

```
In [2]: 1 import pandas as pd
        2 import numpy as np
        3 import seaborn as sns
        4 import matplotlib.pyplot as plt
```

```
In [3]: 1 mydata=pd.read_csv("mobile_price.csv")
```

```
In [4]: 1 mydata
```

Out[4]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411
...
1995	794	1	0.5	1	0	1	2	0.8	106	6	...	1222	1890	668
1996	1965	1	2.6	1	0	0	39	0.2	187	4	...	915	1965	2032
1997	1911	0	0.9	1	1	1	36	0.7	108	8	...	868	1632	3057
1998	1512	0	0.9	0	4	1	46	0.1	145	5	...	336	670	869
1999	510	1	2.0	1	5	1	45	0.9	168	6	...	483	754	3919

2000 rows × 21 columns



In [5]: 1 mydata.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 21 columns):
#   Column          Non-Null Count  Dtype
---  -
0   battery_power    2000 non-null   int64
1   blue             2000 non-null   int64
2   clock_speed      2000 non-null   float64
3   dual_sim         2000 non-null   int64
4   fc              2000 non-null   int64
5   four_g          2000 non-null   int64
6   int_memory       2000 non-null   int64
7   m_dep           2000 non-null   float64
8   mobile_wt        2000 non-null   int64
9   n_cores          2000 non-null   int64
10  pc               2000 non-null   int64
11  px_height        2000 non-null   int64
12  px_width         2000 non-null   int64
13  ram              2000 non-null   int64
14  sc_h             2000 non-null   int64
15  sc_w             2000 non-null   int64
16  talk_time        2000 non-null   int64
17  three_g          2000 non-null   int64
18  touch_screen     2000 non-null   int64
19  wifi             2000 non-null   int64
20  price_range      2000 non-null   int64
dtypes: float64(2), int64(19)
memory usage: 328.2 KB
```

In [6]: 1 mydata.describe()

Out[6]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores
count	2000.000000	2000.0000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000	2000.000000
mean	1238.518500	0.4950	1.522250	0.509500	4.309500	0.521500	32.046500	0.501750	140.249000	4.520500
std	439.418206	0.5001	0.816004	0.500035	4.341444	0.499662	18.145715	0.288416	35.399655	2.287830
min	501.000000	0.0000	0.500000	0.000000	0.000000	0.000000	2.000000	0.100000	80.000000	1.000000
25%	851.750000	0.0000	0.700000	0.000000	1.000000	0.000000	16.000000	0.200000	109.000000	3.000000
50%	1226.000000	0.0000	1.500000	1.000000	3.000000	1.000000	32.000000	0.500000	141.000000	4.000000
75%	1615.250000	1.0000	2.200000	1.000000	7.000000	1.000000	48.000000	0.800000	170.000000	7.000000
max	1998.000000	1.0000	3.000000	1.000000	19.000000	1.000000	64.000000	1.000000	200.000000	8.000000

8 rows × 11 columns



```
In [7]: 1 mydata.isnull().sum()
```

```
Out[7]: battery_power    0  
blue                    0  
clock_speed            0  
dual_sim               0  
fc                     0  
four_g                 0  
int_memory             0  
m_dep                  0  
mobile_wt              0  
n_cores                0  
pc                     0  
px_height              0  
px_width               0  
ram                    0  
sc_h                   0  
sc_w                   0  
talk_time              0  
three_g                0  
touch_screen           0  
wifi                   0  
price_range            0  
dtype: int64
```

```
In [8]: 1 mydata_corr=mydata.corr()
```

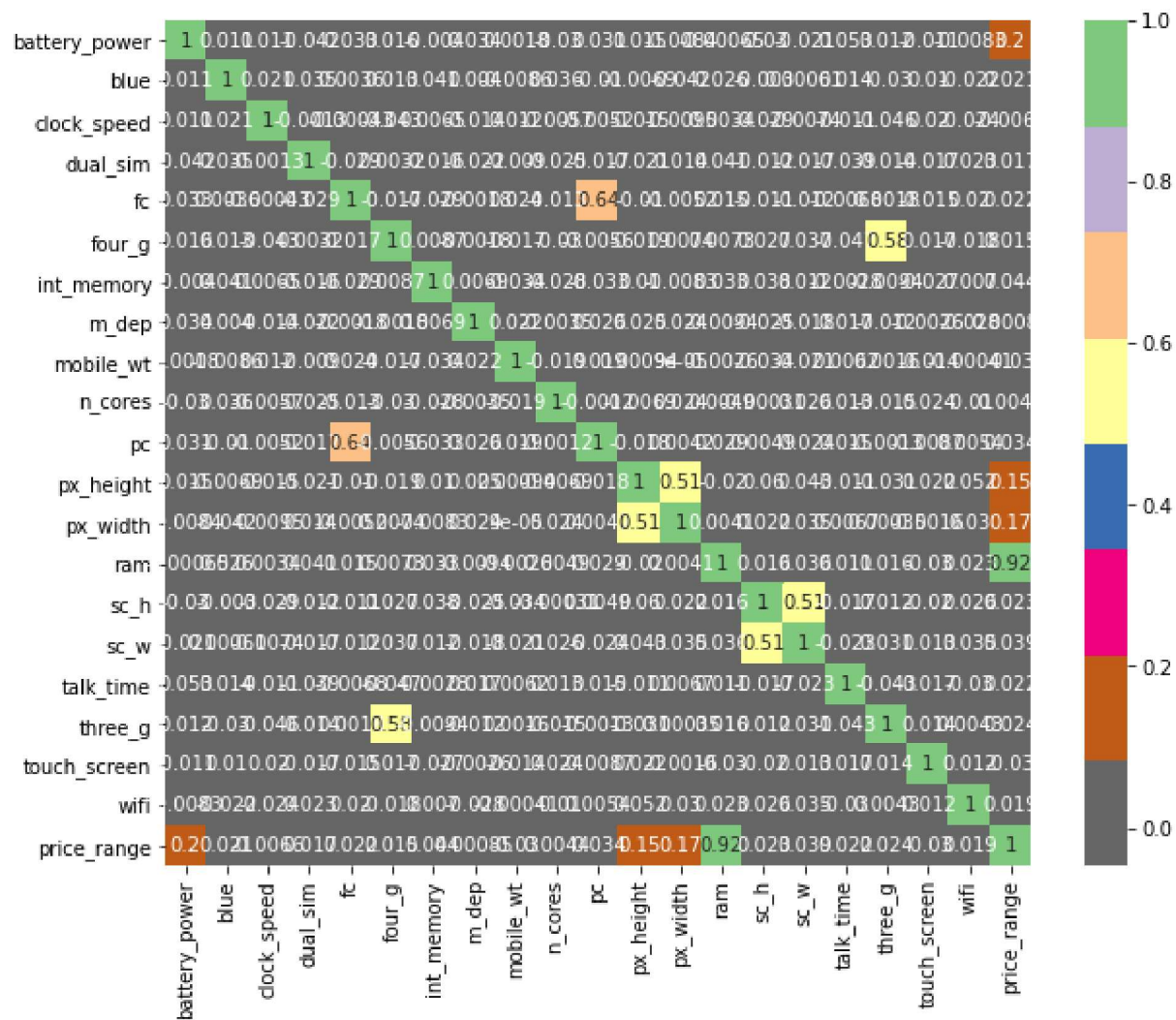
In [9]: 1 mydata_corr

Out[9]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...
battery_power	1.000000	0.011252	0.011482	-0.041847	0.033334	0.015665	-0.004004	0.034085	0.001844	-0.029727	...
blue	0.011252	1.000000	0.021419	0.035198	0.003593	0.013443	0.041177	0.004049	-0.008605	0.036161	...
clock_speed	0.011482	0.021419	1.000000	-0.001315	-0.000434	-0.043073	0.006545	-0.014364	0.012350	-0.005724	...
dual_sim	-0.041847	0.035198	-0.001315	1.000000	-0.029123	0.003187	-0.015679	-0.022142	-0.008979	-0.024658	...
fc	0.033334	0.003593	-0.000434	-0.029123	1.000000	-0.016560	-0.029133	-0.001791	0.023618	-0.013356	...
four_g	0.015665	0.013443	-0.043073	0.003187	-0.016560	1.000000	0.008690	-0.001823	-0.016537	-0.029706	...
int_memory	-0.004004	0.041177	0.006545	-0.015679	-0.029133	0.008690	1.000000	0.006886	-0.034214	-0.028310	...
m_dep	0.034085	0.004049	-0.014364	-0.022142	-0.001791	-0.001823	0.006886	1.000000	0.021756	-0.003504	...
mobile_wt	0.001844	-0.008605	0.012350	-0.008979	0.023618	-0.016537	-0.034214	0.021756	1.000000	-0.018989	...
n_cores	-0.029727	0.036161	-0.005724	-0.024658	-0.013356	-0.029706	-0.028310	-0.003504	-0.018989	1.000000	...
pc	0.031441	-0.009952	-0.005245	-0.017143	0.644595	-0.005598	-0.033273	0.026282	0.018844	-0.001193	...
px_height	0.014901	-0.006872	-0.014523	-0.020875	-0.009990	-0.019236	0.010441	0.025263	0.000939	-0.006872	...
px_width	-0.008402	-0.041533	-0.009476	0.014291	-0.005176	0.007448	-0.008335	0.023566	0.000090	0.024480	...
ram	-0.000653	0.026351	0.003443	0.041072	0.015099	0.007313	0.032813	-0.009434	-0.002581	0.004868	...
sc_h	-0.029959	-0.002952	-0.029078	-0.011949	-0.011014	0.027166	0.037771	-0.025348	-0.033855	-0.000315	...
sc_w	-0.021421	0.000613	-0.007378	-0.016666	-0.012373	0.037005	0.011731	-0.018388	-0.020761	0.025826	...
talk_time	0.052510	0.013934	-0.011432	-0.039404	-0.006829	-0.046628	-0.002790	0.017003	0.006209	0.013148	...
three_g	0.011522	-0.030236	-0.046433	-0.014008	0.001793	0.584246	-0.009366	-0.012065	0.001551	-0.014733	...
touch_screen	-0.010516	0.010061	0.019756	-0.017117	-0.014828	0.016758	-0.026999	-0.002638	-0.014368	0.023774	...
wifi	-0.008343	-0.021863	-0.024471	0.022740	0.020085	-0.017620	0.006993	-0.028353	-0.000409	-0.009964	...
price_range	0.200723	0.020573	-0.006606	0.017444	0.021998	0.014772	0.044435	0.000853	-0.030302	0.004399	...

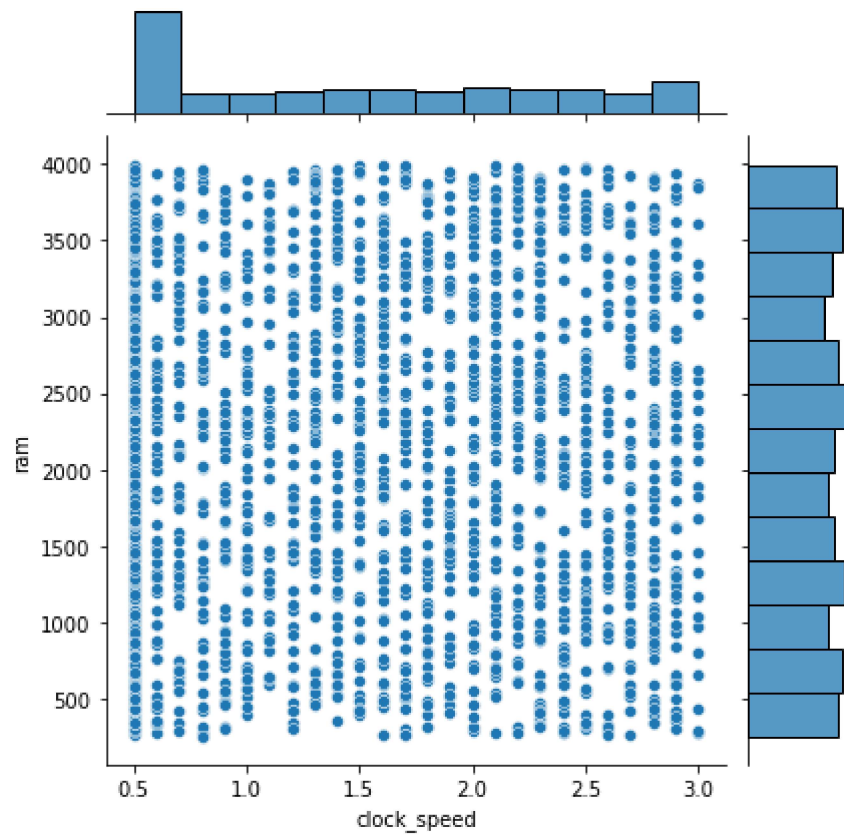
21 rows × 21 columns

