Understanding theTrends in Indian Education

Imports

In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import random
```

In [2]:

```
random.seed(a=42)
```

Import Data

In [3]:

```
xls_06_07 = pd.ExcelFile('Data/processed/SRC_Rawdata_2006-07_mod.xls')
basic_06_07 = pd.read_excel(xls_06_07, 'Basic Data')
facilities_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
condition_06_07 = pd.read_excel(xls_06_07, 'School Condition')
enrolment_06_07 = pd.read_excel(xls_06_07, 'Enrolment')
teacher_06_07 = pd.read_excel(xls_06_07, 'Teacher')
```

In [4]:

```
xls_07_08 = pd.ExcelFile('Data/processed/SRC_Rawdata_2007-08_mod.xls')
basic_07_08 = pd.read_excel(xls_07_08, 'Basic Data')
facilities_07_08 = pd.read_excel(xls_07_08, 'School Facilities')
condition_07_08 = pd.read_excel(xls_07_08, 'School Condition')
enrolment_07_08 = pd.read_excel(xls_07_08, 'Enrolment')
teacher_07_08 = pd.read_excel(xls_07_08, 'Teacher')
```

In [5]:

```
xls_08_09 = pd.ExcelFile('Data/processed/SRC_Rawdata_2008-09_mod.xls')
basic_08_09 = pd.read_excel(xls_08_09, 'Basic Data')
facilities_08_09 = pd.read_excel(xls_08_09, 'School Facilities')
condition_08_09 = pd.read_excel(xls_08_09, 'School Condition')
enrolment_08_09 = pd.read_excel(xls_08_09, 'Enrolment')
teacher_08_09 = pd.read_excel(xls_08_09, 'Teacher')
```

In [6]:

```
xls_09_10 = pd.ExcelFile('Data/processed/SRC_Rawdata_2009-10_mod.xls')
basic_09_10 = pd.read_excel(xls_09_10, 'Basic Data')
facilities_09_10 = pd.read_excel(xls_09_10, 'School Facilities')
condition_09_10 = pd.read_excel(xls_09_10, 'School Condition')
enrolment_09_10 = pd.read_excel(xls_09_10, 'Enrolment')
teacher_09_10 = pd.read_excel(xls_09_10, 'Teacher')
```

In [7]:

```
xls_10_11 = pd.ExcelFile('Data/processed/SRC_Rawdata_2010-11_mod.xls')
basic_10_11 = pd.read_excel(xls_10_11, 'Basic Data')
facilities_10_11 = pd.read_excel(xls_10_11, 'School Facilities')
condition_10_11 = pd.read_excel(xls_10_11, 'School Condition')
enrolment_10_11 = pd.read_excel(xls_10_11, 'Enrolment')
teacher_10_11 = pd.read_excel(xls_10_11, 'Teacher')
```

In [8]:

```
xls_11_12 = pd.ExcelFile('Data/processed/SRC_Rawdata_2011-12_mod.xls')
basic_11_12 = pd.read_excel(xls_11_12, 'Basic Data')
facilities_11_12 = pd.read_excel(xls_11_12, 'School Facilities')
condition_11_12 = pd.read_excel(xls_11_12, 'School Condition')
enrolment_11_12 = pd.read_excel(xls_11_12, 'Enrolment')
teacher_11_12 = pd.read_excel(xls_11_12, 'Teacher')
```

In [9]:

```
xls_12_13 = pd.ExcelFile('Data/processed/SRC_Rawdata_2012-13_mod.xls')
basic_12_13 = pd.read_excel(xls_12_13, 'Basic Data')
facilities_12_13 = pd.read_excel(xls_12_13, 'School Facilities')
condition_12_13 = pd.read_excel(xls_12_13, 'School Condition')
enrolment_12_13 = pd.read_excel(xls_12_13, 'Enrolment')
teacher_12_13 = pd.read_excel(xls_12_13, 'Teacher')
```

In [10]:

```python
xls_13_14 = pd.ExcelFile('Data/processed/SRC_Rawdata_2013-14_mod.xls')
basic_13_14 = pd.read_excel(xls_13_14, 'Basic Data')
facilities_13_14 = pd.read_excel(xls_13_14, 'School Facilities')
condition_13_14 = pd.read_excel(xls_13_14, 'School Condition')
enrolment_13_14 = pd.read_excel(xls_13_14, 'Enrolment')
teacher_13_14 = pd.read_excel(xls_13_14, 'Teacher')
```

Literacy Rate v/s Teacher:Student ratio

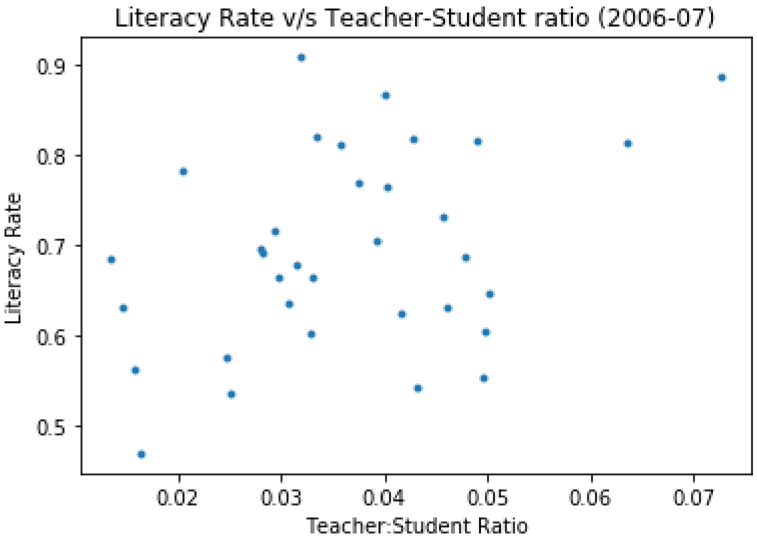
Literacy rate against Teacher-Student ratio was plotted for different years. As we can see, the Teacher-Student ratio is increasing through the years as we look from 2006 to 2013. This implies that there has been an increase in the number of teachers. Also, the literacy rates have also increased over these years implying that the increase in the number of teachers has a positive influence on the literacy rate.

In [11]:

```python
lit_06_07 = basic_06_07['literacy_rate']/100
tch_stud_06_07 = (teacher_06_07['govt_tch_3'] + teacher_06_07['pvt_tch_3']) / (enrolment_06_07['govt_enr_3'
```

In [12]:

```
plt.plot(tch_stud_06_07, lit_06_07, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2006-07)')
plt.show()
```

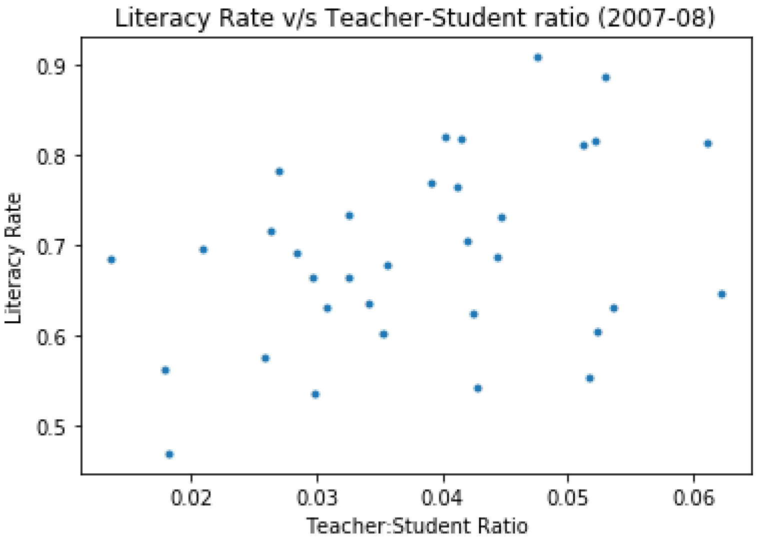


In [13]:

```
lit_07_08 = basic_07_08['literacy_rate']/100
tch_stud_07_08 = (teacher_07_08['govt_tch_3'] + teacher_07_08['pvt_tch_3']) / (enrolment_07_08['govt_enr_3'
```

In [14]:

```
plt.plot(tch_stud_07_08, lit_07_08, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2007-08)')
plt.show()
```

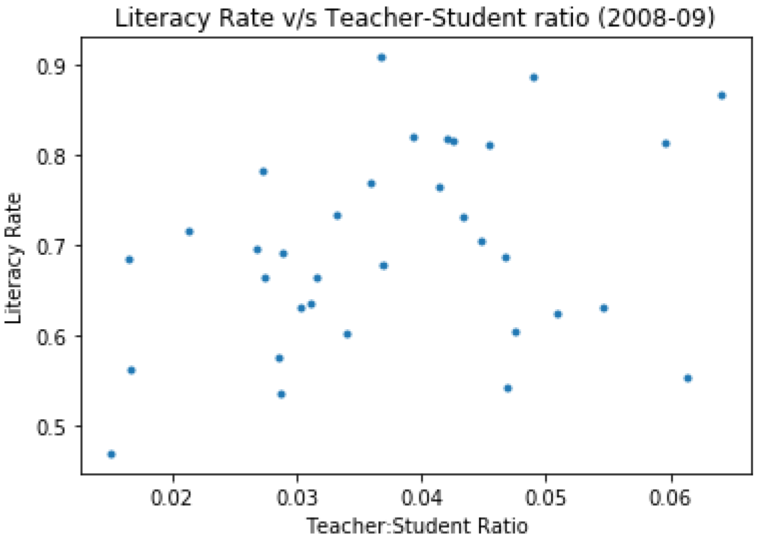


In [15]:

```
lit_08_09 = basic_08_09['literacy_rate']/100
tch_stud_08_09 = (teacher_08_09['govt_tch_3'] + teacher_08_09['pvt_tch_3']) / (enrolment_08_09['govt_enr_3'
```

In [16]:

```
plt.plot(tch_stud_08_09, lit_08_09, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2008-09)')
plt.show()
```

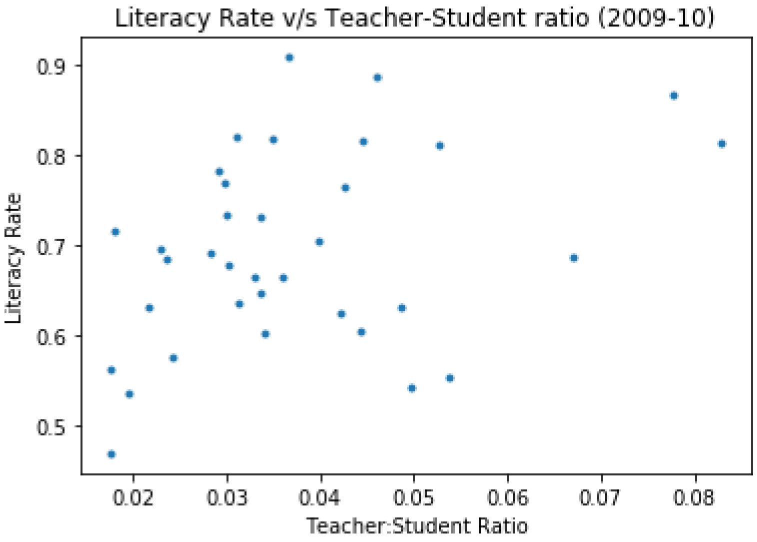


In [17]:

```
lit_09_10 = basic_09_10['literacy_rate']/100
tch_stud_09_10 = (teacher_09_10['govt_tch_3'] + teacher_09_10['pvt_tch_3']) / (enrolment_09_10['govt_enr_3'
```

In [18]:

```python
plt.plot(tch_stud_09_10, lit_09_10, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2009-10)')
plt.show()
```

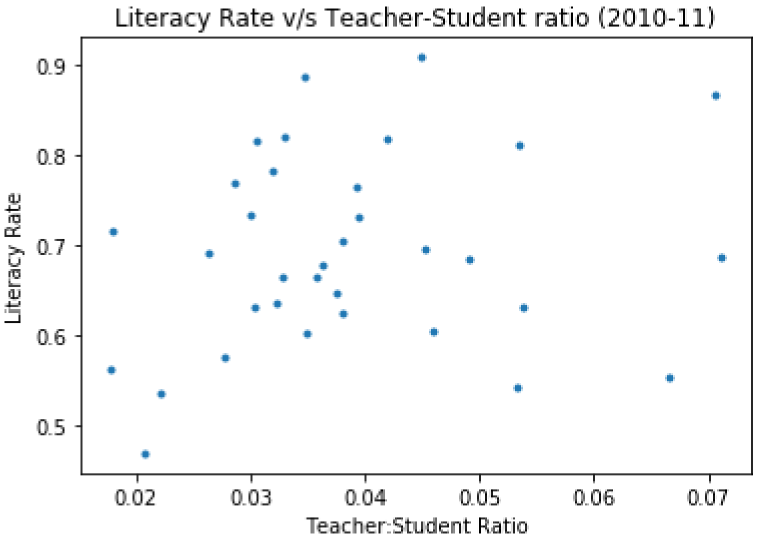


In [19]:

```python
lit_10_11 = basic_10_11['Literacy Rate']/100
tch_stud_10_11 = (teacher_10_11['Govt Tch 3'] + teacher_10_11['Pvt Tch 3']) / (enrolment_10_11['govt_enr_3'
```

In [20]:

```
plt.plot(tch_stud_10_11, lit_10_11, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2010-11)')
plt.show()
```



In [21]:

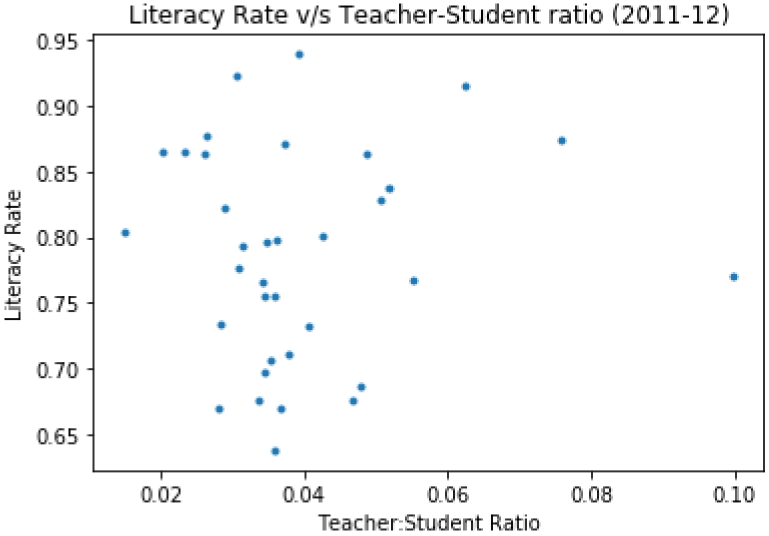
```
lit_11_12 = basic_11_12['literacy_rate']/100
tch_stud_11_12 = (teacher_11_12['govt_tch_3'] + teacher_11_12['pvt_tch_3']) / (enrolment_11_12['govt_enr_3'
```

In [22]:

```
tch_stud_11_12 = tch_stud_11_12[1:]
```

In [23]:

```
plt.plot(tch_stud_11_12, lit_11_12, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2011-12)')
plt.show()
```

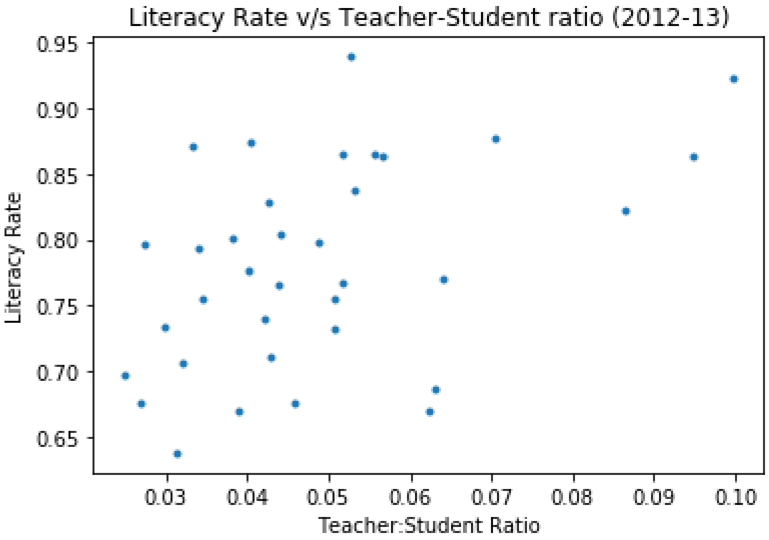


In [24]:

```
lit_12_13 = basic_12_13['literacy_rate']/100
tch_stud_12_13 = (teacher_12_13['govt_tch_3'] + teacher_12_13['pvt_tch_3']) / (enrolment_12_13['govt_enr_3'
```

In [25]:

```
plt.plot(tch_stud_12_13, lit_12_13, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2012-13)')
plt.show()
```

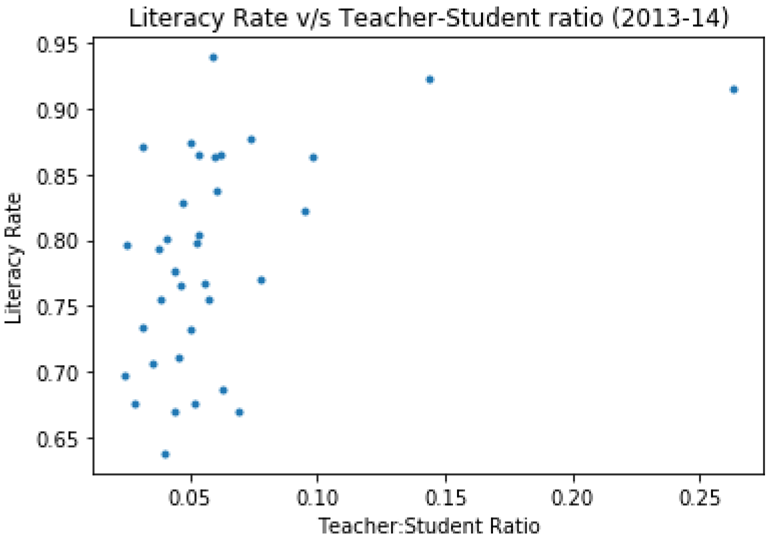


In [26]:

```
lit_13_14 = basic_13_14['literacy_rate']/100
tch_stud_13_14 = (teacher_13_14['govt_tch_3'] + teacher_13_14['pvt_tch_3']) / (enrolment_13_14['govt_enr_3'
```

In [27]:

```
plt.plot(tch_stud_13_14, lit_13_14, '.')
plt.xlabel('Teacher:Student Ratio')
plt.ylabel('Literacy Rate')
plt.title('Literacy Rate v/s Teacher-Student ratio (2013-14)')
plt.show()
```



Data Cleaning

In [28]:

```
df1 = pd.DataFrame()
states = list(basic_13_14['statname'])
literacy = list(basic_13_14['literacy_rate'])
```

Basic

In [29]:

```
cols = ['area_sqkm', 'schools', 'tot_population', 'literacy_rate']
```

In [30]:

```
df1.insert(0, "area_sqkm", basic_13_14['area_sqkm'])
df1.insert(1, "schools", basic_13_14['schools'])
df1.insert(2, "tot_population", basic_13_14['tot_population'])
df1.insert(3, "literacy_rate", basic_13_14['literacy_rate'])
```

School Facilities

In [31]:

```
comp_cols = ['statcd', 'ac_year', 'statname', 'govt_sch_9', 'pvt_sch_9', 'govt_sch_r_9', 'pvt_sch_r_9']
```

In [32]:

```
for col in comp_cols:
    del facilities_13_14[col]
```

In [33]:

```
index = 4
for col in list(facilities_13_14.columns.values):
    df1.insert(index, col, facilities_13_14[col])
```

School Condition

In [34]:

```
comp_cols = ['statcd', 'ac_year', 'statname', 'sdg', 'tlm']
for col in comp_cols:
    del condition_13_14[col]
for col in list(condition_13_14.columns.values):
    df1.insert(index, col, condition_13_14[col])
```

Enrolment

In [35]:

```python
comp_cols = ['statcd', 'ac_year', 'statname', 'govt_enr_9', 'pvt_enr_9', 'govt_enr_r_9', 'pvt_enr_r_9', 'en
```

In [36]:

```python
for col in comp_cols:
    del enrolment_13_14[col]
```

In [37]:

```python
for col in list(enrolment_13_14.columns.values):
    df1.insert(index, col, enrolment_13_14[col])
```

Teacher

In [38]:

```python
comp_cols = ['statcd', 'ac_year', 'statname', 'govt_tch_9', 'pvt_tch_9', 'tch_bs', 'tch_s', 'tch_hs', 'tch_
```

In [39]:

```python
for col in comp_cols:
    del teacher_13_14[col]
```

In [40]:

```python
for col in list(teacher_13_14.columns.values):
    df1.insert(index, col, teacher_13_14[col])
```

Clustering

Removing nans & infs

In [41]:

```python
df1 = df1.fillna(0)
df1 = df1.replace(np.inf, 0)
np.any(np.isnan(df1))
```

Out[41]:

```
False
```

In [42]:

```python
np.all(np.isfinite(df1))
```

Out[42]:

```
True
```

PCA + K-Means

In [43]:

```python
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
from mpl_toolkits.mplot3d import Axes3D
```

In [44]:

```python
pca = PCA(n_components=2)
```

In [45]:

```python
df1_trnsfrm = pca.fit_transform(df1)
```

In [46]:

```python
km_pca = KMeans(n_clusters=3)
```

In [47]:

```python
km_pca.fit(df1_trnsfrm)
```

Out[47]:

```
KMeans(algorithm='auto', copy_x=True, init='k-means++', max_iter=300,
       n_clusters=3, n_init=10, n_jobs=None, precompute_distances='auto',
       random_state=None, tol=0.0001, verbose=0)
```

In [48]:

```python
km_pca.labels_
```

Out[48]:

```
array([1, 1, 1, 1, 1, 1, 1, 0, 2, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1])
```

In [49]:

```python
clus0 = [[], []]
clus1 = [[], []]
clus2 = [[], []]
for i, lab in enumerate(km_pca.labels_):
    if lab == 0:
        clus0[0].append(df1_trnsfrm[i, 0])
        clus0[1].append(df1_trnsfrm[i, 1])
    elif lab == 1:
        clus1[0].append(df1_trnsfrm[i, 0])
        clus1[1].append(df1_trnsfrm[i, 1])
    else:
        clus2[0].append(df1_trnsfrm[i, 0])
        clus2[1].append(df1_trnsfrm[i, 1])
```

In [50]:

```
c1 = []
c2 = []
c3 = []
c4 = []
c5 = []
c6 = []
c7 = []
c8 = []
c9 = []
c10 = []
for index, label in enumerate(km_pca.labels_):
    if label == 0:
        c1.append((states[index], literacy[index]))
    elif label == 1:
        c2.append((states[index], literacy[index]))
    elif label == 2:
        c3.append((states[index], literacy[index]))
#     elif label == 3:
#         c4.append((states[index], literacy[index]))
#     elif label == 4:
#         c5.append((states[index], literacy[index]))
#     elif label == 5:
#         c6.append((states[index], literacy[index]))
#     elif label == 6:
#         c7.append((states[index], literacy[index]))
#     elif label == 7:
#         c8.append((states[index], literacy[index]))
#     elif label == 8:
#         c9.append((states[index], literacy[index]))
#     else:
#         c10.append((states[index], literacy[index]))
```

In [51]:

```
c1
```

Out[51]:

```
[('RAJASTHAN                                                                                           ',
  67.06),
 ('BIHAR                                                                                               ',
  63.82),
 ('WEST BENGAL                                                                                         ',
  77.08),
 ('JHARKHAND                                                                                           ',
  67.63),
 ('ODISHA                                                                                              ',
  73.45),
 ('MADHYA PRADESH                                                                                      ',
  70.63),
 ('GUJARAT                                                                                             ',
  79.31),
 ('MAHARASHTRA                                                                                         ',
  82.91),
 ('ANDHRA PRADESH                                                                                      ',
  67.66),
 ('KARNATAKA                                                                                           ',
  75.6),
 ('TAMIL NADU                                                                                          ',
  80.33)]
```

```
In [52]: c2
```

```
Out[52]: [('JAMMU AND KASHMIR                                                                                   ',
           68.74),
          ('HIMACHAL PRADESH                                                                                    ',
           83.78),
          ('PUNJAB                                                                                              ',
           76.68),
          ('CHANDIGARH                                                                                          ',
           86.43),
          ('UTTARANCHAL                                                                                         ',
           79.63),
          ('HARYANA                                                                                             ',
           76.64),
          ('DELHI                                                                                               ',
           86.34),
          ('SIKKIM                                                                                              ',
           82.2),
          ('ARUNACHAL PRADESH                                                                                   ',
           66.95),
          ('NAGALAND                                                                                            ',
           80.11),
          ('MANIPUR                                                                                             ',
           79.85),
          ('MIZORAM                                                                                             ',
           91.58),
          ('TRIPURA                                                                                             ',
           87.75),
          ('MEGHALAYA                                                                                           ',
           75.48),
          ('ASSAM                                                                                               ',
           73.18),
          ('CHHATTISGARH                                                                                        ',
           71.04),
          ('DAMAN & DIU                                                                                         ',
           87.07),
          ('DADRA & NAGAR HAVELI                                                                                ',
           77.65),
          ('GOA                                                                                                 ',
           87.4),
          ('LAKSHADWEEP                                                                                         ',
           92.28),
          ('KERALA                                                                                              ',
  93.91),
 ('PONDICHERRY                                                                                          ',
  86.55),
 ('ANDAMAN & NICOBAR ISLANDS                                                                            ',
  86.27)]
```

In [53]:

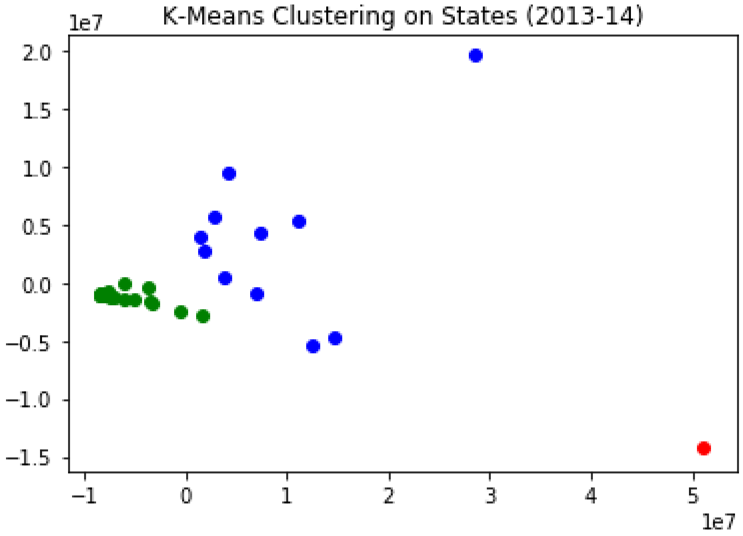
```
c3
```

Out[53]:

```
[('UTTAR PRADESH                                                                                        ',
  69.72)]
```

In [54]:

```
plt.scatter(clus0[0], clus0[1], c=['blue'])
plt.scatter(clus1[0], clus1[1], c=['green'])
plt.scatter(clus2[0], clus2[1], c=['red'])
plt.title("K-Means Clustering on States (2013-14)")
plt.show()
```



Observation of clustering

K-Means clustering was applied to the states of India.

Clustering was performed for k=3 clusters.

Two clusters (represented by blue and green) have a good number of points in the third cluster. However, the third cluster (red) contains only a single state.

The state in the third cluster is Uttar Pradesh with a literacy rate of 69.72%.

Random points were sampled from the other two clusters and the details about those points were observed.

Visualization School

In [55]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
```

Import Data

In [56]:

```
xls_06_07 = pd.ExcelFile('Data/processed/SRC_Rawdata_2006-07_mod.xls')
facilities_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
condition_06_07 = pd.read_excel(xls_06_07, 'School Condition')

xls_07_08 = pd.ExcelFile('Data/processed/SRC_Rawdata_2007-08_mod.xls')
facilities_07_08 = pd.read_excel(xls_07_08, 'School Facilities')
condition_07_08 = pd.read_excel(xls_07_08, 'School Condition')

xls_08_09 = pd.ExcelFile('Data/processed/SRC_Rawdata_2008-09_mod.xls')
facilities_08_09 = pd.read_excel(xls_08_09, 'School Facilities')
condition_08_09 = pd.read_excel(xls_08_09, 'School Condition')

xls_09_10 = pd.ExcelFile('Data/processed/SRC_Rawdata_2009-10_mod.xls')
facilities_09_10 = pd.read_excel(xls_09_10, 'School Facilities')
condition_09_10 = pd.read_excel(xls_09_10, 'School Condition')

xls_10_11 = pd.ExcelFile('Data/processed/SRC_Rawdata_2010-11_mod.xls')
facilities_10_11 = pd.read_excel(xls_10_11, 'School Facilities')
condition_10_11 = pd.read_excel(xls_10_11, 'School Condition')

xls_11_12 = pd.ExcelFile('Data/processed/SRC_Rawdata_2011-12_mod.xls')
facilities_11_12 = pd.read_excel(xls_11_12, 'School Facilities')
condition_11_12 = pd.read_excel(xls_11_12, 'School Condition')

xls_12_13 = pd.ExcelFile('Data/processed/SRC_Rawdata_2012-13_mod.xls')
facilities_12_13 = pd.read_excel(xls_12_13, 'School Facilities')
condition_12_13 = pd.read_excel(xls_12_13, 'School Condition')

xls_13_14 = pd.ExcelFile('Data/processed/SRC_Rawdata_2013-14_mod.xls')
facilities_13_14 = pd.read_excel(xls_13_14, 'School Facilities')
condition_13_14 = pd.read_excel(xls_13_14, 'School Condition')
```

Normalization

In [57]:

```
def normalize(df, f_range=(0,100)):
    tdf = f_range[0] + ((df - df.min())*(f_range[1]-f_range[0]))/(df.max()-df.min())
    return tdf
```

Visualization

In [58]:

```
states = facilities_13_14['statname'][1:]
states = [s[:3] for s in states]
states_dict = dict()
for i, s in enumerate(states):
    states_dict[i] = s
```

In [59]:

```
years = ['2006-07', '2007-08', '2008-09', '2009-10', '2010-11', '2011-12', '2012-13', '2013-14']
```

Govt. Primary Schools

On observing the heatmaps below, we can observe that there has been a change in the number of schools in most states. States such as Uttarakhand, West Bengal, who have seen an increase in the number of schools have also recorded an increase in literacy rates. However, states such as Mizoram have seen a decline in the number of schools. This could be a result of shutting down of schools due to environmental crisis or other external factors. Despite the fall in the school count, the literacy rate has been stable on the higher side of the scale.

In [60]:

```
total_govt_schools_primary = pd.concat([facilities_06_07['govt_sch_1'], facilities_07_08['govt_sch_1'], fac

col_6 = pd.DataFrame(facilities_12_13['govt_sch_1'])
col_6 = col_6.drop(col_6.index[0]).reset_index()
del col_6['index']

col_7 = pd.DataFrame(facilities_13_14['govt_sch_1'])
col_7 = col_7.drop(col_7.index[0]).reset_index()
del col_7['index']

total_govt_schools_primary = pd.concat([total_govt_schools_primary, col_6, col_7], ignore_index=True, axis=
```

In [61]:

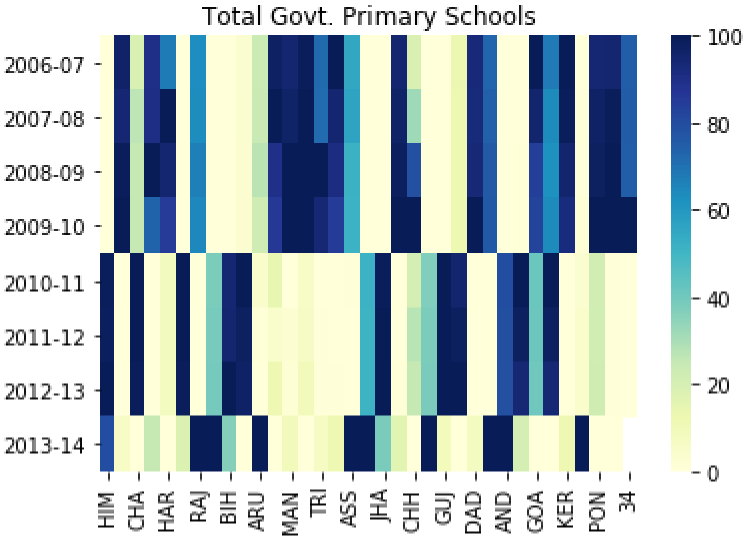
```
total_govt_schools_primary = total_govt_schools_primary.T
total_govt_schools_primary.index = years
total_govt_schools_primary = total_govt_schools_primary.rename(columns=states_dict)
```

In [62]:

```
trans_data = normalize(total_govt_schools_primary)
```

In [63]:

```python
ax = sns.heatmap(trans_data, cmap="YlGnBu")
ax.set_title('Total Govt. Primary Schools')
plt.show()
```



In [64]:

```python
total_priv_schools_primary = pd.concat([facilities_06_07['pvt_sch_1'], facilities_07_08['pvt_sch_1'], facil

col_6 = pd.DataFrame(facilities_12_13['pvt_sch_1'])
col_6 = col_6.drop(col_6.index[0]).reset_index()
del col_6['index']

col_7 = pd.DataFrame(facilities_13_14['pvt_sch_1'])
col_7 = col_7.drop(col_7.index[0]).reset_index()
del col_7['index']

total_priv_schools_primary = pd.concat([total_priv_schools_primary, col_6, col_7], ignore_index=True, axis=
```

In [65]:

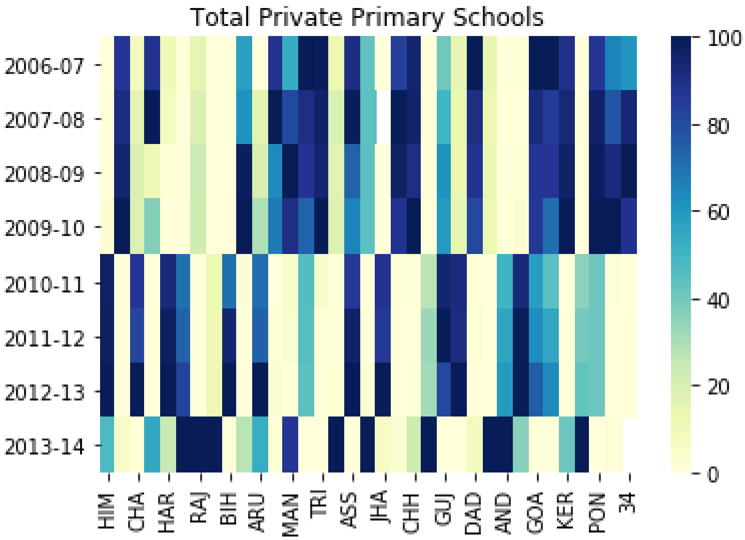
```python
total_priv_schools_primary = total_priv_schools_primary.T
total_priv_schools_primary.index = years
total_priv_schools_primary = total_priv_schools_primary.rename(columns=states_dict)
```

In [66]:

```python
trans_data = normalize(total_priv_schools_primary)
```

In [67]:

```
ax = sns.heatmap(trans_data, cmap="YlGnBu")
ax.set_title('Total Private Primary Schools')
plt.show()
```



Schools with less than 50 Students

In [68]:

```
total_lt_50 = pd.concat([facilities_06_07['less_50_e_all'], facilities_07_08['less_50_e_all'], facilities_0

col_6 = pd.DataFrame(facilities_12_13['less_50_e_all'])
col_6 = col_6.drop(col_6.index[0]).reset_index()
del col_6['index']

col_7 = pd.DataFrame(facilities_13_14['less_50_e_all'])
col_7 = col_7.drop(col_7.index[0]).reset_index()
del col_7['index']

total_lt_50 = pd.concat([total_lt_50, col_6, col_7], ignore_index=True, axis=1)
```

In [69]:

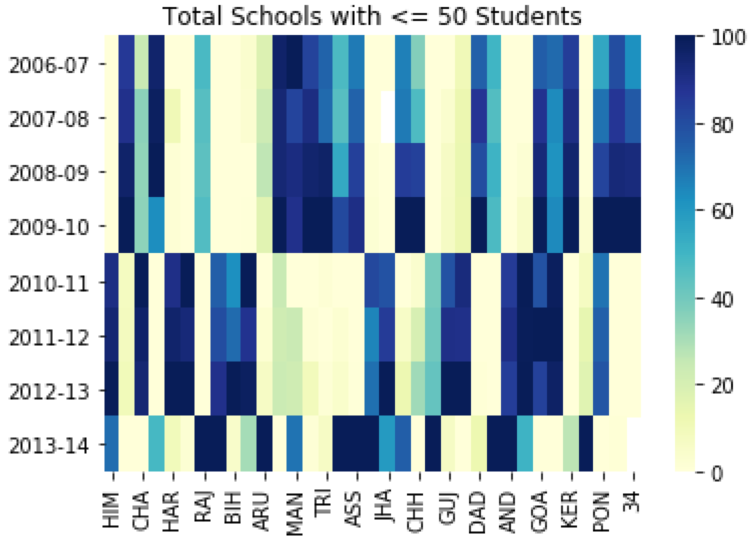
```
total_lt_50 = total_lt_50.T
total_lt_50.index = years
total_lt_50 = total_lt_50.rename(columns=states_dict)
```

In [70]:

```python
trans_data = normalize(total_lt_50)
```

In [71]:

```python
ax = sns.heatmap(trans_data, cmap="YlGnBu")
ax.set_title('Total Schools with <= 50 Students')
plt.show()
```



Total Classrooms which need Repairs

In [72]:

```python
total_cls_repairs = pd.concat([condition_06_07['cls_major5'], condition_07_08['cls_major5'], condition_08_0

col_6 = pd.DataFrame(condition_12_13['cls_major5'])
col_6 = col_6.drop(col_6.index[0]).reset_index()
del col_6['index']

col_7 = pd.DataFrame(condition_13_14['cls_major5'])
col_7 = col_7.drop(col_7.index[0]).reset_index()
del col_7['index']

total_cls_repairs = pd.concat([total_cls_repairs, col_6, col_7], ignore_index=True, axis=1)
```

In [73]:

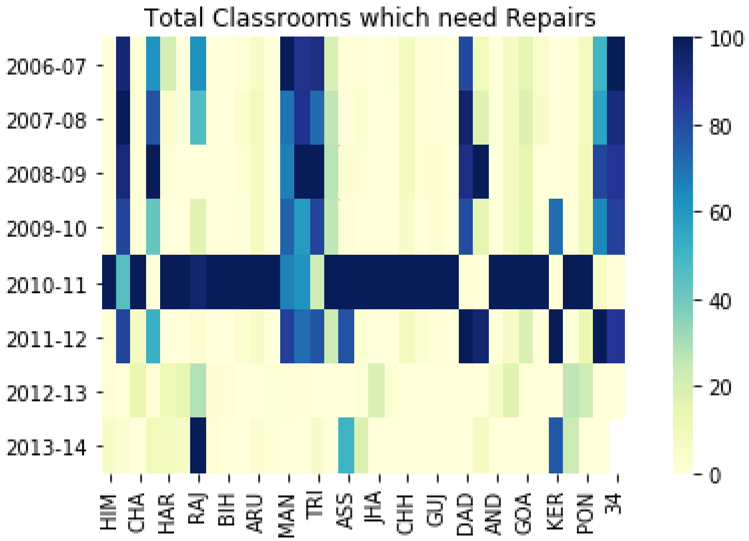
```
total_cls_repairs = total_cls_repairs.T
total_cls_repairs.index = years
total_cls_repairs = total_cls_repairs.rename(columns=states_dict)
```

In [74]:

```
trans_data = normalize(total_cls_repairs)
```

In [75]:

```
ax = sns.heatmap(trans_data, cmap="YlGnBu")
ax.set_title('Total Classrooms which need Repairs')
plt.show()
```



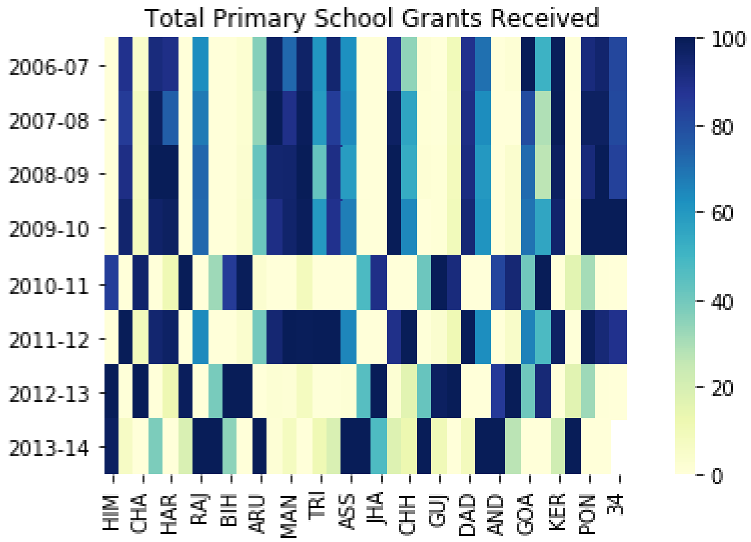
No. of Primary Schools Receiving Grants

In [76]:
```
total_prim_sch_grants = pd.concat([condition_06_07['sdg_1'], condition_07_08['sdg_1'], condition_08_09['sdg

col_6 = pd.DataFrame(condition_12_13['sdg_1'])
col_6 = col_6.drop(col_6.index[0]).reset_index()
del col_6['index']

col_7 = pd.DataFrame(condition_13_14['sdg_1'])
col_7 = col_7.drop(col_7.index[0]).reset_index()
del col_7['index']

total_prim_sch_grants = pd.concat([total_prim_sch_grants, col_6, col_7], ignore_index=True, axis=1)
```

In [77]:
```
total_prim_sch_grants = total_prim_sch_grants.T
total_prim_sch_grants.index = years
total_prim_sch_grants = total_prim_sch_grants.rename(columns=states_dict)
```

In [78]:
```
trans_data = normalize(total_prim_sch_grants)
```

In [79]:
```
ax = sns.heatmap(trans_data, cmap="YlGnBu")
ax.set_title('Total Primary School Grants Received')
plt.show()
```



In [ ]:

```

```

Imports

In [80]:

```python
#imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px

from sklearn.cluster import KMeans
import pandas as pd
import numpy as np
from sklearn import preprocessing
from sklearn.decomposition import PCA
from numpy import linalg as LA
import numpy.linalg as linalg
import copy

from collections import defaultdict
from sklearn.manifold import TSNE
```

In [81]:

```python
#Hashing corresponding to features.
metadata = pd.read_csv("data processed/2015_16_Statewise_Elementary_Metadata.csv")
metadata_hash = {}
for i in range(len(metadata)):
    metadata_hash[metadata.iat[i,0]] = metadata.iat[i,1]
```

for read file

In [82]:

```
def make_dataframe(xls_06_07):
    basicdata_06_07 = pd.read_excel(xls_06_07, 'Basic Data')
    schoolfacility_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
    schoolfacility_06_07.drop(['statcd','ac_year','statname'], axis = 1, inplace=True)
    schoolcondition_06_07 = pd.read_excel(xls_06_07, 'School Condition')
    schoolcondition_06_07.drop(['statcd','ac_year','statname'], axis = 1, inplace=True)
    enrolment_06_07 = pd.read_excel(xls_06_07, 'Enrolment')
    enrolment_06_07.drop(['statcd','ac_year','statname'], axis = 1, inplace=True)
    teacher_06_07 = pd.read_excel(xls_06_07, 'Teacher')
    teacher_06_07.drop(['statcd','ac_year','statname'], axis = 1, inplace=True)
    df = pd.concat([basicdata_06_07, schoolfacility_06_07, schoolcondition_06_07, enrolment_06_07, teacher_
    return df
```

In [83]:

```
# basicdata_06_07_features = set(list(basicdata_06_07.columns))
# schoolfacility_06_07_features = set(list(schoolfacility_06_07.columns))
# schoolcondition_06_07_featuers = set(list(schoolcondition_06_07.columns))
# enrolment_06_07_features = set(list(enrolment_06_07.columns))
# teacher_06_07_features = set(list(teacher_06_07.columns))
xls_06_07 = pd.ExcelFile('data processed/SRC_Rawdata_2006-07_mod.xls')
basicdata_06_07 = pd.read_excel(xls_06_07, 'Basic Data')
colmn = ['statcd', 'ac_year', 'statname']
schoolfacility_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
schoolfacility_06_07.drop(colmn, inplace=True, axis=1)
schoolcondition_06_07 = pd.read_excel(xls_06_07, 'School Condition')
schoolcondition_06_07.drop(colmn, inplace=True, axis=1)
enrolment_06_07 = pd.read_excel(xls_06_07, 'Enrolment')
enrolment_06_07.drop(colmn, inplace=True, axis=1)
teacher_06_07 = pd.read_excel(xls_06_07, 'Teacher')
teacher_06_07.drop(colmn, inplace=True, axis=1)

#Concate all excel into one
df06_07 = pd.concat([basicdata_06_07, schoolfacility_06_07,schoolcondition_06_07,enrolment_06_07,teacher_06

statename = list(df06_07['statname'])
```

Read and combine all year files

In [84]:

```
#read 2006-07
xls_06_07 = pd.ExcelFile('data processed/SRC_Rawdata_2006-07_mod.xls')
df_06_07 = make_dataframe(xls_06_07)
df_06_07.drop(['ner_p','ner_up','retentionrate'], axis = 1, inplace=True)
df_06_07 = df_06_07.sort_index(axis=1)
```

In [85]:

```
f0 = set(list(df_06_07.sort_index(axis=1).columns))

#read 2007-08
xls_07_08 = pd.ExcelFile('data processed/SRC_Rawdata_2007-08_mod.xls')
df_07_08 = make_dataframe(xls_07_08)
df_07_08.drop(['attendance_b_p','attendance_g_p','cls_minor','kitshed_py_1','kitshed_py_2','kitshed_py_3','
df_07_08 = df_07_08.sort_index(axis=1)

f2 = set(list(df_07_08.sort_index(axis=1).columns))

# concate 2006_08
frames = [df_06_07, df_07_08]
df_06_08 = pd.concat(frames,ignore_index=True)
df_06_08.drop(['ptr_py_1','ptr_py_2','ptr_py_3','ptr_py_4','ptr_py_5','ptr_py_all','scr_py_1','scr_py_2','s
print(df_06_08.shape)
```

```
(70, 630)
```

In [86]:

```
f0 = set(list(df_06_08.sort_index(axis=1).columns))

#read 2008-09
xls_08_09 = pd.ExcelFile('data processed/SRC_Rawdata_2008-09_mod.xls')
df_08_09 = make_dataframe(xls_08_09)
df_08_09.drop(['attendance_b_p','attendance_g_p','cls_minor','kitshed_py_1','kitshed_py_2','kitshed_py_3','
df_08_09 = df_08_09.sort_index(axis=1)

f3 = set(list(df_08_09.sort_index(axis=1).columns))

# concate 2006_09
frames = [df_06_08, df_08_09]
df_06_09 = pd.concat(frames,ignore_index=True)
df_06_09.drop(['avg_tch_py_1','avg_tch_py_2','avg_tch_py_3','avg_tch_py_4','avg_tch_py_5','avg_tch_py_all',
print(df_06_09.shape)
```

```
(105, 612)
```

In [ ]:

```

```

In [87]:

```
f0 = set(list(df_06_09.sort_index(axis=1).columns))

#read 2009-10
xls_09_10 = pd.ExcelFile('data processed/SRC_Rawdata_2009-10_mod.xls')
df_09_10 = make_dataframe(xls_09_10)
df_09_10.drop(['attendance_b_p','attendance_g_p','cls_minor','kitshed_py_1','kitshed_py_2','kitshed_py_3','
df_09_10 = df_09_10.sort_index(axis=1)

f4 = set(list(df_09_10.sort_index(axis=1).columns))

# concate 2006_10
frames = [df_06_09, df_09_10]
df_06_10 = pd.concat(frames,ignore_index=True)
print(df_06_10.shape)
```

```
(140, 612)
```

In [88]:

```
f0 = set(list(df_06_10.sort_index(axis=1).columns))

#read 2010-11
xls_10_11 = pd.ExcelFile('data processed/SRC_Rawdata_2010-11_mod.xls')
df_10_11 = make_dataframe(xls_10_11)
df_10_11.columns = map(str.lower, df_10_11.columns)
df_10_11.columns = df_10_11.columns.str.replace(' ', '_')
df_10_11 = df_10_11.sort_index(axis=1)

f5 = set(list(df_10_11.sort_index(axis=1).columns))
d5_1 = list(f5.difference(f0))

df_10_11.drop(d5_1, axis = 1, inplace=True)
f5 = set(list(df_10_11.sort_index(axis=1).columns))
d5_2 = list(f0.difference(f5))

df_06_10.drop(d5_2, axis = 1, inplace=True)
f0 = set(list(df_06_10.sort_index(axis=1).columns))

df_10_11 = df_10_11.loc[:, ~df_10_11.columns.duplicated()]

# concate 2006_11
frames = [df_06_10, df_10_11]
df_06_11 = pd.concat(frames,ignore_index=True)
print(df_06_11.shape)
```

```
(175, 604)
```

In [89]:

```
f0 = set(list(df_06_11.columns))

#read 2011-12
xls_11_12 = pd.ExcelFile('data processed/SRC_Rawdata_2011-12_mod.xls')
df_11_12 = make_dataframe(xls_11_12)
df_11_12.columns = map(str.lower, df_11_12.columns)
df_11_12.columns = df_11_12.columns.str.replace(' ', '_')
df_11_12 = df_11_12.sort_index(axis=1)

f6 = set(list(df_11_12.columns))
d6_1 = list(f6.difference(f0))
df_11_12.drop(d6_1, axis = 1, inplace=True)

f6 = set(list(df_11_12.columns))
d6_2 = list(f0.difference(f6))
df_06_11.drop(d6_2, axis = 1, inplace=True)
f0 = set(list(df_06_11.columns))

# concate 2006_12
frames = [df_06_11, df_11_12]
df_06_12 = pd.concat(frames,ignore_index=True)
print(df_06_12.shape)

```

```
(211, 512)
```

In [90]:

```
f0 = set(list(df_06_12.columns))

#read 2012-13
xls_12_13 = pd.ExcelFile('data processed/SRC_Rawdata_2012-13_mod.xls')
df_12_13 = make_dataframe(xls_12_13)
df_12_13.columns = map(str.lower, df_12_13.columns)
df_12_13.columns = df_12_13.columns.str.replace(' ', '_')
df_12_13 = df_12_13.sort_index(axis=1)

f7 = set(list(df_12_13.columns))
d7_1 = list(f0.difference(f7))
df_06_12.drop(d7_1, axis = 1, inplace=True)
f0 = set(list(df_06_12.columns))

d7_2 = list(f7.difference(f0))
df_12_13.drop(d7_2, axis = 1, inplace=True)
f7 = set(list(df_12_13.columns))

# concate 2006_13
frames = [df_06_12, df_12_13]
df_06_13 = pd.concat(frames,ignore_index=True)
print(df_06_13.shape)
```

```
(247, 464)
```

In [91]:

```
f0 = set(list(df_06_13.columns))

#read 2013-14
xls_13_14 = pd.ExcelFile('data processed/SRC_Rawdata_2013-14_mod.xls')
df_13_14 = make_dataframe(xls_13_14)
df_13_14.columns = map(str.lower, df_13_14.columns)
df_13_14.columns = df_13_14.columns.str.replace(' ', '_')
df_13_14 = df_13_14.sort_index(axis=1)

f8 = set(list(df_13_14.columns))
d8_1 = list(f0.difference(f8))
df_06_13.drop(d8_1, axis = 1, inplace=True)
f0 = set(list(df_06_13.columns))

d8_2 = list(f8.difference(f0))
df_13_14.drop(d8_2, axis = 1, inplace=True)
f8 = set(list(df_13_14.columns))

# concate 2006_14
frames = [df_06_13, df_13_14]
df_06_14 = pd.concat(frames,ignore_index=True)
print(df_06_14.shape)
f0 = set(list(df_06_14.columns))
```

```
(283, 435)
```

In [92]:

```
#file combined from 2006-2014 and stored in allyear.csv
print(df_06_14.shape)
df_06_14.to_csv('allyear.csv')
```

```
(283, 435)
```

In [93]:

```
all_feature = list(df06_07.columns)
```

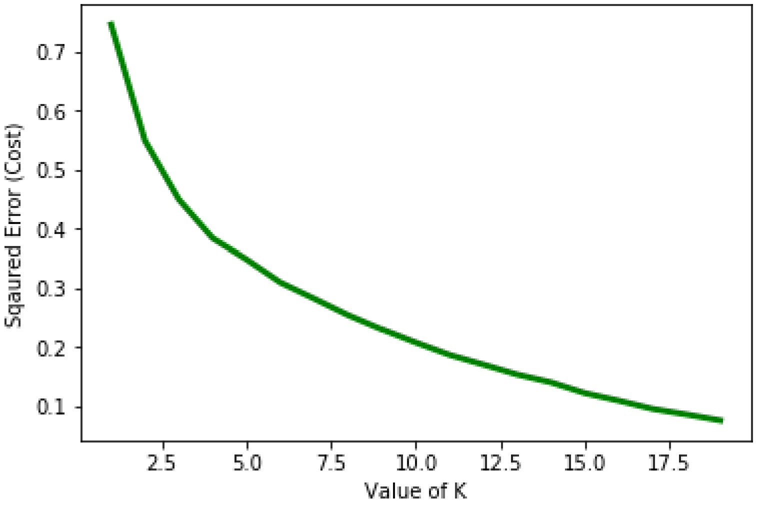
In [94]:

```
#remove unnecessary data field
df06_07 = df06_07.drop('statcd', axis=1)
df06_07 = df06_07.drop('ac_year', axis=1)
df06_07 = df06_07.drop('statname', axis=1)
df06_07 = df06_07.drop('area_sqkm', axis=1)

all_features_df06_07 = list(df06_07.columns)
df06_07  = df06_07.fillna(0)

#Normalization
x = df06_07.values
min_max_scaler = preprocessing.Normalizer()
x_scaled = min_max_scaler.fit_transform(x)

x_scaled= x_scaled.tolist()
dataset=copy.deepcopy(x_scaled)
```

Finding Value of K for performing K-Means Clustering

In [95]:

```
#Here we find out best k value to perform kmeans clustering.
dataset=copy.deepcopy(x_scaled)
cost =[]
for i in range(1, 20):
    KM = KMeans(n_clusters = i, max_iter = 500)
    KM.fit(dataset)
    cost.append(KM.inertia_)

plt.plot(range(1, 20), cost, color ='g', linewidth ='3')
plt.xlabel("Value of K")
plt.ylabel("Sqaured Error (Cost)")
plt.show()
```



Data visualization

In [96]:

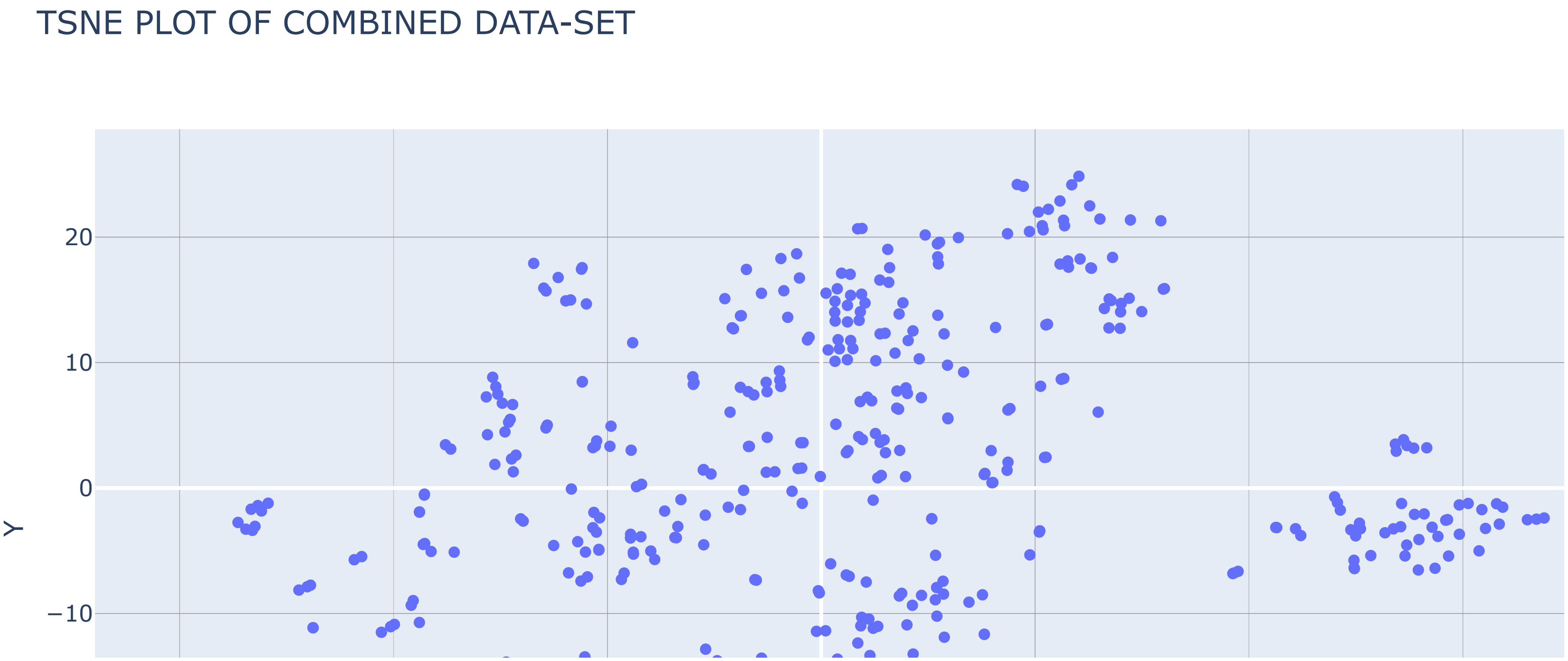
```
from sklearn.decomposition import TruncatedSVD
```

In [97]:

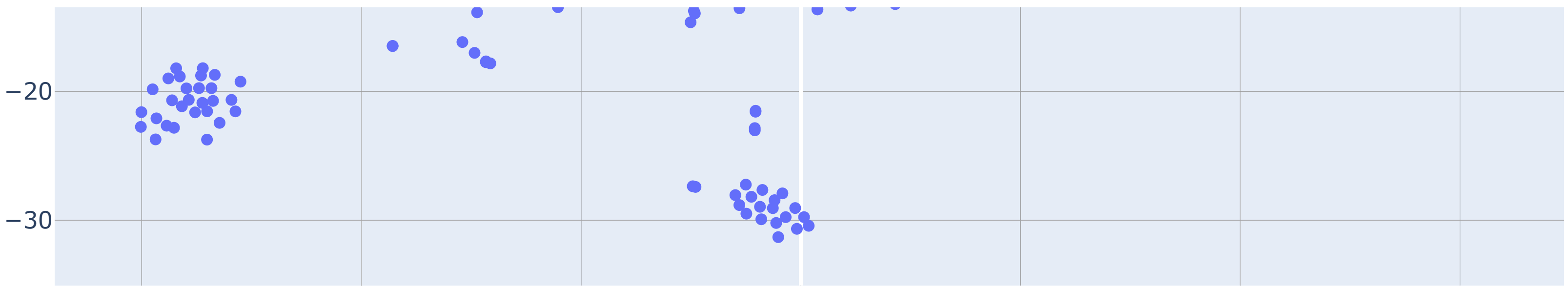
```
final_total_feature = list(df_06_14.columns)
feature_drop4clustering = ['statcd','area_sqkm','blocks','clusters','villages']
df_06_14.drop(feature_drop4clustering, axis = 1, inplace=True)

update_statename= []
for i in list(df_06_14['statname']):
    update_statename.append(str(i).split('    ')[0])

new_df = pd.DataFrame({'statname': update_statename})
df_06_14.update(new_df)

# df_06_14['year_state'] = df_06_14[['ac_year', 'statname']].apply(lambda x: ''.join(str(x)), axis=1)

labels = []
for i,j in zip((list(df_06_14['ac_year'])),(list(df_06_14['statname']))):
    labels.append(str(str(i)+" "+str(j)))

# new_df = pd.DataFrame({'year_state': year_state})
# df_06_14.update(new_df)

f12= ['ac_year','statname']
df_06_14.drop(f12, axis = 1, inplace=True)

all_features = list(df_06_14.columns)
df_06_14  = df_06_14.fillna(0)

for i in list(df_06_14.columns):
    try:
        df_06_14[i].replace('-', 0, inplace=True)
        df_06_14[i].replace(' ', 0, inplace=True)
    except:
        pass
```

In [98]:

```
# Normalization of dataset
x = df_06_14.values.T
min_max_scaler = preprocessing.Normalizer()
x_scaled = min_max_scaler.fit_transform(x)

# df = pd.DataFrame(x_scaled)

#TSNE PLOTS
x_scaled= x_scaled.tolist()
dataset=copy.deepcopy(x_scaled)

X_embedded = TSNE(n_components=2).fit_transform(dataset)

x= [i[0] for i in X_embedded]
y= [i[1] for i in X_embedded]
df123 = pd.DataFrame(list(zip(x, y)), columns =['X', 'Y'])
fig = px.scatter(df123,x='X', y='Y',title="TSNE PLOT OF COMBINED DATA-SET")
fig.show()

```



TSNE PLOT OF COMBINED DATA-SET

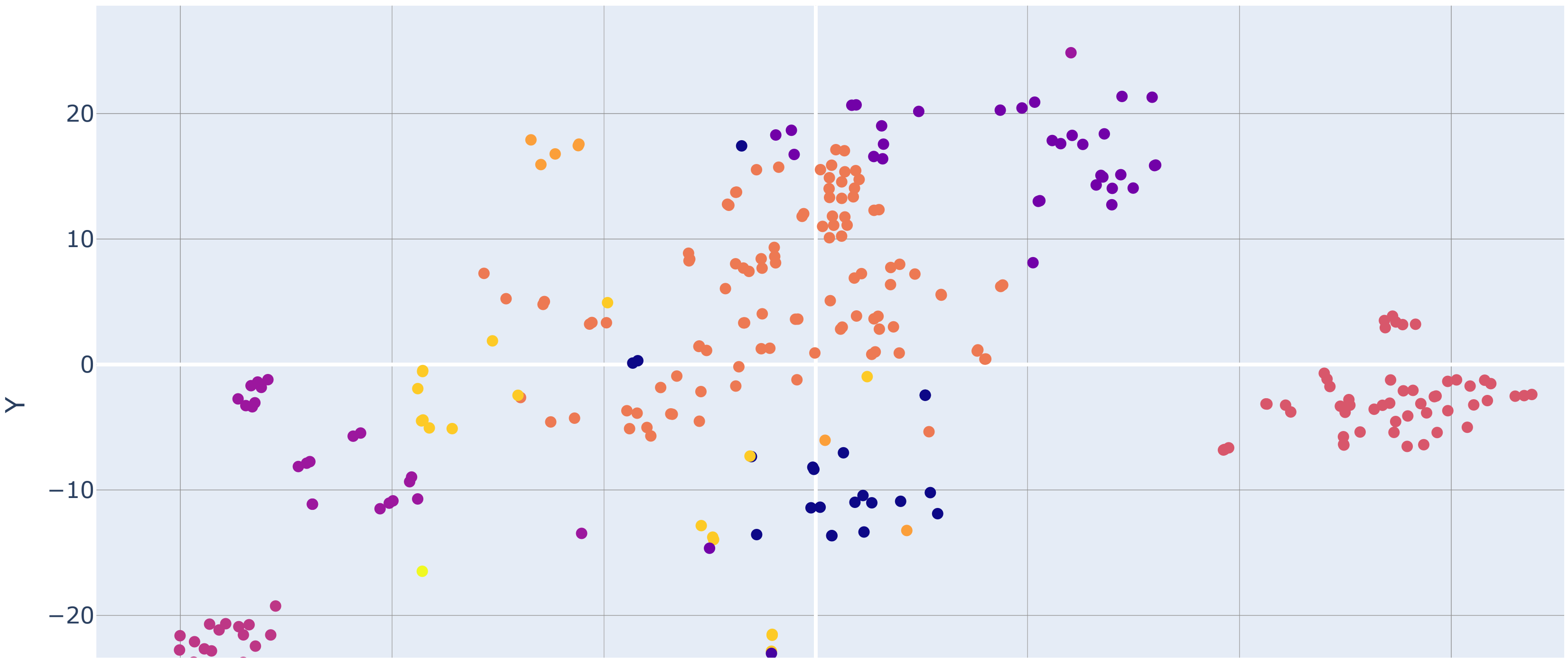


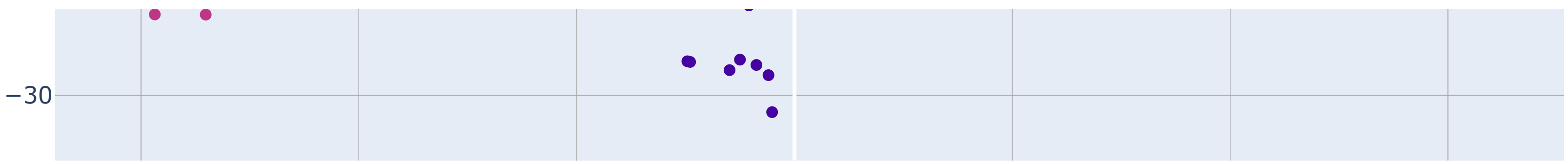
K_Means

In [99]:

```
#APPLY KMEANS AND PLOT CLUSTERS
kmeans = KMeans(n_clusters=10, random_state=0).fit(dataset)
# kmeans.labels_

knn_labels = list(kmeans.labels_)

cluster_name = defaultdict(list)
for i,j in zip(knn_labels,labels):
    try:
        cluster_name[i].append(j)
    except:
        cluster_name[i] = j

df = pd.DataFrame(list(zip(x, y,knn_labels,labels)), columns =['X', 'Y','Label','distname'])
fig = px.scatter(df, x='X', y='Y', color='Label',title = "K-MEANS CLUSTERING")
fig.show()
```

K-MEANS CLUSTERING





In [100]:

```
cluster_name
```

Out[100]:

```
defaultdict(list,
            {6: ['2006-07 A & N Islands',
              '2006-07 Andhra Pradesh',
              '2006-07 Arunachal Pradesh',
              '2006-07 Assam',
              '2006-07 Chhattisgarh',
              '2006-07 D & N Haveli',
              '2006-07 Daman & Diu',
              '2006-07 Haryana',
              '2006-07 Himachal Pradesh',
              '2006-07 Kerala',
              '2006-07 Lakshadweep',
              '2006-07 Madhya Pradesh',
              '2006-07 Maharashtra',
              '2006-07 Nagaland',
              '2006-07 Orissa',
              '2006-07 Puducherry',
              '2006-07 Punjab',
              '2006-07 Rajasthan',
              '2006-07 Sikkim'
```

Clustering performed on the whole dataset : After combining the data for all the years and then perform clustering over the dataset, we find the value of k by assigning the value of k and find squared error between the centroid and after a threshold select the value of k which 10.

In [101]:

```
# Normalization of dataset
x = df_06_14.values.T
min_max_scaler = preprocessing.Normalizer()
x_scaled = min_max_scaler.fit_transform(x)
# df = pd.DataFrame(x_scaled)
```

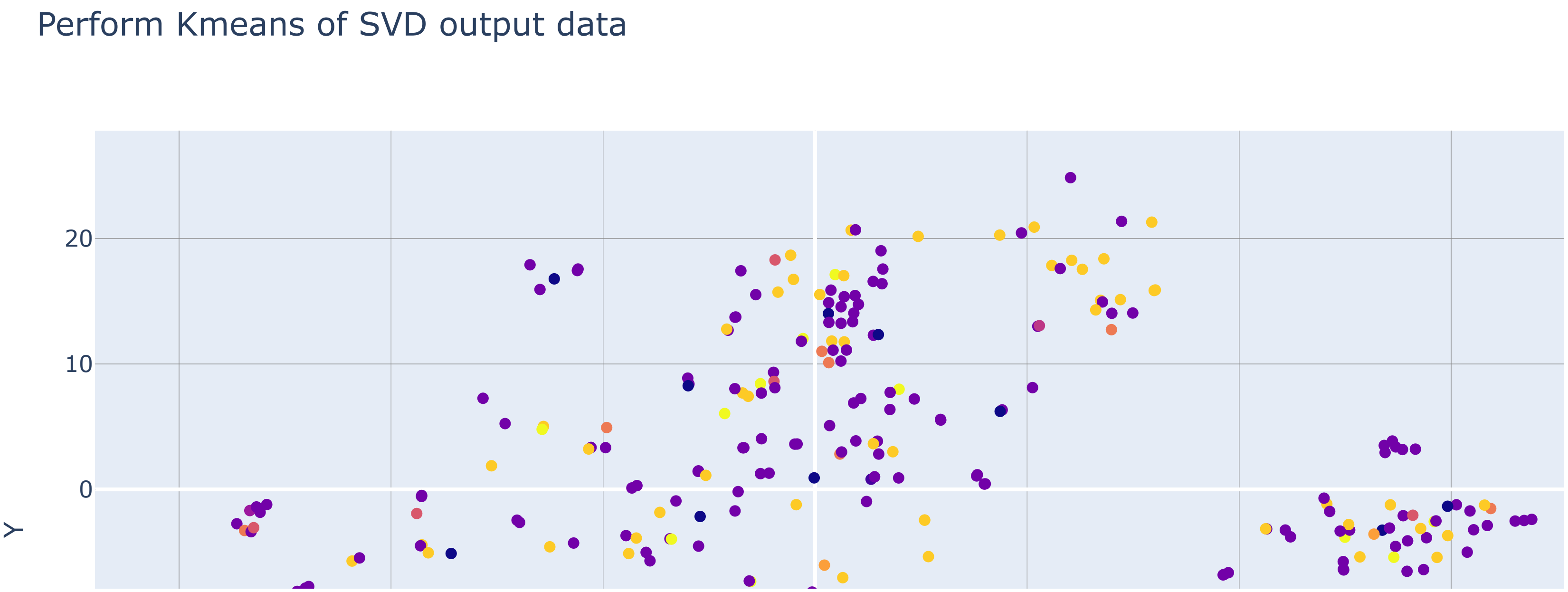
SVD performed

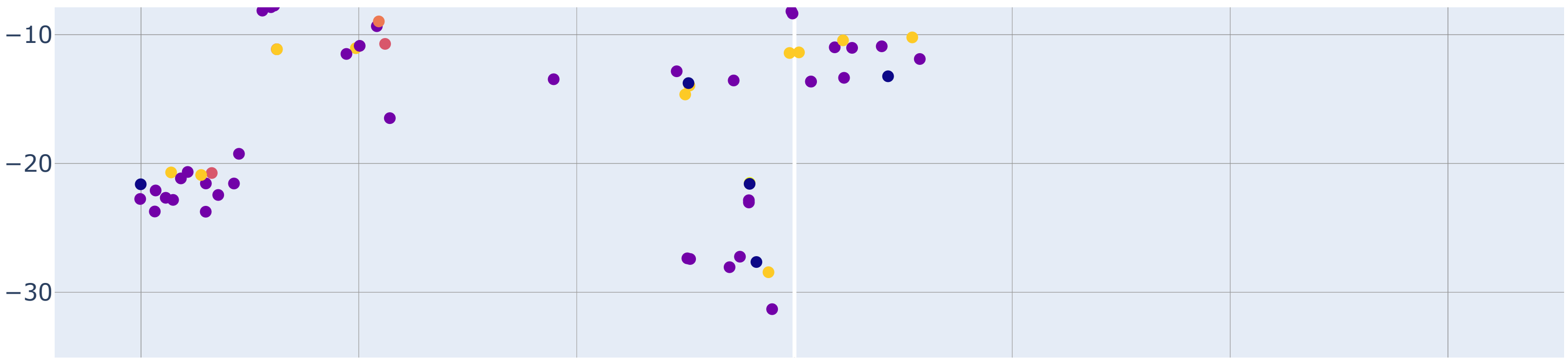
In [102]:

```
svd = TruncatedSVD(n_components=282, n_iter=7, random_state=42)
x_scaled = svd.fit_transform(x_scaled)
dataset=copy.deepcopy(x_scaled)
```

In [103]:

```
x = df_06_14.values.T
min_max_scaler = preprocessing.Normalizer()
x_scaled = min_max_scaler.fit_transform(x)
x_scaled= x_scaled.tolist()
dataset=copy.deepcopy(x_scaled)

kmeans = KMeans(n_clusters=10, random_state=0).fit(np.array(dataset).T)
# kmeans.labels_
knn_labels = list(kmeans.labels_)
cluster_name = defaultdict(list)
for i,j in zip(knn_labels,labels):
    try:
        cluster_name[i].append(j)
    except:
        cluster_name[i] = j
x= [i[0] for i in X_embedded]
y= [i[1] for i in X_embedded]

df = pd.DataFrame(list(zip(x, y,knn_labels,labels)), columns =['X', 'Y','Label','distname'])
fig = px.scatter(df, x='X', y='Y', color='Label',title = "Perform Kmeans of SVD output data")
fig.show()

```





conclusion - After SVD, K-Means clustering is not performed well

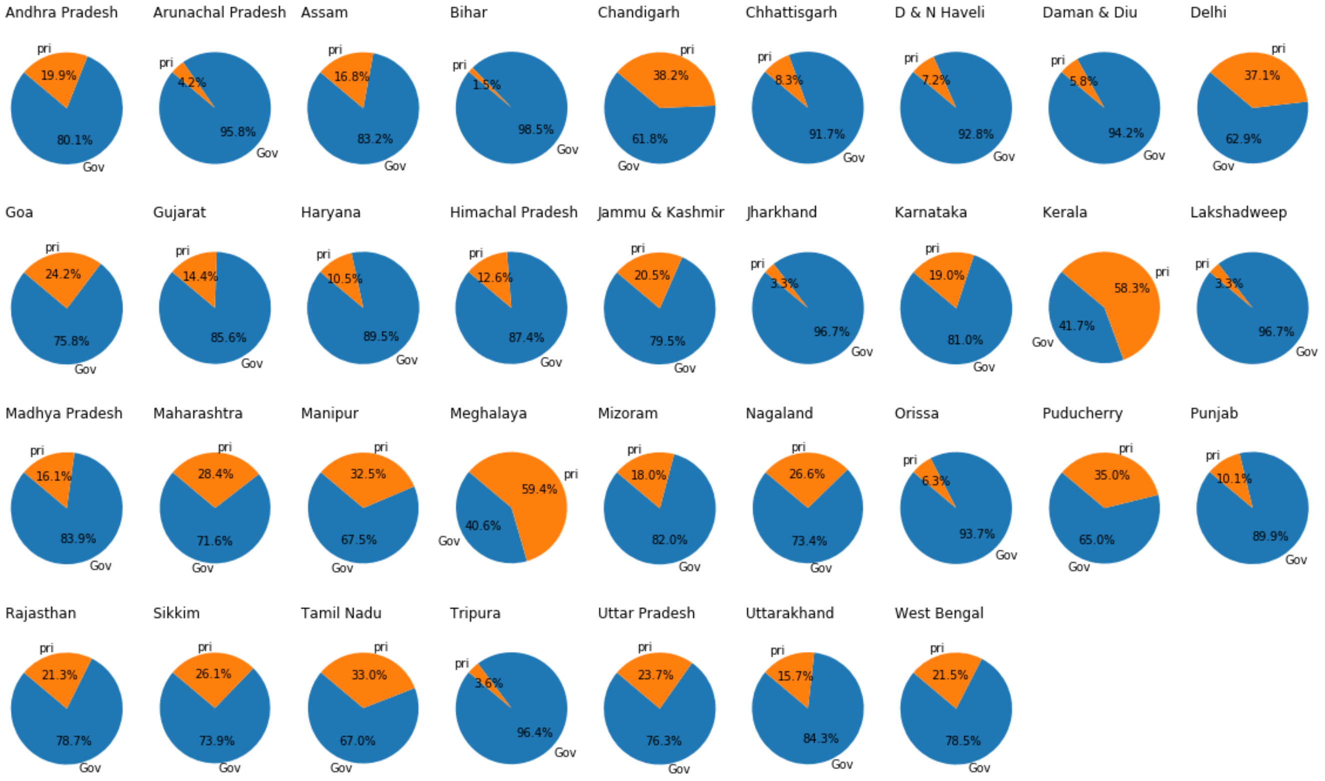
Visualization

Read file of 2006-07 and 2016-17(For Comparision between Goverment vs private school ratio through visualization)

In [104]:

```
#read
xls_06_07 = pd.ExcelFile('data processed/SRC_Rawdata_2006-07_mod.xls')
basicdata_06_07 = pd.read_excel(xls_06_07, 'Basic Data')
schoolfacility_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
schoolcondition_06_07 = pd.read_excel(xls_06_07, 'School Condition')
enrolment_06_07 = pd.read_excel(xls_06_07, 'Enrolment')
teacher_06_07 = pd.read_excel(xls_06_07, 'Teacher')

#drop unnecessary features
basicdata_06_07.drop(['statcd','ac_year','area_sqkm','districts_covered','blocks','clusters'], axis = 1, in

#prepare data frame for plot
df1 = basicdata_06_07.filter(['statname','schools','literacy_rate'], axis=1)
df2 = schoolfacility_06_07.govt_sch_1 + schoolfacility_06_07.govt_sch_2 + schoolfacility_06_07.govt_sch_3 +
df3 = schoolfacility_06_07.pvt_sch_1 + schoolfacility_06_07.pvt_sch_2 + schoolfacility_06_07.pvt_sch_3 + sc

govpercentage = (df2/df1.schools)*100
privpercentage = (df3/df1.schools)*100

df = pd.concat([df1, df2, df3,govpercentage,privpercentage], axis=1, sort=False)
df = df.rename(columns = {"statname": "STATNAME", "schools":"SCHTOT", 0:"SCHTOTG", 1:"SCHTOTPR", 2:"GOVPER"

plt.figure(figsize=(20,12))
print("Government vs Private School Ratio In 2006-07")
for i in range(1,len(df)):
    plt.subplot(4,9,i)
    plt.title(df['STATNAME'][i],loc='left')
    top = ['Gov','pri']
    uttar = df.loc[df['STATNAME'] == df['STATNAME'][i],:]
    value =[float(uttar['SCHTOTG']/uttar['SCHTOT'])*100,float(uttar['SCHTOTPR']/uttar['SCHTOT'])*100]
    plt.pie(value, labels=top, autopct='%1.1f%%',startangle=140)
    plt.axis('equal')
plt.show()

```

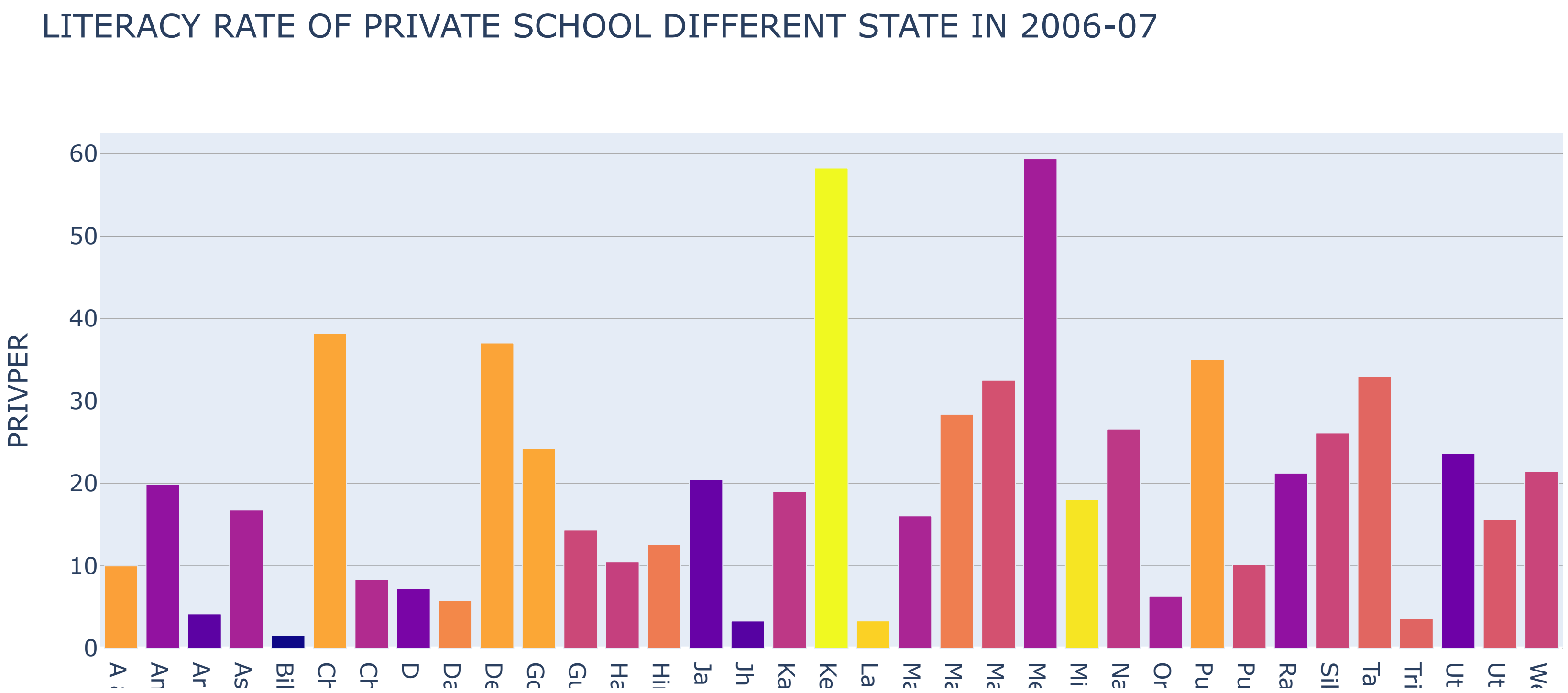
```
Government vs Private School Ratio In 2006-07
```



Literacy rate of gov. and private school

In [105]:

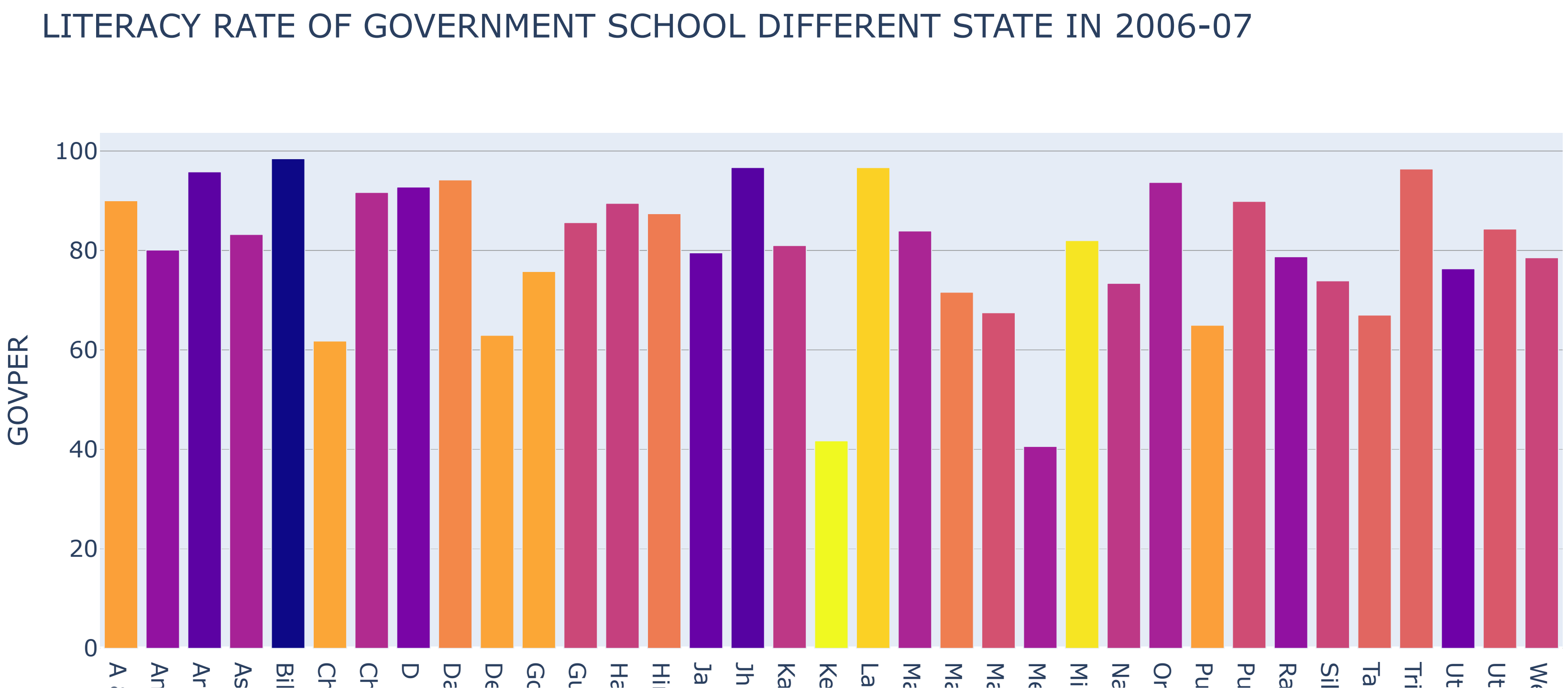
```
fig = px.bar(df, x='STATNAME', y='PRIVPER',color='literacy_rate', title = "LITERACY RATE OF PRIVATE SCHOOL
fig.show()
```



LITERACY RATE OF PRIVATE SCHOOL DIFFERENT STATE IN 2006-07

In [106]:

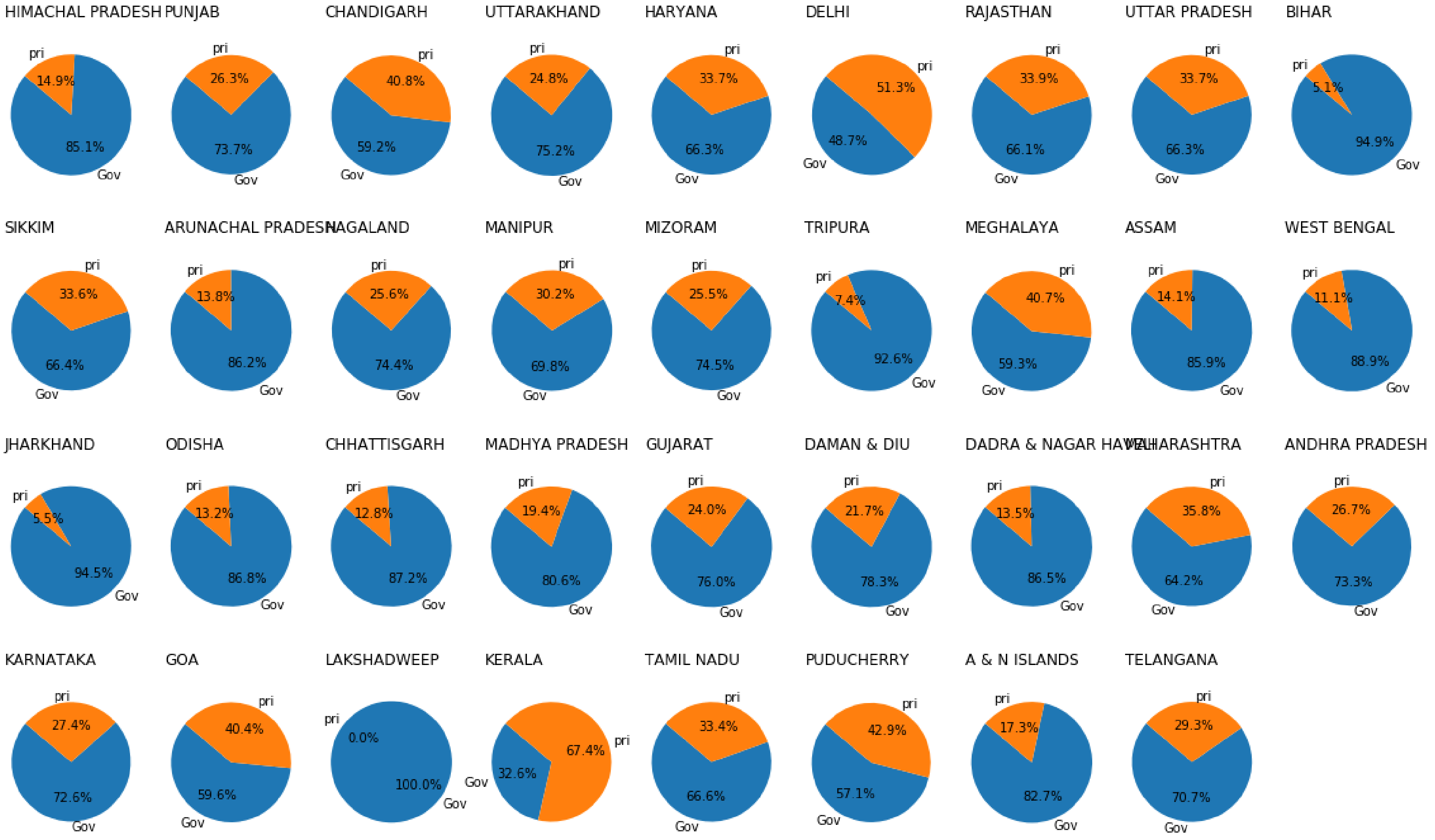
```
fig = px.bar(df, x='STATNAME', y='GOVPER',color='literacy_rate', title = "LITERACY RATE OF GOVERNMENT SCHOO
fig.show()
```



In [107]:

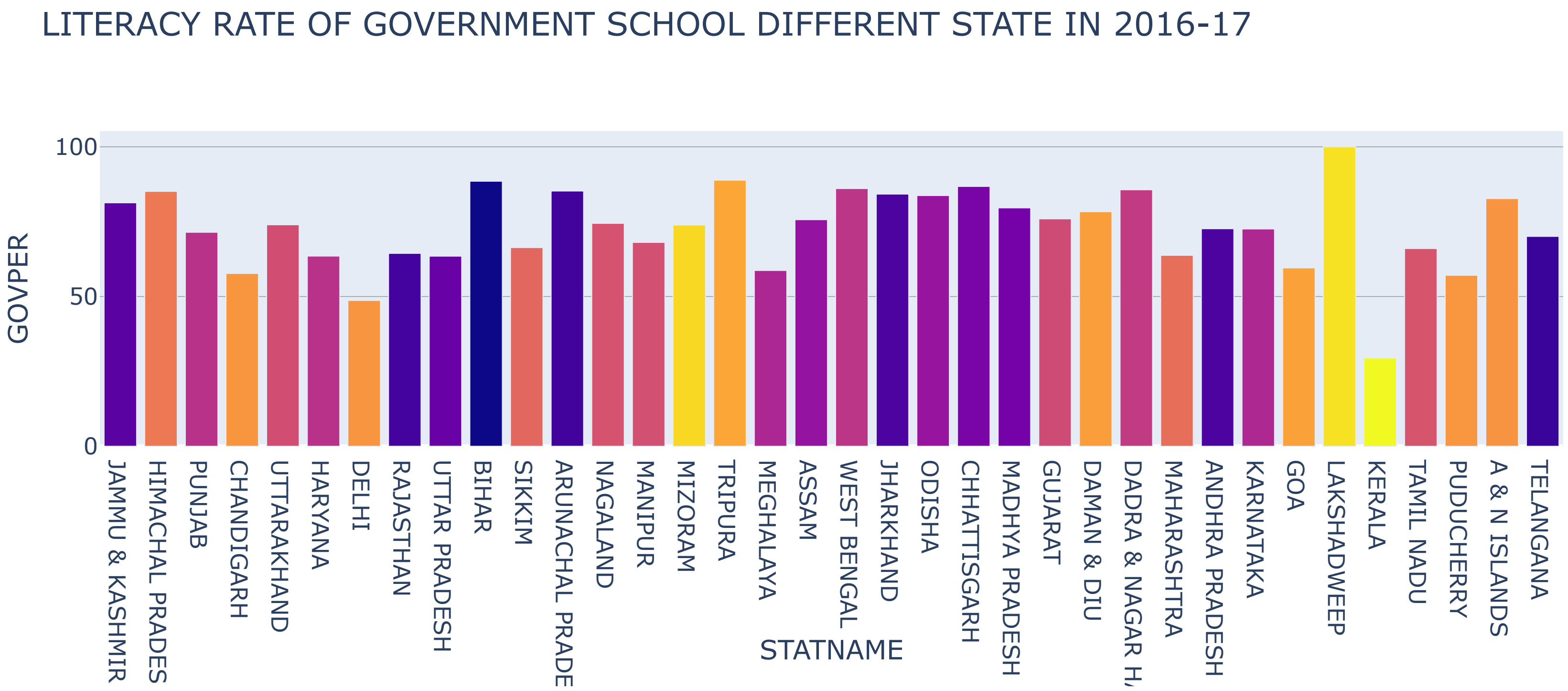
```
data = pd.read_csv("data processed/2016-17_Elementary.csv")
features = list(data.columns)
unique = list(set(list(metadata_hash.keys())).intersection(set(features)))

x = []
for i in features:
    if i not in list(metadata_hash.keys()):
        x.append(i)
plt.figure(figsize=(20,12))
print("Government vs Private School Ratio In 2016-17")
for i in range(1,len(data)):
    plt.subplot(4,9,i)
    plt.title(data['STATNAME'][i],loc='left')
    top = ['Gov','pri']
    uttar = data.loc[data['STATNAME'] == data['STATNAME'][i],:]
    value =[float(uttar['SCHTOTG']/uttar['SCHTOT'])*100,float(uttar['SCHTOTP']/uttar['SCHTOT'])*100]
    plt.pie(value, labels=top, autopct='%1.1f%%',startangle=140)
    plt.axis('equal')
plt.show()
```

```
Government vs Private School Ratio In 2016-17
```


HIMACHAL PRADESH
pri
14.9%
85.1%
Gov
PUNJAB
pri
26.3%
73.7%
Gov
CHANDIGARH
pri
40.8%
59.2%
Gov
UTTARAKHAND
pri
24.8%
75.2%
Gov
HARYANA
pri
33.7%
66.3%
Gov
DELHI
pri
51.3%
48.7%
Gov
RAJASTHAN
pri
33.9%
66.1%
Gov
UTTAR PRADESH
pri
33.7%
66.3%
Gov
BIHAR
pri
5.1%
94.9%
Gov
SIKKIM
pri
33.6%
66.4%
Gov
ARUNACHAL PRADESH
pri
13.8%
86.2%
Gov
NAGALAND
pri
25.6%
74.4%
Gov
MANIPUR
pri
30.2%
69.8%
Gov
MIZORAM
pri
25.5%
74.5%
Gov
TRIPURA
pri
7.4%
92.6%
Gov
MEGHALAYA
pri
40.7%
59.3%
Gov
ASSAM
pri
14.1%
85.9%
Gov
WEST BENGAL
pri
11.1%
88.9%
Gov
JHARKHAND
pri
5.5%
94.5%
Gov
ODISHA
pri
13.2%
86.8%
Gov
CHHATTISGARH
pri
12.8%
87.2%
Gov
MADHYA PRADESH
pri
19.4%
80.6%
Gov
GUJARAT
pri
24.0%
76.0%
Gov
DAMAN & DIU
pri
21.7%
78.3%
Gov
DADRA & NAGAR HAVELI
pri
13.5%
86.5%
Gov
MAHARASHTRA
pri
35.8%
64.2%
Gov
ANDHRA PRADESH
pri
26.7%
73.3%
Gov
KARNATAKA
pri
27.4%
72.6%
Gov
GOA
pri
40.4%
59.6%
Gov
LAKSHADWEEP
pri
0.0%
100.0%
Gov
Gov
KERALA
pri
67.4%
32.6%
Gov
TAMIL NADU
pri
33.4%
66.6%
Gov
PUDUCHERRY
pri
42.9%
57.1%
Gov
A & N ISLANDS
pri
17.3%
82.7%
Gov
TELANGANA
pri
29.3%
70.7%
Gov

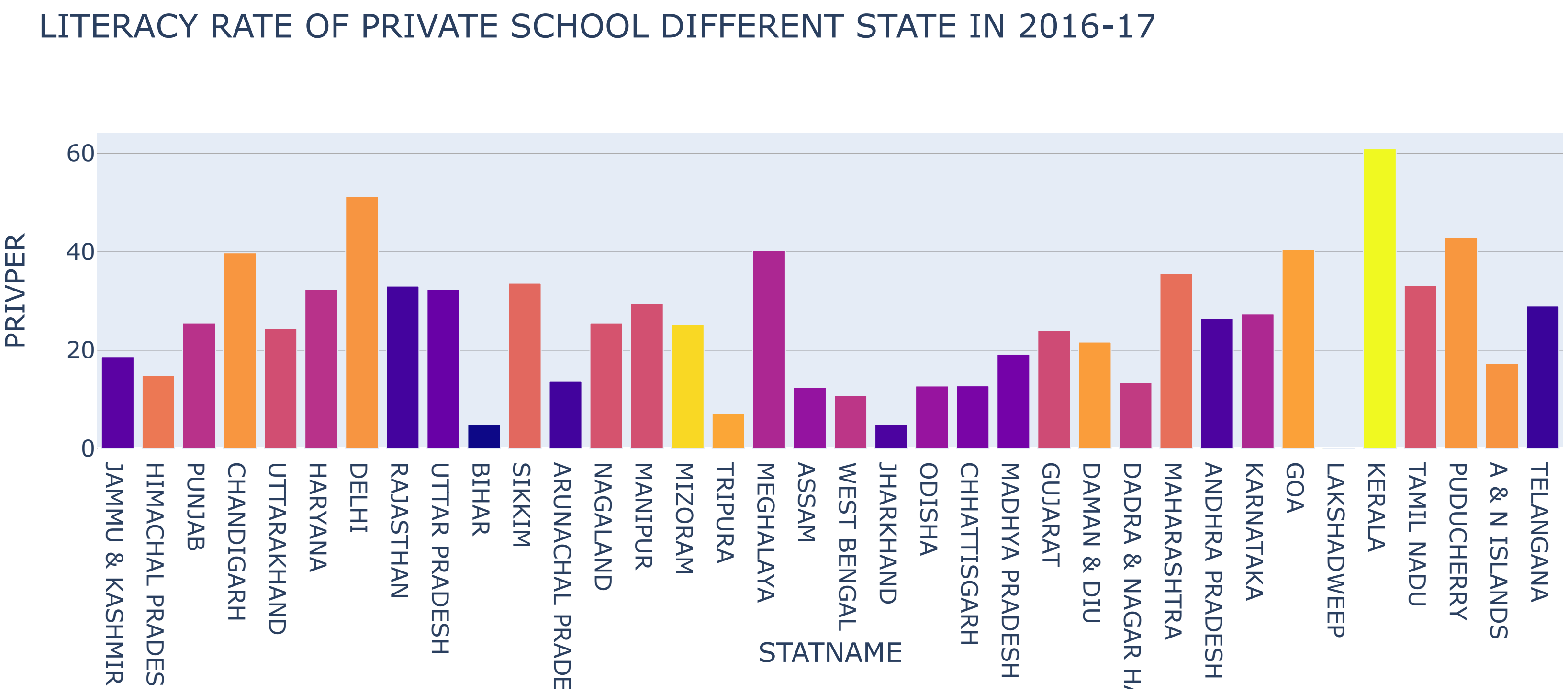
In [108]:

```
GOVPER = (data['SCHTOTG']/data['SCHTOT'])*100
PRIVPER = (data['SCHTOTP']/data['SCHTOT'])*100
data = pd.concat([data,GOVPER,PRIVPER], axis=1, sort=False)
data = data.rename(columns = {0:"GOVPER", 1:"PRIVPER"})

fig = px.bar(data, x='STATNAME', y='GOVPER',color='OVERALL_LI', title = "LITERACY RATE OF GOVERNMENT SCHOOL
fig.show()
```



In [109]:

```
fig = px.bar(data, x='STATNAME', y='PRIVPER',color='OVERALL_LI', title = "LITERACY RATE OF PRIVATE SCHOOL D
fig.show()
```



Observations by comparing literacy rate plots of 2006-07 and 2016-17:

We can observe that if the number of private schools is more in a state, then the literacy rate is also high, with the exception of Meghalaya.

A trend similar to private schools is not observed in the case of government schools.

This means that the resources and quality of education provided by government schools need to be improved instead of building new ones. And, more private schools must be built as they are improving the literacy rate.

The quality of private schools in Meghalaya is not very good.

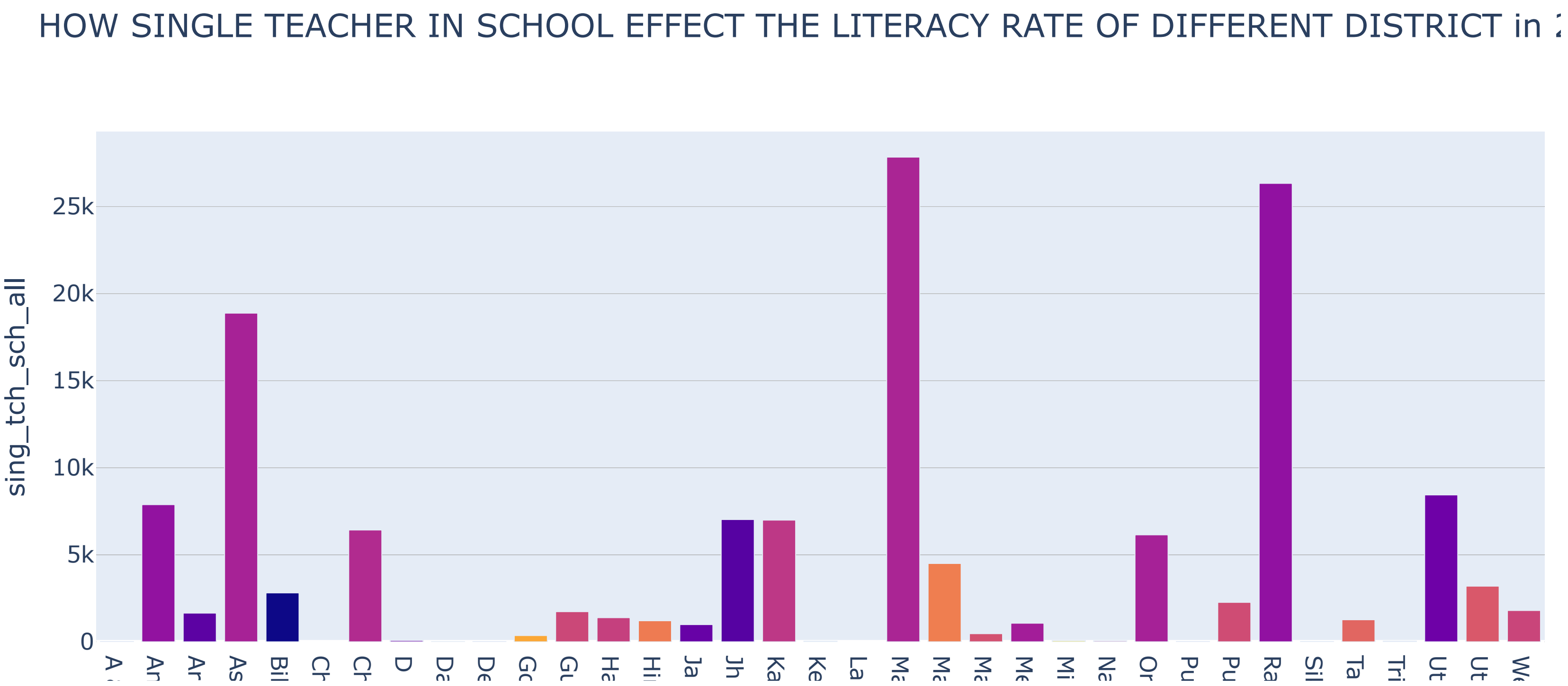
HOW SINGLE ROOM/TEACHER IN SCHOOL EFFECT THE LITERACY RATE OF DIFFERENT DISTRICT in 2006-07

In [110]:

```
xls_06_07 = pd.ExcelFile('data processed/SRC_Rawdata_2006-07_mod.xls')
basicdata_06_07 = pd.read_excel(xls_06_07, 'Basic Data')
colmn = ['statcd', 'ac_year', 'statname']
schoolfacility_06_07 = pd.read_excel(xls_06_07, 'School Facilities')
schoolfacility_06_07.drop(colmn, inplace=True, axis=1)
schoolcondition_06_07 = pd.read_excel(xls_06_07, 'School Condition')
schoolcondition_06_07.drop(colmn, inplace=True, axis=1)
enrolment_06_07 = pd.read_excel(xls_06_07, 'Enrolment')
enrolment_06_07.drop(colmn, inplace=True, axis=1)
teacher_06_07 = pd.read_excel(xls_06_07, 'Teacher')
teacher_06_07.drop(colmn, inplace=True, axis=1)

#Concate all excel into one
df06_07 = pd.concat([basicdata_06_07, schoolfacility_06_07,schoolcondition_06_07,enrolment_06_07,teacher_06

statename = list(df06_07['statname'])
all_feature = list(df06_07.columns)
```

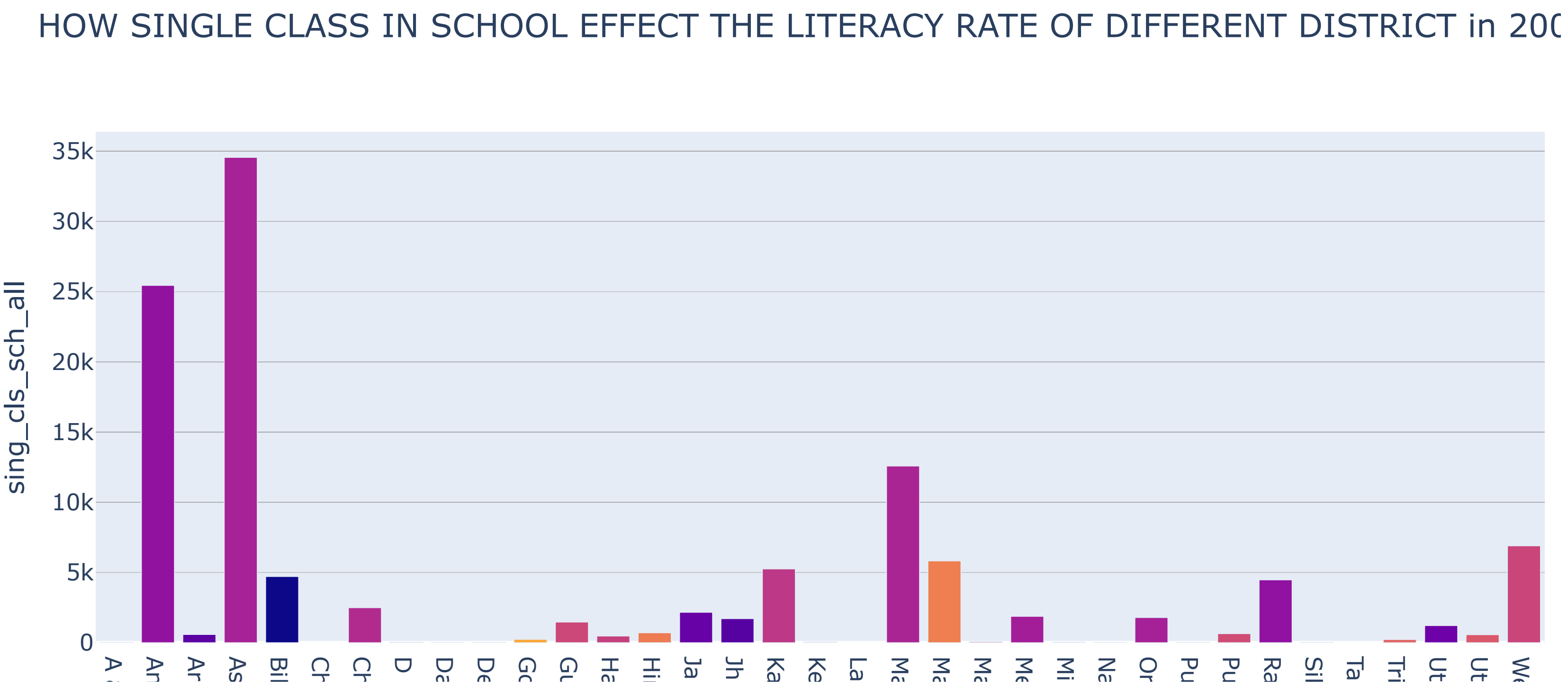
In [111]:

```
fig = px.bar(df06_07, x='statname', y='sing_tch_sch_all',color='literacy_rate', title="HOW SINGLE TEACHER I
fig.show()
```



In [112]:

```
fig = px.bar(df06_07, x='statname', y='sing_cls_sch_all',color='literacy_rate', title="HOW SINGLE CLASS IN
fig.show()
```



