```
import numpy as np
import pandas as pd
import matplotlib
from matplotlib import pyplot as plt
%matplotlib inline
```

```
df=pd.read_csv("D:\\moj_projekat\\all_phones.csv")
df
```

	phone_name	brand	os	inches	resolution	battery	battery_type	ram(GB)	announcement_date	weight(g)	...	video_1080p	vi
0	Y6II Compact	Huawei	Android 5.1	5.00	720x1280	2200	Li-Po	2	2016-09-01	140.0	...	False	
1	K20 plus	LG	Android 7.0	5.30	720x1280	2700	Li-Ion	2	2016-12-01	140.0	...	True	
2	P8 Lite (2017)	Huawei	Android 7.0	5.20	1080x1920	3000	Li-Ion	4	2017-01-01	147.0	...	True	
3	Redmi Note 4	Xiaomi	Android 6.0	5.50	1080x1920	4100	Li-Po	4	2017-01-01	165.0	...	True	
4	P10	Huawei	Android 7.0	5.10	1080x1920	3200	Li-Ion	4	2017-02-01	145.0	...	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	
1507	vivo Y77t	Vivo	Android 13	6.64	1080x2388	5000	Li-Po	8	2023-08-18	190.0	...	True	
1508	11x	Realme	Android 13	6.72	1080x2400	5000	Li-Po	8	2023-08-23	190.0	...	True	
1509	GT5	Realme	Android 13	6.74	1240x2772	5240	Li-Po	16	2023-08-28	205.0	...	True	
1510	GT5 240W	Realme	Android 13	6.74	1240x2772	4600	Li-Po	24	2023-08-28	205.0	...	True	
1511	vivo iQOO Z7 Pro	Vivo	Android 13	6.78	1080x2400	4600	Li-Po	8	2023-08-31	175.0	...	True	

1512 rows × 22 columns

```
df.head(10)
```

	phone_name	brand	os	inches	resolution	battery	battery_type	ram(GB)	announcement_date	weight(g)	...	video_1080p	video_
0	Y6II Compact	Huawei	Android 5.1	5.0	720x1280	2200	Li-Po	2	2016-09-01	140.0	...	False	Fa
1	K20 plus	LG	Android 7.0	5.3	720x1280	2700	Li-Ion	2	2016-12-01	140.0	...	True	Fa
2	P8 Lite (2017)	Huawei	Android 7.0	5.2	1080x1920	3000	Li-Ion	4	2017-01-01	147.0	...	True	Fa
3	Redmi Note 4	Xiaomi	Android 6.0	5.5	1080x1920	4100	Li-Po	4	2017-01-01	165.0	...	True	Fa
4	P10	Huawei	Android 7.0	5.1	1080x1920	3200	Li-Ion	4	2017-02-01	145.0	...	True	Tı
5	Xperia XA1	Sony	Android 7.0	5.0	720x1280	2300	Li-Ion	3	2017-02-01	143.0	...	True	Fa
6	P10 Lite	Huawei	Android 7.0	5.2	1080x1920	3000	Li-Po	4	2017-02-01	146.0	...	True	Fa
7	P10 Plus	Huawei	Android 7.0	5.5	1440x2560	3750	Li-Ion	6	2017-02-01	165.0	...	True	Tı
8	Xperia XA1 Ultra	Sony	Android 7.0	6.0	1080x1920	2700	Li-Ion	4	2017-02-01	188.0	...	True	Fa
9	X power2	LG	Android 7.0	5.5	720x1280	4500	Li-Ion	2	2017-02-01	164.0	...	True	Fa

10 rows × 22 columns

```
df.info() #Informacije o kolonama i tipu podataka u njima
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1512 entries, 0 to 1511
Data columns (total 22 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   phone_name         1512 non-null   object 
 1   brand              1512 non-null   object 
 2   os                 1512 non-null   object 
 3   inches             1512 non-null   float64
 4   resolution         1512 non-null   object 
 5   battery            1512 non-null   int64  
 6   battery_type       1512 non-null   object 
 7   ram(GB)            1512 non-null   int64  
 8   announcement_date  1512 non-null   object 
 9   weight(g)          1512 non-null   float64
 10  storage(GB)        1512 non-null   int64  
 11  video_720p         1512 non-null   bool   
 12  video_1080p        1512 non-null   bool   
 13  video_4K           1512 non-null   bool   
 14  video_8K           1512 non-null   bool   
 15  video_30fps        1512 non-null   bool   
 16  video_60fps        1512 non-null   bool   
 17  video_120fps       1512 non-null   bool   
 18  video_240fps       1512 non-null   bool   
 19  video_480fps       1512 non-null   bool   
 20  video_960fps       1512 non-null   bool   
 21  price(USD)         1512 non-null   float64
dtypes: bool(10), float64(3), int64(3), object(6)
memory usage: 156.6+ KB
```

```
df.isna().sum() #Provera ima li null vrednosti
```

```
phone_name          0
brand               0
os                  0
inches              0
resolution          0
battery             0
battery_type        0
ram(GB)             0
announcement_date   0
weight(g)           0
storage(GB)         0
video_720p          0
video_1080p         0
video_4K            0
```

```
video_8K            0
video_30fps         0
video_60fps         0
video_120fps        0
video_240fps        0
video_480fps        0
video_960fps        0
price(USD)          0
dtype: int64
```

```
df.duplicated().sum()  #Provera ima li duplikata
```

```
0
```

```
df['phone_name'].value_counts() #Uvid u kategorijske vrednosti (model telefona)
```

```
phone_name
V30                         3
K5                          2
7 Pro                       2
9                           2
6                           2
                           ..
Galaxy A11                  1
Redmi Note 9 Pro (India)    1
Redmi Note 9 Pro Max        1
Find X2                     1
vivo iQOO Z7 Pro            1
Name: count, Length: 1496, dtype: int64
```

```
df[df['phone_name']=='7 Pro'] #Vidimo da postoje telefoni od različitih brendova sa istim nazivom modela
```

	phone_name	brand	os	inches	resolution	battery	battery_type	ram(GB)	announcement_date	weight(g)	...	video_1080p	vid
332	7 Pro	OnePlus	Android 9.0	6.67	1440x3120	4000	Li-Po	6	2019-05-14	206.0	...	True	
651	7 Pro	Realme	Android 10	6.40	1080x2400	4500	Li-Po	8	2020-09-03	182.0	...	True	

2 rows × 22 columns

```
df['os'].str.contains('Android',regex=False).value_counts() #1481/1512 telefona koriste Android OS
```

```
os
True     1481
False      31
Name: count, dtype: int64
```

```
df['width'] = df['resolution'].apply(lambda x: x.split('x')[0]).astype('int64')
df['height'] = df['resolution'].apply(lambda x: x.split('x')[1]).astype('int64') #Izdvajanje sirine i visine iz rezolucije
```

```
df['height']
```

```
0       1280
1       1280
2       1920
3       1920
4       1920
        ...
1507    2388
1508    2400
1509    2772
1510    2772
1511    2400
Name: height, Length: 1512, dtype: int64
```

```
df['announcement_year'] = df['announcement_date'].apply(lambda x: x.split('-')[0]).astype('int32') #Izdvajamo godinu objavljivanja telefona
```

```
df['announcement_year'].value_counts() #Najvise telefona ce izaci 2022. godine
```

```
announcement_year
2022    319
2020    294
2021    288
```

```
2019    210
2023    159
2018    155
2017     85
2016      2
Name: count, dtype: int64
```

```
df
```

	phone_name	brand	os	inches	resolution	battery	battery_type	ram(GB)	announcement_date	weight(g)	...	video_30fps	vi
0	Y6II Compact	Huawei	Android 5.1	5.00	720x1280	2200	Li-Po	2	2016-09-01	140.0	...	True	
1	K20 plus	LG	Android 7.0	5.30	720x1280	2700	Li-Ion	2	2016-12-01	140.0	...	True	
2	P8 Lite (2017)	Huawei	Android 7.0	5.20	1080x1920	3000	Li-Ion	4	2017-01-01	147.0	...	True	
3	Redmi Note 4	Xiaomi	Android 6.0	5.50	1080x1920	4100	Li-Po	4	2017-01-01	165.0	...	True	
4	P10	Huawei	Android 7.0	5.10	1080x1920	3200	Li-Ion	4	2017-02-01	145.0	...	True	
...	...	...	...	...	...	...	...	...	...	...	...	...	
1507	vivo Y77t	Vivo	Android 13	6.64	1080x2388	5000	Li-Po	8	2023-08-18	190.0	...	True	
1508	11x	Realme	Android 13	6.72	1080x2400	5000	Li-Po	8	2023-08-23	190.0	...	True	
1509	GT5	Realme	Android 13	6.74	1240x2772	5240	Li-Po	16	2023-08-28	205.0	...	False	
1510	GT5 240W	Realme	Android 13	6.74	1240x2772	4600	Li-Po	24	2023-08-28	205.0	...	False	
1511	vivo iQOO Z7 Pro	Vivo	Android 13	6.78	1080x2400	4600	Li-Po	8	2023-08-31	175.0	...	True	

1512 rows × 25 columns

```
df.drop(columns=['resolution'],inplace=True)
df.shape
```

```
(1512, 24)
```

```
df.describe()
```

	inches	battery	ram(GB)	weight(g)	storage(GB)	price(USD)	width	height	announcement_year
count	1512.000000	1512.000000	1512.000000	1512.000000	1512.000000	1512.000000	1512.000000	1512.000000	1512.000000
mean	6.422460	4389.798942	6.683862	187.636243	109.164683	337.847036	1035.212963	2207.190476	2020.410053
std	0.477043	784.607022	2.701433	26.200115	74.436484	266.740821	253.488940	469.734578	1.700190
min	3.800000	1821.000000	1.000000	130.000000	1.000000	40.000000	480.000000	800.000000	2016.000000
25%	6.300000	4000.000000	4.000000	175.000000	64.000000	179.997500	720.000000	1647.500000	2019.000000
50%	6.500000	4500.000000	8.000000	187.000000	128.000000	260.000000	1080.000000	2400.000000	2021.000000
75%	6.670000	5000.000000	8.000000	197.250000	128.000000	400.000000	1080.000000	2400.000000	2022.000000
max	10.400000	7250.000000	24.000000	500.000000	512.000000	2300.000000	3840.000000	3840.000000	2023.000000

```
gornja_granica=df['weight(g)'].mean()+3*df['weight(g)'].std()
donja_granica=df['weight(g)'].mean()-3*df['weight(g)'].std()
```

```
df1=df[(df['weight(g)']>donja_granica) & (df['weight(g)']<gornja_granica)]
df1.shape
```

```
(1493, 24)
```

```
df1[df1['storage(GB)']==1]
```

	phone_name	brand	os	inches	battery	battery_type	ram(GB)	announcement_date	weight(g)	storage(GB)	...	video_30fps	v
1510	GT5 240W	Realme	Android 13	6.74	4600	Li-Po	24	2023-08-28	205.0	1	...	False	

1 rows × 24 columns

```
df1[df1['price(USD)']==40]
```

	phone_name	brand	os	inches	battery	battery_type	ram(GB)	announcement_date	weight(g)	storage(GB)	...	video_30fps	vi
452	C2s	Realme	Android 9.0	6.1	4000	Li-Po	3	2020-01-01	166.0	32	...	True	

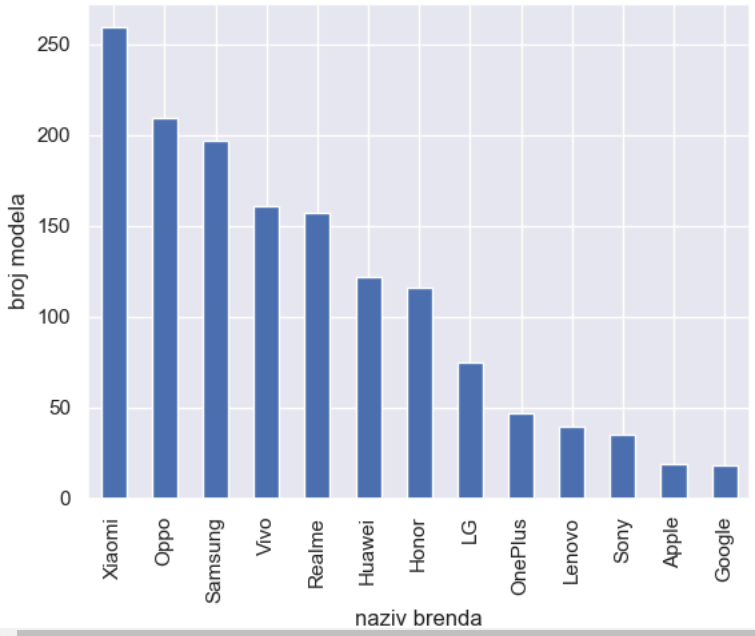
1 rows × 24 columns

```
for kolona in ['storage(GB)','price(USD)']:
    gornja_granica=df[kolona].mean()+3*df[kolona].std()
    donja_granica=df[kolona].mean()-3*df[kolona].std()
    df1=df1[(df1[kolona]>donja_granica) & (df1[kolona]<gornja_granica)]

df1.shape #Finalno otklanjanje autlajera
```

```
(1454, 24)
```

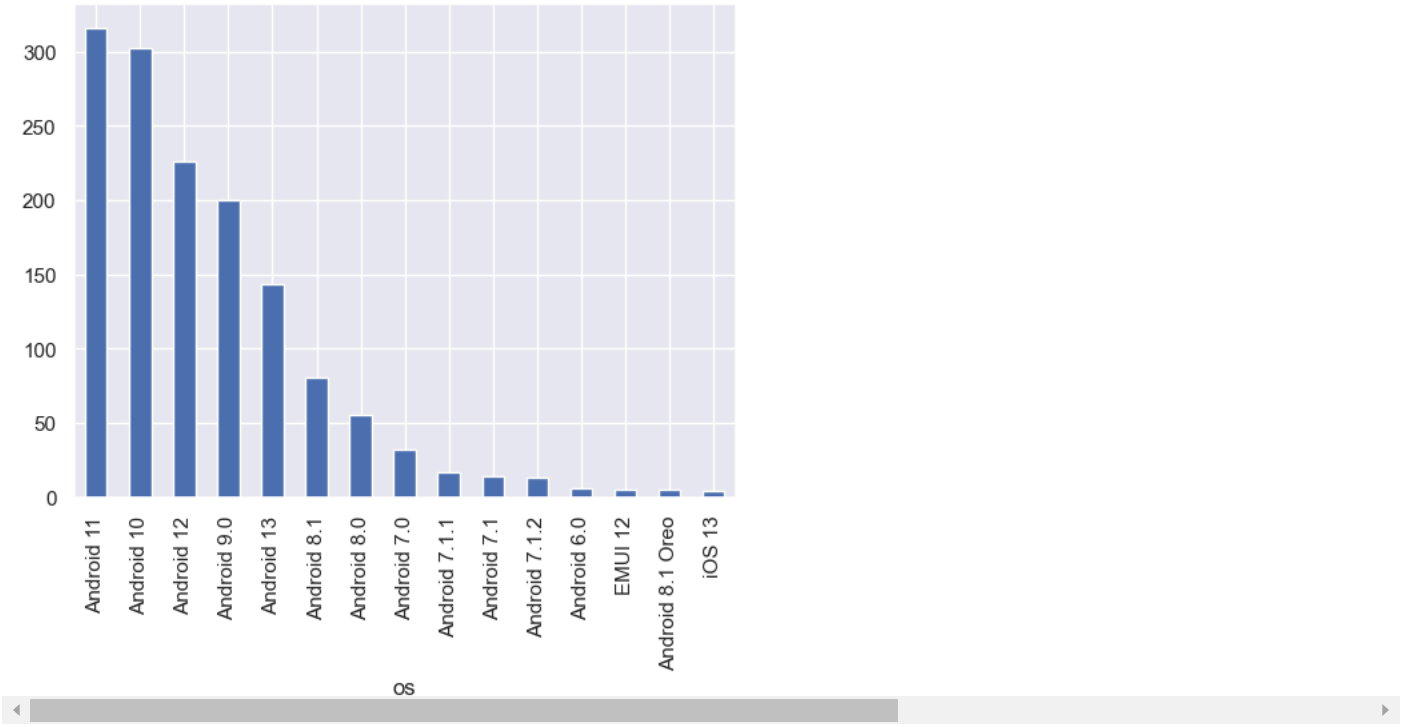
```
import seaborn as sns
sns.set()
df1['brand'].value_counts().plot(kind = 'bar',xlabel='naziv brenda',ylabel='broj modela')
```

```
<Axes: xlabel='naziv brenda', ylabel='broj modela'>
```

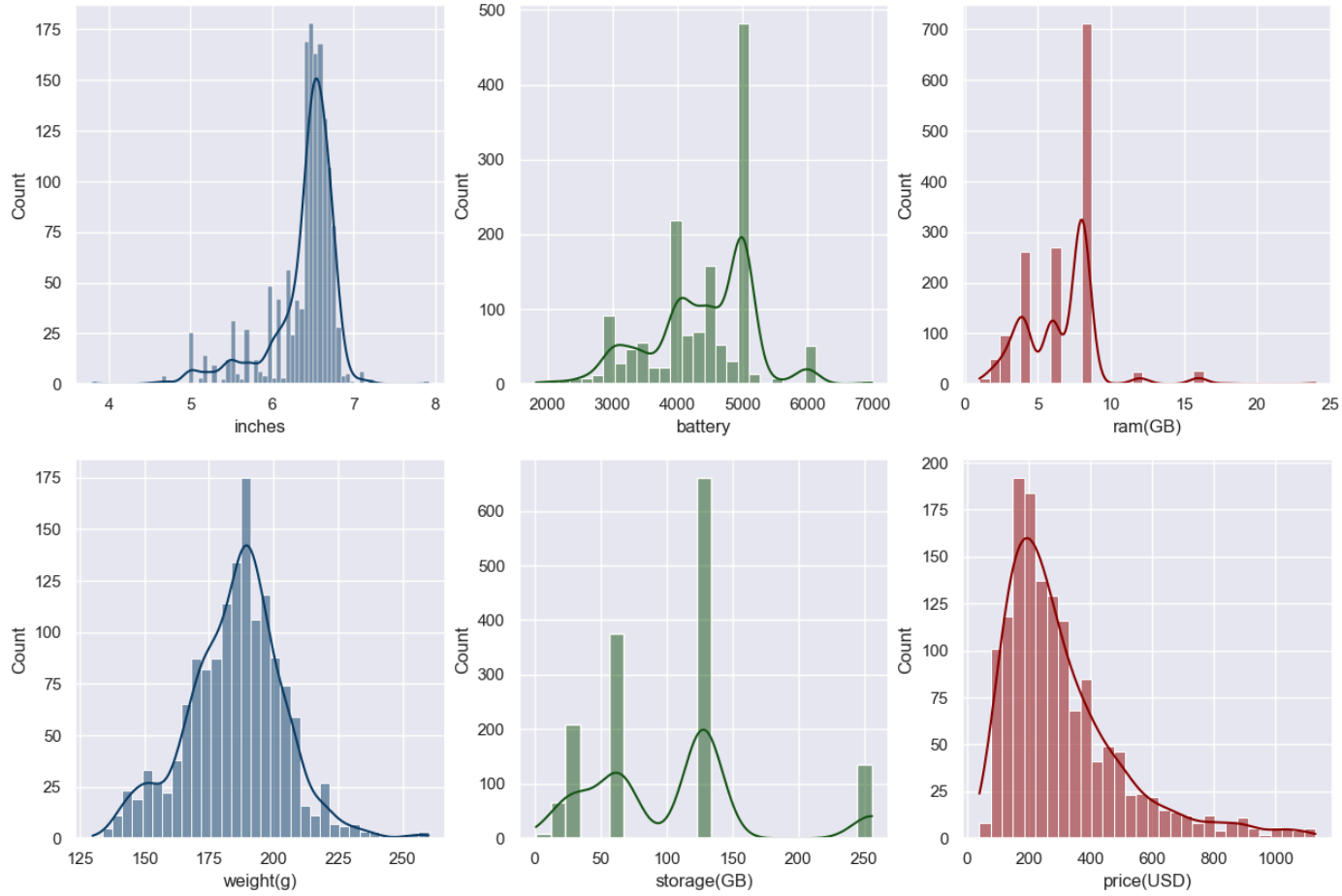



```
os_stats=df1.groupby('os')['os'].agg('count').sort_values(ascending=False).head(15)
os_stats.plot(kind='bar')
```

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



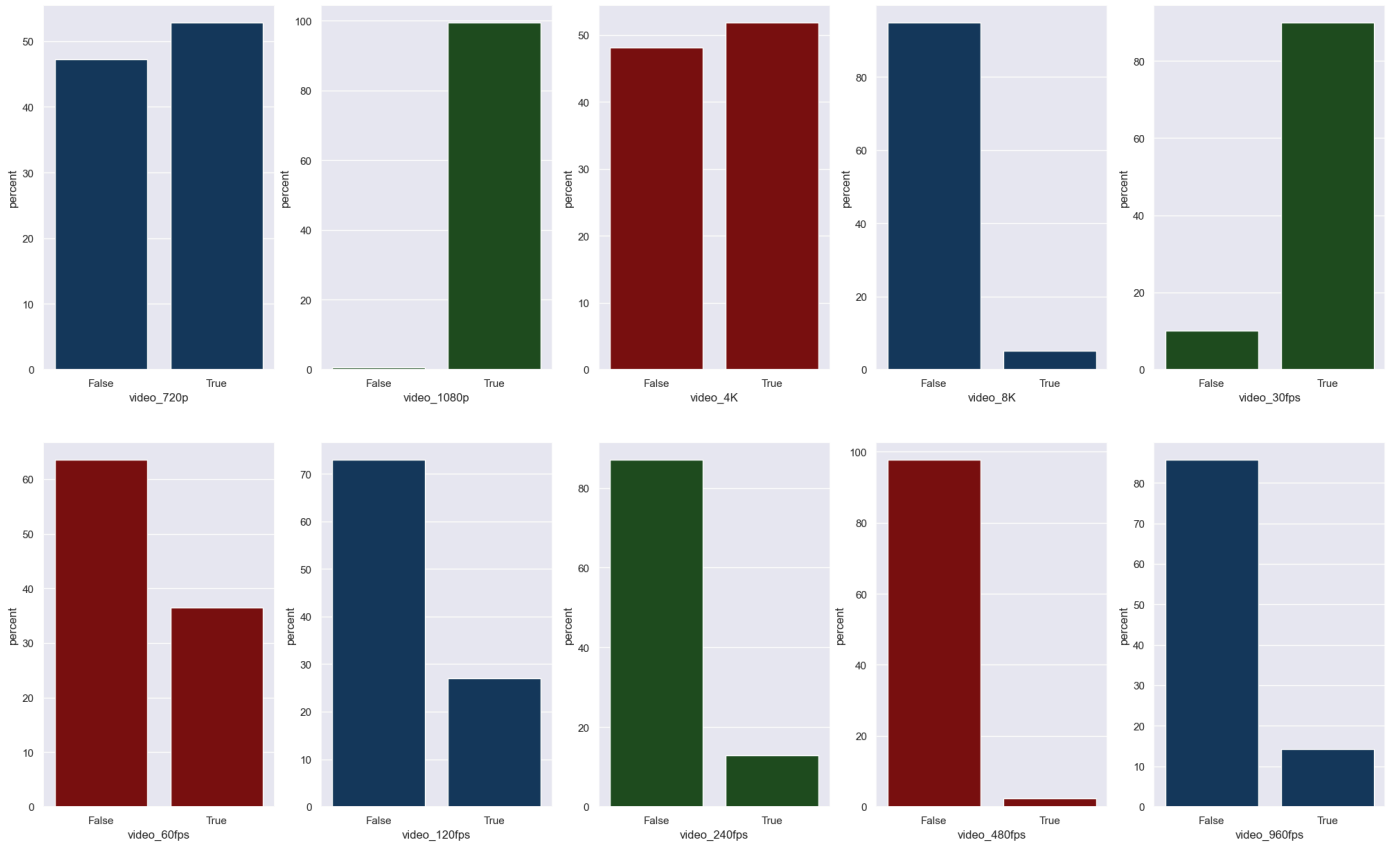

```
fig, ax = plt.subplots(2, 3, figsize=(15, 10))
features=['inches', 'battery',  'ram(GB)',  'weight(g)',    'storage(GB)',  'price(USD)']
colors = ['#0b3c68', '#175618', '#8b0000']
colors = (colors * (len(features) // len(colors) + 1))[:len(features)]
ax=ax.flatten()

for idx, (feature, color) in enumerate(zip(features, colors)):
    sns.histplot(df1, x=feature,color=color, ax=ax[idx], kde=True)

plt.show()
```




```
df1.columns
```

```
Index(['phone_name', 'brand', 'os', 'inches', 'battery', 'battery_type',
       'ram(GB)', 'announcement_date', 'weight(g)', 'storage(GB)',
       'video_720p', 'video_1080p', 'video_4K', 'video_8K', 'video_30fps',
       'video_60fps', 'video_120fps', 'video_240fps', 'video_480fps',
       'video_960fps', 'price(USD)', 'width', 'height', 'announcement_year'],
      dtype='object')
```

