# ANÁLISES TESE ANA - 2025 ------------------------------------------------

## PREPARANDO O AMBIENTE R -------------------------------------------------  
   
#Limpando Objetos  
rm(list = ls(all=TRUE))  
  
#Definindo opção de codificação  
aviso <- getOption("warn")  
options(warn=-1)  
options(encoding= "latin1")  
options(warn=aviso)  
rm(aviso)  
  
#Opção exibição números sem exponencial  
aviso <- getOption("warn")  
options(warn=-1)  
options(scipen = 999)  
options(warn=aviso)  
rm(aviso)  
  
#Definindo opção de repositóriorio para instalação pacotes necessários  
aviso <- getOption("warn")  
options(warn=-1)  
options(repos=structure(c(cran="https://cran.r-project.org/")))  
rm(aviso)  
  
#Abrindo Pacotes  
library(tidyverse)  
library(survey)  
library(srvyr)  
library(PNADcIBGE)  
library(dplyr)  
  
## CARREGANDO AS BASES DE DADOS --------------------------------------------  
  
# Definindo o diretório onde estão os arquivos e onde serão salvos os filtrados  
setwd("C:/Users/Ana Oliveira/Desktop/pnadc 2023") #Altere para o nome do seu diretório  
dir\_filtrados <- "filtrados2"  
if (!dir.exists(dir\_filtrados)) {  
 dir.create(dir\_filtrados)  
}  
  
# Vetor com as variáveis de interesse  
variaveis\_selecionadas <- c(  
 "Ano", "Trimestre", "UF", "UPA", "Estrato", "V1008", "V1014", "V1016",   
 "V4017", "V1023", "V1028", "V2001", "V2003", "V2005", "V2007", "V2008",   
 "V20081", "V20082", "V2009", "V2010", "VD3005", "V4001", "V4009", "V4010",   
 "V4012", "V4013", "V4025", "V4026", "V4029", "V4032", "V4039", "V4041", "V4043",   
 "V4044", "V4048", "V40501", "V405012", "V4056", "V4057", "V4039C", "V4040",   
 "VD2003", "VD2004", "VD3004", "VD3005", "VD4009", "VD4002", "VD4008", "VD4011",  
 "VD4015", "V40331", "V403311", "VD4016", "VD4017", "VD4018", "VD4019",   
 "VD4020"  
)  
  
# Definindo os anos e trimestres para processar  
anos <- 2016:2024  
trimestres <- c("01", "02", "03", "04")  
  
# Processando cada trimestre individualmente  
for (ano in anos) {  
 for (trim in trimestres) {  
   
 # Montando o nome do arquivo – ajuste conforme o padrão dos seus arquivos  
 nome\_arquivo <- paste0("PNADC\_", trim, ano, ".txt")  
   
 # Lendo o arquivo usando a função do PNADcIBGE  
 dados\_trim <- read\_pnadc(  
 microdata = nome\_arquivo,  
 input\_txt = "input\_PNADC\_trimestral.txt",  
 vars = variaveis\_selecionadas  
 )  
   
 # Filtrando ocupados: apenas quem trabalhou ou estagiou pelo menos 1 hora  
 # - V4001 == "1" (trabalhou ou estagiou pelo menos 1 hora)  
 dados\_filtrados <- dados\_trim %>%  
 filter(V4001 == "1")  
   
 # Salvando os dados filtrados para este trimestre em um arquivo RDS  
 nome\_saida <- paste0(dir\_filtrados, "/pnadc\_filtrado\_", trim, "\_", ano, ".rds")  
 saveRDS(dados\_filtrados, file = nome\_saida)  
   
 # Limpando memória para o próximo ciclo  
 rm(dados\_trim, dados\_filtrados)  
 gc()  
 }  
}  
  
# Após o processamento, carregue os arquivos filtrados e una-os  
arquivos\_filtrados <- list.files(dir\_filtrados, pattern = "\\.rds$", full.names = TRUE)  
lista\_filtrados <- lapply(arquivos\_filtrados, readRDS)  
pnad\_final <- do.call(rbind, lista\_filtrados)  
  
# Adicionando o dicionário (rotulação das variáveis)  
pnad\_final <- pnadc\_labeller(  
 data\_pnadc = pnad\_final,   
 dictionary.file = "dicionario\_PNADC\_microdados\_trimestral.xls"  
)  
  
# Adicionando o deflator  
pnad\_final <- pnadc\_deflator(  
 data\_pnadc = pnad\_final,   
 deflator.file = "deflator\_PNADC\_2024\_trimestral\_101112.xls"  
)  
  
#Salvando base. Caso vc não tenha memória no pc suficiente para esta etapa, use este arquivo: pnad\_final\_21\_06\_25.rds  
pnad\_final\_21\_06\_25 <- pnad\_final  
saveRDS(pnad\_final\_21\_06\_25, file = "pnad\_final\_21\_06\_25.rds")  
  
# Carregando o arquivo .rds  
pnad\_final\_21\_06\_25 <- readRDS("pnad\_final\_21\_06\_25.rds")  
  
# Atribuindo à variável 'pnad\_final'  
pnad\_final <- pnad\_final\_21\_06\_25  
rm(pnad\_final\_21\_06\_25)  
  
# Filtrando por tipo de emprego, remuneração e idade  
pnad\_filtrada <- pnad\_final %>%  
 filter(V2009 >= 14, V4012 == "Empregado do setor privado", V40331 == "Em dinheiro")  
  
pnad\_final <- pnad\_filtrada  
rm(pnad\_filtrada)  
  
  
## VERIFICANDO E FILTRANDO A BASE DE DADOS ---------------------------------------------  
  
# Quantos indivíduos tenho na base?  
library(openxlsx)  
# cria o resumo  
resumo <- pnad\_final %>%   
 summarise(  
 n\_obs = n()  
 )  
  
# Criando o workbook   
wb <- createWorkbook()  
addWorksheet(wb, "Resumo PNADC")  
writeData(wb, "Resumo PNADC", resumo)  
  
# Salvando o arquivo  
saveWorkbook(wb, "resumo\_pnad\_final.xlsx", overwrite = TRUE)  
  
# Contando linhas por entrevista  
# Criando o tibble de linhas por entrevista  
tabela\_linhas <- pnad\_final %>%  
 group\_by(V1016) %>%  
 summarise(n\_linhas = n()) %>%  
 arrange(V1016)  
  
# Salvando em um arquivo .xlsx  
write\_xlsx(tabela\_linhas, path = "linhas\_por\_entrevista.xlsx")  
  
glimpse(pnad\_final)  
  
  
# criando variável salário EFETIVO REAL - DEFLACIONADO  
pnad\_final <- pnad\_final %>%  
 mutate(VD4016\_Real = VD4016 \* Habitual)  
  
# Verificando salários por entrevistas  
stats\_salario <- pnad\_final %>%  
 group\_by(V1016) %>%  
 summarise(  
 media\_salario = mean(VD4016\_Real, na.rm = TRUE),  
 sd\_salario = sd(VD4016\_Real, na.rm = TRUE),  
 n\_linhas = n()  
 ) %>%  
 arrange(V1016)  
  
stats\_salario  
  
# Gravando a tabela em um arquivo .xlsx  
write\_xlsx(stats\_salario, path = "salarios\_por\_entrevistas.xlsx")  
  
# Identificando indivíduos com pelo menos 2 entrevistas  
  
# Chave individuos  
# Excluindo indivíduos com valores NA nos dados de nascimento (V2008, V20081, V20082)  
pnad\_final <- pnad\_final %>%   
 filter(!is.na(V2008) & !is.na(V20081) & !is.na(V20082))  
  
# Criando a chave de identificação para cada indivíduo  
pnad\_final <- pnad\_final %>%   
 mutate(chave\_individuo = paste(UPA, V1008, V1014, V2003, V2008, V20081, V20082, sep = "\_"))  
  
  
# Verificando salários dos individuos que responderam a 2 ou mais entrevistas  
ind2mais <- pnad\_final %>%  
 count(chave\_individuo) %>%  
 filter(n >= 2) %>%  
 pull(chave\_individuo)  
  
# Logaritimizando variável - Considerando horas mensais  
pnad\_final <- pnad\_final %>%  
 mutate(Salario\_Hora = if\_else(!is.na(VD4016\_Real) & !is.na(V4039), VD4016\_Real / (V4039 \* 4.35), NA\_real\_), # Considera horas mensais  
 VD4016\_log\_hora = if\_else(!is.na(Salario\_Hora), log(Salario\_Hora), NA\_real\_)) # Calcula log apenas se válido  
  
head(pnad\_final$V4039)  
head(pnad\_final$VD4016\_log\_hora)  
  
# Extraindo salário da 1ª e da última entrevista de cada um  
sal\_pair\_log <- pnad\_final %>%  
 filter(chave\_individuo %in% ind2mais) %>%  
 arrange(chave\_individuo, V1016) %>%  
 group\_by(chave\_individuo) %>%  
 summarise(  
 log\_ini = first(VD4016\_log\_hora), # log na 1ª entrevista  
 log\_fim = last(VD4016\_log\_hora) # log na última entrevista  
 ) %>%  
 ungroup()  
  
# Criando o vetor de diferenças  
dif\_log <- sal\_pair\_log$log\_fim - sal\_pair\_log$log\_ini  
  
# Write\_xlsx recebe um data.frame ou tibble e grava em um arquivo .xlsx  
write\_xlsx(sal\_pair\_log, path = "sal\_pair\_log.xlsx")  
  
# Verificando normalidade das diferenças - nortest para amostras grandes + de 5000 obs  
# install.packages("nortest") # só na primeira vez  
library(nortest)  
  
# Executando o teste e guardando num objeto  
ad\_res <- ad.test(dif\_log)  
  
# Extraindo os elementos que interessam num data.frame  
df\_ad <- data.frame(  
 Estatística = as.numeric(ad\_res$statistic),  
 p\_value = ad\_res$p.value,  
 row.names = NULL  
)  
  
# Salvando em Excel  
write\_xlsx(df\_ad, path = "resultado\_ad\_test.xlsx")  
  
# Gráficos normalidade  
qqnorm(dif\_log); qqline(dif\_log)  
hist(dif\_log, breaks = 50)  
  
# Salvando apenas o QQ-plot em PDF - qq plot nao ta saindo no PDF!  
# Abrindo o PDF  
pdf("qqplot\_dif\_log.pdf", width = 6, height = 6)  
  
# Gerando o QQ-plot  
qqnorm(dif\_log, main = "QQ-Plot de dif\_log")  
qqline(dif\_log)  
  
# Fechando o PDF  
dev.off()  
  
# Salvando apenas o histograma em PDF  
pdf("histograma\_dif\_log.pdf", width = 6, height = 6)  
hist(  
 dif\_log,  
 breaks = 50,  
 main = "Histograma de dif\_log",  
 xlab = "dif\_log",  
 ylab = "Frequência"  
)  
dev.off()  
  
# Teste t pareado nos logs  
t\_log <- t.test(sal\_pair\_log$log\_fim,  
 sal\_pair\_log$log\_ini,  
 paired = TRUE)  
  
t\_log  
  
# E variação média aproximada em %:  
(exp(t\_log$estimate) - 1) \* 100  
  
# Filtrando dados apenas para 1ª entrevista  
pnad\_entrev1 <- pnad\_final %>%  
 filter(V1016 == 1)  
  
# Emprego secundário  
  
# vetor com códigos do turismo  
turismo\_codes <- c(  
 "79000","55000","77020","92000","93011","93020",  
 "90000","91000","56011","56012","56020","51000",  
 "50000","49010","49030","49090"  
)  
  
# Montando a tabela de setor principal  
tabela\_setor\_principal <- pnad\_entrev1 %>%  
 mutate(  
 setor\_principal = if\_else(V4013 %in% turismo\_codes, "Turismo", "Outros")  
 ) %>%  
 count(setor\_principal, name = "n\_individuos") %>%  
 arrange(desc(n\_individuos))  
  
tabela\_setor\_principal  
  
tabela\_secundario\_turismo\_por\_setor <- pnad\_entrev1 %>%  
 mutate(setor\_principal = if\_else(V4013 %in% turismo\_codes, "Turismo", "Outros")) %>%  
 group\_by(setor\_principal) %>%  
 summarise(  
 n\_individuos = n(),  
 # quem tem ≥2 empregos  
 n\_secundario = sum(V4009 %in% c("Dois", "Três ou mais"), na.rm = TRUE),  
 # desses, quantos têm 2º emprego NO TURISMO  
 n\_secundario\_tur = sum(V4009 %in% c("Dois", "Três ou mais") &  
 V4044 %in% turismo\_codes,  
 na.rm = TRUE),  
 # percentuais  
 perc\_secundario = n\_secundario / n\_individuos \* 100,  
 perc\_secundario\_tur= n\_secundario\_tur/ n\_individuos \* 100  
 ) %>%  
 arrange(setor\_principal)  
  
tabela\_secundario\_turismo\_por\_setor  
  
# Lista com as duas tabelas nomeadas  
minhas\_tabelas <- list(  
 Setor\_Principal = tabela\_setor\_principal,  
 Secundario\_Por\_Setor = tabela\_secundario\_turismo\_por\_setor  
)  
  
# Gravando em um arquivo .xlsx na sua pasta de trabalho  
library(writexl)  
write\_xlsx(minhas\_tabelas, path = "resultados\_tabelas.xlsx")  
  
#excluindo variáveis data nascimento e emprego secundário  
pnad\_filtrada<- pnad\_entrev1 %>%   
 select(-V2008, -V20081, -V20082, -V4041, -V4043, -V4044,   
 -V4048, -V40501, -V405012, -V4056, -V4057)  
  
## VERIFICANDO E ELIMINANDO OUTLIERS ---------------------------------------  
  
# Gráficos de densidade e linha quantis salário hora  
# Abre o dispositivo PDF  
pdf("densidade\_salario\_hora.pdf", width = 7, height = 5)  
  
plot(density(pnad\_filtrada$Salario\_Hora, na.rm = TRUE),  
 main = "Curva de Densidade - Salário por Hora",  
 xlab = "Salário por hora (R$)",  
 col = "blue",  
 lwd = 2)  
abline(v = quantile(pnad\_filtrada$Salario\_Hora, probs = c(0.01, 0.99), na.rm = TRUE),  
 col = "red", lty = 2)  
  
# Fechando o PDF e gravando o arquivo  
dev.off()  
  
# Para visualizar melhor a cauda a esquerda calcular a densidade  
densidade <- density(pnad\_filtrada$Salario\_Hora, na.rm = TRUE)  
  
# Obtendo a moda (ponto de maior densidade)  
moda\_salario <- densidade$x[which.max(densidade$y)]  
  
# Abrindo o PDF  
pdf("densidade\_ate\_moda.pdf", width = 7, height = 5)  
  
# Plotando a curva de densidade até a moda  
plot(densidade,  
 main = "Curva de Densidade - Salário por Hora (até a Moda)",  
 xlab = "Salário por hora (R$)",  
 col = "blue",  
 lwd = 2,  
 xlim = c(0, moda\_salario)) # Limita até a moda  
  
# Adicionando as linhas verticais dos quantis  
abline(v = quantile(pnad\_filtrada$Salario\_Hora, probs = c(0.01, 0.99), na.rm = TRUE),  
 col = "red", lty = 2)  
  
# Adicionando uma linha na moda  
abline(v = moda\_salario, col = "darkgreen", lty = 3)  
  
# Fechando o PDF  
dev.off()  
  
# Calculando os quantis de 1% e 99% para a variável Salario\_Hora  
quantis <- quantile(pnad\_filtrada$Salario\_Hora, probs = c(0.01, 0.99), na.rm = TRUE)  
limite\_inferior\_salhora <- quantis[1]  
  
limite\_superior\_salhora <- quantis[2]  
  
# Contando o número de observações abaixo do quantil de 1%  
num\_inferior\_salhora <- sum(pnad\_filtrada$Salario\_Hora < limite\_inferior\_salhora, na.rm = TRUE)  
  
# Contando o número de observações acima do quantil de 99%  
num\_superior\_salhora <- sum(pnad\_filtrada$Salario\_Hora > limite\_superior\_salhora, na.rm = TRUE)  
  
# Exibindo os resultados  
cat("Observações abaixo do 1%:", num\_inferior\_salhora, "\n")  
cat("Observações acima do 99%:", num\_superior\_salhora, "\n")  
cat("Total de outliers a serem removidos:", num\_inferior\_salhora + num\_superior\_salhora, "\n")  
  
# Montando o data.frame com os resultados  
df\_outliers <- data.frame(  
 métricas = c("ABAIXO\_1%", "ACIMA\_99%", "TOTAL\_OUTLIERS"),  
 quantidade = c(num\_inferior\_salhora,  
 num\_superior\_salhora,  
 num\_inferior\_salhora + num\_superior\_salhora)  
)  
  
# Exportando para um arquivo .xlsx  
write\_xlsx(df\_outliers, path = "outliers\_salario\_hora.xlsx")  
  
# Graficos densidade e linha quantis hora semanal  
  
# Abrindo o dispositivo PDF (ou troque para png() conforme desejar)  
pdf("densidade\_horas\_trabalhadas.pdf", width = 7, height = 5)  
  
plot(density(pnad\_filtrada$V4039, na.rm = TRUE),  
 main = "Curva de Densidade - Horas Trabalhadas",  
 xlab = "Hora Semanal",  
 col = "blue",  
 lwd = 2)  
abline(v = quantile(pnad\_filtrada$V4039, probs = c(0.01, 0.99), na.rm = TRUE),  
 col = "red", lty = 2)  
  
# Fechando o dispositivo, gravando o PDF  
dev.off()  
  
# Calculando os quantis de 1% e 99% para a variável Salario\_Hora  
quantis <- quantile(pnad\_filtrada$V4039, probs = c(0.01, 0.99), na.rm = TRUE)  
limite\_inferior\_V4039 <- quantis[1]  
limite\_superior\_V4039 <- quantis[2]  
  
# Contando o número de observações abaixo do quantil de 1%  
num\_inferior\_V4039 <- sum(pnad\_filtrada$V4039 < limite\_inferior\_V4039, na.rm = TRUE)  
  
# Contando o número de observações acima do quantil de 99%  
num\_superior\_V4039 <- sum(pnad\_filtrada$V4039 > limite\_superior\_V4039, na.rm = TRUE)  
  
# Exibindo os resultados  
cat("Observações abaixo do 1%:", num\_inferior\_V4039, "\n")  
cat("Observações acima do 99%:", num\_superior\_V4039, "\n")  
cat("Total de outliers a serem removidos:", num\_inferior\_V4039 + num\_superior\_V4039, "\n")  
  
# Montando o data.frame com os resultados de horas trabalhadas  
df\_outliers\_horas <- data.frame(  
 métricas = c("ABAIXO\_1%", "ACIMA\_99%", "TOTAL\_OUTLIERS"),  
 quantidade = c(  
 num\_inferior\_V4039,  
 num\_superior\_V4039,  
 num\_inferior\_V4039 + num\_superior\_V4039  
 )  
)  
  
# Exportando para um arquivo .xlsx  
write\_xlsx(df\_outliers\_horas, path = "outliers\_horas\_trabalhadas.xlsx")  
  
# Excluindo outliers  
  
# Calculando os limites para Salario\_Hora  
quantis\_sh <- quantile(pnad\_filtrada$Salario\_Hora, probs = c(0.01, 0.99), na.rm = TRUE)  
limite\_inferior\_sh <- quantis\_sh[1]  
limite\_superior\_sh <- quantis\_sh[2]  
  
# Calculando os limites para V4039 (Horas Semanais)  
quantis\_v4039 <- quantile(pnad\_filtrada$V4039, probs = c(0.01, 0.99), na.rm = TRUE)  
limite\_inferior\_v4039 <- quantis\_v4039[1]  
limite\_superior\_v4039 <- quantis\_v4039[2]  
  
# Filtrando os dados removendo outliers de ambas as variáveis   
pnad\_filtrada\_sem\_outliers <- pnad\_filtrada[  
 pnad\_filtrada$Salario\_Hora >= limite\_inferior\_sh &  
 pnad\_filtrada$Salario\_Hora <= limite\_superior\_sh &  
 pnad\_filtrada$V4039 >= limite\_inferior\_v4039 &  
 pnad\_filtrada$V4039 <= limite\_superior\_v4039, ]  
  
# A amostra restante conta com pessoas que trabalham entre   
# 10 e 65 horas e o salário hora está entre R$1,99 - R$80,00  
cat("Número de observações originais:", nrow(pnad\_filtrada), "\n")  
cat("Número de observações após remoção:", nrow(pnad\_filtrada\_sem\_outliers), "\n")  
  
resumo\_outliers <- data.frame(  
 etapa = c("Original", "Sem\_Outliers"),  
 n\_observacoes = c(nrow(pnad\_filtrada),  
 nrow(pnad\_filtrada\_sem\_outliers))  
)  
  
write\_xlsx(resumo\_outliers, "resumo\_outliers.xlsx")  
  
summary(pnad\_filtrada\_sem\_outliers %>% select(Salario\_Hora))  
summary(pnad\_filtrada\_sem\_outliers %>% select(V4039))  
  
  
# Extraindo os summaries como vetores  
sum\_sh <- summary(pnad\_filtrada\_sem\_outliers$Salario\_Hora)  
sum\_horas<- summary(pnad\_filtrada\_sem\_outliers$V4039)  
  
# Convertendo cada summary em data.frame “longo”  
df\_sum\_sh <- data.frame(  
 Estatística = names(sum\_sh),  
 Salario\_Hora = as.numeric(sum\_sh),  
 row.names = NULL,  
 stringsAsFactors = FALSE  
)  
  
df\_sum\_horas <- data.frame(  
 Estatística = names(sum\_horas),  
 Horas\_Semanais = as.numeric(sum\_horas),  
 row.names = NULL,  
 stringsAsFactors = FALSE  
)  
  
# Colocando ambos num lista  
minhas\_summaries <- list(  
 Resumo\_Salario\_Hora = df\_sum\_sh,  
 Resumo\_Horas = df\_sum\_horas  
)  
  
# Escrevendo a planilha com duas abas  
write\_xlsx(minhas\_summaries, path = "summaries\_pnad\_sem\_outliers.xlsx")  
  
  
# Coincidindo nome arquivo para fazer transformações das variáveis  
  
pnad\_c16241Entrevista <- pnad\_filtrada\_sem\_outliers  
  
# Verificando quantos trabalham no turismo  
resumo\_setor <- pnad\_c16241Entrevista %>%  
 mutate(setor\_principal = if\_else(V4013 %in% turismo\_codes, "Turismo", "Outros")) %>%  
 distinct(chave\_individuo, setor\_principal) %>% # garante 1 linha por pessoa × setor  
 count(setor\_principal, name = "n\_individuos") %>%  
 arrange(desc(n\_individuos))  
  
resumo\_setor  
  
  
rm(pnad\_filtrada\_sem\_outliers)  
rm(pnad\_filtrada)  
rm(pnad\_c1624\_1Entrevista\_1ouMaisEmp2)  
rm(pnad\_final, pnad\_entrev1)  
  
## TRANSFORMANDO VARIÁVEIS -------------------------------------------------  
  
# Recodificando Escolaridade categórica - 4 categorias  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(VD3004\_nivel\_Escol = case\_when(  
 VD3004 %in% c("Sem instrução e menos de 1 ano de estudo", "Fundamental incompleto ou equivalente") ~ "Fundamental\_Incompleto",  
 VD3004 %in% c("Fundamental completo ou equivalente", "Médio incompleto ou equivalente") ~ "Fundamental\_Completo",  
 VD3004 %in% c("Médio completo ou equivalente", "Superior incompleto ou equivalente") ~ "Médio\_Completo",  
 VD3004 %in% c("Superior completo") ~ "Superior\_Completo",  
 TRUE ~ NA\_character\_ # Ignora "Sem instrução" e outros casos  
 ))  
  
head(pnad\_c16241Entrevista$VD3004\_nivel\_Escol, n = 100)  
table(pnad\_c16241Entrevista$VD3004\_nivel\_Escol)  
  
  
# Recodificando Escolaridade categórica - 2 categorias - Médio x não médio  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(VD3004\_nivel\_Escol2 = case\_when(  
 VD3004 %in% c("Sem instrução e menos de 1 ano de estudo",   
 "Fundamental incompleto ou equivalente",   
 "Fundamental completo ou equivalente",   
 "Médio incompleto ou equivalente") ~ "Sem\_Ensino\_Medio",  
   
