Healthcare Analytics 1 (2021) 100005

Contents lists available at ScienceDirect

Healthcare Analytics

journal homepage: www.elsevier.com/locate/health

SCORECOVID: A Python Package Index for scoring the individual policies against COVID-19

Yoshiyasu Takefuji

Faculty of Data Science, Musashino University, 3-3-3 Ariake Koto-ku, Tokyo 135-8181, Japan

ARTICLE INFO

Keywords:
COVID-19
Scoring policy
Python Package Index (PyPI)
Pandemic
Mitigation
Open-source software

ABSTRACT

This study proposes SCORECOVID, a new Python Package Index (PyPI) for scoring individual policies against covid-19 and mitigating the pandemic. The new PyPI package consists of two modules. The first module automatically scrapes the latest information on the number of deaths and population by COVID-19 to score individual policies for a given country. The second module calculates the score by dividing the number of deaths by the population in millions. The Federal Communications Commission (FCC) in the US estimates the economic value of a statistical life to be $9.5 million per individual. The higher the number of deaths, the greater the economic loss. To use the best policies to reduce the number of deaths, we should adopt measures and methods from exceptional countries with high scores. The proposed method reveals two groups: a high-scored group and a low-scored group. The number of deaths is an indicator of economic and health policy scores. SCORECOVID is the world's first open-source policy scoring tool for COVID-19. It is designed to help many countries utilize state-of-the-art analytics methods to effectively mitigate the COVID-19 pandemic.

1. Introduction

With the rapid advancement of open-source software, open-source programs are dominating many applications such as file servers, mail servers, web servers, and artificial intelligence frameworks. In order to use open-source programs, it is essential to use a package manager. Packaging plays an important role in making open-source libraries accessible to users. Therefore, this paper briefly introduces open-source packaging that many researchers use around the world. The following packaging managers will be introduced: npm, Maven, and Go.

The popularity of a programming language can be measured by two indicators. Two indicators show that the Python is the most popular language. The first goal of this paper shows how to package a Python library in public with PyPI. The second goal is to show advantages of the proposed algorithm over the existing algorithms on health policy scoring. In scoring health policies on COVID-19, the number of deaths due to COVID1-19 per population in millions is used in this paper while the existing algorithms use the number of infected cases with a dataset. This paper addresses what was missing in the existing scoring schemes or algorithms and shows advantages of the proposed algorithm over the existing algorithms.

1.1. Open-source packaging and language popularity

There are many package managers for installing open-source software codes [1]. The largest package manager in the world is npm managing 1.85M packages. npm is a package manager for the JavaScript programming language. npm, npm is the default package manager for the JavaScript runtime environment Node.js.

The second largest package manager is Maven managing 418K packages [2]. Maven is a build automation tool used primarily for Java projects. Maven can also be used to build and manage projects written in C#, Ruby, Scala, and other languages. The Maven project is hosted by the Apache Software Foundation.

The third package manager is Go managing 382K packages [3]. Go is a statically typed, compiled programming language designed at Google. Go is syntactically similar to C, but with memory safety, garbage collection, structural typing, and CSP-style concurrency.

Many developers are always wondering what is the best programming language to develop their products in terms of cost, reusability, development time, security, durability, maintenance, future prospects, and many other factors. Determining which programming language is the "most widely used" is difficult because its meaning varies from context to context. Therefore, two indicators are often used to measure the popularity of a programming language: PYPL [4] and TIOBE [5].

The PYPL (PopularitY of Programming Language) index is an indicator based on Google Trends, reflecting the developers' searches for "<programming language> tutorial", instead of what pages are available

According to Wikipedia, the TIOBE programming community index is a measure of popularity of programming languages, created and maintained by the TIOBE Software.

The latest PYPL index as of September 2021 is as follows:

E-mail address: takefuji@keio.jp.

https://doi.org/10.1016/j.health.2021.100005
Received 15 August 2021; Received in revised form 18 September 2021; Accepted 18 September 2021
Available online xxxx

Rank	Language	Share	Trend
1	Python	29.48%	−2.4%
2	Java	17.18%	+0.7%
3	JavaScript	9.14%	+0.8%

The TIOBE index as of September 2021 is as follows,

Sept. 2021	Programming Language	Ratings	Change
1	C	11.83%	−4.12%
2	Python	11.67%	+1.20%
3	Java	11.12%	−2.37%

The de-facto standard package artificial intelligence manager is the fourth largest package manager in the world, PyPI managing 370K packages [6]. PyPI is a Python language package manager. Python is a language of the first ranked by PYPL index and the second ranked by TIOBE index as of June 2021. The proposed algorithm uses a Python language with the PyPI packaging manager.

1.2. Advantages of the proposed algorithm

This paper shows how to package a new PyPI library or an executable application using several open-source libraries. Introduced scorecovid is an example of the new PyPI package for scoring individual policies against COVID-19 [7].

Scoring the performance of individual policies is calculated by dividing the number of deaths due to the covid-19 by the population in millions [8]. The conventional algorithms use the number of infected cases [9–11]. The economic value of a statistical life (VSL) varies from country to country, but the FCC estimates it at $9.5 million per person [12]. In Japan, the VSL is from 217 million yen to 264 million [13].

The existing algorithms based on dataset analysis for scoring individual policies can only indicate the spread of COVID-19 with infected cases but they cannot indicate the effectiveness of the policies. A dataset is composed of instances with necessary parameters. The fatal drawback of the existing algorithms lies in that if the necessary parameters are lacking an important feature or a parameter, they cannot measure policy performance. The existing algorithms do not have a way to measure how well policies are implemented or enforced, nor do they measure the degree of compliance with official policies [9].

The Oxford Covid-19 Government Response Tracker (OxCGRT) collects systematic information on policy measures that governments have taken to tackle COVID-19 [9]. Although OxCGRT covers more than 180 countries and are coded into 23 indicators or features, it lacks a key indicator, “digital fences” [8]. In data science analysis, if a dataset lacks important indicators, it cannot be analyzed well forever. OxCGRT’s scoring is based on the number of daily infected individuals while that of the proposed paper is based on the number of daily deaths.

Lazarus et al. proposed COVID-19 Assessment Scorecard (COVID-SCORE) for keeping governments accountable [10,11]. However, their scorecard does not allow them to reach the key indicator, “digital fences” since the scorecard is based on a multiple-choice questionnaire.

As far as we know, there are no objective evaluation indicators for COVID-19 policies based on data science. The higher the number of deaths, the greater the economic loss or the greater the poor policy performance. To use the best policies to reduce the number of deaths, we should adopt measures and methods from exceptional countries with high scores.

This paper proposes SCORECOVID, a new Python Package Index (PyPI) for scoring individual policies against covid-19 and mitigating the pandemic. Scoring is based on the number of daily deaths per the population in millions [8]. The proposed algorithm allows us to compare individual policies and to discover the high-performance countries. In the high-performance countries, we must investigate what policies or features may influence the performance in order to mitigate and end the pandemic. In other words, we must create a new dataset by taking account of the investigated features or indicators for the future work.

2. PyPI packaging

For PyPI packaging, the necessary procedures are composed of seven steps:

1. Create a new github repository and make a README.md.
2. Build a new program for PyPI packaging.
3. Create a setup.py file as shown in Fig. 1.
4. Modify _init_.py and _main_.py
5. Create dist directory and build directory using setup.py.
6. Create a PyPI account.
7. Upload files in dist directory.

Seven steps are detailed as follows.

1. If you have your account on github.com site, create a new repository with README.md file. If you have no account on github, create a new account.

The following site is an example of scorecovid:

The example can be examined by a reviewer.

2. In this paper, scorecovid is a Python program of scorecovid.py as PyPI packaging candidate. The program is shown in Fig. 2.

3. Create a setup.py file for creating three files: .whl file, .egg file, and .tar.gz file.

The setup.py template is stored at the following site:

https://raw.githubusercontent.com/ytakefuji/score-covid-19-policy/main/setup.py

Red-colored nine lines must be modified for your packing.

4. Modify _init_.py and _main_.py. Create a src directory. Then, move two files to src. A template of _init_.py and that of _main_.py are shown.

```
$ cat _init_.py
import scorecovid
import _main_
$ cat _main_.py
import scorecovid
```

To create a src directory:

```
$ mkdir src
```

Move two files to src directory.

```
$ mv _*.py src
```

Move scorecovid.py file to src directory.

```
$ mv scorecovid.py src
```

5. Use setup.py for generating the necessary files in dist directory and build directory.

Run the following line command:

```
$ python setup.py install
$ python setup.py sdist bdist_wheel
```

In order to test a new PyPI package, scorecovid, run the following command. Create a countries file.

Then, with or without countries file, run the following command.

```
$ scorecovid
```

It should show the scores of 12 countries.

6. In order to create a new account of PyPI. access to the following site:

https://pypi.org/account/register/

7. Install twine library for checking the new PyPI package and uploading the files to PyPI.

```
$ pip install twine
```

```
import setuptools

with open("README.md", "r", encoding="utf-8") as fh:
    long_description = fh.read()

setuptools.setup(
    name="scorecovid",
    version="0.0.6",
    author="yoshiyasu takefuji",
    author_email="takefuji@keio.jp",
    description="A package for scoring policies of covid-19",
    long_description=long_description,
    long_description_content_type="text/markdown",
    url="https://github.com/ytakefuji/score-covid-19-policy",
    project_urls={
        "Bug Tracker": "https://github.com/ytakefuji/score-covid-19-policy",
    },
    classifiers=[
        "Programming Language :: Python :: 3",
        "License :: OSI Approved :: MIT License",
        "Operating System :: OS Independent",
    ],
    package_dir={"": "src"},
    py_modules=['scorecovid'],
    packages=setuptools.find_packages(where="src"),
    python_requires=">=3.6",
    entry_points = {
        'console_scripts': [
            'scorecovid = scorecovid:main'
        ]
    },
)
```

Fig. 1. setup.py Python program.

In order to check three files, run the following command:

$ twine check dist/*

Finally, upload three files to PyPI site.

$ twine upload dist/*

If PyPI shows the link site, a new package was successfully uploaded.

3. scorecovid.py

scorecovid is very useful for scoring the performance of individual health policies of countries. From the perspective of an evidence-based approach, we should learn from countries that have good infection control measures and methods during the pandemic[7].

The calculated result forces us to navigate and update a policy against the pandemic. Scoring the performance of individual policies is calculated by dividing the number of deaths due to the covid-19 by the population in millions[8].

scorecovid is a Python program composed of several open-source libraries such as scraping library and pandas DataFrame. scorecovid automatically scrapes the latest information on total deaths and population from websites on the Internet and generates a result.csv file.

4. How to install and run scorecovid

Prepare a list of countries in a file entitled “countries” for scoring, for example:

South Korea, India, Brazil, France, New Zealand, Taiwan, Sweden, Japan, United States, Canada, United Kingdom, Israel

In the countries file, country names should be separated by commas. In order to run scorecovid, run the following command for installation:

$ pip install scorecovid

pip is a PyPI package manager command for installation.

Type the following command to run scorecovid, the result will be shown on the screen:

$ scorecovid

scoring the following 12 countries...

score is created in result.csv

date is 2021-06-15

```
import requests,re
import pandas as pd

def main():
 url='https://www.worldometers.info/world-population/population-by-country/'
 print('scraping population...')
 page=requests.get(url)
 df = pd.read_html(page.text)[0]
 df.columns.values[1]='Country'
 df.columns.values[2]='Population'
 #df = pd.read_html(page.text,flavor='html5lib')[0]
 df.to_csv('pop.csv')
 print('pop.csv was created')
 print('downloading total_deaths.csv file')
 import subprocess as sp
 sp.call("wget https://github.com/owid/covid-19-
data/raw/master/public/data/jhu/total_deaths.csv",shell=True)
 p=pd.read_csv('total_deaths.csv')
 date=p['date'][len(p)-1]
 print('countries file was read...')
 d=open('countries').read().strip()
 d=d.split(',')
 print('scoring the following ',len(d),' countries...')
 print(d)
 dd=pd.DataFrame(
  {
   "country": d,
   "deaths": range(len(d)),
   "population": range(len(d)),
   "score": range(len(d)),
  })
 pp=pd.read_csv('pop.csv')
 print('calculating scores of countries\n')
 print('score is created in result.csv')
 print('date is ',date)
 for i in d:
  dd.loc[dd.country==i,'deaths']=int(p[i][len(p)-1])
  dd.loc[dd.country==i,'population']=int(pp.loc[pp.Country==i,'Population']/1000000)

dd.loc[dd.country==i,'score']=int(dd.loc[dd.country==i,'deaths']/dd.loc[dd.country==i,'populatio
n'])
 dd=dd.sort_values(by=['score'])
 dd.to_csv('result.csv',index=False)
 dd=pd.read_csv('result.csv',index_col=0)
 print(dd)
 sp.call("rm total_deaths.csv pop.csv",shell=True)

if __name__ == "__main__":
 main()
```

Fig. 2. scorecovid.py Python program.

country	deaths	population	score
New Zealand	26	4	6
Taiwan	460	23	20
South Korea	1993	51	39
Japan	14150	126	112
India	379573	1380	275
Canada	25943	37	701
Israel	6428	8	803
Sweden	14574	10	1457
France	110692	65	1702
United States	600285	331	1813
United Kingdom	128181	67	1913
Brazil	490696	212	2314

5. Discussion

While the conventional scoring policies [9–11] is based on the number of daily infected individuals (cases), the proposed scoring is calculated by the number of daily deaths per the population in millions. In dataset analysis and policy analysis, necessary indicators must be provided. Without key indicators, it is impossible to discover what is the most effective indicators of policies against COVID-19.

The conventional scoring policies [9–11] failed in lacking a key indicator, "digital fences". Until the vaccine was available, many countries except countries using digital fences did not use any effective policy against COVID-19.

Digital fences play a key role in mitigating the COVID-19 pandemic [8.14,15] as long as infection testing is available. The stronger the digital fence, the more effective the policy will be against COVID-19 [8, 14,15].

6. Conclusion and future work

This paper proposes the first open-source tool for scoring individual policies against COVID-19. The number of deaths is an indicator of economic and health policy scores. Important policy issues in response to a pandemic such as this one would include a comprehensive review of the healthcare delivery system and the development of emergency response systems, strengthening of vaccine development as a means of economic security, flexible responses to various regulations related to vaccination, and focused and flexible fiscal spending based on data analysis and the vision of a future society.

Several indicators or features (vaccination rates, number of ICU beds, etc.) should be evaluated in the future work. Cost-effectiveness analysis based on such a score for the degree of effort in addressing each feature will make it possible to verify each effectiveness in the future.

This paper discloses an objective scoring system using the ongoing data for pandemic containment and explains the open-source use of PyPI for two reasons. First, the essence of science is repeatability, and this paper provides an opportunity to do so by releasing the program. Secondly, this paper provides the opportunity to do so by releasing the program, and also offers the possibility of modification to the readers through opensource. The proposed simple method is for scoring COVID-19 policies, normalized by the mortality rate per 1,000,000 people, but there is room for improvement depending on future research and data provided in the future.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] npm, Node package manager, https://www.npmjs.com.
[2] maven, the maintainers of Maven Central Repository, http://maven.org/.
[3] go, go.dev is the hub for Go users providing centralized and curated resources from across the Go ecosystem, https://pkg.go.dev.
[4] PYPL, PopularitY of Programming Language, https://pypl.github.io/PYPL.html.
[5] tiobe, tiobe-index, https://www.tiobe.com/tiobe-index/.
[6] PyPi, Find, install and publish Python packages with the Python Package Index, https://pypi.org/.
[7] scorecovid, A package for scoring policies of covid-19, https://pypi.org/project/scorecovid/.
[8] Y. Takefuji, Correspondence, N. Engl. J. Med. 384 (2021) e66, http://dx.doi.org/10.1056/NEJMc2101280.
[9] T. Hale, N. Angrist, R. Goldszmidt, et al., A global panel database of pandemic policies (Oxford COVID-19 Government Response Tracker), Nat. Hum. Behav. 5 (2021) 529–538, http://dx.doi.org/10.1038/s41562-021-01079-8.
[10] J.V. Lazarus, A. Binagwaho, A.A.E. El-Mohandes, et al., Keeping governments accountable: the COVID-19 Assessment Scorecard (COVID-SCORE), Nat. Med. 26 (2020) 1005–1008, http://dx.doi.org/10.1038/s41591-020-0950-0.
[11] J.V. Lazarus, S. Ratzan, A. Palayew, F.C. Billari, A. Binagwaho, S. Kimball, et al., COVID-SCORE: A global survey to assess public perceptions of government responses to COVID-19 (COVID-SCORE-10), PLoS One 15 (10) (2020) e0240011, http://dx.doi.org/10.1371/journal.pone.0240011.
[12] FCC, Remarks of FCC Commissioner Michael O'Rielly TPRC 44, https://docs.fcc.gov/public/attachments/DOC-341544A1.pdf.
[13] Masaaki KAWAGOE, How Can japanese extended longevity be evaluated? https://www.esri.cao.go.jp/jp/esri/archive/bun/eibun/eibun197.pdf.
[14] S.C. Chen, Taiwan's experience in fighting COVID-19, Nat. Immunol. 22 (2021) 393–394, http://dx.doi.org/10.1038/s41590-021-00908-2.
[15] Dyani Lewis, Contact-tracing apps help reduce COVID infections, data suggest, Nature 591 (2021) 18–19, http://dx.doi.org/10.1038/d41586-021-00451-y.