Homework1-submit

September 16, 2014

1 CSE 591 Homework1

Xiufeng Yang 107992992

```
In [2]: %install_ext http://raw.github.com/jrjohansson/version_information/master/version_information.py
```

```
Installed version_information.py. To use it, type:
  %load_ext version_information
```

```
In [1]: #%load_ext version_information
        #%version_information numpy, scipy, matplotlib, sympy, scikit_learn, nltk, pandas
```

1.1 Plots

1.1.1 Capital Map

```
In [142]: import numpy as np
          import pandas
          from __future__ import print_function
          %matplotlib inline
          import matplotlib.pyplot as plt
          from mpl_toolkits.basemap import Basemap
```

```
In [193]: np_array = pandas.io.parsers.read_csv("country-data-linux.csv").as_matrix()
```

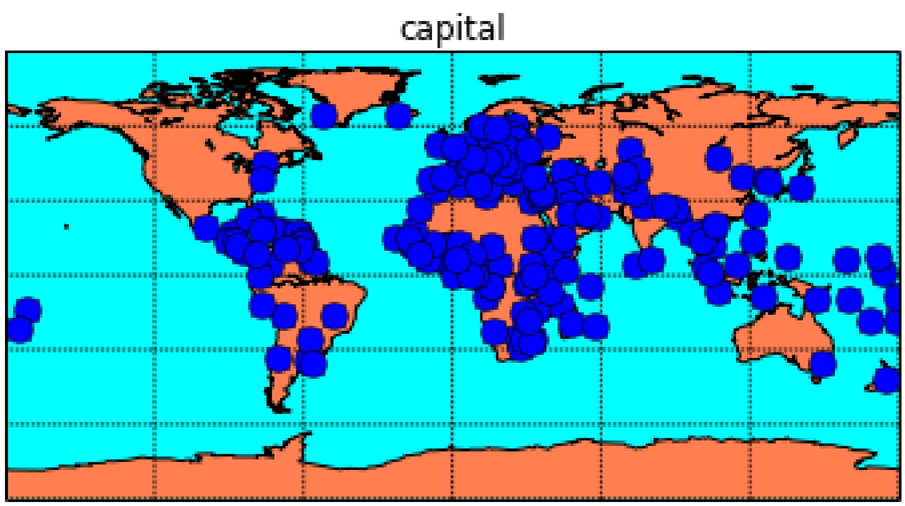
```
In [194]: print(np_array)
```

```
[['Afghanistan' 'AF' 'Islamic republic' ..., '7512000' '1000000' '15']
 ['Albania' 'AL' 'parliamentary democracy' ..., '1098000' '1300000' '9.52']
 ['Algeria' 'DZ' 'republic' ..., '11150000' '4700000' '8']
 ...,
 ['Yemen' 'YE' 'republic' ..., '7100000' '2349000' '22']
 ['Zambia' 'ZM' 'republic' ..., '6275000' '816200' '10.4']
 ['Zimbabwe' 'ZW' 'parliamentary democracy' ..., '3939000' '1423000' '28']]
```

```
In [195]: def conversion(old):
              new = old
              direction = {'N':1, 'S':-1, 'E': 1, 'W':-1}
              #new = old.replace(u'°',' ').replace('\'',' ').replace('"',' ')
              new = new.split()
              new_dir = new.pop()
              new.extend([0,0,0])
              return (int(new[0])+int(new[1])/60.0+int(new[2])/3600.0) * direction[new_dir]

          for x in np_array:
```

```
        try:
            x[5]=conversion(x[5])
            x[6]=conversion(x[6])
        except AttributeError:
            x[5]=0
            x[6]=0
    #print(np_array[:,5])
    #print(np_array[:,6])
```

In [196]:
```
m = Basemap(projection='cyl',llcrnrlat=-90,urcrnrlat=90,\
            llcrnrlon=-180,urcrnrlon=180,resolution='c')
m.drawcoastlines()
m.fillcontinents(color='coral',lake_color='aqua')
m.drawparallels(np.arange(-90.,91.,30.))
m.drawmeridians(np.arange(-180.,181.,60.))
m.drawmapboundary(fill_color='aqua')
plt.title("capital")
for x in np_array:
    try:
        x, y = m(x[6], x[5])
        m.plot(x, y, 'bo', markersize=10)
    except:
        continue
plt.show()
```

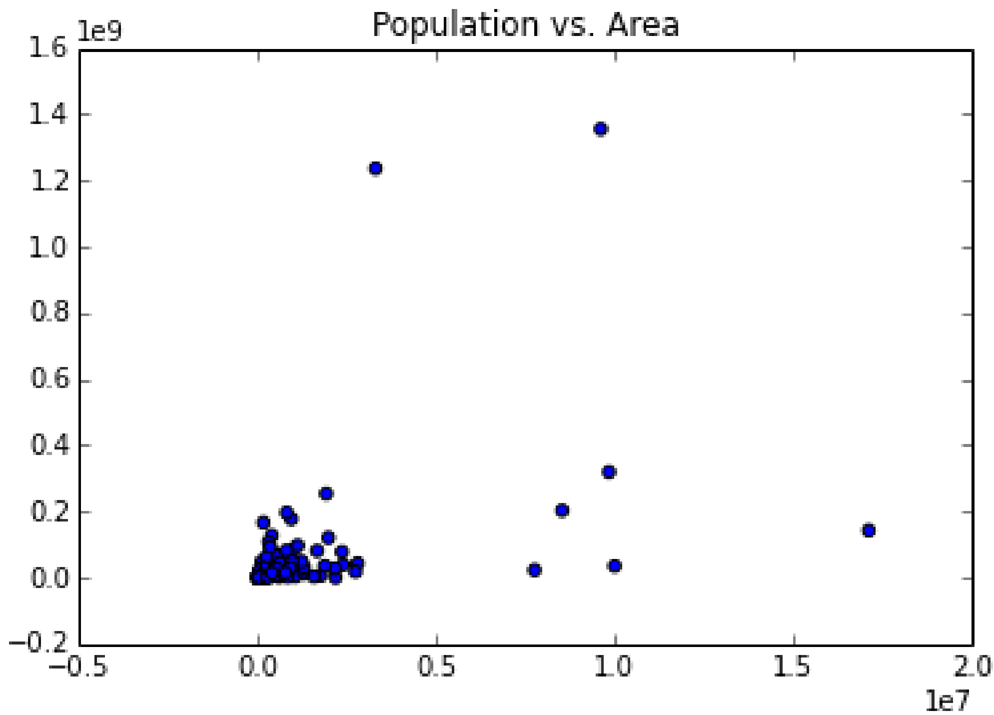


Conclusion: This datamap maps all capital cities on the world map

1.1.2 Population vs. Area

In [11]:
```
fig, ax = plt.subplots()

ax.scatter(np_array[:,10], np_array[:,7])
ax.set_title("Population vs. Area")
```

Out[11]: <matplotlib.text.Text at 0x7f6420cc0650>



Conclusion The x-axis is area, y-axis is population, most countries stays in a relatively compact area, while only a few stand out. Specifically, two countries have very large population compared with their areas, and 5 countries have very low populations compared with their area.

1.1.3 Government Type Distribution

```
In [12]: print(np_array[80,2])
```

```
republic; parliamentary democracy
```

```
In [13]: for x in np.nditer(np_array[:,2], op_flags=['readwrite'], flags=['refs_ok']):
             #print(type(x))
             if x=="Islamic republic":
                 x[...]=1
             elif x=="parliamentary democracy":
                 x[...]=2
             elif x=='republic':
                 x[...]=3
             elif x=='constitutional monarchy':
                 x[...]=4
             elif x=='federal parliamentary democracy':
                 x[...]=5
             elif x=='federal republic':
                 x[...]=6
             elif x=='constitutional parliamentary democracy':
                 x[...]=7
             elif x=='parliamentary republic':
```

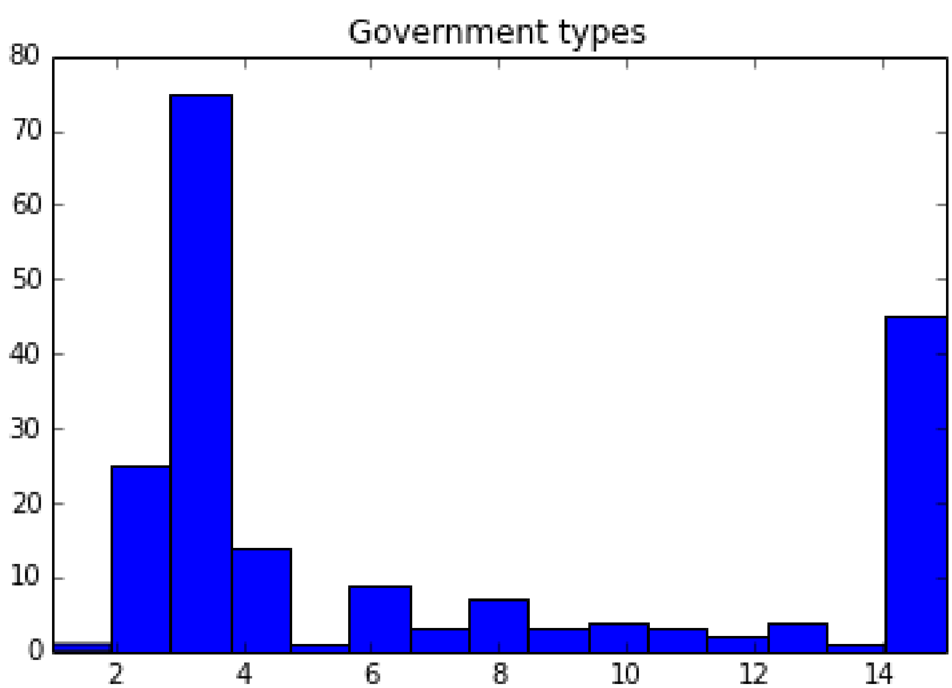
```
            x[...]=8
        elif x=='constitutional government':
            x[...]=9
        elif x=='constitutional republic':
            x[...]=10
        elif x=='monarchy':
            x[...]=11
        elif x=='democratic republic':
            x[...]=12
        elif x=='Communist state':
            x[...]=13
        elif x=='federation':
            x[...]=14
        else:
            x[...]=15
```

```
In [14]: print(np_array[:,2])
```

```
[1 2 3 2 3 4 3 3 5 6 3 7 4 2 2 3 15 2 3 4 3 15 8 6 15 2 8 3 3 15 3 15 3 3 3
 13 3 3 3 3 12 3 2 13 15 2 4 3 2 12 3 3 3 3 15 8 6 3 3 3 3 3 3 6 15 8 15 2
 15 3 3 3 3 15 2 10 6 3 15 2 15 2 3 7 15 4 15 3 3 15 3 3 15 3 13 2 3 15 3
 15 15 2 4 2 3 15 15 3 3 3 9 15 2 6 9 3 4 15 3 4 3 15 3 3 15 4 15 3 3 6 4
 11 6 9 15 7 10 10 3 3 15 15 3 14 3 2 2 15 2 3 3 11 3 3 3 15 8 2 8 2 15 3 3
 15 3 15 15 11 4 15 15 15 3 3 4 3 15 4 2 3 15 15 2 3 3 15 4 15 10 3 8 6 13
 15 3 3 2]
```

```
In [15]: fig, ax1 = plt.subplots()
         ax1.hist(np_array[:,2], bins=15)
         ax1.set_title("Government types")
         ax1.set_xlim(min(np_array[:,2]), max(np_array[:,2]))
```

```
Out[15]: (1, 15)
```



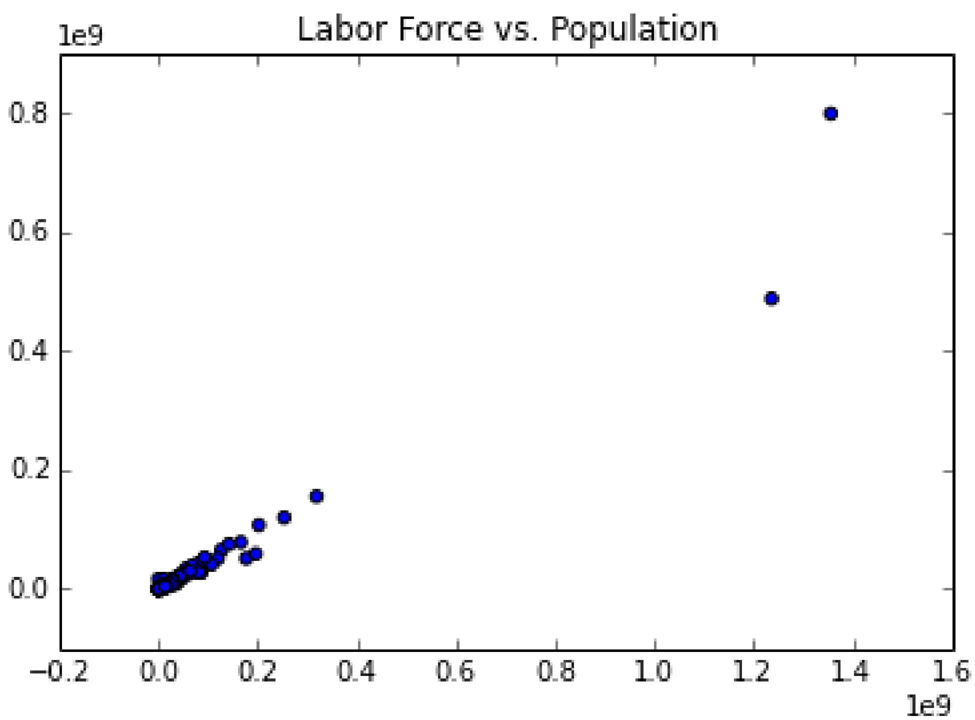
Conclusion For plot legend, please refer to above python code. From this plot, we can see that most countries uses government type is republic, and then parliamentary democracy

1.1.4 Labor Force vs. Population

In [16]:
```
totalLaborforce = 0
countriesCounted = 0;
for x in np.nditer(np_array[:,18], op_flags=['readwrite'], flags=['refs_ok']):
    try:
        if int(str(x)) > 0:
            totalLaborforce+=int(str(x))
            countriesCounted=countriesCounted+1
    except ValueError:
        x[...]=-1

    #print(x)

mean = totalLaborforce//countriesCounted
#print(mean)

for x in np.nditer(np_array[:,18], op_flags=['readwrite'], flags=['refs_ok']):
    if int(str(x)) == -1:
        x[...] = mean
```

In [17]:
```
fig, ax2 = plt.subplots()

ax2.scatter(np_array[:,7], np_array[:,18])
ax2.set_title("Labor Force vs. Population")
```

Out[17]:
```
<matplotlib.text.Text at 0x7f6420b049d0>
```

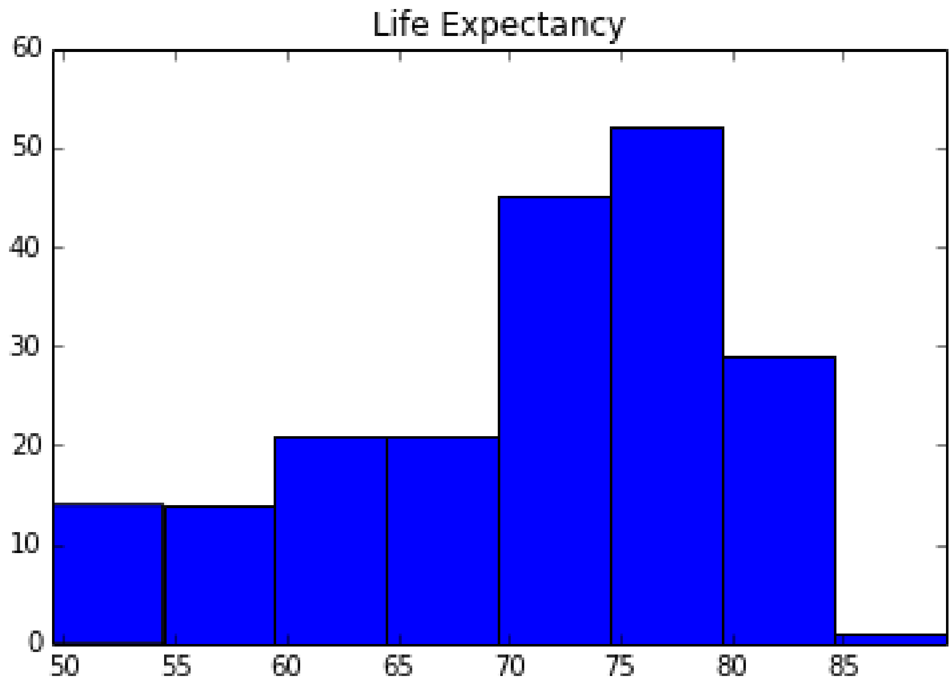


Conclusion In this plot, we can see that the labor forces of each countries have an almost linear relationship with population. The two dots that are standing out are calculated by the values got from Mean value imputation. This implies that mean value imputation is not appropriate in this situation. Based on the pattern of other countries, imputation by interpolation is a much better way to fill in the missing values in this kind of scenario.

1.1.5 Life Expectancy Distribution

```
In [18]: fig, ax3 = plt.subplots()
         ax3.hist(np_array[:,8], bins=8)
         ax3.set_title("Life Expectancy")
         ax3.set_xlim(min(np_array[:,8]), max(np_array[:,8]))
```

```
Out[18]: (49.44, 89.57)
```



In this plot we can see the distribution of life expectancy of all countries. The most countries has life expectancy of 75-80, following by 70-75. Very few countries has life expectancy of 85-90. No countries have it over 90 yet.

1.2 Pairwise correlation

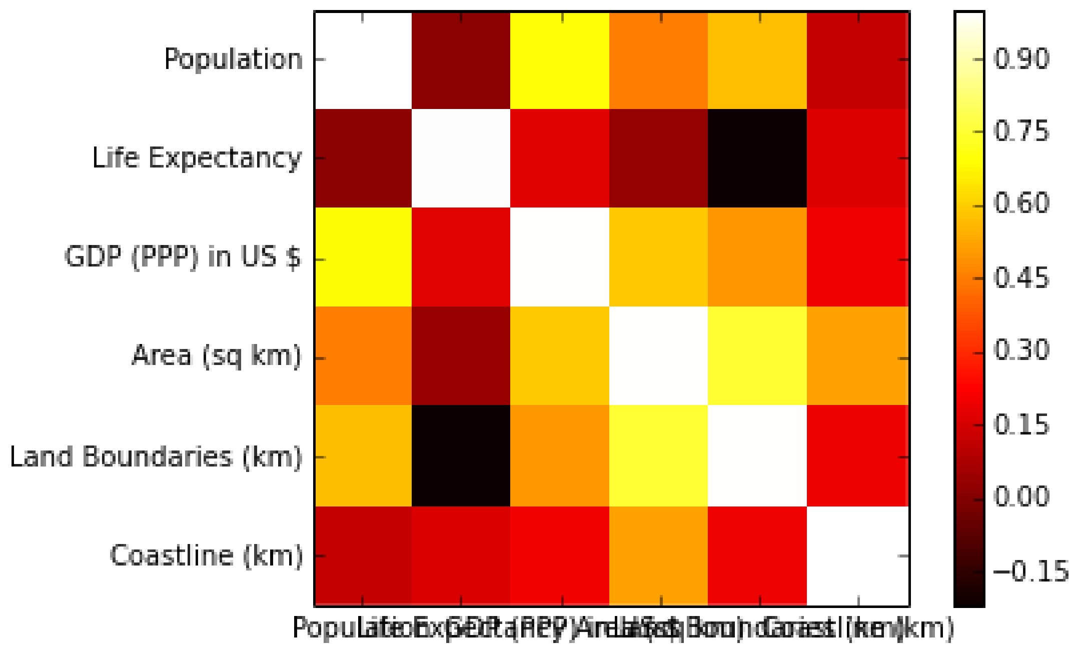
Existing problem: The code above can only deal with columns that don't have missing values, but the funcion is working perfectly. Just need to calculate the missing values (done some columns in the following sections) and fill in the file, re-run the code, it will give all the permutations.

```
In [114]: pd_array = pandas.io.parsers.read_csv("country-data-linux.csv")
          #print(pd_array)
          corr=pd_array.corr()
          print(corr)
          plt.imshow(corr, cmap='hot', interpolation='none')
          plt.colorbar()
          plt.xticks(range(len(corr)), corr.columns)
          plt.yticks(range(len(corr)), corr.columns)
```

```
Population  Life Expectancy  GDP (PPP) in US $  \
Population               1.000000         0.014249           0.697280
Life Expectancy          0.014249         1.000000           0.175071
GDP (PPP) in US $        0.697280         0.175071           1.000000
Area (sq km)             0.453228         0.033022           0.592445
Land Boundaries (km)     0.575146        -0.219494           0.500468
Coastline (km)           0.120474         0.162933           0.204414

                     Area (sq km)  Land Boundaries (km)  Coastline (km)
Population               0.453228              0.575146        0.120474
Life Expectancy          0.033022             -0.219494        0.162933
```

```
GDP (PPP) in US $         0.592445              0.500468        0.204414
Area (sq km)              1.000000              0.749098        0.521336
Land Boundaries (km)      0.749098              1.000000        0.195977
Coastline (km)            0.521336              0.195977        1.000000

[6 rows x 6 columns]
```

```
Out[114]: ([<matplotlib.axis.YTick at 0x7f99eac25790>,
            <matplotlib.axis.YTick at 0x7f99eadd3750>,
            <matplotlib.axis.YTick at 0x7f99eac6a190>,
            <matplotlib.axis.YTick at 0x7f99eac6a890>,
            <matplotlib.axis.YTick at 0x7f99eadb1e10>,
            <matplotlib.axis.YTick at 0x7f99eadb16d0>],
           <a list of 6 Text yticklabel objects>)
```



1.2.1 Conclusion:

From the plot, I can conclude that for the existing columns, most significantly correlated is Land Boundaries and Area; least significantly correlated is Land Boundaries and Life Expectancy. The result makes perfect sense because area and land boundaries have strong relationship, while land boundaries and life expectancy don't necessarily have relationship.

1.3 Linear regression of GDP per capita

I think the factor that will most likely have a linear relationship with GDP per capita is the ratios of (internet users / population). Because modern technologies like the Internet has a big impact on the economy of a country, and the development of economy will also have impact on the development of the Internet.

Because GDP per capita is GDP / population, and internet user rate is internet users / population, so the problem can be simplified by compare GDP and internet users.

First step is to fill in the unknown values in the internet users column here I use linear regression, first calculate the average ratio of all countries that have valid values, and fill in the missing values using the calculated ratio and the existing population data. The whole process is below:

```
In [34]: #print(np_array[:,19])
```

```
In [116]: #np_array = pandas.io.parsers.read_csv("country-data-linux.csv").as_matrix()
          totalInternetUsers = 0
          totalGDP = 0
          for x in np_array:
              try:
                  #print()
                  #print(int(x[19]))
                  totalGDP = totalGDP+int(x[9])
                  #print(totalPopulation)
                  totalInternetUsers = totalInternetUsers+int(x[19])
                  #print(totalInternetUsers)
              except ValueError:
                  #print(x[19])
                  x[19] = -1
                  #print(x[19])
```

```
In [117]: from __future__ import division
          print(totalInternetUsers)
          print(totalGDP)
          avgRatio = totalInternetUsers/totalGDP
          print(avgRatio)
```

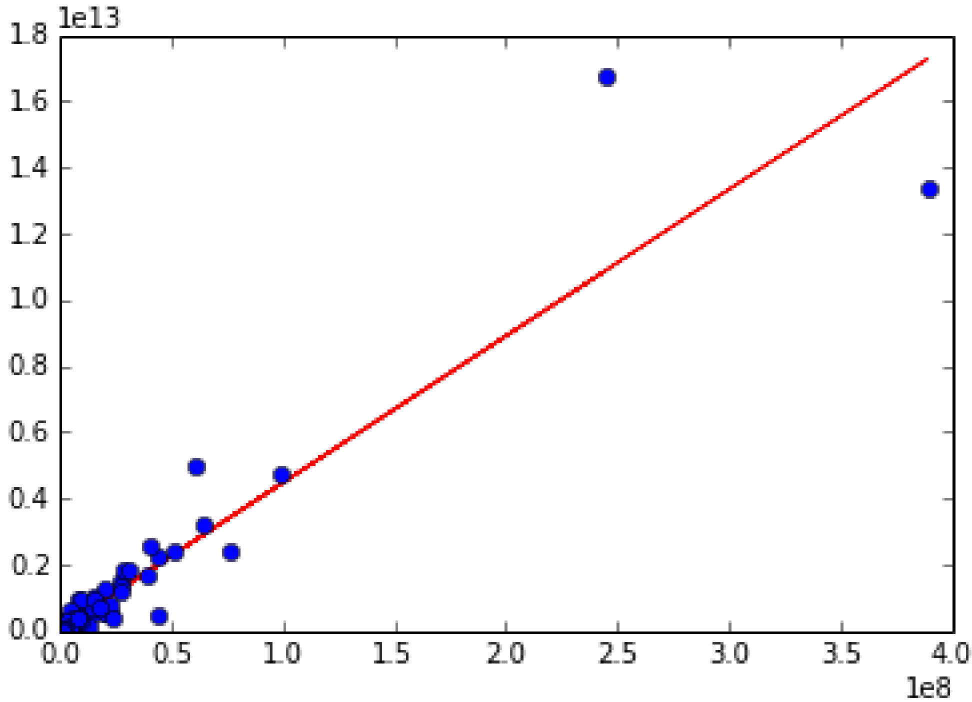
```
1805576800
86690158000000
2.08279329702e-05
```

```
In [118]: for x in np_array:
              if x[19]==-1:
                  x[19] = int(x[9]*avgRatio)
                  print(x[19])
```

```
833117
293882
1249
5113
306378
1873889
833
18880
```

```
In [119]: for x in np_array:
              x[19] = int(x[19])

          #A = array([ xi, ones(9)])
          InternetUsers = array([np_array[:,19], ones(len(np_array[:,19]))])
          GDP = np_array[:,9]

          #print(InternetUsers)
          w = linalg.lstsq(InternetUsers.T, GDP)[0]
```

```
In [82]: line = w[0]*np_array[:,19]+w[1] # regression line
         plot(np_array[:,19],line,'r-',np_array[:,19],np_array[:,9],'o')
         show()
```



1.3.1 Conclusion

From the plot we can see that most countries has similar linear relationship between internet users/population and average income (GDP/population). The two countries that are clearly standing out are China and United States. China has a relatively low average income compared to my prediction, while the United States has a relatively higher average income compared to internet users per population. The Reason is China has a huge population, which lower it's average income. United State is point 0 of modern Internet industry, the Internet is highly developed, it's not the Internet user rate is low, it's the average income of the United State is high.

1.4 Ranking

For social welfare, I use 4 values to evaluate: GDP/population(average income), life expectancy, literacy rate, and health expenditure, because average income is the basis of providing social welfare, without money, there will be no social welfare. Similarly, a country that provides good social welfare often has a higher life expectancy and literacy rate because medical and education are very importatnt part of social welfare.

define: GDP/population: ai; Life expectancy: le; literacy rate: lr; health expenditure: h;

The equation I come up with is: index = (ai) X (le) X (lr) X (h)

The first task is still filling in the missing values. Because literacy rate and health expenditure have no very clear patter with other values, so I just fill in the mean value.

```
In [99]: np_array = pandas.io.parsers.read_csv("country-data-linux.csv").as_matrix()
         totalRate = 0.0
         countriesCounted = 0
```

```
for x in np_array:
    try:
        totalRate+=float(x[13])
        countriesCounted=countriesCounted + 1
    except ValueError:
        x[13]=-1.0
lrMean = totalRate/countriesCounted

for x in np_array:
    if x[13]==-1.0:
        x[13]=float(lrMean)
    x[13]=float(x[13])
#print(np_array[:,13])
```

In [102]:
```
totalExp = 0.0
countriesCounted = 0
for x in np_array:
    try:
        totalExp+=float(x[14])
        countriesCounted=countriesCounted + 1
    except ValueError:
        x[14]=-1.0
heMean = totalExp/countriesCounted
#print(heMean)
for x in np_array:
    if x[14]==-1.0:
        x[14]=float(heMean)
    x[14]=float(x[14])
#print(np_array[:,14])
```

In [109]:
```
index = []
for x in np_array:
    index.append(x[9]/x[7]*x[8]*x[13]*x[14])
print("The country that has the highest social welfare index is:")
print(np_array[:,0][index.index(max(index))])
print("The country that has the lowest social welfare index is:")
print(np_array[:,0][index.index(min(index))])
```

```
The country that has the highest social welfare index is:
Monaco
The country that has the lowest social welfare index is:
Niger
```

1.4.1 Conclusion:

By my measure, Monaco does the best in social welfare, while Niger does the worst. This result makes sense because it is commonly known that Monaco is a rich country and provides high social welfare, while Niger is a poor African country, clearly is lacks the resource and wealth to provide its citizens a good social welfare.

1.5 Similarity

For the similarities and differences, I measure from two aspects: natural difference and social difference.

For natual difference, measure Area, land boundaries, and coastline. For social difference, measure population, life expectancy, GDP, literacy rate, health expenditure, internet users.

I'll measure ####the percentage of each value above of each country among all countries, and add them up to create a ranking index. Then by comparing the index, I can conclude which two countries are most apart and which two countries are most close.

I will start by calculating the total values of each column:

```
In [120]: totalArea = 0
          totalLandBoundaries = 0
          totalCoastline = 0
          totalPopulation = 0
          totalLifeExpectancy = 0
          totalGDP = 0
          totalLiteracyRate = 0
          totalHealthExpenditure = 0
          totalInternetUsers = 0

          for x in np_array:
              totalArea = totalArea + x[10]
              totalLandBoundaries = totalLandBoundaries + x[11]
              totalCoastline = totalCoastline + x[12]
              totalPopulation = totalPopulation + x[7]
              totalLifeExpectancy = totalLifeExpectancy + x[8]
              totalGDP = totalGDP + x[9]
              totalLiteracyRate = totalLiteracyRate + x[13]
              totalHealthExpenditure = totalHealthExpenditure + x[14]
              totalInternetUsers = totalInternetUsers + x[19]
          print(totalArea)
          print(totalLandBoundaries)
          print(totalCoastline)
          print(totalPopulation)
          print(totalLifeExpectancy)
          print(totalGDP)
          print(totalLiteracyRate)
          print(totalHealthExpenditure)
          print(totalInternetUsers)
```

```
136159276.6
541082.2
762467.2
7156890822
13921.61
86690158000000
16716.8279793
1377.01957672
1808910141
```

For each country, calculate the index and then create a new array to store the calculated index

```
In [121]: indexArray = []
          for x in np_array:
              indexArray.append(x[10]/totalArea+x[11]/totalLandBoundaries+x[12]/totalCoastline+x[7]/tota
          print(indexArray)
```

```
[0.03365614847675576, 0.019396153816547593, 0.05521456383073162, 0.0174545329912076, 0.0364247633517585
```

```
In [123]: print("The fathest neighbors are:")
          print(np_array[:,0][indexArray.index(max(indexArray))])
```

```
    print("And")
    print(np_array[:,0][indexArray.index(min(indexArray))])
```

```
The fathest neighbors are:
China
And
Gambia, The
```

```
In [130]: sortedIndexArray = list(indexArray)
          sortedIndexArray.sort()
          intervals = []
          for i in range(0, size(sortedIndexArray)-1):
              print(np_array[:,0][indexArray.index(sortedIndexArray[i])])
              intervals.append(abs(sortedIndexArray[i]-sortedIndexArray[i+1]))
          print(np_array[:,0][indexArray.index(sortedIndexArray[i+1])])
          countryOneIndex = sortedIndexArray[intervals.index(min(intervals))]
          countryTwoIndex = sortedIndexArray[intervals.index(min(intervals))+1]
          print("\n\n\n")
          print("The closest neighbors are:")
          print(np_array[:,0][indexArray.index(countryOneIndex)])
          print("And")
          print(np_array[:,0][indexArray.index(countryTwoIndex)])
```

```
Gambia, The
Comoros
Bhutan
Guinea-Bissau
Timor-Leste
Brunei
Seychelles
Saint Kitts and Nevis
Sao Tome and Principe
Saint Vincent and the Grenadines
Equatorial Guinea
Bahrain
Cabo Verde
Fiji
Belize
Monaco
Dominica
Grenada
Tonga
Mauritius
Antigua and Barbuda
Djibouti
Swaziland
Saint Lucia
Trinidad and Tobago
Vanuatu
Qatar
Samoa
Benin
Nauru
San Marino
Liechtenstein
```

Jamaica
Barbados
Andorra
Kosovo
Kuwait
Malta
Armenia
Maldives
Haiti
Burundi
Kiribati
Luxembourg
Cyprus
El Salvador
Eritrea
Lebanon
Macedonia
Togo
Albania
Montenegro
Suriname
Palau
Rwanda
Lesotho
Gabon
Sri Lanka
Latvia
Bahamas, The
Dominican Republic
Singapore
Western Sahara
Burkina Faso
Senegal
Slovenia
Lithuania
Cambodia
Tuvalu
Oman
Guyana
Guinea
Nicaragua
Guatemala
Estonia
Marshall Islands
United Arab Emirates
Honduras
Jordan
Bosnia and Herzegovina
Laos
Azerbaijan
Malawi
Congo, Republic of the
Panama
Solomon Islands

Uruguay
Georgia
Nepal
Sierra Leone
Moldova
Papua New Guinea
Ghana
Costa Rica
Syria
Tunisia
Tajikistan
Bulgaria
Turkmenistan
Somalia
Zimbabwe
Botswana
Madagascar
Slovakia
Iceland
Central African Republic
Cote d'Ivoire
Ireland
Israel
Liberia
Yemen
Kyrgyzstan
Korea, North
Belarus
Serbia
Micronesia, Federated States of
South Sudan
Cuba
Hungary
Namibia
Uganda
Cameroon
Czech Republic
Mauritania
Ecuador
Zambia
Portugal
Paraguay
Croatia
Chad
Switzerland
Niger
Finland
Belgium
Austria
Romania
Afghanistan
Iraq
Kenya
Mozambique

Denmark
Angola
Mali
Morocco
Bolivia
Tanzania
Uzbekistan
Libya
Burma
Sweden
Taiwan
Mongolia
Netherlands
Ethiopia
New Zealand
Malaysia
Greece
Bangladesh
Venezuela
Ukraine
Sudan
Poland
Peru
South Africa
Algeria
Chile
Egypt
Thailand
Saudi Arabia
Vietnam
Congo, Democratic Republic of the
Norway
Colombia
Kazakhstan
Iran
Korea, South
Spain
Pakistan
Turkey
Italy
Nigeria
Philippines
Argentina
Greenland
France
United Kingdom
Mexico
Germany
Australia
Indonesia
Japan
Brazil
Russia
India

```
Canada
United States
China



The closest neighbors are:
Congo, Republic of the
And
Panama
```

1.5.1 Conclusion:

By looking at the ranking of the countries using my index, what I did right was, on the head of the list (bottom on the table), these countries either have large area, big population, high GDP, all in all ,these countries have at lease one aspect that makes it stand out of all other countries. While on the tail of the list (top of the table), these countries are most ones that I've never heard of, meaning that they have least influence on the world. Both the result, farthest and closest neighbors make perfect sense.

One interesting pair could be France and Greenland. I don't think these two countries have much in common, the reason why they stay together is probably one have advantage in natural aspects while the other is stronger in social aspects.

One example that my model does something wrong is that Monaco, the country that I just identified as best social-welfare country, falls to the end part of the list, which pacific lsland countries, caribbean countries, and African countries. The only thing that Monaco has in common with those countries might be the area. This is where my ranking system needs refine.

```
In []:
```