```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import numpy as np
```

```
df=pd.read_csv("shades.csv")
print(df)
```

```
          brand brand_short      product product_short     hex     H     S  \
0    Maybelline          mb       Fit Me           fmf  f3cfb3  26.0  0.26   
1    Maybelline          mb       Fit Me           fmf  ffe3c2  32.0  0.24   
2    Maybelline          mb       Fit Me           fmf  ffe0cd  23.0  0.20   
3    Maybelline          mb       Fit Me           fmf  ffd3be  19.0  0.25   
4    Maybelline          mb       Fit Me           fmf  bd9584  18.0  0.30   
..          ...         ...          ...           ...     ...   ...   ...   
620      L'Oréal          lo   True Match           tms  eecfba  24.0  0.22   
621      L'Oréal          lo   True Match           tms  e8c7b8  19.0  0.21   
622      L'Oréal          lo   True Match           tms  f0cbb9  20.0  0.23   
623      L'Oréal          lo   True Match           tms  e9c4b1  20.0  0.24   
624      L'Oréal          lo   True Match           tms  eabea1  24.0  0.31   

        V   L  group  
0    0.95  86      2  
1    1.00  92      2  
2    1.00  91      2  
3    1.00  88      2  
4    0.74  65      2  
..    ...  ..    ...  
620  0.93  85      7  
621  0.91  83      7  
622  0.94  85      7  
623  0.91  82      7  
624  0.92  80      7  

[625 rows x 10 columns]
```

```
df.head()
```

	brand	brand_short	product	product_short	hex	H	S	V	L	group
0	Maybelline	mb	Fit Me	fmf	f3cfb3	26.0	0.26	0.95	86	2
1	Maybelline	mb	Fit Me	fmf	ffe3c2	32.0	0.24	1.00	92	2
2	Maybelline	mb	Fit Me	fmf	ffe0cd	23.0	0.20	1.00	91	2
3	Maybelline	mb	Fit Me	fmf	ffd3be	19.0	0.25	1.00	88	2
4	Maybelline	mb	Fit Me	fmf	bd9584	18.0	0.30	0.74	65	2

```
df.tail(4)
```

	brand	brand_short	product	product_short	hex	H	S	V	L	group
621	L'Oréal	lo	True Match	tms	e8c7b8	19.0	0.21	0.91	83	7
622	L'Oréal	lo	True Match	tms	f0cbb9	20.0	0.23	0.94	85	7
623	L'Oréal	lo	True Match	tms	e9c4b1	20.0	0.24	0.91	82	7
624	L'Oréal	lo	True Match	tms	eabea1	24.0	0.31	0.92	80	7

```
df.columns
```

```
Index(['brand', 'brand_short', 'product', 'product_short', 'hex', 'H', 'S',
       'V', 'L', 'group'],
      dtype='object')
```

```
df.isnull().sum()
```

```
brand             0
brand_short       0
product           0
product_short     0
hex               0
H                12
S                12
V                12
L                 0
group             0
dtype: int64
```

```
df.describe()
```

	H	S	V	L	group
count	613.000000	613.000000	613.000000	625.000000	625.000000
mean	25.314845	0.459494	0.779543	65.920000	3.472000
std	5.327852	0.154089	0.173955	17.512267	1.976529
min	4.000000	0.100000	0.200000	11.000000	0.000000
25%	23.000000	0.350000	0.690000	55.000000	2.000000
50%	26.000000	0.440000	0.840000	71.000000	3.000000
75%	29.000000	0.560000	0.910000	79.000000	5.000000
max	45.000000	1.000000	1.000000	95.000000	7.000000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 625 entries, 0 to 624
Data columns (total 10 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   brand          625 non-null    object 
 1   brand_short    625 non-null    object 
 2   product        625 non-null    object 
 3   product_short  625 non-null    object 
 4   hex            625 non-null    object 
 5   H              613 non-null    float64
 6   S              613 non-null    float64
 7   V              613 non-null    float64
 8   L              625 non-null    int64  
 9   group          625 non-null    int64  
dtypes: float64(3), int64(2), object(5)
memory usage: 49.0+ KB
```

```
df.shape
```

```
(625, 10)
```

```
df.corr()
```

```
<ipython-input-10-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.c
  df.corr()
```

	H	S	V	L	group
H	1.000000	-0.166436	0.409831	0.451416	0.118561
S	-0.166436	1.000000	-0.707797	-0.810619	-0.048267
V	0.409831	-0.707797	1.000000	0.980690	0.165535
L	0.451416	-0.810619	0.980690	1.000000	0.132859
group	0.118561	-0.048267	0.165535	0.132859	1.000000

```
corr=df.corr()
corr.shape
```

```
<ipython-input-11-0a53fa01a22c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future ve
  corr=df.corr()
(5, 5)
```

```
df.nunique()
```

```
brand             36
brand_short       36
product           38
product_short     37
hex              617
H                 35
S                 74
V                 74
L                 78
group              8
dtype: int64
```

```
df.dtypes
```

```
brand             object
brand_short       object
product           object
product_short     object
hex               object
H                float64
S                float64
V                float64
L                  int64
group              int64
dtype: object
```

```
df.brand.value_counts()
```

```
Maybelline          54
Estée Lauder        42
MAC                 42
Make Up For Ever    40
Fenty               40
Lancôme             40
L'Oréal             36
Beauty Bakerie      30
Bobbi Brown         30
bareMinerals        29
Revlon              22
Black Up            18
Addiction           17
Laws of Nature      17
NARS                13
Trim & Prissy       13
Black Opal          12
Covergirl + Olay    12
House of Tara       11
Elsas Pro           11
Shu Uemera          11
Hegai and Ester     10
RMK                  9
Iman                 8
Bharat & Doris       7
Dior                 6
IPSA                 6
Kate                 6
Shiseido             6
Kuddy                5
Nykaa                5
Lakmé                4
Olivia               4
Lotus Herbals        4
Colorbar             3
Blue Heaven          2
Name: brand, dtype: int64
```

```
df.count()
```

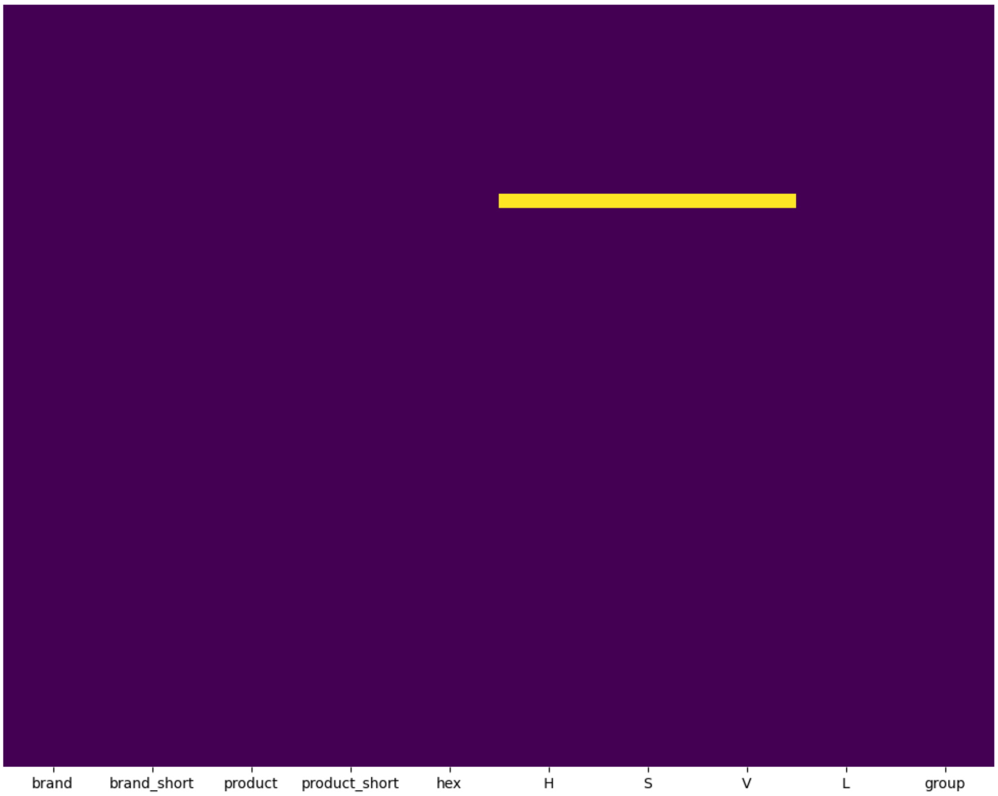
```
brand            625
brand_short      625
product          625
product_short    625
hex              625
H                613
S                613
V                613
L                625
group            625
dtype: int64
```

```
df.duplicated( )
```

```
0      False
1      False
2      False
3      False
4      False
       ...  
620    False
621    False
622    False
623    False
624    False
Length: 625, dtype: bool
```

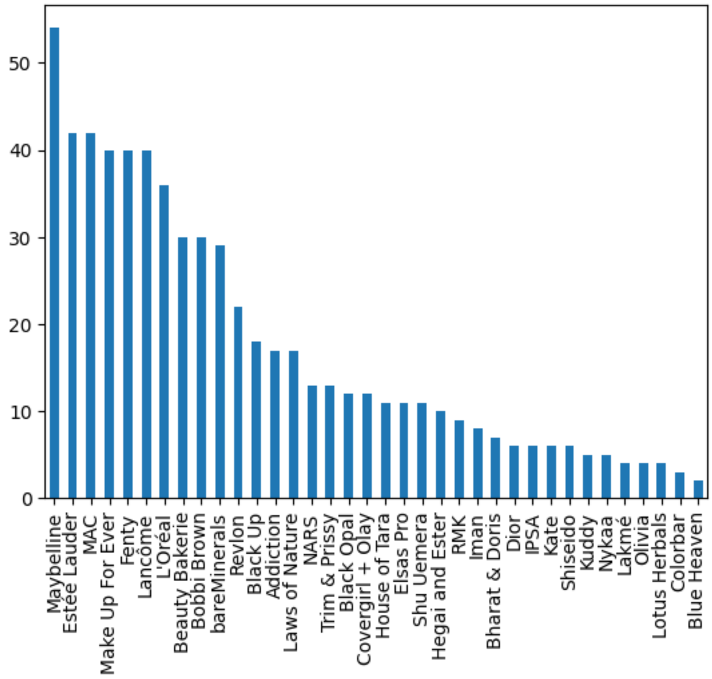
```
import seaborn as sns
import matplotlib.pyplot as plt
def get_heatmap(df):
    #This function gives heatmap of all NaN values
    plt.figure(figsize=(10,8))
    sns.heatmap(df.isnull(), yticklabels=False, cbar=False, cmap='viridis')
    plt.tight_layout()
    return plt.show()

get_heatmap(df)
```



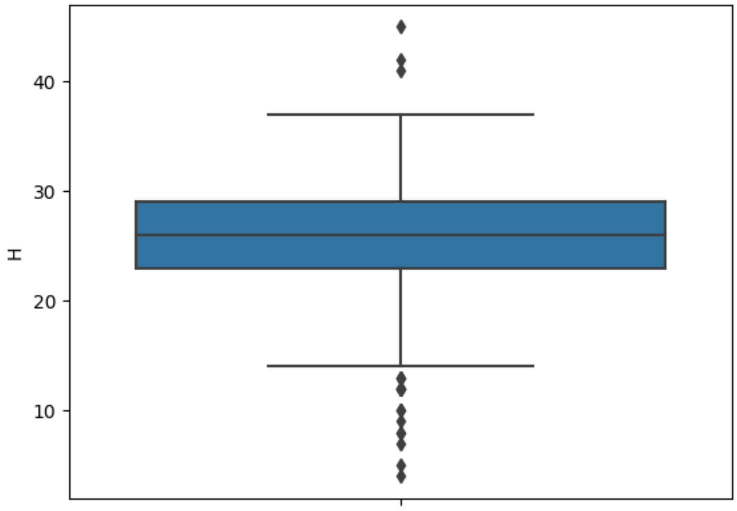
```
df.brand.value_counts().plot.bar(_)
```

```
<Axes: >
```



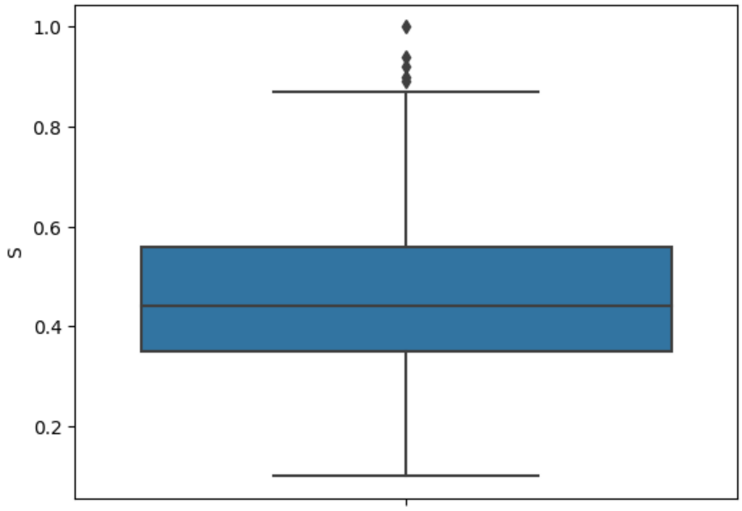
```
sns.boxplot(y="H",data=df)
```

```
<Axes: ylabel='H'>
```



```
sns.boxplot(y="S",data=df)
```

```
<Axes: ylabel='S'>
```



```
print(df.isna().sum())
```

```
brand             0
brand_short       0
product           0
product_short     0
hex               0
H                12
S                12
V                12
L                 0
group             0
dtype: int64
```

```
ds=df.H.mean()
ds
```

```
25.31484502446982
```

```
df.H.fillna(ds)
```

```
0      26.0
1      32.0
2      23.0
3      19.0
4      18.0
       ...
620    24.0
621    19.0
622    20.0
623    20.0
```

```
624    24.0
Name: H, Length: 625, dtype: float64
```

```
print(df.isna().sum())
```

```
brand            0
brand_short      0
product          0
product_short    0
hex              0
H               12
S               12
V               12
L                0
group            0
dtype: int64
```

```
mean_value=df['H'].mean()
```

```
df['H'].fillna(value=mean_value,inplace=True)
df
```

	brand	brand_short	product	product_short	hex	H	S	V	L	group
0	Maybelline	mb	Fit Me	fmf	f3cfb3	26.0	0.26	0.95	86	2
1	Maybelline	mb	Fit Me	fmf	ffe3c2	32.0	0.24	1.00	92	2
2	Maybelline	mb	Fit Me	fmf	ffe0cd	23.0	0.20	1.00	91	2
3	Maybelline	mb	Fit Me	fmf	ffd3be	19.0	0.25	1.00	88	2
4	Maybelline	mb	Fit Me	fmf	bd9584	18.0	0.30	0.74	65	2
...	...	...	...	...	...	...	...	...	...	...
620	L'Oréal	lo	True Match	tms	eecfba	24.0	0.22	0.93	85	7
621	L'Oréal	lo	True Match	tms	e8c7b8	19.0	0.21	0.91	83	7
622	L'Oréal	lo	True Match	tms	f0cbb9	20.0	0.23	0.94	85	7
623	L'Oréal	lo	True Match	tms	e9c4b1	20.0	0.24	0.91	82	7
624	L'Oréal	lo	True Match	tms	eabea1	24.0	0.31	0.92	80	7

625 rows × 10 columns

```
mean_value=df['S'].mean()
```

```
df['S'].fillna(value=mean_value,inplace=True)
df
```

	brand	brand_short	product	product_short	hex	H	S	V	L	group
0	Maybelline	mb	Fit Me	fmf	f3cfb3	26.0	0.26	0.95	86	2
1	Maybelline	mb	Fit Me	fmf	ffe3c2	32.0	0.24	1.00	92	2
2	Maybelline	mb	Fit Me	fmf	ffe0cd	23.0	0.20	1.00	91	2
3	Maybelline	mb	Fit Me	fmf	ffd3be	19.0	0.25	1.00	88	2
4	Maybelline	mb	Fit Me	fmf	bd9584	18.0	0.30	0.74	65	2
...	...	...	...	...	...	...	...	...	...	...
620	L'Oréal	lo	True Match	tms	eecfba	24.0	0.22	0.93	85	7
621	L'Oréal	lo	True Match	tms	e8c7b8	19.0	0.21	0.91	83	7
622	L'Oréal	lo	True Match	tms	f0cbb9	20.0	0.23	0.94	85	7
623	L'Oréal	lo	True Match	tms	e9c4b1	20.0	0.24	0.91	82	7
624	L'Oréal	lo	True Match	tms	eabea1	24.0	0.31	0.92	80	7

625 rows × 10 columns

```
mean_value=df['V'].mean()
```

```
df['V'].fillna(value=mean_value,inplace=True)
df
```

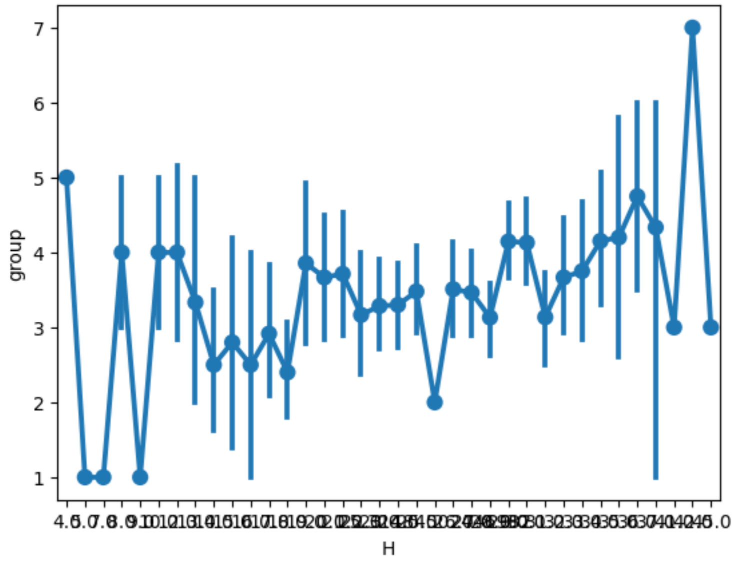
	brand	brand_short	product	product_short	hex	H	S	V	L	group
0	Maybelline	mb	Fit Me	fmf	f3cfb3	26.0	0.26	0.95	86	2
1	Maybelline	mb	Fit Me	fmf	ffe3c2	32.0	0.24	1.00	92	2
2	Maybelline	mb	Fit Me	fmf	ffe0cd	23.0	0.20	1.00	91	2
3	Maybelline	mb	Fit Me	fmf	ffd3be	19.0	0.25	1.00	88	2
4	Maybelline	mb	Fit Me	fmf	bd9584	18.0	0.30	0.74	65	2
...	...	...	...	...	...	...	...	...	...	...
620	L'Oréal	lo	True Match	tms	eecfba	24.0	0.22	0.93	85	7
621	L'Oréal	lo	True Match	tms	e8c7b8	19.0	0.21	0.91	83	7
622	L'Oréal	lo	True Match	tms	f0cbb9	20.0	0.23	0.94	85	7
623	L'Oréal	lo	True Match	tms	e9c4b1	20.0	0.24	0.91	82	7
624	L'Oréal	lo	True Match	tms	eabea1	24.0	0.31	0.92	80	7

```
df.isna().sum()
```

```
brand            0
brand_short      0
product          0
product_short    0
hex              0
H                0
S                0
V                0
L                0
group            0
dtype: int64
```

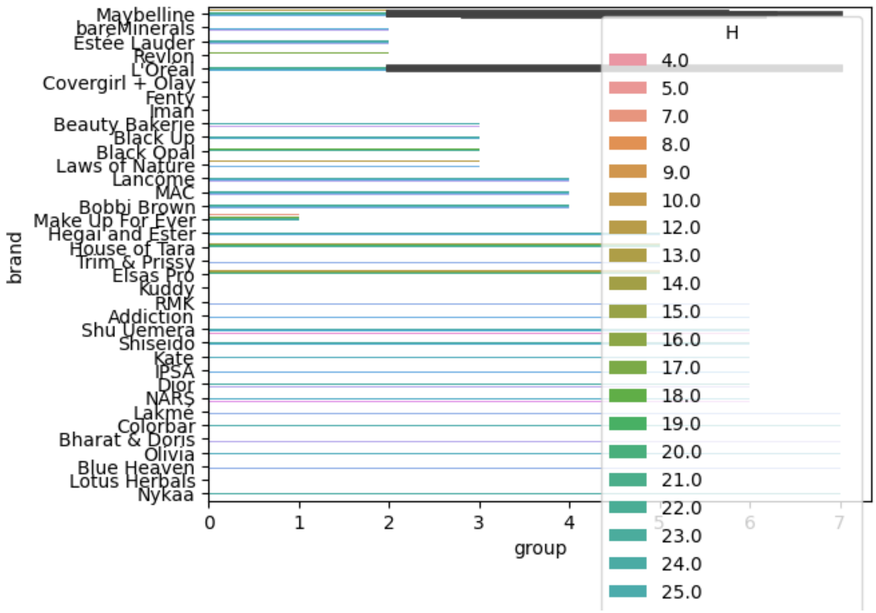
```
sns.pointplot(data=df,x="H",y="group")
```

```
<Axes: xlabel='H', ylabel='group'>
```



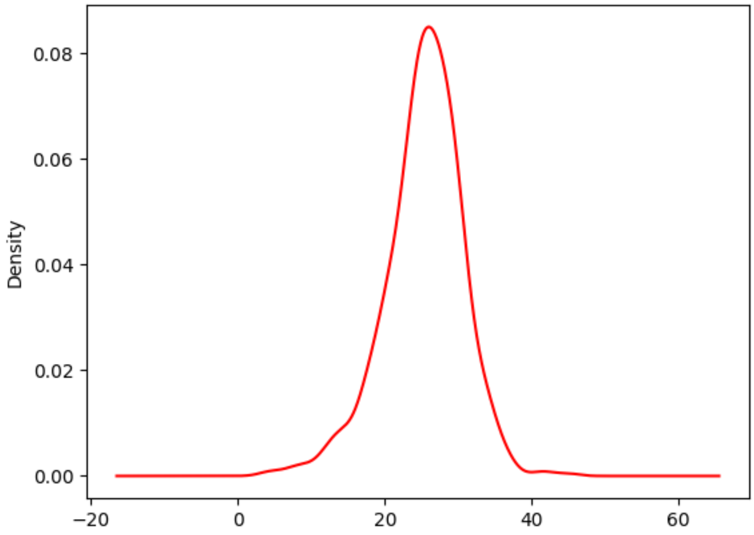
```
sns.barplot(x="group",y="brand",data=df,hue="H")
```

```
<Axes: xlabel='group', ylabel='brand'>
```



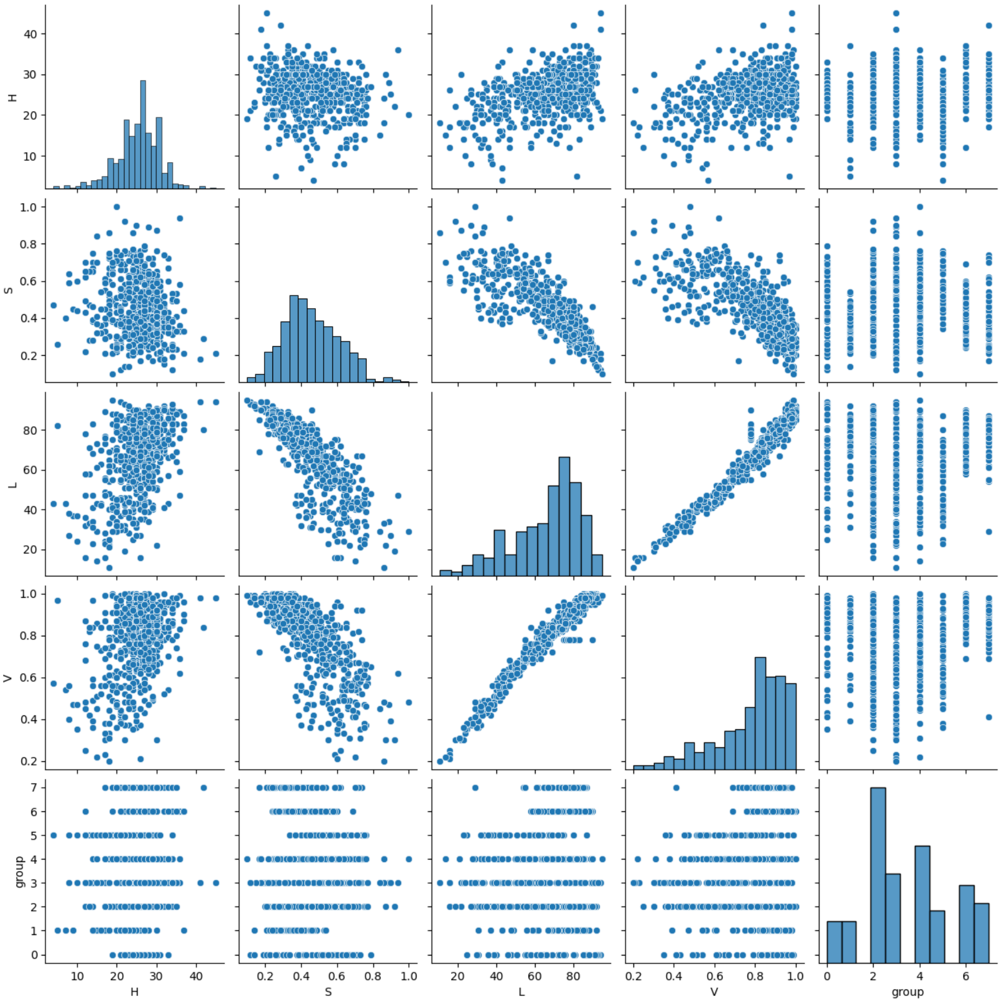
```
df.H.plot.kde(color="red")
```

```
<Axes: ylabel='Density'>
```



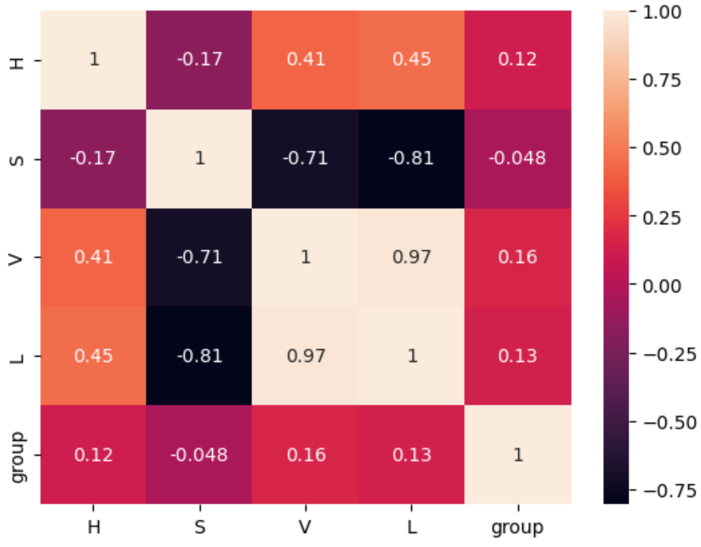
```
df2=df[["brand","H","S","L","V","group"]]
plt.figure()
sns.pairplot(df2)
plt.show()
```

```
<Figure size 640x480 with 0 Axes>
```



```
sns.heatmap(df.corr(), annot=True)
```

```
<ipython-input-36-f169729a0461>:1: FutureWarning: The default value of numeric_only in DataFrame.c
  sns.heatmap(df.corr(), annot=True)
<Axes: >
```



CLUSTERING

```
import seaborn as sns
from sklearn.cluster import KMeans
```

```
X=df[['H','L','S','V']].copy()
```

```
df.brand
```

```
0      Maybelline
1      Maybelline
2      Maybelline
3      Maybelline
4      Maybelline
          ...
620       L'Oréal
621       L'Oréal
622       L'Oréal
623       L'Oréal
624       L'Oréal
Name: brand, Length: 625, dtype: object
```

```
print(X)
```

```
        H   L     S     V
0    26.0  86  0.26  0.95
1    32.0  92  0.24  1.00
2    23.0  91  0.20  1.00
3    19.0  88  0.25  1.00
4    18.0  65  0.30  0.74
..    ...  ..   ...   ...
620  24.0  85  0.22  0.93
621  19.0  83  0.21  0.91
622  20.0  85  0.23  0.94
623  20.0  82  0.24  0.91
624  24.0  80  0.31  0.92

[625 rows x 4 columns]
```

```
wcss=[]
for i in range (1,11):
  km=KMeans(n_clusters=i, random_state=0)
  km.fit(X)
  wcss.append(km.inertia_)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change fro
  warnings.warn(
```

```
print(wcss)
```

```
[208773.28512561176, 68103.11646453798, 40100.01758317912, 30169.277013951607, 24460.43347795205, 20653.447951924318, 18296.9899019
```

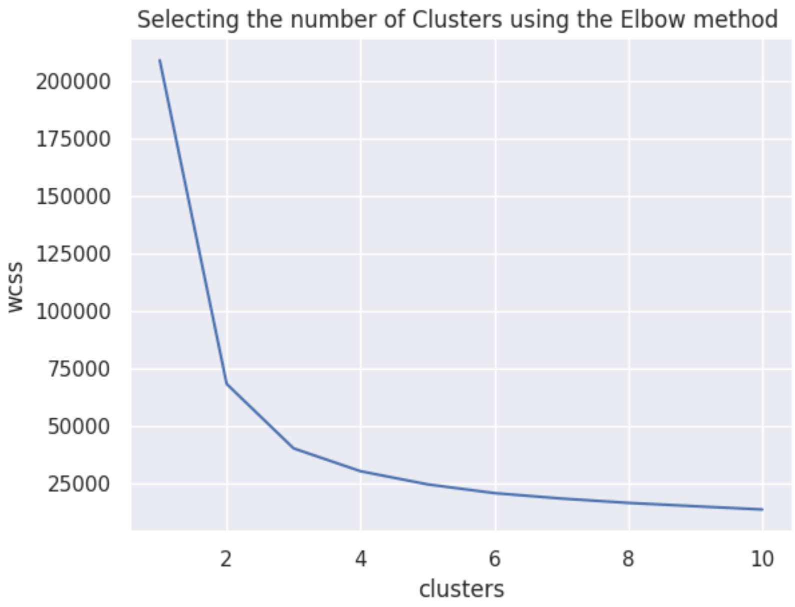
```
print(km)
```

```
KMeans(n_clusters=10, random_state=0)
```

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

```
sns.set()
```

```
plt.title("Selecting the number of Clusters using the Elbow method ")
plt.plot(range(1,11), wcss)
plt.xlabel('clusters')
plt.ylabel('wcss')
plt.show()
```



```
km=KMeans(n_clusters=2, random_state=0)
km.fit(X)
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default \
  warnings.warn(
▾             KMeans
KMeans(n_clusters=2, random_state=0)
```

```
print(km.cluster_centers_)
b=0
g=0
aa=km.labels_
aa1=np.array(aa)
for i in range(len(aa1)):
  if(aa1[i]==0):
    g=g+1
  else:
    b=b+1
```

```
[[26.69431488 76.07981221  0.38827214  0.87752234]
 [22.36180905 44.17085427  0.6119598   0.56979899]]
```

```
print(km.cluster_centers_)
aa=km.labels_
aa1=np.array(aa)
print(aa1)
```

```
[[26.69431488 76.07981221  0.38827214  0.87752234]
 [22.36180905 44.17085427  0.6119598   0.56979899]]
[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1
 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0
 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1
 1 1 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 1 1 1 0 0 1 1 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1 1 1 1 1 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1
 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 1 1 1
 0 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1
 1 1 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0]
```

```
print(g)
print(b)
```

```
426
199
```