DATA MINING

Project Report
By Ruchita Parulekar

Table of Contents

Problem 1

Clustering:

Digital Ads Data:

The ads24x7 is a Digital Marketing company which has now got seed funding of $10 Million. They are expanding their wings in Marketing Analytics. They collected data from their Marketing Intelligence team and now wants you (their newly appointed data analyst) to segment type of ads based on the features provided. Use Clustering procedure to segment ads into homogeneous groups.

The following three features are commonly used in digital marketing:

***CPM = (Total Campaign Spend / Number of Impressions) * 1,000**. Note that the Total Campaign Spend refers to the 'Spend' Column in the dataset and the Number of Impressions refers to the 'Impressions' Column in the dataset.*

***CPC = Total Cost (spend) / Number of Clicks**. Note that the Total Cost (spend) refers to the 'Spend' Column in the dataset and the Number of Clicks refers to the 'Clicks' Column in the dataset.*

***CTR = Total Measured Clicks / Total Measured Ad Impressions x 100.** Note that the Total Measured Clicks refers to the 'Clicks' Column in the dataset and the Total Measured Ad Impressions refers to the 'Impressions' Column in the dataset.*

1. **Read the data and perform basic analysis such as printing a few rows (head and tail), info, data summary, null values duplicate values, etc.**

Solution:

```
In [5]: data_df.head(10)
```

Out[5]:

	Timestamp	InventoryType	Ad - Length	Ad- Width	Ad Size	Ad Type	Platform	Device Type	Format	Available_Impressions	Matched_Queries	Impressions	Clicks	Spend	F
0	2020-9-2-17	Format1	300	250	75000	Inter222	Video	Desktop	Display	1806	325	323	1	0.00	0
1	2020-9-2-10	Format1	300	250	75000	Inter227	App	Mobile	Video	1780	285	285	1	0.00	0
2	2020-9-1-22	Format1	300	250	75000	Inter222	Video	Desktop	Display	2727	356	355	1	0.00	0
3	2020-9-3-20	Format1	300	250	75000	Inter228	Video	Mobile	Video	2430	497	495	1	0.00	0
4	2020-9-4-15	Format1	300	250	75000	Inter217	Web	Desktop	Video	1218	242	242	1	0.00	0
5	2020-9-4-5	Format1	300	250	75000	Inter219	Video	Desktop	Display	490	64	64	2	0.00	0
6	2020-9-4-6	Format1	300	250	75000	Inter221	App	Mobile	Video	1197	202	202	1	0.01	0
7	2020-9-6-7	Format1	300	250	75000	Inter228	Video	Mobile	Video	1363	198	196	1	0.00	0
8	2020-9-8-6	Format1	300	250	75000	Inter223	Web	Mobile	Video	1402	137	136	1	0.00	0
9	2020-9-11-17	Format1	300	250	75000	Inter228	Video	Mobile	Display	1816	312	311	1	0.00	0

```
In [6]: data_df.tail(10)
```

Out[6]:

	Timestamp	InventoryType	Ad - Length	Ad- Width	Ad Size	Ad Type	Platform	Device Type	Format	Available_Impressions	Matched_Queries	Impressions	Clicks	Spe
23056	2020-11-23-4	Format4	120	600	72000	Inter223	Web	Mobile	Video	2	2	2	1	0.
23057	2020-11-20-2	Format4	120	600	72000	Inter224	Web	Desktop	Display	5	2	2	1	0.
23058	2020-11-4-3	Format5	720	300	216000	Inter223	Web	Mobile	Video	1	1	1	1	0.
23059	2020-11-13-4	Format5	720	300	216000	Inter228	Video	Mobile	Display	2	2	2	1	0.
23060	2020-11-16-5	Format4	120	600	72000	Inter225	Video	Mobile	Display	4	4	4	1	0.
23061	2020-9-13-7	Format5	720	300	216000	Inter220	Web	Mobile	Video	1	1	1	1	0.
23062	2020-11-2-7	Format5	720	300	216000	Inter224	Web	Desktop	Video	3	2	2	1	0.
23063	2020-9-14-22	Format5	720	300	216000	Inter218	App	Mobile	Video	2	1	1	1	0.
23064	2020-11-18-2	Format4	120	600	72000	inter230	Video	Mobile	Video	7	1	1	1	0.
23065	2020-9-14-0	Format5	720	300	216000	Inter221	App	Mobile	Video	2	2	2	1	0.

```
In [7]: data_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23066 entries, 0 to 23065
Data columns (total 19 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Timestamp              23066 non-null  object 
 1   InventoryType          23066 non-null  object 
 2   Ad - Length            23066 non-null  int64  
 3   Ad- Width              23066 non-null  int64  
 4   Ad Size                23066 non-null  int64  
 5   Ad Type                23066 non-null  object 
 6   Platform               23066 non-null  object 
 7   Device Type            23066 non-null  object 
 8   Format                 23066 non-null  object 
 9   Available_Impressions  23066 non-null  int64  
 10  Matched_Queries        23066 non-null  int64  
 11  Impressions            23066 non-null  int64  
 12  Clicks                 23066 non-null  int64  
 13  Spend                  23066 non-null  float64
 14  Fee                    23066 non-null  float64
 15  Revenue                23066 non-null  float64
 16  CTR                    18330 non-null  float64
 17  CPM                    18330 non-null  float64
 18  CPC                    18330 non-null  float64
dtypes: float64(6), int64(7), object(6)
memory usage: 3.3+ MB
```

In [10]:
```
data_df.describe()
```

Out[10]:

	Ad - Length	Ad- Width	Ad Size	Available_Impressions	Matched_Queries	Impressions	Clicks	Spend	Fee	Revenue
count	23066.000000	23066.000000	23066.000000	2.306600e+04	2.306600e+04	2.306600e+04	23066.000000	23066.000000	23066.000000	23066.000000
mean	385.163097	337.896037	96674.468048	2.432044e+06	1.295099e+06	1.241520e+06	10678.518816	2706.625689	0.335123	1924.252331
std	233.651434	203.092885	61538.329557	4.742888e+06	2.512970e+06	2.429400e+06	17353.409363	4067.927273	0.031963	3105.238410
min	120.000000	70.000000	33600.000000	1.000000e+00	1.000000e+00	1.000000e+00	1.000000	0.000000	0.210000	0.000000
25%	120.000000	250.000000	72000.000000	3.367225e+04	1.828250e+04	7.990500e+03	710.000000	85.180000	0.330000	55.365375
50%	300.000000	300.000000	72000.000000	4.837710e+05	2.580875e+05	2.252900e+05	4425.000000	1425.125000	0.350000	926.335000
75%	720.000000	600.000000	84000.000000	2.527712e+06	1.180700e+06	1.112428e+06	12793.750000	3121.400000	0.350000	2091.338150
max	728.000000	600.000000	216000.000000	2.759286e+07	1.470202e+07	1.419477e+07	143049.000000	26931.870000	0.350000	21276.180000

In [11]:
```
data_df.duplicated().sum()
```

Out[11]: 0

In [13]:
```
data_df.isnull().sum()
```

Out[13]:
```
Timestamp                 0
InventoryType             0
Ad - Length               0
Ad- Width                 0
Ad Size                   0
Ad Type                   0
Platform                  0
Device Type               0
Format                    0
Available_Impressions     0
Matched_Queries           0
Impressions               0
Clicks                    0
Spend                     0
Fee                       0
Revenue                   0
CTR                    4736
CPM                    4736
CPC                    4736
dtype: int64
```

There are missing values in CTR CPM and CPC.

2. Treat missing values in CPC, CTR and CPM using the formula given.

Solution:

The missing values in CPC, CTR and CPM are treated by writing a user-defined function and calling it.

CPM = (Total Campaign Spend / Number of Impressions) * 1,000

CPC = Total Cost (spend) / Number of Clicks

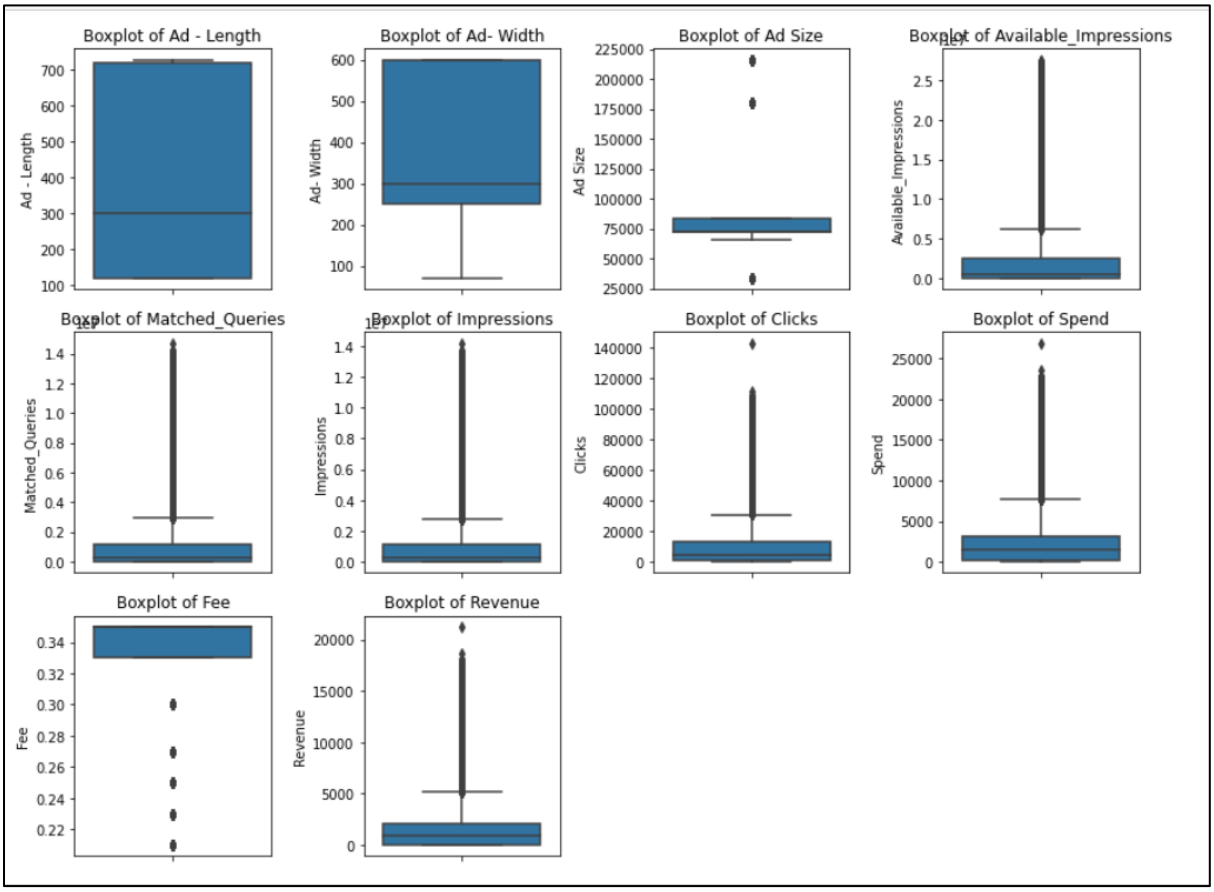
CTR = Total Measured Clicks / Total Measured Ad Impressions x 100.

