MANOJ.S

2018 Karnataka (India) State Election Results

Elections were held in Karnataka on 12 May 2018 in 222 constituencies of the Karnataka Legislative Assembly. This dataset provides the outcomes of the election.

In [1]:

```
#importing the libraries and data
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from datetime import date, timedelta, datetime
```

In [2]:

```
df = pd.read_csv('Karnataka Assembly Elections 2018 All State MLA Winners List  OpenCity.
csv')
df
```

Out[2]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Win Ma
0	1	Nippani	BJP	SHASHIKALA JOLLE	87,006	INC	KAKASO PANDURANG PATIL	78,500	81.19%	211,827	8
1	2	Chikkodi-Sadalga	INC	GANESH HUKKERI	91,467	BJP	ANNASAHEB SHANKAR JOLLE	80,898	84.78%	210,480	10
2	3	Athani	INC	MAHESH IRANAGOUDA KUMATHALLI	82,094	BJP	LAXMAN SANGAPPA SAVADI	79,763	80.67%	213,935	2
3	4	Kagwad	INC	BALASAHEB PATIL	83,060	BJP	BHARAMAGOUDA ALAGOUDA KAGE	50,118	79.98%	181,486	32
4	5	Kudachi	BJP	P RAJIV	67,781	INC	AMIT SHAMA GHATAGE	52,773	75.86%	180,233	15
...	...	...	...	...	...	...	...	...	...	...	
219	220	T. Narasipur	JDS	ASHWIN KUMAR	83,929	INC	DR. H.C.MAHADEVAPPA	55,451	78.08%	198,434	28
220	221	Hanur	INC	R NARENDRA	60,444	BJP	DR. PREETHAN NAGAPPA	56,931	81.61%	207,603	3
221	222	Kollegal	OTHERS	N MAHESH	71,792	INC	A.R. KRISHNA MURTHY	52,338	79.15%	211,522	19
222	223	Chamarajanagar	INC	PUTTARANGA SHETTY	75,963	BJP	K R MALLIKARJUNAPPA	71,050	80.41%	206,146	4
223	224	Gundlupet	BJP	C S NIRANJANKUMAR	94,151	INC	M C MOHAN KUMARI URUF GEETHA	77,467	87.50%	205,616	16

224 rows × 12 columns

In [3]:

```
df.shape
```

Out[3]:

```
(224, 12)
```

In [4]:

```
df.head()
```

Out[4]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Winning Margin	District Name
0	1	Nippani	BJP	SHASHIKALA JOLLE	87,006	INC	KAKASO PANDURANG PATIL	78,500	81.19%	211,827	8,506	BELAGAVI
1	2	Chikkodi-Sadalga	INC	GANESH HUKKERI	91,467	BJP	ANNASAHEB SHANKAR JOLLE	80,898	84.78%	210,480	10,569	BELAGAVI
2	3	Athani	INC	MAHESH IRANAGOUDA KUMATHALLI	82,094	BJP	LAXMAN SANGAPPA SAVADI	79,763	80.67%	213,935	2,331	BELAGAVI
3	4	Kagwad	INC	BALASAHEB PATIL	83,060	BJP	BHARAMAGOUDA ALAGOUDA KAGE	50,118	79.98%	181,486	32,942	BELAGAVI
4	5	Kudachi	BJP	P RAJIV	67,781	INC	AMIT SHAMA GHATAGE	52,773	75.86%	180,233	15,008	BELAGAVI

In [5]:

```
df.columns
```

Out[5]:

```
Index(['AC No.', 'AC Name', 'Winning Party', 'Winning Candidate',
       'Winner Votes', 'Runner Up Party', 'Runner Up Candidate',
       'Runner Up Votes', 'Voting Turnout %', 'Total Voters', 'Winning Margin',
       'District Name'],
      dtype='object')
```

In [6]:

```
df.tail()
```

Out[6]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Win Ma
219	220	T. Narasipur	JDS	ASHWIN KUMAR	83,929	INC	DR. H.C.MAHADEVAPPA	55,451	78.08%	198,434	28
220	221	Hanur	INC	R NARENDRA	60,444	BJP	DR. PREETHAN NAGAPPA	56,931	81.61%	207,603	3
221	222	Kollegal	OTHERS	N MAHESH	71,792	INC	A.R. KRISHNA MURTHY	52,338	79.15%	211,522	19
222	223	Chamarajanagar	INC	PUTTARANGA SHETTY	75,963	BJP	K R MALLIKARJUNAPPA	71,050	80.41%	206,146	4
223	224	Gundlupet	BJP	C S NIRANJANKUMAR	94,151	INC	M C MOHAN KUMARI URUF GEETHA	77,467	87.50%	205,616	16

In [7]:

```
np.random.seed(42)
```

```
obs, feat = df.shape
df.sample(5)
```

Out[7]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Winning Margin	
9	10	Yemkanamardi	INC	SATISH. L. JARKIHOLI	73,512	BJP	ASTAGI MARUTI MALLAPPA	70,662	79.79%	186,859	2,850	
84	85	Byadgi	BJP	BALLARY VIRUPAKSHAPPA RUDRAPPA	91,721	INC	S.R.PATIL	70,450	82.57%	200,760	21,271	
117	118	Baindur	BJP	B. M. SUKUMAR SHETTY	96,029	INC	K. GOPALA POOJARY	71,636	78.93%	222,427	24,393	
144	145	Mulbagal	OTHERS	H.NAGESH	74,213	JDS	SAMRUDDHI MANJUNATH	67,498	79.96%	203,376	6,715	
221	222	Kollegal	OTHERS	N MAHESH	71,792	INC	A.R. KRISHNA MURTHY	52,338	79.15%	211,522	19,454	CH.

In [8]:

```
#calculating missing values in rows
df.isnull().sum()
```

Out[8]:

```
AC No.                 0
AC Name                0
Winning Party          0
Winning Candidate      0
Winner Votes           0
Runner Up Party        0
Runner Up Candidate    0
Runner Up Votes        0
Voting Turnout %       0
Total Voters           1
Winning Margin         0
District Name          0
dtype: int64
```

In [9]:

```
Totals = pd.read_csv('Karnataka Assembly Elections 2018 All State MLA Winners List  OpenC
ity.csv')
Totals.sample(5)
```

Out[9]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Winning Margin	
69	70	Kundgol	INC	C S SHIVALLI	64,871	BJP	CHIKKANAGOUDRA SIDDANAGOUD ISHWARAGOD	64,237	78.27%	187,513	634	
30	31	Nagthan	JDS	DEVANAND FULASING CHAVAN	59,709	INC	KATAKADOND VITTAL DONDIBA	54,108	62.95%	260,382	5,601	
39	40	Chittapur	INC	PRIYANK KHARIGE	69,700	BJP	VALMIK NAIK	65,307	60.34%	231,920	4,393	
221	222	Kollegal	OTHERS	N MAHESH	71,792	INC	A.R. KRISHNA MURTHY	52,338	79.15%	211,522	19,454	CHA
123	124	Mudigere	BJP	M P	58,783	INC	MOTAMMA	46,271	76.79%	170,250	12,512	C

KUMARSWAMY Runner Runner Voting

AC Winning Winning Winner Runner Up Total Winning

In [10]:

```
df.describe()
```

Out[10]:

	AC No.
count	224.000000
mean	112.500000
std	64.807407
min	1.000000
25%	56.750000
50%	112.500000
75%	168.250000
max	224.000000

In [11]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 224 entries, 0 to 223
Data columns (total 12 columns):
 #    Column               Non-Null Count  Dtype
---   ------               --------------  -----
 0    AC No.               224 non-null    int64
 1    AC Name              224 non-null    object
 2    Winning Party        224 non-null    object
 3    Winning Candidate    224 non-null    object
 4    Winner Votes         224 non-null    object
 5    Runner Up Party      224 non-null    object
 6    Runner Up Candidate  224 non-null    object
 7    Runner Up Votes      224 non-null    object
 8    Voting Turnout %     224 non-null    object
 9    Total Voters         223 non-null    object
 10   Winning Margin       224 non-null    object
 11   District Name        224 non-null    object
dtypes: int64(1), object(11)
memory usage: 21.1+ KB
```

In [12]:

```
df.dtypes
```

Out[12]:

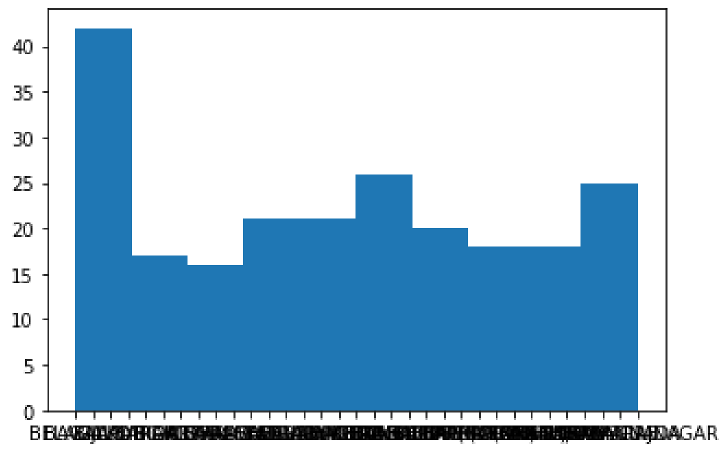
```
AC No.                  int64
AC Name                object
Winning Party          object
Winning Candidate      object
Winner Votes           object
Runner Up Party        object
Runner Up Candidate    object
Runner Up Votes        object
Voting Turnout %       object
Total Voters           object
Winning Margin         object
District Name          object
dtype: object
```

In [13]:

```
plt.hist(df['District Name'])
```

Out[13]:

```
(array([42., 17., 16., 21., 21., 26., 20., 18., 18., 25.]),
 array([ 0. ,  3.2,  6.4,  9.6, 12.8, 16. , 19.2, 22.4, 25.6, 28.8, 32. ]),
 <BarContainer object of 10 artists>)
```



In [14]:

```
df.describe(include=['object'])
```

Out[14]:

	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Winning Margin	District Name
count	224	224	224	224	224	224	224	224	223	224	224
unique	223	5	221	223	9	223	222	214	223	222	33
top	Vijayanagar	BJP	M KRISHNAPPA	-	INC	-	59,335	72.34%	211,827	21,271	BELAGAVI
freq	2	104	2	2	112	2	2	2	1	2	18

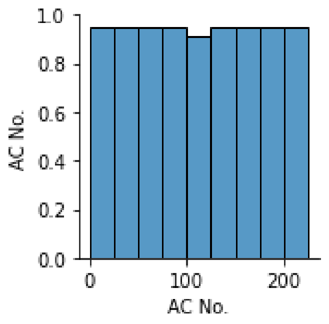
In [15]:

```
# Visulaizing the Pairplot of complete dataset

sns.pairplot(df)
```

Out[15]:

```
<seaborn.axisgrid.PairGrid at 0x21a30331b20>
```

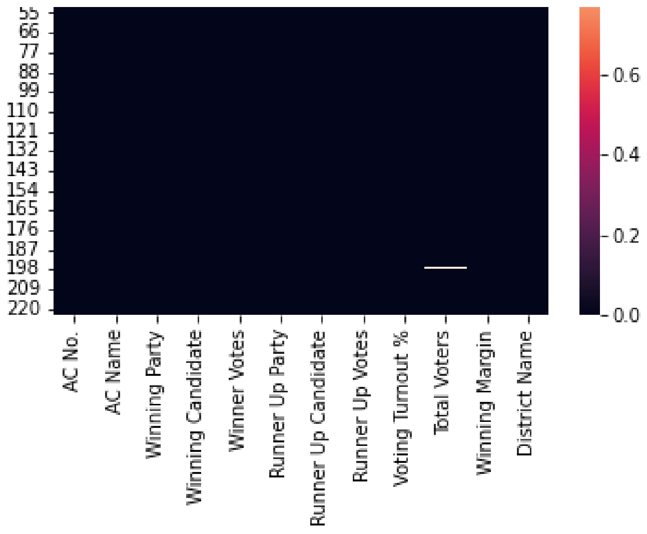


In [16]:

```
sns.heatmap(df.isna())
```

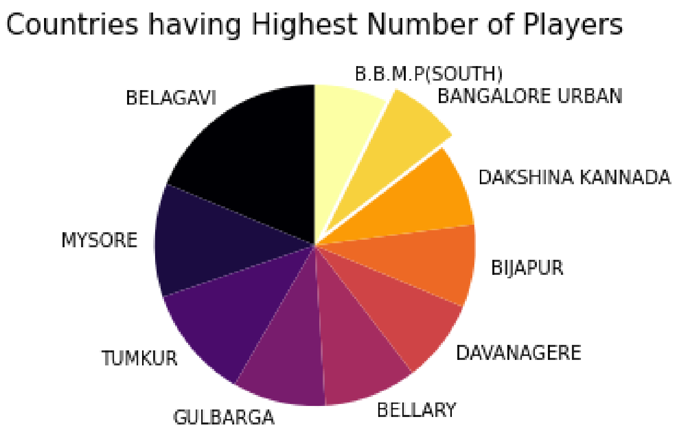
Out[16]:

```
<AxesSubplot:>
```



In [17]:

```
df['District Name'].value_counts().head(10).plot(kind='pie',cmap='inferno',startangle=90
,explode=[0,0,0,0,0,0,0,0,0.1,0])
plt.title('Countries having Highest Number of Players',fontsize=15)
plt.axis('off')
plt.show()
```



data cleaning

In [18]:

```
#1.missing value treatment
df.isnull().sum()
```

Out[18]:

```
AC No.                 0
AC Name                0
Winning Party          0
Winning Candidate      0
Winner Votes           0
Runner Up Party        0
Runner Up Candidate    0
Runner Up Votes        0
Voting Turnout %       0
Total Voters           1
Winning Margin         0
District Name          0
dtype: int64
```

In [19]:

```
df['Total Voters'].dtype
```

Out[19]:

```
dtype('O')
```

In [20]:

```
def removecomma(values):
    values=str(values)
    values=int(values.replace(',',''))
    return values
```

In [21]:

```
def handlecomma(value):
    value = str(value)
    if ',' in value:
        value = value.replace(',', '')
        return float(value)
    else:
        return float(value)
```

In [22]:

