Learning From Failure

Noah Wright

Table of contents

## Background

Body-worn cameras became a standard tool of policing in the mid-2010s. By recording officer interactions with the public, law enforcement agencies could achieve a greater degree of accountability. Not only could credible claims of police abuse against civilians be easily verified, the argument went, but false accusations would decline as well, saving the agency time and resources that would otherwise be wasted on spurious allegations. Initial studies seemed to support this argument, and by the end of the decade it became common practice in the United States for police to have these devices.

It was also in the late 2010s that new leadership came to the Texas Juvenile Justice Department (TJJD).[[1]](#footnote-20) Incarcerated youth are an exceptionally vulnerable population even in a well-run system. Texas did not have a well-run system. With one notable period of exception,[[2]](#footnote-21) scandal after scandal had rocked the juvenile justice system of Texas from its inception in the late 1800s through the mid-2010s. Once again, new leadership had been appointed following yet another abuse scandal. This time, however, the new executive director had a tool unavailable to her predecessors: body-worn cameras.

Secure youth residential facilities in Texas all had overhead cameras, but these were very old (they still ran on tape) and captured no audio. This presented a number of problems when it came to deciphering contested incidents, not to mention that these cameras had clearly not prevented any of the prior scandals from taking place. TJJD received special funding from the legislature to roll out body-worn cameras system-wide, and after the cameras had been successfully deployed, I was tasked with seeing if staff were complying with the usage policies we had put in place.

Part of the issue was that the use of body-worn cameras was new in the field of corrections. There was not a large literature to draw from, and our first attempt at regulating and monitoring their usage fell flat on its face. This is a case study of how the Texas Juvenile Justice Department developed a methodology for comparing data on correctional officers’ body-worn camera usage with data from its shift management software. This is also a case study of analysis failure, as it became clear that real-world failures of implementation were corrupting the data and rendering the methodology unusable. However, the silver lining of this failure was that it helped the agency identify previously unrecognized problems. The purpose of this case study is to demonstrate how negative findings can still be incredibly useful in real-world settings.

In 2019, body worn cameras were mostly a tool for law enforcement, and their use in a correctional environment posed unique new challenges. TJJD had no automated mechanism for reviewing footage through machine learning, nor did the agency have the IT capacity to develop one. The only way we had to analyze footage was to view it. Unlike police officers, correctional officers deal directly with their charges for virtually their entire shift. In an 8-hour shift, a police officer might record a few calls and traffic stops. A correctional officer, on the other hand, would record 8 consecutive hours. And because TJJD recorded round-the-clock for hundreds of employees at a time, every hour of real time that passed had, on average, an eye-popping 100 hours of footage. To put that another way, reviewing a single hour of TJJD operations across all facilities would take 2.5 full weeks of work.

As a result, footage review had to be done in a limited, reactive manner. If we received an incident report we could easily zero in on the cameras of the officers involved and review the incident accordingly. But our executive director had hoped to be able to use the footage proactively, looking for ‘red flags’ in order to prevent potential abuses.

Because the agency had no way of automating the monitoring of footage, any proactive analysis had to be metadata based. But what to look for in the metadata? Once again, the lack of best-practice literature left us in the lurch. So, we brainstormed ideas for ‘red flags’ and came up with the following:

1. **Minimal quantity of footage** – our camera policy required correctional officers to have their cameras on at all times in the presence of youth.
2. **Frequently turning the camera on and off** – outside of bathroom breaks, a correctional officer working a dorm should not be turning their camera on and off repeatedly throughout the day.
3. **Large gaps between clips** – A three-hour gap in the middle of a shift should be looked into, even if it’s an accident or honest mistake. It defeats the purpose of having cameras if they’re not turned on.
4. **Mismatch between clips recorded and shifts worked** ­– the agency had very recently rolled out a new shift tracking software. We should expect to see the hours logged by the body cameras roughly match the hours logged as shifts worked.

## Analysis Part 1 – QC and Footage Analysis

For this analysis, I gathered the most recent three weeks of body-worn camera data, which, at the time, covered April 1–21, 2019. I also pulled data from Shifthound (our shift management software) covering the same time period. Finally, I gathered HR data from CAPPS, the system that most of the State of Texas used at the time for personnel management and finance.[[3]](#footnote-23)

An initial quality control review of these data sources revealed a variety of issues. See the R script below for details, but the most important ones I discovered were missing and/or bad employee IDs. Each employee had a unique identifier within CAPPS, which in turn was used to identify their assigned body camera and log their shifts. So it came as something of a surprise to find employee IDs in both the body camera and shift data that did not exist in CAPPS. Likewise, I found IDs of correctional officers in CAPPS who should have appeared in the body camera or shift data but did not. Lastly, I found IDs of staff who did appear in both systems, but were not employed at any point between April 1–21, 2019.

