[stephenpaff](https://github.com/stephenpaff)/[**World\_Bank\_Dataset**](https://github.com/stephenpaff/World_Bank_Dataset)

Git

World Bank Data Analysis

In this project, I cleaned up the data from the World Bank. The dataset was initially in JSON format and can be found on this website: <http://jsonstudio.com/resources/>.

I conducted three tasks with the dataset:

1) I found the 10 countries with most projects.

2) I found the top 10 major project themes, using column 'mjtheme\_namecode'.

3) In 2, I noticed that some entries have only the code and the name is missing, so I created a new dataframe with the missing names filled in.

In [53]:

*## I will start by importing the dataset and the pandas and JSON packages.*

In [54]:

*# Imports* **import** **pandas** **as** **pd** **import** **json** **from** **pandas.io.json** **import** json\_normalize

In [57]:

*# Creates a Pandas Dataframe out of the json file.*  df = pd.read\_json('world\_bank\_projects.json')

In [ ]:

Task 1:

Find the 10 countries with the most projects

In [59]:

*'''* *Outputs the countries with the most projects.*  *There is one problem, though: it includes Africa as a country* *in 9th spot.*  *Several projects which are in multiple African countries have*  *"Africa" stored as their country.*  *'''* df.countryshortname.value\_counts().head(10)

Out[59]:

China 19 Indonesia 19 Vietnam 17 India 16 Yemen, Republic of 13 Morocco 12 Nepal 12 Bangladesh 12 Mozambique 11 Africa 11 Name: countryshortname, dtype: int64

In [6]:

*'''* *The database does not store which specific* *African countries are involved in these projects with Africa as the label,* *so I do not have access to which specific African countries these projects are in,*  *meaning I can do little with those data values.*  *Thus I will create a new data frame that excludes the projects* *with Africa as their country label and find its 10 most common countries.*  *'''* *# df\_notAfr is the copy of the dataframe excluding projects labelled with Africa as their company name.*  df\_notAfr = df[df.countryshortname != "Africa"] df\_notAfr.countryshortname.value\_counts().head(10)

Out[6]:

Indonesia 19 China 19 Vietnam 17 India 16 Yemen, Republic of 13 Nepal 12 Bangladesh 12 Morocco 12 Mozambique 11 Pakistan 9 Name: countryshortname, dtype: int64

In [ ]:

Task 2:

Find the 10 major project themes, using column 'mjtheme\_namecode'

In [60]:

*'''* *A detailed inspection of mjtheme\_namecode in df reveals that* *each project contains list of dictionaries containing the code* *values and corresponding code name (with some names missing,*  *which I will address in Task 3).* *Note: Upon closer inspection, all the codes are between*  *1 and 11, and all filled code names are consistent.*  *For example, all code 8 values with a non-empty name are* *named consistently.*  *Also note: Some projects include code repeats. For example, the third* *project includes Code 7 twice. The counting method below will make*  *that multiple projects from one code are only counted once.*  *'''* *# code\_max is the highest code value in the dataset* code\_max = 11 df.mjtheme\_namecode.head(20)

Out[60]:

0 [{'code': '8', 'name': 'Human development'}, {... 1 [{'code': '1', 'name': 'Economic management'},... 2 [{'code': '5', 'name': 'Trade and integration'... 3 [{'code': '7', 'name': 'Social dev/gender/incl... 4 [{'code': '5', 'name': 'Trade and integration'... 5 [{'code': '6', 'name': 'Social protection and ... 6 [{'code': '2', 'name': 'Public sector governan... 7 [{'code': '11', 'name': 'Environment and natur... 8 [{'code': '10', 'name': 'Rural development'}, ... 9 [{'code': '2', 'name': 'Public sector governan... 10 [{'code': '10', 'name': 'Rural development'}, ... 11 [{'code': '10', 'name': 'Rural development'}, ... 12 [{'code': '4', 'name': ''}] 13 [{'code': '5', 'name': 'Trade and integration'... 14 [{'code': '6', 'name': 'Social protection and ... 15 [{'code': '10', 'name': 'Rural development'}, ... 16 [{'code': '10', 'name': 'Rural development'}, ... 17 [{'code': '8', 'name': 'Human development'}, {... 18 [{'code': '8', 'name': 'Human development'}, {... 19 [{'code': '2', 'name': 'Public sector governan... Name: mjtheme\_namecode, dtype: object

In [61]:

*'''* *Counts the number of projects that invoke each code,*  *storing that as a dictionary and then converting it to a Series.*  *md, short of 'master dictionary', will store the number of* *projects that invoke each code number.* *'''* md = { } *# Initialize each code's count at zero.*  **for** i **in** range(1, code\_max+1): md[i] = 0 *# Move through each set of stored codes for each project to count the number of projects that include each code.*  **for** x **in** range(len(df)): *# d stores the codes used for each project* d = {} **for** y **in** range(len(df.mjtheme\_namecode[x])): d[int(df.mjtheme\_namecode[x][y]['code'])] = 1 *# Adds the d to overall count in md* **for** i **in** d.keys(): md[i] += 1 *'''* *mdf, short for 'master dataframe', stores the number of projects* *that invoke each code number, converting md to a series*  *that I can now use Pandas to analyze the top occurences.* *'''* mdf = pd.Series(md) mdf

Out[61]:

1 33 2 140 3 14 4 119 5 61 6 116 7 114 8 128 9 40 10 148 11 157 dtype: int64

In [10]:

*# The 10 most frequently invoked codes can now be easily obtained by simply selecting the top 10 counts in mdf.*  mdf.sort\_values(ascending = **False**).head(10)

Out[10]:

11 157 10 148 2 140 8 128 4 119 6 116 7 114 5 61 9 40 1 33 dtype: int64

In [ ]:

Alternate Method for Tasks 2:

In [12]:

*'''* *This alternate method uses json\_normalize to determine the most common theme codes and to fill in empty name values.*  *One big difference between this method and my initial approach is that does counts repeats of theme code values.*  *Many projects include one theme more than once.*  *For example, for a project that includes Theme #7 'Social dev/gender/inclusion' twice,*  *my initial approach would only count that as one instance of Theme 7, but the alternative method will count it twice.*  *I am not sure what it means that a project would include a project more than once, so I simply counted how many times a*  *project included a theme as a 0 or 1 value instead of including the number of occurences. Depending on the use of the data,*  *either way is useful. Hence it is helpful to be able to include both for comparison.*  *'''* *# Creates js, which allows me to to conduct the alternative method* js = json.load(open('world\_bank\_projects.json'))

In [52]:

*# major\_themes normalizes, storing all the mj\_theme\_namecode dictionary values as a DataFrame*  major\_themes = json\_normalize(js, 'mjtheme\_namecode', ['id']) *# Finds the top 10 most common codes* major\_themes.name.value\_counts().head(10)

Out[52]:

Environment and natural resources management 223 Rural development 202 Human development 197 Public sector governance 184 Social protection and risk management 158 Financial and private sector development 130 122 Social dev/gender/inclusion 119 Trade and integration 72 Urban development 47 Name: name, dtype: int64

In [ ]:

Task 3:

Created a new dataframe with the missing names filled in.

In [62]:

*'''* *Printed are all the codes for all the projects. Looking through this*  *large output, one notices that several code names are blank.*  *I will fill in the gaps.*  *limit will provide how many times it will print through the long output, and counter will increase until it hits limit.* *'''* limit = 25 counter = 0 **for** x **in** range(len(df.mjtheme\_namecode)): **for** y **in** range(len(df.mjtheme\_namecode[x])): print(df.mjtheme\_namecode[x][y]) counter += 1 **if**(counter >= limit): **break**

{'code': '8', 'name': 'Human development'} {'code': '11', 'name': ''} {'code': '1', 'name': 'Economic management'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '5', 'name': 'Trade and integration'} {'code': '2', 'name': 'Public sector governance'} {'code': '11', 'name': 'Environment and natural resources management'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '7', 'name': 'Social dev/gender/inclusion'} {'code': '7', 'name': 'Social dev/gender/inclusion'} {'code': '5', 'name': 'Trade and integration'} {'code': '4', 'name': 'Financial and private sector development'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '6', 'name': ''} {'code': '2', 'name': 'Public sector governance'} {'code': '4', 'name': 'Financial and private sector development'} {'code': '11', 'name': 'Environment and natural resources management'} {'code': '8', 'name': ''} {'code': '10', 'name': 'Rural development'} {'code': '7', 'name': ''} {'code': '2', 'name': 'Public sector governance'} {'code': '2', 'name': 'Public sector governance'} {'code': '2', 'name': 'Public sector governance'} {'code': '10', 'name': 'Rural development'} {'code': '2', 'name': ''}

In [47]:

*'''* *code\_names are the list of the proper code names in order.*  *Note: The first list value is empty because there is no name for code: '0'.*  *Thus the first non-empty code name corresponds with code\_names[1].*  *'''* code\_names = [ "", 'Economic management', 'Public sector governance', 'Rule of law', 'Financial and private sector development', 'Trade and integration', 'Social protection and risk management', 'Social dev/gender/inclusion', 'Human development', 'Urban development', 'Rural development', 'Environment and natural resources management' ]

In [18]:

*# df\_copy will be the copy of the dataframe with the correct names* df\_copy = df.copy() *# Looks through each code name, and if it is empty, stores the proper name in its place* **for** x **in** range(len(df\_copy.mjtheme\_namecode)): **for** y **in** range(len(df\_copy.mjtheme\_namecode[x])): **if**(df\_copy.mjtheme\_namecode[x][y]['name'] == ""): df\_copy.mjtheme\_namecode[x][y]['name'] = code\_names[int(df\_copy.mjtheme\_namecode[x][y]['code'])]

In [19]:

*# Inspecting the code names of df\_copy, one see that none are empty.* *# limit will provide how many times it will print through the long output, and counter will increase until it hits limit.* limit = 25 counter = 0 **for** x **in** range(len(df\_copy.mjtheme\_namecode)): **for** y **in** range(len(df\_copy.mjtheme\_namecode[x])): print(df\_copy.mjtheme\_namecode[x][y]) counter += 1 **if**(counter >= limit): **break**

{'code': '8', 'name': 'Human development'} {'code': '11', 'name': 'Environment and natural resources management'} {'code': '1', 'name': 'Economic management'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '5', 'name': 'Trade and integration'} {'code': '2', 'name': 'Public sector governance'} {'code': '11', 'name': 'Environment and natural resources management'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '7', 'name': 'Social dev/gender/inclusion'} {'code': '7', 'name': 'Social dev/gender/inclusion'} {'code': '5', 'name': 'Trade and integration'} {'code': '4', 'name': 'Financial and private sector development'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '6', 'name': 'Social protection and risk management'} {'code': '2', 'name': 'Public sector governance'} {'code': '4', 'name': 'Financial and private sector development'} {'code': '11', 'name': 'Environment and natural resources management'} {'code': '8', 'name': 'Human development'} {'code': '10', 'name': 'Rural development'} {'code': '7', 'name': 'Social dev/gender/inclusion'} {'code': '2', 'name': 'Public sector governance'} {'code': '2', 'name': 'Public sector governance'} {'code': '2', 'name': 'Public sector governance'} {'code': '10', 'name': 'Rural development'} {'code': '2', 'name': 'Public sector governance'}