Análise de Evasão de Alunos da Ciência da Computação

Mariana Freitas, Aline Pires e Heitor Quartezani

Introdução

Ao longo do relatório, algumas partes de código em R serão mostradas. No entanto, para obter o código completo e compreender o tratamento dos dados, o repositório em *github* está disponível em https://github.com/marianacfreitas/Analise_evasao_CT.git.

Análise descritiva

Análise de sobrevivência e modelagem

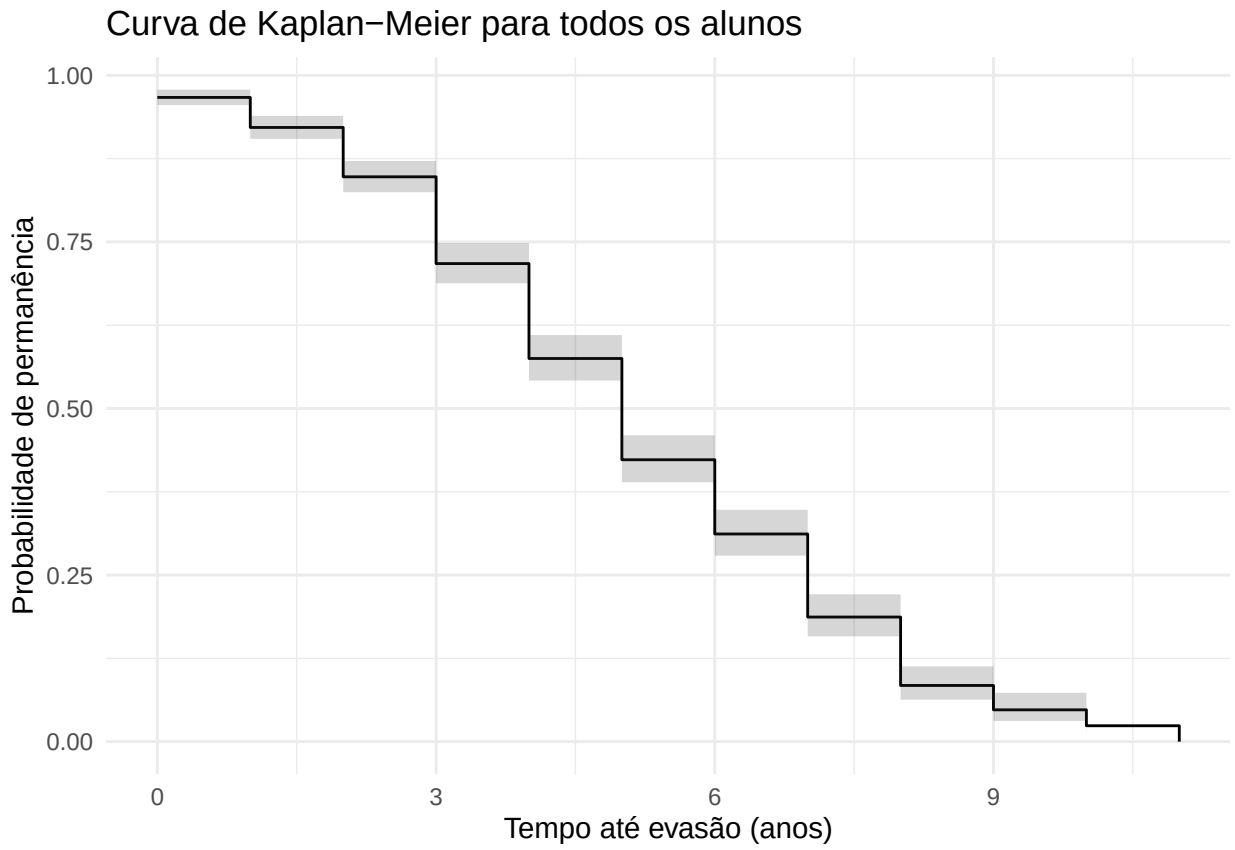
Análise das Curvas de Kaplan-Meier

A análise de sobrevivência é uma área da estatística voltada para o estudo do tempo até a ocorrência de um evento de interesse. No caso em análise, esse evento corresponde à evasão estudantil - o momento em que um aluno deixa o curso, seja devido a formatura ou antes da conclusão. Uma das ferramentas mais utilizadas nesse tipo de análise são as curvas de Kaplan-Meier, que estimam a função de sobrevivência de forma não paramétrica, levando em conta a censura nos dados, como nos casos em que o aluno ainda está matriculado ao final do período de observação. Essas curvas representam, ao longo do tempo, a proporção acumulada de alunos que permaneceram no curso, permitindo comparações entre diferentes grupos. No âmbito de interpretação, a cada ponto da curva, a altura indica a probabilidade de um aluno ainda estar vinculado ao curso até aquele período. Quedas mais acentuadas na curva representam momentos de maior evasão, enquanto curvas mais altas e estáveis indicam maior retenção. Por isso, as curvas de Kaplan-Meier são essenciais para entender padrões de permanência e abandono.

A seguir são apresentadas três curvas de Kaplan-Meier: a primeira levando em consideração todos os alunos, independente da forma de evasão; a segunda composta apenas por alunos que tiveram evasão por formatura ou que ainda estão no curso; e a terceira constituída por aqueles que abandonaram antes de concluir o curso ou ainda estão cursando.

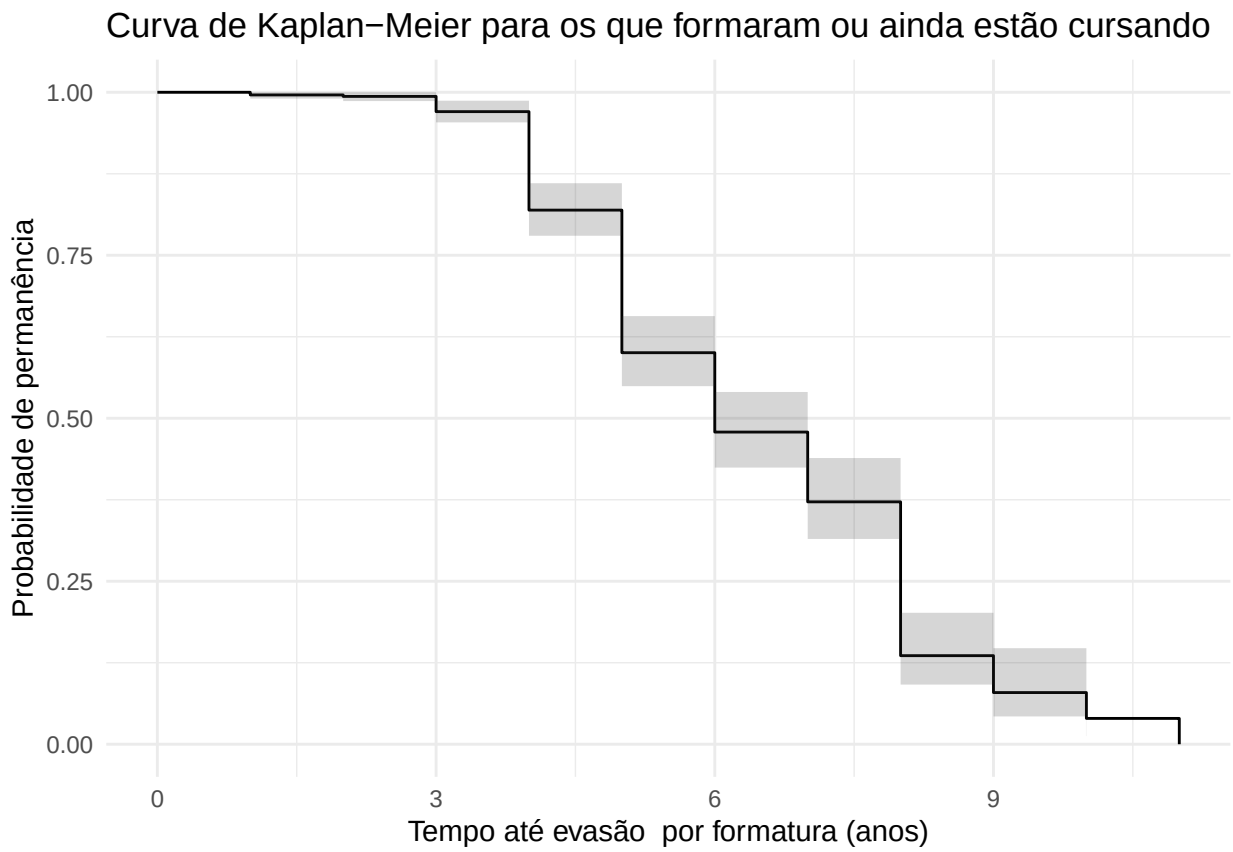
```
# Carregamento de pacotes
library(survival)
library(ggsurvfit)
library(readr)
library(dplyr)
library(survminer)
library(ggplot2)
library(ggsci)
library(survivalsvm)

# Carregando dados
dados <- read_csv("data/dados_tratados.csv") |>
```

```r
  select(-c(`...1`, id_aluno))

# Criar objeto de sobrevivência
fit <- survfit(Surv(anos_ate_evasao, status) ~ 1, data = dados)

# Plotar curva de Kaplan-Meier
ggsurvplot_geral <- ggsurvfit(fit) +
  labs(
    title = "Curva de Kaplan-Meier para todos os alunos",
    x = "Tempo até evasão (anos)",
    y = "Probabilidade de permanência"  ) +
  add_confidence_interval() +
  theme_minimal()

print(ggsurvplot_geral)
```



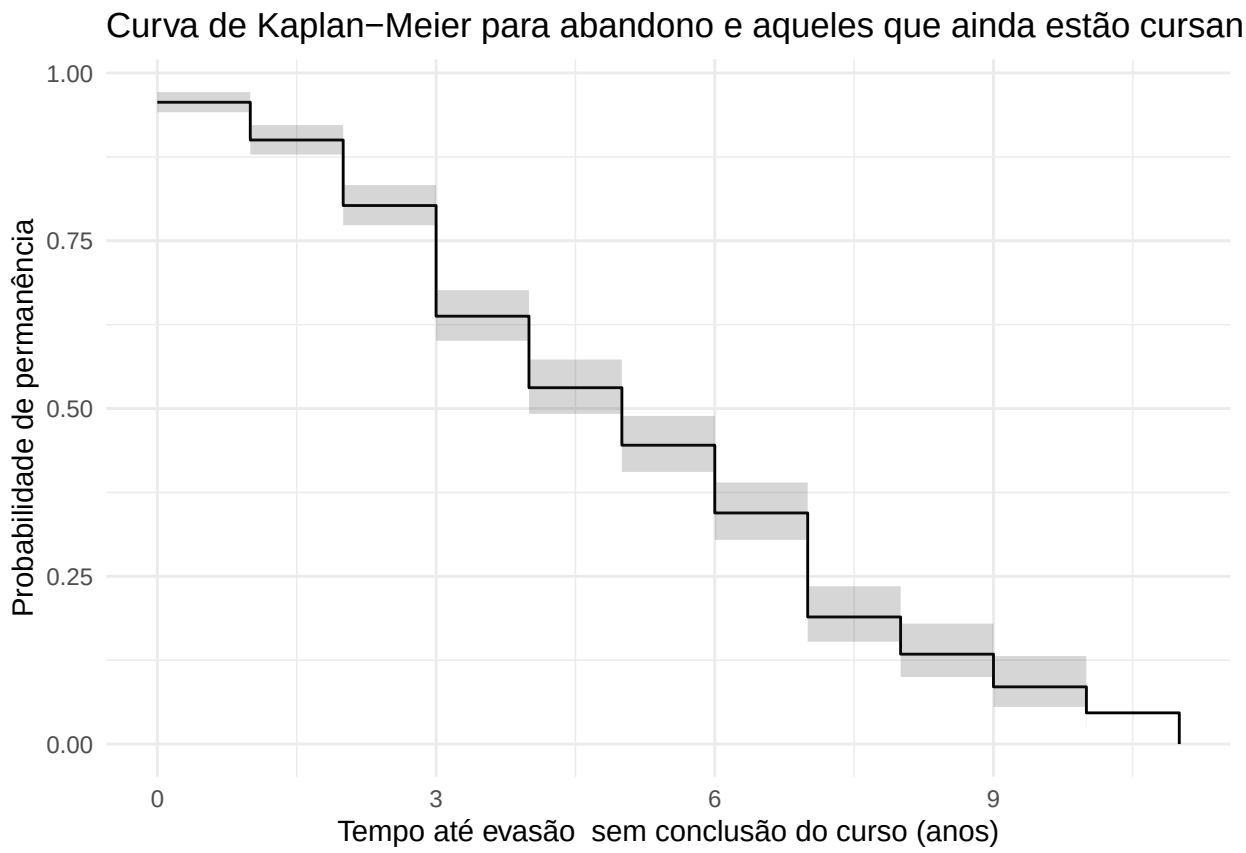

```r
# Apenas aqueles que formaram ou estçao cursando
dados_formados <- dados |>
  filter(forma_evasao == "'Formado'" | forma_evasao == "'Sem evasão'")

# Criar objeto de sobrevivência
fit_formados <- survfit(Surv(anos_ate_evasao, status) ~ 1, data = dados_formados)

# Plotar curva de Kaplan-Meier
ggsurvplot_formados <- ggsurvfit(fit_formados) +
```

```
  labs(
    title = "Curva de Kaplan-Meier para os que formaram ou ainda estão cursando",
    x = "Tempo até evasão  por formatura (anos)",
    y = "Probabilidade de permanência"  ) +
  add_confidence_interval() +
  theme_minimal()

print(ggsurvplot_formados)
```




```
dados_quit <- dados |>
  filter(forma_evasao != "'Formado'")


# Criar objeto de sobrevivência
fit_quit <- survfit(Surv(anos_ate_evasao, status) ~ 1, data = dados_quit)

# Plotar curva de Kaplan-Meier
ggsurvplot_quit <- ggsurvfit(fit_quit) +
  labs(
    title = "Curva de Kaplan-Meier para abandono e aqueles que ainda estão cursando",
    x = "Tempo até evasão  sem conclusão do curso (anos)",
    y = "Probabilidade de permanência"  ) +
  add_confidence_interval() +
  theme_minimal()
```

```r
print(ggsurvplot_quit)
```



As curvas de Kaplan-Meier por estrato de variáveis não foram incluídas nesse relatório para não torná-lo tão extenso, mas podem ser visualizadas no link: https://marianacfreitas.shinyapps.io/km_evasao/. Foram consideradas apenas as variáveis referntes ao início do curso e ingresso do aluno, por no geral serem as mais importantes para determinar evasão.

Modelagem considerando dados censurados

Support Vector Machine (SVM) é um algoritmo de aprendizado de máquina amplamente utilizado para tarefas de classificação e regressão. Sua principal ideia é encontrar um hiperplano ótimo que separe os dados em diferentes classes ou que se ajuste ao padrão de uma variável resposta, maximizando a margem entre as observações e esse hiperplano. Para lidar com dados que não são linearmente separáveis no espaço original, o SVM utiliza o chamado *kernel trick*, que transforma os dados para um espaço de maior dimensão onde essa separação se torna possível. Essa flexibilidade permite aplicar o SVM em diversos problemas com diferentes formas de relacionamentos entre variáveis.

Na área de análise de sobrevivência, o SVM pode ser adaptado para lidar com dados censurados, ou seja, situações em que o evento de interesse (evasão de um estudante) ainda não ocorreu até o final do acompanhamento. Uma das versões dessa adaptação é o *Survival SVM*, que busca encontrar uma função de predição que minimize o erro entre o tempo observado e o valor predito, penalizando predições inválidas em relação ao tempo censurado. Isso é feito, por exemplo, com o uso de uma margem de tolerância que define um intervalo aceitável de erro, considerando a censura à direita ao penalizar predições que estejam abaixo do tempo censurado.

Para avaliar o desempenho preditivo de modelos em análise de sobrevivência, uma métrica amplamente utilizada é o C-índex. Ele avalia o grau de concordância entre os tempos preditos e os tempos observados de eventos, levando em conta a censura. Um C-índex igual a 1 indica predições perfeitas.

Aqui foi utilizado o *Survival SVM* com *kernel aditivo*, já que outros *kernels* testados, como o gaussiano e linear, obtiveram um baixo C-índex, indicando que não são ideais para descrever as relações entre as vaiáveis e o tempo de permanência nesse caso. Para o modelo, foram consideradas apenas variáveis a respeito da forma de evasão, informações sobre os alunos e aquelas referentes a matérias iniciais, pois o uso de todas as variáveis não era viável para treinamento do modelo. Para possibilitar o treinamento do modelo, também foi feito o tratamento de valores ausentes, removendo variáveis com maioria dos valores ausentes e imputando valores em outras.

```r
set.seed(2402)

# Criar índice para divisão
indices <- sample(1:nrow(dados), size = floor(0.8 * nrow(dados)), replace = FALSE)

# Tratamento de valores ausentes
imputar_dados <- function(dados) {
  #Remover variáveis com >50% NAs
  limiar_remocao <- 0.5
  vars_para_remover <- sapply(dados, function(x) {
    sum(is.na(x)) / length(x) > limiar_remocao
  }) %>% which() %>% names()

  if (length(vars_para_remover) > 0) {
    warning(paste("Variáveis removidas (mais de 50% NAs):",
                  paste(vars_para_remover, collapse = ", ")))
    dados <- dados %>% select(-all_of(vars_para_remover))
  }

  # Passo 2: Imputação conforme regras específicas
  dados_imputados <- dados |>
    mutate(across(
      .cols = everything(),
      .fns = ~ {
        nome_var <- cur_column()

        if (is.numeric(.x)) {
          # Variáveis que começam com "numero_tentativas": quando NA, o aluno não fez a matéria
          if (startsWith(nome_var, "numero_tentativas")) {
            ifelse(is.na(.x), 0, .x)
          }
          # Variáveis que começam com "media": imputar mediana
          else if (startsWith(nome_var, "media")) {
            ifelse(is.na(.x), median(.x, na.rm = TRUE), .x)
          }
          # Demais variáveis numéricas: imputar mediana
          else {
            ifelse(is.na(.x), median(.x, na.rm = TRUE), .x)
          }
        }
        else {
          # Variáveis que começam com "fez": imputar moda
```

```
          if (startsWith(nome_var, "fez")) {
            moda <- names(sort(table(.x), decreasing = TRUE))[1]
            ifelse(is.na(.x), moda, .x)
          }
          # Demais variáveis categóricas: imputar moda
          else {
            moda <- names(sort(table(.x), decreasing = TRUE))[1]
            ifelse(is.na(.x), moda, .x)
          }
        }
      }
    ))

  return(dados_imputados)
}

# Na variável cra, trocar "," por ".", para não ser confundido com character
dados$cra <- as.numeric(gsub(",", ".", dados$cra))


# Aplicando a função
dados <- imputar_dados(dados) |>
  select(
    forma_evasao,
    cra,
    forma_ingresso,
    cotista,
    status,
    anos_ate_evasao,
    fez_calculo_i,
    fez_calculo_ii,
    fez_programacao_i,
    fez_introducao_a_ciencia_da_computacao,
    fez_aspectos_teoricos_da_computacao,
    fez_elementos_de_logica_digital,
    fez_programacao_ii,
    numero_tentativas_calculo_i,
    numero_tentativas_calculo_ii,
    numero_tentativas_programacao_i,
    numero_tentativas_introducao_a_ciencia_da_computacao,
    numero_tentativas_aspectos_teoricos_da_computacao,
    numero_tentativas_elementos_de_logica_digital,
    numero_tentativas_programacao_ii,
    media_final_calculo_i_1,
    media_final_programacao_i_1,
    media_final_elementos_de_logica_digital_1,
    media_final_programacao_ii_1
  )

# Dividir os dados
treino <- dados[indices, ]
teste <- dados[-indices, ]
```

```r
formula_dados <- as.formula(
  paste0("Surv(anos_ate_evasao, status) ~ ",
         paste(setdiff(names(treino), c("anos_ate_evasao", "status")), collapse = " + "))
)

svm_add <- survivalsvm(formula_dados,
                       data = treino,
                       type = "regression",
                       gamma.mu = 1,
                       kernel = "add_kernel")

pred_svm_add <- predict(svm_add, newdata = teste)$predicted

cindex_svm_add <- concordance(Surv(teste$anos_ate_evasao, teste$status) ~
                                as.vector(pred_svm_add))$concordance
```

Ao final, o C-índex obtido para o modelo foi de aproximadamente 0.73, indicando bom poder preditivo dos tempos de permanência dos alunos no curso.

Conclusão