Final Project

December 12, 2022

[1]:
```python
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import datetime as dt
import warnings
```

Data upload

[78]:
```python
df = pd.read_csv('/Users/polina/Desktop/Life Expectancy Data.csv')
df
```

[78]:
```
          Country  Year      Status  Life expectancy   Adult Mortality  \
0     Afghanistan  2015  Developing              65.0            263.0
1     Afghanistan  2014  Developing              59.9            271.0
2     Afghanistan  2013  Developing              59.9            268.0
3     Afghanistan  2012  Developing              59.5            272.0
4     Afghanistan  2011  Developing              59.2            275.0
…             …     …         …                …               …
2933     Zimbabwe  2004  Developing              44.3            723.0
2934     Zimbabwe  2003  Developing              44.5            715.0
2935     Zimbabwe  2002  Developing              44.8             73.0
2936     Zimbabwe  2001  Developing              45.3            686.0
2937     Zimbabwe  2000  Developing              46.0            665.0

      infant deaths  Alcohol  percentage expenditure  Hepatitis B  Measles   \
0                62     0.01               71.279624         65.0     1154
1                64     0.01               73.523582         62.0      492
2                66     0.01               73.219243         64.0      430
3                69     0.01               78.184215         67.0     2787
4                71     0.01                7.097109         68.0     3013
…               …       …                      …           …       …
2933             27     4.36                0.000000         68.0       31
2934             26     4.06                0.000000          7.0      998
2935             25     4.43                0.000000         73.0      304
2936             25     1.72                0.000000         76.0      529
```

```
2937             24      1.68                0.000000         79.0      1483

      …  Polio  Total expenditure  Diphtheria   HIV/AIDS         GDP  \
0     …    6.0               8.16        65.0        0.1  584.259210
1     …   58.0               8.18        62.0        0.1  612.696514
2     …   62.0               8.13        64.0        0.1  631.744976
3     …   67.0               8.52        67.0        0.1  669.959000
4     …   68.0               7.87        68.0        0.1   63.537231
…   …     …                 …          …          …          …
2933  …   67.0               7.13        65.0       33.6  454.366654
2934  …    7.0               6.52        68.0       36.7  453.351155
2935  …   73.0               6.53        71.0       39.8   57.348340
2936  …   76.0               6.16        75.0       42.1  548.587312
2937  …   78.0               7.10        78.0       43.5  547.358878

      Population   thinness  1-19 years   thinness 5-9 years  \
0     33736494.0                    17.2                  17.3
1       327582.0                    17.5                  17.5
2     31731688.0                    17.7                  17.7
3      3696958.0                    17.9                  18.0
4      2978599.0                    18.2                  18.2
…            …                       …                  …
2933  12777511.0                     9.4                   9.4
2934  12633897.0                     9.8                   9.9
2935    125525.0                     1.2                   1.3
2936  12366165.0                     1.6                   1.7
2937  12222251.0                    11.0                  11.2

      Income composition of resources  Schooling
0                               0.479       10.1
1                               0.476       10.0
2                               0.470        9.9
3                               0.463        9.8
4                               0.454        9.5
…                               …         …
2933                            0.407        9.2
2934                            0.418        9.5
2935                            0.427       10.0
2936                            0.427        9.8
2937                            0.434        9.8

[2938 rows x 22 columns]
```

Data Info

[79]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 2938 entries, 0 to 2937
Data columns (total 22 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Country                          2938 non-null   object
 1   Year                             2938 non-null   int64
 2   Status                           2938 non-null   object
 3   Life expectancy                  2928 non-null   float64
 4   Adult Mortality                  2928 non-null   float64
 5   infant deaths                    2938 non-null   int64
 6   Alcohol                          2744 non-null   float64
 7   percentage expenditure           2938 non-null   float64
 8   Hepatitis B                      2385 non-null   float64
 9   Measles                          2938 non-null   int64
 10   BMI                             2904 non-null   float64
 11  under-five deaths                2938 non-null   int64
 12  Polio                            2919 non-null   float64
 13  Total expenditure                2712 non-null   float64
 14  Diphtheria                       2919 non-null   float64
 15   HIV/AIDS                        2938 non-null   float64
 16  GDP                              2490 non-null   float64
 17  Population                       2286 non-null   float64
 18   thinness  1-19 years            2904 non-null   float64
 19   thinness 5-9 years              2904 non-null   float64
 20  Income composition of resources  2771 non-null   float64
 21  Schooling                        2775 non-null   float64
dtypes: float64(16), int64(4), object(2)
memory usage: 505.1+ KB
```

[80]:
```
df.shape
```

[80]:
```
(2938, 22)
```

[81]:
```
df.describe()
```

[81]:
```
              Year  Life expectancy  Adult Mortality  infant deaths  \
count  2938.000000      2928.000000      2928.000000    2938.000000
mean   2007.518720        69.224932       164.796448      30.303948
std       4.613841         9.523867       124.292079     117.926501
min    2000.000000        36.300000         1.000000       0.000000
25%    2004.000000        63.100000        74.000000       0.000000
50%    2008.000000        72.100000       144.000000       3.000000
75%    2012.000000        75.700000       228.000000      22.000000
max    2015.000000        89.000000       723.000000    1800.000000

           Alcohol  percentage expenditure  Hepatitis B       Measles   \
count  2744.000000             2938.000000  2385.000000    2938.000000
mean      4.602861              738.251295    80.940461    2419.592240
```

```
std        4.052413          1987.914858    25.070016   11467.272489
min        0.010000             0.000000     1.000000       0.000000
25%        0.877500             4.685343    77.000000       0.000000
50%        3.755000            64.912906    92.000000      17.000000
75%        7.702500           441.534144    97.000000     360.250000
max       17.870000         19479.911610    99.000000  212183.000000

               BMI  under-five deaths        Polio  Total expenditure  \
count  2904.000000        2938.000000  2919.000000         2712.00000
mean     38.321247          42.035739    82.550188            5.93819
std      20.044034         160.445548    23.428046            2.49832
min       1.000000           0.000000     3.000000            0.37000
25%      19.300000           0.000000    78.000000            4.26000
50%      43.500000           4.000000    93.000000            5.75500
75%      56.200000          28.000000    97.000000            7.49250
max      87.300000        2500.000000    99.000000           17.60000

        Diphtheria     HIV/AIDS            GDP    Population  \
count  2919.000000  2938.000000    2490.000000  2.286000e+03
mean     82.324084     1.742103    7483.158469  1.275338e+07
std      23.716912     5.077785   14270.169342  6.101210e+07
min       2.000000     0.100000       1.681350  3.400000e+01
25%      78.000000     0.100000     463.935626  1.957932e+05
50%      93.000000     0.100000    1766.947595  1.386542e+06
75%      97.000000     0.800000    5910.806335  7.420359e+06
max      99.000000    50.600000  119172.741800  1.293859e+09

         thinness  1-19 years   thinness 5-9 years  \
count             2904.000000          2904.000000
mean                 4.839704             4.870317
std                  4.420195             4.508882
min                  0.100000             0.100000
25%                  1.600000             1.500000
50%                  3.300000             3.300000
75%                  7.200000             7.200000
max                 27.700000            28.600000

       Income composition of resources    Schooling
count                      2771.000000  2775.000000
mean                          0.627551    11.992793
std                           0.210904     3.358920
min                           0.000000     0.000000
25%                           0.493000    10.100000
50%                           0.677000    12.300000
75%                           0.779000    14.300000
max                           0.948000    20.700000
```

Data Cleaning

[82]:
```python
# Looking for null value in the data
df.isnull().sum()
```

[82]:
```
Country                              0
Year                                 0
Status                               0
Life expectancy                     10
Adult Mortality                     10
infant deaths                        0
Alcohol                            194
percentage expenditure               0
Hepatitis B                        553
Measles                              0
 BMI                                34
under-five deaths                    0
Polio                               19
Total expenditure                  226
Diphtheria                          19
 HIV/AIDS                            0
GDP                                448
Population                         652
 thinness  1-19 years               34
 thinness 5-9 years                 34
Income composition of resources    167
Schooling                          163
dtype: int64
```

[83]:
```python
# Replacing the Null Values with mean values of the data
from sklearn.impute import SimpleImputer
#reference: https://scikit-learn.org/stable/modules/generated/sklearn.impute.
 ↪SimpleImputer.html
imputer=SimpleImputer(missing_values=np.nan,strategy='mean',fill_value=None)
df['Life expectancy ']=imputer.fit_transform(df[['Life expectancy ']])
df['Adult Mortality']=imputer.fit_transform(df[['Adult Mortality']])
df['Alcohol']=imputer.fit_transform(df[['Alcohol']])
df['Hepatitis B']=imputer.fit_transform(df[['Hepatitis B']])
df[' BMI ']=imputer.fit_transform(df[[' BMI ']])
df['Polio']=imputer.fit_transform(df[['Polio']])
df['Total expenditure']=imputer.fit_transform(df[['Total expenditure']])
df['Diphtheria ']=imputer.fit_transform(df[['Diphtheria ']])
df['GDP']=imputer.fit_transform(df[['GDP']])
df['Population']=imputer.fit_transform(df[['Population']])
df[' thinness  1-19 years']=imputer.fit_transform(df[[' thinness  1-19 years']])
df[' thinness 5-9 years']=imputer.fit_transform(df[[' thinness 5-9 years']])
df['Income composition of resources']=imputer.fit_transform(df[['Income␣
 ↪composition of resources']])
df['Schooling']=imputer.fit_transform(df[['Schooling']])
```

[84]:
```
# Looking for null value in the data after fitting
df.isnull().sum()
```

[84]:
```
Country                            0
Year                               0
Status                             0
Life expectancy                    0
Adult Mortality                    0
infant deaths                      0
Alcohol                            0
percentage expenditure             0
Hepatitis B                        0
Measles                            0
 BMI                               0
under-five deaths                  0
Polio                              0
Total expenditure                  0
Diphtheria                         0
 HIV/AIDS                          0
GDP                                0
Population                         0
 thinness  1-19 years              0
 thinness 5-9 years                0
Income composition of resources    0
Schooling                          0
dtype: int64
```

[85]:
```
# Changing/Renaming the columns for easy access.
df = df.rename(columns={'Country': 'country', 'Year': 'year', 'Status':␣
 ↪'status', 'Life expectancy ': 'life_expectancy', 'Adult Mortality':␣
 ↪'adult_mortality',
       'infant deaths':'infant_death', 'Alcohol':'alcohol', 'percentage␣
 ↪expenditure': 'percentage_expenditure', 'Hepatitis B':'Hepatitis_b',
       'Measles ':'measles', ' BMI ':'bmi', 'under-five deaths ':
 ↪'under_five_deaths', 'Polio':'polio', 'Total expenditure':␣
 ↪'total_expenditure','Diphtheria ':'diphtheria', ' HIV/AIDS':'hiv_Aids',␣
 ↪'GDP':'gdp', 'Population':'population',
       ' thinness  1-19 years':'thinness_1_to_19', ' thinness 5-9 years':
 ↪'thinness_5_to_9',
       'Income composition of resources':'income_composition_of_resources',␣
 ↪'Schooling': 'schooling'})
```

[86]:
```
# Looking for columns after rename
df.columns
```

[86]:
```
Index(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_death', 'alcohol', 'percentage_expenditure', 'Hepatitis_b',
```

```
 'measles', 'bmi', 'under_five_deaths', 'polio', 'total_expenditure',
 'diphtheria', 'hiv_Aids', 'gdp', 'population', 'thinness_1_to_19',
 'thinness_5_to_9', 'income_composition_of_resources', 'schooling'],
dtype='object')
```

[87]:
```
#remove empty space
orig_cols = list(df.columns)
new_cols = []
for col in orig_cols:
    new_cols.append(col.strip().replace('  ', ' ').replace(' ', '_').lower())
df.columns = new_cols
```

[ ]:
```
# save the clean data into a CSV file
df.to_csv('clean_df.csv', index=False)
```

[ ]:
```
clean_df = pd.read_csv("clean_df.csv")
clean_df
```

Data Describe

[12]:
```
df.describe(include='object')
```

[12]:

	country	status
count	2938	2938
unique	193	2
top	Afghanistan	Developing
freq	16	2426

[13]:
```
#Top 10 Countries
print("Top 10 Countries with Most Life Expectancy")
print("="*50)
print(df.groupby("country").life_expectancy.mean().sort_values(ascending␣
 ↪=False).head(10))
print("="*50)
print("Top 10 Countries with Least Life Expectancy")
print("="*50)
print(df.groupby("country").life_expectancy.mean().sort_values(ascending =True).
 ↪head(10))
```

```
Top 10 Countries with Most Life Expectancy
==================================================
country
Japan          82.53750
Sweden         82.51875
Iceland        82.44375
Switzerland    82.33125
France         82.21875
Italy          82.18750
```

```
Spain           82.06875
Australia       81.81250
Norway          81.79375
Canada          81.68750
Name: life_expectancy, dtype: float64
==================================================
Top 10 Countries with Least Life Expectancy
==================================================
country
Sierra Leone                46.11250
Central African Republic    48.51250
Lesotho                     48.78125
Angola                      49.01875
Malawi                      49.89375
Chad                        50.38750
Côte d'Ivoire               50.38750
Zimbabwe                    50.48750
Swaziland                   51.32500
Nigeria                     51.35625
Name: life_expectancy, dtype: float64
```

[14]:
```python
# Countries with Highest Life Expectancy
country_vs_life = df.groupby('country', as_index=False)['life_expectancy'].
 ↪mean()
country_vs_life.sort_values(by = 'life_expectancy', ascending=False).head(10)
```

[14]:

	country	life_expectancy
84	Japan	82.53750
165	Sweden	82.51875
75	Iceland	82.44375
166	Switzerland	82.33125
60	France	82.21875
82	Italy	82.18750
160	Spain	82.06875
7	Australia	81.81250
125	Norway	81.79375
30	Canada	81.68750

[15]:
```python
# Countries with Lowest Life Expectancy
country_vs_life.sort_values(by = 'life_expectancy', ascending = True).head(10)
```

[15]:

	country	life_expectancy
152	Sierra Leone	46.11250
31	Central African Republic	48.51250
94	Lesotho	48.78125
3	Angola	49.01875
100	Malawi	49.89375
32	Chad	50.38750

```
44            Côte d'Ivoire        50.38750
192                Zimbabwe        50.48750
164               Swaziland        51.32500
123                 Nigeria        51.35625
```

[16]:
```
df.corr()['life_expectancy'].abs().sort_values(ascending=False)[1:]
```

[16]:
```
schooling                          0.715066
adult_mortality                    0.696359
income_composition_of_resources    0.692483
bmi                                0.559255
hiv_aids                           0.556457
diphtheria                         0.475418
thinness_1_to_19                   0.472162
thinness_5_to_9                    0.466629
polio                              0.461574
gdp                                0.430493
alcohol                            0.391598
percentage_expenditure             0.381791
under_five_deaths                  0.222503
total_expenditure                  0.207981
hepatitis_b                        0.203771
infant_death                       0.196535
year                               0.169623
measles                            0.157574
population                         0.019638
Name: life_expectancy, dtype: float64
```

[17]:
```
df["status"].value_counts()
```

[17]:
```
Developing    2426
Developed      512
Name: status, dtype: int64
```

[19]:
```
df.groupby(['status'])[["life_expectancy"]].mean()
```

[19]:
```
            life_expectancy
status
Developed         79.197852
Developing        67.120177
```

[20]:
```
corr = df.corr()
corr.style.background_gradient(cmap='coolwarm')
```

[20]:
```
<pandas.io.formats.style.Styler at 0x7fb145779ac0>
```

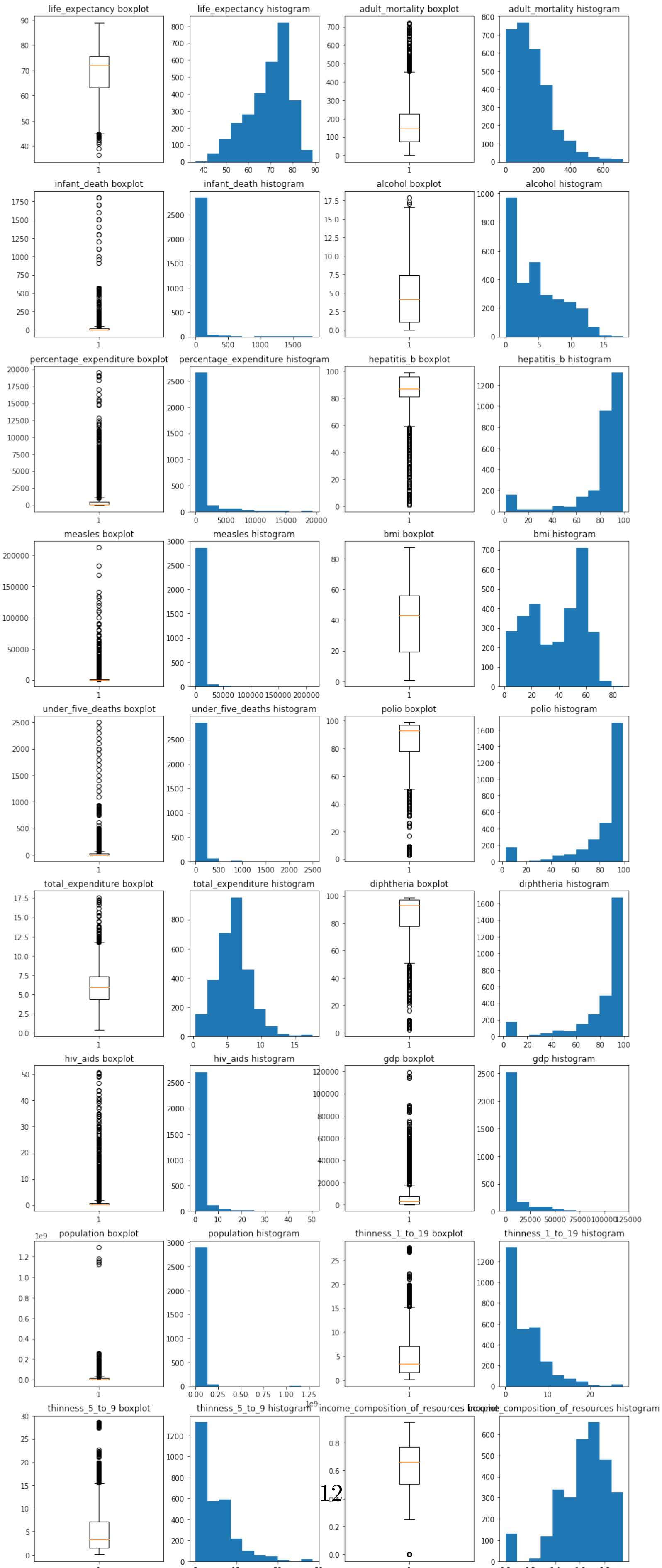
Outliers

[21]:

```python
cont_vars = list(df.columns)[3:]
def outliers_visual(data):
    plt.figure(figsize=(15, 40))
    i = 0
    for col in cont_vars:
        i += 1
        plt.subplot(9, 4, i)
        plt.boxplot(data[col])
        plt.title('{} boxplot'.format(col))
        i += 1
        plt.subplot(9, 4, i)
        plt.hist(data[col])
        plt.title('{} histogram'.format(col))
    plt.show()
outliers_visual(df)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [21], in <cell line: 15>()
     13         plt.title('{} histogram'.format(col))
     14     plt.show()
---> 15 outliers_visual(df)

Input In [21], in outliers_visual(data)
      5 for col in cont_vars:
      6     i += 1
----> 7     plt.subplot(9, 4, i)
      8     plt.boxplot(data[col])
      9     plt.title('{} boxplot'.format(col))

File ~/opt/anaconda3/lib/python3.9/site-packages/matplotlib/pyplot.py:1268, in
 ↪subplot(*args, **kwargs)
   1265 fig = gcf()
   1267 # First, search for an existing subplot with a matching spec.
-> 1268 key = SubplotSpec._from_subplot_args(fig, args)
   1270 for ax in fig.axes:
   1271     # if we found an axes at the position sort out if we can re-use it
   1272     if hasattr(ax, 'get_subplotspec') and ax.get_subplotspec() == key:
   1273         # if the user passed no kwargs, re-use

File ~/opt/anaconda3/lib/python3.9/site-packages/matplotlib/gridspec.py:608, in
 ↪SubplotSpec._from_subplot_args(figure, args)
    606 else:
    607     if not isinstance(num, Integral) or num < 1 or num > rows*cols:
--> 608         raise ValueError(
    609             f"num must be 1 <= num <= {rows*cols}, not {num!r}")
    610     i = j = num
```

```
    611 return gs[i-1:j]

ValueError: num must be 1 <= num <= 36, not 37
```


life_expectancy boxplot
life_expectancy histogram
adult_mortality boxplot
adult_mortality histogram
infant_death boxplot
infant_death histogram
alcohol boxplot
alcohol histogram
percentage_expenditure boxplot
percentage_expenditure histogram
hepatitis_b boxplot
hepatitis_b histogram
measles boxplot
measles histogram
bmi boxplot
bmi histogram
under_five_deaths boxplot
under_five_deaths histogram
polio boxplot
polio histogram
total_expenditure boxplot
total_expenditure histogram
diphtheria boxplot
diphtheria histogram
hiv_aids boxplot
hiv_aids histogram
gdp boxplot
gdp histogram
population boxplot
population histogram
thinness_1_to_19 boxplot
thinness_1_to_19 histogram
thinness_5_to_9 boxplot
thinness_5_to_9 histogram
income_composition_of_resources boxplot
income_composition_of_resources histogram

Visually, it is plain to see that there are a number of outliers for all of these variables - including the target variable, life expectancy.

Uisng Tukey's method below - outliers being considered anything outside of 1.5 times the IQR

[22]:
```
def outlier_count(col, data=df):
    print(15*'-' + col + 15*'-')
    q75, q25 = np.percentile(data[col], [75, 25])
    iqr = q75 - q25
    min_val = q25 - (iqr*1.5)
    max_val = q75 + (iqr*1.5)
    outlier_count = len(np.where((data[col] > max_val) | (data[col] <
 ↪min_val))[0])
    outlier_percent = round(outlier_count/len(data[col])*100, 2)
    print('Number of outliers: {}'.format(outlier_count))
    print('Percent of data that is outlier: {}%'.format(outlier_percent))
```

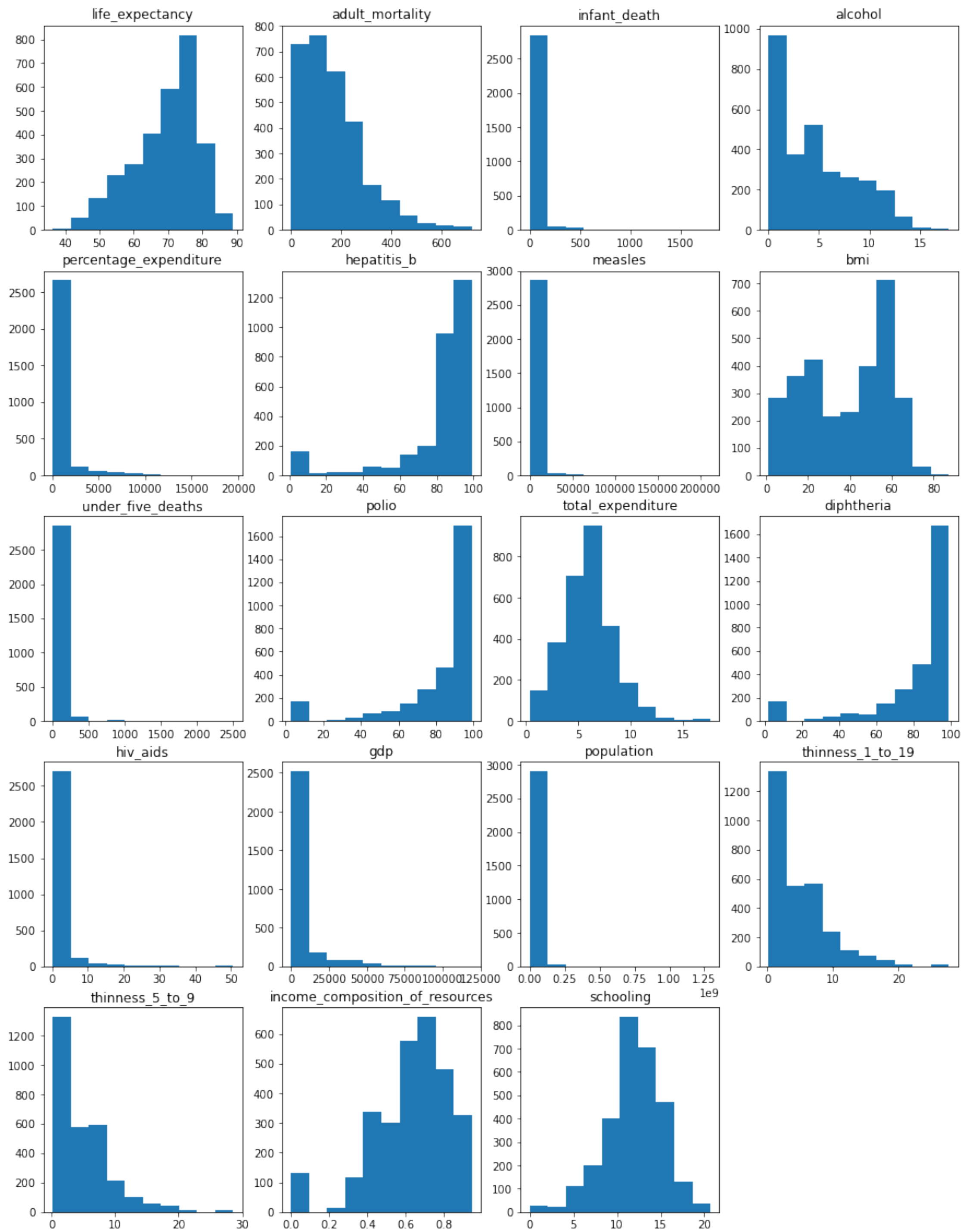
[23]:
```
for col in cont_vars:
    outlier_count(col)
```

```
---------------life_expectancy---------------
Number of outliers: 17
Percent of data that is outlier: 0.58%
---------------adult_mortality---------------
Number of outliers: 86
Percent of data that is outlier: 2.93%
---------------infant_death---------------
Number of outliers: 315
Percent of data that is outlier: 10.72%
---------------alcohol---------------
Number of outliers: 3
Percent of data that is outlier: 0.1%
---------------percentage_expenditure---------------
Number of outliers: 389
Percent of data that is outlier: 13.24%
---------------hepatitis_b---------------
Number of outliers: 316
Percent of data that is outlier: 10.76%
---------------measles---------------
Number of outliers: 542
Percent of data that is outlier: 18.45%
---------------bmi---------------
Number of outliers: 0
Percent of data that is outlier: 0.0%
---------------under_five_deaths---------------
Number of outliers: 394
```

```
Percent of data that is outlier: 13.41%
---------------polio---------------
Number of outliers: 279
Percent of data that is outlier: 9.5%
---------------total_expenditure---------------
Number of outliers: 51
Percent of data that is outlier: 1.74%
---------------diphtheria---------------
Number of outliers: 298
Percent of data that is outlier: 10.14%
---------------hiv_aids---------------
Number of outliers: 542
Percent of data that is outlier: 18.45%
---------------gdp---------------
Number of outliers: 300
Percent of data that is outlier: 10.21%
---------------population---------------
Number of outliers: 194
Percent of data that is outlier: 6.6%
---------------thinness_1_to_19---------------
Number of outliers: 100
Percent of data that is outlier: 3.4%
---------------thinness_5_to_9---------------
Number of outliers: 99
Percent of data that is outlier: 3.37%
---------------income_composition_of_resources---------------
Number of outliers: 130
Percent of data that is outlier: 4.42%
---------------schooling---------------
Number of outliers: 77
Percent of data that is outlier: 2.62%
```

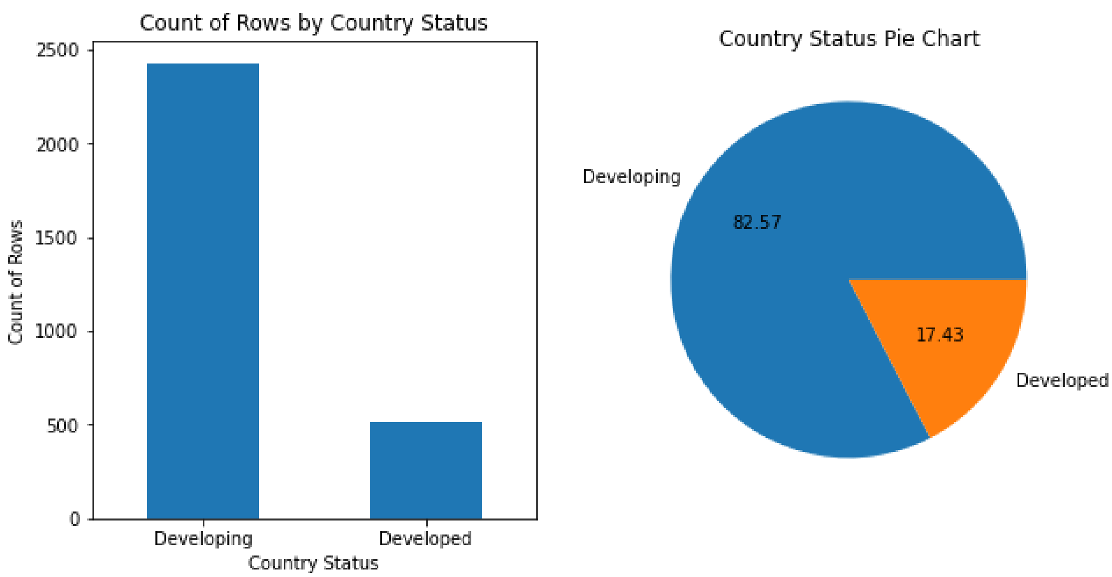
Data Viz

[24]:
```
# Visual Distributions
plt.figure(figsize=(15, 20))
for i, col in enumerate(cont_vars, 1):
    plt.subplot(5, 4, i)
    plt.hist(df[col])
    plt.title(col)
```



[25]:
```python
#. Values of rows for status
plt.figure(figsize=(10, 5))
plt.subplot(121)
df.status.value_counts().plot(kind='bar')
```

```
plt.title('Count of Rows by Country Status')
plt.xlabel('Country Status')
plt.ylabel('Count of Rows')
plt.xticks(rotation=0)

plt.subplot(122)
df.status.value_counts().plot(kind='pie', autopct='%.2f')
plt.ylabel('')
plt.title('Country Status Pie Chart')

plt.show()
```

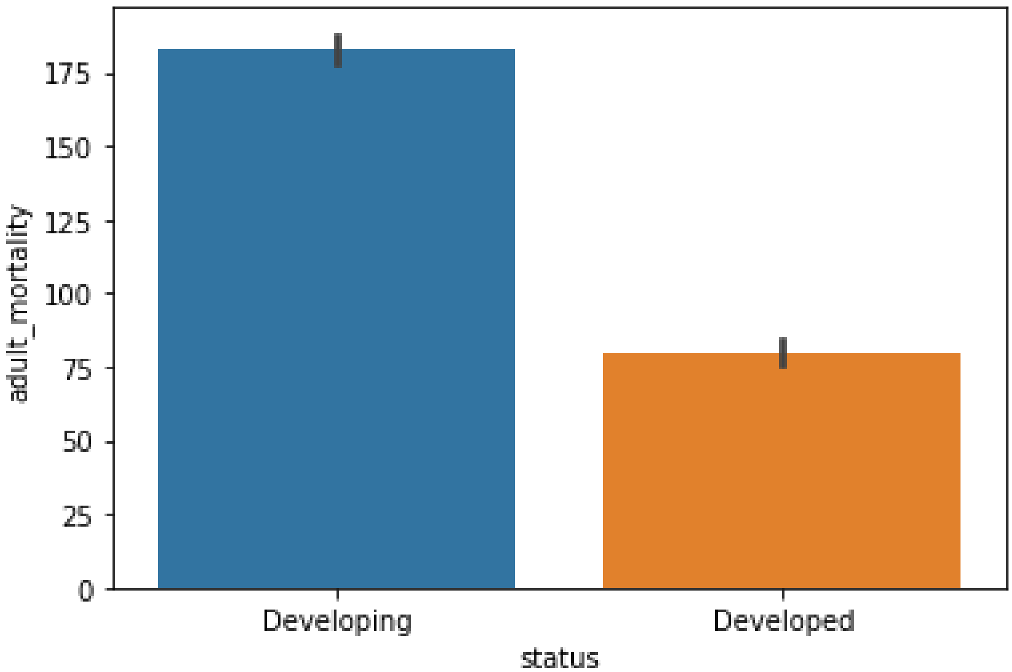


The above displays that the majority of our data comes from countries listed as 'Developing' - 82.57% to be exact. It is likely that any model used will more accurately depict results for 'Developing' countries over 'Developed' countries as the majority of the data lies within countries that are 'Developing' rather than 'Developed'.

[26]:
```
#Adult mortality
sns.barplot(df["status"],df['adult_mortality'])
```

```
/Users/polina/opt/anaconda3/lib/python3.9/site-
packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variables
as keyword args: x, y. From version 0.12, the only valid positional argument
will be `data`, and passing other arguments without an explicit keyword will
result in an error or misinterpretation.
  warnings.warn(
```

[26]: <AxesSubplot:xlabel='status', ylabel='adult_mortality'>



[27]:
```
#Distribution of Life Expectancy according to the age
fig = px.histogram(df, x = 'life_expectancy', template = 'plotly_dark')
fig.show()
```

[28]:
```
#Comparing the life expectancy of Developing and Developed Countries
fig = px.violin(df, x= 'status', y= 'life_expectancy',
                color = 'status',template = 'plotly_dark', box =␣
 ↪True,title='Life Expectancy on the Basis of Country Status')
fig.show()
```

[29]:
```
#Country Wise Life Expectancy over the years
fig = px.line((df.sort_values(by = 'year')), x = 'year', y = 'life_expectancy',
    animation_frame= 'country',template = 'plotly_dark',␣
 ↪animation_group='year',color='country',
    markers=True,title='Country Wise Life Expectancy over the years')
fig.show()
```

[30]:
```
country_df = px.data.gapminder()
country_df.tail()
```

[30]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	\
1699	Zimbabwe	Africa	1987	62.351	9216418	706.157306	ZWE	
1700	Zimbabwe	Africa	1992	60.377	10704340	693.420786	ZWE	
1701	Zimbabwe	Africa	1997	46.809	11404948	792.449960	ZWE	

```
1702  Zimbabwe    Africa  2002   39.989  11926563  672.038623       ZWE
1703  Zimbabwe    Africa  2007   43.487  12311143  469.709298       ZWE

