

Introduction to BTYD (Buy Until You Die) Models

Berlin Bayesians MeetUp - September 2022

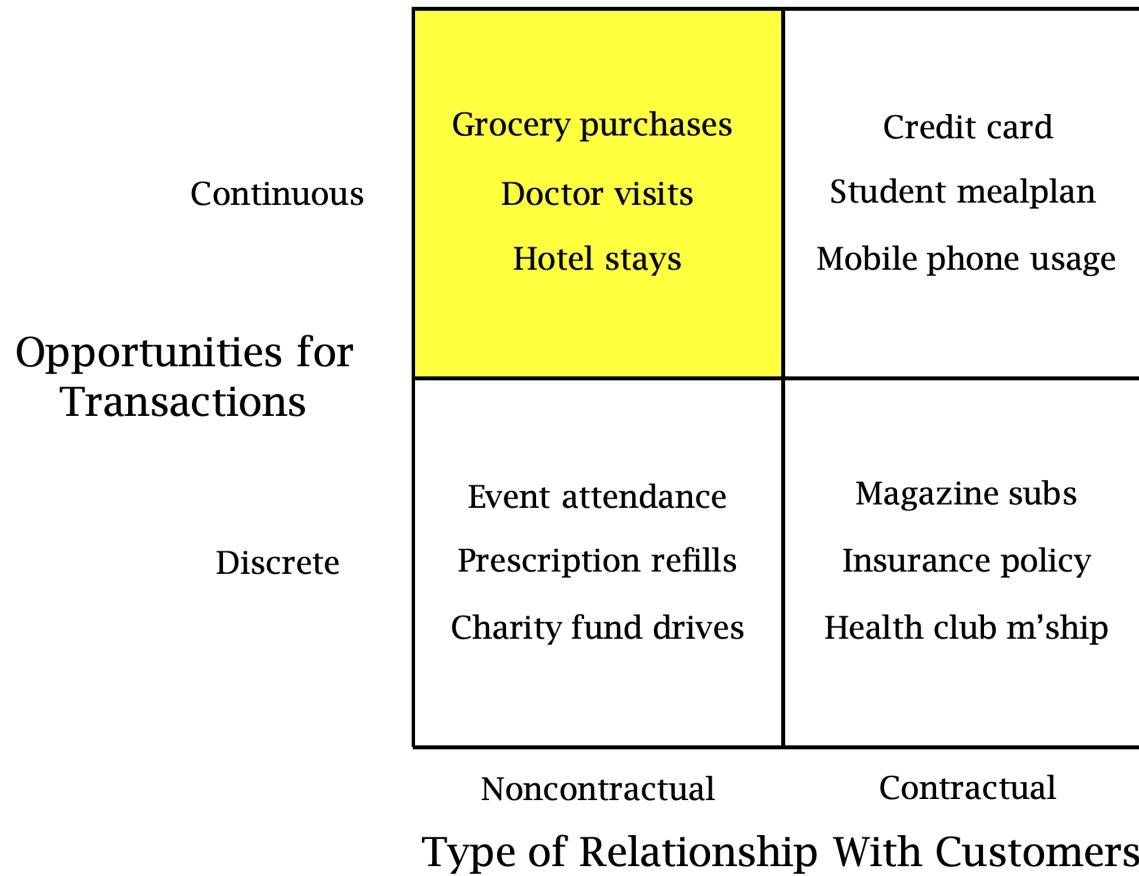
Dr. Juan Orduz

Outline

1. Introduction to BTYD Models
2. BG/NBD Model: Model Specification
3. BG/NBD Model: Maximum Likelihood Estimation
(lifetimes)
4. BG/NBD Model: Bayesian Estimation (**pymc**)
 - External Regressors
 - Hierarchical Model
5. References and Resources

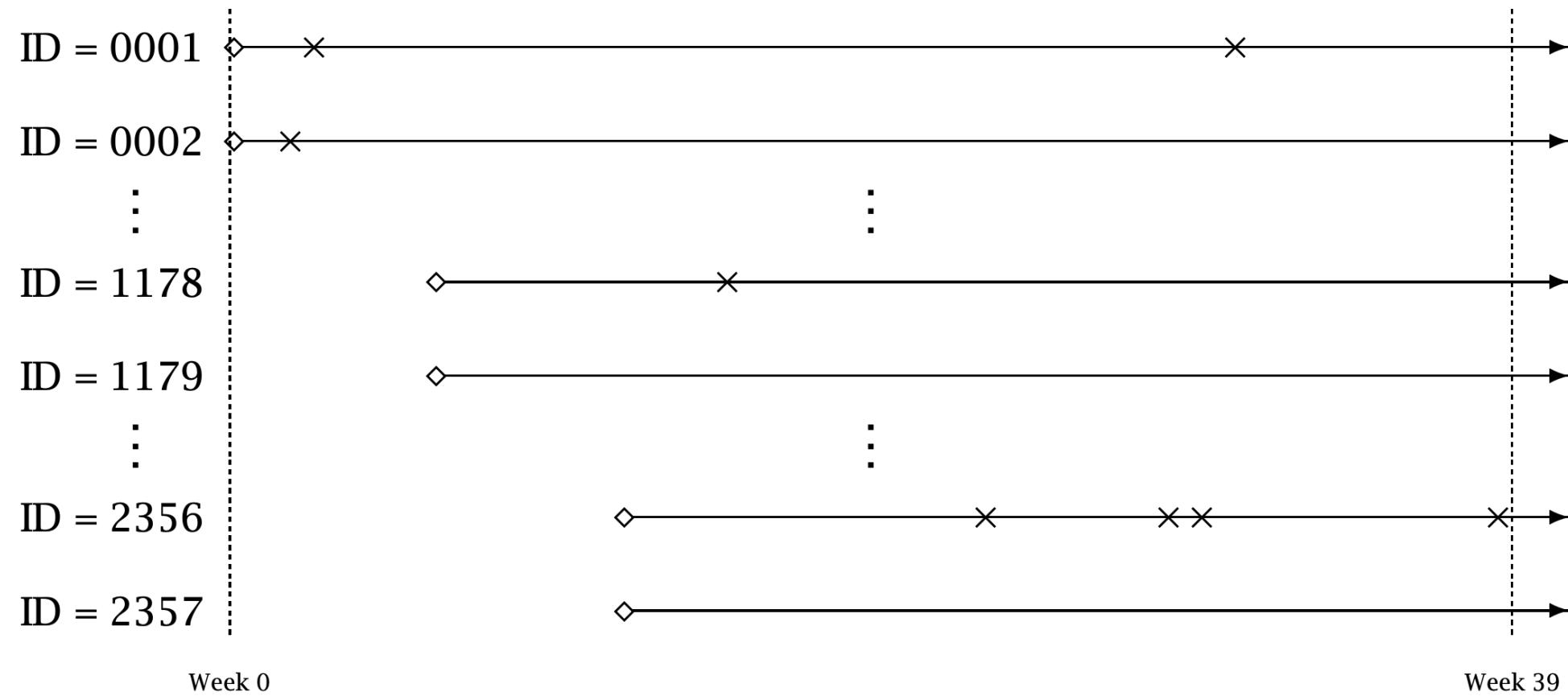
Classifying Customer Bases

Classifying Customer Bases



Purchase Histories

Purchase Histories



The Pareto/NBD Model

The Pareto/NBD Model (Schmittlein, Morrison and Colombo 1987)

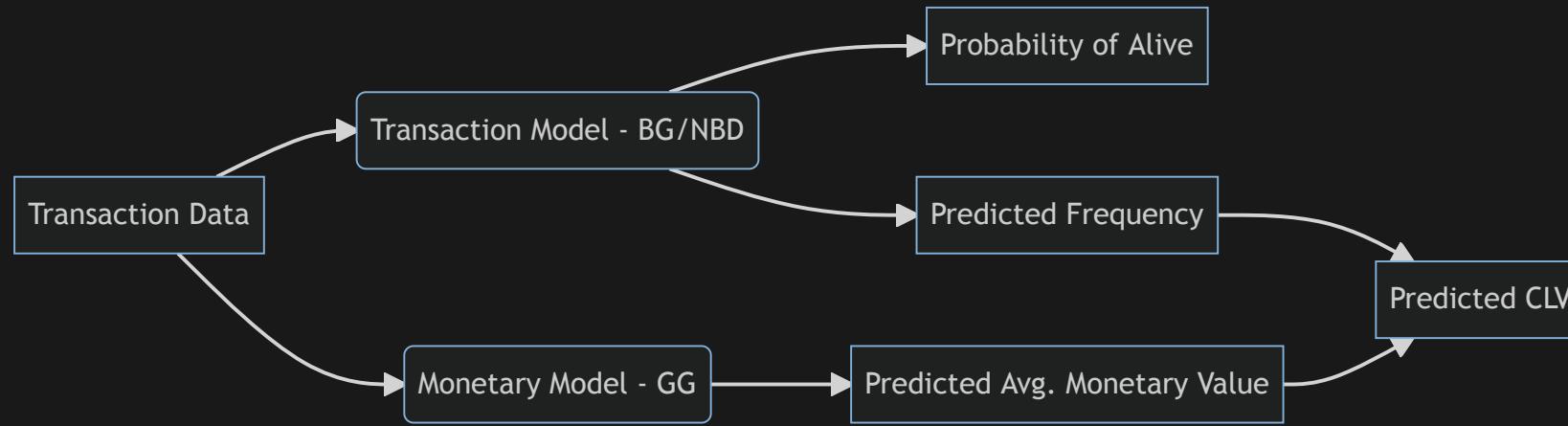
Transaction Process:

- While active, # transactions made by a customer follows a Poisson process with transaction rate λ .
- Heterogeneity in transaction rates across customers is distributed $\text{gamma}(r, \alpha)$.

Dropout Process:

- Each customer has an unobserved “lifetime” of length ω , which is distributed exponential with dropout rate μ .
- Heterogeneity in dropout rates across customers is distributed $\text{gamma}(s, \beta)$.

Models Workflow



- **Transaction Model:** Number of transactions per customer.
- **Monetary Model:** Average monetary value of a transaction.

CLV Formula

Discounted cash flow (DCF) method:

$$CLV = \sum_{i=1}^{\infty} \frac{M_i}{(1 + \delta)^t}$$

- M_i : cash flow in period .
- δ : discount rate.

```
1 for i in steps:  
2     df["clv"] += (  
3         monetary_value * expected_number_of_transactions)  
4         / (1 + discount_rate) ** i  
5     )
```

The BG/NBD Transaction Model

The BG/NBD Model (Fader, Hardie and Lee 2005c)

Purchase Process:

- While active, # transactions made by a customer follows a Poisson process with transaction rate λ .
- Heterogeneity in transaction rates across customers is distributed $\text{gamma}(r, \alpha)$.

Dropout Process:

- After any transaction, a customer becomes inactive with probability p .
- Heterogeneity in dropout probabilities across customers is distributed $\text{beta}(a, b)$.

Purchase Metrics

- **frequency**: Number of repeat purchases the customer has made. More precisely, It's the count of time periods the customer had a purchase in.
- **T**: Age of the customer in whatever time units chosen. This is equal to the duration between a customer's first purchase and the end of the period under study.
- **recency**: Age of the customer when they made their most recent purchases. This is equal to the duration between a customer's first purchase and their latest purchase.

BG/NBD Assumptions (Frequency)

1. While active, the time between transactions is distributed exponential with transaction rate, i.e.,

$$f(t_j | t_{j-1}; \lambda) = \lambda \exp(-\lambda(t_j - t_{j-1}))$$

2. Heterogeneity in follows a gamma distribution with pdf

$$\lambda$$

$$f(\lambda | r, \alpha) = \frac{\alpha^r \lambda^{r-1} \exp(-\lambda\alpha)}{\Gamma(r)},$$

BG/NBD Assumptions (Dropout)

3. After any transaction, a customer becomes inactive with probability .
4. Heterogeneity in p follows a beta distribution with pdf

$$f(p|a, b) = \frac{\Gamma(a + b)}{\Gamma(a)\Gamma(b)} p^{a-1} (1 - p)^{b-1}$$

5. The transaction rate and the dropout probability vary independently across customers.

Likelihood: Easy to Compute!

$$L(a, b, \alpha, r | X = x, t_x, T) = A_1 A_2$$

where

$$A_1 = \frac{\Gamma(r + x)\alpha^r}{\Gamma(x)} \quad A_2 = \frac{\Gamma(a)}{\Gamma(b)}$$

Strategy: Write this in `numpy` as ~~A₃~~ pass it through $A_3 = \left(\frac{1}{\alpha + T} \right)^{r+x}$
`scipy.optimize.minimize` $A_4 = \left(\frac{b}{b - x} \right)^a$

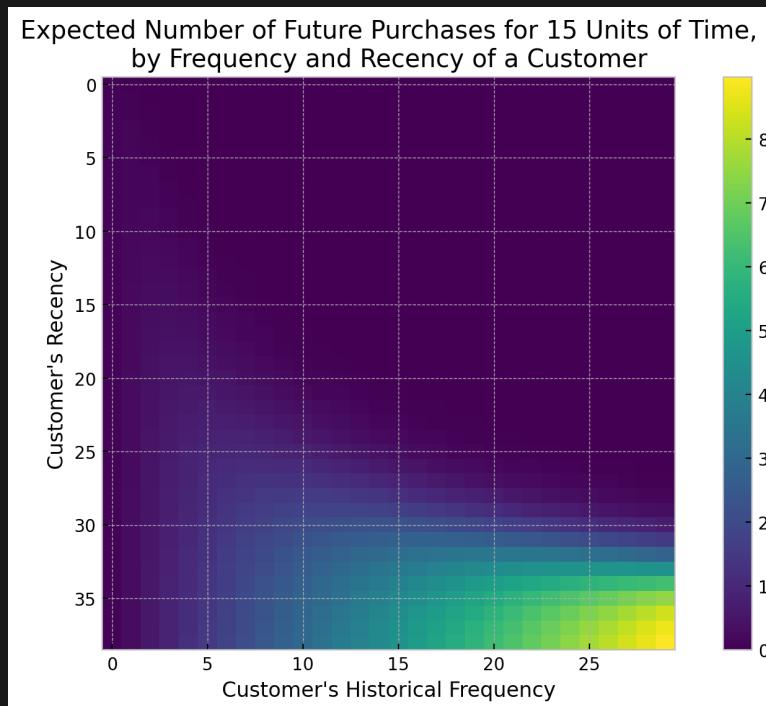
Inference: lifetimes Package

The four BG/NBD model parameters can be estimated via the method of **maximum likelihood**.

```
1 import numpy as np
2 import pandas as pd
3 from lifetimes.datasets import load_cdnow_summary
4 from lifetimes import BetaGeoFitter
5
6 data_df: pd.DataFrame = load_cdnow_summary(index_col=[0])
7
8 n = data_df.shape[0]
9 x = data_df["frequency"].to_numpy()
10 t_x = data_df["recency"].to_numpy()
11 T = data_df["T"].to_numpy()
12
13 bgf = BetaGeoFitter()
14 bgf.fit(frequency=x, recency=t_x, T=T)
```

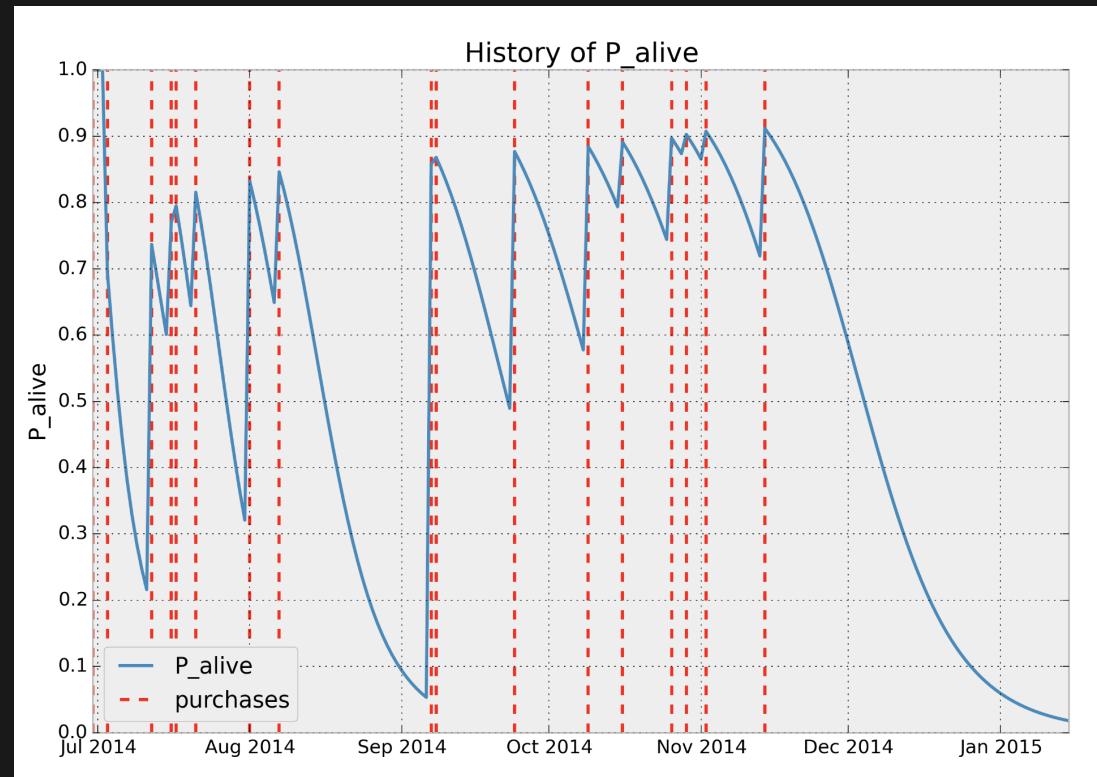
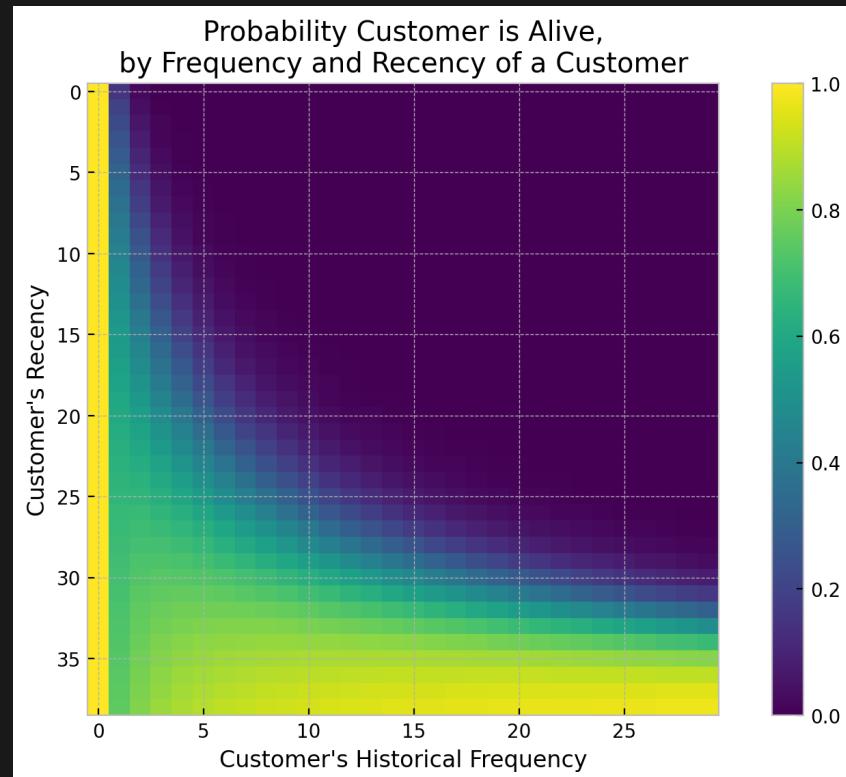
Predicted Future Purchases

```
1 from lifetimes.plotting import plot_frequency_recency_matrix  
2  
3 ax = plot_frequency_recency_matrix(model=bgf, T=15)
```

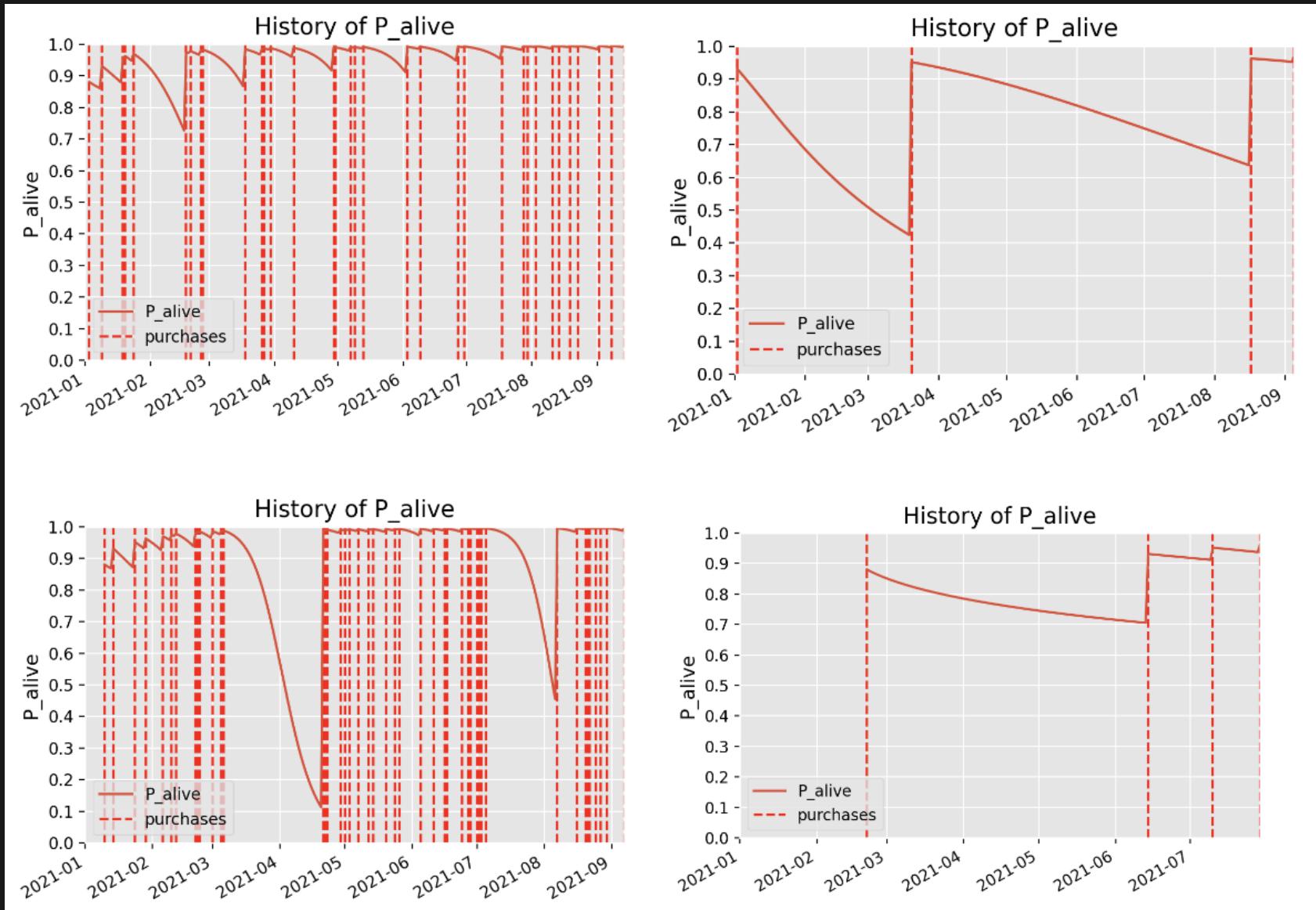


Probability of Alive

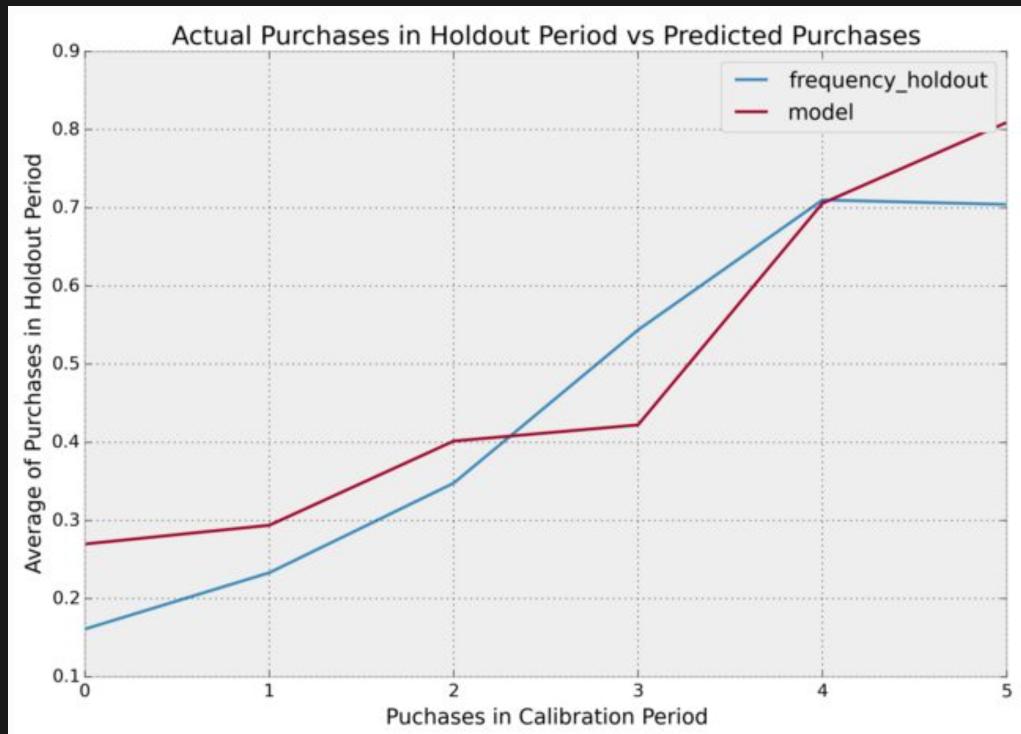
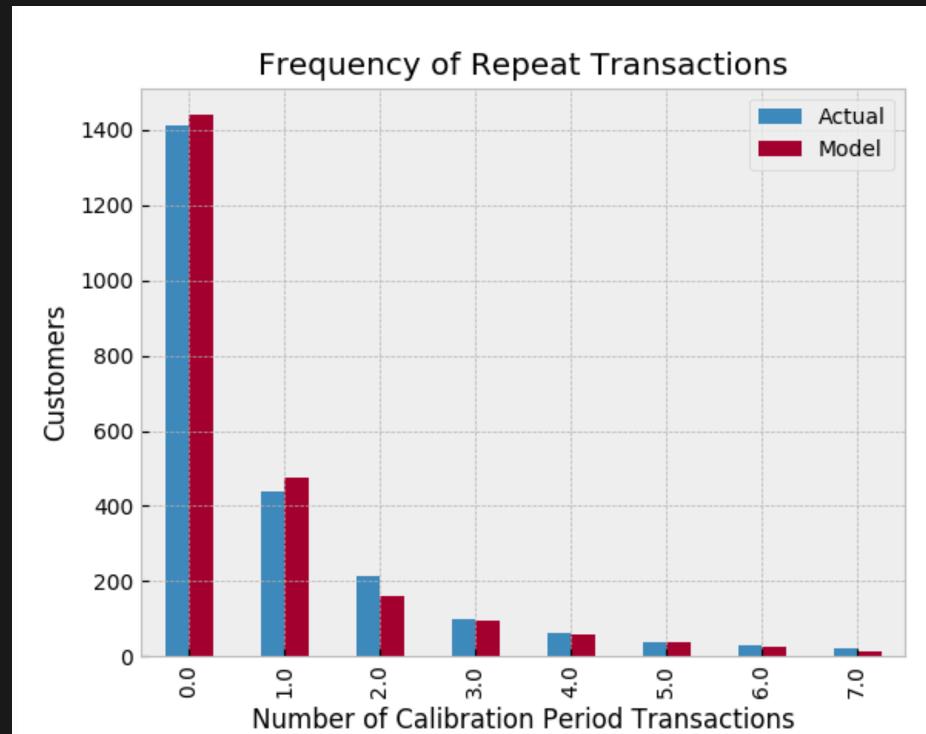
```
1 from lifetimes.plotting import plot_probability_alive_matrix  
2  
3 ax = plot_probability_alive_matrix(model=bgf)
```



Probability of Alive



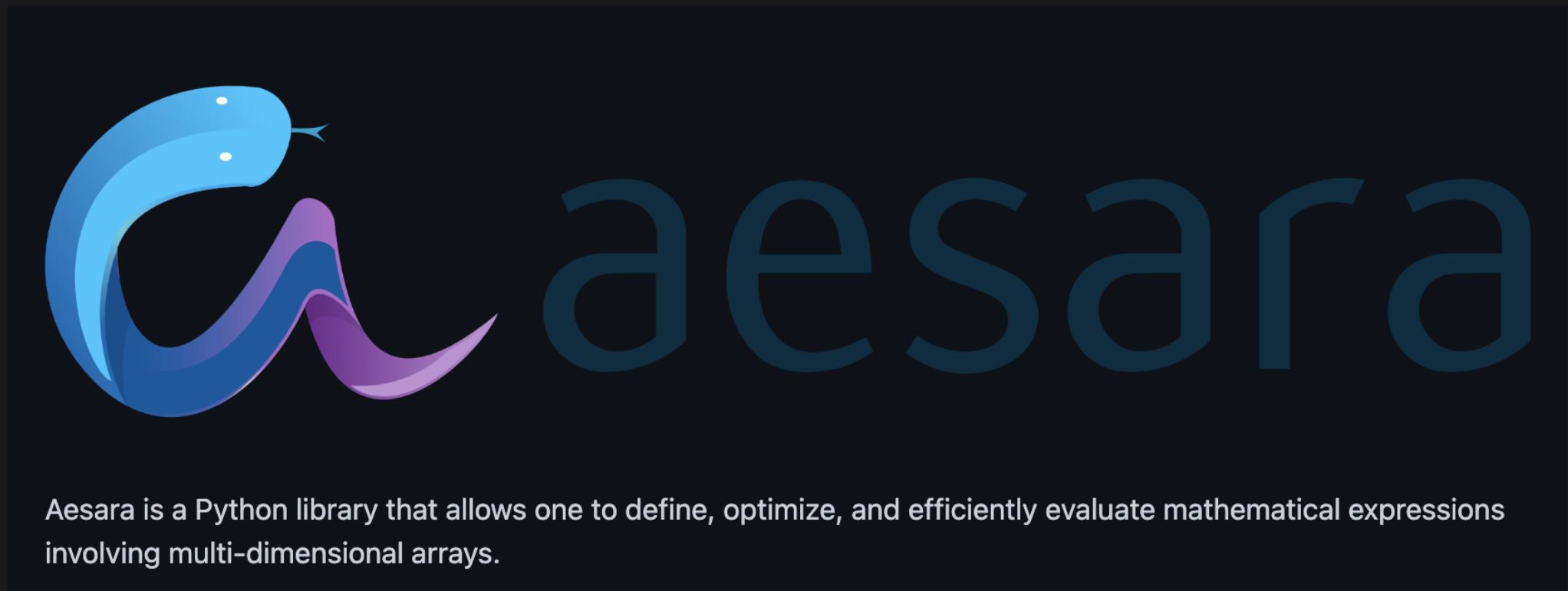
Model Evaluation



```
1 summary["model_predictions"] = model.conditional_expected_number_of_purchas  
2 duration_holdout, summary["frequency_cal"], summary["recency_cal"], sum  
3 )  
4  
5 summary.groupby("frequency_cal")[[ "frequency_holdout", "model_predictions"]]
```

From Numpy to Aesara

We can re-write the **numpy** likelihood implementation in **aesara**.



Aesara is a Python library that allows one to define, optimize, and efficiently evaluate mathematical expressions involving multi-dimensional arrays.

Log-Likelihood in Aesara

The translation bewtween `numpy` and `aesara` is very easy
(replace `np` by `at`)!

```
1 import aesara.tensor as at
2
3
4 def logp(x, t_x, T, x_zero):
5     a1 = at.gammaln(r + x) - at.gammaln(r) + r * at.log(alpha)
6     a2 = (
7         at.gammaln(a + b)
8         + at.gammaln(b + x)
9         - at.gammaln(b)
10        - at.gammaln(a + b + x)
11    )
12    a3 = -(r + x) * at.log(alpha + T)
13    a4 = (
14        at.log(a) - at.log(b + at.maximum(x, 1) - 1) - (r + x) * at.log(t_x)
15    )
16    max_a3_a4 = at.maximum(a3, a4)
17    ll_1 = a1 + a2
18    ll_2 = (
```

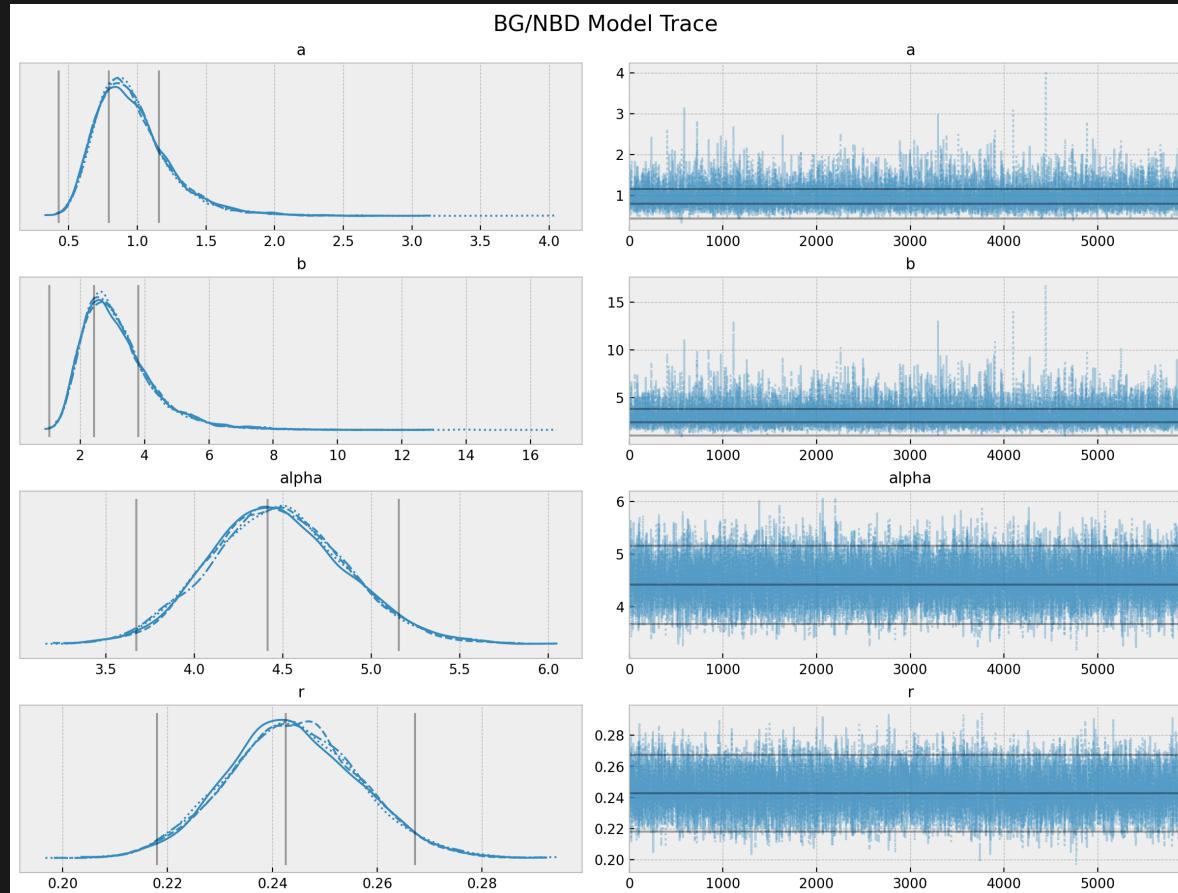
BG/NBD PyMC Model

Model Structure

```
1 import pymc as pm
2 import pymc.sampling_jax
3
4
5 with pm.Model() as model:
6
7     a = pm.HalfNormal(name="a", sigma=10)
8     b = pm.HalfNormal(name="b", sigma=10)
9
10    alpha = pm.HalfNormal(name="alpha", sigma=10)
11    r = pm.HalfNormal(name="r", sigma=10)
12
13    def logp(x, t_x, T, x_zero):
14        ...
15
16    likelihood = pm.Potential(
17        name="likelihood",
18        var=logp(x=x, t_x=t_x, T=T, x_zero=x_zero),
19    )
```

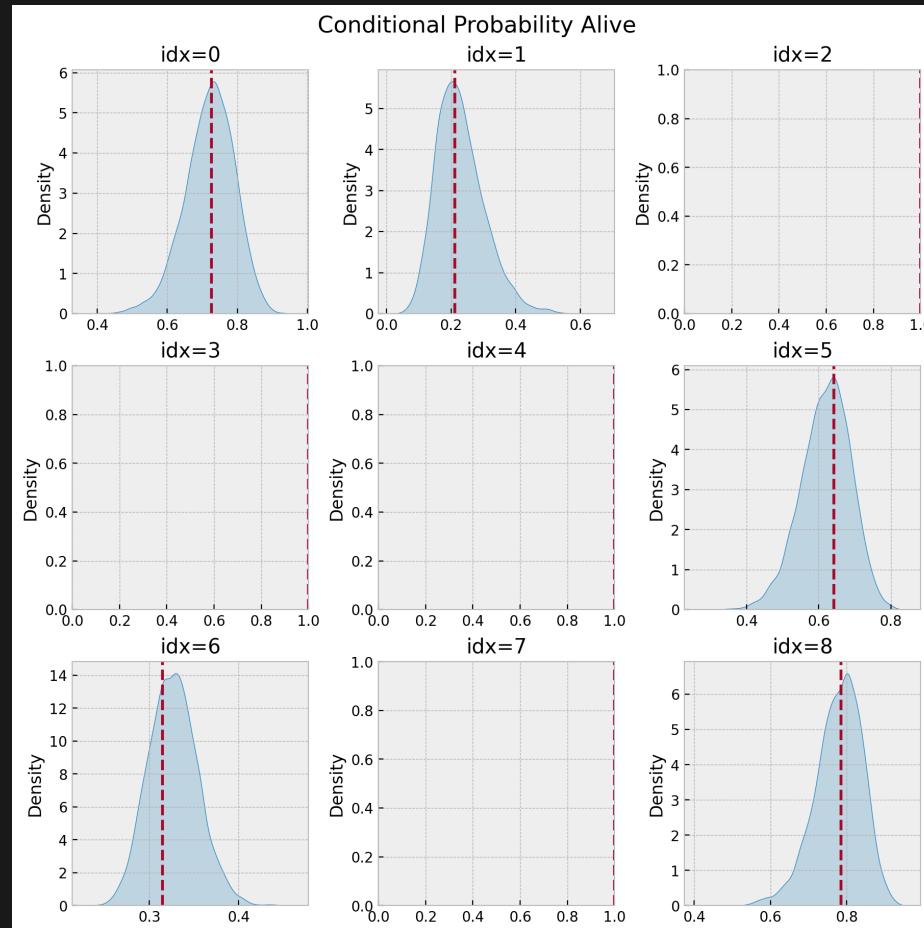
Model Sampling

```
1 with model:  
2     trace = pm.sampling_jax.sample_numpyro_nuts(  
3         tune=3000, draws=6000, chains=4, target_accept=0.95  
4     )
```



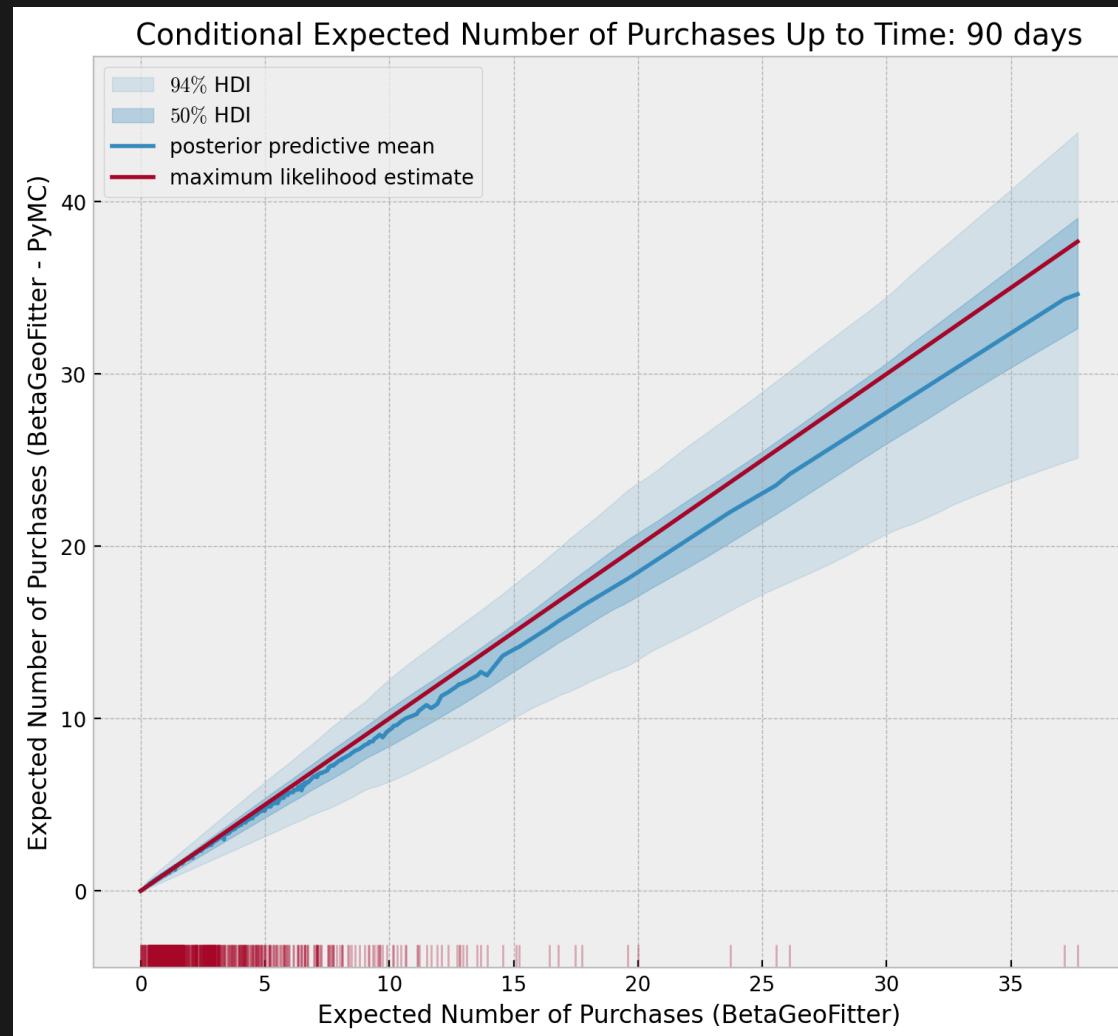
Probability of Active

How? Simply use the posterior samples (`xarray`) and broadcast the `numpy` expressions.



Future Purchases

Similarty, using the analytical expressions:



Time-Independent Covariates?

- Allow covariates α and β to explain the cross-sectional heterogeneity in the purchasing process and cross-sectional heterogeneity in the dropout process respectively.
- The likelihood and quantities of interests computed by computing expectations, remain almost the same. One only has to replace:

$$\alpha \longmapsto \alpha_0 \exp(-\gamma_1^T z_1)$$

$$a \longmapsto a_0 \exp(\gamma_2^T z_2)$$

$$b \longmapsto b_0 \exp(\gamma_3^T z_2)$$

Simulated Example

```
1 np.random.seed(42)
2
3 # construct covariate
4 mu = 0.4
5 rho = 0.7
6 z = np.random.binomial(n=1, p=mu, size=x.size)
7
8 # change frequency values by reducing it the values where z !=0
9 x_z = np.floor(x * (1 - (rho * z)))
10
11 # make sure the recency is zero whenever the frequency is zero
12 t_x_z = t_x.copy()
13 t_x_z[np.argwhere(x_z == 0).flatten()] = 0
14
15 # sanity checks
16 assert x_z.min() == 0
17 assert np.all(t_x_z[np.argwhere(x_z == 0).flatten()] == 0)
```

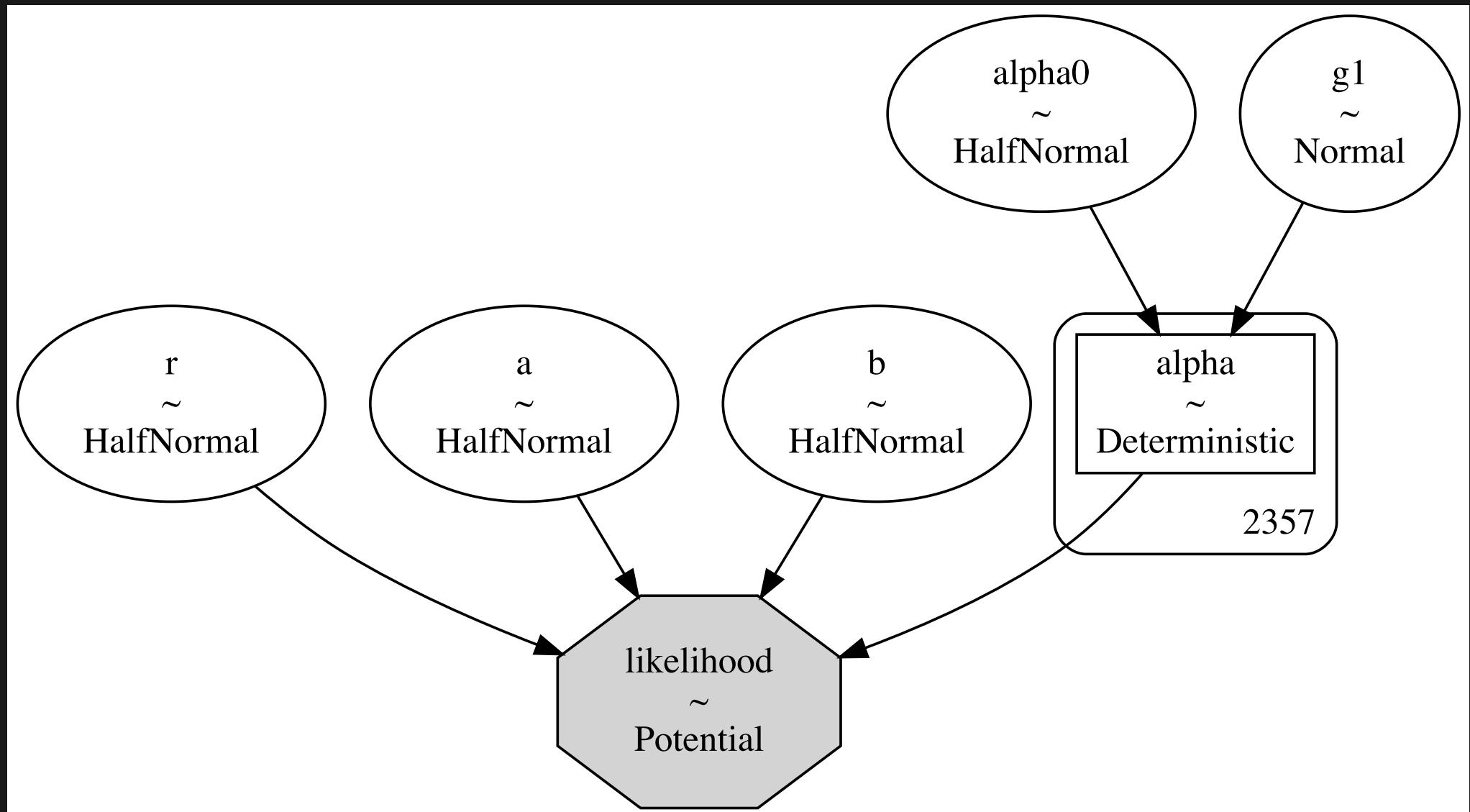
the frequency is reduced by .

$z = 1 \Rightarrow 70\%$

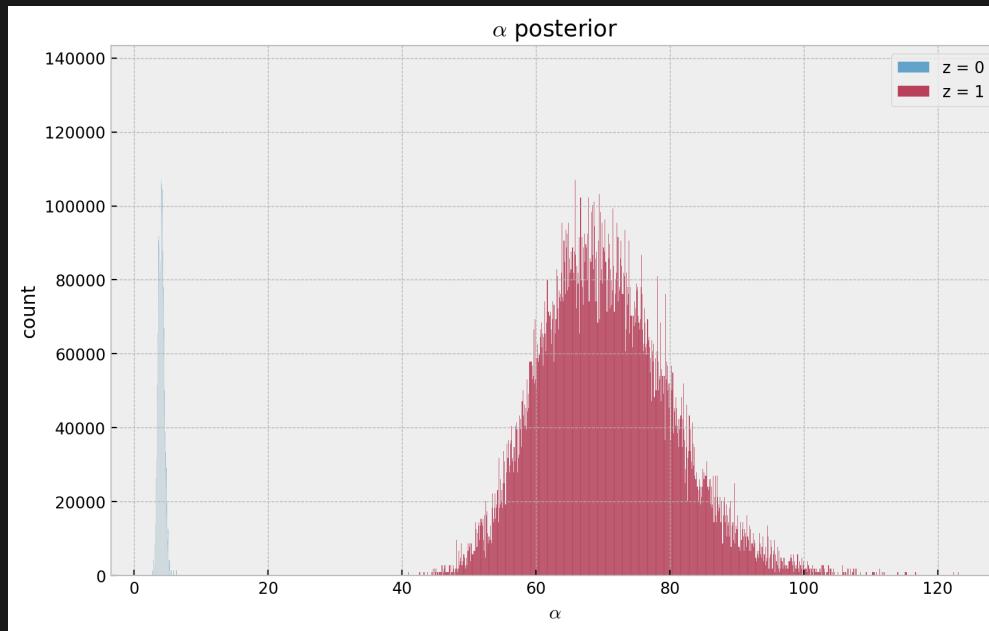
PyMC Model

```
1 with pm.Model() as model_cov:
2
3     a = pm.HalfNormal(name="a", sigma=10)
4     b = pm.HalfNormal(name="b", sigma=10)
5
6     alpha0 = pm.HalfNormal(name="alpha0", sigma=10)
7     g1 = pm.Normal(name="g1", mu=0, sigma=10)
8     alpha = pm.Deterministic(name="alpha", var=alpha0 * at.exp(- g1 * z))
9
10    r = pm.HalfNormal(name="r", sigma=10)
11
12    def logp(x, t_x, T, x_zero):
13        ...
14
15    likelihood = pm.Potential(
16        name="likelihood",
17        var=logp(x=x_z, t_x=t_x_z, T=T, x_zero=x_zero_z),
18    )
```

Model Parameters



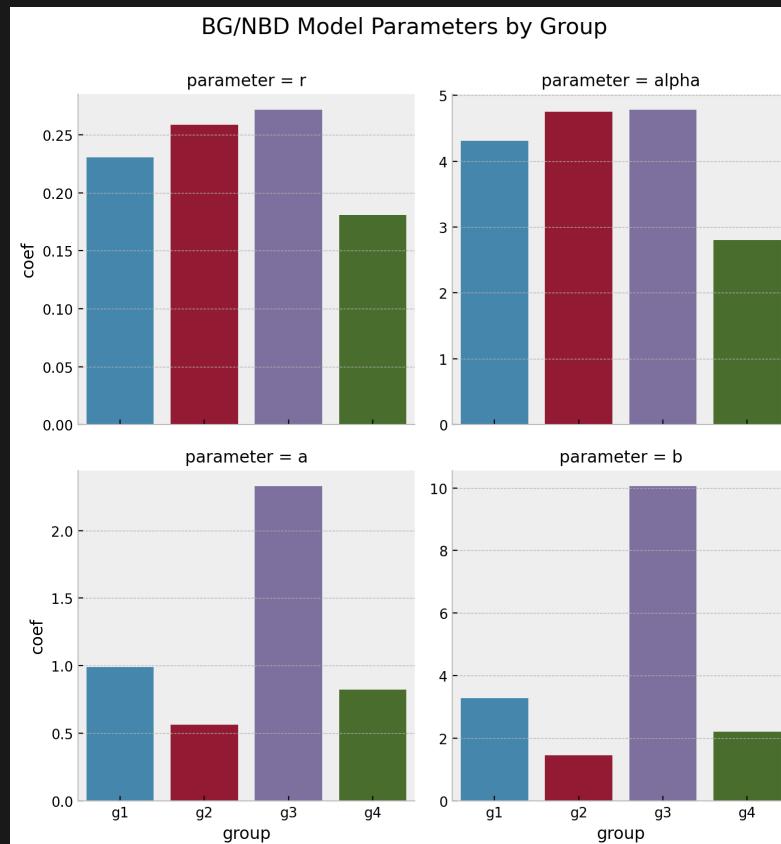
Effect in Parameters



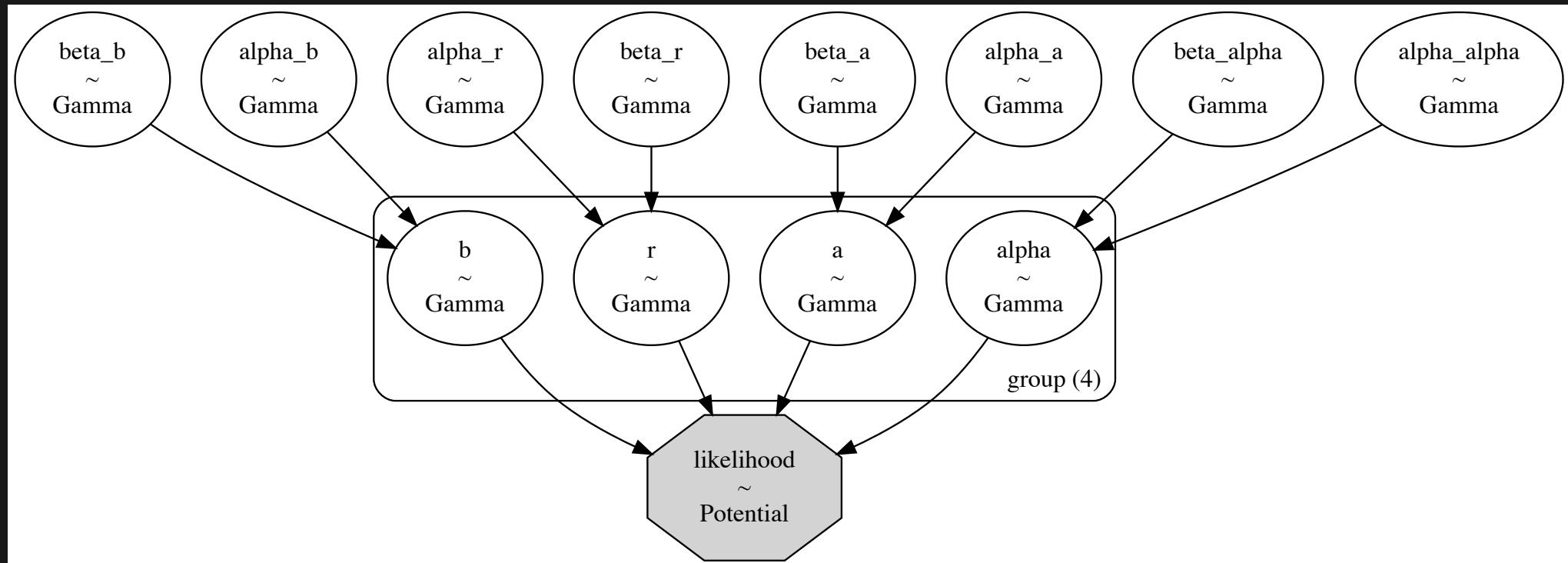
- Recall , which has expected value .
- Hence, as , then which is consistent with the data generation process (where implies a lower frequency).
 $z = 1$

Hierarchical BG/NBD Model - Data

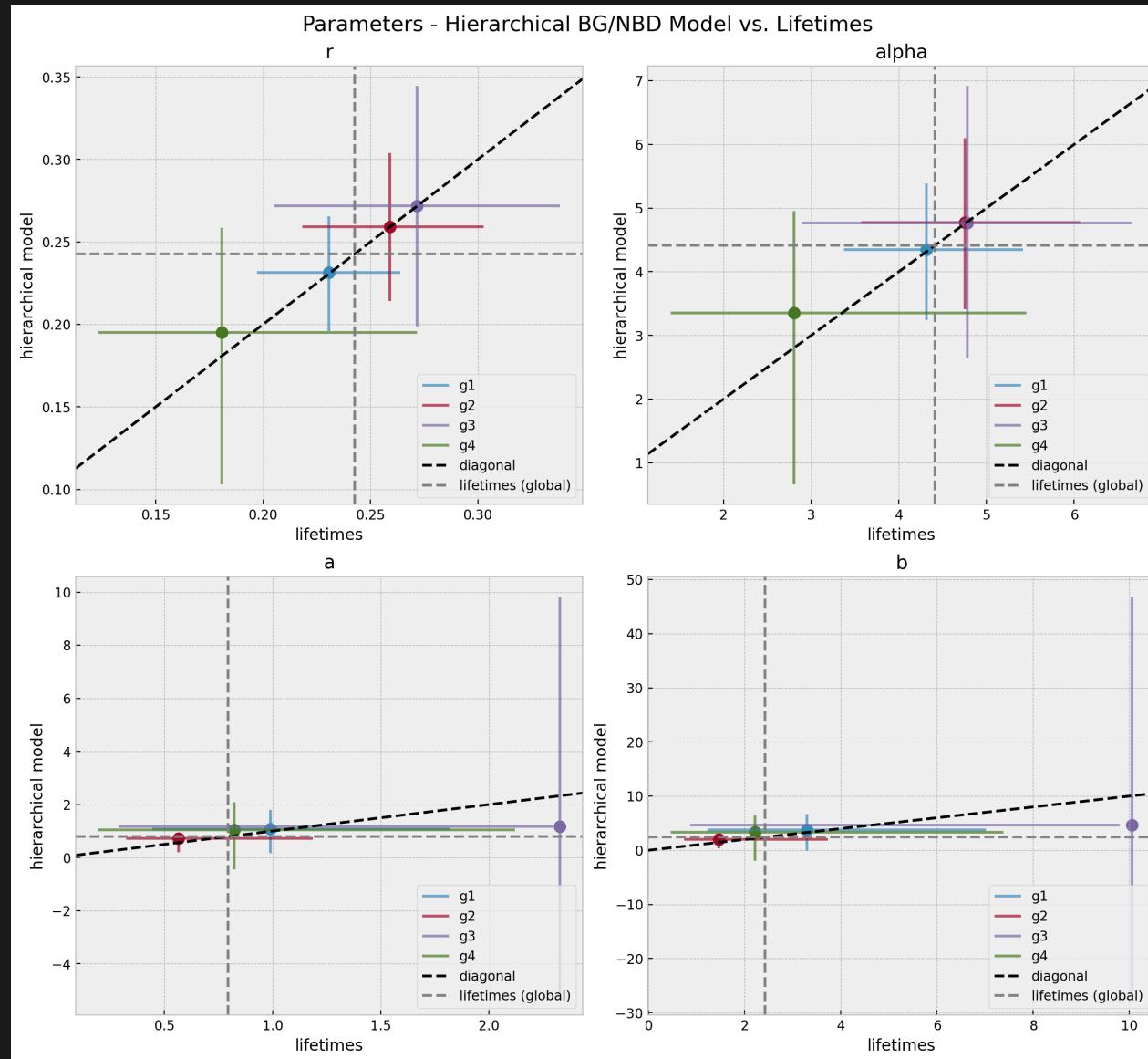
```
1 groups = [ "g1", "g2", "g3", "g4" ]  
2  
3 data_df[ "group" ] = rng.choice(  
4     a=groups, p=[0.45, 0.35, 0.15, 0.05], size=n_obs  
5 )
```



Hierarchical BG/NBD Model Structure



Hierarchical BG/NBD Model Results



Bayesian BG/NBD Model

It opens many possibilities and opportunities!

- Time-Invariant Covariates
- Hierarchical Models
- ...

A successor package of lifetimes

ColtAllen/btyd Public
forked from CamDavidsonPilon/lifetimes

Watch 10 Fork 354 Starred 60

Code Issues 8 Pull requests Discussions Actions Projects 7 Wiki Security Insights

main 1 branch 2 tags Go to file Add file Code

This branch is 147 commits ahead, 3 commits behind CamDavidsonPilon:master. Contribute Sync fork

| ColtAllen Update README.md | 6478741 on 29 Jul 723 commits |
|----------------------------|--|
| .github/workflows | rm python 3.10 as scipy shows installs problems |
| btyd | Ran black code formatter on models module |
| docs | Update Changelog.rst |
| tests | Model saving and loading now internally supported. |
| .gitignore | Model saving and loading now internally supported. |
| CHANGELOG.md | Update CHANGELOG.md |
| CODE_OF_CONDUCT.md | New Code of Conduct added. |
| CONTRIBUTING.md | Minor changes to CONTRIBUTING guide. |
| LICENSE | Switched to Apache 2.0 License. |
| Makefile | remove lock file |

About

Buy Till You Die and Customer Lifetime Value statistical models in Python.

btyd.readthedocs.io/

python data-science bayesian customer-lifetime-value buy-til-you-die

Readme Apache-2.0 license Code of conduct 60 stars 10 watching 354 forks

Releases 2 tags Create a new release

Contributors needed!

References

Blog Posts

- BG/NBD Model in PyMC
- Gamma-Gamma Model of Monetary Value in PyMC

Papers

- Probability Models for Customer-Base Analysis
- Counting Your Customers: Who Are They and What Will They Do Next?
- “Counting Your Customers” the Easy Way: An Alternative to the Pareto/NBD Model
- Computing P(alive) Using the BG/NBD Model

Videos

- New Perspectives on CLV and E-Commerce Buying Patterns, Peter Fader

References

Software

Python

- `lifetimes`
- A successor package `btyd` of `lifetimes`.
- `PyMC`

R

- `BTYD`
- `BTYDplus`

Thank you!

juanitorduz.github.io