```
In [10]: 1 plt.figure(figsize=(10,8))
2         sns.heatmap(mydata_corr,annot=True,cmap='Accent_r')
3         plt.show()
```



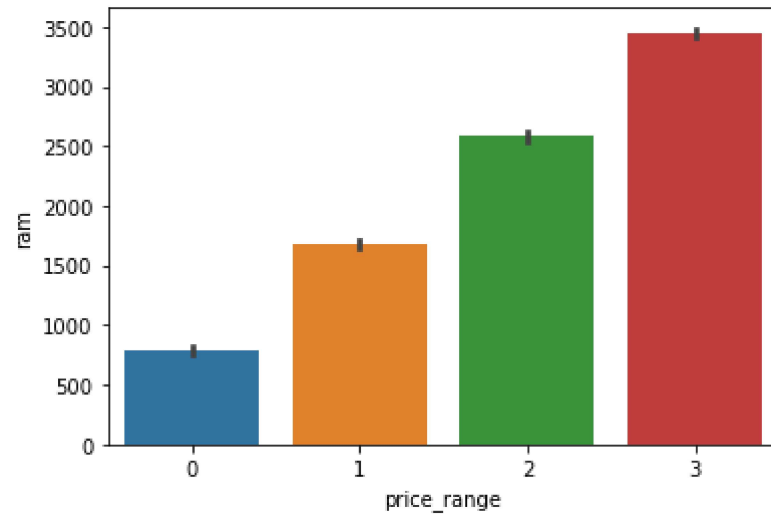
```
In [11]: 1 sns.jointplot(x='clock_speed', y='ram', data=mydata)
```

```
Out[11]: <seaborn.axisgrid.JointGrid at 0x20608862580>
```

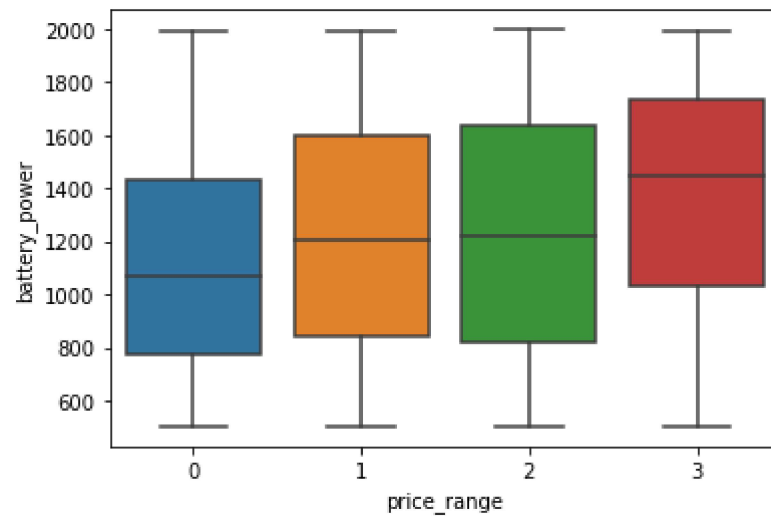


```
In [12]: 1 sns.barplot(x='price_range', y='ram', data=mydata)
```

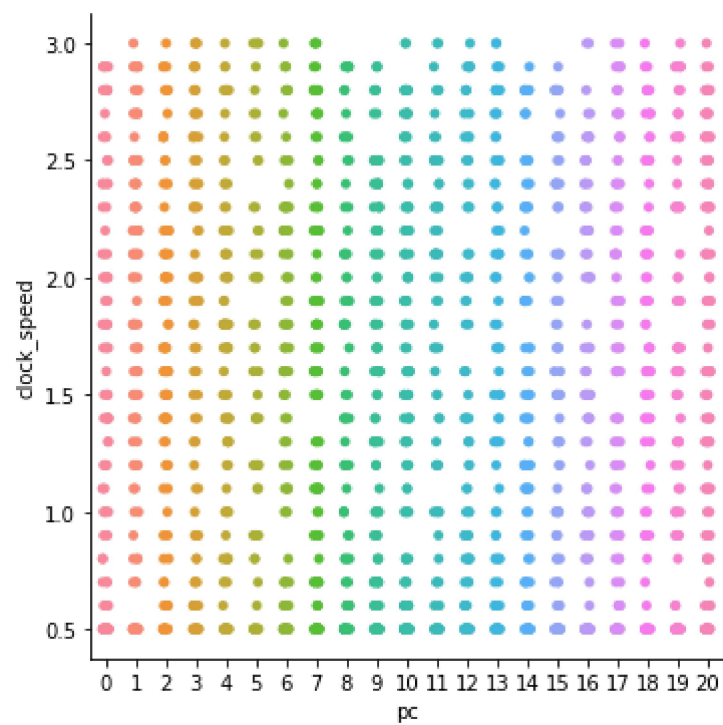
```
Out[12]: <AxesSubplot:xlabel='price_range', ylabel='ram'>
```




```
In [13]: 1 sns.boxplot(x='price_range', y='battery_power', data=mydata);
```

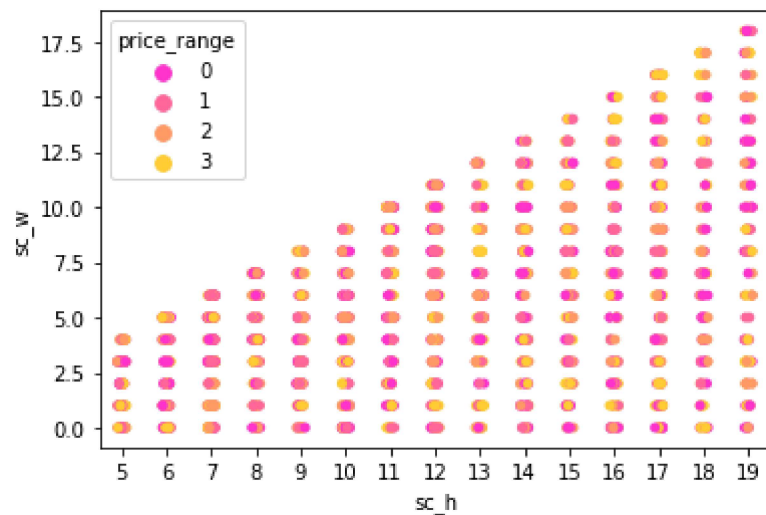


```
In [14]: 1 sns.catplot(x='pc',y='clock_speed', data=mydata);
```



```
In [17]: 1 sns.stripplot(x='sc_h',y='sc_w', data=mydata,palette='spring',hue='price_range')
```

```
Out[17]: <AxesSubplot:xlabel='sc_h', ylabel='sc_w'>
```



```
In [18]: 1 x_ind=mydata.drop('price_range', axis=1)
```

In [19]: 1 x_ind

Out[19]:

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	pc	px_height	px_width	ram
0	842	0	2.2	0	1	0	7	0.6	188	2	2	20	756	2549
1	1021	1	0.5	1	0	1	53	0.7	136	3	6	905	1988	2631
2	563	1	0.5	1	2	1	41	0.9	145	5	6	1263	1716	2603
3	615	1	2.5	0	0	0	10	0.8	131	6	9	1216	1786	2769
4	1821	1	1.2	0	13	1	44	0.6	141	2	14	1208	1212	1411
...
1995	794	1	0.5	1	0	1	2	0.8	106	6	14	1222	1890	668
1996	1965	1	2.6	1	0	0	39	0.2	187	4	3	915	1965	2032
1997	1911	0	0.9	1	1	1	36	0.7	108	8	3	868	1632	3057
1998	1512	0	0.9	0	4	1	46	0.1	145	5	5	336	670	869
1999	510	1	2.0	1	5	1	45	0.9	168	6	16	483	754	3919

2000 rows × 20 columns



In [20]: 1 y_dep=mydata.price_range

```
In [21]: 1 y_dep
```

```
Out[21]: 0      1
          1      2
          2      2
          3      2
          4      1
          ..
        1995     0
        1996     2
        1997     3
        1998     0
        1999     3
        Name: price_range, Length: 2000, dtype: int64
```

```
In [22]: 1 from sklearn.model_selection import train_test_split
```

```
In [23]: 1 x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,train_size=0.8,random_state=2)
```

```
In [24]: 1 from sklearn.naive_bayes import GaussianNB
```

```
In [25]: 1 model=GaussianNB()
```

```
In [27]: 1 model.fit(x_train,y_train)
```

```
Out[27]: GaussianNB()
```

```
In [28]: 1 y_pred=model.predict(x_test)
```

```
In [29]: 1 from sklearn.metrics import confusion_matrix, accuracy_score
```

```
In [30]: 1 confusion_matrix(y_test,y_pred)
```

```
Out[30]: array([[94,  4,  0,  0],  
               [16, 80, 13,  0],  
               [ 0, 17, 67, 10],  
               [ 0,  0,  4, 95]], dtype=int64)
```

```
In [31]: 1 accuracy_score(y_test,y_pred)
```

```
Out[31]: 0.84
```

```
In [ ]: 1
```