Longitudinal Data Cleaning

A Case of The NLSY79 Data

Dewi Lestari Amaliah

24/02/2021

1 Introduction

Real world data set often

2 The NLSY79

3 The NLSY79 Data Cleaning

3.1 Getting and Tyding the Data

The NLYS data is stored in a database and could be downloaded by variables. Several variables are available to be downloaded and could be browsed by index. For the wages data set, we only extracted these variables:

- Education, Training & Achievement Scores
  - Education -> Summary measures -> All schools -> By year -> Highest grade completed
    - Downloaded all of the 78 variables in Highest grade completed.
- Employment
  - Summary measures -> By job -> Hours worked and Hourly wages
    - Downloaded all of the 427 variables in Hours worked
    - Downloaded all of the 151 variables in Hourly wages

    Both hours worked and hourly wages are recorded by the job, up to five jobs for each id/subject.
- Household, Geography & Contextual Variables
  - Context -> Summary measures -> Basic demographics
    - Downloaded year and month of birth, race, and sex variables.

    There are two versions of the year and month of birth, i.e. 1979 and 1981 data. We downloaded these two versions.

The downloaded data set came in a csv (NLSY79.csv) and dat (NLSY79.dat) format. We only used the .dat format. It also came along with these files:

- NLSY79.NLSY79: This is the tagset of variables that can be uploaded to the web site to recreate the data set.
- NLSY79.R: This is an R script provided automatically by the database for reading the data into R and convert the variables' name and its label into something more sensible. We utilized this code to do an initial data tidying. It produced two data set, `categories_qnames` (the observations are stored in categorical/interval values) and `new_data_qnames` (the observations are stored in integer form). In this case, we only used the latter.

```
source(here::here("data-raw/NLSY79/NLSY79.R"))
```

`new_data_qnames` is still untidy, where all of the variables for each year and each job being stored in one column, hence the data contains a huge number of columns (686 columns). Thus, the data should be tidied

and wrangled first to extract the demographic and employment variables that we want to put in the final data set. We mainly used `tidyr` (Wickham 2020), `dplyr` (Wickham et al. 2020), and `stringr` (Wickham 2019) to do this job.

3.1.1 Tidying demographic variables

`Age in 1979`, `gender`, `race`, `highest grade completed` (factor and integer), and the `year when the highest grade completed` are the variables that we want to use in the cleaned data set.

Age in 1979 are derived from year of birth and month of birth variables in the raw data set. The variables have two versions, which are the 1979 version and the 1981 version. We only used the 1979 data and did a consistency check of those years and flag the inconsistent observations.

```
## tidy the date of birth data
dob_tidy <- new_data_qnames %>%
  dplyr::select(CASEID_1979,
         starts_with("Q1-3_A~")) %>%
  mutate(dob_year = case_when(
                    # if the years recorded in both sets match, take 79 data
                    `Q1-3_A~Y_1979` == `Q1-3_A~Y_1981` ~ `Q1-3_A~Y_1979`,
                    # if the year in the 81 set is missing, take the 79 data
                    is.na(`Q1-3_A~Y_1981`) ~ `Q1-3_A~Y_1979`,
                    # if the sets don't match for dob year, take the 79 data
                    `Q1-3_A~Y_1979` != `Q1-3_A~Y_1981` ~ `Q1-3_A~Y_1979`),
         dob_month = case_when(
                    # if months recorded in both sets match, take 79 data
                    `Q1-3_A~M_1979` == `Q1-3_A~M_1981` ~ `Q1-3_A~M_1979`,
                    # if month in 81 set is missing, take the 79 data
                    is.na(`Q1-3_A~M_1981`) ~ `Q1-3_A~M_1979`,
                    # if sets don't match for dob month, take the 79 data
                    `Q1-3_A~M_1979` != `Q1-3_A~M_1981` ~ `Q1-3_A~M_1979`),
         # flag if there is a conflict between dob recorded in 79 and 81
         dob_conflict = case_when(
                      (`Q1-3_A~M_1979` != `Q1-3_A~M_1981`) & !is.na(`Q1-3_A~M_1981`)
                      ~ TRUE,
                      (`Q1-3_A~Y_1979` != `Q1-3_A~Y_1981`) & !is.na(`Q1-3_A~Y_1981`)
                      ~ TRUE,
                      (`Q1-3_A~Y_1979` == `Q1-3_A~Y_1981`) &
                      (`Q1-3_A~M_1979` == `Q1-3_A~M_1981`) ~ FALSE,
                      is.na(`Q1-3_A~M_1981`) | is.na(`Q1-3_A~Y_1981`) ~ FALSE)) %>%
  dplyr::select(CASEID_1979,
         dob_month,
         dob_year,
         dob_conflict)
```

For `gender` and `race`, we only renamed these variables.

```
## tidy the gender and race variables
demog_tidy <- categories_qnames %>%
  dplyr::select(CASEID_1979,
         SAMPLE_RACE_78SCRN,
         SAMPLE_SEX_1979) %>%
  rename(gender = SAMPLE_SEX_1979,
         race = SAMPLE_RACE_78SCRN)
```

The `highest grade completed` came with several version in each year. We chose the revised May data

because the May data seemed to have less missing and presumably the revised data has been checked. However, there was no revised May data for 2012, 2014, 2016, and 2018 so we just used the ordinary May data.

```
# tidy the grade
demog_education <- new_data_qnames %>%
  as_tibble() %>%
  # in 2018, the variable's name is Q3-4_2018, instead of HGC_2018
  rename(HGC_2018 = `Q3-4_2018`) %>%
  dplyr::select(CASEID_1979,
         starts_with("HGCREV"),
         "HGC_2012",
         "HGC_2014",
         "HGC_2016",
         "HGC_2018") %>%
  pivot_longer(!CASEID_1979,
               names_to = "var",
               values_to = "grade") %>%
  separate("var", c("var", "year"), sep = -4) %>%
  filter(!is.na(grade)) %>%
  dplyr::select(-var)
```

In the final data, we only used the highest grade completed ever and derived the year of when the highest grade completed its categorical value. Therefore, we wrangled the highest grade completed in each year to mutate these variables.

```
## getting the highest year of completed education ever
highest_year <- demog_education %>%
  group_by(CASEID_1979) %>%
  mutate(hgc_i = max(grade)) %>%
  filter(hgc_i == grade) %>%
  filter(year == first(year)) %>%
  rename(yr_hgc = year) %>%
  dplyr::select(CASEID_1979, yr_hgc, hgc_i) %>%
  ungroup() %>%
  mutate('hgc' = ifelse(hgc_i == 0, "NONE", ifelse(hgc_i == 1, "1ST GRADE",
                 ifelse(hgc_i == 2, "2ND GRADE", ifelse(hgc_i == 3, "3RD GRADE",
                 ifelse(hgc_i >= 4 & hgc_i <= 12, paste0(hgc_i,"TH GRADE"),
                 ifelse(hgc_i == 13, "1ST YEAR COL",
                 ifelse(hgc_i == 14, "2ND YEAR COL",
                 ifelse(hgc_i == 15, "3RD YEAR COL",
                 ifelse(hgc_i == 95, "UNGRADED",
                        paste0((hgc_i - 12), "TH YEAR COL")))))))))))
```

Finally, we join all the tidy variables in a data set called `full_demographics`.

```
full_demographics <- full_join(dob_tidy, demog_tidy, by = "CASEID_1979") %>%
  full_join(highest_year, by = "CASEID_1979") %>%
  rename("id" = "CASEID_1979")

head(full_demographics)
```

```
##   id dob_month dob_year dob_conflict                    race gender yr_hgc
## 1  1         9       58        FALSE NON-BLACK, NON-HISPANIC FEMALE   1979
## 2  2         1       59        FALSE NON-BLACK, NON-HISPANIC FEMALE   1985
## 3  3         8       61        FALSE NON-BLACK, NON-HISPANIC FEMALE   1993
```

```
## 4  4         8       62        FALSE NON-BLACK, NON-HISPANIC FEMALE   1986
## 5  5         7       59        FALSE NON-BLACK, NON-HISPANIC   MALE   1984
## 6  6        10       60        FALSE NON-BLACK, NON-HISPANIC   MALE   1983
##   hgc_i          hgc
## 1    12   12TH GRADE
## 2    12   12TH GRADE
## 3    12   12TH GRADE
## 4    14 2ND YEAR COL
## 5    18 6TH YEAR COL
## 6    16 4TH YEAR COL
```

3.1.2 Tidying employment variables

The employment data comprises of three variables, i.e. `total hours of work per week`, `number of jobs that an individual has`, and `mean hourly wage`. For hours worked per week, initially only one version per job, no choice from 1979 to 1987 (QES-52A). From 1988 onward, when we had more options, we chose the variable for total hours including time spent working from home (QES-52D). However, 1993 did not have all the five D variables (the first one and the last one were missing), so we used QES-52A variable instead. In addition, 2008 only had jobs 1-4 for the QES-52D variable (whereas the other years had 1-5), so we just used these.

```
# make a list for years where we used the "QES-52A"
year_A <- c(1979:1987, 1993)
#function to get the hour of work
get_hour <- function(year){
  if(year %in% year_A){
   temp <- new_data_qnames %>%
    dplyr::select(CASEID_1979,
            starts_with("QES-52A") &
              ends_with(as.character(year)))}
  else{
    temp <- new_data_qnames %>%
    dplyr::select(CASEID_1979,
            starts_with("QES-52D") &
              ends_with(as.character(year)))}
    temp %>%
      pivot_longer(!CASEID_1979,
                   names_to = "job",
                   values_to = "hours_work") %>%
      separate("job", c("job", "year"), sep = -4) %>%
      mutate(job = paste0("job_", substr(job, 9, 10))) %>%
      rename(id = CASEID_1979)
}

# list to save the iteration result
hours <- list()
# getting the hours of work of all observations
for(ayear in c(1979:1994, 1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010,
               2012, 2014, 2016, 2018)) {
   hours[[ayear]] <- get_hour(ayear)
}
# unlist the hours of work
hours_all <- bind_rows(!!!hours)
```

The same algorithm was also deployed to tidy the rate of wage by year and by ID. The difference is that the

hourly rate had only one version of each year. The hours of work and the hourly rate were then joined to calculate the number of jobs that an ID has and their mean hourly wage. Some observations had 0 in their hourly rate, which is considered as invalid value. Thus, their hourly rate set to be N.A.

```
get_rate <- function(year) {
  new_data_qnames %>%
    dplyr::select(CASEID_1979,
            starts_with("HRP") &
              ends_with(as.character(year))) %>%
    pivot_longer(!CASEID_1979, names_to = "job", values_to = "rate_per_hour") %>%
    separate("job", c("job", "year"), sep = -4) %>%
    mutate(job = paste0("job_0", substr(job, 4, 4))) %>%
    rename(id = CASEID_1979)
}
rates <- list()
for(ayear in c(1979:1994, 1996, 1998, 2000, 2002, 2004, 2006, 2008, 2010,
               2012, 2014, 2016, 2018)) {
   rates[[ayear]] <- get_rate(ayear)
}
rates_all <- bind_rows(!!!rates)
# join hours and rates variable
hours_wages <- left_join(rates_all,
                         hours_all,
                         by = c("id", "year", "job")) %>%
  # set the 0 value in rate_per_hour as NA
  mutate(rate_per_hour = ifelse(rate_per_hour == 0, NA,
                                rate_per_hour))
head(hours_wages)
```

```
## # A tibble: 6 x 5
##      id job    year  rate_per_hour hours_work
##   <int> <chr>  <chr>         <int>      <int>
## 1     1 job_01 1979            328         38
## 2     1 job_02 1979             NA         15
## 3     1 job_03 1979             NA         NA
## 4     1 job_04 1979             NA         NA
## 5     1 job_05 1979             NA         NA
## 6     2 job_01 1979            385         35
```

Since our ultimate goal is to calculate the mean hourly wage, the number of jobs is calculate based on the availability of the `rate_per_hour` information. For example, the number of jobs of ID 1, based on `hours_work`, is two. However, since the information of hourly rate of `job_02` is not available, the number of job is considered as 1.

Further, we calculated the mean hourly wage for each ID in each year using a weighted mean with the hours of work as the weight. However, there are a lot of missing value in `hours_work` variable. In that case, we only calculated the mean hourly wage based on arithmetic/regular mean method. Hence, we created a new variable to flag whether the mean hourly wage is a weighted or a regular mean. Additionally, if an ID only had one job, we directly used their hourly wages information and flagged it as an arithmetic mean.

```
# calculate number of jobs that a person has in one year
no_job <- hours_wages %>%
  filter(!is.na(rate_per_hour)) %>%
  group_by(id, year) %>%
  summarise(no_jobs = length(rate_per_hour))
```

```r
# filter the observations with available rate per hour
eligible_wages <- hours_wages %>%
  filter(!is.na(rate_per_hour)) %>%
  left_join(no_job, by = c("id", "year"))

# calculate the mean_hourly_wage
# flag1 = code 1 for weighted mean
# code 0 for arithmetic mean
mean_hourly_wage <-
  eligible_wages %>%
  group_by(id, year) %>%
  #calculate the weighted mean if the number of jobs > 1
  mutate(wages = ifelse(no_jobs == 1, rate_per_hour/100,
                        weighted.mean(rate_per_hour, hours_work, na.rm = TRUE)/100)) %>%
  #give the flag if it the weighted mean
  mutate(flag1 = ifelse(!is.na(wages) & no_jobs != 1, 1,
                        0)) %>%
  #calculate the arithmetic mean for the na
  mutate(wages = ifelse(is.na(wages), mean(rate_per_hour)/100,
                        wages)) %>%
  group_by(id, year) %>%
  summarise(wages = mean(wages),
            total_hours = sum(hours_work),
            number_of_jobs = mean(no_jobs),
            flag1 = mean(flag1)) %>%
  mutate(year = as.numeric(year)) %>%
  ungroup() %>%
  rename(mean_hourly_wage = wages) %>%
  mutate(is_wm = ifelse(flag1 == 1, TRUE,
                        FALSE)) %>%
  dplyr::select(-flag1)

head(mean_hourly_wage, n = 10)
```

```
## # A tibble: 10 x 6
##       id  year mean_hourly_wage total_hours number_of_jobs is_wm
##    <int> <dbl>            <dbl>       <int>          <dbl> <lgl>
##  1     1  1979             3.28          38              1 FALSE
##  2     1  1981             3.61          NA              1 FALSE
##  3     2  1979             3.85          35              1 FALSE
##  4     2  1980             4.57          NA              1 FALSE
##  5     2  1981             5.14          NA              1 FALSE
##  6     2  1982             5.71          35              1 FALSE
##  7     2  1983             5.71          NA              1 FALSE
##  8     2  1984             5.14          NA              1 FALSE
##  9     2  1985             7.71          NA              1 FALSE
## 10     2  1986             7.69          NA              1 FALSE
```

The `mean_hourly_wage` and `full_demographic` data are then joined. We also filtered the data to only have the cohort who completed the education up to 12th grade and participated at least five rounds in the survey and save it to an object called `wages_demog_hs`.

```r
# join the wages information and the demographic information by case id.
wages_demog <- left_join(mean_hourly_wage, full_demographics, by="id")
# calculate the years in work force and the age of the subjects in 1979
```

```r
wages_demog <- wages_demog %>%
  mutate(yr_hgc = as.numeric(yr_hgc)) %>%
  mutate(years_in_workforce = year - yr_hgc) %>%
  mutate(age_1979 = 1979 - (dob_year + 1900))
# filter only the id with high school education
wages_demog_hs <- wages_demog  %>% filter(grepl("GRADE", hgc))
# calculate the number of observation
keep_me <- wages_demog_hs %>%
  count(id) %>%
  filter(n > 4)
wages_demog_hs <- wages_demog_hs %>%
  filter(id %in% keep_me$id)
```

3.2 Initial Data Analysis

According to Huebner, Vach, and Cessie (2016), Initial Data Analysis (IDA) is the step of inspecting and screening the data after being collected to ensure that the data is clean, valid, and ready to be deployed in the later formal statistical analysis. Moreover, Chatfield (1985) argued that the two main objectives of IDA is data description, which is to assess the structure and the quality of the data; and model formulation without any formal statistical inference.

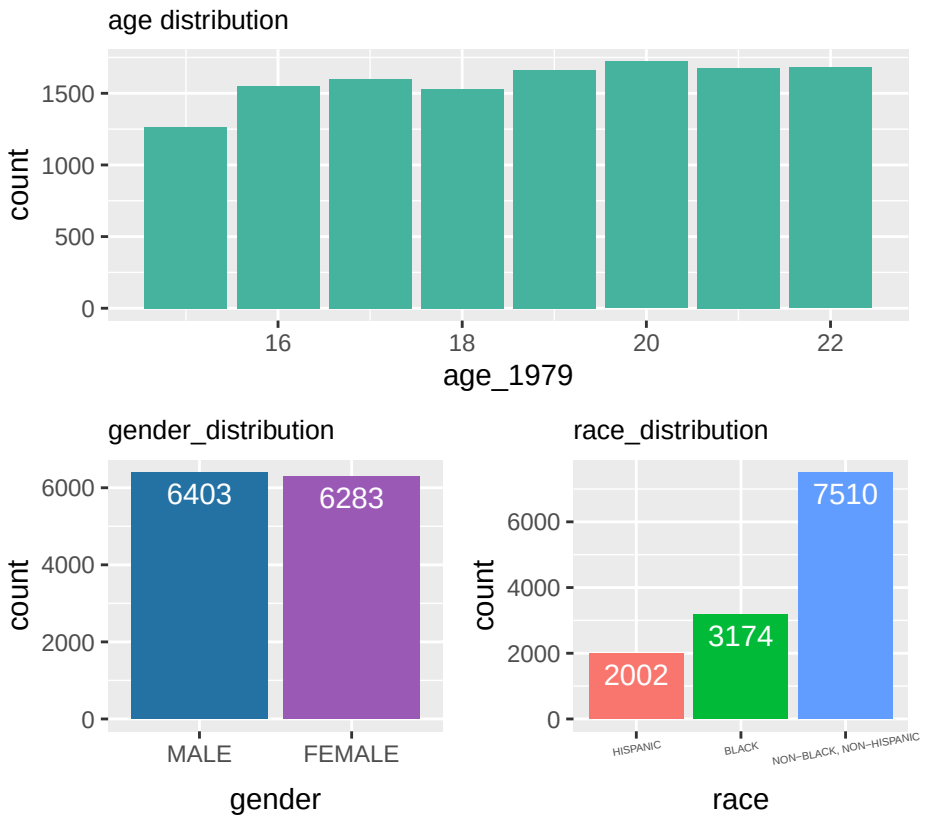

Figure 1: Demography Profile of the NLSY79 Cohort

In this paper, we conducted an IDA or a preliminary data analysis to assess the consistency of the data with the cohort information that is provided by the NLSY. In addition, we also aimed to find the anomaly in the

wages values using this approach. We mainly used graphical summary to do the IDA using `ggplot2`(Wickham 2016) and `brolgar` (Tierney, Cook, and Prvan 2020).

As stated previously, the respondents' ages ranged from 12 to 22 when first interviewed in 1979. Hence, we would like to validate whether all of the respondents were in this range. Additionally, the NLSY also provided the number of the survey cohort by their gender (6,403 males and 6,283 females) and race (7,510 Non-Black/Non-Hispanic; 3,174 Black; 2,002 Hispanic). To validate this, we used the `full_demographic` i.e. the data with the survey years 1979 sample. Figure 1 suggests that the demographic data we had is consistent with the sample information in the database.

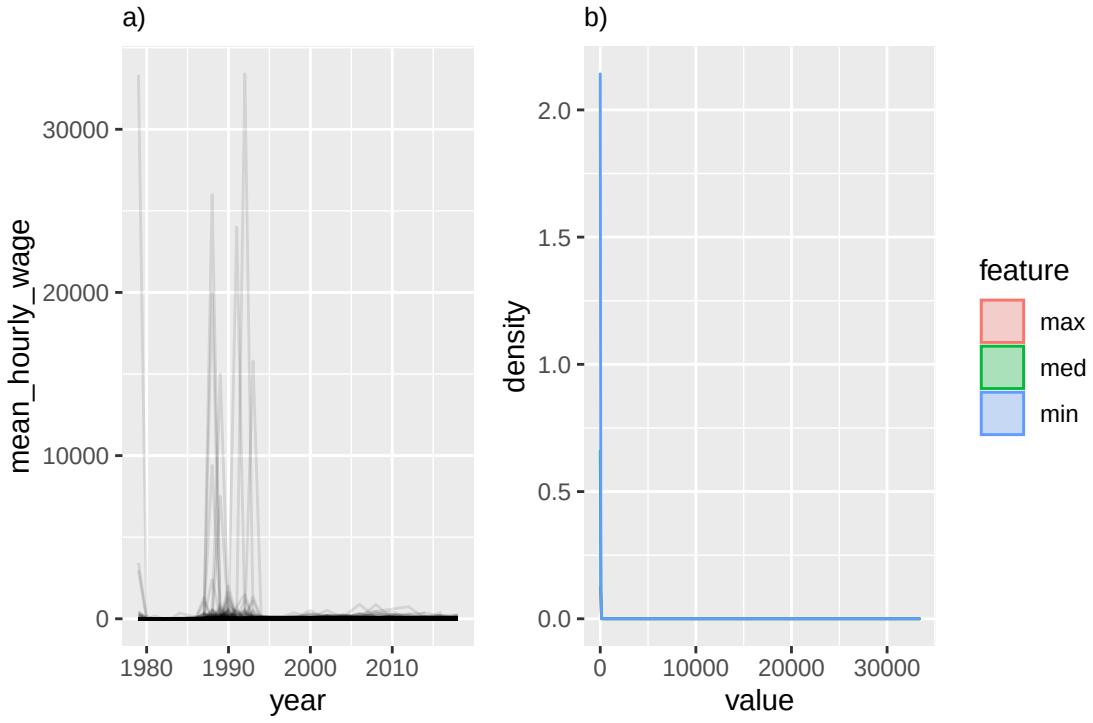

Figure 2: a) Pattern of Mean Hourly Wage of High School Cohort; b) The Three Features of the Cohort

The next step is that we explored the mean hourly wage data, in this case, we only explored the wages data in `wages_demog_hs`. Figure 2 conveys that there is clearly a problem in the mean hourly wage values. Figure 2 a.) shows that some observations had an exceptionally high figure of wages, even more than US$10,000 per hour. In Figure 2 b.), we barely see any difference in the minimum, median, an maximum value of the wages since the distribution is heavily skewed to the right. Additionally, Table 1 shows that the overall wages median of the cohort is only 7.2, while the mean is 11.87. It indicates that the data might contains a lot of extreme values.

Further, we took 36 samples randomly from the data and plotted it as is seen in Figure 3. It implies that not only that some observations earned an extremely high figures of wages, but some also had a very fluctuate wages, fo example in panel number 5, 7, and 11.

3.2.1 Robust Linear Model for Noises Treatment

As it is seen from figure 3, there are many spikes in the mean hourly wage data. As part of the IDA, which is the model formulation, we built a robust linear regression model to address this issue. The notion of robust linear regression is to yield an estimation that is robust to the influence of noise or contamination (Koller 2016). It also aims to detect the contamination by weighting each observation based on how "well-behave"

Table 1: Summary Statistics of Wages of High School Data

Statistics	Value
Min.	0.01000
1st Qu.	4.50000
Median	7.20000
Mean	11.86578
3rd Qu.	11.74000
Max.	33400.00000

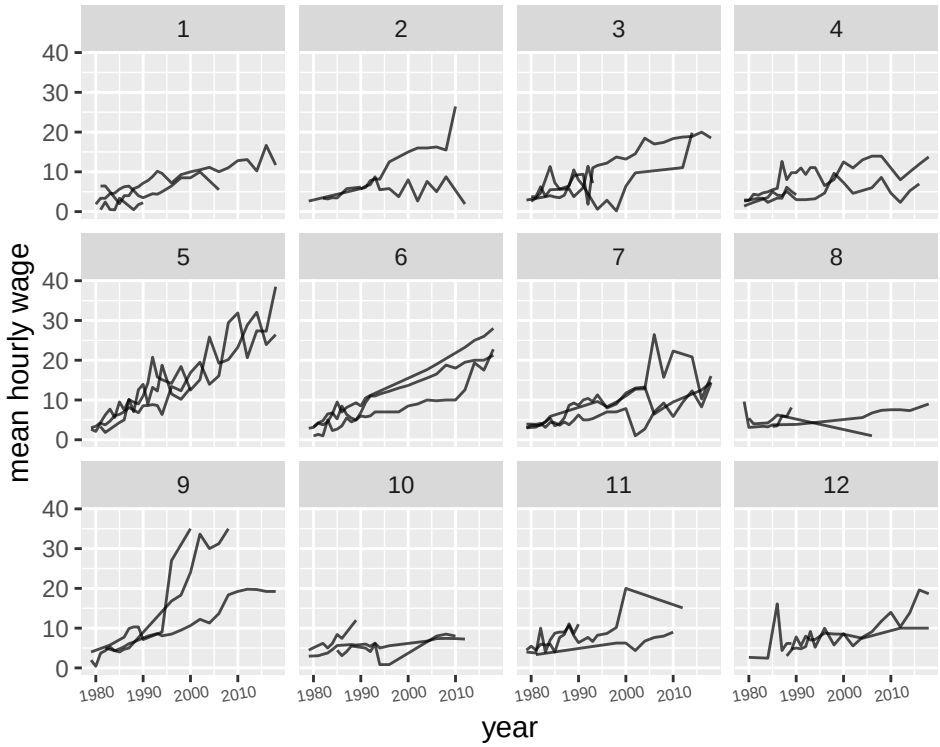

Figure 3: Some samples of observations

they are, known as robustness weight. Observations with lower robustness weight are suggested as an outliers by this method (Koller 2016).

In this paper, we built the model using the `rlm` function from `MASS` package (Venables and Ripley 2002). We set the `mean_hourly_wage` and `year` as the dependent and predictor respectively. Furthermore, we used M-Estimation with Huber weighting where the observation with small residual get a weight of 1, while the larger the residual, the smaller the weight (less than 1) (UCLA: Statistical Consulting Group 2021).

Since we worked with longitudinal data, we should built the model for each ID, instead of the overall data. The robust mixed model is actually the best model to be employed in this case. However, this method is too computationally and memory expensive, especially for a large data set, like the NLSY79 data. Thus, the model for each ID is built utilizing the `nest` and `map` function from `tidyr` (Wickham 2020) and `purrr` (Henry and Wickham 2020) respectively. After getting the weight values, we flagged the observations with the weight less than 1, and imputed their mean hourly wage with the models' predicted value.

Figure 4 a) shows that after imputing the outliers with the models' predicted value, the highest wages value is not exceed US$200. The spikes were still observed, but are not as extreme as the original data. In Figure 4 b), although the distributions of the features are still positively skewed, we can still examine it clearly. We also learned that some IDs' have a minimum wages that is higher than others' maximum wages.

```
# nest the data by id to build a robust linear model
by_id <- wages_demog_hs %>%
  dplyr::select(id, year, mean_hourly_wage) %>%
  group_by(id) %>%
  nest()

# build a robust linear model
id_rlm <- by_id %>%
  mutate(model = map(.x = data,
                     .f = function(x){
                       rlm(mean_hourly_wage ~ year, data = x)
                     }))
# extract the property of the regression model
id_aug <- id_rlm %>%
  mutate(augmented = map(model, broom::augment)) %>%
  unnest(augmented)

# extract the weight of each observation
id_w <- id_rlm %>%
  mutate(w = map(.x = model,
                 .f = function(x){
                   x$w
                 })) %>%
  unnest(w) %>%
  dplyr::select(w)

# bind the property of each observation with their weight
id_aug_w <- cbind(id_aug, id_w) %>%
  dplyr::select(`id...1`,
                year,
                mean_hourly_wage,
                .fitted,
                .resid,
                .hat,
                .sigma,
                w) %>%
```

```
  rename(id = `id...1`)

# if the weight < 1, the mean_hourly_wage is replaced by the model's fitted/predicted value.
# and add the flag whether the observation is predicted value or not.
# since the fitted value is sometimes <0, and wages value could never be negative,
# we keep the mean hourly wage value even its weight < 1.

wages_rlm_dat <- id_aug_w %>%
  mutate(wages_rlm = ifelse(w != 1 & .fitted >= 0, .fitted,
                            mean_hourly_wage)) %>%
  mutate(is_pred = ifelse(w != 1 & .fitted >= 0, TRUE,
                          FALSE)) %>%
  dplyr::select(id, year, wages_rlm, is_pred)

# join back the `wages_rlm_dat` to `wages_demog_hs`

wages_demog_hs <- left_join(wages_demog_hs, wages_rlm_dat, by = c("id", "year"))
```

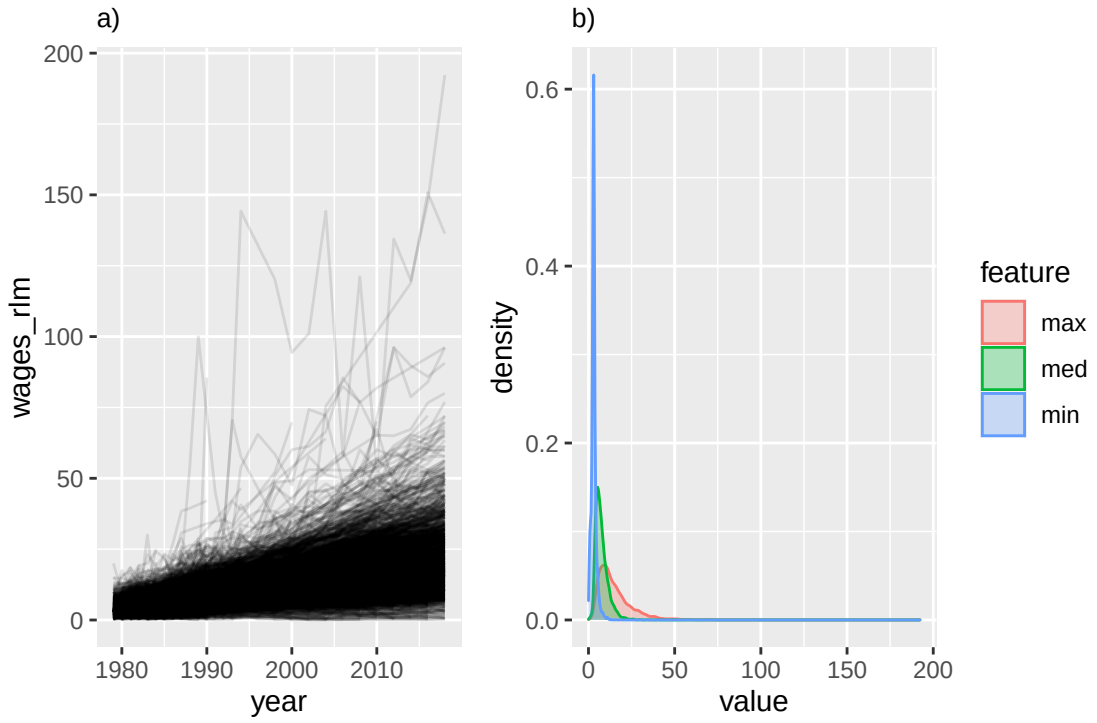

Figure 4: a) The Wages and b) The Feature Distribution after the Noises Treatment

Furthermore, the robust linear regression has reduce the level of noise in the data set as is seen in Figure 5. The result is that we got a data set with the reasonable degree of fluctuation.

Finally, we saved the imputed data and set the appropriate data type for the variables. We also saved the NLSY79 cohort's demographic information and the high school dropout cohort in a separate data sets.

```
# select out the old value of mean hourly wage and change it with the wages_rlm value
wages_demog_hs <- wages_demog_hs %>%
  dplyr::select(-mean_hourly_wage) %>%
```

Figure 5: Comparison between the Original and Treated Mean Hourly Wage

```
  rename(mean_hourly_wage = wages_rlm)

# rename and select the wages in tidy
wages_hs2020 <- wages_demog_hs %>%
  dplyr::select(id, year, mean_hourly_wage, age_1979, gender, race, hgc, hgc_i, yr_hgc,
                number_of_jobs, total_hours, is_wm, is_pred) %>%
  mutate(hgc = as.factor(hgc),
         year = as.integer(year),
         age_1979 = as.integer(age_1979),
         yr_hgc = as.integer(yr_hgc),
         number_of_jobs = as.integer(number_of_jobs))

# Create a data set for demographic variables
demographic_nlsy79 <- full_demographics %>%
  mutate(age_1979 = 1979 - (dob_year + 1900)) %>%
  dplyr::select(id,
         age_1979,
         gender,
         race,
         hgc,
         hgc_i,
         yr_hgc) %>%
  mutate(age_1979 = as.integer(age_1979),
         hgc = as.factor(hgc),
         yr_hgc = as.integer(yr_hgc))
```

```r
# Create a data set for the high school dropouts cohort
wages_hs_dropout <- wages_hs2020 %>%
  mutate(dob = 1979 - age_1979,
         age_hgc = yr_hgc - dob) %>%
  filter((hgc %in% c("9TH GRADE",
                     "10TH GRADE",
                     "11TH GRADE")) |
           (hgc == "12TH GRADE" &
              age_hgc >= 19)) %>%
  dplyr::select(-dob,
         -age_hgc)
```

4 Summary

Reference

Chatfield, C. 1985. "The Initial Examination of Data." *Journal of the Royal Statistical Society. Series A. General* 148 (3): 214–53.

Henry, Lionel, and Hadley Wickham. 2020. *Purrr: Functional Programming Tools*. https://CRAN.R-project.org/package=purrr.

Huebner, PhD, Marianne, Dr rer. nat Vach Werner, and PhD le Cessie Saskia. 2016. "A Systematic Approach to Initial Data Analysis Is Good Research Practice." *The Journal of Thoracic and Cardiovascular Surgery* 151 (1): 25–27.

Koller, Manuel. 2016. "Robustlmm: An R Package for Robust Estimation of Linear Mixed-Effects Models." *Journal of Statistical Software* 75 (6): 1–24.

Tierney, Nicholas, Di Cook, and Tania Prvan. 2020. *Brolgar: BRowse over Longitudinal Data Graphically and Analytically in R*. https://github.com/njtierney/brolgar.

UCLA: Statistical Consulting Group. 2021. "Robust Regression | R Data Analysis Examples." February 2021. https://stats.idre.ucla.edu/r/dae/robust-regression/.

Venables, W. N., and B. D. Ripley. 2002. *Modern Applied Statistics with S*. Fourth. New York: Springer. http://www.stats.ox.ac.uk/pub/MASS4.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

———. 2019. *Stringr: Simple, Consistent Wrappers for Common String Operations*. https://CRAN.R-project.org/package=stringr.

———. 2020. *Tidyr: Tidy Messy Data*. https://CRAN.R-project.org/package=tidyr.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2020. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.