      iso_num
1699      716
1700      716
1701      716
1702      716
1703      716
```

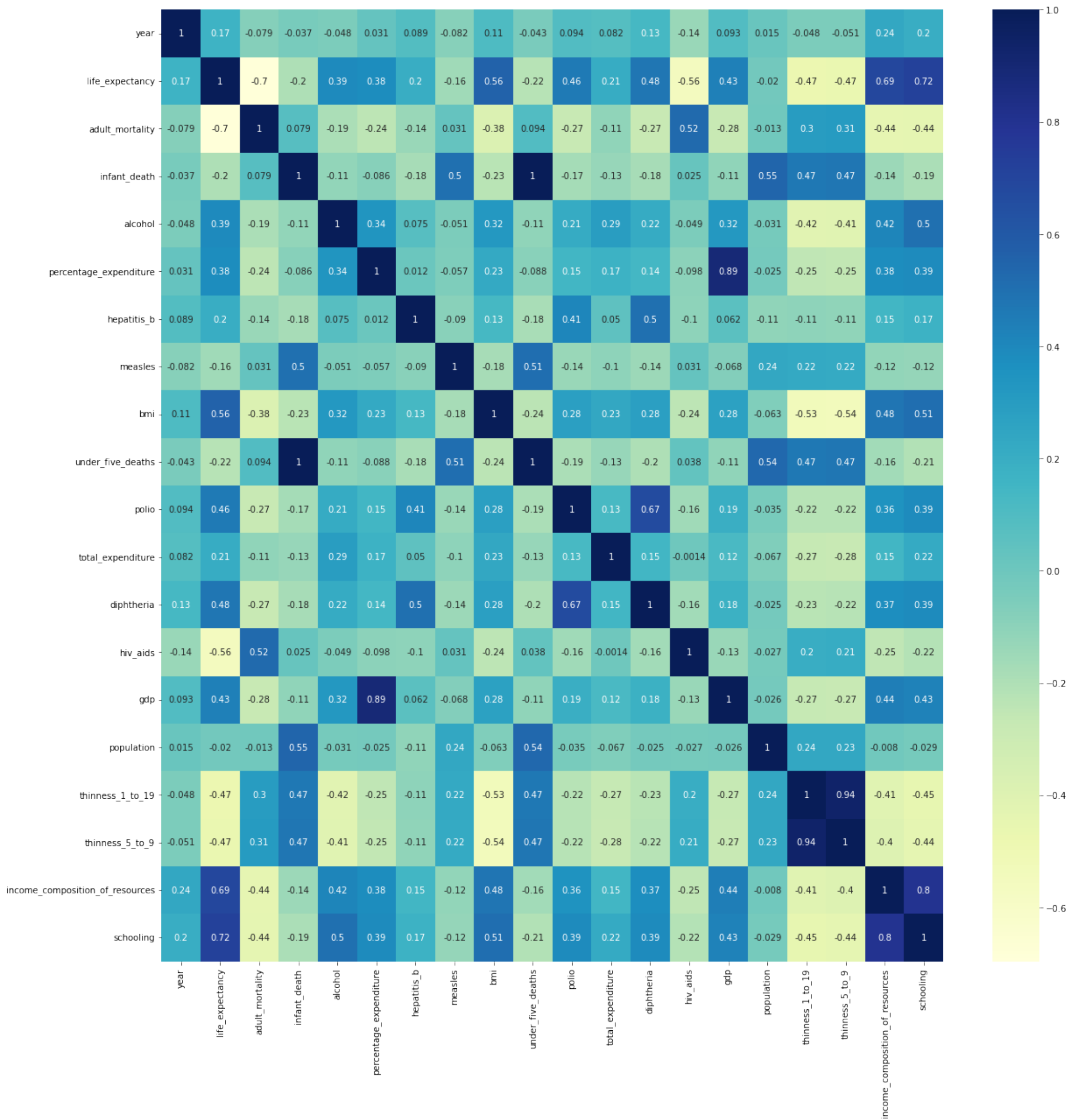
[31]:
```
#Life Expectancy over the World Map
map_fig = px.scatter_geo(country_df,locations = 'iso_alpha', projection =␣
 ↪'orthographic',
                         opacity = 0.8, color = 'country', hover_name =␣
 ↪'country',
                         hover_data = ['lifeExp', 'year'],template =␣
 ↪'plotly_dark',title = 'Life Expectancy over the World Map')
map_fig.show()
```

[32]:
```
# Life Expectancy versus the adult Mortality in different countries every year¶
px.scatter(df, x = 'life_expectancy', y = 'adult_mortality',
           color = 'country', template = 'plotly_dark', size =␣
 ↪'life_expectancy',opacity = 0.6,
           title = '<b>Life Expectancy Vs Adult Mortality in Countries')
```

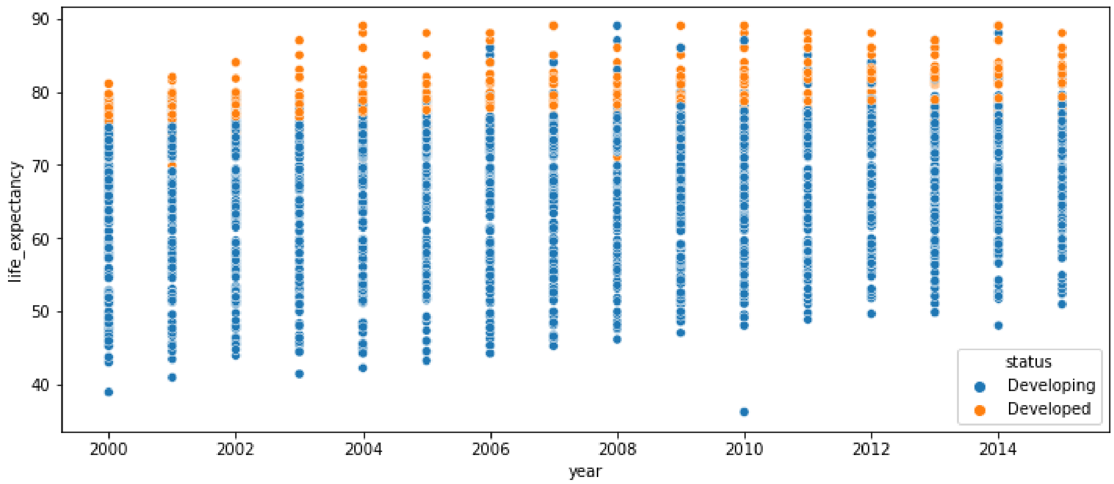
[33]:
```
#Life Expectancy Versus GDP of Counntries all over the World¶
px.scatter(df.sort_values(by='year'), x = 'life_expectancy', y = 'gdp', color =␣
 ↪'country',
          size = 'year',animation_frame = 'year', animation_group =␣
 ↪'country',template = 'plotly_dark',
           title = '<b>Life Expectancy Vs GDP in Countries')
```

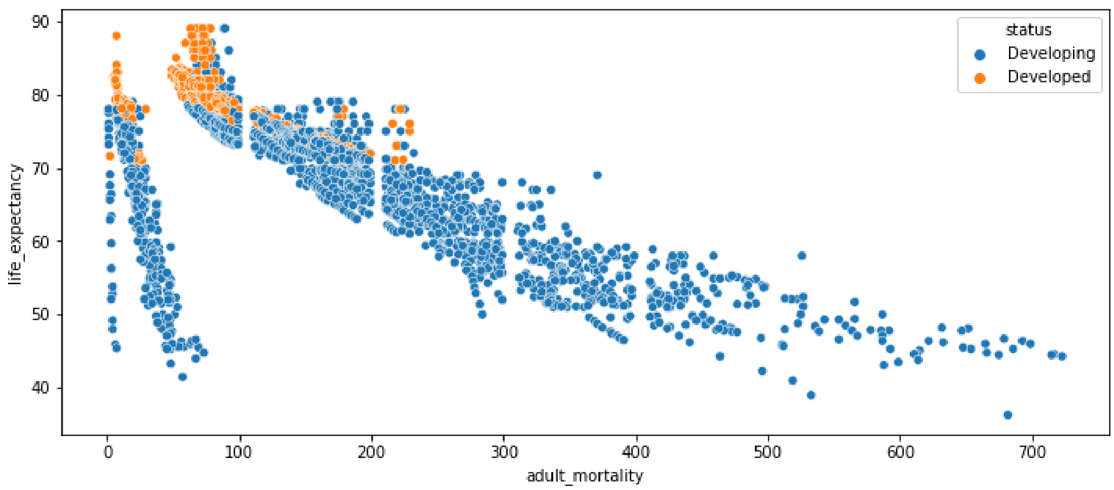
[34]:
```
# country and the sum population
fig=px.histogram(df,x='country',y = 'population', template='seaborn')
fig.show()
```

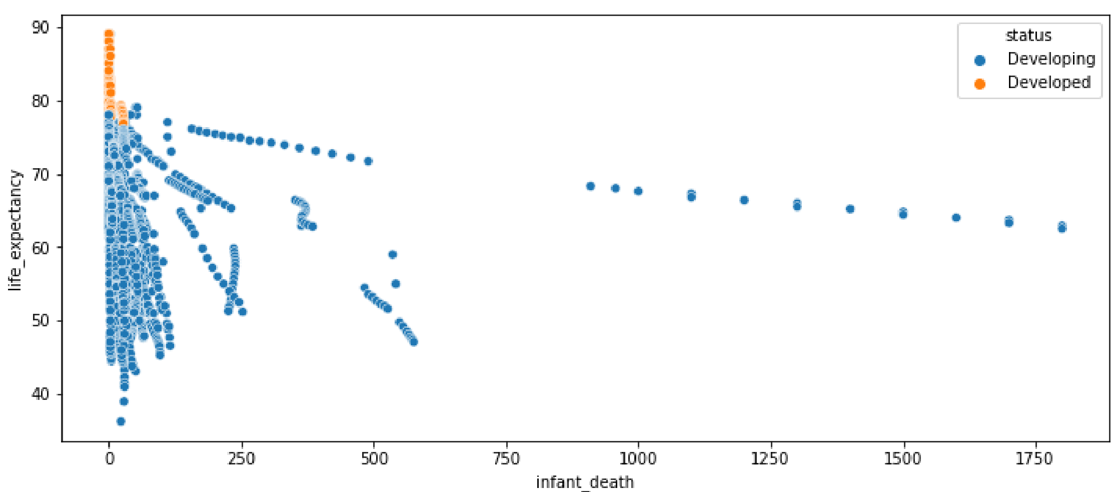
[35]:
```
#correlation plot
plt.figure(figsize=(20,20))
sns.heatmap(df.corr(),annot = True,cmap = "YlGnBu")
plt.show()
```

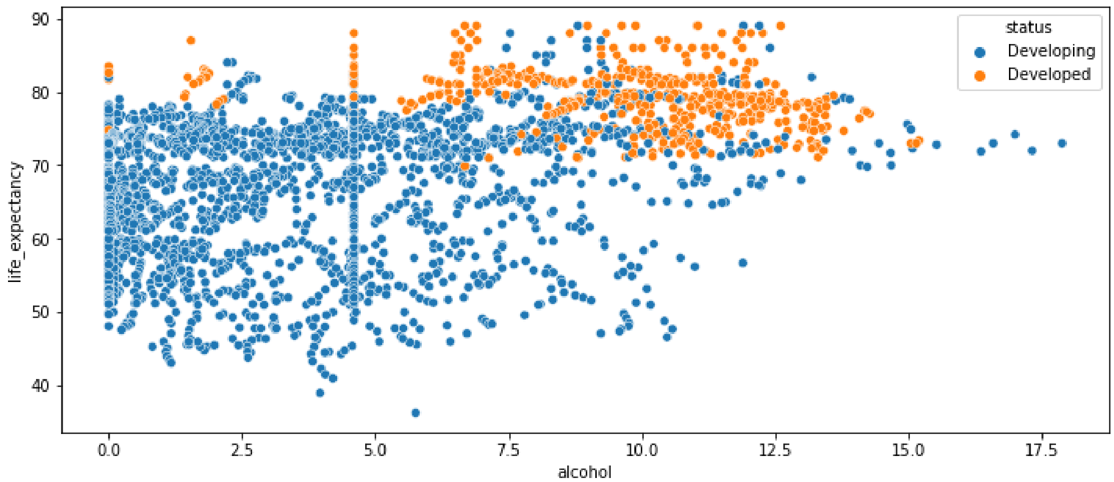


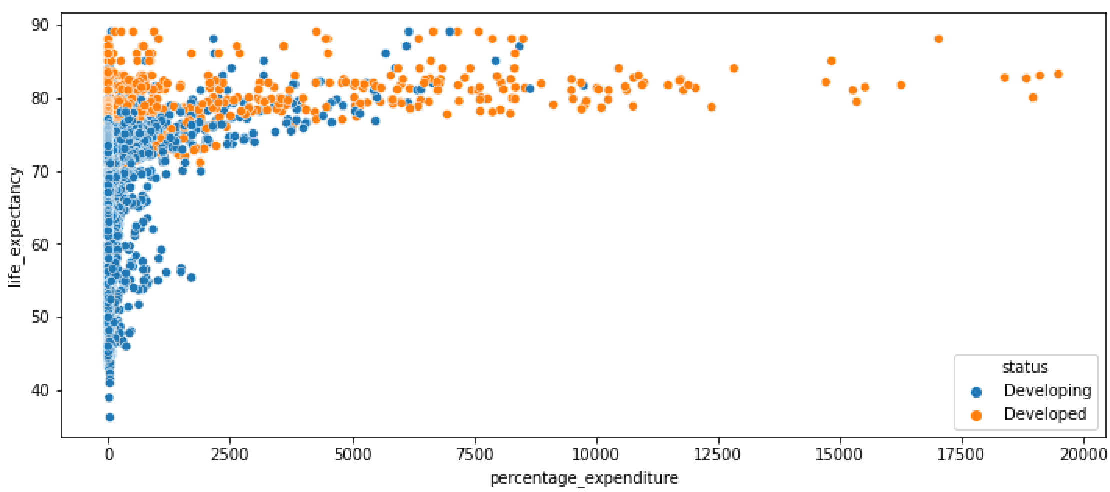
[36]:
```python
# plot by. status
y = df["life_expectancy"]
df_clean2 = df.drop("life_expectancy",axis=1)