 VD3004 %in% c("Médio completo ou equivalente",   
 "Superior incompleto ou equivalente",   
 "Superior completo") ~ "Com\_Ensino\_Medio",  
   
 TRUE ~ NA\_character\_  
 ))  
  
  
head(pnad\_c16241Entrevista$VD3004\_nivel\_Escol2, n = 100)  
table(pnad\_c16241Entrevista$VD3004\_nivel\_Escol2)  
  
# Recodificando Escolaridade categórica - 2 categorias - Superior x nao Superior  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(VD3004\_nivel\_Escol3 = case\_when(  
 VD3004 %in% c(  
 "Sem instrução e menos de 1 ano de estudo",   
 "Fundamental incompleto ou equivalente",   
 "Fundamental completo ou equivalente",   
 "Médio incompleto ou equivalente",   
 "Médio completo ou equivalente",   
 "Superior incompleto ou equivalente"  
 ) ~ "Sem\_Ensino\_Superior",  
   
 VD3004 == "Superior completo" ~ "Com\_Ensino\_Superior",  
   
 TRUE ~ NA\_character\_  
 ))  
  
head(pnad\_c16241Entrevista$VD3004\_nivel\_Escol3, n = 100)  
table(pnad\_c16241Entrevista$VD3004\_nivel\_Escol3)  
  
# Transformando 'VD3005' em uma variável contínua (de 0 a 16 anos)  
# Verificando os valores únicos de VD3005  
unique(pnad\_c16241Entrevista$VD3005)  
# Aplicando o case\_when com os rótulos textuais corretos  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(VD3005\_cont = case\_when(  
 VD3005 == "Sem instrução e menos de 1 ano de estudo" ~ 0,  
 VD3005 == "1 ano de estudo" ~ 1,  
 VD3005 == "2 anos de estudo" ~ 2,  
 VD3005 == "3 anos de estudo" ~ 3,  
 VD3005 == "4 anos de estudo" ~ 4,  
 VD3005 == "5 anos de estudo" ~ 5,  
 VD3005 == "6 anos de estudo" ~ 6,  
 VD3005 == "7 anos de estudo" ~ 7,  
 VD3005 == "8 anos de estudo" ~ 8,  
 VD3005 == "9 anos de estudo" ~ 9,  
 VD3005 == "10 anos de estudo" ~ 10,  
 VD3005 == "11 anos de estudo" ~ 11,  
 VD3005 == "12 anos de estudo" ~ 12,  
 VD3005 == "13 anos de estudo" ~ 13,  
 VD3005 == "14 anos de estudo" ~ 14,  
 VD3005 == "15 anos de estudo" ~ 15,  
 VD3005 == "16 anos ou mais de estudo" ~ 16,  
 TRUE ~ NA\_real\_  
 ))  
  
# Verificando os valores transformados  
head(pnad\_c16241Entrevista$VD3005\_cont, n = 100)  
  
  
# Recodificando Raça  
  
# Variável grupos separados e sem o 'NÃO INFORMADO'  
  
# Variável grupos separados com Não informado para evitar NA  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V2010\_sep = case\_when(  
 V2010 == "Branca" ~ "Branca",  
 V2010 %in% c("Preta") ~ "Preta",  
 V2010 %in% c("Parda") ~ "Parda",  
 V2010 %in% c("Indígena") ~ "Indígena",  
 V2010 %in% c("Amarela") ~ "Amarela",  
 TRUE ~ "Não\_Informado" # Substituir os casos não informados por "Desconhecido"  
 ))  
  
table(pnad\_c16241Entrevista$V2010\_sep)  
  
# Variável BRANCA, PP E -99  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(  
 V2010\_grupo = case\_when(  
 V2010 == "Branca" ~ "Branca",  
 V2010 %in% c("Preta", "Parda") ~ "PP",  
 TRUE ~ NA\_character\_  
 ),  
 V2010\_grupo = replace(V2010\_grupo, is.na(V2010\_grupo), "-99")  
 )  
  
  
table(pnad\_c16241Entrevista$V2010\_grupo)  
  
# Variável BRANCA, PPI E -99  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(  
 V2010\_grupo2 = case\_when(  
 V2010 == "Branca" ~ "Branca",  
 V2010 %in% c("Preta", "Parda", "Indígena", "Amarela") ~ "Nao\_Branca",  
 TRUE ~ "Não\_Informado"))  
  
table(pnad\_c16241Entrevista$V2010\_grupo2)  
  
# Recodificando Posição Domicílio  
unique(pnad\_c16241Entrevista$V2005)  
  
# Recodificando a variável V2005 em "Pessoa responsável", "Cônjuge", "Filhos..." e "Outros"  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V2005 = case\_when(  
 V2005 == 'Pessoa responsável pelo domicílio' ~ 'Pessoa\_responsável',  
 V2005 %in% c('Cônjuge ou companheiro(a) de sexo diferente',   
 'Cônjuge ou companheiro(a) do mesmo sexo') ~ 'Cônjuge',   
 V2005 %in% c('Filho(a) do responsável e do cônjuge') ~ 'Filho\_Resp\_e\_Conj',   
 V2005 %in% c('Filho(a) somente do responsável') ~ 'Filho\_Resp',  
 V2005 %in% c('Enteado(a)') ~ 'Enteado',  
 V2005 %in% c('Genro ou nora',   
 'Pai, mãe, padrasto ou madrasta',   
 'Sogro(a)',   
 'Neto(a)',   
 'Bisneto(a)',   
 'Irmão ou irmã',   
 'Avô ou avó',   
 'Outro parente',   
 'Agregado(a) - Não parente que não compartilha despesas',   
 'Convivente - Não parente que compartilha despesas',   
 'Pensionista',   
 'Empregado(a) doméstico(a)',   
 'Parente do(a) empregado(a) doméstico(a)') ~ 'Outros',  
 TRUE ~ 'Não Informado' # Para qualquer outro valor não previsto  
 ))  
table(pnad\_c16241Entrevista$V2005)  
# Criando a variável que indica se o responsável ou cônjuge tem filhos no domicílio  
# Verificando se há filhos no domicílio  
  
  
# Codificando filhos  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V2005\_filhos = case\_when(  
 V2005 == 'Filho\_Resp\_e\_Conj' ~ 'Filho\_Resp\_e\_Conj',   
 V2005 == 'Filho\_Resp' ~ 'Filho\_Resp',  
 V2005 == 'Enteado' ~ 'Enteado',  
 TRUE ~ NA\_character\_ # Alterar 'Outro' para NA  
 ))  
  
table(pnad\_c16241Entrevista$V2005\_filhos)  
  
# Recodificando a variável V2005 em "Pessoa responsável", "Cônjuge", e "Outros"  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V2005\_recode = case\_when(  
 V2005 == 'Pessoa\_responsável' ~ 'Pessoa\_responsável',  
 V2005 %in% c('Cônjuge') ~ 'Cônjuge',  
 TRUE ~ 'Outros'  
 ))  
  
table(pnad\_c16241Entrevista$V2005\_recode)  
  
# Recategorizando Idade  
  
# Criando histograma da variável de IDADE  
ggplot(pnad\_c16241Entrevista, aes(x = V2009)) +  
 geom\_histogram(binwidth = 5, fill = "blue", color = "black", alpha = 0.7) +  
 labs(  
 title = "idade",  
 x = "idade",  
 y = "Frequência"  
 ) +  
 theme\_minimal()  
  
# Regressão Local para Idade  
library(ggplot2)  
library(dplyr)  
  
dados\_idade <- pnad\_c16241Entrevista %>%  
 filter(!is.na(V2009), !is.na(VD4016\_log\_hora), V2009 >= 14, V2009 <= 80)  
  
ggplot(dados\_idade, aes(x = V2009, y = VD4016\_log\_hora)) +  
 geom\_point(alpha = 0.1) +  
 geom\_smooth(method = "loess", span = 0.75, se = FALSE, color = "blue") +  
 labs(  
 x = "Idade (V2009)",  
 y = "Log do salário-hora (VD4016\_log\_hora)",  
 title = "Relação entre idade e salário-hora (regressão local)"  
 ) +  
 theme\_minimal()  
  
  
# Recategorizando a idade após regressão local  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(idade\_faixa = case\_when(  
 V2009 < 30 ~ "Ate\_29", # crescimento do salário-hora  
 V2009 >= 30 & V2009 < 55 ~ "De\_30\_a\_54", # salário estabilizado  
 V2009 >= 55 ~ "Mais\_de\_55", # possível saída do mercado  
 TRUE ~ NA\_character\_  
 ))  
  
table(pnad\_c16241Entrevista$idade\_faixa)  
  
  
# Recategorizando Horas Trabalhadas na Semana  
  
# Criando histograma da variável de horas trabalhadas  
ggplot(pnad\_c16241Entrevista, aes(x = V4039)) +  
 geom\_histogram(binwidth = 5, fill = "blue", color = "black", alpha = 0.7) +  
 labs(  
 title = "Distribuição das Horas Trabalhadas (V4039)",  
 x = "Horas Trabalhadas",  
 y = "Frequência"  
 ) +  
 theme\_minimal()  
  
# Opção com base nos Regimes CLT  
# Recategorizando V4039  
# Criando uma nova variável categorizada com base em V4039  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(Horas\_trabalhadas2 = case\_when(  
 V4039 <= 32 ~ "Parcial", #regime parcial  
 V4039 >= 33 ~ "Integral", #regime integral  
 TRUE ~ NA\_character\_ # Para lidar com valores NA ou não definidos  
 ))  
  
# Verificando a distribuição da nova variável  
table(pnad\_c16241Entrevista$Horas\_trabalhadas2, useNA = "ifany")  
  
  
# Criando variável idade ao quadrado  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V2009\_quad = V2009^2) # Criando a variável idade ao quadrado  
  
head(pnad\_c16241Entrevista$V2009\_quad)  
  
  
# Mapeamento UF-região:  
mapa\_regioes <- data.frame (UF = c("Acre", "Alagoas", "Amapá", "Amazonas", "Bahia", "Ceará", "Distrito Federal",  
 "Espírito Santo", "Goiás", "Maranhão", "Mato Grosso", "Mato Grosso do Sul",   
 "Minas Gerais", "Pará", "Paraíba", "Paraná", "Pernambuco", "Piauí", "Rio de Janeiro",  
 "Rio Grande do Norte", "Rio Grande do Sul", "Rondônia", "Roraima", "Santa Catarina",   
 "São Paulo", "Sergipe", "Tocantins"),  
 Regiao = c("Norte", "Nordeste", "Norte", "Norte", "Nordeste", "Nordeste",   
 "Centro\_Oeste", "Sudeste", "Centro\_Oeste", "Nordeste", "Centro\_Oeste",   
 "Centro\_Oeste", "Sudeste", "Norte", "Nordeste", "Sul", "Nordeste",   
 "Nordeste", "Sudeste", "Nordeste", "Sul", "Norte", "Norte", "Sul",   
 "Sudeste", "Nordeste", "Norte"))  
  
# Unindo 'meu\_survey' com 'mapa\_regioes' para obter a região correspondente para cada UF  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 left\_join(mapa\_regioes, by = c("UF" = "UF"))  
  
glimpse(pnad\_c16241Entrevista)  
  
head(pnad\_c16241Entrevista$V1023)  
  
# Renomeando V1023 em capital, METROPOLITANA,E RESTO UF   
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V1023\_recode = case\_when(  
 V1023 == "Capital" ~ "Capital",  
 V1023 == "Resto da RM (Região Metropolitana, excluindo a capital)" ~ "Região\_Metropolitana",  
 V1023 == "Resto da RIDE (Região Integrada de Desenvolvimento Econômico, excluindo a capital)" ~ "RIDE",  
 V1023 == "Resto da UF (Unidade da Federação, excluindo a região metropolitana e a RIDE)" ~ "Resto\_UF",  
 TRUE ~ NA\_character\_  
 ))  
  
table(pnad\_c16241Entrevista$V1023\_recode)  
  
# Renomeando V1023 em capital+ METROPOLITANA E RIDE + RESTO UF   
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V1023\_recode2 = case\_when(  
 V1023 == "Capital" ~ "Capital",  
 V1023 == "Resto da RM (Região Metropolitana, excluindo a capital)" ~ "Regiao\_Metropolitana",  
 V1023 %in% c(  
 "Resto da RIDE (Região Integrada de Desenvolvimento Econômico, excluindo a capital)",  
 "Resto da UF (Unidade da Federação, excluindo a região metropolitana e a RIDE)"  
 ) ~ "Resto\_UF",  
 TRUE ~ NA\_character\_  
 ),  
 V1023\_recode = factor(V1023\_recode2, levels = c("Capital", "Regiao\_Metropolitana", "Resto\_UF")))  
  
table(pnad\_c16241Entrevista$V1023\_recode2)  
  
# Capital + Metropolitana x Resto UF  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V1023\_recode3 = case\_when(  
 V1023 %in% c(  
 "Capital",   
 "Resto da RM (Região Metropolitana, excluindo a capital)"  
 ) ~ "Capital\_e\_Metropolitana",  
   
 V1023 %in% c(  
 "Resto da RIDE (Região Integrada de Desenvolvimento Econômico, excluindo a capital)",   
 "Resto da UF (Unidade da Federação, excluindo a região metropolitana e a RIDE)"  
 ) ~ "Resto\_UF",  
   
 TRUE ~ NA\_character\_  
 ))  
  
table(pnad\_c16241Entrevista$V1023\_recode3)  
  
# Recodificando a variável V1023 como "Capital" e "Não Capital"  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(  
 V1023\_capncap = case\_when(  
 V1023 == "Capital" ~ "Capital", # Caso 1: Capital  
   
 V1023 %in% c(  
 "Resto da RM (Região Metropolitana, excluindo a capital)",   
 "Resto da RIDE (Região Integrada de Desenvolvimento Econômico, excluindo a capital)",   
 "Resto da UF (Unidade da Federação, excluindo a região metropolitana e a RIDE)"  
 ) ~ "Não\_Capital", # Caso 2: Todas as demais categorias agrupadas  
   
 TRUE ~ NA\_character\_ # Segurança: se houver algo inesperado  
 )  
 )  
  
table(pnad\_c16241Entrevista$V1023\_capncap)  
  
  
# Carregando dicionário CNAES  
cnae\_mapping <- readRDS("cnae\_mapping.rds")  
  
# Verificando se o dicionário CNAE está carregado corretamente  
head(cnae\_mapping) # Deve mostrar os primeiros elementos  
  
# Função para obter o nome da atividade a partir do código CNAE  
  
get\_activity\_name <- function(cnae\_code) {  
 if (cnae\_code %in% names(cnae\_mapping)) {  
 return(cnae\_mapping[[cnae\_code]])  
 } else {  
 return("Código CNAE não encontrado")  
 }  
}  
  
# Exemplos de uso da função  
cnae\_code <- "01101"  
activity\_name <- get\_activity\_name(cnae\_code)  
print(activity\_name) # Isso imprimirá "Cultivo de arroz"  
  
cnae\_code <- "03002"  
activity\_name <- get\_activity\_name(cnae\_code)  
print(activity\_name) # Isso imprimirá "Aqüicultura"  
  
  
# Substituindo os códigos pelos nomes correspondentes  
pnad\_c16241Entrevista$cnae\_mapping <- cnae\_mapping[as.character(pnad\_c16241Entrevista$V4013)]  
  
# Exibindo o resultado - VER TBL\_DF  
glimpse(pnad\_c16241Entrevista)  
  
  
# Dicionário Códigos de ocupação  
  
# Carregando dicionário CNAES  
nomes\_ocupacao <- readRDS("nomes\_ocupacao.rds")  
  
# Verificando se o dicionário CNAE está carregado corretamente  
head(nomes\_ocupacao) # Deve mostrar os primeiros elementos  
  
# Função para obter o nome da atividade a partir do código CNAE  
get\_job\_name <- function(job\_code) {  
 if (job\_code %in% names(nomes\_ocupacao)) {  
 return(nomes\_ocupacao[[job\_code]])  
 } else {  
 return("Código CBO não encontrado")  
 }  
}  
  
# Exemplos de uso da função  
job\_code <- "0512"  
job\_name <- get\_job\_name(job\_code)  
print(job\_name) # Isso imprimirá "Graduados e praças do corpo de bombeiros"  
  
job\_code <- "0110"  
job\_name <- get\_job\_name(job\_code)  
print(job\_name) # Isso imprimirá "Oficiais das forças armadas"  
  
  
# Substituindo os códigos pelos nomes correspondentes  
pnad\_c16241Entrevista$nomes\_ocupacao <- nomes\_ocupacao[as.character(pnad\_c16241Entrevista$V4010)]  
  
# Exibir o resultado  
  
glimpse(pnad\_c16241Entrevista)  
  
# Definindo a função de atribuição de grupos de CNAE  
atribuir\_grupo <- function(cnae) {  
 ifelse(cnae %in% c("79000"), "Agencias\_de\_viagens\_e\_operadoras",  
 ifelse(cnae %in% c("55000"), "Alojamento",  
 ifelse(cnae %in% c("77020"), "Alugue\_de\_automoveis",  
 ifelse(cnae %in% c("92000", "93011", "93020"), "Atividade\_desportiva\_e\_recreativa",  
 ifelse(cnae %in% c("90000", "91000"), "Atividade\_Cultural",  
 ifelse(cnae %in% c("56011", "56012", "56020"), "Alimentacao",  
 ifelse(cnae %in% c("51000"), "Transporte\_Aereo",  
 ifelse(cnae %in% c("50000"), "Transporte\_Aquaviario",  
 ifelse(cnae %in% c("49010"), "Transporte\_rodoviario",  
 ifelse(cnae %in% c("49030", "49090"), "Transporte\_ferroviario",  
 "Outro"))))))))))  
}  
  
# Aplicando a função aos seus dados e criar uma nova coluna  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(grupo\_cnae = atribuir\_grupo(V4013))  
  
# Criando a variável dummy para atividades de turismo - 1 se trabalha em turismo, 0 se não  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(turismo\_dummy = ifelse(grupo\_cnae %in% c("Agencias\_de\_viagens\_e\_operadoras",   
 "Alojamento",   
 "Aluguel\_de\_automoveis",   
 "Atividade\_desportiva\_e\_recreativa",   
 "Atividade\_Cultural",   
 "Alimentacao",   
 "Transporte\_Aereo",   
 "Transporte\_Aquaviario",   
 "Transporte\_rodoviario",   
 "Transporte\_ferroviario"), 1, 0))  
head(pnad\_c16241Entrevista$turismo\_dummy, n = 200)  
glimpse(pnad\_c16241Entrevista)  
  
table(pnad\_c16241Entrevista$turismo\_dummy)  
  
# Recodificando Composição domiciliar VD2004  
  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(grupo\_moradia = case\_when(  
 VD2004 == "Unipessoal" ~ "Unipessoal",  
 VD2004 %in% c("Nuclear", "Estendida", "Composta") ~ "Coletivo",  
 TRUE ~ NA\_character\_ # Caso queira lidar com outros valores  
 ))   
table(pnad\_c16241Entrevista$grupo\_moradia)  
  
# Recodificando tempo no emprego  
# Resumo estatístico das horas trabalhadas  
summary(pnad\_c16241Entrevista$V4040)  
  
# Média salarial por tempo no emprego  
pnad\_c16241Entrevista %>%  
 group\_by(V4040) %>%  
 summarise(media\_log\_hora = mean(VD4016\_log\_hora, na.rm = TRUE),  
 n = n()) %>%  
 arrange(desc(n))  
  
  
# Criando um gráfico de barras para variável categórica V4040  
ggplot(pnad\_c16241Entrevista, aes(x = V4040)) +  
 geom\_bar(fill = "blue", color = "black", alpha = 0.7) +  
 labs(  
 title = "Distribuição do Tempo no Emprego (V4040)",  
 x = "Tempo no Emprego",  
 y = "Frequência"  
 ) +  
 theme\_minimal()  
  
# Recodificando tempo no emprego em 2 categorias  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(V4040\_recode = case\_when(  
 V4040 %in% c('Menos de 1 mês', 'De 1 mês a menos de 1 ano', 'De 1 ano a menos de 2 anos') ~ "Menos\_de\_2\_anos",   
 V4040 == '2 anos ou mais' ~ "Mais\_de\_2\_anos",   
 TRUE ~ NA\_character\_ # Ignorar "Não aplicável" e outros valores  
 ))  
  
table(pnad\_c16241Entrevista$V4040\_recode)  
  
# Criando variável experiência  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(  
 experiencia = V2009 - VD3005\_cont - 6, # Idade menos anos de estudo menos 6  
 experiencia\_quad = experiencia^2 # Quadrado da experiência  
 )  
  
summary(pnad\_c16241Entrevista$VD4011)  
  
library(ggplot2)  
  
# Criando o gráfico de barras para a variável VD4011  
ggplot(pnad\_c16241Entrevista, aes(x = VD4011)) +  
 geom\_bar(fill = "steelblue", color = "black", alpha = 0.7) +  
 labs(  
 title = "Distribuição das Ocupações (VD4011)",  
 x = "Categorias Ocupacionais",  
 y = "Frequência"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) # Rotacionar os textos para melhor visualização  
  
# Recategorização  
library(dplyr)  
  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 # Excluir a categoria "Membros das forças armadas, policiais e bombeiros militares"  
 filter(VD4011 != "Membros das forças armadas, policiais e bombeiros militares") %>%  
 # Recodificar as categorias em três grupos  
 mutate(  
 VD4011\_recode = case\_when(  
 VD4011 %in% c("Diretores e gerentes",   
 "Profissionais das ciências e intelectuais") ~ "Alta\_qualificacao",  
   
 VD4011 %in% c("Técnicos e profissionais de nível médio",  
 "Trabalhadores de apoio administrativo",  
 "Trabalhadores dos serviços, vendedores dos comércios e mercados",  
 "Trabalhadores qualificados da agropecuária, florestais, da caça e da pesca",  
 "Trabalhadores qualificados, operários e artesões da construção, das artes mecânicas e outros ofícios") ~ "Media\_qualificacao",  
   
 VD4011 %in% c("Operadores de instalações e máquinas e montadores",  
 "Ocupações elementares",  
 "Ocupações maldefinidas") ~ "Baixa\_qualificacao",  
   
 TRUE ~ NA\_character\_  
 )  
 )  
  
table(pnad\_c16241Entrevista$VD4011\_recode)  
  
# Criando períodos Pandemia  
pnad\_c16241Entrevista <- pnad\_c16241Entrevista %>%  
 mutate(Pandemia = case\_when(  
 Ano %in% c(2016, 2017, 2018, 2019) ~ "Antes\_Pandemia",  
 Ano %in% c(2020, 2021) ~ "Durante\_Pandemia",  
 Ano %in% c(2022, 2023, 2024) ~ "Apos\_Pandemia"  
 )) %>%  
 mutate(Pandemia = factor(Pandemia, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia")))  
  
table(pnad\_c16241Entrevista$Pandemia)  
  
table(pnad\_c16241Entrevista$Pandemia, useNA = "ifany")  
  
  
## INCORPORANDO DESENHO AMOSTRAL COMPLEXO ----------------------------------  
  
library(survey)  
  
# Definindo o desenho amostral  
pnad\_survey <- svydesign(  
 ids = ~UPA,  
 strata = ~Estrato,  
 weights = ~V1028,  
 data = pnad\_c16241Entrevista,  
 nest = TRUE  
)  
  
# Verificando medias com e sem plano amostral  
# Média simples (sem pesos)  
mean\_sem\_pesos\_salario <- mean(pnad\_c16241Entrevista$VD4016\_Real, na.rm = TRUE)  
mean\_sem\_pesos\_hora <- mean(pnad\_c16241Entrevista$Salario\_Hora, na.rm = TRUE)  
  
# Média com pesos (desenho amostral)  
mean\_com\_pesos\_salario <- svymean(~VD4016\_Real, pnad\_survey, na.rm = TRUE)  
mean\_com\_pesos\_hora <- svymean(~Salario\_Hora, pnad\_survey, na.rm = TRUE)  
  
# Exibindo resultados  
cat("🟡 Média SALÁRIO MENSAL\n")  
cat("- Sem pesos: ", round(mean\_sem\_pesos\_salario, 2), "\n")  
print(mean\_com\_pesos\_salario)  
  
cat("\n🟢 Média SALÁRIO HORA\n")  
cat("- Sem pesos: ", round(mean\_sem\_pesos\_hora, 2), "\n")  
print(mean\_com\_pesos\_hora)  
  
library(fastDummies)  
  
# Selecionando variáveis desejadas e criando dummies  
pnad\_Filtrada <- pnad\_c16241Entrevista %>%  
 dplyr::select(Ano, Trimestre, UPA, Estrato, V1028, ID\_DOMICILIO, V2007, V2009, V2009\_quad, idade\_faixa,   
 V2010\_sep, V2010\_grupo, V2010\_grupo2, V2005\_recode, grupo\_moradia, VD2003, VD3004\_nivel\_Escol,VD3004\_nivel\_Escol2, VD3004\_nivel\_Escol3,   
 VD3005\_cont, experiencia, Regiao, V1023\_recode, V1023\_recode2, V1023\_recode3, V1023\_capncap, V4029, V4025, V4039, V4040\_recode,   
 Horas\_trabalhadas2, VD4011\_recode, V4013, grupo\_cnae, nomes\_ocupacao, Salario\_Hora, VD4016\_Real,   
 VD4016\_log\_hora, VD4017, turismo\_dummy, Pandemia) %>%  
 dummy\_cols(select\_columns = c("V2007", "idade\_faixa", "V2010\_sep", "V2010\_grupo", "V2010\_grupo2",   
 "V2005\_recode", "V1023\_recode", "V1023\_recode2", "V1023\_recode3", "V1023\_capncap", "grupo\_moradia",   
 "VD3004\_nivel\_Escol", "VD3004\_nivel\_Escol2", "VD3004\_nivel\_Escol3", "Regiao", "V1023\_recode", "V4029", "V4025",   
 "V4040\_recode", "Horas\_trabalhadas2", "VD4011\_recode", "grupo\_cnae", "Ano", "Trimestre", "Pandemia"),  
 remove\_first\_dummy = FALSE, # Mantém todas as dummies  
 remove\_selected\_columns = FALSE) # Mantém as colunas originais  
  
glimpse(pnad\_Filtrada)  
  
## CRIANDO A BASE DE AUDITORIA COM AMOSTRAGEM ESTRATIFICADA ----------------------------------  
  
# O objetivo é garantir a presença de TODOS os anos, trimestres e ambos os setores,  
# e corrigir o problema de Estratos com apenas uma UPA (PSU).  
  
# Definindo o percentual de amostragem por estrato  
proporcao\_amostra\_estratificada <- 0.05   
  
# Amostragem Estratificada  
pnad\_auditoria\_bruta <- pnad\_Filtrada %>%  
 # Agrupando pelos estratos cruciais para manter a representação (PSM)  
 group\_by(Ano, Trimestre, turismo\_dummy) %>%  
 slice\_sample(prop = proporcao\_amostra\_estratificada, replace = FALSE) %>%  
 ungroup()  
  
# Verificação e correção do prob lema de estrato/PSU único (Solução para 'onestrat()')  
# Identificando estratos na amostra que têm apenas 1 PSU (UPA)  
psu\_por\_estrato <- pnad\_auditoria\_bruta %>%  
 group\_by(Estrato) %>%  
 summarise(n\_psu = n\_distinct(UPA), .groups = "drop")  
  
estratos\_problematicos <- psu\_por\_estrato %>%  
 filter(n\_psu <= 1) %>% # Estratos com 0 ou 1 UPA (o erro ocorre em 1)  
 pull(Estrato)  
  
# Removendo todas as observações pertencentes aos estratos problemáticos  
# Isso garante que a condição do pacote survey seja atendida.  
pnad\_auditoria\_limpa <- pnad\_auditoria\_bruta %>%  
 filter(!Estrato %in% estratos\_problematicos)  
  
# Contagem para auditoria:  
cat(  
 "Observações originais na base auditável: ", nrow(pnad\_auditoria\_bruta), "\n",  
 "Estratos problemáticos removidos: ", length(estratos\_problematicos), "\n",  
 "Observações finais na base auditável: ", nrow(pnad\_auditoria\_limpa), "\n"  
)  
  
# Salvando a versão final limpa  
saveRDS(pnad\_auditoria\_limpa, file = "pnad\_auditoria\_amostra\_estratificada\_limpa.rds")  
  
  
# Substituindo a base grande pela amostra auditável limpa  
pnad\_Filtrada <- pnad\_auditoria\_limpa  
  
# Limpando objetos temporários  
rm(pnad\_auditoria\_bruta, pnad\_auditoria\_limpa, psu\_por\_estrato, estratos\_problematicos)  
gc()  
  
  
## ABRINDO BASE AUDITÁVEL -------------------------------------  
  
setwd("C:/Users/Ana Oliveira/Desktop/pnadc 2023") #Altere para o nome do seu diretório  
  
# Carregando o arquivo .rds  
pnad\_auditoria\_limpa <- readRDS("pnad\_auditoria\_amostra\_estratificada\_limpa.rds")  
  
# Substituindo a base grande pela amostra auditável limpa  
pnad\_Filtrada <- pnad\_auditoria\_limpa  
  
# Abrindo pacotes, caso ainda nao tenha feito isso  
library(tidyverse)  
library(survey)  
library(srvyr)  
library(PNADcIBGE)  
library(dplyr)  
  
  
## TRANSFORMANDO EM OBJETO SRVYR -------------------------------------  
  
pnad\_Filtrada\_srvyr <- pnad\_Filtrada %>%  
 as\_survey\_design(  
 ids = UPA,  
 strata = Estrato,  
 weights = V1028,  
 nest = TRUE  
 )  
nrow(pnad\_Filtrada\_srvyr)  
  
## ANÁLISES DESCRITIVAS ----------------------------------------------------  
  
# Pacotes necessários  
library(srvyr)  
library(dplyr)  
library(rlang)  
library(purrr)  
library(ggplot2)  
library(writexl)  
library(tidyr)  
  
  
# Preparando para obter rótulos corretos  
pnad\_Filtrada\_srvyr <- pnad\_Filtrada\_srvyr %>%  
 mutate(  
 Carteira\_assinada\_f = factor(V4029\_Sim,  
 levels = c(0,1),  
 labels = c("Não","Sim")),  
 Contrato\_temp\_f = factor(V4025\_Não,  
 levels = c(1,0),  
 labels = c("Não","Sim"))  
 )  
  
  
# Lista das variáveis categóricas  
vars\_cat <- c(  
 "Ano",  
 "V2007", # Sexo  
 "V2010\_grupo", # Raça (grupo)  
 "idade\_faixa", # Faixa etária  
 "VD3004\_nivel\_Escol", # Escolaridade Nível  
 "VD3004\_nivel\_Escol2", # Escolaridade Nível 2  
 "VD3004\_nivel\_Escol3", # Escolaridade Nível 3  
 "V2005\_recode", # Posição domiciliar  
 "grupo\_moradia", # Grupo moradia  
 "Regiao", # Região  
 "V1023\_recode3", # Tipo região 3  
 "V4040\_recode", # Tempo de emprego  
 "VD4011\_recode", # Qualificação  
 "Carteira\_assinada\_f", # Carteira assinada  
 "Contrato\_temp\_f", # Contrato temporário  
 "Horas\_trabalhadas2"  
)  
  
glimpse(pnad\_Filtrada)  
  
# Vetor de rótulos (nome\_original = rótulo\_amigável)  
rotulos <- c(  
 Ano = "Ano",  
 Trimestre = "Trimestre",  
 V2007 = "Sexo",  
 V2010\_grupo = "Raça",  
 idade\_faixa = "Faixa etária",  
 VD3004\_nivel\_Escol = "Escolaridade Nível",  
 VD3004\_nivel\_Escol2 = "Escolaridade Nível\_",  
 VD3004\_nivel\_Escol3 = "Escolaridade Nível\_\_",  
 V2005\_recode = "Posição domiciliar",  
 grupo\_moradia = "Grupo moradia",  
 Regiao = "Região",  
 V1023\_recode3 = "Tipo região",  
 Horas\_trabalhadas2 = "Regime Horas Trabalhadas",  
 Horas\_trabalhadas2\_Integral = "Integral",  
 Horas\_trabalhadas2\_Parcial = "Parcial",  
 V4040\_recode = "Tempo de emprego",  
 VD4011\_recode = "Qualificação",  
 Carteira\_assinada\_f = "Carteira assinada",  
 Contrato\_temp\_f = "Contrato temporário",  
 V2009 = "Idade (anos)",  
 experiencia = "Experiência (anos)",  
 VD3005\_cont = "Anos de Escolaridade",  
 VD4016\_Real = "Salário Real (R$)",  
 Salario\_Hora = "Salário por Hora (R$)",  
 grupo\_cnae = "Atividade Econômica",  
 nomes\_ocupacao = "Ocupação"  
)  
  
# Descritivas com percentual interno por grupo   
  
# Função que calcula pct interna e salário médio por categoria dentro de cada grupo  
tabela\_cat\_dist\_interna <- function(var\_nome) {  
 var\_sym <- sym(var\_nome)  
   
 pnad\_Filtrada\_srvyr %>%  
 group\_by(  
 Tipo = factor(turismo\_dummy, levels = c(1,0),  
 labels = c("Turismo","Não turismo")),  
 Categoria = !!var\_sym  
 ) %>%  
 summarise(  
 total = survey\_total(vartype = NULL),  
 sal\_med = survey\_mean(VD4016\_Real, vartype = NULL),  
 .groups = "drop\_last"  
 ) %>%  
 mutate(  
 Categoria = as.character(Categoria),  
 pct\_interna = total / sum(total) \* 100,  
 Variavel = rotulos[var\_nome]  
 ) %>%  
 ungroup() %>%  
 select(Variavel, Tipo, Categoria, pct\_interna, sal\_med)  
}  
  
# Aplicando a todas as variáveis categóricas  
dist\_interna <- map\_dfr(vars\_cat, tabela\_cat\_dist\_interna)  
  
# Organizando em wide para Excel (pct\_interna\_\* e sal\_med\_\*)  
dist\_interna\_wide <- dist\_interna %>%  
 pivot\_wider(  
 names\_from = Tipo,  
 values\_from = c(pct\_interna, sal\_med),  
 names\_glue = "{.value}\_{Tipo}"  
 )  
  
# Exportando  
write\_xlsx(  
 list(Distribuição\_Interna = dist\_interna\_wide),  
 path = "Descritivas\_PCT\_interno\_por\_tipo\_trabalhador\_com\_salario\_medio.xlsx"  
)  
  
# Gráficos para descritivas separadas por categorias de trabalhadores  
  
library(ggplot2)  
library(dplyr)  
library(tidyr)  
  
# Sem a categoria -99 de raça:  
  
# Abrindo o PDF de várias páginas  
pdf("graficos\_dist\_interna\_com\_salario.pdf", onefile = TRUE, width = 10, height = 6)  
  
for (var in unique(dist\_interna$Variavel)) {  
 df <- dist\_interna %>%   
 filter(Variavel == var)  
   
 # Removendo o -99 somente para Raça  
 if (var == "Raça") {  
 df <- df %>% filter(Categoria != "-99")  
 }  
   
 # Gráfico de distribuição interna por tipo de trabalhador  
 p1 <- ggplot(df, aes(x = Categoria, y = pct\_interna, fill = Tipo)) +  
 geom\_col(position = "dodge") +  
 labs(  
 title = paste0("Distribuição interna – ", var),  
 x = var,  
 y = "Percentual interno (%)"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 print(p1)  
   
 # Gráfico de salário médio por categoria e tipo de trabalhador  
 p2 <- ggplot(df, aes(x = Categoria, y = sal\_med, fill = Tipo)) +  
 geom\_col(position = "dodge") +  
 labs(  
 title = paste0("Salário médio – ", var),  
 x = var,  
 y = "Salário médio (R$)"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
 print(p2)  
}  
  
dev.off()  
  
  
# Descritivas Numéricas para cada grupo Turismo vs não Turismo  
  
# Lista das variáveis numéricas para média + desvio-padrão  
vars\_cont <- c("V2009", "V4039", "experiencia", "VD3005\_cont", "VD4016\_Real", "Salario\_Hora")  
  
# Função que calcula média e DP dentro de cada grupo, com labels sem espaço  
tabela\_cont\_dist\_interna <- function(var\_nome) {  
 var\_sym <- sym(var\_nome)  
   
 pnad\_Filtrada\_srvyr %>%  
 group\_by(  
 Tipo = factor(turismo\_dummy,  
 levels = c(1, 0),  
 labels = c("Turismo", "Nao\_turismo"))  
 ) %>%  
 summarise(  
 media = survey\_mean(!!var\_sym, vartype = NULL),  
 desvio = survey\_sd( !!var\_sym, vartype = NULL),  
 .groups = "drop"  
 ) %>%  
 mutate(  
 Variavel = rotulos[var\_nome]  
 ) %>%  
 select(Variavel, Tipo, media, desvio)  
}  
  
# Empilhando  
tabelas\_cont\_dist\_interna <- map\_dfr(vars\_cont, tabela\_cont\_dist\_interna)  
  
# Pivot\_wider e adiciona a diferença de médias  
t\_cont\_wide <- tabelas\_cont\_dist\_interna %>%  
 pivot\_wider(  
 names\_from = Tipo,  
 values\_from = c(media, desvio),  
 names\_glue = "{.value}\_{Tipo}"  
 ) %>%  
 mutate(  
 diff\_media = media\_Turismo - media\_Nao\_turismo  
 )  
  
# Exporta  
write\_xlsx(  
 list(`Contínuas\_Dist\_Interna` = t\_cont\_wide),  
 path = "descr\_cont\_dist\_interna\_por\_trab.xlsx"  
)  
  
  
# Gráficos variáveis por grupo turismo vs nao turismo  
library(ggplot2)  
  
# Abrindo o PDF de várias páginas  
pdf("graficos\_continuas\_e\_salario\_por\_trab.pdf",  
 onefile = TRUE, width = 10, height = 6)  
  
# Para cada variável contínua, média ±1DP  
for (var in unique(tabelas\_cont\_dist\_interna$Variavel)) {  
 dfp <- tabelas\_cont\_dist\_interna %>%  
 filter(Variavel == var)  
   
 p <- ggplot(dfp, aes(x = Tipo, y = media, fill = Tipo)) +  
 geom\_col(width = 0.6) +  
 geom\_errorbar(aes(  
 ymin = media - desvio,  
 ymax = media + desvio  
 ), width = 0.2) +  
 labs(  
 title = paste0("Média ±1 DP de ", var, " por tipo de trabalhador"),  
 x = "Tipo de trabalhador",  
 y = var  
 ) +  
 theme\_minimal() +  
 theme(  
 legend.position = "none",  
 axis.text.x = element\_text(angle = 45, hjust = 1)  
 )  
 print(p)  
}  
  
# Distribuição salarial (densidade) por grupo  
salary\_df <- pnad\_Filtrada\_srvyr %>%  
 mutate(  
 Tipo = factor(turismo\_dummy,  
 levels = c(1, 0),  
 labels = c("Turismo", "Não turismo"))  
 ) %>%  
 select(Tipo, Salario = VD4016\_Real) %>%  
 filter(!is.na(Salario))  
  
p\_hist <- ggplot(salary\_df, aes(x = Salario, fill = Tipo)) +  
 geom\_density(alpha = 0.4) +  
 labs(  
 title = "Densidade da distribuição salarial por tipo de trabalhador",  
 x = "Salário (R$)",  
 y = "Densidade"  
 ) +  
 theme\_minimal()  
print(p\_hist)  
  
# Boxplot salarial por grupo  
p\_box <- ggplot(salary\_df, aes(x = Tipo, y = Salario, fill = Tipo)) +  
 geom\_boxplot() +  
 labs(  
 title = "Boxplot salarial por tipo de trabalhador",  
 x = "Tipo de trabalhador",  
 y = "Salário (R$)"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
print(p\_box)  
  
# Fechando o dispositivo e gera o PDF  
dev.off()  
  
# Frequência por ACT  
freq\_turismo\_cnae <- pnad\_Filtrada\_srvyr %>%  
 filter(turismo\_dummy == 1) %>%  
 group\_by(grupo\_cnae) %>%  
 summarise(freq\_ponderada = survey\_total(vartype = NULL)) %>%  
 arrange(desc(freq\_ponderada))  
  
# Vendo resultado  
freq\_turismo\_cnae  
  
  
total\_turismo <- pnad\_Filtrada\_srvyr %>%  
 filter(turismo\_dummy == 1) %>%  
 summarise(total = survey\_total(vartype = NULL)) %>%  
 pull(total)  
  
freq\_turismo\_cnae <- freq\_turismo\_cnae %>%  
 mutate(freq\_pct = freq\_ponderada / total\_turismo \* 100)  
  
freq\_turismo\_cnae  
  
# Salvando no Excel  
write\_xlsx(freq\_turismo\_cnae, "freq\_turismo\_grupo\_cnae.xlsx")  
  
  
# Contando pessoas por CNAE e por quantis  
  
# Percentis de interesse  
quantis\_percent <- c(10, 25, 50, 75, 90)  
  
# Criando percent\_rank (de 1 a 100) por grupo CNAE — percentil aproximado  
# mas filtrando SÓ turismo\_dummy == 1  
pnad\_srvyr\_percent\_rank <- pnad\_Filtrada\_srvyr %>%  
 filter(turismo\_dummy == 1) %>% # só trabalhadores do turismo  
 group\_by(grupo\_cnae) %>%  
 mutate(percent\_rank = ntile(VD4016\_Real, 100)) %>%  
 ungroup()  
  
# Filtrando só os percent\_rank que interessam e calcula frequência ponderada  
freq\_por\_quantil <- pnad\_srvyr\_percent\_rank %>%  
 filter(percent\_rank %in% quantis\_percent) %>%  
 group\_by(grupo\_cnae, percent\_rank) %>%  
 summarise(  
 freq\_ponderada = survey\_total(vartype = NULL),  
 .groups = "drop"  
 ) %>%  
 group\_by(grupo\_cnae) %>%  
 mutate(  
 total\_cnae = sum(freq\_ponderada),  
 percentual = freq\_ponderada / total\_cnae \* 100  
 ) %>%  
 arrange(grupo\_cnae, percent\_rank)  
  
# Transformando para formato largo -> cada quantil vira uma coluna  
tabela\_final <- freq\_por\_quantil %>%  
 select(grupo\_cnae, percent\_rank, percentual) %>%  
 pivot\_wider(  
 names\_from = percent\_rank,  
 values\_from = percentual,  
 names\_prefix = "quantil\_"  
 )  
  
# Visualizando no console ou View()  
print(tabela\_final)  
  
# View(tabela\_final) # se quiser abrir no Viewer do RStudio  
  
# Exportando para Excel  
write\_xlsx(tabela\_final,  
 "Salario\_percentil\_cnae\_turismo.xlsx")  
  
  
# Verificando proporção por ACT com valores salariais por pecentis  
  
  
# Definindo os percentis de interesse (em proporção)  
  
quantis\_prop <- c(0.10, 0.25, 0.50, 0.75, 0.90)  
quantis\_int <- quantis\_prop \* 100 # = c(10, 25, 50, 75, 90)  
  
# (Re)Criando o objeto srvyr, caso ainda não exista  
# Ajuste "ids", "strata" e "weights" de acordo com sua base PNAD  
  
# pnad\_Filtrada\_srvyr <- pnad\_original %>%   
as\_survey\_design(  
 ids = UPA, # ou outra coluna de conglomerado  
 strata = Estrato, # ou outra coluna de estrato  
 weights = V1028, # ajuste para a variável de peso  
 nest = TRUE  
)  
  
# Criando o subconjunto “turismo” (turismo\_dummy == 1 e salário não-NA)  
pnad\_turismo\_srvyr <- pnad\_Filtrada\_srvyr %>%  
 filter(  
 turismo\_dummy == 1,  
 !is.na(VD4016\_Real)  
 )  
  
# Calculando o valor de salário em cada percentil (10, 25, 50, 75, 90)  
# Este “survey\_quantile(…, vartype = NULL)” gera colunas:  
# quantil\_10, quantil\_25, quantil\_50, quantil\_75, quantil\_90  
  
valores\_quantis <- pnad\_turismo\_srvyr %>%  
 group\_by(grupo\_cnae) %>%  
 summarise(  
 quantil\_10 = survey\_quantile(VD4016\_Real, quantile = 0.10, vartype = NULL),  
 quantil\_25 = survey\_quantile(VD4016\_Real, quantile = 0.25, vartype = NULL),  
 quantil\_50 = survey\_quantile(VD4016\_Real, quantile = 0.50, vartype = NULL),  
 quantil\_75 = survey\_quantile(VD4016\_Real, quantile = 0.75, vartype = NULL),  
 quantil\_90 = survey\_quantile(VD4016\_Real, quantile = 0.90, vartype = NULL),  
 .groups = "drop"  
 )  
  
# Verificando que o data.frame “valores\_quantis” ficou com colunas:  
# grupo\_cnae | quantil\_10 | quantil\_25 | quantil\_50 | quantil\_75 | quantil\_90  
# Cada quantil\_X é o valor de salário que separa exatamente o Xº percentil.  
  
# Criando “percent\_rank” e agrupar cada registro em faixas:  
# Faixas definidas em “cut\_bins”:  
#  
# 1) “0\_10” = percent\_rank <= 10,  
# 2) “10\_25” = 10 < percent\_rank <= 25,  
# 3) “25\_50” = 25 < percent\_rank <= 50,  
# 4) “50\_75” = 50 < percent\_rank <= 75,  
# 5) “75\_90” = 75 < percent\_rank <= 90,  
# 6) “90\_100” = percent\_rank > 90  
# ————————————————————————————————————————————————————————  
faixas\_percentil <- pnad\_turismo\_srvyr %>%  
 # Gerando percent\_rank de 1 a 100 por grupo\_cnae  
 mutate(percent\_rank = ntile(VD4016\_Real, 100)) %>%  
 # Criando uma coluna “bin” que descreve a faixa de percentil de cada registro  
 mutate(  
 bin = case\_when(  
 percent\_rank <= 10 ~ "0\_10",  
 percent\_rank <= 25 ~ "10\_25",  
 percent\_rank <= 50 ~ "25\_50",  
 percent\_rank <= 75 ~ "50\_75",  
 percent\_rank <= 90 ~ "75\_90",  
 TRUE ~ "90\_100"  
 )  
 ) %>%  
 # Agora agrupando por (grupo\_cnae, bin) e somando os pesos  
 group\_by(grupo\_cnae, bin) %>%  
 summarise(  
 freq\_bin = survey\_total(vartype = NULL), # soma de pesos naquela faixa  
 .groups = "drop"  
 )  
  
  
# Calculando para cada grupo\_cnae, o TOTAL (soma de todos os pesos)  
# para depois transformar freq\_bin em percentual.  
total\_por\_cnae <- pnad\_turismo\_srvyr %>%  
 group\_by(grupo\_cnae) %>%  
 summarise(  
 total\_cnae = survey\_total(vartype = NULL),  
 .groups = "drop"  
 )  
  
  
# Juntando “faixas\_percentil” com “total\_por\_cnae” e calcular % de cada bin  
perc\_por\_faixa <- faixas\_percentil %>%  
 left\_join(total\_por\_cnae, by = "grupo\_cnae") %>%  
 mutate(  
 perc\_ft = freq\_bin / total\_cnae \* 100  
 ) %>%  
 select(grupo\_cnae, bin, perc\_ft)  
  
# Transformando “perc\_por\_faixa” em formato WIDE, de modo que cada “bin”  
# vire uma coluna separada: “perc\_ft\_0\_10”, “perc\_ft\_10\_25”, …, “perc\_ft\_90\_100”  
  
perc\_por\_faixa\_wide <- perc\_por\_faixa %>%  
 pivot\_wider(  
 names\_from = bin,  
 values\_from = perc\_ft,  
 names\_prefix = "perc\_ft\_"  
 )  
  
  
# Unindo “valores\_quantis” + “perc\_por\_faixa\_wide” em uma tabela única  
tabela\_final <- valores\_quantis %>%  
 left\_join(perc\_por\_faixa\_wide, by = "grupo\_cnae") %>%  
 # selecionando colunas na ordem que desejar  
 select(  
 grupo\_cnae,  
 quantil\_10, quantil\_25, quantil\_50, quantil\_75, quantil\_90,  
 perc\_ft\_0\_10, perc\_ft\_10\_25, perc\_ft\_25\_50,  
 perc\_ft\_50\_75, perc\_ft\_75\_90, perc\_ft\_90\_100  
 )  
  
names(valores\_quantis)  
# Renomeando percentis  
valores\_quantis <- valores\_quantis %>%  
 rename(  
 quantil\_10 = quantil\_10\_q10,  
 quantil\_25 = quantil\_25\_q25,  
 quantil\_50 = quantil\_50\_q50,  
 quantil\_75 = quantil\_75\_q75,  
 quantil\_90 = quantil\_90\_q90  
 ) %>%  
 select(grupo\_cnae, quantil\_10, quantil\_25, quantil\_50, quantil\_75, quantil\_90)  
names(valores\_quantis)  
# Rodando passo anterior novamente  
  
  
# Visualizando e exportando para Excel  
  
print(tabela\_final)  
# View(tabela\_final) # se quiser abrir no Viewer do RStudio  
  
write\_xlsx(  
 tabela\_final,  
 "Salario\_quantis\_e\_percentual\_por\_faixa\_cnae\_turismo.xlsx"  
)  
  
  
# Obtendo ocupações mais frequentes por grupo e percentil  
library(dplyr)  
library(tidyr)  
glimpse(pnad\_Filtrada\_srvyr)  
  
# Definindo “cortes” (percentis) de interesse  
  
# Esses são os valores de percent\_rank que definem o topo de cada faixa  
cortes <- c(10, 25, 50, 75, 90)   
  
  
# Criando percent\_rank (1 a 100) dentro de cada grupo “turismo\_dummy”  
pnad\_classificada <- pnad\_Filtrada %>%  
 group\_by(turismo\_dummy) %>%  
 mutate(  
 # percent\_rank de 1 a 100  
 percent\_rank = ntile(VD4016\_Real, 100)  
 ) %>%  
 ungroup()  
  
  
# Criando coluna “bin” para agrupar percent\_rank em intervalos:  
# 0\_10 (1≤r≤10), 10\_25 (11≤r≤25), 25\_50 (26≤r≤50),  
# 50\_75 (51≤r≤75), 75\_90 (76≤r≤90), 90\_100 (91≤r≤100).  
  
pnad\_classificada <- pnad\_classificada %>%  
 mutate(  
 bin = case\_when(  
 percent\_rank <= 10 ~ "0\_10",  
 percent\_rank <= 25 ~ "10\_25",  
 percent\_rank <= 50 ~ "25\_50",  
 percent\_rank <= 75 ~ "50\_75",  
 percent\_rank <= 90 ~ "75\_90",  
 TRUE ~ "90\_100"  
 )  
 )  
  
  
# Calculando o “tamanho” (quantidade de linhas) de cada bin  
# sem ponderação, para cada combinação (turismo\_dummy, bin)  
  
group\_sizes <- pnad\_classificada %>%  
 group\_by(turismo\_dummy, bin) %>%  
 summarise(  
 group\_size = n(), # número de linhas naquele bin  
 .groups = "drop"  
 ) %>%  
 # Para juntar mais tarde com os “cortes de quantil”, vamos associar  
 # a cada “bin” seu percent\_rank de corte correspondente:  
 mutate(  
 percent\_rank = case\_when(  
 bin == "0\_10" ~ 10L,  
 bin == "10\_25" ~ 25L,  
 bin == "25\_50" ~ 50L,  
 bin == "50\_75" ~ 75L,  
 bin == "75\_90" ~ 90L,  
 bin == "90\_100" ~ 100L # opcional; se você não quiser usar 100, remova ou ajuste  
 )  
 )  
  
  
# Calculando os valores “reais” de salário que definem cada percent\_rank de corte  
# (0.10, 0.25, 0.50, 0.75, 0.90, e opcionalmente 1.00)  
# Aqui usamos quantile() sem ponderação   
group\_quantiles <- pnad\_Filtrada %>%  
 group\_by(turismo\_dummy) %>%  
 summarise(  
 p10 = quantile(VD4016\_Real, probs = 0.10, na.rm = TRUE),  
 p25 = quantile(VD4016\_Real, probs = 0.25, na.rm = TRUE),  
 p50 = quantile(VD4016\_Real, probs = 0.50, na.rm = TRUE),  
 p75 = quantile(VD4016\_Real, probs = 0.75, na.rm = TRUE),  
 p90 = quantile(VD4016\_Real, probs = 0.90, na.rm = TRUE),  
 p100 = quantile(VD4016\_Real, probs = 1.00, na.rm = TRUE), # opcional  
 .groups = "drop"  
 ) %>%  
 pivot\_longer(  
 cols = starts\_with("p"),  
 names\_to = "percent\_rank",  
 names\_prefix = "p",  
 values\_to = "VD4016\_Real\_valor"  
 ) %>%  
 mutate(percent\_rank = as.integer(percent\_rank))  
  
  
# Encontrando o Top-5 de “nomes\_ocupacao” POR FREQUÊNCIA (não ponderada)  
# dentro de cada bin (“0\_10”, “10\_25”, etc.) E cada “turismo\_dummy”.  
# Em seguida, recalcule o percentual \*\*dentro do Top-5\*\*, de modo que   
# sum(freq\_pct) == 100 para as 5 linhas de cada bin.  
  
top\_extended <- pnad\_classificada %>%  
 # Filtrando apenas quem caiu em algum dos bins de interesse  
 filter(bin %in% c("0\_10", "10\_25", "25\_50", "50\_75", "75\_90", "90\_100")) %>%  
   
 # Para cada (turismo\_dummy, bin, nomes\_ocupacao), contar quantas linhas  
 group\_by(turismo\_dummy, bin, nomes\_ocupacao) %>%  
 summarise(  
 quantidade = n(),   
 .groups = "drop"  
 ) %>%  
   
 # Dentro de cada (turismo\_dummy, bin), pegar as 5 ocupações de maior frequência:  
 group\_by(turismo\_dummy, bin) %>%  
 slice\_max(quantidade, n = 5) %>% # top 5 por quantidade  
 ungroup() %>%  
   
 # Recalculando, para cada (turismo\_dummy, bin), a soma das quantidades do Top-5:  
 group\_by(turismo\_dummy, bin) %>%  
 mutate(  
 sum\_top5 = sum(quantidade) # soma das 5 quantidades daquele bin  
 ) %>%  
 ungroup() %>%  
   
 # Juntando “group\_sizes” (que tem group\_size e percent\_rank) e   
 # “group\_quantiles” (que tem VD4016\_Real\_valor para cada percent\_rank)  
 # Precisamos do percent\_rank em group\_sizes → já está lá  
 left\_join(group\_sizes, by = c("turismo\_dummy", "bin")) %>%  
 left\_join(group\_quantiles, by = c("turismo\_dummy", "percent\_rank")) %>%  
   
 # Recalculando “freq\_pct” de modo que some 100% somente dentro do Top-5:  
 mutate(  
 freq\_pct = quantidade / sum\_top5 \* 100  
 ) %>%  
   
 # Ordenando para facilitar visualização  
 arrange(turismo\_dummy, bin, desc(freq\_pct)) %>%  
   
 # Selecionando as colunas finais na ordem em que queremos exportar  
 select(  
 turismo\_dummy,  
 bin, # “0\_10”, “10\_25” etc.  
 percent\_rank, # o corte do percent\_rank (10, 25, 50, 75, 90 ou 100)  
 VD4016\_Real\_valor, # valor de salário que define aquele percent\_rank  
 group\_size, # total de linhas naquele bin  
 nomes\_ocupacao,  
 quantidade, # contagem bruta daquela ocupação no bin  
 sum\_top5, # soma das quantidades das 5 ocupações mais frequentes no bin  
 freq\_pct # porcentual (dentro do Top-5) que cada ocupação representa  
 )  
  
  
# Visualizando e exportando para Excel  
  
print(top\_extended)  
# View(top\_extended) # se estiver usando RStudio  
  
write\_xlsx(  
 top\_extended,  
 "top5\_ocupacoes\_faixas\_percentil\_turismo\_vs\_nao\_turismo.xlsx"  
)  
# Top-10 ocupações mais frequentes com freq% e salário médio  
top\_freq\_pct <- pnad\_Filtrada %>%  
 # conta por ocupação em cada grupo  
 group\_by(turismo\_dummy, nomes\_ocupacao) %>%  
 summarise(  
 frequencia = n(),  
 salario\_medio = mean(VD4016\_Real, na.rm = TRUE),  
 .groups = "drop\_last"  
 ) %>%  
 # calcula % sobre o total de cada turismo\_dummy  
 group\_by(turismo\_dummy) %>%  
 mutate(freq\_pct = frequencia / sum(frequencia) \* 100) %>%  
 # seleciona top-10  
 arrange(desc(frequencia)) %>%  
 slice\_head(n = 10) %>%  
 ungroup()  
  
View(top\_freq\_pct)  
  
# Top-10 ocupações com maiores salários médios, com freq%  
top\_salario\_pct <- pnad\_Filtrada %>%  
 group\_by(turismo\_dummy, nomes\_ocupacao) %>%  
 summarise(  
 salario\_medio = mean(VD4016\_Real, na.rm = TRUE),  
 frequencia = n(),  
 .groups = "drop\_last"  
 ) %>%  
 group\_by(turismo\_dummy) %>%  
 mutate(freq\_pct = frequencia / sum(frequencia) \* 100) %>%  
 arrange(desc(salario\_medio)) %>%  
 slice\_head(n = 10) %>%  
 ungroup()  
  
View(top\_salario\_pct)  
  
# Top-10 ocupações com menores salários médios, com freq%  
bottom\_salario\_pct <- pnad\_Filtrada %>%  
 group\_by(turismo\_dummy, nomes\_ocupacao) %>%  
 summarise(  
 salario\_medio = mean(VD4016\_Real, na.rm = TRUE),  
 frequencia = n(),  
 .groups = "drop\_last"  
 ) %>%  
 group\_by(turismo\_dummy) %>%  
 mutate(freq\_pct = frequencia / sum(frequencia) \* 100) %>%  
 arrange(salario\_medio) %>%  
 slice\_head(n = 10) %>%  
 ungroup()  
  
View(bottom\_salario\_pct)  
  
# Montando uma lista com nomes das abas e os respectivos data.frames  
tabelas\_para\_excel <- list(  
 Percentis = top\_extended,  
 Top10\_Frequentes = top\_freq\_pct,  
 Top10\_MaioresSalarios= top\_salario\_pct,  
 Top10\_MenoresSalarios= bottom\_salario\_pct  
)  
  
# Escrevendo tudo em um único arquivo, cada elemento da lista vira uma aba  
write\_xlsx(tabelas\_para\_excel,  
 path = "tabelas\_salários ocupação\_turismo\_vs\_naoturismo.xlsx")  
  
  
## ANÁLISE DE MATCHING -----------------------------------------------------  
  
### Análise de Matching Anos e Trimestres ---------------------------------------  
  
# Carregando pacotes  
library(Matching)  
library(tableone)  
library(writexl)  
library(dplyr)  
library(ggplot2)  
library(tidyr) # Necessário para a função drop\_na()  
  
  
# Preparando dados  
df <- pnad\_Filtrada  
  
# Convertendo as colunas de Ano e Trimestre para formato numérico  
df <- df %>%  
 mutate(  
 Ano = as.numeric(as.character(Ano)),  
 Trimestre = as.numeric(as.character(Trimestre))  
 )  
  
# Obtendo as combinações únicas de Ano e Trimestre para o loop  
groups <- df %>% distinct(Ano, Trimestre) %>% arrange(Ano, Trimestre)  
print(groups)  
  
glimpse(df)  
  
# Definindo as covariáveis para o modelo de propensity score  
xvars <- c(  
 "V2007\_Mulher", "V2009", "V2010\_grupo\_PP",  
 "V2005\_recode\_Pessoa\_responsável", "V2005\_recode\_Cônjuge",  
 "grupo\_moradia\_Coletivo", "VD3004\_nivel\_Escol\_Fundamental\_Completo",  
 "VD3004\_nivel\_Escol\_Médio\_Completo", "VD3004\_nivel\_Escol\_Superior\_Completo",  
 "Regiao\_Nordeste", "Regiao\_Norte",  
 "Regiao\_Sudeste", "Regiao\_Sul", "V1023\_recode3\_Capital\_e\_Metropolitana",  
 "V4029\_Sim", "V4040\_recode\_Mais\_de\_2\_anos", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4025\_Não"  
)  
  
# Função logit para transformar os escores  
logit <- function(p) log(p/(1-p))  
  
# Inicializando listas para armazenar os resultados de cada iteração  
balance\_list <- list()  
ttest\_list <- list()  
pares\_global <- list()  
dados\_salvos\_para\_analise\_final <- list() # Lista para análise pós-loop  
  
  
# Loop principal de matching por período  
for (i in 1:nrow(groups)) {  
   
 current\_year <- groups$Ano[i]  
 current\_trimester <- groups$Trimestre[i]  
   
 cat("Processando Ano =", current\_year, "Trimestre =", current\_trimester, "\n")  
   
 # Filtra os dados para o período atual  
 mydata\_subset <- df %>% filter(Ano == current\_year, Trimestre == current\_trimester)  
   
 # Remove explicitamente linhas com NAs nas variáveis do modelo para evitar erros  
 mydata\_subset\_clean <- mydata\_subset %>%  
 drop\_na(turismo\_dummy, all\_of(xvars))  
   
 cat("Observações iniciais:", nrow(mydata\_subset), " | Observações após limpeza de NAs:", nrow(mydata\_subset\_clean), "\n")  
   
   
   
 # Estimando o propensity score  
 formula <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
 psmodel <- glm(formula, family = binomial(), data = mydata\_subset\_clean)  
   
 mydata\_subset\_clean$pscore <- psmodel$fitted.values  
 mydata\_subset\_clean$pscore <- pmin(pmax(mydata\_subset\_clean$pscore, 1e-6), 1 - 1e-6)  
   
 # Definindo e aplicando o common support  
 pscore\_t <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 1]  
 pscore\_c <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 0]  
 min\_common <- max(min(pscore\_t), min(pscore\_c))  
 max\_common <- min(max(pscore\_t), max(pscore\_c))  
   
 mydata\_subset\_cs <- mydata\_subset\_clean %>%  
 filter(pscore >= min\_common & pscore <= max\_common)  
   
 cat("Observações dentro do common support:", nrow(mydata\_subset\_cs), "\n")  
   
 # Realizando o matching  
 psmatch <- Match(  
 Tr = mydata\_subset\_cs$turismo\_dummy,  
 M = 1,  
 X = log(mydata\_subset\_cs$pscore / (1 - mydata\_subset\_cs$pscore)),  
 replace = FALSE,  
 caliper = 0.2  
 )  
   
 # Avaliação de balanceamento (original)  
 balance\_check <- MatchBalance(formula, data = mydata\_subset\_cs, match.out = psmatch, nboots = 500)  
 print(balance\_check)  
   
 # Selecionando os pares encontrados, removendo NAs de pares não formados  
 indices\_pareados <- unlist(psmatch[c("index.treated", "index.control")])  
 indices\_limpos <- na.omit(indices\_pareados)  
 matched <- mydata\_subset\_cs[indices\_limpos, ]  
   
 # Filtrando por Salario\_Hora > 0  
 matched <- matched %>% filter(Salario\_Hora > 0)  
   
 # Tabela de balanceamento pós-match (original)  
 if (nrow(matched) > 0 && length(unique(matched$turismo\_dummy)) > 1) {  
 table1 <- CreateTableOne(vars = xvars, strata = "turismo\_dummy", data = matched, test = FALSE)  
 table1\_df <- as.data.frame(print(table1, smd = TRUE))  
 table1\_df$Ano <- current\_year  
 table1\_df$Trimestre <- current\_trimester  
 table1\_df$Variable <- rownames(table1\_df)  
   
 filename\_table <- paste0("matchedtab1\_", current\_year, "\_", current\_trimester, ".xlsx")  
 write\_xlsx(table1\_df, filename\_table)  
 balance\_list[[paste0("Balance\_", current\_year, "\_", current\_trimester)]] <- table1\_df  
 }  
   
 # Analisando salários (original)  
 y\_trt <- matched$Salario\_Hora[matched$turismo\_dummy == 1]  
 y\_con <- matched$Salario\_Hora[matched$turismo\_dummy == 0]  
   
 cat("Tamanho dos grupos pareados: Tratados =", length(y\_trt), "Controles =", length(y\_con), "\n")  
   
 if (length(y\_trt) == length(y\_con) && length(y\_trt) > 0) {  
 diff\_log <- log(y\_trt) - log(y\_con)  
 t\_test\_result <- t.test(diff\_log, mu = 0)  
   
 media\_perc <- (exp(mean(diff\_log)) - 1) \* 100  
   
 t\_test\_df <- data.frame(  
 Ano = current\_year,  
 Trimestre = current\_trimester,  
 Estatistica\_t = t\_test\_result$statistic,  
 Graus\_de\_Liberdade = t\_test\_result$parameter,  
 p\_valor = t\_test\_result$p.value,  
 Media\_diferenca\_log = mean(diff\_log),  
 Media\_diferenca\_percentual\_aprox = media\_perc,  
 Conf\_Inf\_Lower = t\_test\_result$conf.int[1],  
 Conf\_Inf\_Upper = t\_test\_result$conf.int[2],  
 Hipotese\_Alternativa = t\_test\_result$alternative  
 )  
   
 filename\_ttest <- paste0("t\_test\_result\_", current\_year, "\_", current\_trimester, ".xlsx")  
 write\_xlsx(t\_test\_df, filename\_ttest)  
 ttest\_list[[paste0("TTest\_", current\_year, "\_", current\_trimester)]] <- t\_test\_df  
   
 pares\_df <- data.frame(  
 Ano = current\_year,  
 Trimestre = current\_trimester,  
 salario\_turismo = y\_trt,  
 salario\_controle = y\_con,  
 diff\_log = diff\_log,  
 diff\_percentual\_aprox = (exp(diff\_log) - 1) \* 100  
 )  
   
 pares\_global[[paste0(current\_year, "\_T", current\_trimester)]] <- pares\_df  
   
 # Gráfico de densidade do propensity score (original)  
 df\_plot <- mydata\_subset\_cs  
 df\_plot$logit\_pscore <- log(df\_plot$pscore / (1 - df\_plot$pscore))  
 df\_plot$grupo <- ifelse(df\_plot$turismo\_dummy == 1, "Turismo (Tratado)", "Controle")  
   
 caliper\_limite <- 0.2  
 media\_t <- mean(df\_plot$logit\_pscore[df\_plot$turismo\_dummy == 1])  
 lim\_inf <- media\_t - caliper\_limite  
 lim\_sup <- media\_t + caliper\_limite  
   
 g <- ggplot(df\_plot, aes(x = logit\_pscore, fill = grupo)) +  
 geom\_density(alpha = 0.4) +  
 geom\_vline(xintercept = c(lim\_inf, lim\_sup), linetype = "dashed", color = "red") +  
 labs(  
 title = paste("Distribuição do Logit do Propensity Score -", current\_year, "T", current\_trimester),  
 x = "Logit do Propensity Score",  
 y = "Densidade"  
 ) +  
 scale\_fill\_manual(values = c("Turismo (Tratado)" = "#1b9e77", "Controle" = "#d95f02")) +  
 theme\_minimal()  
   
 print(g)  
 filename\_grafico <- paste0("logit\_pscore\_densidade\_", current\_year, "\_T", current\_trimester, ".png")  
 ggsave(filename\_grafico, plot = g, width = 9, height = 5)  
   
 } else {  
 cat("Não é possível realizar o teste t pareado para este grupo.\n")  
 }  
   
 # Salvando os dados necessários para a análise final  
 if (exists("psmatch")) {  
 dados\_salvos\_para\_analise\_final[[paste0(current\_year, "\_T", current\_trimester)]] <- list(  
 dados\_com\_pscore = mydata\_subset\_clean,  
 dados\_pos\_cs = mydata\_subset\_cs,  
 objeto\_match = psmatch,  
 variaveis\_x = xvars  
 )  
 }  
   
 cat("Processo concluído para Ano =", current\_year, "Trimestre =", current\_trimester, "\n\n")  
 rm(mydata\_subset, mydata\_subset\_clean, mydata\_subset\_cs, psmodel, psmatch, matched)  
 gc()  
}  
  
# Análise pós-loop: Observações Excluídas e Gráficos SMD  
cat("\n\nIniciando análises pós-execução (Contagem Common Support e Gráficos SMD)...\n")  
  
lista\_stats\_cs\_final <- list()  
  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
   
 cat("Processando análises para o período:", periodo, "\n")  
   
 # RecuperaNDO os dados salvos do loop principal  
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_inicial\_limpo <- dados\_periodo$dados\_com\_pscore  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
   
 # Funcionalidade 1: Contagem de Observações Excluídas --  
 n\_antes\_cs <- nrow(df\_inicial\_limpo)  
 n\_depois\_cs <- nrow(df\_depois\_cs)  
 n\_excluidas <- n\_antes\_cs - n\_depois\_cs  
   
 stats\_cs\_df <- data.frame(  
 Periodo = periodo,  
 Obs\_Iniciais\_Limpas = n\_antes\_cs,  
 Obs\_Excluidas\_CS = n\_excluidas,  
 Obs\_Mantidas\_CS = n\_depois\_cs,  
 Perc\_Excluido = round((n\_excluidas / n\_antes\_cs) \* 100, 2)  
 )  
 lista\_stats\_cs\_final[[periodo]] <- stats\_cs\_df  
   
 # Funcionalidade 2: Gráfico Comparativo de SMDs --  
   
 contagem\_grupos\_cs <- table(df\_depois\_cs$turismo\_dummy)  
 if (length(contagem\_grupos\_cs) < 2 || any(contagem\_grupos\_cs == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos após o common support. Gráfico não será gerado.\n")  
 next  
 }  
   
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
   
 contagem\_grupos\_pareado <- table(df\_pareado$turismo\_dummy)  
 if (length(contagem\_grupos\_pareado) < 2 || any(contagem\_grupos\_pareado == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos APÓS O MATCHING. Gráfico não será gerado.\n")  
 next   
 }  
   
 df\_depois\_cs$turismo\_dummy <- as.factor(df\_depois\_cs$turismo\_dummy)  
 df\_pareado$turismo\_dummy <- as.factor(df\_pareado$turismo\_dummy)  
   
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs), test = FALSE)  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado), test = FALSE)  
   
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 smd\_antes <- smd\_df\_antes[xvars\_usadas, "SMD"]  
 smd\_depois <- smd\_df\_depois[xvars\_usadas, "SMD"]  
 names(smd\_antes) <- xvars\_usadas  
 names(smd\_depois) <- xvars\_usadas  
   
 df\_smd\_plot <- data.frame(  
 Variable = names(smd\_antes),  
 SMD = c(smd\_antes, smd\_depois),  
 Status = factor(rep(c("Antes do Pareamento", "Depois do Pareamento"), each = length(smd\_antes)),  
 levels = c("Antes do Pareamento", "Depois do Pareamento"))  
 )  
   
 love\_plot <- ggplot(df\_smd\_plot, aes(x = SMD, y = reorder(Variable, SMD), color = Status, shape = Status)) +  
 geom\_point(size = 3.5, alpha = 0.8) +  
 geom\_vline(xintercept = 0, linetype = "solid", color = "black") +  
 geom\_vline(xintercept = c(-0.1, 0.1), linetype = "dashed", color = "grey50") +  
 labs(  
 title = paste("Balanceamento das Covariáveis -", periodo),  
 subtitle = "Diferença Média Padronizada (SMD)",  
 x = "SMD",  
 y = "Covariável",  
 color = "Status",  
 shape = "Status"  
 ) +  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "blue")) +  
 theme\_minimal(base\_size = 12) +  
 theme(legend.position = "bottom", plot.title = element\_text(hjust = 0.5), plot.subtitle = element\_text(hjust = 0.5))  
   
 filename\_love\_plot <- paste0("GRAFICO\_SMD\_LOVEPLOT\_", gsub("T", "T", periodo), ".png")  
 ggsave(filename\_love\_plot, plot = love\_plot, width = 9, height = 10, dpi = 300)  
 cat("Gráfico SMD salvo em:", filename\_love\_plot, "\n")  
}  
  
# Consolidação dos Resultados (Original e novos)  
# Salvando a contagem de observações excluídas --  
summary\_common\_support\_final <- do.call(rbind, lista\_stats\_cs\_final)  
rownames(summary\_common\_support\_final) <- NULL  
print("-----------------------------------------------------------")  
print("Resumo Final das Observações Excluídas no Common Support:")  
print(summary\_common\_support\_final)  
write\_xlsx(summary\_common\_support\_final, "resumo\_final\_exclusoes\_common\_support.xlsx")  
cat("\nResumo das exclusões salvo em: resumo\_final\_exclusoes\_common\_support.xlsx\n")  
  
  
# Seção de análise original --  
ttest\_summary\_df <- do.call(rbind, ttest\_list) %>%  
 arrange(Ano, Trimestre)  
print(head(ttest\_summary\_df))  
write\_xlsx(ttest\_summary\_df, "resumo\_t\_test\_log\_por\_ano\_trimestre.xlsx")  
  
df\_global\_pares <- do.call(rbind, pares\_global)  
write\_xlsx(df\_global\_pares, "resultado\_global\_pares\_log\_2016\_2024.xlsx")  
  
teste\_global <- t.test(df\_global\_pares$diff\_log, mu = 0)  
print(teste\_global)  
  
teste\_global\_df <- data.frame(  
 Estatistica\_t = teste\_global$statistic,  
 Graus\_de\_Liberdade = teste\_global$parameter,  
 p\_valor = teste\_global$p.value,  
 Media\_diferenca\_log = mean(df\_global\_pares$diff\_log),  
 Media\_diferenca\_percentual\_aprox = (exp(mean(df\_global\_pares$diff\_log)) - 1) \* 100,  
 Conf\_Inf\_Lower = teste\_global$conf.int[1],  
 Conf\_Inf\_Upper = teste\_global$conf.int[2],  
 Hipotese\_Alternativa = teste\_global$alternative  
)  
write\_xlsx(teste\_global\_df, "resultado\_t\_test\_global\_2016\_2024.xlsx")  
  
smd\_summary\_df <- do.call(rbind, balance\_list)  
smd\_summary\_df <- dplyr::select(smd\_summary\_df, Ano, Trimestre, Variable, SMD)  
print(head(smd\_summary\_df))  
write\_xlsx(smd\_summary\_df, "resumo\_balanceamento\_smds\_por\_ano\_trimestre.xlsx")  
  
smd\_resumo <- smd\_summary\_df %>%  
 group\_by(Ano, Trimestre) %>%  
 summarise(Media\_SMD = mean(as.numeric(SMD), na.rm = TRUE))  
  
grafico\_smd <- ggplot(smd\_resumo, aes(x = interaction(Ano, Trimestre), y = Media\_SMD)) +  
 geom\_line(group = 1) +  
 geom\_point() +  
 labs(x = "Ano-Trimestre", y = "Média dos SMDs",  
 title = "Evolução da Média dos SMDs após o Pareamento") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 90, hjust = 1))  
print(grafico\_smd)  
ggsave("grafico\_media\_smds\_anoetrimestre.png", plot = grafico\_smd, width = 10, height = 6, dpi = 300)  
  
ttest\_summary\_df <- ttest\_summary\_df %>%  
 mutate(  
 Ano\_Trim = paste0(Ano, "T", Trimestre),  
 significativo = ifelse(p\_valor < 0.05, TRUE, FALSE)  
 )  
  
grafico\_linha <- ggplot(ttest\_summary\_df, aes(x = Ano\_Trim, y = Media\_diferenca\_percentual\_aprox, group = 1)) +  
 geom\_line(color = "blue", linewidth = 1) +  
 geom\_point(aes(color = significativo), size = 2.5) +  
 scale\_color\_manual(values = c("FALSE" = "black", "TRUE" = "red")) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Diferença percentual média de salários (Turismo vs Outros)",  
 subtitle = "Trimestres com p < 0.05 destacados em vermelho",  
 x = "Ano e Trimestre",  
 y = "Diferença percentual média (%)",  
 color = "Significativo"  
 ) +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))  
print(grafico\_linha)  
ggsave("grafico\_diferenca\_percentual\_por\_trimestre.png", plot = grafico\_linha, width = 10, height = 6, dpi = 300)  
  
df\_global\_pares <- df\_global\_pares %>%  
 mutate(Ano = as.factor(Ano))  
  
grafico\_boxplot <- ggplot(df\_global\_pares, aes(x = Ano, y = diff\_log)) +  
 geom\_boxplot(fill = "lightblue", color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Distribuição da diferença logarítmica dos salários por ano",  
 subtitle = "log(salário\_turismo / salário\_controle)",  
 x = "Ano",  
 y = "Diferença logarítmica"  
 ) +  
 theme\_minimal()  
ggsave("boxplot\_diferenca\_log\_por\_ano.png", plot = grafico\_boxplot, width = 10, height = 6, dpi = 300)  
  
ttest\_summary\_df <- ttest\_summary\_df %>%  
 mutate(  
 IC\_inferior\_perc = (exp(Conf\_Inf\_Lower) - 1) \* 100,  
 IC\_superior\_perc = (exp(Conf\_Inf\_Upper) - 1) \* 100  
 )  
  
grafico\_ic <- ggplot(ttest\_summary\_df, aes(x = Ano\_Trim, y = Media\_diferenca\_percentual\_aprox)) +  
 geom\_line(aes(group = 1), color = "steelblue", linewidth = 1) +  
 geom\_ribbon(aes(ymin = IC\_inferior\_perc, ymax = IC\_superior\_perc, group = 1),  
 fill = "steelblue", alpha = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray50") +  
 labs(  
 title = "Diferença Percentual Média de Salários por Trimestre",  
 subtitle = "Com intervalo de confiança (95%)",  
 x = "Ano e Trimestre",  
 y = "Diferença percentual média (%)"  
 ) +  
 theme\_minimal() +  
 theme(  
 axis.text.x = element\_text(angle = 45, hjust = 1),  
 axis.title = element\_text(face = "italic"),  
 panel.grid.major = element\_line(color = "gray90"),  
 panel.background = element\_rect(fill = "#f5f5f5", color = NA),  
 plot.background = element\_rect(fill = "#f5f5f5", color = NA)  
 )  
print(grafico\_ic)  
ggsave("grafico\_ic\_diferenca\_percentual.png", plot = grafico\_ic, width = 10, height = 6, dpi = 300)  
  
balance\_list\_trimestre <- balance\_list  
ttest\_list\_trimestre <- ttest\_list  
pares\_global\_trimestre <- pares\_global  
ttest\_summary\_df\_trimestre <- ttest\_summary\_df  
df\_global\_pares\_trimestre <- df\_global\_pares  
smd\_summary\_df\_trimestre <- smd\_summary\_df  
  
calcular\_ajuste\_propensity <- function(dados, formula) {  
 dados\_limpos <- dados %>% drop\_na(all\_of(all.vars(formula)))  
   
 psmodel <- glm(formula, family = binomial(), data = dados\_limpos)  
   
 nullmodel <- glm(as.formula(paste(all.vars(formula)[1], "~ 1")),  
 family = binomial(), data = dados\_limpos)  
   
 aic <- AIC(psmodel)  
 bic <- BIC(psmodel)  
 loglik <- as.numeric(logLik(psmodel))  
   
 pseudo\_r2 <- 1 - (logLik(psmodel) / logLik(nullmodel))  
   
 resultados <- data.frame(  
 AIC = round(aic, 2),  
 BIC = round(bic, 2),  
 LogLikelihood = round(loglik, 2),  
 Pseudo\_R2 = round(as.numeric(pseudo\_r2), 4)  
 )  
   
 return(resultados)  
}  
  
formula\_trimestre <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
resultado\_ajuste\_trimestre <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_trimestre)  
print(resultado\_ajuste\_trimestre)  
  
cat("\n\n-- ANÁLISE COMPLETA FINALIZADA --\n")  
  
# Gráfico SMDs antes e depois do pareamento  
# Evolução da Média de SMD (Antes vs. Depois) ##  
  
cat("\n\nGerando gráfico comparativo da evolução da média de SMD...\n")  
  
# Inicializando uma lista para guardar as médias de SMD de cada período  
smd\_evolution\_list <- list()  
  
# Loop através dos dados já salvos para extrair as médias  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
   
 # Recupera os dados do período  
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
   
 # Recria o dataframe pareado  
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
   
 # Pula se algum dos dataframes não tiver os dois grupos  
 if (length(unique(df\_depois\_cs$turismo\_dummy)) < 2 || length(unique(df\_pareado$turismo\_dummy)) < 2) {  
 next  
 }  
   
 # Usa o método robusto para extrair os SMDs  
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs))  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado))  
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 # Calcula a média dos SMDs absolutos (a magnitude do desbalanceamento)  
 media\_smd\_antes <- mean(abs(as.numeric(smd\_df\_antes[xvars\_usadas, "SMD"])), na.rm = TRUE)  
 media\_smd\_depois <- mean(abs(as.numeric(smd\_df\_depois[xvars\_usadas, "SMD"])), na.rm = TRUE)  
   
 # Guarda os resultados do período  
 smd\_evolution\_list[[periodo]] <- data.frame(  
 Periodo = periodo,  
 Media\_SMD\_Antes = media\_smd\_antes,  
 Media\_SMD\_Depois = media\_smd\_depois  
 )  
}  
  
# Consolida e prepara os dados para o ggplot  
# Une todos os dataframes da lista em um só  
smd\_evolution\_df <- do.call(rbind, smd\_evolution\_list)  
  
# Separa Ano e Trimestre para ordenação correta  
smd\_evolution\_df <- smd\_evolution\_df %>%  
 separate(Periodo, into = c("Ano", "Trimestre"), sep = "\_T", remove = FALSE) %>%  
 mutate(Ano = as.numeric(Ano), Trimestre = as.numeric(Trimestre)) %>%  
 arrange(Ano, Trimestre) %>%  
 # Garante que 'Periodo' seja um fator ordenado para o gráfico  
 mutate(Periodo = factor(Periodo, levels = unique(Periodo)))  
  
# Transforma o dataframe do formato "wide" para "long", ideal para o ggplot  
smd\_long\_df <- smd\_evolution\_df %>%  
 pivot\_longer(  
 cols = c("Media\_SMD\_Antes", "Media\_SMD\_Depois"),  
 names\_to = "Status",  
 values\_to = "Media\_SMD"  
 ) %>%  
 # Renomeia as categorias para a legenda do gráfico  
 mutate(Status = ifelse(Status == "Media\_SMD\_Antes", "Antes do Pareamento", "Depois do Pareamento"))  
  
  
# Gera o gráfico final  
grafico\_smd\_comparativo <- ggplot(smd\_long\_df, aes(x = Periodo, y = Media\_SMD, group = Status, color = Status)) +  
 geom\_line(linewidth = 1.2) +  
 geom\_point(size = 2.5) +  
 # Adiciona uma linha de referência (limite comum de balanceamento aceitável)  
 geom\_hline(yintercept = 0.1, linetype = "dashed", color = "red") +  
 # Adiciona um texto explicando a linha de referência  
 annotate("text", x = nrow(smd\_evolution\_df) - 2, y = 0.11, label = "Limite (SMD = 0.1)", color = "red", size = 3) +  
 scale\_y\_continuous(labels = scales::number\_format(accuracy = 0.01)) +  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "steelblue")) +  
 labs(  
 title = "Evolução da Média do SMD Absoluto",  
 subtitle = "Comparativo Antes e Depois do Pareamento por Propensity Score",  
 x = "Ano e Trimestre",  
 y = "Média do SMD Absoluto",  
 color = "Situação"  
 ) +  
 theme\_minimal(base\_size = 14) +  
 theme(  
 axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust = 1, size = 10),  
 legend.position = "bottom",  
 plot.title = element\_text(hjust = 0.5, face = "bold"),  
 plot.subtitle = element\_text(hjust = 0.5)  
 )  
  
# Exibe e salva o gráfico  
print(grafico\_smd\_comparativo)  
ggsave("grafico\_smd\_comparativo\_evolucao.png", plot = grafico\_smd\_comparativo, width = 14, height = 8, dpi = 300)  
  
cat("\nGráfico comparativo da evolução do SMD salvo como 'grafico\_smd\_comparativo\_evolucao.png'\n")  
  
  
# Preparação para comparação de modelos e ajuste  
cat("\n\nPreparando objetos e calculando ajuste do modelo...\n")  
  
balance\_list\_trimestre <- balance\_list  
ttest\_list\_trimestre <- ttest\_list  
pares\_global\_trimestre <- pares\_global  
ttest\_summary\_df\_trimestre <- ttest\_summary\_df  
df\_global\_pares\_trimestre <- df\_global\_pares  
smd\_summary\_df\_trimestre <- smd\_summary\_df  
  
# Função para calcular indicadores do modelo de propensity score  
calcular\_ajuste\_propensity <- function(dados, formula) {  
   
 # Remove NAs para garantir que os modelos sejam comparados nos mesmos dados  
 dados\_limpos <- dados %>% drop\_na(all\_of(all.vars(formula)))  
   
 # Rodando modelo logit de propensity  
 psmodel <- glm(formula, family = binomial(), data = dados\_limpos)  
   
 # Modelo nulo (apenas intercepto) para Pseudo R²  
 nullmodel <- glm(as.formula(paste(all.vars(formula)[1], "~ 1")),  
 family = binomial(), data = dados\_limpos)  
   
 # Calculando métricas  
 aic <- AIC(psmodel)  
 bic <- BIC(psmodel)  
 loglik <- as.numeric(logLik(psmodel))  
   
 # Pseudo R² de McFadden  
 pseudo\_r2 <- 1 - (logLik(psmodel) / logLik(nullmodel))  
   
 # Retornando resultados como data frame  
 resultados <- data.frame(  
 AIC = round(aic, 2),  
 BIC = round(bic, 2),  
 LogLikelihood = round(loglik, 2),  
 Pseudo\_R2 = round(as.numeric(pseudo\_r2), 4)  
 )  
   
 return(resultados)  
}  
  
formula\_trimestre <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
resultado\_ajuste\_trimestre <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_trimestre)  
  
print("--- Métricas de Ajuste do Modelo de Propensity Score (em toda a base) ---")  
print(resultado\_ajuste\_trimestre)  
  
# Salvando em Excel  
write\_xlsx(resultado\_ajuste\_trimestre, "metricas\_ajuste\_modelo\_anoetrimestre.xlsx")  
cat("\n\n-- ANÁLISE COMPLETA FINALIZADA --\n")  
  
  
### Análise de Matching sem Trimestres - Apenas Anos ---------------------------------------  
  
# Carregando pacotes  
library(Matching)  
library(tableone)  
library(writexl)  
library(dplyr)  
library(ggplot2)  
library(tidyr) # Necessário para a função drop\_na()  
  
# Preparando dados  
df <- pnad\_Filtrada  
  
# Converte a coluna Ano para formato numérico  
df <- df %>%  
 mutate(Ano = as.numeric(as.character(Ano)))  
  
# Obtém os anos únicos para o loop  
groups <- df %>% distinct(Ano) %>% arrange(Ano)  
print(groups)  
  
# Define as covariáveis para o modelo de propensão  
xvars <- c(  
 "V2007\_Mulher", "V2009", "V2010\_grupo\_PP",  
 "V2005\_recode\_Pessoa\_responsável", "V2005\_recode\_Cônjuge",  
 "grupo\_moradia\_Coletivo", "VD3004\_nivel\_Escol\_Fundamental\_Completo",  
 "VD3004\_nivel\_Escol\_Médio\_Completo", "VD3004\_nivel\_Escol\_Superior\_Completo",  
 "Regiao\_Nordeste", "Regiao\_Norte",  
 "Regiao\_Sudeste", "Regiao\_Sul", "V1023\_recode3\_Capital\_e\_Metropolitana",  
 "V4029\_Sim", "V4040\_recode\_Mais\_de\_2\_anos", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4025\_Não"  
)  
  
# Inicializa listas para armazenar os resultados de cada iteração  
balance\_list <- list()  
ttest\_list <- list()  
pares\_global <- list()  
dados\_salvos\_para\_analise\_final <- list() # Lista para análise pós-loop  
  
  
# Loop principal de Matching por ano  
for (i in 1:nrow(groups)) {  
   
 current\_year <- groups$Ano[i]  
   
 cat("Processando Ano =", current\_year, "\n")  
   
 # Filtra os dados para o ano atual  
 mydata\_subset <- df %>% filter(Ano == current\_year)  
   
 # Remove explicitamente linhas com NAs nas variáveis do modelo para evitar erros  
 mydata\_subset\_clean <- mydata\_subset %>%  
 drop\_na(turismo\_dummy, all\_of(xvars))  
   
 cat("Observações iniciais:", nrow(mydata\_subset), " | Observações após limpeza de NAs:", nrow(mydata\_subset\_clean), "\n")  
   
   
   
 # Estima o propensity score  
 formula <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
 psmodel <- glm(formula, family = binomial(), data = mydata\_subset\_clean)  
   
 mydata\_subset\_clean$pscore <- psmodel$fitted.values  
 mydata\_subset\_clean$pscore <- pmin(pmax(mydata\_subset\_clean$pscore, 1e-6), 1 - 1e-6)  
   
 # Define e aplica o common support  
 pscore\_t <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 1]  
 pscore\_c <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 0]  
 min\_common <- max(min(pscore\_t), min(pscore\_c))  
 max\_common <- min(max(pscore\_t), max(pscore\_c))  
   
 mydata\_subset\_cs <- mydata\_subset\_clean %>%  
 filter(pscore >= min\_common & pscore <= max\_common)  
   
 cat("Observações dentro do common support:", nrow(mydata\_subset\_cs), "\n")  
   
 # Realiza o matching  
 psmatch <- Match(  
 Tr = mydata\_subset\_cs$turismo\_dummy,  
 M = 1,  
 X = log(mydata\_subset\_cs$pscore / (1 - mydata\_subset\_cs$pscore)),  
 replace = FALSE,  
 caliper = 0.2  
 )  
   
 # Avaliação de balanceamento (original)  
 balance\_check <- MatchBalance(formula, data = mydata\_subset\_cs, match.out = psmatch, nboots = 500)  
 print(balance\_check)  
   
 # Seleciona os pares encontrados, removendo NAs de pares não formados  
 indices\_pareados <- unlist(psmatch[c("index.treated", "index.control")])  
 indices\_limpos <- na.omit(indices\_pareados)  
 matched <- mydata\_subset\_cs[indices\_limpos, ]  
   
 # Filtra por Salario\_Hora > 0  
 matched <- matched %>% filter(Salario\_Hora > 0)  
   
 # Tabela de balanceamento pós-match (original)  
 if (nrow(matched) > 0 && length(unique(matched$turismo\_dummy)) > 1) {  
 table1 <- CreateTableOne(vars = xvars, strata = "turismo\_dummy", data = matched, test = FALSE)  
 table1\_df <- as.data.frame(print(table1, smd = TRUE))  
 table1\_df$Ano <- current\_year  
 table1\_df$Variable <- rownames(table1\_df)  
   
 filename\_table <- paste0("matchedtab1\_ano\_", current\_year, ".xlsx")  
 write\_xlsx(table1\_df, filename\_table)  
 balance\_list[[as.character(current\_year)]] <- table1\_df  
 }  
   
 # Análise de salários (original)  
 y\_trt <- matched$Salario\_Hora[matched$turismo\_dummy == 1]  
 y\_con <- matched$Salario\_Hora[matched$turismo\_dummy == 0]  
   
 cat("Tamanho dos grupos pareados: Tratados =", length(y\_trt), "Controles =", length(y\_con), "\n")  
   
 if (length(y\_trt) == length(y\_con) && length(y\_trt) > 0) {  
 diff\_log <- log(y\_trt) - log(y\_con)  
 t\_test\_result <- t.test(diff\_log, mu = 0)  
   
 media\_perc <- (exp(mean(diff\_log)) - 1) \* 100  
   
 t\_test\_df <- data.frame(  
 Ano = current\_year,  
 Estatistica\_t = t\_test\_result$statistic,  
 Graus\_de\_Liberdade = t\_test\_result$parameter,  
 p\_valor = t\_test\_result$p.value,  
 Media\_diferenca\_log = mean(diff\_log),  
 Media\_diferenca\_percentual\_aprox = media\_perc,  
 Conf\_Inf\_Lower = t\_test\_result$conf.int[1],  
 Conf\_Inf\_Upper = t\_test\_result$conf.int[2],  
 Hipotese\_Alternativa = t\_test\_result$alternative  
 )  
   
 filename\_ttest <- paste0("t\_test\_result\_ano\_", current\_year, ".xlsx")  
 write\_xlsx(t\_test\_df, filename\_ttest)  
 ttest\_list[[as.character(current\_year)]] <- t\_test\_df  
   
 pares\_df <- data.frame(  
 Ano = current\_year,  
 salario\_turismo = y\_trt,  
 salario\_controle = y\_con,  
 diff\_log = diff\_log,  
 diff\_percentual\_aprox = (exp(diff\_log) - 1) \* 100  
 )  
   
 pares\_global[[as.character(current\_year)]] <- pares\_df  
   
 # Gráfico de densidade do propensity score   
 df\_plot <- mydata\_subset\_cs  
 df\_plot$logit\_pscore <- log(df\_plot$pscore / (1 - df\_plot$pscore))  
 df\_plot$grupo <- ifelse(df\_plot$turismo\_dummy == 1, "Turismo (Tratado)", "Controle")  
   
 # Código para calcular os limites do caliper adicionado novamente  
 caliper\_limite <- 0.2  
 media\_t <- mean(df\_plot$logit\_pscore[df\_plot$turismo\_dummy == 1], na.rm = TRUE)  
 lim\_inf <- media\_t - caliper\_limite  
 lim\_sup <- media\_t + caliper\_limite  
   
 g <- ggplot(df\_plot, aes(x = logit\_pscore, fill = grupo)) +  
 geom\_density(alpha = 0.4) +  
 # Linha para desenhar os limites do caliper adicionada novamente  
 geom\_vline(xintercept = c(lim\_inf, lim\_sup), linetype = "dashed", color = "red") +  
 labs(  
 title = paste("Distribuição do Logit do Propensity Score -", current\_year),  
 x = "Logit do Propensity Score",  
 y = "Densidade"  
 ) +  
 scale\_fill\_manual(values = c("Turismo (Tratado)" = "#1b9e77", "Controle" = "#d95f02")) +  
 theme\_minimal()  
   
 print(g)  
 filename\_grafico <- paste0("logit\_pscore\_densidade\_ano\_", current\_year, ".png")  
 ggsave(filename\_grafico, plot = g, width = 9, height = 5)  
   
 } else {  
 cat("Não é possível realizar o teste t pareado para este grupo.\n")  
 }  
   
 # Salvando os dados necessários para a análise final  
 if (exists("psmatch")) {  
 dados\_salvos\_para\_analise\_final[[as.character(current\_year)]] <- list(  
 dados\_com\_pscore = mydata\_subset\_clean,  
 dados\_pos\_cs = mydata\_subset\_cs,  
 objeto\_match = psmatch,  
 variaveis\_x = xvars  
 )  
 }  
   
 cat("Processo concluído para Ano =", current\_year, "\n\n")  
 rm(mydata\_subset, mydata\_subset\_clean, mydata\_subset\_cs, psmodel, psmatch, matched)  
 gc()  
}  
  
# Análise pós-loop: Observações excluídas e gráficosSMD  
cat("\n\nIniciando análises pós-execução (Contagem Common Support e Gráficos SMD)...\n")  
  
lista\_stats\_cs\_final <- list()  
  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
   
 cat("Processando análises para o período:", periodo, "\n")  
   
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_inicial\_limpo <- dados\_periodo$dados\_com\_pscore  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
   
 # Funcionalidade 1: Contagem de Observações Excluídas --  
 n\_antes\_cs <- nrow(df\_inicial\_limpo)  
 n\_depois\_cs <- nrow(df\_depois\_cs)  
 n\_excluidas <- n\_antes\_cs - n\_depois\_cs  
   
 stats\_cs\_df <- data.frame(  
 Periodo = periodo,  
 Obs\_Iniciais\_Limpas = n\_antes\_cs,  
 Obs\_Excluidas\_CS = n\_excluidas,  
 Obs\_Mantidas\_CS = n\_depois\_cs,  
 Perc\_Excluido = round((n\_excluidas / n\_antes\_cs) \* 100, 2)  
 )  
 lista\_stats\_cs\_final[[periodo]] <- stats\_cs\_df  
   
 # Funcionalidade 2: Gráfico Comparativo de SMDs --  
   
 contagem\_grupos\_cs <- table(df\_depois\_cs$turismo\_dummy)  
 if (length(contagem\_grupos\_cs) < 2 || any(contagem\_grupos\_cs == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos após o common support. Gráfico não será gerado.\n")  
 next  
 }  
   
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
   
 contagem\_grupos\_pareado <- table(df\_pareado$turismo\_dummy)  
 if (length(contagem\_grupos\_pareado) < 2 || any(contagem\_grupos\_pareado == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos APÓS O MATCHING. Gráfico não será gerado.\n")  
 next   
 }  
   
 df\_depois\_cs$turismo\_dummy <- as.factor(df\_depois\_cs$turismo\_dummy)  
 df\_pareado$turismo\_dummy <- as.factor(df\_pareado$turismo\_dummy)  
   
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs), test = FALSE)  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado), test = FALSE)  
   
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 smd\_antes <- smd\_df\_antes[xvars\_usadas, "SMD"]  
 smd\_depois <- smd\_df\_depois[xvars\_usadas, "SMD"]  
 names(smd\_antes) <- xvars\_usadas  
 names(smd\_depois) <- xvars\_usadas  
   
 df\_smd\_plot <- data.frame(  
 Variable = names(smd\_antes),  
 SMD = c(smd\_antes, smd\_depois),  
 Status = factor(rep(c("Antes do Pareamento", "Depois do Pareamento"), each = length(smd\_antes)),  
 levels = c("Antes do Pareamento", "Depois do Pareamento"))  
 )  
   
 love\_plot <- ggplot(df\_smd\_plot, aes(x = SMD, y = reorder(Variable, SMD), color = Status, shape = Status)) +  
 geom\_point(size = 3.5, alpha = 0.8) +  
 geom\_vline(xintercept = 0, linetype = "solid", color = "black") +  
 geom\_vline(xintercept = c(-0.1, 0.1), linetype = "dashed", color = "grey50") +  
 labs(  
 title = paste("Balanceamento das Covariáveis - Ano", periodo),  
 subtitle = "Diferença Média Padronizada (SMD)",  
 x = "SMD",  
 y = "Covariável",  
 color = "Status",  
 shape = "Status"  
 ) +  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "blue")) +  
 theme\_minimal(base\_size = 12) +  
 theme(legend.position = "bottom", plot.title = element\_text(hjust = 0.5), plot.subtitle = element\_text(hjust = 0.5))  
   
 filename\_love\_plot <- paste0("GRAFICO\_SMD\_LOVEPLOT\_ano\_", periodo, ".png")  
 ggsave(filename\_love\_plot, plot = love\_plot, width = 9, height = 10, dpi = 300)  
 cat("Gráfico SMD salvo em:", filename\_love\_plot, "\n")  
}  
  
# Consolidando os resultados  
# Salvando a contagem de observações excluídas --  
summary\_common\_support\_final <- do.call(rbind, lista\_stats\_cs\_final)  
rownames(summary\_common\_support\_final) <- NULL  
print("-----------------------------------------------------------")  
print("Resumo Final das Observações Excluídas no Common Support (por Ano):")  
print(summary\_common\_support\_final)  
write\_xlsx(summary\_common\_support\_final, "resumo\_final\_exclusoes\_common\_support\_ano.xlsx")  
cat("\nResumo das exclusões salvo em: resumo\_final\_exclusoes\_common\_support\_ano.xlsx\n")  
  
# Gráfico de Evolução da Média de SMDs por Ano   
  
smd\_evolution\_list <- list()  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
 if (length(unique(df\_depois\_cs$turismo\_dummy)) < 2 || length(unique(df\_pareado$turismo\_dummy)) < 2) next  
   
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs))  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado))  
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 media\_smd\_antes <- mean(abs(as.numeric(smd\_df\_antes[xvars\_usadas, "SMD"])), na.rm = TRUE)  
 media\_smd\_depois <- mean(abs(as.numeric(smd\_df\_depois[xvars\_usadas, "SMD"])), na.rm = TRUE)  
   
 smd\_evolution\_list[[periodo]] <- data.frame(Ano = as.numeric(periodo), Media\_SMD\_Antes = media\_smd\_antes, Media\_SMD\_Depois = media\_smd\_depois)  
}  
smd\_evolution\_df <- do.call(rbind, smd\_evolution\_list) %>% arrange(Ano)  
smd\_long\_df <- smd\_evolution\_df %>%  
 pivot\_longer(cols = c("Media\_SMD\_Antes", "Media\_SMD\_Depois"), names\_to = "Status", values\_to = "Media\_SMD") %>%  
 mutate(Status = ifelse(Status == "Media\_SMD\_Antes", "Antes do Pareamento", "Depois do Pareamento"))  
  
grafico\_smd\_comparativo\_ano <- ggplot(smd\_long\_df, aes(x = Ano, y = Media\_SMD, group = Status, color = Status)) +  
 geom\_line(linewidth = 1.2) +   
 geom\_point(size = 2.5) +  
 geom\_hline(yintercept = 0.1, linetype = "dashed", color = "red") +  
 annotate("text", x = max(smd\_evolution\_df$Ano) - 0.5, y = 0.11, label = "Limite (SMD = 0.1)", color = "red", size = 3.5) +  
   
 # Adicionando a paleta de cores manual para consistência  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "steelblue")) +  
   
 scale\_x\_continuous(breaks = scales::pretty\_breaks()) +  
 labs(  
 title = "Evolução da Média do SMD Absoluto por Ano",   
 subtitle = "Antes e Depois do Pareamento",   
 x = "Ano",   
 y = "Média do SMD Absoluto",   
 color = "Situação"  
 ) +  
 theme\_minimal(base\_size = 14) +   
 theme(  
 legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5, face = "bold"),   
 plot.subtitle = element\_text(hjust = 0.5)  
 )  
  
print(grafico\_smd\_comparativo\_ano)  
ggsave("grafico\_smd\_comparativo\_evolucao\_ano.png", plot = grafico\_smd\_comparativo\_ano, width = 12, height = 7, dpi = 300)  
  
# Início da sua seção de análise original --  
ttest\_summary\_df <- do.call(rbind, ttest\_list) %>% arrange(Ano)  
print(head(ttest\_summary\_df))  
write\_xlsx(ttest\_summary\_df, "resumo\_t\_test\_log\_por\_ano.xlsx")  
  
df\_global\_pares <- do.call(rbind, pares\_global)  
write\_xlsx(df\_global\_pares, "resultado\_global\_pares\_log\_por\_ano.xlsx")  
  
teste\_global <- t.test(df\_global\_pares$diff\_log, mu = 0)  
print(teste\_global)  
  
teste\_global\_df <- data.frame(  
 Estatistica\_t = teste\_global$statistic,  
 Graus\_de\_Liberdade = teste\_global$parameter,  
 p\_valor = teste\_global$p.value,  
 Media\_diferenca\_log = mean(df\_global\_pares$diff\_log),  
 Media\_diferenca\_percentual\_aprox = (exp(mean(df\_global\_pares$diff\_log)) - 1) \* 100,  
 Conf\_Inf\_Lower = teste\_global$conf.int[1],  
 Conf\_Inf\_Upper = teste\_global$conf.int[2],  
 Hipotese\_Alternativa = teste\_global$alternative  
)  
write\_xlsx(teste\_global\_df, "resultado\_t\_test\_global\_por\_ano.xlsx")  
  
smd\_summary\_df <- do.call(rbind, balance\_list)  
smd\_summary\_df <- dplyr::select(smd\_summary\_df, Ano, Variable, SMD)  
print(head(smd\_summary\_df))  
write\_xlsx(smd\_summary\_df, "resumo\_balanceamento\_smds\_por\_ano.xlsx")  
  
ttest\_summary\_df <- ttest\_summary\_df %>%  
 mutate(  
 significativo = ifelse(p\_valor < 0.05, TRUE, FALSE)  
 )  
  
grafico\_linha <- ggplot(ttest\_summary\_df, aes(x = as.factor(Ano), y = Media\_diferenca\_percentual\_aprox, group = 1)) +  
 geom\_line(color = "blue", linewidth = 1) +  
 geom\_point(aes(color = significativo), size = 2.5) +  
 scale\_color\_manual(values = c("FALSE" = "black", "TRUE" = "red")) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(title = "Diferença percentual média de salários por Ano", x = "Ano", y = "Diferença percentual média (%)", color = "Significativo (p < 0.05)") +  
 theme\_minimal()  
print(grafico\_linha)  
ggsave("grafico\_diferenca\_percentual\_por\_ano.png", plot = grafico\_linha, width = 10, height = 6, dpi = 300)  
  
ttest\_summary\_df <- ttest\_summary\_df %>%  
 mutate(  
 IC\_inferior\_perc = (exp(Conf\_Inf\_Lower) - 1) \* 100,  
 IC\_superior\_perc = (exp(Conf\_Inf\_Upper) - 1) \* 100  
 )  
  
grafico\_ic <- ggplot(ttest\_summary\_df, aes(x = as.factor(Ano), y = Media\_diferenca\_percentual\_aprox, group = 1)) +  
 geom\_line(color = "steelblue", linewidth = 1) +  
 geom\_ribbon(aes(ymin = IC\_inferior\_perc, ymax = IC\_superior\_perc), fill = "steelblue", alpha = 0.2) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray50") +  
 labs(title = "Diferença Percentual Média de Salários por Ano com IC (95%)", x = "Ano", y = "Diferença percentual média (%)") +  
 theme\_minimal()  
print(grafico\_ic)  
ggsave("grafico\_ic\_diferenca\_percentual\_por\_ano.png", plot = grafico\_ic, width = 10, height = 6, dpi = 300)  
  
df\_global\_pares <- df\_global\_pares %>%  
 mutate(Ano = as.factor(Ano))  
  
grafico\_boxplot <- ggplot(df\_global\_pares, aes(x = Ano, y = diff\_log)) +  
 geom\_boxplot(fill = "lightblue", color = "black") +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(title = "Distribuição da diferença logarítmica dos salários por ano", x = "Ano", y = "Diferença logarítmica") +  
 theme\_minimal()  
print(grafico\_boxplot)  
ggsave("boxplot\_diferenca\_log\_por\_ano.png", plot = grafico\_boxplot, width = 10, height = 6, dpi = 300)  
  
  
# Preparando para comparanção de modelos e ajuste  
cat("\n\nPreparando objetos e calculando ajuste do modelo...\n")  
  
balance\_list\_ano <- balance\_list  
ttest\_list\_ano <- ttest\_list  
pares\_global\_ano <- pares\_global  
ttest\_summary\_df\_ano <- ttest\_summary\_df  
df\_global\_pares\_ano <- df\_global\_pares  
smd\_summary\_df\_ano <- smd\_summary\_df  
  
calcular\_ajuste\_propensity <- function(dados, formula) {  
 dados\_limpos <- dados %>% drop\_na(all\_of(all.vars(formula)))  
   
 psmodel <- glm(formula, family = binomial(), data = dados\_limpos)  
 nullmodel <- glm(as.formula(paste(all.vars(formula)[1], "~ 1")), family = binomial(), data = dados\_limpos)  
   
 aic <- AIC(psmodel)  
 bic <- BIC(psmodel)  
 loglik <- as.numeric(logLik(psmodel))  
 pseudo\_r2 <- 1 - (logLik(psmodel) / logLik(nullmodel))  
   
 resultados <- data.frame(  
 AIC = round(aic, 2),  
 BIC = round(bic, 2),  
 LogLikelihood = round(loglik, 2),  
 Pseudo\_R2 = round(as.numeric(pseudo\_r2), 4)  
 )  
 return(resultados)  
}  
  
formula\_ano <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
resultado\_ajuste\_ano <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_ano)  
  
print("--- Métricas de Ajuste do Modelo de Propensity Score (Modelo por Ano) ---")  
print(resultado\_ajuste\_ano)  
  
# Adicionando para salvar em excel  
write\_xlsx(resultado\_ajuste\_ano, "metricas\_ajuste\_modelo\_ano.xlsx")  
  
  
cat("\n\n-- ANÁLISE COMPLETA FINALIZADA --\n")  
  
  
### Análise de Matching com Períodos da Pandemia ---------------------------------------  
  
# Carregando pacotes  
library(Matching)  
library(tableone)  
library(writexl)  
library(dplyr)  
library(ggplot2)  
library(tidyr) # Necessário para a função drop\_na()  
  
# Preparando dados  
df <- pnad\_Filtrada  
  
# Garante que a variável Pandemia seja um fator com a ordem correta  
df <- df %>%  
 mutate(Pandemia = factor(Pandemia, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia")))  
  
# Obtendo os períodos únicos da pandemia para o loop  
groups <- df %>% distinct(Pandemia) %>% arrange(Pandemia)  
print(groups)  
  
# Define as covariáveis para o modelo de propensão  
xvars <- c(  
 "V2007\_Mulher", "V2009", "V2010\_grupo\_PP",  
 "V2005\_recode\_Pessoa\_responsável", "V2005\_recode\_Cônjuge",  
 "grupo\_moradia\_Coletivo", "VD3004\_nivel\_Escol\_Fundamental\_Completo",  
 "VD3004\_nivel\_Escol\_Médio\_Completo", "VD3004\_nivel\_Escol\_Superior\_Completo",  
 "Regiao\_Nordeste", "Regiao\_Norte",  
 "Regiao\_Sudeste", "Regiao\_Sul", "V1023\_recode3\_Capital\_e\_Metropolitana",  
 "V4029\_Sim", "V4040\_recode\_Mais\_de\_2\_anos", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4025\_Não"  
)  
  
# Inicializa listas para armazenar os resultados de cada iteração  
balance\_list <- list()  
ttest\_list <- list()  
pares\_global <- list()  
dados\_salvos\_para\_analise\_final <- list() # Lista para análise pós-loop  
  
  
# Loop principal de Matching por períodos da pandemia  
for (i in 1:nrow(groups)) {  
   
 current\_period <- as.character(groups$Pandemia[i])  
   
 cat("Processando Período da Pandemia =", current\_period, "\n")  
   
 # Filtra os dados para o período atual  
 mydata\_subset <- df %>% filter(Pandemia == current\_period)  
   
 # Remove explicitamente linhas com NAs nas variáveis do modelo para evitar erros  
 mydata\_subset\_clean <- mydata\_subset %>%  
 drop\_na(turismo\_dummy, all\_of(xvars))  
   
 cat("Observações iniciais:", nrow(mydata\_subset), " | Observações após limpeza de NAs:", nrow(mydata\_subset\_clean), "\n")  
   
   
   
 # Estima o propensity score  
 formula <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
 psmodel <- glm(formula, family = binomial(), data = mydata\_subset\_clean)  
   
 mydata\_subset\_clean$pscore <- psmodel$fitted.values  
 mydata\_subset\_clean$pscore <- pmin(pmax(mydata\_subset\_clean$pscore, 1e-6), 1 - 1e-6)  
   
 # Define e aplica o common support  
 pscore\_t <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 1]  
 pscore\_c <- mydata\_subset\_clean$pscore[mydata\_subset\_clean$turismo\_dummy == 0]  
 min\_common <- max(min(pscore\_t), min(pscore\_c))  
 max\_common <- min(max(pscore\_t), max(pscore\_c))  
   
 mydata\_subset\_cs <- mydata\_subset\_clean %>%  
 filter(pscore >= min\_common & pscore <= max\_common)  
   
 cat("Observações dentro do common support:", nrow(mydata\_subset\_cs), "\n")  
   
 # Realiza o matching  
 psmatch <- Match(  
 Tr = mydata\_subset\_cs$turismo\_dummy,  
 M = 1,  
 X = log(mydata\_subset\_cs$pscore / (1 - mydata\_subset\_cs$pscore)),  
 replace = FALSE,  
 caliper = 0.2  
 )  
   
 # Avaliação de balanceamento (original)  
 balance\_check <- MatchBalance(formula, data = mydata\_subset\_cs, match.out = psmatch, nboots = 500)  
 print(balance\_check)  
   
 # Seleciona os pares encontrados, removendo NAs de pares não formados  
 indices\_pareados <- unlist(psmatch[c("index.treated", "index.control")])  
 indices\_limpos <- na.omit(indices\_pareados)  
 matched <- mydata\_subset\_cs[indices\_limpos, ]  
   
 # Filtra por Salario\_Hora > 0  
 matched <- matched %>% filter(Salario\_Hora > 0)  
   
 # Tabela de balanceamento pós-match (original)  
 if (nrow(matched) > 0 && length(unique(matched$turismo\_dummy)) > 1) {  
 table1 <- CreateTableOne(vars = xvars, strata = "turismo\_dummy", data = matched, test = FALSE)  
 table1\_df <- as.data.frame(print(table1, smd = TRUE))  
 table1\_df$Pandemia <- current\_period  
 table1\_df$Variable <- rownames(table1\_df)  
   
 filename\_table <- paste0("matchedtab1\_pandemia\_", current\_period, ".xlsx")  
 write\_xlsx(table1\_df, filename\_table)  
 balance\_list[[current\_period]] <- table1\_df  
 }  
   
 # Análise de salários (original)  
 y\_trt <- matched$Salario\_Hora[matched$turismo\_dummy == 1]  
 y\_con <- matched$Salario\_Hora[matched$turismo\_dummy == 0]  
   
 cat("Tamanho dos grupos pareados: Tratados =", length(y\_trt), "Controles =", length(y\_con), "\n")  
   
 if (length(y\_trt) == length(y\_con) && length(y\_trt) > 0) {  
 diff\_log <- log(y\_trt) - log(y\_con)  
 t\_test\_result <- t.test(diff\_log, mu = 0)  
   
 media\_perc <- (exp(mean(diff\_log)) - 1) \* 100  
   
 t\_test\_df <- data.frame(  
 Pandemia = current\_period,  
 Estatistica\_t = t\_test\_result$statistic,  
 Graus\_de\_Liberdade = t\_test\_result$parameter,  
 p\_valor = t\_test\_result$p.value,  
 Media\_diferenca\_log = mean(diff\_log),  
 Media\_diferenca\_percentual\_aprox = media\_perc,  
 Conf\_Inf\_Lower = t\_test\_result$conf.int[1],  
 Conf\_Inf\_Upper = t\_test\_result$conf.int[2],  
 Hipotese\_Alternativa = t\_test\_result$alternative  
 )  
   
 filename\_ttest <- paste0("t\_test\_result\_pandemia\_", current\_period, ".xlsx")  
 write\_xlsx(t\_test\_df, filename\_ttest)  
 ttest\_list[[current\_period]] <- t\_test\_df  
   
 pares\_df <- data.frame(  
 Pandemia = current\_period,  
 salario\_turismo = y\_trt,  
 salario\_controle = y\_con,  
 diff\_log = diff\_log,  
 diff\_percentual\_aprox = (exp(diff\_log) - 1) \* 100  
 )  
   
 pares\_global[[current\_period]] <- pares\_df  
   
 # Gráfico de densidade do propensity score (CORRIGIDO)  
 df\_plot <- mydata\_subset\_cs  
 df\_plot$logit\_pscore <- log(df\_plot$pscore / (1 - df\_plot$pscore))  
 df\_plot$grupo <- ifelse(df\_plot$turismo\_dummy == 1, "Turismo (Tratado)", "Controle")  
   
 # Adicionando Cálculo do caliper para as linhas pontilhadas  
 caliper\_limite <- 0.2  
 media\_t <- mean(df\_plot$logit\_pscore[df\_plot$turismo\_dummy == 1], na.rm = TRUE)  
 lim\_inf <- media\_t - caliper\_limite  
 lim\_sup <- media\_t + caliper\_limite  
   
 g <- ggplot(df\_plot, aes(x = logit\_pscore, fill = grupo)) +  
 geom\_density(alpha = 0.4) +  
 # ADICIONADO: Camada para desenhar as linhas do caliper  
 geom\_vline(xintercept = c(lim\_inf, lim\_sup), linetype = "dashed", color = "red") +  
 labs(  
 title = paste("Distribuição do Logit do Propensity Score -", current\_period),  
 x = "Logit do Propensity Score",  
 y = "Densidade"  
 ) +  
 scale\_fill\_manual(values = c("Turismo (Tratado)" = "#1b9e77", "Controle" = "#d95f02")) +  
 theme\_minimal()  
   
 print(g)  
 filename\_grafico <- paste0("logit\_pscore\_densidade\_pandemia\_", current\_period, ".png")  
 ggsave(filename\_grafico, plot = g, width = 9, height = 5)  
   
 } else {  
 cat("Não é possível realizar o teste t pareado para este grupo.\n")  
 }  
   
 # Salva os dados necessários para a análise final  
 if (exists("psmatch")) {  
 dados\_salvos\_para\_analise\_final[[current\_period]] <- list(  
 dados\_com\_pscore = mydata\_subset\_clean,  
 dados\_pos\_cs = mydata\_subset\_cs,  
 objeto\_match = psmatch,  
 variaveis\_x = xvars  
 )  
 }  
   
 cat("Processo concluído para o período =", current\_period, "\n\n")  
 rm(mydata\_subset, mydata\_subset\_clean, mydata\_subset\_cs, psmodel, psmatch, matched)  
 gc()  
}  
  
  
  
# Análise de pós-loop: Observações excluídas e gráficos SMD  
cat("\n\nIniciando análises pós-execução (Contagem Common Support e Gráficos SMD)...\n")  
  
lista\_stats\_cs\_final <- list()  
  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
   
 cat("Processando análises para o período:", periodo, "\n")  
   
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_inicial\_limpo <- dados\_periodo$dados\_com\_pscore  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
   
 # Funcionalidade 1: Contagem de Observações Excluídas --  
 n\_antes\_cs <- nrow(df\_inicial\_limpo)  
 n\_depois\_cs <- nrow(df\_depois\_cs)  
 n\_excluidas <- n\_antes\_cs - n\_depois\_cs  
   
 stats\_cs\_df <- data.frame(  
 Periodo = periodo,  
 Obs\_Iniciais\_Limpas = n\_antes\_cs,  
 Obs\_Excluidas\_CS = n\_excluidas,  
 Obs\_Mantidas\_CS = n\_depois\_cs,  
 Perc\_Excluido = round((n\_excluidas / n\_antes\_cs) \* 100, 2)  
 )  
 lista\_stats\_cs\_final[[periodo]] <- stats\_cs\_df  
   
 # Funcionalidade 2: Gráfico Comparativo de SMDs --  
   
 contagem\_grupos\_cs <- table(df\_depois\_cs$turismo\_dummy)  
 if (length(contagem\_grupos\_cs) < 2 || any(contagem\_grupos\_cs == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos após o common support. Gráfico não será gerado.\n")  
 next  
 }  
   
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
   
 contagem\_grupos\_pareado <- table(df\_pareado$turismo\_dummy)  
 if (length(contagem\_grupos\_pareado) < 2 || any(contagem\_grupos\_pareado == 0)) {  
 cat("--- AVISO: Período:", periodo, "não possui ambos os grupos APÓS O MATCHING. Gráfico não será gerado.\n")  
 next   
 }  
   
 df\_depois\_cs$turismo\_dummy <- as.factor(df\_depois\_cs$turismo\_dummy)  
 df\_pareado$turismo\_dummy <- as.factor(df\_pareado$turismo\_dummy)  
   
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs), test = FALSE)  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado), test = FALSE)  
   
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 smd\_antes <- smd\_df\_antes[xvars\_usadas, "SMD"]  
 smd\_depois <- smd\_df\_depois[xvars\_usadas, "SMD"]  
 names(smd\_antes) <- xvars\_usadas  
 names(smd\_depois) <- xvars\_usadas  
   
 df\_smd\_plot <- data.frame(  
 Variable = names(smd\_antes),  
 SMD = c(smd\_antes, smd\_depois),  
 Status = factor(rep(c("Antes do Pareamento", "Depois do Pareamento"), each = length(smd\_antes)),  
 levels = c("Antes do Pareamento", "Depois do Pareamento"))  
 )  
   
 love\_plot <- ggplot(df\_smd\_plot, aes(x = SMD, y = reorder(Variable, SMD), color = Status, shape = Status)) +  
 geom\_point(size = 3.5, alpha = 0.8) +  
 geom\_vline(xintercept = 0, linetype = "solid", color = "black") +  
 geom\_vline(xintercept = c(-0.1, 0.1), linetype = "dashed", color = "grey50") +  
 labs(  
 title = paste("Balanceamento das Covariáveis -", periodo),  
 subtitle = "Diferença Média Padronizada (SMD)",  
 x = "SMD",  
 y = "Covariável",  
 color = "Status",  
 shape = "Status"  
 ) +  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "blue")) +  
 theme\_minimal(base\_size = 12) +  
 theme(legend.position = "bottom", plot.title = element\_text(hjust = 0.5), plot.subtitle = element\_text(hjust = 0.5))  
   
 filename\_love\_plot <- paste0("GRAFICO\_SMD\_LOVEPLOT\_pandemia\_", periodo, ".png")  
 ggsave(filename\_love\_plot, plot = love\_plot, width = 9, height = 10, dpi = 300)  
 cat("Gráfico SMD salvo em:", filename\_love\_plot, "\n")  
}  
  
# Conslidação dos resultados  
# Salva a contagem de observações excluídas --  
summary\_common\_support\_final <- do.call(rbind, lista\_stats\_cs\_final)  
rownames(summary\_common\_support\_final) <- NULL  
print("-----------------------------------------------------------")  
print("Resumo Final das Observações Excluídas no Common Support (por Período Pandêmico):")  
print(summary\_common\_support\_final)  
write\_xlsx(summary\_common\_support\_final, "resumo\_final\_exclusoes\_common\_support\_pandemia.xlsx")  
cat("\nResumo das exclusões salvo em: resumo\_final\_exclusoes\_common\_support\_pandemia.xlsx\n")  
  
# Gráfico de Barras da Média de SMDs por Período da Pandemia --  
smd\_evolution\_list <- list()  
for (periodo in names(dados\_salvos\_para\_analise\_final))   
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
match\_salvo <- dados\_periodo$objeto\_match  
xvars\_usadas <- dados\_periodo$variaveis\_x  
indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
indices\_limpos <- na.omit(indices\_pareados)  
df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
if (length(unique(df\_depois\_cs$turismo\_dummy)) < 2 || length(unique(df\_pareado$turismo\_dummy)) < 2) next  
  
tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs))  
tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado))  
smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
  
media\_smd\_antes <- mean(abs(as.numeric(smd\_df\_antes[xvars\_usadas, "SMD"])), na.rm = TRUE)  
media\_smd\_depois <- mean(abs(as.numeric(smd\_df\_depois[xvars\_usadas, "SMD"])), na.rm = TRUE)  
  
smd\_evolution\_list <- list()  
for (periodo in names(dados\_salvos\_para\_analise\_final)) {  
 dados\_periodo <- dados\_salvos\_para\_analise\_final[[periodo]]  
 df\_depois\_cs <- dados\_periodo$dados\_pos\_cs  
 match\_salvo <- dados\_periodo$objeto\_match  
 xvars\_usadas <- dados\_periodo$variaveis\_x  
 indices\_pareados <- c(match\_salvo$index.treated, match\_salvo$index.control)  
 indices\_limpos <- na.omit(indices\_pareados)  
 df\_pareado <- df\_depois\_cs[indices\_limpos, ]  
 if (length(unique(df\_depois\_cs$turismo\_dummy)) < 2 || length(unique(df\_pareado$turismo\_dummy)) < 2) next  
   
 tabela\_antes <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_depois\_cs))  
 tabela\_depois <- CreateTableOne(vars = xvars\_usadas, strata = "turismo\_dummy", data = as.data.frame(df\_pareado))  
 smd\_df\_antes <- as.data.frame(print(tabela\_antes, smd = TRUE))  
 smd\_df\_depois <- as.data.frame(print(tabela\_depois, smd = TRUE))  
   
 media\_smd\_antes <- mean(abs(as.numeric(smd\_df\_antes[xvars\_usadas, "SMD"])), na.rm = TRUE)  
 media\_smd\_depois <- mean(abs(as.numeric(smd\_df\_depois[xvars\_usadas, "SMD"])), na.rm = TRUE)  
   
 smd\_evolution\_list[[periodo]] <- data.frame(Periodo = periodo, Media\_SMD\_Antes = media\_smd\_antes, Media\_SMD\_Depois = media\_smd\_depois)  
}  
smd\_evolution\_df <- do.call(rbind, smd\_evolution\_list)  
smd\_evolution\_df$Periodo <- factor(smd\_evolution\_df$Periodo, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia"))  
  
smd\_long\_df <- smd\_evolution\_df %>%  
 pivot\_longer(cols = c("Media\_SMD\_Antes", "Media\_SMD\_Depois"), names\_to = "Status", values\_to = "Media\_SMD") %>%  
 mutate(Status = ifelse(Status == "Media\_SMD\_Antes", "Antes do Pareamento", "Depois do Pareamento"))  
  
# Alterado de geom\_bar para geom\_line + geom\_point  
grafico\_smd\_comparativo\_pandemia <- ggplot(smd\_long\_df, aes(x = Periodo, y = Media\_SMD, group = Status, color = Status)) +  
 geom\_line(linewidth = 1.2) +  
 geom\_point(size = 3) +  
 geom\_text(aes(label = round(Media\_SMD, 3)), vjust = -1.5, size = 3.5, show.legend = FALSE) + # Adiciona rótulos de texto  
 geom\_hline(yintercept = 0.1, linetype = "dashed", color = "red") +  
 scale\_color\_manual(values = c("Antes do Pareamento" = "orange", "Depois do Pareamento" = "steelblue")) +  
 # Expande um pouco o limite do eixo Y para os rótulos caberem  
 ylim(0, max(smd\_long\_df$Media\_SMD) \* 1.15) +   
 labs(  
 title = "Média do SMD Absoluto por Período da Pandemia",   
 subtitle = "Antes e Depois do Pareamento",   
 x = "Período",   
 y = "Média do SMD Absoluto",   
 color = "Situação"  
 ) +  
 theme\_minimal(base\_size = 14) +   
 theme(  
 legend.position = "bottom",   
 plot.title = element\_text(hjust = 0.5, face = "bold"),   
 plot.subtitle = element\_text(hjust = 0.5)  
 )  
  
print(grafico\_smd\_comparativo\_pandemia)  
ggsave("grafico\_smd\_comparativo\_pandemia\_linha.png", plot = grafico\_smd\_comparativo\_pandemia, width = 10, height = 7, dpi = 300)  
  
# Início da sua seção de análise original --  
ttest\_summary\_df <- do.call(rbind, ttest\_list)  
print(head(ttest\_summary\_df))  
write\_xlsx(ttest\_summary\_df, "resumo\_t\_test\_log\_por\_pandemia.xlsx")  
  
df\_global\_pares <- do.call(rbind, pares\_global)  
write\_xlsx(df\_global\_pares, "resultado\_global\_pares\_log\_pandemia.xlsx")  
  
teste\_global <- t.test(df\_global\_pares$diff\_log, mu = 0)  
print(teste\_global)  
  
teste\_global\_df <- data.frame(  
 Estatistica\_t = teste\_global$statistic,  
 Graus\_de\_Liberdade = teste\_global$parameter,  
 p\_valor = teste\_global$p.value,  
 Media\_diferenca\_log = mean(df\_global\_pares$diff\_log),  
 Media\_diferenca\_percentual\_aprox = (exp(mean(df\_global\_pares$diff\_log)) - 1) \* 100,  
 Conf\_Inf\_Lower = teste\_global$conf.int[1],  
 Conf\_Inf\_Upper = teste\_global$conf.int[2],  
 Hipotese\_Alternativa = teste\_global$alternative  
)  
write\_xlsx(teste\_global\_df, "resultado\_t\_test\_global\_pandemia.xlsx")  
  
smd\_summary\_df <- do.call(rbind, balance\_list)  
smd\_summary\_df <- dplyr::select(smd\_summary\_df, Pandemia, Variable, SMD)  
print(head(smd\_summary\_df))  
write\_xlsx(smd\_summary\_df, "resumo\_balanceamento\_smds\_por\_pandemia.xlsx")  
  
  
# Gráfico de Diferença Percentual Média de Salários por Pandemia   
  
ttest\_summary\_df <- ttest\_summary\_df %>%  
 mutate(  
 Pandemia = factor(Pandemia, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia")),  
 significativo = ifelse(p\_valor < 0.05, TRUE, FALSE)  
 )  
  
# ## CÓDIGO DO GRÁFICO ATUALIZADO PARA O ESTILO FINAL ##  
grafico\_linha\_diferenca <- ggplot(ttest\_summary\_df, aes(x = Pandemia, y = Media\_diferenca\_percentual\_aprox, group = 1)) +  
 # Linha azul conectando os pontos  
 geom\_line(color = "blue", linewidth = 1) +  
 # Pontos coloridos por significância  
 geom\_point(aes(color = significativo), size = 3.5) +  
 # Escala de cores manual (preto/vermelho) com drop = FALSE para sempre mostrar a legenda completa  
 scale\_color\_manual(values = c("FALSE" = "black", "TRUE" = "red"),   
 name = "Significativo (p < 0.05)",   
 drop = FALSE) +  
 # Linha de referência no zero  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Diferença Percentual Média de Salários (Turismo vs. Outros)",  
 subtitle = "Períodos com p < 0.05 destacados em vermelho",  
 x = "Período",  
 y = "Diferença Percentual Média (%)"  
 ) +  
 theme\_minimal(base\_size = 14)  
  
print(grafico\_linha\_diferenca)  
ggsave("grafico\_diferenca\_percentual\_por\_pandemia\_linha.png", plot = grafico\_linha\_diferenca, width = 10, height = 6, dpi = 300)  
  
# Boxplot: Diferença logarítmica por Pandemia  
df\_global\_pares <- df\_global\_pares %>%  
 mutate(Pandemia = factor(Pandemia, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia")))  
  
grafico\_boxplot <- ggplot(df\_global\_pares, aes(x = Pandemia, y = diff\_log, fill = Pandemia)) +  
 geom\_boxplot() +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Distribuição da Diferença Logarítmica dos Salários por Período",  
 x = "Período da Pandemia",  
 y = "Diferença Logarítmica"  
 ) +  
 theme\_minimal() +  
 theme(legend.position = "none")  
print(grafico\_boxplot)  
ggsave("boxplot\_diferenca\_log\_por\_pandemia.png", plot = grafico\_boxplot, width = 8, height = 6, dpi = 300)  
  
  
# Gráfico de Intervalo de Confiança (95%) por Período da Pandemia  
  
cat("\nGerando gráfico de diferença salarial com Intervalo de Confiança...\n")  
  
# Calcula os limites do intervalo de confiança em percentual  
ttest\_summary\_df\_ic <- ttest\_summary\_df %>%  
 mutate(  
 Pandemia = factor(Pandemia, levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia")),  
 IC\_inferior\_perc = (exp(Conf\_Inf\_Lower) - 1) \* 100,  
 IC\_superior\_perc = (exp(Conf\_Inf\_Upper) - 1) \* 100  
 )  
  
# Gera o gráfico  
grafico\_ic\_pandemia <- ggplot(ttest\_summary\_df\_ic, aes(x = Pandemia, y = Media\_diferenca\_percentual\_aprox, group = 1)) +  
 # A faixa sombreada para o intervalo de confiança  
 geom\_ribbon(aes(ymin = IC\_inferior\_perc, ymax = IC\_superior\_perc),  
 fill = "steelblue", alpha = 0.2) +  
 # A linha para a média  
 geom\_line(color = "steelblue", linewidth = 1.2) +  
 geom\_point(color = "steelblue", size = 3) +  
 # Linha de referência no zero  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Diferença Percentual Média de Salários por Período da Pandemia",  
 subtitle = "Com intervalo de confiança (95%)",  
 x = "Período",  
 y = "Diferença Percentual Média (%)"  
 ) +  
 theme\_minimal(base\_size = 14)  
  
print(grafico\_ic\_pandemia)  
ggsave("grafico\_ic\_diferenca\_percentual\_por\_pandemia.png", plot = grafico\_ic\_pandemia, width = 10, height = 7, dpi = 300)  
  
cat("\nGráfico com Intervalo de Confiança salvo como 'grafico\_ic\_diferenca\_percentual\_por\_pandemia.png'\n")  
  
  
# Comparando sa há diferença esttística entre resultdo dos períodos períodos- pré, durante e após pandemia  
# Certificando-se de que 'Antes\_Pandemia' seja a categoria base  
df\_global\_pares$Pandemia <- factor(df\_global\_pares$Pandemia,   
 levels = c("Antes\_Pandemia", "Durante\_Pandemia", "Apos\_Pandemia"))  
  
# Rodando um modelo de regressão para comparar as médias das diferenças logarítmicas por período  
# Este é um modelo ANOVA, que testará a significância das diferenças entre os grupos de período  
model\_diferenca\_entre\_periodos <- lm(diff\_log ~ Pandemia, data = df\_global\_pares)  
  
# Exibindo os resultados  
summary(model\_diferenca\_entre\_periodos)  
  
# Guardando resultados comparação períodos  
# Extraindo a tabela de coeficientes para um dataframe  
resultados\_df <- as.data.frame(coef(summary(model\_diferenca\_entre\_periodos)))  
  
# Adicionando os nomes das variáveis como uma coluna  
resultados\_df$Variavel <- rownames(resultados\_df)  
  
# Salvando o dataframe em um arquivo Excel  
write\_xlsx(resultados\_df, "resultado\_comparacao\_entre\_periodos.xlsx")  
  
# Opção para salvar em formato mais bonito  
# install.packages("modelsummary")  
library(modelsummary)  
  
# Apenas para visualizar a tabela no RStudio  
modelsummary(model\_diferenca\_entre\_periodos)  
  
# Para salvar a tabela em um arquivo HTML ou Word  
modelsummary(model\_diferenca\_entre\_periodos, output = "tabela\_modelo\_final.html")  
  
  
# Preparação para comparação de modelos e ajuste  
cat("\n\nPreparando objetos e calculando ajuste do modelo...\n")  
  
balance\_list\_pandemia <- balance\_list  
ttest\_list\_pandemia <- ttest\_list  
pares\_global\_pandemia <- pares\_global  
ttest\_summary\_df\_pandemia <- ttest\_summary\_df  
df\_global\_pares\_pandemia <- df\_global\_pares  
smd\_summary\_df\_pandemia <- smd\_summary\_df  
  
# Função para calcular indicadores do modelo de propensity score  
calcular\_ajuste\_propensity <- function(dados, formula) {  
 dados\_limpos <- dados %>% drop\_na(all\_of(all.vars(formula)))  
   
 psmodel <- glm(formula, family = binomial(), data = dados\_limpos)  
 nullmodel <- glm(as.formula(paste(all.vars(formula)[1], "~ 1")), family = binomial(), data = dados\_limpos)  
   
 aic <- AIC(psmodel)  
 bic <- BIC(psmodel)  
 loglik <- as.numeric(logLik(psmodel))  
 pseudo\_r2 <- 1 - (logLik(psmodel) / logLik(nullmodel))  
   
 resultados <- data.frame(  
 AIC = round(aic, 2),  
 BIC = round(bic, 2),  
 LogLikelihood = round(loglik, 2),  
 Pseudo\_R2 = round(as.numeric(pseudo\_r2), 4)  
 )  
 return(resultados)  
}  
  
formula\_pandemia <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
resultado\_ajuste\_pandemia <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_pandemia)  
  
print("--- Métricas de Ajuste do Modelo de Propensity Score (Modelo Pandemia) ---")  
print(resultado\_ajuste\_pandemia)  
  
# Adição para salvar em excel  
write\_xlsx(resultado\_ajuste\_pandemia, "metricas\_ajuste\_modelo\_pandemia.xlsx")  
  
cat("\n\n-- ANÁLISE COMPLETA FINALIZADA --\n")  
  
  
  
# Comparação final dos 3 modelos de Matching  
  
# Carregando pacotes  
library(dplyr)  
library(ggplot2)  
library(writexl)  
library(tidyr) # Para drop\_na()  
  
# Funções de apoio  
  
# Função para calcular a média dos SMDs de um dataframe de resumo  
calcular\_media\_smd <- function(smd\_summary\_df) {  
 smd\_summary\_df$SMD <- as.numeric(smd\_summary\_df$SMD)  
 mean(smd\_summary\_df$SMD, na.rm = TRUE)  
}  
  
# Função para calcular a proporção de observações pareadas  
# Nota: Esta função é uma aproximação, pois não considera as exclusões em cada etapa.  
# Uma métrica mais precisa é a contagem de exclusão que já foi salva.  
  
calcular\_proporcao\_pareados <- function(df\_global\_pares, total\_obs) {  
 total\_pares <- nrow(df\_global\_pares)  
 perc\_pareado <- (total\_pares \* 2) / total\_obs \* 100  
 return(perc\_pareado)  
}  
  
# Função robusta para calcular o ajuste do modelo de propensity score  
calcular\_ajuste\_propensity <- function(dados, formula) {  
 dados\_limpos <- dados %>% drop\_na(all\_of(all.vars(formula)))  
   
 psmodel <- glm(formula, family = binomial(), data = dados\_limpos)  
 nullmodel <- glm(as.formula(paste(all.vars(formula)[1], "~ 1")), family = binomial(), data = dados\_limpos)  
   
 aic <- AIC(psmodel)  
 bic <- BIC(psmodel)  
 loglik <- as.numeric(logLik(psmodel))  
 pseudo\_r2 <- 1 - (logLik(psmodel) / logLik(nullmodel))  
   
 resultados <- data.frame(  
 AIC = round(aic, 2),  
 BIC = round(bic, 2),  
 LogLikelihood = round(loglik, 2),  
 Pseudo\_R2 = round(as.numeric(pseudo\_r2), 4)  
 )  
 return(resultados)  
}  
  
  
# Cálculo das métricas de ajuste para cada modelo  
cat("Calculando métricas de ajuste para os 3 modelos...\n")  
  
# Re-define as covariáveis para a função  
xvars <- c(  
 "V2007\_Mulher", "V2009", "V2010\_grupo\_PP",  
 "V2005\_recode\_Pessoa\_responsável", "V2005\_recode\_Cônjuge",  
 "grupo\_moradia\_Coletivo", "VD3004\_nivel\_Escol\_Fundamental\_Completo",  
 "VD3004\_nivel\_Escol\_Médio\_Completo", "VD3004\_nivel\_Escol\_Superior\_Completo",  
 "Regiao\_Nordeste", "Regiao\_Norte",  
 "Regiao\_Sudeste", "Regiao\_Sul", "V1023\_recode3\_Capital\_e\_Metropolitana",  
 "V4029\_Sim", "V4040\_recode\_Mais\_de\_2\_anos", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4025\_Não"  
)  
  
# A fórmula é a mesma para todos os modelos  
formula\_ps <- as.formula(paste("turismo\_dummy ~", paste(xvars, collapse = " + ")))  
  
# Calcula o ajuste para cada modelo  
# (Assumindo que pnad\_Filtrada está carregado)  
resultado\_ajuste\_trimestre <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_ps)  
resultado\_ajuste\_ano <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_ps)  
resultado\_ajuste\_pandemia <- calcular\_ajuste\_propensity(pnad\_Filtrada, formula\_ps)  
  
# Monta a tabela comparativa de ajuste  
tabela\_ajuste\_modelos <- rbind(  
 cbind(Modelo = "Ano + Trimestre", resultado\_ajuste\_trimestre),  
 cbind(Modelo = "Ano", resultado\_ajuste\_ano),  
 cbind(Modelo = "Pandemia", resultado\_ajuste\_pandemia)  
)  
  
print("--- Tabela Comparativa de Ajuste dos Modelos de Propensity Score ---")  
print(tabela\_ajuste\_modelos)  
write\_xlsx(tabela\_ajuste\_modelos, "tabela\_comparativa\_ajuste\_propensity\_scores.xlsx")  
  
  
# Cálculo das métricas de resultado do Matching  
cat("\nCalculando métricas de resultado do matching...\n")  
  
total\_obs <- nrow(pnad\_Filtrada)  
  
# Métricas para o modelo Ano + Trimestre  
media\_smd\_trimestre <- calcular\_media\_smd(smd\_summary\_df\_trimestre)  
perc\_pareado\_trimestre <- calcular\_proporcao\_pareados(df\_global\_pares\_trimestre, total\_obs)  
media\_dif\_trimestre <- mean(ttest\_summary\_df\_trimestre$Media\_diferenca\_percentual\_aprox, na.rm = TRUE)  
  
# Métricas para o modelo Ano  
media\_smd\_ano <- calcular\_media\_smd(smd\_summary\_df\_ano)  
perc\_pareado\_ano <- calcular\_proporcao\_pareados(df\_global\_pares\_ano, total\_obs)  
media\_dif\_ano <- mean(ttest\_summary\_df\_ano$Media\_diferenca\_percentual\_aprox, na.rm = TRUE)  
  
# Métricas para o modelo Pandemia  
media\_smd\_pandemia <- calcular\_media\_smd(smd\_summary\_df\_pandemia)  
perc\_pareado\_pandemia <- calcular\_proporcao\_pareados(df\_global\_pares\_pandemia, total\_obs)  
media\_dif\_pandemia <- mean(ttest\_summary\_df\_pandemia$Media\_diferenca\_percentual\_aprox, na.rm = TRUE)  
  
# Monta a tabela comparativa de resultados  
tabela\_comparativa <- data.frame(  
 Modelo = c("Ano + Trimestre", "Ano", "Pandemia"),  
 Media\_SMD\_Apos\_Match = round(c(media\_smd\_trimestre, media\_smd\_ano, media\_smd\_pandemia), 4),  
 Perc\_Obs\_Pareadas\_Aprox = round(c(perc\_pareado\_trimestre, perc\_pareado\_ano, perc\_pareado\_pandemia), 2),  
 Media\_Dif\_Salarial\_Perc = round(c(media\_dif\_trimestre, media\_dif\_ano, media\_dif\_pandemia), 2)  
)  
  
print("--- Tabela Comparativa de Resultados do Matching ---")  
print(tabela\_comparativa)  
write\_xlsx(tabela\_comparativa, "tabela\_comparativa\_resultados\_matching.xlsx")  
  
  
# Gráficos Comparativos  
cat("\nGerando gráficos comparativos...\n")  
  
# Gráfico 1: Média dos SMDs  
grafico\_comp\_smd <- ggplot(tabela\_comparativa, aes(x = Modelo, y = Media\_SMD\_Apos\_Match, fill = Modelo)) +  
 geom\_bar(stat = "identity", color = "black") +  
 geom\_text(aes(label = round(Media\_SMD\_Apos\_Match, 4)), vjust = -0.5) +  
 labs(  
 title = "Média dos SMDs após o Matching",  
 subtitle = "Comparação entre Modelos",  
 x = "Modelo", y = "Média dos SMDs"  
 ) +  
 theme\_minimal(base\_size = 12) +  
 theme(legend.position = "none")  
  
print(grafico\_comp\_smd)  
ggsave("grafico\_comparativo\_smd.png", plot = grafico\_comp\_smd, width = 8, height = 6, dpi = 300)  
  
# Gráfico 2: Diferença Percentual Média dos Salários  
grafico\_comp\_dif <- ggplot(tabela\_comparativa, aes(x = Modelo, y = Media\_Dif\_Salarial\_Perc, fill = Modelo)) +  
 geom\_bar(stat = "identity", color = "black") +  
 geom\_text(aes(label = paste0(round(Media\_Dif\_Salarial\_Perc, 2), "%")), vjust = 1.5, color="white", size=5) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray40") +  
 labs(  
 title = "Diferença Percentual Média de Salários (Turismo vs Outros)",  
 subtitle = "Comparação entre Modelos",  
 x = "Modelo", y = "Diferença Percentual Média (%)"  
 ) +  
 theme\_minimal(base\_size = 12) +  
 theme(legend.position = "none")  
  
print(grafico\_comp\_dif)  
ggsave("grafico\_comparativo\_diferenca\_salarial.png", plot = grafico\_comp\_dif, width = 8, height = 6, dpi = 300)  
  
cat("\n\n-- COMPARAÇÃO DE MODELOS FINALIZADA --\n")  
  
  
## FILTRANDO APENAS EMPREGADOS ACTS PARA AS PRÓXIMAS ANÁLISES --------------------------------  
pnad\_tur\_Filtrada <- pnad\_Filtrada %>%  
 filter(turismo\_dummy == 1)  
  
# Corrigindo idade ao quadrado  
pnad\_tur\_Filtrada <- pnad\_tur\_Filtrada %>%  
 mutate(V2009\_quad = V2009^2)  
  
variable.names(pnad\_tur\_Filtrada)  
  
# Como data.frame base  
# pnad\_tur\_Filtrada <- as.data.frame(pnad\_tur\_Filtrada\_srvyr)  
  
  
## DECOMPOSIÇÃO DE OAXACA BLINDER THREEFOLD --------------------------------  
  
glimpse(pnad\_tur\_Filtrada)  
  
library(oaxaca)  
  
### Decomposição de Oaxaca Blinder por Gênero--------------------------------  
  
oaxaca\_result <- oaxaca(  
 formula = VD4016\_log\_hora ~ V2010\_grupo\_PP + V2009 + I(V2009^2) + VD3005\_cont +   
 VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior + V1023\_recode3\_Resto\_UF + Regiao\_Sul + Regiao\_Centro\_Oeste + Regiao\_Norte + Regiao\_Nordeste +  
 V4029\_Sim + V4025\_Não + V4039 + Horas\_trabalhadas2\_Integral + V4040\_recode\_Mais\_de\_2\_anos +  
 Ano\_2017 + Ano\_2018 + Ano\_2019 + Ano\_2020 + Ano\_2021 + Ano\_2022 + Ano\_2023 + Ano\_2024 + Trimestre\_2 + Trimestre\_3 + Trimestre\_4  
 | V2007\_Mulher,   
 data = pnad\_tur\_Filtrada,   
 reg.fun = lm,   
 R = 500, # Bootstrap para ICs  
 group.weights = c(0.5, 0.5) # Ponderação de coeficientes segundo Jann (2008)  
)  
  
