Members: GAURAV | DHAVAL | KRISH | SIDDHANT Prediction for number of reverse vending machines

```
import pandas as pd import numpy as np import
matplotlib.pyplot as plt from
sklearn.feature_selection import SelectKBest from
sklearn.feature_selection import chi2 import seaborn
as sns from sklearn import model_selection from
sklearn.linear_model import LinearRegression from
sklearn.model_selection import train_test_split


total_waste=pd.read_csv("final dataset 1.csv")
total_waste
```

	Sr.no	state	2011	2012	2013	2014	2015	2016	2017	
0	1	Andaman & Nicobar	2675.00	2555.00	2123.000	2737.00	1583.00	567.0000	983.000	
1	2	Andhra Pradesh	28888.00	243820.00	243820.000	128480.00	128480.00	82863.0000	74588.000	
2	3	Arunachal Pradesh	453.00	673.00	863.000	1273.00	1454.00	3836.0000	6000.000	
3	4	Assam	1226.00	1116.00	781.000	1363.00	24010.00	24030.0000	28153.000	
4	5	Bihar	1527.00	1673.00	1723.000	1893.00	2314.00	2280.0000	2280.000	
5	6	Chandigarh	5548.00	4964.00	4818.000	8992.00	13167.00	21516.7500	12775.000	
6	7	Chhattisgarh	4678.00	5840.00	6123.000	6345.00	6897.00	7300.0000	6650.000	
7	8	Daman	1274.00	1347.00	14736.000	1573.00	1637.00	1733.0000	1836.000	
8	9	Delhi	261234.00	251850.00	23456.000	24567.00	128649.00	232732.0000	228771.000	2
9	10	Goa	415.00	1642.50	117.730	104.00	106.00	28273.0000	26242.000	
10	11	Gujarat	23666.00	251796.65	251796.650	265568.20	269294.88	271092.0000	269808.000	3
11	12	Haryana	52883.00	55480.00	50332.690	44536.00	32452.00	23369.0900	46353.000	
12	13	Himachal Pradesh	202.67	106.72	2326.800	1004.00	1577.00	2525.0000	2677.000	
13	14	Jammu & Kashmir	10273.00	11748.00	7832.000	4164.45	6243.20	4645.0000	27870.000	
14	15	Jharkhand	16691.05	81030.00	75817.800	35262.36	35853.52	36772.0000	43644.000	
15	16	Karnataka	67533.00	77247.00	77247.000	103423.00	129600.00	419600.0000	346188.000	2
16	17	Kerala	109500.00	50370.00	27794.000	26423.00	25566.00	21663.0000	15325.000	1
17	18	Lakshadweep	109.00	110.00	120.000	150.00	200.00	170.0000	160.000	
18	19	Madhya Pradesh	16887.00	23400.00	22973.950	27763.60	30884.47	50457.0700	61037.000	
19	20	Maharashtra	1045.24	10950.00	587235.000	537735.00	469098.00	21420.3300	272652.000	4
20	21	Manipur	3041.00	4380.00	5475.000	2562.00	3041.00	2732.0000	2423.000	
21	22	Meghalaya	6773.00	4599.00	4015.000	13.94	13.94	13.2650	15.096	
22	23	Mizoram	32536.00	1682.50	45625.000	45625.00	6396.00	55480.0000	3234.000	
23	24	Nagaland	3846.00	5736.00	6786.000	8977.00	10237.00	12836.0000	14052.500	
24	25	Odisha	600.00	2902.00	896.618	1871.13	27859.17	6890.8050	12092.205	
25	26	Pondicherry	3284.00	4927.50	10374.000	16425.00	10376.00	9252.2500	8842.500	
26	27	Punjab	31000.00	22664.00	377.765	43282.27	48073.22	163423.4000	54066.100	1
27	28	Sikkim	1047.00	1668.05	1245.000	894.00	575.00	102.7000	98.000	
28	29	Tamil Nadu	22746.00	205724.95	17884.000	16784.00	150323.47	79114.7920	50873.000	4
29	30	Telangana	17895.00	19554.00	243820.000	123005.00	120961.00	15231.0000	17562.000	1
30	31	Tripura	182.50	1095.00	1168.000	33.00	31.00	30.0000	28.500	

```
df1=pd.DataFrame(total_waste)

#state wise average for machines quantity prediction
df1['Mean'] = df1.mean(axis=1) df1
```

```
<ipython-input-238-610f5af04074>:2: FutureWarning: Dropping of nuisance columns in DataFrame reductions
df1['Mean'] = df1.mean(axis=1)
```

	Sr.no	state	2011	2012	2013	2014	2015	2016	2017
0	1	Andaman & Nicobar	2675.00	2555.00	2123.000	2737.00	1583.00	567.0000	983.000

1	2	Andhra Pradesh	28888.00	243820.00	243820.000	128480.00	128480.00	82863.0000	74588.000			
2	3	Arunachal Pradesh	453.00	673.00	863.000	1273.00	1454.00	3836.0000	6000.000			
3	4	Assam	1226.00	1116.00	781.000	1363.00	24010.00	24030.0000	28153.000			
4	5	Bihar	1527.00	1673.00	1723.000	1893.00	2314.00	2280.0000	2280.000			
5	6	Chandigarh		5548.00	4964.00	4818.000	8992.00	13167.00	21516.7500	12775.000		
6	7	Chhattisgarh		4678.00	5840.00	6123.000	6345.00	6897.00	7300.0000	6650.000		
7	8	Daman	1274.00	1347.00	14736.000	1573.00	1637.00	1733.0000	1836.000			
8	9	Delhi	261234.00	251850.00	23456.000	24567.00	128649.00	232732.0000	228771.000	2		
9	10	Goa	415.00	1642.50	117.730	104.00	106.00	28273.0000	26242.000			
10	11	Gujarat	23666.00	251796.65	251796.650		265568.20	269294.88	271092.0000	269808.000	3	
11	12	Haryana	52883.00	55480.00	50332.690	44536.00	32452.00	23369.0900	46353.000			
12	13	Himachal Pradesh	202.67	106.72	2326.800	1004.00	1577.00	2525.0000	2677.000			
13	14	Jammu & Kashmir	10273.00	11748.00	7832.000	4164.45	6243.20	4645.0000	27870.000			
14	15	Jharkhand	16691.05	81030.00	75817.800	35262.36	35853.52	36772.0000	43644.000			
15	16	Karnataka	67533.00	77247.00	77247.000	103423.00	129600.00	419600.0000	346188.000	2		
16	17	Kerala	109500.00	50370.00	27794.000	26423.00	25566.00	21663.0000	15325.000	1		
17	18	Lakshadweep		109.00	110.00	120.000	150.00	200.00	170.0000	160.000		
18	19	Madhya Pradesh	16887.00	23400.00	22973.950	27763.60	30884.47	50457.0700	61037.000			
19	20	Maharashtra		1045.24	10950.00	587235.000		537735.00	469098.00	21420.3300	272652.000	4
20	21	Manipur	3041.00	4380.00	5475.000	2562.00	3041.00	2732.0000	2423.000			
21	22	Meghalaya	6773.00	4599.00	4015.000	13.94	13.94	13.2650	15.096			

Double-click (or enter) to edit

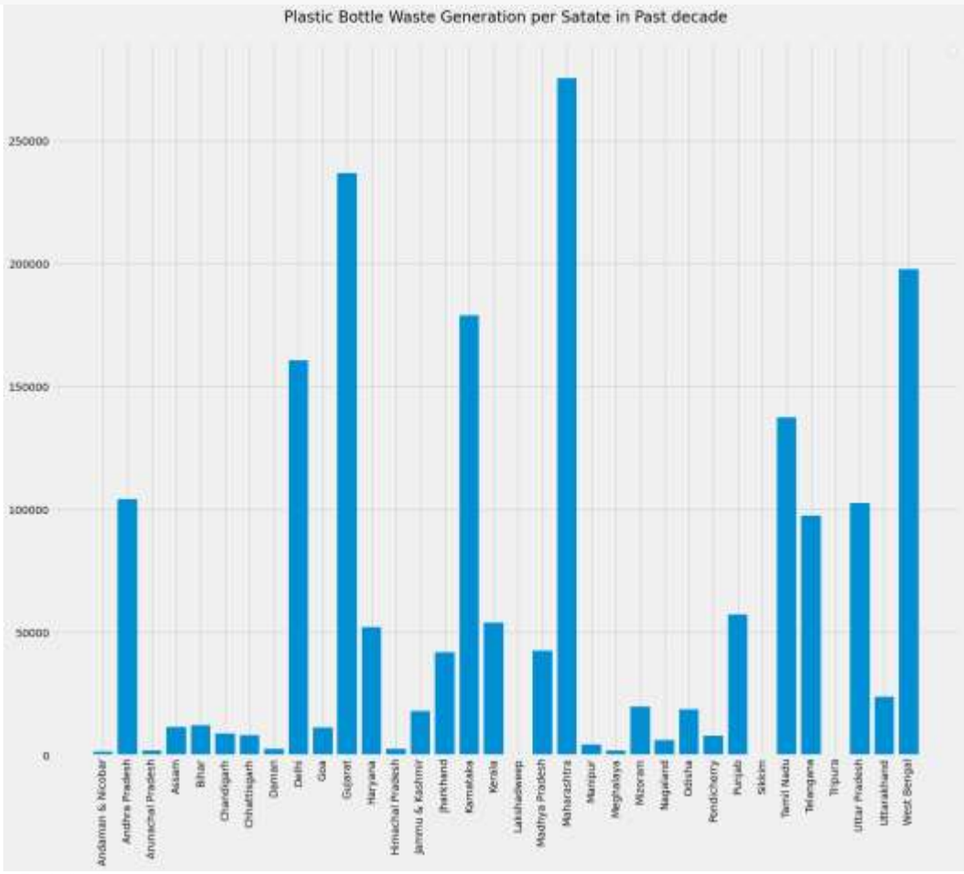
22	23	Mizoram	32536.00	1682.50	45625.000	45625.00	6396.00	55480.0000	3234.000		
23	24	Nagaland	3846.00	5736.00	6786.000	8977.00	10237.00	12836.0000	14052.500		
24	25	Odisha)	600.00	2902.00	896.618	1871.13	27859.17	6890.8050	12092.205		
25	26	Pondicherry]	3284.00	4927.50	10374.000	16425.00	10376.00	9252.2500	8842.500		
26	27	Punjab	31000.00	22664.00	377.765	43282.27	1047.00	48073.22	163423.4000	54066.100	
27	28	Sikkim	1668.05	1245.000	894.00			575.00	102.7000	98.000	1
28	29	Tamil Nadu	22746.00	205724.95	17884.000	16784.00	150323.47	79114.7920	50873.000		
29	30	Telangana	17895.00	19554.00	243820.000	123005.00	120961.00	15231.0000	17562.000	4	
30	31	Tripura	182.50	1095.00	1168.000	33.00	31.00	30.0000	28.500		1
31	32	Uttar Pradesh	56214.00	72351.00	86222.000	93597.97	130777.39	150265.0000	22003.000	2	

```
plt.style.use('fivethirtyeight')
plt.figure(figsize=(19,16)) plt.title("Generation per Satate in Past decade\n")
plt.bar(x=df1['state'], height=df1['Mean'])
plt.xticks(rotation=90)
plt.rcdefaults()
plt.legend()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label
<matplotlib.legend.Legend at 0x7f45c5674190>
```

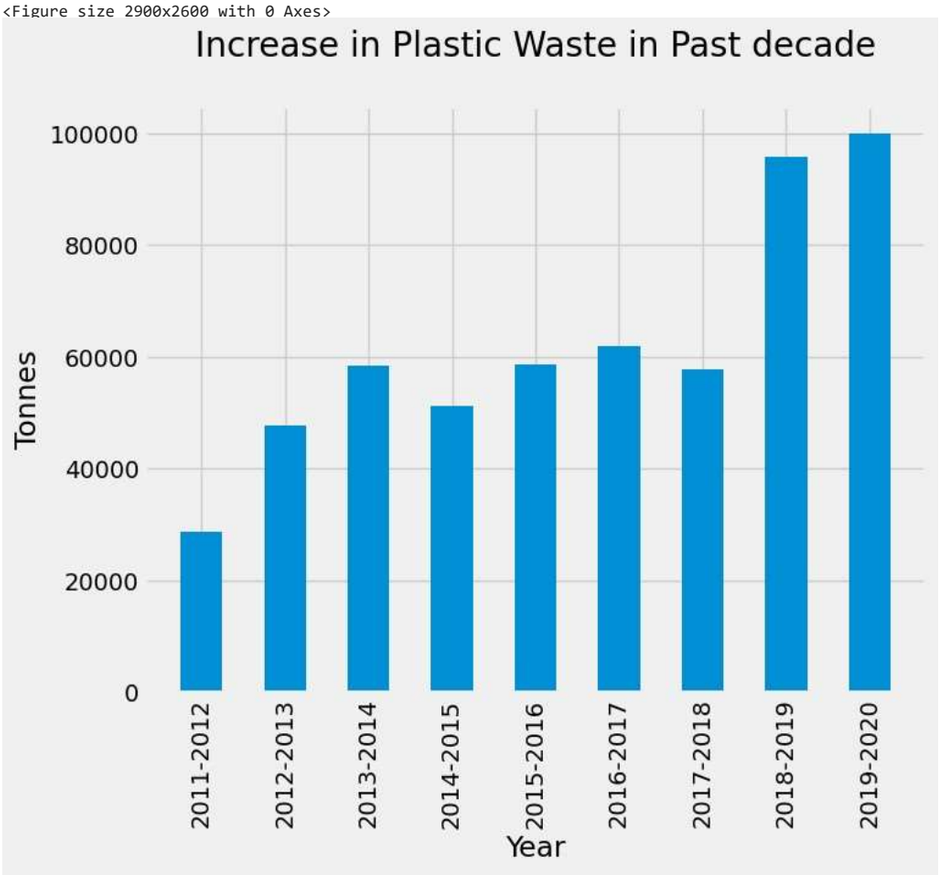
```
x = ['2011','2012','2013','2014','2015','2016','2017','2019']
y = df1['2019']
```

```
plt.style.use('fivethirtyeight')

plt.figure(figsize=(29,26))
y=mean_year fig =
plt.figure()
ax = fig.add_axes([0,0,1,1])
Years = ['2011-2012','2012-2013','2013-2014','2014-2015','2015-2016','2016-2017','2017-2018','2018-2019','2019-2020']
plt.title("Increase in Plastic Waste in Past decade\n") ax.bar(Years,y,width=0.5) plt.xlabel("Year")
plt.ylabel("Tonnes") plt.xticks(rotation=90)
```




```
([0, 1, 2, 3, 4, 5, 6, 7, 8],
 [Text(0, 0, ''),
Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
  Text(0, 0, ''),
```

```
  Text(0, 0, '')])
<Figure size 2900x2600 with 0 Axes>
```




```
#here 1.91625 tonnes is the calculation for each machine capacity for the year
df1['Number of reverse vending machines']=df1["Mean"]//1.91625 df1
```

	Sr.no	state	2011	2012	2013	2014	2015	2016	2017
0	1	Andaman & Nicobar	2675.00	2555.00	2123.000	2737.00	1583.00	567.0000	983.000
1	2	Andhra Pradesh	28888.00	243820.00	243820.000		128480.00	128480.00	82863.0000	74588.000
2	3	Arunachal Pradesh	453.00	673.00	863.000	1273.00	1454.00	3836.0000	6000.000
3	4	Assam	1226.00	1116.00	781.000	1363.00	24010.00	24030.0000	28153.000
4	5	Bihar	1527.00	1673.00	1723.000	1893.00	2314.00	2280.0000	2280.000
5	6	Chandigarh	12775.000	5548.00	4964.00	4818.000	8992.00	13167.00	21516.7500
6	7	Chhattisgarh		4678.00	5840.00	6123.000	6345.00	6897.00	7300.0000	6650.000
7	8	Daman	1274.00	1347.00	14736.000	1573.00	1637.00	1733.0000	1836.000
8	9	Delhi	2	261234.00	251850.00	23456.000	24567.00	128649.00	232732.0000	228771.000
9	10	Goa	415.00	1642.50	117.730	104.00	106.00	28273.0000	26242.000
10	11	Gujarat	269808.000	23666.00	251796.653	251796.650		265568.20	269294.88	271092.0000
11	12	Haryana	52883.00	55480.00	50332.690	44536.00	32452.00	23369.0900	46353.000
12	13	Himachal Pradesh	202.67	106.72	2326.800	1004.00	1577.00	2525.0000	2677.000
13	14	Jammu & Kashmir	10273.00	11748.00	7832.000	4164.45	6243.20	4645.0000	27870.000
14	15	Jharkhand	16691.05	81030.00	75817.800	35262.36	35853.52	36772.0000	43644.000
15	16	Karnataka	2	67533.00	77247.00	77247.000	103423.00	129600.00	419600.0000	346188.000
16	17	Kerala	109500.00	50370.00	27794.000	26423.00	25566.00	21663.0000	15325.000	1
17	18	Lakshadweep		109.00	110.00	120.000	150.00	200.00	170.0000	160.000
18	19	Madhya Pradesh	16887.00	23400.00	22973.950	27763.60	30884.47	50457.0700	61037.000

19	20	Maharashtra	1045.24	10950.00	587235.000	537735.00	469098.00	21420.3300	272652.000	
20	21	Manipur	3041.00	4380.00	5475.000	2562.00	3041.00	2732.0000	2423.000	
21	22	Meghalaya	6773.00	4599.00	4015.000	13.94	13.94	13.2650	15.096	
22	23	Mizoram	32536.00	1682.50	45625.000	45625.00	6396.00	55480.0000	3234.000	
23	24	Nagaland	3846.00	5736.00	6786.000	8977.00	10237.00	12836.0000	14052.500	
24	25	Odisha	600.00	2902.00	896.618	1871.13	27859.17	6890.8050	12092.205	
25	26	Pondicherry		3284.00	4927.50	10374.000	16425.00	10376.00	9252.2500	8842.500
26	27	Punjab	31000.00	22664.00	377.765	43282.27	48073.22	163423.4000	54066.100	1
27	28	Sikkim	1047.00	1668.05	1245.000	894.00	575.00	102.7000	98.000	
28	29	Tamil Nadu	22746.00	205724.95	17884.000	16784.00	150323.47	79114.7920	50873.000	4

```
from numpy.lib.arraysetops import setdiff1d
m1=df1['2011'].mean() m2=df1['2012'].mean()
m3=df1['2013'].mean() m4=df1['2014'].mean()
m5=df1['2015'].mean() m6=df1['2016'].mean()
m7=df1['2017'].mean() m8=df1['2018'].mean()
m9=df1['2019'].mean() sum_m=df1['Number of reverse
vending machines'].sum()
mean_year=[m1,m2,m3,m4,m5,m6,m7,m8,m9] mean_year
sum_m
```

```
994260.0
```

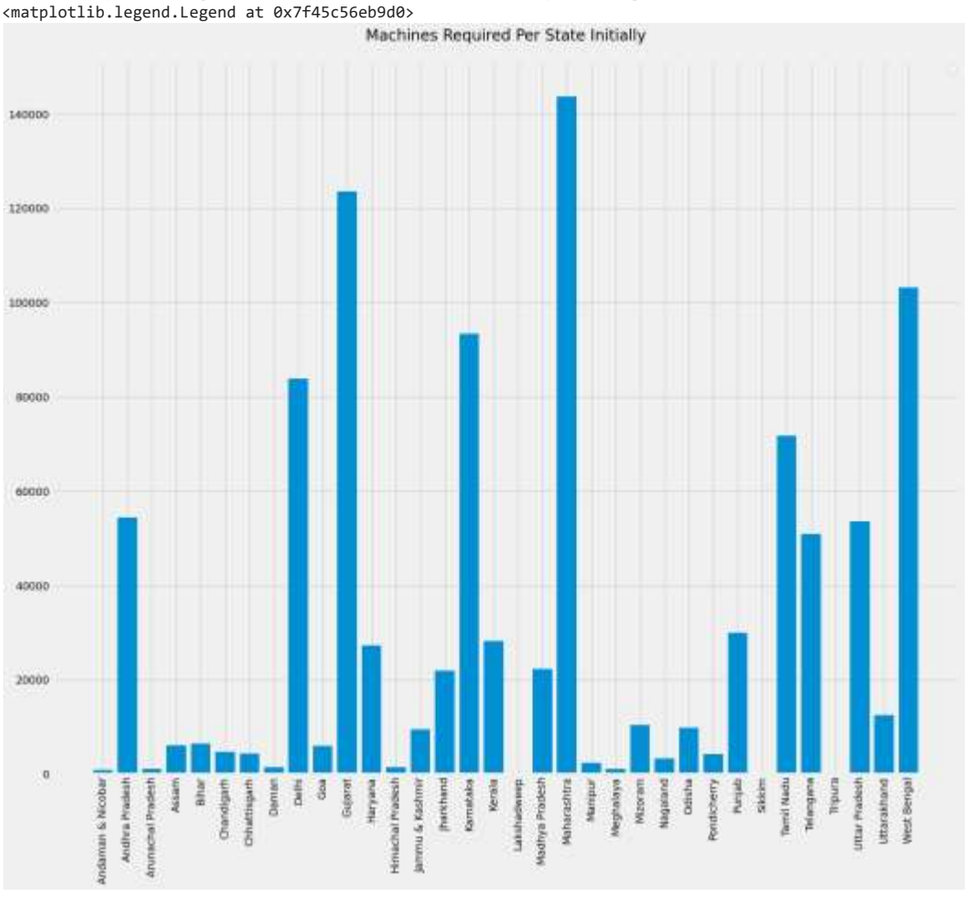
```
x= df1.iloc[:,2:11]  y=
df1.iloc[:,-2]
```

Double-click (or enter) to edit

```
plt.style.use('fivethirtyeight')
plt.figure(figsize=(19,16)) plt.title("Machines
Required Per State Initially\n")
plt.bar(x=df1['state'],
         height=df1['Number of reverse vending
machines'])

plt.xticks(rotation=90)
plt.rcdefaults()
```

```
plt legend()
plt.legend()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label
<matplotlib.legend.Legend at 0x7f45c56eb9d0>
```

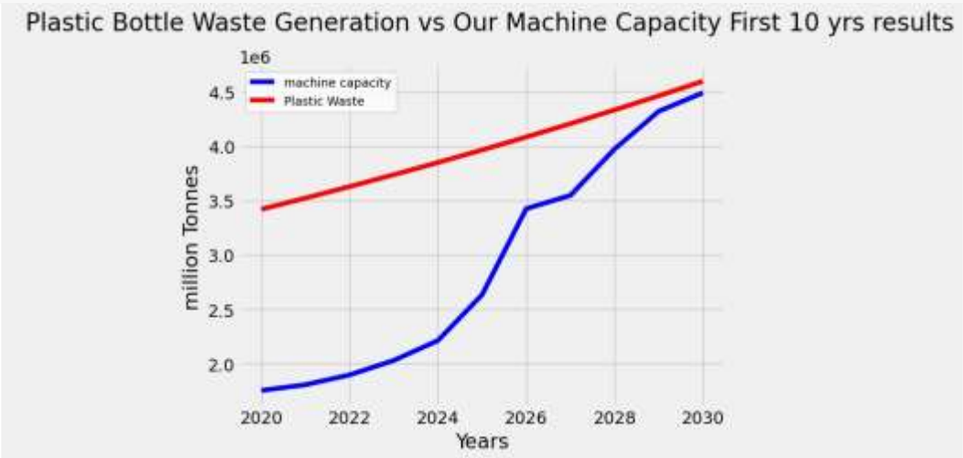


```
df2=pd.read_csv("rate.csv")
df2
```

	Year	PlasticProductionWaste	MachineCapacity	Recycle
0	2020	3419480	1755140	-1664340
1	2021	3522046	1807794	-1714252
2	2022	3627726	1898184	-1729542
3	2023	3736558	2031057	-1705501
4	2024	3848654	2213852	-1634802
5	2025	3964114	2634483	-1329631
6	2026	4083037	3424829	-658208
7	2027	4205529	3546902	-658627
8	2028	4331694	3972530	-359164
9	2029	4461645	4319644	-142001
10	2030	4595499	4489528	-105971

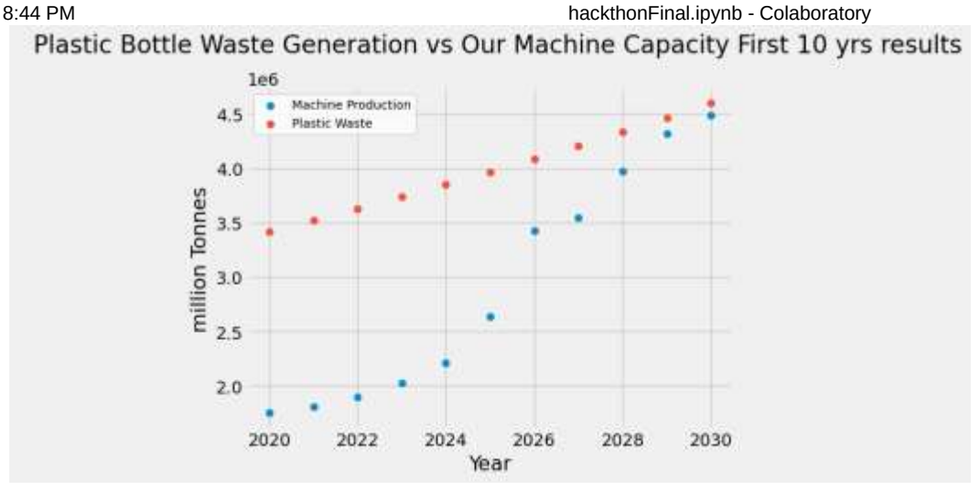
Plastic Waste Generation is Increasing by 2% to 7% For rst decade. Machine Numbers Increasing by 1%-2% per Year and Machine Recycling Capacity Increasing by 3%-4% per Year respectively.

```
plt.style.use('fivethirtyeight')

x = df2['Year'] y = df2['PlasticProductionWaste']
plt.plot(x,df2['MachineCapacity'],label="machine capacity",color='blue') plt.plot(x,
y,label="Plastic Waste",color='red') plt.title("  Plastic Bottle Waste Generation vs Our
Machine Capacity First 10 yrs results\n")

plt.rcdefaults()
plt.legend()

plt.xlabel("Years")
plt.ylabel("million Tonnes")
plt.savefig("plot.png")
plt.show()
```



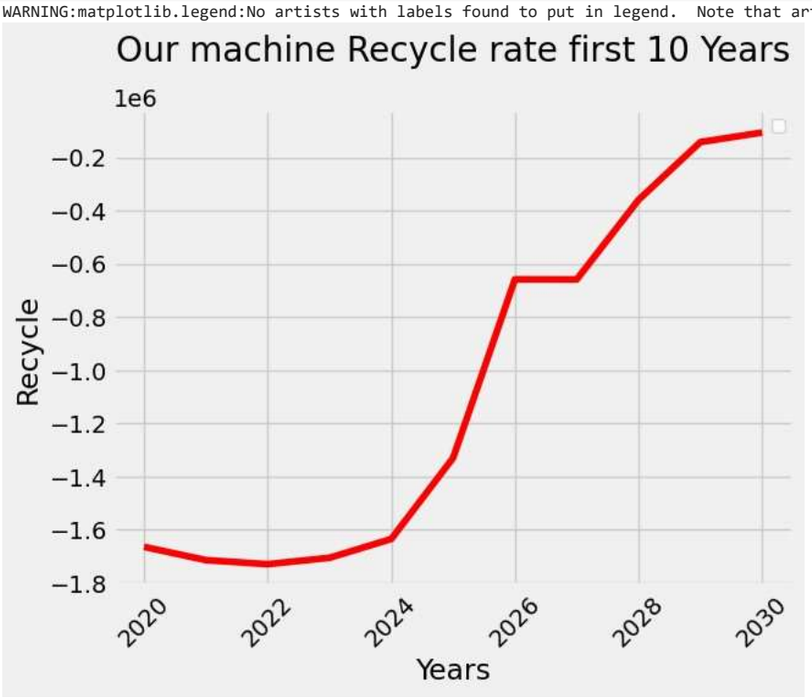

```
plt.style.use('fivethirtyeight')

plt.title("  Plastic Bottle Waste Generation vs Our Machine Capacity
First 10 yrs results\n")
plt.scatter(df2['Year'],df2['MachineCapacity'],label='Machine
Production')
plt.scatter(df2['Year'],df2['PlasticProductionWaste'],label='Plastic
Waste') plt.xlabel("Year") plt.ylabel("million Tonnes") plt.rcdefaults()
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f45c595ac40>
```




```
x = df2['Year'] y
= df2['Recycle']
  plt.style.use('fivethirtyeight')


plt.plot(x, y,color='red') plt.title("Our machine
Recycle rate first 10 Years\n") plt.xlabel("Years")
plt.ylabel("Recycle")

plt.rcdefaults()
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label
```



Recycling Rate is negative initally since machine capacity is less then total Plastic Bottle waste generation. Recycling Rate will increase down the decades.

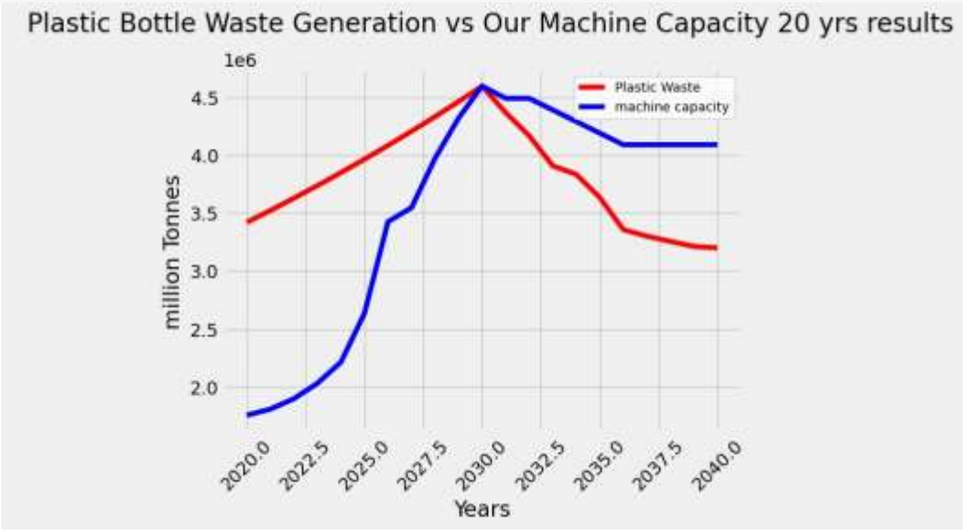
```
df3=pd.read_csv("rate1.csv")
df3
```

	Year	PlasticProductionWaste	MachineCapacity	Recycle
0	2020	3419480	1755140	-1664340
1	2021	3522046	1807794	-1714252
2	2022	3627726	1898184	-1729542
3	2023	3736558	2031057	-1705501
4	2024	3848654	2213852	-1634802
5	2025	3964114	2634483	-1329631
6	2026	4083037	3424829	-658208
7	2027	4205529	3546902	-658627
8	2028	4331694	3972530	-359164
9	2029	4461645	4319644	-142001

10	2030	4595499	4595499	0
11	2031	4366365	4489529	123164
12	2032	4166635	4489530	322895
13	2033	3907436	4389531	482095
14	2034	3833635	4289532	455897
15	2035	3633666	4189533	555867
16	2036	3356766	4089534	732768
17	2037	3299898	4089535	789637
18	2038	3255554	4089536	833982
19	2039	3211441	4089537	878096
20	2040	3199988	4089538	889550

Double-click (or enter) to edit

```
x = df3['Year'] y =
df3['PlasticProductionWaste']
  plt.style.use('fivethirtyeight')


plt.plot(x, y,label="Plastic Waste",color='red') plt.title("  Plastic Bottle Waste
Generation vs Our Machine Capacity 20 yrs results\n")
plt.plot(x,df3['MachineCapacity'],label="machine capacity",color='blue')
plt.xlabel("Years") plt.ylabel("million Tonnes")

plt.rcdefaults()
plt.legend()
plt.xticks(rotation=45)
plt.show()
```



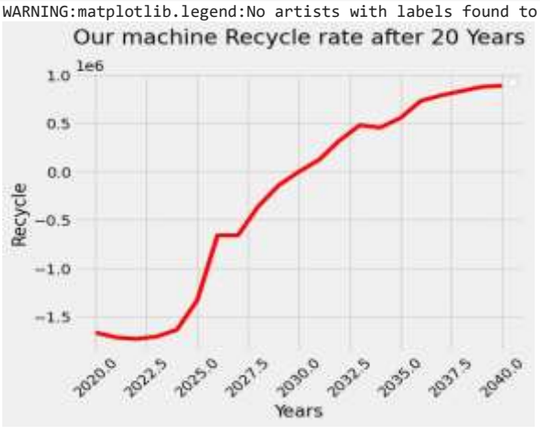
Here Machine Recycling Capacity has Surpassed Plastic Bottle Waste Generation.

```
plt.style.use('fivethirtyeight')


plt.scatter(df3['Year'],df3['MachineCapacity'],label='Machine Production') plt.title("
Plastic Bottle Waste Generation vs Our Machine Capacity 20 yrs results\n")
plt.scatter(df3['Year'],df3['PlasticProductionWaste'],label='Plastic Waste')
plt.xlabel("Year") plt.ylabel("million Tonnes")

plt.rcdefaults()
plt.xticks(rotation=45)
plt.legend()
```

```
<matplotlib.legend.Legend at 0x7f45c586b850>
```

```
x = df3['Year'] y
= df3['Recycle']
  plt.style.use('fivethirtyeight')


plt.plot(x, y,color='red') plt.title("Our machine
Recycle rate after 20 Years\n") plt.xlabel("Years")
plt.ylabel("Recycle")

plt.rcdefaults()
plt.legend()
plt.xticks(rotation=45)
plt.show()
```

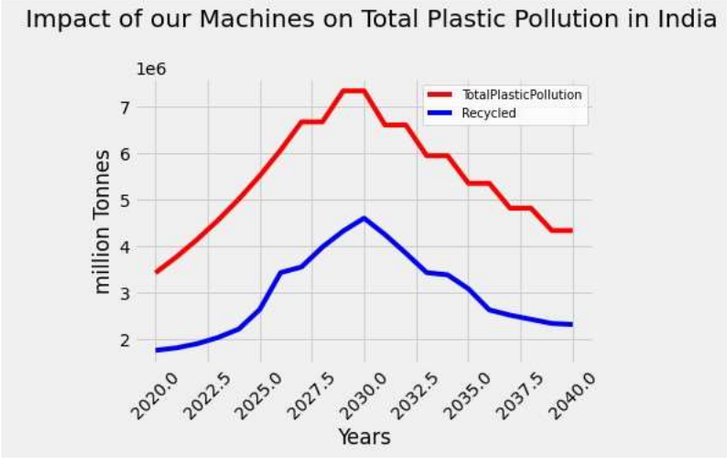
```
WARNING:matplotlib.legend:No artists with labels found to put in legend.  Note that artists whose label
```




```
df4=pd.read_csv("pollution.csv")
df4
```

	Year	xx	yy	zz	Recycled	TotalPlasticPollution	ww
0	2020	3419480	1755140	-1664340	1755140	3419480.000	10%
1	2021	3522046	1807794	-1714252	1807794	3761428.000	NaN
2	2022	3627726	1898184	-1729542	1898184	4137570.800	NaN
3	2023	3736558	2031057	-1705501	2031057	4551327.880	NaN
4	2024	3848654	2213852	-1634802	2213852	5006460.668	NaN
5	2025	3964114	2634483	-1329631	2634483	5507106.735	NaN
6	2026	4083037	3424829	-658208	3424829	6057817.408	NaN
7	2027	4205529	3546902	-658627	3546902	6663599.149	NaN
8	2028	4331694	3972530	-359164	3972530	6663599.149	NaN
9	2029	4461645	4319644	-142001	4319644	7329959.064	NaN
10	2030	4595499	4595499	0	4595499	7329959.064	NaN
11	2031	4366365	4489529	123164	4243201	6596963.158	NaN
12	2032	4166635	4489530	322895	3843740	6596963.158	NaN
13	2033	3907436	4389531	482095	3425341	5937266.842	NaN
14	2034	3833635	4289532	455897	3377738	5937266.842	NaN
15	2035	3633666	4189533	555867	3077799	5343540.158	NaN
16	2036	3356766	4089534	732768	2623998	5343540.158	NaN
17	2037	3299898	4089535	789637	2510261	4809186.142	NaN
18	2038	3255554	4089536	833982	2421572	4809186.142	NaN
19	2039	3211441	4089537	878096	2333345	4328267.528	NaN
20	2040	3199988	4089538	889550	2310438	4328267.528	NaN

```
x = df4['Year'] y =
df4['TotalPlasticPollution']
  plt.style.use('fivethirtyeight')


plt.plot(x, y,label="TotalPlasticPollution",color='red') plt.title("
Impact of our Machines on Total Plastic Pollution in India\n")
plt.plot(x,df4['Recycled'],label="Recycled",color='blue')
plt.xlabel("Years") plt.ylabel("million Tonnes")

plt.rcdefaults()
plt.legend()
plt.xticks(rotation=45)
plt.show()
```



As the recycling rate increases the total plastic pollution also decreases gradually.

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import seaborn as sn
```

```
X = df4[['xx', 'yy']]
y = df4['Recycled']
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.25,random_state=0)
logistic_regression= LogisticRegression() logistic_regression.fit(X_train,y_train)
y_pred=logistic_regression.predict(X_test) print (X_test)
```

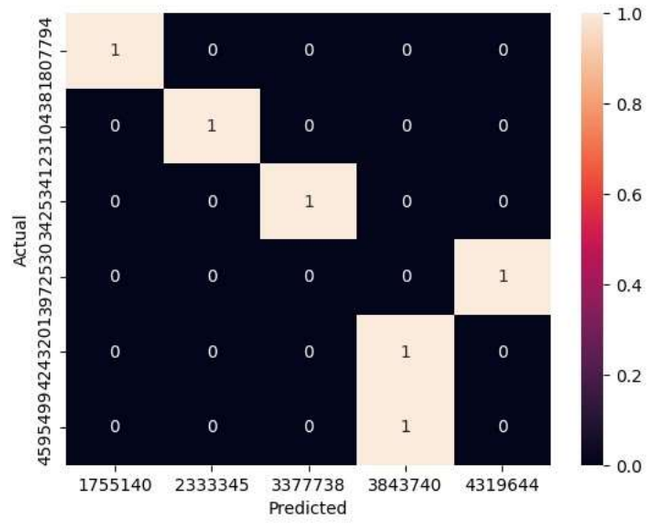
```
          xx        yy
8   4331694  3972530
13  3907436  4389531
20  3199988  4089538
1   3522046  1807794
11  4366365  4489529
10  4595499  4595499 /usr/local/lib/python3.9/dist-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to
converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html  Please
also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
```

```
print (y_pred)
```

```
[4319644 3377738 2333345 1755140 3843740 3843740]
```

```
confusion_matrix = pd.crosstab(y_test, y_pred, rownames=['Actual'], colnames=['Predicted'])
sn.heatmap(confusion_matrix, annot=True)
```

```
<Axes: xlabel='Predicted', ylabel='Actual'>
```




```
import statsmodels.api as sm
x=sm.add_constant(X)
model=sm.OLS(y,X)
fitted_model=model.fit()
print(fitted_model.summary())
```

```
                                 OLS Regression Results
==============================================================================
Dep. Variable:               Recycled   R-squared (uncentered):                   0.977
Model:                            OLS   Adj. R-squared (uncentered):              0.975
Method:                 Least Squares   F-statistic:                              409.2
Date:                Wed, 29 Mar 2023   Prob (F-statistic):                    2.40e-16
Time:                        07:28:49   Log-Likelihood:                         -303.89
No. Observations:                  21   AIC:                                      611.8
Df Residuals:                      19   BIC:                                      613.9
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
coef    std err          t      P>|t|      [0.025      0.975] ----------------
```

```
------------------------------------------------------------xx
0.4120     0.106     3.886     0.001      0.190      0.634 yy           0.4088
0.110            3.703            0.002              0.178              0.640
==============================================================================
Omnibus:                        3.121   Durbin-Watson:                   0.144
Prob(Omnibus):                  0.210   Jarque-Bera (JB):                1.705
Skew:                           0.418   Prob(JB):                        0.426
Kurtosis:                                           1.883      Cond.  No.                                         7.48
==============================================================================

Notes:
[1] R² is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```