The Popularity of Data Science Software

by Robert A. Muenchen

Abstract

This article presents various ways of measuring the popularity or market share of software for advanced analytics software. Such software is also referred to as tools for data science, statistical analysis, machine learning, artificial intelligence, predictive analytics, business analytics, and is also a subset of business intelligence.

Updates: The most recent update was the Job Advertisements section 2/28/2017. I announce the updates to this article on Twitter: http://twitter.com/BobMuenchen

Introduction

When choosing a tool for data analysis, now more commonly referred to as analytics or data science, there are many factors to consider:

There are many ways to measure popularity or market share and each has its advantages and disadvantages. In rough order of the quality of the data, these include:

- Job Advertisements
- Scholarly Articles
- IT Research Firm Reports
- Surveys of Use
- Books
- Blogs
- Discussion Forum Activity
- Programming Popularity Measures
- Sales & Downloads
- Competition Use
- Growth in Capability

Let’s examine each of them in turn.

Job Advertisements

One of the best ways to measure the popularity or market share of software for data science is to count the number of job advertisements for each. Job advertisements are rich in information and are backed by money so they are perhaps the best measure of how popular each software is now. Plots of job trends give us a good idea of what is likely to become more popular in the future.

Indeed.com is the biggest job site in the U.S., making its collection the best around. As their co-founder and former CEO Paul Forster stated, Indeed.com includes “all the jobs from over 1,000 unique sources, comprising the major job boards – Monster, Careerbuilder, Hotjobs, Craigslist – as well as hundreds of newspapers, associations, and company websites." Indeed.com also has superb search capabilities and it includes a tool for tracking long-term trends.

Searching for jobs using Indeed.com is easy, but searching for software in a way that ensures fair comparisons across packages is tricky. Some software is used only for data science (e.g. SPSS, Apache Spark) while others are used in data science jobs and more broadly in report-writing jobs (e.g. SAS, Tableau). General-purpose languages (e.g. C, Java) are heavily used in data science jobs, but the vast majority of jobs that use them have nothing to do with data science. To level the playing field I developed a protocol to focus the search for each software within only jobs for data scientists. The details of this protocol are described in a separate article, *How to Search for Data Science Jobs*. All of the graphs in this section use those procedures to make the required queries.

I collected the job counts discussed in this section on February 24, 2017. One might think that a sample of on a single day might not be very stable, but the large number of job sources makes the counts in Indeed.com’s collection of jobs quite consistent. The last time I collected this data was February 20, 2014, and those that were collected using the same protocol (the general purpose languages) yielded quite similar results. They grew between 7% and 11%, and correlated r=.94, p=.002.

Figure 1a shows that SQL is in the lead with nearly 18,000 jobs, followed by Python and Java in the 13,000’s. Hadoop comes next with just over 10,000 jobs, then R, the C variants, and SAS. (The C, C++, and C# are combined in a single search since job advertisements usually seek any of them). This is the first time this report has shown more jobs for R than SAS, but keep in mind these are jobs specific to data science. If you open up the search to include jobs for report writing, you’ll find twice as many SAS jobs.

Next comes Apache Spark, which was too new to be included in the 2014 report. It has come a long way in an incredibly short time. For a detailed analysis of Spark’s status, see Spark is the Future of Analytics, by Thomas Dinsmore.

Tableau follows, with around 5,000 jobs. The 2014 report excluded Tableau due to its jobs being dominated by report writing. Including report writing will quadruple the number of jobs for Tableau expertise to just over 2o,ooo.

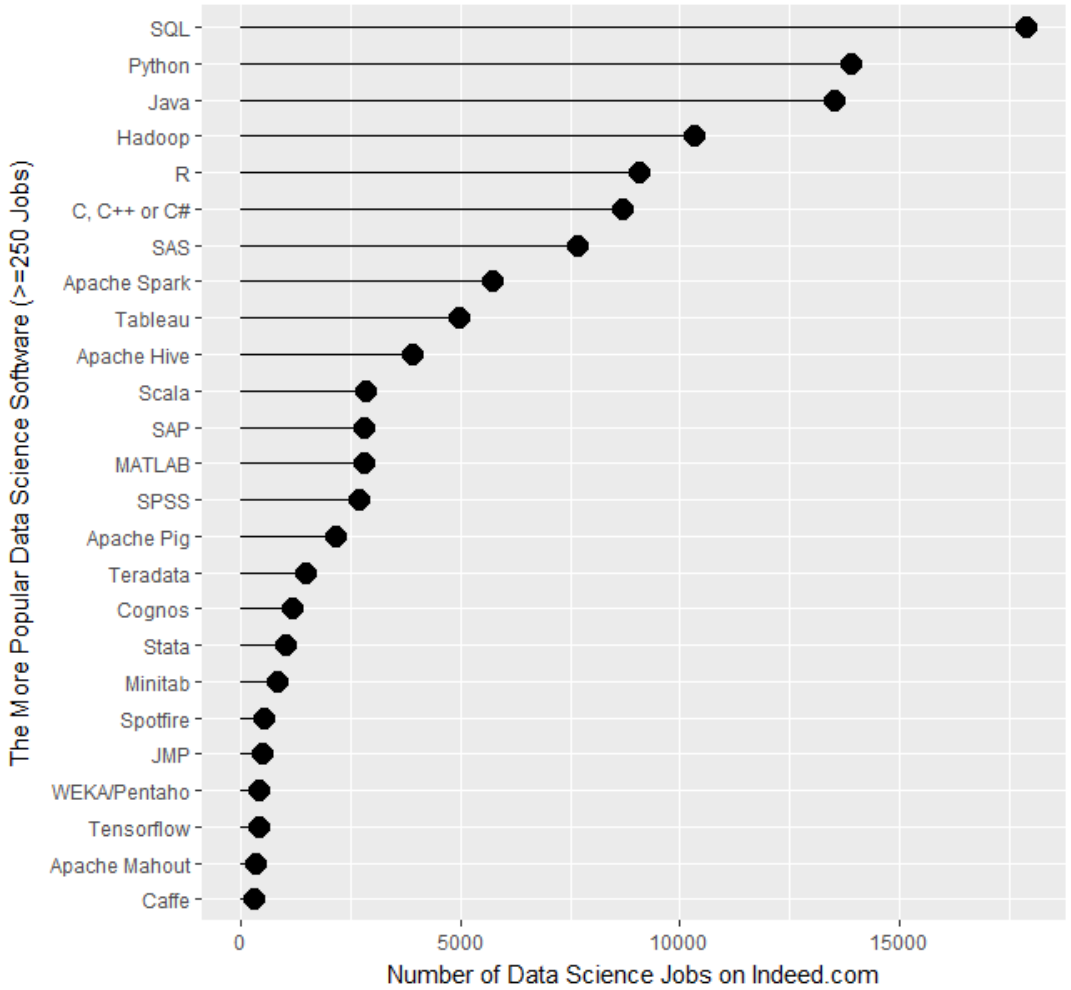

Figure 1a. The number of data science jobs for the more popular software (those with 250 jobs or more, 2/2017).

Apache Hive is next, with around 3,900 jobs, then a very diverse set of software comes next, with Scala, SAP, MATLAB, and SPSS, each having just over 2,500 data science jobs. After those, we see a slow decline from Teradata on down.

Much of the software had fewer than 250 job listings. When displayed on the same graph as the industry leaders, their job counts appear to be zero; therefore I have plotted them separately in Figure 1b. Alteryx comes out the leader of this group with 240 jobs. Microsoft was a difficult search since it appears in data science ads that mention other Microsoft products such as Windows or SQL Server. To eliminate such over-counting, I treated Microsoft different from the rest by including product names such as Azure Machine Learning and Microsoft Cognitive Toolkit. So there's a good chance I went from over-emphasizing Microsoft to under-emphasizing it with only 157 jobs.

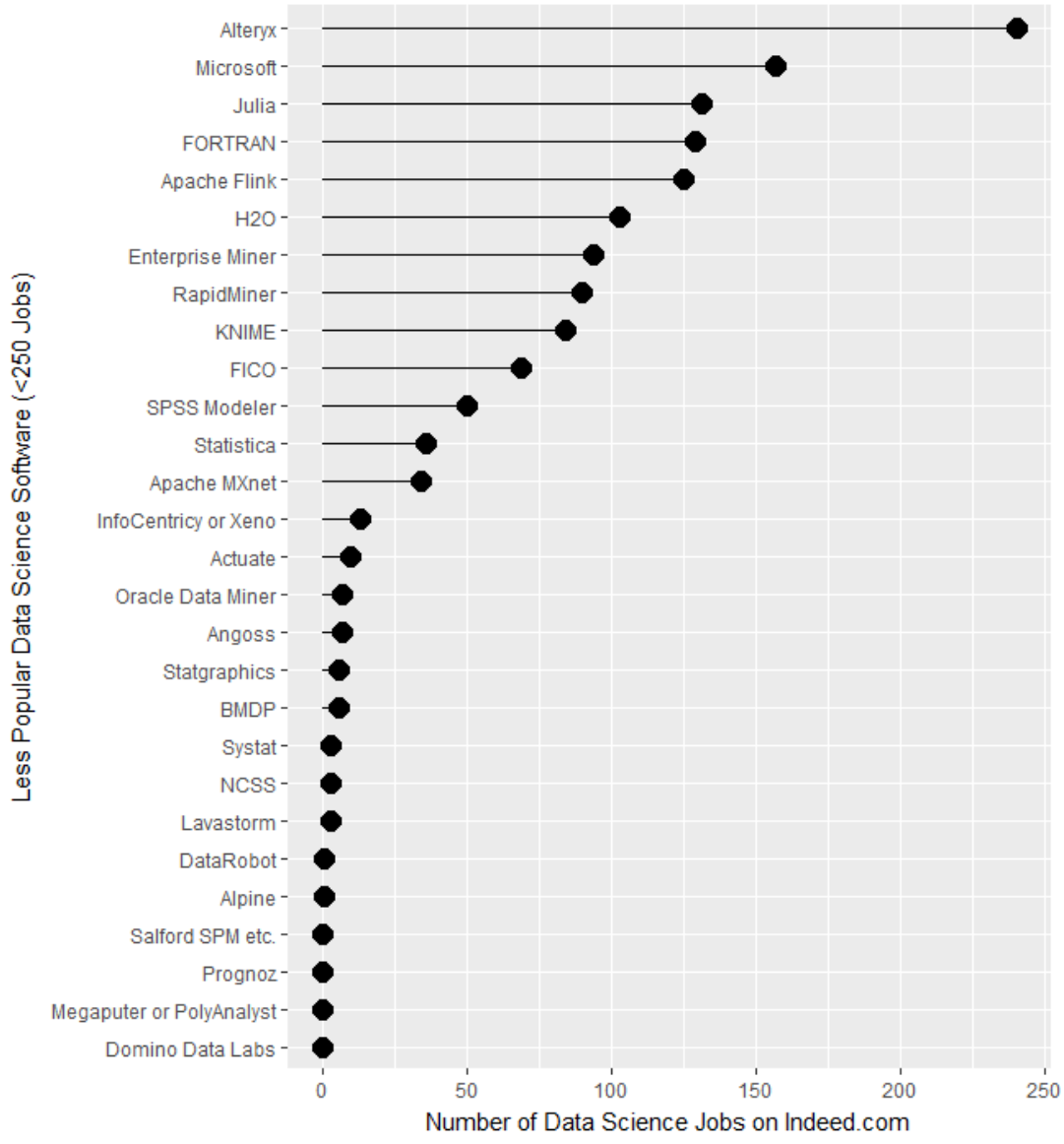

Figure 1b. The number of analytics jobs for the less popular software (under 250 jobs, 2/2017).

Next comes the fascinating new high-performance language Julia. I added FORTRAN just for fun and was surprised to see it still hanging in there after all these years. Apache Flink is also in this grouping, which all have around 125 jobs.

H2O follows, with just over 100 jobs.

I find it fascinating that SAS Enterprise Miner, RapidMiner, and KNIME appear with a similar number of jobs (around 90). Those three share a similar workflow user interface that make them particularly easy to use. The companies advertise the software as not needing much training, so it may be possible that companies feel little need to hire expertise if their existing staff picks it up more easily. SPSS Modeler also uses that type of interface, but its job count is about half that of the others, at 50 jobs.

Bringing up the rear is Statistica, which was sold to Dell, then sold to Quest. Its 36 jobs trails far behind its similar competitor, SPSS, which has a staggering 74-fold job advantage.

The open source MXNet deep learning framework, shows up next with 34 jobs. Tensorflow is a similar project with a 12-fold job advantage, but these two are both young enough that I expect both will be growing rapidly in the future.

In the final batch that has few, if any, jobs, we see a few newcomers such as DataRobot and Domino Data Labs. Others have been around for years, leaving us to wonder how they manage to stay afloat given all the competition.

It's important to note that the values shown in Figures 1a and 1b are single points in time. The number of jobs for the more popular software do not change much from day to day. Therefore the relative rankings of the software shown in Figure 1a is unlikely to change much over the coming year. The less popular packages shown in Figure 1b have such low job counts that their ranking is more likely to shift from month to month, though their position relative to the major packages should remain more stable.

Each software has an overall trend that shows how the demand for jobs changes across the years. You can plot these trends using Indeed.com's Job Trends tool. However, as before, focusing just on analytics jobs requires carefully constructed queries, and when comparing two trends at a time, they *both* have to fit in the same query limit. Those details are described here.

I'm particularly interested in trends involving R so let's see how it compares to SAS. In Figure 1c we see that the number of data science jobs for SAS has remained relatively flat from 2012 until February 28, 2017 when I made this plot. During that same period, jobs for R grew steadily and finally surpassed jobs for SAS in early 2016. As noted in a blog post (and elsewhere in this report), use of R in scholarly publications surpassed those for SAS in 2015.

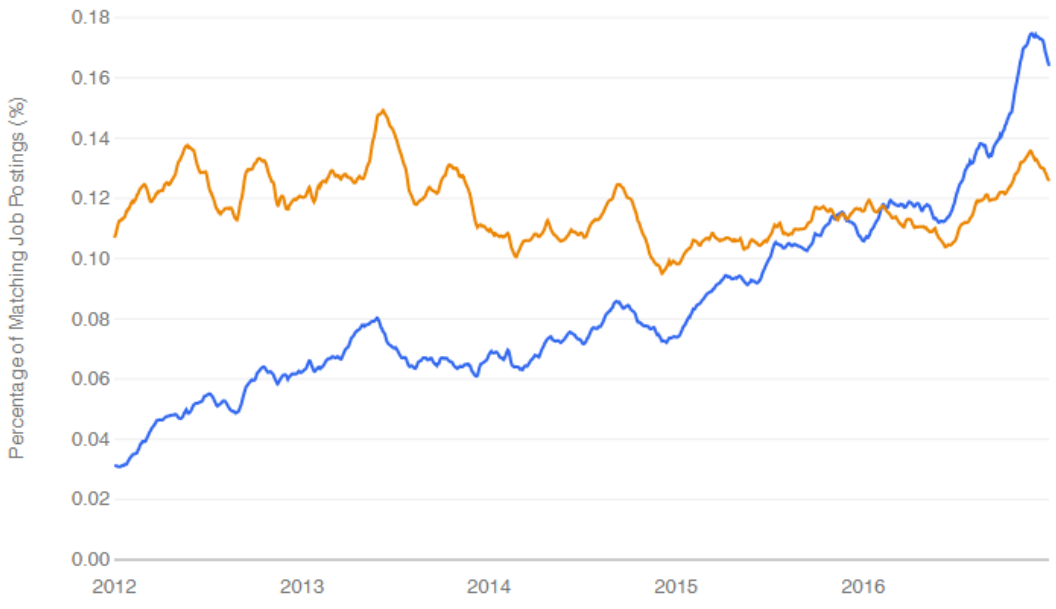

Figure 1c. Data science job trends for R (blue) and SAS (orange).

A long-standing debate has been taking place on the Internet regarding the relative place of Python and R. Ironically, this debate about data science software has involved very little actual data. However, it is possible now to at least study the job trends. Figure 1a showed us that Python is well out in front of R, at least on that single day the searches were run. What has the data looked like over time? The answer is shown in

Figure 1d.

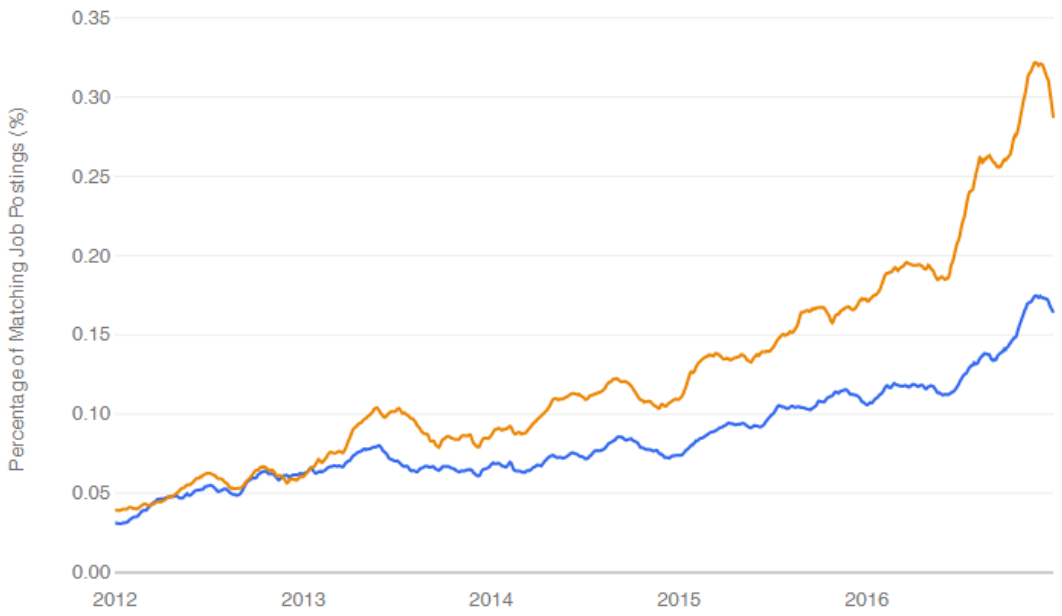

Figure 1d. Jobs trends for R (blue & lower) and Python (orange & upper).

As we see, Python surpassed R in terms of data science jobs back in 2013. These are, of course, very different languages and a quick scan of job descriptions will show that the R jobs are much more focused on the use of existing methods of analysis, while the Python jobs have more of a custom-programming angle to them.

Scholarly Articles

Scholarly articles are also rich in information and backed by significant amounts of effort. The more popular a software package is, the more likely it will appear in scholarly publications as an analysis tool or even an object of study. The software that is used in scholarly articles is what the next generation of analysts will graduate knowing, so it's a leading indicator of where things are headed. Google Scholar offers a way to measure such activity. However, no search of this magnitude is perfect; each will include some irrelevant articles and reject some relevant ones. The details of the search terms I used are complex enough to move to a companion article, *How to Search For Data Science Articles*. Since Google regularly improves its search algorithm, each year I re-collect the data for all years.

Figure 2a shows the number of articles found for each software package for 2015. SPSS is by far the most dominant package, as it has been for over 15 years. This may be due to its balance between power and ease-of-use. For the first time ever, R is in second place with around half as many articles. Although now in third place, SAS is nearly tied with R. Stata and MATLAB are essentially tied for fourth and fifth place. Starting with Java, usage slowly tapers off. Note that the general-purpose software C, C++, C#, MATLAB, Java, and Python are included only when found in combination with data science terms, so view those as much rougher counts than the rest. Since Scala

and Julia have a heavy data science angle to them, I cut them some slack by *not* filtering the search by adding data science terms. So any articles that used Scala or Julia were included in the total count for these languages regardless of usage, not that it helped them much!

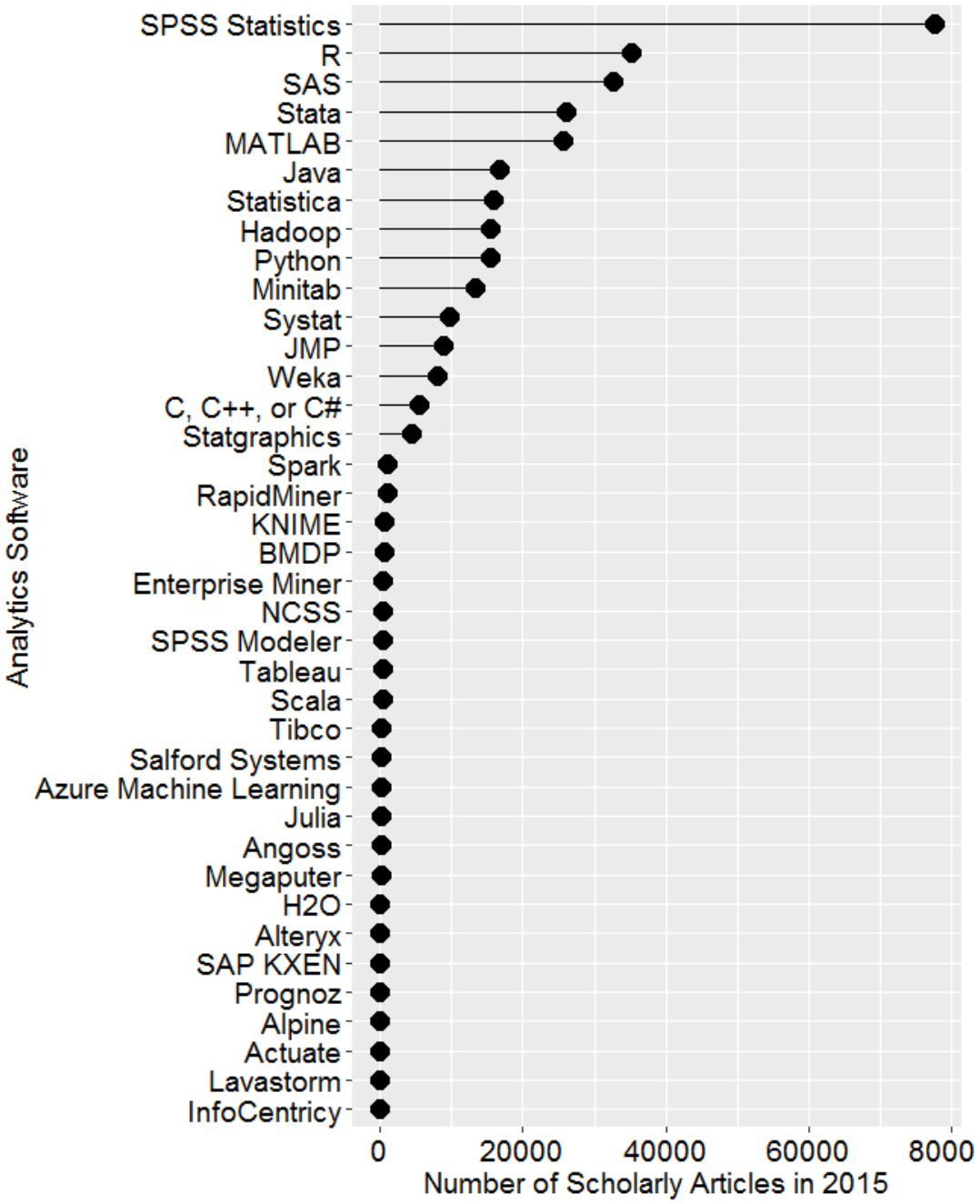

Figure 2a. Number of scholarly articles found in the most recent complete year (2015) for each software package.

From Spark on down, the counts appear to be zero, but that's not the case. The counts are just very low compared to the more popular packages, used in tens of thousands articles. Figure 2b shows the software only for those packages that have fewer than 1,200 articles (i.e. the bottom part of Fig. 2a), so we can see how they compare to other software packages with smaller counts. Spark and RapidMiner top out the list of these packages, followed by KNIME and BMDP. There's a slow decline in the group that goes from Enterprise Miner through Salford Systems. Then comes a group of mostly relative new arrivals beginning with Microsoft's Azure Machine Learning. However, this group includes Megaputer, whose Polyanalyst software has been around for many years now, with little progress to show for it. Dead last is Lavastorm, which to my knowledge is the only commercial package that includes Tibco's internally written

version of R, TERR.

