

```
In [1]: # importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

In [2]: # loading the dataset
crop_data=pd.read_csv("crop_production_indian.csv")
crop_data

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
...	...	...	...	...	...	...	...
246086	West Bengal	PURULIA	2014	Summer	Rice	306.0	801.0
246087	West Bengal	PURULIA	2014	Summer	Sesamum	627.0	463.0
246088	West Bengal	PURULIA	2014	Whole Year	Sugarcane	324.0	16250.0
246089	West Bengal	PURULIA	2014	Winter	Rice	279151.0	597899.0
246090	West Bengal	PURULIA	2014	Winter	Sesamum	175.0	88.0

246091 rows × 7 columns

```
In [3]: crop_data.shape
#rows X columns

Out[3]: (246091, 7)

In [4]: # statistical inference of the dataset
crop_data.columns

Out[4]: Index(['State_Name', 'District_Name', 'Crop_Year', 'Season', 'Crop', 'Area',
        'Production'],
        dtype='object')

In [5]: # statistical inference of the dataset
crop_data.describe()

Out[5]:
```

	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%	2005.000000	5.820000e+02	7.290000e+02
75%	2010.000000	4.392000e+03	7.023000e+03
max	2015.000000	8.580100e+06	1.250800e+09

```
In [6]: # Checking missing values of the dataset in each column
crop_data.isnull().sum()

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

In [7]: # Dropping missing values
crop_data = crop_data.dropna()
crop_data

Out[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
...	...	...	...	...	...	...	...
246086	West Bengal	PURULIA	2014	Summer	Rice	306.0	801.0
246087	West Bengal	PURULIA	2014	Summer	Sesamum	627.0	463.0
246088	West Bengal	PURULIA	2014	Whole Year	Sugarcane	324.0	16250.0
246089	West Bengal	PURULIA	2014	Winter	Rice	279151.0	597899.0
246090	West Bengal	PURULIA	2014	Winter	Sesamum	175.0	88.0

242361 rows × 7 columns

```
In [8]: crop_data.isnull().values.any()

Out[8]: False

In [9]: #displaying State Names present in the dataset
crop_data.State_Name.unique()

Out[9]: array(['Andaman and Nicobar Islands', 'Andhra Pradesh',
        'Arunachal Pradesh', 'Assam', 'Bihar', 'Chandigarh',
        'Chhattisgarh', 'Dadra and Nagar Haveli', 'Goa', 'Gujarat',
        'Haryana', 'Himachal Pradesh', 'Jammu and Kashmir', 'Jharkhand',
        'Karnataka', 'Kerala', 'Madhya Pradesh', 'Maharashtra', 'Manipur',
        'Meghalaya', 'Mizoram', 'Nagaland', 'Odisha', 'Puducherry',
        'Punjab', 'Rajasthan', 'Sikkim', 'Tamil Nadu', 'Telangana',
        'Tripura', 'Uttar Pradesh', 'Uttarakhand', 'West Bengal'],
        dtype=object)

In [10]: crop_data.isnull().values.any()

Out[10]: False

In [11]: #Adding a new column Yield which indicates Production per unit Area.
crop_data['Yield'] = (crop_data['Production'] / crop_data['Area'])
crop_data.head(10)

Out[11]:
```

	State_Name	District_Name	Crop_Year	Season	Crop	Area	Production	Yield
0	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Arecanut	1254.0	2000.0	1.594996
1	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Other Kharif pulses	2.0	1.0	0.500000
2	Andaman and Nicobar Islands	NICOBARS	2000	Kharif	Rice	102.0	321.0	3.147059
3	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Banana	176.0	641.0	3.642045
4	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Cashewnut	720.0	165.0	0.229167
5	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Coconut	18168.0	65100000.0	3583.223250
6	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Dry ginger	36.0	100.0	2.777778
7	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sugarcane	1.0	2.0	2.000000
8	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Sweet potato	5.0	15.0	3.000000
9	Andaman and Nicobar Islands	NICOBARS	2000	Whole Year	Taploca	40.0	169.0	4.225000

```
In [12]: data = crop_data.drop(['State_Name'], axis = 1)

In [13]: data.corr()

Out[13]:
```

