CROP PRODUCTION ANALYSIS IN INDIA

Ragi Rajesh

July - 2024

Abstract

This project aims to analyze crop production data in India to identify key indicators and metrics that influence crop production. Using data science techniques, the goal is to predict crop production and provide valuable insights. Various tools and technologies, including Python, Tableau, and Power BI, are used to visualize the data and develop predictive models.

Introduction

Background

The agriculture sector is crucial for the overall supply chain and is expected to undergo significant evolution due to advancements in technology. This project aims to develop a Business-to-Business collaboration platform for the agri-food sector, facilitating effective and flexible collaboration among various stakeholders.

Problem Statement

The dataset provides extensive information on crop production in India over several years. The primary goal is to predict crop production and uncover important insights, highlighting key indicators and metrics that influence crop production. The project involves creating views, dashboards, and a compelling story from the data.

Objectives

1. Analyze crop production data to identify trends and patterns.
2. Develop predictive models to forecast crop production.
3. Create visualizations and dashboards to present insights.
4. Highlight key indicators and metrics influencing crop production.

Literature Review

This section summarizes existing work and research relevant to the project and identifies gaps that the project aims to fill.

Data Description

Data Sources:
https://drive.google.com/file/d/1b3E1vpDSYpHe8YlNs3jkt30Lx6acf0Uo/view?usp=share_link

Data Characteristics: The dataset includes attributes such as State_Name, District_Name, Crop_Year, Season, Crop, Area, and Production. It contains data over several years, providing a comprehensive view of crop production in India.

Data Preprocessing:

```
python Copy code
import pandas as pd
import numpy as np

# Load the data
crp_production = pd.read_csv('crop production data.csv')

# Check for missing values and duplicates
print(crp_production.isnull().sum()) crp_production
= crp_production.dropna()
print(crp_production.isnull().sum())
print(crp_production.duplicated().sum())

# Describe the data
print(crp_production.describe())

# Encode categorical variables crp_production['State_Name']
=
crp_production['State_Name'].astype('category').cat.codes + 1
crp_production['District_Name'] =
crp_production['District_Name'].astype('category').cat.codes + 1
crp_production['Season'] =
crp_production['Season'].astype('category').cat.codes + 1
crp_production['Crop'] =
crp_production['Crop'].astype('category').cat.codes + 1
```

Methodology

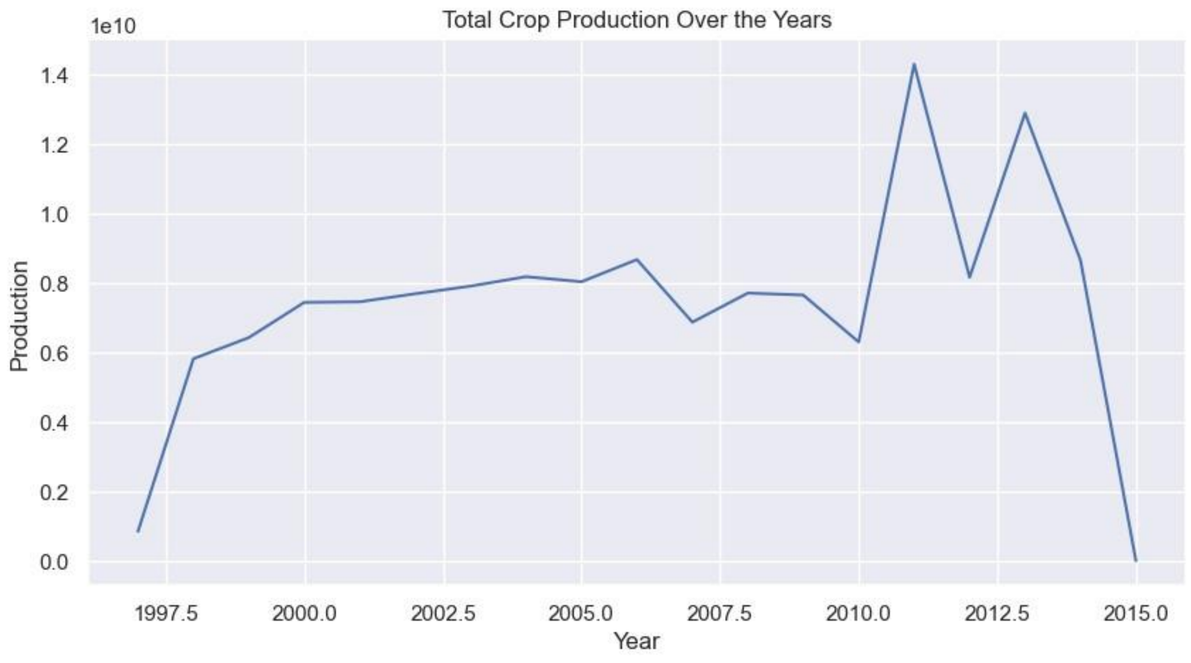
Exploratory Data Analysis (EDA):

```
python Copy
code
import matplotlib.pyplot as plt import
seaborn as sns

# Total crop production over the years plt.figure(figsize=(10,
5))
crp_production.groupby('Crop_Year')['Production'].sum().plot(kind='line')
plt.title('Total Crop Production Over the Years') plt.xlabel('Year')
plt.ylabel('Production') plt.show()
```

```
# Crop production distribution by state fig,
ax = plt.subplots(figsize=(15, 10))
sns.lineplot(x='State_Name', y='Production', data=crp_production, ax=ax)
ax.set_title('Crop Production Distribution by State')
ax.set_xlabel('State') ax.set_ylabel('Production')
ax.tick_params(axis='x', rotation=90)
plt.tight_layout() plt.show()
```

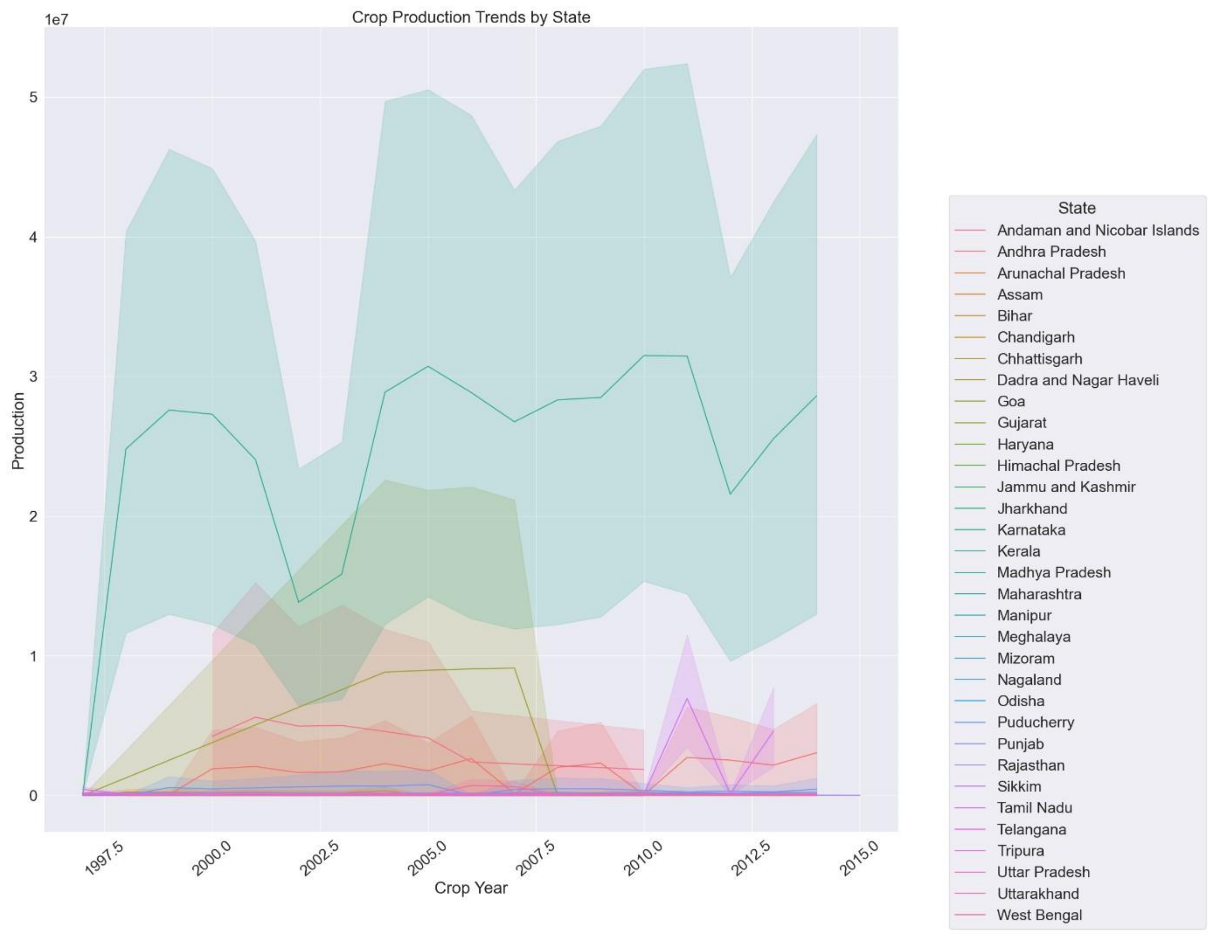
Visualizations:

Total Crop Production Over the Years



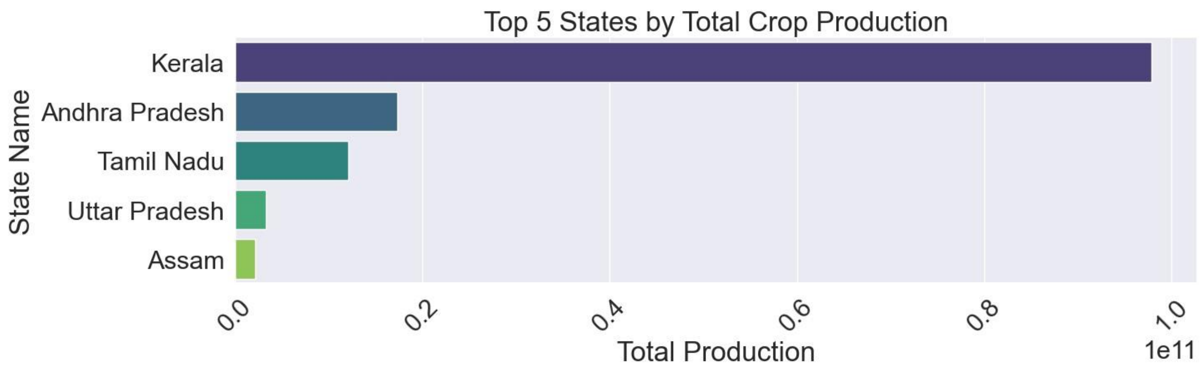
Crop Production Trends by State

```
# Crop production trends by state over the years fig,
ax = plt.subplots(figsize=(25, 20))
sns.lineplot(x='Crop_Year', y='Production', hue='State_Name',
data=crp_production, ax=ax)
ax.set_title('Crop Production Trends by State')
ax.set_xlabel('Crop Year') ax.set_ylabel('Production')
ax.legend(title='State', bbox_to_anchor=(1.05, 0.8))
ax.tick_params(axis='x', rotation=40)
plt.tight_layout() plt.show()
```



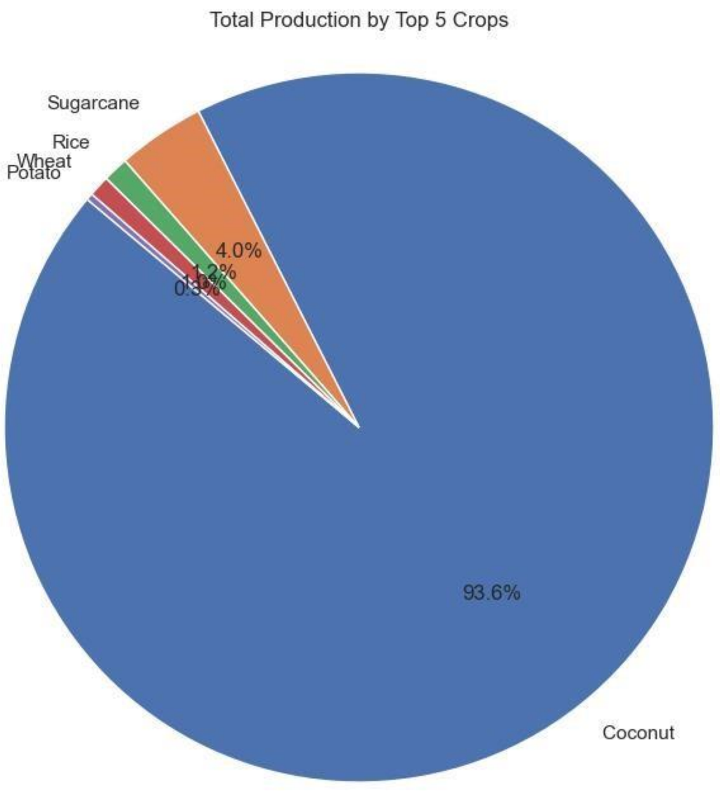
Top 5 States by Total Crop Production

```
# Top 5 states by total production state_production
=
crp_production.groupby('State_Name')['Production'].sum().reset_index()
top_states = state_production.nlargest(5, 'Production')

fig, ax = plt.subplots(figsize=(15, 5))
sns.barplot(x='Production', y='State_Name', data=top_states,
palette='viridis', ax=ax)
ax.set_title('Top 5 States by Total Crop Production')
ax.set_xlabel('Total Production')
ax.set_ylabel('State Name') ax.tick_params(axis='x',
rotation=45) plt.tight_layout() plt.show()
```

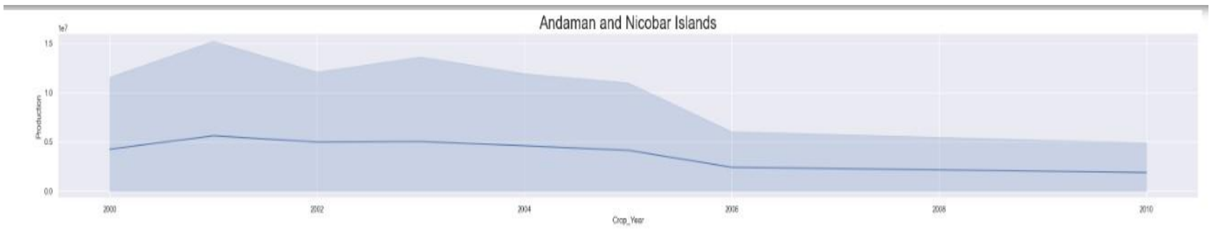


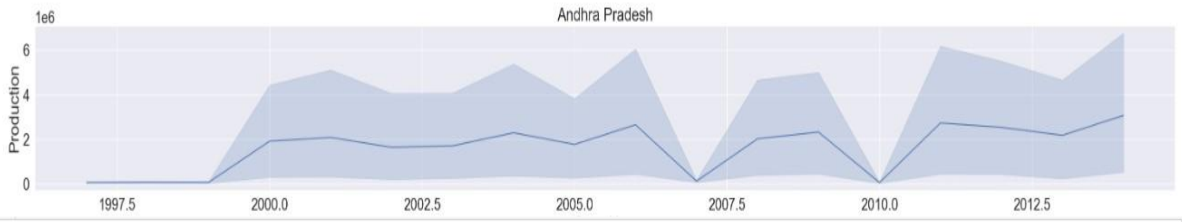
Total Production by Top 5 Crops

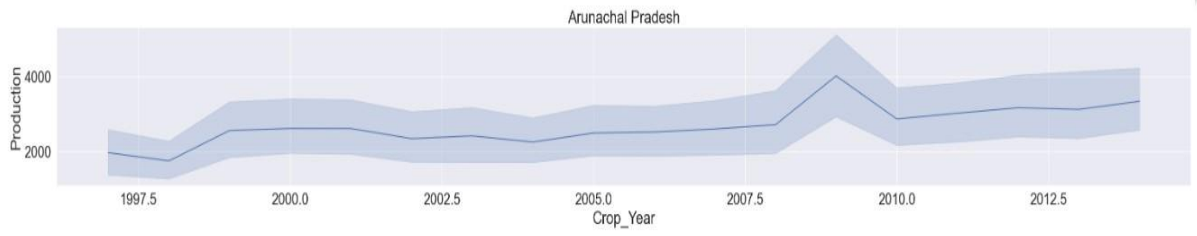
```
# Total production by top 5 crops crop_production
=
crp_production.groupby('Crop')['Production'].sum().reset_index().sort_value
s(by='Production', ascending=False) top_5_crops = crop_production.head(5)

plt.figure(figsize=(12, 8))
plt.pie(top_5_crops['Production'], labels=top_5_crops['Crop'],
autopct='%1.1f%%', startangle=140) plt.title('Total Production
by Top 5 Crops') plt.axis('equal') plt.show()
```

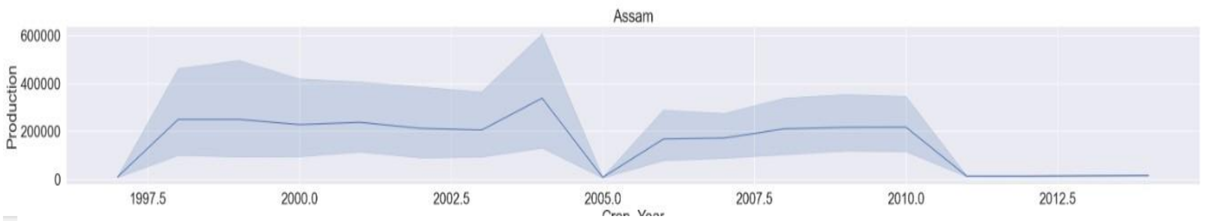


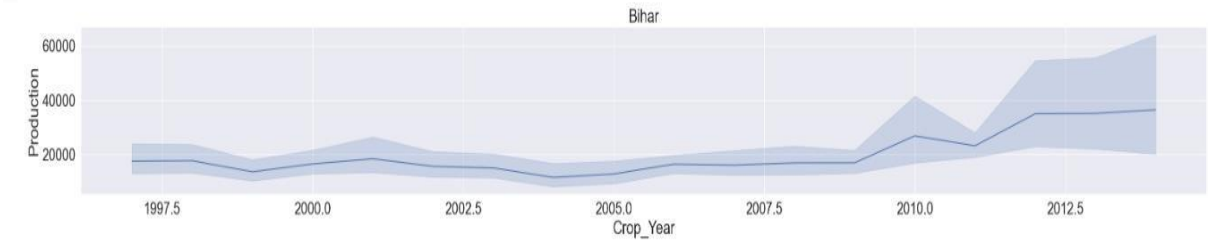
Year-wise Production of individual State and Union Territory