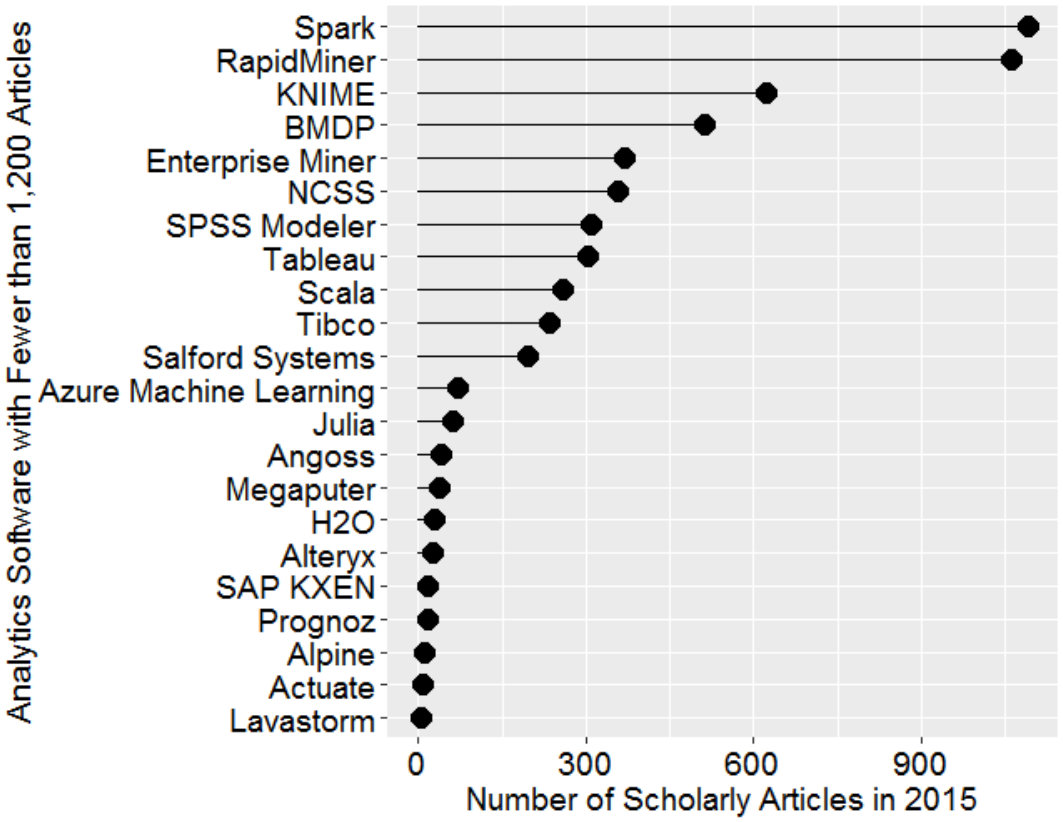

Figure 2b. The number of scholarly articles for software that was used by fewer than 1,200 scholarly articles (i.e. the bottom part of Fig. 2a, rescaled.)

Figures 2a and 2b are useful for studying market share as it is now, but they don't show how things are changing. It would be ideal to have long-term growth trend graphs for each of the analytics packages, but collecting such data is too time consuming since it must be re-collected every year. What I've done instead is collect data only for the past two complete years, 2014 and 2015. This provides the data we need to study year-over-year changes. Figure 2c shows the percent change across those years, with the "hot" packages whose use is growing shown in red. Those whose use is declining or "cooling" are shown in blue. Since the number of articles tends to be in the thousands or tens of thousands, I have removed any software that had fewer than 500 articles in 2014.

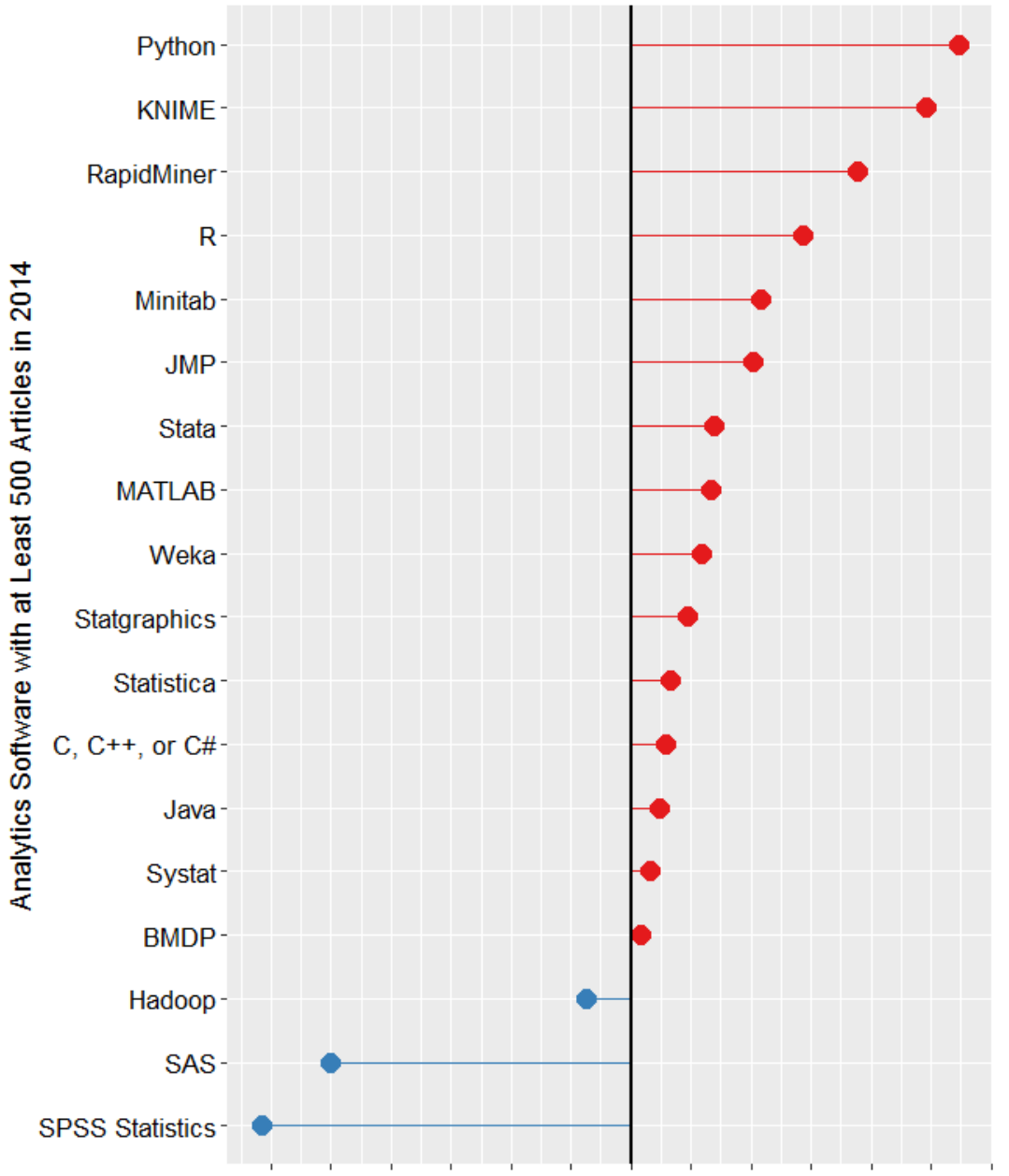

Figure 2c. Change in the number of scholarly articles using each software in the most recent two complete years (2014 to 2015). Packages shown in red are “hot” and growing, while those shown in blue are “cooling down” or declining.

Python is the fastest growing. Note that the Python figures are strictly for data science use as defined here. The open-source KNIME and RapidMiner are the second and third fastest growing, respectively. Both use the easy yet powerful workflow approach to data science. Figure 2b shows that RapidMiner has almost twice the marketshare of KNIME, but here we see use of KNIME is growing faster. That may be due to KNIME’s greater customer satisfaction, as shown in the Rexer Analytics Data Science Survey. The companies are two of only four chosen by IT advisory firm Gartner, Inc. as having both a complete vision of the future and the ability to execute that vision (Fig. 3a).

R is in fourth place in growth, and given its second place in overall marketshare, it is in an enviable position.

At the other end of the scale are SPSS and SAS, both of which declined in use by 25% or more. Recall that Fig. 2a shows that despite recent years of decline, SPSS is still extremely dominant for scholarly use. Hadoop use declined slightly, perhaps as people turned to alternatives Spark and H2O.

I'm particularly interested in the long-term trends of the classic statistics packages. So in Figure 2d I've plotted the same scholarly-use data for 1995 through 2015, the last complete year of data when this graph was made. As in Figure 2a, SPSS has a clear lead, but now you can see that its dominance peaked in 2008 and its use is in sharp decline. SAS never came close to SPSS' level of dominance, and it also peaked around 2008. Note that the decline in the number of articles that used SPSS or SAS is not balanced by the increase in the other software shown in this particular graph. However, if you add up all the other software shown in Figure 2a, you come close. There still seems to be a slight decline in people reporting the particular software tool they used.

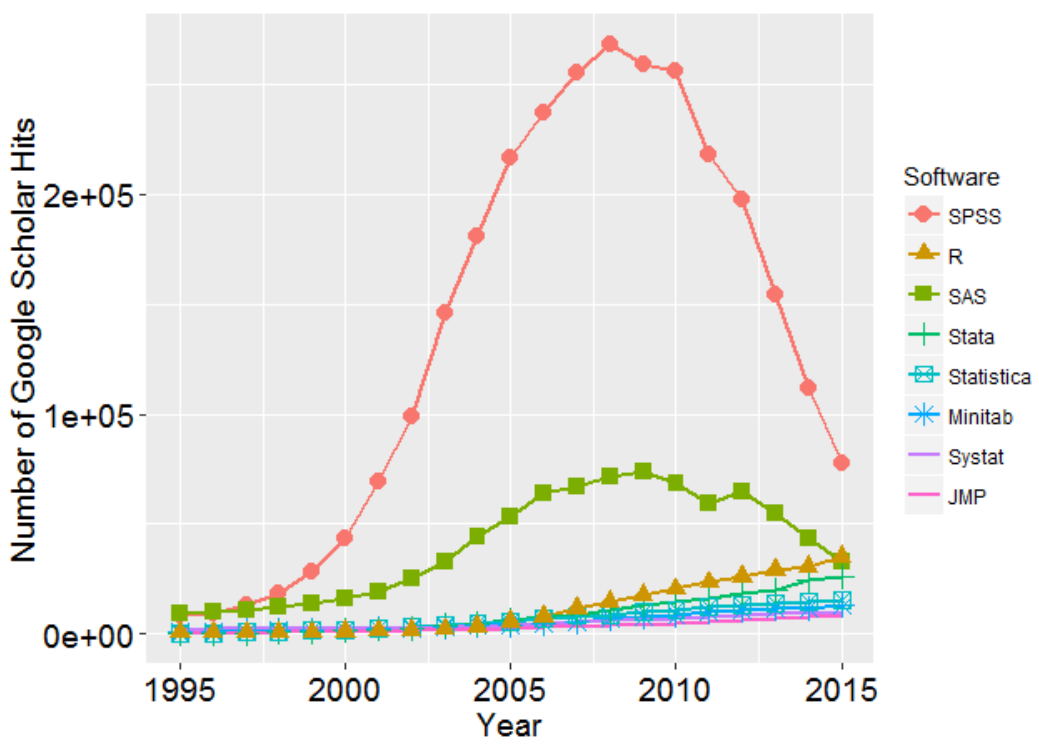

Figure 2d. The number of scholarly articles found in each year by Google Scholar. Only the top six "classic" statistics packages are shown.

Since SAS and SPSS dominate the vertical space in Figure 2d by such a wide margin, I removed those two curves, leaving only a single point of SAS usage in 2015. The result is shown in Figure 2e. Freeing up so much space in the plot now allows us to see that the growth in the use of R is quite rapid and is pulling away from the pack (recall that the curve for SAS has a steep downward slope). If the current trends continue, R will cross SPSS to become the #1 software for scholarly data science use by the end of 2017. Stata use is also growing more quickly than the rest. Note that trends have shifted before as discussed here. The use of Statistica, Minitab, Systat and JMP are next in popularity, respectively, with their growth roughly parallel to one another.

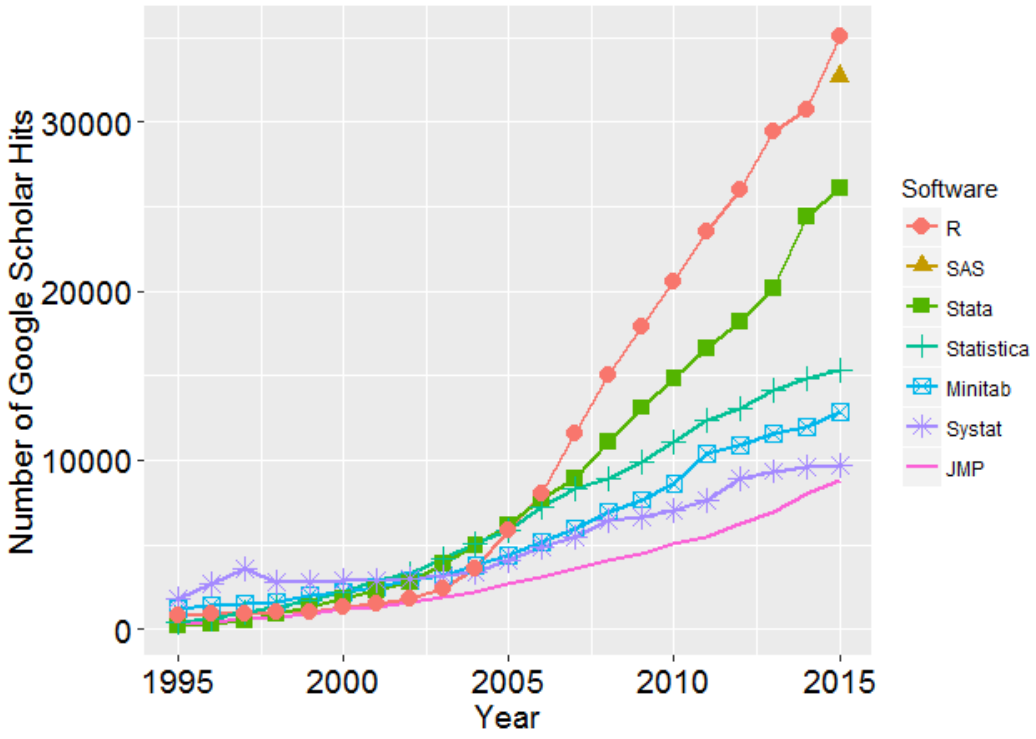

Figure 2e. The number of scholarly articles found in each year by Google Scholar for classic statistics packages after the curves for SPSS and SAS have been removed.

Using a logarithmic y-axis scales down the more popular packages, allowing us to see the full picture in a single image (Figure 2f.) This view makes it clear that R use has passed that of SAS, and that Stata use is closing in on it. However, even when one studies the y-axis values carefully, it can be hard to grasp how much the logarithmic transformation has changed the values. For example, in 2015 value for SPSS is well over twice the value for R. The original scale shown in Figure 2d makes that quite clear.

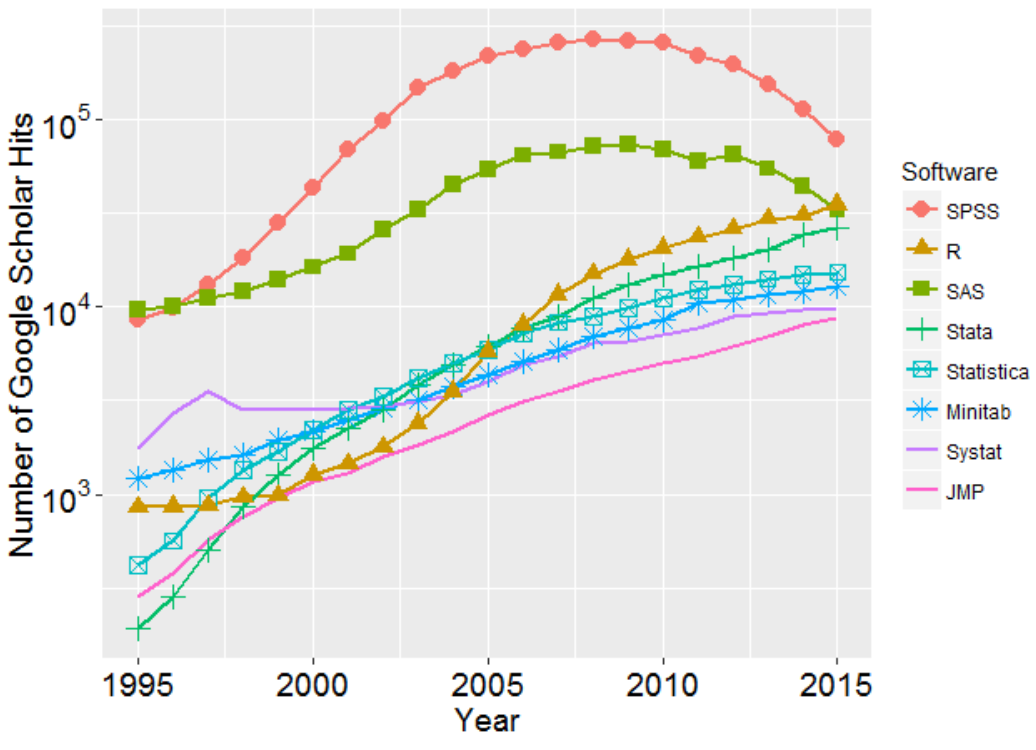

Figure 2f. A logarithmic view of the number of scholarly articles found in each year by Google Scholar. This combines the previous two figures into one by compressing the y-axis with a base 10 logarithm.

IT Research Firms

IT research firms study software products and corporate strategies, they survey

customers regarding their satisfaction with the products and services, and then provide their analysis on each in reports they sell to their clients. Each company has its own criteria for rating companies, so they don't always agree. However, I find the reports extremely interesting reading. While these reports are expensive, the companies that receive good ratings often purchase copies to give away to potential customers. An Internet search of the report title will often reveal the companies that are distributing such copies.

Gartner, Inc. is one of the companies that provides such reports. Out of the roughly 100 companies selling data science software, Gartner selected 16 which had either high revenue or lower revenue but high growth (see full report for details). After extensive input from both customers and company representatives, Gartner analysts rated the companies on their "completeness of vision" and their "ability to execute" that vision. Figure 3 shows the resulting plot. Note that purely open source software is not rated by Gartner, but nearly all the software in Figure 3 includes the ability to interact with R and Python.

The Leader's Quadrant is the place for companies who have a future direction in line with their customer's needs and the resources to execute that vision. The four companies in the Leaders quadrant have remained the same for the last three reports: IBM, KNIME, RapidMiner, and SAS. Of these, they rate IBM as having slightly greater "completeness of vision" due to the extensive integration they offer to open source software compared to SAS Institute. KNIME and RapidMiner are quite similar as the are driven by an easy to use workflow interface. Both offer free and open source versions, but RapidMiner's is limited by a cap on the amount of data that it can analyze. IBM and SAS are market leaders based on revenue and, as we have seen, KNIME and RapidMiner are the ones with high growth.

Figure 3a. Gartner Magic Quadrant for Data Science Platforms

The companies in the Visionaries quadrant are those that have a good future plans but which may not have the resources to execute that vision. Of these, Microsoft increased its ability to execute compared to the 2016 report, and Alpine, one of the smallest companies, declined sharply in their ability to execute. The remaining three companies in this quadrant have just been added: H2O.ai, Dataiku, and Domino Data Lab.

Those in the Challenger's quadrant have ample resources but less customer confidence on their future plans. Mathworks, the makers of MATLAB, is new to the report. Quest purchased Statistica from Dell, and it appears in roughly the same position as Dell did last year.

The Niche Players quadrant offer tools that are not as broadly applicable.

In 2017 Gartner dropped coverage of Accenture, Lavastorm, Megaputer, Predixion Software, and Prognoz.

Forrester Research, Inc. is another company that provides similar reports. The KDnuggets site overlaid the "Wave" plots from their two most recent reports, *The Forrester Wave: Big Data Predictive Analytics Solutions*, shown in Figure 3b. The lighter circles represent each company's position in the previous report (2013) and the more solid white circles represent their position in 2015.

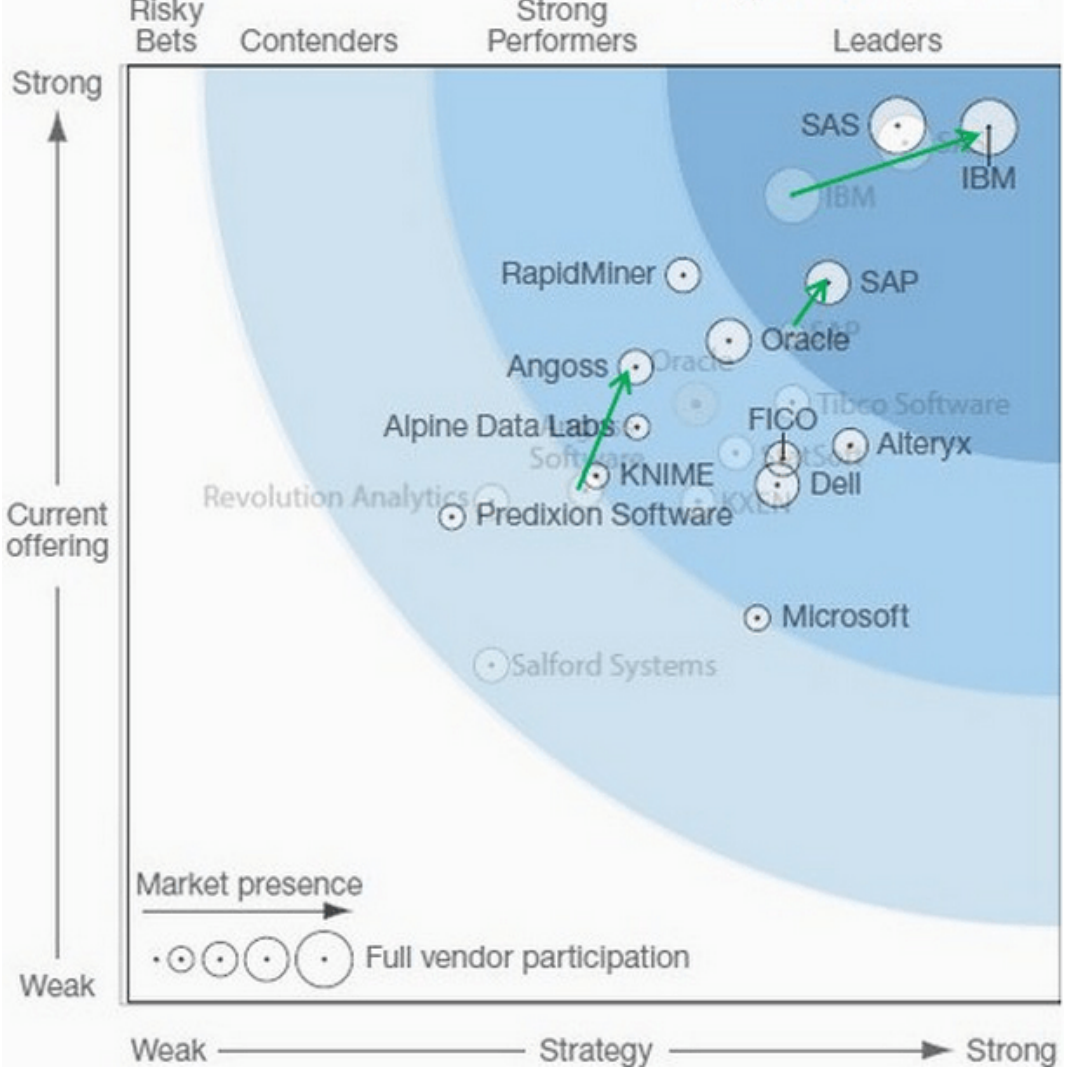

Figure 3b. Forrester Wave plot of big data predictive analytics solutions, Q1 2015 vs. Q1 2013.

Again, IBM (SPSS) and SAS are the strongest companies, but that seems to be all that the two reports seem to agree on! The reports emphasize different aspects of the companies being rated, which accounts for the radically different plots. While the Gartner report showed Angoss losing strength, the Forrester analysts see it gaining strength! RapidMiner is not in the Leaders area here, but at least it's close. KNIME on the other hand is barely inside the Strong Performer area.

Hurwitz & Associates released their *Advanced Analytics: The Hurwitz Victory Index Report* in mid 2014. Figure 3c shows their plot of strength of company strategy vs. viability. This is similar to the measures plotted in the Gartner Group's Magic Quadrant plot (Fig. 3a). These two plots both show IBM and SAS in the best position, but after that there's not much similarity. Gartner sees RapidMiner in the same ballpark as IBM and SAS, while Hurwitz shows it towards the opposite end. KNIME is also toward the top of Gartner's plot and not covered by Hurwitz at all.

Figure 3c. Hurwitz & Associates plot of corporate vision vs. viability.

Surveys of Use

Survey data adds additional information regarding software popularity, but they are commonly done using "snowball sampling" in which the survey provider tries to widely distribute the link and then vendors vie to see who can get the most of their users to participate. So long as they all do so with equal effect, the results can be useful. However, the information is often limited, because the questions are short and precise (e.g. "tools data mining" or "program languages for data mining") and responding requires just a few mouse clicks, rather than the commitment required to place a job advertisement or publish a scholarly article, book or blog post. As a result, it's not unusual to see market share jump 100% or drop 50% in a single year, which is *very* unlikely to reflect changes in actual use.

Rexer Analytics conducts a survey of data scientists every other year, asking a wide range of questions regarding data science (previously referred to as data mining by the survey itself.) Figure 4a shows the tools that the 1,220 respondents reported using in 2015.

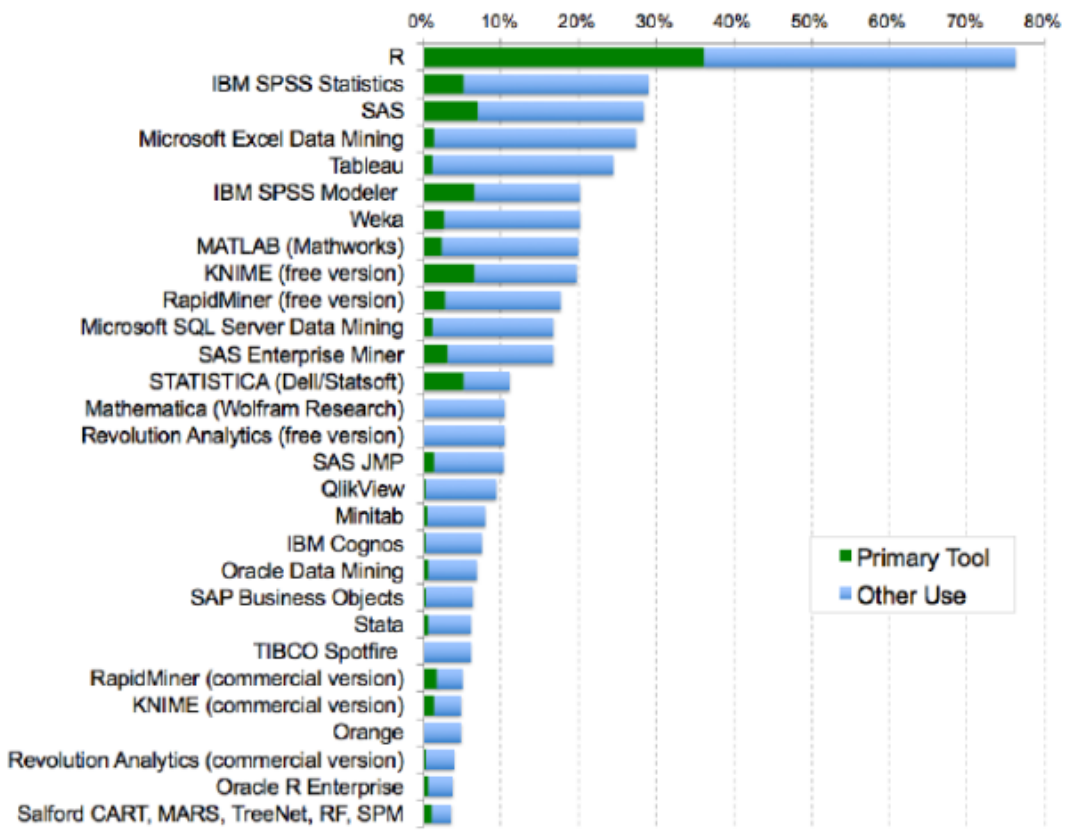

Figure 4a. Analytics tools used by respondents to the 2015 Rexer Analytics Survey. In this view, each respondent was free to check multiple tools.

We see that R has a more than 2-to-1 lead over the next most popular packages, SPSS Statistics and SAS. Microsoft's Excel Data Mining software is slightly less popular, but note that it is rarely used as the primary tool. Tableau comes next, also rarely used as the primary tool. That's to be expected as Tableau is principally a visualization tool with minimal capabilities for advanced analytics.

The next batch of software appears at first to be all in the 15% to 20% range, but KNIME and RapidMiner are listed both in their free versions and, much further down, in their commercial versions. These data come from a "check all that apply" type of question, so if we add the two amounts, we may be over counting. However, the survey also asked, "What *one* (my emphasis) data mining / analytic software package did you use most frequently in the past year?" Using these data, I combined the free and commercial versions and plotted the top 10 packages again in figure 4b. Since other software combinations are likely, e.g. SAS and Enterprise Miner; SPSS Statistics and SPSS Modeler; etc. I combined a few others as well.

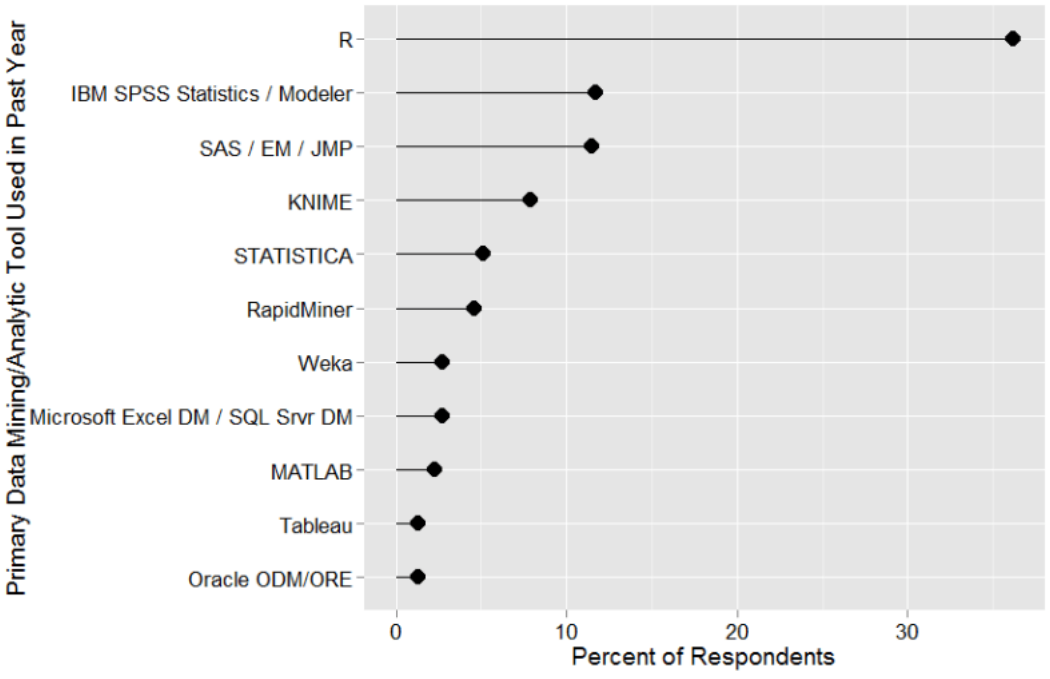

Figure 4b. The percent of survey respondents who checked each package as their *primary* tool in 2015. Note that free and commercial versions of KNIME and RapidMiner are combined. Multiple tools from the same company are also combined. Only the top 10 are shown.

In this view we see R even more dominant, with a 3-to-1 advantage compared to the software from IBM SPSS and SAS Institute. However, the overall ranking of the top three didn't change. KNIME however rises from 9th place to 4th. RapidMiner rises as well, from 10th place to 6th. KNIME has roughly a 2-to-1 lead over RapidMiner, even though these two packages have similar capabilities and both use a workflow user interface. This may be due to RapidMiner's move to a more commercially oriented licensing approach. For free, you can still get an older version of RapidMiner or a version of the latest release that is quite limited in the types of data files it can read. Even the academic license for RapidMiner is constrained by the fact that the company views "funded activity" (e.g. research done on government grants) the same as commercial work. The KNIME license is much more generous as the company makes its money from add-ons that increase productivity, collaboration and performance, rather than limiting analytic features or access to popular data formats.

The results of a similar poll done by the KDnuggets.com web site in May of 2015 are shown in Figure 4c. This one shows R in first place with 46.9% of users reporting having used it for a "real project." RapidMiner, SQL, and Python follow quite a bit lower with around 30% of users. Then at around 20% are Excel, KNIME and HADOOP. It's interesting to see that these survey results reverse the order in the previous one, showing RapidMiner as being more popular than KNIME. Both are still the top two "point-and-click" type packages generally used by non-programmers.

What Analytics, Big Data, Data mining, Data Science software you used in the past 12 months for a real project? [2759 voters]	
Legend: Red: Free/Open Source tools Green: Commercial tools Fuchsia: Hadoop/Big Data tools	% users in 2015 % users in 2014 % users in 2013
R (1293), 3.6% alone	46.9% 38.5% 37.4%
RapidMiner (870), 13.7% alone	31.5% 44.2% 39.2%
SQL (853), 0% alone	30.9% 25.3% na
Python (837), 0% alone	30.3% 19.5% 13.3%
Excel (631), 0% alone	22.9% 25.8% 28.0%
KNIME (553), 6.7% alone	20% 15.0% 5.9%
Hadoop (507), 0% alone	18.4% 12.7% 9.3%
Tableau (341), 0% alone	12.4% 9.1% 6.3%
SAS base (313), 0.6% alone	11.3% 10.9% 10.7%
Spark (311), 0% alone	11.3% 2.6% na
Weka (310), 0% alone	11.2% 17.0% 14.3%
SAS Enterprise Miner (302), 3.6% alone	10.9% 7.2% 5.9%
Microsoft SQL Server (268), 0% alone	9.7% 10.5% 7.0%
MATLAB (243), 0% alone	8.8% 8.4% 9.9%
scikit-learn (229), 0% alone	8.3% na na
Unix shell/awk/gawk (221), 0% alone	8.0% 5.8% na
IBM SPSS Statistics (213), 0% alone	7.7% 7.7% 8.7%
IBM SPSS Modeler (197), 7.1% alone	7.1% 5.7% 6.1%
Alteryx (155), 39.4% alone	5.6% 3.1% 0.3%
Pig (150), 0% alone	5.4% 3.5% na
Other programming languages (140), 0% alone	5.1% 3.0% na
Other free analytics/data mining tools (138), 0% alone	5.0% 5.1% 3.4%
Other Hadoop/HDFS-based tools (125), 0% alone	4.5% 3.9% na

Figure 4c. Percent of respondents that used each software in KDnuggets' 2015 poll. Only software with 5% market share are shown. The % alone is the percent of tool voters that used only that tool alone. For example, only 3.6% of R users have used only R, while 13.7% of RapidMiner users indicated they used that tool alone. Years are color coded, with 2015, 2014, 2013 from top to bottom.

O'Reilly Media conducts an annual Data Science Salary Survey which also asks questions about analytics tools. Although the full report of results As their report notes,"O'Reilly content—in books, online, and at conferences—is focused on technology, in particular new technology, so it makes sense that our audience would tend to be early adopters of some of the newer tools." The results from their "over 600" respondents are shown in figures 6d and 6e.

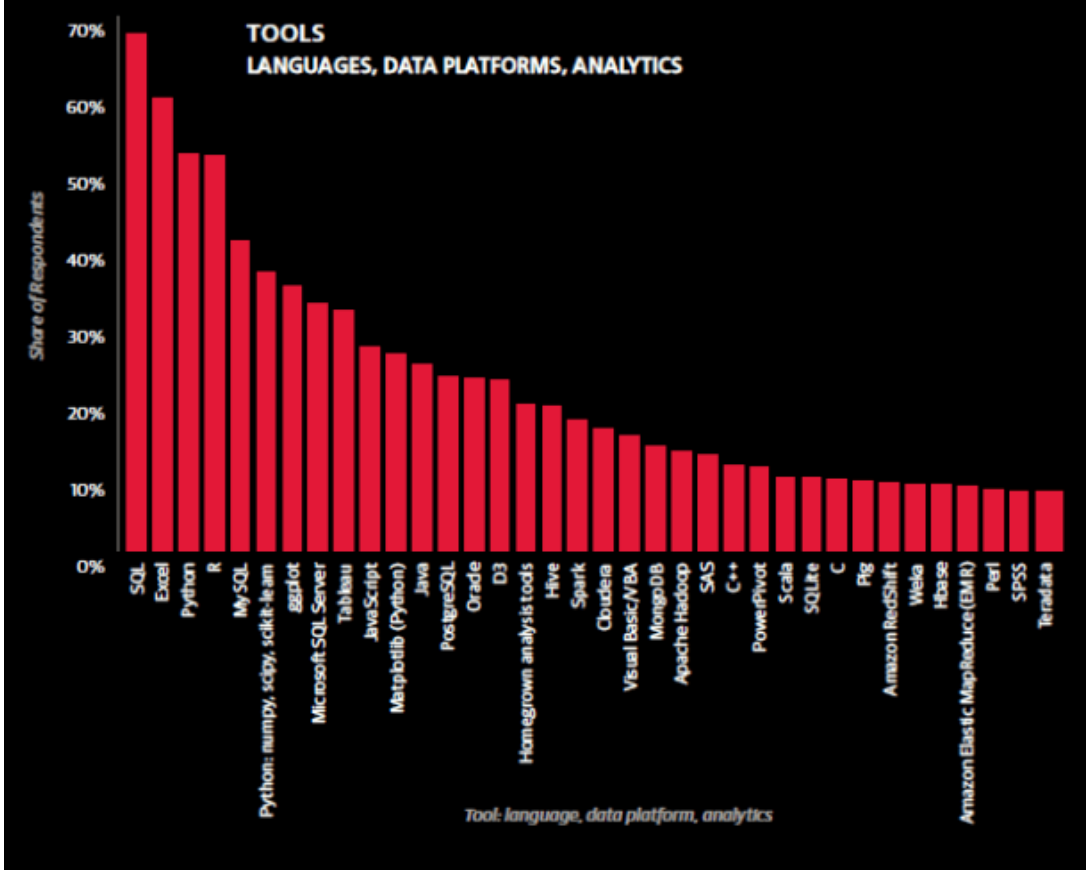

Figure 4d. Tools used by 2015 respondents to O'Reilly 2015 salary survey. The less popular tools among this audience are shown in the following figure.

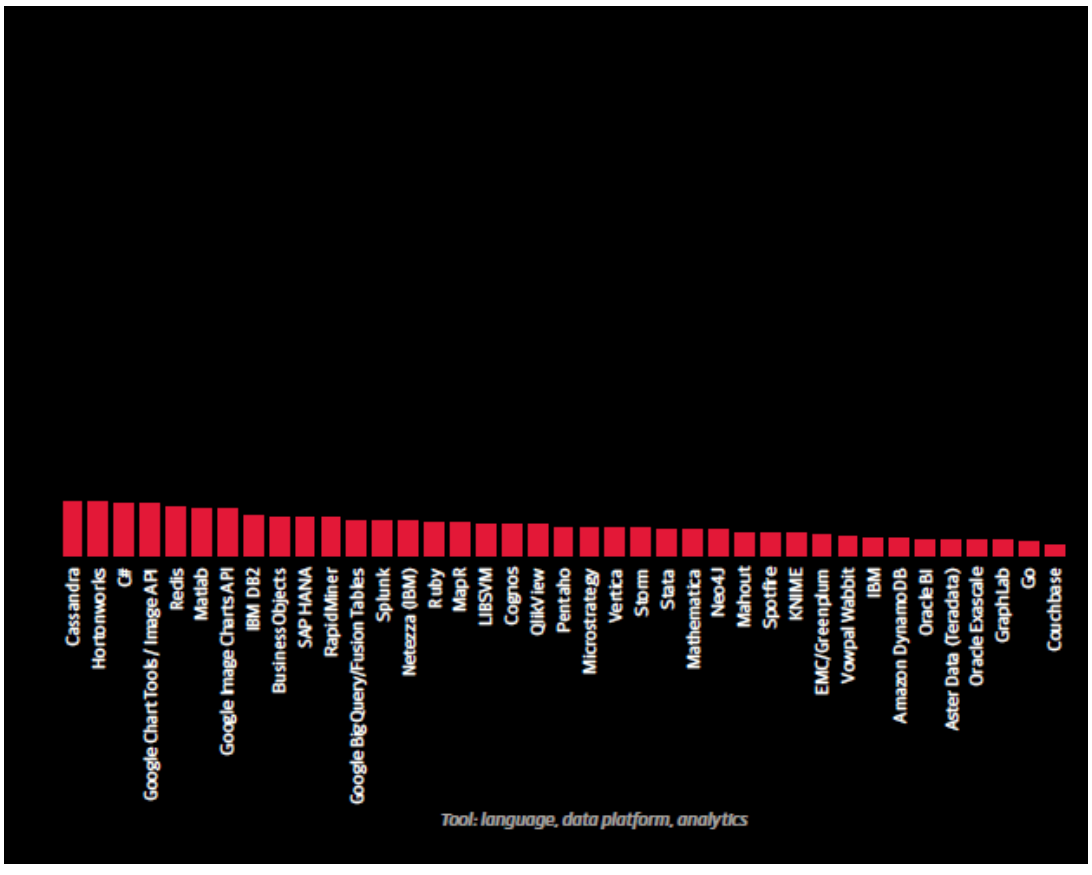

Figure 4e. The less popular tools used by the respondents of O'Reilly's 2015 salary survey.

The O'Reilly results have SQL in first place with 70% of users reporting it, followed closely by Excel. Python and R follow seemingly tied for third place with 55%. However, Python also appears in 6th place with its subroutine libraries numpy, etc., and R's popular ggplot package appears in 7th place, with around 38% market share. The first commercial package with deep analytic capabilities is SAS in 23rd place! This emphasizes that the O'Reilly sample is heavily weighted towards their usual open source audience. Hopefully in the future they will advertise the survey to a wide

audience and do so as more than just a salary survey. Tool surveys gain additional respondents since they are advertised by advocates of the various tools (vendors, fans, etc.)

Lavastorm, Inc. conducted a survey of analytic communities including LinkedIn's Lavastorm Analytics Community Group, Data Science Central and KDnuggets. The results were published in March, 2013, and the bar chart of "self-service analytic tool" usage among their respondents is shown in Figure 6f. Excel comes out as the top tool, with 75.6% of respondents reporting its use.

R comes out as the top advanced analytics tool with 35.3% of respondents, followed closely by SAS. MS Access' position in 4th place is a bit of an outlier as no other surveys include it at all. Lavastorm comes out with 3.4%, while other surveys don't show them at all. That's hardly a surprise given than the survey was aimed at the Lavastorm's LinkedIn community group.

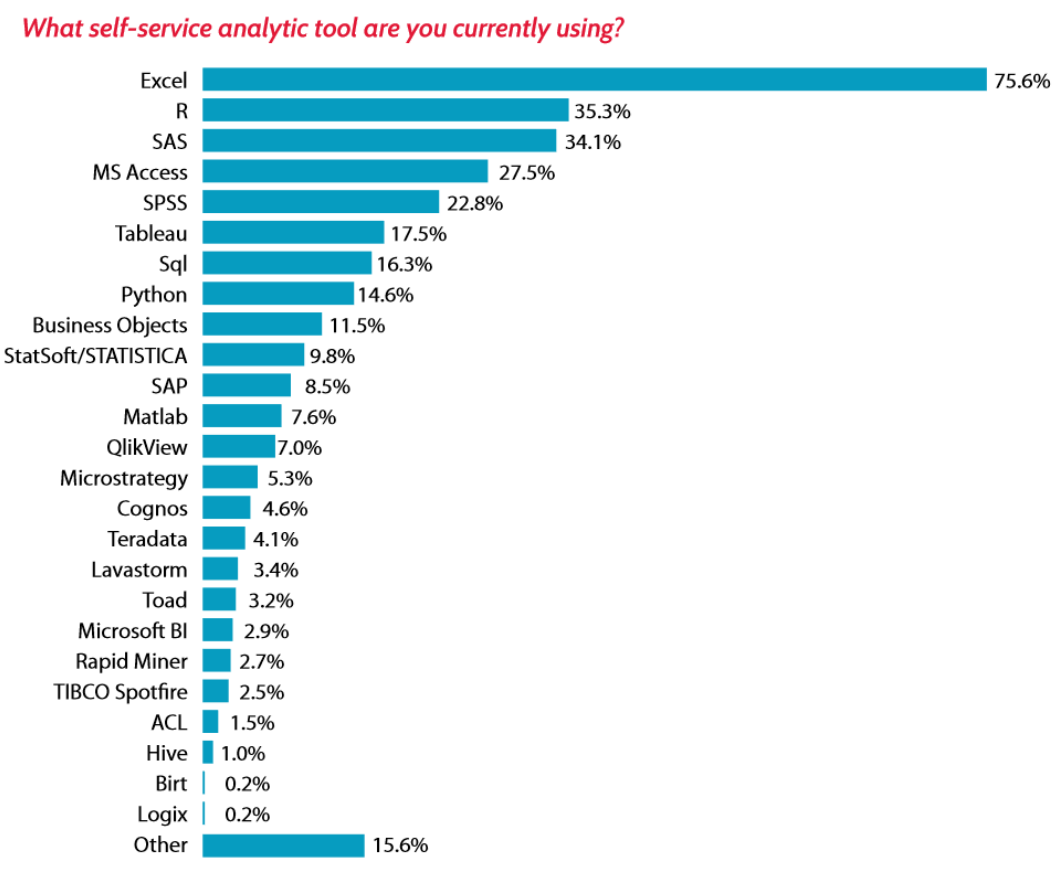

Figure 4f. Lavastorm survey of analytics tools.

Books

The number of books that include a software's name in its title is a particularly useful information since it requires a significant effort to write one and publishers do their own study of market share before taking the risk of publishing. However, it can be difficult to do searches to find books that use general-purpose languages which also focus only on analytics. Amazon.com offers an advanced search method which works well for all the software except R and the general-purpose languages such as Java, C, and MATLAB. I did not find a way to easily search for books on analytics that used such general purpose languages, so I've excluded them in this section.

The Amazon.com advanced search configuration that I used was (using SAS as an example):

```
Title: SAS -excerpt -chapter -changes -articles
Subject: Computers & Technology
Condition: New
Format: All formats
Publication Date: After January, 2000
```

The “title” parameter allowed me to focus the search on books that included the software names in their titles. Other books may use a particular software in their examples, but they’re impossible to search for easily. SAS has many manuals for sale as individual chapters or excerpts. They contain “chapter" or “excerpt" in their title so I excluded them using the minus sign, e.g. “-excerpt". SAS also has short “changes and enhancements" booklets that the developers of other packages release only in the form of flyers and/or web pages, so I excluded “changes" as well. Some software listed brief “articles” which I also excluded. I did the search on June 1, 2015, and I excluded excerpts, chapters, changes, and articles from all searches.

“R” is a difficult term to search for since it’s used in book titles to indicate Registered Trademark as in “SAS(R)”. Therefore I verified all the R books manually.

The results are shown in the table immediately below, where it’s clear that a very small number of analytics software packages dominate the world of book publishing. SAS has a huge lead with 576 titles, followed by SPSS with 339 and R with 240. SAS and SPSS both have many versions of the same book or manual still for sale, so their numbers are both inflated as a result. JMP and Hadoop both had fewer than half of R’s count and then Minitab and Enterprise Miner had fewer then half again as many. Although I obtained counts on all 27 of the domain-specific (i.e. not general-purpose) analytics software packages or languages shown in Figure 2a, I cut the table off at software that had 8 or fewer books to save space.

Software	**Number of Books**	
SAS	576	
SPSS Statistics	339	
R	240	[Corrected from blog post: 172]
JMP	97	
Hadoop	89	
Stata	62	
Minitab	33	
Enterprise Miner	32	

Table 1. The number of books whose titles contain the name of each software package.

Blogs

On Internet blogs, people write about software that interests them, showing how to solve problems and interpreting events in the field. Blog posts contain a great deal of information about their topic, and although it's not as time consuming as a book to write, maintaining a blog certainly requires effort. Therefore, the number of bloggers writing about analytics software has potential as a measure of popularity or market share. Unfortunately, counting the number of *relevant* blogs is often a difficult task. General purpose software such as Java, Python, the C language variants and MATLAB have many more bloggers writing about general programming topics than just analytics. But separating them out isn't easy. The name of a blog and the title of its latest post may not give you a clue that it routinely includes articles on analytics.

Another problem arises from the fact that what some companies would write up as a newsletter, others would do as a set of blogs, where several people in the company each contribute their own blog. Those individual blogs may also combined into a single company blog inflating the count further still. Statsoft and Minitab offer examples of this. So what's really interesting is not company employees who are assigned to write blogs, but rather those written by outside volunteers. In a few lucky cases, lists of such blogs are maintained, usually by blog consolidators, who combine many blogs into a large "metablog." All I have to do is find such lists and count the blogs. I don't attempt to extract the few vendor employees that I know are blended into such lists. I only skip those lists that are exclusively employee-based (or very close to it). The results are shown here:

Software	Number of Blogs	Source
R	550	R-Bloggers.com
Python	60	SciPy.org
SAS	40	PROC-X.com, sasCommunity.org Planet
Stata	11	Stata-Bloggers.com

Table 2. Number of blogs devoted to each software package on April 7, 2014, and the source of the data.

R's 550 blogs is quite an impressive number. For Python, I could only find that list of 60 that were devoted to the SciPy subroutine library. Some of those are likely cover topics besides analytics, but to determine which never cover the topic would be quite time consuming. The 40 blogs about SAS is still an impressive figure given that Stata was the only other company that even garnered a list anywhere. That list is at the vendor

itself, Statacorp, but it consists of non-employees except for one.

While searching for lists of blogs on other software, I did find individual blogs that at least occasionally covered a particular topic. However, keeping this list up to date is far too time consuming given the relative ease with which other popularity measures are collected.

If you know of other lists of relevant blogs, please let me know and I'll add them. If you're a software vendor employee reading this, and your company does not build a metablog or at least maintain a list of your bloggers, I recommend taking advantage of this important source of free publicity.

Discussion Forum Activity

Another way to measure software popularity is to see how many people are helping one another use each package or language. While such data is readily available, it too has its problems. Menu-driven software like SPSS or workflow-driven software such as KNIME are quite easy to use and tend to generate fewer questions. Software controlled by programming requires the memorization of many commands and requiring more support. Even within languages, some are harder to use than others, generating more questions (see Why R is Hard to Learn).

Another problem with this type of data is that there are many places to ask questions and each has its own focus. Some are interested in a classical statistics perspective while others have a broad view of software as general-purpose programming languages. In recent years, companies have set up support sites within their main corporate web site, further splintering the places you can go to get help. Usage data for such sites is not readily available.

Another problem is that it's not as easy to use logic to focus in on specific types of questions as it was with the data from job advertisements and scholarly articles discussed earlier. It's also not easy to get the data across time to allow us to study trends. Finally, the things such sites measure include: software group members (a.k.a. followers), individual topics (a.k.a. questions or threads), and total comments across all topics (a.k.a. total posts). This makes combining counts across sites problematic.

Two of the biggest sites used to discuss software are LinkedIn and Quora. They both display the number of people who follow each software topic, so combining their figures makes sense. However, since the sites lack any focus on analytics, I have not collected their data on general purpose languages like Java, MATLAB, Python or variants of C. The results of data collected on 10/17/2015 are shown here:

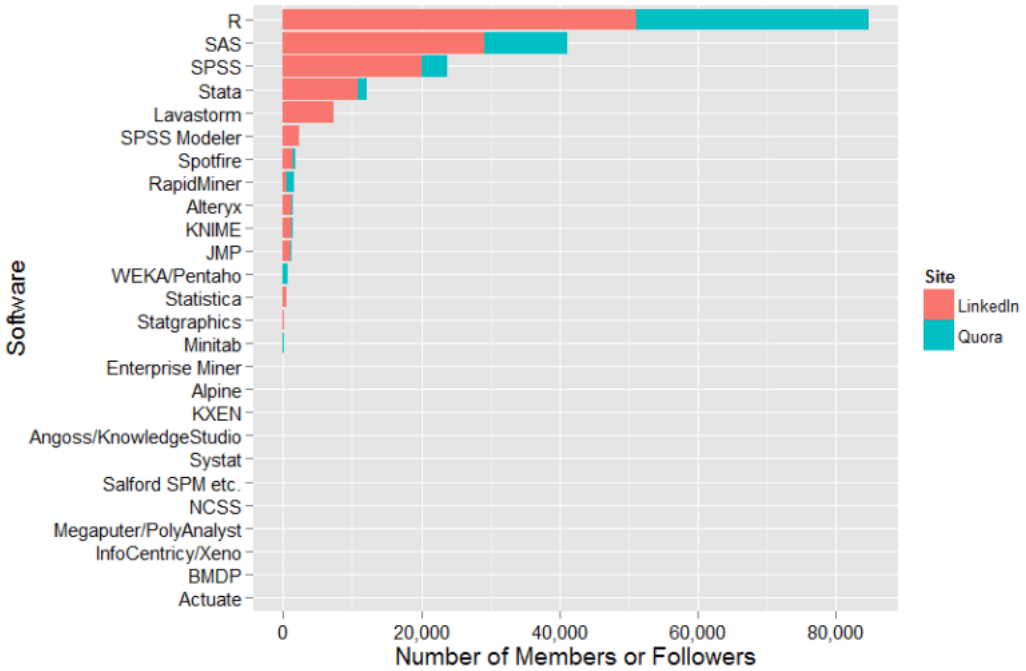

Figure 7a. Number of people who follow each software on LinkedIn and Quara.

We see that R is the dominant software and that moving down through SAS, SPSS, and Stata results in a loss of roughly half the number of people in each step. Lavastorm follows Stata, but I find it odd that there was absolutely zero discussion of Lavastorm on Quora. The last bar that you can even see on this plot is the 62 people who follow Minitab. All the ones below that have tiny audiences of fewer than 10.

Next let's examine two sites that focus only on statistical questions: Talk Stats and Cross Validated. They both report the number of questions (a.k.a. threads) for a given piece of software, allowing me to total their counts:

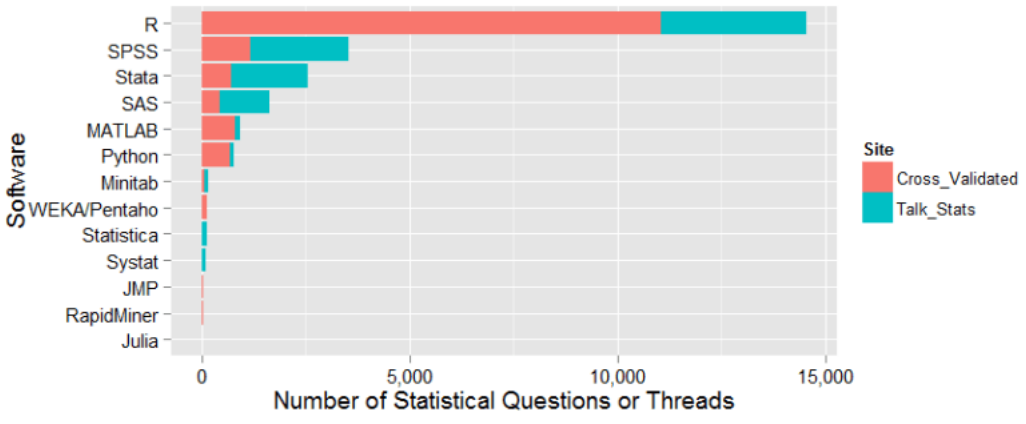

Figure 7b. Number of questions and for each software on Talk Stats and Cross Validated. Those not shown had no questions.

We see that R has a 4-to-1 lead over the next most popular package, SPSS. Stata comes in at 3rd place, followed by SAS. The fact that SAS is in fourth place here may be due to the fact that it is strong in data management and report writing, which are not the types of questions that these two sites focus on. Although MATLAB and Python are general purpose languages, I include them here because the questions on this site are within the realm of analytics. Note that I collected data on as many packages as were shown in the previous graph, but those not shown have a count of zero. Julia

appears to have a count of zero due to the scale of the graph, but it actually had 5 questions on Cross Validated.

Programming Popularity Measures

Several web sites rank the popularity of programming languages. Unfortunately, they don't differentiate between general-purpose languages and application-specific ones used for analytics. However, it's easy to choose the few analytics languages their results.

The most comprehensive of these sites is the IEEE Spectrum Ranking. This site combines 12 metrics from 10 different sites. These include some of the measures discussed above, such as popularity on job sites and search engines. They also include fascinating and useful measures such as how much new programming code was added to the popular GitHub repository in the last year. This figure shows their top 10 languages for 2015:

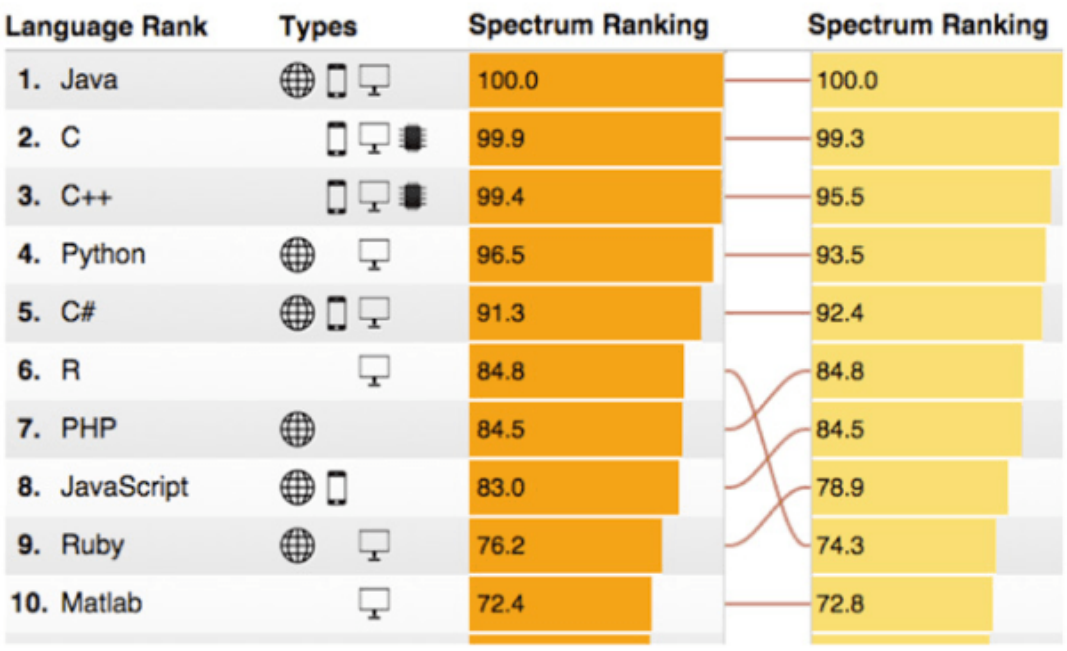

Figure 8a. IEEE Spectrum language popularity rankings. The left column (orange) shows the 2015 ranking, while the left (yellow) one shows the 2014 ranking.

We see that R is in 6th place and that it has increased from 9th place in 2014. Not shown on this is SAS in 26th place. Python is ranked in 4th place, but that's for all purposes, while the use of R is more focused on analytics. No other analytics-specific language makes it in their rankings at all. This ranking is based on a weighted composite score and the site is interactive, allowing you to generate a ranking more suited to your needs.

The next most comprehensive analysis is provded by RedMonk. Their analysis is simple and objective. They plot the number of lines of code written using each language on the popular Github repository against the number of tagged comments on the discussion forum StackOverflow.com. Here is the result:

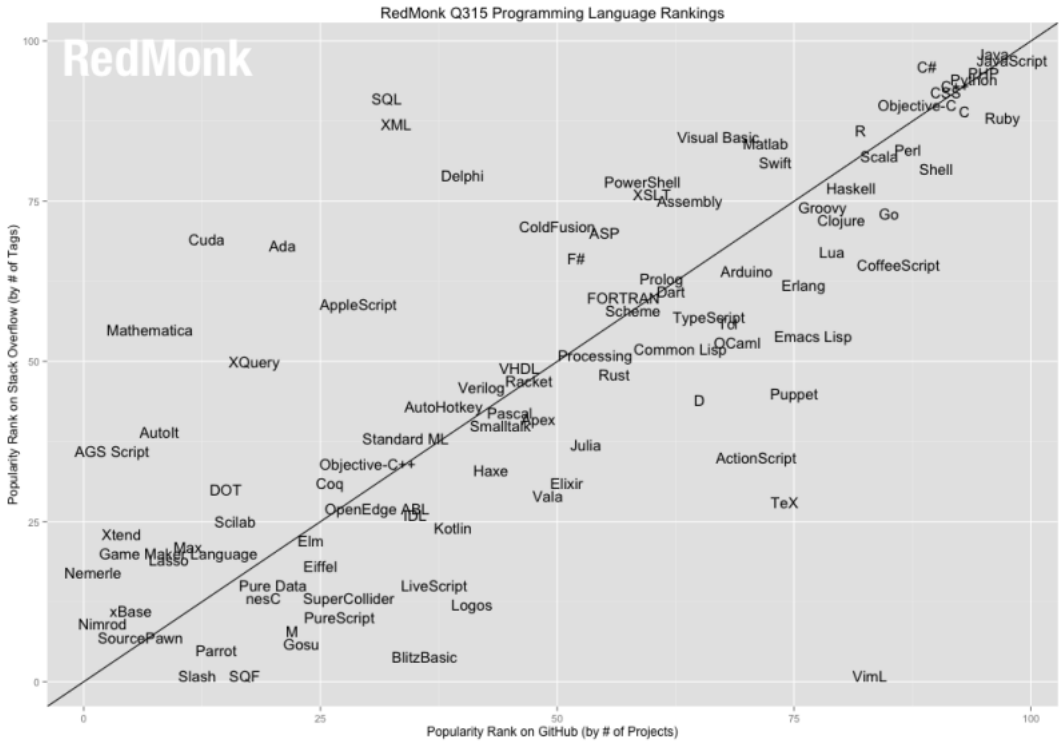

Figure 8b. RedMonkpProgramming language popularity as measured by number of projects on GitHub and the amount of discussion on StackOverflow.

Moving from the upper right corner downward the lower left, we can see that Redmonk's approach shows R as a very popular language, around 12th place. Although a substantial amount of the metrics for Python, MATLAB, and Julia may be due to analytics use, we have no way of knowing how much.

The TIOBE Community Programming Index also ranks the popularity of programming languages. It extracts measurements from the 25 most popular search engines including Google, YouTube, Wikipedia, Amazon.com, and combines them into a single index. In their October 2015 rankings, they place R in 20th place and SAS in 23rd. Stata is in a bundle they call "the next 50" languages, whose popularity among general-purpose languages is so sparse that their relative rankings are too unstable to bother giving individual ranks. SPSS is a language they monitor, but it doesn't make it into their top 100. This brings us to an important limitation of the Tiobe index: it searches for one single string: "X programming." So if it didn't find "SPSS programming" then it doesn't count. The complex searches that I used for jobs and scholarly articles was far more useful in estimating each package's popularity. Another limitation to the Tiobe index is that it measures what is on the Internet now, so it's a lagging indicator. There's no way to plot trends without purchasing their data, which is quite expensive.

A very similar popularity index is PYPL PopularitY of Programming Language. It only tracks the top 15 languages and, in October of 2015 it placed R in 11th place. It searches on the single string, "X tutorial" making it a leading indicator of what's likely to be more popular in the future.

The Transparent Language Popularity Index is very similar to the TIOBE Index with

except that its ranking software, algorithm and data are published for all to see. Work on this index ceased as of July, 2013.

Sales & Downloads

Sales figures reported by some commercial vendors include products that have little to do with analysis. Many vendors don't release sales figures, or they release them in a form that combines many different products, making the examination of a particular product impossible. For open source software such as R, you could count downloads, but one confused person can download many copies, inflating the total. Conversely, many people can use a single download on a server, deflating it.

Download counts for the R-based Bioconductor project are located here. Similar figures for downloads of Stata add-ons (not Stata itself) are available here. A list of Stata repositories is available here. The many sources of downloads both in repositories and individuals' web sites makes counting downloads a very difficult task.

Competition Use

Kaggle.com is a web site that sponsors data analysis contests. People post data analysis problems there along the amount of money they are willing pay the person or team who solves their problem the best. Figure 9 the software used by the data analysts working on the problems. R is in the lead by a wide margin. R's dominance is even greater among the contest winners, over 50% of whom used R. A potential source of bias in these figures is that the licenses of most proprietary software prohibits its use for the benefit of outside organizations (universities can help federal grant-providing agencies such as NSF and NIH, but cannot even solve problems for government agencies in general or nonprofits). However, I manage the research software site licenses at the University of Tennessee, and I can attest to the fact that people are often unaware of this limitation; those who are aware, often ignore it. (Note that as of 4/19/2016 this graph of 2011 data is still Kaggle's most current graph.)

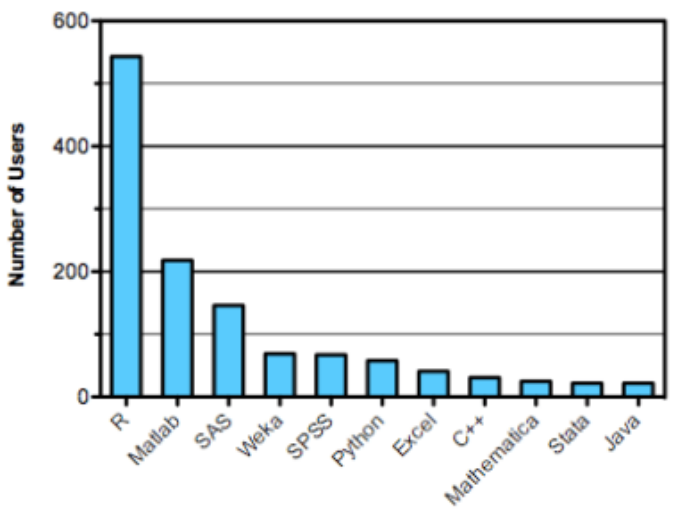

Figure 9. Software used in data analysis competitions in 2011 (checked in 2016).

Growth in Capability

The capability of analytics software has grown significantly over the years. It would be helpful to be able to plot the growth of each software package's capabilities, but such data are hard to obtain. John Fox (2009) acquired them for R's main distribution site http://cran.r-project.org/ for each version of R. To simplify ongoing data collection, I kept only the values for the last version of R released each year (usually in November or December), and collected data through the most recent complete year.

These data are displayed in Figure 10. The right-most point is for version 3.2.3, released 12/10/2015. The growth curve follows a rapid parabolic arc (quadratic fit with R-squared=.995).

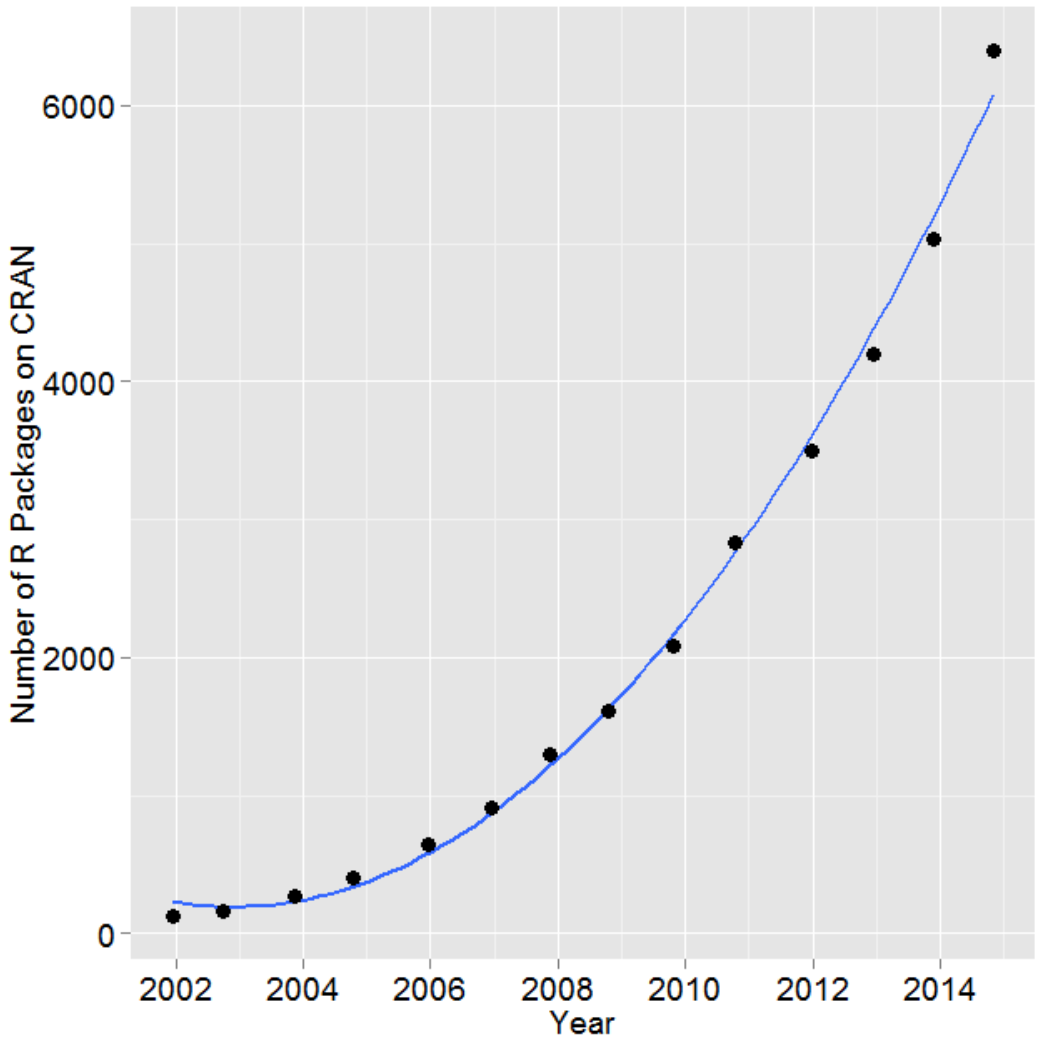

Figure 10. Number of R packages available on its main distribution site for the last version released in each year.

To put this astonishing growth in perspective, let us compare it to the most dominant commercial package, SAS. In version, 9.3, SAS contained around 1,200 commands that are roughly equivalent to R functions (procs, functions, etc. in Base, Stat, ETS, HP Forecasting, Graph, IML, Macro, OR, and QC). In 2015, R added 1,357 packages, counting only CRAN, or approximately 27,642 functions. During 2015 alone, R added more functions/procs than SAS Institute has written *in its entire history*.

Of course while SAS and R commands solve many of the same problems, they are

certainly not perfectly equivalent. Some SAS procedures have many more options to control their output than R functions do, so one SAS procedure may be equivalent to many R functions. On the other hand, R functions can nest inside one another, creating nearly infinite combinations. SAS is now out with version 9.4 and I have not repeated the arduous task of recounting its commands. If SAS Institute would provide the figure, I would include it here. While the comparison is far from perfect, it does provide an interesting perspective on the size and growth rate of R.

As rapid as R's growth has been, these data represent only the main CRAN repository. R has eight other software repositories, such as Bioconductor, that are not included in Fig. 10. A program run on 4/19/2016 counted 11,531 R packages at all major repositories, 8,239 of which were at CRAN. (I excluded the GitHub repository since it contains duplicates to CRAN that I could not easily remove.) So the growth curve for the software at all repositories would be approximately 40% higher on the y-axis than the one shown in Figure 10.

As with any analysis software, individuals also maintain their own separate collections available on their web sites. However, those are not easily counted.

What's the total number of R functions? The Rdocumentation site shows the latest counts of both packages and functions on CRAN, Bioconductor and GitHub. They indicate that there is an average of 19.78 functions per package. Given the package count of 11,531, as of 4/19/2016 there were approximately 228,103 total functions in R. In total, R has approximately 190 times as many commands as its main commercial competitor, SAS.

What's Missing?

I previously included graphs from Google Trends. That site tracks not what's actually on the Internet via searches, but rather the keywords and phrases that people are entering into their Google searches. That ended up being so variable as to be essentially worthless. For an interesting discussion of this topic, see this article by Rick Wicklin.

Website Popularity – in previous editions I have included measures of this. However, as the corporate landscape has consolidated, we end up comparing huge companies with interests far outside the field of analytics (e.g. IBM) with relatively small focused ones, which no longer makes sense.

Conclusion

Although the ranking of each package varies depending on the criteria used, we can still see major trends. Among the software that tends to be used as a collection of pre-

written methods, R, SAS, SPSS and Stata tend to always be in the top, with R and SAS occasionally swapping places depending on the criteria used. I don't include Python in this group as I rarely see someone using it exclusively to call pre-written routines.

Among software that tends to be used as a language for analytics, C/C#/C++, Java, MATLAB, Python, R and SAS are always towards the top. I list those in alphabetical order since many of the measures cover not only use for analytics but for other uses as well. Among my colleagues, those who are more towards the computer science side of the data science field tend to prefer Python, while those who are more towards the statistics send tend to prefer R. A language worth mentioning is Julia, whose goal is to have syntax as clean as Pythons while maintaining the top speed reached by the C/C#/C++ group.

A trend that I find very interesting is the rise of software that uses the workflow (or flowchart) style of control. While menu-driven software is easy to learn, it's not easy to re-use the work. Workflow-driven software is almost as easy — the dialog boxes that control each node are almost identical to menu-driven software — but you also get to save and re-use the work. Software that uses this approach includes: KNIME, Microsoft Azure Machine Learning, RapidMiner, SPSS Modeler (the first to popularize this approach), SAS Enterprise Miner, SAS Studio, and even two cloud-based system that I have not been tracking, Dotplot Designer and Microsoft Azure Machine Learning. The wide use of this interface is allowing non-programmers to make use of advanced analytics.

I'm interested in other ways to measure software popularity. If you have any ideas on the subject, please contact me at muenchen.bob@gmail.com.

If you are a SAS or SPSS user interested in learning more about R, you might consider my book, *R for SAS and SPSS Users*. Stata users might want to consider reading *R for Stata Users*, which I wrote with Stata guru Joe Hilbe. I also teach workshops on these topics both online and with site visits.

Acknowledgments

I am grateful to the following people for their suggestions that improved this article: John Fox (2009) provided the data on R package growth; Marc Schwartz (2009) suggested plotting the amount of activity on e-mail discussion lists; Duncan Murdoch clarified the pitfalls of counting downloads; Martin Weiss pointed out both how to query Statlist for its number of subscribers; Christopher Baum provided information regarding counting Stata downloads; John (Jiangtang) HU suggested I add more detail from the TIOBE index; Andre Wielki suggested the addition of SAS Institute's support forums; Kjetil Halvorsen provided the location of the expanded list of Internet R

discussions; Dario Solari and Joris Meys suggested how to improve Google Insight searches; Keo Ormsby provded useful suggestions regarding Google Scholar; Karl Rexer provided his data mining survey data; Gregory Piatetsky-Shapiro provided his KDnuggets data mining poll; Tal Galili provided advice on blogs and consolidation, as well as Stack Exchange and Stack Overflow; Patrick Burns provided general advice; Nick Cox clarified the role of Stata's software repositories and of popularity itself; Stas Kolenikov provided the link of known Stata repositories; Rick Wicklin convinced me to stop trying to get anything useful out of Google Insights; Drew Schmidt automated some of the data collection; Peter Hedström greatly improved my search string for Stata; Josh Price and Janet Miles provided expert editorial advice.

Bibliography

J. Fox. Aspects of the Social Organization and Trajectory of the R Project. *R Journal*, http://journal.r-project.org/archive/2009-2/RJournal_2009-2_Fox.pdf

R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5:299–314, 1996.

R. Muenchen, *R for SAS and SPSS Users*, Springer, 2009

R. Muenchen, J. Hilbe, *R for Stata Users*, Springer, 2010

M. Schwartz, 1/7/2009, http://tolstoy.newcastle.edu.au/R/e6/help/09/01/0517.html

Trademarks