NYOUG_JupyterNotebook

June 20, 2018

NYOUG 6/21/2018 - by Linda Li

The following is the case studay of world public air transport data over the past 40 years. It creates Pandas DataFrame on the dataset, uses data wrangling features,analysis and plotting packages for growth linechart and heatmap volume visualization. The geospatial cartopy package is used to draw related countries with different colors to represent different growth rate.

Installation of dependency packages:

pip install pandas-summary conda install -c conda-forge cartopy pip install pandas-msgpack

```
In [1]: import numpy as np
```

```
In [2]: # A Numpy matrix multiplication example
        # use 2D list to generate matrix

        ad1=np.matrix([[1,2,3],[4,5,6]])
        ad2=np.matrix([10,11,12])
```

```
In [3]: ad2
```

```
Out[3]: matrix([[10, 11, 12]])
```

```
In [4]: # numpy array simplifies matrix operation

        ad=np.multiply(ad1, ad2)
        ad
```

```
Out[4]: matrix([[10, 22, 36],
                [40, 55, 72]])
```

numpy matrics, Matrices objects are subclasses of ndarray

```
In [5]: %matplotlib inline
        import pandas as pd
        from pandas import Series, DataFrame
        import numpy as np
        from pandas_summary import DataFrameSummary
        import matplotlib.pyplot as plt
```

In [6]:
```
#show all columns
pd.options.display.max_columns=None

#no sci notation
pd.set_option('display.float_format', lambda x:'%.3f' %x)

#plot sizes
plt.rcParams['figure.figsize']=[20, 10] # change the plot default size
```

In [9]:
```
file_path='/Users/yli/Downloads/'
file_name='AirTravelCountry.csv'
```

In [10]:
```
filepath_or_buffer=file_path+'/'+file_name
```

In [14]:
```
#index_col=0, use the col Country name as index column
#index_col by default is None, meaning it creates 0 based rownumber as index, in this
#the first column country name in the csv file "Country Name" as index

df=pd.read_csv(filepath_or_buffer, sep=',',skiprows=[0],index_col=0)
```

In [15]:
```
#df.shape
#df.head(4)
```

In [16]:
```
df.info(verbose=False)
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 264 entries, Aruba to Zimbabwe
Columns: 61 entries, Country Code to 2017
dtypes: float64(58), object(3)
memory usage: 127.9+ KB
```

In [17]:
```
# In this exercise, the intention is to only analyze the 5 countries
country_code_list=['CHN','DEU','GBR','IND','USA']
df_countries=(df.loc[df['Country Code'].isin(country_code_list)])
```

In [19]:
```
#df_countries.head(4)
```

In [20]:
```
del df_countries.index.name
```

In [22]:
```
#df_countries.head(2)
```

In [23]:
```
df1=df_countries.drop(['Country Code','Indicator Name','Indicator Code'], axis=1)
```

In [25]:
```
#df1
```

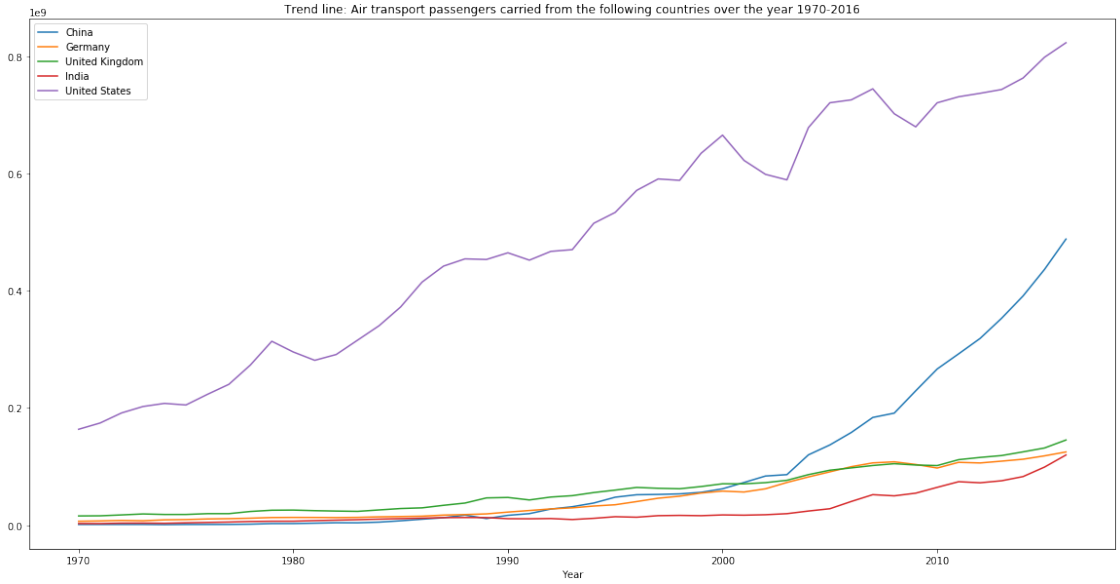
In [26]:
```
df_transpose=df1.transpose()
```

In [27]:
```
#type(df_transpose)
#df_transpose
```

```
In [28]: df_transpose.reset_index(drop=False, inplace=True)

In [30]: #df_transpose.head(15)

In [31]: df_transpose.rename(columns={'index':'Year'}, inplace=True)

In [33]: #df_transpose.head(3)

In [34]: df_transpose['Year'].dtype

Out[34]: dtype('O')

In [35]: df_transpose[['Year']]=df_transpose[['Year']].astype(int)
         df2=df_transpose.loc[(df_transpose['Year']>1969) & (df_transpose['Year']<2017)]

In [36]: type(df2)

Out[36]: pandas.core.frame.DataFrame

In [38]: #df2.head()

In [39]: #fill NaN
         #There are about 40 years of data, for the countries without data at the 70s, we assu
         #the data should be filled with the average the lowest part of the quantile
         df2.China.quantile([0.04, 0.1,0.25, 0.5, 0.75])

Out[39]: 0.040     1034000.000
         0.100     1735800.000
         0.250     6150000.000
         0.500    47564500.000
         0.750   147367487.000
         Name: China, dtype: float64

In [40]: # supress SettingwithCopyWarning -- this warning here could be false positive, need f
         pd.set_option('mode.chained_assignment',None)

In [41]: df2.China.fillna(value=1034000.00, inplace=True)

In [43]: #df2.head()

In [44]: df3=df2.set_index('Year')

In [45]: df3.head()

Out[45]:            China     Germany  United Kingdom       India  United States
         Year
         1970 1034000.000 6498000.000    15568800.000 2671600.000  163448992.000
         1971 1034000.000 7029200.000    15796000.000 2554000.000  174143104.000
         1972 1034000.000 7919100.000    17308500.000 3285800.000  191325408.000
         1973 1034000.000 7371300.000    18959700.000 3391600.000  202309200.000
         1974  710000.000 8886900.000    18062600.000 3037300.000  207612400.000
```

In [46]:
```
#Number of Passengers from those countries grow over the years.
#df2.plot.line()
df3.plot()
plt.title('Trend line: Air transport passengers carried from the following countries
```

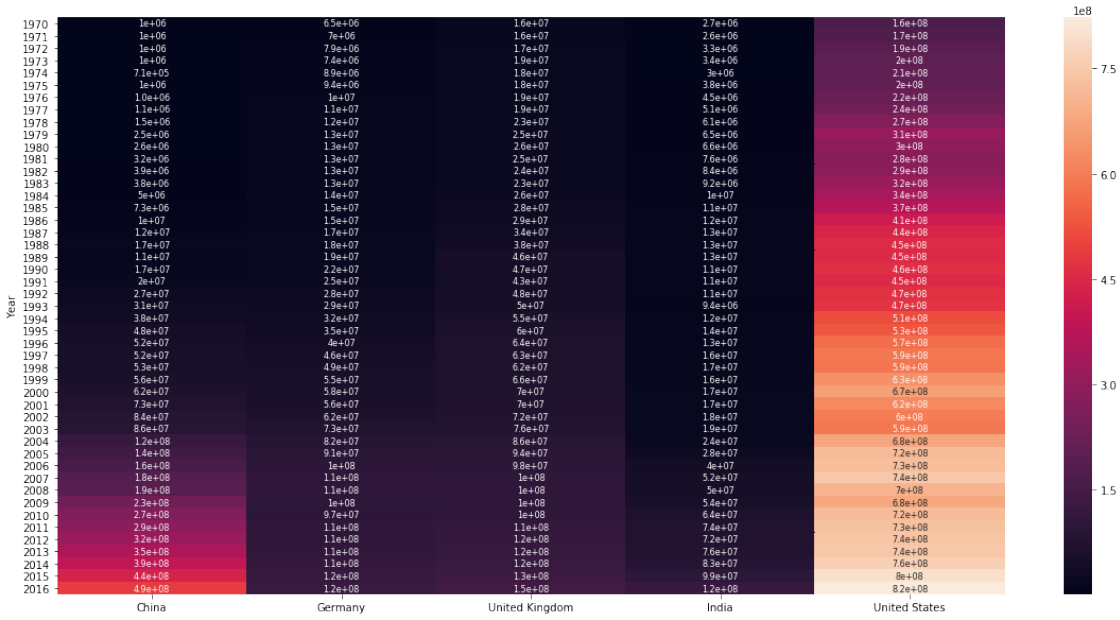
Out[46]: Text(0.5,1,'Trend line: Air transport passengers carried from the following countries



In [47]:
```
import seaborn as sns
```

In [48]:
```
sns.heatmap(df3, annot=True, annot_kws={"size":8})
```

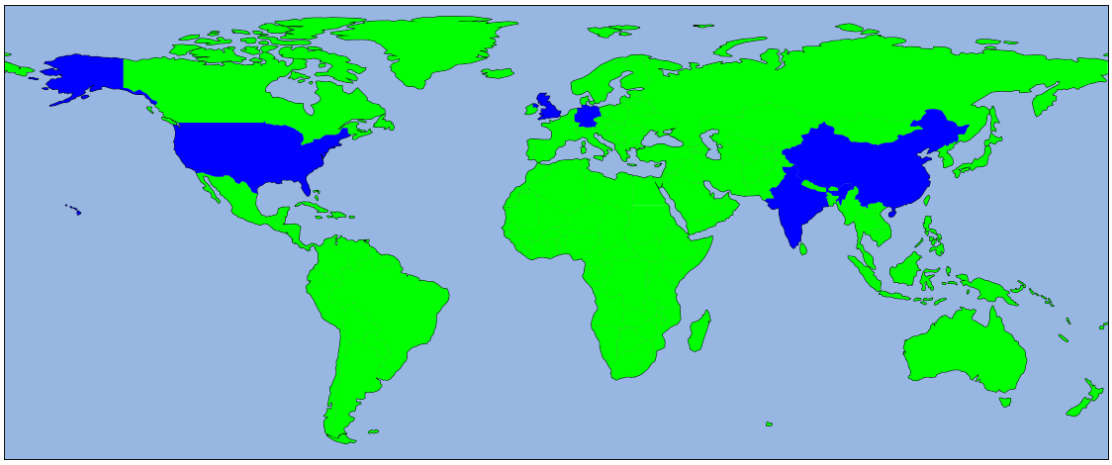
Out[48]: <matplotlib.axes._subplots.AxesSubplot at 0x1a238898d0>



In [49]:
```
import matplotlib.pyplot as plt
import cartopy
import cartopy.io.shapereader as shpreader
import cartopy.crs as ccrs
```

In [50]:
```
ax = plt.axes(projection=ccrs.PlateCarree())
#ax.add_feature(cartopy.feature.LAND)
ax.add_feature(cartopy.feature.OCEAN)
ax.add_feature(cartopy.feature.COASTLINE)
ax.add_feature(cartopy.feature.BORDERS, linestyle='-', alpha=.5)
ax.add_feature(cartopy.feature.LAKES, alpha=0.95)
ax.add_feature(cartopy.feature.RIVERS)
ax.set_extent([-150, 60, -25, 80])

shpfilename = shpreader.natural_earth(resolution='110m',
                                      category='cultural',
                                      name='admin_0_countries')
reader = shpreader.Reader(shpfilename)
countries = reader.records()

print(countries)


#c_list=['CHN','DEU','GBR','IND','USA']

for country in countries:
    if country.attributes['ADM0_A3'] in country_code_list:
        #print(country.attributes['ADM0_A3'],country_color[country.attributes['ADM0_A
        ax.add_geometries(country.geometry, ccrs.PlateCarree(),
        #facecolor=(growth_rank_dic[country.attributes['ADM0_A3']],0.0,0.0),
        facecolor=(0.0,0.0,1.0)),
        label=country.attributes['ADM0_A3']

    else:
        ax.add_geometries(country.geometry, ccrs.PlateCarree(),
                          facecolor=(0, 1, 0),
                          label=country.attributes['ADM0_A3'])

plt.show()

print("Country Code List")
country_code_list
```

```
<generator object Reader.records at 0x1a2387deb8>
```



Country Code List

```
Out[50]: ['CHN', 'DEU', 'GBR', 'IND', 'USA']

In [51]: country_list=['China','Germany','United Kingdom','India','United States']
         growth_list=[df3['China'].pct_change().mean(), df3['Germany'].pct_change().mean(),df3
                      df3['India'].pct_change().mean(),df3['United States'].pct_change().mean()

In [52]: growth_list

Out[52]: [0.1586012345933269,
          0.06804200140477858,
          0.05156417408902534,
          0.09275293619437107,
          0.03712254086680223]

In [53]: # make country growth tuple list
         country_growth_list=list(zip(country_list,growth_list))

In [54]: country_growth_list

Out[54]: [('China', 0.1586012345933269),
          ('Germany', 0.06804200140477858),
          ('United Kingdom', 0.05156417408902534),
          ('India', 0.09275293619437107),
          ('United States', 0.03712254086680223)]

In [55]: #sort the list
         sorted(country_growth_list,key=lambda x:x[1])
```

```
Out[55]: [('United States', 0.03712254086680223),
          ('United Kingdom', 0.05156417408902534),
          ('Germany', 0.06804200140477858),
          ('India', 0.09275293619437107),
          ('China', 0.1586012345933269)]

In [56]: growth_list=[0.16,0.07, 0.05,0.1,0.04]
         # this list was defined in the code country_code_list=['CHN','DEU','GBR','IND','USA']

In [57]: #x[1] sort by 2nd element of the tuple
         sort_by_growth=sorted(list(zip(country_code_list, growth_list)), key=lambda x:x[1])

In [58]: sort_by_growth

Out[58]: [('USA', 0.04), ('GBR', 0.05), ('DEU', 0.07), ('IND', 0.1), ('CHN', 0.16)]

In [59]: sorted_country_code_list=[item[0] for item in sort_by_growth]

In [60]: rank_list=list(range(5))
         rank_list

Out[60]: [0, 1, 2, 3, 4]

In [61]: # I cannot use the growth rate as color code, they are not far apart and the color di
         # difficult to tell
         # introduce the ranking list
         # need to associate the country code to the rank value
         color_rank_list=[item/10 for item in rank_list]
         color_rank_list

Out[61]: [0.0, 0.1, 0.2, 0.3, 0.4]

In [62]: growth_rank_dic=dict(zip(sorted_country_code_list, color_rank_list))

In [63]: growth_rank_dic

Out[63]: {'USA': 0.0, 'GBR': 0.1, 'DEU': 0.2, 'IND': 0.3, 'CHN': 0.4}

In [64]: # the following is the geo map which highlights the countries in the travel dataframe

In [65]: ax = plt.axes(projection=ccrs.PlateCarree())
         #ax.add_feature(cartopy.feature.LAND)
         ax.add_feature(cartopy.feature.OCEAN)
         ax.add_feature(cartopy.feature.COASTLINE)
         ax.add_feature(cartopy.feature.BORDERS, linestyle='-', alpha=.5)
         ax.add_feature(cartopy.feature.LAKES, alpha=0.95)
         ax.add_feature(cartopy.feature.RIVERS)
         ax.set_extent([-150, 60, -25, 80])

         shpfilename = shpreader.natural_earth(resolution='110m',
```

```
                                        category='cultural',
                                        name='admin_0_countries')
reader = shpreader.Reader(shpfilename)
countries = reader.records()

#print("countries")
#print(countries)



for country in countries:
    if country.attributes['ADM0_A3'] in growth_rank_dic.keys():
        #print(c_list.attributes['ADM0_A3'],country_color[c_list.attributes['ADM0_A3'
        ax.add_geometries(country.geometry, ccrs.PlateCarree(),
        facecolor=(growth_rank_dic[country.attributes['ADM0_A3']],0.0,0.0),
        label=country.attributes['ADM0_A3'])

    else:
        ax.add_geometries(country.geometry, ccrs.PlateCarree(),
                          facecolor=(0, 1, 0),
                          label=country.attributes['ADM0_A3'])

plt.show()
```