imports

In [192…

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
```

Loading Dataset

In [193…

```
data= pd.read_csv('vitamins_data.csv')
data.head(8)
```

Out[193…

	State	Population(0-6)years	VitA_deficit%	VitD_deficit%
0	India	163819614	17.6	13.8
1	Delhi	2016849	17.8	32.5
2	Haryana	3335537	26.1	27.6
3	Himachal Pradesh	793137	5.9	4.6
4	Jammu & Kashmir	1485803	8.7	22.9
5	Punjab	3171829	17.2	52.1
6	Rajasthan	10651002	NaN	25.2
7	Uttarakhand	1360032	14.3	46.4

Removing Outlier

In [194…

```
#As row 1 represents the information of india,while other rows represents
states, so  we have to remove it as it is an outlier.
```

In [195…

```
data = data.drop(labels=0, axis=0)
```

In [196…

```
data.head()
```

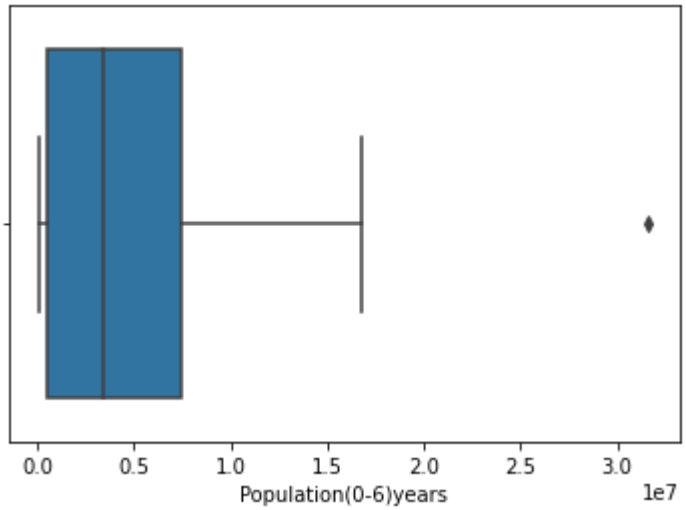
Out[196…

	State	Population(0-6)years	VitA_deficit%	VitD_deficit%
1	Delhi	2016849	17.8	32.5
2	Haryana	3335537	26.1	27.6

	State	Population(0-6)years	VitA_deficit%	VitD_deficit%
3	Himachal Pradesh	793137	5.9	4.6
4	Jammu & Kashmir	1485803	8.7	22.9
5	Punjab	3171829	17.2	52.1

In [197…
```
sns.boxplot(x=data['Population(0-6)years'])
```

Out[197… <AxesSubplot:xlabel='Population(0-6)years'>



In [198…
```
#one more outlier is present but we cannot drop that as it represents some state and if we remove that we will not get information about that state.
```

In [199…
```
data.shape
```

Out[199… (30, 4)

Statistical Summary

In [200…
```
data.describe()
```

Out[200…

	Population(0-6)years	VitA_deficit%	VitD_deficit%
count	3.000000e+01	28.000000	30.000000
mean	5.516359e+06	17.425000	15.860000
std	6.736190e+06	9.912343	13.488248
min	7.819500e+04	2.400000	1.100000
25%	5.492685e+05	9.550000	5.800000
50%	3.445226e+06	17.100000	12.300000

	Population(0-6)years	VitA_deficit%	VitD_deficit%
75%	7.458093e+06	21.925000	22.850000
max	3.162463e+07	43.200000	52.100000

In [201…
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 30 entries, 1 to 30
Data columns (total 4 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   State                 30 non-null     object 
 1   Population(0-6)years  30 non-null     int64  
 2   VitA_deficit%         28 non-null     float64
 3   VitD_deficit%         30 non-null     float64
dtypes: float64(2), int64(1), object(1)
memory usage: 1.2+ KB
```

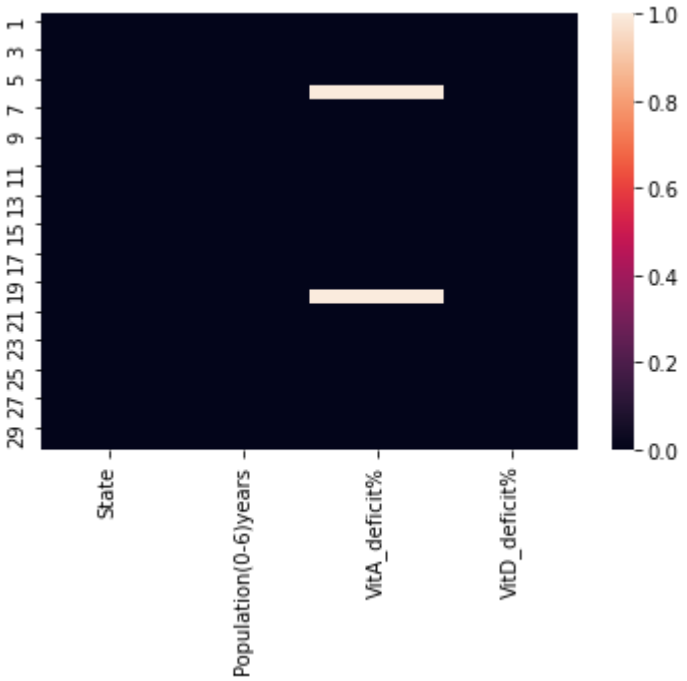
Checking for null Values

In [202…
```
data.isnull().sum()
```

Out[202…
```
State                   0
Population(0-6)years    0
VitA_deficit%           2
VitD_deficit%           0
dtype: int64
```

