Universidade Federal do Rio Grande do Sul (UFRGS)

Faculdade de Medicina - Estatística II - Notas de Aula EPI0110/2022 - Doutorado

Prof. Dr. Lucas Helal

Sumário

Modelos Binomiais, Modelos de Regressão com Distribuição Binomial e Aplicações em Epidemiologia

As distribuições binomiais são fundamentais na Estatística aplicada à Epidemiologia, pois permitem modelar a ocorrência de eventos classificados como **sucesso ou falha** - por exemplo, na comparação de testes diagnósticos; é também poderoso pela robustez algébrica do estimando: o **logito**.

A robustez de construto do logito se dá por sua característica de ser uma função monótona crescente; pela operação com bases logarítimicas, o que otimiza a capacidade computacional em grande magnitude; e por conseguir alcançar o conceito de odds, que é uma medida de probabilidade (incerteza) do *sucesso* em relação ao *fracasso*.

Contra a noção de probabilidade bruta, falar de *chances* é como pensar nas probabilidades de um evento ocorrer em relação a ele não ocorrer - e que essas duas probabilidades não necessariamente são determinadas pelos mesmos fatores. Por outro lado, quantitativamente, comparar chances e probabilidades é uma tarefa mais complexa, pois as chances são tal qual as probabilidade uma medida de incerteza; porém, a probabilidade

traduz a incerteza como os potenciais eventos dentro de todos os eventos possíveis (incluindo o evento de interesse), enquanto as chances colocam a incerteza como uma divisão aritmética entre duas probabilidades.

Assumir que probabilidades e chances, em suas escalas, traduzem a mesma magnitude de incerteza é um erro **que se deve evitar a todo custo**, porque a diferença entre os estimadores pode ser *marginal* ou *muito significativa*, por fatores que já vimos em aula.

Desfechos Binomiais

Os desfechos binomiais são aqueles que só podem ter em seu espaço amostral dois resultados independentes e mutuamente excludentes: **sucesso** ou **fracasso**.

Por sucessos e fracassos entende-se que o sucesso é o evento de interesse, e o fracasso é o evento que não é de interesse. Por exemplo:

- *Evento a ser mensurado*: resultado de um teste diagnóstico para COVID-19.
- *Resultados possíveis*: a) o teste é **positivo** (sucesso); b) o teste é **negativo** (fracasso).

Veja que não há nenhuma outra possibilidade para esse teste, a não ser ser positivo ou negativo. Já uma sorologia para níveis de hemoglobina glicada (HbA1c) em um paciente para diagnóstico de diabetes mellitus tipo 2, por exemplo, por natureza não é binomial: o espectro de valores possíveis é contínuo, e não discreto, virtualmente variando de **0%** a **100%** de HbA1c como resultado possível. Importante não confundir a convenção de se *discretizar* uma variável contínua para fins de diagnóstico, como no caso do T2DM. Assume-se que se a HbA1c for maior que **6,5%**, o paciente tem T2DM, e, portanto, é um **sucesso**. Arbitra-se um valor de corte, mas não implica dizer que a distribuição de HbA1c é binomial no ser humano.

O que é uma distribuição binomial?

A distribuição binomial é uma distribuição de probabilidade discreta que descreve o número de sucessos em uma sequência de tentativas de Bernoulli, ou seja, tentativas que são independentes e têm uma probabilidade constante de sucesso.

Para relembrar:

A **distribuição de Bernoulli** é uma distribuição de probabilidade discreta que modela um experimento com exatamente dois resultados possíveis: **sucesso** e **fracasso**. Tem um único parâmetro, **p**, que é a probabilidade de sucesso do evento - ou seja, é a esperança matemática da distribuição. Costumeiramente, o sucesso é representado por **1** e o fracasso por **0**, e a utilizamos para descrever situações como:

```
- A probabilidade de um paciente _ter_ ou _não ter_ diabetes mellitus tipo 2.
- A probabilidade de um teste diagnóstico ser _positivo_ ou _negativo_.
- A probabilidade de um paciente _sobreviver_ ou _não sobreviver_ a um procedimento cirúrgico.
- A probabilidade de um paciente _ter_ ou _não ter_ uma complicação em pós-operatório.
```

A função densidade de probabilidade da distribuição de Bernoulli é dada por:

$$f(x) = p^x(1-p)^{1-x}$$

Onde:

- $x = 0, 1$
- p é a probabilidade de sucesso.
- $1 - p$ é a probabilidade de fracasso.
- $f(x)$ é a função densidade de probabilidade.

O estimador da esperança matemática da distribuição de Bernoulli é dada por:

$$E(X) = p$$

onde $E(x)$ é calculado por:

$$E(X) = \sum_{x=0}^{1} x \cdot f(x)$$

isto é, a soma de todos os valores possíveis de x multiplicados pela função densidade de probabilidade - ou, de forma simplificada, a probabilidade de sucesso multiplicada pelo valor de sucesso, mais a probabilidade de fracasso multiplicada pelo valor de fracasso, onde *valor de sucesso e fracasso* são, respectivamente, **1** e **0**.

A variância da distribuição de Bernoulli é dada por:

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

onde $Var(X)$ é calculado por:

$$Var(X) = \sum_{x=0}^{1} (x - E(X))^2 \cdot f(x)$$

isto é, a soma de todos os valores possíveis de x menos a esperança matemática, ao quadrado, multiplicado pela função densidade de probabilidade.

Visualização gráfica da distribuição de Bernoulli dadas diferentes probabilidades de sucesso para k tentativas:

O evento de Bernoulli é um evento de uma única tentativa, como jogar uma moeda para cima uma única vez. Já a distribuição de Bernoulli é o conjunto de probabilidades de sucesso e fracassos futuros, dados os resultados dos k eventos de Bernoulli realizado.

EXEMPLO PRÁTICO:

Imagine que você joga uma moeda para cima uma única vez, não viciada. A probabilidade de sucesso (cara) P(cara) é de 0,5 - ou, cotidianamente, 50%. Entretanto, se você lançar a mesma moeda para cima 10 vezes, para uma décima primeira tentativa, a probabilidade de sucesso não será mais necessariamente 0,5.

Seja o vetor A o vetor de número de lançamentos da moeda (Ω) e o vetor B o vetor correspondente ao resultado da tentativa:

```
A <- seq(1, 10, 1)
B <- rbinom(10, 1, 0.5)

tabelaResultados <- data.frame(A, B)
tabelaResultados
```

```
##     A B
## 1   1 0
## 2   2 0
## 3   3 1
## 4   4 0
## 5   5 0
## 6   6 1
## 7   7 0
## 8   8 1
## 9   9 0
## 10 10 0
```

Calculando a frequência absoluta e relativa de sucessos e fracassos, temos que:

```
library(tidyverse)
tabelaResultados |>
  group_by(B) |>
  reframe(n = n(), freq = n()/sum(n))
```

```
## # A tibble: 2 x 3
##       B     n  freq
##   <int> <int> <dbl>
## 1     0     7     1
## 2     1     3     1
```

Ou seja, antes do experimento ser realizado - lançar uma moeda não viciada 10 vezes para cima - a esperança de sucesso é de 50%, mas não foi efetivamente isso que ocorreu. A frequência de sucessos foi de 60%, e a frequência de fracassos foi de 40%. Em frequência cumulativa, a probabilidade se distribui da seguinte maneira. Guardamos agora a frequência cumulativa de sucessos e fracassos para uma probabilidade antecipada de 0,5 em uma tabela, e simularemos o experimento mais 5 vezes com probabilidades a priori diferentes, de 0.1, 0.3, 0.7 e 0.9.

```
probabilidadesBernoulli <- c(0.1, 0.3, 0.5, 0.7, 0.9)
tabelaResultados <- data.frame(A, B)
```

> Dica de código: para aplicar as diferentes probabilidades, podemos repetir manualmente o código já executado trocando o parâmetro de probabilidade e dando um novo nome para o objeto gerado; ou, podemos criar uma **função** que itere sobre as probabilidades e aplique o código de forma automática.

A seguir as duas maneiras:

1. Repetindo o código manualmente:

```
# Probabilidade 0.1
B_1 <- rbinom(10, 1, 0.1)
B_3 <- rbinom(10, 1, 0.3)
B_5 <- rbinom(10, 1, 0.5)
B_7 <- rbinom(10, 1, 0.7)
B_9 <- rbinom(10, 1, 0.9)

tabelaResultados_1 <- data.frame(A, B_1)
tabelaResultados_3 <- data.frame(A, B_3)
tabelaResultados_5 <- data.frame(A, B_5)
tabelaResultados_7 <- data.frame(A, B_7)
tabelaResultados_9 <- data.frame(A, B_9)

tabelaResultados_1
```

```
##     A B_1
## 1   1   0
## 2   2   0
## 3   3   0
## 4   4   0
## 5   5   1
## 6   6   0
## 7   7   1
## 8   8   0
## 9   9   0
## 10 10   0
```

```
tabelaResultados_3
```

```
##     A B_3
## 1   1   0
## 2   2   1
## 3   3   0
## 4   4   0
## 5   5   0
## 6   6   1
## 7   7   1
## 8   8   1
## 9   9   1
## 10 10   1
```

```
tabelaResultados_5
```

```
##     A B_5
## 1   1   1
## 2   2   0
## 3   3   1
## 4   4   0
## 5   5   1
## 6   6   1
## 7   7   0
## 8   8   1
## 9   9   0
## 10 10   0
```

```
tabelaResultados_7
```

```
##     A B_7
## 1   1   0
## 2   2   1
## 3   3   1
## 4   4   1
## 5   5   0
## 6   6   0
## 7   7   0
## 8   8   1
## 9   9   1
## 10 10   1
```

```
tabelaResultados_9
```

```
##     A B_9
## 1   1   1
## 2   2   0
## 3   3   1
## 4   4   1
## 5   5   1
## 6   6   1
## 7   7   1
## 8   8   1
## 9   9   1
## 10 10   1
```

2. Criando uma função para iterar sobre as probabilidades:

```r
# Função para simular experimentos de Bernoulli
simulaBernoulli <- function(probabilidades) {
  for (i in probabilidades) {
    B <- rbinom(10, 1, i)
    tabelaResultados <- data.frame(A, B)
    print(tabelaResultados)
  }
}

simulaBernoulli(probabilidadesBernoulli)
```

```
##     A B
## 1   1 0
## 2   2 0
## 3   3 0
## 4   4 0
## 5   5 0
## 6   6 0
## 7   7 0
## 8   8 0
## 9   9 1
## 10 10 0
##     A B
## 1   1 0
## 2   2 1
## 3   3 0
## 4   4 0
## 5   5 0
## 6   6 0
## 7   7 1
## 8   8 1
## 9   9 0
## 10 10 0
##     A B
## 1   1 0
## 2   2 0
## 3   3 0
## 4   4 0
## 5   5 1
## 6   6 1
## 7   7 0
## 8   8 0
## 9   9 1
## 10 10 0
##     A B
## 1   1 1
## 2   2 0
## 3   3 1
## 4   4 1
## 5   5 1
## 6   6 1
## 7   7 1
## 8   8 1
## 9   9 0
```

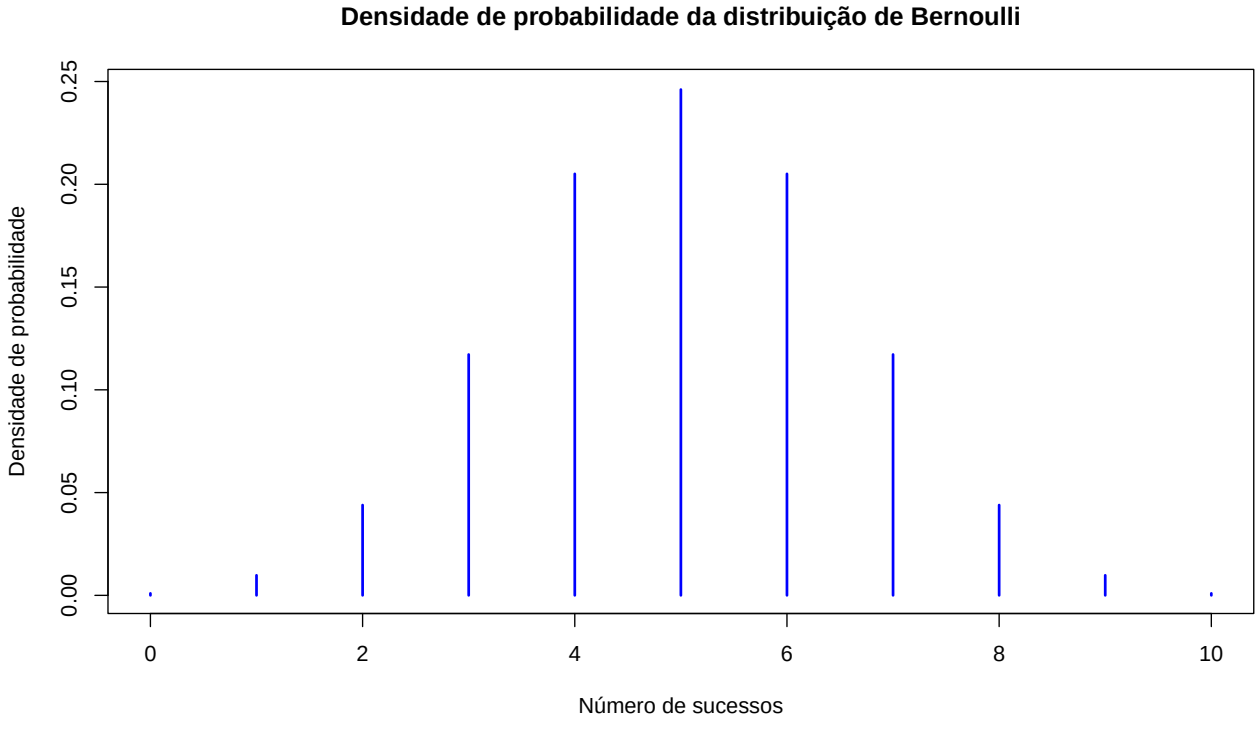
```
## 10 10 1
##     A B
## 1   1 1
## 2   2 1
## 3   3 1
## 4   4 1
## 5   5 1
## 6   6 1
## 7   7 1
## 8   8 1
## 9   9 1
## 10 10 1
```

Onde:

```
- o objeto `simulaBernoulli` guarda uma função não existente no ambiente de trabalho, criada pelo usuári
- dentro do parênteses da função, declaramos que a mesma é funnção de uma variável chamada `probabilidad
  no ambiente de trabalho, com $i$ termos;
- então, chama-se o loop `for` para iterar sobre os valores de `probabilidades` - isto é, para cada val
  aplique a função `rbinom` para gerar 10 valores binomiais com probabilidade $i_j$ de sucesso, e guard
- após, a função indica para que se guarde os resultados dos vetores `A` e `B` em um `data.frame` chamad
- a função é fechada, e então chamada para ser executada com o argumento `probabilidadesBernoulli`.
- note que o argumento `probabilidadesBernoulli` não foi previamente definido no ambiente de trabalho -
```

`probabilidadesBernoulli` é provocado pelo usuário no momento da chamada da função, podendo ficar livre para atribuir qualquer valor (nome) à saída da função. - a única restrição é que, a partir desse momento, o objeto `probabilidadesBernoulli` passa a existir no ambiente de trabalho, e sempre que for chamado, guardará o resultado das iterações, já que a linguagem R é tem paradigma funcional e não procedural, e é orientada a objetos, além de que sua tipagem é dinâmica. Isso implica que sempre que a função for chamada, o objeto `probabilidadesBernoulli` será sobrescrito/modificado com o novo resultado.

Visualização gráfica da densidade de probabilidade da distribuição de Bernoulli por estratos de probabilidade prévia:



E no quê a Distribuição de Bernoulli se relaciona com a Distribuição Binomial e Modelos Logísticos?

A distribuição binomial é uma **generalização** da distribuição de Bernoulli, e é utilizada para modelar o número de sucessos em uma sequência de n tentativas.

NÃO CONFUNDIR

A distribuição de Bernoulli é utilizada para modelar um único **futuro** evento de sucesso ou fracasso, enquanto a distribuição binomial é utilizada nos permite modelar o número de sucessos e fracassos para **futuras** tentativas. Para o exemplo da moeda, a distribuição de Bernoulli modela a probabilidade de sucesso exclusivamente dada **próxima** tentativa dadas as k tentativas anteriores. Já a distribuição binomial modela a probabilidade de sucesso e fracasso para as próximas n tentativas (virtualmente **todas**), dadas as k tentativas anterioes.

Isto é particularmente útil para inferência estatística buscando estimar parâmetros populacionais a partir de amostras, que sejam generalizáveis "sem prazo de validade" para a população de interesse. Do ponto de vista matemático, a distribuição binomial é:

- **Discreta**: o número de sucessos em n tentativas é sempre um número inteiro.
- **Esperança matemática**: $E(X) = n \cdot p$, onde n é o número de tentativas e p é a probabilidade de sucesso. Ou seja, a esperança matemática é influenciada pelo número de tentativas e pela probabilidade de sucesso do evento.
- **Variância**: $Var(X) = n \cdot p \cdot (1 - p)$, onde n é o número de tentativas e p é a probabilidade de sucesso. Ou seja, a variância é influenciada pelo número de tentativas e pela probabilidade de sucesso do evento, com maior peso para a probabilidade de sucesso..

Distribuição Binomial

Como dito anteriormente, os modelos de distribuição binomial são uma generalização dos modelos de Bernoulli. A distribuição binomial é uma distribuição de probabilidades discreta, assumindo valores inteiros não negativos.

Se a variável aleatória X segue uma distribuição binomial com parâmetros $n \in \mathbb{N}$ e $p \in [0, 1]$, denotamos $X \sim B(n, p)$. Isto é, a V.A. X segue uma distribuição de probabilidades binomial, aqui chamada de B, explicada pelos parâmetros n e p.

Função de Probabilidade de Massa

A probabilidade de massa de uma distribuição de probabilidades significa a probabilidade de que a variável aleatória assuma um valor específico.

Dado que o contra-domínio da distribuição binomial é finito - $X \in \{0, 1, 2, ..., n\}$, estimar a probabilidade de que a V.A. assuma um dos valores possíveis do C.D. é produto da **função probabilidade de massa**.

Função de Distribuição Acumulada

A função de distribuição acumulada (f.d.a.) de uma distribuição de probabilidades é a probabilidade de que a variável aleatória assuma um valor menor ou igual a um valor específico. No caso da distribuição binomial, a f.d.a. se aproxima de uma função beta incompleta regularizada. Aqui, podemos chamar a f.d.a. por função cumulativa da distribuição F (caso especial). Sua notação completa se dá por:

$$f(k, n, p) = (n - k)\binom{n}{k} \int_0^{1-p} t^{n-k-1}(1 - t)^k dt$$

Função Densidade de Probabilidade

A função densidade de probabilidade (f.d.p.) de uma distribuição de probabilidades é a derivada da função de distribuição acumulada. É provavelmente a função mais importante de uma distribuição de probabilidades, pois é a partir dela que podemos calcular a probabilidade de um intervalo de valores, particularmente em inferência estatística, testes de hipóteses e modelos com população desconhecida.

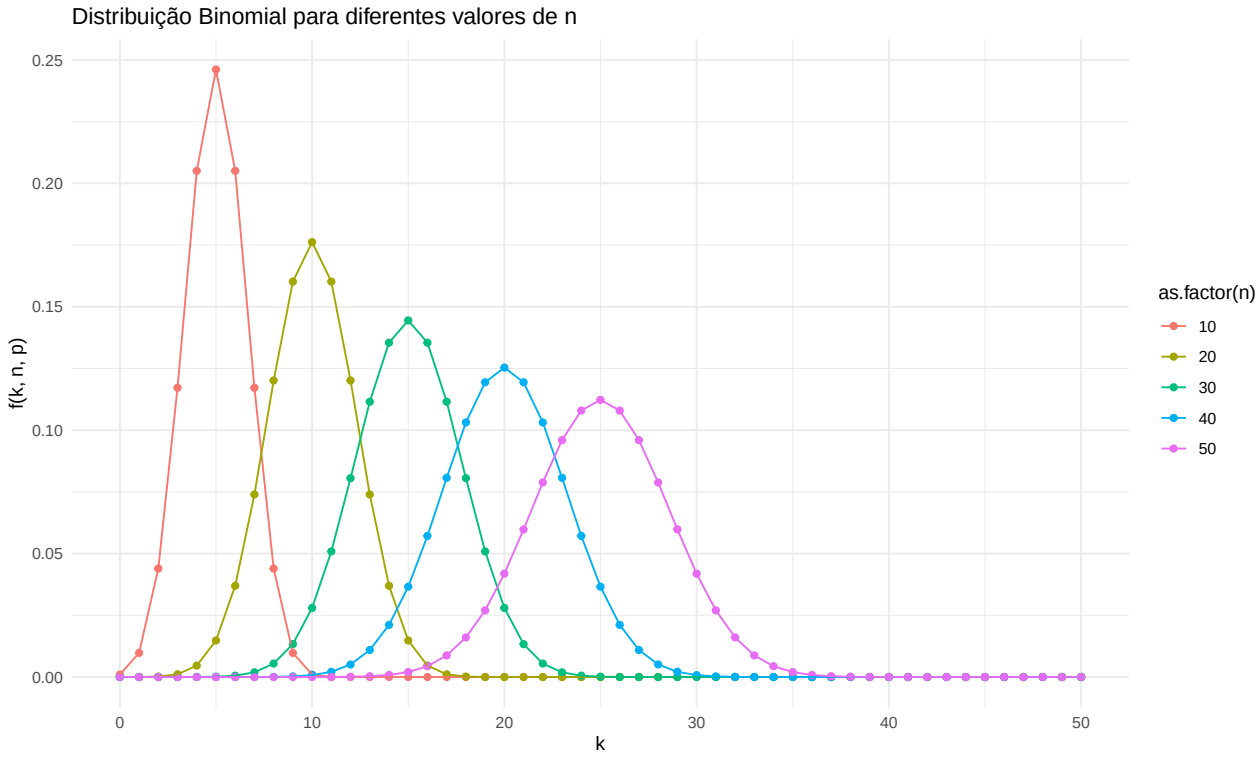
Para a distribuição binomial, a f.d.p. tem as seguintes características:

- A f.d.p. não assume valores negativos;
- A área sob a curva da f.d.p. é igual a 1;
- Pode ser aproximada por uma distribuição normal, quando $n \to \infty$ e $p \to 0.5$ - i.e., Teorema do Limite Central.

Representação Gráfica da Distribuição Binomial para diferentes parâmetros (n, p)

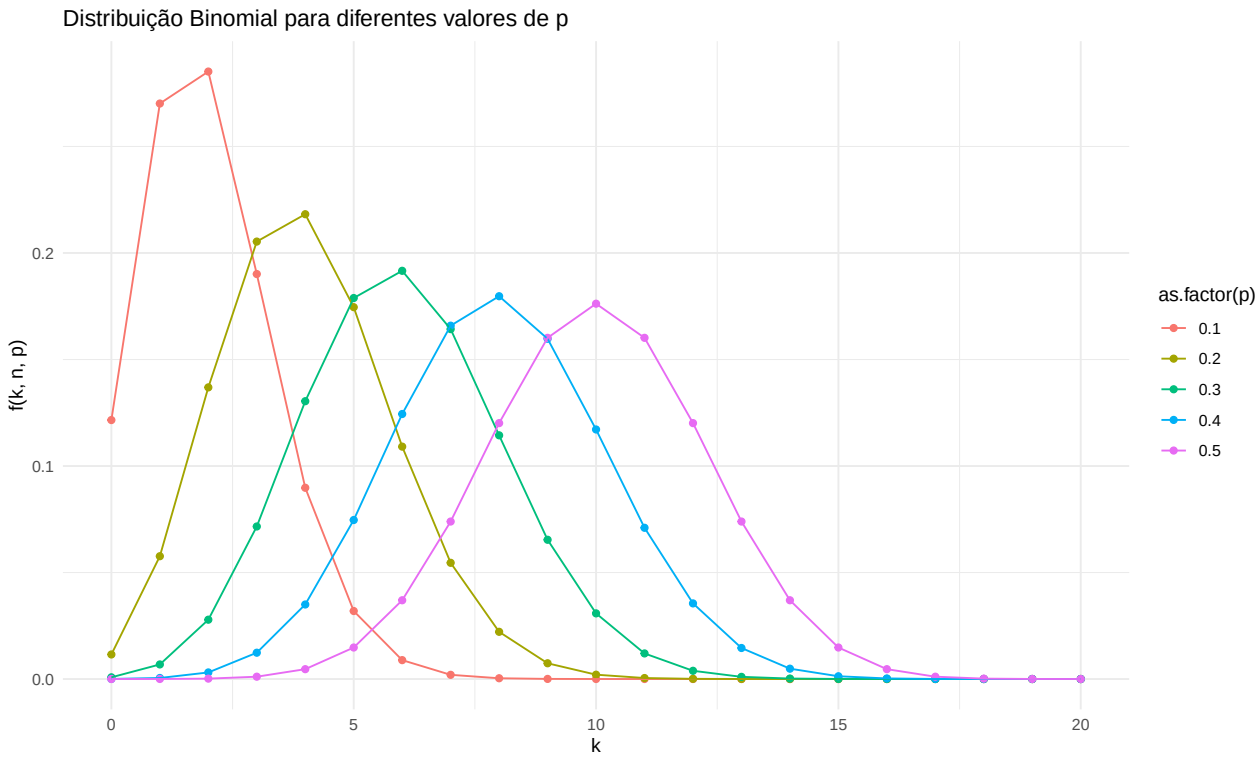
A seguir, temos a representação gráfica da distribuição binomial para diferentes valores de n e p. Assumamos, primeiramente, p fixo e n variável, sobrepostas em um mesmo gráfico.

```
# com ggplot2

library(ggplot2)

# parâmetros
n <- c(10, 20, 30, 40, 50)
p <- 0.5

# função de probabilidade de massa
f <- function(k, n, p) {
  return(choose(n, k) * p^k * (1-p)^(n-k))
}

# valores de k
k <- seq(0, max(n), by = 1)

# data frame
df <- expand.grid(k = k, n = n)
df$p <- p
df$f <- mapply(f, df$k, df$n, df$p)

# gráfico
ggplot(df, aes(x = k, y = f, color = as.factor(n))) +
  geom_point() +
  geom_line() +
  labs(title = "Distribuição Binomial para diferentes valores de n",
       x = "k", y = "f(k, n, p)") +
  theme_minimal()
```



Agora, assumamos n fixo e p variável, sobrepostas em um mesmo gráfico.

```r
# parâmetros
n <- 20
p <- c(0.1, 0.2, 0.3, 0.4, 0.5)

# função de probabilidade de massa
f <- function(k, n, p) {
  return(choose(n, k) * p^k * (1-p)^(n-k))
}

# valores de k
k <- seq(0, n, by = 1)

# data frame
df <- expand.grid(k = k, p = p)
df$n <- n
df$f <- mapply(f, df$k, df$n, df$p)

# gráfico
ggplot(df, aes(x = k, y = f, color = as.factor(p))) +
  geom_point() +
  geom_line() +
  labs(title = "Distribuição Binomial para diferentes valores de p",
       x = "k", y = "f(k, n, p)") +
  theme_minimal()
```



Modelo Binomial

Quando V.A. de interesse têm distribuição binomial e deseja-se verificar a associação entre uma ou mais variáveis potenciais preditoras da resposta da V.A., comumente, mas não exclusivamente, recorre-se a modelos de regressão.

Neste caso, os modelos não são lineares - isto é, o comportamento da variável resposta não é linear em relação às variáveis preditoras, diferente da regressão linear simples ou múltipla. Porém, para a estimação, é possível, por meio de transformações, aproximar o modelo a um modelo linear.

Não que seja problemática a ausência de linearidade entre as variáveis; a opção por se buscar alternativas para "linearizar" modelos, como neste caso, se dá por razões tanto algébricas, quanto computacionais e de interpretação dos resultados, além de poder enquadrar os estimandos no Teorema de Gauss-Markov e no Teorema do Limite Central.

Logito

Da "linearização" de modelos por meio de intermediários, surge uma nova família de modelos, que são os modelos lineares generalizados (GLM). Tal como os binomiais, exemplifica-se o modelo de Poisson, que é um caso particular dos GLM; o modelo binomial negativo; o modelo gamma; o modelo logit; o modelo probit; entre outros. A diferença entre LMs e GLMs é que, no primeiro, a variável resposta é confrontada com conjuntos de dados que representam as variáveis preditoras e/ou confundidoras, e há um método (usualmente numérico, mas pode ser analítico) para a estimação do(s) parâmetro(s) do modelo - como o método dos mínimos quadrados.

Já nos GLMs, a variável resposta é confrontada com um conjunto de dados que representam as variáveis preditoras e/ou confundidoras, há um método para a estimação dos parâmetros do modelo, mas adiciona-se um recurso algébrico e probabilístico, que é a **função de ligação**. Stricto sensu, a função de ligação é uma função que relaciona a média da variável resposta com as variáveis preditoras e/ou confundidoras, e é a partir dela que se estima o modelo. A partir da distribuição da V.A. resposta, para uma mesma

estratégia de regressão, a critério do pesquisador, pode-se escolher diferentes distribuições de probabilidades "intermediárias", que, por sua vez, implicam em diferentes resultados para o modelo, já linearizado.

Definição de Logito

O logito é um artefato algébrico representado por:

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

onde $p \in [0, 1]$. Ou seja, o logito é uma **transformação logarítmica** da razão entre a probabilidade de sucesso e a probabilidade de fracasso.

Na modelagem de eventos do tipo sim/não, o logito surge temporalmente na dedução algébrica para linearizar o modelo, e é uma das funções de ligação mais utilizadas em modelos binomiais. Não é novidade que a transformação logarítmica comumente é utilizada para linearizar dados ou tornar dados mais simétricos sem viola-los. Com este conjunto de dados simulados para duas variáveis contínuas A e B, que num primeiro momento não apresentam relação linear, demonstrarei como a transformação logarítmica consegue fazer isto - tanto por inspeção visual, quanto por meio do coeficiente de correlação de Pearson (também exponencializado), quanto por meio de um modelo de regressão linear simples e testes de diagnóstico do modelo de regressão.

```
# fake data

set.seed(123)
A <- rnorm(100, mean = 10, sd = 2)
B <- rnorm(100, mean = 5, sd = 1)

# plot por ggplot2

library(ggplot2)

df <- data.frame(A, B)

ggplot(df, aes(x = A, y = B)) +
  geom_point() +
  labs(title = "Transformação logarítmica de dados contínuos",
       x = "Variável A",
       y = "Variável B")
```

Correlação de Pearson entre A e B:

```
cor(df$A, df$B)
```

```
## [1] -0.04953215
```

Coeficiente de Determinação entre A e B:

```
cor(df$A, df$B)^2
```

```
## [1] 0.002453434
```

Relação entre variância de A e B:

```
var(df$A) / var(df$B)
```

```
## [1] 3.564392
```

Modelo de Regressão Linear Simples entre A e B:

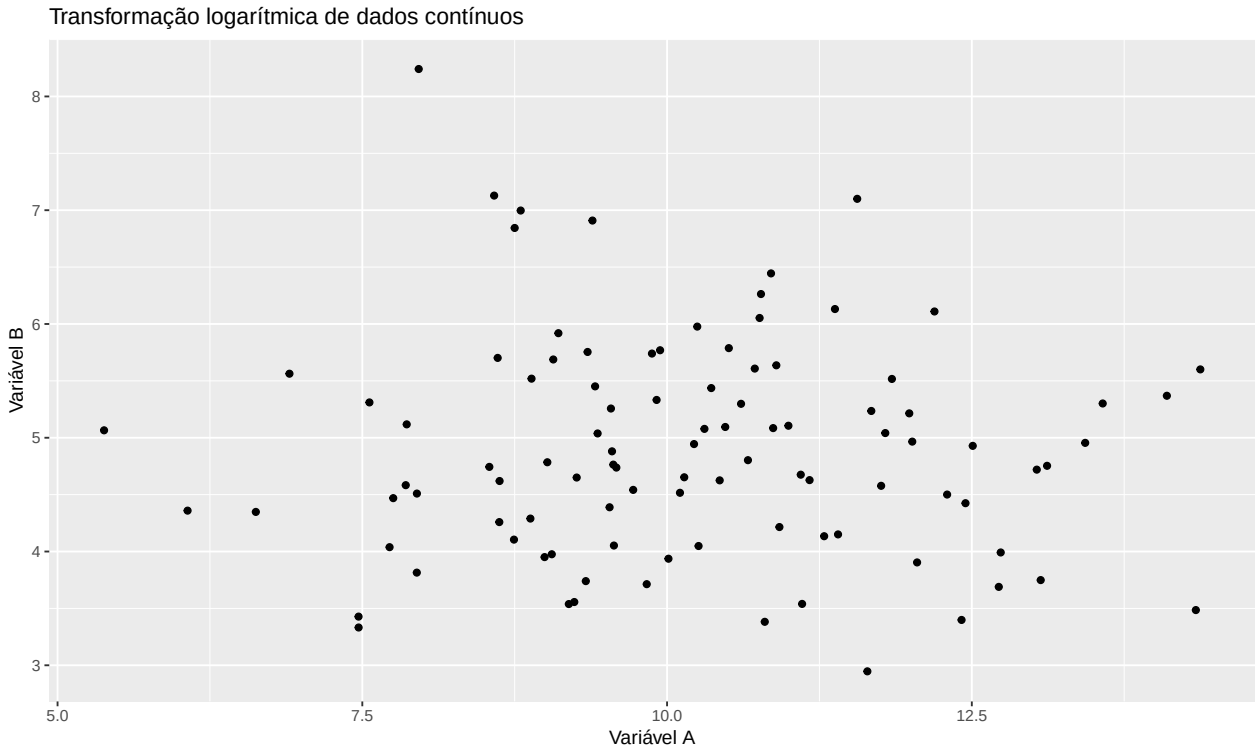


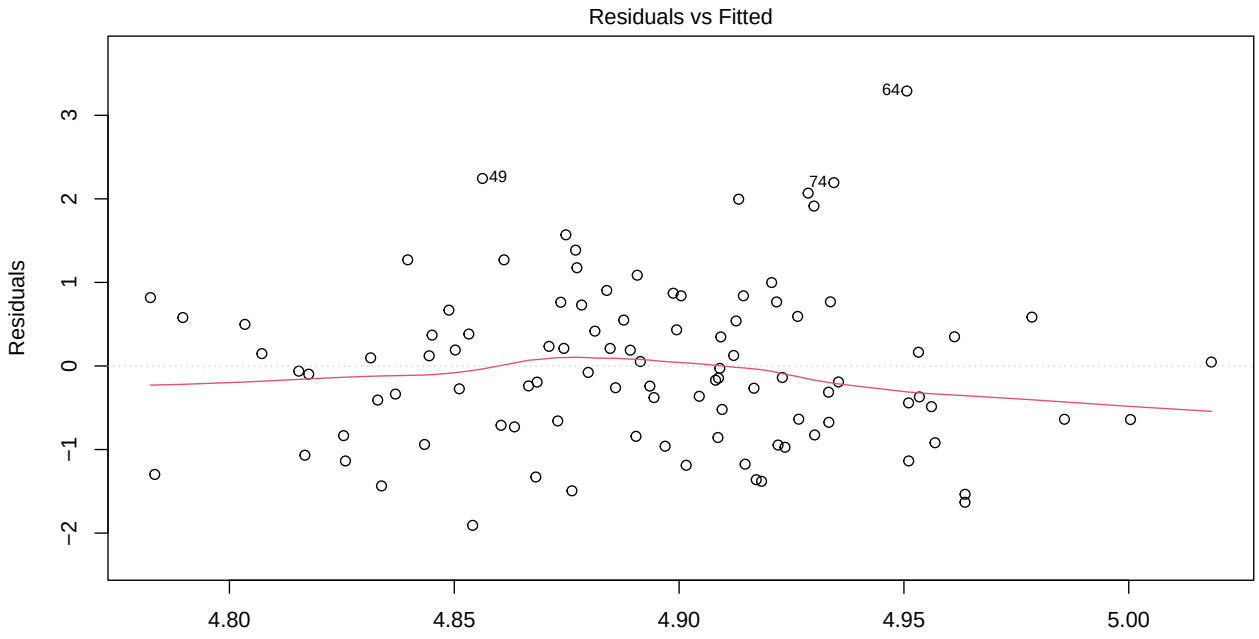
Figure 1: Transformação logarítmica de dados contínuos

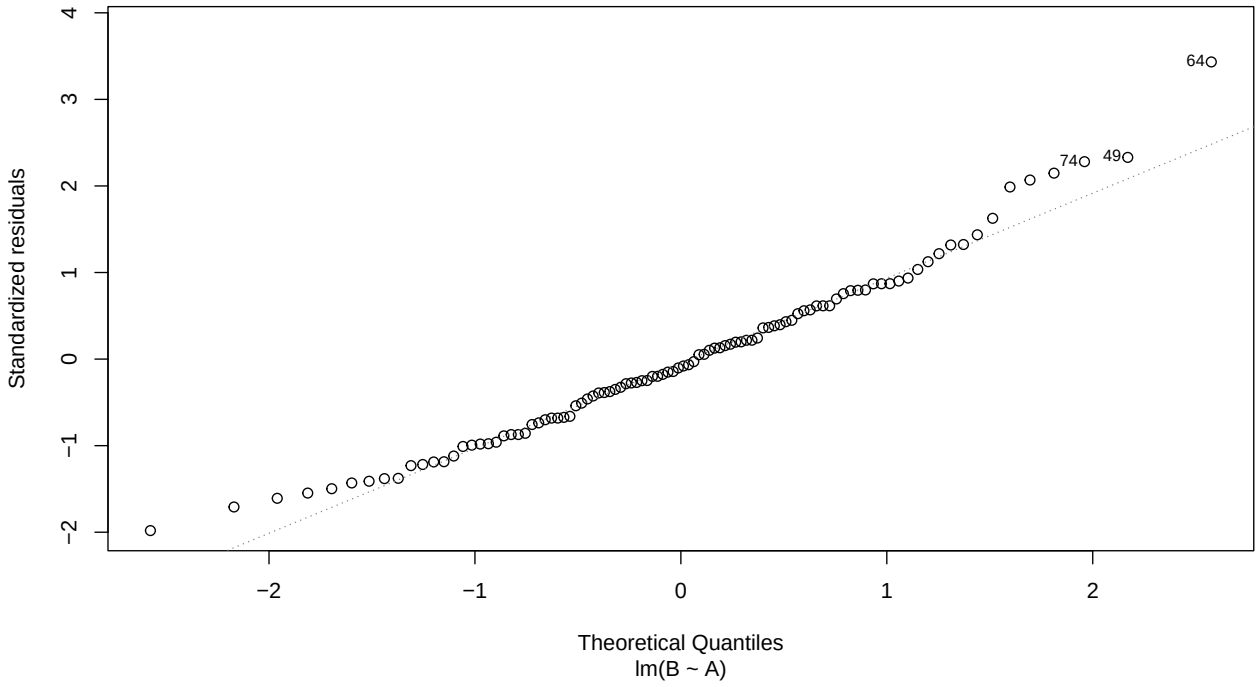
```
lm1 <- lm(B ~ A, data = df)
summary(lm1)
```

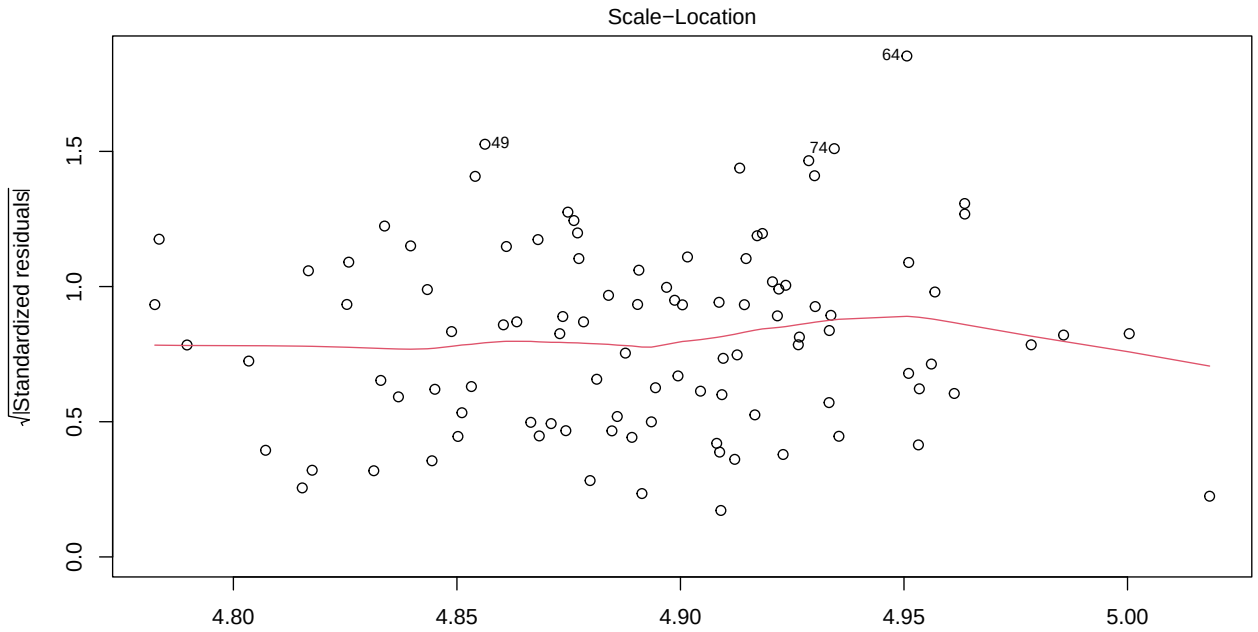
```
##
## Call:
## lm(formula = B ~ A, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9073 -0.6835 -0.0875  0.5806  3.2904
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.15956    0.55265   9.336 3.34e-15 ***
## A           -0.02624    0.05344  -0.491    0.625
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9707 on 98 degrees of freedom
## Multiple R-squared:  0.002453,   Adjusted R-squared:  -0.007726
## F-statistic: 0.241 on 1 and 98 DF,  p-value: 0.6246
```

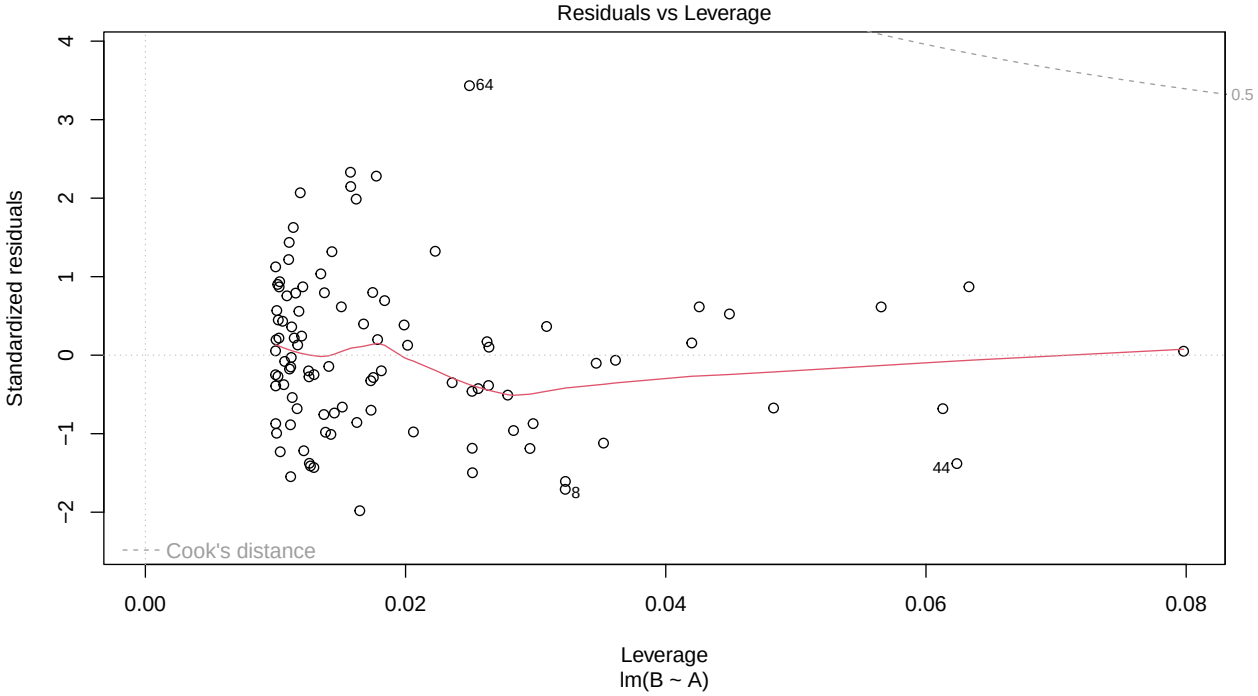
Diagnóstico do Modelo de Regressão Linear Simples entre A e B:

```
plot(lm1)
```


Residuals vs Fitted
Residuals
Fitted values
lm(B ~ A)
64
49
74
4.80
4.85
4.90
4.95
5.00
−2
−1
0
1
2
3


Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(B ~ A)
64
74
49
−2
−1
0
1
2
3
4





Agora, aplica-se a transformação logarítmica:

$$\log\left(\frac{B}{A}\right)$$

```r
df$logito <- log(df$B / df$A)

ggplot(df, aes(x = A, y = logito)) +
  geom_point() +
```

```
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Transformação logarítmica de dados contínuos",
       x = "Variável A",
       y = "Logito")
```

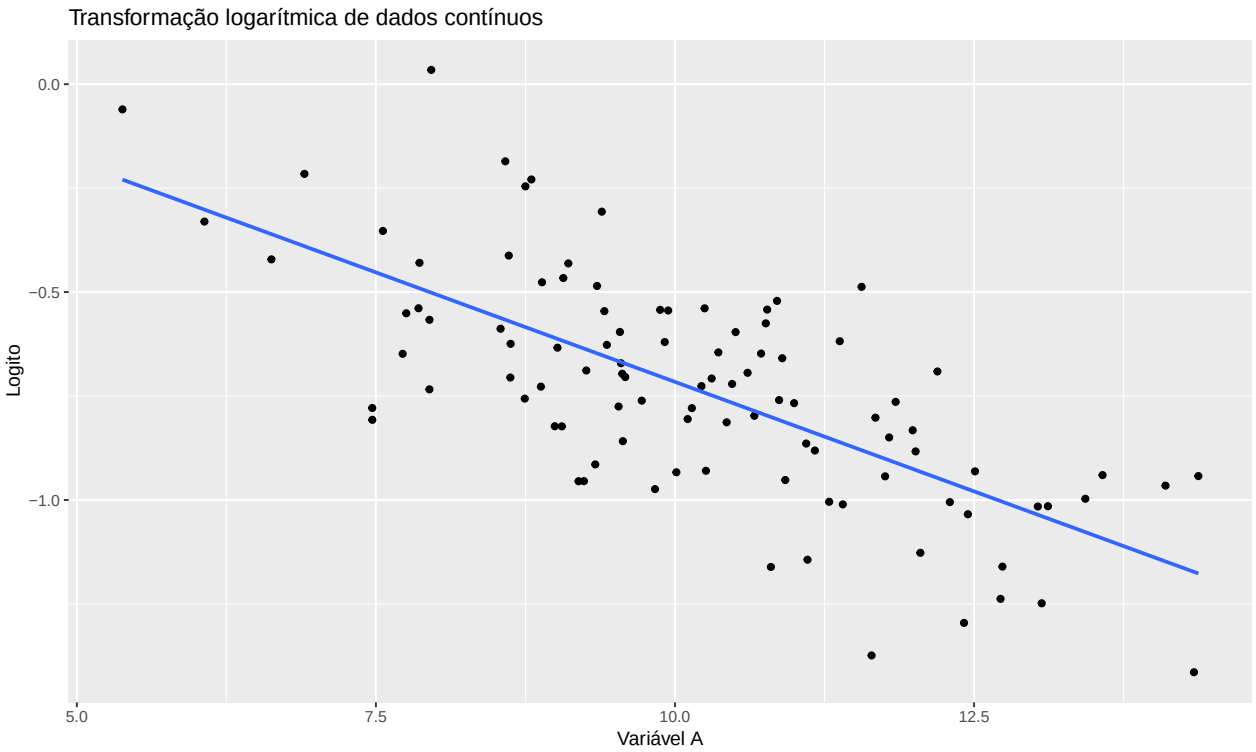


Figure 2: Transformação logarítmica de dados contínuos

Correlação de Pearson entre A e Logito:

```
cor(df$A, df$logito)
```

```
## [1] -0.7044913
```

Coeficiente de Determinação entre A e Logito:

```
cor(df$A, df$logito)^2
```

```
## [1] 0.496308
```

Relação entre variância de A e Logito:

```
var(df$A) / var(df$logito)
```

```
## [1] 44.72517
```

Modelo de Regressão Linear Simples entre A e Logito:

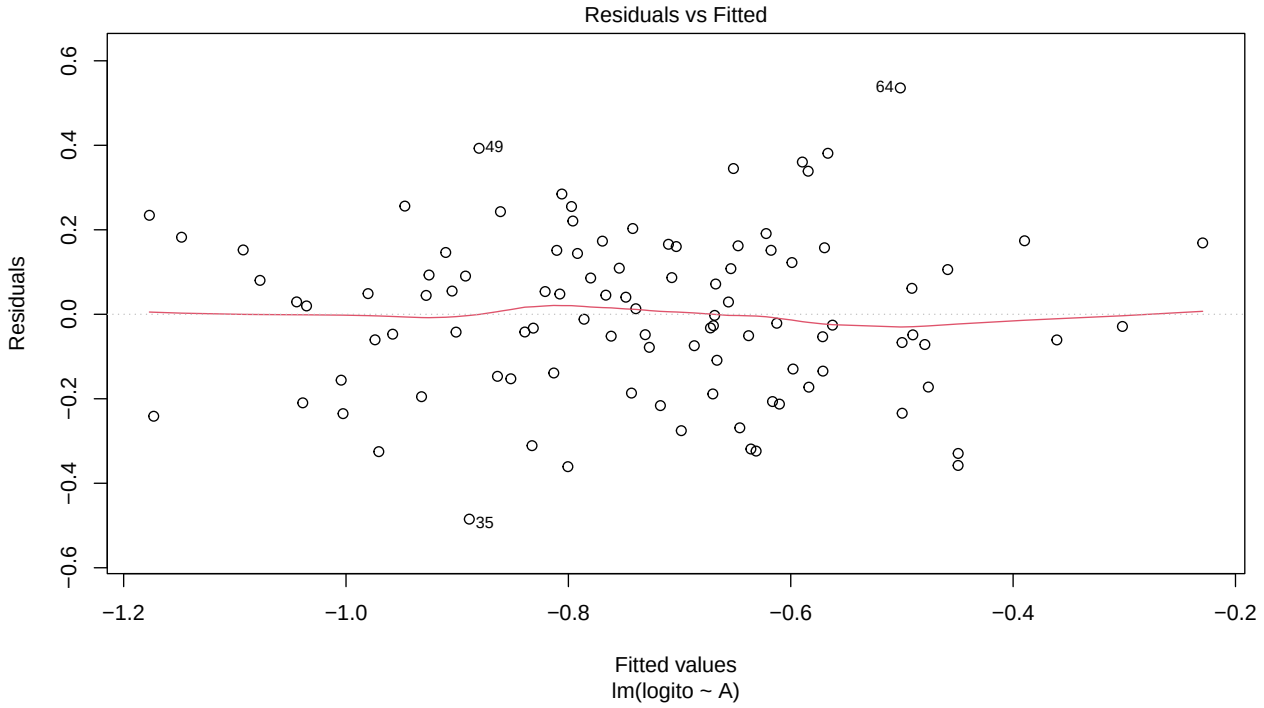
```
lm2 <- lm(logito ~ A, data = df)
summary(lm2)
```

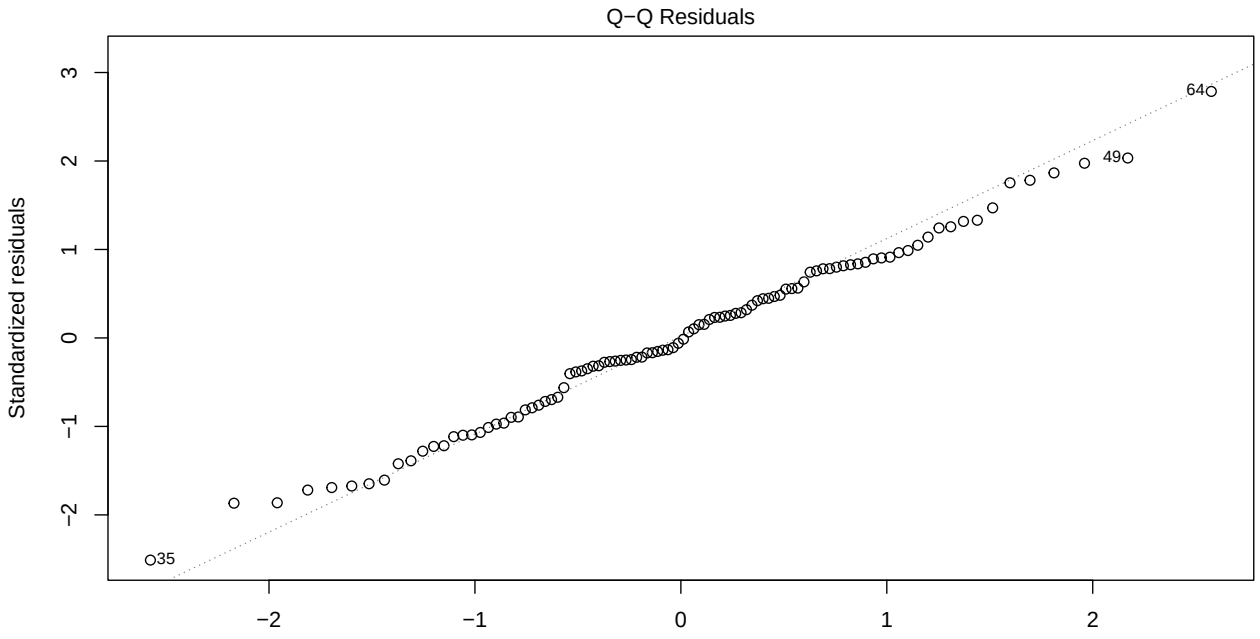
```
##
## Call:
## lm(formula = logito ~ A, data = df)
##
```

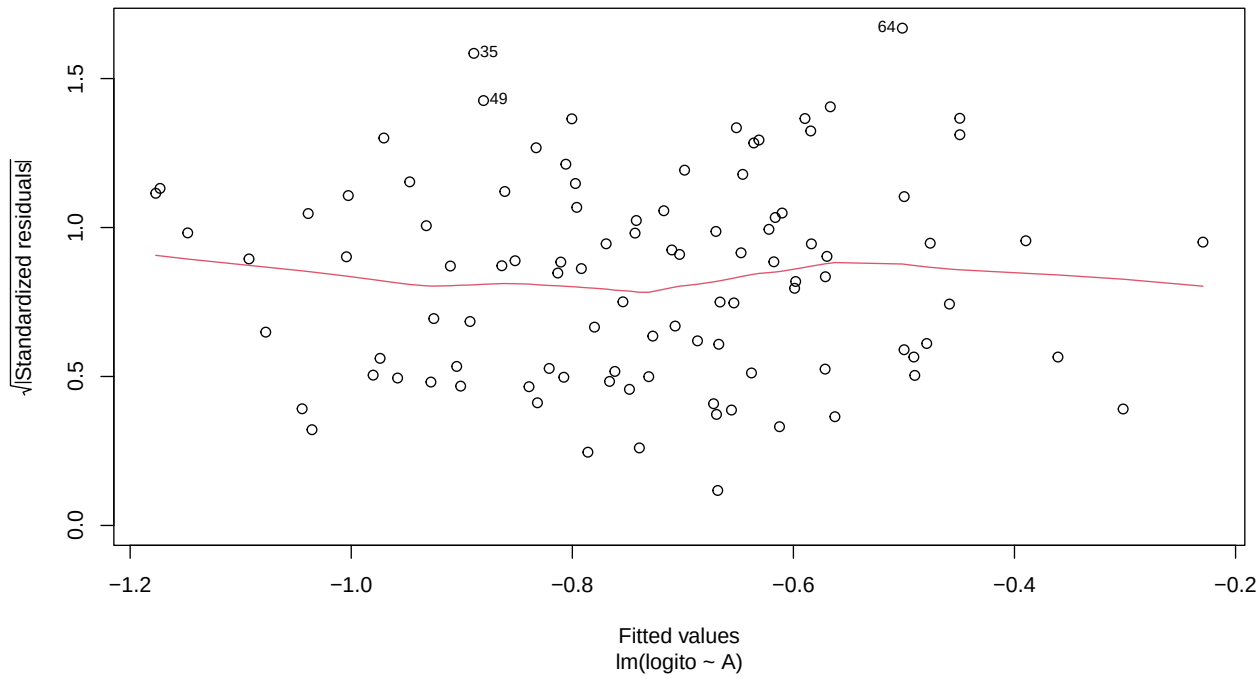
```
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.48490 -0.14094 -0.00719  0.14753  0.53576
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.33739    0.11086   3.043    0.003 **
## A           -0.10534    0.01072  -9.827 2.88e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1947 on 98 degrees of freedom
## Multiple R-squared:  0.4963, Adjusted R-squared:  0.4912
## F-statistic: 96.56 on 1 and 98 DF,  p-value: 2.878e-16
```

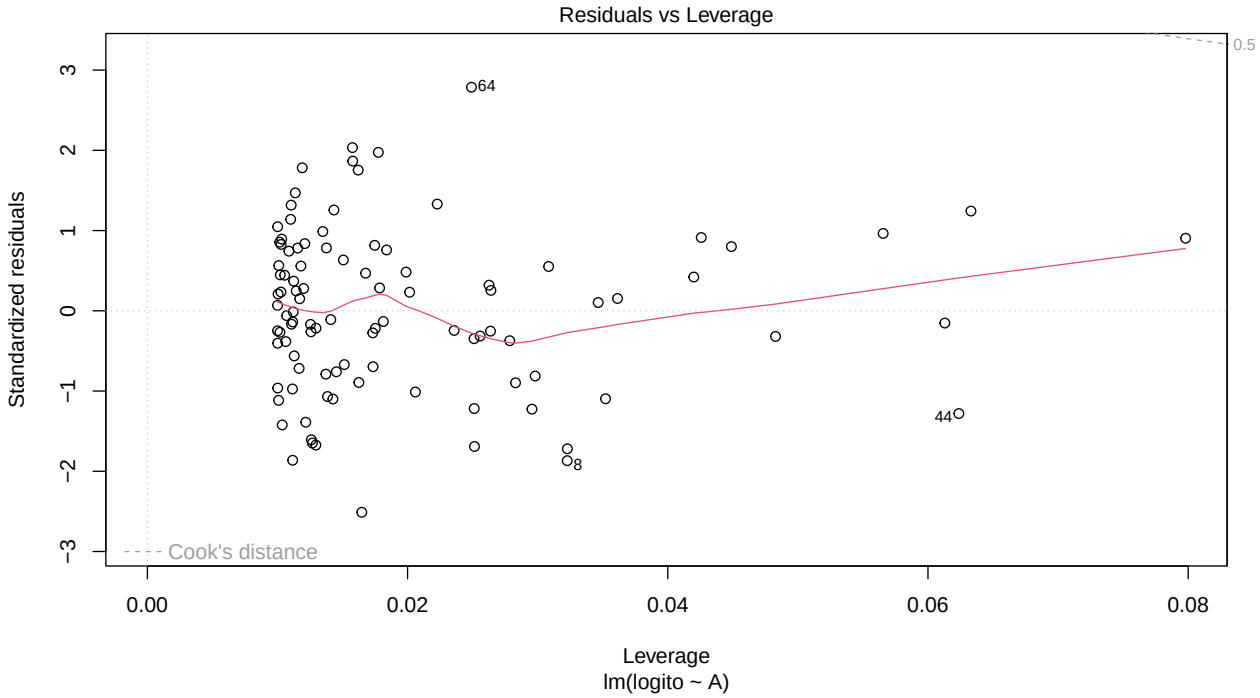
Diagnóstico do Modelo de Regressão Linear Simples entre A e Logito:

```
plot(lm2)
```




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(logito ~ A)
64
49
35
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
Fitted values
lm(logito ~ A)
64
35
49
0.0
0.5
1.0
1.5
−1.2
−1.0
−0.8
−0.6
−0.4
−0.2



Conclusão

```
anova(lm1, lm2)
```

Contrastes com e sem transformação logarítmica

```
## Analysis of Variance Table
##
## Response: B
##           Df Sum Sq Mean Sq F value Pr(>F)
## A          1  0.227 0.22712   0.241 0.6246
## Residuals 98 92.344 0.94229
```

```
AIC(lm1, lm2)
```

Comparação de modelos

```
##     df       AIC
## lm1  3 281.82290
## lm2  3 -39.46478
```

```
BIC(lm1, lm2)
```

Comparação de ajustes

```
##     df       BIC
## lm1  3 289.63841
## lm2  3 -31.64927
```

```
abs(cor(df$A, df$B) - cor(df$A, df$logito))
```

Diferença entre Coefientes de Correlação de Pearson (absoluta e relativa)

```
## [1] 0.6549591
```

```
cor(df$A, df$logito) / cor(df$A, df$B)
```

```
## [1] 14.22291
```

Regressão Binomial Logit

O modelo de regressão binomial logit é um modelo de regressão que, a partir de uma variável resposta binomial, busca-se associar uma ou mais variáveis preditoras e/ou confundidoras à variável resposta.

A função de ligação do modelo binomial logit é o logito, que é uma transformação logarítmica da razão entre a probabilidade de sucesso e a probabilidade de fracasso. A partir da função de ligação, é possível estimar o modelo de regressão binomial logit, que é um modelo não linear, mas que, por meio de transformações, pode ser aproximado a um modelo linear.

Dado que os desfechos são do tipo contagem/frequência, o estimando pode ser exponencializado - isto é, a saída primária do modelo traz a unidade do desfecho em escala logarítmica (logito). Por definição, o logito elevado à base do número de Euler (logito^e) é a razão de chances (odds ratio), que é um estimador de associação entre as variáveis preditoras e a variável resposta em escala multiplicativa, sendo a divisão entre a probabilidade de sucesso e a probabilidade de fracasso.

Para os exemplos em diante, será usado o seguinte conjunto de dados fictício:

Dados:

- `tempo`: tempo de exposição ao fator de risco (em categorias de 10 anos: 0-10, 10-20, 20-30, 30-40)
- `idade`: idade do indivíduo (em categorias de 10 anos: 50-60, 60-70 e 70-80)
- `sexo biológico`: sexo biológico do indivíduo (em categorias: m/f)
- `desfecho`: se o indivíduo desenvolveu IAM ou não (em categorias: sim/não)
- `fumante`: se o indivíduo é fumante ou não (em categorias: sim/não)
- `historia_previa`: se o indivíduo tem história prévia de IAM na família ou não (em categorias: sim/não)

O banco de dados deverá ter a seguinte estrutura: o tempo de exposição varia de 0 a 40 anos, com mediana de 30 anos, com maior concentração nos homens e que são da faixa etária de 60-70 (60% da exposição). Dentre os expostos (fumantes), 70% são homens, 30% tem menos dee 60 anos, e deram conta de 70% dos infartos. Dentre os pacientes que fizeram IAM, 70% tinham história prévia de IAM na família em primeiro grau. Com base nisso, monta-se o banco com sampling aleatório simples, e atribui-se probabilidades pelo conhecimento da literatura.

```
# mock data

set.seed(123)
n <- 1000

tempo <- sample(c(0, 10, 20, 30, 40), n, replace = TRUE, prob = c(0.05, 0.1, 0.15, 0.3, 0.4))
idade <- sample(c(50, 60, 70, 80), n, replace = TRUE, prob = c(0.1, 0.3, 0.6, 0))
sexo <- sample(c("m", "f"), n, replace = TRUE, prob = c(0.6, 0.4))
desfecho <- sample(c("sim", "não"), n, replace = TRUE, prob = c(0.7, 0.3))
fumante <- sample(c("sim", "não"), n, replace = TRUE, prob = c(0.7, 0.3))
historia_previa <- sample(c("sim", "não"), n, replace = TRUE, prob = c(0.7, 0.3))

# ajuste de probabilidade para historia previa - dentre iam, 70% tem historia previa

historia_previa[desfecho == "sim"] <- sample(c("sim", "não"), sum(desfecho == "sim"), replace = TRUE, pr

df <- data.frame(tempo, idade, sexo, desfecho, fumante, historia_previa)
```

Estrutura do banco de dados:

```
head(df)
```

```
##   tempo idade sexo desfecho fumante historia_previa
## 1    40    70    m      sim     sim            sim
## 2    20    70    m      não     não            não
## 3    30    70    m      sim     sim            sim
## 4    10    60    m      sim     não            sim
## 5    10    60    m      sim     sim            sim
## 6    40    70    f      sim     sim            sim
```

Atribuição de 0 ou 1 para as variáveis binárias (mulher == 0, não == 0):

```
df$sexo <- ifelse(df$sexo == "m", 1, 0)
df$desfecho <- ifelse(df$desfecho == "sim", 1, 0)
df$fumante <- ifelse(df$fumante == "sim", 1, 0)
df$historia_previa <- ifelse(df$historia_previa == "sim", 1, 0)
```

Distribuição de Frequência das Variáveis:

```
table(df$desfecho)
```

```
##
##   0   1
## 306 694
```

```
table(df$fumante)
```

```
##
##   0   1
## 282 718
```

```
table(df$historia_previa)
```

```
##
##   0   1
## 299 701
```

Modelo de Regressão Binomial Logit sem história prévia de IAM na família:

```
library(MASS)

modelo1 <- glm(desfecho ~ tempo + idade + sexo + fumante, data = df, family = binomial(link = "logit"))
summary(modelo1)
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + sexo + fumante, family = binomial(link = "logit"),
##     data = df)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.108050   0.721518  -0.150   0.8810
## tempo       -0.004193   0.005895  -0.711   0.4769
## idade        0.020512   0.010412   1.970   0.0488 *
## sexo        -0.197411   0.141017  -1.400   0.1615
## fumante     -0.233282   0.156553  -1.490   0.1362
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1231.7  on 999  degrees of freedom
## Residual deviance: 1223.2  on 995  degrees of freedom
## AIC: 1233.2
##
## Number of Fisher Scoring iterations: 4
```

Modelo de Regressão Binomial Logit sem história prévia de IAM na família e com interação entre sexo e fumante:

```
modelo2 <- glm(desfecho ~ tempo + idade + sexo + fumante + sexo:fumante, data = df, family = binomial(li
summary(modelo2)
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + sexo + fumante + sexo:fumante,
##     family = binomial(link = "logit"), data = df)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.077059   0.743867   0.104   0.9175
## tempo        -0.004229   0.005904  -0.716   0.4738
## idade         0.020379   0.010415   1.957   0.0504 .
## sexo         -0.463090   0.287317  -1.612   0.1070
## fumante      -0.459603   0.266387  -1.725   0.0845 .
## sexo:fumante  0.353263   0.330095   1.070   0.2845
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1231.7  on 999  degrees of freedom
## Residual deviance: 1222.1  on 994  degrees of freedom
## AIC: 1234.1
##
## Number of Fisher Scoring iterations: 4
```

Modelo de Regressão Binomial Logit com história prévia de IAM na família:

```
modelo3 <- glm(desfecho ~ tempo + idade + sexo + fumante + historia_previa, data = df, family = binomial
summary(modelo3)
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + sexo + fumante + historia_previa,
##     family = binomial(link = "logit"), data = df)
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.179837   0.727284  -0.247   0.8047
## tempo           -0.004085   0.005898  -0.693   0.4885
## idade            0.020315   0.010414   1.951   0.0511 .
## sexo            -0.203895   0.141308  -1.443   0.1490
## fumante         -0.228301   0.156689  -1.457   0.1451
```

```
## historia_previa  0.117297   0.149742   0.783   0.4334
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1231.7  on 999  degrees of freedom
## Residual deviance: 1222.6  on 994  degrees of freedom
## AIC: 1234.6
##
## Number of Fisher Scoring iterations: 4
```

Modelo de Regressão Binomial Logit com história prévia de IAM na família e interação história prévia e fumante:

```
modelo4 <- glm(desfecho ~ tempo + idade + sexo + fumante + historia_previa + fumante:historia_previa, da
summary(modelo4)
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + sexo + fumante + historia_previa +
##     fumante:historia_previa, family = binomial(link = "logit"),
##     data = df)
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -0.142853   0.756162  -0.189   0.8502
## tempo                   -0.004049   0.005901  -0.686   0.4927
## idade                    0.020243   0.010422   1.942   0.0521 .
## sexo                    -0.202647   0.141482  -1.432   0.1521
## fumante                 -0.272865   0.293974  -0.928   0.3533
## historia_previa          0.070215   0.302315   0.232   0.8163
## fumante:historia_previa  0.062477   0.347975   0.180   0.8575
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1231.7  on 999  degrees of freedom
## Residual deviance: 1222.6  on 993  degrees of freedom
## AIC: 1236.6
##
## Number of Fisher Scoring iterations: 4
```

Análise Estratificada por Sexo:

```
summary(glm(desfecho ~ tempo + idade + fumante + historia_previa, data = df[df$sexo == 1,], family = bi
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + fumante + historia_previa,
##     family = binomial(link = "logit"), data = df[df$sexo == 1,
##         ])
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept)     -0.7487217  0.9324706  -0.803   0.4220
## tempo           -0.0005961  0.0078501  -0.076   0.9395
## idade            0.0238793  0.0134512   1.775   0.0759 .
## fumante         -0.1025540  0.1957452  -0.524   0.6003
## historia_previa  0.0346028  0.1988606   0.174   0.8619
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 733.08  on 582  degrees of freedom
## Residual deviance: 729.64  on 578  degrees of freedom
## AIC: 739.64
##
## Number of Fisher Scoring iterations: 4
```

```
summary(glm(desfecho ~ tempo + idade + fumante + historia_previa, data = df[df$sexo == 0,], family = bi
```

```
##
## Call:
## glm(formula = desfecho ~ tempo + idade + fumante + historia_previa,
##     family = binomial(link = "logit"), data = df[df$sexo == 0,
##         ])
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.493613   1.158752   0.426   0.6701
## tempo           -0.008494   0.009046  -0.939   0.3477
## idade            0.013547   0.016537   0.819   0.4127
## fumante         -0.471159   0.266918  -1.765   0.0775 .
## historia_previa  0.243564   0.229962   1.059   0.2895
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 496.85  on 416  degrees of freedom
## Residual deviance: 490.60  on 412  degrees of freedom
## AIC: 500.6
##
## Number of Fisher Scoring iterations: 4
```

Análise de Sensibilidade por Subgrupos com teste de interação para análises estratificadas:

```
anova(modelo1, modelo2)
```

```
## Analysis of Deviance Table
##
## Model 1: desfecho ~ tempo + idade + sexo + fumante
## Model 2: desfecho ~ tempo + idade + sexo + fumante + sexo:fumante
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       995     1223.2
## 2       994     1222.1  1   1.1602   0.2814
```

```
anova(modelo3, modelo4)
```

```
## Analysis of Deviance Table
```

```
##
## Model 1: desfecho ~ tempo + idade + sexo + fumante + historia_previa
## Model 2: desfecho ~ tempo + idade + sexo + fumante + historia_previa +
##     fumante:historia_previa
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       994     1222.6
## 2       993     1222.6  1 0.032305   0.8574
```

Comparação de Modelos:

```
AIC(modelo1, modelo2, modelo3, modelo4)
```

```
##         df      AIC
## modelo1  5 1233.229
## modelo2  6 1234.069
## modelo3  6 1234.618
## modelo4  7 1236.586
```

```
BIC(modelo1, modelo2, modelo3, modelo4)
```

```
##         df      BIC
## modelo1  5 1257.768
## modelo2  6 1263.515
## modelo3  6 1264.065
## modelo4  7 1270.940
```

Comparação de Ajustes:

```
anova(modelo1, modelo2)
```

```
## Analysis of Deviance Table
##
## Model 1: desfecho ~ tempo + idade + sexo + fumante
## Model 2: desfecho ~ tempo + idade + sexo + fumante + sexo:fumante
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       995     1223.2
## 2       994     1222.1  1   1.1602   0.2814
```

```
anova(modelo3, modelo4)
```

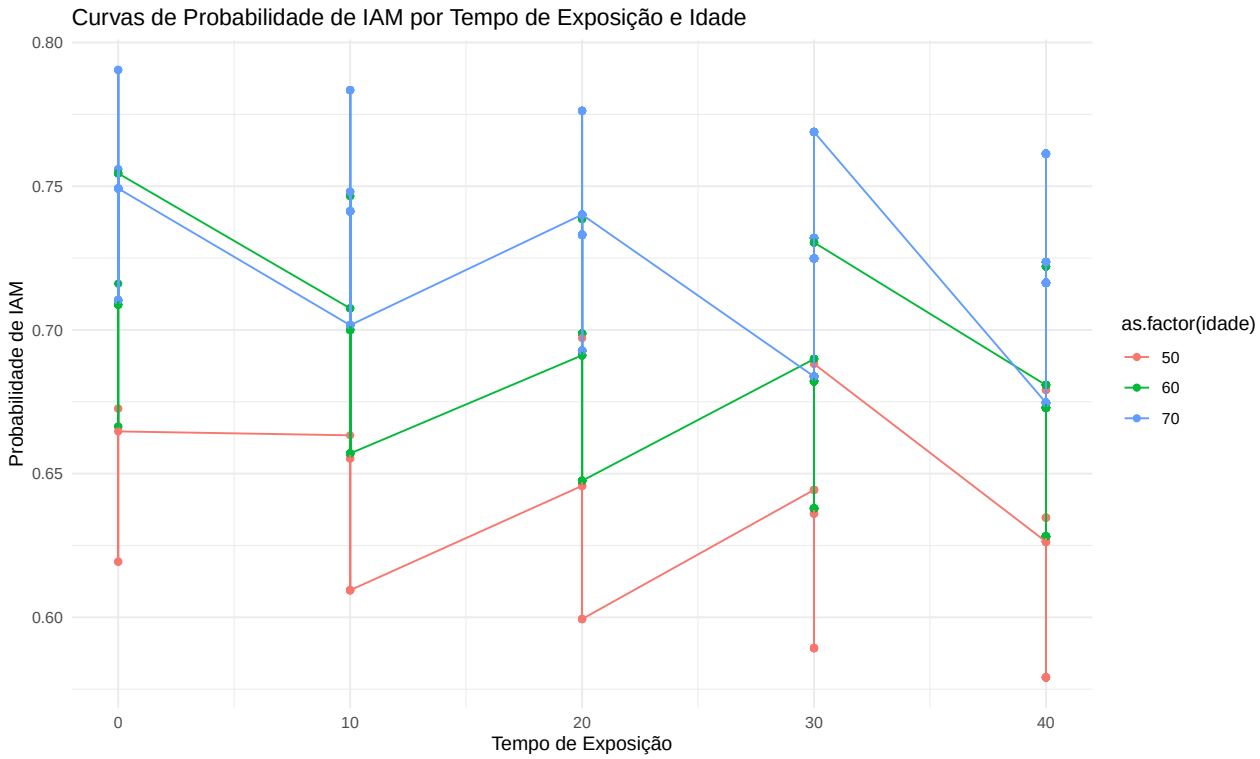
```
## Analysis of Deviance Table
##
## Model 1: desfecho ~ tempo + idade + sexo + fumante + historia_previa
## Model 2: desfecho ~ tempo + idade + sexo + fumante + historia_previa +
##     fumante:historia_previa
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       994     1222.6
## 2       993     1222.6  1 0.032305   0.8574
```

Visualização gráfica dos resultados

Curvas de Probabilidade de IAM por Tempo de Exposição e Idade:

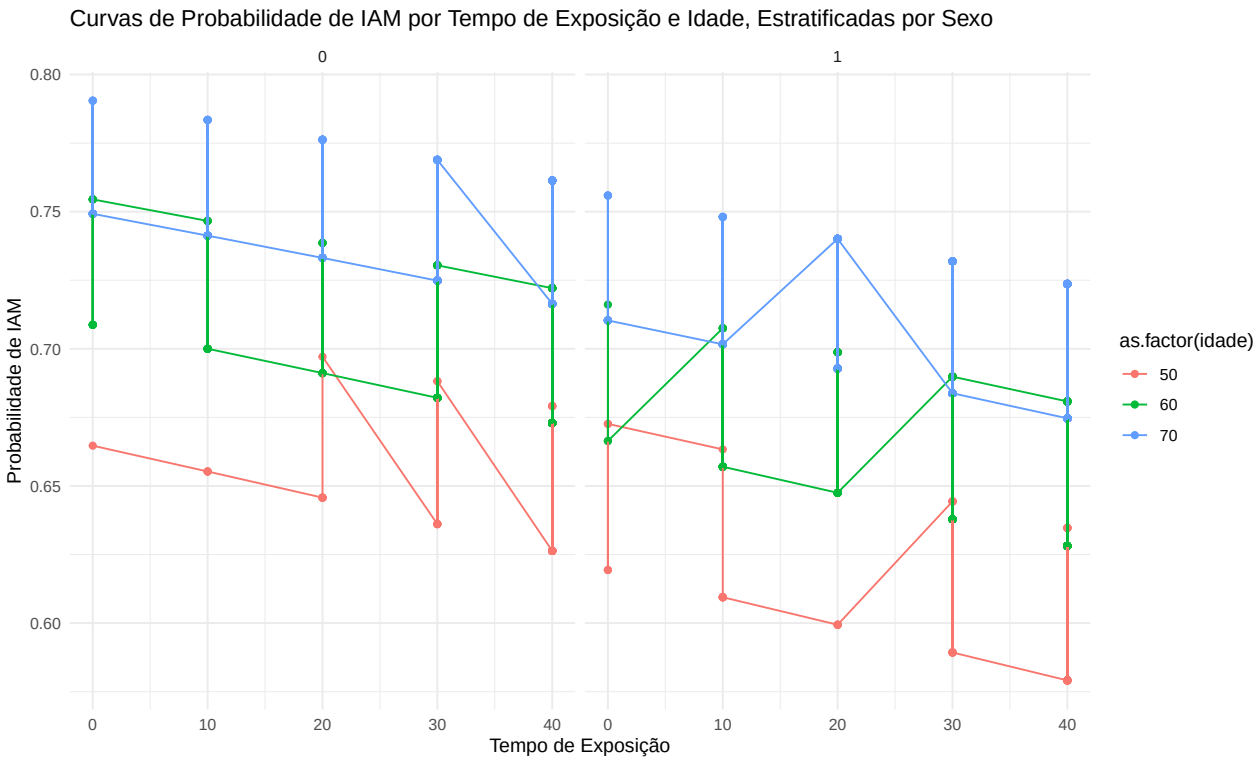
```
library(ggplot2)

df$prob <- predict(modelo1, type = "response")

ggplot(df, aes(x = tempo, y = prob, color = as.factor(idade))) +
  geom_point() +
```

```
  geom_line() +
  labs(title = "Curvas de Probabilidade de IAM por Tempo de Exposição e Idade",
       x = "Tempo de Exposição",
       y = "Probabilidade de IAM") +
  theme_minimal()
```



Curvas de Probabilidade de IAM por Tempo de Exposição e Idade, Estratificadas por Sexo:

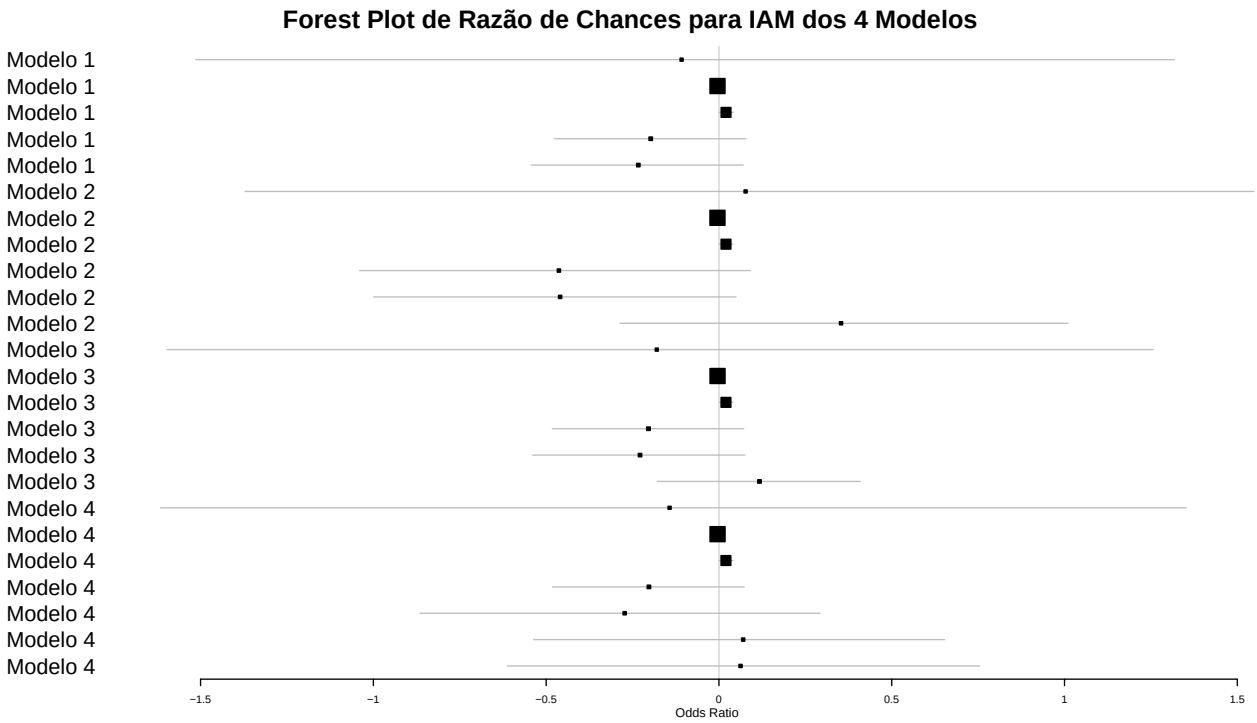
```
df$prob <- predict(modelo1, type = "response")

ggplot(df, aes(x = tempo, y = prob, color = as.factor(idade))) +
  geom_point() +
  geom_line() +
  facet_wrap(~sexo) +
  labs(title = "Curvas de Probabilidade de IAM por Tempo de Exposição e Idade, Estratificadas por Sexo"
       x = "Tempo de Exposição",
       y = "Probabilidade de IAM") +
  theme_minimal()
```



Forest plot de razão de chances (odds ratio) para IAM dos 4 modelos no mesmo gráfico

```
library(forestplot)

# extração dos coeficientes e intervalos de confiança

coef1 <- coef(modelo1)
confint1 <- confint(modelo1)

coef2 <- coef(modelo2)
confint2 <- confint(modelo2)

coef3 <- coef(modelo3)
confint3 <- confint(modelo3)

coef4 <- coef(modelo4)
confint4 <- confint(modelo4)

# criação do gráfico

# Create a matrix for labeltext
# Assuming each model has the same number of coefficients
num_coef1 <- length(coef1)
num_coef2 <- length(coef2)
num_coef3 <- length(coef3)
num_coef4 <- length(coef4)

# Total number of coefficients
```

```r
total_coef <- num_coef1 + num_coef2 + num_coef3 + num_coef4

# Create labeltext matrix with the correct number of rows
labeltext <- matrix(c(rep("Modelo 1", num_coef1), rep("Modelo 2", num_coef2), rep("Modelo 3", num_coef3

forestplot(labeltext = labeltext,
           mean = c(coef1, coef2, coef3, coef4),
           lower = c(confint1[,1], confint2[,1], confint3[,1], confint4[,1]),
           upper = c(confint1[,2], confint2[,2], confint3[,2], confint4[,2]),
           xlab = "Odds Ratio",
           title = "Forest Plot de Razão de Chances para IAM dos 4 Modelos"
           )
```



onde Modelo 4 na última linha é a variável história médica pregressa.