Assignment

In this project, you will visualize and make calculations from medical examination data using matplotlib, seaborn, and pandas. The dataset values were collected during medical examinations.

Data description

The rows in the dataset represent patients and the columns represent information like body measurements, results from various blood tests, and lifestyle choices. You will use the dataset to explore the relationship between cardiac disease, body measurements, blood markers, and lifestyle choices.

File name: medical_examination.csv

Feature	Variable Type	Variable	Value Type
Age	Objective Feature	age	int (days)
Height	Objective Feature	height	int (cm)
Weight	Objective Feature	weight	float (kg)
Gender	Objective Feature	gender	categorical code
Systolic blood pressure	Examination Feature	ap_hi	int
Diastolic blood pressure	Examination Feature	ap_lo	int
Cholesterol	Examination Feature	cholesterol	1: normal, 2: above normal, 3: well above normal
Glucose	Examination Feature	gluc	1: normal, 2: above normal, 3: well above normal
Smoking	Subjective Feature	smoke	binary
Alcohol intake	Subjective Feature	alco	binary
Physical activity	Subjective Feature	active	binary
Presence or absence of cardiovascular disease	Target Variable	cardio	binary

Tasks

Create a chart similar to `examples/Figure_1.png`, where we show the counts of good and bad outcomes for the `cholesterol`, `gluc`, `alco`, `active`, and `smoke` variables for patients with cardio=1 and cardio=0 in different panels.

Use the data to complete the following tasks in `medical_data_visualizer.py`:

- Add an `overweight` column to the data. To determine if a person is overweight, first calculate their BMI by dividing their weight in kilograms by the square of their height in meters. If that value is > 25 then the person is overweight. Use the value 0 for NOT overweight and the value 1 for overweight.
- Normalize the data by making 0 always good and 1 always bad. If the value of `cholesterol` or `gluc` is 1, make the value 0. If the value is more than 1, make the value 1.
- Convert the data into long format and create a chart that shows the value counts of the categorical features using seaborn's `catplot()`. The dataset should be split by 'Cardio' so there is one chart

for each `cardio` value. The chart should look like `examples/Figure_1.png`.

- Clean the data. Filter out the following patient segments that represent incorrect data:
  - diastolic pressure is higher than systolic (Keep the correct data with `df['ap_lo'] <= df['ap_hi'])`)
  - height is less than the 2.5th percentile (Keep the correct data with `(df['height'] >= df['height'].quantile(0.025))`)
  - height is more than the 97.5th percentile
  - weight is less than the 2.5th percentile
  - weight is more than the 97.5th percentile
- Create a correlation matrix using the dataset. Plot the correlation matrix using seaborn's `heatmap()`. Mask the upper triangle. The chart should look like `examples/Figure_2.png`.

In [1]:
```
import pandas as pd
import os
import numpy as np
import matplotlib as plt
from matplotlib import pyplot
import seaborn as sns
```

In [2]:
```
os.chdir('C:\\Users\\ANAND\\FreeCodeCamp\csv data')
```

In [3]:
```
df = pd.read_csv('medical_examination.csv')
```

In [4]:
```
df_estimate = df['weight']/np.square(df['height']/100)
```

In [5]:
```
df['overweight'] = np.where(df_estimate > 25, 1,0)
```

In [6]:
```
df['cholesterol'] = np.where(df['cholesterol']>1,1,0)
df['gluc'] = np.where(df['gluc']>1,1,0)
```

In [7]:
```
#Now all these variables are categorical: cholesterol   gluc    smok
e       alco    active  cardio  overweight
df.head()
```

Out[7]:

	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active
0	0	18393	2	168	62.0	110	80	0	0	0	0	1
1	1	20228	1	156	85.0	140	90	1	0	0	0	1
2	2	18857	1	165	64.0	130	70	1	0	0	0	0
3	3	17623	2	169	82.0	150	100	0	0	0	0	1
4	4	17474	1	156	56.0	100	60	0	0	0	0	0

In [12]:
```
cat_plot = pd.melt(df, id_vars = ['cardio'], value_vars = ['active
','alco','cholesterol', 'gluc', 'overweight', 'smoke'])
```