In [203…
```
sns.heatmap(data.isnull())
```

Out[203… <AxesSubplot:>



Removing Null with Mean Value

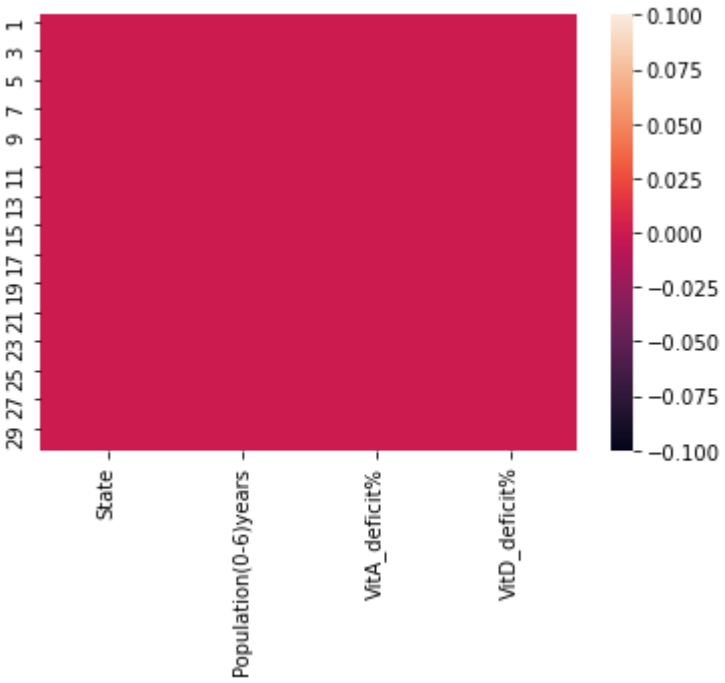
In [204…
```
data['VitA_deficit%'].fillna(data['VitA_deficit%'].mean(), inplace = True)
```

In [205…
```
data.isnull().sum()
```

Out[205…
```
State                   0
Population(0-6)years    0
VitA_deficit%           0
VitD_deficit%           0
dtype: int64
```

In [206…
```
sns.heatmap(data.isnull())
```

Out[206… <AxesSubplot:>

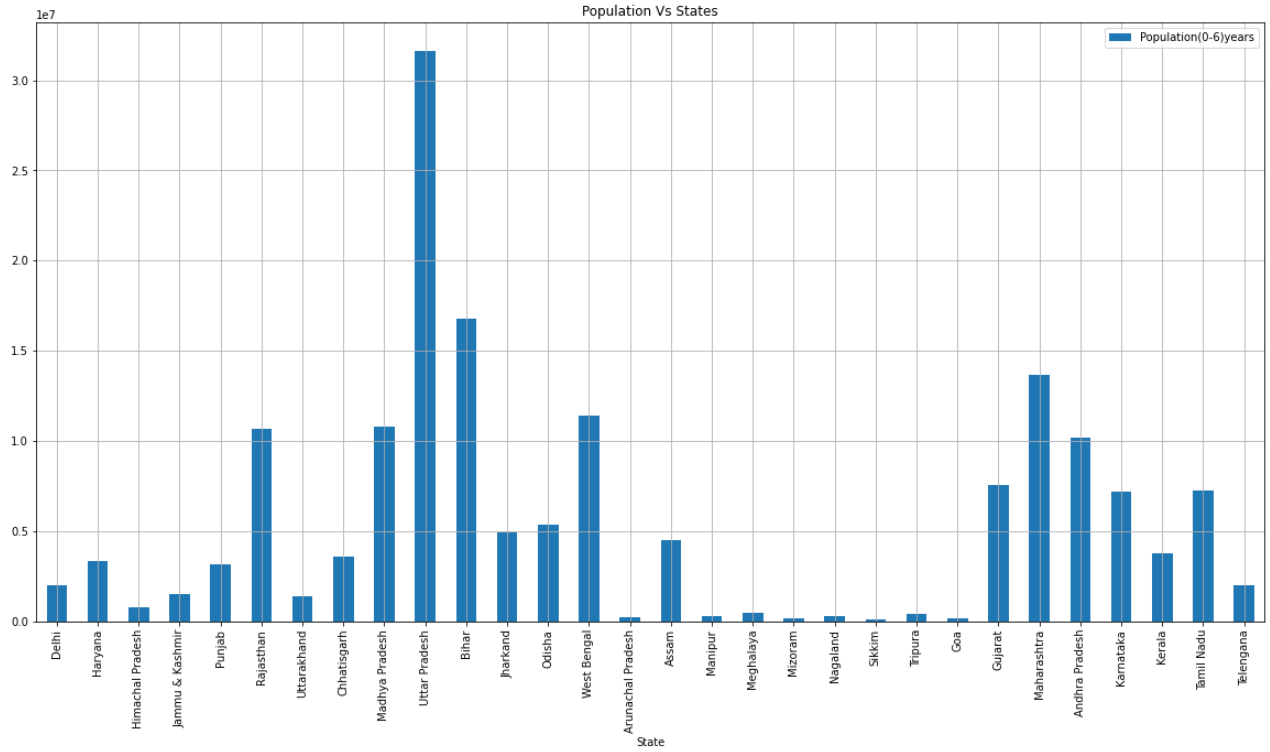


Exploratory Data Analysis

In [207…
```
#first lets relate the population of 0-6 years old in different states of
india
```

In [208…
```
fig, ax= plt.subplots(figsize=(20,10))
data.plot(x= 'State', y ="Population(0-6)years", kind = 'bar',
        title ='Population Vs States', grid=True,ax=ax)
```

Out[208… <AxesSubplot:title={'center':'Population Vs States'}, xlabel='State'>



So there are many states which can be selected for our start up to launch their services in solely based on population count.

