PROJECT ON CROP PRODUCTION

NAME : DEVAGUPTAPU BHANU SRI

EMAIL ID : bhanudevaguptapu4@gmail.com

COURSE : DATA ANALYSIS

PROJECT : CROP PRODUCTION ANALYSIS USING PYTHON

IMPORTING DIFFERENT PAKAGES AND DATASET INTO KERNEL.

SOURCE CODE:

```
#1STEP.First know the problem statements.

#2STEP.Collect the data from the respective sources.

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import random

import seaborn as sns
```

LOADING DATASET:

```
#First load the csv file into the kernal

csv_file_path = 'Crop Production data intern.csv'

dataset = pd.read_csv(csv_file_path)

print(dataset)
```

INFORMATION REGARDING THE DATASET:

```
#Find the type of the data in dataset

type(dataset)

dataset.info()

#Shape of data

dataset.shape

#To find the dimensions of the data set

#Find the bottom columns and rows in dataset

dataset.head(10)

#Find the bottom columns and rows in dataset

dataset.tail(10)
```

TO CLEAN THE NULL VALUES AND REPLACING THEM WITH APPROPRIATE MEAN VALUES:

```
#3STEP.Clean and Prepare the data

###To find null values in the dataset

f=dataset.isna().sum(axis=0)

f
```

#There are no null values in any column except Production column with 3730 null values

#As the production column values of the dataset is integer then and also not a categorical values so we can replace the null values as mean of the column

dataset['Production'].fillna(dataset['Production'].mean(),inplace = True) #inplace= True is used to change the values directly in the dataframe itself

dataset.isna().sum(axis=0)

#So,there are no null values left in the dataset

ANALYSING THE UNIQUE VALUES OF ALL COLUMNS IN DATASET:

#4STEP.Analyzing of the given dataset.

######################

#We can find the count of unique values of dataset

dataset.nunique()

#So, we can find the different unique values of the columns in the dataset.

#Here we can find there are 33 different states with 646 different districts and 6 different seasons at 19 different years.

#To find the different States in the given dataset

dataset['State_Name'].unique()

#To find different District Names of the States

f=dataset['District_Name'].unique()

print(f)

print(len(f))

#To find the different years of crop production

f=dataset['Crop_Year'].unique()

f.sort()

print(f)

#So,the crop production is done from 1997 to 2015 in the given dataset

#To find the different seasons in which the crop production is done

dataset['Season'].unique()

#To find different types of crop that are being produced in the 19 years of time period

b=dataset['Crop'].unique()

print(b)

print(len(b))

DESCRIBING THE GIVEN DATASET:

#To know some statistical details regarding the dataset

dataset.describe()

#As there are only 3 columns in numerical data type . So, we know their mean,min,max standard deviation etc

OUTPUT:

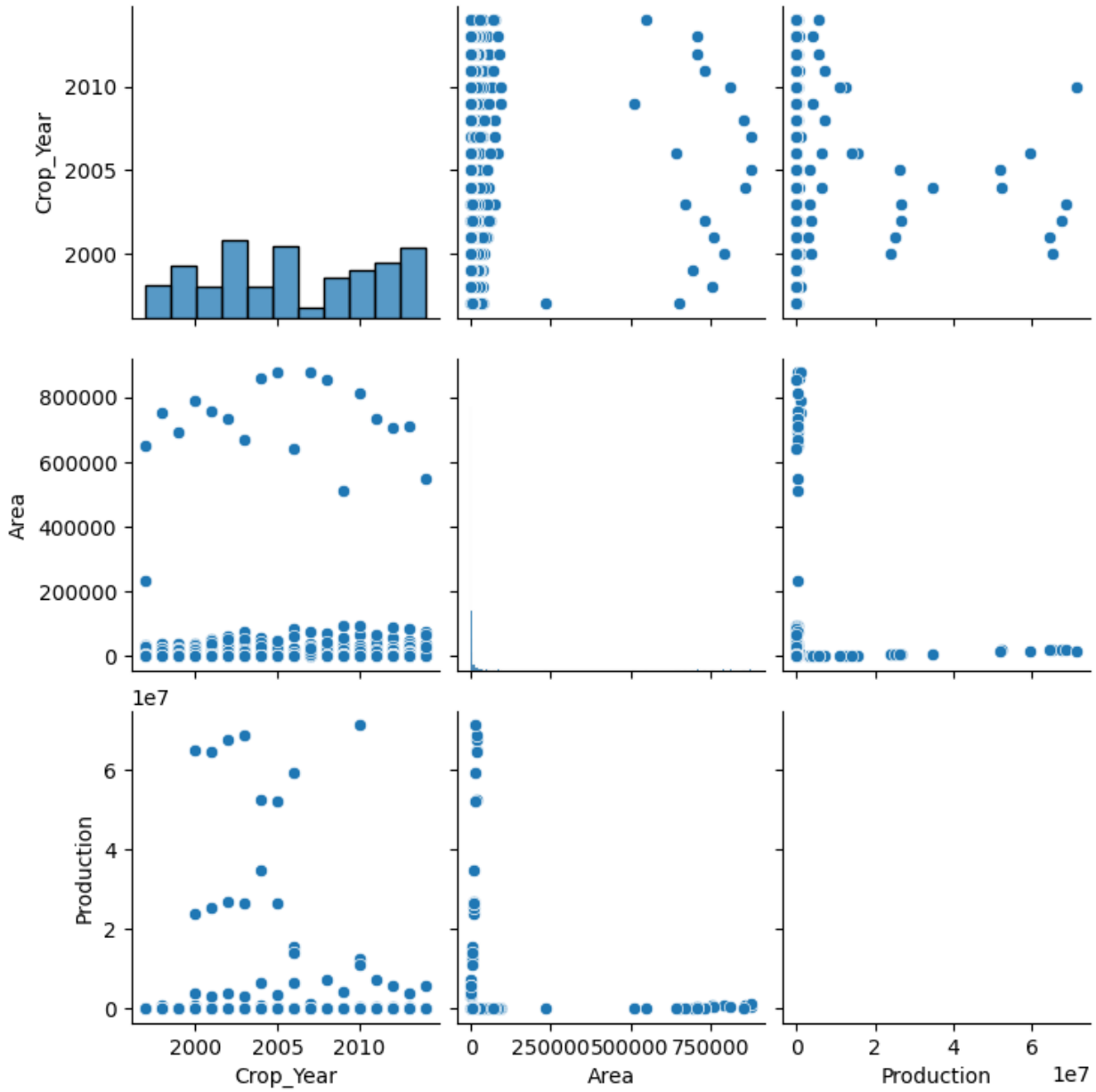
	Crop_Year	Area	Production
count	246091.000000	2.460910e+05	2.460910e+05
mean	2005.643018	1.200282e+04	5.825034e+05
std	4.952164	5.052340e+04	1.693599e+07
min	1997.000000	4.000000e-02	0.000000e+00
25%	2002.000000	8.000000e+01	9.100000e+01
50%	2006.000000	5.820000e+02	7.880000e+02
75%	2010.000000	4.392000e+03	8.000000e+03
max	2015.000000	8.580100e+06	1.250800e+09

PAIR PLOT:

```
import pandas as pd

import seaborn as sns

import matplotlib.pyplot as plt

from ipywidgets import interactive

df = dataset.head(1000)

# To know the correlation and distribution between the variables

def size_widget(height=2.5, aspect=1):

  sns.pairplot(df, height=height, aspect=aspect)

interactive(size_widget, height=(1, 3.5, 0.5), aspect=(0.5, 2, 0.25))
```

#This pair plot shows us the relational distribution between the different columns of the dataset.

Another pairplot using seaborn library

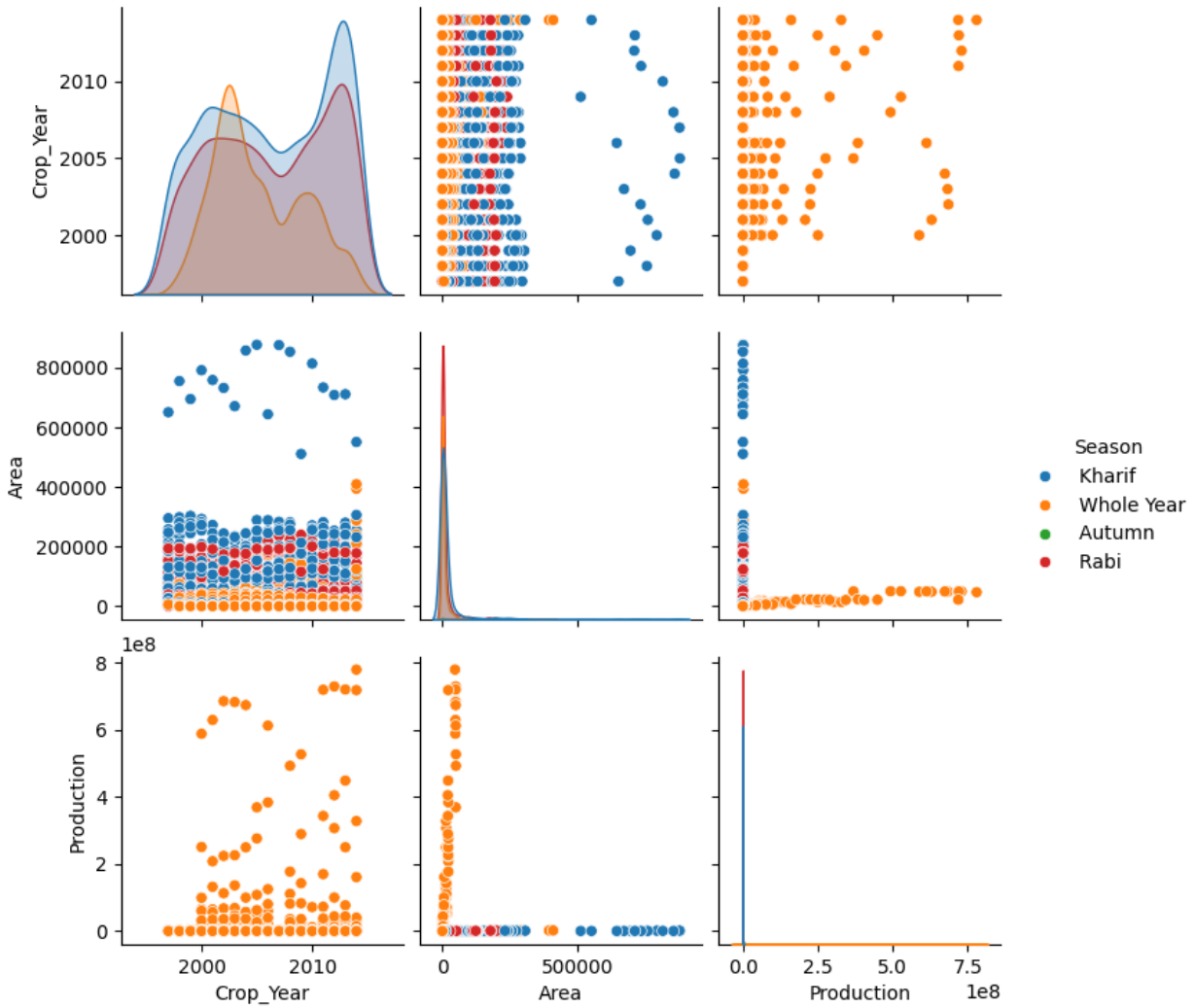


df=dataset.head(10000)

print(df)

To know the correlation and distribution between the variables

sns.pairplot(df,hue = 'Season')



FINDING OUTLIERS IN THE DATA:

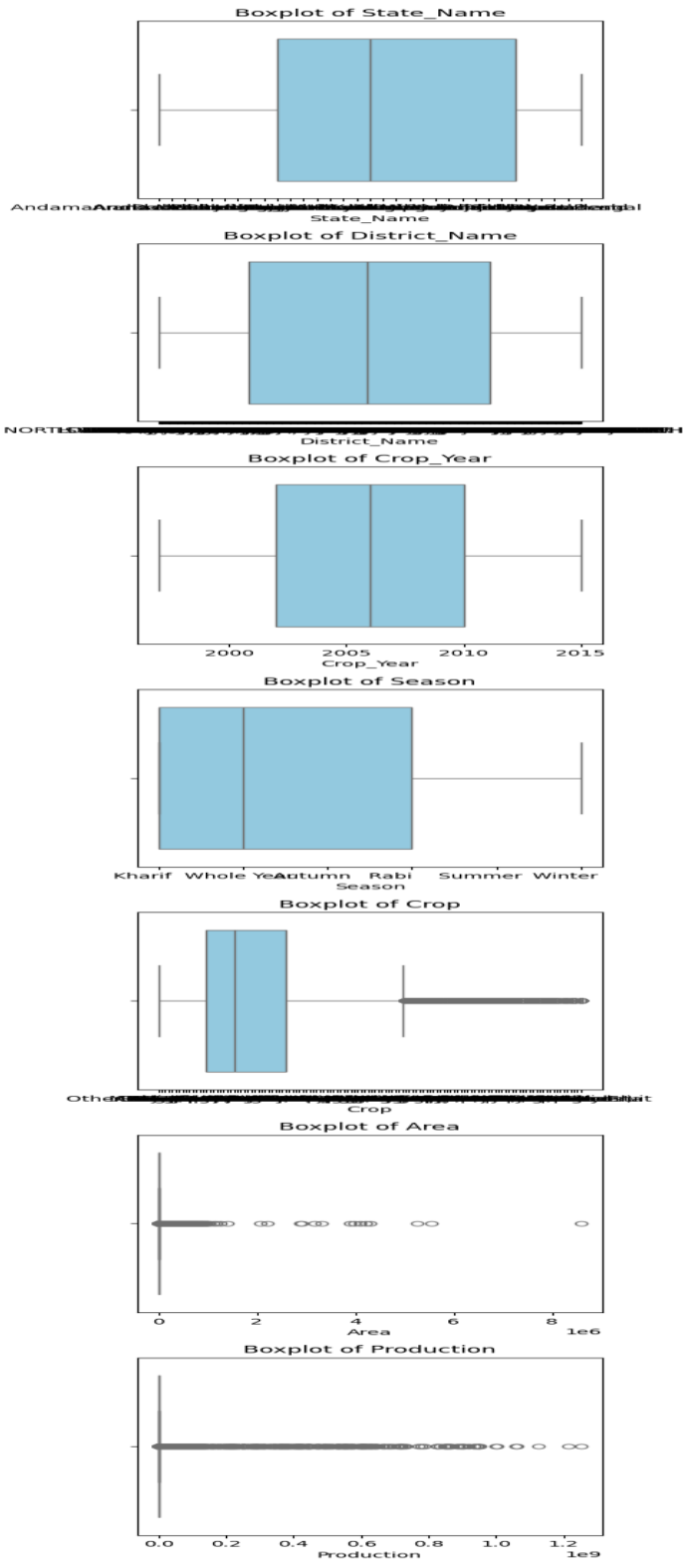
#To show the outliers in graph

plt.figure(figsize=(5, 4 * len(dataset.columns)))

Iterate through each column and plot boxplot

for i, column in enumerate(dataset.columns):

  plt.subplot(len(dataset.columns), 1, i+1)



sns.boxplot(x=dataset[column], color='skyblue')

plt.title(f'Boxplot of {column}')

plt.tight_layout()

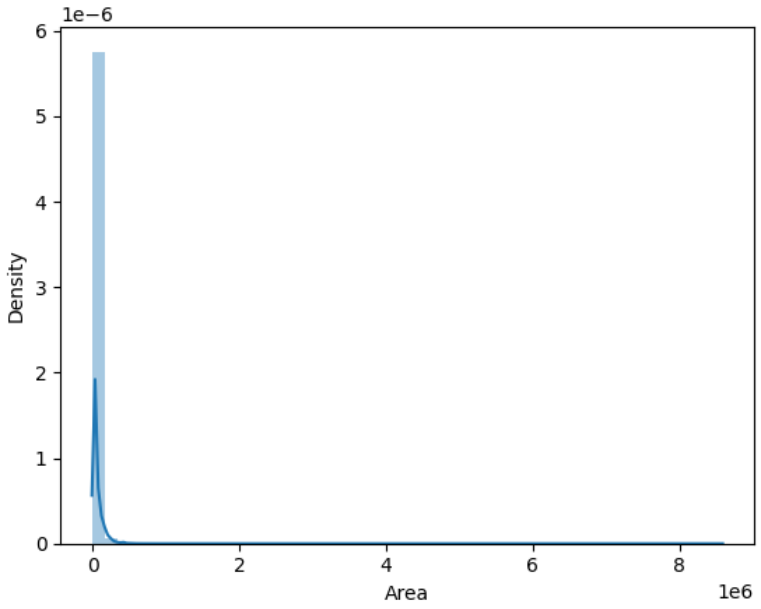
plt.show()

#DISTRIBUTION OF AREA COLUMN:

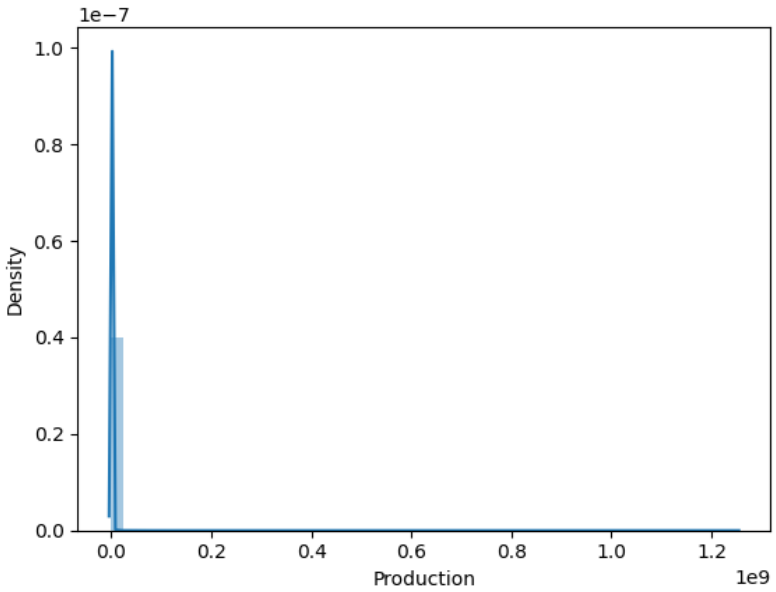
#Skewness of the AREA column

sns.distplot(dataset['Area'])

#It is positively skewed



DISTRIBUTION OF PRODUCTION:



SORTING THE CROP YEARS AND CREATING A NEW DATASET:

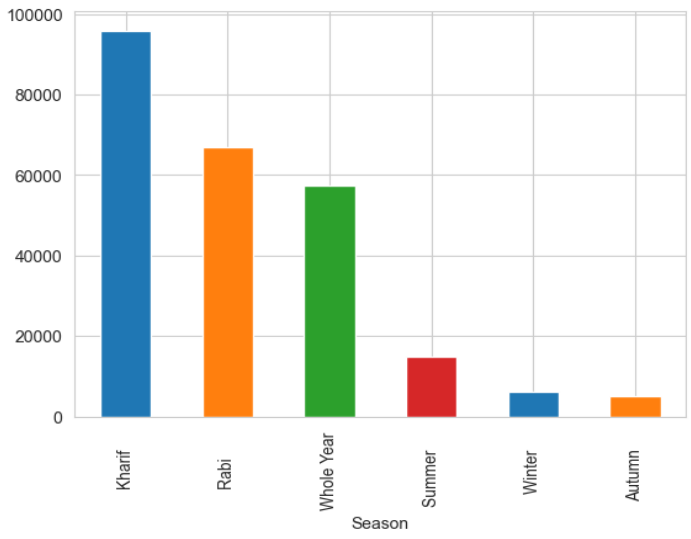
```
#Changing the column according the sorting values of Crop_Year and reseting of index.

new_df = dataset.sort_values(by='Crop_Year')

new_df.reset_index(drop=True,inplace = True)

print(new_df)
```

SEPERATING THE STATE NAMES WITH THEIR COLUMN NAMES:

```
# Group the DataFrame by 'state' column and count the number of unique districts in each group

# Define a custom function to get both district names and count

def get_district_info(group):

    district_names = ', '.join(sorted(group['District_Name'].unique()))

    district_count = group['District_Name'].nunique()

    return pd.Series({'Districts': district_names, 'District_Count': district_count})

## Group the DataFrame by 'state' column and apply the custom function

district_info = new_df.groupby('State_Name').apply(get_district_info)


# Print the district names and count for each state

print(district_info)
```

#To know which season has highest -production

```
colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728']

new_df['Season'].value_counts().plot.bar(color=colors)

#By this graph we can say that Kharif seson has highest number of crops grown.
```



#To know the kind of crops grown in different sesons

```
#pd.set_option('display.max_rows', None)

#pd.set_option('display.max_columns', None)

cross_table = pd.crosstab(new_df.Crop,new_df.Season)

print(cross_table)

#From this we can understand that different types of crops are grown in different sesons in which '0' reperesents its absence of production.
```

Group the dataset by Season and Crop_Year and sum the production for each combination

```
season_year_production = new_df.groupby(['Season', 'Crop_Year'])['Production'].sum()


# Find the season and year with the highest production

season_year_with_highest_production = season_year_production.idxmax()

highest_production_value = season_year_production.max()


# Print the result

print(f"The combination of season and year with the highest production is {season_year_with_highest_production} with a total production of {highest_production_value}")
```

#To know the maximim production in different seasons

```
max_production_index = new_df.groupby('Season')['Production'].idxmax()

max_production_index
```

Define a custom function to get the corresponding year for maximum production

```
def get_max_year(group):

    max_index = group['Production'].idxmax()

    max_year = group.loc[max_index, 'Crop_Year']

    return max_year
```

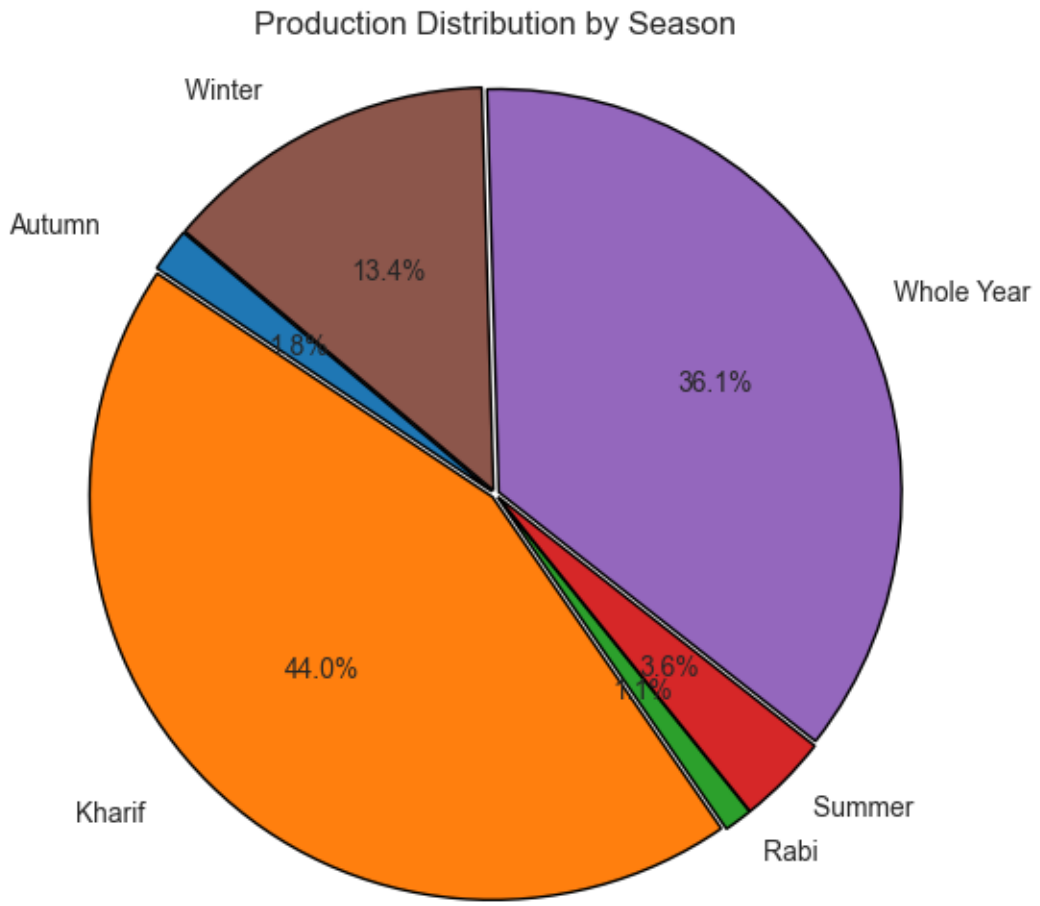
Group by Season and apply the custom function to get the corresponding year for maximum production

```
max_years = new_df.groupby('Season').apply(get_max_year)

# Reset index to align with DataFrame format
```

```
max_years = max_years.reset_index(name='Max_Crop_Year')

# Print the result

print(max_years)
```

#pie plot showing the percentage of production done in every season

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

plt.pie(max_info['Production'], labels=max_info['Season'], autopct='%1.1f%%', startangle=140,explode=[0.01,0.01 , 0.01,0.01,0.01, 0.01], wedgeprops = {"edgecolor" : "black"})

plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle

plt.title('Production Distribution by Season')

plt.show()
```



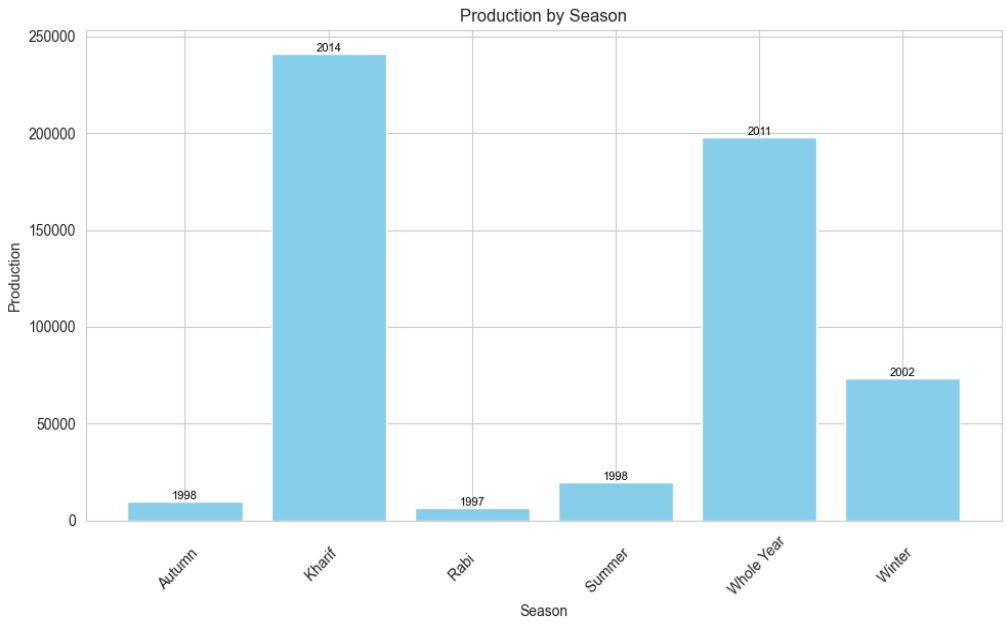
Merge max_years and max_production_index DataFrames on 'Season'

```
max_info = pd.merge(max_years, max_production_index, on='Season')

# Print the result

print(max_info)

##################################
```

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

bars = plt.bar(max_info['Season'], max_info['Production'], color='skyblue')


# Add years on top of each bar

for bar, year in zip(bars, max_info['Max_Crop_Year']):

    height = bar.get_height()

    plt.text(bar.get_x() + bar.get_width() / 2, height, year,

             ha='center', va='bottom', color='black', fontsize=8)


plt.xlabel('Season')

plt.ylabel('Production')

plt.title('Production by Season')

plt.xticks(rotation=45)  # Rotate x-axis labels for better readability

plt.tight_layout()

plt.show()
```

OUTPUT:

	Season	Max_Crop_Year	Production
0	Autumn	1998	9641
1	Kharif	2014	240975
2	Rabi	1997	6253
3	Summer	1998	19955
4	Whole Year	2011	197646
5	Winter	2002	73569



Add a new column 'State_Name' to max_info DataFrame based on state codes

```
# Define a custom function to get the corresponding year for maximum production
def get_max_state(group):
    max_index = group['Production'].idxmax()
    max_state = group.loc[max_index, 'State_Name']
    return max_state

# Group by Season and apply the custom function to get the corresponding year for maximum production
max_state = new_df.groupby('Season').apply(get_max_state)

# Reset index to align with DataFrame format
max_state = max_state.reset_index(name='Max_State_Names')

# Print the result
print(max_state)
```

Merge max_years with max_State_name based on the 'Season' column

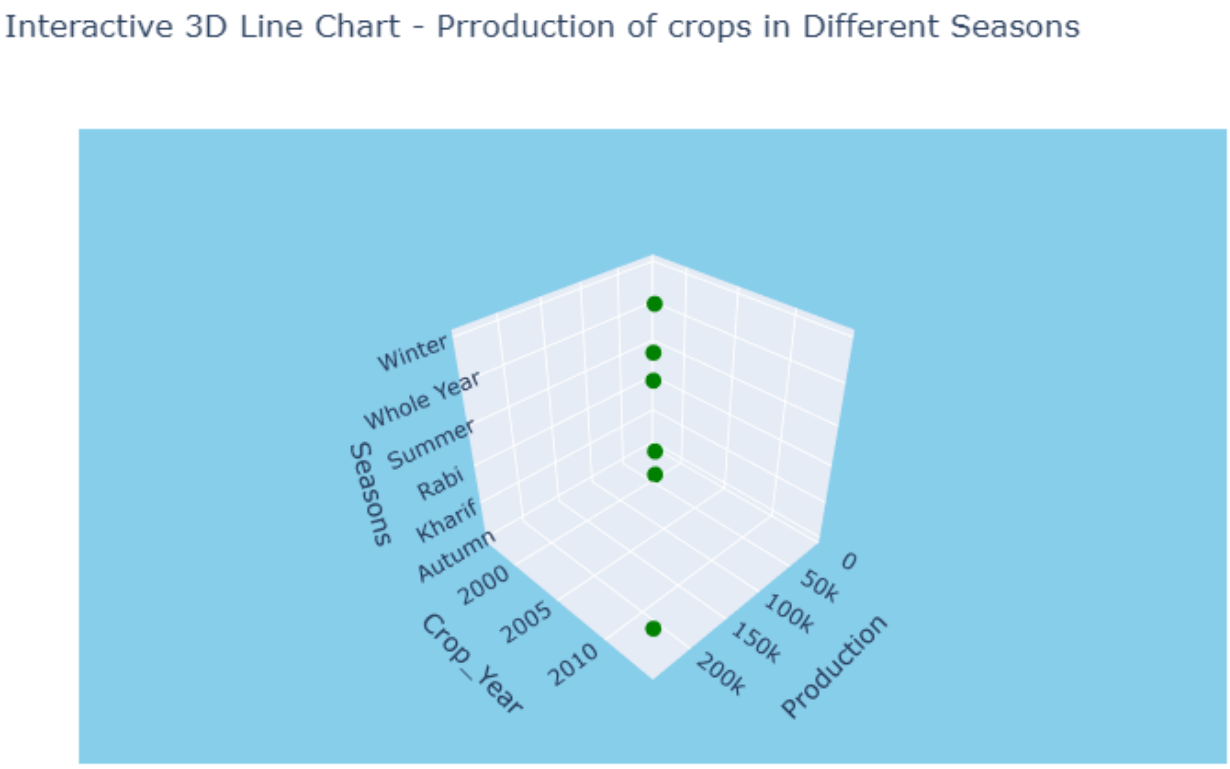
```
merged_info = pd.merge(max_years, max_state, on='Season', how='inner')
# Merge the result with max_production_index based on the 'Season' column
merged_info = pd.merge(merged_info, max_production_index, on='Season', how='inner')
print(merged_info)
```

#3D graph

```
import plotly.graph_objects as go
import pandas as pd
#colors='green'
# Creating Interactive 3D Line Chart
fig = go.Figure(data=[go.Scatter3d(x=merged_info['Production'], y=merged_info['Max_Crop_Year'],
z=merged_info['Season'], mode='markers',marker=dict(color='green',size=5))])
# Adding Title and Labels
fig.update_layout(title='Interactive 3D Line Chart - Prroduction of crops in Different Seasons',
scene=dict(xaxis_title='Production', yaxis_title='Crop_Year', zaxis_title='Seasons',bgcolor='skyblue'))
#The scene attribute allows you to specify various settings for the 3D scene, including axis titles, camera settings,
background color, and more.
```

Displaying the Plot

fig.show()




```
def get_max_district(group):
    max_index = group['Production'].idxmax()
    max_district = group.loc[max_index, 'District_Name']
    return max_district

# Group by Season and apply the custom function to get the corresponding year for maximum production
max_district = new_df.groupby('Season').apply(get_max_district)

# Reset index to align with DataFrame format
max_district = max_district.reset_index(name='District_Names')
# Print the result # District_Name
print(max_district)
```

```
merged_df = pd.merge(merged_info[['Season', 'Max_Crop_Year', 'Max_State_Names', 'Production']],
                     max_district[['Season', 'District_Names']],
                     on='Season',
                     how='inner')

# Display the merged dataframe
print(merged_df)
```

##From the above we can observe that different seasons have high production in differnt crop_yearsand thir corresponding states and district names with the about of productions.

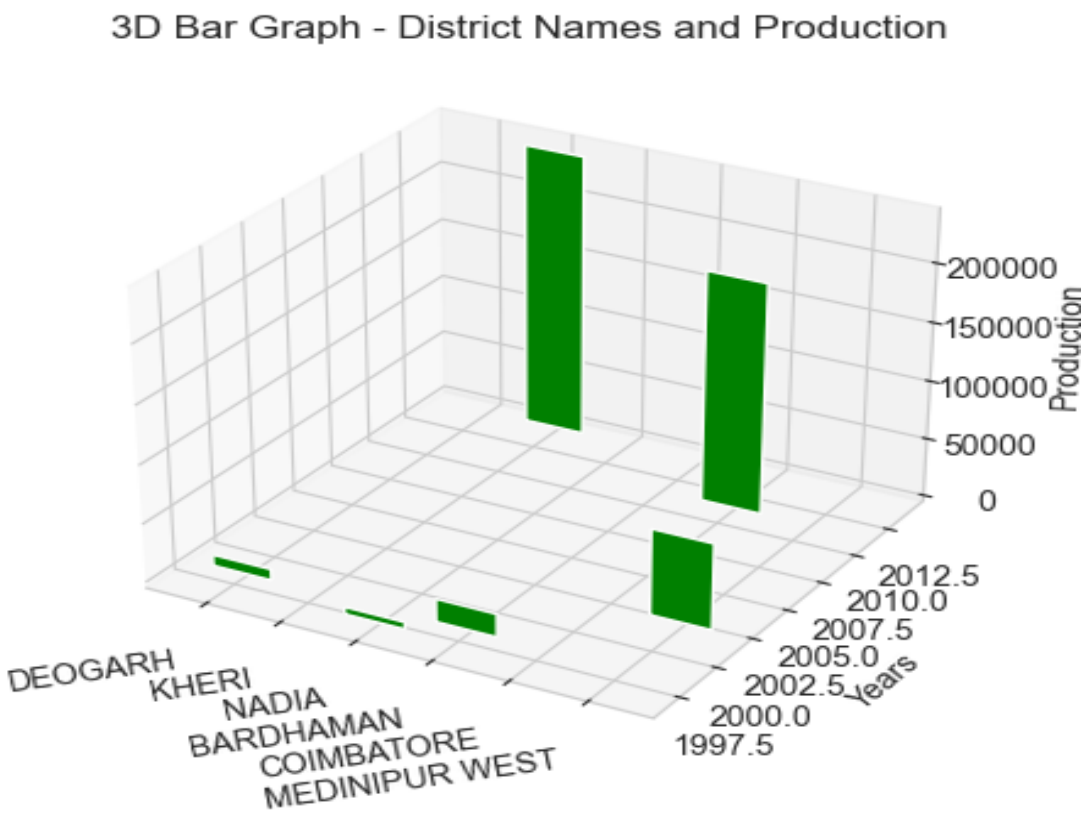
OUTPUT:

```
Season  Max_Crop_Year Max_State_Names  Production  District_Names
0  Autumn                1998          Odisha        9641         DEOGARH
1  Kharif                2014   Uttar Pradesh      240975           KHERI
2  Rabi                  1997     West Bengal        6253           NADIA
3  Summer                1998     West Bengal       19955       BARDHAMAN
4  Whole Year            2011      Tamil Nadu      197646      COIMBATORE
5  Winter                2002     West Bengal       73569  MEDINIPUR WEST
```

#3D GRAPH:

```
import matplotlib.pyplot as plt
import pandas as pd
from mpl_toolkits.mplot3d import Axes3D

# Plotting 3D Bar Graph
fig = plt.figure()
ax = fig.add_subplot(1,1,1, projection='3d')
ax.bar(merged_df['District_Names'], merged_df['Production'], zs=merged_df[ 'Max_Crop_Year'], zdir='y', width=0.8, color='green')

# Adding Title and Labels
ax.set_title('3D Bar Graph - District Names and Production')
ax.set_xlabel('')
ax.set_ylabel('Years')
ax.set_zlabel('Production')

# Rotating x-axis labels for better readability
```

```
ax.set_xticks(merged_df['District_Names'])

ax.set_xticklabels(merged_df['District_Names'], rotation=10, ha='right')


# Displaying the Plot

plt.show()
```



#CROPS THAT ARE GROWN IN ALL THE SEASONS:

```
crops_grown_in_all_seasons = cross_table.index[cross_table.all(axis=1)]


# Print the filtered crops

print("Crops that grow in all seasons:")

for crop in crops_grown_in_all_seasons:

    print(crop)
```

OUTPUT:

```
Crops that grow in all seasons:
Arhar/Tur
Banana
Cotton(lint)
Dry chillies
Dry ginger
```

```
Groundnut
Maize
Moong(Green Gram)
Onion
Peas & beans (Pulses)
Potato
Ragi
Rice
Sesamum
Sugarcane
Turmeric
Urad
```

```
for index, row in cross_table.iterrows():
   # Print the crop name
   print(f"Crop: {index}")
   print("Seasons:")
 # Iterate over seasons and their counts for the current crop
   for season, count in row.items():
      if count > 0:
         print(f"{season}: {count}")
 print()
```

Remove extra spaces from column names

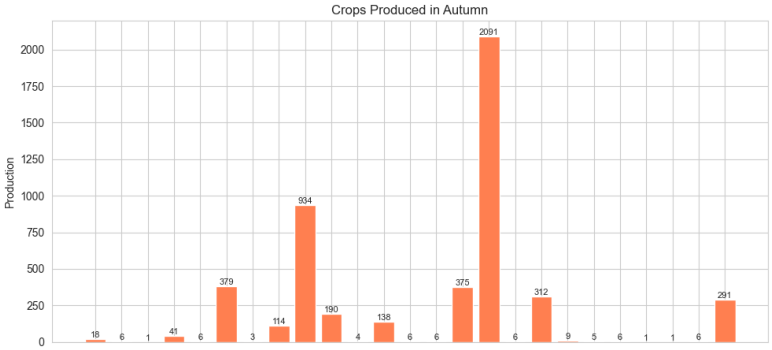
```
cross_table.columns = cross_table.columns.str.strip()
# Get crops grown in autumn season
autumn_crops = cross_table.index[cross_table['Autumn'] > 0]
# Print the crops grown in autumn season
print("Crops grown in autumn season:")
print(autumn_crops)
max_production_crop = cross_table.idxmax(axis=0)
# Print the result
print("Maximum produced crop for each season:")
print(max_production_crop)
#So we can see that Rice is the crop which has highest production that covered most of the seasons like Autumn,Summer, and Winter
#Maize in Kharif season,
#Wheat in Rabi season
```

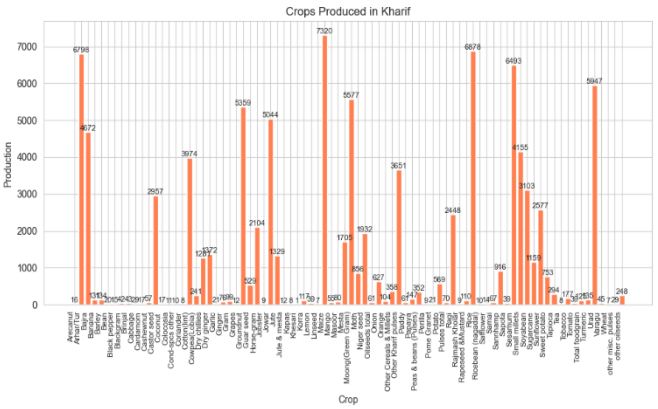
#Sugercane has highest production in WholeYear

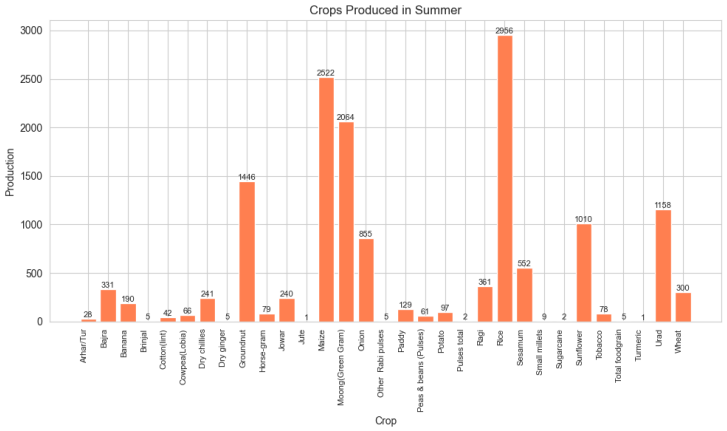
Find the crop with minimum production for each season, excluding crops with production equal to 0

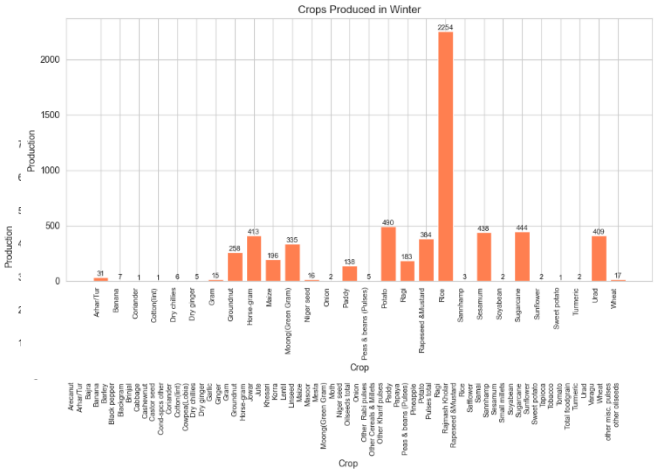
```
min_production_crop = cross_table[cross_table.ne(0)].idxmin(axis=0)


# Print the result
print("Minimum produced crop for each season (excluding crops with production equal to 0):")
print(min_production_crop)
#From this data we can observe that there are few crops which has least production in different seasons like
#Cotton(lint) has least production in Autumn season
# Khesari has least production in Kharif season
#Moth has least production in Rabi season
# Jute has least production in Summer season
# Other Dry Fruit has least production in Whole Year season
# Coriander has least production in Winter season
import pandas as pd
import matplotlib.pyplot as plt
# Assuming cross_table is your crosstab DataFrame
# Iterate over seasons
for season in cross_table.columns:
    # Filter crops produced in the current season
    crops_in_season = cross_table[cross_table[season] > 0].index
    # Plotting crops produced in the current season
    plt.figure(figsize=(10, 6))  # Adjust figure size as needed
    bars = plt.bar(crops_in_season, cross_table.loc[crops_in_season, season], color='coral')
    # Adding production values on top of the bars
    for bar in bars:
        height = bar.get_height()
        plt.text(bar.get_x() + bar.get_width() / 2, height, str(int(height)), ha='center', va='bottom', fontsize=8)
    # Adding title and labels
    plt.title(f'Crops Produced in {season}')
    plt.xlabel('Crop')
```

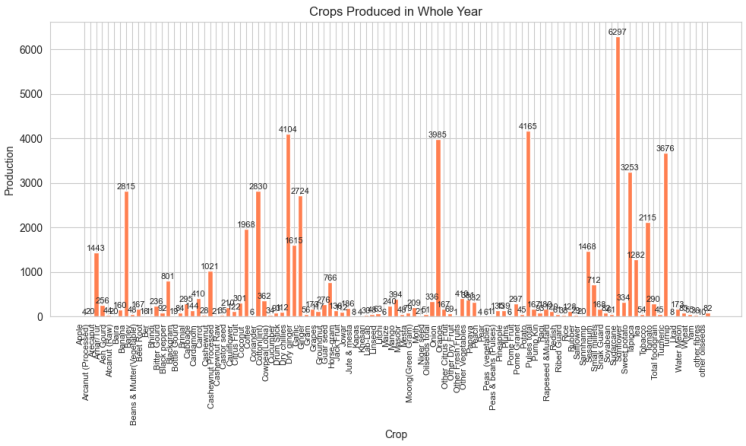
```
plt.ylabel('Production')
plt.xticks(rotation=90, ha='right', fontsize=8)  # Rotate x-axis labels for better readability
# Show plot
plt.tight_layout()  # Adjust layout to prevent overlap of labels
plt.show()
```





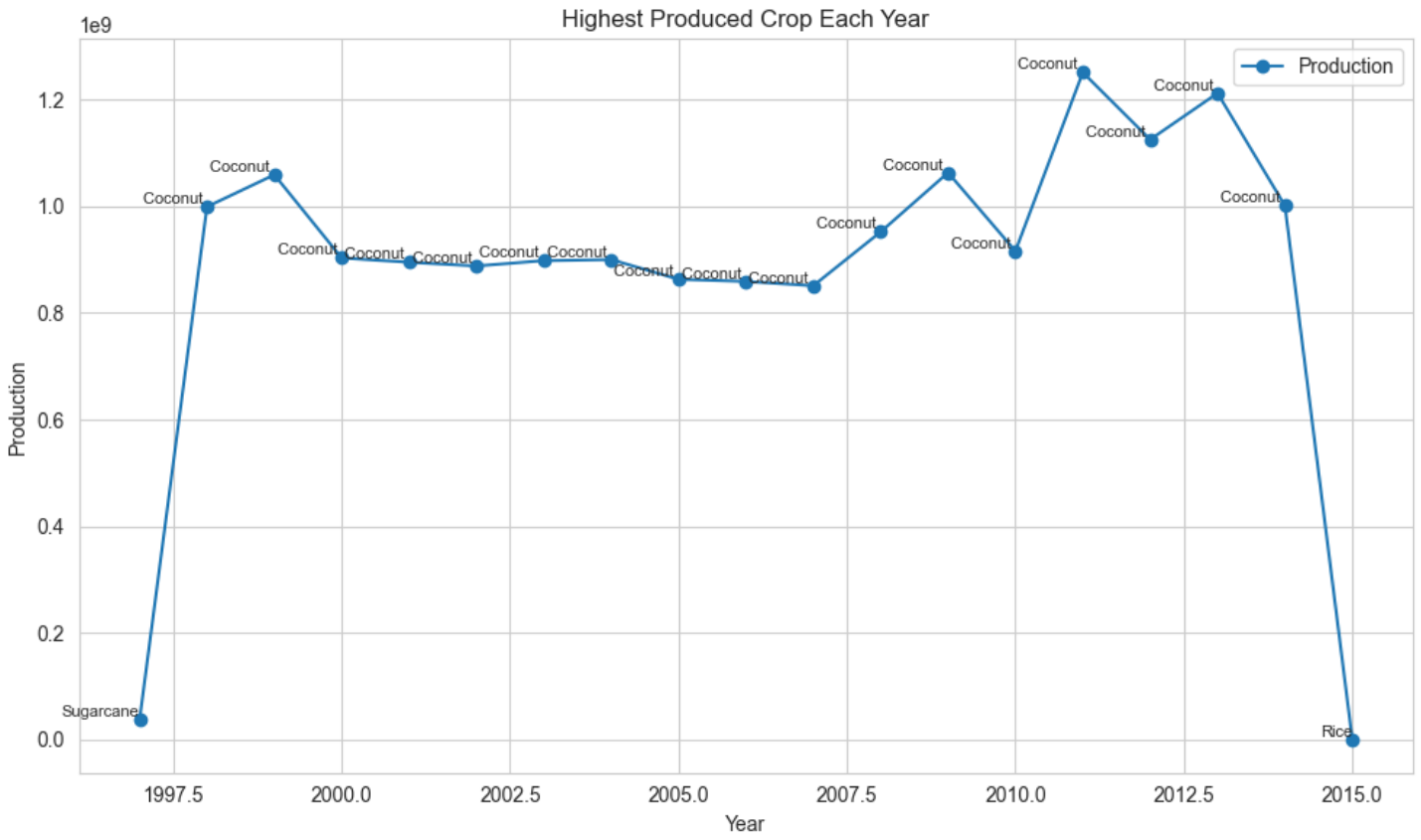






```
# Count the number of zeros for each crop across all seasons
zero_counts = (cross_table == 0).sum(axis=1)


# Find the crop with the most zeros
most_zeros_crop = zero_counts.idxmax()


# Print the result
print("Crop with the least and no production across all seasons:", most_zeros_crop)
#From this data we can see that Apple is the crop that doest not have any production in most of the seasons.
def get_max_crop(group):
   max_index = group['Production'].idxmax()
   max_crop = group.loc[max_index, 'Crop']
   return max_crop


# Group by Season and apply the custom function to get the corresponding year for maximum production
max_crop = new_df.groupby('Season').apply(get_max_crop)


# Reset index to align with DataFrame format
max_crop = max_crop.reset_index(name='Crop')


# Print the result # District_Name
print(max_crop)
merged_fdf = pd.merge(merged_df[['Season', 'Max_Crop_Year', 'Max_State_Names', 'Production','District_Names']],
                      max_crop[['Season', 'Crop']],
                      on='Season',
                      how='inner')


# Display the merged dataframe
print(merged_fdf)
#From this table we can observe that there are few crops which given the highest production when compared to all the 19 years only in a specific season, State,District.
# Find the index of the row with maximum production for each year
```

```
max_production_index_per_year = new_df.groupby('Crop_Year')['Production'].idxmax()


# Select the corresponding rows from the original DataFrame

max_production_per_year = new_df.loc[max_production_index_per_year]


# Reset index and drop the original index column

max_production_per_year = max_production_per_year[['Crop_Year', 'Crop', 'Season',
'Production']].reset_index(drop=True)


# Print the result

print("Highest produced crop for each year:")

print(max_production_per_year)

#From this data we can see that all the 19 different years has their highest produced crops and 'Coconut' crops covers 17
years

#Sugarcane is the highest produced crop in 1997

#Rice is highest produced crop in 2015

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

plt.plot(max_production_per_year['Crop_Year'], max_production_per_year['Production'], marker='o', linestyle='-',
label='Production')


# Adding crop names and seasons as labels for each data point

for i, row in max_production_per_year.iterrows():

    plt.text(row['Crop_Year'], row['Production'], f"{row['Crop']}", fontsize=8, ha='right', va='bottom')


# Adding title and labels

plt.title('Highest Produced Crop Each Year')

plt.xlabel('Year')

plt.ylabel('Production')


# Show plot

plt.grid(True)

plt.legend()

plt.tight_layout()
```


Highest Produced Crop Each Year
1e9
Production
Coconut
Sugarcane
Rice
Production
Year
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
2015.0


Thank
You