Project: Rephetio: Repurposing drugs on a hetnet [rephetio]

Computing standardized logistic regression coefficients

Daniel Himmelstein Researcher April 21, 2016

It appears that there are multiple ways to compute standardized coefficients for logistic regression [1, 2, 3]. We'd like a way to make coefficients comparable across features. **@alizee**, can you look over this debate and decide on a method? In addition, we should be aware of other approaches for comparing feature importance in logistic regression.

Daniel Himmelstein Researcher April 21, 2016

Agresti method

@alizee and I were talking this morning, prior to this discussion, and he arrived at the Agresti method [1]. Agresti describes the method as follows (§ 4.5.2 Standardized Interpretations [2]):

> With multiple predictors, it is tempting to compare magnitudes of $\{\hat{\beta}_j\}$ to compare effects of predictors. For binary predictors, this gives a comparison of conditional log odds ratios, given the other predictors in the model. For quantitative predictors, this is relevant if the predictors have the same units, so a 1-unit change means the same thing for each. Otherwise, it is not meaningful.
>
> An alternative comparison of effects of quantitative predictors having different units uses *standardized coefficients*. The model is fitted to standardized predictors, replacing each x_j by $(x_j - \bar{x}_j) \,/\, s_{x_j}$. A 1-unit change in the standardized predictor is a standard deviation change in the original predictor. Then, each regression coefficient represents the effect of a standard deviation change in a predictor, controlling for the other variables. The standardized estimate for predictor x_j is the unstandardized estimate $\hat{\beta}_j$ multiplied by s_{x_j}.

This method is simple and intuitive. I believe it corresponds to coefficients from converting each predictor to *z*-scores and then fitting the logistic regression model. I updated our `hetior` package — which was previously using a flawed approach — to use the Agresti method.

Antoine Lizee Researcher May 1, 2016

A note on standardized coefficients for logistic regression

This note aims at (i) understanding what standardized coefficients are, (ii) sketching the landscape of standardization approaches for logistic regression, (iii) drawing conclusions and guidelines to follow in general, and for our study in particular. I apologize for the length of the piece, but I find the exhaustivity of the result desirable — don't hesitate to jump to the conclusion if short in time.

This note draws mainly from the most recent review of Menart (2011) [1], as well as a few other papers [2, 3] for exhaustivity and clarification purposes.

Standardization is a transformation of the coefficients (not the model)

The goal of standardized coefficients is to specify a same model with different nominal values of its parameters. These transformed values present the main advantage of relying on an objectively defined scale rather than depending on the original metric of the corresponding predictor.

Status: Open

Views

1621

Topics

Regression
Logistic Regression GLM
Coefficients Statistics

Referenced by

- Our hetnet edge prediction methodology: the modeling framework for Project Rephetio
- Research report: Rephetio: Repurposing drugs on a hetnet

Cite this as

Daniel Himmelstein, Antoine Lizee (2016) Computing standardized logistic regression coefficients. *Thinklab*. doi:10.15363/thinklab.d205

Standardizing the coefficients is a matter of presentation and interpretation of a given model; it does not modify the model, its hypotheses, or its output. It *happens* that the approaches presented here sometimes results in parameters that coincide with the *expected* values for a different model, eg fitted with transformed input data. These are properties of some methods, not the goal of standardization.

Standardization enables three things that are not possible with unstandardized coefficients. First, it offers an objective scale to coefficients corresponding to variables that have no natural metric. Second, and most importantly, it lets the user compare the effect of different predictor variables within the same model, by simply comparing the values of the corresponding standardized coefficients. Third, standardized coefficients provide an alternative interpretation of the model parameters, based on 'standard deviation' units of the predicted variables (and optionally of the predictor).

Nevertheless, a few strong cases remain in favor of using unstandardized coefficients. The first one is the practical meaning of unstandardized coefficients when the predictor variable has a meaningful natural metric (like time, in years), or if the predictor is a boolean category (gender). In these cases, the interpretative power associated with unstandardized coefficients and their related parameters (e.g. odd ratios) promotes intuitive understanding and easy communication of the results. The second important problem that comes with (over)using standardized coefficients is their *sample specificity*. Even when studying the same phenomena, the standardized coefficients will have a different practical unit depending on the sample variance. This can pose significant challenges in interpreting the data, particularly when comparing models.

Standardization approaches

While standardized coefficients in classic linear regression are well-defined, logistic regression, like other generalized linear models, present additional complexity as a result of the non-linear link function (logit), and non-normal error function (binomial).

The — historically — first *Goodman*'s standardization method standardizes each coefficient by its standard-error, effectively performing the equivalent of a Wald-Test. Beyond this first method, irrelevant here, two approaches have been taken. We call the first *partial standardization*; it leaves the predicted boolean variable untouched when standardizing the coefficients, and solely relies on the dispersion of each corresponding predictor. The second approach, *full standardization*, incorporates the dispersion of the predicted variable in an attempt to improve the general relevance of the resulting standardized coefficients.

All these approaches result in sets of coefficients that are proportional to each other for a given model.

Partial standardization

We list here three variants of partial standardization. In all cases, each standardized coefficient b^* is a scaling of the corresponding unstandardized coefficient b by the sample standard deviation of the corresponding predictor σ_x:

$$b^*_{PS} = b \cdot \sigma_x / C$$

...where C is a constant that depends on the variation.

Agresti: $C = 1$. This is the most straightforward and clear approach, where the coefficient is specified in 'per standard deviation' unit of the predictor. The *Agresti* coefficients are equal to the expected values for the coefficients of the same model fitted on the standardized predictors, if the Maximum Likelihood Estimator is the fitting algorithm.

SAS: $C = \mathrm{sd}(logis) = \frac{\pi}{\sqrt{3}}$. This method scales by the standard deviation of the logistic distribution of unit scale. The resulting coefficients are equal to the expected values for the coefficients of the logistic regression on the standardized predictors, if fitted with Ordinary Least Square. This approach is used in the software SAS.

Long: $C = \mathrm{sd}(logis) + \mathrm{sd}(norm) = \frac{\pi}{\sqrt{3}} + 1$, where the standard deviation of the normal distribution is added to the normalization.

Full standardization

Here, we use the variance of the predicted variable to further scale the coefficients. This provides a closer equivalent to standardized coefficients of classic linear regression.

The main difficulty resides in the non-meaningful variance of the class outcome, which points to using the variance of the linear predictor $\text{logit}(Y)$. Nevertheless, because $\text{logit}(Y)$ takes only values of $\pm\infty$, its variance is not defined and we need proxies for it, based on $\hat{Y}$, the *predicted* outcome. *Menard*[4] proposes the variance estimate $\sigma^2_{\text{logit}(\hat{Y})}/R^2$, while *Long* proposes to use the variance of the underlying latent variable $\sigma^2_{\text{logit}(\hat{Y})} + \pi^2/3$. The results can be expressed with the same formalism as above:

Menard: $C = \sigma_{\text{logit}(\hat{Y})}/R$ This method, presented as superior than the others by its author [1], has the downside of relying on the R-square of the model, whose computation is ambiguous (Cox & Snell, Nagelkerke, and others have proposed *pseudo*-R-squared measures for logistic regression).

Long (full): $C = \sqrt{\sigma^2_{\text{logit}(\hat{Y})} + \pi^2/3}$

Conclusion

Standardized coefficients are extremely valuable, mainly to (i) give a meaning to the coefficient affecting a predictor that has no natural metric and (ii) compare effects of predictors reported in different units.

To achieve these two goals, I advise using the most straightforward, simple *Agresti* method of standardization:

$$b^*_A = b \cdot \sigma_X$$

Further, to improve the general relevance of the standardized coefficients, one can account for the sample dispersion of the outcome variable, and use *fully standardized* coefficients as first introduced by *Menard* (1995). When using *Menard*'s standardization, one must choose and report the measure of pseudo-R-squared used:

$$b^*_M = b \cdot \sigma_X \cdot \frac{R}{\sigma_{\text{logit}(\hat{Y})}}$$

Standardized coefficients don't change the model. If they are used as the main specification of the model, the variables need to be scaled accordingly, and the intercept should be transformed as the result.

For ML models on hetionet

For *hetionet*'s edge prediction problems, we use DWPCs features, whose unit is highly non-intuitive and of variable range across metapaths. Therefore, we settled on using exclusively standardized coefficients of the *Agresti* method.

Daniel Himmelstein Researcher May 2, 2016

I agree that using the *Agresti* makes the most sense. Since we're not interested in comparing our logistic models to models created using different regression techniques, I don't think the complexity of meddling with *C* is justified. Also most modern implementations will be using maximum likelihood estimatation, right **@alizee**? So of the partial methods, *Agresti* will create coefficients equivalent to the a logistic model fit on *z*-score features.

Regarding the full standardization and the choice of a pseudo R-squared: Would you advise against using the Tjur coefficient of discrimination [1]? Or would Tjur's statistic also work?

> the corresponding intercept should be computed ad hoc

What formula should be used for computing the intercept?

> - Antoine Lizee: For the intercept, it depends on the way you transform your variables to reflect the changes in the standardized coefficient. For instance, if you center your variable in addition to scaling it by the standard deviation, as your previous post suggest, you get:
> $x^* = (x - \mu_x)/\sigma_x$
> and, μ_x referring to the centering parameter (here the mean):
> $b^*_0 = b_0 + \sum b_x \cdot \mu_x$

Antoine Lizee Researcher May 2, 2016

Standardized coefficients & glmnet

In the edge prediction problem for *rephetio*, we use the R-package `glmnet` to perform lasso and ridge regression, in order to perform feature selection while fitting the model.

In the light of the note above, we wanted to adapt the *Artesi* standardization to the tools we are using.

Summary

`glmnet`, by default, standardizes the predictor variables **before** fitting the model. After checking in the source code and testing (see below) we came to the conclusion that the computed coefficients were then *reverse* standardized, with the inverse of the Artesi transformation, in order to report the coefficients in their natural metric †.

Hence, there are three way to use standardization with the `glmnet` package:

1. **Untransformed variables, specifying `standardize = FALSE`**: this corresponds to taking into account the units of the variables when fitting the regularisation. Because of the nature of regularization, this setting is usually undesirable and should be reserved to specific, well understood use-cases where keeping the variables in their natural metrics is justified. This is the only method where standardization of the coefficients after fitting of the model, as described in the note above, is appropriate.

2. **Untransformed variables, keeping the default `standardize = TRUE`**: This is the easiest option and advised for quick analysis. In order to get the standardized coefficients that actually were the result of the fitting process, apply the Agresti transformation.

3. **Standardized variables, with `standardize = FALSE`**: This last method has three key features: (i) it lets the user diagnose the regularisation in the correct units for the coefficients (using, e.g. `glmnet:::plot.glmnet()`); (ii) it lets the user deal differently with boolean or categorical variables if necessary; (iii) it makes obvious that the regularization has been done on transformed variables. The main disadvantage is that one must separately keep record of the scaling coefficients for future use of the model. If the variables are each standardized with the standard deviation (eg with `scale()`), this approach lead to the same *model* as the previous one.

Proof

We needed to understand how were the variables standardized when using the `glmnet` package, and more importantly how were the coefficients transformed back in their natural metric after fitting the model with those standardized variables.

Unfortunately for our digging purposes (i) the code is written in FORTRAN, which made us nostalgic but required some getting used to; (ii) the code does not live on a source sharing platform, but fortunately has a mirror on github, like all packages published on CRAN.

Everything we are interested in is in the fortran source code file, whose detailed comments on the top alieviated the need to reverse-enginner the variable names.

1. Definition of the `lognet` function (`routine`) starts at line 2032. We will follow the track of the standardization flag `isd`.

2. Line 2154-2159, we withness the centering and standardization of the variable array `x` by the vector of means `xms` and the vector of standard deviations `xs`, within the subfunction `lstandard1`:

```
if(ju(j).eq.0)goto 12561                                1519
xm(j)=dot_product(w,x(:,j))                             1519
x(:,j)=x(:,j)-xm(j)                                     1520
if(isd .le. 0)goto 12581                                1520
xs(j)=sqrt(dot_product(w,x(:,j)**2))                    1520
x(:,j)=x(:,j)/xs(j)                                     1520
```

3. Line 2117:

```
ca(l,ic,k)=ca(l,ic,k)/xs(ia(l))                         1499
```

4. And finally, line 2125, the intercept is computed (if the flag `intr` is 1):

```
a0(ic,k)=a0(ic,k)-dot_product(ca(1:nk,ic,k),xm(ia(1:nk)))            1501
```

Test

I wrote a quick R report to test our conclusions on the famous `diamonds` dataset.

---

† It also must be noted that `glmnet`, being written in fortran, does not make any difference between types of variables. As a result, the categorical and boolean predictor are treated as numeric vectors and standardized accordingly.