Importing Data

In [1]:
```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns
from matplotlib import pyplot as plt

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_squared_log_error
```

Disable Warnings

In [2]:
```
import warnings
warnings.filterwarnings("ignore")
warnings.simplefilter("ignore")
```

Loading Data

In [3]:
```
data = pd.read_csv("/kaggle/input/gdp-analysis-dataset/world.csv")
```

In [4]:
```
data
```

Out[4]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infa mortal (p 10 birtl
0	Afghanistan	ASIA (EX. NEAR EAST)	31056997	647500	48,0	0,00	23,06	163,
1	Albania	EASTERN EUROPE	3581655	28748	124,6	1,26	-4,93	21,
2	Algeria	NORTHERN AFRICA	32930091	2381740	13,8	0,04	-0,39	
3	American Samoa	OCEANIA	57794	199	290,4	58,29	-20,71	9,
4	Andorra	WESTERN EUROPE	71201	468	152,1	0,00	6,6	4,
...	...	...	...	...	...	...	...	
222	West Bank	NEAR EAST	2460492	5860	419,9	0,00	2,98	19,
223	Western Sahara	NORTHERN AFRICA	273008	266000	1,0	0,42	NaN	N
224	Yemen	NEAR EAST	21456188	527970	40,6	0,36	0	6
225	Zambia	SUB-SAHARAN AFRICA	11502010	752614	15,3	0,00	0	88,
226	Zimbabwe	SUB-SAHARAN AFRICA	12236805	390580	31,3	0,00	0	67,

227 rows × 20 columns

Data Statistics

In [5]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 227 entries, 0 to 226
Data columns (total 20 columns):
 #   Column                                Non-Null Count  Dtype
---  ------                                --------------  -----
 0   Country                               227 non-null    object
 1   Region                                227 non-null    object
 2   Population                            227 non-null    int64
 3   Area (sq. mi.)                        227 non-null    int64
 4   Pop. Density (per sq. mi.)            227 non-null    object
 5   Coastline (coast/area ratio)          227 non-null    object
 6   Net migration                         224 non-null    object
 7   Infant mortality (per 1000 births)    224 non-null    object
 8   GDP ($ per capita)                    226 non-null    float64
 9   Literacy (%)                          209 non-null    object
 10  Phones (per 1000)                     223 non-null    object
 11  Arable (%)                            225 non-null    object
 12  Crops (%)                             225 non-null    object
 13  Other (%)                             225 non-null    object
 14  Climate                               205 non-null    object
 15  Birthrate                             224 non-null    object
 16  Deathrate                             223 non-null    object
 17  Agriculture                           212 non-null    object
 18  Industry                              211 non-null    object
 19  Service                               212 non-null    object
dtypes: float64(1), int64(2), object(17)
memory usage: 35.6+ KB
```

In [6]:
```
data.describe()
```

Out[6]:

	Population	Area (sq. mi.)	GDP ($ per capita)
count	2.270000e+02	2.270000e+02	226.000000
mean	2.874028e+07	5.982270e+05	9689.823009
std	1.178913e+08	1.790282e+06	10049.138513
min	7.026000e+03	2.000000e+00	500.000000
25%	4.376240e+05	4.647500e+03	1900.000000
50%	4.786994e+06	8.660000e+04	5550.000000
75%	1.749777e+07	4.418110e+05	15700.000000
max	1.313974e+09	1.707520e+07	55100.000000

In [7]:
```
data.isna().sum
```

Out[7]:
```
<bound method DataFrame.sum of      Country  Region  Population  Area (sq.
mi.)  Pop. Density (per sq. mi.)  \
0      False   False       False           False                       Fals
e
1      False   False       False           False                       Fals
e
2      False   False       False           False                       Fals
e
3      False   False       False           False                       Fals
e
4      False   False       False           False                       Fals
e
..       ...     ...         ...             ...
...
222    False   False       False           False                       Fals
e
223    False   False       False           False                       Fals
e
224    False   False       False           False                       Fals
e
225    False   False       False           False                       Fals
e
226    False   False       False           False                       Fals
e

     Coastline (coast/area ratio)  Net migration  \
0                           False          False
1                           False          False
2                           False          False
3                           False          False
4                           False          False
..                            ...            ...
222                         False          False
223                         False           True
224                         False          False
225                         False          False
226                         False          False

     Infant mortality (per 1000 births)  GDP ($ per capita)  Literacy (%)
\
0                                 False              False         False
1                                 False              False         False
2                                 False              False         False
3                                 False              False         False
4                                 False              False         False
..                                  ...                ...           ...
222                               False              False          True
223                                True               True          True
224                               False              False         False
225                               False              False         False
226                               False              False         False

     Phones (per 1000)  Arable (%)  Crops (%)  Other (%)  Climate  Birthrat
e  \
0                False       False      False      False    False      Fals
e
```

