Catalogue no. 11-522-x
ISSN: 1709-8211

Proceedings of Statistics Canada Symposium 2021
Adopting Data Science in Official Statistics to Meet Society's Emerging Needs

Collaborative Data Science within Government of Canada: Development of R libraries for common tasks with Open Canada data

by Dmitry O. Gorodnichy and Patrick Little

Release date: October 29, 2021

Statistics Canada Statistique Canada

Canada

Proceedings of Statistics Canada Symposium 2021
Adopting Data Science in Official Statistics to Meet Society's Emerging Needs

Collaborative Data Science within Government of Canada: Development of R libraries for common tasks with Open Canada data

Dmitry O. Gorodnichy[1] and Patrick Little [2]

Abstract

Many Government of Canada groups are developing codes to process and visualize various kinds data, often duplicating each other's efforts, with sub-optimal efficiency and limited level of code quality reviewing. This paper informally presents a working-level approach to addressing this technical problem. The idea is to collaboratively build a common repository of code and knowledgebase for use by anyone in the public sector to perform many common data science tasks, and, in doing that, help each other to master both the data science coding skills and the industry standard collaborative practices. The paper explains why R language is used as the language of choice for collaborative data science code development. It summaries R advantages and addresses its limitations, establishes the taxonomy of discussion topics of highest interested to the GC data scientists working with R, provides an overview of used collaborative platforms, and presents the results obtained to date. Even though the code knowledgebase is developed mainly in R, it is meant to be valuable also for data scientists coding in Python and other development environments.

Key Words: Collaboration; Data science; Data Engineering; R; Open Government; Open Data; Open Science

1. Introduction

Data is the 'electricity' of the 21st century. It is everywhere. Everybody is using it, and one may not need special training or certification to work with data. However, in contrast to working with electricity, working with data relies on the use of data processing tools (codes), the number and complexity of which increases daily, and which are developed by other data practitioners. In other words, data science - by its nature - is the science that is fueled by collaboration of data scientists and that heavily relies on the quality of the shared data processing codes.

Globally, and in the Government of Canada (GC) in particular, data practitioners come from many different backgrounds and may not have equal level of code programming training, which hinders the development of high-quality (efficient, scalable and re-usable) codes for data science problems. This gap and the need for collaboration for data scientists, specifically those coding in R language, has been raised at the 2021 GC Data Conference Workshop on Data Engineering (Gorodnichy, 2021) in February of this year. This presentation describes a pan-government collaborative approach that has started following this workshop in order to address the identified need.

First (in Section 2), we review one the main challenges in data science, which is data engineering and which triggered the creation of the R4GC community of practice - the community that brings together data practitioners interested in using (or learning) the R language. The reasons why R is chosen as the language of choice for building the data science collaboration are explained in Section 3. Then (in Section 4), we describe the collaboration portals that have been set in support of the R4GC community activities and knowledgebase and overview key repositories and discussion threads that have been created there. Finally (in Section 5 and Appendix), we summarize the main outputs produced by the community to date. Whereas this paper serves as an introduction to the R4GC community efforts, the complete collection of the community knowledgebase is maintained as a shareable, free to use and collaboratively edited book (The R4GC Book) hosted at https://open-canada.github.io/r4gc. Feedback and contributions to this book are welcome, noting that only unclassified public domain knowledge is archived there.

[1]Dmitry Gorodnichy, Data Science Division, Chief Data Office, Canada Border Services Agency (Dmitry.Gorodnichy@cbsa-asfc.gc.ca)
[2]Patrick Little, Open Government Portal, Office of the Chief Information Officer, Treasury Board of Canada Secretariat (Patrick.Little@tbs-sct.gc.ca). Disclaimer: The views and opinions expressed in this article are those of the authors and do not necessarily reflect the official policy or position of any agency of the Government of Canada.

2. Data Engineering challenge

This effort has started from addressing the problem of data engineering, where it is understood - in analogy with the definition of software engineering by IEEE (Bourque and Fairley, 2014) - as the field of science and technology that deals with *"developing scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation"* of data-driven systems and solutions; or - in analogy with the definition of software engineering at Google (Winters et al., 2020) - as the field that *"encompasses not just the act of writing code* [for data analysis, in our case]*, but all of the tools and processes an organization uses to build and maintain that code over time"*.

Whereas globally the spectacular growth of data science tools development is overwhelmingly attributed to the collaborative open nature of the current data science code development practices, Canadian government is at the early stages of adopting open source industry standards for data coding and reporting. Major shift towards enabling and promoting such practices within the government has started recently, when Government of Canada adopted a number of policies in support of Digital, Open Science, and Open Government[3], when Shared Services of Canada deployed a number of collaboration platforms that are now available to all GC organizations, and when more GC IT organizations are embracing the use of open-source data science tools such as R and R Studio.

3. R advantages

A quick review of the presentations made at the 2021 International Methodology Symposium showed that about one third of projects presented at the symposium were done using R and RStudio. If counting the percentage of projects dealing with data visualization, including geo/spatial mapping, then this number becomes close to two thirds of all presented projects. Clearly, even though Python remains the most frequently used language for machine learning and complex data analysis tasks, R has become de-facto the language of choice in many government departments when it comes to developing interactive reports, web applications and complex visualizations, thanks to such its popular packages as "*rmarkdown*", "*ggplot*" and "*shiny*".

Other advantages of R highlighted at the Symposium include: common tidy data approach shared across all R packages; peer-reviewing and curation of packages by CRAN (Comprehensive R Archive Network), which is facilitated through the "*devtool*" R package that is specifically designed for this purpose; and RStudio-led movement for R education and deployment. All of these are in contrast to the 'wild-west' development of Python packages (which is a popular expression even among Python users), and are very important for enabling collaborative development of data science codes and knowledgebase. To summarize, Table 1 presents the Top 10, most quoted by the R4GC community, reasons to use R for data science.

Table 1: The R4GC community Top 10 reasons to use R for Data Science.

1.	Advanced graphics with *'ggplot2'* and its extensions
2.	Automated generation of reports, tutorials, and textbooks with *'rmarkdown'*
3.	Streamlined package development with *'devtools'*
4.	Development and deployment of interactive interfaces, web applications and dashboards with 'shiny'
5.	Convenient for geo/spatial computation and visualization
6.	Common tidy data design shared across packages
7.	Curated peer-tested repo of packages at CRAN (Comprehensive R Archive Network)
8.	RStudio IDE (Integrated Development Environment) on desktop and cloud (at *rstudio.cloud*)
9.	Full support and inter-operability with Python from the same IDE
10.	Global RStudio-led movement for R education and advancement (at *rstudio.com*)

[3] https://www.tbs-sct.gc.ca/pol/doc-eng.aspx?id=32602

3.1 Tackling the limitations of R

The main deficiency of R (in particular, when compared to Python) is that many R functions and packages are not developed to be memory- or processing time- efficient, and "by default", i.e., unless special care is taken in developing the code, most codes written in R are very slow and consume prohibitively large amount of memory, making it often impossible to run data processing codes on ordinarily laptops (with not more than 16 GB of RAM) such as those used by majority of GC employees. This is because all variables in R, including data array and data frame variables (which are naturally very large) are passed to a function by copying the entire variable content from one memory location to another , and then copying it back after the function completes its operation. In other programming languages, such as Python, C++, or Java, such operations normally are not done by passing the entire object to a function, but rather by passing only a pointer to (or memory address of) an object that needs to be processed by the function, which is much faster and does not consume a lot of memory. This deficiency however can be practically entirely eliminated, should the R code be developed using the "efficiency-by-design" methodology that is offered by the 'data.table' R package. This is why much emphasis in the R4GC community discussions is put on encouraging the community members to use 'data.table' class by default (or always) instead of base 'data.frame' class.

Another deficiency of R is that object-oriented programming (OOP) is not native in R. There are a number of ways to implement an object in R - using S3, S4 and R6 classes, however each of them has its own limitations. This deficiency however is not critical, because fast and memory-efficient codes can still be developed with and without OOP, and, if needed, a portion of the OOP code can be also built in Python (or C++) and then used from within R using 'Rcpp' and 'reticulate' packages.

4. Overview of Collaborative Platforms

4.1 GCcode group: r4gc - https://gccode.ssc-spc.gc.ca/r4gc

GCcode is the GitLab solution that is accessible from within the GC network. As such, it allows one to view and update (pull and push) codes and documentation with a single click of button on a GC laptop from an RStudio. A step-by-step tutorial on how to do it is developed. The 'r4gc' group has been created in GCcode, where R4GC community codes, tutorials and other resources are gathered. It contains three main folders:

- */codes.* - This is where "raw" (not-reviewed, unedited) R codes contributed by GC community are uploaded. Currently, this includes codes for analyzing and visualizing PSES (Public Service Employee Survey) results, ATIP requests, COVID-19 statistics, and various codes for ease of day to day work and maintenance. Some codes are readily available to become packages, some are short code snippets taken from various blogs, question and answer portals, such as www.stackoverflow.org and www.rseek.org, and open-source textbooks.

- */gc-packages.* - This is where the work on packages being developed from the submitted "raw" codes is happening. Currently it includes repositories for building packages to process PSES results, COVID-19 data, and the utility functions package for data engineering and efficient data processing.

- */resources.* - This where the rest of knowledge-base is gathered, including the tutorials, slides, and codes presented at the community weekly ‘Lunch and Learn’ meetups.

4.2 GCcollab group: R4GC (Use R!) - https://gccollab.ca/groups/profile/7391537/R4GC

GCcollab allows one to participate in the discussion from within and outside GC network (for registered users). This makes it convenient for gathering information from any sources, including those that may not be available from within the GC network. A group called “R4GC” (originally called “Use R!”) is created in GCcollab, where a number of discussion threads have been established to address the topics of highest interest for the R4GC community. They are listed in Table 2, divided into four main categories (parts).

Table 2: Taxonomy of R4GC community discussion topics

Part I: General discussions	*Part III: Visualization and Reporting*
1. Why R?	12. Literate programming and automated reports with 'rmarkdown'
2. Learn R: *R*ight way!	13. Data visualization with 'ggplot2' and its extension
3. Open-source textbooks	14. Geo/Spatial coding and visualization in R
4. Events and forums for R users	15. Interactive interfaces, applications and dashboards with 'shiny'
5. Using R with GC data infrastructure	16. Interactive html with 'plotly', 'Datatable', 'reactable'
6. Open source policies and guidelines	*Part IV: Machine Learning and AI*
Part II: Art of R programming	17. Entity resolution and record linking in R
7. Use R efficiently with 'data.table'!	18. Statistical tests and mixed-effects analysis in R
8. Python and R, unite!	19. Machine Learning and Modeling in R
9. From Excel to R	20. Text Analysis in R
10. Reading various kinds of data in R	21. Computer vision and Deep learning in R
11. Object oriented programming in R	22. Simulation and Optimization in R

These discussions are reviewed and updated regularly, commonly as part of weekly community meetups. A dedicated GCcollab subgroup (https://gccollab.ca/groups/about/7855030) is created for sharing minutes, notes and video-recordings from these meetups. Additionally, a GCwiki page (https://wiki.gccollab.ca/UseR!) is also created to consolidate all discussion topics in one place and link them with other data science resources in the wiki space.

4.3 GitHub: open-canada - https://open-canada.github.io/UseR/

Inline with the GC Directive on Service and Digital, since most information gathered by the R4GC community is unclassified and comes from public domain, a public facing organizational account has been created on GitHub (https://github.com/open-canada) for sharing and growing the R4GC community knowledgebase. This is where public-facing community outputs are gathered, including the growing collection of Web Apps built with contributions from GC data scientists using open source tools and data (https://open-canada.github.io/Apps), as further described.

5. Outputs to date

Table 3 provides the list of tutorials and interactive applications developed by the community to date. The descriptions and screenshots of the applications are also provided in the Appendix. The updated list and more details are provided in "The R4GC Book" at https://open-canada.github.io/r4gc.

Table 3: R4GC community outputs

Tutorials:	**Interactive Shiny Apps :**
• R101: Building COVID-19 Tracker App from scratch • GCcode 101 for GC employees • R packages 101 for GC employees • Python and R unite! • Geo-spatial analysis and visualization (working with polygons, spatial intensity smoothing) • Text analysis (topics modeling, similar pages detection, string alignment and matching) • Working with Open Government Portal API within R (using 'ckanr' and 'adobeanalyticsr') • Automating R scripts with GitHub Actions	• PSES results interactive analysis and visualization • NLP analysis of ATIP requests • Automated COVID-19 results search and tracker • Geo-mapped current, historical, and predicted border wait times • Data engineering testbed demo tool for cleaning and linking noisy data records • Web page comparison tool to automate the detection of similar web pages (in gccode only) • Large data summarizer (in gccode only) • Masked name comparison tool (in gccode only)

Acknowledgements

R4GC is a collaborative effort of many people who have contributed to the development of the codes and knowledgebase that is gathered in the R4GC portals and highlighted in this paper. In particular, contributions from Jonathan Dench, Joseph Stinziano, Henry Luan, Eric Littlewood, Philippe-Israel Morin, Tony Machado, Chris Lavoie, Sylvain Paquet, Dejan Pavlic, Utku Suleymanoglu are gratefully acknowledged, as is the support from the wider international R community through stackoverflow.org portal, and conferences and webinars organized by the RStudio.

References

Bourque, P. and R.E. Fairley, eds., (2014), "Guide to the Software Engineering Body of Knowledge", IEEE Computer Society; https://www.computer.org/education/bodies-of-knowledge/software-engineering

Gorodnichy, D. (2021), "Data Engineering Challenges and Solutions: Demo of Shiny", Presentation at the 2021 GC Data Conference, Data Literacy Fest Workshop: https://youtu.be/QWv6E3e7bek.

Winters, T., T. Manshreck and H. Wright (2020), "Software Engineering at Google", O'Reilly Media: https://www.oreilly.com/library/view/software-engineering-at/9781492082781/preface01.html

Appendix: Web Apps developed by the R4GC community

A. PSES App: https://open-canada.github.io/Apps/pses

No other Open Canada data is of as much common interest across the government as the PSES results[4]. These data contain the information about all GC departments, their organizational structure and performance. A Shiny App prototype is developed to perform three most desired tasks one wished to do with these data (see Figure 1):

1. Vertical result tracking: results comparison across an organization, automated detection and visualization of the comparative performance within the organization - for any given PSES question.

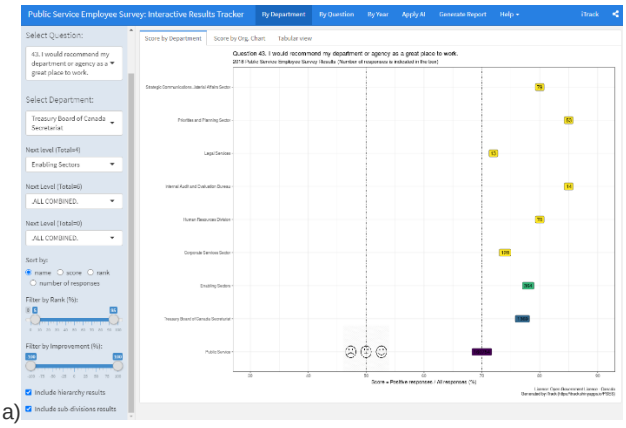
2. Horizontal results tracking: results comparison over time - for any given unit, in comparison to the organization and Public Service averages

3. Performance summary by theme: automated generation of report cards that show the performance at each level of the organization for each of theme and in relation to the rest of the organization and Public Service average.

[4] https://www.canada.ca/en/treasury-board-secretariat/services/innovation/public-service-employee-survey.html

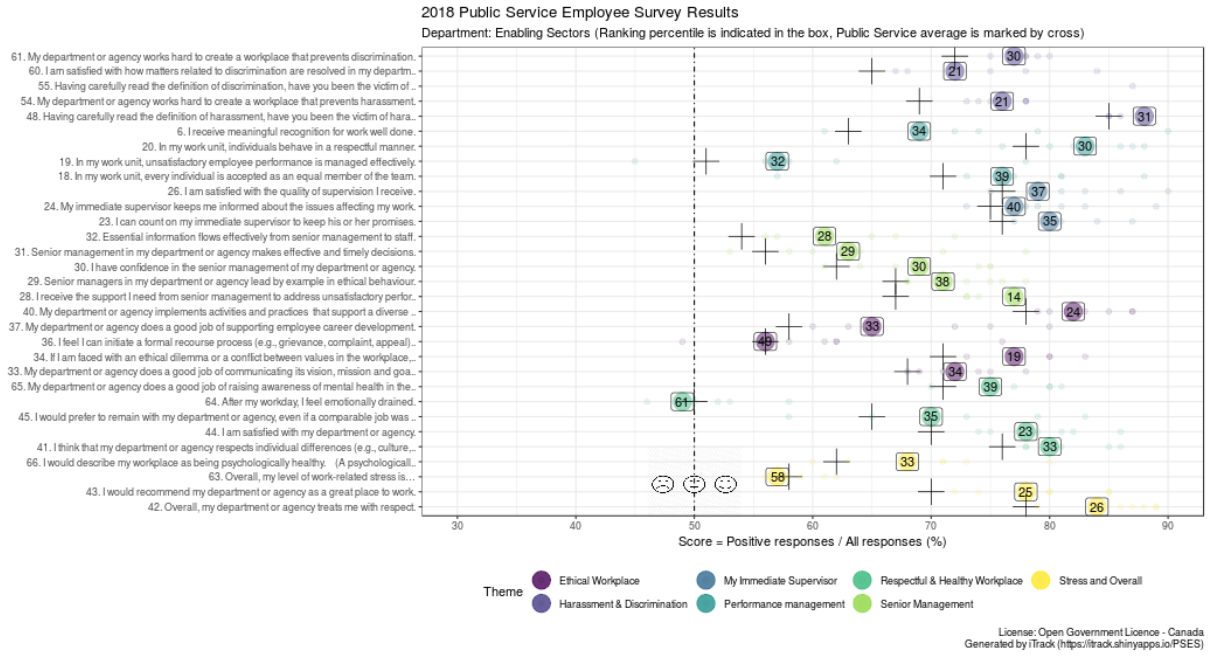
Figure 1.
Key functionalities of the PSES App prototype: a) vertical results tracking - for any question over entire organization, b) horizontal results tracking - for any unit over time, and c) performance report summary - for each unit, by theme, and in comparison to the Public Service average (shown as crosses) and other units within organization (shown as small dots). The results can be displayed filtered and sorted by score, ranking percentile, org. structure, number of responses, by theme or question.

a)



b)

c)



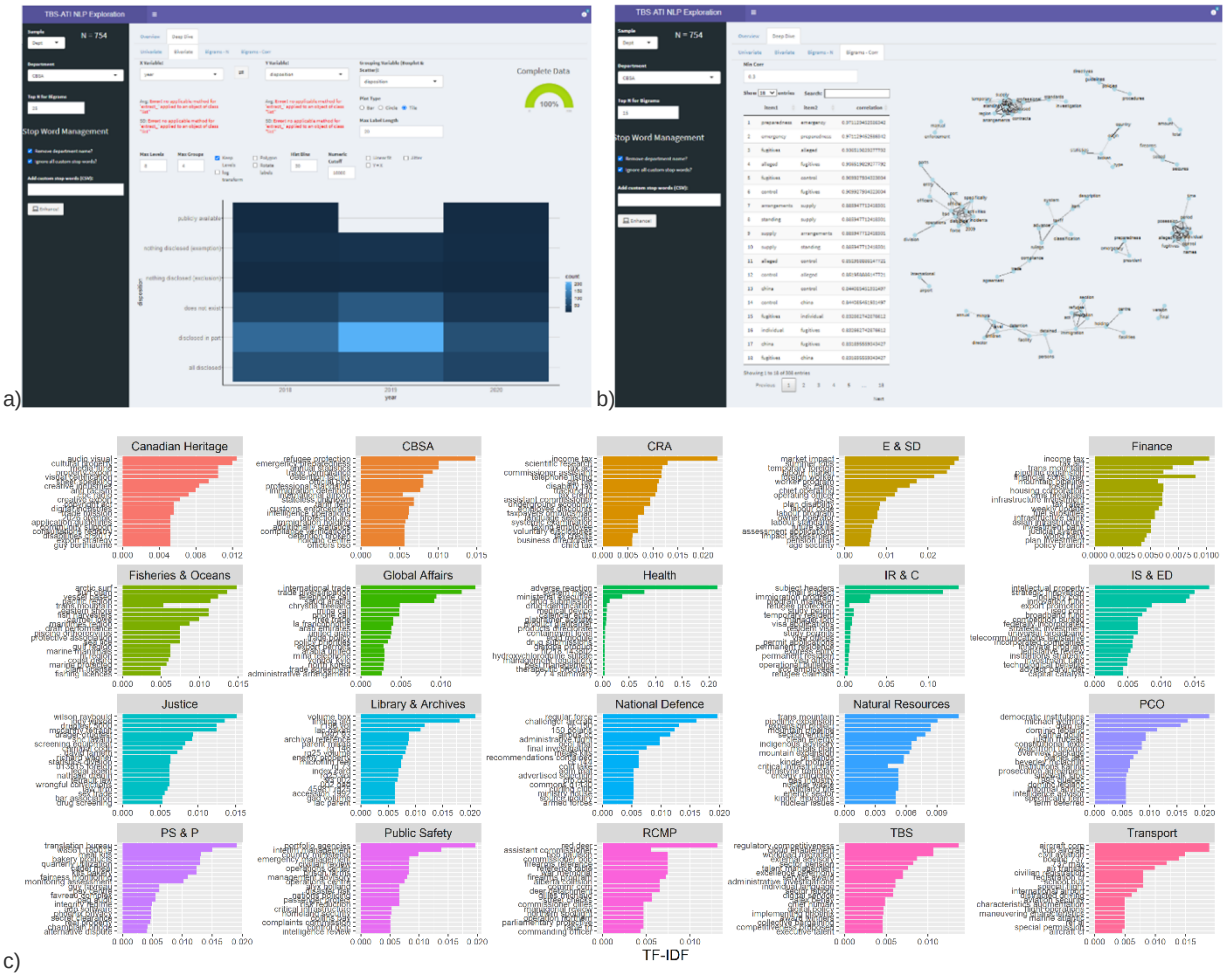
B. ATIP App: https://open-canada.github.io/Apps/atip

Another Open Canada dataset that relates to the performance of many GC departments is the ATIP requests dataset [5]. An interactive Natural Language Processing (NLP) application has been developed to enable the analysis and visualization of these requests for each participating department (see Figure 2). Its functionalities include: statistics summary, automated key-words and topic extraction using N-grams, document term matrix and Latent Dirichlet Allocation[6]. The topics can be visualized as word-clouds or as graphs that connect the related words.

[5] https://open.canada.ca/en/search/ati

[6] See "Text Mining with R!" by Julia Silge and David Robinson (https://www.tidytextmining.com) for definitions of the terms.

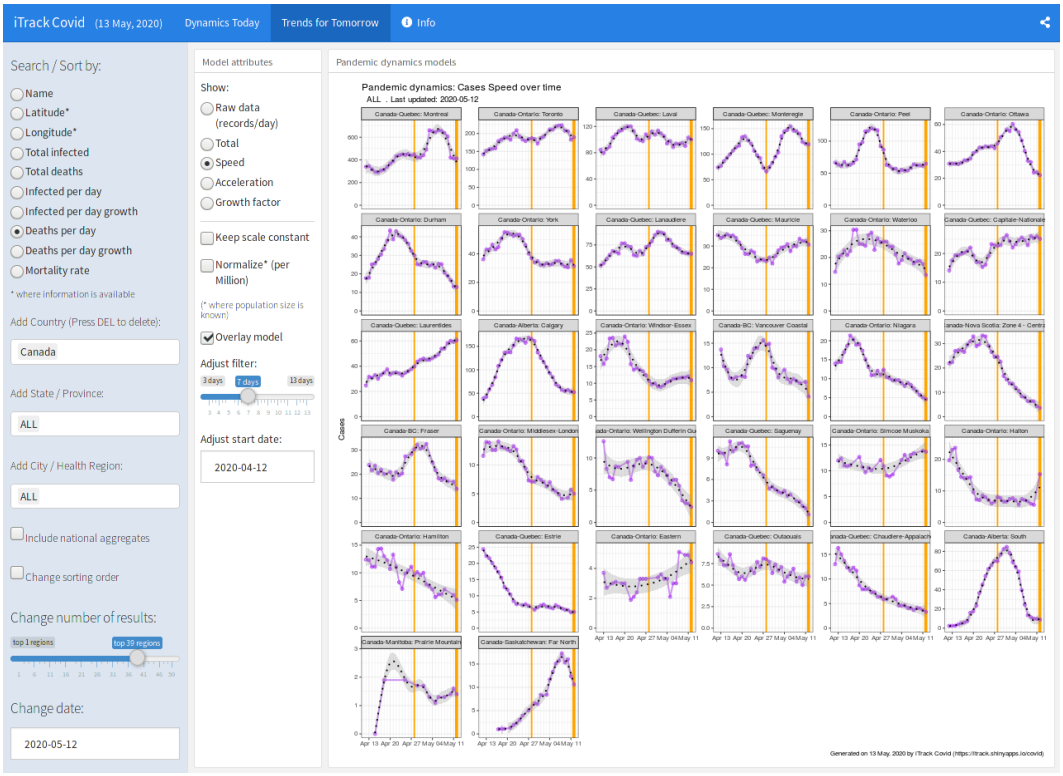
Figure 2.
Key functionalities of the ATIP App: a) department specific bi-variable statistics (such as dispositions by year, shown in the image), b) department specific key topics, visualized as correlated terms graphs , c) key topics for each participating department, visualized as N-gram frequency bars (such as 2-gram, or two-word combination, shown in the image).



C. COVID App: https://open-canada.github.io/Apps/covid

This App was built at the beginning of the pandemic. It uses open COVID-19 data for Canada and US to allow one to summarize, search and sort the results by geographical proximity using a variety of criteria (see Figure 3). More details about it and a tutorial on how to build it are provided at https://open-canada.github.io/UseR/learn2020

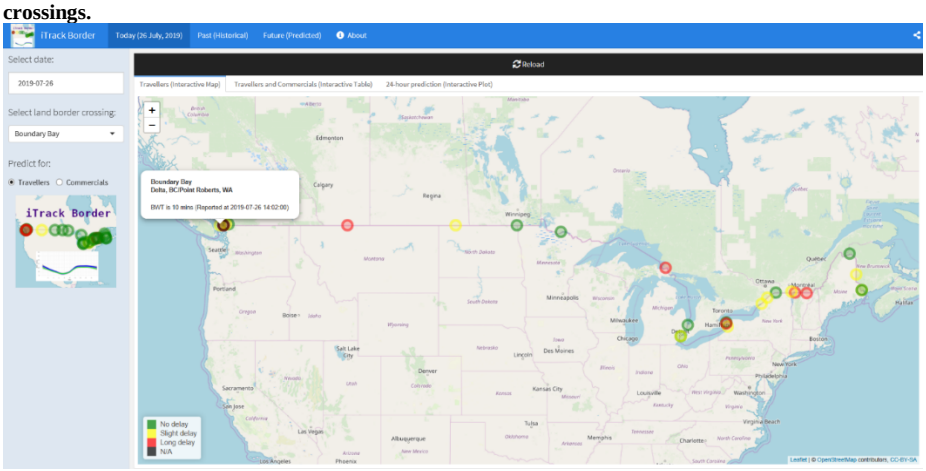
Figure 3.
COVID App allows one to summarize, search and sort COVID-19 results by geographical location using a variety of criteria.



D. Border Wait Times App: https://open-canada.github.io/Apps/border

This App (shown in Figure 4) combines Open geo-spatial data with Open historical and current border wait times data to predict and visualize delays at Canadian land border crossings. The App is included (entitled iTrack-Border) in the Open Canada Apps Gallery at https://open.canada.ca/en/apps, where more information about it can be found.

Figure 4.
Border Wait Time Interactive Tracker application allows one to visualize and predict current Border Wait Time at Canadian land border crossings.



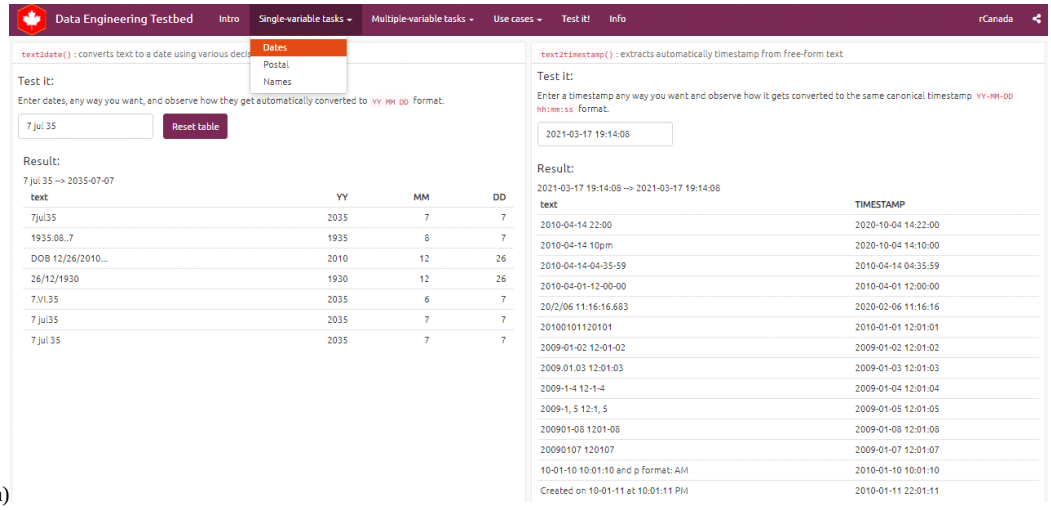
E. Data Engineering Testbed App: https://rCanada.shinyapps.io/demo

This App was demonstrated and used for training in the "Data Engineering Challenges and Solutions: Demo of Shiny" workshop at 2021 GC Data Conference, Data Literacy Fest (Gorodnichy, 2021). Its key functionalities include:

1. Single variable cleaning tasks (demo mode): automated extraction of date and timestamps from arbitrarily typed strings (as shown in Figure 5.a); rectification and correction of Canadian postal codes; fuzzy matching of names.
2. Multi-variable linking tasks (demo mode): city names deduplication; deduplication and linking of name records.
3. Use cases (live Web crawling mode): automated extraction of name variants and important dates from Web.
4. "Test it!" mode: fuzzy linking and deduplication of user-uploaded multi-variable data records, using a variety of string similarity metrics and thresholds (as shown in Figure 5.b).

Figure 5. Data Engineering Testbed App allows one to test various data cleaning (a) and data linking (b) techniques.

a)



b)