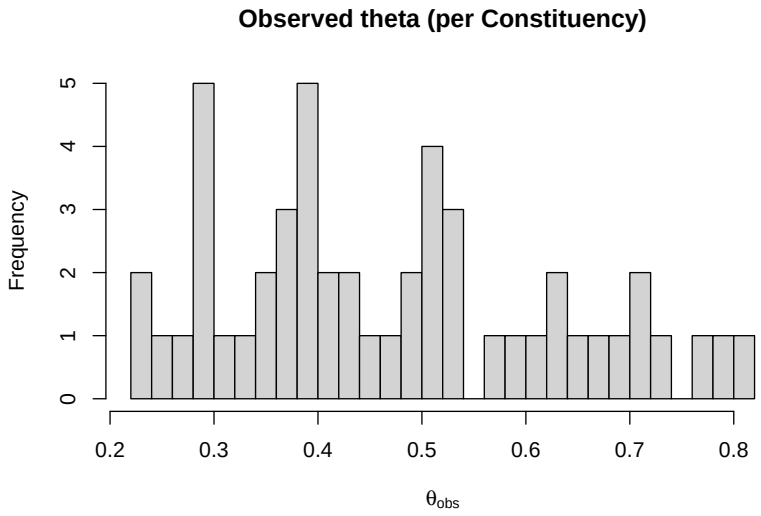
```r
set.seed(123)
library(rstan);library(bayesplot);library(loo)
load(url("https://acaimo.github.io/teaching/data/referendum.RData"))
```

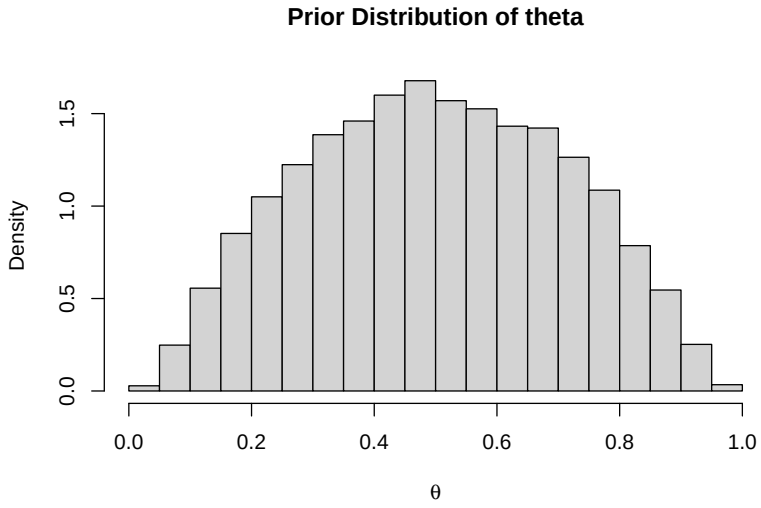
```r
# Define parameters
n <- nrow(referendum)
x_i <- x_ij <- referendum$n_yes
n_votes <- referendum$n_votes
j <- referendum$region
m <- length(table(referendum$region))

# Compute observed theta
observed_theta <- x_i / n_votes

# Plot histogram of observed theta
hist(observed_theta, breaks = 30, main = "Observed theta (per Constituency)",
     xlab = expression(theta[obs]))
```



Upon inspection of the observed probabilities (x_i divided by n_i), there appears to be a great deal of variation, with values ranging roughly from 0.2 to 0.8. The distribution does not seem to follow any standard distribution. It is irregular and has multiple peaks. This suggests that different constituencies behave quite differently when it comes to voting Yes.

Since theta is a probability, it's bounded between 0 and 1. Given how spread out the values are, using a single probability for all constituencies might be too simple. Although modelling with a single parameter may be difficult, we go ahead and use a Normal(0, 1) prior on the parameter mu. This mu is passed through the inverse logit function to calculate theta, our probability of voting Yes. The Normal(0, 1) prior is a common weakly informative choice. It centres theta around 0.5 but still allows for a wide range of values.

```r
# Simulate prior samples
mu_prior <- rnorm(10000, mean = 0, sd = 1)
theta_prior <- plogis(mu_prior)

# Histogram of prior distribution
hist(theta_prior, breaks = 30, main = "Prior Distribution of theta",
     xlab = expression(theta), probability = TRUE)
```



The prior distribution is centred around 0.5, as expected, and spans the full range from 0 to 1 (slightly wider than the observed distribution). Its shape is quite different from that of the observed data, which is no surprise given the irregular nature of the observed probabilities. Since the prior is weakly informative, we expect the posterior distribution to be shaped by the data, although it will most likely still be quite different from the observed distribution.

