```
import numpy as np
array1=np.array([[1,2,3],[4,5,6]
,[7,8,9]])
```

OUTPUT:

```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

```
array2=np.array([[11,12,13],[14,15,16],[17,18,19]])
array2
```

Output:

```
array([[11, 12, 13],
       [14, 15, 16],
       [17, 18, 19]])
```

1. Matrix Operation

1.1 Addition

```
resultarray=array1+array2
print("\nUsing
Operator:\n",resultarray)
resultarray=np.add(array1,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output:

```
Using Operator:
 [[12 14 16]
 [18 20 22]
 [24 26 28]]
```

Using Numpy Function

1.2. Subtraction

```
resultarray=array1-array2
print("\nUsing
Operator:\n",resultarray)
resultarray=np.subtract(array1
,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
 [[-10 -10 -10]
 [-10 -10 -10]
 [-10 -10 -10]]
```

Using Numpy

1.3. Multiplication

```
resultarray=array1*array2
print("\nUsing
Operator:\n",resultarray)
resultarray=np.multiply(array1
,array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
 [[ 11  24  39]
 [ 56  75  96]
 [119 144 171]]

Using Numpy
 Function: [[ 11
 24  39]
 [ 56  75  96]
 [119 144 171]]
```

1.4. Division

```
resultarray=array1/array2
print("\nUsing
Operator:\n",resultarray)
resultarray=np.divide(array1,
array2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
 [[0.09090909 0.16666667 0.23076923]
 [0.28571429 0.33333333 0.375     ]
 [0.41176471 0.44444444 0.47368421]]

Using Numpy Function:
 [[0.09090909 0.16666667 0.23076923]
 [0.28571429 0.33333333 0.375     ]
 [0.41176471 0.44444444 0.47368421]]
```

1.5. Mod

```
resultarray=array1%array2
print("\nUsing
Operator:\n",resultarray)
resultarray=np.mod(array1,arr
ay2)
print("\nUsing Numpy Function:\n",resultarray)
```

Output

```
Using Operator:
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
```

Using Numpy
Function:

1.6. dot Product

```
resultarray=np.dot(array1,arra
y2) print("",resultarray)
```

Output

```
[[ 90  96 102]
 [216 231 246]
 [342 366 390]]
```

.7. Transpose

```
resultarray=np.transpose(array
1) print(resultarray)
#Or
resultarray=array1.trans
pose()
print(resultarray)
```

Output

```
[[1 4 7]
 [2 5 8]
 [3 6 9]]
[[1 4 7]
 [2 5 8]
 [3 6 9]]
```

2. Horizontal and vertical stacking of Numpy Arrays

2.1. Horizontal Stacking

```
resultarray=np.hstack((array1,
array2)) resultarray
```

Output

```
array([[ 1, 2, 3, 11, 12, 13],
    [ 4, 5, 6, 14, 15, 16],
    [ 7, 8, 9, 17, 18, 19]])
```

2.2. Vertical Stacking

```
resultarray=np.vstack((array1,
array2)) resultarray
```

Output

```
array([[ 1, 2, 3],
    [ 4, 5, 6],
    [ 7, 8, 9],
    [11, 12, 13],
    [14, 15, 16],
    [17, 18, 19]])
```

3. Custom sequence generation

3.1. Range

```
import numpy as np
nparray=np.arange(0,12,1).reshape(3,4)
nparray
```

Output

```
array([[ 0, 1, 2, 3],
    [ 4, 5, 6, 7],
    [ 8, 9, 10, 11]])
```

3.2. Linearly Separable

```
nparray=np.linspace(start=0,stop=24,num=12).reshape
(3,4) nparray
```

Output

```
array([[ 0. , 2.18181818, 4.36363636, 6.54545455],

     [ 8.72727273, 10.90909091, 13.09090909, 15.27272727],

    [17.45454545, 19.63636364, 21.81818182, 24. ]])
```

3.3. Empty Array

```
nparray=np.empty((3,3),i
nt) nparray
```

Output

```
 array([[ 11,
   24, 39],

[ 56, 75, 96],

   [119, 144, 171]])
```

3.4. Emply Like Some other array

```
nparray=np.empty_like(ar
ray1) nparray
```

Output

```
array([[ 90,  96, 102],
     [216, 231, 246],
     [342, 366, 390]])
```

3.5. Identity Matrix

```
nparray=np.identity(3
) nparray
```

Output

```
  array([[1.,
     0., 0.],

[0., 1., 0.],

      [0., 0.,
         1.]])
```

4. Arithmetic and Statistical Operations, MathematicalOperations, Bitwise Operators

4.1. Arithmetic Operation

```
array1=np.array([1,2,3,4,5])
array2=np.array([11,12,13,14,15
]) print(array1)
print(array2)
```

Output

```
 [1 2 3 4 5]
[11 12 13 14 15]
```

```
# Addition print(np.add(array1,array2))
# Subtraction
print(np.subtract(array1,array2)) #
Multiplication
print(np.multiply(array1,array2)) #
Division print(np.divide(array1,array2))
```

Output

```
 [12 14 16 18 20]
 [-10 -10 -10 -10 -10]
 [11 24 39 56 75]
 [0.09090909 0.16666667 0.23076923 0.28571429 0.33333333]
```

4.2. Statistical and Mathematical Operations

```
array1=np.array([1,2,3,4,5,9,6,7,8,9,9])
# Standard Deviation
print(np.std(array1))
#Minimum print(np.min(array1))
#Summation
print(np.sum(array1)) #Median
print(np.median(array1)) #Mean
print(np.mean(array1)) #Mode
```

```
from scipy import stats
print("Most                        Frequent
element=",stats.mode(array1)[0])
print("Number                             of
Occarances=",stats.mode(array1)[1])        #
Variance
print(np.var(array1))
```

Output

```
2.7990553306073913
1
63
6.0
5.7272727272727275
Most Frequent
element= [9] Number
of Occarances= [3]
7.834710743801653
```

4.3. Bitwise Operations

```
array1=np.array([1,2,3],dtype=np
.uint8)
array2=np.array([4,5,6])
# AND
resultarray=np.bitwise_and(array1,arra
y2) print(resultarray)
# OR
resultarray=np.bitwise_or(array1
,array2) print(resultarray)
#LeftShift
resultarray=np.left_shift(array1
,2) print(resultarray)
#RightShift
resultarray=np.right_shift(array
1,2) print(resultarray)
```

Output

```
[0 0 2]

[5 7 7]

[ 4  8 12]
```

[0 0 0]

```
### You can get Binary Representation of Number ######
print(np.binary_repr(10,8))
resultarray=np.left_shift(10,2)
print(resultarray)
print(np.binary_repr(np.left_shift(10
,2),8))
```

Output

```
00001010
40
00101000
```

5. Copying and viewing arrays

5.1 Copy

```
array1=np.arange(1,10)
print(array1)
newarray=array1.copy()
print(newarray)
##modification in
Original Array
array1[0]=100
print(array1)
print(newarray)
```

Output

```
 [1  2  3  4  5  6  7  8  9]
[1  2  3  4  5  6  7  8  9]
[100 2  3  4  5  6  7  8  9]
[1  2  3  4  5  6  7  8  9]
```

5.2 View

```
array1=np.arange(1,10)
print(array1)
newarray=array1.view()
```

```
print(newarray)
##modification in
Original Array
array1[0]=100
print(array1)
```

Output

```
 [1  2  3  4  5  6  7  8  9]
[1  2  3  4  5  6  7  8  9]
[100 2  3  4  5  6  7  8  9]
[100 2  3  4  5  6  7  8  9]
```

6. Searching

```
array1=np.array([[1,2,3,12,5,7],[94,5,6,7,89,44],[7,8,9
,11,13,14]])
```

Output

```
 [[ 1  2  3 12  5  7]
 [94  5  6  7 89 44]
 [ 7  8  9 11 13 14]]
```

```
 np.sort(array1,axis=0)
```

Output

```
array([[ 1, 2, 3, 7, 5, 7],

     [ 7, 5, 6, 11, 13, 14],

     [94, 8, 9, 12, 89, 44]])
```

```
 np.sort(array1,axis=1)
```

Output

```
array([[ 1, 2, 3, 5, 7, 12],

     [ 5, 6, 7, 44, 89, 94]
[ 7, 8, 9, 11, 13, 14]])
```

7. Searching

```
array1=np.array([1,2,3,12,5,7])
np.searchsorted(array1,7,side="left")#Perform Search After
sorting
```

Output

```
3
```

8. Counting

```
array1=np.array([1,2,3,12,5,7,0])
print(np.count_nonzero(array1))#Return total Non
Zero element print(np.nonzero(array1))#Return
Index
```

Output

```
6
(array([0, 1, 2, 3, 4, 5], dtype=int64),)
7
```

9. Data Stacking

```
array1=np.array(np.arange(1,5).resh
ape(2,2)) print(array1)
array2=np.array(np.arange(11,15).resh
ape(2,2)) print(array2)
```

Output

```
 [[1 2]
 [3 4]]
[[11 12]
 [13 14]]
```

```
newarray=np.stack([array1,array2],axis=0)
```

```
print(newarray)
```

Output

```
[[1 2]
 [3 4]]
[[11 12]
 [13 14]]
```

```
newarray=np.stack([array1,array2],axi
s=1) print(newarray)
```

Output

```
[[1 2]
 [11 12]]
[[3 4]
 [13 14]]
```

10. Append

```
array1=np.arange(1,10).reshape
(3,3) print(array1)
array2=np.arange(21,30).reshap
e(3,3) print(array2)
```

Output

```
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
[[21 22 23]
 [24 25 26]
 [27 28 29]]
```

```
np.append(array1,array2,axis=0)
```

Output

```
array([[ 1,
      2, 3],

 [ 4, 5, 6],

 [ 7, 8, 9],

      [21, 22, 23],

     [24, 25, 26],

     [27, 28, 29]])
```

```
np.append(array1,array2,axis=1)
```

Output

```
   array([[ 1, 2, 3,
        21, 22, 23],

  [ 4, 5, 6, 24, 25,
                26],

  [ 7, 8, 9, 27, 28,
               29]])
```

11. Concat

```
array1=np.arange(1,10).reshape
(3,3) print(array1)
array2=np.arange(21,30).reshap
e(3,3) print(array2)
```

Output

```
 [[1 2 3]
 [4 5 6]
 [7 8 9]]
[[21 22 23]
 [24 25 26]
 [27 28 29]]
```

```
np.concatenate((array1,array2),axis=0)
```

Output

```
 array([[ 1,
          2,  3],

  [ 4,  5,  6],

 [ 7,  8,  9],

        [21,  22,  23],

        [24,  25,  26],

        [27,  28,  29]])
```

```
 np.concatenate((array1,array2),axis=1)
```

Output

```
    array([[ 1,  2,  3,
            21,  22,  23],

   [ 4,  5,  6,  24,  25,
                    26],

      [ 7,  8,  9,  27,  28,  29]])
```

```
import numpy as np

# using
loadtxt()
arr =
np.loadtxt("F:\\ISO\\EDS\\NOTES\\dataset\\testmarks1.csv",
delimiter=",",s kiprows=1)
print(type(a
rr))
```

Output

```
<class
'numpy.ndarray'> (10,
5)
```

```
EDS=arr[:,1]
print(EDS)
```

Output

```
[43.05 43.47 42.24 39.24 40.9 39.47 41.68 42.19 44.75
46.95]
```

```
SON=arr[:,2]
print(SON)
```

Output

```
[27.79 28.52 28.16 26.16 26.03 26.31 25.63 27.61 28.35
28.88]
```

SHRUTI BABASAHEB GARAD
DIV A
A4
183
202201070140

```
import pandas as pd
import numpy as np
f1 = open("F:\grainsales.csv","r")
data = pd.read_csv(f1)
df = pd.DataFrame(data)
maindata = df
df['Sales'].describe()
df=df.groupby('Months').sum()
df=df.sort_values (by= [ 'Sales'], ascending=False) df.head(1)
print("Best Month for Sales: July")
print("Revenue Earned was: 16000000")
df
maindata
df = df.groupby("GrainName").sum()
df = df.sort_values(by=["Sales"], ascending = False)
df.head (1)
print("Most Sold Grain is: Wheat")
print("The Best Month for sales is July and this product has occured in July
so this is most sold product with highest sales")
df
maindata
df= df.groupby("City").sum()
df = df.sort_values (by = ['Sales'], ascending= False)
df.head (1)
print("'Asansole' Has sold highest no. of products")
df
maindata
df = df.groupby('State').sum()
df = df.sort_values (by = ['Sales'], ascending = False) print("West
Bengol has highest sales")
```

```
Best Month for Sales: July
Revenue Earned was: 16000000
```

Most Sold Grain is: Wheat

The Best Month for sales is July and this product has occured in July so this is most sold product with highest sales

'Asansole' Has sold highest no. of products

West Bengol has highest sales.

NAME : Shruti Garad

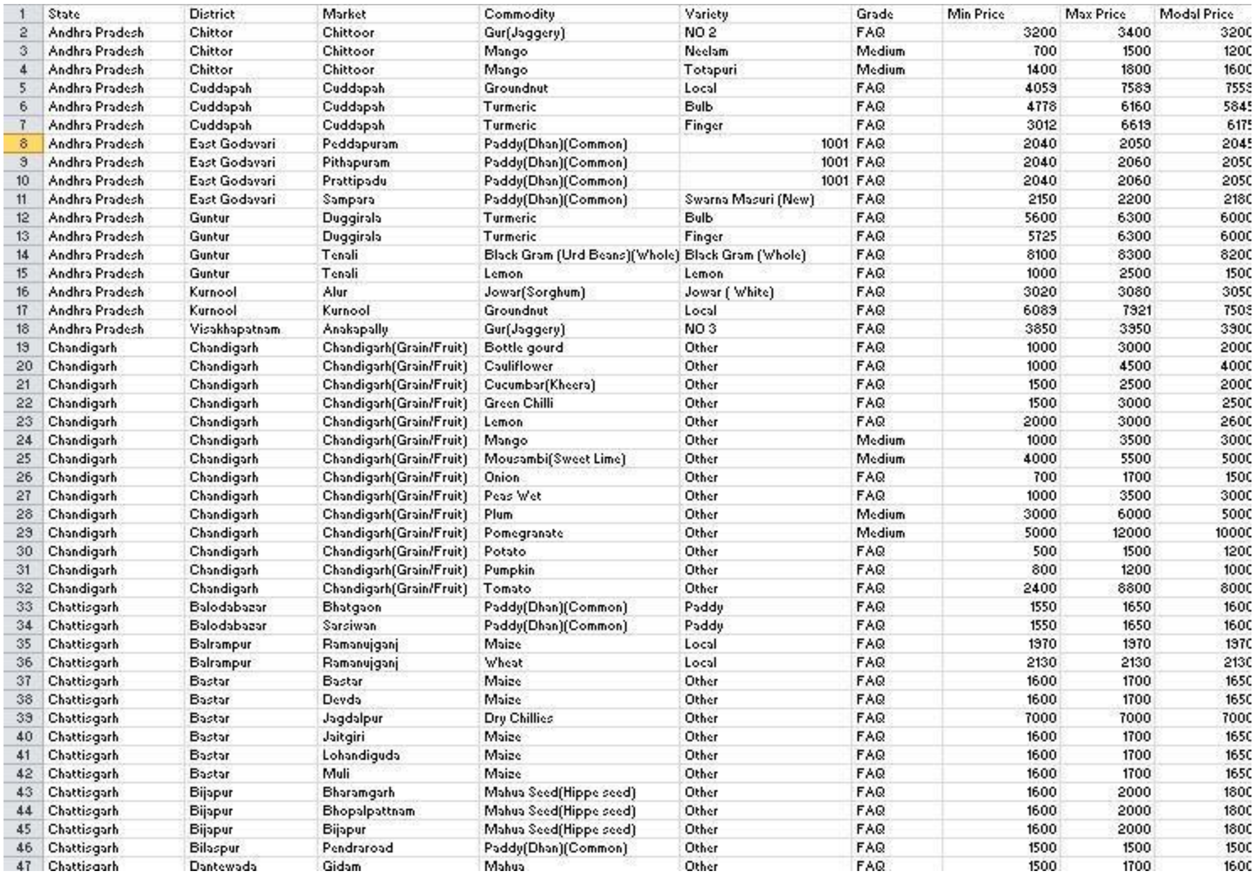
CLASS : A
BATCH : A4
ROLL NO : 183

Name:- Shruti Garad Roll No :- 183

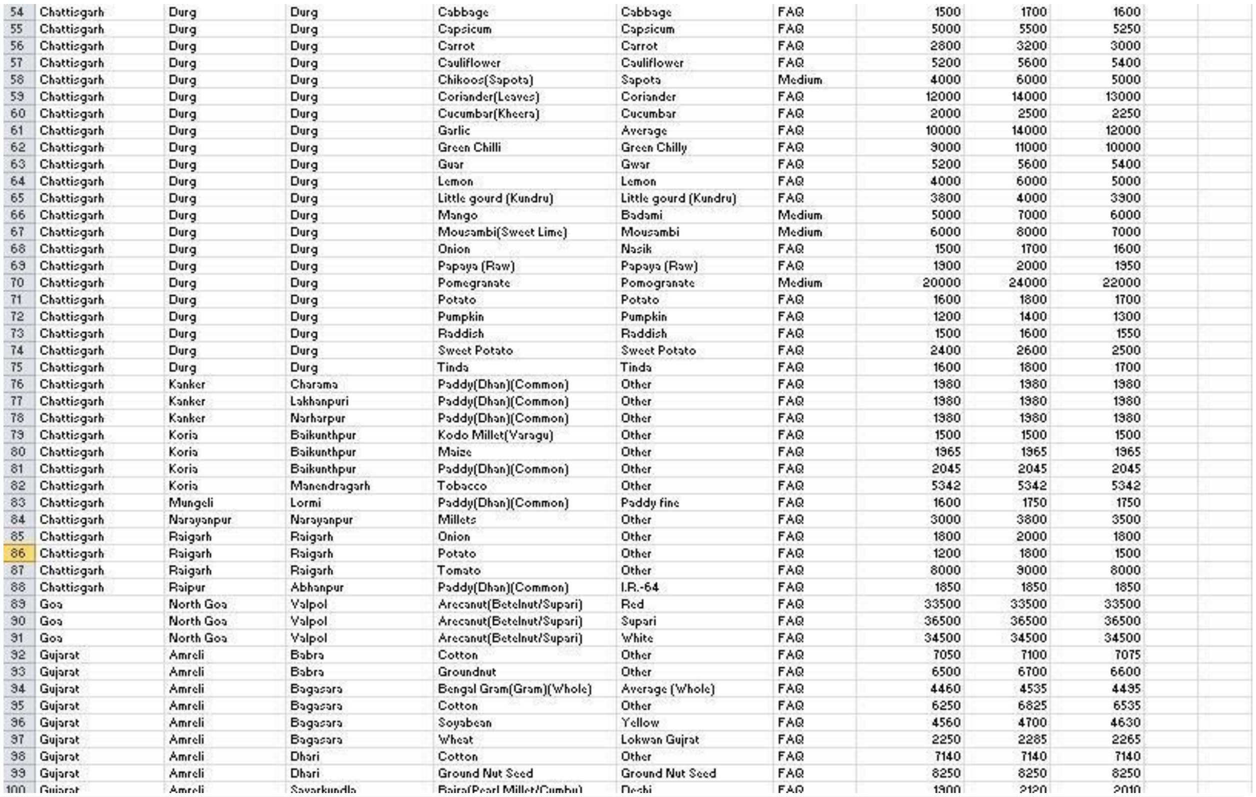
PRn :- 202201070140

Select any one real-life dataset. Perform data analysis. Identify 10 grains for a given dataset. Develop an interactive dashboard using the matplotlib/Seaborn library. (Use any 10 different graphs with proper titles, legends, axis names, etc. to map identified grains)

Table:-



1	State	District	Market	Commodity	Variety	Grade	Min Price	Max Price	Modal Price
2	Andhra Pradesh	Chittor	Chittoor	Gur(Jaggery)	NO 2	FAQ	3200	3400	3200
3	Andhra Pradesh	Chittor	Chittoor	Mango	Neelam	Medium	700	1500	1200
4	Andhra Pradesh	Chittor	Chittoor	Mango	Totapuri	Medium	1400	1800	1600
5	Andhra Pradesh	Cuddapah	Cuddapah	Groundnut	Local	FAQ	4059	7589	7553
6	Andhra Pradesh	Cuddapah	Cuddapah	Turmeric	Bulb	FAQ	4778	6160	5845
7	Andhra Pradesh	Cuddapah	Cuddapah	Turmeric	Finger	FAQ	3012	6619	6175
8	Andhra Pradesh	East Godavari	Peddapuram	Paddy(Dhan)(Common)	1001	FAQ	2040	2050	2045
9	Andhra Pradesh	East Godavari	Pithapuram	Paddy(Dhan)(Common)	1001	FAQ	2040	2060	2050
10	Andhra Pradesh	East Godavari	Prattipadu	Paddy(Dhan)(Common)	1001	FAQ	2040	2060	2050
11	Andhra Pradesh	East Godavari	Sampara	Paddy(Dhan)(Common)	Swarna Masuri (New)	FAQ	2150	2200	2180
12	Andhra Pradesh	Guntur	Duggirala	Turmeric	Bulb	FAQ	5600	6300	6000
13	Andhra Pradesh	Guntur	Duggirala	Turmeric	Finger	FAQ	5725	6300	6000
14	Andhra Pradesh	Guntur	Tenali	Black Gram (Urd Beans)(Whole)	Black Gram (Whole)	FAQ	8100	8300	8200
15	Andhra Pradesh	Guntur	Tenali	Lemon	Lemon	FAQ	1000	2500	1500
16	Andhra Pradesh	Kurnool	Alur	Jowar(Sorghum)	Jowar (White)	FAQ	3020	3080	3050
17	Andhra Pradesh	Kurnool	Kurnool	Groundnut	Local	FAQ	6089	7921	7503
18	Andhra Pradesh	Visakhapatnam	Anakapally	Gur(Jaggery)	NO 3	FAQ	3850	3950	3900
19	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Bottle gourd	Other	FAQ	1000	3000	2000
20	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Cauliflower	Other	FAQ	1000	4500	4000
21	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Cucumbar(Kheera)	Other	FAQ	1500	2500	2000
22	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Green Chilli	Other	FAQ	1500	3000	2500
23	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Lemon	Other	FAQ	2000	3000	2600
24	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Mango	Other	Medium	1000	3500	3000
25	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Mousambi(Sweet Lime)	Other	Medium	4000	5500	5000
26	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Onion	Other	FAQ	700	1700	1500
27	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Peas Wet	Other	FAQ	1000	3500	3000
28	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Plum	Other	Medium	3000	6000	5000
29	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Pomegranate	Other	Medium	5000	12000	10000
30	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Potato	Other	FAQ	500	1500	1200
31	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Pumpkin	Other	FAQ	800	1200	1000
32	Chandigarh	Chandigarh	Chandigarh(Grain/Fruit)	Tomato	Other	FAQ	2400	8800	8000
33	Chattisgarh	Balodabazar	Bhatgaon	Paddy(Dhan)(Common)	Paddy	FAQ	1550	1650	1600
34	Chattisgarh	Balodabazar	Sarsiwan	Paddy(Dhan)(Common)	Paddy	FAQ	1550	1650	1600
35	Chattisgarh	Balrampur	Ramanujganj	Maize	Local	FAQ	1970	1970	1970
36	Chattisgarh	Balrampur	Ramanujganj	Wheat	Local	FAQ	2130	2130	2130
37	Chattisgarh	Bastar	Bastar	Maize	Other	FAQ	1600	1700	1650
38	Chattisgarh	Bastar	Devda	Maize	Other	FAQ	1600	1700	1650
39	Chattisgarh	Bastar	Jagdalpur	Dry Chillies	Other	FAQ	7000	7000	7000
40	Chattisgarh	Bastar	Jaitgiri	Maize	Other	FAQ	1600	1700	1650
41	Chattisgarh	Bastar	Lohandiguda	Maize	Other	FAQ	1600	1700	1650
42	Chattisgarh	Bastar	Muli	Maize	Other	FAQ	1600	1700	1650
43	Chattisgarh	Bijapur	Bharamgarh	Mahua Seed(Hippe seed)	Other	FAQ	1600	2000	1800
44	Chattisgarh	Bijapur	Bhopalpattnam	Mahua Seed(Hippe seed)	Other	FAQ	1600	2000	1800
45	Chattisgarh	Bijapur	Bijapur	Mahua Seed(Hippe seed)	Other	FAQ	1600	2000	1800
46	Chattisgarh	Bilaspur	Pendraroad	Paddy(Dhan)(Common)	Other	FAQ	1500	1500	1500
47	Chattisgarh	Dantewada	Gidam	Mahua	Other	FAQ	1500	1700	1600



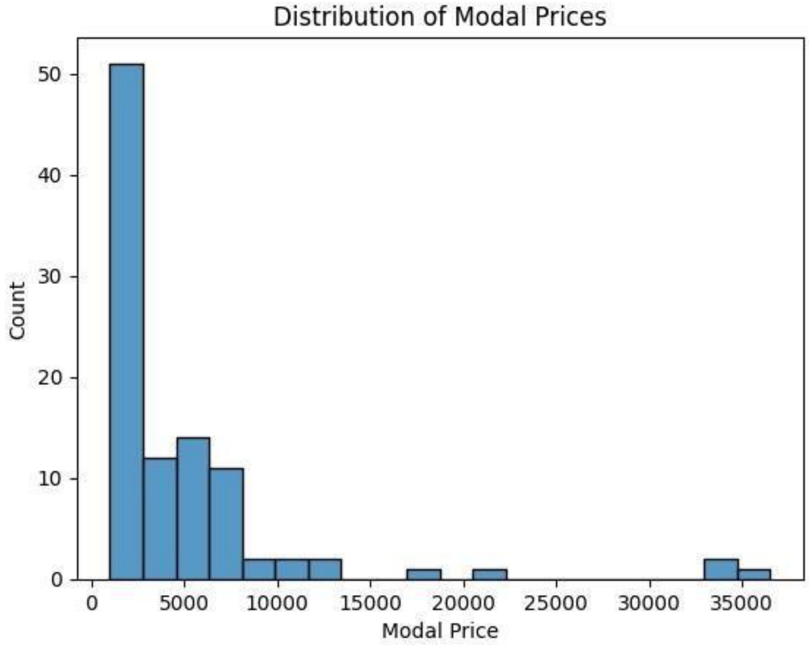
54	Chattisgarh	Durg	Durg	Cabbage	Cabbage	FAQ	1500	1700	1600
55	Chattisgarh	Durg	Durg	Capsicum	Capsicum	FAQ	5000	5500	5250
56	Chattisgarh	Durg	Durg	Carrot	Carrot	FAQ	2800	3200	3000
57	Chattisgarh	Durg	Durg	Cauliflower	Cauliflower	FAQ	5200	5600	5400
58	Chattisgarh	Durg	Durg	Chikoos(Sapota)	Sapota	Medium	4000	6000	5000
59	Chattisgarh	Durg	Durg	Coriander(Leaves)	Coriander	FAQ	12000	14000	13000
60	Chattisgarh	Durg	Durg	Cucumbar(Kheera)	Cucumbar	FAQ	2000	2500	2250
61	Chattisgarh	Durg	Durg	Garlic	Average	FAQ	10000	14000	12000
62	Chattisgarh	Durg	Durg	Green Chilli	Green Chilly	FAQ	9000	11000	10000
63	Chattisgarh	Durg	Durg	Guar	Gwar	FAQ	5200	5600	5400
64	Chattisgarh	Durg	Durg	Lemon	Lemon	FAQ	4000	6000	5000
65	Chattisgarh	Durg	Durg	Little gourd (Kundru)	Little gourd (Kundru)	FAQ	3800	4000	3900
66	Chattisgarh	Durg	Durg	Mango	Badami	Medium	5000	7000	6000
67	Chattisgarh	Durg	Durg	Mousambi(Sweet Lime)	Mousambi	Medium	6000	8000	7000
68	Chattisgarh	Durg	Durg	Onion	Nasik	FAQ	1500	1700	1600
69	Chattisgarh	Durg	Durg	Papaya (Raw)	Papaya (Raw)	FAQ	1900	2000	1950
70	Chattisgarh	Durg	Durg	Pomegranate	Pomogranate	Medium	20000	24000	22000
71	Chattisgarh	Durg	Durg	Potato	Potato	FAQ	1600	1800	1700
72	Chattisgarh	Durg	Durg	Pumpkin	Pumpkin	FAQ	1200	1400	1300
73	Chattisgarh	Durg	Durg	Raddish	Raddish	FAQ	1500	1600	1550
74	Chattisgarh	Durg	Durg	Sweet Potato	Sweet Potato	FAQ	2400	2600	2500
75	Chattisgarh	Durg	Durg	Tinda	Tinda	FAQ	1600	1800	1700
76	Chattisgarh	Kanker	Charama	Paddy(Dhan)(Common)	Other	FAQ	1980	1980	1980
77	Chattisgarh	Kanker	Lakhanpuri	Paddy(Dhan)(Common)	Other	FAQ	1980	1980	1980
78	Chattisgarh	Kanker	Narharpur	Paddy(Dhan)(Common)	Other	FAQ	1980	1980	1980
79	Chattisgarh	Koria	Baikunthpur	Kodo Millet(Varagu)	Other	FAQ	1500	1500	1500
80	Chattisgarh	Koria	Baikunthpur	Maize	Other	FAQ	1965	1965	1965
81	Chattisgarh	Koria	Baikunthpur	Paddy(Dhan)(Common)	Other	FAQ	2045	2045	2045
82	Chattisgarh	Koria	Manendragarh	Tobacco	Other	FAQ	5342	5342	5342
83	Chattisgarh	Mungeli	Lormi	Paddy(Dhan)(Common)	Paddy fine	FAQ	1600	1750	1750
84	Chattisgarh	Narayanpur	Narayanpur	Millets	Other	FAQ	3000	3800	3500
85	Chattisgarh	Raigarh	Raigarh	Onion	Other	FAQ	1800	2000	1800
86	Chattisgarh	Raigarh	Raigarh	Potato	Other	FAQ	1200	1800	1500
87	Chattisgarh	Raigarh	Raigarh	Tomato	Other	FAQ	8000	9000	8000
88	Chattisgarh	Raipur	Abhanpur	Paddy(Dhan)(Common)	I.R.-64	FAQ	1850	1850	1850
89	Goa	North Goa	Valpol	Arecanut(Betelnut/Supari)	Red	FAQ	33500	33500	33500
90	Goa	North Goa	Valpol	Arecanut(Betelnut/Supari)	Supari	FAQ	36500	36500	36500
91	Goa	North Goa	Valpol	Arecanut(Betelnut/Supari)	White	FAQ	34500	34500	34500
92	Gujarat	Amreli	Babra	Cotton	Other	FAQ	7050	7100	7075
93	Gujarat	Amreli	Babra	Groundnut	Other	FAQ	6500	6700	6600
94	Gujarat	Amreli	Bagasara	Bengal Gram(Gram)(Whole)	Average (Whole)	FAQ	4460	4535	4495
95	Gujarat	Amreli	Bagasara	Cotton	Other	FAQ	6250	6825	6535
96	Gujarat	Amreli	Bagasara	Soyabean	Yellow	FAQ	4560	4700	4630
97	Gujarat	Amreli	Bagasara	Wheat	Lokwan Gujrat	FAQ	2250	2285	2265
98	Gujarat	Amreli	Dhari	Cotton	Other	FAQ	7140	7140	7140
99	Gujarat	Amreli	Dhari	Ground Nut Seed	Ground Nut Seed	FAQ	8250	8250	8250
100	Gujarat	Amreli	Savarkundla	Bajra(Pearl Millet/Cumbu)	Deshi	FAQ	1900	2120	2010

