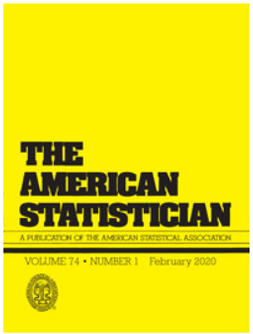

The American Statistician

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

Fostering Undergraduate Data Science

Fulya Gokalp Yavuz & Mark Daniel Ward

To cite this article: Fulya Gokalp Yavuz & Mark Daniel Ward (2020) Fostering Undergraduate Data Science, The American Statistician, 74:1, 8-16, DOI: 10.1080/00031305.2017.1407360

To link to this article: https://doi.org/10.1080/00031305.2017.1407360

Accepted author version posted online: 19 Dec 2017.
Published online: 05 Jun 2018.

THE AMERICAN STATISTICIAN
2020, VOL. 74, NO. 1, 8–16: Data Science
https://doi.org/10.1080/00031305.2017.1407360

Taylor & Francis
Taylor & Francis Group

Fostering Undergraduate Data Science

Fulya Gokalp Yavuz and Mark Daniel Ward

Department of Statistics, Purdue University, West Lafayette, IN

ABSTRACT

Data Science is one of the newest interdisciplinary areas. It is transforming our lives unexpectedly fast. This transformation is also happening in our learning styles and practicing habits. We advocate an approach to data science training that uses several types of computational tools, including R, bash, awk, regular expressions, SQL, and XPath, often used in tandem. We discuss ways for undergraduate mentees to learn about data science topics, at an early point in their training. We give some intuition for researchers, professors, and practitioners about how to effectively embed real-life examples into data science learning environments. As a result, we have a unified program built on a foundation of team-oriented, data-driven projects.

ARTICLE HISTORY
Received January 2017
Revised October 2017

KEYWORDS
Computation; Learning; Mentoring; Statistical projects; Teamwork

1. Introduction

Our goal is to discuss the introductory data science course that we have designed at Purdue University for the NSF-funded Statistics Living Learning Community. Although the focus of this article is on the implementation of the course itself, we first want to give some *motivation* about why we are offering such a new course, during a student's sophomore undergraduate training. The second author recently gave a talk at the American Statistical Association's (ASA) Conference on Statistical Practice (CSP) in February 2017 about the design and implementation of such a course. Most importantly, he focused on the skills and tools that a student will have learned, as a result of taking such a cutting edge course. We will (jointly) give a related CSP workshop in February 2018. Our goal in this article is to disseminate these ideas more broadly.

"Data Science" has become a ubiquitous priority, not only in academia, but throughout the workforce. Along with the rise of data science, other closely related topics such as computational statistics and machine learning are receiving attention. We emphasize that there are many computational skills that students and professionals need, which are not necessarily in either the traditional realm of statistics or computer science. Some of these skills might be called "data wrangling." Introductory topics in data science can be quite broad. Only a few books have managed to simultaneously touch on the broad number of topics in data science; see, for instance, Baumer, Kaplan, and Horton (2017); Boehmke (2016); Murrell (2009, 2012); Nolan and Temple Lang (2014, 2015); Wickham and Grolemund (2016).

As Champkin stated, statisticians

> "should stake out and prove the claim that analysis of Big Data is a statistical skill. They should make sure that those skills are part of statistical training; should grab the people who have or who want to acquire those skills, and grab them young and, in the process of giving them those skills, give them also the belief that what they are becoming is statisticians and that what they are doing is statistics. It is a brave new world out there, and it is a very large chunk of the future." (Champkin 2012)

We would embellish this sentiment to add that not only statisticians, but rather, people working in all data-driven disciplines, should have familiarity with tools for data analysis.

We consistently emphasize datasets as the motivation for the tools that we introduce. Applications of data analysis and the resulting benefits to society are key motivators for our students. Having a data-driven learning environment was rare just one decade ago, but the need for data analysis training is now well recognized.

In a recent report (De Veaux et al. 2017), the ASA stated that there are 530 programs related to data science around the world. They are typically for a master's degree or a certificate program. We emphasize the need, however, to integrate data science at the earliest parts of the undergraduate curriculum. The ASA report lists key components of data science as "computational and statistical thinking, mathematical foundations, model building and assessment, algorithms and software foundation, data curation and knowledge transference" (De Veaux et al. 2017). In the following sections, we will try to explain our pioneering effort in data science and its implementation, harmonizing these key components.

1.1. Overview of Early Efforts to Foster Data Science at Purdue

At Purdue, we began our efforts to incorporate data science into the undergraduate curriculum by creating the NSF-funded Purdue Statistics Living Learning Community (STAT-LLC; see *http://llc.stat.purdue.edu*). More broadly, this year, Purdue's Department of Statistics is partnering with our Department of Computer Science to introduce a Data Science major.

CONTACT Fulya Gokalp Yavuz *fulyagokalp@gmail.com* Department of Statistics, Purdue University, West Lafayette, IN 47907, USA and Yildiz Technical University, 34349 Istanbul, Turkey.

We give some insights into how to start such an initiative, by offering an introductory, hands-on data science course that is driven by large, real data examples, and is offered to undergraduate students from different disciplines. We endeavor to explain our intuition and motivation when we create examples and projects. In the ASA report mentioned above (De Veaux et al. 2017), the practice and experiential aspects of data science are emphasized. Project-based groups and presentations are recommended. We believe that the STAT-LLC implements many of the best practices that are promulgated in the ASA report.

We are committed to training for students and professionals with the following attributes:

1. No prior background knowledge is needed for succeeding in the learning environment.
2. A strong emphasis is devoted to large datasets, rather than small examples.
3. The computational environment includes external server equipment.
4. The training enables and empowers learners to wrangle large datasets into desired forms.
5. Everything is "hands-on," that is, students are "learning by doing."
6. The learning environment is welcoming. Everyone is encouraged to work hard, but with the acknowledgment that many learners will make mistakes along the way.
7. The students and their projects both come from interdisciplinary areas.

A recent pair of articles (Leswing 2016; McBride 2016) claim that it is insufficient to go to a bootcamp or to graduate from a CS program. Mentees should actually do some hands-on data analysis work in their early career, as they do in the STAT-LLC.

We have worked with many mentees—at the undergraduate and graduate levels—who have been enthusiastic about such learning environments. James Marshall Reber, who was a sophomore undergraduate in our data science course, wrote on Facebook that "I feel like I've finally learned how to learn effectively." This is typical of the type of feedback that we frequently receive from students.

The students in our courses really enjoy the data-driven and application-driven style of our seminars. We do not provide PowerPoint slides. Instead, we provide videos and code for the students to use overnight (in between classes). Our projects allow for a balance of specific questions and objectives, as well as some open-ended data exploration.

The interactive environment that we offer for learning data science is somewhat unique. The students interact a great deal with their peers. One may justly wonder why one more type of learning environment is needed. There are lots of data science courses online. We believe that our students value the support of their peers, as well as the interactive feedback we are able to offer to students, when we interact with them one-on-one. The students support each other by working in teams. In addition, we often find that one team helps another, so that a true sense of community is established. We routinely offer data science courses to about 20 students at a time. In fall 2016, we organized a seminar with more than 60 students enrolled, and we used Piazza to help encourage discussions in between the meetings.

In the next section, we introduce the data science course with more details.

2. Introduction to Data Science Course

We assume no prerequisite knowledge, for example, we do not assume that students know any programming languages in advance. Of course, we recognize that some of them will have computational experience, but we do not rely on this.

The students take this data analysis course and also probability theory in tandem, during the fall of their sophomore year. During the second semester of their sophomore year, they take their first statistical theory course. During their junior and senior years, they have the option of taking data mining courses, applied statistics courses, computational statistics courses, etc. Data science courses—like the one we describe here—prepare the students for the ASA DataFest, Hackathons, emerging major programs of study in data science, applied statistics projects in other disciplines, etc.

The students in the STAT-LLC work in groups on all of the projects. The decision to have students work in groups throughout the course is not one to be taken lightly. It is worthwhile to think carefully about what style of group learning fits a course, and to be adaptable to the needs of students. Felder (2017) and his co-authors had written prolific, insightful material about students working in teams; see especially Haller et al. (2000); Oakley et al. (2004). If the teams are organized and supported appropriately, having a group-oriented environment enables everybody to come quickly up to a stronger level of computational expertise. Group work simultaneously improves students' communication, teamwork, and time management skills. At its best, group work enables students to be peer mentors who strongly support each other. At Purdue, we had extensive discussions with colleagues who have expertise in active learning, team-oriented environments. We would urge our colleagues to seek advice and guidance from experts as well, to use best practices related to teamwork. We are also in the fortunate situation that our students live together as roommates on the same floor of a co-ed residence hall, so they are already a tightly knit group. Moreover, our student cohort consisted of 70% females in 2016–2017, and 60% females in 2017–2018, so our female students do not feel isolated. The female/male ratio, and the ratio of minorities, both require careful attention. Felder discussed methodologies for appropriate group assignments that take such course design elements into consideration. Our students work in randomly assigned groups of three students each. Since we have 10 projects per semester, we chose a combinatorial design that enables every student to work with every other student one time during the semester. Our students consistently report that they enjoy this aspect of the course design, but we recognize that this is a rare situation, and it requires a great deal of thought and planning.

The developing data scientist needs to have the freedom to do exploratory data analysis, build some intuition, read about domain knowledge, learn new tools for data analysis, make some mistakes, have discussions with a wide variety of peers and mentors, etc. This all takes time to come to fruition, and it is best realized in an interactive learning environment.

When we discuss introductory data science topics, we intend to reach more than just instructors and early-career students. We also reaffirm the need for practitioners and researchers to know fundamental computational skills for data wrangling, such as bash shell scripting, awk, R, SQL, XML scraping and parsing, etc. With reproducible research in mind, we also work in RStudio with RMarkdown for the project output.

2.1. Using an External Server Environment

Large datasets are pervasive. They are found in atmospheric science, finance, genetics, healthcare, physics, etc. The European Organization for Nuclear Research (CERN), for instance, has generated untold petabytes of data. During a decade of organizing data science courses for undergraduate and graduate courses, we have noticed that students perceive the size of data as it progresses: from small data that can all be seen with one's own eyes, to larger data that might be opened in a spreadsheet but has too many rows and/or columns to easily see, to data that will fit onto a laptop computer but can only be summarized and viewed in plots to be seen, to even larger data that cannot fit onto a student's laptop or into portable memory devices, but must (instead) be stored in external server equipment and studied remotely.

Each level of complexity necessitates students becoming more mature with their data analysis skills. We provide Unix servers for the students' (remote) work. The server used in 2014–2016 featured 384 GB of RAM, 50 TB of disk, and 24 processing cores with hyperthreading. For the 2016–2017 term, we tested an expansion of this, to four more rack mounted servers with 512 GB of RAM each. We provide installations of R, RStudio, and many R libraries, as well as a MySQL server. (Some explanations about how to setup one's own virtual server can also be found online; see, e.g., Andrade and Golab 2016.) When the second author began offering data science courses back in 2009, each student brought her/his own laptop to class, but having such a variety of operating systems and R libraries proved to be a frequent roadblock for the class working in a unified way. Working in a homogenous environment on an external server provides unity and uniformity to the student experience.

Although server equipment is readily available on many campuses, we recognize that some colleagues and/or their students will want (or need) to build their own computational environment. For this purpose, we present some suggestions about free resources. Of course, R (*www.r-project.org*) and RStudio (*www.rstudio.com*) have free, prebuilt binaries (i.e., nothing needed to compile) that are easy to install. Similarly, SQLite (*www.sqlite.org*) has free, prebuilt binaries, in case an SQL server is not provided on one's campus. The bash shell is available in every Macintosh and UNIX machine without cost; for Windows users, starting with Windows 10, it has recently become possible to easily install a free bash shell. All of these resources help to further reduce the differences in operating systems, and to make computational environments broadly accessible. We have chosen to use a prebuilt environment at Purdue, for uniformity across the students' experiences, and because most alumni have told us that environments are provided for them on their job sites. Nonetheless, we hope that these resources are useful for those who want to build environments themselves, in the easiest way possible, at no cost.

3. Projects

In this part of the article, we give some examples from the projects that we have used to train mentees in the Data Analysis course. Each project takes approximately 2 weeks to complete. The students have different approaches to the same problem; they share their previous knowledge by working in groups. They do not just learn how to use the required tools, they also learn how to work within a group. The students are able to discuss freely, without critique or pressure; this increases the students' creativity. As mentors, we keep in mind that the Data Science topics and problems require some subjective interpretations.

We rely on open source tools throughout the course, so it is not necessary for students to buy software licenses. This enables students to be able to continue using the tools after the course is over, and even in the workforce, after their college experience is over.

Also, we advocate giving them freedom about the method by which they learn the material. We prepare videos in advance, to make sure they know basic concepts, such as downloading a tool onto their laptop/desktop or onto the server. We give them comprehensive guidance about how to connect to the server, so that they are empowered to use the external environment from the very beginning.

3.1. Unix Projects

For the first 2 weeks, we want mentees to be familiar with Unix. We try to keep them away from the ready-to-use programs (e.g., Bioconductor), because we prefer that students understanding some underlying fundamental skills. During the first 2 years of the STAT-LLC, the mentees learned Unix in the middle of the semester, but this year, they started directly with Unix from week 1. The students responded well to this change. They learned some basic computer fundamentals (like working with the bash shell) directly from the beginning. This increased their comfort level with the computational environment.

More generally, we observed that by starting with the fundamentals of each computational tool—and building the knowledge from zero—we enable the students to more deeply understand the computational concepts and logic. Also, they benefit from knowing the fundamental principles behind each tool that they are using (the ideas behind the tool).

Instead of introducing a tool without motivation, we consistently start with datasets and emphasize the ways that a tool can be used for wrangling datasets. This gives students the ability and freedom to have more tools at their disposal while they are performing exploratory data analysis.

On the first day of the semester, we introduce the STAT-LLC mentees to a project using the dataset from the Data Expo (2009). The data was downloaded onto the server in advance, as "csv" files corresponding to each year from 1987 to 2008, with 123,534,970 flights and 29 variables: 3,582,514,130 entries altogether. All mentees have access to same readable, formatted data. This dataset has already been used in many courses nationwide; see, for instance, Nolan and Temple Lang (2015, chap. 5). Rather

than importing this data into R, we encourage students to work with fundamental, low-level Unix tools during their first analysis of this data.

Since many students are not familiar with Unix at the start of the course, we encourage students to read Peek, Todino, and Strang (2002, chap. 1, 3, 4, 5, and 7) and also Robbins and Beebe (2005, chap. 1–5).

In particular, the students find it helpful to learn how to apply the pipe "|" for redirecting the output of one Unix utility to become the input of another Unix utility. Very few of the students knew about the concept of the pipe before joining our class.

We mainly show them how to use some basic Unix bash shell commands such as "cut," "cat," "sort," "grep," "wc," In particular, we showed them how to daisy-chain commands together. For instance, we show them how to cut one column of data out of a comma-delimited file, and pipe this column of output to the sort function, and then pipe the results to the uniq function, and finally pipe those results to the wc function.

Used in tandem, these fundamental shell commands become much more useful for data analysis. We usually use the Data Expo (2009) airline data to exemplify this methodology.

For instance, the comma separated file of airline data can be examined by extracting the airport code (17th column). Then we can sort the resulting codes, filter them to get one of each (using uniq), and finally obtain the number of codes altogether (using wc). Here is the bash code to accomplish this:

```
cut -d, -f17 FileName.csv | grep -v "ColumnName" | sort | uniq | wc -l
```

In another project, we want the students to use the individual campaign contributions data reported by the United States Federal Election Commission (*http://www.fec.gov/finance/disclosure/ftpdet.shtml*). This dataset motivates our need to do pattern matching using the awk tool in Unix. Although we recommend that the students read a few portions of Dougherty and Robbins (1997) and Robbins (1997) before working on the project, we find that (in practice) the students understand the awk tool with relatively little difficulty. Eventually the students discover that awk provides a very convenient (and easy) means of systematically processing files.

We also include awk variables, which allow students to do things like adding the elements in a column, or summing one column according to the value of the elements in another column. The students gradually begin to understand logical expressions with this learning style.

Students are familiar with "csv" files (comma separated values). Most of the students had never thought about the idea of having other kinds of delimiters between records in a structured data file. This concept is natural to introduce during the discussion about awk. The awk tool is versatile for handling structured data with many kinds of delimiters.

More generally, the Presidential dataset was of great interest to the students, since the campaigns took place during the fall 2016 semester. We emphasize that the choice/usage of timely, interesting, relevant data for students is crucial part of the course design.

Going back to the airport data, mentees can be asked to answer some key questions such as "How many distinct origins? Destinations?" "How many flights from a certain airport?" "Erroneous tailnums?" "Which airplanes took the most flights overall?" "Which airplane took the most flights in a single day?" "Which cities are hubs for which airlines?"

At the end of this project, they are able to dissect, sort, summarize, trim, cut, and aggregate the data.

One goal in returning to a dataset that was previously studied during the course is to enable students to realize that they are going deeper into their mastery of each data science tool, as well as learning a broader suite of tools that can be applied. This helps their self-efficacy as a learner. It also enables them to learn more about a dataset, as the semester progresses.

One or more questions can be dedicated to finding and managing missing values in the data. Missing data should be handled in advance of any inferences. At least, mentees should be aware of the existence of missing values and their effects on data analyses.

After the students master the basics of awk, we subtly introduce the concept of vectors as well. In fall 2016, for example, our students used vectorized methods to simultaneously analyze campaign contributions from many different donors in the recent Presidential campaign. This gives some clarity to the concept of vectors, because students learn about vectors *as they need them*.

For instance, students identified the campaign that received the greatest amount of donations altogether. This is perhaps not suitable for the first project in awk. Students enjoy learning these techniques around the same time that they learn the comparable idea of the "tapply" function in R.

3.2. Transitioning Projects to R

After some work on Unix, we motivate the mentees to use R for some projects. As the students begin to learn R, we try to keep them away from some concepts which extend the computational time, such as "for loops." The issue only arises for students who already know how to program in another language (e.g., C, Java, Python), and for students who rely often on internet searches (e.g., stackexchange, stackoverflow, wikipedia) for solutions. We try to prepare them for both functional and efficient programming.

We show the students some examples that let them compare the necessary runtime used in a for loop, as compared to an "apply" function.

This year, as mentioned above, we used the U.S. F.E.C. election data for two different projects, but instead of working with the data in awk, the second time we used R. Mentees benefit from acquiring another way of looking into a dataset that they have already seen.

The election data for fall 2016 was 2.2 GB in size, with 12,395,164 donations and 21 variables—260,298,444 entries altogether. After reading in the data, we encourage students to use the functions such as class, head, and tail, to see some of the rows of the data. It is not possible (and not necessary) for them to see the whole data at once.

To summarize the data, we discuss the usage of the "tapply" function, for example, for each candidate, how to find the average amount of the candidate's contributions. We also emphasize errors in the data, for example, misspelled names of candidates, counties, etc. Sometimes the donation sizes are negative (e.g., if a donation is returned/rejected or erroneous) or are encoded incorrectly (outliers).

We also mention and demonstrate the "sapply" and "mapply" functions very early in the students' training as well.

Since the datasets we use throughout the training are large, we incorporate best practices in visualization as early as possible. We provide examples from Cleveland (1993, 1994); Emerson and Tactical Technology Collective (2008); O'Conner and Tactical Technology Collective (2008); Tufte (2001); Wainer (1984). To encourage the students to be critical and thoughtful about visualizations, we devote time to examining the visualizations of others. In particular, our students discuss the positive and negative aspects of the visualization choices takes by the authors of posters in the ASA Data Expo.

The analysis of the airline data from the ASA Data Expo 2009, for instance, can provide one excellent running theme throughout the semester, because many students have familiarity with air travel. This dataset is often one of the first ones we study during the semester (it lends itself well to analysis with bash shell and awk). A different approach (which we have tried and enjoyed too) is to ask the students to critique visualizations from newspapers and from social media.

Our goal throughout the visualization project is to understand best practices. We have critical discussions about the visualizations created by the participants in the ASA Data Expo, to see what best practices were used, as outlined in Cleveland (1993, 1994) and Tufte (2001). We find that students are familiar with the need to revise their writing, but they have not considered the need to revise their visualizations.

In creating a new course on data science, some colleagues might choose to cover the bulk of the techniques from Murrell (2012), which covers traditional graphics, trellis/lattice, ggplot2, grid, maps, 3D graphics, and interactive graphics, with illuminating examples throughout. Indeed, an entire course can be constructed around the many methods of visualizing data in R, and if desired, the "grammar of graphics" can be discussed in-depth.

Since our course emphasizes data science rather than data visualization, we have chosen to introduce students to as many tools for data technologies as possible, roughly following the topics in Murrell (2009), as well as chap. 1–4 of Murrell (2012). Although we only scratch the surface of R's many methods of visualization in our course, this is only due to limited time. Many of the students who took our course in the past have reported that they learned ggplot2 (and, more broadly, other tools by Hadley Wickham in the tidyverse family) during or after our course, according to their research needs and interests. This is characteristic of our entire philosophy about data science: There are many topics to introduce. Each instructor must ultimately pick-and-choose, but it is always *key* to encourage students to continue their learning throughout their studies and to maintain that thirst for new knowledge in the workforce as well.

We encourage students to integrate some of their previous code into RMarkdown. The use of RMarkdown promotes reproducible research and coursework (Gandrud 2016; Xie 2016). When students write in RMarkdown, they can generate pdf, html, or docx (Microsoft Word) documents. RMarkdown is free and easy to use. Gandrud (2016) gave details about the installation.

3.3. Extracting Data from a Database

For one of the projects, we work with data in "netcdf" format, from the database of the NSF Center for Coastal Margin Observation & Prediction. This data is publicly available on the SATURN Observation Network (Saturn 2016).

A key thing to note about data in "netcdf" format is that—unlike with "csv" or plain text files—we do not see the data itself, before we extract the data in R. It is an array-oriented, program-independent data format. The "ncdf4" package is useful to extract variables from these datasets in R. This is one extra level of complexity for the mentees.

One merit of this dataset is that the students need to consider issues regarding timestamps, including time zones. (The students have not previously studied any topics in time series.) Another aspect of the project is that the students get a more tangible understanding of the "mapply" function, as applied to data concerning temperature, salinity, and electrical conductivity as explanatory variables. We consider these variables from a certain station (SATURN03) on the Columbia River Estuary, measured at several distinct depths within the river.

First, we create a vector corresponding to the 84 months from November 2009 through October 2016, and create a second vector containing the corresponding years. Then we use a vectorized approach, with the "mapply" function, to easily obtain all 84 months of data about the water temperature, salinity, and electrical conductivity at the SATURN03 station at the depth 2.4 m. The result is a list that contains 84 data.frames. By this point in the course, the students really appreciate the power of vectorized functions and of automating processes that would (otherwise) have been tedious or repetitive.

A word of caution: In general, with "netcdf" files, we have no guarantee (a priori) that the variables from the "netcdf" files are all stored in the same order. So it is a best practice to use the "sapply" function, to verify that all 84 data.frames indeed have the variable names in the same order. (From experience, we have seen that this is not always the case, and it can cause frustration for novices.)

Finally, we use a do.call function to rbind these 84 data.frames into one data.frame. This is the first time that the students see this function, which allows us to apply a function over a dataset; this is another conceptual understanding for the students. Afterward, our students take pleasure in seeing that they have assembled a resulting data.frame (which we refer to as bigDF in the code below), which has more than 7 million observations.

Here are some example questions that we ask the mentees, once they have assembled the data. For instance, restricting attention to the 2.4m data, what is the longest time period for

which no data is available, that is, what is the longest time period in which no data is collected?

```
tail(sort(diff(bigDF$time)), 1)
```

On which day does that biggest gap occur?

```
bigDF[which.max(diff(bigDF$time)), c("year","month","days")]
```

We also point out that visualization can be a key tool for the students, especially for detecting missing values, etc. In our experience over the years, this dataset is one of the first times that students encounter data with significant periods of missing data. It is always somewhat of a shock for them to see this, in data drawn from the real world.

3.4. Extracting Data from a Website

The Hot 100 chart is posted every Saturday by Billboard. (Perhaps needless to say, we only use this data for academic purposes, not for any commercial purpose.) The first chart in the Hot 100 is located at *http://www.billboard.com/charts/hot-100/1958-08-09*.

We scrape more than 3000 such webpages from Billboard. Scraping such data from a website can be automated, by using the system command in tandem with either the wget or curl command. We show the students how to make such system calls from inside the R environment, with the result that the students can scrape all of these charts (in XML format) into a folder.

One of our aims for the mentees to focus on is methods for scraping (i.e., downloading data automatically) from an online source. It is important to find example from the web, including some types of consecutive records. Students will have already experienced the impact of XML in their lives: they have browsed HTML pages on the internet; they have worked with Microsoft Office files; and they have familiarity with iTunes libraries. All of these examples are stored using XML markup. Thus, we use XML for the students' first experience with data formats. We do discuss JSON and NCDF formats in our course too, but the XML discussion gets the bulk of our attention. Murrell (2009) devoted several chapters to the key role of XML in an introductory data science course. For some courses and applications, we readily emphasize that JSON might be the more appropriate data format to introduce at the start.

```
mydate <- seq(as.Date("1958-08-09"), as.Date("2016-10-08"), by = "week")
sapply( paste("curl http://www.billboard.com/charts/hot-100/", mydate, ">
    FolderLocation/chart", mydate, ".html", sep=""), system)
```

Calling the "sapply" function creates 3044 html files, which are roughly 400 MB in total.

The XML data from Billboard charts that were scraped from the internet can be parsed and studied by the mentees, using XPath calls from R. XPath is a specification that enables us to extract portions of an XML document that satisfy certain specific patterns. More information is found at *http://commons.oreilly.com/wiki/index.php/XPath_and_XPointer*. Also, Nolan and Temple Lang (2014) gave numerous, in-depth examples for how to use XML-based methods of scraping and parsing data.

As a final note about the Billboard project, we point out that a very small number of weeks have slightly incomplete data, that is, do not have 100 songs per week. This is a subtle point that can be a conceptual stumbling block, but it is a worthwhile topic of discussion.

We can use similar automated methods to download (in parallel) data from another online source, concerning yellow taxi cabs in New York (New York City Taxi and Limousine Commission 2016). A data dictionary is provided at this site. The mentees automate the process of downloading the data into the files in the server using either the system command (called from R) or directly from the bash shell. The data for the yellow taxi cabs is 94 GB altogether. There are 1,249,152,441 trips and 19 variables—23,733,896,379 entries—in this dataset, which is too large to load into R. There is a "csv" file for each month from 2009 to 2016. We use bash tools in R to download these files from the website to our server. Please see the related code below.

The reason for using the system command is to be able to automate the following method of transferring all files from the website:

```
system("curl https://s3.amazonaws.com/nyc-tlc/trip+data/yellow_tripdata_
  2016-01.csv > FileLocation/taxi2016-01.csv")

```

```
mymonths <- c(rep(sprintf("%02d", 1:12), times=7), sprintf("%02d", 1:6))
myyears <- c(rep(2009:2015, each=12), rep(2016, each=6))
sapply( paste("nohup curl https://s3.amazonaws.com/nyc-tlc/trip+data/
   yellow_tripdata_", myyears, "-", mymonths, ".csv > FileLocation/taxi",
    myyears, "-", mymonths, ".csv &", sep=""), system)
```

The nohup command helps ensures that a command keeps running after a user logs out. This may be necessary in this example, since the transaction process will take time (perhaps an hour, depending on the internet speed).

After downloading the data onto the local server, several questions can be asked. For example, on which day did the most taxi cab rides occur? (If a ride goes past midnight, use the start of the ride for the date of the ride.)

There are (at least) two different methods to answer this question. The first one is to merge all the data files into one common file, then sort this data, and find the day in which most rides occur. The second method is to sort each file and save the days within each file, in which the most rides occur, and then aggregate the results from the individual files into one (at the end). Of course, the second methodology is faster, and can be executed in parallel. Sorting a big file is much slower work than sorting small files, saving the individual results, and then sorting the accumulated results. We encourage students to try multiple methods of solution, and to contrast the pros and cons of each approach, for example, with regard to speed, efficiency, ease-of-coding, ease-of-understanding, etc. Here are two possible methods of solution:

First method: use the raw data

```
system("cat *.csv > alltaxi.csv")    #Put all data into a file:
system("awk -F'[, ]' '{print $2}' FileLocation/alltaxi.csv | sort -T /
   scratchdirectory |uniq -c | sort")
```

Second method: use "sapply" and sort for each file:

```
sapply(paste("awk -F'[, ]' '{print $2}' FileLocation/taxi",myyears,"-",
   mymonths,".csv | sort | uniq -c | sort | tail -1 >> FileLocation/
   taxisummaries.csv", sep = ""),system)
```

The answer is found by sorting the file taxisummaries.csv, and taking the biggest entry, to find the most taxi cab rides:

```
system("sort FileLocation/taxisummaries.csv | tail -1")
```

Another example question is to determine the average distance of a taxi cab ride for each day. Similar to the previous example, this can be done with two different methods. The first one is using all data, and the second one is using each data file separately.

First method: use the raw data:

```
system("awk -F'[, ]' '{col[$2]++; count[$2]+=$6} END {for (var in col)
   print var, count[var]/col[var]}' /FileLocation/alltaxi.csv")
```

Second method: use "sapply" and take the averages for each file separately:

```
sapply(paste("awk -F'[, ]' '{col[$2]++; count[$2]+=$6} END {for (var in
   col) print var, count[var]/col[var]}' FileLocation/taxi",myyears,"-",
   mymonths,".csv > FileLocation/average.pass.taxi",myyears,"-",mymonths,
   ".csv", sep=""), system)
```

Now, we can see the average values for January 2016 as an example.

```
system("cat FileLocation/average.pass.taxi2016-01.csv | sort")
```

Also, this dataset has the potential to be used for time series and spatial data analysis, since the latitude and longitude data are included for the starting and ending locations of each ride. This could (potentially) yield a way for the students to have some projects which span multiple courses.

3.5. SQL

Another project covers SQL and related topics for accessing databases. Beaulieu (2009) is a user-friendly introduction source for this part. Before the project begins, we establish accounts on a MySQL server for the students. Alternative methods would include asking students to download a free MySQL server from *http://www.mysql.com*, or SQLite, etc.

Mentees can download the examples of the book (Beaulieu 2009) using (for instance) wget or scp in the bash shell.

The Lahman database (Lahman 2015) is used for SQL examples about baseball. The examples from the baseball database can be downloaded from *seanlahman.com*.

Before starting the SQL examples, there is one thing that mentees might want to change in the Lahman database. Some fields are initially *text* and they should be changed to *varchar(20)* and be indexed, so that the analogous SQL queries run much faster. For example, to change the playerID's in the batting table:

```
alter table batting modify playerID varchar(20);
alter table batting add index batting_playerID(playerID);
```

For instance, mentees may want to discover the player ID's, years, and teams in which players have hit more than 60 home runs. This can be elaborated as much as desired by the mentees. For instance, they might expand the queries to include the full names of the teams, by joining with the teams table.

Students also usually enjoy exploring their hometown's team. For instance, many students enjoyed seeing how many players were employed by the Chicago Cubs during each year since 1960. (Purdue is relatively near Chicago.) As another example, students can discover the teams that won at least 105 games in a year. We encourage our colleagues who never performed SQL queries to give such projects a chance. The coding is succinct and self-explanatory. For instance:

```
SELECT yearID, teamID, SUM(W) FROM teams GROUP BY teamID, yearID HAVING
   SUM(W)>= 105;
```

3.6. Final Project

For the final project, students again work in teams. We have several guidelines for the students, namely: They should pick a topic that all members of the group find interesting. The data that they analyze should have at least 2 million data points. Moreover, we encourage them to find data with a rich structure, rather than (say) only 2 million data points occurring in the same time series. The data should be rich in structure, for example, lots of variables. It is desirable to have data from more than one source, so that the students need to combine the data. We strongly encourage the students to "scrape" the data from websites, rather than just use large "csv" files that they find on the internet. We also encourage the students to use best practices (learned during the semester) about data visualization. The students should aim to give insight into the data that they analyzed. For comparison, the result of the final project should be suitable for a data-driven competition, for example, for the ASA Data Expo. Our students have a great deal of freedom in designing the project, from the beginning to the end.

The students always pick final projects according to their diverse interests. This year, for instance, one team worked on data concerning consumer complaints related to debt collection, money transfers, mortgages, etc., using data from the Consumer Financial Protection Bureau. Three teams this year analyzed crime data, one from Los Angeles, and the other two from Chicago. Crime statistics seems to be a popular topic over the years, and indeed, Los Angeles crime data were used in the first DataFest competition, back in 2011. The ASA DataFest is a friendly competition, in which groups of students work on a big and complex data *http://ww2.amstat.org/education/datafest/*.

Almost every year, one team will use the Internet Movie DataBase (*www.imdb.com*) for their final project. The students enjoy asking questions about actors, popularity of movies, various genres, etc., and then using their knowledge of data analysis to answer their own questions. Another team scraped data from a website dedicated to the weather and then studied long-term trends related to global warming. One team used an extensive amount of XPath code to scrape information from StackOverflow.com, to extract votes, answers, and number of views, and they also were cognizant of the authors (although many responses are anonymous).

We emphasize that, at this level of experience, we are primarily interested in the students' ability to scrape data and then wrangle it into a shape that is suitable for analysis. They have not (yet) at this point in their training had a significant focus on statistical analysis; that comes afterward in the statistics major. We are laying a solid foundation for them to work with large datasets, and are preparing them to do more advanced statistical analysis.

4. Student Reactions

The students tend to have a positive reaction to our STAT 29000 data analysis course. As we wrote this article, we also asked all of the students on the grant if they were interested to send feedback. Six of them quickly responded, and we give all of their feedback here, without filtering. Among these six students, one is a Chemical Engineering student, one is in the College of Pharmacy, one in a dual Computer Science/Statistics major, two are dual Mathematics/Statistics majors, and one is a triple Economics/Mathematics/Statistics major.

"The STAT 290 data analysis course was the most realistic and applicable course I have taken at Purdue. However, what I loved most about the class was that we genuinely had fun learning multiple languages and analyzing big data! Dr. Ward would choose interesting data sets, he would have us working in teams, and he would guide us in the flipped classroom toward bringing life to data."

"I am a pharmacy student who had no statistical background before this. Stat 290 gave me the knowledge I needed to do amazing statistical cancer research, the opportunity of which I would not have received had I not been a part of this learning community."

"STAT 29000 sparked my interest in working in data analytics with large data sets. The Living Learning Community as a whole promoted my interest in the research opportunities available to undergraduates and solidified my choice of picking up an Applied Statistics degree."

"STAT 290 is a unique class that has challenged me to think critically about Data Analysis. It gave the tools and skills to be a valuable asset at the lab I work in, and feel savvy as a statistician. The statistics LLC was a formative, and fun, experience that opened my eyes to the world of data analysis and research. The opportunity to work on undergraduate research, and immerse myself in a community of driven students, has been more valuable than any one class!"

"Our STAT 29000 class initialized my desire to try to learn as much as I can about using R. Working in groups in a low-pressure situation really facilitated the learning experience, and I feel prepared for any research project or internship I have in the future."

"The Purdue Stat LLC was the best way to figure out if statistics is actually something I wanted to do. Rather than just talking about what you can do with statistics, you're thrown in the deep end with a project driven class and actual real world research, and it's probably why I stuck with math and stat rather than transferring over to industrial engineering."

Even if the students were not Statistics majors before taking this data analysis course, it has turned out to be a great recruiting tool. Many of them decide to add a major in Statistics and/or to choose a career working as a data scientist. Usually about 13 out of the 20 students change their major or add a second major or minor. Oftentimes they add an applied major, for example, they choose to add a major such as Biology or Health and Human Sciences.

5. Conclusion

We reiterate our belief that the Purdue Statistics Living Learning Community implements many of the best practices espoused by the ASA (De Veaux et al. 2017), including interaction with data, and a strong focus on teamwork. These two components are cornerstones of the STAT-LLC and are woven throughout the data science curriculum we have implemented.

Students almost always adapt to data science concepts more quickly and easily than we anticipate. They are eager to learn and they recognize the significance of new, innovative, data-driven learning environments. They come up with new ideas faster than we expect. They learn in different ways than previous generations of statisticians. Building a solid foundation in data science gives students more intuition for later courses in statistics and data analysis. They benefit greatly from being able to apply concepts as they work together in teams.

It is feasible to implement these initiatives in undergraduate statistics programs. We have described many data analysis tools that we manage to pack into one semester, but all of the tools are freely available. Computational servers are becoming more and more affordable for departments and colleges to purchase. A key take-away of our message is that it is possible for other departments and programs to have these course offerings too. Such courses, in turn, naturally lead to expanded opportunities for undergraduate research.

The second author is willing to be a liaison to departments and programs who are interested in offering intensive, early-career experiences in data science for their students. He is routinely giving workshops on these topics and is delighted to visit and support colleagues who want to make a push into these cutting-edge opportunities for undergraduate student training.

Acknowledgments

The second author heartily thanks Deborah Nolan and Duncan Temple Lang for offering workshops at UC-Berkeley which inspired him to apply for NSF funding for the STAT-LLC. Many data science mentors (including ourselves) have greatly benefited from the numerous resources that they have created and freely shared with the data science world. The authors are sincerely thankful for their continued leadership in the world of data science training and mentoring.

At Purdue, the authors are thankful to dozens of colleagues for their support throughout their grant. The authors especially thank Dave LeFevre (Unix Systems Administrator) and Doug Crabill (Senior Academic Site Specialist) for their guidance on all computational issues. They have provided tireless support for them and for their students.

The authors are also indebted to the editors and referees for their insightful suggestions and improvements.

Funding

This article is supported by the Scientific and Technological Research Council of Turkey under Tubitak-2219. This material is based upon work supported by the National Science Foundation under Grant Numbers 0939370, 1246818, and 1560332.

References

Andrade, A., and Golab, L. (2016), "Beginners Tutorial for How to Get Started Doing Data Science Using Servers," available at *http://datascienceguide.github.io/beginner-tutorial-how-to-get-started-with-data-science-using-servers* [10]

Baumer, B. S., Kaplan, D. T., and Horton, N. J. (2017), *Modern Data Science with R*, Boca Raton, FL: Chapman and Hall/CRC. [8]

Beaulieu, A. (2009), *Learning SQL* (2nd ed.), Sebastopol, CA: O'Reilly. [14]

Boehmke, B. C. (2016), *Data Wrangling with R*, New York: Springer. [8]

Champkin, J. (ed.) (2012), *Significance, the magazine of the American Statistical Association and Royal Statistical Society*, Special issue, dedicated to Big Data. [8]

Cleveland, W. S. (1993), *Visualizing Data*, Murray Hill, NJ: Hobart Press. [12]

——— (1994), *The Elements of Graphing Data* (revised ed.), Murray Hill, NJ: Hobart Press. [12]

Data Expo (2009), "Airline On-Time Performance," available at *http://stat-computing.org/dataexpo/2009/* [10,11]

De Veaux, R. D., Agarwal, M., Averett, M., Baumer, B. S., Bray, A., Bressoud, T. C., Bryant, L., Cheng, L. Z., Francis, A., Gould, R., Kim, A. Y., Kretchmar, M., Lu, Q., Moskol, A., Nolan, D., Pelayo, R., Raleigh, S., Sethi, R. J., Sondjaja, M., Tiruviluamala, N., Uhlig, P. X., Washington, T. M., Wesley, C. L., White, D., and Ye, P. (2017), "Curriculum Guidelines for Undergraduate Programs in Data Science," *Annual Review of Statistics and Its Application*, 4, 15–30. [8,9,15]

Dougherty, D., and Robbins, A. (1997), *sed&awk* (2nd ed.), Sebastopol, CA: O'Reilly. [11]

Emerson, J., and Tactical Technology Collective. (2008), *Visualizing Information for Advocacy: An Introduction to Information Design*, India. [12]

Felder, R. (2017), "Resources in Science and Engineering Education," available at *http://www4.ncsu.edu/unity/lockers/users/f/felder/public/RMF.html* [9]

Gandrud, C. (2016), *Reproducible Research with R and R Studio* (2nd ed.), Boca Raton, FL: CRC Press. [12]

Haller, C. R., Gallagher, V. J., Weldon, T. L., and Felder, R. M. (2000), "Dynamics of Peer Education in Cooperative Learning Workgroups," *Journal of Engineering Education*, 89, 285–293. [9]

Lahman, S. (2015), "Baseball Database," available at *http://www.seanlahman.com/baseball-archive/statistics/* [14]

Leswing, K. (2016), "Google Says Coding School Graduates 'not Quite Prepared' to Work at Google," available at *http://www.businessinsider.com/google-says-coding-bootcamp-graduates-need-additional-training-2016-12* [9]

McBride, S. (2016), "Want A Job in Silicon Valley? Keep Away from Coding Schools," available at *https://www.bloomberg.com/news/features/2016-12-06/want-a-job-in-silicon-valley-keep-away-from-coding-schools* [9]

Murrell, P. (2009), *Introduction to Data Technologies*, Boca Raton, FL: CRC Press. [8,12,13]

——— (2012), *R Graphics* (2nd ed.), Boca Raton, FL: CRC Press. [8,12]

New York City Taxi & Limousine Commission (2016), "Trip Record Data," available at *http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml*; the data dictionary is located at *http://www.nyc.gov/html/tlc/downloads/pdf/data_dictionary_trip_records_yellow.pdf*. [13]

Nolan, D., and Temple Lang, D. (2014), *XML and Web Technologies for Data Sciences with R*, New York: Springer. [8,13]

——— (2015), *Data Science in R: A Case Studies Approach to Computational Reasoning and Problem Solving*, Boca Raton, FL: CRC Press. [8,10]

Oakley, B., Felder, R. M., Brent, R., and Elhajj, I. (2004), "Turing Student Groups into Effective Teams," *Journal of Student Centered Learning*, 2, 9–34. [9]

O'Conner, S., and Tactical Technology Collective (2008), *Maps for Advocacy: An Introduction to Geographical Mapping Techniques*, India. [12]

Peek, J., Todino, G., and Strang, J. (2002), *Learning the Unix Operating System* (5th ed.), Sebastopol, CA: O'Reilly. [11]

Robbins, A. (1997), *Effective awk Programming* (4th ed.), O'Reilly. available at *http://www.gnu.org/software/gawk/manual/*. [11]

Robbins, A., and Beebe, N. (2005), *Classic Shell Scripting*, Sebastopol, CA: O'Reilly. [11]

Saturn (2016), "SATURN Observation Network: Endurance Stations," available at *http://www.stccmop.org/datamart/observation_network*. [12]

Tufte, E. R. (2001), *Visual Display of Quantitative Information* (2nd ed.), Cheshire, CT: Graphics Press. [12]

Wainer, H. (1984), "How to Display Data Badly," *American Statistician*, 38, 137–147. [12]

Wickham, H., and Grolemund, G. (2016), *R for Data Science*, Sebastopol, CA: O'Reilly. [8]

Xie, Y. (2016), *Dynamic Documents with R and knitr* (2nd ed.), Boca Raton, FL: CRC Press. [12]