The missing values are treated using the above formulae and user defined function and calling it using return function.

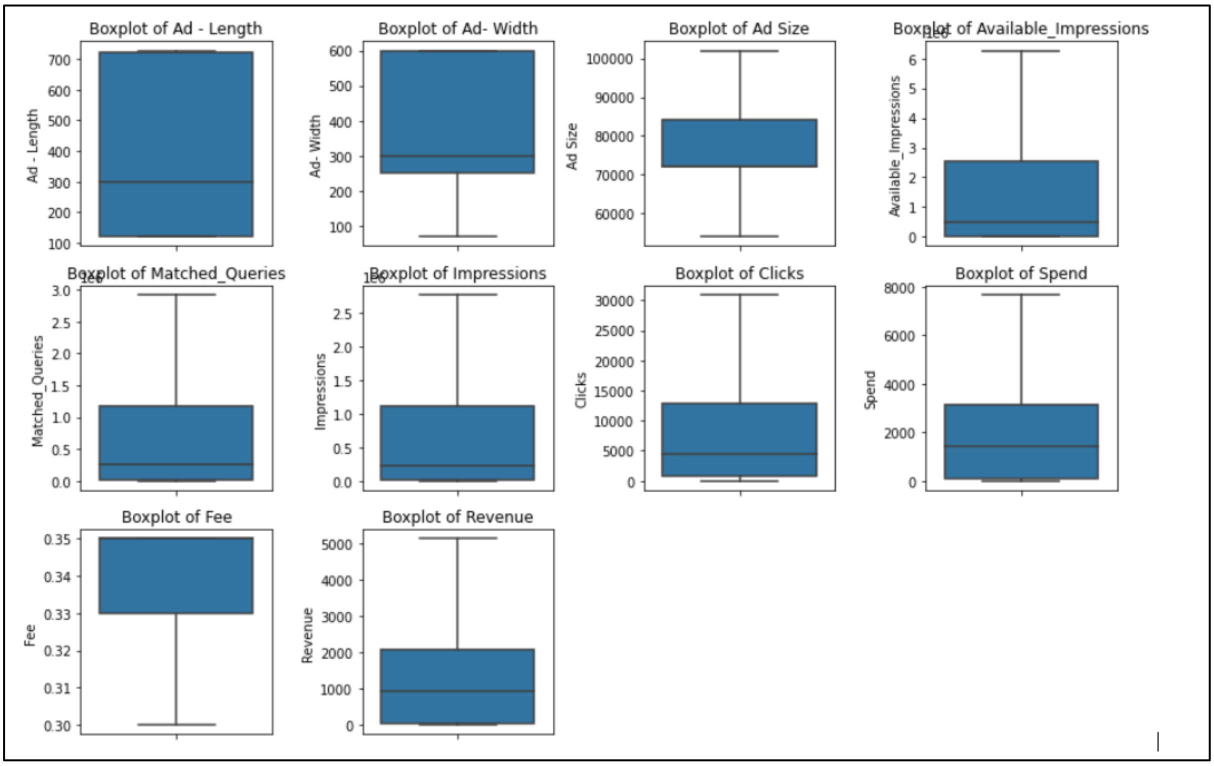
The above data set has columns timestamp, inventory type which are not very useful for clustering, also columns CTR, CPM, CPC are dependent variables, so we need to drop these columns.

3. **Check if there are any outliers. Do you think treating outliers is necessary for K-Means clustering? Based on your judgement decide whether to treat outliers and if yes, which method to employ. (As an analyst your judgement may be different from another analyst).**

   Solution:



It is better to identify and remove outliers before applying K-means clustering algorithm.



4. Perform z-score scaling and discuss how it affects the speed of the algorithm.

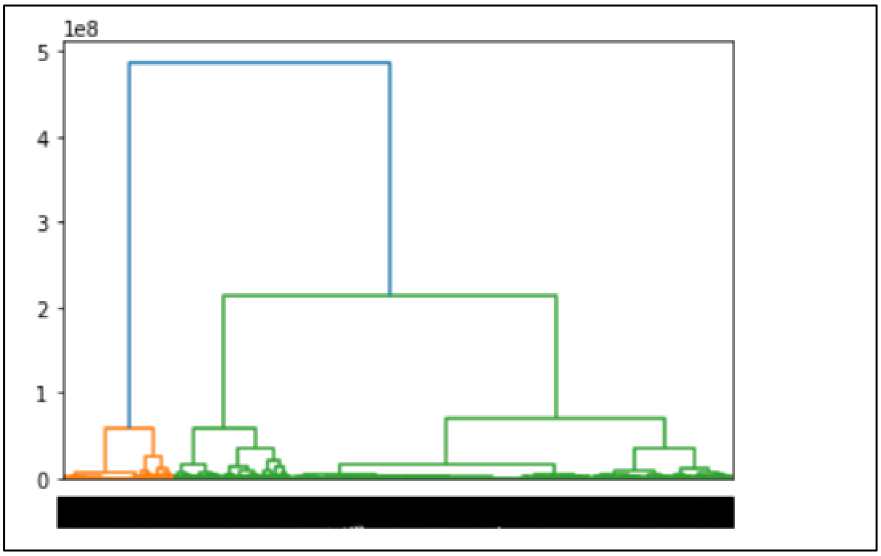
Solution:

	Ad - Length	Ad- Width	Ad Size	Available_Impressions	Matched_Queries	Impressions	Clicks	Spend	Fee	Revenue
0	-0.364496	-0.432797	-0.102518	-0.755333	-0.778949	-0.768478	-0.867488	-0.89317	0.535724	-0.880093
1	-0.364496	-0.432797	-0.102518	-0.755345	-0.778988	-0.768516	-0.867488	-0.89317	0.535724	-0.880093
2	-0.364496	-0.432797	-0.102518	-0.754900	-0.778919	-0.768445	-0.867488	-0.89317	0.535724	-0.880093
3	-0.364496	-0.432797	-0.102518	-0.755040	-0.778781	-0.768302	-0.867488	-0.89317	0.535724	-0.880093
4	-0.364496	-0.432797	-0.102518	-0.755610	-0.779030	-0.768560	-0.867488	-0.89317	0.535724	-0.880093

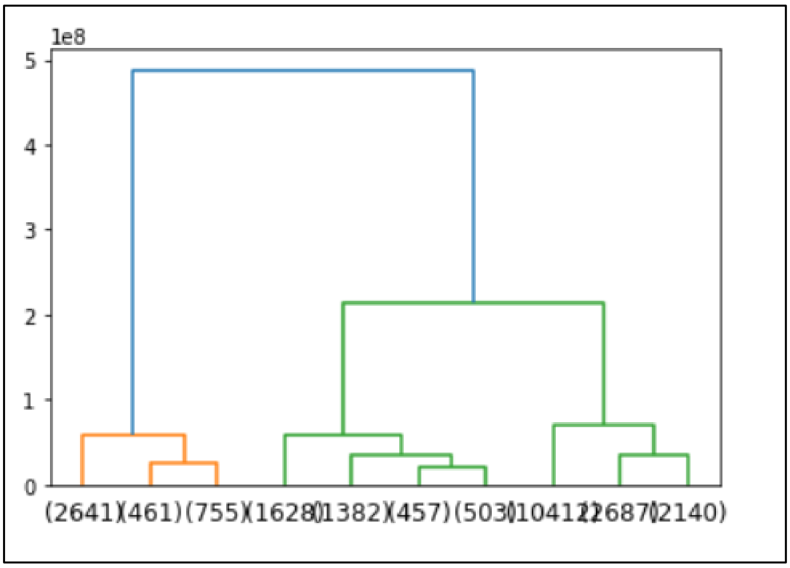
5. Perform Hierarchical by constructing a Dendrogram using WARD and Euclidean distance.

Solution:

Constructing Dendrogram using WARD:



Viewing the last 10 merged clusters using truncate, given p=10, we get:



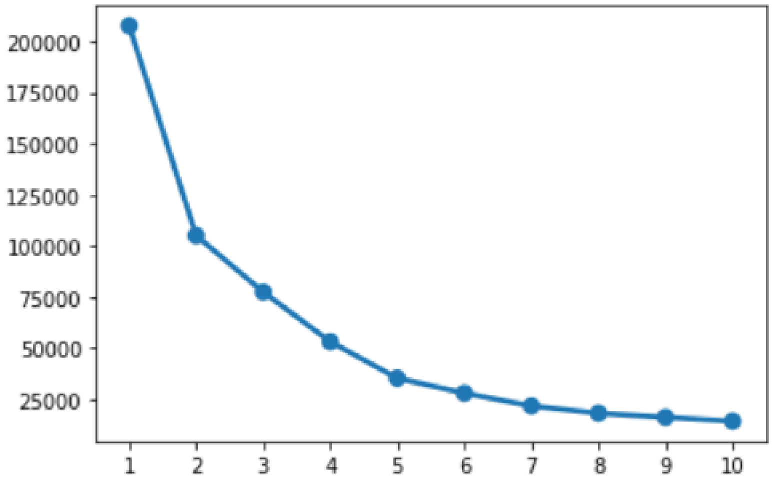
6. Make Elbow plot (up to n=10) and identify optimum number of clusters for k-means algorithm.

Solution:

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 there is a significant drop as well.

But from k=4 to k=5 , k=5 to k=6 , the drop in values reduces significantly.

In other words, the WSS is not significantly dropping beyond 4, so 4 is optimal number of clusters.



7. **Print silhouette scores for up to 10 clusters and identify optimum number of clusters.**

Solution:

the Silhouette Score for the values of K from 2 to 10

```
For n_clusters=2, the silhouette score is 0.5204199824415183
For n_clusters=3, the silhouette score is 0.41665752126605
For n_clusters=4, the silhouette score is 0.4754240028101093
For n_clusters=5, the silhouette score is 0.5503625819255761
For n_clusters=6, the silhouette score is 0.5509515863487943
For n_clusters=7, the silhouette score is 0.5799614923116835
For n_clusters=8, the silhouette score is 0.583635767867305
For n_clusters=9, the silhouette score is 0.5909492227519072
For n_clusters=10, the silhouette score is 0.5954912596231999
```

8. **Profile the ads based on optimum number of clusters using silhouette score and your domain understanding [Hint: Group the data by clusters and take sum or mean to identify trends in Clicks, spend, revenue, CPM, CTR, & CPC based on Device Type. Make bar plots].**

Solution:

	Ad - Length	Ad- Width	Ad Size	Available_Impressions	Matched_Queries	Impressions	Clicks	Spend	Fee	Revenue	Clus_kmeans4
0	300.0	250.0	75000.0	1806.0	325.0	323.0	1.0	0.0	0.35	0.0	1
1	300.0	250.0	75000.0	1780.0	285.0	285.0	1.0	0.0	0.35	0.0	1
2	300.0	250.0	75000.0	2727.0	356.0	355.0	1.0	0.0	0.35	0.0	1
3	300.0	250.0	75000.0	2430.0	497.0	495.0	1.0	0.0	0.35	0.0	1
4	300.0	250.0	75000.0	1218.0	242.0	242.0	1.0	0.0	0.35	0.0	1

Clus_kmeans4	Ad - Length	Ad- Width	Ad Size	Available_Impressions	Matched_Queries	Impressions	Clicks	Spend	Fee	Revenue	freq
0	465.202289	199.201294	72970.868375	5.715755e+06	2.813606e+06	2.678101e+06	11316.414780	5759.011398	0.313135	3892.227369	4019
1	361.458745	450.436228	84805.650187	1.693707e+05	9.146048e+04	7.931212e+04	6369.075280	625.744073	0.349967	406.837553	12035
2	441.898028	123.449944	61189.683662	2.004554e+06	9.546327e+05	9.122723e+05	3603.598623	1652.710983	0.349051	1077.348276	5374
3	176.805861	554.884005	75446.886447	7.878064e+05	5.514885e+05	4.654792e+05	30590.395681	6351.878309	0.307051	4336.248727	1638

9. Conclude the project by providing summary of your learnings.

Solution:

- The dataset has 25857 rows and 19 columns.
- The missing values in CPC, CTR and CPM are treated by using the formulae given and writing a user-defined function and calling it.
- We check for outliers; we can see there are outliers in the variables.
- Dendrogram is the visualization and linkage is for computing the distances and merging the clusters from n to 1.
- The output of Linkage is visualized by Dendrogram.
-
- We will create linkage using Ward's method and run linkage function on the usable columns of the data.
- The linkage now stores the various distance at which the n clusters are sequentially merged into a single cluster.
- using fit – transform function and viewing the output - The data frame is now stored in an array.
- Using this array, we can now perform k-means.
- The one requirement before we run the k-means algorithm, is to know how many clusters we require as output.
- We map the elbow plot using WSS values.
- From the plot we have following observations:
- 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 there is a significant drop as well.

- But from k=4 to k=5, k=5 to k=6, the drop in values reduces significantly.
- In other words, the WSS is not significantly dropping beyond 4,
- So, 4 is optimal number of clusters.

Problem 2

PCA

PCA FH (FT): Primary census abstract for female headed households excluding institutional households (India & States/UTs - District Level), Scheduled tribes - 2011 PCA for Female Headed Household Excluding Institutional Household. The Indian Census has the reputation of being one of the best in the world. The first Census in India was conducted in the year 1872. This was conducted at different points of time in different parts of the country. In 1881 a Census was taken for the entire country simultaneously. Since then, Census has been conducted every ten years, without a break. Thus, the Census of India 2011 was the fifteenth in this unbroken series since 1872, the seventh after independence and the second census of the third millennium and twenty first century. The census has been uninterruptedly continued despite of several adversities like wars, epidemics, natural calamities, political unrest, etc. The Census of India is conducted under the provisions of the Census Act 1948 and the Census Rules, 1990. The Primary Census Abstract which is important publication of 2011 Census gives basic information on Area, Total Number of Households, Total Population, Scheduled Castes, Scheduled Tribes Population, Population in the age group 0-6, Literates, Main Workers and Marginal Workers classified by the four broad industrial categories, namely, (i) Cultivators, (ii) Agricultural Laborers, (iii) Household Industry Workers, and (iv) Other Workers and also non-Workers. The characteristics of the Total Population include Scheduled Castes, Scheduled Tribes, Institutional and Houseless Population and are presented by sex and rural-urban residence. Census 2011 covered 35 States/Union Territories, 640 districts, 5,924 sub-districts, 7,935 Towns and 6,40,867 Villages.
The data collected has so many variables thus making it difficult to find useful details without using Data Science Techniques. You are tasked to perform detailed EDA and identify Optimum Principal Components that explains the most variance in data. Use Sklearn only.

1. **Read the data and perform basic checks like checking head, info, summary, nulls, and duplicates, etc.**

Solution:

	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
0	1	1	Jammu & Kashmir	Kupwara	7707	23388	29796	5862	6196	3	...	1150	749	180	
1	1	2	Jammu & Kashmir	Badgam	6218	19585	23102	4482	3733	7	...	525	715	123	
2	1	3	Jammu & Kashmir	Leh(Ladakh)	4452	6546	10964	1082	1018	3	...	114	188	44	
3	1	4	Jammu & Kashmir	Kargil	1320	2784	4206	563	677	0	...	194	247	61	
4	1	5	Jammu & Kashmir	Punch	11654	20591	29981	5157	4587	20	...	874	1928	465	
5	1	6	Jammu & Kashmir	Rajouri	16345	25290	37426	6155	5294	2588	...	1808	3536	1277	
6	1	7	Jammu & Kashmir	Kathua	12510	22793	30491	3928	3200	5357	...	502	561	160	
7	1	8	Jammu & Kashmir	Baramula	9414	22960	30509	4246	4099	0	...	849	878	168	
8	1	9	Jammu & Kashmir	Bandipore	3814	10319	13058	1646	1779	0	...	515	901	108	
9	1	10	Jammu & Kashmir	Srinagar	15095	39014	52278	6269	5704	11	...	308	432	10	

10 rows × 61 columns

	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
630	33	631	Tamil Nadu	Krishnagiri	65952	82958	134294	10629	10083	13602	...	1027	2295	101
631	33	632	Tamil Nadu	Coimbatore	133255	125297	239223	12101	11624	21087	...	723	2137	8
632	33	633	Tamil Nadu	Tiruppur	98258	77174	163526	7201	6957	13016	...	401	1574	5
633	34	634	Puducherry	Yanam	2219	2618	4659	281	275	496	...	11	30	0
634	34	635	Puducherry	Puducherry	37786	47268	80943	5629	5407	10062	...	528	951	10
635	34	636	Puducherry	Mahe	3333	8154	11781	1146	1203	21	...	32	47	0
636	34	637	Puducherry	Karaikal	10612	12346	21691	1544	1533	2234	...	155	337	3
637	35	638	Andaman & Nicobar Island	Nicobars	1275	1549	2630	227	225	0	...	104	134	9
638	35	639	Andaman & Nicobar Island	North & Middle Andaman	3762	5200	8012	723	664	0	...	136	172	24
639	35	640	Andaman & Nicobar Island	South Andaman	7975	11977	18049	1470	1358	0	...	173	122	6

10 rows × 61 columns

```
(640, 61)
```

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

	State Code	Dist.Code	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	F_SC	M_ST
count	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	6191.807813
std	9.426486	184.896367	48135.405475	73384.511114	113600.717282	11500.906881	11326.294567	14426.373130	21727.887713	9912.668948
min	1.000000	1.000000	350.000000	391.000000	698.000000	56.000000	56.000000	0.000000	0.000000	0.000000
25%	9.000000	160.750000	19484.000000	30228.000000	46517.750000	4733.750000	4672.250000	3466.250000	5603.250000	293.750000
50%	18.000000	320.500000	35837.000000	58339.000000	87724.500000	9159.000000	8663.000000	9591.500000	13709.000000	2333.500000
75%	24.000000	480.250000	68892.000000	107918.500000	164251.750000	16520.250000	15902.250000	19429.750000	29180.000000	7658.000000
max	35.000000	640.000000	310450.000000	485417.000000	750392.000000	96223.000000	95129.000000	103307.000000	156429.000000	96785.000000

8 rows × 59 columns

```
State Code        0
Dist.Code         0
State             0
Area Name         0
No_HH             0
                 ..
MARG_HH_0_3_F     0
MARG_OT_0_3_M     0
MARG_OT_0_3_F     0
NON_WORK_M        0
NON_WORK_F        0
Length: 61, dtype: int64
```

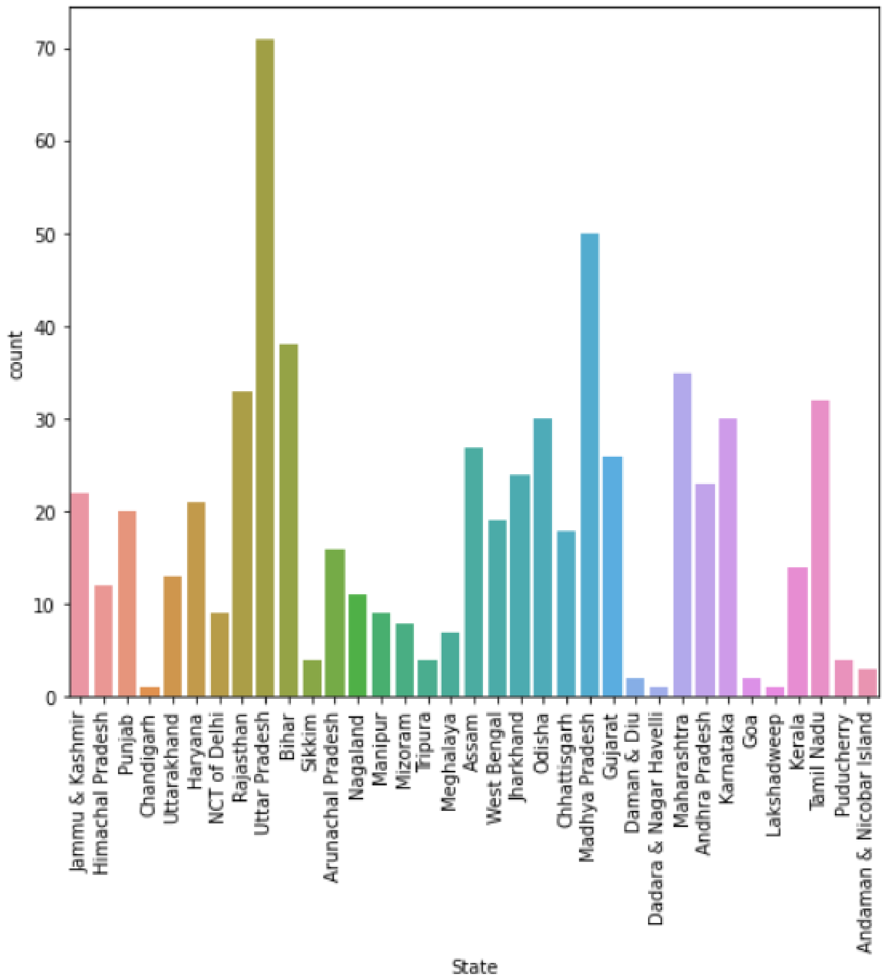
2. **Perform detailed Exploratory analysis by creating certain questions like (i) Which state has highest gender ratio and which has the lowest? (ii) Which district has the highest & lowest gender ratio? (Example Questions). Pick 5 variables out of the given 24 variables below for EDA: No_HH, TOT_M, TOT_F, M_06, F_06, M_SC, F_SC, M_ST, F_ST, M_LIT, F_LIT, M_ILL, F_ILL, TOT_WORK_M, TOT_WORK_F, MAINWORK_M, MAINWORK_F, MAIN_CL_M, MAIN_CL_F, MAIN_AL_M, MAIN_AL_F, MAIN_HH_M, MAIN_HH_F, MAIN_OT_M, MAIN_OT_F**

Solution:

I have picked 5 Variables such as 'TOT_M', 'TOT_F','M_LIT','F_LIT', and 'TOT_WORK_M'. And comparing those 5 variable against 'State' and 'Dist. Code'.

Which State has the highest Population?

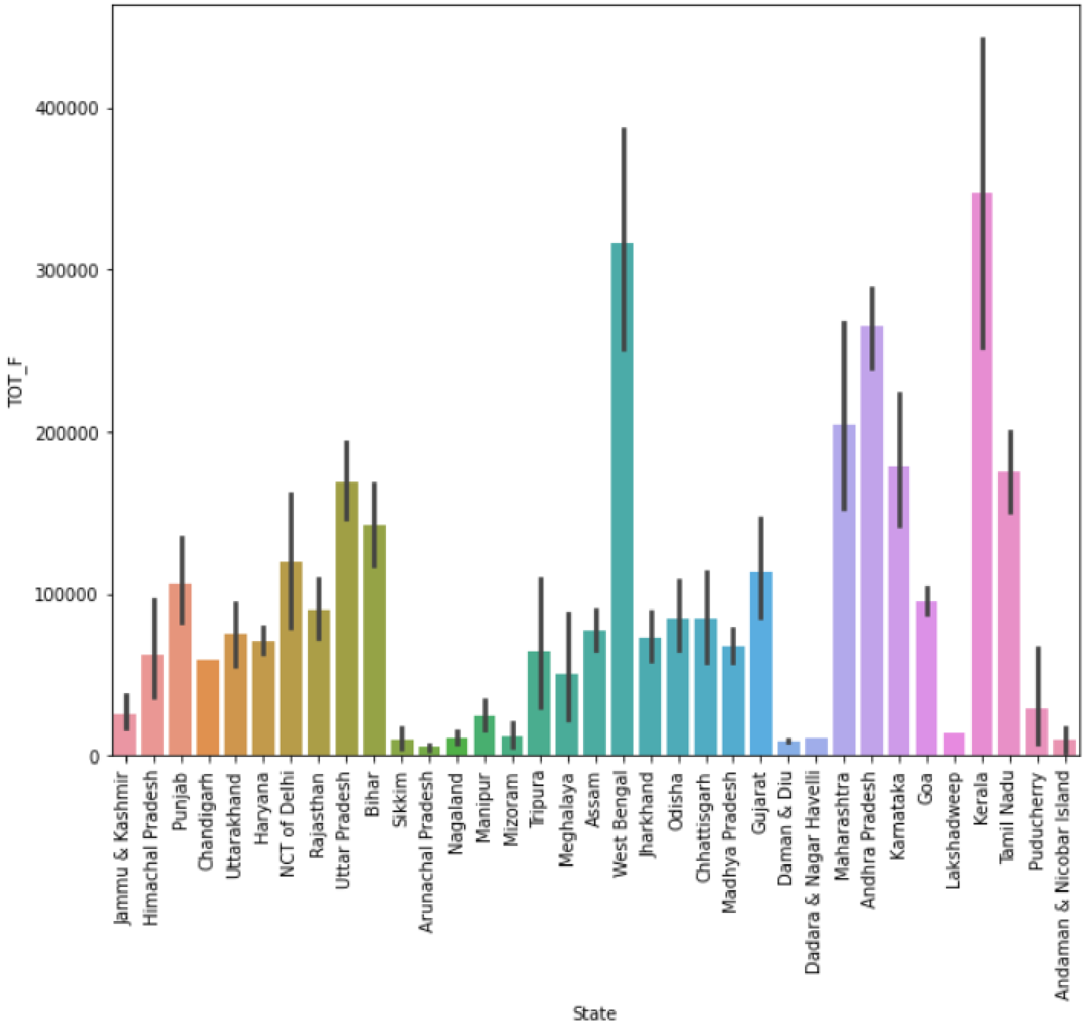
```
<AxesSubplot:xlabel='State', ylabel='count'>
```



Which state has highest Total population of Female?

Which state has lowest Total population of Female?

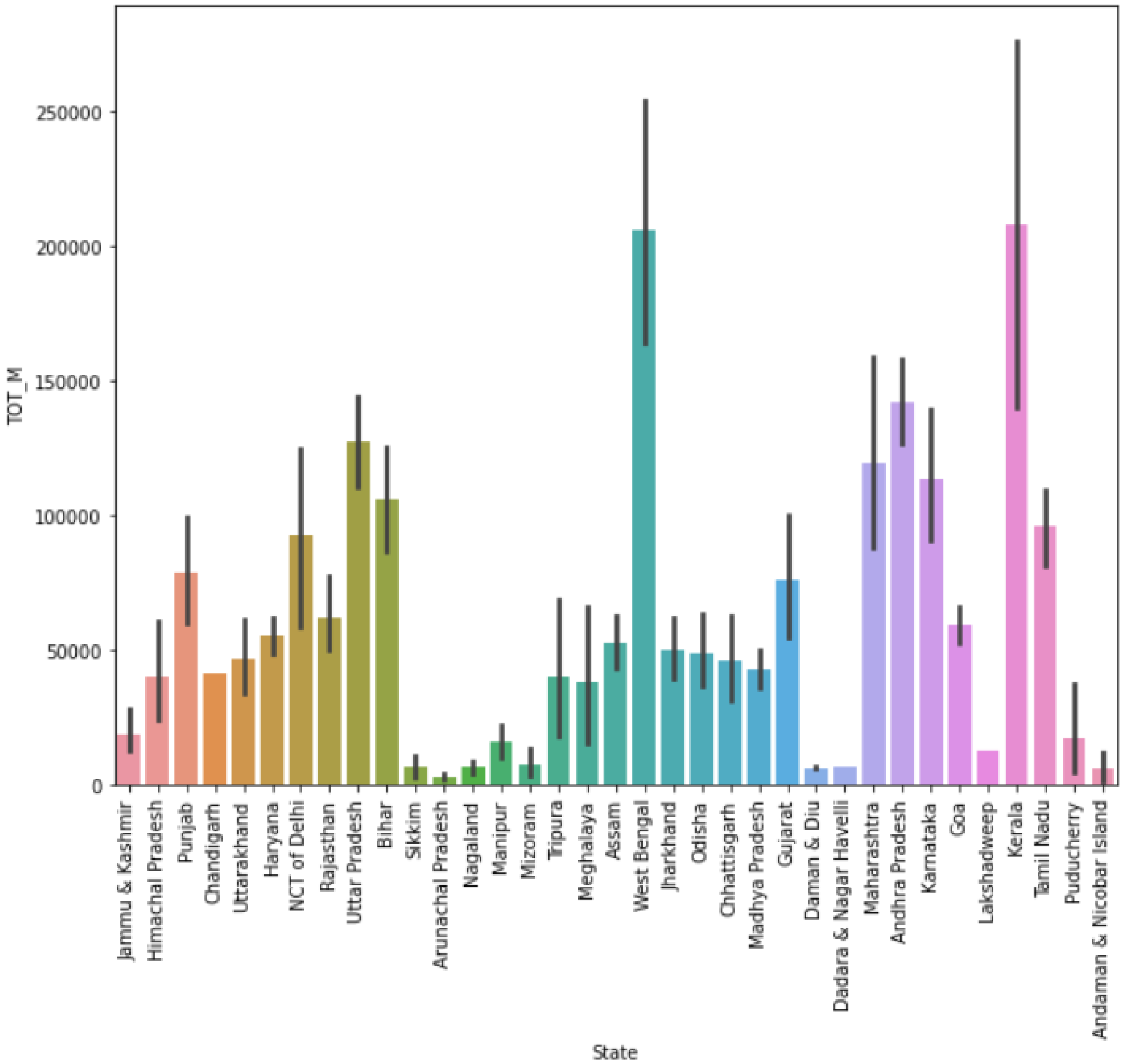
```
<AxesSubplot:xlabel='State', ylabel='TOT_F'>
```



Which state has highest Total population of Male?

Which state has lowest Total population of Male?

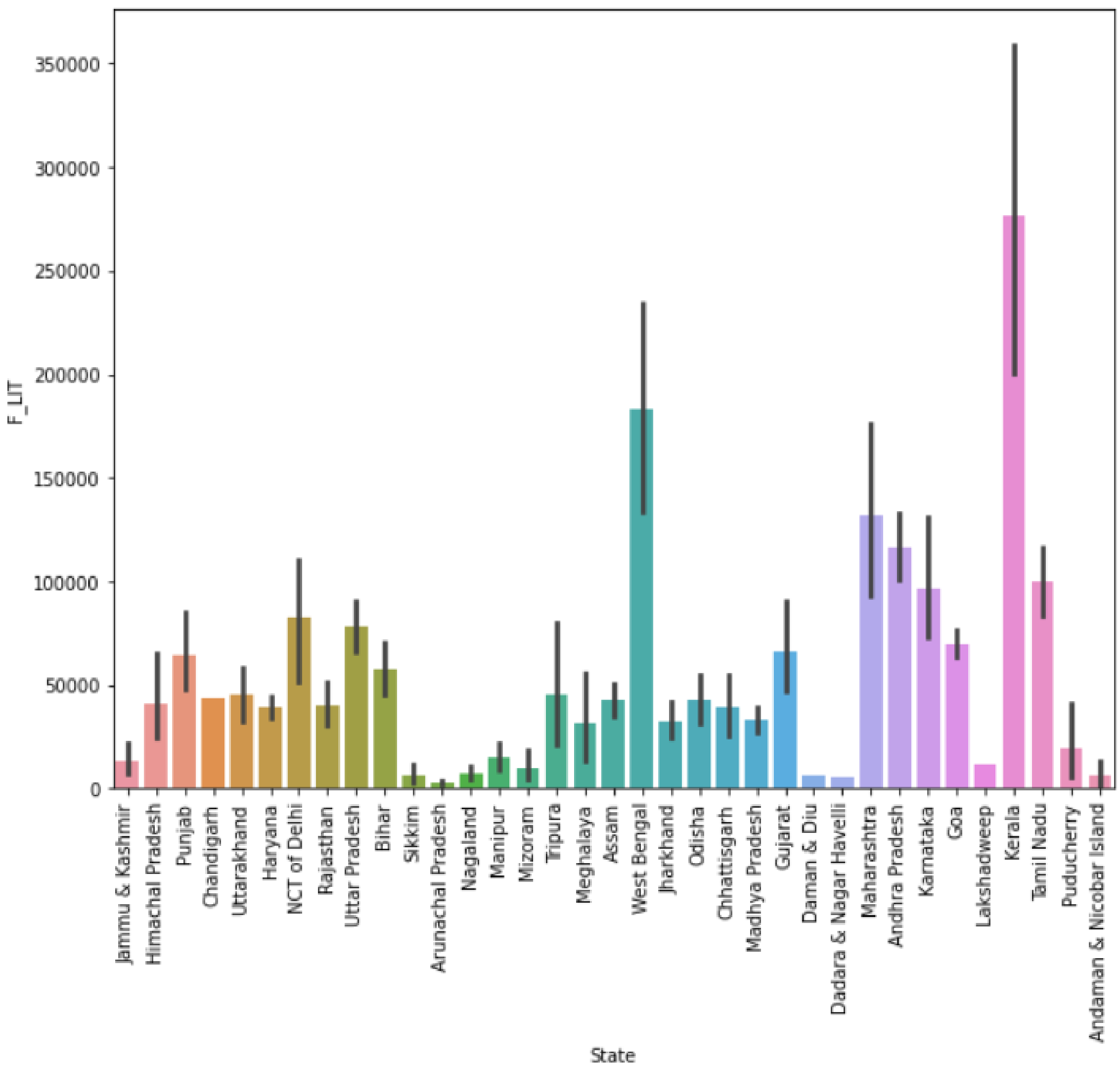
<AxesSubplot:xlabel='State', ylabel='TOT_M'>



Which state has highest Literate population of Female?

Which state has lowest Literate population of Female?

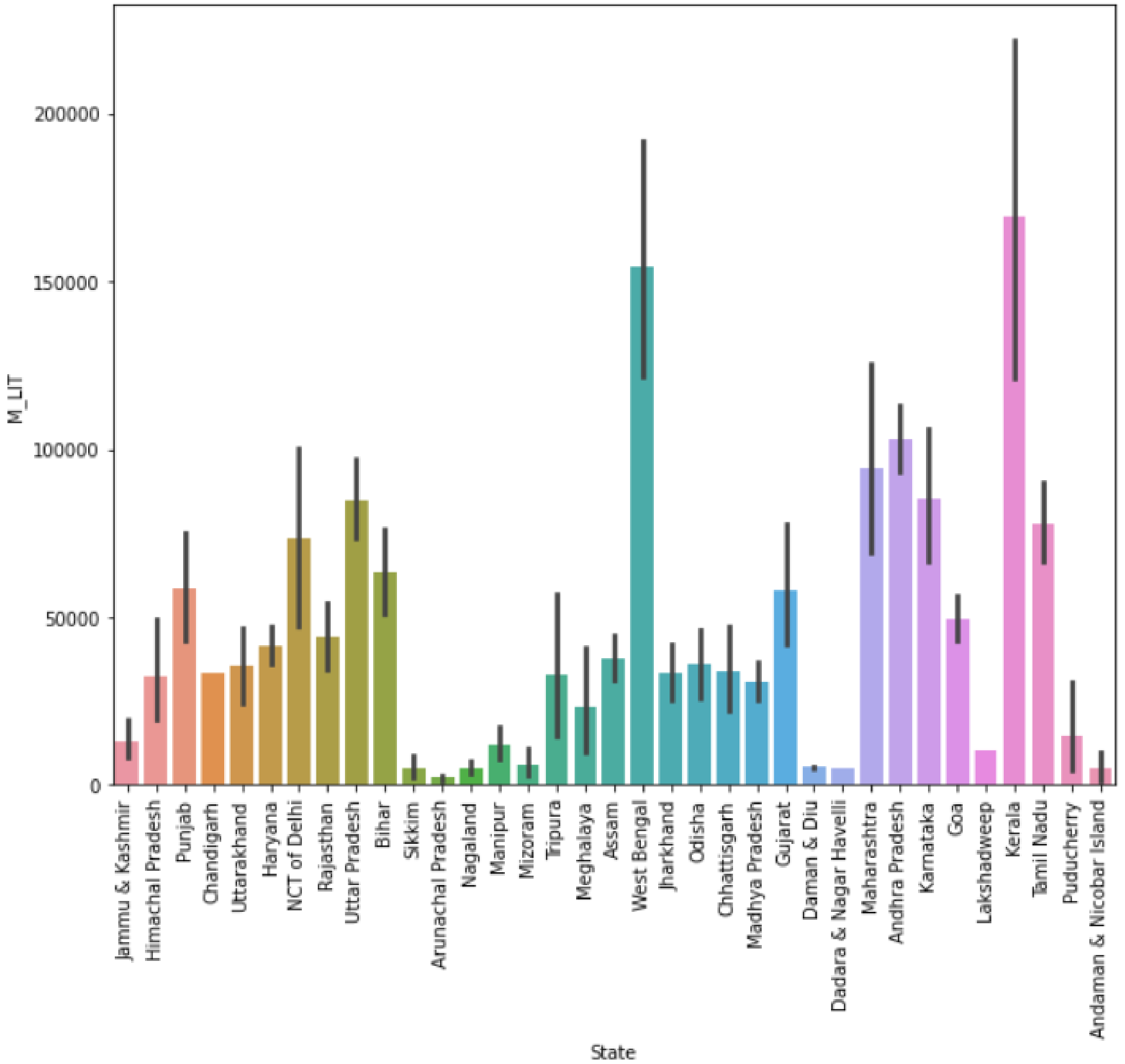
```
<AxesSubplot:xlabel='State', ylabel='F_LIT'>
```



Which state has highest Literate population of Male?

Which state has lowest Literate population of Male?

```
<AxesSubplot:xlabel='State', ylabel='M_LIT'>
```



3. We choose not to treat outliers for this case. Do you think that treating outliers for this case is necessary?

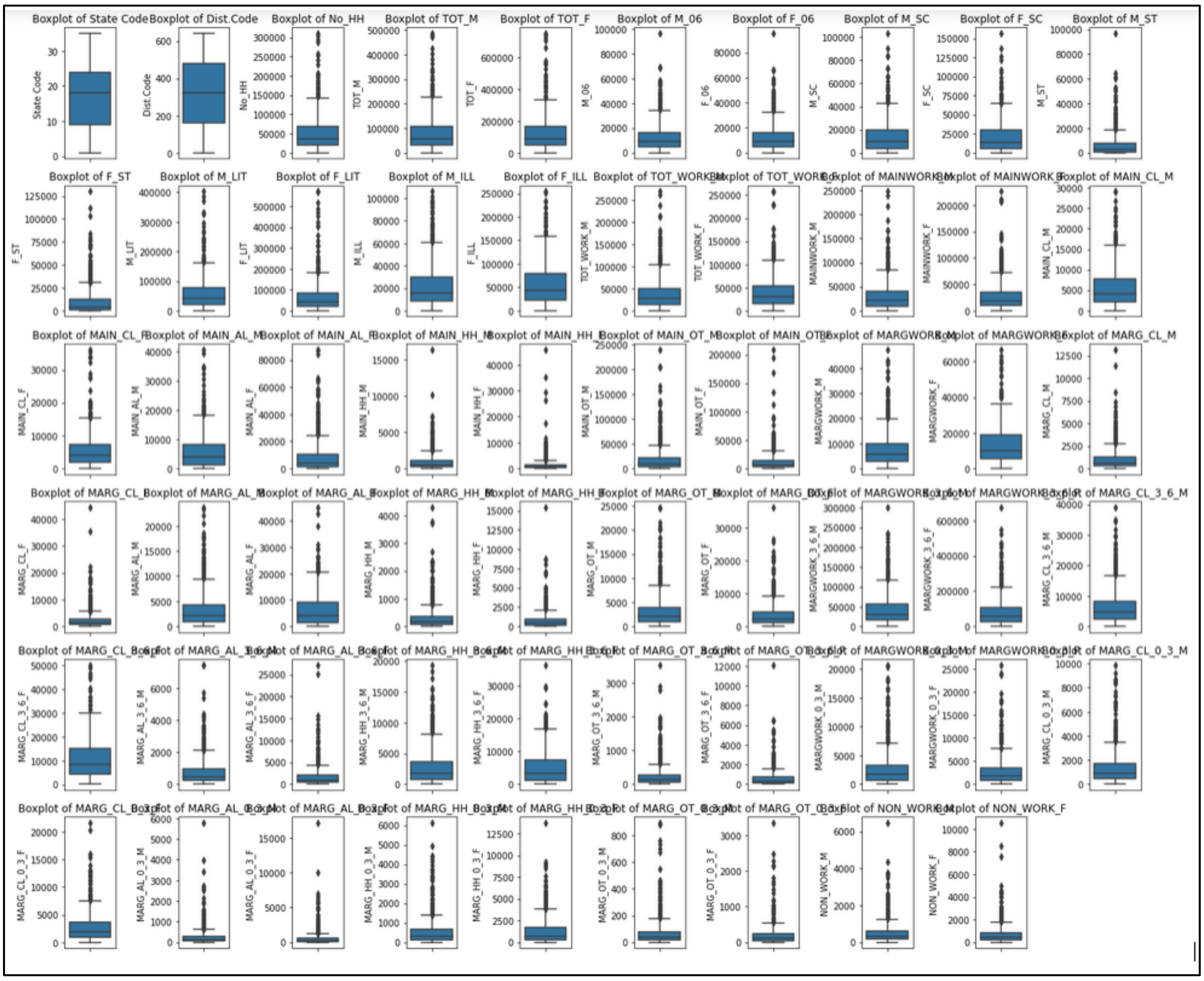
Solution:

Outliers' treatment is not necessary unless they are the result from a processing mistake or wrong measurement. True outliers must be kept in the data.

4. Scale the Data using z-score method. Does scaling have any impact on outliers? Compare boxplots before and after scaling and comment.

Solution:

Outliers before Scaling

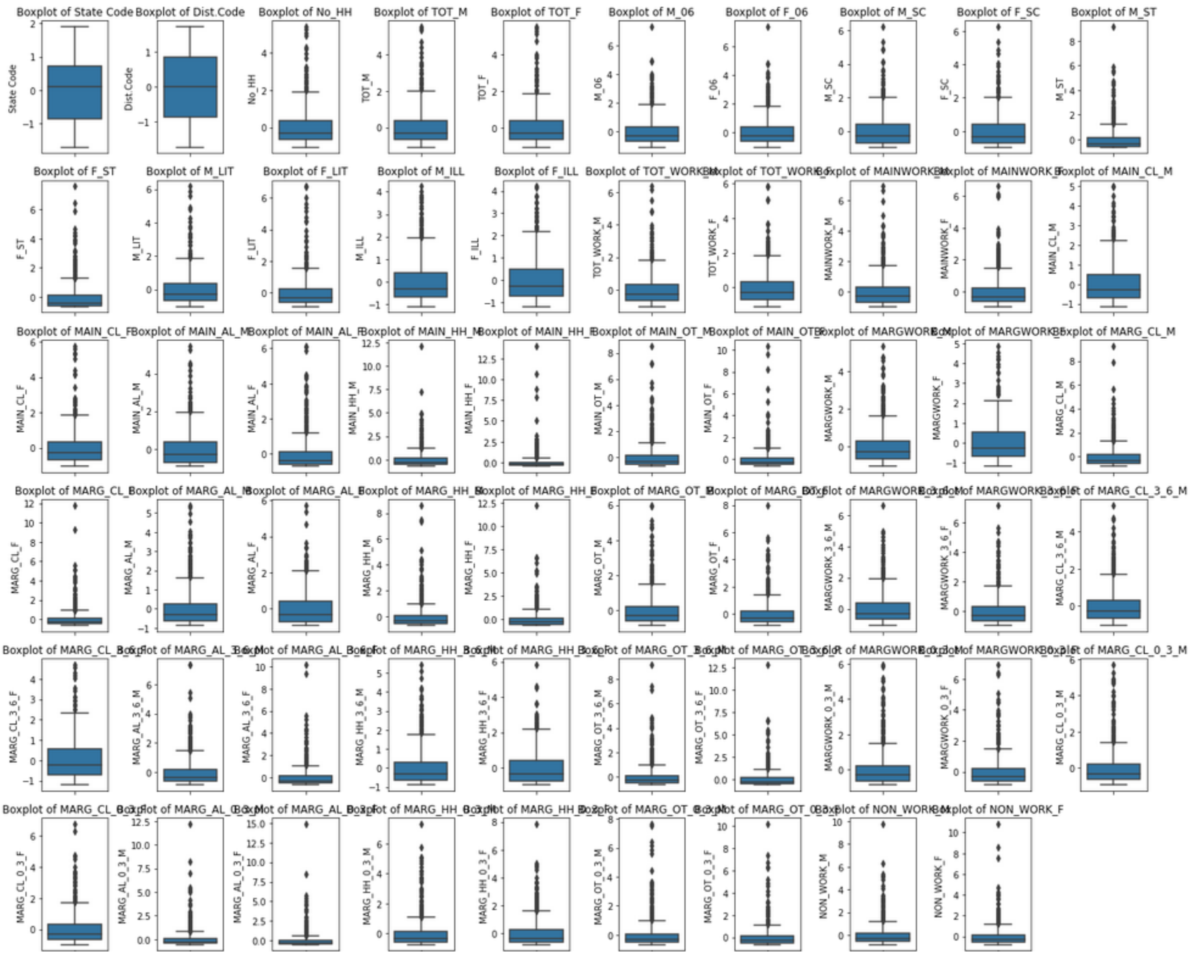


Scaled Data

	Dist.Code	No_HH	TOT_M	TOT_F	M_06	F_06	M_SC	F_SC	M_ST	F_ST	...	MARG_CL_0_3_M	MARG_CL_0_3_F	MARG_AL_0_3_M	MARG_AL_0_3_F	MA
0	-1.73	-0.90	-0.77	-0.82	-0.56	-0.51	-0.96	-0.96	-0.42	-0.48	...	-0.16	-0.72	-0.16	-0.29	
1	-1.72	-0.94	-0.82	-0.87	-0.68	-0.73	-0.96	-0.96	-0.58	-0.61	...	-0.58	-0.73	-0.28	-0.29	
2	-1.72	-0.97	-1.00	-0.98	-0.98	-0.97	-0.96	-0.96	-0.04	-0.03	...	-0.86	-0.92	-0.46	-0.42	
3	-1.71	-1.04	-1.05	-1.04	-1.02	-1.00	-0.96	-0.96	-0.36	-0.39	...	-0.81	-0.90	-0.42	-0.39	
4	-1.71	-0.82	-0.81	-0.81	-0.62	-0.65	-0.96	-0.96	0.15	0.04	...	-0.35	-0.30	0.47	0.43	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
635	1.71	-1.00	-0.98	-0.97	-0.97	-0.95	-0.96	-0.96	-0.63	-0.64	...	-0.91	-0.97	-0.55	-0.50	
636	1.71	-0.84	-0.92	-0.89	-0.94	-0.92	-0.80	-0.77	-0.63	-0.64	...	-0.83	-0.87	-0.55	-0.49	
637	1.72	-1.04	-1.07	-1.05	-1.05	-1.04	-0.96	-0.96	-0.52	-0.53	...	-0.87	-0.94	-0.53	-0.50	
638	1.72	-0.99	-1.02	-1.01	-1.01	-1.00	-0.96	-0.96	-0.62	-0.64	...	-0.84	-0.93	-0.50	-0.46	
639	1.73	-0.90	-0.93	-0.92	-0.94	-0.94	-0.96	-0.96	-0.61	-0.62	...	-0.82	-0.95	-0.54	-0.50	

640 rows × 58 columns

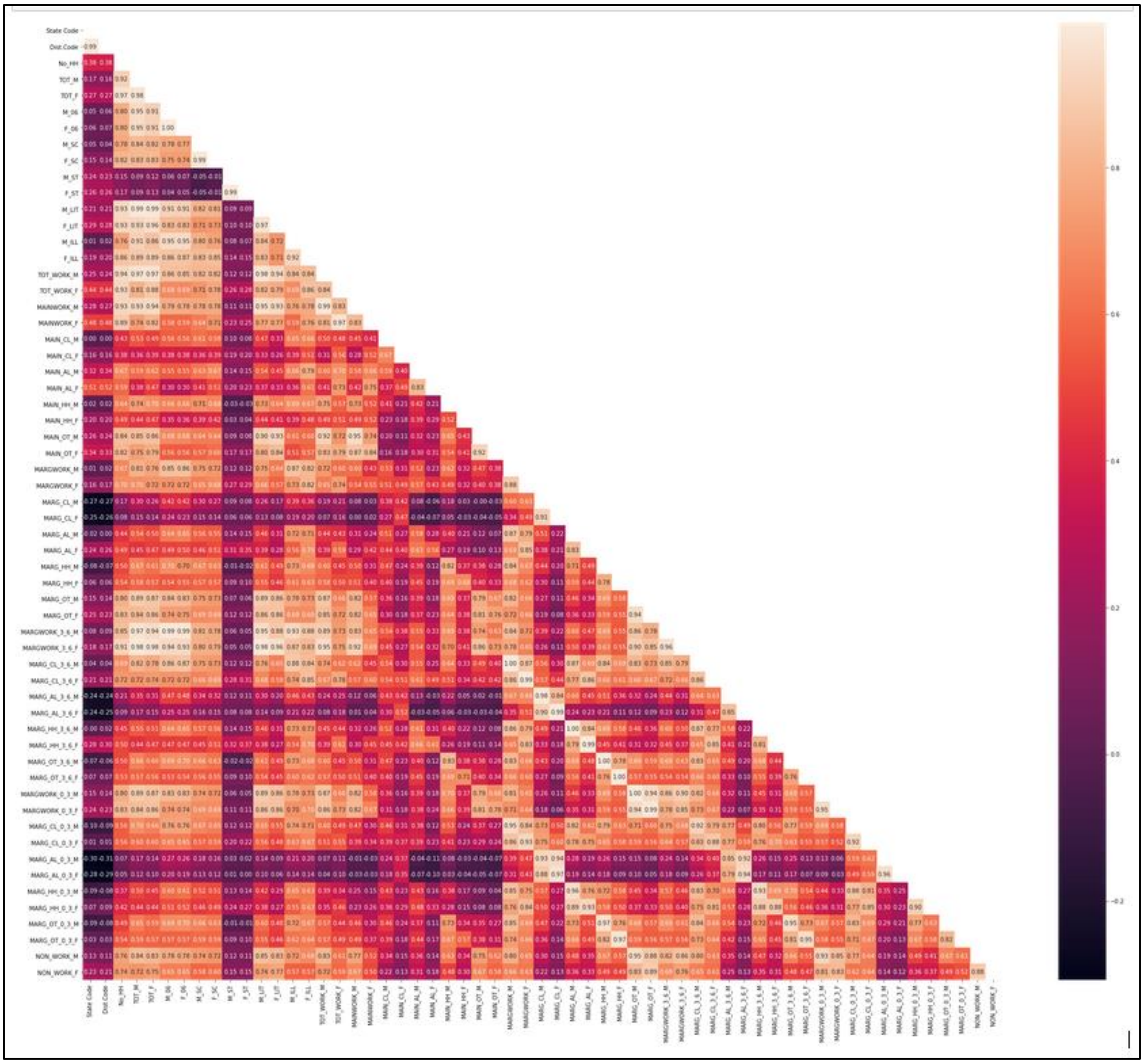
Outliers after scaling



Hence, we can clearly see that scaling does not impact the outliers.

5. Perform all the required steps for PCA (use Sklearn only) Create the covariance Matrix Get eigen values and eigen vector.

Solution:



Eigen Vectors:

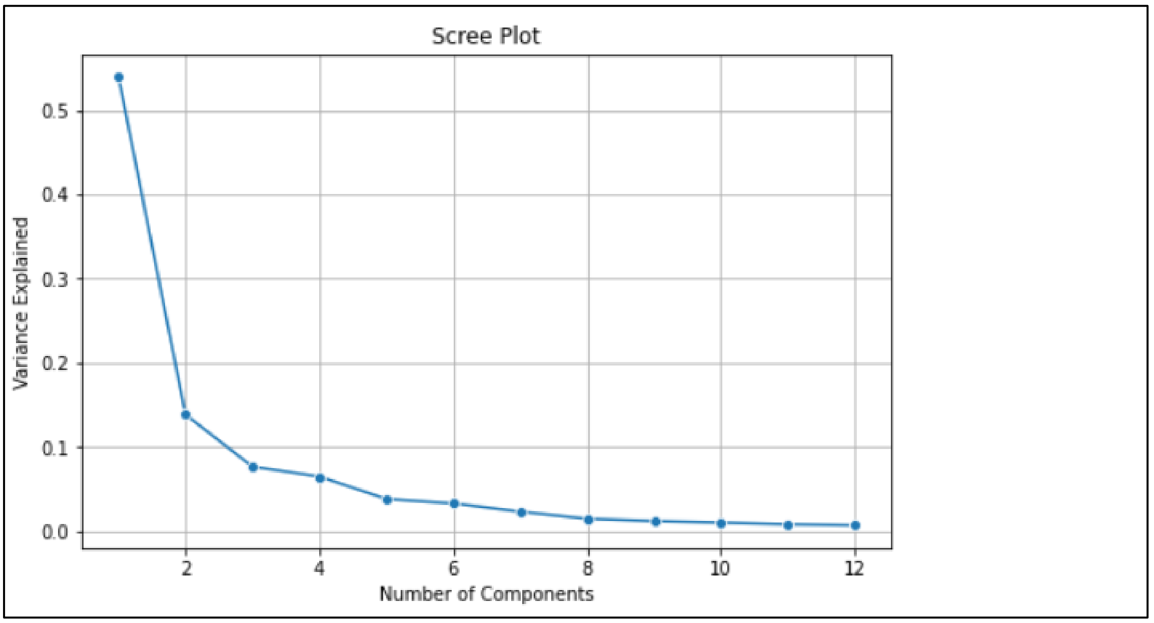
```
array([[ 3.00700521e-02,  3.00751392e-02,  1.56432451e-01,
         1.67038499e-01,  1.65701886e-01,  1.61870848e-01,
         1.62266320e-01,  1.51067631e-01,  1.51483487e-01,
         2.76635864e-02,  2.86559949e-02,  1.62028968e-01,
         1.47117900e-01,  1.61354631e-01,  1.65216191e-01,
         1.59988739e-01,  1.46484663e-01,  1.46446784e-01,
         1.24700922e-01,  1.02841551e-01,  7.46387972e-02,
         1.13762012e-01,  7.47868720e-02,  1.31280497e-01,
         8.36015471e-02,  1.23789890e-01,  1.11498595e-01,
         1.64144005e-01,  1.55258801e-01,  8.14703494e-02,
         4.84108523e-02,  1.28166982e-01,  1.14462067e-01,
         1.40274353e-01,  1.27424449e-01,  1.55154856e-01,
         1.47413552e-01,  1.64714317e-01,  1.61211005e-01,
         1.65089659e-01,  1.55618244e-01,  9.21330578e-02,
         5.07812312e-02,  1.28188765e-01,  1.10910853e-01,
         1.39029295e-01,  1.24330759e-01,  1.54196780e-01,
         1.46411774e-01,  1.49444956e-01,  1.39705021e-01,
         5.16456518e-02,  4.09693847e-02,  1.21254301e-01,
         1.15790305e-01,  1.39259946e-01,  1.31868671e-01,
```

Eigen Values:

```
array([31.86742634,  8.18907061,  4.54275124,  3.84336785,  2.27105793,
        1.95992589,  1.37548006,  0.88734267,  0.71989796,  0.61405955,
        0.49439969,  0.42414799])
```

6. Identify the optimum number of PCs (for this project, take at least 90% explained variance). Show Scree plot.

Solution:



The cumulative explained variance ratio to find a cut off for selecting the number of PCs

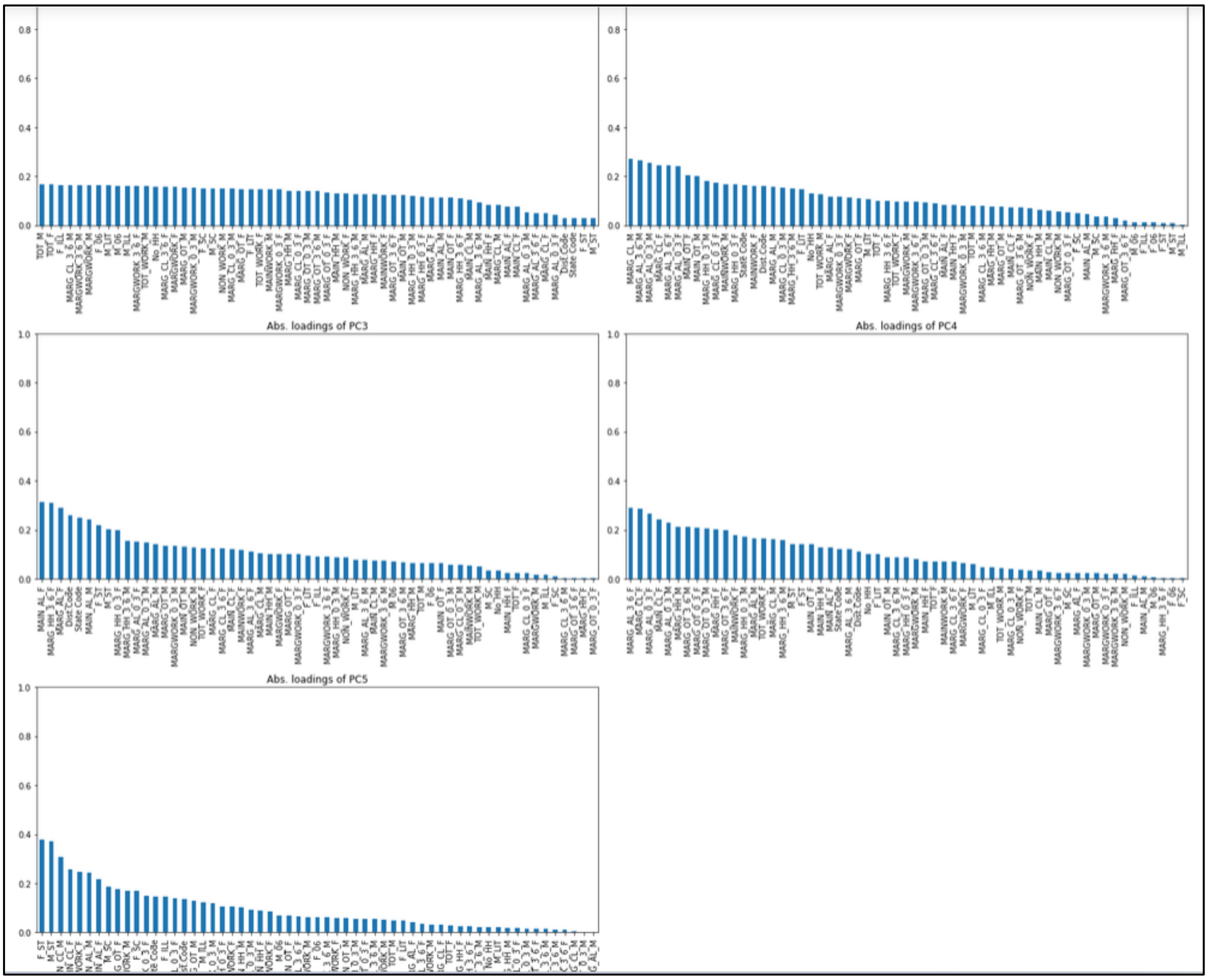
```
array([0.53928192, 0.67786286, 0.75473834, 0.81977838, 0.85821074,
       0.89137792, 0.91465472, 0.92967092, 0.94185352, 0.95224504,
       0.96061161, 0.96778932])
```

For this project, we need to consider at least 90% explained variance, so cut off for selecting the number of PCs is: ‘5’.

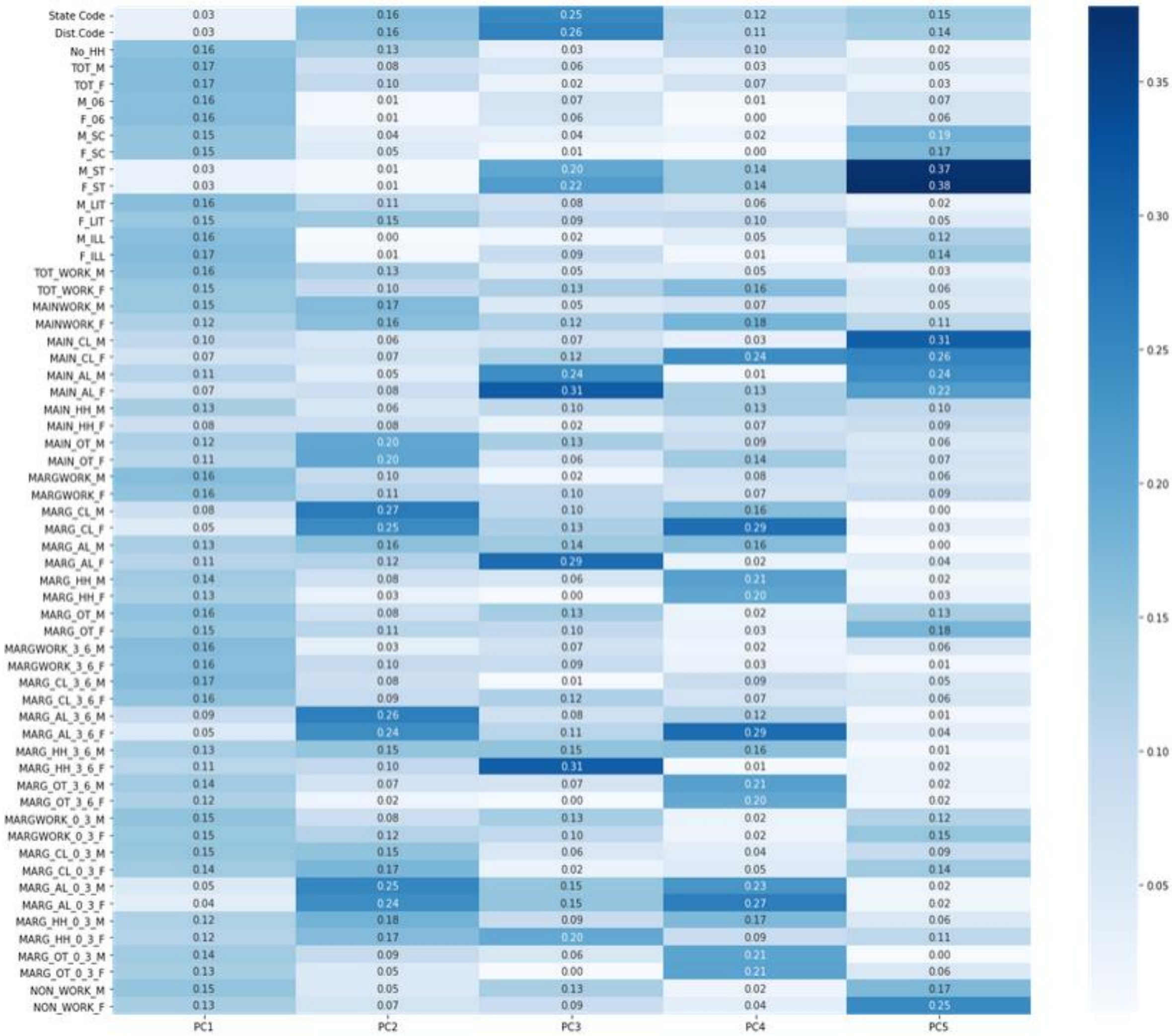
7. **Compare PCs with Actual Columns and identify which is explaining most variance. Write inferences about all the principal components in terms of actual variables.**

Solution:

How the original features matter to each PC



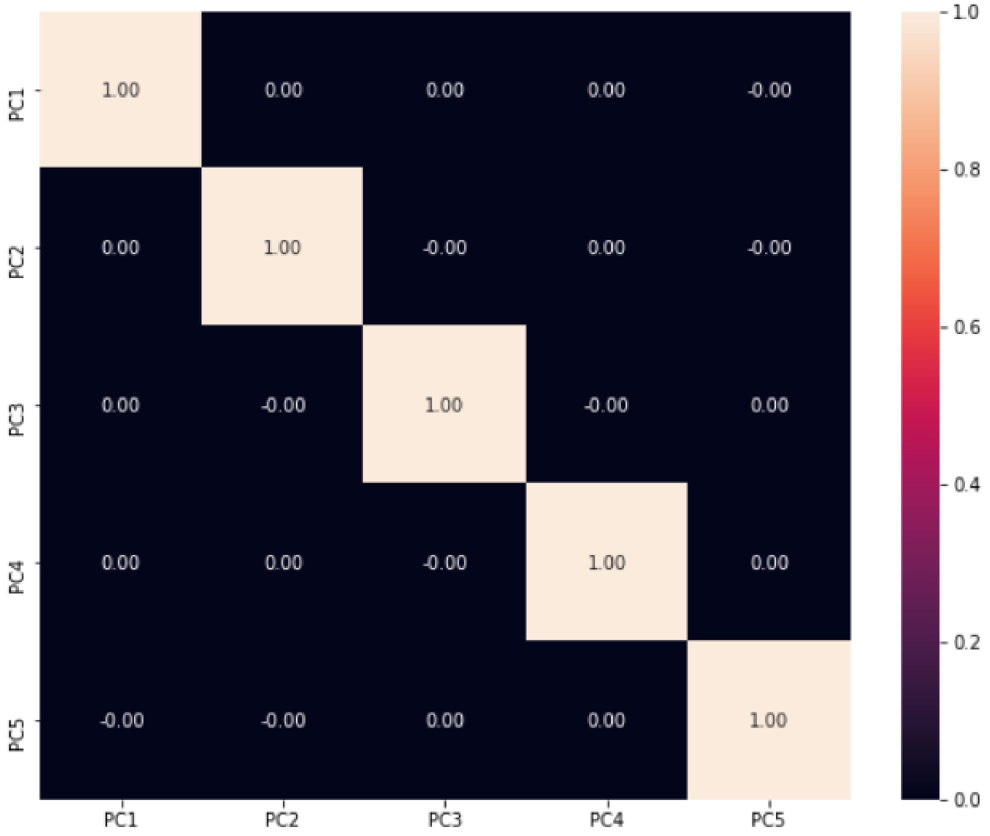
Compare how the original features influence various PCs



Extract the required number of PCs (5 in our case):

	PC1	PC2	PC3	PC4	PC5
0	-4.719381	0.717504	1.632266	-1.524984	0.090256
1	-4.873297	0.492001	1.752127	-1.938533	-0.262973
2	-6.062948	0.233751	1.333068	-0.710272	0.152168
3	-6.378387	0.042766	1.404373	-1.187672	0.013921
4	-4.581259	1.431602	1.722496	-0.231724	0.579575
5	-3.429451	3.370505	2.725939	1.662326	0.711022
6	-5.120804	0.230986	1.759260	-0.917209	-0.343377
7	-4.709479	0.602594	1.706348	-1.520298	-0.033930
8	-5.286297	0.506676	1.568660	-1.746378	0.037731
9	-4.323849	-0.705453	2.108597	-1.356074	0.027921

Check for presence of correlations among the PCs:



8. Write linear equation for first PC.

Solution:

PC1 = a1x1 + a2x2 + a3X3 +a4X4 + ……. + a57x57