

Hova tovább?

A Quality of Life mutatószám értékelése a Föld körül

1. Bevezetés

Kezdeti probléma az életminőséggel

Az életminőség visszatérő kérdése akár a hétköznapiakban is egy egyszerű családi, esetleg munkatársi beszélgetés során. Alapvetően mindig az lesz a visszatérő motívum ezekből a beszélgetésekből, hogy mitől jobb egyik ország a másiknál? Mitől annyira jobb az élet nyugaton, mint nálunk keleten, esetleg mi befolyásolhatja és miért is annyival előnyösebb szerencsét próbálni nyugaton?

Az életminőség felmérése egy komplex, gazdasági, de akár nem gazdasági szempontból is relatíve sok adatot igénylő folyamat, amelynek eredménye általában valamilyen mérőszám. A közgazdaságban a közgazdász hallgatók számára leggyakrabban a $GDP/fő$ -t nevezik meg, mint egy lehetséges mutatószáma az életminőségnek, azonban minden esetben felhívják az oktatók a hallgatók figyelmét arra, hogy közel sem pontos, vagy éppen tökéletesen. A legjobb példa, amit erre valaha hallottam: lehet a GDP-t növelni kószobrokkal vagy kórházakkal, mindenki kitalálhatja melyik növeli valójában az életminőséget, természetesen szabadon idézve.

Azonban vannak olyan szervezetek, illetve vállalatok, akik egy következő szintre emelték az életminőség felmérését, ilyen példaképpen a *Numbeo* is. A *Numbeo* valójában egy adatbázisként nevezi meg magát, különböző területekről, mint példaképpen az életminőség. Megragadott a *Numbeo* megközelítése a problémára, hiszen ahelyett, hogy ismételnék a gazdasági mutatók alapján kialakított következtetéseket, új megközelítést alkotnak. Nem-gazdasági szintre emelik azt, legnagyobb részt, azonban fel-fel bukkannak gazdasági elemek, ezek mind megtekinthetőek és értelmezhetőek az összeállított vizualizációban, ahol egyesekre részletesen kitérek: szemléltetem a mutatószám alakulását országokra lebontva évenként, majd a komponenseit is. Végül kitérek egy érdekes összefüggésre arra, hogy a közgazdászoknak évtizedek óta igazuk van-e, amikor az életminőséget a GNI/fő alapján akarják értékelni? Röviden a válasz, hogy részben képes értékelni, de sosem lesz elég önmagában.

Adatok forrása

A *Numbeo* az életminőségre vonatkozó adatok a weboldalán közli, táblázatos formában¹. Mindig fontos észben tartani azt, hogy valójában városokra lebontva gyűjt adatokat, majd ezek súlyozott átlagával országokra lebontott statisztikákat is publikál.

Az első gond már ott jelentkezett, hogy kiderült, hogy a *Numbeo* által szolgáltatott REST API, ami JSON formátumban szolgáltatná az adatokat, valójában

¹Quality of Life Index by Country 2021 Mid-Year (numbeo.com)

fizető. Kiterjesztettsége miatt igen drága, azonban hasznos. Ezért saját magam szerettem volna összeállítani egy adatszerzési scriptet Python-ban²:

```
def scrapeData(year):
    URL = f'https://www.numbeo.com/quality-of-life/\
rankings_by_country.jsp?title={year}'
    page = requests.get(URL)
    soup = BeautifulSoup(page.content, 'html.parser')

    table = soup.find('table', id='t2')
    tableContent = table.find('tbody')
    tableRows = tableContent.find_all("tr")

    data = []
    for tableRow in tableRows:
        data.append(list(filter(lambda row : (row != ''), \
            tableRow.text.split('\n'))))

    return data
```

A fenti script annyit tesz, hogy lekéri HTML dokumentum formájában a *Numbeo* adott évi összeállítását, majd a **BeautifulSoup** Python könyvtár értelmezi ezt a dokumentumot és megkeresi azokat a DOM elemeket, amelyek engem érdekelnek, ebben az esetben az adott táblázat sorait. A táblázat fejléce nincs ebbe befoglalva, hiszen nem akartam, hogy minden évben ismétlődjön, egyszer, külön kértem azt le.

Ezt követően egy újabb függvényt írtam, amely CSV formátumban exportálta évekre lebontva az adatokat:

```
counterYear = 2012
endYear = 2020

while counterYear <= endYear:
    with open(f'{counterYear}.csv', 'w', newline='') as outputFile:
        writer = csv.writer(outputFile)

        writer.writerow(headings)

        for row in data[counterYear]:
            writer.writerow(row)

    counterYear += 1
```

Ez által lényegében már rendelkezésemre álltak az adatok, amikre a *Numbeo* weboldaláról szükségem volt, egy adathalmaz hiányzott még ezen kívül, a GNI/fő, amit nagyon egyszerűen CSV formátumban ellehetett érni.

²A teljes IPYNB megtalálható a GitHubon: github.com/wheresouldigoviz

Így két lépésből megvolt minden adat, ami a vizualizáció elkészítéséhez szükséges:

- rendelkezésemre állnak a *Numbeo* évre lebontott adatai;
- rendelkezésemre állnak a GNI/fő adatok ugyancsak évre lebontva.

Illetve, zárójelesen ország ISO kódokat is összegyűjtöttem, erre azért volt szükség, mert az első ábrám, ami egy térképvizualizáció az egyes országokat koordináták hiányában ISO nevek alapján azonosítja.

2. Adatfelfedezés és megismerés

Miután rendelkezésemre álltak az adatok, a következő lépésem az volt, hogy olyan formában hozzam azt, ami számomra és az összeállítandó vizualizáció számára könnyen értelmezhető és feldolgozható.

A cél alapvetően az volt, hogy egy CSV állományt hozzak létre, amiben minden adat rendelkezésemre áll egyetlen helyen normalizált formában:

Év	Országnev	Quality of Life komponensek	...	GNI/fő
----	-----------	-----------------------------	-----	--------

Természetesen, mivel az adatokat én gyűjtöttem össze és táblázatos formában láttam, hogy mit fogok gyűjteni, viszonylag könnyű volt átlátni azt és a normalizálási folyamatot is ez nagyban megkönnyítette.

A Pandas Python könyvtárt használva az első lépés a korábban létrehozott CSV állományaim betöltése volt, majd ezeket kellett összesíteni, egyetlen DataFrame-be vonva:

```
import pandas as pd

qlf = {}

counterYear = 2012
endYear = 2020

while counterYear <= endYear:
    qlf[counterYear] = pd.read_csv(f'raw/{counterYear}.csv')
    counterYear += 1

counterYear = 2012
endYear = 2020

while counterYear <= endYear:
    # ezzel beszúrunk egy új Year-nek nevezett oszlopot a
    # DataFrame-be, amelynek értéke minden sorban
    # az adott évvel lesz megegyező
```

```

qlf[counterYear]['Year'] = counterYear

counterYear += 1

frames = []

for key in qlf.keys():
    frames.append(qlf[key])

# az összes tömbben található DataFrame-t egyetlenbe hozza össze,
# mindenik
# DF-nek megegyező oszlopszáma és nevei vannak
mergedData = pd.concat(frames)

```

Ezt követően megtekintve az adathalmazom rövid összefoglalóját a `mergedData.info()` metódussal az alábbi következtetésekre jutottam:

- 606 sorból álló adathalmazom van, hozzávetőlegesen ~79 országot lefedve;
- 606 sor mindenikének esetében rendelkezésemre áll 9, 2015 után már 10 dimenzió, a különböző életminőség komponensek ábrázolásra;
- azonban vannak üres helyek, hiányzó értékek, példaképpen 2015-ig a Klímaindex nem létezett a komponensek között, illetve emiatt a 'Climate Index' oszlopom adattípusa sem megfelelő, szám helyett objektum típust értelmezett neki a feldolgozó;
- azonban nincs null értékem, tehát csak a hiányzó klímaindex-el kell foglalkozni, legalábbis a *Numbeo* adatai esetében;

Következő lépés a hiányzó Klímaindex értékek átalakítása volt, egy olyan értékre, ami lehetővé teszi, hogy a továbbiakban számként tudjam azt használni. Tehát, fogtam az összes olyan értéket a Klímaindex oszlopból, ami '-'al egyezett meg, és lecseréltem -1-re, hiszen egyetlen mutatószám komponens sem lehet negatív. Egy érvényes megoldás lehetett volna a NaN-ra állítás is, azonban számomra könnyebben értelmezhető volt ez.

```

mergedData.loc[mergedData['Climate Index'] == '-', 'Climate Index'] = -1
mergedData['Climate Index'] = mergedData['Climate Index'].astype('float64')
# lecseréli a Climate Index neű oszlopot, ami igazából Series
# adattípusú ugyanazzal, azonban a típusát átváltva lebegőpontos számra

```

A már kész *Numbeo* adatok mellett, ahogy korábban is említettem még szükségem volt két adatsorra: GNI és ország ISO kódok és nevek.

Az ország ISO kódok valójában már korábbi projektjeimből hátra maradtak JSON formátumban, azonban röviden a Wikipédiáról kerültek begyűjtésre ugyancsak egy megfelelő BeautifulSoup script segítségével, ami valahogy így nézhetett ki:

```

import csv
from bs4 import BeautifulSoup
import requests

```

```

import json

url = 'https://en.wikipedia.org/wiki/ISO_3166-1_alpha-3'
headers = {'User-Agent': 'Mozilla/5.0'}

response = requests.get(url, headers=headers)
soup = BeautifulSoup(response.content, 'lxml')
relevant = \
    soup.select('#mw-content-text > div > div.div-col.columns.column-width')
list_elements = relevant[0].find_all('li')

codes = {}

for l in list_elements:
    code = l.find('span')
    country = l.find('a')

    codes[country.text] = code.text
    codes[code.text] = country.text

with open('codes.json', 'w') as codes_file:
    json.dump(codes, codes_file)

```

Beolvasás után, már csak megfelelő formátumba kellett hozni az összevonáshoz a fenti JSON-t.

```

codes = pd.read_json('raw/codes.json', orient='index')
codes = codes.reset_index()
codes = codes.rename(columns={'index': 'Code', 0: 'Country'})
# átnevezi az oszlopait

```

Azonban összevonáskor derült ki az, hogy bizonyos országok eredeti nevei, amit a *Numbeo* adott meg nem egyeznek meg az ISO nevükkel, ezek mindegyikét manuálisan át kellett neveznem.

```

mergedData.loc[mergedData['Country'] == 'Russia', 'Country']\
    = 'Russian Federation'
mergedData.loc[mergedData['Country'] == 'United States', 'Country']\
    = 'United States of America'
mergedData.loc[mergedData['Country'] == 'United Kingdom', 'Country']\
    = 'United Kingdom of Great Britain'
mergedData.loc[mergedData['Country'] == 'Czech Republic', 'Country']\
    = 'Czechia'
mergedData.loc[mergedData['Country'] == 'Bosnia And Herzegovina', 'Country']\
    = 'Bosnia and Herzegovina'
mergedData.loc[mergedData['Country'] == 'South Korea', 'Country']\
    = 'Korea, Republic of'
mergedData.loc[mergedData['Country'] == 'Vietnam', 'Country']\

```

```

    = 'Viet Nam'
mergedData.loc[mergedData['Country'] == 'Iran', 'Country']\
    = 'Iran (Islamic Republic of)'
mergedData.loc[mergedData['Country'] == 'Moldova', 'Country']\
    = 'Moldova, Republic of'
mergedData.loc[mergedData['Country'] == 'Venezuela', 'Country']\
    = 'Venezuela (Bolivarian Republic of)'
mergedData.loc[mergedData['Country'] == 'Bolivia', 'Country']\
    = 'Bolivia (Plurinational State of)'

```

Ezután az összevonás viszonylag egyszerű volt.

```
mergedData = mergedData.merge(codes, how='left', left_on='Country',\
    right_on='Country')
```

Tehát, most már az adathalmazom tartalmazta mindegyik ország ISO nevét és kódját, még egyszer a későbbi térképvizualizációért volt fontos ez a lépés.

Ezután következett a GNI hozzáfűzése a jelenlegi adathalmazunkhoz. Mind korábban mondtam a Világbank weboldaláról származnak az adatok, CSV formátumban, 1960tól egészen 2021-ig, 2011 előttire nincs szükségem. Azonban, a formátuma sem normalizált hiszen országcódokat, majd az éveket tartalmazza, mint oszlopnevek, tehát normalizált formára kellett hozzam mielőtt hozzáfűzhettem volna az eddigiehez.

```

gni = pd.read_csv('raw/gni.csv')
gni.drop(labels=list(map(lambda x : str(x), range(1960, 2011))),\
    axis=1, inplace=True)
gni.drop(labels=['Country Name'], axis=1, inplace=True)

gni = gni.melt(id_vars=['Country Code'], value_vars=\
list(map(lambda x : str(x), range(2011, 2021))), var_name='Year',\
    value_name='GNI')\
.sort_values(['Country Code', 'Year'])
gni = gni.round({'GNI': 2})
gni['Year'] = gni['Year'].astype('int64')
gni = gni.merge(mergedData, how='right', left_on=['Country Code', 'Year'],\
    right_on=['Code', 'Year'])
gni.drop(labels=['Country Code'], axis=1, inplace=True)

```

Ezzel nagyjából készen is volt, mind az adataim felfedezése, megértése és normalizálása is, de mégis hiányzott még plusz egy lépés. Mivel a vizualizációt magyarul készítem, ezért magyar országnevekre is szükségem volt. Ismételten, az ISO formátum használata a legkönnyebb, már eleve a kódok megléte miatt, így gyorsan “beszereztem” azokat is a Wikipédiáról és hozzáfűztem a meglévő adathalmazomhoz.

```
import unicodedcsv as csv
```

```

def scrapeHunISO():
    URL = 'https://hu.wikipedia.org/wiki/ISO_3166-1'
    page = requests.get(URL)
    soup = BeautifulSoup(page.content, 'html.parser')

    table = soup.find('table', class_='wikitable')
    tableContent = table.find('tbody')
    tableRows = tableContent.find_all("tr")

    data = []
    for tableRow in tableRows:
        data.append(list(map(lambda row : str.strip(row), \
            filter(lambda row : (row != ''), tableRow.text.split('\n')))))

    return data

data = scrapeHunISO()

with open('raw/hungarianISO.csv', 'wb') as outputFile:
    writer = csv.writer(outputFile, encoding='utf-8')

    for row in data:
        writer.writerow(row)

hunISO = pd.read_csv('raw/hungarianISO.csv')
hunISO.drop(labels=['Helyi ISO kódok', 'Alpha-2', 'Numerikus'],\
    axis=1, inplace=True)
hunISO.rename(columns={'Ország / Régió': 'Country'}, inplace=True)
hunISO.rename(columns={'Alpha-3': 'Code'}, inplace=True)
merged = normalizedData.merge(hunISO, how='left', left_on='Code',\
    right_on='Code')
merged.rename(columns={'Country_x': 'Country_EN', 'Country_y': 'Country_HU'},\
    inplace=True)

Végül, a munkám eredményét egyszerűen exportáltam az általam kijelölt formá-
tumban.

merged.to_csv('normalized/normalized.csv', index=False)

```

3. Vizualizáció tervezése és elkészítése

Mindennek alapja a tervezés és minden esetben a vizualizációim elkészítésének is így fogok neki, a tervezésen belül két fő kérdésre keresem a választ:

- milyen ábrákra lesz szükségem és miért azokra?
- milyen külalakja legyen a vizualizációnak, ami viszonylag jól néz ki, de mégis kiemeli az adatot?

Eszközre vonatkozó kérdést nem igazán teszek fel magamnak. Viszonylag ki vannak alakulva már azok a módszerek, amelyekkel egy vizualizáció elkészítését végrehajtom: minden esetben Python használok az előző fejezetben tárgyalt adat-felfedezési és tisztítási részre, ami nálam az esetek legnagyobb többségében egy folyamatot jelent, végül D3-at a tényleges ábrák kirajzolására, együtt különböző JS konvenciókkal. A D3-ra azért esett már első alkalommal is a választásom, mert rendkívül magasszintű szabadságot ad az ábrák kinézetére, azonban pont ebből az okból kifolyólag borzasztó nehéz tanulási görbéje is van. Végül minden webvizualizációt nyilvánosan elérhetővé teszek a GitHub Pages ingyenes szolgáltatást kihasználva.

Vizualizációim mindegyike valamilyen weboldal és a lehető legegyszerűbb HTML formátum elkészítésére törekszek, csakis olyan elemekkel, amelyek statikus tartalommal rendelkeznek, vagy maga a szerkezetük nem fog változni, minden egyebet dinamikusan JS segítségével az adataim alapján “renderek”.

A “Hova tovább?” vizualizáció³ esetében is az volt a cél, hogy legyen egy “köszöntő” rész, ahol a vizualizáció megtekintője általánosan megismerkedhet a bemutatott témával, elmondom mi a bemutatott mérőszám, mit mér és milyen komponensek vesz figyelembe, valamilyen találó fő- és alcím mellett, majd a lehető legkevesebb “mellébeszeléssel” és zavaró tényezővel a vizualizációra térek.

Mielőtt a tényleges ábrákra térnék megemlíteném azt, hogy minden vizualizációm a következő szerkezetet követi weboldaltervezésileg:

- minden ábra egy külön JS állomány,
- van egy közös “core” csomag, ahol minden vizualizáció által használt adattagok és metódusok fognak helyet kapni,
- lesz egy “main” JS állomány, amely a teljes vizualizációt elindítja, az ábrák önmagukban kirajzolásukért nem felelnek, minden esetben a main fogja beindítani a vizualizációt a szükséges adatok betöltése után,
- bizonyos esetekben lesz egy “misc” csomagom is, olyan elemekkel, amelyek egyik ábrához, sem a közös elemekhez nem tartoznak.

A bemutatási idő rövideje, a téma és adatok egyszerűsége miatt 3 ábrára koncentráltam. Mindenképpen akartam egy ábrát, ami általánosan mutatja az életminőség alakulását, minden országra és területre vonatkozóan, az volt a cél, hogy bárki ránézésre megmondhassa hogyan alakult az életminőség az elmúlt évtizedben. Viszonylag sok országom van, minél közelebb kerülünk a jelenlegi évhez annál több és ennyi ország egyidejű megjelenítése viszonylag korlátozott. Az én személyes tapasztalataim alapján a térképes vizualizációk megértése a legegyszerűbb, ezen belül is a choropleth térképeket. Itt is erre esett a választás, rendelkezésemre állnak ország-pont párosok, statikus jellegű maga az adat, így megfelelő. Egyéb alternatívák és lehetőségek viszonylag korlátozottak, de mégis rendelkezésre állnak. Egyéb megfontolt lehetőségek:

- ponttal vagy oszloppal ábrázolt térképdiaagram,

³A teljes vizualizáció kódja megtalálható a GitHubon: github.com/wheresouldigoviz

- területekre bontott (pl. Észak- és Dél-Amerika, Európa, ázsiai országok) oszlopdiagrammok,
- és a legerősebb alternatíva a körkörös oszlopdiagram készítése, ahol az oszlopok 360 fokban a kör mentén helyezkednek el. Mindegyiknek megemlített alternatívának megvan a maga előnye és hátránya is.

A ponttal és oszloppal ábrázolt térképdiagram majdnem teljesen ugyanaz, mint a chloropeth térkép, csak egy másik megközelítés, de véleményem szerint nehezebben értelmezhető és kevésbé “izgalmas”, jobban szeretek magukkal ragadó színeket nézni területeken, amelyek ránézésre könnyen értelmezhetőek, minthogy értelmezsem egy adott pont nagyságát, vagy oszlop magasságát miközben az adott országot keresem a térképen, persze egy “tooltip” bevezetése ebben a részben segíthet.

Területekre bontott oszlopdiagrammok haszontalanok ilyen bemutatók esetében, a területek összeállítása nagyon nehézkes a végső megtekintő számára.

A körkörös oszlopdiagram elkészítése nagyon vonzott, már korábban is használtam, azonban azért döntöttem mégis a térképdiagram mellett, mert mindenképpen akartam és a következő diagram amúgy is a téma korlátozottsága miatt oszlopdiagram lesz. Körkörös oszlopdiagram esetében az országokat pontszám szerint rendeztem volna, és ahogy haladunk a 360 fok felé egyre kisebb oszlopok lennének. Az összehasonlíthatóság megint kétséges, de mindenképp vizuálisan hivatkozó ábra.

Tehát, a chloropeth térkép elkészítésére esett a választás több okból kifolyólag is:

- könnyen értelmezhetőek,
- könnyen összehasonlíthatóak,
- és látványosak az adatok. Egyedüli hátránya, hogy sok esetben nehéz a kis területű országokat “felismerni” a térképen.

Az, hogy ténylegesen hogyan készítettem a térképdiagramomat eléggé könnyen végigkövethető a kellő tapasztalattal a mellékelt *GitHub* “repo-n” keresztül. Jelen esetben a `viz_map.js` állomány tartalmazza az ábrát, túl hosszú lenne bármelyiket is bemutatni.

Minden ábra elkészítésének is van egy általános struktúrája:

- először elvégzek minden olyan műveletet és adattag inicializálást, amire a későbbiekben általános szinten szükség lesz,
- majd következhet az ábra kirajzolásának inicializálása, minden ábraelemmel: ábra, jelmagyarázat, tengelyek, esetleges befolyásoló mezők,
- végül az ábra “elindítása”.

A következő ábra célja, hogy belemerüljön a részletekbe. Miért alacsony egyes országok pontszáma, míg magas másoké? Melyek azok a tényezők, amik jobba tesznek egy adott országot egy adott évben. Tömören, itt a hangsúly az elemezhetőségen volt, hogy az adathalmazomat minél lebontottabban darabokra

szedve tudja a felhasználó megtekinteni, kiragadni belőle a legfontosabb részleteit és azokból következtetéseket levonni. Annyiban akartam megkönnyíteni munkáját, hogy az ábra tengelyei segítségével és az oszlopok irányával jelzem azokat a komponenseket, amelyek az életminőséget csökkentik és amelyek növelik azt.

Ebben az esetben túl sok alternatív lehetőség nem merülhet fel, egyedül a pókábra, ami ugyancsak megfontolásra került, azonban annak az értelmezése sokkal, de sokkal nehezebb lett volna, abból az okból kifolyólag, hogy vannak olyan komponenseim, amelyeket pozitív előjellel kell értelmezni, és vannak olyanok, amelyek negatívval, így alkalmazhatatlanná válik véleményem szerint a pókábra ebben az esetben.

Így részben komponens szerint rendezett vízszintes oszlopdiagram mellett döntöttem. Azért került csak részben rendezésre, mert a klímaindex csak 2015 után képezi a mutató részét, így nem akartam, hogy egyéb évek esetében egy “lyuk” legyen az ábra közepén. Könnyű az adatok leolvasása, a tooltip még segít is ezen, de még könnyebb megérteni azt, amit az olvasó valójában lát. Az adat könnyen felfogható ezen az ábrán keresztül. Talán a vizsgálódás időigényessége az egyedüli hátránya ennek az ábrának.

Évre és országra lebontva használható az ábra, még egyszer itt a lényeg az egyes országok esetében a “lehúzó” komponensek azonosítása volt a cél, illetve az, hogy miben kellene javítaniuk ezek az országoknak. A tényleges ábrát létrehozó D3js kód a `viz_components.js` állományban található.

Végül, a harmadik ábra a korábban kiszámolt GNI/fő és életminőség mutató közötti kapcsolat megjelenítése. Itt arra törekedtem, hogy minél inkább megtartsam azt a formát, ahogyan valójában egy ilyen összefüggés vizsgálatot elvégezne. Tehát megtartottam a jól ismert pontdiagrammot és arra egy regressziós görbét rajzoltam fel, bemutattva, hogy a másodfokú polinomiális modellem valójában hogyan és miként illeszkedik a mintámra. Természetesen, ezt az összefüggést az olvasó minden évre vonatkozóan elvégezheti. Ámbár lehetőség van arra is, hogy egyesével országokra vonatkozóan az olvasó megtekintse a tooltip segítségével, hogy milyen GNI/fő és életminőség adatokkal rendelkezik, de ez másodlagos, az igazi cél a két mutató közötti összefüggés minél egyszerűbb kikövetkeztetése, ami a görbe alakulásával már ránézésre megmondható. Könnyedén értelmezhető és leolvasható az, hogy hogyan mozog együtt a GNI/fő és az életminőség, hogyan magyarázza a gazdasági teljesítmény az életminőséget és legfőképpen, hogy meddig, milyen szintig.

Ez volt az az egyedüli ábra, ahol semmilyen alternatívában nem gondolkodtam, hiszen a használt ábrán kívül szerintem mondható az, hogy nincs olyan, amivel ennyi egyszerű a fentebb említett következtetések levonása. Valójában hátrányt sem tudnék önmagamtól megemlíteni.

Ahogy a korábbi ábrák esetében, itt is megemlíteném, hogy a teljes ábrát elkészítő D3js kód a `viz_regression.js` állományban található.

4. Mit is tanulhatunk az ábráimból?

Ebben a részben kisseretnék egyrészt térni arra is, hogy mit lehet tanulni magából a vizualizációból a következtetések levonása után, illetve, hogy én, mint programozó és tervező mit tanultam egy újabb vizualizációs projekt elkészítéséből.

Amennyiben az olvasó teljes egészében megtekinti⁴ a vizualizációmát nagyon fontos és érdekes következtetéseket tud levonni. Amennyiben a “köszöntő” térképábrát vizsgálja, a legfontosabb következtetés az, hogy 2020-ban, akkora tengernyi különbségek a különböző országok és területek között nincs, mondhatni az életminőség az adatokat szolgáltató országok esetében azonos, illetve nagyon alacsony a szórás. Természetesen vannak olyan “kimagasló” területek, ahol ez nem igaz, példaképpen az afrikai országok, de ezeket mindenki könnyűszerrel teljesen magától is képes kitalálni. Ebből kikövetkeztethető, hogy mindegy hogy az ember hova tekint, akár üzleti, akár magánmegfontolásból nagyon hasonló életkörülményeket fog találni, ami arra enged alapvetően következtetni, hogy a lehetőségek is nagyjából azonosak, az azonos jövedelemmel rendelkező országok esetében. Illetve, ami még mondhatni teljesen magától értetődő, hogy az elmúlt évtizedben tisztán állítható az, hogy a világon mindenhol az életminőség növekedett, illetve a különbségek pedig csökkentek az egyes területek között. Ami még kiemelendő az Kína, Kína 2012-ben utolsó helyen állt, 2020-ra ez abszolút megváltozott és már valójában nem annyira rossz Kínában élni, hiába a diktatúra. Ugyanakkora, ezt nagyban befolyásolhatja a nyugati terjeszkedés az országba.

A második ábra alapján minden országról lehetne néhány bekezdés rövid elemzést összeállítani, azonban rövidre fogva megint Kínát emelném ki. Ha arra kérnék bárkit, hogy nevezze meg miért “rossz” Kína valószínűleg két dolgot említene: a diktatúra és a környezetszennyezés. Pont az utóbbi miatt került utolsó helyre 2012-ben az ország. A környezetszennyezés már majdnem önmagában egy negatív életminőségi pontszámra csökkentette Kínát és nem volt, amit pozitívként lehetett volna az országnak elismerni. Azonban az évtized végére minden területen sikerült javítania és a környezetszennyezési mutatóját jelentős mértékben csökkentette, majdnem megfelelte.

Ami Romániáról elmondható a második ábra alapján az az, hogy látható egy általános javulás az életminőségben, azonban vannak olyan fontos területek, ahol stagnált vagy éppenséggel rontott a helyzetén. Ami pozitív, hogy ma már több terméket tudunk megvásárolni, mint 8 évvel ezelőtt átlagfizetéssel, jobb egészségügyi ellátást kapunk, azonban ma már érthető okokból kifolyólag sokkal több időt töltünk a napjainkból ingázással és emiatt környezetünket is, a nem megfelelő közlekedési módszerek miatt jobban szennyezzük. De még mindig viszonylag sokat kell dolgozniuk a romániai lakosoknak ahhoz, hogy olyan nélkülözhetetlen vagyonelemeket, mint egy lakás megengedhessenek magunknak, kisebb-nagyobb segítséggel.

Végül, számomra a legérdekesebb és tanulságosabb az utolsó regressziós ábra volt.

⁴A teljes vizualizáció megtekinthető: rockdonald2.github.io/wheresouldigoviz

Egyetlen perc alatt bebizonyította azt, hogy igazunk akkor, amikor azt mondjuk, hogy a GNI növeli az életminőséget, de csak egy adott szintig. Ez a gondolat nem csak országokra, de akár mikro szinten is igaz. Egy újabb egy lej boldogabbá tesz és jobbá teszi az életet, de csak valameddig. Pont ez tükröződik itt is, hiszen a görbe hozzávetőlegesen 65 ezer dollár fölött visszakonyul és afölött pedig már nem vezet életminőség pontszámbeli növekedéshez. Ez a következtetés minden évre vonatkozóan igaz, azonban a GNI “határ” az változó. Könnyedén kikövetkeztethető az, hogy megéri szegényebb országokba fektetni, hiszen ez által növekszik az életminőségük. A “kakkuktojás” minden évben Szingapúr, hiába a hatalmas GNI, az életminőség elenyésző, hatalmas a társadalomban a jövedelembeli szakadék. Az ország egy része nagyon jól, míg más része nagyon rosszul él.

Összefoglalóképpen, a D3js könyvtár segítségével elkészített vizualizációmon keresztül, ami a *Numbeo* adatgyűjtő vállalat adathalmazát használja fel, bárki könnyedén megértheti, értelmezheti és akár terjesztheti azt, hogy hogyan alakul az életminőség a Föld körül, melyek azok a tényezők, amelyek az adott országot lehúzzák és melyek azok amelyek felemelik, illetve, hogy milyen összefüggés ismerhető fel a GNI és az életminőség között.