Most likely more business will be genereted from states like :

Rajasthan Uttarpradesh Bihar West Bengal Madhya Pradesh Maharastra Andhar Pradesh

Note :- These are states with population greater than 10 Millions and this does not visualize whole scenario it is just a speculation based on Total population count of the above given states.

In [209…

```
#Now lets see the distribution of population(0-6 years), vitamin A,B deficiency
```

In [210…

```
fig,ax = plt.subplots(1,3,figsize=(16,4))
fig.subplots_adjust(wspace = 0.5)
sns.distplot(data['Population(0-6)years'],
ax=ax[0]).set_title('Distribution of Population')
sns.distplot(data['VitA_deficit%'], ax=ax[1]).set_title('Distribution of the column VitA_deficit%')
sns.distplot(data['VitD_deficit%'], ax=ax[2]).set_title('Distribution of the column VitD_deficit%')
```

```
C:\Users\brainiac Abhinav\anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
```

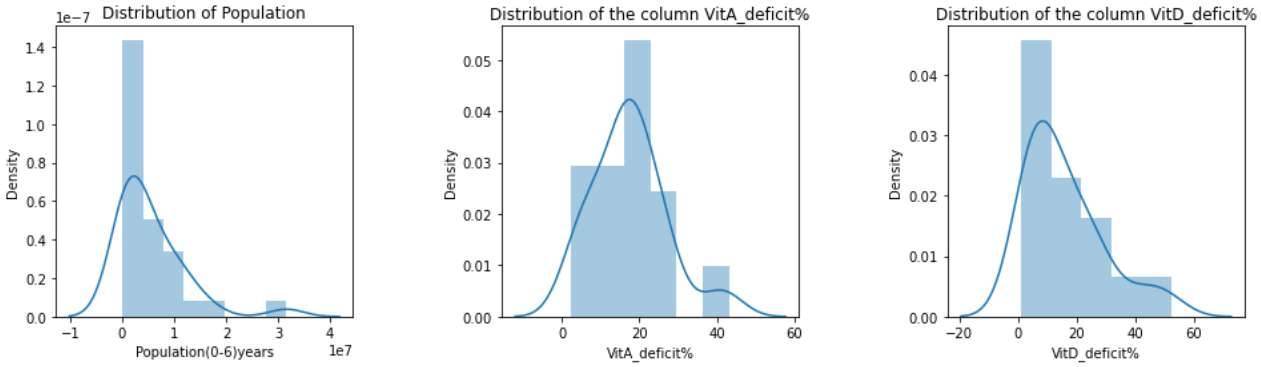
```
C:\Users\brainiac Abhinav\anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).

C:\Users\brainiac Abhinav\anaconda3\lib\site-packages\seaborn\distributions.py:2557: Fut
ureWarning:

`distplot` is a deprecated function and will be removed in a future version. Please adap
t your code to use either `displot` (a figure-level function with similar flexibility) o
r `histplot` (an axes-level function for histograms).
```

Out[210… `Text(0.5, 1.0, 'Distribution of the column VitD_deficit%')`



Adding columns in data repersenting total number of poulation having viatmin A and D deficiency respectively for each state

In [211…
```
data['VitA_deficit_total'] = data['VitA_deficit%'] * data['Population(0-
6)years'] /100
data['VitD_deficit_total'] = data['VitD_deficit%'] * data['Population(0-
6)years'] /100
```

