH	No of Household
TOT_M	Total population Male
TOT_F	Total population Female
M_06	Population in the age group 0-6 Male
F_06	Population in the age group 0-6 Female
M_SC	Scheduled Castes population Male
F_SC	Scheduled Castes population Female
M_ST	Scheduled Tribes population Male
F_ST	Scheduled Tribes population Female
M_LIT	Literates population Male
F_LIT	Literates population Female
M_ILL	Illiterate Male
F_ILL	Illiterate Female
TOT_WORK_M	Total Worker Population Male
TOT_WORK_F	Total Worker Population Female
MAINWORK_M	Main Working Population Male
MAINWORK_F	Main Working Population Female
MAIN_CL_M	Main Cultivator Population Male

MAIN_CL_F	Main Cultivator Population Female
MAIN_AL_M	Main Agricultural Labourers Population Male
MAIN_AL_F	Main Agricultural Labourers Population Female
MAIN_HH_M	Main Household Industries Population Male
MAIN_HH_F	Main Household Industries Population Female
MAIN_OT_M	Main Other Workers Population Male
MAIN_OT_F	Main Other Workers Population Female
MARGWORK_M	Marginal Worker Population Male
MARGWORK_F	Marginal Worker Population Female
MARG_CL_M	Marginal Cultivator Population Male
MARG_CL_F	Marginal Cultivator Population Female
MARG_AL_M	Marginal Agriculture Labourers Population Male
MARG_AL_F	Marginal Agriculture Labourers Population Female
MARG_HH_M	Marginal Household Industries Population Male
MARG_HH_F	Marginal Household Industries Population Female
MARG_OT_M	Marginal Other Workers Population Male
MARG_OT_F	Marginal Other Workers Population Female
MARGWORK_3_6_M	Marginal Worker Population 3-6 Male
MARGWORK_3_6_F	Marginal Worker Population 3-6 Female
MARG_CL_3_6_M	Marginal Cultivator Population 3-6 Male
MARG_CL_3_6_F	Marginal Cultivator Population 3-6 Female
MARG_AL_3_6_M	Marginal Agriculture Labourers Population 3-6 Male
MARG_AL_3_6_F	Marginal Agriculture Labourers Population 3-6 Female
MARG_HH_3_6_M	Marginal Household Industries Population 3-6 Male
MARG_HH_3_6_F	Marginal Household Industries Population 3-6 Female
MARG_OT_3_6_M	Marginal Other Workers Population Person 3-6 Male
MARG_OT_3_6_F	Marginal Other Workers Population Person 3-6 Female
MARGWORK_0_3_M	Marginal Worker Population 0-3 Male
MARGWORK_0_3_F	Marginal Worker Population 0-3 Female
MARG_CL_0_3_M	Marginal Cultivator Population 0-3 Male
MARG_CL_0_3_F	Marginal Cultivator Population 0-3 Female
MARG_AL_0_3_M	Marginal Agriculture Labourers Population 0-3 Male
MARG_AL_0_3_F	Marginal Agriculture Labourers Population 0-3 Female
MARG_HH_0_3_M	Marginal Household Industries Population 0-3 Male
MARG_HH_0_3_F	Marginal Household Industries Population 0-3 Female
MARG_OT_0_3_M	Marginal Other Workers Population 0-3 Male
MARG_OT_0_3_F	Marginal Other Workers Population 0-3 Female
NON_WORK_M	Non Working Population Male

NON_WORK_F Non Working Population Female

STEPS:

Step 1: Data Loading and Overview:

Loaded the dataset using pandas and displayed the first and last few rows to understand the data structure.

PCA India Data_Census.xlsx dataset is loaded into the dataframe.

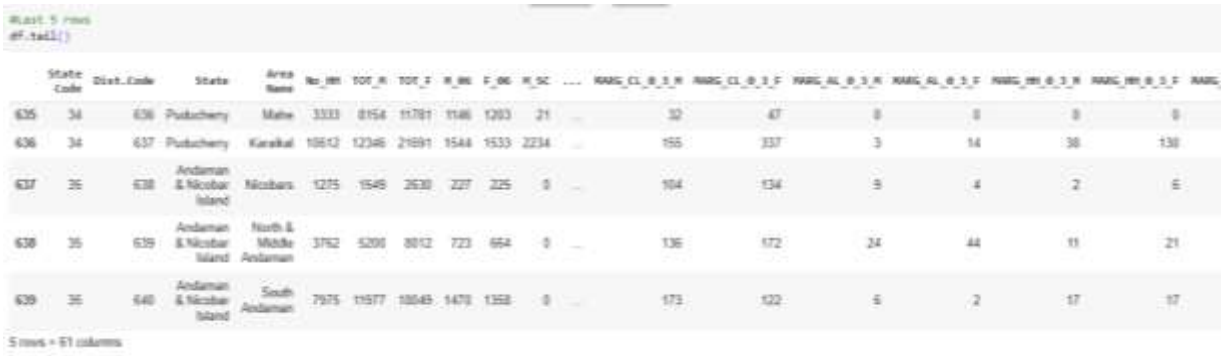
Data Frame printing rows with Head (Prints top 5 rows) function as below :

```
#top rows
df.head()
```