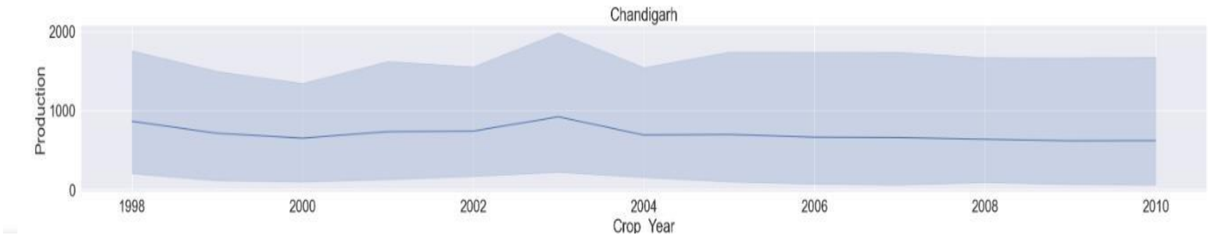


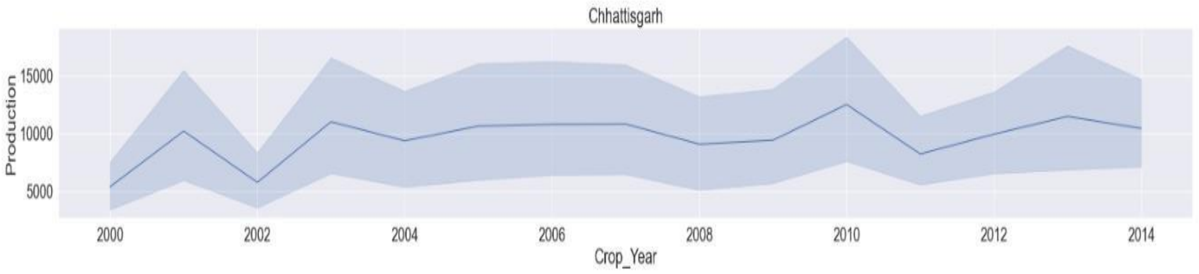


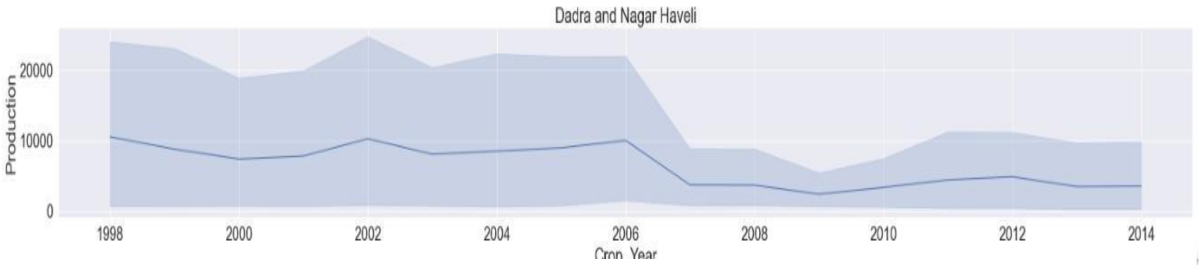


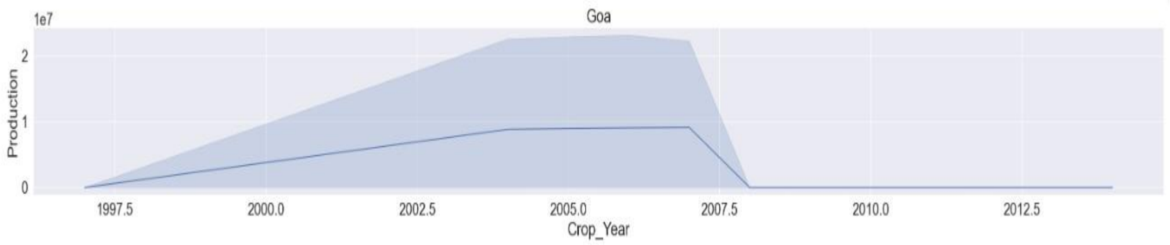


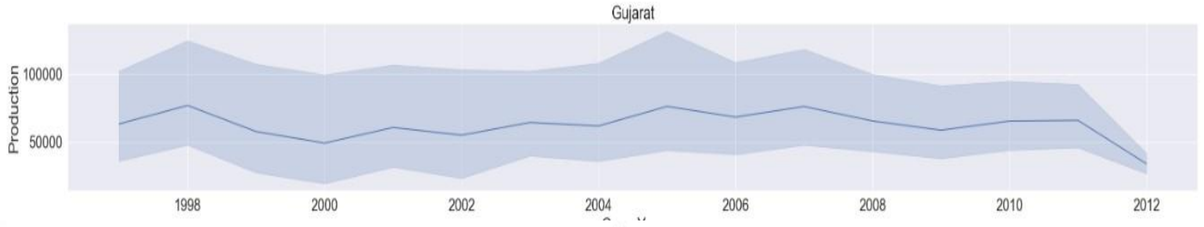


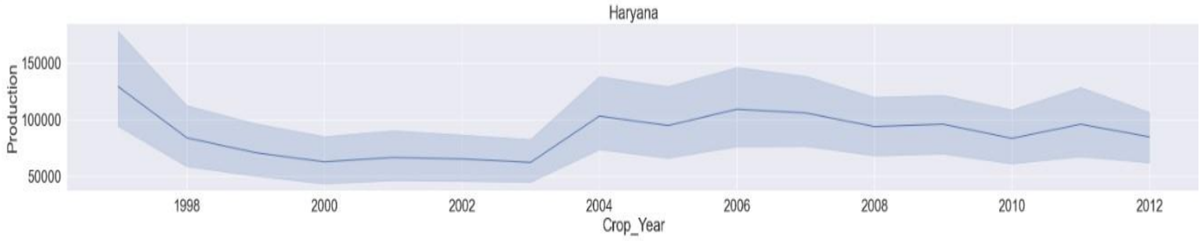


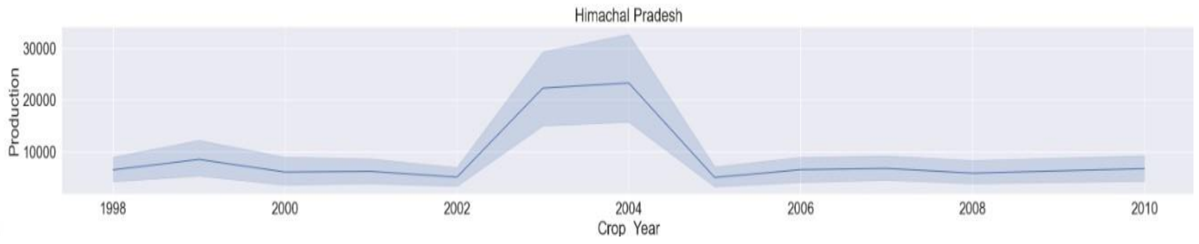

Chhattisgarh
Production
5000
10000
15000
2000
2002
2004
2006
2008
2010
2012
2014
Crop_Year

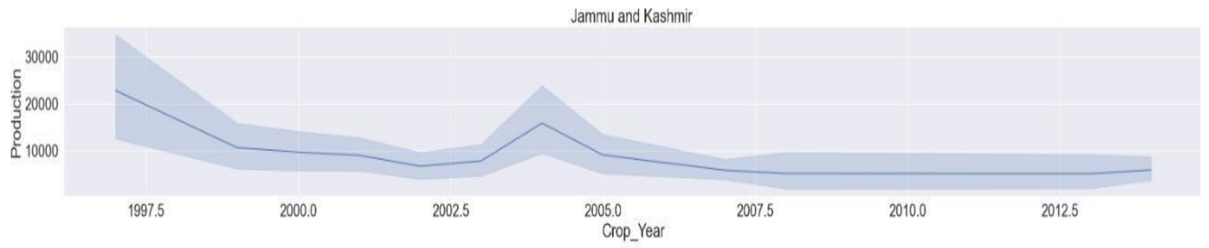

Dadra and Nagar Haveli
Production
0
10000
20000
1998
2000
2002
2004
2006
2008
2010
2012
2014
Crop_Year

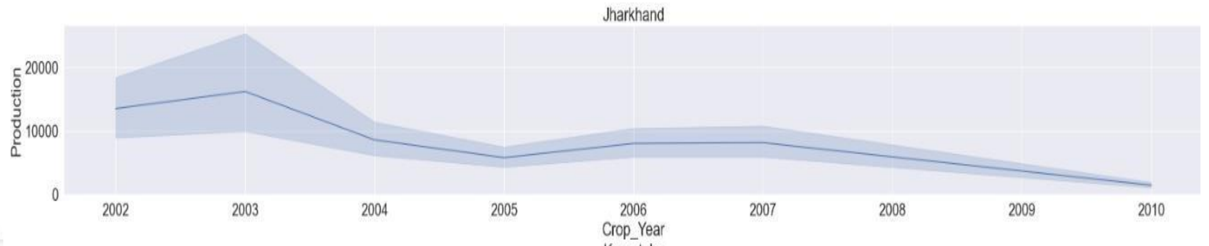

Goa
1e7
Production
0
1
2
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

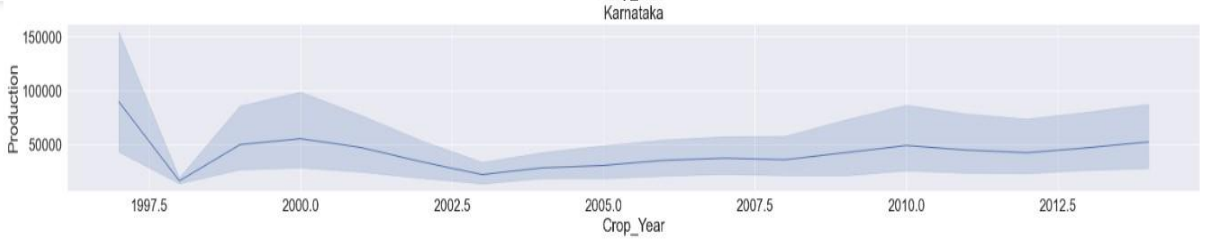

Gujarat
Production
50000
100000
1998
2000
2002
2004
2006
2008
2010
2012

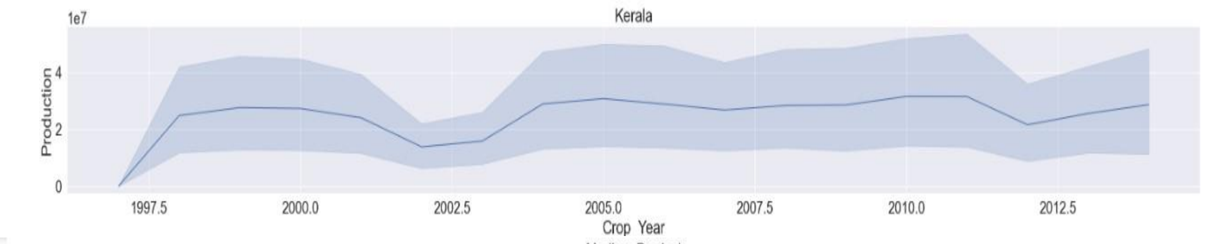

Haryana
Production
50000
100000
150000
1998
2000
2002
2004
2006
2008
2010
2012
Crop_Year

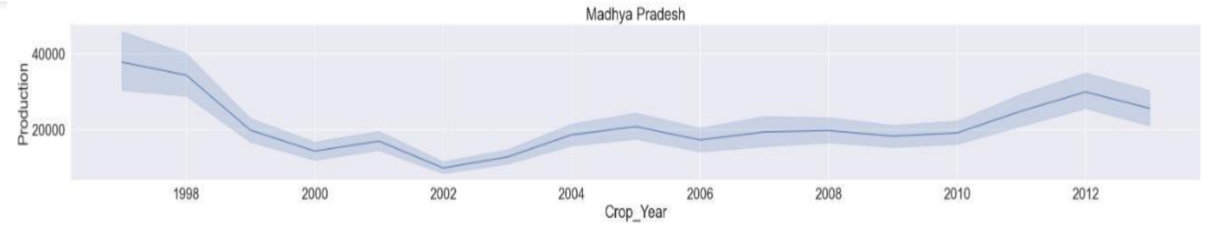

Himachal Pradesh
Production
10000
20000
30000
1998
2000
2002
2004
2006
2008
2010
Crop_Year

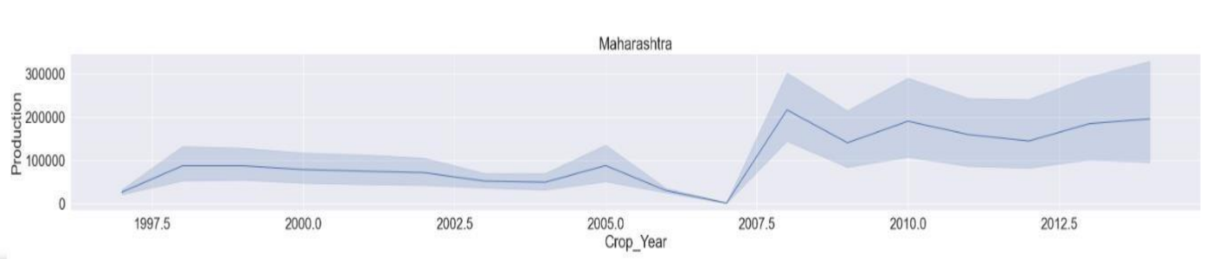

Jammu and Kashmir
Production
30000
20000
10000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

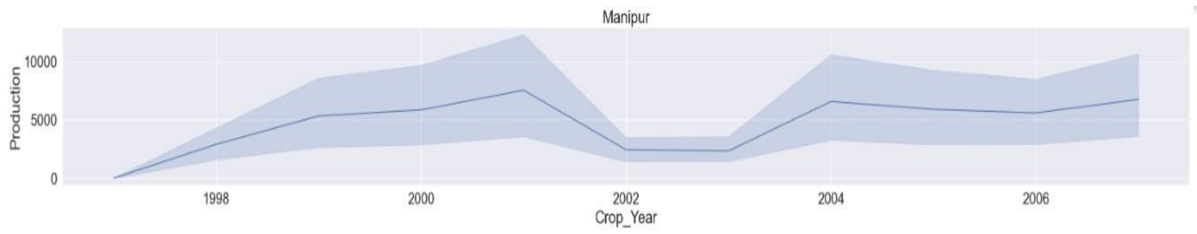

Jharkhand
Production
20000
10000
0
2002
2003
2004
2005
2006
2007
2008
2009
2010
Crop_Year

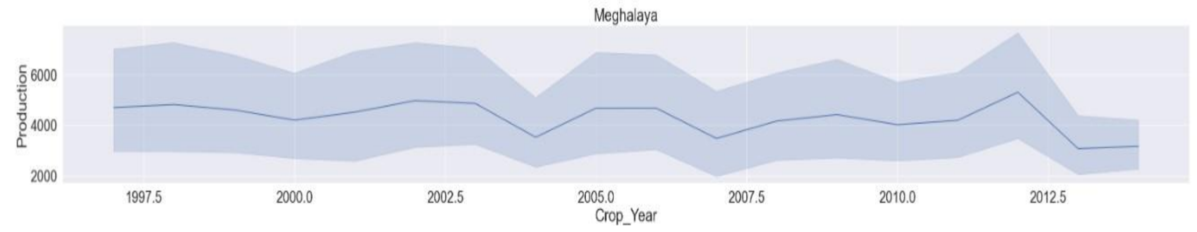

Karnataka
Production
150000
100000
50000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

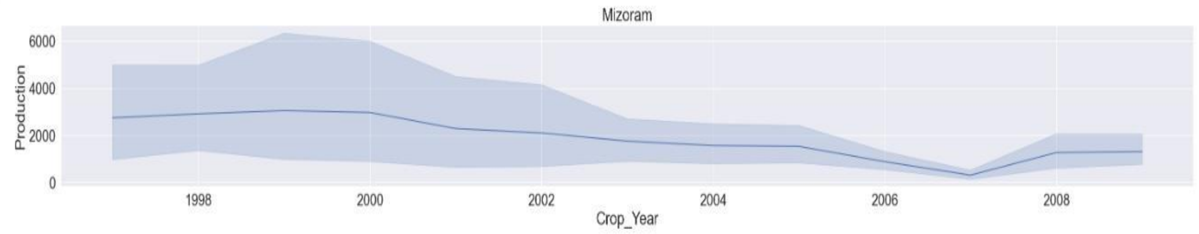

Kerala
1e7
Production
4
2
0
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop Year

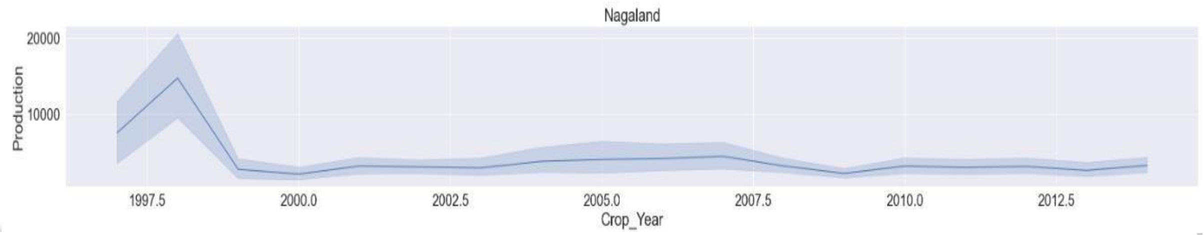

Madhya Pradesh
Production
40000
20000
1998
2000
2002
2004
2006
2008
2010
2012
Crop_Year

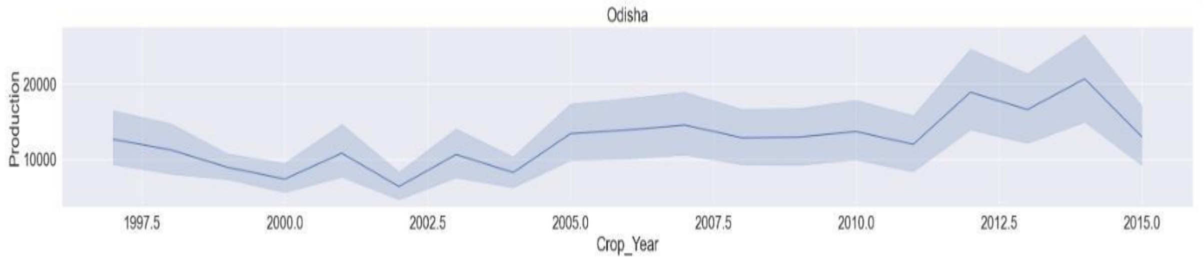

Maharashtra
Production
300000
200000
100000
0
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

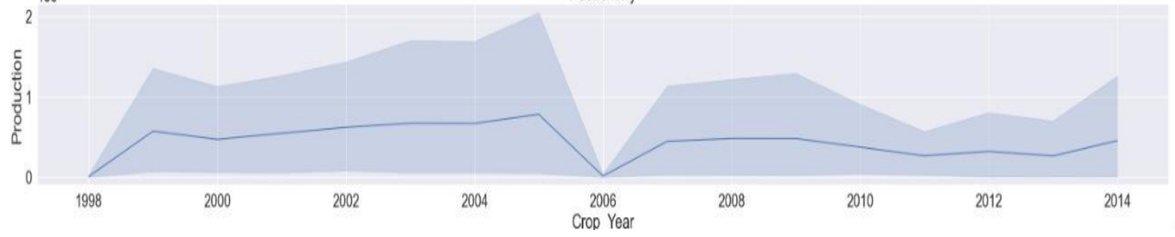

Manipur
Production
0
5000
10000
1998
2000
2002
2004
2006
Crop_Year

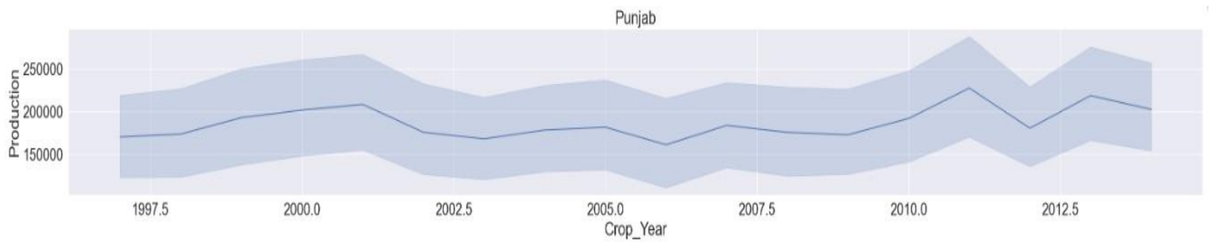

Meghalaya
Production
2000
4000
6000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

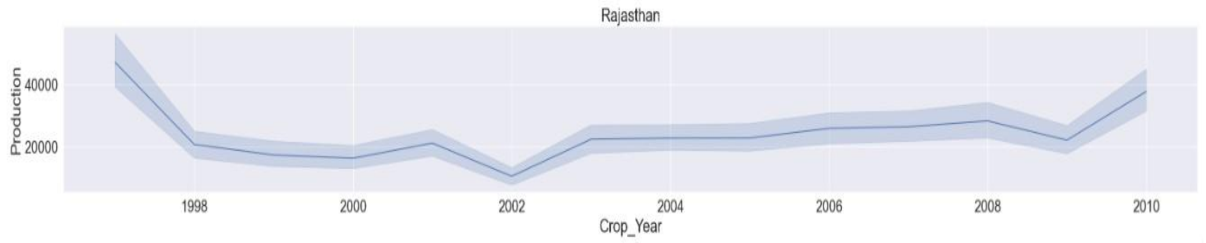

Mizoram
Production
0
2000
4000
6000
1998
2000
2002
2004
2006
2008
Crop_Year

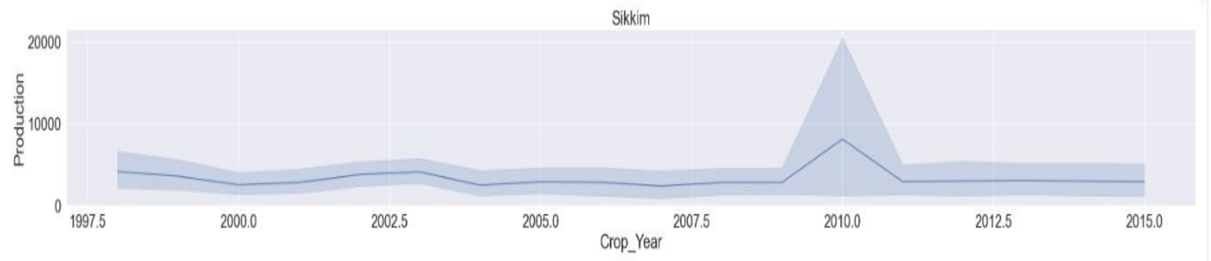

Nagaland
Production
10000
20000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

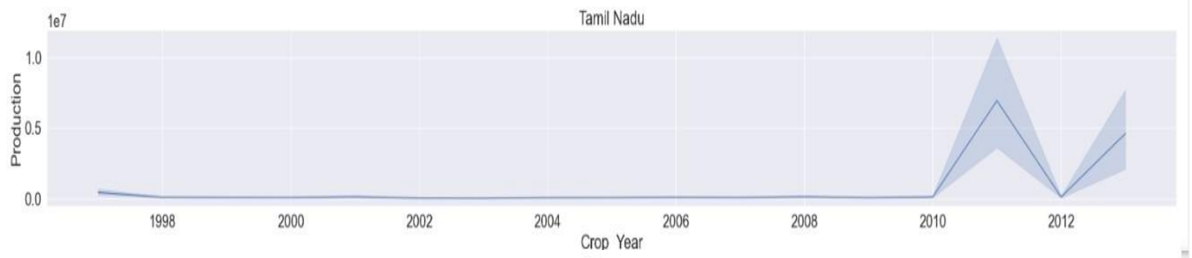

Odisha
Production
10000
20000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
2015.0
Crop_Year

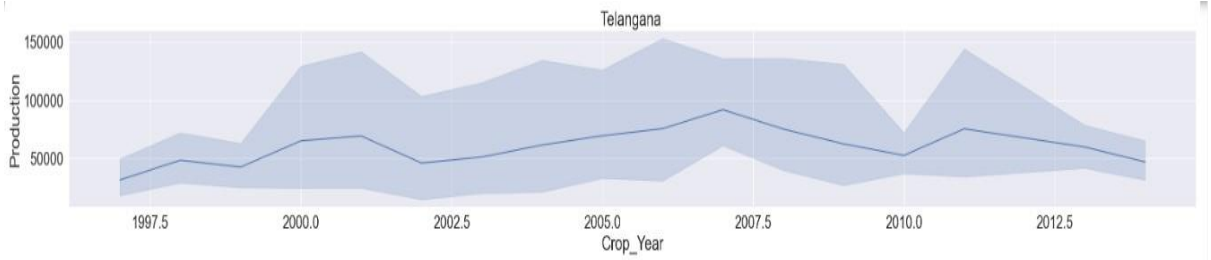

Puducherry
1e6
Production
0
1
2
1998
2000
2002
2004
2006
2008
2010
2012
2014
Crop_Year

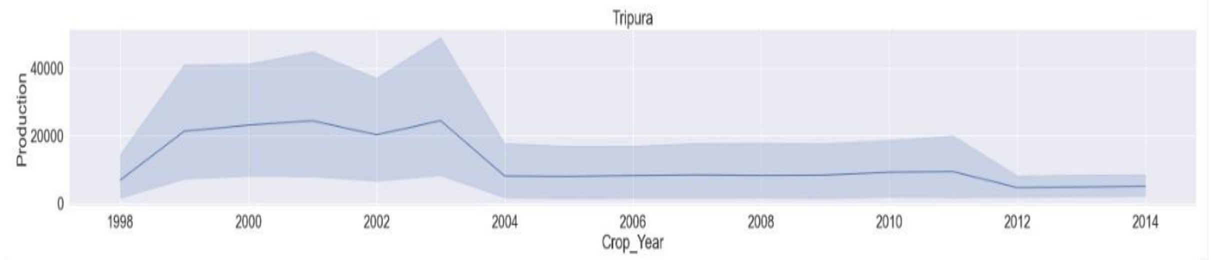

Punjab
Production
250000
200000
150000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year

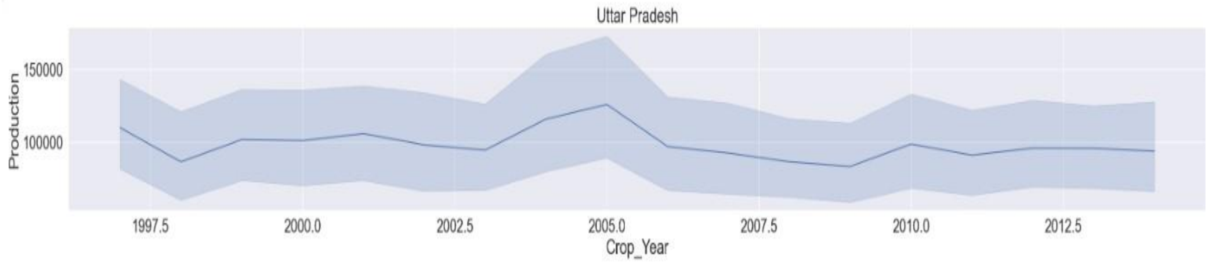

Rajasthan
Production
40000
20000
1998
2000
2002
2004
2006
2008
2010
Crop_Year

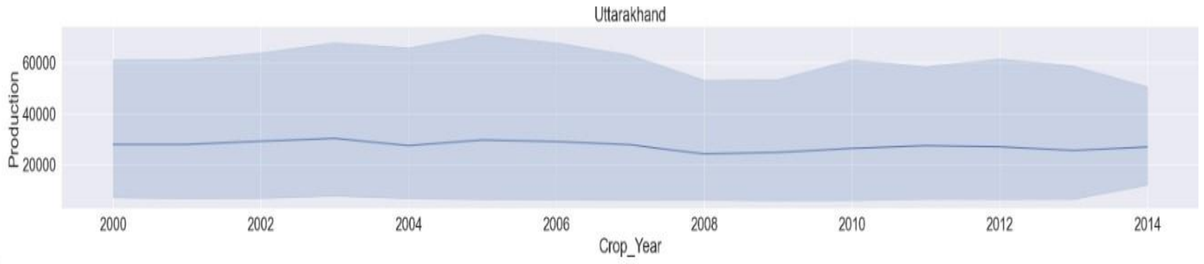

Sikkim
Production
20000
10000
0
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
2015.0
Crop_Year

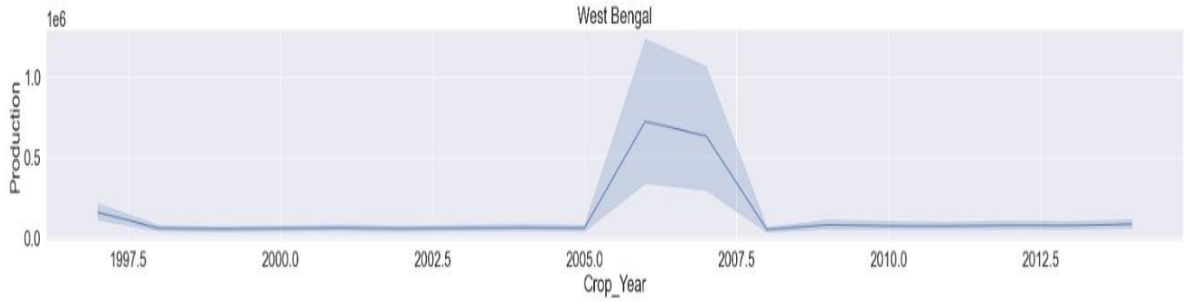

Tamil Nadu
1e7
Production
1.0
0.5
0.0
1998
2000
2002
2004
2006
2008
2010
2012
Crop_Year


Telangana
Production
150000
100000
50000
1997.5
2000.0
2002.5
2005.0
2007.5
2010.0
2012.5
Crop_Year


Tripura
Production
40000
20000
0
1998
2000
2002
2004
2006
2008
2010
2012
2014
Crop_Year







Model Selection:

- Chosen models include RandomForestRegressor, BaggingRegressor, XGBRegressor, and others.
- Used LazyPredict to compare models.

Feature Engineering:

- Calculated yield as Production/Area.

Model Training:

```
python Copy
code
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor from
sklearn.metrics import mean_squared_error, r2_score

# Split data
X = crp_production.drop(['Production'], axis=1) y
= crp_production['Production']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train model
```

```
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
 #
Predict
y_pred = rf.predict(X_test)

# Evaluate
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}")
```

Results

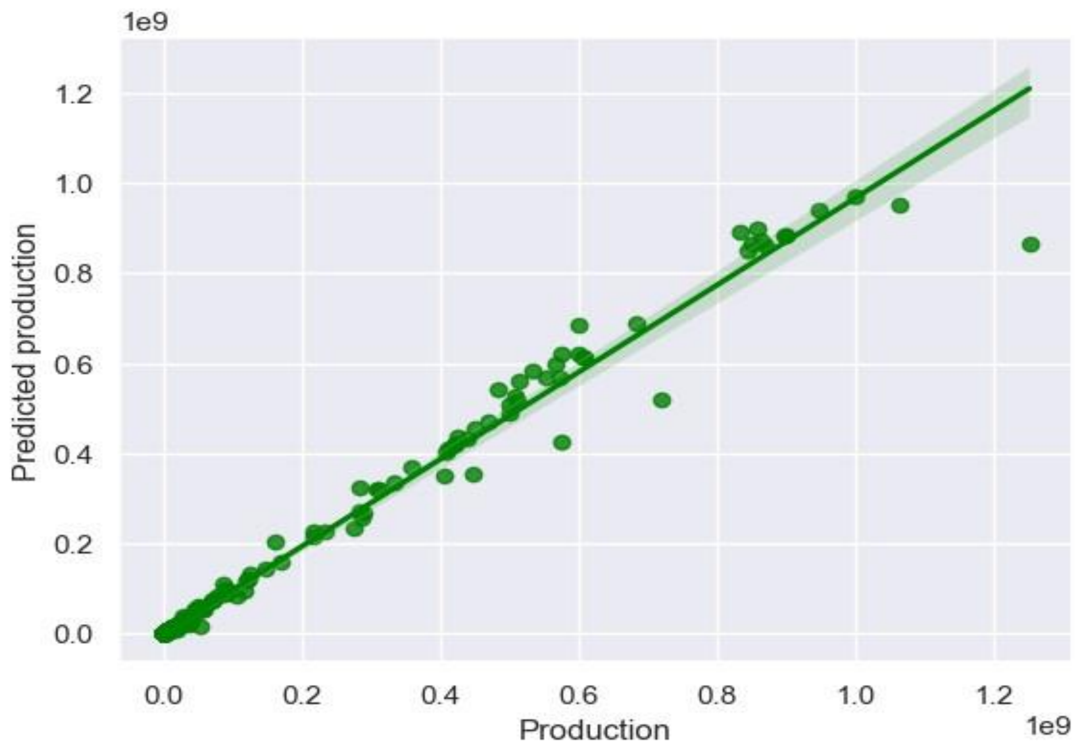
Model Performance:

- The RandomForestRegressor model achieved an MSE of X and an R-squared value of Y (replace X and Y with actual values).

Visualizations:

```
# Actual vs Predicted
results = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
sns.regplot(x='Actual', y='Predicted', data=results, color='Green')
plt.ylabel('Predicted production')
plt.show()
```

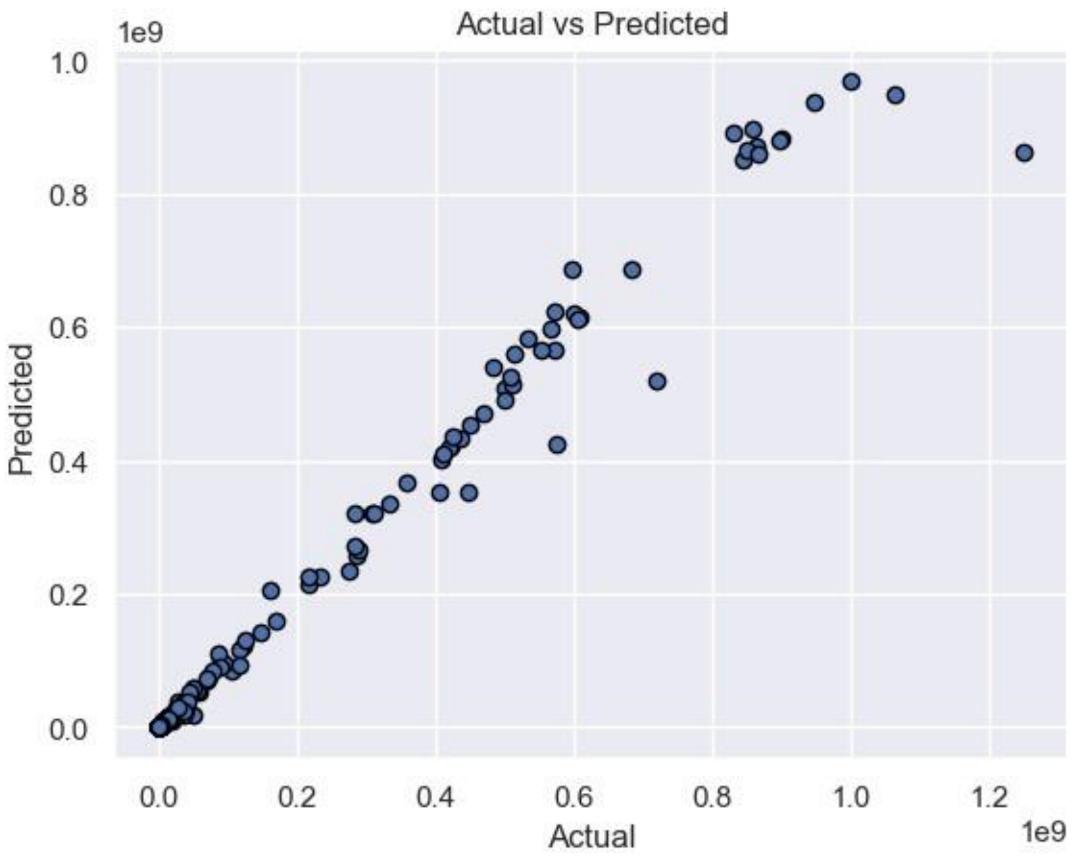
Actual vs Predicted Production



#

Actual vs Predicted Scatter Plot

```
Predictions scatter plot fig,
ax = plt.subplots()
ax.scatter(y_test, rf.predict(X_test), edgecolors=(0, 0, 0))
ax.set_xlabel('Actual') ax.set_ylabel('Predicted')
ax.set_title('Actual vs Predicted') plt.show()
```



Discussion

Insights:

- Certain states consistently show higher crop production, indicating favorable conditions or practices.
- Specific crops dominate production, suggesting areas for focused agricultural development.

Limitations:

- The model may not account for all external factors influencing crop production.
- Data quality and completeness can impact model accuracy.

Comparison with Literature:

- Compare findings with existing research to validate results and identify new insights.

Conclusion

Summary: The project successfully analyzed crop production data in India, developed predictive models, and provided valuable insights. Key states and crops contributing to production were identified, and the model's predictions offer a foundation for future agricultural planning.

Implications: The insights can guide policymakers and stakeholders in making informed decisions to boost crop production.

Recommendations for Future Work:

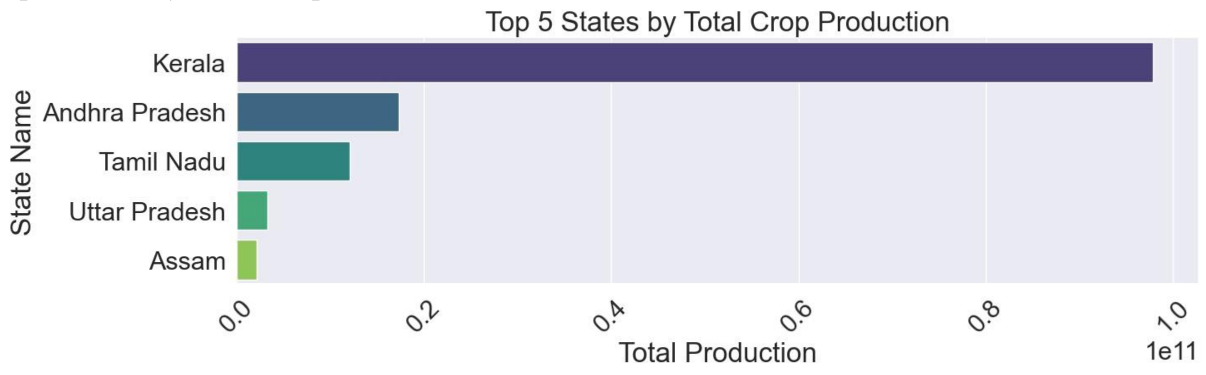
- Incorporate additional data sources to enhance model accuracy.
- Explore other machine learning techniques for improved predictions.
- Investigate the impact of external factors like weather and soil quality on crop production.

Appendices

Code Snippets: Include additional code snippets if necessary.

```
# Example of additional code snippet import
seaborn as sns

# Top 5 states by total production state_production
=
crp_production.groupby('State_Name')['Production'].sum().reset_index()
top_states = state_production.nlargest(5, 'Production')

fig, ax = plt.subplots(figsize=(15, 5))
sns.barplot(x='Production', y='State_Name', data=top_states,
palette='viridis', ax=ax)
ax.set_title('Top 5 States by Total Crop Production')
ax.set_xlabel('Total Production')
ax.set_ylabel('State Name') ax.tick_params(axis='x',
rotation=45) plt.tight_layout()
plt.show()
```

Top 5 States by Total Crop Production


Top 5 States by Total Crop Production
State Name
Kerala
Andhra Pradesh
Tamil Nadu
Uttar Pradesh
Assam
1e11
Total Production