#CROP YIELD PREDICTION

#Dataset 1

##Reading Dataset

In [2]:

```
import pandas as pd
df = pd.read_csv('finalised_dataset.csv',na_values='=')
df
```

Out[2]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	are
0	125191	Maharashtra	AHMEDNAGAR	1997	Autumn	Maize	1.
1	125192	Maharashtra	AHMEDNAGAR	1997	Kharif	Arhar/Tur	17600.
2	125193	Maharashtra	AHMEDNAGAR	1997	Kharif	Bajra	274100.
3	125194	Maharashtra	AHMEDNAGAR	1997	Kharif	Gram	40800.
4	125195	Maharashtra	AHMEDNAGAR	1997	Kharif	Jowar	900.
...	...	...	...	...	...	...	.
12623	137814	Maharashtra	YAVATMAL	2014	Rabi	Jowar	4000.
12624	137815	Maharashtra	YAVATMAL	2014	Rabi	Maize	1300.
12625	137816	Maharashtra	YAVATMAL	2014	Rabi	Wheat	29100.
12626	137817	Maharashtra	YAVATMAL	2014	Summer	Groundnut	9400.
12627	137818	Maharashtra	YAVATMAL	2014	Whole Year	Sugarcane	8100.

12628 rows × 17 columns

In [3]:

```
df=df.drop('Yield', axis = 1)
```

In [4]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12628 entries, 0 to 12627
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Unnamed: 0      12628 non-null  int64  
 1   state_names     12628 non-null  object 
 2   district_names  12628 non-null  object 
 3   crop_year       12628 non-null  int64  
 4   season_names    12628 non-null  object 
 5   crop_names      12628 non-null  object 
 6   area            12628 non-null  float64
 7   temperature     12628 non-null  float64
 8   wind_speed      12628 non-null  float64
 9   pressure        12628 non-null  float64
 10  humidity        12628 non-null  float64
 11  soil_type       12628 non-null  object 
 12  N               12628 non-null  float64
 13  P               12628 non-null  float64
 14  K               12628 non-null  float64
 15  production      12496 non-null  float64
dtypes: float64(9), int64(2), object(5)
memory usage: 1.5+ MB
```

In [5]:

```
df.columns

```

Out[5]:

```
Index(['Unnamed: 0', 'state_names', 'district_names', 'crop_year',
       'season_names', 'crop_names', 'area', 'temperature', 'wind_speed',
       'pressure', 'humidity', 'soil_type', 'N', 'P', 'K', 'production'],
      dtype='object')
```

##Reducing Data to One State for Ease

In [6]:

```
df = df[df['state_names'] == "Maharashtra"]

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12628 entries, 0 to 12627
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Unnamed: 0      12628 non-null  int64  
 1   state_names     12628 non-null  object 
 2   district_names  12628 non-null  object 
 3   crop_year       12628 non-null  int64  
 4   season_names    12628 non-null  object 
 5   crop_names      12628 non-null  object 
 6   area            12628 non-null  float64
 7   temperature     12628 non-null  float64
 8   wind_speed      12628 non-null  float64
 9   pressure        12628 non-null  float64
 10  humidity        12628 non-null  float64
 11  soil_type       12628 non-null  object 
 12  N               12628 non-null  float64
 13  P               12628 non-null  float64
 14  K               12628 non-null  float64
 15  production      12496 non-null  float64
dtypes: float64(9), int64(2), object(5)
memory usage: 1.6+ MB
```

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12628 entries, 0 to 12627
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Unnamed: 0      12628 non-null  int64  
 1   state_names     12628 non-null  object 
 2   district_names  12628 non-null  object 
 3   crop_year       12628 non-null  int64  
 4   season_names    12628 non-null  object 
 5   crop_names      12628 non-null  object 
 6   area            12628 non-null  float64
 7   temperature     12628 non-null  float64
 8   wind_speed      12628 non-null  float64
 9   pressure        12628 non-null  float64
 10  humidity        12628 non-null  float64
 11  soil_type       12628 non-null  object 
 12  N               12628 non-null  float64
 13  P               12628 non-null  float64
 14  K               12628 non-null  float64
 15  production      12496 non-null  float64
dtypes: float64(9), int64(2), object(5)
memory usage: 1.6+ MB
```

