Project Name - Crop Production Analysis in India

Project Type - EDA

Contribution - Individual

Presented by - *Priyanshu Tiwari*

**Project Summary **

Project Summary

This project focuses on Exploratory Data Analysis (EDA) of a given dataset.

EDA is a crucial initial step in any data analysis project, aiming to uncover patterns, and gain a comprehensive understanding of the data.

The project, conducted individually by Priyanshu Tiwari, involves the following steps:

1. Data Collection: The dataset is obtained from a reliable source. Details about the source and the process of data acquisition are documented.

2. Data Cleaning: The dataset is thoroughly cleaned to address missing values, and inconsistencies. Techniques used for data cleaning are explained, ensuring data quality for subsequent analysis.

3. Data Exploration: Various exploratory techniques are employed to gain insights into the data. These include:

- Descriptive statistics to summarize central tendencies, dispersion, and distribution of variables.
- Data visualization using bar chart, pie chart, doughnut chart, etc., to reveal patterns and relationships.
- Correlation analysis to identify potential associations between variables.

5. Insights and Findings: Key insights and findings derived from the EDA are presented in a clear and concise manner. These insights could include:

- Identification of important variables influencing the target variable.
- Detection of trends and patterns in the data.

6. Conclusion: The project concludes with a summary of the key findings and their implications. Recommendations for future analysis or modeling based on the EDA results are also provided.

Overall, this EDA project provides a comprehensive understanding of the dataset, laying the foundation for further data-driven decision-making.

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. Make views and dashboards first and also make a story out of it

Let's Begin !

1. Know Your Data

Import Libraries

```
# Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

Dataset Loading

```
# Load Dataset
from google.colab import drive
drive.mount('/content/drive')
path = ('/content/drive/My Drive/unified mentor/Crop Productio
df = pd.read_csv(path)
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.m
```

Dataset First View

```
# Dataset First Look
#Here we use .head() to get first five rows of our dataset and
df.head()
```

	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	NICOBARS	2000	Whole	P	176.0	641.0

Dataset Rows & Columns count

```
# Dataset Rows & Columns count
#Here we use .shape to get the number of rows and columns
df.shape
```

```
(246091, 7)
```

Dataset Information

```
# Dataset Info
#Here we use .info() to get the more information of our datase
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
```

Duplicate Values

```
# Dataset Duplicate Value Count
#Here we use .duplicated() to get the number of duplicate pres
df.duplicated().sum()
```

```
0
```

Missing Values/Null Values

```
# Missing Values/Null Values Count
#Here we use isnull,sum,sort_value where ascending = false bec
df.isnull().sum().sort_values(ascending = False)
```

	0
Production	3730
State_Name	0
District_Name	0
Crop_Year	0
Season	0
Crop	0
Area	0

dtype: int64

```
# Visualizing the missing values
df['Production'][df['Production'].isnull()]
```

	Production
46	NaN
51	NaN
623	NaN
630	NaN
698	NaN
...	...
244128	NaN
244581	NaN
245606	NaN
245644	NaN
245865	NaN

3730 rows × 1 columns

dtype: float64

What did you know about your dataset?

In our dataset their are 246091 rows and 7 columns their name are given below:

1).State_Name - It shows the state.

2).District_Name - It represent the district name of their respective state name.

3).Crop_Year - It represent year of crop cultivated.

4).Season - It represent season of crop.

5).Crop - What type of crop is to be produced

6).Area - Total area in hectares.

7).Production - Total number of production in tonnes.

-->Their are 3730 null values in Production column.

-->Their is no duplicate values in any columns of our dataset.

2. Understanding Your Variables

```
# Dataset Columns
columns = df.columns
columns
```

```
Index(['State_Name', 'District_Name', 'Crop_Year', 'Season', 'Crop', 'Area',
       'Production'],
      dtype='object')
```

```
# Dataset Describe
df.describe()
```

	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

Variables Description

1).State_Name - It shows the state.

2).District_Name - It represent the district name of their respective state name.

3).Crop_Year - It represent year of crop cultivated.

4).Season - It represent season of crop.

5).Crop - What type of crop is to be produced

6).Area - Total area in hectares.

7).Production - Total number of production in tonnes.

Check Unique Values for each variable.

```
# Check Unique Values for each variable.
#Write a function to get unique value of each column
def uni_val(column):
  return column.unique()

#Apply above function in the given dataset to get unique value
unique_value = df.apply(uni_val)
unique_value.rename('unique values')
```

	unique values
State_Name	[Andaman and Nicobar Islands, Andhra Pradesh, ...
District_Name	[NICOBARS, NORTH AND MIDDLE ANDAMAN, SOUTH AND...
Crop_Year	[2000, 2001, 2002, 2003, 2004, 2005, 2006, 201...
Season	[Kharif , Whole Year , Autumn , Rabi ...
Crop	[Arecanut, Other Kharif pulses, Rice, Banana, ...
Area	[1254.0, 2.0, 102.0, 176.0, 720.0, 18168.0, 36...
Production	[2000.0, 1.0, 321.0, 641.0, 165.0, 65100000.0,...

dtype: object

3. ***Data Wrangling***

Data Wrangling Code

```
#Let see the percentage of null value present in production co
print((df['Production'].isnull().sum()*100)/len(df))

#So their are 1.51% data is null in production column
#Here we apply median to fill it in place of null value
production_median = df['Production'].median()
df['Production'].fillna(production_median,inplace = True)
```

