BAYESIAN ANALYSIS OF WINE RATINGS

2024-12-08

LOAD IN + UNDERSTANDING THE DATA

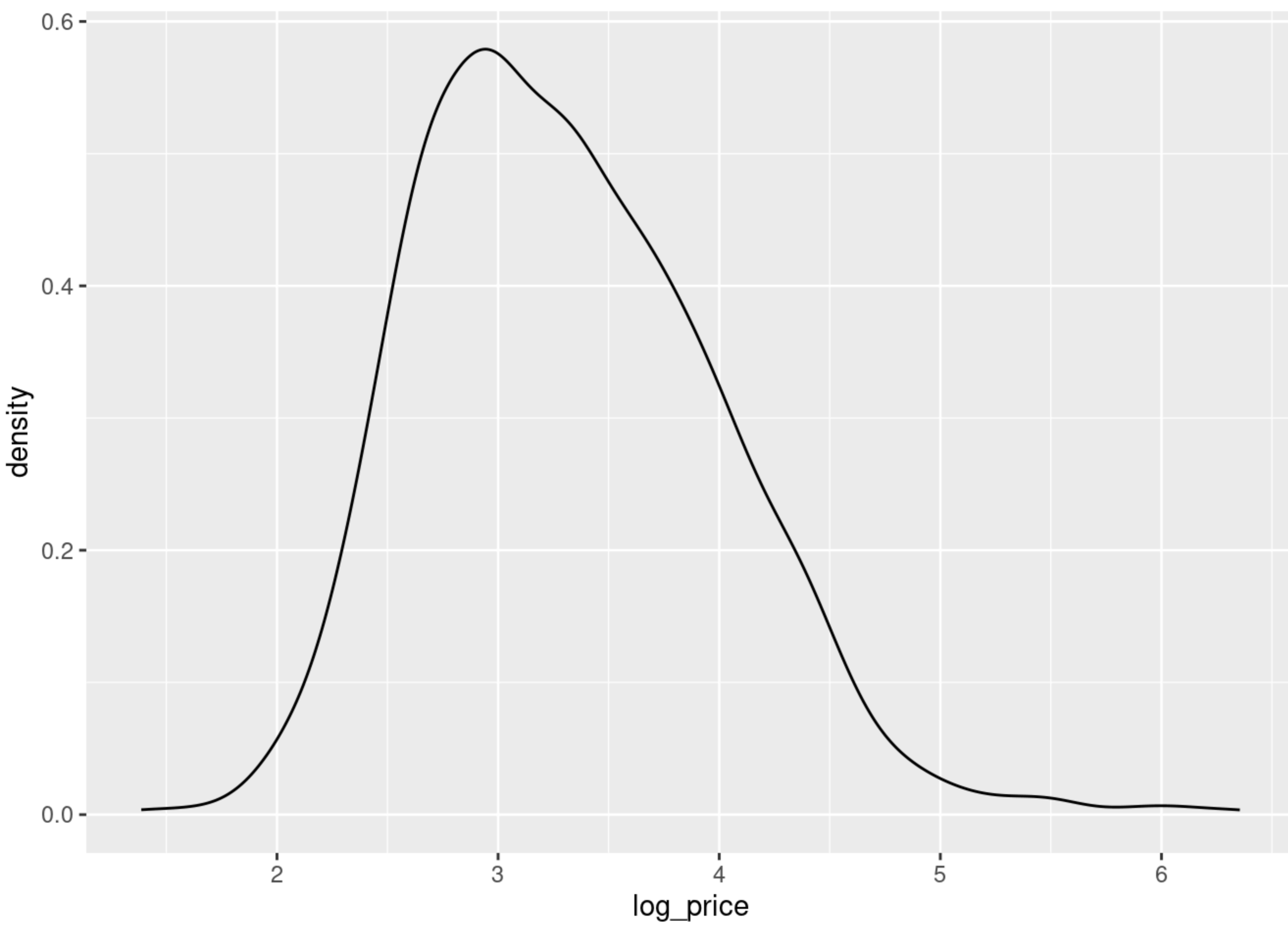
```
set.seed(36501)
wine_data<-drop_na(read.csv("winemag-data-130k-v2.csv"))%>%
  ## ONLY USE DATA FROM COUNTRIES WITH OVER 400 DISTINCT REVIEWS
  sample_n(1000)%>%
  filter(country=="Argentina"|country=="Australia"|country=="Austria"|country=="Chile"|country=
="France"|country=="Germany"|country=="Italy"|country=="New Zealand"|country=="Portugal"|country
=="South Africa"|country=="Spain"|country=="US")
wine_data <- wine_data %>%
  mutate(log_price = log(price))
glimpse(wine_data)
```

```
## Rows: 972
## Columns: 15
## $ X                     <int> 48954, 22922, 83248, 12855, 99833, 98673, 42224,…
## $ country               <chr> "Portugal", "US", "US", "Australia", "US", "US",…
## $ description           <chr> "This is a finely perfumed wine, full of ripe Ca…
## $ designation           <chr> "Pegos Claros Reserva", "", "", "Phoenix", "West…
## $ points                <int> 92, 88, 88, 91, 89, 82, 88, 81, 84, 90, 88, 89, …
## $ price                 <dbl> 17, 36, 14, 20, 19, 14, 23, 8, 28, 54, 13, 38, 4…
## $ province              <chr> "Palmela", "Oregon", "Oregon", "South Australia"…
## $ region_1              <chr> "", "Willamette Valley", "Willamette Valley", "C…
## $ region_2              <chr> "", "Willamette Valley", "Willamette Valley", ""…
## $ taster_name           <chr> "Roger Voss", "Paul Gregutt", "Paul Gregutt", "J…
## $ taster_twitter_handle <chr> "@vossroger", "@paulgwine ", "@paulgwine ", "@Jo…
## $ title                 <chr> "Wines & Winemakers 2011 Pegos Claros Reserva Ca…
## $ variety               <chr> "Castelão", "Pinot Noir", "Riesling", "Cabernet …
## $ winery                <chr> "Wines & Winemakers", "Trisaetum", "Willamette V…
## $ log_price             <dbl> 2.833213, 3.583519, 2.639057, 2.995732, 2.944439…
```

```
tabyl(wine_data$country)
```

```
##  wine_data$country   n     percent
##         Argentina  32 0.032921811
##         Australia  15 0.015432099
##           Austria  21 0.021604938
##             Chile  45 0.046296296
##            France 164 0.168724280
##           Germany  21 0.021604938
##             Italy 133 0.136831276
##       New Zealand   9 0.009259259
##          Portugal  44 0.045267490
##      South Africa   9 0.009259259
##             Spain  47 0.048353909
##                US 432 0.444444444
```

```
ggplot(wine_data,aes(x=log_price))+geom_density()
```




```
mean(wine_data$log_price)
```

```
## [1] 3.313282
```

```
sd(wine_data$log_price)
```

```
## [1] 0.6791097
```

```
wine_data%>%
  group_by(country)%>%
  summarise(mean_points = mean(points, na.rm= TRUE),
            mean_log_price = mean(log_price, na.rm=TRUE),
            n= n())
```

```
## # A tibble: 12 × 4
##    country      mean_points mean_log_price     n
##    <chr>              <dbl>          <dbl> <int>
##  1 Argentina           86.1           2.98    32
##  2 Australia           88.7           3.19    15
##  3 Austria             89.6           3.13    21
##  4 Chile               86.5           2.76    45
##  5 France              89.3           3.46   164
##  6 Germany             90.5           3.44    21
##  7 Italy               88.8           3.40   133
##  8 New Zealand         88.4           3.13     9
##  9 Portugal            88.3           2.93    44
## 10 South Africa        88.2           3.31     9
## 11 Spain               86.9           2.96    47
## 12 US                  88.4           3.40   432
```

BAYESIAN ESTIMATION MODEL

BAYESIAN REGRESSION MODEL

```
wine_reg_model<- stan_glmer(
  price ~ points + (points | country),
  data = wine_data, family = gaussian,
  #we know that in this data, the global mean price is around 35, but can be between 22 and 48
  prior_intercept = normal(35,6.5),
  #we believe that a typical country's wine's price will on average be higher when it has a high
er rating, yet we're unsure of the exact rate. We'll say its likely between 1$-5$ per point, how
ever it really could be anything
  prior = normal(3, 1, autoscale =TRUE),
  #we have no idea of the variability between countries or how a country's point ratings might f
luctuate from their trend
  prior_aux = exponential(1, autoscale = TRUE),
  prior_covariance = decov(regularization = 1, concentration = 1, shape = 1, scale = 1),
  chains = 4, iter = 5000*2, seed= 36501)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.000183 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.83 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 411.795 seconds (Warm-up)
## Chain 1:                245.016 seconds (Sampling)
## Chain 1:                656.811 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.0001 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 2: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 2: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 2: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 2: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 2: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 2: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 2: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 2: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 2: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 2: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 2: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 327.727 seconds (Warm-up)
## Chain 2:                337.567 seconds (Sampling)
## Chain 2:                665.294 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
```

```
## Chain 3:
## Chain 3: Gradient evaluation took 0.0001 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 3: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 3: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 3: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 3: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 3: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 3: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 3: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 3: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 3: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 3: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 3: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 393.687 seconds (Warm-up)
## Chain 3:                239.748 seconds (Sampling)
## Chain 3:                633.435 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0.000119 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.19 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 10000 [  0%]  (Warmup)
## Chain 4: Iteration: 1000 / 10000 [ 10%]  (Warmup)
## Chain 4: Iteration: 2000 / 10000 [ 20%]  (Warmup)
## Chain 4: Iteration: 3000 / 10000 [ 30%]  (Warmup)
## Chain 4: Iteration: 4000 / 10000 [ 40%]  (Warmup)
## Chain 4: Iteration: 5000 / 10000 [ 50%]  (Warmup)
## Chain 4: Iteration: 5001 / 10000 [ 50%]  (Sampling)
## Chain 4: Iteration: 6000 / 10000 [ 60%]  (Sampling)
## Chain 4: Iteration: 7000 / 10000 [ 70%]  (Sampling)
## Chain 4: Iteration: 8000 / 10000 [ 80%]  (Sampling)
## Chain 4: Iteration: 9000 / 10000 [ 90%]  (Sampling)
## Chain 4: Iteration: 10000 / 10000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 333.504 seconds (Warm-up)
## Chain 4:                297.088 seconds (Sampling)
## Chain 4:                630.592 seconds (Total)
## Chain 4:
```

DIAGNOSTICS

```
neff_ratio(wine_reg_model)
```

```
##                            (Intercept)                                 points 
##                                0.30050                                0.29950 
##       b[(Intercept) country:Argentina]           b[points country:Argentina] 
##                                0.66650                                0.67400 
##       b[(Intercept) country:Australia]           b[points country:Australia] 
##                                0.61355                                0.61650 
##         b[(Intercept) country:Austria]             b[points country:Austria] 
##                                0.72895                                0.73900 
##           b[(Intercept) country:Chile]               b[points country:Chile] 
##                                0.68940                                0.68965 
##          b[(Intercept) country:France]              b[points country:France] 
##                                0.40045                                0.39650 
##         b[(Intercept) country:Germany]             b[points country:Germany] 
##                                0.66285                                0.66370 
##           b[(Intercept) country:Italy]               b[points country:Italy] 
##                                0.44205                                0.43975 
##     b[(Intercept) country:New_Zealand]         b[points country:New_Zealand] 
##                                1.03855                                1.04565 
##        b[(Intercept) country:Portugal]            b[points country:Portugal] 
##                                0.72290                                0.72290 
##    b[(Intercept) country:South_Africa]        b[points country:South_Africa] 
##                                1.06350                                1.06915 
##           b[(Intercept) country:Spain]               b[points country:Spain] 
##                                0.59200                                0.59625 
##              b[(Intercept) country:US]                  b[points country:US] 
##                                0.34460                                0.34310 
##                                  sigma Sigma[country:(Intercept),(Intercept)] 
##                                1.02380                                0.42940 
##      Sigma[country:points,(Intercept)]           Sigma[country:points,points] 
##                                0.42780                                0.42660
```

```
rhat(wine_reg_model)
```

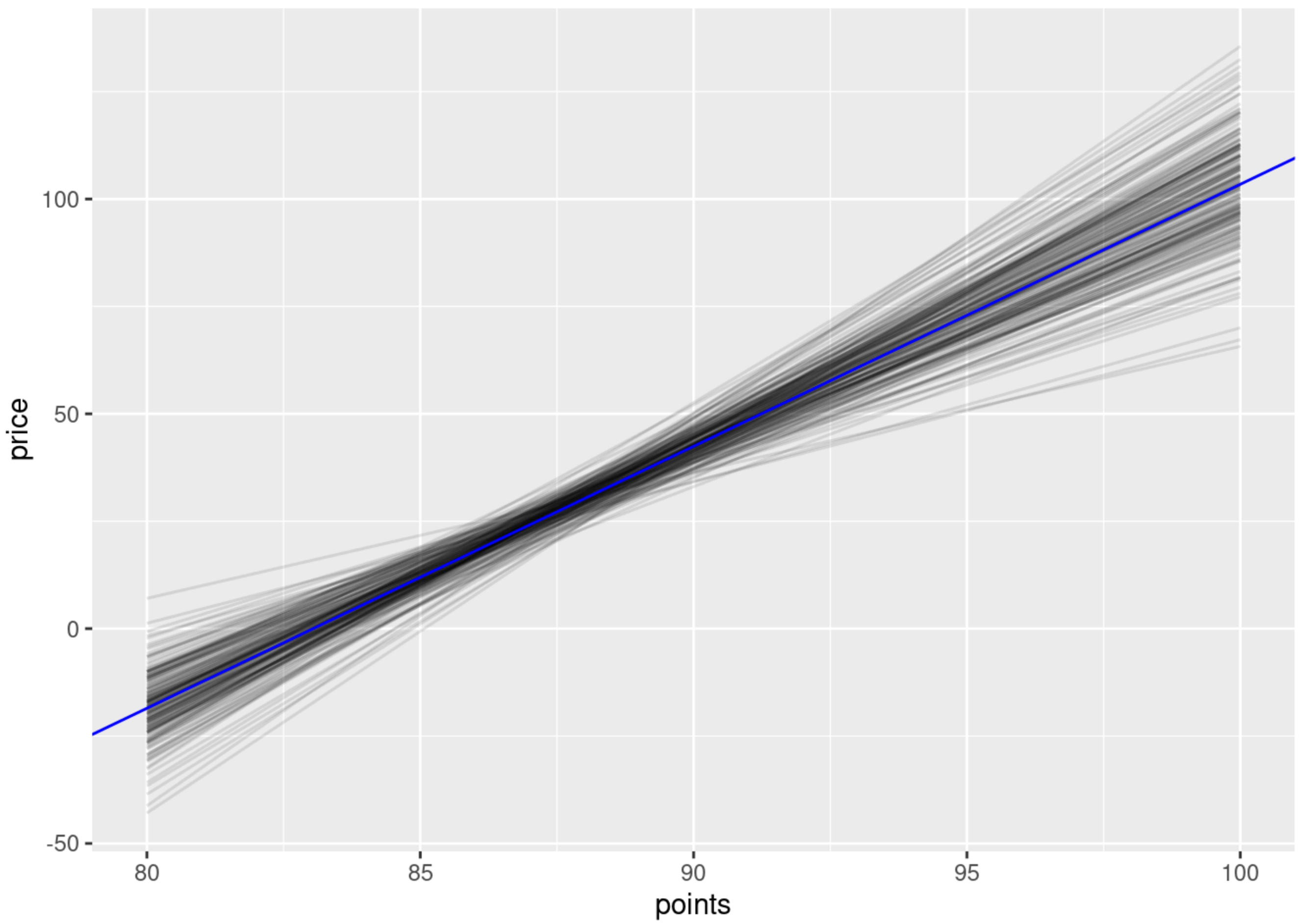
```
##                          (Intercept)                               points 
##                            1.0008012                            1.0008084 
##      b[(Intercept) country:Argentina]          b[points country:Argentina] 
##                            1.0000986                            1.0000983 
##      b[(Intercept) country:Australia]          b[points country:Australia] 
##                            1.0006402                            1.0006334 
##        b[(Intercept) country:Austria]            b[points country:Austria] 
##                            1.0001205                            1.0001172 
##          b[(Intercept) country:Chile]              b[points country:Chile] 
##                            1.0002377                            1.0002365 
##         b[(Intercept) country:France]             b[points country:France] 
##                            1.0009156                            1.0009183 
##        b[(Intercept) country:Germany]            b[points country:Germany] 
##                            0.9999520                            0.9999478 
##          b[(Intercept) country:Italy]              b[points country:Italy] 
##                            1.0001172                            1.0001284 
##    b[(Intercept) country:New_Zealand]        b[points country:New_Zealand] 
##                            1.0000847                            1.0000709 
##       b[(Intercept) country:Portugal]           b[points country:Portugal] 
##                            1.0000758                            1.0000827 
##   b[(Intercept) country:South_Africa]       b[points country:South_Africa] 
##                            1.0001118                            1.0001042 
##          b[(Intercept) country:Spain]              b[points country:Spain] 
##                            1.0004439                            1.0004295 
##             b[(Intercept) country:US]                 b[points country:US] 
##                            1.0006993                            1.0007044 
##                                sigma Sigma[country:(Intercept),(Intercept)] 
##                            1.0002660                            1.0002212 
##     Sigma[country:points,(Intercept)]         Sigma[country:points,points] 
##                            1.0002307                            1.0002394
```

```
tidy_sum<- tidy(wine_reg_model, effects= "fixed",
     conf.int = TRUE, conf.level = 0.90)
B0<- tidy_sum$estimate[1]
B1<- tidy_sum$estimate[2]
tidy_sum
```

```
## # A tibble: 2 × 5
##   term        estimate std.error conf.low conf.high
##   <chr>          <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)  -506.      75.4    -632.     -379. 
## 2 points          6.10     0.864     4.64      7.54
```

```
wine_data %>%
  add_fitted_draws(wine_reg_model, n = 200, re_formula = NA)%>%
  ggplot(aes(x = points, y = price))+
  geom_line(aes(y = .value, group= .draw), alpha = 0.1) +
  geom_abline(intercept = B0, slope = B1, color = "blue")
```

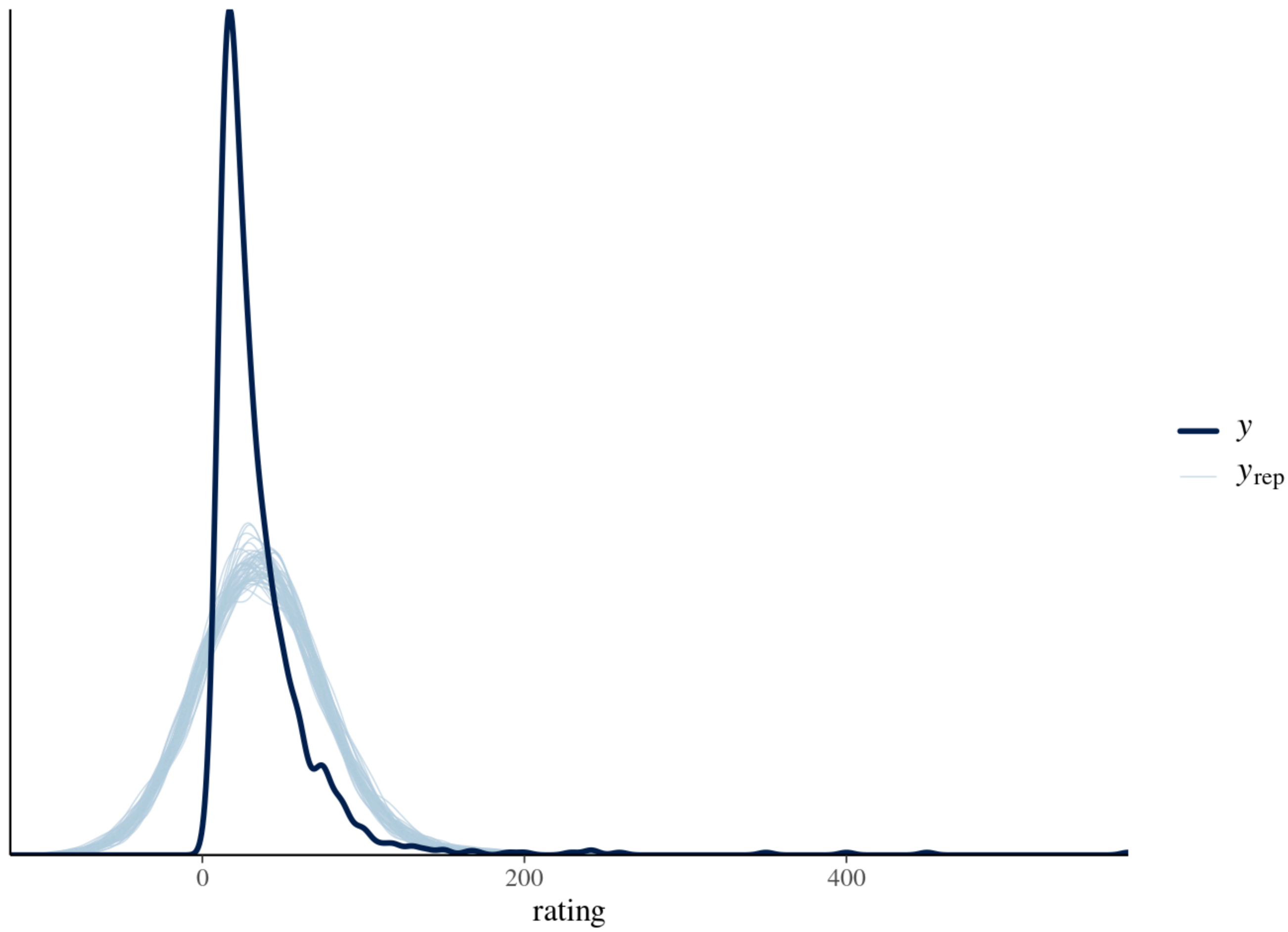
```
## Warning in fitted_draws.default(model = model, newdata = newdata, ..., n = n): `fitted_draws`
and `add_fitted_draws` are deprecated as their names were confusing.
## - Use [add_]epred_draws() to get the expectation of the posterior predictive.
## - Use [add_]linpred_draws() to get the distribution of the linear predictor.
## - For example, you used [add_]fitted_draws(..., scale = "response"), which
##   means you most likely want [add_]epred_draws(...).
## NOTE: When updating to the new functions, note that the `model` parameter is now
##   named `object` and the `n` parameter is now named `ndraws`.
```




```
# Get MCMC chains for the runner-specific intercepts & slopes
country_chains_2 <- wine_reg_model %>%
  spread_draws(`(Intercept)`, b[term, country], `points`) %>%
  pivot_wider(names_from = term, names_glue = "b_{term}",
              values_from = b) %>%
  mutate(country_intercept = `(Intercept)` + `b_(Intercept)`,
         country_points = points + b_points)
# Posterior medians of country-specific models
country_summaries_2 <- country_chains_2 %>%
  group_by(country) %>%
  summarize(country_intercept = median(country_intercept),
            country_points = median(country_points))
# Check it out
country_summaries_2
```

```
## # A tibble: 12 × 3
##    country              country_intercept country_points
##    <chr>                            <dbl>          <dbl>
##  1 country:Argentina                -411.           5.03
##  2 country:Australia                -863.          10.2 
##  3 country:Austria                  -338.           4.12
##  4 country:Chile                    -362.           4.42
##  5 country:France                   -972.          11.4 
##  6 country:Germany                  -427.           5.16
##  7 country:Italy                    -520.           6.27
##  8 country:New_Zealand              -455.           5.49
##  9 country:Portugal                 -367.           4.46
## 10 country:South_Africa             -514.           6.19
## 11 country:Spain                    -494.           5.98
## 12 country:US                       -330.           4.14
```

```
pp_check(wine_reg_model)+
  labs(x = "rating")
```




```
set.seed(36501)
prediction_summary(model = wine_reg_model, data = wine_data)
```

```
##        mae mae_scaled within_50 within_95
## 1 11.72726   0.366392 0.7849794 0.9794239
```