#Import and Format Data, Establish Parameters  
  
library(tidyverse)  
library(readxl)  
  
Footage\_Data <- read\_excel("Data/Original Data/Footage Data.xlsx", col\_types = "text") %>%   
 mutate(Duration\_Hours = as.numeric(duration\_seconds)/3600) %>%   
 rename(Footage\_Job\_Type = `Employee Role`)  
  
Shift\_Data <- read\_excel("Data/Original Data/Shift Data.xlsx") %>%   
 mutate(`Employee ID` = as.character(`Employee ID`)) %>%   
 rename(Shift\_Job\_Type = `Job Type`, Shift\_Location = Facility)  
  
HR\_Data <- read\_csv("Data/Original Data/HR Data.csv", col\_types = "c") %>%   
 mutate(across(contains("Date"), ~as.Date(.x, "%m/%d/%Y"))) %>%   
 rename(HR\_Job\_Type = `Job Type`, HR\_Location = Location)  
  
Secure\_Facilities <- c("Alpha", "Beta", "Gamma", "Delta", "Epsilon")  
  
#Underscore all variable names to make R coding easier  
  
colnames(Footage\_Data) <- str\_replace\_all(colnames(Footage\_Data), " ", "\_")  
colnames(Shift\_Data) <- str\_replace\_all(colnames(Shift\_Data), " ", "\_")  
colnames(HR\_Data) <- str\_replace\_all(colnames(HR\_Data), " ", "\_")  
  
#Now that we have the data, let's start with an exploratory skim.  
  
library(skimr)  
  
#If you don't use skim on new data sets I don't know what you're doing.  
  
skim(Footage\_Data)  
skim(HR\_Data)  
skim(Shift\_Data)  
  
#Looks like we have a few missing Employee IDs in footage data. Also, time  
#fields in shift data have been cursed by Excel date formatting, will need to be  
#exorcised later on.  
  
#All IDs should appear in HR data. Check for IDs in footage and shift data  
#that do not appear in HR data.  
  
Footage\_ID\_Missing\_in\_HR <- Footage\_Data %>%   
 select(Employee\_ID) %>%   
 unique() %>%   
 na.omit() %>%   
 anti\_join(HR\_Data, by = "Employee\_ID")  
  
Shift\_ID\_Missing\_in\_HR <- Shift\_Data %>%   
 select(Employee\_ID) %>%   
 unique() %>%   
 na.omit() %>%   
 anti\_join(HR\_Data, by = "Employee\_ID")  
  
Invalid\_IDs <- Footage\_ID\_Missing\_in\_HR %>%   
 bind\_rows(Shift\_ID\_Missing\_in\_HR) %>%   
 mutate(Reason = "ID not in HR system")  
  
#HR system is authoritative source, so these 44 IDs are errors.  
  
#Let's see how much the data labels match up  
#In other words, are the roles and locations consistent from one data set to another?  
  
QC\_Labels <- HR\_Data %>%   
 left\_join(Footage\_Data %>%   
 select(Employee\_ID, Footage\_Job\_Type) %>%   
 unique()) %>%   
 left\_join(Shift\_Data %>%   
 select(Employee\_ID, Shift\_Job\_Type, Shift\_Location) %>%   
 unique())  
  
Label\_Mismatch <- QC\_Labels %>%   
 filter(Footage\_Job\_Type != HR\_Job\_Type |  
 Footage\_Job\_Type != Shift\_Job\_Type |  
 HR\_Job\_Type != Shift\_Job\_Type|  
 HR\_Location != Shift\_Location) %>%   
 mutate(Reason = "Job Type or Location Label Mismatch")  
  
#Role and location mismatch (77 IDs) minimal.  
#Not a huge deal, might have just been mislabeled in shift or camera system  
  
#So far so good, but this is the easy stuff.  
#We expect to see somewhere in the footage data all Employee IDs that  
#  
#-are correctional (only correctional staff are required to wear cameras at all times on the dorm)  
#-work at a secure facility (roll out to halfway houses was still ongoing)  
#-were employed during the time frame of the study, 2019-04-01 through 2019-04-21  
#-not in their first 3 months of employment, a mandatory training period prior to interacting with youth  
  
HR\_Relevant <- HR\_Data %>%   
 filter(HR\_Job\_Type %in% c("Correctional", "Correctional Supervisor"),  
 HR\_Location %in% Secure\_Facilities,  
 is.na(Termination\_Date) | Termination\_Date >= "2019-04-01",  
 Hire\_Date <= "2019-01-21")  
  
write\_csv(HR\_Relevant, "Data/Relevant HR Data.csv")  
  
#HR data indicates 863 staff that meet this criteria.   
#How many are missing in the footage and shift data?  
  
Missing\_From\_Footage\_Data <- HR\_Relevant %>%   
 anti\_join(Footage\_Data, by = "Employee\_ID") %>%   
 mutate(Reason = "Missing from footage data")  
  
Missing\_From\_Shift\_Data <- HR\_Relevant %>%   
 anti\_join(Shift\_Data, by = "Employee\_ID") %>%   
 mutate(Reason = "Missing from shift data")  
  
Expected\_IDs\_Missing <- Missing\_From\_Footage\_Data %>%   
 bind\_rows(Missing\_From\_Shift\_Data)  
  
#149 expected HR IDs missing in footage and 120 expected HR IDs missing from HR system.   
#269 IDs missing in total. That could definitely be better.  
  
#What about staff whose IDs exist but were not employed during the time frame of the study?  
  
Footage\_Not\_Employed <- Footage\_Data %>%   
 anti\_join(HR\_Relevant, by = "Employee\_ID") %>%   
 anti\_join(Invalid\_IDs) %>%   
 count(Employee\_ID) %>%   
 inner\_join(HR\_Data) %>%   
 filter(Hire\_Date > "2019-04-21"|  
 Termination\_Date < "2019-04-01") %>%   
 select(Employee\_ID) %>%   
 unique() %>%   
 mutate(Reason = "Not employed at time of footage")  
  
Shift\_Not\_Employed <- Shift\_Data %>%   
 anti\_join(HR\_Relevant, by = "Employee\_ID") %>%   
 anti\_join(Invalid\_IDs) %>%   
 count(Employee\_ID) %>%   
 inner\_join(HR\_Data) %>%   
 filter(Hire\_Date > "2019-04-21"|  
 Termination\_Date < "2019-04-01") %>%   
 select(Employee\_ID) %>%   
 unique() %>%   
 mutate(Reason = "Not employed at time of shift")  
  
Not\_Employed <- Footage\_Not\_Employed %>%   
 bind\_rows(Shift\_Not\_Employed)  
  
#27 footage IDs and 13 Shift IDs appear in data but were assigned to staff not  
#employed during the time frame.  
  
#All of the problem IDs will be excluded from analysis and handed off to  
#facility staff for follow-up. Label mismatches will be followed up on but not  
#excluded from analysis.  
  
Excluded\_ID <- Invalid\_IDs %>%   
 bind\_rows(Expected\_IDs\_Missing) %>%   
 bind\_rows(Not\_Employed)  
  
Followup\_ID <- Excluded\_ID %>%   
 bind\_rows(Label\_Mismatch)  
  
write\_csv(Followup\_ID, "Output/ID's needing follow-up 1.csv")  
  
#How much of the total quantity of footage needs to be excluded from the  
#analysis for reasons of data quality?  
  
Excluded\_ID\_Footage <- Footage\_Data %>%   
 semi\_join(Excluded\_ID, by = "Employee\_ID")  
  
nrow(Excluded\_ID\_Footage)/nrow(Footage\_Data)  
sum(Excluded\_ID\_Footage$Duration\_Hours)/sum(Footage\_Data$Duration\_Hours)  
  
#11.3% of clips representing 10.6% of total footage excluded from analysis  
#for data quality issues  
  
#How much of the total quantity of footage is being excluded for any reason?  
#Some footage will not be included because it does not fall under the  
#"always-on" policy. For example, oversight employees (such as Monitoring or  
#Inspector General staff) use cameras when interacting with youth but do not  
#work full-time on the dorm. We would not expect to see their camera usage line  
#up with their recorded shifts.  
  
library(lubridate)  
  
Analysis\_Footage <- Footage\_Data %>%   
 semi\_join(HR\_Relevant, by = "Employee\_ID") %>%   
 arrange(Employee\_ID, Camera\_Serial\_Number, created\_date\_record\_start) %>%   
 mutate(Clip\_Start = ymd\_hms(created\_date\_record\_start, tz = "US/Central"),  
 Clip\_End = ymd\_hms(date\_record\_end, tz = "US/Central"))  
  
1-nrow(Analysis\_Footage)/nrow(Footage\_Data)  
1-sum(Analysis\_Footage$Duration\_Hours)/sum(Footage\_Data$Duration\_Hours)  
  
#12.8% of clips representing 14.0% of total footage excluded from analysis  
#for either data quality issues or employee not subject to always-on policy  
  
#Remove irrelevant staff from Shift data and store copies of formatted tables  
  
Shift\_Format <- Shift\_Data %>%   
 semi\_join(HR\_Relevant) %>%  
 mutate(Shift\_Start = ymd\_hms(str\_c(Start\_Date, " ", str\_sub(Start\_Time, -8, -1)), tz = "US/Central"),  
 Shift\_End = ymd\_hms(str\_c(End\_Date, " ", str\_sub(End\_Time, -8, -1)), tz = "US/Central"))  
  
write\_csv(Analysis\_Footage, "Data/Analysis Footage.csv")  
write\_csv(Shift\_Format, "Data/Shift Data Formatted.csv")  
  
#Samples for reference in Quarto  
  
write\_csv(Analysis\_Footage %>%   
 filter(Employee\_ID == 9001005,  
 Clip\_Start <= "2019-04-03") %>%   
 select(Employee\_ID, Clip\_ID, Clip\_Start, Clip\_End),  
 "Output/Footage Sample.csv")  
  
write\_csv(Shift\_Format %>%   
 filter(Employee\_ID == 9001005,  
 Shift\_Start <= "2019-04-03") %>%   
 select(Employee\_ID, Shift\_ID, Shift\_Start, Shift\_End),  
 "Output/Shift Sample.csv")

In total, 10.6% of clips representing 11.3% of total footage had to be excluded from analysis due to these initial data quality issues.

In order to operationalize our brainstorming session, I needed to see what exactly the cameras captured in their metadata. The variables most relevant to our purposes were:

* Clip start
* Clip end
* Camera Used
* Who was assigned to the camera at the time
* The role of the person assigned to the camera

Using these fields, I first created the following aggregations per employee ID:

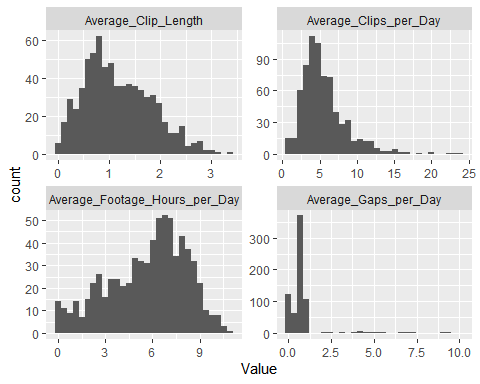
* **Number of clips** = Number of clips recorded.
* **Days with footage** = Number of discrete dates that appear in these clips.
* **Footage Hours** = Total duration of all shot footage.
* **Significant Gaps** = Number of clips where the previous clip’s end date was either greater than 15 minutes or less than 8 hours before this clip’s start date.

I then devised the following staff metrics using the following definitions:

* **Average clips per day** = Number of clips / Days with footage
* **Average footage per day** = Footage hours / Days with footage
* **Average clip length** = Footage hours / Number of clips
* **Average gaps per day** = Gaps / Days with footage

Then I looked at the distributions of each of the 4 per-staff metrics.

library(tidyverse)  
  
Footage\_Metrics\_by\_Employee <- read\_csv("Output/Footage Metrics by Employee.csv")  
  
Footage\_Metrics\_by\_Employee %>%   
 select(-Clips, -Days\_With\_Footage, -Footage\_Hours, -Gaps) %>%   
 pivot\_longer(-Employee\_ID, names\_to = "Metric", values\_to = "Value") %>%   
 ggplot(aes(x = Value)) +  
 geom\_histogram() +  
 facet\_wrap(~Metric, scales = "free")



By eyeballing I could tell most staff were recording fewer than 10 clips a day, shooting about 0.5–2 hours for each clip, for a total of 3–10 hours of daily footage, with the majority of employees having less than one significant gap per day. Superficially, this appeared to provide evidence of widespread attempts at complying with the policy and no systemic rejection or resistance. If this were indeed the case, then we could turn our attention to individual outliers.

First, though, we thought we’d see if we could validate this with another assumption. If each employee works on average 40 hours per week – a substantial underestimate given how common overtime was – we should expect, over a three-week period, to see about 120 hours of footage per employee in the dataset. This is *not* what we found.

Percent\_Display <- function(x) {  
 str\_c(100 \* round(x, 3), "%")  
}  
  
Total\_Footage <- sum(Footage\_Metrics\_by\_Employee$Footage\_Hours)  
Total\_Employees <- nrow(Footage\_Metrics\_by\_Employee)  
Average\_per\_Employee <- round((Total\_Footage/Total\_Employees),1)  
  
print(str\_c("Average Footage per Employee - ", Average\_per\_Employee, " hours"))

[1] "Average Footage per Employee - 70.2 hours"

print(str\_c(Average\_per\_Employee, " footage hours / 120 expected work hours = ", 100\*Average\_per\_Employee/120, "% of average shift recorded"))

[1] "70.2 footage hours / 120 expected work hours = 58.5% of average shift recorded"

Even with the unrealistic assumption of no overtime pushing the estimated total working hours down, less than 60% of these hours appear in the footage. A perfect match would be highly unlikely, but the fact that more than 40% of anticipated hours are unrecorded for unknown reasons warranted a follow-up.

Surely the shift data would clarify this.

## Analysis Part 2 – Footage and Shift Comparison

With the data on shifts worked from our timekeeping system, I could theoretically compare actual shifts worked to the amount of footage recorded. If there were patterns in where the gaps in footage fell, that might help to explain why.

In order to join the shift data to the camera data, I needed a common unit of analysis beyond “Employee ID”. Using only this value would produce a nonsensical table that joined up every clip of footage to every shift worked.

For example, let’s take employee #9001005 at Facility Epsilon between April 1–3. This employee has the following clips recorded during that time period:

Sample\_Clips <- read\_csv("Output/Footage Sample.csv")  
  
Sample\_Clips %>%   
 knitr::kable(align = 'l')

| Employee\_ID | Clip\_ID | Clip\_Start | Clip\_End |
| --- | --- | --- | --- |
| 9001005 | 156421 | 2019-04-01 05:54:34 | 2019-04-01 08:34:34 |
| 9001005 | 155093 | 2019-04-01 08:40:59 | 2019-04-01 08:54:51 |
| 9001005 | 151419 | 2019-04-01 09:03:16 | 2019-04-01 11:00:30 |
| 9001005 | 153133 | 2019-04-01 11:10:09 | 2019-04-01 12:39:51 |
| 9001005 | 151088 | 2019-04-01 12:57:51 | 2019-04-01 14:06:44 |
| 9001005 | 150947 | 2019-04-02 05:56:34 | 2019-04-02 09:48:50 |
| 9001005 | 151699 | 2019-04-02 09:54:23 | 2019-04-02 12:17:15 |

We can join this to a similar table of shifts logged. This particular employee had the following shifts scheduled from April 1–3:

library(lubridate)  
  
Sample\_Shifts <- read\_csv("Output/Shift Sample.csv")  
  
Sample\_Shifts %>%   
 knitr::kable(align = 'l')

| Employee\_ID | Shift\_ID | Shift\_Start | Shift\_End |
| --- | --- | --- | --- |
| 9001005 | E050603 | 2019-04-01 06:00:00 | 2019-04-01 14:00:00 |
| 9001005 | E051303 | 2019-04-02 06:00:00 | 2019-04-02 14:00:00 |

Two 8-hour morning shifts from 6:00 AM to 2:00 PM. We can join the two tables together by ID on a messy many-to-many join, but that tells us nothing about how much they overlap (or fail to overlap) without extensive additional work.

We have a unique identifier for employee-clip and employee-shift, but what we need is a unique identifier that can be used to join the two. Fortunately, for this particular data we can *create* a unique identifier since both clips and shifts are fundamentally measures of *time*.

So, while Employee ID is not in itself unique (i.e., one employee can have multiple clips), Employee ID combined with time of day is unique. After all, a person can only be in one place at one time!

To reshape the data for joining, I created a function that takes any data frame with a start and end column and unfolds it into discrete units of time. Using the code below to create the “Interval\_Convert” function, the shift data above for employee 9001005 converts into one entry per hour of the day per shift. As a result, two 8-hour shifts get turned into 16 Employee-Hours.

library(sqldf)  
library(lubridate)  
  
Interval\_Convert <- function(DF, Start\_Col, End\_Col, Int\_Unit, Int\_Length = 1) {  
   
 Start\_Col2 <- enquo(Start\_Col)  
 End\_Col2 <- enquo(End\_Col)  
   
 Start\_Col\_Text <- deparse(substitute(Start\_Col))  
 End\_Col\_Text <- deparse(substitute(End\_Col))  
   
 Start\_End <- DF %>%  
 ungroup() %>%  
 summarize(Min\_Start = min(!!Start\_Col2),  
 Max\_End = max(!!End\_Col2)) %>%  
 mutate(Start = floor\_date(Min\_Start, Int\_Unit),  
 End = ceiling\_date(Max\_End, Int\_Unit))  
   
 DF <- DF %>%  
 mutate(Start\_Col\_Numeric = as.numeric(!!Start\_Col2),  
 End\_Col\_Numeric = as.numeric(!!End\_Col2),  
 Single = Start\_Col\_Numeric == End\_Col\_Numeric)  
   
 Interval\_Table <- data.frame(Interval\_Start = seq.POSIXt(Start\_End$Start[1], Start\_End$End[1], by = str\_c(Int\_Length, " ", Int\_Unit))) %>%  
 mutate(Interval\_End = lead(Interval\_Start),  
 Interval\_Start\_Numeric = as.numeric(Interval\_Start),  
 Interval\_End\_Numeric = as.numeric(Interval\_End)) %>%  
 filter(is.na(Interval\_End)==FALSE)  
   
 Interval\_Data\_Table<-sqldf('SELECT \* from Interval\_Table  
 LEFT JOIN DF  
 ON (Start\_Col\_Numeric>=Interval\_Start\_Numeric AND End\_Col\_Numeric<=Interval\_End\_Numeric)  
 OR (Start\_Col\_Numeric<Interval\_Start\_Numeric AND End\_Col\_Numeric>Interval\_Start\_Numeric AND End\_Col\_Numeric<=Interval\_End\_Numeric)  
 OR (Start\_Col\_Numeric>=Interval\_Start\_Numeric AND Start\_Col\_Numeric<Interval\_End\_Numeric AND End\_Col\_Numeric>Interval\_End\_Numeric)  
 OR (Start\_Col\_Numeric<Interval\_Start\_Numeric AND End\_Col\_Numeric>Interval\_End\_Numeric)') %>%  
 mutate(Seconds\_Duration\_Within\_Interval = if\_else(Start\_Col\_Numeric>=Interval\_Start\_Numeric & End\_Col\_Numeric<=Interval\_End\_Numeric, End\_Col\_Numeric - Start\_Col\_Numeric,  
 if\_else(Start\_Col\_Numeric<=Interval\_Start\_Numeric & End\_Col\_Numeric > Interval\_Start\_Numeric & End\_Col\_Numeric <= Interval\_End\_Numeric, End\_Col\_Numeric - Interval\_Start\_Numeric,  
 if\_else(Start\_Col\_Numeric>=Interval\_Start\_Numeric & Start\_Col\_Numeric<Interval\_End\_Numeric & End\_Col\_Numeric>Interval\_End\_Numeric, Interval\_End\_Numeric - Start\_Col\_Numeric,  
 if\_else(Start\_Col\_Numeric<Interval\_Start\_Numeric & End\_Col\_Numeric>Interval\_End\_Numeric, Interval\_End\_Numeric - Interval\_Start\_Numeric, Interval\_End\_Numeric - Interval\_Start\_Numeric)  
 )))  
 ) %>%  
 filter(!(Single & Interval\_End\_Numeric == Start\_Col\_Numeric)) %>%  
 select(-contains("Numeric"))  
   
 return(Interval\_Data\_Table)  
}

| Interval\_Start | Interval\_End | Employee\_ID | Shift\_ID | Shift\_Start | Shift\_End | Seconds\_Duration\_Within\_Interval |
| --- | --- | --- | --- | --- | --- | --- |
| 2019-04-01 07:00:00 | 2019-04-01 08:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 08:00:00 | 2019-04-01 09:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 09:00:00 | 2019-04-01 10:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 10:00:00 | 2019-04-01 11:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 11:00:00 | 2019-04-01 12:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 12:00:00 | 2019-04-01 13:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 13:00:00 | 2019-04-01 14:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-01 14:00:00 | 2019-04-01 15:00:00 | 9001005 | E050603 | 2019-04-01 07:00:00 | 2019-04-01 15:00:00 | 3600 |
| 2019-04-02 07:00:00 | 2019-04-02 08:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 08:00:00 | 2019-04-02 09:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 09:00:00 | 2019-04-02 10:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 10:00:00 | 2019-04-02 11:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 11:00:00 | 2019-04-02 12:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 12:00:00 | 2019-04-02 13:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 13:00:00 | 2019-04-02 14:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |
| 2019-04-02 14:00:00 | 2019-04-02 15:00:00 | 9001005 | E051303 | 2019-04-02 07:00:00 | 2019-04-02 15:00:00 | 3600 |

The footage can be converted in a similar manner, and this way I could break down both the shift data and the clip data into an hour-by-hour view and compare them to one another. Using this new format, I joined together the full tables of footage and shifts to determine how much footage was recorded with no corresponding shift in the timekeeping system, and vice-versa.

Mismatch\_Statistics <- read\_csv("Output/Footage-Shift Mismatch Summary Statistics.csv")  
  
Mismatch\_Statistics %>%   
 select(HR\_Location, Footage\_Hours\_No\_Shift, Employee\_IDs\_With\_Missing\_Shift) %>%   
 knitr::kable(align = 'l')

| HR\_Location | Footage\_Hours\_No\_Shift | Employee\_IDs\_With\_Missing\_Shift |
| --- | --- | --- |
| Alpha | 1805 | 122 |
| Beta | 3749 | 114 |
| Delta | 1208 | 133 |
| Epsilon | 2899 | 157 |
| Gamma | 4153 | 170 |

To summarize, almost every employee has footage hours that do not match with logged shifts, totaling nearly 14,000 hours. But what about the opposite? How many hours of shifts got logged with no corresponding footage?

Mismatch\_Statistics %>%   
 select(HR\_Location, Shift\_Hours\_No\_Footage, Employee\_IDs\_With\_Missing\_Footage) %>%   
 knitr::kable(align = 'l')

| HR\_Location | Shift\_Hours\_No\_Footage | Employee\_IDs\_With\_Missing\_Footage |
| --- | --- | --- |
| Alpha | 7338 | 127 |
| Beta | 6014 | 118 |
| Delta | 12830 | 141 |
| Epsilon | 9000 | 168 |
| Gamma | 11960 | 183 |

Oh dear.

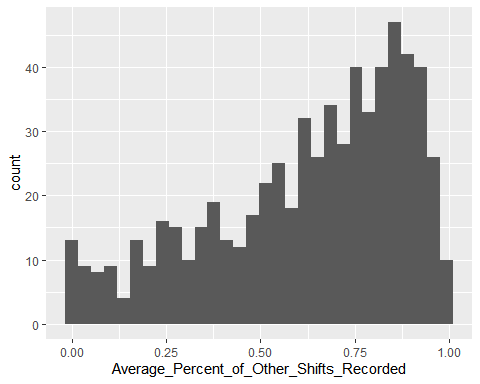
That is also almost every employee, but this time totaling about 47,000 hours. To put it another way, that’s an entire work week per employee not showing up in camera footage.

At this point, we could probably rule out deliberate noncompliance. The clip data already implied that most employees were following the policy, and our facility leadership would have noticed a mass refusal large enough to show up this clearly in the data.

One way to check for deliberate noncompliance would be to first exclude shifts that contain zero footage whatsoever. This would rule out total mismatches, where – for whatever reason – the logged shifts had totally failed to overlap with recorded clips. For those remaining shifts that *do* contain footage, we could look at the proportion of the shift covered by footage. So, if an 8-hour shift had 4 hours of recorded footage associated with it, then we could say that 50% of the shift had been recorded. The following histogram is a distribution of the number of employees organized by the percent of their shift-hours they recorded (but only shifts that had a nonzero amount of footage).

All\_Employee\_Metrics <- read\_csv("Output/All Employee Metrics.csv")  
  
All\_Employee\_Metrics %>%  
 ggplot(aes(Average\_Percent\_of\_Other\_Shifts\_Recorded)) +  
 geom\_histogram()

Warning: Removed 156 rows containing non-finite values (`stat\_bin()`).



As it turned out, most employees recorded the majority of their matching shifts, a finding that roughly aligns with the initial clip analysis.

## Causes of Failure

Here, I believed, we had reached the end of what I could do with data alone. I presented these findings (or lack thereof) to executive leadership, and the failure to gather reliable data from linking the clip data to the shift data prompted follow-ups into what exactly was going wrong. As it turned out, *many* things were going wrong.

First, a number of technical problems plagued the early roll out of the cameras.

* All of our facilities suffered from high turnover, and camera ownership was not consistently updated. Employees who no longer worked at the agency could therefore appear in the clip data – somebody else had taken over their camera but had not put their name and ID on it.
* We had no way of telling if a camera was not recording due to being docked and recharging or not recording due to being switched off.
* In the early days of the roll out, footage got assigned to an owner based on the owner of the *dock*, not the camera. In other words, if Employee A had recorded their shift but uploaded the footage using a dock assigned to Employee B then the footage would show up in the system as belonging to Employee B.

The shift data was, unsurprisingly, even worse, and it was here we came across our most important finding. While the evidence showed that there wasn’t any mass non-compliance with the use of the cameras, there *was* mass non-compliance with the use of our shift management software. To explain why we’ll need to visit the reasons we even had a shift management software separate from our HR system in the first place.

Our HR system, CAPPS, had a feature that tracked hours worked in order to calculate leave and overtime pay. However, CAPPS was a statewide application designed for 9–5 office workers, and could not capture the irregular working hours of our staff (much less aid in planning future shifts).

We had obtained the shift management software to fill these gaps, but not realized how inconsistently it was being used. First, individual facilities used the shift software differently. All were required to have their employees use it, but some followed through on this better than others. And even for the ones that did make a good-faith effort at follow-through, quality control was nonexistent. In CAPPS, time entry determined pay, so strong incentives existed to ensure accurate entry. But for our shift management software, no incentives existed at all for making sure that entries were correct. For example, a correctional officer could have a 40-hour work week scheduled in the shift software but miss the entire week due to an injury, and the software would still show them as having worked 40 hours that week. Nobody bothered to go back and correct these types of errors because there was no reason to.

## What we learned from failure

Whatever means we used to monitor compliance with the camera policy, we learned that it couldn’t be fully automated. The agency followed up this analysis with a random sampling approach, in which monitors would randomly select times of day they knew a given staff member would have to have their cameras turned on and actually watch the associated clips. This review process confirmed the first impressions from the statistical review above: most employees *were* making good faith attempts at complying with the policy despite technical glitches, short-staffing, and administrative confusion.

As for the shift management software, we had to rethink a number of aspects of its implementation. In the process, leadership also came to clarify that the software’s primary purpose was to help facilities schedule future shifts. As a result, ensuring accuracy after the fact was never something the agency would devote significant time and resources to, rendering it much less useful as a source for future data analysis.

This confirmed that proactive monitoring of correctional officers was a human process which had to come from supervisors and staff. We already had so many other checks in place, it wasn’t worth the effort to flag staff based on unreliable camera usage metrics or micromanage their after-the-fact shift logs.

The one piece of the analysis we did use going forward was the clip analysis (converted into a Power BI dashboard, included in the [GitHub repository](https://github.com/enndubbs/Body-Worn-Camera-Monitoring) for this article), but only as a supplement for already-launched investigations, not a prompt for one. Body-worn camera footage remained immensely useful for investigations after-the-fact, but inconsistencies in clip data in and of themselves were not particularly noteworthy ‘red flags’. At the end of the day, analytics can contextualize and enhance human judgment, but it cannot replace it.

In academia, the bias in favor of positive findings is well-documented. The failure to find something, or a lack of statistical significance, does not lend itself to publication in the same way that a novel discovery does. But, in an applied setting, where results matter more than publication criteria, negative findings can be highly insightful. They can falsify erroneous assumptions, bring unknown problems to light, and prompt the creation of new processes and tools. In this context, a failure is only truly a failure if nothing is learned from it.

1. https://www.texastribune.org/2018/01/26/texas-juvenile-justice-department-camille-cain/ [↑](#footnote-ref-20)
2. A major reform that came from a court decision, *Turman v. Morales* in 1973 kicked off the only stable period of the juvenile justice system run by the state. A single executive director, Ron Jackson, ran the agency for 20 years with a focus on reform and rehabilitation. In 1993, a bipartisan tough-on-crime effort resulted in his ousting and a rapid escalation of youth incarceration. This tragically led to a resumption of the historical norm of abuse scandals that defined the agency prior to 1973. [↑](#footnote-ref-21)
3. The underlying data for the analysis as presented in this article was requested through the Texas Public Information Act and went through TJJD’s approval process for ensuring anonymity of records. It is available on GitHub along with the rest of the code used to write this article. [↑](#footnote-ref-23)