```
df['Winner Votes'].unique()
```

Out[22]:

```
array(['87,006', '91,467', '82,094', '83,060', '67,781', '67,502',
       '83,588', '96,144', '90,249', '73,512', '79,060', '84,498',
       '102,040', '36,649', '73,155', '47,040', '62,480', '68,349',
       '76,431', '87,213', '49,245', '85,135', '67,599', '85,653',
       '65,012', '63,512', '48,245', '58,647', '98,339', '76,308',
       '59,709', '50,401', '70,865', '71,735', '68,508', '104,426',
       '78,642', '62,227', '79,627', '69,700', '80,668', '73,905',
       '61,750', '64,788', '64,311', '76,815', '61,425', '74,945',
       '55,107', '73,270', '84,673', '75,061', '66,656', '56,511',
       '53,548', '67,003', '54,230', '71,514', '60,387', '87,567',
       '87,735', '67,617', '79,072', '98,783', '91,967', '77,699',
       '83,735', '73,045', '65,718', '64,871', '85,123', '77,080',
       '75,794', '96,462', '83,267', '61,577', '60,339', '59,392',
       '83,172', '70,595', '66,290', '80,529', '83,868', '86,565',
       '91,721', '72,461', '63,910', '54,097', '78,337', '83,214',
       '80,592', '82,546', '79,186', '76,589', '78,106', '50,085',
       '84,018', '72,874', '82,896', '77,733', '90,562', '107,976',
       '78,948', '67,603', '64,801', '76,540', '71,369', '50,556',
       '73,794', '80,624', '69,326', '75,722', '104,027', '67,527',
       '86,983', '72,091', '78,475', '96,029', '103,434', '84,946',
       '75,893', '91,245', '62,780', '58,783', '70,863', '44,940',
       '62,232', '69,612', '61,383', '60,710', '58,697', '60,421',
       '82,740', '81,598', '55,572', '74,338', '72,974', '88,521',
       '69,000', '65,710', '82,006', '76,240', '87,753', '93,571',
       '74,213', '71,151', '70,871', '82,788', '75,677', '120,110',
       '135,404', '114,964', '115,273', '-', '94,044', '88,218', '83,130',
       '74,453', '97,574', '109,955', '58,887', '59,742', '60,009',
       '47,354', '56,271', '79,135', '73,353', '65,339', '57,312',
       '76,018', '77,868', '67,085', '141,682', '111,863', '152,469',
       '113,894', '98,824', '86,966', '73,225', '69,277', '119,492',
       '92,626', '127,552', '87,995', '103,038', '109,239', '96,003',
       '69,421', '101,307', '112,396', '88,016', '105,516', '93,986',
       '64,268', '63,348', '108,541', '85,064', '62,262', '98,417',
       '87,444', '98,648', '86,545', '80,813', '97,802', '90,073',
       '95,205', '70,631', '77,944', '77,770', '85,011', '91,667',
       '76,652', '78,030', '121,325', '78,573', '51,683', '62,268',
       '96,435', '83,929', '60,444', '71,792', '75,963', '94,151'],
      dtype=object)
```

In [23]:

```
def handleline(value):
    value = str(value)
    if ',' or '-' or '%' in value:
        value = value.replace(',','')
        value = value.replace('-', '3000')
        value = value.replace('%','')
        return float(value)
    else:
        return float(value)
```

In [24]:

```
df['Total Voters']=df['Total Voters'].apply(handlecomma)
```

In [25]:

```
df['Winner Votes']=df['Winner Votes'].apply(handleline)
```

In [26]:

```
df['Runner Up Votes']=df['Runner Up Votes'].apply(handleline)
```

In [27]:

```
df['Voting Turnout %']=df['Voting Turnout %'].apply(handleline)
```

In [28]:

```
df['Winning Margin']=df['Winning Margin'].apply(handleline)
```

In [29]:

```
df.head()
```

Out[29]:

	AC No.	AC Name	Winning Party	Winning Candidate	Winner Votes	Runner Up Party	Runner Up Candidate	Runner Up Votes	Voting Turnout %	Total Voters	Winning Margin	Distr Nar
0	1	Nippani	BJP	SHASHIKALA JOLLE	87006.0	INC	KAKASO PANDURANG PATIL	78500.0	81.19	211827.0	8506.0	BELAGA
1	2	Chikkodi-Sadalga	INC	GANESH HUKKERI	91467.0	BJP	ANNASAHEB SHANKAR JOLLE	80898.0	84.78	210480.0	10569.0	BELAGA
2	3	Athani	INC	MAHESH IRANAGOUDA KUMATHALLI	82094.0	BJP	LAXMAN SANGAPPA SAVADI	79763.0	80.67	213935.0	2331.0	BELAGA
3	4	Kagwad	INC	BALASAHEB PATIL	83060.0	BJP	BHARAMAGOUDA ALAGOUDA KAGE	50118.0	79.98	181486.0	32942.0	BELAGA
4	5	Kudachi	BJP	P RAJIV	67781.0	INC	AMIT SHAMA GHATAGE	52773.0	75.86	180233.0	15008.0	BELAGA

In [30]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 224 entries, 0 to 223
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   AC No.                224 non-null    int64  
 1   AC Name               224 non-null    object 
 2   Winning Party         224 non-null    object 
 3   Winning Candidate     224 non-null    object 
 4   Winner Votes          224 non-null    float64
 5   Runner Up Party       224 non-null    object 
 6   Runner Up Candidate   224 non-null    object 
 7   Runner Up Votes       224 non-null    float64
 8   Voting Turnout %      224 non-null    float64
 9   Total Voters          223 non-null    float64
 10  Winning Margin        224 non-null    float64
 11  District Name        224 non-null    object
dtypes: float64(5), int64(1), object(6)
memory usage: 21.1+ KB
```

In [ ]:

```

```

In [ ]:

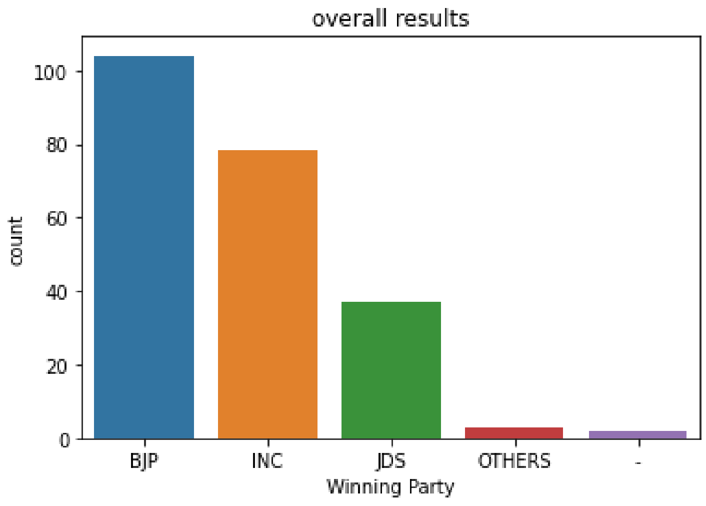
```

```

Data visualization

In [31]:

```python
sns.countplot(x='Winning Party',data=df)
plt.title('overall results')
plt.show()
```

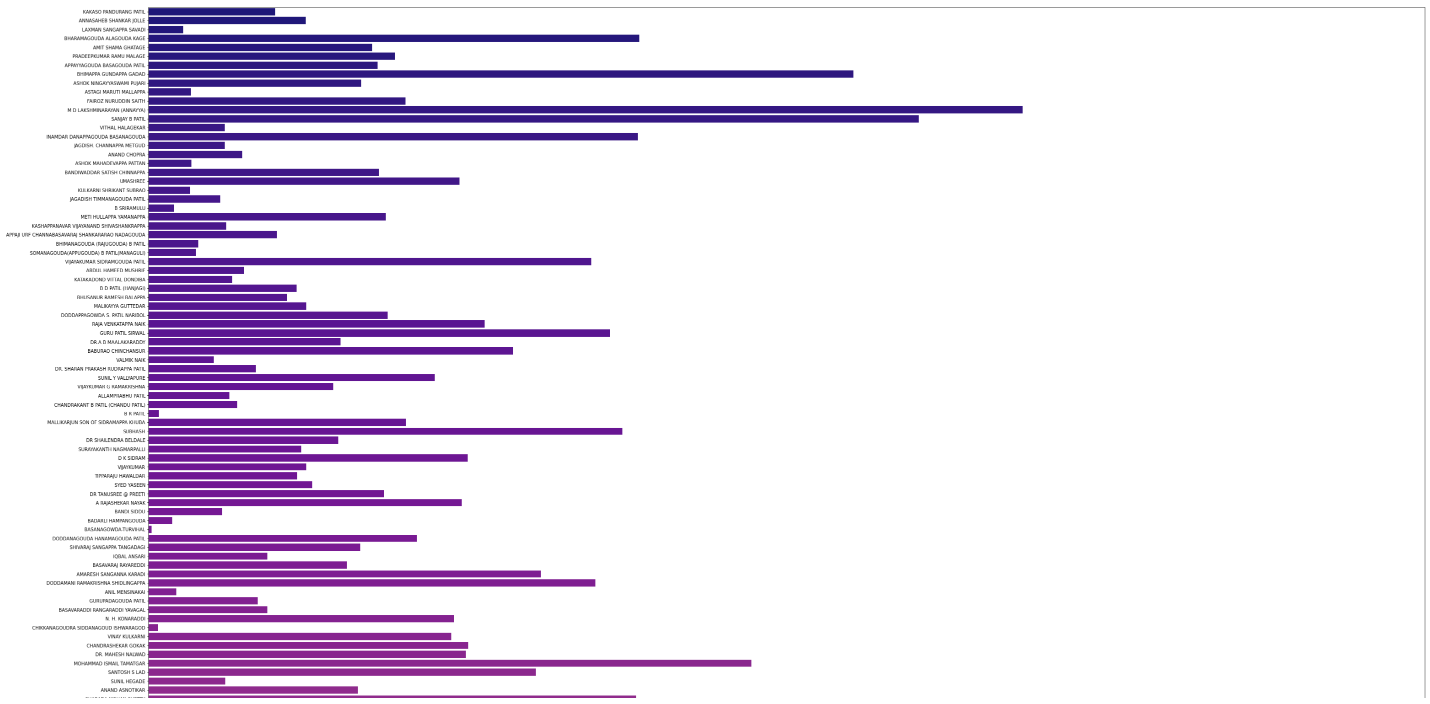


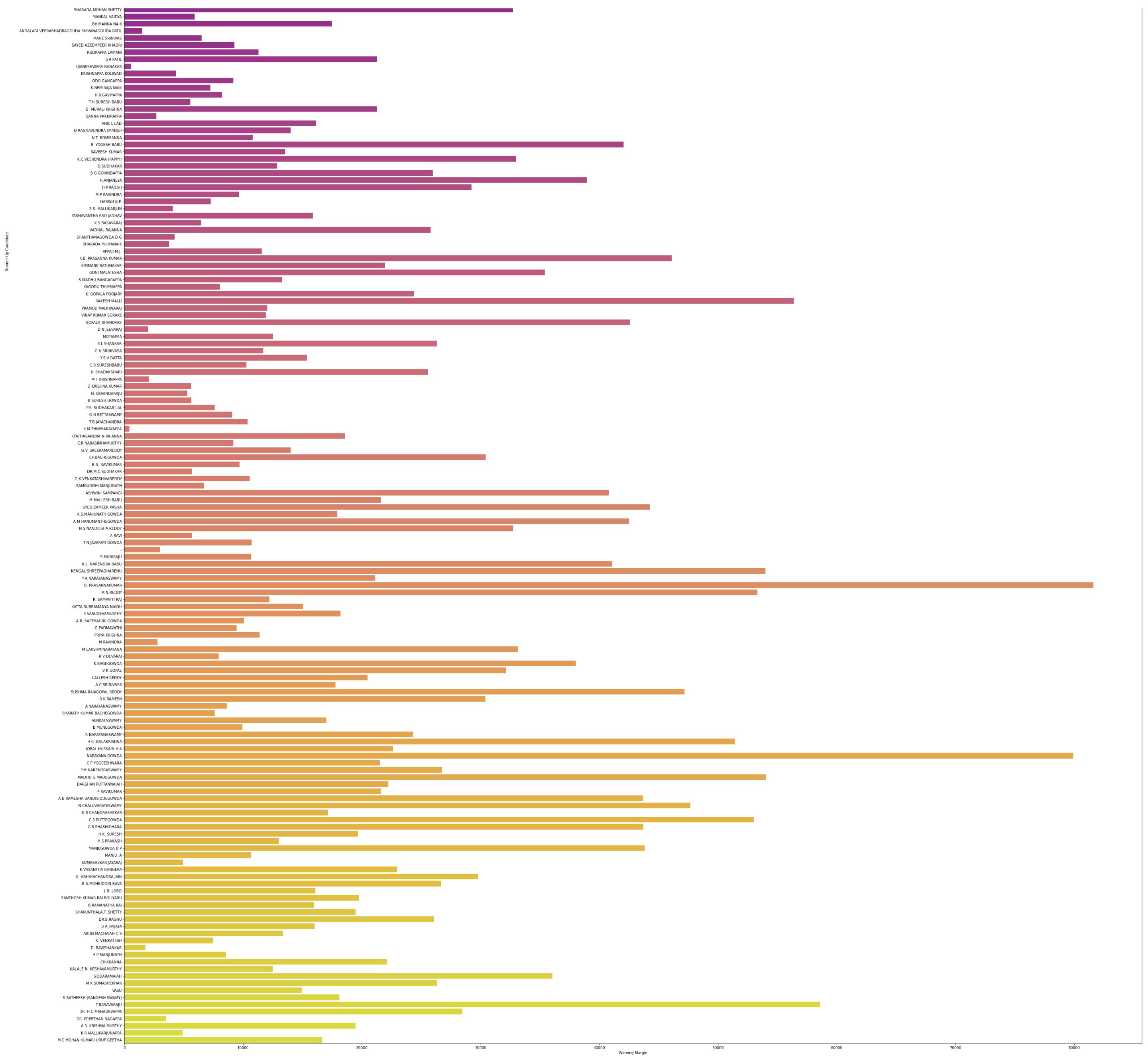
In [32]:

```python
plt.figure(figsize=(50,80))
sns.barplot(y ='Runner Up Candidate', x ='Winning Margin', data = df, palette ='plasma',
orient="h")
```

Out[32]:

```
<AxesSubplot:xlabel='Winning Margin', ylabel='Runner Up Candidate'>
```



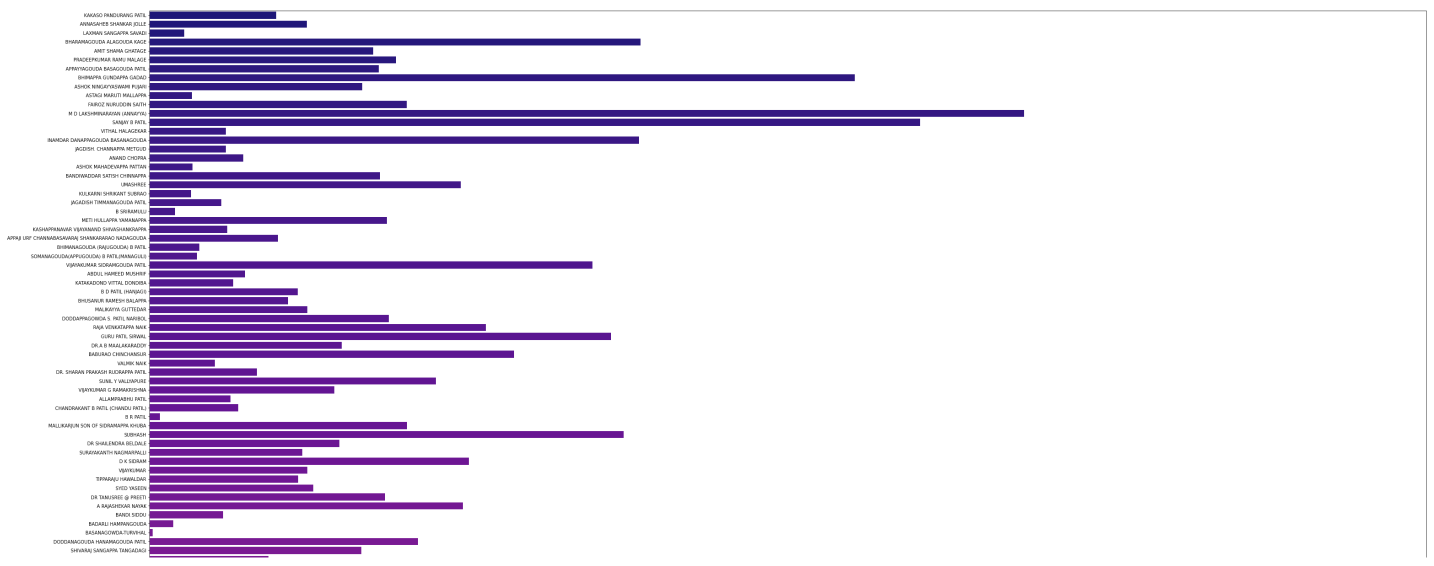


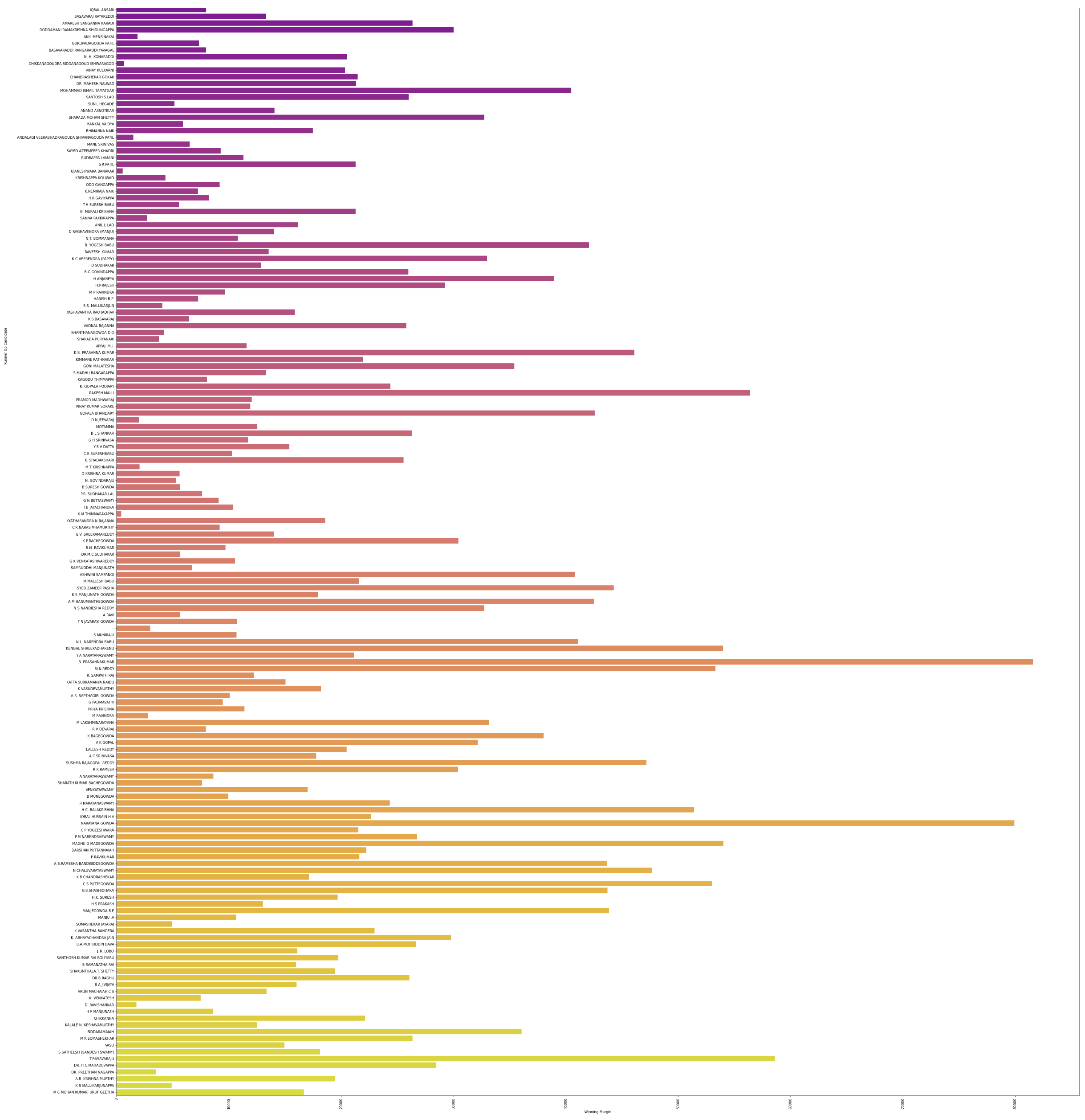
In [33]:

```
import matplotlib.pylab as plt
plt.figure(figsize=(50,80))
plt.xticks(rotation=90)
sns.barplot(y='Runner Up Candidate', x ='Winning Margin', data = df, palette ='plasma')
```

Out[33]:

```
<AxesSubplot:xlabel='Winning Margin', ylabel='Runner Up Candidate'>
```



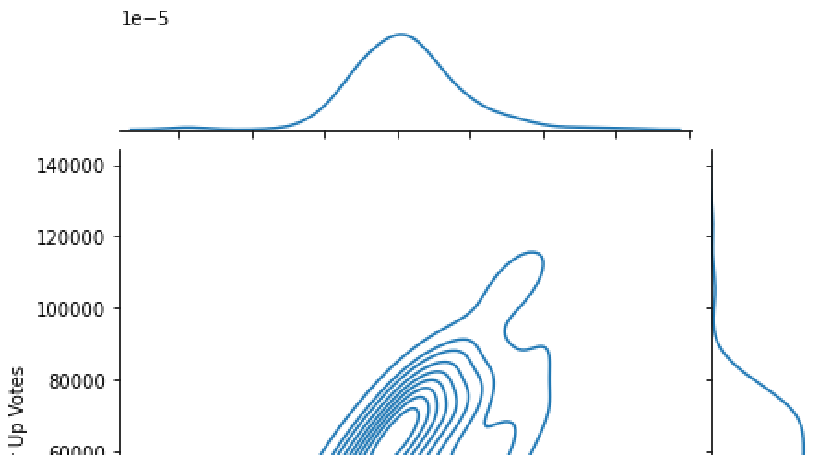


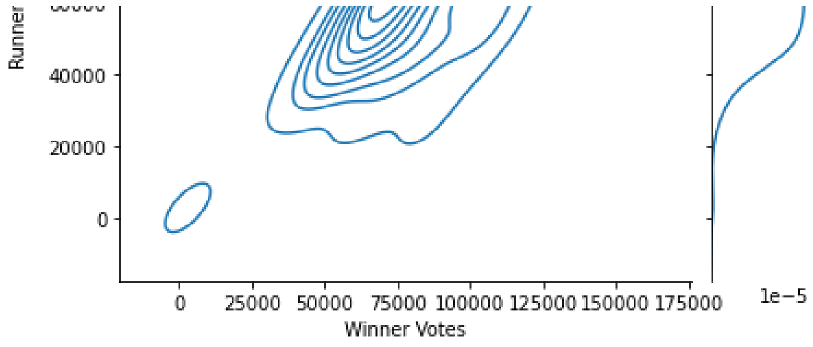
In [34]:

```
sns.jointplot(y ='Runner Up Votes', x ='Winner Votes', data = df, kind ='kde')
# KDE shows the density where the points match up the most
```

Out[34]:

```
<seaborn.axisgrid.JointGrid at 0x21a32254940>
```



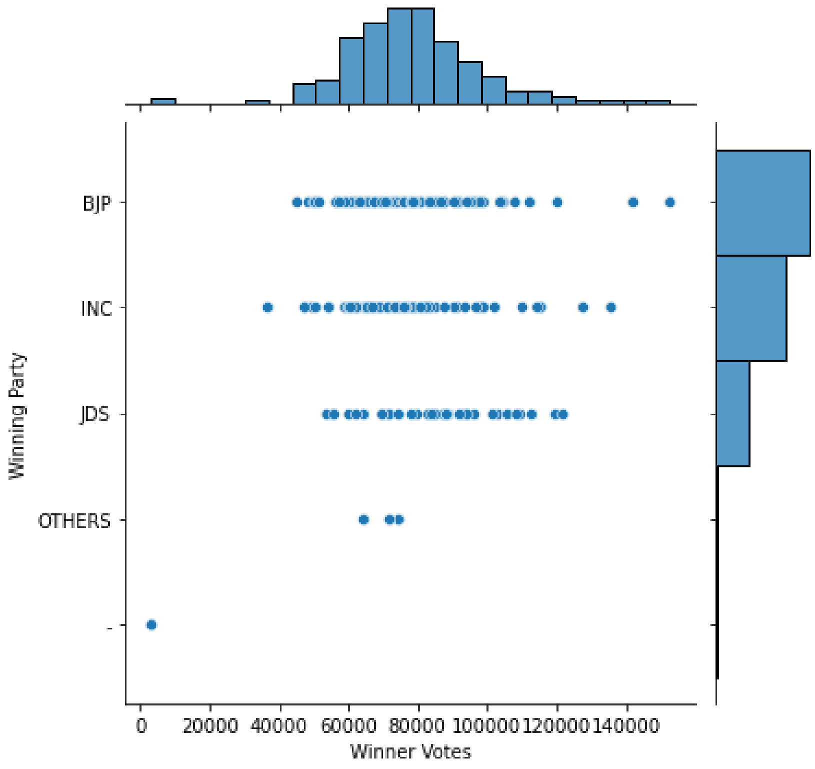


In [35]:

```
#Joinplot
#It is used to draw a plot of two variables with bivariate and univariate graphs. It basi
cally combines two different plots.
sns.jointplot(x ='Winner Votes', y ='Winning Party', data = df)
```

Out[35]:

```
<seaborn.axisgrid.JointGrid at 0x21a39918b20>
```



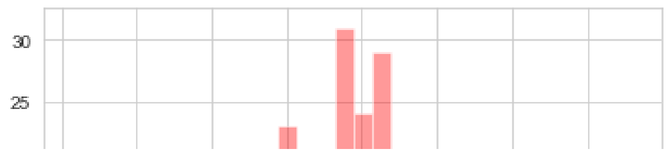
In [36]:

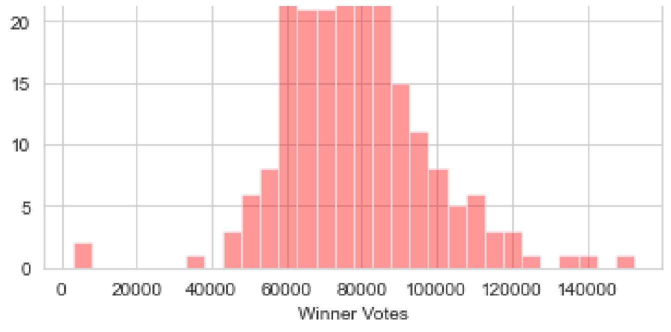
```
# set the background style of the plot
#Displot t is used basically for univariant set of observations and visualizes it through
a histogram i.e. only one observation and hence we choose one particular column of the da
taset.
sns.set_style('whitegrid')
sns.distplot(df['Winner Votes'], kde = False, color ='red', bins = 30)
```

```
C:\Users\Manoj\anaconda3\lib\site-packages\seaborn\distributions.py:2619: 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).
  warnings.warn(msg, FutureWarning)
```

Out[36]:

```
<AxesSubplot:xlabel='Winner Votes'>
```



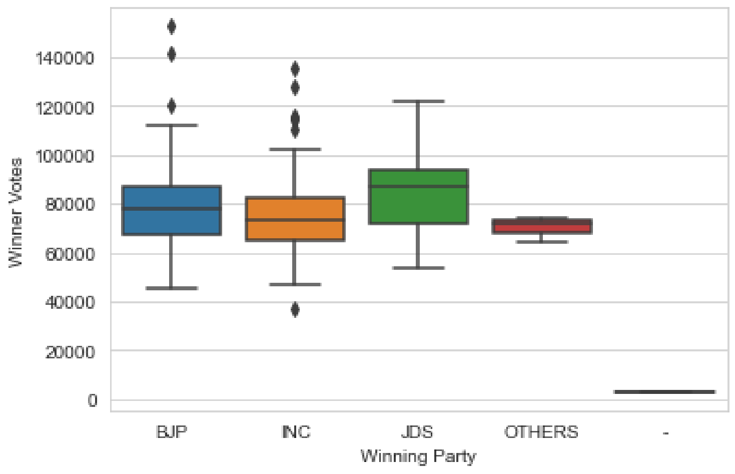


In [37]:

```
# x takes the categorical column and y is a numerical column
#box plot
sns.boxplot(x ='Winning Party', y ='Winner Votes', data = df)
```

Out[37]:

```
<AxesSubplot:xlabel='Winning Party', ylabel='Winner Votes'>
```

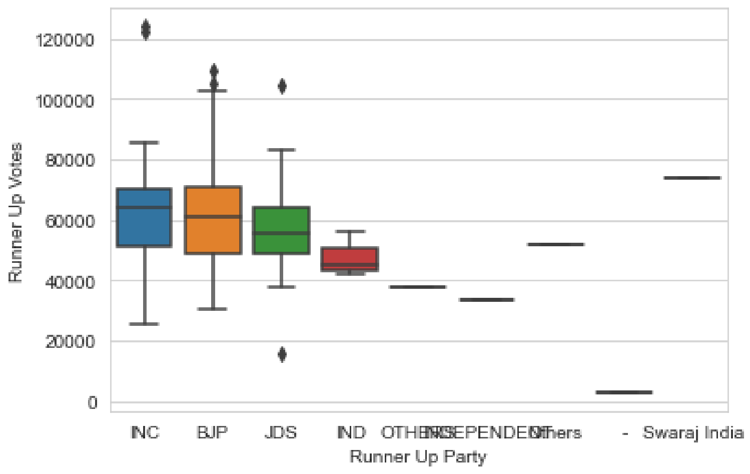


In [38]:

```
# x takes the categorical column and y is a numerical column
#box plot
sns.boxplot(x ='Runner Up Party', y ='Runner Up Votes', data = df)
```

Out[38]:

```
<AxesSubplot:xlabel='Runner Up Party', ylabel='Runner Up Votes'>
```

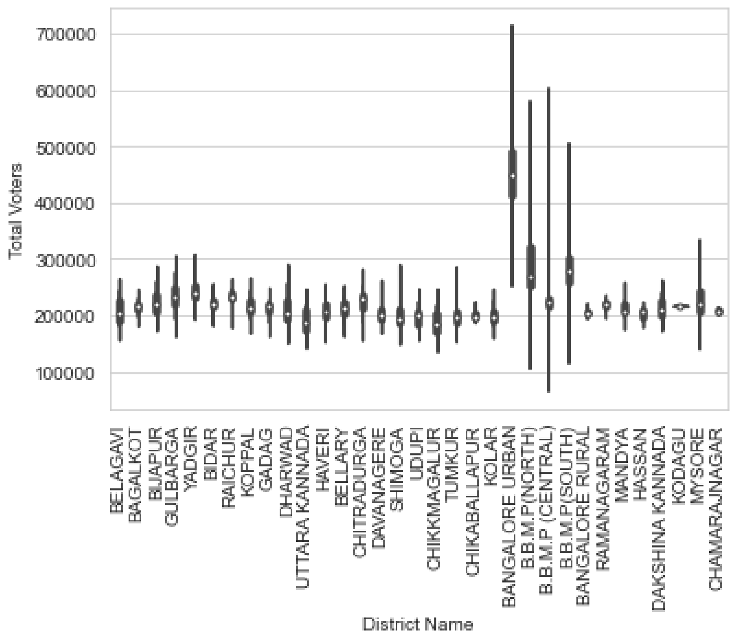


In [39]:

```
#Violinplot
#It is similar to the boxplot except that it provides a higher, more advanced visualizati
on and uses the kernel density
#estimation to give a better description about the data distribution.
sns.violinplot(x ='District Name', y ='Total Voters', data = df)
plt.xticks(rotation=90)
```

Out[39]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32]),
 [Text(0, 0, 'BELAGAVI'),
  Text(1, 0, 'BAGALKOT'),
  Text(2, 0, 'BIJAPUR'),
  Text(3, 0, 'GULBARGA'),
  Text(4, 0, 'YADGIR'),
  Text(5, 0, 'BIDAR'),
  Text(6, 0, 'RAICHUR'),
  Text(7, 0, 'KOPPAL'),
  Text(8, 0, 'GADAG'),
  Text(9, 0, 'DHARWAD'),
  Text(10, 0, 'UTTARA KANNADA'),
  Text(11, 0, 'HAVERI'),
  Text(12, 0, 'BELLARY'),
  Text(13, 0, 'CHITRADURGA'),
  Text(14, 0, 'DAVANAGERE'),
  Text(15, 0, 'SHIMOGA'),
  Text(16, 0, 'UDUPI'),
  Text(17, 0, 'CHIKKMAGALUR'),
  Text(18, 0, 'TUMKUR'),
  Text(19, 0, 'CHIKABALLAPUR'),
  Text(20, 0, 'KOLAR'),
  Text(21, 0, 'BANGALORE URBAN'),
  Text(22, 0, 'B.B.M.P(NORTH)'),
  Text(23, 0, 'B.B.M.P (CENTRAL)'),
  Text(24, 0, 'B.B.M.P(SOUTH)'),
  Text(25, 0, 'BANGALORE RURAL'),
  Text(26, 0, 'RAMANAGARAM'),
  Text(27, 0, 'MANDYA'),
  Text(28, 0, 'HASSAN'),
  Text(29, 0, 'DAKSHINA KANNADA'),
  Text(30, 0, 'KODAGU'),
  Text(31, 0, 'MYSORE'),
  Text(32, 0, 'CHAMARAJNAGAR')])
```

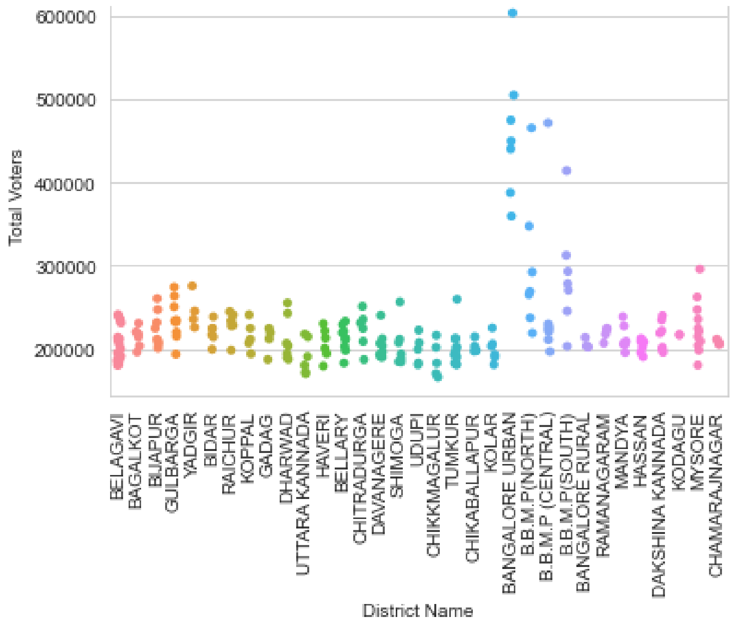


In [40]:

```python
#Stripplot
#It basically creates a scatter plot based on the category.
sns.stripplot(x ='District Name', y ='Total Voters', data = df,orient="v")
plt.xticks(rotation=90)
plt.show
```

Out[40]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```

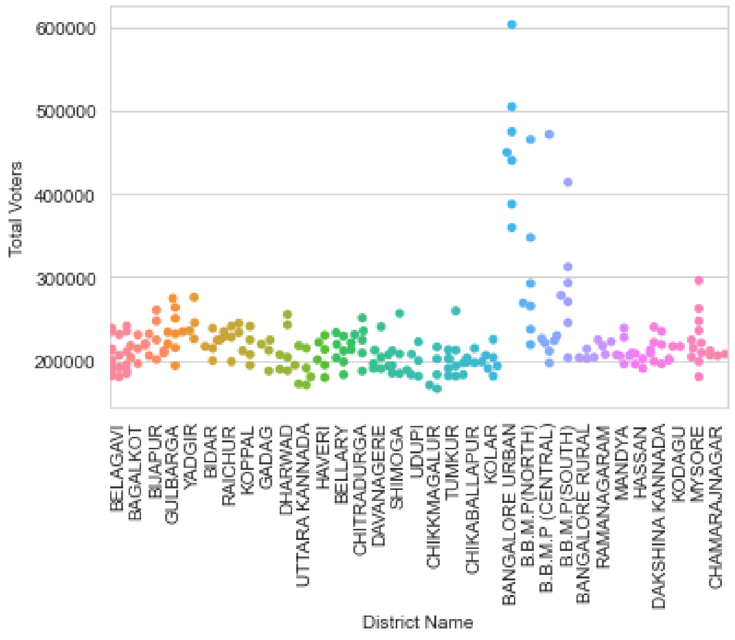


In [41]:

```
#Swarmplot
#It is very similar to the stripplot except the fact that the points are adjusted so that
they do not overlap.
#Some people also like combining the idea of a violin plot and a stripplot to form this p
lot.
#One drawback to using swarmplot is that sometimes they dont scale well to really large n
umbers and takes a lot of computation to arrange them.
#So in case we want to visualize a swarmplot properly we can plot it on top of a violinpl
ot.
import warnings
warnings.filterwarnings("ignore")
plt.xticks(rotation=90)
sns.swarmplot(x ='District Name', y ='Total Voters', data = df,orient="v")
```

Out[41]:

```
<AxesSubplot:xlabel='District Name', ylabel='Total Voters'>
```

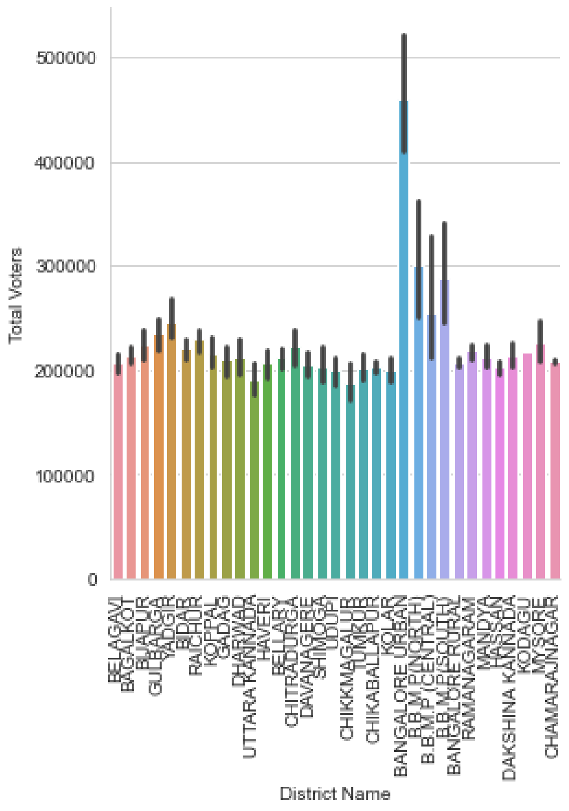


In [42]:

```
#Factorplot
#It is the most general of all these plots and provides a parameter called kind to choose
the kind of plot we want thus saving us from the trouble of writing these plots separatel
y.
#The kind parameter can be bar, violin, swarm etc.
sns.factorplot(x ='District Name', y ='Total Voters', data = df, kind ='bar',orient="v")
plt.xticks(rotation=90)
```

Out[42]:

```
(array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
        17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32]),
 [Text(0, 0, 'BELAGAVI'),
  Text(1, 0, 'BAGALKOT'),
  Text(2, 0, 'BIJAPUR'),
  Text(3, 0, 'GULBARGA'),
  Text(4, 0, 'YADGIR'),
  Text(5, 0, 'BIDAR'),
  Text(6, 0, 'RAICHUR'),
  Text(7, 0, 'KOPPAL'),
  Text(8, 0, 'GADAG'),
  Text(9, 0, 'DHARWAD'),
  Text(10, 0, 'UTTARA KANNADA'),
  Text(11, 0, 'HAVERI'),
  Text(12, 0, 'BELLARY'),
  Text(13, 0, 'CHITRADURGA'),
  Text(14, 0, 'DAVANAGERE'),
  Text(15, 0, 'SHIMOGA'),
  Text(16, 0, 'UDUPI'),
  Text(17, 0, 'CHIKKMAGALUR'),
  Text(18, 0, 'TUMKUR'),
  Text(19, 0, 'CHIKABALLAPUR'),
  Text(20, 0, 'KOLAR'),
  Text(21, 0, 'BANGALORE URBAN'),
  Text(22, 0, 'B.B.M.P(NORTH)'),
  Text(23, 0, 'B.B.M.P (CENTRAL)'),
  Text(24, 0, 'B.B.M.P(SOUTH)'),
  Text(25, 0, 'BANGALORE RURAL'),
  Text(26, 0, 'RAMANAGARAM'),
  Text(27, 0, 'MANDYA'),
  Text(28, 0, 'HASSAN'),
  Text(29, 0, 'DAKSHINA KANNADA'),
  Text(30, 0, 'KODAGU'),
  Text(31, 0, 'MYSORE'),
  Text(32, 0, 'CHAMARAJNAGAR')])
```


Total Voters
BELAGAVI
BAGALKOT
BIJAPUR
GULBARGA
YADGIR
BIDAR
RAICHUR
KOPPAL
GADAG
DHARWAD
UTTARA KANNADA
HAVERI
BELLARY
CHITRADURGA
DAVANAGERE
SHIMOGA
UDUPI
CHIKKMAGALUR
TUMKUR
CHIKABALLAPUR
KOLAR
BANGALORE URBAN
B.B.M.P(NORTH)
B.B.M.P (CENTRAL)
B.B.M.P(SOUTH)
BANGALORE RURAL
RAMANAGARAM
MANDYA
HASSAN
DAKSHINA KANNADA
KODAGU
MYSORE
CHAMARAJNAGAR
District Name