library(outbreaks)

library(tidyverse)

library(plyr)

library(mice)

library(caret)

library(purrr)

**The data**

The dataset contains case ID, date of onset, date of hospitalization, date of outcome, gender, age, province and of course outcome: Death or Recovery.

**Pre-processing**

**Change: variable names (i.e. column names) have been renamed, dots have been replaced with underscores, letters are all lower case now.**

**Change: I am using the tidyverse notation more consistently.**

First, I'm doing some preprocessing, including:

* renaming missing data as NA
* adding an ID column
* setting column types
* gathering date columns
* changing factor names of dates (to make them look nicer in plots) and of province (to combine provinces with few cases)

fluH7N9\_china\_2013$age[which(fluH7N9\_china\_2013$age == "?")] <- NA

fluH7N9\_china\_2013\_gather <- fluH7N9\_china\_2013 %>%

mutate(case\_id = paste("case", case\_id, sep = "\_"),

age = as.numeric(age)) %>%

gather(Group, Date, date\_of\_onset:date\_of\_outcome) %>%

mutate(Group = as.factor(mapvalues(Group, from = c("date\_of\_onset", "date\_of\_hospitalisation", "date\_of\_outcome"),

to = c("date of onset", "date of hospitalisation", "date of outcome"))),

province = mapvalues(province, from = c("Anhui", "Beijing", "Fujian", "Guangdong", "Hebei", "Henan", "Hunan", "Jiangxi", "Shandong", "Taiwan"), to = rep("Other", 10)))

I'm also

* adding a third gender level for unknown gender

levels(fluH7N9\_china\_2013\_gather$gender) <- c(levels(fluH7N9\_china\_2013\_gather$gender), "unknown")

fluH7N9\_china\_2013\_gather$gender[is.na(fluH7N9\_china\_2013\_gather$gender)] <- "unknown"

head(fluH7N9\_china\_2013\_gather)

*## case\_id outcome gender age province Group Date*

*## 1 case\_1 Death m 58 Shanghai date of onset 2013-02-19*

*## 2 case\_2 Death m 7 Shanghai date of onset 2013-02-27*

*## 3 case\_3 Death f 11 Other date of onset 2013-03-09*

*## 4 case\_4 <NA> f 18 Jiangsu date of onset 2013-03-19*

*## 5 case\_5 Recover f 20 Jiangsu date of onset 2013-03-19*

*## 6 case\_6 Death f 9 Jiangsu date of onset 2013-03-21*

For plotting, I am defining a custom ggplot2 theme:

my\_theme <- function(base\_size = 12, base\_family = "sans"){

theme\_minimal(base\_size = base\_size, base\_family = base\_family) +

theme(

axis.text = element\_text(size = 12),

axis.text.x = element\_text(angle = 45, vjust = 0.5, hjust = 0.5),

axis.title = element\_text(size = 14),

panel.grid.major = element\_line(color = "grey"),

panel.grid.minor = element\_blank(),

panel.background = element\_rect(fill = "aliceblue"),

strip.background = element\_rect(fill = "lightgrey", color = "grey", size = 1),

strip.text = element\_text(face = "bold", size = 12, color = "black"),

legend.position = "bottom",

legend.justification = "top",

legend.box = "horizontal",

legend.box.background = element\_rect(colour = "grey50"),

legend.background = element\_blank(),

panel.border = element\_rect(color = "grey", fill = NA, size = 0.5)

)

}

And use that theme to visualize the data:

ggplot(data = fluH7N9\_china\_2013\_gather, aes(x = Date, y = age, fill = outcome)) +

stat\_density2d(aes(alpha = ..level..), geom = "polygon") +

geom\_jitter(aes(color = outcome, shape = gender), size = 1.5) +

geom\_rug(aes(color = outcome)) +

scale\_y\_continuous(limits = c(0, 90)) +

labs(

fill = "Outcome",

color = "Outcome",

alpha = "Level",

shape = "Gender",

x = "Date in 2013",

y = "Age",

title = "2013 Influenza A H7N9 cases in China",

subtitle = "Dataset from 'outbreaks' package (Kucharski et al. 2014)",

caption = ""

) +

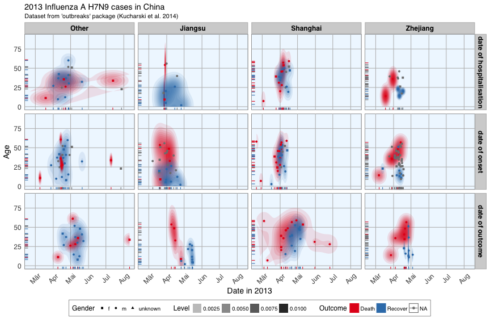
facet\_grid(Group ~ province) +

my\_theme() +

scale\_shape\_manual(values = c(15, 16, 17)) +

scale\_color\_brewer(palette="Set1", na.value = "grey50") +

scale\_fill\_brewer(palette="Set1")

Gives this plot:  
[](https://datascienceplus.com/wp-content/uploads/2018/05/main-1.png)

ggplot(data = fluH7N9\_china\_2013\_gather, aes(x = Date, y = age, color = outcome)) +

geom\_point(aes(color = outcome, shape = gender), size = 1.5, alpha = 0.6) +

geom\_path(aes(group = case\_id)) +

facet\_wrap( ~ province, ncol = 2) +

my\_theme() +

scale\_shape\_manual(values = c(15, 16, 17)) +

scale\_color\_brewer(palette="Set1", na.value = "grey50") +

scale\_fill\_brewer(palette="Set1") +

labs(

color = "Outcome",

shape = "Gender",

x = "Date in 2013",

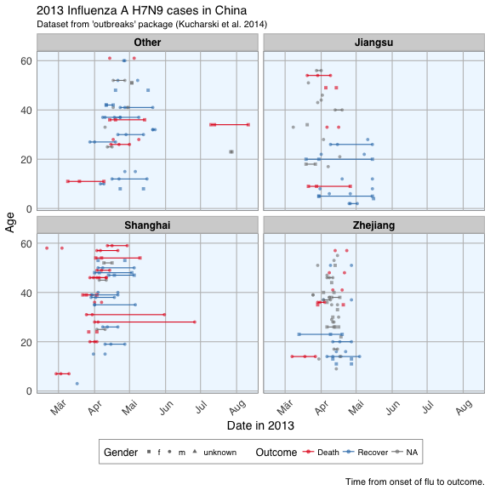
y = "Age",

title = "2013 Influenza A H7N9 cases in China",

subtitle = "Dataset from 'outbreaks' package (Kucharski et al. 2014)",

caption = "\nTime from onset of flu to outcome."

)

Gives this plot:  
[](https://datascienceplus.com/wp-content/uploads/2018/05/unnamed-chunk-6-1.png)

**Features**

In machine learning-speak features are what we call the variables used for model training. Using the right features dramatically influences the accuracy and success of your model. For this example, I am keeping age, but I am also generating new features from the date information and converting gender and province into numerical values.

dataset <- fluH7N9\_china\_2013 %>%

mutate(hospital = as.factor(ifelse(is.na(date\_of\_hospitalisation), 0, 1)),

gender\_f = as.factor(ifelse(gender == "f", 1, 0)),

province\_Jiangsu = as.factor(ifelse(province == "Jiangsu", 1, 0)),

province\_Shanghai = as.factor(ifelse(province == "Shanghai", 1, 0)),

province\_Zhejiang = as.factor(ifelse(province == "Zhejiang", 1, 0)),

province\_other = as.factor(ifelse(province == "Zhejiang" | province == "Jiangsu" | province == "Shanghai", 0, 1)),

days\_onset\_to\_outcome = as.numeric(as.character(gsub(" days", "",

as.Date(as.character(date\_of\_outcome), format = "%Y-%m-%d") -

as.Date(as.character(date\_of\_onset), format = "%Y-%m-%d")))),

days\_onset\_to\_hospital = as.numeric(as.character(gsub(" days", "",

as.Date(as.character(date\_of\_hospitalisation), format = "%Y-%m-%d") -

as.Date(as.character(date\_of\_onset), format = "%Y-%m-%d")))),

age = age,

early\_onset = as.factor(ifelse(date\_of\_onset < summary(fluH7N9\_china\_2013$date\_of\_onset)[[3]], 1, 0)),

early\_outcome = as.factor(ifelse(date\_of\_outcome < summary(fluH7N9\_china\_2013$date\_of\_outcome)[[3]], 1, 0))) %>%

subset(select = -c(2:4, 6, 8))

rownames(dataset) <- dataset$case\_id

dataset[, -2] <- as.numeric(as.matrix(dataset[, -2]))

head(dataset)

*## case\_id outcome age hospital gender\_f province\_Jiangsu province\_Shanghai*

*## 1 1 Death 87 0 0 0 1*

*## 2 2 Death 27 1 0 0 1*

*## 3 3 Death 35 1 1 0 0*

*## 4 4 <NA> 45 1 1 1 0*

*## 5 5 Recover 48 1 1 1 0*

*## 6 6 Death 32 1 1 1 0*

*## province\_Zhejiang province\_other days\_onset\_to\_outcome*

*## 1 0 0 13*

*## 2 0 0 11*

*## 3 0 1 31*

*## 4 0 0 NA*

*## 5 0 0 57*

*## 6 0 0 36*

*## days\_onset\_to\_hospital early\_onset early\_outcome*

*## 1 NA 1 1*

*## 2 4 1 1*

*## 3 10 1 1*

*## 4 8 1 NA*

*## 5 11 1 0*

*## 6 7 1 1*

summary(dataset$outcome)

*## Death Recover NA's*

*## 32 47 57*

**Imputing missing values**

I am using the [mice package for imputing missing values](https://gerkovink.github.io/miceVignettes/Ad_hoc_and_mice/Ad_hoc_methods.html).

Working with mice Package

##### Working with mice

**1. Open R and load the packages mice and lattice**

**require**(mice)

**require**(lattice)

set.seed(123)

If mice is not yet installed, run:

install.packages("mice")

**2. Inspect the incomplete data**

The mice package contains several datasets. Once the package is loaded, these datasets can be used. Have a look at the nhanes dataset (Schafer, 1997, Table 6.14) by typing

nhanes

## age bmi hyp chl

## 1 1 NA NA NA

## 2 2 22.7 1 187

## 3 1 NA 1 187

## 4 3 NA NA NA

## 5 1 20.4 1 113

## 6 3 NA NA 184

## 7 1 22.5 1 118

## 8 1 30.1 1 187

## 9 2 22.0 1 238

## 10 2 NA NA NA

## 11 1 NA NA NA

## 12 2 NA NA NA

## 13 3 21.7 1 206

## 14 2 28.7 2 204

## 15 1 29.6 1 NA

## 16 1 NA NA NA

## 17 3 27.2 2 284

## 18 2 26.3 2 199

## 19 1 35.3 1 218

## 20 3 25.5 2 NA

## 21 1 NA NA NA

## 22 1 33.2 1 229

## 23 1 27.5 1 131

## 24 3 24.9 1 NA

## 25 2 27.4 1 186

The nhanes dataset is a small data set with non-monotone missing values. It contains 25 observations on four variables: age group, body mass index, hypertension and cholesterol (mg/dL).

To learn more about the data, use one of the two following help commands:

help(nhanes)

?nhanes

**3. Get an overview of the data by the summary() command:**

summary(nhanes)

## age bmi hyp chl

## Min. :1.00 Min. :20.40 Min. :1.000 Min. :113.0

## 1st Qu.:1.00 1st Qu.:22.65 1st Qu.:1.000 1st Qu.:185.0

## Median :2.00 Median :26.75 Median :1.000 Median :187.0

## Mean :1.76 Mean :26.56 Mean :1.235 Mean :191.4

## 3rd Qu.:2.00 3rd Qu.:28.93 3rd Qu.:1.000 3rd Qu.:212.0

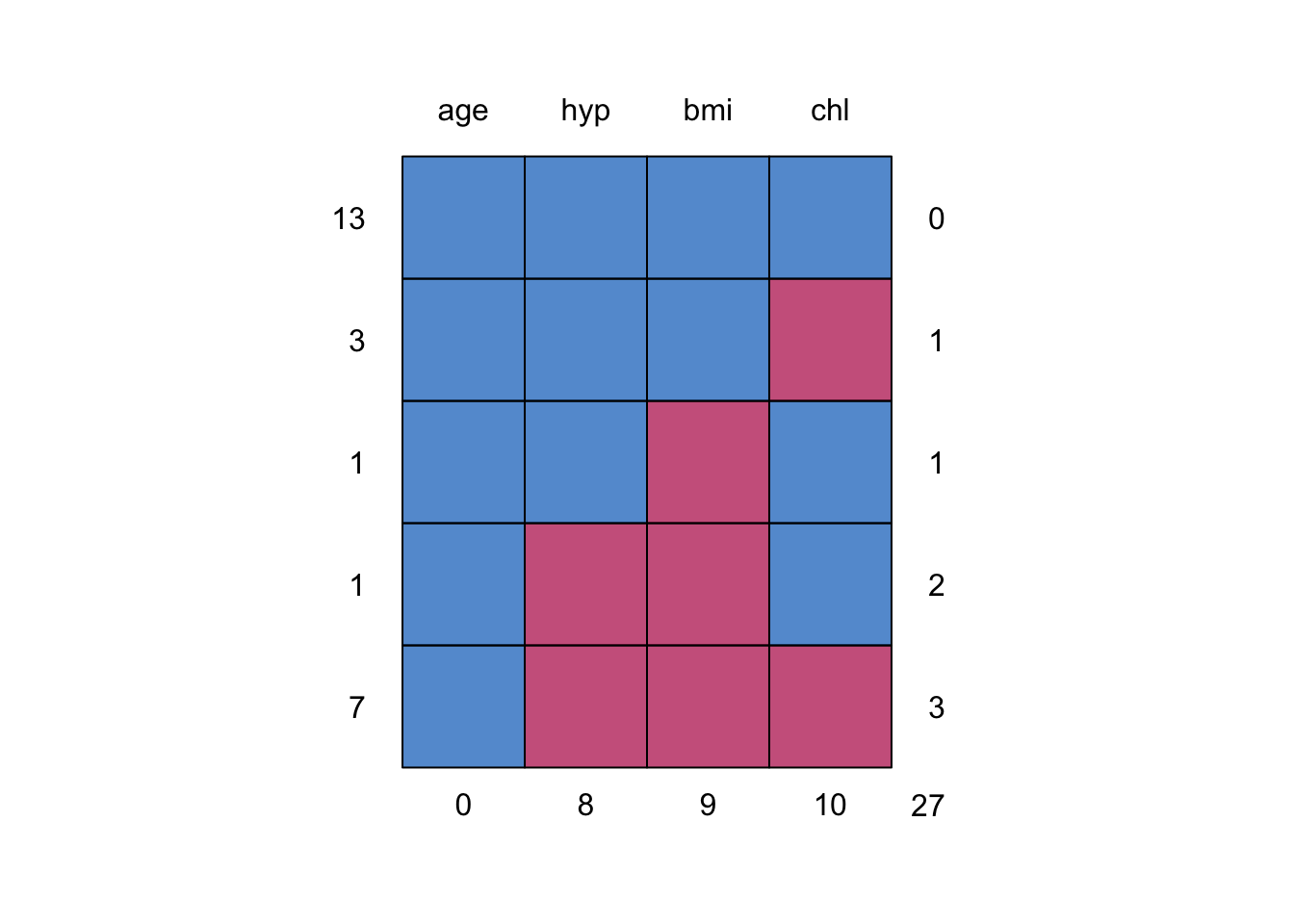
## Max. :3.00 Max. :35.30 Max. :2.000 Max. :284.0

## NA's :9 NA's :8 NA's :10

**4. Inspect the missing data pattern**

Check the missingness pattern for the nhanes dataset

md.pattern(nhanes)



## age hyp bmi chl

## 13 1 1 1 1 0

## 3 1 1 1 0 1

## 1 1 1 0 1 1

## 1 1 0 0 1 2

## 7 1 0 0 0 3

## 0 8 9 10 27

The missingness pattern shows that there are 27 missing values in total: 10 for chl , 9 for bmi and 8 for hyp. Moreover, there are thirteen completely observed rows, four rows with 1 missing, one row with 2 missings and seven rows with 3 missings. Looking at the missing data pattern is always useful (but may be difficult for datasets with many variables). It can give you an indication on how much information is missing and how the missingness is distributed.

##### Ad Hoc imputation methods

**5. Form a regression model where age is predicted from bmi.**

fit <- with(nhanes, lm(age ~ bmi))

summary(fit)

##

## Call:

## lm(formula = age ~ bmi)

##

## Residuals:

## Min 1Q Median 3Q Max

## -1.2660 -0.5614 -0.1225 0.4660 1.2344

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) 3.76718 1.31945 2.855 0.0127 \*

## bmi -0.07359 0.04910 -1.499 0.1561

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.8015 on 14 degrees of freedom

## (9 observations deleted due to missingness)

## Multiple R-squared: 0.1383, Adjusted R-squared: 0.07672

## F-statistic: 2.246 on 1 and 14 DF, p-value: 0.1561

**6. Impute the missing data in the nhanes dataset with mean imputation.**

imp <- mice(nhanes, method = "mean", m = 1, maxit = 1)

##

## iter imp variable

## 1 1 bmi hyp chl

The imputations are now done. As you can see, the algorithm ran for 1 iteration (maxit = 1) and presented us with only 1 imputation (m = 1) for each missing datum. This is correct, as substituting each missing data multiple times with the observed data mean would not make any sense (the inference would be equal, no matter which imputed dataset we would analyze). Likewise, more iterations would be computationally inefficient as the observed data mean does not change based on our imputations. We named the imputed object imp following the convention used in mice, but if you wish you can name it anything you’d like.

**7. Explore the imputed data with the complete() function. What do you think the variable means are? What happened to the regression equation after imputation?**

complete(imp)

## age bmi hyp chl

## 1 1 26.5625 1.235294 191.4

## 2 2 22.7000 1.000000 187.0

## 3 1 26.5625 1.000000 187.0

## 4 3 26.5625 1.235294 191.4

## 5 1 20.4000 1.000000 113.0

## 6 3 26.5625 1.235294 184.0

## 7 1 22.5000 1.000000 118.0

## 8 1 30.1000 1.000000 187.0

## 9 2 22.0000 1.000000 238.0

## 10 2 26.5625 1.235294 191.4

## 11 1 26.5625 1.235294 191.4

## 12 2 26.5625 1.235294 191.4

## 13 3 21.7000 1.000000 206.0

## 14 2 28.7000 2.000000 204.0

## 15 1 29.6000 1.000000 191.4

## 16 1 26.5625 1.235294 191.4

## 17 3 27.2000 2.000000 284.0

## 18 2 26.3000 2.000000 199.0

## 19 1 35.3000 1.000000 218.0

## 20 3 25.5000 2.000000 191.4

## 21 1 26.5625 1.235294 191.4

## 22 1 33.2000 1.000000 229.0

## 23 1 27.5000 1.000000 131.0

## 24 3 24.9000 1.000000 191.4

## 25 2 27.4000 1.000000 186.0

We see the repetitive numbers 26.5625 for bmi, 1.2352594 for hyp, and 191.4 for chl. These can be confirmed as the means of the respective variables (columns):

colMeans(nhanes, na.rm = TRUE)

## age bmi hyp chl

## 1.760000 26.562500 1.235294 191.400000

We saw during the inspection of the missing data pattern that variable age has no missings. Therefore nothing is imputed for age because we would not want to alter the observed (and bonafide) values.

To inspect the regression model with the imputed data, run:

fit <- with(imp, lm(age ~ bmi))

summary(fit)

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

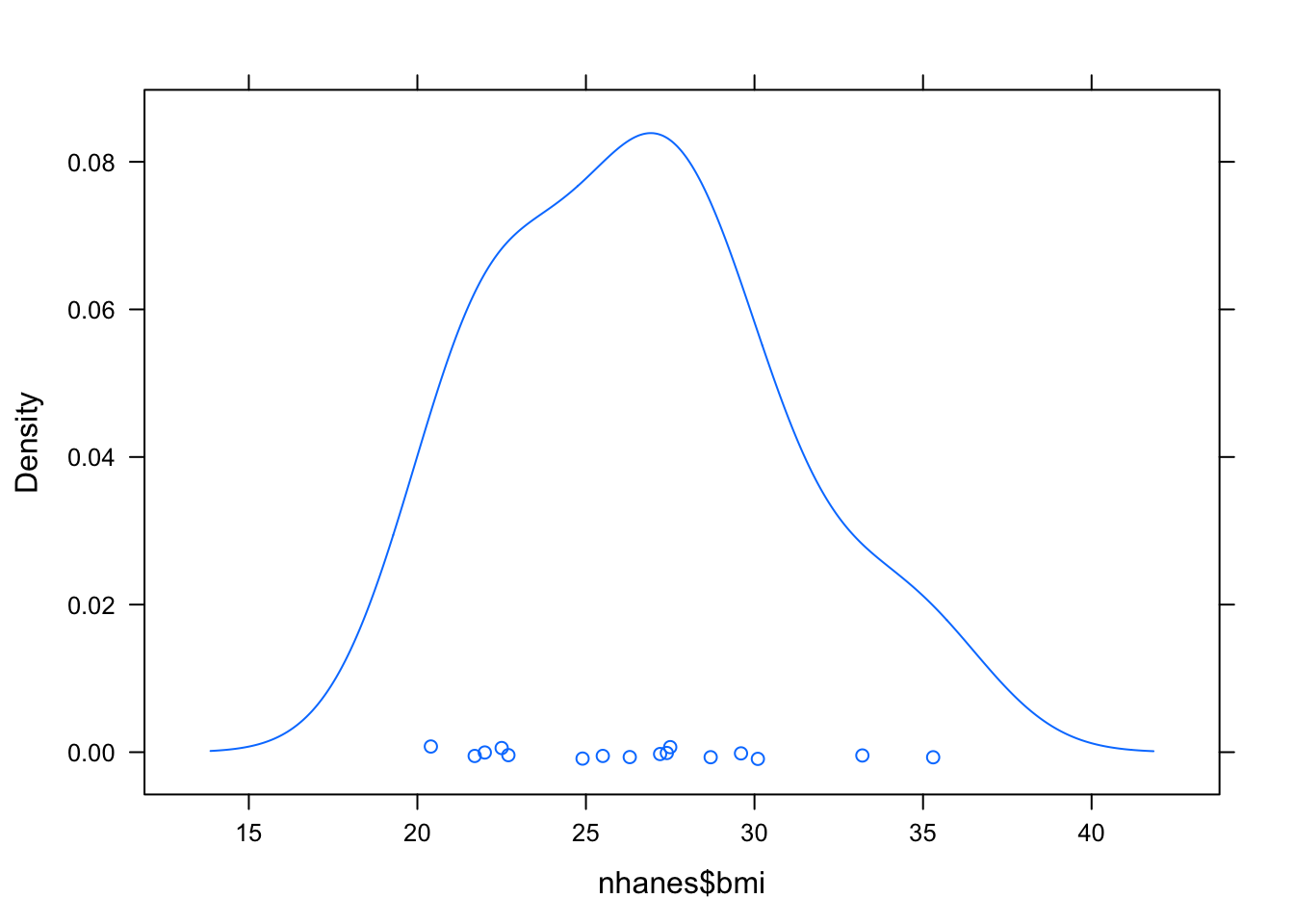
## <chr> <dbl> <dbl> <dbl> <dbl>

## 1 (Intercept) 3.71 1.33 2.80 0.0103

## 2 bmi -0.0736 0.0497 -1.48 0.152

It is clear that nothing changed, but then again this is not surprising as variable bmi is somewhat normally distributed and we are just adding weight to the mean.

densityplot(nhanes$bmi)



**8. Impute the missing data in the nhanes dataset with regression imputation.**

imp <- mice(nhanes, method = "norm.predict", m = 1, maxit = 1)

##

## iter imp variable

## 1 1 bmi hyp chl

The imputations are now done. This code imputes the missing values in the data set by the regression imputation method. The argument method = "norm.predict" first fits a regression model for each observed value, based on the corresponding values in other variables and then imputes the missing values with the predicted values.

**9. Again, inspect the completed data and investigate the imputed data regression model.**

complete(imp)

## age bmi hyp chl

## 1 1 31.98171 1.132574 198.1082

## 2 2 22.70000 1.000000 187.0000

## 3 1 28.83478 1.000000 187.0000

## 4 3 23.21098 1.530991 228.5499

## 5 1 20.40000 1.000000 113.0000

## 6 3 21.11303 1.475446 184.0000

## 7 1 22.50000 1.000000 118.0000

## 8 1 30.10000 1.000000 187.0000

## 9 2 22.00000 1.000000 238.0000

## 10 2 31.05181 1.423268 238.5342

## 11 1 31.37488 1.123040 193.6441

## 12 2 25.06646 1.264801 194.8752

## 13 3 21.70000 1.000000 206.0000

## 14 2 28.70000 2.000000 204.0000

## 15 1 29.60000 1.000000 181.1354

## 16 1 28.64966 1.044355 173.8032

## 17 3 27.20000 2.000000 284.0000

## 18 2 26.30000 2.000000 199.0000

## 19 1 35.30000 1.000000 218.0000

## 20 3 25.50000 2.000000 242.8954

## 21 1 34.82013 1.207723 218.8124

## 22 1 33.20000 1.000000 229.0000

## 23 1 27.50000 1.000000 131.0000

## 24 3 24.90000 1.000000 244.1845

## 25 2 27.40000 1.000000 186.0000

The repetitive numbering is gone. We have now obtained a more natural looking set of imputations: instead of filling in the same bmi for all ages, we now take age (as well as hyp and chl) into account when imputing bmi.

To inspect the regression model with the imputed data, run:

fit <- with(imp, lm(age ~ bmi))

summary(fit)

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

## <chr> <dbl> <dbl> <dbl> <dbl>

## 1 (Intercept) 4.63 0.925 5.00 0.0000463

## 2 bmi -0.105 0.0335 -3.14 0.00462

It is clear that something has changed. In fact, we extrapolated (part of) the regression model for the observed data to missing data in bmi. In other words; the relation (read: information) gets stronger and we’ve obtained more observations.

**10. Impute the missing data in the nhanes dataset with stochastic regression imputation.**

imp <- mice(nhanes, method = "norm.nob", m = 1, maxit = 1)

##

## iter imp variable

## 1 1 bmi hyp chl

The imputations are now done. This code imputes the missing values in the data set by the stochastic regression imputation method. The function does not incorporate the variability of the regression weights, so it is not ‘proper’ in the sense of Rubin (1987). For small samples, the variability of the imputed data will be underestimated.

**11. Again, inspect the completed data and investigate the imputed data regression model.**

complete(imp)

## age bmi hyp chl

## 1 1 35.46313 1.1192657 208.1269

## 2 2 22.70000 1.0000000 187.0000

## 3 1 28.04489 1.0000000 187.0000

## 4 3 28.36409 1.1892939 209.5659

## 5 1 20.40000 1.0000000 113.0000

## 6 3 16.34286 0.9233658 184.0000

## 7 1 22.50000 1.0000000 118.0000

## 8 1 30.10000 1.0000000 187.0000

## 9 2 22.00000 1.0000000 238.0000

## 10 2 28.72079 1.4989875 252.2370

## 11 1 30.15465 1.3168140 162.6556

## 12 2 27.21004 1.3627723 194.0366

## 13 3 21.70000 1.0000000 206.0000

## 14 2 28.70000 2.0000000 204.0000

## 15 1 29.60000 1.0000000 209.3729

## 16 1 31.14788 1.5316578 180.7741

## 17 3 27.20000 2.0000000 284.0000

## 18 2 26.30000 2.0000000 199.0000

## 19 1 35.30000 1.0000000 218.0000

## 20 3 25.50000 2.0000000 215.6881

## 21 1 27.72316 1.9274899 167.2897

## 22 1 33.20000 1.0000000 229.0000

## 23 1 27.50000 1.0000000 131.0000

## 24 3 24.90000 1.0000000 246.9890

## 25 2 27.40000 1.0000000 186.0000

We have once more obtained a more natural looking set of imputations, where instead of filling in the same bmi for all ages, we now take age (as well as hyp and chl) into account when imputing bmi. We also add a random error to allow for our imputations to be off the regression line.

To inspect the regression model with the imputed data, run:

fit <- with(imp, lm(age ~ bmi))

summary(fit)

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

## <chr> <dbl> <dbl> <dbl> <dbl>

## 1 (Intercept) 4.23 0.918 4.60 0.000125

## 2 bmi -0.0909 0.0334 -2.72 0.0122

**12. Re-run the stochastic imputation model with seed 123 and verify if your results are the same as the ones below**

## # A tibble: 2 x 5

## term estimate std.error statistic p.value

## <chr> <dbl> <dbl> <dbl> <dbl>

## 1 (Intercept) 4.13 1.13 3.66 0.00129

## 2 bmi -0.0904 0.0426 -2.12 0.0449

The imputation procedure uses random sampling, and therefore, the results will be (perhaps slightly) different if we repeat the imputations. In order to get exactly the same result, you can use the seed argument

imp <- mice(nhanes, method = "norm.nob", m = 1, maxit = 1, seed = 123)

fit <- with(imp, lm(age ~ bmi))

summary(fit)

where 123 is some arbitrary number that you can choose yourself. Re-running this command will always yields the same imputed values. The ability to replicate one’s findings exactly is considered essential in today’s reproducible science.

##### Multiple imputation

**13. Let us impute the missing data in the nhanes dataset**

imp <- mice(nhanes)

##

## iter imp variable

## 1 1 bmi hyp chl

## 1 2 bmi hyp chl

## 1 3 bmi hyp chl

## 1 4 bmi hyp chl

## 1 5 bmi hyp chl

## 2 1 bmi hyp chl

## 2 2 bmi hyp chl

## 2 3 bmi hyp chl

## 2 4 bmi hyp chl

## 2 5 bmi hyp chl

## 3 1 bmi hyp chl

## 3 2 bmi hyp chl

## 3 3 bmi hyp chl

## 3 4 bmi hyp chl

## 3 5 bmi hyp chl

## 4 1 bmi hyp chl

## 4 2 bmi hyp chl

## 4 3 bmi hyp chl

## 4 4 bmi hyp chl

## 4 5 bmi hyp chl

## 5 1 bmi hyp chl

## 5 2 bmi hyp chl

## 5 3 bmi hyp chl

## 5 4 bmi hyp chl

## 5 5 bmi hyp chl

imp

## Multiply imputed data set

## Call:

## mice(data = nhanes)

## Number of multiple imputations: 5

## Missing cells per column:

## age bmi hyp chl

## 0 9 8 10

## Imputation methods:

## age bmi hyp chl

## "" "pmm" "pmm" "pmm"

## VisitSequence:

## bmi hyp chl

## 2 3 4

## PredictorMatrix:

## age bmi hyp chl

## age 0 0 0 0

## bmi 1 0 1 1

## hyp 1 1 0 1

## chl 1 1 1 0

## Random generator seed value: NA

The imputations are now done. As you can see, the algorithm ran for 5 iterations (the default) and presented us with 5 imputations for each missing datum. For the rest of this document we will omit printing of the iteration cycle when we run mice. We do so by adding print=F to the mice call.

The object imp contains a multiply imputed data set (of class mids). It encapsulates all information from imputing the nhanes dataset, such as the original data, the imputed values, the number of missing values, number of iterations, and so on.

To obtain an overview of the information stored in the object imp, use the attributes() function:

attributes(imp)

## $names

## [1] "call" "data" "m"

## [4] "nmis" "imp" "method"

## [7] "predictorMatrix" "visitSequence" "form"

## [10] "post" "seed" "iteration"

## [13] "lastSeedValue" "chainMean" "chainVar"

## [16] "loggedEvents" "pad"

##

## $class

## [1] "mids"

For example, the original data are stored as

imp$data

## age bmi hyp chl

## 1 1 NA NA NA

## 2 2 22.7 1 187

## 3 1 NA 1 187

## 4 3 NA NA NA

## 5 1 20.4 1 113

## 6 3 NA NA 184

## 7 1 22.5 1 118

## 8 1 30.1 1 187

## 9 2 22.0 1 238

## 10 2 NA NA NA

## 11 1 NA NA NA

## 12 2 NA NA NA

## 13 3 21.7 1 206

## 14 2 28.7 2 204

## 15 1 29.6 1 NA

## 16 1 NA NA NA

## 17 3 27.2 2 284

## 18 2 26.3 2 199

## 19 1 35.3 1 218

## 20 3 25.5 2 NA

## 21 1 NA NA NA

## 22 1 33.2 1 229

## 23 1 27.5 1 131

## 24 3 24.9 1 NA

## 25 2 27.4 1 186

and the imputations are stored as

imp$imp

## $age

## NULL

##

## $bmi

## 1 2 3 4 5

## 1 30.1 27.2 29.6 35.3 29.6

## 3 29.6 29.6 29.6 26.3 30.1

## 4 27.4 20.4 21.7 27.4 25.5

## 6 24.9 24.9 20.4 21.7 20.4

## 10 27.5 27.5 27.4 24.9 22.0

## 11 30.1 28.7 29.6 22.0 33.2

## 12 27.5 29.6 29.6 27.5 28.7

## 16 26.3 30.1 29.6 28.7 27.2

## 21 26.3 22.0 27.2 35.3 24.9

##

## $hyp

## 1 2 3 4 5

## 1 1 1 1 1 1

## 4 2 1 1 2 2

## 6 2 1 2 2 1

## 10 2 1 1 2 1

## 11 1 1 1 1 1

## 12 2 1 2 1 1

## 16 1 1 1 1 1

## 21 1 1 1 1 1

##

## $chl

## 1 2 3 4 5

## 1 187 131 187 206 199

## 4 184 187 186 204 186

## 10 218 187 186 131 187

## 11 199 187 238 131 204

## 12 186 187 218 204 218

## 15 199 187 238 229 199

## 16 187 238 131 187 187

## 20 184 218 218 186 206

## 21 187 131 187 204 187

## 24 186 187 206 218 218

**14. Extract the completed data**

By default, mice() calculates five (m = 5) imputed data sets. In order to get the third imputed data set, use the complete() function

c3 <- complete(imp, 3)

md.pattern(c3)

## age bmi hyp chl

## [1,] 1 1 1 1 0

## [2,] 0 0 0 0 0

The collection of the mm imputed data sets can be exported by function complete() in long, broad and repeated formats. For example,

c.long <- complete(imp, "long")

c.long

## .imp .id age bmi hyp chl

## 1 1 1 1 30.1 1 187

## 2 1 2 2 22.7 1 187

## 3 1 3 1 29.6 1 187

## 4 1 4 3 27.4 2 184

## 5 1 5 1 20.4 1 113

## 6 1 6 3 24.9 2 184

## 7 1 7 1 22.5 1 118

## 8 1 8 1 30.1 1 187

## 9 1 9 2 22.0 1 238

## 10 1 10 2 27.5 2 218

## 11 1 11 1 30.1 1 199

## 12 1 12 2 27.5 2 186

## 13 1 13 3 21.7 1 206

## 14 1 14 2 28.7 2 204

## 15 1 15 1 29.6 1 199

## 16 1 16 1 26.3 1 187

## 17 1 17 3 27.2 2 284

## 18 1 18 2 26.3 2 199

## 19 1 19 1 35.3 1 218

## 20 1 20 3 25.5 2 184

## 21 1 21 1 26.3 1 187

## 22 1 22 1 33.2 1 229

## 23 1 23 1 27.5 1 131

## 24 1 24 3 24.9 1 186

## 25 1 25 2 27.4 1 186

## 26 2 1 1 27.2 1 131

## 27 2 2 2 22.7 1 187

## 28 2 3 1 29.6 1 187

## 29 2 4 3 20.4 1 187

## 30 2 5 1 20.4 1 113

## 31 2 6 3 24.9 1 184

## 32 2 7 1 22.5 1 118

## 33 2 8 1 30.1 1 187

## 34 2 9 2 22.0 1 238

## 35 2 10 2 27.5 1 187

## 36 2 11 1 28.7 1 187

## 37 2 12 2 29.6 1 187

## 38 2 13 3 21.7 1 206

## 39 2 14 2 28.7 2 204

## 40 2 15 1 29.6 1 187

## 41 2 16 1 30.1 1 238

## 42 2 17 3 27.2 2 284

## 43 2 18 2 26.3 2 199

## 44 2 19 1 35.3 1 218

## 45 2 20 3 25.5 2 218

## 46 2 21 1 22.0 1 131

## 47 2 22 1 33.2 1 229

## 48 2 23 1 27.5 1 131

## 49 2 24 3 24.9 1 187

## 50 2 25 2 27.4 1 186

## 51 3 1 1 29.6 1 187

## 52 3 2 2 22.7 1 187

## 53 3 3 1 29.6 1 187

## 54 3 4 3 21.7 1 186

## 55 3 5 1 20.4 1 113

## 56 3 6 3 20.4 2 184

## 57 3 7 1 22.5 1 118

## 58 3 8 1 30.1 1 187

## 59 3 9 2 22.0 1 238

## 60 3 10 2 27.4 1 186

## 61 3 11 1 29.6 1 238

## 62 3 12 2 29.6 2 218

## 63 3 13 3 21.7 1 206

## 64 3 14 2 28.7 2 204

## 65 3 15 1 29.6 1 238

## 66 3 16 1 29.6 1 131

## 67 3 17 3 27.2 2 284

## 68 3 18 2 26.3 2 199

## 69 3 19 1 35.3 1 218

## 70 3 20 3 25.5 2 218

## 71 3 21 1 27.2 1 187

## 72 3 22 1 33.2 1 229

## 73 3 23 1 27.5 1 131

## 74 3 24 3 24.9 1 206

## 75 3 25 2 27.4 1 186

## 76 4 1 1 35.3 1 206

## 77 4 2 2 22.7 1 187

## 78 4 3 1 26.3 1 187

## 79 4 4 3 27.4 2 204

## 80 4 5 1 20.4 1 113

## 81 4 6 3 21.7 2 184

## 82 4 7 1 22.5 1 118

## 83 4 8 1 30.1 1 187

## 84 4 9 2 22.0 1 238

## 85 4 10 2 24.9 2 131

## 86 4 11 1 22.0 1 131

## 87 4 12 2 27.5 1 204

## 88 4 13 3 21.7 1 206

## 89 4 14 2 28.7 2 204

## 90 4 15 1 29.6 1 229

## 91 4 16 1 28.7 1 187

## 92 4 17 3 27.2 2 284

## 93 4 18 2 26.3 2 199

## 94 4 19 1 35.3 1 218

## 95 4 20 3 25.5 2 186

## 96 4 21 1 35.3 1 204

## 97 4 22 1 33.2 1 229

## 98 4 23 1 27.5 1 131

## 99 4 24 3 24.9 1 218

## 100 4 25 2 27.4 1 186

## 101 5 1 1 29.6 1 199

## 102 5 2 2 22.7 1 187

## 103 5 3 1 30.1 1 187

## 104 5 4 3 25.5 2 186

## 105 5 5 1 20.4 1 113

## 106 5 6 3 20.4 1 184

## 107 5 7 1 22.5 1 118

## 108 5 8 1 30.1 1 187

## 109 5 9 2 22.0 1 238

## 110 5 10 2 22.0 1 187

## 111 5 11 1 33.2 1 204

## 112 5 12 2 28.7 1 218

## 113 5 13 3 21.7 1 206

## 114 5 14 2 28.7 2 204

## 115 5 15 1 29.6 1 199

## 116 5 16 1 27.2 1 187

## 117 5 17 3 27.2 2 284

## 118 5 18 2 26.3 2 199

## 119 5 19 1 35.3 1 218

## 120 5 20 3 25.5 2 206

## 121 5 21 1 24.9 1 187

## 122 5 22 1 33.2 1 229

## 123 5 23 1 27.5 1 131

## 124 5 24 3 24.9 1 218

## 125 5 25 2 27.4 1 186

and

c.broad <- complete(imp, "broad")

c.broad

## age.1 bmi.1 hyp.1 chl.1 age.2 bmi.2 hyp.2 chl.2 age.3 bmi.3 hyp.3 chl.3

## 1 1 30.1 1 187 1 27.2 1 131 1 29.6 1 187

## 2 2 22.7 1 187 2 22.7 1 187 2 22.7 1 187

## 3 1 29.6 1 187 1 29.6 1 187 1 29.6 1 187

## 4 3 27.4 2 184 3 20.4 1 187 3 21.7 1 186

## 5 1 20.4 1 113 1 20.4 1 113 1 20.4 1 113

## 6 3 24.9 2 184 3 24.9 1 184 3 20.4 2 184

## 7 1 22.5 1 118 1 22.5 1 118 1 22.5 1 118

## 8 1 30.1 1 187 1 30.1 1 187 1 30.1 1 187

## 9 2 22.0 1 238 2 22.0 1 238 2 22.0 1 238

## 10 2 27.5 2 218 2 27.5 1 187 2 27.4 1 186

## 11 1 30.1 1 199 1 28.7 1 187 1 29.6 1 238

## 12 2 27.5 2 186 2 29.6 1 187 2 29.6 2 218

## 13 3 21.7 1 206 3 21.7 1 206 3 21.7 1 206

## 14 2 28.7 2 204 2 28.7 2 204 2 28.7 2 204

## 15 1 29.6 1 199 1 29.6 1 187 1 29.6 1 238

## 16 1 26.3 1 187 1 30.1 1 238 1 29.6 1 131

## 17 3 27.2 2 284 3 27.2 2 284 3 27.2 2 284

## 18 2 26.3 2 199 2 26.3 2 199 2 26.3 2 199

## 19 1 35.3 1 218 1 35.3 1 218 1 35.3 1 218

## 20 3 25.5 2 184 3 25.5 2 218 3 25.5 2 218

## 21 1 26.3 1 187 1 22.0 1 131 1 27.2 1 187

## 22 1 33.2 1 229 1 33.2 1 229 1 33.2 1 229

## 23 1 27.5 1 131 1 27.5 1 131 1 27.5 1 131

## 24 3 24.9 1 186 3 24.9 1 187 3 24.9 1 206

## 25 2 27.4 1 186 2 27.4 1 186 2 27.4 1 186

## age.4 bmi.4 hyp.4 chl.4 age.5 bmi.5 hyp.5 chl.5

## 1 1 35.3 1 206 1 29.6 1 199

## 2 2 22.7 1 187 2 22.7 1 187

## 3 1 26.3 1 187 1 30.1 1 187

## 4 3 27.4 2 204 3 25.5 2 186

## 5 1 20.4 1 113 1 20.4 1 113

## 6 3 21.7 2 184 3 20.4 1 184

## 7 1 22.5 1 118 1 22.5 1 118

## 8 1 30.1 1 187 1 30.1 1 187

## 9 2 22.0 1 238 2 22.0 1 238

## 10 2 24.9 2 131 2 22.0 1 187

## 11 1 22.0 1 131 1 33.2 1 204

## 12 2 27.5 1 204 2 28.7 1 218

## 13 3 21.7 1 206 3 21.7 1 206

## 14 2 28.7 2 204 2 28.7 2 204

## 15 1 29.6 1 229 1 29.6 1 199

## 16 1 28.7 1 187 1 27.2 1 187

## 17 3 27.2 2 284 3 27.2 2 284

## 18 2 26.3 2 199 2 26.3 2 199

## 19 1 35.3 1 218 1 35.3 1 218

## 20 3 25.5 2 186 3 25.5 2 206

## 21 1 35.3 1 204 1 24.9 1 187

## 22 1 33.2 1 229 1 33.2 1 229

## 23 1 27.5 1 131 1 27.5 1 131

## 24 3 24.9 1 218 3 24.9 1 218

## 25 2 27.4 1 186 2 27.4 1 186

are completed data sets in long and broad format, respectively.

**Note:** Since publishing this blog post I learned that the idea behind using mice is to compare different imputations to see how stable they are, instead of picking one imputed set as fixed for the remainder of the analysis. Therefore, I changed the focus of this post a little bit: in the old post I compared many different algorithms and their outcome; in this updated version I am only showing the Random Forest algorithm and focus on comparing the different imputed datasets. I am ignoring feature importance and feature plots because nothing changed compared to the old post.

* md.pattern() shows the pattern of missingness in the data:

md.pattern(dataset)

*## case\_id hospital province\_Jiangsu province\_Shanghai province\_Zhejiang*

*## 42 1 1 1 1 1*

*## 27 1 1 1 1 1*

*## 2 1 1 1 1 1*

*## 2 1 1 1 1 1*

*## 18 1 1 1 1 1*

*## 1 1 1 1 1 1*

*## 36 1 1 1 1 1*

*## 3 1 1 1 1 1*

*## 3 1 1 1 1 1*

*## 2 1 1 1 1 1*

*## 0 0 0 0 0*

*## province\_other age gender\_f early\_onset outcome early\_outcome*

*## 42 1 1 1 1 1 1*

*## 27 1 1 1 1 1 1*

*## 2 1 1 1 1 1 0*

*## 2 1 1 1 0 1 1*

*## 18 1 1 1 1 0 0*

*## 1 1 1 1 1 1 0*

*## 36 1 1 1 1 0 0*

*## 3 1 1 1 0 1 0*

*## 3 1 1 1 0 0 0*

*## 2 1 0 0 0 1 0*

*## 0 2 2 10 57 65*

*## days\_onset\_to\_outcome days\_onset\_to\_hospital*

*## 42 1 1 0*

*## 27 1 0 1*

*## 2 0 1 2*

*## 2 0 0 3*

*## 18 0 1 3*

*## 1 0 0 3*

*## 36 0 0 4*

*## 3 0 0 4*

*## 3 0 0 5*

*## 2 0 0 6*

*## 67 74 277*

* mice() generates the imputations

dataset\_impute <- mice(data = dataset[, -2], print = FALSE)

* by default, mice() calculates five (m = 5) imputed data sets
* we can combine them all in one output with the complete("long") function
* I did not want to impute missing values in the outcome column, so I have to merge it back in with the imputed data

datasets\_complete <- right\_join(dataset[, c(1, 2)],

complete(dataset\_impute, "long"),

by = "case\_id") %>%

select(-.id)

head(datasets\_complete)

*## case\_id outcome .imp age hospital gender\_f province\_Jiangsu*

*## 1 1 Death 1 87 0 0 0*

*## 2 2 Death 1 27 1 0 0*

*## 3 3 Death 1 35 1 1 0*

*## 4 4 <NA> 1 45 1 1 1*

*## 5 5 Recover 1 48 1 1 1*

*## 6 6 Death 1 32 1 1 1*

*## province\_Shanghai province\_Zhejiang province\_other days\_onset\_to\_outcome*

*## 1 1 0 0 13*

*## 2 1 0 0 11*

*## 3 0 0 1 31*

*## 4 0 0 0 20*

*## 5 0 0 0 57*

*## 6 0 0 0 36*

*## days\_onset\_to\_hospital early\_onset early\_outcome*

*## 1 5 1 1*

*## 2 4 1 1*

*## 3 10 1 1*

*## 4 8 1 1*

*## 5 11 1 0*

*## 6 7 1 1*

Let's compare the distributions of the five different imputed datasets:

datasets\_complete %>%

gather(x, y, age:early\_outcome) %>%

ggplot(aes(x = y, fill = .imp, color = .imp)) +

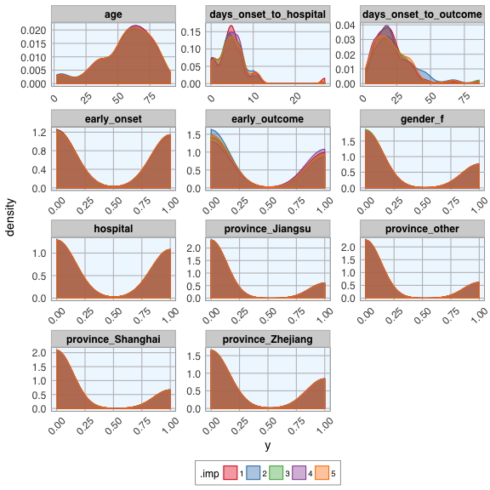
facet\_wrap(~ x, ncol = 3, scales = "free") +

geom\_density(alpha = 0.4) +

scale\_fill\_brewer(palette="Set1", na.value = "grey50") +

scale\_color\_brewer(palette="Set1", na.value = "grey50") +

my\_theme()

Gives this plot:  
[](https://datascienceplus.com/wp-content/uploads/2018/05/unnamed-chunk-13-1.png)

**Test, train and validation data sets**

Now, we can go ahead with machine learning!

The dataset contains a few missing values in the outcome column; those will be the test set used for final predictions (see the old blog post for this).

train\_index <- which(is.na(datasets\_complete$outcome))

train\_data <- datasets\_complete[-train\_index, ]

test\_data <- datasets\_complete[train\_index, -2]

The remainder of the data will be used for modeling. Here, I am splitting the data into 70% training and 30% test data.

Because I want to model each imputed dataset separately, I am using the nest() and map() functions.

set.seed(42)

val\_data <- train\_data %>%

group\_by(.imp) %>%

nest() %>%

mutate(val\_index = map(data, ~ createDataPartition(.$outcome, p = 0.7, list = FALSE)),

val\_train\_data = map2(data, val\_index, ~ .x[.y, ]),

val\_test\_data = map2(data, val\_index, ~ .x[-.y, ]))

**Machine Learning algorithms**

**Random Forest**

To make the code tidier, I am first defining the modeling function with the parameters I want.

model\_function <- function(df) {

caret::train(outcome ~ .,

data = df,

method = "rf",

preProcess = c("scale", "center"),

trControl = trainControl(method = "repeatedcv", number = 5, repeats = 3, verboseIter = FALSE))

}

Next, I am using the nested tibble from before to map() the model function, predict the outcome and calculate confusion matrices.

set.seed(42)

val\_data\_model <- val\_data %>%

mutate(model = map(val\_train\_data, ~ model\_function(.x)),

predict = map2(model, val\_test\_data, ~ data.frame(prediction = predict(.x, .y[, -2]))),

predict\_prob = map2(model, val\_test\_data, ~ data.frame(outcome = .y[, 2],

prediction = predict(.x, .y[, -2], type = "prob"))),

confusion\_matrix = map2(val\_test\_data, predict, ~ confusionMatrix(.x$outcome, .y$prediction)),

confusion\_matrix\_tbl = map(confusion\_matrix, ~ as.tibble(.x$table)))

**Comparing accuracy of models**

To compare how the different imputations did, I am plotting

* the confusion matrices:

val\_data\_model %>%

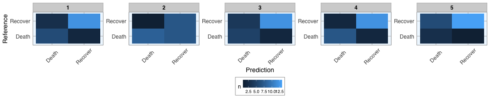
unnest(confusion\_matrix\_tbl) %>%

ggplot(aes(x = Prediction, y = Reference, fill = n)) +

facet\_wrap(~ .imp, ncol = 5, scales = "free") +

geom\_tile() +

my\_theme()

Gives this plot:  
[](https://datascienceplus.com/wp-content/uploads/2018/05/unnamed-chunk-18-1.png)

* and the prediction probabilities for correct and wrong predictions:

val\_data\_model %>%

unnest(predict\_prob) %>%

gather(x, y, prediction.Death:prediction.Recover) %>%

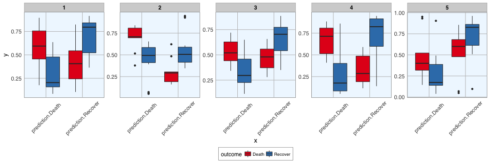
ggplot(aes(x = x, y = y, fill = outcome)) +

facet\_wrap(~ .imp, ncol = 5, scales = "free") +

geom\_boxplot() +

scale\_fill\_brewer(palette="Set1", na.value = "grey50") +

my\_theme()

Gives this plot:  
[](https://datascienceplus.com/wp-content/uploads/2018/05/unnamed-chunk-19-1.png)

Hope, you found that example interesting and helpful!

sessionInfo()

*## R version 3.5.0 (2018-04-23)*

*## Platform: x86\_64-apple-darwin15.6.0 (64-bit)*

*## Running under: macOS High Sierra 10.13.4*

*##*

*## Matrix products: default*

*## BLAS: /System/Library/Frameworks/Accelerate.framework/Versions/A/Frameworks/vecLib.framework/Versions/A/libBLAS.dylib*

*## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib*

*##*

*## locale:*

*## [1] de\_DE.UTF-8/de\_DE.UTF-8/de\_DE.UTF-8/C/de\_DE.UTF-8/de\_DE.UTF-8*

*##*

*## attached base packages:*

*## [1] stats graphics grDevices utils datasets methods base*

*##*

*## other attached packages:*

*## [1] bindrcpp\_0.2.2 knitr\_1.20 RWordPress\_0.2-3 caret\_6.0-79*

*## [5] mice\_2.46.0 lattice\_0.20-35 plyr\_1.8.4 forcats\_0.3.0*

*## [9] stringr\_1.3.1 dplyr\_0.7.4 purrr\_0.2.4 readr\_1.1.1*

*## [13] tidyr\_0.8.0 tibble\_1.4.2 ggplot2\_2.2.1 tidyverse\_1.2.1*

*## [17] outbreaks\_1.3.0*

*##*

*## loaded via a namespace (and not attached):*

*## [1] nlme\_3.1-137 bitops\_1.0-6 lubridate\_1.7.4*

*## [4] RColorBrewer\_1.1-2 dimRed\_0.1.0 httr\_1.3.1*

*## [7] rprojroot\_1.3-2 tools\_3.5.0 backports\_1.1.2*

*## [10] R6\_2.2.2 rpart\_4.1-13 lazyeval\_0.2.1*

*## [13] colorspace\_1.3-2 nnet\_7.3-12 withr\_2.1.2*

*## [16] tidyselect\_0.2.4 mnormt\_1.5-5 compiler\_3.5.0*

*## [19] cli\_1.0.0 rvest\_0.3.2 xml2\_1.2.0*

*## [22] labeling\_0.3 bookdown\_0.7 scales\_0.5.0*

*## [25] sfsmisc\_1.1-2 DEoptimR\_1.0-8 psych\_1.8.4*

*## [28] robustbase\_0.93-0 randomForest\_4.6-14 digest\_0.6.15*

*## [31] foreign\_0.8-70 rmarkdown\_1.9 pkgconfig\_2.0.1*

*## [34] htmltools\_0.3.6 highr\_0.6 rlang\_0.2.0*

*## [37] readxl\_1.1.0 ddalpha\_1.3.3 rstudioapi\_0.7*

*## [40] XMLRPC\_0.3-0 bindr\_0.1.1 jsonlite\_1.5*

*## [43] ModelMetrics\_1.1.0 RCurl\_1.95-4.10 magrittr\_1.5*

*## [46] Matrix\_1.2-14 Rcpp\_0.12.16 munsell\_0.4.3*

*## [49] abind\_1.4-5 stringi\_1.2.2 yaml\_2.1.19*

*## [52] MASS\_7.3-50 recipes\_0.1.2 grid\_3.5.0*

*## [55] parallel\_3.5.0 crayon\_1.3.4 haven\_1.1.1*

*## [58] splines\_3.5.0 hms\_0.4.2 pillar\_1.2.2*

*## [61] reshape2\_1.4.3 codetools\_0.2-15 stats4\_3.5.0*

*## [64] CVST\_0.2-1 magic\_1.5-8 XML\_3.98-1.11*

*## [67] glue\_1.2.0 evaluate\_0.10.1 blogdown\_0.6*

*## [70] modelr\_0.1.2 foreach\_1.4.4 cellranger\_1.1.0*

*## [73] gtable\_0.2.0 kernlab\_0.9-26 assertthat\_0.2.0*

*## [76] DRR\_0.0.3 xfun\_0.1 gower\_0.1.2*

*## [79] prodlim\_2018.04.18 broom\_0.4.4 e1071\_1.6-8*

*## [82] class\_7.3-14 survival\_2.42-3 geometry\_0.3-6*

*## [85] timeDate\_3043.102 RcppRoll\_0.2.2 iterators\_1.0.9*

*## [88] lava\_1.6.1 ipred\_0.9-6*