RAINFALL DATA CLEANING

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
rain = pd.read_excel('Rajasthan Rainfall 1901-2002 + 2004-2010.xlsx')
df = pd.DataFrame(rain)
df.head()
```

Out[2]:

	State	District	Year	January	February	March	April	May	June	July	August	September	October	November	December
0	Rajasthan	Ajmer	2004	2.3	0.0	0.0	0.0	13.4	21.1	120.4	282.5	15.4	11.6	0.0	0.0
1	Rajasthan	Ajmer	2005	0.2	2.9	9.5	28.1	2.4	57.0	148.1	79.9	170.6	0.0	0.0	0.0
2	Rajasthan	Ajmer	2006	0.0	0.0	0.7	0.0	29.0	48.3	101.7	219.4	43.4	0.1	0.0	0.0
3	Rajasthan	Ajmer	2007	0.4	10.6	4.5	0.5	0.9	47.1	172.9	92.6	46.3	0.0	0.0	1.8
4	Rajasthan	Ajmer	2008	0.0	0.0	0.4	5.7	15.4	89.7	86.4	189.7	85.9	19.1	0.0	0.0

IDENTIFYING MISSING YEARS AND ADDING 2011 TO 2019 DATA

In [3]:

```
df.Year.unique()
```

Out[3]:

```
array([2004, 2005, 2006, 2007, 2008, 2009, 2010, 1901, 1902, 1903, 1904,
       1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912, 1913, 1914, 1915,
       1916, 1917, 1918, 1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926,
       1927, 1928, 1929, 1930, 1931, 1932, 1933, 1934, 1935, 1936, 1937,
       1938, 1939, 1940, 1941, 1942, 1943, 1944, 1945, 1946, 1947, 1948,
       1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959,
       1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970,
       1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981,
       1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992,
       1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002],
      dtype=int64)
```

CLEANING DATA

In [5]:

```
df.District.value_counts()
```

Out[5]:

```
Barmer             109
Ajmer              109
Bharatpur          109
Jaisalmer          109
Jaipur             102
Baran              102
Chittaurgarh       102
Churu              102
Jalor              102
Sawai Madhopur     102
Pali               102
```

```
Sikar              102
Jhalawar           102
Bundi              102
Bikaner            102
Sirohi             102
Bhilwara           102
Alwar              102
Nagaur             102
Udaipur            102
Tonk               102
Ganganagar         102
Jodhpur            102
Rajsamand          102
Banswara           102
Hanumangarh        102
Dungarpur          102
Kota               102
Karauli            102
Jhunjhunun         102
Dhaulpur           102
Dausa              102
 Rajsamand           7
 Tonk                7
 Jhunjhunu           7
 Jodhpur             7
 Banswara            7
 Karauli             7
 Dholpur             7
 Sikar               7
 Jhalawar            7
 Jaipur              7
 Dungarpur           7
 Baran               7
 Jalore              7
 Sri Ganganagar      7
 Chittorgarh         7
 Pali                7
 Alwar               7
 Bundi               7
 Udaipur             7
 Sirohi              7
 Churu               7
 Hanumangarh         7
 Dausa               7
 Sawaimadhopur       7
 Bhilwara            7
 Bikaner             7
 Kota                7
 Nagaur              7
Name: District, dtype: int64
```

In [6]:

```
df.District.replace({' Jaipur': 'Jaipur', ' Kota': 'Kota', ' Bundi':'Bundi', ' Jalore':'Jalore', ' Karauli':'Karauli',
                     ' Rajsamand': 'Rajsamand', ' Tonk': 'Tonk', ' Dholpur':'Dholpur', ' Sri Ganganagar':'Ganganagar',
                    ' Sawaimadhopur': 'Sawai Madhopur', ' Pali': 'Pali', ' Jodhpur':'Jodhpur', ' Bikaner':'Bikaner', ' Bhilwara':'Bhilwara', ' Alwar':'Alwar',
                     ' Baran':'Baran', ' Sirohi':'Sirohi', ' Chittorgarh':'Chittorgarh', 'Chittaurgarh':'Chittorgarh',
                     ' Udaipur':'Udaipur', ' Hanumangarh':'Hanumangarh', ' Jhalawar':'Jhalawar', 'Dausa': 'Dausa',
                     ' Nagaur': 'Nagaur', ' Sikar': 'Sikar', 'Dhaulpur':'Dholpur', ' Churu':'Churu' ' Banswara':'Banswara',
                    ' Jhunjhunu': 'Jhunjhunu', 'Jalor':'Jalore', ' Dungarpur' : "Dungarpur"}, regex = True, inplace = True)
```

In [7]:

```
df.District.value_counts()
```

Out[7]:

```
Dausa              109
```

```
Churu              109
Barmer             109
Udaipur            109
Jaipur             109
Baran              109
Ajmer              109
Bundi              109
Bhilwara           109
Pali               109
Alwar              109
Sirohi             109
Bikaner            109
Chittorgarh        109
Sawai Madhopur     109
Jhalawar           109
Sikar              109
Jaisalmer          109
Ganganagar         109
Jodhpur            109
Rajsamand          109
Banswara           109
Dungarpur          109
Nagaur             109
Kota               109
Karauli            109
Tonk               109
Dholpur            109
Bharatpur          109
Hanumangarh        109
Jhunjhunun         102
Jalore             102
Jhunjhunu            7
Jaloree              7
Name: District, dtype: int64
```

In [8]:

```
df.District.replace({'Jhunjhunun':'Jhunjhunu', 'Jaloree':'Jalore'}, regex=True, inplace=True)
```

In [9]:

```
df.District.value_counts()
```

Out[9]:

```
Dausa              109
Jhalawar           109
Barmer             109
Udaipur            109
Jaipur             109
Jhunjhunu          109
Baran              109
Jalore             109
Ajmer              109
Bundi              109
Bhilwara           109
Pali               109
Alwar              109
Sirohi             109
Bikaner            109
Chittorgarh        109
Sawai Madhopur     109
Sikar              109
Churu              109
Jaisalmer          109
Nagaur             109
Bharatpur          109
Dholpur            109
Tonk               109
Karauli            109
Ganganagar         109
Kota               109
Dungarpur          109
Banswara           109
```

```
Rajsamand          109
Jodhpur            109
Hanumangarh        109
Name: District, dtype: int64
```

ADDING 2003 DATA FOR ALL DISTRICTS

In [10]:

```
for i in df.District.value_counts().index:
    x = list(df[df['District']==i].mean(axis=0))
    x[0] = 2003
    x.insert(0,"Rajasthan")
    x.insert(1,i)
    df.loc[len(df.index)] = x
```

In [11]:

```
df.tail(32)
```

Out[11]:

	State	District	Year	January	February	March	April	May	June	July	August	Septen
3488	Rajasthan	Dausa	2003	5.123330	6.305101	3.813725	4.424523	11.837266	61.463211	239.584018	210.307312	99.367
3489	Rajasthan	Jhalawar	2003	4.988413	1.783037	2.723917	1.411697	5.684174	97.202771	277.382807	277.548606	187.924
3490	Rajasthan	Barmer	2003	1.116064	2.516844	3.459339	2.686266	3.976193	27.135422	97.847917	92.807514	46.639
3491	Rajasthan	Udaipur	2003	3.503450	1.375183	3.266119	2.286229	8.990587	84.871716	276.576294	244.719450	135.130
3492	Rajasthan	Jaipur	2003	5.036688	6.200963	4.511459	4.905807	12.760055	58.239954	208.908349	172.594266	85.475
3493	Rajasthan	Jhunjhunu	2003	6.316679	9.208706	5.124541	6.669752	14.393055	42.753899	160.850927	131.647321	63.046
3494	Rajasthan	Baran	2003	6.897055	2.673128	3.403541	1.975972	5.273459	75.900257	269.347128	259.275165	156.376
3495	Rajasthan	Jalore	2003	1.569073	1.720853	3.380670	1.685734	5.089413	44.412358	163.481936	136.795349	73.434
3496	Rajasthan	Ajmer	2003	3.119945	3.192862	4.688596	4.092156	9.927257	54.930037	179.536578	164.711596	87.627
3497	Rajasthan	Bundi	2003	4.781110	2.576064	4.005239	1.771303	8.175899	71.165807	236.649761	229.856358	126.978
3498	Rajasthan	Bhilwara	2003	3.117688	2.455872	5.533817	2.494147	9.690936	68.119128	200.653303	193.914578	111.178
3499	Rajasthan	Pali	2003	2.825972	2.437853	4.007266	3.446862	7.928578	49.718450	177.276266	161.930633	87.147
3500	Rajasthan	Alwar	2003	7.900954	10.117651	5.797982	6.166294	13.446349	54.573606	209.480927	190.955532	91.942
3501	Rajasthan	Sirohi	2003	2.819606	1.739523	3.515862	1.904725	6.783202	57.609807	234.841569	190.274101	106.019
3502	Rajasthan	Bikaner	2003	3.489872	6.053606	5.291835	10.246128	10.845945	27.748165	80.392440	72.128358	33.208
3503	Rajasthan	Chittorgarh	2003	2.618789	1.616312	3.780477	1.931872	8.453523	93.386404	263.768165	247.260505	148.734
3504	Rajasthan	Sawai Madhopur	2003	5.575101	4.426165	3.463807	3.290945	10.044275	62.338257	254.529092	233.661780	111.858
3505	Rajasthan	Sikar	2003	5.333752	7.595917	4.534716	6.703284	13.858798	46.889890	169.912358	133.459881	63.939
3506	Rajasthan	Churu	2003	5.165156	7.530220	5.500734	9.177899	13.918771	35.814303	119.530431	101.304532	45.668
3507	Rajasthan	Jaisalmer	2003	1.708954	4.452156	3.598615	3.213174	3.870385	20.521110	57.535165	60.101358	24.871
3508	Rajasthan	Nagaur	2003	3.411661	5.130339	4.150936	6.330266	10.621073	41.178394	140.737982	122.203844	58.245
3509	Rajasthan	Bharatpur	2003	7.999936	7.189128	4.747000	4.760890	11.539872	52.580661	237.530073	237.552450	106.666
3510	Rajasthan	Dholpur	2003	8.128606	5.557908	4.053606	3.802083	9.904789	53.364615	271.285917	266.654339	120.466
3511	Rajasthan	Tonk	2003	4.121358	2.666505	4.271339	2.682312	9.502697	59.097174	216.025844	199.695908	104.100
3512	Rajasthan	Karauli	2003	6.761202	5.532917	3.690028	3.525523	10.628128	62.009404	268.315752	251.904367	112.825
3513	Rajasthan	Ganganagar	2003	4.283046	8.202321	7.737945	10.048422	10.142128	24.487275	71.000716	63.066606	27.973
3514	Rajasthan	Kota	2003	5.893431	2.288550	3.518881	1.735037	6.835495	77.701606	259.416679	255.930578	147.234
3515	Rajasthan	Dungarpur	2003	2.402495	0.432450	1.355523	2.073266	8.666523	112.480092	332.180294	277.972303	146.901
3516	Rajasthan	Banswara	2003	1.119688	0.615817	2.042569	1.452165	6.201642	122.264734	324.438128	283.912927	171.575
3517	Rajasthan	Rajsamand	2003	4.889991	2.813367	6.352725	3.696706	8.817642	60.934358	200.247266	185.291826	108.451
3518	Rajasthan	Jodhpur	2003	1.885560	4.495826	3.723312	5.399670	7.842550	31.369376	100.273018	98.726367	47.160

3519	Rajasthan	Hanumangarh	2003	5.819826	8.147743	7.068633	8.905798	11.815018	30.705697	96.372330	84.289119	39.125
	State	**District**	**Year**	**January**	**February**	**March**	**April**	**May**	**June**	**July**	**August**	**Septer**

ADDING 2011 TO 2017 DATA

In [12]:

```
rain_11_17 = pd.read_excel("Book (2).xlsx")
df1 = pd.DataFrame(rain_11_17)
df1.head()
```

Out[12]:

	State	District	Year	January	February	March	April	May	June	July	August	September	October	November	December
0	Rajasthan	Ajmer	2015	0.0	1.5	2.0	0.0	1.2	51.5	220.7	110.7	4.1	0.0	0.0	0.0
1	Rajasthan	Alwar	2015	0.0	1.0	164.0	0.0	47.6	34.4	116.2	133.2	55.3	5.4	3.2	0.0
2	Rajasthan	Banswara	2015	0.0	0.0	0.0	0.0	0.0	122.6	407.7	109.1	26.1	0.7	0.0	0.0
3	Rajasthan	Baran	2015	0.0	0.0	2.5	0.0	5.6	130.4	398.3	329.5	20.3	0.5	0.0	1.6
4	Rajasthan	Bharatpur	2015	0.0	0.0	0.4	0.0	23.7	54.0	161.7	158.8	185.0	16.1	1.0	3.7

In [13]:

```
df1.District.unique()
```

Out[13]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Baran', 'Bharatpur', 'Bhilwara',
       'Bundi', 'Chittorgarh', 'Dausa', 'Dholpur', 'Dungarpur', 'Jaipur',
       'Jhalawar', 'Jhunjhunu', 'Karauli', 'Kota', 'Pratapgarh',
       'Rajsamamd', 'Sawai Madhopur', 'Sikar', 'Sirohi', 'Tonk',
       'Udaipur', 'Barmer', 'Bikaner', 'Churu', 'Hanumangarh',
       'Jaisalmer', 'Jalore', 'Jodhpur', 'Nagour', 'Pali', 'Ganganagar',
       'AJMER', 'JAIPUR', 'DAUSA', 'TONK', 'SIKAR', 'JHUNJHUNU', 'NAGAUR',
       'ALWAR', 'BHARATPUR', 'DHOLPUR', 'SAWAI MADHOPUR', 'KARAULI',
       'BIKANER', 'CHURU', 'JAISALMER', 'GANGANAGAR', 'HANUMANGARH',
       'JODHPUR', 'BARMER', 'JALORE', 'PALI', 'SIROHI', 'KOTA', 'BARAN',
       'BUNDI', 'JHALAWAR', 'BANSWARA', 'DUNGARPUR', 'UDAIPUR',
       'BHILWARA', 'CHITTORGARH', 'RAJSAMAND'], dtype=object)
```

In [14]:

```
df1.District.replace({'AJMER':'Ajmer','JAIPUR':'Jaipur','DAUSA':'Dausa','TONK':'Tonk','SIKAR':'Sika
r','JHUNJHUNU':'Jhunjhunu',

'NAGAUR':'Nagaur','Nagour':'Nagaur','ALWAR':'Alwar','BHARATPUR':'Bharatpur','DHOLPUR':'Dholpur',
                    'SAWAI MADHOPUR':'Sawai Madhopur','KARAULI':'Karauli','BIKANER':'Bikaner','CHUR
U':'Churu',
                    'JAISALMER':'Jaisalmer','GANGANAGAR':'Ganganagar','HANUMANGARH':'Hanumangarh','
JODHPUR':'Jodhpur',
                    'BARMER':'Barmer','JALORE':'Jalore','PALI':'Pali','SIROHI':'Sirohi','KOTA':'Kot
','BARAN':'Baran',

'BUNDI':'Bundi','JHALAWAR':'Jhalawar','BANSWARA':'Banswara','DUNGARPUR':'Dungarpur','UDAIPUR':'Udai
pur',
                    'BHILWARA':'Bhilwara','CHITTORGARH':'Chittorgarh','RAJSAMAND':'Rajsamand'}, reg
ex=True, inplace=True)
```

In [15]:

```
df1.District.unique()
```

Out[15]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Baran', 'Bharatpur', 'Bhilwara',
       'Bundi', 'Chittorgarh', 'Dausa', 'Dholpur', 'Dungarpur', 'Jaipur',
       'Jhalawar', 'Jhunjhunu', 'Karauli', 'Kota', 'Pratapgarh',
       'Rajsamamd', 'Sawai Madhopur', 'Sikar', 'Sirohi', 'Tonk',
```

```
       'Udaipur', 'Barmer', 'Bikaner', 'Churu', 'Hanumangarh',
       'Jaisalmer', 'Jalore', 'Jodhpur', 'Nagaur', 'Pali', 'Ganganagar',
       'Jaipur', 'Tonk', 'Hanumangarh', 'Barmer', 'Jalore', 'Pali',
       'Baran', 'Rajsamand'], dtype=object)
```

In [16]:

```
df1.District.replace({'Rajsamamd':'Rajsamand','Baran':'Baran','Jaipur':'Jaipur','Jalore':'Jalore','
Barmer':'Barmer',
                      'Hanumangarh':'Hanumangarh','Tonk':'Tonk','Pali':'Pali'}, regex=True,
inplace=True)
```

In [17]:

```
df1.District.value_counts()
```

Out[17]:

```
Dausa             7
Jhalawar          7
Bhilwara          7
Bharatpur         7
Bikaner           7
Karauli           7
Jalore            7
Hanumangarh       7
Dungarpur         7
Ganganagar        7
Dholpur           7
Udaipur           7
Banswara          7
Chittorgarh       7
Barmer            7
Rajsamand         7
Jaipur            7
Sirohi            7
Pali              7
Sawai Madhopur    7
Tonk              7
Sikar             7
Alwar             7
Ajmer             7
Kota              7
Jodhpur           7
Jhunjhunu         7
Churu             7
Jaisalmer         7
Bundi             7
Baran             7
Nagaur            7
Pratapgarh        3
Name: District, dtype: int64
```

In [18]:

```
df1.District.unique()
```

Out[18]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Baran', 'Bharatpur', 'Bhilwara',
       'Bundi', 'Chittorgarh', 'Dausa', 'Dholpur', 'Dungarpur', 'Jaipur',
       'Jhalawar', 'Jhunjhunu', 'Karauli', 'Kota', 'Pratapgarh',
       'Rajsamand', 'Sawai Madhopur', 'Sikar', 'Sirohi', 'Tonk',
       'Udaipur', 'Barmer', 'Bikaner', 'Churu', 'Hanumangarh',
       'Jaisalmer', 'Jalore', 'Jodhpur', 'Nagaur', 'Pali', 'Ganganagar'],
      dtype=object)
```

In [19]:

```
df1.drop(df1[df1["District"]=="Pratapgarh"].index,inplace=True)
```

In [20]:

```
df1.District.value_counts()
```

Out[20]:

```
Dausa             7
Sirohi            7
Bhilwara          7
Bharatpur         7
Bikaner           7
Karauli           7
Jalore            7
Hanumangarh       7
Dungarpur         7
Ganganagar        7
Dholpur           7
Udaipur           7
Banswara          7
Chittorgarh       7
Barmer            7
Rajsamand         7
Jhalawar          7
Jaipur            7
Pali              7
Sawai Madhopur    7
Tonk              7
Sikar             7
Ajmer             7
Alwar             7
Kota              7
Jodhpur           7
Jhunjhunu         7
Churu             7
Jaisalmer         7
Bundi             7
Baran             7
Nagaur            7
Name: District, dtype: int64
```

MERGING THE 2 DATA FRAMES

In [21]:

```
frames = [df,df1]
res = pd.concat(frames)
```

In [22]:

```
res.head()
```

Out[22]:

	State	District	Year	January	February	March	April	May	June	July	August	September	October	November	December
0	Rajasthan	Ajmer	2004	2.3	0.0	0.0	0.0	13.4	21.1	120.4	282.5	15.4	11.6	0.0	0.0
1	Rajasthan	Ajmer	2005	0.2	2.9	9.5	28.1	2.4	57.0	148.1	79.9	170.6	0.0	0.0	0.0
2	Rajasthan	Ajmer	2006	0.0	0.0	0.7	0.0	29.0	48.3	101.7	219.4	43.4	0.1	0.0	0.0
3	Rajasthan	Ajmer	2007	0.4	10.6	4.5	0.5	0.9	47.1	172.9	92.6	46.3	0.0	0.0	1.8
4	Rajasthan	Ajmer	2008	0.0	0.0	0.4	5.7	15.4	89.7	86.4	189.7	85.9	19.1	0.0	0.0

In [23]:

```
res.Year.unique()
```

Out[23]:

```
array([2004, 2005, 2006, 2007, 2008, 2009, 2010, 1901, 1902, 1903, 1904,
       1905, 1906, 1907, 1908, 1909, 1910, 1911, 1912, 1913, 1914, 1915,
       1916, 1917, 1918, 1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926,
```

```
       1916, 1917, 1918, 1919, 1920, 1921, 1922, 1923, 1924, 1925, 1926,
       1927, 1928, 1929, 1930, 1931, 1932, 1933, 1934, 1935, 1936, 1937,
       1938, 1939, 1940, 1941, 1942, 1943, 1944, 1945, 1946, 1947, 1948,
       1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959,
       1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970,
       1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981,
       1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992,
       1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003,
       2015, 2016, 2017, 2011, 2012, 2013, 2014], dtype=int64)
```

In [24]:

```
res.Year.value_counts()
```

Out[24]:

```
2017    32
1986    32
1962    32
1964    32
1966    32
        ..
1947    32
1949    32
1951    32
1953    32
1902    32
Name: Year, Length: 117, dtype: int64
```

In [25]:

```
res.District.value_counts()
```

Out[25]:

```
Dausa             117
Nagaur            117
Udaipur           117
Jhunjhunu         117
Ajmer             117
Bundi             117
Bhilwara          117
Alwar             117
Sirohi            117
Bikaner           117
Chittorgarh       117
Sawai Madhopur    117
Jhalawar          117
Churu             117
Jaisalmer         117
Sikar             117
Bharatpur         117
Kota              117
Jodhpur           117
Rajsamand         117
Banswara          117
Dholpur           117
Dungarpur         117
Ganganagar        117
Karauli           117
Tonk              117
Pali              110
Jalore            110
Baran             110
Jaipur            110
Barmer            110
Hanumangarh       110
Hanumangarh         7
Baran               7
Jalore              7
Pali                7
Barmer              7
Jaipur              7
Name: District, dtype: int64
```

In [26]:

```
res.District.unique()
```

Out[26]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Baran', 'Barmer', 'Bharatpur',
       'Bhilwara', 'Bikaner', 'Bundi', 'Chittorgarh', 'Churu', 'Dausa',
       'Dholpur', 'Dungarpur', 'Hanumangarh', 'Jaipur', 'Jaisalmer',
       'Jalore', 'Jhalawar', 'Jhunjhunu', 'Jodhpur', 'Karauli', 'Kota',
       'Nagaur', 'Pali', 'Rajsamand', 'Sawai Madhopur', 'Sikar', 'Sirohi',
       'Ganganagar', 'Tonk', 'Udaipur', 'Baran', 'Jaipur', 'Barmer',
       'Hanumangarh', 'Jalore', 'Pali'], dtype=object)
```

In [27]:

```
res.replace({'Baran':'Baran','Barmer':'Barmer','Pali':'Pali','Jalore':'Jalore','Jaipur':'Jaipur',
             'Hanumangarh':'Hanumangarh'},regex=True,inplace=True)
```

In [28]:

```
res.District.unique()
```

Out[28]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Baran', 'Barmer', 'Bharatpur',
       'Bhilwara', 'Bikaner', 'Bundi', 'Chittorgarh', 'Churu', 'Dausa',
       'Dholpur', 'Dungarpur', 'Hanumangarh', 'Jaipur', 'Jaisalmer',
       'Jalore', 'Jhalawar', 'Jhunjhunu', 'Jodhpur', 'Karauli', 'Kota',
       'Nagaur', 'Pali', 'Rajsamand', 'Sawai Madhopur', 'Sikar', 'Sirohi',
       'Ganganagar', 'Tonk', 'Udaipur'], dtype=object)
```

In [29]:

```
res.District.value_counts()
```

Out[29]:

```
Dausa             117
Jhalawar          117
Jhunjhunu         117
Barmer            117
Ajmer             117
Bundi             117
Bhilwara          117
Baran             117
Alwar             117
Sirohi            117
Bikaner           117
Chittorgarh       117
Sawai Madhopur    117
Sikar             117
Churu             117
Jaisalmer         117
Nagaur            117
Bharatpur         117
Dholpur           117
Tonk              117
Pali              117
Jalore            117
Karauli           117
Ganganagar        117
Kota              117
Hanumangarh       117
Dungarpur         117
Banswara          117
Rajsamand         117
Jaipur            117
Jodhpur           117
Udaipur           117
Name: District, dtype: int64
```

ADDING 2018 AND 2019 DATA

In [30]:

```
for i in res.District.value_counts().index:
    x = list(res[res['District']==i].mean(axis=0))
    x[0] = 2018
    x.insert(0,"Rajasthan")
    x.insert(1,i)
    res.loc[len(res.index)] = x
```

In [31]:

```
res.tail(32)
```

Out[31]:

	State	District	Year	January	February	March	April	May	June	July	August	Septen
3744	Rajasthan	Dausa	2018	5.187746	6.419326	4.335981	4.346987	11.691447	62.132933	235.884975	215.794909	97.870
3745	Rajasthan	Jhalawar	2018	4.766029	1.963539	3.031033	1.499032	5.772301	100.420554	320.026571	282.121766	182.18
3746	Rajasthan	Jhunjhunu	2018	6.319955	9.033827	5.434184	7.100622	16.428513	43.848965	159.134205	133.044490	62.303
3747	Rajasthan	Barmer	2018	1.106556	2.456862	3.317328	2.675977	4.075053	26.925610	100.055307	93.189116	47.845
3748	Rajasthan	Ajmer	2018	3.157213	3.201836	4.867056	4.049890	9.874344	55.361573	178.437808	167.755347	86.933
3749	Rajasthan	Bundi	2018	4.768565	2.684334	4.865609	1.835413	8.003837	71.093494	236.745930	233.782900	123.440
3750	Rajasthan	Bhilwara	2018	3.049108	2.670478	5.807007	2.485950	9.640196	68.368411	202.421909	197.422253	109.485
3751	Rajasthan	Baran	2018	6.616889	2.791830	3.611022	1.939803	5.268209	81.653233	276.587899	267.215113	151.692
3752	Rajasthan	Alwar	2018	7.813718	10.183262	7.888701	6.417028	13.683747	54.380313	206.190615	191.771013	92.061
3753	Rajasthan	Sirohi	2018	2.786809	1.672201	3.472178	1.995040	6.593609	57.638280	251.363868	194.171377	109.135
3754	Rajasthan	Bikaner	2018	3.377657	6.124757	5.744460	10.409181	11.173965	28.235027	81.082636	73.485636	33.777
3755	Rajasthan	Chittorgarh	2018	2.630485	1.605080	3.938055	1.951332	8.394765	92.460721	266.887164	254.005603	145.984
3756	Rajasthan	Sawai Madhopur	2018	5.621890	4.535711	3.919819	3.425675	9.688635	65.152207	252.751283	238.525605	108.223
3757	Rajasthan	Sikar	2018	5.456519	7.567957	4.801015	6.823601	14.004853	47.233230	169.931277	137.012708	62.713
3758	Rajasthan	Churu	2018	5.404847	7.464310	5.910092	9.398879	14.383460	37.497208	121.333739	102.755543	46.057
3759	Rajasthan	Jaisalmer	2018	1.647735	4.460147	3.669638	3.344010	4.109764	20.543779	58.427078	60.654268	25.899
3760	Rajasthan	Nagaur	2018	3.601561	5.036217	4.311991	6.493413	10.713830	41.373704	139.904085	124.051477	58.312
3761	Rajasthan	Bharatpur	2018	8.240111	6.988924	5.086068	5.088871	11.540050	53.382672	234.656479	234.358713	106.724
3762	Rajasthan	Dholpur	2018	8.197834	5.813418	4.591424	3.859223	9.695101	53.471860	267.482486	265.575875	120.228
3763	Rajasthan	Tonk	2018	4.155123	2.781329	4.761088	2.795336	9.432450	59.623839	215.617460	205.798717	101.186
3764	Rajasthan	Pali	2018	2.838094	2.350118	4.023071	3.372264	7.885843	50.023329	181.461447	165.539057	86.080
3765	Rajasthan	Jalore	2018	1.684599	1.707640	3.214305	1.710519	4.995174	44.067174	169.214640	136.807593	74.861
3766	Rajasthan	Karauli	2018	6.685745	5.627529	4.155581	3.604338	10.517899	62.716533	263.766947	250.579319	109.298
3767	Rajasthan	Ganganagar	2018	4.182351	8.303037	8.019435	9.896807	9.855847	24.244447	70.064775	63.365185	31.373
3768	Rajasthan	Kota	2018	5.806645	2.350774	3.978435	1.820120	6.573543	80.231424	259.808843	259.529603	143.045
3769	Rajasthan	Hanumangarh	2018	5.706674	8.156853	7.438886	8.913998	11.553436	31.472023	95.291080	86.333360	40.039
3770	Rajasthan	Dungarpur	2018	2.273286	0.525380	1.453056	2.095378	8.189893	108.886411	332.437883	277.159430	147.481
3771	Rajasthan	Banswara	2018	1.077485	0.684956	1.996432	1.503745	5.939151	118.668553	326.131574	291.052324	170.457
3772	Rajasthan	Rajsamand	2018	4.631615	2.740772	6.392305	3.575536	8.592655	60.878456	202.743583	186.894879	107.767
3773	Rajasthan	Jaipur	2018	5.055006	6.232530	4.885132	4.908024	12.446206	61.378589	216.164259	197.777515	85.459
3774	Rajasthan	Jodhpur	2018	2.021466	4.542229	3.644994	5.521912	8.153680	31.901123	101.213094	101.090601	49.453
3775	Rajasthan	Udaipur	2018	3.415209	1.351027	3.355326	2.334062	8.744996	83.857169	278.854635	243.649055	136.114

In [32]:

```
for i in res.District.value_counts().index:
    x = list(res[res['District']==i].mean(axis=0))
    x[0] = 2019
    x.insert(0,"Rajasthan")
    x.insert(1,i)
    res.loc[len(res.index)] = x
```

In [33]:

```
res.tail(32)
```

Out[33]:

	State	District	Year	January	February	March	April	May	June	July	August	Septen
3776	Rajasthan	Dausa	2019	5.187746	6.419326	4.335981	4.346987	11.691447	62.132933	235.884975	215.794909	97.870
3777	Rajasthan	Jhalawar	2019	4.766029	1.963539	3.031033	1.499032	5.772301	100.420554	320.026571	282.121766	182.181
3778	Rajasthan	Jhunjhunu	2019	6.319955	9.033827	5.434184	7.100622	16.428513	43.848965	159.134205	133.044490	62.303
3779	Rajasthan	Barmer	2019	1.106556	2.456862	3.317328	2.675977	4.075053	26.925610	100.055307	93.189116	47.845
3780	Rajasthan	Ajmer	2019	3.157213	3.201836	4.867056	4.049890	9.874344	55.361573	178.437808	167.755347	86.933
3781	Rajasthan	Bundi	2019	4.768565	2.684334	4.865609	1.835413	8.003837	71.093494	236.745930	233.782900	123.440
3782	Rajasthan	Bhilwara	2019	3.049108	2.670478	5.807007	2.485950	9.640196	68.368411	202.421909	197.422253	109.485
3783	Rajasthan	Baran	2019	6.616889	2.791830	3.611022	1.939803	5.268209	81.653233	276.587899	267.215113	151.692
3784	Rajasthan	Alwar	2019	7.813718	10.183262	7.888701	6.417028	13.683747	54.380313	206.190615	191.771013	92.061
3785	Rajasthan	Sirohi	2019	2.786809	1.672201	3.472178	1.995040	6.593609	57.638280	251.363868	194.171377	109.135
3786	Rajasthan	Bikaner	2019	3.377657	6.124757	5.744460	10.409181	11.173965	28.235027	81.082636	73.485636	33.777
3787	Rajasthan	Chittorgarh	2019	2.630485	1.605080	3.938055	1.951332	8.394765	92.460721	266.887164	254.005603	145.984
3788	Rajasthan	Sawai Madhopur	2019	5.621890	4.535711	3.919819	3.425675	9.688635	65.152207	252.751283	238.525605	108.223
3789	Rajasthan	Sikar	2019	5.456519	7.567957	4.801015	6.823601	14.004853	47.233230	169.931277	137.012708	62.713
3790	Rajasthan	Churu	2019	5.404847	7.464310	5.910092	9.398879	14.383460	37.497208	121.333739	102.755543	46.057
3791	Rajasthan	Jaisalmer	2019	1.647735	4.460147	3.669638	3.344010	4.109764	20.543779	58.427078	60.654268	25.899
3792	Rajasthan	Nagaur	2019	3.601561	5.036217	4.311991	6.493413	10.713830	41.373704	139.904085	124.051477	58.312
3793	Rajasthan	Bharatpur	2019	8.240111	6.988924	5.086068	5.088871	11.540050	53.382672	234.656479	234.358713	106.724
3794	Rajasthan	Dholpur	2019	8.197834	5.813418	4.591424	3.859223	9.695101	53.471860	267.482486	265.575875	120.228
3795	Rajasthan	Tonk	2019	4.155123	2.781329	4.761088	2.795336	9.432450	59.623839	215.617460	205.798717	101.186
3796	Rajasthan	Pali	2019	2.838094	2.350118	4.023071	3.372264	7.885843	50.023329	181.461447	165.539057	86.080
3797	Rajasthan	Jalore	2019	1.684599	1.707640	3.214305	1.710519	4.995174	44.067174	169.214640	136.807593	74.861
3798	Rajasthan	Karauli	2019	6.685745	5.627529	4.155581	3.604338	10.517899	62.716533	263.766947	250.579319	109.298
3799	Rajasthan	Ganganagar	2019	4.182351	8.303037	8.019435	9.896807	9.855847	24.244447	70.064775	63.365185	31.373
3800	Rajasthan	Kota	2019	5.806645	2.350774	3.978435	1.820120	6.573543	80.231424	259.808843	259.529603	143.045
3801	Rajasthan	Hanumangarh	2019	5.706674	8.156853	7.438886	8.913998	11.553436	31.472023	95.291080	86.333360	40.039
3802	Rajasthan	Dungarpur	2019	2.273286	0.525380	1.453056	2.095378	8.189893	108.886411	332.437883	277.159430	147.481
3803	Rajasthan	Banswara	2019	1.077485	0.684956	1.996432	1.503745	5.939151	118.668553	326.131574	291.052324	170.457
3804	Rajasthan	Rajsamand	2019	4.631615	2.740772	6.392305	3.575536	8.592655	60.878456	202.743583	186.894879	107.767
3805	Rajasthan	Jaipur	2019	5.055006	6.232530	4.885132	4.908024	12.446206	61.378589	216.164259	197.777515	85.459
3806	Rajasthan	Jodhpur	2019	2.021466	4.542229	3.644994	5.521912	8.153680	31.901123	101.213094	101.090601	49.453
3807	Rajasthan	Udaipur	2019	3.415209	1.351027	3.355326	2.334062	8.744996	83.857169	278.854635	243.649055	136.114

In [34]:

```
res.shape
```

Out[34]:

```
(3808, 16)
```

DROPPING ROWS THAT ARE NOT REQUIRED

In [35]:

```
m = res[(res.Year >= 1901) & (res.Year <= 1996)].index
res.drop(m,inplace=True)
```

In [36]:

```
res.Year.unique()
```

Out[36]:

```
array([2004, 2005, 2006, 2007, 2008, 2009, 2010, 1997, 1998, 1999, 2000,
       2001, 2002, 2003, 2015, 2016, 2017, 2011, 2012, 2013, 2014, 2018,
       2019], dtype=int64)
```

In [37]:

```
res.shape
```

Out[37]:

```
(733, 16)
```

In [38]:

```
res.head()
```

Out[38]:

	State	District	Year	January	February	March	April	May	June	July	August	September	October	November	December
0	Rajasthan	Ajmer	2004	2.3	0.0	0.0	0.0	13.4	21.1	120.4	282.5	15.4	11.6	0.0	0.0
1	Rajasthan	Ajmer	2005	0.2	2.9	9.5	28.1	2.4	57.0	148.1	79.9	170.6	0.0	0.0	0.0
2	Rajasthan	Ajmer	2006	0.0	0.0	0.7	0.0	29.0	48.3	101.7	219.4	43.4	0.1	0.0	0.0
3	Rajasthan	Ajmer	2007	0.4	10.6	4.5	0.5	0.9	47.1	172.9	92.6	46.3	0.0	0.0	1.8
4	Rajasthan	Ajmer	2008	0.0	0.0	0.4	5.7	15.4	89.7	86.4	189.7	85.9	19.1	0.0	0.0

EXPORTING FILE

In [39]:

```
res.sort_values(['District','Year'],inplace=True)
```

In [40]:

```
res.to_excel("Rainfall_1997_to_2019.xlsx",index=False)
```

DATA CLEANING

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
crop_data = pd.read_excel('D:\DataScience\MINOR PROJECT\Rajasthan_Crop_Final.xlsx')
```

In [3]:

```
df = pd.DataFrame(crop_data)
```

In [4]:

```
df.head()
```

Out[4]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0
2	Rajasthan	AJMER	1997	Kharif	Maize	43600.0	33100.0
3	Rajasthan	AJMER	1997	Kharif	Onion	2800.0	4500.0
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0

In [5]:

```
df.shape
```

Out[5]:

```
(5152, 7)
```

In [6]:

```
df.Crop.value_counts()
```

Out[6]:

```
Rapeseed &Mustard    748
Wheat                748
Barley               747
Bajra                742
Jowar                739
Onion                723
Maize                705
Name: Crop, dtype: int64
```

In [7]:

```
df.District_Name.value_counts()
```

Out[7]:

```
TONK              161
JAIPUR            161
JHALAWAR          161
NAGAUR            161
BHARATPUR         161
```

```
BHARATPUR          161
AJMER              161
SAWAI MADHOPUR     161
PALI               161
ALWAR              161
JALORE             161
CHITTORGARH        160
RAJSAMAND          160
BARAN              160
BHILWARA           160
SIROHI             160
SIKAR              160
JODHPUR            160
BANSWARA           160
DAUSA              160
UDAIPUR            159
GANGANAGAR         159
BARMER             159
KOTA               159
DUNGARPUR          159
BUNDI              157
DHOLPUR            157
KARAULI            155
JHUNJHUNU          155
HANUMANGARH        154
BIKANER            151
CHURU              148
JAISALMER          146
PRATAPGARH          84
Name: District_Name, dtype: int64
```

In [8]:

```
df.Area.isnull().sum()
```

Out[8]:

```
33
```

DROPPING DATA THAT IS NOT REQUIRED

In [9]:

```
df.drop(df[df['Crop']=="Onion"].index, inplace = True)
```

In [10]:

```
df.shape
```

Out[10]:

```
(4429, 7)
```

In [11]:

```
df.drop(df[df['Crop']=="Maize"].index, inplace = True)
```

In [12]:

```
df.shape
```

Out[12]:

```
(3724, 7)
```

In [13]:

```
df.drop(df[df['District_Name']=="PRATAPGARH"].index, inplace = True)
```

In [14]:

```
df.shape
```

Out[14]:

```
(3664, 7)
```

In [15]:

```
df.head()
```

Out[15]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0
5	Rajasthan	AJMER	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0
6	Rajasthan	AJMER	1997	Rabi	Wheat	79300.0	144500.0

ADDING OUR TARGET FEATURE - YIELD

In [16]:

```
df["Yield"] = df['Production']/df['Area']
```

In [17]:

```
df.head(15)
```

Out[17]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0	0.537102
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0	0.326723
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0	1.170040
5	Rajasthan	AJMER	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098
6	Rajasthan	AJMER	1997	Rabi	Wheat	79300.0	144500.0	1.822194
7	Rajasthan	AJMER	1998	Kharif	Bajra	55089.0	6045.0	0.109732
8	Rajasthan	AJMER	1998	Kharif	Jowar	105177.0	6080.0	0.057807
10	Rajasthan	AJMER	1998	Rabi	Barley	21534.0	40167.0	1.865283
11	Rajasthan	AJMER	1998	Rabi	Rapeseed &Mustard	27203.0	13797.0	0.507187
12	Rajasthan	AJMER	1998	Rabi	Wheat	74805.0	134057.0	1.792086
14	Rajasthan	AJMER	1999	Kharif	Bajra	57959.0	5922.0	0.102176
15	Rajasthan	AJMER	1999	Kharif	Jowar	112136.0	21196.0	0.189020
17	Rajasthan	AJMER	1999	Rabi	Barley	12259.0	17582.0	1.434212
18	Rajasthan	AJMER	1999	Rabi	Rapeseed &Mustard	41337.0	20544.0	0.496988
19	Rajasthan	AJMER	1999	Rabi	Wheat	40412.0	41161.0	1.018534

In [18]:

```
df.head()
```

Out[18]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0	0.537102
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0	0.326723
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0	1.170040
5	Rajasthan	AJMER	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098
6	Rajasthan	AJMER	1997	Rabi	Wheat	79300.0	144500.0	1.822194

ADDING SOIL DATA FOR ALL DISTRICTS

In [19]:

```
soil = []
for x in df['District_Name']:

    if x=="BARMER":
        soil.append("Desert soils and sand dunes aeolian soil, coarse sand in texture some places
calcareous")

    elif x=="GANGANAGAR" or x=="HANUMANGARH":
        soil.append("Alluvial deposites calcareous, high soluble salts & exchangeable sodium")

    elif x=="BIKANER" or x=="JAISALMER":
        soil.append("Desert soils and sand dunes aeolian soil, loamycoarse in texture &
calcareous")

    elif x=="NAGAUR" or x=="SIKAR" or x=="JHUNJHUNU":
        soil.append("Sandy loam, sallow depth red soils in depressions")

    elif x=="JALORE" or x=="PALI":
        soil.append("Red desert soils")

    elif x=="JAIPUR" or x=="AJMER" or x=="DAUSA" or x=="TONK":
        soil.append("Sierozens, eastern part alluvial, west north west lithosols, foot hills, browr
soils")

    elif x=="ALWAR" or x=="DHOLPUR" or x=="BHARATPUR" or x=="KARAULI" or x=="SAWAI MADHOPUR":
        soil.append("Alluvial prone to water logging, nature of recently alluvial calcareous has b
een observed")

    elif x=="BHILWARA" or x=="RAJSAMAND":
        soil.append("Soil are lithosolsat foot hills & alluvials in plains")

    elif x=="DUNGARPUR" or x=="BANSWARA":
        soil.append("Predominantly reddish medium texture, well drained calcareous, shallow on
hills, deep soils in valleys")

    elif x=="KOTA" or x=="JHALAWAR" or x=="BUNDI" or x=="BARAN":
        soil.append("Black of alluvial origin, clay loam, groundwater salinity")

    elif x=="JODHPUR":
        soil.append("Desert soils and sand dunes aeolian soil, coarse sand in texture some places
calcareous, Red desert soils")

    elif x=="CHURU":
        soil.append("Desert soils and sand dunes aeolian soil, loamycoarse in texture &
calcareous, Sandy loam, sallow depth red soils in depressions")

    elif x=="SIROHI":
        soil.append("Red desert soils, Soil are lithosolsat foot hills & alluvials in plains")

    elif x=="UDAIPUR" or x=="CHITTORGARH":
        soil.append("Soil are lithosolsat foot hills & alluvials in plains, Predominantly reddish
medium texture, well drained calcareous, shallow on hills, deep soils in valleys")
```

In [20]:

```
df["Soil_Type"] = soil
```

In [21]:

```
df.head()
```

Out[21]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Soil_Type
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0	0.537102	Sierozens, eastern part alluvial, west north w...
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0	0.326723	Sierozens, eastern part alluvial, west north w...
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0	1.170040	Sierozens, eastern part alluvial, west north w...
5	Rajasthan	AJMER	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	Sierozens, eastern part alluvial, west north w...
6	Rajasthan	AJMER	1997	Rabi	Wheat	79300.0	144500.0	1.822194	Sierozens, eastern part alluvial, west north w...

In [22]:

```
df.shape
```

Out[22]:

```
(3664, 9)
```

In [23]:

```
df.to_excel("Data_With_SoilTypeCol_and_SeasonalRainfall.xlsx",index=False)
```

In [24]:

```
df1 = df['Soil_Type'].str.get_dummies(sep=',')
```

In [25]:

```
df1.head()
```

Out[25]:

	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	coarse sand in texture some places calcareous	deep soils in valleys	eastern part alluvial	foot hills	...	west north west lithosols	Alluvial deposites calcareous	Alluvial prone to water logging	Bla alluv orig
0	0	0	0	0	1	0	0	0	1	1	...	1	0	0	
1	0	0	0	0	1	0	0	0	1	1	...	1	0	0	
4	0	0	0	0	1	0	0	0	1	1	...	1	0	0	
5	0	0	0	0	1	0	0	0	1	1	...	1	0	0	
6	0	0	0	0	1	0	0	0	1	1	...	1	0	0	

5 rows × 27 columns

In [26]:

```
frames = [df,df1]
df = pd.concat(frames,axis=1)
df.head()
```

Out[26]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Soil_Type	Predominantly reddish medium texture	...	west north west lithosols
0	Rajasthan	AJMER	1997	Kharif	Bajra	56600.0	30400.0	0.537102	Sierozens, eastern part alluvial, west north w...	0	...	1
1	Rajasthan	AJMER	1997	Kharif	Jowar	105900.0	34600.0	0.326723	Sierozens, eastern part alluvial, west north w...	0	...	1
4	Rajasthan	AJMER	1997	Rabi	Barley	24700.0	28900.0	1.170040	Sierozens, eastern part alluvial, west north w...	0	...	1
5	Rajasthan	AJMER	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	Sierozens, eastern part alluvial, west north w...	0	...	1
6	Rajasthan	AJMER	1997	Rabi	Wheat	79300.0	144500.0	1.822194	Sierozens, eastern part alluvial, west north w...	0	...	1

5 rows × 36 columns

FURTHER CLEANING AND ADDING RAINFALL DATA

In [27]:

```
df.District_Name.replace({'AJMER':'Ajmer','JAIPUR':'Jaipur','DAUSA':'Dausa','TONK':'Tonk','SIKAR':
'Sikar','JHUNJHUNU':'Jhunjhunu',
                    'NAGAUR':'Nagaur','ALWAR':'Alwar','BHARATPUR':'Bharatpur','DHOLPUR':'Dholpur',
                    'SAWAI MADHOPUR':'Sawai Madhopur','KARAULI':'Karauli','BIKANER':'Bikaner','CHUR
U':'Churu',
                    'JAISALMER':'Jaisalmer','GANGANAGAR':'Ganganagar','HANUMANGARH':'Hanumangarh','
JODHPUR':'Jodhpur',
                    'BARMER':'Barmer','JALORE':'Jalore','PALI':'Pali','SIROHI':'Sirohi','KOTA':'Kot
','BARAN':'Baran',

'BUNDI':'Bundi','JHALAWAR':'Jhalawar','BANSWARA':'Banswara','DUNGARPUR':'Dungarpur','UDAIPUR':'Udai
pur',
                    'BHILWARA':'Bhilwara','CHITTORGARH':'Chittorgarh','RAJSAMAND':'Rajsamand'}, reg
ex=True, inplace=True)
```

In [28]:

```
df.District_Name.value_counts()
```

Out[28]:

```
Dholpur           115
Jaipur            115
Tonk              115
Rajsamand         115
Alwar             115
Sawai Madhopur    115
Bundi             115
```

```
Jhalawar          115
Ajmer             115
Jalore            115
Pali              115
Bharatpur         115
Bikaner           115
Baran             115
Nagaur            115
Bhilwara          115
Dausa             115
Jaisalmer         115
Jodhpur           115
Kota              114
Udaipur           114
Sikar             114
Jhunjhunu         114
Ganganagar        114
Chittorgarh       114
Barmer            114
Banswara          114
Sirohi            114
Dungarpur         114
Hanumangarh       113
Churu             113
Karauli           113
Name: District_Name, dtype: int64
```

In [29]:

```
len(df.District_Name.unique())
```

Out[29]:

32

In [30]:

```
df.head()
```

Out[30]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Soil_Type	Predominantly reddish medium texture	...	west north west lithosols
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	Sierozens, eastern part alluvial, west north w...	0	...	1
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	Sierozens, eastern part alluvial, west north w...	0	...	1
4	Rajasthan	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	Sierozens, eastern part alluvial, west north w...	0	...	1
5	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	Sierozens, eastern part alluvial, west north w...	0	...	1
6	Rajasthan	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	Sierozens, eastern part alluvial,	0	...	1

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Soil_Type	Predominantly reddish medium	...	west north west
									west north w...			

5 rows × 36 columns

In [31]:

```
df.drop(columns = ['Soil_Type'],inplace=True)
```

In [32]:

```
df.head()
```

Out[32]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	west north west lithosols	d ca
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	0	0	...	1	
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	0	0	...	1	
4	Rajasthan	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	0	0	...	1	
5	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	0	0	...	1	
6	Rajasthan	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	0	0	...	1	

5 rows × 35 columns

In [33]:

```
df.shape
```

Out[33]:

```
(3664, 35)
```

In [34]:

```
df = df.rename(columns = {'State_Name':'State','District_Name':'District','Crop_Year':'Year'},inpla
ce=False)
```

In [35]:

```
df.head()
```

Out[35]:

	State	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	west north west lithosols	Alluvial deposites calcareous	Alluv pro wa loggi
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	0	0	...	1	0	
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	0	0	...	1	0	
4	Rajasthan	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	0	0	...	1	0	
5	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	0	0	...	1	0	
6	Rajasthan	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	0	0	...	1	0	

5 rows × 35 columns

In [36]:

```
df.Year.unique()
```

Out[36]:

```
array([1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
       2008, 2009, 2010, 2011, 2015, 2016, 2012, 2013, 2014, 2017, 2018,
       2019], dtype=int64)
```

In [37]:

```
len(df.District.unique())
```

Out[37]:

```
32
```

In [38]:

```
rainfall = pd.read_excel("Rainfall_1997_to_2019.xlsx")
rain = pd.DataFrame(rainfall)
rain.head()
```

Out[38]:

	State	District	Year	January	February	March	April	May	June	July	August	September	October	November	De
0	Rajasthan	Ajmer	1997	3.947	0.000	0.609	1.791	14.522	93.433	129.173	281.991	76.899	22.970	53.270	
1	Rajasthan	Ajmer	1998	0.034	15.187	2.397	23.993	0.956	157.776	58.179	117.911	193.063	35.003	7.943	
2	Rajasthan	Ajmer	1999	6.968	10.736	0.000	0.000	5.152	91.344	131.338	172.254	56.180	26.921	0.039	
3	Rajasthan	Ajmer	2000	0.347	5.932	0.142	1.483	14.242	14.651	327.974	37.004	10.270	0.000	3.826	
4	Rajasthan	Ajmer	2001	0.640	0.442	2.747	11.572	47.413	119.721	221.669	135.901	16.909	16.385	0.370	

In [39]:

```
rain.shape
```

Out[39]:

```
(733, 16)
```

In [40]:

```
rain.Year.unique()
```

Out[40]:

```
array([1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
       2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018,
       2019], dtype=int64)
```

In [41]:

```
rain.District.unique()
```

Out[41]:

```
array(['Ajmer', 'Alwar', 'Banswara', 'Bharatpur', 'Bhilwara', 'Bikaner',
       'Bundi', 'Chittorgarh', 'Churu', 'Dausa', 'Dholpur', 'Dungarpur',
       'Ganganagar', 'Jaisalmer', 'Jhalawar', 'Jhunjhunu', 'Jodhpur',
       'Karauli', 'Kota', 'Nagaur', 'Rajsamand', 'Sawai Madhopur',
       'Sikar', 'Sirohi', 'Tonk', 'Udaipur', 'Jaipur', 'Jalore', 'Barmer',
       'Baran', 'Hanumangarh', 'Pali'], dtype=object)
```

In [42]:

```
df.District.unique()
```

Out[42]:

```
array(['Ajmer', 'Jaipur', 'Dausa', 'Tonk', 'Sikar', 'Jhunjhunu', 'Nagaur',
       'Alwar', 'Bharatpur', 'Dholpur', 'Sawai Madhopur', 'Karauli',
       'Bikaner', 'Churu', 'Jaisalmer', 'Ganganagar', 'Hanumangarh',
       'Jodhpur', 'Barmer', 'Jalore', 'Pali', 'Sirohi', 'Kota', 'Baran',
       'Bundi', 'Jhalawar', 'Banswara', 'Dungarpur', 'Udaipur',
       'Bhilwara', 'Chittorgarh', 'Rajsamand'], dtype=object)
```

In [43]:

```
x = []
for i in df.District.unique():
    if i not in rain.District.unique():
        x.append(i)
x
```

Out[43]:

```
['Jaipur', 'Hanumangarh', 'Barmer', 'Jalore', 'Pali', 'Baran']
```

In [44]:

```
df.replace({'Jaipur':'Jaipur','Hanumangarh':'Hanumangarh','Barmer':'Barmer','Jalore':'Jalore',
            'Pali':'Pali','Baran':'Baran'},inplace=True)
```

In [45]:

```
x = []
for i in df.District.unique():
    if i not in rain.District.unique():
        x.append(i)
x
```

Out[45]:

```
[]
```

In [46]:

```
col = ['State','District','Year']
result = pd.merge(df,rain,on=col)
```

In [47]:

```
result.shape
```

Out[47]:

```
(3648, 48)
```

In [48]:

```
result.head(15)
```

Out[48]:

	State	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	April	May	June	
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	0	0	...	1.791	14.522	93.433	1
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	0	0	...	1.791	14.522	93.433	1
2	Rajasthan	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	0	0	...	1.791	14.522	93.433	1

	State	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	April	May	June	
3	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	0	0	...	1.791	14.522	93.433	1
4	Rajasthan	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	0	0	...	1.791	14.522	93.433	1
5	Rajasthan	Ajmer	1998	Kharif	Bajra	55089.0	6045.0	0.109732	0	0	...	23.993	0.956	157.776	
6	Rajasthan	Ajmer	1998	Kharif	Jowar	105177.0	6080.0	0.057807	0	0	...	23.993	0.956	157.776	
7	Rajasthan	Ajmer	1998	Rabi	Barley	21534.0	40167.0	1.865283	0	0	...	23.993	0.956	157.776	
8	Rajasthan	Ajmer	1998	Rabi	Rapeseed &Mustard	27203.0	13797.0	0.507187	0	0	...	23.993	0.956	157.776	
9	Rajasthan	Ajmer	1998	Rabi	Wheat	74805.0	134057.0	1.792086	0	0	...	23.993	0.956	157.776	
10	Rajasthan	Ajmer	1999	Kharif	Bajra	57959.0	5922.0	0.102176	0	0	...	0.000	5.152	91.344	1
11	Rajasthan	Ajmer	1999	Kharif	Jowar	112136.0	21196.0	0.189020	0	0	...	0.000	5.152	91.344	1
12	Rajasthan	Ajmer	1999	Rabi	Barley	12259.0	17582.0	1.434212	0	0	...	0.000	5.152	91.344	1
13	Rajasthan	Ajmer	1999	Rabi	Rapeseed &Mustard	41337.0	20544.0	0.496988	0	0	...	0.000	5.152	91.344	1
14	Rajasthan	Ajmer	1999	Rabi	Wheat	40412.0	41161.0	1.018534	0	0	...	0.000	5.152	91.344	1

15 rows × 48 columns

In [49]:

```
result.sort_values(['District','Year'],inplace=True)
```

In [50]:

```
result.to_excel("Final_Dataset_Minor_Project.xlsx",index=False)
```

In [51]:

```
result.District.value_counts()
```

Out[51]:

```
Dholpur           115
Pali              115
Tonk              115
Alwar             115
Sawai Madhopur    115
Bundi             115
Jhalawar          115
Jaipur            115
Ajmer             115
Bikaner           115
Nagaur            115
Bharatpur         115
Jaisalmer         115
Baran             115
Jalore            115
Bhilwara          115
Dausa             115
Jhunjhunu         114
Dungarpur         114
Udaipur           114
Sikar             114
Kota              114
Chittorgarh       114
Banswara          114
Sirohi            114
Ganganagar        114
Barmer            114
Jodhpur           114
Karauli           113
Hanumangarh       113
Churu             113
Rajsamand         100
Name: District, dtype: int64
```

In [52]:

```
result.Season.value_counts()
```

Out[52]:

```
Rabi      2198
Kharif    1450
Name: Season, dtype: int64
```

In [53]:

```
result.shape
```

Out[53]:

```
(3648, 48)
```

In [54]:

```
Seasonal_Rain = []
count = 0
for i in result["Season"]:

    if i=="Kharif":
        x = (result["July"]+result["August"]+result["September"]+result["October"])/4
        Seasonal_Rain.append(x[count])
        count+=1

    else:
        x =
(result["November"]+result["December"]+result["January"]+result["February"]+result["March"])/4
        Seasonal_Rain.append(x[count])
        count+=1
```

In [55]:

```
result["Mean_Seasonal_Rainfall"] = Seasonal_Rain
result["Mean_Seasonal_Rainfall"][0]
```

Out[55]:

```
127.75825
```

In [56]:

```
result.head()
```

Out[56]:

	State	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	May	June	July	A
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	0	0	...	14.522	93.433	129.173	28
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	0	0	...	14.522	93.433	129.173	28
2	Rajasthan	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	0	0	...	14.522	93.433	129.173	28
3	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	0	0	...	14.522	93.433	129.173	28
4	Rajasthan	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	0	0	...	14.522	93.433	129.173	28

5 rows × 49 columns

In [57]:

```
result.columns
```

Out[57]:

```
Index(['State', 'District', 'Year', 'Season', 'Crop', 'Area', 'Production',
       'Yield', ' Predominantly reddish medium texture', ' Red desert soils',
       ' Sandy loam', ' Soil are lithosolsat foot hills & alluvials in plains',
       ' brown soils', ' clay loam',
       ' coarse sand in texture some places calcareous',
       ' deep soils in valleys', ' eastern part alluvial', ' foot hills',
       ' groundwater salinity', ' high soluble salts & exchangeable sodium',
       ' loamycoarse in texture & calcareous',
       ' nature of recently alluvial calcareous has been observed',
       ' sallow depth red soils in depressions', ' shallow on hills',
       ' well drained calcareous', ' west north west lithosols',
       'Alluvial deposites calcareous', 'Alluvial prone to water logging',
       'Black of alluvial origin', 'Desert soils and sand dunes aeolian soil',
       'Predominantly reddish medium texture', 'Red desert soils',
       'Sandy loam', 'Sierozens',
       'Soil are lithosolsat foot hills & alluvials in plains', 'January',
       'February', 'March', 'April', 'May', 'June', 'July', 'August',
       'September', 'October', 'November', 'December', 'Annual Total',
       'Mean_Seasonal_Rainfall'],
      dtype='object')
```

In [58]:

```
col = ['State','January','February', 'March', 'April', 'May', 'June', 'July', 'August',
       'September', 'October', 'November', 'December', 'Annual Total']

result.drop(col, axis=1, inplace=True)
```

In [59]:

```
result.head(5)
```

Out[59]:

	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	Sandy loam	...	Alluvial deposites calcareous	Alluvial prone to water logging	Black of alluvial origin
0	Ajmer	1997	Kharif	Bajra	56600.0	30400.0	0.537102	0	0	0	...	0	0	0
1	Ajmer	1997	Kharif	Jowar	105900.0	34600.0	0.326723	0	0	0	...	0	0	0
2	Ajmer	1997	Rabi	Barley	24700.0	28900.0	1.170040	0	0	0	...	0	0	0
3	Ajmer	1997	Rabi	Rapeseed &Mustard	36700.0	25400.0	0.692098	0	0	0	...	0	0	0
4	Ajmer	1997	Rabi	Wheat	79300.0	144500.0	1.822194	0	0	0	...	0	0	0

5 rows × 35 columns

ENCODING THE DATASET

In [60]:

```
pred_data = pd.get_dummies(result)
pred_data.head(15)
```

Out[60]:

	Year	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	...	District_Baran	District_Hanuma
0	1997	56600.0	30400.0	0.537102	0	0	0	0	1	0	...	0	
1	1997	105900.0	34600.0	0.326723	0	0	0	0	1	0	...	0	

	Year	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	...	District_Baran	District_Hanuma
2	1997	24700.0	28900.0	1.170040	0	0	0	0	1	0	...	0	
3	1997	36700.0	25400.0	0.692098	0	0	0	0	1	0	...	0	
4	1997	79300.0	144500.0	1.822194	0	0	0	0	1	0	...	0	
5	1998	55089.0	6045.0	0.109732	0	0	0	0	1	0	...	0	
6	1998	105177.0	6080.0	0.057807	0	0	0	0	1	0	...	0	
7	1998	21534.0	40167.0	1.865283	0	0	0	0	1	0	...	0	
8	1998	27203.0	13797.0	0.507187	0	0	0	0	1	0	...	0	
9	1998	74805.0	134057.0	1.792086	0	0	0	0	1	0	...	0	
10	1999	57959.0	5922.0	0.102176	0	0	0	0	1	0	...	0	
11	1999	112136.0	21196.0	0.189020	0	0	0	0	1	0	...	0	
12	1999	12259.0	17582.0	1.434212	0	0	0	0	1	0	...	0	
13	1999	41337.0	20544.0	0.496988	0	0	0	0	1	0	...	0	
14	1999	40412.0	41161.0	1.018534	0	0	0	0	1	0	...	0	

15 rows × 71 columns

In [61]:

```
pred_data.to_excel("Dataset_for_Prediction.xlsx",index=False)
```

DATA ANALYSIS PLOTS

In [1]:

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

In [2]:

```
crop = pd.read_excel("Final_Dataset_Minor_Project.xlsx")
```

In [3]:

```
df = pd.DataFrame(crop)
```

In [4]:

```
df.head()
```

Out[4]:

	State	District	Year	Season	Crop	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	...	April	May	June	Jul
0	Rajasthan	Ajmer	1997	Kharif	Bajra	56600	30400.0	0.537102	0	0	...	1.791	14.522	93.433	129.17
1	Rajasthan	Ajmer	1997	Kharif	Jowar	105900	34600.0	0.326723	0	0	...	1.791	14.522	93.433	129.17
2	Rajasthan	Ajmer	1997	Rabi	Barley	24700	28900.0	1.170040	0	0	...	1.791	14.522	93.433	129.17
3	Rajasthan	Ajmer	1997	Rabi	Rapeseed &Mustard	36700	25400.0	0.692098	0	0	...	1.791	14.522	93.433	129.17
4	Rajasthan	Ajmer	1997	Rabi	Wheat	79300	144500.0	1.822194	0	0	...	1.791	14.522	93.433	129.17

5 rows × 48 columns

In [5]:

```
df.Crop.value_counts()
```

Out[5]:

```
Wheat                733
Rapeseed &Mustard    733
Barley               732
Bajra                726
Jowar                724
Name: Crop, dtype: int64
```

In [6]:

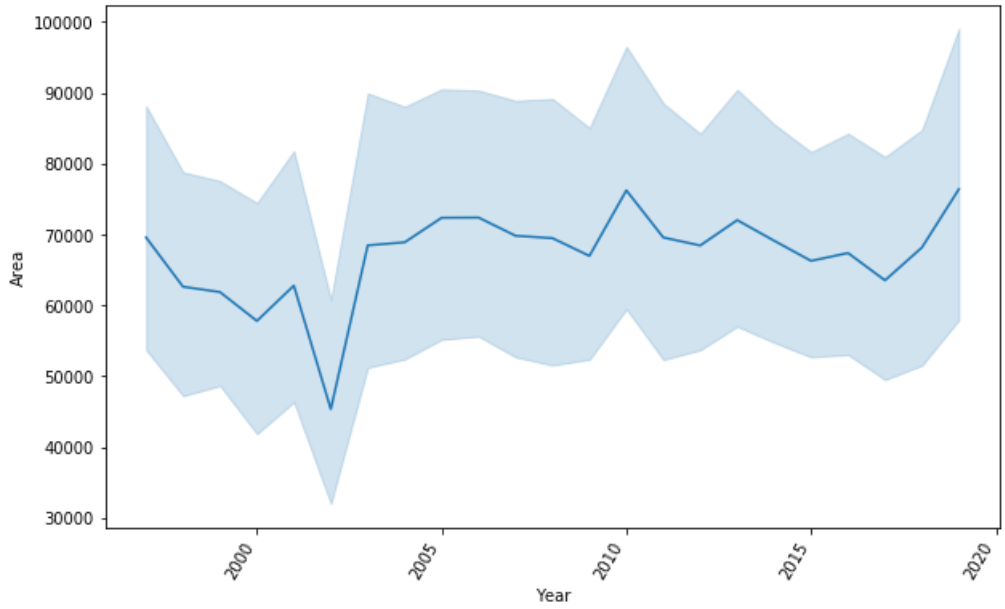
```
def data_graph (axis, width, height):
    axis.spines['top'].set_visible(False)
    axis.spines['right'].set_visible(False)
    for p in axis.patches:
        axis.annotate ("{0:.1f}".format(p.get_height()), (p.get_x()+width, p.get_height()+height))
```

TOTAL AREA UNDER IRRIGATION vs YEAR

In [7]:

```
plt.figure(figsize = (10,6))
```

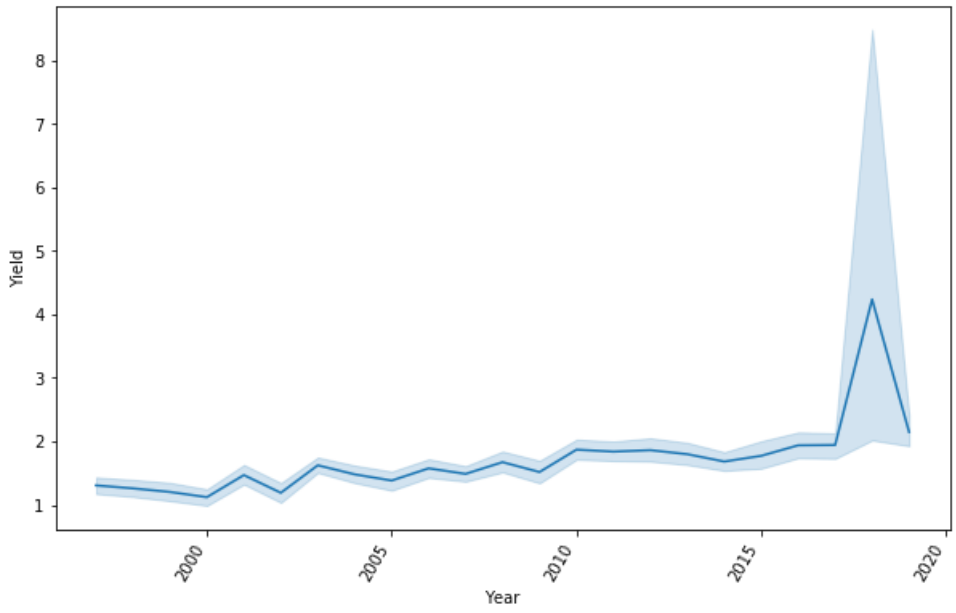
```
ax = sns.lineplot(x=df.Year, y=df.Area)
plt.xticks(rotation = 60, ha = "right")
plt.ylabel("Area")
plt.xlabel("Year")

plt.show()
```



TOTAL YIELD vs YEAR

In [8]:

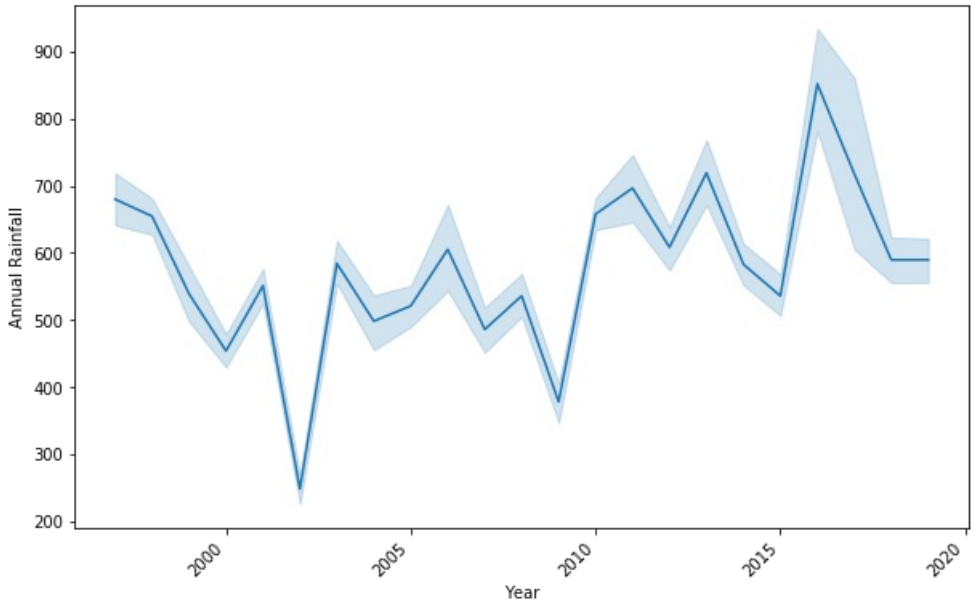
```
plt.figure(figsize = (10,6))
ax = sns.lineplot(x=df.Year, y=df.Yield)
plt.xticks(rotation = 60, ha = "right")
plt.ylabel("Yield")
plt.xlabel("Year")

plt.show()
```



ANNUAL TOTAL RAINFALL vs YEAR

In [9]:

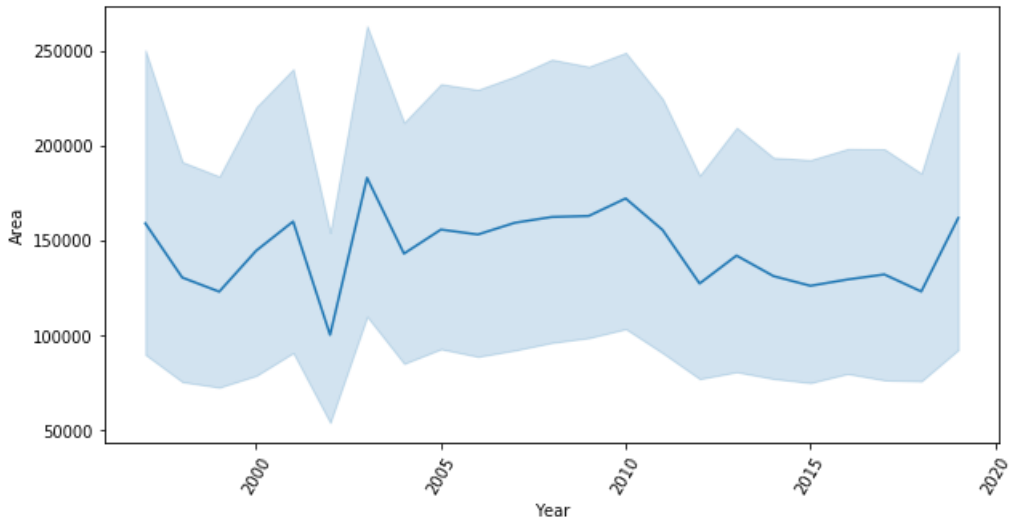
```
plt.figure(figsize = (10,6))
ax = sns.lineplot(x=df.Year, y=df["Annual Total"])
plt.xticks(rotation = 45, ha = "right")
plt.ylabel("Annual Rainfall")
plt.xlabel("Year")

plt.show()
```



PLOTS FOR BAJRA

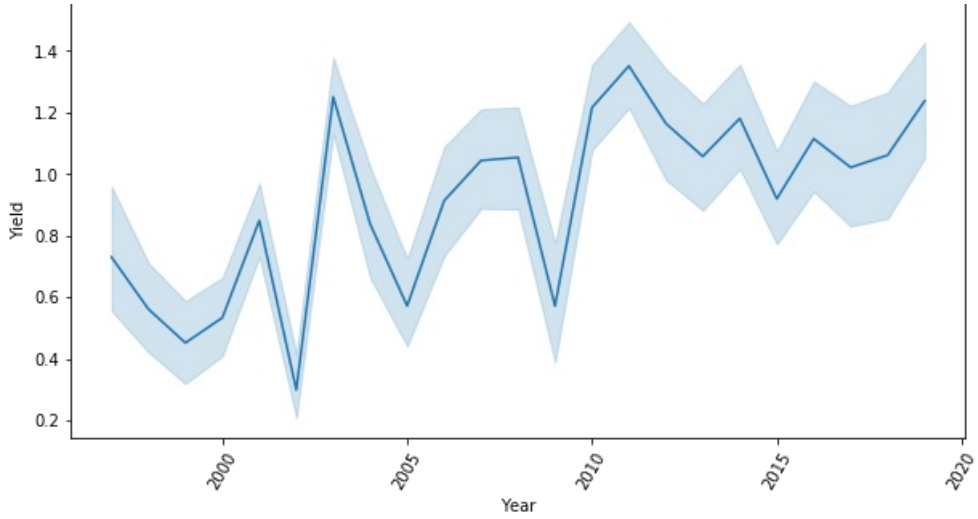
In [10]:

```
plt.figure(figsize=(10,5))
bajra_df = df[df["Crop"]=="Bajra"]
sns.lineplot("Year","Area",data=bajra_df)
plt.xticks(rotation=60)
plt.show()
```



In [11]:

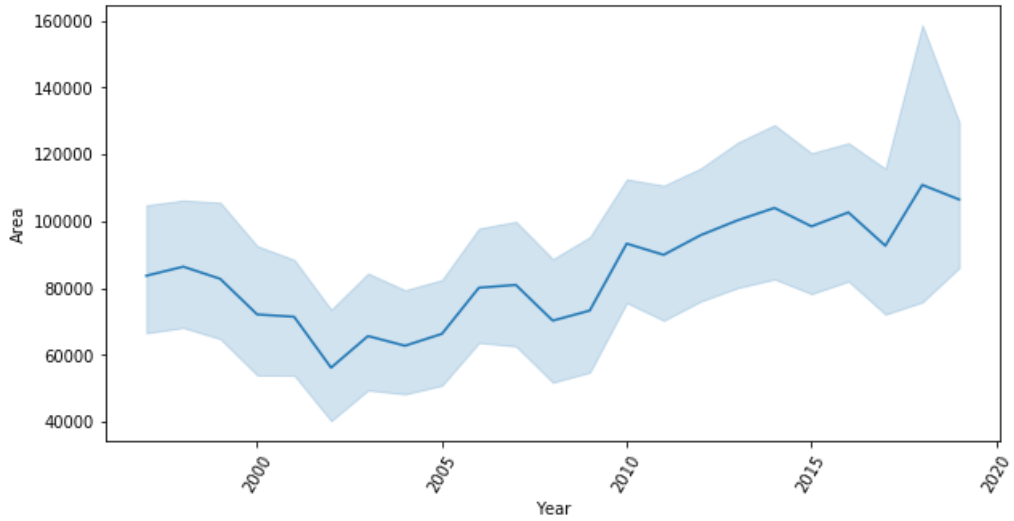
```
plt.figure(figsize=(10,5))
bajra_df = df[df["Crop"]=="Bajra"]
sns.lineplot("Year","Yield",data=bajra_df)
plt.xticks(rotation=60)
plt.show()
```



PLOTS FOR WHEAT

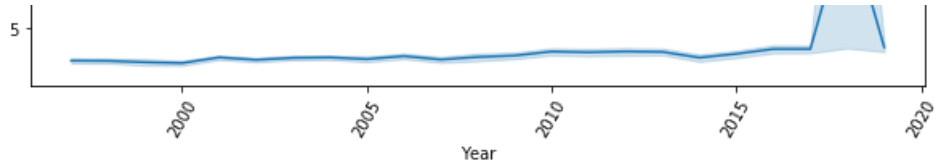
In [12]:

```
plt.figure(figsize=(10,5))
wheat_df = df[df["Crop"]=="Wheat"]
sns.lineplot("Year","Area",data=wheat_df)
plt.xticks(rotation=60)
plt.show()
```



In [13]:

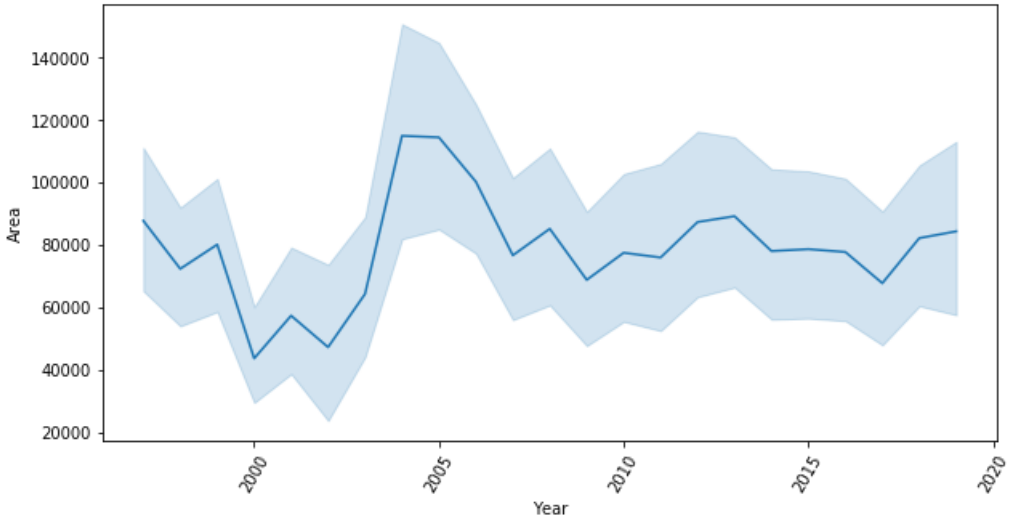
```
plt.figure(figsize=(10,5))
wheat_df = df[df["Crop"]=="Wheat"]
sns.lineplot("Year","Yield",data=wheat_df)
plt.xticks(rotation=60)
plt.show()
```



PLOTS FOR RAPESEED AND MUSTARD

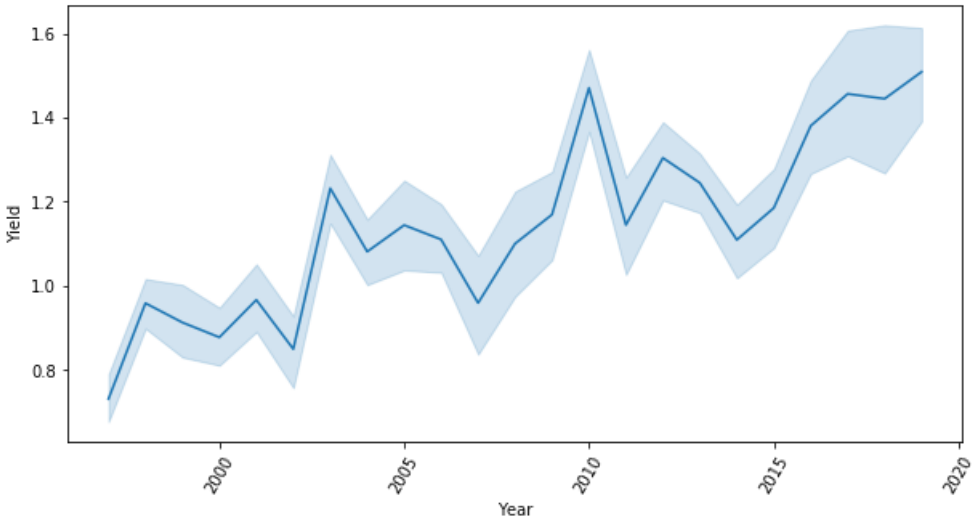
In [14]:

```
plt.figure(figsize=(10,5))
rnm_df = df[df["Crop"]=="Rapeseed &Mustard"]
sns.lineplot("Year","Area",data=rnm_df)
plt.xticks(rotation=60)
plt.show()
```



In [15]:

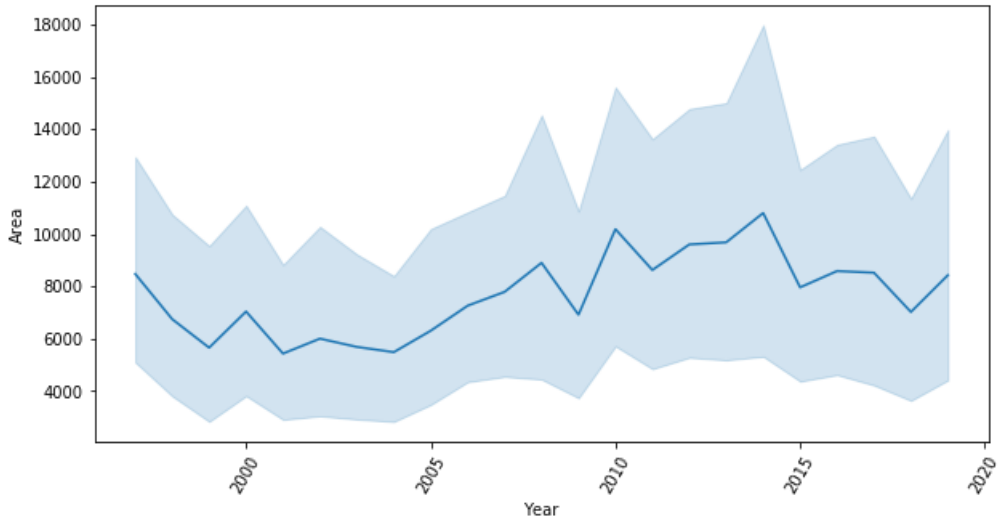
```
plt.figure(figsize=(10,5))
rnm_df = df[df["Crop"]=="Rapeseed &Mustard"]
sns.lineplot("Year","Yield",data=rnm_df)
plt.xticks(rotation=60)
plt.show()
```



PLOTS FOR BARLEY

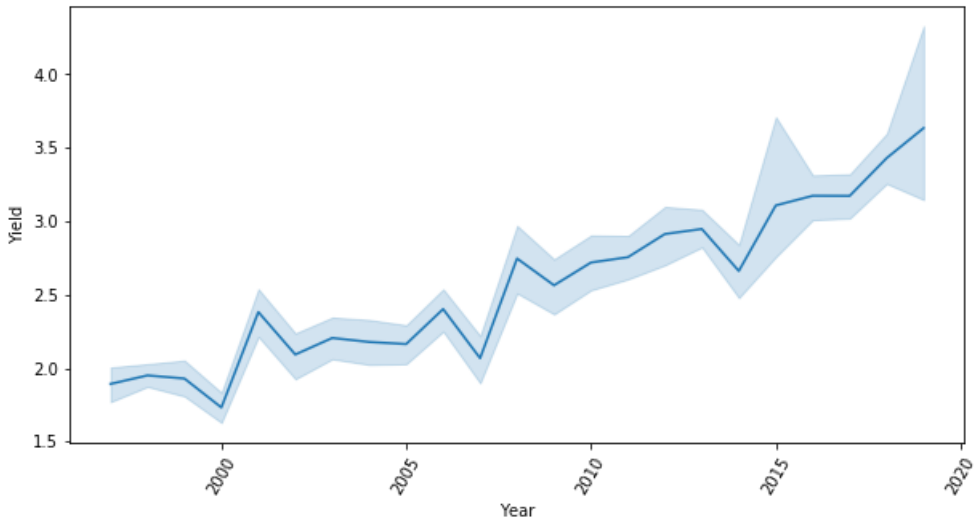
In [16]:

```
plt.figure(figsize=(10,5))
barley_df = df[df["Crop"]=="Barley"]
sns.lineplot("Year","Area",data=barley_df)
plt.xticks(rotation=60)
plt.show()
```



In [17]:

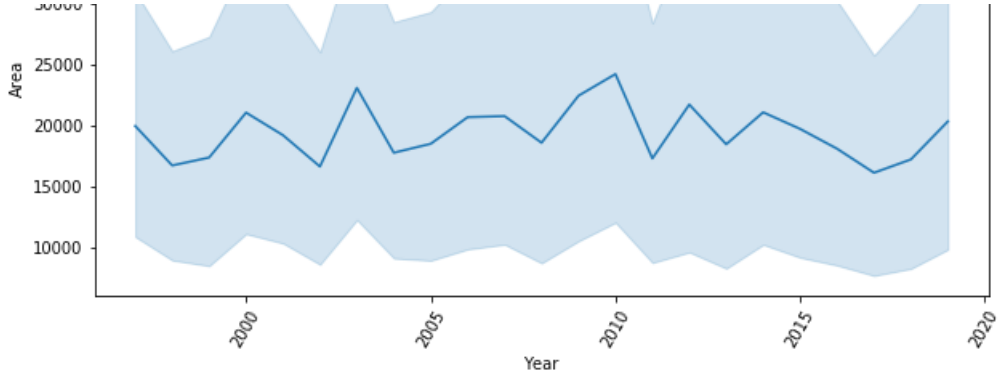
```
plt.figure(figsize=(10,5))
barley_df = df[df["Crop"]=="Barley"]
sns.lineplot("Year","Yield",data=barley_df)
plt.xticks(rotation=60)
plt.show()
```



PLOTS FOR JOWAR

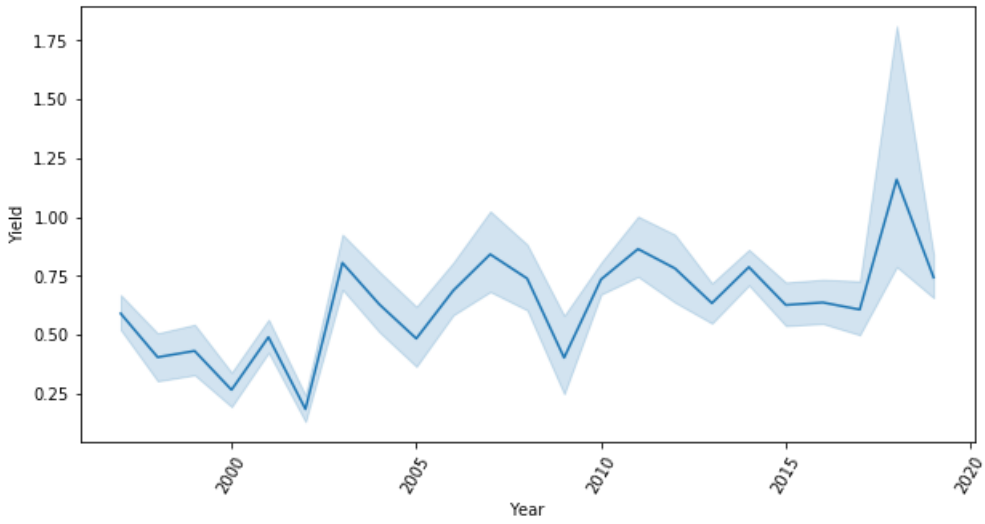
In [18]:

```
plt.figure(figsize=(10,5))
jowar_df = df[df["Crop"]=="Jowar"]
sns.lineplot("Year","Area",data=jowar_df)
plt.xticks(rotation=60)
plt.show()
```



In [19]:

```
plt.figure(figsize=(10,5))
jowar_df = df[df["Crop"]=="Jowar"]
sns.lineplot("Year","Yield",data=jowar_df)
plt.xticks(rotation=60)
plt.show()
```



MACHINE LEARNING ANALYSIS

In [1]:

```
import pandas as pd
import numpy as np
data = pd.read_excel("Dataset_for_Prediction.xlsx")
data.head()
```

Out[1]:

	Year	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	...	District_Baran	District_Hanumanga
0	1997	56600	30400.0	0.537102	0	0	0	0	1	0	...	0	
1	1997	105900	34600.0	0.326723	0	0	0	0	1	0	...	0	
2	1997	24700	28900.0	1.170040	0	0	0	0	1	0	...	0	
3	1997	36700	25400.0	0.692098	0	0	0	0	1	0	...	0	
4	1997	79300	144500.0	1.822194	0	0	0	0	1	0	...	0	

5 rows × 71 columns

In [2]:

```
data.isnull().sum()
```

Out[2]:

```
Year                                    0
Area                                    0
Production                              9
Yield                                  20
 Predominantly reddish medium texture   0
                                       ..
Crop_Bajra                              0
Crop_Barley                             0
Crop_Jowar                              0
Crop_Rapeseed &Mustard                  0
Crop_Wheat                              0
Length: 71, dtype: int64
```

In [3]:

```
data = data.dropna()
```

In [4]:

```
#all parameters
x = data.drop("Yield",axis=1)
y = data[["Yield"]]

cols = ['Yield',' Predominantly reddish medium texture', ' Red desert soils',
       ' Sandy loam', ' Soil are lithosolsat foot hills & alluvials in plains',
       ' brown soils', ' clay loam',
       ' coarse sand in texture some places calcareous',
       ' deep soils in valleys', ' eastern part alluvial', ' foot hills',
       ' groundwater salinity', ' high soluble salts & exchangeable sodium',
       ' loamycoarse in texture & calcareous',
       ' nature of recently alluvial calcareous has been observed',
       ' sallow depth red soils in depressions', ' shallow on hills',
       ' well drained calcareous', ' west north west lithosols',
       'Alluvial deposites calcareous', 'Alluvial prone to water logging',
       'Black of alluvial origin', 'Desert soils and sand dunes aeolian soil',
       'Predominantly reddish medium texture', 'Red desert soils',
```

```
        'Sandy loam', 'Sierozens',
        'Soil are lithosolsat foot hills & alluvials in plains',
        'Mean_Seasonal_Rainfall']

#selected parameters
x1 = data.drop(cols,axis=1)

from sklearn.preprocessing import StandardScaler
std = StandardScaler()
x = std.fit_transform(x)
y = std.fit_transform(y)
x1 = std.fit_transform(x1)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.02, random_state=71)
x1_train, x1_test, y_train, y_test = train_test_split(x1, y, test_size=0.02, random_state=71)
```

RANDOM FOREST

USING ALL PARAMETERS

In [5]:

```
#Using all parameters
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
model.fit(x_train,y_train)
pred_all = model.predict(x_test)
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

print("R2score:", r2_score(y_test,pred_all))
print('RMSE:', np.sqrt(mean_squared_error(y_test, pred_all)))
print('MAE:', mean_absolute_error(y_test, pred_all))
```

```
C:\Users\Sanchit\anaconda3\lib\site-packages\ipykernel_launcher.py:4: DataConversionWarning: A col
umn-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  after removing the cwd from sys.path.
```

```
R2score: 0.9638436515081396
RMSE: 0.03564416484020771
MAE: 0.025122868659073126
```

USING SELECTED PARAMETERS

In [7]:

```
#using selected parameters
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
model.fit(x1_train,y_train)
pred_selected = model.predict(x1_test)
from sklearn.metrics import r2_score
r = r2_score(y_test,pred_selected)
print("R2score: ",r)
print('RMSE:',np.sqrt(mean_squared_error(y_test, pred_selected)))
print('MAE:',mean_absolute_error(y_test, pred_selected))
```

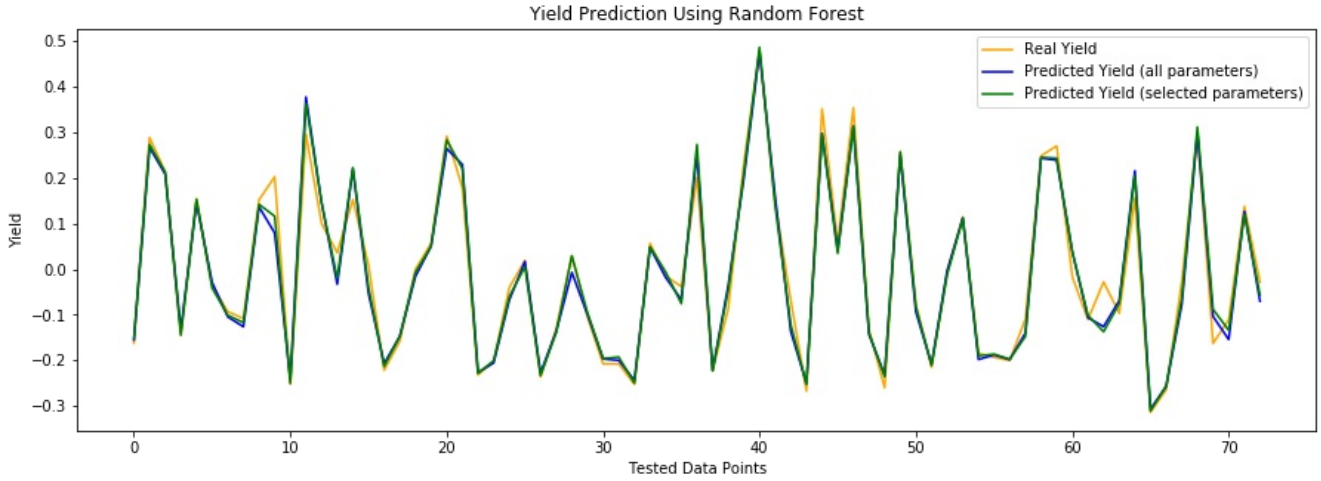
```
C:\Users\Sanchit\anaconda3\lib\site-packages\ipykernel_launcher.py:4: DataConversionWarning: A col
umn-vector y was passed when a 1d array was expected. Please change the shape of y to
(n_samples,), for example using ravel().
  after removing the cwd from sys.path.
```

```
R2score:  0.9706683562457551
RMSE: 0.03210438555877871
MAE: 0.02171092742447838
```

PERFORMANCE COMPARISON PLOT

In [8]:

```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
plt.plot(y_test, color='orange',label='Real Yield')
plt.plot(pred_all, color='blue',label='Predicted Yield (all parameters)')
plt.plot(pred_selected, color='green',label='Predicted Yield (selected parameters)')
plt.title('Yield Prediction Using Random Forest')
plt.xlabel("Tested Data Points")
plt.ylabel('Yield')
plt.legend()
plt.show()
```



SUPPORT VECTOR MACHINE

USING ALL PARAMETERS

In [9]:

```
#using all parameters
from sklearn.svm import SVR
from sklearn import metrics
svr=SVR() #Default hyperparameters
svr.fit(x_train,y_train)
pred_all=svr.predict(x_test)
#print('r2 Score:')
print('R2 score:',r2_score(y_test, pred_all))
print('RMSE:',np.sqrt(mean_squared_error(y_test, pred_all)))
print('MAE:',mean_absolute_error(y_test, pred_all))
```

```
R2 score: 0.8982433161230687
RMSE: 0.059796757425198625
MAE: 0.047852253033345137
```

```
C:\Users\Sanchit\anaconda3\lib\site-packages\sklearn\utils\validation.py:760:
DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

USING SELECTED PARAMETERS

In [10]:

```
#using selected parameters
from sklearn.svm import SVR
from sklearn import metrics
```

```
svr=SVR() #Default hyperparameters
svr.fit(x1_train,y_train)
pred_sel = svr.predict(x1_test)
#print('r2 Score:')
print('R2 score:',r2_score(y_test,pred_sel))
print('RMSE:',np.sqrt(mean_squared_error(y_test, pred_sel)))
print('MAE:',mean_absolute_error(y_test, pred_sel))
```

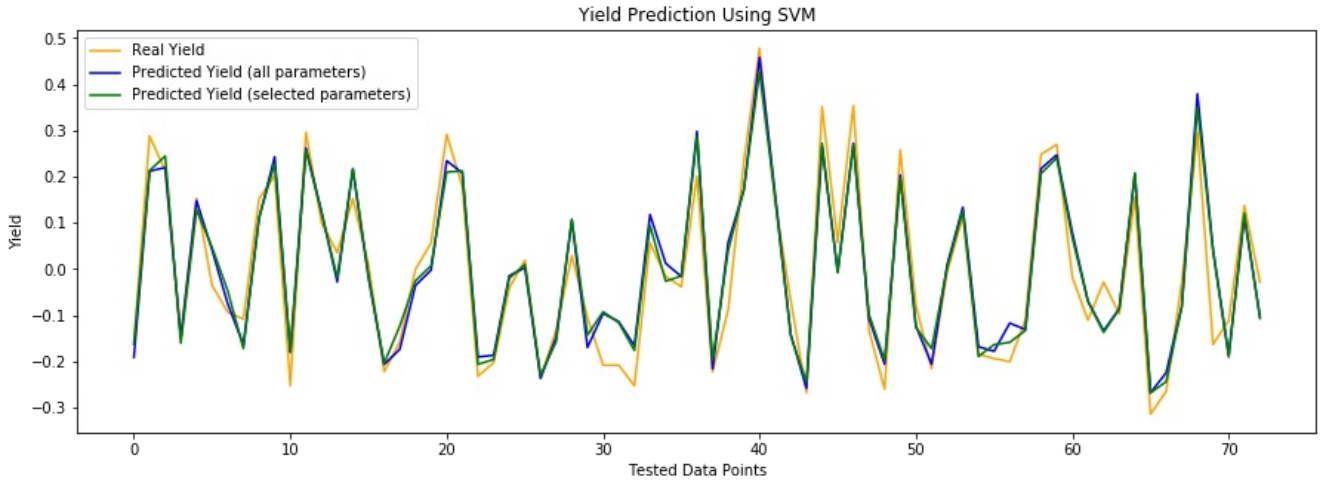
```
R2 score: 0.9033044480088029
RMSE: 0.0582907187068775
MAE: 0.04744273781182322
```

```
C:\Users\Sanchit\anaconda3\lib\site-packages\sklearn\utils\validation.py:760:
DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change th
e shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

PERFORMANCE COMPARISON ANALYSIS

In [11]:

```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
plt.plot(y_test, color='orange',label='Real Yield')
plt.plot(pred_all, color='blue',label='Predicted Yield (all parameters)')
plt.plot(pred_sel, color='green',label='Predicted Yield (selected parameters)')
plt.title('Yield Prediction Using SVM')
plt.xlabel("Tested Data Points")
plt.ylabel('Yield')
plt.legend()
plt.show()
```



LASSO REGRESSION

USING ALL PARAMETERS

In [12]:

```
#using all parameters
from sklearn.linear_model import Lasso

model_lasso = Lasso(alpha=0.01)
model_lasso.fit(x_train, y_train)
pred_all= model_lasso.predict(x_test)

print('R2 score:',r2_score(y_test, pred_all))
print('RMSE:',np.sqrt(mean_squared_error(y_test,pred_all)))
print('MAE:',mean_absolute_error(y_test, pred_all))
```

```
R2 score: 0.81468242122217434
RMSE: 0.08069645755381685
MAE: 0.0588093331954562
```

USING SELECTED PARAMETERS

In [13]:

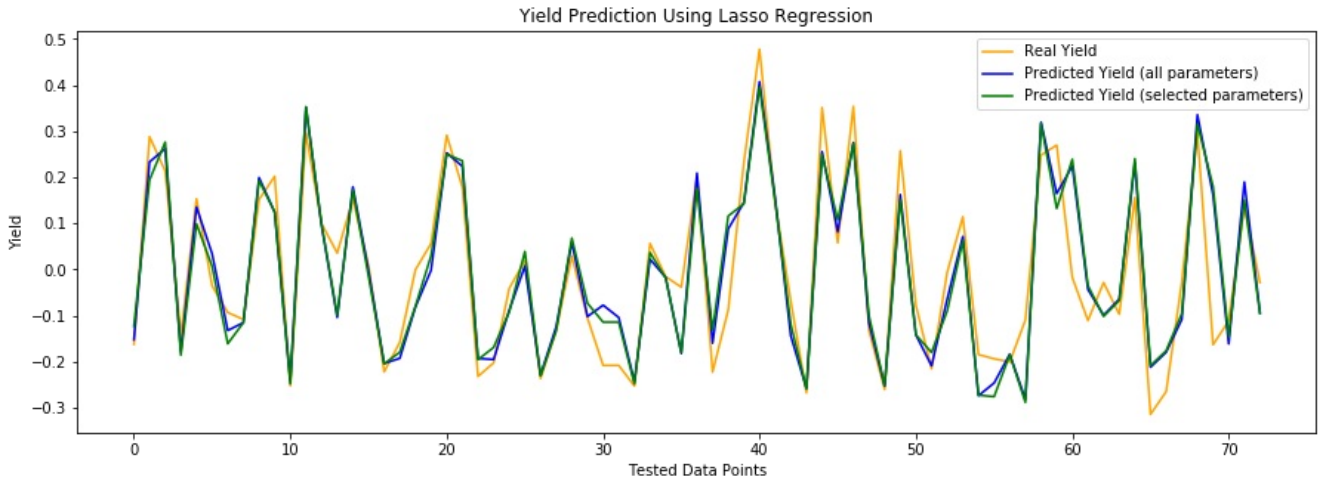
```
#using selected parameters
from sklearn.linear_model import Lasso

model_lasso = Lasso(alpha=0.01)
model_lasso.fit(x1_train, y_train)
pred_sel= model_lasso.predict(x1_test)

print('R2 score:',r2_score(y_test, pred_sel))
print('RMSE:',np.sqrt(mean_squared_error(y_test,pred_sel)))
print('MAE:',mean_absolute_error(y_test, pred_sel))
```

```
R2 score: 0.7929212370102475
RMSE: 0.08530292881366584
MAE: 0.06279053471454211
```

PERFORMANCE COMPARISON ANALYSIS

In [14]:

```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
plt.plot(y_test, color='orange',label='Real Yield')
plt.plot(pred_all, color='blue',label='Predicted Yield (all parameters)')
plt.plot(pred_sel, color='green',label='Predicted Yield (selected parameters)')
plt.title('Yield Prediction Using Lasso Regression')
plt.xlabel("Tested Data Points")
plt.ylabel('Yield')
plt.legend()
plt.show()
```



GRADIENT DESCENT ALGORITHM

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
data = pd.DataFrame(pd.read_excel("Dataset_for_Prediction.xlsx"))
data.head()
```

Out[2]:

	Year	Area	Production	Yield	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	...	District_Baran	District_Hanumanga
0	1997	56600	30400.0	0.537102	0	0	0	0	1	0	...	0	
1	1997	105900	34600.0	0.326723	0	0	0	0	1	0	...	0	
2	1997	24700	28900.0	1.170040	0	0	0	0	1	0	...	0	
3	1997	36700	25400.0	0.692098	0	0	0	0	1	0	...	0	
4	1997	79300	144500.0	1.822194	0	0	0	0	1	0	...	0	

5 rows × 71 columns

In [3]:

```
data.isnull().sum()
```

Out[3]:

```
Year                                    0
Area                                    0
Production                              9
Yield                                  20
 Predominantly reddish medium texture   0
                                       ..
Crop_Bajra                              0
Crop_Barley                             0
Crop_Jowar                              0
Crop_Rapeseed &Mustard                  0
Crop_Wheat                              0
Length: 71, dtype: int64
```

In [4]:

```
data.dropna(inplace=True)
```

In [5]:

```
x = data.drop("Yield",axis=1)
y = data[["Yield"]]

cols = ['Yield',' Predominantly reddish medium texture', ' Red desert soils',
       ' Sandy loam', ' Soil are lithosolsat foot hills & alluvials in plains',
       ' brown soils', ' clay loam',
       ' coarse sand in texture some places calcareous',
       ' deep soils in valleys', ' eastern part alluvial', ' foot hills',
       ' groundwater salinity', ' high soluble salts & exchangeable sodium',
       ' loamycoarse in texture & calcareous',
       ' nature of recently alluvial calcareous has been observed',
       ' sallow depth red soils in depressions', ' shallow on hills',
       ' well drained calcareous', ' west north west lithosols',
```

```
        'Alluvial deposites calcareous', 'Alluvial prone to water logging',
        'Black of alluvial origin', 'Desert soils and sand dunes aeolian soil',
        'Predominantly reddish medium texture', 'Red desert soils',
        'Sandy loam', 'Sierozens',
        'Soil are lithosolsat foot hills & alluvials in plains',
        'Mean_Seasonal_Rainfall']

#selected parameters
x1 = data.drop(cols,axis=1)

from sklearn.preprocessing import StandardScaler
std = StandardScaler()
x = std.fit_transform(x)
y = std.fit_transform(y)
x1 = std.fit_transform(x1)

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.02, random_state=71)
x1_train, x1_test, y_train, y_test = train_test_split(x1, y, test_size=0.02, random_state=71)
```

ANALYSIS USING ALL PARAMETERS

In [6]:

```
x_train = x_train.reshape(x_train.shape[0],-1).T
x_test =  x_test.reshape(x_test.shape[0],-1).T
y_train = y_train.reshape(y_train.shape[0],-1).T
```

In [7]:

```
def model(x,y,dim,learning_rate):
    w = np.zeros((dim,1))
    b = 0.0

    m = x.shape[1]
    A = np.dot(w.T,x)+b
    cost = (1/(m))*np.sum((A-y)**2)

    dz = (1/m)*(A-y)
    dw = np.dot(x,dz.T)
    db = np.sum(dz)

    costs = []

    for i in range (200):

        w = w - (learning_rate*dw)
        b = b - (learning_rate*db)

        if i%1000 == 0:
            costs.append(cost)

    parameters = {"w":w,"b":b}
    gradients = {"dw":dw,"db":db}

    return parameters, gradients, cost
```

In [8]:

```
def pred (w,b,x):
    m = x.shape[1]
    y_pred = np.zeros((1,m))
    w = w.reshape(x.shape[0],1)

    y_pred = np.dot(w.T,x)+b

    return y_pred
```

In [9]:

```
parameters, gradients, cost = model(x_train,y_train,70,0.001)
```

```
cost
```

Out[9]:

```
1.0198119517753892
```

In [10]:

```
w = parameters["w"]
b = parameters["b"]
```

In [11]:

```
pred_all = pred(w,b,x_test)
```

In [13]:

```
pred_all = pred_all.reshape(pred_all.shape[0],-1).T
print(pred_all.shape)
print(y_test.shape)
```

```
(73, 1)
(73, 1)
```

In [14]:

```
auc_test = 100 - np.mean(np.abs(pred_all - y_test)*100)
auc_test
```

Out[14]:

```
92.09301584199703
```

In [15]:

```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

print("R2score:", r2_score(y_test,pred_all))
print('RMSE:', np.sqrt(mean_squared_error(y_test, pred_all)))
print('MAE:', mean_absolute_error(y_test, pred_all))
```

```
R2score: 0.737347767385024
RMSE: 0.09606976093705671
MAE: 0.07906984158002965
```

ANALYSIS USING SELECTED PARAMETERS

In [16]:

```
x1_train = x1_train.reshape(x1_train.shape[0],-1).T
x1_test =  x1_test.reshape(x1_test.shape[0],-1).T
```

In [19]:

```
parameters, gradients, cost = model(x1_train,y_train,42,0.001)
cost
```

Out[19]:

```
1.0198119517753892
```

In [20]:

```
w = parameters["w"]
b = parameters["b"]
```

In [21]:

```
pred_sel = pred(w,b,x1_test)
pred_sel = pred_sel.reshape(pred_sel.shape[0],-1).T
```

In [22]:

```
auc_test1 = 100 - np.mean(np.abs(pred_sel - y_test)*100)
auc_test1
```

Out[22]:

```
91.16665107775859
```

In [23]:

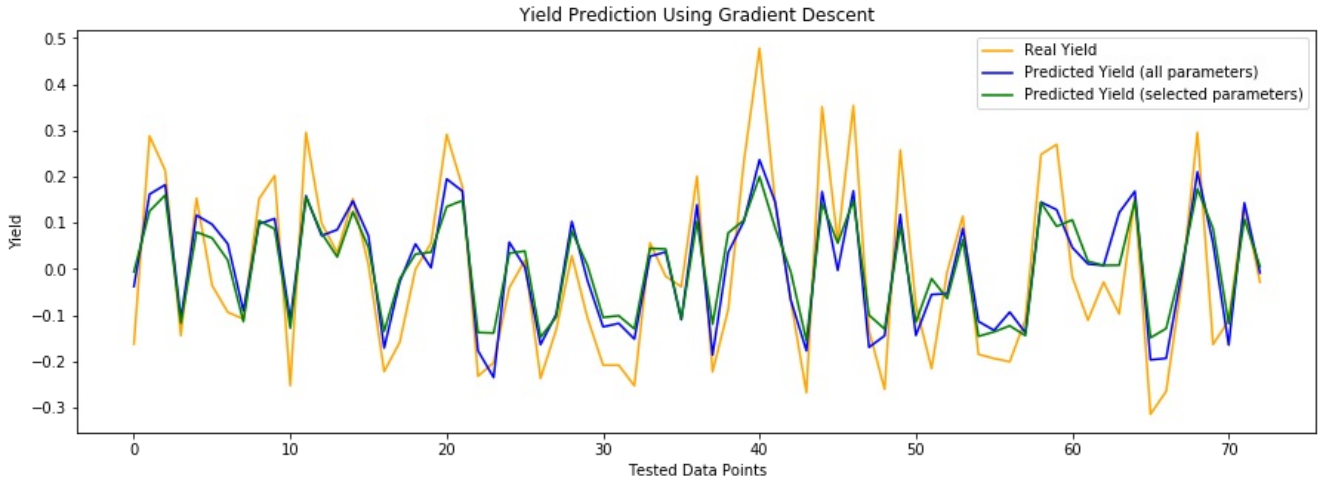
```
from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

print("R2score:", r2_score(y_test,pred_sel))
print('RMSE:', np.sqrt(mean_squared_error(y_test, pred_sel)))
print('MAE:', mean_absolute_error(y_test, pred_sel))
```

```
R2score: 0.6721446803333357
RMSE: 0.10733399123627024
MAE: 0.08833348922241419
```

PERFORMANCE COMPARISON PLOT

In [25]:

```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
plt.plot(y_test, color='orange',label='Real Yield')
plt.plot(pred_all, color='blue',label='Predicted Yield (all parameters)')
plt.plot(pred_sel, color='green',label='Predicted Yield (selected parameters)')
plt.title('Yield Prediction Using Gradient Descent')
plt.xlabel("Tested Data Points")
plt.ylabel('Yield')
plt.legend()
plt.show()
```



In [ ]:

LONG SHORT TERM MEMORY (LSTM)

In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
df = pd.read_excel("HeatMap_Dataset.xlsx", engine='openpyxl')
```

In [3]:

```
df.head()
```

Out[3]:

	District	Year	Season	Crop	Area	Production	Yield	Soil_Type	Mean_Seasonal_Rainfall
0	Ajmer	1997	Kharif	Bajra	56600	30400.0	0.537102	Sierozens, eastern part alluvial, west north w...	127.75825
1	Ajmer	1997	Kharif	Jowar	105900	34600.0	0.326723	Sierozens, eastern part alluvial, west north w...	127.75825
2	Ajmer	1997	Rabi	Barley	24700	28900.0	1.170040	Sierozens, eastern part alluvial, west north w...	16.10950
3	Ajmer	1997	Rabi	Rapeseed &Mustard	36700	25400.0	0.692098	Sierozens, eastern part alluvial, west north w...	16.10950
4	Ajmer	1997	Rabi	Wheat	79300	144500.0	1.822194	Sierozens, eastern part alluvial, west north w...	16.10950

In [4]:

```
df.isnull().sum()
```

Out[4]:

```
District                  0
Year                      0
Season                    0
Crop                      0
Area                      0
Production                9
Yield                    20
Soil_Type                 0
Mean_Seasonal_Rainfall    0
dtype: int64
```

In [5]:

```
df.dropna(inplace=True)
```

In [6]:

```
df.isnull().sum()
```

Out[6]:

```
District                  0
Year                      0
Season                    0
Crop                      0
Area                      0
Production                0
Yield                     0
Soil_Type                 0
Mean_Seasonal_Rainfall    0
dtype: int64
```

In [7]:

```
df1 = df['Soil_Type'].str.get_dummies(sep=',')
```

In [8]:

```
df1.head()
```

Out[8]:

	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	clay loam	coarse sand in texture some places calcareous	deep soils in valleys	eastern part alluvial	foot hills	...	west north west lithosols	Alluvial deposites calcareous	Alluvial prone to water logging	Black of alluvial origin	Desert soils and sand dunes aeolian soil
0	0	0	0	0	1	0	0	0	1	1	...	1	0	0	0	0
1	0	0	0	0	1	0	0	0	1	1	...	1	0	0	0	0
2	0	0	0	0	1	0	0	0	1	1	...	1	0	0	0	0
3	0	0	0	0	1	0	0	0	1	1	...	1	0	0	0	0
4	0	0	0	0	1	0	0	0	1	1	...	1	0	0	0	0

5 rows × 27 columns

In [9]:

```
frames = [df,df1]
df = pd.concat(frames,axis=1)
df.head()
```

Out[9]:

	District	Year	Season	Crop	Area	Production	Yield	Soil_Type	Mean_Seasonal_Rainfall	Predominantly reddish medium texture	...	west north west lithosols	Alluvial deposites calcareous
0	Ajmer	1997	Kharif	Bajra	56600	30400.0	0.537102	Sierozens, eastern part alluvial, west north w...	127.75825	0	...	1	0
1	Ajmer	1997	Kharif	Jowar	105900	34600.0	0.326723	Sierozens, eastern part alluvial, west north w...	127.75825	0	...	1	0
2	Ajmer	1997	Rabi	Barley	24700	28900.0	1.170040	Sierozens, eastern part alluvial, west north w...	16.10950	0	...	1	0
3	Ajmer	1997	Rabi	Rapeseed &Mustard	36700	25400.0	0.692098	Sierozens, eastern part alluvial, west north w...	16.10950	0	...	1	0
4	Ajmer	1997	Rabi	Wheat	79300	144500.0	1.822194	Sierozens, eastern part alluvial, west north w...	16.10950	0	...	1	0

5 rows × 36 columns

In [10]:

```
df = df.drop("Soil_Type",axis=1)
```

In [11]:

```
df.head()
```

Out[11]:

	District	Year	Season	Crop	Area	Production	Yield	Mean_Seasonal_Rainfall	Predominantly reddish medium texture	Red desert soils	...	west north west lithosols	Alluvial deposites calcareous	Allu pr wa logg
0	Ajmer	1997	Kharif	Bajra	56600	30400.0	0.537102	127.75825	0	0	...	1	0	
1	Ajmer	1997	Kharif	Jowar	105900	34600.0	0.326723	127.75825	0	0	...	1	0	
2	Ajmer	1997	Rabi	Barley	24700	28900.0	1.170040	16.10950	0	0	...	1	0	
3	Ajmer	1997	Rabi	Rapeseed &Mustard	36700	25400.0	0.692098	16.10950	0	0	...	1	0	
4	Ajmer	1997	Rabi	Wheat	79300	144500.0	1.822194	16.10950	0	0	...	1	0	

5 rows × 35 columns

In [12]:

```
df = pd.get_dummies(df)
```

In [13]:

```
df.head()
```

Out[13]:

	Year	Area	Production	Yield	Mean_Seasonal_Rainfall	Predominantly reddish medium texture	Red desert soils	Sandy loam	Soil are lithosolsat foot hills & alluvials in plains	brown soils	...	District_Baran	District_Ha
0	1997	56600	30400.0	0.537102	127.75825	0	0	0	0	1	...	0	
1	1997	105900	34600.0	0.326723	127.75825	0	0	0	0	1	...	0	
2	1997	24700	28900.0	1.170040	16.10950	0	0	0	0	1	...	0	
3	1997	36700	25400.0	0.692098	16.10950	0	0	0	0	1	...	0	
4	1997	79300	144500.0	1.822194	16.10950	0	0	0	0	1	...	0	

5 rows × 71 columns

USING ALL PARAMETERS

In [14]:

```
x = np.array(df.drop("Yield",axis=1))
y = np.array(df[["Yield"]])
```

In [15]:

```
from sklearn.preprocessing import StandardScaler
std = StandardScaler()
x = std.fit_transform(x)
y = std.fit_transform(y)
```

In [16]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.01, random_state=71)
print("x_train :",x_train.shape)
print("x_test :",x_test.shape)
print("y_train :",y_train.shape)
print("y_test :",y_test.shape)
```

```
x_train : (3591, 70)
x_test : (37, 70)
y_train : (3591, 1)
y_test : (37, 1)
```

In [17]:

```
x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))
```

In [18]:

```
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
import math
```

In [19]:

```
reg = Sequential()

reg.add(LSTM(units=50, return_sequences=True, input_shape=(x_train.shape[1],1)))
reg.add(Dropout(0.2))

reg.add(LSTM(units=50, return_sequences=True))
reg.add(Dropout(0.2))

reg.add(LSTM(units=50, return_sequences=True))
reg.add(Dropout(0.2))

reg.add(LSTM(units=50))
reg.add(Dropout(0.2))

reg.add(Dense(units=1))

reg.compile(optimizer='adam', loss='mean_squared_error')

reg.fit(x_train,y_train,epochs=50,batch_size=64)
```

```
Epoch 1/50
57/57 [==============================] - 10s 181ms/step - loss: 1.0078
Epoch 2/50
57/57 [==============================] - 10s 179ms/step - loss: 1.0071
Epoch 3/50
57/57 [==============================] - 12s 206ms/step - loss: 1.0092
Epoch 4/50
57/57 [==============================] - 13s 230ms/step - loss: 1.0080
Epoch 5/50
57/57 [==============================] - 13s 228ms/step - loss: 1.0063
Epoch 6/50
57/57 [==============================] - 10s 180ms/step - loss: 1.0044
Epoch 7/50
57/57 [==============================] - 9s 164ms/step - loss: 1.0050
Epoch 8/50
57/57 [==============================] - 12s 203ms/step - loss: 1.0032
Epoch 9/50
57/57 [==============================] - 10s 174ms/step - loss: 0.9977
Epoch 10/50
57/57 [==============================] - 10s 180ms/step - loss: 1.0086
Epoch 11/50
57/57 [==============================] - 10s 181ms/step - loss: 1.0013
Epoch 12/50
57/57 [==============================] - 9s 165ms/step - loss: 0.9989
Epoch 13/50
57/57 [==============================] - 9s 164ms/step - loss: 1.0002
Epoch 14/50
57/57 [==============================] - 10s 175ms/step - loss: 0.9996
Epoch 15/50
57/57 [==============================] - 11s 187ms/step - loss: 0.9980
Epoch 16/50
57/57 [==============================] - 10s 179ms/step - loss: 0.9967
Epoch 17/50
57/57 [==============================] - 10s 178ms/step - loss: 0.9982
Epoch 18/50
57/57 [==============================] - 9s 162ms/step - loss: 0.9951
Epoch 19/50
57/57 [==============================] - 9s 165ms/step - loss: 0.9972
Epoch 20/50
57/57 [==============================] - 10s 172ms/step - loss: 0.9940
Epoch 21/50
57/57 [==============================] - 10s 177ms/step - loss: 0.9955
Epoch 22/50
57/57 [==============================] - 10s 173ms/step - loss: 0.9930
Epoch 23/50
57/57 [==============================] - 9s 165ms/step - loss: 0.9920
```

```
Epoch 24/50
57/57 [==============================] - 9s 158ms/step - loss: 0.9921
Epoch 25/50
57/57 [==============================] - 10s 169ms/step - loss: 0.9904
Epoch 26/50
57/57 [==============================] - 10s 168ms/step - loss: 0.9905
Epoch 27/50
57/57 [==============================] - 11s 193ms/step - loss: 0.9844
Epoch 28/50
57/57 [==============================] - 11s 188ms/step - loss: 0.9833
Epoch 29/50
57/57 [==============================] - 11s 191ms/step - loss: 0.9842
Epoch 30/50
57/57 [==============================] - 11s 197ms/step - loss: 0.9808
Epoch 31/50
57/57 [==============================] - 10s 170ms/step - loss: 0.9819
Epoch 32/50
57/57 [==============================] - 9s 156ms/step - loss: 0.9801
Epoch 33/50
57/57 [==============================] - 10s 177ms/step - loss: 0.9798
Epoch 34/50
57/57 [==============================] - 10s 168ms/step - loss: 0.97551s -
Epoch 35/50
57/57 [==============================] - 9s 160ms/step - loss: 0.9776
Epoch 36/50
57/57 [==============================] - 10s 173ms/step - loss: 0.9754
Epoch 37/50
57/57 [==============================] - 10s 170ms/step - loss: 0.9788
Epoch 38/50
57/57 [==============================] - 10s 168ms/step - loss: 0.9770
Epoch 39/50
57/57 [==============================] - 10s 175ms/step - loss: 0.9776
Epoch 40/50
57/57 [==============================] - 9s 166ms/step - loss: 0.9758
Epoch 41/50
57/57 [==============================] - 10s 178ms/step - loss: 0.9757
Epoch 42/50
57/57 [==============================] - 10s 176ms/step - loss: 0.9761
Epoch 43/50
57/57 [==============================] - 10s 174ms/step - loss: 0.9769
Epoch 44/50
57/57 [==============================] - 10s 172ms/step - loss: 0.9740
Epoch 45/50
57/57 [==============================] - 9s 157ms/step - loss: 0.9745
Epoch 46/50
57/57 [==============================] - 10s 173ms/step - loss: 0.9755
Epoch 47/50
57/57 [==============================] - 10s 171ms/step - loss: 0.9773
Epoch 48/50
57/57 [==============================] - 9s 157ms/step - loss: 0.9775
Epoch 49/50
57/57 [==============================] - 10s 172ms/step - loss: 0.9738
Epoch 50/50
57/57 [==============================] - 9s 163ms/step - loss: 0.9746
```

Out[19]:

```
<tensorflow.python.keras.callbacks.History at 0x20a6963e308>
```

In [20]:

```
x_test = np.reshape(x_test, (x_test.shape[0],x_test.shape[1],1))
y_pred = reg.predict(x_test)
y_pred = std.inverse_transform(y_pred)
y_test = std.inverse_transform(y_test)
```

In [21]:

```
from sklearn.metrics import mean_squared_error
import math

MSE = mean_squared_error(y_test, y_pred)

RMSE = math.sqrt(MSE)
print("Root Mean Square Error:\n")
print(RMSE)
```

```
Root Mean Square Error:

0.48569142342050614
```

In [22]:

```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_test,y_pred)
```

Out[22]:

```
0.40382534549934435
```

In [23]:

```
from sklearn.metrics import r2_score
r = r2_score(y_test,y_pred)
r
```

Out[23]:

```
0.7731194397385898
```

USING SELECTED PARAMETERS

In [24]:

```
cols = ['Yield',
       ' Predominantly reddish medium texture', ' Red desert soils',
       ' Sandy loam', ' Soil are lithosolsat foot hills & alluvials in plains',
       ' brown soils', ' clay loam',
       ' coarse sand in texture some places calcareous',
       ' deep soils in valleys', ' eastern part alluvial', ' foot hills',
       ' groundwater salinity', ' high soluble salts & exchangeable sodium',
       ' loamycoarse in texture & calcareous',
       ' nature of recently alluvial calcareous has been observed',
       ' sallow depth red soils in depressions', ' shallow on hills',
       ' well drained calcareous', ' west north west lithosols',
       'Alluvial deposites calcareous', 'Alluvial prone to water logging',
       'Black of alluvial origin', 'Desert soils and sand dunes aeolian soil',
       'Predominantly reddish medium texture', 'Red desert soils',
       'Sandy loam', 'Sierozens',
       'Soil are lithosolsat foot hills & alluvials in plains',
       'Mean_Seasonal_Rainfall']

x1 = np.array(df.drop(cols,axis=1))
y1 = np.array(df[["Yield"]])
```

In [25]:

```
from sklearn.preprocessing import StandardScaler
std = StandardScaler()
x1 = std.fit_transform(x1)
y1 = std.fit_transform(y1)
```

In [26]:

```
from sklearn.model_selection import train_test_split
x1_train, x1_test, y1_train, y1_test = train_test_split(x1,y1,test_size = 0.01 , random_state=71)
print(x1_train.shape)
print(y1_train.shape)
print(x1_test.shape)
print(y1_test.shape)
```

```
(3591, 42)
(3591, 1)
(37, 42)
(37, 1)
```

In [27]:

```
x1_train = np.reshape(x1_train,(x1_train.shape[0],x1_train.shape[1],1))
```

In [28]:

```
reg1 = Sequential()

reg1.add(LSTM(units=50, return_sequences=True, input_shape=(x1_train.shape[1],1)))
reg1.add(Dropout(0.2))

reg1.add(LSTM(units=50, return_sequences=True))
reg1.add(Dropout(0.2))

reg1.add(LSTM(units=50, return_sequences=True))
reg1.add(Dropout(0.2))
```

```
reg1.add(LSTM(units=50))
reg1.add(Dropout(0.2))

reg1.add(Dense(units=1))

reg1.compile(optimizer='adam', loss='mean_squared_error')

reg1.fit(x1_train,y1_train,epochs=50,batch_size=64)
```

```
Epoch 1/50
57/57 [==============================] - 6s 111ms/step - loss: 1.0081
Epoch 2/50
57/57 [==============================] - 6s 103ms/step - loss: 1.0088
Epoch 3/50
57/57 [==============================] - 6s 101ms/step - loss: 1.0067
Epoch 4/50
57/57 [==============================] - 6s 107ms/step - loss: 1.0074
Epoch 5/50
57/57 [==============================] - 6s 105ms/step - loss: 1.0074
Epoch 6/50
57/57 [==============================] - 6s 109ms/step - loss: 1.0065
Epoch 7/50
57/57 [==============================] - 6s 103ms/step - loss: 1.0005
Epoch 8/50
57/57 [==============================] - 6s 97ms/step - loss: 1.0078
Epoch 9/50
57/57 [==============================] - 6s 113ms/step - loss: 1.0023
Epoch 10/50
57/57 [==============================] - 6s 102ms/step - loss: 1.0002
Epoch 11/50
57/57 [==============================] - 6s 100ms/step - loss: 1.0047
Epoch 12/50
57/57 [==============================] - 7s 117ms/step - loss: 0.9991
Epoch 13/50
57/57 [==============================] - 6s 99ms/step - loss: 0.9973
Epoch 14/50
57/57 [==============================] - 6s 105ms/step - loss: 0.9958
Epoch 15/50
57/57 [==============================] - 6s 103ms/step - loss: 1.0013
Epoch 16/50
57/57 [==============================] - 6s 98ms/step - loss: 0.9976
Epoch 17/50
57/57 [==============================] - 6s 113ms/step - loss: 0.9952
Epoch 18/50
57/57 [==============================] - 6s 103ms/step - loss: 0.9908
Epoch 19/50
57/57 [==============================] - 6s 104ms/step - loss: 0.9950
Epoch 20/50
57/57 [==============================] - 6s 108ms/step - loss: 0.9923
Epoch 21/50
57/57 [==============================] - 6s 105ms/step - loss: 0.9919
Epoch 22/50
57/57 [==============================] - 6s 111ms/step - loss: 0.9929
Epoch 23/50
57/57 [==============================] - 6s 107ms/step - loss: 0.9936
Epoch 24/50
57/57 [==============================] - 6s 102ms/step - loss: 0.9883
Epoch 25/50
57/57 [==============================] - 6s 110ms/step - loss: 0.9871
Epoch 26/50
57/57 [==============================] - 6s 99ms/step - loss: 0.9841
Epoch 27/50
57/57 [==============================] - 6s 105ms/step - loss: 0.9812
Epoch 28/50
57/57 [==============================] - 7s 114ms/step - loss: 0.9823
Epoch 29/50
57/57 [==============================] - 6s 100ms/step - loss: 0.9786
Epoch 30/50
57/57 [==============================] - 6s 107ms/step - loss: 0.9770
Epoch 31/50
57/57 [==============================] - 6s 97ms/step - loss: 0.9773
Epoch 32/50
57/57 [==============================] - 6s 103ms/step - loss: 0.9773
Epoch 33/50
57/57 [==============================] - 6s 106ms/step - loss: 0.9751
Epoch 34/50
```

```
57/57 [==============================] - 6s 98ms/step - loss: 0.9745
Epoch 35/50
57/57 [==============================] - 6s 110ms/step - loss: 0.9704
Epoch 36/50
57/57 [==============================] - 6s 105ms/step - loss: 0.9762
Epoch 37/50
57/57 [==============================] - 6s 99ms/step - loss: 0.9727
Epoch 38/50
57/57 [==============================] - 7s 117ms/step - loss: 0.9705
Epoch 39/50
57/57 [==============================] - 6s 106ms/step - loss: 0.9694
Epoch 40/50
57/57 [==============================] - 6s 110ms/step - loss: 0.9786
Epoch 41/50
57/57 [==============================] - 6s 108ms/step - loss: 0.9721
Epoch 42/50
57/57 [==============================] - 6s 102ms/step - loss: 0.9722
Epoch 43/50
57/57 [==============================] - 6s 113ms/step - loss: 0.9760
Epoch 44/50
57/57 [==============================] - 6s 105ms/step - loss: 0.9749
Epoch 45/50
57/57 [==============================] - 6s 108ms/step - loss: 0.9711
Epoch 46/50
57/57 [==============================] - 6s 111ms/step - loss: 0.9700
Epoch 47/50
57/57 [==============================] - 6s 102ms/step - loss: 0.9714
Epoch 48/50
57/57 [==============================] - 7s 126ms/step - loss: 0.9721
Epoch 49/50
57/57 [==============================] - 6s 110ms/step - loss: 0.9679
Epoch 50/50
57/57 [==============================] - 6s 113ms/step - loss: 0.9659
```

Out[28]:

```
<tensorflow.python.keras.callbacks.History at 0x20a696fc348>
```

In [29]:

```
x1_test = np.reshape(x1_test, (x1_test.shape[0],x1_test.shape[1],1))
y1_pred = reg.predict(x1_test)
y1_pred = std.inverse_transform(y1_pred)
y1_test = std.inverse_transform(y1_test)
```

```
WARNING:tensorflow:Model was constructed with shape (None, 70, 1) for input Tensor("lstm_input:0",
shape=(None, 70, 1), dtype=float32), but it was called on an input with incompatible shape (None, 42,
1).
```

In [30]:

```
from sklearn.metrics import mean_squared_error
import math

MSE = mean_squared_error(y1_test, y1_pred)

RMSE = math.sqrt(MSE)
print("Root Mean Square Error:\n")
print(RMSE)
```

```
Root Mean Square Error:

0.48683427076780483
```

In [31]:

```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y1_test,y1_pred)
```

Out[31]:

```
0.4047155005956562
```

In [32]:

```
from sklearn.metrics import r2_score
r = r2_score(y1_test,y1_pred)
r
```

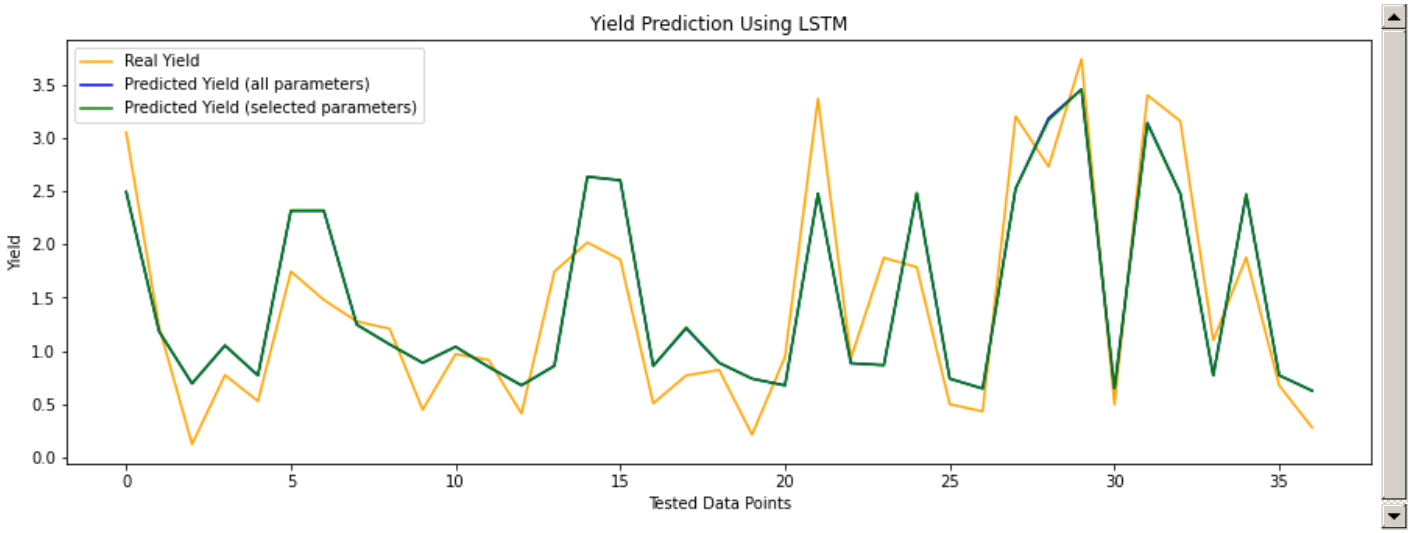
Out[32]:

```
0.7720504692254079
```

PERFORMANCE COMPARISON PLOTS

In [33]:

```
import matplotlib.pyplot as plt
plt.figure(figsize = (15,5))
plt.plot(y_test, color='orange',label='Real Yield')
plt.plot(y_pred, color='blue',label='Predicted Yield (all parameters)')
plt.plot(y1_pred, color='green',label='Predicted Yield (selected parameters)')
plt.title('Yield Prediction Using LSTM')
plt.xlabel("Tested Data Points")
plt.ylabel('Yield')
plt.legend()
plt.show()
```

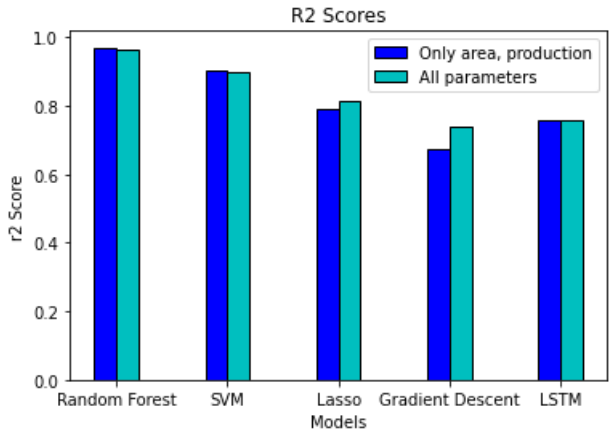


PERFORMANCE COMPARISON PLOTS

R2 SCORE

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt

area_production =
[0.9706683562457551,0.9033044480088029,0.7929212370102475,0.6721446803333357,0.7605970571904346]
Allparameters =
[0.9638436515081396,0.8982433161230687,0.8146824212217434,0.737347767385024,0.75781934584411]

n=5
r = np.arange(n)
width = 0.2


plt.bar(r, area_production, color = 'b',
        width = width, edgecolor = 'black',
        label='Only area, production')
plt.bar(r + width, Allparameters, color = 'c',
        width = width, edgecolor = 'black',
        label='All parameters')

plt.xlabel("Models")
plt.ylabel("r2 Score")
plt.title("R2 Scores")

# plt.grid(linestyle='--')
plt.xticks(r + width/2,['Random Forest','SVM','Lasso','Gradient Descent','LSTM'])
plt.legend()

plt.show()
```

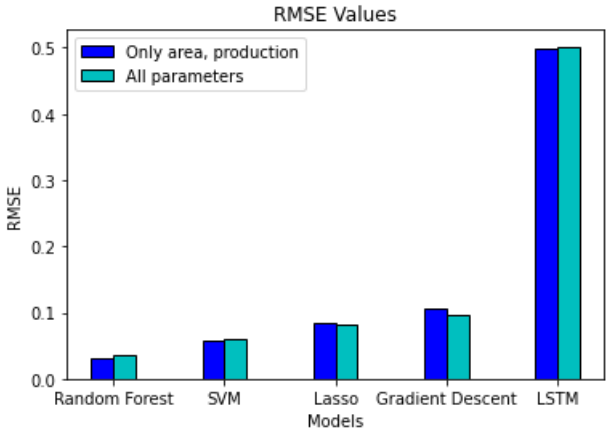


RMSE VALUES

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt

area_production = [0.03210438555877871,0.0582907187068775,0.08530292881366584,0.10733399123627024,0
.4989149684766394]
Allparameters = [0.03564416484020771,0.059796757425198625,0.08069645755381685, 0.09606976093705671,
0.50180099201939]

n=5
r = np.arange(n)
```

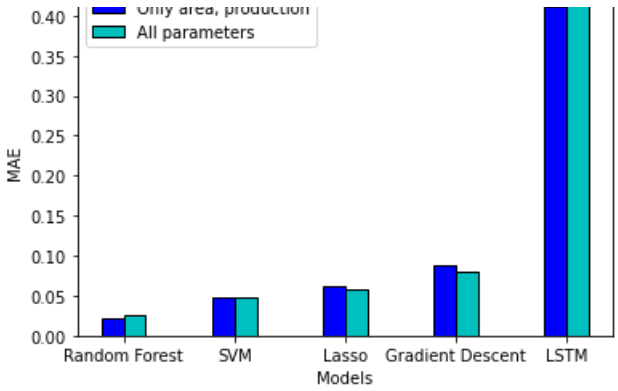
```
width = 0.2


plt.bar(r, area_production, color = 'b',
        width = width, edgecolor = 'black',
        label='Only area, production')
plt.bar(r + width, Allparameters, color = 'c',
        width = width, edgecolor = 'black',
        label='All parameters')

plt.xlabel("Models")
plt.ylabel("RMSE")
plt.title("RMSE Values")

# plt.grid(linestyle='--')
plt.xticks(r + width/2,['Random Forest','SVM','Lasso','Gradient Descent','LSTM'])
plt.legend()

plt.show()
```



MAE VALUES

In [3]:

```
import numpy as np
import matplotlib.pyplot as plt

area_production = [0.02171092742447838,0.04744273781182322,0.06279053471454211,0.08833348922241419,
0.4128063470375437]
Allparameters = [0.025122868659073126,0.047852253033345137,0.0588093331954562,0.07906984158002965,
0.4164927767185484]

n=5
r = np.arange(n)
width = 0.2


plt.bar(r, area_production, color = 'b',
        width = width, edgecolor = 'black',
        label='Only area, production')
plt.bar(r + width, Allparameters, color = 'c',
        width = width, edgecolor = 'black',
        label='All parameters')

plt.xlabel("Models")
plt.ylabel("MAE")
plt.title("MAE Values")

# plt.grid(linestyle='--')
plt.xticks(r + width/2,['Random Forest','SVM','Lasso','Gradient Descent','LSTM'])
plt.legend()

plt.show()
```


Only area, production
All parameters
MAE
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
Random Forest
SVM
Lasso
Gradient Descent
LSTM
Models