Bayesian Workflow with PyMC and ArviZ

Corrie Bartelheimer

Data Scientist at Europace AG

@corrieaar

europace®

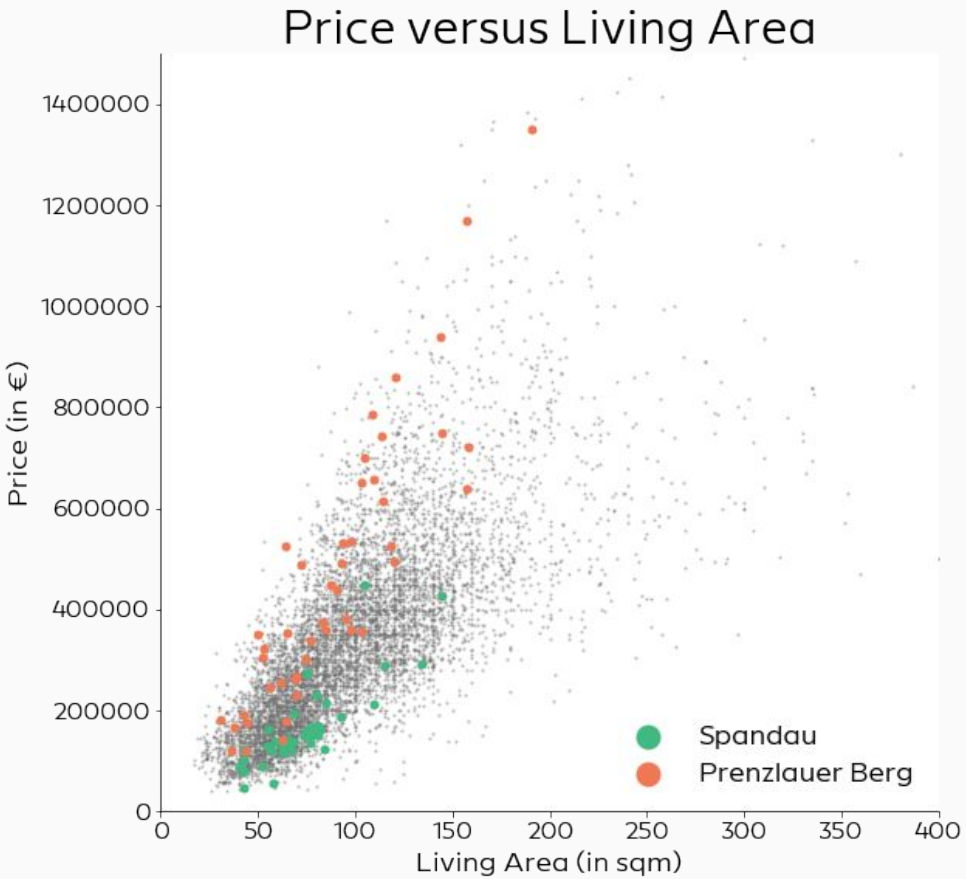
PyData

First, the problem

First, the problem

First, the problem

First, the problem

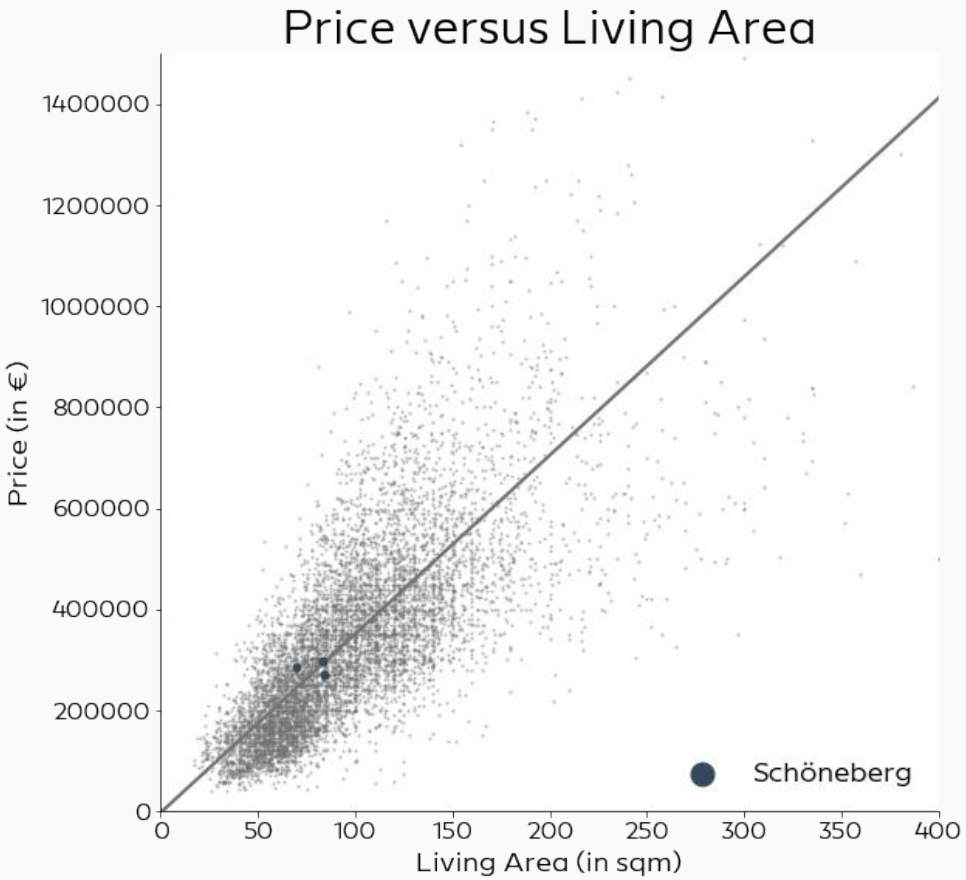


First, the problem

First, the problem

First, the problem

First, the problem



Solution:
Hierarchical Bayesian Model

Solution: Hierarchical Bayesian Model

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha + \beta \, \text{area}$$

Solution: Hierarchical Bayesian Model

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha + \beta \, \text{area}$$

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$

$$\sigma \sim \text{Exponential}(1/100)$$

Solution: Hierarchical Bayesian Model

$$y \sim \text{Normal}(\mu, \sigma)$$

$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

Solution: Hierarchical Bayesian Model

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

Solution: Hierarchical Bayesian Model

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]} \, \text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha, \mu_\beta \sim \text{Normal}(0, 100)$$
$$\sigma_\alpha, \sigma_\beta, \sigma \sim \text{Exponential}(1/100)$$

Getting this into Python

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha + \beta \, \text{area}$$

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$

$$\sigma \sim \text{Exponential}(1/100)$$

```
import pymc3 as pm

with pm.Model() as lin_model:

    α = pm.Normal("α", 0, 100)
    β = pm.Normal("β", 0, 100)
    σ = pm.Exponential("σ", 1/100)

    μ = α + β*d["area"]

    y = pm.Normal("y", μ, σ,
            observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$

$$\mu = \alpha + \beta \, \text{area}$$

$$\alpha \sim \text{Normal}(0, 100)$$

$$\beta \sim \text{Normal}(0, 100)$$

$$\sigma \sim \text{Exponential}(1/100)$$

```
import pymc3 as pm

with pm.Model() as lin_model:

    α = pm.Normal("α", 0, 100)
    β = pm.Normal("β", 0, 100)
    σ = pm.Exponential("σ", 1/100)

    μ = α + β*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$

$$\mu = \alpha + \beta \, \text{area}$$

$$\alpha \sim \text{Normal}(0, 100)$$

$$\beta \sim \text{Normal}(0, 100)$$

$$\sigma \sim \text{Exponential}(1/100)$$

```
import pymc3 as pm

with pm.Model() as lin_model:

    α = pm.Normal("α", 0, 100)
    β = pm.Normal("β", 0, 100)
    σ = pm.Exponential("σ", 1/100)

    μ = α + β*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha + \beta \, \text{area}$$

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$

$$\sigma \sim \text{Exponential}(1/100)$$

```
import pymc3 as pm

with pm.Model() as lin_model:

    α = pm.Normal("α", 0, 100)
    β = pm.Normal("β", 0, 100)
    σ = pm.Exponential("σ", 1/100)

    μ = α + β*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha, \mu_\beta \sim \text{Normal}(0, 100)$$
$$\sigma_\alpha, \sigma_\beta, \sigma \sim \text{Exponential}(1/100)$$

```
with pm.Model() as hier_model:

    μ_α = pm.Normal("μ_α", 0, 100)
    μ_β = ...
    σ = pm.Exponential("σ", 1/100)
    σ_α = σ_β = ...

    α = pm.Normal("α", μ_α, σ_α,
                  shape=num_zip)
    β = pm.Normal("β", μ_β, σ_β,
                  shape=num_zip)

    μ = α[d["zip"]] + β[d["zip"]]*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$

$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha, \mu_\beta \sim \text{Normal}(0, 100)$$
$$\sigma_\alpha, \sigma_\beta, \sigma \sim \text{Exponential}(1/100)$$

```
with pm.Model() as hier_model:

    μ_α = pm.Normal("μ_α", 0, 100)
    μ_β = ...
    σ = pm.Exponential("σ", 1/100)
    σ_α = σ_β = ...

    α = pm.Normal("α", μ_α, σ_α,
                  shape=num_zip)
    β = pm.Normal("β", μ_β, σ_β,
                  shape=num_zip)

    μ = α[d["zip"]] + β[d["zip"]]*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha, \mu_\beta \sim \text{Normal}(0, 100)$$
$$\sigma_\alpha, \sigma_\beta, \sigma \sim \text{Exponential}(1/100)$$

```
with pm.Model() as hier_model:

    μ_α = pm.Normal("μ_α", 0, 100)
    μ_β = ...
    σ = pm.Exponential("σ", 1/100)
    σ_α = σ_β = ...

    α = pm.Normal("α", μ_α, σ_α,
                  shape=num_zip)
    β = pm.Normal("β", μ_β, σ_β,
                  shape=num_zip)

    μ = α[d["zip"]] + β[d["zip"]]*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

Getting this into Python: PyMC3

$$y \sim \text{Normal}(\mu, \sigma)$$
$$\mu = \alpha_{[ZIP]} + \beta_{[ZIP]}\,\text{area}$$

$$\alpha_{[ZIP]} \sim \text{Normal}(\mu_\alpha, \sigma_\alpha)$$
$$\beta_{[ZIP]} \sim \text{Normal}(\mu_\beta, \sigma_\beta)$$

$$\mu_\alpha, \mu_\beta \sim \text{Normal}(0, 100)$$
$$\sigma_\alpha, \sigma_\beta, \sigma \sim \text{Exponential}(1/100)$$

```
with pm.Model() as hier_model:

    μ_α = pm.Normal("μ_α", 0, 100)
    μ_β = ...
    σ = pm.Exponential("σ", 1/100)
    σ_α = σ_β = ...

    α = pm.Normal("α", μ_α, σ_α,
                  shape=num_zip)
    β = pm.Normal("β", μ_β, σ_β,
                  shape=num_zip)

    μ = α[d["zip"]] + β[d["zip"]]*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
```

What about the priors?

What about the priors?

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$

What about the priors?

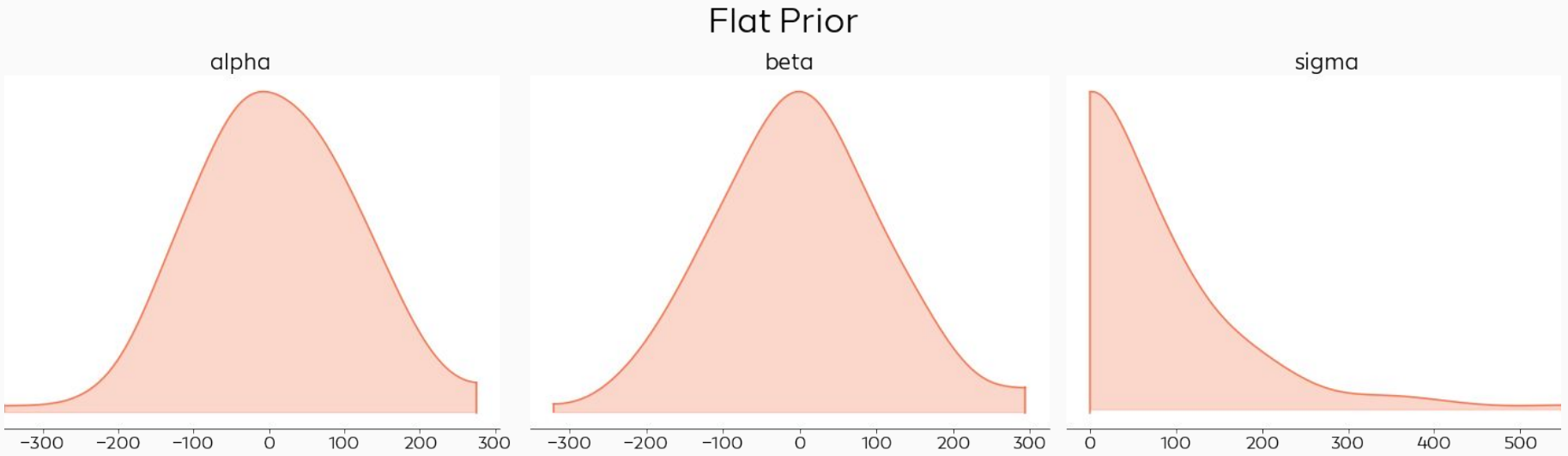
$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$

```
with model:
    prior = pm.sample_prior_predictive()
```

What about the priors?

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$

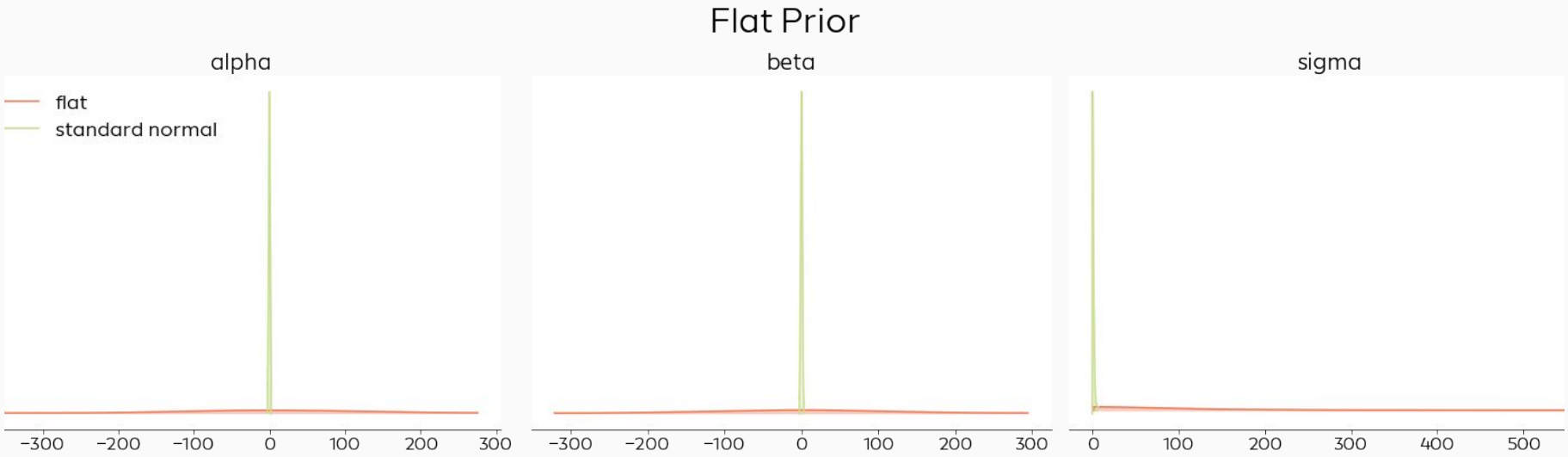
```
with model:
    prior = pm.sample_prior_predictive()
```



What about the priors?

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$

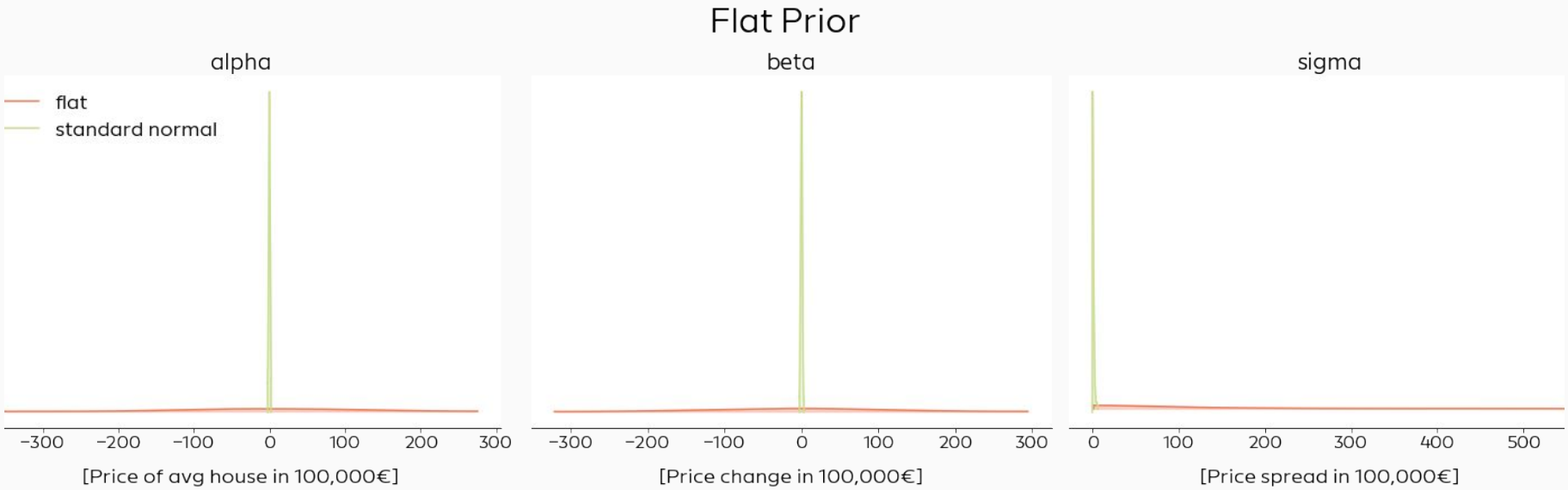
```
with model:
    prior = pm.sample_prior_predictive()
```



What about the priors?

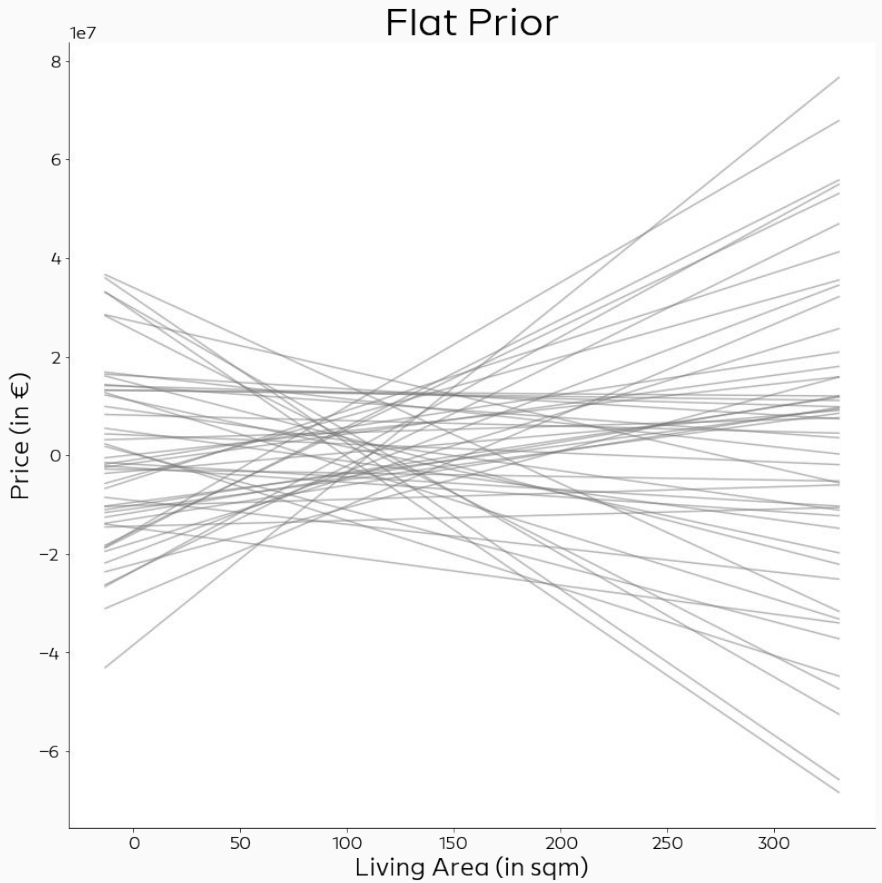
$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$

```
with model:
    prior = pm.sample_prior_predictive()
```



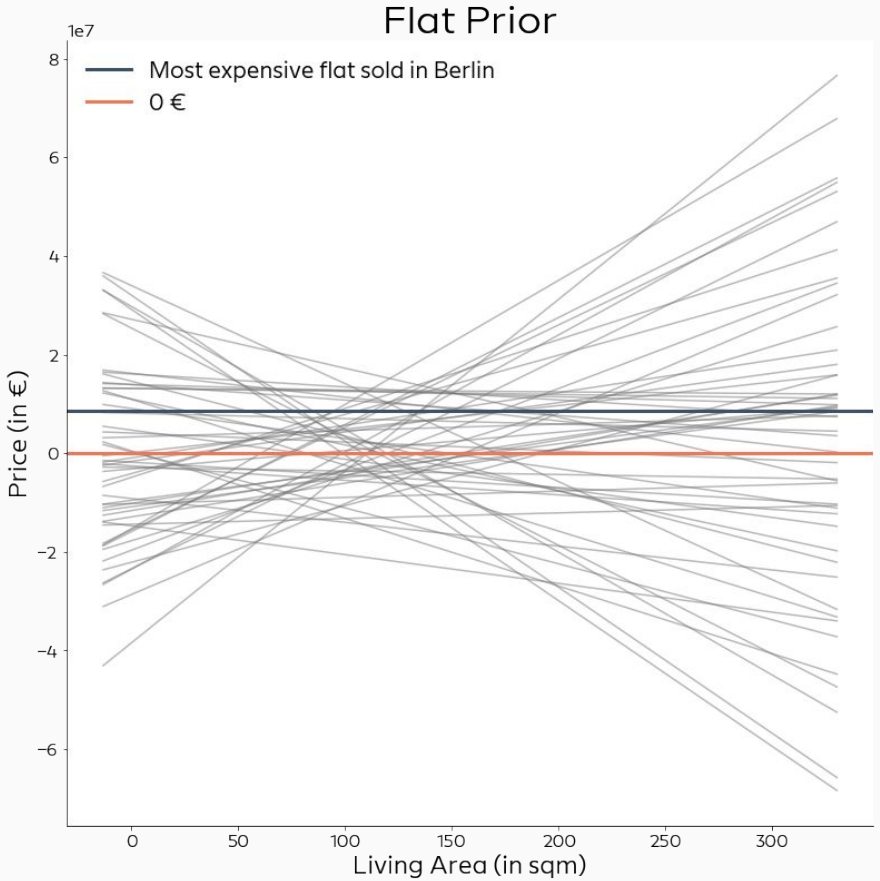
What about the priors?

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$



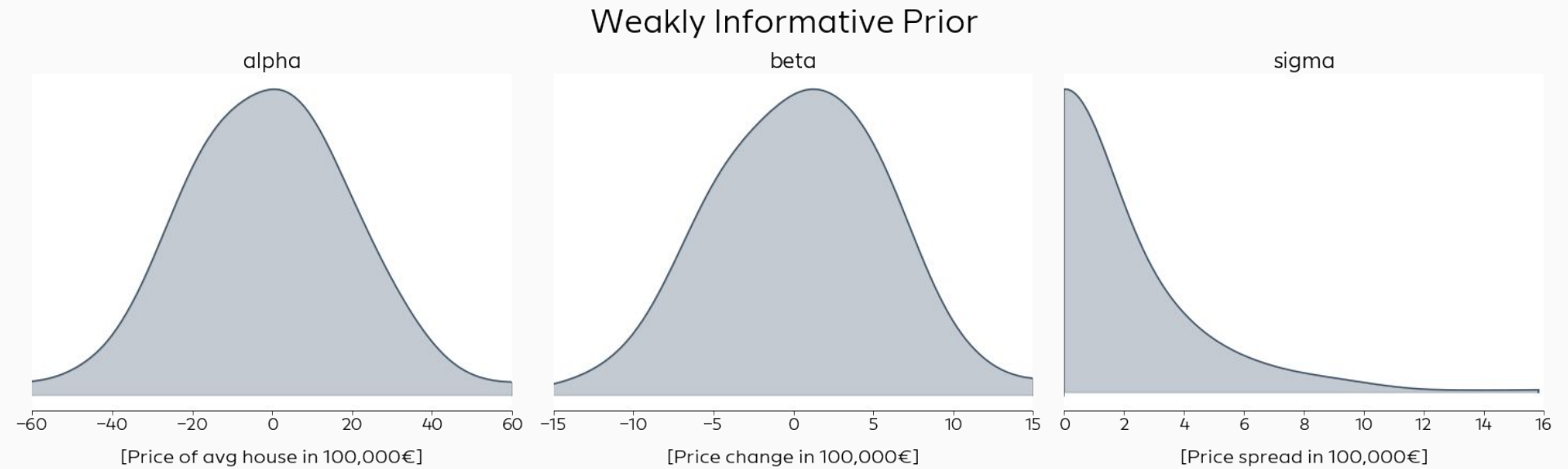
What about the priors?

$$\alpha \sim \text{Normal}(0, 100)$$
$$\beta \sim \text{Normal}(0, 100)$$
$$\sigma \sim \text{Exponential}(1/100)$$



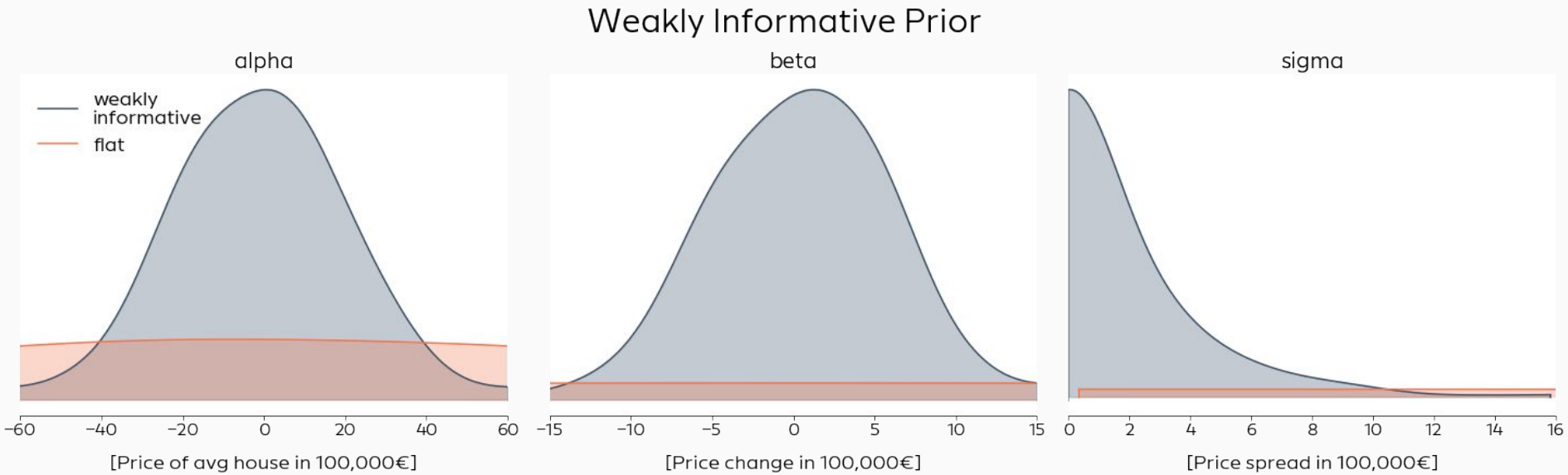
What about the priors?

$$\alpha \sim \text{Normal}(0, 20)$$
$$\beta \sim \text{Normal}(0, 5)$$
$$\sigma \sim \text{Exponential}(1/5)$$



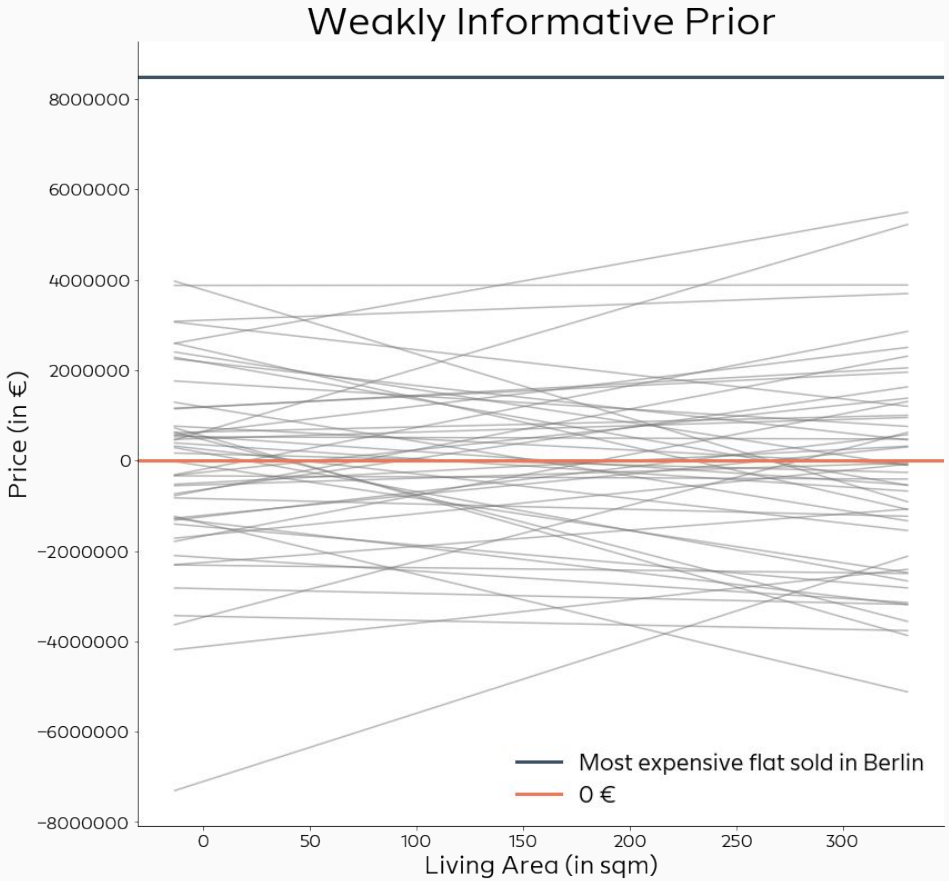
What about the priors?

$$\alpha \sim \text{Normal}(0, 20)$$
$$\beta \sim \text{Normal}(0, 5)$$
$$\sigma \sim \text{Exponential}(1/5)$$



What about the priors?

$$\alpha \sim \text{Normal}(0, 20)$$
$$\beta \sim \text{Normal}(0, 5)$$
$$\sigma \sim \text{Exponential}(1/5)$$



What about the priors?

$$\alpha \sim \text{Normal}(0, 20)$$
$$\beta \sim \text{Normal}(0, 5)$$
$$\sigma \sim \text{Exponential}(1/5)$$

```
with pm.Model() as hier_model:

    μ_α = pm.Normal("μ_α", 0, 20)
    μ_β = pm.Normal("μ_β", 0, 5)
    σ = pm.Exponential("σ", 1/5)
    σ_α = σ_β = ...

    α = pm.Normal("α", μ_α, σ_α,
                  shape=num_zip)
    β = pm.Normal("β", μ_β, σ_β,
                  shape=num_zip)

    μ = α[d["zip"]] + β[d["zip"]]*d["area"]

    y = pm.Normal("y", μ, σ,
                  observed=d["price"])
    trace = pm.sample()
```

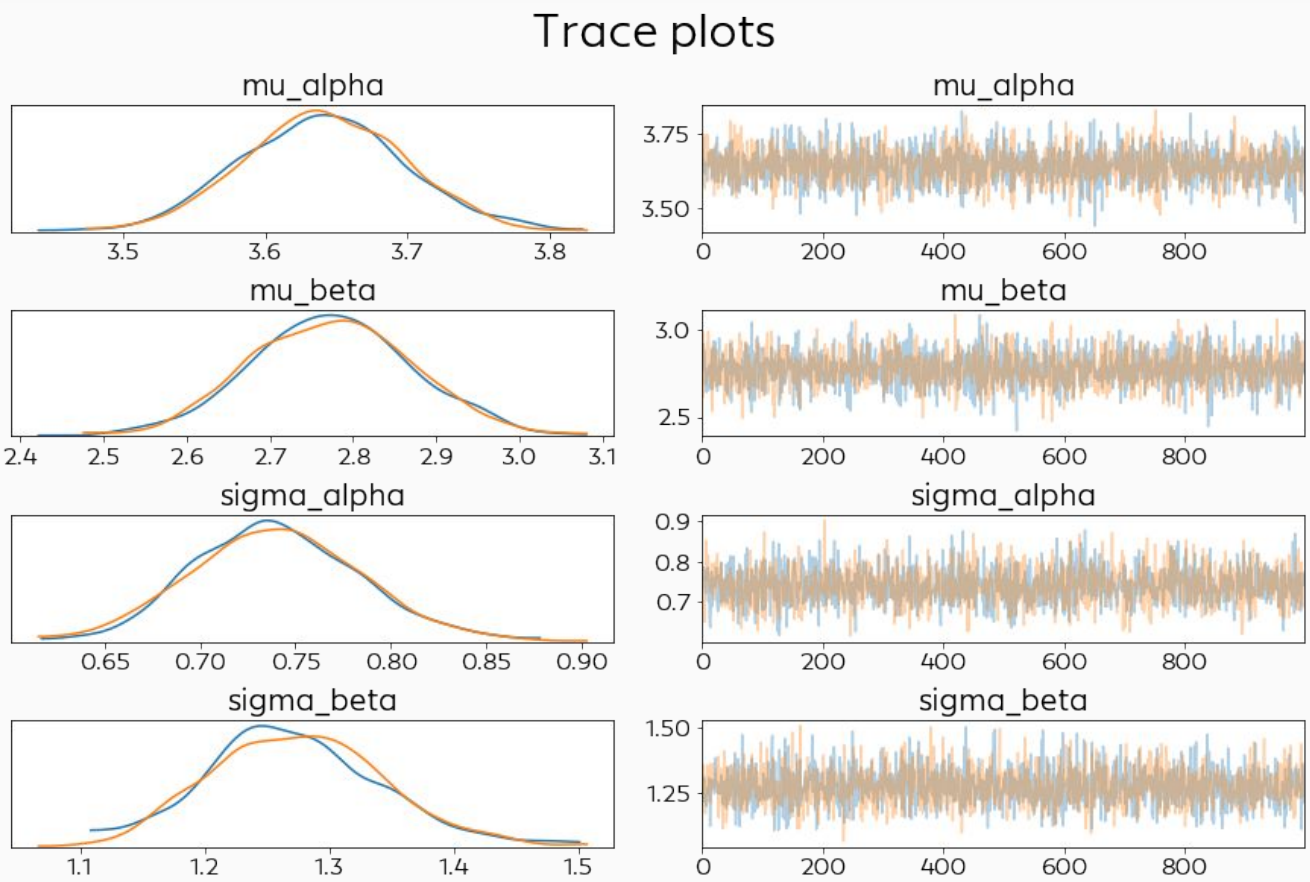
Did it converge?

Did it converge?

```
import arviz as az

az.plot_trace(trace)
```

Did it converge?

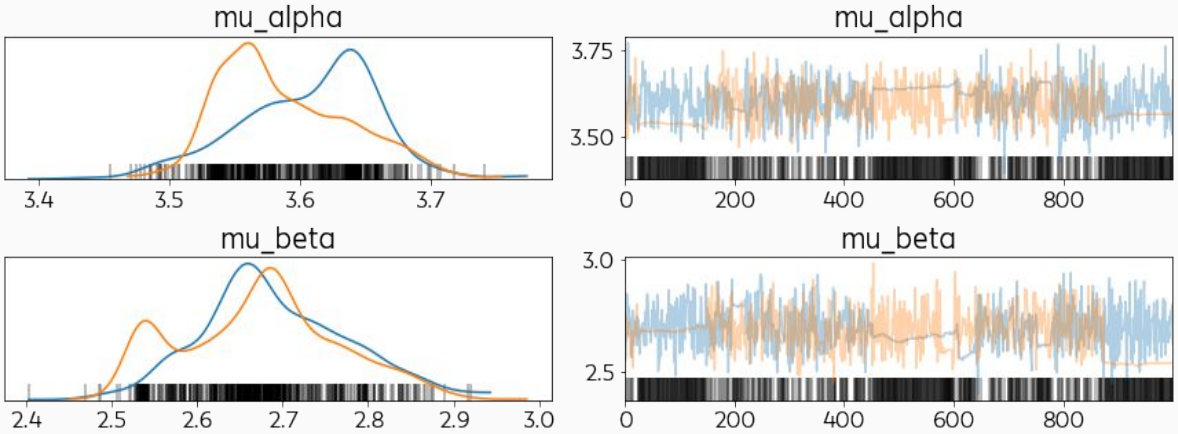


Did it converge?

Some Bad Examples

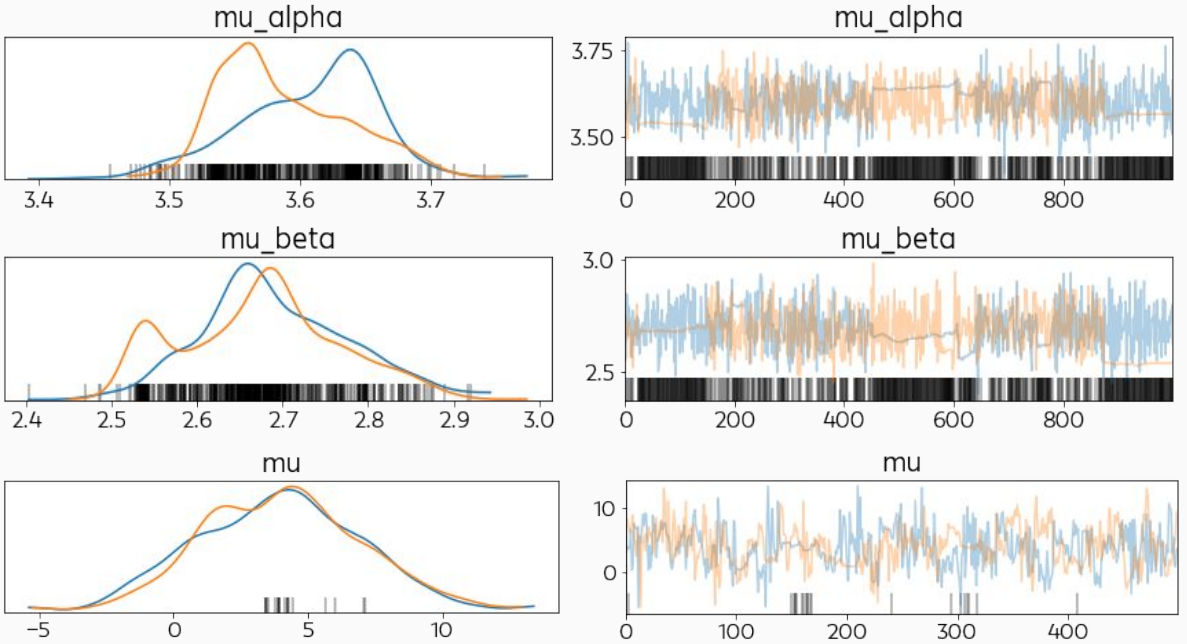
Did it converge?

Some Bad Examples



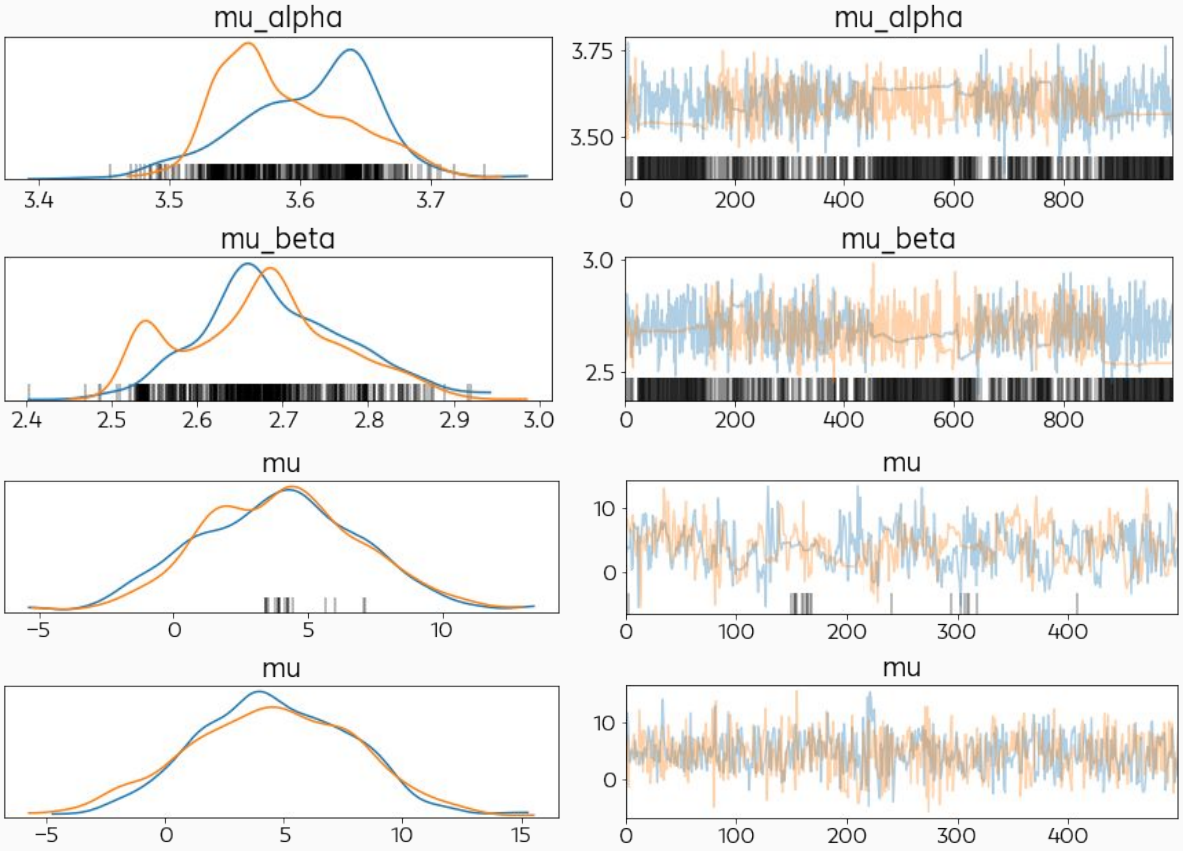
Did it converge?

Some Bad Examples



Did it converge?

Some Bad Examples



Did it converge?

```
az.summary(trace)
```

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	ess_bulk	ess_tail	r_hat
mu_alpha	3.642	0.057	3.527	3.741	0.001	0.001	5524.0	5524.0	5535.0	3184.0	1.0
mu_beta	2.775	0.099	2.596	2.966	0.001	0.001	5782.0	5781.0	5766.0	3343.0	1.0
alpha[0]	4.976	0.161	4.688	5.289	0.002	0.002	5618.0	5590.0	5626.0	3249.0	1.0
alpha[1]	5.097	0.289	4.545	5.616	0.004	0.003	6018.0	6018.0	6017.0	3309.0	1.0
alpha[2]	5.128	0.201	4.731	5.483	0.002	0.002	7223.0	7223.0	7272.0	3517.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...
beta[217]	2.668	1.280	0.290	4.958	0.014	0.012	7809.0	5449.0	7833.0	2742.0	1.0
beta[218]	0.504	0.792	-0.999	1.999	0.009	0.011	7653.0	2519.0	7656.0	3127.0	1.0
sigma_alpha	0.741	0.044	0.664	0.830	0.001	0.000	4491.0	4491.0	4466.0	3015.0	1.0
sigma_beta	1.274	0.071	1.146	1.415	0.001	0.001	4779.0	4779.0	4726.0	3035.0	1.0
sigma	1.203	0.010	1.184	1.221	0.000	0.000	7403.0	7403.0	7381.0	3018.0	1.0

443 rows × 11 columns

Did it converge?

```
az.summary(trace)
```

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	ess_bulk	ess_tail	r_hat
mu_alpha	3.642	0.057	3.527	3.741	0.001	0.001	5524.0	5524.0	5535.0	3184.0	1.0
mu_beta	2.775	0.099	2.596	2.966	0.001	0.001	5782.0	5781.0	5766.0	3343.0	1.0
alpha[0]	4.976	0.161	4.688	5.289	0.002	0.002	5618.0	5590.0	5626.0	3249.0	1.0
alpha[1]	5.097	0.289	4.545	5.616	0.004	0.003	6018.0	6018.0	6017.0	3309.0	1.0
alpha[2]	5.128	0.201	4.731	5.483	0.002	0.002	7223.0	7223.0	7272.0	3517.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...
beta[217]	2.668	1.280	0.290	4.958	0.014	0.012	7809.0	5449.0	7833.0	2742.0	1.0
beta[218]	0.504	0.792	-0.999	1.999	0.009	0.011	7653.0	2519.0	7656.0	3127.0	1.0
sigma_alpha	0.741	0.044	0.664	0.830	0.001	0.000	4491.0	4491.0	4466.0	3015.0	1.0
sigma_beta	1.274	0.071	1.146	1.415	0.001	0.001	4779.0	4779.0	4726.0	3035.0	1.0
sigma	1.203	0.010	1.184	1.221	0.000	0.000	7403.0	7403.0	7381.0	3018.0	1.0

443 rows × 11 columns

Did it converge?

```
az.summary(trace)
```

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	ess_bulk	ess_tail	r_hat
mu_alpha	3.642	0.057	3.527	3.741	0.001	0.001	5524.0	5524.0	5535.0	3184.0	1.0
mu_beta	2.775	0.099	2.596	2.966	0.001	0.001	5782.0	5781.0	5766.0	3343.0	1.0
alpha[0]	4.976	0.161	4.688	5.289	0.002	0.002	5618.0	5590.0	5626.0	3249.0	1.0
alpha[1]	5.097	0.289	4.545	5.616	0.004	0.003	6018.0	6018.0	6017.0	3309.0	1.0
alpha[2]	5.128	0.201	4.731	5.483	0.002	0.002	7223.0	7223.0	7272.0	3517.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...
beta[217]	2.668	1.280	0.290	4.958	0.014	0.012	7809.0	5449.0	7833.0	2742.0	1.0
beta[218]	0.504	0.792	-0.999	1.999	0.009	0.011	7653.0	2519.0	7656.0	3127.0	1.0
sigma_alpha	0.741	0.044	0.664	0.830	0.001	0.000	4491.0	4491.0	4466.0	3015.0	1.0
sigma_beta	1.274	0.071	1.146	1.415	0.001	0.001	4779.0	4779.0	4726.0	3035.0	1.0
sigma	1.203	0.010	1.184	1.221	0.000	0.000	7403.0	7403.0	7381.0	3018.0	1.0

443 rows × 11 columns

Did it converge?

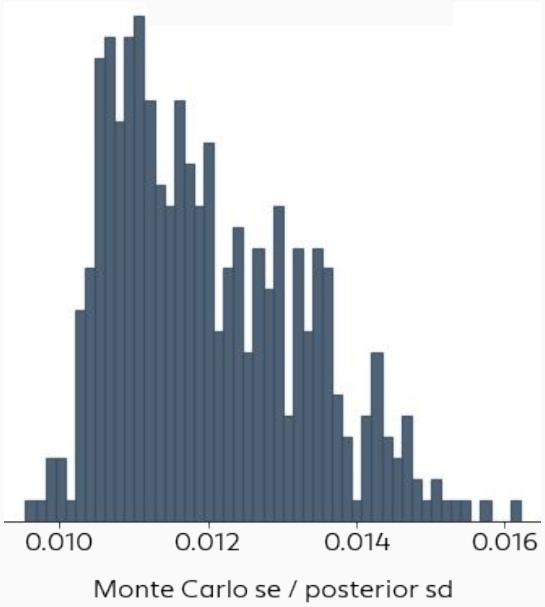
```
az.summary(trace)
```

	mean	sd	hpd_3%	hpd_97%	mcse_mean	mcse_sd	ess_mean	ess_sd	ess_bulk	ess_tail	r_hat
mu_alpha	3.642	0.057	3.527	3.741	0.001	0.001	5524.0	5524.0	5535.0	3184.0	1.0
mu_beta	2.775	0.099	2.596	2.966	0.001	0.001	5782.0	5781.0	5766.0	3343.0	1.0
alpha[0]	4.976	0.161	4.688	5.289	0.002	0.002	5618.0	5590.0	5626.0	3249.0	1.0
alpha[1]	5.097	0.289	4.545	5.616	0.004	0.003	6018.0	6018.0	6017.0	3309.0	1.0
alpha[2]	5.128	0.201	4.731	5.483	0.002	0.002	7223.0	7223.0	7272.0	3517.0	1.0
...	...	...	...	...	...	...	...	...	...	...	...
beta[217]	2.668	1.280	0.290	4.958	0.014	0.012	7809.0	5449.0	7833.0	2742.0	1.0
beta[218]	0.504	0.792	-0.999	1.999	0.009	0.011	7653.0	2519.0	7656.0	3127.0	1.0
sigma_alpha	0.741	0.044	0.664	0.830	0.001	0.000	4491.0	4491.0	4466.0	3015.0	1.0
sigma_beta	1.274	0.071	1.146	1.415	0.001	0.001	4779.0	4779.0	4726.0	3035.0	1.0
sigma	1.203	0.010	1.184	1.221	0.000	0.000	7403.0	7403.0	7381.0	3018.0	1.0

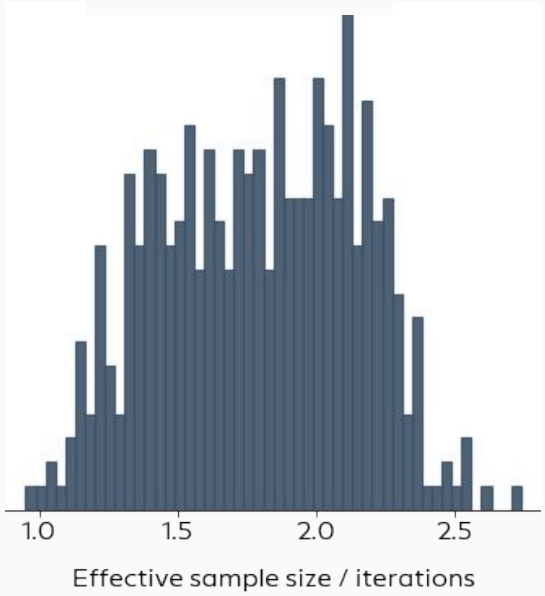
443 rows × 11 columns

Did it converge?

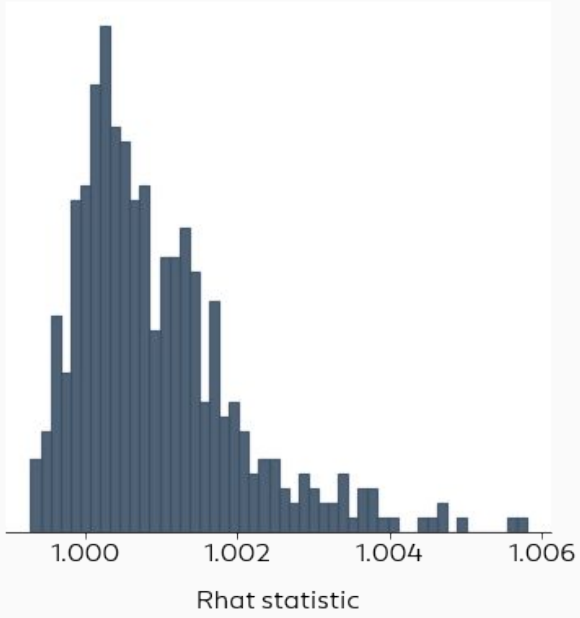
Monte Carlo se / posterior sd smaller 10%?



Effective sample size / iterations greater 10%?



Rhat statistic smaller 1.05?

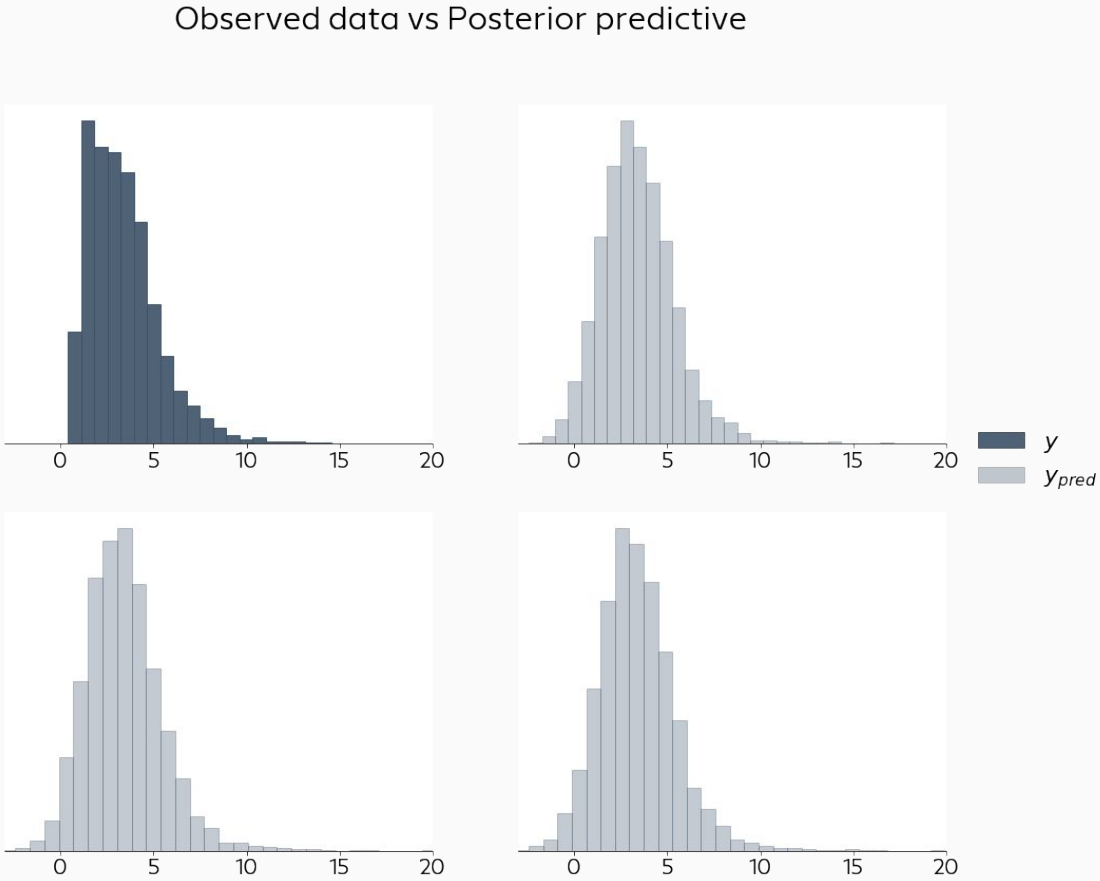


How good does my model fit the data?

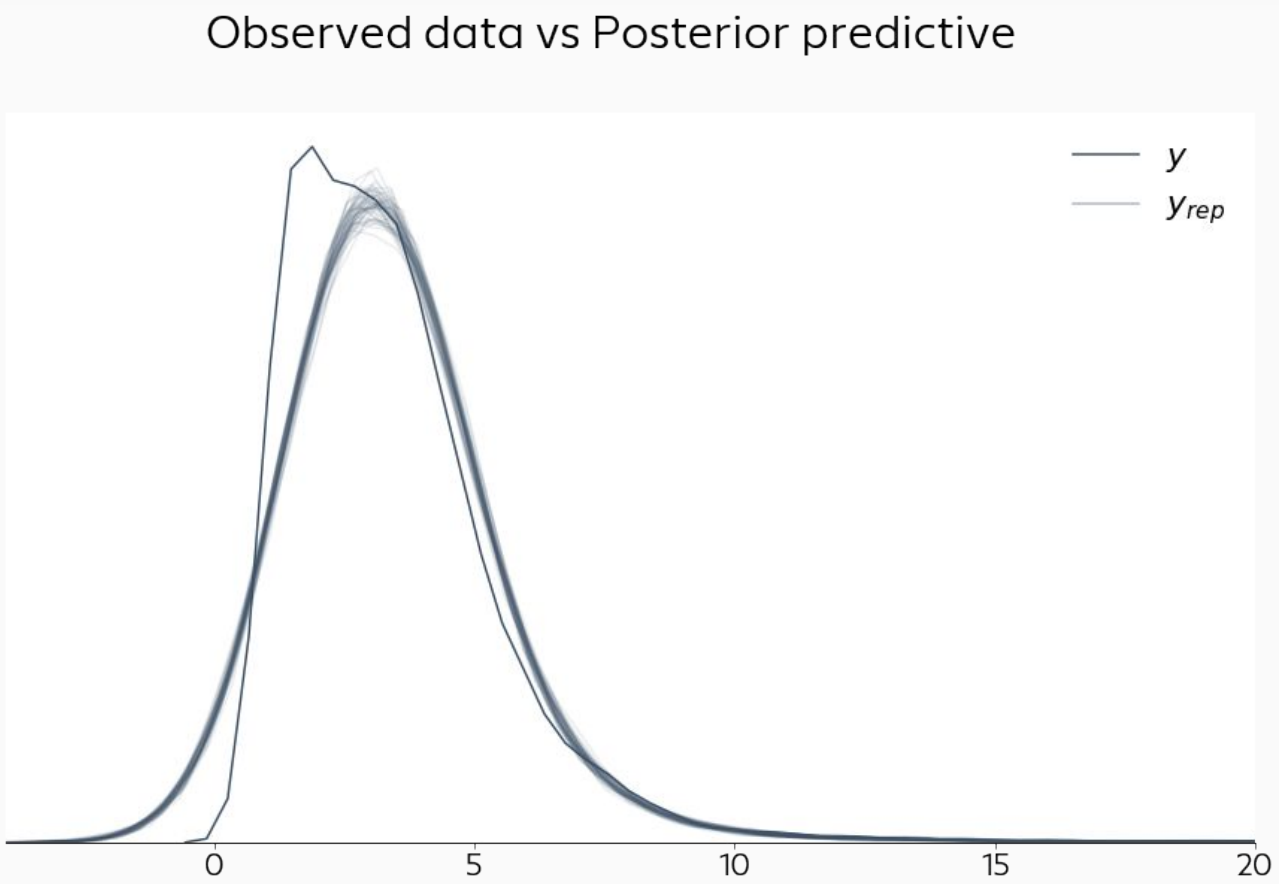
How good does my model fit the data?

```
with hier_model:
    posterior_predictive = pm.sample_posterior_predictive(trace)
```

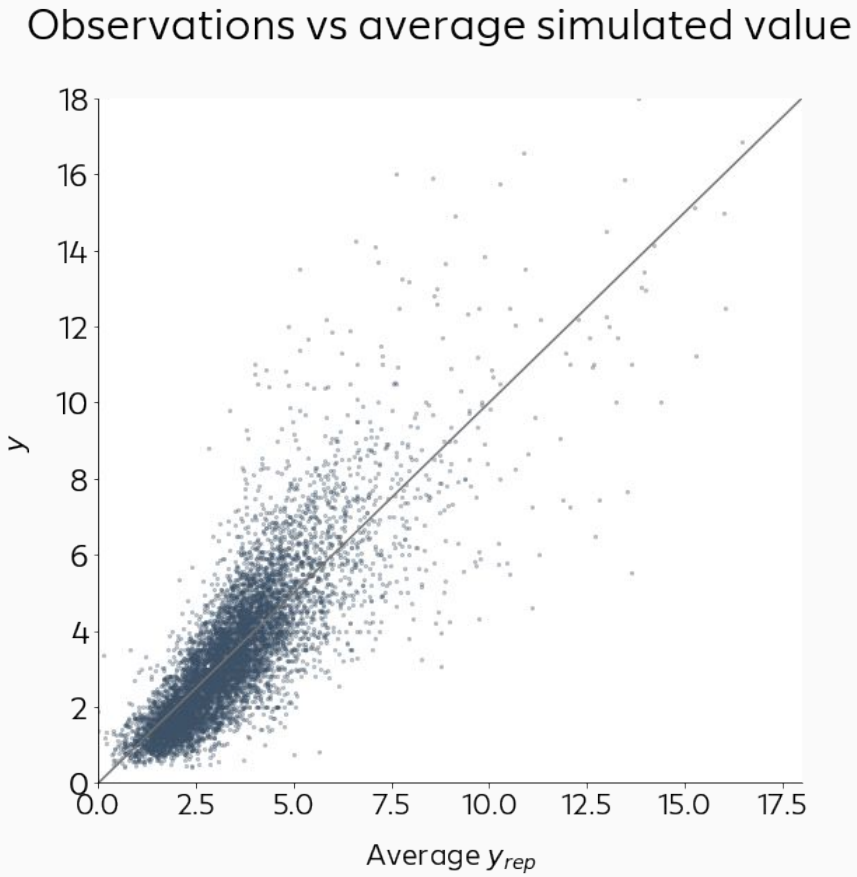
How good does my model fit the data?



How good does my model fit the data?

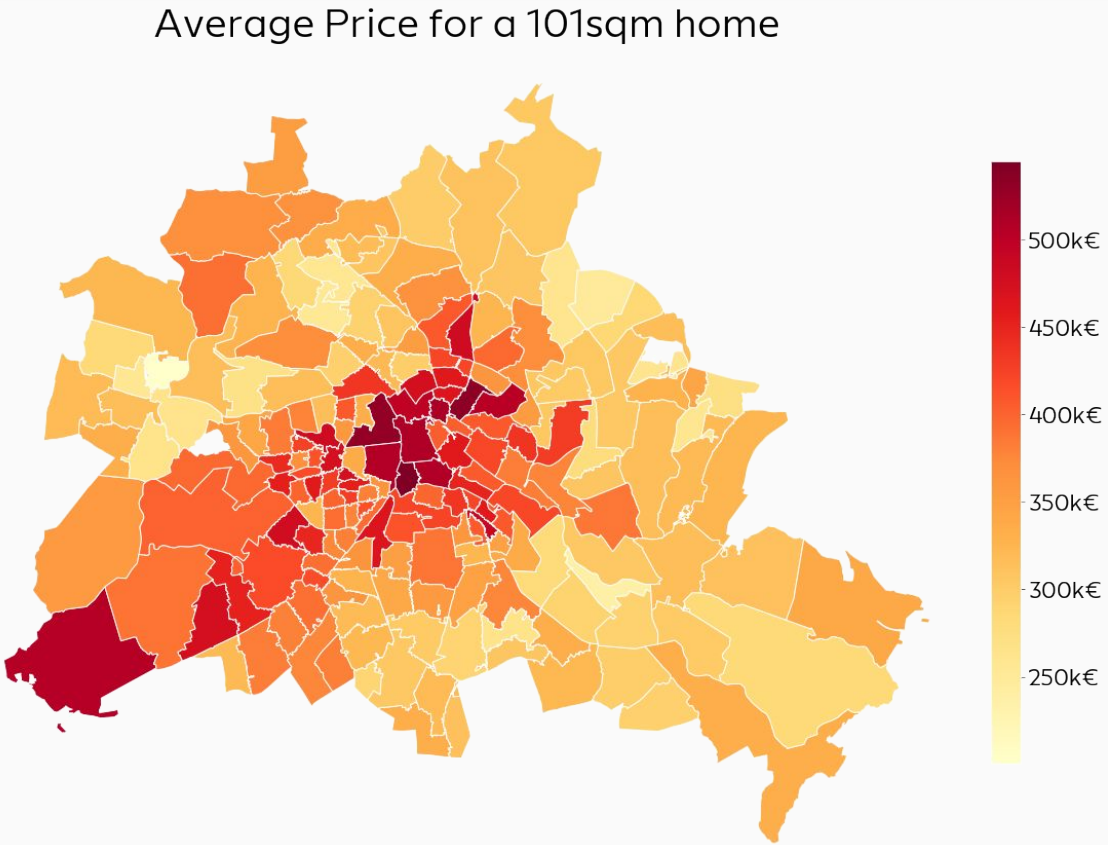


How good does my model fit the data?

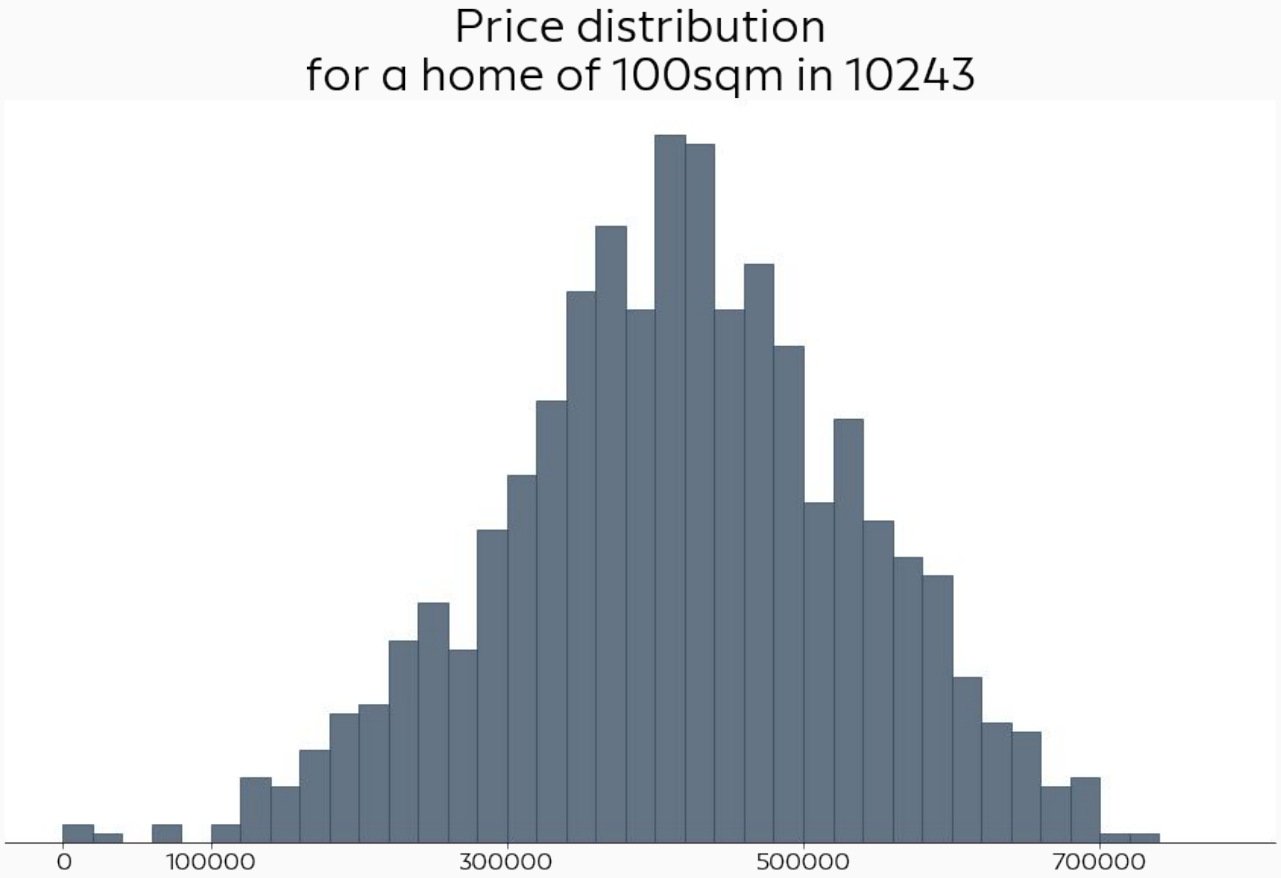


Results, please!

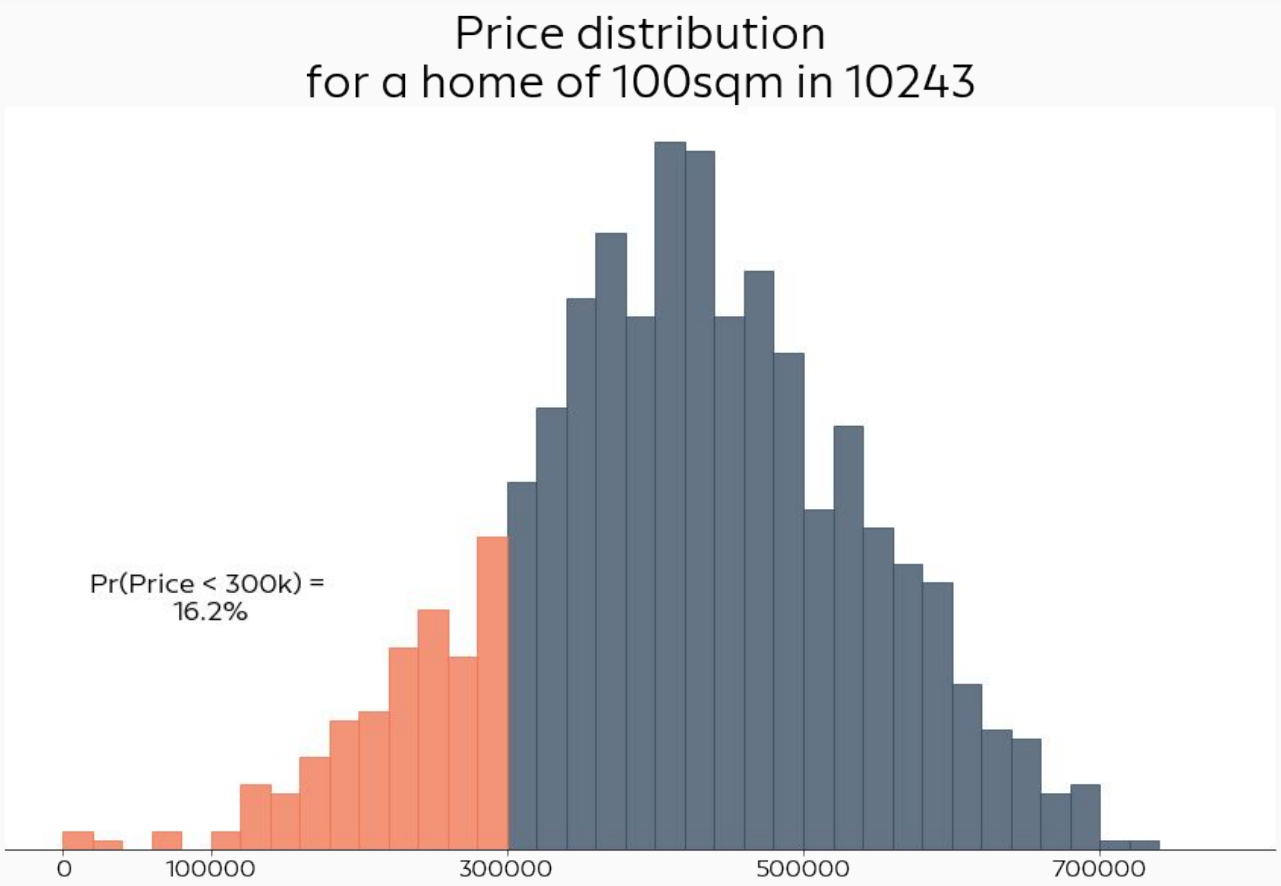
Results, please!



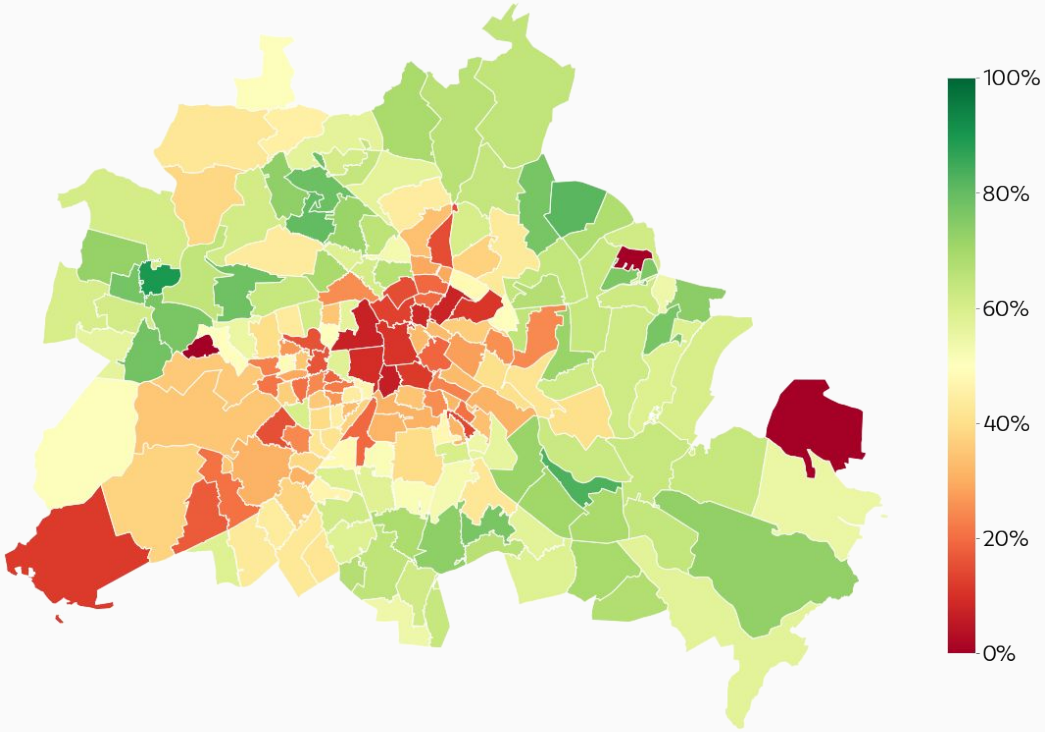
Results, please!



Results, please!



Results, please!



What’s next?

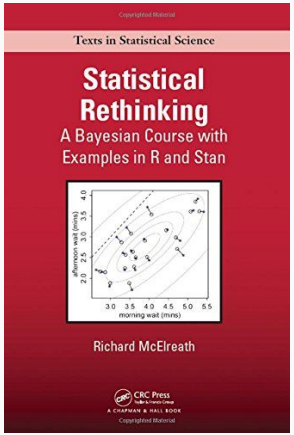
What's next?

- Iterate!
- More predictors!
  - Year of construction
  - House type
  - ...
- More hierarchies!
- Add group predictors!
  - Percentage of green areas
  - Economical indices
- Try different likelihoods
- Probably save more money...

Further resources

Richard McElreath: Statistical Rethinking

- Port to PyMC3



Prior Recommendation by Stan Team

Michael Betancourts Case Studies

BerlinBayesians

Thanks!

@corrieaar

corriebar
Code and Notebooks

www.samples-of-thoughts.com

Icons by icons8