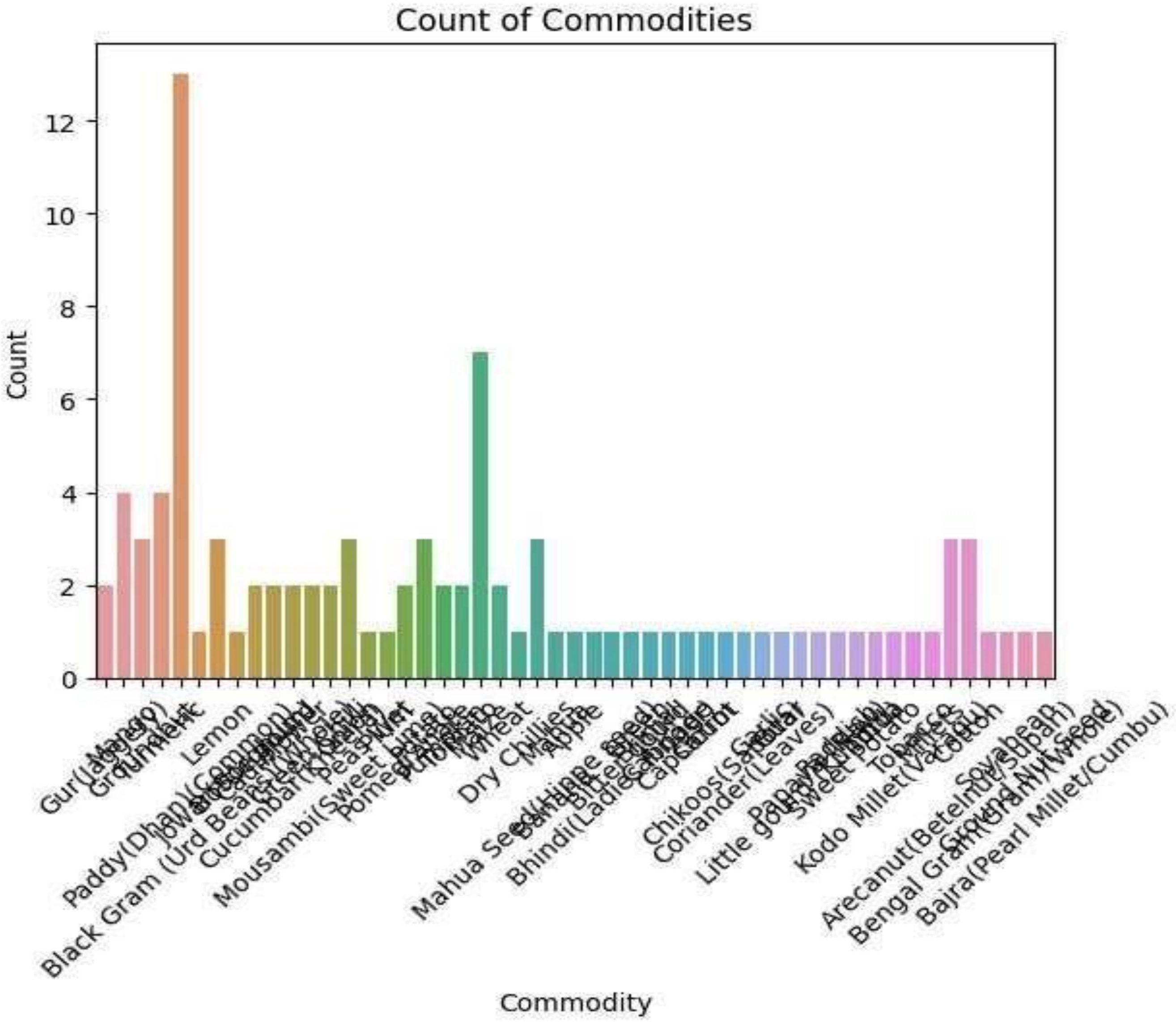
```
import seaborn as sns import
matplotlib.pyplot as plt import pandas
as pd

# Load the CSV file into a DataFrame df = pd.read_csv("/content/daily
price.csv")

# Statement 1: Distribution of Modal Prices
sns.histplot(data=df, x='Modal Price') plt.xlabel('Modal
Price') plt.ylabel('Count') plt.title('Distribution of
Modal Prices') plt.show()

# Statement 2: Count of Each Commodity
sns.countplot(data=df, x='Commodity')
plt.xlabel('Commodity') plt.ylabel('Count')
plt.title('Count of Commodities') plt.xticks(rotation=45)
plt.show()

# Statement 3: Average Min and Max Prices by State
```

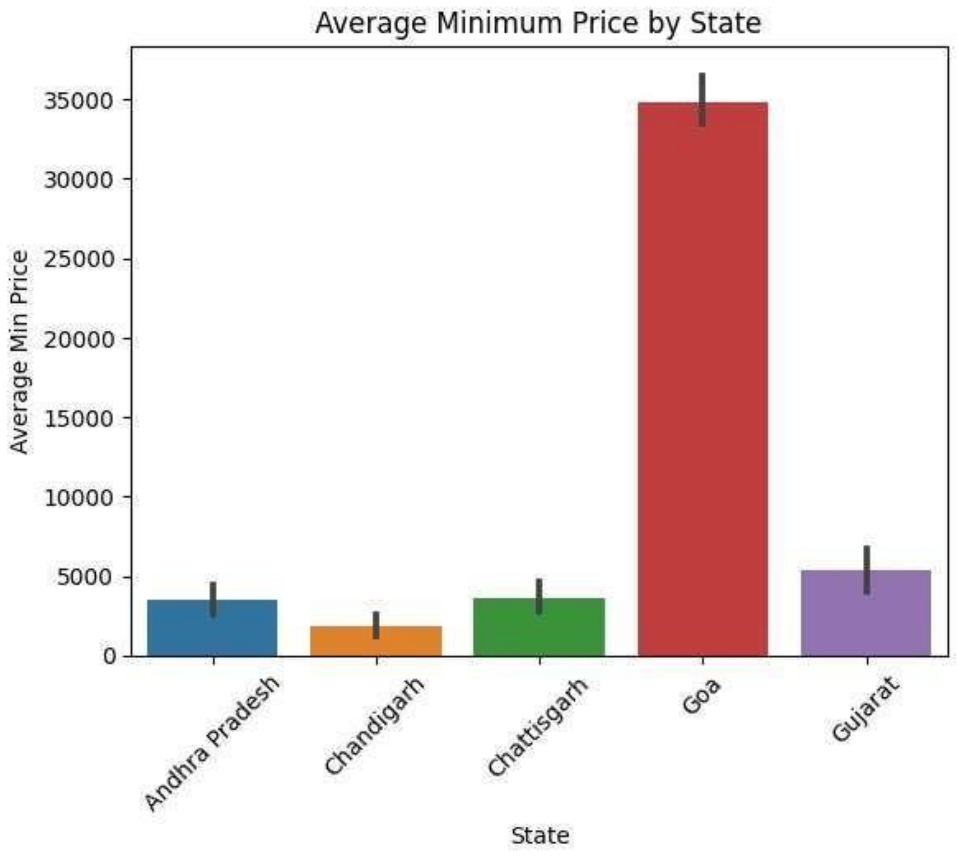
```
sns.barplot(data=df, x='State', y='Min Price')
plt.xlabel('State') plt.ylabel('Average Min
Price') plt.title('Average Minimum Price by
State') plt.xticks(rotation=45) plt.show()
sns.barplot(data=df, x='State', y='Max Price')
plt.xlabel('State') plt.ylabel('Average Max
Price') plt.title('Average Maximum Price by
State') plt.xticks(rotation=45) plt.show()

# Statement 4: Relationship between Min Price and Max Price
sns.scatterplot(data=df, x='Min Price', y='Max Price')
plt.xlabel('Min Price') plt.ylabel('Max Price') plt.title('Min
Price vs Max Price') plt.show()

# Statement 5: Average Modal Price by Market
sns.barplot(data=df, x='Market', y='Modal Price')
plt.xlabel('Market') plt.ylabel('Average Modal Price')
plt.title('Average Modal Price by
Market') plt.xticks(rotation=45) plt.show()

# Statement 6: Modal Price Comparison across Grades
sns.boxplot(data=df, x='Grade', y='Modal Price')
plt.xlabel('Grade') plt.ylabel('Modal Price')
plt.title('Modal Price by Grade') plt.xticks(rotation=45)
plt.show()

# Statement 7: Modal Price Distribution for Each Variety
sns.violinplot(data=df, x='Variety', y='Modal Price')
plt.xlabel('Variety') plt.ylabel('Modal Price') plt.title('Modal
Price Distribution by Variety') plt.xticks(rotation=45)
plt.show()
```

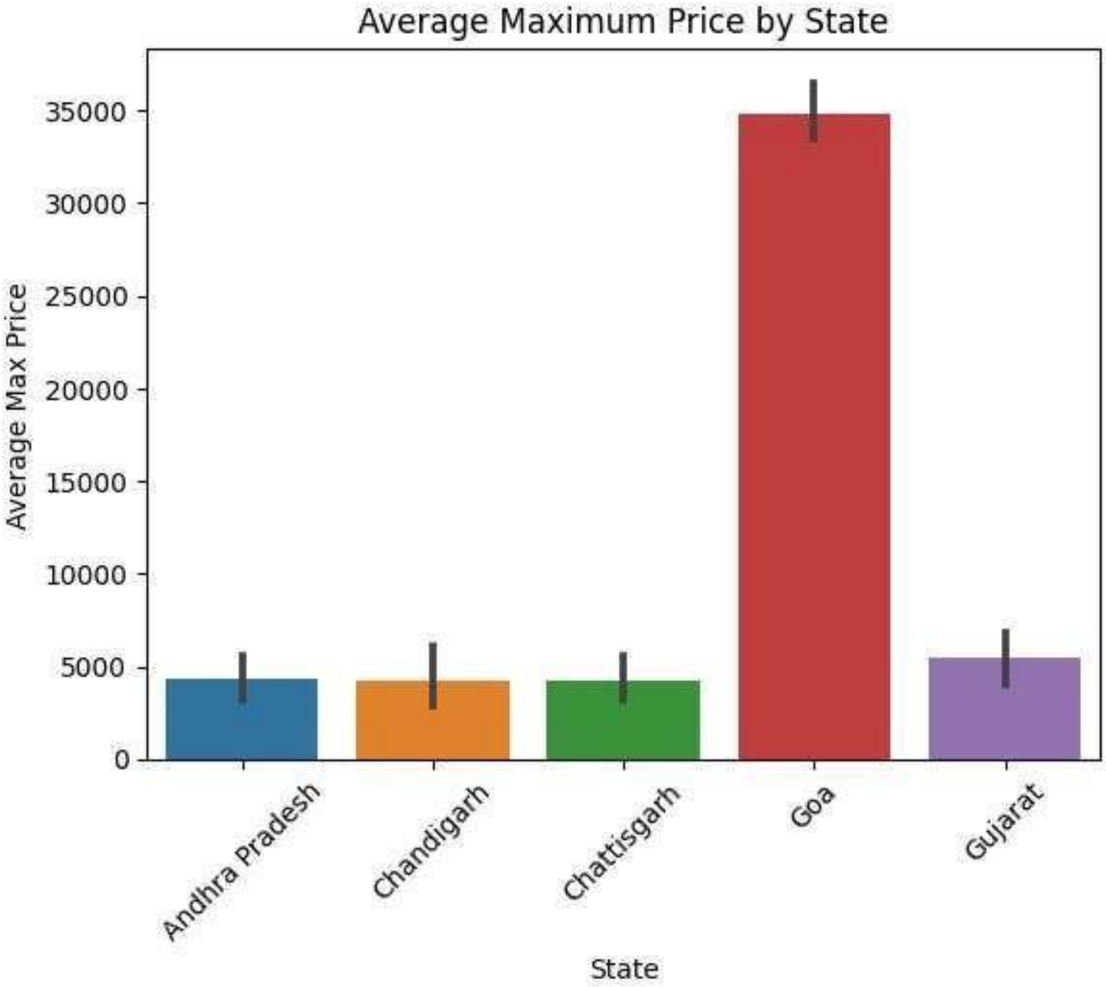
```
# Statement 8: Modal Price Trend by District
sns.lineplot(data=df, x='District', y='Modal Price')
plt.xlabel('District') plt.ylabel('Modal Price') plt.title('Modal
Price Trend by District') plt.xticks(rotation=45) plt.show()

# Statement 9: Average Modal Price by Commodity
sns.barplot(data=df, x='Commodity', y='Modal Price')
plt.xlabel('Commodity') plt.ylabel('Average Modal Price')
plt.title('Average Modal Price by
Commodity') plt.xticks(rotation=45) plt.show()

# Statement 10: Count of Each District
sns.countplot(data=df, x='District')
plt.xlabel('District')
plt.ylabel('Count') plt.title('Count of
Districts') plt.xticks(rotation=45)
plt.show()

# Statement 11: Modal Price Distribution by State and Market
sns.boxplot(data=df, x='State', y='Modal Price', hue='Market')
plt.xlabel('State') plt.ylabel('Modal Price') plt.title('Modal Price
Distribution by State and Market')
plt.xticks(rotation=45) plt.legend(title='Market') plt.show()

# Statement 12: Modal Prices for Each Variety by Market
sns.barplot(data=df, x='Variety', y='Modal Price', hue='Market')
plt.xlabel('Variety') plt.ylabel('Modal Price') plt.title('Modal
Price by Variety and Market') plt.xticks(rotation=45)
plt.legend(title='Market') plt.show()

# Statement 13: Average Modal Price by Commodity and Grade
sns.barplot(data=df, x='Commodity', y='Modal Price', hue='Grade')
plt.xlabel('Commodity')
plt.ylabel('Average Modal Price') plt.title('Average Modal
Price by Commodity and Grade')
plt.xticks(rotation=45) plt.legend(title='Grade') plt.show()

# Statement 14: Modal Price Distribution by State and Commodity
sns.violinplot(data=df, x='State', y='Modal Price', hue='Commodity')
plt.xlabel('State') plt.ylabel('Modal Price') plt.title('Modal Price
Distribution by State and Commodity') plt.xticks(rotation=45)
plt.legend(title='Commodity') plt.show()

# Statement 15: Relationship between Modal Price and Market by District
sns.scatterplot(data=df, x='Modal Price', y='Market', hue='District')
plt.xlabel('Modal Price') plt.ylabel('Market') plt.title('Modal Price vs
Market by District') plt.legend(title='District') plt.show()
```

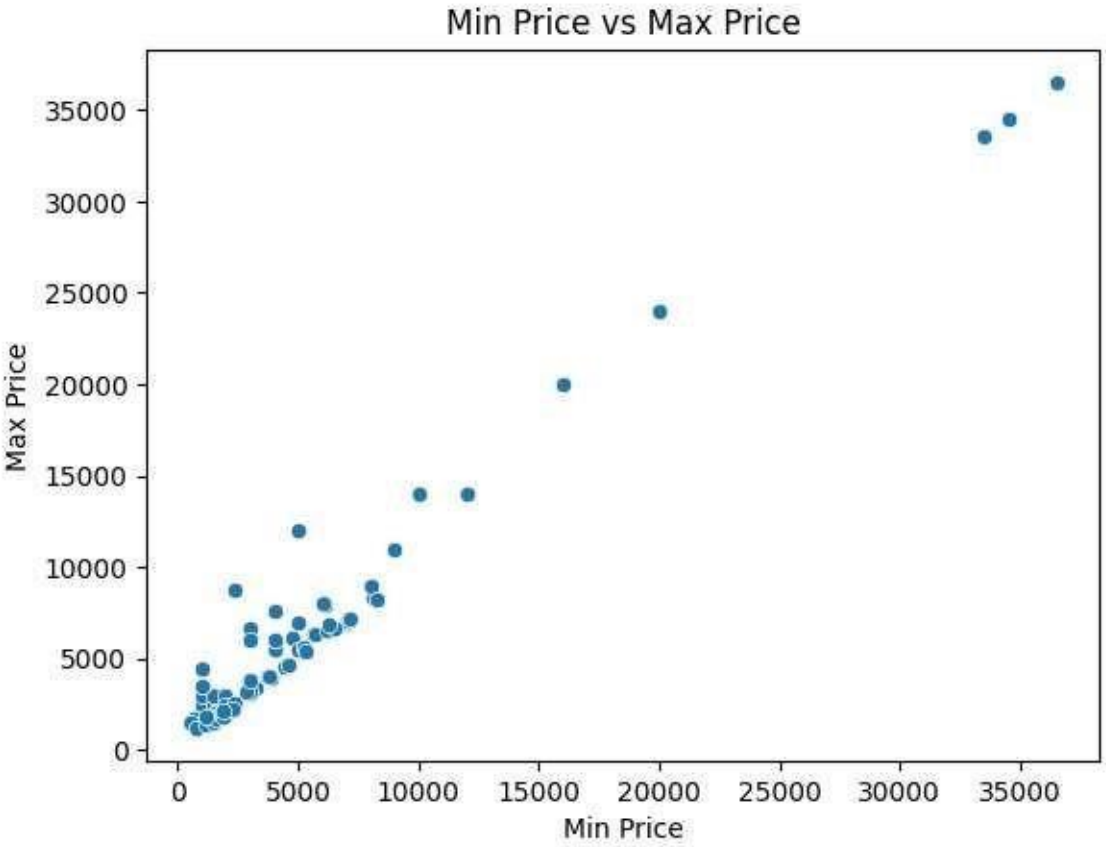
output :-

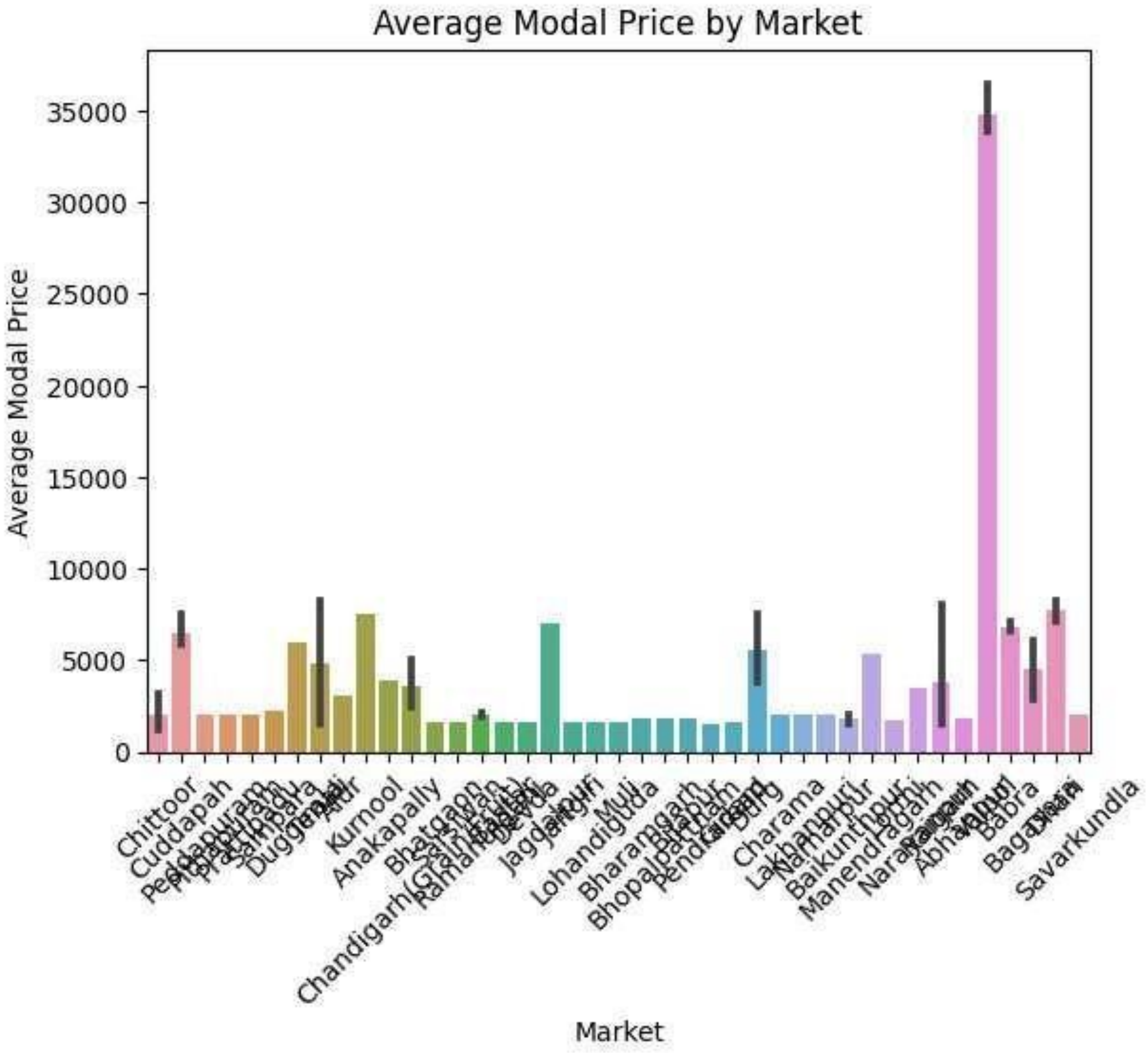


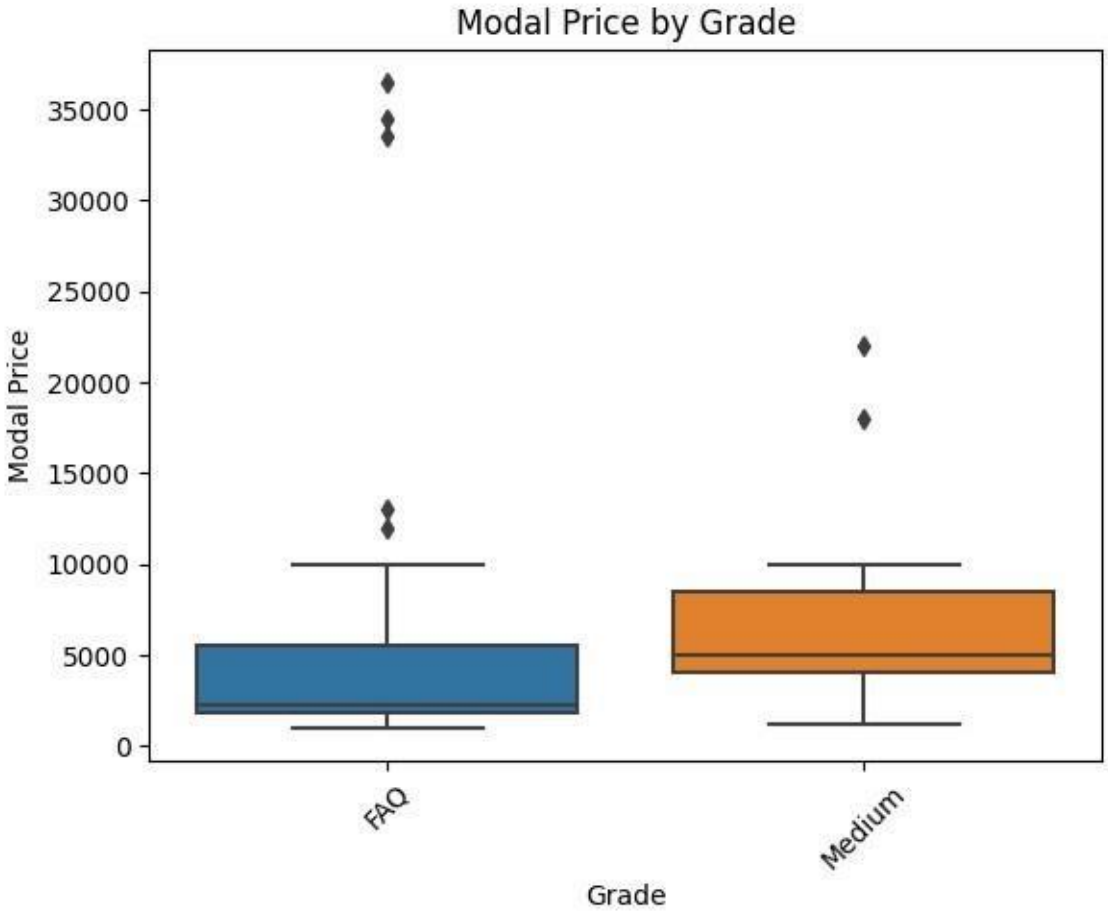

Count of Commodities
Count
0
2
4
6
8
10
12
Commodity
Gur(Jaggery)
Mango
Groundnut
Turmeric
Lemon
Paddy(Dhan)(Common)
Jowar
Black Gram (Urd Beans)(Whole)
Cucumbar(Kheera)
Peas Wet
Mousambi(Sweet Lime)
Pomegranate
Potato
Tomato
Maize
Wheat
Dry Chillies
Apple
Mahua Seed(Hippe seed)
Banana
Bitter gourd
Bhindi(Ladies Finger)
Cabbage
Capsicum
Carrot
Chikoos(Sapota)
Garlic
Coriander(Leaves)
Little gourd (Kundru)
Papaya
Radish
Sweet Potato
Tobacco
Kodo Millet(Varagu)
Cotton
Arecanut(Betelnut/Supari)
Soyabean
Ground Nut Seed
Bengal Gram(Gram)(Whole)
Bajra(Pearl Millet/Cumbu)

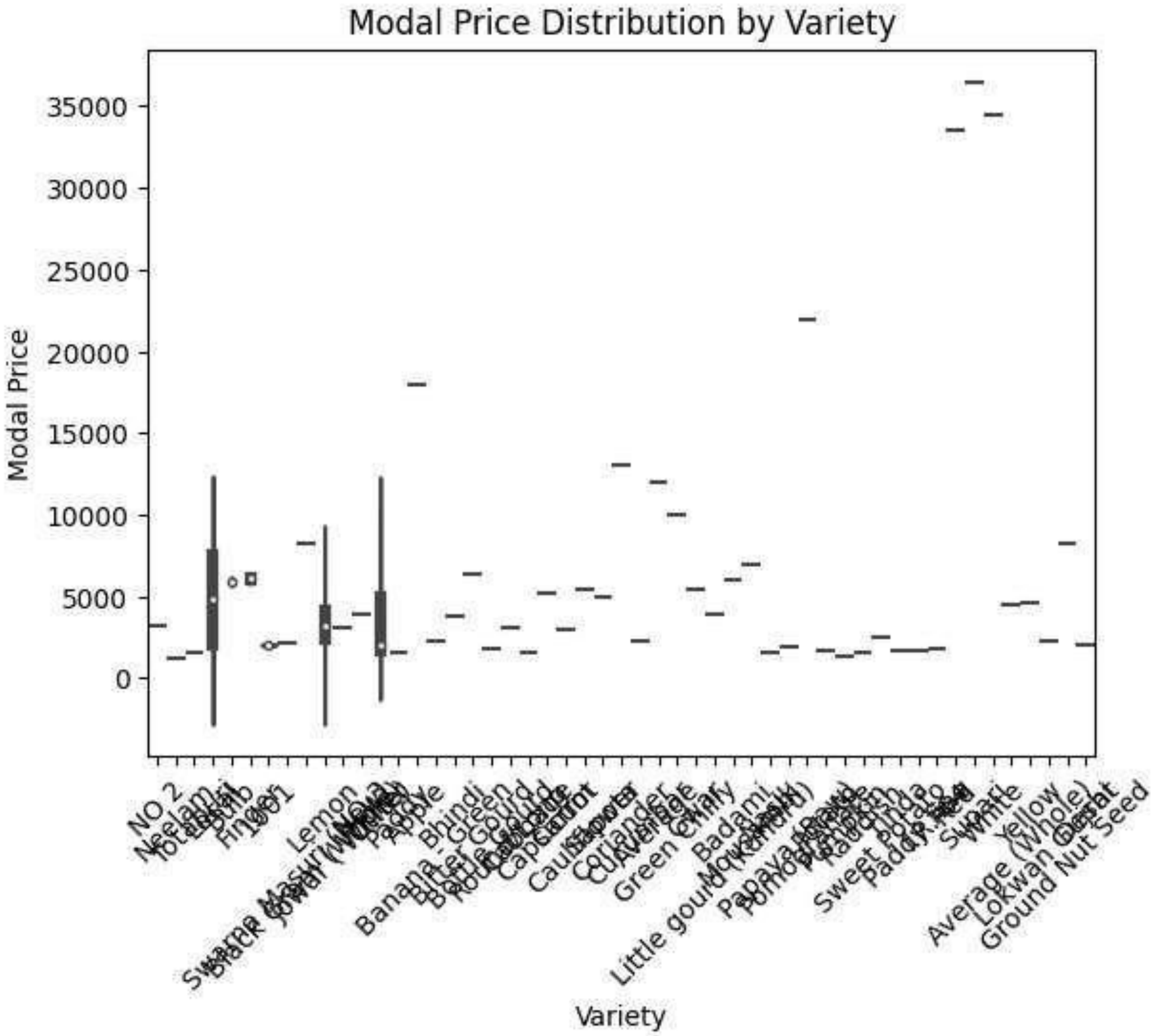

Average Minimum Price by State
Average Min Price
35000
30000
25000
20000
15000
10000
5000
0
Andhra Pradesh
Chandigarh
Chattisgarh
Goa
Gujarat
State

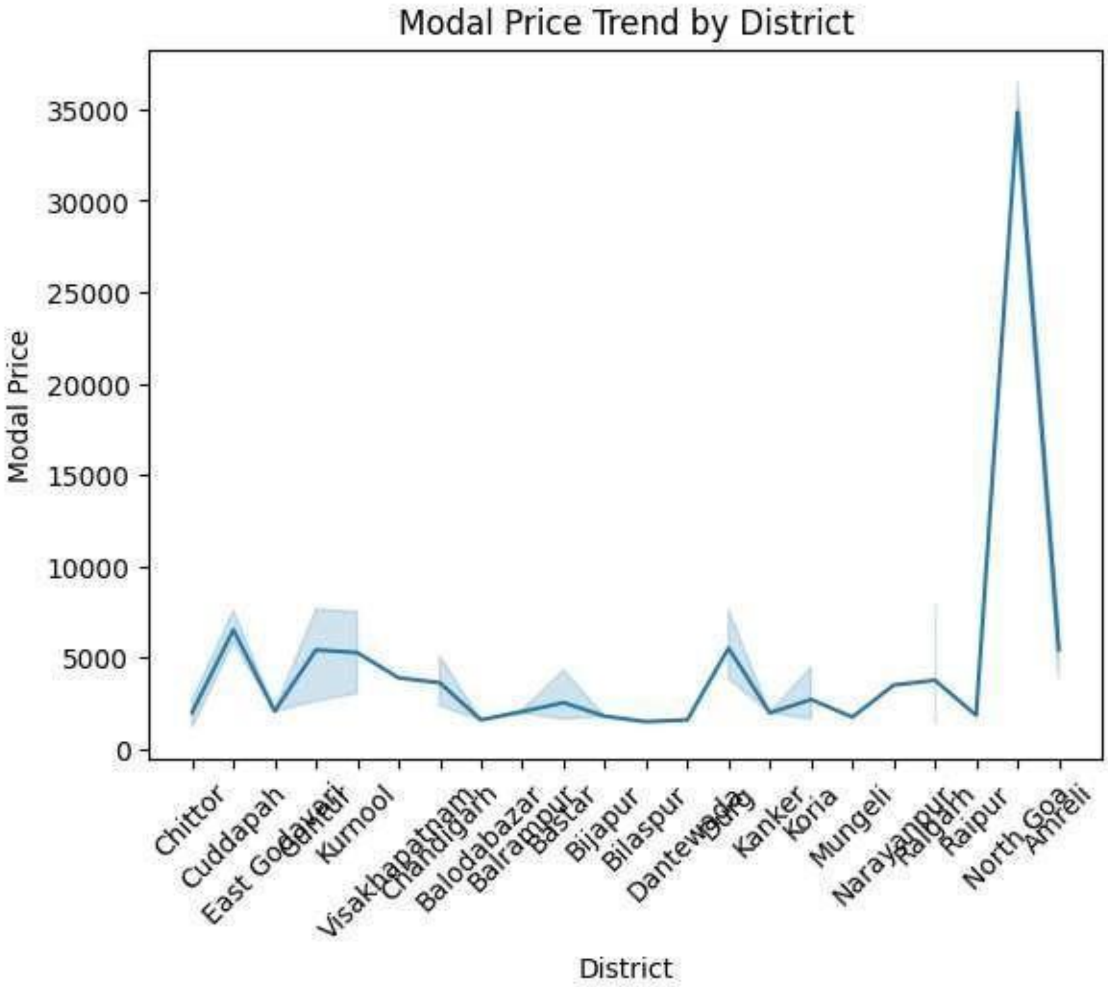

Average Maximum Price by State
Average Max Price
35000
30000
25000
20000
15000
10000
5000
0
Andhra Pradesh
Chandigarh
Chattisgarh
Goa
Gujarat
State

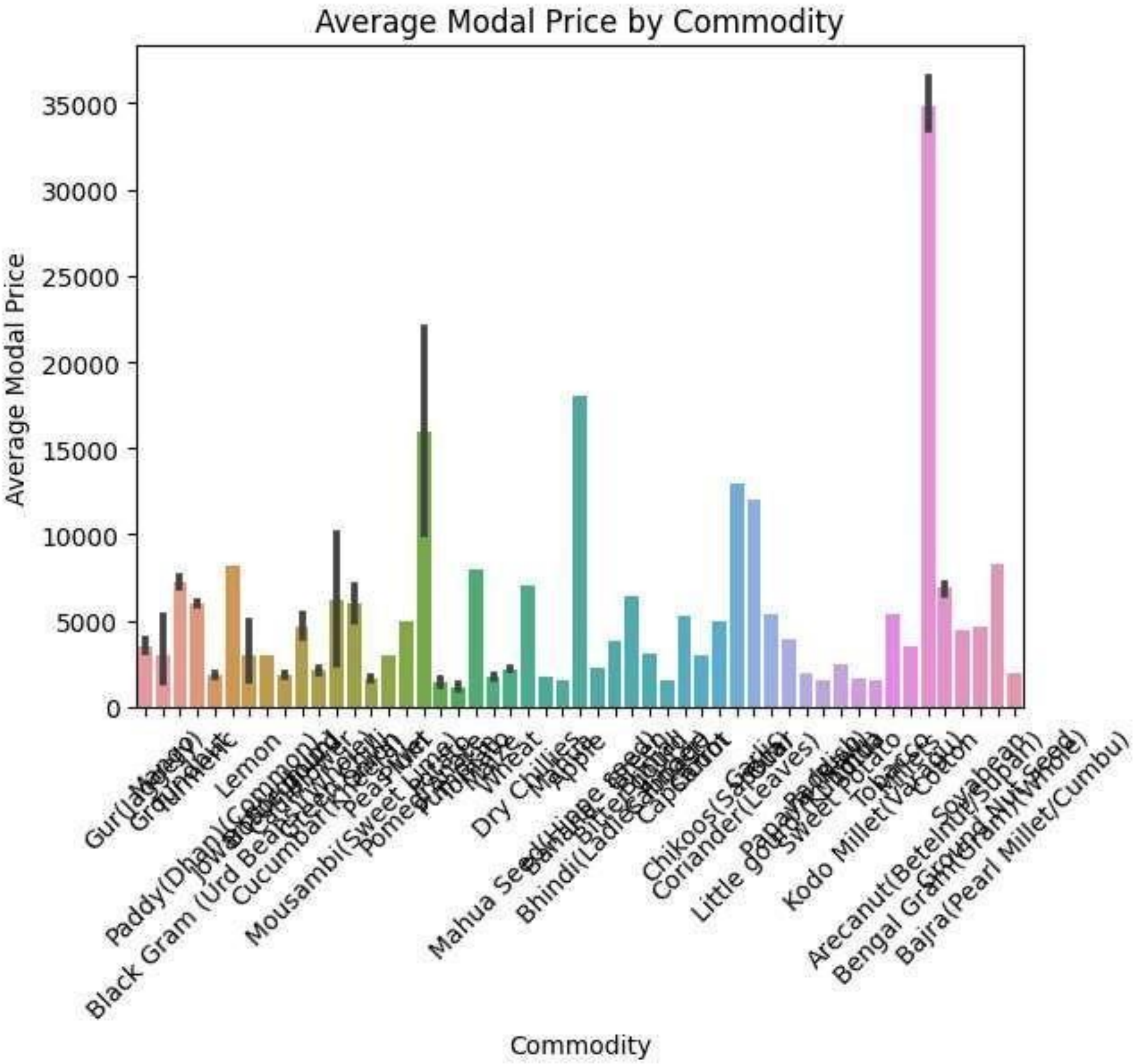

Min Price vs Max Price
Max Price
Min Price
0
5000
10000
15000
20000
25000
30000
35000
0
5000
10000
15000
20000
25000
30000
35000

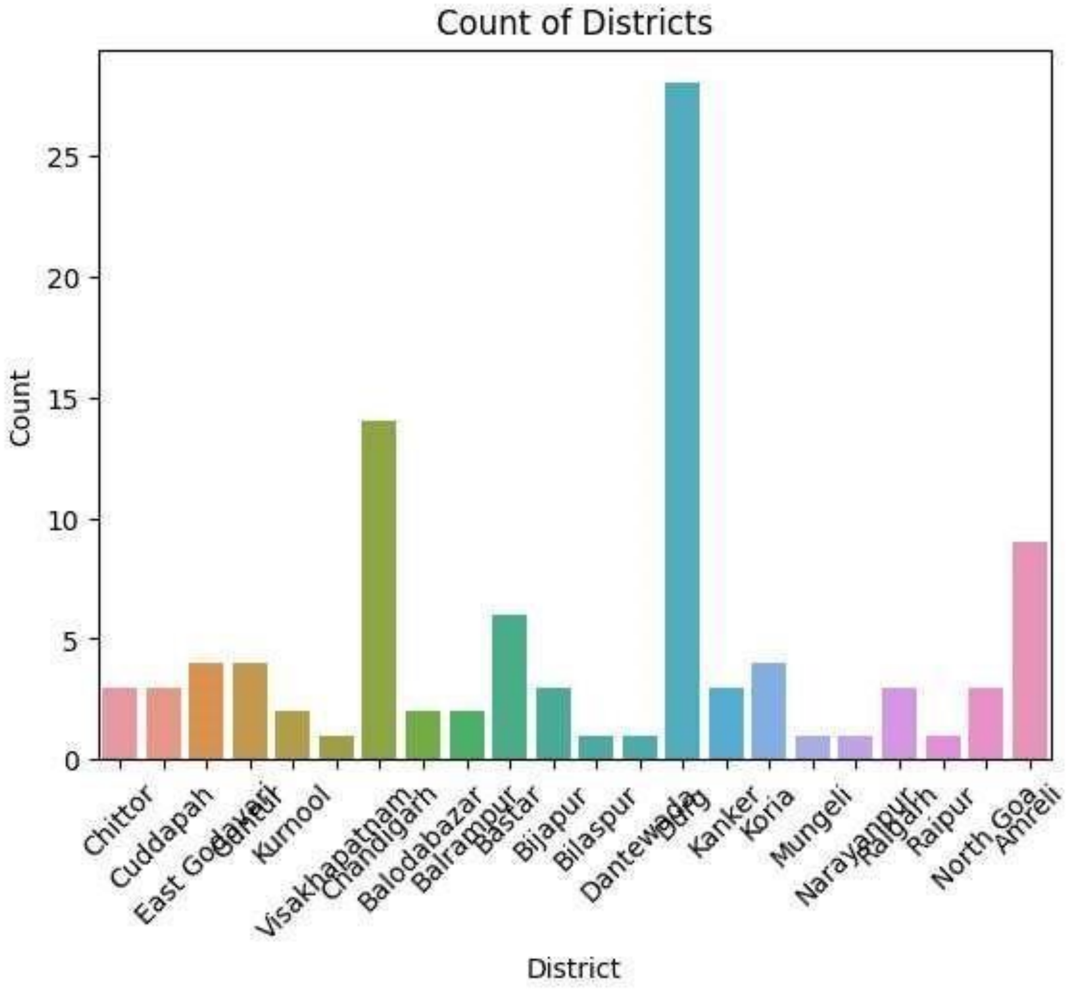

Average Modal Price by Market
Average Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Chittoor
Cuddapah
Duggirala
Kurnool
Anakapally
Bhatgaon
Chandigarh(Grain/Fruit)
Jagdalpur
Muli
Lohandiguda
Bharamgarh
Bhopalpatnam
Durg
Charama
Lakhanpur
Baikunthpur
Manendragarh
Narayanpur
Abhanpur
Babra
Savarkundla
Market

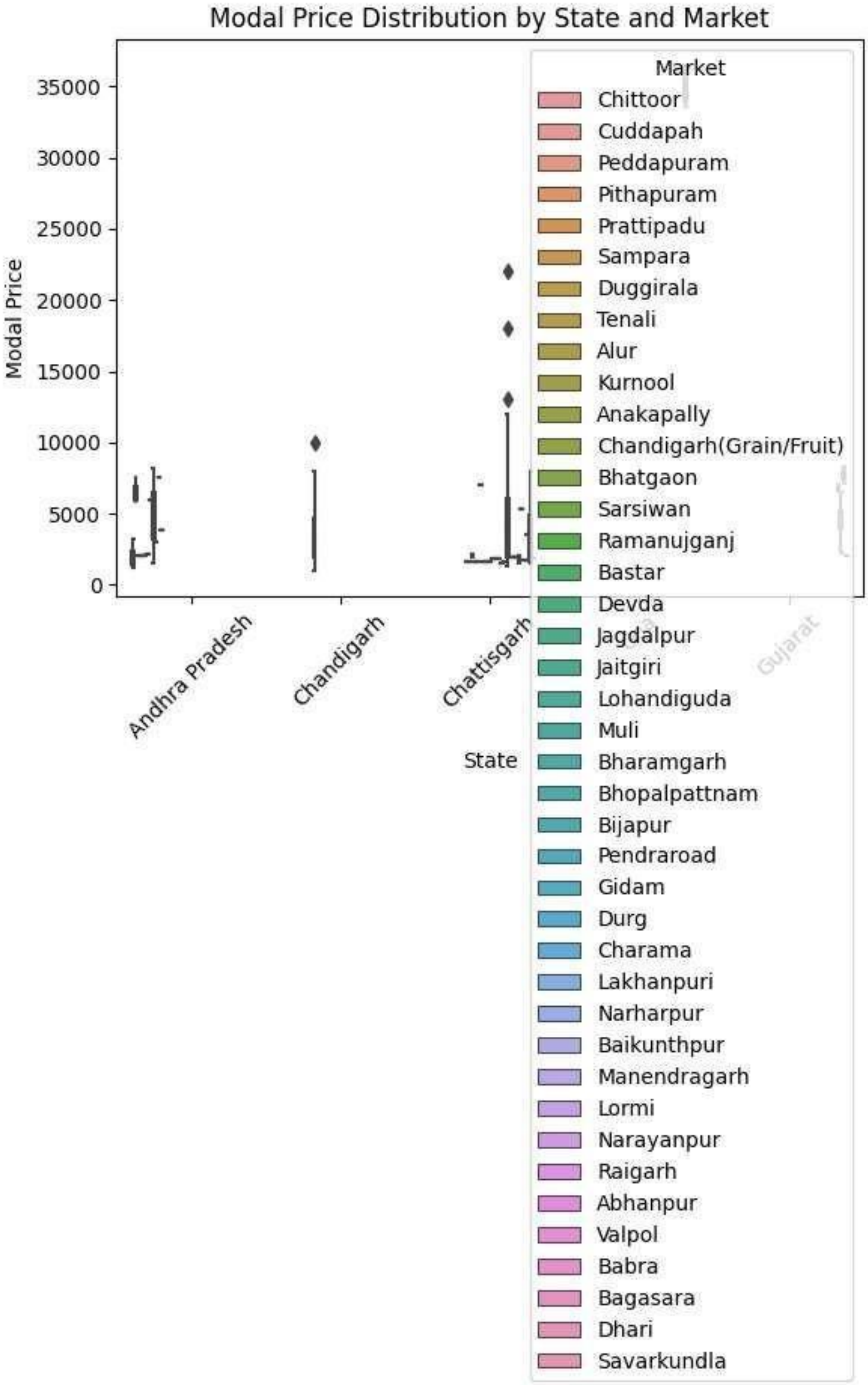

Modal Price by Grade
Modal Price
35000
30000
25000
20000
15000
10000
5000
0
FAQ
Medium
Grade

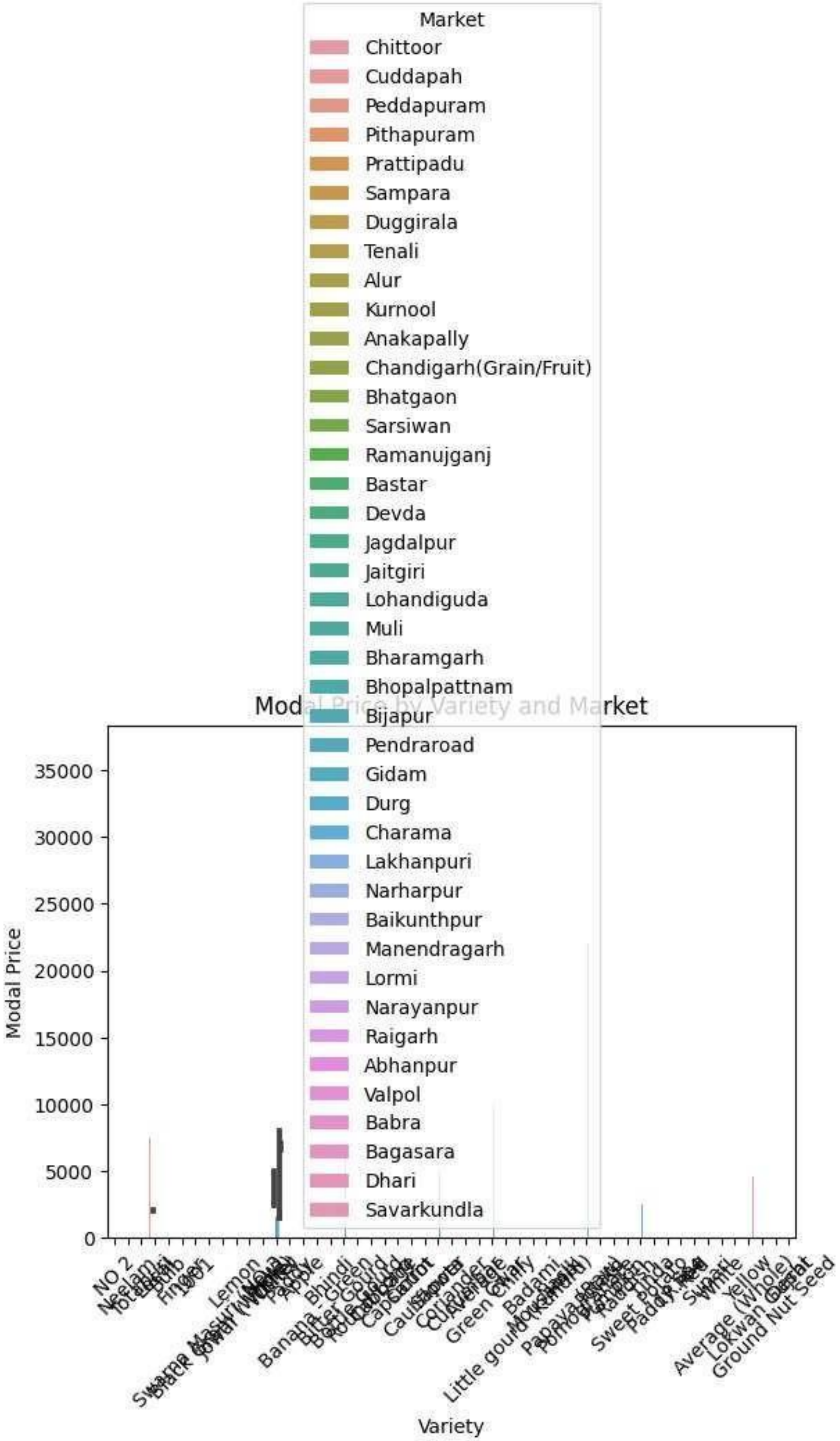

Modal Price Distribution by Variety
Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Variety
NO 2
Neelam
Local
Bulb
Finger
1001
Lemon
Swarna Masuri
Black Jowar
Paddy
Apple
Bhindi
Banana - Green
Bitter Gourd
Bottle Gourd
Round
Cabbage
Capsicum
Carrot
Cauliflower
Coriander
Average
Green Chilly
Badami
Little gourd (Kundru)
Papaya (Raw)
Pomogranate
Raddish
Sweet Potato
Paddy
Red
Sunari
White
Yellow
Average (Whole)
Lokwan Gehun
Ground Nut Seed

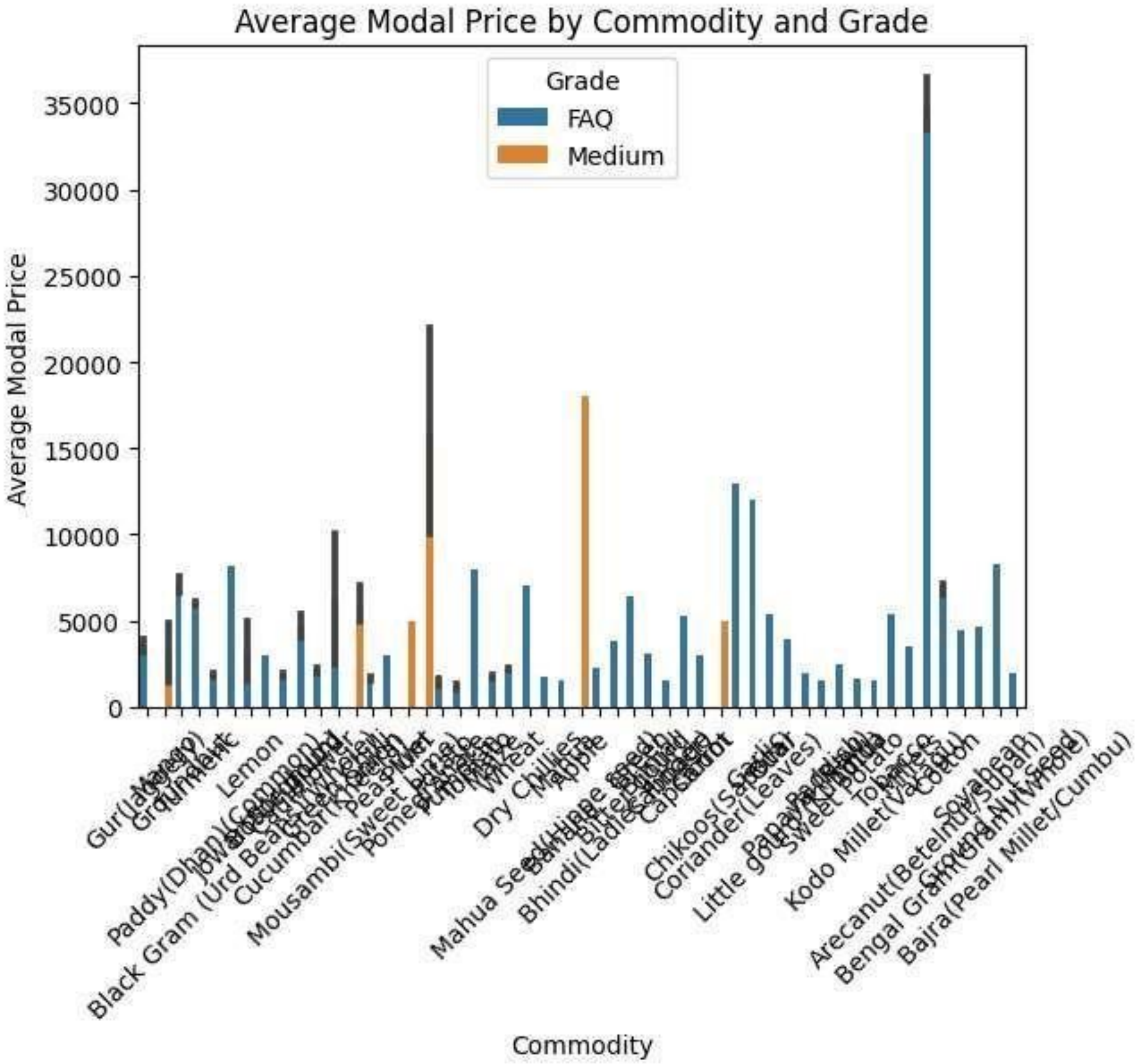

Modal Price Trend by District
Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Chittor
Cuddapah
East Godavari
Guntur
Kurnool
Visakhapatnam
Chandigarh
Balodabazar
Balrampur
Bastar
Bijapur
Bilaspur
Dantewada
Durg
Kanker
Koria
Mungeli
Narayanpur
Raigarh
Raipur
North Goa
Amreli
District

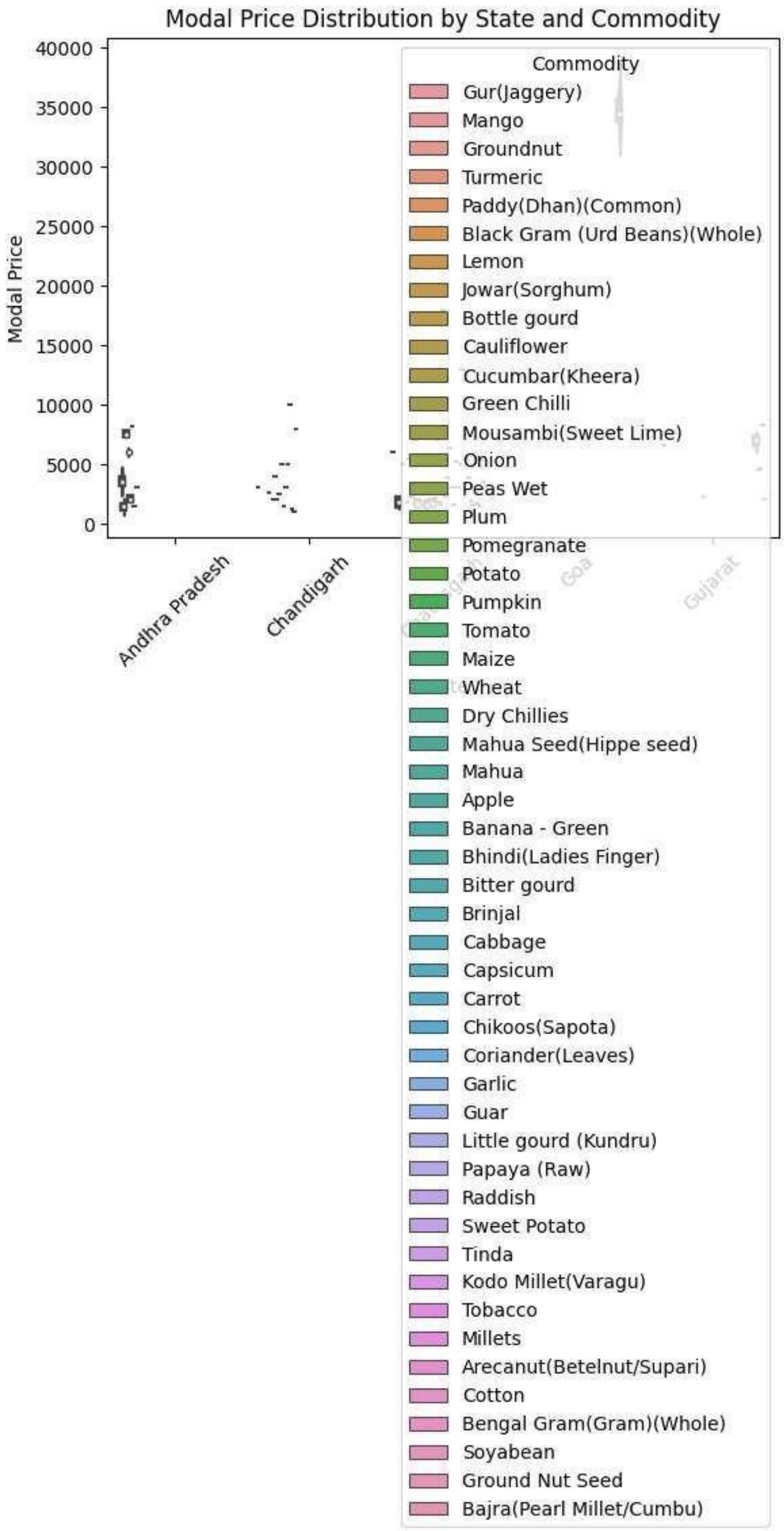

Average Modal Price by Commodity
Average Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Gur(Jaggery)
Mango
Groundnut
Turmeric
Lemon
Paddy(Dhan)(Common)
Jowar
Black Gram (Urd Beans)
Cucumbar(Kheera)
Peas
Mousambi(Sweet Lime)
Pomegranate
Pumpkin
Maize
Wheat
Dry Chillies
Mango
Apple
Mahua Seed(Hippe seed)
Banana
Bhindi(Ladies Finger)
Capsicum
Carrot
Chikoos(Sapota)
Garlic
Coriander(Leaves)
Little gourd(Kundru)
Papaya
Sweet Potato
Tobacco
Kodo Millet(Varagu)
Cotton
Arecanut(Betelnut/Supari)
Soyabean
Bengal Gram(Gram)(Whole)
Ground Nut Seed
Bajra(Pearl Millet/Cumbu)
Commodity

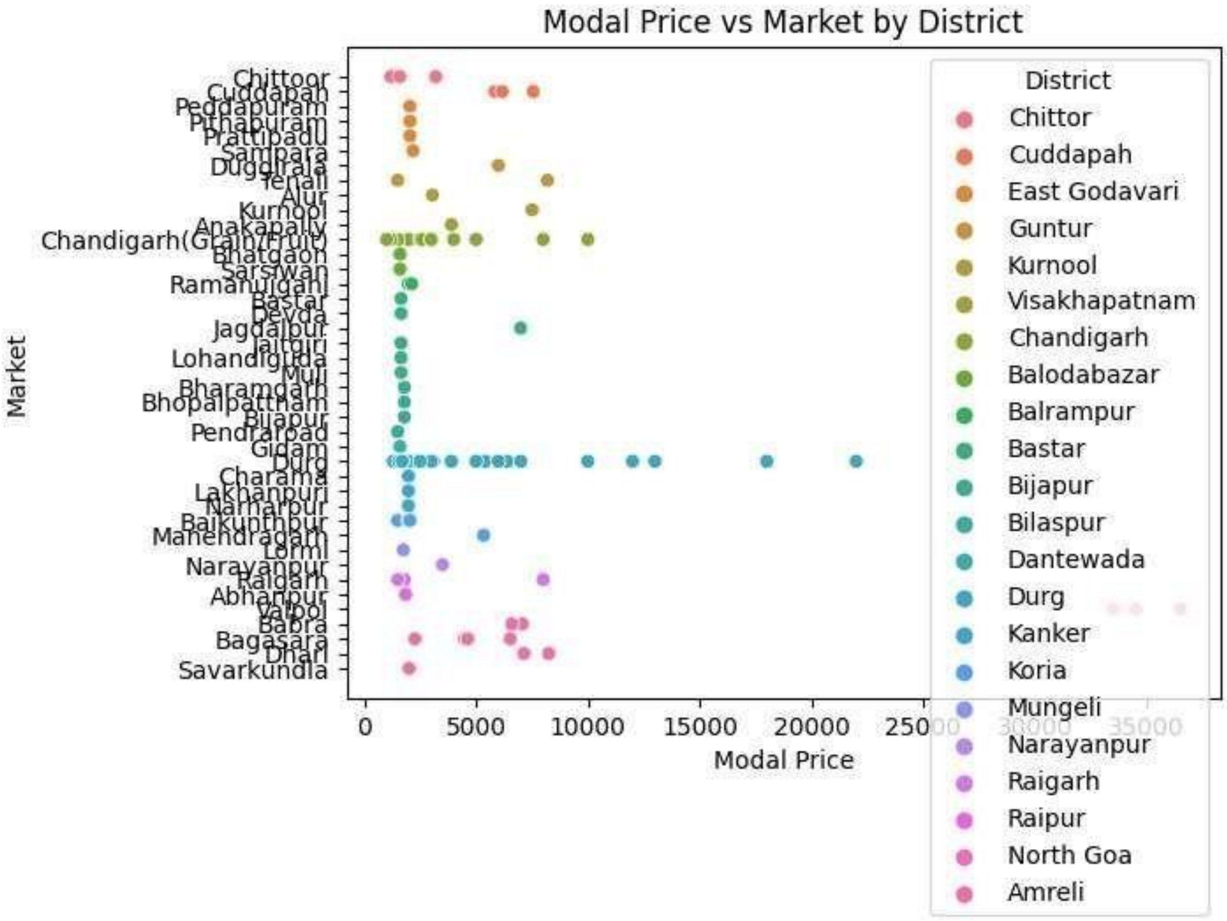

Count of Districts
Count
0
5
10
15
20
25
District
Chittor
Cuddapah
East Godavari
Guntur
Kurnool
Visakhapatnam
Chandigarh
Balodabazar
Balrampur
Bastar
Bijapur
Bilaspur
Dantewada
Durg
Kanker
Koria
Mungeli
Narayanpur
Raigarh
Raipur
North Goa
Amreli


Modal Price Distribution by State and Market
Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Andhra Pradesh
Chandigarh
Chattisgarh
Gujarat
State
Market
Chittoor
Cuddapah
Peddapuram
Pithapuram
Prattipadu
Sampara
Duggirala
Tenali
Alur
Kurnool
Anakapally
Chandigarh(Grain/Fruit)
Bhatgaon
Sarsiwan
Ramanujganj
Bastar
Devda
Jagdalpur
Jaitgiri
Lohandiguda
Muli
Bharamgarh
Bhopalpattnam
Bijapur
Pendraroad
Gidam
Durg
Charama
Lakhanpuri
Narharpur
Baikunthpur
Manendragarh
Lormi
Narayanpur
Raigarh
Abhanpur
Valpol
Babra
Bagasara
Dhari
Savarkundla

Market
Chittoor
Cuddapah
Peddapuram
Pithapuram
Prattipadu
Sampara
Duggirala
Tenali
Alur
Kurnool
Anakapally
Chandigarh(Grain/Fruit)
Bhatgaon
Sarsiwan
Ramanujganj
Bastar
Devda
Jagdalpur
Jaitgiri
Lohandiguda
Muli
Bharamgarh
Bhopalpattnam
Bijapur
Pendraroad
Gidam
Durg
Charama
Lakhanpuri
Narharpur
Baikunthpur
Manendragarh
Lormi
Narayanpur
Raigarh
Abhanpur
Valpol
Babra
Bagasara
Dhari
Savarkundla
Modal Price by Variety and Market
Modal Price
35000
30000
25000
20000
15000
10000
5000
0
NO 2
Neelam
Totapuri
Local
Bulb
Finger
1001
Lemon
Swarna Masuri (New)
Black Jowar (Whole)
Paddy
Apple
Bhindi
Banana - Green
Bitter Gourd
Bottle Gourd
Round
Capsicum
Carrot
Cauliflower
Sapota
Coriander
Green Chilly
Badami
Little gourd (Kundru)
Papaya (Raw)
Pomogranate
Raddish
Sweet Potato
Paddy
Red
Supari
White
Yellow
Average (Whole)
Lokwan Gujrat
Desi
Ground Nut Seed
Variety


Average Modal Price by Commodity and Grade
Grade
FAQ
Medium
Average Modal Price
35000
30000
25000
20000
15000
10000
5000
0
Gur(Jaggery)
Mango
Groundnut
Turmeric
Lemon
Paddy(Dhan)(Common)
Jowar
Black Gram (Urd Beans)
Cucumbar(Kheera)
Mousambi(Sweet Lime)
Peas Wet
Pomegranate
Pumpkin
Maize
Wheat
Dry Chillies
Apple
Mahua Seed(Hippe seed)
Banana
Bitter gourd
Bhindi(Ladies Finger)
Cabbage
Capsicum
Carrot
Chikoos(Sapota)
Garlic
Coriander(Leaves)
Papaya
Little gourd(Kundru)
Sweet Potato
Tomato
Tobacco
Kodo Millet(Varagu)
Cotton
Arecanut(Betelnut/Supari)
Soyabean
Ground Nut Seed
Bengal Gram(Gram)(Whole)
Bajra(Pearl Millet/Cumbu)
Commodity


Modal Price Distribution by State and Commodity
Modal Price
40000
35000
30000
25000
20000
15000
10000
5000
0
Andhra Pradesh
Chandigarh
Goa
Gujarat
Commodity
Gur(Jaggery)
Mango
Groundnut
Turmeric
Paddy(Dhan)(Common)
Black Gram (Urd Beans)(Whole)
Lemon
Jowar(Sorghum)
Bottle gourd
Cauliflower
Cucumbar(Kheera)
Green Chilli
Mousambi(Sweet Lime)
Onion
Peas Wet
Plum
Pomegranate
Potato
Pumpkin
Tomato
Maize
Wheat
Dry Chillies
Mahua Seed(Hippe seed)
Mahua
Apple
Banana - Green
Bhindi(Ladies Finger)
Bitter gourd
Brinjal
Cabbage
Capsicum
Carrot
Chikoos(Sapota)
Coriander(Leaves)
Garlic
Guar
Little gourd (Kundru)
Papaya (Raw)
Raddish
Sweet Potato
Tinda
Kodo Millet(Varagu)
Tobacco
Millets
Arecanut(Betelnut/Supari)
Cotton
Bengal Gram(Gram)(Whole)
Soyabean
Ground Nut Seed
Bajra(Pearl Millet/Cumbu)


Modal Price vs Market by District
Market
Chittoor
Cuddapah
Peddapuram
Pithapuram
Prattipadu
Sampara
Duggirala
Tenali
Alur
Kurnool
Anakapally
Chandigarh(Grain/Fruit)
Bhatgaon
Sarsiwan
Ramanujganj
Bastar
Devda
Jagdalpur
Jaitgiri
Lohandiguda
Muli
Bharamdarh
Bhopalpattnam
Bijapur
Pendraroad
Gidam
Durg
Charama
Lakhanpuri
Narharpur
Baikunthpur
Manendragarh
Lormi
Narayanpur
Raigarh
Abhanpur
Valpoi
Babra
Bagasara
Dhari
Savarkundla
0
5000
10000
15000
20000
25000
30000
35000
Modal Price
District
Chittor
Cuddapah
East Godavari
Guntur
Kurnool
Visakhapatnam
Chandigarh
Balodabazar
Balrampur
Bastar
Bijapur
Bilaspur
Dantewada
Durg
Kanker
Koria
Mungeli
Narayanpur
Raigarh
Raipur
North Goa
Amreli

assignment2

May 27, 2023

[ ]:

```
product_details=[]
supplier_details=dict()
customer_details=[]
gender={}
fp1= open("sales.csv","r")
data=fp1.readline()
while(True):
  data=fp1.readline()
  if not data:
    break
  print(data)
  data= data.replace("\n","")
  temp= data.split(",")
  product_details.append(temp[1])
  customer_details.append(temp[3])
  supplier_details.update({temp[0]:temp[2]})
  gender.update({temp[3]:temp[4]})
```

```
P00001,Lenovo laptop,Raka Ele.,Kaustoobh Mahajan,male

P00002,Samsung Laptop,Vijay Sales,Siddhi kivale,female

P00003,Realmi 10pro,Gada Ele.,Sanket Kandalkar,male

P00004,Oppo f21,Surya Ele.,Yash mali,male

P00005,Lenovo laptop,Raka Ele.,Yash Bagul,male

P00006,Samsung M31,Gada Ele.,Siddhi kivale,female

P00007,LG TV 32*,Vijay Sales,Sanket Kandalkar,male

P00008,Oppo f21,Surya Ele.,Kaustoobh Mahajan,male

P00009,Lenovo laptop,Raka Ele.,Yash mali,male

P00010,Samsung M31,Gada Ele.,Siddhi kivale,female
```

```
P00011,LG TV 32*,Surya Ele.,Sanket Kandalkar,male

P00012,Lenovo laptop,Raka Ele.,Kaustoobh Mahajan,male

P00013,Samsung M31,Surya Ele.,Yash mali,male

P00014,Realmi 10pro,Raka Ele.,Siddhi kivale,female

P00015,Lenovo laptop,Gada Ele.,Tanuja Mali,female

P00016,Oppo f21,Vijay Sales,Kaustoobh Mahajan,male

P00017,LG TV 32*,Deshmukh Sales,Sanket Kandalkar,male

P00018,Lenovo laptop,Raka Ele.,Siddhi kivale,female

P00019,Samsung M21,Deshmukh Sales,Kaustoobh Mahajan,male

P00020,LG TV 32*,Gada Ele.,Yash mali,male
```

```
[ ]: fp1.close()
```

```
[ ]: customer_details= tuple(customer_details)
     print(type(customer_details))
     print("\nproduct_details\n",product_details,end='')
     print("\ncustomer_details\n",customer_details,end='')
     print("\nsupplier_details\n",supplier_details,end='')
     print("\ngender\n",gender,end='')
```

```
<class 'tuple'>

product_details
 ['Lenovo laptop', 'Samsung Laptop', 'Realmi 10pro', 'Oppo f21', 'Lenovo
laptop', 'Samsung M31', 'LG TV 32*', 'Oppo f21', 'Lenovo laptop', 'Samsung M31',
'LG TV 32*', 'Lenovo laptop', 'Samsung M31', 'Realmi 10pro', 'Lenovo laptop',
'Oppo f21', 'LG TV 32*', 'Lenovo laptop', 'Samsung M21', 'LG TV 32*']
customer_details
 ('Kaustoobh Mahajan', 'Siddhi kivale', 'Sanket Kandalkar', 'Yash mali', 'Yash
Bagul', 'Siddhi kivale', 'Sanket Kandalkar', 'Kaustoobh Mahajan', 'Yash mali',
'Siddhi kivale', 'Sanket Kandalkar', 'Kaustoobh Mahajan', 'Yash mali', 'Siddhi
kivale', 'Tanuja Mali', 'Kaustoobh Mahajan', 'Sanket Kandalkar', 'Siddhi
kivale', 'Kaustoobh Mahajan', 'Yash mali')
supplier_details
 {'P00001': 'Raka Ele.', 'P00002': 'Vijay Sales', 'P00003': 'Gada Ele.',
'P00004': 'Surya Ele.', 'P00005': 'Raka Ele.', 'P00006': 'Gada Ele.', 'P00007':
'Vijay Sales', 'P00008': 'Surya Ele.', 'P00009': 'Raka Ele.', 'P00010': 'Gada
Ele.', 'P00011': 'Surya Ele.', 'P00012': 'Raka Ele.', 'P00013': 'Surya Ele.',
'P00014': 'Raka Ele.', 'P00015': 'Gada Ele.', 'P00016': 'Vijay Sales', 'P00017':
'Deshmukh Sales', 'P00018': 'Raka Ele.', 'P00019': 'Deshmukh Sales', 'P00020':
'Gada Ele.'}
gender
 {'Kaustoobh Mahajan': 'male', 'Siddhi kivale': 'female', 'Sanket Kandalkar':
'male', 'Yash mali': 'male', 'Yash Bagul': 'male', 'Tanuja Mali': 'female'}
```

[ ]:

```python
frequency= {}
for item in product_details:
  if item in frequency:
    frequency[item] += 1
  else:
    frequency[item] = 1
print(frequency)
marklist= sorted(frequency.items(), key=lambda x: x[1],reverse=True)
sortdict = dict(marklist)
print(sortdict)
print('The most popular product for sales',list(sortdict.
  ↪keys())[0],'sold',list(sortdict.values())[0],'times')
```

```
{'Lenovo laptop': 6, 'Samsung Laptop': 1, 'Realmi 10pro': 2, 'Oppo f21': 3,
'Samsung M31': 3, 'LG TV 32*': 4, 'Samsung M21': 1}
{'Lenovo laptop': 6, 'LG TV 32*': 4, 'Oppo f21': 3, 'Samsung M31': 3, 'Realmi
10pro': 2, 'Samsung Laptop': 1, 'Samsung M21': 1}
The most popular product for sales Lenovo laptop sold 6 times
```

[ ]:

```python
from collections import Counter
counter = dict(Counter(list(supplier_details.values())))
sorted_counter = sorted(counter.items(), key= lambda x:x[1],reverse=True)
sorted_counter = dict(sorted_counter)
print('The most popular product for sales',list(sorted_counter.keys())[0],␣
  ↪'sold',list(sorted_counter.values())[0],'Items')
```

```
The most popular product for sales Raka Ele. sold 6 Items
```

[ ]:

```python
frequency= {}
for item in customer_details:
  if item in frequency:
    frequency[item] += 1
  else:
    frequency[item] = 1
print('Frequency is as below:\n',frequency)
marklist= sorted(frequency.items(), key=lambda x: x[1],reverse=True)
sortdict = dict(marklist)
print('\nSorted dict is as below:\n',sortdict)
print('\n\nThe customer who buys most of the products',list(sortdict.
  ↪keys())[0],'buy',list(sortdict.values())[0],'Items')
```

```
Frequency is as below:
 {'Kaustoobh Mahajan': 5, 'Siddhi kivale': 5, 'Sanket Kandalkar': 4, 'Yash
mali': 4, 'Yash Bagul': 1, 'Tanuja Mali': 1}

Sorted dict is as below:
 {'Kaustoobh Mahajan': 5, 'Siddhi kivale': 5, 'Sanket Kandalkar': 4, 'Yash
mali': 4, 'Yash Bagul': 1, 'Tanuja Mali': 1}


The customer who buys most of the products Kaustoobh Mahajan buy 5 Items
```

[ ]:

```python
from collections import Counter
counter = dict(Counter(customer_details))
sorted_counter = sorted(counter.items(), key= lambda x:x[1],reverse=True)
sorted_counter = dict(sorted_counter)
print('The customer who buys most of the products',list(sorted_counter.
 ↪keys())[0], 'buy',list(sorted_counter.values())[0],'Items')
```

```
The customer who buys most of the products Kaustoobh Mahajan buy 5 Items
```

[ ]:

```python
from collections import Counter
counter = dict(Counter(customer_details))
names = list(counter.keys())
print(names)
male = 0
female = 0
for name in names:
  if gender[name]=='male':
    male = male+1
  if gender[name]=='female':
    female=female+1
print('Total no of Male=',male)
print('Total no of Female=',female)
```

```
['Kaustoobh Mahajan', 'Siddhi kivale', 'Sanket Kandalkar', 'Yash mali', 'Yash
Bagul', 'Tanuja Mali']
Total no of Male= 4
Total no of Female= 2
```

183 Shruti Garad

May 10, 2023

```
[1]: file=open('stud_info.csv','r')
     info_dataset=[]
     while True:
         data=file.readline()
         if data:
             info_dataset.append(data.replace("\n",  "").split(','))
         else:
             break
     print(info_dataset)
```

```
[['Roll No', 'name', 'Gender', 'DOB'], ['1', 'John', 'Male', '05-04-1988'],
['2', 'Mayur', 'Male', '04-05-1987'], ['3', 'Mangesh', 'Male', '25-05-1989'],
['4', 'Jessica', 'Female', '12-08-1990'], ['5', 'Jennifer', 'Female',
'02-09-1989'], ['6', 'Ramesh', 'Male', '03-09-1989'], ['7', 'Suresh', 'Male',
'04-09-1990'], ['8', 'Ganesh', 'Male', '05-10-1989'], ['9', 'Komal', 'Female',
'06-09-1989'], ['10', 'Mayuri', 'Female', '07-02-1988']]
```

```
[2]: R
     o
     l
```

```
[3]: for row in
         info_dataset[1:]:
         RollNo.append(row
         [0])
```

```
[4]: p
     r
     i
```

```
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
['John', 'Mayur', 'Mangesh', 'Jessica', 'Jennifer', 'Ramesh', 'Suresh',
'Ganesh', 'Komal', 'Mayuri']
['Male', 'Male', 'Male', 'Female', 'Female', 'Male', 'Male', 'Male', 'Female',
'Female']
```

```
['05-04-1988', '04-05-1987', '25-05-1989', '12-08-1990', '02-09-1989',
'03-09-1989', '04-09-1990', '05-10-1989', '06-09-1989', '07-02-1988']
```

[5]:

```
file=open('student_mar
ks.csv','r')
marks_dataset=[]
while True:
    data=file.readline()
    if data:
        marks_dataset.append(data.replace("\n",
        "").split(','))
```

```
[['Roll', 'Maths', 'Physics', 'Chemistry', 'Total', 'Percentage'], ['1', '55',
'45', '56', '156', '52.00'], ['2', '75', '55', '55', '185', '61.67'], ['3',
'25', '54', '89', '168', '56.00'], ['4', '78', '55', '86', '219', '73.00'],
['5', '58', '96', '78', '232', '77.33'], ['6', '88', '78', '58', '224',
'74.67'], ['7', '56', '89', '69', '214', '71.33'], ['8', '54', '55', '88',
'197', '65.67'], ['9', '46', '66', '65', '177', '59.00'], ['10', '89', '87',
'54', '230', '76.67']]
```

[6]:

```
M
a
t
h
```

[7]:

```
for row in
    marks_dataset[1:]
    :
    Maths.append(row[
    1])
```

[8]:

```
print(Ma
ths)
print(Ph
ysics)
```

```
['55', '75', '25', '78', '58', '88', '56', '54', '46', '89']
['45', '55', '54', '55', '96', '78', '89', '55', '66', '87']
['56', '55', '89', '86', '78', '58', '69', '88', '65', '54']
['156', '185', '168', '219', '232', '224', '214', '197', '177', '230']
['52.00', '61.67', '56.00', '73.00', '77.33', '74.67', '71.33', '65.67',
'59.00', '76.67']
```

```
[9]: file=open('stud_placem
     ent.csv','r')
     placement_dataset=[]
     while True:
         data=file.readline()
         if data:
             placement_dataset.append(data.replace("\n",
```

```
[['Roll No', 'Company', 'JobRole', 'Package'], ['1', 'Infosys', 'Data Analyst',
'10.2'], ['2', 'TCS', 'Java Developer', '9.6'], ['3', 'TCS', 'Data Scientist',
'12.60'], ['4', 'Infosys', 'Data Analyst', '10.2'], ['5', 'Oracle', 'Java
Developer', '9.6'], ['6', 'Oracle', 'Data Scientist', '12.60'], ['7', 'TCS',
'Tester', '6.50'], ['8', 'Infosys', 'Tester', '6.51'], ['9', 'Mindtree',
'Database Admin', '8.30'], ['10', 'Mindtree', 'Database Admin', '8.31']]
```

```
[10]: C
      o
      m
```

```
[11]: for row in
          placement_dataset
          [1:]:
```

```
[12]: p
      r
      i
```

```
['Infosys', 'TCS', 'TCS', 'Infosys', 'Oracle', 'Oracle', 'TCS', 'Infosys',
'Mindtree', 'Mindtree']
['Data Analyst', 'Java Developer', 'Data Scientist', 'Data Analyst', 'Java
Developer', 'Data Scientist', 'Tester', 'Tester', 'Database Admin', 'Database
Admin']
['10.2', '9.6', '12.60', '10.2', '9.6', '12.60', '6.50', '6.51', '8.30', '8.31']
```

```
[14]: studentdata=[]
      studentdata.append(Ro
      llNo)
      studentdata.append(Na
      me)
      studentdata.append(Ge
      nder)
      studentdata.append(DO
      studentdata.append(Company)
      studentdata.append(JobRole)
      studentdata.append(Package)
      print(studentdata)
```

```
[['1', '2', '3', '4', '5', '6', '7', '8', '9', '10'], ['John', 'Mayur',
'Mangesh', 'Jessica', 'Jennifer', 'Ramesh', 'Suresh', 'Ganesh', 'Komal',
'Mayuri'], ['Male', 'Male', 'Male', 'Female', 'Female', 'Male', 'Male', 'Male',
'Female', 'Female'], ['05-04-1988', '04-05-1987', '25-05-1989', '12-08-1990',
'02-09-1989', '03-09-1989', '04-09-1990', '05-10-1989', '06-09-1989',
'07-02-1988'], ['55', '75', '25', '78', '58', '88', '56', '54', '46', '89'],
['45', '55', '54', '55', '96', '78', '89', '55', '66', '87'], ['56', '55', '89',
'86', '78', '58', '69', '88', '65', '54'], ['156', '185', '168', '219', '232',
'224', '214', '197', '177', '230'], ['52.00', '61.67', '56.00', '73.00',
'77.33', '74.67', '71.33', '65.67', '59.00', '76.67'], ['Infosys', 'TCS', 'TCS',
'Infosys', 'Oracle', 'Oracle', 'TCS', 'Infosys', 'Mindtree', 'Mindtree'], ['Data
Analyst', 'Java Developer', 'Data Scientist', 'Data Analyst', 'Java Developer',
'Data Scientist', 'Tester', 'Tester', 'Database Admin', 'Database Admin'],
['10.2', '9.6', '12.60', '10.2', '9.6', '12.60', '6.50', '6.51', '8.30',
'8.31']]
```

[15]:
```
fw=open("StudentDetails.csv","w")
```

[16]:
```
data_to_write=[]
for i in
    range(len(student
    data[0])):
    row=list()
    for j in
        range(len(stu
```

```
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data     Analyst,10.2,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data     Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data     Scientist,12.60,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data     Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data     Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n',
'6,Ramesh,Male,03-09-1989,88,78,58,224,74.67,Oracle,Data Scientist,12.60,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n',
'6,Ramesh,Male,03-09-1989,88,78,58,224,74.67,Oracle,Data Scientist,12.60,\n',
'7,Suresh,Male,04-09-1990,56,89,69,214,71.33,TCS,Tester,6.50,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data  Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n',
'6,Ramesh,Male,03-09-1989,88,78,58,224,74.67,Oracle,Data Scientist,12.60,\n',
'7,Suresh,Male,04-09-1990,56,89,69,214,71.33,TCS,Tester,6.50,\n',
'8,Ganesh,Male,05-10-1989,54,55,88,197,65.67,Infosys,Tester,6.51,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data  Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n',
'6,Ramesh,Male,03-09-1989,88,78,58,224,74.67,Oracle,Data Scientist,12.60,\n',
'7,Suresh,Male,04-09-1990,56,89,69,214,71.33,TCS,Tester,6.50,\n',
'8,Ganesh,Male,05-10-1989,54,55,88,197,65.67,Infosys,Tester,6.51,\n',
'9,Komal,Female,06-09-1989,46,66,65,177,59.00,Mindtree,Database Admin,8.30,\n']
['1,John,Male,05-04-1988,55,45,56,156,52.00,Infosys,Data Analyst,10.2,\n',
'2,Mayur,Male,04-05-1987,75,55,55,185,61.67,TCS,Java Developer,9.6,\n',
'3,Mangesh,Male,25-05-1989,25,54,89,168,56.00,TCS,Data Scientist,12.60,\n',
'4,Jessica,Female,12-08-1990,78,55,86,219,73.00,Infosys,Data Analyst,10.2,\n',
'5,Jennifer,Female,02-09-1989,58,96,78,232,77.33,Oracle,Java Developer,9.6,\n',
'6,Ramesh,Male,03-09-1989,88,78,58,224,74.67,Oracle,Data Scientist,12.60,\n',
'7,Suresh,Male,04-09-1990,56,89,69,214,71.33,TCS,Tester,6.50,\n',
'8,Ganesh,Male,05-10-1989,54,55,88,197,65.67,Infosys,Tester,6.51,\n',
'9,Komal,Female,06-09-1989,46,66,65,177,59.00,Mindtree,Database Admin,8.30,\n',
'10,Mayuri,Female,07-02-1988,89,87,54,230,76.67,Mindtree,Database
Admin,8.31,\n']
```

```
[17]: fw.writelines(data_to_write)
```

```
[18]: fw.close()
```

```
[19]: print("Math
      Marks=",Maths)
      print("Phyics
      Marks=",Physics)
      print("Chemistry
      Marks=",Chemistry)
      math=[int(i) for i in
      Maths]
      physics=[int(i) for i
      in Physics]
      chemistry=[int(i) f
```

```
Math Marks= ['55', '75', '25', '78', '58', '88', '56', '54', '46', '89']
Phyics Marks= ['45', '55', '54', '55', '96', '78', '89', '55', '66', '87']
Chemistry Marks= ['56', '55', '89', '86', '78', '58', '69', '88', '65', '54']
Sum of Marks= [156, 185, 168, 219, 232, 224, 214, 197, 177, 230]
Average Marks= [156, 185, 168, 219, 232, 224, 214, 197, 177, 230]
```

```
[20]: print("Maximum Marks=",max(avg))
```

```
Maximum Marks= 232
```

```
[21]: print("Minimum Marks=",min(avg))
```

```
Minimum Marks= 156
```

```
[22]: print("Total No of Student=",len(studentdata[0]))
```

```
Total No of Student= 10
```

```
[23]: per=[]
      for i in range(len(sum_of_marks)):
          per.append(round((100*sum_of_marks[
          i]/270), 2))
```

```
Percentage= [57.78, 68.52, 62.22, 81.11, 85.93, 82.96, 79.26, 72.96, 65.56,
85.19]
```