# MA678Homework4

Chang Lu

10/4/2022

## 13.5 Interpreting logistic regression coefficients

Here is a fitted model from the Bangladesh analysis predicting whether a person with high-arsenic drinking water will switch wells, given the arsenic level in their existing well and the distance to the nearest safe well:

```
stan_glm(formula = switch ~ dist100 + arsenic, family=binomial(link="logit"), data=wells)
             Median MAD_SD
(Intercept)   0.00   0.08
dist100      -0.90   0.10
arsenic       0.46   0.04
```

Compare two people who live the same distance from the nearest well but whose arsenic levels differ, with one person having an arsenic level of 0.5 and the other person having a level of 1.0. You will estimate how much more likely this second person is to switch wells. Give an approximate estimate, standard error, 50% interval, and 95% interval, using two different methods:

### (a)

Use the divide-by-4 rule, based on the information from this regression output.

According to the regression output, $0.46/4 = 0.115$, and this implies that a 1-unit increase in arsenic (from 0.5 to 1.0) increases the probability of switching wells by approximately 0.115 (11.5 percentage points).

To estimate the standard error of this effect, we can divide the standard deviation of the coefficient by 4:

$$SE = \frac{0.04}{4} = 0.01.$$

For the 50% Interval: The 50% confidence interval is approximately the median plus/minus 0.67 standard errors. For the 95% Interval: The 95% confidence interval is approximately the median plus/minus 1.96 standard errors. Therefore,

$$50\%\ CI = 0.115 \pm 0.67(0.01) = (0.108, 0.122)$$

$$95\%\ CI = 0.115 \pm 1.96(0.01) = (0.095, 0.135)$$

### (b)

Use predictive simulation from the fitted model in R, under the assumption that these two people each live 50 meters from the nearest safe well.

```r
library(rstanarm)
```

```
## Loading required package: Rcpp
```

```
## This is rstanarm version 2.32.1
```

```
## - See https://mc-stan.org/rstanarm/articles/priors for changes to default priors!
```

```
## - Default priors may change, so it's safest to specify priors, even if equivalent to the defaults.
```

```
## - For execution on a local, multicore CPU with excess RAM we recommend calling
```

```
##   options(mc.cores = parallel::detectCores())
```

```r
wells <- read.csv("wells.csv")

# Fit the logistic regression model using stan_glm
fit <- stan_glm(switch ~ dist100 + arsenic, family = binomial(link = "logit"), data = wells, refresh = 0

# Data for two individuals: one with arsenic level 0.5, the other with arsenic level 1.0
new_data <- data.frame(
  dist100 = c(0.5, 0.5),  # Both live 50 meters from the nearest safe well
  arsenic = c(0.5, 1.0)   # Arsenic levels 0.5 and 1.0
)

# Simulate posterior predictive probabilities
posterior_predictions <- posterior_epred(fit, newdata = new_data)

# Calculate the difference in probabilities between the two individuals
prob_diff <- posterior_predictions[, 2] - posterior_predictions[, 1]

# Summarize the results
mean_diff <- mean(prob_diff)
se_diff <- sd(prob_diff)
ci_50 <- quantile(prob_diff, probs = c(0.25, 0.75))
ci_95 <- quantile(prob_diff, probs = c(0.025, 0.975))


list(
  mean_difference = mean_diff,
  standard_error = se_diff,
  ci_50 = ci_50,
  ci_95 = ci_95
)
```

```
## $mean_difference
## [1] 0.05742066
##
## $standard_error
## [1] 0.005104929
##
```

```
## $ci_50
##        25%        75%
## 0.05392749 0.06096672
##
## $ci_95
##       2.5%      97.5%
## 0.04746958 0.06742724
```

## 13.7 Graphing a fitted logistic regression

We downloaded data with weight (in pounds) and age (in years) from a random sample of American adults. We then defined a new variable:

```
heavy <- weight > 200
```

and fit a logistic regression, predicting heavy from `height` (in inches):

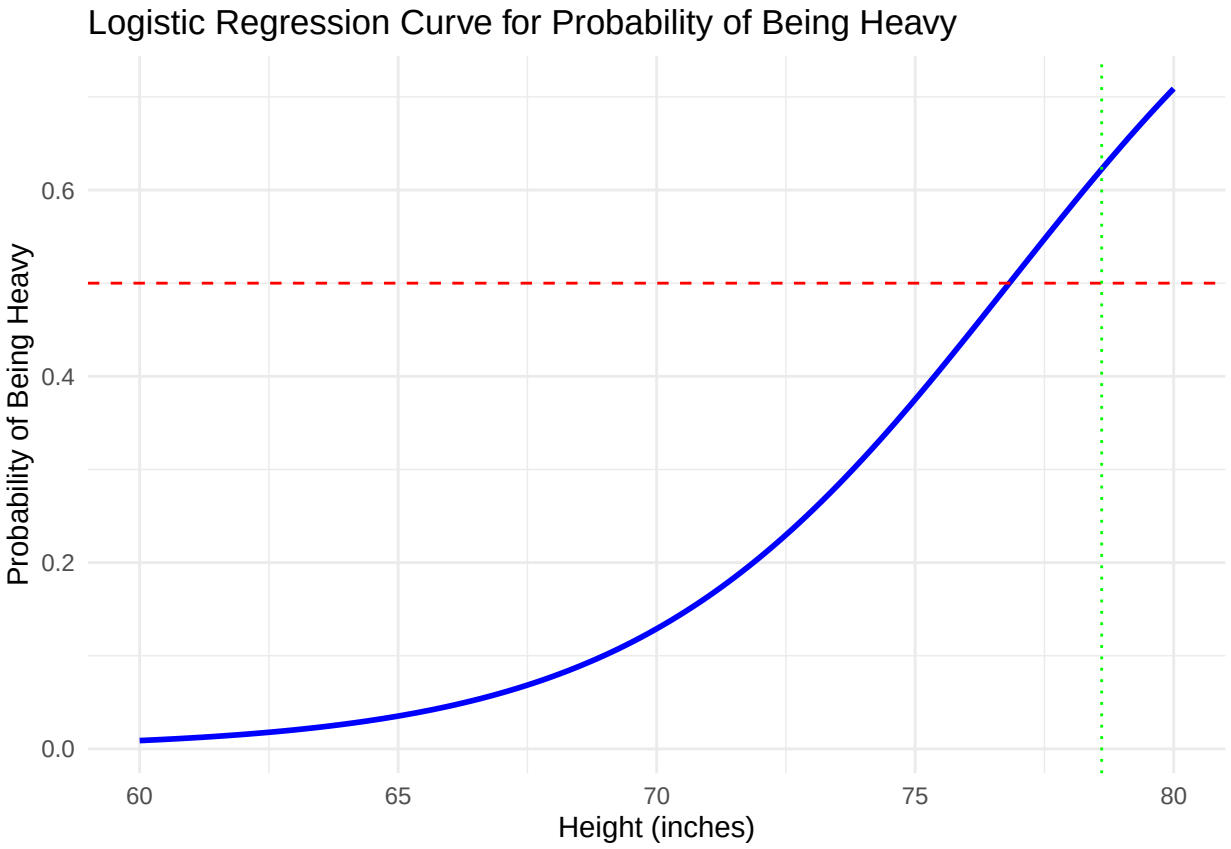
```
stan_glm(formula = heavy ~ height, family=binomial(link="logit"), data=health, refresh = 0)
             Median MAD_SD
(Intercept)  -21.51   1.60
height         0.28   0.02
```

### (a)

Graph the logistic regression curve (the probability that someone is heavy) over the approximate range of the data. Be clear where the line goes through the 50% probability point.

```
# Load necessary libraries
library(rstanarm)
library(ggplot2)

intercept <- -21.51
height_coef <- 0.28

# Define the logistic function
logistic_function <- function(height) {
  odds <- intercept + height_coef * height
  prob <- 1 / (1 + exp(-odds))
  return(prob)
}

# Create a range of height values (in inches) over which to plot the logistic curve
height_values <- seq(60, 80, by = 0.1)  # Range of heights from 60 to 80 inches (adjust based on your d

# Compute the predicted probabilities
probabilities <- logistic_function(height_values)

# Create a data frame for plotting
plot_data <- data.frame(height = height_values, probability = probabilities)

# Create the plot using ggplot2
ggplot(plot_data, aes(x = height, y = probability)) +
```

```
  geom_line(color = "blue", linewidth = 1) +
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "red") +  # Add a line at 50% probability
  geom_vline(xintercept = (0.5 - intercept) / height_coef, linetype = "dotted", color = "green") +  # P
  labs(title = "Logistic Regression Curve for Probability of Being Heavy",
       x = "Height (inches)",
       y = "Probability of Being Heavy") +
  theme_minimal()
```




**(b)**

Fill in the blank: near the 50% point, comparing two people who differ by one inch in height, you'll expect a difference of 0.07 in the probability of being heavy.

## 13.8 Linear transformations

In the regression from the previous exercise, suppose you replaced height in inches by height in centimeters. What would then be the intercept and slope?

As we all know, 1 inch is equal to 2.54 centimeters.

For the slope: The current slope for height in inches is 0.28. Since 1 inch equals 2.54 centimeters, the slope in centimeters will be adjusted by dividing the original slope by 2.54.

$$Slope(in\ cm) = \frac{0.28}{2.54} = 0.1102$$

For the intercept: The new intercept remains the same numerically as the original intercept because the conversion affects only the slope.

## 13.10 Expressing a comparison of proportions as a logistic regression

A randomized experiment is performed within a survey, and 1000 people are contacted. Half the people contacted are promised a $5 incentive to participate, and half are not promised an incentive. The result is a 50% response rate among the treated group and 40% response rate among the control group.

### (a)

Set up these results as data in R. From these data, fit a logistic regression of response on the treatment indicator.

```
# Load necessary libraries
library(rstanarm)

# Set up the data: create a data frame with binary variables for response and treatment
survey_data <- data.frame(
  response = c(rep(1, 250), rep(0, 250), rep(1, 200), rep(0, 300)),  # Response: 1 = responded, 0 = did
  treatment = c(rep(1, 500), rep(0, 500))  # Treatment: 1 = promised incentive, 0 = no incentive
)

# Fit logistic regression model (response ~ treatment)
fit <- stan_glm(response ~ treatment, family = binomial(link = "logit"), data = survey_data, refresh = 

# Display model summary
summary(fit)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      response ~ treatment
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 1000
##  predictors:   2
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) -0.4    0.1 -0.5  -0.4  -0.3
## treatment    0.4    0.1  0.2   0.4   0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.4    0.0  0.4   0.4   0.5
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
```

```
##                mcse Rhat n_eff
## (Intercept)   0.0  1.0  2875
## treatment     0.0  1.0  2892
## mean_PPD      0.0  1.0  2980
## log-posterior 0.0  1.0  1573
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

**(b)**

Compare to the results from Exercise 4.1.

```r
# Proportions and sample sizes
p1 <- 0.50  # Response rate in treated group
p2 <- 0.40  # Response rate in control group
n1 <- 500   # Number of people in treated group
n2 <- 500   # Number of people in control group

# Calculate the ATE (difference in proportions)
ATE <- p1 - p2

# Calculate the standard error (SE) of the ATE
SE <- sqrt((p1 * (1 - p1)) / n1 + (p2 * (1 - p2)) / n2)

# Output the ATE and SE
list(ATE = ATE, SE = SE)
```

```
## $ATE
## [1] 0.1
##
## $SE
## [1] 0.03130495
```

Comparison:

```r
# Re-fit the logistic regression model to ensure the correct variables are used
fit <- stan_glm(response ~ treatment, family = binomial(link = "logit"), data = survey_data, refresh = 

# Predict probabilities for control and treated groups
new_data <- data.frame(treatment = c(0, 1))  # Control = 0, Treated = 1
predicted_probs <- posterior_epred(fit, newdata = new_data)

# Calculate the ATE from the logistic regression (difference in probabilities)
ATE_logistic <- predicted_probs[2] - predicted_probs[1]

# Output the ATE from logistic regression
ATE_logistic
```

```
## [1] -0.007966549
```

There is no obvious difference.

## 13.11 Building a logistic regression model

The folder `Rodents` contains data on rodents in a sample of New York City apartments.

### (a)

Build a logistic regression model to predict the presence of rodents (the variable `rodent2` in the dataset) given indicators for the ethnic groups (`race`). Combine categories as appropriate. Discuss the estimated coefficients in the model.

```
rodents_data <- read.csv("hvs02_sorted.csv")

# Convert `rodent2` and `race` into factor variables
rodents_data$rodent2 <- as.factor(rodents_data$rodent2)
rodents_data$race <- as.factor(rodents_data$race)

# Fit the logistic regression model: Predict rodent2 using race
fit <- stan_glm(rodent2 ~ race, family = binomial(link = "logit"), data = rodents_data, refresh = 0)

# Display the summary of the model
summary(fit)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      rodent2 ~ race
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 13931
##  predictors:   7
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) -2.2    0.0 -2.2  -2.2  -2.1
## race2        1.5    0.1  1.4   1.5   1.5
## race3        1.6    0.1  1.5   1.6   1.7
## race4        1.7    0.1  1.6   1.7   1.8
## race5        0.8    0.1  0.7   0.8   0.9
## race6        1.4    0.4  0.8   1.4   1.9
## race7        1.1    0.3  0.8   1.1   1.4
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.2    0.0  0.2   0.2   0.2
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  1448
## race2         0.0  1.0  1948
```

```
## race3         0.0  1.0  2559
## race4         0.0  1.0  2151
## race5         0.0  1.0  2583
## race6         0.0  1.0  4991
## race7         0.0  1.0  4361
## mean_PPD      0.0  1.0  4513
## log-posterior 0.0  1.0  1744
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

Summary of the Results:

1. Intercept: 1.1 The intercept represents the log-odds of having rodents for the reference group (the category that is omitted from the model, race1). 1.2 The intercept has a mean of -2.2, meaning that the baseline log-odds of rodents being present in apartments for race1 are quite low. 1.3 On the probability scale, this would correspond to a very low probability of rodent presence for race1.
2. Race Categories Each race coefficient represents the log-odds of rodent presence for that race compared to the reference race (race1). 2.1 Individuals in race groups race2, race3, race4, race5, race6, and race7 all have a higher likelihood of rodents being present in their apartments compared to the reference group (race1). 2.2 The risk of rodent presence is highest for individuals in race4, followed by race3, race2, and race6. 2.3 Individuals in race5 and race7 have a moderately higher likelihood of rodent presence compared to race1.
3. MCMC Diagnostics: 3.1 The Rhat values for all parameters are 1.0, which indicates good convergence of the Markov Chain Monte Carlo (MCMC) algorithm. Rhat values close to 1.0 show that the chains have mixed well and the model has converged. 3.2 The n_eff values (effective sample sizes) are all sufficiently large, which means the estimates are reliable. 3.3 mcse (Monte Carlo standard error) values are close to zero, indicating that the estimates from the model are precise.
4. Mean PPD (Posterior Predictive Distribution): The mean posterior predictive distribution (mean_PPD) is the model's average predicted probability of rodent presence across all observations. It's approximately 0.2, suggesting that the average predicted probability of rodent presence in the dataset is around 20%.

**(b)**

Add to your model some other potentially relevant predictors describing the apartment, building, and community district. Build your model using the general principles explained in Section 12.6. Discuss the coefficients for the ethnicity indicators in your model.

```r
# Convert relevant variables into factors if necessary
rodents_data$rodent2 <- as.factor(rodents_data$rodent2)
rodents_data$race <- as.factor(rodents_data$race)

# Fit the logistic regression model with additional predictors
fit_extended <- stan_glm(rodent2 ~ race + numunits + stories + personrm + vacrate + inthole2_Mean + int
                         family = binomial(link = "logit"),
                         data = rodents_data, refresh = 0)

# Display the summary of the extended model
summary(fit_extended)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      rodent2 ~ race + numunits + stories + personrm + vacrate + inthole2_Mean +
##     intcrack2_Mean
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 13931
##  predictors:   13
##
## Estimates:
##                  mean   sd   10%   50%   90%
## (Intercept)    -3.3    0.1 -3.4  -3.3  -3.2
## race2           0.9    0.1  0.8   0.9   1.0
## race3           1.0    0.1  0.9   1.0   1.1
## race4           1.1    0.1  1.0   1.1   1.2
## race5           0.6    0.1  0.5   0.6   0.7
## race6           0.8    0.4  0.2   0.8   1.3
## race7           0.7    0.3  0.4   0.7   1.1
## numunits        0.2    0.0  0.1   0.2   0.2
## stories        -0.3    0.0 -0.3  -0.3  -0.2
## personrm        0.6    0.1  0.5   0.6   0.6
## vacrate        -2.4    0.8 -3.4  -2.4  -1.4
## inthole2_Mean   7.5    1.4  5.7   7.5   9.3
## intcrack2_Mean  1.8    0.9  0.6   1.8   3.0
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.2    0.0  0.2   0.2   0.2
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##                mcse Rhat n_eff
## (Intercept)    0.0  1.0  3910
## race2          0.0  1.0  3233
## race3          0.0  1.0  3449
## race4          0.0  1.0  3201
## race5          0.0  1.0  3699
## race6          0.0  1.0  4368
## race7          0.0  1.0  4362
## numunits       0.0  1.0  2600
## stories        0.0  1.0  2692
## personrm       0.0  1.0  4757
## vacrate        0.0  1.0  3755
## inthole2_Mean  0.0  1.0  2783
## intcrack2_Mean 0.0  1.0  2522
## mean_PPD       0.0  1.0  4252
## log-posterior  0.1  1.0  1562
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

Ethnicity: The coefficients for race suggest that individuals from certain racial groups (especially race4, race3, and race2) are more likely to have rodent presence in their apartments compared to the reference group. This might reflect socioeconomic or neighborhood-level factors associated with different racial groups.

## 14.3 Graphing logistic regressions

The well-switching data described in Section 13.7 are in the folder `Arsenic`.

### (a)

Fit a logistic regression for the probability of switching using log (distance to nearest safe well) as a predictor.

```
library(rstanarm)

wells_data <- read.csv("wells.csv")

# Transform the distance variable to log(distance)
wells_data$log_dist <- log(wells_data$dist)

# Fit a logistic regression model using log(distance) as the predictor
fit <- stan_glm(switch ~ log_dist, family = binomial(link = "logit"), data = wells_data, refresh = 0)

# Display model summary
summary(fit)
```

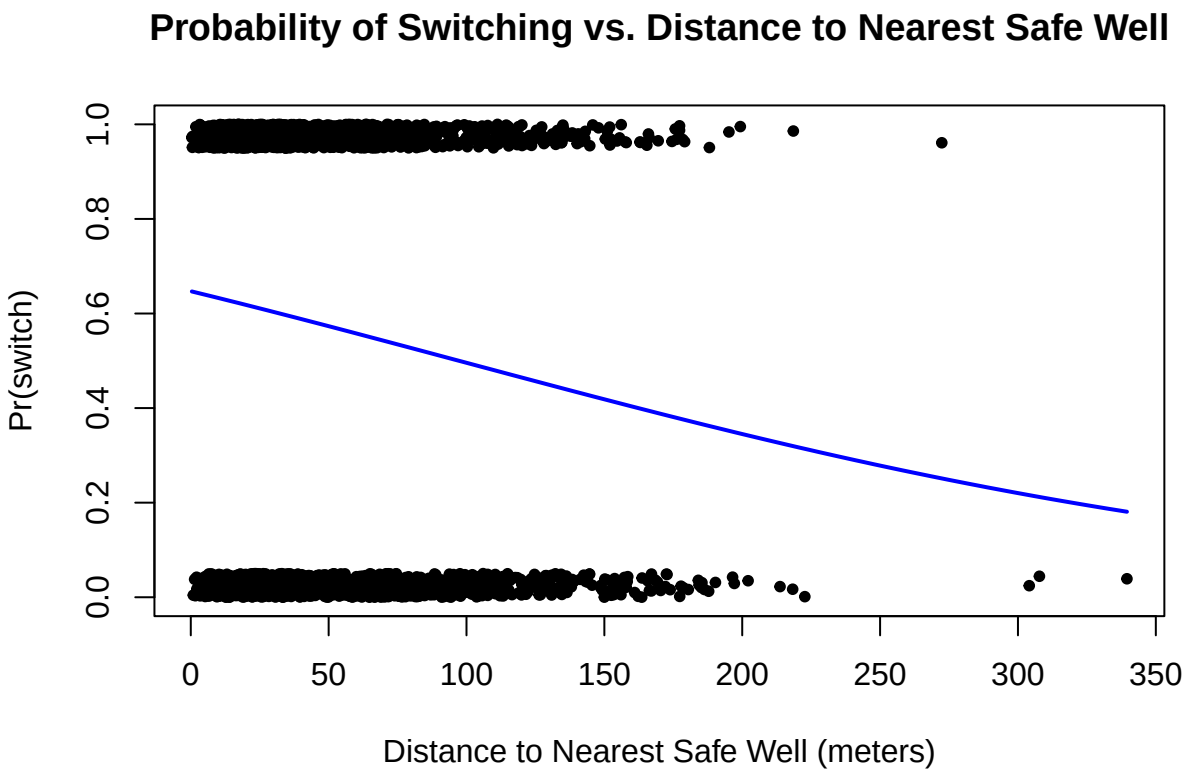
```
##
## Model Info:
##  function:     stan_glm
##  family:       binomial [logit]
##  formula:      switch ~ log_dist
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 3020
##  predictors:   2
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  1.0    0.2  0.8   1.0   1.2
## log_dist    -0.2    0.0 -0.3  -0.2  -0.1
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 0.6    0.0  0.6   0.6   0.6
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  2918
## log_dist      0.0  1.0  3022
## mean_PPD      0.0  1.0  3391
```

```
## log-posterior 0.0  1.0  1651
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

**(b)**

Make a graph similar to Figure 13.8b displaying Pr(switch) as a function of distance to nearest safe well, along with the data.

```r
wells_data$dist100 <- wells_data$dist / 100

# Fit a logistic regression model with distance as the predictor
fit_1 <- stan_glm(switch ~ dist100, family = binomial(link = "logit"), data = wells_data, refresh = 0)

# Function to jitter binary data
jitter_binary <- function(a, jitt = 0.05) {
  ifelse(a == 0, runif(length(a), 0, jitt), runif(length(a), 1 - jitt, 1))
}

# Jitter the switch data to avoid overplotting
wells_data$switch_jitter <- jitter_binary(wells_data$switch)

# Plot the jittered data
plot(wells_data$dist, wells_data$switch_jitter,
     xlab = "Distance to Nearest Safe Well (meters)",
     ylab = "Pr(switch)",
     main = "Probability of Switching vs. Distance to Nearest Safe Well",
     pch = 20, col = "black", ylim = c(0, 1))

# Add the fitted logistic regression curve to the plot
curve(plogis(coef(fit_1)[1] + coef(fit_1)[2] * (x / 100)), add = TRUE, col = "blue", lwd = 2)
```



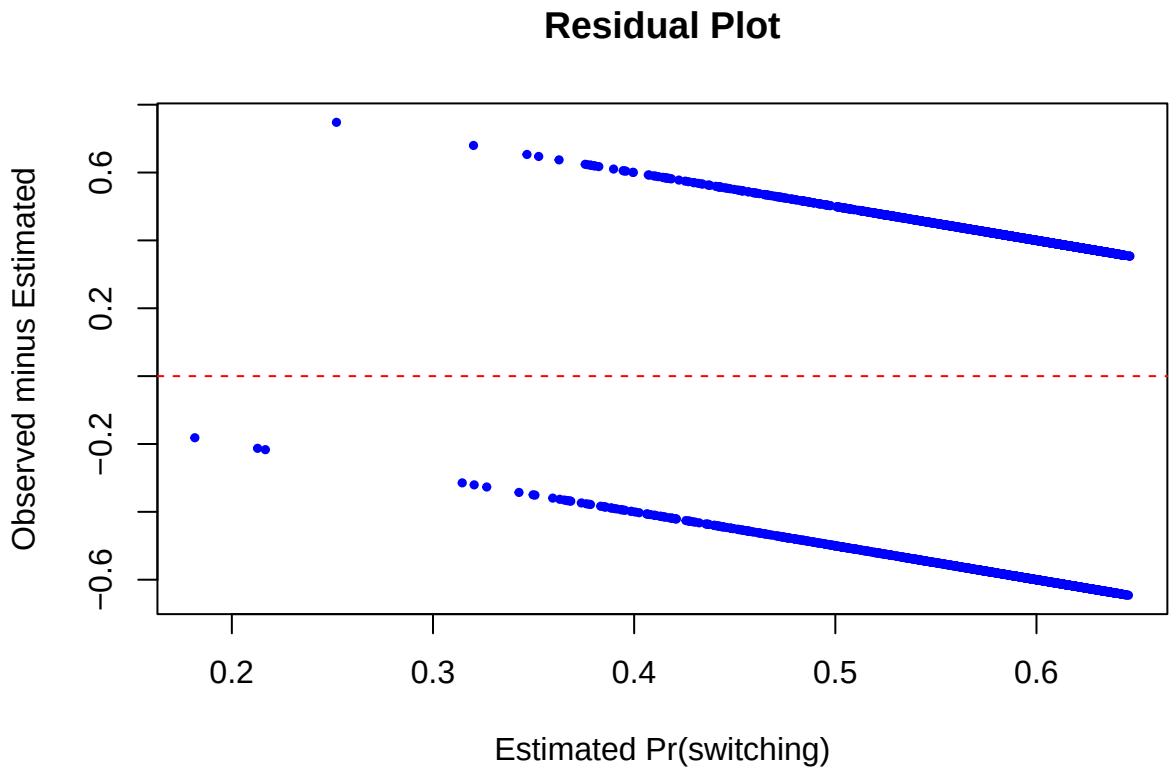

(c)

Make a residual plot and binned residual plot as in Figure 14.8.

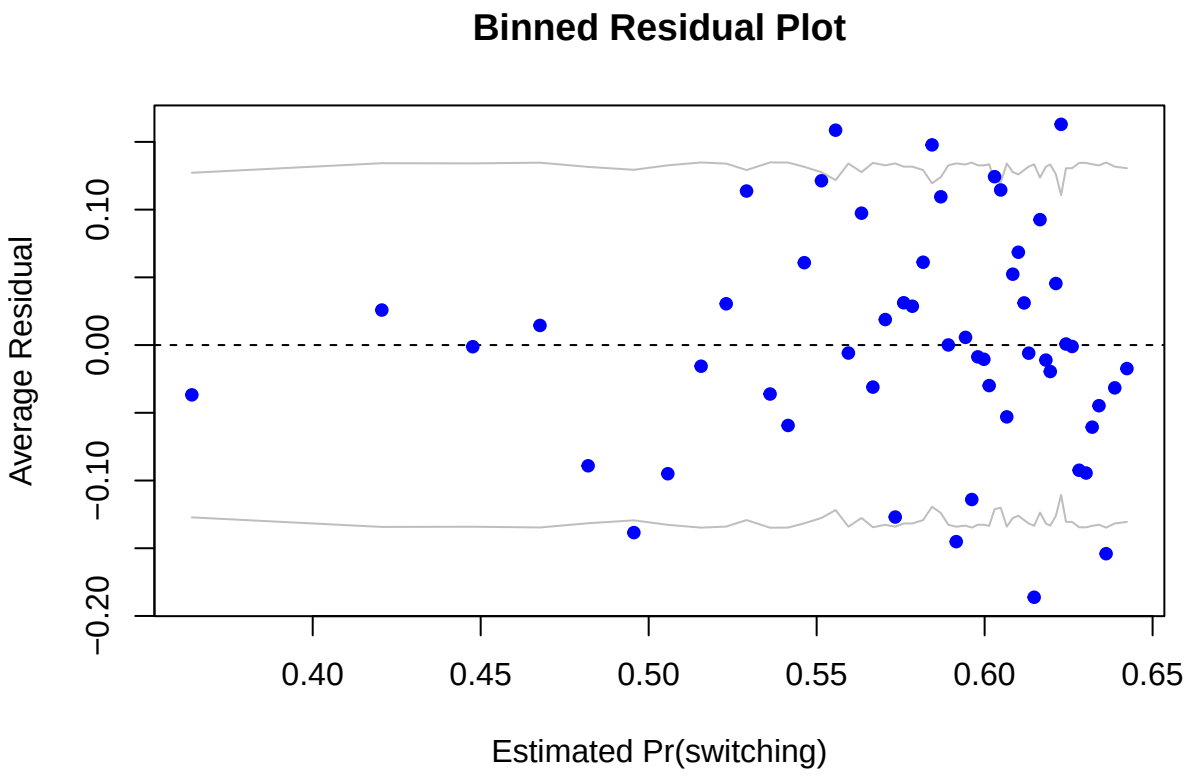
```
# Load necessary libraries
library(arm)  # For binnedplot()
```

```
## Loading required package: MASS

## Loading required package: Matrix

## Loading required package: lme4

##
## arm (Version 1.14-4, built: 2024-4-1)

## Working directory is /Users/lclouisbu.edu/Desktop/678HW4

##
## Attaching package: 'arm'

## The following objects are masked from 'package:rstanarm':
##
##     invlogit, logit
```

```r
# Fit a logistic regression model using glm()
logwellsmodel <- glm(switch ~ dist100, family = binomial(link = "logit"), data = wells_data)

# Predict probabilities using the fitted model
wells_data$predicted_prob <- predict(logwellsmodel, type = "response")

# Calculate the residuals
wells_data$residuals <- residuals(logwellsmodel, type = "response")

# Create the Residual Plot
plot(wells_data$predicted_prob, wells_data$residuals,
     xlab = "Estimated Pr(switching)",
     ylab = "Observed minus Estimated",
     main = "Residual Plot",
     pch = 19, col = "blue", cex = 0.5)
abline(h = 0, col = "red", lty = 2)
```




```r
# Create the Binned Residual Plot
binnedplot(wells_data$predicted_prob, wells_data$residuals,
           xlab = "Estimated Pr(switching)",
           ylab = "Average Residual",
           main = "Binned Residual Plot",
           col.pts = "blue", col.int = "gray")
```




(d)

Compute the error rate of the fitted model and compare to the error rate of the null model.

```r
# Fit the logistic regression model
logwellsmodel <- glm(switch ~ dist100, family = binomial(link = "logit"), data = wells_data)

# Get the predicted probabilities from the fitted model
wells_data$predicted_prob <- predict(logwellsmodel, type = "response")

# Classify the predicted probabilities into 0 or 1
wells_data$predicted_class <- ifelse(wells_data$predicted_prob > 0.5, 1, 0)

# Calculate the error rate of the fitted model
error_rate_fitted <- mean(wells_data$predicted_class != wells_data$switch)

# Null model: predict the majority class (0 or 1 based on the average)
null_prediction <- ifelse(mean(wells_data$switch) > 0.5, 1, 0)

# Calculate the error rate of the null model
error_rate_null <- mean(null_prediction != wells_data$switch)

# Print the error rates
error_rate_fitted
```

```
## [1] 0.4046358
```

```
error_rate_null
```

```
## [1] 0.4248344
```

### (e)

Create indicator variables corresponding to `dist < 100`; `dist` between 100 and 200; and `dist > 200`. Fit a logistic regression for Pr(switch) using these indicators. With this new model, repeat the computations and graphs for part (a) of this exercise.

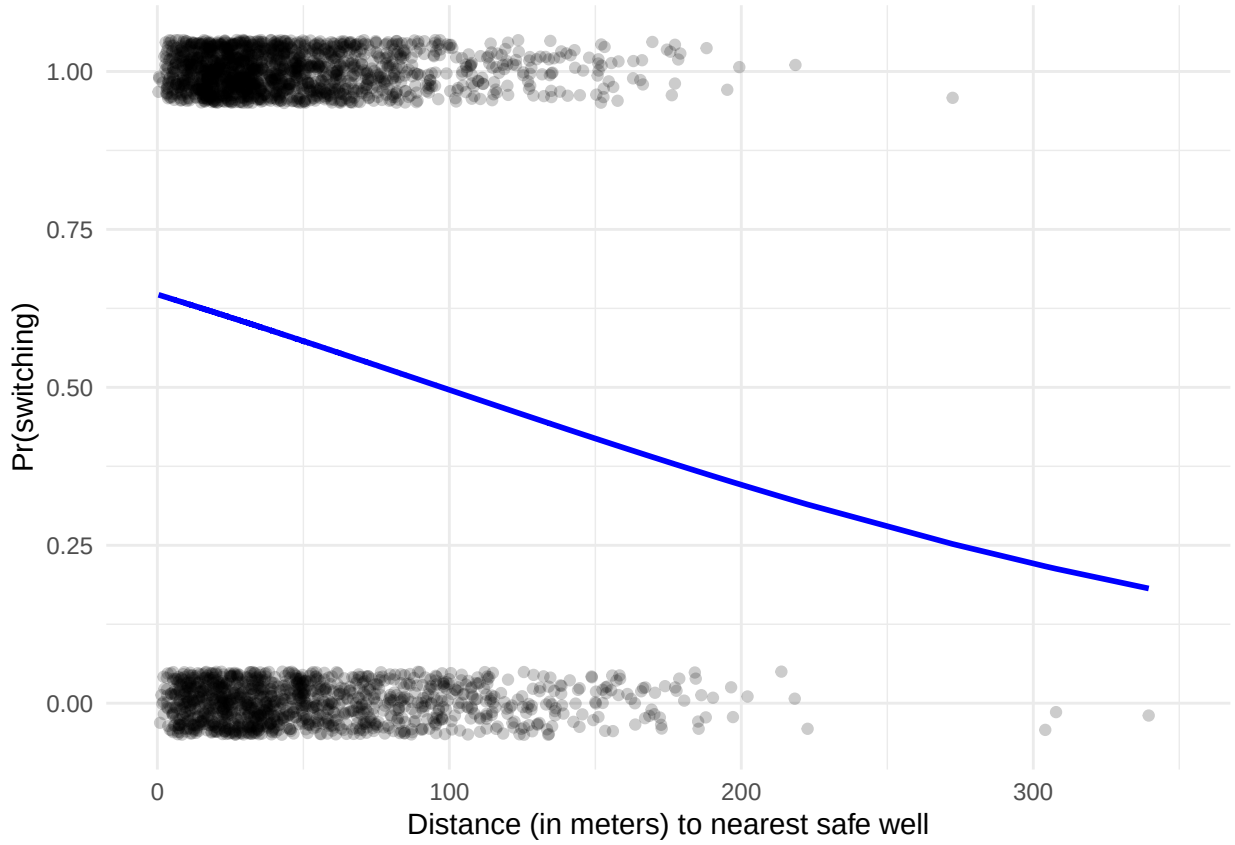
```
# Create indicator variables
wells_data$dist_lt_100 <- ifelse(wells_data$dist < 100, 1, 0)
wells_data$dist_100_200 <- ifelse(wells_data$dist >= 100 & wells_data$dist <= 200, 1, 0)
wells_data$dist_gt_200 <- ifelse(wells_data$dist > 200, 1, 0)

# Fit the logistic regression model with the indicators
wellsmodel <- glm(switch ~ dist_lt_100 + dist_100_200 + dist_gt_200,
                  family = binomial(link = "logit"), data = wells_data)

# Summary of the fitted model
summary(wellsmodel)
```

```
##
## Call:
## glm(formula = switch ~ dist_lt_100 + dist_100_200 + dist_gt_200,
##     family = binomial(link = "logit"), data = wells_data)
##
## Coefficients: (1 not defined because of singularities)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.2528     0.8018  -1.562   0.1182
## dist_lt_100    1.6264     0.8027   2.026   0.0428 *
## dist_100_200   0.9690     0.8103   1.196   0.2317
## dist_gt_200        NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 4084.7  on 3017  degrees of freedom
## AIC: 4090.7
##
## Number of Fisher Scoring iterations: 4
```

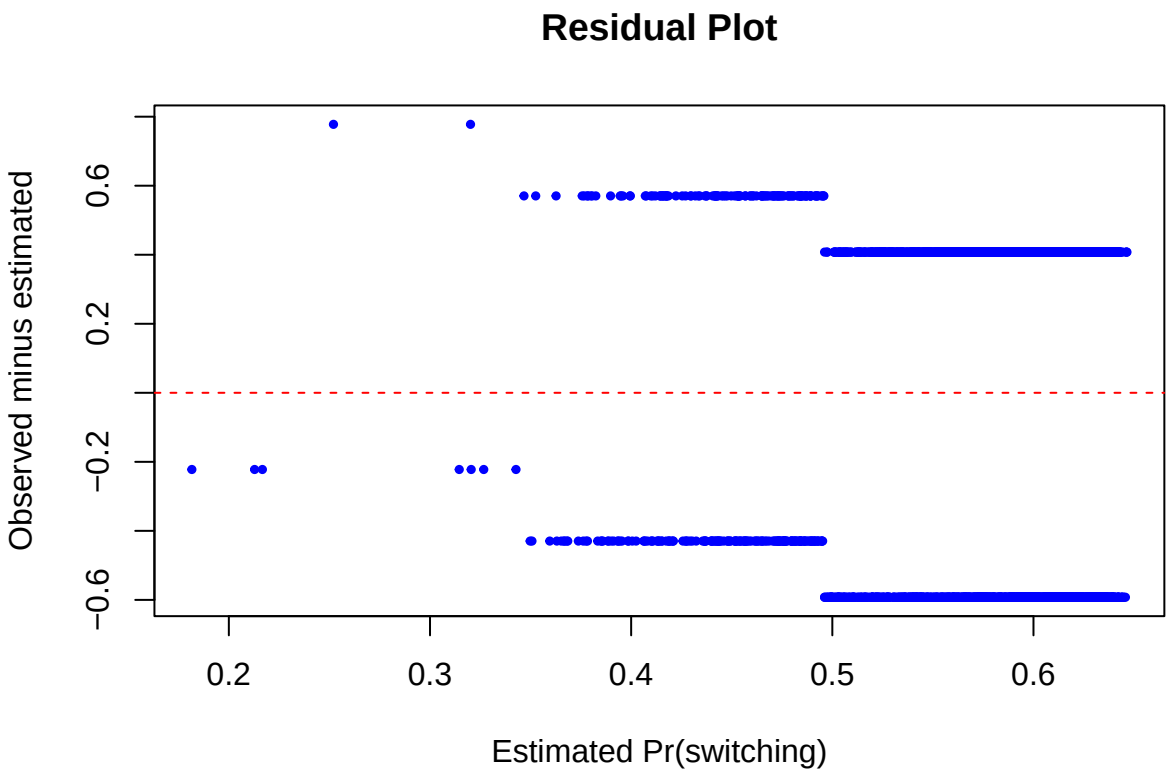
```
# Predicted probabilities
wells_data$predicted_prob3 <- predict(wellsmodel, type = "response")

# Plot of predicted probabilities
library(ggplot2)
ggplot(wells_data, aes(x = dist, y = switch)) +
```

```
  geom_jitter(width = 0, height = 0.05, alpha = 0.2, color = "black") +
  geom_line(aes(y = predicted_prob), color = "blue", size = 1) +
  labs(x = "Distance (in meters) to nearest safe well",
       y = "Pr(switching)") +
  theme_minimal() +
  scale_x_continuous(limits = c(0, 350))
```

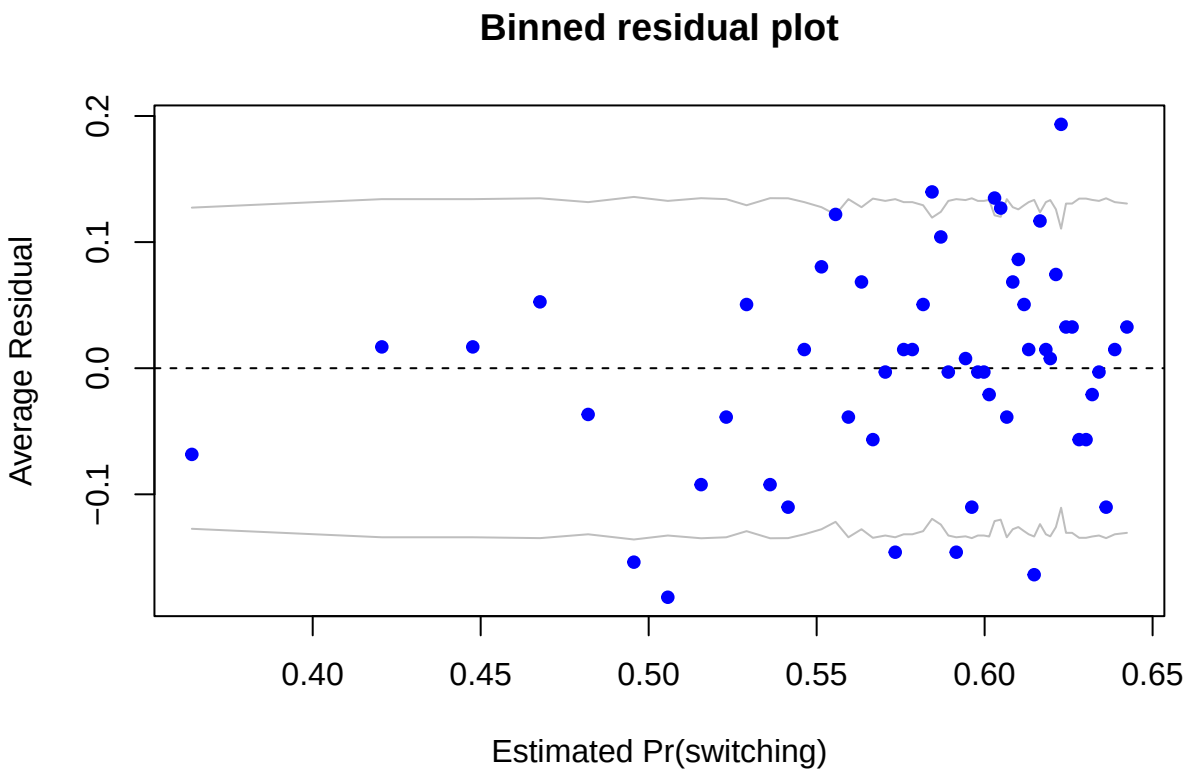
```
## Warning: Using 'size' aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use 'linewidth' instead.
## This warning is displayed once every 8 hours.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.
```




```
# Residual plot
wells_data$residuals <- residuals(wellsmodel, type = "response")
plot(wells_data$predicted_prob, wells_data$residuals,
     xlab = "Estimated Pr(switching)",
     ylab = "Observed minus estimated",
     main = "Residual Plot",
     pch = 19, col = "blue", cex = 0.5)
abline(h = 0, col = "red", lty = 2)
```




```
binnedplot(wells_data$predicted_prob, wells_data$residuals,
           xlab = "Estimated Pr(switching)",
           ylab = "Average Residual",
           main = "Binned residual plot",
           col.pts = "blue", col.int = "gray")
```




```
# Calculate error rates
wells_data$predicted_class <- ifelse(wells_data$predicted_prob > 0.5, 1, 0)
error_rate_fitted <- mean(wells_data$predicted_class != wells_data$switch)

# Null model error rate
null_prediction <- ifelse(mean(wells_data$switch) > 0.5, 1, 0)
error_rate_null <- mean(null_prediction != wells_data$switch)

# Print the error rates
error_rate_fitted
```

```
## [1] 0.4046358
```

```
error_rate_null
```

```
## [1] 0.4248344
```

## 14.7 Model building and comparison

Continue with the well-switching data described in the previous exercise.

### (a)

Fit a logistic regression for the probability of switching using, as predictors, distance, log(arsenic), and their interaction. Interpret the estimated coefficients and their standard errors.

```
wellsmodel2 <- glm(switch ~ dist * log(arsenic), family = binomial(link = "logit"), data = wells)
summary(wellsmodel2)
```

```
##
## Call:
## glm(formula = switch ~ dist * log(arsenic), family = binomial(link = "logit"),
##     data = wells)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        0.491350   0.068119   7.213 5.47e-13 ***
## dist              -0.008735   0.001342  -6.510 7.52e-11 ***
## log(arsenic)       0.983414   0.109694   8.965  < 2e-16 ***
## dist:log(arsenic) -0.002309   0.001826  -1.264    0.206
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3896.8  on 3016  degrees of freedom
## AIC: 3904.8
##
## Number of Fisher Scoring iterations: 4
```

Constant term: logitˆ-1(0.49) = 0.62 is the estimated probability of switching, if dist =log(arsenic) = 0, that is, if distance to nearest safe well and arsenic level are at their averages in the data.

Coefficient for distance: this is the coefficient for distance (on the logit scale) if arsenic level is at its average value. To quickly interpret this on the probability scale, we divide by 4: -0.008735/4 = -0.00218375.Thus, at the mean level of arsenic in the data, each 1 meters of distance corresponds to an approximate 0.218% negative difference in probability of switching.

Coefficient for log(arsenic): this is the coefficient for arsenic level if distance to nearest safe well is at its average value. To quickly interpret this on the probability scale, we divide by 4: 0.983414/4 = 0.2458535. Thus, at the mean level of distance in the data, each additional unit of log(arsenic) corresponds to an approximate 25% positive difference in probability of switching.

Coefficient for the interaction term: If the log(arsenic) is fixed, we divide by 4: -0.002309 / 4 = -0.00057725. Thus, at the mean level of log(arsenic) in the data, each 1 meters of distance corresponds to an approximate 0.057725% negative difference in probability of switching.

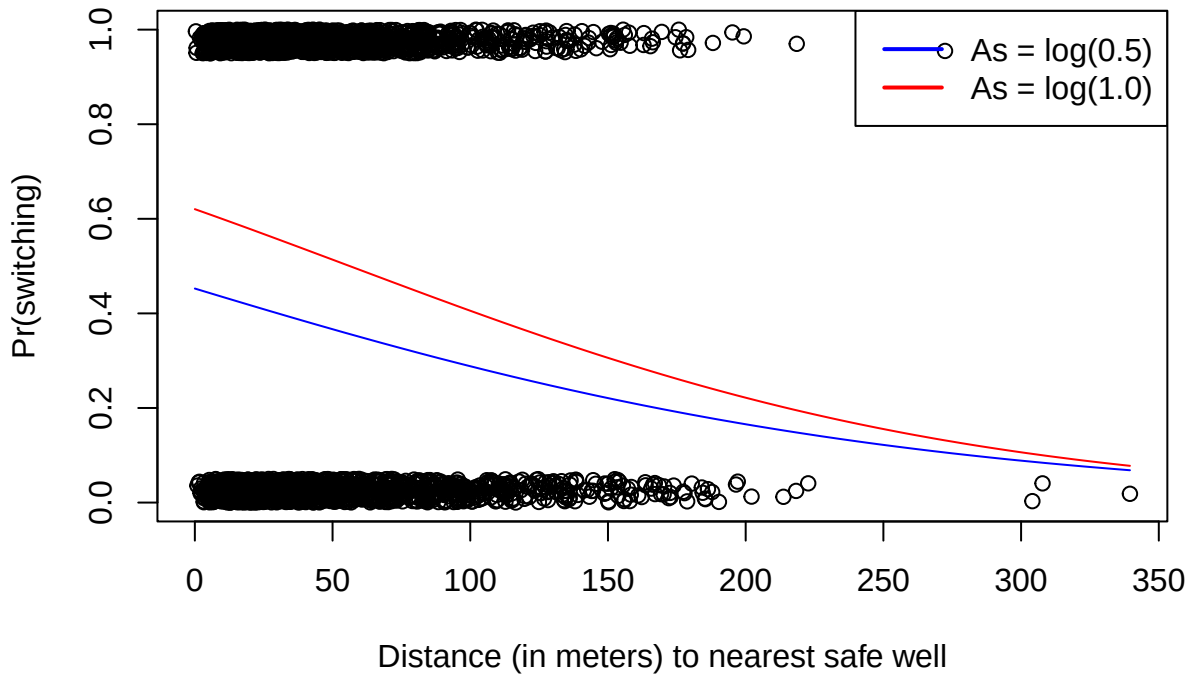
Except for the standard error of the interaction, all the other standard errors indicate that their coefficients are all statistically significant, because the estimates are two standard errors away from zero. The estimate of the interaction is not two standard errors away from zero and so is not statistically significant.

### (b)

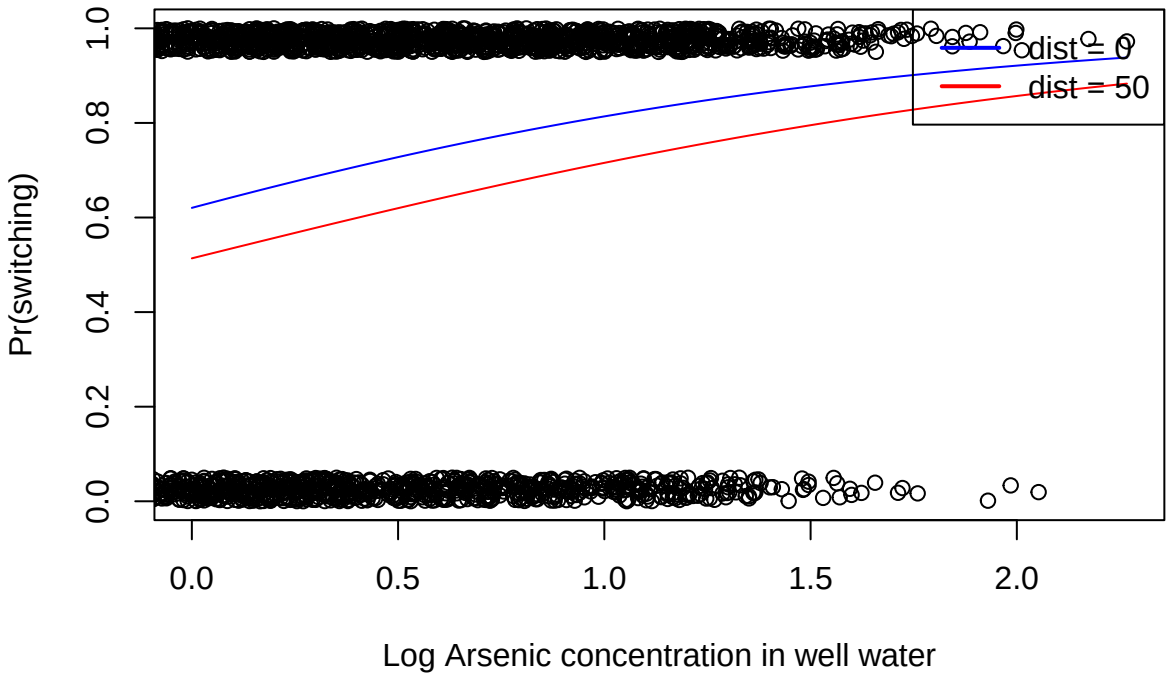
Make graphs as in Figure 14.3 to show the relation between probability of switching, distance, and arsenic level.

```
wells$switch_jitter <- jitter_binary(wells$switch)
plot(wells$dist, wells$switch_jitter,xlim=c(0,max(wells$dist)),
     xlab = "Distance (in meters) to nearest safe well",
```

```
     ylab = "Pr(switching)")
curve(invlogit(cbind(1, x, log(0.5), log(0.5)*x) %*% coef(wellsmodel2)), add=TRUE, col = "blue")
curve(invlogit(cbind(1, x, log(1.0), log(1.0)*x) %*% coef(wellsmodel2)), add=TRUE, col = "red")
legend("topright", legend = c("As = log(0.5)", "As = log(1.0)"),
       col = c("blue", "red"), lty = 1, lwd = 2)
```




```
plot(log(wells$arsenic), wells$switch_jitter,xlim=c(0,max(log(wells$arsenic))),
     xlab = "Log Arsenic concentration in well water",
     ylab = "Pr(switching)")
curve(invlogit(cbind(1, 0, x, 0*x) %*% coef(wellsmodel2)), add=TRUE, col = "blue")
curve(invlogit(cbind(1, 50, x, 50*x) %*% coef(wellsmodel2)), add=TRUE, col = "red")
legend("topright", legend = c("dist = 0", "dist = 50"),
       col = c("blue", "red"), lty = 1, lwd = 2)
```




**(c)**

Following the procedure described in Section 14.4, compute the average predictive differences corresponding to:

i. A comparison of `dist` = 0 to `dist` = 100, with `arsenic` held constant.

ii. A comparison of `dist` = 100 to `dist` = 200, with `arsenic` held constant.

iii. A comparison of `arsenic` = 0.5 to `arsenic` = 1.0, with `dist` held constant.

iv. A comparison of `arsenic` = 1.0 to `arsenic` = 2.0, with `dist` held constant.

Discuss these results.

```
#i
b <- coef(wellsmodel2)
hi <- 100
lo <- 0
deltai <- invlogit(b[1] + b[2]*hi + b[3]*log(wells$arsenic) + b[4]*hi*log(wells$arsenic)) -
invlogit(b[1] + b[2]*lo + b[3]*log(wells$arsenic) + b[4]*lo*log(wells$arsenic))
round(mean(deltai), 2)
```

```
## [1] -0.21
```

```
#ii
hi <- 200
lo <- 100
deltaii <- invlogit(b[1] + b[2]*hi + b[3]*log(wells$arsenic) + b[4]*hi*log(wells$arsenic)) -
invlogit(b[1] + b[2]*lo + b[3]*log(wells$arsenic) + b[4]*lo*log(wells$arsenic))
round(mean(deltaii), 2)
```

```
## [1] -0.21
```

```
#iii
hi <- 1
lo <- 0.5
deltaiii <- invlogit(b[1] + b[2]*wells$dist + b[3]*log(hi) + b[4]*wells$dist*log(hi)) -
invlogit(b[1] + b[2]*wells$dist + b[3]*log(lo) + b[4]*wells$dist*log(lo))
round(mean(deltaiii), 2)
```

```
## [1] 0.15
```

```
#iv
hi <- 2
lo <- 1
deltaiv <- invlogit(b[1] + b[2]*wells$dist + b[3]*log(hi) + b[4]*wells$dist*log(hi)) -
invlogit(b[1] + b[2]*wells$dist + b[3]*log(lo) + b[4]*wells$dist*log(lo))
round(mean(deltaiv), 2)
```

```
## [1] 0.14
```

In both (i) and (ii), this comparison corresponds to a 21% difference in probability of switching.

In (iii), this comparison corresponds to a 15% difference in probability of switching.

In (iv), this comparison corresponds to a 14% difference in probability of switching.