Statistical Learning

Lecture 07a - Missing Data

ANU - RSFAS

Last Updated: Mon Apr 18 18:52:29 2022

Missing Data

- Notes based on Linear Models with R by Julian Faraway - Chapter 13.
- For a fuller discussion see Statistical Analysis with Missing Data by Roderick Little and Donald Rubin.
- Another nice discussion: https://data.library.virginia.edu/getting-started-with-multiple-imputation-in-r/
- Types of missing data:
  - **Missing cases**: Sometimes we fail to observe a complete case (x_i, y_i). Indeed, when we draw a sample from a population, we do not observe the unsampled cases.
  - **Incomplete values**: Suppose we run an experiment to study the lifetimes of light bulbs. We might run out of time waiting for all the light-bulbs to die and decide to end the experiment. This leads to censored data.
  - **Missing values**: Sometimes we observe some components of a case but not others. We might observe the values of some predictors but not others. Perhaps the predictors are observed but not the response.

Why are the Data Missing?

- **Missing Completely at Random (MCAR)**: The probability that a value is missing is the same for all cases. If we simply delete all cases with missing values from the analysis, we will cause no bias, although we may lose some information.
- **Missing at Random (MAR)**: The probability of a value being missing depends on a known mechanism. For example, in social surveys, certain groups are less likely to provide information than others. As long as we know the group membership of the individual being sampled, then this is an example of MAR.
- **Missing not at Random (MNAR)**: The probability that a value is missing depends on some unobserved variable or, more seriously, on what value would have been observed.

Some Missing Data - Chicago Insurance Data

Data from a 1970's study on the relationship between insurance redlining in Chicago and racial composition, fire and theft rates, age of housing and income in 47 zip codes. Missing values have been randomly added.

- **race**: racial composition in percent minority
- **fire**: fires per 100 housing units
- **theft**: theft per 1000 population
- **age**: percent of housing units built before 1939
- **income**: median family income in thousands of dollars
- **involact**: new FAIR plan policies and renewals per 100 housing units

```
library(faraway)
data(chmiss)
head(chmiss)
```

```
##       race fire theft  age involact income
## 60626 10.0  6.2    29 60.4       NA 11.744
## 60640 22.2  9.5    44 76.5      0.1  9.323
## 60613 19.6 10.5    36   NA      1.2  9.948
## 60657 17.3  7.7    37   NA      0.5 10.656
## 60614 24.5  8.6    53 81.4      0.7  9.730
## 60610 54.0 34.1    68 52.6      0.3  8.231
```

```
summary(chmiss)
```

```
##       race            fire           theft
##  Min.   : 1.00   Min.   : 2.00   Min.   :  3.00
##  1st Qu.: 3.75   1st Qu.: 5.60   1st Qu.: 22.00
##  Median :24.50   Median : 9.50   Median : 29.00
##  Mean   :35.61   Mean   :11.42   Mean   : 32.65
##  3rd Qu.:57.65   3rd Qu.:15.10   3rd Qu.: 38.00
##  Max.   :99.70   Max.   :36.20   Max.   :147.00
##  NA's   :4       NA's   :2       NA's   :4
##       age            involact          income
##  Min.   : 2.00   Min.   :0.0000   Min.   : 5.583
##  1st Qu.:48.30   1st Qu.:0.0000   1st Qu.: 8.564
##  Median :64.40   Median :0.5000   Median :10.694
##  Mean   :59.97   Mean   :0.6477   Mean   :10.736
##  3rd Qu.:78.25   3rd Qu.:0.9250   3rd Qu.:12.102
##  Max.   :90.10   Max.   :2.2000   Max.   :21.480
##  NA's   :5       NA's   :3        NA's   :2
```

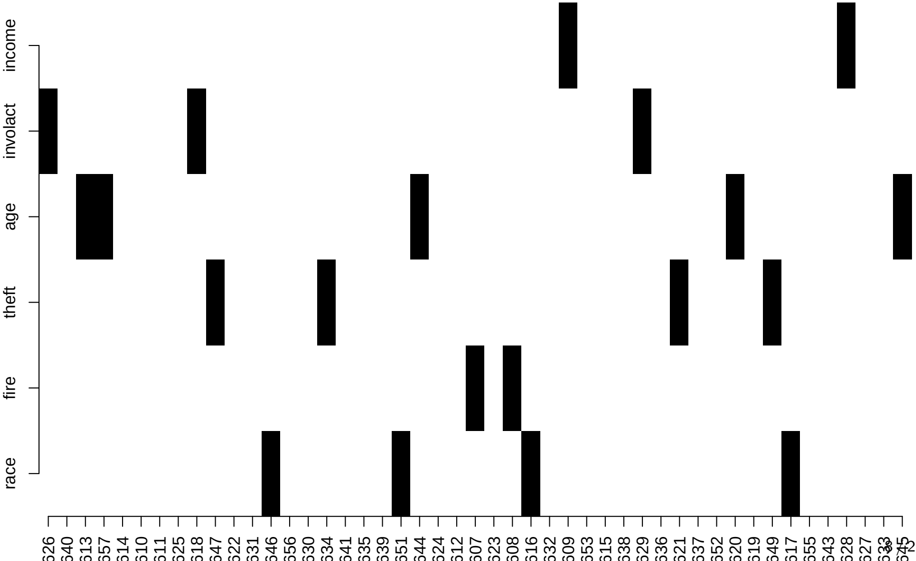
We can see missing values in all variables. It is also helpful to see how many missing values appear in each case.

```
rowSums(is.na(chmiss))
```

```
## 60626 60640 60613 60657 60614 60610 60611 60625 60618 60647
##     1     0     1     1     0     0     0     0     1     1
## 60622 60631 60646 60656 60630 60634 60641 60635 60639 60651
##     0     0     1     0     0     1     0     0     0     1
## 60644 60624 60612 60607 60623 60608 60616 60632 60609 60653
##     1     0     0     1     0     1     1     0     1     0
## 60615 60638 60629 60636 60621 60637 60652 60620 60619 60649
##     0     0     1     0     1     0     0     1     0     1
## 60617 60655 60643 60628 60627 60633 60645
##     1     0     0     1     0     0     1
```

Not good - there is a missing value for almost every row!

```r
image(is.na(chmiss), axes = FALSE, col = gray(1:0))
axis(2, at = 0:5/5, labels = colnames(chmiss))
axis(1, at = 0:46/46, labels = row.names(chmiss),
    las = 2)
```



Deletion Strategy

Using the full data which does not have the missing data, let's fit a regression model.

```
data(chredlin, package = "faraway")
modfull <- lm(involact ~ race + fire +
    theft + age + income, chredlin)
sumary(modfull)
```

```
##               Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -0.6089790  0.4952601 -1.2296 0.2258512
## race         0.0091325  0.0023158  3.9435 0.0003067
## fire         0.0388166  0.0084355  4.6015     4e-05
## theft       -0.0102976  0.0028529 -3.6096 0.0008269
## age          0.0082707  0.0027815  2.9734 0.0049143
## income       0.0245001  0.0316965  0.7730 0.4439816
##
## n = 47, p = 6, Residual SE = 0.33513, R-Squared = 0.75
```

Deletion Strategy

```
modmiss <- lm(involact ~ race + fire +
    theft + age + income, chmiss)
sumary(modmiss)
```

```
##                Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -1.1164827  0.6057615 -1.8431 0.0794750
## race         0.0104867  0.0031283  3.3522 0.0030180
## fire         0.0438757  0.0103190  4.2519 0.0003557
## theft       -0.0172198  0.0059005 -2.9184 0.0082154
## age          0.0093766  0.0034940  2.6837 0.0139041
## income       0.0687006  0.0421558  1.6297 0.1180775
##
## n = 27, p = 6, Residual SE = 0.33822, R-Squared = 0.79
```

- Note that in this case we still get the right sign, but the standard errors are inflated.

Single Imputation

A simple solution to the problem is to fill in or impute the missing values. For example, we can fill in the missing values by the variable means:

```r
cmeans <- colMeans(chmiss, na.rm = TRUE)
mchm <- chmiss
for (i in c(1:4, 6)) {
    mchm[is.na(chmiss[, i]), i] <- cmeans[i]
}
```

- Here we are only imputing the covariates, as we can naturally use our model:

$$y \sim x$$

to impute y.

- What if we have a categorical covariate?

Single Imputation

```r
imod <- lm(involact ~ race + fire +
    theft + age + income, mchm)
sumary(imod)
```

```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0708021  0.5094531  0.1390 0.890203
## race         0.0071173  0.0027057  2.6305 0.012245
## fire         0.0287418  0.0093855  3.0624 0.004021
## theft       -0.0030590  0.0027457 -1.1141 0.272242
## age          0.0060795  0.0032079  1.8952 0.065695
## income      -0.0270917  0.0316782 -0.8552 0.397791
##
## n = 44, p = 6, Residual SE = 0.38412, R-Squared = 0.68
```

Single Imputation

Compared to the complete data fit:

- *theft* and *age* are no longer significant.
- Estimated coefficients are biased toward zero. This is similar to the errors in variables situation.
- The bias introduced by the fill-in method can be substantial and may not be compensated by the attendant reduction in variance. For this reason, mean imputation is not recommended except where the fraction of filled values is small.

Single Imputation - Regression Methods

- A more sophisticated approach to imputation is through regression methods.

- Let's try to fill in the missing *race* values.

```
lmodr <- lm(race ~ fire + theft + age +
    income + involact, chmiss)
chmiss[is.na(chmiss$race), ]
predict(lmodr, chmiss[is.na(chmiss$race),
    ])
```

Single Imputation - Regression Methods

```
##        race fire theft  age involact income
## 60646    NA  5.7    11 27.9      0.0 16.250
## 60651    NA 15.1    30 89.8      0.8 10.510
## 60616    NA 12.2    46 48.0      0.6  8.212
## 60617    NA 10.8    34 58.0      0.9 11.156
```

```
##     60646     60651     60616     60617
## -21.75110  26.10017  78.52071  49.75061
```

- Notice that we get a negative value for the percentage.
- One trick that can be applied when the response is bounded between zero and one is the logit transformation:

$$y \to log(y/(1-y))$$

This transformation maps the interval to the whole real line.

Single Imputation - Regression Methods

```
lmodr <- lm(logit(race/100) ~ fire +
    theft + age + income + involact,
    chmiss)
ilogit(predict(lmodr, chmiss[is.na(chmiss$race),
    ])) * 100
```

```
##      60646      60651      60616      60617
##  0.3248019 14.2932964 90.0765080 43.8484035
```

- We can see how our predicted values compare to the actual values:

```
chredlin$race[is.na(chmiss$race)]
```

```
## [1]  1.0 13.4 62.3 36.4
```

- Like the mean fill-in method, the regression fill-in method will also introduce a bias toward zero in the coefficients while tending to reduce the variance.

Multiple Imputation

- The single imputation methods described above cause bias while deletion causes a loss of information from the data.
- Multiple imputation is a way to reduce the bias.
- **The problem with single imputation is that the imputed value, be it a mean or a regression-predicted value, tends to be less variable than the value we would have seen because the imputed value does not include the error variation that would normally be seen in observed data.**

Multiple Imputation

- Multiple imputation is implemented in the *Amelia* package of Honaker, King, and Blackwell (2011).

- A basic assumption is that the data is multivariate normal.

- The methodology is quite robust to this assumption but we need to make modifications in some cases. Heavily skewed variables are best log-transformed before imputation and categorical variables need to be declared for special treatment.

Multiple Imputation

```r
library(Amelia)
set.seed(123)
am.imp <- amelia(chmiss, m = 25, p2s = 0)
```

- We can now fit our linear model to **each** of the 25 data sets.

Multiple Imputation

```r
betas <- NULL
ses <- NULL
for (i in 1:am.imp$m) {
    lmod <- lm(involact ~ race + fire +
        theft + age + income, am.imp$imputations[[i]])
    betas <- rbind(betas, coef(lmod))
    ses <- rbind(ses, coef(summary(lmod))[,
        2])
}
```

Combining the Results

$$\hat{\beta}_j = \frac{1}{m} \sum_i \hat{\beta}_{ij}$$

$$\mathrm{se}_j^2 = \frac{1}{m} \sum_i \mathrm{se}_{ij}^2 + Var(\hat{\beta}_j)(1 + 1/m)$$

Where $Var(\hat{\beta}_j)$ is the sample variance over the imputed $\hat{\beta}_{ij}$s.

Combining the Results

```
output <- mi.meld(q = betas, se = ses)
output
```

```
## $q.mi
##      (Intercept)        race       fire        theft
## [1,]  -0.5303132 0.008587492 0.03748539 -0.009591052
##             age     income
## [1,] 0.00803049 0.02030737
##
## $se.mi
##      (Intercept)        race        fire       theft
## [1,]   0.6224611 0.002776556 0.009757678 0.005905341
##              age     income
## [1,] 0.003256562 0.04372946
```

Combining the Results

- Compute our observed *t*-statistics:

```
output$q.mi/output$se.mi
```

```
##      (Intercept)     race    fire     theft      age
## [1,]   -0.851962 3.092857 3.84163 -1.624132 2.465941
##         income
## [1,] 0.4643864
```