In [212…
```
data.head()
```

Out[212…

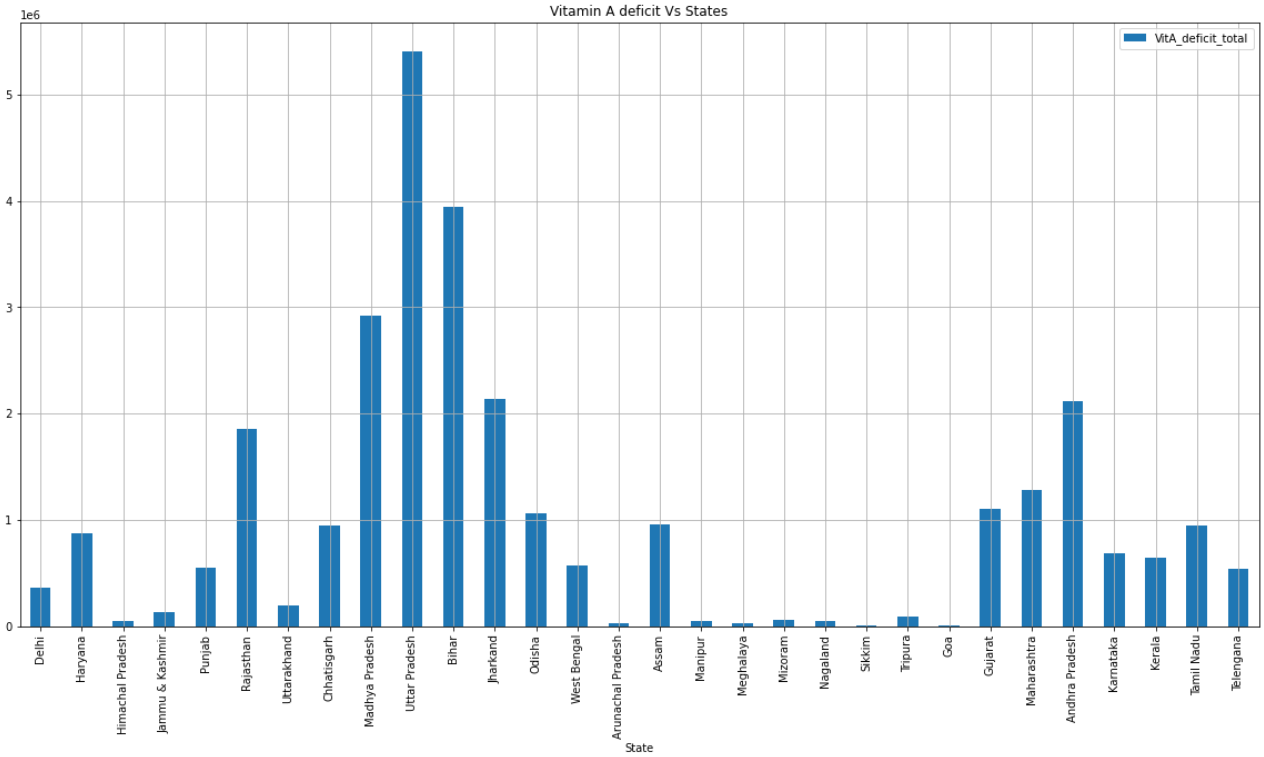
	State	Population(0-6)years	VitA_deficit%	VitD_deficit%	VitA_deficit_total	VitD_deficit_total
1	Delhi	2016849	17.8	32.5	358999.122	655475.925
2	Haryana	3335537	26.1	27.6	870575.157	920608.212
3	Himachal Pradesh	793137	5.9	4.6	46795.083	36484.302
4	Jammu & Kashmir	1485803	8.7	22.9	129264.861	340248.887
5	Punjab	3171829	17.2	52.1	545554.588	1652522.909

In [213…
```
#Now lets see the vitamin A deficit numbers in each states
```

In [214…

```
fig, ax= plt.subplots(figsize=(20,10))
data.plot(x= 'State', y ="VitA_deficit_total", kind = 'bar',
        title ='Vitamin A deficit Vs States', grid=True,ax=ax)
```

Out[214… <AxesSubplot:title={'center':'Vitamin A deficit Vs States'}, xlabel='State'>



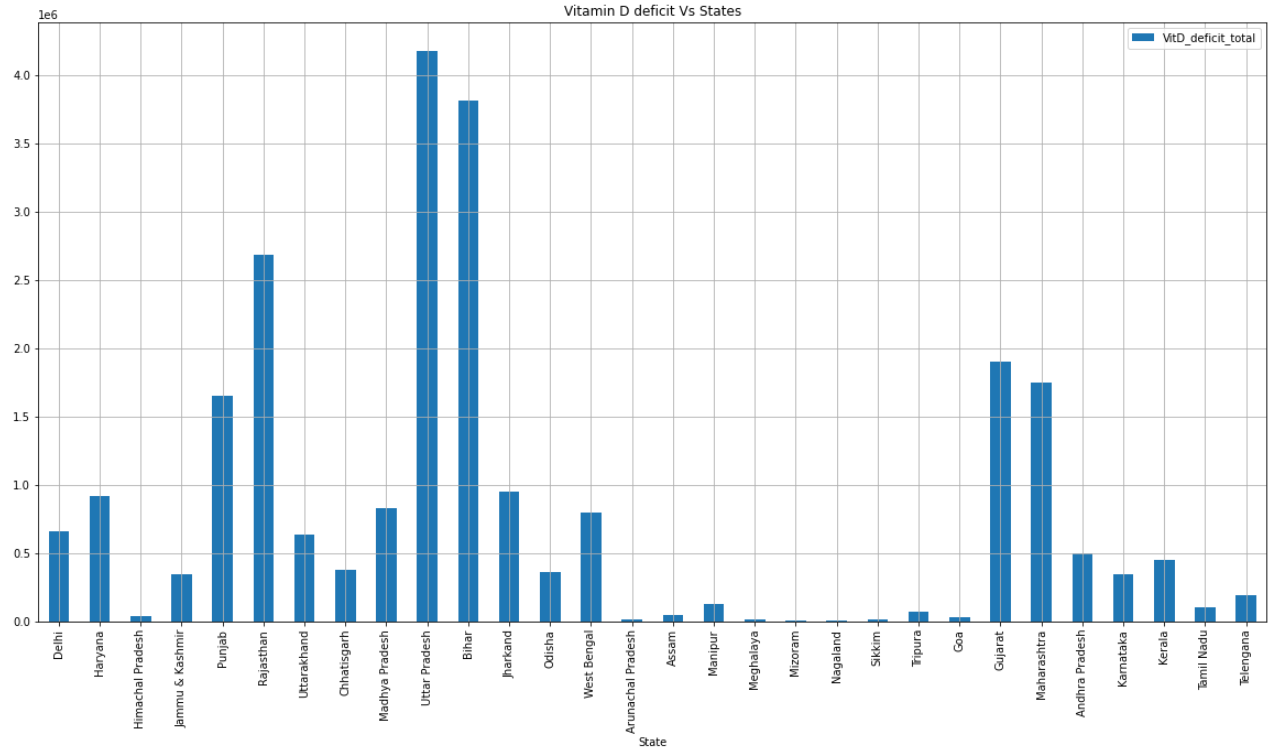
In [215…

```
#Now lets see the vitamin D deficit numbers in each states
```

In [216…

```
fig, ax= plt.subplots(figsize=(20,10))
data.plot(x= 'State', y = "VitD_deficit_total", kind = 'bar',
        title ='Vitamin D deficit Vs States', grid=True,ax=ax)
```

Out[216… <AxesSubplot:title={'center':'Vitamin D deficit Vs States'}, xlabel='State'>



In [217…

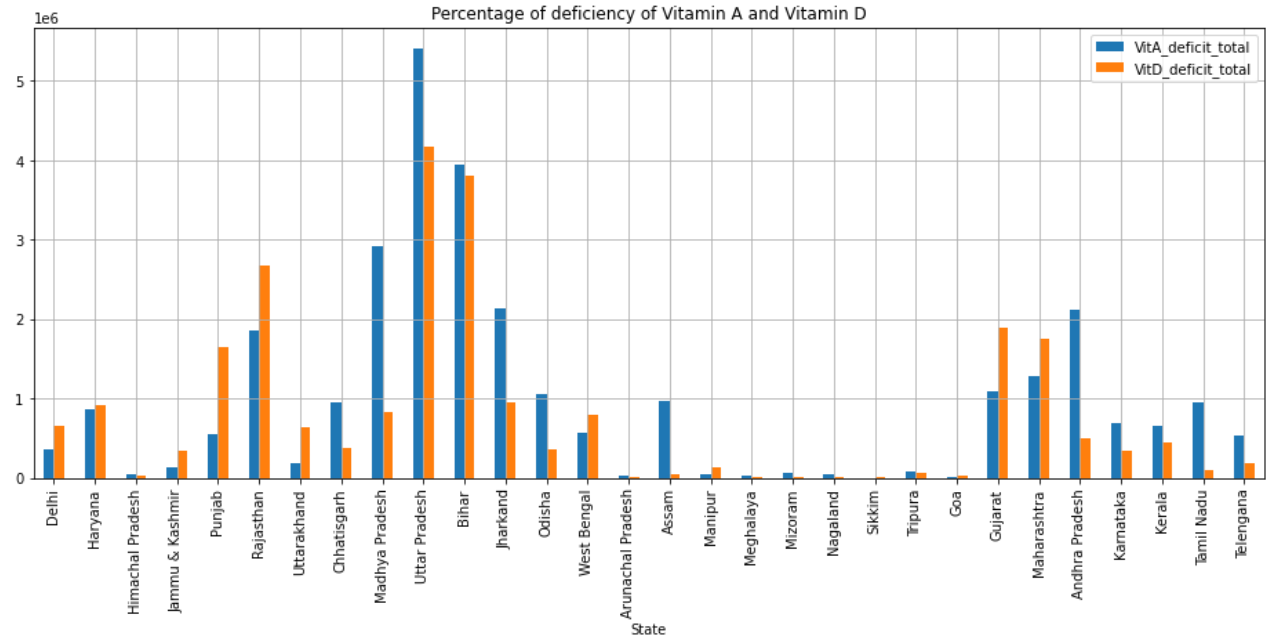
```
#Comparison of Vitamin A and D deficiency in differecnt states
```

In [218…

```
fig, ax= plt.subplots(figsize=(16,6))
data.plot(x= 'State', y =['VitA_deficit_total', 'VitD_deficit_total'], kind
= 'bar',
        title ='Percentage of deficiency of Vitamin A and Vitamin D',
grid=True,ax=ax)
```

Out[218…

```
<AxesSubplot:title={'center':'Percentage of deficiency of Vitamin A and Vitamin D'}, xla
bel='State'>
```

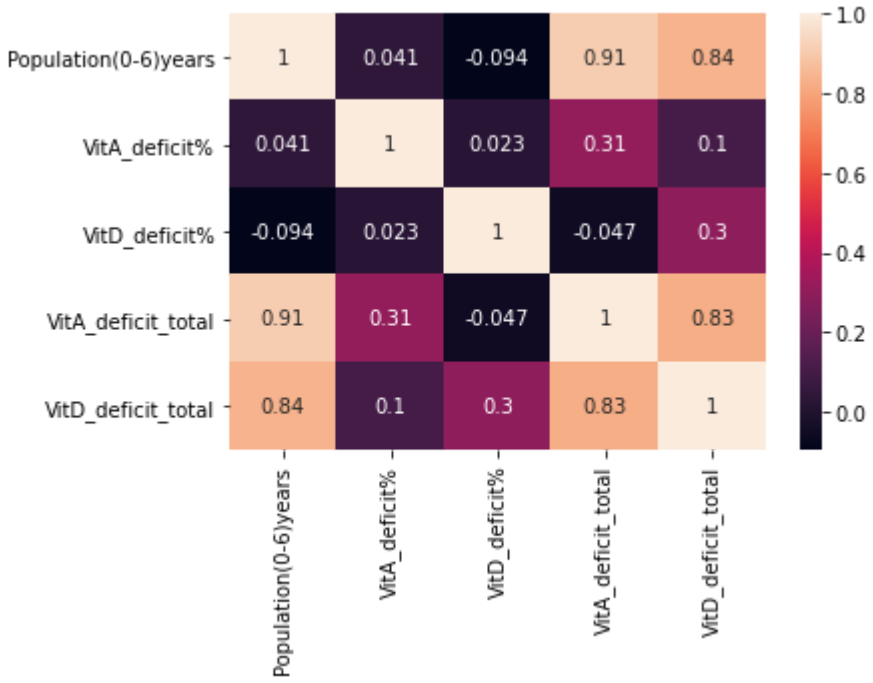


With this We can conclude in most states vitamin A Defficiency is more common, So our medical service should focus more vitamin A rich medicines.

In [219…
```
#Correlation between features using heatmap
```

In [220…
```
sns.heatmap(data.corr(),annot=True)
```

Out[220… <AxesSubplot:>



1.According to this total childs with Vitamin A and Vitamin D defficiency is highly correlated with its population. Hence more bussiness will be generated if we setup our services in more populated area. 2.Vitamin A deficiency is highly correlated with vitamin D i.e., if a person is having vitamin A deficency then there is higher probability that they have vitamin D deficiency and vise versa.

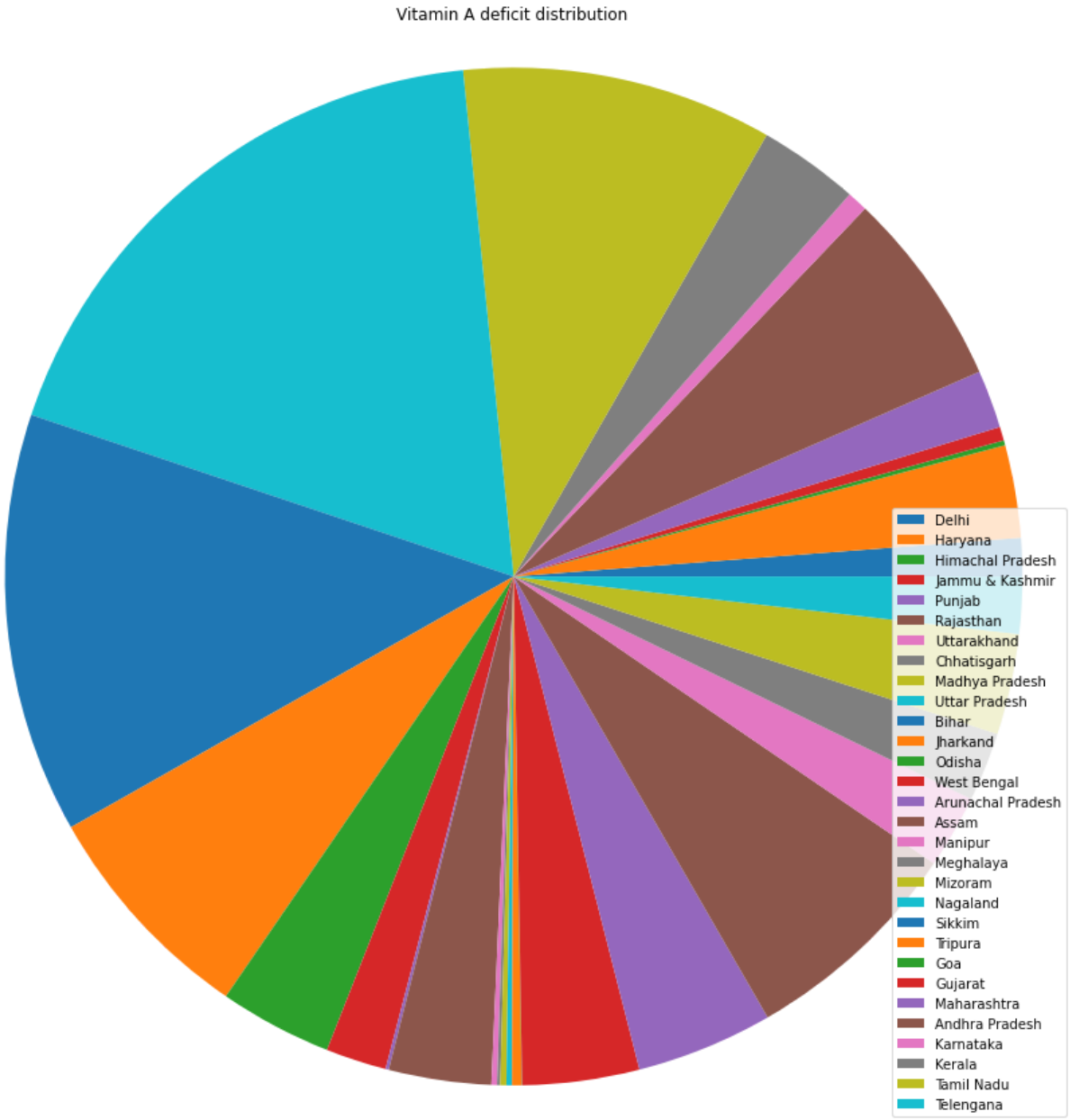
In [221…
```
#Distribution of Vitamin A deficiency over different states
```