In [13]:
```
cat_plot
```

Out[13]:

	cardio	variable	value
0	0	active	1
1	1	active	1
2	1	active	0
3	1	active	1
4	0	active	0
...	...	...	...
419995	0	smoke	1
419996	1	smoke	0
419997	1	smoke	0
419998	1	smoke	0
419999	0	smoke	0

420000 rows × 3 columns

In [14]:
```
cat_plot = pd.DataFrame(cat_plot.groupby(['cardio','variable', 'value'])['value'].count()).rename(columns={'value': 'total'}).reset_index()
cat_plot
```

Out[14]:

	cardio	variable	value	total
0	0	active	0	6378
1	0	active	1	28643
2	0	alco	0	33080
3	0	alco	1	1941
4	0	cholesterol	0	29330
5	0	cholesterol	1	5691
6	0	gluc	0	30894
7	0	gluc	1	4127
8	0	overweight	0	15915
9	0	overweight	1	19106
10	0	smoke	0	31781
11	0	smoke	1	3240
12	1	active	0	7361
13	1	active	1	27618
14	1	alco	0	33156
15	1	alco	1	1823
16	1	cholesterol	0	23055
17	1	cholesterol	1	11924
18	1	gluc	0	28585
19	1	gluc	1	6394
20	1	overweight	0	10539
21	1	overweight	1	24440
22	1	smoke	0	32050
23	1	smoke	1	2929

In [9]:
```
cat_plot['total'] = 1
```

In [10]:
```
cat_plot.head()
```

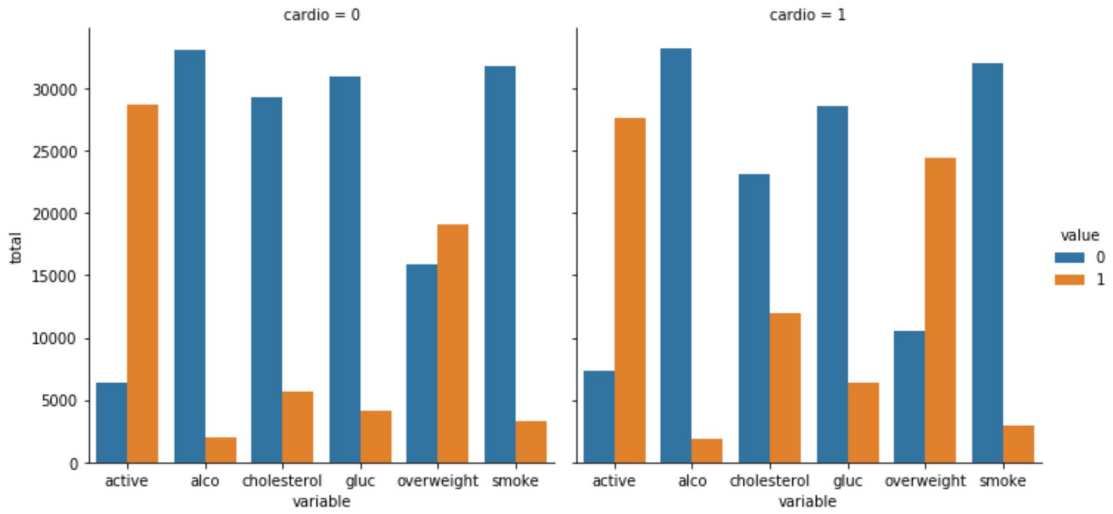
Out[10]:

	cardio	variable	value	total
0	0	active	1	1
1	1	active	1	1
2	1	active	0	1
3	1	active	1	1
4	0	active	0	1

In [11]:
```
cat_plot = cat_plot.groupby(['cardio','variable','value'],as_index=False).count()
```

In [12]:
```
sns.catplot(data=cat_plot,x='variable',y='total' , hue = 'value', col = 'cardio', kind = 'bar' )
```

Out[12]:
```
<seaborn.axisgrid.FacetGrid at 0x202f99674c8>
```



In [13]:
```
#or without creating any group just use kind = count in sns.cat_plot
#sns.catplot(data=cat_plot, kind='count', x='variable', hue='value', col='cardio')
```

In [14]:
```
df = df[~(df['ap_lo'] >= df['ap_hi'])]
```

In [15]:
```
#percentiles should all be in the interval [0, 1].
df = df[~((df['height'] < df['height'].quantile(0.025))&(df['height'] > df['height'].quantile(0.975)))]
```

In [16]:
```
df = df[~((df['weight'] < df['weight'].quantile(0.025))&(df['weight'] > df['weight'].quantile(0.975)))]
```

In [17]:
```
df
```

Out[17]:

	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco
0	0	18393	2	168	62.0	110	80	0	0	0	0
1	1	20228	1	156	85.0	140	90	1	0	0	0
2	2	18857	1	165	64.0	130	70	1	0	0	0
3	3	17623	2	169	82.0	150	100	0	0	0	0
4	4	17474	1	156	56.0	100	60	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
69995	99993	19240	2	168	76.0	120	80	0	0	1	0
69996	99995	22601	1	158	126.0	140	90	1	1	0	0
69997	99996	19066	2	183	105.0	180	90	1	0	0	1
69998	99998	22431	1	163	72.0	135	80	0	1	0	0
69999	99999	20540	1	170	72.0	120	80	1	0	0	0

68764 rows × 14 columns

Create a correlation matrix using the dataset. Plot the correlation matrix using seaborn's heatmap(). Mask the upper triangle. The chart should look like examples/Figure_2.png

In [18]:
```
df_corr = round(df.corr(),1)
```

In [19]:
```
df_corr
```

Out[19]:

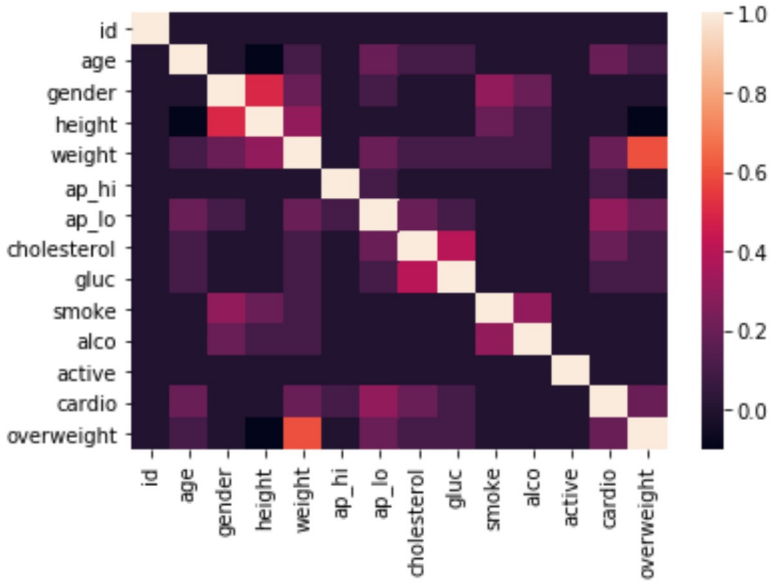
	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alc
id	1.0	0.0	0.0	-0.0	-0.0	0.0	-0.0	0.0	0.0	-0.0	0.
age	0.0	1.0	-0.0	-0.1	0.1	0.0	0.2	0.1	0.1	-0.0	-0.
gender	0.0	-0.0	1.0	0.5	0.2	0.0	0.1	-0.0	-0.0	0.3	0.
height	-0.0	-0.1	0.5	1.0	0.3	0.0	0.0	-0.0	-0.0	0.2	0.
weight	-0.0	0.1	0.2	0.3	1.0	0.0	0.2	0.1	0.1	0.1	0.
ap_hi	0.0	0.0	0.0	0.0	0.0	1.0	0.1	0.0	0.0	-0.0	0.
ap_lo	-0.0	0.2	0.1	0.0	0.2	0.1	1.0	0.2	0.1	0.0	0.
cholesterol	0.0	0.1	-0.0	-0.0	0.1	0.0	0.2	1.0	0.4	0.0	0.
gluc	0.0	0.1	-0.0	-0.0	0.1	0.0	0.1	0.4	1.0	0.0	0.
smoke	-0.0	-0.0	0.3	0.2	0.1	-0.0	0.0	0.0	0.0	1.0	0.
alco	0.0	-0.0	0.2	0.1	0.1	0.0	0.0	0.0	0.0	0.3	1.
active	0.0	-0.0	0.0	-0.0	-0.0	-0.0	-0.0	0.0	-0.0	0.0	0.
cardio	0.0	0.2	0.0	-0.0	0.2	0.1	0.3	0.2	0.1	-0.0	-0.
overweight	-0.0	0.1	-0.0	-0.1	0.6	0.0	0.2	0.1	0.1	-0.0	0.