	State Code	Dist.Code	State	Area Name	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	...	MARG_CL_0_3_M	MARG_CL_0_3_F	MARG_AL_0_3_M	MARG_AL_0_3_F	MARG_HH_0_3_M	MARG_HH_0_3_F	MARG_O
0	1	1	Jammu & Kashmir	Kupwara	7707	23388	25796	5862	6196	3	...	1150	749	180	237	680	252	
1	1	2	Jammu & Kashmir	Badgam	6218	19585	23102	4462	3733	7	...	525	715	123	229	186	148	
2	1	3	Jammu & Kashmir	Leh(Ladakh)	4452	6546	10964	1082	1018	3	...	114	188	44	89	3	34	
3	1	4	Jammu & Kashmir	Kargil	1320	2764	4206	563	677	0	...	194	247	61	128	13	50	
4	1	5	Jammu & Kashmir	Punch	11654	20591	29981	5157	4587	20	...	874	1928	466	1043	205	302	

5 rows × 61 columns

Data Frame printing rows with Tail (Prints last 5 rows) function as below :



	State Code	Dist.Code	State	Area Name	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	...	MARG_CL_0_3_M	MARG_CL_0_3_F	MARG_AL_0_3_M	MARG_AL_0_3_F	MARG_HH_0_3_M	MARG_HH_0_3_F	MARG
635	34	636	Puducherry	Mahe	3333	8154	11781	1146	1203	21	...	32	47	0	0	0	0	
636	34	637	Puducherry	Karaikal	10612	12346	21691	1544	1533	2234	...	155	337	3	14	30	130	
637	35	638	Andaman & Nicobar Island	Nicobars	1275	1549	2630	227	225	0	...	104	134	9	4	2	6	
638	35	639	Andaman & Nicobar Island	North & Middle Andaman	3762	5200	8012	723	654	0	...	136	172	24	44	11	21	
639	35	640	Andaman & Nicobar Island	South Andaman	7975	11577	18045	1470	1358	0	...	173	122	6	2	17	17	

5 rows × 61 columns

Checking the shape of Dataset

```
640, 61)
```

There are 640 Rows and 61 Columns.

Below is the Info of Dataset

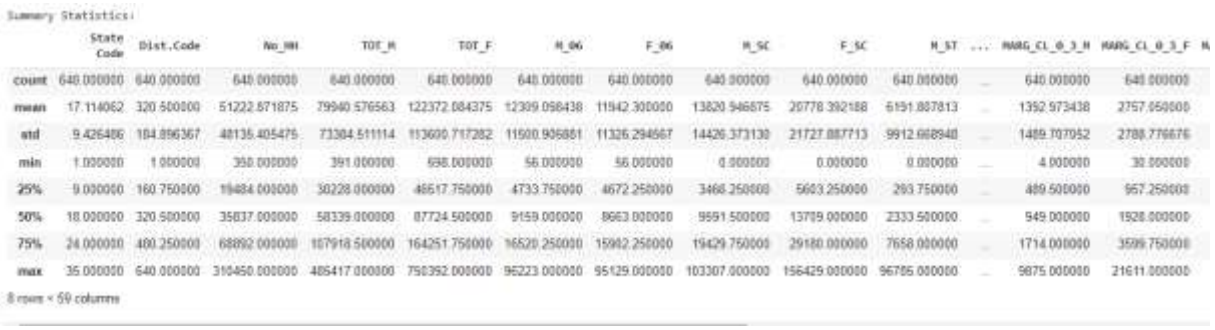
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 640 entries, 0 to 639
Data columns (total 61 columns):
```

#	Column	Non-Null Count	Dtype
0	State Code	640 non-null	int64
1	Dist.Code	640 non-null	int64
2	State	640 non-null	object
3	Area Name	640 non-null	object
4	No_HH	640 non-null	int64
5	TOT_M	640 non-null	int64
6	TOT_F	640 non-null	int64
7	M_06	640 non-null	int64
8	F_06	640 non-null	int64
9	M_SC	640 non-null	int64
10	F_SC	640 non-null	int64
11	M_ST	640 non-null	int64
12	F_ST	640 non-null	int64
13	M_LIT	640 non-null	int64
14	F_LIT	640 non-null	int64
15	M_ILL	640 non-null	int64
16	F_ILL	640 non-null	int64
17	TOT_WORK_M	640 non-null	int64
18	TOT_WORK_F	640 non-null	int64
19	MAINWORK_M	640 non-null	int64
20	MAINWORK_F	640 non-null	int64
21	MAIN_CL_M	640 non-null	int64
22	MAIN_CL_F	640 non-null	int64
23	MAIN_AL_M	640 non-null	int64
24	MAIN_AL_F	640 non-null	int64
25	MAIN_HH_M	640 non-null	int64
26	MAIN_HH_F	640 non-null	int64
27	MAIN_OT_M	640 non-null	int64
28	MAIN_OT_F	640 non-null	int64
29	MARGWORK_M	640 non-null	int64

```
 30  MARGWORK_F       640 non-null    int64
 31  MARG_CL_M        640 non-null    int64
 32  MARG_CL_F        640 non-null    int64
 33  MARG_AL_M        640 non-null    int64
 34  MARG_AL_F        640 non-null    int64
 35  MARG_HH_M        640 non-null    int64
 36  MARG_HH_F        640 non-null    int64
 37  MARG_OT_M        640 non-null    int64
 38  MARG_OT_F        640 non-null    int64
 39  MARGWORK_3_6_M   640 non-null    int64
 40  MARGWORK_3_6_F   640 non-null    int64
 41  MARG_CL_3_6_M    640 non-null    int64
 42  MARG_CL_3_6_F    640 non-null    int64
 43  MARG_AL_3_6_M    640 non-null    int64
 44  MARG_AL_3_6_F    640 non-null    int64
 45  MARG_HH_3_6_M    640 non-null    int64
 46  MARG_HH_3_6_F    640 non-null    int64
 47  MARG_OT_3_6_M    640 non-null    int64
 48  MARG_OT_3_6_F    640 non-null    int64
 49  MARGWORK_0_3_M   640 non-null    int64
 50  MARGWORK_0_3_F   640 non-null    int64
 51  MARG_CL_0_3_M    640 non-null    int64
 52  MARG_CL_0_3_F    640 non-null    int64
 53  MARG_AL_0_3_M    640 non-null    int64
 54  MARG_AL_0_3_F    640 non-null    int64
 55  MARG_HH_0_3_M    640 non-null    int64
 56  MARG_HH_0_3_F    640 non-null    int64
 57  MARG_OT_0_3_M    640 non-null    int64
 58  MARG_OT_0_3_F    640 non-null    int64
 59  NON_WORK_M       640 non-null    int64
 60  NON_WORK_F       640 non-null    int64
dtypes: int64(59), object(2)
memory usage: 305.1+ KB
```

There are ‘No duplicate values’ and ‘No Null values’ in to the dataset.

Using Describe Function :



Summary Statistics:

	State Code	Dist.Code	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	F_SC	M_ST	...	MARG_CL_0_3_M	MARG_CL_0_3_F	M
count	640.000000	640.000000	640.000000	640.000000	640.000000	640.000000	640.000000	640.000000	640.000000	640.000000	...	640.000000	640.000000	
mean	17.114062	320.500000	51222.871875	79940.576563	122372.084375	12309.098438	11942.300000	13820.946875	20778.392188	6151.887813	...	1352.973438	2757.050000	
std	9.426486	184.896367	48135.405475	73384.511114	113600.717282	11500.906881	11326.294667	14426.373130	21727.887713	9912.668948	...	1489.707052	2788.776676	
min	1.000000	1.000000	350.000000	391.000000	698.000000	56.000000	56.000000	0.000000	0.000000	0.000000	...	4.000000	30.000000	
25%	9.000000	160.750000	19484.000000	30228.000000	46617.750000	4733.750000	4672.250000	3468.250000	5603.250000	293.750000	...	489.500000	957.250000	
50%	18.000000	320.500000	35837.000000	58339.000000	87724.500000	9159.000000	8663.000000	9591.500000	13709.000000	2333.500000	...	949.000000	1928.000000	
75%	24.000000	480.250000	68892.000000	107918.500000	164251.750000	16520.250000	15982.250000	19429.750000	29180.000000	7658.000000	...	1714.000000	3599.750000	
max	35.000000	640.000000	310460.000000	485417.000000	750392.000000	96223.000000	95129.000000	103307.000000	156429.000000	96786.000000	...	9875.000000	21611.000000	

8 rows × 59 columns

Step 2: EDA

- **Feature Selection**

We Selected ('No_HH', 'TOT_M', 'TOT_F', 'M_06', 'F_06') Variables for further analysis..

I have picked 5 Variables such as: ['No_HH', 'TOT_M', 'TOT_F', 'M_06', 'F_06'].

TOT_M	Total population Male
TOT_F	Total population Female
No_HH	No. of House Holds
M_06	Males below 6 years
F_06	Females below 6 years

- **Summary statistics for selected variables:**

	No_HH	TOT_M	TOT_F	M_06	F_06
count	640.000000	640.000000	640.000000	640.000000	640.000000
mean	51222.871875	79940.576563	122372.084375	12309.098438	11942.300000
std	48135.405475	73384.511114	113600.717282	11500.906881	11326.294567
min	350.000000	391.000000	698.000000	56.000000	56.000000
25%	19484.000000	30228.000000	46517.750000	4733.750000	4672.250000
50%	35837.000000	58339.000000	87724.500000	9159.000000	8663.000000
75%	68892.000000	107918.500000	164251.750000	16520.250000	15902.250000
max	310450.000000	485417.000000	750392.000000	96223.000000	95129.000000

Step 3: Data Scaling and Covariance Matrix:

We Selected Numeric columns and applied z-score normalization using the StandardScaler.
Then we calculated covariance matrix and obtained eigenvalues and eigenvectors .

Numerical Columns

```
numeric_columns
```

	State Code	Dist.Code	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	F_SC	M_ST	...	MARG_CL_0_3_M	MARG_CL_0_3_F	MARG_AL_0_3_M	MARG_AL_0_3_F	MARG_HH_0_3_M	MARG_HH_0
0	1	1	7707	23388	29796	5862	6196	3	0	1999	...	1150	749	180	237	680	
1	1	2	6210	19565	23102	4482	3733	7	6	427	...	525	715	123	229	186	
2	1	3	4452	6546	10964	1082	1018	3	6	5806	...	114	188	44	88	3	
3	1	4	1320	2784	4206	563	677	0	0	2666	...	194	247	61	128	13	
4	1	5	11554	20591	28981	5157	4587	20	33	7570	...	874	1928	465	1043	205	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
635	34	636	3333	8154	11781	1146	1203	21	30	0	...	32	47	0	0	0	
636	34	637	10612	12345	21691	1544	1533	2234	4155	0	...	155	337	3	14	38	
637	35	638	1275	1549	2630	227	225	0	0	1012	...	104	134	9	4	2	
638	35	639	3762	5200	8012	723	664	0	0	26	...	136	172	24	44	11	
639	35	640	7975	11977	18049	1470	1358	0	0	161	...	173	122	6	2	17	

640 rows × 39 columns

Find below Covariance Matrix for scaled dataset

```
array([[1.00156495, 0.91760364, 0.97210871, ..., 0.53769433, 0.76357722,
        0.73684378],
       [0.91760364, 1.00156495, 0.98417823, ..., 0.5891007 , 0.84621844,
        0.71718181],
       [0.97210871, 0.98417823, 1.00156495, ..., 0.572748  , 0.82894851,
        0.74775097],
       ...,
       [0.53769433, 0.5891007 , 0.572748  , ..., 1.00156495, 0.61052325,
        0.52191235],
       [0.76357722, 0.84621844, 0.82894851, ..., 0.61052325, 1.00156495,
        0.88228018],
       [0.73684378, 0.71718181, 0.74775097, ..., 0.52191235, 0.88228018,
        1.00156495]])
```

Eigen Vectors

