```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

```
data=pd.read_csv('train.csv')
```

```
data
```

	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

```
len(data)
```

```
2000
```

```
data.head()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram	sc_
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	1
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603	1
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	2769	1
4	1821	1	1.2	0	13	1	44	0.6	141	2	...	1208	1212	1411	

5 rows × 21 columns

```
data.count()
```

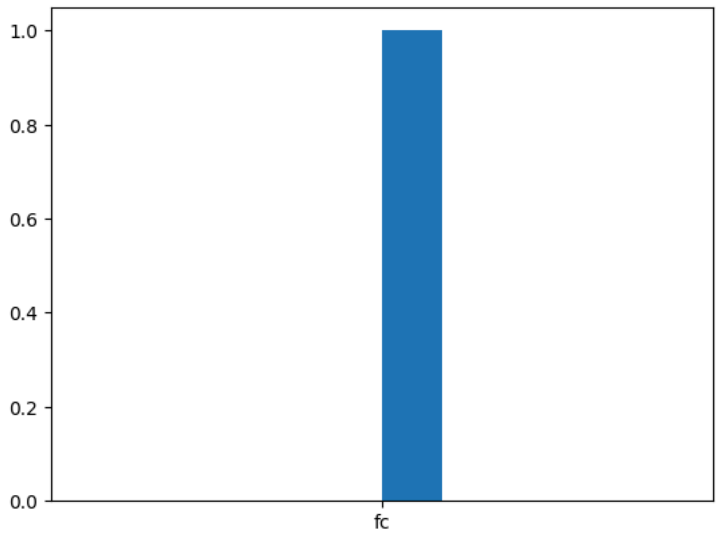
```
battery_power    2000
blue             2000
clock_speed      2000
dual_sim         2000
fc               2000
four_g           2000
int_memory       2000
m_dep            2000
mobile_wt        2000
n_cores          2000
pc               2000
px_height        2000
px_width         2000
ram              2000
sc_h             2000
sc_w             2000
talk_time        2000
three_g          2000
touch_screen     2000
wifi             2000
price_range      2000
dtype: int64
```

```
data.describe()
```

	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	2.287837
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 × 21 columns

```
plt.hist('fc')
plt.show()
```




```
data['battery_power'].min(),data['battery_power'].max()
```

```
(501, 1998)
```

```
data['blue'].value_counts()
```

```
blue
0    1010
1     990
Name: count, dtype: int64
```

```
data['blue'].value_counts()*100/len(data)
```

```
blue
0    50.5
1    49.5
Name: count, dtype: float64
```

```
data['clock_speed'].value_counts()
```

```
clock_speed
0.5    413
2.8     85
2.3     78
2.1     76
1.6     76
2.5     74
0.6     74
1.4     70
1.3     68
1.5     67
2.0     67
1.9     65
0.7     64
2.9     62
1.8     62
1.0     61
```

```
1.7     60
2.2     59
0.9     58
2.4     58
0.8     58
1.2     56
2.6     55
2.7     55
1.1     51
3.0     28
Name: count, dtype: int64
```

```
data['clock_speed'].value_counts()*100/len(data)
```

```
clock_speed
0.5    20.65
2.8     4.25
2.3     3.90
2.1     3.80
1.6     3.80
2.5     3.70
0.6     3.70
1.4     3.50
1.3     3.40
1.5     3.35
2.0     3.35
1.9     3.25
0.7     3.20
2.9     3.10
1.8     3.10
1.0     3.05
1.7     3.00
2.2     2.95
0.9     2.90
2.4     2.90
0.8     2.90
1.2     2.80
2.6     2.75
2.7     2.75
1.1     2.55
3.0     1.40
Name: count, dtype: float64
```

```
data['dual_sim'].value_counts()
```

```
dual_sim
1    1019
0     981
Name: count, dtype: int64
```

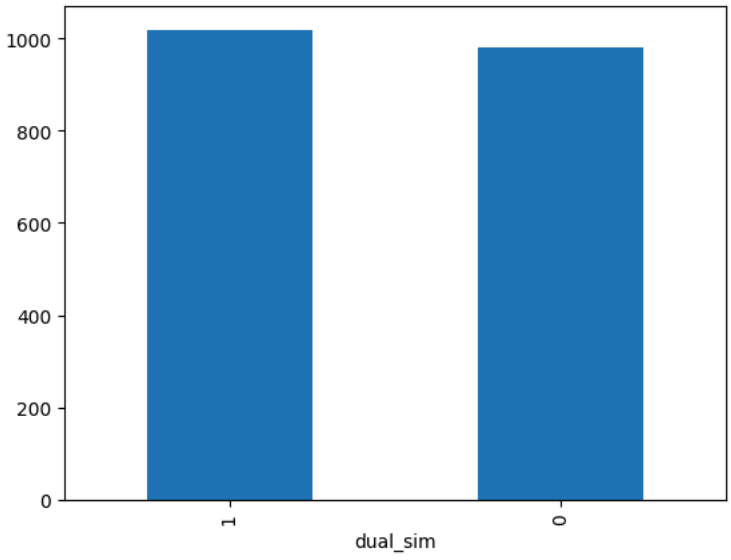
```
data['fc'].value_counts()
```

```
fc
0     474
1     245
2     189
3     170
5     139
4     133
6     112
7     100
9      78
8      77
10     62
11     51
12     45
13     40
16     24
15     23
14     20
18     11
17      6
19      1
Name: count, dtype: int64
```

```
%matplotlib inline
alpha_color=0.5
```

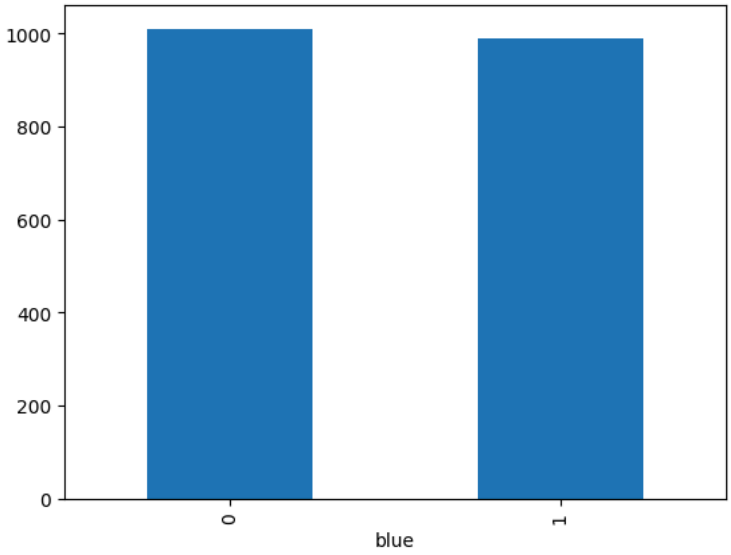
```
data['dual_sim'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='dual_sim'>
```



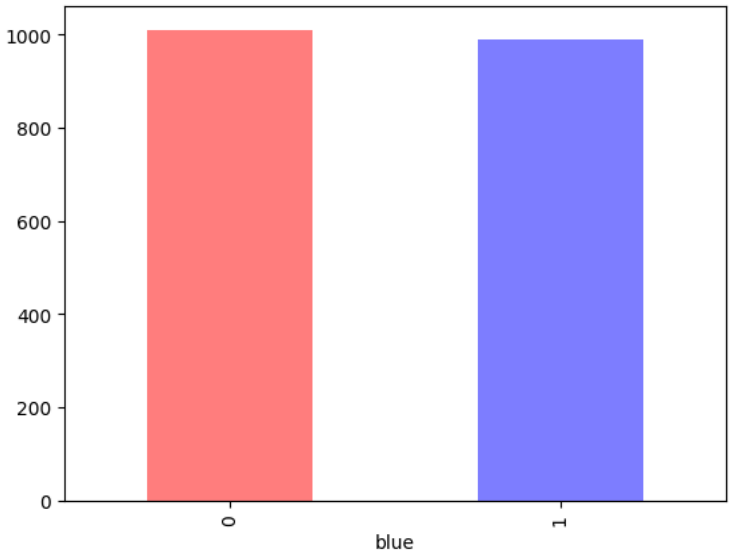

```
data['blue'].value_counts().plot(kind='bar')
```

```
<Axes: xlabel='blue'>
```




```
data['blue'].value_counts().plot(kind='bar',color=['r','b'],alpha=alpha_color)
```

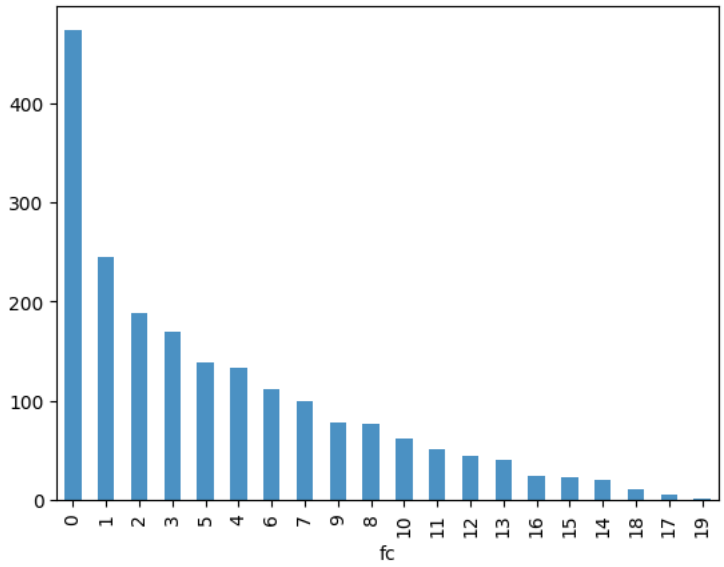
```
<Axes: xlabel='blue'>
```




```
alpha_color=0.8
```

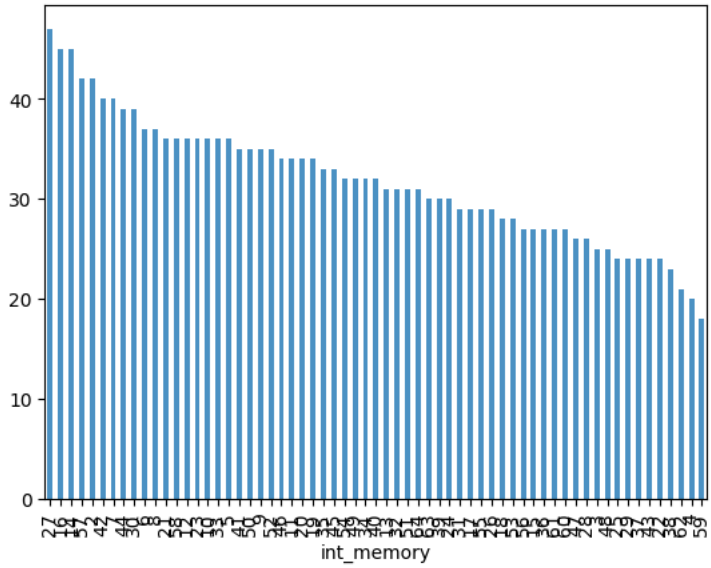
```
data['fc'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='fc'>
```



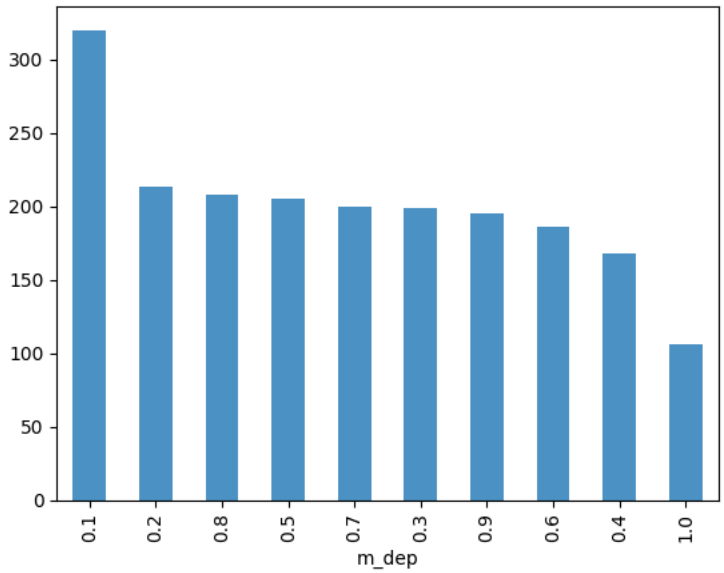

```
data['int_memory'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='int_memory'>
```



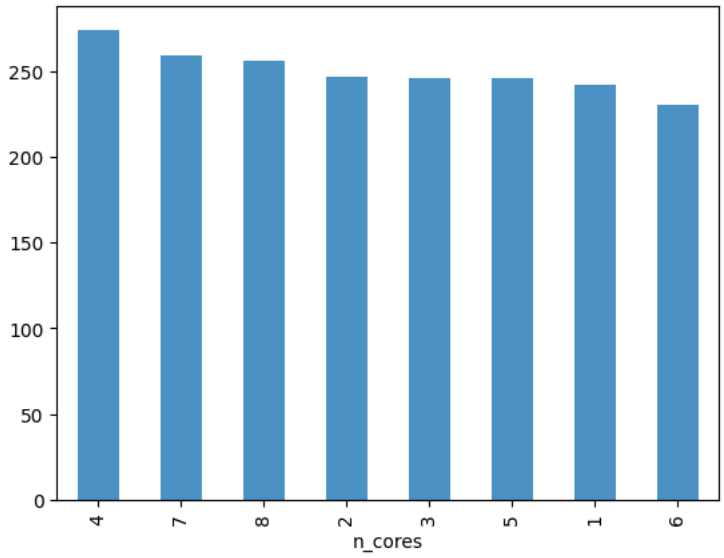

```
data['m_dep'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='m_dep'>
```



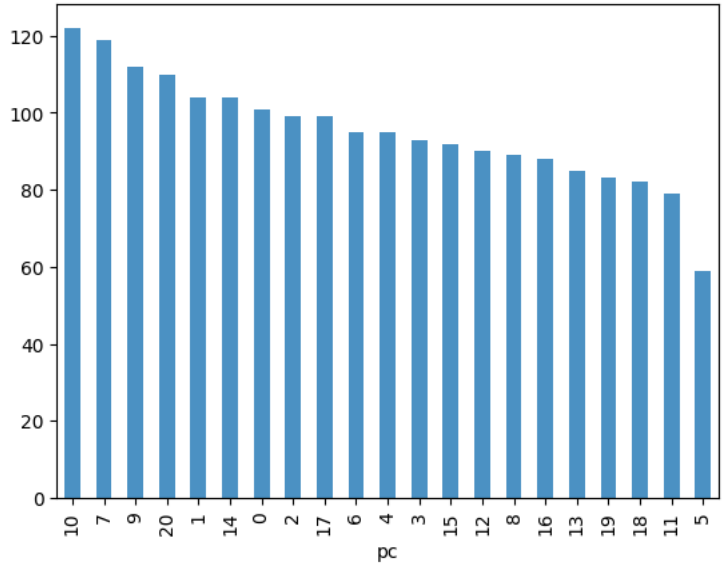
```
data['n_cores'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='n_cores'>
```



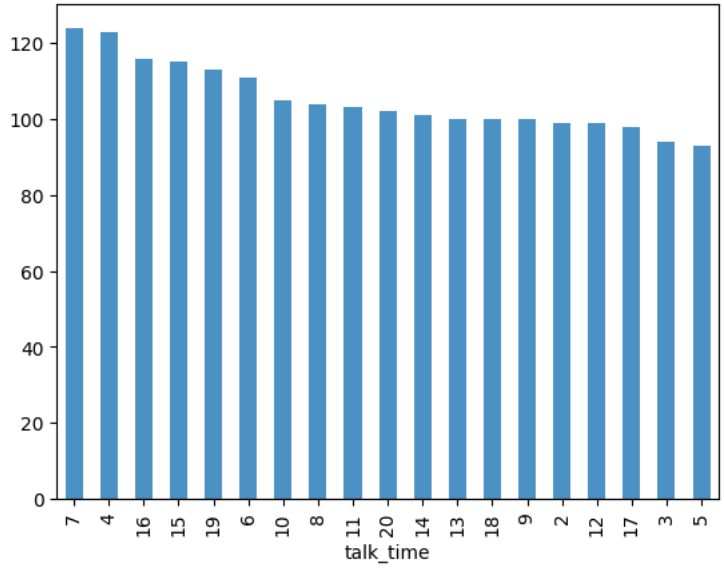

```
data['pc'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='pc'>
```



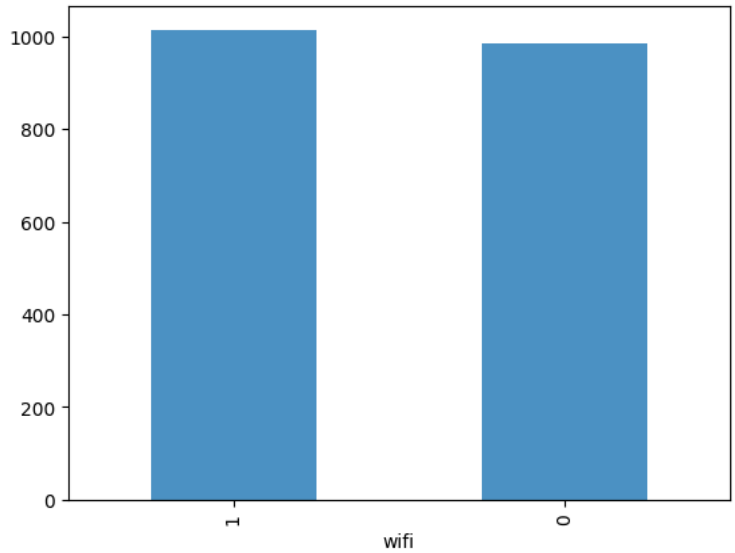

```
data['talk_time'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='talk_time'>
```



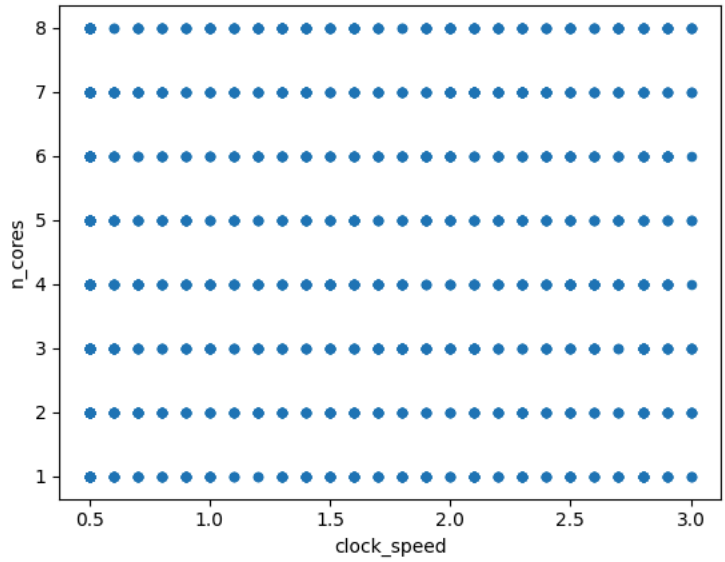
```
data['wifi'].value_counts().plot(kind='bar',alpha=alpha_color)
```

```
<Axes: xlabel='wifi'>
```



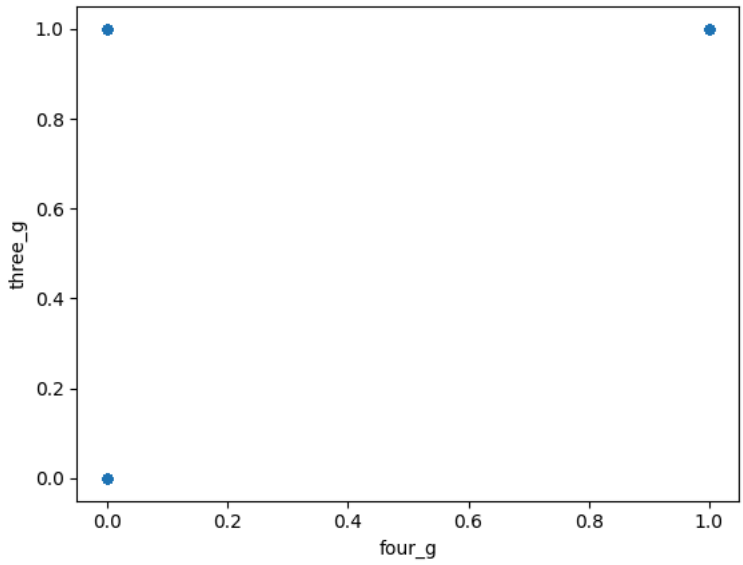

```
data.plot(kind='scatter', x='clock_speed', y='n_cores')
```

```
<Axes: xlabel='clock_speed', ylabel='n_cores'>
```



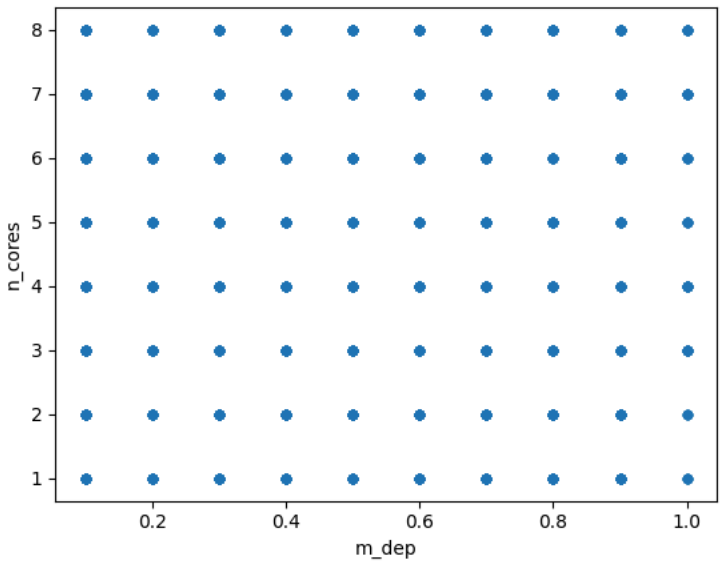

```
data.plot(kind='scatter', x='four_g', y='three_g')
```

```
<Axes: xlabel='four_g', ylabel='three_g'>
```



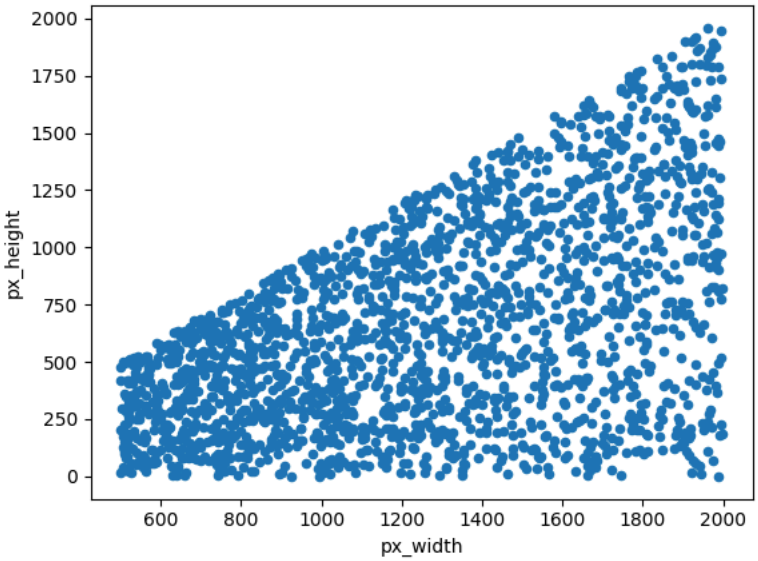
```
data.plot(kind='scatter', x='m_dep', y='n_cores')
```

```
<Axes: xlabel='m_dep', ylabel='n_cores'>
```



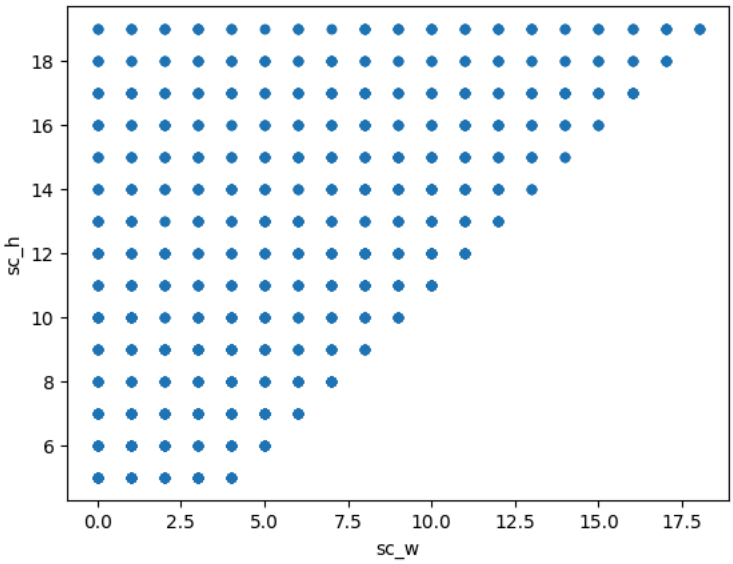

```
data.plot(kind='scatter', x='px_width', y='px_height')
```

```
<Axes: xlabel='px_width', ylabel='px_height'>
```



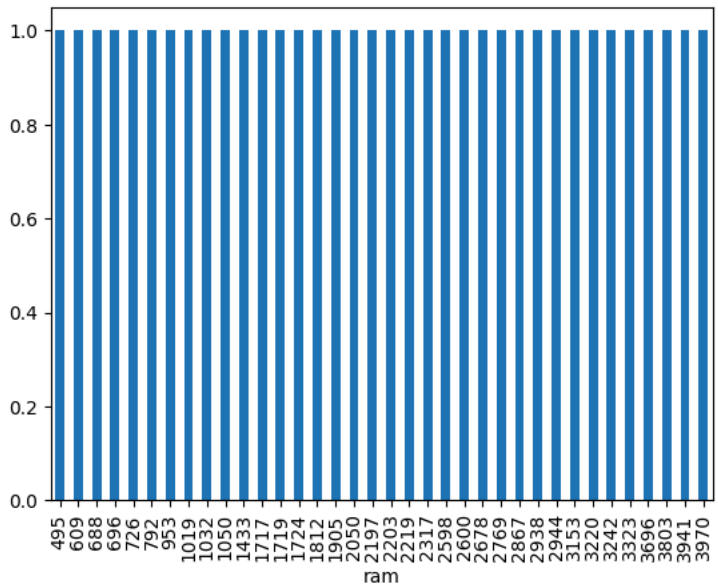

```
data.plot(kind='scatter', x='sc_w', y='sc_h')
```

```
<Axes: xlabel='sc_w', ylabel='sc_h'>
```



```
data[data['int_memory']==10]['ram'].value_counts().sort_index().plot(kind='bar')
```

```
<Axes: xlabel='ram'>
```




```
data[data['Survived']==0]['AgeBin'].value_counts().sort_index().plot(kind='bar')
```

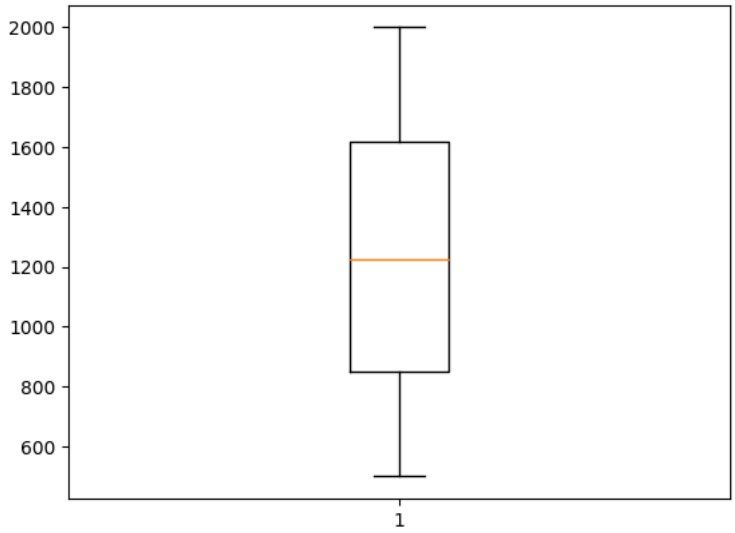
```
data.corr()
```

	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	p
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

```
plt.boxplot(data.battery_power)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ba550f449a0>,
  <matplotlib.lines.Line2D at 0x7ba550f44c40>],
 'caps': [<matplotlib.lines.Line2D at 0x7ba550f44d90>,
  <matplotlib.lines.Line2D at 0x7ba550f45060>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ba550f44700>],
 'medians': [<matplotlib.lines.Line2D at 0x7ba550f45300>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ba550f455a0>],
 'means': []}
```




```
data
```

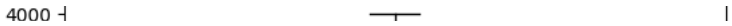
	battery_power	blue	clock_speed	dual_sim	fc	four_g	int_memory	m_dep	mobile_wt	n_cores	...	px_height	px_width	ram
0	842	0	2.2	0	1	0	7	0.6	188	2	...	20	756	2549
1	1021	1	0.5	1	0	1	53	0.7	136	3	...	905	1988	2631
2	563	1	0.5	1	2	1	41	0.9	145	5	...	1263	1716	2603
3	615	1	2.5	0	0	0	10	0.8	131	6	...	1216	1786	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

```
plt.boxplot(data.ram)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ba550fa2c50>,
  <matplotlib.lines.Line2D at 0x7ba550fa2ef0>],
 'caps': [<matplotlib.lines.Line2D at 0x7ba550fa3190>,
  <matplotlib.lines.Line2D at 0x7ba550fa3310>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ba550fa29b0>],
 'medians': [<matplotlib.lines.Line2D at 0x7ba550fa35b0>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ba550fa3850>],
```

```
df = data[data['price_range'].notna()]
```



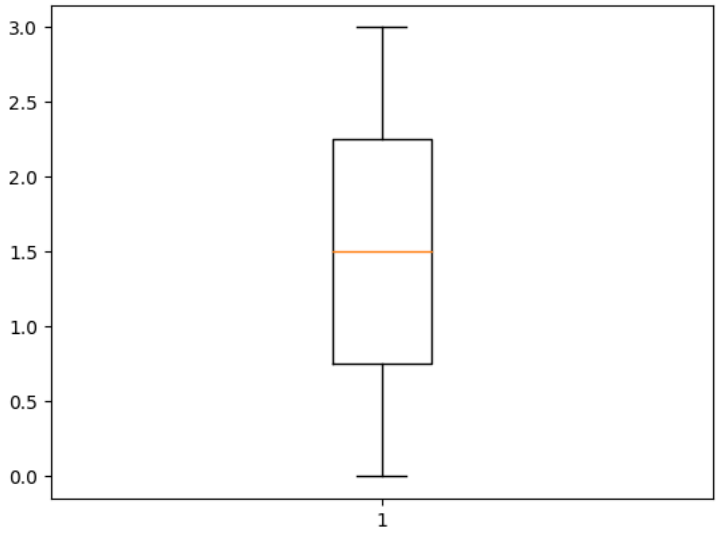

```
df
```

	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

```
plt.boxplot(df.price_range)
```

```
{'whiskers': [<matplotlib.lines.Line2D at 0x7ba550e31990>,
  <matplotlib.lines.Line2D at 0x7ba550e31c30>],
 'caps': [<matplotlib.lines.Line2D at 0x7ba550e31ed0>,
  <matplotlib.lines.Line2D at 0x7ba550e32170>],
 'boxes': [<matplotlib.lines.Line2D at 0x7ba550e316f0>],
 'medians': [<matplotlib.lines.Line2D at 0x7ba550e32410>],
 'fliers': [<matplotlib.lines.Line2D at 0x7ba550e326b0>],
 'means': []}
```



Start coding or generate with AI.