POC of Data Visualization using immigration data of Canada

as available on Kaggle https://www.kaggle.com/roshansharma/immigration-to-canada-ibm-dataset/version/1 (https://www.kaggle.com/roshansharma/immigration-to-canada-ibm-dataset/version/1)

In [155]:
```
import numpy as np
import pandas as pd
```

Load the data into pandas dataframe

In [156]:
```
df_can = pd.read_excel('Canada.xlsx',
                       sheet_name=2
```

In [157]:

Out[157]: 8385

First 5 rows

In [158]:

Out[158]:

	Type	Coverage	OdName	AREA	AreaName	REG	RegName	DEV	DevName	1980	...	2004	2005
0	Immigrants	Foreigners	Afghanistan	935	Asia	5501	Southern Asia	902	Developing regions	16	...	2978	3436
1	Immigrants	Foreigners	Albania	908	Europe	925	Southern Europe	901	Developed regions	1	...	1450	1223
2	Immigrants	Foreigners	Algeria	903	Africa	912	Northern Africa	902	Developing regions	80	...	3616	3626
3	Immigrants	Foreigners	American Samoa	909	Oceania	957	Polynesia	902	Developing regions	0	...	0	0
4	Immigrants	Foreigners	Andorra	908	Europe	925	Southern Europe	901	Developed regions	0	...	0	0

5 rows × 43 columns

Columns in data frame

In [159]:

```
Out[159]: Index([    'Type', 'Coverage',    'OdName',      'AREA', 'AreaName',       'REG',
                'RegName',       'DEV',   'DevName',      1980,       1981,       1982,
                     1983,       1984,       1985,       1986,       1987,       1988,
                     1989,       1990,       1991,       1992,       1993,       1994,
                     1995,       1996,       1997,       1998,       1999,       2000,
                     2001,       2002,       2003,       2004,       2005,       2006,
                     2007,       2008,       2009,       2010,       2011,       2012,
                     2013],
              dtype='object')
```

Remove columns not useful for visualization

In [160]:

```
df_can.drop(['AREA', 'REG', 'DEV', 'Type', 'Coverage'], axis=1, inplace=True)
```

Out[160]:

	OdName	AreaName	RegName	DevName	1980	1981	1982	1983	1984	1985	...	2004	2005	2006	200
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	2978	3436	3009	265
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1450	1223	856	70
2	Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	...	3616	3626	4807	362
3	American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	...	0	0	1	
4	Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	...	0	0	1	

5 rows × 38 columns

Rename some of the columns to sensible names

In [161]:

```
df_can.rename(columns={'OdName':'Country', 'AreaName':'Continent','RegName':'Region'}

df_can.head()
```

Out[161]:

	Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	...	2004	2005	2006	2007
0	Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	...	2978	3436	3009	2652
1	Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	...	1450	1223	856	702
2	Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	...	3616	3626	4807	3623
3	American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	...	0	0	1	0
4	Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	...	0	0	1	1

5 rows × 38 columns

In [162]:

```
Out[162]: False
```

In [163]:
```
# change column names to strings
df_can.columns = list(map(str, df_can.columns))
```

Out[163]: True

In [164]:
```
# Set the country name as index
df_can_org = df_can.copy()
df_can.set_index('Country', inplace=True)
```

Out[164]:

Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	1986	...	2004	2005	2006	2
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	2978	3436	3009	2
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	1	...	1450	1223	856	
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	69	...	3616	3626	4807	3
American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	0	...	0	0	1	
Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	2	...	0	0	1	

5 rows × 37 columns

In [165]:
```
# Add a total column
df_can['Total'] = df_can.sum(axis=1)
```

Out[165]:

Country	Continent	Region	DevName	1980	1981	1982	1983	1984	1985	1986	...	2005	2006	2007	2
Afghanistan	Asia	Southern Asia	Developing regions	16	39	39	47	71	340	496	...	3436	3009	2652	
Albania	Europe	Southern Europe	Developed regions	1	0	0	0	0	0	1	...	1223	856	702	
Algeria	Africa	Northern Africa	Developing regions	80	67	71	69	63	44	69	...	3626	4807	3623	4
American Samoa	Oceania	Polynesia	Developing regions	0	1	0	0	0	0	0	...	0	1	0	
Andorra	Europe	Southern Europe	Developed regions	0	0	0	0	0	0	2	...	0	1	1	

5 rows × 38 columns

In [166]:
```
# list of data columns for plotting data
years = list(map(str, range(1980, 2014)))
```

Out[166]:
```
['1980',
 '1981',
 '1982',
 '1983',
 '1984',
 '1985',
 '1986',
 '1987',
 '1988',
 '1989',
 '1990',
 '1991',
 '1992',
 '1993',
 '1994',
 '1995',
 '1996',
 '1997',
 '1998',
 '1999',
 '2000',
 '2001',
 '2002',
 '2003',
 '2004',
 '2005',
 '2006',
 '2007',
 '2008',
 '2009',
 '2010',
 '2011',
 '2012',
 '2013']
```

In [167]:
```
# Check out the statics of the data
```

Out[167]:

	1980	1981	1982	1983	1984	1985	1986	
count	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.000000	195.0
mean	508.394872	566.989744	534.723077	387.435897	376.497436	358.861538	441.271795	691.1
std	1949.588546	2152.643752	1866.997511	1204.333597	1198.246371	1079.309600	1225.576630	2109.2
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.500000	0.5
50%	13.000000	10.000000	11.000000	12.000000	13.000000	17.000000	18.000000	26.0
75%	251.500000	295.500000	275.000000	173.000000	181.000000	197.000000	254.000000	434.0
max	22045.000000	24796.000000	20620.000000	10015.000000	10170.000000	9564.000000	9470.000000	21337.0

