Codealong – Lecture 4

Intro to Data Science for Public Policy, Spring 2016

by Jeff Chen & Dan Hammer, Georgetown University McCourt School of Public Policy

Contents

Demo

To put EDA into context, we will rely upon the American Community Survey (ACS), which is one of the most relied upon public data sources in the United States. A survey that is produced by the U.S. Census Bureau, the ACS provides a highly detailed socioeconomic snapshot of households and communities, allowing for data-driven insight to inform public policy as well as business decisions. The data dictionary containing variable definitions and descriptions can be found here.

Note: While each record is associated with a sampling weight, meaning that one record represents more than one record. For simplicity, we will treat each record with equal weight. We will also only focus on one state in this exercise – in this case, we have selected Iowa. The same analysis can be easily replicated for other states if not the entire United States.

Get data

To start, we will need to retrieve data from the Census website: "https://www2.census.gov/programs-surveys/acs/data/pums/2015/1-Year/csv_pia.zip".

First, we create a temporary file using `tempfile()`, which will be used to hold a the zipfile. Then, we will need to use the `download.file()` method, specifying the `mode = "wb"` to ensure the file downloads the binary values – a simple trick to get around security problems associated with "https". We download the zipfile to the `temp` file location.

```
temp <- tempfile()
url <- "https://www2.census.gov/programs-surveys/acs/data/pums/2015/1-Year/csv_pia.zip"
download.file(url, temp, mode="wb")
```

Now that the data has been downloaded, we will need to unzip the file using `unzip()`, and place it in our current working drive `getwd()`. As the dataset is a .csv, we can use the `read.csv()` method to import the data.

```
unz <- unzip(temp, exdir=getwd())
acs <- read.csv(unz[1])
```

Examine data structure

Now, let's take a look at the structure of the first 20 variables using the '`str()` method. "PUMA" and "ST" are codes for Public Use Microdata Areas and States, both of which are geographic units, yet the data represents them as integers. These variables as well as CIT (citizenship), SEX (sex) and a few others are also coded as integers although they represent factors. We will need to clean these up before using them for visual analysis.

```r
str(acs[,1:20])
```

```
## 'data.frame':    31900 obs. of  20 variables:
##  $ RT      : Factor w/ 1 level "P": 1 1 1 1 1 1 1 1 1 1 ...
##  $ SERIALNO: int  141 215 215 215 227 262 312 312 327 327 ...
##  $ SPORDER : int  1 1 2 3 1 1 1 2 1 2 ...
##  $ PUMA    : int  1800 1500 1500 1500 200 1400 100 100 2100 2100 ...
##  $ ST      : int  19 19 19 19 19 19 19 19 19 19 ...
##  $ ADJINC  : int  1001264 1001264 1001264 1001264 1001264 1001264 1001264 1001264 1001264 1001264 ..
##  $ PWGTP   : int  22 41 44 52 24 43 100 84 74 142 ...
##  $ AGEP    : int  89 52 57 19 68 89 61 59 66 66 ...
##  $ CIT     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ CITWP   : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ COW     : int  NA 3 3 NA 1 6 6 1 NA NA ...
##  $ DDRS    : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ DEAR    : int  2 2 2 1 2 2 2 2 1 2 ...
##  $ DEYE    : int  2 2 2 2 2 2 2 2 2 2 ...
##  $ DOUT    : int  2 2 2 1 2 2 2 2 2 2 ...
##  $ DPHY    : int  2 2 2 2 2 2 2 2 1 1 ...
##  $ DRAT    : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ DRATX   : int  NA NA 2 NA 2 2 NA NA 2 NA ...
##  $ DREM    : int  2 2 2 1 2 2 2 2 2 2 ...
##  $ ENG     : int  NA NA NA NA NA NA NA NA NA NA ...
```

Extract variables and clean Up

To make the analysis a bit more manageable, we will extract 14 demographic variables, limit the analysis to ages 16 and above.

```r
var_list <- c("HICOV","RAC1P","MAR","SEX","ESR","CIT","AGEP","PINCP","POVPIP","WKHP","SCHL")
df <- acs[acs$AGEP>=16, var_list]

#Healthcare coverage (target variable):
#Create one binary variable for calculations
df$coverage[df$HICOV == 1] <- 0
df$coverage[df$HICOV == 2] <- 1

#another with characters
df$hicov2[df$HICOV == 1] <- "With Healthcare"
df$hicov2[df$HICOV == 2] <- "Without Healthcare"

#Gender
df$sex2[df$SEX == 1] <- "Male"
df$sex2[df$SEX == 2] <- "Female"

#Race
df$race2[df$RAC1P == 1] <- "White alone"
df$race2[df$RAC1P == 2] <- "Black or African Amer. alone"
df$race2[df$RAC1P == 3] <- "Amer. Indian alone"
df$race2[df$RAC1P == 4] <- "Alaska Native alone"
df$race2[df$RAC1P == 5] <- "Amer. Indian + Alaska Nat. tribes"
df$race2[df$RAC1P == 6] <- "Asian alone"
df$race2[df$RAC1P == 7] <- "Nat. Hawaiian + Other Pac. Isl."
```

```
  df$race2[df$RAC1P == 8] <- "Some other race alone"
  df$race2[df$RAC1P == 9] <- "Two or more"

  #Marital Status
  df$mar2[df$MAR == 1] <- "Married"
  df$mar2[df$MAR == 2] <- "Widowed"
  df$mar2[df$MAR == 3] <- "Divorced"
  df$mar2[df$MAR == 4] <- "Separated"
  df$mar2[df$MAR == 5] <- "Never Married"

  #Employment Status
  df$esr2[df$ESR %in% c(1, 2, 4, 5)] <- "Employed"
  df$esr2[df$ESR == 3] <- "Unemployed"
  df$esr2[df$ESR == 6] <- "Not in labor force"

  #Citizenship
  df$cit2[df$CIT %in% c(1, 2, 3, 4)] <- "Citizen"
  df$cit2[df$CIT == 5] <- "Not citizen"

  #School
  df$schl2[df$SCHL<16 ] <- "Less than HS"
  df$schl2[df$SCHL>=16 & df$SCHL<21] <- "HS Degree"
  df$schl2[df$SCHL==21] <- "Undergraduate Degree"
  df$schl2[df$SCHL>21] <- "Graduate Degree"
```

Making sense of discrete variables

With the data in place, we now can run a few basic cross-tabulations. We'll compare education attainment against healthcare coverage using the `table()` function to produce a tabulation by combination of levels. Then, use that table to feed into the `prop.table()` method, specifying 1 to indicate that we would like to see numbers presented as proportions of each row. We then will use the `chisq.test()` method to determine if people who have health care have statistically different levels of educational attainment.

In the proportions table, we can see that 9.2% of people with less than a HS degree are without healthcare, which is notably more than the others. We also can see from the chi-squared test, used to determine if there is a significant difference between observed and expected frequencies in one or more categories. In this case, we're comparing if the distribution of education attainment are proportional among those who have and do not have healthcare coverage. Based on the test statistic, we see that the p-value is less than 0.01, allowing us to conclude that education attainment is different among those who have and do not have healthcare coverage.

```
##Comparison of education attainment vs. healthcare coverage
  tab <- table(df$schl2, df$hicov2)

  #Get proportions by row
  prop.table(tab, 1)
```

```
##
##                         With Healthcare Without Healthcare
##   Graduate Degree             0.98825602         0.01174398
##   HS Degree                   0.95367693         0.04632307
##   Less than HS                0.90847578         0.09152422
##   Undergraduate Degree        0.97780713         0.02219287
```

```r
  #Chi-square
  chisq.test(tab)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 229.61, df = 3, p-value < 2.2e-16
```

Using this basic process, we can easily loop through all the categorical variables in our abridged dataset.

```r
#Set the variables
  master <- data.frame(var = c("esr2", "mar2", "race2", "sex2", "schl2", "cit2"),
                       descrip = c("Employment", "Marital Status",
                                   "Race", "Sex", "Education", "Citizenship"))
  master[,1] <- as.character(master[,1])
  master[,2] <- as.character(master[,2])

#Loop through each variable and print result
  for(i in 1:nrow(master)){
    print(master[i, 2])
    tab <- table(df[, master[i, 1]], df$hicov2)
    print(prop.table(tab, 1))
    print(chisq.test(tab))
  }
```

```
## [1] "Employment"
##
##                       With Healthcare Without Healthcare
##   Employed                 0.95632330         0.04367670
##   Not in labor force       0.95899153         0.04100847
##   Unemployed               0.83806344         0.16193656
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 192.94, df = 2, p-value < 2.2e-16
##
## [1] "Marital Status"
##
##                  With Healthcare Without Healthcare
##   Divorced             0.92605887         0.07394113
##   Married              0.97285345         0.02714655
##   Never Married        0.91821862         0.08178138
##   Separated            0.82375479         0.17624521
##   Widowed              0.98907104         0.01092896
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 511.98, df = 4, p-value < 2.2e-16
##
## [1] "Race"
##
##                                     With Healthcare Without Healthcare
```

```
##   Amer. Indian + Alaska Nat. tribes      0.82352941         0.17647059
##   Amer. Indian alone                     0.75308642         0.24691358
##   Asian alone                            0.90291262         0.09708738
##   Black or African Amer. alone           0.88560158         0.11439842
##   Nat. Hawaiian + Other Pac. Isl.        0.66666667         0.33333333
##   Some other race alone                  0.79310345         0.20689655
##   Two or more                            0.92549020         0.07450980
##   White alone                            0.95898771         0.04101229
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 300.79, df = 7, p-value < 2.2e-16
##
## [1] "Sex"
##
##          With Healthcare Without Healthcare
##   Female      0.96413470         0.03586530
##   Male        0.94459768         0.05540232
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 56.594, df = 1, p-value = 5.357e-14
##
## [1] "Education"
##
##                        With Healthcare Without Healthcare
##   Graduate Degree           0.98825602         0.01174398
##   HS Degree                 0.95367693         0.04632307
##   Less than HS              0.90847578         0.09152422
##   Undergraduate Degree      0.97780713         0.02219287
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 229.61, df = 3, p-value < 2.2e-16
##
## [1] "Citizenship"
##
##               With Healthcare Without Healthcare
##   Citizen          0.95791252         0.04208748
##   Not citizen      0.77983539         0.22016461
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  tab
## X-squared = 344.67, df = 1, p-value < 2.2e-16
```

Making sense of continuous variables

Continuous variables offer a lot more opportunity to visualize patterns. Typically, two general types of graphs are common: univariate distribution graphs and bivariate graphs.

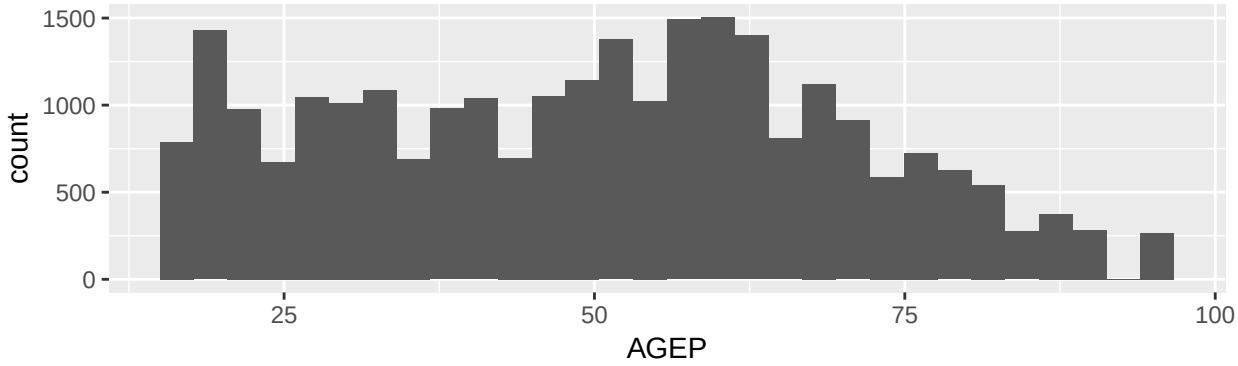
Univariate graphs

Two univariate graphs are typically used: (1) histograms, and (2 kernel density diagrams.

Histograms are a way to depict the distribution of continuous variables. For values between the minimum and maximum value of a variable, the data is partitioned into equal intervals or "bins". For each equal interval, a count is taken of the number of records in each bin. Using the `AGEP` variable (age), we can plot a histogram using ggplot2.

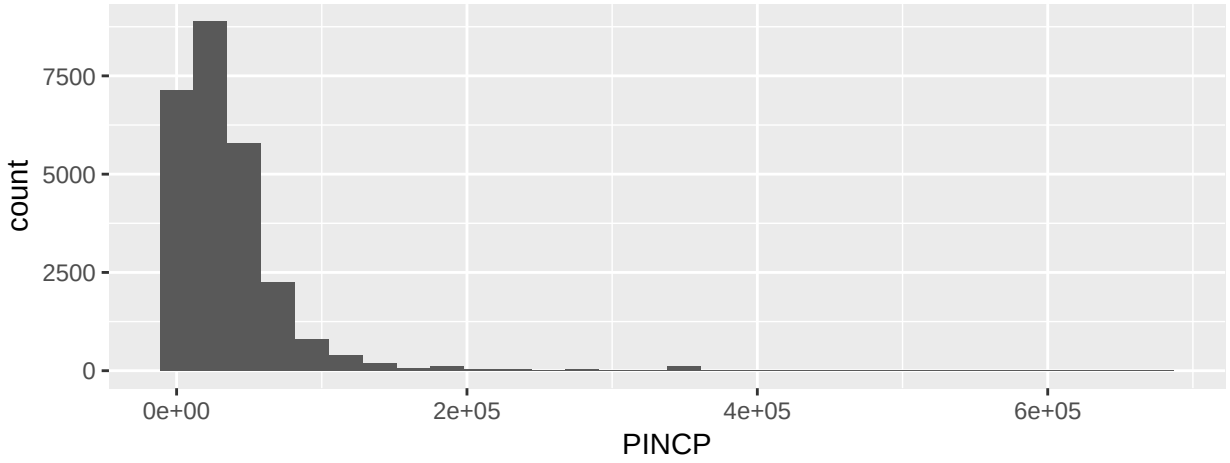
In `ggplot2`, we use the `ggplot()` method to specify the data frame and variables that will be used to create graphs, then add various `geom` arguments to specify the type of graph. For a histogram, we specify the dataset `df` and the x variable `aes(x = AGEP)`, then tag on `+geom_histogram()`.

```
library(ggplot2)
ggplot(df, aes(x = AGEP)) + geom_histogram()
```



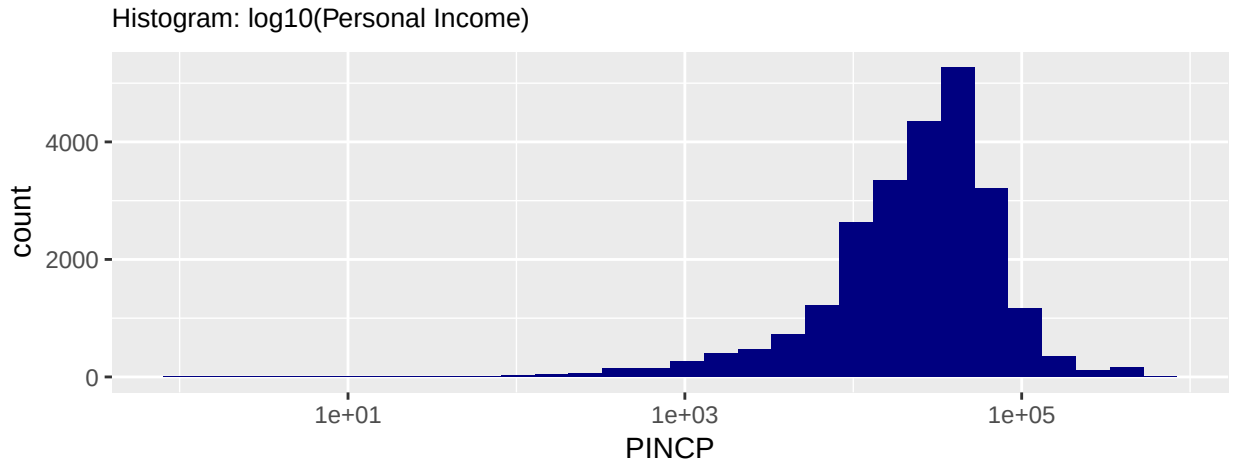
`AGEP` can be replaced with `PINCP`(personal income). Notice that the distribution occupies only a small part of the graph.

```
ggplot(df, aes(x = PINCP)) + geom_histogram()
```



This can be adjusted by adding the `+ scale_x_log10()` argument in order to transform the x variable using a log_{10} transformation. Within the `geom_histogram()` arguement, we change the color of the histogram to "navy", add a title using `ggtitle()` and adjust the size of the label text using `theme()`.

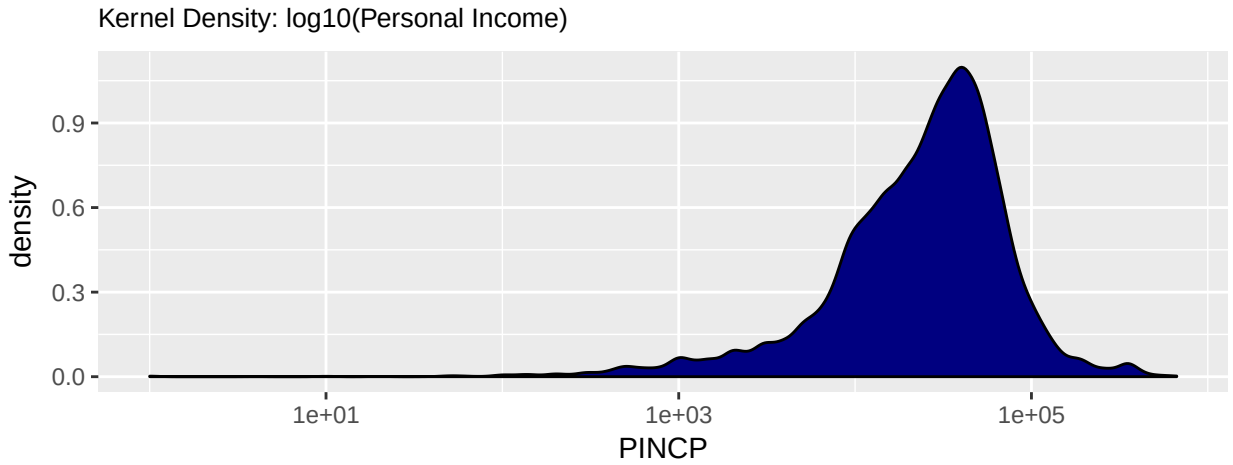
```
ggplot(df, aes(x =  PINCP)) + geom_histogram(fill = "navy") + scale_x_log10() +
  ggtitle("Histogram: log10(Personal Income)") +
  theme(plot.title = element_text(size=10))
```



Another type of univariate graph is the kernel density graph. Whereas the histogram is boxy, the kernel density graph applies a rolling window approach to calculate the count of values at small but equally spaced intervals, weighing the count using a bandwidth – a threshold that determines how wide or narrow the rolling window should be.

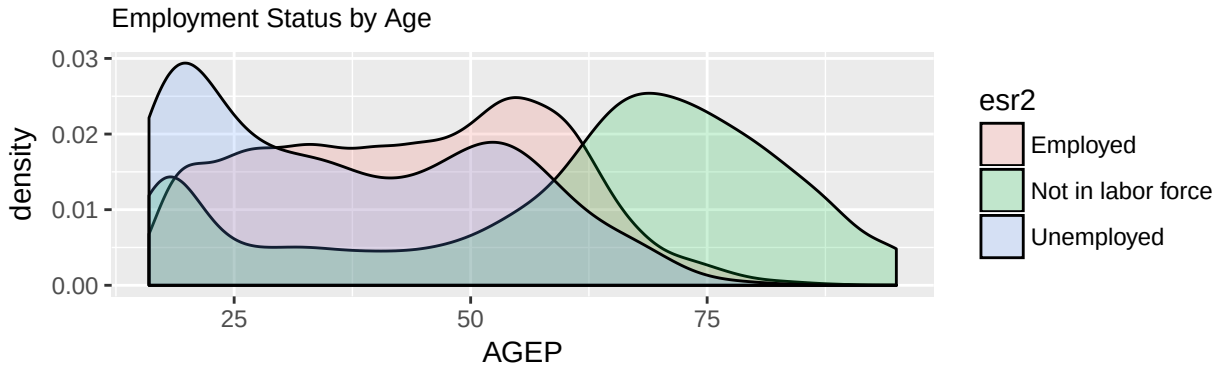
`ggplot` provides facility to use a kernel density graph using the `+ geom_density()`. Kernel density diagrams provide a more organic representation of a distribution and are more sensitive to small fluctuations in the frequencies. This is useful for finding more discrete natural breaks in the data that can be used for classification. Furthermore, kernel densities are generally easier to compare when overlaying two more or more distributions.

```
ggplot(df, aes(x = PINCP)) + geom_density(fill = "navy") + scale_x_log10() +
  ggtitle("Kernel Density: log10(Personal Income)") +
  theme(plot.title = element_text(size=10))
```



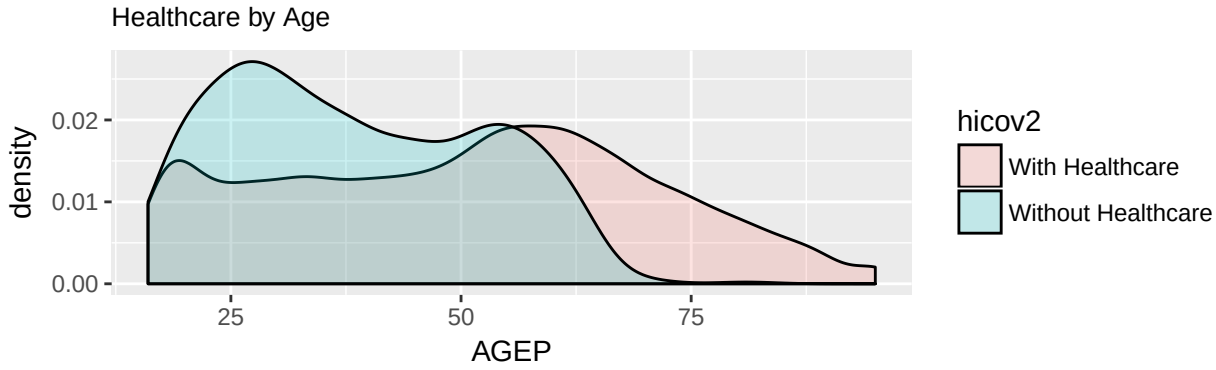
To compare two or more groups, simply set the `colours =` and `fill =` arguments to a categorical variable. For example, we can compare the ages of Iowans by employment status. As may be expected, most Iowans who are not in the labor force (not seeking employment) are past retirement age while there is a slight bump in unemployment among younger members of society.

```
ggplot(df, aes(x = AGEP, colours = esr2, fill = esr2)) +
  geom_density(alpha = 0.2) +
  ggtitle("Employment Status by Age") +
  theme(plot.title = element_text(size=10))
```



Likewise, comparing healthcare coverage by age, younger Iowans are a greater proportion of people without healthcare.

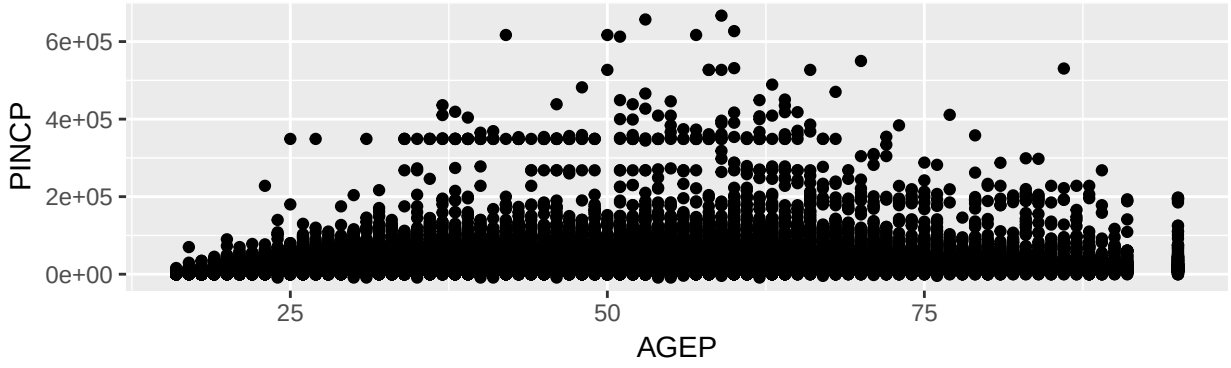
```
ggplot(df, aes(x = AGEP, colours = hicov2, fill = hicov2)) +
  geom_density(alpha = 0.2) +
  ggtitle("Healthcare by Age") +
  theme(plot.title = element_text(size=10))
```



Bivariate graphs

Scatter plots are among the most standard bivariate graphs and can be easily called by adding `geom_point()` to the base `ggplot()` method. They also can be easily stylized. Below, we plot `x = AGEP` and `y = PINCP`. The result is a mess of points without a clear trend.

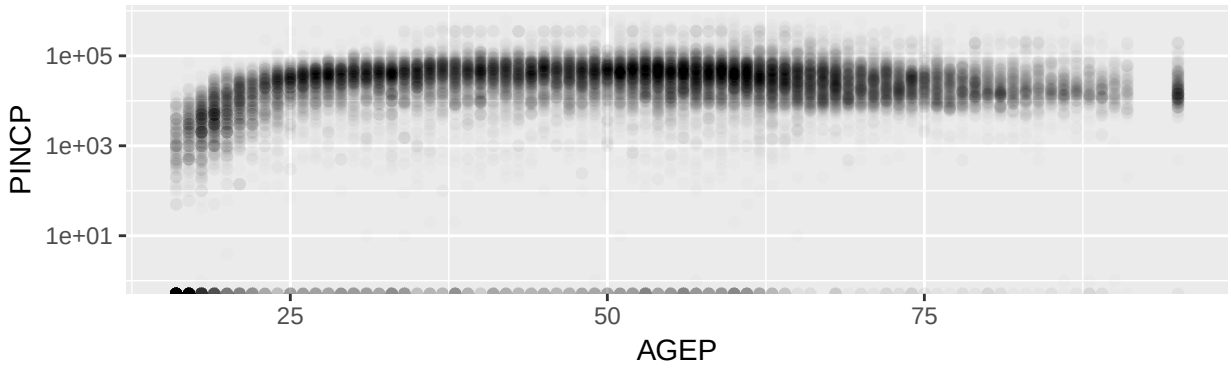
```
ggplot(df, aes(x = AGEP, y = PINCP)) + geom_point()
```



To move the mass of points above the x-axis, we can apply a log_{10} transformation to the `PINCP` variable. In addition, changing the `alpha` to a value less than 1.0 will allow us to take advantage of additive transparency – that is allowing the transparencies of multiple overlapping points to darken areas of creater density. We now

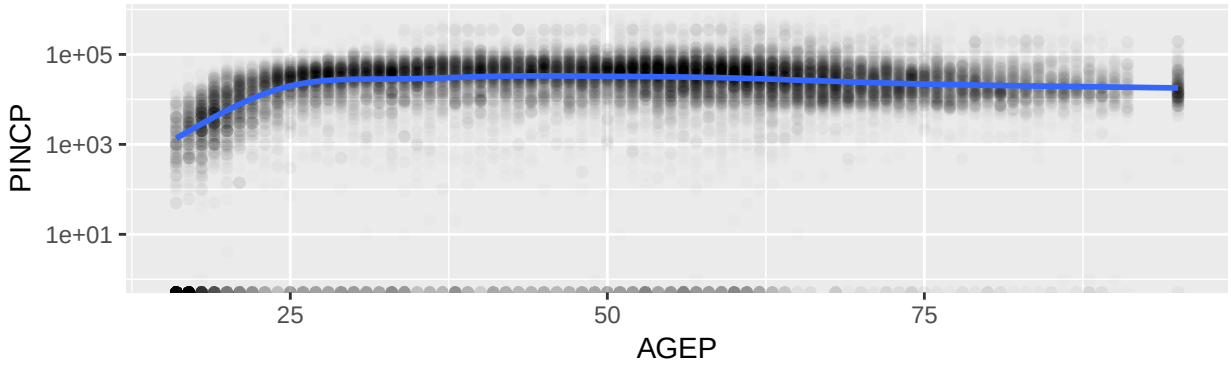
can see that income generally rises quickly before the age of 25, then plateaus.

```
ggplot(df, aes(x = AGEP, y = PINCP)) + geom_point(alpha = 0.01) + scale_y_log10()
```



In addition, we can apply a kernel density smoothed line to show the moving average trend of income as a function of age.

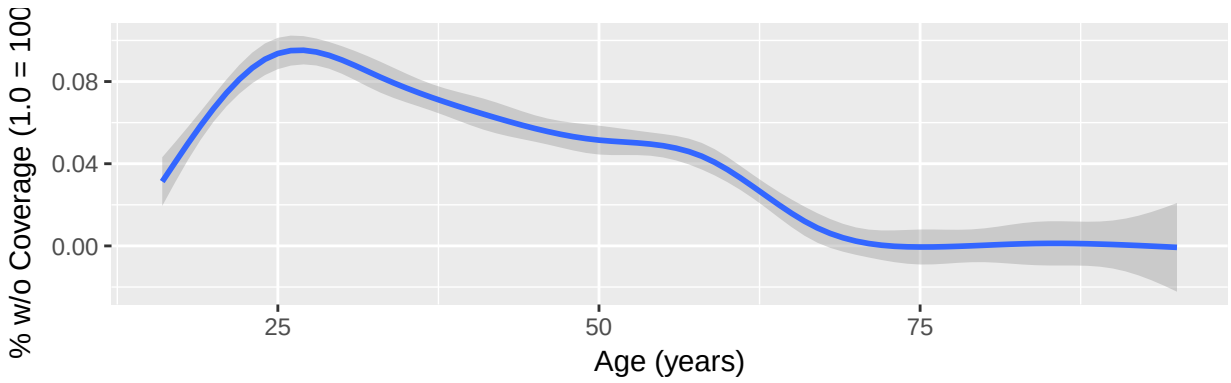
```
ggplot(df, aes(x = AGEP, y = PINCP)) + geom_point(alpha = 0.01) + scale_y_log10() +
  geom_smooth()
```



Using the same principles, we can find the bivariate relationship between age and health coverage. As health coverage (`coverage`) is coded as a binary, the smoothed estimate is equivalent to a locally estimated probability of no healthcare coverage.

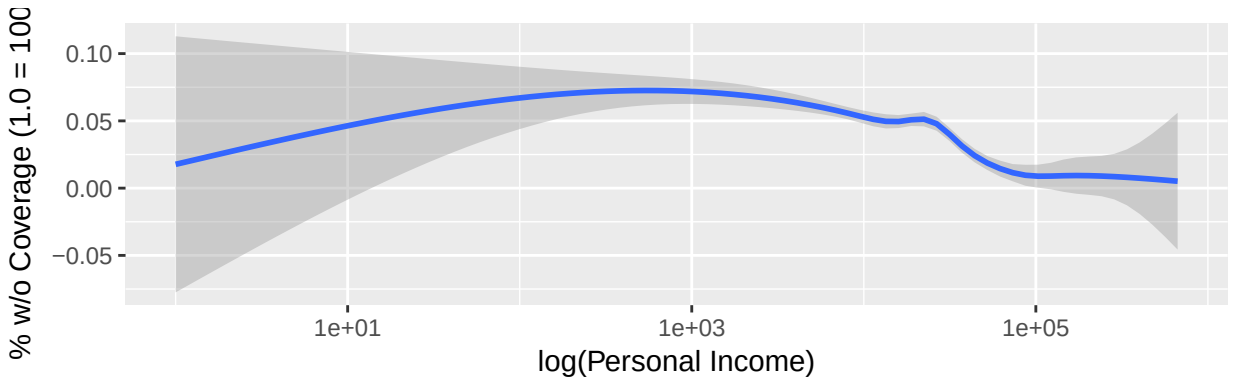
First up is coverage versus age. Lack of coverage spikes around `AGEP = 25` at a value of about 9%, then drops for older ages. Notice that the grey region around the smoothed mean estimate hugs the line fairly closely, indicating that the estimates are fairly certain with low variability.

```
ggplot(df, aes(x = AGEP, y = coverage))  + geom_smooth() +
   labs(x = "Age (years)", y = "% w/o Coverage (1.0 = 100%)")
```



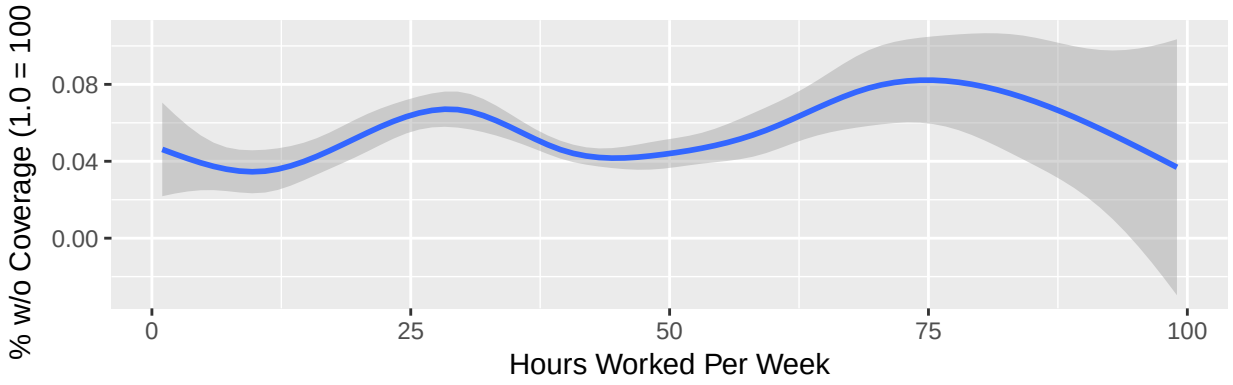
A comparison of coverage and income yields challenging results to interpret due to the notably large uncertainties at the very low and very high incomes, largely due to lower densities of records.

```
ggplot(df, aes(x = PINCP, y = coverage)) + geom_smooth() +
  labs(x = "log(Personal Income)", y = "% w/o Coverage (1.0 = 100%)") +
  scale_x_log10()
```



Hours worked per week sheds some interesting insights on coverage patterns, particularly around the 20 to 32 hours a week range as well as beyond 50 hours a week. This may be due to the nature of part-time work or multiple jobs.

```
ggplot(df, aes(x = WKHP, y = coverage)) + geom_smooth() +
  labs(x = "Hours Worked Per Week", y = "% w/o Coverage (1.0 = 100%)")
```



Lastly, there is less coverage among those who are closer to and below the coverage line.

```
ggplot(df, aes(x = POVPIP, y = coverage))+ geom_smooth() +
  labs(x = "Poverty Level (100 = at level)", y = "% w/o Coverage (1.0 = 100%)")
```