

COHORT REVENUE & RETENTION ANALYSIS: A BAYESIAN APPROACH

JUAN CAMILO ORDUZ

ABSTRACT. We present a bayesian approach to model cohort-level retention rates and revenue over time. We use bayesian additive regression trees (BART) to model the retention component which we couple with a linear model to model the revenue component. This method is flexible enough to allow adding additional covariates to both model components. This bayesian model allows us to quantify the uncertainty in the estimation, understand the effect of the covariates on the retention through partial dependence plots (PDP), individual conditional expectation (ICE) plots and last but not least forecast the future revenue and retention rates.

CONTENTS

1. Introduction	1
2. Data and Exploratory	1
3. Model Specification and Diagnostics	1
4. Predictions	1
4.1. In-Sample Predictions	1
4.2. Out-of-Sample Predictions	1

1. INTRODUCTION

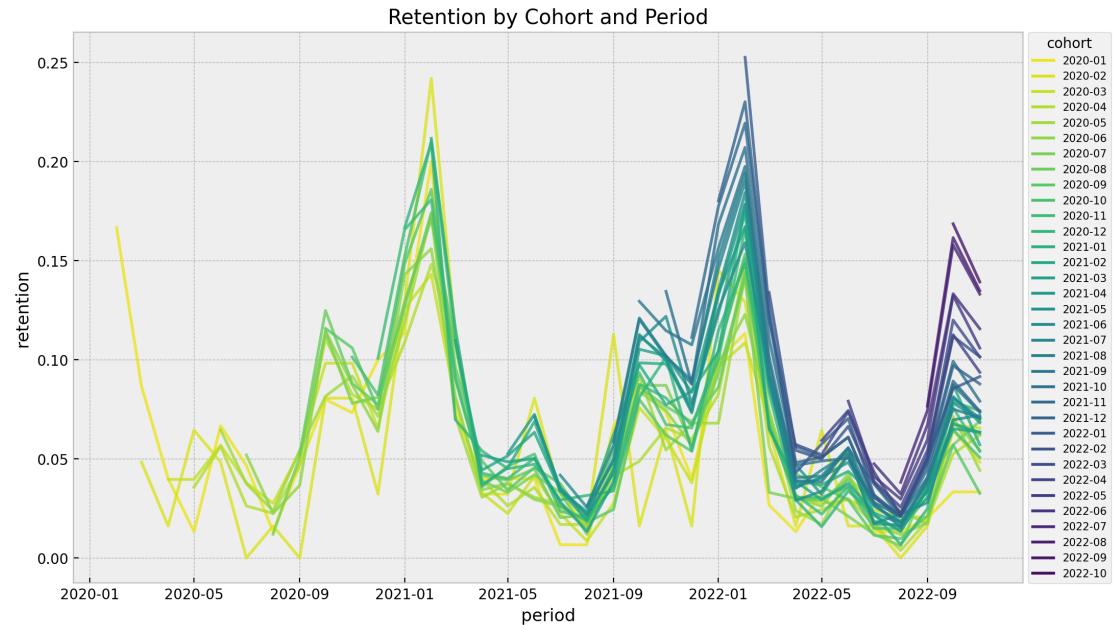
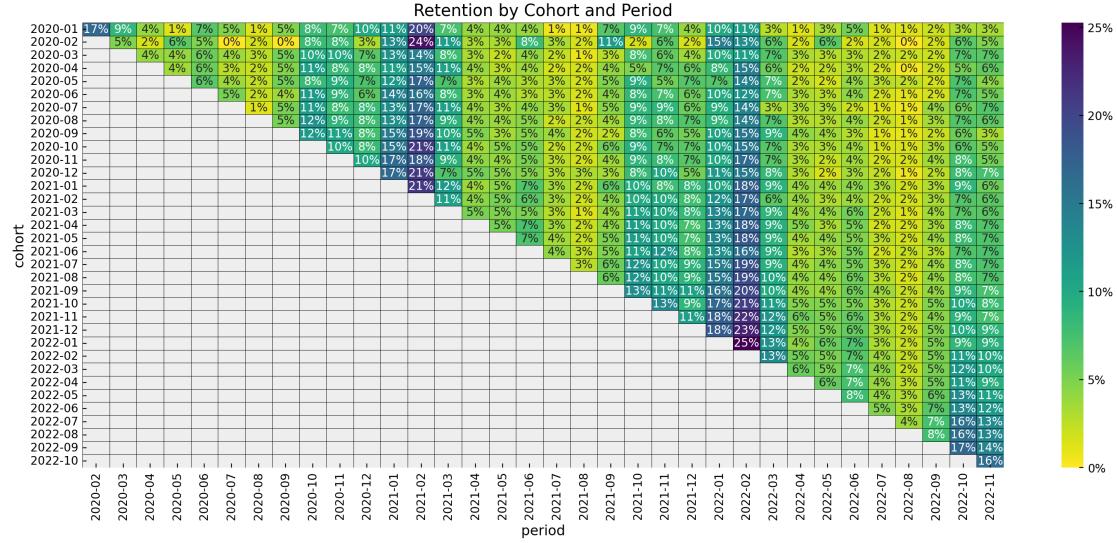
2. DATA AND EXPLORATORY