8 rows × 35 columns

Comparision Charts

In [168]:
```
### Immigration from India and China
df_CI = df_can.loc[['India', 'China'], years]
```

Out[168]:

Country	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2004	2005	2006	2007	2008
India	8880	8670	8147	7338	5704	4211	7150	10189	11522	10343	...	28235	36210	33848	28742	2826
China	5123	6682	3308	1863	1527	1816	1960	2643	2758	4323	...	36619	42584	33518	27642	3003

2 rows × 34 columns

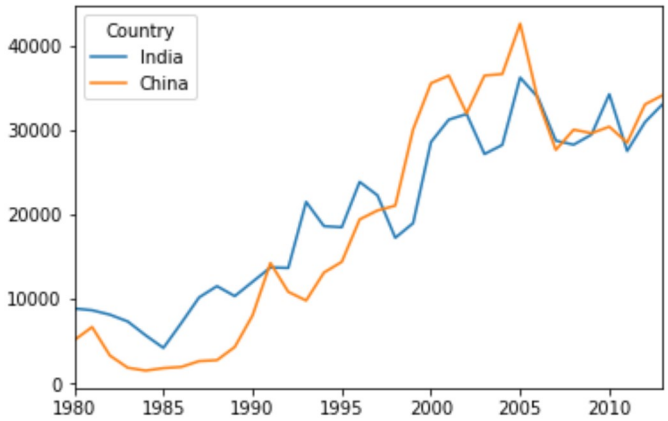
In [169]:
```
df_CI = df_CI.transpose()
```

Out[169]:

Country	India	China
1980	8880	5123
1981	8670	6682
1982	8147	3308
1983	7338	1863
1984	5704	1527

In [170]:
```

```

Out[170]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x18c48d0>
```

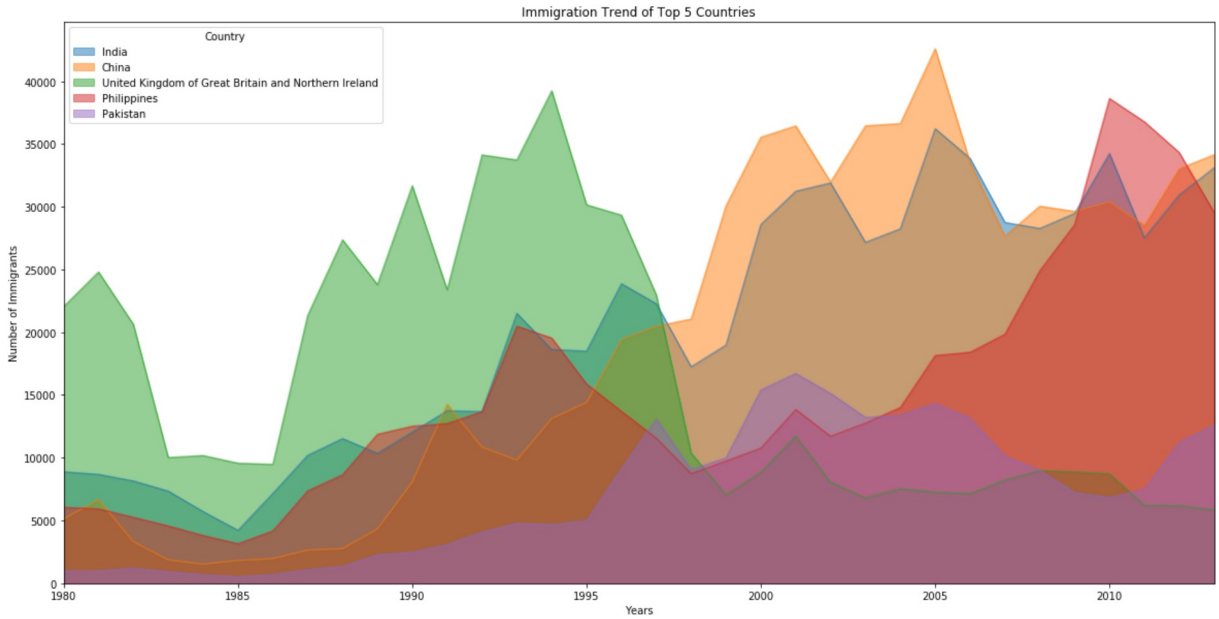


In [171]:
```
# Top 5 immigration trends
df_can.sort_values(by='Total', ascending=False, axis=0, inplace=True)

df_top5 = df_can.head(5)
df_top5 = df_top5[years].transpose()
print(df_top5)

df_top5.index = df_top5.index.map(int) # let's change the index values of df_top5 to
df_top5.plot(kind='line', figsize=(14, 8)) # pass a tuple (x, y) size
plt.show()
```

```
Country  India  China  United Kingdom of Great Britain and Northern Ireland  \
1980      8880   5123                                                22045
1981      8670   6682                                                24796
1982      8147   3308                                                20620
1983      7338   1863                                                10015
1984      5704   1527                                                10170
1985      4211   1816                                                 9564
1986      7150   1960                                                 9470
1987     10189   2643                                                21337
1988     11522   2758                                                27359
1989     10343   4323                                                23795
1990     12041   8076                                                31668
1991     13734  14255                                                23380
1992     13673  10846                                                34123
1993     21496   9817                                                33720
1994     18620  13128                                                39231
1995     18489  14398                                                30145
1996     23859  19415                                                29322
1997     22268  20475                                                22965
1998     17241  21049                                                10367
1999     18974  30069                                                 7045
2000     28572  35529                                                 8840
2001     31223  36434                                                11728
2002     31889  31961                                                 8046
2003     27155  36439                                                 6797
2004     28235  36619                                                 7533
2005     36210  42584                                                 7258
2006     33848  33518                                                 7140
2007     28742  27642                                                 8216
2008     28261  30037                                                 8979
2009     29456  29622                                                 8876
2010     34235  30391                                                 8724
2011     27509  28502                                                 6204
2012     30933  33024                                                 6195
2013     33087  34129                                                 5827

Country  Philippines  Pakistan
1980            6051       978
1981            5921       972
1982            5249      1201
1983            4562       900
1984            3801       668
1985            3150       514
1986            4166       691
1987            7360      1072
1988            8639      1334
1989           11865      2261
1990           12509      2470
1991           12718      3079
1992           13670      4071
1993           20479      4777
1994           19532      4666
1995           15864      4994
```

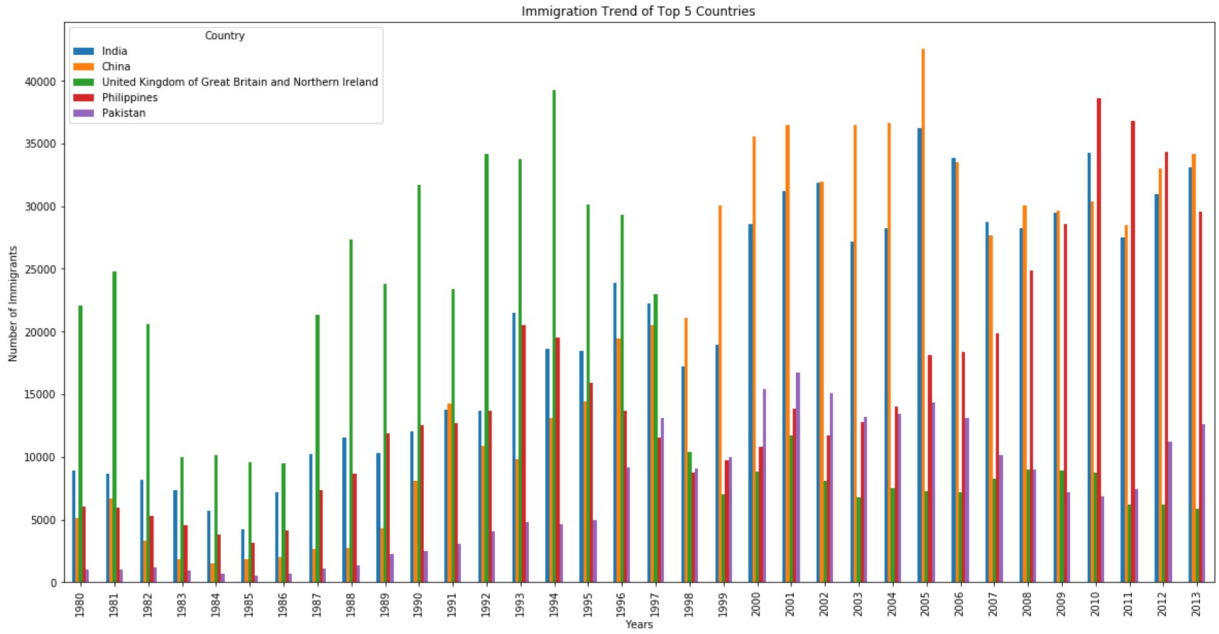
Above as Area Plot

In [172]:
```
df_top5.index = df_top5.index.map(int) # let's change the index values of df_top5 to
df_top5.plot(kind='area',
             stacked=False,
             figsize=(20, 10), # pass a tuple (x, y) size
             )

plt.title('Immigration Trend of Top 5 Countries')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')

plt.show()
```



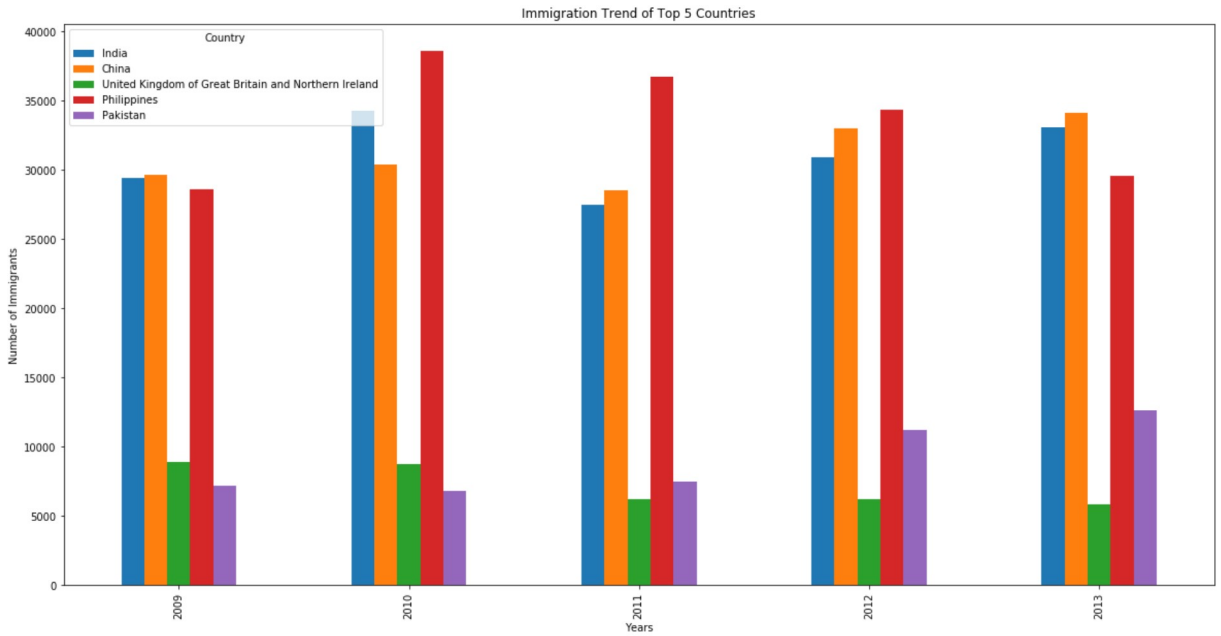
And as Bar Plot

In [173]:
```
df_top5.plot(kind='bar',
             stacked=False,
             figsize=(20, 10), # pass a tuple (x, y) size
             )

plt.title('Immigration Trend of Top 5 Countries')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')
```



The last 5 years of the top 5

In [174]:
```
df_top5last5years = df_top5.transpose()[[2009, 2010,2011, 2012, 2013]].transpose()
df_top5last5years.plot(kind='bar',
                       stacked=False,
                       figsize=(20, 10), # pass a tuple (x, y) size
                       )

plt.title('Immigration Trend of Top 5 Countries')
plt.ylabel('Number of Immigrants')
plt.xlabel('Years')
```



Pie Charts

In [175]:
```
# get countries by continents
df_continents = df_can.groupby('Continent', axis=0).sum()

print(type(df_can.groupby('Continent', axis=0)))
```

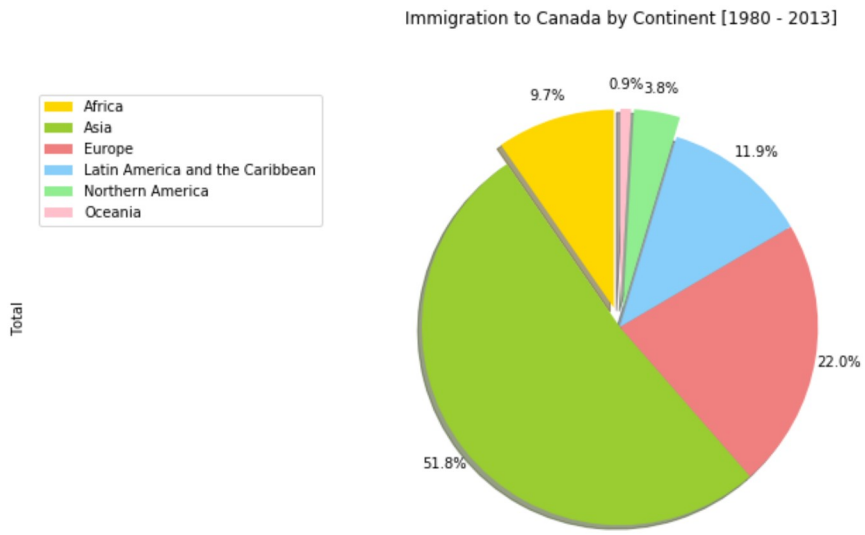
```
<class 'pandas.core.groupby.generic.DataFrameGroupBy'>
```

Out[175]:

Continent	1980	1981	1982	1983	1984	1985	1986	1987	1988	1989	...	2005	2006	2007
Africa	3951	4363	3819	2671	2639	2650	3782	7494	7552	9894	...	27523	29188	28284
Asia	31025	34314	30214	24696	27274	23850	28739	43203	47454	60256	...	159253	149054	133459
Europe	39760	44802	42720	24638	22287	20844	24370	46698	54726	60893	...	35955	33053	33495
Latin America and the Caribbean	13081	15215	16769	15427	13678	15171	21179	28471	21924	25060	...	24747	24676	26011
Northern America	9378	10030	9074	7100	6661	6543	7074	7705	6469	6790	...	8394	9613	9463

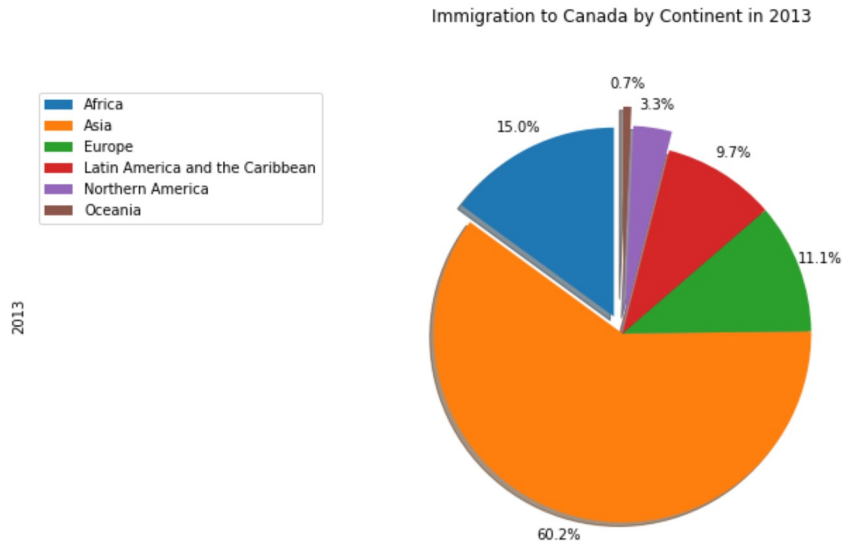
5 rows × 35 columns

In [176]:
```
colors_list = ['gold', 'yellowgreen', 'lightcoral', 'lightskyblue', 'lightgreen', 'pi
explode_list = [0.1, 0, 0, 0, 0.1, 0.1] # ratio for each continent with which to offs

df_continents['Total'].plot(kind='pie',
                            figsize=(15, 6),
                            autopct='%1.1f%%',
                            startangle=90,
                            shadow=True,
                            labels=None,         # turn off labels on pie chart
                            pctdistance=1.12,    # the ratio between the center of ea
                            colors=colors_list,  # add custom colors
                            explode=explode_list # 'explode' lowest 3 continents
                            )

# scale the title up by 12% to match pctdistance
plt.title('Immigration to Canada by Continent [1980 - 2013]', y=1.12)

plt.axis('equal')

# add legend
plt.legend(labels=df_continents.index, loc='upper left')

plt.show()
```



Pie for 2013 immigration

In [177]:
```
explode_list = [0.1, 0, 0, 0, 0.1, 0.2] # ratio for each continent with which to offs
df_continents['2013'].plot(kind='pie',
                           figsize=(15, 6),
                           autopct='%1.1f%%',
                           startangle=90,
                           shadow=True,
                           labels=None,                 # turn off labels on pie cha
                           pctdistance=1.12,            # the ratio between the pie
                           explode=explode_list         # 'explode' lowest 3 contine
                           )
plt.title('Immigration to Canada by Continent in 2013', y=1.12)
plt.axis('equal')
plt.legend(labels=df_continents.index, loc='upper left')
plt.show()
```



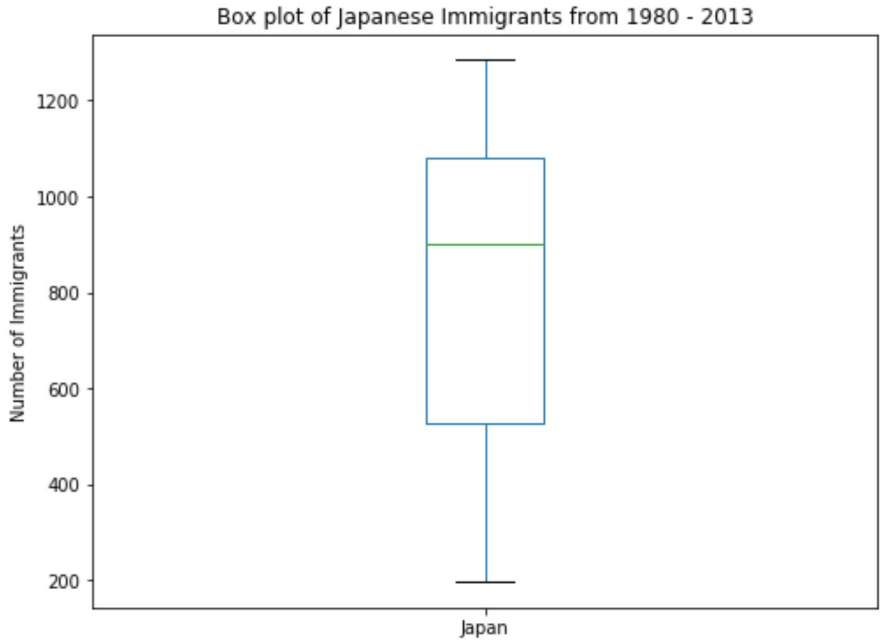
Box Plots

In [178]:
```
df_japan = df_can.loc[['Japan'], years].transpose()
```

Out[178]:

Country	Japan
1980	701
1981	756
1982	598
1983	309
1984	246

In [179]:
```
df_japan.plot(kind='box', figsize=(8, 6))

