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

df = pd.read_csv("HappinessAlcoholConsumption.csv")
df.head()
```

```
       Country          Region Hemisphere  HappinessScore  HDI
GDP_PerCapita  \
0      Denmark  Western Europe      north           7.526  928
53.579
1  Switzerland  Western Europe      north           7.509  943
79.866
2      Iceland  Western Europe      north           7.501  933
60.530
3       Norway  Western Europe      north           7.498  951
70.890
4      Finland  Western Europe      north           7.413  918
43.433

   Beer_PerCapita  Spirit_PerCapita  Wine_PerCapita
0             224                81             278
1             185               100             280
2             233                61              78
3             169                71             129
4             263               133              97
```

this dataframe contains 121 countries and the amount and type of alcohol in each one of them.

```
df = df.drop_duplicates().dropna().drop(columns="GDP_PerCapita")
```

cleaning data. gpd percapita removed because it's data was unacurate.

```
encoder = LabelEncoder()
encoded_country_list = []
df["encoded_country"] = encoder.fit_transform(df["Country"])

i = 0
for encoded in encoder.inverse_transform(df["encoded_country"]):
    encoded_country_list.append((i, encoded))
    i+=1
print(encoded_country_list)

df["encoded_region"] = encoder.fit_transform(df["Region"])
print(encoder.inverse_transform([0, 1, 2, 3, 4, 5]))
```

```
df["encoded_hemisphere"] = encoder.fit_transform(df["Hemisphere"])
print(encoder.inverse_transform([0, 1, 2]))
```

```
[(0, 'Denmark'), (1, 'Switzerland'), (2, 'Iceland'), (3, 'Norway'),
(4, 'Finland'), (5, 'Canada'), (6, 'Netherlands'), (7, 'New Zealand'),
(8, 'Australia'), (9, 'Sweden'), (10, 'Israel'), (11, 'Austria'), (12,
'United States'), (13, 'Costa Rica'), (14, 'Germany'), (15, 'Brazil'),
(16, 'Belgium'), (17, 'Ireland'), (18, 'Luxembourg'), (19, 'Mexico'),
(20, 'Singapore'), (21, 'United Kingdom'), (22, 'Chile'), (23,
'Panama'), (24, 'Argentina'), (25, 'Czech Republic'), (26, 'United
Arab Emirates'), (27, 'Uruguay'), (28, 'Malta'), (29, 'Colombia'),
(30, 'France'), (31, 'Thailand'), (32, 'Qatar'), (33, 'Spain'), (34,
'Guatemala'), (35, 'Suriname'), (36, 'Bahrain'), (37, 'Trinidad and
Tobago'), (38, 'Venezuela'), (39, 'Slovakia'), (40, 'El Salvador'),
(41, 'Nicaragua'), (42, 'Uzbekistan'), (43, 'Italy'), (44, 'Ecuador'),
(45, 'Belize'), (46, 'Japan'), (47, 'Kazakhstan'), (48, 'Moldova'),
(49, 'Russian Federation'), (50, 'Poland'), (51, 'South Korea'), (52,
'Bolivia'), (53, 'Lithuania'), (54, 'Belarus'), (55, 'Slovenia'), (56,
'Peru'), (57, 'Turkmenistan'), (58, 'Mauritius'), (59, 'Latvia'), (60,
'Cyprus'), (61, 'Paraguay'), (62, 'Romania'), (63, 'Estonia'), (64,
'Jamaica'), (65, 'Croatia'), (66, 'Turkey'), (67, 'Jordan'), (68,
'Azerbaijan'), (69, 'Philippines'), (70, 'China'), (71, 'Kyrgyzstan'),
(72, 'Serbia'), (73, 'Bosnia and Herzegovina'), (74, 'Montenegro'),
(75, 'Dominican Republic'), (76, 'Morocco'), (77, 'Hungary'), (78,
'Lebanon'), (79, 'Portugal'), (80, 'Macedonia'), (81, 'Vietnam'), (82,
'Tunisia'), (83, 'Greece'), (84, 'Mongolia'), (85, 'Nigeria'), (86,
'Honduras'), (87, 'Zambia'), (88, 'Albania'), (89, 'Sierra Leone'),
(90, 'Namibia'), (91, 'Cameroon'), (92, 'South Africa'), (93,
'Egypt'), (94, 'Armenia'), (95, 'Kenya'), (96, 'Ukraine'), (97,
'Ghana'), (98, 'Dem. Rep. Congo'), (99, 'Georgia'), (100, 'Rep.
Congo'), (101, 'Senegal'), (102, 'Bulgaria'), (103, 'Zimbabwe'), (104,
'Malawi'), (105, 'Gabon'), (106, 'Mali'), (107, 'Haiti'), (108,
'Botswana'), (109, 'Comoros'), (110, "Cote d'Ivoire"), (111,
'Cambodia'), (112, 'Angola'), (113, 'Niger'), (114, 'Chad'), (115,
'Burkina Faso'), (116, 'Madagascar'), (117, 'Tanzania'), (118,
'Liberia'), (119, 'Benin'), (120, 'Togo'), (121, 'Syria')]
['Australia and New Zealand' 'Central and Eastern Europe' 'Eastern
Asia'
 'Latin America and Caribbean' 'Middle East and Northern Africa'
 'North America']
['both' 'north' 'noth']
```

encoding non_number values for heatmap.

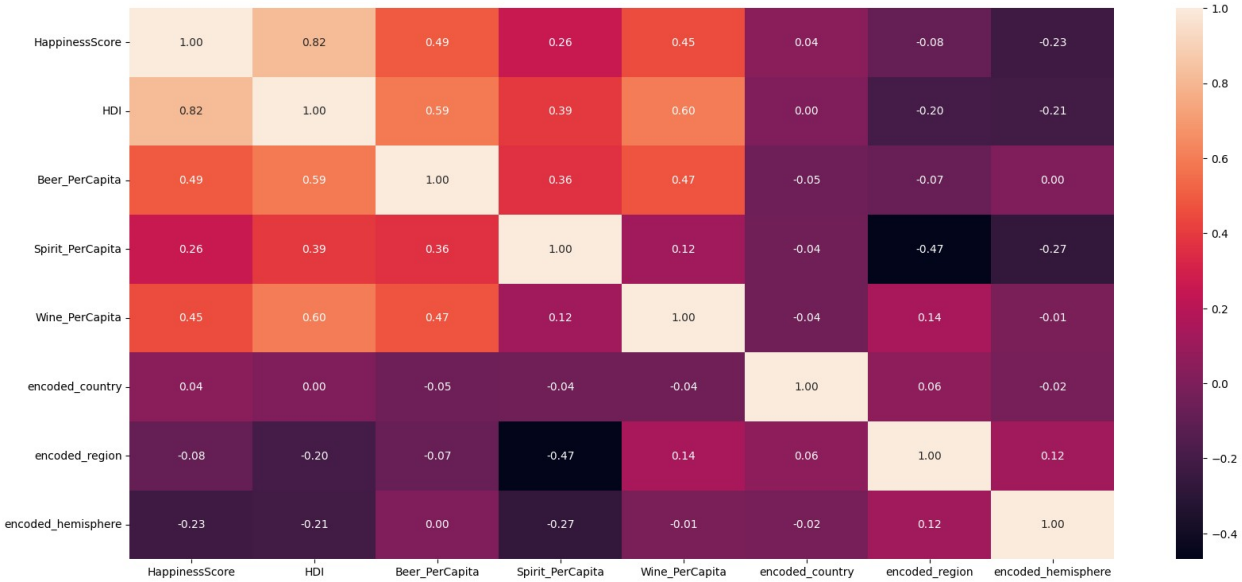
```
df.head()
```