for feature in df_clean2.select_dtypes(exclude="O").columns:
    plt.figure(figsize=(12,5))
    sns.scatterplot(x=df_clean2[feature],y=y,hue=df_clean2["status"])
```

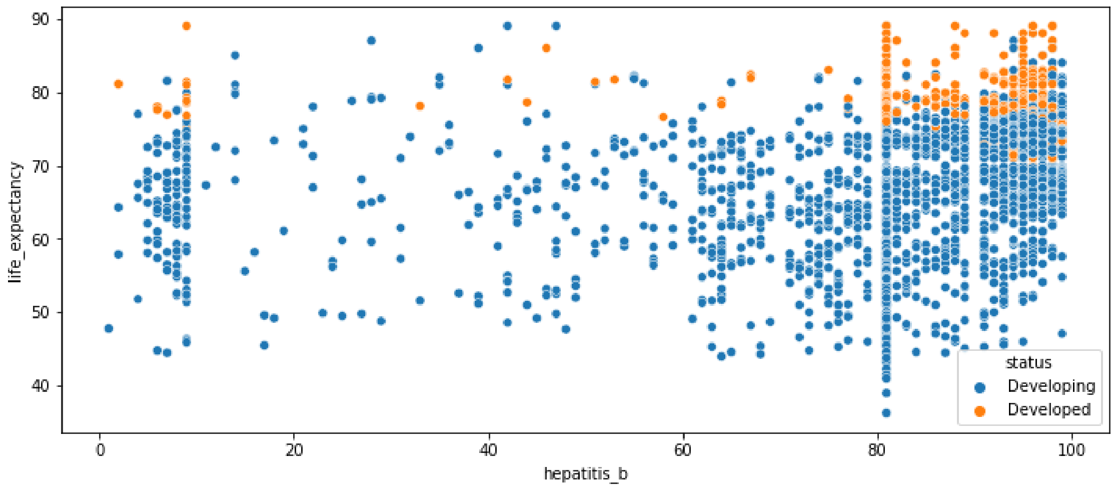

life_expectancy
year
status
Developing
Developed

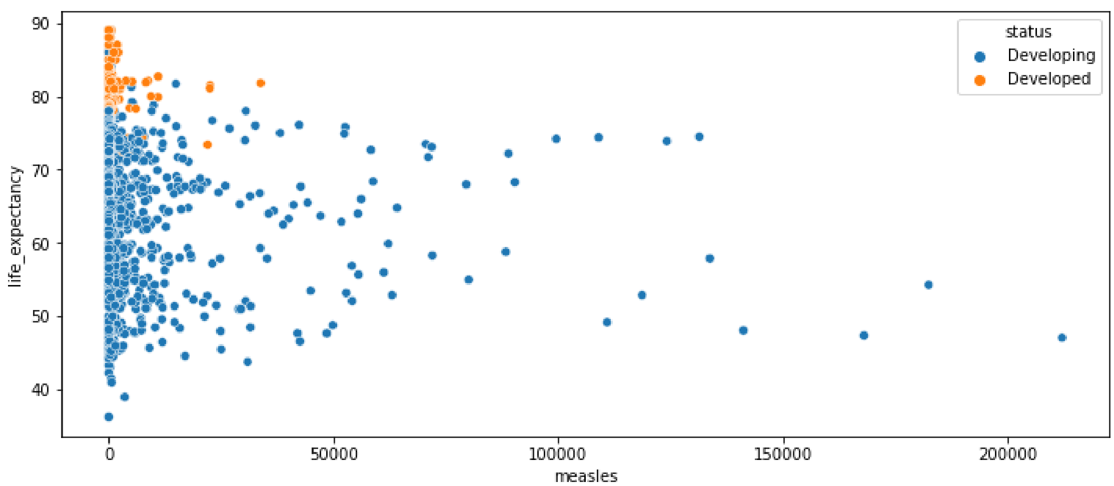

life_expectancy
adult_mortality
status
Developing
Developed

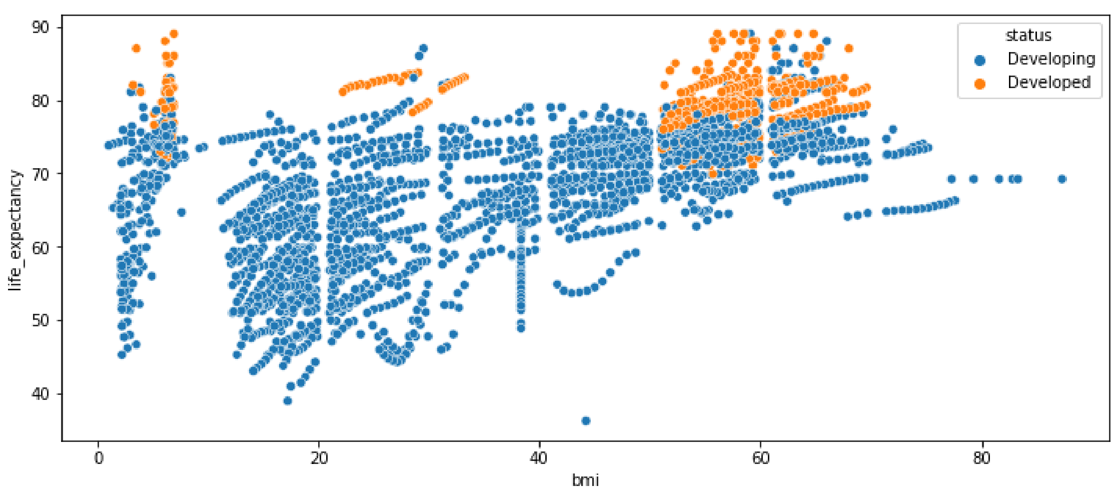

life_expectancy
infant_death
status
Developing
Developed

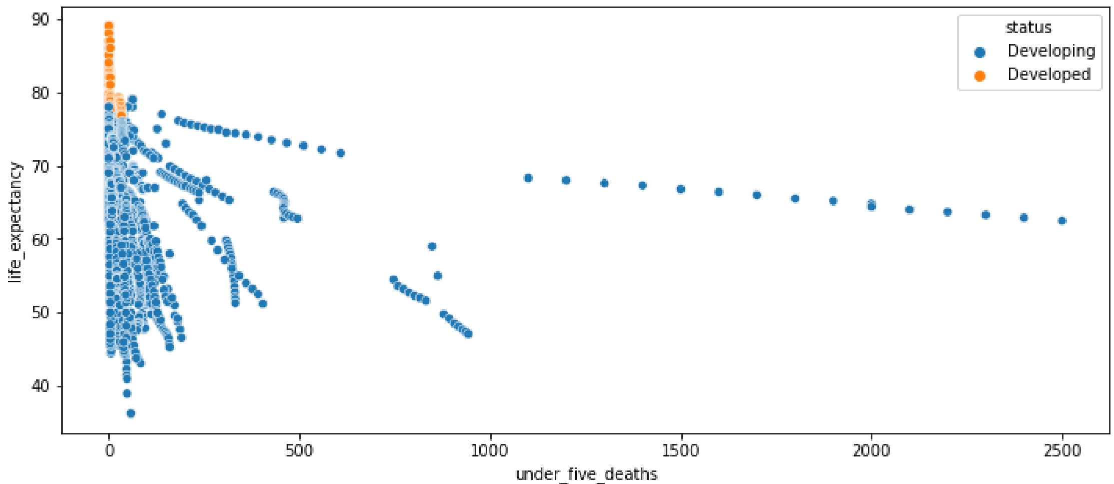

status
Developing
Developed
life_expectancy
alcohol
90
80
70
60
50
40
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

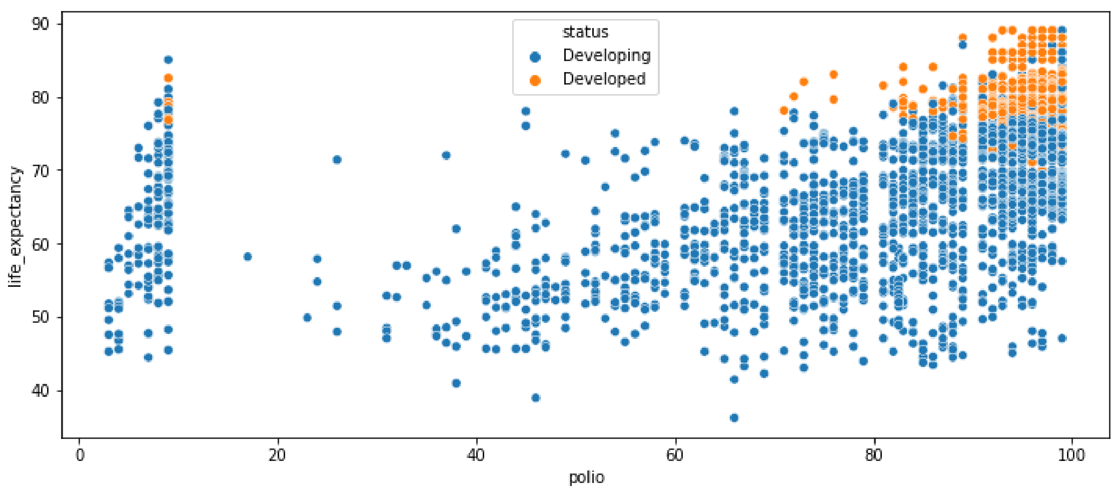

status
Developing
Developed
life_expectancy
percentage_expenditure
90
80
70
60
50
40
0
2500
5000
7500
10000
12500
15000
17500
20000

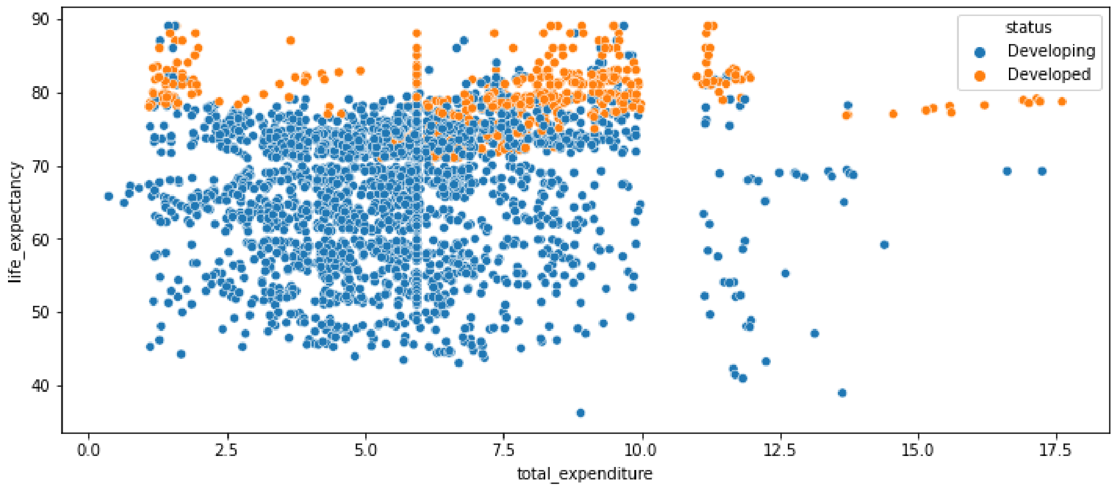

life_expectancy
hepatitis_b
status
Developing
Developed

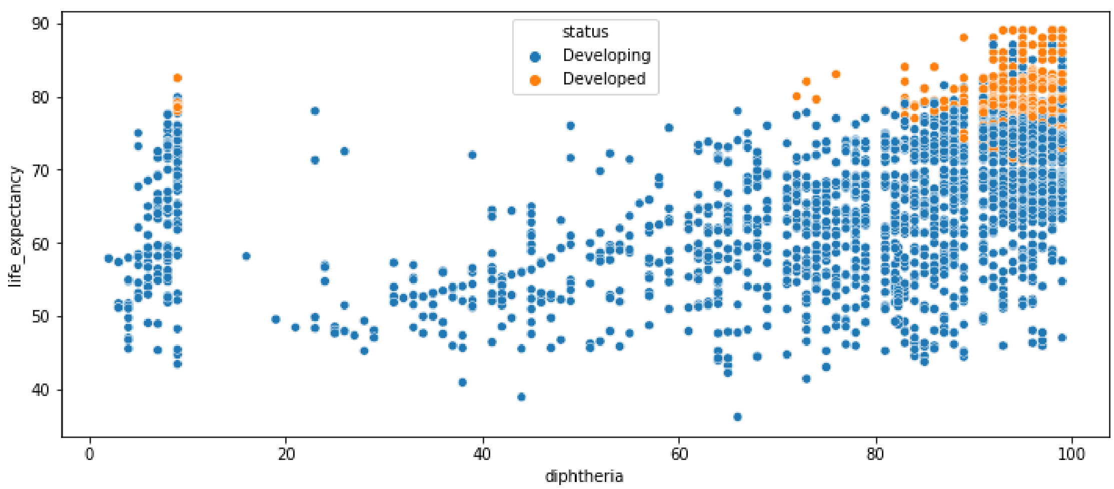

life_expectancy
measles
status
Developing
Developed

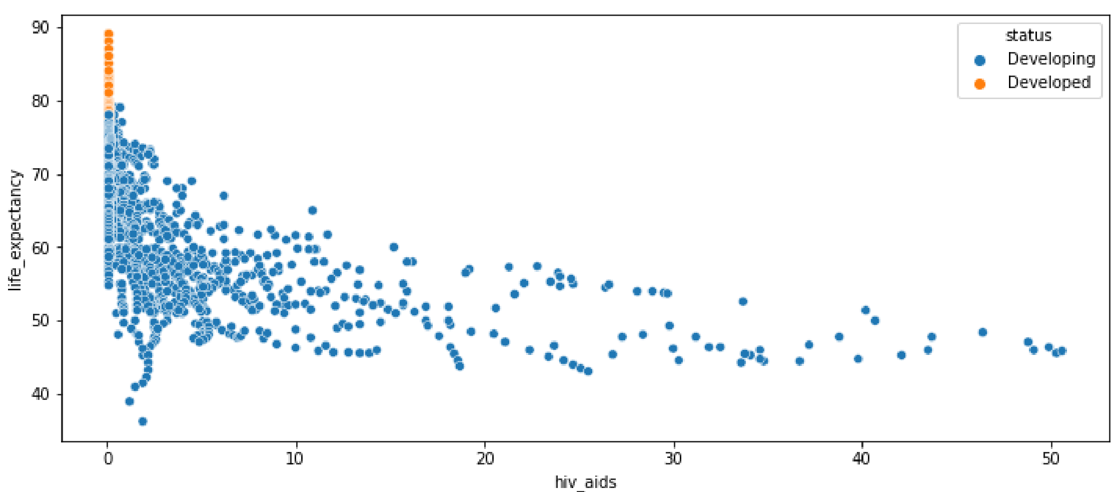

life_expectancy
bmi
status
Developing
Developed

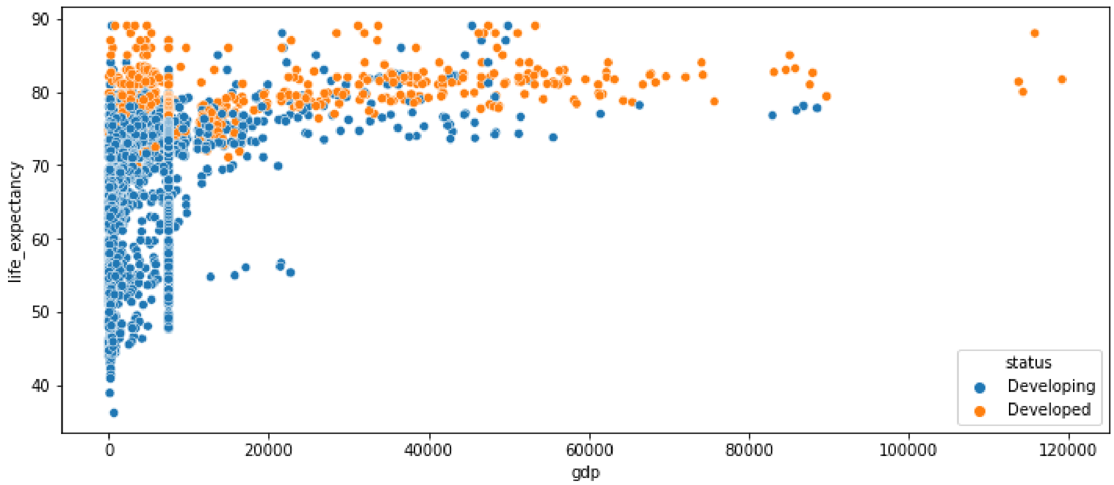

status
Developing
Developed
life_expectancy
under_five_deaths
90
80
70
60
50
40
0
500
1000
1500
2000
2500


status
Developing
Developed
life_expectancy
polio
90
80
70
60
50
40
0
20
40
60
80
100

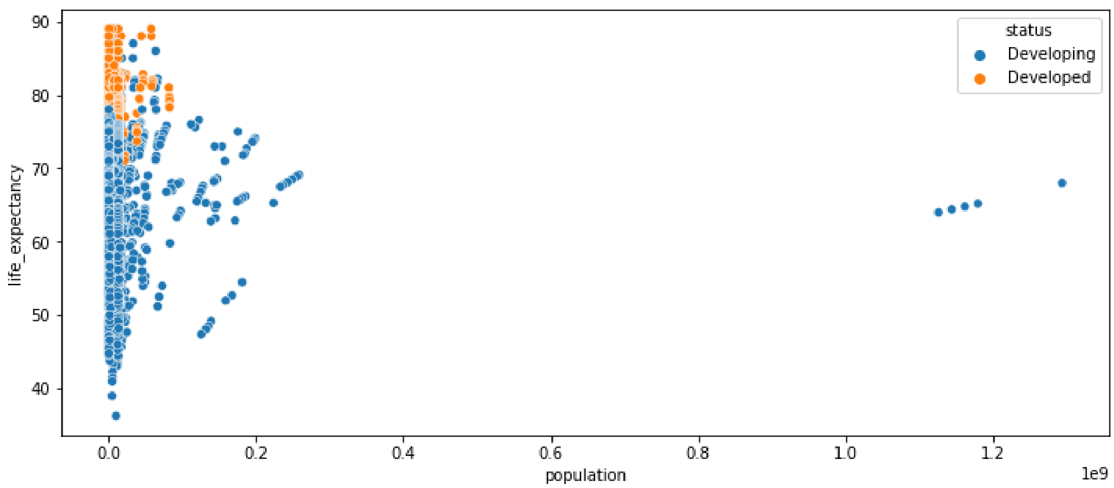

status
Developing
Developed
life_expectancy
total_expenditure
90
80
70
60
50
40
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5


status
Developing
Developed
life_expectancy
diphtheria
90
80
70
60
50
40
0
20
40
60
80
100

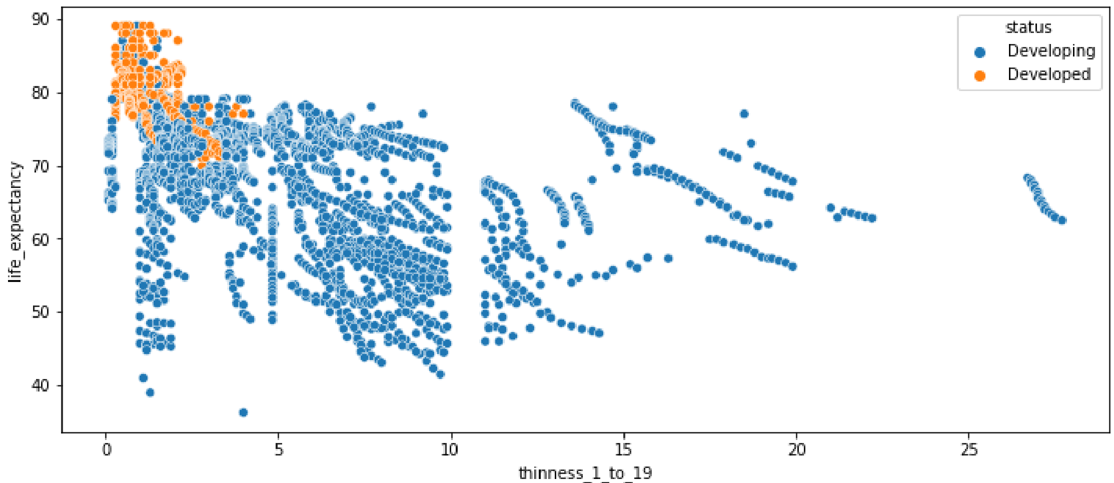

status
Developing
Developed
life_expectancy
hiv_aids
90
80
70
60
50
40
0
10
20
30
40
50

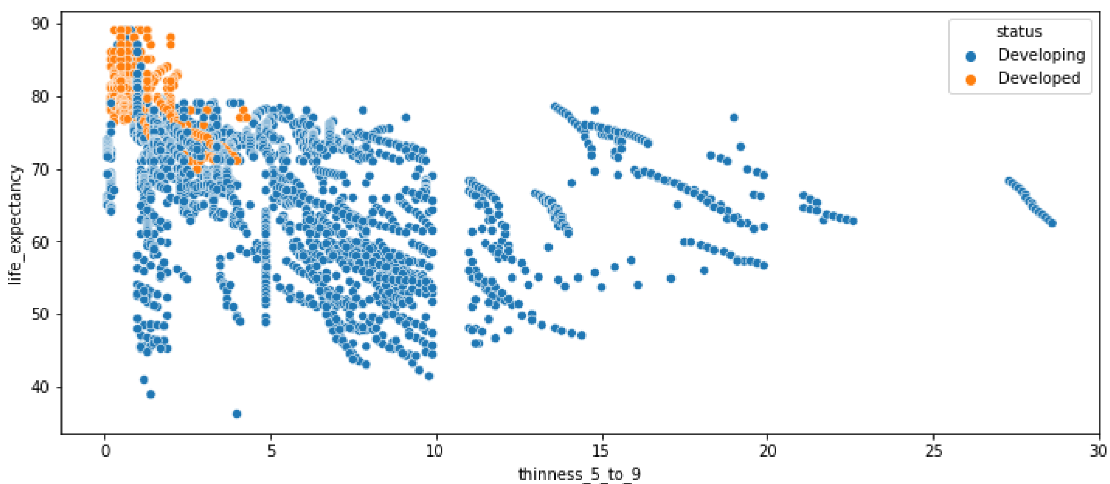


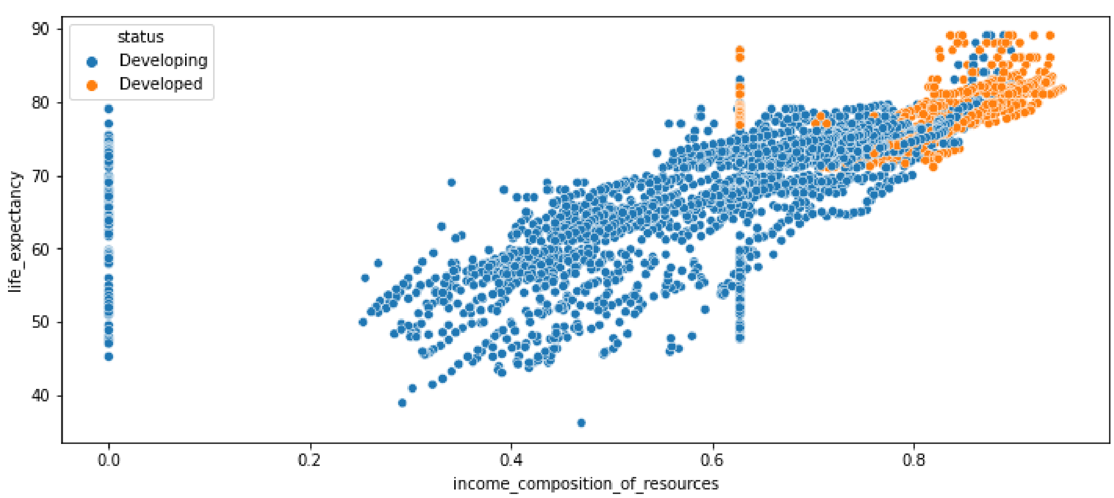
life_expectancy
gdp
status
Developing
Developed

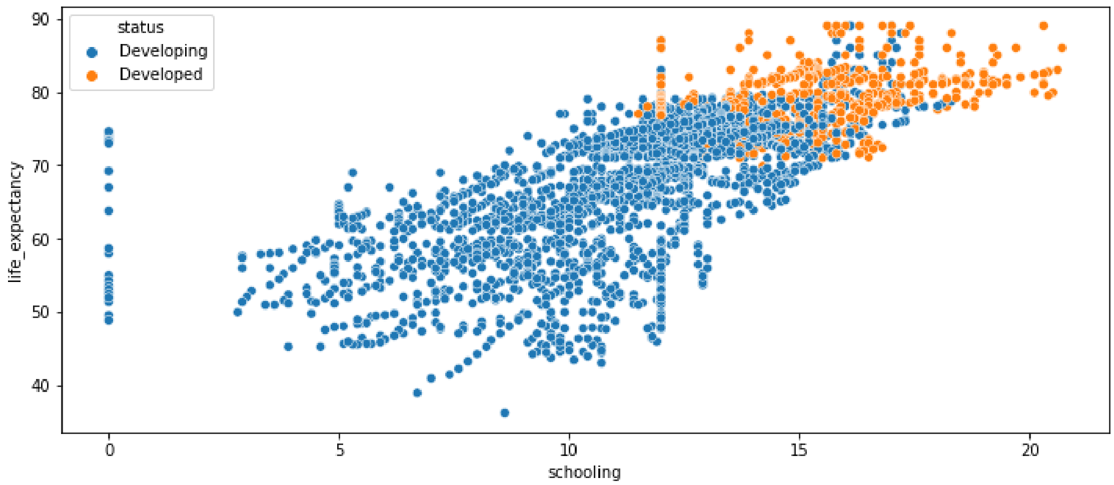


life_expectancy
population
1e9
status
Developing
Developed


status
Developing
Developed
life_expectancy
thinness_1_to_19
90
80
70
60
50
40
0
5
10
15
20
25


status
Developing
Developed
life_expectancy
thinness_5_to_9
90
80
70
60
50
40
0
5
10
15
20
25
30


status
Developing
Developed
life_expectancy
income_composition_of_resources
90
80
70
60
50
40
0.0
0.2
0.4
0.6
0.8



[94]:

```
# Let's made an interactive plot with the help of plotly
numerical_features = df.copy()
```

[95]:

```
countries = numerical_features["country"].unique()
def make_scatter_plot_country_wise(feature):
    first_title = countries[0]
    traces = []
    buttons = []
    frame = []

    for index,country in enumerate(countries):
        visible = [False] * len(countries)
        visible[index] = True
        name = country
        # Get the dataFrame currensponding to that country
        country_data = numerical_features.query('country == @country')

        traces.append(
            px.scatter(country_data, x=feature, y="life_expectancy",␣
↪color="year").update_traces(visible=True if index==0 else False).data[0]
        )

        buttons.append(dict(label=name,
                            method="update",
                            args=[{"visible":visible}, {"title":f"{name}"}]))

    fig = go.Figure(data=traces)
    fig.update_layout( xaxis_title=f"<b>{feature}</b>",
```

```
                         yaxis_title="<b>life_expectancy</b>",
                         legend_title="Year",
                         updatemenus=[go.layout.Updatemenu(
                             active=0,
                             buttons=buttons
                             )
                         ])
    fig.update_traces(marker_size=40)

    fig.update_layout(title=f"<b>{feature}</b>")
    fig.show()
```

[96]:
```
numerical_features.columns.values
```

[96]:
```
array(['country', 'year', 'status', 'life_expectancy', 'adult_mortality',
       'infant_death', 'alcohol', 'percentage_expenditure', 'hepatitis_b',
       'measles', 'bmi', 'under_five_deaths', 'polio',
       'total_expenditure', 'diphtheria', 'hiv_aids', 'gdp', 'population',
       'thinness_1_to_19', 'thinness_5_to_9',
       'income_composition_of_resources', 'schooling'], dtype=object)
```

[97]:
```
import plotly.graph_objs as go
```

[98]:
```
for i in numerical_features.columns.values:
    if i == "Country" or i == "Status" or i == "Life expectancy ":
        continue
    make_scatter_plot_country_wise(i)
```

[104]:
```
# worldmap plotly
from plotly.offline import init_notebook_mode, iplot
count = [ dict(
        # set the map type is choropleth
        type = 'choropleth',
        locations = df['country'],
        locationmode='country names',
        z = df['life_expectancy'],
        text = df['country'],
        colorscale = 'Viridis',
        autocolorscale = False,
        reversescale = True,
        # set the plotly gragh color
        marker = dict(
            line = dict (
                color = 'rgb(180,180,180)',
                width = 0.5
            ) ),
        # add a color bar
```

```
        colorbar = dict(
            autotick =False,
            title = 'Life Expectancy Country-based'),
      ) ]
# create layout for gragh
layout = dict(
    title = 'Life Expectancy across the Global',
    #
    geo = dict(
        showframe = True,
        showcoastlines = True,
        projection = dict(
            type = 'Mercator'
        )
    )
)
# prepare the fig parameter
fig = dict( data=count, layout=layout )
iplot( fig, validate=False, filename='d3-world-map' )
```

Dashboard with pandas-profiling

[ ]:
```
pip install pandas-profiling
```

[37]:
```
import pandas_profiling
report = df.profile_report(
    sort=None, html={"style": {"full_width": True}}, progress_bar=False
)
report
```

```
<IPython.core.display.HTML object>
```

[37]:

Linear Regression

[38]:
```
df_status = df.replace({"Developed":1,"Developing":0})
```

[40]:
```
#check status
df_status['status']
df_status
```

[40]:
```
          country  year  status  life_expectancy  adult_mortality  \
0     Afghanistan  2015       0             65.0            263.0
1     Afghanistan  2014       0             59.9            271.0
2     Afghanistan  2013       0             59.9            268.0
3     Afghanistan  2012       0             59.5            272.0
4     Afghanistan  2011       0             59.2            275.0
…             …     …     …                …                …
```

```
2933     Zimbabwe  2004       0             44.3            723.0
2934     Zimbabwe  2003       0             44.5            715.0
2935     Zimbabwe  2002       0             44.8             73.0
2936     Zimbabwe  2001       0             45.3            686.0
2937     Zimbabwe  2000       0             46.0            665.0

      infant_death  alcohol  percentage_expenditure  hepatitis_b  measles  \
0               62     0.01               71.279624         65.0     1154
1               64     0.01               73.523582         62.0      492
2               66     0.01               73.219243         64.0      430
3               69     0.01               78.184215         67.0     2787
4               71     0.01                7.097109         68.0     3013
…              …       …                      …            …        …
2933            27     4.36                0.000000         68.0       31
2934            26     4.06                0.000000          7.0      998
2935            25     4.43                0.000000         73.0      304
2936            25     1.72                0.000000         76.0      529
2937            24     1.68                0.000000         79.0     1483

      …  polio  total_expenditure  diphtheria  hiv_aids         gdp  \
0     …    6.0               8.16        65.0       0.1  584.259210
1     …   58.0               8.18        62.0       0.1  612.696514
2     …   62.0               8.13        64.0       0.1  631.744976
3     …   67.0               8.52        67.0       0.1  669.959000
4     …   68.0               7.87        68.0       0.1   63.537231
…   …     …                 …         …         …          …
2933  …   67.0               7.13        65.0      33.6  454.366654
2934  …    7.0               6.52        68.0      36.7  453.351155
2935  …   73.0               6.53        71.0      39.8   57.348340
2936  …   76.0               6.16        75.0      42.1  548.587312
2937  …   78.0               7.10        78.0      43.5  547.358878

      population  thinness_1_to_19  thinness_5_to_9  \
0     33736494.0              17.2             17.3
1       327582.0              17.5             17.5
2     31731688.0              17.7             17.7
3      3696958.0              17.9             18.0
4      2978599.0              18.2             18.2
…          …                …                …
2933  12777511.0               9.4              9.4
2934  12633897.0               9.8              9.9
2935    125525.0               1.2              1.3
2936  12366165.0               1.6              1.7
2937  12222251.0              11.0             11.2

      income_composition_of_resources  schooling
0                               0.479       10.1
```

```
1                                0.476       10.0
2                                0.470        9.9
3                                0.463        9.8
4                                0.454        9.5
…                                …         …
2933                             0.407        9.2
2934                             0.418        9.5
2935                             0.427       10.0
2936                             0.427        9.8
2937                             0.434        9.8

[2938 rows x 22 columns]
```

[41]:
```
from sklearn.linear_model import LinearRegression

Y = df_status["life_expectancy"]
X = df_status.drop(["life_expectancy","country"], axis=1)
lrm = LinearRegression()
lrm.fit(X,Y)
```

[41]:
```
LinearRegression()
```

[42]:
```
import statsmodels.api as sm

X = sm.add_constant(X)
results = sm.OLS(Y, X).fit()
results.summary()
```

[42]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:        life_expectancy   R-squared:                       0.820
Model:                            OLS   Adj. R-squared:                  0.819
Method:                 Least Squares   F-statistic:                     663.3
Date:                Mon, 12 Dec 2022   Prob (F-statistic):               0.00
Time:                        17:58:51   Log-Likelihood:                -8268.0
No. Observations:                2938   AIC:                         1.658e+04
Df Residuals:                    2917   BIC:                         1.670e+04
Df Model:                          20
Covariance Type:            nonrobust
================================================================================
===================
                                      coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
-------------------
```

```
const                                73.4394     34.723      2.115      0.035
5.356     141.523
year                                 -0.0092      0.017     -0.533      0.594
-0.043       0.025
status                                1.5897      0.270      5.886      0.000
1.060       2.119
adult_mortality                      -0.0198      0.001    -24.926      0.000
-0.021      -0.018
infant_death                          0.0998      0.008     11.839      0.000
0.083       0.116
alcohol                               0.0620      0.026      2.381      0.017
0.011       0.113
percentage_expenditure             8.534e-05   8.47e-05      1.008      0.314
-8.07e-05       0.000
hepatitis_b                          -0.0147      0.004     -3.752      0.000
-0.022      -0.007
measles                            -1.96e-05   7.66e-06     -2.558      0.011
-3.46e-05   -4.58e-06
bmi                                   0.0444      0.005      8.998      0.000
0.035       0.054
under_five_deaths                    -0.0747      0.006    -12.094      0.000
-0.087      -0.063
polio                                 0.0285      0.004      6.385      0.000
0.020       0.037
total_expenditure                     0.0661      0.034      1.930      0.054
-0.001       0.133
diphtheria                            0.0402      0.005      8.544      0.000
0.031       0.049
hiv_aids                             -0.4708      0.018    -26.667      0.000
-0.505      -0.436
gdp                                3.347e-05    1.3e-05      2.571      0.010
7.94e-06     5.9e-05
population                         2.751e-10   1.69e-09      0.163      0.871
-3.04e-09    3.59e-09
thinness_1_to_19                     -0.0818      0.050     -1.624      0.105
-0.181       0.017
thinness_5_to_9                       0.0073      0.050      0.147      0.883
-0.090       0.105
income_composition_of_resources       5.7738      0.641      9.003      0.000
4.516       7.031
schooling                             0.6574      0.042     15.693      0.000
0.575       0.740
==============================================================================
Omnibus:                      135.918   Durbin-Watson:                   0.701
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              398.080
Skew:                          -0.175   Prob(JB):                     3.62e-87
Kurtosis:                       4.769   Cond. No.                     2.57e+10
```

```
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.57e+10. This might indicate that there are
strong multicollinearity or other numerical problems.
"""
```

KNN

[43]:
```python
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.25,␣
 ↪random_state=123)

X_test = sm.add_constant(X_test)
y_preds = results.predict(X_test)
```

[44]:
```python
from sklearn.neighbors import KNeighborsRegressor

knn = KNeighborsRegressor()
knn.fit(X_train,y_train)
```

[44]:
```
KNeighborsRegressor()
```

[45]:
```python
from sklearn.model_selection import GridSearchCV

knn_parameters = {"n_neighbors":range(1,10),
                  "weights":["uniform","distance"],
                }

grid_knn = GridSearchCV(estimator=knn,
                       param_grid = knn_parameters,
                       cv = 10)

grid_knn.fit(X, Y)
```

[45]:
```
GridSearchCV(cv=10, estimator=KNeighborsRegressor(),
             param_grid={'n_neighbors': range(1, 10),
                         'weights': ['uniform', 'distance']})
```

[46]:
```python
print("Best R-squared score::{}".format(grid_knn.best_score_))
print("Best parameters::\n{}".format(grid_knn.best_params_))
```

```
Best R-squared score::0.003953374919898l9
Best parameters::
{'n_neighbors': 9, 'weights': 'uniform'}
```

PCA

PCA is used to preprocess the data to perform K-Means Clustering

[47]:
```python
from sklearn.decomposition import PCA
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
```

[48]:
```python
df.iloc[:,3:].apply(pd.to_numeric,errors='coerce')
```

[48]:
```
      life_expectancy  adult_mortality  infant_death  alcohol  \
0                65.0            263.0            62     0.01
1                59.9            271.0            64     0.01
2                59.9            268.0            66     0.01
3                59.5            272.0            69     0.01
4                59.2            275.0            71     0.01
…                 …               …             …      …
2933             44.3            723.0            27     4.36
2934             44.5            715.0            26     4.06
2935             44.8             73.0            25     4.43
2936             45.3            686.0            25     1.72
2937             46.0            665.0            24     1.68

      percentage_expenditure  hepatitis_b  measles   bmi  under_five_deaths  \
0                  71.279624         65.0     1154  19.1                 83
1                  73.523582         62.0      492  18.6                 86
2                  73.219243         64.0      430  18.1                 89
3                  78.184215         67.0     2787  17.6                 93
4                   7.097109         68.0     3013  17.2                 97
…                        …           …       …    …                …
2933                0.000000         68.0       31  27.1                 42
2934                0.000000          7.0      998  26.7                 41
2935                0.000000         73.0      304  26.3                 40
2936                0.000000         76.0      529  25.9                 39
2937                0.000000         79.0     1483  25.5                 39

      polio  total_expenditure  diphtheria  hiv_aids         gdp  population  \
0       6.0               8.16        65.0       0.1  584.259210  33736494.0
1      58.0               8.18        62.0       0.1  612.696514    327582.0
2      62.0               8.13        64.0       0.1  631.744976  31731688.0
3      67.0               8.52        67.0       0.1  669.959000   3696958.0
4      68.0               7.87        68.0       0.1   63.537231   2978599.0
…     …                  …           …         …          …           …
2933   67.0               7.13        65.0      33.6  454.366654  12777511.0
2934    7.0               6.52        68.0      36.7  453.351155  12633897.0
2935   73.0               6.53        71.0      39.8   57.348340    125525.0
2936   76.0               6.16        75.0      42.1  548.587312  12366165.0
```

```
2937   78.0               7.10        78.0      43.5  547.358878  12222251.0

      thinness_1_to_19  thinness_5_to_9  income_composition_of_resources  \
0                 17.2             17.3                             0.479
1                 17.5             17.5                             0.476
2                 17.7             17.7                             0.470
3                 17.9             18.0                             0.463
4                 18.2             18.2                             0.454
…                  …                …                               …
2933               9.4              9.4                             0.407
2934               9.8              9.9                             0.418
2935               1.2              1.3                             0.427
2936               1.6              1.7                             0.427
2937              11.0             11.2                             0.434

      schooling
0          10.1
1          10.0
2           9.9
3           9.8
4           9.5
…           …
2933        9.2
2934        9.5
2935       10.0
2936        9.8
2937        9.8

[2938 rows x 19 columns]
```

[49]:

```
drop_list = ["life_expectancy","country"]
df.drop(drop_list, axis=1 ,inplace=True)
df1 = df.copy()
group = []
for i in df1.columns:
    if (df1[i].dtypes == "object"):
        group.append(i)

#print(group)

lbl_encode = LabelEncoder()
for i in group :
    df1[i]=lbl_encode.fit_transform(df1[[i]])
```

```
/Users/polina/opt/anaconda3/lib/python3.9/site-
packages/sklearn/preprocessing/_label.py:115: DataConversionWarning:

A column-vector y was passed when a 1d array was expected. Please change the
```

```
shape of y to (n_samples, ), for example using ravel().
```

[50]:

```python
scaler = StandardScaler()
scaler.fit(df1)
scaled_df = pd.DataFrame(scaler.transform(df1), columns=df1.columns)
```

[51]:

```python
scaled_df.head()
```

[51]:

```
       year    status  adult_mortality  infant_death   alcohol  \
0  1.621762  0.459399         0.791586      0.268824 -1.172958
1  1.404986  0.459399         0.856072      0.285786 -1.172958
2  1.188210  0.459399         0.831890      0.302749 -1.172958
3  0.971434  0.459399         0.864132      0.328193 -1.172958
4  0.754658  0.459399         0.888314      0.345155 -1.172958

   percentage_expenditure  hepatitis_b   measles       bmi  under_five_deaths  \
0               -0.335570    -0.705861 -0.110384 -0.964715           0.255359
1               -0.334441    -0.838704 -0.168124 -0.989810           0.274060
2               -0.334594    -0.750142 -0.173531 -1.014905           0.292761
3               -0.332096    -0.617299  0.032045 -1.040000           0.317696
4               -0.367862    -0.573018  0.051757 -1.060076           0.342631

      polio  total_expenditure  diphtheria  hiv_aids       gdp  population  \
0 -3.278638           0.925806   -0.732952 -0.323445 -0.525248    0.389975
1 -1.051482           0.934140   -0.859877 -0.323445 -0.523083   -0.230936
2 -0.880163           0.913306   -0.775260 -0.323445 -0.521632    0.352715
3 -0.666013           1.075815   -0.648335 -0.323445 -0.518723   -0.168315
4 -0.623183           0.804966   -0.606027 -0.323445 -0.564893   -0.181666

   thinness_1_to_19  thinness_5_to_9  income_composition_of_resources  \
0          2.813130         2.773279                        -0.725401
1          2.881408         2.817902                        -0.740050
2          2.926927         2.862526                        -0.769349
3          2.972446         2.929461                        -0.803531
4          3.040724         2.974085                        -0.847480

   schooling
0  -0.579931
1  -0.610570
2  -0.641209
3  -0.671847
4  -0.763764
```

[52]:

```python
# The number of dimensions as 3
pca = PCA(n_components=3)
pca.fit(scaled_df)
```

```
pca_data = pd.DataFrame(pca.transform(scaled_df), columns=["c1", "c2",␣
↪"c3"])#"c4","c5"])
```

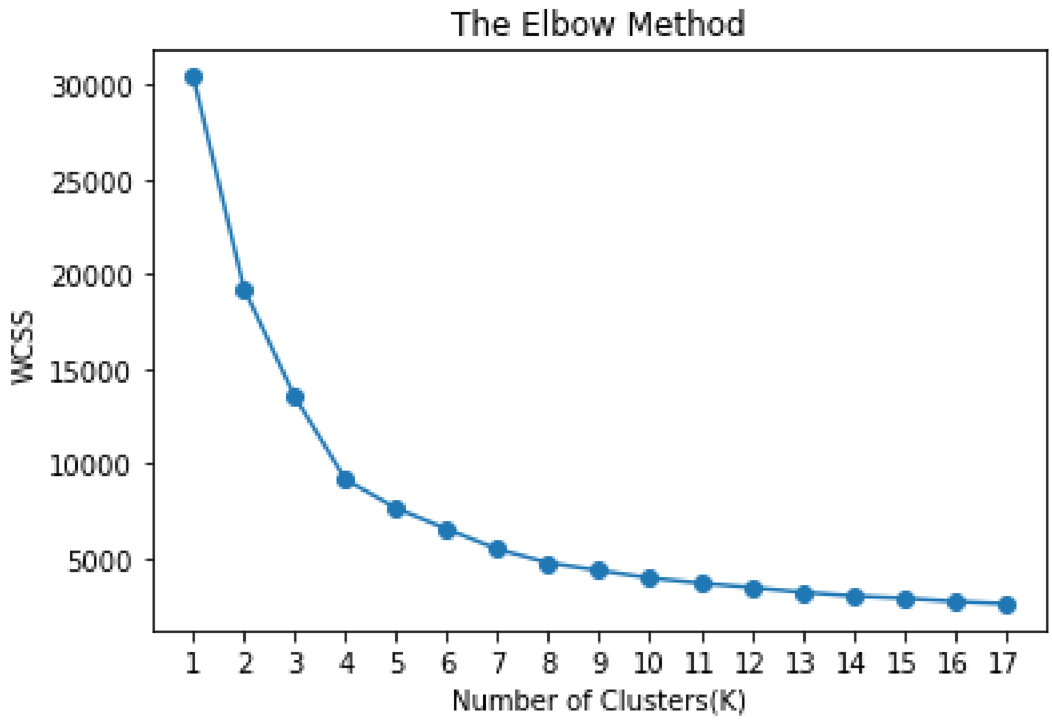
[54]:
```
import plotly.graph_objs as go
x = pca_data["c1"]
y = pca_data["c2"]
z = pca_data["c3"]

fig = go.Figure(data=[go.Scatter3d(
    x=x,y=y,z=z,mode='markers',
    marker=dict(size=6,color=x,opacity=0.8))])

fig.update_layout( title={'text': "3D Plot of Size-Reduced Data",'y':0.9,
        'x':0.5,'xanchor': 'center','yanchor': 'top'},
                  margin=dict(l=200, r=220, b=0, t=0))
fig.show()
```

K Means Clustering using Elbow Method

[55]:
```
from sklearn.cluster import KMeans
```

- Each cluster is formed by calculation and comparing the distance of data point withon a cluster to its center
- Within-Cluster-Sum-of-Squares(WCSS) - to fund the right number of clusters. WCSS is the sum of squares of the distances of each data point in all clusters to their respective centers, and the goal is to minimize the sum. Assume there are n observations in a dataset and we specify n number of clusters, which means k = n; so WCSS turns to 0 since data points themselves become centers and the distance will be 0, in turn this will perform a perfect cluster; but this is almost impossible as we have many clusters as the observations. Thus, we use Elbow point graph to find the optimum value for K by fitting the model in a range of values of K. We randomly initialize the K-Means algorithm for a range of K values and plot it against the WCSS for each K value.

[56]:
```
#https://www.geeksforgeeks.org/elbow-method-for-optimal-value-of-k-in-kmeans/
wcss = []
k = range(1,18)
for i in k:
    model = KMeans(n_clusters=i)
    model.fit(pca_data)
    wcss.append(model.inertia_)

plt.plot(k, wcss, '-o')
plt.title('The Elbow Method')
plt.xlabel('Number of Clusters(K)')
plt.ylabel('WCSS')
plt.xticks(k)
plt.show()
```

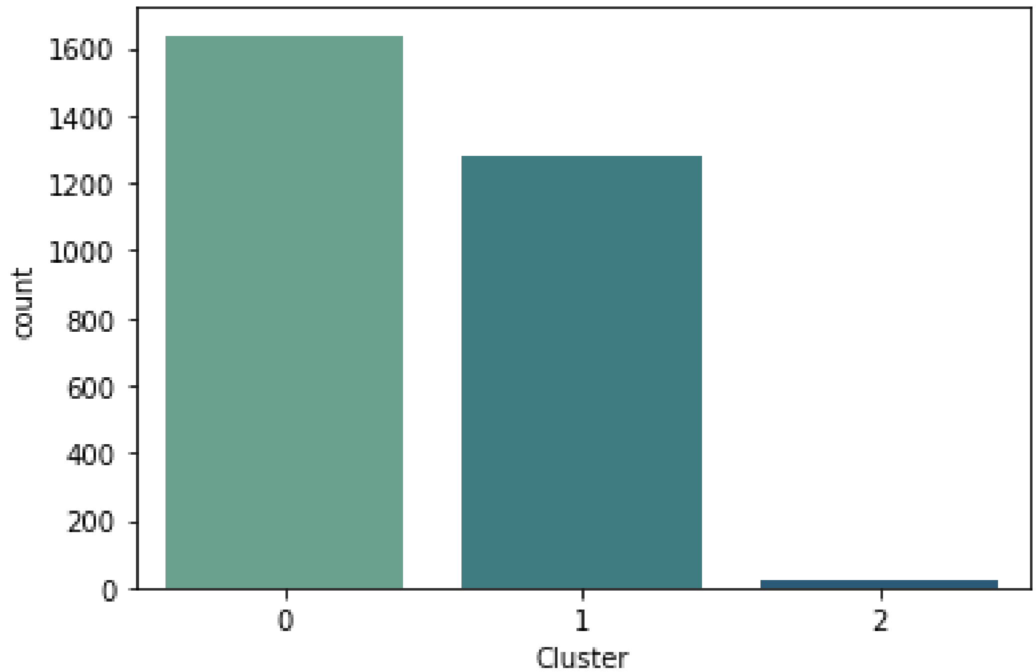


- optimum value for K = 3
- increase in the number of clusters, the WCSS value decreases
- select the value for K, the "elbow", on the basis of the rate of decrease, to indicate the model fits best at that point. In the graph, from cluster 1 to 2 to 3 in the above graph there is a huge drop in WCSS. After 3 the drop is minimal, thus we chose 3 to be the optimal value for K. Based on the Elbow Method, we can find the optimal number of clusters is 3. https://en.wikipedia.org/wiki/Elbow_method_(clustering)

[57]:
```
k_means = KMeans(n_clusters = 3, random_state = 100)
y_pred = k_means.fit_predict(pca_data)
pca_data['Cluster'] = y_pred
df['Cluster'] = y_pred
```

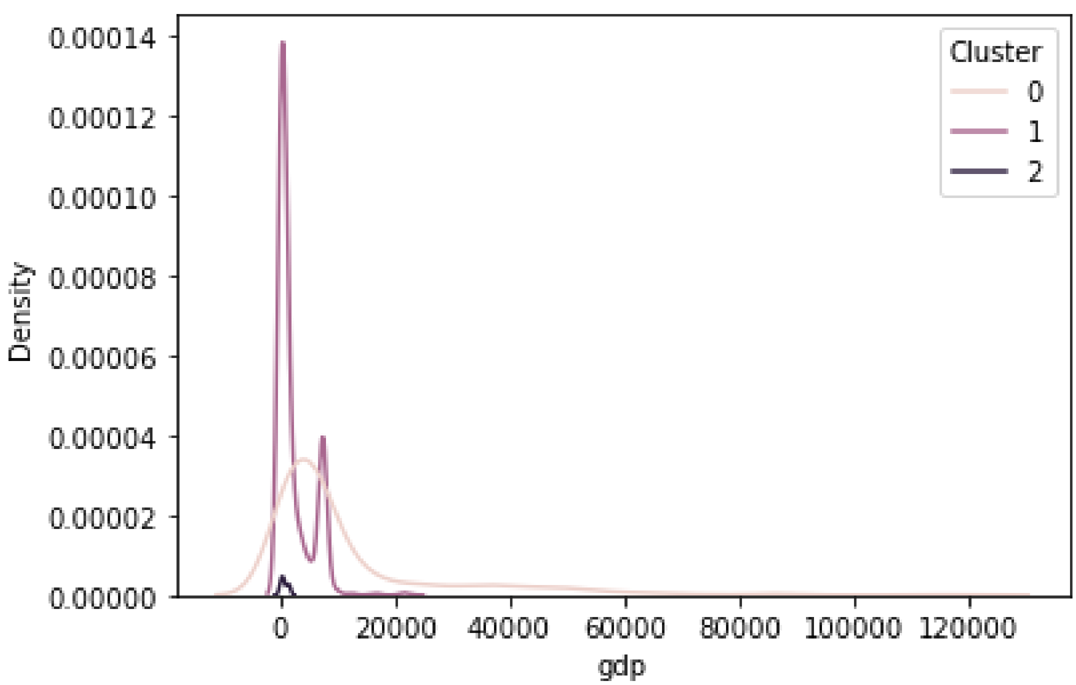
[58]:
```
sns.countplot(x=pca_data['Cluster'], palette = 'crest')
```

[58]: <AxesSubplot:xlabel='Cluster', ylabel='count'>



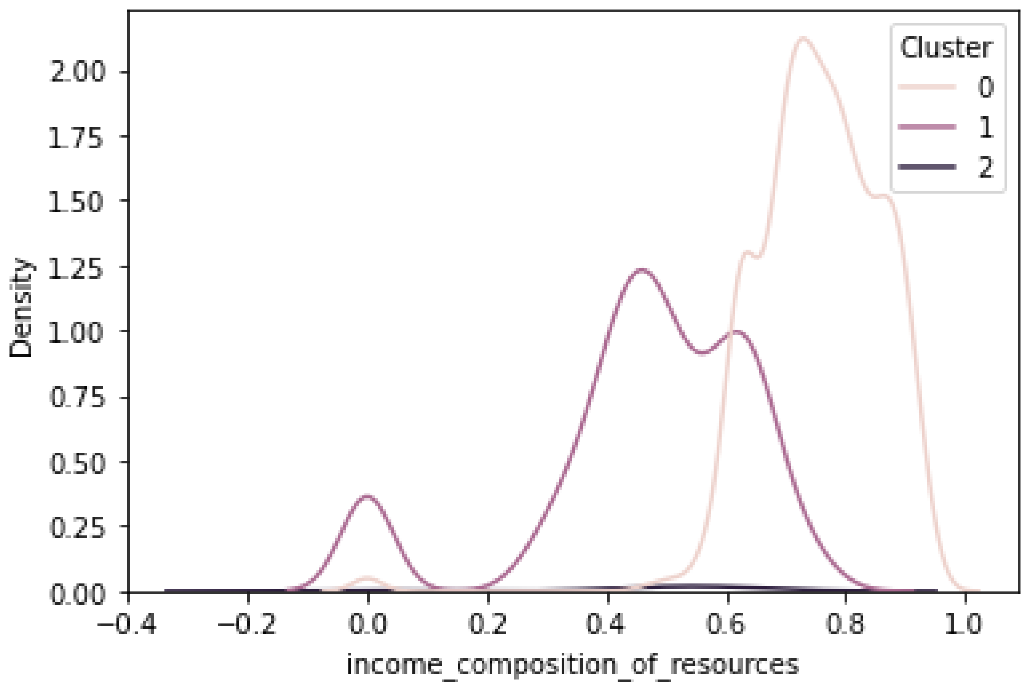
[59]:
```
#GDP
sns.kdeplot(data=df, x='gdp', hue='Cluster')
```

[59]: <AxesSubplot:xlabel='gdp', ylabel='Density'>

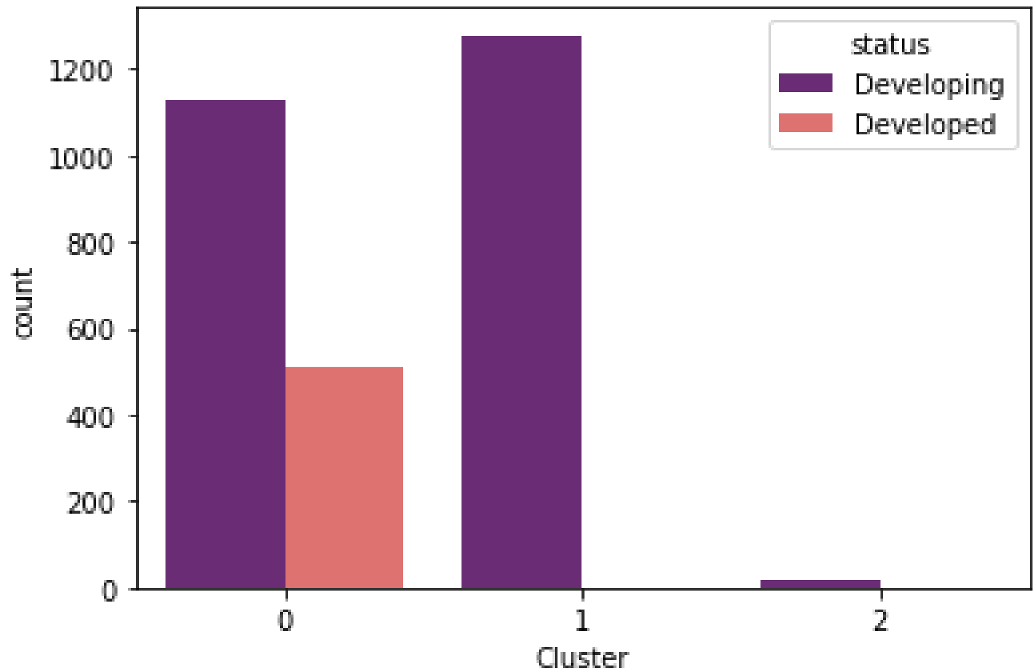


[60]:
```
#Income composition of resources
sns.kdeplot(data=df, x='income_composition_of_resources', hue='Cluster')
```

[60]: <AxesSubplot:xlabel='income_composition_of_resources', ylabel='Density'>



[61]:
```
#Status
profile = ['status']

for i in profile:
    plt.figure()
    sns.countplot(x='Cluster', data=df, hue=df[i],palette = 'magma')
    plt.show()
```



status
Developing
Developed
count
0
200
400
600
800
1000
1200
0
1
2
Cluster

[ ]: