# Binary variables and logistic regression

Frank Edwards

# Binary/Bernoulli data

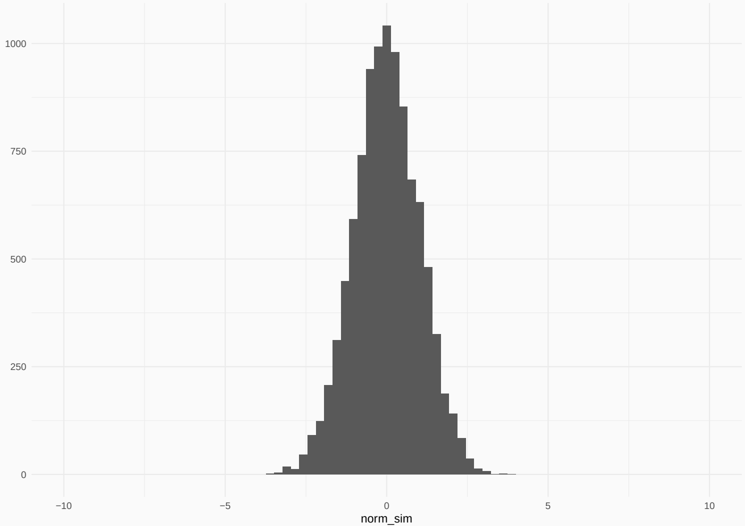
# Variables are sampled from probability distributions

Recall that a normally distributed random variable $y$ with mean $\mu$ and variance $\sigma^2$ can be expressed as:

$$y \sim Normal(\mu, \sigma^2)$$

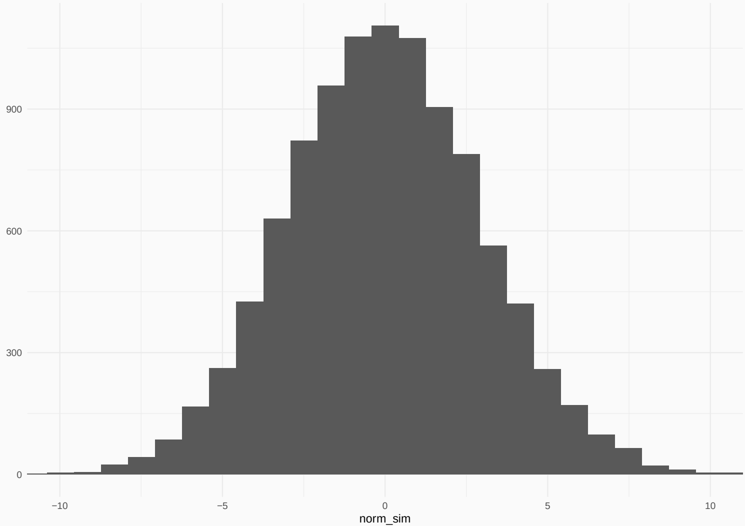
# Parameters and shape

```
norm_sim <- rnorm(10000, mean = 0, sd = 1)
qplot(norm_sim) + coord_cartesian(xlim = c(-10, 10))
```



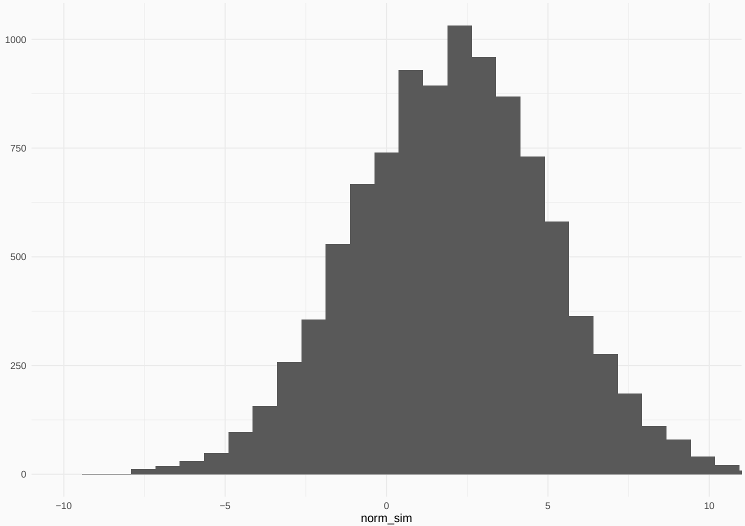
# Parameters and shape

```
norm_sim <- rnorm(10000, mean = 0, sd = 3)
qplot(norm_sim) + coord_cartesian(xlim = c(-10, 10))
```



# Parameters and shape

```
norm_sim <- rnorm(10000, mean = 2, sd = 3)
qplot(norm_sim) + coord_cartesian(xlim = c(-10, 10))
```



# All regressions model outcomes as random variables

Recall that a linear regression treats $y$ as a random variable with mean expectation such that each $y_i$ can be modeled as

$$y_i = X\beta + \varepsilon$$

or

$$y \sim Normal(X\beta, \sigma^2)$$

So each observation $y_i$ is treated as a draw from a Normal distribution with $\mu = X\beta$ and variance $\sigma^2$.

# Does one size fit all?

Does the normal model describe all phenomena we study well?

# An alternative: the Bernoulli distribution for binary data

The Bernoulli distribution for random variable $X$

$$\Pr(X = 1) = p = 1 - \Pr(X = 0)$$

Parameterization:

$$y \sim Bernoulli(p)$$

# Properties of random binary variables

If $y$ is an i.i.d. Bernoulli variable with probability $p$:

$$y \sim Bernoulli(p)$$

$$E(y) = p$$
$$Var(y) = p(1-p)$$

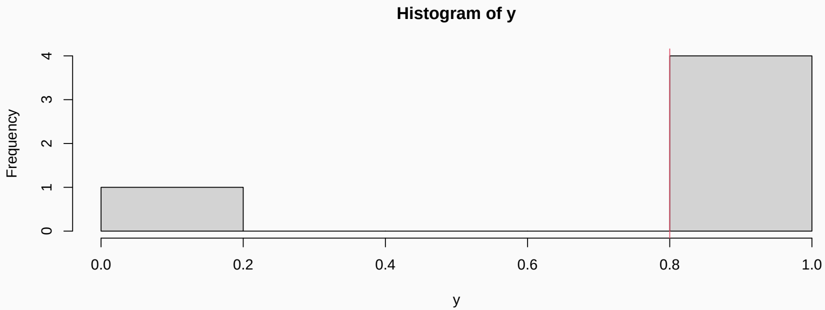
# A Bernoulli variable as a coin flip

```
flip_n_coins <- function(n) {
    rbinom(n, 1, 0.5)
}
flip_n_coins(10)
```

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

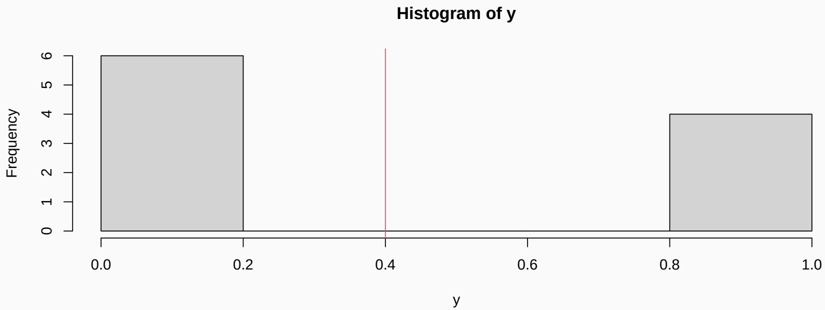
# What does the distribution of a binary variable look like?

```
y <- flip_n_coins(5)
hist(y)
abline(v = mean(y), col = 2)
```



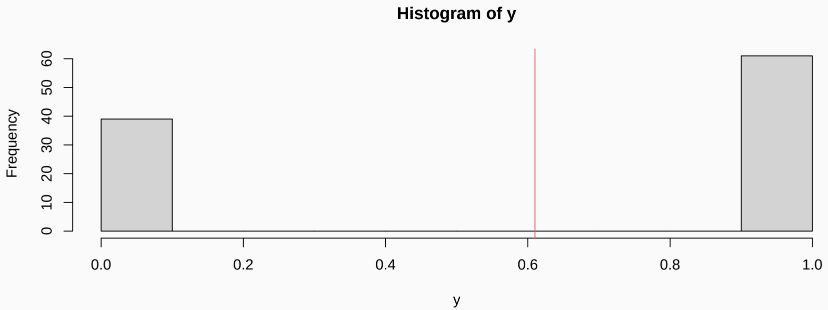
# What does the distribution of a binary variable look like?

```r
y <- flip_n_coins(10)
hist(y)
abline(v = mean(y), col = 2)
```



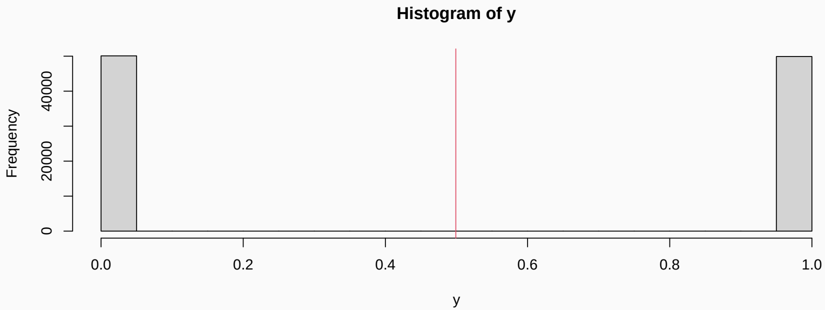
# What does the distribution of a binary variable look like?

```r
y <- flip_n_coins(100)
hist(y)
abline(v = mean(y), col = 2)
```



# What does the distribution of a binary variable look like?

```r
y <- flip_n_coins(1e+05)
hist(y)
abline(v = mean(y), col = 2)
```



A binary variable $y$ takes on the values 1 or 0, with probability

$$Pr(y = 1) = p$$

and variance

$$Var(y) = p(1 - p)$$

# Logistic regression

# Read in the data for today

```
admissions <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
head(admissions)
```

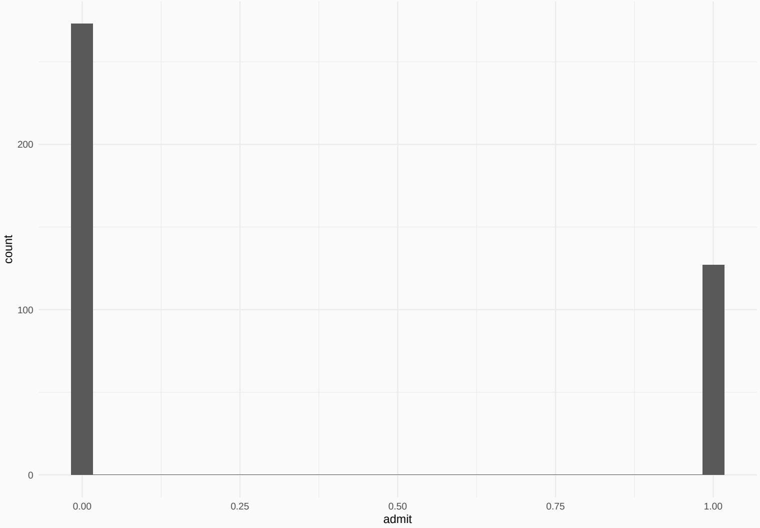
```
##   admit gre  gpa rank
## 1     0 380 3.61    3
## 2     1 660 3.67    3
## 3     1 800 4.00    1
## 4     1 640 3.19    4
## 5     0 520 2.93    4
## 6     1 760 3.00    2
```

```
nrow(admissions)
```

```
## [1] 400
```

# Evaluate distribution of binary admission variable

```
ggplot(admissions, aes(x = admit)) + geom_histogram()
```



# Properties of Bernoulli variables

If $y$ is an i.i.d. Bernoulli variable with probability $p$:

$$y \sim Bernoulli(p)$$

$$\Pr(y = 1) = p = 1 - \Pr(y = 0)$$

$$E(y) = \bar{y} = p$$

$$Var(y) = p(1 - p)$$

# Summary of admit: What can we say about the probability of admission?

```r
mean(admissions$admit)
```

```
## [1] 0.3175
```

```r
sum(admissions$admit == 1)/nrow(admissions)
```

```
## [1] 0.3175
```

```r
var(admissions$admit)
```

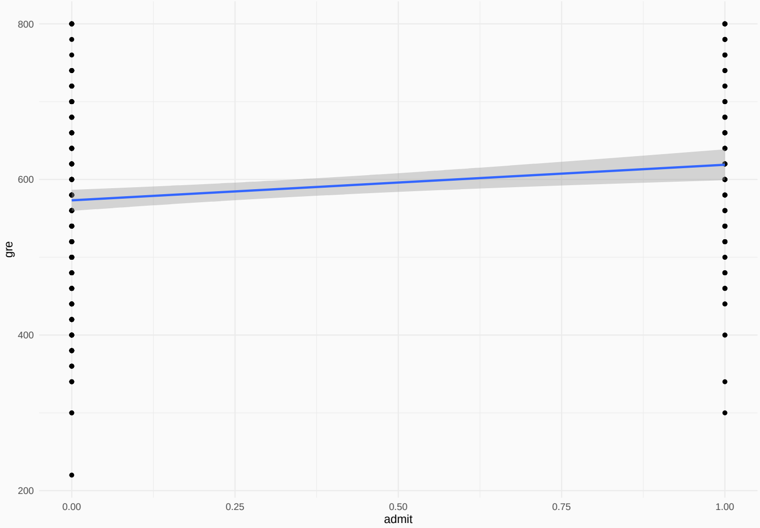
```
## [1] 0.2172368
```

```r
mean(admissions$admit) * (1 - mean(admissions$admit))
```

```
## [1] 0.2166937
```

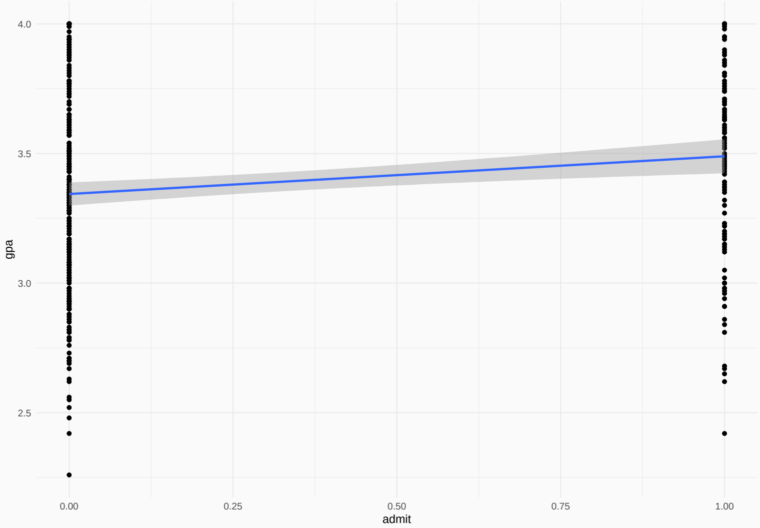
# How does GRE relate to admission?

```
ggplot(admissions, aes(x = admit, y = gre)) + geom_point() + geom_s
```



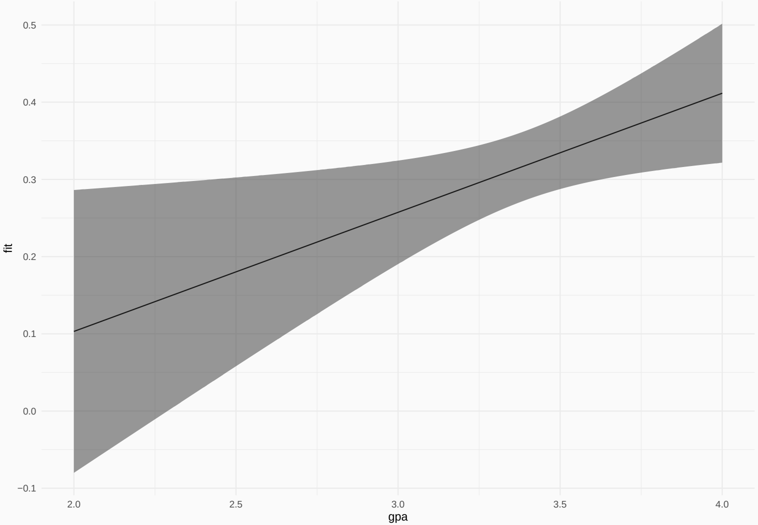
# GPA?

```
ggplot(admissions, aes(x = admit, y = gpa)) + geom_point() + geom_s
```



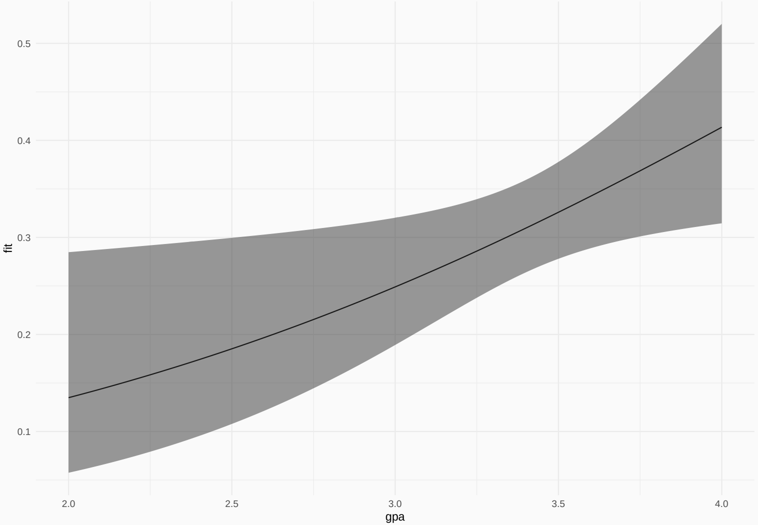
# Can we fit a model to predict admission?

```
m1 <- lm(admit ~ gre + gpa, data = admissions)
```



# Let’s try a different approach

```r
m2 <- glm(admit ~ gre + gpa, data = admissions, family = "binomial")
```



# A generalized linear model

Our linear probability model was:

$$Pr(admit = 1) = \beta_0 + \beta_1 GRE + \beta_2 GPA + \beta_3 Rank + \varepsilon$$

Our logistic regression model takes the form:

$$logit(Pr(admit = 1)) = \beta_0 + \beta_1 GRE + \beta_2 GPA + \beta_3 Rank$$

The logit function is our link between the linear predictor term $X\beta$ and the outcome *admit*.

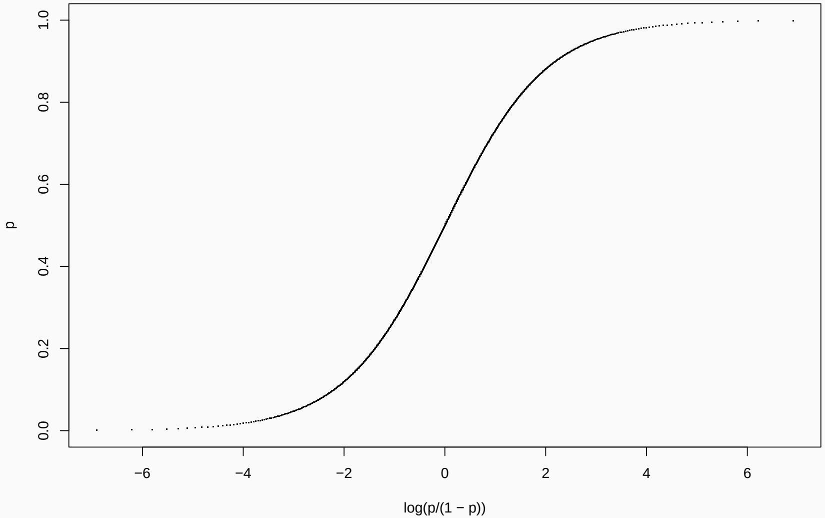
# The logit function

The logit function transforms a probability value on $[0, 1]$ to a continuous distribution

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

# The logit function

```r
p <- seq(0, 1, 0.001)
plot(log(p/(1 - p)), pch = ".", p)
```



# Logistic regression is a GLM with a logit link

A generalized linear model with link function $g$ takes the form:

$$g(y) = X\beta$$

For OLS, the link function is the identity function $g(y) = y$

For logistic regression, the link function is the logit function

$$logit(y) = X\beta$$

$$y = logit^{-1}(X\beta)$$

# Defining logit and its inverse

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

$$logit^{-1}(x) = \frac{exp(x)}{exp(x)+1}$$

We can use these functions to transform values back and forth from our logit-linear scale and the probability scale.

Uses the logit function to model the probability of a binary outcome being equal to 1. The logit function transforms the bounded interval $[0, 1]$ to a continuous distribution, allowing us to proceed with building a regression model as we ordinarily would.

Logistic regression may have more accurate uncertainty estimates than a linear probability model for binary outcomes. Logistic regression also constrains model predictions to $[0, 1]$.

# Running logistic models in R: the glm() function

```r
m1 <- glm(admit ~ gpa, data = admissions, family = "binomial")
tidy(m1)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    -4.36     1.04      -4.21 0.0000257
## 2 gpa             1.05     0.299      3.52 0.000437
```

How do we interpret the coefficients?

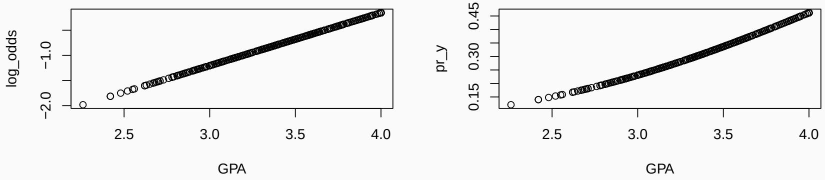
- Log odds: $\beta_1$
- Odds ratio: $e^{\beta_1}$
- Probability: $logit^{-1}(x) = \frac{exp(X\beta)}{exp(X\beta)+1}$

I tend to prefer transforming to a probability scale, as log odds and odds ratios are a bit confusing to define and are not especially intuitive.

# To get predicted probabilities from m1

We need $X\beta$, then apply the logit inverse function

```
x <- cbind(rep(1, nrow(admissions)), admissions$gpa)
log_odds <- coef(m1) %*% t(x)
pr_y <- exp(log_odds)/(exp(log_odds) + 1)
par(mfrow = c(1, 2))
plot(x[, 2], log_odds, xlab = "GPA")
plot(x[, 2], pr_y, xlab = "GPA")
```



# Alternatively

```
log_odds <- predict(m1)
pr_y <- predict(m1, type = "response")
```