Generalized Linear Models

Generalized linear models currently supports estimation using the one-parameter exponential families.

See Module Reference [#module-reference] for commands and arguments.

Examples

```
# Load modules and data
In [1]: import statsmodels.api as sm

In [2]: data = sm.datasets.scotland.load(as_pandas=False)

In [3]: data.exog = sm.add_constant(data.exog)

# Instantiate a gamma family model with the default link function.
In [4]: gamma_model = sm.GLM(data.endog, data.exog, family=sm.families.Gamma())

In [5]: gamma_results = gamma_model.fit()

In [6]: print(gamma_results.summary())
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                      y   No. Observations:                   32
Model:                            GLM   Df Residuals:                       24
Model Family:                   Gamma   Df Model:                            7
Link Function:          inverse_power   Scale:                       0.0035843
Method:                          IRLS   Log-Likelihood:                -83.017
Date:                Tue, 02 Feb 2021   Deviance:                      0.087389
Time:                        07:07:06   Pearson chi2:                    0.0860
No. Iterations:                     6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.0178      0.011     -1.548      0.122      -0.040       0.005
x1          4.962e-05   1.62e-05      3.060      0.002    1.78e-05    8.14e-05
x2             0.0020      0.001      3.824      0.000       0.001       0.003
x3         -7.181e-05   2.71e-05     -2.648      0.008      -0.000   -1.87e-05
x4             0.0001   4.06e-05      2.757      0.006    3.23e-05       0.000
x5         -1.468e-07   1.24e-07     -1.187      0.235   -3.89e-07    9.56e-08
x6             -0.0005      0.000     -2.159      0.031      -0.001   -4.78e-05
x7         -2.427e-06   7.46e-07     -3.253      0.001   -3.89e-06   -9.65e-07
==============================================================================
```

Detailed examples can be found here:

- GLM [examples/notebooks/generated/glm.html]
- Formula [examples/notebooks/generated/glm_formula.html]

Technical Documentation

The statistical model for each observation i is assumed to be

$$Y_i \sim F_{EDM}(\cdot|\theta, \phi, w_i) \text{ and } \mu_i = E[Y_i|x_i] = g^{-1}(x_i'\beta).$$

where g is the link function and $F_{EDM}(\cdot|\theta, \phi, w)$ is a distribution of the family of exponential dispersion models (EDM) with natural parameter θ, scale parameter ϕ and weight w. Its density is given by

$$f_{EDM}(y|\theta,\phi,w) = c(y,\phi,w)\exp\left(\frac{y\theta-b(\theta)}{\phi}w\right).$$

It follows that $\mu = b'(\theta)$ and $Var[Y|x] = \frac{\phi}{w}b''(\theta)$. The inverse of the first equation gives the natural parameter as a function of the expected value $\theta(\mu)$ such that

$$Var[Y_i|x_i] = \frac{\phi}{w_i}v(\mu_i)$$

with $v(\mu) = b''(\theta(\mu))$. Therefore it is said that a GLM is determined by link function g and variance function $v(\mu)$ alone (and x of course).

Note that while ϕ is the same for every observation y_i and therefore does not influence the estimation of β, the weights w_i might be different for every y_i such that the estimation of β depends on them.

Distribution	Domain	$\mu = E[Y\|x]$	$v(\mu)$	$\theta(\mu)$	$b(\theta)$	ϕ
Binomial $B(n,p)$	$0,1,\ldots,n$	np	$\mu - \frac{\mu^2}{n}$	$\log\frac{p}{1-p}$	$n\log(1+e^\theta)$	1
Poisson $P(\mu)$	$0,1,\ldots,\infty$	μ	μ	$\log(\mu)$	e^θ	1
Neg. Binom. $NB(\mu,\alpha)$	$0,1,\ldots,\infty$	μ	$\mu+\alpha\mu^2$	$\log(\frac{\alpha\mu}{1+\alpha\mu})$	$-\frac{1}{\alpha}\log(1-\alpha e^\theta)$	1
Gaussian/Normal $N(\mu,\sigma^2)$	$(-\infty,\infty)$	μ	1	μ	$\frac{1}{2}\theta^2$	σ^2
Gamma $N(\mu,\nu)$	$(0,\infty)$	μ	μ^2	$-\frac{1}{\mu}$	$-\log(-\theta)$	$\frac{1}{\nu}$
Inv. Gauss. $IG(\mu,\sigma^2)$	$(0,\infty)$	μ	μ^3	$-\frac{1}{2\mu^2}$	$-\sqrt{-2\theta}$	σ^2
Tweedie $p \geq 1$	depends on p	μ	μ^p	$\frac{\mu^{1-p}}{1-p}$	$\frac{\alpha-1}{\alpha}\left(\frac{\theta}{\alpha-1}\right)^\alpha$	ϕ

The Tweedie distribution has special cases for $p = 0, 1, 2$ not listed in the table and uses $\alpha = \frac{p-2}{p-1}$.

Correspondence of mathematical variables to code:

- Y and y are coded as `endog`, the variable one wants to model
- x is coded as `exog`, the covariates alias explanatory variables
- β is coded as `params`, the parameters one wants to estimate
- μ is coded as `mu`, the expectation (conditional on x) of Y
- g is coded as `link` argument to the `class Family`
- ϕ is coded as `scale`, the dispersion parameter of the EDM
- w is not yet supported (i.e. $w = 1$), in the future it might be `var_weights`
- p is coded as `var_power` for the power of the variance function $v(\mu)$ of the Tweedie distribution, see table
- α is either
  - Negative Binomial: the ancillary parameter `alpha`, see table
  - Tweedie: an abbreviation for $\frac{p-2}{p-1}$ of the power p of the variance function, see table

References

- Gill, Jeff. 2000. Generalized Linear Models: A Unified Approach. SAGE QASS Series.
- Green, PJ. 1984. “Iteratively reweighted least squares for maximum likelihood estimation, and some robust and resistant alternatives.” Journal of the Royal Statistical Society, Series B, 46, 149-192.

- Hardin, J.W. and Hilbe, J.M. 2007. “Generalized Linear Models and Extensions.” 2nd ed. Stata Press, College Station, TX.
- McCullagh, P. and Nelder, J.A. 1989. “Generalized Linear Models.” 2nd ed. Chapman & Hall, Boca Rotan.

Module Reference

Model Class

`GLM`
[generated/statsmodels.genmod.generalized_linear_model.GLM.html#statsmodels.genmod.generalized_linear_model.GLM](e

---

Results Class

`GLMResults` [generated/statsmodels.genmod.generalized_linear_model.GLMResults.html#statsmodels.genmod.generalized_li

---

`PredictionResults`
[generated/statsmodels.genmod.generalized_linear_model.PredictionResults.html#statsmodels.genmod.generalized_linear_n

---

Families

The distribution families currently implemented are

`Family` [generated/statsmodels.genmod.families.family.Family.html#statsmodels.genmod.families.family.Family](link, varianc

---

`Binomial` [generated/statsmodels.genmod.families.family.Binomial.html#statsmodels.genmod.families.family.Binomial]([link]

`Gamma` [generated/statsmodels.genmod.families.family.Gamma.html#statsmodels.genmod.families.family.Gamma]([link])

`Gaussian` [generated/statsmodels.genmod.families.family.Gaussian.html#statsmodels.genmod.families.family.Gaussian]([link

---

`InverseGaussian`
[generated/statsmodels.genmod.families.family.InverseGaussian.html#statsmodels.genmod.families.family.InverseGaussian

---

`NegativeBinomial`
[generated/statsmodels.genmod.families.family.NegativeBinomial.html#statsmodels.genmod.families.family.NegativeBinomi

---

`Poisson` [generated/statsmodels.genmod.families.family.Poisson.html#statsmodels.genmod.families.family.Poisson]([link])

`Tweedie` [generated/statsmodels.genmod.families.family.Tweedie.html#statsmodels.genmod.families.family.Tweedie]([link, va

Link Functions

The link functions currently implemented are the following. Not all link functions are available for each distribution family. The list of available link functions can be obtained by

```
>>> sm.families.family.<familyname>.links
```

`Link` [generated/statsmodels.genmod.families.links.Link.html#statsmodels.genmod.families.links.Link]()

`CDFLink` [generated/statsmodels.genmod.families.links.CDFLink.html#statsmodels.genmod.families.links.CDFLink]([dbn])

`CLogLog` [generated/statsmodels.genmod.families.links.CLogLog.html#statsmodels.genmod.families.links.CLogLog]()

`Log` [generated/statsmodels.genmod.families.links.Log.html#statsmodels.genmod.families.links.Log]()

`Logit` [generated/statsmodels.genmod.families.links.Logit.html#statsmodels.genmod.families.links.Logit]()

`NegativeBinomial`
[generated/statsmodels.genmod.families.links.NegativeBinomial.html#statsmodels.genmod.families.links.NegativeBinomial]

`Power` [generated/statsmodels.genmod.families.links.Power.html#statsmodels.genmod.families.links.Power]([power])

`cauchy` [generated/statsmodels.genmod.families.links.cauchy.html#statsmodels.genmod.families.links.cauchy]()

`cloglog` [generated/statsmodels.genmod.families.links.cloglog.html#statsmodels.genmod.families.links.cloglog]()

`identity` [generated/statsmodels.genmod.families.links.identity.html#statsmodels.genmod.families.links.identity]()

`inverse_power`
[generated/statsmodels.genmod.families.links.inverse_power.html#statsmodels.genmod.families.links.inverse_power]()

`inverse_squared`
[generated/statsmodels.genmod.families.links.inverse_squared.html#statsmodels.genmod.families.links.inverse_squared]()

`log` [generated/statsmodels.genmod.families.links.log.html#statsmodels.genmod.families.links.log]()

`logit` [generated/statsmodels.genmod.families.links.logit.html#statsmodels.genmod.families.links.logit]()

`nbinom` [generated/statsmodels.genmod.families.links.nbinom.html#statsmodels.genmod.families.links.nbinom]([alpha])

`probit` [generated/statsmodels.genmod.families.links.probit.html#statsmodels.genmod.families.links.probit]([dbn])

Variance Functions

Each of the families has an associated variance function. You can access the variance functions here:

```
>>> sm.families.<familyname>.variance
```

`VarianceFunction` [generated/statsmodels.genmod.families.varfuncs.VarianceFunction.html#statsmodels.genmod.families.varfuncs.VarianceFu

`constant` [generated/statsmodels.genmod.families.varfuncs.constant.html#statsmodels.genmod.families.varfuncs.constant]

`Power` [generated/statsmodels.genmod.families.varfuncs.Power.html#statsmodels.genmod.families.varfuncs.Power]([power]

`mu` [generated/statsmodels.genmod.families.varfuncs.mu.html#statsmodels.genmod.families.varfuncs.mu]

`mu_squared` [generated/statsmodels.genmod.families.varfuncs.mu_squared.html#statsmodels.genmod.families.varfuncs.mu_

`mu_cubed` [generated/statsmodels.genmod.families.varfuncs.mu_cubed.html#statsmodels.genmod.families.varfuncs.mu_cub

`Binomial` [generated/statsmodels.genmod.families.varfuncs.Binomial.html#statsmodels.genmod.families.varfuncs.Binomial]

`binary` [generated/statsmodels.genmod.families.varfuncs.binary.html#statsmodels.genmod.families.varfuncs.binary]

`NegativeBinomial` [generated/statsmodels.genmod.families.varfuncs.NegativeBinomial.html#statsmodels.genmod.families.varfuncs.NegativeB

`nbinom` [generated/statsmodels.genmod.families.varfuncs.nbinom.html#statsmodels.genmod.families.varfuncs.nbinom]