	Crop_Year	Area	Production	Yield
Crop_Year	1.000000	-0.025305	0.006989	0.013499
Area	-0.025305	1.000000	0.040587	0.001822
Production	0.006989	0.040587	1.000000	0.330961
Yield	0.013499	0.001822	0.330961	1.000000

```
In [14]: #correlation
sns.heatmap(data.corr(), annot=True)
plt.title('Correlation Matrix')

Out[14]: Text(0.5, 1.0, 'Correlation Matrix')
```

```
In [15]: dummy = pd.get_dummies(data)
dummy

Out[15]:
```

	Crop_Year	Area	Production	Yield	District_Name_24 PARAGANAS NORTH	District_Name_24 PARAGANAS SOUTH	District_Name_ADILABAD	District_Name_AGAR MALWA	District_Name_AGRA	District_Name_AHMADABAD	...	Crop_Turmeric	Crop_T
0	2000	1254.0	2000.0	1.594996	0	0	0	0	0	0	...	0	0
1	2000	2.0	1.0	0.500000	0	0	0	0	0	0	...	0	0
2	2000	102.0	321.0	3.147059	0	0	0	0	0	0	...	0	0
3	2000	176.0	641.0	3.642045	0	0	0	0	0	0	...	0	0
4	2000	720.0	165.0	0.229167	0	0	0	0	0	0	...	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
246086	2014	306.0	801.0	2.617647	0	0	0	0	0	0	...	0	0
246087	2014	627.0	463.0	0.738437	0	0	0	0	0	0	...	0	0
246088	2014	324.0	16250.0	50.154321	0	0	0	0	0	0	...	0	0
246089	2014	279151.0	597899.0	2.141848	0	0	0	0	0	0	...	0	0
246090	2014	175.0	88.0	0.502857	0	0	0	0	0	0	...	0	0

242361 rows × 780 columns

```
In [16]: from sklearn.model_selection import train_test_split

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

# Splitting data set - 25% test dataset and 75%

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)

x_train : (181770, 778)
x_test : (60591, 778)
y_train : (181770,)
y_test : (60591,)

In [17]: print(x_train)
print(y_train)

Crop_Year      Area  District_Name_24 PARAGANAS NORTH \
201072      2013    16.0                0
191897      1998   5400.0                0
43814       2000   2968.0                0
32815       2013    211.0                0
62249       2006   1700.0                0
...
236131      2000    207.0                0
127145      2007    39.0                0
20536       2005    43.0                0
18709       2011   2489.0                0
35767       1999    67.0                0

District_Name_24 PARAGANAS SOUTH District_Name_ADILABAD \
201072                0                0
191897                0                0
43814                 0                0
32815                 0                0
62249                 0                0
...
236131                0                0
127145                0                0
20536                 0                0
18709                 0                0
35767                 0                0

District_Name_AGAR MALWA District_Name_AGRA District_Name_AHMADABAD \
201072                0                0                0
191897                0                0                0
43814                 0                0                0
32815                 0                0                0
62249                 0                0                0
...
236131                0                0                0
127145                0                0                0
20536                 0                0                0
18709                 0                0                0
35767                 0                0                0

District_Name_AHMEDNAGAR District_Name_AIZAWL ... Crop_Turmeric \
201072                0                0 ...                0
191897                0                0 ...                0
43814                 0                0 ...                0
32815                 0                0 ...                0
62249                 0                0 ...                0
...
236131                0                0 ...                0
127145                0                0 ...                0
20536                 0                0 ...                0
18709                 0                0 ...                0
35767                 0                0 ...                0

Crop_Turnip  Crop_Urad  Crop_Varagu  Crop_Water Melon  Crop_Wheat \
201072        0         0            0            0      0
191897        0         0            0            0      0
43814         0         0            0            0      0
32815         0         0            0            0      0
62249         0         0            0            0      0
...
236131        0         0            0            0      0
127145        0         0            0            0      0
20536         0         0            0            0      0
18709         0         0            0            0      0
35767         0         0            0            0      0

Crop_Yam  Crop_other fibres  Crop_other misc. pulses \
201072        0             0                0
191897        0             0                0
43814         0             0                0
32815         0             0                0
62249         0             0                0
...
127145        0             0                0
20536         0             0                0
18709         0             0                0
35767         0             0                0

Crop_other oilseeds
201072        0
191897        0
43814         0
32815         0
62249         0
...
236131        0
127145        0
20536         0
18709         0
35767         0

[181770 rows x 778 columns]
201072      11.0
191897     2000.0
43814      2555.0
32815      175.0
62249     1400.0
...
236131      120.0
127145      44.0
20536       27.0
18709     4779.0
35767      81.0
Name: Production, Length: 181770, dtype: float64

In [18]: #LINEAR REGRESSION

#Training the Simple Linear Regression model .

from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(x_train,y_train)

Out[18]: LinearRegression()

In [19]: #Predicting the test Results
lr_predict = model.predict(x_test)

Out[19]: array([ -181489.171875 ,    489530.109375 , -1267951.8828125, ...,
        140833.75      , -1730617.9921875,    363764.015625 ])
```

```
In [20]: model.score(x_test,y_test)

Out[20]: -12663.421745320156

In [21]: from sklearn.metrics import r2_score
r = r2_score(y_test,lr_predict)
print("R2 score :",r)

R2 score : -12663.421745320156

In [22]: # Clearly, the dataset is not good for linear regression.
#
# <b> Assumptions of Linear Regression </b>
# <ol>
# <li> Linearity.</li>
# <li> Homoscedasticity </li>
# <li> Multivariate normality </li>
# <li> Lack of multicollinearity </li>
#
#
# # R2 score: This is pronounced as R-squared, and this score refers to the coefficient of determination.
# # This tells us how well the unknown samples will be predicted by our model.
plt.scatter(y_test,lr_predict)
plt.xlabel('Actual')
plt.ylabel('Predicted')
plt.title('Linear Regression')

Out[22]: Text(0.5, 1.0, 'Linear Regression')
```

```
In [23]: #RANDOMFOREST REGRESSION
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor(n_estimators = 11)
model.fit(x_train,y_train)
rf_predict = model.predict(x_test)
rf_predict

Out[23]: array([3823.74545455,    654.81818182, 2246.90909091, ...,   196.3
        9516.         ,   100.         ])
```

```
In [24]: model.score(x_test,y_test)

Out[24]: 0.9683643479130472

In [25]: from sklearn.metrics import r2_score
r1 = r2_score(y_test,rf_predict)
print("accuracy: ",r1)

accuracy:  0.9683643479130472

In [26]: #XGBOOSTING ALGORITHM
import xgboost as xgb
xgbr = xgb.XGBRegressor(verbose=0)
xgbr.fit(x_train,y_train)

Out[26]: XGBRegressor(base_score=0.5, boosters='gbtree', callbacks=None,
        colsample_bylevel=1, colsample_bynode=1, colsample_bynode=1,
        early_stopping_rounds=None, enable_categorical=False,
        eval_metric=None, feature_types=None, gamma=0, gpu_id=-1,
        grow_policy='depthwise', importance_type=None,
        interaction_constraints='', learning_rate=0.300000012, max_bin=256,
        max_cat_threshold=64, max_cat_to_onehot=4, max_delta_step=0,
        max_depth=6, max_leaves=0, min_child_weight=1, missing=nan,
        monotone_constraints=('',), n_estimators=100, n_jobs=0,
        num_parallel_tree=1, predictor='auto', random_state=0, ...)

In [28]: preds = xgbr.predict(x_test)

r2_score(y_test,preds)

Out[28]: 0.9745502426880536

In [ ]:
```