In [15]:

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

In [16]:

```python
df=pd.read_csv('/Users/aditinarayan/Documents/sih/India Agriculture Crop Pro
```

In [17]:

```python
df
```

Out[17]:

	State	District	Crop	Year	Season	Area	Area Units	Production	Pro
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	1254.0	Hectare	2061.0	
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	1258.0	Hectare	2083.0	
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	1261.0	Hectare	1525.0	
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	3100.0	Hectare	5239.0	
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	3105.0	Hectare	5267.0	
...	...	...	...	...	...	...	...	...	
345402	Manipur	IMPHAL WEST	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345403	Manipur	SENAPATI	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345404	Manipur	TAMENGLONG	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345405	Manipur	THOUBAL	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345406	Manipur	UKHRUL	NaN	2019-20	Rabi	NaN	Hectare	NaN	

345407 rows × 10 columns

EDA

In [18]:

```python
df.describe()
```

Out[18]:

	Area	Production	Yield
count	3.453740e+05	3.404140e+05	345374.000000
mean	1.167019e+04	9.583711e+05	79.407569
std	4.583843e+04	2.152986e+07	916.628744
min	4.000000e-03	0.000000e+00	0.000000
25%	7.400000e+01	8.700000e+01	0.546742
50%	5.320000e+02	7.170000e+02	1.000000
75%	4.110000e+03	7.176000e+03	2.467080
max	8.580100e+06	1.597800e+09	43958.333333

In [19]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 345407 entries, 0 to 345406
Data columns (total 10 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   State             345407 non-null  object 
 1   District          345407 non-null  object 
 2   Crop              345375 non-null  object 
 3   Year              345407 non-null  object 
 4   Season            345406 non-null  object 
 5   Area              345374 non-null  float64
 6   Area Units        345407 non-null  object 
 7   Production        340414 non-null  float64
 8   Production Units  345407 non-null  object 
 9   Yield             345374 non-null  float64
dtypes: float64(3), object(7)
memory usage: 26.4+ MB
```

In [20]:

```
df.describe().T
```

Out[20]:

	count	mean	std	min	25%	50%	75%	
Area	345374.0	11670.191258	4.583843e+04	0.004	74.000000	532.0	4110.00000	8
Production	340414.0	958371.148664	2.152986e+07	0.000	87.000000	717.0	7176.00000	1
Yield	345374.0	79.407569	9.166287e+02	0.000	0.546742	1.0	2.46708	4

In [21]:

```
unique_crop_list = df["Crop"].unique()
print("Total number of unique crops - ", len(unique_crop_list))
print("\nWe have following unique crops in the dataset - \n", unique_crop_li
```

```
Total number of unique crops -  57

We have following unique crops in the dataset - 
 ['Arecanut' 'Banana' 'Black pepper' 'Cashewnut' 'Coconut' 'Dry chillies'
 'Ginger' 'Other Kharif pulses' 'other oilseeds' 'Rice' 'Sugarcane'
 'Sweet potato' 'Arhar/Tur' 'Bajra' 'Castor seed' 'Coriander'
 'Cotton(lint)' 'Gram' 'Groundnut' 'Horse-gram' 'Jowar' 'Linseed' 'Maize'
 'Mesta' 'Moong(Green Gram)' 'Niger seed' 'Onion' 'Other Rabi pulses'
 'Potato' 'Ragi' 'Rapeseed &Mustard' 'Safflower' 'Sesamum' 'Small millets'
 'Soyabean' 'Sunflower' 'Tapioca' 'Tobacco' 'Turmeric' 'Urad' 'Wheat'
 'Oilseeds total' 'Jute' 'Masoor' 'Peas & beans (Pulses)' 'Barley'
 'Garlic' 'Khesari' 'Sannhamp' 'Guar seed' 'Moth' 'Cardamom'
 'Other Cereals' 'Cowpea(Lobia)' 'Dry Ginger' 'Other Summer Pulses' nan]
```

In [22]:

```
unique_states = df["State"].unique()
print("Total number of states and union territories found in records - ", le
print("\n Name of unique states and union territories in the record dataset
```

```
Total number of states and union territories found in records -  36

 Name of unique states and union territories in the record dataset -
 ['Andaman and Nicobar Islands' 'Andhra Pradesh' 'Arunachal Pradesh'
 'Assam' 'Bihar' 'Chandigarh' 'Chhattisgarh' 'Dadra and Nagar Haveli'
 'Daman and Diu' 'Delhi' 'Goa' 'Gujarat' 'Haryana' 'Himachal Pradesh'
 'Jammu and Kashmir' 'Jharkhand' 'Karnataka' 'Kerala' 'Madhya Pradesh'
 'Maharashtra' 'Manipur' 'Meghalaya' 'Mizoram' 'Nagaland' 'Odisha'
 'Puducherry' 'Punjab' 'Rajasthan' 'Sikkim' 'Tamil Nadu' 'Tripura'
 'Uttar Pradesh' 'Uttarakhand' 'West Bengal' 'Telangana' 'Laddakh']
```

In [23]:

```
unique_districts = df["District"].unique()
print("Total number of districts found in records - ", len(unique_districts)
```

```
Total number of districts found in records -  729
```

In [24]:

```
unique_units = list(df["Production Units"].unique())
print(unique_units)
```

```
['Tonnes', 'Nuts', 'Bales']
```

In [25]:

```
def unit_standardization(df):
    """
    Converts Nuts and Bales into Ton and standardize the unit of production
    """

    if df["Production Units"] == "Nuts":
        new_production = df["Production"] / 50
        return new_production

    elif df["Production Units"] == "Tonnes":
        return df["Production"]

    else:
        new_production = df["Production"] / 4.59
        return new_production


df["New Production"] = df.apply(unit_standardization, axis = 1)
df.sample(10)
```

Out[25]:

	State	District	Crop	Year	Season	Area	Area Units	Production
37380	Uttar Pradesh	PRATAPGARH	Ragi	2002-03	Rabi	3.0	Hectare	4.0
30430	Rajasthan	TONK	Groundnut	2001-02	Kharif	22347.0	Hectare	12240.0
329909	Punjab	LUDHIANA	Rice	2000-01	Kharif	238000.0	Hectare	939000.0
49564	Bihar	LAKHISARAI	Potato	2004-05	Whole Year	294.0	Hectare	2188.0
69594	Odisha	JAJAPUR	Groundnut	2004-05	Autumn	209.0	Hectare	100.1
243202	Karnataka	KOLAR	Coconut	2017-18	Whole Year	1976.0	Hectare	15624000.0
233181	Bihar	BEGUSARAI	Urad	2016-17	Kharif	920.0	Hectare	783.0
336734	Uttar Pradesh	KAUSHAMBI	Gram	2000-01	Rabi	18946.0	Hectare	18473.0
8525	Bihar	SHEOHAR	Tobacco	2002-03	Whole Year	65.0	Hectare	73.0
221722	Uttar Pradesh	LALITPUR	Urad	2015-16	Kharif	180311.0	Hectare	19474.0

In [26]:

```python
df.drop(columns = ["Production", "Production Units"], inplace = True)
```

In [27]:

```python
df["Crop"].value_counts().head()
```

Out[27]:

```
Rice                 21611
Maize                20507
Moong(Green Gram)    15101
Urad                 14581
Sesamum              13049
Name: Crop, dtype: int64
```

In [28]:

```python
total_production_list = []
for state in unique_states:
    total_crop = df.loc[df["State"] == state, "New Production"].sum()
    total_production_list.append(total_crop)


crop_production_df = pd.DataFrame({"State" : unique_states,
             "Total Crop Production" : total_production_list})
```

In [29]:

```python
crop_production_df.sort_values("Total Crop Production", ascending = False).h
```

Out[29]:

	State	Total Crop Production
31	Uttar Pradesh	4.442549e+09
17	Kerala	2.685620e+09
29	Tamil Nadu	2.563321e+09
16	Karnataka	2.316393e+09
19	Maharashtra	1.790398e+09

In [30]:
```
df.groupby(['State', 'District', 'Crop', 'Year']).size()
```

Out[30]:
```
State                         District                      Crop          Year
Andaman and Nicobar Islands   Andaman and Nicobar Islands   Arecanut      2007-08
2
                                                                          2008-09
2
                                                                          2009-10
2
                                                            Arhar/Tur     2007-08
1
                                                                          2008-09
1

..
West Bengal                   PURULIA                       Wheat         2015-16
1
                                                                          2016-17
1
                                                                          2017-18
1
                                                                          2018-19
1
                                                                          2019-20
1
Length: 297482, dtype: int64
```

In [31]:
```
g = df[df['District'] == "Andaman and Nicobar Islands"][df['Year'] == "2007-
g
```

```
/var/folders/8h/80kyp88j21b75qcg83zjngfm0000gn/T/ipykernel_82682/1731573671.
py:1: UserWarning: Boolean Series key will be reindexed to match DataFrame i
ndex.
  g = df[df['District'] == "Andaman and Nicobar Islands"][df['Year'] == "200
7-08"][df['Crop'] == "Arecanut"]
```

Out[31]:

	State	District	Crop	Year	Season	Area	Area Units	Yield	New Production
85043	Andaman and Nicobar Islands	Andaman and Nicobar Islands	Arecanut	2007-08	Kharif	2439.6	Hectare	1.4	3415.44
85044	Andaman and Nicobar Islands	Andaman and Nicobar Islands	Arecanut	2007-08	Rabi	1626.4	Hectare	1.4	2276.96

In [32]:
```
duplicate = df[df.duplicated()]

duplicate.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 0 entries
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   State           0 non-null      object 
 1   District        0 non-null      object 
 2   Crop            0 non-null      object 
 3   Year            0 non-null      object 
 4   Season          0 non-null      object 
 5   Area            0 non-null      float64
 6   Area Units      0 non-null      object 
 7   Yield           0 non-null      float64
 8   New Production  0 non-null      float64
dtypes: float64(3), object(6)
memory usage: 0.0+ bytes
```

In [33]:
```
df.count()
```

Out[33]:
```
State             345407
District          345407
Crop              345375
Year              345407
Season            345406
Area              345374
Area Units        345407
Yield             345374
New Production    340414
dtype: int64
```

In [34]:
```
df
```

Out[34]:

	State	District	Crop	Year	Season	Area	Area Units	Yield	Produ
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	1254.0	Hectare	1.643541	2
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	1258.0	Hectare	1.655803	2
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	1261.0	Hectare	1.209358	1
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	3100.0	Hectare	1.690000	5
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	3105.0	Hectare	1.696296	5
...	...	...	...	...	...	...	...	...	
345402	Manipur	IMPHAL WEST	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345403	Manipur	SENAPATI	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345404	Manipur	TAMENGLONG	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345405	Manipur	THOUBAL	NaN	2019-20	Rabi	NaN	Hectare	NaN	
345406	Manipur	UKHRUL	NaN	2019-20	Rabi	NaN	Hectare	NaN	

345407 rows × 9 columns

In [35]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 345407 entries, 0 to 345406
Data columns (total 9 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   State           345407 non-null  object 
 1   District        345407 non-null  object 
 2   Crop            345375 non-null  object 
 3   Year            345407 non-null  object 
 4   Season          345406 non-null  object 
 5   Area            345374 non-null  float64
 6   Area Units      345407 non-null  object 
 7   Yield           345374 non-null  float64
 8   New Production  340414 non-null  float64
dtypes: float64(3), object(6)
memory usage: 23.7+ MB
```

In [36]:

```
df.groupby(['District', 'Crop', 'Year', 'Season']).size()
```

Out[36]:
```
District           Crop            Year     Season
24 PARAGANAS NORTH Arecanut        1997-98  Whole Year     1
                                   1998-99  Whole Year     1
                                   1999-00  Whole Year     1
                                   2000-01  Whole Year     1
                                   2001-02  Whole Year     1
                                                          ..
ZUNHEBOTO          other oilseeds  2015-16  Kharif         1
                                   2016-17  Kharif         1
                                   2017-18  Kharif         1
                                   2018-19  Kharif         1
                                   2019-20  Kharif         1
Length: 344012, dtype: int64
```

In [37]:
```
catg = df.select_dtypes("object")
cont = df.select_dtypes("float")
catg.head()
```

Out[37]:

	State	District	Crop	Year	Season	Area Units
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	Hectare
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	Hectare
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	Hectare
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	Hectare
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	Hectare

In [38]:
```
cont.head()
```

Out[38]:

	Area	Yield	New Production
0	1254.0	1.643541	2061.0
1	1258.0	1.655803	2083.0
2	1261.0	1.209358	1525.0
3	3100.0	1.690000	5239.0
4	3105.0	1.696296	5267.0

In [39]:
```
catg.describe()
```

Out[39]:

	State	District	Crop	Year	Season	Area Units
count	345407	345407	345375	345407	345406	345407
unique	36	729	56	24	6	1
top	Uttar Pradesh	BILASPUR	Rice	2019-20	Kharif	Hectare
freq	44781	1244	21611	19296	138400	345407

In [40]:
```
df["Yield"] = df["Yield"].round(2)

df.head()
```

Out[40]:

	State	District	Crop	Year	Season	Area	Area Units	Yield	New Production
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	1254.0	Hectare	1.64	2061.0
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	1258.0	Hectare	1.66	2083.0
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	1261.0	Hectare	1.21	1525.0
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	3100.0	Hectare	1.69	5239.0
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	3105.0	Hectare	1.70	5267.0

In [41]:

```
df["State"].value_counts()
```

Out[41]:

```
Uttar Pradesh                  44781
Madhya Pradesh                 29906
Karnataka                      27493
Bihar                          24697
Rajasthan                      20363
Tamil Nadu                     18525
Assam                          18186
Maharashtra                    17922
Andhra Pradesh                 16363
Odisha                         16153
Chhattisgarh                   15285
Gujarat                        14053
West Bengal                    12596
Haryana                         8305
Uttarakhand                     6702
Nagaland                        5676
Himachal Pradesh                5043
Jharkhand                       5004
Kerala                          4870
Telangana                       4704
Jammu and Kashmir               4348
Arunachal Pradesh               4345
Meghalaya                       4322
Punjab                          4142
Manipur                         3120
Tripura                         2557
Mizoram                         2112
Puducherry                      1127
Sikkim                           876
Andaman and Nicobar Islands      728
Goa                              399
Dadra and Nagar Haveli           332
Delhi                            203
Chandigarh                       124
Daman and Diu                     44
Laddakh                            1
Name: State, dtype: int64
```

In [42]:

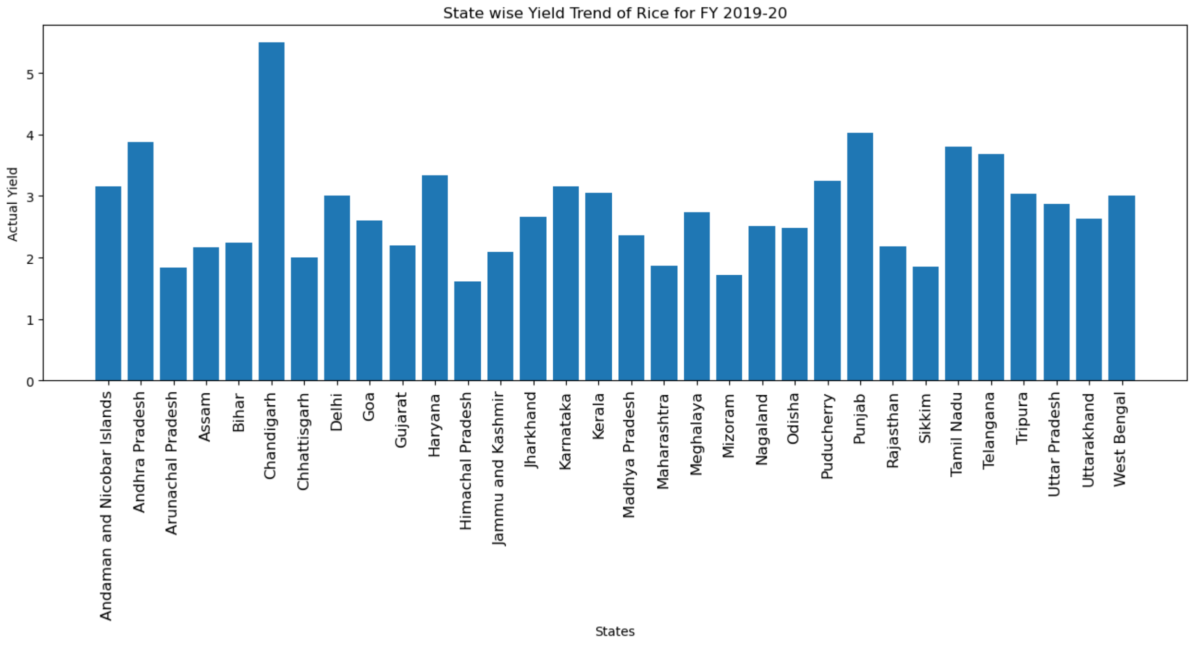
```
# Filter the DataFrame based on Year and Crop conditions
dfa2020 = df[(df['Year'] == "2019-20") & (df['Crop'] == "Rice")]

# Select specific columns 'State', 'Area', and 'Production'
dfa2020 = dfa2020[['State', 'Area', 'New Production']]

# Group by 'State', calculate the sum of 'Area' and 'Production', and reset
g = dfa2020.groupby('State').agg({'Area': 'sum', 'New Production': 'sum'}).r

# Calculate 'Actual Yield' as the ratio of 'Production' to 'Area'
g['Actual Yield'] = g['New Production'] / g['Area']
g
```

Out[42]:

	State	Area	New Production	Actual Yield
0	Andaman and Nicobar Islands	5701.12	17981.25	3.153986
1	Andhra Pradesh	2355982.00	9140091.00	3.879525
2	Arunachal Pradesh	133500.00	244741.00	1.833266
3	Assam	2400949.00	5214804.00	2.171976
4	Bihar	3097390.00	6952518.00	2.244638
5	Chandigarh	80.00	440.00	5.500000
6	Chhattisgarh	4266022.00	8569367.00	2.008749
7	Delhi	5848.00	17580.00	3.006156
8	Goa	34698.00	90375.00	2.604617
9	Gujarat	904350.00	1982633.00	2.192329
10	Haryana	1558900.00	5194600.00	3.332221
11	Himachal Pradesh	72620.00	116879.00	1.609460
12	Jammu and Kashmir	280513.00	587101.28	2.092956
13	Jharkhand	1357726.00	3612589.00	2.660764
14	Karnataka	1248054.00	3947973.00	3.163303
15	Kerala	198180.00	605541.00	3.055510
16	Madhya Pradesh	3110311.00	7363430.39	2.367426
17	Maharashtra	1552989.00	2897433.00	1.865714
18	Meghalaya	110997.00	303476.00	2.734092
19	Mizoram	35210.00	60239.00	1.710849
20	Nagaland	216950.00	544970.00	2.511961
21	Odisha	3940710.00	9755050.00	2.475455
22	Puducherry	18238.00	59345.00	3.253920
23	Punjab	3142000.00	12675000.00	4.034055
24	Rajasthan	219525.00	480554.00	2.189063
25	Sikkim	8685.00	16137.00	1.858031
26	Tamil Nadu	1907407.00	7265161.00	3.808920
27	Telangana	3234445.00	11923901.00	3.686537
28	Tripura	267335.00	810244.00	3.030819
29	Uttar Pradesh	5924349.00	17027889.00	2.874221
30	Uttarakhand	257781.00	677429.00	2.627924
31	West Bengal	5490975.00	16476021.00	3.000564

In [43]:

```
plt.figure(figsize = (16, 5))
plt.bar(g['State'], g['Actual Yield'])
plt.xticks(g['State'], rotation = 'vertical', size=12)
plt.xlabel('States')
plt.ylabel('Actual Yield')
plt.title('State wise Yield Trend of Rice for FY 2019-20')
plt.show()
```



In [44]:
```
g.nlargest(5, 'Actual Yield')
```

Out[44]:

	State	Area	New Production	Actual Yield
5	Chandigarh	80.0	440.0	5.500000
23	Punjab	3142000.0	12675000.0	4.034055
1	Andhra Pradesh	2355982.0	9140091.0	3.879525
26	Tamil Nadu	1907407.0	7265161.0	3.808920
27	Telangana	3234445.0	11923901.0	3.686537

In [45]:
```
g.nlargest(5, 'New Production')
```

Out[45]:

	State	Area	New Production	Actual Yield
29	Uttar Pradesh	5924349.0	17027889.0	2.874221
31	West Bengal	5490975.0	16476021.0	3.000564
23	Punjab	3142000.0	12675000.0	4.034055
27	Telangana	3234445.0	11923901.0	3.686537
21	Odisha	3940710.0	9755050.0	2.475455

In [46]:
```
df
```

Out[46]:

	State	District	Crop	Year	Season	Area	Area Units	Yield	Ne Productic
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	1254.0	Hectare	1.64	2061
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	1258.0	Hectare	1.66	2083
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	1261.0	Hectare	1.21	1525
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	3100.0	Hectare	1.69	5239
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	3105.0	Hectare	1.70	5267
...	...	...	...	...	...	...	...	...	
345402	Manipur	IMPHAL WEST	NaN	2019-20	Rabi	NaN	Hectare	NaN	Na
345403	Manipur	SENAPATI	NaN	2019-20	Rabi	NaN	Hectare	NaN	Na
345404	Manipur	TAMENGLONG	NaN	2019-20	Rabi	NaN	Hectare	NaN	Na
345405	Manipur	THOUBAL	NaN	2019-20	Rabi	NaN	Hectare	NaN	Na
345406	Manipur	UKHRUL	NaN	2019-20	Rabi	NaN	Hectare	NaN	Na

345407 rows × 9 columns

In [47]:

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

Out[47]:

```
State                0
District             0
Crop                32
Year                 0
Season               1
Area                33
Area Units           0
Yield               33
New Production    4993
dtype: int64
```

In [48]:

```
df['Area'].fillna(df['Area'].mean(), inplace=True)
df['New Production'].fillna(df['New Production'].mean(), inplace=True)
df['Yield'].fillna(df['Yield'].mean(), inplace=True)
```

In [49]:

```
df['Crop'].fillna(df['Crop'].mode()[0], inplace=True)
df['Season'].fillna(df['Season'].mode()[0], inplace=True)
```

In [50]:

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

In [51]:

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

Out[51]:

```
State             0
District          0
Crop              0
Year              0
Season            0
Area              0
Area Units        0
Yield             0
New Production    0
dtype: int64
```

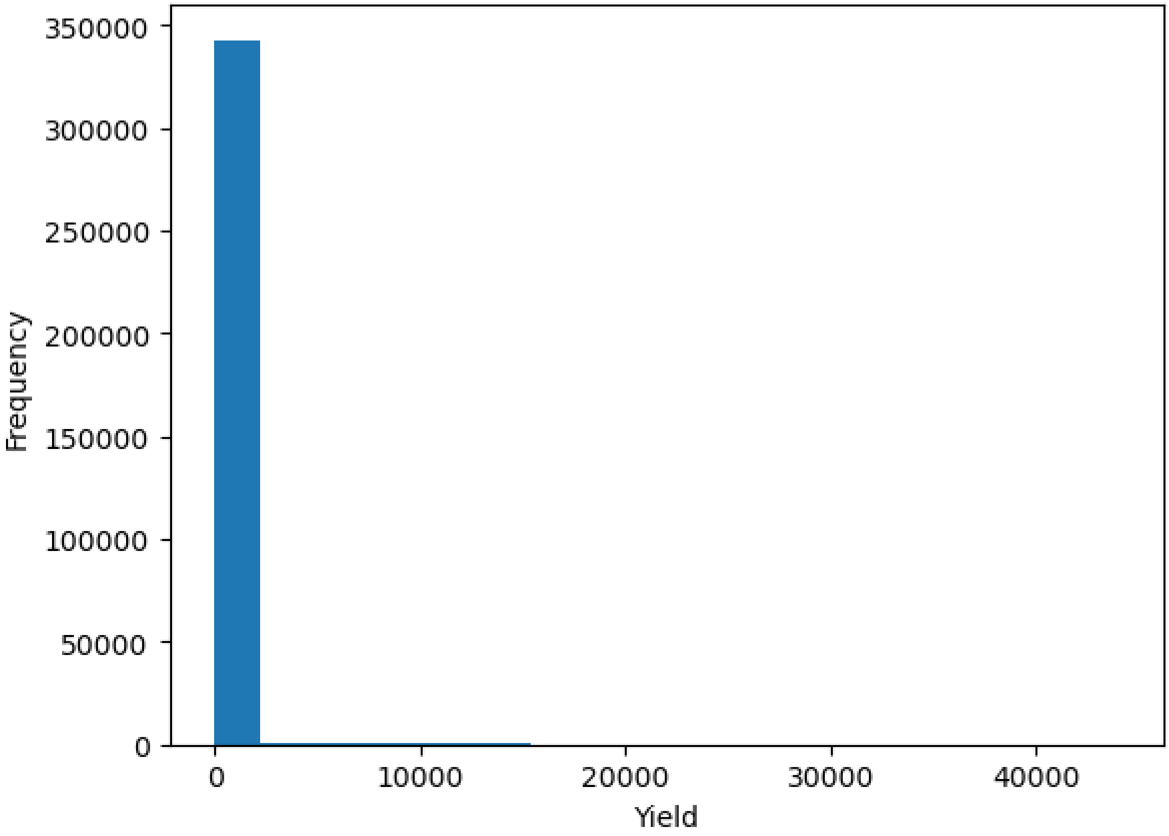
In [52]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 345407 entries, 0 to 345406
Data columns (total 9 columns):
 #   Column          Non-Null Count   Dtype  
---  ------          --------------   -----  
 0   State           345407 non-null  object 
 1   District        345407 non-null  object 
 2   Crop            345407 non-null  object 
 3   Year            345407 non-null  object 
 4   Season          345407 non-null  object 
 5   Area            345407 non-null  float64
 6   Area Units      345407 non-null  object 
 7   Yield           345407 non-null  float64
 8   New Production  345407 non-null  float64
dtypes: float64(3), object(6)
memory usage: 23.7+ MB
```

In [53]:

```
plt.hist(df['Yield'], bins=20)
plt.xlabel('Yield')
plt.ylabel('Frequency')
plt.show()
```



In [54]:
```
df.head()
```

Out[54]:

	State	District	Crop	Year	Season	Area	Area Units	Yield	New Production
0	Andaman and Nicobar Islands	NICOBARS	Arecanut	2001-02	Kharif	1254.0	Hectare	1.64	2061.0
1	Andaman and Nicobar Islands	NICOBARS	Arecanut	2002-03	Whole Year	1258.0	Hectare	1.66	2083.0
2	Andaman and Nicobar Islands	NICOBARS	Arecanut	2003-04	Whole Year	1261.0	Hectare	1.21	1525.0
3	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	Arecanut	2001-02	Kharif	3100.0	Hectare	1.69	5239.0
4	Andaman and Nicobar Islands	SOUTH ANDAMANS	Arecanut	2002-03	Whole Year	3105.0	Hectare	1.70	5267.0

In [55]:
```
df['Year']
```

Out[55]:

```
0         2001-02
1         2002-03
2         2003-04
3         2001-02
4         2002-03
            ...
345402    2019-20
345403    2019-20
345404    2019-20
345405    2019-20
345406    2019-20
Name: Year, Length: 345407, dtype: object
```

In [56]:

```
df[['Year', 'Month']] = df['Year'].str.split('-', n=1, expand=True)

# Convert 'Year' and 'Month' to numeric
df['Year'] = df['Year'].astype(int)
df['Month'] = df['Month'].astype(int)

# Resulting DataFrame with numeric 'Year' and 'Month' columns
print(df)
```

```
                              State                  District      Crop  Yea
r  \
0       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
1
1       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
2
2       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
3
3       Andaman and Nicobar Islands  NORTH AND MIDDLE ANDAMAN  Arecanut  200
1
4       Andaman and Nicobar Islands            SOUTH ANDAMANS  Arecanut  200
2
...                             ...                       ...       ...
...
345402                      Manipur               IMPHAL WEST      Rice  201
9
345403                      Manipur                  SENAPATI      Rice  201
9
345404                      Manipur                TAMENGLONG      Rice  201
9
345405                      Manipur                   THOUBAL      Rice  201
9
345406                      Manipur                    UKHRUL      Rice  201
9

            Season          Area Area Units      Yield  New Production  Mont
h
0           Kharif   1254.000000    Hectare   1.640000     2061.000000
2
1       Whole Year   1258.000000    Hectare   1.660000     2083.000000
3
2       Whole Year   1261.000000    Hectare   1.210000     1525.000000
4
3           Kharif   3100.000000    Hectare   1.690000     5239.000000
2
4       Whole Year   3105.000000    Hectare   1.700000     5267.000000
3
...            ...           ...        ...        ...             ...
...
345402        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345403        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345404        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345405        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345406        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0

[345407 rows x 10 columns]
```

```
In [57]: df.columns
```

```
Out[57]: Index(['State', 'District', 'Crop', 'Year', 'Season', 'Area', 'Area Units',
                'Yield', 'New Production', 'Month'],
               dtype='object')
```

```
In [58]: df.drop
```

Out[58]:

```
<bound method DataFrame.drop of                               State
District      Crop  Year  \
0       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
1
1       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
2
2       Andaman and Nicobar Islands                  NICOBARS  Arecanut  200
3
3       Andaman and Nicobar Islands  NORTH AND MIDDLE ANDAMAN  Arecanut  200
1
4       Andaman and Nicobar Islands            SOUTH ANDAMANS  Arecanut  200
2
...                             ...                       ...       ...
...
345402                      Manipur               IMPHAL WEST      Rice  201
9
345403                      Manipur                  SENAPATI      Rice  201
9
345404                      Manipur                TAMENGLONG      Rice  201
9
345405                      Manipur                   THOUBAL      Rice  201
9
345406                      Manipur                    UKHRUL      Rice  201
9

            Season          Area Area Units      Yield  New Production  Mont
h
0           Kharif   1254.000000    Hectare   1.640000     2061.000000
2
1       Whole Year   1258.000000    Hectare   1.660000     2083.000000
3
2       Whole Year   1261.000000    Hectare   1.210000     1525.000000
4
3           Kharif   3100.000000    Hectare   1.690000     5239.000000
2
4       Whole Year   3105.000000    Hectare   1.700000     5267.000000
3
...            ...           ...        ...        ...             ...
...
345402        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345403        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345404        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345405        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0
345406        Rabi  11670.191258    Hectare  79.407556    61938.064644     2
0

[345407 rows x 10 columns]>
```

In [59]:

```
df.drop('Area Units', axis=1, inplace=True)
```

In [60]:

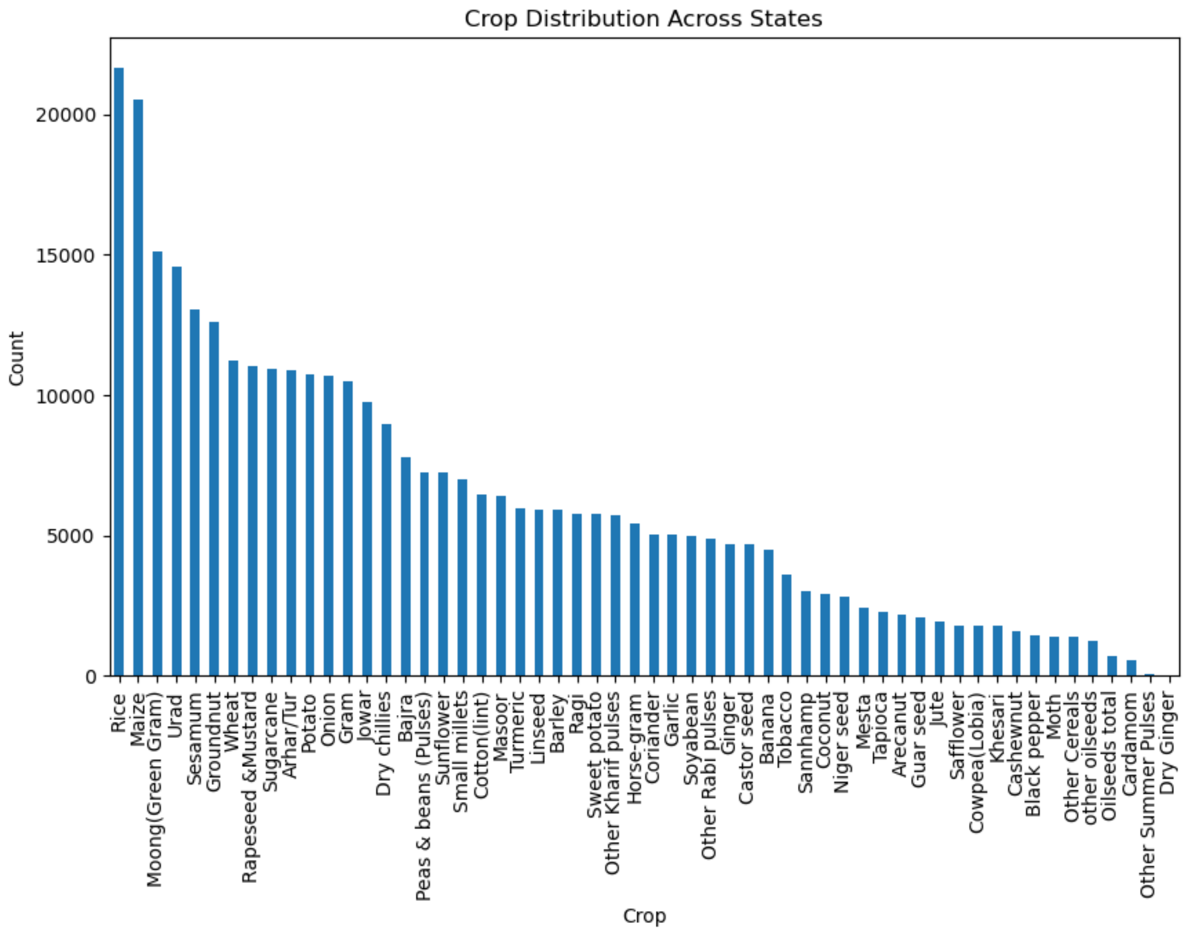
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

In [61]:

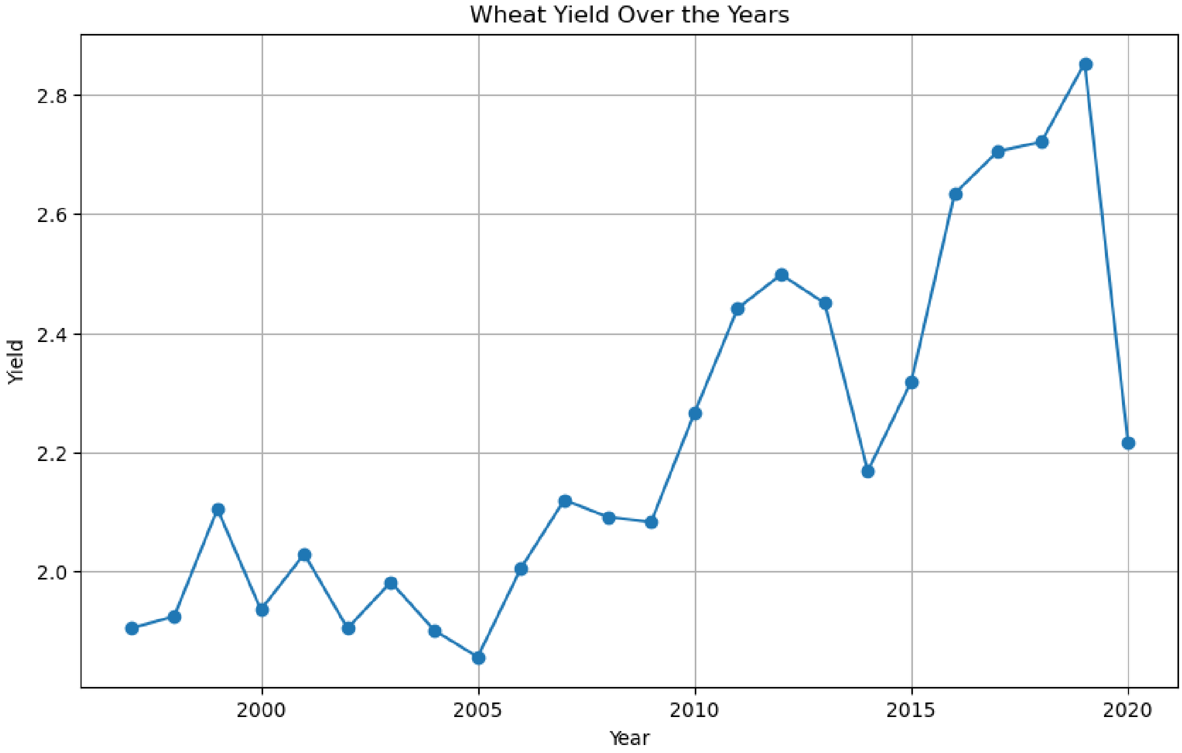
```
df.columns
```

Out[61]:
```
Index(['State', 'District', 'Crop', 'Year', 'Season', 'Area', 'Yield',
       'New Production', 'Month'],
      dtype='object')
```

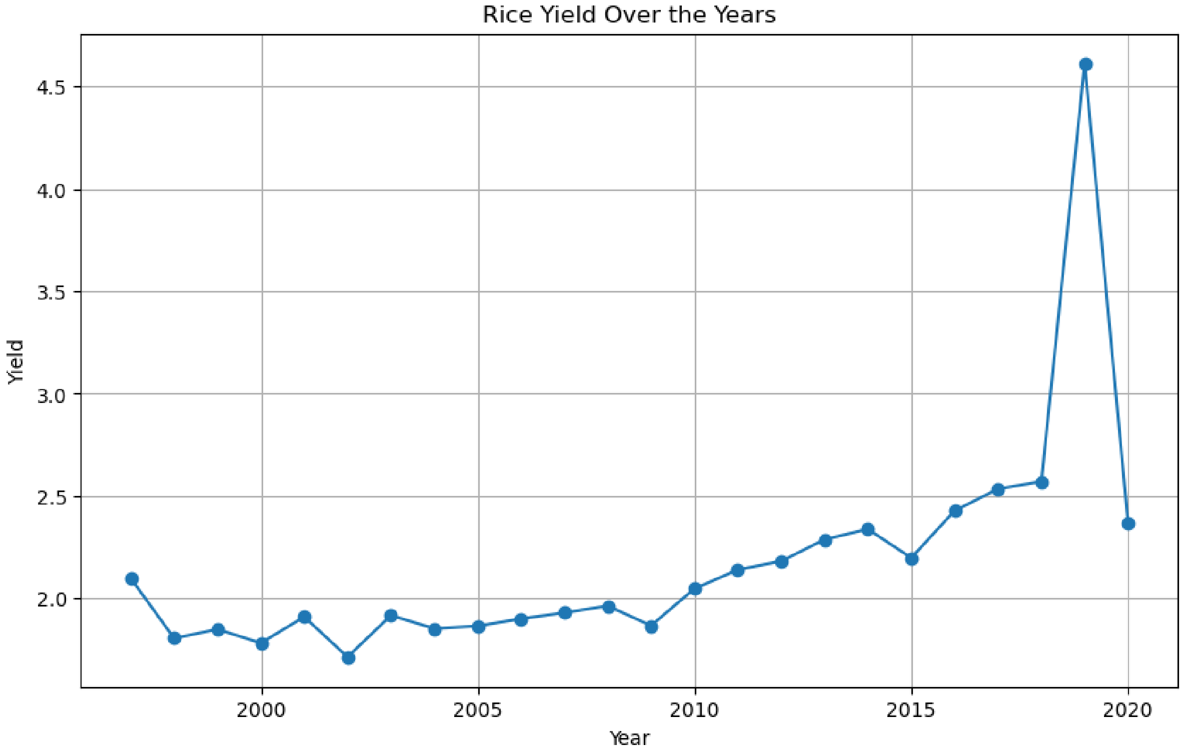
In [62]:
```python
import matplotlib.pyplot as plt
import pandas as pd

# Assuming your data is in a DataFrame named df
# Example: df = pd.read_csv('your_data.csv')

# Count the occurrences of each crop in the dataset
crop_counts = df['Crop'].value_counts()

# Plot a bar chart
plt.figure(figsize=(10, 6))
crop_counts.plot(kind='bar')
plt.title('Crop Distribution Across States')
plt.xlabel('Crop')
plt.ylabel('Count')
plt.xticks(rotation=90)
plt.show()
```



In [63]:
```python
# Assuming you want to analyze a specific crop (e.g., 'Wheat')
wheat_data = df[df['Crop'] == 'Wheat']

# Group the data by year and calculate the mean yield
yearly_yield = wheat_data.groupby('Year')['Yield'].mean()

# Plot a line chart
plt.figure(figsize=(10, 6))
yearly_yield.plot(kind='line', marker='o')
plt.title('Wheat Yield Over the Years')
plt.xlabel('Year')
plt.ylabel('Yield')
plt.grid(True)
plt.show()
```

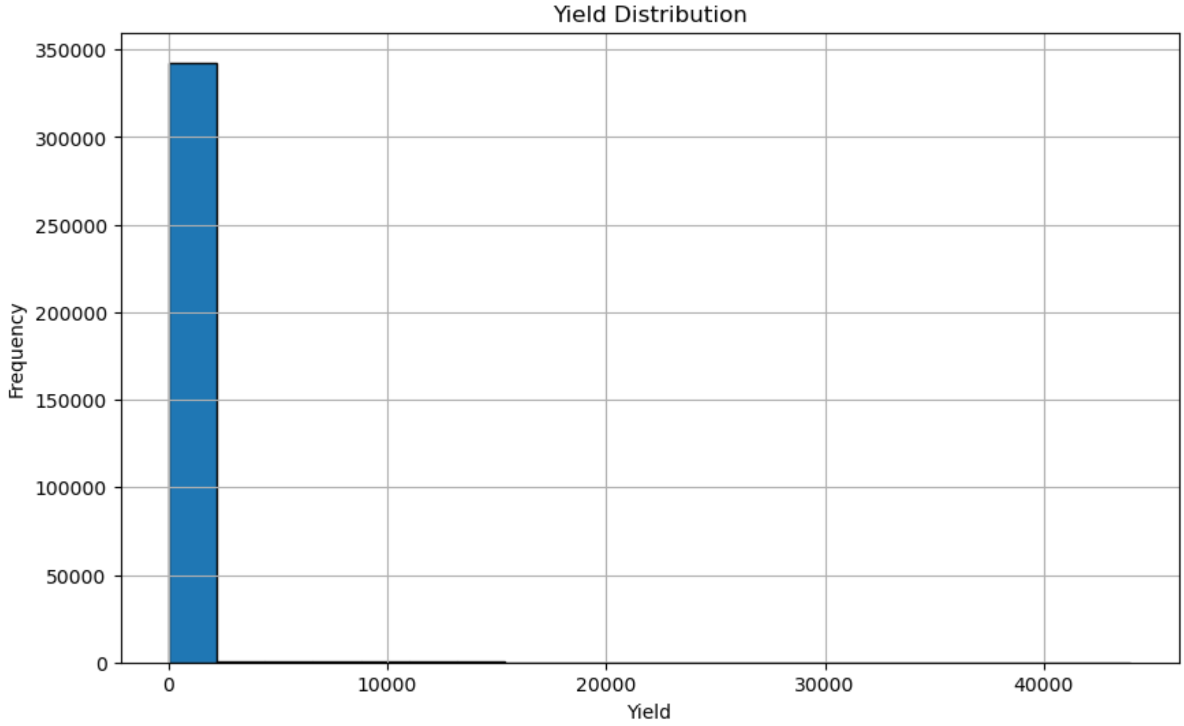


```
In [65]: # Assuming you want to analyze a specific crop (e.g., 'Wheat')
         Rice_data = df[df['Crop'] == 'Rice']

         # Group the data by year and calculate the mean yield
         yearly_yield = Rice_data.groupby('Year')['Yield'].mean()

         # Plot a line chart
         plt.figure(figsize=(10, 6))
         yearly_yield.plot(kind='line', marker='o')
         plt.title('Rice Yield Over the Years')
         plt.xlabel('Year')
         plt.ylabel('Yield')
         plt.grid(True)
         plt.show()
```



In [66]:

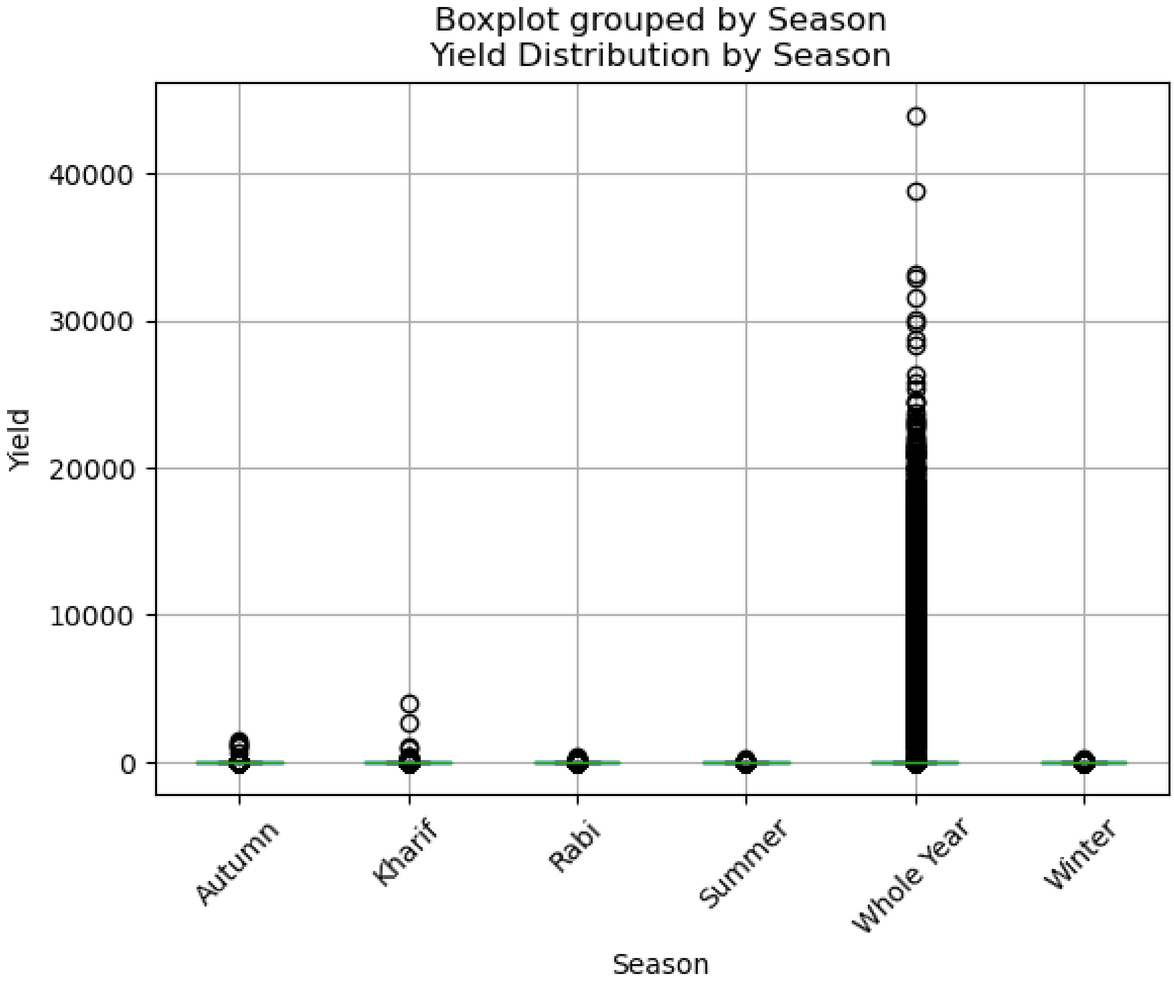
```
# Histogram
plt.figure(figsize=(10, 6))
plt.hist(df['Yield'], bins=20, edgecolor='k')
plt.title('Yield Distribution')
plt.xlabel('Yield')
plt.ylabel('Frequency')
plt.grid(True)
plt.show()
```



In [67]:

```
# Box plot
plt.figure(figsize=(10, 6))
df.boxplot(column='Yield', by='Season')
plt.title('Yield Distribution by Season')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.xticks(rotation=45)
plt.grid(True)
plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```



In [73]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler

# Assuming your data is in a DataFrame named df
# Example: df = pd.read_csv('your_data.csv')

# Preprocessing: Encoding categorical features
le = LabelEncoder()
df['Crop'] = le.fit_transform(df['Crop'])
df['Season'] = le.fit_transform(df['Season'])
df['State'] = le.fit_transform(df['State'])
df['District'] = le.fit_transform(df['District'])
df['Month'] = le.fit_transform(df['Month'])

# Feature selection: Define your features and target variable
features = ['State', 'District', 'Crop', 'Year', 'Season', 'Area', 'Month']
target = 'Yield'

X = df[features]
y = df[target]

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, ran
# Standardize the features (optional, depending on the model used)
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Create and train the machine learning model (Random Forest Regressor in th
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
Mean Squared Error: 68573.50048251107
R-squared: 0.915660975372566
```

In [71]:

```
from fbprophet import Prophet
import pandas as pd

# Prepare data for Prophet
data_prophet = df[['Year', 'Yield']].rename(columns={'Year': 'ds', 'Yield':

# Create and fit the Prophet model
model = Prophet()
model.fit(data_prophet)

# Create a future DataFrame for predictions
future = model.make_future_dataframe(periods=365)  # Extend for future predi

# Make predictions
forecast = model.predict(future)

# Evaluate the forecast (you may need to split the data for evaluation)
print(forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail())

# Plot the forecast
fig = model.plot(forecast)
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
/var/folders/8h/80kyp88j21b75qcg83zjngfm0000gn/T/ipykernel_82682/1081136150.
py in <module>
----> 1 from fbprophet import Prophet
      2 import pandas as pd
      3 
      4 # Prepare data for Prophet
      5 data_prophet = df[['Year', 'Yield']].rename(columns={'Year': 'ds',
'Yield': 'y'})

ModuleNotFoundError: No module named 'fbprophet'
```

In [72]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```
# Load your dataset
# df = pd.read_csv('your_dataset.csv')

# Encode categorical variables
encoder = LabelEncoder()
df['State'] = encoder.fit_transform(df['State'])
df['District'] = encoder.fit_transform(df['District'])
df['Crop'] = encoder.fit_transform(df['Crop'])
df['Season'] = encoder.fit_transform(df['Season'])

# Split data into features (X) and target (Y)
X = df[['State', 'District', 'Crop', 'Year', 'Season', 'Area', 'Month']]
Y = df['Yield']

# Split data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, ran

# Create and train the XGBoost Regressor
xgb_model = XGBRegressor(n_estimators=100, random_state=42)
xgb_model.fit(X_train, Y_train)

# Make predictions on the test set
Y_pred = xgb_model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(Y_test, Y_pred)
r2 = r2_score(Y_test, Y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
/var/folders/8h/80kyp88j21b75qcg83zjngfm0000gn/T/ipykernel_82682/4149804152.
py in <module>
      2 import numpy as np
      3 from sklearn.model_selection import train_test_split
----> 4 from xgboost import XGBRegressor
      5 from sklearn.metrics import mean_squared_error, r2_score
      6 from sklearn.preprocessing import LabelEncoder, StandardScaler

ModuleNotFoundError: No module named 'xgboost'
```

In [ ]:

In [ ]:

```
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense

# Load your dataset
# df = pd.read_csv('your_dataset.csv')

# Encode categorical variables
encoder = LabelEncoder()
df['State'] = encoder.fit_transform(df['State'])
df['District'] = encoder.fit_transform(df['District'])
df['Crop'] = encoder.fit_transform(df['Crop'])
df['Season'] = encoder.fit_transform(df['Season'])
```

```
# Split data into features (X) and target (Y)
X = df[['State', 'District', 'Crop', 'Year', 'Season', 'Area', 'Month']]
Y = df['Yield']

# Split data into training and testing sets
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, ran

# Standardize numerical features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Reshape data for LSTM input (assuming you have time series data)
X_train = np.reshape(X_train, (X_train.shape[0], 1, X_train.shape[1]))
X_test = np.reshape(X_test, (X_test.shape[0], 1, X_test.shape[1]))

# Create and train the LSTM model
model = Sequential()
model.add(LSTM(50, input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mean_squared_error')
model.fit(X_train, Y_train, epochs=50, batch_size=32, validation_data=(X_tes

# Make predictions on the test set
Y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(Y_test, Y_pred)
r2 = r2_score(Y_test, Y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

In [ ]:

In [ ]: