Importing Libraries:

```
In [1]:
import pandas as pd
import numpy as np
```

Loading the dataset:

```
In [2]:
df=pd.read_csv(r"C:\Users\HP\Documents\DS INTERNSHIP\Crop Production data.csv")
df.head()
```

Out[2]:

	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

Analysing and Preprocessing:

```
In [3]:
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
```

Insights: 1. The dataset has 246091 records and 7 attributes. 2. There are 4 non numerical columns and 3 numerical columns.

```
In [4]:
df.isnull().sum()
```

Out[4]:

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

Insight: 1. Attribute production has 3730 null values.

```
In [4]:
df1=df.dropna()
```

Dropping the na value records since we cannot predict the production of those crops.

```
In [5]:
categorical_columns=[col for col in df.columns if df[col].dtype in ['O','Object']]
categorical_columns
```

Out[5]:

```
['State_Name', 'District_Name', 'Season', 'Crop']
```

```
In [6]:
numerical_columns=[col for col in df.columns if df[col].dtype not in ['O','Object']]
numerical_columns
```

Out[6]:

```
['Crop_Year', 'Area', 'Production']
```

```
In [7]:
for col in categorical_columns:
    print('COLUMN NAME:',col)
    print(df[col].value_counts())
    print("\n")
```

```
COLUMN NAME: State_Name
Uttar Pradesh               33189
Madhya Pradesh              22604
Karnataka                   21079
Bihar                       18874
Assam                       14622
Odisha                      13524
Tamil Nadu                  12266
Maharashtra                 12496
Rajasthan                   12066
Chhattisgarh                10368
West Bengal                  9597
Andhra Pradesh               9561
Gujarat                      8365
Telangana                    5591
Uttarakhand                  4825
Haryana                      4540
Kerala                       4003
Nagaland                     3904
Punjab                       3143
Meghalaya                    2867
Arunachal Pradesh            2545
Himachal Pradesh             2456
Jammu and Kashmir            1632
Tripura                      1412
Manipur                      1266
Jharkhand                    1266
Mizoram                       954
Puducherry                    872
Sikkim                        714
Dadra and Nagar Haveli        463
Goa                           207
Andaman and Nicobar Islands   201
Chandigarh                     89
Name: State_Name, dtype: int64


COLUMN NAME: District_Name
TUMKUR          931
BELGAUM         924
BIJAPUR         905
RAICHUR         895
BELLARY         887
                ...
HYDERABAD         8
KHOWAI            6
RAMGARH           6
KAMRAJ            1
MUMBAI            1
Name: District_Name, Length: 646, dtype: int64


COLUMN NAME: Season
Kharif     94283
Rabi       66160
Whole Year  56127
Summer      14811
Winter       6050
Autumn       4930
Name: Season, dtype: int64


COLUMN NAME: Crop
Rice                    15082
Maize                   13787
Moong(Green Gram)       10206
Urad                     9710
Sesamum                  8921
                        ...
Litchi                      6
Coffee                      6
Apple                       4
Peach                       4
Other Dry Fruit             1
Name: Crop, Length: 124, dtype: int64
```

```
In [33]:
df1['State_Name'].nunique()
```

Out[33]:

```
33
```

```
In [34]:
df1['District_Name'].nunique()
```

Out[34]:

```
646
```

```
In [35]:
df1['Crop_Year'].value_counts()
```

Out[35]:

```
2003    17139
2002    16536
2007    14269
2008    14230
2006    13976
2004    13858
2010    13793
2011    13790
2005    13753
2009    13563
2015    13539
2013    13475
2001    13293
2012    13184
1999    12441
1998    11262
2014    10815
1997     8899
2016      561
Name: Crop_Year, dtype: int64
```

Insights: 1. There are 33 unique values in the column State_Name. 2. There are 646 unique values in the column District_Name. 3. There are 6 unique seasons. 4. There are 124 different crops in production. 5. The data record lies between the years 1997 to 2015.# After Preprocessing, the dataset is exported as excel file to create dashboard in Tableau: filename=r'Crop Production Analysis in India_ETL.xlsx' df1.to_excel(filename) print('Dataframe has been exported as excel file successfully')

Converting all alphabetical columns to numerical columns for prediction:

```
In [36]:
import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import LabelEncoder
lb=LabelEncoder()
lb.fit(df1['District_Name'])
df1['District_Name']=lb.transform(df1['District_Name'])
```

```
In [37]:
df1.head()
```

Out[37]:

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production
0	Andaman and Nicobar Islands	427	2000	Kharif	Arecanut	1254.0	2000.0
1	Andaman and Nicobar Islands	427	2000	Kharif	Other Kharif pulses	2.0	1.0
2	Andaman and Nicobar Islands	427	2000	Kharif	Rice	102.0	321.0
3	Andaman and Nicobar Islands	427	2000	Whole Year	Banana	176.0	641.0
4	Andaman and Nicobar Islands	427	2000	Whole Year	Cashewnut	720.0	165.0

```
In [38]:
df1['State_Name'].value_counts()
```

Out[38]:

```
Uttar Pradesh               33189
Madhya Pradesh              22604
Karnataka                   21079
Bihar                       18874
Assam                       14622
Odisha                      13524
Tamil Nadu                  12266
Maharashtra                 12496
Rajasthan                   12066
Chhattisgarh                10368
West Bengal                  9597
Andhra Pradesh               9561
Gujarat                      8365
Telangana                    5591
Uttarakhand                  4825
Haryana                      4540
Kerala                       4003
Nagaland                     3904
Punjab                       3143
Meghalaya                    2867
Arunachal Pradesh            2545
Himachal Pradesh             2456
Jammu and Kashmir            1632
Tripura                      1412
Manipur                      1266
Jharkhand                    1266
Mizoram                       954
Puducherry                    872
Sikkim                        714
Dadra and Nagar Haveli        463
Goa                           207
Andaman and Nicobar Islands   201
Chandigarh                     89
Name: State_Name, dtype: int64
```

#Based on the number of frequencies, replacing numerical values to the values in the State_Name column:

```
In [39]:
import warnings
warnings.filterwarnings('ignore')

df1.replace({'Uttar Pradesh':'Madhya Pradesh':'Karnataka':'Bihar':'Assam':'Odisha':'Tamil Nadu':'Maharashtra':'
df1.replace({'West Bengal':'Andhra Pradesh':'Gujarat':'Telangana':'Uttarakhand':'Haryana':'Kerala':'Nagaland',
df1.replace({'Meghalaya':'Arunachal Pradesh':'Himachal Pradesh':'Jammu and Kashmir':'Tripura':'Manipur':'Shark
df1.replace({'Mizoram':'Puducherry':'Sikkim':'Dadra and Nagar Haveli':'Goa':'Andaman and Nicobar Islands':'Chan
```

```
In [40]:
import warnings
warnings.filterwarnings('ignore')

df1.drop(['Crop_Year'],axis=1,inplace=True)
df1.head()
```

Out[40]:

	State_Name	District_Name	Season	Crop	Area	Production
0	1	427	Kharif	Arecanut	1254.0	2000.0
1	1	427	Kharif	Other Kharif pulses	2.0	1.0
2	1	427	Kharif	Rice	102.0	321.0
3	1	427	Whole Year	Banana	176.0	641.0
4	1	427	Whole Year	Cashewnut	720.0	165.0

```
In [41]:
dummies= pd.get_dummies(df1['Season','Crop']],drop_first=True)
dummies.head()
```

Out[41]:

	Season_Kharif	Season_Rabi	Season_Summer	Season_Whole Year	Season_Winter	Crop_Arecanut (Processed)	Crop_Arecanut	Crop_Arhar/Tur	Crop_Ash Gourd	Crop_
0	1	0	0	0	0	0	1	0	0	
1	1	0	0	0	0	0	0	0	0	
2	1	0	0	0	0	0	0	0	0	
3	0	0	0	1	0	0	0	0	0	
4	0	0	0	1	0	0	0	0	0	

5 rows × 128 columns

```
In [42]:
df1.drop(['Season','Crop'],axis=1,inplace=True)
df2=pd.concat([df1,dummies],axis=1)
df2.head()
```

Out[42]:

	State_Name	District_Name	Area	Production	Season_Kharif	Season_Rabi	Season_Summer	Season_Whole Year	Season_Winter	Crop_Arecanut (Processed)
0	1	427	1254.0	2000.0	1	0	0	0	0	0
1	1	427	2.0	1.0	1	0	0	0	0	0
2	1	427	102.0	321.0	1	0	0	0	0	0
3	1	427	176.0	641.0	0	0	0	1	0	0
4	1	427	720.0	165.0	0	0	0	1	0	0

5 rows × 132 columns

```
In [43]:
from sklearn.preprocessing import MinMaxScaler
mis_sc=MinMaxScaler()
df2['Area']=mis(df2['Area'])
df2.head()
```

Out[43]:

	State_Name	District_Name	Area	Production	Season_Kharif	Season_Rabi	Season_Summer	Season_Whole Year	Season_Winter	Crop_Arecanut (Processed)	
0	1	427	0.1	2000.0	1	0	0	0	0	0	...
1	1	427	0.1	1.0	1	0	0	0	0	0	...
2	1	427	0.1	321.0	1	0	0	0	0	0	...
3	1	427	0.1	641.0	0	0	0	1	0	0	...
4	1	427	0.1	165.0	0	0	0	1	0	0	...

5 rows × 132 columns

Prediction on Crop Production:

```
In [44]:
X=df2.drop('Production',axis=1)
y=df2.Production
```

```
In [45]:
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=0)
```

```
In [22]:
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train)
y_pred=lr.predict(X_test)
```

```
In [24]:
lr.score(X_test,y_test)
```

Out[24]:

```
0.09985713881106324
```

Linear Regression gives a minimum accuracy. Therefore prediction is made using Random Forest Regressor.

```
In [45]:
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor(n_estimators=100,max_depth=10,n_jobs=-1,random_state=42)
rf.fit(X_train,y_train)
y_pred=rf.predict(X_test)
```

```
In [47]:
rf.score(X_test,y_test)
```

Out[58]:

```
0.8301047747467512
```

```
In [ ]:
The Random Forest Regressor provides prediction on Crop Production with an accuracy of 83%.
```