Beta Regression: Shaken, Stirred, Mixed, and Partitioned

Achim Zeileis, Francisco Cribari-Neto, Bettina Grün

http://eeecon.uibk.ac.at/~zeileis/
Overview

- Motivation
- Shaken or stirred: Single or double index beta regression for mean and/or precision in **betareg**
- Mixed: Latent class beta regression via **flexmix**
- Partitioned: Beta regression trees via **party**
- Summary
Motivation

**Goal:** Model dependent variable $y \in (0, 1)$, e.g., rates, proportions, concentrations etc.

**Common approach:** Model transformed variable $\tilde{y}$ by a linear model, e.g., $\tilde{y} = \text{logit}(y)$ or $\tilde{y} = \text{probit}(y)$ etc.

**Disadvantages:**
- Model for mean of $\tilde{y}$, not mean of $y$ (Jensen’s inequality).
- Data typically heteroskedastic.

**Idea:** Model $y$ directly using suitable parametric family of distributions plus link function.

**Specifically:** Maximum likelihood regression model using alternative parametrization of beta distribution (Ferrari & Cribari-Neto 2004).
Beta regression

**Beta distribution:** Continuous distribution for $0 < y < 1$, typically specified by two shape parameters $p, q > 0$.

**Alternatively:** Use mean $\mu = p/(p + q)$ and precision $\phi = p + q$.

**Probability density function:**

$$f(y) = \frac{\Gamma(p + q)}{\Gamma(p) \, \Gamma(q)} \, y^{p-1} \, (1 - y)^{q-1}$$

$$= \frac{\Gamma(\phi)}{\Gamma(\mu \phi) \, \Gamma((1 - \mu) \phi)} \, y^{\mu \phi - 1} \, (1 - y)^{(1-\mu)\phi - 1}$$

where $\Gamma(\cdot)$ is the gamma function.

**Properties:** Flexible shape. Mean $E(y) = \mu$ and

$$\text{Var}(y) = \frac{\mu (1 - \mu)}{1 + \phi}.$$
Beta regression

\[ \phi = 5 \]

\[ \phi = 100 \]
Beta regression

Regression model:

- Observations $i = 1, \ldots, n$ of dependent variable $y_i$.
- Link parameters $\mu_i$ and $\phi_i$ to sets of regressor $x_i$ and $z_i$.
- Use link functions $g_1$ (logit, probit, . . . ) and $g_2$ (log, identity, . . . ).

\[
  g_1(\mu_i) = x_i^\top \beta, \\
g_2(\phi_i) = z_i^\top \gamma.
\]

Inference:

- Coefficients $\beta$ and $\gamma$ are estimated by maximum likelihood.
- The usual central limit theorem holds with associated asymptotic tests (likelihood ratio, Wald, score/LM).
Implementation in R

Model fitting:
- Package `betareg` with main model fitting function `betareg()`.
- Interface and fitted models are designed to be similar to `glm()`.
- Model specification via formula plus data.
- Two part formula, e.g., \( y \sim x_1 + x_2 + x_3 | z_1 + z_2 \).
- Log-likelihood is maximized numerically via `optim()`.
- Extractors: `coef()`, `vcov()`, `residuals()`, `logLik()`, ...

Inference:
- Base methods: `summary()`, `AIC()`, `confint()`.
- Methods from `lmtest` and `car`: `lrtest()`, `waldtest()`, `coefTest()`, `linearHypothesis()`.
- Moreover: Multiple testing via `multcomp` and structural change tests via `strucchange`. 
Illustration: Reading accuracy

- 44 Australian primary school children.
- Dependent variable: Score of test for reading accuracy.
- Regressors: Indicator dyslexia (yes/no), nonverbal iq score.

Analysis:
- OLS for transformed data leads to non-significant effects.
- OLS residuals are heteroskedastic.
- Beta regression captures heteroskedasticity and shows significant effects.
Illustration: Reading accuracy

```r
R> data("ReadingSkills", package = "betareg")
R> rs_ols <- lm(qlogis(accuracy) ~ dyslexia * iq,
+     data = ReadingSkills)
R> coefTest(rs_ols)

t test of coefficients:

         Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.60107    0.22586  7.0888 1.411e-08 ***
dyslexia   -1.20563    0.22586 -5.3380 4.011e-06 ***
     iq       0.35945    0.22548  1.5941 0.11878
  dyslexia:iq -0.42286    0.22548  1.8754 0.06805 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R> bptest(rs_ols)
studentized Breusch-Pagan test

data:  rs_ols
BP = 21.692, df = 3, p-value = 7.56e-05
```
### Illustration: Reading accuracy

R> rs_beta <- betareg(accuracy ~ dyslexia * iq | dyslexia + iq, + data = ReadingSkills)
R> coefstest(rs_beta)

z test of coefficients:

| Estimate | Std. Error | z value | Pr(>|z|) |
|----------|------------|---------|----------|
| (Intercept) | 1.12323 | 0.14283 | 7.8638 | 3.725e-15 *** |
| dyslexia | -0.74165 | 0.14275 | -5.1952 | 2.045e-07 *** |
| iq | 0.48637 | 0.13315 | 3.6528 | 0.0002594 *** |
| dyslexia:iq | -0.58126 | 0.13269 | -4.3805 | 1.184e-05 *** |
| (phi)_(Intercept) | 3.30443 | 0.22274 | 14.8353 | < 2.2e-16 *** |
| (phi)_dyslexia | 1.74656 | 0.26232 | 6.6582 | 2.772e-11 *** |
| (phi)_iq | 1.22907 | 0.26720 | 4.5998 | 4.228e-06 *** |

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Illustration: Reading accuracy

![Graph showing the relationship between IQ and reading accuracy for control and dyslexic groups, with fits for betareg and lm models.](image-url)
Extensions: Partitions and mixtures

So far: Reuse standard inference methods for fitted model objects.

Now: Reuse fitting functions in more complex models.

Model-based recursive partitioning: Package party.
  • Idea: Recursively split sample with respect to available variables.
  • Aim: Maximize partitioned likelihood.
  • Fit: One model per node of the resulting tree.

Latent class regression, mixture models: Package flexmix.
  • Idea: Capture unobserved heterogeneity by finite mixtures of regressions.
  • Aim: Maximize weighted likelihood with $k$ components.
  • Fit: Weighted combination of $k$ models.
Beta regression trees

**Partitioning variables:** dyslexia and further random noise variables.

```r
R> set.seed(1071)
R> ReadingSkills$x1 <- rnorm(nrow(ReadingSkills))
R> ReadingSkills$x2 <- runif(nrow(ReadingSkills))
R> ReadingSkills$x3 <- factor(rnorm(nrow(ReadingSkills)) > 0)
```

**Fit beta regression tree:** In each node accuracy’s mean and precision depends on iq, partitioning is done by dyslexia and the noise variables x1, x2, x3.

```r
R> rs_tree <- betatree(accuracy ~ iq | iq,
+ ~ dyslexia + x1 + x2 + x3,
+ data = ReadingSkills, minsplit = 10)
R> plot(rs_tree)
```

**Result:** Only relevant regressor dyslexia is chosen for splitting.
Beta regression trees

Node 2 (n = 25)

Node 3 (n = 19)

dyslexia

p < 0.001

1

no

yes

−2.1 2.2

1

−2.1 2.2
Latent class beta regression

Setup:

- No dyslexia information available.
- Look for $k = 3$ clusters: Two different relationships of type accuracy $\sim$ iq, plus component for ideal score of 0.99.

Fit beta mixture regression:

R> rs_mix <- betamix(accuracy $\sim$ iq, data = ReadingSkills, k = 3,
+   nstart = 10, extra_components = extraComponent(
+   type = "uniform", coef = 0.99, delta = 0.01))

Result:

- Dyslexic children separated fairly well.
- Other children are captured by mixture of two components: ideal reading scores, and strong dependence on iq score.
Latent class beta regression

![Latent class beta regression](image)
Latent class beta regression
Latent class beta regression
Latent class beta regression
Computational infrastructure

Model-based recursive partitioning:

- **party** provides the recursive partitioning.
- **betareg** provides the models in each node.
  - Model-fitting function: `betareg.fit()` (conveniently without formula processing).
  - Extractor for empirical estimating functions (aka scores or case-wise gradient contributions): `estfun()` method.
  - Some additional (and somewhat technical) S4 glue...

Latent class regression, mixture models:

- **flexmix** provides the E-step for the EM algorithm.
- **betareg** provides the M-step.
  - Model-fitting function: `betareg.fit()`.
  - Extractor for case-wise log-likelihood contributions: `dbeta()`.
  - Some additional (and somewhat more technical) S4 glue...
Summary

Beta regression and extensions:

- Flexible regression model for proportions, rates, concentrations.
- Can capture skewness and heteroskedasticity.
- R implementation `betareg`, similar to `glm()`.
- Due to design, standard inference methods can be reused easily.
- Fitting functions can be plugged into more complex fitters.
- Convenience interfaces available for: Model-based partitioning, finite mixture models.
References


