The American Statistician

ISSN: 0003-1305 (Print) 1537-2731 (Online) Journal homepage: http://amstat.tandfonline.com/loi/utas20

How R Helps Airbnb Make the Most of Its Data

Ricardo Bion, Robert Chang & Jason Goodman

To cite this article: Ricardo Bion, Robert Chang & Jason Goodman (2017): How R Helps Airbnb Make the Most of Its Data, The American Statistician, DOI: 10.1080/00031305.2017.1392362

To link to this article: https://doi.org/10.1080/00031305.2017.1392362

How R Helps Airbnb Make the Most of Its Data

Ricardo Bion
Airbnb
and
Robert Chang
Airbnb
and
Jason Goodman
Airbnb

September 20, 2017

Abstract

At Airbnb, R has been among the most popular tools for doing data science work in many different contexts, including generating product insights, interpreting experiments, and building predictive models. Airbnb supports R usage by creating internal R tools and by creating a community of R users. We provide some specific advice for practitioners who wish to incorporate R into their day-to-day workflow.

Keywords: Exploratory Data Analysis, Machine Learning, Statistical Computing, Inference

1 Introduction

Airbnb's data science team relies on R every day to make sense of its data. While many of our teammates use Python, R is the most commonly used tool for data analysis at Airbnb. In a recent survey, we found that 73% of our data scientists and analysts rated themselves as closer to "Expert" than "Beginner" in using R, and 58% regularly use R as a language for data analysis. R comes into play at all stages of the analysis pipeline, from exploratory data analysis and predictive modeling, to sharing results with business partners. In this article, we highlight the role that R plays at Airbnb and share some practical insights for others who seek to use R to help their teams make the most of their data.

2 Data Science at Airbnb

Airbnb is a community marketplace that provides access to millions of unique accommodations in more than 65,000 cities and 191 countries. In addition to accommodations, with Experiences, Airbnb offers unprecedented access to local communities and interests, while Places lets people discover recommendations by the people that live there.

The company's success is partly due to the data science team. Data science has played a crucial role at Airbnb since the company's beginning. The first member of our team was among the first 10 hires at Airbnb. Today, Airbnb employs a broad range of people to help make sense of its data, including data scientists, data engineers, business analysts, machine learning engineers, data infrastructure engineers, and data product engineers. Data scientists work on nearly every part of Airbnb's website, app, and operations.

To start, what do we mean by data science? Put simply, our team's mandate is to use data to inform decision-making. When guests and hosts use the Airbnb website or app, their actions are captured as data. By sifting through that data for trends and patterns, we are better able to listen to the voice of our customers, and make better decisions as a result.

Data scientists at Airbnb perform many types of tasks to empower Airbnb with data. We specify which data to record. We build data pipelines. We define metrics. We develop educational resources for other teams. We build internal data tools. We create reports and

dashboards. However, we can categorize the core of our work into three main focus areas: *Product Insights*, *Experimentation*, and *Predictive Modeling*.

2.1 Product Insights

The goal of product insight work is to find opportunities to help Airbnb work better for guests and hosts by improving the website and app, which constitute Airbnb's 'product.' Product insight work is usually exploratory in nature, as it's not possible to know in advance what one will find in the data. This type of work might address questions such as 'Which types of guests are staying on Airbnb in these regions?', 'Why aren't some new hosts getting booked?', or 'Which cities are supply constrained?' The insights gleaned from this work often lead directly to new product ideas and hypotheses about user behavior which can then be tested in experiments.

2.2 Experimentation

Experimentation, also commonly referred to as A/B testing, plays an important role in data-informed product development. The goal of experimentation is to validate or invalidate the team's hypotheses about the performance of new product features. We use experiments to measure how a new feature affects key business metrics such as bookings, customer service tickets, and host or guest review scores. If the feature performs well, we implement it in the app and website for all Airbnb users. Conversely, if important metrics are hurt by the new feature, we may make further changes to the feature and try it out again in a new experiment. Sometimes, new features will boost some metrics and harm others. In these cases, product teams review the experiment results together and make a judgement call about whether or not to launch the new feature. Almost all new product ideas at Airbnb are validated in this way through randomized experimentation before they are released broadly. Additionally, experimentation is a way to attribute the gains associated with new product features to the teams that worked hard to design, test, and implement them.

Data scientists are involved in all aspects of the experimentation cycle. Before experiments, we perform power analyses to determine how long the experiments should be run.

We design experiments by specifying how they should be run. We determine which metrics should be tracked in experiments to measure the success of the product feature being tested. Finally, we often perform additional "deep-dives" beyond basic t-tests to better determine the impact of the product feature in question.

2.3 Predictive Modeling

The third kind of work is what most people think of when they hear the term data science: machine learning and predictive analytics. One well-known application of machine learning at Airbnb is our Smart Pricing feature. Smart Pricing suggests nightly prices to hosts for their listings. The suggestions are generated by a machine learning algorithm that takes into account a variety of points of information, such as the date of the night to price, the listing's location and amenities, and the listing's booking history. Smart Pricing allows hosts to automatically set their prices higher or lower based on changes in demand for listings like theirs. Hosts are always responsible for setting prices, and are free to accept or reject any suggested prices.

3 R at Airbnb

R, as a tool, does not exist in a vacuum. It is usually the last tool used in a long series of tools for capturing, processing, and analyzing data.

The first step is logging. Whenever users interact with Airbnb, such as by making a search, contacting a host, or even calling customer support, client-side and server-side event data is logged, meaning it is sent back to Airbnb for collection and processing. Meanwhile, stateful information - information that can be updated over time, such as listing's availability or a user's profile description - is recorded in MySQL databases. These two main sources of data, logs and stateful information, are ingested by our data warehouse on AWS (Amazon Web Services) and securely stored on HDFS (Hadoop Distributed File System) using S3, Amazon's Simple Storage System, which is an object storage system with a simple web service interface to store and retrieve data from anywhere on the web).

Every night, a large number of ETL (Extract, Transform, and Load) jobs are kicked off

in our open-source workflow manager Airflow, calculating useful metrics and aggregations from the raw data for further analysis and predictive modeling. The tables that result from these ETL jobs are organized in the star schema style, with 'fact' tables that record the occurrence of events, such as bookings or reviews, that can be joined to 'dimension' tables that describe attributes of the events, such as a user's or listing's information. This system allows analysts to efficiently query our data and uncover insights.

Different data analysis tools are available for users of varying technical backgrounds. For visual analysis, users often use Tableau and our open-source data visualization tool Superset to understand the state of the business. Analysts who are comfortable with SQL use SQL Lab to interact with the data warehouse directly. Finally, data scientists use a wide variety of tools. Among the most important is the R language.

R is essential for all three focus areas we mentioned in section 2. For product insight work, packages such as `dplyr` (Wickham & Francois 2016) make it easy to quickly slice-and-dice data. Combined with graphing packages like `ggplot2` (Wickham 2009), `dplyr` allows data scientists to scan through datasets in search of insights. For experimentation work, R's `pwr` (Champely 2017) package makes it easy to perform statistical tests, helping us hold ourselves to high standards of rigor in our experimentation. For machine learning, data scientists build proof-of-concept predictive models as prototypes to prove that more sophisticated and intelligent product features are good investments before building them for production systems. In the next section, we go into more detail and provide some examples.

3.1 Product Insights

R enables product insight work at Airbnb in three main ways: exploratory data analysis combining our internal packages with `tidyr` (Wickham 2017) and `dplyr`, data visualization with internal branded themes and packages such as `ggplot2`, and reproducible research with our internal Knowledge Repository and packages such as `rmarkdown` (Allaire et al. 2017).

```r
# install and load Rbnb package
devtools::install_github("airbnb/Rbnb")

library(Rbnb)

# move data from Presto into R and run model
presto_get("select * from bookings") %>%
  lm(n_bookings ~ market + booking_channel, data=.)

# move data from R into S3
bookings %>%
  impute_data(impute_col="n_bookings", ds_col="date") %>%
  yoy("n_bookings", "date") %>%
  s3_put("bookings_yoy.csv")
```

Figure 1: Example of simple workflow in R: Querying data from Presto, performing analysis, and sending results back to S3

3.1.1 Exploratory Data Analysis

Our data analysis workflow normally starts with data extraction. At Airbnb, we use Hive and Presto, two SQL-like languages, to bring data from HDFS to our local machines. We have written R code that allows users to access this infrastructure directly. Users can simply type in a SQL query and get the data back in an R data.frame. Once data is in memory, we use `magrittr` (Bache & Wickham 2014) pipes to chain operations together to perform data munging and analysis. The entire process of querying the database, manipulating data, graphing and predictive modeling building can happen in a few lines, as demonstrated in Figure 1, allowing us to rapidly iterate through analyses and visualizations. We use a consistent style guide and set of packages for data exploration, relying mainly on `tidyr`, `dplyr`, `broom` (Robinson 2017), and `purrr` (Henry & Wickham 2017).

3.1.2 Data Visualization

We use ggplot2 as our main package to create ad-hoc exploratory graphics as well as polished-looking customized visualizations. When combined with tools to clean and transform data, ggplot2 allows analysts to quickly translate insights into high quality, compelling visualizations. In addition to the static graphics of `ggplot2`, we often make interactive visualizations or dashboards using R packages such as `plotly` (Sievert et al. 2017), `leaflet` (Cheng et al. 2017), `dygraphs` (Vanderkam et al. 2017), `DiagrammeR` (Sveidqvist et al. 2017), and `shiny` (Chang et al. 2017).

3.1.3 Reproducible Research

At Airbnb, all R analyses are documented in `rmarkdown`, where code and visualizations are combined in a single written report. Posts are carefully reviewed by experts in the content area and techniques used, both in terms of methodology and code style, before being published and shared with business partners. The peer review process is often repeated until both the author and the reviewers are satisfied with the analysis. This careful code review process helps bring credibility to the team's reports, reduce the number of bugs in analysis code, and often leads to more approachable insights and concrete recommendations.

Once a report is ready, the code is merged into a Git repository and the final report is shared in an internal website called the Knowledge Repository (Sharma & Overgoor 2016). The Knowledge Repository, pictured in Figure 2, is a Github-enabled internal web application powered by Python and Flask (a micro web framework for building websites) that has search, subscription, and tagging capabilities. Some of these reports eventually are featured in academic journals or public-facing blog posts - see "How Airbnb uses Machine Learning to Detect Host Preferences" (Ifrach 2015) and "How well does NPS predict rebooking?" (Qian 2015) as examples. The Knowledge Repository attempts to incorporate best practices from software engineering (e.g., code review, reproducibility, version control), with best practices from academic research (e.g., peer review, clear hypothesis, review of the literature).

Housing analysis in this format offers many advantages. It makes it easy for anyone in the company, including non-technical business partners, to find analyses that might be relevant to their team. It allows data scientists to easily learn both context and new analytical techniques from previous work. It enables analysts to reuse code from past reports. It simplifies coordination across teams by keeping everyone in the loop on the latest research. None of this would be possible without the common usage of `rmarkdown` across the team.

3.2 Experimentation

In our team's early days, we used R to perform experiment analyses manually. We would combine experiment assignment logs with data on bookings and other important metrics

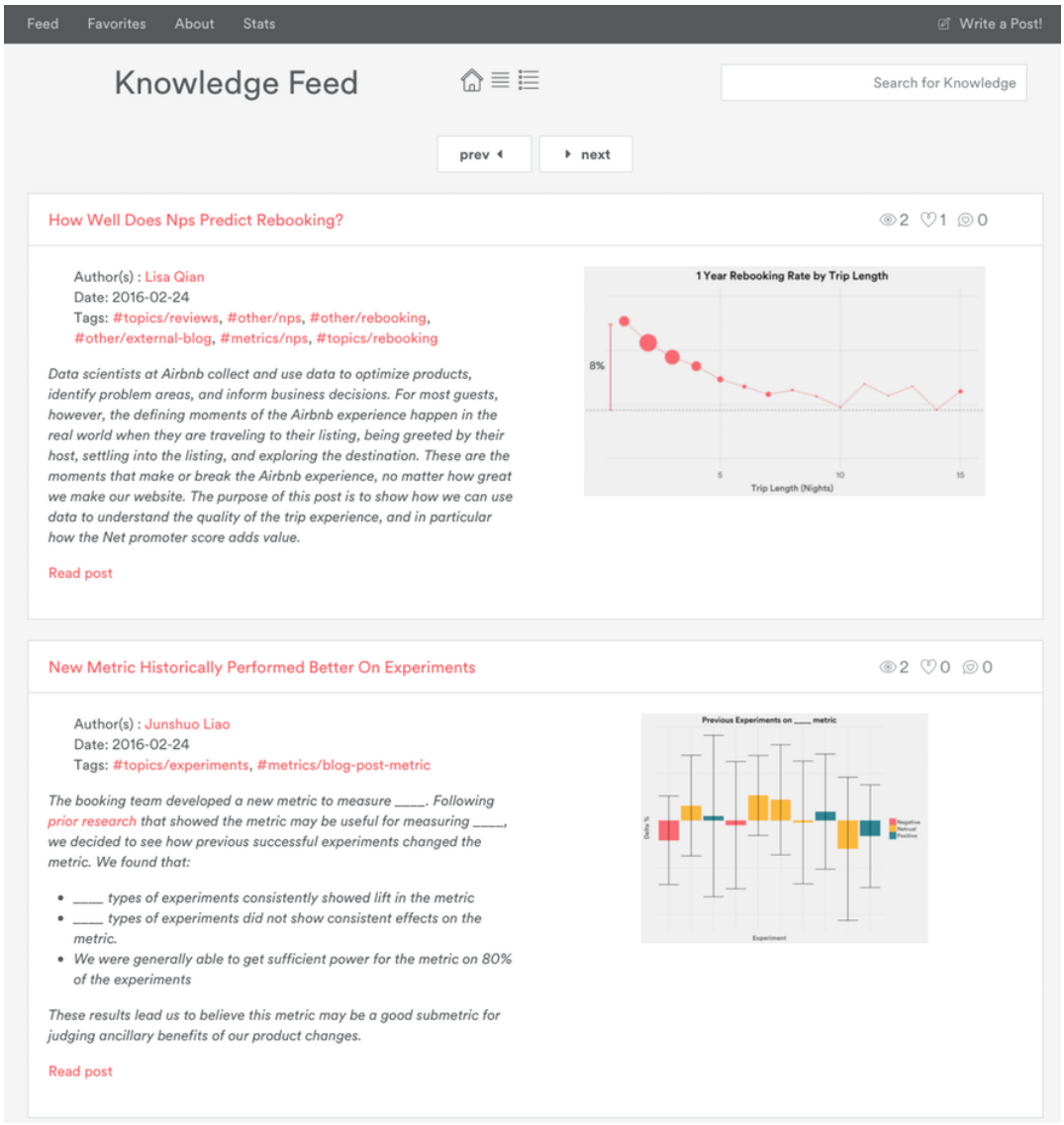

Figure 2: Example of product insights written as RMarkdown documents, rendered in our Knowledge Repository. Taken with permission from Scaling Knowledge at Airbnb. The posts include branded ggplot2 visualizations created in ggplot2.

before performing statistical tests to evaluate our experiments. Eventually, we built a Shiny app that did these analyses for us at scale. It reported the results of dozens of experiments to data scientists, product managers and engineers. Over time, we began running hundreds of experiments simultaneously and Shiny could no longer scale to meet our needs. We needed a fully customized solution that could be supported by our core engineering team, and easily integrated into our data infrastructure, which runs mainly in Ruby and Python. The solution was to build a production pipeline and website to analyze experiments called the Experiment Reporting Framework (Moss 2014) to display experiment statuses. This is a great example of how R can be used to prototype new data tools. It helped to demonstrate the potential value of the tool and enabled us to better understand the needs of those who might use it before we built a more scalable solution.

While today, much of our experimentation work is automated, R still plays a crucial role in more complex experiment analyses and deep dives. These analyses might, for instance, involve handling particularly complicated experimental setups that are not supported by our automated tools, target metrics that must be computed separately, or statistical analyses more sophisticated than t-tests (for more reference on experimentation at Airbnb, see Overgoor (2014)). Additionally, we use R to verify that experimental assignment is occurring properly. As with the move from our Shiny app to the production experiment tool, we try to automate our work whenever possible. If we find ourselves repeating the same ad-hoc experimental analyses in R, we try to build that functionality into our tooling. For example, we used to compute by hand metrics that depended on the time since a user was assigned to an experiment. After repeating this sort of analysis manually a few times, we wrote an R function to perform the computation. Finally, this capability was incorporated into our main experiment tool.

3.3 Predictive Modeling

While R is an extremely effective tool for data cleaning and data munging, it is an even more powerful language for data modeling. Its power lies in the wide range of optimization, machine learning, and econometric techniques that the R community has built over the years. At Airbnb, many of our data scientists build models using these packages. Next, we

describe four of our teammates' projects that used R's data predictive modeling capabilities.

3.3.1 Prototyping Lifetime Value with R

It is often costly and time-consuming to build machine learning systems in production development environments. Therefore, it can be very valuable to validate a machine learning approach with a prototype before fully investing in the engineering work to bring a model to scale. R is an extremely powerful tool for this purpose. A data scientist can quickly turn raw data into usable training data with R. Subsequently, the data scientist can use R to try out a wide variety of models in order to understand how these models perform compared to a naive, non-modeling solution.

For example, when our data scientists tried to predict future revenue at the listing and guest level, we built a wide variety of prototypes in R to validate that a model-based approach was worth pursuing by comparing the RMSE of the challenger models with that of a baseline model. After the prototyping step, we worked with engineers to bring the prototype into production.

3.3.2 Airbnb Guest Service Fee Optimization using NLopt package

When a reservation is confirmed, Airbnb charges guests a service fee between 5% and 15% of the reservation subtotal. These fees are calculated using a variety of factors, such as the reservation's length and the booked listing's characteristics. One of the data science team's projects was to optimize these fees, while ensuring that our pricing policies were fair to our guests and hosts. For example, Airbnb tends to charge lower guest service fee percentages to higher reservation subtotals. However, we do not have surge service fees for high-demand dates such as New Years or holidays. In the past, we experimented with different service fee structures, which allowed us to learn the demand elasticity of guest bookings to service fees. Given this elasticity, we could then find the fee curve that maximized revenue on the platform. This exercise was entirely modeled as a nonlinear optimization problem using `NLopt` (Johnson 2017). The result of the model provided guidance on how we should set the guest fees. We ran a follow-up experiment to validate this new payout structure.

3.3.3 Building A Marginal Returns Model in R

Where should we invest to attract new hosts to Airbnb? How much should we spend on these efforts? To better answer these questions, our data scientists built a Cobb-Douglas-like econometric matching model to estimate bookings by market as a function of supply and demand in that market.

$$B(S, D) = AD^{\alpha}S^{\gamma}$$

Here, $\boldsymbol{B}$ represents nights booked in a particular market and $\boldsymbol{D}$ and $\boldsymbol{S}$ represent demand and supply respectively. $\boldsymbol{A}$ stands for matching efficiency (i.e. how well the marketplace is matching supply and demand). The exponents α and γ can be interpreted as the booking elasticity of supply and demand - for a one percent increase in demand/supply, α and γ represent the percent increase in bookings.

Using historical supply (nights available to book) and demand (number of searchers), we were able to estimate the elasticities in R by taking the log of the equation on both side. With the fitted model and future forecasts for supply and demand, we were able to predict the marginal returns in bookings of adding an additional listing to the market (by taking the derivative of **B** with respect to **S**). Finally, the marginal returns of supply helped us determine which markets have the highest marginal returns. We could then prioritize our supply and demand acquisition efforts accordingly.

3.3.4 Causal Tree

After experiments are concluded, data scientists are often asked which subsets of users were most affected by the new feature being tested. To answer, data scientists usually manually investigate the experiment's impact on different sets of users. However, Susan Athey's `CausalTree` (Athey & Imbens 2015) algorithm allows data scientists to cleverly automate some of this work by applying machine learning techniques to causal inference problems. The algorithm recursively partitions an experiment's data on features that show the largest difference between the experiment's treatment and control groups. The algorithm was first implemented and made publicly available in R. Since we used R natively at Airbnb, we were able to use this algorithm as soon as it was released. This is an example of a common

pattern. Since R is used by so many leading statisicians and academics, the latest techniques are often made available very quickly in R, which is useful for data scientists in industry who wish to take advantage of new methods as they are developed.

4 How We Support R Usage

4.1 Community

We help our team members develop as R programmers in a variety of ways. The most structured way is by teaching classes on R. Every new data scientist at Airbnb starts with a week of classes in a Data Bootcamp. The R Bootcamp classes are taught by experienced Airbnb data scientists who use R as their tool of choice. In the class, they help new team members set up their development environments and walk them through a series of hands-on tutorials demonstrating commonly-used R package with real Airbnb data.

Once on the job, team members mostly improve their skills by learning from one another. As detailed in section 3.1.3, analyses are peer-reviewed by fellow data scientists before they are shared with external business partners. The process allows team members to give feedback not only on content and analysis but on the R coding itself. One of the best ways for data scientists to improve is to read others' code in the Knowledge Repository to see how they solved problems that resemble their own. Another way we learn from one another is through our R Slack channel (an enterprise instant message communication platform), where team members can ask one another for help or advice on using R.

Internally, we actively run learning lunches, learning groups, and tutorials to showcase new packages and development in R. We allocate education funds and actively encourage our data scientists to learn from the best in the industry by attending R training courses such as DataCamp (a popular website for learning data science in the browser) and conferences such as UseR!

We recently developed three R classes for our internal Data University, which is open to all Airbnb employees. These classes cover introduction to analyses in R, data visualization, and how to use and contribute to our internal R packages. They are taught in person by expert developers, and include custom-made online resources.

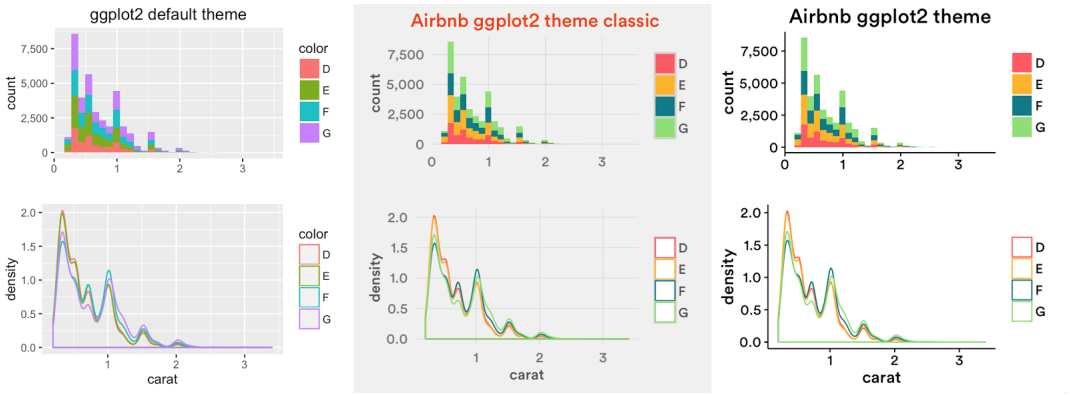

Figure 3: Example of Airbnb branded ggplot2 themes and scales to standardize our data visualizations. These extensions create a consistent internal data science brand, and can be found on Github. Taken with permission from 'Using R packages and education to scale data science at Airbnb.'

We also engage with the broader R community outside Airbnb. We participate in and sponsor rOpenSci, contribute to open source projects such as `ggtech` (Bion 2016*a*) and `RPresto` (Filiz & Goder 2017), and give talks at conferences such as the Shiny Developer Conference and UseR! We have been fortunate to have influential R developers visit our headquarters in San Francisco.

4.2 Tooling

In order to aid R analyses across the team, we developed an internal R package, `Rbnb` (Bion 2016*b*), full of useful functions and templates for data analysis specifically at Airbnb. The package is hosted on our internal Github instance, so that anyone on the team can see its contents and contribute. We constantly push out updates using the `devtools` (Wickham & Chang 2017) package.

Rbnb's primary purposes are twofold. First, it allows easy access to our infrastructure from R as mentioned in section 3.1.1. Without Rbnb, this process would be extremely time-consuming and inefficient. Second, it provides an Airbnb-styled theme to ggplot2, ensuring that our plots have a consistent look and feel across the team.

While these two aspects are by far the most commonly used features of the package, it contains over 90 functions contributed by more than 20 different members of the team.

These functions perform a wide variety of operations, from statistical procedures like imputing data to producing customizable graphics. The concept is that anytime anyone on the team solves a problem that they think others might encounter, they can generalize their code, include it into Rbnb and now the whole team can access it.

We recently launched a series of classes to teach any data scientist how to develop R packages. More than half of the team that uses R has participated, and Rbnb contributions are on the rise. We are also establishing a mailing list to distribute news of contributions to the package to the team and give credit to those who create code that the whole team can use.

For its first few years, Rbnb had relatively few contributors. The team was small, and thus the returns to writing generalizable code were low. Now that the team has grown much larger, scaling our work is more valuable. However, in reality, no team or person is solely responsible for the package's maintenance, as it is a collective effort. Therefore, building the right incentives and rewards for others to continue contributing is a crucial and challenging task.

Recently, we have begun exploring new ways to encourage contributions, for instance, emphasizing the role of contributing beyond one's product team in performance reviews and developing special stickers that team members can display on their laptops for taking classes on how to contribute to Rbnb, and for actually contributing code. Ideally, team members can strike a middle-ground, where they spend a small amount of time learning how to contribute to the package, and then integrate contributing code the whole team can use as part of their normal workflow if they happen to solve a problem that can be easily generalized.

For a more extensive discussion on tool building and education at Airbnb, see our blog post on this topic: "Using R packages and Education to Scale Data Science at Airbnb" (Bion 2016*b*).

5 Advice For Practitioners

We have come a long way as a team, and we have learned a lot along the way. Here are a few recommendations to practitioners who are interested in using R to help their team

make the most of its data. First, we recommend building helper functions to connect R to data infrastructure, such as production databases, as early as possible. In the early days, R is useful for prototyping new data tools when building production-level data tools is either too costly or requires evidence of usage to justify. Once a company has a sufficiently large team, it should create an internal R package to share code between analysts and encourage a culture of contribution and tool-building that can make the whole team more productive. In that internal package, it's useful to develop a standardized set of visualization themes to reinforce the data team's brand. We suggest using Github, `rmarkdown`, and our newly open-sourced Knowledge Repository for peer-reviewing analysis and for making research reproducible. Consider investing in ongoing education and training for team members to help them deepen their R skills. Finally, remember that R is not the only language for data work, and that it is greatly beneficial to foster a diverse team.

References

Allaire, J., Cheng, J., Xie, Y., McPherson, J., Chang, W., Allen, J., Wickham, H., Atkins, A., Hyndman, R. & Arslan, R. (2017), *rmarkdown: Dynamic Documents for R*. R package version 1.6.
URL: *https://CRAN.R-project.org/package=rmarkdown*

Athey, S. & Imbens, G. (2015), 'Recursive partitioning for heterogeneous causal effects'.

Bache, S. M. & Wickham, H. (2014), *magrittr: A Forward-Pipe Operator for R*. R package version 1.5.
URL: *https://CRAN.R-project.org/package=magrittr*

Bion, R. (2016*a*), *ggtech: ggplot2 tech themes and scales*. R package version 0.1.

Bion, R. (2016*b*), *Using R packages and education to scale Data Science at Airbnb*, Airbnb.
URL: *Using R packages and education to scale Data Science at Airbnb*

Champely, S. (2017), *pwr: Basic Functions for Power Analysis*. R package version 1.2-1.
URL: *https://CRAN.R-project.org/package=pwr*

Chang, W., Cheng, J., Allaire, J., Xie, Y. & McPherson, J. (2017), *shiny: Web Application Framework for R*. R package version 1.0.3.
URL: *https://CRAN.R-project.org/package=shiny*

Cheng, J., Karambelkar, B. & Xie, Y. (2017), *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*. R package version 1.1.0.
URL: *https://CRAN.R-project.org/package=leaflet*

Filiz, O. I. & Goder, S. (2017), *RPresto: DBI Connector to Presto*. R package version 1.2.1.9000.
URL: *https://github.com/prestodb/RPresto*

Henry, L. & Wickham, H. (2017), *purrr: Functional Programming Tools*. R package version 0.2.2.2.
URL: *https://CRAN.R-project.org/package=purrr*

Ifrach, B. (2015), *How Airbnb uses Machine Learning to Detect Host Preferences*, Airbnb.
URL: *https://medium.com/airbnb-engineering/how-airbnb-uses-machine-learning-to-detect-host-preferences-18ce07150fa3*

Johnson, S. G. (2017), *The NLopt nonlinear-optimization package*.
URL: *http://ab-initio.mit.edu/nlopt*

Moss, W. (2014), *Experiment Reporting Framework*, Airbnb.
URL: *https://medium.com/airbnb-engineering/experiment-reporting-framework-4e3fcd29e6c0*

Overgoor, J. (2014), *Experiments at Airbnb*, Airbnb.
URL: *https://medium.com/airbnb-engineering/experiments-at-airbnb-e2db3abf39e7*

Qian, L. (2015), *How well does NPS predict rebooking?*, Airbnb.
URL: *https://medium.com/airbnb-engineering/how-well-does-nps-predict-rebooking-9c84641a79a7*

Robinson, D. (2017), *broom: Convert Statistical Analysis Objects into Tidy Data Frames*. R package version 0.4.2.
URL: *https://CRAN.R-project.org/package=broom*

Sharma, C. & Overgoor, J. (2016), *Scaling Knowledge at Airbnb*, Airbnb.
URL: *https://medium.com/airbnb-engineering/scaling-knowledge-at-airbnb-875d73eff091*

Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M. & Despouy, P. (2017), *plotly: Create Interactive Web Graphics via 'plotly.js'*. R package version 4.7.0.
URL: *https://CRAN.R-project.org/package=plotly*

Sveidqvist, K., Bostock, M., Pettitt, C., Daines, M., Kashcha, A. & Iannone, R. (2017), *DiagrammeR: Create Graph Diagrams and Flowcharts Using R*. R package version 0.9.0.
URL: *https://CRAN.R-project.org/package=DiagrammeR*

Vanderkam, D., Allaire, J., Owen, J., Gromer, D., Shevtsov, P. & Thieurmel, B. (2017), *dygraphs: Interface to 'Dygraphs' Interactive Time Series Charting Library*. R package version 1.1.1.4.
URL: *https://CRAN.R-project.org/package=dygraphs*

Wickham, H. (2009), *ggplot2: Elegant Graphics for Data Analysis*, Springer-Verlag New York.
URL: *http://ggplot2.org*

Wickham, H. (2017), *tidyr: Easily Tidy Data with 'spread()' and 'gather()' Functions*. R package version 0.6.3.
URL: *https://CRAN.R-project.org/package=tidyr*

Wickham, H. & Chang, W. (2017), *devtools: Tools to Make Developing R Packages Easier*. R package version 1.13.2.
URL: *https://CRAN.R-project.org/package=devtools*

Wickham, H. & Francois, R. (2016), *dplyr: A Grammar of Data Manipulation*. R package

version 0.5.0.

URL: *https://CRAN.R-project.org/package=dplyr*