In [8]:

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

Out[8]:

```
Unnamed: 0          0
state_names         0
district_names      0
crop_year           0
season_names        0
crop_names          0
area                0
temperature         0
wind_speed          0
pressure            0
humidity            0
soil_type           0
N                   0
P                   0
K                   0
production        132
dtype: int64
```

In [9]:

```
df.head(6)
```

Out[9]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	area	te
0	125191	Maharashtra	AHMEDNAGAR	1997	Autumn	Maize	1.0	
1	125192	Maharashtra	AHMEDNAGAR	1997	Kharif	Arhar/Tur	17600.0	
2	125193	Maharashtra	AHMEDNAGAR	1997	Kharif	Bajra	274100.0	
3	125194	Maharashtra	AHMEDNAGAR	1997	Kharif	Gram	40800.0	
4	125195	Maharashtra	AHMEDNAGAR	1997	Kharif	Jowar	900.0	
5	125196	Maharashtra	AHMEDNAGAR	1997	Kharif	Maize	4400.0	

###Making Yield Column

In [10]:

```
df
```

Out[10]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	are
0	125191	Maharashtra	AHMEDNAGAR	1997	Autumn	Maize	1.
1	125192	Maharashtra	AHMEDNAGAR	1997	Kharif	Arhar/Tur	17600.
2	125193	Maharashtra	AHMEDNAGAR	1997	Kharif	Bajra	274100.
3	125194	Maharashtra	AHMEDNAGAR	1997	Kharif	Gram	40800.
4	125195	Maharashtra	AHMEDNAGAR	1997	Kharif	Jowar	900.
...	...	...	...	...	...	...	.
12623	137814	Maharashtra	YAVATMAL	2014	Rabi	Jowar	4000.
12624	137815	Maharashtra	YAVATMAL	2014	Rabi	Maize	1300.
12625	137816	Maharashtra	YAVATMAL	2014	Rabi	Wheat	29100.
12626	137817	Maharashtra	YAVATMAL	2014	Summer	Groundnut	9400.
12627	137818	Maharashtra	YAVATMAL	2014	Whole Year	Sugarcane	8100.

12628 rows × 16 columns

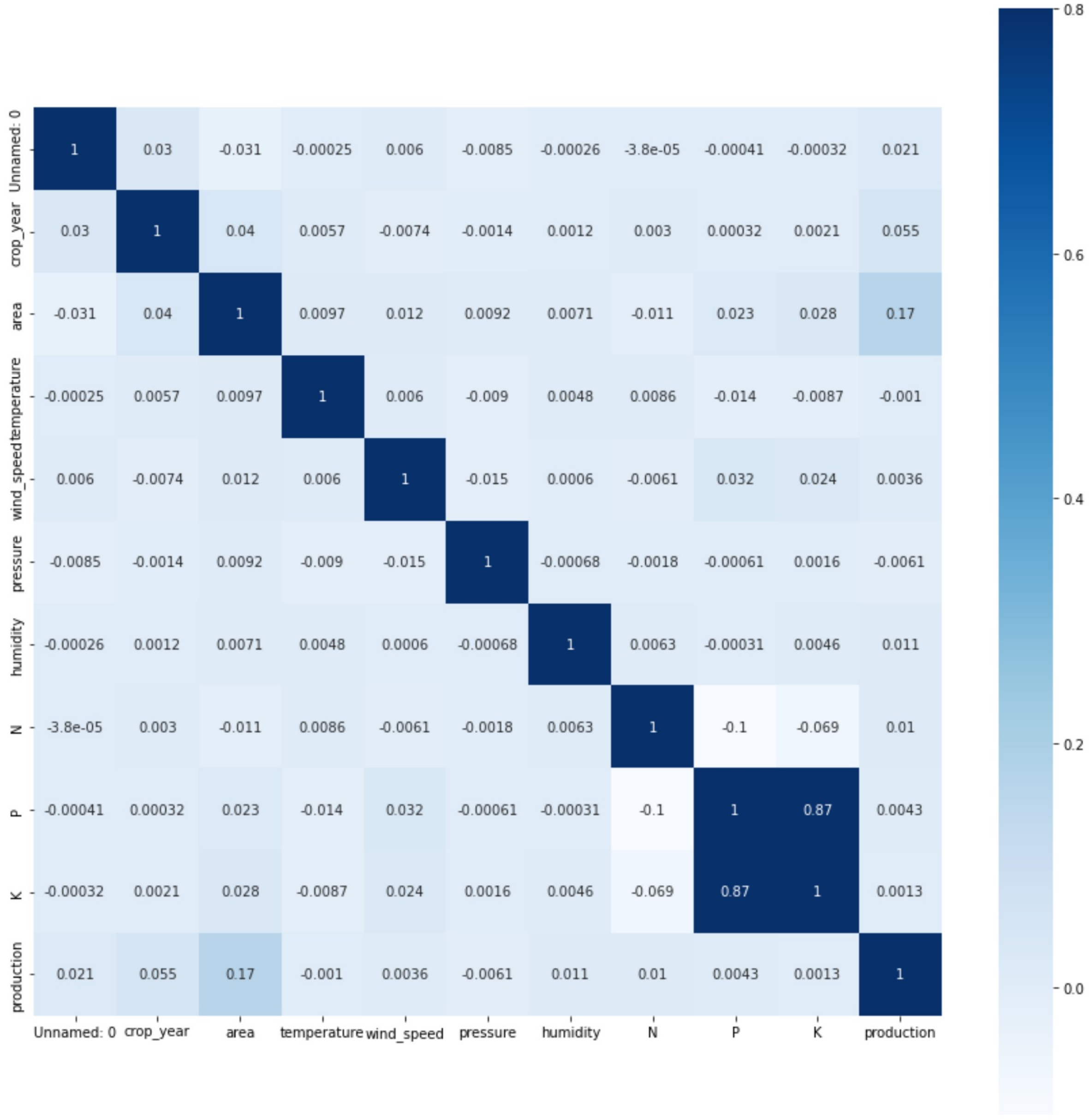
##Corelation Heatmap

In [11]:

```
import matplotlib.pyplot as plt
import seaborn as sb

C_mat = df.corr()
fig = plt.figure(figsize = (15,15))

sb.heatmap(C_mat, vmax = .8, square = True,cmap='Blues',annot=True)
plt.show()
```



##Taking Data only after 2004 because the data after 2004 is affecting the rersult alot

In [12]:

```
df = df[df['crop_year']>=2004]
df
```

Out[12]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	are
212	125403	Maharashtra	AHMEDNAGAR	2004	Kharif	Arhar/Tur	12200.
213	125404	Maharashtra	AHMEDNAGAR	2004	Kharif	Bajra	240500.
214	125405	Maharashtra	AHMEDNAGAR	2004	Kharif	Groundnut	5300.
215	125406	Maharashtra	AHMEDNAGAR	2004	Kharif	Jowar	100.
216	125407	Maharashtra	AHMEDNAGAR	2004	Kharif	Maize	11400.
...	...	...	...	...	...	...	.
12623	137814	Maharashtra	YAVATMAL	2014	Rabi	Jowar	4000.
12624	137815	Maharashtra	YAVATMAL	2014	Rabi	Maize	1300.
12625	137816	Maharashtra	YAVATMAL	2014	Rabi	Wheat	29100.
12626	137817	Maharashtra	YAVATMAL	2014	Summer	Groundnut	9400.
12627	137818	Maharashtra	YAVATMAL	2014	Whole Year	Sugarcane	8100.

7255 rows × 16 columns

In [13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7255 entries, 212 to 12627
Data columns (total 16 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Unnamed: 0      7255 non-null   int64  
 1   state_names     7255 non-null   object 
 2   district_names  7255 non-null   object 
 3   crop_year       7255 non-null   int64  
 4   season_names    7255 non-null   object 
 5   crop_names      7255 non-null   object 
 6   area            7255 non-null   float64
 7   temperature     7255 non-null   float64
 8   wind_speed      7255 non-null   float64
 9   pressure        7255 non-null   float64
 10  humidity        7255 non-null   float64
 11  soil_type       7255 non-null   object 
 12  N               7255 non-null   float64
 13  P               7255 non-null   float64
 14  K               7255 non-null   float64
 15  production      7143 non-null   float64
dtypes: float64(9), int64(2), object(5)
memory usage: 963.6+ KB
```

##converting data to numerical form

In [14]:

```python
df = df.join(pd.get_dummies(df['district_names']))
df = df.join(pd.get_dummies(df['season_names']))
df = df.join(pd.get_dummies(df['crop_names']))
df = df.join(pd.get_dummies(df['state_names']))
df = df.join(pd.get_dummies(df['soil_type']))
df
```

Out[14]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	are
212	125403	Maharashtra	AHMEDNAGAR	2004	Kharif	Arhar/Tur	12200.
213	125404	Maharashtra	AHMEDNAGAR	2004	Kharif	Bajra	240500.
214	125405	Maharashtra	AHMEDNAGAR	2004	Kharif	Groundnut	5300.
215	125406	Maharashtra	AHMEDNAGAR	2004	Kharif	Jowar	100.
216	125407	Maharashtra	AHMEDNAGAR	2004	Kharif	Maize	11400.
...	...	...	...	...	...	...	.
12623	137814	Maharashtra	YAVATMAL	2014	Rabi	Jowar	4000.
12624	137815	Maharashtra	YAVATMAL	2014	Rabi	Maize	1300.
12625	137816	Maharashtra	YAVATMAL	2014	Rabi	Wheat	29100.
12626	137817	Maharashtra	YAVATMAL	2014	Summer	Groundnut	9400.
12627	137818	Maharashtra	YAVATMAL	2014	Whole Year	Sugarcane	8100.

7255 rows × 88 columns

In [15]:

```python
df['Yield'] = df['production']/df['area']
df
```

Out[15]:

	Unnamed: 0	state_names	district_names	crop_year	season_names	crop_names	are
212	125403	Maharashtra	AHMEDNAGAR	2004	Kharif	Arhar/Tur	12200.
213	125404	Maharashtra	AHMEDNAGAR	2004	Kharif	Bajra	240500.
214	125405	Maharashtra	AHMEDNAGAR	2004	Kharif	Groundnut	5300.
215	125406	Maharashtra	AHMEDNAGAR	2004	Kharif	Jowar	100.
216	125407	Maharashtra	AHMEDNAGAR	2004	Kharif	Maize	11400.
...	...	...	...	...	...	...	.
12623	137814	Maharashtra	YAVATMAL	2014	Rabi	Jowar	4000.
12624	137815	Maharashtra	YAVATMAL	2014	Rabi	Maize	1300.
12625	137816	Maharashtra	YAVATMAL	2014	Rabi	Wheat	29100.
12626	137817	Maharashtra	YAVATMAL	2014	Summer	Groundnut	9400.
12627	137818	Maharashtra	YAVATMAL	2014	Whole Year	Sugarcane	8100.

7255 rows × 89 columns

In [16]:

```python
df = df.drop('production', axis=1)
```

##Dropping Unecessory Columns

In [17]:

```python
df=df.drop('district_names', axis=1)
df = df.drop('season_names',axis=1)
df = df.drop('crop_names',axis=1)



```

In [18]:

```python
df = df.drop('state_names', axis=1)
df = df.drop('soil_type', axis=1)
df
```

Out[18]:

	Unnamed: 0	crop_year	area	temperature	wind_speed	pressure	humidity	N
212	125403	2004	12200.0	20.768143	2.002031	1013.280471	20.427922	10.50
213	125404	2004	240500.0	20.722713	2.105239	1015.061641	20.468584	39.72
214	125405	2004	5300.0	21.419190	2.046843	1015.770055	21.836158	8.00
215	125406	2004	100.0	20.425919	2.024060	1013.971163	21.028403	5.82
216	125407	2004	11400.0	20.823344	1.989898	1015.453191	20.340815	0.00
...	...	...	...	...	...	...	...	..
12623	137814	2014	4000.0	21.635879	2.000060	1014.302213	20.060662	7.84
12624	137815	2014	1300.0	21.709611	2.053609	1015.803912	21.263478	7.50
12625	137816	2014	29100.0	21.851730	2.027476	1014.031903	20.059945	0.00
12626	137817	2014	9400.0	21.569380	2.004421	1013.989125	21.835158	2.89
12627	137818	2014	8100.0	21.666723	2.008003	1015.081619	21.754799	3.15

7255 rows × 83 columns

##Preprocessing

In [19]:

```python
from sklearn import preprocessing
```

In [20]:

```

# Create x, where x the 'scores' column's values as floats
x = df[['area']].values.astype(float)
x
# Create a minimum and maximum processor object
min_max_scaler = preprocessing.MinMaxScaler()

# Create an object to transform the data to fit minmax processor
x_scaled = min_max_scaler.fit_transform(x)

# Run the normalizer on the dataframe
#df_normalized = pd.DataFrame(x_scaled)
x_scaled

df['area'] = x_scaled
df
```

Out[20]:

	Unnamed: 0	crop_year	area	temperature	wind_speed	pressure	humidity	N
212	125403	2004	0.017150	20.768143	2.002031	1013.280471	20.427922	10.50
213	125404	2004	0.338112	20.722713	2.105239	1015.061641	20.468584	39.72
214	125405	2004	0.007450	21.419190	2.046843	1015.770055	21.836158	8.00
215	125406	2004	0.000139	20.425919	2.024060	1013.971163	21.028403	5.82
216	125407	2004	0.016026	20.823344	1.989898	1015.453191	20.340815	0.00
...	...	...	...	...	...	...	...	..
12623	137814	2014	0.005622	21.635879	2.000060	1014.302213	20.060662	7.84
12624	137815	2014	0.001826	21.709611	2.053609	1015.803912	21.263478	7.50
12625	137816	2014	0.040910	21.851730	2.027476	1014.031903	20.059945	0.00
12626	137817	2014	0.013214	21.569380	2.004421	1013.989125	21.835158	2.89
12627	137818	2014	0.011386	21.666723	2.008003	1015.081619	21.754799	3.15

7255 rows × 83 columns

In [21]:

```
df.head()
```

Out[21]:

	Unnamed: 0	crop_year	area	temperature	wind_speed	pressure	humidity	N
212	125403	2004	0.017150	20.768143	2.002031	1013.280471	20.427922	10.500
213	125404	2004	0.338112	20.722713	2.105239	1015.061641	20.468584	39.720
214	125405	2004	0.007450	21.419190	2.046843	1015.770055	21.836158	8.008
215	125406	2004	0.000139	20.425919	2.024060	1013.971163	21.028403	5.824
216	125407	2004	0.016026	20.823344	1.989898	1015.453191	20.340815	0.000

5 rows × 83 columns

Filling Empty Values With Mean

In [22]:
```
df = df.fillna(df.mean())
```

##Train and Test Split

In [23]:
```
from sklearn.model_selection import train_test_split
```

In [24]:
```
a=df
```

In [25]:
```
b = df['Yield']
#a = df.drop('Yield', axis = 1)


```

In [26]:
```
c = df.drop('Unnamed: 0', axis = 1)

```

In [ ]:
```

```

In [27]:
```
a=c.drop('Yield', axis = 1)
```

In [28]:
```
len(a.columns)
```

Out[28]:
```
81
```

In [29]:
```
a.columns
```

Out[29]:
```
Index(['crop_year', 'area', 'temperature', 'wind_speed', 'pressure',
       'humidity', 'N', 'P', 'K', 'AHMEDNAGAR', 'AKOLA', 'AMRAVATI',
       'AURANGABAD', 'BEED', 'BHANDARA', 'BULDHANA', 'CHANDRAPUR', 'DHULE',
       'GADCHIROLI', 'GONDIA', 'HINGOLI', 'JALGAON', 'JALNA', 'KOLHAPUR',
       'LATUR', 'NAGPUR', 'NANDED', 'NANDURBAR', 'NASHIK', 'OSMANABAD',
       'PALGHAR', 'PARBHANI', 'PUNE', 'RAIGAD', 'RATNAGIRI', 'SANGLI',
       'SATARA', 'SINDHUDURG', 'SOLAPUR', 'THANE', 'WARDHA', 'WASHIM',
       'YAVATMAL', 'Kharif     ', 'Rabi       ', 'Summer     ', 'Whole Year ',
       'Arhar/Tur', 'Bajra', 'Castor seed', 'Cotton(lint)', 'Gram',
       'Groundnut', 'Jowar', 'Linseed', 'Maize', 'Moong(Green Gram)',
       'Niger seed', 'Other  Rabi pulses', 'Other Cereals & Millets',
       'Other Kharif pulses', 'Ragi', 'Rapeseed &Mustard', 'Rice', 'Safflower',
       'Sesamum', 'Soyabean', 'Sugarcane', 'Sunflower', 'Tobacco', 'Urad',
       'Wheat', 'other oilseeds', 'Maharashtra', 'chalky', 'clay', 'loamy',
       'peaty', 'sandy', 'silt', 'silty'],
      dtype='object')
```

In [30]:

```
features_list=['crop_year', 'area', 'temperature', 'wind_speed', 'pressure',
       'humidity', 'N', 'P', 'K', 'AHMEDNAGAR', 'AKOLA', 'AMRAVATI',
       'AURANGABAD', 'BEED', 'BHANDARA', 'BULDHANA', 'CHANDRAPUR', 'DHULE',
       'GADCHIROLI', 'GONDIA', 'HINGOLI', 'JALGAON', 'JALNA', 'KOLHAPUR',
       'LATUR', 'NAGPUR', 'NANDED', 'NANDURBAR', 'NASHIK', 'OSMANABAD',
       'PALGHAR', 'PARBHANI', 'PUNE', 'RAIGAD', 'RATNAGIRI', 'SANGLI',
       'SATARA', 'SINDHUDURG', 'SOLAPUR', 'THANE', 'WARDHA', 'WASHIM',
       'YAVATMAL', 'Kharif     ', 'Rabi       ', 'Summer     ', 'Whole Year ',
       'Arhar/Tur', 'Bajra', 'Castor seed', 'Cotton(lint)', 'Gram',
       'Groundnut', 'Jowar', 'Linseed', 'Maize', 'Moong(Green Gram)',
       'Niger seed', 'Other  Rabi pulses', 'Other Cereals & Millets',
       'Other Kharif pulses', 'Ragi', 'Rapeseed &Mustard', 'Rice', 'Safflower',
       'Sesamum', 'Soyabean', 'Sugarcane', 'Sunflower', 'Tobacco', 'Urad',
       'Wheat', 'other oilseeds', 'Maharashtra', 'chalky', 'clay', 'loamy',
       'peaty', 'sandy', 'silt', 'silty']
```

In [31]:

```
features_list123=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

len(features_list123)
```

Out[31]:

```
81
```

In [32]:

```
len(features_list)
```

Out[32]:

```
81
```

In [33]:

```
a=df[features_list]
```

In [34]:

```
a.head()
```

Out[34]:

	crop_year	area	temperature	wind_speed	pressure	humidity	N	P	
212	2004	0.017150	20.768143	2.002031	1013.280471	20.427922	10.500	27.300	27
213	2004	0.338112	20.722713	2.105239	1015.061641	20.468584	39.720	105.920	52
214	2004	0.007450	21.419190	2.046843	1015.770055	21.836158	8.008	8.008	0
215	2004	0.000139	20.425919	2.024060	1013.971163	21.028403	5.824	14.560	5
216	2004	0.016026	20.823344	1.989898	1015.453191	20.340815	0.000	0.000	0

5 rows × 81 columns

In [35]:

```
a_train, a_test, b_train, b_test = train_test_split(a, b, test_size = 0.3, random_state = 42)

print(a_train)
print(a_test)
print(b_train)
print(b_test)
```

```
       crop_year      area  temperature  wind_speed     pressure   humidity  \
8196        2008  0.058343    21.287581    2.066838  1014.904551  20.709584
1674        2014  0.058764    21.080175    2.103093  1014.113752  20.359939
7662        2005  0.013354    20.922660    2.022428  1013.501715  20.642090
2732        2009  0.002389    20.331978    2.019252  1014.653555  21.467039
12588       2012  0.083789    20.773241    2.099441  1013.971550  21.542199
...          ...       ...          ...         ...          ...        ...
6581        2014  0.000702    20.090749    1.983827  1013.157407  20.411220
9120        2006  0.007731    21.544164    2.089016  1013.109528  21.377246
9155        2008  0.003935    20.960905    2.006062  1013.565065  21.525920
9319        2014  0.000420    21.965921    2.061763  1014.285744  21.873364
1582        2010  0.015604    20.139821    2.106225  1013.705632  20.377305

            N       P      K  AHMEDNAGAR  ...  Wheat  other oilseeds  \
8196   38.250  38.250  38.25           0  ...      0               0
1674    0.000   0.000   0.00           0  ...      0               0
7662    1.600   2.000   0.00           0  ...      0               0
2732    0.500   0.000   0.00           0  ...      0               0
12588   0.000  14.128   0.00           0  ...      0               0
...       ...     ...    ...         ...  ...    ...             ...
6581    0.224   0.280   0.00           0  ...      0               0
9120    0.000  26.832   0.00           0  ...      0               0
9155    7.588   7.588   0.00           0  ...      0               0
9319    2.921   2.921   0.00           0  ...      0               0
1582    7.476   7.476   0.00           0  ...      0               0

       Maharashtra  chalky  clay  loamy  peaty  sandy  silt  silty
8196             1       0     1      0      0      0     0      0
1674             1       0     0      0      0      0     1      0
7662             1       0     0      0      0      0     1      0
2732             1       0     1      0      0      0     0      0
12588            1       0     0      0      0      0     0      1
...            ...     ...   ...    ...    ...    ...   ...    ...
6581             1       0     0      0      0      0     1      0
9120             1       0     0      0      1      0     0      0
9155             1       0     0      0      0      0     1      0
9319             1       0     1      0      0      0     0      0
1582             1       0     0      1      0      0     0      0

[5078 rows x 81 columns]
       crop_year      area  temperature  wind_speed     pressure   humidity  \
5229        2010  0.000280    21.626783    2.093701  1015.509013  20.251363
3067        2009  0.044143    20.390844    2.036080  1015.153060  20.485877
6937        2010  0.000420    20.764626    1.999956  1014.064322  21.919638
11148       2005  0.000702    21.920160    2.047580  1014.961353  20.462800
11818       2005  0.000842    22.004754    1.993841  1015.757024  20.567282
...          ...       ...          ...         ...          ...        ...
5269        2012  0.406297    20.091516    2.095636  1014.099868  21.299885
3845        2012  0.007309    20.721014    1.991891  1013.609739  20.340941
```

```
2394       2011  0.000139   21.757283   1.973096  1013.299136  21.814687
12214      2011  0.086601   20.883769   2.045719  1015.076670  20.170742
9475       2007  0.000014   21.392058   2.033645  1014.615628  21.072524

            N       P      K  AHMEDNAGAR  ...  Wheat  other oilseeds  \
5229    1.500   0.000   0.00           0  ...      0               0
3067    5.980   5.980   0.00           0  ...      0               0
6937    8.740   8.740   0.00           0  ...      0               0
11148   3.496   3.496   0.00           0  ...      0               0
11818   0.000   0.000   0.00           0  ...      0               0
...       ...     ...    ...         ...  ...    ...             ...
5269   38.250  38.250  38.25           0  ...      0               0
3845    2.775   0.000   0.00           0  ...      0               0
2394    0.000   0.000   0.00           0  ...      0               0
12214   2.484   2.484   0.00           0  ...      0               0
9475    0.000   0.000   0.00           0  ...      0               0

       Maharashtra  chalky  clay  loamy  peaty  sandy  silt  silty
5229             1       1     0      0      0      0     0      0
3067             1       0     1      0      0      0     0      0
6937             1       0     1      0      0      0     0      0
11148            1       0     1      0      0      0     0      0
11818            1       0     1      0      0      0     0      0
...            ...     ...   ...    ...    ...    ...   ...    ...
5269             1       0     0      0      0      0     1      0
3845             1       0     0      0      0      1     0      0
2394             1       0     0      0      0      0     1      0
12214            1       0     0      1      0      0     0      0
9475             1       1     0      0      0      0     0      0

[2177 rows x 81 columns]
8196     0.681928
1674     0.389952
7662     0.726316
2732     0.470588
12588    1.152685
           ...
6581     0.200000
9120     1.836364
9155     1.142857
9319     0.666667
1582     0.936937
Name: Yield, Length: 5078, dtype: float64
5229     0.500000
3067     1.165605
6937     0.666667
11148    0.800000
11818    0.333333
           ...
5269     0.658131
3845     1.211538
2394     0.400000
12214    0.779221
9475     0.727273
Name: Yield, Length: 2177, dtype: float64
```

In [36]:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as seabornInstance
from sklearn.linear_model import LinearRegression
from sklearn import metrics
%matplotlib inline
```

In [37]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
a_train = sc.fit_transform(a_train)
a_test = sc.transform(a_test)
```

Random Forest Regressor

In [38]:

```
  from sklearn.ensemble import RandomForestRegressor
  regr = RandomForestRegressor(max_depth=2, random_state=0, n_estimators=100)
  regr.fit(a_train, b_train)
  b_pred = regr.predict(a_test)

  from sklearn.metrics import mean_squared_error as mse
  from sklearn.metrics import mean_absolute_error as mae
  from sklearn.metrics import r2_score

  print('MSE =', mse(b_pred, b_test))
  print('MAE =', mae(b_pred, b_test))
  print('R2 Score =', r2_score(b_pred, b_test))
```

```
MSE = 7.67104887996405
MAE = 0.8953650873829122
R2 Score = 0.9589614680509004
```

Polynomial Support Vector Machine

In [39]:

```
from sklearn.svm import SVR
regressorpoly=SVR(kernel='poly',epsilon=1.0)
regressorpoly.fit(a_train,b_train)
pred=regressorpoly.predict(a_test)
print(regressorpoly.score(a_test,b_test))
print(r2_score(b_test,b_pred))
```

```
0.6312485850825429
0.9598856437260073
```

XGBRegressor

In [42]:

```
from xgboost import XGBRegressor
from sklearn.metrics import mean_absolute_error
XGBModel = XGBRegressor()
XGBModel.fit(a_train,b_train , verbose=False)

# Get the mean absolute error on the validation data :
XGBpredictions = XGBModel.predict(a_test)
MAE = mean_absolute_error(b_test , XGBpredictions)
print('XGBoost validation MAE = ',MAE)
XGBpredictions
```

```
XGBoost validation MAE =  0.6670485576475581
```

Out[42]:

```
array([1.0828081 , 0.64165634, 0.80906236, ..., 1.2118115 , 0.9264982 ,
       0.68652374], dtype=float32)
```

In [43]:

```

print(r2_score(b_test , XGBpredictions))
```

```
0.9654928330252374
```

In [ ]:

```

```