LESSON PLANS

LESSON #1

Aim: Data reading and selection part 1

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: What is a data science? https://www.youtube.com/watch?v=oMBE2TeH2no

Standards: 9-12. IC.7 Impacts of Computing, Career Paths

9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

Mini-Lesson:

1. Creating a data frame object
   Code Below: use a pandas constructor with a dictionary of list arguments
   Why pandas? Pandas offers several useful functions such as: aggregation, manipulation, and transformation.
2. Reading the Open Government Data from Eurostat
   a. Data downloaded and stored as: educ_figdp_1_Data.csv
   b. Download in same directory as Python. Open file, then read file
   c. Viewing top and bottom of data frame
   d. Viewing summary of the data frame
3. How to select a subset of data from the data frame?
   a. Selecting one column
   b. Selecting a subset of rows
   c. Selecting a subset of columns and rows
4. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

DATA FRAME OBJECT CODE

Import

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
data = {'year': [
         2010, 2011, 2012,
         2010, 2011, 2012,
         2010, 2011, 2012
     ],
     'team': [
         'FCBarcelona', 'FCBarcelona',
         'FCBarcelona', 'RMadrid',
         'RMadrid', 'RMadrid',
         'ValenciaCF', 'ValenciaCF',
         'ValenciaCF'
     ],
     'wins':   [30, 28, 32, 29, 32, 26, 21, 17, 19],
     'draws':  [6, 7, 4, 5, 4, 7, 8, 10, 8],
     'losses': [2, 3, 2, 4, 2, 5, 9, 11, 11]
     }
football = pd.DataFrame(data, columns = [
     'year', 'team', 'wins', 'draws', 'losses'
     ]
)
```

Output

	year	team	wins	draws	losses
0	2010	FCBarcelona	30	6	2
1	2011	FCBarcelona	28	7	3
2	2012	FCBarcelona	32	4	2
3	2010	RMadrid	29	5	4
4	2011	RMadrid	32	4	2
5	2012	RMadrid	26	7	5
6	2010	ValenciaCF	21	8	9
7	2011	ValenciaCF	17	10	11
8	2012	ValenciaCF	19	8	11

READING THE OPEN GOVERNMENT DATA

```
edu = pd.read_csv('files/ch02/educ_figdp_1_Data.csv',
                  na_values = ':',
                  usecols = ["TIME","GEO","Value"])
edu
```

Output

	TIME	GEO	Value
0	2000	European Union ...	NaN
1	2001	European Union ...	NaN
2	2002	European Union ...	5.00
3	2003	European Union ...	5.03
...	...	...	...
382	2010	Finland	6.85
383	2011	Finland	6.76

384 rows × 5 columns

Viewing top and bottom of data frame

```
edu.head()
```

	TIME	GEO	Value
0	2000	European Union ...	NaN
1	2001	European Union ...	NaN
2	2002	European Union ...	5.00
3	2003	European Union ...	5.03
4	2004	European Union ...	4.95

```
edu.tail()
```

379	2007	Finland	5.90
380	2008	Finland	6.10
381	2009	Finland	6.81
382	2010	Finland	6.85
383	2011	Finland	6.76

```
edu.describe()
```

	TIME	Value
count	384.000000	361.000000
mean	2005.500000	5.203989
std	3.456556	1.021694
min	2000.000000	2.880000
25%	2002.750000	4.620000
50%	2005.500000	5.060000
75%	2008.250000	5.660000
max	2011.000000	8.810000

```
Name: Value, dtype: float64
```

SELECTING DATA

Selecting one column

```
edu['Value']
```

```
0    NaN
1    NaN
2   5.00
3   5.03
4   4.95
...  ...
380 6.10
381 6.81
382 6.85
383 6.76
Name: Value, dtype: float64
```

Selecting a subset of rows or slicing

```
edu[10:14]
```

	TIME	GEO	Value
10	2010	European Union (28 countries)	5.41
11	2011	European Union (28 countries)	5.25
12	2000	European Union (27 countries)	4.91
13	2001	European Union (27 countries)	4.99

Selecting a subset of columns and rows

```
edu.ix[90:94, ['TIME','GEO']]
```

	TIME	GEO
90	2006	Belgium
91	2007	Belgium
92	2008	Belgium
93	2009	Belgium
94	2010	Belgium

LESSON #2

Aim: Data reading and selection part 2

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on at least two other datasets.

Do Now: Big data statistics? https://www.youtube.com/watch?v=GKLu8Oq5BXY

Standards: 9-12. IC.7 Impacts of Computing, Career Paths

9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.7 Computational Thinking, Algorithms, and Programming

Mini-Lesson:

1. Review yesterday's lessons
   a. Creating a data frame
   b. Reading data
   c. Selecting data
2. Student's will choose at least three datasets and/or websites on which they perform the same data procedures, while working in groups of 2s or 3s. Each group must choose dataset #1.
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

LESSON #3

Aim: Filtering data and handling missing data part 1

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: What is a data science? https://www.youtube.com/watch?v=X3paOmcrTjQ

Standards: 9-12. CT.7 Computational Thinking, Algorithms, and Programming

9-12. DL.5 Digital Literacy, Digital Use

Mini-Lesson:

1. Filtering data
   a. Filtering by using Boolean indexing
   b. Code example: filtering values less than or equal to 6.5
2. Handling missing data
   a. Ways of handling missing data
   b. Code example: filtering missing data
   c. Subtle feature of NaN and the use of isnull( )
3. Conclusion and summary
   Discussion:
   - » What are some of the problems or challenges you encountered?
   - » How did you resolve them?
   - » What did you learn from this lesson?
   - » Do you have any lingering questions on today's lesson or data science in general?

CODE

Filtering by using Boolean indexing

```
edu[edu['Value'] > 6.5].tail()
```

Output

	TIME	GEO	Value
218	2002	Cyprus	6.60
281	2005	Malta	6.58
94	2010	Belgium	6.58
93	2009	Belgium	6.57
95	2011	Belgium	6.55

Handing of missing data

```
edu[edu["Value"].isnull()].head()
```

	TIME	GEO	Value
0	2000	European Union (28 countries)	NaN
1	2001	European Union (28 countries)	NaN
36	2000	Euro area (18 countries)	NaN
37	2001	Euro area (18 countries)	NaN
48	2000	Euro area (17 countries)	NaN

LESSON #4

Aim: Filtering data and handling missing data part 2

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: Why data science? https://www.youtube.com/watch?v=a33rDTGfQ7M

Standards: 9-12. CT.7 Computational Thinking, Algorithms, and Programming

9-12. DL.5 Digital Literacy, Digital Use

Mini-Lesson:

1. Review yesterday's lessons
   a. Filtering data
   b. Handling missing data
2. Student's will use at least three datasets and/or websites on which they will perform the same data procedures illustrated in class yesterday, while working in groups of 2s or 3s. Each group must continue processing two of the three datasets selected on Monday. You may not use the first data instead you must select a new dataset in its place.
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

LESSON #5

Aim: Data reading and selection

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: Statistically Funny https://www.youtube.com/watch?v=UASWcDsH_nM

Standards: 9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.10 Computational Thinking, Algorithms, and Programming

Mini-Lesson:

1. Manipulating data
   a. List of most commonly used aggregation function

Function	Description
count()	Number of non-null observations
sum()	Sum of values
mean()	Mean of values
median()	Arithmetic median of values
min()	Minimum
max()	Maximum
prod()	Product of values
std()	Unbiased standard deviation
var()	Unbiased variance

   b. The above functions can either be applied on the rows for each column (axis = 0) or it should be applied on the columns of each row (axis = 1)
   c. The difference between pandas max function and standard Python max function
   d. Applying operations over all the values in rows, columns, or a selection or both. We can apply any binary arithmetical operations on an entire row.
   e. Applying any function to a data frame or series by just setting its name as an argument of the apply method. Example: applying the sqrt function from the Numpy library to perform the square root of each value in the Value column
   f. Using an in-line-function or λ-function
   g. Setting values in a data frame. Warning: Use a new column name!
   h. Dropping the column or row by using the drop function by setting axis = 1 or axis = 0, respectively. Note: If you do not want to keep the old data frame values set inplace = True

i. Inserting a row at the bottom of the data frame
j. Removing a row
k. The drop function and removing missing values
l. Removing missing values
m. Filling/replacing missing values

2. Sorting Data
   a. Sorting a column in a data frame by column in descending order
   b. Sorting a column in ascending order, while over writing the data frame
3. Conclusion and summary

   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

CODE

Data Manipulation

```
edu.max(axis = 0)
```

```
TIME          2011
GEO          Spain
Value         8.81
dtype: object
```

A Word of Note

```
print "Pandas max function:", edu['Value'].max()
print "Python max function:", max(edu['Value'])
```

```
Pandas max function: 8.81
Python max function: nan
```

Apply aggregate functions over all the values in rows, columns, or a selection of both

```
s = edu["Value"]/100
s.head()
```

```
0          NaN
1          NaN
2       0.0500
3       0.0503
4       0.0495
Name: Value, dtype: float64
```

Using the apply method

```
s = edu["Value"].apply(np.sqrt)
s.head()
```

```
0           NaN
1           NaN
2      2.236068
3      2.242766
4      2.224860
Name: Value, dtype: float64
```

Using an in-line-function or λ-function

```
s = edu["Value"].apply(lambda d: d**2)
s.head()
```

```
0           NaN
1           NaN
2       25.0000
3       25.3009
4       24.5025
Name: Value, dtype: float64
```

Setting new values in a data frame

```
edu['ValueNorm'] = edu['Value']/edu['Value'].max()
edu.tail()
```

	TIME	GEO	Value	ValueNorm
379	2007	Finland	5.90	0.669694
380	2008	Finland	6.10	0.692395
381	2009	Finland	6.81	0.772985
382	2010	Finland	6.85	0.777526
383	2011	Finland	6.76	0.767310

Deleting a column or row

```
edu.drop('ValueNorm', axis = 1, inplace = True)
edu.head()
```

	TIME	GEO	Value
0	2000	European Union (28 countries)	NaN
1	2001	European Union (28 countries)	NaN
2	2002	European Union (28 countries)	5
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95

Inserting a new row at the bottom of the data frame

```
edu = edu.append({"TIME": 2000,"Value": 5.00,"GEO": 'a'},
                 ignore_index = True)
edu.tail()
```

	TIME	GEO	Value
380	2008	Finland	6.1
381	2009	Finland	6.81
382	2010	Finland	6.85
383	2011	Finland	6.76
384	2000	a	5

Removing a row

```
edu.drop(max(edu.index), axis = 0, inplace = True)
edu.tail()
```

	TIME	GEO	Value
379	2007	Finland	5.9
380	2008	Finland	6.1
381	2009	Finland	6.81
382	2010	Finland	6.85
383	2011	Finland	6.76

The drop function and removing missing values

```
eduDrop = edu.drop(edu["Value"].isnull(), axis = 0)
eduDrop.head()
```

	TIME	GEO	Value
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
5	2005	European Union (28 countries)	4.92
6	2006	European Union (28 countries)	4.91

Removing missing values

```
eduDrop = edu.dropna(how = 'any', subset = ["Value"])
eduDrop.head()
```

	TIME	GEO	Value
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	5.03
4	2004	European Union (28 countries)	4.95
5	2005	European Union (28 countries)	4.92
6	2006	European Union (28 countries)	4.91

Filling/replacing missing values

```
eduFilled = edu.fillna(value = {"Value": 0})
eduFilled.head()
```

	TIME	GEO	Value
0	2000	European Union (28 countries)	0.00
1	2001	European Union (28 countries)	0.00
2	2002	European Union (28 countries)	5.00
3	2003	European Union (28 countries)	4.95
4	2004	European Union (28 countries)	4.95

Sorting Data

Sorting a column in a data frame by column in descending order

```
edu.sort_values(by = 'Value', ascending = False,
                inplace = True)
edu.head()
```

	TIME	GEO	Value
130	2010	Denmark	8.81
131	2011	Denmark	8.75
129	2009	Denmark	8.74
121	2001	Denmark	8.44
122	2002	Denmark	8.44

Sorting a column in ascending order, while over writing the data frame

```
edu.sort_index(axis = 0, ascending = True, inplace = True)
edu.head()
```

	TIME	GEO	Value
0	2000	European Union ...	NaN
1	2001	European Union ...	NaN
2	2002	European Union ...	5.00
3	2003	European Union ...	5.03
4	2004	European Union ...	4.95

LESSON #6

Aim: Data manipulation and sorting part 2

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: Who uses student data? https://www.youtube.com/watch?v=v1uj0JkCpgM

Standards: 9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.10 Computational Thinking, Algorithms, and Programming

Mini-Lesson:

1. Review yesterday's lessons
   a. Filtering data.
   b. Handling Missing data
2. Student's will use at least two datasets and/or websites on which they will perform the same data procedures illustrated in class on Friday, while working in groups of 2s or 3s. Each group must continue processing two of the datasets you selected last Monday. You must use the first dataset and at least one from your earlier choice.
3. Conclusion and summary:
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

LESSON #7

Aim: Grouping and rearranging data part 1

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: Why is data science important? Statistics to make you think! [0 – 3.5 minutes]

https://www.youtube.com/watch?v=mGEm3oT32BA

Standards: 9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.10 Computational Thinking, Algorithms, and Programming

9-12. CT.7 Computational Thinking, Algorithms, and Programming

9-12. DL.5 Digital Literacy, Digital Use

Mini-Lesson:

1. Grouping data
   a. Grouping according to a criterion and aggregating
2. Rearranging data
   a. Transforming the arrangement of data, redistributing the indexes and columns for better manipulation of the data. Example: specifying which columns will be he indexes, the new values and the new columns
   b. Using the new index to specify specific rows
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

CODE

Grouping according to a criterion and aggregating

```
group = edu[["GEO", "Value"]].groupby('GEO').mean()
group.head()
```

	Value
GEO	
Austria	5.618333
Belgium	6.189091
Bulgaria	4.093333
Cyprus	7.023333
Czech Republic	4.16833

Rearranging Data

Transforming the arrangement of data, redistributing the indexes and columns for better manipulation of the data. Example: specifying which columns will be he indexes, the new values and the new columns

```
filtered_data = edu[edu["TIME"] > 2005]
pivedu = pd.pivot_table(filtered_data, values = 'Value',
                        index = ['GEO'],
                        columns = ['TIME'])
pivedu.head()
```

TIME	2006	2007	2008	2009	2010	2011
GEO						
Austria	5.40	5.33	5.47	5.98	5.91	5.80
Belgium	5.98	6.00	6.43	6.57	6.58	6.55
Bulgaria	4.04	3.88	4.44	4.58	4.10	3.82
Cyprus	7.02	6.95	7.45	7.98	7.92	7.87
Czech Republic	4.42	4.05	3.92	4.36	4.25	4.51

Using the new index to specify specific rows

```
pivedu.ix[['Spain','Portugal'], [2006,2011]]
```

TIME	2006	2011
GEO		
Spain	4.26	4.82
Portugal	5.07	5.27

LESSON #8

Aim: Grouping and rearranging data part 2

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: What is data mining and why is it important?

https://www.youtube.com/watch?v=mOfPG5ZIY-k

Standards: 9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.10 Computational Thinking, Algorithms, and Programming

9-12. CT. 7 Computational Thinking, Algorithms, and Programming

9-12. DL. 5 Digital Literacy, Digital Use

Mini-Lesson:

1. Review yesterday's lessons
   a. Grouping data
   b. Rearranging data
2. Student's will use at least three datasets and/or websites on which they will perform the same data procedures illustrated in class on Friday, while working in groups of 2s or 3s. Each group must continue processing the datasets they selected last Thursday of this week. You may not use the first dataset instead you must select another dataset in its place.
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

LESSON #9

Aim: Ranking and plotting data part 1

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: Why is data science important? Statistics to make you think! [4 – 8:10 minutes]

https://www.youtube.com/watch?v=Qck5Ae3F3RY&list=RDUASWcDsH_nM&index=20

Standards: 9-12. IC.7 Computational Thinking, Algorithms, and Programming

9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. DL.5 Digital Literacy, Digital Use

Mini-Lesson:

1. Ranking data
   a. Cleanup previous pivot table entries
   b. Drop the Euro area entries and shorten the Germany name entry by using the rename function
   c. Drop all rows containing the NaN by using dropna function.
   d. Use the rank function by setting ascending to false – highest to lowest
   e. To make global ranking across all years – sum all columns and rank the result

   "dense" – items that compare equals get same ranking and the next not equal item receives the immediately following ranking

2. Plotting Data
   a. Plotting the accumulated values for each country over the last six years by taking the Series obtained in the last example and plot it directly
   b. Plotting a data frame directly – treat columns as a separate Series by plotting the value for each year
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

CODE AND OUTPUT

Ranking Data

```
pivedu = pivedu.drop([
                      'Euro area (13 countries)',
                      'Euro area (15 countries)',
                      'Euro area (17 countries)',
                      'Euro area (18 countries)',
                      'European Union (25 countries)',
                      'European Union (27 countries)',
                      'European Union (28 countries)'
                     ],
                     axis = 0)
pivedu = pivedu.rename(index = {'Germany (until 1990 former territory
    of the FRG)': 'Germany'})
pivedu = pivedu.dropna()
pivedu.rank(ascending = False, method = 'first').head()
```

Output

TIME	2006	2007	2008	2009	2010	2011
GEO						
Austria	10	7	11	7	8	8
Belgium	5	4	3	4	5	5
Bulgaria	21	21	20	20	22	21
Cyprus	2	2	2	2	2	3
Czech Republic	19	20	21	21	20	18

To make global ranking across all years

Sum all columns and rank the result

```
totalSum = pivedu.sum(axis = 1)
totalSum.rank(ascending = False, method = 'dense')
        .sort_values().head()
```

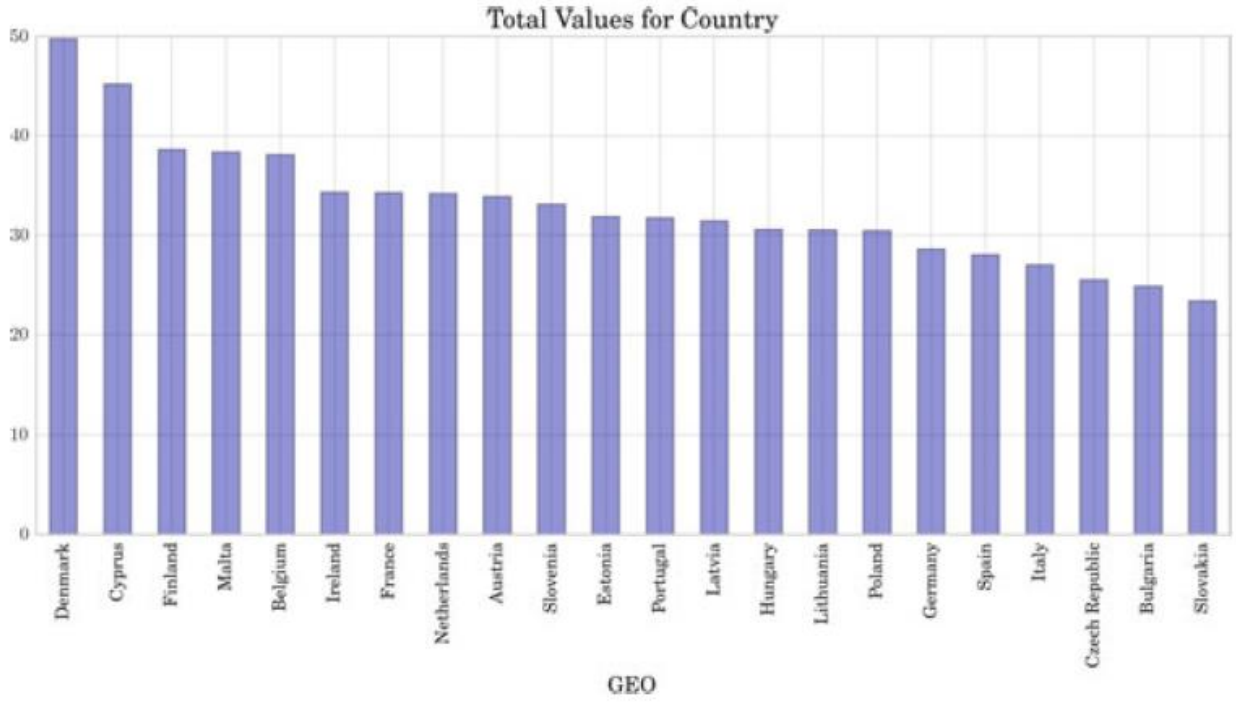
```
GEO
Denmark                1
Cyprus                 2
Finland                3
Malta                  4
Belgium                5
dtype: float64
```

Plotting

Plotting the accumulated values for each country over the last six years by taking the series obtained in the last example and plot it directly.

```
totalSum = pivedu.sum(axis = 1)
                 .sort_values(ascending = False)
totalSum.plot(kind = 'bar', style = 'b', alpha = 0.4,
              title = "Total Values for Country")
```

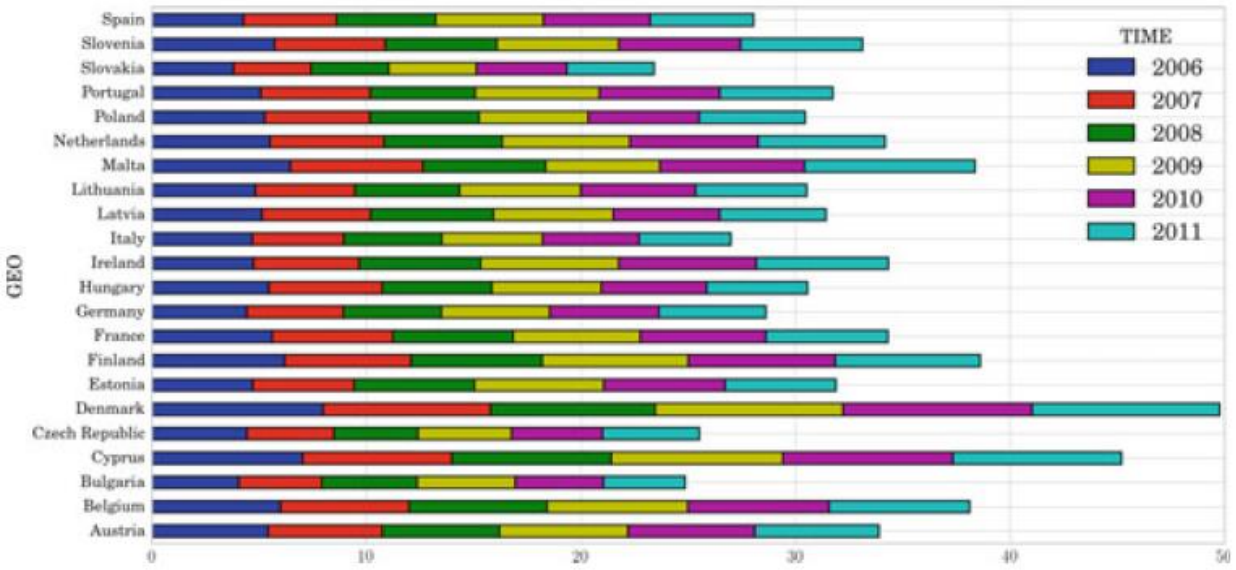
Output



Plotting a data frame directly

Treat columns as a separate series and plot the value for reach year

```
my_colors = ['b', 'r', 'g', 'y', 'm', 'c']
ax = pivedu.plot(kind = 'barh',
                 stacked = True,
                 color = my_colors)
ax.legend(loc = 'center left', bbox_to_anchor = (1, .5))
```



LESSON #10

Aim: Ranking and plotting data part 2

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: What is a data science? https://www.youtube.com/watch?v=X3paOmcrTjQ

Standards: 9-12. IC.7 Computational Thinking, Algorithms, and Programming

9-12. CT.2 Computational Thinking, Data Analysis, and Visualization

9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. DL.5 Digital Literacy, Digital Use

Mini-Lesson:

1. Review yesterday's lessons
   a. Grouping data
   b. Rearranging data
2. Student's will use at least three datasets and/or websites on which they will perform the same data procedures illustrated in class on Friday, while working in groups of 2s or 3s. Each group must continue processing the datasets they selected on Wednesday of this week. You may not use the first dataset; instead, you must select another dataset in its place.
   Note: Play with the "kind" parameter in your plot function.
3. Conclusion and summary
   Discussion:
   - What are some of the problems or challenges you encountered?
   - How did you resolve them?
   - What did you learn from this lesson?
   - Do you have any lingering questions on today's lesson or data science in general?

LESSON #11

Aim: Data preparation

Objective: After a lesson on reading and selection of data, students will be able to use Python to perform the same procedures on two other datasets.

Do Now: What is a data science? https://www.youtube.com/watch?v=X3paOmcrTjQ

Standards: 9-12. CT.3 Computational Thinking, Data Analysis, and Visualization

9-12. CT.7 Computational Thinking, Algorithms, and Programming

9-12. CT.10 Computational Thinking, Algorithms, and Programming

9-12. DL.5 Digital Literacy, Digital Use