homework7

Christopher Huong

2024-03-11

```
library(rethinking)
```

7E1. State the three motivating criteria that define information entropy. Try to express each in your own words.

Information entropy is a measure of uncertainty in a probability distribution.
(1) Because probabilities are continuous (e.g., can exist as any value between 0 and 1), the corresponding uncertainty measure should also be continuous.
(2) Because as the number of possible events increases, there are more ways to be wrong, the measure of uncertainty should also be a function of number of events.
(3) Because the events are mutually exclusive, the measure of total uncertainty should be sum of the uncertainty of each individual event (I think)

7E2. Suppose a coin is weighted such that, when it is tossed and lands on a table, it comes up heads 70% of the time. What is the entropy of this coin?

```
p_coin <- c(0.70, 0.30)
-sum(p_coin*log(p_coin))
```

```
## [1] 0.6108643
```

7E3. Suppose a four-sided die is loaded such that, when tossed onto a table, it shows “1” 20%, “2” 25%, “3” 25%, and “4” 30% of the time. What is the entropy of this die?

```
p_dice <- c(0.20, 0.25, 0.25, 0.30)
-sum(p_dice*log(p_dice))
```

```
## [1] 1.376227
```

7E4. Suppose another four-sided die is loaded such that it never shows “4”. The other three sides show equally often. What is the entropy of this die?

” In other words, events that never happen drop out. Just remember that when an event never happens, there’s no point in keeping it in the model.”

```
p_dice2 <- c(1/3, 1/3, 1/3)
-sum(p_dice2*log(p_dice2))
```

```
## [1] 1.098612
```

7M1. Write down and compare the definitions of AIC and WAIC. Which of these criteria is most general? Which assumptions are required to transform the more general criterion into a less general one?

AIC = Deviance(training data) + 2p = -2lppd + 2p
Deviance = -2*lppd = log-probability score for Bayesian models

```
data("WaffleDivorce"); d<- WaffleDivorce
d$A <- standardize(d$MedianAgeMarriage)
d$D <- standardize(d$Divorce)
d$M <- standardize(d$Marriage)
```

Define three Bayesian linear models

```
# D ~ A
m_DA <- quap(
  alist(D ~ dnorm(mu, sigma),
        mu <- a * bA*A,
        a ~ dnorm(0, 0.2),
        bA ~ dnorm(0, 0.5),
        sigma ~ dexp(1)
        ),data=d)

# D ~ M
m_DM <- quap(
  alist(D ~ dnorm(mu, sigma),
        mu <- a + bM*M,
        a ~ dnorm(0, 0.2),
        bM ~ dnorm(0, 0.5),
        sigma ~ dexp(1)
        ),data=d)

# D ~ M + A
m_DMA <- quap(
  alist(
    D ~ dnorm(mu, sigma),
    mu <- a + bM*M + bA*A,
    a ~ dnorm(0, 0.2),
    c(bM, bA) ~ dnorm(0, 0.5),
    sigma ~ dexp(1)
  ), data=d)
```

Write function to compute lppd (log-probability score, i.e., the relative distance of a probability distribution from the true(?) distribution)

```
lppd_fun <- function(m) {
  #10k samples of log-probabilities for each observation, given the posterior probability distribution
  logprob <- sim(m, ll=T, n=1e4)
  #number of observations
  n <- ncol(logprob)
  #number of log-prob samples we simulated
  ns <- nrow(logprob)
  #function to exponentiate a vector of values, then log their sum
  lse <- function(x) {
    xmax <- max(x)
    xsum <- sum(exp(x-xmax))
    xmax + log(xsum)
  }
  #take the log of summed exponentiated log-probability samples for each observation.
  #subtract log of sample size for each, which equals dividing by the number of samples (not sure how that is)
  f <- function(i) lse(logprob[,i]) - log(ns)
  #apply this function to all 50 observations
  lppd <- sapply(1:n, f)
  #sum them for total log-probability score for the data
  lppd <- sum(lppd)
  return(lppd)
}
```

Use lppd to compute AIC

```
aic_fun <- function(m){
  lppd <- lppd_fun(m)
  #compute AIC by multiplying the lppd by -2, and adding the penalty term
  # number of parameters should be number of rows in precis() output
  aic <- -2*lppd + 2*(precis(m) |> nrow())
  return(aic)
}
```

```
sapply(c(m_DA, m_DM, m_DMA), aic_fun)
```

```
## [1] 125.8255 139.1177 125.5580
```

According to this AIC function I hopefully made correctly, the D~M model has the worst out-of-sample predictive accuracy

Now write function to compute WAIC

WAIC(y,0) = -2(lppd - sum(var0 * log P(y|0)))

y = observations
0 = posterior probability distribution
penalty = variance in log-probabilities for each observation, summed for total penalty

WAIC for D ~ M

```
n_samples <- 1000
# draw probability samples from posterior probability distribution
post <- extract.samples(m_DM, n=n_samples)


logprob <- sapply(1:n_samples,
                  function(s) {
                    # compute a vector of predicted mu's (describing the Gaussian distribution of D for a certain value of M) for each 50 observations of M, for each 1000 posterior probability sample
                    mu <- post$a[s] + post$bM[s] * d$M
                    # log-likelihoods of each observed value of D, given the model (mu's and sigma's computed from the posterior distribution)
                    dnorm(d$D, mu, post$sigma[s], log=T)
                  })


lppd <- lppd_fun(m_DM) #total lppd score
n_cases <- nrow(d)
pWAIC <- sapply(1:n_cases, function(i) var(logprob[i, ]))

-2*(lppd - sum(pWAIC))
```

```
## [1] 139.7559
```

```
#compare to WAIC function
WAIC(m_DM)
```

```
##       WAIC      lppd  penalty  std_err
## 1 140.1787 -66.56473 3.524605 10.72589
```

Pretty close. Now for D ~ A

```
n_samples <- 1000
# draw probability samples from posterior probability distribution
post <- extract.samples(m_DA, n=n_samples)


logprob <- sapply(1:n_samples,
                  function(s) {
                    mu <- post$a[s] + post$bA[s] * d$A
                    dnorm(d$D, mu, post$sigma[s], log=T)
                  })

lppd <- lppd_fun(m_DA)

n_cases <- nrow(d)
pWAIC <- sapply(1:n_cases, function(i) var(logprob[i, ]))

-2*(lppd - sum(pWAIC))
```

```
## [1] 212.2811
```

```
#compare to WAIC function
WAIC(m_DA)
```

```
##       WAIC      lppd  penalty  std_err
## 1 126.5267 -59.93308 3.330284 12.12564
```

No clue what went wrong with this one

Anyways, the WAIC is more general for reasons I do not understand. They both have lppd + a penalty term. The penalty term in WAIC is estimated from each observation, while may make it more robust to stuff.
I supposed when the AIC assumptions are met (priors flat or overhwelmed by the likelihood, multivariate Gaussian posterior distribution, and N >> p) then the criterions converge.

7M2. Explain the difference between model selection and model comparison. What information is lost under model selection?

Model selection drops the “worst fitting” models from all interpretation. Model comparison inspects the difference in predictive accuracy metrics, and which variables are responsible for those differences.

7M3. When comparing models with an information criterion, why must all models be fit

to exactly the same observations? What would happen to the information criterion values, if the models were fit to different numbers of observations? Perform some experiments, if you are not sure.

Since the raw magnitudes of information criterions are uninterpretable, and only relative scores are used, those relative scores must be derived from the same models to be comparable.

7M4. What happens to the effective number of parameters, as measured by PSIS or WAIC, as a prior becomes more concentrated? Why? Perform some experiments, if you are not sure.

I’m guessing since regularization (concentrated priors) improves out-of-sample predictive accuracy, which are assessed by information criterions, that the penalty term will shrink. Thus, the effective number of parameters shrinks.

7M5. Provide an informal explanation of why informative priors reduce overfitting.

Flat (i.e., uninformed) priors are flexible and will be sensitive to learning from the data. When we have an idea of the plausible range of values of a parameter, we can restrict the priors of those parameters, making those parameters less sensitive to the training data.

7M6. Provide an informal explanation of why overly informative priors result in underfitting.

Highly concentrated priors (i.e., small sigmas) will resist changing given the data, since the data will be seen as extreme, and thus unlikely values. Thus highly informative priors may be insensitive to true signals in the data

---

7H1. In 2007, The Wall Street Journal published an editorial (“We’re Number One,

Alas”) with a graph of corporate tax rates in 29 countries plotted against tax revenue. A badly fit curve was drawn in (reconstructed at right), seemingly by hand, to make the argument that the relationship between tax rate and tax revenue increases and then declines, such that higher tax rates can actually produce less tax revenue. I want you to actually fit a curve to these data, found in data(Laffer). Consider models that use tax rate to predict tax revenue. Compare, using WAIC or PSIS, a straight-line model to any curved models you like. What do you conclude about the relationship between tax rate and tax revenue?

```
data(Laffer); d <- Laffer
d$TRev <- standardize(d$tax_revenue)
d$TRate <- standardize(d$tax_rate)
d$TRate_2 <- d$TRate^2
```

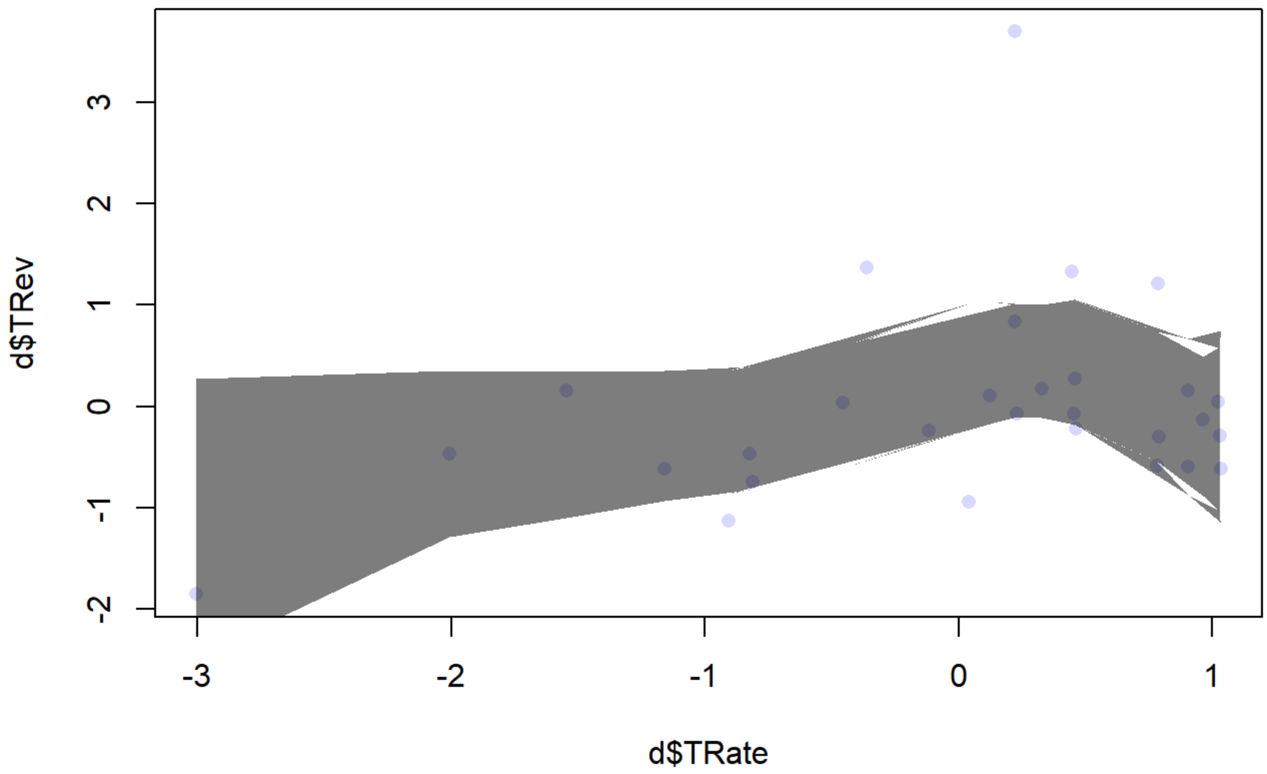
```r
# fit a linear model
m1 <- quap(alist(
  TRev ~ dnorm(mu, sigma),
  mu <- a + b*TRate,
  a ~ dnorm(0, 0.2),
  b ~ dnorm(0, 0.5),
  sigma ~ dexp(1)
 ),data=d)

# fit a quadratic model
m2 <- quap(alist(
  TRev ~ dnorm(mu, sigma),
  mu <- a + b1*TRate + b2*TRate_2,
  a ~ dnorm(0, 0.2),
  c(b1, b2) ~ dnorm(0, 0.5),
  sigma ~ dexp(1)
),data=d)

#try a spline I guess
#polynomial degree. 3 = cubic splines
library(splines)
num_knots <- 5
knot_list <- quantile(d$TRate, probs=seq(0,1, length.out=num_knots))
#get basis functions
B <- bs(d$TRate,
        knots=knot_list[-c(1,num_knots)], #drop first and last knot
        degree=3, intercept=T)


m3 <- quap(
  alist(
    TRev ~ dnorm(mu, sigma),
    mu <- a + B %*% w,
    a ~ dnorm(0, 0.2),
    w ~ dnorm(0,1),
    sigma ~ dexp(1)),
  data=list(TRev=d$TRev, B=B),
  start=list(w=rep(0, ncol(B))))


mu <- link(m3)
mu_PI <- apply(mu,2,PI,0.97)
plot(d$TRate, d$TRev, col=col.alpha(rangi2,0.3),pch=16)
shade(mu_PI, d$TRate, col=col.alpha("black",0.5))
```



```
compare(m1, m2, m3, func = WAIC)
```

```
##        WAIC       SE    dWAIC      dSE     pWAIC     weight
## m2 87.94867 23.73971 0.000000       NA  6.540005 0.63368723
## m1 89.23551 22.67424 1.286842 2.002672  6.209947 0.33299738
## m3 93.83974 26.07308 5.891072 3.290555 10.606341 0.03331539
```

The linear model seems to have the best predictive accuracy based on the lowest WAIC score, but the differences between the 3 models do not seem to be reliably large. Since sample size is relatively small, check PSIS score too

```
compare(m1, m2, m3, func = PSIS)
```

```
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
```

```
##        PSIS       SE    dPSIS      dSE     pPSIS    weight
## m2 92.37293 28.58874 0.000000       NA  8.710893 0.7349363
## m1 94.92498 28.20701 2.552053 1.760674  9.025265 0.2051532
## m3 97.38679 29.55646 5.013864 2.807162 12.485368 0.0599105
```

Again, differences between models do not seem reliably different. There seems to be a problematic outlier based on warning message.

```
PSIS(m1, pointwise=T) |> round(3)
```

```
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
```

```
##      PSIS    lppd penalty std_err      k
## 1   3.914  -1.957   0.536  27.006  0.689
## 2   1.849  -0.925   0.029  27.006  0.482
## 3   2.265  -1.133   0.061  27.006  0.399
## 4   1.828  -0.914   0.023  27.006  0.288
## 5   1.767  -0.884   0.019  27.006  0.144
## 6   2.035  -1.018   0.024  27.006  0.184
## 7   2.693  -1.346   0.051  27.006  0.271
## 8   2.829  -1.415   0.025  27.006  0.392
## 9   1.722  -0.861   0.017  27.006 -0.256
## 10  1.729  -0.865   0.017  27.006 -0.242
## 11  4.575  -2.287   0.141  27.006  0.483
## 12 29.011 -14.505   6.812  27.006  1.713
## 13  3.564  -1.782   0.062  27.006  0.185
## 14  2.446  -1.223   0.017  27.006  0.134
## 15  1.685  -0.842   0.017  27.006 -0.327
## 16  1.703  -0.851   0.017  27.006 -0.227
## 17  1.691  -0.845   0.017  27.006 -0.340
## 18  1.712  -0.856   0.017  27.006 -0.299
## 19  2.965  -1.482   0.052  27.006  0.012
## 20  1.735  -0.867   0.016  27.006 -0.105
## 21  1.840  -0.920   0.016  27.006 -0.051
## 22  1.720  -0.860   0.017  27.006 -0.101
## 23  1.790  -0.895   0.019  27.006  0.146
## 24  1.918  -0.959   0.021  27.006  0.284
## 25  2.050  -1.025   0.022  27.006  0.192
## 26  2.545  -1.273   0.039  27.006  0.145
## 27  2.663  -1.331   0.052  27.006  0.220
## 28  2.828  -1.414   0.073  27.006  0.309
## 29  2.157  -1.078   0.032  27.006  0.251
```

12th row has large k. Try a robust regression using Student’s t with thicker tail (parameter=2)

```
m4 <- quap(alist(
  TRev ~ dstudent(2, mu, sigma),
  mu <- a + b*TRate,
  a ~ dnorm(0, 0.2),
  b ~ dnorm(0, 0.5),
  sigma ~ dexp(1)
),data=d)
```

Compare PSIS scores

```
compare(m1, m2, m3, m4, func=PSIS)
```

```
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
## Some Pareto k values are very high (>1). Set pointwise=TRUE to inspect individual points.
```

```
##        PSIS       SE    dPSIS      dSE     pPSIS       weight
## m4 73.98650 13.58034  0.00000       NA  3.509588 9.999258e-01
## m3 94.47017 26.79308 20.48367 18.20788 10.944761 3.564474e-05
## m2 94.88199 30.97214 20.89549 22.24245 10.075337 2.901145e-05
## m1 97.11114 30.18303 23.12465 21.40461 10.118504 9.517313e-06
```

The linear model with Student’s T distribution (as opposed to a Gaussian distribution) seems to have the best predictive accuracy.

7H2. In the Laffer data, there is one country with a high tax revenue that is an outlier. Use PSIS and WAIC to measure the importance of this outlier in the models you fit in the previous problem. Then use robust regression with a Student’s t distribution to revisit the curve fitting problem. How much does a curved relationship depend upon the outlier point?

Oh I already did that