In the linear regression section of our book *Practical Data Science in R*, we use the example of predicting income from a number of demographic variables (age, sex, education and employment type). In the text, we choose to regress against log10(income) rather than directly against income.

One obvious reason for not regressing directly against income is that (in our example) income is restricted to be non-negative, a restraint that linear regression can’t enforce. Other reasons include the wide distribution of values and the relative or multiplicative structure of errors on outcomes. A common practice in this situation is to use *Poisson regression*, or generalized linear regression with a log-link function. Like all generalized linear regressions, Poisson regression is unbiased and *calibrated*: it preserves the conditional expectations and rollups of the training data. A calibrated model is important in many applications, particularly when financial data is involved.

Regressing against the log of the outcome will not be calibrated; however it has the advantage that the resulting model will have lower relative error than a Poisson regression against income. Minimizing relative error is appropriate in situations when differences are naturally expressed in percentages rather than in absolute amounts. Again, this is common when financial data is involved: raises in salary tend to be in terms of percentage of income, not in absolute dollar increments.

Unfortunately, a full discussion of the differences between Poisson regression and regressing against log amounts was outside of the scope of our book, so we will discuss it in this note.

**The data**

As we did in the book, we’ll use data from the 2016 US Census American Community Survay (ACS) Public Use Microdata Sample (PUMS) for our example. More information about the data can be found [here](https://github.com/WinVector/PDSwR2/tree/master/PUMS). First, we’ll get the training and test data, and show how the expected income varies along different groupings (by sex, by employment, and by education):

library(zeallot)

library(wrapr)

location <- "https://github.com/WinVector/PDSwR2/raw/master/PUMS/incomedata.rds"

incomedata <- readRDS(url(location))

c(test, train) %<-% split(incomedata, incomedata$gp)

# A convenience function to calculate and display

# the conditional expected incomes

show\_conditional\_means <- function(d, outcome = "income") {

cols <- qc(sex, employment, education)

lapply(

cols := cols,

function(colname) {

aggregate(d[, outcome, drop = FALSE],

d[, colname, drop = FALSE],

FUN = mean)

})

}

display\_tables <- function(tlist) {

for(vi in tlist) {

print(knitr::kable(vi))

}

}

display\_tables(show\_conditional\_means(train))

| **sex** | **income** |
| --- | --- |
| Male | 55755.51 |
| Female | 47718.52 |

| **employment** | **income** |
| --- | --- |
| Employee of a private for profit | 51620.39 |
| Federal government employee | 64250.09 |
| Local government employee | 54740.93 |
| Private not-for-profit employee | 53106.41 |
| Self employed incorporated | 66100.07 |
| Self employed not incorporated | 41346.47 |
| State government employee | 53977.20 |

| **education** | **income** |
| --- | --- |
| no high school diploma | 31883.18 |
| Regular high school diploma | 38052.13 |
| GED or alternative credential | 37273.30 |
| some college credit, no degree | 42991.09 |
| Associate’s degree | 47759.61 |
| Bachelor’s degree | 65668.51 |
| Master’s degree | 79225.87 |
| Professional degree | 97772.60 |
| Doctorate degree | 91214.55 |

**Three models**

Now we’ll model income as a function of age, sex, employment, and education three different ways:

# linear model for income

model\_income <- lm(income ~ age+sex+employment+education,

data=train)

# linear model for log10(income)

model\_logincome <- lm(log10(income) ~ age+sex+employment+education,

data=train)

# Quasipoisson model for income

model\_pincome <- glm(income ~ age+sex+employment+education,

data=train,

family=quasipoisson)

Note that we are fitting a quasipoisson model for income; strictly speaking, a Poisson model assumes that the mean and variance of the data are the same, which is not true in general. A quasipoisson model relaxes the restriction on the variance of the data. We’ll still refer to this as a Poisson model for brevity.

Now we can use all three models to predict income for the training data.

train <- transform(train,

pred\_lm = predict(model\_income, train),

pred\_lmlog = 10^predict(model\_logincome, train),

pred\_pois = predict(model\_pincome,

train, type="response"))

knitr::kable(

summary(train[, qc(income, pred\_lm, pred\_pois, pred\_lmlog)]))

|  | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
|  | Min. : 1200 | Min. : -4682 | Min. : 15704 | Min. : 11977 |
|  | 1st Qu.: 26700 | 1st Qu.: 36877 | 1st Qu.: 36480 | 1st Qu.: 30546 |
|  | Median : 41200 | Median : 50180 | Median : 47450 | Median : 40281 |
|  | Mean : 52373 | Mean : 52373 | Mean : 52373 | Mean : 44478 |
|  | 3rd Qu.: 66000 | 3rd Qu.: 65962 | 3rd Qu.: 63669 | 3rd Qu.: 54397 |
|  | Max. :250000 | Max. :125969 | Max. :159583 | Max. :129216 |

Note that even though all actual incomes were positive, the linear model (model\_income) sometimes predicted negative income.

**Estimating aggregates**

Now let’s compare how the predicted incomes roll up.

display\_tables(

show\_conditional\_means(train,

qc(income, pred\_lm, pred\_pois, pred\_lmlog))

)

| **sex** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| Male | 55755.51 | 55755.51 | 55755.51 | 47081.99 |
| Female | 47718.52 | 47718.52 | 47718.52 | 40895.21 |

| **employment** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| Employee of a private for profit | 51620.39 | 51620.39 | 51620.39 | 43169.85 |
| Federal government employee | 64250.09 | 64250.09 | 64250.09 | 58542.64 |
| Local government employee | 54740.93 | 54740.93 | 54740.93 | 49988.61 |
| Private not-for-profit employee | 53106.41 | 53106.41 | 53106.41 | 47475.45 |
| Self employed incorporated | 66100.07 | 66100.07 | 66100.07 | 53189.40 |
| Self employed not incorporated | 41346.47 | 41346.47 | 41346.47 | 31151.47 |
| State government employee | 53977.20 | 53977.20 | 53977.20 | 50023.27 |

| **education** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| no high school diploma | 31883.18 | 31883.18 | 31883.18 | 26978.21 |
| Regular high school diploma | 38052.13 | 38052.13 | 38052.13 | 32437.46 |
| GED or alternative credential | 37273.30 | 37273.30 | 37273.30 | 30816.91 |
| some college credit, no degree | 42991.09 | 42991.09 | 42991.09 | 36184.14 |
| Associate’s degree | 47759.61 | 47759.61 | 47759.61 | 40585.89 |
| Bachelor’s degree | 65668.51 | 65668.51 | 65668.51 | 55130.77 |
| Master’s degree | 79225.87 | 79225.87 | 79225.87 | 69437.91 |
| Professional degree | 97772.60 | 97772.60 | 97772.60 | 81612.18 |
| Doctorate degree | 91214.55 | 91214.55 | 91214.55 | 80679.19 |

The rollups of the predictions for the linear and Poisson models (model\_income and model\_pincome) match the rollups of the training data. The predictions from model\_logincome roll up too low. In fact, one can prove that by Jensen’s inequality, a linear model fit to log-income will always have a systematic bias (underprediction) when estimating expected income. This means that if one of the intended uses of the model is to estimate aggregates (grouped sums, conditional means), then a calibrated model like a linear or Poisson model is more appropriate.

**Predictions on individuals**

If the primary purpose of the model is predictions on individuals, then biased models may still be acceptable, or even preferable. When predicting income, it’s often the case that you want to express uncertainty in relative terms: that is, predict income to within 5%, rather than predict income to within $50. So let’s see how each of the models performs in terms of relative error (on the training data):

rel\_err <- function(x, y) {

mean(abs(y-x)/y)

}

lapply(train[, qc(pred\_lm, pred\_lmlog, pred\_pois)],

function(p) rel\_err(p, train$income)) %.>%

as.data.frame(.) %.>%

knitr::kable(.)

| **pred\_lm** | **pred\_lmlog** | **pred\_pois** |
| --- | --- | --- |
| 0.74858 | 0.615897 | 0.7437119 |

model\_logincome has a lower average relative error on estimated income than either of the models fit directly to income — not a great relative error, but that’s because our set of input variables isn’t informative enough. We can also compare the models’ performances in terms of root mean squared error (an absolute difference):

rmse <- function(x, y) {

sqrt(mean((y-x)^2))

}

lapply(train[, qc(pred\_lm, pred\_lmlog, pred\_pois)],

function(p) rmse(p, train$income)) %.>%

as.data.frame(.) %.>%

knitr::kable(.)

| **pred\_lm** | **pred\_lmlog** | **pred\_pois** |
| --- | --- | --- |
| 31625.35 | 32616.57 | 31395.38 |

The models that are fit directly to income have lower RMSE than model\_logincome, but not dramatically so. In other words, model\_logincome seems to improve relative error, at the cost of a slightly larger RMSE.

**Performance on new data**

The real test of the three models for income is how they perform on data not used to train the models. First, we’ll compare the rollups.

test <- transform(test,

pred\_lm = predict(model\_income, test),

pred\_lmlog = 10^predict(model\_logincome, test),

pred\_pois = predict(model\_pincome,

test, type="response"))

rollups <- show\_conditional\_means(test,

qc(income, pred\_lm,

pred\_pois,pred\_lmlog))

display\_tables(rollups)

| **sex** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| Male | 55408.95 | 55903.57 | 55899.83 | 47173.10 |
| Female | 46261.99 | 46876.96 | 47111.01 | 40361.71 |

| **employment** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| Employee of a private for profit | 50717.96 | 51314.69 | 51362.44 | 42947.99 |
| Federal government employee | 66268.05 | 64635.60 | 64881.32 | 58993.59 |
| Local government employee | 52565.89 | 53730.43 | 54119.83 | 49450.23 |
| Private not-for-profit employee | 52887.52 | 52830.80 | 53259.07 | 47642.49 |
| Self employed incorporated | 67744.61 | 65538.68 | 66096.20 | 53189.42 |
| Self employed not incorporated | 41417.25 | 41671.41 | 41507.17 | 31265.77 |
| State government employee | 51314.92 | 54106.89 | 53973.39 | 50029.83 |

| **education** | **income** | **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- | --- | --- |
| no high school diploma | 29903.70 | 31738.07 | 31783.60 | 26923.95 |
| Regular high school diploma | 36979.33 | 37538.76 | 37746.81 | 32162.33 |
| GED or alternative credential | 39636.86 | 37336.08 | 37177.50 | 30666.80 |
| some college credit, no degree | 43490.42 | 43199.50 | 43270.86 | 36421.74 |
| Associate’s degree | 48384.19 | 47167.06 | 47234.56 | 40140.43 |
| Bachelor’s degree | 65268.96 | 66077.47 | 66141.27 | 55535.11 |
| Master’s degree | 77180.40 | 79521.83 | 79594.17 | 69750.68 |
| Professional degree | 94976.75 | 98649.58 | 99009.56 | 82575.73 |
| Doctorate degree | 87535.83 | 91403.52 | 91742.54 | 81524.25 |

# see how close the rollups get to ground truth for employment

err\_mag <- function(x, y) {

delta = y-x

sqrt(sum(delta^2))

}

employment <- rollups$employment

lapply(employment[, qc(pred\_lm, pred\_pois, pred\_lmlog)],

function(p) err\_mag(p, employment$income)) %.>%

as.data.frame(.) %.>%

knitr::kable(.)

| **pred\_lm** | **pred\_pois** | **pred\_lmlog** |
| --- | --- | --- |
| 4135.96 | 3831.967 | 21611.7 |

None of the models reproduced the true rollups perfectly. Just looking at the employment rollup, you can see that the rollups from model\_income and model\_pincome are usually fairly close to the actual rollups, while the rollups from model\_logincome are off — and consistently under. The pattern holds for the rollups by sex and education as well.

Let’s compare the models on individual predictions.

# relative error

lapply(test[, qc(pred\_lm, pred\_lmlog, pred\_pois)],

function(p) rel\_err(p, test$income)) %.>%

as.data.frame(.) %.>%

knitr::kable(.)

| **pred\_lm** | **pred\_lmlog** | **pred\_pois** |
| --- | --- | --- |
| 0.7508259 | 0.6222302 | 0.7543232 |

# root mean square error

lapply(test[, qc(pred\_lm, pred\_lmlog, pred\_pois)],

function(p) rmse(p, test$income)) %.>%

as.data.frame(.) %.>%

knitr::kable(.)

| **pred\_lm** | **pred\_lmlog** | **pred\_pois** |
| --- | --- | --- |
| 31589.5 | 32389.97 | 31341.14 |

Again, model\_logincome returns predictions with the lowest relative error, but a slightly higher RMSE than the other two models.

**Conclusion**

In this note we have shown the consequences of different modeling decisions, in particular the trade-off between bias and relative error. Notice that transforming the outcome and using a link function have different advantages. Which procedure you use depends on what is most important to your application: correctly estimating summary statistics, minimizing relative error, or minimizing squared error.

In our next article, we will show that common tree models are also non-calibrated, which means that despite their high accuracy on individual predictions, they do not correctly estimate summary statistics in an unbiased way. Later, we will address how to mitigate this issue.