---

output:

pdf\_document: default

html\_document: default

---

# Logistic Regression For Binary Classification

## Introduction

For individuals who have studied cell biology or biochemistry, logistic regression may be familiar as dose-response curves, enzyme kinetic curves, sigmoidal curves, median lethal dose curve (LD-50) or even an exponential growth curve given limited resources.

However, in the context of predictive modeling, Logistic Regression is used as a binary classifier that can toggle between logical values of zero or one.

Logistic regression derives its name from its similarity to linear regression, as we shall see below. In Logistic Regression, the dependent variable ($y$) is NOT calculated, as is found with Linear Regression. Instead, Logistic Regression is used for classification between two states. Instead, the term logistic should be a reminder that the function's output is logical values, 0's and 1's.

$$f(x) = ~~ \left \{ \begin{matrix} 0 ~~for~~ x < 0 \\ 1 ~~for~~ x \geq 0 \end{matrix} \right.$$

Using Logistic Regression, we may now calculate the presence or absence of a product or quality that we wish to model.

As we can see in the figure below, the function's domain ($x$) is ($-\infty$ to $\infty$), whereby its range is largely zero or one. In our simple case, the \*\*decision boundary\*\* at $x ~=$ 0, our system changes from \*zero\*, absence, to \*one\*, the presence of quality or item. In the Logistic/Sigmoidal Curve figure below, the \*\*decision boundary\*\* is denoted by the \*red dotted line\*.

---

```{r echo=FALSE, fig.align="center"}

x <- seq(-6, 6, 0.05)

y <- 1 / (1 + exp(-x))

plot(x, y,

type = "l",

main = "Logistic Curve",

ylim = c(-0.3, 1.3))

abline(h = 0, col = "blue")

abline(v = 0, col = "red", lty = 3, lwd = 3)

abline(h = 1, col = "blue", lty = 3)

text(-3.5, 0.48, cex = 1.75, "if x < 0 then y = 0")

text(3.5, 0.48, cex = 1.75, "if x >= 0 then y = 1")

```

\pagebreak

The logistic growth curve is commonly denoted by:

$$f(x) ~=~ \frac{M}{1 + Ae^{-r(x - x\_0)}}$$

where $M$ is the curve's maximum value, $r$ is the maximum growth rate (also called the Malthusian parameter[^41]), $x\_0$ is the midpoint of the curve, $A$ is the number of times that the initial population must double to reach $M$.[^42]

[^41]:https://en.wikipedia.org/wiki/Malthusian\_growth\_model

[^42]:https://en.wikipedia.org/wiki/Logistic\_function

In the specific case of \*Logistic Regression for Binary Classification\* where we have a probability between 0 and 1, $M$, and $A$ take on the value one.

---

Since the logistic equation is exponential, it is easier to work with the formula in terms of its odds or \*\*log-odds\*\*. Odds are the probabilities of success over failure denoted as $\Large \frac{p}{1-p}$ or more precisely log-odds as $\Large ln \left (\frac{p}{1-p} \right )$.

Simply by using log-odds, logistic regression may be more easily expressed as a set of linear equations in x.[^43] Hence we can now go from linear regression to logistic regression.

[^43]:http://juangabrielgomila.com/en/logistic-regression-derivation/

$$ln ~ \left ( \frac{Pr(y\_i ~=~ 1|x\_i)}{Pr(y\_i ~=~ 0|x\_i)} \right ) =~ \beta\_0 + \beta\_1 x\_1 +~ ... ~+ \beta\_{n} x\_{n}$$

Substitute ($p$ for $Pr(y\_i ~=~ 1|x\_i)$) and ($1-p$ for $Pr(y\_i ~=~ 0|x\_i)$) and change notation to summation on the right hand side:

$$ln \left( \frac {p}{1-p} \right) =~ \sum\_i^{k} \beta\_i x\_i$$

Eliminate the natural log by taking the exponent on both sides:

$$\large \frac {p}{1-p} =~ exp \left ( \sum\_i^{k} \beta\_i x\_i \right )$$

Substitute $u = \sum\_i^{k} \beta\_i x\_i$:

$$\large \frac {p}{1-p} =~ e^u$$

Rearrange to solve for $\large p$:

$$\large p(u) ~=~ \frac{e^u}{1 + e^u}$$

Take the derivative of both sides using quotient rule:

$$p'(u) ~=~ \frac {(e^u)(1 + e^u) - (e^u)(e^u)}{(1 + e^u)^2}$$

\pagebreak

Simplify:

$$p'(u) ~=~ \frac {e^u}{(1 + e^u)^2}$$

Separate out to produce two fractions:

$$p'(u) ~=~ \left ( \frac {e^u}{1 + e^u} \right ) \cdot \left ( \frac{1}{1 + e^u} \right )$$

Substitute our previous success and failure variables back into place:

$$p'(u) ~=~ p(u) \cdot ( 1 - p(u))$$

Now we can calculate the probabilities as well as the values for any given x value.

## Load libraries and protein dataset

```{r message=FALSE, warning=FALSE}

# Load Libraries

Libraries <- c("doMC", "knitr", "readr", "tidyverse", "caret")

for (p in Libraries) { # Install Library if not present

if (!require(p, character.only = TRUE)) { install.packages(p) }

library(p, character.only = TRUE)

}

```

```{r include=FALSE}

opts\_chunk$set(cache = TRUE, warning = FALSE, message = FALSE)

```

```{r}

# Import relevant data

c\_m\_TRANSFORMED <- read\_csv("../00-data/02-aac\_dpc\_values/c\_m\_TRANSFORMED.csv",

col\_types = cols(Class = col\_factor(levels = c("0", "1")),

PID = col\_skip(),

TotalAA = col\_skip()))

```

```{r}

# Partition data into training and testing sets

set.seed(1000)

index <- createDataPartition(c\_m\_TRANSFORMED$Class, p = 0.8, list = FALSE)

training\_set.1 <- c\_m\_TRANSFORMED[index, ]

```

The `test.set.1` and `Class.test` data sets are not produced since the Logit run with 20 features was not deemed useful. The reason for its dismissal was that is contained extraneous features.

## Logit Training #1 using 20 Features

- The first training (\*\*test\*\*) is to determine if all 20 features (amino acids) are necessary for our logistic regression model.

```{r, cache = TRUE}

set.seed(1000)

registerDoMC(cores = 3) # Start multi-processor mode

start\_time <- Sys.time() # Start timer

# Create model, 10X fold CV repeated 5X

tcontrol <- trainControl(method = "repeatedcv",

number = 10,

repeats = 5)

model\_obj.1 <- train(Class ~ .,

data = training\_set.1,

trControl = tcontrol,

method = "glm",

family = "binomial")

end\_time <- Sys.time() # End timer

end\_time - start\_time # Display time

registerDoSEQ() # Stop multi-processor mode

```

## Logit Results #1

```{r}

summary(model\_obj.1)

```

The Akaike information criterion (AIC)[^44] for model #1 is 699.72. This will be used later to compare the models generated to rate their ability to utilize the features best.

- The list of probabilities for the estimates leaves us with only \*\*9 important features\*\* to try re-modeling, R, H, P, C, E, Y, T, D, G.

[^44]:https://en.wikipedia.org/wiki/Akaike\_information\_criterion

## Logit Training #2 using 9 Features

- This test uses \*\*ONLY\*\* 9 features: (R, H, P, C, E, Y, T, D, G)

```{r}

# Data import & handling

c\_m\_9aa <- read\_csv("../00-data/02-aac\_dpc\_values/c\_m\_TRANSFORMED.csv",

col\_types = cols(Class = col\_factor(levels = c("0", "1")),

A = col\_skip(),

F = col\_skip(),

I = col\_skip(),

K = col\_skip(),

L = col\_skip(),

M = col\_skip(),

N = col\_skip(),

PID = col\_skip(),

Q = col\_skip(),

V = col\_skip(),

S = col\_skip(),

TotalAA = col\_skip(),

W = col\_skip()))

```

```{r}

# Partition data into training and testing sets

set.seed(1000)

index <- createDataPartition(c\_m\_9aa$Class, p = 0.8, list = FALSE)

training\_set.2 <- c\_m\_9aa[ index, ]

test\_set.2 <- c\_m\_9aa[-index, ]

Class\_test.2 <- as.factor(test\_set.2$Class)

```

### Logit Training #2 with 9 Features

```{r, cache = TRUE}

set.seed(1000)

registerDoMC(cores = 3) # Start multi-core

start\_time <- Sys.time() # Start timer

# Create model, 10X fold CV repeated 5X

fitControl <- trainControl(method = "repeatedcv",

number = 10,

repeats = 5,

savePredictions = "final") # IMPORTANT: Saves predictions

model\_obj.2 <- train(Class ~ .,

data = training\_set.2,

trControl = fitControl,

method = "glm",

family = "binomial")

end\_time <- Sys.time() # End timer

end\_time - start\_time # Display time

registerDoSEQ() # Stop multi-core

```

## Logit Summary #2

```{r}

summary(model\_obj.2)

```

\pagebreak

## Logit Confusion Matrix #2

```{r, cache=TRUE}

Predicted\_test\_vals <- predict(model\_obj.2, test\_set.2[, -1])

confusionMatrix(Predicted\_test\_vals, Class\_test.2, positive = "1")

```

- The Akaike information criterion (AIC) for model #2 is 708.96. This will be used later to compare the models generated to rate their ability to utilize the features best.

- The number of unique false-positives and false-negatives is 26.

## Obtain List of False Positives & False Negatives

```{r}

fp\_fn\_logit <- model\_obj.2 %>% pluck("pred") %>% dplyr::filter(obs != pred)

# Write CSV in R

write.table(fp\_fn\_logit,

file = "../00-data/03-ml\_results/fp\_fn\_logit.csv",

row.names = FALSE,

na = "",

col.names = TRUE,

sep = ",")

nrow(fp\_fn\_logit) ## NOTE: NOT UNIQUE NOR SORTED

```

- The logistic regression second test produced 536 protein samples, which are either false-positives or false-negatives. The list of 536 proteins may have duplicates. Therefore they are NOT UNIQUE NOR SORTED.

## Conclusion

Logit is easy to implement and understand and can be used for parameter importance measurements.

Considering the Table Logit Models, below, it is clear that model #2 with nine features best describes the better of the two models.

Akaike Information Criterion:[^45]

[^45]:https://en.wikipedia.org/wiki/Akaike\_information\_criterion

$$AIC ~=~ 2 K ~-~ ln (\widehat{L})$$

Where $ln (\widehat{L})$ is the log-likelihood, $K$ is the number of parameters.

| Model # | Features | AIC |

| :-----: | :------: | :----: |

| 1 | 20 | 699.72 |

| 2 | 9 | 708.96 |

Logit is a common machine learning method. It is easy to understand and explain. This supervised binary classification method is very useful for determining the importance of the features which can be applied. As we saw in Model#1, there were 11 features that had probabilities of the estimates used above the 5% threshold cut-off. In Model#2, only nine features were used to describe the model, and the AIC increased by 9.24.

The nine features which best described the logistic regression model were R, H, P, C, E, Y, T, D, G. If we compare this to the Boruta test carried out in the EDA, we find the overlap interesting.

\*\*Test Model / Order of Importance (From Left To Right)\*\*

| Test Model | | | | | | | | | | | | | | | | | | | | |

| :--------: |:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|:--|

| Order | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11| 12| 13| 14| 15| 16| 17| 18| 19| 20|

| Boruta | R | H | P | K | C | E | Y | T | S | A | V | U | I | F | D | G | N | L | M | Q |

| Logit | R | H | P | . | C | E | Y | T | . | . | . | . | . | . | D | G | . | . | . | . |

The first 7 out of 8 amino acid features are seen in the proper order, as described by the Boruta Random Forest model. This is confirmation that Logit can pick up the importance of features similar to Boruta.

Logit produced 536 proteins, which are false-negatives or false-positives. It should be noted that the 536 are NOT UNIQUE NOR SORTED. The number of FN/FP from the confusion matrix is 26. These proteins will be investigated further in the Outliers section, which compares these FN/FP proteins to the PCA outliers.