In [20]:
```
df_corr.min()
```

Out[20]:
```
id             -0.0
age            -0.1
gender         -0.0
height         -0.1
weight         -0.0
ap_hi           0.0
ap_lo          -0.0
cholesterol     0.0
gluc           -0.0
smoke          -0.0
alco            0.0
active         -0.0
cardio         -0.0
overweight     -0.1
dtype: float64
```

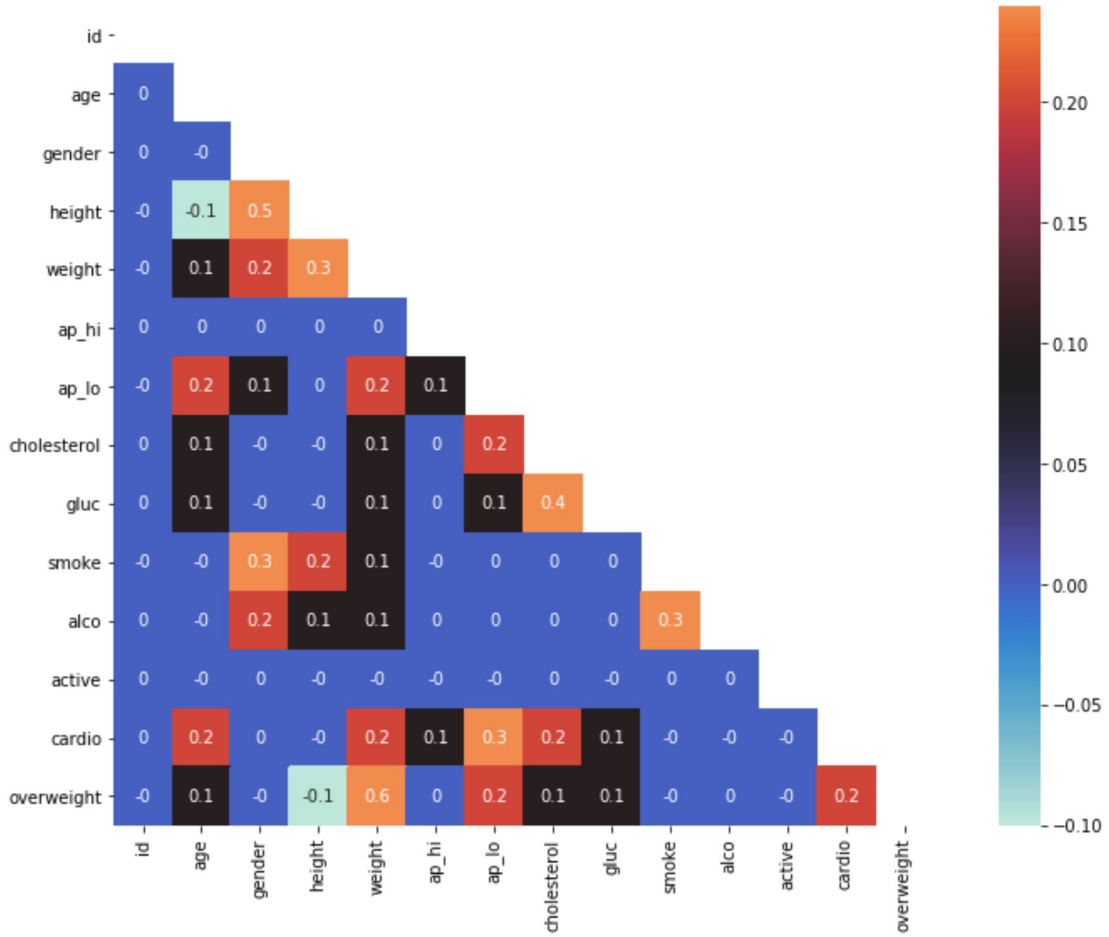
In [21]:
```
sns.heatmap(df_corr)
```

Out[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x202f86ebf48>
```



In [22]:
```
mask = np.zeros_like(df_corr)
mask[np.triu_indices_from(mask)] = True #Return the indices for the upper-triangle of arr.
```

In [23]:
```
f, ax = pyplot.subplots(figsize=(14, 9))
ax = sns.heatmap(df_corr, mask = mask, annot=True, vmax =0.24, vmin
= -0.10, center = 0.09, square= True)
```



In [24]:
```
df.head()
```

Out[24]:

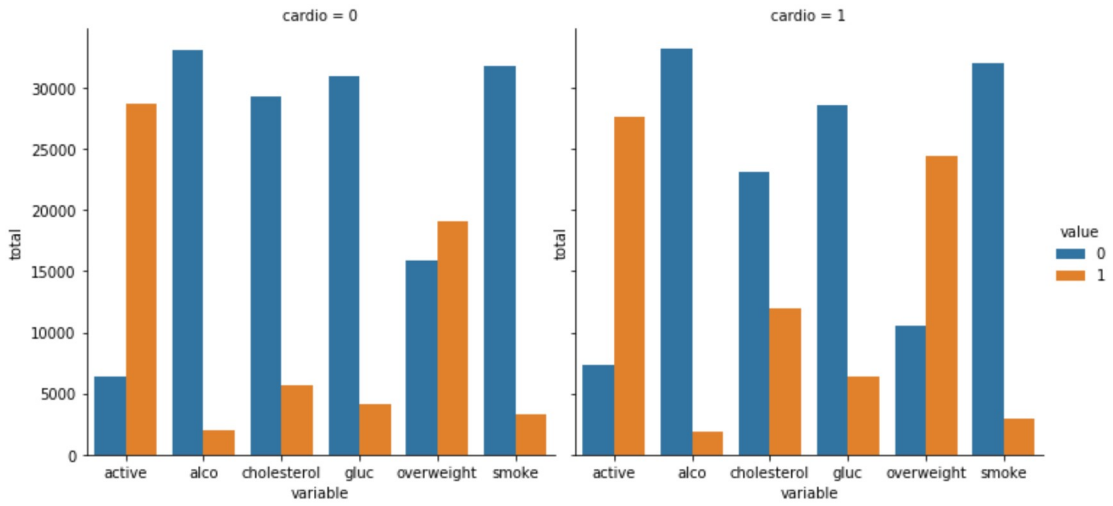
	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active
0	0	18393	2	168	62.0	110	80	0	0	0	0	1
1	1	20228	1	156	85.0	140	90	1	0	0	0	1
2	2	18857	1	165	64.0	130	70	1	0	0	0	0
3	3	17623	2	169	82.0	150	100	0	0	0	0	1
4	4	17474	1	156	56.0	100	60	0	0	0	0	0

In [25]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Import data
df = pd.read_csv('medical_examination.csv')

# Add 'overweight' column
df_estimate = df['weight']/np.square(df['height']/100)
df['overweight'] = np.where(df_estimate > 25, 1,0)

# Normalize data by making 0 always good and 1 always bad. If the va
lue of 'cholestorol' or 'gluc' is 1, make the value 0. If the value
is more than 1, make the value 1.
df['cholesterol'] = np.where(df['cholesterol']>1,1,0)
df['gluc'] = np.where(df['gluc']>1,1,0)

# Draw Categorical Plot
def draw_cat_plot():
    # Create DataFrame for cat plot using `pd.melt` using just the v
alues from 'cholesterol', 'gluc', 'smoke', 'alco', 'active', and 'ov
erweight'.
    df_cat = pd.melt(df, id_vars = ['cardio'], value_vars = ['active
','alco','cholesterol', 'gluc', 'overweight', 'smoke'])


    # Group and reformat the data to split it by 'cardio'. Show the
counts of each feature. You will have to rename one of the collumns
for the catplot to work correctly.
    #df_cat = Not Needed, can plot directly

    # Draw the catplot with 'sns.catplot()'
    ax = sns.catplot(data=df_cat, kind='count', x='variable', hue='v
alue', col='cardio')
    ax.set(ylabel = 'total')
    ax.set(xlabel = 'variable')
    fig=ax
    # Do not modify the next two lines
    fig.savefig('catplot.png')
    return fig


# Draw Heat Map
def draw_heat_map():
    # Clean the data
    df = pd.read_csv('medical_examination.csv')

    # Add 'overweight' column
    df_estimate = df['weight']/np.square(df['height']/100)
    df['overweight'] = np.where(df_estimate > 25, 1,0)

    # Normalize data by making 0 always good and 1 always bad. If th
e value of 'cholestorol' or 'gluc' is 1, make the value 0. If the va
lue is more than 1, make the value 1.
    df['cholesterol'] = np.where(df['cholesterol']>1,1,0)
    df['gluc'] = np.where(df['gluc']>1,1,0)
    df = df[~(df['ap_lo'] >= df['ap_hi'])]
    df = df[~((df['height'] < df['height'].quantile(0.025))|(df['hei
```

```
ght'] > df['height'].quantile(0.975))]
    df = df[~((df['weight'] < df['weight'].quantile(0.025))|(df['wei
ght'] > df['weight'].quantile(0.975)))]
    df_heat = df

    # Calculate the correlation matrix
    corr = round(df_heat.corr(),1)

    # Generate a mask for the upper triangle
    mask = np.zeros_like(corr)
    mask[np.triu_indices_from(mask)] = True


    # Set up the matplotlib figure
    fig, ax = plt.subplots(figsize=(14, 9))
    ax = sns.heatmap(corr, mask = mask, annot=True, vmax =0.24, vmin
= -0.10, center = 0.09, square= True)

    # Draw the heatmap with 'sns.heatmap()'
    plt.close()
    # Do not modify the next two lines
    fig.savefig('heatmap.png')
    return fig
```

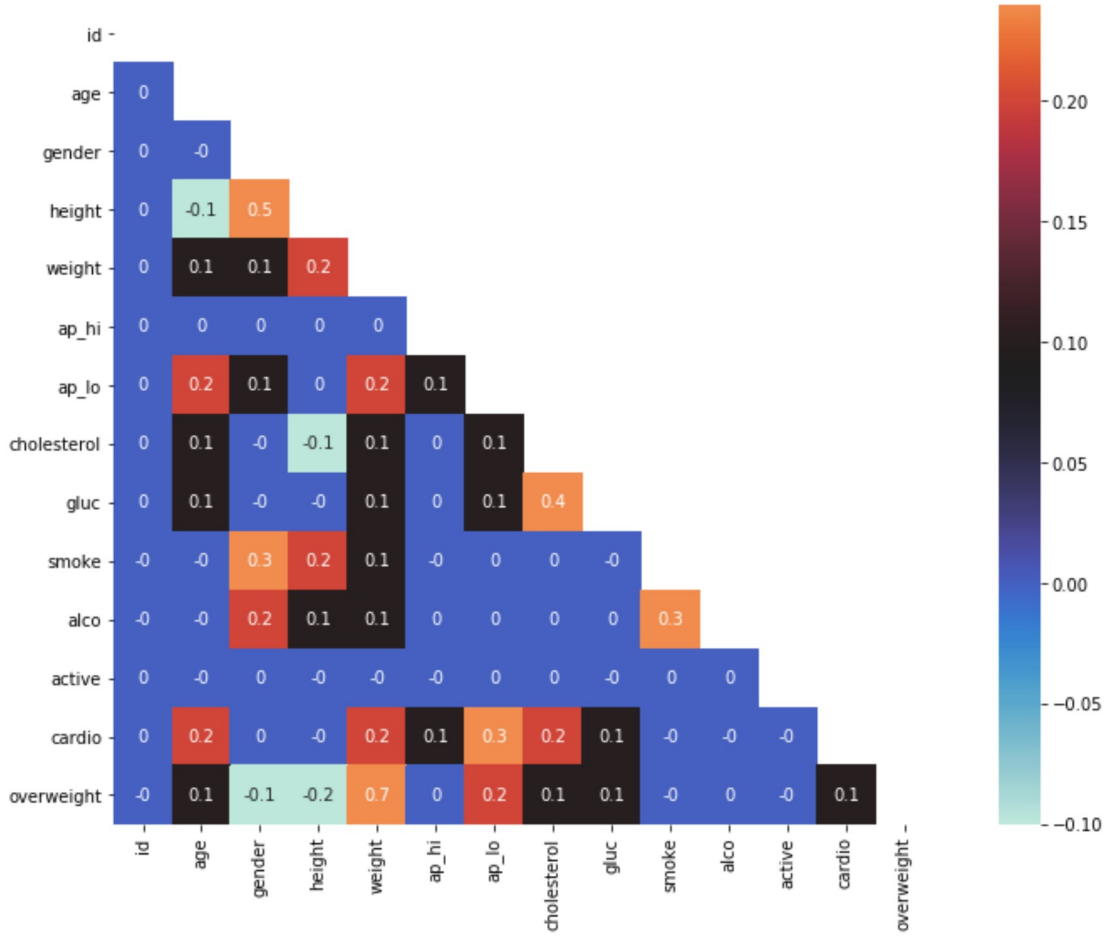
In [26]:
```
draw_cat_plot()
```

Out[26]: <seaborn.axisgrid.FacetGrid at 0x202fb407fc8>



In [27]:
```
draw_heat_map()
```

Out[27]:



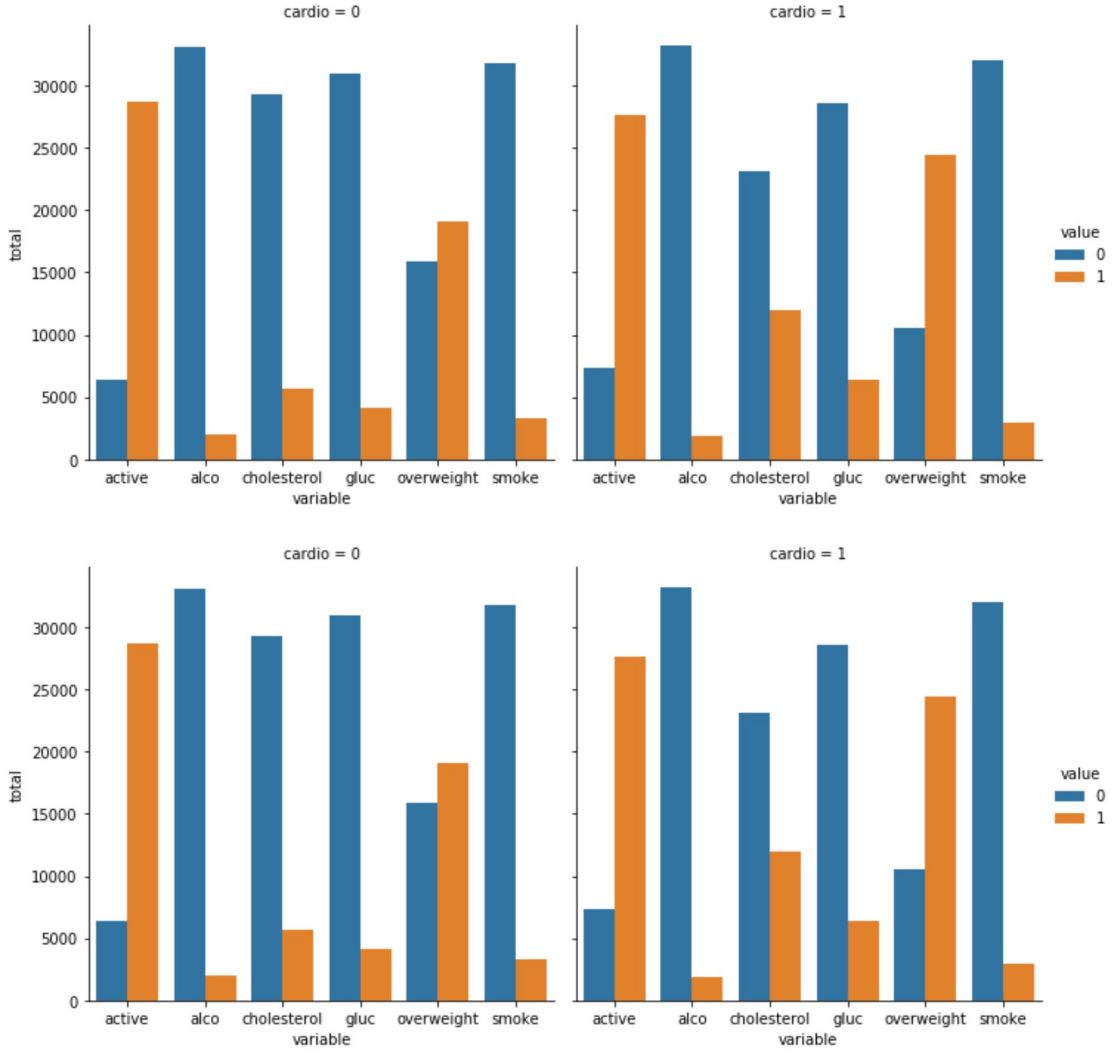
medical_dict = { 1: 0, 2: 1, 3: 1} df['cholesterol'] = df['cholesterol'].map(medical_dict) df['gluc'] = df['gluc'].map(medical_dict)

In [20]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Import data
df = pd.read_csv('medical_examination.csv')

# Add 'overweight' column
df_estimate = df['weight']/np.square(df['height']/100)
df['overweight'] = np.where(df_estimate > 25, 1,0)

# Normalize data by making 0 always good and 1 always bad. If the va
lue of 'cholestorol' or 'gluc' is 1, make the value 0. If the value
is more than 1, make the value 1.
df['cholesterol'] = np.where(df['cholesterol']>1,1,0)
df['gluc'] = np.where(df['gluc']>1,1,0)

# Draw Categorical Plot
def draw_catplot():
    # Create DataFrame for cat plot using `pd.melt` using just the v
alues from 'cholesterol', 'gluc', 'smoke', 'alco', 'active', and 'ov
erweight'.
    cat_plot = pd.melt(
      df, id_vars = ['cardio'], value_vars = ['active','alco','chole
sterol', 'gluc','overweight', 'smoke']
      )

    cat_plot = pd.DataFrame(
      cat_plot.groupby(
        ['variable', 'value', 'cardio'])['value'].count()).rename(co
lumns={'value': 'total'}).reset_index()

    g = sns.catplot(x='variable', y='total', data=cat_plot, hue='val
ue', col='cardio', kind='bar')
    fig = g.fig #using.fig to indicate that the plot elements is pre
sent in g# get_children/artist error solved
    fig.savefig('catplot.png')
    return fig

# Draw Heat Map
def draw_heatmap():
    # Clean the data
  df = pd.read_csv('medical_examination.csv')

    # Add 'overweight' column
  df_estimate = df['weight']/np.square(df['height']/100)
  df['overweight'] = np.where(df_estimate > 25, 1,0)
  df['cholesterol'] = np.where(df['cholesterol']>1,1,0)
  df['gluc'] = np.where(df['gluc']>1,1,0)
# Clean the data
  df_heat = df[(df['ap_lo'] <= df['ap_hi']) & (df['height'] >= df['h
eight'].quantile(0.025)) & (df['height'] <= df['height'].quantile(0.
975)) & (df['weight'] >= df['weight'].quantile(0.025)) & (df['weight
'] <= df['weight'].quantile(0.975))]

  # Calculate the correlation matrix
  corr = df_heat.corr()
```

```
  # Generate a mask for the upper triangle
  mask = np.zeros_like(corr)
  mask[np.triu_indices_from(mask)] = True

   # Set up the matplotlib figure
  fig, ax = plt.subplots(figsize=(10, 12))
  #ax = sns.heatmap(corr, mask = mask, annot=True, vmax =0.24, vmin
= -0.10, center = 0.09, square= True)
  ax = sns.heatmap(corr, annot=True, fmt='.1f',mask=mask, vmin=.16,
vmax=.32, center=0, square=True,linewidths=.5,cbar_kws={'shrink':.4
5, 'format':'%.2f'})

  # Draw the heatmap with 'sns.heatmap()'
  plt.close()
  # Do not modify the next two lines
  fig.savefig('heatmap.png')
  return fig
```

In [21]:
```
draw_catplot()
```

Out[21]:



In [22]:
```
draw_heatmap()
```

Out[22]: