Monash University Malaysia

School of IT

FIT 3179: Data Visualization

Should a country prioritise investing in education?

17 October 2018

Tan Wei Shen

Project visualization:
https://public.tableau.com/profile/tan.wei.shen#!/vizhome/Asgn-Mystory/Shouldacountryprioritiseinvestingineducation

Contents

Domain

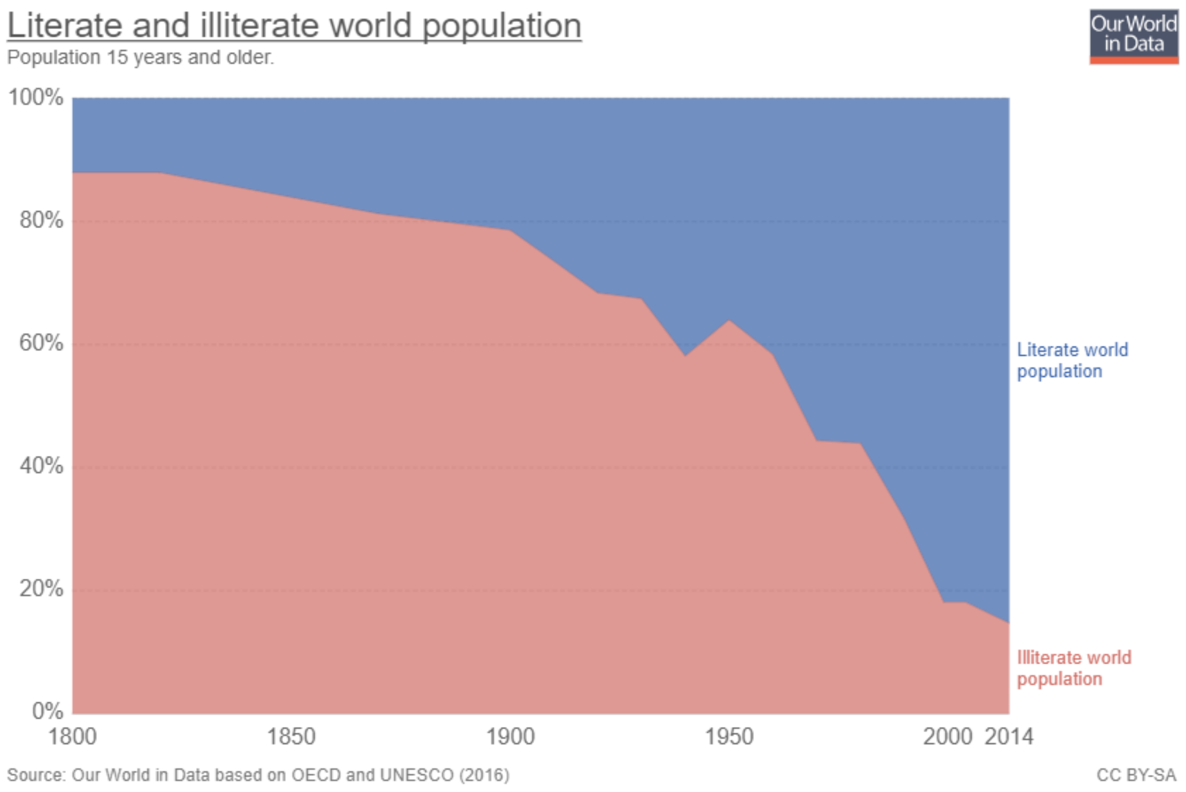
The project domain that I have selected is education. Specifically, my visualization addresses the importance of education and its affect on a country. The motivation of the visualizations is to ultimately convince the government to invest more resources into the education sector. The visualizations achieve this by highlighting its role in positively influencing various factors that lead to the success of a country. It is worth mentioning that these factors are not randomly selected factors, but instead legitimate factors that were taken into consideration when ranking the success of countries across the world. This ranking was done by U.S. News [1]. Thus, the target audience for this project visualization are the government bodies, and those authority figures who have control over the government expenditure of a country.

Related Visualizations

We will first be analyzing the current existing visualizations related to our domain. We will identify the message that the visualization is trying convey. We will also judge various elements of the visualization that have been proven to significantly impact the efficiency and effectiveness of a visualization.

1. World population literacy

Taken from: https://ourworldindata.org/global-rise-of-education#literacy



The visualization above tells the story of the change in world population literacy over time. The visualization has good interactivity. This is because hovering over the visualization will produce a pop up displaying the percentage of literates and illiterates as well as the year at that point.

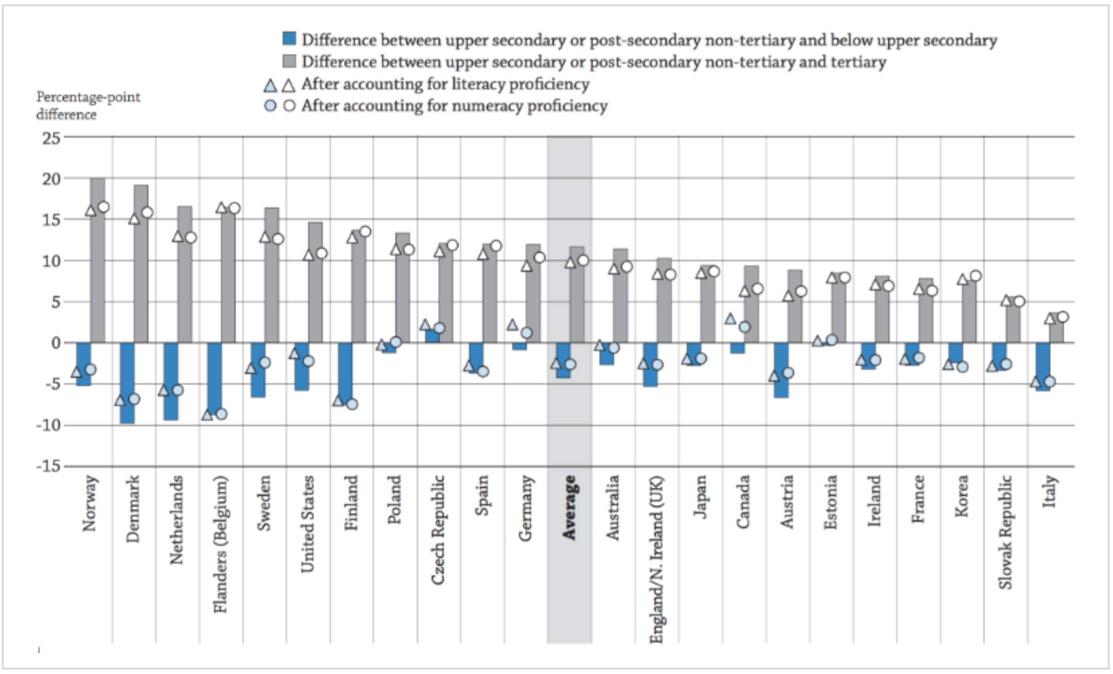
The color elements of the visualization is good as red tends to signify negativity and blue tends to signify positivity.

However, the visualization is inefficient as it has a low data-ink ratio. Instead of using an area chart, the visualization could convey the same message by using a line chart.

2. Correlation between education and trust in others.

Taken from: *https://ourworldindata.org/global-rise-of-education#social-returns-to-education*

Likelihood of reporting to trust others, by educational attainment, OECD 2012 – Figure A8.4 in Education at a Glance (2015)[22]



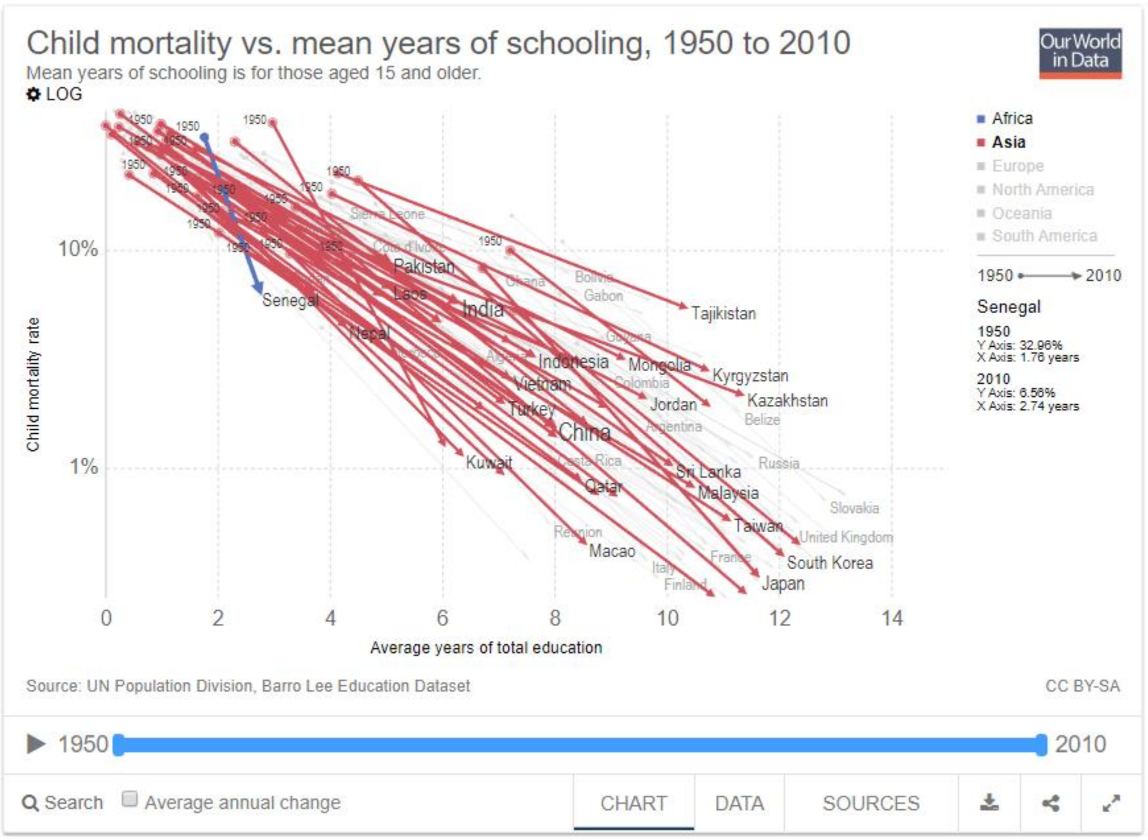
The above visualization tells the story that individuals who are more educated tend to trust people more.

The visualization is very informative and efficient as it has maximised its data-ink ratio.

However, the visualization does not have any interactivity whatsoever. Besides that, the color selection for the bars are questionable. It would be better if they used a diverging color (red-blue or red-green) for different comparisons of education levels.

3. Years of schooling vs Child mortality rate

Taken from: https://ourworldindata.org/global-rise-of-education#social-returns-to-education



The visualization above shows the correlation between years of schooling and child mortality and attempts to show that a society that is more educated tends to have lower child mortality rates.

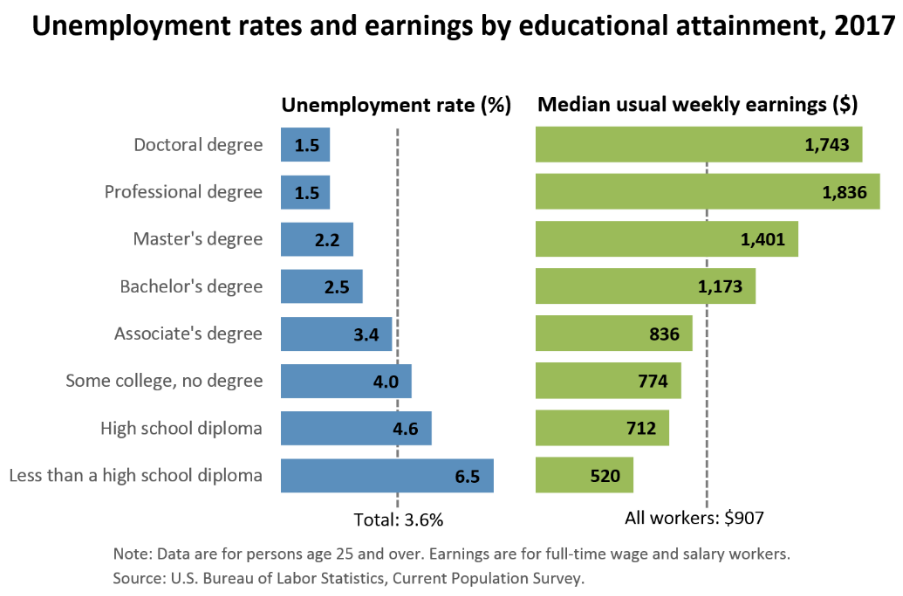
The scatter plot idiom is a good choice for this visualization as it is trying to show the correlation between two quantitative attributes.

It has a poor filtering system as countries that are not selected by the filter are not hidden, they are merely faded. This makes the visualization really messy as you can still see the lines of unselected countries.

The visualization also has poor interactivity. Hovering over a line does not display an annotation or any relevant information about that line. Instead, it merely highlights the line in an attempt to make it distinct.

4. Unemployment rates and earnings by educational attainment

Taken from: https://www.bls.gov/emp/chart-unemployment-earnings-education.htm



The above visualization tells the story that invidividuals who are more educated tend to have a higher chance of getting a job as well as higher pay.

The visualization is able to convey the information clearly. However, it has a relatively low data-ink ratio as it uses a bar chart. Ultimately, the goal of the visualization is to show that the employment rate and earnings increases as the educational attainment level increases. Thus, it would be much better fitted to use a line chart. The line chart will be able to show the trendlines clearly whilst maximizing the data-ink ratio.

The visualization also has no interactivity at all. It would be better if there was some interactivity such as a filter for the educational attainment level.

Project Design

As mentioned previously, the project visualization aims to answer the question of whether a country should make education a priority investment. However, before we can answer that question, we have to first answer the question of why does a country invest in anything? The answer to this question is that a country invests in something to ultimately become more successful.

Therefore, in order to determine whether education is a priority investment, we have to identify the factors that contribute to a country's success. If we find that the education sector positively influences these key factors, then we can conclude that education is indeed a sector that is worth heavily investing in.

Thus, the project visualization goes over various key factors that determine the success of a country and shows how education impacts these factors. The factors that are used in the visualization include:

- Quality of Life
- Citizenship
- Entrepreneurship
- Open for business

These factors were taken from the U.S. News webiste [1]. In each of these factors, there are subcategories that further describe these factors. As there are too many subcategories per factor, we handpicked only a few of them for our visualization.

Below is a screenshot of our Five Design Sheet Methodology:

Sheet 1

Ideas

- Introduction – current status of education
- Why Invest in anything?
  ↳ To be SUCCESSFUL

SO.. What makes a country SUCCESSFUL?

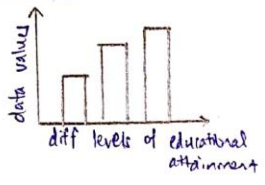
↓ poverty ↓ employment | ↓ corruption ↓ citizenship | ↓ economy | ↓ healthcare ↓ advancement

choropleth map for quantitative attributes by country

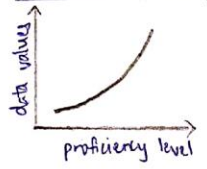
Datasets

1. Corruption Perception Index vs. years of schooling
2. poverty rates vs years of schooling
3. % of NEETS by proficiency level ~~Education level~~
4. trust level in others by ed level
5. Frequency of ICT use by ed level
6. Awareness of greenhouse gases by ed level.
7. child mortality rate vs. years of schooling

Bar graph – use to compare



Line graph – use to show trend



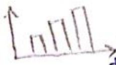
Filter.

Datasets 1,3,4,5,6

U.S. News uses certain factors to determine ranking of best countries:

- Citizenship
  ↳ cares ~~about~~ about the environment (6)
  ↳ trustworthy (4)
- Quality of Life
  ↳ job market and employment rate (3)
- Open for business
  ↳ corruption rate (1)
- Entrepreneurship
  ↳ Technical expertise (5)

Categorise



Use bar chart to compare values between diff levels of educational attainment.

Use line chart to show increase/decrease of values based on proficiency level.

- Use for dataset ~~dset~~ (3)
  ~~[illegible] NEETS [illegible]~~
  ↳ NEETS % ↓ as literacy proficiency ↑

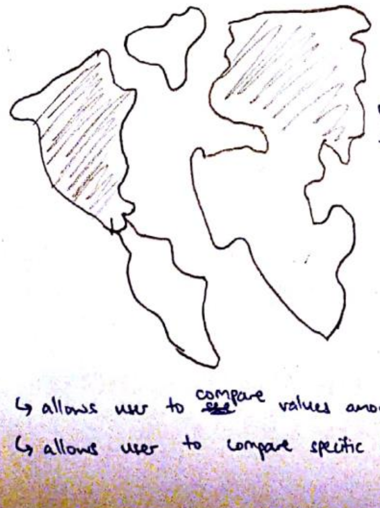
scatterplot to show correlation.

For dataset (1).

- Use diff colours for marks to represent countries.

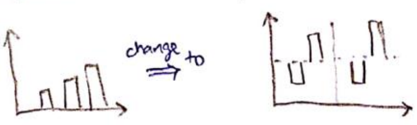
Combine & Refine

Use a choropleth map to filter ~~the~~ other charts.



Use this to filter ⟹

↳ allows user to ~~see~~ compare values among countries
↳ allows user to compare specific categories in a country.



When comparing between different levels of education, use diverging bar chart to show difference.

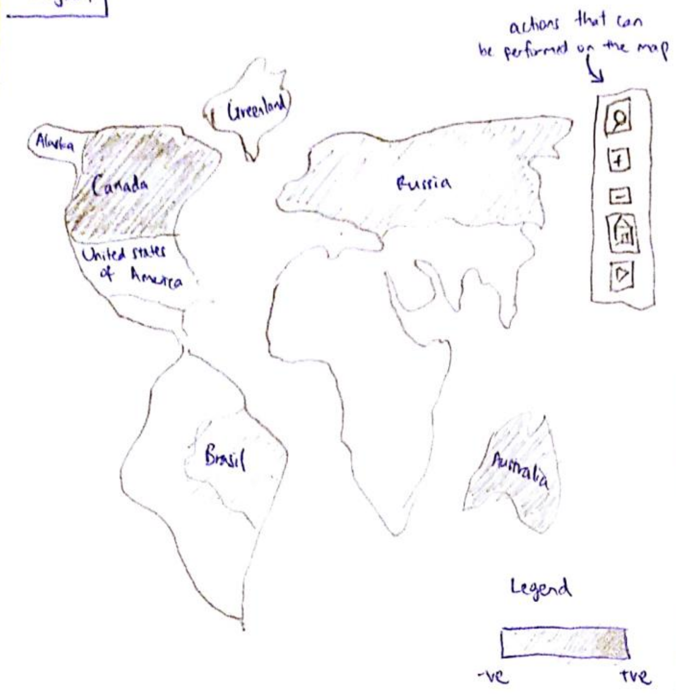
↳ higher data-ink ratio.
↳ clearer comparison.

Question

Should a country prioritise investing in education?

Sheet 2

Layout



Title : Interactive choropleth Map

Author : TAN WEI SHEN

Date : 17 Oct 2018

Sheet : 2

Task : Should a country ~~invest~~ prioritise investing in education

Operations

- User has initial view as the entire world map.
- Clicking on a country will highlight that section of the map and display relevant details.
- Hovering over a country will pop up an annotation displaying relevant details.

ACTIONS TAB — see FOCUS section

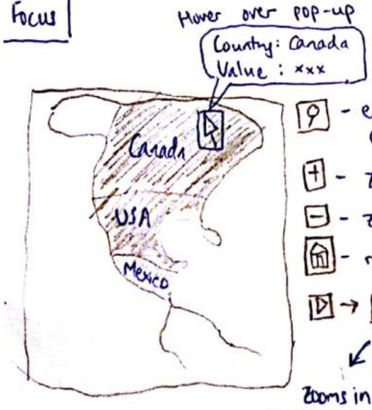
↳ can search for a country's location on the map by entering its name with 🔍 icon.

↳ Can pan the map by clicking and dragging the mouse with ✥ icon.

↳ Can perform different kinds of selection by using ⬚, ◌, ⬚ icons.

↳ Similar actions can be done in ACTIONS TAB. ~~See FOCUS section~~

Focus



🔍 - enter a country name
↳ Map will zoom to that country

⊞ - Zoom in

⊟ - Zoom out

⌂ - return to default map view

▷ → ☐ ✥ ⬚ ◌ ⬚

Zooms in on selected area

selects countries in a selected area

Discussion

Pros :

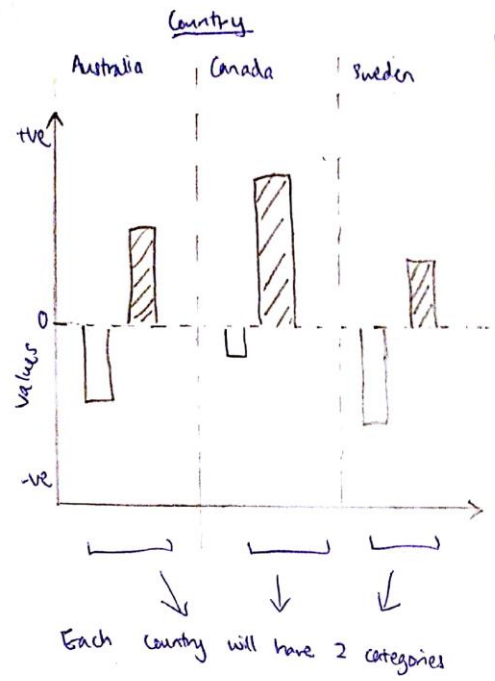
+ Good interactivity provided to allow users to clearly understand the map.
+ Choropleth map clearly shows intensity of values among countries which allow users to easily compare different countries.
+ annotations allow users to view exact details and values.

Cons :

- Exact values are not clearly shown by just looking at the map.
  ↳ Users will have to hover over or select a particular country to see the exact value.

Sheet 3

Layout



Legend

▨ Difference between tertiary and upper secondary education

☐ Difference between upper and below-upper secondary education

Filter - Country

☐ (All)

☑ Australia

☐ Austria

☑ Canada

☑ Sweden

. . . .

Title : Diverging bar chart

Author : TAN WEI SHEN

Date : 17 Oct 2018

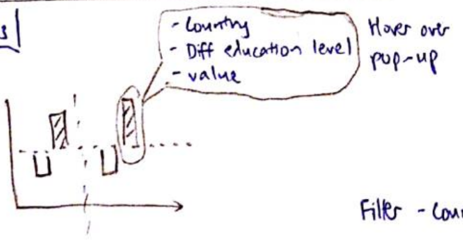
Sheet : 3

Task : should a country prioritise investing in ~~educat~~ education?

Operations

- Clicking on a particular bar will highlight that bar.
- Hovering over any bar will show the relevant details of the bar. Ex:
  - ↳ Country
  - ↳ Diff in education level
  - ↳ value
- Clicking on the legend will highlight bars corresponding to that legend.
- Checking/unchecking the filter checkbox will hide/show the data corresponding to that country.

Focus



Filter - Country

☐ (All)

☐ Australia

☒ Austria

☐ Canada

. . .

↳ on click
hides/shows the bar for that country.

Legend

▨ Diff between tertiary & upper-secondary

☐ Diff between upper & below-upper secondary

↓ on click
highlights the bars in that category

Discussion

Pros :

+ Good interactivity provided to allow users to clearly understand the bar chart.
+ diverging bar chart shows difference in different education levels.
  ↳ allows users to easily compare.
+ high data-ink ratio.

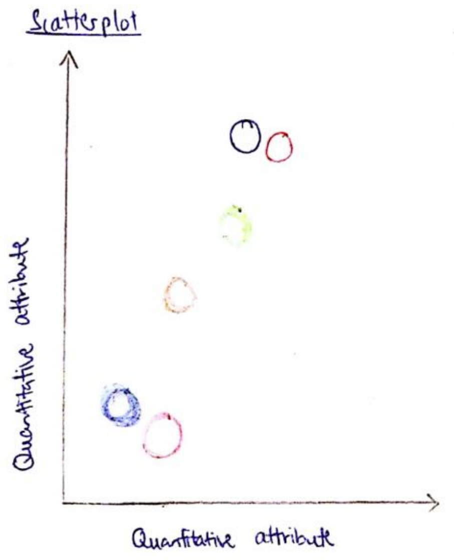
Cons :

- Exact values are not clearly shown.
  ↳ users will have to hover over a bar chart to see the exact values.

Sheet 4

Title : Interactive scatterplot
Author : TAN WEI SHEN
Date : 17 Oct 2018
Sheet : 4
Task : Should a country prioritise investing in education ?

Layout



Legend - Country

- 🟩 Australia
- 🟥 Canada
- 🟦 Portugal
- 🟫 Sweden
- 🟥 Brazil
- 🟦 England
- . . . .

Use scatterplot to show correlation :

↳ Corruption rate (x-axis) vs years of schooling (y-axis)

Operations

- User has initial view of all the marks on the scatter plot
- Clicking on a mark will highlight it
- Clicking on a country in the Legends will highlight the mark in the scatterplot
- Hovering over a mark will pop up an annotation displaying relevant details

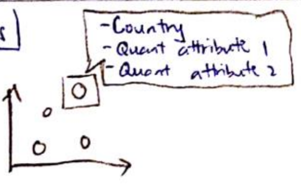
Ex: ↳ Country
↳ Corruption Perception Index
↳ Total years of schooling

↓

Example for the case of corruption vs education.

- Users can select multiple marks by clicking and dragging over the marks.
  ↳ This will highlight all selected marks.

Focus



Hover over pop-up

Example:

- Russia
- Corruption Perception Index : 28.00
- Total years of schooling : 11.77

Legend

- ▨ England
- 🟥 Brazil
- . . . .

on click → Highlights the particular mark on the ~~sc~~ scatterplot.

~~Discussion~~ Discussion

Pros :

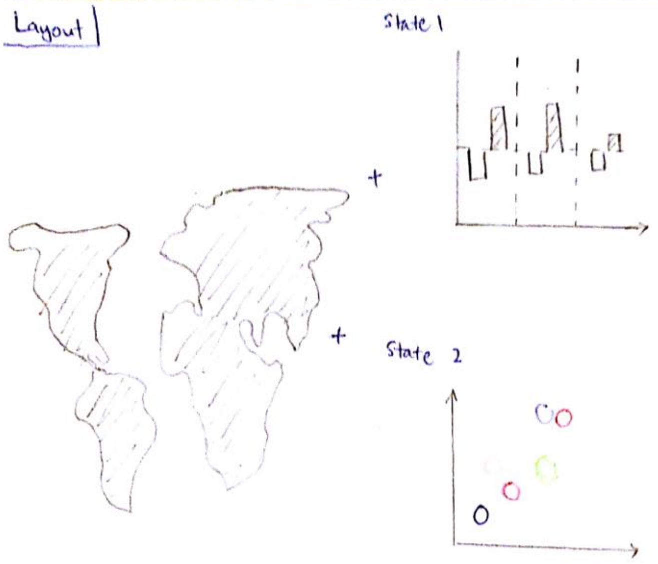
- + Good interactivity provided to allow users to understand the scatterplot.
- + scatterplot shows correlation between 2 quantitative attributes.
- + annotations allow user to clearly view exact details and values.
- + ~~Filters~~ Interactivity with legend helps identify marks clearly as there are a lot of marks.

Cons :

- Exact values are not clearly shown on the scatterplot.
  ↳ Users have to hover over marks to get exact details.

Sheet 5

Layout



↳ Use choropleth map in combination with scatterplot / bar charts.

- Choropleth map allows comparison between countries.
- barchart / scatterplot show ~~exact~~ specific details in a particular country.

↳ choropleth map can ~~acts~~ act as a filter for bar chart / scatterplots.

Title : Interactive map, bar chart, scatterplot (Realization)

Author : TAN WEI SHEN

Date : 17 Oct 2018

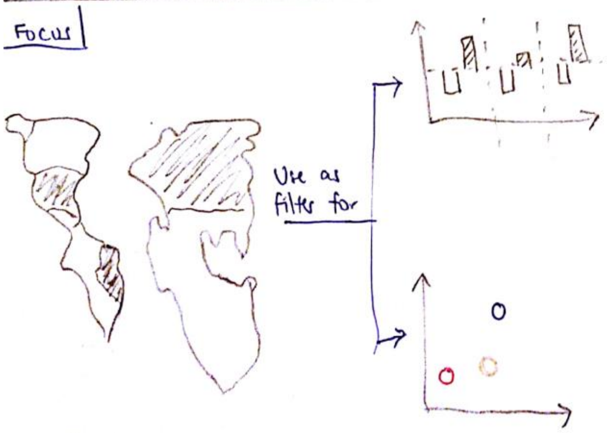
Sheet : 5

Task : Should a country prioritise investing in education ?

Operations

- User has initial view as the entire world map on the left ~~hand~~ and all unfiltered values on the right side represented by bar chart / scatterplot.
- Hovering over any of the marks will display a pop up annotation with relevant details.
- Clicking on any ~~section~~ location on the map will filter the corresponding values ~~is~~ being displayed in the bar chart / scatterplot on the right.

Focus



With the choropleth map on the left, and the scatterplot / bar chart on the right, we use the map as a filter. Clicking on a location ~~the~~ on the map, will filter the countries ~~to~~ being displayed on the bar chart / scatterplot.

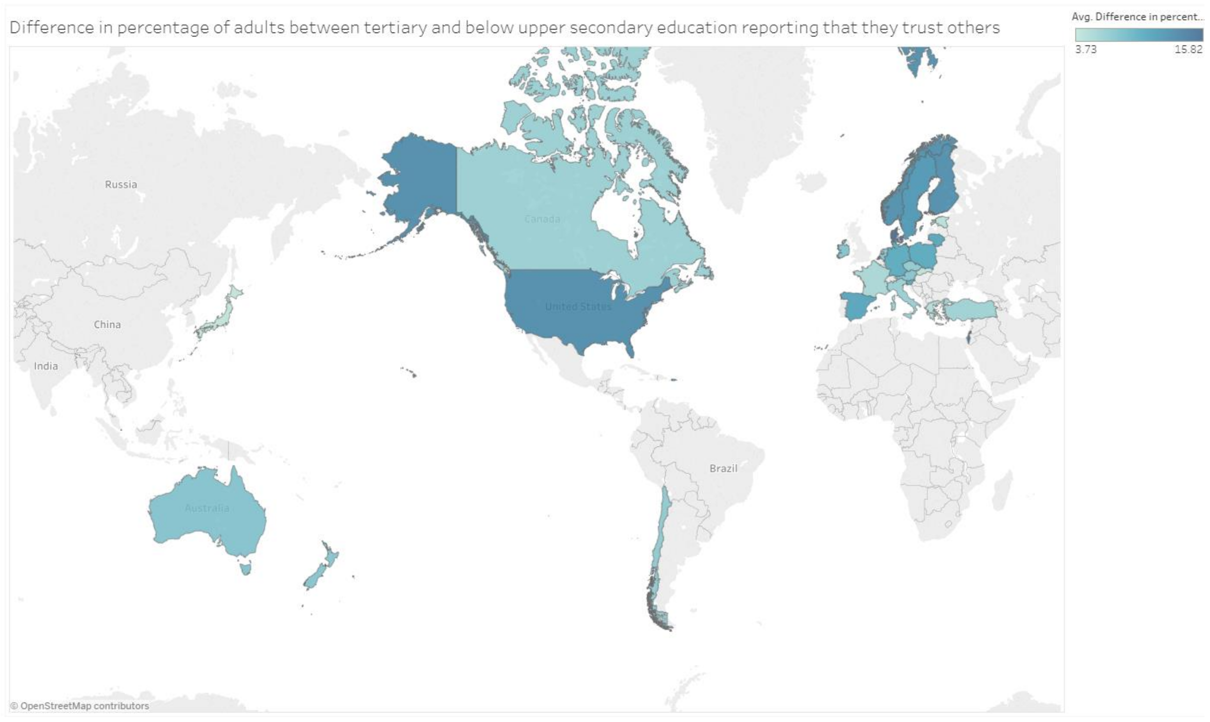
~~Discuss~~ Detail

- Time to build estimated at 16 hours.
- Datasets acquired from OECD and Our World in Data
- Most values will be normalized to % differences.
- Use red-blue diverging colour when representing certain quantitative attributes
  - ↳ Corruption rate
  - ↳ trust levels
  - ↳ awareness of greenhouse gases.
  - ... etc.
- Datasets will require some cleaning up and transformation using ~~hand written~~ self-deve scripts.

Project Visualization

As the project visualization consists of many similar visualizations, we will only review in detail each unique visualization as the rest will carry a similar format.

Viz 1 – Percentage difference of trust levels in others between Tertiary and below upper secondary education by country


Difference in percentage of adults between tertiary and below upper secondary education reporting that they trust others
Avg. Difference in percent..
3.73
15.82
Russia
China
India
Canada
United States
Brazil
Australia
© OpenStreetMap contributors

Idiom: Choropleth map

What

Attribute type: Quantitative – Trust levels in others

Dataset type: Geometry (Spatial)

Why

Shows the percentage difference of trust levels in others between tertiary and below upper secondary education by country. From the visualization, we can see that it is a global trend that people with higher levels of education tend to trust others more. With this visualization, users will be able to compare different countries by how much people with higher levels of education trust others.

How

Marks: Area of the map

Channel: Colour intensity – percentage difference in trust levels in others

Advantages

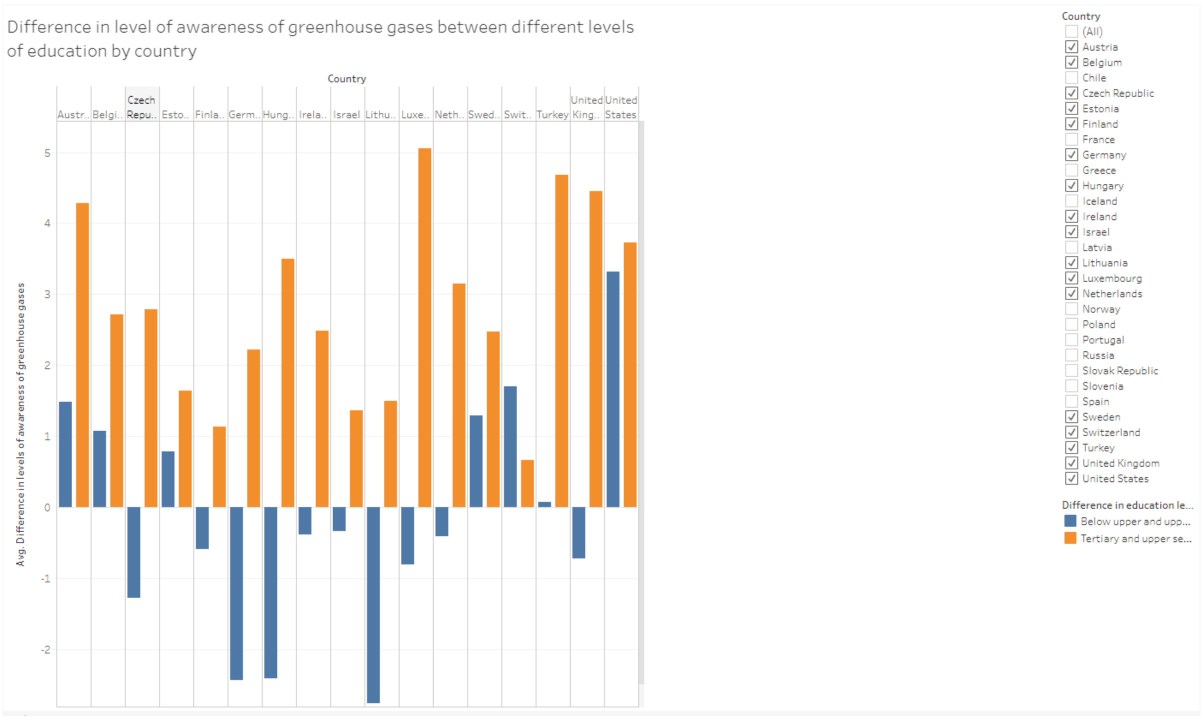
- The color element used in the visualization is a red-blue diverging colour. Because the trend that a more educated individual tends to trust others more exists in every country, we do not see the red color element anywhere. Thus, the color element used clearly shows that it is a global trend that people trust others more as they get more educated.

- Good interactivity.
  - Clicking on a location in the map will highlight that section and pop up an annotation displaying relevant information.
  - Hovering over a location will also pop up an annotation.
- Good figure-ground. Selected countries are highlighted with colour while unselected countries will be displayed as a faded gray.

Disadvantages

- Usage of choropleth map means that the user cannot clearly distinguish values between different countries. Instead, the user will have to hover over the country in order to see the exact values.

Viz 2 – Percentage difference of awareness of greenhouses gases between different levels of educational attainment by country



Idiom: Diverging bar chart

What

Attribute type: Categorical – Country & Different levels of education

Dataset type: Table

- Rows: Difference in percentage of trust levels in others
- Column: Countries & Difference in education levels

Why

Shows the percentage difference of awareness of greenhouse gases between different levels of educational attainment by country. From the visualization, we can see that it is a global trend that people who are more educated tend to have more awareness of greenhouse gases. With this visualization, users will be able to compare the specific differences in awareness of greenhouse gases by educational attainment level.

How

Marks: Bars

Channel: Height of the bar – percentage difference of awareness of greenhouse gases

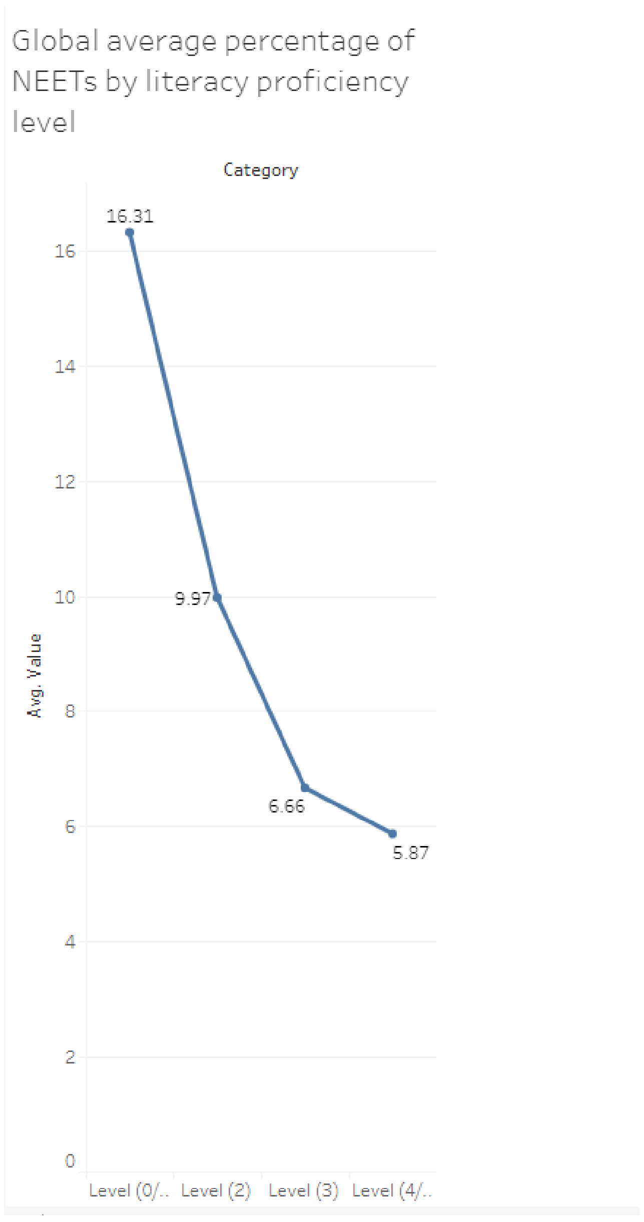
Advantages

- Informative and efficient with high data-ink ratio. Diverging bar chart communicates the information clearly whilst maximising data-ink ratio.
- Good interactivity.
    - Selected bars will be highlighted with saturated colours while unselected bars will be faded with unsaturated colours.
    - Hovering over any bar will display a pop up annotation with relevant information.
- Color element used can be perceived by color blinded individuals. It also clearly distinguishes between different levels of education.
- Good figure-ground. Selected bars are highlighted with saturated colours while unselected bars will be displayed with unsaturated colours.

Disadvantages

- The visualization only works well by using the filter to select particular countries. If it is completely unfiltered, there will be too many countries being displayed. This will make the visualization messy as it is overwhelmed with data.

Viz 3 – Global average percentage of NEETs by literacy proficiency



Idiom: Line chart

What

Attribute type: Ordered ordinal – Level of literacy proficiency

Dataset type: Table

- Rows: Average percentage of NEETs
- Columns: Literacy proficiency level

Why

Shows the global average percentage of NEETs by literacy proficiency level. From the visualization, we can see that the average percentage of NEETs decreases as the literacy proficiency level increases. This implies that the more educated an individual is, the lower the chances that the individual will end up unemployed.

How

Marks: Points and line connections

Channels: Vertical position of the points – Average percentage of NEETs

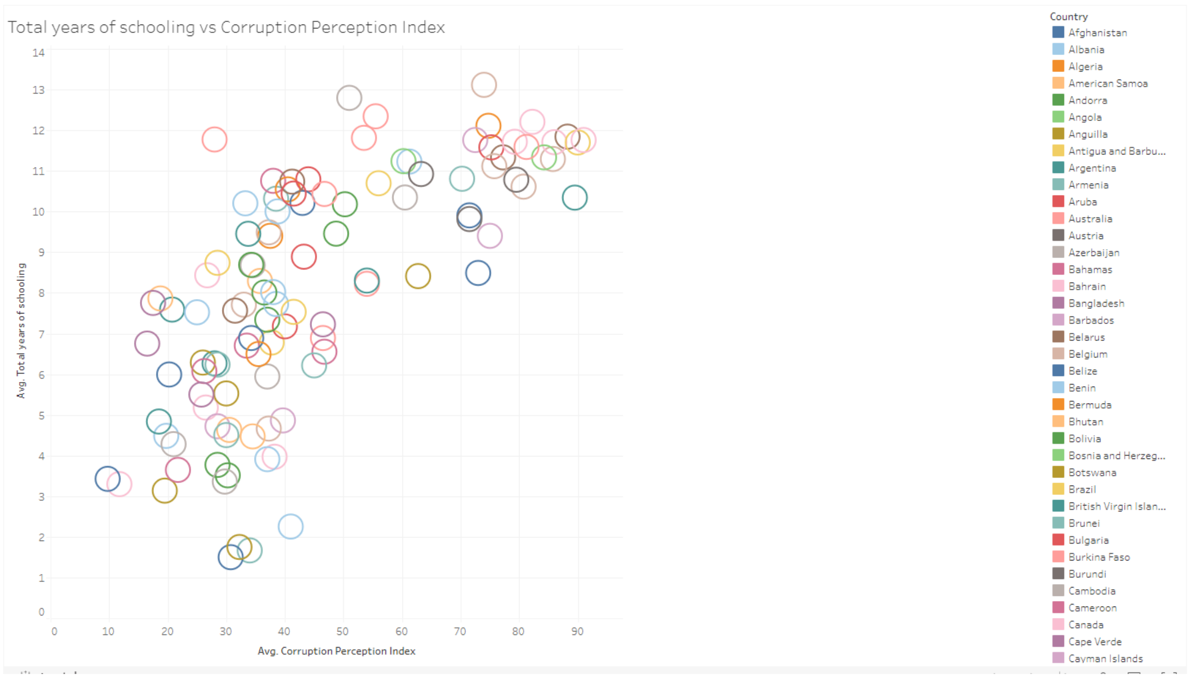
Advantages

- Informative and efficient with high data-ink ratio. The line chart idiom clearly shows the trend whilst minimising chart junk.
- Good interactivity. Hovering over any point in the visualization will display a pop up annotation with relevant information.
- Annotated points on the line chart allows user to clearly identify the values at each literacy level.

Disadvantages

- Lacks attractiveness as it is just a simple line chart. However, according to Edward Tufte, adding ornamental elements to make a visualization attractive is unnecessary and redundant.

Viz 4 – Corruption Perception Index (CPI) vs Total years of schooling



Idiom: Scatterplot

What

Attribute type: Quantitative – Corruption Perception Index (CPI) & total years of schooling

Dataset type: Table

- Rows: Total years of schooling
- Column: Corruption Perception Index

Why

It is important to know that the higher the CPI, the less corrupted a country is. Thus, from the visualization above, we can see that countries with more years of schooling tend to be less corrupted. This implies that education plays a role in positively influencing the corruption rates of a country.

How

Marks: Circles

Channels: Color hue – different colours represent different countries

Advantages

- Informative and efficient with high data-ink ratio. The scatterplot idiom clearly shows the correlation between education and corruption whilst minimizing chartjunk.
- Good interactivty.
  - Hovering over any mark in the scatterplot will display a pop up annotation with relevant information.
  - Clicking on marks in the scatterplot will highlight them and cause the unclicked circles to be displayed with unsaturated colours.
  - Countries selected in the filter on the right hand side will be shown while countries that were not selected will be hidden from the user.
- Good figure-ground. Selected marks will be displayed with saturated colours while unselected marks are displayed with unsaturated colours.

Disadvantages

- As there are a lot of different countries, the colour hue used to represent countries becomes hard to distinguish. Therefore, users are required to use the filter in order to compare the details of different countries.

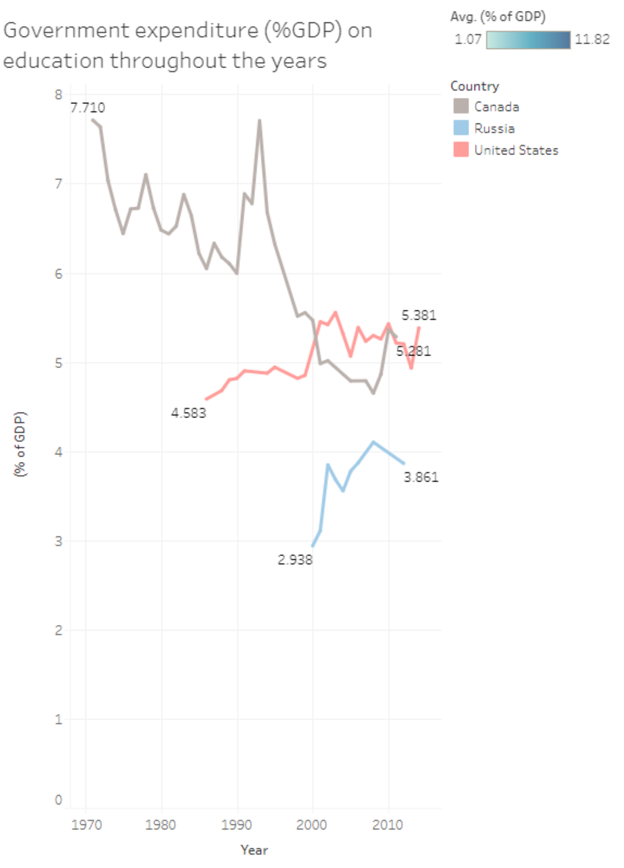
Thus, in the above section, we have discussed in details the various unqiue visualizations used in our final project. As mentioned previously, we will not review all of the visualizations as they are relatively similar in format. Below is a screenshot of all the visualizations used in the project:

Project Visualizations:

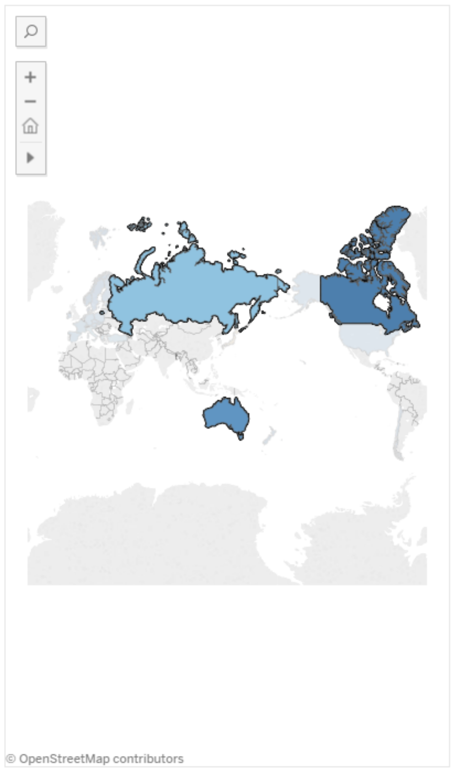
Global average government expenditure (%GDP) on education over the years

© OpenStreetMap contributors

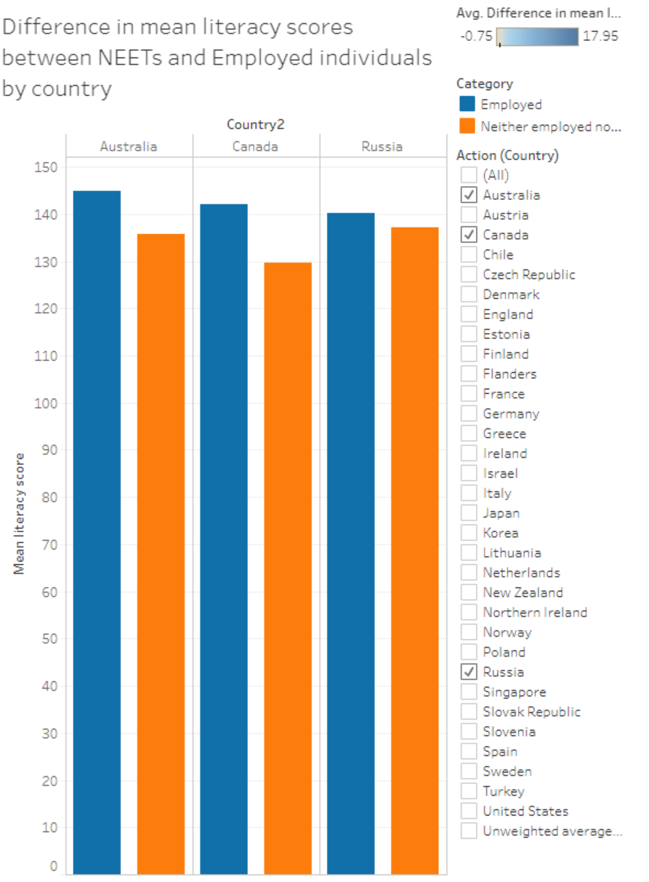
Government expenditure (%GDP) on education throughout the years



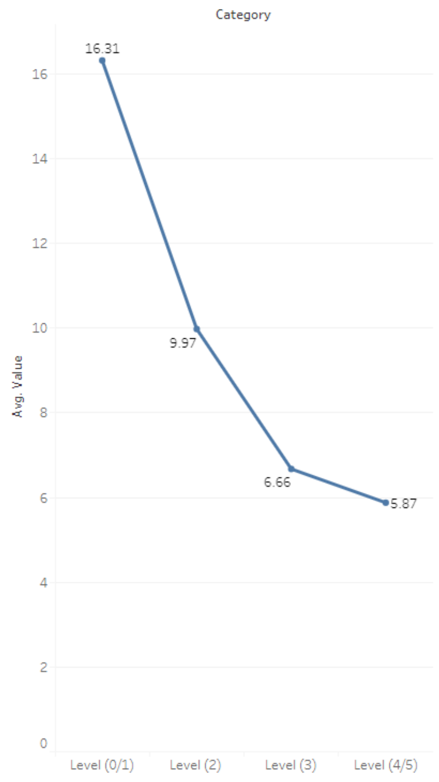
Difference in mean literacy scores between NEETs and Employed individuals by country



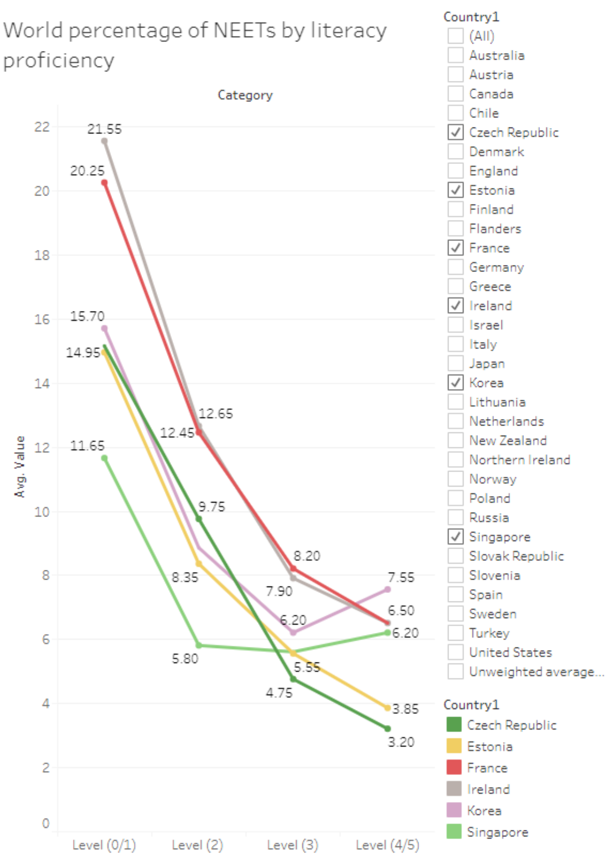
Difference in mean literacy scores between NEETs and Employed individuals by country



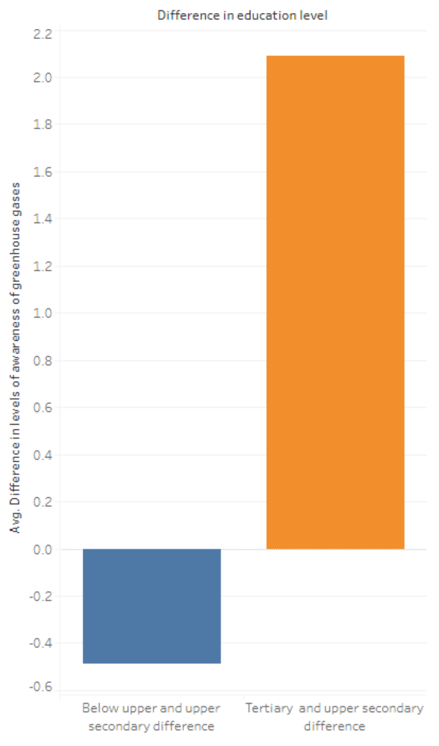
Global average percentage of NEETs by literacy proficiency level



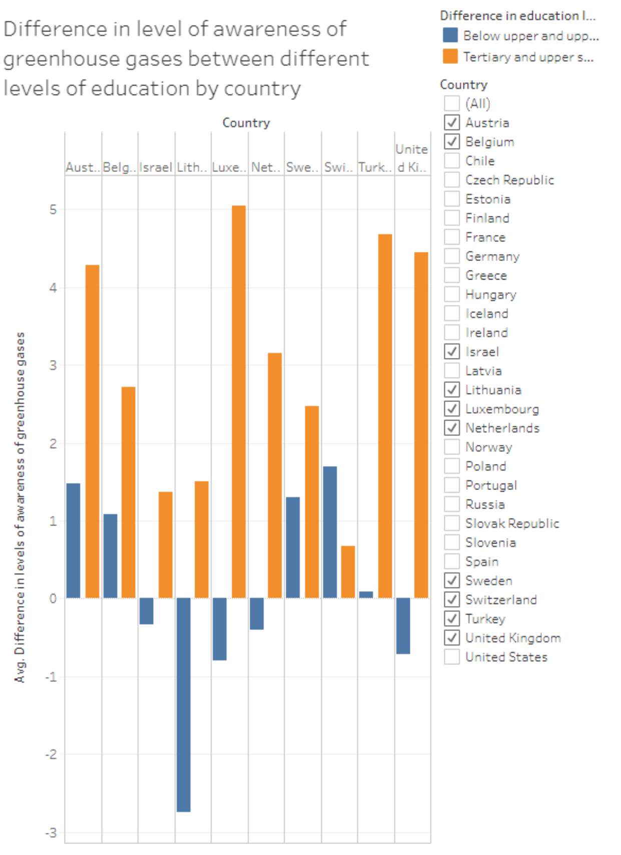
World percentage of NEETs by literacy proficiency



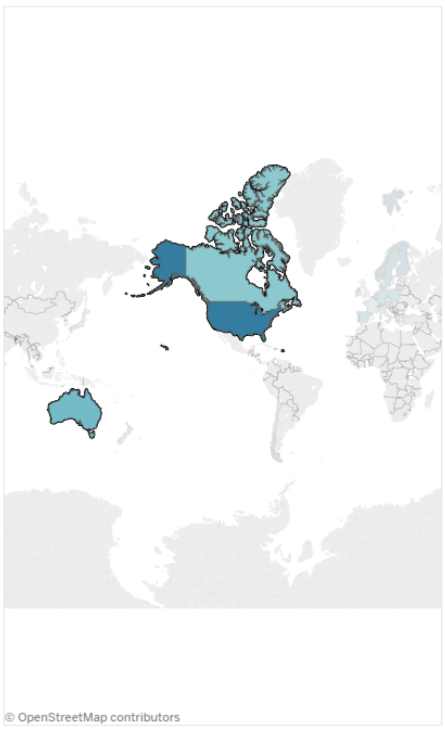
Global differences in level of awareness of greenhouses gases between different levels of education



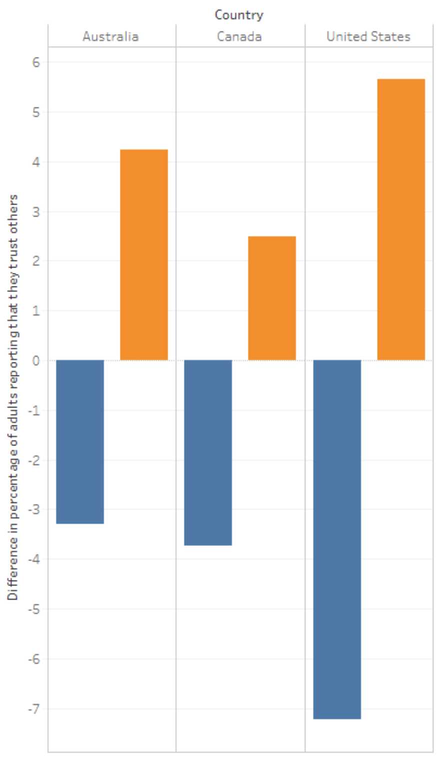
Difference in level of awareness of greenhouse gases between different levels of education by country

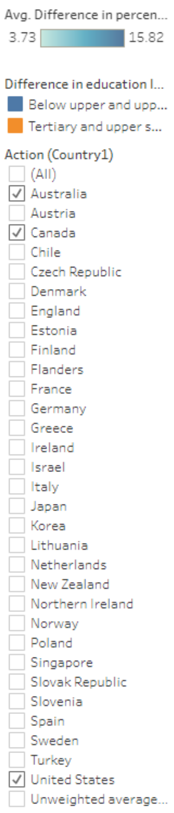


Difference in percentage of adults between tertiary and below upper secondary education reporting that they trust others

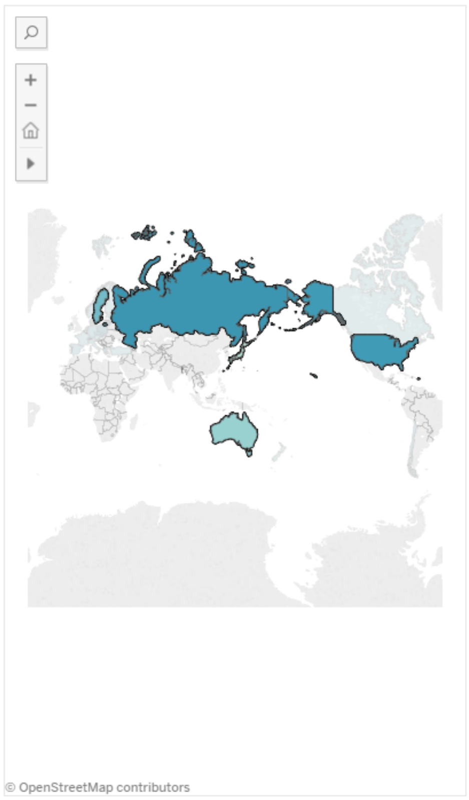


Difference in percentage of adults reporting that they trust others between different levels of education by country

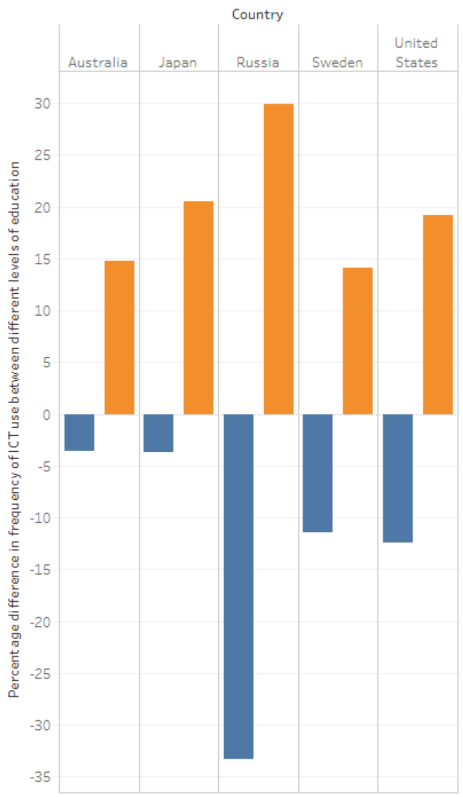


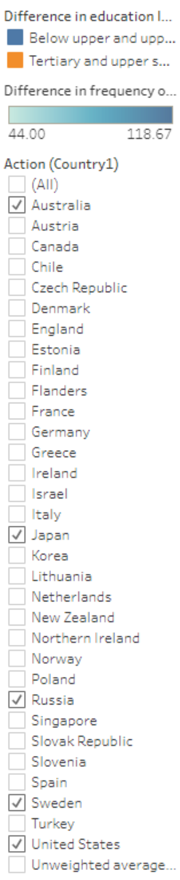


Difference of frequency of ICT use between tertiary and below upper secondary education

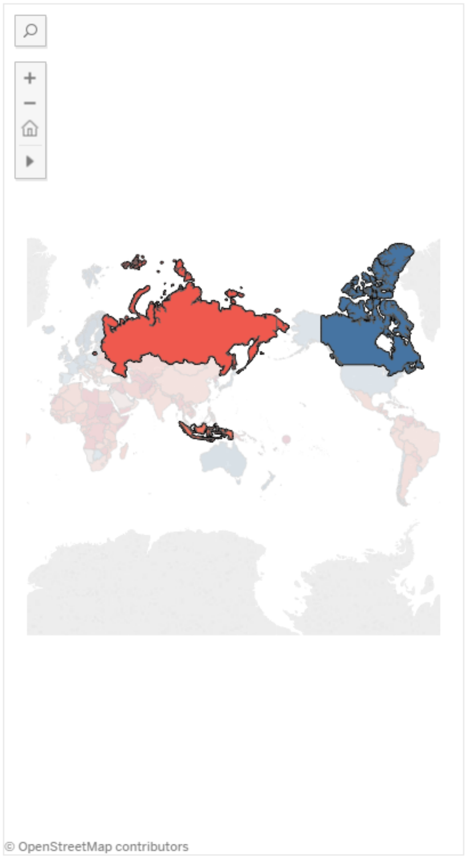


Percentage difference of frequency of ICT use between different levels of education by country

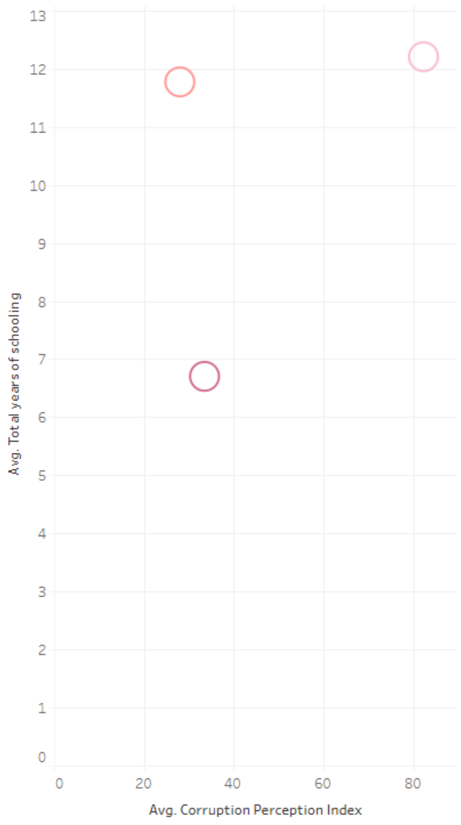




World Corruption Perception Index



Total years of schooling vs Corruption Perception Index



Avg. Corruption Percepti...

8.00 91.00

Country

🟪 Canada

🟪 Indonesia

🟥 Russia

Sources of data

It is worth mentioning that the data sets that were attained from the sources below were cleaned up and modified using self-developed scripts. Screenshots of these scripts can be seen in the Appendix.

1. Awareness of greenhouse gases by level of educational attainment.

OECD, *Education and social outcomes: Environment*, 2015, URL: https://stats.oecd.org/

2. Self-reported trust levels in others by level of educational attainment.

OECD, *Education and social outcomes: Social Connections*, 2012-2015, URL: https://stats.oecd.org/

3. Frequency of use of ICT at work by level of educational attainment.

OECD, *Educational attainment and labour market outcomes by skills: Frequency of use of ICT at work, by educational attainment*, 2012-2015, URL: https://stats.oecd.org/

4. Percentage of NEETs by literacy proficiency and mean literacy score

OECD, *Educational attainment and labour market outcomes by skillsNEETs, by literacy proficiency level and mean score*, 2012-2015, URL: https://stats.oecd.org/

5. Total years of schooling vs. Corruption Perception Index

Our World In Data, *Average years of schooling vs. Corruption Perception Index,* 2010, URL: https://ourworldindata.org/corruption#correlates-determinants-and-consequences

6. Total government expenditure on education by country

Our World In Data, *Total government expenditure on education (% GDP)*, 1970-2016, URL: https://ourworldindata.org/global-rise-of-education#cross-country-spending-patters

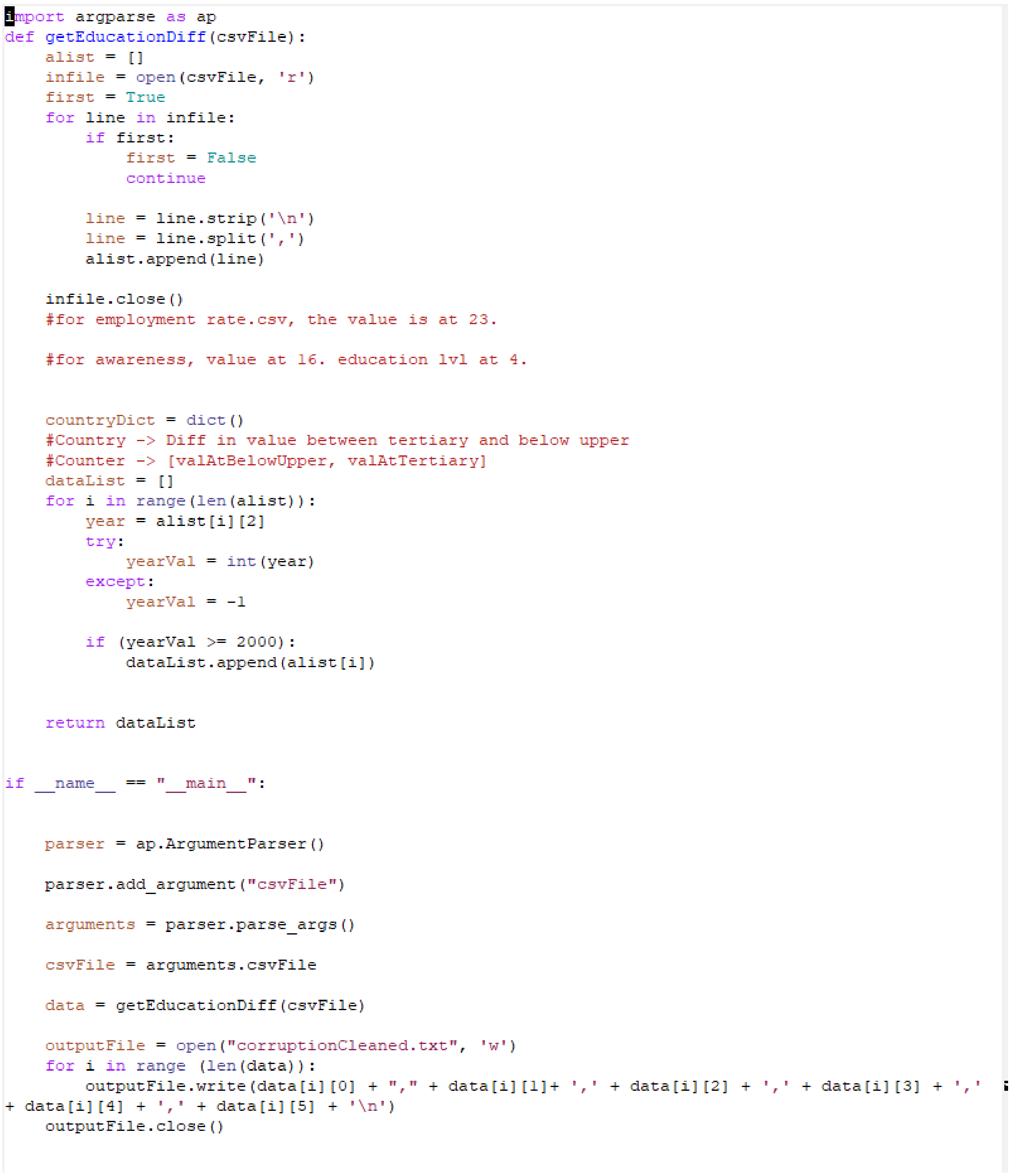
References

[1] McPhilips, D. (2018, Jan 23). *Methodology: How the 2018 Best Countries Were Ranked*. Retrieved from https://www.usnews.com/news/best-countries/articles/methodology

Appendix

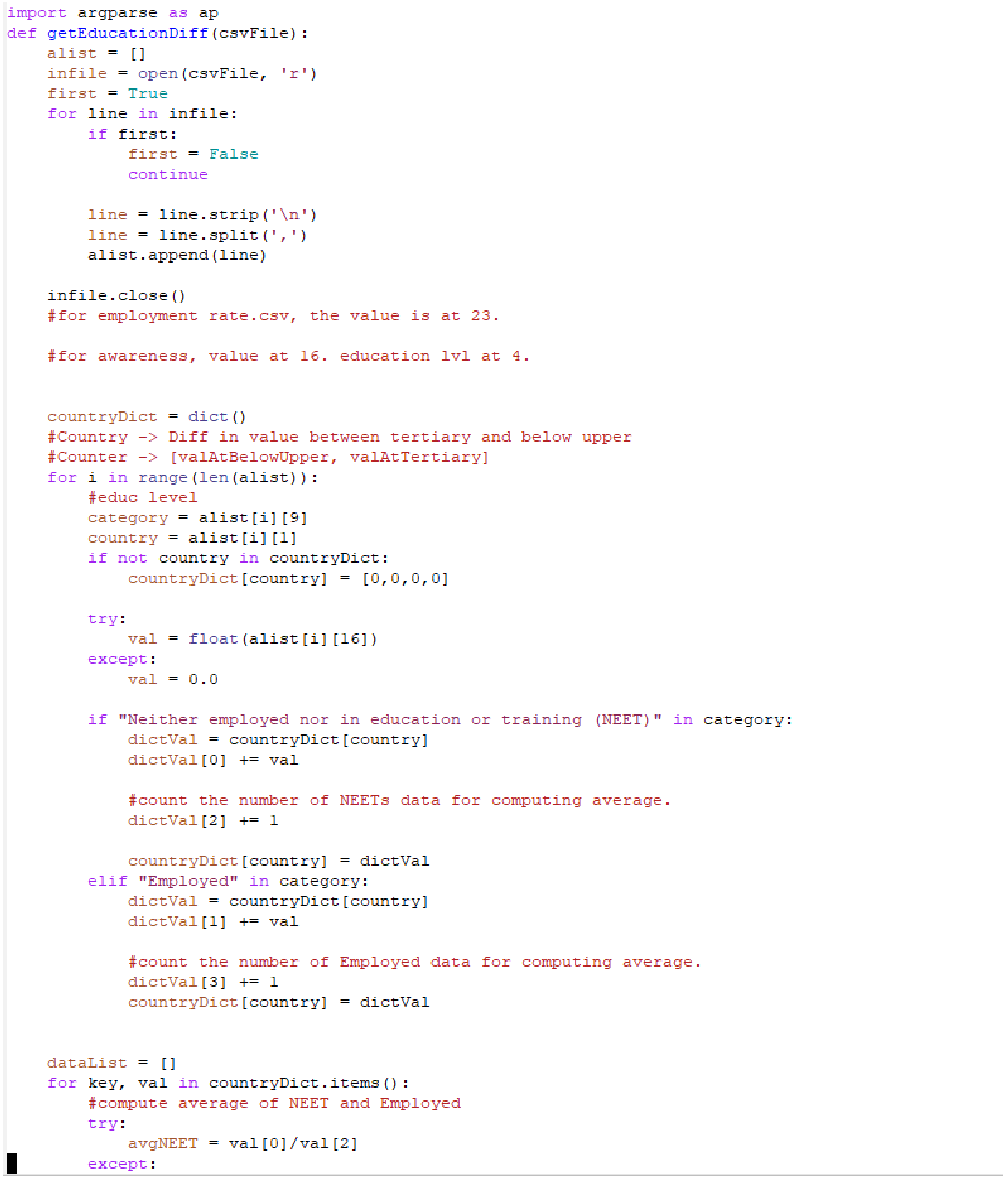
1. Source codes

1.1 Corruption Perception Index vs Total years of schooling



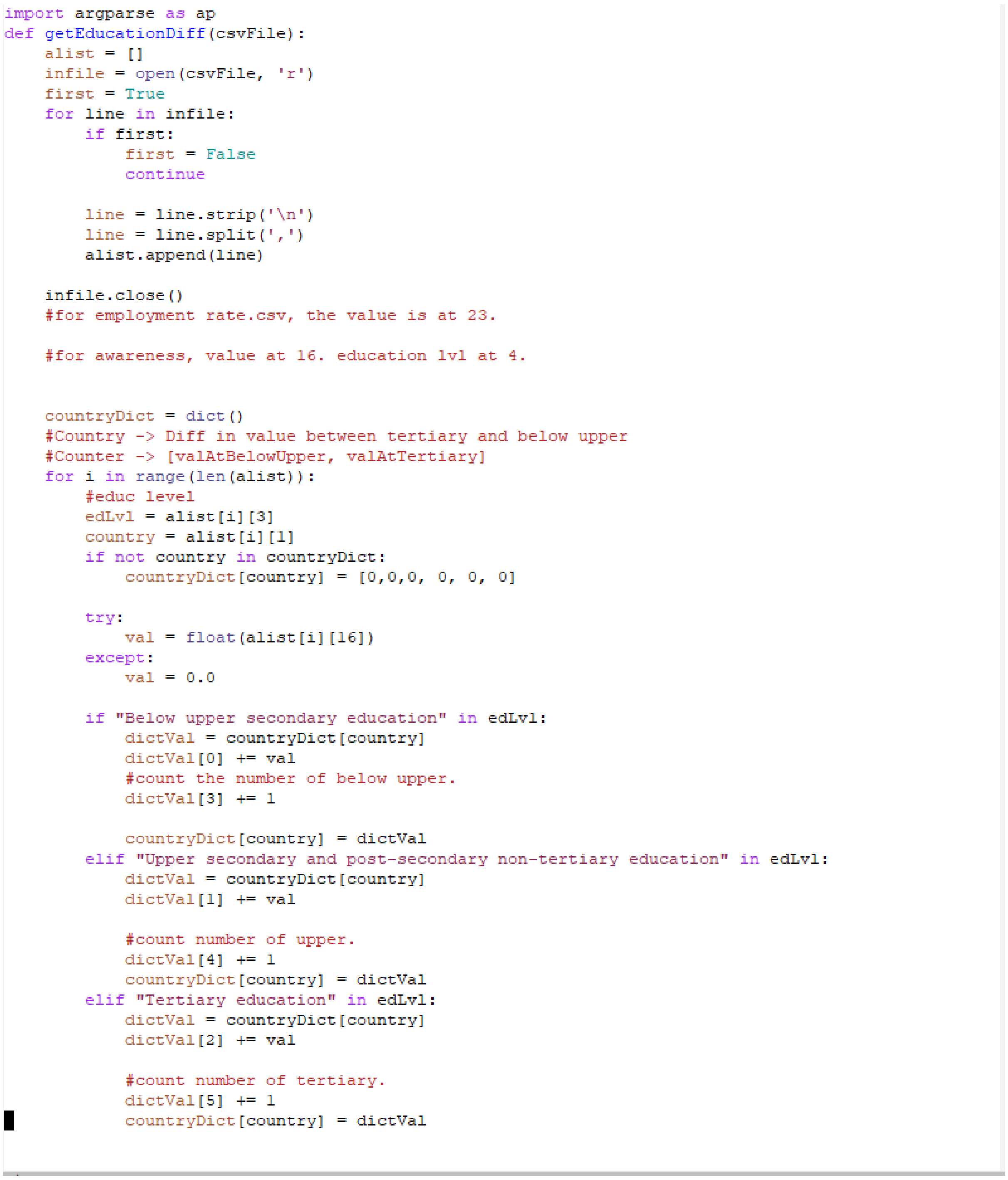
```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    dataList = []
    for i in range(len(alist)):
        year = alist[i][2]
        try:
            yearVal = int(year)
        except:
            yearVal = -1

        if (yearVal >= 2000):
            dataList.append(alist[i])


    return dataList


if __name__ == "__main__":


    parser = ap.ArgumentParser()

    parser.add_argument("csvFile")

    arguments = parser.parse_args()

    csvFile = arguments.csvFile

    data = getEducationDiff(csvFile)

    outputFile = open("corruptionCleaned.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + data[i][1]+ ',' + data[i][2] + ',' + data[i][3] + ','
+ data[i][4] + ',' + data[i][5] + '\n')
    outputFile.close()
```

1.2 Average NEETs percentage



```python
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        category = alist[i][9]
        country = alist[i][1]
        if not country in countryDict:
            countryDict[country] = [0,0,0,0]

        try:
            val = float(alist[i][16])
        except:
            val = 0.0

        if "Neither employed nor in education or training (NEET)" in category:
            dictVal = countryDict[country]
            dictVal[0] += val

            #count the number of NEETs data for computing average.
            dictVal[2] += 1

            countryDict[country] = dictVal
        elif "Employed" in category:
            dictVal = countryDict[country]
            dictVal[1] += val

            #count the number of Employed data for computing average.
            dictVal[3] += 1
            countryDict[country] = dictVal


    dataList = []
    for key, val in countryDict.items():
        #compute average of NEET and Employed
        try:
            avgNEET = val[0]/val[2]
        except:
            avgNEET = 0.0
        try:
            avgEmployed = val[1]/val[3]
        except:
            avgEmployed = 0.0

        diffAvg = avgEmployed - avgNEET
        if (diffAvg != 0.0) :
            dataList.append([key, diffAvg])


    return dataList


if __name__ == "__main__":


    parser = ap.ArgumentParser()

    parser.add_argument("csvFile")

    arguments = parser.parse_args()

    csvFile = arguments.csvFile

    data = getEducationDiff(csvFile)

    outputFile = open("NEETDiffVal.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + str(data[i][1])+'\n')
    outputFile.close()
```

1.3 Awareness of greenhouse gases - script 1



```python
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        if not country in countryDict:
            countryDict[country] = [0,0,0, 0, 0, 0]

        try:
            val = float(alist[i][16])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            #count the number of below upper.
            dictVal[3] += 1

            countryDict[country] = dictVal
        elif "Upper secondary and post-secondary non-tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val

            #count number of upper.
            dictVal[4] += 1
            countryDict[country] = dictVal
        elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val

            #count number of tertiary.
            dictVal[5] += 1
            countryDict[country] = dictVal
```

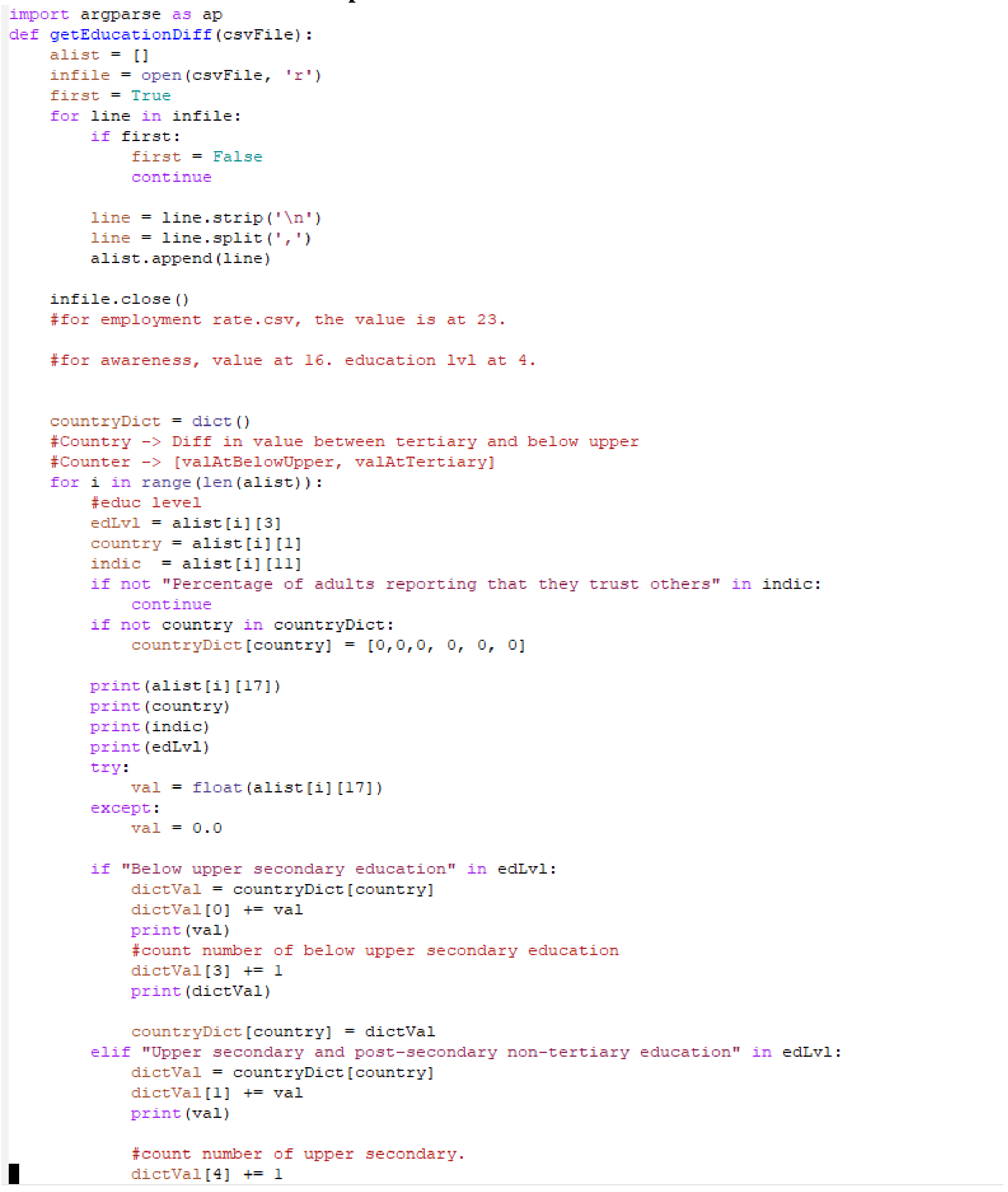
-

```
    dataList = []
    for key, val in countryDict.items():
        try:
            avgBelow = val[0]/val[3]
        except:
            avgBelow = 0.0
        try:
            avgUpper = val[1]/val[4]
        except:
            avgUpper = 0.0
        try:
            avgTer = val[2]/val[5]
        except:
            avgTer = 0.0
        belowUpperDiff = avgBelow - avgUpper
        terSecDiff = avgTer - avgUpper

        if (belowUpperDiff != 0.0 or terSecDiff != 0.0) :
            dataList.append([key, "Below upper and upper secondary difference", belowUpperDiff])
            dataList.append([key, "Tertiary  and upper secondary difference", terSecDiff])


    return dataList


if __name__ == "__main__":


    parser = ap.ArgumentParser()

    parser.add_argument("csvFile")

    arguments = parser.parse_args()

    csvFile = arguments.csvFile

    data = getEducationDiff(csvFile)

    outputFile = open("diffValue.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + str(data[i][1])+ ',' + str(data[i][2]) +'\n')
    outputFile.close()
```

1.4 Awareness of greenhouse gases – script 2

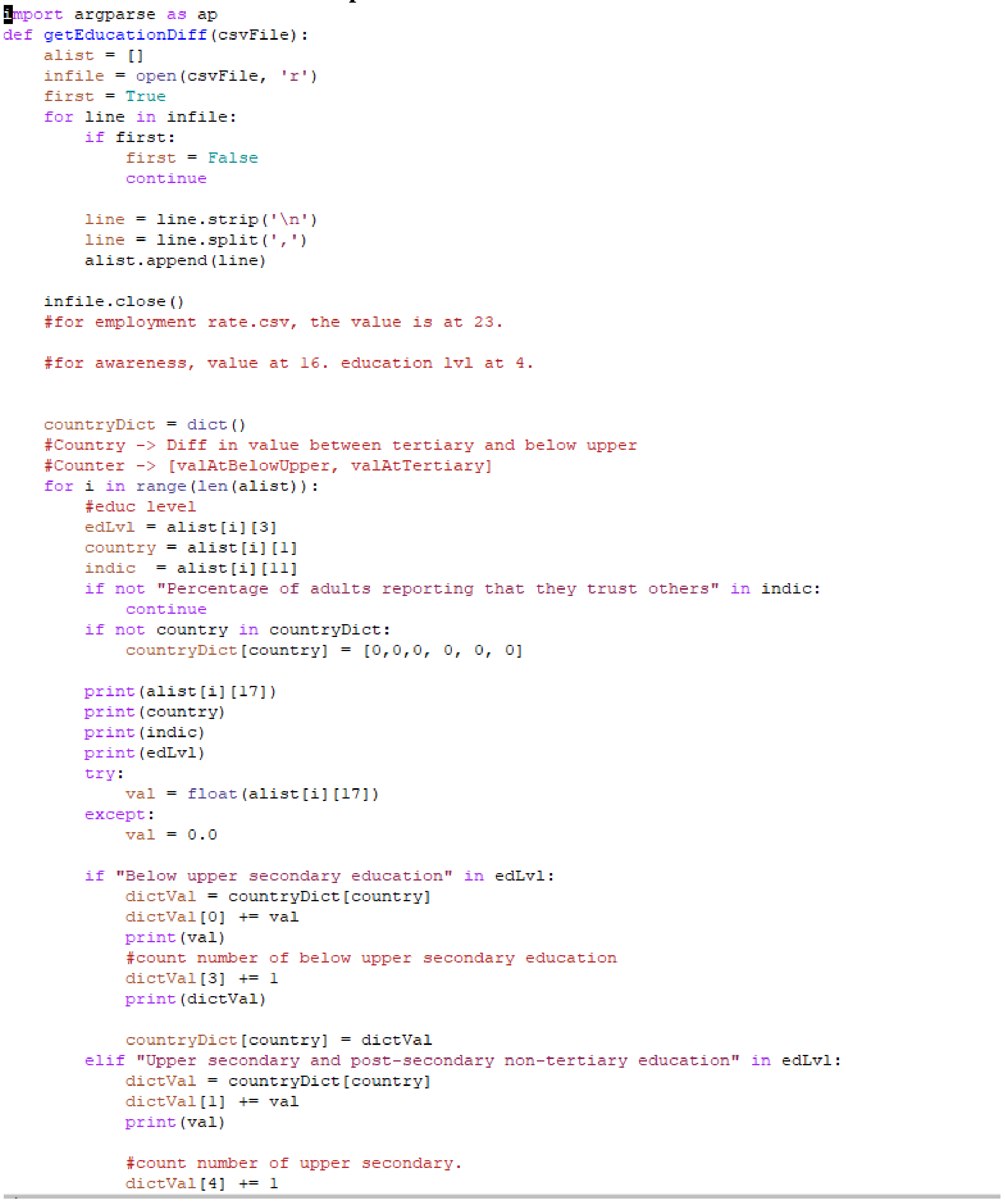
```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        if not country in countryDict:
            countryDict[country] = [0,0,0]

        try:
            val = float(alist[i][16])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            countryDict[country] = dictVal
        elif "Upper secondary and post-secondary non-tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val
            countryDict[country] = dictVal
        elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val
            countryDict[country] = dictVal


    dataList = []
    for key, val in countryDict.items():
        belowUpperDiff = val[0] - val[1]
        #get percentage difference.
        try:
            percentBU = (belowUpperDiff / val[1]) * 100
        except:
            percentBU = 0.0
        terSecDiff = val[2]-val[1]
        try:
            percentTS = (terSecDiff / val[1]) * 100
        except:
            percentTS = 0.0

        if (belowUpperDiff != 0.0 or terSecDiff != 0.0) :
            dataList.append([key, "Below upper and upper secondary difference", percentBU])
            dataList.append([key, "Tertiary  and upper secondary difference", percentTS])


    return dataList


if __name__ == "__main__":


    parser = ap.ArgumentParser()

    parser.add_argument("csvFile")

    arguments = parser.parse_args()

    csvFile = arguments.csvFile

    data = getEducationDiff(csvFile)

    outputFile = open("diffValue2.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + str(data[i][1])+ ',' + str(data[i][2]) +'\n')
    outputFile.close()
```

1.5 Trust levels in others – script 1



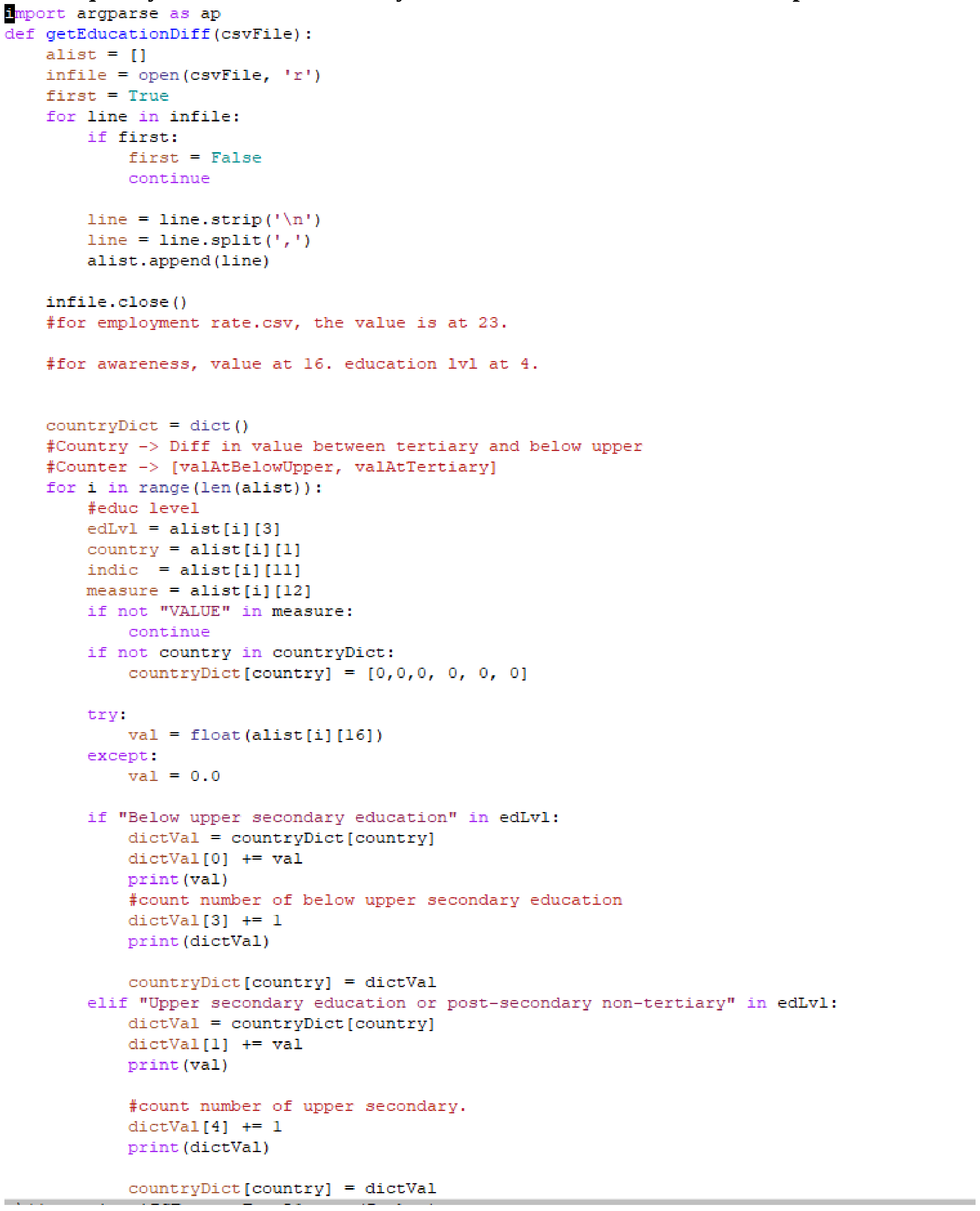
```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        indic  = alist[i][11]
        if not "Percentage of adults reporting that they trust others" in indic:
            continue
        if not country in countryDict:
            countryDict[country] = [0,0,0, 0, 0, 0]

        print(alist[i][17])
        print(country)
        print(indic)
        print(edLvl)
        try:
            val = float(alist[i][17])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            print(val)
            #count number of below upper secondary education
            dictVal[3] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Upper secondary and post-secondary non-tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val
            print(val)

            #count number of upper secondary.
            dictVal[4] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val
            print(val)
            #count number of tertiary.
            dictVal[5] += 1
            print(dictVal)
            countryDict[country] = dictVal
    print("---- ")

    dataList = []
    for key, val in countryDict.items():
        #compute averages.

        avgBelow = val[0]/val[3]
        avgUpper = val[1]/val[4]
        avgTer = val[2]/val[5]
        """
        print(str(val[0]) + "," + str(val[3]))
        print(str(val[1]) + "," + str(val[4]))
        print(str(val[2]) + "," + str(val[5]))
        print(" " )
        """
        belowUpperDiff = avgBelow - avgUpper

        #get percentage difference. -- incase we wanna see diff percetange.
        try:
            percentBU = (belowUpperDiff / val[1]) * 100
        except:
            percentBU = 0.0
        terSecDiff = avgTer - avgUpper
        try:
            percentTS = (terSecDiff / val[1]) * 100
        except:
            percentTS = 0.0

        if (belowUpperDiff != 0.0 or terSecDiff != 0.0) :
            dataList.append([key, "Below upper and upper secondary difference",belowUpperDiff])
            dataList.append([key, "Tertiary  and upper secondary difference", terSecDiff])
    return dataList


if __name__ == "__main__":
    parser = ap.ArgumentParser()
    parser.add_argument("csvFile")
    arguments = parser.parse_args()
    csvFile = arguments.csvFile
    data = getEducationDiff(csvFile)
    outputFile = open("diffTrust.txt", 'w')
    for i in range (len(data)):
        print(data[i][0] +"," + str(data[i][1]))
        outputFile.write(data[i][0] + "," + str(data[i][1])+ ',' + str(data[i][2]) +'\n')
    outputFile.close()
```

1.6 Trust levels in others – script 2



```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        indic  = alist[i][11]
        if not "Percentage of adults reporting that they trust others" in indic:
            continue
        if not country in countryDict:
            countryDict[country] = [0,0,0, 0, 0, 0]

        print(alist[i][17])
        print(country)
        print(indic)
        print(edLvl)
        try:
            val = float(alist[i][17])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            print(val)
            #count number of below upper secondary education
            dictVal[3] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Upper secondary and post-secondary non-tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val
            print(val)

            #count number of upper secondary.
            dictVal[4] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val
            print(val)
            #count number of tertiary.
            dictVal[5] += 1
            print(dictVal)
            countryDict[country] = dictVal
    print("---- ")

    dataList = []
    for key, val in countryDict.items():
        #compute averages.

        avgBelow = val[0]/val[3]
        avgUpper = val[1]/val[4]
        avgTer = val[2]/val[5]
        belowUpperDiff = avgBelow - avgUpper
        #get percentage difference. -- incase we wanna see diff percetange.
        try:
            percentBU = (belowUpperDiff / val[1]) * 100
        except:
            percentBU = 0.0
        terSecDiff = avgTer - avgUpper
        try:
            percentTS = (terSecDiff / val[1]) * 100
        except:
            percentTS = 0.0
        diffVal = avgTer - avgBelow
        if (diffVal != 0.0):
            dataList.append([key, diffVal])
    return dataList


if __name__ == "__main__":
    parser = ap.ArgumentParser()
    parser.add_argument("csvFile")
    arguments = parser.parse_args()
    csvFile = arguments.csvFile
    data = getEducationDiff(csvFile)
    outputFile = open("diffTrust2.txt", 'w')
    for i in range (len(data)):
        print(data[i][0] +"," + str(data[i][1]))
        outputFile.write(data[i][0] + "," + str(data[i][1])+'\n')
    outputFile.close()
```

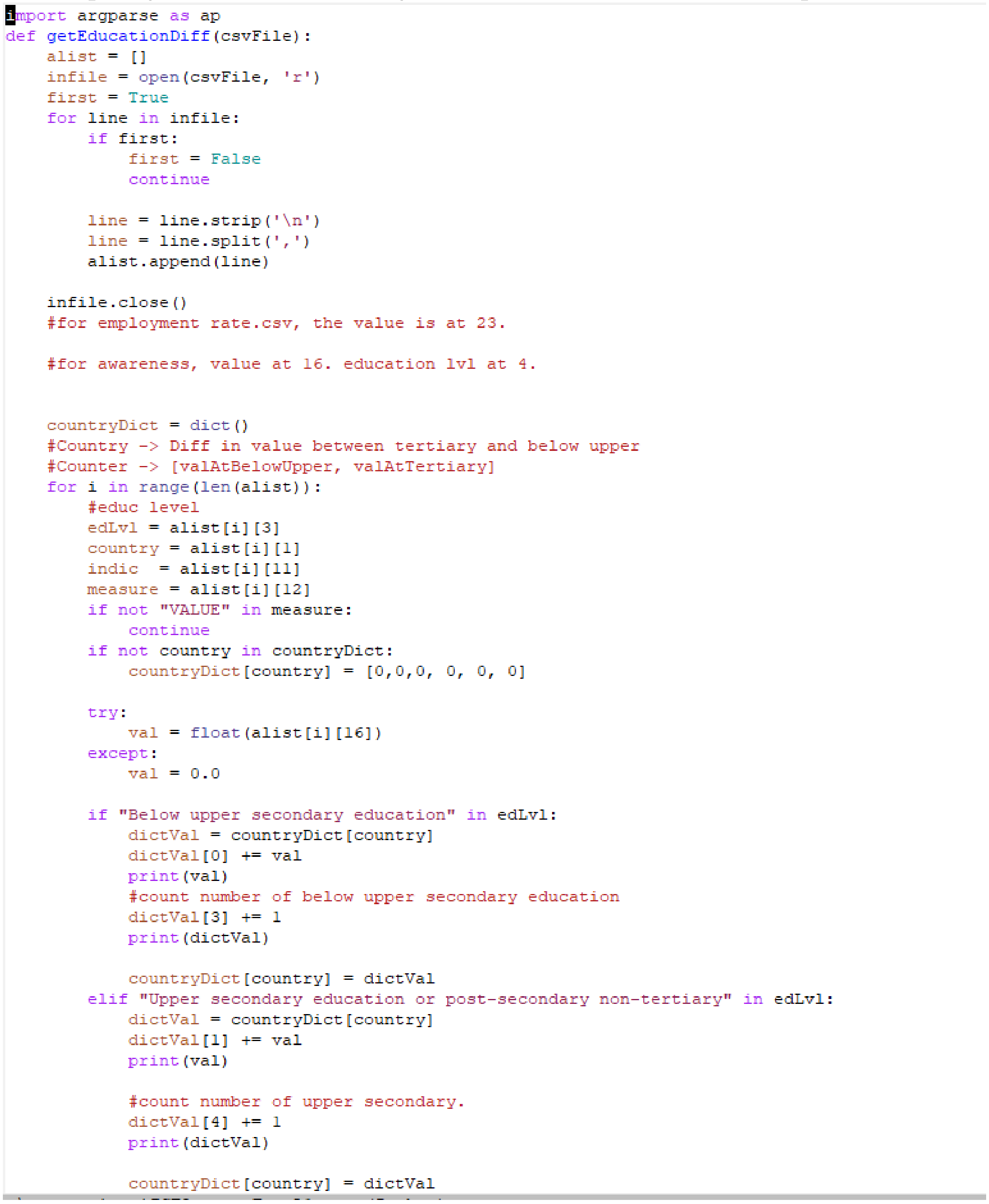
1.7 Frequency of ICT use at work by educational attainment level – script 1



```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        indic  = alist[i][11]
        measure = alist[i][12]
        if not "VALUE" in measure:
            continue
        if not country in countryDict:
            countryDict[country] = [0,0,0, 0, 0, 0]

        try:
            val = float(alist[i][16])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            print(val)
            #count number of below upper secondary education
            dictVal[3] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Upper secondary education or post-secondary non-tertiary" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val
            print(val)

            #count number of upper secondary.
            dictVal[4] += 1
            print(dictVal)

            countryDict[country] = dictVal
```

```
-       elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val
            print(val)
            #count number of tertiary.
            dictVal[5] += 1
            print(dictVal)
            countryDict[country] = dictVal
    print("---- ")

    dataList = []
    for key, val in countryDict.items():
        #compute averages.

        avgBelow = val[0]/val[3]
        avgUpper = val[1]/val[4]
        avgTer = val[2]/val[5]
        belowUpperDiff = avgBelow - avgUpper

        #get percentage difference. -- incase we wanna see diff percetange.
        try:
            percentBU = (belowUpperDiff / val[1]) * 100
        except:
            percentBU = 0.0
        terSecDiff = avgTer - avgUpper
        try:
            percentTS = (terSecDiff / val[1]) * 100
        except:
            percentTS = 0.0

        if (belowUpperDiff != 0.0 or terSecDiff != 0.0) :
            dataList.append([key, "Below upper and upper secondary difference",percentBU])
            dataList.append([key, "Tertiary  and upper secondary difference", percentTS])

    return dataList


if __name__ == "__main__":
    parser = ap.ArgumentParser()
    parser.add_argument("csvFile")
    arguments = parser.parse_args()
    csvFile = arguments.csvFile
    data = getEducationDiff(csvFile)
    outputFile = open("diffTech.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + str(data[i][1])+ ',' + str(data[i][2]) +'\n')
    outputFile.close()
```

1.8 Frequency of ICT use at work by educational attainment level – script 2



```
import argparse as ap
def getEducationDiff(csvFile):
    alist = []
    infile = open(csvFile, 'r')
    first = True
    for line in infile:
        if first:
            first = False
            continue

        line = line.strip('\n')
        line = line.split(',')
        alist.append(line)

    infile.close()
    #for employment rate.csv, the value is at 23.

    #for awareness, value at 16. education lvl at 4.


    countryDict = dict()
    #Country -> Diff in value between tertiary and below upper
    #Counter -> [valAtBelowUpper, valAtTertiary]
    for i in range(len(alist)):
        #educ level
        edLvl = alist[i][3]
        country = alist[i][1]
        indic  = alist[i][11]
        measure = alist[i][12]
        if not "VALUE" in measure:
            continue
        if not country in countryDict:
            countryDict[country] = [0,0,0, 0, 0, 0]

        try:
            val = float(alist[i][16])
        except:
            val = 0.0

        if "Below upper secondary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[0] += val
            print(val)
            #count number of below upper secondary education
            dictVal[3] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Upper secondary education or post-secondary non-tertiary" in edLvl:
            dictVal = countryDict[country]
            dictVal[1] += val
            print(val)

            #count number of upper secondary.
            dictVal[4] += 1
            print(dictVal)

            countryDict[country] = dictVal
        elif "Tertiary education" in edLvl:
            dictVal = countryDict[country]
            dictVal[2] += val
            print(val)
            #count number of tertiary.
            dictVal[5] += 1
            print(dictVal)
            countryDict[country] = dictVal
    print("---- ")

    dataList = []
    for key, val in countryDict.items():
        #compute averages.

        avgBelow = val[0]/val[3]
        avgUpper = val[1]/val[4]
        avgTer = val[2]/val[5]
        belowUpperDiff = avgBelow - avgUpper

        #get percentage difference. -- incase we wanna see diff percetange.
        try:
            percentBU = (belowUpperDiff / val[1]) * 100
        except:
            percentBU = 0.0
        terSecDiff = avgTer - avgUpper
        try:
            percentTS = (terSecDiff / val[1]) * 100
        except:
            percentTS = 0.0
        '''
        if (belowUpperDiff != 0.0 or terSecDiff != 0.0) :
            dataList.append([key, "Below upper and upper secondary difference",percentBU])
            dataList.append([key, "Tertiary  and upper secondary difference", percentTS])
        '''

        diffVal = avgTer - avgBelow
        if (diffVal != 0.0):
            dataList.append([key, diffVal])

    return dataList


if __name__ == "__main__":
    parser = ap.ArgumentParser()
    parser.add_argument("csvFile")
    arguments = parser.parse_args()
    csvFile = arguments.csvFile
    data = getEducationDiff(csvFile)
    outputFile = open("diffTech2.txt", 'w')
    for i in range (len(data)):
        outputFile.write(data[i][0] + "," + str(data[i][1])+'\n')
    outputFile.close()
```