```
       Country          Region Hemisphere  HappinessScore  HDI  \
0      Denmark  Western Europe      north           7.526  928
1  Switzerland  Western Europe      north           7.509  943
```

```
2       Iceland  Western Europe       north            7.501  933
3        Norway  Western Europe       north            7.498  951
4       Finland  Western Europe       north            7.413  918

   Beer_PerCapita  Spirit_PerCapita  Wine_PerCapita
encoded_country  \
0             224                81             278                32

1             185               100             280               103

2             233                61              78                49

3             169                71             129                81

4             263               133              97                38


   encoded_region  encoded_hemisphere
0               8                   1
1               8                   1
2               8                   1
3               8                   1
4               8                   1
```

```
plt.figure(figsize=(20,9))
tmp = df.drop(columns=["Country","Region","Hemisphere"])
sns.heatmap(tmp.corr(), annot=True, fmt="0.2f")
```

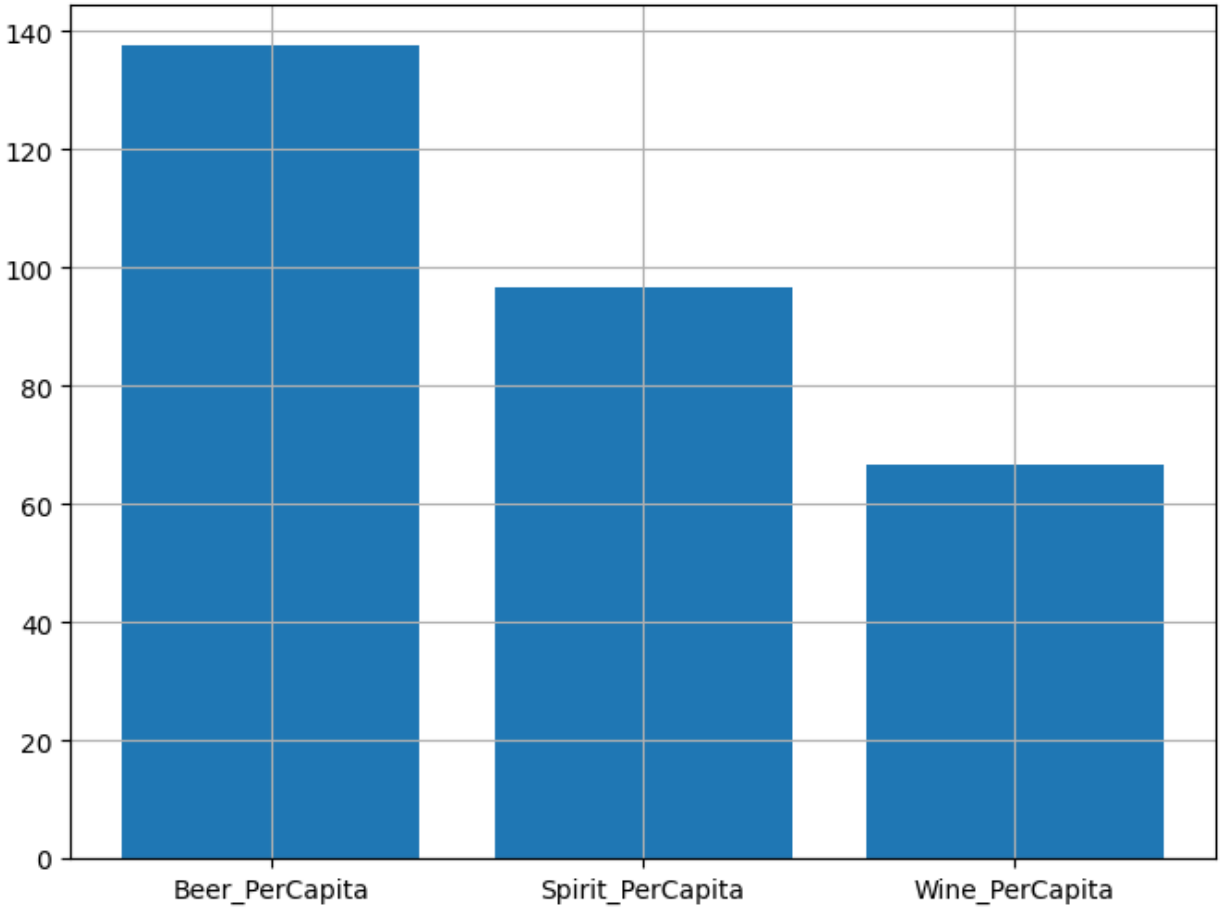
```
<Axes: >
```



heatmap shows:

1. HDI and Happiness score are directly dependent on each other; meaning that the higher the HDI the higher the happines score is.
2. HDI with wine percapita and beer percapita are half correlated. the heatmap shows that the higher the HDI goes drinking wine and beer increases.
3. Happiness with beer and wine percapita are half correlated.

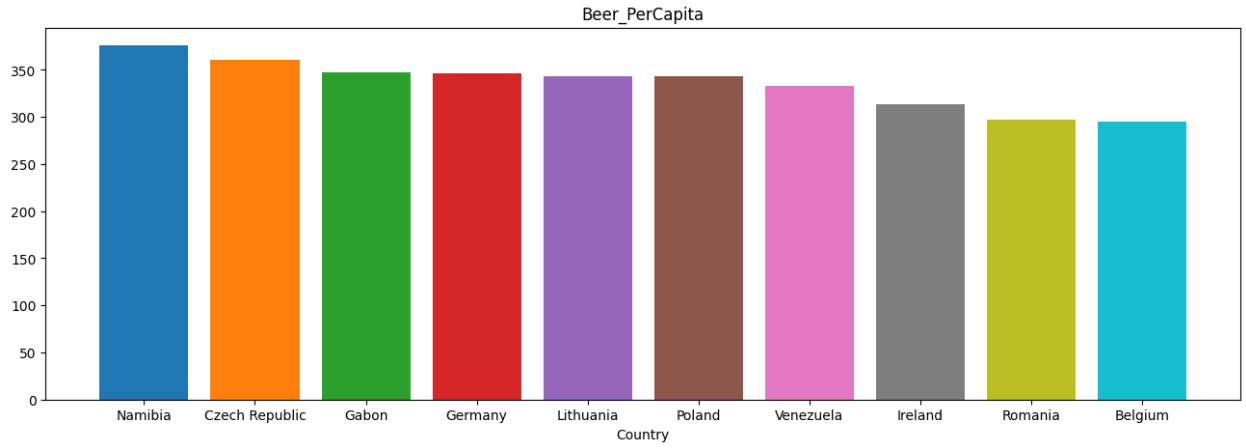
```
fig = plt.figure(figsize = (8,6))
ax = fig.add_subplot()
x = [0,1,2]
y = [df["Beer_PerCapita"].mean(), df["Spirit_PerCapita"].mean(),
df["Wine_PerCapita"].mean()]
ax.bar(x,y)
ax.set_xticks(x, ["Beer_PerCapita", "Spirit_PerCapita",
"Wine_PerCapita"], rotation=0)
ax.grid()
```



this plot shows that how much of each type of alcohol has been consumed.

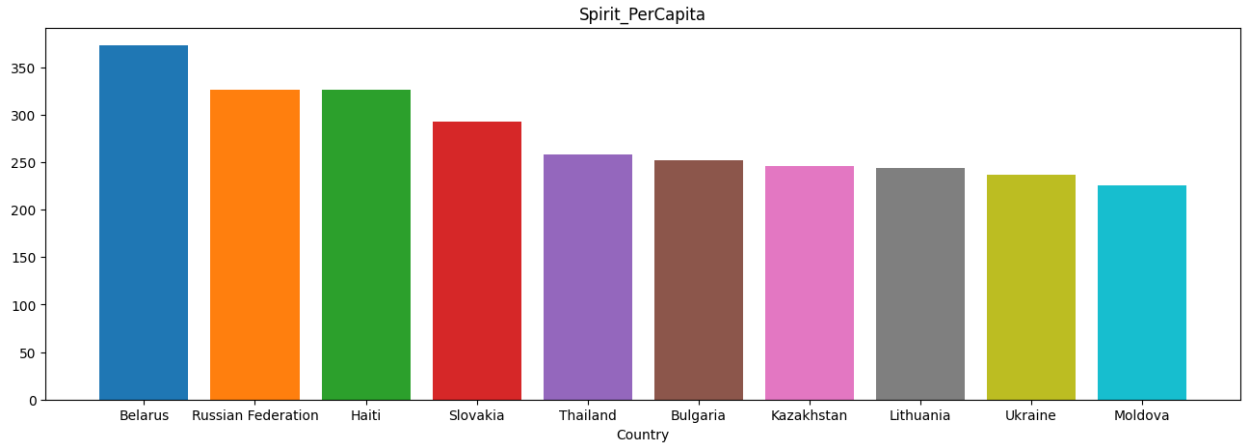
```
fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot()
ax.set_xlabel("Country")
```

```
ax.set_title("Beer_PerCapita")

j = 0
for i in df.sort_values(by=["Beer_PerCapita"], ascending=False).index:
    j = j + 1
    if j <= 10:
        ax.bar(df.iloc[i]["Country"], df.iloc[i]["Beer_PerCapita"])
    elif j > 10:
        break
```



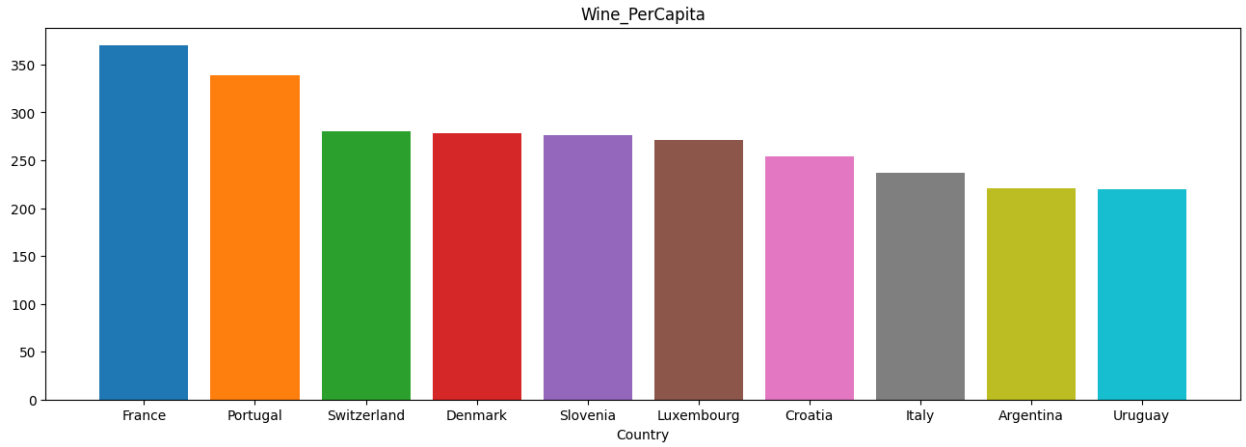
according to this plot this are the top 10 countries wich has the most beer consumption.

```
fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot()
ax.set_xlabel("Country")
ax.set_title("Spirit_PerCapita")

j = 0
for i in df.sort_values(by=["Spirit_PerCapita"],
ascending=False).index:
    j = j + 1
    if j <= 10:
        ax.bar(df.iloc[i]["Country"], df.iloc[i]["Spirit_PerCapita"])
    elif j > 10:
        break
```



according to this plot this are the top 10 countries wich has the most spirit consumption.

```
fig = plt.figure(figsize=(16,5))
ax = fig.add_subplot()
ax.set_xlabel("Country")
ax.set_title("Wine_PerCapita")

j = 0
for i in df.sort_values(by=["Wine_PerCapita"], ascending=False).index:
    j = j + 1
    if j <= 10:
        ax.bar(df.iloc[i]["Country"], df.iloc[i]["Wine_PerCapita"])
    elif j > 10:
        break
```



according to this plot this are the top 10 countries wich has the most wine consumption.

```
a_df = pd.DataFrame()
a_df["country"]  =df["Country"]
a_df["alcohol"] = df["Beer_PerCapita"] + df["Spirit_PerCapita"] +
df["Wine_PerCapita"]
```

```
a_df["HappinessScore"] = df["HappinessScore"]
a_df["HDI"] = df["HDI"]
a_df["Beer_PerCapita"] = df["Beer_PerCapita"]
a_df["Spirit_PerCapita"] = df["Spirit_PerCapita"]
a_df["Wine_PerCapita"] = df["Wine_PerCapita"]

descending_a_df = a_df.sort_values(by=["alcohol"], ascending=False)
ascending_a_df = a_df.sort_values(by=["alcohol"], ascending=True)

a_df.head()
```

```
       country  alcohol  HappinessScore  HDI  Beer_PerCapita  \
0      Denmark      583           7.526  928             224
1  Switzerland      565           7.509  943             185
2      Iceland      372           7.501  933             233
3       Norway      369           7.498  951             169
4      Finland      493           7.413  918             263

   Spirit_PerCapita  Wine_PerCapita
0                81             278
1               100             280
2                61              78
3                71             129
4               133              97
```

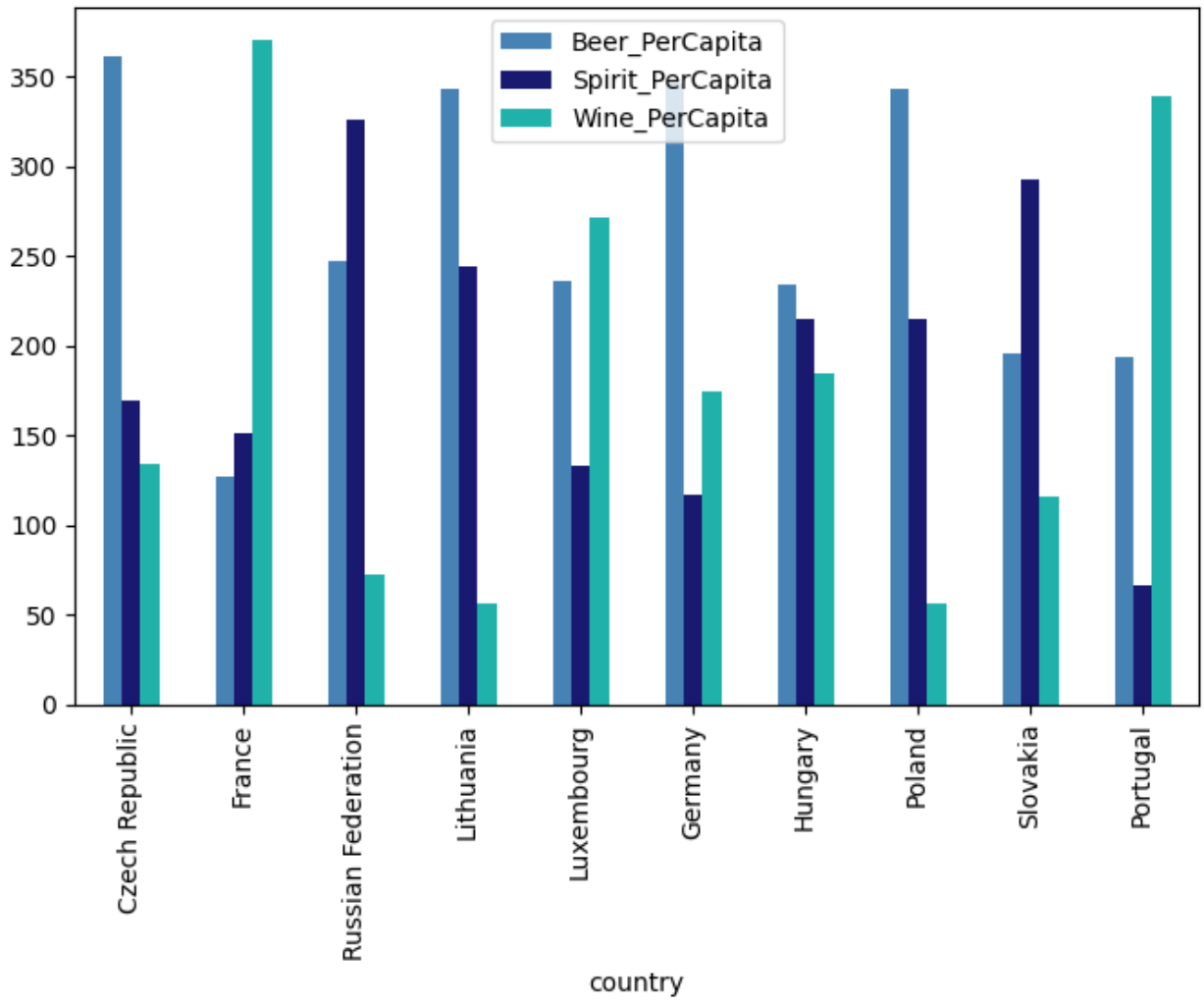
Three dataframes are made:

1. a_df: a dataframe that has an extra column "alcohol", which is ("Beer_PerCapita" + "Spirit_PerCapita" + "Wine_PerCapita") for each country.
2. descending_a_df: sorted descending type of a_df by "alcohol"
3. ascending_a_df: sorted ascending type of a_df by "alcohol"

```
new_df = pd.DataFrame(columns=["country", "alcohol", "HappinessScore",
"HDI", "Beer_PerCapita", "Spirit_PerCapita", "Wine_PerCapita"])
j = 0
for i in descending_a_df.index:
    j = j + 1
    if j <= 10:
        new_df1 = pd.DataFrame([a_df.iloc[i]])
        new_df = pd.concat([new_df, new_df1], ignore_index=True)
    elif j > 10:
        break

my_color =["steelblue", "midnightblue","lightseagreen"]
new_df.plot(x="country", y=["Beer_PerCapita", "Spirit_PerCapita",
"Wine_PerCapita"],
            kind="bar",figsize=(8,5), color=my_color)
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_5572\1186906962.py:7:
FutureWarning: The behavior of DataFrame concatenation with empty or
```

```
all-NA entries is deprecated. In a future version, this will no longer
exclude empty or all-NA columns when determining the result dtypes. To
retain the old behavior, exclude the relevant entries before the
concat operation.
  new_df = pd.concat([new_df, new_df1], ignore_index=True)

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

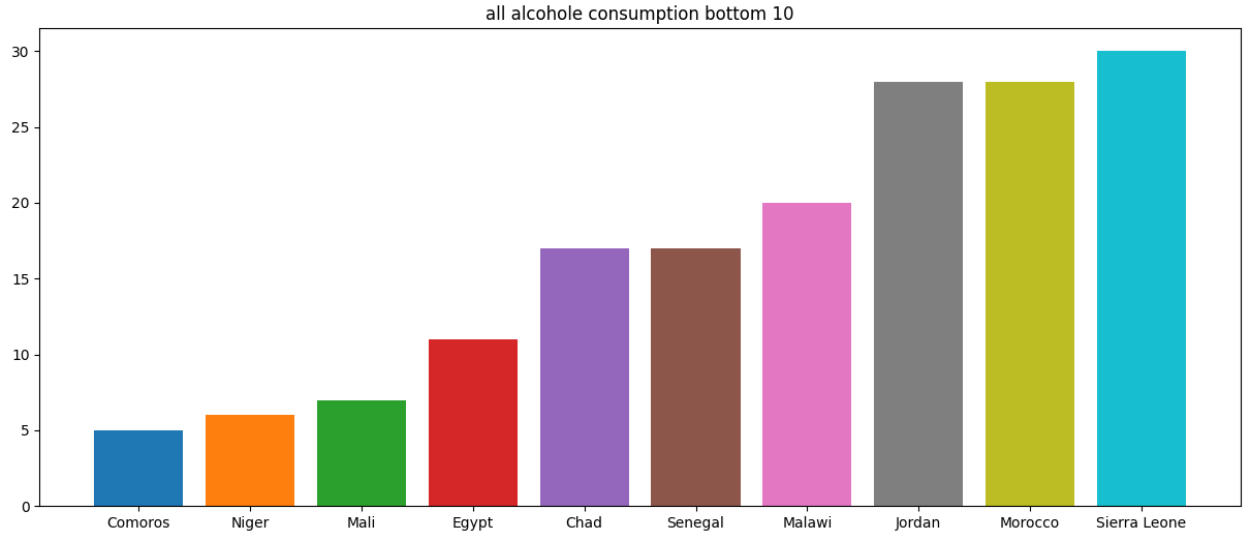


This plot shows top 10 countries which have the most alcohol consumption divided by each type of alcohol. It is intersting that the most used type of alcohol in each of the top 3 countries are completly differnt:

1. Czech Republic: beer
2. France: wine
3. Russian Federation: spirit

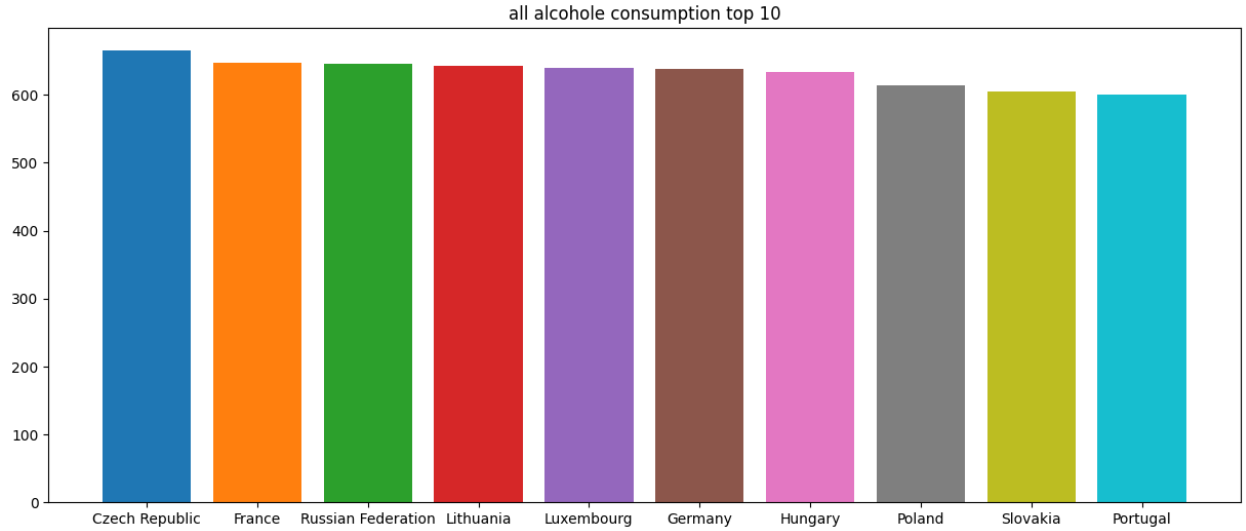
people of 5 of this countries prefered drinking beer over the other two.

```
fig =plt.figure(figsize=(15,6))
ax = fig.add_subplot()
ax.set_title("all alcohole consumption bottom 10")

j = 0
for i in ascending_a_df.index:
    j = j + 1
    if j <= 10:
        ax.bar(a_df.iloc[i]["country"], a_df.iloc[i]["alcohol"])
    elif j > 10:
        break
```



10 bottom countries that consume alcohol.

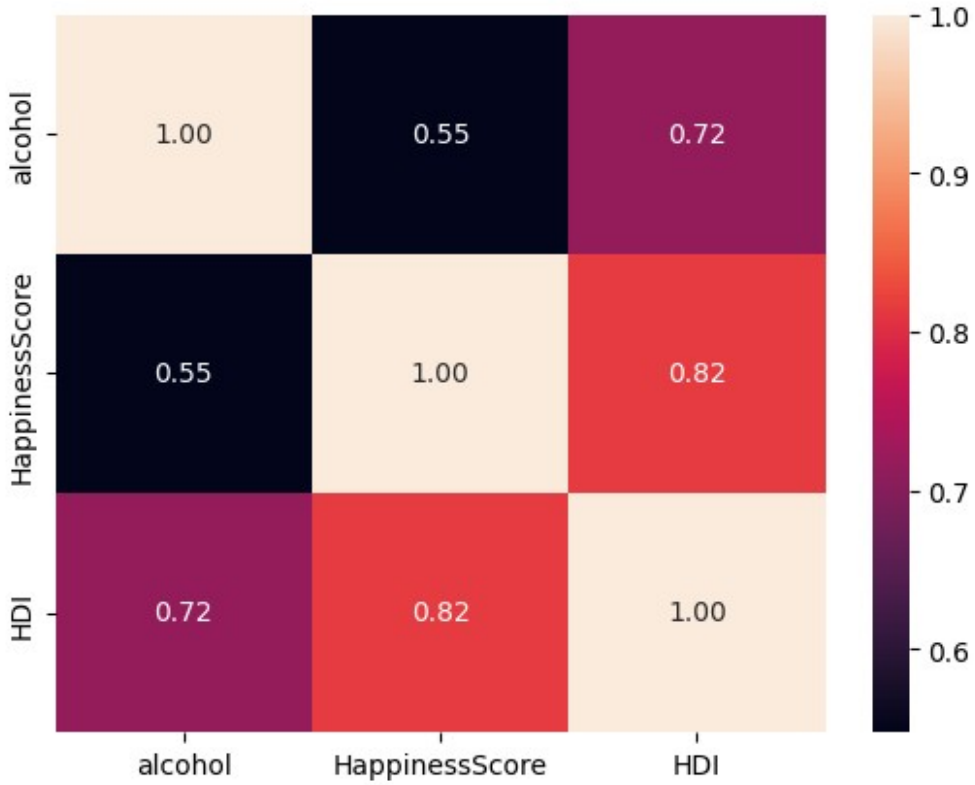
```
fig =plt.figure(figsize=(15,6))
ax = fig.add_subplot()
ax.set_title("all alcohole consumption top 10")
j = 0
for i in descending_a_df.index:
    j = j + 1
    if j <= 10:
        ax.bar(a_df.iloc[i]["country"], a_df.iloc[i]["alcohol"])
    elif j > 10:
        break
```



10 bottom countries that consume alcohol.

```
tmp = a_df.drop(columns=["country", "Beer_PerCapita",
"Spirit_PerCapita", "Wine_PerCapita"])
sns.heatmap(tmp.corr(), annot=True, fmt="0.2f")
```

```
<Axes: >
```



heatmap for HDI, Happiness score, alcohol(sum of "Beer_PerCapita", "Spirit_PerCapita", "Wine_PerCapita"):

1. total alcohol consumption and HDI are correlated by coeff of 0.72 meaning where ever the HDI is higher it's alcohol consumption is higher too.
2. happiness and total alcohol consomption are half correlated.

```
alcohol_df = pd.DataFrame()
alcohol_df["Beer_PerCapita"] = df["Beer_PerCapita"]
alcohol_df["Spirit_PerCapita"] = df["Spirit_PerCapita"]
alcohol_df["Wine_PerCapita"] = df["Wine_PerCapita"]
```

یک دیتافریم جدید میسازیم که تنها سه ستون مربوط به مصرف الکل در آن باشد تا بر اساس بیشترین نوع الکلی که در هر کشور مصرف شده را پیدا idmax() آن بتوانیم با استفاده از تابع کنیم.

```
for i in alcohol_df.index:
    print(i, df.iloc[i]["Country"], alcohol_df.iloc[i].max(),
alcohol_df.iloc[i].idxmax())
```

```
0 Denmark 278 Wine_PerCapita
1 Switzerland 280 Wine_PerCapita
2 Iceland 233 Beer_PerCapita
3 Norway 169 Beer_PerCapita
4 Finland 263 Beer_PerCapita
5 Canada 240 Beer_PerCapita
6 Netherlands 251 Beer_PerCapita
7 New Zealand 203 Beer_PerCapita
8 Australia 261 Beer_PerCapita
9 Sweden 186 Wine_PerCapita
10 Israel 69 Spirit_PerCapita
11 Austria 279 Beer_PerCapita
12 United States 249 Beer_PerCapita
13 Costa Rica 149 Beer_PerCapita
14 Germany 346 Beer_PerCapita
15 Brazil 245 Beer_PerCapita
16 Belgium 295 Beer_PerCapita
17 Ireland 313 Beer_PerCapita
18 Luxembourg 271 Wine_PerCapita
19 Mexico 238 Beer_PerCapita
20 Singapore 60 Beer_PerCapita
21 United Kingdom 219 Beer_PerCapita
22 Chile 172 Wine_PerCapita
23 Panama 285 Beer_PerCapita
24 Argentina 221 Wine_PerCapita
25 Czech Republic 361 Beer_PerCapita
26 United Arab Emirates 135 Spirit_PerCapita
27 Uruguay 220 Wine_PerCapita
28 Malta 149 Beer_PerCapita
29 Colombia 159 Beer_PerCapita
```

30 France 370 Wine_PerCapita
31 Thailand 258 Spirit_PerCapita
32 Qatar 42 Spirit_PerCapita
33 Spain 284 Beer_PerCapita
34 Guatemala 69 Spirit_PerCapita
35 Suriname 178 Spirit_PerCapita
36 Bahrain 63 Spirit_PerCapita
37 Trinidad and Tobago 197 Beer_PerCapita
38 Venezuela 333 Beer_PerCapita
39 Slovakia 293 Spirit_PerCapita
40 El Salvador 69 Spirit_PerCapita
41 Nicaragua 118 Spirit_PerCapita
42 Uzbekistan 101 Spirit_PerCapita
43 Italy 237 Wine_PerCapita
44 Ecuador 162 Beer_PerCapita
45 Belize 263 Beer_PerCapita
46 Japan 202 Spirit_PerCapita
47 Kazakhstan 246 Spirit_PerCapita
48 Moldova 226 Spirit_PerCapita
49 Russian Federation 326 Spirit_PerCapita
50 Poland 343 Beer_PerCapita
51 South Korea 140 Beer_PerCapita
52 Bolivia 167 Beer_PerCapita
53 Lithuania 343 Beer_PerCapita
54 Belarus 373 Spirit_PerCapita
55 Slovenia 276 Wine_PerCapita
56 Peru 163 Beer_PerCapita
57 Turkmenistan 71 Spirit_PerCapita
58 Mauritius 98 Beer_PerCapita
59 Latvia 281 Beer_PerCapita
60 Cyprus 192 Beer_PerCapita
61 Paraguay 213 Beer_PerCapita
62 Romania 297 Beer_PerCapita
63 Estonia 224 Beer_PerCapita
64 Jamaica 97 Spirit_PerCapita
65 Croatia 254 Wine_PerCapita
66 Turkey 51 Beer_PerCapita
67 Jordan 21 Spirit_PerCapita
68 Azerbaijan 46 Spirit_PerCapita
69 Philippines 186 Spirit_PerCapita
70 China 192 Spirit_PerCapita
71 Kyrgyzstan 97 Spirit_PerCapita
72 Serbia 283 Beer_PerCapita
73 Bosnia and Herzegovina 173 Spirit_PerCapita
74 Montenegro 128 Wine_PerCapita
75 Dominican Republic 193 Beer_PerCapita
76 Morocco 12 Beer_PerCapita
77 Hungary 234 Beer_PerCapita
78 Lebanon 55 Spirit_PerCapita

```
79 Portugal 339 Wine_PerCapita
80 Macedonia 106 Beer_PerCapita
81 Vietnam 111 Beer_PerCapita
82 Tunisia 51 Beer_PerCapita
83 Greece 218 Wine_PerCapita
84 Mongolia 189 Spirit_PerCapita
85 Nigeria 42 Beer_PerCapita
86 Honduras 98 Spirit_PerCapita
87 Zambia 32 Beer_PerCapita
88 Albania 132 Spirit_PerCapita
89 Sierra Leone 25 Beer_PerCapita
90 Namibia 376 Beer_PerCapita
91 Cameroon 147 Beer_PerCapita
92 South Africa 225 Beer_PerCapita
93 Egypt 6 Beer_PerCapita
94 Armenia 179 Spirit_PerCapita
95 Kenya 58 Beer_PerCapita
96 Ukraine 237 Spirit_PerCapita
97 Ghana 31 Beer_PerCapita
98 Dem. Rep. Congo 32 Beer_PerCapita
99 Georgia 149 Wine_PerCapita
100 Rep. Congo 76 Beer_PerCapita
101 Senegal 9 Beer_PerCapita
102 Bulgaria 252 Spirit_PerCapita
103 Zimbabwe 64 Beer_PerCapita
104 Malawi 11 Spirit_PerCapita
105 Gabon 347 Beer_PerCapita
106 Mali 5 Beer_PerCapita
107 Haiti 326 Spirit_PerCapita
108 Botswana 173 Beer_PerCapita
109 Comoros 3 Spirit_PerCapita
110 Cote d'Ivoire 37 Beer_PerCapita
111 Cambodia 65 Spirit_PerCapita
112 Angola 217 Beer_PerCapita
113 Niger 3 Beer_PerCapita
114 Chad 15 Beer_PerCapita
115 Burkina Faso 25 Beer_PerCapita
116 Madagascar 26 Beer_PerCapita
117 Tanzania 36 Beer_PerCapita
118 Liberia 152 Spirit_PerCapita
119 Benin 34 Beer_PerCapita
120 Togo 36 Beer_PerCapita
121 Syria 35 Spirit_PerCapita
```

بیشترین مقدار الکلی که از یک نوع در هر کشور مصرف شده و نوع آن

```
a = 0
b = 0
c = 0
```

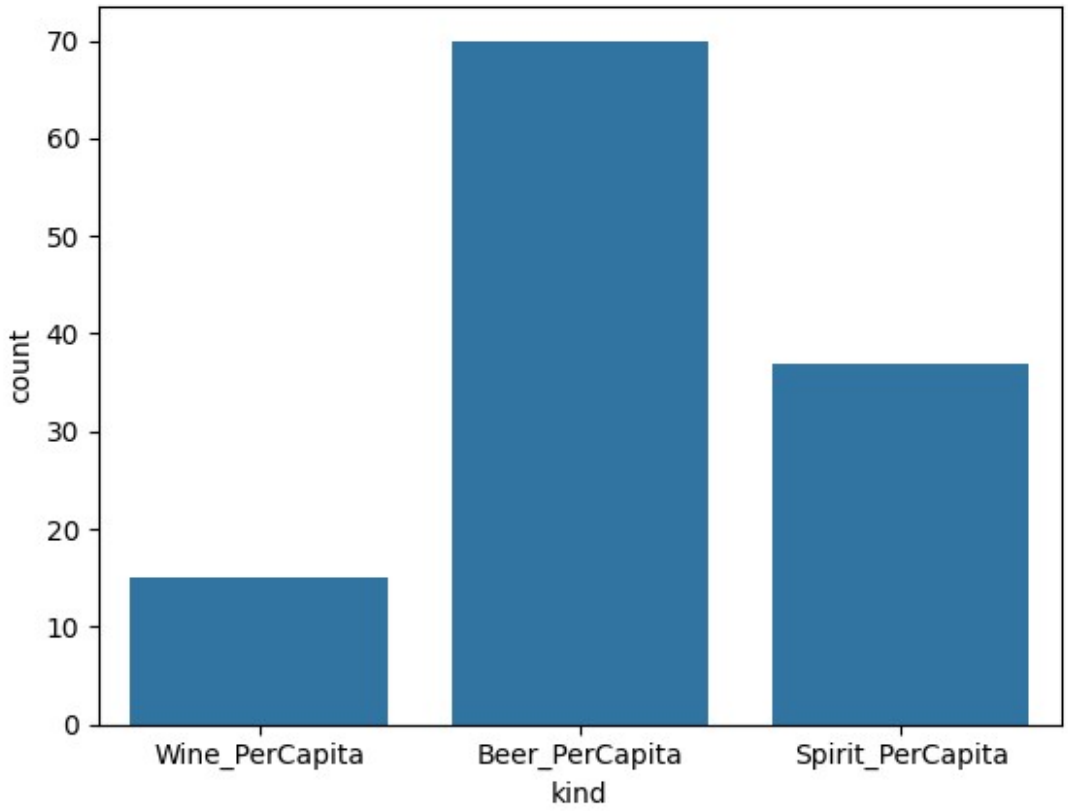
```
for i in alcohol_df.index:
    if alcohol_df.iloc[i].idxmax() == "Beer_PerCapita":
        a += 1
    elif alcohol_df.iloc[i].idxmax() == "Spirit_PerCapita":
        b += 1
    elif alcohol_df.iloc[i].idxmax() == "Wine_PerCapita":
        c += 1
print(a,b,c)
```

```
70 37 15
```

هر نوع الکل چند بار بیشترین مقدار مصرف را در هر کشور داشته و نمودار آن:

```
emp_list = []
for i in alcohol_df.index:
    if alcohol_df.iloc[i].idxmax() == "Beer_PerCapita":
        emp_list.append(alcohol_df.iloc[i].idxmax())
    elif alcohol_df.iloc[i].idxmax() == "Spirit_PerCapita":
        emp_list.append(alcohol_df.iloc[i].idxmax())
    elif alcohol_df.iloc[i].idxmax() == "Wine_PerCapita":
        emp_list.append(alcohol_df.iloc[i].idxmax())

count_df = pd.DataFrame()
count_df["kind"] = emp_list
sns.countplot(x=dataframe["kind"], data=count_df)
```

```
<Axes: xlabel='kind', ylabel='count'>
```




```
j = 0
for i in df.sort_values(by=["HappinessScore"]).index:
    j = j + 1
    if j <= 10:
        print(a_df.iloc[i]["alcohol"], a_df.iloc[i]["country"],
df.sort_values(by=["HappinessScore"], ascending=False).iloc[i]
["HappinessScore"])
    else:
        break
```

```
56 Syria 3.069
57 Togo 3.303
51 Benin 3.484
173 Liberia 3.622
43 Tanzania 3.666
45 Madagascar 3.695
39 Burkina Faso 3.739
17 Chad 3.763
6 Niger 3.856
319 Angola 3.866
```

total alcohol, country, HappinessScore

shows the 10 bottom countries sorted by happiness score and their total alcohol consumption

```
j = 0
for i in df.sort_values(by=["HappinessScore"], ascending=False).index:
    j = j + 1
    if j <= 10:
        print(a_df.iloc[i]["alcohol"], a_df.iloc[i]["country"],
df.sort_values(by=["HappinessScore"], ascending=False).iloc[i]
["HappinessScore"])
    else:
        break
```

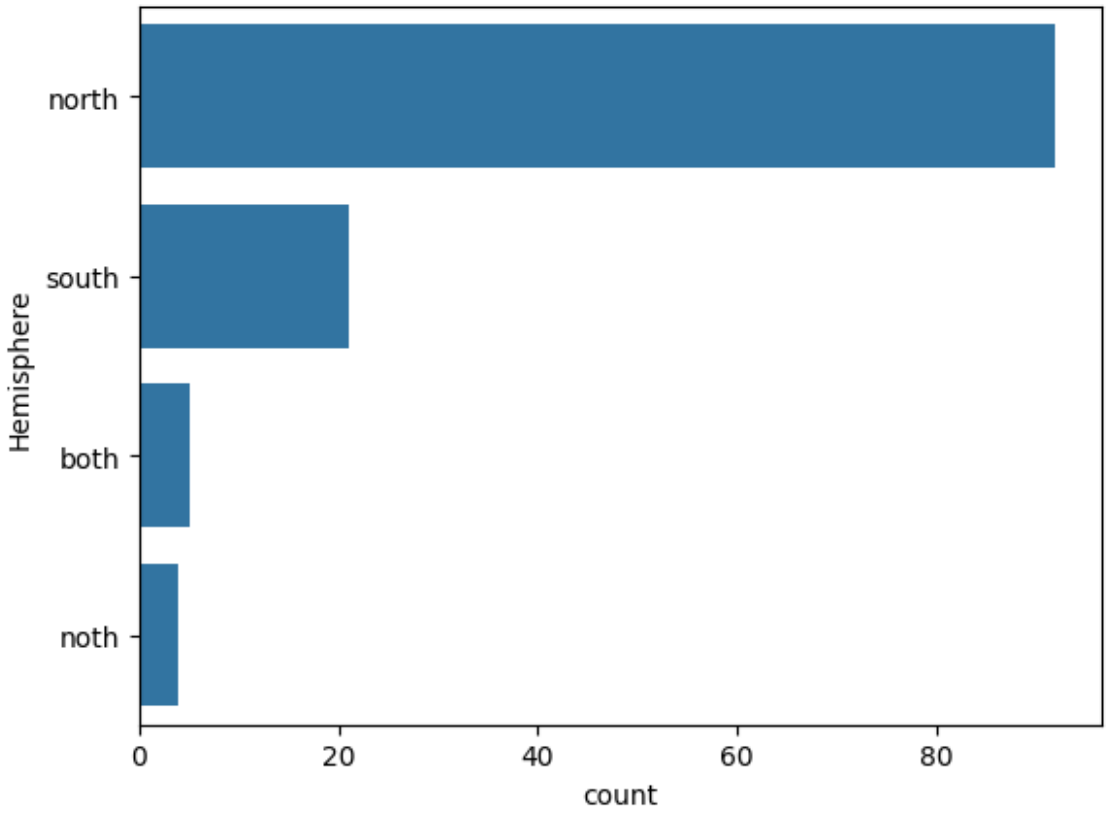
```
583 Denmark 7.526
565 Switzerland 7.509
372 Iceland 7.501
369 Norway 7.498
493 Finland 7.413
462 Canada 7.404
529 Netherlands 7.339
457 New Zealand 7.334
545 Australia 7.313
398 Sweden 7.291
```

total alcohol, country, HappinessScore

shows the 10 top countries sorted by happiness score and their total alcohol consumption

```
sns.countplot(df["Hemisphere"])
```

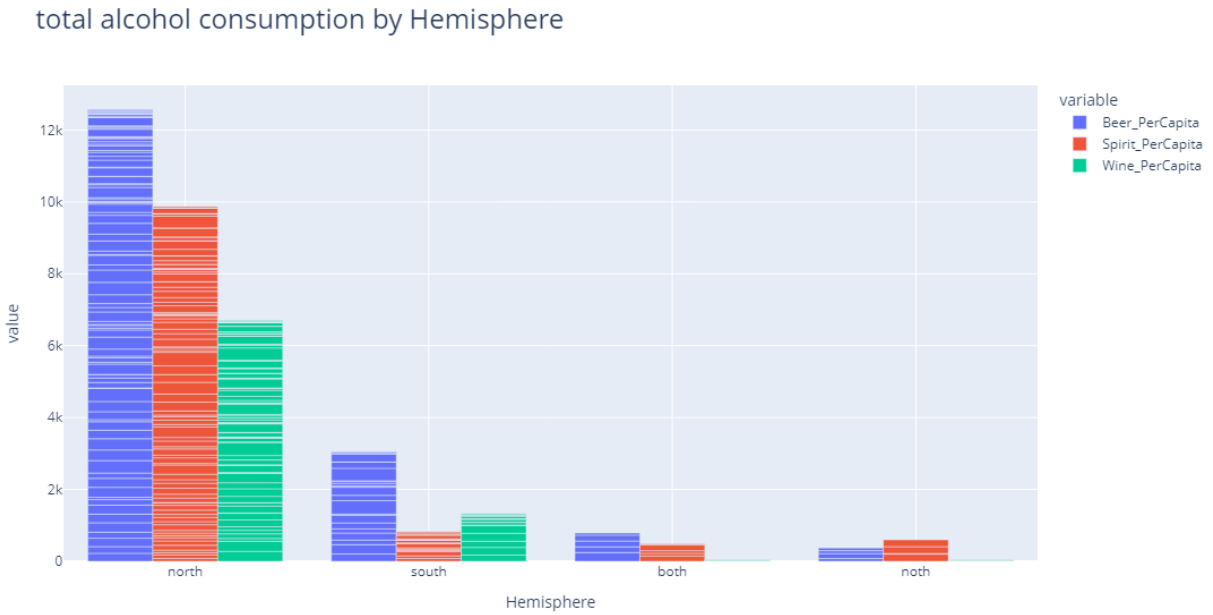
```
<Axes: xlabel='count', ylabel='Hemisphere'>
```



تعداد کشورهای هر نیمکره که در این دیتاست آمده اند

کشورهای نیمکره ی شمالی بیشتر در این دیتاست وجود دارند

```
y=["Beer_PerCapita", "Spirit_PerCapita", "Wine_PerCapita"]
(px.bar(df, x="Hemisphere",y=y, title="total alcohol consumption by
Hemisphere",
        barmode="group", width=800,
height=600).update_layout(title_font_size=24).update_xaxes(showgrid=Tr
ue)).show()
```



A plot that shows how much is each Hemisphere's alcohol consumption by type of alcohol.

```
Hemisphere_df = pd.DataFrame()
Beer_PerCapita_list= []
Spirit_PerCapita_list= []
Wine_PerCapita_list= []

for i in df.groupby("Hemisphere"):
    new_i = i[1]
    Beer_PerCapita_list.append(new_i["Beer_PerCapita"].sum())
    Spirit_PerCapita_list.append(new_i["Spirit_PerCapita"].sum())
    Wine_PerCapita_list.append(new_i["Wine_PerCapita"].sum())

Hemisphere_df["Hemisphere"] = df.groupby("Hemisphere").dtypes.index
Hemisphere_df["Beer_PerCapita"] = Beer_PerCapita_list
Hemisphere_df["Spirit_PerCapita"] = Spirit_PerCapita_list
Hemisphere_df["Wine_PerCapita"] = Wine_PerCapita_list

Hemisphere_df.head()
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_5572\3680581275.py:12:
FutureWarning:

DataFrameGroupBy.dtypes is deprecated and will be removed in a future
version. Check the dtypes on the base object instead
```

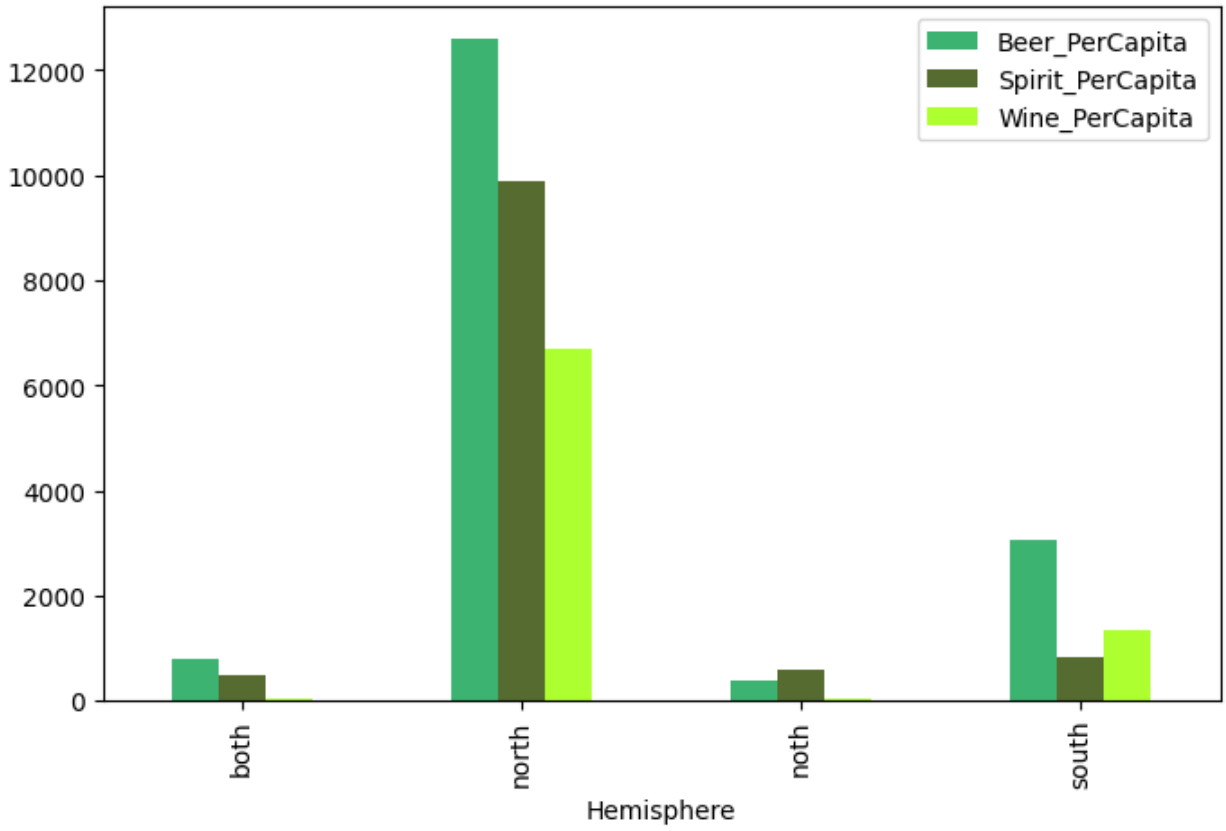
	Hemisphere	Beer_PerCapita	Spirit_PerCapita	Wine_PerCapita
0	both	787	477	45
1	north	12581	9870	6707

```
2       noth              373                599                41
3      south             3042                839              1332
```

Making a dataframe that shows the total and type of alcohol consumption of each Hemisphere. it is for making a

```
my_color =['mediumseagreen', 'darkolivegreen','greenyellow']
Hemisphere_df.plot(x="Hemisphere", y=["Beer_PerCapita",
"Spirit_PerCapita", "Wine_PerCapita"],
            kind="bar",figsize=(8,5), color=my_color)
```

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




```
plt.show()
```