Logistic regression

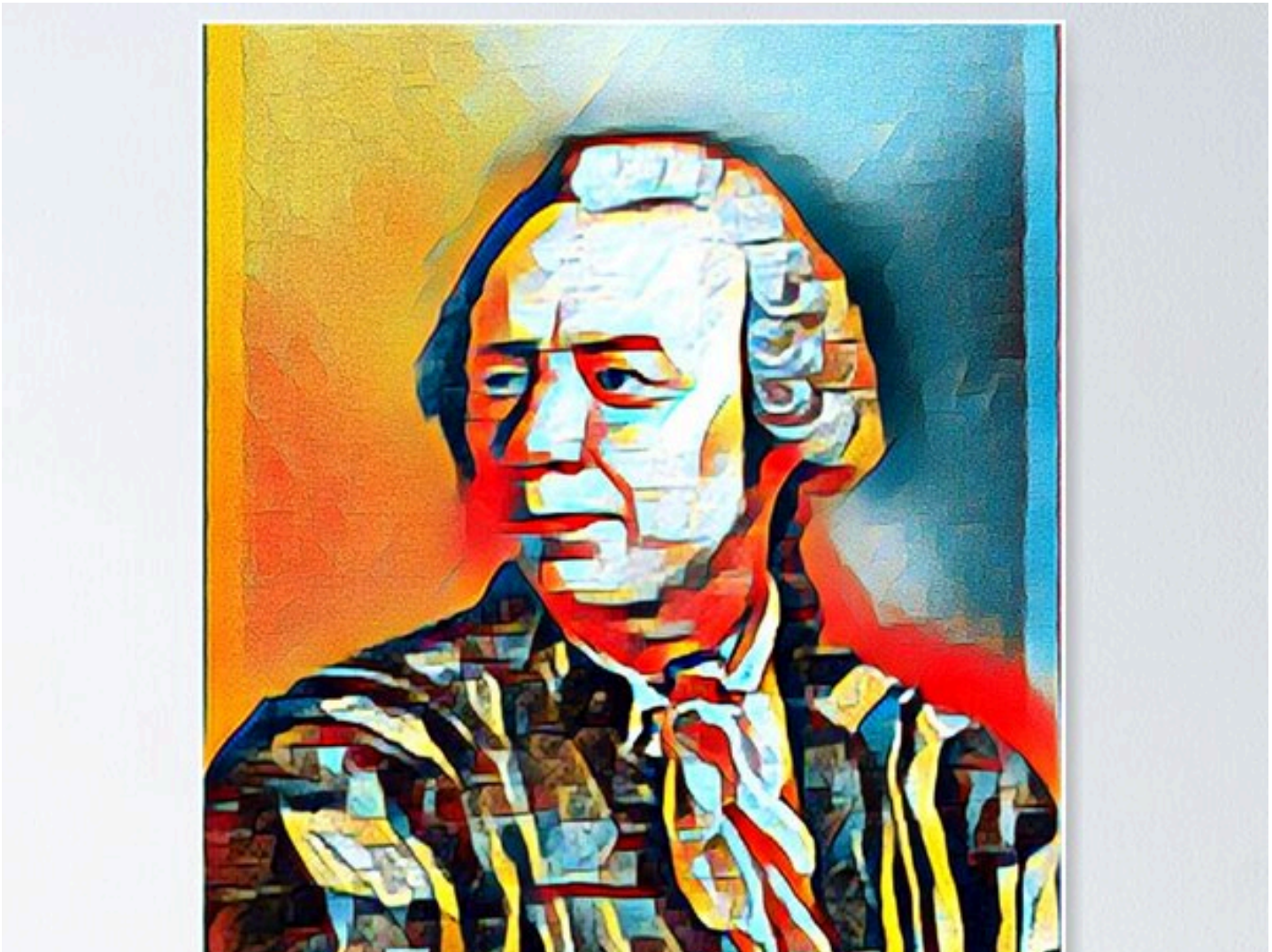
Mabel Carabali

Part 1 & Part 2

EBOH, McGill University

2023/02/08 (updated: 2024-10-07)

Have you met Euler?



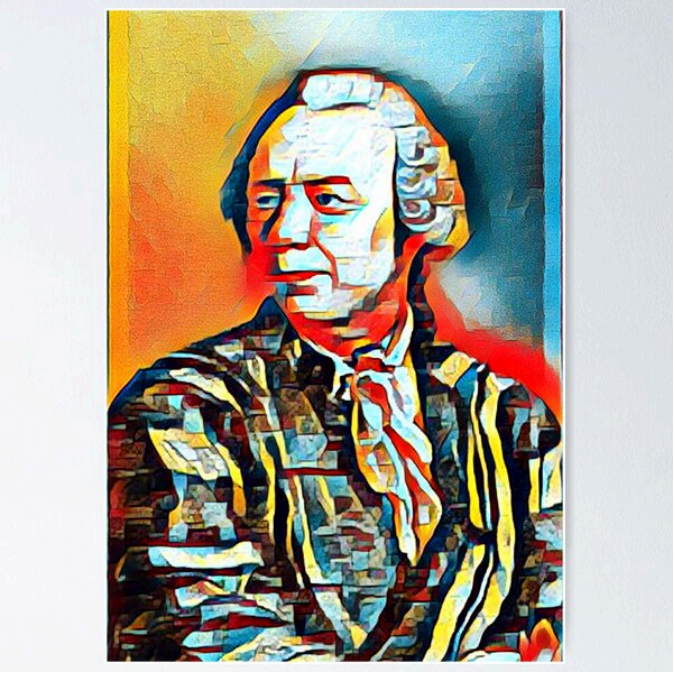
Expected competencies

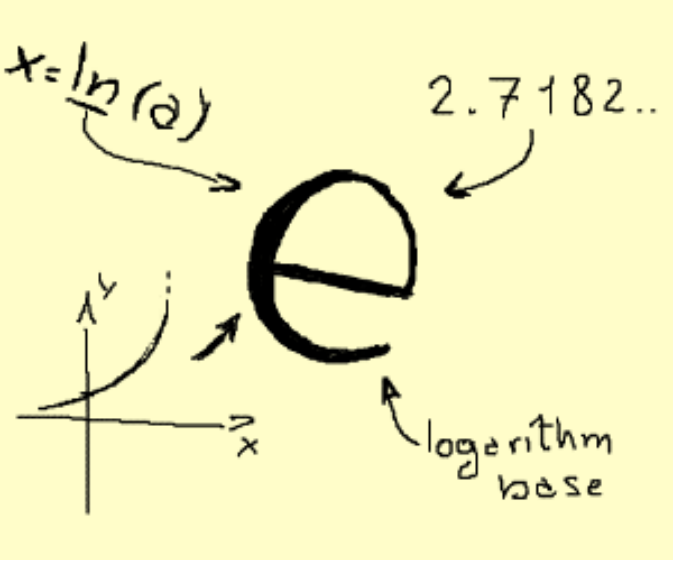
- Knows when logistic regression analysis is needed (Questions and type of data).
- Can describe the logistic regression model, (assumptions & implications).
- Knows the relationship between ORs and Logistic regression coefficients.
- Knows how a statistical package is used to fit a logistic regression model to continuous and categorical predictors.
    - Interpret logistic regression model output, and assess model's fit.

Objectives

- To review core concepts of logistic regression
- Provide tools to improve the inference when assessing binary outcomes
- Illustrate advantages of Bayesian inference.

Have you met Euler?





Eulers' number a.k.a.' *e* ' is a numerical constant 2.7182818...

- Described under logarithm concepts & it's basically the base of the natural logarithm.
- Mostly used to represent the non-linear increase or decrease of a function.
- In R we use the expression `exp()`
- e^1 = exp(1) =2.7182818; e^0 = exp(0) = 1 & e^∞ = 0

Basic math - review of logarithms

Laws of Exponents:

$$e^{\alpha} \cdot e^{\beta} = e^{\alpha+\beta}$$

$$(e^{\alpha})^{\beta} = e^{\alpha\beta}$$

where α and β can be any real numbers and be any positive real number

Logarithms:

$$e^{x} = N$$

The value of x which solves this equation is written as:

$$x = log_e N$$

The right-hand side is expressed as "log to the base e of N"

Basic math

Other manipulations

$$log_e M + log_e N = log_e(MN)$$

$$log_e M - log_e N = log_e(M/N)$$

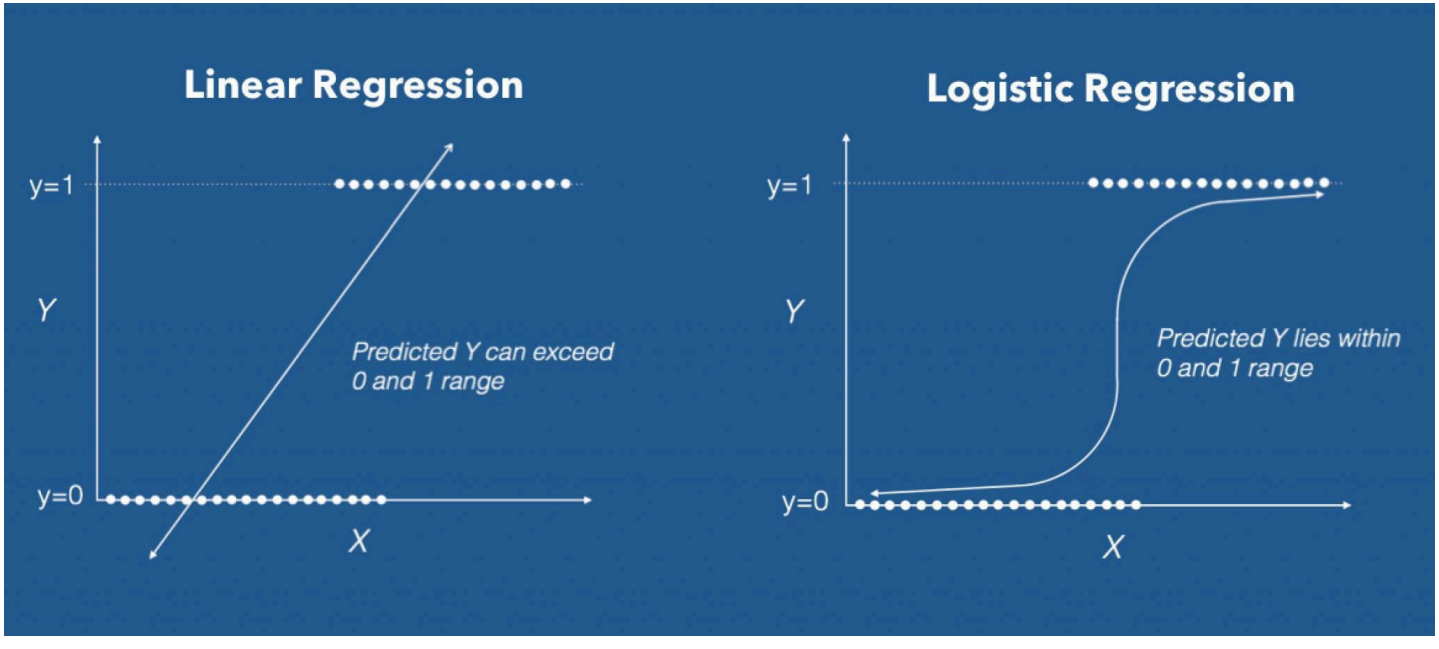
$$log_e(M^N) = N \cdot log_e(M)$$

The purpose of these early logarithmic tables was to take advantage of the law of exponents in order to avoid messy multiplications in the era before electronic calculators.

- Any positive base is possible for logarithms. For example, the exponential equation $4^3 = 64$ can be written in terms of a logarithm as $log_4(64) = 3$
- In practice, the only bases that actually get used to any extent are 10 (“common logarithms”, written as `log10(x)`) and e = 2.71828 ("natural logarithm", `log(x)`) ← Euler's

$$e = \sum_{n=0}^{\infty} \frac{1}{n!} = 1 + \frac{1}{1} + \frac{1}{1 \cdot 2} + \frac{1}{1 \cdot 2 \cdot 3} + \cdots$$

Predicting categorical outcome data



Spam filters

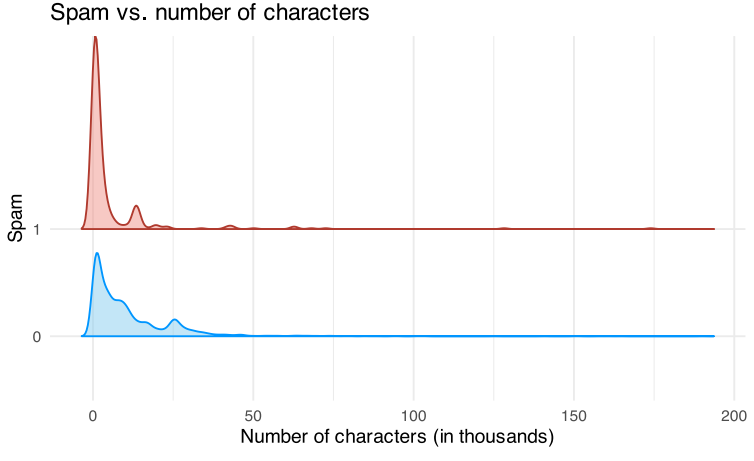
- Data from 3921 emails and 21 variables on them contained in `openintro` package
- **Outcome:** whether the email is spam or not
- **Predictors:** number of characters, whether the email had "Re:" in the subject, time at which email was sent, number of times the word "inherit" shows up in the email, etc. Can use `glimpse(email)` or `srt(email) to see the dataset

Characteristic	N = 3,921[1]
spam	
0	3,554 (91%)
1	367 (9.4%)
num_char	6 (1, 14)
re_subj	
0	2,896 (74%)
1	1,025 (26%)
to_multiple	
0	3,301 (84%)
1	620 (16%)
urgent_subj	
0	3,914 (100%)
1	7 (0.2%)
winner	64 (1.6%)
attach	0.00 (0.00, 0.00)

Spam filters

Would you expect longer or shorter emails to be spam?

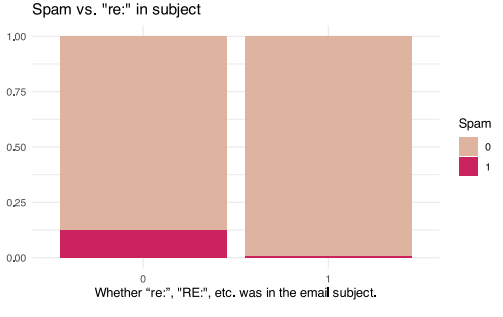
```
plotspam<- email %>%
  ggplot(aes(x = num_char, y = spam,  fill
  geom_density_ridges2(alpha = 0.5) +
  labs(x = "Number of characters (in thousa
    y = "Spam", title = "Spam vs. number of
  guides(color = FALSE, fill = FALSE) +
  scale_fill_manual(values = c("#89CFF0", "
  scale_color_manual(values = c("#0096FF",
plotspam
```




```
t<-email %>% group_by(spam) %>%
  summarise(mean_num_char = mean(num_char))
kable(t)
```

spam	mean_num_char
0	11.250517
1	5.439204

--



Modelling spam

- Both number of characters and whether the message has "re:" in the subject might be related to whether the email is spam.

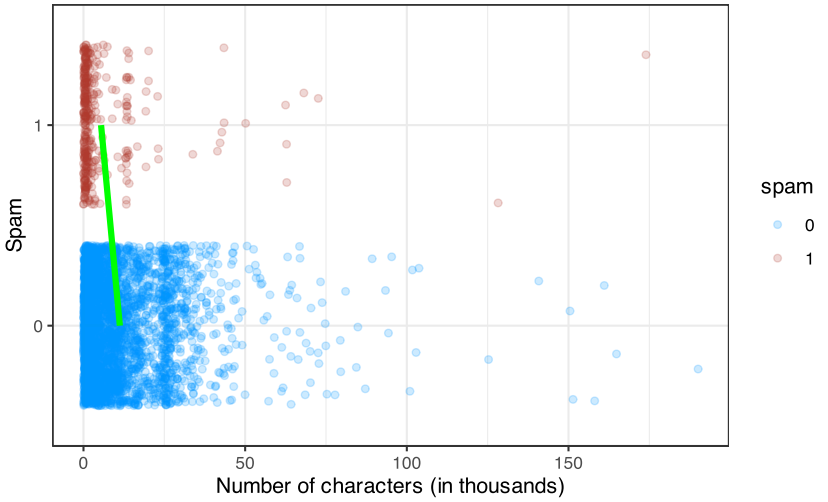
How do we come up with a model that will let us explore this relationship?

- For simplicity, we'll focus on the number of characters (`num_char`) as predictor, but the model we describe can be expanded to take multiple predictors as well.

Modelling spam

Let's first look at the data

```
means <- email %>%
  group_by(spam) %>%
  summarise(mean_num_char = mean(num_char))
  mutate(group = 1)

g <- ggplot(email, aes(x = num_char,
     y = spam, color = spam)) +
  geom_jitter(alpha = 0.2) +
  geom_line(data = means,
  aes(x = mean_num_char, y = spam, group =
  color = "green", size = 1.5) +
  labs(x = "Number of characters (in thousa
  scale_color_manual(values = c("#0096FF",
  #guides(color = FALSE)+
  theme_bw()
```



This isn't something we can reasonably fit a linear model to, we need something different!

Framing the problem

- We can treat each outcome (spam and not) as successes and failures arising from separate *Bernoulli* trials
  - **Bernoulli trial**: a random experiment with exactly two possible outcomes, "success" and "failure", in which Pr(success) is the same every time the experiment is conducted
  - Each Bernoulli trial can have a separate probability of success

$$y_i \sim Bern(p_i)$$

- We can then use the predictor variables to model that probability of success, p_i
- We can't just use a linear model for p_i (since p_i must be between 0 and 1) but we can transform the linear model to have the appropriate range

Example:

How many heads from 20 flips of a fair coin **rbinom(20,1,.5)** for reproducibility need `set.seed(704)`

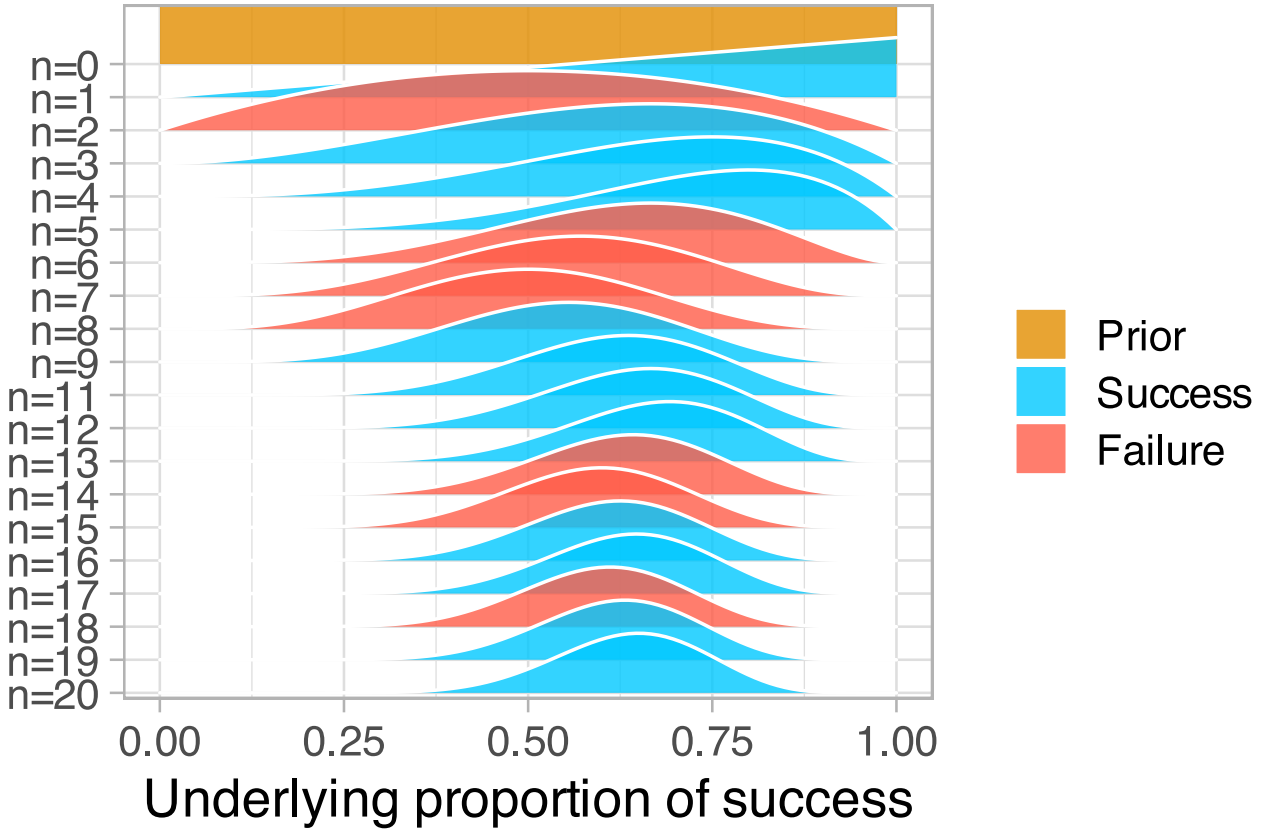
```
set.seed(704)
sample1 <- rbinom(20,1,.5) #<< 20 = n0 tries, 1 = sample size (Bernoulli), 0.5 probability e
sample1
```

```
##  [1] 1 0 1 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1
```

Sequential learning

Here is the sequence of data outcomes (0/1) of 20 trials with p = 0.5

Graphically our sequential learning process



Generalized linear models

- This is a very general way of addressing many problems in regression and the resulting models are called **generalized linear models (GLMs)**
- Logistic regression is just one example
- Other examples

Choice of Link g[ ]	Choice of Distribution	Expression	Resulting Form:
Identity function g(u)=u	Gaussian (normal)	$E(Y\|x) = \alpha+\beta_1 x$	Linear regression
Logit function	Bernoulli (binomial)	$\text{logit}[E(Y\|x)] = \alpha+\beta_1 x$	Logistic regression
Complementary log-log $g(u) = [\ln(-\ln(1-u))]$	Bernoulli (binomial)	$\text{cloglog}[E(Y\|x)] = \alpha+\beta_1 x$	Complementary log-log regression
Probit function	Bernoulli (binomial)	$\Phi^{-1}[E(Y\|x)] = \alpha+\beta_1 x$	Probit regression
Natural log	Poisson	$\ln[E(Y\|x)] = \alpha+\beta_1 x$	Poisson regression
Natural log	Bernoulli (binomial)	$\ln[E(Y\|x)] = \alpha+\beta_1 x$	Binomial regression

- What is the difference btw the last 2 equations?
- Last equation gives risk ratio for binary outcomes.
- Can also use identity function E(Y=1 | X) for risk differences

Three characteristics of GLMs

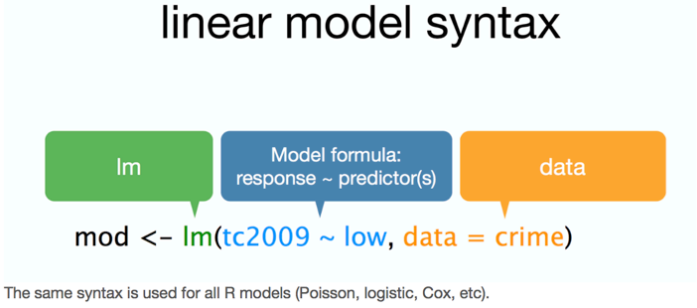
All GLMs have the following three characteristics:

1. A probability distribution describing a generative model for the outcome variable
2. A linear model:

$$\eta = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$$

3. A link function that relates the linear model to the parameter of the outcome distribution

R linear syntax



Default link for linear regression is the identity function (Gaussian distribution)
Logistic model link most often is logit function and binomial distribution

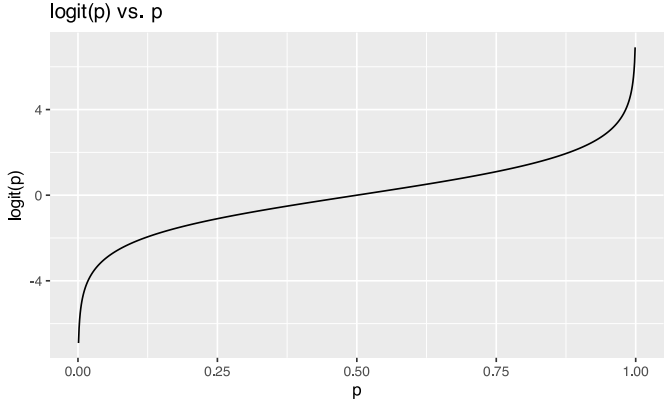
- Standard logistic model -> glm(model formula, data, family = "binomial")
- Bayesian logistic model -> rstanarm::stan_glm(model formula, data, family=binomial(link="logit"))
- Bayesian logistic model -> brms::brm(model formula, data, family=binomial(link="logit"))

Logistic regression

- Logistic regression is a GLM used to model a binary categorical outcome using numerical and categorical predictors
- To finish specifying the Logistic model we just need to define a reasonable link function that connects η_i (linear outcome variable) to p_i -> logit function
- **Logit function:** For $0 \leq p \leq 1$

$$logit(p) = \log(\frac{p}{1-p}) = log(Odds)$$

```
d <- tibble(p = seq(0.001, 0.999,
        length.out = 1000)) %>%
  mutate(logit_p = log(p/(1-p)))
g <- ggplot(d, aes(x = p, y = logit_p)) +
  geom_line() +  xlim(0,1) +
  ylab("logit(p)") +
  labs(title = "logit(p) vs. p")
```



Properties of the logit and logistic (inverse logit) functions

Logit function takes a value between 0 and 1 and maps it to a value between $-\infty$ and ∞

$$logit(p) = \log\left(\frac{p}{1-p}\right) = \alpha + \beta_1 x$$

- Take the inverse of the above equation, applies (*exp*) to both sides to get the **inverse logit** function, AKA **logistic** or **expit** functions:

$$\frac{p}{1-p} = exp^{\alpha+\beta_1 x}$$

- Rearranging leads to

$$p = \frac{exp^{\alpha+\beta_1 x}}{1+exp^{\alpha+\beta_1 x}} \in (0,1) \, or \, equivalently \, p = \frac{1}{1+exp^{-(\alpha+\beta_1 x)}} \in (0,1)$$

- **The above is the logistic function** & takes a value between $-\infty$ & ∞ and maps it to a value between 0 & 1, a probability.

Properties of the inverse logit or logistic function:

Logit is interpreted as the log odds of a success & inverse logit gives the probability of success.

$$p = \frac{exp^{\alpha+\beta_1 x}}{1 + exp^{\alpha+\beta_1 x}} \in (0, 1)$$

If p = P(D = 1 | X = x), what is P(D = 0 | X = x)?

$$P(D = 0|X = x) = \frac{1}{1 + exp^{\alpha+\beta_1 x}} \in (0, 1)$$

Facilitates the development of the logit model

$$Odds = \frac{P(D = 1|X = x)}{P(D = 0|X = x)} = \frac{\frac{exp^{\alpha+\beta_1 x}}{1+exp^{\alpha+\beta_1 x}}}{\frac{1}{1+exp^{\alpha+\beta_1 x}}} = exp^{\alpha+\beta_1 x}$$

Taking logs of both sides

$$log(\frac{P(D = 1|X = x)}{P(D = 0|X = x)}) = log(Odds) = logit = \alpha + \beta_1 x$$

The logistic regression model

Based on the three GLM criteria we have

- $y_i \sim \text{Bern}(p_i)$
- $\eta_i = \beta_0 + \beta_1 x_{1,i} + \cdots + \beta_n x_{n,i}$
- $\text{logit}(p_i) = \eta_i$
- fit is done with maximum likelihood and not minimizing the sum of squared residuals, actually there are no residuals!

$$L(success) = \hat{\pi}$$

$$L(failure) = 1 - \hat{\pi}$$

$$L(model) = \prod_{i=1}^{n} L(y_i)$$

- Likelihood values are often very small, one reason to use log-likelihood.
- Log likelihood is always negative as the likelihood is always between 0-1.
- Best fit is smallest value (i.e closest to 0)

Likelihood Ratio Test (LRT)

Here, our goal is to compare the log-likelihoods of two models: the one we build vs. the constant model.

Similar to comparing the sum of the squares explained by a LR model to the model that consists solely of the grand mean.

The null hypothesis in the LRT is that $\beta_1 = \beta_2 = \cdots = \beta_k = 0$.

The alternative hypothesis is that at least one of these coefficients is non-zero. The test statistic is:

$$G = -2\log(\text{constant model}) - (-2\log(\text{model}))$$

These two quantities are known as deviances. It can be shown that G follows a χ^2 distribution with k degrees of freedom (k=no. of parameters in the model).

Likelihood Ratio Test (LRT)

Spam example - model GLM (same format as LR except)

- use `"glm"` instead of `"lm"`
- define `family = "binomial"` for the link function

```
mod <- glm(spam ~ num_char, data = email, family = "binomial")
jtools::summ(mod, confint=T, exp=T, digits = 3, model.info = F, model.fit = F)
```

	exp(Est.)	2.5%	97.5%	z val.	p
(Intercept)	0.166	0.144	0.190	-25.135	0.000
num_char	0.940	0.925	0.955	-7.746	0.000
Standard errors: MLE					

```
logLik(mod)
```

```
## 'log Lik.' -1173.201 (df=2)
```

Likelihood Ratio Test (LRT)

```
library(lmtest)
lrtest(mod)
```

```
## Likelihood ratio test
##
## Model 1: spam ~ num_char
## Model 2: spam ~ 1
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -1173.2
## 2   1 -1218.6 -1 90.779  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model is improved with the addition of the `num_char` variable

The logistic regression model

3 spaces to consider

- logistic space - for 1 unit change in x -> linear change in β_1 (change in log odds)
- odds space - for 1 unit change in x -> $\frac{e^{\beta_1(x+1)}}{e^{\beta_1 x}} = e^{\beta_1}$ **multiplier** -> (change in odds ratio)
- probability space - for 1 unit change in x -> no nice sentence to describe the change since non-linear function but intuitively easier to understand

$$p_i = \frac{\exp(\beta_0 + \beta_1 x_{1,i} + \cdots + \beta_k x_{k,i})}{1 + \exp(\beta_0 + \beta_1 x_{1,i} + \cdots + \beta_k x_{k,i})} \in (0, 1)$$

Odds space

e^{β_1} is the **multiplier** for for 1 unit change in x (odds ratio - constant change)
Consider predicting spam based on inclusion of the term `winner`

```
two.way <- table(email$winner, email$spam); two.way <- two.way[c(2,1),] #changing row order
kable(two.way); mosaic::oddsRatio(two.way)
```

	0	1
yes	44	20
no	3510	347

```
## [1] 4.597852
```

```
mod <- glm(spam ~ winner, data = email, family = "binomial"); exp(coef(mod))
```

```
## (Intercept)   winneryes
##   0.0988604   4.5978517
```

This shows that OR calculated from 2X2 table = $exp(\beta)$ from logistic model

Spam model

```
#Usual model approach
spam_fit <- glm(spam ~ num_char, data = email, family = "binomial")
jtools::summ(spam_fit, digits=3, confint=T, model.info = F, model.fit = F) #use summary(spam
```

	Est.	2.5%	97.5%	z val.	p
(Intercept)	-1.799	-1.939	-1.658	-25.135	0.000
num_char	-0.062	-0.078	-0.046	-7.746	0.000
Standard errors: MLE					

Prediction

P(spam) for an email with 2000 characters
Model:

$$\log\left(\frac{p}{1-p}\right) = -1.80 - 0.0621 \times \text{num\_char}$$

$\log\left(\frac{p}{1-p}\right)$ = -1.80-0.0621* 2 = -1.9242

$$\frac{p}{1-p} = \exp(-1.9242) = 0.15 \rightarrow p = 0.15 \times (1-p)$$

$$p = 0.15 - 0.15p \rightarrow 1.15p = 0.15$$

$$p = 0.15/1.15 = 0.13$$

Somewhat easier with less calculations

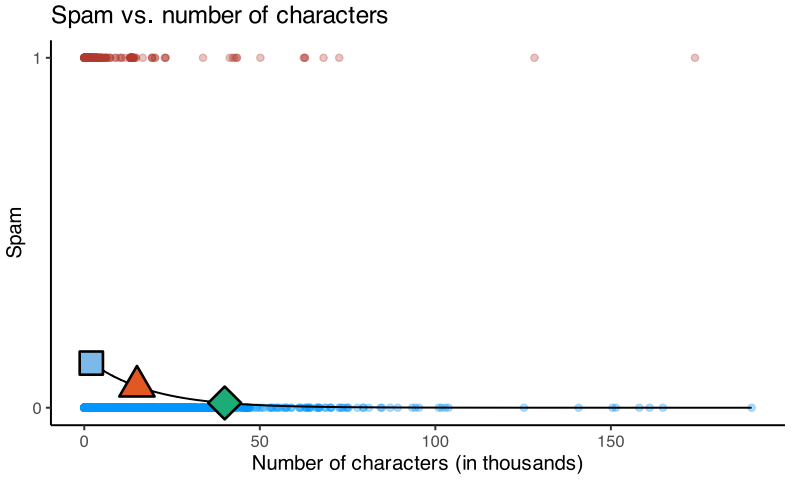
```
rstanarm::invlogit(-1.924)
```

```
## [1] 0.1274162
```

Prediction

What is the probability that an email with 15000 or 40000 characters is spam?

```
# General formula from tidymodels

spam_fit <- logistic_reg() %>% set_engine("
  fit(spam ~ num_char,
      family="binomial", data = email)
newdata <- tibble(num_char = c(2, 15, 40),
            color= c("#7CB9E8", "#E25822",
            shape  = c(22, 24, 23))

y_hat <- predict(spam_fit, newdata, type =

p_hat <- exp(y_hat) / (1 + exp(y_hat))

newdata <- newdata %>%
  bind_cols(y_hat = y_hat,p_hat = p_hat)

spam_aug <- augment(spam_fit$fit) %>%
  mutate(prob =
           exp(.fitted) / (1 + exp(.fitted))
```

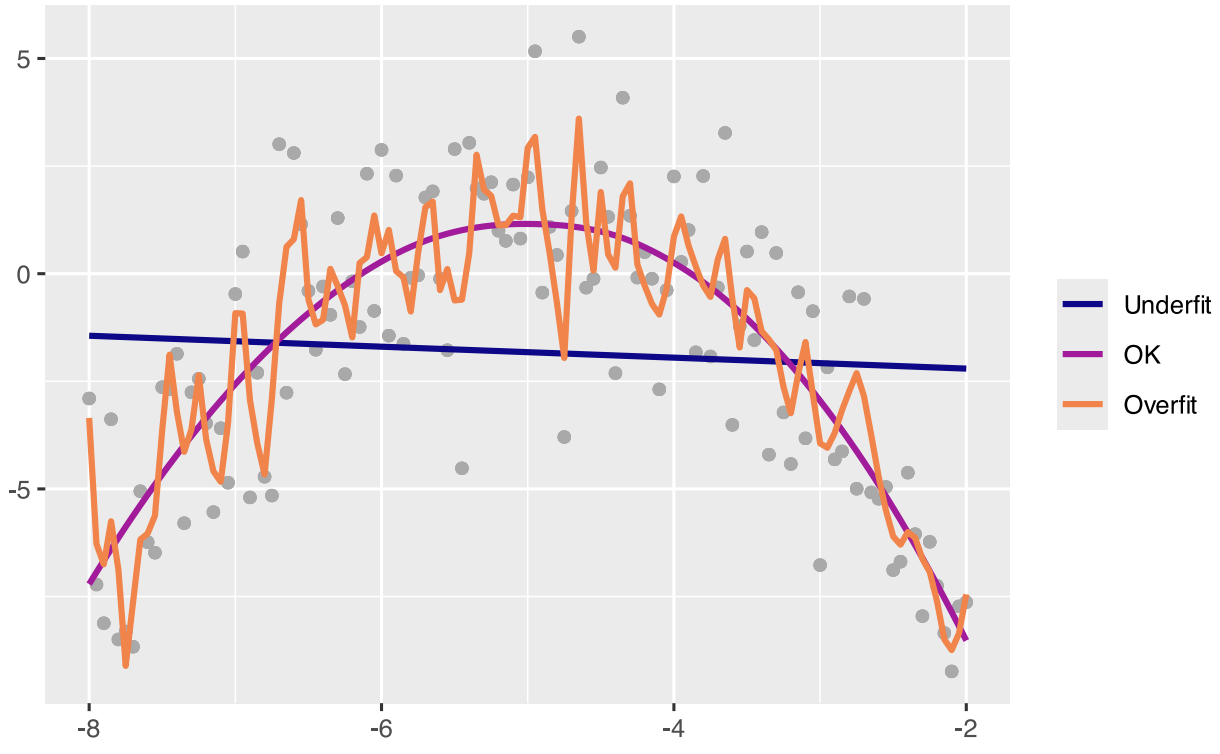


- 2K chars: P(spam) = 0.13
- 15K chars, P(spam) = 0.06
- 40K chars, P(spam) = 0.01

Prediction

- The mechanics of prediction is **easy**:
  - Plug in values of predictors to the model equation
  - Calculate the predicted value of the response variable, $\hat{y}$
- **Getting it right is hard!**
  - There is no guarantee the model estimates you have are correct
  - Or that your model will perform as well with new data as it did with your sample data

Underfitting and overfitting



Sensitivity and specificity

We already learned how to compare models using the deviance, but how do we know how well our model works?

One techinque for assessing the goodness-of-fit in a logistic regression model is to examine the percentage of the time that our model was *right*."

	Email is spam	**Email is not spam**
Email labelled spam	True positive	False positive (Type 1 error)
Email labelled not spam	False negative (Type 2 error)	True negative

- Sensitivity = P(Labelled spam | Email spam) = TP / (TP + FN)
  - Sensitivity = 1 - False negative rate = 1 - (FN / (TP + FN))
- Specificity = P(Labelled not spam | Email not spam) = TN / (FP + TN)
  - Specificity = 1 - False positive rate = 1 - (FP / (FP + TN))

Classification

```
mod1 <- glm(spam ~ ., data = email, family = "binomial") #full model
email <- email %>%
  mutate(fitted = fitted.values(mod1)) %>%
  mutate(fitspam = ifelse(fitted >= 0.5, 1, 0))
tbl <- table(email$spam, email$fitspam)
tbl
```

```
##
##         0    1
##   0 3521   33
##   1  299   68
```

```
sum(diag(tbl)) / nrow(email)
```

```
## [1] 0.9153277
```

Model is correct 91.5% of the time

Accuracy of other models

Average spam risk of the null model

```
av_risk <- sum(as.numeric(email$spam==1)) /
email <- email %>%
  mutate(fitspam= sample(c(0,1), size=3921,
         replace=TRUE, c(1-av_risk, av_risk)
table(email$spam, email$fitspam)
```

```
##
##         0    1
##    0 3267  287
##    1  329   38
```

```
(table(email$spam, email$fitspam)[1,1] +
    table(email$spam,
    email$fitspam)[2,2]) / nrow(email)
```

```
## [1] 0.8428972
```

model accuracy = 82%

Model probability as coin toss

```
email <- email %>% mutate(fitspam= sample(c
          size=3921, replace=TRUE))
table(email$spam, email$fitspam)
```

```
##
##         0    1
##    0 1809 1745
##    1  188  179
```

```
(table(email$spam, email$fitspam)[1,1] +
    table(email$spam,
    email$fitspam)[2,2]) / nrow(email)
```

```
## [1] 0.5070135
```

model accuracy = 50%

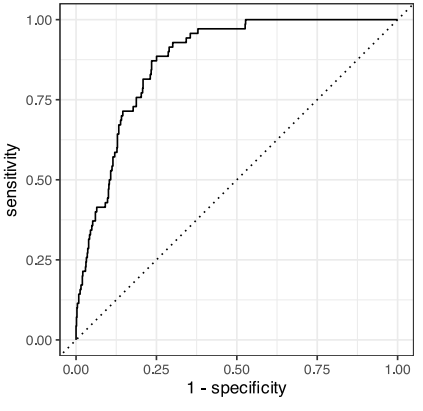
Trade-offs

If you were designing a spam filter, would you want sensitivity and specificity to be high or low?

```
library (rsample) # get initial_split, trai
email_split <- initial_split(email, prop =
# Create data frames for the two sets:
train_data <- training(email_split)
test_data  <- testing(email_split)
email_fit <- logistic_reg() %>%
  set_engine("glm") %>%
  fit(spam ~ ., data = train_data,
      family = "binomial")
email_pred <- predict(email_fit, test_data,
                      type = "prob") %>%
  bind_cols(test_data %>%
              dplyr::select(spam, time))
```

What are the trade-offs associated with each decision?

```
email_pred %>%
  roc_curve(truth = spam, .pred_1, event_le
```




```
email_pred %>% roc_auc(truth = spam,.pred_1
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.875
```

Types of logistic regression

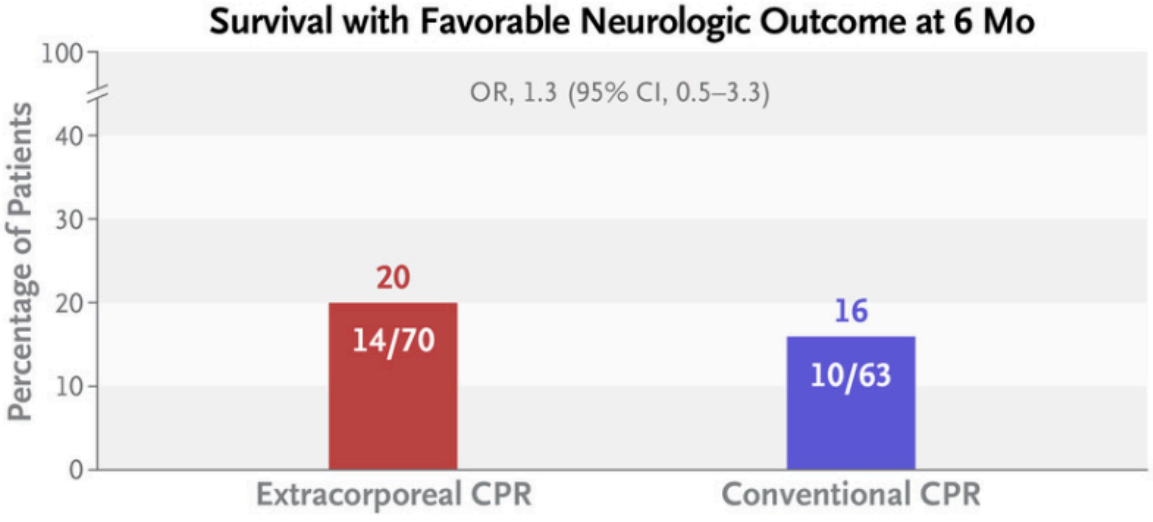
The three types of logistic regression

1. Binary logistic regression - What we have been doing
2. Multinomial logistic regression - When we have multiple outcomes, e.g. predict whether someone may have the flu, an allergy, a cold, or COVID-19
3. Ordinal logistic regression - When the outcome is ordered, e.g. determine the severity of a COVID-19 infection, sorting it into mild, moderate, and severe cases

   For what it's worth, in machine learning (ML) supervised learning is used to classify something or predict a value, and logistic regression is a common classifier used ML

Illustration of reproducing results and more...

A recent paper in the NEJM concluded that *"In patients with refractory out-of-hospital cardiac arrest, extracorporeal CPR and conventional CPR had similar effects on survival with a favorable neurologic outcome"*.



Two questions

1. Can we reproduce the analysis?
2. Accepting the authors' analysis, do we accept their conclusion of similar effects?

Illustration of reproduction of results

Given the use of OR, the analysis was most likely been a logistic regression with the binary outcome of success / failure at 180 days.

```
# simulate dataset - one line per individual
dat <- data.frame(trt = 1:133 > 63,  #0 = control, 1 = extracorporeal
                #   time = sample(1:180, size = 133, replace = TRUE),  if wanted to simulate a
                   event = c(1:63 < 11, 1:70 < 15)) %>%
  mutate(trt = ifelse (trt=="FALSE", 0, 1),event = ifelse (event=="FALSE", 0, 1))
fit <- glm(event ~ trt, data = dat, family = binomial(link="logit"))
```

Log-Odds Coefficients

	Est.	S.E.	z val.	p
(Intercept)	-1.7	0.3	-4.8	0.0
trt	0.3	0.5	0.6	0.5
Standard errors: MLE				

Exponentiated coefficients which reproduces the NEJM results.

	exp(Est.)	2.5%	97.5%	z val.	p
(Intercept)	0.19	0.10	0.37	-4.84	0.00
trt	1.33	0.54	3.24	0.62	0.54
Standard errors: MLE					

Illustration of reproduction of results

A simpler approach is to use the aggregated data and apply a `weight` argument within the `glm` function which gives the exact same answer as expected.

```
# create data frame
dat1 <- tibble(Trial = c("INCEPTION", "INCEPTION"), Tx = c("cCPR", "eCPR"),
               fail = c(53, 56), success = c(10,14)) %>%
  mutate(total = fail + success, prop_success = success / total)
fit1 <- glm(prop_success ~ Tx, data = dat1, family = binomial(link="logit"), weights = total
```

Log-Odds Coefficients

	Est.	S.E.	z val.	p
(Intercept)	-1.7	0.3	-4.8	0.0
TxeCPR	0.3	0.5	0.6	0.5
Standard errors: MLE				

Exponentiated coefficients

```
jtools::summ(fit1, confint=T, exp=T, digits
```

	exp(Est.)	2.5%	97.5%	z val.	p
(Intercept)	0.2	0.1	0.4	-4.8	0.0
TxeCPR	1.3	0.5	3.2	0.6	0.5
Standard errors: MLE					

Illustration: Reproduction of Results, using the Bayesian approach

Here are **4 methods** which all give the same answer as the frequentist method using the default vague priors (to see defaults use `prior_summary`)

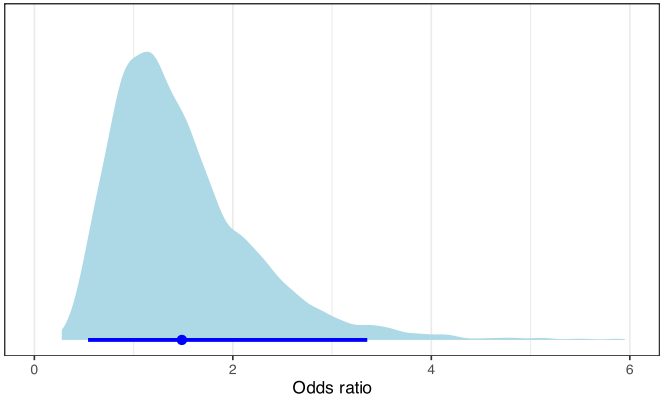
- 2 approaches with `rstanarm` - one with aggregated data, one with individual data
- 2 approaches with `brms` - one with aggregated data, one with individual data

```
#using stan_glm
fit2 <- stan_glm(prop_success ~ Tx, data = dat1, family = binomial(link="logit"),
                 weights = total, refresh=0) #aggregate data
fit2a <- stan_glm(event ~ trt, data = dat, family = binomial(link="logit"), refresh=0)  #ind

#Using brms
fit3 <- brm(success | trials(total) ~ Tx, data = dat1, family = binomial(link="logit"), refr
fit3a <- brm(event ~ trt, data = dat, family = binomial(link="logit"), refresh=0) #individua
```

Reproduction of Results, using the Bayesian approach

How to assess the **Odds Ratio** from logistic regression model

```
library(knitr)

fit3 <- brm(success | trials(total) ~ Tx,
            data = dat1,
            family = binomial(link="logit")
draws3 <- as_draws_df(fit3)  %>% # rename a
  mutate(or = exp(b_TxeCPR)) # compute the

gg <- draws3 %>%
  ggplot(aes(x = or)) +
  stat_halfeye(point_interval = mean_qi,
               .width = .95, color="blue",
  scale_y_continuous(NULL, breaks = NULL) +
  xlim(0, 6) +
  xlab("Odds ratio")+ theme_bw()
```



Do we really want the OR?

OR is a measure of association that is difficult to interpret

- Popular because outcome from logistic regression models

But other options exist for binomial data

- The log-binomial and binomial regression models estimate the risk ratio and the risk difference, respectively
- These models are sometimes referred to collectively as binomial regression models with, respectively, a log link and an identity link

Reproduction of Results, using the Bayesian approach

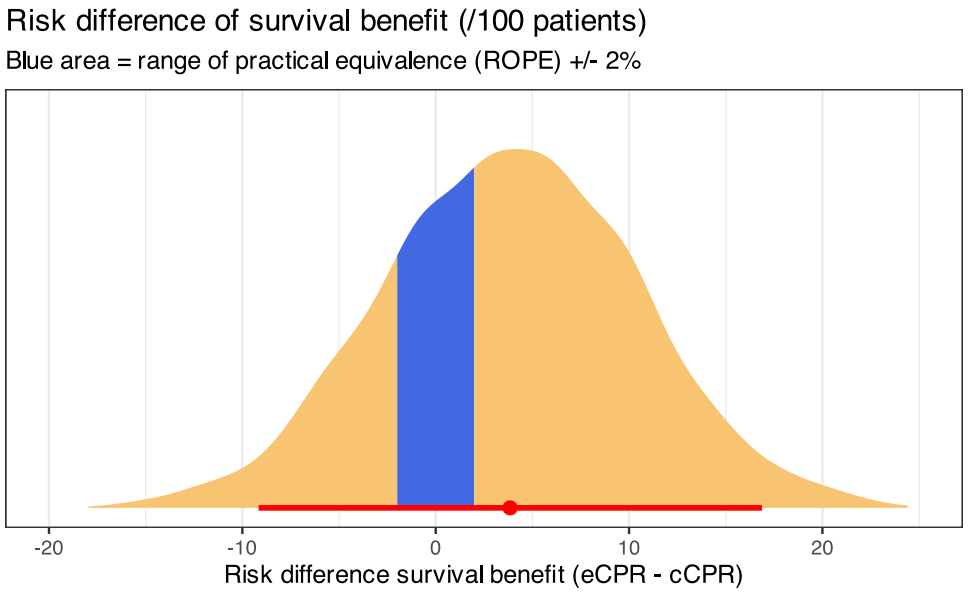
Risk difference using binomial regression

```
fit4 <- brm(success | trials(total) ~ Tx, data = dat1,
            family = binomial(link="identity"), refresh=0)
draws4 <- as_draws_df(fit4)  %>% #draws from posterior
   transmute(diff= 100*b_TxeCPR) # rename and drop the unneeded columns

print(fit4, digits=2)
```

```
##  Family: binomial
##   Links: mu = identity
## Formula: success | trials(total) ~ Tx
##    Data: dat1 (Number of observations: 2)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Regression Coefficients:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.17      0.05     0.09     0.26 1.00     2501     2461
## TxeCPR        0.04      0.07    -0.09     0.17 1.00     2655     2058
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

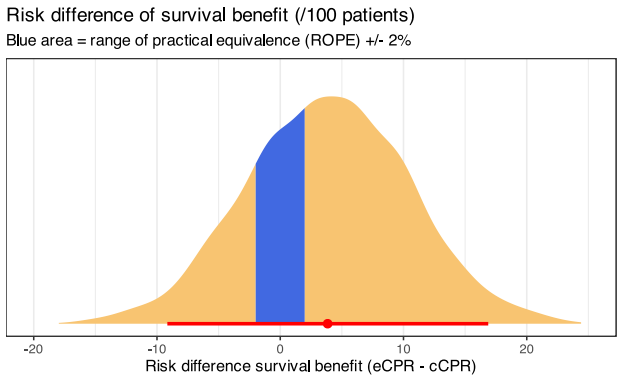
Reproduction of Results, using the Bayesian approach

Risk difference using binomial regression



Remember the authors' conclusions **"eCPR and cCPR had similar effects on survival"**
What is the probability this is true? Need to define *similar*?

Region/Range of Practical Equivalence (ROPE)



Assuming +/- 2 lives / 100 is similar,

- Blue area represents this equivalence probability, 19.775%.
- There remains a 61.475% that eCPR offers a clinically meaningful survival benefit (orange area to the right of blue area).

A good resource about the Range of Practical Equivalence -ROPE Here

Bayes has certainly deepened our appreciation of this data

Another advantage of using the Bayesian approach

Some prior information exists from 2 previous RCTs and the Bayesian analysis can take this information into account.

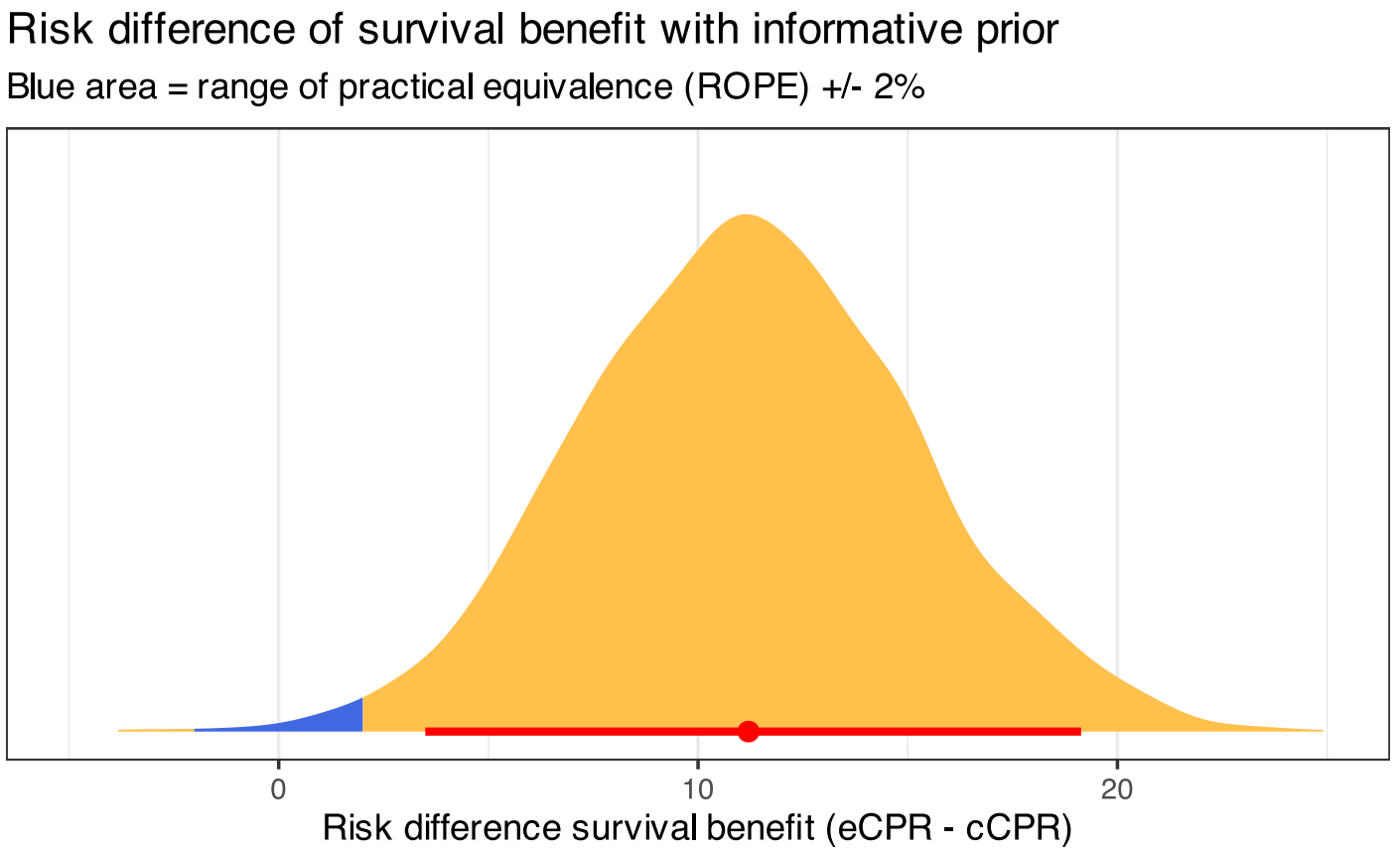
- PRAGUE & ARREST trials (combined) 25 successes and 122 failures in cCRP (beta(25, 122)).
- PRAGUE & ARREST trials (combined) 44 successes and 94 failures in eCRP (beta(44, 94)).

```
fit5 <- brm(success | trials(total) ~ 0 + Tx, data = dat1,
            family = binomial(link="identity"), refresh = 0)

fit5 <- brm(success | trials(total) ~ 0 + Tx, data = dat1,
            family = binomial(link="identity"),
            prior = c(prior(beta(1, 1), class = b, lb = 0, ub = 1),
            prior(beta(25,122), class = b, coef = "TxcCPR"),
            prior(beta(44,94),class = b, coef = "TxeCPR")),
  chains = 4, warmup = 1000, iter = 2000, seed = 123, refresh = 0)
```

Another advantage of using the Bayesian approach

```
##  Family: binomial
##   Links: mu = identity
## Formula: success | trials(total) ~ 0 + Tx
##    Data: dat1 (Number of observations: 2)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Regression Coefficients:
##        Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## TxcCPR     0.17      0.02     0.12     0.22 1.00     3457     2813
## TxeCPR     0.28      0.03     0.22     0.34 1.00     3925     2855
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Another advantage of using the Bayesian approach



Despite inconclusive result from INCEPTION trial, the totality of the evidence suggest a meaningful benefit from eCRP.

QUESTIONS?

COMMENTS?

RECOMMENDATIONS?

Code for the prediction plot

```
g1 <- ggplot(spam_aug, aes(x = num_char)) +
  geom_point(aes(y = as.numeric(spam)-1, color = spam), alpha = 0.3) +
  scale_color_manual(values = c("#0096FF", "#b03a2e")) +
  scale_y_continuous(breaks = c(0, 1)) +
  guides(color = FALSE) +
  geom_line(aes(y = prob)) +
  geom_point(data = newdata, aes(x = num_char, y = p_hat),
             fill = newdata$color, shape = newdata$shape,
             stroke = 1, size = 6) +
  labs(x = "Number of characters (in thousands)", y = "Spam",
       title = "Spam vs. number of characters"
  ) + theme_classic()
#g1
```

Code for the Odds Ratio's Posterior Distributions

```
fit3 <- brm(success | trials(total) ~ Tx,
      data = dat1,
      family = binomial(link="logit"), refresh=0) #aggregate data
draws3 <- as_draws_df(fit3)  %>% # rename and drop the unneeded columns
  mutate(or = exp(b_TxeCPR)) # compute the OR

gg <- draws3 %>%
  ggplot(aes(x = or, color ="lightblue")) +
  stat_halfeye(point_interval = mean_qi,
               .width = .95, color="blue") +# slab_colour
  scale_y_continuous(NULL, breaks = NULL) +
  xlim(0, 9) +
  xlab("Odds ratio")+ theme_minimal()
```

Code for the ROPE

```
gg <- ggplot(draws4, aes(x = diff)) +
  stat_halfeye(aes(fill = after_stat(abs(x) < 2)), point_interval = mean_qi, .width = .95) +
  scale_y_continuous(NULL, breaks = NULL) +
  xlim(-20,25) +
  xlab("Risk difference survival benefit (eCPR - cCPR)") +
  scale_fill_manual(values = c("orange", "skyblue")) +
  ggtitle("Risk difference of survival benefit (/100 patients)",
          subtitle = "Blue area = range of practical equivalence (ROPE) +/- 2%") +
  theme_bw() +
  theme(legend.position="none")
```

Also check here:
https://easystats.github.io/bayestestR/articles/region_of_practical_equivalence.html

Makowski, D., Ben-Shachar, M. S., & Lüdecke, D. (2019). bayestestR: Describing Effects and their Uncertainty, Existence and Significance within the Bayesian Framework. Journal of Open Source Software, 4(40), 1541. https://doi.org/10.21105/joss.01541