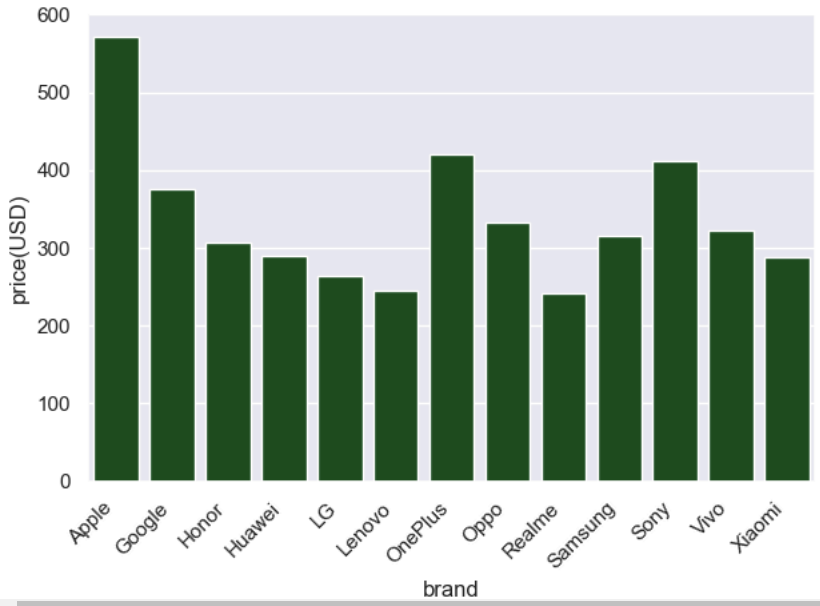
```
fig, ax = plt.subplots(2, 5, figsize=(25, 15))
features=['video_720p', 'video_1080p', 'video_4K', 'video_8K', 'video_30fps',
       'video_60fps', 'video_120fps', 'video_240fps', 'video_480fps',
       'video_960fps']
colors = ['#0b3c68', '#175618', '#8b0000']
colors = (colors * (len(features) // len(colors) + 1))[:len(features)]
ax=ax.flatten()

for idx, (feature, color) in enumerate(zip(features, colors)):
    sns.countplot(df1, x=feature,color=color, ax=ax[idx],stat='percent')

plt.show()
```




```
brand_price=df1.pivot_table(index='brand',values='price(USD)',aggfunc='mean')
brand_price
```

brand	price(USD)
Apple	572.038421
Google	375.753889
Honor	306.736552
Huawei	290.165410
LG	264.000000
Lenovo	244.358974
OnePlus	421.010723
Oppo	332.510603
Realme	242.407732
Samsung	315.821533
Sony	412.027429
Vivo	322.151640
Xiaomi	287.294286

```
sns.barplot(brand_price,x=brand_price.index,y=brand_price['price(USD)'],color='#175618')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

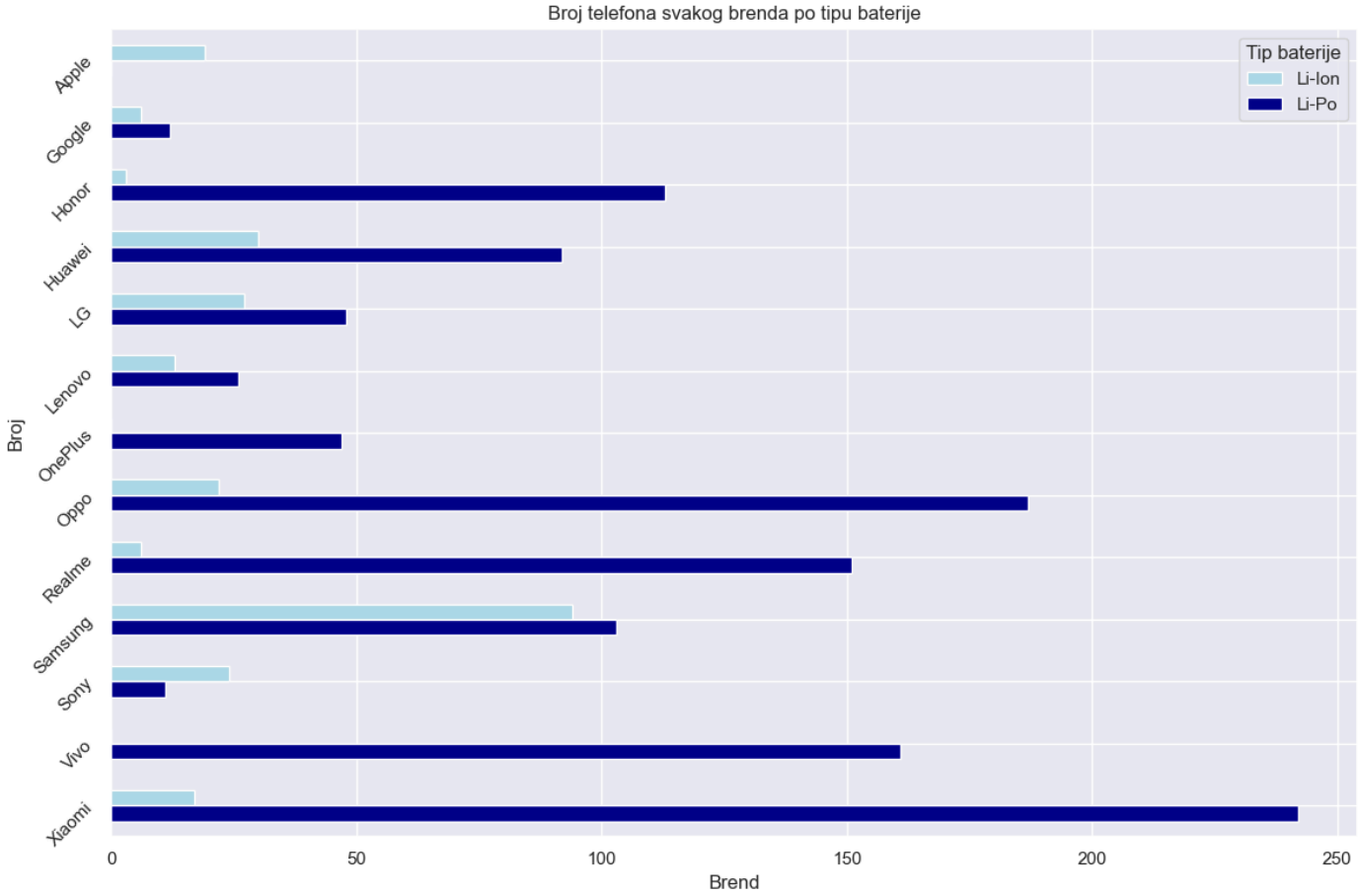



```
brand_battery_type=df1.pivot_table(index='brand',columns='battery_type',aggfunc='size')
brand_battery_type
```

brand	battery_type Li-Ion	Li-Po
Apple	19.0	NaN
Google	6.0	12.0
Honor	3.0	113.0
Huawei	30.0	92.0
LG	27.0	48.0
Lenovo	13.0	26.0
OnePlus	NaN	47.0
Oppo	22.0	187.0
Realme	6.0	151.0
Samsung	94.0	103.0
Sony	24.0	11.0
Vivo	NaN	161.0
Xiaomi	17.0	242.0

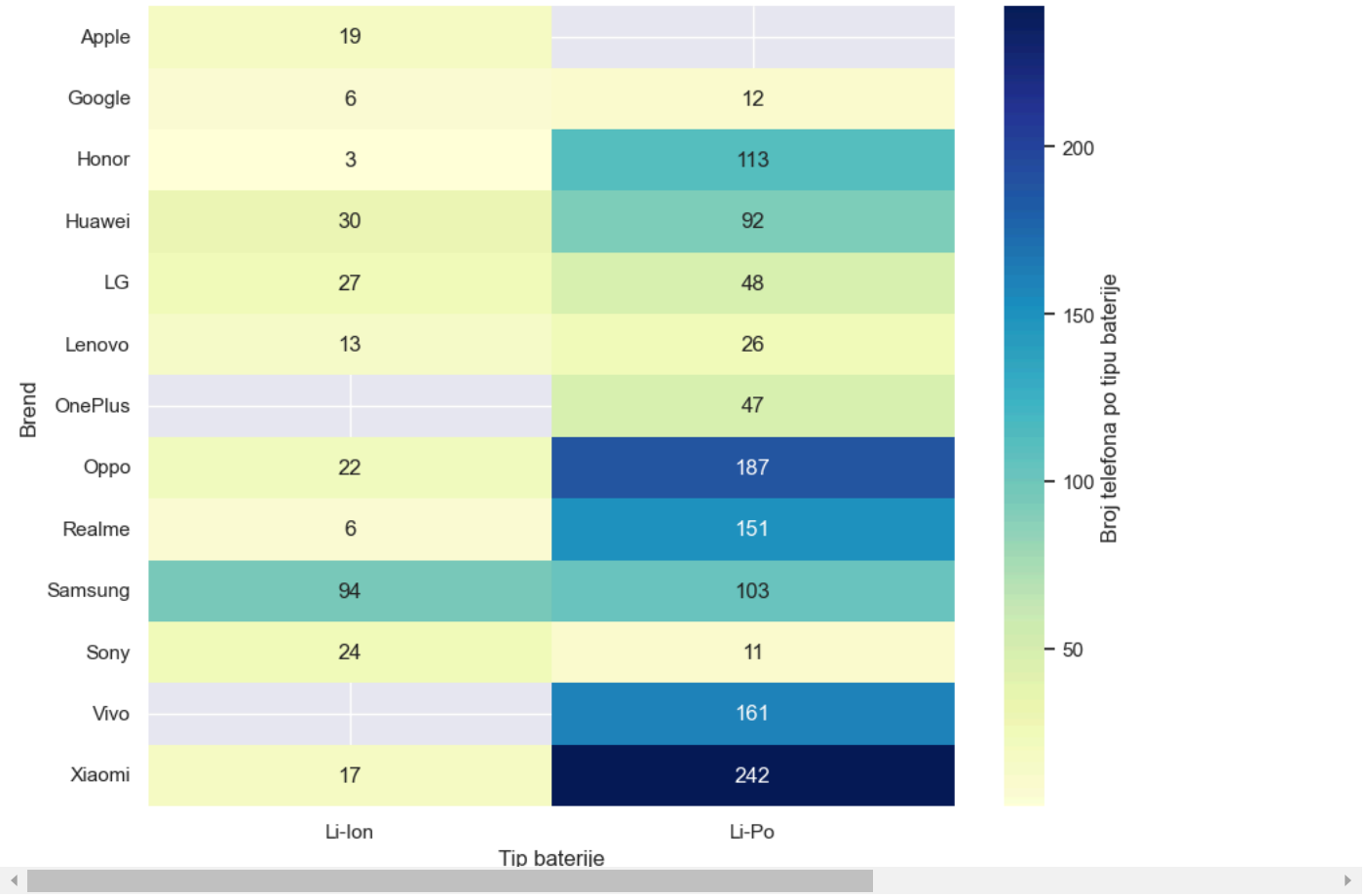
```
import matplotlib.colors as mcolors
plt.figure(figsize=(12, 8))
blue_cmap = mcolors.LinearSegmentedColormap.from_list('blue_cmap', ['lightblue', 'darkblue'])
ax = brand_battery_type.plot(kind='barh', figsize=(12, 8),colormap=blue_cmap)
ax.invert_yaxis()

plt.title('Broj telefona svakog brenda po tipu baterije')
plt.xlabel('Brend')
plt.ylabel('Broj')
plt.yticks(rotation=45, ha='right')
ax.legend(title='Tip baterije')

plt.tight_layout()
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```




```
plt.figure(figsize=(10, 8))
ax=sns.heatmap(brand_battery_type, annot=True, fmt='g',cmap='YlGnBu', cbar=True)
colorbar = ax.collections[0].colorbar
colorbar.set_label('Broj telefona po tipu baterije')
plt.xlabel('Tip baterije')
plt.ylabel('Brend')
```

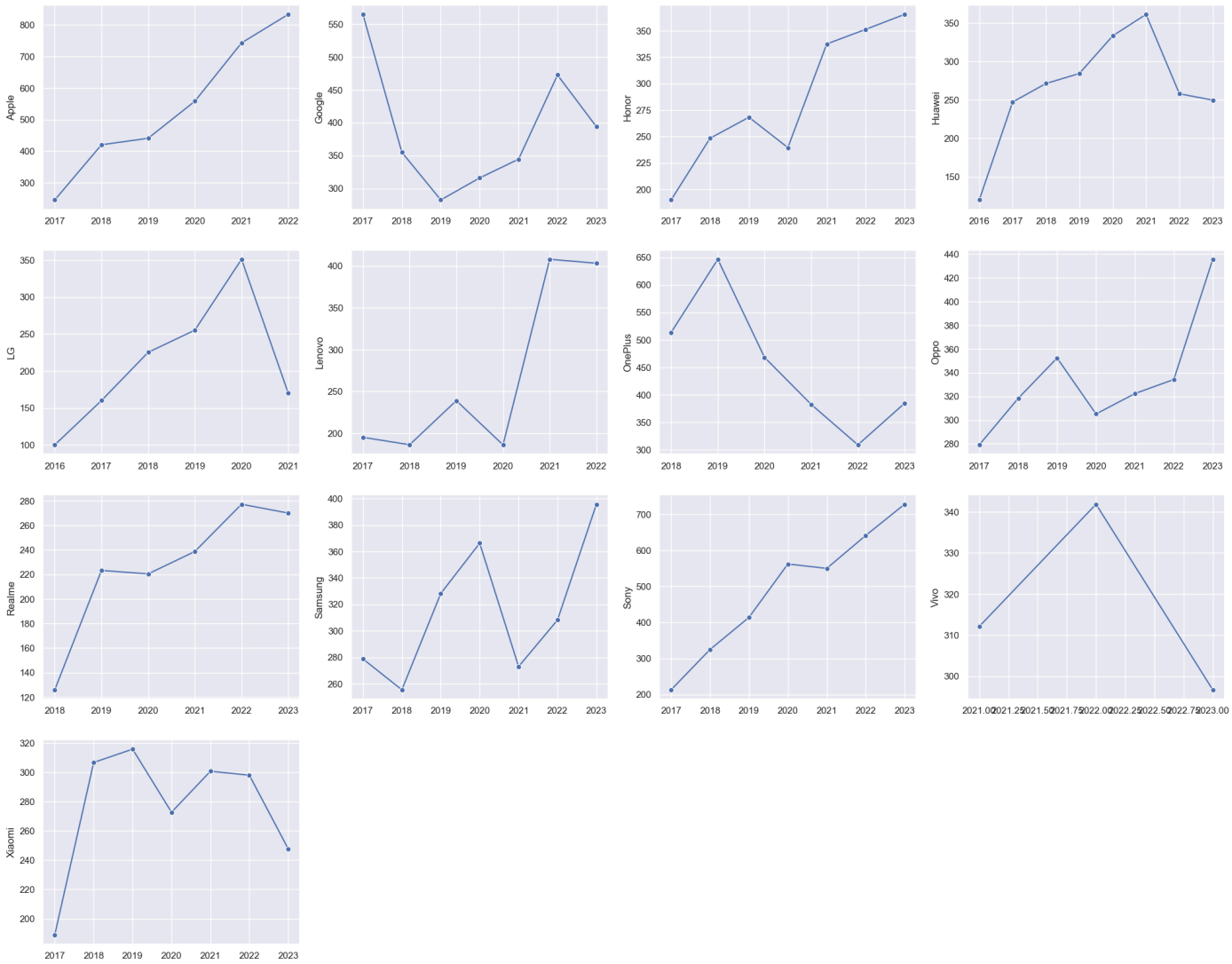
```
Text(92.25, 0.5, 'Brend')
```




```
year_price=df1.groupby(['announcement_year', 'brand'])['price(USD)'].mean().unstack()
year_price
```

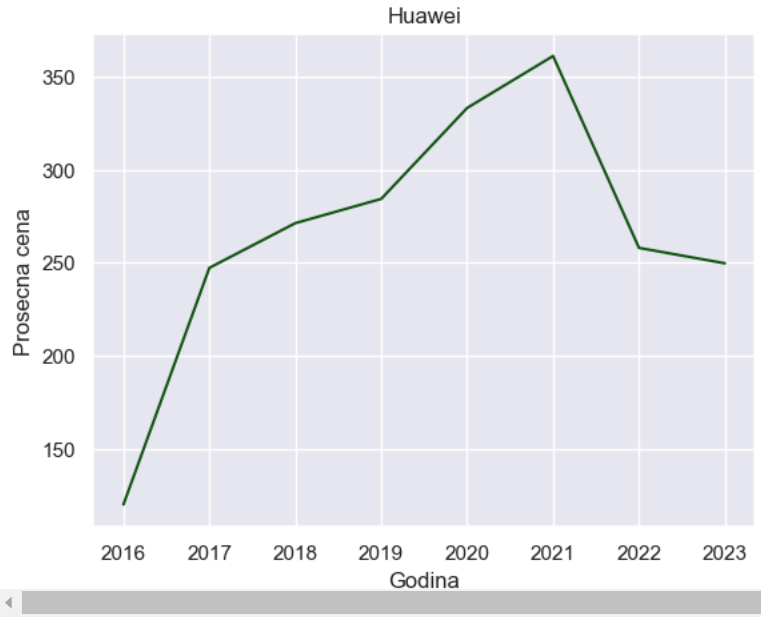
brand	Apple	Google	Honor	Huawei	LG	Lenovo	OnePlus	Oppo	Realme	Samsung	
announcement_year											
2016	NaN	NaN	NaN	120.000000	100.000000	NaN	NaN	NaN	NaN	NaN	
2017	245.000000	565.000000	190.000000	247.222222	160.000000	195.000000	NaN	279.166667	NaN	278.750000	2
2018	420.000000	355.000000	248.333333	271.250000	225.263158	186.153846	513.333333	318.352941	126.000000	255.416667	32
2019	440.556667	282.500000	268.235294	284.300000	255.263158	238.888889	646.416667	352.400000	223.285714	328.235294	4
2020	558.150000	316.000000	239.523810	333.168235	351.200000	186.000000	468.171111	305.195182	220.471415	366.268140	56
2021	743.206667	344.523333	337.619048	361.113333	170.000000	408.000000	382.512857	322.390667	238.710368	272.962895	5
2022	833.710000	472.666667	351.394400	257.963333	NaN	403.333333	308.992923	334.267404	277.243611	308.192462	64
2023	NaN	394.000000	365.609474	249.666667	NaN	NaN	384.551778	435.577895	270.040087	395.732375	7

```
fig, ax = plt.subplots(nrows=4, ncols=4, figsize=(25, 20))
ax = ax.flatten()
kolone=year_price.columns
brojac=0

for idx, brand in enumerate(year_price.columns):
    sns.lineplot(data=year_price[brand], ax=ax[idx], marker='o')
    ax[idx].set_ylabel(kolone[brojac])
    ax[idx].set_xlabel('')
    brojac=brojac+1

for i in range(len(year_price.columns), len(ax)):
    fig.delaxes(ax[i])
```




```
plt.plot(year_price.index,year_price['Huawei'],color='#175618')
plt.title('Huawei')
plt.xlabel('Godina')
plt.ylabel('Prosecna cena')
```

```
Text(0, 0.5, 'Prosecna cena')
```




```
df1=df1.reset_index()


df1['resolution']=df1['width']*df1['height']
df1.iloc[:,-4:]
```

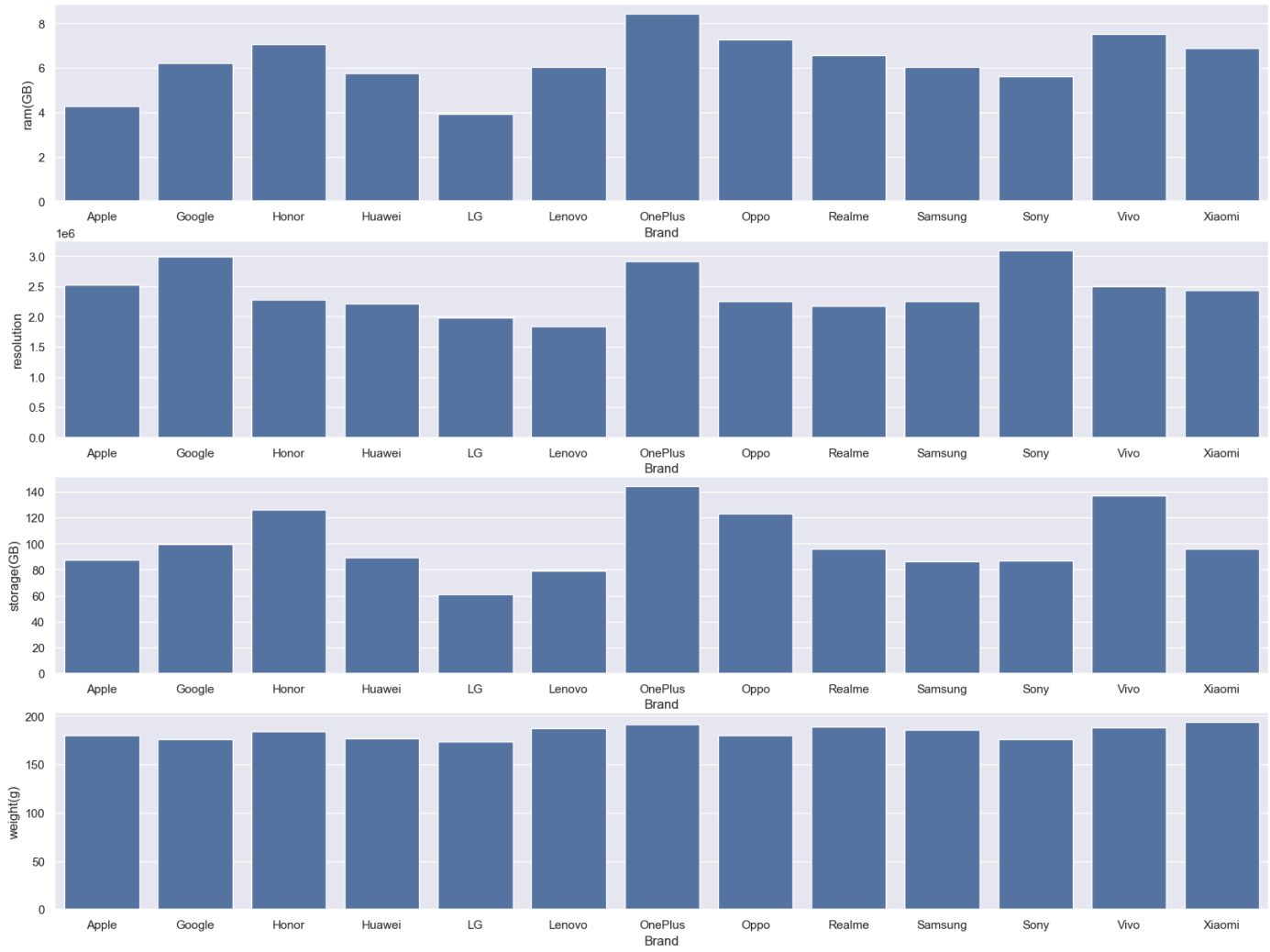
	width	height	announcement_year	resolution
0	720	1280	2016	921600
1	720	1280	2016	921600
2	1080	1920	2017	2073600
3	1080	1920	2017	2073600
4	1080	1920	2017	2073600
...	...	...	...	...
1449	1080	2388	2023	2579040
1450	1080	2400	2023	2592000
1451	1240	2772	2023	3437280
1452	1240	2772	2023	3437280
1453	1080	2400	2023	2592000

1454 rows × 4 columns

```
brand_specs = df1.pivot_table(index = 'brand', values = ['ram(GB)', 'weight(g)', 'storage(GB)','resolution'] )
brand_specs
```

brand	ram(GB)	resolution	storage(GB)	weight(g)
Apple	4.263158	2.522060e+06	87.578947	180.263158
Google	6.222222	2.995200e+06	99.555556	175.611111
Honor	7.043103	2.275056e+06	126.344828	184.362069
Huawei	5.745902	2.216465e+06	89.508197	177.057377
LG	3.933333	1.983651e+06	60.693333	173.386667
Lenovo	6.025641	1.835022e+06	78.769231	187.589744
OnePlus	8.425532	2.919523e+06	144.340426	191.553191
Oppo	7.282297	2.245713e+06	122.947368	179.956938
Realme	6.579618	2.172804e+06	95.802548	188.815287
Samsung	6.025381	2.257615e+06	86.091371	185.751269
Sony	5.600000	3.100937e+06	86.857143	175.542857
Vivo	7.521739	2.497380e+06	136.944099	188.167702
Xiaomi	6.872587	2.436203e+06	95.660232	193.664093

```
fig, axs = plt.subplots(4, 1, figsize=(20, 15))
axs = axs.flatten()

features = brand_specs.columns

for idx, feature in enumerate(features):
    sns.barplot(x=brand_specs.index, y=brand_specs[feature], ax=axs[idx])
    axs[idx].set_xlabel('Brand')
    axs[idx].set_ylabel(feature)
```




```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1454 entries, 0 to 1453
Data columns (total 25 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   index              1454 non-null   int64  
 1   phone_name         1454 non-null   object 
 2   brand              1454 non-null   object 
 3   os                 1454 non-null   object 
 4   inches             1454 non-null   float64
 5   battery            1454 non-null   int64  
 6   battery_type       1454 non-null   object 
 7   ram(GB)            1454 non-null   int64  
 8   weight(g)          1454 non-null   float64
 9   storage(GB)        1454 non-null   int64  
 10  video_720p         1454 non-null   bool   
 11  video_1080p        1454 non-null   bool   
 12  video_4K           1454 non-null   bool   
 13  video_8K           1454 non-null   bool   
```

```
 14  video_30fps        1454 non-null   bool
 15  video_60fps        1454 non-null   bool
 16  video_120fps       1454 non-null   bool
 17  video_240fps       1454 non-null   bool
 18  video_480fps       1454 non-null   bool
 19  video_960fps       1454 non-null   bool
 20  price(USD)         1454 non-null   float64
 21  width              1454 non-null   int64
 22  height             1454 non-null   int64
 23  announcement_year  1454 non-null   int32
 24  resolution         1454 non-null   int64
dtypes: bool(10), float64(3), int32(1), int64(7), object(4)
memory usage: 179.0+ KB
```

```
df1.drop('index',axis=1,inplace=True)
```

```
df2=df1.drop(['phone_name','os','battery_type','weight(g)'],axis=1)
```

```
df2
```

	brand	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	video_60fps	video_120fps
0	Huawei	5.00	2200	2	16	True	False	False	False	True	False	False
1	LG	5.30	2700	2	16	False	True	False	False	True	False	False
2	Huawei	5.20	3000	4	16	False	True	False	False	True	False	False
3	Xiaomi	5.50	4100	4	32	True	True	False	False	True	False	True
4	Huawei	5.10	3200	4	32	True	True	True	False	True	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
1449	Vivo	6.64	5000	8	256	False	True	False	False	True	False	False
1450	Realme	6.72	5000	8	128	False	True	False	False	True	False	False
1451	Realme	6.74	5240	16	256	True	True	True	False	False	True	False
1452	Realme	6.74	4600	24	1	True	True	True	False	False	True	False
1453	Vivo	6.78	4600	8	128	True	True	True	False	True	False	False

1454 rows × 20 columns

```
for kolona in ['video_720p','video_1080p','video_4K',   'video_8K', 'video_30fps'   ,'video_60fps', 'video_120fps','video_240fps',  'video_4
    df2[kolona]=df2[kolona].astype(int)
```

```
df2
```

	brand	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	video_60fps	video_120fps
0	Huawei	5.00	2200	2	16	1	0	0	0	1	0	0
1	LG	5.30	2700	2	16	0	1	0	0	1	0	0
2	Huawei	5.20	3000	4	16	0	1	0	0	1	0	0
3	Xiaomi	5.50	4100	4	32	1	1	0	0	1	0	1
4	Huawei	5.10	3200	4	32	1	1	1	0	1	1	0
...	...	...	...	...	...	...	...	...	...	...	...	...
1449	Vivo	6.64	5000	8	256	0	1	0	0	1	0	0
1450	Realme	6.72	5000	8	128	0	1	0	0	1	0	0
1451	Realme	6.74	5240	16	256	1	1	1	0	0	1	0
1452	Realme	6.74	4600	24	1	1	1	1	0	0	1	0
1453	Vivo	6.78	4600	8	128	1	1	1	0	1	0	0

1454 rows × 20 columns

```
dummies=pd.get_dummies(df2.brand)
dummies=dummies.astype(int)
dummies.head(3)
```

	Apple	Google	Honor	Huawei	LG	Lenovo	OnePlus	Oppo	Realme	Samsung	Sony	Vivo	Xiaomi
0	0	0	0	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0	0	0	0	0

```
df3=pd.concat([df2,dummies],axis='columns')
df3
```

	brand	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	...	Huawei	LG	Lenovo	O
0	Huawei	5.00	2200	2	16	1	0	0	0	1	...	1	0	0	
1	LG	5.30	2700	2	16	0	1	0	0	1	...	0	1	0	
2	Huawei	5.20	3000	4	16	0	1	0	0	1	...	1	0	0	
3	Xiaomi	5.50	4100	4	32	1	1	0	0	1	...	0	0	0	
4	Huawei	5.10	3200	4	32	1	1	1	0	1	...	1	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1449	Vivo	6.64	5000	8	256	0	1	0	0	1	...	0	0	0	
1450	Realme	6.72	5000	8	128	0	1	0	0	1	...	0	0	0	
1451	Realme	6.74	5240	16	256	1	1	1	0	0	...	0	0	0	
1452	Realme	6.74	4600	24	1	1	1	1	0	0	...	0	0	0	
1453	Vivo	6.78	4600	8	128	1	1	1	0	1	...	0	0	0	

1454 rows × 33 columns

```
df3.drop('brand',axis=1,inplace=True)
df3
```

	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	video_60fps	...	Huawei	LG	Leno
0	5.00	2200	2	16	1	0	0	0	1	0	...	1	0	
1	5.30	2700	2	16	0	1	0	0	1	0	...	0	1	
2	5.20	3000	4	16	0	1	0	0	1	0	...	1	0	
3	5.50	4100	4	32	1	1	0	0	1	0	...	0	0	
4	5.10	3200	4	32	1	1	1	0	1	1	...	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1449	6.64	5000	8	256	0	1	0	0	1	0	...	0	0	
1450	6.72	5000	8	128	0	1	0	0	1	0	...	0	0	
1451	6.74	5240	16	256	1	1	1	0	0	1	...	0	0	
1452	6.74	4600	24	1	1	1	1	0	0	1	...	0	0	
1453	6.78	4600	8	128	1	1	1	0	1	0	...	0	0	

1454 rows × 32 columns

```
modeli=df3.iloc[:,19:31]
modeli
```

	Apple	Google	Honor	Huawei	LG	Lenovo	OnePlus	Oppo	Realme	Samsung	Sony	Vivo
0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0
2	0	0	0	1	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...
1449	0	0	0	0	0	0	0	0	0	0	0	1
1450	0	0	0	0	0	0	0	0	1	0	0	0
1451	0	0	0	0	0	0	0	0	1	0	0	0
1452	0	0	0	0	0	0	0	0	1	0	0	0
1453	0	0	0	0	0	0	0	0	0	0	0	1

1454 rows × 12 columns

```
df4=df3.drop(modeli,axis='columns')
df4
```

	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	video_60fps	video_120fps	video_2
0	5.00	2200	2	16	1	0	0	0	1	0	0	
1	5.30	2700	2	16	0	1	0	0	1	0	0	
2	5.20	3000	4	16	0	1	0	0	1	0	0	
3	5.50	4100	4	32	1	1	0	0	1	0	1	
4	5.10	3200	4	32	1	1	1	0	1	1	0	
...	...	...	...	...	...	...	...	...	...	...	...	
1449	6.64	5000	8	256	0	1	0	0	1	0	0	
1450	6.72	5000	8	128	0	1	0	0	1	0	0	
1451	6.74	5240	16	256	1	1	1	0	0	1	0	
1452	6.74	4600	24	1	1	1	1	0	0	1	0	
1453	6.78	4600	8	128	1	1	1	0	1	0	0	

1454 rows × 20 columns

```
df5=df4[df4['Xiaomi']==1]
df5
```

	inches	battery	ram(GB)	storage(GB)	video_720p	video_1080p	video_4K	video_8K	video_30fps	video_60fps	video_120fps	video_2
3	5.50	4100	4	32	1	1	0	0	1	0	1	
10	5.50	4100	4	16	1	1	0	0	1	0	1	
22	5.15	3350	6	64	1	1	1	0	1	0	1	
29	6.44	5300	4	32	1	1	1	0	1	0	1	
30	5.00	4100	4	16	0	1	0	0	1	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	
1417	6.67	5000	12	256	0	1	0	0	0	1	0	
1419	6.79	5000	8	128	0	1	0	0	1	0	0	
1444	6.79	5000	8	128	0	1	0	0	1	0	0	
1445	6.79	5000	6	64	0	1	0	0	1	0	0	
1447	6.67	5000	24	256	1	1	1	1	0	1	0	

259 rows × 20 columns

```
X=df5.drop(['price(USD)','resolution','video_480fps','announcement_year'],axis=1)
y=df5['price(USD)']
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 259 entries, 3 to 1447
Data columns (total 16 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   inches        259 non-null    float64
 1   battery       259 non-null    int64
 2   ram(GB)       259 non-null    int64
 3   storage(GB)   259 non-null    int64
 4   video_720p    259 non-null    int32
 5   video_1080p   259 non-null    int32
 6   video_4K      259 non-null    int32
 7   video_8K      259 non-null    int32
 8   video_30fps   259 non-null    int32
 9   video_60fps   259 non-null    int32
 10  video_120fps  259 non-null    int32
 11  video_240fps  259 non-null    int32
 12  video_960fps  259 non-null    int32
 13  width         259 non-null    int64
 14  height        259 non-null    int64
 15  Xiaomi        259 non-null    int32
dtypes: float64(1), int32(10), int64(5)
memory usage: 24.3 KB
```

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

```
X.columns
```

```
Index(['inches', 'battery', 'ram(GB)', 'storage(GB)', 'video_720p',
       'video_1080p', 'video_4K', 'video_8K', 'video_30fps', 'video_60fps',
       'video_120fps', 'video_240fps', 'video_960fps', 'width', 'height',
       'Xiaomi'],
      dtype='object')
```

```
from sklearn.linear_model import Ridge
from sklearn.linear_model import LinearRegression

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=0)
X_train_scaled=scaler.fit_transform(X_train)
X_test_scaled=scaler.fit_transform(X_test)
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Lasso
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score

def find_best_model(X,y):
    algos={
        'linear_regression':{
          'model':LinearRegression(),
          'params':{
              'fit_intercept':[True,False]
          }

        },
        'lasso':{
            'model':Lasso(),
            'params':{
                'alpha':[1,2],
                'selection':['random','cyclic']
            }
        },
        'ridge': {
            'model':Ridge(),
            'params':{
                'alpha':[0.01,0.1,1,10,100],
                'max_iter':[100,1000,10000]

            }

        }
```

```
    }
    scores=[]
    cv=ShuffleSplit(n_splits=5, test_size=0.2, random_state=0)
    for algo_name,config in algos.items():
        gs=GridSearchCV(config['model'],config['params'],cv=cv,return_train_score=False)
        gs.fit(X,y)
        scores.append({
            'model':algo_name,
            'best_score':gs.best_score_,
            'best_params':gs.best_params_
        })

    return pd.DataFrame(scores,columns=['model','best_score','best_params'])


find_best_model(X,y)
```

	model	best_score	best_params
0	linear_regression	0.627059	{'fit_intercept': False}
1	lasso	0.645257	{'alpha': 2, 'selection': 'random'}
2	ridge	0.648051	{'alpha': 10, 'max_iter': 100}

```
def predict_price(model, scaler, inches, battery, ram, storage, video_720p, video_1080p, video_4K, video_8K,
                  video_30fps, video_60fps, video_120fps, video_240fps, video_960fps, width, height,Xiaomi):
    # Create a dictionary for the input data
    input_data = {
        'inches': inches,
        'battery': battery,
        'ram(GB)': ram,
        'storage(GB)': storage,
        'video_720p': video_720p,
        'video_1080p': video_1080p,
        'video_4K': video_4K,
        'video_8K': video_8K,
        'video_30fps': video_30fps,
        'video_60fps': video_60fps,
        'video_120fps': video_120fps,
        'video_240fps': video_240fps,
        'video_960fps': video_960fps,
        'width': width,
        'height': height,
        'Xiaomi': Xiaomi
    }
    input_df = pd.DataFrame([input_data])

    input_scaled = scaler.transform(input_df)
    predicted_price = model.predict(input_scaled)

    return predicted_price[0]


ridge_best_model = Ridge(alpha=10,max_iter=100)
ridge_best_model.fit(X_train_scaled, y_train)

predicted_price = predict_price(
    ridge_best_model, scaler,
    inches=6.5, battery=4000, ram=8, storage=128,
    video_720p=1, video_1080p=1, video_4K=1, video_8K=0,
    video_30fps=1, video_60fps=1, video_120fps=0, video_240fps=0, video_960fps=0,
    width=75, height=150, Xiaomi=1
)

print(f"Predicted price: {predicted_price:.2f} USD")
```

```
Predicted price: 142.20 USD
```