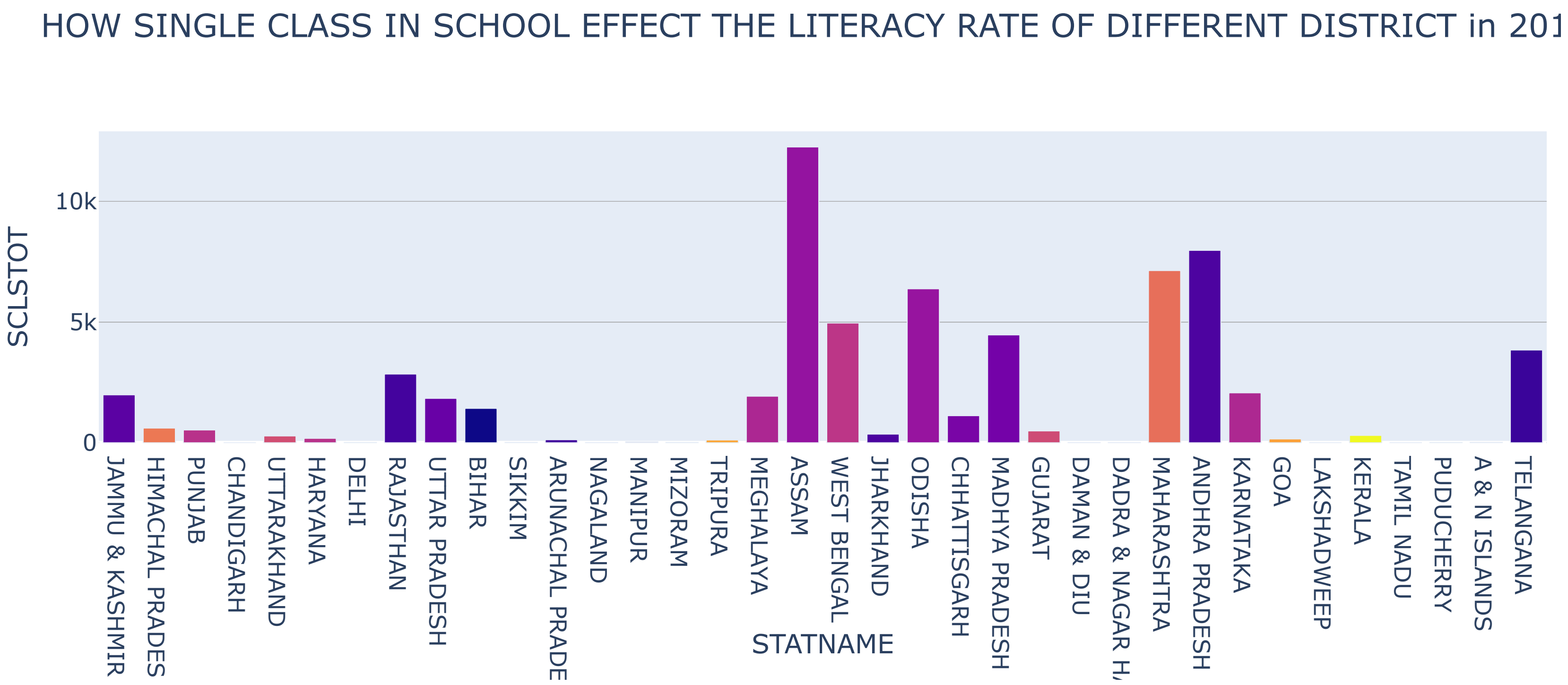
HOW SINGLE CLASS IN SCHOOL EFFECT THE LITERACY RATE OF DIFFERENT DISTRICT in 200

In [113]:

```
#READ 2016-17
data16_17 = pd.read_csv("data processed/2016-17_Elementary.csv")
data16_17.head()
enrol_per = (data16_17['ENRTOT']/(data16_17['TOTPOPULAT']*1000))*100
df16_17 = pd.concat([data16_17, enrol_per], axis=1, sort=False)
df16_17 = df16_17.rename(columns = {0:"enrolment_score"})
```

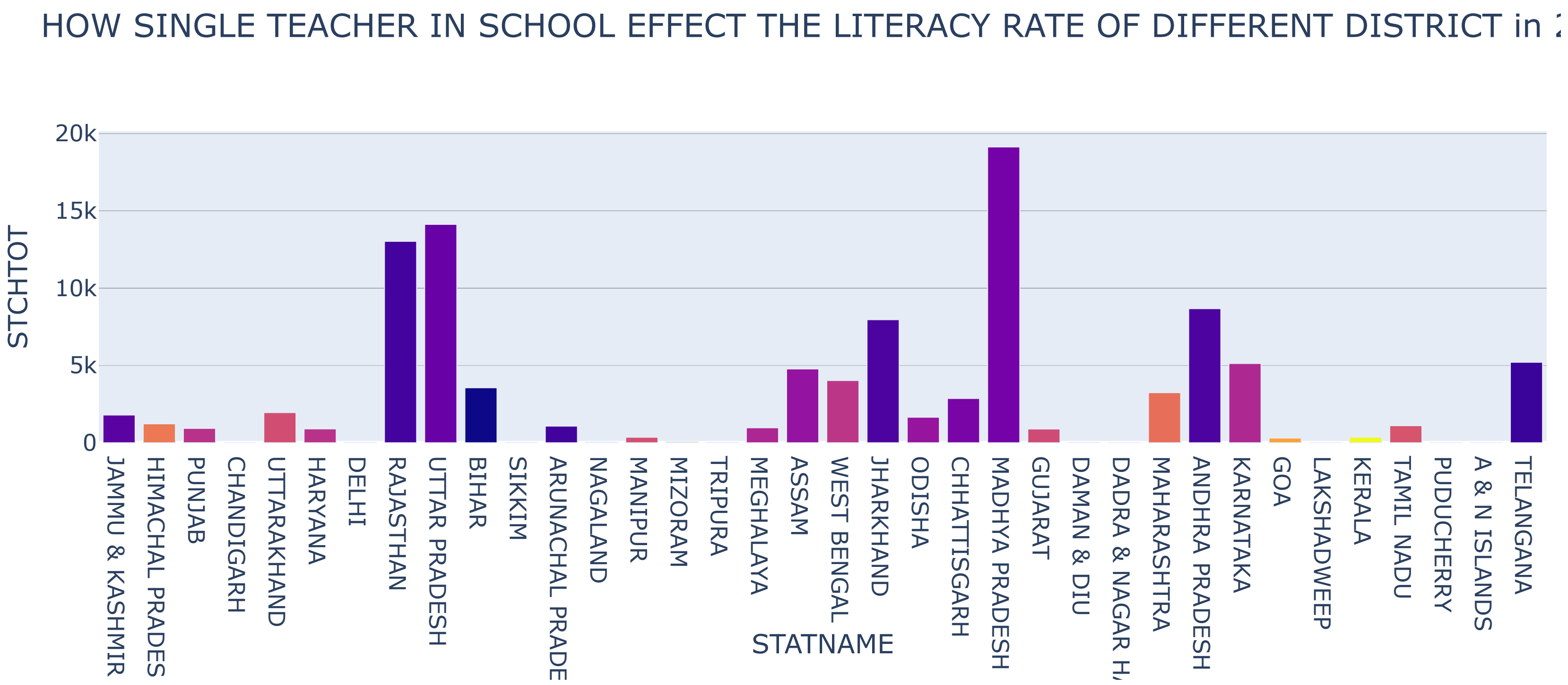
In [114]:

```
fig = px.bar(data16_17, x='STATNAME', y='SCLSTOT',color='OVERALL_LI', title="HOW SINGLE CLASS IN SCHOOL EFF
fig.show()
```



In [115]:

```
fig = px.bar(data16_17, x='STATNAME', y='STCHTOT',color='OVERALL_LI', title="HOW SINGLE TEACHER IN SCHOOL E
fig.show()
```



Observations by SINGLE ROOM/TEACHER IN SCHOOL EFFECT THE LITERACY RATE OF DIFFERENT DISTRICT in 2006-07 and 2016-17:

The literacy rate is generally poor for a single class or single teacher schools.

It is clear that the literacy rate decreases if we keep increasing the number of a single class or single teacher schools.

Single class and single teacher schools of Maharashtra have a decent literacy rate. It should be worked out why they are working so well in Maharashtra and then applied to other states as well.

In [116]:

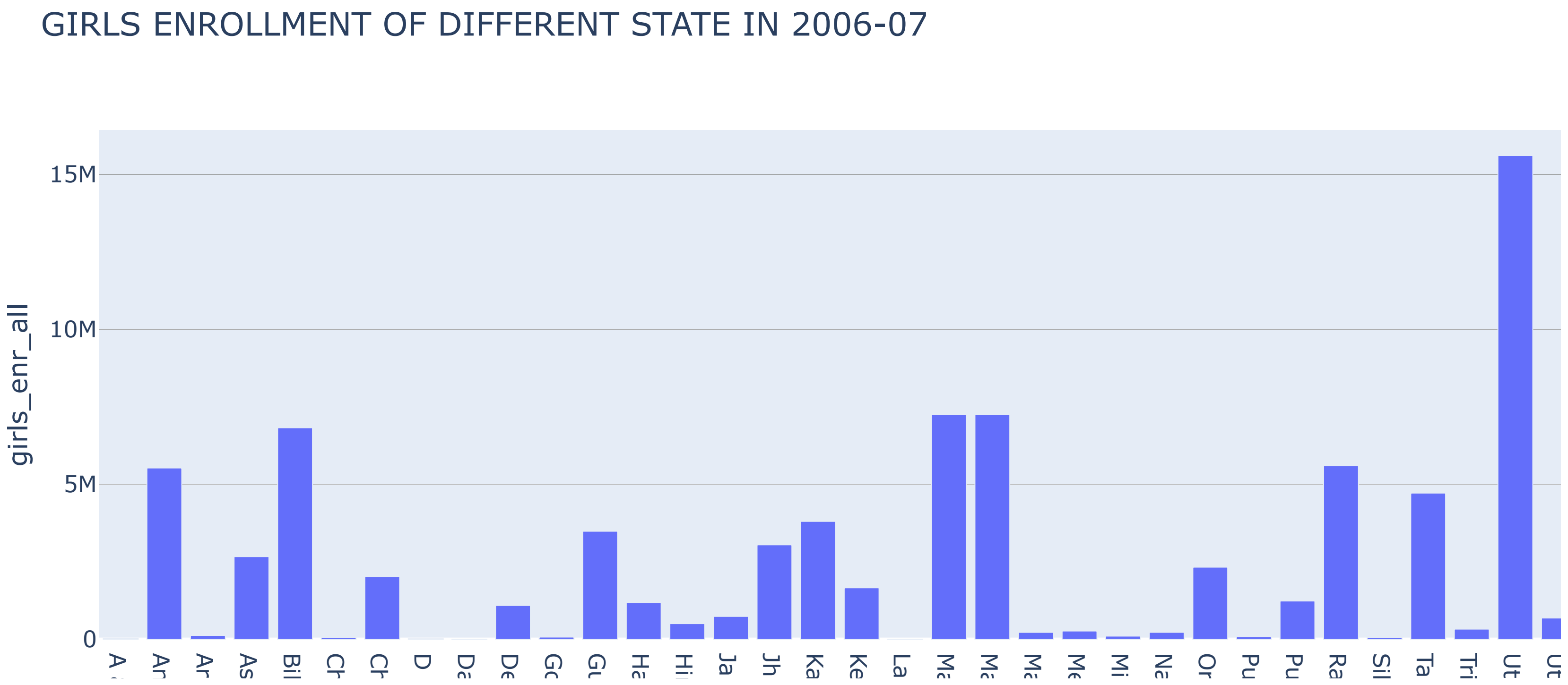
```
total_enrol = ['govt_enr_2','govt_enr_3','govt_enr_4','govt_enr_5','govt_enr_9','pvt_enr_1','pvt_enr_2','pv
```

In [117]:

```
val = df06_07['govt_enr_1']
for i in total_enrol:
    df06_07['govt_enr_1'] += df06_07[i]
val = pd.DataFrame(val)
```

In [118]:

```
# girls_enr_all
fig = px.bar(df06_07, x='statname', y='girls_enr_all',title="GIRLS ENROLLMENT OF DIFFERENT STATE IN 2006-07
fig.show()
```



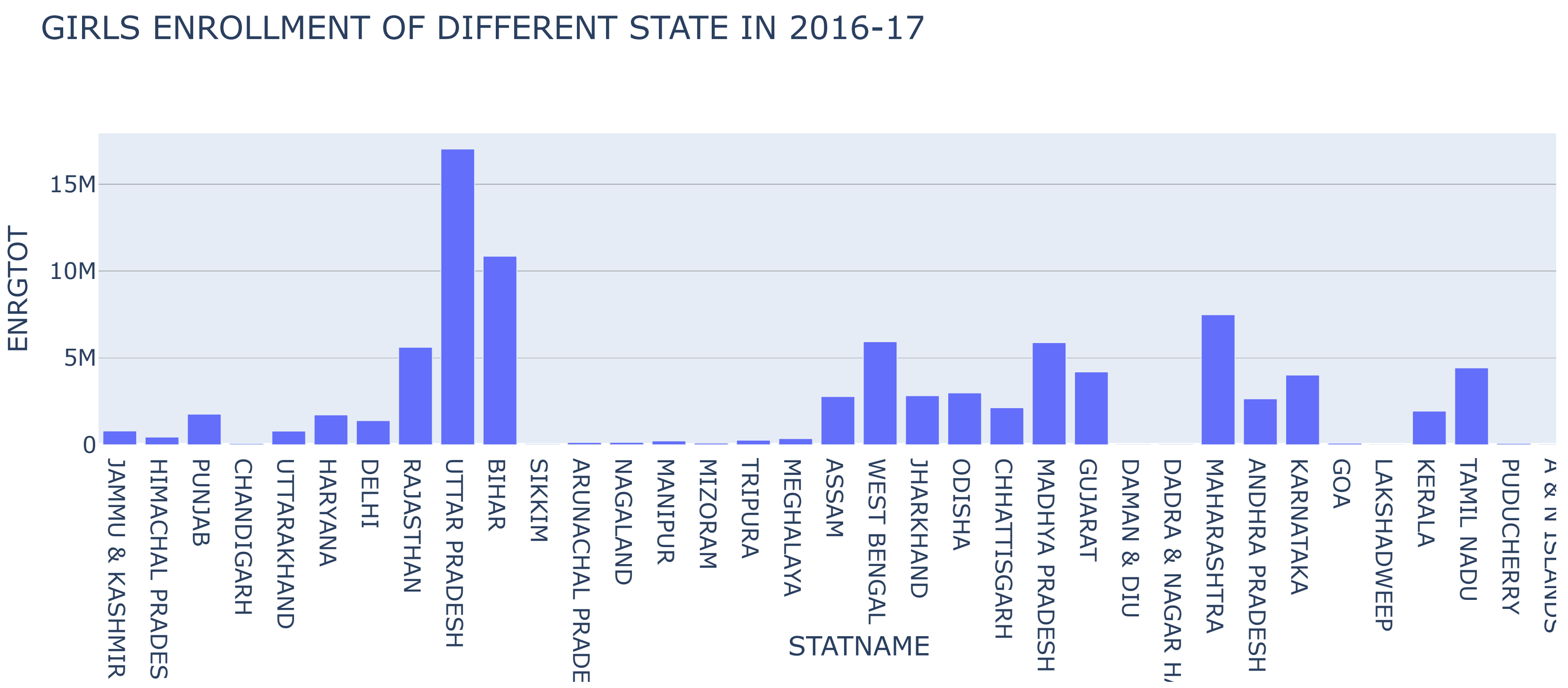
GIRLS ENROLLMENT OF DIFFERENT STATE IN 2006-07

In [119]:

```
data16_17['ENRTOTG'] += data16_17['ENRTOTP']
```

In [120]:

```
# girls_enr_all
fig = px.bar(data16_17, x='STATNAME', y='ENRGTOT',title="GIRLS ENROLLMENT OF DIFFERENT STATE IN 2016-17", h
fig.show()
```



In [ ]:

In [ ]: