In [3]:
```
import pandas as pd
```

In [4]:
```
labour  = pd.read_csv('C:\\Users\\mayan\\Downloads\\flabourdataV.csv')
```

1. Display Top 5 Rows of The Dataset

In [5]:
```
labour.head()
```

Out[5]:

	Year	China	East Asia & Pacific	European Union	United Kingdom	India	Middle East & North Africa	South Asia	Sub-Saharan Africa	United States	Latin America & the Caribbean	World
0	1991	72.778000	66.142748	46.098997	52.318001	30.452999	17.641757	29.557188	62.961940	56.428001	41.856903	51.230600
1	1992	72.531998	66.044488	46.077873	52.470001	30.493000	17.746870	29.511810	62.978518	56.942001	42.699031	51.255409
2	1993	72.285004	65.750821	45.903345	52.542999	30.570000	17.708256	29.613806	63.019299	57.057999	43.597722	51.075818
3	1994	72.037003	65.716154	45.937356	52.589001	30.691999	18.056743	29.739605	63.069545	57.945999	44.462199	51.176450
4	1995	71.788002	65.560973	45.838367	52.562000	30.656000	18.046995	29.507352	63.205321	58.141998	45.324111	51.080439

2. Check Last 5 Rows of The Dataset

In [6]:
```
labour.tail()
```

Out[6]:

	Year	China	East Asia & Pacific	European Union	United Kingdom	India	Middle East & North Africa	South Asia	Sub-Saharan Africa	United States	Latin America & the Caribbean	World
26	2017	63.292000	60.095748	50.948850	57.790001	20.934000	20.525804	23.760232	61.480552	56.183998	50.857551	47.736266
27	2018	63.113998	60.194908	51.045100	58.026001	20.525999	19.775121	23.359795	61.477325	56.237000	51.136339	47.642200
28	2019	63.014000	60.251577	51.248444	58.460999	21.179001	19.418452	23.860563	61.504738	56.596001	51.339354	47.761914
29	2020	61.818001	59.120181	50.724894	58.603001	18.603001	18.489932	21.437748	59.663110	55.389999	46.293968	45.922901
30	2021	61.612000	59.043895	51.322057	58.043999	19.233000	18.606198	21.990497	60.219909	55.227001	48.745922	46.296757

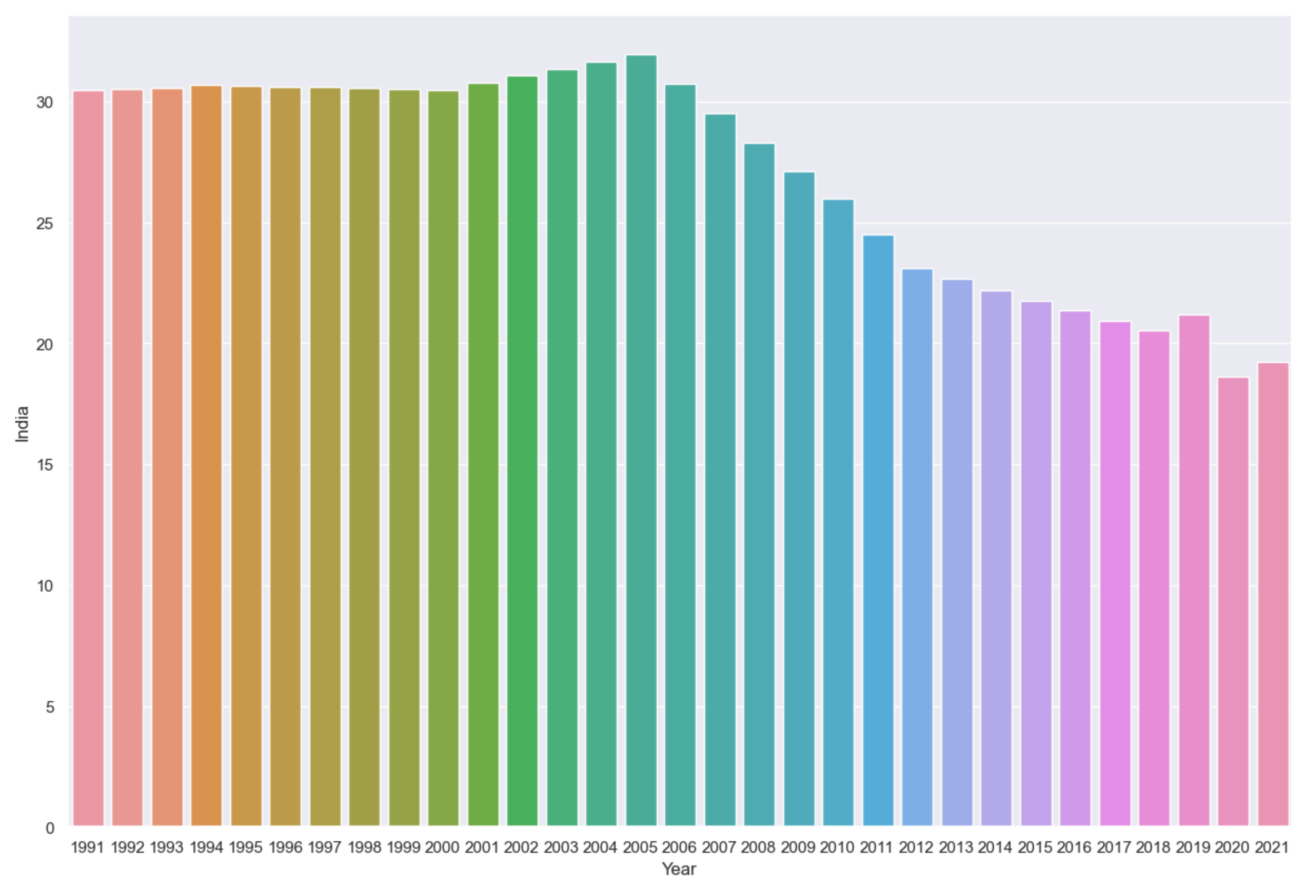
In [5]:
```
labour.shape
```

Out[5]:
```
(31, 12)
```

In [6]:
```
import seaborn as sns
import numpy as np
from matplotlib  import pyplot as plt
```

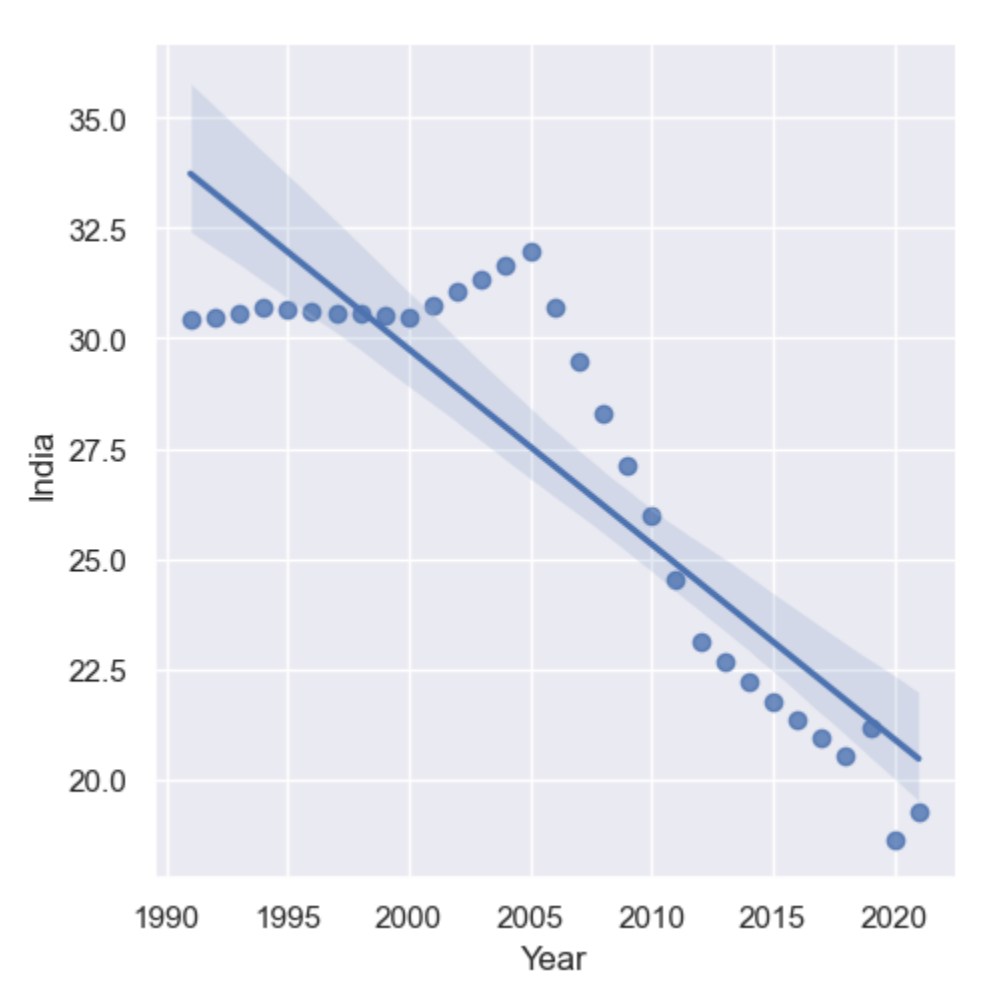
In [7]:
```
sns.set(rc = {'figure.figsize':(15,10)})
sns.barplot(x = "Year",y ="India",data = labour)
```

Out[7]:
```
<AxesSubplot:xlabel='Year', ylabel='India'>
```



In [8]:
```
sns.lmplot(x = "Year",y ="India",data = labour)
```

Out[8]:
```
<seaborn.axisgrid.FacetGrid at 0x1d31b632880>
```



In [9]:
```
print("Number of Rows",labour.shape[0])
print("Number of Columns",labour.shape[1])
```

```
Number of Rows 31
Number of Columns 12
```

In [10]:
```
labour.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 12 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   Year                           31 non-null     int64  
 1   China                          31 non-null     float64
 2   East Asia & Pacific            31 non-null     float64
 3   European Union                 31 non-null     float64
 4   United Kingdom                 31 non-null     float64
 5   India                          31 non-null     float64
 6   Middle East & North Africa     31 non-null     float64
 7   South Asia                     31 non-null     float64
 8   Sub-Saharan Africa             31 non-null     float64
 9   United States                  31 non-null     float64
 10  Latin America & the Caribbean  31 non-null     float64
 11  World                          31 non-null     float64
dtypes: float64(11), int64(1)
memory usage: 3.0 KB
```

In [11]:
```
labour.isnull().sum()
```

Out[11]:
```
Year                             0
China                            0
East Asia & Pacific              0
European Union                   0
United Kingdom                   0
India                            0
Middle East & North Africa       0
South Asia                       0
Sub-Saharan Africa               0
United States                    0
Latin America & the Caribbean    0
World                            0
dtype: int64
```

Get Overall Statistics About The Dataset

In [12]:
```
labour.describe(include='all')
```

Out[12]:

	Year	China	East Asia & Pacific	European Union	United Kingdom	India	Middle East & North Africa	South Asia	Sub-Saharan Africa	United States	Latin America & the Caribbean	World
count	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000	31.000000
mean	2006.000000	67.065129	62.425145	48.463674	55.341968	27.098548	19.172954	27.506486	62.738803	57.508194	48.301182	49.500409
std	9.092121	3.793647	2.383392	1.970094	1.979694	4.513799	0.943396	2.981553	1.173825	1.216463	2.744605	1.598724
min	1991.000000	61.612000	59.043895	45.831440	52.318001	18.603001	17.641757	21.437748	59.663110	55.227001	41.856903	45.922901
25%	1998.500000	63.567501	60.303610	46.588138	53.610001	22.429000	18.290002	24.434873	61.551201	56.371500	46.471089	47.964452
50%	2006.000000	66.475998	61.581240	48.537177	55.674000	30.452999	19.418452	29.507352	63.069545	57.945999	49.473216	50.116997
75%	2013.500000	70.905499	64.854066	50.400699	56.884001	30.638500	19.904982	29.739036	63.636655	58.484001	50.504669	50.942678
max	2021.000000	72.778000	66.142748	51.322057	58.603001	31.955000	20.865668	31.008938	64.094025	59.112000	51.339354	51.255409

In [ ]:

In [13]:
```
#Using Linear Regression Algorithm
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [15]:
```
y = labour[['India']]
```

In [16]:
```
x = labour[['Year']]
```

In [17]:
```
# test size = 0.20 we are storing 20% data in test size
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [18]:
```
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.tree import DecisionTreeRegressor
```

In [215…
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [216…
```
x_train.head()
```

Out[216]:

	Year
27	2018
5	1996
10	2001
4	1995
21	2012

In [217…
```
x_test.head()
```

Out[217]:

	Year
28	2019
25	2016
15	2006
1	1992
6	1997

In [218…
```
y_train.head()
```

Out[218]:

	India
27	20.525999
5	30.621000
10	30.771000
4	30.656000
21	23.099001

In [219…
```
y_test.head()
```

Out[219]:

	India
28	21.179001
25	21.351000
15	30.712000
1	30.493000
6	30.584000

PREDICTION BY LINEAR REGRESSION

In [220…
```
from sklearn.linear_model import LinearRegression
import sklearn.metrics as sm
```

In [221…
```
x_train,x_test,y_train,y_Test = train_test_split(x,y,test_size=0.3)
```

In [222…
```
lr = LinearRegression()
```

In [223…
```
lr.fit(x_train,y_train)
```

Out[223]:
```
LinearRegression()
```

In [224…
```
lr.predict(x_test)
```

Out[224]:
```
array([[21.1528477 ],
       [30.09190246],
       [32.32666615],
       [34.1144771 ],
       [30.98580794],
       [31.43276067],
       [25.62237508],
       [22.04675318],
       [20.70589496],
       [26.51628056]])
```

In [225…
```
y_test.head()
```

Out[225]:

	India
28	21.179001
25	21.351000
15	30.712000
1	30.493000
6	30.584000

In [226…
```
y_pred = lr.predict(x_test)
```

In [227…
```
y_pred[0:5]
```

Out[227]:
```
array([[21.1528477 ],
       [30.09190246],
       [32.32666615],
       [34.1144771 ],
       [30.98580794]])
```

In [228…
```
from sklearn.metrics import mean_squared_error
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
```

In [229…
```
model = LinearRegression()
```

In [230…
```
mean_squared_error(y_test,y_pred)
```

Out[230]:
```
24.94370769204662
```

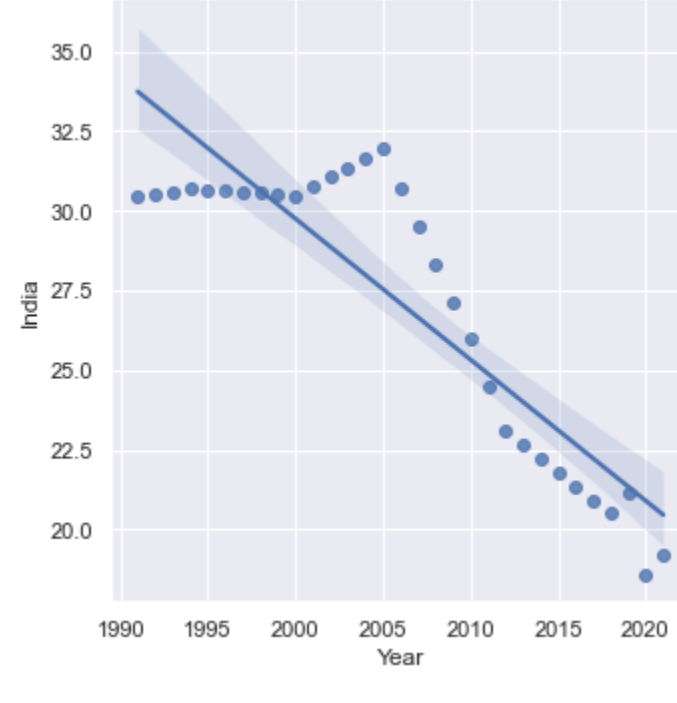
In [231…
```
print("R2 score =", round(sm.r2_score(y_test, y_pred), 2))
```

```
R2 score = -0.4
```

In [151…
```
# DONT IMPLEMENT THIS
import seaborn as sns
labour.head()

sns.lmplot(x ='Year', y ='India', data = labour)
# putting labels
```

Out[151]:
```
<seaborn.axisgrid.FacetGrid at 0x251f88fc760>
```



In [54]:
```
 import statsmodels.api as sm
```

In [55]:
```
x = sm.add_constant(x)
```

In [56]:
```
model = sm.OLS(y, x)
```

In [57]:
```
results = model.fit()
```

In [58]:
```
 print(results.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:                  India   R-squared:                       0.795
Model:                            OLS   Adj. R-squared:                  0.787
Method:                 Least Squares   F-statistic:                     112.1
Date:                Sun, 04 Dec 2022   Prob (F-statistic):           1.77e-11
Time:                        13:11:05   Log-Likelihood:                -65.674
No. Observations:                  31   AIC:                             135.3
Df Residuals:                      29   BIC:                             138.2
Df Model:                           1                                         
Covariance Type:            nonrobust                                         
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const        914.7818     83.831     10.912      0.000     743.327    1086.236
Year          -0.4425      0.042    -10.589      0.000      -0.528      -0.357
==============================================================================
Omnibus:                        2.682   Durbin-Watson:                   0.129
Prob(Omnibus):                  0.262   Jarque-Bera (JB):                2.355
Skew:                           0.589   Prob(JB):                        0.308
Kurtosis:                       2.339   Cond. No.                     4.50e+05
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 4.5e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

END OF LINEAR REGRESSION

Using decison tree Algorithm

In [19]:
```
from sklearn.model_selection import train_test_split
```

In [20]:
```
x_train,x_test,y_train,y_Test = train_test_split(x,y,test_size=0.3)
```

In [21]:
```
from sklearn.tree import DecisionTreeRegressor
```

In [22]:
```
dtr = DecisionTreeRegressor()
```

In [23]:
```
dtr.fit(x_train,y_train)
```

Out[23]:
```
DecisionTreeRegressor()
```

In [24]:
```
y_predc =dtr.predict(x_test)
```

In [25]:
```
y_test.head()
```

Out[25]:

	India
29	18.603001
9	30.476999
5	30.621000
26	20.934000
19	25.965000

In [26]:
```
y_predc[0:5]
```

Out[26]:
```
array([30.51199913, 21.17900085, 31.06500053, 30.69199944, 27.11400032])
```

In [27]:
```
mean_squared_error(y_test,y_predc)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_19872\244100754.py in <module>
----> 1 mean_squared_error(y_test,y_predc)

NameError: name 'mean_squared_error' is not defined
```

In [169…
```
print(dtr.score(y_test, y_pred))
```

```
-0.4631404090918345
```

In [ ]:

In [ ]:

Using Random Forest Algorithm

In [170…
```
y = labour[['India']]
```

In [171…
```
x = labour[['Year']]
```

In [172…
```
from sklearn.model_selection import train_test_split
```

In [173…
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3)
```

In [174…
```
from sklearn.ensemble import RandomForestRegressor
```

In [175…
```
rfg  = RandomForestRegressor()
```

In [176…
```
rfg.fit(x_train,y_train)
```

```
<ipython-input-176-58d3d03db7b6>:1: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example usin
g ravel().
  rfg.fit(x_train,y_train)
```

Out[176]:
```
RandomForestRegressor()
```

In [177…
```
y_pred = rfg.predict(x_test)
```

In [178…
```
y_test.head(),y_pred[0:5]
```

Out[178]:
```
(        India
 21  23.099001
 2   30.570000
 27  20.525999
 26  20.934000
 10  30.771000,
 array([24.05380026, 30.62491944, 21.23738054, 21.45966984, 30.53649929]))
```

In [179…
```
mean_squared_error(y_test,y_pred)
```

Out[179]:
```
0.6475582234215433
```

In [188…
```
from sklearn import metrics
```

In [189…
```
rfg.score(y_test, y_pred)
```

Out[189]:
```
-0.8936081407489771
```

In [ ]:

In [6]:
```
pip install pandoc
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting pandoc
  Downloading pandoc-2.3.tar.gz (33 kB)
  Preparing metadata (setup.py): started
  Preparing metadata (setup.py): finished with status 'done'
Collecting plumbum
  Downloading plumbum-1.8.1-py3-none-any.whl (126 kB)
     -------------------------------------- 126.7/126.7 kB 1.1 MB/s eta 0:00:00
Collecting ply
  Downloading ply-3.11-py2.py3-none-any.whl (49 kB)
     ---------------------------------------- 49.6/49.6 kB 93.2 kB/s eta 0:00:00
Requirement already satisfied: pywin32 in c:\programdata\anaconda3\lib\site-packages (from plumbum->pandoc) (302)
Building wheels for collected packages: pandoc
  Building wheel for pandoc (setup.py): started
  Building wheel for pandoc (setup.py): finished with status 'done'
  Created wheel for pandoc: filename=pandoc-2.3-py3-none-any.whl size=33263 sha256=90d380e32357f2ce8b79ceae0cd7a787a07028aac0bd5b38e8d559fc74507c99
  Stored in directory: c:\users\mayan\appdata\local\pip\cache\wheels\69\e6\a1\1daa96d919c9e09a71473649b717b8da286f3f8d7719d1cfc5
Successfully built pandoc
Installing collected packages: ply, plumbum, pandoc
Successfully installed pandoc-2.3 plumbum-1.8.1 ply-3.11
Note: you may need to restart the kernel to use updated packages.
```

In [5]:

```
  File "C:\Users\mayan\AppData\Local\Temp\ipykernel_10688\888248936.py", line 1
    winget install pandoc
           ^
SyntaxError: invalid syntax
```

In [ ]: