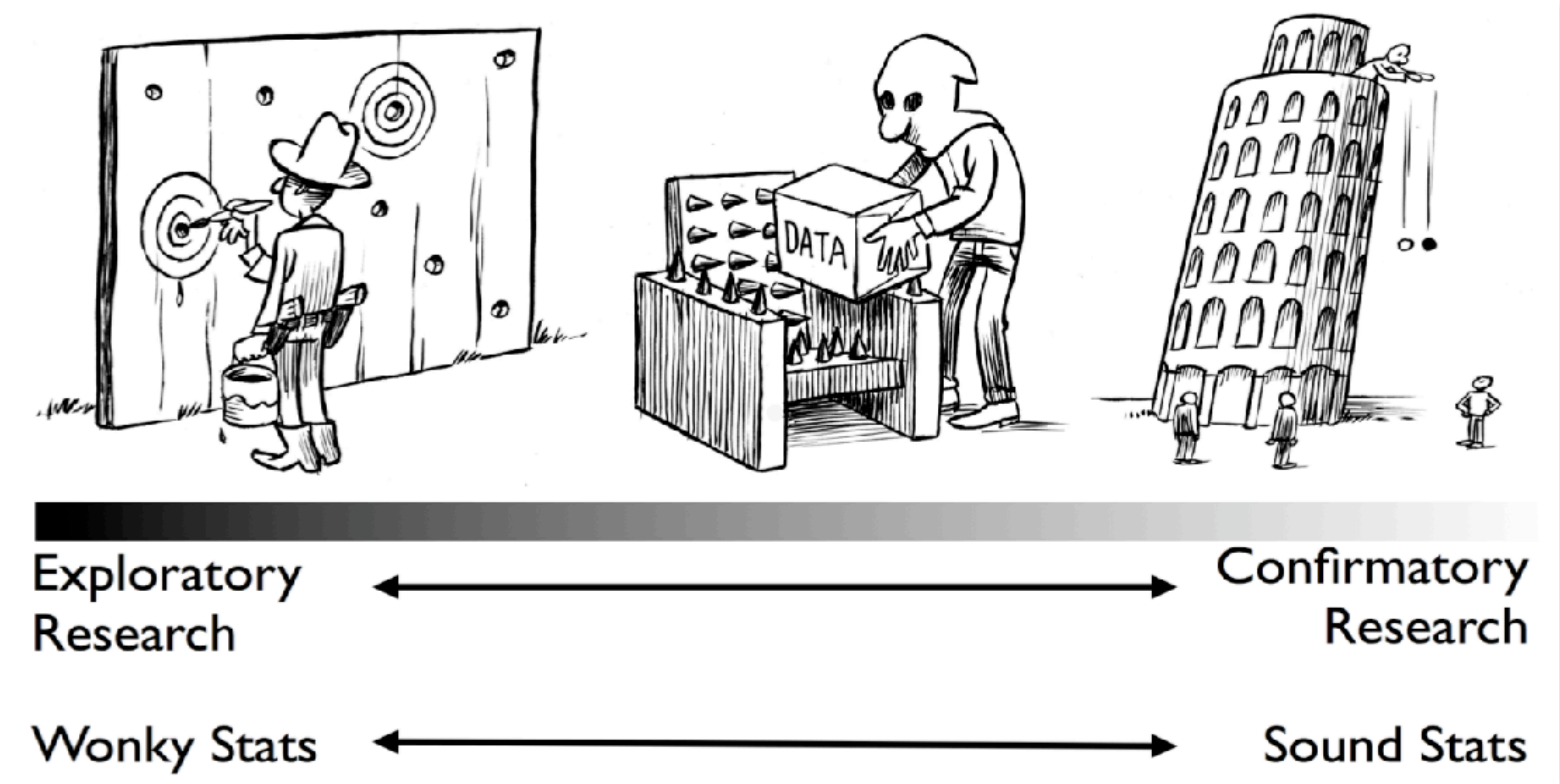
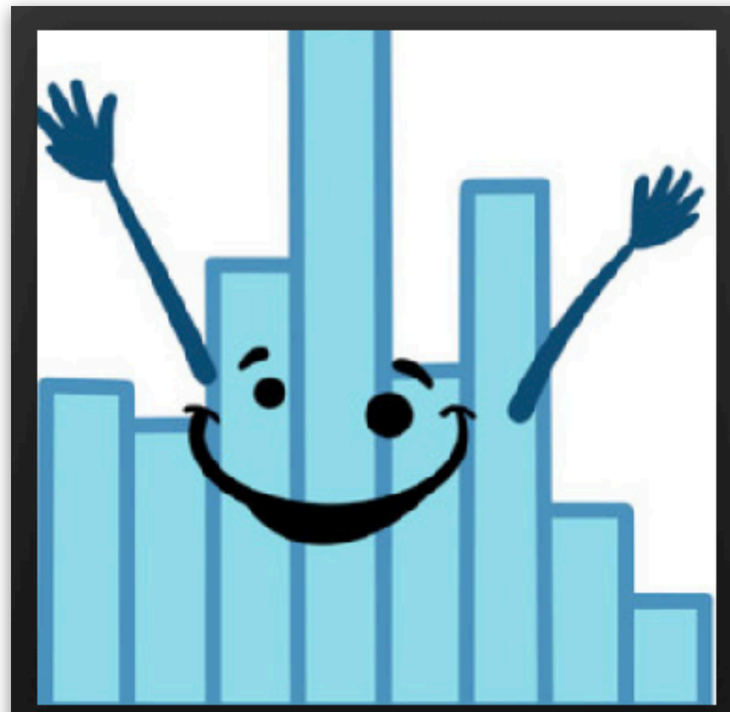
PSYCH 252: Statistical Methods for Behavioral and Social Sciences

Power analysis





COLLABORATIVE PLAYLIST

psych252

https://tinyurl.com/psych252spotify25

PLAY

02/10/2024

Logistics

This week

no class on Wednesday (more time to work on midterm)

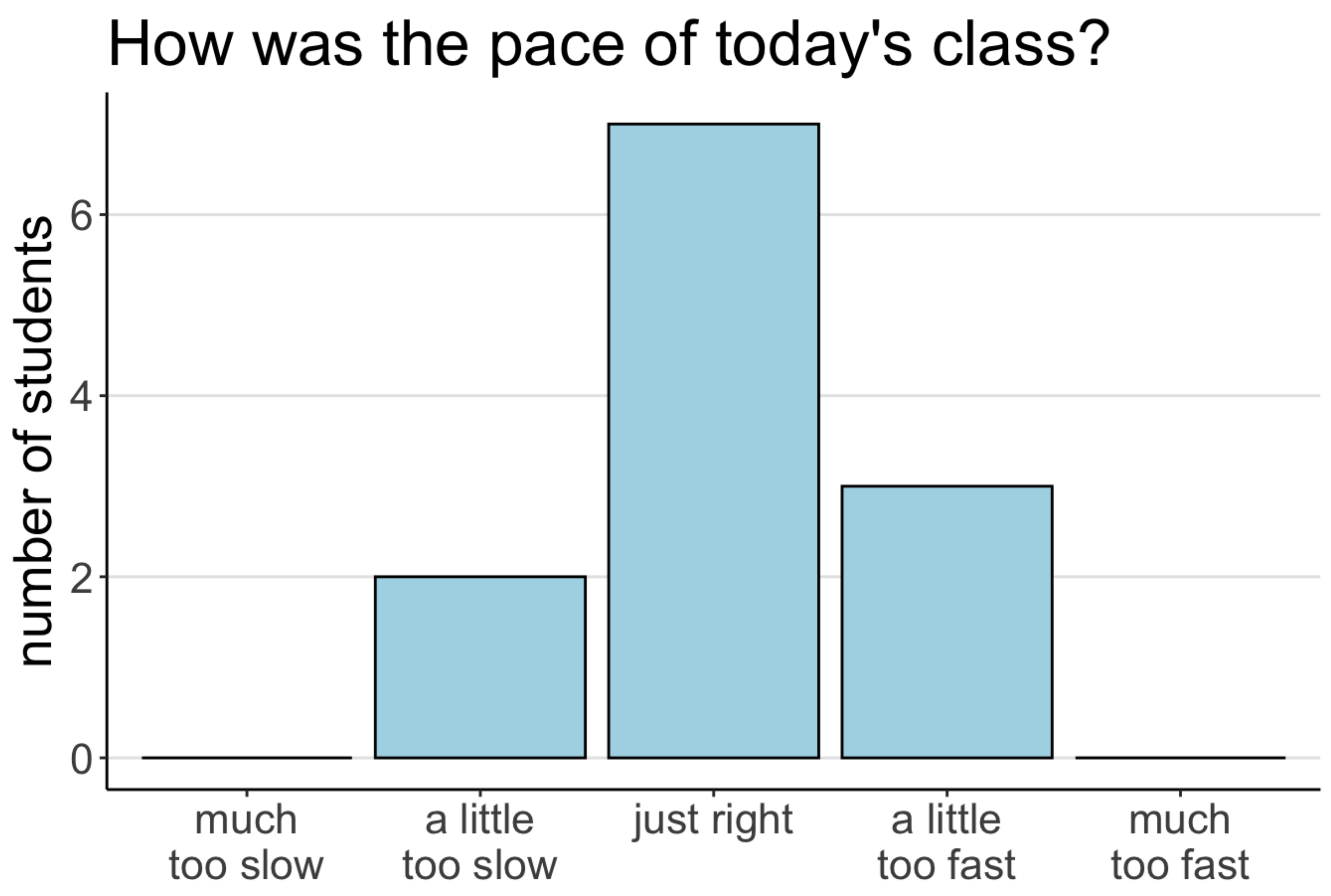
no sections

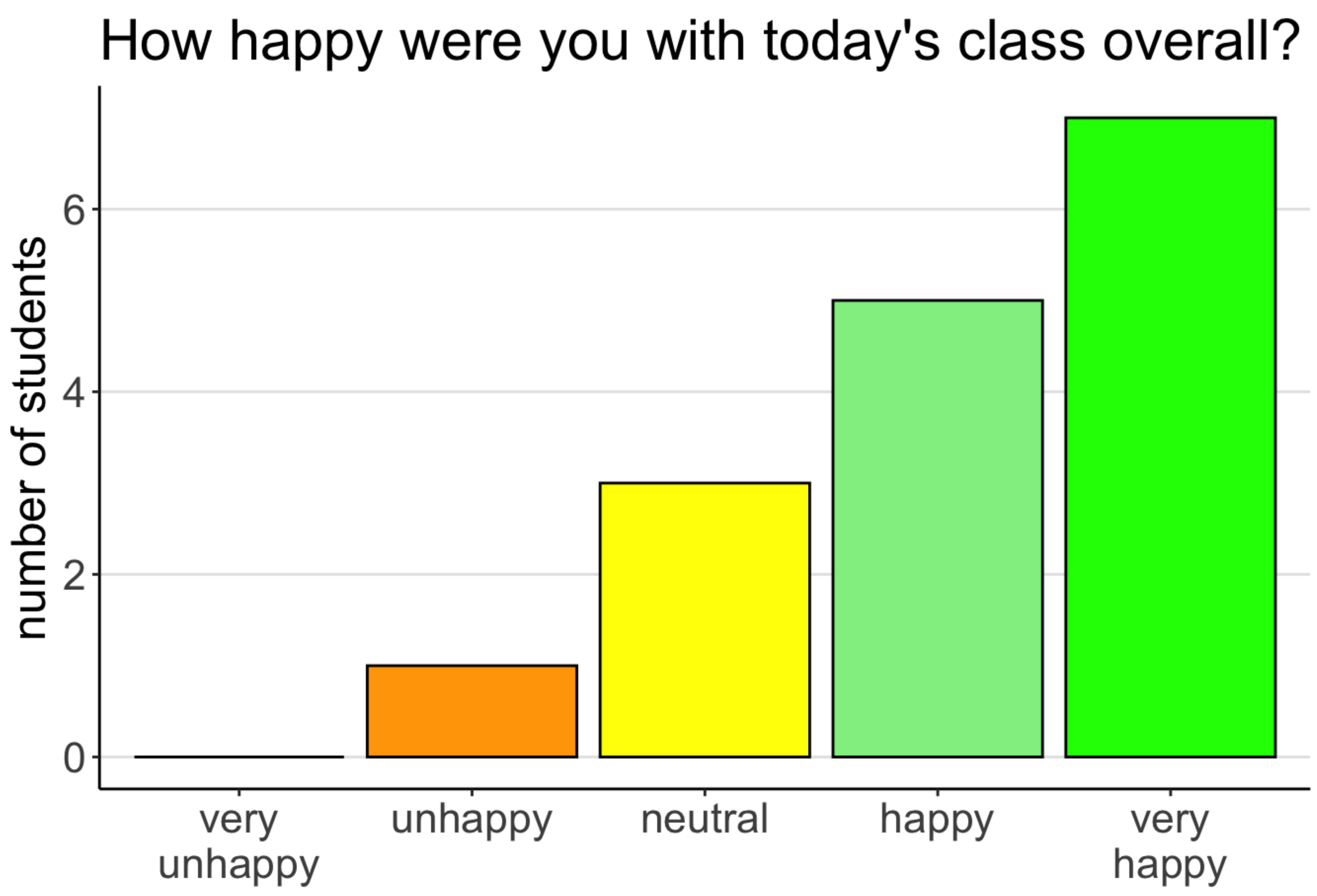
no office hours (at least not for questions about the midterm)

midterm is due **Friday 14th at 8pm**

Feedback

Feedback





thanks for responding questions so clearly :)

Today was the first day I felt **lost in the details** and wasn't sure **what exactly to take away** from the lecture.

I was a little distracted by how long the question and answers were, but understand thats a difficult balance to strike.

It would be great if at the end of each statistical test, or at least some, you could **interpret as you would when writing it for a publication**.

Is there any way we could also **put maths equations** in just to explain some of the things we are going through in class? Sometimes I get what it means theoretically but still wanna get a glimpse of how it really works underneath the code.

Things that came up

Correcting for multiple comparisons

First, he wrote, she should break up the diners into all kinds of groups: "males, females, lunch goers, dinner goers, people sitting alone, people eating with groups of 2, people eating in groups of 2+, people who order alcohol, people who order soft drinks, people who sit close to buffet, people who sit far away, and so on..."

He concluded on an encouraging note: "Work hard, squeeze some blood out of this rock, and we'll see you soon."

https://www.buzzfeednews.com/article/stephaniemlee/brian-wansink-cornell-p-hacking

Correcting for multiple comparisons

For each statistical test, we have a 5% chance that we're incorrectly rejecting the null hypothesis.

If we run many such tests, the overall likelihood of incorrectly rejecting a null hypothesis increases.

Two solutions:
1) pre-registering our hypotheses
2) correcting for multiple comparisons

Bonferroni correction: $\frac{\alpha}{m}$ m = number of tests

R often just multiplies the p-value by m

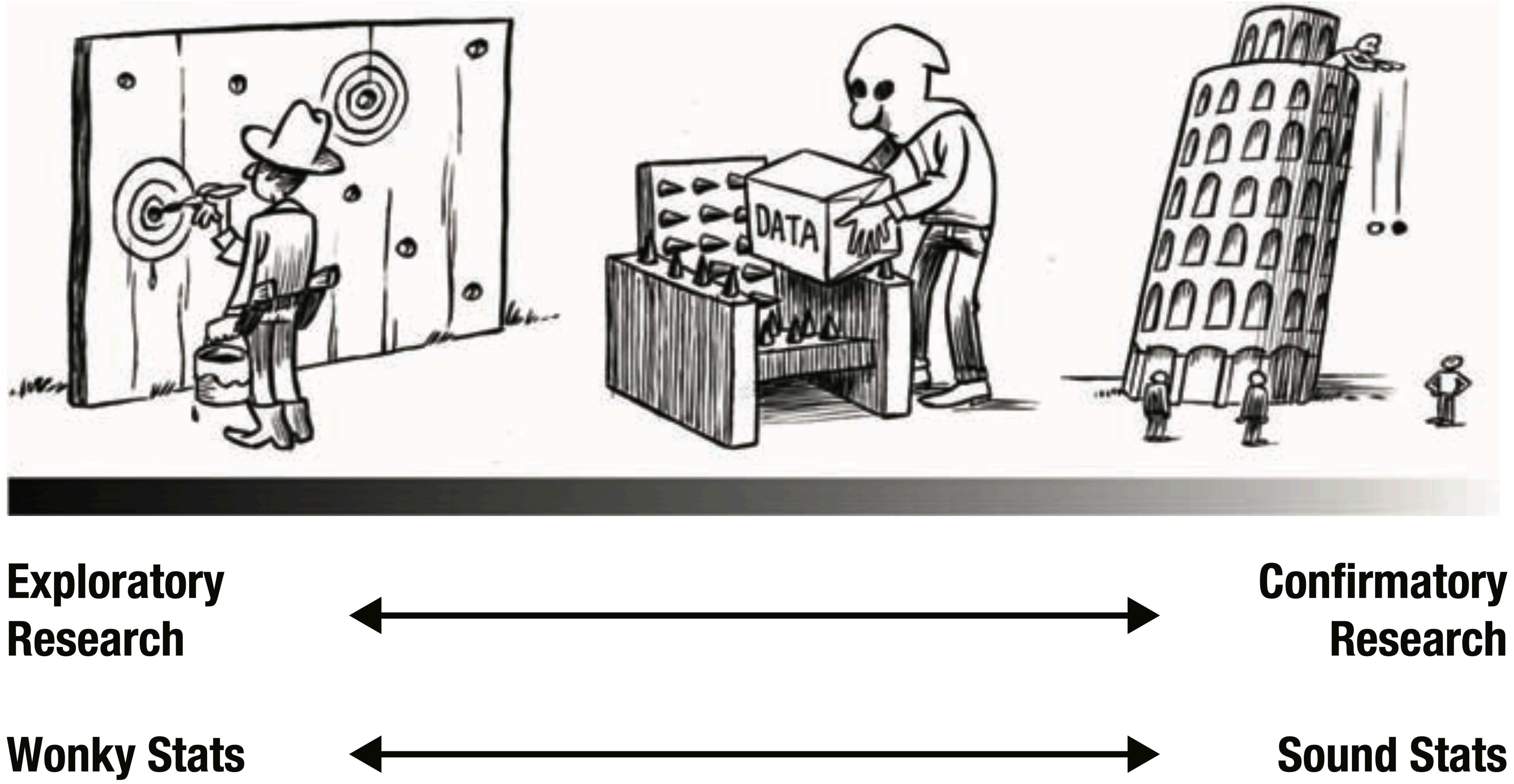
Correcting for multiple comparisons

exploratory analyses:

- estimate effect sizes (e.g. correlation coefficients, differences between groups, etc.)
- don't do inferential statistics (don't report p-values)
- reason: you could be running many many tests ($m = \infty$), so nothing would ever be significant once correcting for multiple comparisons $\frac{\alpha}{m} \approx 0$

confirmatory analyses:

- pre-register your analysis
- conduct frequentist tests, reporting p-values
- pre-register criteria for rejecting null hypotheses (one sided / two sided tests?, $p < .05$ or $p < .01$)



Wagenmakers, E.-J., Wetzels, R., Borsboom, D., van der Maas, H. L. J., & Kievit, R. A. (2012). An Agenda for Purely Confirmatory Research. *Perspectives on Psychological Science*, 7(6), 632–638.

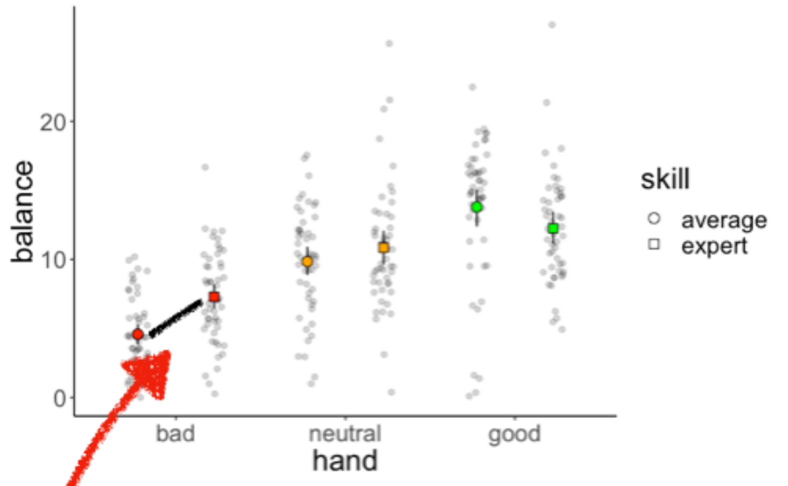
Plan for today

- Quick recap
- Generalized linear model
- Power analysis
  - Making decisions
  - Calculating power
  - Effect sizes
  - Determining sample size

Quick recap

Quick recap: Parameter interpretation

Parameter interpretation

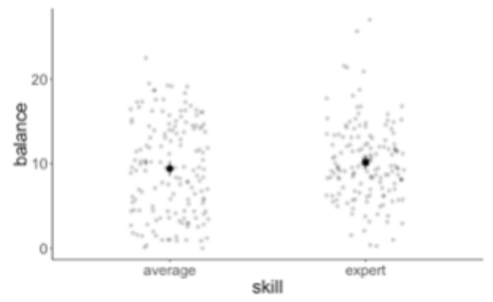



```
Call:
lm(formula = balance ~ hand * skill, data = df.poker)

Residuals:
     Min       1Q   Median       3Q      Max
-13.6976  -2.4740   0.0348   2.4644  14.7806

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)              4.5866     0.5686   8.067 1.85e-14 ***
handneutral              5.2572     0.8041   6.538 2.75e-10 ***
handgood                 9.2110     0.8041  11.455  < 2e-16 ***
skillexpert              2.7098     0.8041   3.370 0.000852 ***
handneutral:skillexpert -1.7042     1.1372  -1.499 0.135038
handgood:skillexpert    -4.2522     1.1372  -3.739 0.000222 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.02 on 294 degrees of freedom
Multiple R-squared:  0.3731,	Adjusted R-squared:  0.3624
F-statistic: 34.99 on 5 and 294 DF,  p-value: < 2.2e-16
```

```
lm(formula = balance ~ hand * skill,
   data = df.poker) %>%
  anova()
```

```
Analysis of Variance Table

Response: balance
           Df Sum Sq Mean Sq F value    Pr(>F)
hand        2 2559.4 1279.70 79.1692 < 2.2e-16 ***
skill       1   39.3   39.35  2.4344 0.1197776
hand:skill  2  229.0  114.49  7.0830 0.0009901 ***
Residuals 294 4752.3   16.16
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

there was no main effect of skill!



is this difference significantly different from 0?

hand	average	expert	difference
bad	4.59	7.3	2.71

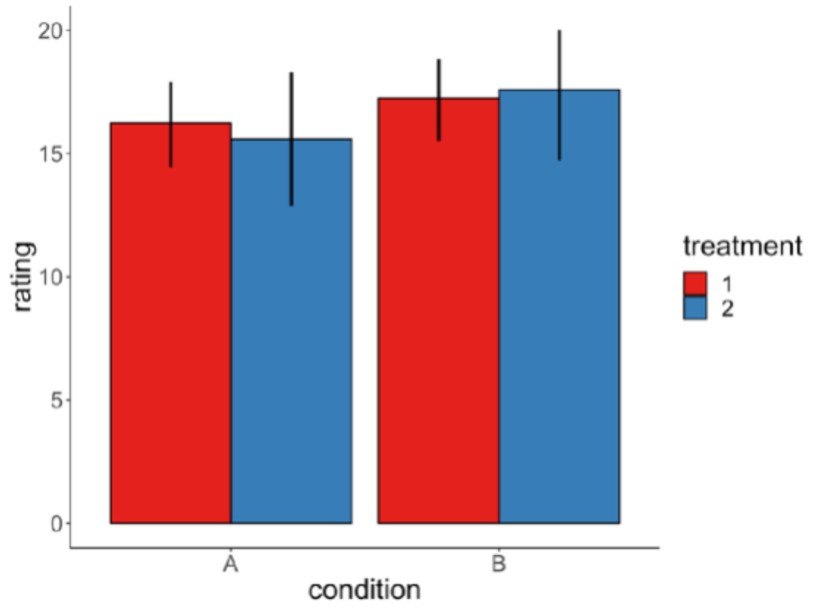
16

Different effect terms

- **main effect**: effect of one independent variable on the dependent variable
- **interaction effect**: when the effect of one independent variable depends on the level of another
- **simple effect**: comparison between two specific cell means

Who is the ANOVA champ?

Which effects are significant?



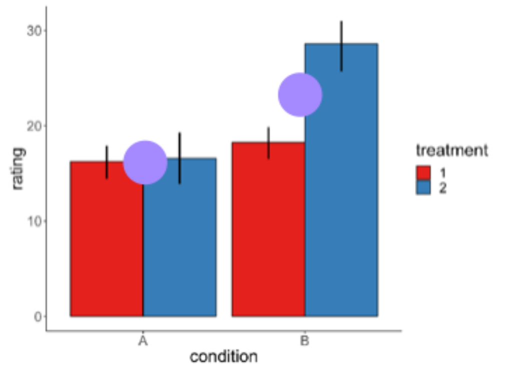
- Condition
- Treatment
- Condition x Treatment **interaction effect**
- Condition, Treatment **two main effects**
- Condition, Condition x Treatment
- Treatment, Condition x Treatment
- Condition, Treatment, Condition x Treatment

The winner gets chocolate!

21

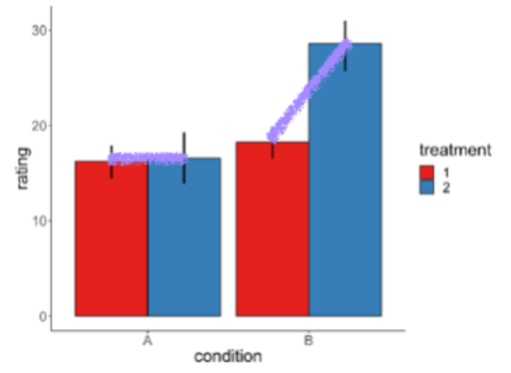
Solution

to detect main effects, try to visualize what the averaged group means would look like



main effect of condition

to detect interaction effects, try to visualize whether the slopes are different from each other



interaction effect

31

Quick recap: Unbalanced designs

Beware of unbalanced designs

```
lm(formula = balance ~ skill + hand, data = df.poker.unbalanced) %>%
  anova()
```

```
Analysis of Variance Table

Response: balance
           Df Sum Sq Mean Sq F value   Pr(>F)
skill       1   74.3   74.28  4.2904  0.03922 *
hand        2 2385.1 1192.57 68.8827  < 2e-16 ***
Residuals 286 4951.5   17.31
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

flipped the order

```
lm(formula = balance ~ hand + skill, data = df.poker.unbalanced) %>%
  anova()
```

```
Analysis of Variance Table

Response: balance
           Df Sum Sq Mean Sq F value Pr(>F)
hand        2 2419.8 1209.92 69.8845 <2e-16 ***
skill       1   39.6   39.59  2.2867 0.1316
Residuals 286 4951.5   17.31
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

34

Type I Sums of Squares

Type I Sums of Squares are Sequential, so the order of variables in the models makes a difference. This is rarely what we want in practice!

Sums of Squares are Mathematically defined as:

- SS(A) for independent variable A
- SS(B | A) for independent variable B
- SS(AB | B, A) for the interaction effect

A

caution: this is what `anova()` uses by default

Type III Sums of Squares

The Type III Sums of Squares are also called partial sums of squares again another way of computing Sums of Squares:

- Like Type II, the Type III Sums of Squares are not sequential, so the order of specification does not matter.
- Unlike Type II, the Type III Sums of Squares do specify an interaction effect.

Sums of Squares are Mathematically defined as:

- SS(A | B, AB) for independent variable A
- SS(B | A, AB) for independent variable B

this is the default in the literature (e.g. SPSS, SAS, Stata etc use it)

38

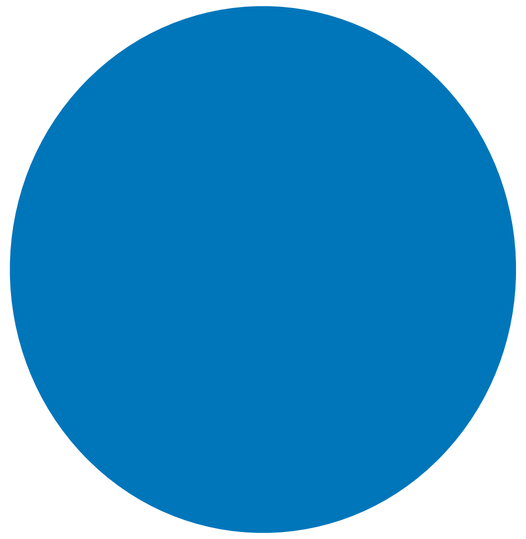
Unbalanced design

- There are different kinds of ANOVAs, for which the sums of squares are calculated differently.
- This makes a difference when we have an unbalanced design (i.e. the number of participants is not the same for each cell in our design).

`joint_tests()` is your friend!

Quick recap: Unbalanced designs

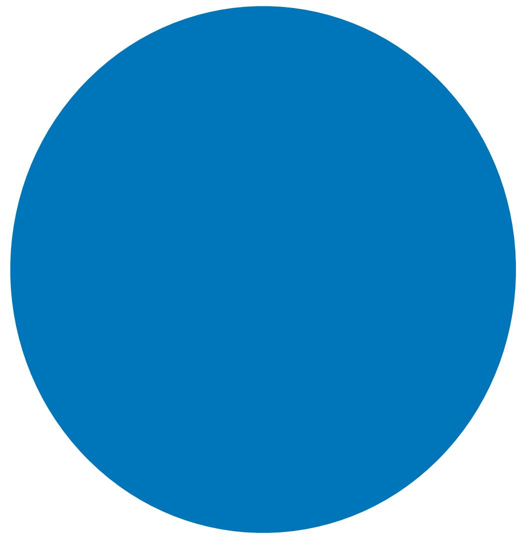
balanced designs



predictors are **uncorrelated**
(when we use sum contrasts to code categorical variables)

order of entering predictors doesn't matter ...

unbalanced designs



predictors are **correlated**

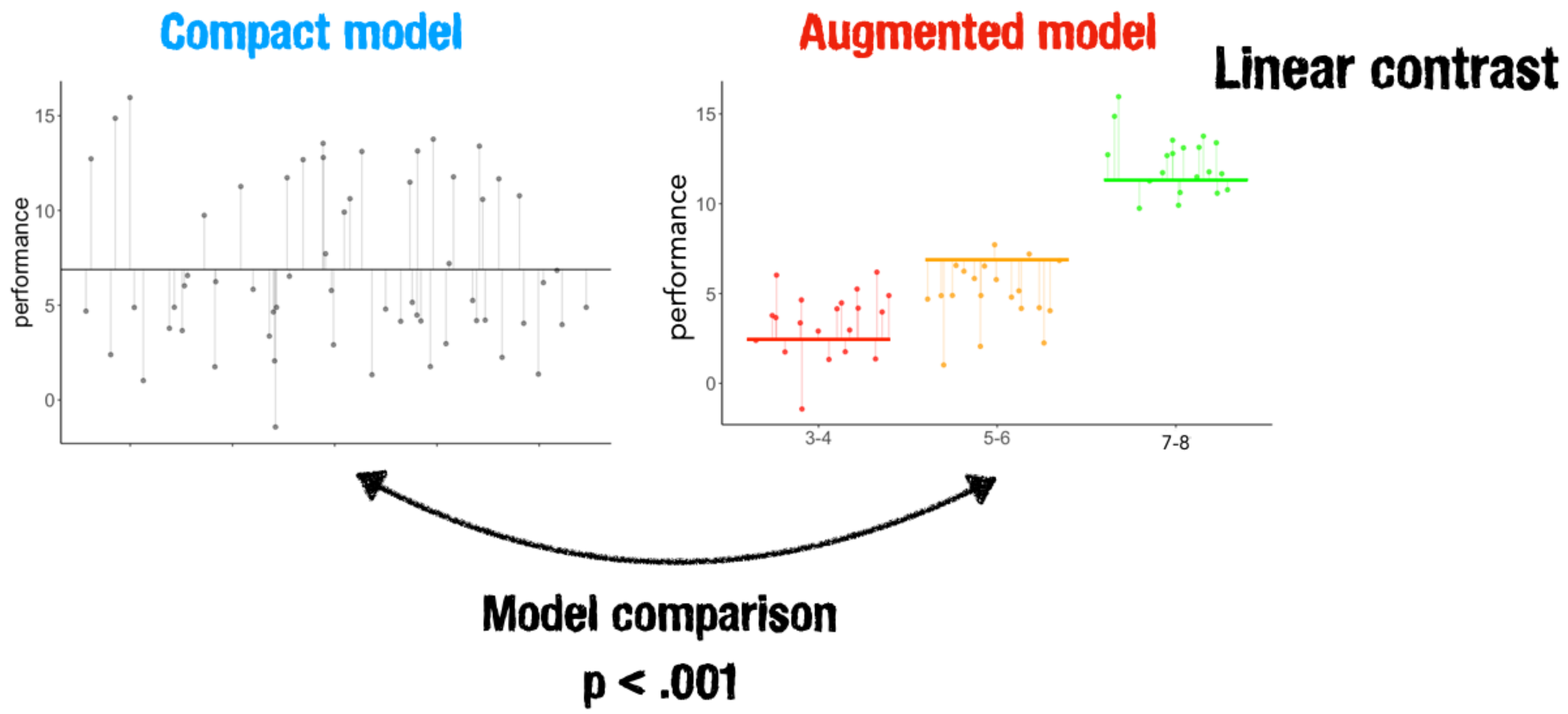
order of entering predictors would matter, so use Type 3 sums of squares

Quick recap: Linear contrasts

Contrasts

Does performance increase with age?

```
contrasts = c(-1, 0, 1)
```



46

Defining contrasts

- groups that we don't want to include in the comparison get a 0
- groups that we want to compare with one another should sum to 0
- this also means that all the contrasts together should sum to 0

Example:

```
contrasts = list(young_vs_old = c(-1, -1, 2),
                 three_vs_five = c(-1, 1, 0))
```

Linear contrasts in R

```r
library("emmeans") # for calculating contrasts

#  fit the linear model
fit = lm(formula = performance ~ group,
         data = df.development)

# check factor levels
levels(df.development$group)

# define the contrasts of interest
contrasts = list(young_vs_old = c(-0.5, -0.5, 1),
                 three_vs_five = c(-0.5, 0.5, 0))

# compute significance test on contrasts
fit %>%
  emmeans("group",
          contr = contrasts,
          adjust = "bonferroni") %>%
  pluck("contrasts")
```

```
[1] "3-4" "5-6" "7-8"
```

compute the results

```
[1] "3-4" "5-6" "7-8"
 contrast       estimate        SE df t.ratio p.value
 young_vs_old  16.093541 0.4742322 57  33.936  <.0001
 three_vs_five  1.606009 0.5475962 57   2.933  0.0097

P value adjustment: bonferroni method for 2 tests
```

53

Generalized linear model

Titanic dataset

Titanic data set

891 passengers

passenger_id	survived	pclass	name	sex	age	sib_sp	parch	ticket	fare	cabin	embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs	female	38	1	0	PC 17599	71.28	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.92		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.10	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. James	male	NA	0	0	330877	8.46		Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.86	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.07		S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth	female	27	0	2	347742	11.13		S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.07		C

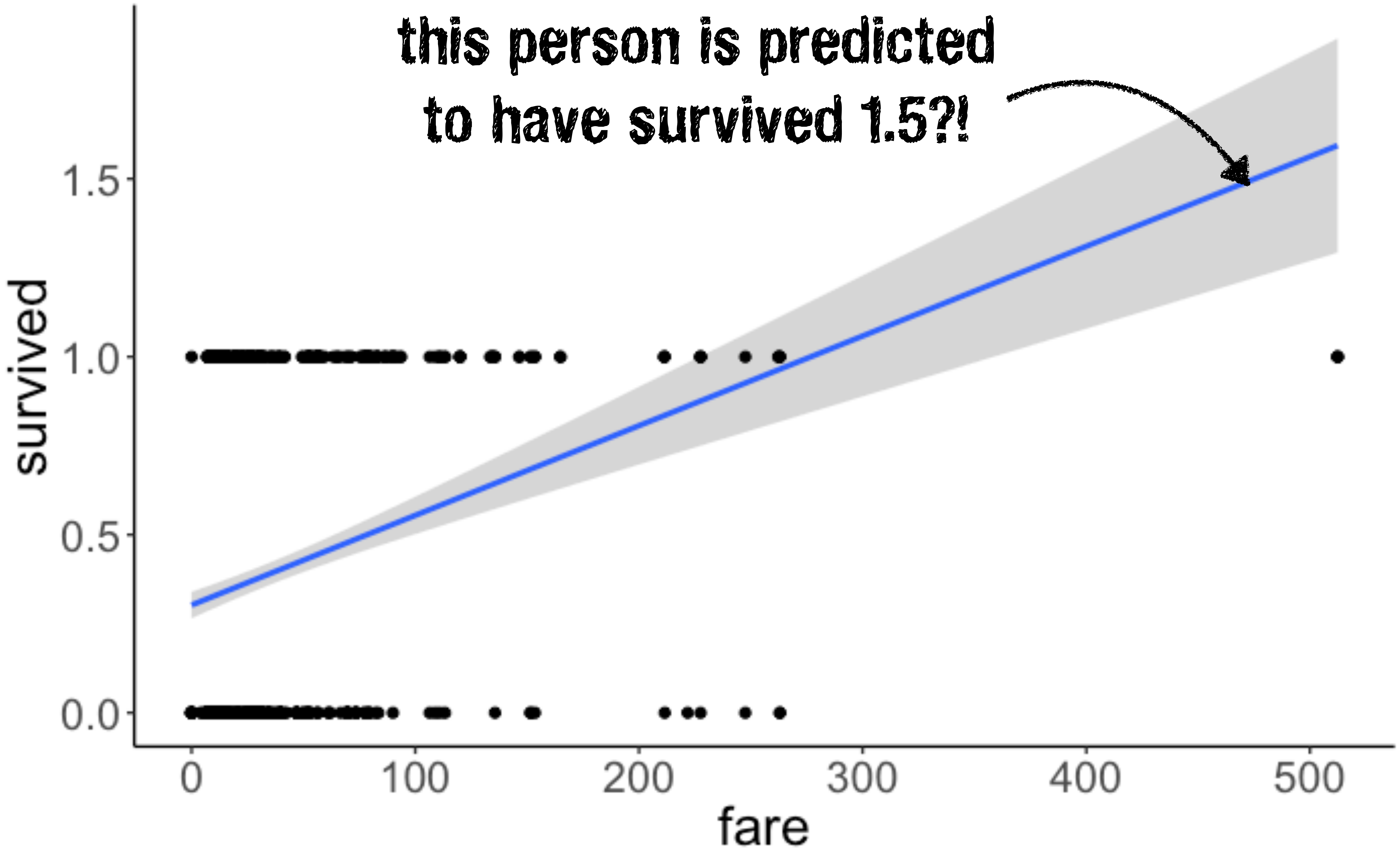
Is there a relationship between `fare` and `survived`?

```r
fit.lm = lm(formula = survived ~ 1 + fare,
            data = df.titanic)

fit.lm %>% summary()
```

```
Call:
lm(formula = survived ~ 1 + fare, data = df.titanic)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.9653 -0.3391 -0.3222  0.6044  0.6973 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 0.3026994  0.0187849  16.114  < 2e-16 ***
fare        0.0025195  0.0003174   7.939 6.12e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4705 on 889 degrees of freedom
Multiple R-squared:  0.06621, Adjusted R-squared:  0.06516 
F-statistic: 63.03 on 1 and 889 DF,  p-value: 6.12e-15
```

How should we interpret this parameter?

Is there a relationship between `fare` and `survived`?



Generalized linear model

- so far, we have only looked at situations where our dependent variable was continuous
- what about situations in which we have a binary dependent variable?
  - survived vs. died
  - correct vs. incorrect
  - benign vs. malignent
  - yes vs. no
  - ...

Logistic regression

Demo

http://mfviz.com/binary-predictions/

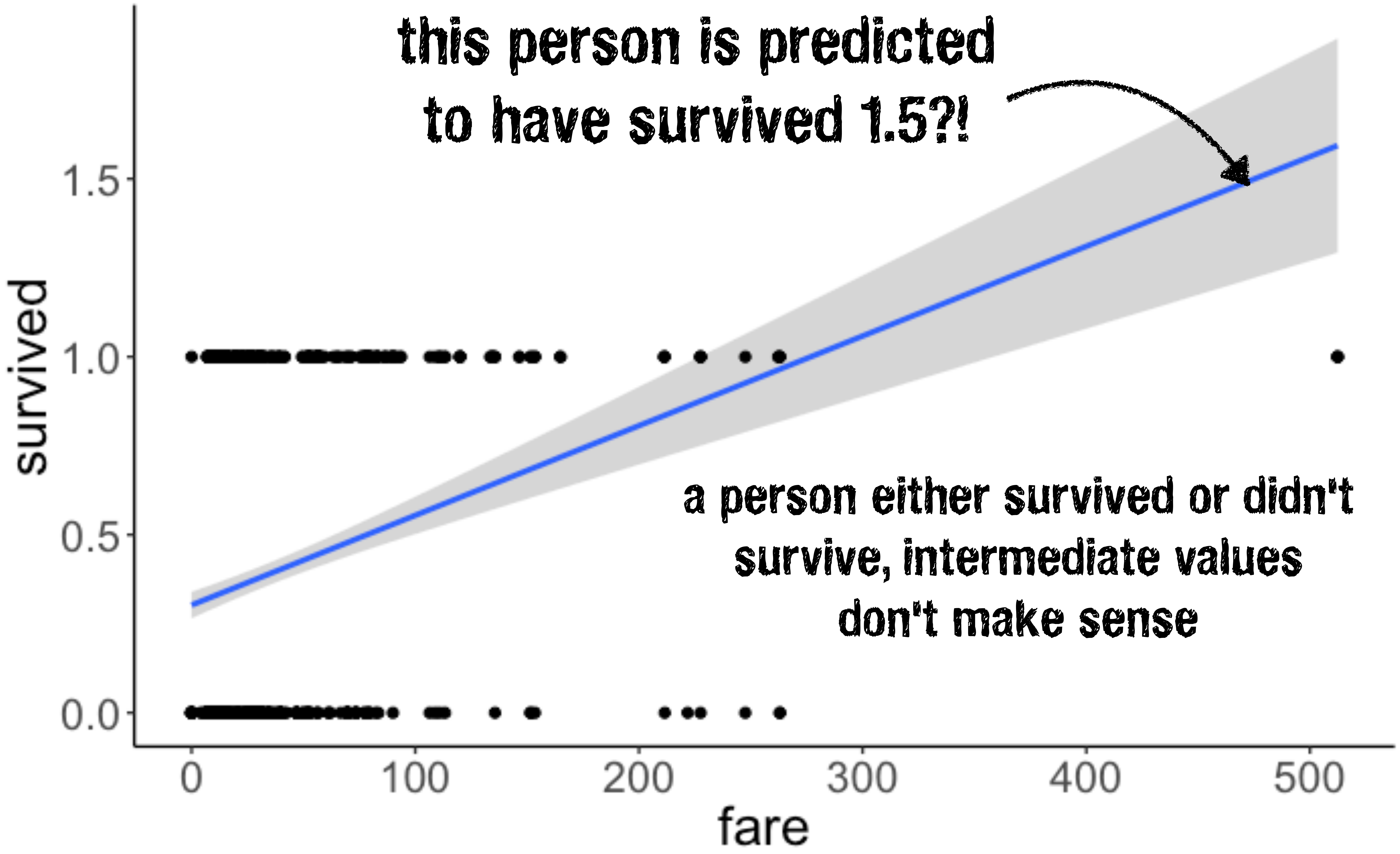
Is there a relationship between `fare` and `survived`?

Can we still use a linear model to make predictions about a binary outcome variable?

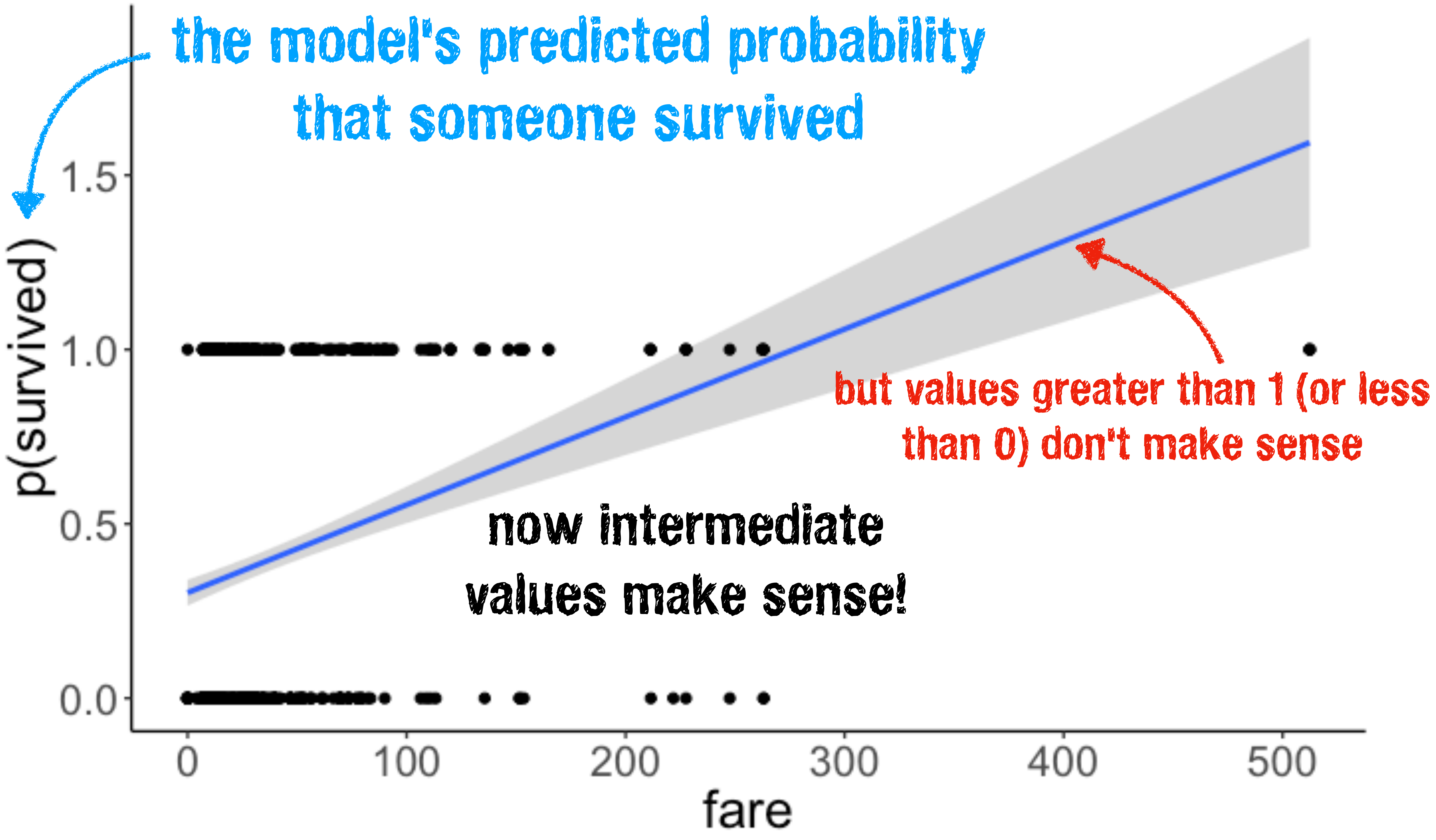
The fact that this class is called "**Generalized linear model**" suggests we can!

Is there a relationship between `fare` and `survived`?

```
fit.lm = lm(formula = survived ~ 1 + fare, data = df.titanic)
```



Is there a relationship between `fare` and `survived`?



From linear regression to logistic regression

$$Y_i = b_0 + b_1 \cdot X_i + e_i$$

predict the value of Y

$$\pi_i = b_0 + b_1 \cdot X_i + e_i$$

predict the probability of Y

$$\pi_i = P(Y_i = 1)$$

let's just do a **logit** transform

we need to map from **[-∞, +∞]** to **[0, 1]**

Logit transform

$$\pi_i = b_0 + b_1 \cdot X_i + e_i$$

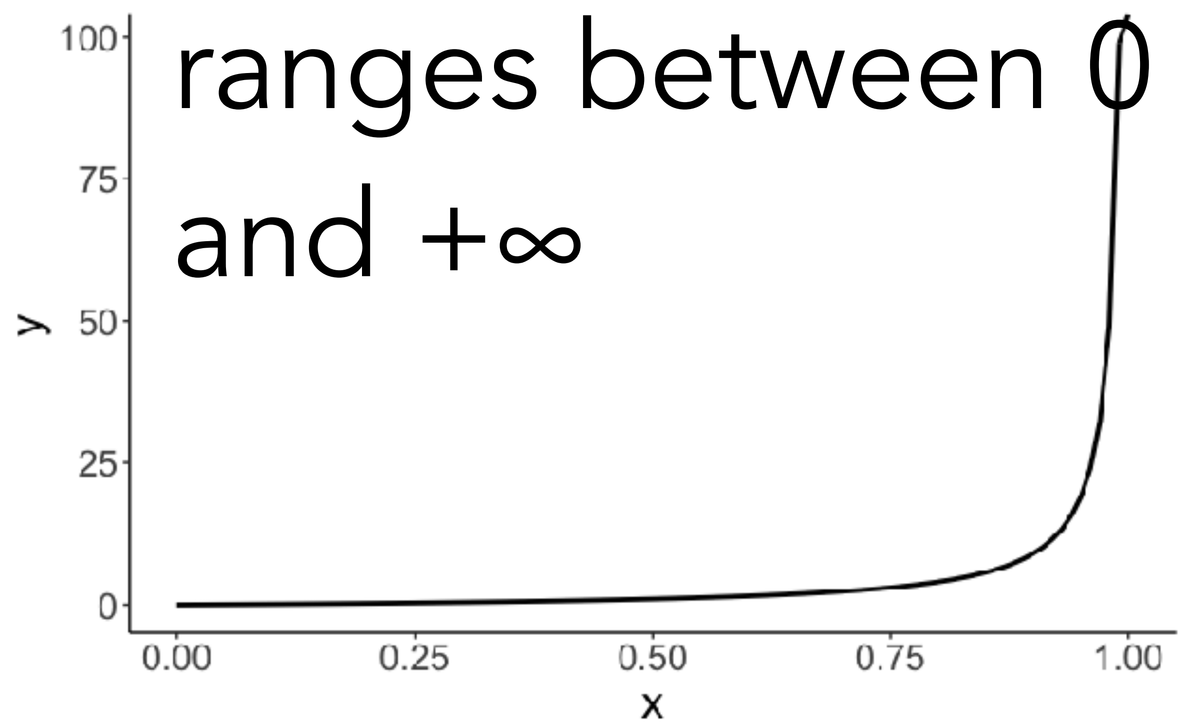
predict the probability of Y

we need to transform the dependent variable so that it can take any value between -∞ and +∞ (we can then transform it back into a probability later)

$$\pi_i = P(Y_i = 1)$$

Step 1: Calculate the "odds"

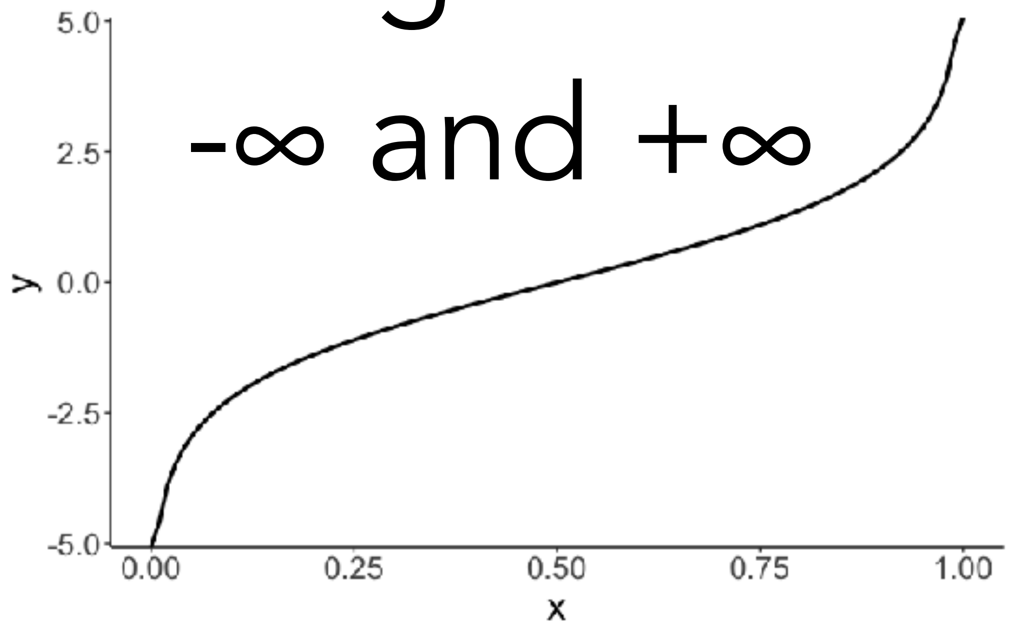
$$\frac{P(Y_i = 1)}{P(Y_i = 0)} = \frac{\pi_i}{1 - \pi_i}$$



Step 2: Take the (natural) log

$$\ln\left(\frac{\pi_i}{1 - \pi_i}\right) = b_0 + b_1 \cdot X_i + e_i$$

ranges between



Logit transform

after transforming from a binary variable, to a probability, to odds, to log odds, the model looks like a normal linear model

log odds

$$\ln\left(\frac{\pi_i}{1-\pi_i}\right) = b_0 + b_1 \cdot X_i + e_i$$

$$\pi_i = P(Y_i = 1)$$

if log odds == 0: $P(Y_i = 1) = P(Y_i = 0)$

if log odds > 0: $P(Y_i = 1) > P(Y_i = 0)$

if log odds < 0: $P(Y_i = 1) < P(Y_i = 0)$

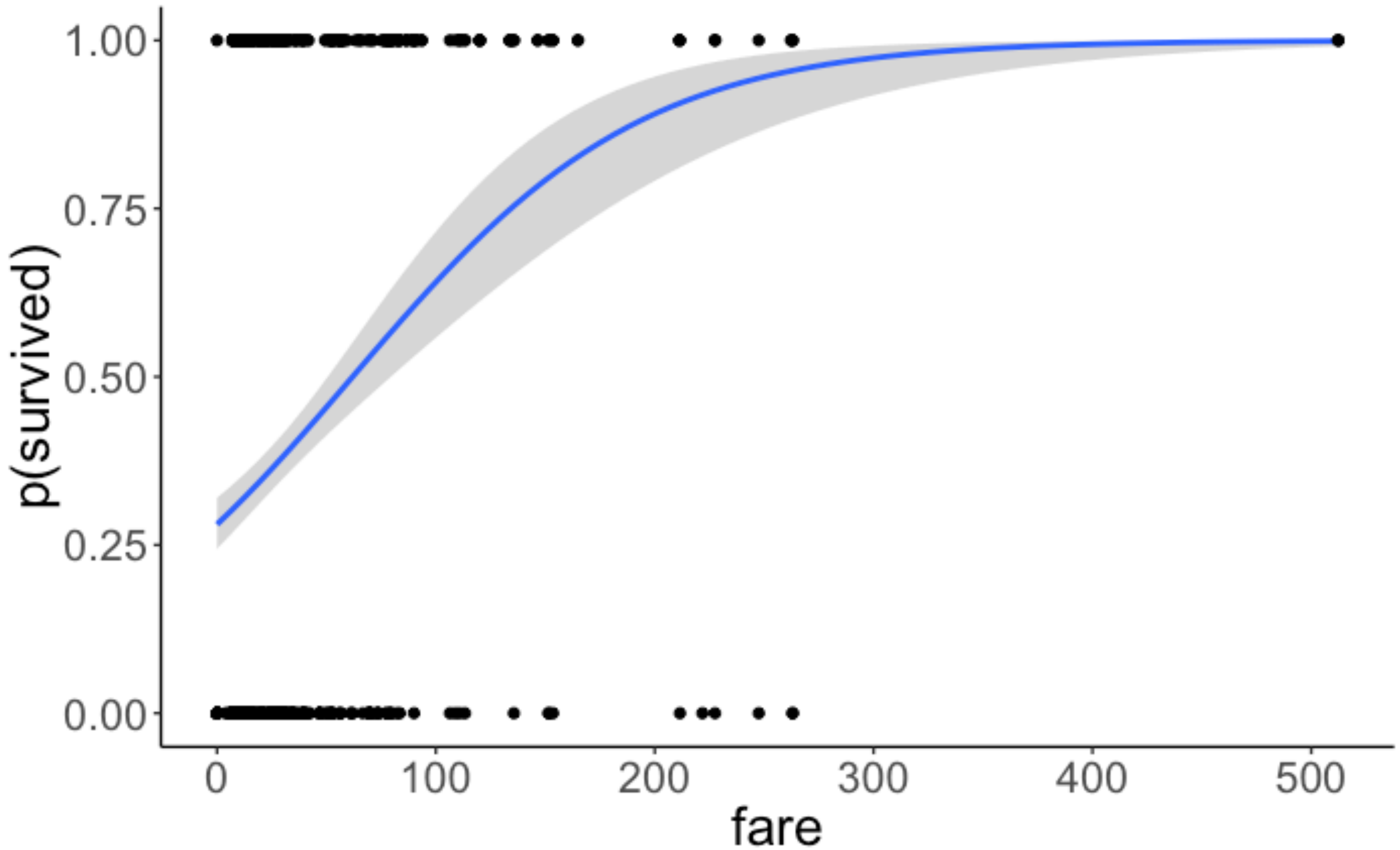
Fitting a logistic regression in R

```r
fit.glm = glm(formula = survived ~ 1 + fare,
              family = "binomial",
              data = df.titanic)

fit.glm %>% summary()
```

```
Call:
glm(formula = survived ~ 1 + fare, family = "binomial", data = df.titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4906  -0.8878  -0.8531   1.3429   1.5942

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1186.7  on 890  degrees of freedom
Residual deviance: 1117.6  on 889  degrees of freedom
AIC: 1121.6

Number of Fisher Scoring iterations: 4
```

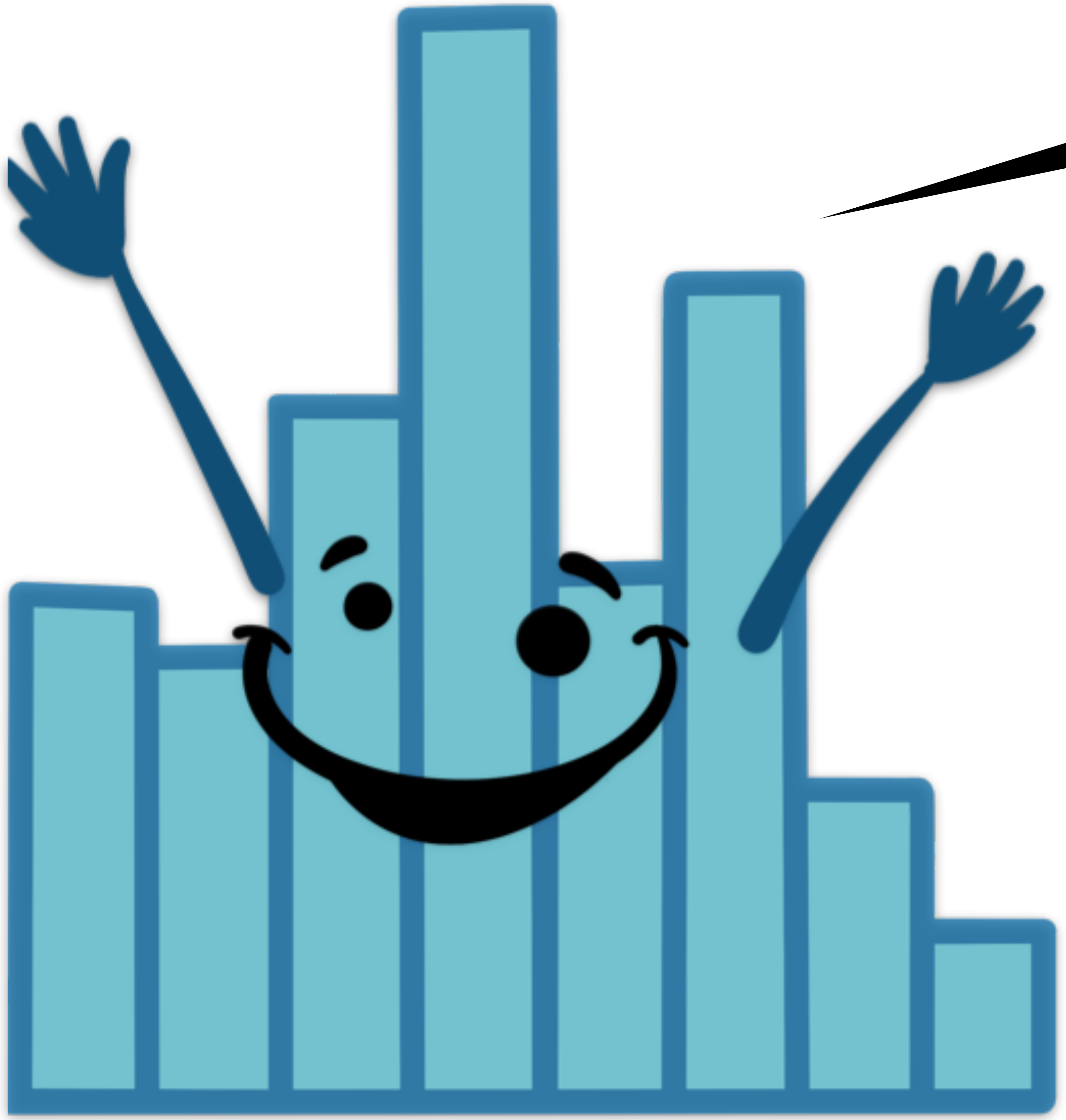
Visualize the model's predictions

```r
ggplot(data = df.titanic,
       mapping = aes(x = fare,
                     y = survived)) +
  geom_smooth(method = "glm",
              method.args = list(family = "binomial")) +
  geom_point() +
  labs(y = "p(survived)")
```



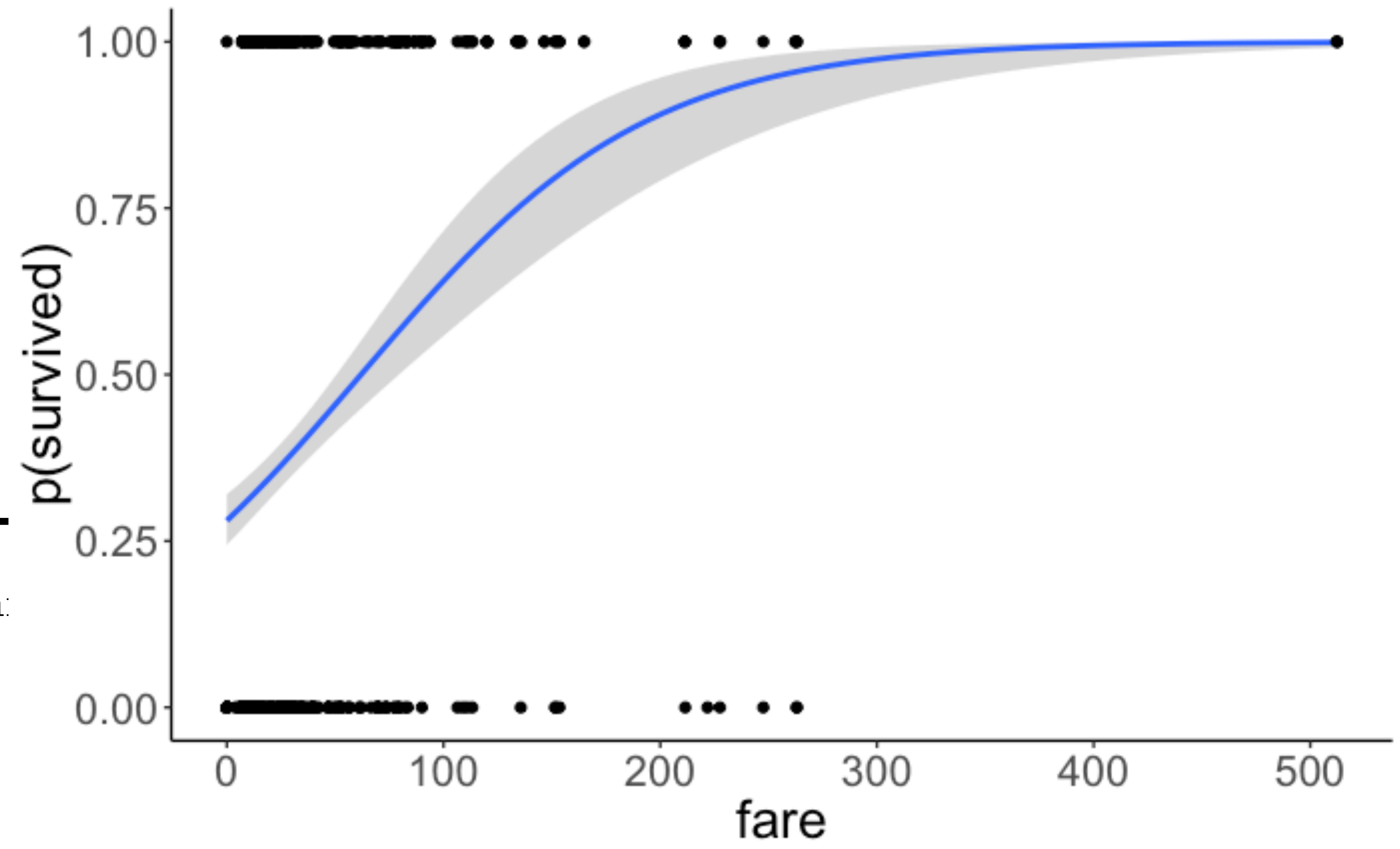
02:00

stretch break!



Interpreting the model output

Interpreting the model output




```
Call:
glm(formula = survived ~ 1 + fare, fa

Deviance Residuals:
    Min       1Q   Median       3Q
-2.4906  -0.8878  -0.853    1.3429

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1186.7  on 890  degrees of freedom
Residual deviance: 1117.6  on 889  degrees of freedom
AIC: 1121.6

Number of Fisher Scoring iterations: 4
```

log odds ?

Transform log odds into probability

$$\pi = P(Y = 1)$$

just a placeholder

$$\ln\left(\frac{\pi}{1-\pi}\right) = V$$

logit transformation

$$\pi = \frac{e^V}{1+e^V}$$

inverse logit

gives us back the probability
(which is much easier to interpret)

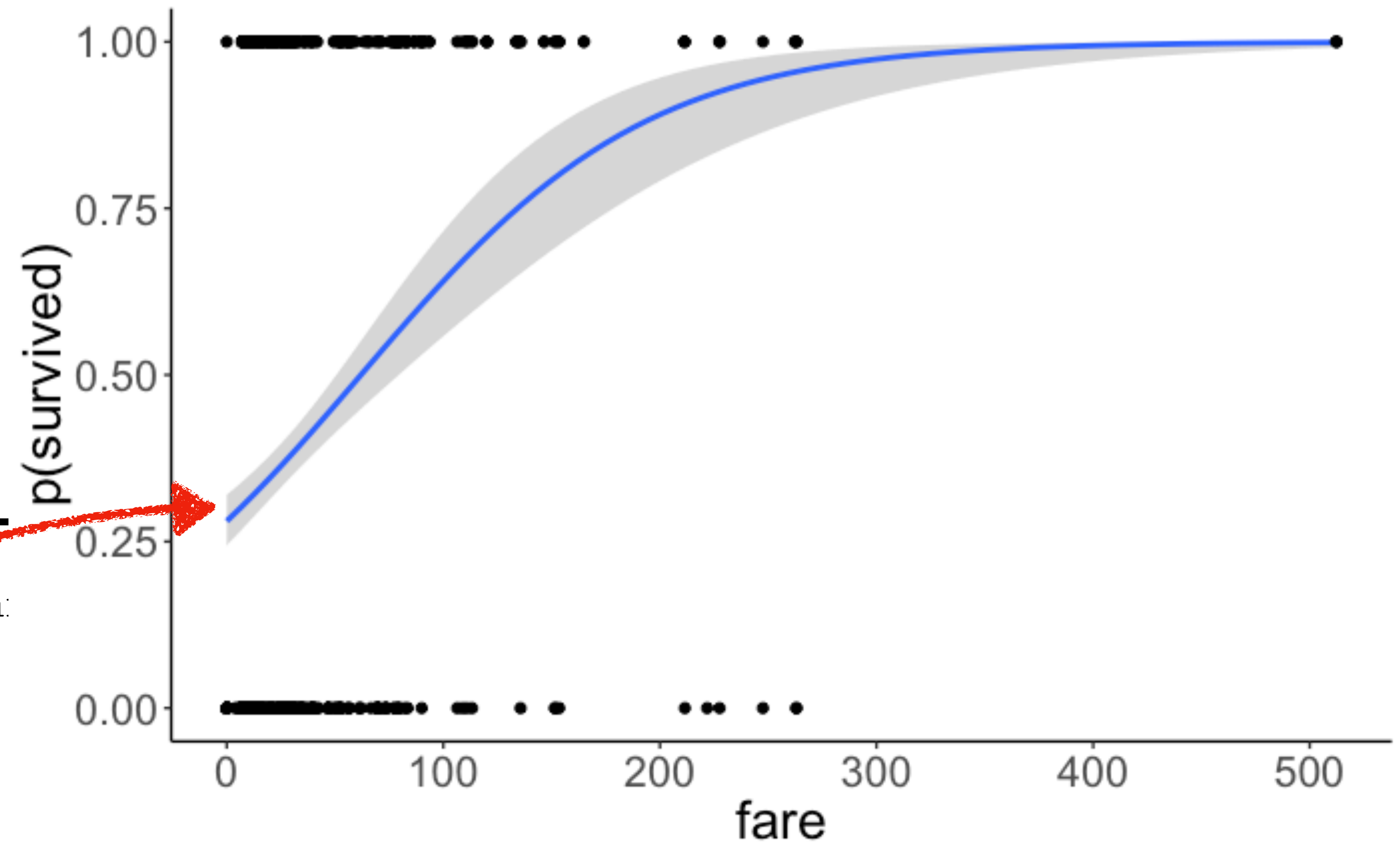
$$\pi_i = \frac{e^{b_0+b_1 \cdot X_i}}{1+e^{b_0+b_1 \cdot X_i}}$$

another way to
specify the model

Interpreting the model output

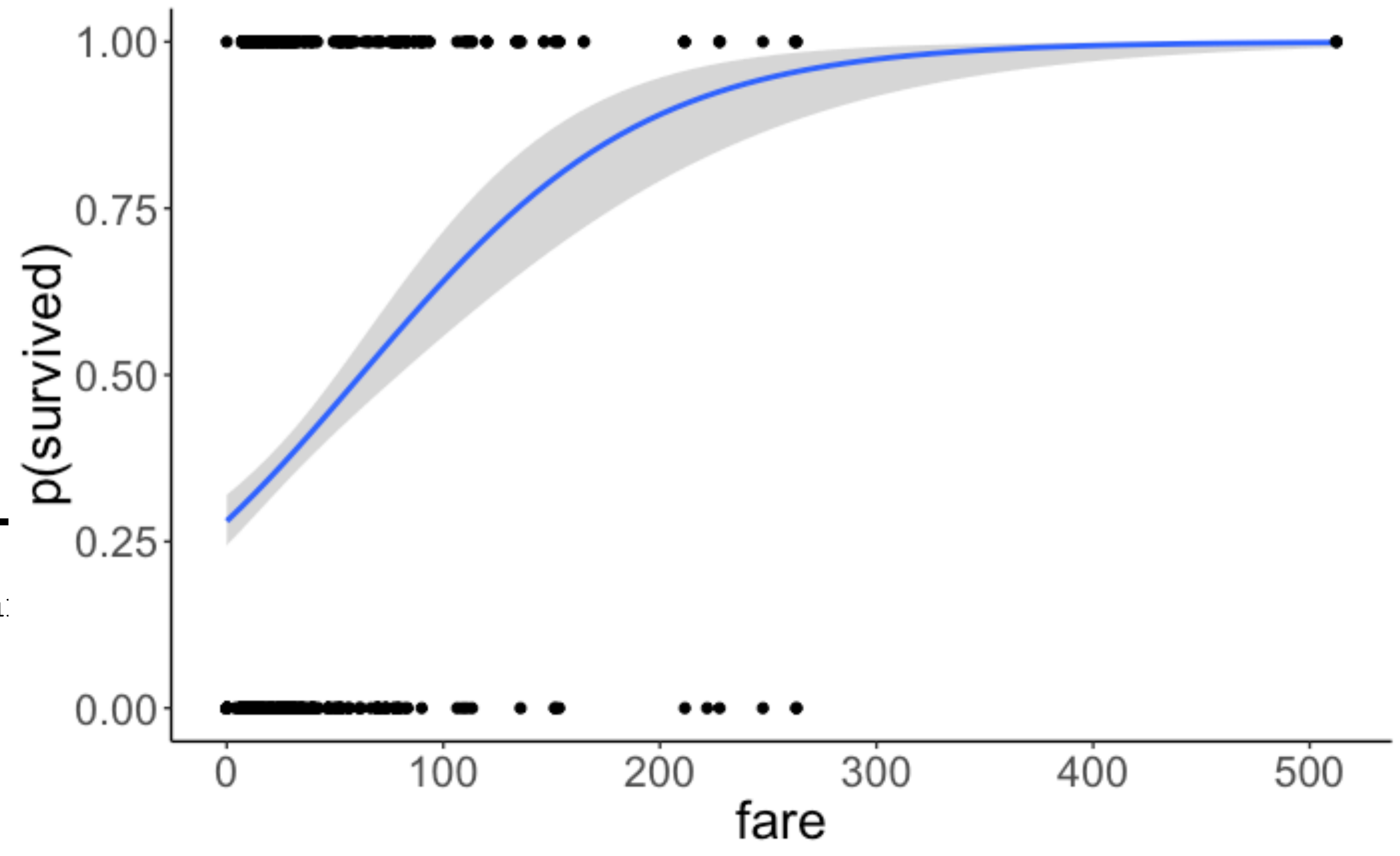
inverse logit

$$\pi = \frac{e^{-0.94}}{1 + e^{-0.94}} \approx 0.28$$




```
Call:
glm(formula = survived ~ 1 + fare, fa

Deviance Residuals:
    Min       1Q   Median       3Q
-2.4906  -0.8878  -0.8531   1.3429

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1186.7  on 890  degrees of freedom
Residual deviance: 1117.6  on 889  degrees of freedom
AIC: 1121.6

Number of Fisher Scoring iterations: 4
```

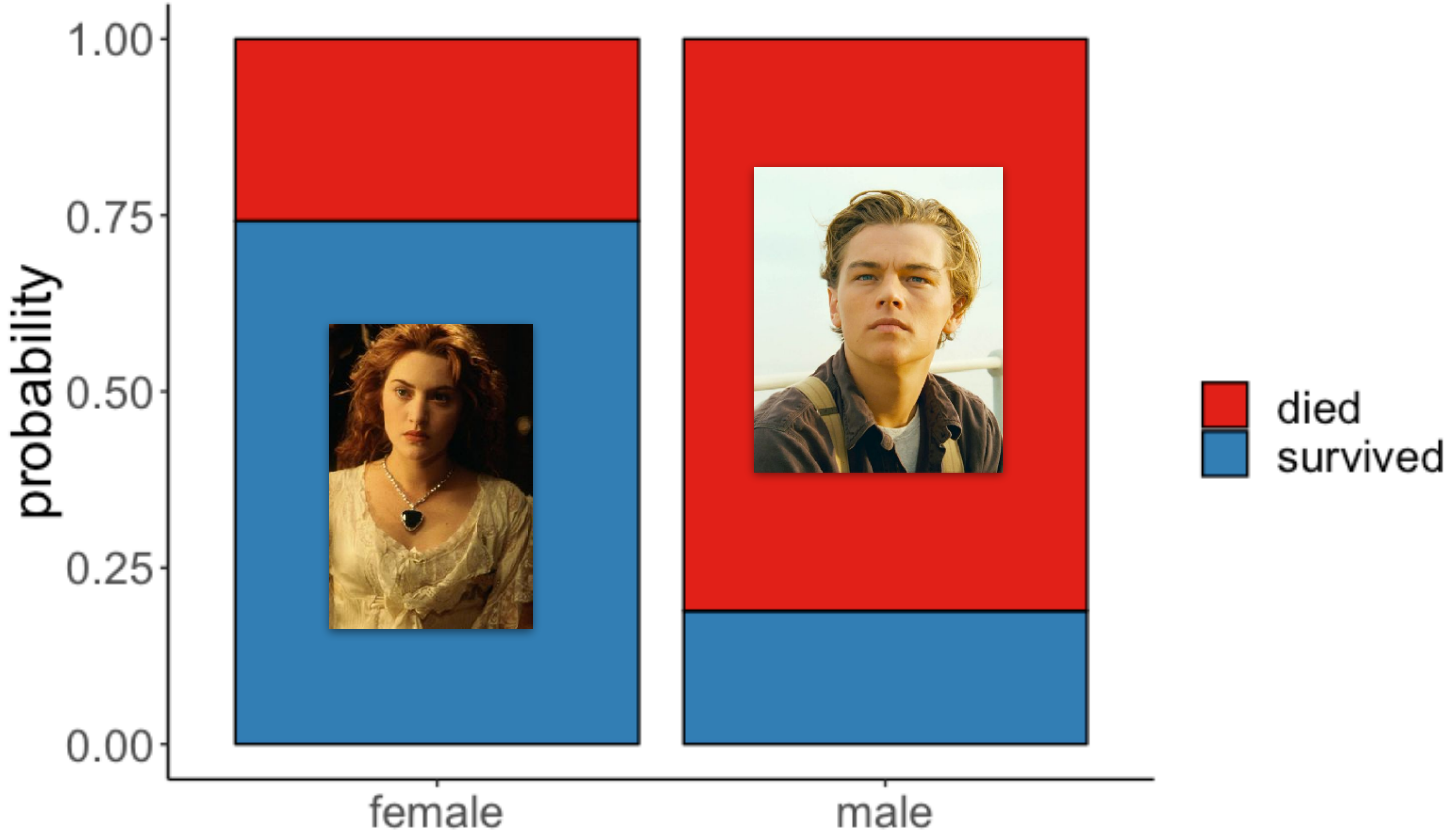
Interpreting the model output




```
Call:
glm(formula = survived ~ 1 + fare, fa

Deviance Residuals:
    Min       1Q   Median       3Q
-2.4906  -0.8878  -0.8521   1.3429

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1186.7  on 890  degrees of freedom
Residual deviance: 1117.6  on 889  degrees of freedom
AIC: 1121.6

Number of Fisher Scoring iterations: 4
```

Let's consider a binary predictor

Was the probability of survival different between female and male passengers on the Titanic?



Let's consider a binary predictor

```r
fit.glm2 = glm(formula = survived ~ sex,
               family = "binomial",
               data = df.titanic)

fit.glm2 %>% summary()
```

```
Call:
glm(formula = survived ~ sex, family = "binomial", data = df.titanic)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6462  -0.6471  -0.6471   0.7725   1.8256

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.0566     0.1290   8.191 2.58e-16 ***
sexmale      -2.5137     0.1672 -15.036  < 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: 1186.7  on 890  degrees of freedom
Residual deviance:  917.8  on 889  degrees of freedom
AIC: 921.8

Number of Fisher Scoring iterations: 4
```

sex was significantly associated with survival

Let's consider a binary predictor

$$\ln\left(\frac{\text{p(survived)}_i}{1 - \text{p(survived)}_i}\right) = b_0 + b_1 \cdot \text{sex}_i$$

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.0566     0.1290   8.191 2.58e-16 ***
sexmale      -2.5137     0.1672 -15.036  < 2e-16 ***
```

sex	survived	n	p	p(survived I sex)
female	0	81	0.09	0.26
female	1	233	0.26	0.74
male	0	468	0.53	0.81
male	1	109	0.12	0.19

if sex == 0:

$$\ln\left(\frac{\widehat{\text{p(survived)}}_i}{1 - \text{p(survived)}_i}\right) = b_0$$

$$\text{p(survived)}_i = \frac{e^{b_0}}{1 + e^{b_0}} = 0.74$$

Let's consider a binary predictor

$$\ln\left(\frac{\text{p(survived)}_i}{1 - \text{p(survived)}_i}\right) = b_0 + b_1 \cdot \text{sex}_i$$

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.0566     0.1290   8.191 2.58e-16 ***
sexmale      -2.5137     0.1672 -15.036  < 2e-16 ***
```

sex	survived	n	p	p(survived I sex)
female	0	81	0.09	0.26
female	1	233	0.26	0.74
male	0	468	0.53	0.81
male	1	109	0.12	0.19

if sex == 1:

$$\ln\left(\frac{\widehat{\text{p(survived)}}_i}{1 - \text{p(survived)}_i}\right) = b_0 + b_1$$

$$\text{p(survived)}_i = \frac{e^{b_0+b_1}}{1 + e^{b_0+b_1}} = 0.19$$

Now let's go back to a continuous predictor

$$\ln(\frac{\text{p(survived)}_i}{1 - \text{p(survived)}_i}) = b_0 + b_1 \cdot \text{fare}_i$$

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
```

fare	prediction	p(survival)
0	-0.94	0.28
10	-0.79	0.31
50	-0.18	0.45
100	0.58	0.64
500	6.66	1.00

$$\ln(\frac{\widehat{\text{p(survived)}}}{1 - \text{p(survived)}}) = -0.94 + 0.015 \cdot 10$$

$$\text{p(survived)}_i = \frac{e^{-0.94 + 0.015 \cdot 10}}{1 + e^{-0.94 + 0.015 \cdot 10}} = 0.31$$

Now let's go back to a continuous predictor

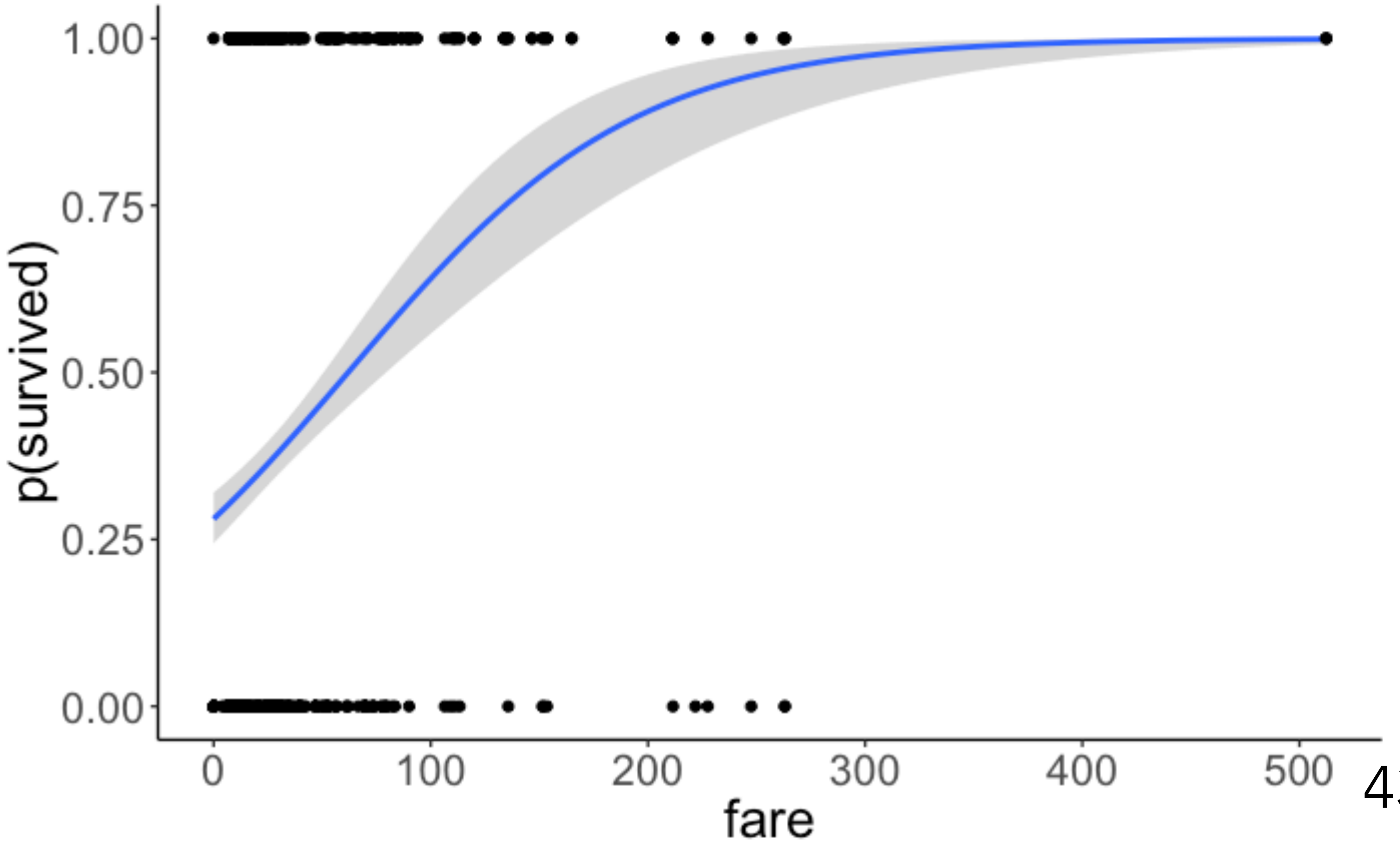
$$\ln\left(\frac{\text{p(survived)}_i}{1 - \text{p(survived)}_i}\right) = b_0 + b_1 \cdot \text{fare}_i$$

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.941330   0.095129  -9.895  < 2e-16 ***
fare         0.015197   0.002232   6.810 9.79e-12 ***
```

For a one-unit increase in the fare, the expected increase in the odds of survival is 16%.

$$e^{0.015} \approx 1.16$$

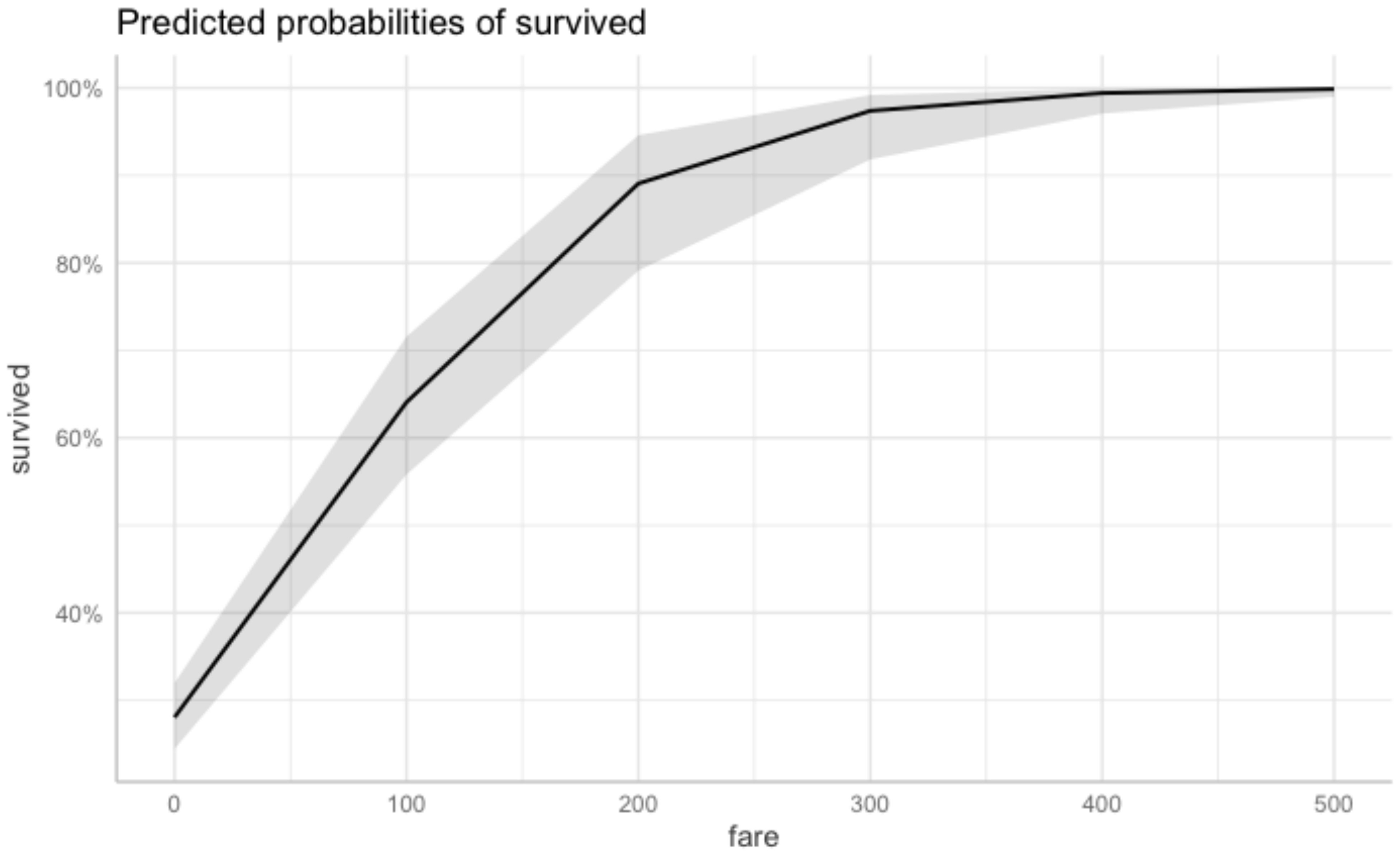
fare	prediction	p(survival)
0	-0.94	0.28
10	-0.79	0.31
50	-0.18	0.45
100	0.58	0.64
500	6.66	1.00



Do we have to do this by hand?

```r
ggpredict(model = fit.glm,
          terms = "fare [0, 100, 200, 300, 400, 500]")
```

```
# Predicted probabilities of survived
# x = fare

  x | Predicted |       95% CI
--------------------------------
  0 |      0.28 | [0.24, 0.32]
100 |      0.64 | [0.56, 0.72]
200 |      0.89 | [0.79, 0.95]
300 |      0.97 | [0.92, 0.99]
400 |      0.99 | [0.97, 1.00]
500 |      1.00 | [0.99, 1.00]
```



Models with several predictors

$$\ln\left(\frac{\text{p(survived)}_i}{1 - \text{p(survived)}_i}\right) = b_0 + b_1 \cdot \text{sex}_i + b_2 \cdot \text{fare}_i$$

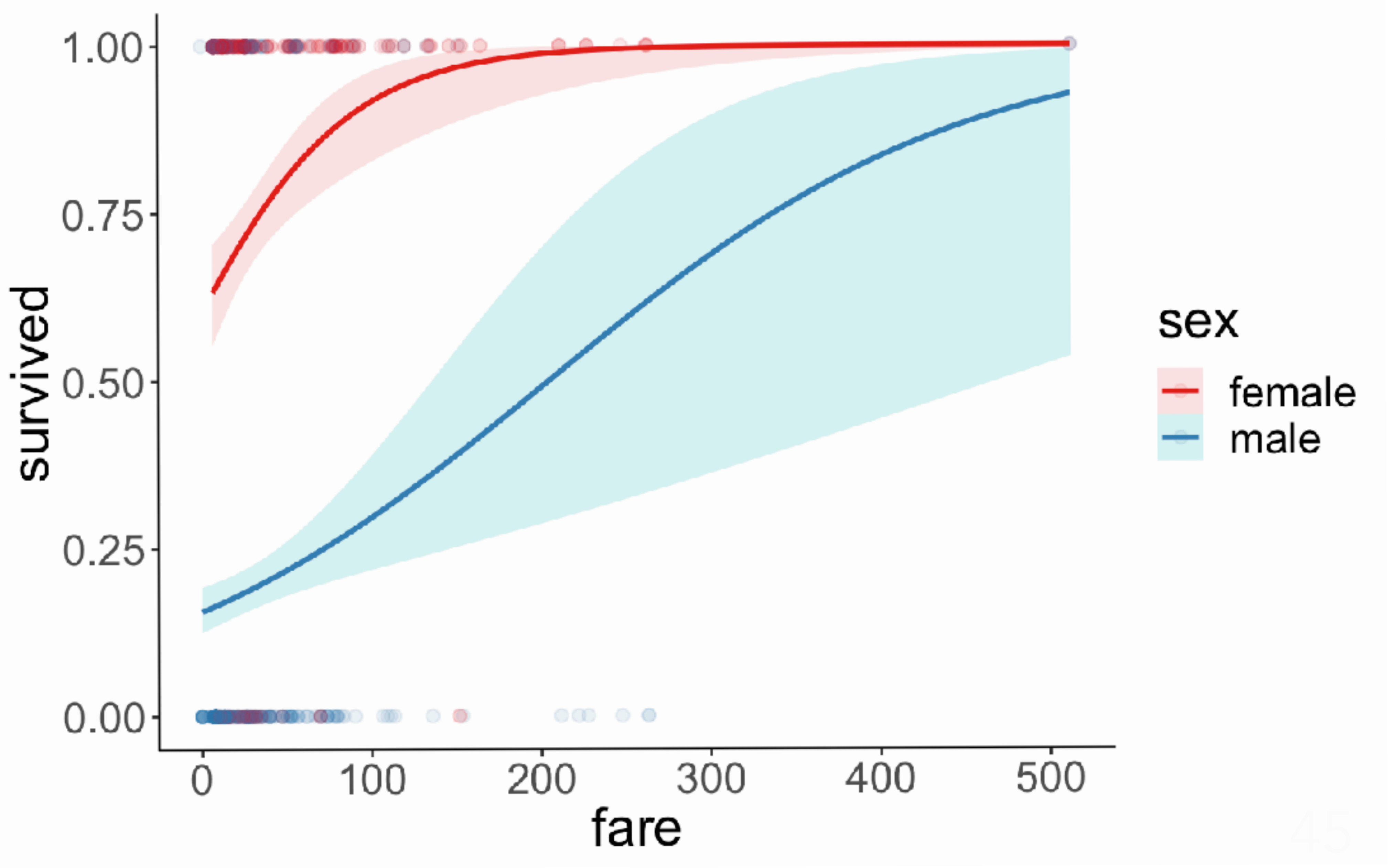
```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.647100   0.148502   4.358 1.32e-05 ***
sexmale     -2.422760   0.170515 -14.208  < 2e-16 ***
fare         0.011214   0.002295   4.886 1.03e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

controlling for "fare" there is still a significant difference between female and male passengers

```
ggpredict(fit.glm,
          terms = c("sex"))
```

```
# Predicted values of survived
# x = sex

x      | Predicted |   SE |        95% CI
-----------------------------------------
female |      0.73 | 0.13 | [0.68, 0.78]
male   |      0.20 | 0.11 | [0.16, 0.23]

Adjusted for:
* fare = 32.20
```

```
df.titanic %>%
  mutate(sex = as.factor(sex)) %>%
  ggplot(data = .,
         mapping = aes(x = fare,
                       y = survived,
                       color = sex)) +
  geom_point(alpha = 0.1, size = 2) +
  geom_smooth(method = "glm",
              method.args = list(family = "binomial"),
              alpha = 0.2,
              aes(fill = sex)) +
  scale_color_brewer(palette = "Set1")
```



Fitting and reporting models

Simulating a logistic regression

```r
# make example reproducible
set.seed(1)

# set parameters
sample_size = 1000
b0 = 0
b1 = 1

# generate data
df.data = tibble(
  x = rnorm(n = sample_size),
  y = b0 + b1 * x,
  p = inv.logit(y)) %>%
  mutate(response = rbinom(n(), size = 1, p = p))

# fit model
fit = glm(formula = response ~ 1 + x,
          family = "binomial",
          data = df.data)

# model summary
fit %>% summary()
```

set some parameters

linear model (y is in log odds)

transform into probability

randomly draw response

fit a logistic regression

summarize the result

Simulating a logistic regression

```r
# make example reproducible
set.seed(1)

# set parameters
sample_size = 1000
b0 = 0
b1 = 1

# generate data
df.data = tibble(
  x = rnorm(n = sample_size),
  y = b0 + b1 * x,
  p = inv.logit(y)) %>%
  mutate(response = rbinom(n(), size = 1, p = p))

# fit model
fit = glm(formula = response ~ 1 + x,
          family = "binomial",
          data = df.data)

# model summary
fit %>% summary()
```

```
Call:
glm(formula = response ~ 1 + x, family = "binomial", data = df.data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1137  -1.0118  -0.4591   1.0287   2.2591

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.06214    0.06918  -0.898    0.369
x            0.92905    0.07937  11.705   <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: 1385.4  on 999  degrees of freedom
Residual deviance: 1209.6  on 998  degrees of freedom
AIC: 1213.6

Number of Fisher Scoring iterations: 3
```

Assessing the model fit

actual value predicted value

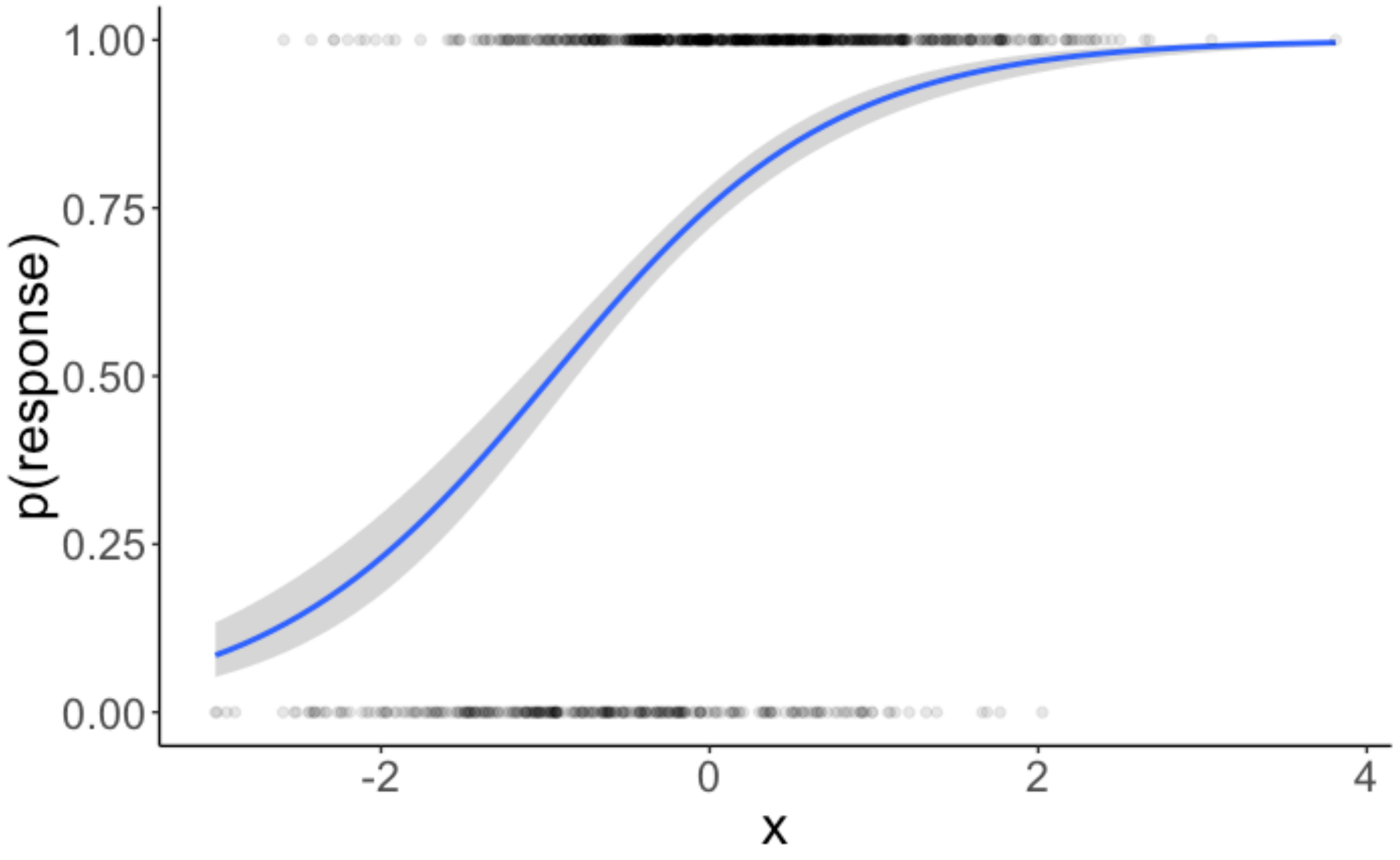
$$\text{log-likelihood} = \sum_{i=1}^{n} [Y_i \cdot \ln(P(Y_i)) + (1 - Y_i) \cdot \ln(1 - P(Y_i))]$$

- calculate the probability of the observed response
- take the log of these probabilities
- sum them up to get the log-likelihood of the data (given the model)

response	p(Y = 1)	p(Y = response)	log(p(Y = response))
1	0.34	0.34	-1.07
0	0.53	0.47	-0.75
1	0.30	0.30	-1.20
1	0.81	0.81	-0.22
1	0.56	0.56	-0.58
0	0.30	0.70	-0.36
1	0.60	0.60	-0.52
1	0.65	0.65	-0.43
1	0.62	0.62	-0.48
0	0.41	0.59	-0.54

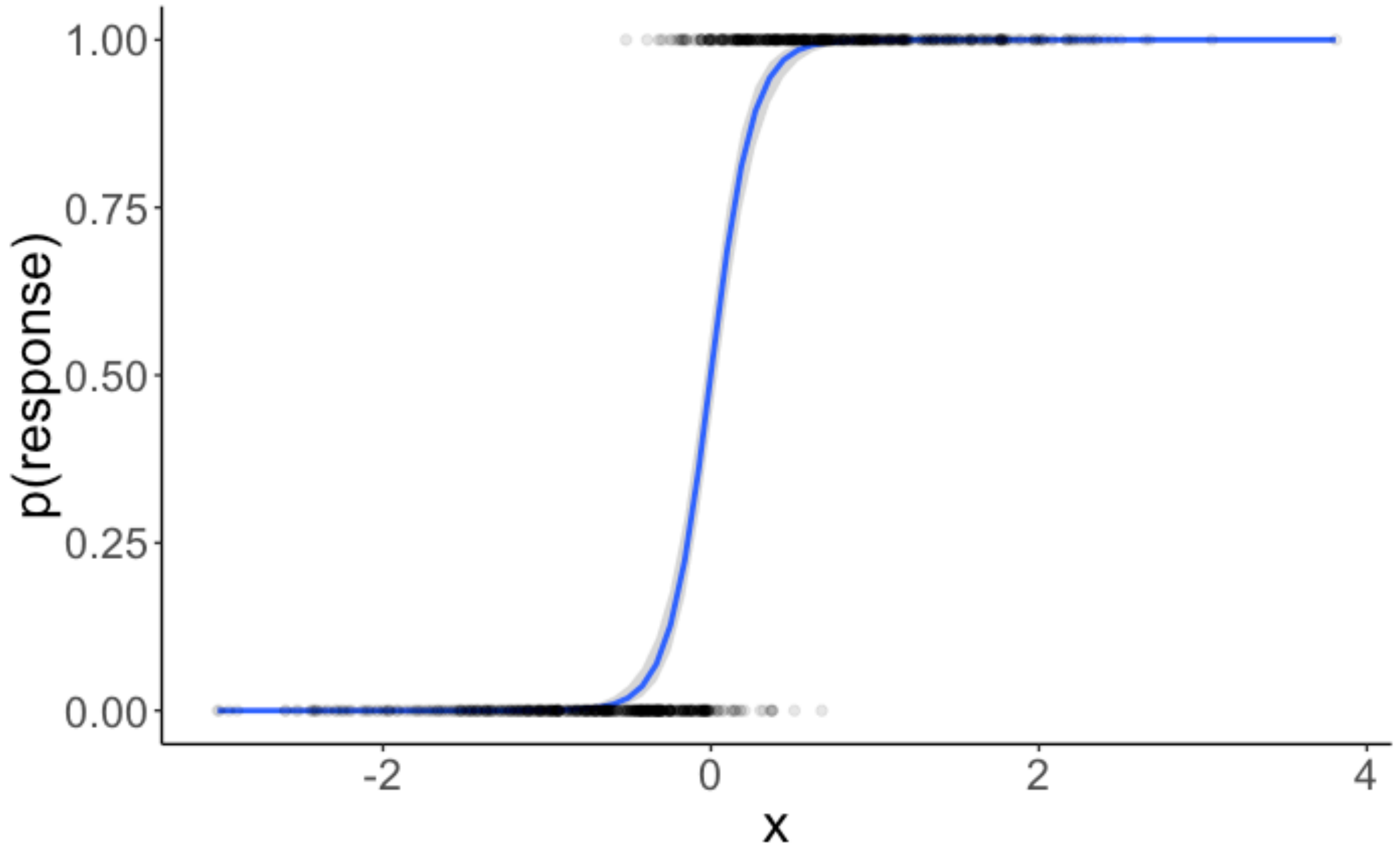
Assessing the model fit

doesn't predict the response very well



logLik	AIC	BIC
-501.65	1007.3	1017.12

predicts the response much better



logLik	AIC	BIC
-156.37	316.74	326.55

Testing hypotheses

aka checking whether it's **worth it**

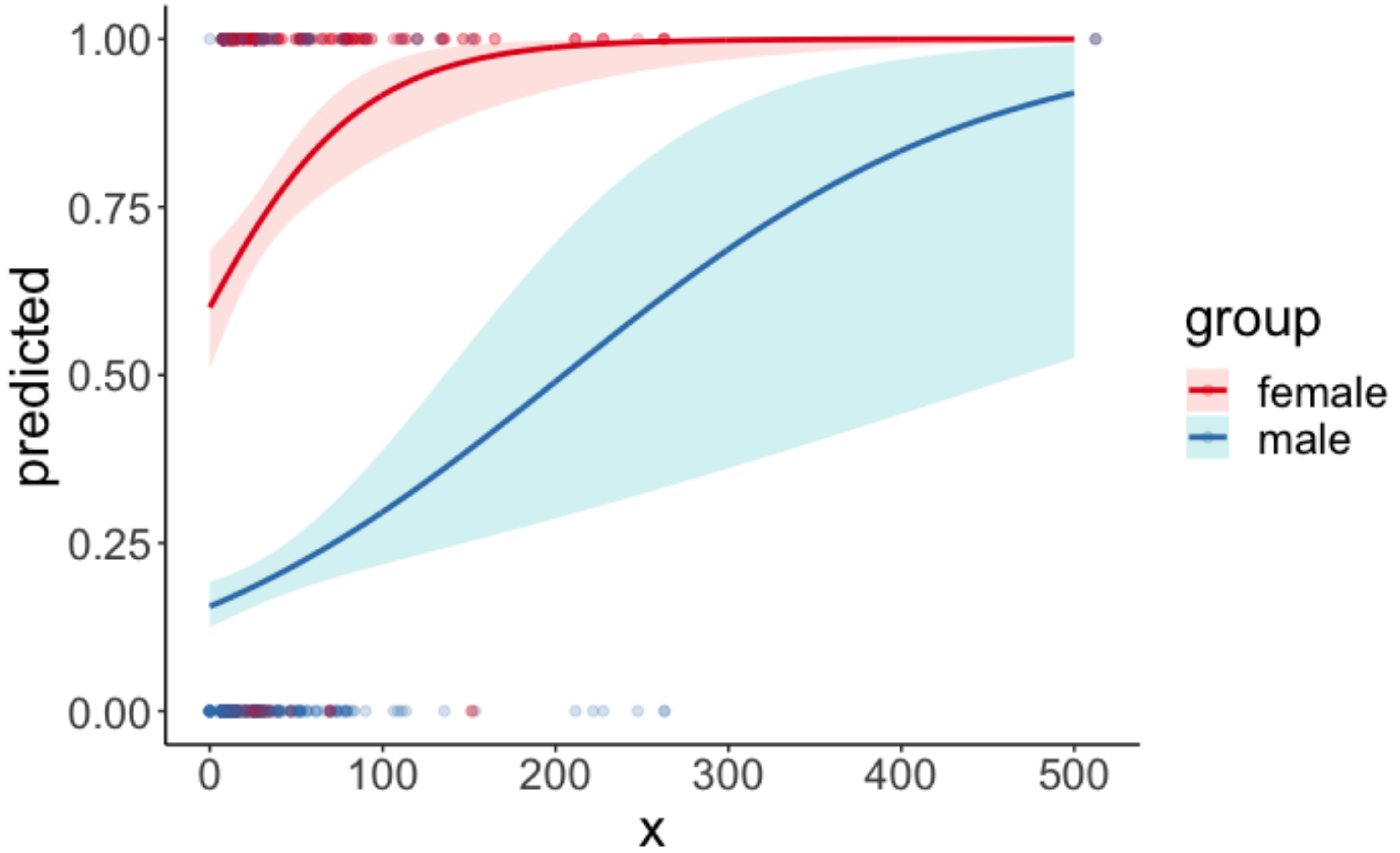
```r
# fit compact model
fit.compact = glm(formula = survived ~ 1 + fare,
                  family = "binomial",
                  data = df.titanic)

# fit augmented model
fit.augmented = glm(formula = survived ~ 1 + sex + fare,
                    family = "binomial",
                    data = df.titanic)

# likelihood ratio test
anova(fit.compact, fit.augmented, test = "LRT")
```

we need to specify that we want a likelihood ratio test

```
Analysis of Deviance Table

Model 1: survived ~ 1 + fare
Model 2: survived ~ 1 + sex + fare
  Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
1       889    1117.57
2       888     884.31  1   233.26 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Reporting results

- Visualize the data
- Show a table with the regression results
- Report significance of different factors
- Interpreting parameter estimates is tricky -- probably best to report probabilities for a few example cases




```
# Predicted values of survived
# x = fare

# sex = female

  x | Predicted |   SE |         95% CI
-----------------------------------------
  0 |      0.60 | 0.19 | [0.51, 0.69]
100 |      0.92 | 0.42 | [0.83, 0.96]
200 |      0.99 | 0.95 | [0.93, 1.00]
300 |      1.00 | 1.48 | [0.97, 1.00]
400 |      1.00 | 2.02 | [0.99, 1.00]
500 |      1.00 | 2.55 | [1.00, 1.00]

# sex = male

  x | Predicted |   SE |         95% CI
-----------------------------------------
  0 |      0.16 | 0.13 | [0.12, 0.19]
100 |      0.30 | 0.21 | [0.22, 0.39]
200 |      0.49 | 0.44 | [0.29, 0.70]
300 |      0.69 | 0.69 | [0.36, 0.90]
400 |      0.83 | 0.94 | [0.44, 0.97]
500 |      0.92 | 1.19 | [0.53, 0.99]
```

Assumptions

- linearity (between predictors and log odds)
- independence
- no multi-collinearity

- model fails to converge when there is **complete separation**:
  - if outcome variable can be perfectly predicted by a (combination of) predictor(s)

Different kinds of generalized models

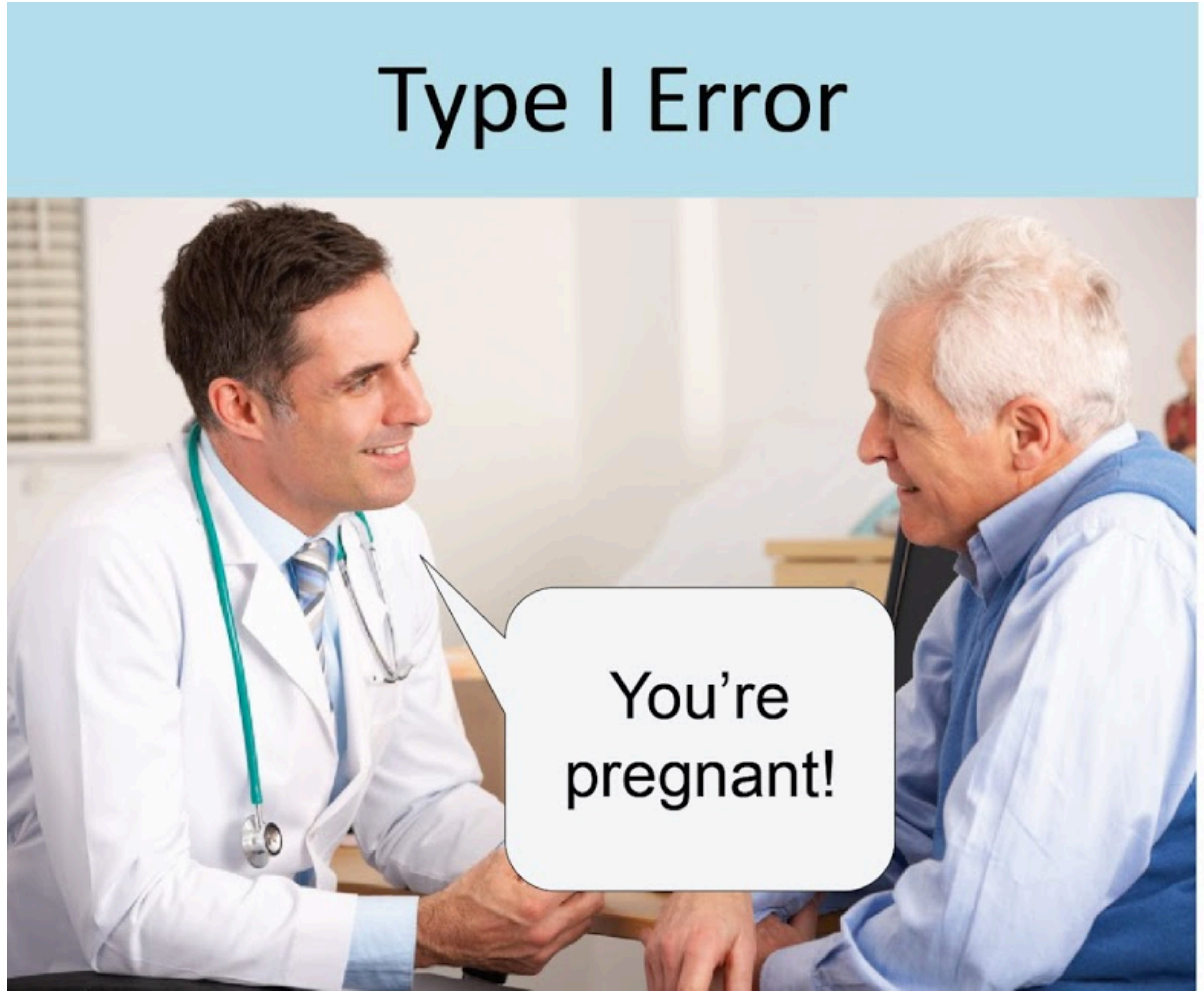
Different linking functions

```
binomial(link = "logit")

gaussian(link = "identity")

Gamma(link = "inverse")

inverse.gaussian(link = "1/mu^2")

poisson(link = "log")

quasi(link = "identity", variance = "constant")

quasibinomial(link = "logit")

quasipoisson(link = "log")
```

apply different transformations to the dependent variable

Power analysis

Making decisions



H_0: Not pregnant. **H_1**: Pregnant.

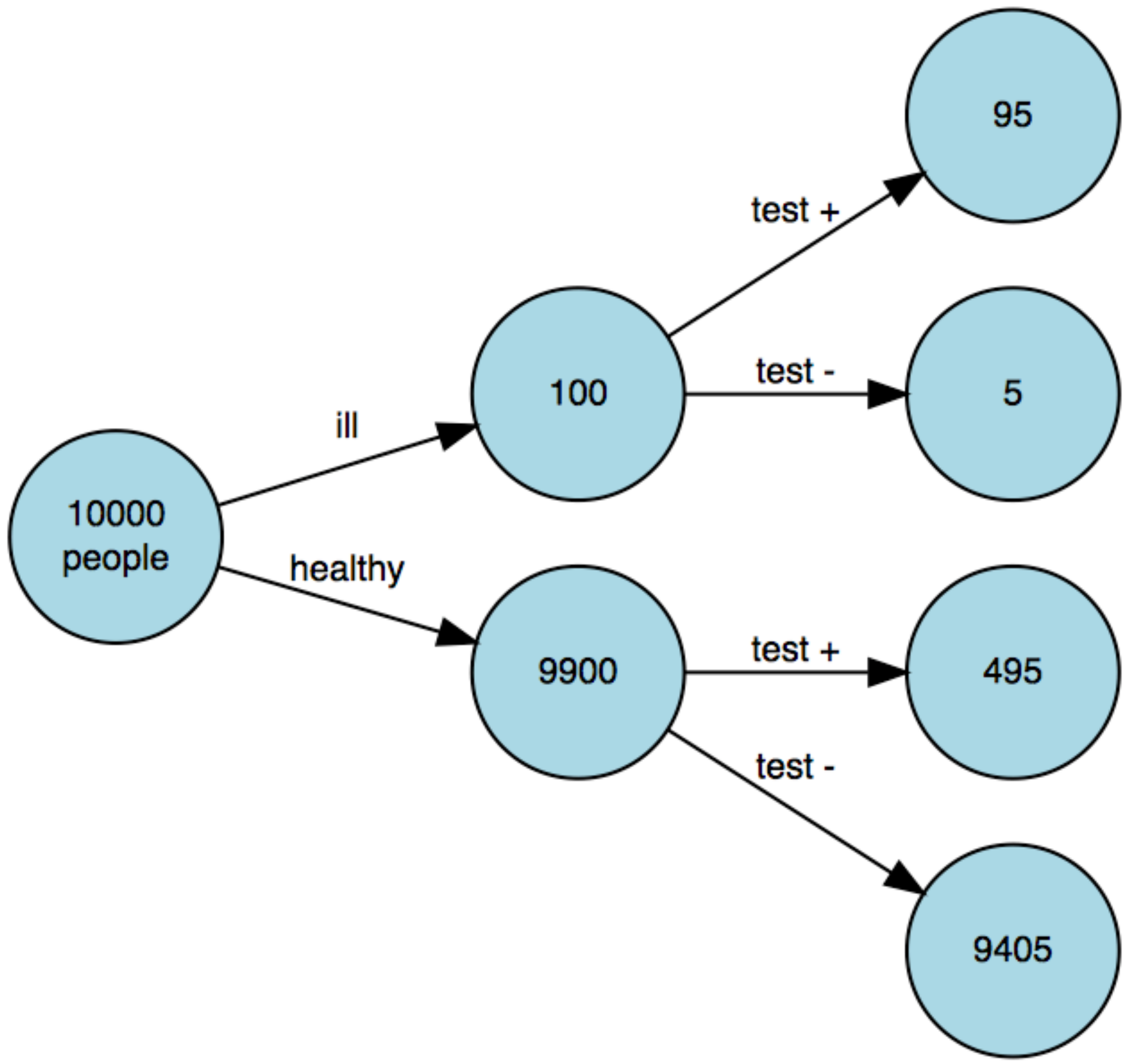
Type I Error: Falsely rejecting the null hypothesis (even though it is true).

Type II Error: Failing to reject the null hypothesis (even though it is false).

Clue guide to probability

H_0: The person is healthy.

H_1: The person is ill.



Power

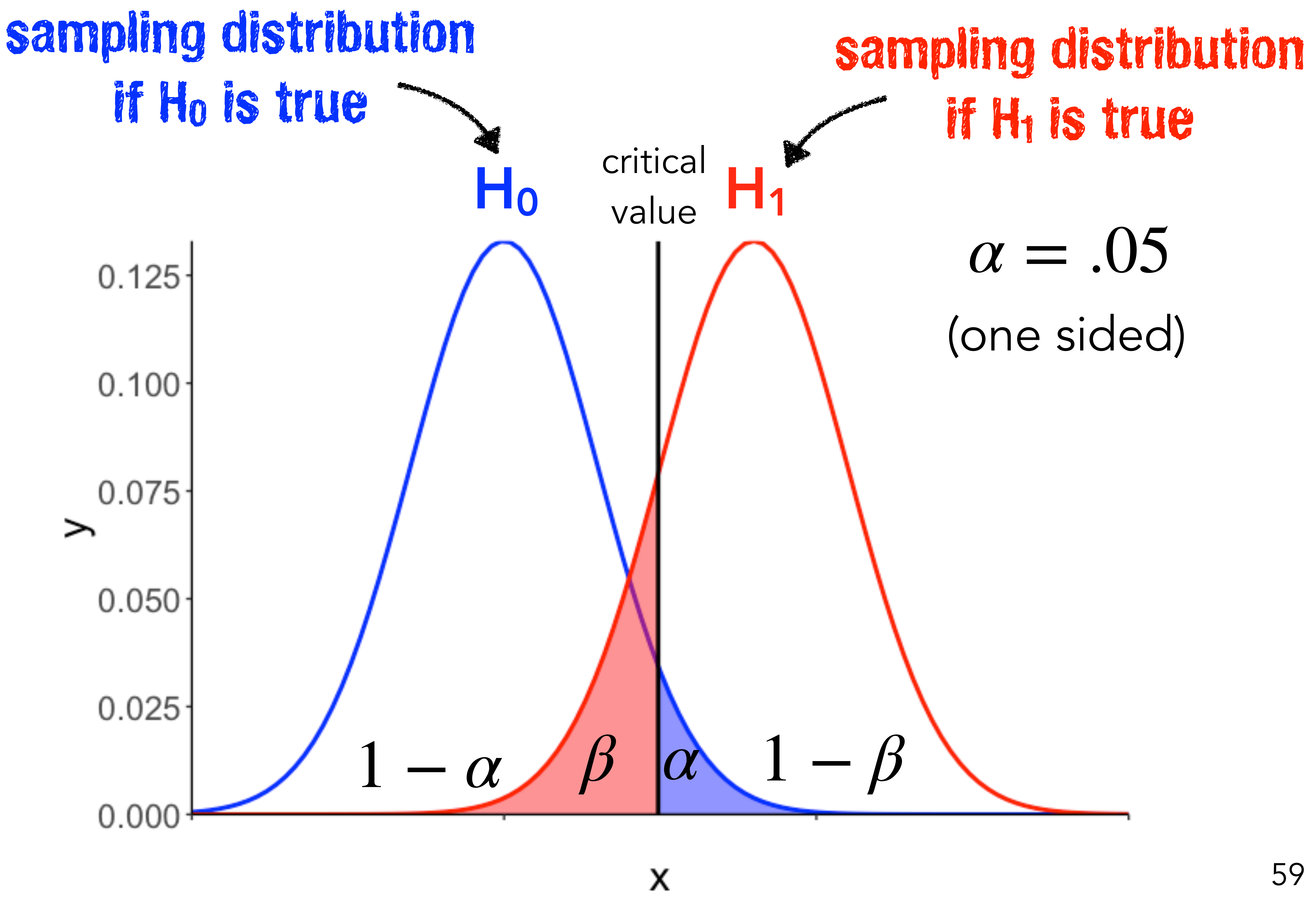
true positive — **Sensitivity** — $p(\text{reject } H_0 \,|\, H_1 \text{ is true})$ — $\boxed{1-\beta}$

false negative — **Type II error** — $p(\text{not reject } H_0 \,|\, H_1 \text{ is true})$ — β

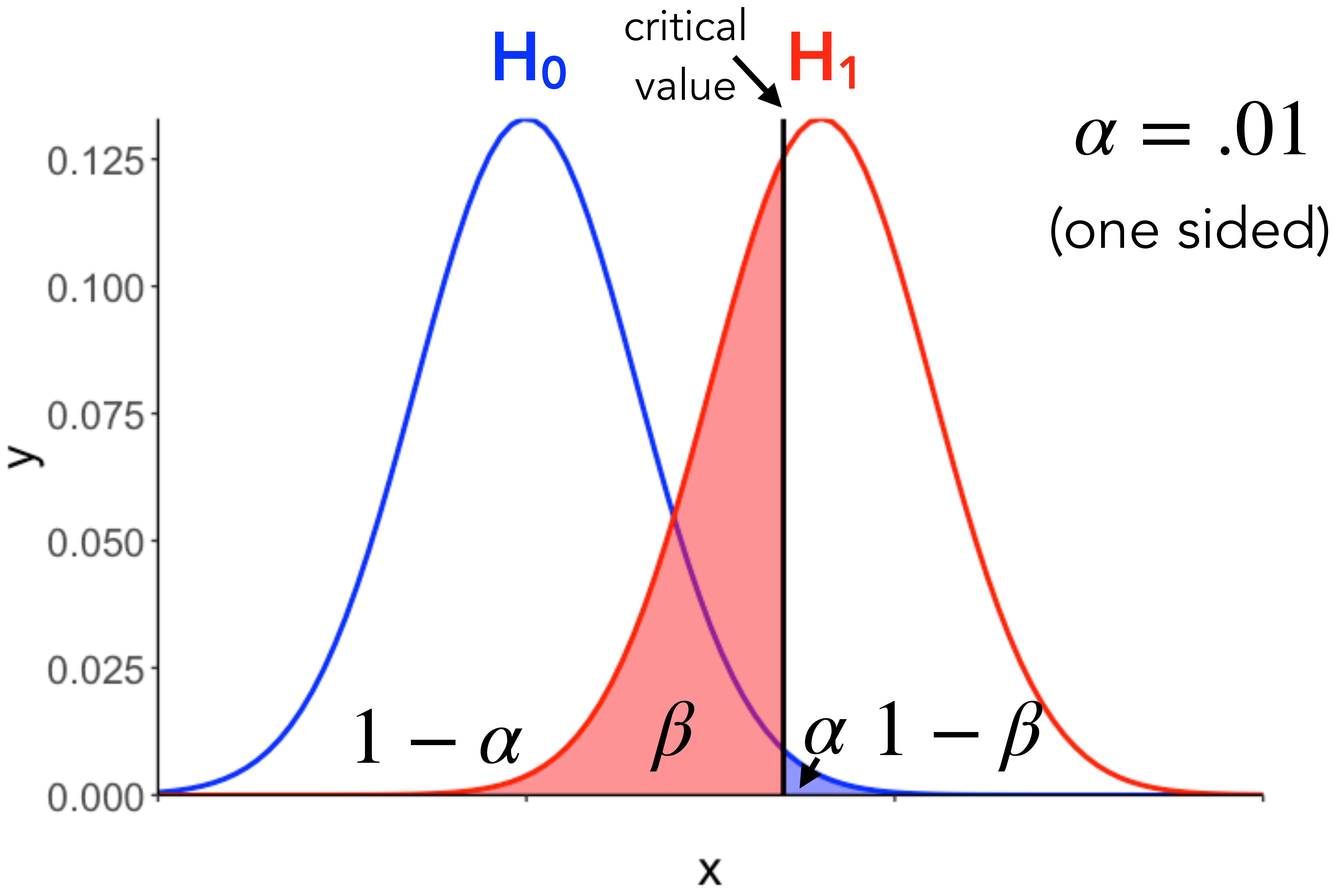
false positive — **Type I error** — $p(\text{reject } H_0 \,|\, H_0 \text{ is true})$ — α

true negative — $p(\text{not reject } H_0 \,|\, H_0 \text{ is true})$ — **Specificity** — $1-\alpha$

What affects power?

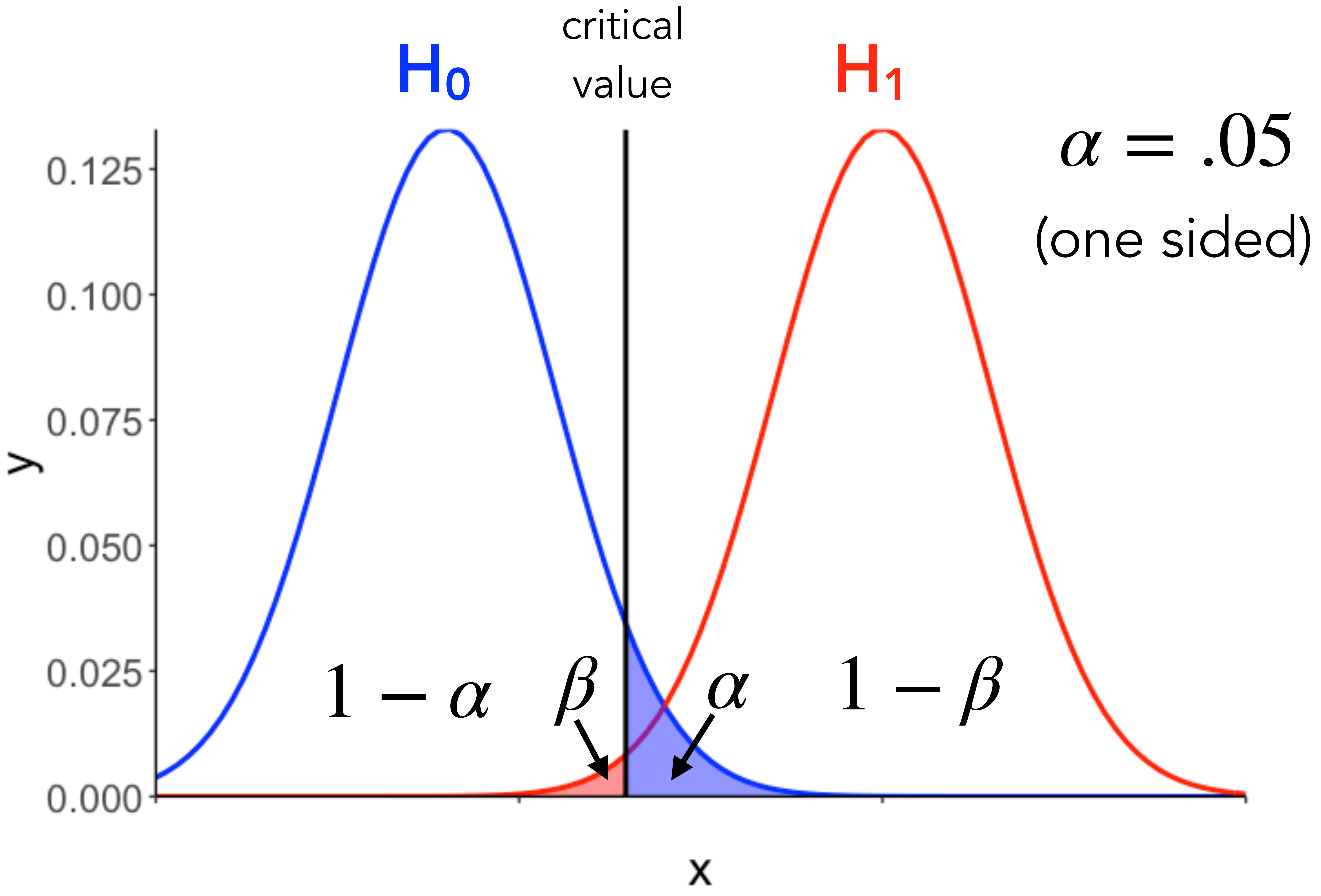


What affects power?



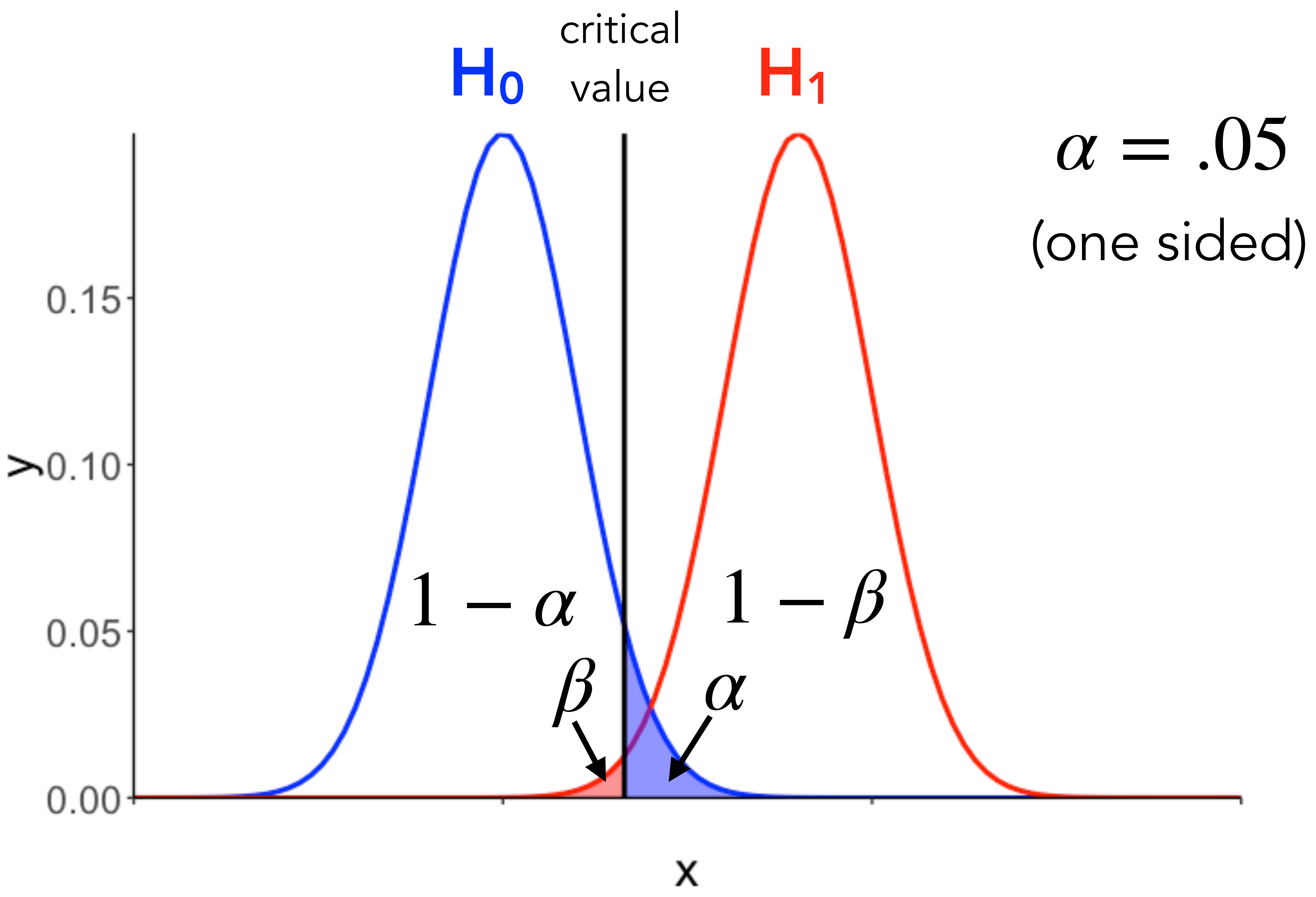
α affects power

What affects power?



distance between means affects power

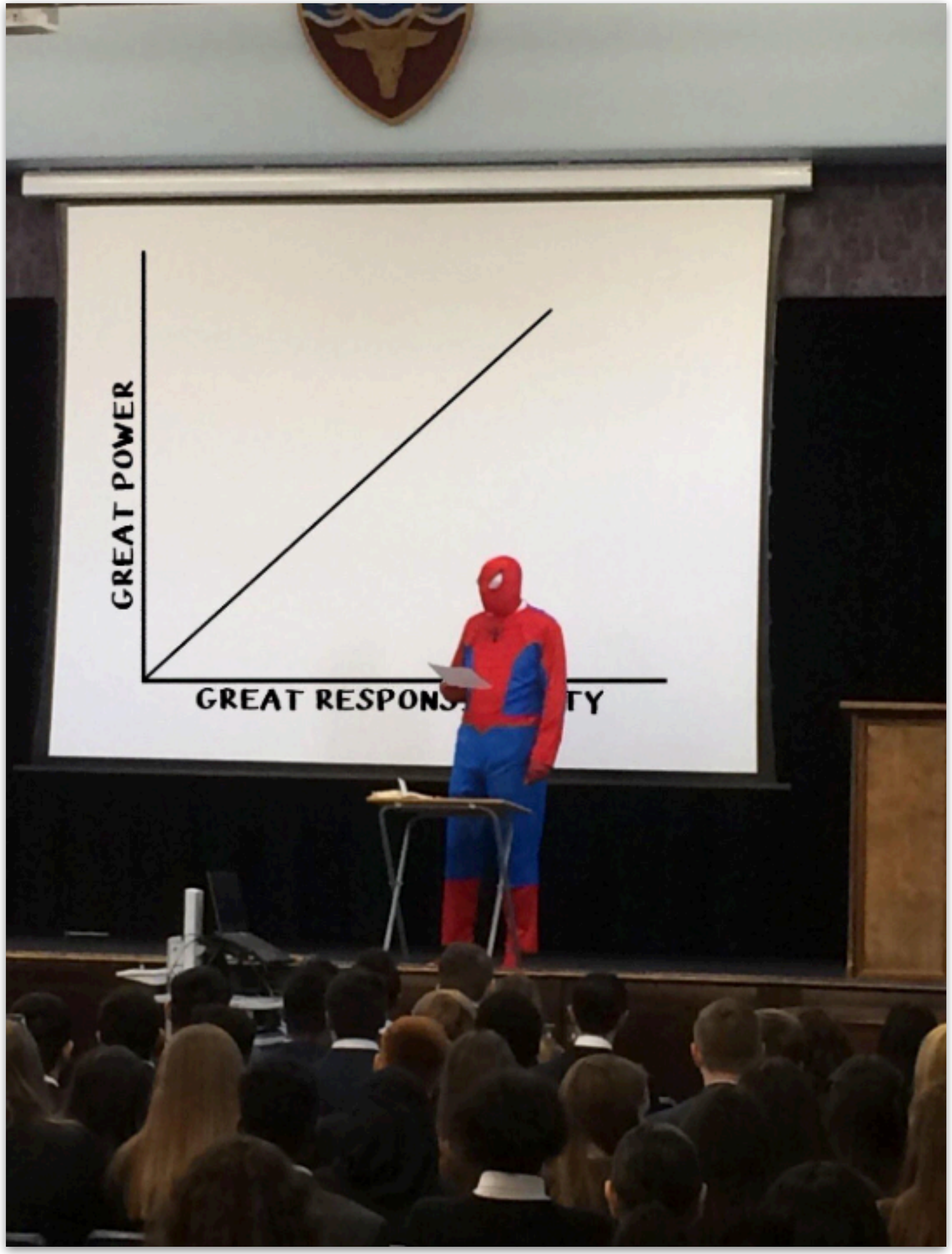
What affects power?



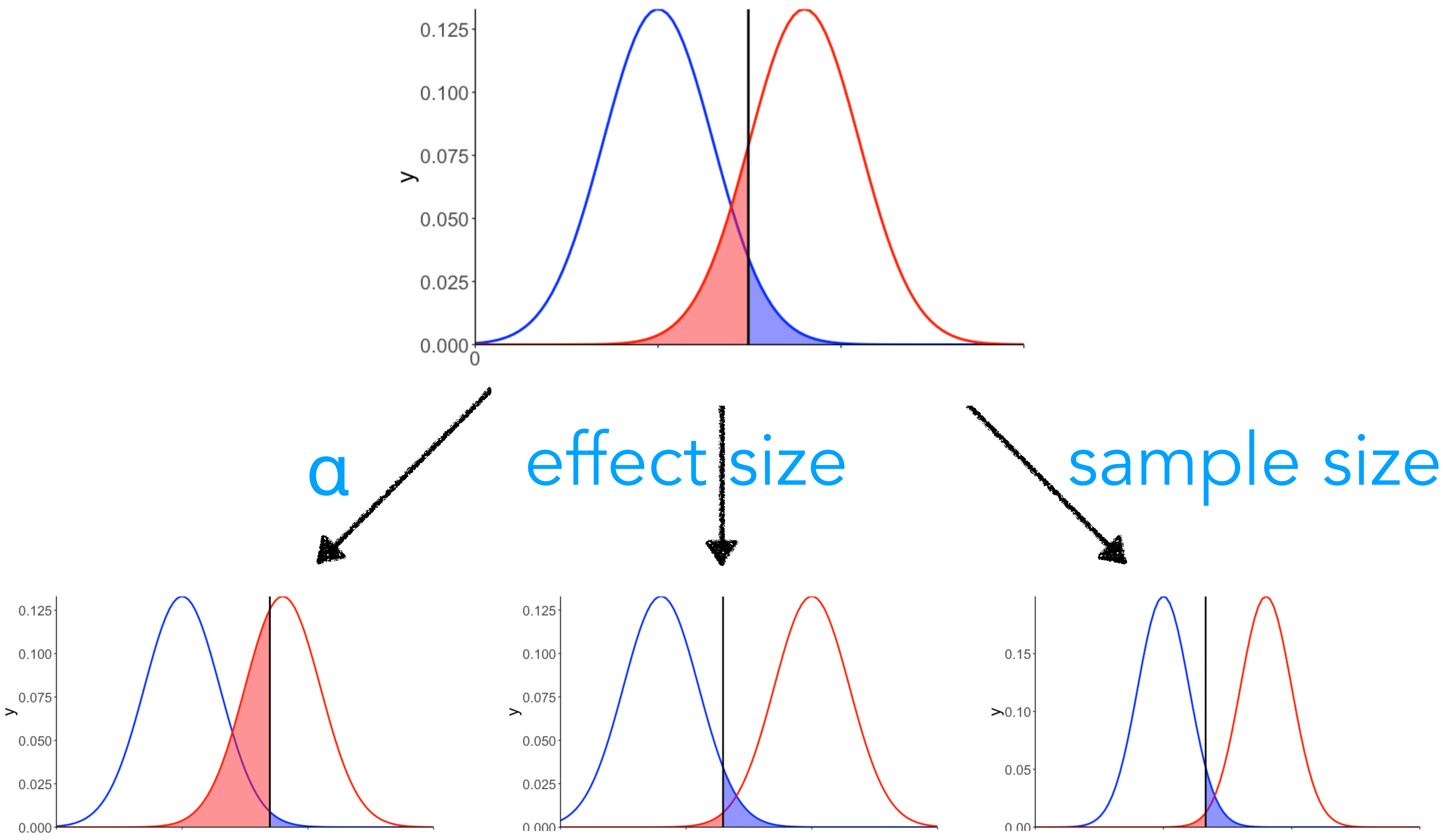
variance affects power

Calculating power

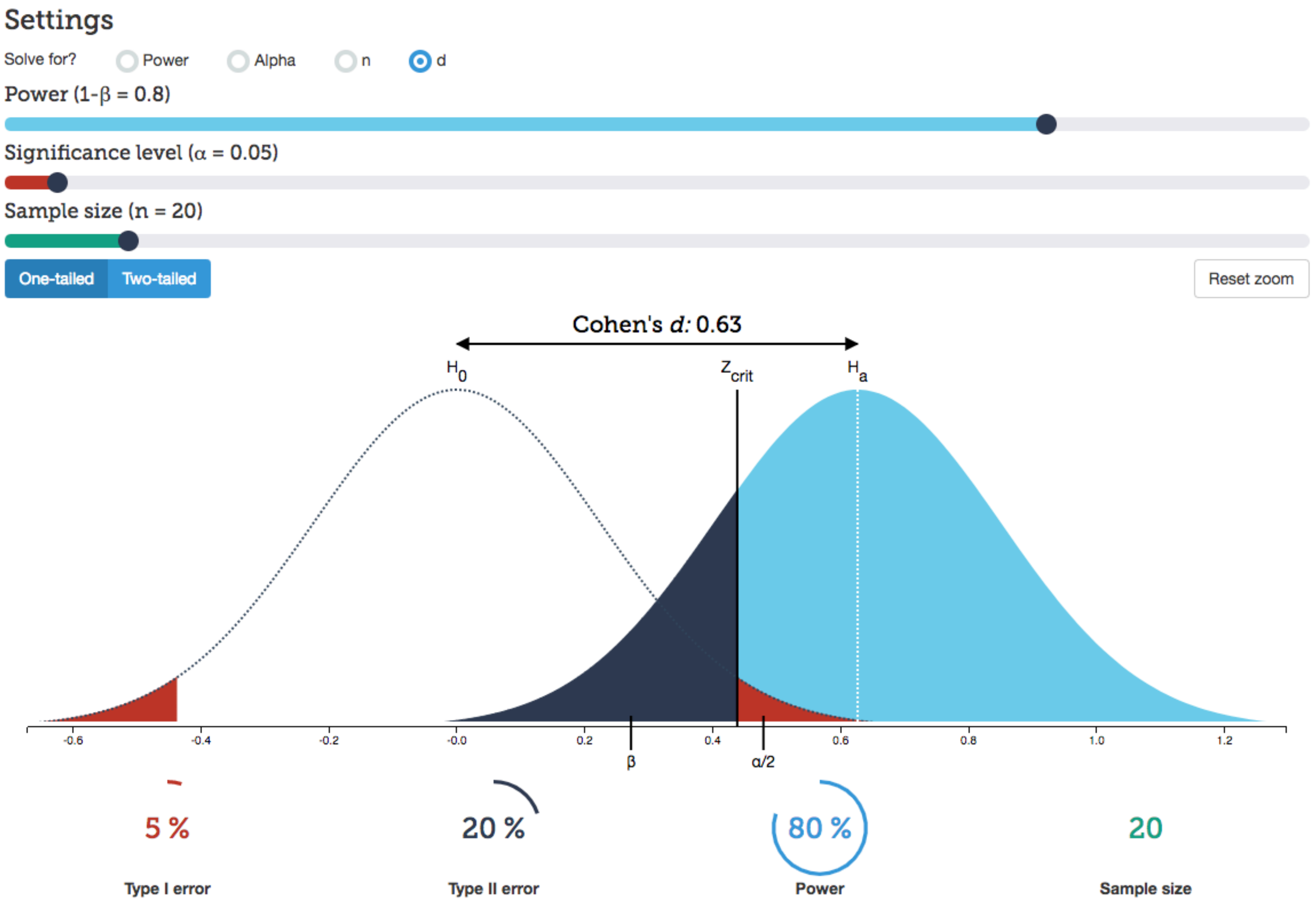
With great power comes …



The knobs we can turn to affect power



Visualization demo



https://rpsychologist.com/d3/NHST/

The **power** of a binary hypothesis test is the probability that the test rejects the null hypothesis (H_0) when a **specific** alternative hypothesis (H_1) is true.

H_0: Students and non-students have the same balance.

Model C

$$Y_i = \beta_0 + \epsilon_i$$

$$\beta_1 = 0$$

H_1: Students and non-students have different balances.

Model A

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

$$\beta_1 \neq 0$$

We cannot calculate power in this case.
We need a specific alternative hypothesis!

The **power** of a binary hypothesis test is the probability that the test rejects the null hypothesis (H_0) when a **specific** alternative hypothesis (H_1) is true.

H_0: Students and non-students have the same balance.

Model C

$$Y_i = \beta_0 + \epsilon_i$$

$$\beta_1 = 0$$

H_1: Students and non-students have different balances.

Model A

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$$

$$\beta_1 = 300$$

We can calculate power in this case (since we have a specific alternative hypothesis)!

Effect sizes

Effect sizes

- a p-value tells us whether we can reject the H_0
- effect sizes is a measure of the strength of the actual effect

Why can't we just use p-values as a measure of the effect size?

$$F = \frac{\mathrm{PRE}/(\mathrm{PA} - \mathrm{PC})}{(1 - \mathrm{PRE})/(n - \mathrm{PA})}$$

any PRE will become significant if *n* gets large enough

PRE = proportional reduction in error
PA = # parameters in the augmented model
PC = # parameters in the compact model
n = sample size

statistical vs. practical significance

Effect sizes

PRE = **p**roportional **r**eduction in **e**rror

Compact model

SSE(C)

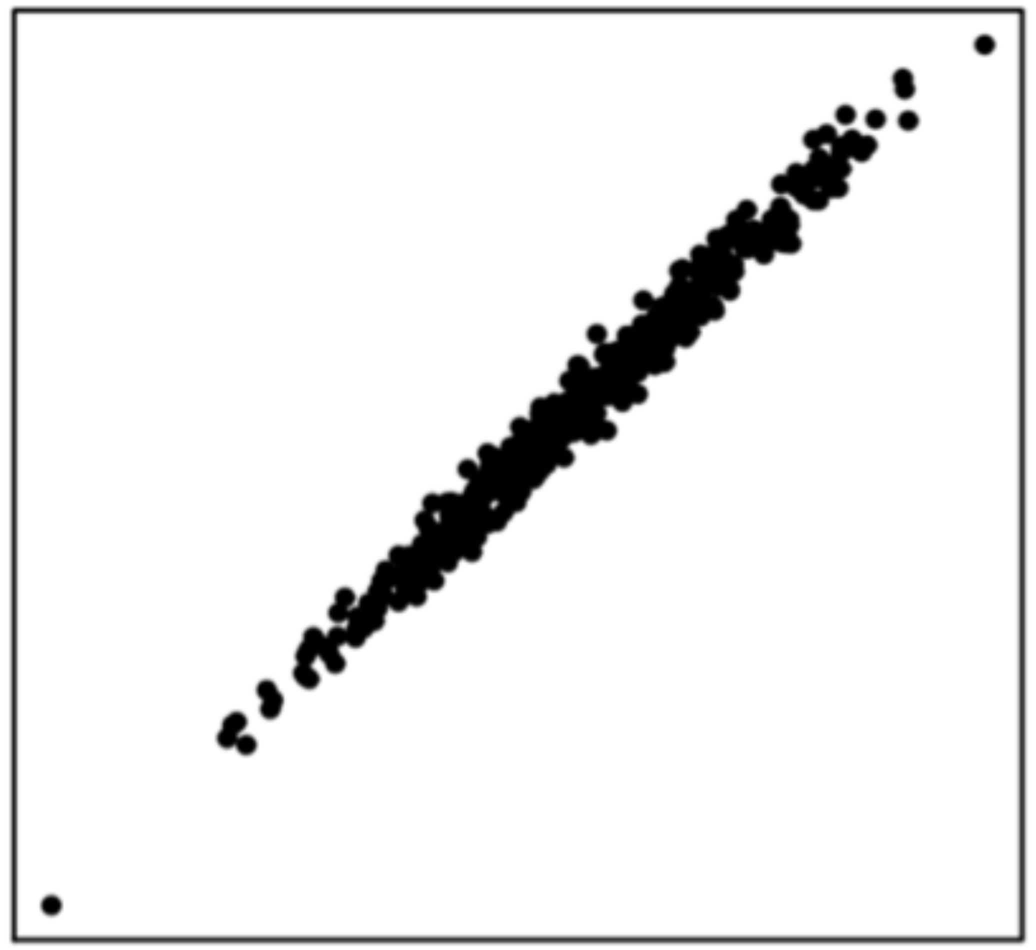
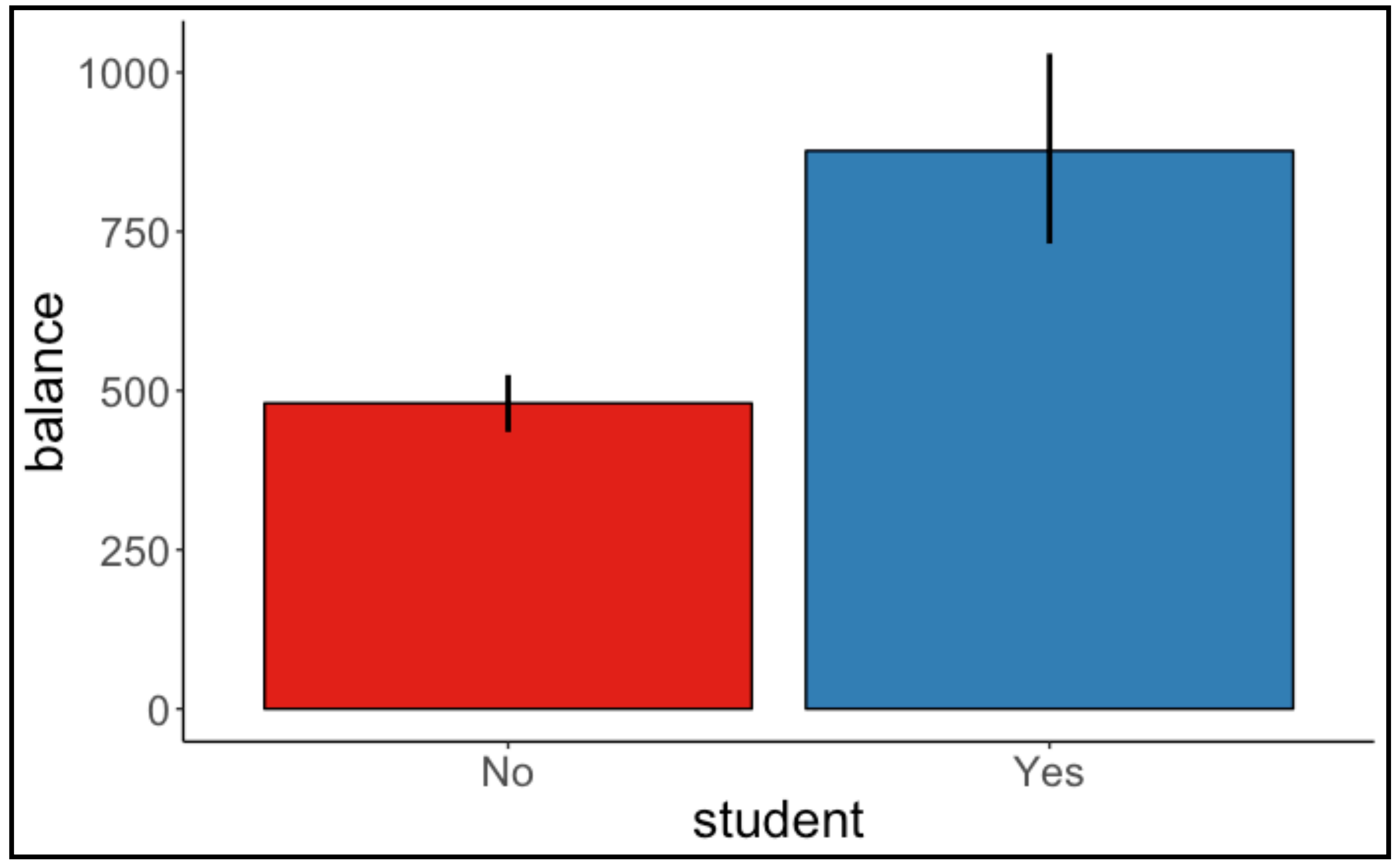
Augmented model

SSE(A)

$$\mathrm{PRE} = 1 - \frac{\mathrm{SSE}(A)}{\mathrm{SSE}(C)}$$

SSE = sum of squared errors

Common effect sizes

Relationships between variables	Differences between groups
	
r correlation	Cohen's d

Correlation

Pearson correlation

$$r(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2} \cdot \sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}$$

Cohen's guidelines for the social sciences

Effect size	*r*
Small	0.1
Medium	0.3
Large	0.5

depends very much on the domain

Cohen's *d*

- standardized difference between two means

absolute difference between means

$$d = \frac{|\bar{y}_1 - \bar{y}_2|}{s_p}$$

pooled standard variation

$$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$$

Effect size	***d***
Very small	0.01
Small	0.20
Medium	0.50
Large	0.80
Very large	1.20
Huge	2.0

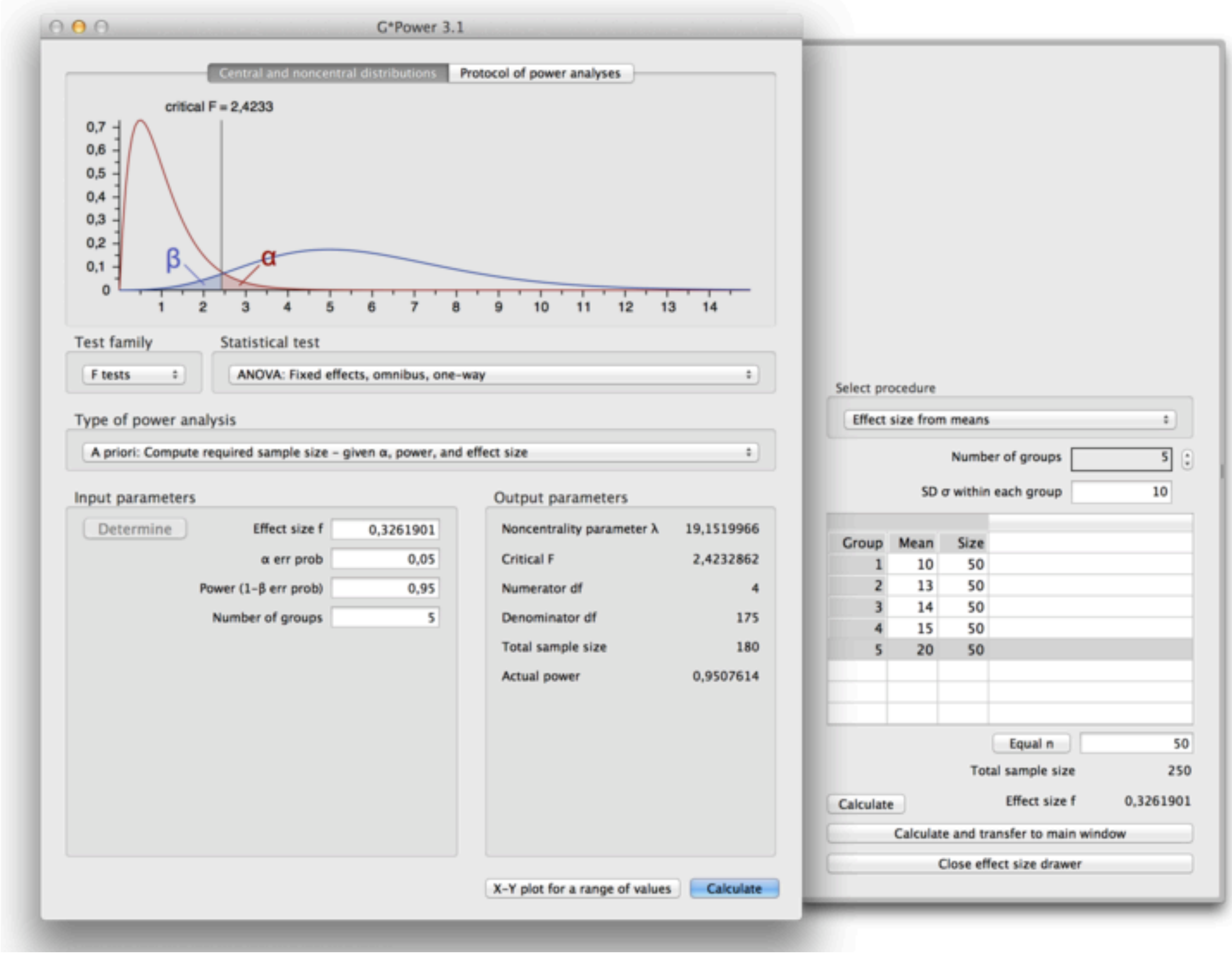
Difference between two means in pooled standard deviation

Determining sample size

How many participants do I need to run to have a good chance of detecting a true effect?

G*Power 3.1: Alternative software for power calculations

Option 1

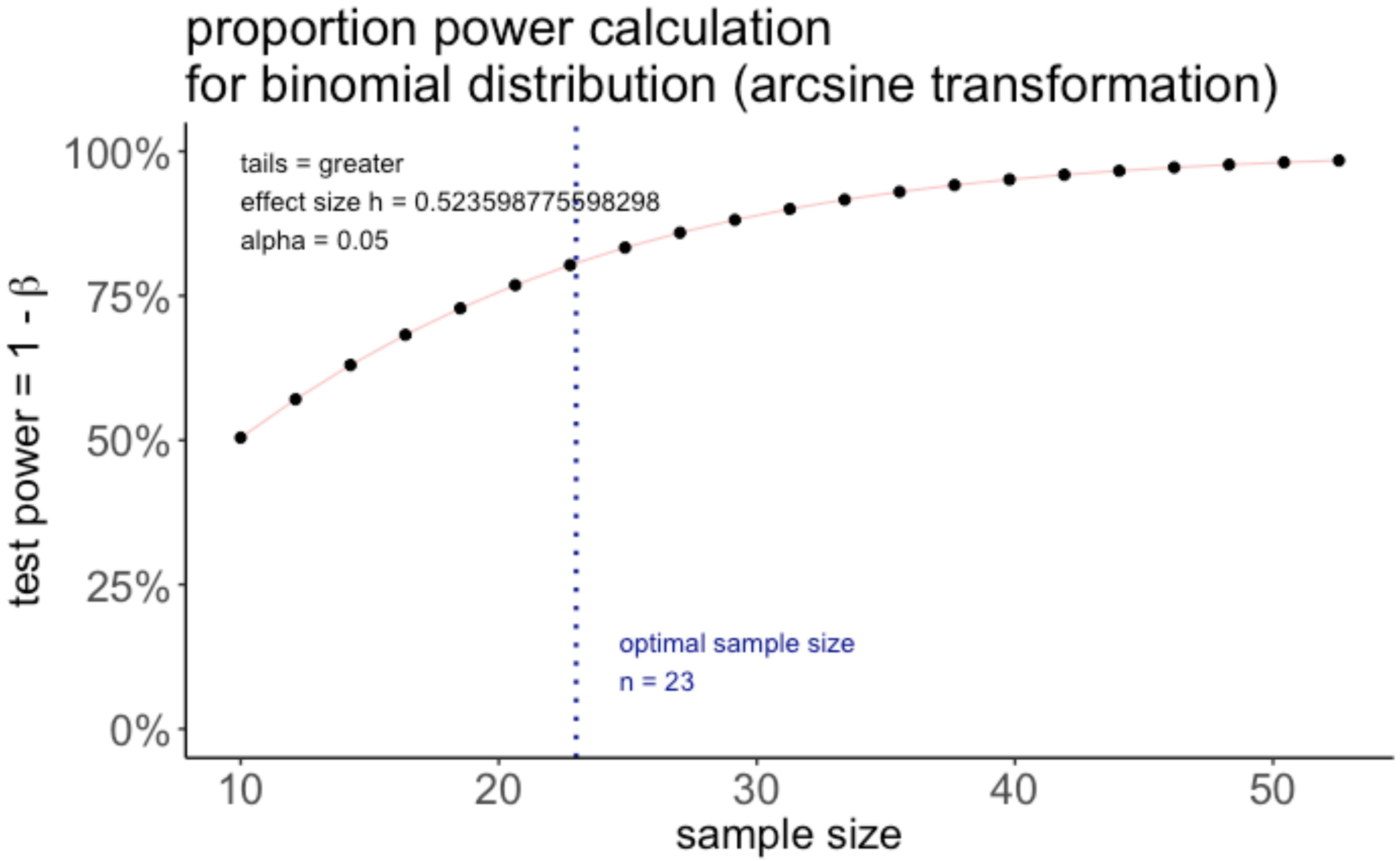


http://www.gpower.hhu.de/

"pwr" package in R

Option 2

```r
library("pwr")
pwr.p.test(h = ES.h(p1 = 0.75, p2 = 0.50),
           sig.level = 0.05,
           power = 0.80,
           alternative = "greater") %>%
  plot()
```



Option 3

Power simulation recipe

- assume:
  - α, *n*, effect size
- simulate a large number of data sets of size *n* with the specified effect size
- for each data set, run a statistical test to calculate the p-value
- determine the probability of rejecting the H_0 (given that H_1 is true)

Simulating a power analysis

Power simulation recipe

- assume:
  - α, n, effect size
- simulate a large number of data sets of size n with the specified effect size
- for each data set, run a statistical test to calculate the p-value
- determine the probability of rejecting the H_0 (given that H_1 is true)

Let's simulate ...

```r
# make reproducible
set.seed(1)

# number of simulations
n_simulations = 5

# run simulation
expand_grid(n = seq(10, 40, 2),
            simulation = 1:n_simulations,
            p = 0.75) %>%
  mutate(index = 1:n(),
         .before = n) %>%
  group_by(index, n, p, simulation) %>%
  mutate(response = rbinom(n = 1,
                           size = n,
                           prob = p),
         p.value = binom.test(x = response,
                              n = n,
                              p = 0.5,
                              alternative = "two.sided")$p.value) %>%
  group_by(n, p) %>%
  summarize(power = sum(p.value < 0.05) / n())
```

Let's simulate ...

```r
# make reproducible
set.seed(1)

# number of simulations
n_simulations = 5

# run simulation
expand_grid(n = seq(10, 40, 2),
            simulation = 1:n_simulations,
            p = 0.75) %>%
  mutate(index = 1:n(),
         .before = n) %>%
```

index	n	simulation	p
1	10	1	0.75
2	10	2	0.75
3	10	3	0.75
4	10	4	0.75
5	10	5	0.75
6	12	1	0.75
7	12	2	0.75
8	12	3	0.75
9	12	4	0.75
10	12	5	0.75

Let's simulate ...

```r
# make reproducible
set.seed(1)

# number of simulations
n_simulations = 5

# run simulation
expand_grid(n = seq(10, 40, 2),
            simulation = 1:n_simulations,
            p = 0.75) %>%
  mutate(index = 1:n(),
         .before = n) %>%
  group_by(index, n, p, simulation) %>%
  mutate(response = rbinom(n = 1,
                           size = n,
                           prob = p),
```

index	n	simulation	p	response
1	10	1	0.75	8
2	10	2	0.75	7
3	10	3	0.75	8
4	10	4	0.75	8
5	10	5	0.75	7
6	12	1	0.75	10
7	12	2	0.75	8
8	12	3	0.75	11
9	12	4	0.75	10
10	12	5	0.75	11

Let's simulate ...

index	n	simulation	p	response	p.value
1	10	1	0.75	8	0.11
2	10	2	0.75	7	0.34
3	10	3	0.75	8	0.11
4	10	4	0.75	8	0.11
5	10	5	0.75	7	0.34
6	12	1	0.75	10	0.04
7	12	2	0.75	8	0.39

```r
# make reproducible
set.seed(1)

# number of simulations
n_simulations = 5

# run simulation
expand_grid(n = seq(10, 40, 2),
            simulation = 1:n_simulations,
            p = 0.75) %>%
  mutate(index = 1:n(),
         .before = n) %>%
  group_by(index, n, p, simulation) %>%
  mutate(response = rbinom(n = 1,
                           size = n,
                           prob = p),
         p.value = binom.test(x = response,
                              n = n,
                              p = 0.5,
                              alternative = "two.sided")$p.value) %>%
```

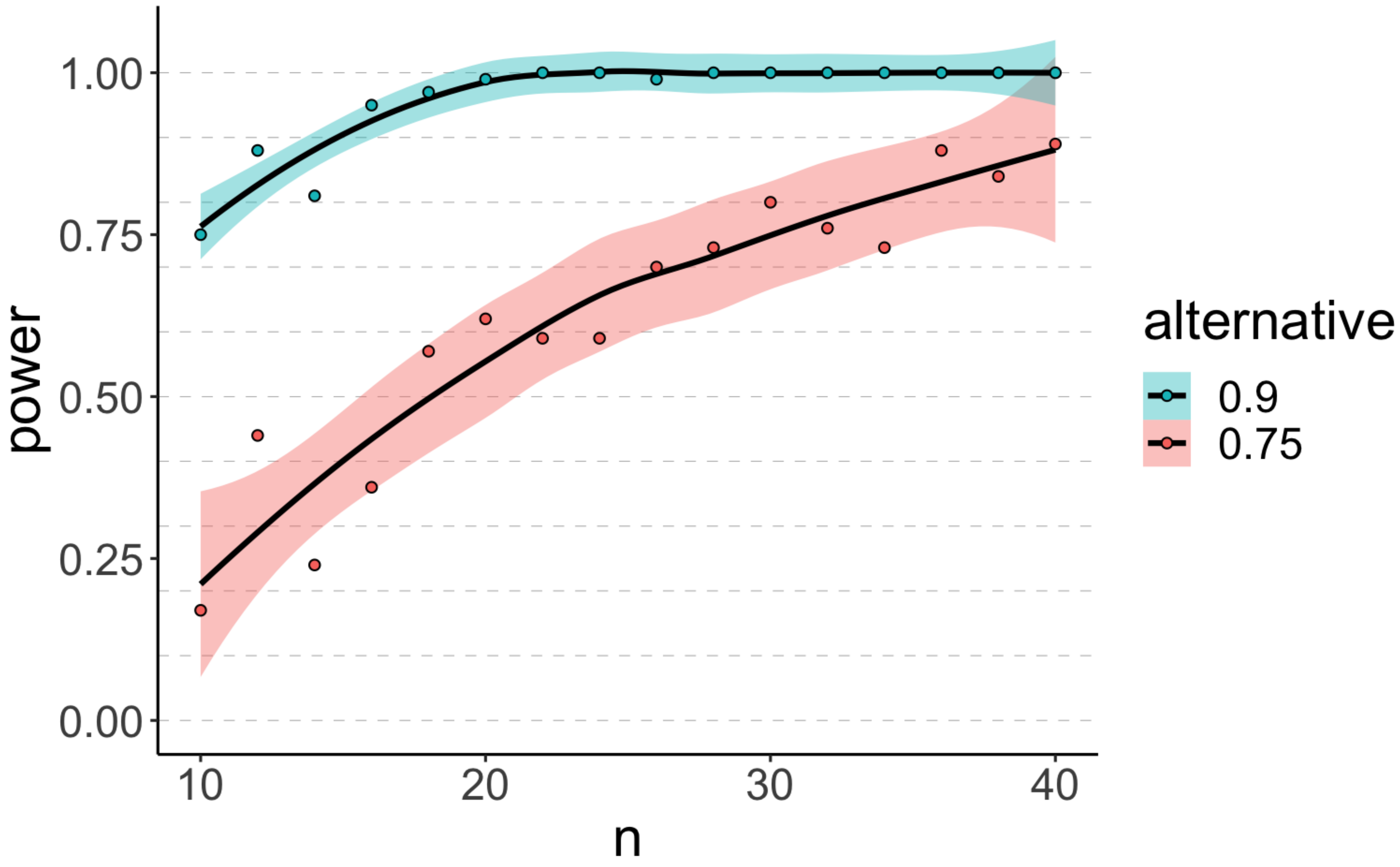
Let's simulate ...

```r
# make reproducible
set.seed(1)

# number of simulations
n_simulations = 5

# run simulation
expand_grid(n = seq(10, 40, 2),
            simulation = 1:n_simulations,
            p = 0.75) %>%
  mutate(index = 1:n(),
         .before = n) %>%
  group_by(index, n, p, simulation) %>%
  mutate(response = rbinom(n = 1,
                           size = n,
                           prob = p),
         p.value = binom.test(x = response,
                              n = n,
                              p = 0.5,
                              alternative = "two.sided")$p.value) %>%
  group_by(n, p) %>%
  summarize(power = sum(p.value < 0.05) / n())
```

n	p	power
10	0.75	0.2
12	0.75	0.2
14	0.75	0.4
16	0.75	0.2
18	0.75	0.6
20	0.75	0.8
22	0.75	0.6
24	0.75	0.4
26	0.75	0.6
28	0.75	0.8
30	0.75	0.8

Let's simulate ...

in this example, I looked at the power for two different alternative hypotheses



Let's simulate ...

- here, I've used a simple example (Binomial test)
- but: we can use the same recipe for any statistical test that we are planning on running

Power simulation recipe

- assume:
  - α, n, effect size
- simulate a large number of data sets of size n with the specified effect size
- for each data set, run a statistical test to calculate the p-value for a given α
- determine the probability of rejecting the H_0 (given that H_1 is true)

Let's simulate

```
library("purrr")
```

automatically loaded with

```
library("tidyverse")
```

`map`( )

map()

vector or list

function

```
map(.x = 1:3, .f = ~ .^2)
```

outputs a list

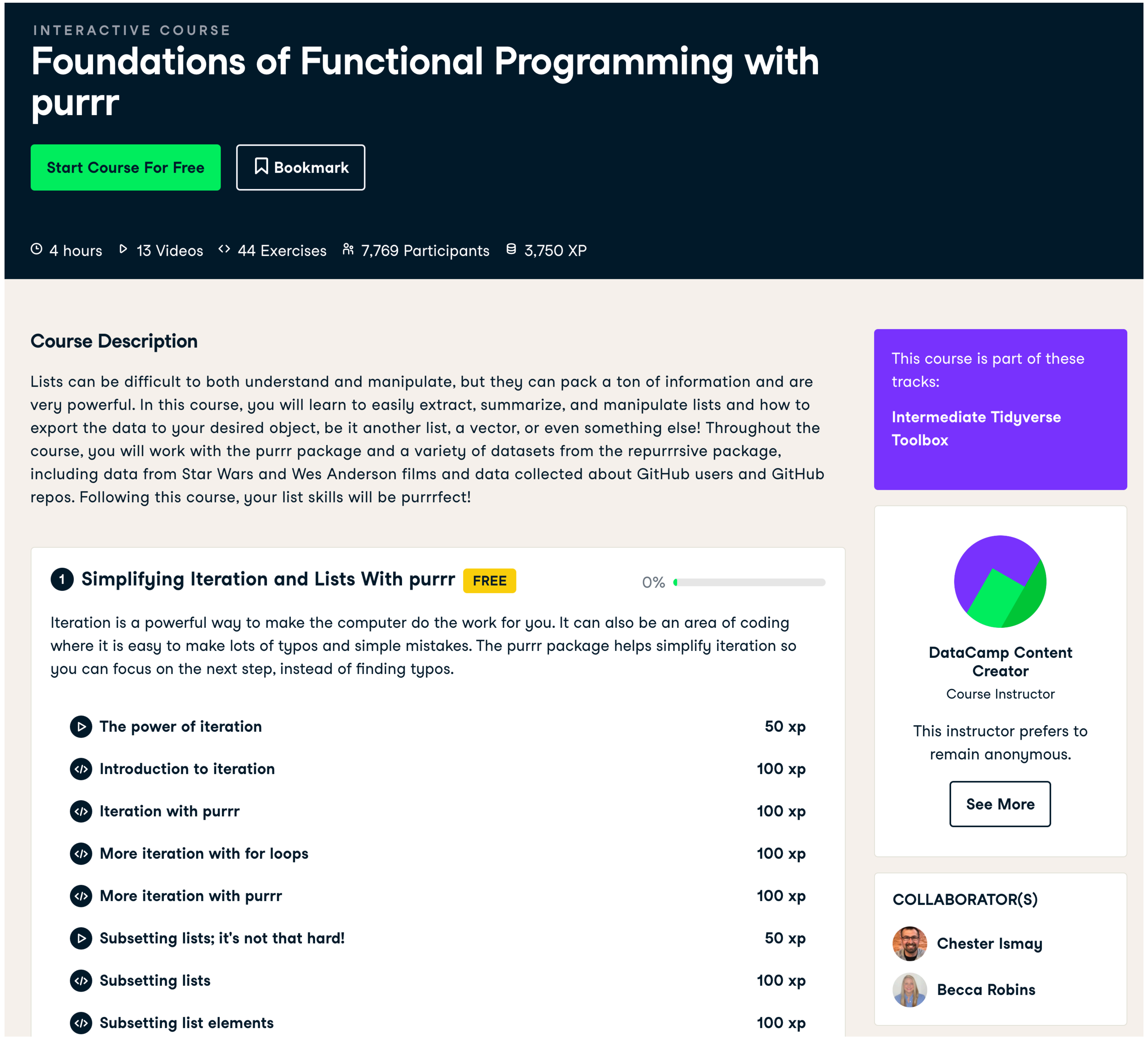
```
[[1]]
[1] 1

[[2]]
[1] 4

[[3]]
[1] 9
```

- **map**(list, function) applies a function to each element of the list
- it's a unified version of the many different **apply**() functions in base R
- you already know a cousin of **map**(): **replicate**()
- use **map**(), don't write for(){} loops!
- it's extremely powerful in combination with data frames

map()

same same but different

```
map(.x = 1:3, .f = ~ .x^2)

map(1:3, ~ .x^2)

map(1:3, ~ .^2)

map(.x = 1:3, .f = function(.x) .x^2)

# using a function
square = function(x){x^2}
map(1:3, square)
```



https://www.datacamp.com/courses/foundations-of-functional-programming-wit

Plan for today

- Quick recap
- Generalized linear model
- Power analysis
  - Making decisions
  - Calculating power
  - Effect sizes
  - Determining sample size

Feedback

How was the pace of today's class?

How happy were you with today's class overall?

What did you like about today's class? What could be improved next time?

Thank you!