Project 2: Crop Production Analysis in India

Project Title	Crop Production Analysis in India
Tools	Jupyter Notebook, Spyder, Excel
Technologies	Data Science
Domain	Agriculture

Problem Statement:

The Agriculture business domain, as a vital part of the overall supply chain, is expected to highly evolve in the upcoming years via the developments, which are taking place on the side of the Future Internet. This paper presents a novel Business-to-Business collaboration platform from the agri-food sector perspective, which aims to facilitate the collaboration of numerous stakeholders belonging to associated business domains, in an effective and flexible manner.
This dataset provides a huge amount of information on crop production in India ranging from several years. Based on the Information the ultimate goal would be to predict crop production and find important insights highlighting key indicators and metrics that influence crop production.

Tools:

- Jupyter Notebook:

Jupyter Notebook was primarily used for data cleaning, exploratory data analysis (EDA), and visualization. Its interactive environment allowed for efficient step-by-step exploration of the dataset, making it easier to identify trends, patterns, and potential data issues. The flexibility of integrating code, markdown, and visual output in the same interface helped in creating a comprehensive narrative of the analysis process.

- Spyder:

Spyder was used for model building and performance evaluation. With its powerful debugging tools and integrated development environment (IDE) for Python, Spyder enabled the implementation of machine learning algorithms and allowed for effective tuning of model parameters. It was particularly useful for writing and executing larger codebases focused on optimizing and refining the prediction models.

1) Data Cleaning:

In the data cleaning process, I began by exploring the dataset to understand its structure and addressed missing values by removing rows with missing "Production" data. I then created a new column for "Yield," calculated as the ratio of production to the area. To ensure consistency, I cleaned the crop names by removing extra spaces and categorized them into broader groups like cereals, fruits, and vegetables. This categorization helped organize the data for easier analysis, and I saved the cleaned dataset for further use.

<u>DataCleaning.ipynb:</u>

```
[5]: import numpy as np
     import pandas as pd
```

```
[6]: df = pd.read_csv("Crop Production data.csv")
```

```
[7]: df.head()
```

[7]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
0	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Arecanut	1254.0	2000.0
1	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Other Kharif pulses	2.0	1.0
2	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Rice	102.0	321.0
3	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Banana	176.0	641.0
4	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Cashewnut	720.0	165.0

```
[9]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 246091 entries, 0 to 246090
Data columns (total 7 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   State_Name     246091 non-null  object
 1   District_Name  246091 non-null  object
 2   Crop_Year      246091 non-null  int64
 3   Season         246091 non-null  object
 4   Crop           246091 non-null  object
 5   Area           246091 non-null  float64
 6   Production     242361 non-null  float64
dtypes: float64(2), int64(1), object(4)
memory usage: 13.1+ MB
```

```
[10]: df.describe()
```

[10]:

	Crop_Year	Area	Production
count	246091.000000	2.460910e+05	2.423610e+05
mean	2005.643018	1.200282e+04	5.825034e+05
std	4.952164	5.052340e+04	1.706581e+07
min	1997.000000	4.000000e-02	0.000000e+00
25%	2002.000000	8.000000e+01	8.800000e+01
50%	2006.000000	5.820000e+02	7.290000e+02
75%	2010.000000	4.392000e+03	7.023000e+03
max	2015.000000	8.580100e+06	1.250800e+09

```
[6]: df.isnull().sum()
```

```
[6]: State_Name          0
     District_Name       0
     Crop_Year           0
     Season              0
     Crop                0
     Area                0
     Production       3730
     dtype: int64
```

[12]:

```
df = df.dropna(subset=["Production"])
```

[13]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 242361 entries, 0 to 246090
Data columns (total 7 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   State_Name     242361 non-null  object
 1   District_Name  242361 non-null  object
 2   Crop_Year      242361 non-null  int64
 3   Season         242361 non-null  object
 4   Crop           242361 non-null  object
 5   Area           242361 non-null  float64
 6   Production     242361 non-null  float64
dtypes: float64(2), int64(1), object(4)
memory usage: 14.8+ MB
```

[14]:

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

[10]:

```
print(df["Crop"].unique())
```

```
['Arecanut' 'Other Kharif pulses' 'Rice' 'Banana' 'Cashewnut' 'Coconut '
 'Dry ginger' 'Sugarcane' 'Sweet potato' 'Tapioca' 'Black pepper'
 'Dry chillies' 'other oilseeds' 'Turmeric' 'Maize' 'Moong(Green Gram)'
 'Urad' 'Arhar/Tur' 'Groundnut' 'Sunflower' 'Bajra' 'Castor seed'
 'Cotton(lint)' 'Horse-gram' 'Jowar' 'Korra' 'Ragi' 'Tobacco' 'Gram'
 'Wheat' 'Masoor' 'Sesamum' 'Linseed' 'Safflower' 'Onion'
 'other misc. pulses' 'Samai' 'Small millets' 'Coriander' 'Potato'
 'Other  Rabi pulses' 'Soyabean' 'Beans & Mutter(Vegetable)' 'Bhindi'
 'Brinjal' 'Citrus Fruit' 'Cucumber' 'Grapes' 'Mango' 'Orange'
 'other fibres' 'Other Fresh Fruits' 'Other Vegetables' 'Papaya'
 'Pome Fruit' 'Tomato' 'Mesta' 'Cowpea(Lobia)' 'Lemon' 'Pome Granet'
 'Sapota' 'Cabbage' 'Rapeseed &Mustard' 'Peas  (vegetable)' 'Niger seed'
 'Bottle Gourd' 'Varagu' 'Garlic' 'Ginger' 'Oilseeds total' 'Pulses total'
 'Jute' 'Peas & beans (Pulses)' 'Blackgram' 'Paddy' 'Pineapple' 'Barley'
 'Sannhamp' 'Khesari' 'Guar seed' 'Moth' 'Other Cereals & Millets'
 'Cond-spcs other' 'Turnip' 'Carrot' 'Redish' 'Arcanut (Processed)'
 'Atcanut (Raw)' 'Cashewnut Processed' 'Cashewnut Raw' 'Cardamom' 'Rubber'
 'Bitter Gourd' 'Drum Stick' 'Jack Fruit' 'Snak Guard' 'Tea' 'Coffee'
 'Cauliflower' 'Other Citrus Fruit' 'Water Melon' 'Total foodgrain'
 'Kapas' 'Colocosia' 'Lentil' 'Bean' 'Jobster' 'Perilla' 'Rajmash Kholar'
 'Ricebean (nagadal)' 'Ash Gourd' 'Beet Root' 'Lab-Lab' 'Ribed Guard'
 'Yam' 'Pump Kin' 'Apple' 'Peach' 'Pear' 'Plums' 'Litchi' 'Ber'
 'Other Dry Fruit' 'Jute & mesta']
```

[17]:

```
df['Crop'] = df['Crop'].str.strip()
```

[18]:

```
# Define a dictionary to map crops to categories
crop_categories = {
    'Arecanut': 'Nuts',
    'Other Kharif pulses': 'Pulses',
    'Rice': 'Cereals',
    'Banana': 'Fruits',
    'Cashewnut': 'Nuts',
    'Coconut': 'Nuts',
    'Dry ginger': 'Spices'
# Create a new column in the dataframe for crop categories
df['Crop_Category'] = df['Crop'].map(crop_categories)

df[['Crop', 'Crop_Category']].head()
```

[18]:

	Crop	Crop_Category
0	Arecanut	Nuts
1	Other Kharif pulses	Pulses
2	Rice	Cereals
3	Banana	Fruits
4	Cashewnut	Nuts

[19]:

```
print(df["Crop"].nunique())
print(df["Crop_Category"].nunique())
```

```
124
10
```

[20]:

```
print(df.isnull().sum())
print(df.info())
```

```
State_Name       0
District_Name    0
Crop_Year        0
Season           0
Crop             0
Area             0
Production       0
Yield            0
Crop_Category    0
dtype: int64
<class 'pandas.core.frame.DataFrame'>
Index: 242361 entries, 0 to 246090
Data columns (total 9 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   State_Name     242361 non-null  object
 1   District_Name  242361 non-null  object
 2   Crop_Year      242361 non-null  int64
 3   Season         242361 non-null  object
 4   Crop           242361 non-null  object
 5   Area           242361 non-null  float64
 6   Production     242361 non-null  float64
 7   Yield          242361 non-null  float64
 8   Crop_Category  242361 non-null  object
dtypes: float64(3), int64(1), object(5)
memory usage: 18.5+ MB
```

[21]:
```
df.head(10)
```

[21]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
0	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Arecanut	1254.0	2000.0	1.594896	Nuts
1	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Other Kharif pulses	2.0	1.0	0.500000	Pulses
2	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Rice	102.0	321.0	3.147059	Cereals
3	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Banana	176.0	641.0	3.642045	Fruits
4	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Cashewnut	720.0	165.0	0.229167	Nuts
5	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Coconut	18168.0	65100000.0	3583.223250	Nuts
6	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Dry ginger	36.0	100.0	2.777778	Spices
7	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sugarcane	1.0	2.0	2.000000	Cash Crops
8	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sweet potato	5.0	15.0	3.000000	Vegetables
9	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Tapioca	40.0	169.0	4.225000	Vegetables

[16]:
```
df.to_csv("Cleaned_Crop_Production.csv", index=False)
```

2) Exploratory data analysis (EDA):

Overall Analysis:

❖ State-wise Insights:

- **Kerala** has demonstrated the **highest** overall crop **production**, highlighting its significant role in India's agricultural output.
- **Puducherry** is notable for achieving the **highest yield** among states, reflecting efficient crop production practices in this region.

❖ Year-wise Highlights:

- The year **2011** stands out with the **highest production** and **yield**, indicating a peak period for agricultural performance across various crops.

❖ Crop Categories:

- **Nuts**: **Coconut** leads both in production and yield within the nuts category.
- **Pulses**: **Pulsestotal** represents the highest production within the pulses category.
- **Cereals**: **Totalfoodgrain** is the top-performing category in cereals.
- **Fruits**: **Banana** tops the production charts in the fruits category.
- **Spices**: **Ginger** is the leading spice crop in terms of production.
- **Cash Crops**: **Sugarcane** stands out as the highest in production among cash crops.
- **Vegetables**: **Tapioca** is the leading vegetable crop in terms of production.
- **Oilseeds**: **Oilseedstotal** shows the highest production figures in the oilseeds category.
- **Fiber Crops**: **Jute** is the top fiber crop.
- **Unknown**: **Jobster** is categorized under unknown crops but shows notable figures.

❖ Crop Category Specific Analysis:

-Rice:
Season: Primarily produced during the **Rabi** season.
Production: Highest **throughout the year.**
State: **Chandigarh** leads in **yield**, and **Punjab** in **production**.
Year: **2014** marks the peak for both **yield** and **production**.

-Wheat:
Season: Majorly grown in the **Rabi** season.
Production: High **throughout the year.**
State: **Chandigarh** leads in **yield**, and **Punjab** in **production**.
Year: **2012** shows the highest **yield**, and **2011** the highest **production**.

-Potato:
Season: Predominantly cultivated during the **Rabi** season.
Production: Significant figures **throughout the year.**
State: **West Bengal** excels in both **yield** and **production**.
Year: **2012** records the highest **yield**, and **2004** the highest **production**.

-Coconut:
Season: Grown **throughout the year.**
State: **Puducherry** leads in **yield**, and **Kerala** in **production**.
Year: **2011** sees the peak **yield**, and **2014** the highest **production**.

-Maize:
Season: Mainly produced during the **Autumn** and **Kharif** seasons.
State: **Maharashtra** leads in **yield**, and **Telangana** in **production**.
Year: **1991** shows the **highest yield**, with a **significant decline in subsequent years** **2013** records highest **production**.

[1]:
```
import pandas as pd
import seaborn as sns

import matplotlib.pyplot as plt
```

[2]:
```
df = pd.read_csv("Cleaned_Crop_Production.csv")
```

[3]:
```
df.describe()
```

[3]:

	Crop_Year	Area	Production	Yield
count	242361.000000	2.423610e+05	2.423610e+05	242361.000000
mean	2005.625773	1.216741e+04	5.825034e+05	41.649059
std	4.958285	5.085744e+04	1.706581e+07	817.572839
min	1997.000000	1.000000e-01	0.000000e+00	0.000000
25%	2002.000000	8.700000e+01	8.800000e+01	0.513514
50%	2006.000000	6.030000e+02	7.290000e+02	1.000000
75%	2010.000000	4.545000e+03	7.023000e+03	2.355450
max	2015.000000	8.580100e+06	1.250800e+09	88000.000000

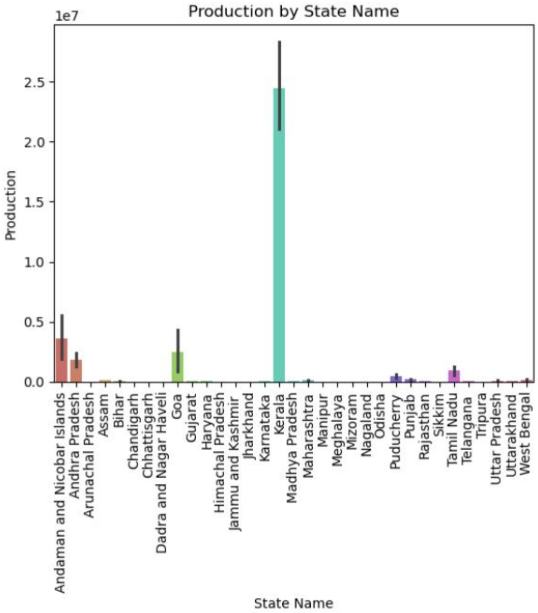
[4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 242361 entries, 0 to 242360
Data columns (total 9 columns):
 #   Column         Non-Null Count   Dtype  
---  ------         --------------   -----  
 0   State_Name     242361 non-null  object 
 1   District_Name  242361 non-null  object 
 2   Crop_Year      242361 non-null  int64  
 3   Season         242361 non-null  object 
 4   Crop           242361 non-null  object 
 5   Area           242361 non-null  float64
 6   Production     242361 non-null  float64
 7   Yield          242361 non-null  float64
 8   Crop_Category  242361 non-null  object 
dtypes: float64(3), int64(1), object(5)
memory usage: 16.6+ MB
```

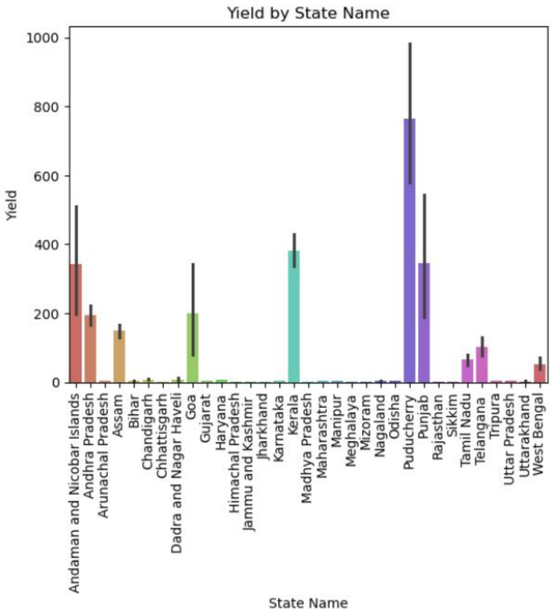
[84]:
```
df['Crop_Category'].value_counts()
```

[84]:
```
Crop_Category
Cereals        63487
Pulses         60272
Oilseeds       45752
Vegetables     25460
Spices         18404
Cash Crops     10561
Fibre Crops     7602
Fruits          6154
Nuts            4660
Unknown            9
Name: count, dtype: int64
```

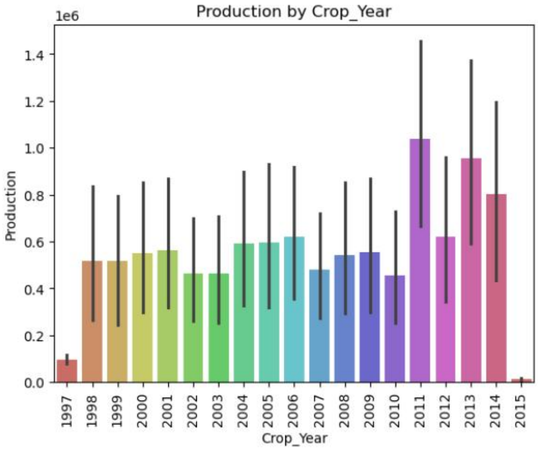
[13]:
```
sns.barplot(x="State_Name", y="Production", data=df,palette="hls", hue="State_Name", legend= False)
plt.xticks(rotation=90)
plt.title('Production by State Name')
plt.ylabel('Production')
plt.xlabel('State Name')
plt.show()
```



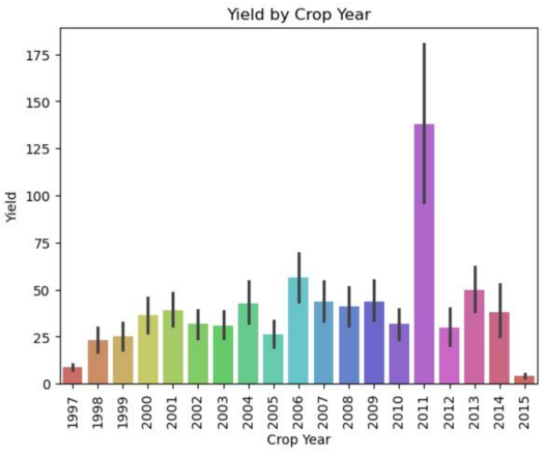
[14]:
```
sns.barplot(x="State_Name", y="Yield", data=df,palette="hls", hue="State_Name", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by State Name')
plt.xlabel('State Name')
plt.ylabel('Yield')
plt.show()
```



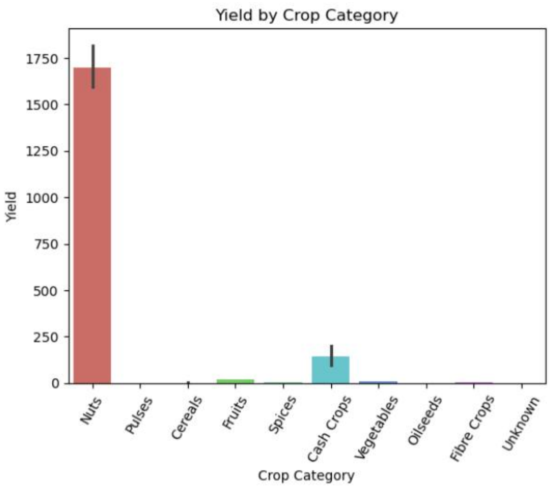
[15]:
```python
sns.barplot(x="Crop_Year", y="Production", data=df,palette="hls", hue="Crop_Year", legend= False)
plt.xticks(rotation=90)
plt.title('Production by Crop_Year')
plt.xlabel('Crop_Year')
plt.ylabel('Production')
plt.show()
```



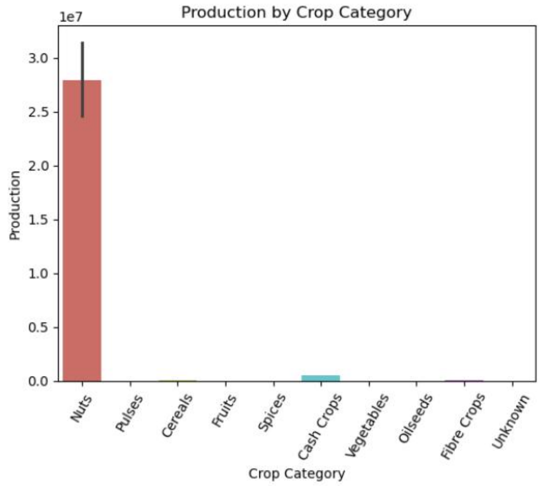
[16]:
```python
sns.barplot(x="Crop_Year", y="Yield", data=df,palette="hls", hue="Crop_Year", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[17]:
```python
sns.barplot(x="Crop_Category", y="Yield", data=df,palette="hls", hue="Crop_Category", legend= False)
plt.xticks(rotation=60)
plt.title('Yield by Crop Category')
plt.xlabel('Crop Category')
plt.ylabel('Yield')
plt.show()
```

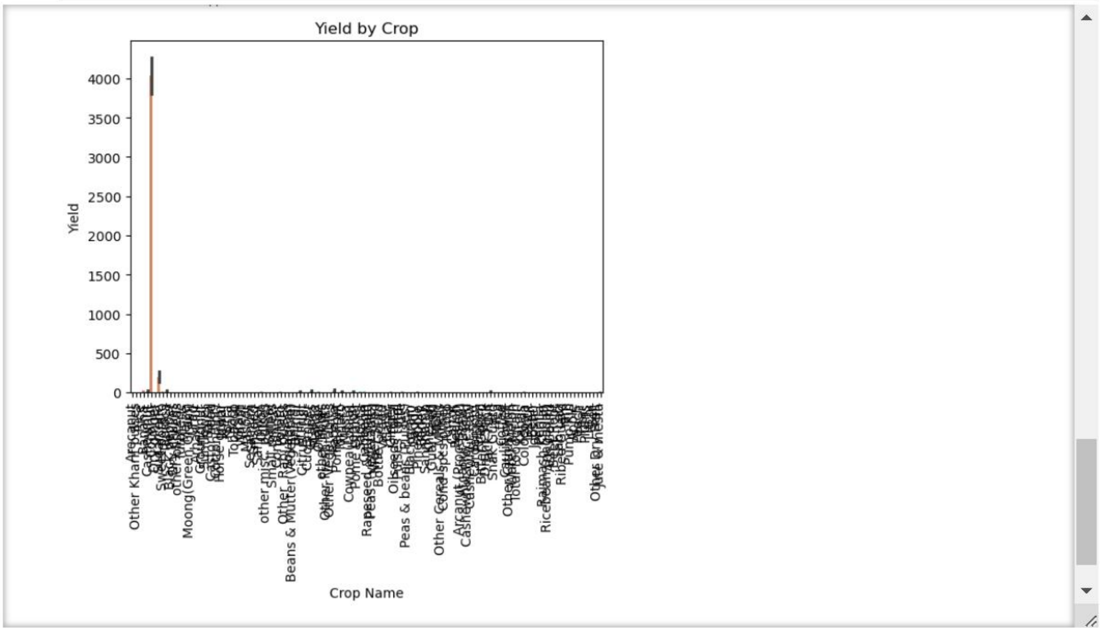


```
[18]: sns.barplot(x="Crop_Category", y="Production", data=df,palette="hls", hue="Crop_Category", legend= False)
plt.xticks(rotation=60)
plt.title('Production by Crop Category')
plt.xlabel('Crop Category')
plt.ylabel('Production')
plt.show()
```



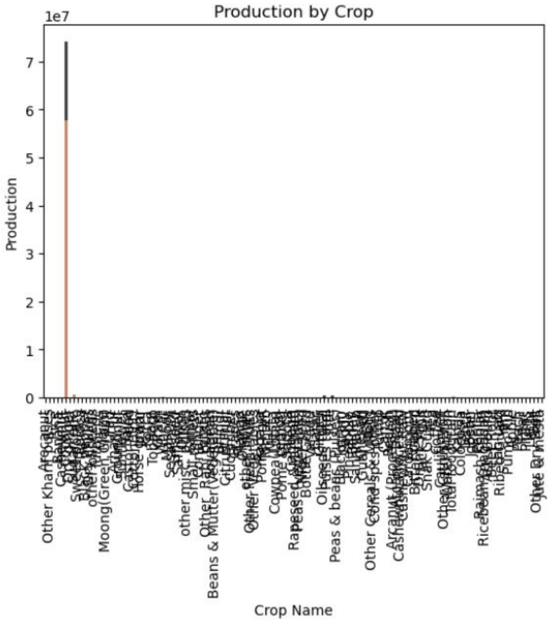

```
[91]: crop_yield = df.groupby('Crop', as_index=False)['Yield'].sum()
crop_yield['Yield'] = crop_yield['Yield'].astype(int)
sorted_crop_yield = crop_yield.sort_values(by='Yield', ascending=False)
pd.set_option('display.max_rows', None)
pd.set_option('display.float_format', '{:.0f}'.format)
print(sorted_crop_yield.to_string(index=False))

sns.barplot(x="Crop", y="Yield", data=df,palette="hls", hue="Crop", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by Crop')
plt.xlabel('Crop Name')
plt.ylabel('Yield')
plt.show()
```

```
         Crop   Yield
      Coconut 7911110
    Sugarcane 1526291
       Potato   88962
       Banana   84793
        Onion   82797
 Sweet potato   36471
        Maize   36297
         Rice   30088
      Tapioca   29737
        Wheat   16482
   Dry ginger   16038
       Papaya   13842
         Jute   13314
     Turmeric   11486
       Garlic   10707
    Groundnut   10480
    Cashewnut   10333
        Mesta    9853
  Cotton(lint)    9041
  Dry chillies    8812
       Barley    7502
        Jowar    7329
```



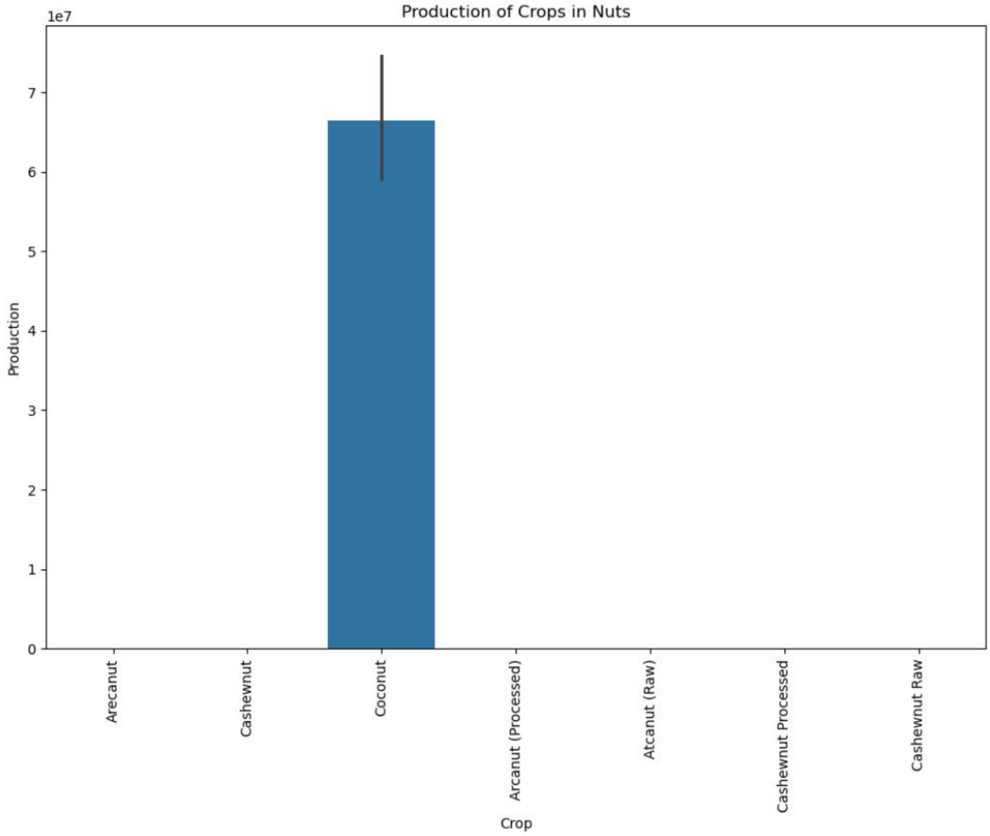
[19]:
```
crop_production = df.groupby('Crop', as_index=False)['Production'].sum()
sorted_crop_production = crop_production.sort_values(by='Production', ascending=False)
print(sorted_crop_production[['Crop', 'Production']])

sns.barplot(x="Crop", y="Production", data=df,palette="hls", hue="Crop", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by Crop')
plt.xlabel('Crop Name')
plt.ylabel('Yield')
plt.show()
```

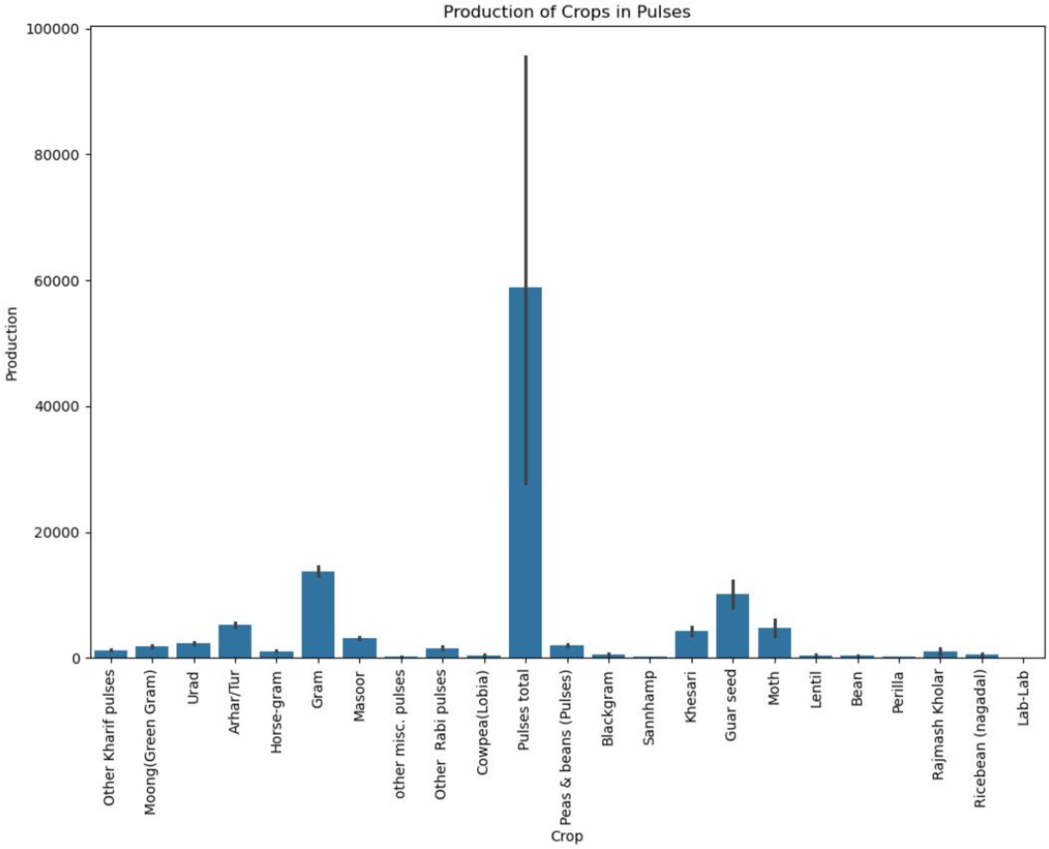
```
                  Crop    Production
28             Coconut  1.299816e+11
106          Sugarcane  5.535682e+09
95                Rice  1.605470e+09
119              Wheat  1.332826e+09
87              Potato  4.248263e+08
..                 ...           ...
71  Other Citrus Fruit  0.000000e+00
35            Cucumber  0.000000e+00
58              Litchi  0.000000e+00
54             Lab-Lab  0.000000e+00
0                Apple  0.000000e+00

[124 rows x 2 columns]
```

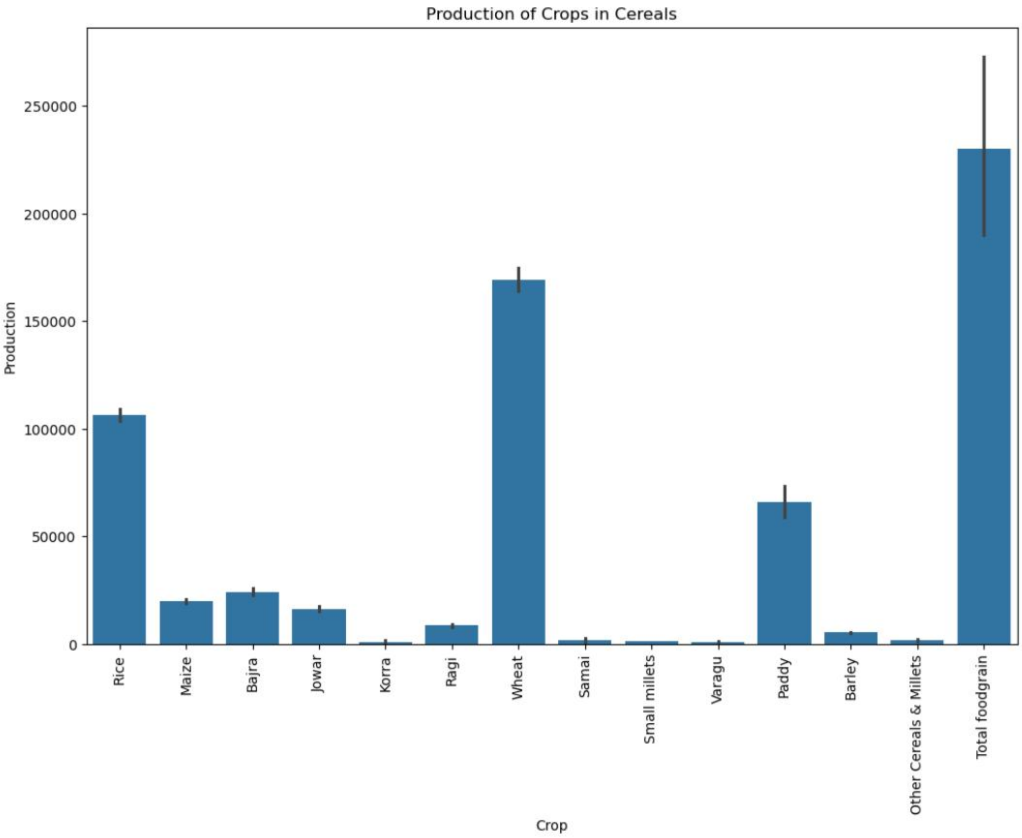


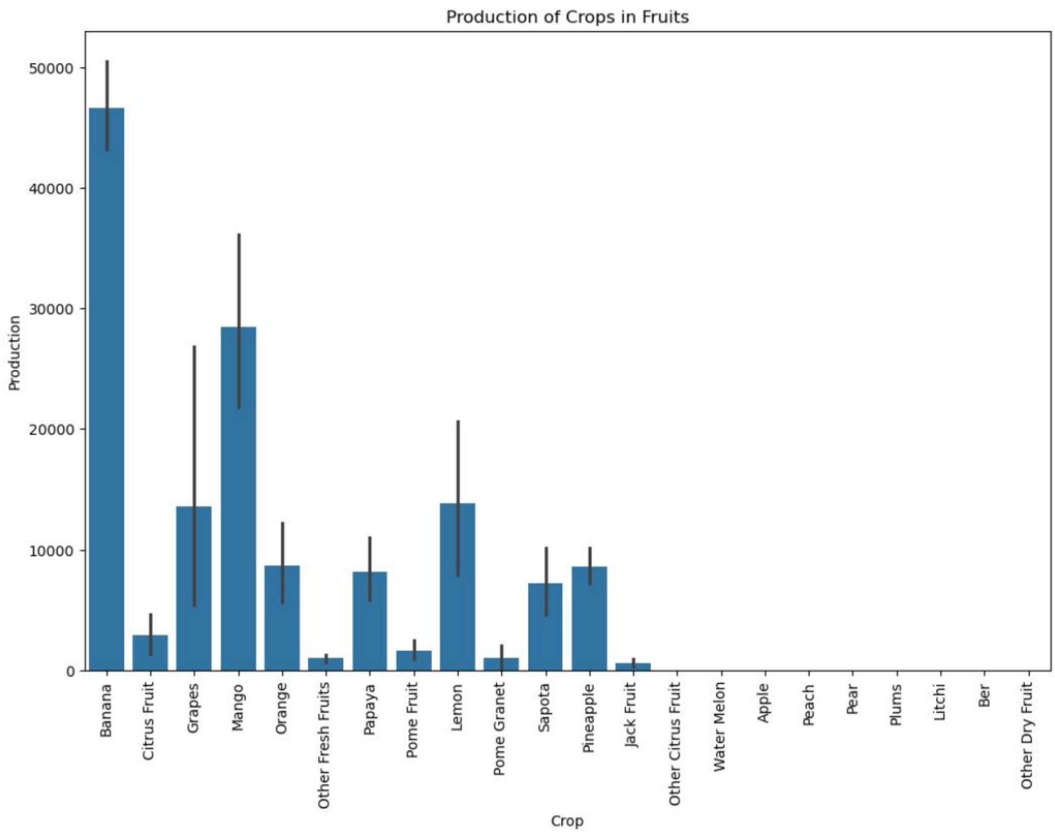
[96]:

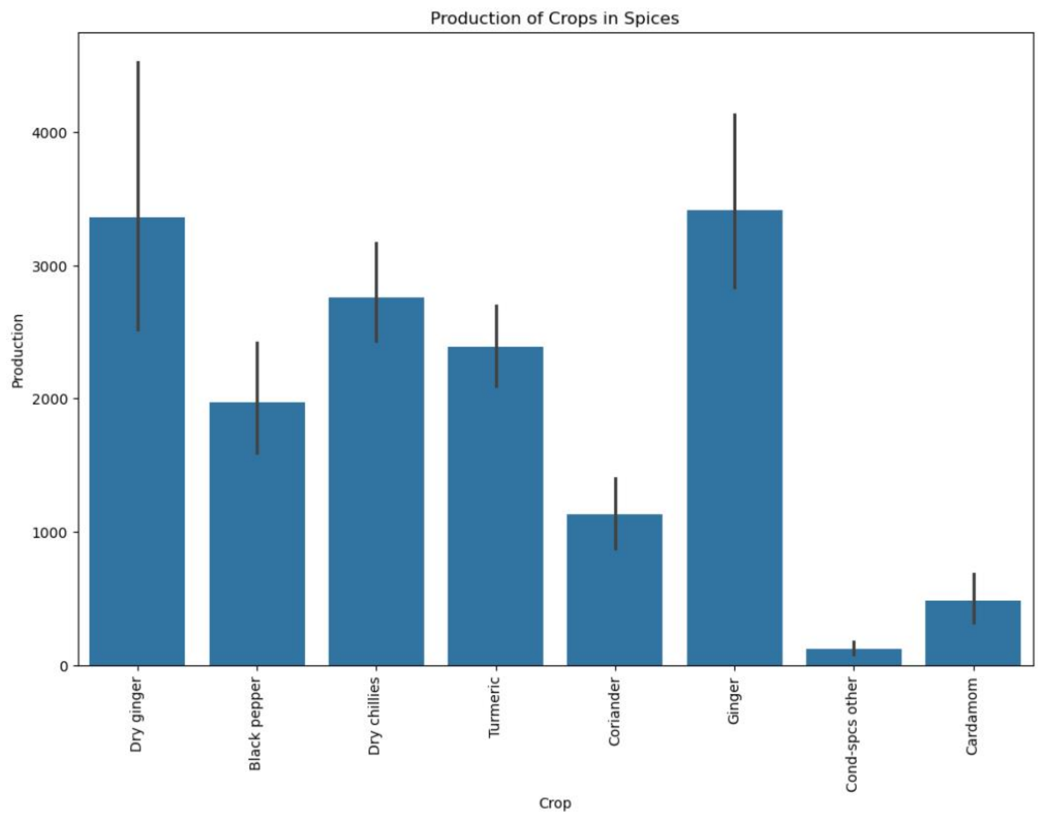
```
crop_categories = df['Crop_Category'].unique()

# Plot for each crop category
for category in crop_categories:
    plt.figure(figsize=(12, 8))
    category_data = df[df['Crop_Category'] == category]

    # Optional: aggregate production if needed
    # e.g., to show average production per crop
    # category_data = category_data.groupby('Crop').agg({'Production': 'mean'}).reset_index()

    sns.barplot(x='Crop', y='Production', data=category_data)
    plt.title(f'Production of Crops in {category}')
    plt.xlabel('Crop')
    plt.ylabel('Production')
    plt.xticks(rotation=90)  # Rotate x labels for better readability
    plt.show()
```

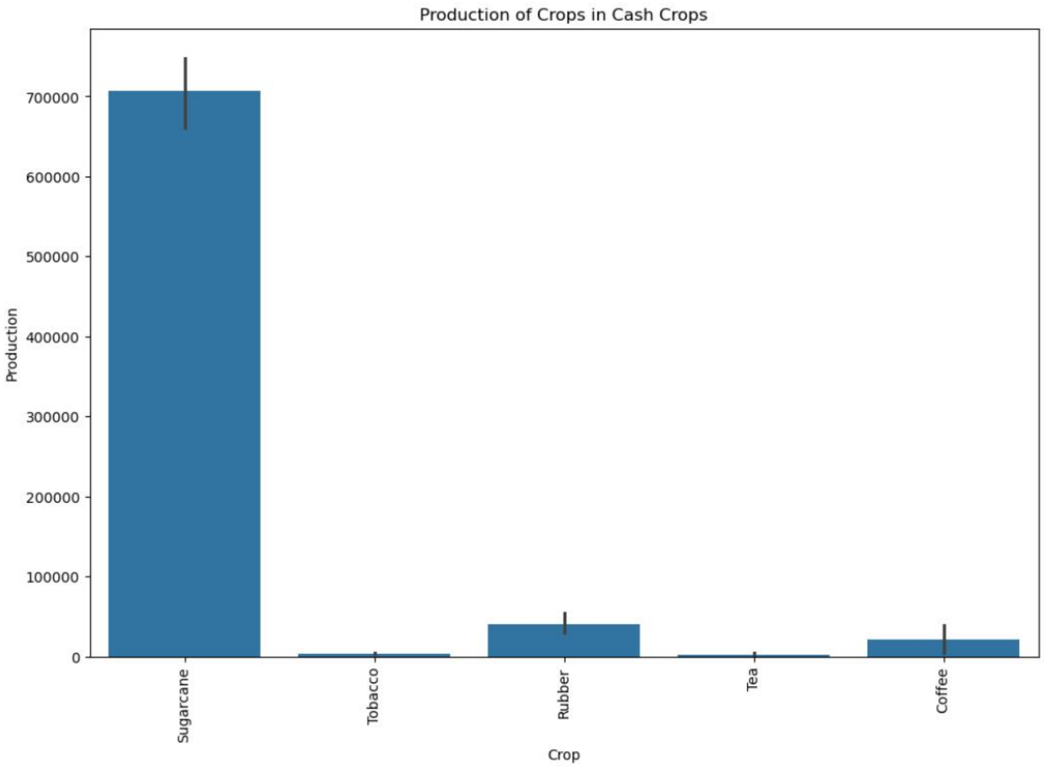


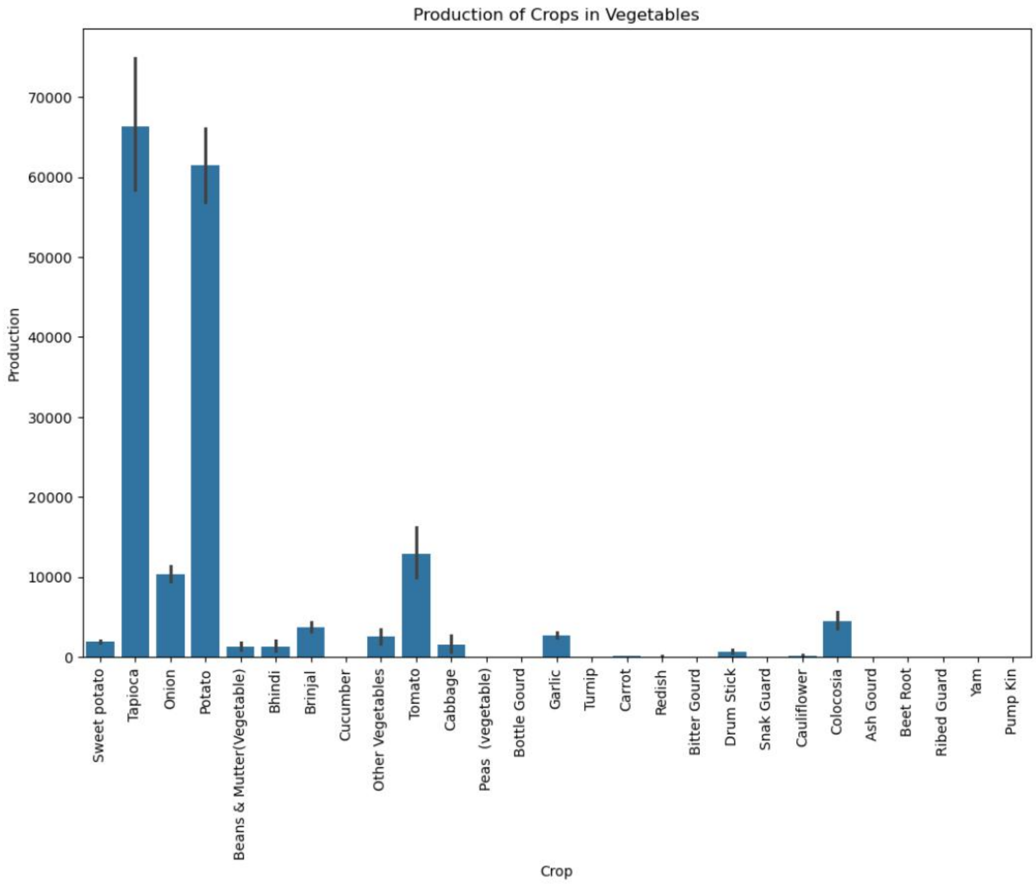

Production of Crops in Pulses
Production
Crop
Other Kharif pulses
Moong(Green Gram)
Urad
Arhar/Tur
Horse-gram
Gram
Masoor
other misc. pulses
Other Rabi pulses
Cowpea(Lobia)
Pulses total
Peas & beans (Pulses)
Blackgram
Sannhamp
Khesari
Guar seed
Moth
Lentil
Bean
Perilla
Rajmash Kholar
Ricebean (nagadal)
Lab-Lab

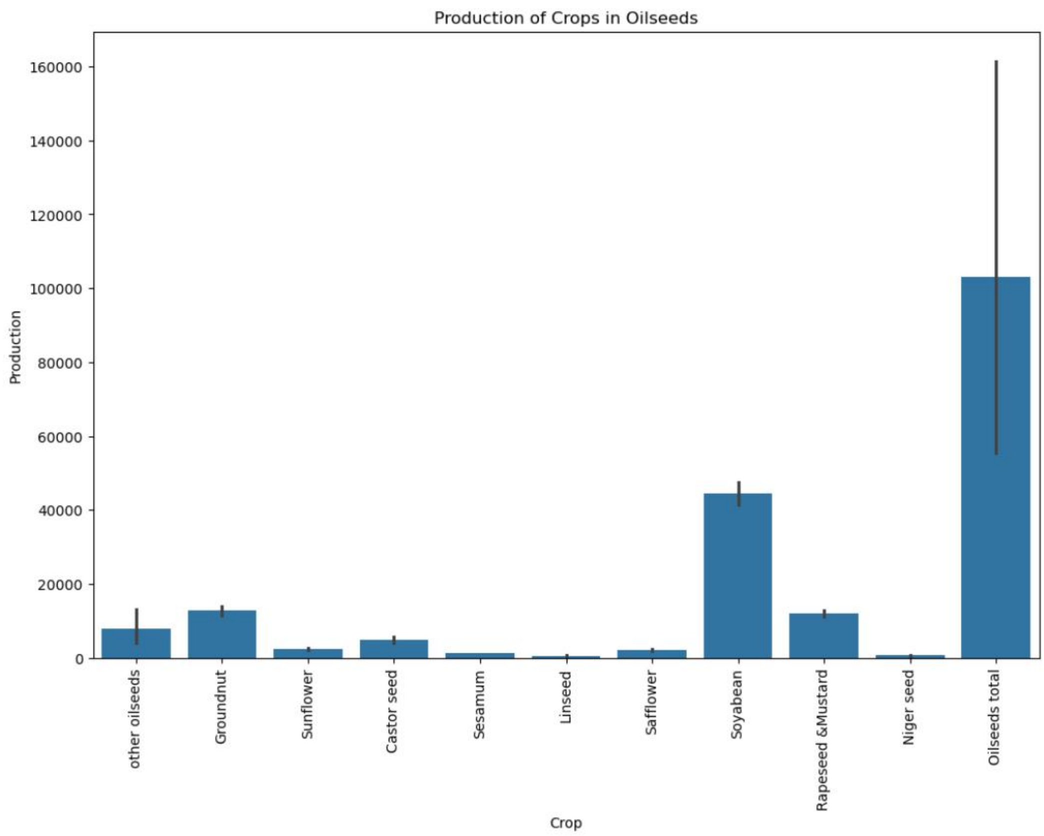

Production of Crops in Cereals
Production
Crop
Rice
Maize
Bajra
Jowar
Korra
Ragi
Wheat
Samai
Small millets
Varagu
Paddy
Barley
Other Cereals & Millets
Total foodgrain


Production of Crops in Fruits
Production
0
10000
20000
30000
40000
50000
Banana
Citrus Fruit
Grapes
Mango
Orange
Other Fresh Fruits
Papaya
Pome Fruit
Lemon
Pome Granet
Sapota
Pineapple
Jack Fruit
Other Citrus Fruit
Water Melon
Apple
Peach
Pear
Plums
Litchi
Ber
Other Dry Fruit
Crop

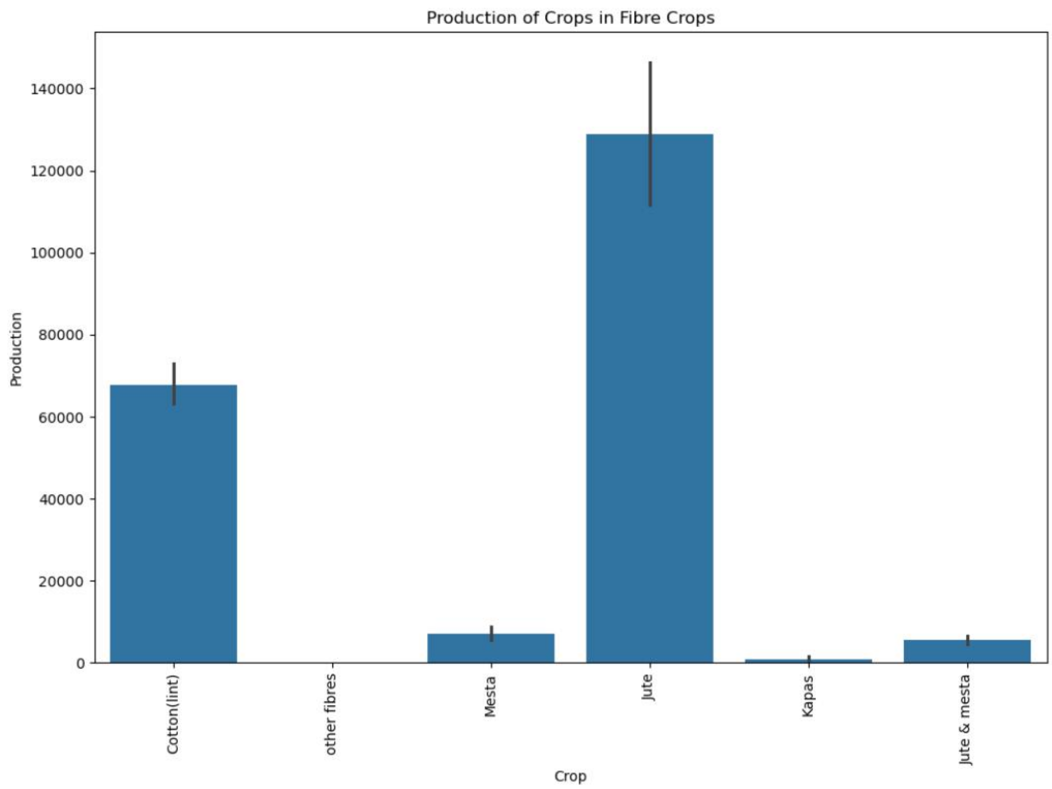

Production of Crops in Spices
Production
0
1000
2000
3000
4000
Dry ginger
Black pepper
Dry chillies
Turmeric
Coriander
Ginger
Cond-spcs other
Cardamom
Crop


Production of Crops in Cash Crops
Production
700000
600000
500000
400000
300000
200000
100000
0
Sugarcane
Tobacco
Rubber
Tea
Coffee
Crop

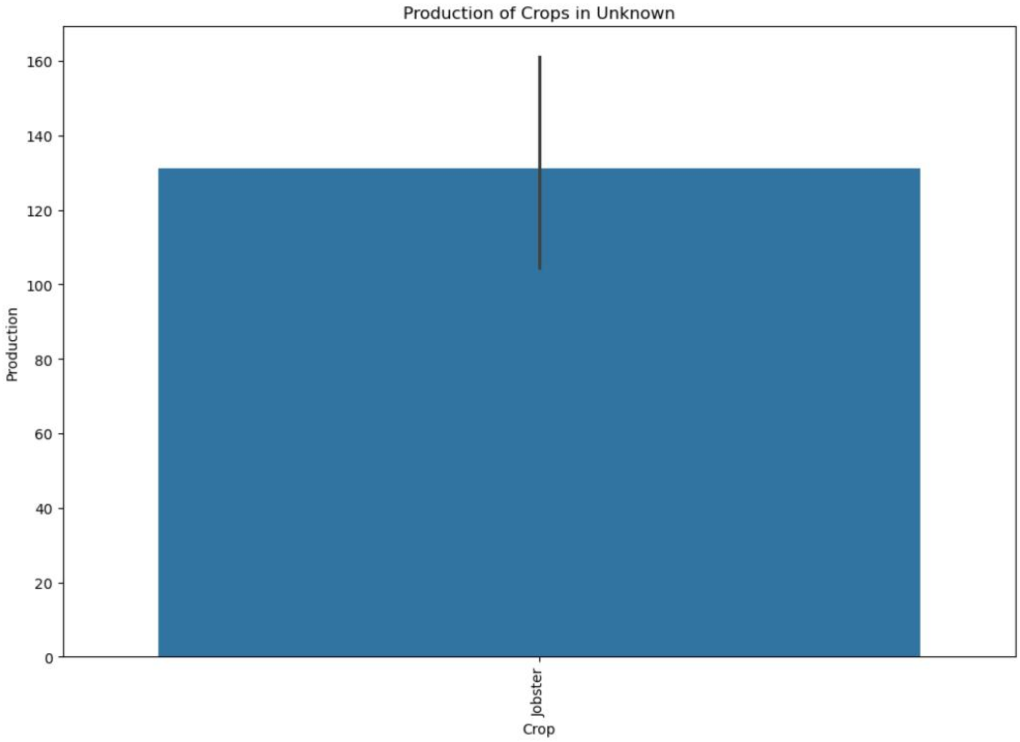

Production of Crops in Vegetables
Production
70000
60000
50000
40000
30000
20000
10000
0
Sweet potato
Tapioca
Onion
Potato
Beans & Mutter(Vegetable)
Bhindi
Brinjal
Cucumber
Other Vegetables
Tomato
Cabbage
Peas (vegetable)
Bottle Gourd
Garlic
Turnip
Carrot
Redish
Bitter Gourd
Drum Stick
Snak Guard
Cauliflower
Colocosia
Ash Gourd
Beet Root
Ribed Guard
Yam
Pump Kin
Crop



Production of Crops in Oilseeds
Production
Crop
0
20000
40000
60000
80000
100000
120000
140000
160000
other oilseeds
Groundnut
Sunflower
Castor seed
Sesamum
Linseed
Safflower
Soyabean
Rapeseed &Mustard
Niger seed
Oilseeds total

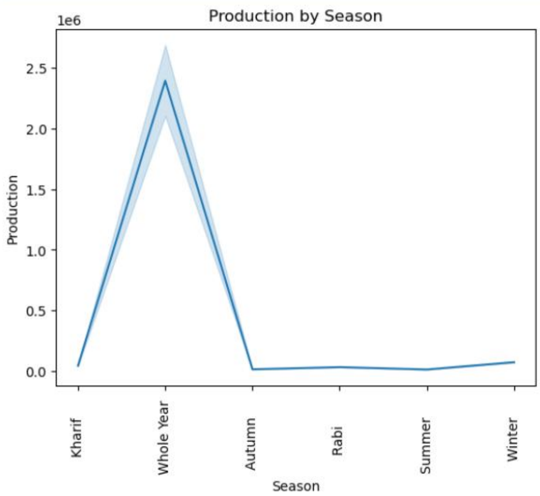


Production of Crops in Fibre Crops
Production
Crop
0
20000
40000
60000
80000
100000
120000
140000
Cotton(lint)
other fibres
Mesta
Jute
Kapas
Jute & mesta

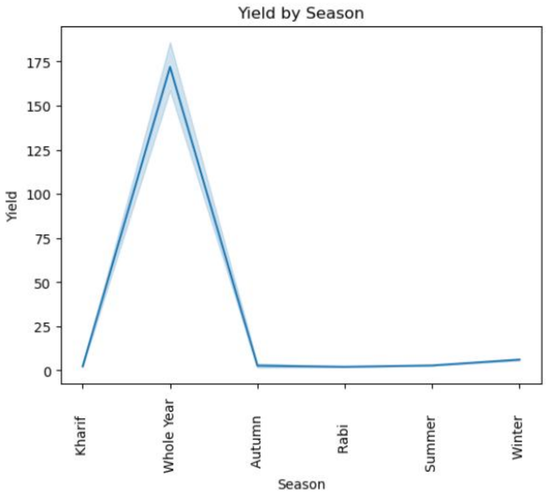



```
[21]: crop_categories = df['Crop_Category'].unique()

# Plot for each crop category
for category in crop_categories:
    plt.figure(figsize=(12, 8))
    category_data = df[df['Crop_Category'] == category]

    # Optional: aggregate production if needed
    # e.g., to show average production per crop
    # category_data = category_data.groupby('Crop').agg({'Production': 'mean'}).reset_index()

    sns.barplot(x='Crop', y='Yield', data=category_data)
    plt.title(f'Production of Crops in {category}')
    plt.xlabel('Crop')
    plt.ylabel('Production')
    plt.xticks(rotation=90)  # Rotate x labels for better readability
    plt.show()
```

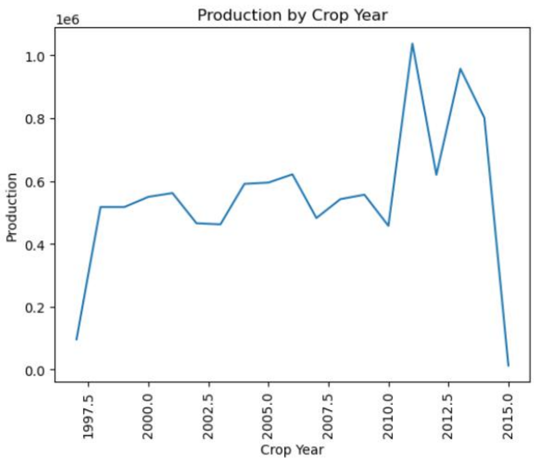
```
[23]: sns.lineplot(x="Season", y="Production", data=df)
plt.xticks(rotation=90)
plt.title('Production by Season')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```



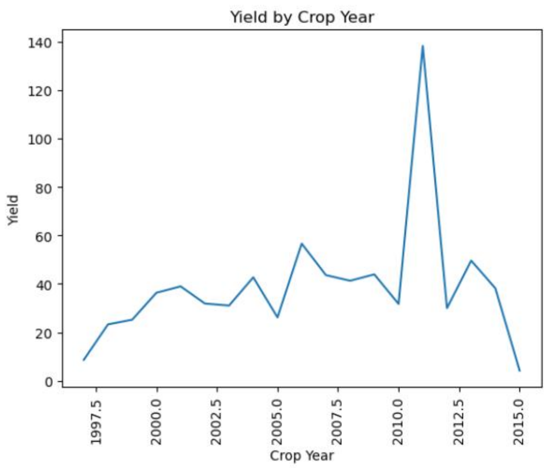
[24]:
```
sns.lineplot(x="Season", y="Yield", data=df)
plt.xticks(rotation=90)
plt.title('Yield by Season')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.show()
```



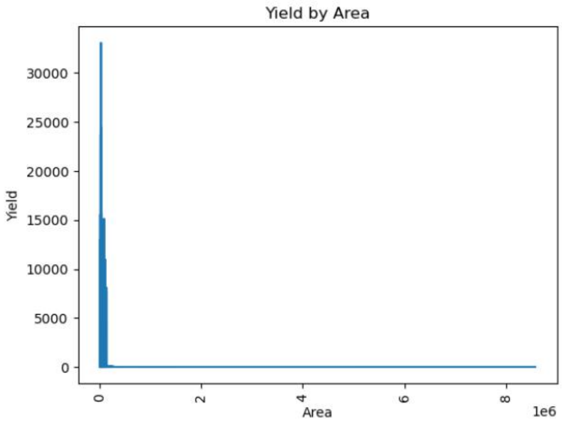
[25]:
```
sns.lineplot(x="Crop_Year", y="Production", data=df, errorbar=None)
plt.xticks(rotation=90)
plt.title('Production by Crop Year')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



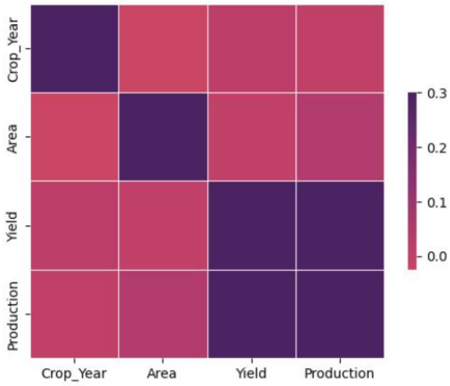
[26]:
```
sns.lineplot(x="Crop_Year", y="Yield", data=df, errorbar=None)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[27]:
```
sns.lineplot(x="Area", y="Yield", data=df, errorbar=None)
plt.xticks(rotation=90)
plt.title('Yield by Area')
plt.xlabel('Area')
plt.ylabel('Yield')
plt.show()
```



[28]:
```
cmap = sns.color_palette("flare", as_cmap=True)
sns.heatmap(df[['Crop_Year', 'Area', 'Yield', 'Production']].corr(),
            cmap=cmap,
            vmax=.3,
            center=0,
            square=True,
            linewidths=.5,
            cbar_kws={"shrink": .5})

plt.show()
```



Now analyzing Each crop (Rice, Wheat, Potato, Cotton, Maize)

1) Rice

[29]:
```
df_rice_data = df[df["Crop"]=="Rice"]
df_rice_data.head()
```

[29]:

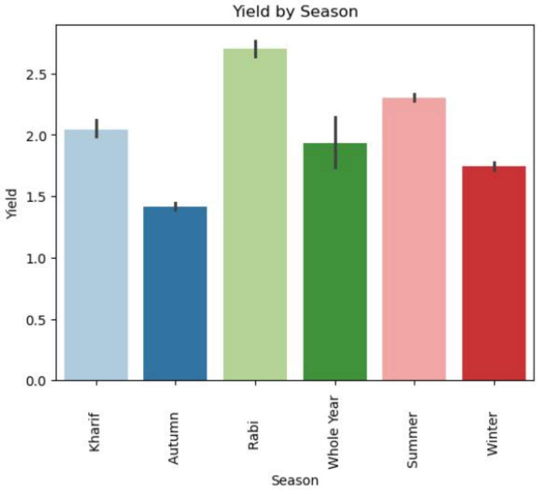
	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
2	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Rice	102.00	321.00	3.147059	Cereals
12	Andaman and Nicobar Islands	NICOBARS	2001	Kharif	Rice	83.00	300.00	3.614458	Cereals
18	Andaman and Nicobar Islands	NICOBARS	2002	Kharif	Rice	189.20	510.84	2.700000	Cereals
27	Andaman and Nicobar Islands	NICOBARS	2003	Kharif	Rice	52.00	90.17	1.734038	Cereals
36	Andaman and Nicobar Islands	NICOBARS	2004	Kharif	Rice	52.94	72.57	1.370797	Cereals

[30]:
```
df_rice_data.shape
```

[30]: (15082, 9)

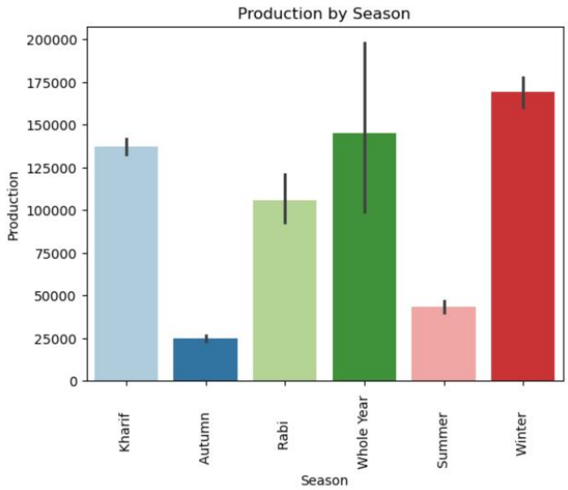
[33]:

```
sns.barplot(x="Season", y="Yield", data=df_rice_data,palette="Paired", hue="Season", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by Season')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.show()
```

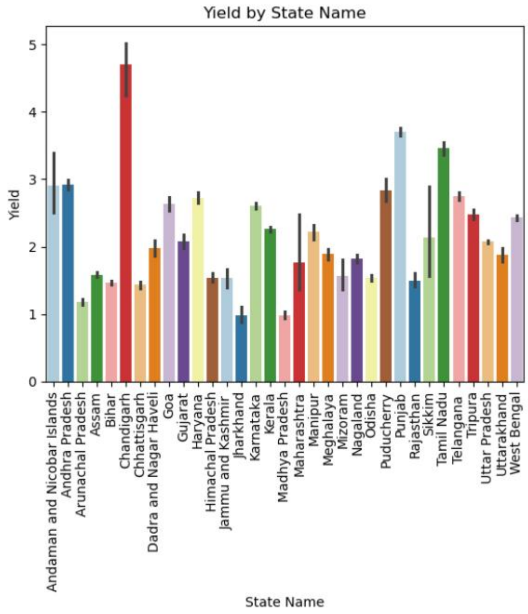


[34]:

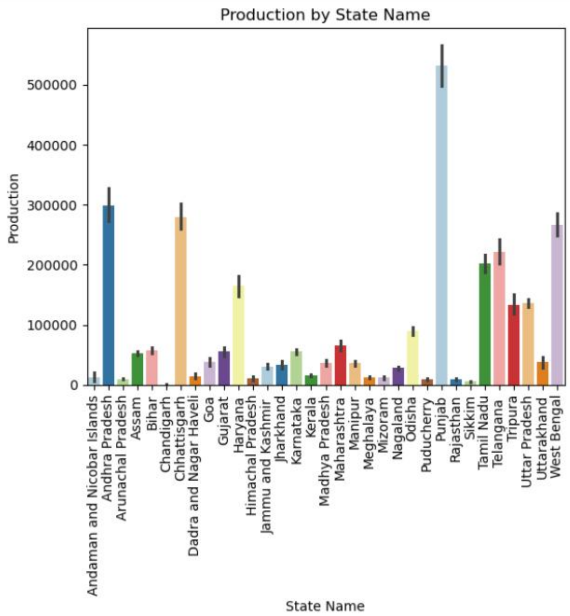
```
sns.barplot(x="Season", y="Production", data=df_rice_data,palette="Paired", hue="Season", legend= False)
plt.xticks(rotation=90)
plt.title('Production by Season')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```



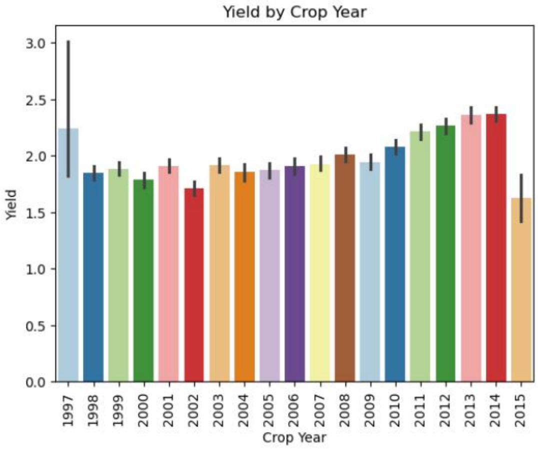
[35]:
```
sns.barplot(x="State_Name", y="Yield", data=df_rice_data,palette="Paired", hue="State_Name", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by State Name')
plt.xlabel('State Name')
plt.ylabel('Yield')
plt.show()
```



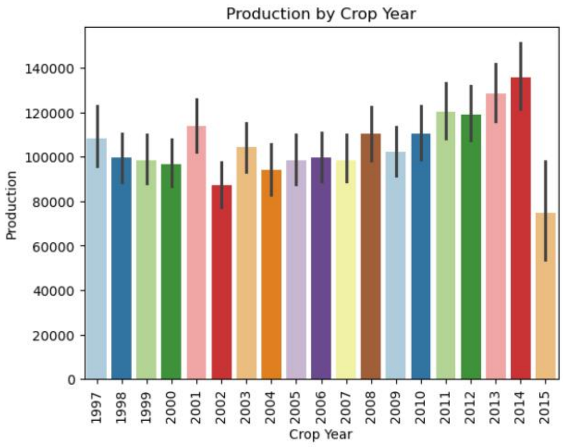
[36]:
```
sns.barplot(x="State_Name", y="Production", data=df_rice_data,palette="Paired", hue="State_Name", legend= False)
plt.xticks(rotation=90)
plt.title('Production by State Name')
plt.xlabel('State Name')
plt.ylabel('Production')
plt.show()
```



[37]:
```
sns.barplot(x="Crop_Year", y="Yield", data=df_rice_data,palette="Paired", hue="Crop_Year", legend= False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[38]:
```
sns.barplot(x="Crop_Year", y="Production", data=df_rice_data,palette="Paired", hue="Crop_Year", legend= False)
plt.xticks(rotation=90)
plt.title('Production by Crop Year')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



2) Wheat

[39]:
```
df_wheat_data = df[df["Crop"] == "Wheat"]
df_wheat_data.head()
```

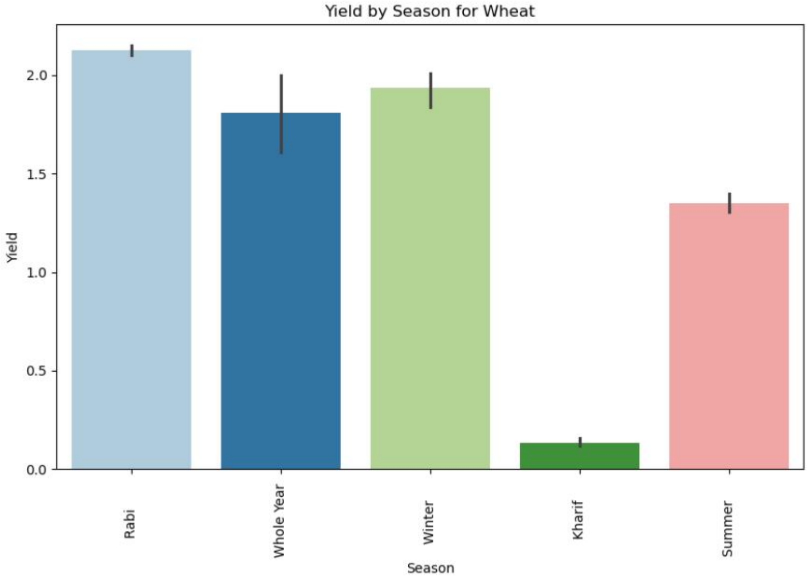
[39]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
228	Andhra Pradesh	ANANTAPUR	1997	Rabi	Wheat	300.0	200.0	0.666667	Cereals
253	Andhra Pradesh	ANANTAPUR	1998	Rabi	Wheat	400.0	200.0	0.500000	Cereals
282	Andhra Pradesh	ANANTAPUR	1999	Rabi	Wheat	439.0	294.0	0.669704	Cereals
324	Andhra Pradesh	ANANTAPUR	2000	Rabi	Wheat	520.0	297.0	0.571154	Cereals
370	Andhra Pradesh	ANANTAPUR	2001	Rabi	Wheat	307.0	213.0	0.693811	Cereals

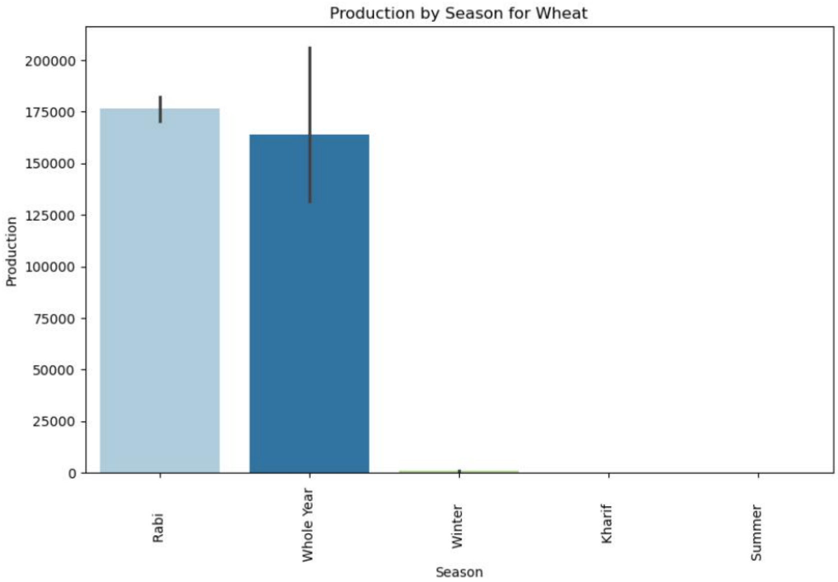
[40]:
```
df_wheat_data.shape
```

[40]: (7878, 9)

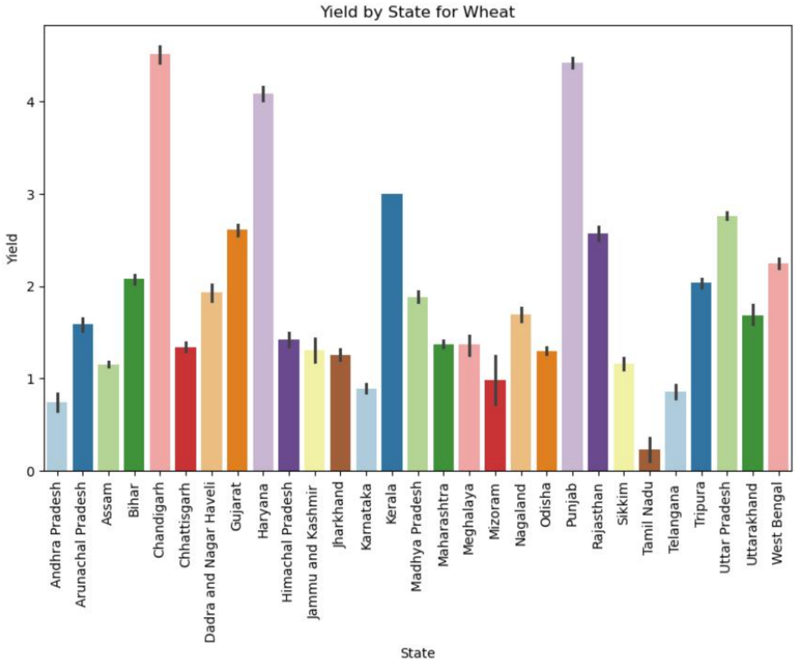
[41]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Yield", data=df_wheat_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Season for Wheat')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.show()
```



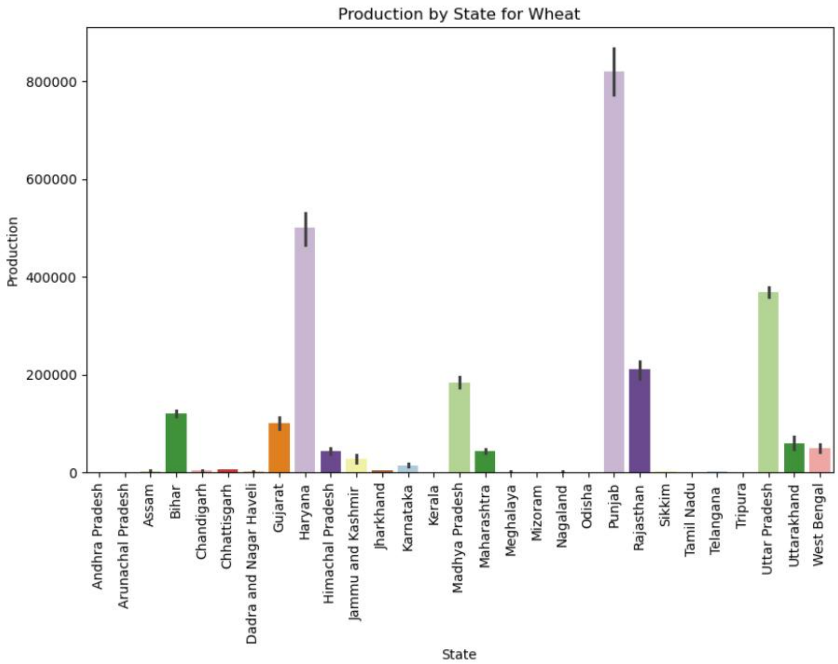
[42]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Production", data=df_wheat_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Season for Wheat')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```



[43]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Yield", data=df_wheat_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by State for Wheat')
plt.xlabel('State')
plt.ylabel('Yield')
plt.show()
```

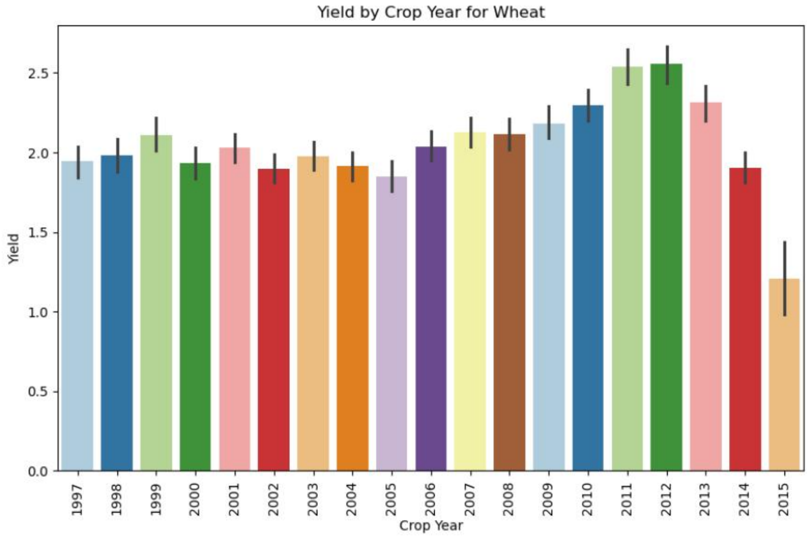


[44]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Production", data=df_wheat_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Production by State for Wheat')
plt.xlabel('State')
plt.ylabel('Production')
plt.show()
```



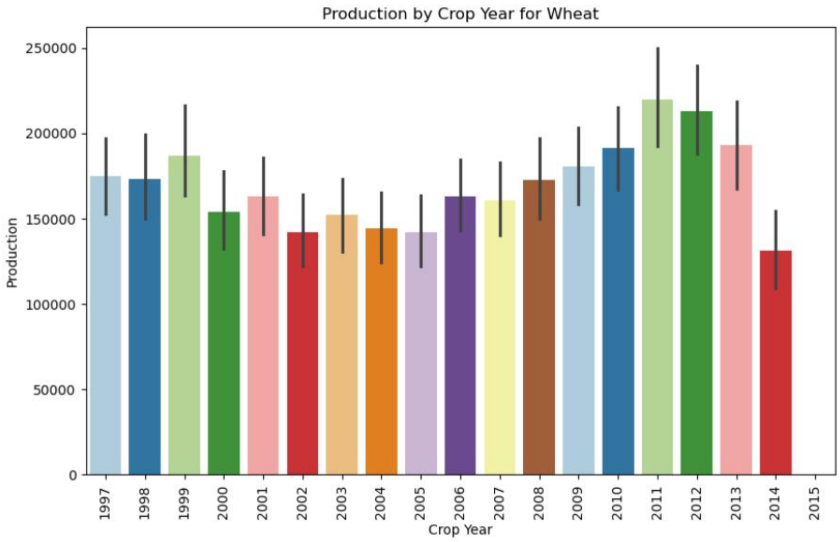
[45]:

```
plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Yield", data=df_wheat_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year for Wheat')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[46]:

```
plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Production", data=df_wheat_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Crop Year for Wheat')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



3) Potato

[47]:
```
df_potato_data = df[df["Crop"] == "Potato"]
df_potato_data.head()
```

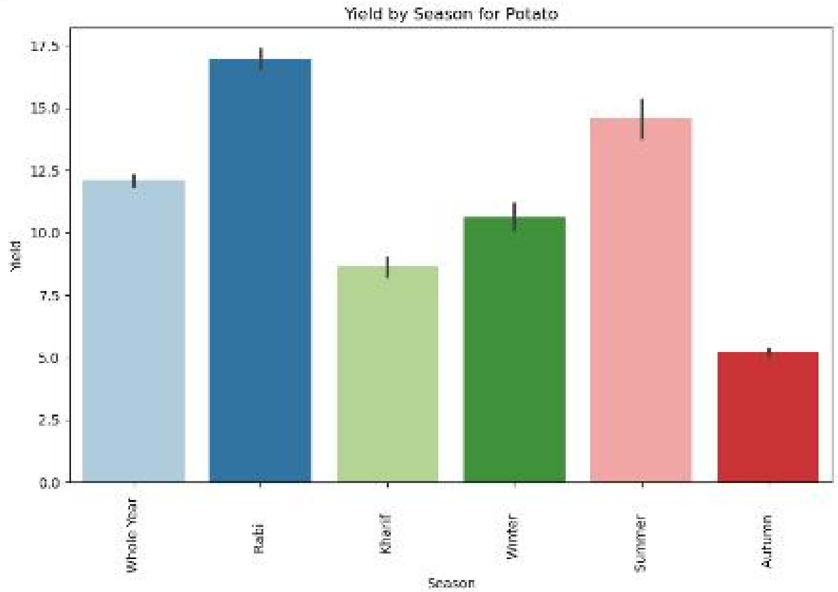
[47]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
329	Andhra Pradesh	ANANTAPUR	2000	Whole Year	Potato	4.0	34.0	8.500000	Vegetables
431	Andhra Pradesh	ANANTAPUR	2002	Whole Year	Potato	2.0	17.0	8.500000	Vegetables
528	Andhra Pradesh	ANANTAPUR	2004	Whole Year	Potato	2.0	20.0	10.000000	Vegetables
739	Andhra Pradesh	ANANTAPUR	2010	Whole Year	Potato	21.0	236.0	11.238095	Vegetables
786	Andhra Pradesh	ANANTAPUR	2011	Whole Year	Potato	18.0	181.0	10.055556	Vegetables

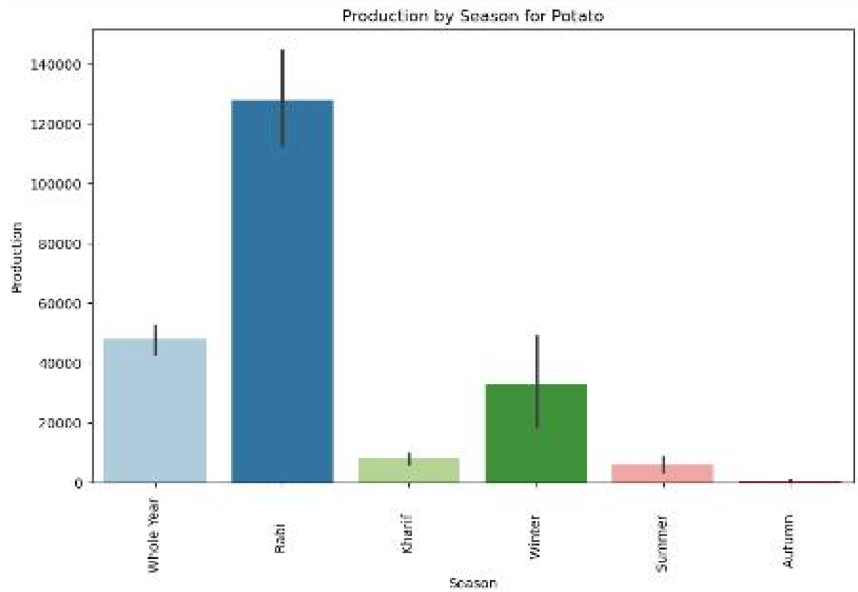
[48]:
```
df_potato_data.shape
```

[48]: (6914, 9)

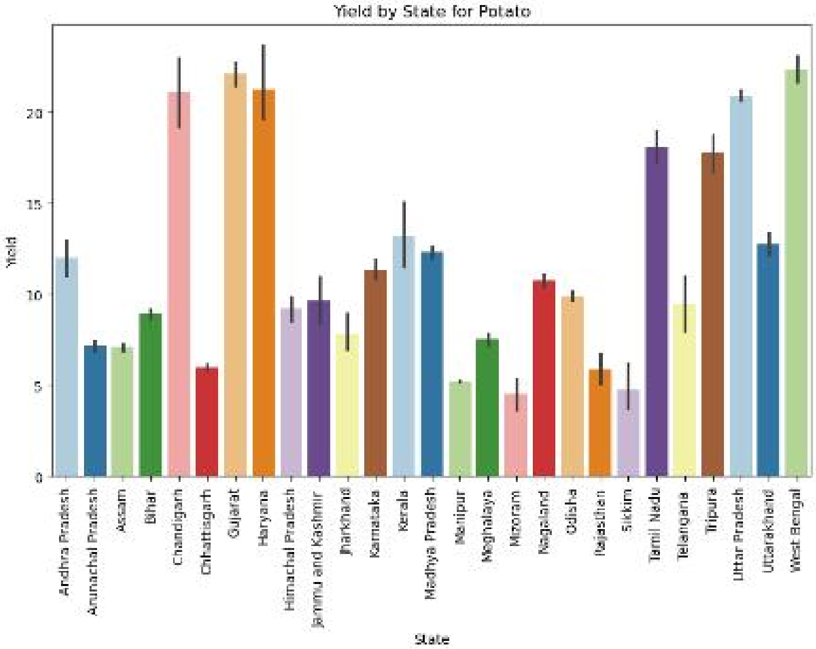
[49]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Yield", data=df_potato_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Season for Potato')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.show()
```



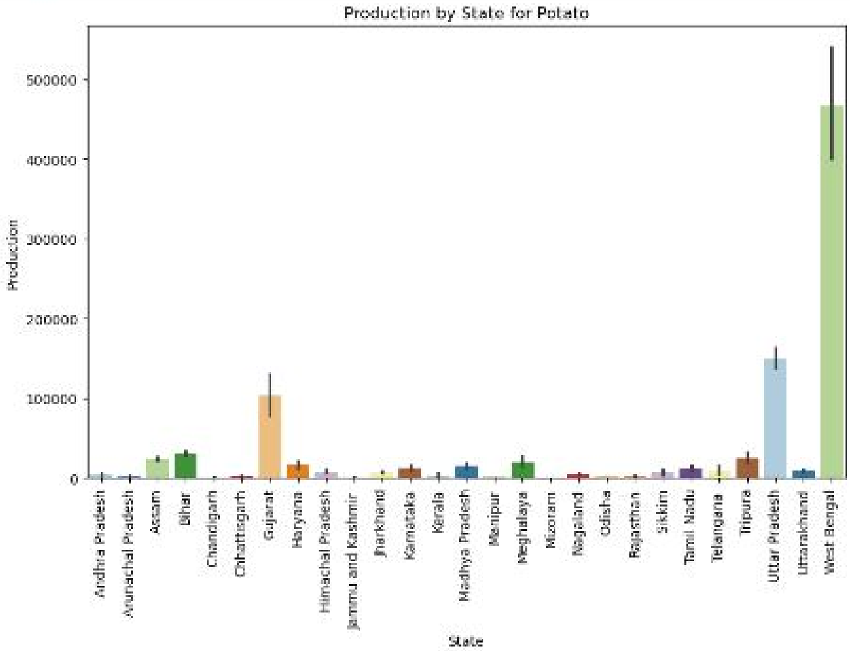
[50]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Production", data=df_potato_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Season for Potato')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```



[51]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Yield", data=df_potato_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by State for Potato')
plt.xlabel('State')
plt.ylabel('Yield')
plt.show()
```

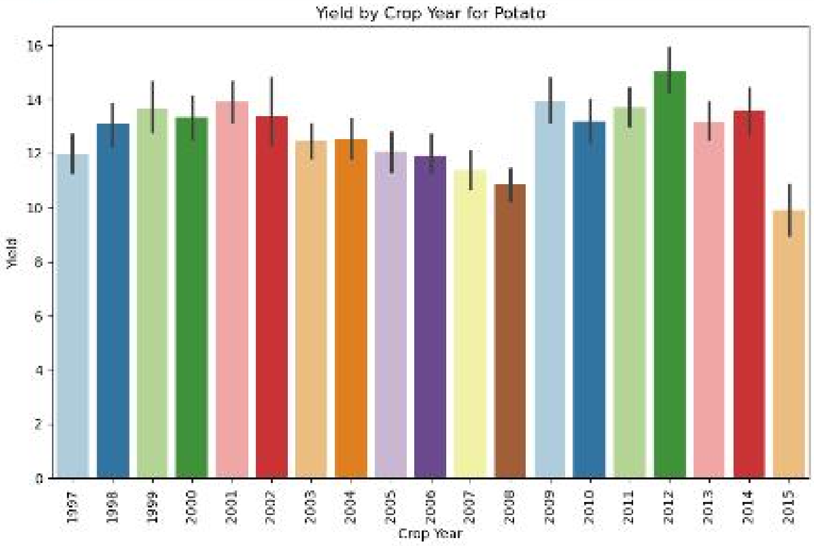


[52]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Production", data=df_potato_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Production by State for Potato')
plt.xlabel('State')
plt.ylabel('Production')
plt.show()
```



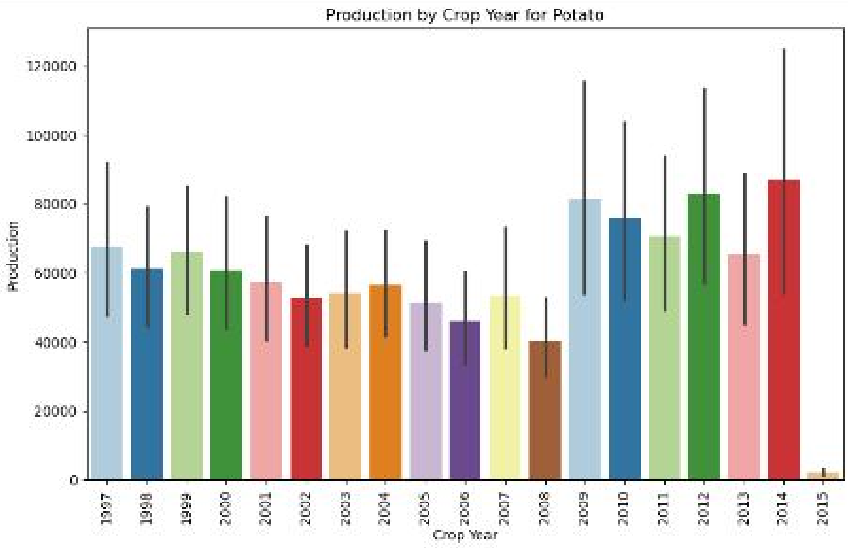
[53]:

```python
# Yield by Crop Year
plt.figure(figsize=(18, 6))
sns.barplot(x="Crop_Year", y="Yield", data=df_potato_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year for Potato')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[54]:

```python
plt.figure(figsize=(18, 6))
sns.barplot(x="Crop_Year", y="Production", data=df_potato_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Crop Year for Potato')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



4) Coconut

```
[55]: df_coconut_data = df[df["Crop"] == "Coconut"]
      df_coconut_data
```

[55]:

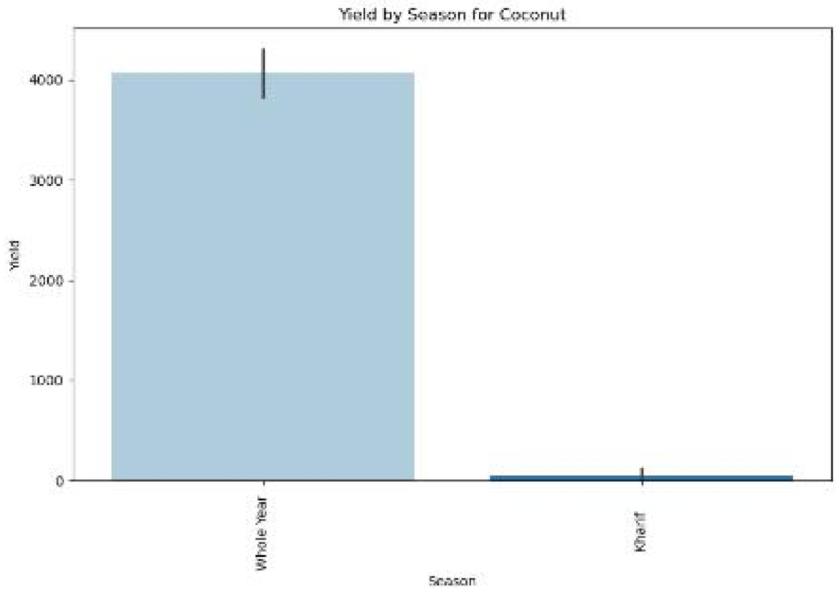
	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
5	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Coconut	18168.00	65100000.0	3583.223250	Nuts
14	Andaman and Nicobar Islands	NICOBARS	2001	Whole Year	Coconut	18190.00	64430000.0	3542.056075	Nuts
23	Andaman and Nicobar Islands	NICOBARS	2002	Whole Year	Coconut	18240.00	67490000.0	3700.109649	Nuts
32	Andaman and Nicobar Islands	NICOBARS	2003	Whole Year	Coconut	18284.74	68580000.0	3750.668590	Nuts
41	Andaman and Nicobar Islands	NICOBARS	2004	Whole Year	Coconut	18394.70	52380000.0	2847.559351	Nuts
...	...	...	...	...	...	...	...	...	...
241990	West Bengal	PURULIA	2004	Whole Year	Coconut	66.00	296.1	4.486364	Nuts
242027	West Bengal	PURULIA	2005	Whole Year	Coconut	74.00	311.0	4.202703	Nuts
242063	West Bengal	PURULIA	2006	Whole Year	Coconut	73.00	365000.0	5000.000000	Nuts
242108	West Bengal	PURULIA	2007	Whole Year	Coconut	58.00	898000.0	15482.758621	Nuts
242149	West Bengal	PURULIA	2008	Whole Year	Coconut	58.00	598.0	10.310345	Nuts

1958 rows × 9 columns

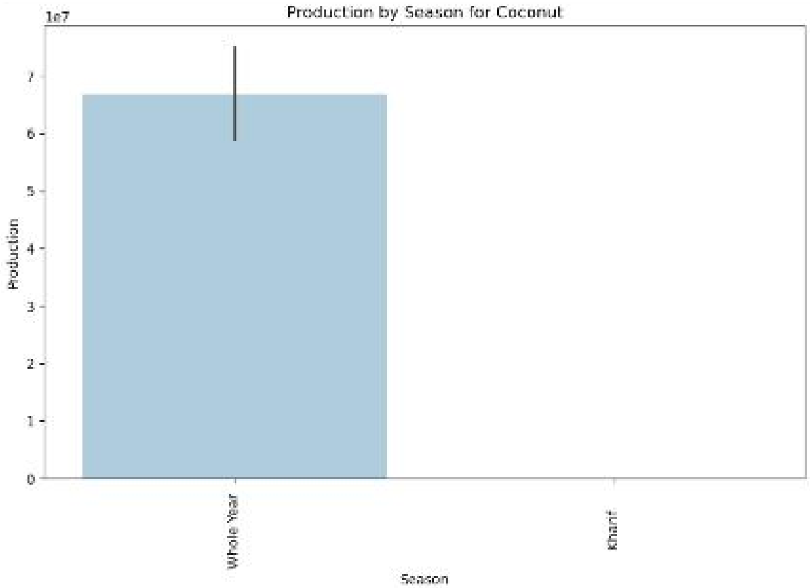
```
[56]: df_coconut_data.shape
```

```
[56]: (1958, 9)
```

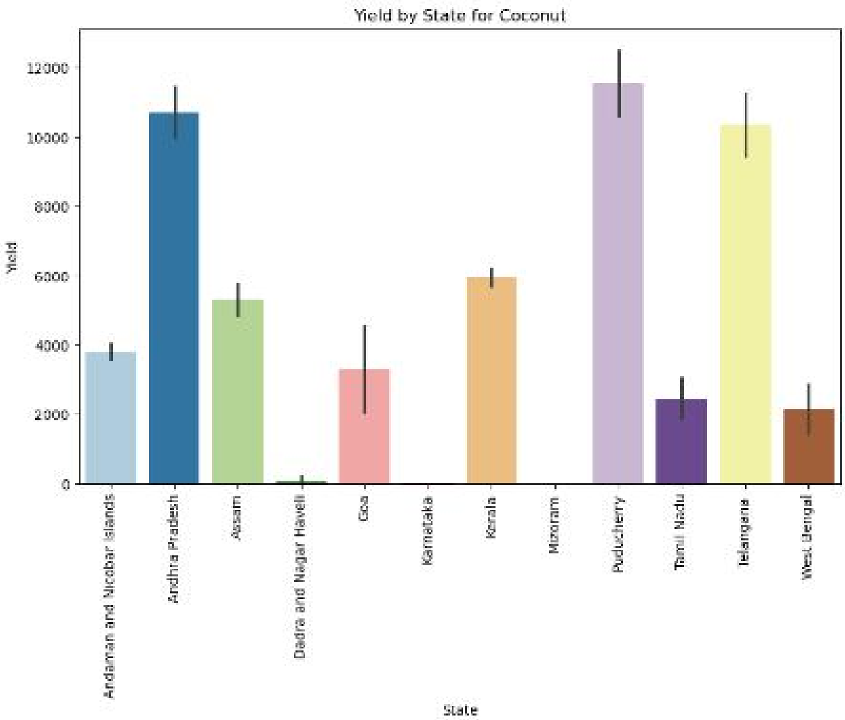
```
[57]: plt.figure(figsize=(10, 6))
      sns.barplot(x="Season", y="Yield", data=df_coconut_data, palette="Paired", hue="Season", legend=False)
      plt.xticks(rotation=90)
      plt.title('Yield by Season for Coconut')
      plt.xlabel('Season')
      plt.ylabel('Yield')
      plt.show()
```



[59]:
```python
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Production", data=df_coconut_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Season for Coconut')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```

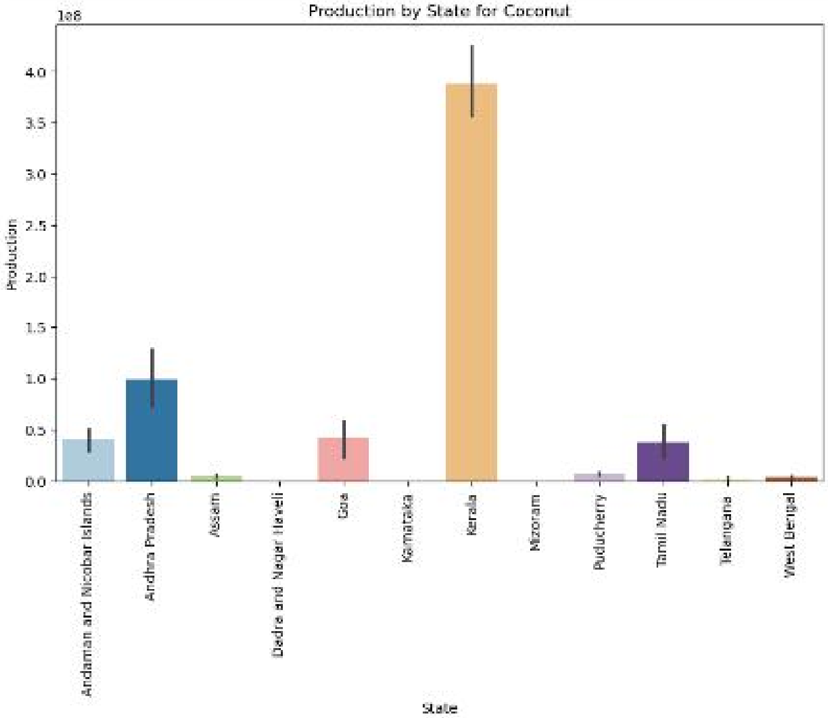


[60]:
```python
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Yield", data=df_coconut_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by State for Coconut')
plt.xlabel('State')
plt.ylabel('Yield')
plt.show()
```



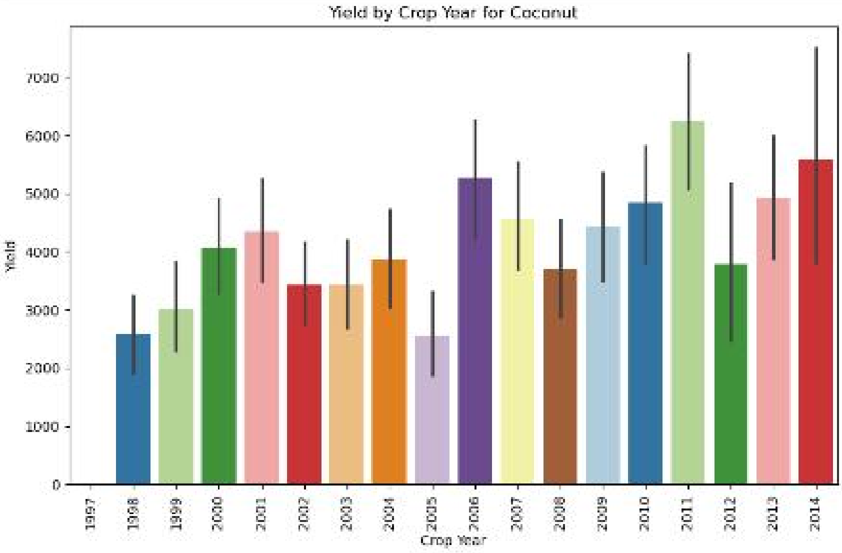
[62]:

```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Production", data=df_coconut_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Production by State for Coconut')
plt.xlabel('State')
plt.ylabel('Production')
plt.show()
```

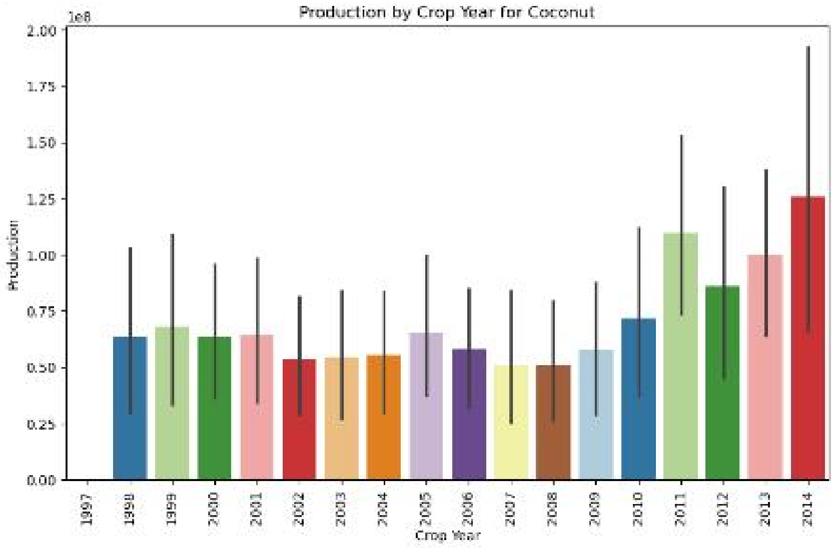


[63]:

```
plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Yield", data=df_coconut_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year for Coconut')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



[65]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Production", data=df_coconut_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Crop Year for Coconut')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



5) Maize

[66]:
```
df_maize_data = df[df["Crop"] == "Maize"]
df_maize_data.head()
```

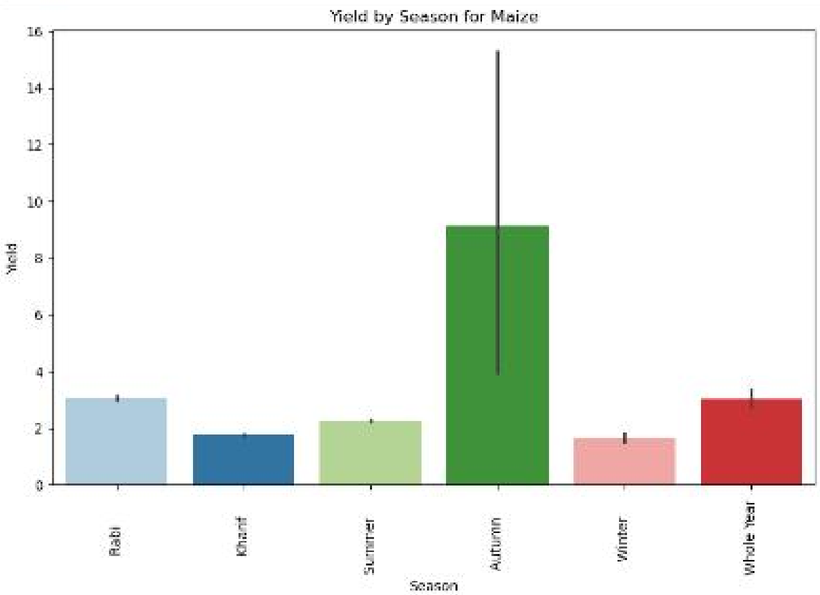
[66]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield	Crop_Category
69	Andaman and Nicobar Islands	NICOBARS	2010	Rabi	Maize	3.84	18.22	4.744792	Cereals
118	Andaman and Nicobar Islands	NORTH AND MIDDLE ANDAMAN	2010	Rabi	Maize	86.70	96.40	1.111880	Cereals
192	Andaman and Nicobar Islands	SOUTH ANDAMANS	2010	Rabi	Maize	73.00	253.00	3.465753	Cereals
210	Andhra Pradesh	ANANTAPUR	1997	Kharif	Maize	2800.00	4900.00	1.750000	Cereals
224	Andhra Pradesh	ANANTAPUR	1997	Rabi	Maize	600.00	2400.00	4.000000	Cereals

[67]:
```
df_maize_data.shape
```

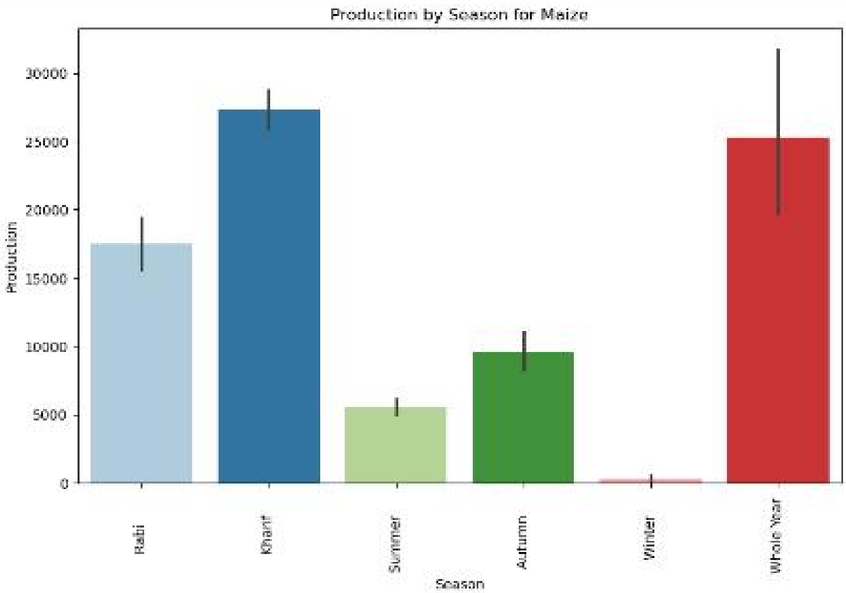
[67]: (13787, 9)

[68]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Season", y="Yield", data=df_maize_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Season for Maize')
plt.xlabel('Season')
plt.ylabel('Yield')
plt.show()
```



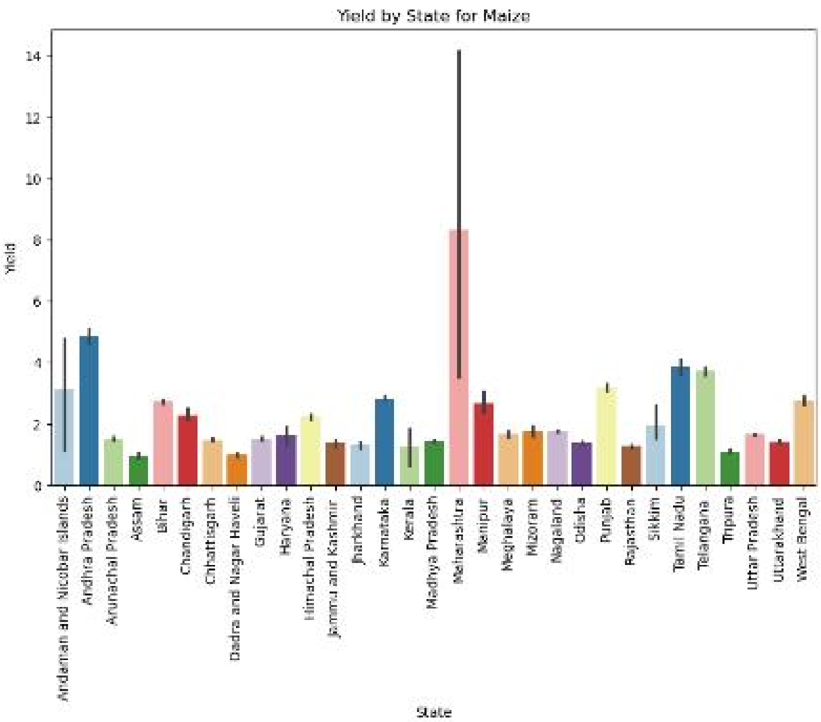
[71]:

```
plt.figure(figsize=(18, 6))
sns.barplot(x="Season", y="Production", data=df_maize_data, palette="Paired", hue="Season", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Season for Maize')
plt.xlabel('Season')
plt.ylabel('Production')
plt.show()
```

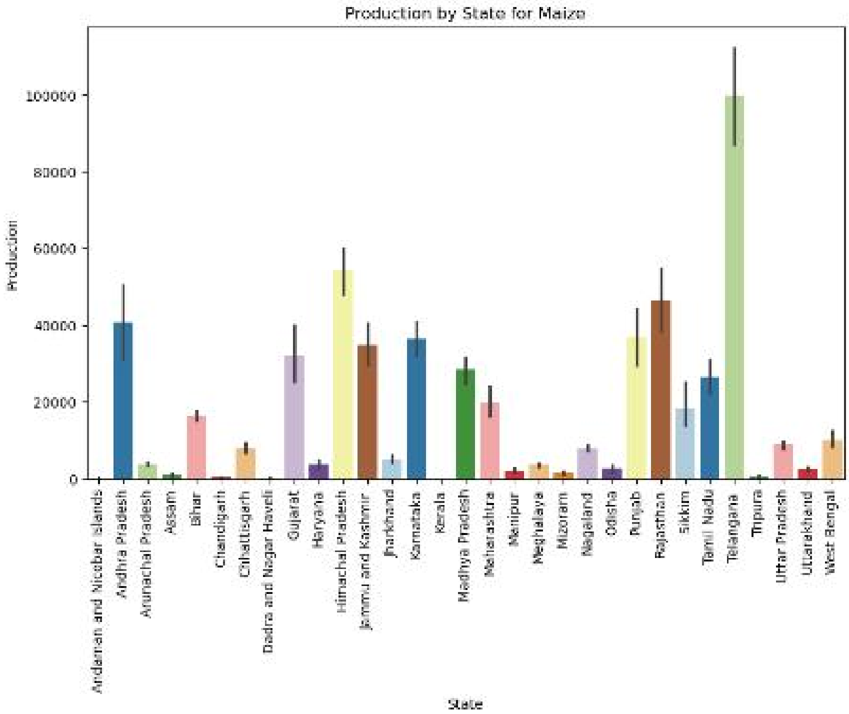


[72]:

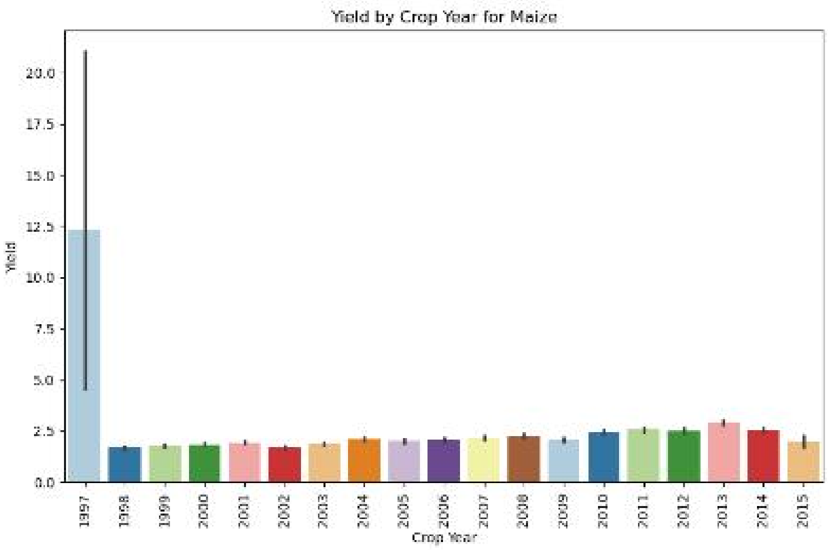
```
plt.figure(figsize=(18, 6))
sns.barplot(x="State_Name", y="Yield", data=df_maize_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by State for Maize')
plt.xlabel('State')
plt.ylabel('Yield')
plt.show()
```



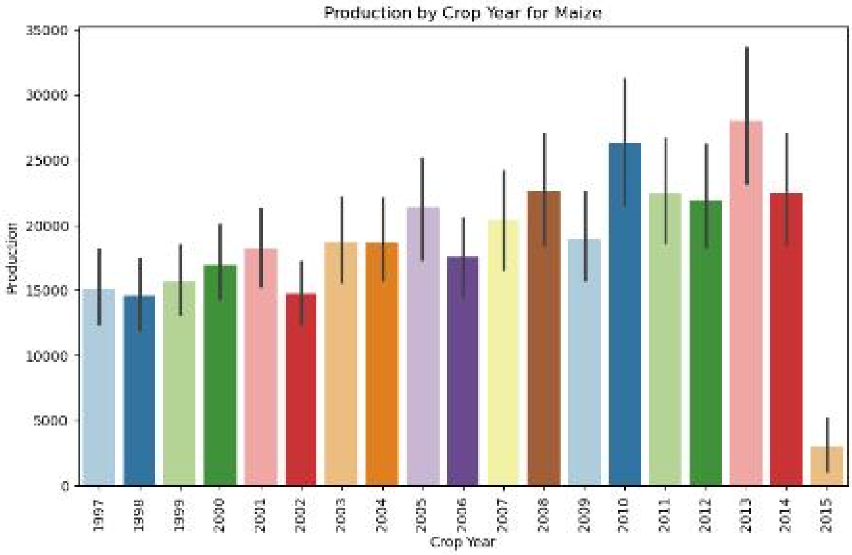
[73]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="State_Name", y="Production", data=df_maize_data, palette="Paired", hue="State_Name", legend=False)
plt.xticks(rotation=90)
plt.title('Production by State for Maize')
plt.xlabel('State')
plt.ylabel('Production')
plt.show()
```



[74]:
```
plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Yield", data=df_maize_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Yield by Crop Year for Maize')
plt.xlabel('Crop Year')
plt.ylabel('Yield')
plt.show()
```



```
[75]: plt.figure(figsize=(10, 6))
sns.barplot(x="Crop_Year", y="Production", data=df_maize_data, palette="Paired", hue="Crop_Year", legend=False)
plt.xticks(rotation=90)
plt.title('Production by Crop Year for Maize')
plt.xlabel('Crop Year')
plt.ylabel('Production')
plt.show()
```



3) Model Building:

Implemented the XGBoost algorithm to predict crop production. After splitting the data into training and testing sets, I trained the model using optimal hyperparameters with both L1 (Lasso) and L2 (Ridge) regularization to prevent overfitting. The model's performance was evaluated using metrics like R^2 and Mean Absolute Error (MAE) on both training and test datasets.

Model.py:

```
# -*- coding: utf-8 -*-
"""
Created on Tue Sep  3 23:18:44 2024

@author: ranea
"""

from sklearn.model_selection import train_test_split
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
from sklearn.metrics import r2_score, mean_absolute_error
from sklearn.model_selection import learning_curve


df = pd.read_csv('Cleaned_Crop_Production.csv')

data = df.drop(['State_Name'], axis=1)
dummy = pd.get_dummies(data)

x = dummy.drop(["Production", "Yield"], axis=1)
```

```
y = dummy["Production"]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=5)

print("x_train :", x_train.shape)
print("x_test :", x_test.shape)
print("y_train :", y_train.shape)
print("y_test :", y_test.shape)

xgb_model = xgb.XGBRegressor(
    reg_alpha=10,
    reg_lambda=0.1,
    n_estimators=100,
    objective='reg:squarederror',
    eval_metric='mae'
)

xgb_model.fit(x_train, y_train)

y_train_pred = xgb_model.predict(x_train)
y_test_pred = xgb_model.predict(x_test)

train_r2 = r2_score(y_train, y_train_pred)
train_mae = mean_absolute_error(y_train, y_train_pred)
test_r2 = r2_score(y_test, y_test_pred)
test_mae = mean_absolute_error(y_test, y_test_pred)

print("XGBoost Training R²:", train_r2)
print("XGBoost Training MAE:", train_mae)
print("XGBoost Test R²:", test_r2)
print("XGBoost Test MAE:", test_mae)

plt.figure(figsize=(10,6))
plt.scatter(y_test, y_test_pred, color='blue', alpha=0.6)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--', lw=2)  # 45-
degree line
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('XGBoost: Test Set Actual vs Predicted Values')
plt.grid(True)
plt.show()

train_sizes, train_scores, test_scores = learning_curve(
    xgb_model, x, y, train_sizes=[100, 500, 1000, 5000, 10000], cv=5,
scoring='neg_mean_absolute_error'
)
train_errors_mean = -train_scores.mean(axis=1)
test_errors_mean = -test_scores.mean(axis=1)

plt.figure(figsize=(10,6))
plt.plot(train_sizes, train_errors_mean, label='Training Error')
plt.plot(train_sizes, test_errors_mean, label='Test Error')
plt.xlabel('Training Set Size')
plt.ylabel('Mean Absolute Error')
plt.title('Learning Curves')
plt.legend()
plt.grid(True)
plt.show()
```

Model Result:

```
In [1]: runfile('C:/Users/ranea/CropProduction/Model.py', wdir='C:/Users/ranea/CropProduction')
x_train : (181770, 788)
x_test : (60591, 788)
y_train : (181770,)
y_test : (60591,)
XGBoost Training R²: 0.9965063498188226
XGBoost Training MAE: 56986.05024930907
XGBoost Test R²: 0.9599334528080093
XGBoost Test MAE: 141306.73901528507
```