```
1.5156994770227274
```

Add **Crop Categories** column

```
#Lets create Crop categories list to add it further into Crop
cereals = ['Wheat', 'Rice', 'Maize', 'Jowar', 'Bajra', 'Ragi',
pulses = ['Moong(Green Gram)', 'Urad, Arhar/Tur', 'Gram', 'Mas
          'Lentil', 'Cowpea(Lobia)', 'Masoor', 'Horse-gram', '
oilseed = ['Groundnut', 'Sunflower', 'Castor seed', 'Sesamum',
Vegetables = ['Potato', 'Onion', 'Beans & Mutter(Vegetable)',
              'Bottle Gourd', 'Garlic', 'Ginger', 'Carrot', 'R
              'Jack Fruit', 'Snak Guard', 'Pump Kin', 'Ash Gou
Fruits = ['Citrus Fruit', 'Grapes', 'Mango', 'Orange', 'Papaya
          'Pear', 'Plums', 'Litchi', 'Ber', 'Other Fresh Fruit
Cash_Crops = ['Cotton(lint)',' Tobacco', 'Jute', 'Rubber', 'Co
Other_Crops = ['Arecanut', 'Cashewnut', 'Sugarcane', 'Sweet po
               'Arcanut (Processed)', 'Atcanut (Raw)', 'Cashew

#Create a function which help to create crop category
def crop_category(row):
  if row['Crop'] in cereals:
    return 'cereals'
  elif row['Crop'] in oilseed:
    return 'oilseed'
  elif row['Crop'] in Vegetables:
    return 'Vegetables'
  elif row['Crop'] in Fruits:
    return 'Fruits'
  elif row['Crop'] in Cash_Crops:
    return 'Cash_Crops'
  else:
    return 'Other_Crops'

#apply the above function to create Crop Categories column
df['Crop Categories'] = df.apply(crop_category,axis = 1)
```

Add **Yield** column

```
#Here we add a new column called yield
df['yield'] = df['Production']/df['Area']
```

What all manipulations have you done and insights you found?

So, Till now we fill null values with median in **Production** column and it is only columns which contain 3730 null values. After that we add two another usefull columns in our dataset first is **Crop Categories** and second is **yield** these two columns will help to get more usefull insights.

4. Data Vizualization, Storytelling & Experimenting with charts : Understand the relationships between variables

Chart - 1

Q1). What is the Total Production by year?

```
year_production = df.groupby('Crop_Year')['Production'].sum().
year_production

fig = px.bar(year_production, x = 'Crop_Year', y = 'Production
                 labels = {'Crop_Year':'Crop Year', 'Productio
fig.update_layout(showlegend = False)
fig.show()
```

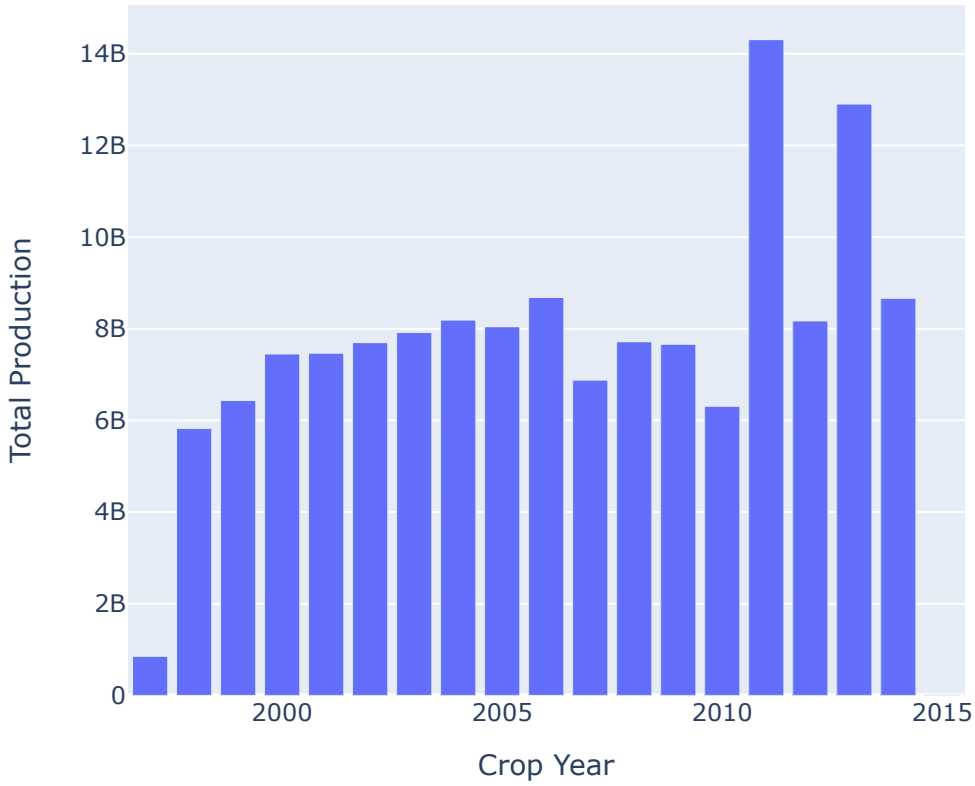


Chart - 2

Q2). What are the most commonly grown crop in each state?

```
#Here we group state name, crop and their count
state_crop = df.groupby(['State_Name','Crop'])['Crop'].count()

#Here we write a function which help tp get most commonly grow
def max_count(Raw_state_crop_df):
  return Raw_state_crop_df.loc[Raw_state_crop_df['count'].idxm

#And finally here our dataframe is ready
most_common_crop = state_crop.groupby('State_Name').apply(max_
most_common_crop
```

	State_Name	Crop	count
0	Andaman and Nicobar Islands	Arecanut	18
1	Andhra Pradesh	Rice	481
2	Arunachal Pradesh	Maize	273
3	Assam	Rice	1356
4	Bihar	Maize	1962
5	Chandigarh	Maize	13
6	Chhattisgarh	Moong(Green Gram)	543
7	Dadra and Nagar Haveli	Rice	31
8	Goa	Rice	38
9	Gujarat	Groundnut	685
10	Haryana	Wheat	317
11	Himachal Pradesh	Potato	145
12	Jammu and Kashmir	Maize	136
13	Jharkhand	Rice	227
14	Karnataka	Maize	1331
15	Kerala	Rice	695
16	Madhya Pradesh	Jowar	1001
17	Maharashtra	Maize	1056
18	Manipur	Rice	107
19	Meghalaya	Rice	286
20	Mizoram	Maize	131
21	Nagaland	Peas & beans (Pulses)	242
22	Odisha	Rice	1346
23	Puducherry	Rice	112
24	Punjab	Rice	338
25	Rajasthan	Rapeseed &Mustard	451
26	Sikkim	Barley	72
27	Tamil Nadu	Groundnut	546
28	Telangana	Rice	313
29	Tripura	Moong(Green Gram)	120
30	Uttar Pradesh	Maize	2552
31	Uttarakhand	Potato	319

32 West Bengal Rice 969

```
#Here we group state name, crop and their count
state_crop = df.groupby(['State_Name','Crop'])['Crop'].count()

#Here we write a function which help tp get most commonly grow
def max_count(Raw_state_crop_df):
  return Raw_state_crop_df.loc[Raw_state_crop_df['count'].idxm

#And finally here our dataframe is ready
most_common_crop = state_crop.groupby('State_Name').apply(max_

fig = px.bar(most_common_crop, x='State_Name', y='count', colo
             title='Most Commonly Grown Crop in Each State',
             labels={'State_Name': 'State', 'count': 'Count of

fig.update_layout(xaxis_tickangle=-45, showlegend=True)
fig.show()
```

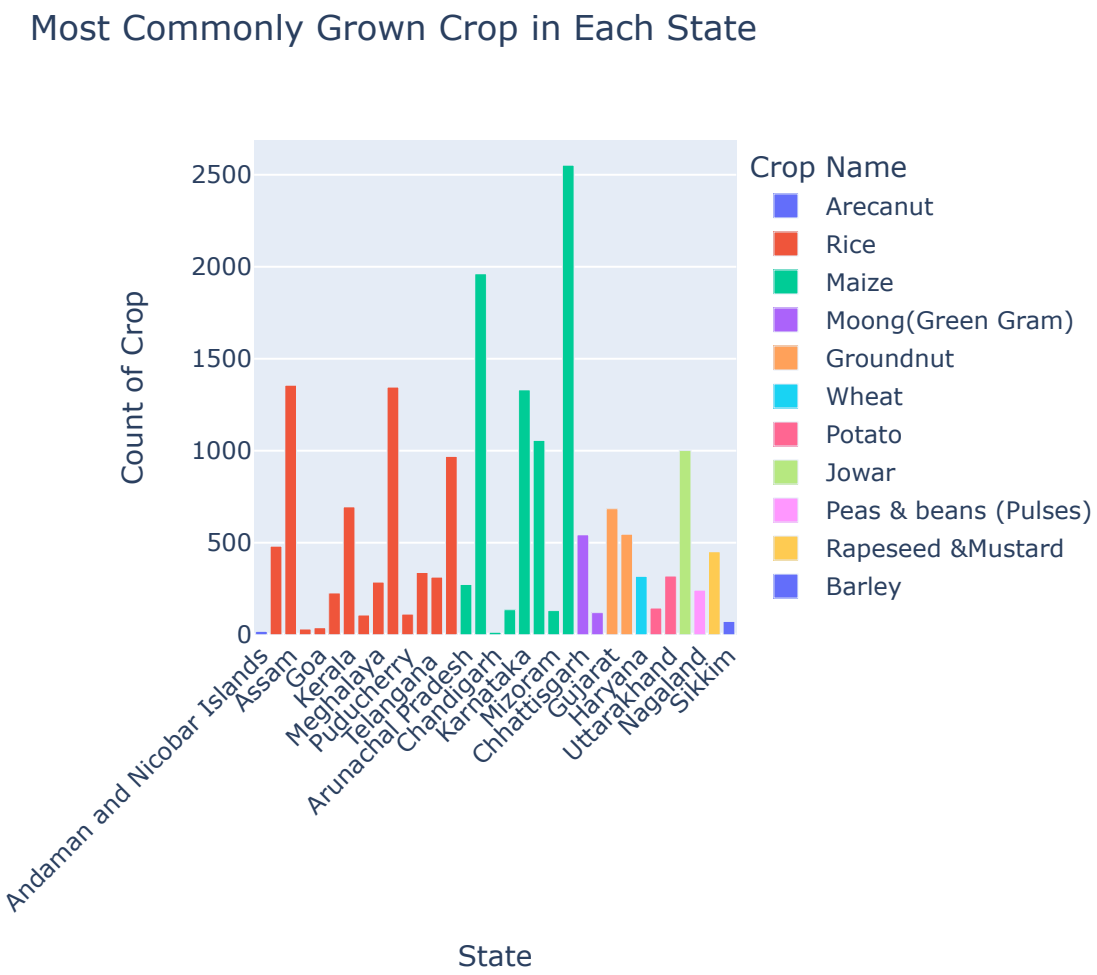
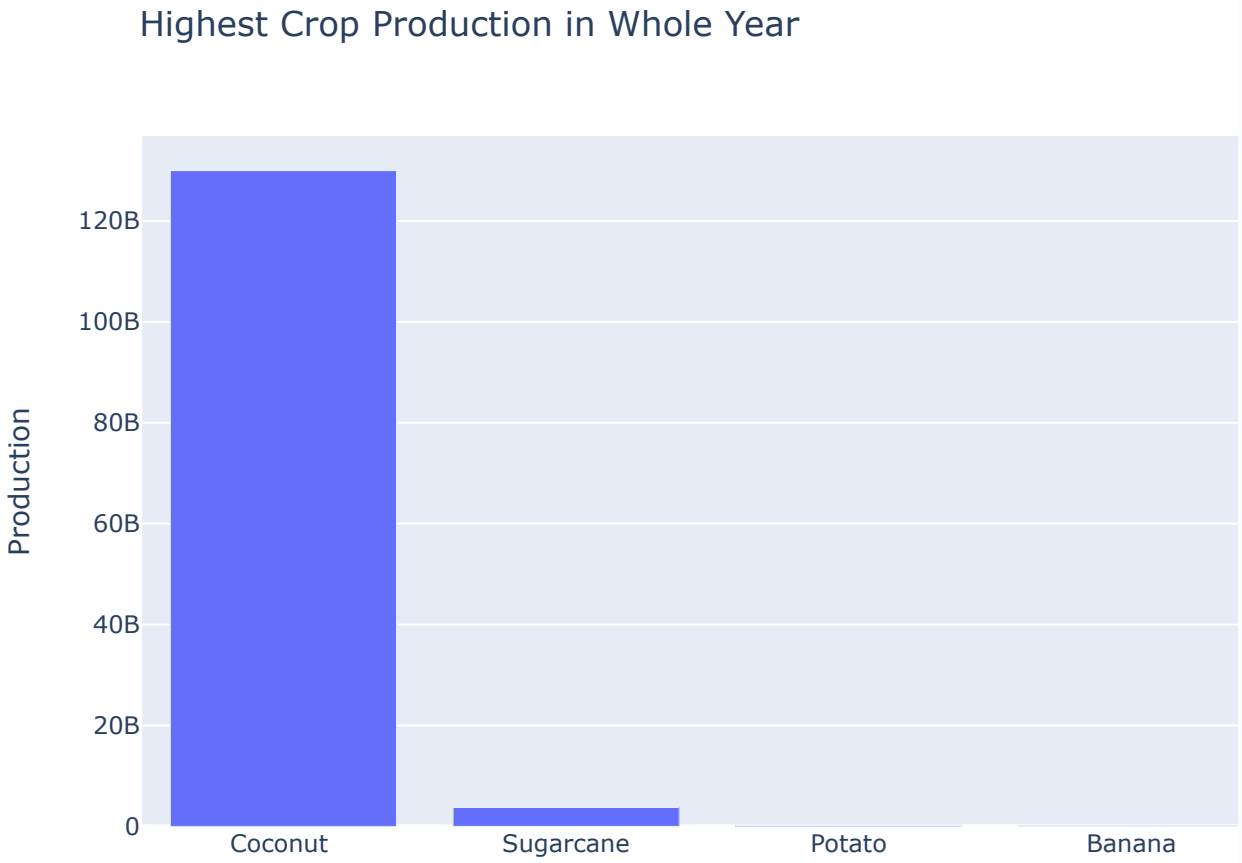
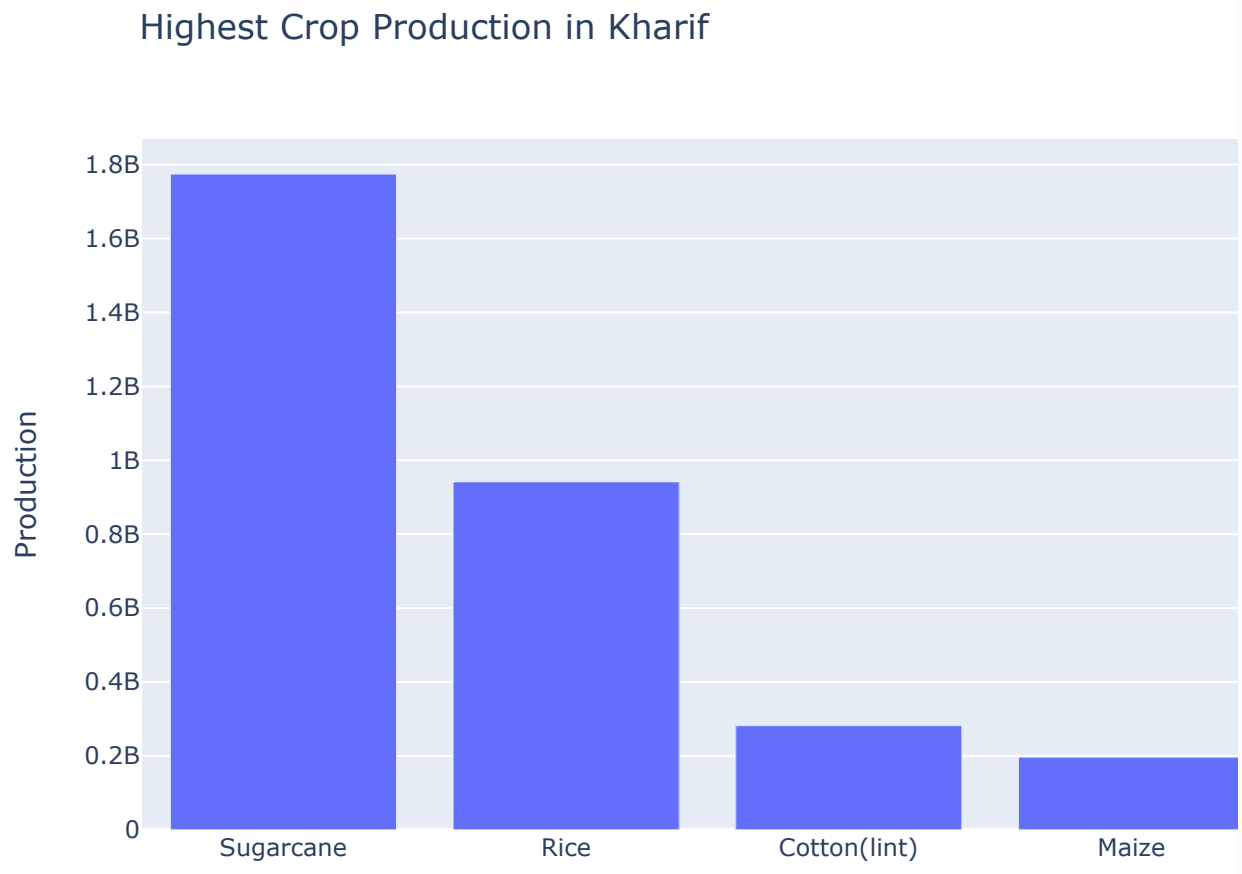


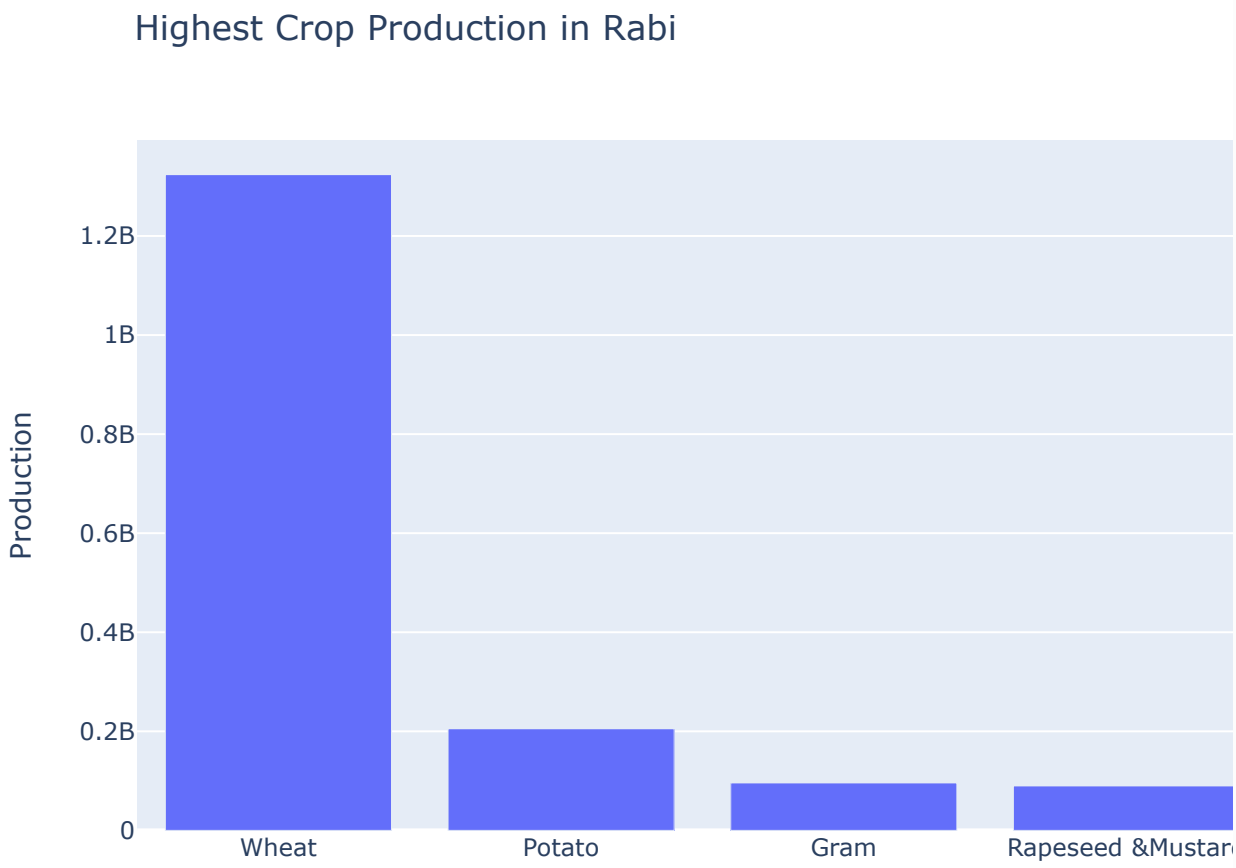
Chart - 3

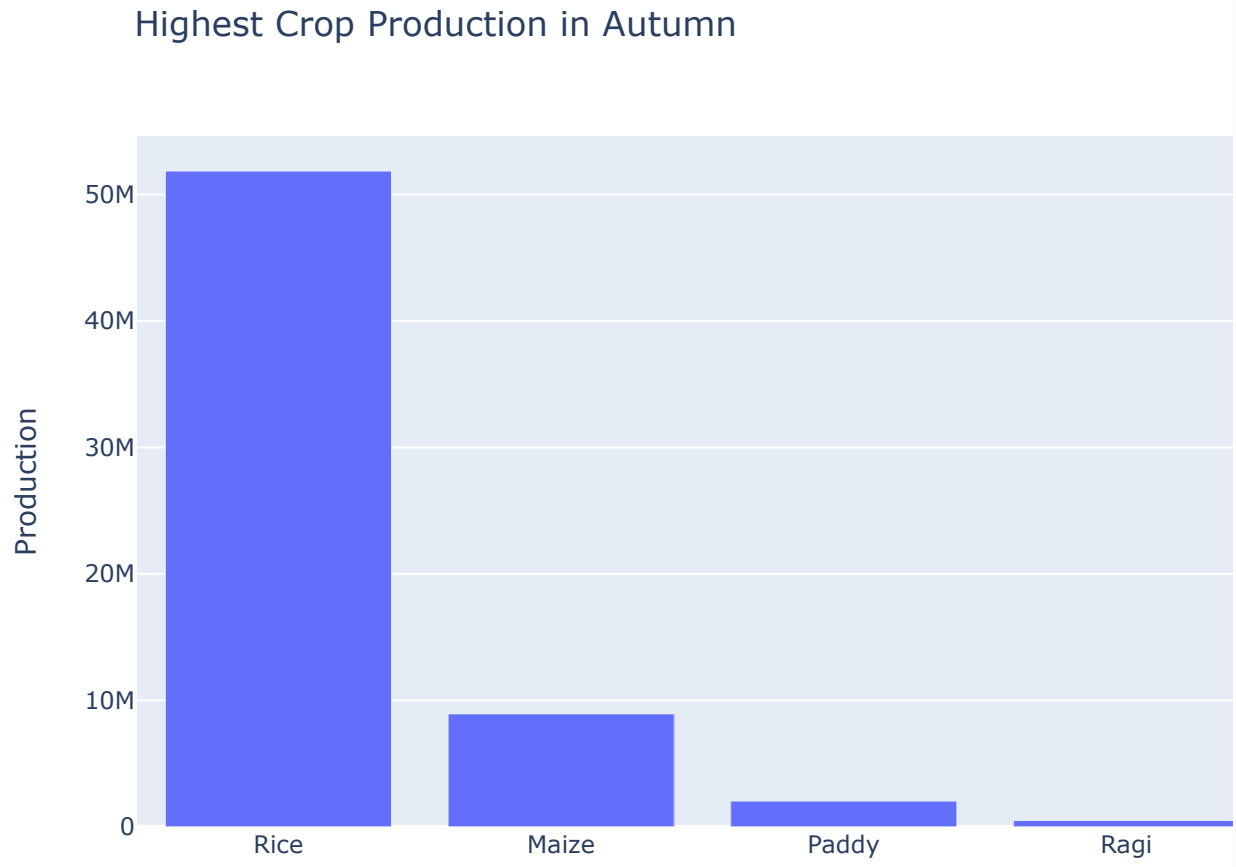
Q3). What is the highest crop production by season?

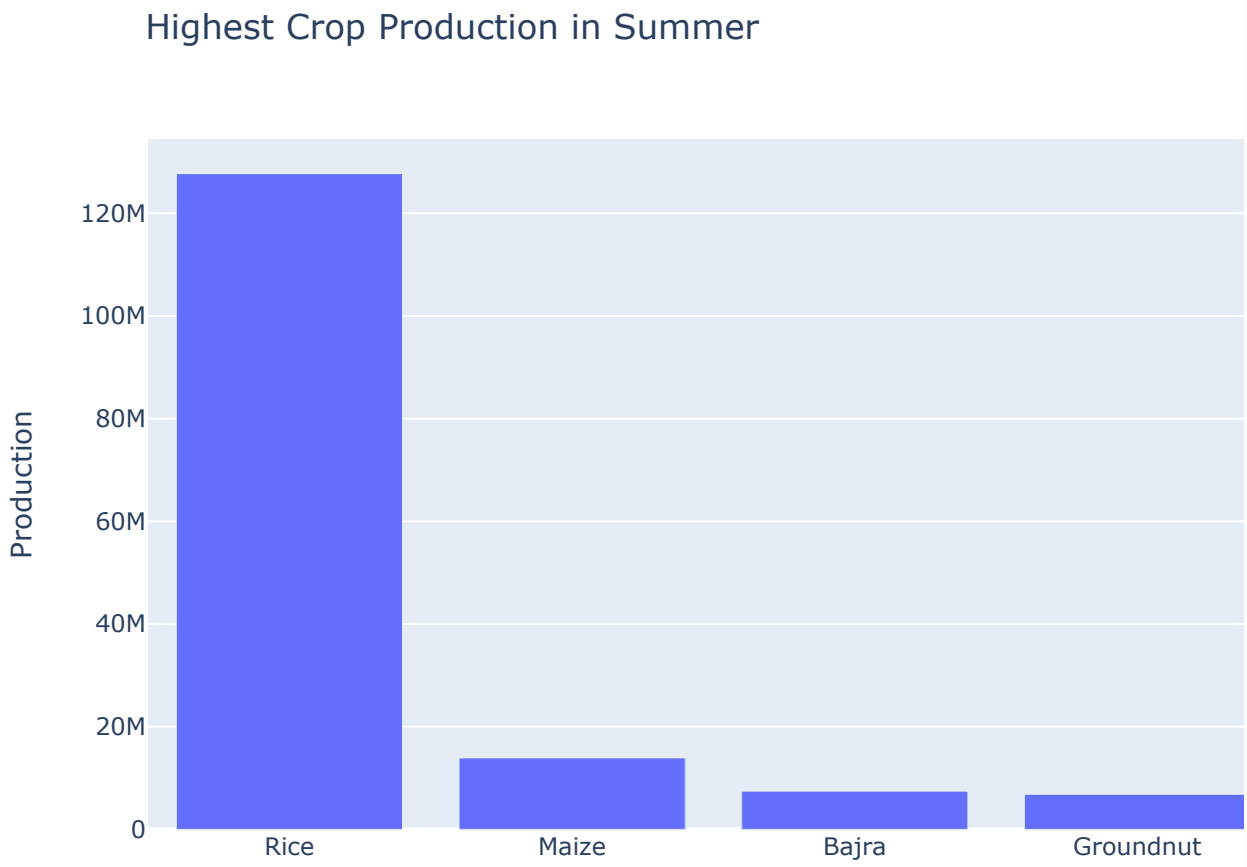
```
season_production = df.groupby(['Crop', 'Season'])['Production
season_list =season_production.Season.unique()

# Filter the data for the winter season
for i in season_list:
  mask =season_production[season_production.Season == i]
  mask=mask.sort_values('Production', ascending=False)
  top_crops = mask.head(10)
  fig = px.bar(top_crops, x='Crop', y='Production', title= 'Hi
  fig.show()
```


Highest Crop Production in Whole Year
Production
120B
100B
80B
60B
40B
20B
0
Coconut
Sugarcane
Potato
Banana


Highest Crop Production in Kharif
Production
1.8B
1.6B
1.4B
1.2B
1B
0.8B
0.6B
0.4B
0.2B
0
Sugarcane
Rice
Cotton(lint)
Maize


Highest Crop Production in Rabi
Production
1.2B
1B
0.8B
0.6B
0.4B
0.2B
0
Wheat
Potato
Gram
Rapeseed &Mustar


Highest Crop Production in Autumn
Production
50M
40M
30M
20M
10M
0
Rice
Maize
Paddy
Ragi


Highest Crop Production in Summer
Production
120M
100M
80M
60M
40M
20M
0
Rice
Maize
Bajra
Groundnut

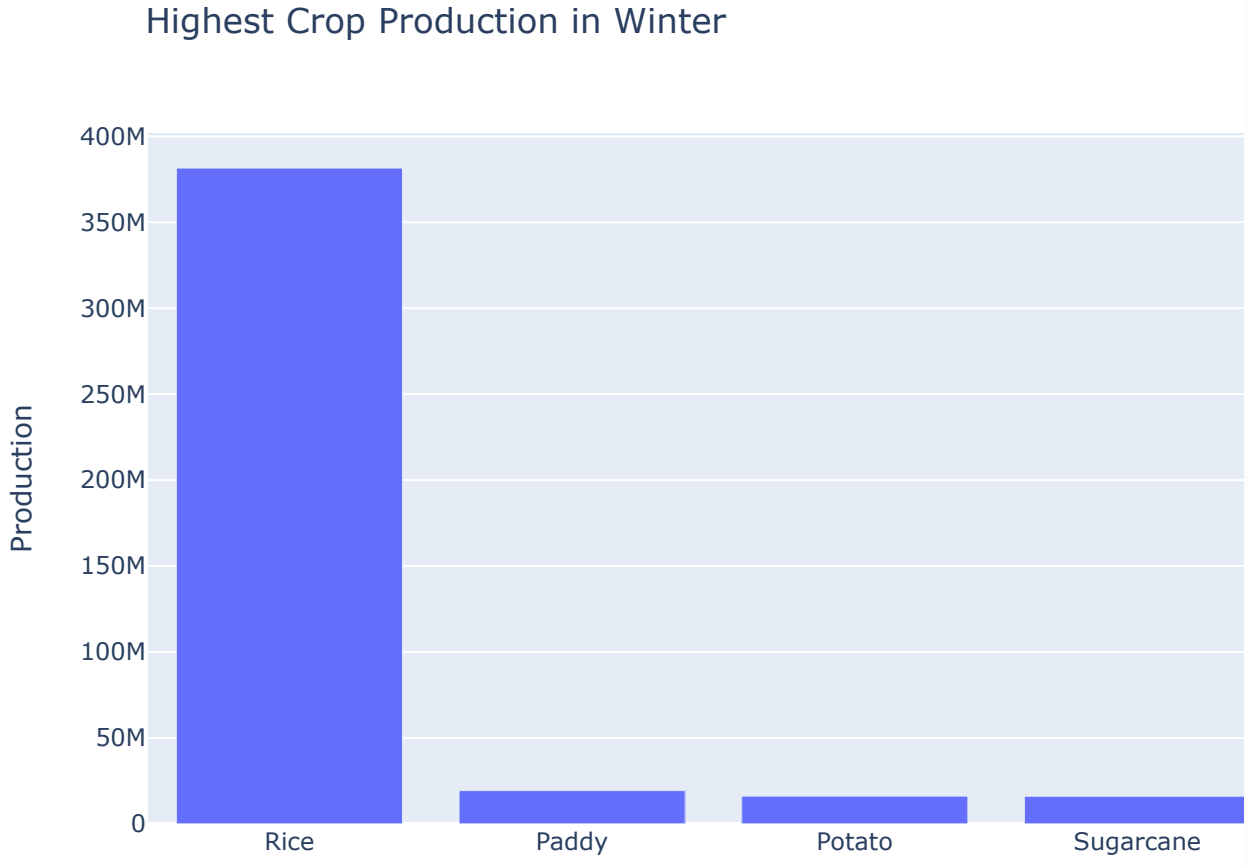

Highest Crop Production in Winter
Production
400M
350M
300M
250M
200M
150M
100M
50M
0
Rice
Paddy
Potato
Sugarcane

Chart - 4

Q4).What is the distribution of crop production across different seasons?

```
crop_seasons = df.groupby('Season')['Production'].count().rese
colors = ['blue', 'yellowgreen', 'red', 'Purple', 'Pink', 'ora

plt.figure(figsize = (10,8))
plt.pie(crop_seasons['Production'],autopct = '%1.2f%%',colors
plt.title('Percentage of Production for Crops in Each Season',
plt.legend(crop_seasons['Season'])
plt.show()
```

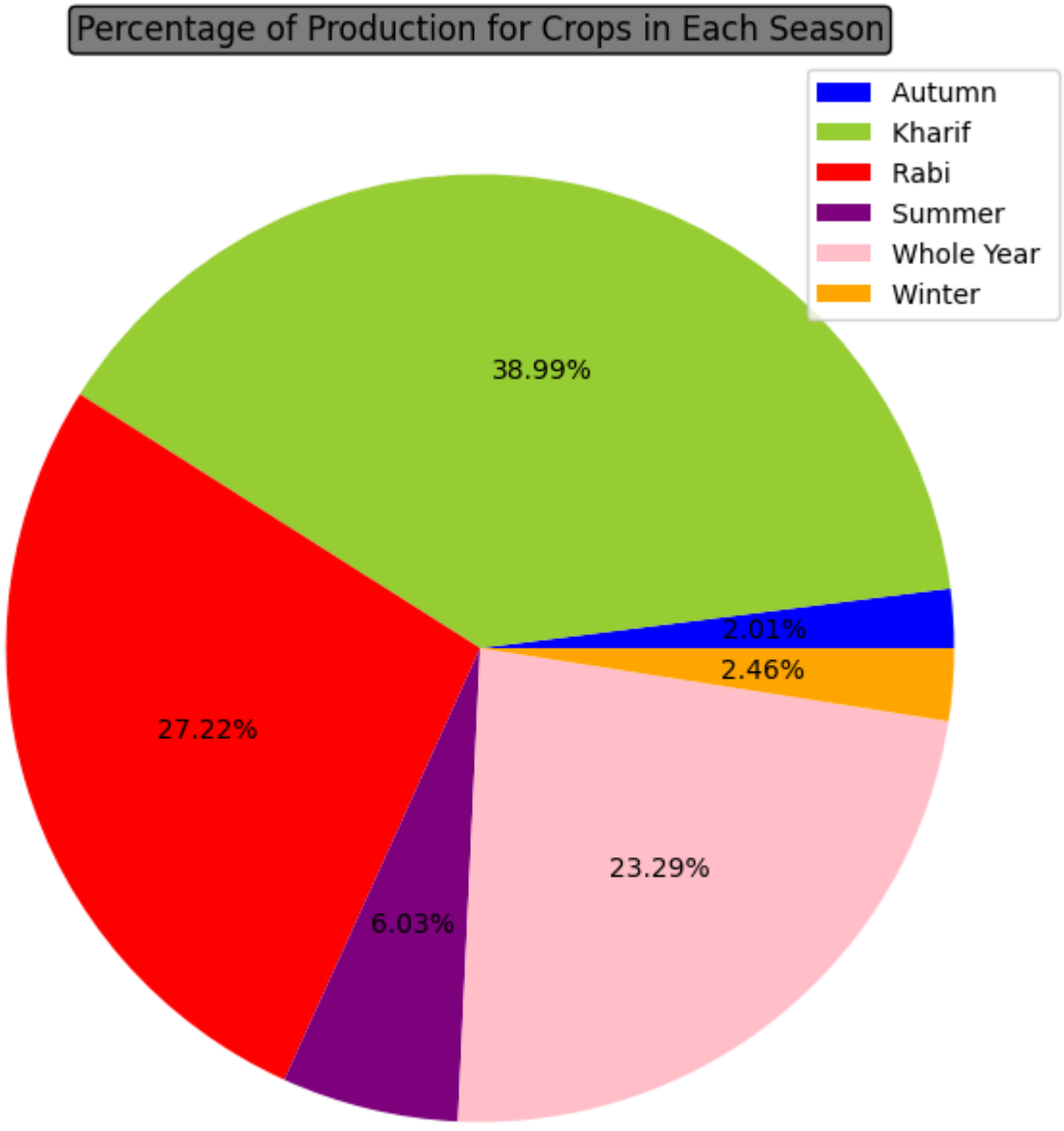


Chart - 5

Q5).What is the change in production over the years in each state?

```
state_production_per_year = df.groupby(['State_Name', 'Crop_Ye
state_production_per_year = state_production_per_year.sort_val

fig = px.line(state_production_per_year, x='Crop_Year', y='Pro
              title='Change in Production in Each State Over t
fig.show()
```

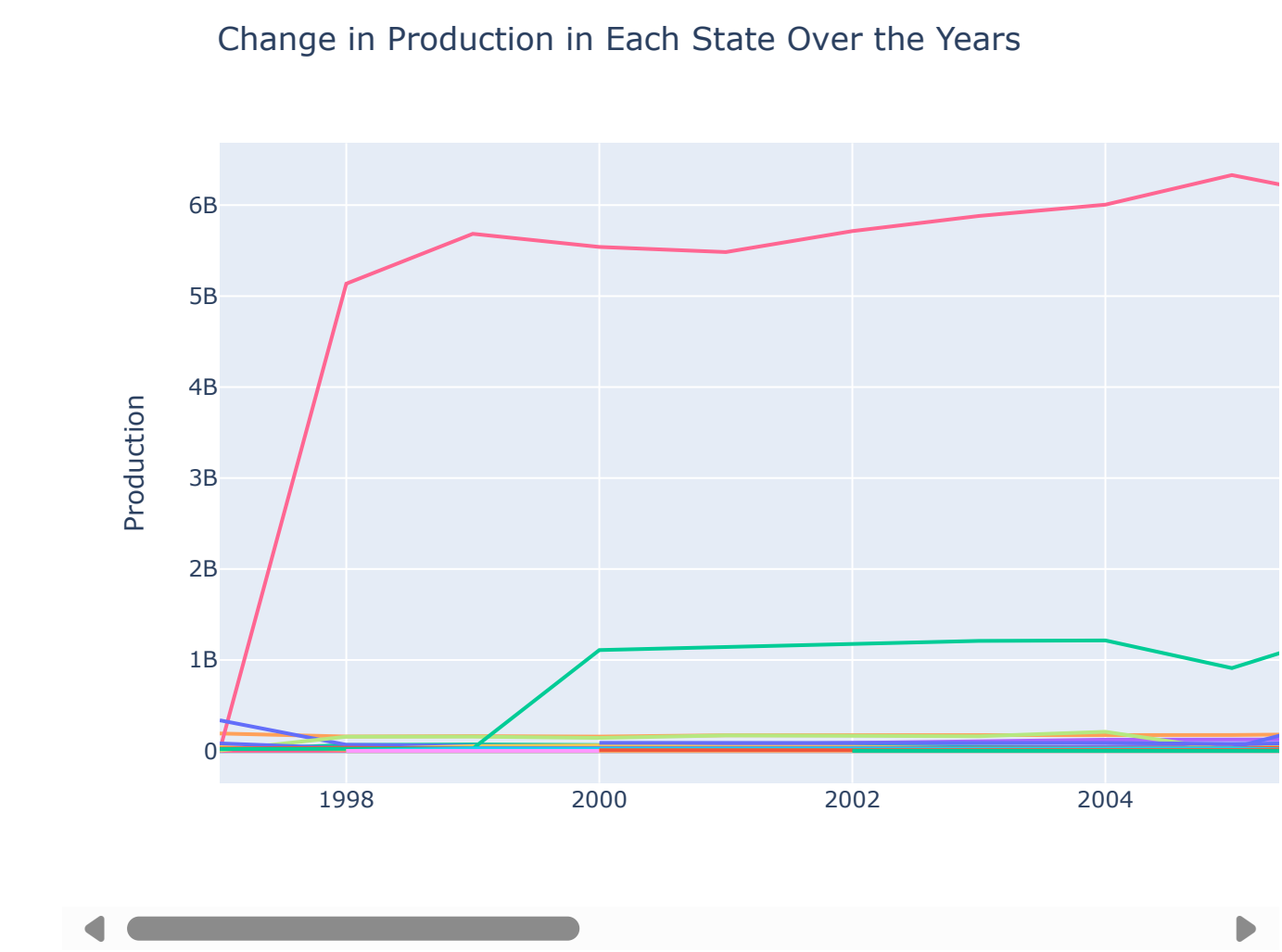


Chart - 6

Q6).What is the avg yield per hectare for crop category?