First let's attach our packages and generate our example data in R.

```
library(wrapr)

d <- build_frame(
   "x1"  , "x2", "y"   |
     1   , 1   , TRUE  |
     1   , 0   , FALSE |
     1   , 0   , TRUE  |
     1   , 1   , FALSE |
     0   , 0   , TRUE  |
     0   , 1   , TRUE  |
     0   , 1   , FALSE |
     0   , 0   , FALSE |
     0   , 0   , TRUE  )
# cat(wrapr::draw_frame(d))

knitr::kable(d)
```

x1	x2	y
1	1	TRUE
1	0	FALSE
1	0	TRUE
1	1	FALSE
0	0	TRUE
0	1	TRUE
0	1	FALSE
0	0	FALSE
0	0	TRUE

Now, we fit our logistic regression model.

```
model <- glm(y ~ x1 + x2,
             data = d,
             family = binomial())
summary(model)

## 
## Call:
## glm(formula = y ~ x1 + x2, family = binomial(), data = d)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.4213  -1.2572   0.9517   1.0996   1.2572  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.5572     1.0784   0.517    0.605
## x1           -0.3715     1.3644  -0.272    0.785
```

```
## x2            -0.3715     1.3644  -0.272    0.785
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 12.365  on 8  degrees of freedom
## Residual deviance: 12.201  on 6  degrees of freedom
## AIC: 18.201
##
## Number of Fisher Scoring iterations: 4
```

We land our model predictions as a new column.

```
d$prediction <- predict(model,
                        newdata = d,
                        type = 'response')
knitr::kable(d)
```

x1	x2	y	prediction
1	1	TRUE	0.4537010
1	0	FALSE	0.5462990
1	0	TRUE	0.5462990
1	1	FALSE	0.4537010
0	0	TRUE	0.6358007
0	1	TRUE	0.5462990
0	1	FALSE	0.5462990
0	0	FALSE	0.6358007
0	0	TRUE	0.6358007

We can see this model is calibrated or unbiased in the sense `E[prediction] = E[outcome]`.

```
colMeans(d[, qc(y, prediction)]) %.>%
  knitr::kable(.)
```

	x
y	0.5555556
prediction	0.5555556

And it is even calibrated in the sense we expect for logistic regression, `E[prediction * x] = E[outcome * x]` (where `x` is any explanatory variable).

```
for(v in qc(x1, x2)) {
  print(paste0(
    v, ' diff: ',
    mean(d[[v]] * d$prediction) - mean(d[[v]] * d$y)))
}
```

```
## [1] "x1 diff: 2.77555756156289e-17"
## [1] "x2 diff: 5.55111512312578e-17"
```

However, we can see this model is not “fully calibrated” in an additional sense requiring that

`E[outcome | prediction] = prediction` for all observed values of `prediction`.

```
cal <- aggregate(y ~ prediction, data = d, FUN = mean)

knitr::kable(cal)
```

prediction	y
0.4537010	0.5000000
0.5462990	0.5000000
0.6358007	0.6666667

We can re-calibrate such a model (in practice you would want to do this on out of sample data using isotonic regression, or using cross-frame methods to avoid nested model bias).

```
cal_map <- cal$prediction := cal$y
d$calibrated <- cal_map[as.character(d$prediction)]

knitr::kable(d)
```

x1	x2	y	prediction	calibrated
1	1	TRUE	0.4537010	0.5000000
1	0	FALSE	0.5462990	0.5000000
1	0	TRUE	0.5462990	0.5000000
1	1	FALSE	0.4537010	0.5000000
0	0	TRUE	0.6358007	0.6666667
0	1	TRUE	0.5462990	0.5000000
0	1	FALSE	0.5462990	0.5000000
0	0	FALSE	0.6358007	0.6666667
0	0	TRUE	0.6358007	0.6666667

This new calibrated prediction is also calibrated in the standard sense.

```
colMeans(d[, qc(y, prediction, calibrated)]) %.>%
  knitr::kable(.)
```

	x
y	0.5555556
prediction	0.5555556
calibrated	0.5555556

And, at least in this case, still obeys the explanatory roll-up conditions.

```
for(v in qc(x1, x2)) {
  print(paste0(
    v, ' diff: ',
    mean(d[[v]] * d$calibrated) - mean(d[[v]] * d$y)))
}

## [1] "x1 diff: 0"
## [1] "x2 diff: 0"
```

The new calibrated predictions are even of lower deviance than the original predictions.

```
deviance <- function(prediction, truth) {
  sum(-2 * (truth * log(prediction) +
              (1 - truth) * log(1 - prediction)))
}
```

```
deviance(prediction = d$prediction, truth = d$y)
```

```
## [1] 12.20102
```

```
deviance(prediction = d$calibrated, truth = d$y)
```

```
## [1] 12.13685
```

The reason the original logistic model could not make the calibrated predictions is: the calibrated predictions are not a linear function of the explanatory variables in link space.