3. MODEL SPECIFICATION AND DIAGNOSTICS

4. PREDICTIONS

4.1. In-Sample Predictions.

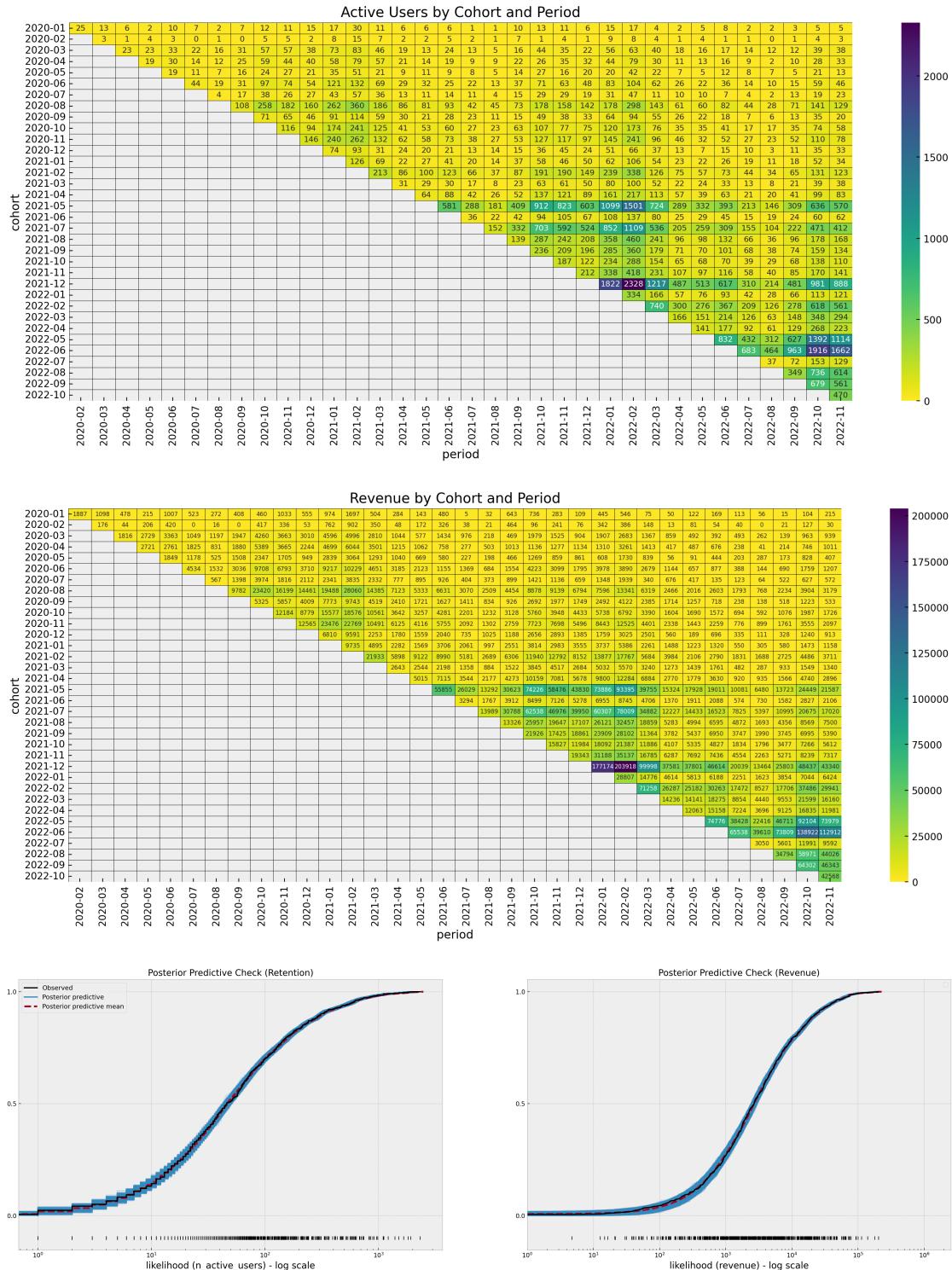
4.2. Out-of-Sample Predictions.



Appendix: Python Code

LISTING 1. Python example

```
1 import pymc_bart as pmb
2 import pymc as pm
3
4
5 with pm.Model(coords={"feature": features}) as model:
```



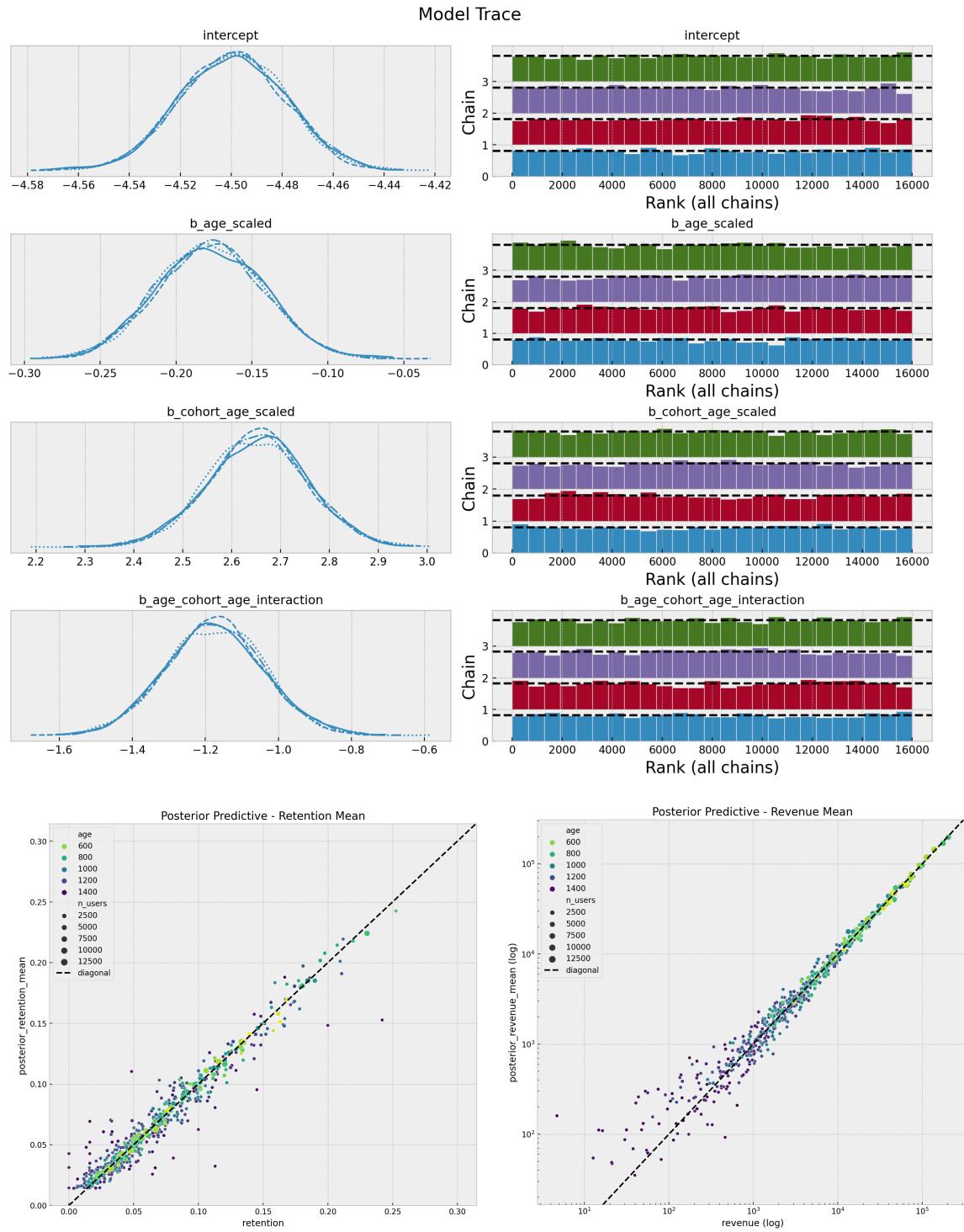


FIGURE 1. My caption

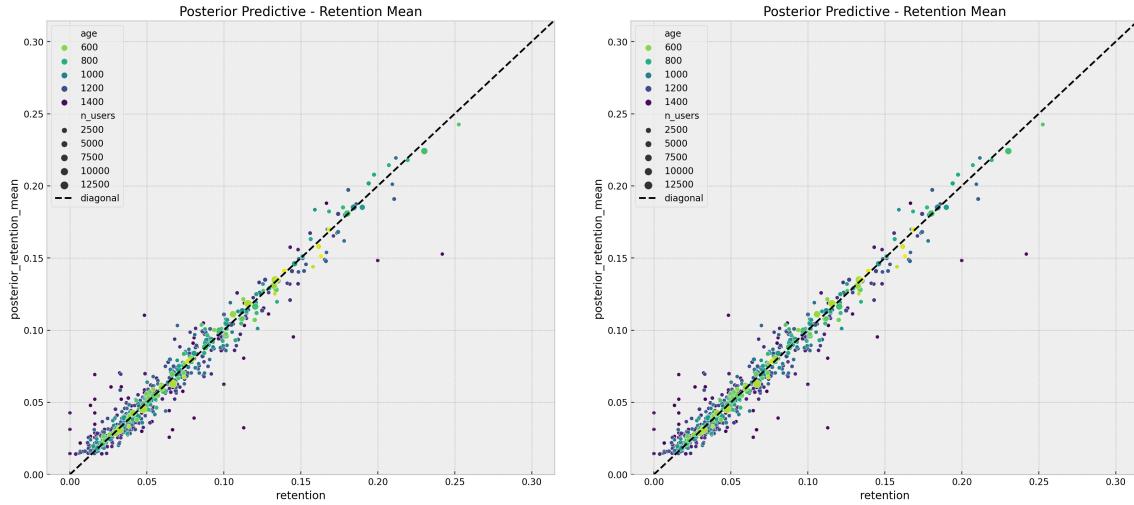
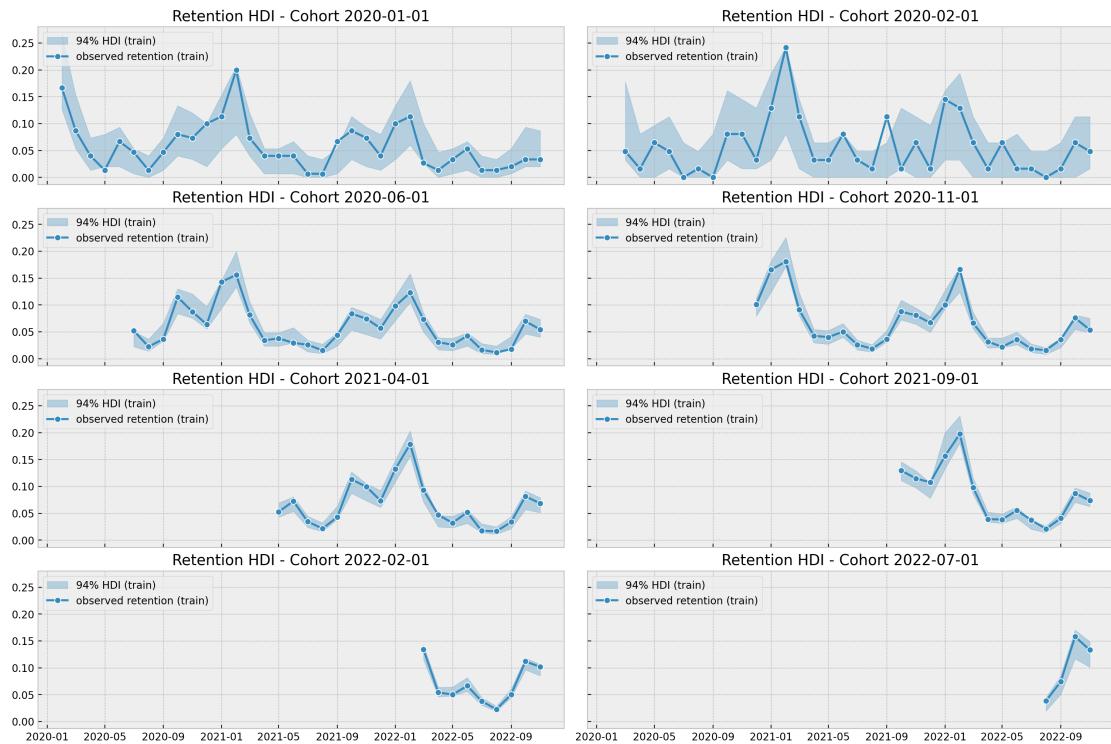


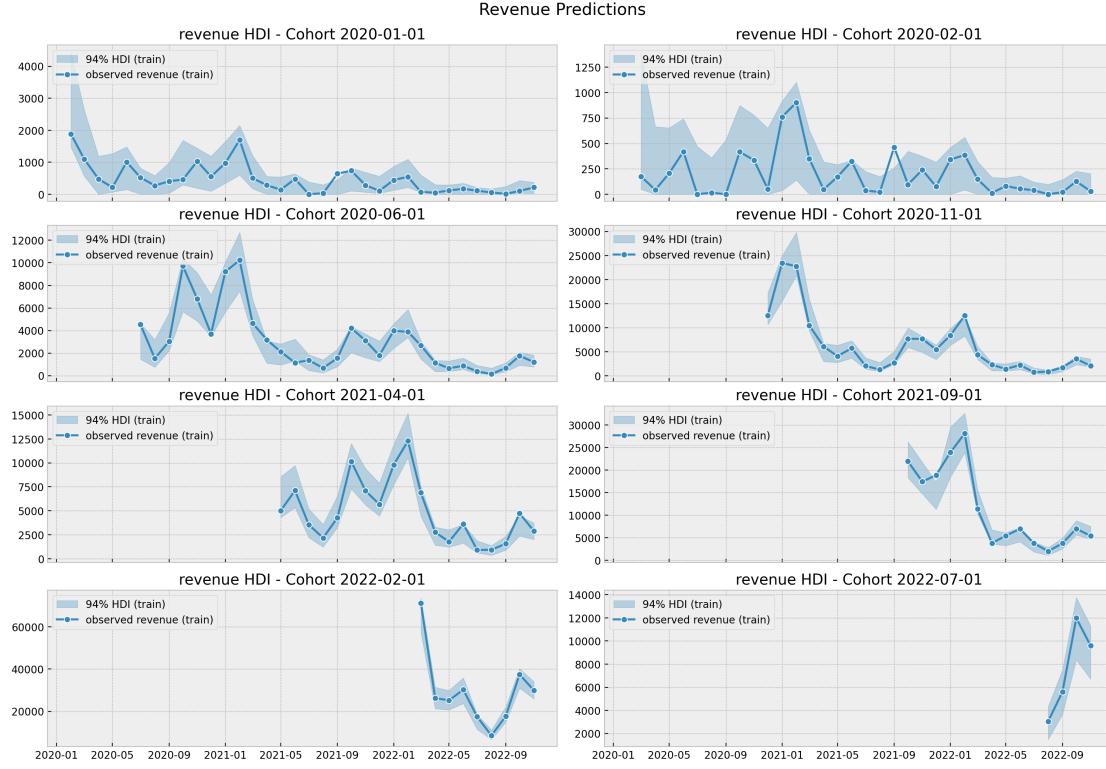
FIGURE 2. My caption



```

7 # --- Data ---
8 model.add_coord(name="obs", values=train_obs_idx, mutable=True)
9 age_scaled = pm.MutableData(
10     name="age_scaled", value=train_age_scaled, dims="obs"
11 )

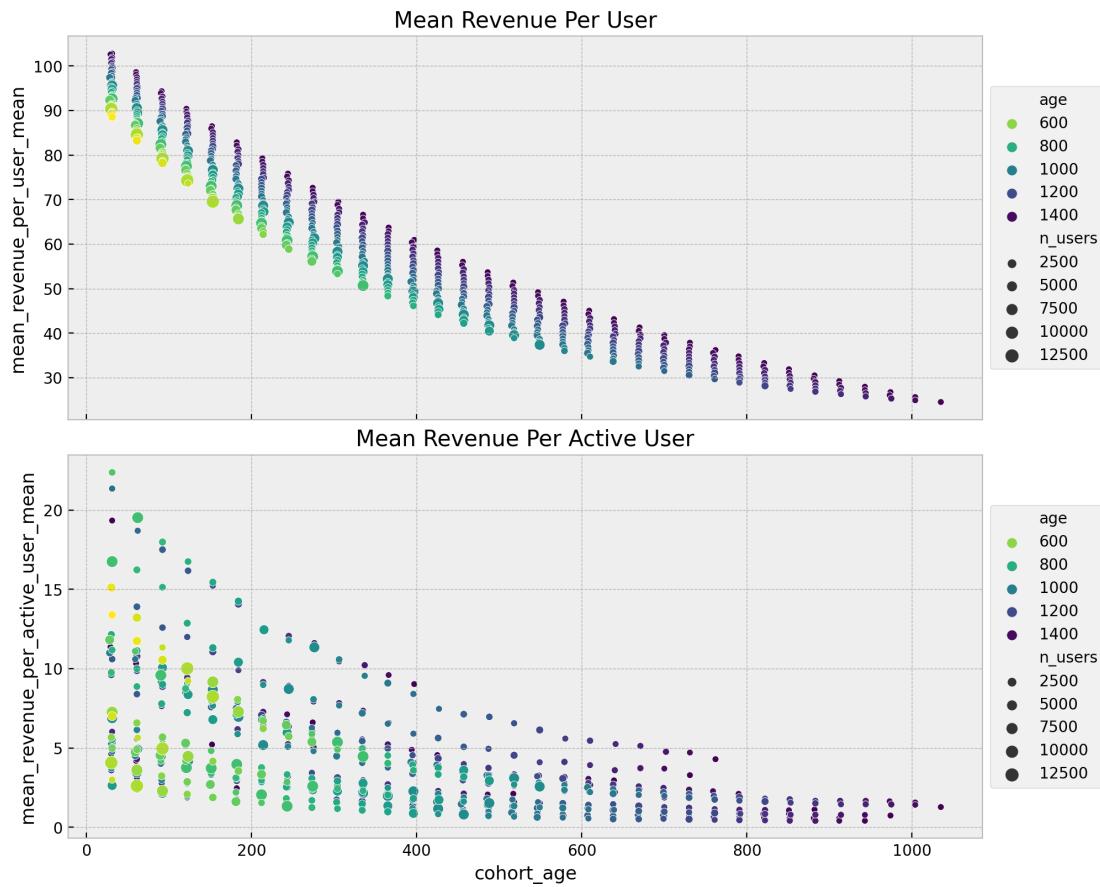
```



```

12 cohort_age_scaled = pm.MutableData(
13     name="cohort_age_scaled", value=train_cohort_age_scaled, dims="obs"
14 )
15 x = pm.MutableData(name="x", value=x_train, dims=("obs", "feature"))
16 n_users = pm.MutableData(name="n_users", value=train_n_users, dims="obs")
17 n_active_users = pm.MutableData(
18     name="n_active_users", value=train_n_active_users, dims="obs"
19 )
20 revenue = pm.MutableData(name="revenue", value=train_revenue, dims="obs")
21
22 # --- Priors ---
23 intercept = pm.Normal(name="intercept", mu=0, sigma=1)
24 b_age_scaled = pm.Normal(name="b_age_scaled", mu=0, sigma=1)
25 b_cohort_age_scaled = pm.Normal(name="b_cohort_age_scaled", mu=0, sigma=1)
26 b_age_cohort_age_interaction = pm.Normal(
27     name="b_age_cohort_age_interaction", mu=0, sigma=1
28 )
29
30 # --- Parametrization ---
31 # The BART component models the image of the retention rate under the
32 # logit transform so that the range is not constrained to [0, 1].
33 mu = pmb.BART(name="mu", X=x, Y=train_retention_logit, m=50, dims="obs")
34 # We use the inverse logit transform to get the retention rate

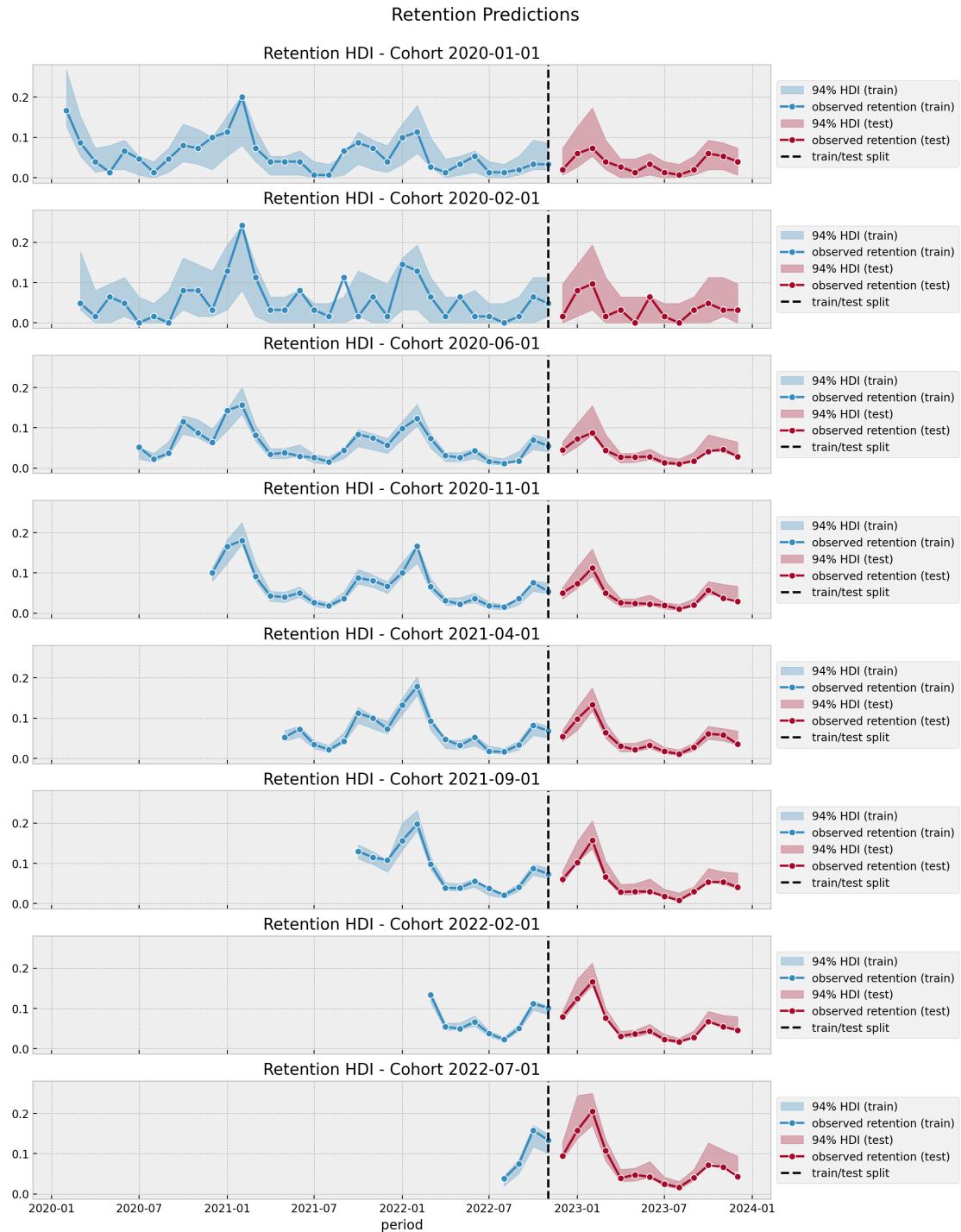
```

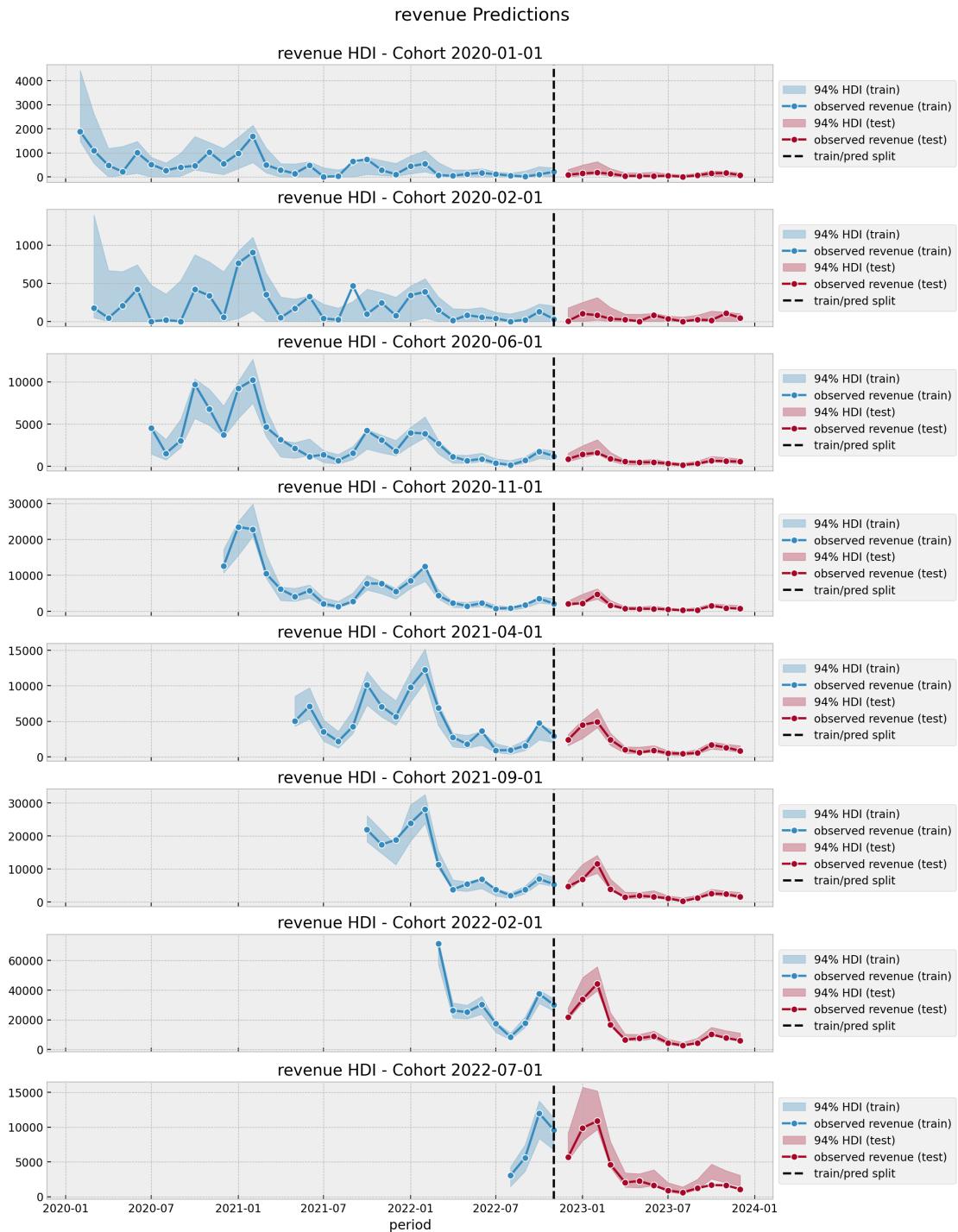


```

35     # back into [0, 1].
36     p = pm.Deterministic(name="p", var=pm.math.invlogit(mu), dims="obs")
37     # We add a small epsilon to avoid numerical issues.
38     p = pt.switch(pt.eq(p, 0), eps, p)
39     p = pt.switch(pt.eq(p, 1), 1 - eps, p)
40
41     # For the revenue component we use a Gamma distribution where we
42     # combine the number of estimated active users with the average
43     # revenue per user.
44     lam_log = pm.Deterministic(
45         name="lam_log",
46         var=intercept
47         + b_age_scaled * age_scaled
48         + b_cohort_age_scaled * cohort_age_scaled
49         + b_age_cohort_age_interaction * age_scaled * cohort_age_scaled,
50         dims="obs",
51     )
52
53     lam = pm.Deterministic(name="lam", var=pm.math.exp(lam_log), dims="obs")

```

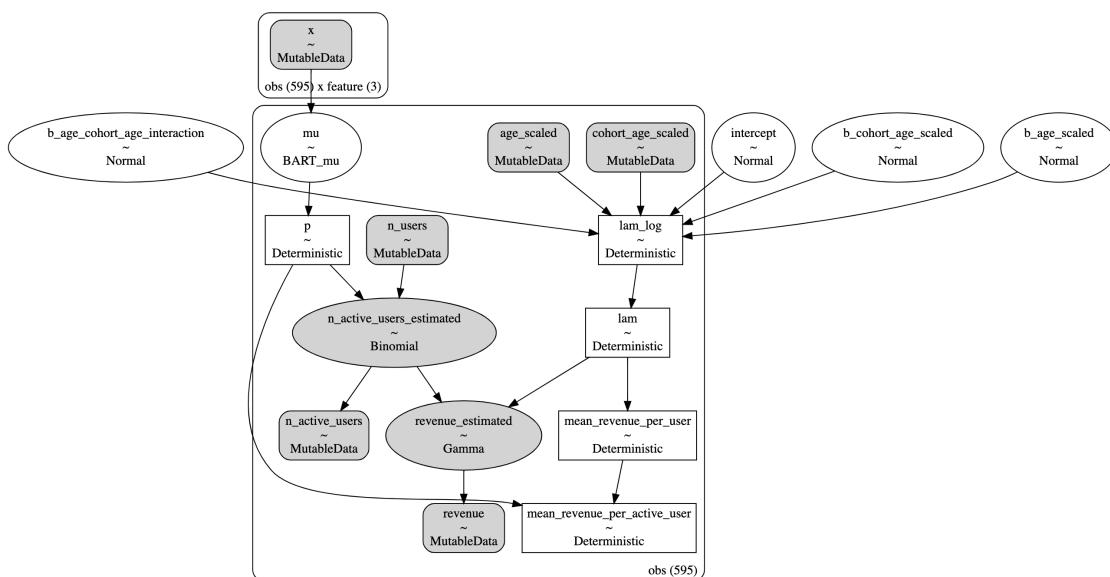




```

54
55     # --- Likelihood ---
56     n_active_users_estimated = pm.Binomial(
57         name="n_active_users_estimated",
58         n=n_users,
59         p=p,
60         observed=n_active_users,
61         dims="obs",
62     )
63
64     x = pm.Gamma(
65         name="revenue_estimated",
66         alpha=n_active_users_estimated + eps,
67         beta=lam,
68         observed=revenue,
69         dims="obs",
70     )
71
72     # --- Derived Quantities ---
73     mean_revenue_per_user = pm.Deterministic(
74         name="mean_revenue_per_user", var=(1 / lam), dims="obs"
75     )
76     pm.Deterministic(
77         name="mean_revenue_per_active_user",
78         var=p * mean_revenue_per_user,
79         dims="obs"
80     )

```



BERLIN, GERMANY