```
eigenvectors
```

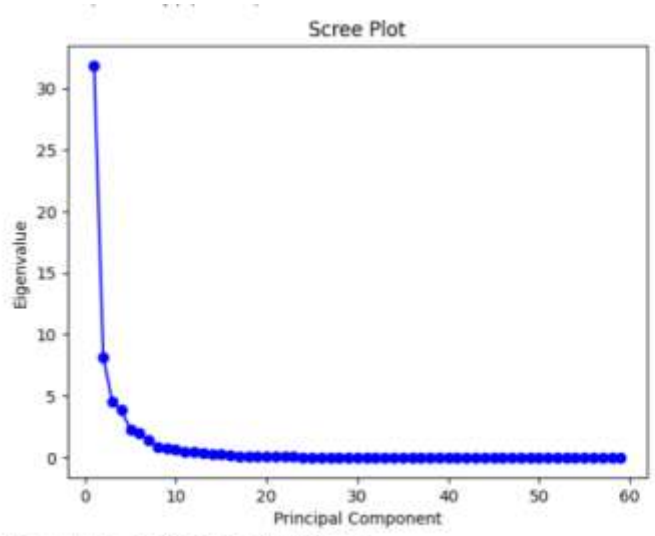
```
array([[-1.56020579e-01+0.j,  1.26346525e-01+0.j, -2.69025037e-03+0.j,
         ...,  8.59813362e-14+0.j, -1.76746604e-13+0.j,
         1.08099785e-13+0.j],
       [-1.67117635e-01+0.j,  8.96765481e-02+0.j,  5.66976191e-02+0.j,
         ...,  5.28021877e-02+0.j, -1.10854618e-01+0.j,
         2.72794503e-02+0.j],
       [-1.65553179e-01+0.j,  1.04912371e-01+0.j,  3.87494746e-02+0.j,
         ...,  2.04945480e-01+0.j, -2.01549789e-01+0.j,
         1.06371673e-01+0.j],
       ...,
       [-1.32192245e-01+0.j, -5.08133220e-02+0.j, -7.87198691e-02+0.j,
         ...,  5.02777797e-03+0.j,  2.99863496e-03+0.j,
         5.44284135e-05+0.j],
       [-1.50375578e-01+0.j,  6.53645529e-02+0.j,  1.11827318e-01+0.j,
         ..., -3.00935627e-02+0.j, -1.48422322e-02+0.j,
         9.62462558e-02+0.j],
       [-1.31066203e-01+0.j,  7.38474208e-02+0.j,  1.02552501e-01+0.j,
         ...,  2.84776742e-02+0.j, -5.59381919e-02+0.j,
         1.00243536e-01+0.j]])
```

Eigen values

```
eigenvalues
```

```
array([ 3.18135647e+01+0.00000000e+00j,  7.86942415e+00+0.00000000e+00j,
        4.15340812e+00+0.00000000e+00j,  3.66879058e+00+0.00000000e+00j,
        2.20652588e+00+0.00000000e+00j,  1.93827502e+00+0.00000000e+00j,
        1.17617374e+00+0.00000000e+00j,  7.51159086e-01+0.00000000e+00j,
        6.17053743e-01+0.00000000e+00j,  5.28300887e-01+0.00000000e+00j,
        4.29831189e-01+0.00000000e+00j,  3.53440201e-01+0.00000000e+00j,
        2.96163013e-01+0.00000000e+00j,  2.81275560e-01+0.00000000e+00j,
        1.92158325e-01+0.00000000e+00j,  1.36267920e-01+0.00000000e+00j,
        1.13389199e-01+0.00000000e+00j,  1.06303946e-01+0.00000000e+00j,
        9.72885376e-02+0.00000000e+00j,  8.01062194e-02+0.00000000e+00j,
        5.76089954e-02+0.00000000e+00j,  4.43955966e-02+0.00000000e+00j,
        3.78910846e-02+0.00000000e+00j,  2.96360194e-02+0.00000000e+00j,
        2.70797618e-02+0.00000000e+00j,  2.34458139e-02+0.00000000e+00j,
        1.45111511e-02+0.00000000e+00j,  7.13559124e-04+0.00000000e+00j,
        1.06789820e-03+0.00000000e+00j,  2.59771182e-03+0.00000000e+00j,
        5.02601514e-03+0.00000000e+00j,  1.09852268e-02+0.00000000e+00j,
        9.31507853e-03+0.00000000e+00j,  8.13540203e-03+0.00000000e+00j,
        7.89250253e-03+0.00000000e+00j, -1.62639278e-15+0.00000000e+00j,
        1.73838880e-15+0.00000000e+00j, -1.23163836e-15+0.00000000e+00j,
       -1.09693403e-15+0.00000000e+00j,  1.22980914e-15+2.39724559e-16j,
        1.22980914e-15-2.39724559e-16j,  1.17710893e-15+0.00000000e+00j,
       -8.30353209e-16+0.00000000e+00j,  1.00394338e-15+0.00000000e+00j,
        9.54474887e-16+0.00000000e+00j, -6.07659961e-16+7.92737922e-17j,
       -6.07659961e-16-7.92737922e-17j,  7.62061776e-16+0.00000000e+00j,
       -3.84222466e-16+0.00000000e+00j, -3.62660144e-16+0.00000000e+00j,
       -1.65061771e-16+0.00000000e+00j,  1.29780694e-17+2.35304366e-17j,
        1.29780694e-17-2.35304366e-17j,  1.54490058e-16+0.00000000e+00j,
        3.05572930e-16+0.00000000e+00j,  4.57999896e-16+0.00000000e+00j,
        4.31159259e-16+0.00000000e+00j])
```

Step 4: Scree Plot and Optimal Number of Principal Components:




```
Optimum Number of Principal Components: 7
```

Data before z-score scaling is as below

	count	mean	std	min	25%	50%	75%	max
Ad - Length	23066.0	3.851631e+02	2.336514e+02	120.0000	120.000000	300.000000	7.200000e+02	728.00
Ad- Width	23066.0	3.378960e+02	2.030929e+02	70.0000	250.000000	300.000000	6.000000e+02	600.00
Ad Size	23066.0	9.667447e+04	6.153833e+04	33600.0000	72000.000000	72000.000000	8.400000e+04	216000.00
Available_Impressions	23066.0	2.432044e+06	4.742888e+06	1.0000	33672.250000	483771.000000	2.527712e+06	27592861.00
Matched_Queries	23066.0	1.295099e+06	2.512970e+06	1.0000	18282.500000	258087.500000	1.180700e+06	14702025.00
Impressions	23066.0	1.241520e+06	2.429400e+06	1.0000	7990.500000	225290.000000	1.112428e+06	14194774.00
Clicks	23066.0	1.067852e+04	1.735341e+04	1.0000	710.000000	4425.000000	1.279375e+04	143049.00
Spend	23066.0	2.706626e+03	4.067927e+03	0.0000	85.180000	1425.125000	3.121400e+03	26931.87
Fee	23066.0	3.351231e-01	3.196322e-02	0.2100	0.330000	0.350000	3.500000e-01	0.35
Revenue	23066.0	1.924252e+03	3.105238e+03	0.0000	55.365375	926.335000	2.091338e+03	21276.18
CTR	23066.0	7.366054e-02	6.700065e-02	0.0001	0.003400	0.073661	1.219000e-01	1.00
CPM	23066.0	7.672045e+00	5.777778e+00	0.0000	1.850000	7.672045	1.134000e+01	81.56
CPC	23066.0	3.510606e-01	3.060619e-01	0.0000	0.100000	0.351061	4.700000e-01	7.26

Here, I have applied z-score method and I got the below output.

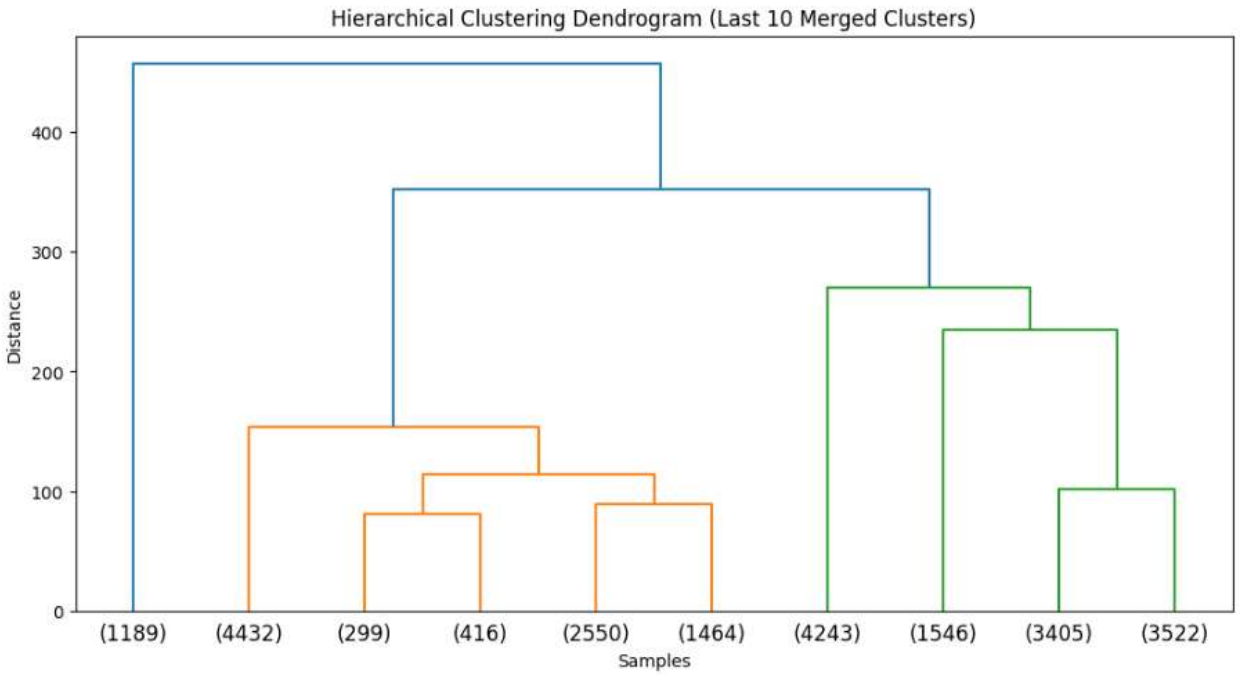
	count	mean	std	min	25%	50%	75%	max
Ad - Length	23066.0	1.281478e-16	1.000022	-1.134891	-1.134891	-3.644957e-01	1.433093	1.467332
Ad- Width	23066.0	-1.182903e-16	1.000022	-1.319110	-0.432797	-1.865987e-01	1.290590	1.290590
Ad Size	23066.0	2.464381e-17	1.000022	-1.024985	-0.400970	-4.009697e-01	-0.205965	1.939086
Available_Impressions	23066.0	-1.971505e-17	1.000022	-0.512788	-0.505688	-4.107866e-01	0.020171	5.305072
Matched_Queries	23066.0	-5.914515e-17	1.000022	-0.515377	-0.508102	-4.126727e-01	-0.045524	5.335208
Impressions	23066.0	-1.971505e-17	1.000022	-0.511050	-0.507761	-4.183138e-01	-0.053138	5.331990
Clicks	23066.0	-3.943010e-17	1.000022	-0.615311	-0.574454	-3.603704e-01	0.121894	7.628089
Spend	23066.0	-3.943010e-17	1.000022	-0.665372	-0.644432	-3.150323e-01	0.101964	5.955310
Fee	23066.0	6.703117e-16	1.000022	-3.914682	-0.160285	4.654474e-01	0.465447	0.465447
Revenue	23066.0	7.886020e-17	1.000022	-0.619693	-0.601863	-3.213727e-01	0.053809	6.232161
CTR	23066.0	9.857525e-18	1.000022	-1.097932	-1.048677	-2.071337e-16	0.720001	13.826128
CPM	23066.0	-9.611087e-17	1.000022	-1.327883	-1.007683	-1.537265e-16	0.634852	12.788576
CPC	23066.0	-9.857525e-17	1.000022	-1.147050	-0.820311	0.000000e+00	0.388621	22.574160

Scaling can increase the computational complexity of algorithms, as it involves additionalcomputations to transform the data.

Step 4: Hierarchical Clustering - Dendrogram:

Dendrogram performed for Hierarchical using WARD and Euclidean Distance onthe Scaled Data such as “data1_scaled”.

In this Dendrogram, value of P = 10, which means that only the last 10 merged clusters are shown.



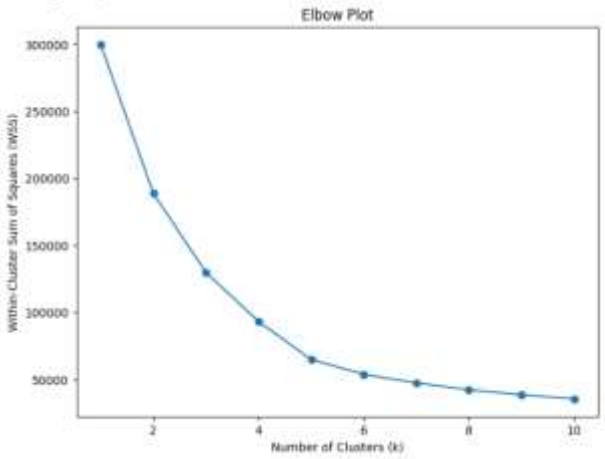
Hierarchical clustering with a dendrogram provides a visual representation of potential clusters, aiding in the selection of an optimal number of clusters for subsequent KMeans analysis.

Step 5: KMeans Clustering:

- **Elbow Plot**

We created an Elbow plot (n=10) and identified optimum number of clusters for k-means algorithm.

Elbow Plot (up to n=10)



For checking the Optimal number of clusters we use WSS (Within Sum Of Square)

As per the check
When we move from K=1 to K=2, We see that there is a significant drop in the value.
Also when we move from k=2 to k-3, k=3 to k=4, k=4 to k=5 there is a significant drop aswell.
k=5 to k=6, the drop in values reduces significantly.

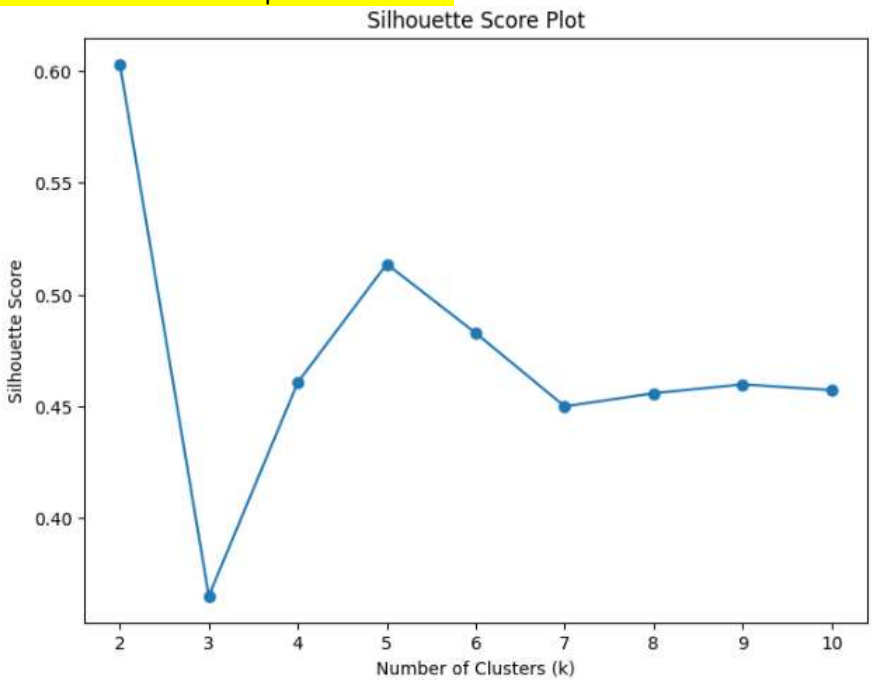
Hence In this case, the WSS is not significantly dropping beyond 5, so 5 is optimal number of clusters.

The Elbow Plot helps determine the optimal number of clusters by identifying the point where adding more clusters does not significantly reduce the within-cluster sum of squares (WSS).

- **Silhouette score:**

Then we Printed silhouette scores for up to 10 clusters and identified optimum number of clusters.

Silhouette scores for up to 10 clusters:



```
Number of Clusters (k) = 2: Silhouette Score = 0.602856419557812
Number of Clusters (k) = 3: Silhouette Score = 0.3652575679239419
Number of Clusters (k) = 4: Silhouette Score = 0.46072044314349486
Number of Clusters (k) = 5: Silhouette Score = 0.5135883146481809
Number of Clusters (k) = 6: Silhouette Score = 0.48271573962694464
Number of Clusters (k) = 7: Silhouette Score = 0.44997366925914933
Number of Clusters (k) = 8: Silhouette Score = 0.45584674165165107
Number of Clusters (k) = 9: Silhouette Score = 0.45983041055564045
Number of Clusters (k) = 10: Silhouette Score = 0.45726048689932824
```

Optimal number of clusters: 5

I have calculated Silhouette Score for scaled data using the silhouette_score() function.
The Silhouette Score is a measure of how similar an object is to its own cluster compared to other clusters, and it ranges from -1 to 1, with higher values indicating better clustering.

As per Elbow plot/scree-plot, we concluded that the optimal number of clusters should be 5.
Because 2 would be very less number of clusters.

Step 6: Conclusion:

- There are 23066 rows, and 19 columns into the Dataset.
- There are no duplicate values in dataframe.
- There are 4636 Null values in CTR, CPM, and CPC Columns.
- I have treated missing values in CPC, CTR, and CPM columns using the given formula
- It seems that there are Outliers into the Dataset
- We treated outliers using IQR method
- I have applied z-score method on the dataframe for scaling.
- I have plotted Dendrogram for value of P = 10
- Plotted elbow plot and got optimum value is 5
- As per Elbow plot/scree-plot, we concluded that the optimal number of clusters should be 5.
- I have created 5 clusters for the Dataset.

Conclusion after Clustering :

- When Click on Ads gets increases then Revenue also increases.
- When amount of money spent on specific ad variations within a specific campaign or ad set is increases then Revenue also increases.
- When impression count of the particular Advertisement increases then Revenue also increases

References:

The dataset "PCA+India+Data_Census.xlsx" utilized in this dimension reduction project was sourced from DataSpace.

Webiste: https://dataspace.mobi/

https://dataspace.mobi/dataset/pca-census-2011

- **Scikit-Learn Documentation on PCA**
  [Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825-2830.]

- **Towards Data Science - Understanding PCA (Principal Component Analysis**
  Author: Will Badr
  Reference: Published on Towards Data Science, a Medium publication.

- **Reduce Data Dimensionality using PCA – Python – GFG**

These references guided the application of dimension reduction techniques and contributed to the methodology employed in the project.