In [222…
```
pie, ax = plt.subplots(figsize=[15,15])
patches, texts = plt.pie(data['VitA_deficit_total'])
plt.title("Vitamin A deficit distribution")
plt.legend(patches, labels=data['State'], loc="lower right")
plt.axis('equal')
```

```
<ipython-input-222-a2848021197f>:4: UserWarning:

You have mixed positional and keyword arguments, some input may be discarded.
```

Out[222…
```
(-1.109281715194373,
 1.1004419864378272,
 -1.1002725335111567,
 1.1056119171905523)
```



In [223…

```
#Distribution of Vitamin D deficiency over different states
```

In [224…

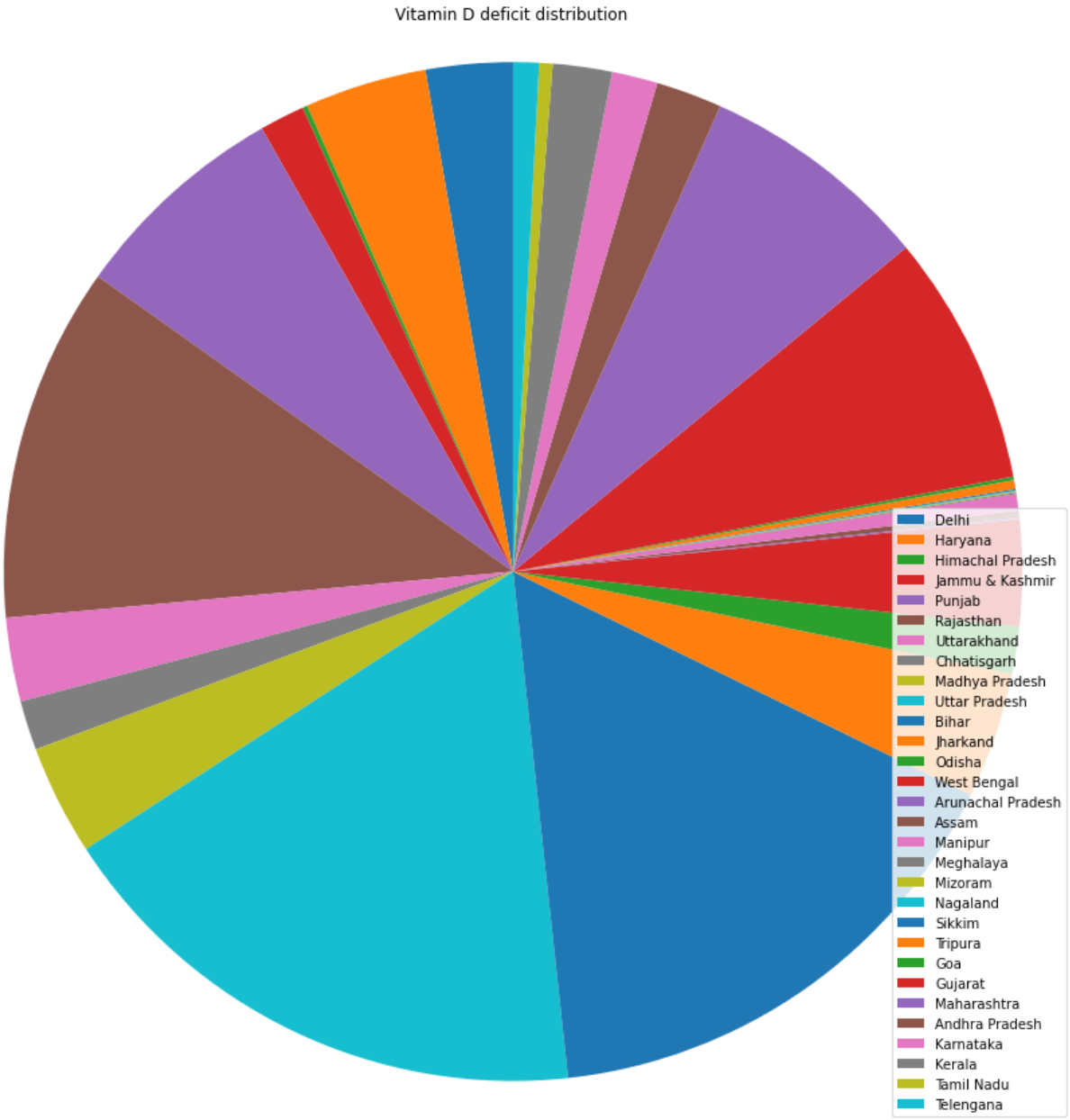
```
pie, ax = plt.subplots(figsize=[15,15])
patches, texts = plt.pie(data['VitD_deficit_total'], startangle=90)
plt.title("Vitamin D deficit distribution")
plt.legend(patches, labels=data['State'], loc="lower right")
plt.axis('equal')
```

```
<ipython-input-224-2bf67607ef90>:4: UserWarning:

You have mixed positional and keyword arguments, some input may be discarded.
```

Out[224…

```
(-1.1069749665484279,
 1.1004165173396536,
 -1.1148496795770613,
 1.1007071305043223)
```



According to these to pie charts:- Gujarat, Rajasthan, Bihar and Uttar Pradesh are prominant areas to setup our business.

In [225…
```
#Clustering similar states together to operate our bussiness similarly in similar states
```

In [226…
```
train =
data[['VitA_deficit%','VitD_deficit%','VitA_deficit_total','VitD_deficit_tota
```

In [227…
```
sc= StandardScaler().fit(train)
train_std = sc.transform(train)
```

In [228…
```
from sklearn.cluster import KMeans
```

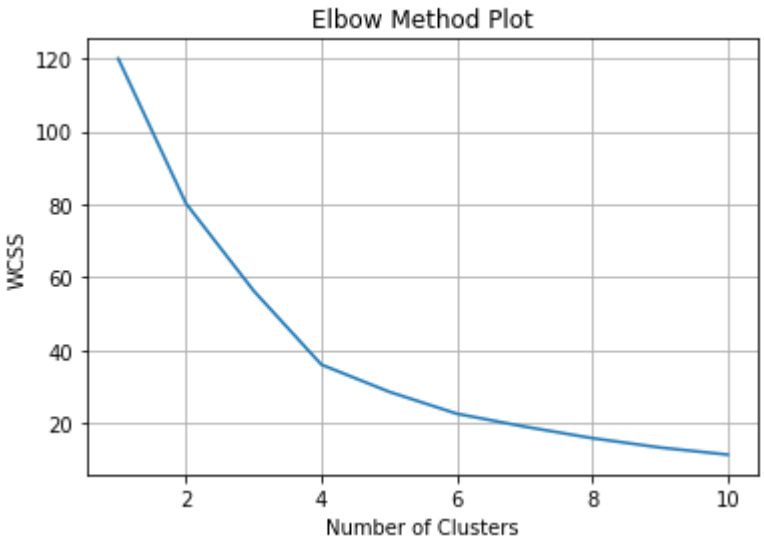
In [229…

```
wcss =[]
#wcss = Within Cluster Sum of Squares
for i in range(1,11):
    kmns= KMeans(n_clusters = i, init = 'k-means++', random_state =23)
    kmns.fit(train_std)
    wcss.append(kmns.inertia_)

#Plotting to find the optimum number of clusters
plt.plot(range(1,11), wcss)
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.title('Elbow Method Plot')
plt.grid(True)
plt.show()
```

```
C:\Users\brainiac Abhinav\anaconda3\lib\site-packages\sklearn\cluster\_kmeans.py:881: UserWarning:

KMeans is known to have a memory leak on Windows with MKL, when there are less chunks than available threads. You can avoid it by setting the environment variable OMP_NUM_THREADS=1.
```



In [230…

```
kmeans = KMeans(n_clusters= 5)
label = kmeans.fit_predict(train_std)
print(label)
```

```
[2 2 3 3 2 4 2 0 0 1 1 0 0 3 3 0 2 3 0 3 3 0 3 4 4 0 3 3 3 0]
```

In [231…

```
data['Cluster']=label
data.head()
```

Out[231...

	State	Population(0-6)years	VitA_deficit%	VitD_deficit%	VitA_deficit_total	VitD_deficit_total	Cluster
1	Delhi	2016849	17.8	32.5	358999.122	655475.925	2
2	Haryana	3335537	26.1	27.6	870575.157	920608.212	2
3	Himachal Pradesh	793137	5.9	4.6	46795.083	36484.302	3
4	Jammu & Kashmir	1485803	8.7	22.9	129264.861	340248.887	3
5	Punjab	3171829	17.2	52.1	545554.588	1652522.909	2

In [232...

```
df1 = data[data['Cluster']==0]
df2 = data[data['Cluster']==1]
df3 = data[data['Cluster']==2]
df4 = data[data['Cluster']==3]
df5 = data[data['Cluster']==4]
```

In [233...

```
cluster1 = df1.State
cluster2 = df2.State
cluster3 = df3.State
cluster4 = df4.State
cluster5 = df5.State

print('States in Cluster1 are ', cluster1.to_numpy())

print('States in Cluster2 are ', cluster2.to_numpy())

print('States in Cluster3 are ', cluster3.to_numpy())

print('States in Cluster4 are ', cluster4.to_numpy())
print('States in Cluster5 are ', cluster5.to_numpy())
```

```
States in Cluster1 are  ['Chhatisgarh' 'Madhya Pradesh' 'Jharkand' 'Odisha' 'Assam' 'Miz
oram'
 'Tripura' 'Andhra Pradesh' 'Telengana']
States in Cluster2 are  ['Uttar Pradesh' 'Bihar']
States in Cluster3 are  ['Delhi' 'Haryana' 'Punjab' 'Uttarakhand' 'Manipur']
States in Cluster4 are  ['Himachal Pradesh' 'Jammu & Kashmir' 'West Bengal' 'Arunachal P
radesh'
 'Meghalaya' 'Nagaland' 'Sikkim' 'Goa' 'Karnataka' 'Kerala' 'Tamil Nadu']
States in Cluster5 are  ['Rajasthan' 'Gujarat' 'Maharashtra']
```

Hence our business strategy should be: -> similar for states ['Uttar Pradesh' 'Bihar'] -> similar for states ['Himachal Pradesh' 'Jammu & Kashmir' 'West Bengal' 'Arunachal Pradesh' 'Meghalaya'

'Nagaland' 'Sikkim' 'Goa' 'Karnataka' 'Kerala' 'Tamil Nadu'] -> similar for states ['Delhi' 'Haryana' 'Punjab' 'Uttarakhand' 'Manipur'] -> similar for states ['Rajasthan' 'Gujarat' 'Maharashtra']