HW4: Truck Factor

Nicholas Tyler

February 26, 2019

```
library(DBI)
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages ----------------------------------------------------------------------
```

```
## v tibble  2.0.1     v purrr   0.2.5
## v tidyr   0.8.2     v dplyr   0.7.8
## v readr   1.3.1     v stringr 1.3.1
## v tibble  2.0.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------------------------- t
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(dplyr)
library(dbplyr)
```

```
##
## Attaching package: 'dbplyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     ident, sql
```

The techniques used in this report for data acquisition, cleaning, and population are taken from the earlier assignments using the methods used in those assignments. The repository list is pulled from github, and will pull in descending order of stars on the repository. This means that if the report is run for 10 repositories, it will use the 10 most starred repositories on github.

Data storage and population is done via a MYSQL database, using a schema listed in the data acquisition folder of the related repository. The data is populated via a python script which makes use of the mysqlimport command line utility.

I also elected to perform the actual calculation of truck factors in python. This decision was driven largely by the nature of calculation truck factors, where you're looping over the data rather then performing a set number of operations on dataframes. The solution would likely need to be moved in the database as scale increases.

Everything above this line is used to generate a csv file used to run the truck factor calculation. If you have your own csv file, it can be substituted by running only the lines below, replacing "raw_data.txt" with the path of your csv.

```
file_factors <- read_csv("file_tfs.txt")
```

```
## Parsed with column specification:
## cols(
##   project = col_character(),
##   file = col_character(),
##   factor = col_double()
## )
```

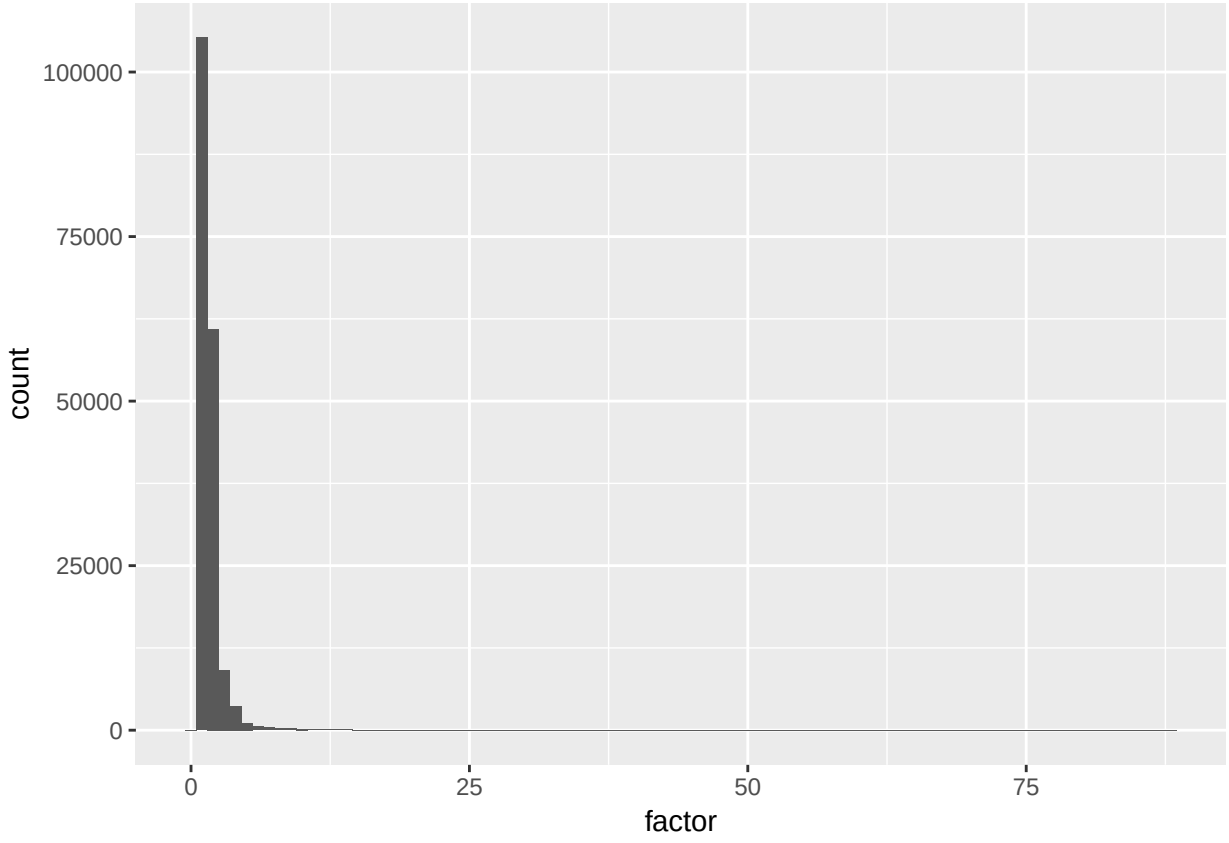
```
## Warning: 4038669 parsing failures.
##    row    col  expected                  actual           file
## 111360 NA     3 columns 4 columns               'file_tfs.txt'
## 111361 factor a double  gardener@tensorflow.org 'file_tfs.txt'
## 111361 NA     3 columns 6 columns               'file_tfs.txt'
## 111362 factor a double  gardener@tensorflow.org 'file_tfs.txt'
## 111362 NA     3 columns 6 columns               'file_tfs.txt'
## ...... ...... ......... ....................... ..............
## See problems(...) for more details.
```

```
project_factors <- read_csv("proj_tfs.txt")
```

```
## Parsed with column specification:
## cols(
##   project = col_character(),
##   factor = col_double()
## )
```

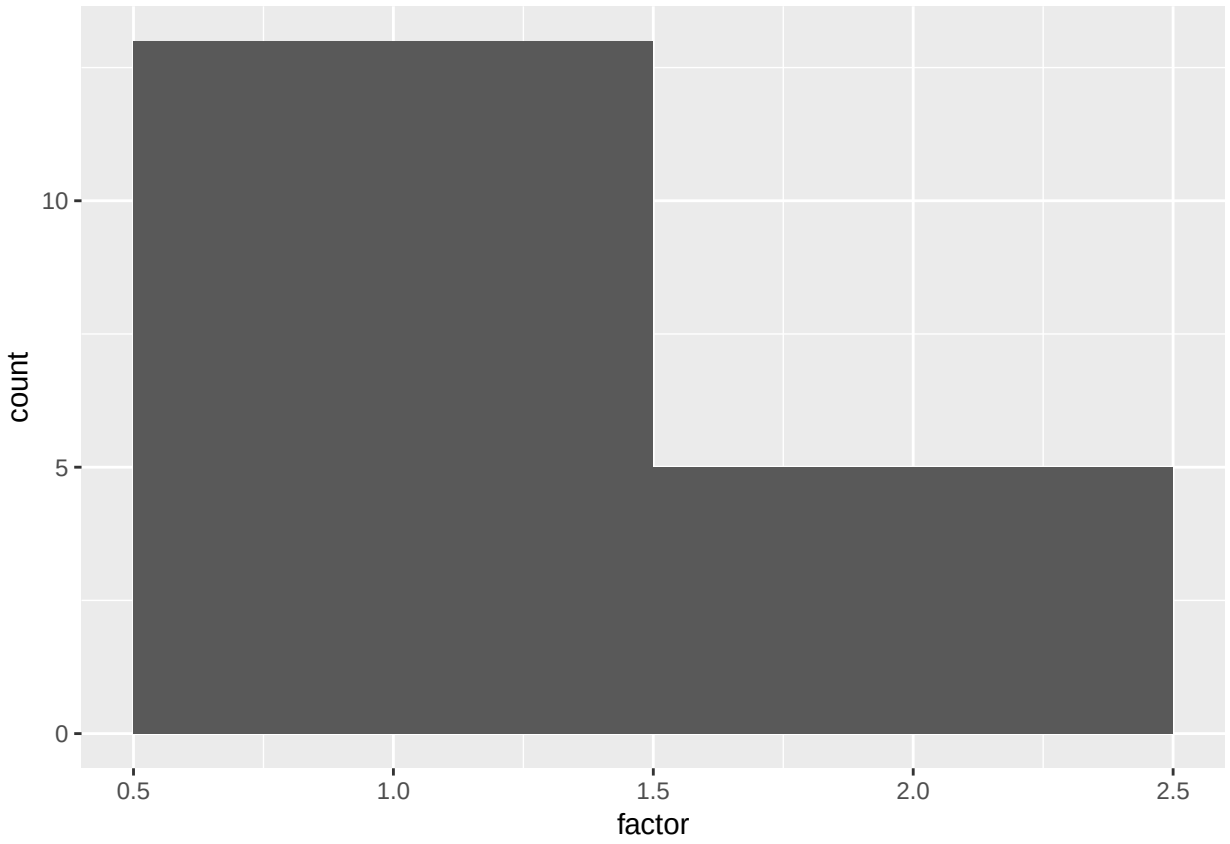
```
ggplot(file_factors) + geom_histogram(aes(x=factor), binwidth = 1)
```

```
## Warning: Removed 2019333 rows containing non-finite values (stat_bin).
```



The above chart shows the distribution of the truck factors of each of the individual files analyzed. The chart is highly concentrated around low values (primarily 1 and 2), showing that the majority of files have low truck factors.

```
ggplot(project_factors) + geom_histogram(aes(x=factor), binwidth = 1)
```



The above chart shows a similar distribution with the majority of projects having a truck factor of one, showing that the distribution is heavily weighted at a low truck factor.