We specify a pooled Binomial model to estimate the probability of voting Yes in the referendum, assuming all constituencies share the same underlying probability. Let x_i represent the number of Yes votes and n_i the total number of votes in constituency i. We model this x_i following a Binomial distribution with parameters n_i and theta, where theta is the success probability across all constituencies. To ensure that theta lies between 0 and 1, we apply a logit transformation: theta = inv_logit(mu). We place a weakly informative prior, Normal(0, 1), on mu.

```
# Define Stan model code
stan_code1 <-'
data { int<lower=1> n; int<lower=0> x_i[n]; int<lower=0> n_votes[n]; }
parameters { real mu; }
transformed parameters { real<lower=0, upper=1> theta; theta = inv_logit(mu); }
model { mu ~ normal(0, 1);
  for (i in 1:n) {
    x_i[i] ~ binomial(n_votes[i], theta);
  } }
generated quantities { vector[n] log_lik; vector[n] theta_tilde;
  for (i in 1:n) {
    theta_tilde[i] = inv_logit(mu);
    log_lik[i] = binomial_lpmf(x_i[i] | n_votes[i], theta_tilde[i]);
  } }
'

# Compile Stan model
PN_model <- stan_model(model_code = stan_code1)

# Fit model (using previously defined x_i, n_votes, n)
PN_posterior <- sampling(PN_model, iter = 4000, seed = 1,
                         data = list(n = n, x_i = x_i, n_votes = n_votes))
```

```
# Summarise posterior
post_summary <- summary(PN_posterior, probs = c(0.025, 0.975),
                        pars = c('mu', 'theta'))$summary
```

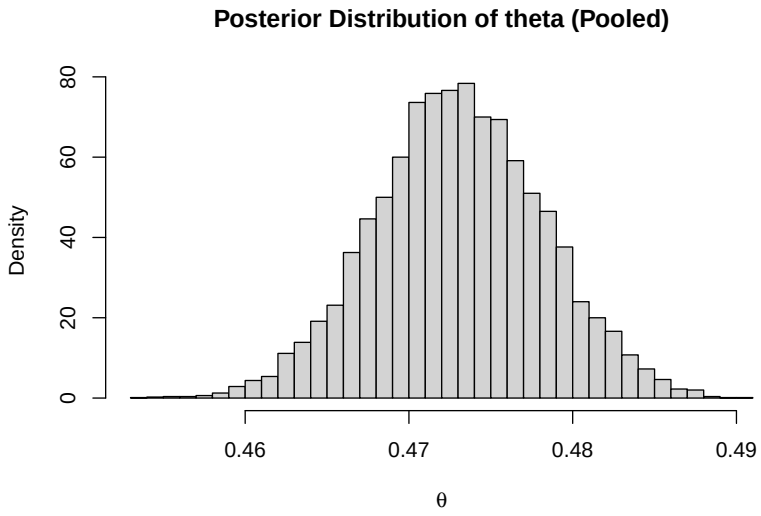
```
round(post_summary, 2)
```

```
##        mean se_mean   sd  2.5% 97.5%   n_eff Rhat
## mu    -0.11       0 0.02 -0.15 -0.07 3007.47    1
## theta  0.47       0 0.01  0.46  0.48 3007.51    1
```

The posterior mean for mu is -0.11, with a 95% credible interval of { -0.15 – -0.07 }. Since mu is transformed via the inverse logit function to compute theta, this suggests that the log-odds of voting Yes across all constituencies is slightly negative on average. The interval lies entirely below 0, indicating moderate evidence that the average support for Yes is below 50% on the log-odds scale.

The posterior mean for theta, the overall probability of voting Yes, is 0.47. The 95% credible interval is { 0.46 – 0.48 }, which is relatively narrow and completely below 0.5. This suggests that, under the pooled model, the probability of voting Yes was estimated to be slightly below even chance. Compared to the prior, which was centred at 0.5, the posterior shows a small shift informed by the data.

```
# Extract posterior samples
PN_posterior_samples <- extract(PN_posterior)

# Get samples for theta
theta_post <- PN_posterior_samples$theta

# Plot histogram of posterior for theta
hist(theta_post, breaks = 30, probability = TRUE,
     main = "Posterior Distribution of theta (Pooled)", xlab = expression(theta))
```



As suspected, the posterior distribution has been shaped by the data and shows some similarity to the observed distribution, however, it is still quite different overall. The posterior is much more concentrated, ranging only from around 0.45 to 0.5, while the observed probabilities range from 0.2 to 0.8. This narrowing reflects the limitations of the pooled model, which assumes all constituencies share a single underlying probability. The model has clearly struggled to capture the true variation in the data which is most likely because the complete pooling approach is too simplistic for this context.

We now relax the assumption that all constituencies share the same probability of voting Yes. Instead, we let each constituency have its own probability. We model the number of Yes votes in each constituency as a Binomial distribution with its own success probability => theta_i.

To keep theta_i between 0 and 1, we write it as the inverse logit of a new parameter mu_i. The mu_i values are drawn from a normal distribution with mean mu and standard deviation sigma. The parameter mu represents the overall average across constituencies, and sigma measures how much the constituencies vary.

We place a Normal(0, 1) prior on mu and an Exponential(1) prior on sigma to keep sigma strictly positive and have some relatively moderate variation.

This model allows us to capture both the overall trend and the variation between constituencies.

```
# Stan code
stan_code2 <- '
data { int<lower=1> n; int<lower=0> x_i[n]; int<lower=0> n_votes[n]; }
parameters { real mu; real<lower=0> sigma; vector[n] mu_i; }
transformed parameters { vector[n] theta; theta = inv_logit(mu_i); }
model { mu ~ normal(0, 1); sigma ~ exponential(1); mu_i ~ normal(mu, sigma);
  x_i ~ binomial(n_votes, theta); }
generated quantities { vector[n] log_lik; vector[n] theta_tilde;
  for (i in 1:n) { theta_tilde[i] = inv_logit(mu_i[i]);
    log_lik[i] = binomial_lpmf(x_i[i] | n_votes[i], theta_tilde[i]);
  } }
'

# Compile Stan model
HN1_model <- stan_model(model_code = stan_code2)

# Fit model (using previously defined x_i, n_votes, n)
HN1_posterior <- sampling(HN1_model, iter = 4000, seed = 1, data = list(n = n,
                                      x_i = x_i, n_votes = n_votes))
```

Since the model estimates 102 parameters, including the global average mu, the standard deviation sigma, and a mu_i and theta_i for each constituency, I will report numerical results for only the key parameters mu and sigma.

```
# Summarise posterior
post_summary <- summary(HN1_posterior, probs = c(0.025, 0.975),
                        pars = c('mu', 'sigma'))$summary
round(post_summary, 2)
```

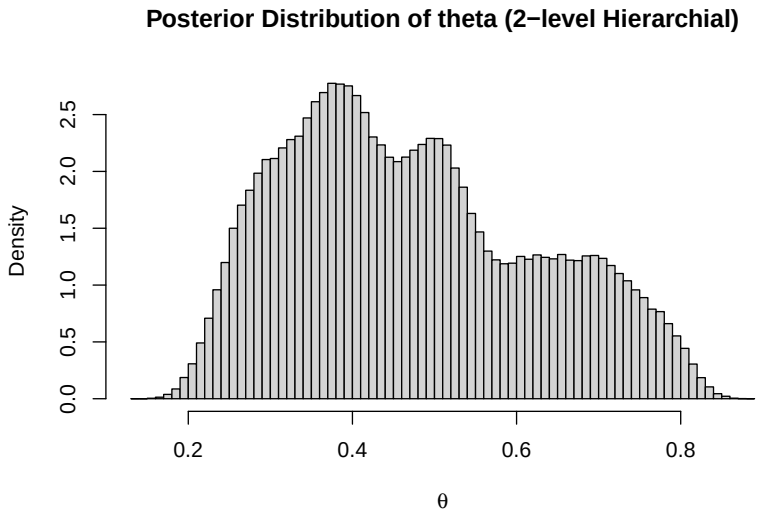
```
##        mean se_mean   sd  2.5% 97.5%    n_eff Rhat
## mu    -0.11       0 0.10 -0.31  0.08 10097.75    1
## sigma  0.67       0 0.07  0.55  0.84  9350.51    1
```

The posterior mean for mu is -0.12, with a 95% credible interval from -0.31 to 0.08. Similar to the pooled model, this suggests that the average log-odds of voting Yes across all constituencies is slightly negative, though the interval contains values close to 0. When transformed via the inverse logit, this corresponds to an average probability of voting Yes just below 0.5 but with greater variation than the last model.

The posterior mean for sigma is 0.68, with a 95% credible interval from 0.55 to 0.84. This is slightly lower than my prior estimate of 1. This suggests substantial variation in the constituency level probabilities, confirming that different constituencies differ meaningfully in their probability of a Yes vote. This variation shows up in the different mu_i values for each constituency and also in the posterior distribution of theta, which is quite spread out and uneven.

```
# Extract posterior samples
HN1_posterior_samples <- extract(HN1_posterior)

# Get samples for theta
theta_post <- HN1_posterior_samples$theta

# Plot histogram of posterior for theta
hist(theta_post, breaks = 100, probability = TRUE,
     main = "Posterior Distribution of theta (2-level Hierarchial)",
```

```
     xlab = expression(theta))
```



The hierarchical model does a noticeably better job at capturing the variation in the data. Compared to the pooled model's posterior for theta, which is very narrow and symmetric (centered around 0.47), the posterior distribution produced by the hierarchical model is wider and more irregular in shape. It spans roughly the same range of values, about 0.2 to 0.8, and more closely resembles the distribution of the observed probabilities.

Each constituency is allowed its own probability of voting Yes, rather than forcing all of them to share a single value. This results in a better overall fit. The multi-peaked and skewed shape of the posterior suggests real differences in voting patterns across constituencies.

```
stan_code3 <- '
data { int<lower=0> n; int<lower=0> m; int<lower=1, upper=m> j[n];
  int<lower=0> x_ij[n]; int<lower=0> n_votes[n]; }
parameters { real mu; real<lower=0> tau; real<lower=0> sigma; real mu_j[m];
  vector[n] eta; }
model { mu ~ normal(0, 1); tau ~ exponential(1); sigma ~ exponential(1);
  mu_j ~ normal(mu, tau); eta ~ normal(0, 1);
  for (i in 1:n) {
    real mu_ij = mu_j[j[i]] + sigma * eta[i];
    x_ij[i] ~ binomial(n_votes[i], inv_logit(mu_ij));
  } }
generated quantities { vector[n] log_lik; vector[n] theta_tilde;
  for (i in 1:n) {
    real mu_ij = mu_j[j[i]] + sigma * eta[i];
    theta_tilde[i] = inv_logit(mu_ij);
    log_lik[i] = binomial_lpmf(x_ij[i] | n_votes[i], theta_tilde[i]);
  } }
'
```

We now extend the previous hierarchical model by adding a third level to capture both within-region and between-region variation in Yes-vote probabilities.

mu represents the overall average log-odds of a Yes vote across all regions and constituencies. It is the global mean.

tau controls the variation between regions => each region j has its own region-level mean mu_j, which is drawn from a Normal distribution centred at mu with standard deviation tau. This allows each region to have a different overall tendency to vote Yes.

mu_j[m] are the region-level means, one for each of the m regions. They represent the average log-odds of voting Yes for that region.

sigma controls the variation within regions i.e. between constituencies in the same region. Each constituency's voting behaviour is allowed to deviate from its region's average via a random offset.

eta[i] represents standardised constituency-level variation, following typical Stan notation. Each eta[i] is drawn from a Normal(0,1) distribution and scaled by sigma. For each constituency i, the log-odds of a Yes vote is calculated as mu_ij = mu_j[j[i]] + sigma * eta[i], combining regional effects (mu_j) with effects from the constituency level (eta[i]).

The transformed value theta[i] = inv_logit(mu_ij) gives the final Yes-vote probability used in the Binomial model for each constituency.

The model allows us to capture: The overall trend (mu) Differences between regions (tau and mu_j) Differences within regions, between constituencies (sigma and eta)

Letting the model account for differences at both the region and constituency level makes it a much better fit for the real variation in voting behaviour.

```
# Compile model
HN2_model <- stan_model(model_code = stan_code3)

# Fit model
HN2_posterior <- sampling(HN2_model, iter = 4000, seed = 1,
              data = list(n = n, m = m, j = j, x_ij = x_ij, n_votes = n_votes))
```

Similar to the last model, as there are 50 estimated parameters for theta_i, I will only report a subset of the numerical results.

```
# Posterior summary
post_summary <- summary(HN2_posterior, probs = c(0.025, 0.975),
                        pars = c('mu', 'tau', 'sigma', 'mu_j'))$summary
round(post_summary, 2)
```

```
##          mean se_mean   sd  2.5% 97.5%   n_eff Rhat
## mu      -0.10       0 0.34 -0.80  0.58 8847.69    1
## tau      0.81       0 0.33  0.40  1.66 6991.84    1
## sigma    0.27       0 0.04  0.20  0.36 3144.86    1
## mu_j[1] -0.53       0 0.10 -0.72 -0.33 4565.94    1
## mu_j[2]  0.04       0 0.10 -0.16  0.22 4379.19    1
## mu_j[3]  0.90       0 0.10  0.70  1.09 4453.32    1
## mu_j[4] -0.85       0 0.10 -1.05 -0.65 4507.93    1
## mu_j[5] -0.14       0 0.10 -0.33  0.06 4380.97    1
```

The posterior mean for mu is -0.10, with a 95% credible interval of {-0.79 - 0.63}. This represents the average log-odds of voting Yes across all constituencies and regions. The mean is similar to the previous models' estimates but with a far wider credible interval suggesting a greater deal of variation in the estimate.

The posterior mean for tau is 0.81, with a 95% credible interval of {0.40 - 1.68}. This reflects the variation between regions, indicating that regional differences in the Yes vote are significant.

The posterior mean for sigma is 0.27 with a 95% credible interval of {0.20 - 0.35}, which measures variation within regions, between constituencies. This is lower than tau, suggesting more consistency within regions compared to between them.

The mu_j parameters represent the average log-odds of voting Yes in each region. => • mu_j[1] has a mean of -0.53 with a 95% credible interval of {-0.73 – -0.33}, indicating below average support for Yes in that region.

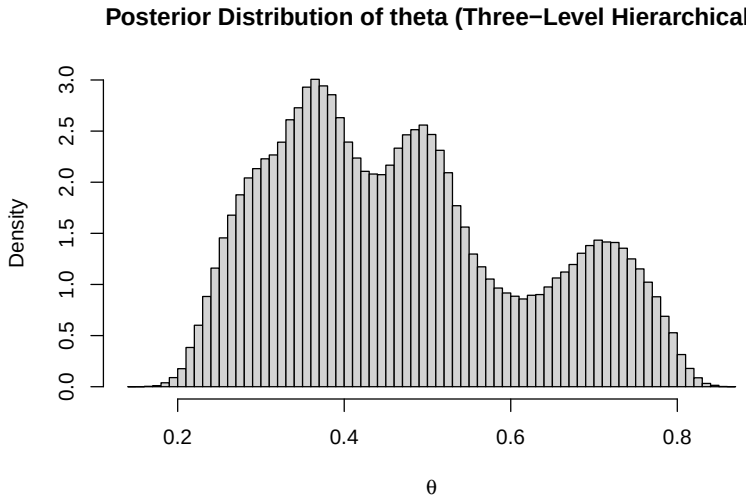
- mu_j[2] has a mean of 0.04 with a 95% credible interval of {-0.15 – 0.23}, suggesting neutral support.

• mu_j[3] has a mean of 0.90 with a 95% credible interval of {0.70 – 1.10}, indicating strong support for Yes.

• mu_j[4] has a mean of -0.85 with a 95% credible interval of {-1.04 – -0.65}, indicating strong opposition to Yes.

• mu_j[5] has a mean of -0.13 with a 95% credible interval of {-0.33 – 0.06}, suggesting weak or neutral support.

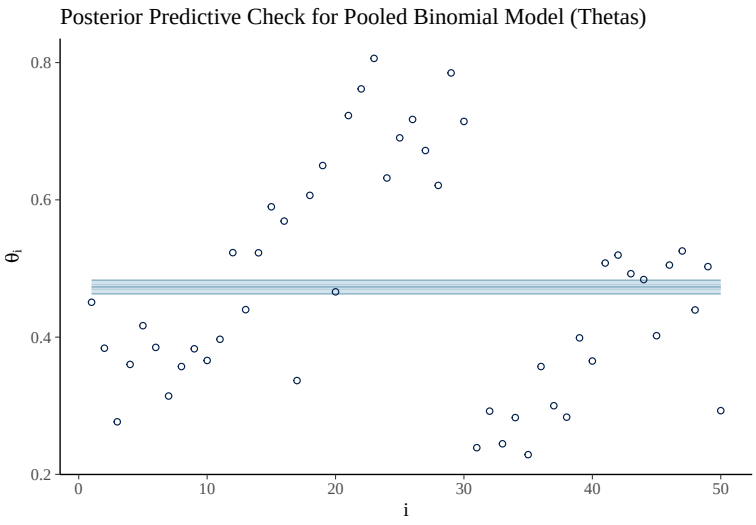
```
# Plot posterior of Theta
HN2_posterior_samples <- extract(HN2_posterior)

theta_post <- HN2_posterior_samples$theta

hist(as.vector(theta_post), breaks = 100, probability = TRUE,
     main = "Posterior Distribution of theta (Three-Level Hierarchical)",
     xlab = expression(theta))
```



As shown by the histogram of the posterior distribution, the three-level hierarchical model improves upon the previous two-level model. The peaks are more defined, and the shape of the distribution now more closely resembles the observed data. The simulated range of values (0.2 - 0.8) is nearly identical to the observed range, indicating that the model is capturing the variation more accurately.

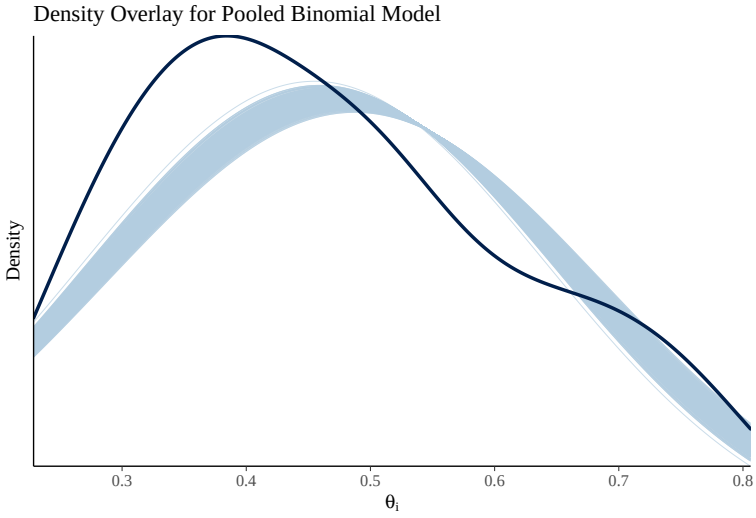
By incorporating region-level effects, the three-level model does a better job of identifying differences in voter behaviour. This is supported by both the numerical estimates and the visual results, which highlight the significance of regional variation.

```
# Pooled Binomial

# Extract simulated probabilities (theta_tilde) from the posterior
theta_tilde <- extract(PN_posterior)$theta_tilde

# Convert theta_tilde to a matrix [draws × observations]
theta_tilde_matrix <- as.matrix(PN_posterior_samples$theta_tilde)

# Posterior Predictive Check for Pooled Binomial Model (using theta_tilde)
ppc_ribbon(x_i / n_votes, theta_tilde_matrix[1:1000, ], prob_outer = 0.95,
           alpha = 0.9, y_draw = 'points') + ggplot2::xlab('i') +
  ggplot2::ylab(expression(paste(theta[i]))) +  # Change the y-axis label to 'theta[i]'
  ggplot2::ggtitle("Posterior Predictive Check for Pooled Binomial Model (Thetas)") +
  legend_none()
```



The plot shows the Posterior Predictive Check for the Pooled Binomial Model (Thetas). The majority of the observed values (white points) fall outside the 95% credible interval (blue region). This suggests that the pooled model does not adequately capture the variation in the observed data, as the predicted probabilities (theta) don't align well with the observed values. This mismatch indicates the need for a more flexible model, such as one that accounts for within-group variation.

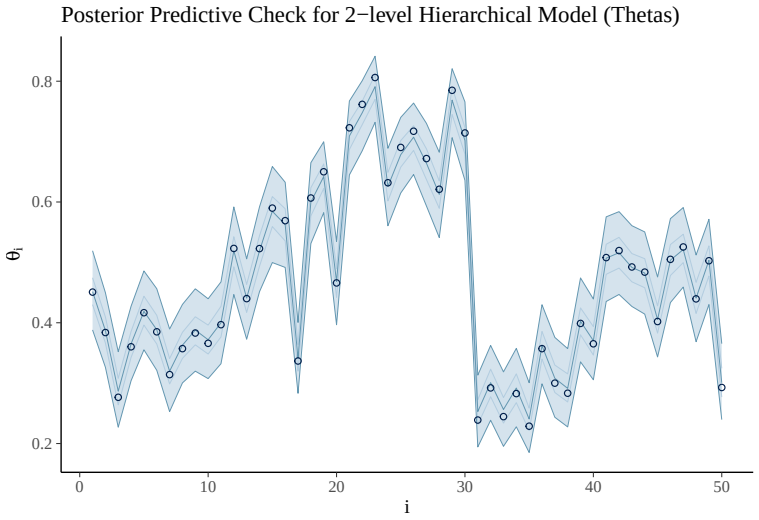
```
# Density plot for theta_tilde, Using observed probabilities x_i / n_votes
ppc_dens_overlay(x_i / n_votes, theta_tilde[1:1000, ]) +
  ggplot2::xlab(expression(paste(theta[i]))) + ggplot2::ylab('Density') +
  ggplot2::ggtitle("Density Overlay for Pooled Binomial Model") + legend_none()
```



The density overlay for the pooled binomial model backs up the previous observation. The simulated distribution (light blue) doesn't quite match the complexity of the observed distribution (dark blue). The model is too narrow and assumes a simpler shape. This suggests that the pooled model, which assumes a single probability for all constituencies, isn't capturing the variation in the data. It again points to the need for a more flexible model that accounts for differences both within and between groups. Overall this indicates very poor fit.

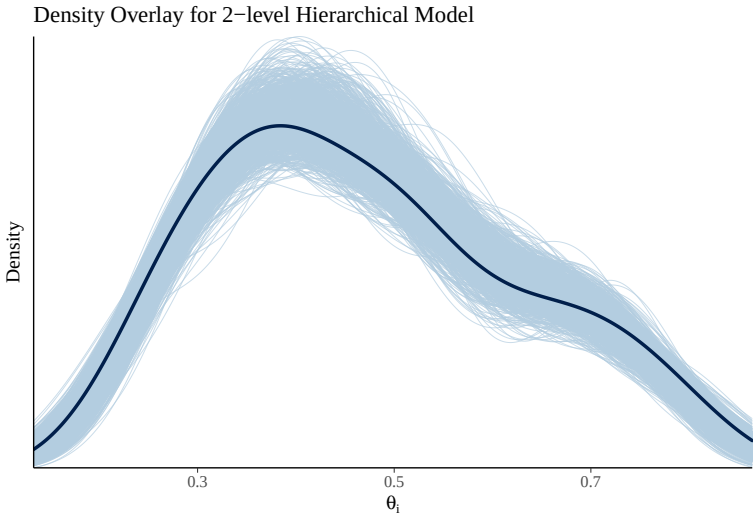
```
# Hierarchical model 1

# Extract simulated probabilities (theta_tilde) from the posterior
theta_tilde <- extract(HN1_posterior)$theta_tilde
```

```r
# Convert theta_tilde to a matrix [draws × observations]
theta_tilde_matrix <- as.matrix(HN1_posterior_samples$theta_tilde)

# Posterior Predictive Check for Simulated Probabilities (using theta_tilde)
ppc_ribbon(x_i / n_votes, theta_tilde_matrix[1:1000, ], prob_outer = 0.95,
           alpha = 0.9, y_draw = 'points') + ggplot2::xlab('i') +
  ggplot2::ylab(expression(paste(theta[i]))) +
  ggplot2::ggtitle("Posterior Predictive Check for 2-level Hierarchical Model (Thetas)") +
  legend_none()
```



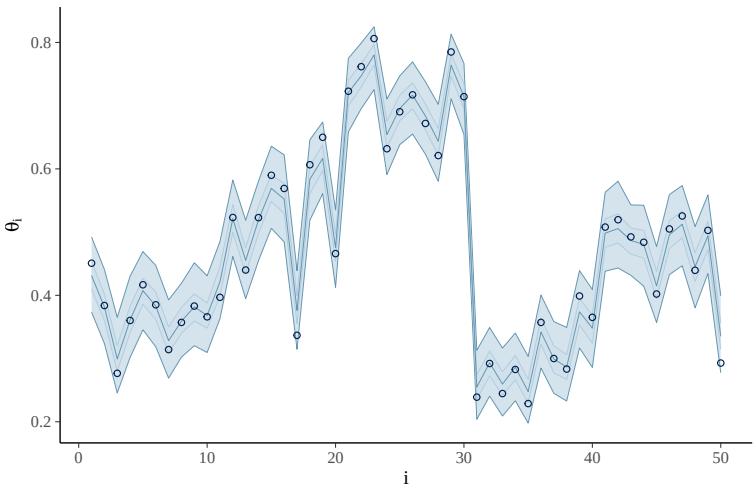
The 2-level hierarchical model does a much better job at capturing the variation in the data compared to the pooled model. The shaded regions show the 95% credible intervals for the predicted probabilities, which vary across constituencies. Most of the observed values (dots) fall within the 95% credible intervals, indicating a good fit. The predicted values (dark line) align closely with the observed data. By accounting for both within- and between-group variation, the hierarchical model captures the data's complexity, offering a more flexible and accurate fit than the simpler pooled model.

```r
# Density plot for simulated probabilities (theta_tilde)
ppc_dens_overlay(x_i / n_votes, theta_tilde[1:1000, ]) +  # Using observed probabilities x_i / n_votes
  ggplot2::xlab(expression(paste(theta[i]))) + ggplot2::ylab('Density') +
  ggplot2::ggtitle("Density Overlay for 2-level Hierarchical Model") +
  legend_none()
```



This density overlay for the 2-level hierarchical model shows a significant improvement from the previous model. The predicted distribution (dark line) aligns closely with the observed distribution (light blue), suggesting a better fit. Unlike the pooled model, which had a narrow and oversimplified distribution, the hierarchical model captures more of the data's variability and resembles the shape of the observed distribution. Overall, this indicates that the 2-level hierarchical model has provided reasonble fit to the data.

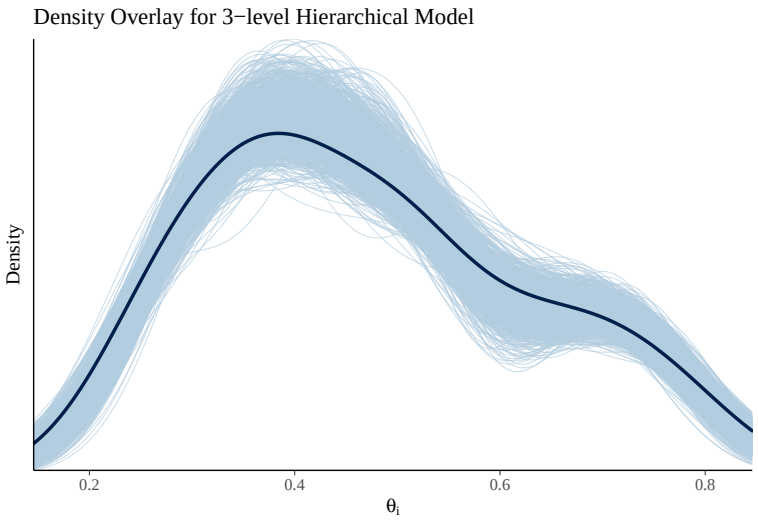
```
# Hierarchical model 2

# Extract simulated probabilities (theta_tilde) from the posterior
theta_tilde <- extract(HN2_posterior)$theta_tilde

# Convert theta_tilde to a matrix [draws × observations]
theta_tilde_matrix <- as.matrix(HN2_posterior_samples$theta_tilde)

# Posterior Predictive Check for Simulated Probabilities (using theta_tilde)
ppc_ribbon(x_ij / n_votes, theta_tilde_matrix[1:1000, ], prob_outer = 0.95,
           alpha = 0.9, y_draw = 'points') + ggplot2::xlab('i') +
  ggplot2::ylab(expression(paste(theta[i]))) +  # Change the y-axis label to 'theta[i]'
  ggplot2::ggtitle("Posterior Predictive Check for 3-level Hierarchical Model (Thetas)") +
  legend_none()
```



The 3-level hierarchical model captures the variation in the data reasonably well, as most of the observed

values (dots) fall within the 95% credible intervals, indicating a good fit. However, the results are very similar to the 2-level model, making it debatable whether the added complexity of the 3-level model significantly improves the fit.

```
# Density plot for simulated probabilities (theta_tilde)
ppc_dens_overlay(x_ij / n_votes, theta_tilde[1:1000, ]) +
  ggplot2::xlab(expression(paste(theta[i]))) + ggplot2::ylab('Density') +
  ggplot2::ggtitle("Density Overlay for 3-level Hierarchical Model") +
  legend_none()
```



Similar to the 2-level hierarchical model, the 3-level model replicates the observed distribution well. However, the results are quite similar, and it is worth investigating whether any significant improvement was made. Overall, this indicates a good fit.

In order to improve Pareto k diagnostic values, I will set moment_match = TRUE.

```
# Print only the important LOO values
PN_loo <- loo(PN_posterior, moment_match = TRUE)
HN1_loo <- loo(HN1_posterior, moment_match = TRUE)
```

```
## Warning: Some Pareto k diagnostic values are too high. See help('pareto-k-diagnostic') for details.
```

```
HN2_loo <- loo(HN2_posterior, moment_match = TRUE)

print(cbind(round(PN_loo$estimates[c("elpd_loo", "looic"),], 1)))
```

```
##          Estimate    SE
## elpd_loo   -624.5  76.7
## looic      1249.0 153.4
```

```
print(cbind(round(HN1_loo$estimates[c("elpd_loo", "looic"),], 1)))
```

```
##          Estimate  SE
## elpd_loo   -210.8 2.4
## looic       421.7 4.9
```

```
print(cbind(round(HN2_loo$estimates[c("elpd_loo", "looic"),], 1)))
```

```
##          Estimate  SE
## elpd_loo   -196.2 4.7
## looic       392.3 9.3
```

```r
loo_compare(list('Pooled model' = PN_loo,'2-level Hierarchical model' = HN1_loo,
                 '3-level Hierarchical model' = HN2_loo))
```

```
##                             elpd_diff se_diff
## 3-level Hierarchical model    0.0       0.0
## 2-level Hierarchical model  -14.7       5.3
## Pooled model                -428.4      76.4
```

Note: elpd_loo measures predictive accuracy, with higher values indicating better performance. looic penalises complexity, so lower values are better.

=> Pooled Binomial Model (PN): This model performs the worst, with an elpd_loo of -624.5 and a looic of 1249.0. It assumes a single shared probability across all constituencies, failing to capture within-group variation, leading to poor predictive accuracy. The elpd_diff of -428.4 highlights the significant difference compared to the hierarchical models.

2-Level Hierarchical Model (HN1): This model is an improvement from the pooled model with an elpd_loo of -215.6 and a looic of 431.2. By including constituency-level variability, it fits the observed data better. The elpd_diff of -19.5 suggests a relatively minor difference in performance compared to the 3-level model, with a se_diff of 5.3, indicating some uncertainty in the comparison.

3-Level Hierarchical Model (HN2): The 3-level model further refines the fit with an elpd_loo of -196.1 and a looic of 392.2. The improvement over the 2-level model is relatively small, with a difference of -19.5, and a se_diff of 5.3 suggests a small level of uncertainty in the improvement.

Conclusion: The Pooled Model performs the worst due to its oversimplification. The 2-level Hierarchical Model provides a reasonable fit by accounting for constituency-level variability. While the 3-level Model yields slightly better statistics, it introduces more complexity without a significant improvement over the 2-level model, suggesting that the 2-level model captures most of the data's variability. Although the 3-level model has the best statistics, there is a trade-off between its ability to generalise to new data and the risk of overfitting. Therefore, in my opinion the 2-level model is likely the most best choice.