

# Rake Manual

Toby Dylan Hocking

March 3, 2011

## 1 Introduction

Rake is an R package that allows you to easily perform Rake analysis.

This manual teaches how to use the Rake package by walking you through a tutorial rake analysis on the data that is provided with the package. For a more thorough overview of the Rake functions and their arguments, please refer to the package documentation pages. For a more detailed discussion of the theory, refer to [1, pp.~269–271].

## 2 Tutorial

For the tutorial analysis, we use the ASA certification data included with the Rake package, which is the same data as described in [1, p.~439].

A standard Rake analysis consists of 3 steps:

1. Creating the "**rake**" class object using the **rake** function.
2. Performing the rake weight adjustment using the **rakeadj** function.
3. Obtaining new sample weights and reweighted data using the **predict.rake** function.

These 3 steps have been wrapped into a function called **simpleRake**. This effectively turns a 3-step analysis into a 1-step analysis, but should only be used if the "**rake**" class object's breakdown of sample weights by class is of no interest.

## 2.1 rake

The "rake" class object contains a summary of sample weights, broken down by classes for 2 categorical variables of interest.

```
> library(rake)
> data(certify)
> certify$COLLEGE[certify$COLLEGE != "P"] <- "N"
> certify$WORKENV[!certify$WORKENV %in% c("I", "A")] <- "O"
> r <- rake(certify, "COLLEGE", "WORKENV")
> print(r)
```

```
data: certify
rowvar: WORKENV
colvar: COLLEGE
      N      P
A  434 1787 2221
I 1011   798 1809
O   520  451  971
      1965 3036 5001
```

The idea behind raking is to re-weight the sample so that the marginal total sample weights, as summarized in the "rake" class object, are equivalent to the known marginal total weights of the population.

## 2.2 rakeadj

This re-weighting is accomplished with the `rakeadj` function:

```
> r <- rakeadj(r, statpoptotal, TRUE)
```

The rake adjustment converged in 20 steps.

```
> print(r)
```

```
data: certify
rowvar: WORKENV
colvar: COLLEGE
      N      P
```

```

A 361.8412 1338.4826 1700.324
I 848.6274 601.7674 1450.395
O 1039.9679 810.3134 1850.281
  2250.4366 2750.5634 5001.000

> print(r/sum(r) * sum(statpoptotal$weight[statpoptotal$name ==
+ rownames(r)]))

data: certify
rowvar: WORKENV
colvar: COLLEGE
      N      P
A 1346.431 4980.569 6327
I 3157.790 2239.210 5397
O 3869.779 3015.221 6885
  8374.000 10235.000 18609

```

Note that in this dataset, we rake on the 2 categorical variables "WORKENV" and "COLLEGE" because we know that the population values for these variables:

```

> print(statpoptotal)

name weight
1    A    6327
2    I    5397
3    O    6885
4    P   10235
5    N    8374

```

The "rake" class object maintains the weight sum of the sample, but its components are adjusted such that the marginal total sample weights now have the same ratio as those in the population.

## 2.3 predict.rake

To obtain the corresponding re-weighted unit weights for the original sample data, use the `predict.rake` function:

```

> for (i in 1:5) {
+   cat("Variable:", names(certify)[i], "\n")
+   data <- summary(as.factor(certify[, i]))
+   pred <- predict(r, i, forcefactor = TRUE)$data.est
+   print(rbind(data, pred))
+   cat("\n")
+ }

```

Variable: CERTIFY

	0	1	2	3	4	5
data	12.00000	1321.000	1114.000	269.0000	337.0000	1948.000
pred	14.64323	1288.808	1114.226	272.1095	345.7145	1965.500

Variable: APPROVE

	0	1	2	3	4	5
data	102.0000	766.0000	1025.000	820.0000	470.0000	1818.000
pred	108.3808	727.1895	1023.779	822.4286	468.2622	1850.960

Variable: SPECCERT

	0	1	2	3	4	5
data	63.00000	552.000	901.0000	451.0000	488.0000	2546.000
pred	67.03614	540.527	913.3438	464.5248	491.8947	2523.673

Variable: WOULDYOU

	0	1	2	3	4	5
data	42.00000	1566.000	1119.000	212.0000	573.0000	1489.000
pred	45.91886	1511.456	1113.057	203.1978	600.5737	1526.797

Variable: RECERT

	0	1	2	3	4	5
data	92.00000	951.0000	936.0000	596.0000	404.0000	2022.000
pred	94.25057	921.3488	924.8003	598.8645	410.3751	2051.361

This function returns a list of three elements:

**weight** Vector of re-weighted sample weights, in the same order as the sample data's rows.

**data** Vector of sample data, in the same order as the sample data's rows.

**data.est** Vector of sample data with the new values after re-weighting. For categorical data, this is a factor summary. For numeric data, this is a vector of re-weighted data, in the same order as the sample data's rows.

## 2.4 simpleRake

This wraps the raking process inside a single function:

```
> example(simpleRake)

smplRk> data(certify)

smplRk> # Merge some classes since we have limited population data
smplRk> # We define classes of COLLEGE as P or not P = N
smplRk> certify$COLLEGE[ certify$COLLEGE != "P" ] <- "N"

smplRk> # We define classes of WORKENV to be I, A, or else = 0
smplRk> certify$WORKENV[ ! certify$WORKENV %in% c("I","A") ] <- "0"

smplRk> rakeresult <- simpleRake(certify, statpoptotal, "WORKENV", "COLLEGE",
smplRk+ "WOULDYOU", TRUE)

smplRk> data <- rakeresult$data

smplRk> pred <- rakeresult$data.est

smplRk> print(rbind(data,pred))
              0          1          2          3          4          5
data 42.00000 1566.000 1119.000 212.0000 573.0000 1489.000
pred 45.91886 1511.456 1113.057 203.1978 600.5737 1526.797
```

Note that you don't get access to the "**rake**" class object. This is usually used if you are only interested in the sample weights after re-weighting.

## 3 Conclusion

Rake is a package that implements the robust technique of sample re-weighting based on the raking technique.

The 3-step raking should be used for most applications to get a better feel for the sample data class weights, but the 1-step `simpleRake` can be performed if only the unit sample weights are needed.

## References

- [1] Sharon~L. Lohr. *Sampling: Design and Analysis*. Duxbury Press, Pacific Grove, CA, 1999.