```
1               False       False      False      False    False      Fals
e
2               False       False      False      False    False      Fals
e
3               False       False      False      False    False      Fals
e
4               False       False      False      False    False      Fals
e
..                ...         ...        ...        ...      ...
...
222             False       False      False      False    False      Fals
e
223              True       False      False      False    False       Tru
e
224             False       False      False      False    False      Fals
e
225             False       False      False      False    False      Fals
e
226             False       False      False      False    False      Fals
e

     Deathrate  Agriculture  Industry  Service
0        False        False     False    False
1        False        False     False    False
2        False        False     False    False
3        False         True      True     True
4        False         True      True     True
..         ...          ...       ...      ...
222      False        False     False    False
223       True         True      True    False
224      False        False     False    False
225      False        False     False    False
226      False        False     False    False

[227 rows x 20 columns]>
```

In [8]:
```
data.isnull().sum()
```

Out[8]:
```
Country                               0
Region                                0
Population                            0
Area (sq. mi.)                        0
Pop. Density (per sq. mi.)            0
Coastline (coast/area ratio)          0
Net migration                         3
Infant mortality (per 1000 births)    3
GDP ($ per capita)                    1
Literacy (%)                         18
Phones (per 1000)                     4
Arable (%)                            2
Crops (%)                             2
Other (%)                             2
Climate                              22
Birthrate                             3
Deathrate                             4
Agriculture                          15
Industry                             16
Service                              15
dtype: int64
```

In [9]:
```
data.isna().sum
```

Out[9]:
```
<bound method DataFrame.sum of      Country  Region  Population  Area (sq.
mi.)  Pop. Density (per sq. mi.)  \
0      False   False       False           False                       Fals
e
1      False   False       False           False                       Fals
e
2      False   False       False           False                       Fals
e
3      False   False       False           False                       Fals
e
4      False   False       False           False                       Fals
e
..       ...     ...         ...             ...
...
222    False   False       False           False                       Fals
e
223    False   False       False           False                       Fals
e
224    False   False       False           False                       Fals
e
225    False   False       False           False                       Fals
e
226    False   False       False           False                       Fals
e

     Coastline (coast/area ratio)  Net migration  \
0                           False          False
1                           False          False
2                           False          False
3                           False          False
4                           False          False
..                            ...            ...
222                         False          False
223                         False           True
224                         False          False
225                         False          False
226                         False          False

     Infant mortality (per 1000 births)  GDP ($ per capita)  Literacy (%)
\
0                                 False              False         False
1                                 False              False         False
2                                 False              False         False
3                                 False              False         False
4                                 False              False         False
..                                  ...                ...           ...
222                               False              False          True
223                                True               True          True
224                               False              False         False
225                               False              False         False
226                               False              False         False

     Phones (per 1000)  Arable (%)  Crops (%)  Other (%)  Climate  Birthrat
e  \
0                False       False      False      False    False      Fals
e
```

```
1               False       False      False      False    False      Fals
e
2               False       False      False      False    False      Fals
e
3               False       False      False      False    False      Fals
e
4               False       False      False      False    False      Fals
e
..                ...         ...        ...        ...      ...
...
222             False       False      False      False    False      Fals
e
223              True       False      False      False    False       Tru
e
224             False       False      False      False    False      Fals
e
225             False       False      False      False    False      Fals
e
226             False       False      False      False    False      Fals
e

     Deathrate  Agriculture  Industry  Service
0        False        False     False    False
1        False        False     False    False
2        False        False     False    False
3        False         True      True     True
4        False         True      True     True
..         ...          ...       ...      ...
222      False        False     False    False
223       True         True      True    False
224      False        False     False    False
225      False        False     False    False
226      False        False     False    False

[227 rows x 20 columns]>
```

Data Preparation - Filling Missing Values

In [10]:
```
# Replace commas with periods and convert to numeric
cols_to_convert = ['Pop. Density (per sq. mi.)', 'Coastline (coast/area rati
                   'Net migration', 'Infant mortality (per 1000 births)',
                   'Literacy (%)', 'Phones (per 1000)', 'Arable (%)', 'Crops
                   'Other (%)', 'Birthrate', 'Deathrate', 'Agriculture', 'Ir

for col in cols_to_convert:
    data[col] = data[col].str.replace(',', '.').astype(float)

# Now try the groupby operation again
data.groupby('Region')[['GDP ($ per capita)', 'Literacy (%)', 'Agriculture']
```

Out[10]:

Region	GDP ($ per capita)	Literacy (%)	Agriculture
ASIA (EX. NEAR EAST)	3450.0	90.60	0.1610
BALTICS	11400.0	99.80	0.0400
C.W. OF IND. STATES	3450.0	99.05	0.1980
EASTERN EUROPE	9100.0	98.60	0.0815
LATIN AMER. & CARIB	6300.0	94.05	0.0700
NEAR EAST	9250.0	83.00	0.0350
NORTHERN AFRICA	6000.0	70.00	0.1320
NORTHERN AMERICA	29800.0	97.50	0.0100
OCEANIA	5000.0	95.00	0.1505
SUB-SAHARAN AFRICA	1300.0	62.95	0.2760
WESTERN EUROPE	27200.0	99.00	0.0220

In [11]:
```
for col in data.columns.values:
    if data[col].isnull().sum() == 0:
        continue
    if col == 'Climate':
        guess_values = data.groupby('Region')['Climate'].apply(lambda x: x.m
    else:
        guess_values = data.groupby('Region')[col].median()
    for region in data['Region'].unique():
        data[col].loc[(data[col].isnull())&(data['Region']==region)] = guess
```

In [12]:
```
data.isnull().sum()
```

Out[12]:
```
Country                               0
Region                                0
Population                            0
Area (sq. mi.)                        0
Pop. Density (per sq. mi.)            0
Coastline (coast/area ratio)          0
Net migration                         0
Infant mortality (per 1000 births)    0
GDP ($ per capita)                    0
Literacy (%)                          0
Phones (per 1000)                     0
Arable (%)                            0
Crops (%)                             0
Other (%)                             0
Climate                               0
Birthrate                             0
Deathrate                             0
Agriculture                           0
Industry                              0
Service                               0
dtype: int64
```

In [13]:
```
data.isnull().sum() # Confirming All Missing Values are Filled
```

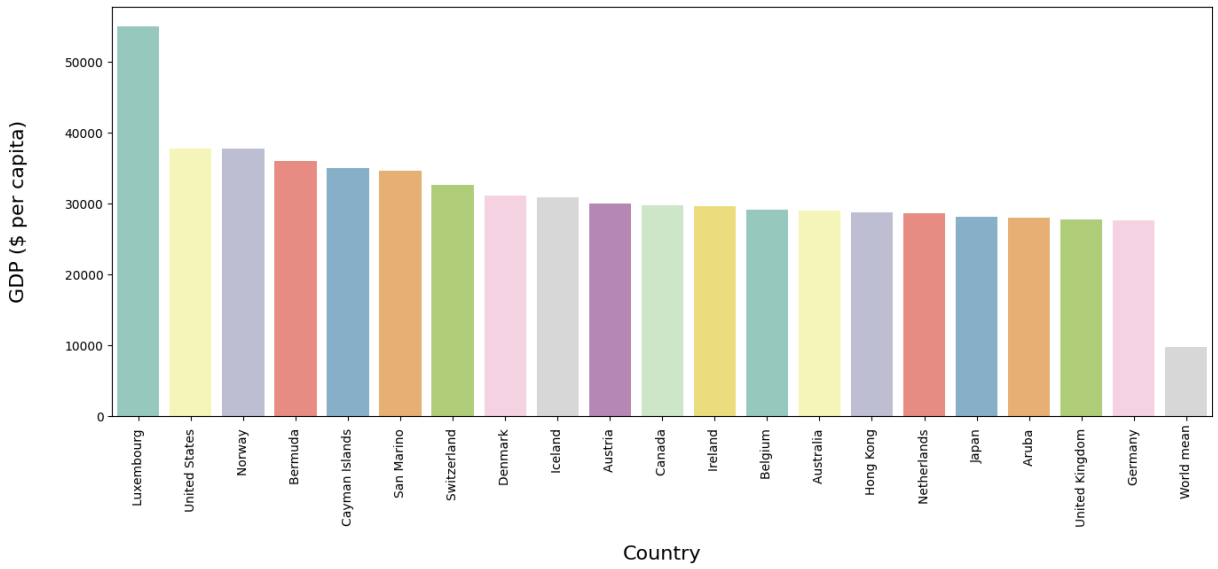
Out[13]:
```
Country                               0
Region                                0
Population                            0
Area (sq. mi.)                        0
Pop. Density (per sq. mi.)            0
Coastline (coast/area ratio)          0
Net migration                         0
Infant mortality (per 1000 births)    0
GDP ($ per capita)                    0
Literacy (%)                          0
Phones (per 1000)                     0
Arable (%)                            0
Crops (%)                             0
Other (%)                             0
Climate                               0
Birthrate                             0
Deathrate                             0
Agriculture                           0
Industry                              0
Service                               0
dtype: int64
```

Data Exploration

Top Countries with highest GDP per capita

In [14]:
```
fig, ax = plt.subplots(figsize=(16,6))
top_gdp_countries = data.sort_values('GDP ($ per capita)',ascending=False).h
mean = pd.DataFrame({'Country':['World mean'], 'GDP ($ per capita)':[data['G
gdps = pd.concat([top_gdp_countries[['Country','GDP ($ per capita)']],mean],
```

```
sns.barplot(x='Country',y='GDP ($ per capita)',data=gdps, palette='Set3')
ax.set_xlabel(ax.get_xlabel(),labelpad=15)
ax.set_ylabel(ax.get_ylabel(),labelpad=30)
ax.xaxis.label.set_fontsize(16)
ax.yaxis.label.set_fontsize(16)
plt.xticks(rotation=90)
plt.show()
```

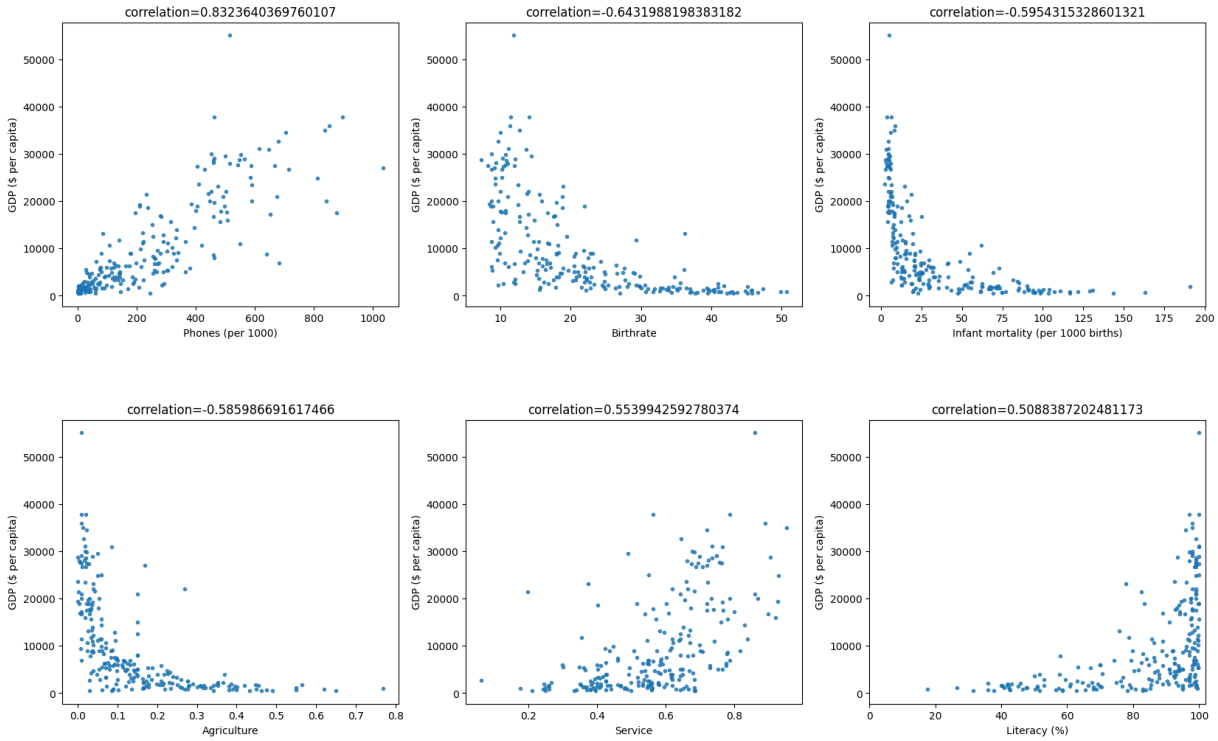


Top Factors affecting GDP per capita

In [15]:
```
fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(20,12))
plt.subplots_adjust(hspace=0.4)

corr_to_gdp = pd.Series()
for col in data.columns.values[2:]:
    if ((col!='GDP ($ per capita)')&(col!='Climate')):
        corr_to_gdp[col] = data['GDP ($ per capita)'].corr(data[col])
abs_corr_to_gdp = corr_to_gdp.abs().sort_values(ascending=False)
corr_to_gdp = corr_to_gdp.loc[abs_corr_to_gdp.index]

for i in range(2):
    for j in range(3):
        sns.regplot(x=corr_to_gdp.index.values[i*3+j], y='GDP ($ per capita)
                    ax=axes[i,j], fit_reg=False, marker='.')
        title = 'correlation='+str(corr_to_gdp[i*3+j])
        axes[i,j].set_title(title)
axes[1,2].set_xlim(0,102)
plt.show()
```



Countries with low Birthrate and low GDP per capita

In [16]:
```
data.loc[(data['Birthrate']<14)&(data['GDP ($ per capita)']<10000)]
```

Out[16]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infant mort... bi...
9	Armenia	C.W. OF IND. STATES	2976372	29800	99.9	0.00	-6.47	2
18	Belarus	C.W. OF IND. STATES	10293011	207600	49.6	0.00	2.54	1
25	Bosnia & Herzegovina	EASTERN EUROPE	4498976	51129	88.0	0.04	0.31	1
30	Bulgaria	EASTERN EUROPE	7385367	110910	66.6	0.32	-4.58	2
42	China	ASIA (EX. NEAR EAST)	1313973713	9596960	136.9	0.15	-0.40	1
51	Cuba	LATIN AMER. & CARIB	11382820	110860	102.7	3.37	-1.58	
75	Georgia	C.W. OF IND. STATES	4661473	69700	66.9	0.44	-4.70	1
123	Macedonia	EASTERN EUROPE	2050554	25333	80.9	0.00	-1.45	1
168	Romania	EASTERN EUROPE	22303552	237500	93.9	0.09	-0.13	2
169	Russia	C.W. OF IND. STATES	142893540	17075200	8.4	0.22	1.02	1
171	Saint Helena	SUB-SAHARAN AFRICA	7502	413	18.2	14.53	0.00	1
174	St Pierre & Miquelon	NORTHERN AMERICA	7026	242	29.0	49.59	-4.86	
181	Serbia	EASTERN EUROPE	9396411	88361	106.3	0.00	-1.33	1
201	Thailand	ASIA (EX. NEAR EAST)	64631595	514000	125.7	0.63	0.00	2
204	Trinidad & Tobago	LATIN AMER. & CARIB	1065842	5128	207.9	7.06	-10.83	1
211	Ukraine	C.W. OF IND.	46710816	603700	77.4	0.46	-0.39	2

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Ir mort bi
		STATES						

Modeling

Training and Testing

In [17]:
```
LE = LabelEncoder()
data['Region_label'] = LE.fit_transform(data['Region'])
data['Climate_label'] = LE.fit_transform(data['Climate'])
data.sample()
```

Out[17]:

	Country	Region	Population	Area (sq. mi.)	Pop. Density (per sq. mi.)	Coastline (coast/area ratio)	Net migration	Infant mortality (per 1000 births)	GDP (pe capita
41	Chile	LATIN AMER. & CARIB	16134219	756950	21.3	0.85	0.0	8.8	9900.

1 rows × 22 columns

In [18]:
```
train, test = train_test_split(data, test_size=0.3, shuffle=True)
training_features = ['Population', 'Area (sq. mi.)',
       'Pop. Density (per sq. mi.)', 'Coastline (coast/area ratio)',
       'Net migration', 'Infant mortality (per 1000 births)',
       'Literacy (%)', 'Phones (per 1000)',
       'Arable (%)', 'Crops (%)', 'Other (%)', 'Birthrate',
       'Deathrate', 'Agriculture', 'Industry', 'Service', 'Region_label',
       'Climate_label','Service']
train_X = train[training_features]
train_Y = train['GDP ($ per capita)']
test_X = test[training_features]
test_Y = test['GDP ($ per capita)']
```

In [19]:
```
model1 = LinearRegression()
model1.fit(train_X, train_Y)
train_pred_Y = model1.predict(train_X)
test_pred_Y = model1.predict(test_X)
train_pred_Y = pd.Series(train_pred_Y.clip(0, train_pred_Y.max()), index=tra
test_pred_Y = pd.Series(test_pred_Y.clip(0, test_pred_Y.max()), index=test_Y

rmse_train = np.sqrt(mean_squared_error(train_pred_Y, train_Y))
```

```
msle_train = mean_squared_log_error(train_pred_Y, train_Y)
rmse_test = np.sqrt(mean_squared_error(test_pred_Y, test_Y))
msle_test = mean_squared_log_error(test_pred_Y, test_Y)

print('rmse_train:',rmse_train,'msle_train:',msle_train)
print('rmse_test:',rmse_test,'msle_test:',msle_test)
```

```
rmse_train: 4549.818294037232 msle_train: 4.8419048234242945
rmse_test: 5241.662851209217 msle_test: 7.360124480705821
```

In [20]:
```
model2 = RandomForestRegressor(n_estimators = 50,
                               max_depth = 6,
                               min_weight_fraction_leaf = 0.05,
                               max_features = 0.8,
                               random_state = 42)
model2.fit(train_X, train_Y)
train_pred_Y = model2.predict(train_X)
test_pred_Y = model2.predict(test_X)
train_pred_Y = pd.Series(train_pred_Y.clip(0, train_pred_Y.max()), index=tra
test_pred_Y = pd.Series(test_pred_Y.clip(0, test_pred_Y.max()), index=test_Y

rmse_train = np.sqrt(mean_squared_error(train_pred_Y, train_Y))
msle_train = mean_squared_log_error(train_pred_Y, train_Y)
rmse_test = np.sqrt(mean_squared_error(test_pred_Y, test_Y))
msle_test = mean_squared_log_error(test_pred_Y, test_Y)

print('rmse_train:',rmse_train,'msle_train:',msle_train)
print('rmse_test:',rmse_test,'msle_test:',msle_test)
```

```
rmse_train: 2961.008564233517 msle_train: 0.1653562494625042
rmse_test: 4627.463993433253 msle_test: 0.28797532498173684
```

Visualization of Results

In [21]:
```
plt.figure(figsize=(18,12))

train_test_Y = pd.concat([train_Y, test_Y])
train_test_pred_Y = pd.concat([train_pred_Y, test_pred_Y])

data_shuffled = data.loc[train_test_Y.index]
label = data_shuffled['Country']

colors = {'ASIA (EX. NEAR EAST)         ':'red',
          'EASTERN EUROPE                     ':'orange',
          'NORTHERN AFRICA                    ':'gold',
          'OCEANIA                            ':'green',
          'WESTERN EUROPE                     ':'blue',
          'SUB-SAHARAN AFRICA                 ':'purple',
          'LATIN AMER. & CARIB    ':'olive',
          'C.W. OF IND. STATES ':'cyan',
          'NEAR EAST                          ':'hotpink',
          'NORTHERN AMERICA                   ':'lightseagreen',
          'BALTICS                            ':'rosybrown'}

for region, color in colors.items():
    X = train_test_Y.loc[data_shuffled['Region']==region]
```

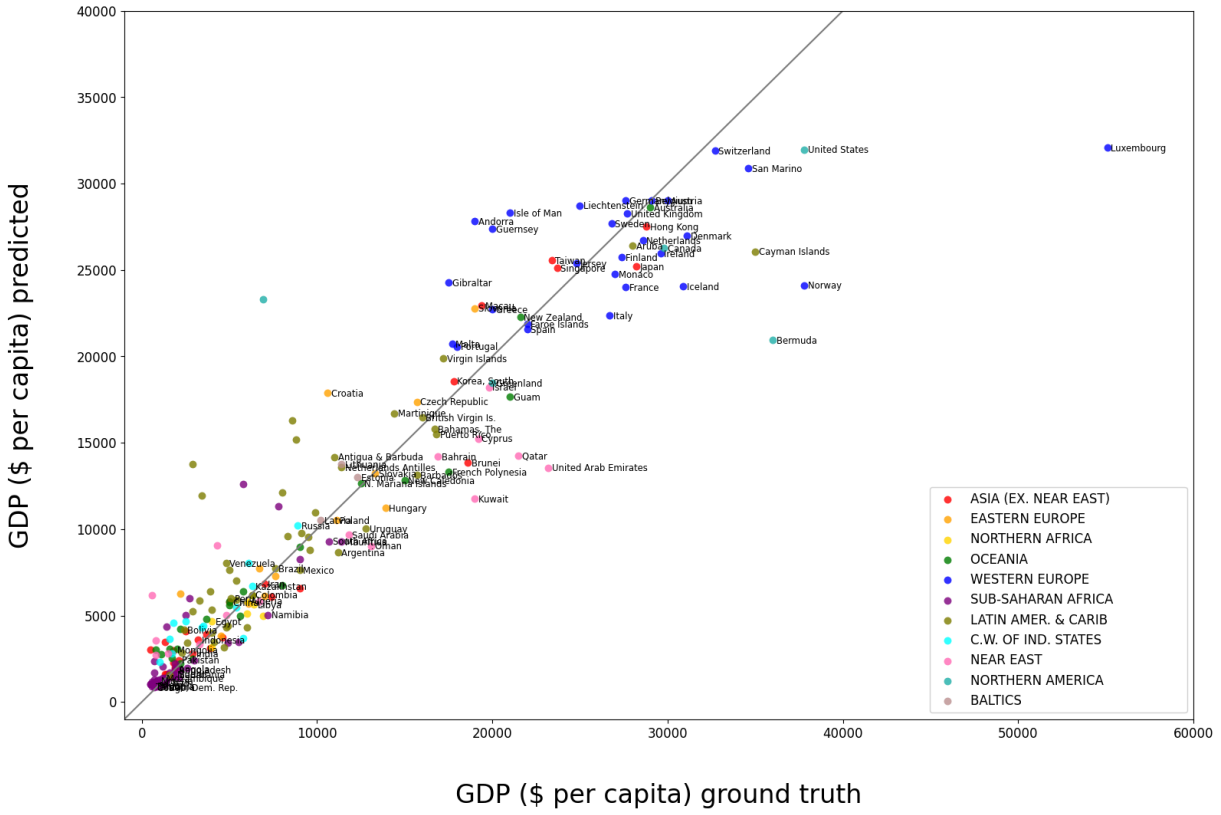
```
    Y = train_test_pred_Y.loc[data_shuffled['Region']==region]
    ax = sns.regplot(x=X, y=Y, marker='.', fit_reg=False, color=color, scatt
plt.legend(loc=4,prop={'size': 12})

ax.set_xlabel('GDP ($ per capita) ground truth',labelpad=40)
ax.set_ylabel('GDP ($ per capita) predicted',labelpad=40)
ax.xaxis.label.set_fontsize(24)
ax.yaxis.label.set_fontsize(24)
ax.tick_params(labelsize=12)

x = np.linspace(-1000,50000,100) # 100 linearly spaced numbers
y = x
plt.plot(x,y,c='gray')

plt.xlim(-1000,60000)
plt.ylim(-1000,40000)

for i in range(0,train_test_Y.shape[0]):
    if((data_shuffled['Area (sq. mi.)'].iloc[i]>8e5) |
       (data_shuffled['Population'].iloc[i]>1e8) |
       (data_shuffled['GDP ($ per capita)'].iloc[i]>10000)):
        plt.text(train_test_Y.iloc[i]+200, train_test_pred_Y.iloc[i]-200, la
```



Total GDP

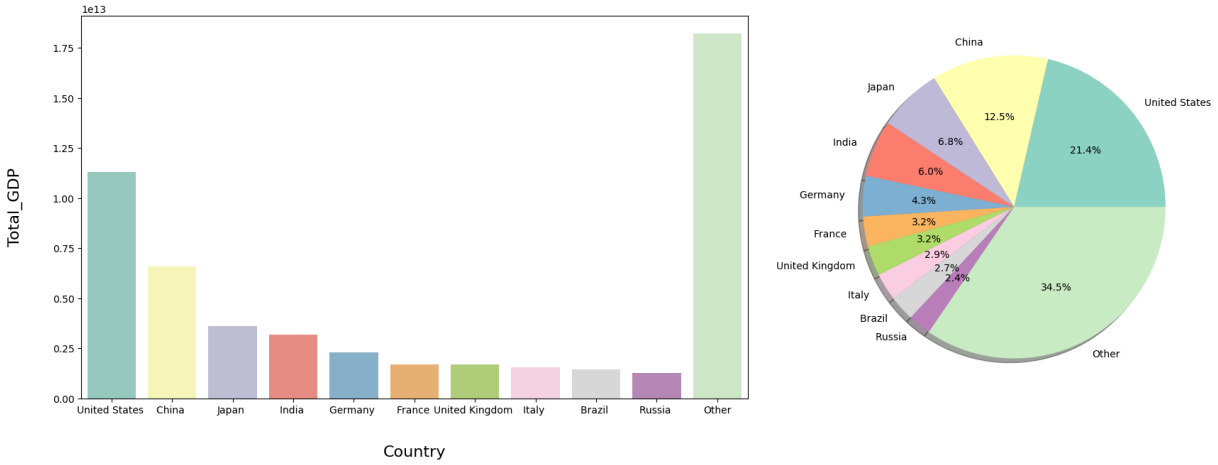
Top Countries

In [22]:
```
data['Total_GDP ($)'] = data['GDP ($ per capita)'] * data['Population']
top_gdp_countries = data.sort_values('Total_GDP ($)',ascending=False).head(1
```

```
other = pd.DataFrame({'Country':['Other'], 'Total_GDP ($)':[data['Total_GDP
gdps = pd.concat([top_gdp_countries[['Country','Total_GDP ($)']],other],ignc

fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(20,7),gridspec_kw = {'wi
sns.barplot(x='Country',y='Total_GDP ($)',data=gdps,ax=axes[0],palette='Set3
axes[0].set_xlabel('Country',labelpad=30,fontsize=16)
axes[0].set_ylabel('Total_GDP',labelpad=30,fontsize=16)

colors = sns.color_palette("Set3", gdps.shape[0]).as_hex()
axes[1].pie(gdps['Total_GDP ($)'], labels=gdps['Country'],colors=colors,autc
axes[1].axis('equal')
plt.show()
```



In [23]:
```
Rank1 = data[['Country','Total_GDP ($)']].sort_values('Total_GDP ($)', ascer
Rank2 = data[['Country','GDP ($ per capita)']].sort_values('GDP ($ per capit
Rank1 = pd.Series(Rank1.index.values+1, index=Rank1.Country)
Rank2 = pd.Series(Rank2.index.values+1, index=Rank2.Country)
Rank_change = (Rank2-Rank1).sort_values(ascending=False)
print('Rank of total GDP - Rank of GDP per capita:')
Rank_change.loc[top_gdp_countries.Country]
```

```
Rank of total GDP - Rank of GDP per capita:
```

Out[23]:
```
Country
United States       1
China             118
Japan              14
India             146
Germany            15
France             15
United Kingdom     12
Italy              17
Brazil             84
Russia             75
dtype: int64
```

Factors affecting Total GDP

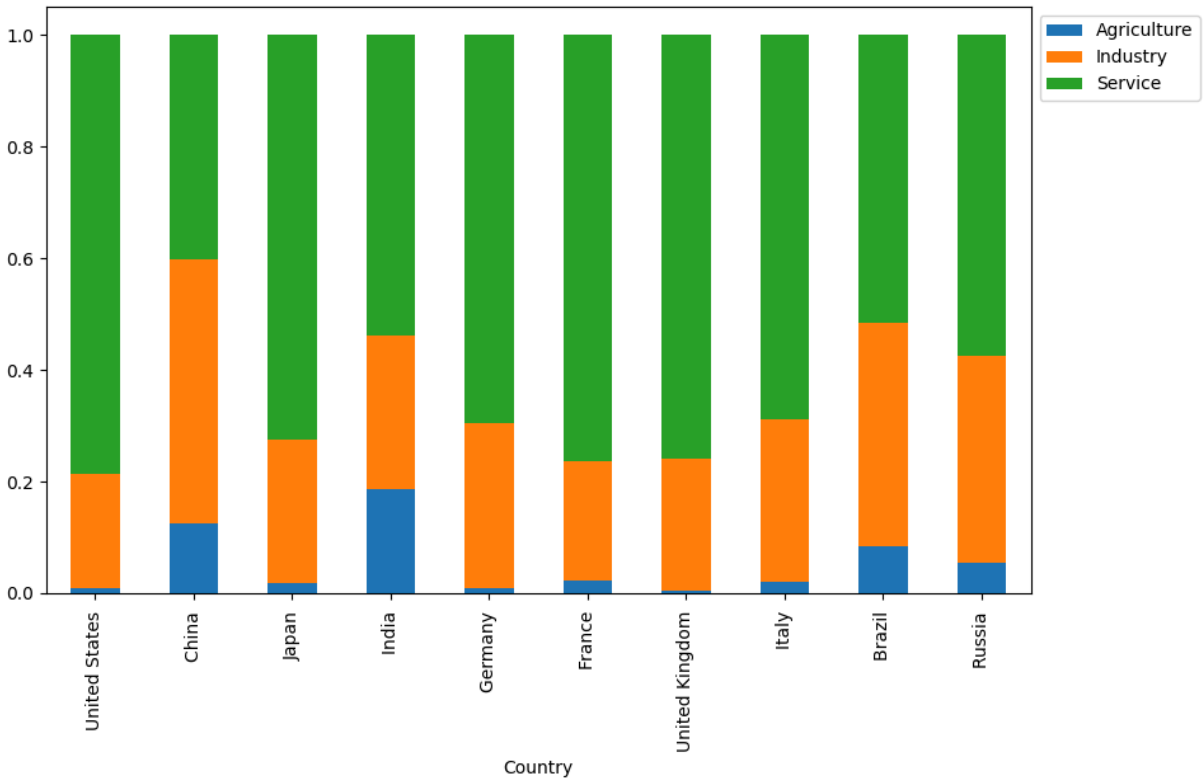
In [24]:
```
corr_to_gdp = pd.Series()
for col in data.columns.values[2:]:
    if ((col!='Total_GDP ($)')&(col!='Climate')&(col!='GDP ($ per capita)'))
        corr_to_gdp[col] = data['Total_GDP ($)'].corr(data[col])
```

```
abs_corr_to_gdp = corr_to_gdp.abs().sort_values(ascending=False)
corr_to_gdp = corr_to_gdp.loc[abs_corr_to_gdp.index]
print(corr_to_gdp)
```

```
Population                             0.639528
Area (sq. mi.)                         0.556396
Phones (per 1000)                      0.233484
Birthrate                             -0.166889
Agriculture                           -0.139516
Arable (%)                             0.129928
Climate_label                          0.125791
Infant mortality (per 1000 births)    -0.122076
Literacy (%)                           0.099417
Service                                0.085096
Region_label                          -0.079745
Crops (%)                             -0.077078
Coastline (coast/area ratio)          -0.065211
Other (%)                             -0.064882
Net migration                          0.054632
Industry                               0.050399
Deathrate                             -0.035820
Pop. Density (per sq. mi.)            -0.028487
dtype: float64
```

Comparison of the Top 10

In [25]:
```
plot_data = top_gdp_countries.head(10)[['Country','Agriculture', 'Industry',
plot_data = plot_data.set_index('Country')
ax = plot_data.plot.bar(stacked=True,figsize=(10,6))
ax.legend(bbox_to_anchor=(1, 1))
plt.show()
```



Land Usage

In [26]:
```
plot_data = top_gdp_countries[['Country','Arable (%)', 'Crops (%)', 'Other (
plot_data = plot_data.set_index('Country')
ax = plot_data.plot.bar(stacked=True,figsize=(10,6))
ax.legend(bbox_to_anchor=(1, 1))
plt.show()
```