plt.title('Box plot of Japanese Immigrants from 1980 - 2013')
plt.ylabel('Number of Immigrants')
```



In [180]:

Out[180]:

Country	Japan
count	34.000000
mean	814.911765
std	337.219771
min	198.000000
25%	529.000000
50%	902.000000
75%	1079.000000
max	1284.000000

Compare with Box plot

In [181]:
```
df_CI = df_can.loc[['China','India'], years].transpose()
df_CI.head()
```

Out[181]:

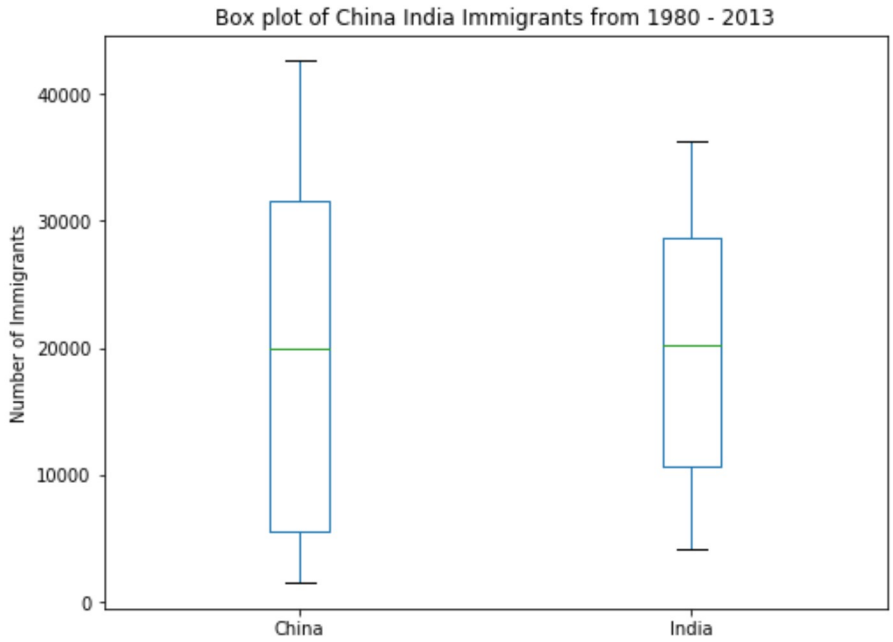
Country	China	India
1980	5123	8880
1981	6682	8670
1982	3308	8147
1983	1863	7338
1984	1527	5704

In [182]:

Out[182]:

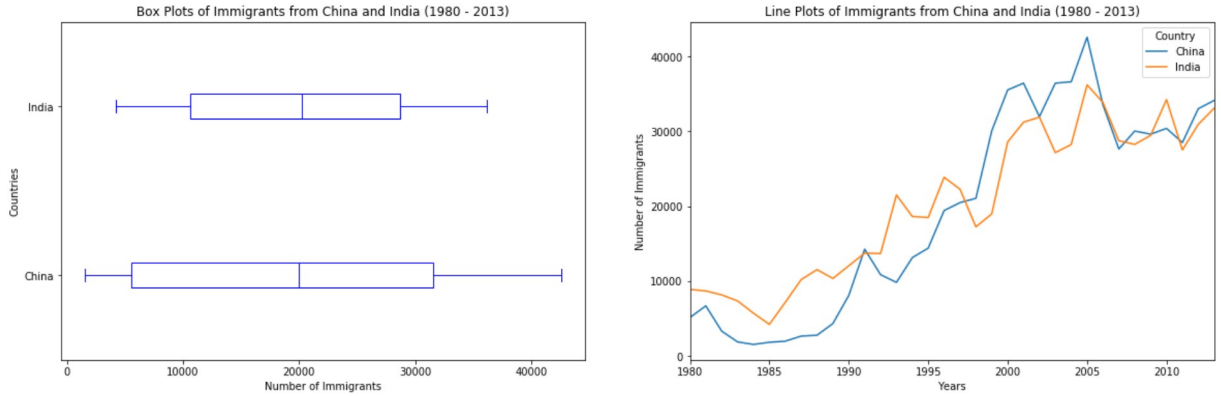
Country	China	India
count	34.000000	34.000000
mean	19410.647059	20350.117647
std	13568.230790	10007.342579
min	1527.000000	4211.000000
25%	5512.750000	10637.750000
50%	19945.000000	20235.000000
75%	31568.500000	28699.500000
max	42584.000000	36210.000000

In [183]:
```
df_CI.plot(kind='box', figsize=(8, 6))

plt.title('Box plot of China India Immigrants from 1980 - 2013')
plt.ylabel('Number of Immigrants')

plt.show()
```



Multiple plots side by side

In [184]:
```
# China India comparision Box and line
fig = plt.figure() # create figure

ax0 = fig.add_subplot(1, 2, 1) # add subplot 1 (1 row, 2 columns, first plot)
ax1 = fig.add_subplot(1, 2, 2) # add subplot 2 (1 row, 2 columns, second plot). See t

# Subplot 1: Box plot
df_CI.plot(kind='box', color='blue', vert=False, figsize=(20, 6), ax=ax0) # add to su
ax0.set_title('Box Plots of Immigrants from China and India (1980 - 2013)')
ax0.set_xlabel('Number of Immigrants')
ax0.set_ylabel('Countries')

# Subplot 2: Line plot
df_CI.plot(kind='line', figsize=(20, 6), ax=ax1) # add to subplot 2
ax1.set_title ('Line Plots of Immigrants from China and India (1980 - 2013)')
ax1.set_ylabel('Number of Immigrants')
ax1.set_xlabel('Years')

plt.show()
```



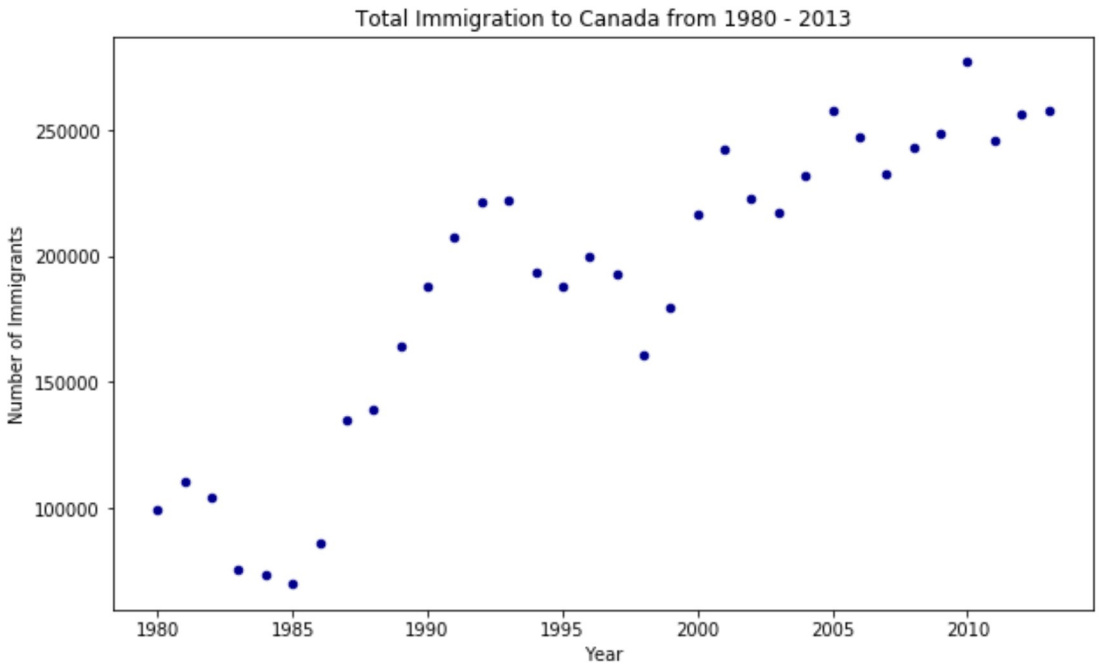
Scatter Plot - immigration trend/correlation

In [185]:
```
# we can use the sum() method to get the total population per year
df_tot = pd.DataFrame(df_can[years].sum(axis=0))

# change the years to type int (useful for regression later on)
df_tot.index = map(int, df_tot.index)

# reset the index to put in back in as a column in the df_tot dataframe
df_tot.reset_index(inplace = True)

# rename columns
df_tot.columns = ['year', 'total']

# view the final dataframe
```

Out[185]:

	year	total
0	1980	99137
1	1981	110563
2	1982	104271
3	1983	75550
4	1984	73417

In [186]:
```
df_tot.plot(kind='scatter', x='year', y='total', figsize=(10, 6), color='darkblue')

plt.title('Total Immigration to Canada from 1980 - 2013')
plt.xlabel('Year')
plt.ylabel('Number of Immigrants')
```



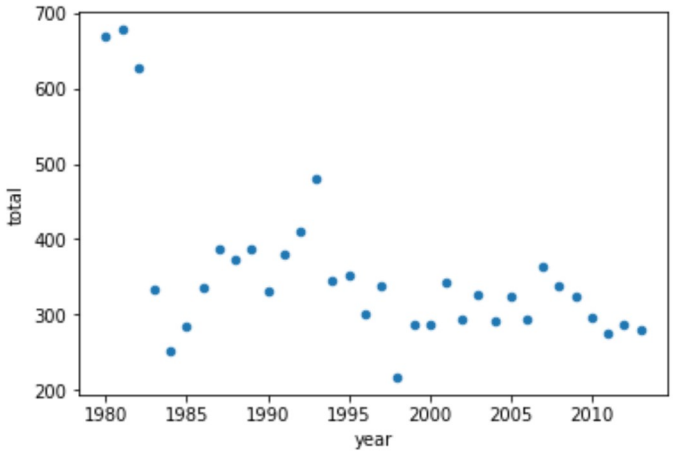
From Denmark, Norway and Sweden

In [187]:
```
df_countries = df_can.loc[['Denmark', 'Norway', 'Sweden'], years].transpose()

df_total = pd.DataFrame(df_countries.sum(axis=1))

df_total.reset_index(inplace=True)
df_total.columns = ['year', 'total']
df_total['year'] = df_total['year'].astype(int)
df_total.head()
```

Out[187]:

	year	total
0	1980	669
1	1981	678
2	1982	627
3	1983	333
4	1984	252

In [188]:
```
df_total.plot(kind='scatter',x='year',y='total')
plt.show()
```



Bubble Plots - India and China

In [189]:
```
df_can_t = df_can[years].transpose() # transposed dataframe

df_can_t.index = map(int, df_can_t.index)

# let's label the index. This will automatically be the column name when we reset the
df_can_t.index.name = 'Year'

# reset index to bring the Year in as a column
df_can_t.reset_index(inplace=True)

# view the changes
```

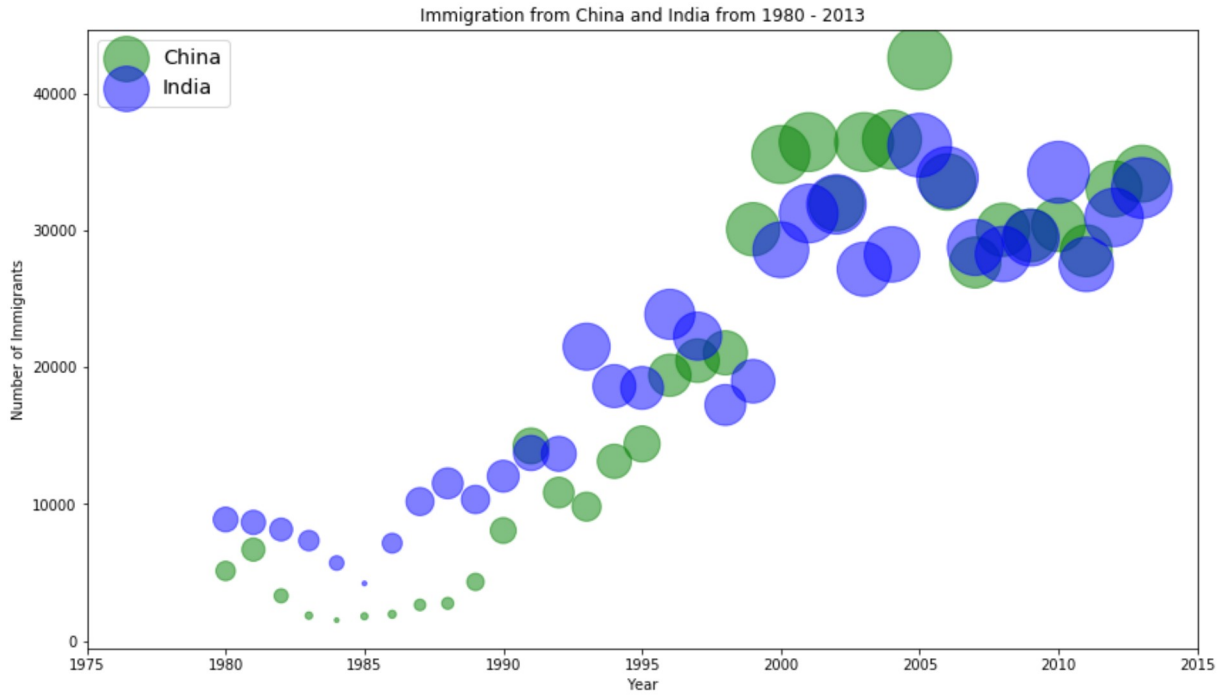
Out[189]:

Country	Year	India	China	United Kingdom of Great Britain and Northern Ireland	Philippines	Pakistan	United States of America	Iran (Islamic Republic of)	Sri Lanka	Republic of Korea	...	Kiribati	V
0	1980	8880	5123	22045	6051	978	9378	1172	185	1011	...	0	
1	1981	8670	6682	24796	5921	972	10030	1429	371	1456	...	0	
2	1982	8147	3308	20620	5249	1201	9074	1822	290	1572	...	0	
3	1983	7338	1863	10015	4562	900	7100	1592	197	1081	...	1	
4	1984	5704	1527	10170	3801	668	6661	1977	1086	847	...	0	

5 rows × 196 columns

In [190]:
```
norm_china = (df_can_t['China'] - df_can_t['China'].min()) / (df_can_t['China'].max()

norm_india = (df_can_t['India'] - df_can_t['India'].min()) / (df_can_t['India'].max()

ax0 = df_can_t.plot(kind='scatter',
                    x='Year',
                    y='China',
                    figsize=(14, 8),
                    alpha=0.5,                  # transparency
                    color='green',
                    s=norm_china * 2000 + 10,  # pass in weights
                    xlim=(1975, 2015)
                   )

ax1 = df_can_t.plot(kind='scatter',
                    x='Year',
                    y='India',
                    alpha=0.5,
                    color="blue",
                    s=norm_india * 2000 + 10,
                    ax = ax0
                   )
ax0.set_ylabel('Number of Immigrants')
ax0.set_title('Immigration from China and India from 1980 - 2013')
ax0.legend(['China', 'India'], loc='upper left', fontsize='x-large')
```

Out[190]: <matplotlib.legend.Legend at 0x191ef30>



Data on maps

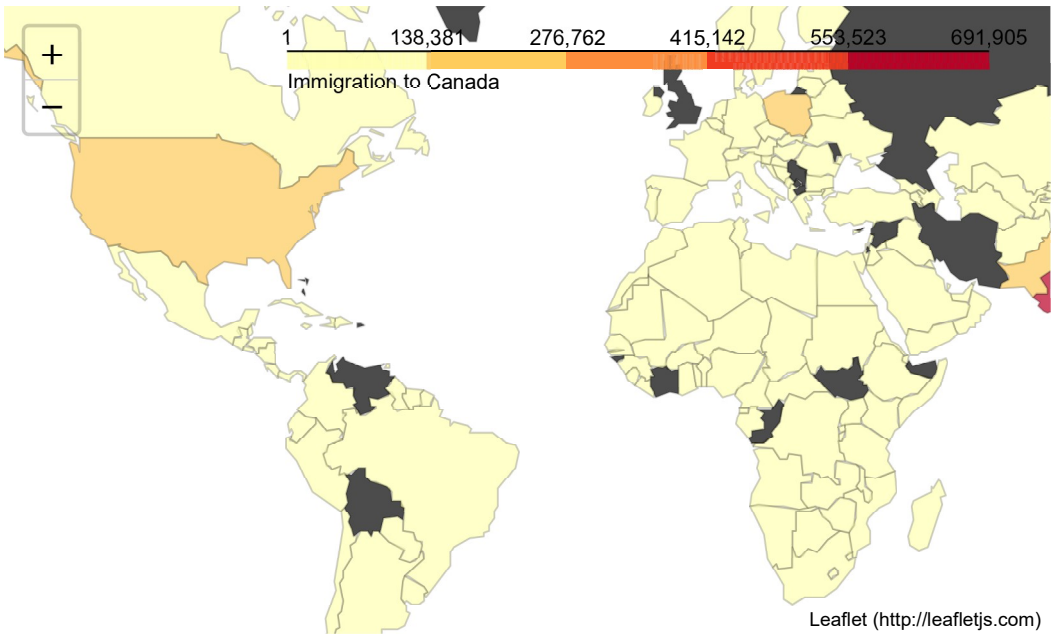
In [191]:

In [192]:
```
world_geo = r'world_countries.json' # geojson file

# create a plain world map
```

In [193]:
```
df_can_org['Total'] = df_can_org.sum(axis=1)
```

In [194]:
```
world_geo = r'world_countries.json'

# create a numpy array of length 6 and has linear spacing from the minium total immig
threshold_scale = np.linspace(df_can_org['Total'].min(),
                              df_can_org['Total'].max(),
                              6, dtype=int)
threshold_scale = threshold_scale.tolist() # change the numpy array to a list
threshold_scale[-1] = threshold_scale[-1] + 1 # make sure that the last value of the

# let Folium determine the scale.
world_map = folium.Map(location=[0, 0], zoom_start=2, tiles='Mapbox Bright')
world_map.choropleth(
    geo_data=world_geo,
    data=df_can_org,
    columns=['Country', 'Total'],
    key_on='feature.properties.name',
    threshold_scale=threshold_scale,
    fill_color='YlOrRd',
    fill_opacity=0.7,
    line_opacity=0.2,
    legend_name='Immigration to Canada',
    reset=True
)
world_map
```

Out[194]:



In [ ]: