Presented at the 2019 ICEAA Professional Development & Training Workshop - www.iceaaonline.com

When Data Isn't Enough

Kellie Scarbrough
Cobec Consulting, Inc

May 2019

Abstract

How can an analyst transform messy data into valuable insights, and why should they bother? This paper argues the importance of expanding the analyst skill set into the broader realm of data science and outlines several techniques to clean and process large volumes of data. Using R, we demonstrate how to look for important relationships among our data and explore some text mining techniques to utilize an otherwise unhelpful text field. Finally, we explore best practices in visualizing and communicating our findings.

This paper discusses the notion of "bad data" and argues that analysts should not only reconsider using previously ignored data, but that they *must* learn new ways of analyzing such data if they are to adapt to the "big data" revolution. Several examples explore some basic techniques that analysts can use to glean new information from any size data set. Finally, we briefly discuss how to visualize, interpret, and communicate our findings.

Bad Data

Every data analyst has heard the adage, "Garbage in, garbage out." Often it seems to offer a convenient excuse for lowering expectations when working with a particularly messy spreadsheet or database. In government, analysts must frequently utilize disparate and inconsistent data sources. But what really makes data "bad"? Surely, data which is inaccurate will provide inaccurate results. A review of consumer-related data found that in their eagerness to collect and harvest the power of big data, companies were using information that was outdated, inaccurate, and inconsistent[1]. Government data frequently suffers a similar fate. The Office of the Inspector General found that only a quarter of federal agencies could accurately report their expenditures[2]. Agencies have focused on the technical ability to collect big data while failing to ensure its accuracy. But should we dismiss such data outright? At a higher level, government agencies could improve their data by strengthening their data governance policies. At the analyst level, semi-accurate big data still has predictive power if used correctly[3]. Analysts can inspect data sets against similar sources of information, perform quality checks, and work around the limitations of their data (once they understand them). Furthermore, sometimes all it takes to salvage "bad" data are simple techniques such as:

- normalizing values,
- removing duplicates,
- correcting blank and inaccurate values,
- correcting spelling errors,
- standardizing values, and
- transforming data.

While these things may take time, they also unlock the potential of otherwise overlooked data. The analyst's role in making data useful will be the focus of our discussion here. We assume the basic corrections listed above are already well within the analyst's skill set, and instead focus our attention on skills previously relegated to data scientists.

Going Rogue

In this era of exponentially increasing data volume, analysts find themselves with increasingly unwieldy starting points for analysis. Data comes in faster than ever, and it is of greater variety. So where does the analyst start, and how do they filter through the noise to get the substance they need? Analysts can no longer manipulate such large volumes of data manually. Instead, analysts now find themselves employing a variety of techniques

[1] John Lucker, Susan Hogan, and Trevor Bischoff. "Predictably inaccurate: the perils and struggles of bad big data". In: *Deloitte Review* 21 (July 2017), pp. 8–25.

[2] Greg Brown. "Just how accurate is our data?" In: *The Fair Observer* (May 2018).

[3] Lucker, Hogan, and Bischoff, "Predictably inaccurate: the perils and struggles of bad big data".

to extract and transform data. They may employ SQL, R, or Python; they may even find themselves creating a data warehouse. To answer complex questions, analysts must sometimes move beyond descriptive analytics and into predictive and prescriptive analytics. They may create regressions, use clustering algorithms, or employ machine learning.

Even on government programs, the skills of analysts have expanded as program data becomes more readily available. Program managers seek to better track the execution of their programs, solve increasingly more difficult problems, and visualize program status in new and innovative ways. Budgetary pressures require cost estimators to find better data and establish more sound bases for their estimates. It is no longer enough to tell your agency how much money you need; in today's political environment, you have to *prove* it. Provost describes the transformation of private business and data as two stages: Data 1.0 and Data 2.0. In these scenarios, Data 1.0 refers to the original adoption of internet technologies and electronic commerce wherein businesses established their online presence. In Data 2.0, businesses establish the capability to process increasing volumes of data and shift toward data-analytic thinking[4].

Government Data 1.0

I propose that the government version of Data 1.0 has been the adoption of systems which streamline data collection and reporting and which establish access to data within their respective organizations. These systems include intra-agency systems as well as inter-agency systems. Examples include:

- Central Accounting Reporting System (CARS), Bureau of the Fiscal Service
- Governmentwide Financial Report System (GFRS) , Department of the Treasury
- Federal Procurement Data System, FPDS
- Acquisition Gateway, GSA
- Delphi, FAA
- Total Force Retention System, US Marines

This phase continues still as agencies constantly build and implement new solutions to centrally capture and store their data.

Government Data 2.0

The government version of Data 2.0, then, would be agencies using data analytics to improve the estimating, budgeting, and execution of their programs. In 2016, the federal government issued a report exploring ways to support big data initiatives within agencies to help them better achieve their missions. The report acknowledged the challenges of big data, especially as they pertain to the federal government, and sought to increase research and development to bolster the infrastructure and security needed to bring big data to agencies[5]. Analysts will play a key role in incorporating this new data-driven mindset into the core functions of the program office. The more expertise analysts have in the larger realm of big data, the more they will contribute to the success of their agency's mission.

[4] Foster Provost and Tom Fawcett. *Data science for business: what you need to know about data mining and analytic thinking.* O'Reilly, 2013.

[5] *The federal big data reseach and development strategic plan.* Tech. rep. NITRD, May 2016.

Murky Waters

Given the evolution of how we collect and use data, the role of data analyst has expanded and, in some ways, transformed to that of a data scientist. Cao describes data science using the following formula[6]:

$$\begin{aligned} \text{data science} = \{\ &\text{statistics}\ \cap\ \text{informatics}\ \cap \\ &\text{computing}\ \cap\ \text{communication}\ \cap\ \text{sociology}\ \cap \\ &\text{management}\ \mid\ \text{data}\ \cap\ \text{domain}\ \cap\ \text{thinking}\ \} \end{aligned} \tag{1}$$

where data science is the union of statistics, informatics, computing, communication, sociology, and management conditional on the union of data, domain, and thinking. While data science encompasses more advanced techniques and algorithm development, data analysts can still benefit greatly by expanding into the data science domain. Analysts who can find novel sources of data and extract, load, and transform them provide greater value to their organization. Analysts who do not adapt to big data and the evolution of data analytics will produce inferior estimates and models. Fortunately, analysts can embrace *Government Data 2.0* by learning a few simple tricks, techniques, and best practices. We will explore just a few of these techniques: manipulating and exploring large data sets with R, text mining, and data visualization for pre-processing.

Pre-Processing and Exploring Data

In order to assess the content and quality of a data set, analysts may need to use a combination of techniques. We will use R to demonstrate some techniques for processing and analyzing data. As the volume of data available grows, spreadsheets reach their processing limits and become unwieldy or unable to perform. R, however, was built as a statistical computing environment and excels at handling large files. R has many functions for inspecting data sets for errors, anomalies, outliers, and trends. R is our preferred choice for manipulating large data files because the software is open source and straightforward to use, especially when used with an interface like R Studio.

In our first example using logistics data, the data requires some initial manual work before it can be used in a platform like R or Tableau. Our logistics data, located in a file called "log.xls", compiles parts data for display system components which were replaced in the field by technicians on five similar legacy systems and returned to the log center for processing and disposal. These display systems vary in age fielded and hardware but are all used to satisfy the same objective for users across the US. Upon first glance, the data seems complete. This data set was previously labeled "messy" and abandoned. Quick analysis reveals why—there are some glaring errors and other formatting issues. Some manual pre-processing of the data can correct these issues. First, the data was compiled in cross-tab format and cannot be easily loaded into R, Tableau, or other data analysis software until it is transformed into a proper data table. Further inspection reveals an unusual trend in the data which was caused by some values being entered into the wrong column. These values are accurate, but they need to be shifted to their corresponding column in order for the data to be useful. Now the data set is ready for more advanced techniques in R. There is a text column in this data set which contains notes entered from the field about all the parts used for that order. We will demonstrate in the next section how the seemingly unhelpful text field can provide a lot of insight with some simple text mining procedures.

[6] Longbing Cao. "Data science: challenges and directions". In: *Communications of the ACM* 60.8 (Aug. 2017), pp. 59–68.

Mining Text

One particularly useful and relatively simple technique that can be exploited in R is mining text fields. Many legacy systems, complaint logs, repair records, and other sources of data have unstructured text fields whose contents contain valuable information[7]. We will use a simple form of text mining called "bag-of-words" to extract information from the text field of our logistics file. Bag-of-words text mining treats each word as an individual item and counts the frequency each word appears. This process disregards sentence structure and requires some additional cleaning to ensure the text being mined has no spelling errors. We will also need to remove common English words, called *stop words*, that would otherwise overwhelm our frequency counts due to their extensive use in the English language[8]. Some common examples of stop words are 'you', 'the', 'is', etc.

To begin, we read in the csv file to a data frame called "parts.df" and label the text column we want to mine as `parts_desc`. We then clean this field by removing punctuation. Note that the condition `stringsAsFactors=FALSE` defaults to true but must be set to false in order for R to treat them like strings.

```
parts.df <- read.csv("logistics.csv", stringsAsFactors=FALSE)
parts_desc <- parts.df$SALES_PART_DESCRIPTION
parts_desc_clean <- removePunctuation(parts_desc)
```

We will use a few different libraries to help perform the necessary functions and visualize the results. These libraries are: `tm`, `qdap`, and `RWeka`. We also need to create a tokenizer function. `Tokenizer` allows us to visualize words as singular words or in groups. Here we have it set to show singular words and paired words (denoted by the input "max" set equal to two).

```
library(tm)
library(qdap)
library(RWeka)
tokenizer <- function(x) {
NGramTokenizer(x, Weka_control(min = 1, max = 2))
}
```

Now we need a corpus, or collection of text. We first create a vector called `parts_source` that we can then feed into the function `VCorpus`, which denotes a "volatile" corpus, or one that is stored in memory. We will call our corpus `parts_corpus`.

```
#Create vector
parts_source <- VectorSource(parts_desc_clean)
#Create corpus
parts_corpus <- VCorpus(parts_source)
```

Next we are going to build a generic function called `clean_corpus` which we can use for other applications. This particular function performs a collection of subfunctions that remove punctuation, transform all text to lower case, remove common English stopwords, and strip whitespace.

```
# Alter the function code to match the instructions
clean_corpus <- function(corpus) {
# Remove punctuation
corpus <- tm_map(corpus, removePunctuation)
# Transform to lower case
```

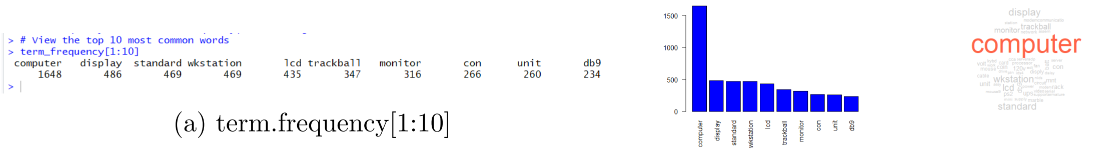
[7] Provost and Fawcett, *Data science for business: what you need to know about data mining and analytic thinking.*

[8] Max Bramer. *Principles of Data Mining.* Springer, 2007.

```
corpus <- tm_map(corpus, content_transformer(tolower))
# Remove stop words using common English stop words
corpus <- tm_map(corpus, removeWords, c(stopwords("en")))
# Strip whitespace
corpus <- tm_map(corpus, stripWhitespace)
return(corpus)
}
```

Now we can clean our corpus with this newly created function and create a term document matrix. We will then transform this into a simple matrix, derive term frequency, and sort by most frequently appearing terms.

```
parts_clean <- clean_corpus(parts_corpus)
parts_tdm <- TermDocumentMatrix(parts_clean)
parts_m <- as.matrix(parts_tdm)
term_frequency <- rowSums(parts_m)
term_frequency<- sort(term_frequency, decreasing = TRUE)
```



```
> # View the top 10 most common words
> term_frequency[1:10]
 computer   display  standard wkstation       lcd trackball   monitor       con      unit       db9 
     1648       486       469       469       435       347       316       266       260       234 
> |
```

(a) term.frequency[1:10]

(b) barplot

(c) wordcloud

Figure 1: R outputs for various functions

After we have established frequency data from our text mining procedures, we can visualize the results. Here, we use the function `term_frequency` to view the top 10 words. This helps establish if any additional stop words need to be added to our cleansing function. You can do this by amending the `removeWords` function to add those words to your stop words vector. We can also create a bar chart and word cloud using the top 50 words.

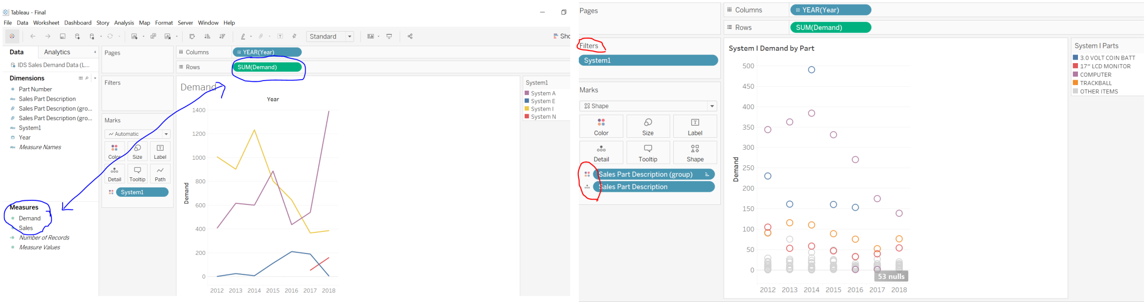
```
# View the top 10 most common words
term_frequency[1:10]

# Plot a barchart of the 10 most common words
barplot(term_frequency[1:10], col ="blue", las=2)

# Load wordcloud package
library(wordcloud)

terms_vec <- names(term_frequency)

wordcloud(terms_vec, term_frequency, max.words = 50, colors = "red")

wordcloud(terms_vec, term_frequency, max.words =50,
colors = c("grey80","darkgoldenrod1","tomato"))
```

The outputs of these functions are available in figure 1. Finally, we save our text mining results to a file called "terms.csv" so that we can perform further analysis in Tableau or other visualization software.

```
setwd("C:/Your Working Directory Path Here")
write.csv(term_frequency, file="terms.csv")
```

By executing some simple code in R, we were able to gain valuable insight about our logistics data.

For quick visual manipulation of our full logistics data set, we can use Tableau. We can explore trends over time by plotting cost over time and then coloring the data according to different variables to look for patterns and trends. We can also quickly interchange different metrics plotted over time. This allows us to rapidly establish whether there are any relationships worth investigating. The logistics data shows some trends in parts demand over time for our systems A and I that warrant further exploration, shown in figure 2a. In figure 2b, we further explore system I demand over time and parse the data by sales part groupings, where parts with low demand are grouped into a field called "other". We can immediately see which parts are most problematic for this system. A story begins to emerge regarding these two systems which we will explore in a dashboard later on.



Columns YEAR(Year)
Rows SUM(Demand)

(a) Interchange variables quickly in Tableau

(b) Annual demand for System I by part

Figure 2: Using Tableau to explore data

Exploring Large Data Set Structure

Cost estimators traditionally use data visualization to tell a story. Beyond this traditional role, data visualization can also provide analysts a powerful tool to:

- identify inaccurate values,
- find missing values,
- find duplicate values,
- establish candidates for bins, and
- highlight potential opportunities to consolidate data[9].

In addition to processing and transforming data, R can also aid in finding errors and outliers as well as assessing relationships via correlation matrices. When working with a large data warehouse, you may need to inspect a table to determine its fields, structure, and the type of data contained therein. Let us inspect a large table of data for online data orders. We have pulled the table from the server already using SQL and loaded it into the data frame called `orders.df`. We need to load the following libraries:

```
#load libraries
library(ggplot2)
library(corrplot)
library(corrgram)
library(GGally)
```

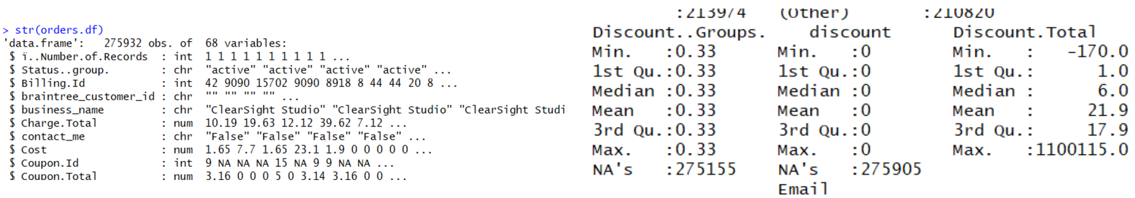
[9] Galit Shmueli et al. *Data Mining for Business Analytics*. Wiley, 2018.

```
library(ggbeeswarm)
library(dplyr)
library(stringr)
```

Now we need to inspect our data frame. The following commands produce various summary information about our data frame:

```
head(orders.df)
str(orders.df)
summary(orders.df)
```

The **head**() command returns the first six rows of our data frame. You can show more or less rows by entering the argument **head**$(x, n = y)$ where x refers to the data frame and y represents the desired number of rows. The **str**() and **summary**() functions are similar arguments which display the structure of the data frame. These commands will tell you how many objects and variables are in your data frame, what those variables are, how they are classified, and provide a sample of values. The function **summary** lists all the variables in the data frame along with descriptive measures for numerical variables such as mean, median, and max.



```
> str(orders.df)
'data.frame':	275932 obs. of  68 variables:
 $ ï..Number.of.Records  : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Status..group.        : chr  "active" "active" "active" "active" ...
 $ Billing.Id            : int  42 9090 15702 9090 8918 8 44 44 20 8 ...
 $ braintree_customer_id : chr  "" "" "" "" ...
 $ business_name         : chr  "ClearSight Studio" "ClearSight Studio" "ClearSight Studi
 $ Charge.Total          : num  10.19 19.63 12.12 39.62 7.12 ...
 $ contact_me            : chr  "False" "False" "False" "False" ...
 $ Cost                  : num  1.65 7.7 1.65 23.1 1.9 0 0 0 0 0 ...
 $ Coupon.Id             : int  9 NA NA NA 15 NA 9 9 NA NA ...
 $ Coupon.Total          : num  3.16 0 0 0 5 0 3.14 3.16 0 0 ...
```

(a) Str(orders.df) example

```
         :213974   (Other)          :210820
 Discount..Groups.    discount        Discount.Total
 Min.   :0.33      Min.   :0        Min.   :   -170.0
 1st Qu.:0.33      1st Qu.:0        1st Qu.:      1.0
 Median :0.33      Median :0        Median :      6.0
 Mean   :0.33      Mean   :0        Mean   :     21.9
 3rd Qu.:0.33      3rd Qu.:0        3rd Qu.:     17.9
 Max.   :0.33      Max.   :0        Max.   :1100115.0
 NA's   :275155    NA's   :275905
                   Email
```

(b) Summary(orders.df) example

Figure 3: Str vs Summary command

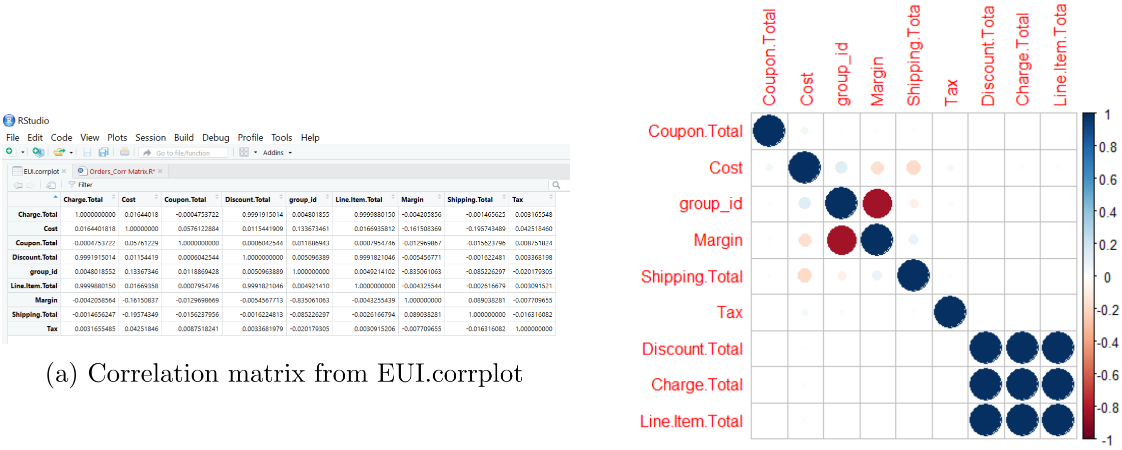
Figure 3a shows some of the output of **str**(**orders.df**), and figure 3b shows some of the output from **summary**(**orders.df**). By examining the contents, reviewing the data types and sample values, and cross-checking the summary stats, we can learn a lot about our data very quickly and instantly identify potential errors. For example, in figure 3b, the minimum value in the column labeled **discount.value** is a negative number. Intuitively, orders should not have negative discounts applied to them (that would be a bad deal for the customer, after all). When working with a new data set, these techniques can quickly help an analyst familiarize themselves with its contents.

Exploring Data Visually

Correlation matrices are a great technique for further visualizing data. By constructing a correlation matrix, the analyst can identify outliers and quickly determine which relationships to explore further. Using "orders.df" and R, we will generate a correlation matrix to demonstrate the utility of this technique. The columns containing numerical variables which might be of interest were first identified to create the correlation matrix and visual plot shown in figure 4.

```
#generate correlation matrix using vector of previously identified columns
EUI.corrplot <- cor(orders.df[c(6,8,10,20,31,38,40,56,62)])
#visualize results
corrplot(EUI.corrplot,order="hclust")
```

Given this information, we can use the **ggpairs**() function to further explore some of the highly correlated variables. Right away, the plot in figure 5a shows an outlier that



	Charge.Total	Cost	Coupon.Total	Discount.Total	group_id	Line.Item.Total	Margin	Shipping.Total	Tax
Charge.Total	1.0000000000	0.01644018	-0.0004753722	0.9991915014	0.004801855	0.9999880150	-0.004205856	-0.001465625	0.003165548
Cost	0.0164401818	1.00000000	0.0576122884	0.0115441909	0.133673461	0.0166935812	-0.161508369	-0.195743489	0.042518460
Coupon.Total	-0.0004753722	0.05761229	1.0000000000	0.0006042544	0.011886943	0.0007954746	-0.012969867	-0.015623796	0.008751824
Discount.Total	0.9991915014	0.01154419	0.0006042544	1.0000000000	0.005096389	0.9991821046	-0.005456771	-0.001622481	0.003368198
group_id	0.0048018552	0.13367346	0.0118869428	0.0050963889	1.000000000	0.0049214102	-0.835061063	-0.085226297	-0.020179305
Line.Item.Total	0.9999880150	0.01669358	0.0007954746	0.9991821046	0.004921410	1.0000000000	-0.004325544	-0.002616679	0.003091521
Margin	-0.0042058564	-0.16150837	-0.0129698669	-0.0054567713	-0.835061063	-0.0043255439	1.000000000	0.089038281	-0.007709655
Shipping.Total	-0.0014656247	-0.19574349	-0.0156237956	-0.0016224813	-0.085226297	-0.0026166794	0.089038281	1.000000000	-0.016316082
Tax	0.0031655485	0.04251846	0.0087518241	0.0033681979	-0.020179305	0.0030915206	-0.007709655	-0.016316082	1.000000000

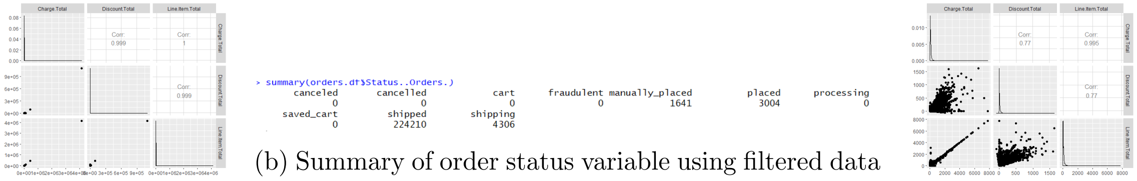
(a) Correlation matrix from EUI.corrplot

(b) EUI corrplot() result

Figure 4: Correlation matrix and EUI corrplot output

needs investigating. Pulling the max value for the variable `Charge.total`, we get a value of $4.1M. A quick call to the client confirms that the max order should be well below $10K. Further exploration reveals that one variable representing order status must be filtered. We can modify our data frame to filter the data after we load the libraries where the | operator represents "or" and we are filtering on the column `Status..orders..` We can verify that this filter works by calling the function `summary`() on the status column. As shown in figure 5b, only the values we filtered on are represented in our data frame now. The revised `ggpairs`() plot in figure 5c shows the correct results.

```
#explore potential drivers of total charge
ggpairs(orders.df[,c(6,20,38)])

#Find max value for troubleshooting
max(orders.df$Charge.Total,   na.rm = FALSE, dims = 1, n = NULL)

#Filter data on Status..Orders
orders.df <-orders.df %>% filter(Status..Orders. == "placed" |
Status..Orders. == "manually_placed" |
Status..Orders. == "shipped" |
Status..Orders. == "shipping")

#confirm filter worked
summary(orders.df$Status..Orders.)
```



```
> summary(orders.df$Status..Orders.)
       canceled       cancelled            cart      fraudulent manually_placed          placed      processing 
              0               0               0               0            1641            3004               0 
     saved_cart         shipped        shipping 
              0          224210            4306 
```

(b) Summary of order status variable using filtered data

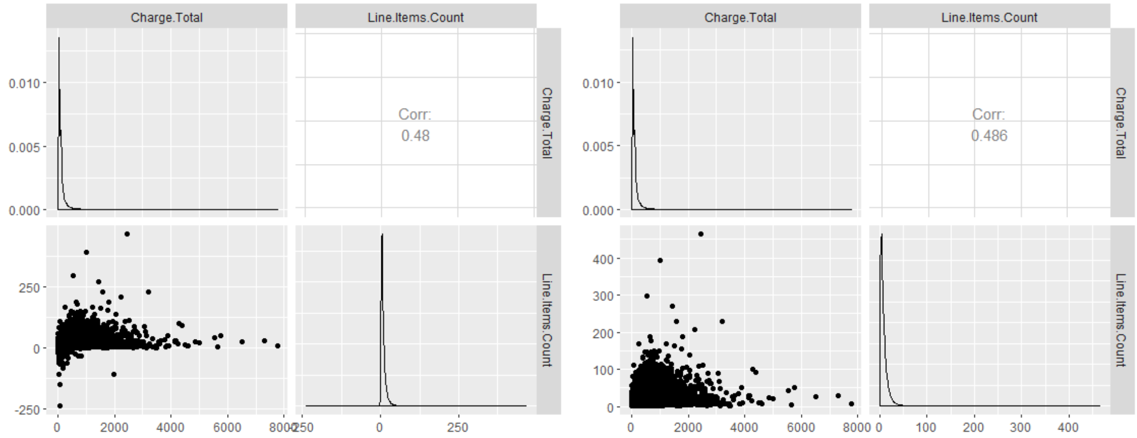
(a) Ggpairs plot showing an outlier

(c) Ggpairs plot with filtered data

Figure 5: Finding and filtering outliers

As a final example, we attempt to determine if a meaningful relationship exists between total quantity of line items and total charges. The variable for line item quantity resides in column 39. The resulting ggpairs() plot shows a large number of orders that have negative line items. Intuitively, an order can not have negative items. If we filter the data to exclude these negative orders (which happen to represent glitches in the system) and rerun ggpairs() we see in figure 6 that the correlation between quantity and total charge is weak and not worth further investigation. The R code to execute these commands is:

```
#explore line item count and total charge
ggpairs(orders.df[,c(6,39)])
#remove negative line item counts and re-run
orders_licorr.df <- orders.df %>% filter(Line.Items.Count > 0)
#run ggpairs again to see new result
ggpairs(orders_licorr.df[,c(6,39)])
```



(a) Before filtering

(b) After filtering

Figure 6: Filtering out negative line items

Visualizing Data

Now that we have expanded our analytical skill set, we must effectively communicate our findings to leadership and stakeholders. The most important thing an analyst can do when designing a graph or chart is to ask, "What do I want my audience to know?" The same set of data can be presented in vastly different ways. Analysts already excel at summarizing and visualizing data. To push their organization towards a data-driven mindset, they must also ensure the data they present provides insights that are:

- clear,
- meaningful, and
- actionable.

Clear

Each graphic in a presentation or dashboard must draw the reader directly to the point being made. Cluttered charts distract the reader from the message. Remove axis lines and unnecessary labels. Highlight the takeaways by using color, size, or shape to guide the reader. Knaflic argues that analysts should reduce the cognitive load on the reader by removing anything in a graphic, chart, or dashboard that is not informative enough to justify the brain power required by the audience to process it[10].

Meaningful

Analysis adds value when it addresses what its audience wants to know, uses data to drive their understanding, and measures how well they act upon that data[11]. Creating an attractive chart with no substance might be pleasing to look at but leaves the audience wondering how your analysis benefits them. Often, analysts make the mistake of showing their audience their exploratory analysis, instead of purposefully developing graphics which are explanatory in nature and designed to inform[12]. Your graphics supplement the defense of your analysis. Make sure they tell your audience the most important things they need to know.

Actionable

Given the immense volume and variety of data available, analysts must take care not to analyze data simply for the sake of doing analysis. Knowing the objective of your analysis will prevent wasted time and ensure findings are relevant to the problem that needs solving.

Returning to the logistics data we text mined, we can summarize several actionable insights into a dashboard, shown in the appendix. Of all parts, total demand for computers for System I dominates total demand more than any other part. System A's demand for standard workstations also occurs more frequently than many other parts. Both of these observations reflect on the age of their respective systems. System I was fielded in 1990 and System A in 2000. The systems appear to be failing catastrophically in the field, resulting in a need for complete replacement rather than replacement of a specific part. Closer inspection of System A demand shows three trends:

- 5E LAN cables experienced unusually high spikes in failures in 2014 and especially in 2018.
- Workstations have been failing at a relatively steady rate since 2012, and demand is generally higher than other spare parts. Since demand is not increasing over time, this may indicate that this system is difficult to repair in the field, resulting in a higher occurrence of total replacement.
- Surge arresters fail at somewhat higher rates, with certain years showing a spike in demand.

Closer inspection of System I demand shows four trends:

- 3.0 volt coin batteries fail at a moderately high rate, with an especially high replacement demand in 2014.
- Whole computers fail much more than spare parts, though the rate at which they have been failing has been declining since 2014.
- Trackball demand is consistently moderately higher than other parts.
- The 17" monitors are replaced more often than most other parts, but not 20" monitors. This is likely not because of failures, but because 17" monitors are frequently deemed unacceptable in the field and therefore have to be replaced with larger screens.

[10] Cole Nussbaumer Knaflic. *Storytelling with Data.* Wiley, 2015.

[11] Tim Dickinson and James Densmore. "Become a Data Science Expert". In: *Talent Development* (Nov. 2018).

[12] Knaflic, *Storytelling with Data.*

The agency is working to replace all legacy systems with one enterprise system. Based on this analysis, they should replace Systems I and A first, then prioritize System E over System N. No monitors under 20" should be considered in the replacement system. Better human factors analysis prior to fielding the replacement system will ensure no other parts are returned simply because of user preference. Furthermore, the trackballs used by System I should be discontinued. What was previously considered unusable data has generated several actionable insights to inform the program manager and human factors team, preventing repeated mistakes by the replacement program. As a cost estimator, we can defend decisions to cost more expensive monitors and other specific hardware specs using this analysis. We can also predict how frequently we can expect systems to fail before we can implement a replacement in the field, impacting legacy operations costs and, ultimately, our business case.

Summary

The way analysts retrieve and analyze data continues to change at a rapid pace. Government initiatives have encouraged the harnessing of big data; how can we as analysts adapt to this changing environment and propel ourselves to the front of our profession? We have discussed several ways in which the fields of analysis and estimating have begun to overlap with the enormous discipline of data science. Analysts can capitalize on these changes, incorporating new skills and techniques to make smarter predictions and improve the accuracy of their estimates. Hopefully the examples demonstrated here will inspire you to rethink the idea of "bad" data and find ways to leverage data you may have previously overlooked.

Bibliography

Bramer, Max. *Principles of Data Mining.* Springer, 2007.

Brown, Greg. "Just how accurate is our data?" In: *The Fair Observer* (May 2018).

Cao, Longbing. "Data science: challenges and directions". In: *Communications of the ACM* 60.8 (Aug. 2017), pp. 59–68.

Dickinson, Tim and James Densmore. "Become a Data Science Expert". In: *Talent Development* (Nov. 2018).

Knaflic, Cole Nussbaumer. *Storytelling with Data.* Wiley, 2015.

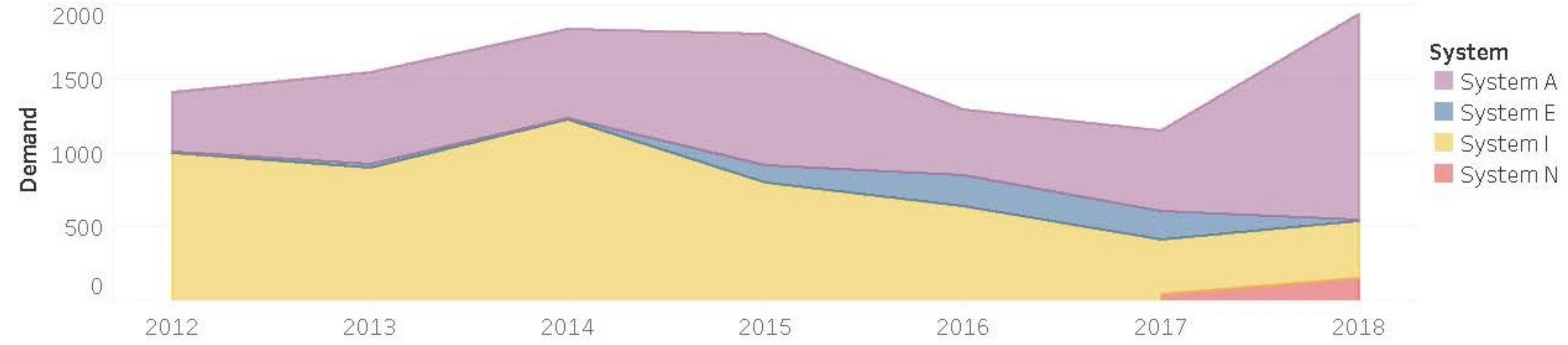
Lucker, John, Susan Hogan, and Trevor Bischoff. "Predictably inaaccurate: the perils and struggles of bad big data". In: *Deloitte Review* 21 (July 2017), pp. 8–25.

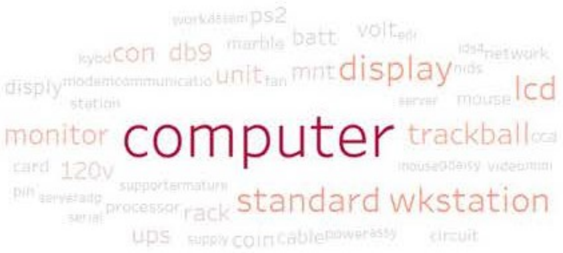
Provost, Foster and Tom Fawcett. *Data science for business: what you need to know about data mining and analytic thinking.* O'Reilly, 2013.

Shmueli, Galit et al. *Data Mining for Business Analytics.* Wiley, 2018.

The federal big data reseach and development strategic plan. Tech. rep. NITRD, May 2016.

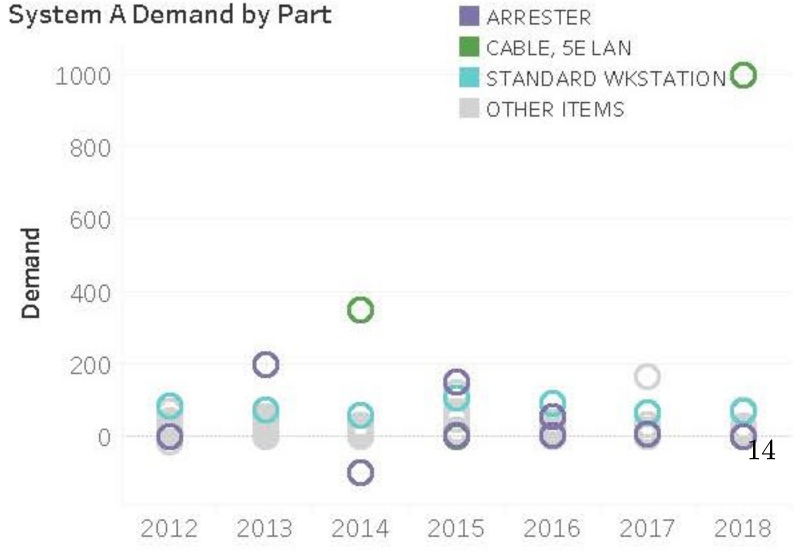
Appendices



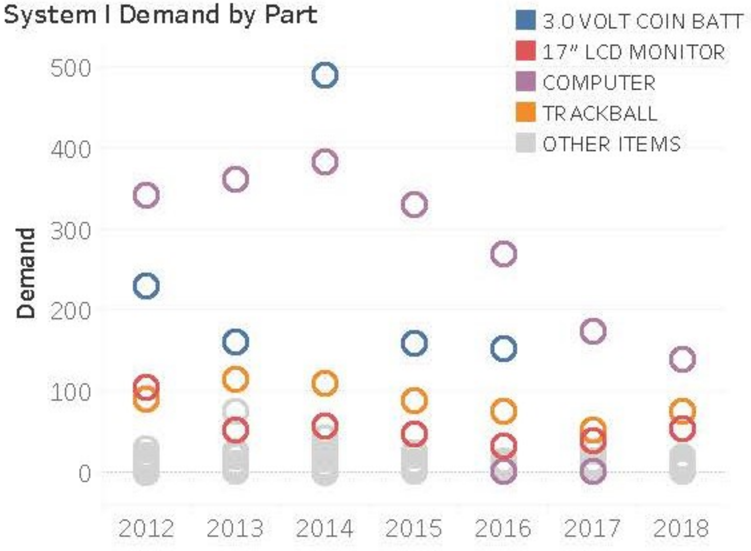


Most demand for parts come from Systems A and I. The highest demand for parts are computers for System I, indicating complete system failures are occurring. Failures spiked in 2014 and have been declining steadily since. Steady failures of 17" monitors, 3.0 volt coin batteries, and trackballs are also occurring. System A experiences a moderate but steady level of workstation replacement, and periodic spikes in 5E LAN cable and arrester part demands.

System A Demand by Part



System I Demand by Part


3.0 VOLT COIN BATT
17" LCD MONITOR
COMPUTER
TRACKBALL
OTHER ITEMS
Demand