Logistic Regression Modeling

Howard Nguyen

2022-10-02

LOGISTIC REGRESSION MODELING

Logistic regression is a popular statistical modeling technique used to analyze the relationship between a binary dependent variable (also called the response variable) and one or more independent variables (also called predictor or explanatory variables). It is a type of regression analysis that is used for predicting the outcome of a categorical dependent variable based on one or more predictor variables.

The general steps to build a logistic regression model are:

1 - Collect and preprocess the data: Collect the data and preprocess it by cleaning, transforming, and encoding the features. Split the data into training and test sets.

2 - Choose the model: Decide on the model by choosing a dependent variable, selecting the independent variables, and choosing the type of logistic regression model (simple or multiple).

3 - Train the model: Train the model using the training set. This involves estimating the coefficients of the independent variables in the model using maximum likelihood estimation.

4 - Validate the model: Validate the model using the test set. This involves evaluating the accuracy, precision, recall, F1 score, or other appropriate metrics to see how well the model performs on new, unseen data.

5 - Interpret the results: Interpret the results of the model by analyzing the estimated coefficients of the independent variables. These coefficients represent the strength and direction of the relationship between the independent variables and the dependent variable.

6 - Tune the hyperparameters: Tune the hyperparameters of the model, such as the regularization parameter, using cross-validation and grid search to get the best performance.

7 - Deploy the model: Once you have a good model, deploy it in production and use it to make predictions on new data.

Overall, logistic regression is a useful and powerful tool for predicting the probability of an event occurring based on one or more predictor variables. It is widely used in various fields, including healthcare, finance, marketing, and social sciences.

INSTALL AND LOAD PACKAGES

```r
# Install pacman ("package manager") if needed
if (!require("pacman")) install.packages("pacman")
```

```r
# pacman must already be installed; then load contributed
# packages (including pacman) with pacman
pacman::p_load(caret, magrittr, pacman, parallel,
  randomForest, rio, tidyverse, skimr, broom)
```

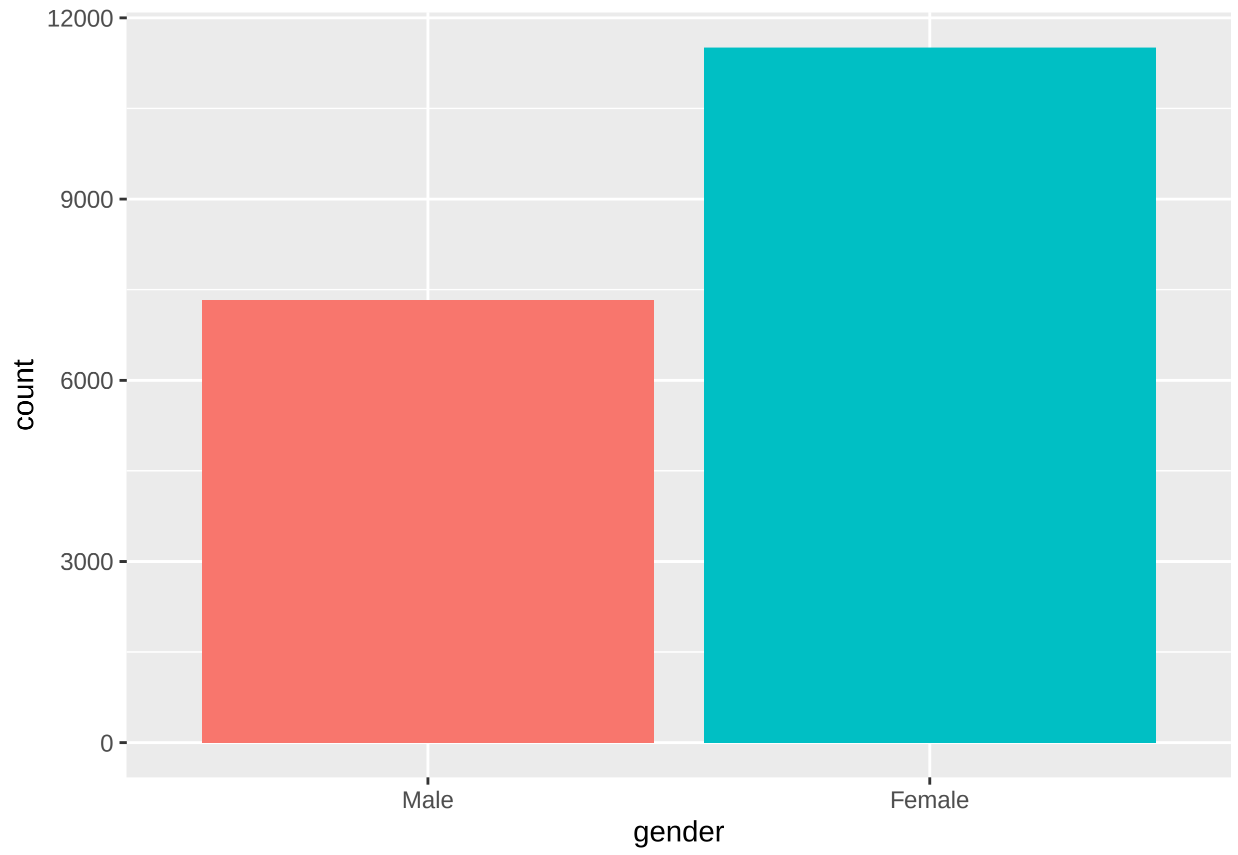
```
# caret: for decision trees
# magrittr: for pipes
# pacman: for loading/unloading packages
# parallel: for parallel processing
# randomForest: for random forests (obviously)
# rio: for importing data
# tidyverse: for so many reasons
# skimr: for summary statistics
# broom: for the tidy() function
```

LOAD AND PREPARE DATA

```
df <- import("../data/b5_df.rds") %>%
  print()
```

```
## # A tibble: 18,837 x 8
##      age gender engnat Extrav Neurot Agree Consc  Open
##    <int> <fct>  <fct>   <dbl>  <dbl> <dbl> <dbl> <dbl>
##  1    53 Male   Yes       4.4    1.1   4.6   4.7   4.3
##  2    46 Female Yes       2.2    3.1   3.5   4.2   2.6
##  3    14 Female No        3.5    4.6   3.8   4.9   4.5
##  4    19 Female No        2.2    4.3   3.7   2.6   4.1
##  5    25 Female No        3.4    3     4.4   3.4   3.4
##  6    31 Female Yes       1.6    2.4   3.6   3.1   3.3
##  7    20 Female Yes       4.6    2.1   4.5   2.8   4.1
##  8    23 Male   No        3.9    1.5   4.1   4.4   4.2
##  9    39 Female Yes       4.5    3.5   4.9   4.1   4.1
## 10    18 Female Yes       1.5    3.7   3.5   4     4.1
## # ... with 18,827 more rows
```

```
# Bar chart of "gender"
df %>%
  ggplot() +
  geom_bar(
    aes(
      x    = gender,  # Variable to chart
      fill = gender   # Color bars by variable
    )
  ) +
  theme(legend.position = "none")
```

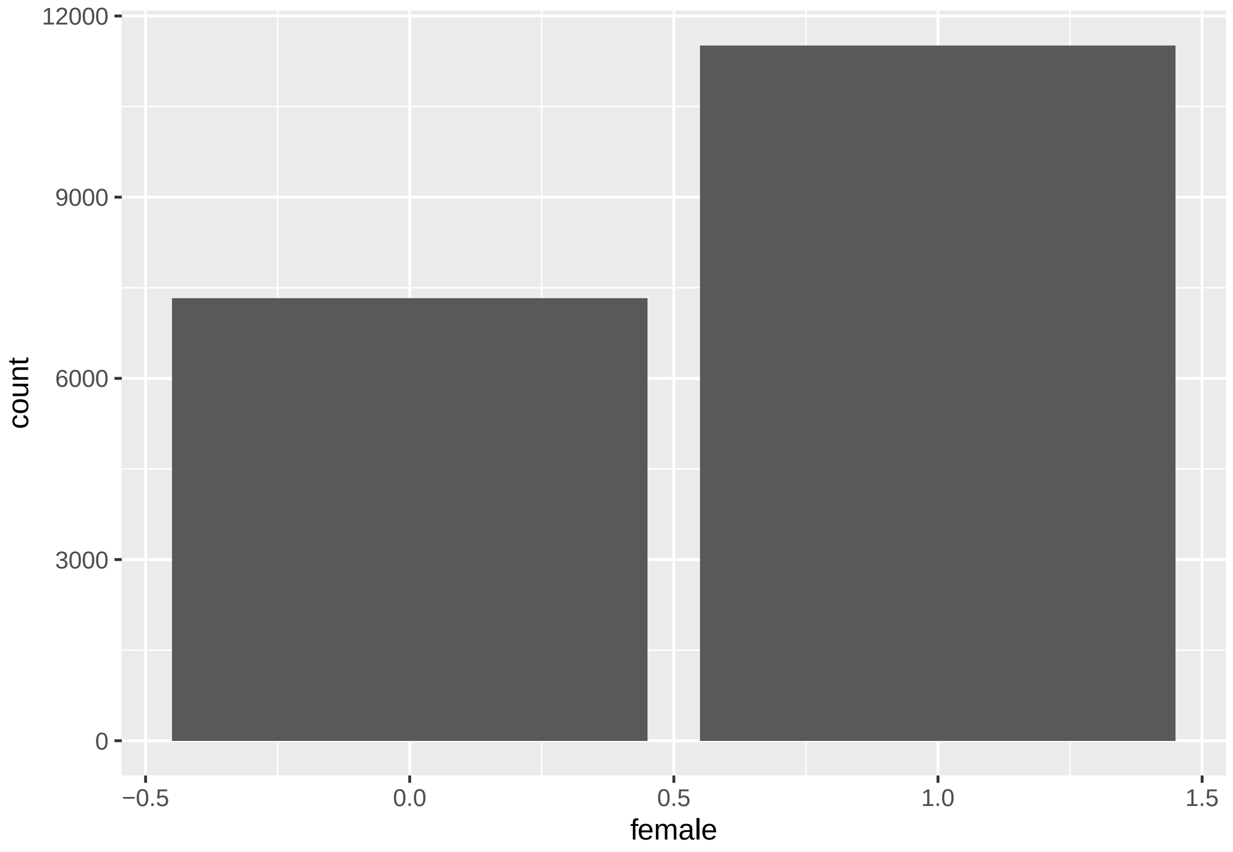


```
# Logistic regression needs a 0/1 outcome variable, so
# we'll create a variable "female" where "Female" gets a 1
# (for "True") and "Male" gets a 0 (for "False")
```

```
df %<>%
  mutate(
    female = ifelse(       # Create new variable "female"
      gender == "Female",  # Test
      1,                   # Value if true
      0                    # Value if false
    )
  ) %>%
  select(gender, female, Extrav:Open) %>%
  print()
```

```
## # A tibble: 18,837 x 7
##    gender female Extrav Neurot Agree Consc  Open
##    <fct>   <dbl>  <dbl>  <dbl> <dbl> <dbl> <dbl>
##  1 Male        0    4.4    1.1   4.6   4.7   4.3
##  2 Female      1    2.2    3.1   3.5   4.2   2.6
##  3 Female      1    3.5    4.6   3.8   4.9   4.5
##  4 Female      1    2.2    4.3   3.7   2.6   4.1
##  5 Female      1    3.4    3     4.4   3.4   3.4
##  6 Female      1    1.6    2.4   3.6   3.1   3.3
##  7 Female      1    4.6    2.1   4.5   2.8   4.1
##  8 Male        0     3.9    1.5   4.1   4.4   4.2
##  9 Female      1     4.5    3.5   4.9   4.1   4.1
## 10 Female      1     1.5    3.7   3.5   4     4.1
## # ... with 18,827 more rows
```

```
# Bar chart of "female"
df %>%
  ggplot() +
  geom_bar(aes(x = female)) +
  theme(legend.position = "none")
```



EXPLORE DATA

```
# Explore data (wide output)
df %>% skim()
```

Table 1: Data summary

Name	Piped data
Number of rows	18837
Number of columns	7

Table 1: Data summary

Column type frequency:	
factor	1
numeric	6
Group variables	None

Variable type: factor

skim_variable	n_missing	complete_rate	ordered	n_unique	top_counts
gender	0	1	FALSE	2	Fem: 11511, Mal: 7326

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
female	0	1	0.61	0.49	0	0.0	1.0	1.0	1	
Extrav	0	1	3.02	0.92	1	2.3	3.0	3.7	5	
Neurot	0	1	3.09	0.86	1	2.5	3.1	3.7	5	
Agree	0	1	3.85	0.71	1	3.4	3.9	4.4	5	
Consc	0	1	3.35	0.73	1	2.8	3.4	3.9	5	
Open	0	1	3.91	0.62	1	3.5	4.0	4.4	5	

BINOMIAL LOGISTIC REGRESSION

```
# Compute model
fit <- glm(
  female ~ Extrav + Neurot + Agree + Consc + Open,
  data = df,
  family = "binomial"
  )
```

```
# Summarize regression model
fit %>% summary()  # Standard output
```

```
##
## Call:
## glm(formula = female ~ Extrav + Neurot + Agree + Consc + Open,
##     family = "binomial", data = df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3225  -1.1892   0.7140   0.9513   2.1995
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.00398    0.16897 -17.779  < 2e-16 ***
## Extrav       0.10480    0.01893   5.537 3.07e-08 ***
## Neurot       0.53715    0.02025  26.528  < 2e-16 ***
## Agree        0.70951    0.02460  28.840  < 2e-16 ***
## Consc        0.14596    0.02276   6.414 1.42e-10 ***
## Open        -0.43503    0.02639 -16.487  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 25176  on 18836  degrees of freedom
## Residual deviance: 23259  on 18831  degrees of freedom
## AIC: 23271
##
## Number of Fisher Scoring iterations: 4
```

```
fit %>% tidy()     # Tidy output
```

```
## # A tibble: 6 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)   -3.00     0.169     -17.8  1.04e- 70
## 2 Extrav         0.105    0.0189      5.54 3.07e-  8
## 3 Neurot         0.537    0.0202     26.5  4.60e-155
## 4 Agree          0.710    0.0246     28.8  6.83e-183
## 5 Consc          0.146    0.0228      6.41 1.42e- 10
## 6 Open          -0.435    0.0264    -16.5  4.53e- 61
```

```
# Confidence intervals for coefficients
fit %>% confint()
```

```
## Waiting for profiling to be done...
```

```
##                   2.5 %     97.5 %
## (Intercept) -3.33579656 -2.6734258
## Extrav       0.06771742  0.1419062
## Neurot       0.49756349  0.5769390
## Agree        0.66144194  0.7578849
## Consc        0.10138320  0.1905959
## Open        -0.48685791 -0.3834214
```

PREDICTED VALUES

```
# Predicted probability are in "fitted values"
#fit %>% View()
```

```
# See the first few predicted values
predict(fit, type = 'response') %>% head()
```

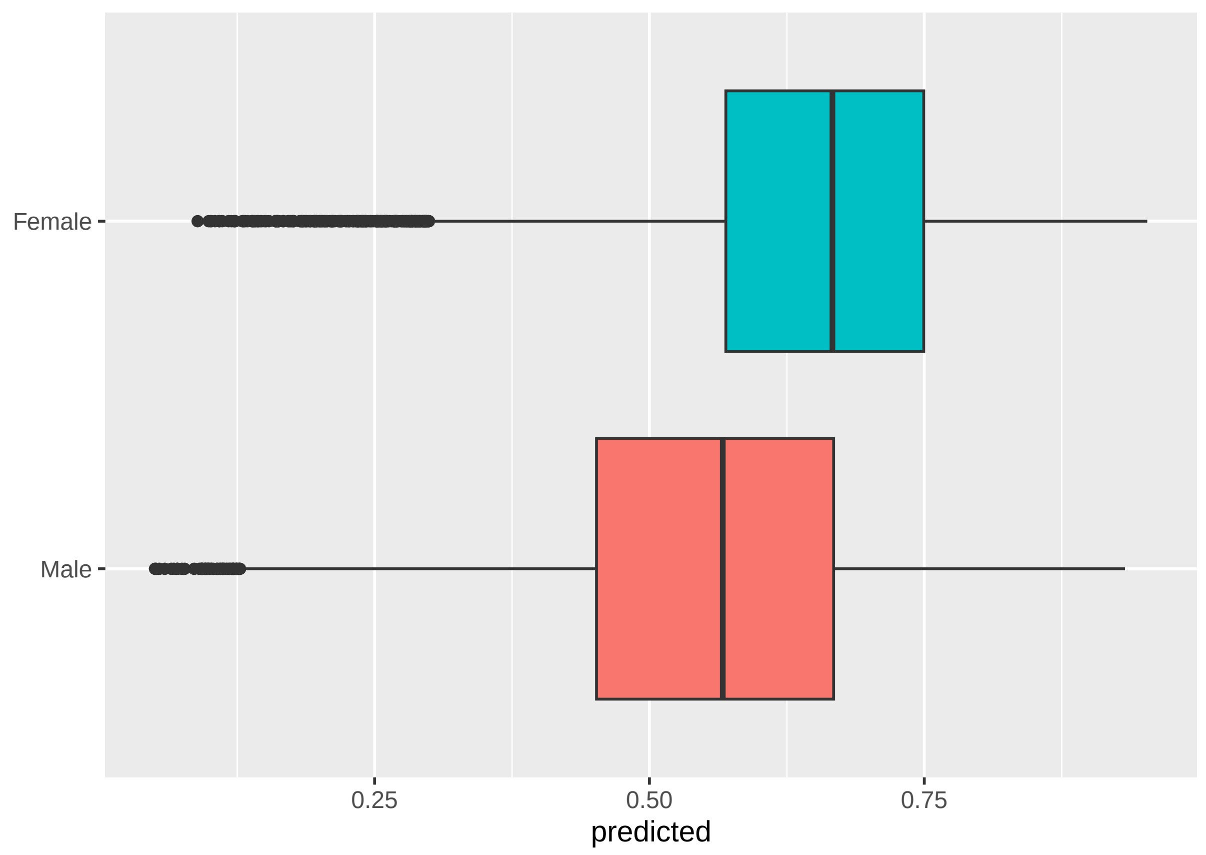
```
##         1         2         3         4         5         6
## 0.5317369 0.7020285 0.7837063 0.6807805 0.7507802 0.5059898
```

```
# Add predicted values to df
df %<>%
  mutate(
    predicted = predict(fit, type = 'response'),
    pred_gender = ifelse(  # Create variable "pred_gender"
      predicted > 0.5,     # Test if predicted values > 0.5
      "P_Female",          # If true, predict female
      "P_Male"             # If false, predict male
    )
  ) %>%
  select(
    gender,
    female,
    predicted,
    pred_gender,
    everything()
  ) %>%
  print()
```

```
## # A tibble: 18,837 x 9
##    gender female predicted pred_gender Extrav Neurot Agree Consc  Open
##    <fct>   <dbl>     <dbl> <chr>        <dbl>  <dbl> <dbl> <dbl> <dbl>
##  1 Male        0     0.532 P_Female       4.4    1.1   4.6   4.7   4.3
##  2 Female      1     0.702 P_Female       2.2    3.1   3.5   4.2   2.6
##  3 Female      1     0.784 P_Female       3.5    4.6   3.8   4.9   4.5
##  4 Female      1     0.681 P_Female       2.2    4.3   3.7   2.6   4.1
##  5 Female      1     0.751 P_Female       3.4    3     4.4   3.4   3.4
##  6 Female      1     0.506 P_Female       1.6    2.4   3.6   3.1   3.3
##  7 Female      1     0.604 P_Female       4.6    2.1   4.5   2.8   4.1
##  8 Male        0     0.484 P_Male         3.9    1.5   4.1   4.4   4.2
##  9 Female      1     0.837 P_Female       4.5    3.5   4.9   4.1   4.1
## 10 Female      1     0.604 P_Female       1.5    3.7   3.5   4     4.1
## # ... with 18,827 more rows
```

VISUALIZE PROBABILITIES

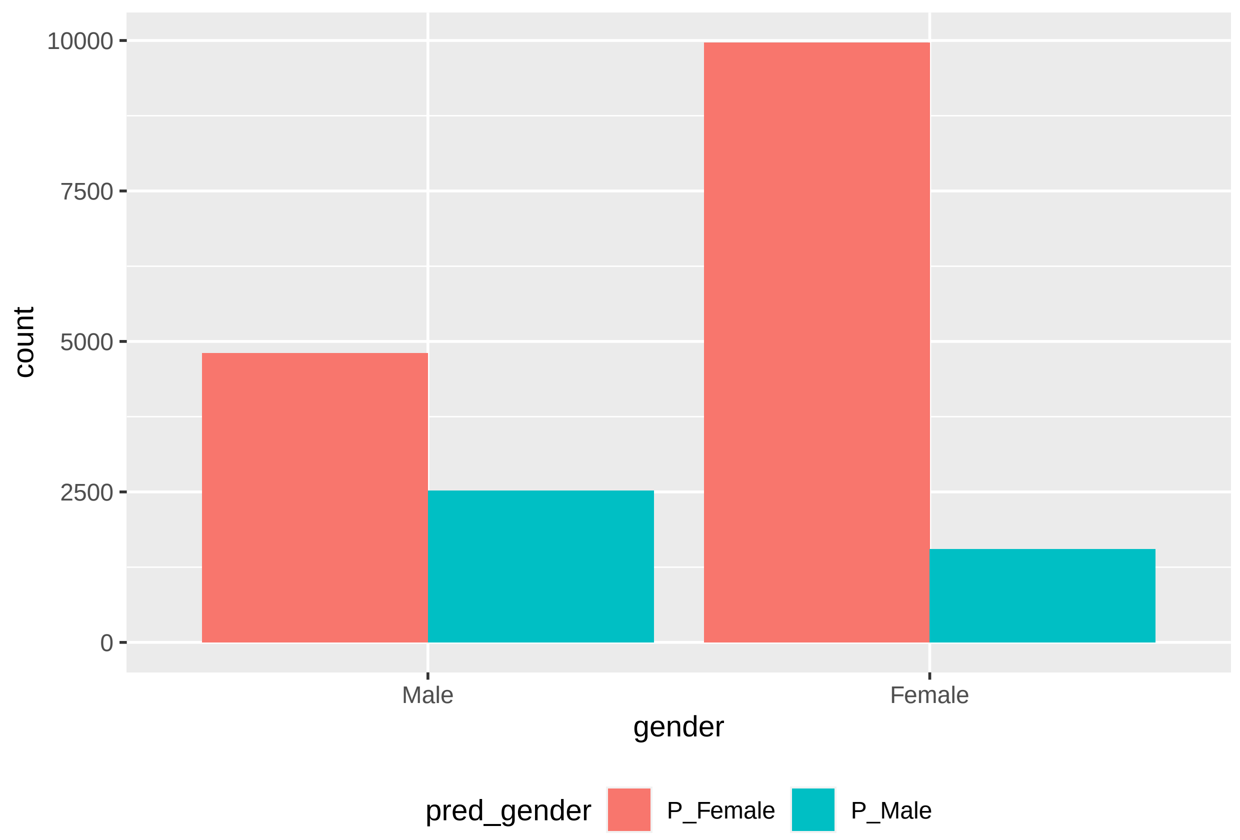
```
# Boxplots of probabilities by actual values
df %>%
  ggplot(aes(x = gender,
    y = predicted,
    fill = gender)) +
  geom_boxplot() +
  coord_flip() +
  xlab("") +
  theme(legend.position = "none")
```



```
# Confusion matrix
df %$%                         # Exposition pipe
  table(gender, pred_gender)  # table(rows, columns)
```

```
##          pred_gender
## gender    P_Female P_Male
##   Male        4806   2520
##   Female      9965   1546
```

```
# Side-by-side bar chart for confusion matrix
df %>%
  ggplot(aes(gender, fill = pred_gender)) +
  geom_bar(position = position_dodge()) +
  theme(legend.position = "bottom")
```

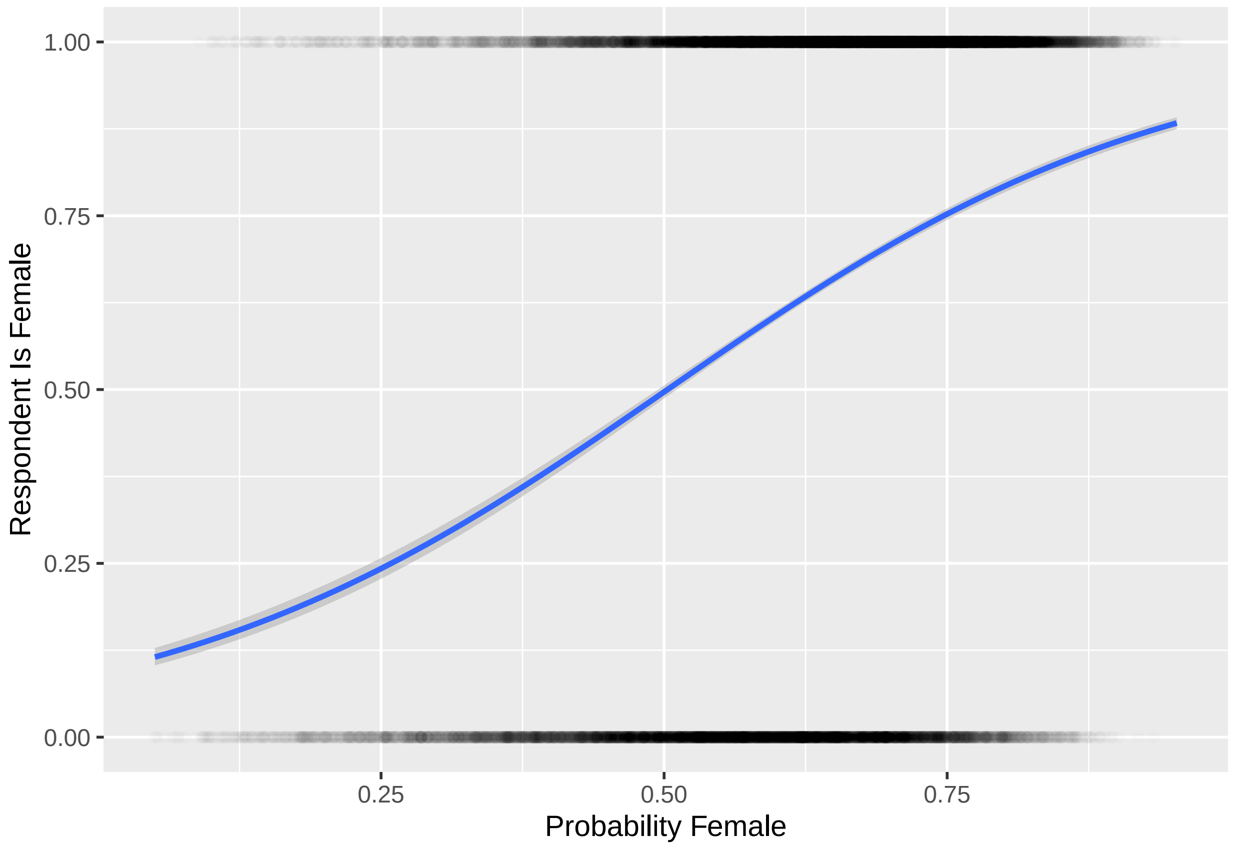


```
# Row percentages for confusion matrix
df %$%
  table(gender, pred_gender) %>%
  prop.table(1) %>%  # 1 is for row percentages
  round(2) %>%       # Round to two decimal places
  `*`(100)           # Multiply by 100 to read as percent
```

```
##         pred_gender
## gender   P_Female P_Male
##   Male         66     34
##   Female       87     13
```

```
# Plot logistic curve
df %>%
  ggplot(
    aes(
      x = predicted,
      y = female
    )
  ) +
  geom_point(alpha = .01) +
  stat_smooth(
    method = "glm",
    method.args = list(family = binomial)
  ) +
  xlab("Probability Female") +
  ylab("Respondent Is Female")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



The End!