  
# Visualizando os resultados  
print(oaxaca\_result)  
summary(oaxaca\_result)  
  
  
  
# Carregar as bibliotecas necessárias  
library(dplyr)  
library(tidyr)  
library(broom)  
library(writexl)  
  
# Definição dos Rótulos (deve estar no início do seu script)  
rotulos <- c(  
 `VD4016\_log\_hora` = "Log Salário por Hora",  
 `V2010\_grupo\_PP` = "Pretas e Pardas",  
 `V2009` = "Idade (anos)",  
 `I(V2009^2)` = "Idade² (anos²)",  
 `VD3005\_cont` = "Escolaridade (Anos)",  
 `VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior` = "Sem Ensino Superior",  
 `V1023\_recode3\_Resto\_UF` = "Resto da UF",  
 `Regiao\_Sul` = "Região Sul",  
 `Regiao\_Centro\_Oeste` = "Região Centro Oeste",  
 `Regiao\_Norte` = "Região Norte",  
 `Regiao\_Nordeste` = "Região Nordeste",  
 `Regiao\_Sudeste` = "Região Sudeste",  
 `V4029\_Sim` = "Carteira Assinada (Sim)",  
 `V4025\_Não` = "Contrato Temporário (Não)",  
 `V4039` = "Horas Semanais Trabalhadas",  
 `Horas\_trabalhadas2\_Integral` = "Integral",  
 `V4040\_recode\_Mais\_de\_2\_anos` = "Tempo de Emprego > 2 anos",  
 `V2007\_Mulher` = "Mulheres"  
)  
  
rotulos\_df <- tibble(  
 Variavel\_tecnica = names(rotulos),  
 Variavel = unname(rotulos)  
)  
  
# Criar os Modelos de Regressão Separados  
dados\_homens <- subset(pnad\_tur\_Filtrada, V2007\_Mulher == 0)  
dados\_mulheres <- subset(pnad\_tur\_Filtrada, V2007\_Mulher == 1)  
formula\_regressao <- VD4016\_log\_hora ~ V2010\_grupo\_PP + V2009 + I(V2009^2) + VD3005\_cont +   
 VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior + V1023\_recode3\_Resto\_UF + Regiao\_Sul + Regiao\_Centro\_Oeste + Regiao\_Norte + Regiao\_Nordeste +  
 V4029\_Sim + V4025\_Não + V4039 + Horas\_trabalhadas2\_Integral + V4040\_recode\_Mais\_de\_2\_anos +  
 Ano\_2017 + Ano\_2018 + Ano\_2019 + Ano\_2020 + Ano\_2021 + Ano\_2022 + Ano\_2023 + Ano\_2024 + Trimestre\_2 + Trimestre\_3 + Trimestre\_4  
modelo\_homens <- lm(formula\_regressao, data = dados\_homens)  
modelo\_mulheres <- lm(formula\_regressao, data = dados\_mulheres)  
  
  
# Extrair o RESUMO GERAL da decomposição  
overall\_raw <- summary(oaxaca\_result)$threefold$overall  
resumo\_geral <- tibble(  
 Componente = c("Endowments (Explicado)", "Coefficients (Não Explicado)", "Interaction"),  
 Valor = overall\_raw[c("coef(endowments)", "coef(coefficients)", "coef(interaction)")],  
 Erro\_Padrao = overall\_raw[c("se(endowments)", "se(coefficients)", "se(interaction)")]  
) %>%  
 mutate(  
 Estatistica\_t = Valor / Erro\_Padrao,  
 Valor\_p = 2 \* (1 - pnorm(abs(Estatistica\_t))),  
 Significancia = case\_when(  
 Valor\_p < 0.001 ~ "\*\*\*", Valor\_p < 0.01 ~ "\*\*", Valor\_p < 0.05 ~ "\*", Valor\_p < 0.1 ~ ".", TRUE ~ ""  
 )  
 )  
  
# Extrair os resultados das REGRESSÕES (`β`) com os rótulos  
regressao\_grupo\_A <- tidy(modelo\_homens) %>%  
 left\_join(rotulos\_df, by = c("term" = "Variavel\_tecnica")) %>%  
 mutate(Variavel = coalesce(Variavel, term)) %>%  
 mutate(Significancia = case\_when(  
 p.value < 0.001 ~ "\*\*\*", p.value < 0.01 ~ "\*\*", p.value < 0.05 ~ "\*", p.value < 0.1 ~ ".", TRUE ~ ""  
 )) %>%  
 # Correção aqui  
 dplyr::select(Variavel, Coeficiente\_beta = estimate, std.error, statistic, p.value, Significancia)  
  
regressao\_grupo\_B <- tidy(modelo\_mulheres) %>%  
 left\_join(rotulos\_df, by = c("term" = "Variavel\_tecnica")) %>%  
 mutate(Variavel = coalesce(Variavel, term)) %>%  
 mutate(Significancia = case\_when(  
 p.value < 0.001 ~ "\*\*\*", p.value < 0.01 ~ "\*\*", p.value < 0.05 ~ "\*", p.value < 0.1 ~ ".", TRUE ~ ""  
 )) %>%  
 # --- CORREÇÃO APLICADA AQUI ---  
 dplyr::select(Variavel, Coeficiente\_beta = estimate, std.error, statistic, p.value, Significancia)  
  
# Extrair a Decomposição detalhada com os rótulos  
contribuicoes\_detalhadas <- as.data.frame(oaxaca\_result$threefold$variables) %>%  
 mutate(Variavel\_tecnica = rownames(.)) %>%  
 pivot\_longer(cols = -Variavel\_tecnica, names\_to = c(".value", "Componente"), names\_pattern = "(coef|se)\\((.\*)\\)") %>%  
 rename(Contribuicao = coef, Erro\_Padrao = se) %>%  
 left\_join(rotulos\_df, by = "Variavel\_tecnica") %>%  
 mutate(Variavel = coalesce(Variavel, Variavel\_tecnica)) %>%  
 mutate(  
 Estatistica\_t = Contribuicao / Erro\_Padrao,  
 Valor\_p = 2 \* (1 - pnorm(abs(Estatistica\_t))),  
 Significancia = case\_when(  
 Valor\_p < 0.001 ~ "\*\*\*", Valor\_p < 0.01 ~ "\*\*", Valor\_p < 0.05 ~ "\*", Valor\_p < 0.1 ~ ".", TRUE ~ ""  
 )  
 ) %>%  
 # Correção aplicada aqui  
 dplyr::select(Variavel, Componente, Contribuicao, Erro\_Padrao, Estatistica\_t, Valor\_p, Significancia)  
  
# Extrair as estatísticas de ajuste do modelo  
ajuste\_modelo\_A <- glance(modelo\_homens)  
ajuste\_modelo\_B <- glance(modelo\_mulheres)  
  
# Criar a lista final com todas as tabelas  
lista\_de\_resultados <- list(  
 "Resumo\_Geral" = resumo\_geral,  
 "Decomposicao\_Detalhada" = contribuicoes\_detalhadas,  
 "Regressao\_Grupo\_A\_Homens" = regressao\_grupo\_A,  
 "Regressao\_Grupo\_B\_Mulheres" = regressao\_grupo\_B,  
 "Ajuste\_Modelo\_A\_Homens" = ajuste\_modelo\_A,  
 "Ajuste\_Modelo\_B\_Mulheres" = ajuste\_modelo\_B  
)  
  
# Salvar a lista em um ÚNICO arquivo Excel  
write\_xlsx(lista\_de\_resultados, path = "Resultados\_Oaxaca\_Completos\_Rotulados.xlsx")  
  
# Mensagem de confirmação  
print("Arquivo 'Resultados\_Oaxaca\_Completos\_Rotulados.xlsx' foi salvo com sucesso!")  
  
  
# Gráficos  
library(readxl)  
library(ggplot2)  
library(dplyr)  
  
# Carregar os dados  
detalhada\_df <- read\_excel("Resultados\_Oaxaca\_Completos\_Rotulados.xlsx", sheet = "Decomposicao\_Detalhada")  
  
  
# Criando um vetor com os nomes exatos dos rótulos a serem excluídos  
exclude\_labels <- c(  
 "Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021",   
 "Ano\_2022", "Ano\_2023", "Ano\_2024",  
 "Trimestre\_2", "Trimestre\_3", "Trimestre\_4"  
)  
  
# Preparando os dados para o gráfico  
detalhada\_plot\_df <- detalhada\_df %>%  
 filter(Variavel != "(Intercept)") %>%  
 # 2. Aplicar o filtro: remova as linhas cujo rótulo ESTÁ na lista 'exclude\_labels'  
 filter(!Variavel %in% exclude\_labels) %>%  
 mutate(  
 Componente\_Label = recode(Componente,  
 "endowments" = "Dotações (Explicado)",  
 "coefficients" = "Retornos (Não Explicado)",  
 "interaction" = "Interação"  
 ),  
 Componente\_Label = factor(Componente\_Label, levels = c(  
 "Dotações (Explicado)",  
 "Retornos (Não Explicado)",  
 "Interação"  
 ))  
 )  
  
# Criar o gráfico  
p2\_final <- ggplot(detalhada\_plot\_df, aes(x = Contribuicao, y = reorder(Variavel, Contribuicao))) +  
 geom\_vline(xintercept = 0, color = "grey") +  
 geom\_point(aes(color = Componente\_Label), size = 3, show.legend = FALSE) +  
 geom\_segment(aes(xend = 0, yend = Variavel, color = Componente\_Label), show.legend = FALSE) +  
 geom\_text(  
 aes(label = Significancia),  
 color = "black",  
 hjust = ifelse(detalhada\_plot\_df$Contribuicao >= 0, -0.5, 1.5),  
 size = 4  
 ) +  
 facet\_wrap(~ Componente\_Label, scales = "free\_y") +  
 labs(  
 title = "Decomposição Detalhada por Variável",  
 x = "Contribuição para a Diferença Salarial",  
 y = ""  
 ) +  
 theme\_minimal()  
  
# Mostrar e salvar  
print(p2\_final)  
ggsave("grafico\_2\_decomposicao\_final.pdf", plot = p2\_final, width = 12, height = 8)  
  
  
  
# Gráficos coeficientes regressao  
# Carregar os dados das regressões  
reg\_homens <- read\_excel("Resultados\_Oaxaca\_Completos\_Rotulados.xlsx", sheet = "Regressao\_Grupo\_A\_Homens") %>%  
 mutate(Grupo = "Homens")  
reg\_mulheres <- read\_excel("Resultados\_Oaxaca\_Completos\_Rotulados.xlsx", sheet = "Regressao\_Grupo\_B\_Mulheres") %>%  
 mutate(Grupo = "Mulheres")  
  
  
# Criando um vetor com os nomes exatos dos rótulos a serem excluídos  
exclude\_labels <- c(  
 "Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021",   
 "Ano\_2022", "Ano\_2023", "Ano\_2024",  
 "Trimestre\_2", "Trimestre\_3", "Trimestre\_4"  
)  
  
  
# Combinando e filtrando  
reg\_combinada <- bind\_rows(reg\_homens, reg\_mulheres) %>%  
 filter(Variavel != "(Intercept)") %>%  
 # 2. Aplicar o filtro: remova as linhas cujo rótulo ESTÁ na lista 'exclude\_labels'  
 filter(!Variavel %in% exclude\_labels)  
  
# Criar o gráfico de comparação  
p3\_final <- ggplot(reg\_combinada, aes(x = Coeficiente\_beta, y = reorder(Variavel, Coeficiente\_beta), color = Grupo)) +  
 geom\_vline(xintercept = 0, color = "grey", linetype = "dashed") +  
 geom\_point(size = 3, alpha = 0.8) +  
 geom\_text(  
 aes(label = Significancia),  
 color = "black",  
 vjust = -1,  
 size = 4,  
 show.legend = FALSE  
 ) +  
 labs(  
 title = "Comparação dos Coeficientes da Regressão (Retornos) por Grupo",  
 x = "Valor do Coeficiente (β)",  
 y = "",  
 color = "Grupo"  
 ) +  
 theme\_minimal() +  
 theme(legend.position = "top")  
  
# Mostrar e salvar  
print(p3\_final)  
ggsave("grafico\_3\_comparacao\_coeficientes\_final.pdf", plot = p3\_final, width = 10, height = 8)  
  
  
  
# Gráficos de barras - Forest Bar Plot  
  
library(readxl)  
library(ggplot2)  
library(dplyr)  
library(forcats)  
  
# Carregar os dados (já está no seu código)  
detalhada\_df <- read\_excel("Resultados\_Oaxaca\_Completos\_Rotulados.xlsx", sheet = "Decomposicao\_Detalhada")  
  
# Definição dos rótulos a excluir (LIMPO)  
exclude\_labels <- c(  
 "Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021",  
 "Ano\_2022", "Ano\_2023", "Ano\_2024",  
 "Trimestre\_2", "Trimestre\_3", "Trimestre\_4"  
)  
  
# Novo objeto para os Gráficos de Barra (incluindo IC e ordenação)  
detalhada\_plot\_df\_clean <- detalhada\_df %>%  
 filter(Variavel != "(Intercept)") %>%  
 filter(!Variavel %in% exclude\_labels) %>%  
 mutate(  
 # \*\*\* CÁLCULO DO INTERVALO DE CONFIANÇA (IC) \*\*\*  
 CI\_low = Contribuicao - 1.96 \* Erro\_Padrao,  
 CI\_high = Contribuicao + 1.96 \* Erro\_Padrao,  
   
 # Ordenação necessária para a função criar\_grafico\_oaxaca\_barra  
 Variavel\_ordenada = forcats::fct\_reorder(Variavel, Contribuicao),  
 posicao\_texto = Contribuicao,  
 hjust\_ajustado = ifelse(Contribuicao >= 0, -0.1, 1.1),  
   
 # Rótulos para o gráfico de dispersão (p2\_final) (LIMPO)  
 Componente\_Label = recode(Componente,  
 "endowments" = "Dotações (Explicado)",  
 "coefficients" = "Retornos (Não Explicado)",  
 "interaction" = "Interação"  
 ),  
 Componente\_Label = factor(Componente\_Label, levels = c(  
 "Dotações (Explicado)",  
 "Retornos (Não Explicado)",  
 "Interação"  
 ))  
 )  
  
  
# Nota: Este código assume que 'detalhada\_plot\_df\_clean' já foi criada  
# com as colunas Contribuicao, Erro\_Padrao, CI\_low, CI\_high e Componente.  
  
# Função para gerar o gráfico de barras horizontais (Forest Bar Plot)  
criar\_grafico\_oaxaca\_barra <- function(dados, componente\_nome, titulo\_grafico, nome\_arquivo) {  
 # Filtrar dados para o componente específico  
 dados\_componente <- dados %>%  
 filter(Componente == componente\_nome) %>%  
 # Ordenar as variáveis pela Contribuição (para a ordem correta no eixo Y)  
 mutate(Variavel\_ordenada = fct\_reorder(Variavel, Contribuicao))  
   
 # Altura do gráfico ajustada  
 altura\_plot <- max(6, length(unique(dados\_componente$Variavel)) \* 0.35)  
   
 # Definir o ponto de ancoragem para o texto de significância (ajuste o 'hjust' para colocar no final da barra)  
 dados\_componente <- dados\_componente %>%  
 mutate(  
 posicao\_texto = Contribuicao,  
 # Ajusta a posição da estrela para o final da barra, usando o sinal da contribuição  
 hjust\_ajustado = ifelse(Contribuicao >= 0, -0.1, 1.1)   
 )  
   
 p <- ggplot(dados\_componente, aes(x = Variavel\_ordenada, y = Contribuicao)) +  
   
 # Gráfico de Barras (geom\_col)  
 geom\_col(fill = "#1FBCB3", color = "#1FBCB3", width = 0.7) +  
   
 # Barras de Erro (Intervalo de Confiança de 95%)  
 # Note que o 'geom\_errorbar' aqui requer ymin/ymax  
 geom\_errorbar(aes(ymin = CI\_low, ymax = CI\_high), width = 0.2, color = "black", linewidth = 0.5) +  
   
 # Linha vertical em zero (agora horizontal no gráfico final)  
 geom\_hline(yintercept = 0, color = "grey", linetype = "solid") +  
   
 # Inverte os eixos para obter o gráfico de barras horizontais  
 coord\_flip() +  
   
 # Texto de Significância (posicionado no final da barra)  
 geom\_text(  
 aes(label = Significancia, x = Variavel\_ordenada, y = posicao\_texto),  
 color = "black",  
 hjust = dados\_componente$hjust\_ajustado,  
 size = 4  
 ) +  
 labs(  
 title = titulo\_grafico,  
 x = "", # O eixo Y agora é o das variáveis  
 y = "Coeficiente"  
 ) +  
 theme\_minimal() +  
 theme(  
 plot.title = element\_text(hjust = 0.5, size = 16),  
 # Remove as linhas de grade que acompanham o eixo Y original  
 panel.grid.major.y = element\_blank(),   
 panel.grid.minor.y = element\_blank(),  
 axis.text.y = element\_text(size = 10)  
 )  
   
 # Salvar o gráfico  
 ggsave(nome\_arquivo, plot = p, width = 9, height = altura\_plot)  
 return(p)  
}  
  
# Gerar e salvar os 3 gráficos separados  
  
# Gráfico 1: Endowments (Dotação / Explicado)  
p\_endowments\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_df\_clean,  
 componente\_nome = "endowments",  
 titulo\_grafico = "Endowments (Dotação / Explicado)",  
 nome\_arquivo = "grafico\_oaxaca\_endowments\_barra.pdf"  
)  
print(p\_endowments\_barra)  
  
# Gráfico 2: Coefficients (Coeficientes / Não-explicado)  
p\_coefficients\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_df\_clean,  
 componente\_nome = "coefficients",  
 titulo\_grafico = "Coefficients (Coeficientes / Não-explicado)",  
 nome\_arquivo = "grafico\_oaxaca\_coefficients\_barra.pdf"  
)  
print(p\_coefficients\_barra)  
  
# Gráfico 3: Interaction (Interação)  
p\_interaction\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_df\_clean,  
 componente\_nome = "interaction",  
 titulo\_grafico = "Interaction (Interação)",  
 nome\_arquivo = "grafico\_oaxaca\_interaction\_barra.pdf"  
)  
print(p\_interaction\_barra)  
  
  
  
### Decomposição de Oaxaca Binder por Raça--------------------------------  
  
# Carregando as bibliotecas necessárias  
library(dplyr)  
library(tidyr)  
library(broom)  
library(writexl)  
library(readxl)  
library(ggplot2)  
  
# Definindo os rótulos para as variáveis (reutilizável para ambas as análises)  
rotulos <- c(  
 `V2010\_grupo\_PP` = "Pretas e Pardas",  
 `V2009` = "Idade (anos)",  
 `I(V2009^2)` = "Idade² (anos²)",  
 `VD3005\_cont` = "Escolaridade (Anos)",  
 `VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior` = "Sem Ensino Superior",  
 `V1023\_recode3\_Resto\_UF` = "Resto da UF",  
 `Regiao\_Sul` = "Região Sul",  
 `Regiao\_Centro\_Oeste` = "Região Centro Oeste",  
 `Regiao\_Norte` = "Região Norte",  
 `Regiao\_Nordeste` = "Região Nordeste",  
 `V4029\_Sim` = "Carteira Assinada (Sim)",  
 `V4025\_Não` = "Contrato Temporário (Não)",  
 `V4039` = "Horas Semanais Trabalhadas",  
 `Horas\_trabalhadas2\_Integral` = "Integral",  
 `V4040\_recode\_Mais\_de\_2\_anos` = "Tempo de Emprego > 2 anos",  
 `V2007\_Mulher` = "Mulheres"  
)  
  
rotulos\_df <- tibble(  
 Variavel\_tecnica = names(rotulos),  
 Label = unname(rotulos)  
)  
  
# Rodando o modelo de Oaxaca-Blinder para RAÇA  
oaxaca\_raca\_result <- oaxaca(  
 formula = VD4016\_log\_hora ~ V2007\_Mulher + V2009 + I(V2009^2) + VD3005\_cont +   
 VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior + V1023\_recode3\_Resto\_UF + Regiao\_Sul + Regiao\_Centro\_Oeste + Regiao\_Norte + Regiao\_Nordeste +  
 V4029\_Sim + V4025\_Não + V4039 + Horas\_trabalhadas2\_Integral + V4040\_recode\_Mais\_de\_2\_anos +  
 Ano\_2017 + Ano\_2018 + Ano\_2019 + Ano\_2020 + Ano\_2021 + Ano\_2022 + Ano\_2023 + Ano\_2024 + Trimestre\_2 + Trimestre\_3 + Trimestre\_4  
 | V2010\_grupo\_PP,   
 data = pnad\_tur\_Filtrada,   
 reg.fun = lm  
)  
  
# Criando os modelos de regressão separados para cada grupo racial  
# Assumindo que 0 = Brancos (grupo de referência) e 1 = Pretos e Pardos  
dados\_brancos <- subset(pnad\_tur\_Filtrada, V2010\_grupo\_PP == 0)  
dados\_pp <- subset(pnad\_tur\_Filtrada, V2010\_grupo\_PP == 1)  
  
formula\_regressao\_raca <- VD4016\_log\_hora ~ V2007\_Mulher + V2009 + I(V2009^2) + VD3005\_cont +   
 VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior + V1023\_recode3\_Resto\_UF + Regiao\_Sul + Regiao\_Centro\_Oeste + Regiao\_Norte + Regiao\_Nordeste +  
 V4029\_Sim + V4025\_Não + V4039 + Horas\_trabalhadas2\_Integral + V4040\_recode\_Mais\_de\_2\_anos +  
 Ano\_2017 + Ano\_2018 + Ano\_2019 + Ano\_2020 + Ano\_2021 + Ano\_2022 + Ano\_2023 + Ano\_2024 + Trimestre\_2 + Trimestre\_3 + Trimestre\_4  
  
modelo\_brancos <- lm(formula\_regressao\_raca, data = dados\_brancos)  
modelo\_pp <- lm(formula\_regressao\_raca, data = dados\_pp)  
  
  
# Extraindo e formatando todas as tabelas  
# Extrair o RESUMO GERAL da decomposição (Método Manual e Robusto)  
overall\_raw\_raca <- summary(oaxaca\_raca\_result)$threefold$overall  
resumo\_geral\_raca <- tibble(  
 Componente = c("Endowments (Explicado)", "Coefficients (Não Explicado)", "Interaction"),  
 Valor = overall\_raw\_raca[c("coef(endowments)", "coef(coefficients)", "coef(interaction)")],  
 Erro\_Padrao = overall\_raw\_raca[c("se(endowments)", "se(coefficients)", "se(interaction)")]  
) %>%  
 mutate(  
 Estatistica\_t = Valor / Erro\_Padrao,  
 Valor\_p = 2 \* (1 - pnorm(abs(Estatistica\_t))),  
 Significancia = case\_when(  
 Valor\_p < 0.001 ~ "\*\*\*", Valor\_p < 0.01 ~ "\*\*", Valor\_p < 0.05 ~ "\*", Valor\_p < 0.1 ~ ".", TRUE ~ ""  
 )  
 )  
  
regressao\_brancos <- tidy(modelo\_brancos) %>%  
 left\_join(rotulos\_df, by = c("term" = "Variavel\_tecnica")) %>% mutate(Variavel = coalesce(Label, term)) %>%  
 mutate(Significancia = case\_when(p.value < 0.001 ~ "\*\*\*", p.value < 0.01 ~ "\*\*", p.value < 0.05 ~ "\*", p.value < 0.1 ~ ".", TRUE ~ "")) %>%  
 dplyr::select(Variavel, Coeficiente\_beta = estimate, std.error, statistic, p.value, Significancia)  
  
regressao\_pp <- tidy(modelo\_pp) %>%  
 left\_join(rotulos\_df, by = c("term" = "Variavel\_tecnica")) %>% mutate(Variavel = coalesce(Label, term)) %>%  
 mutate(Significancia = case\_when(p.value < 0.001 ~ "\*\*\*", p.value < 0.01 ~ "\*\*", p.value < 0.05 ~ "\*", p.value < 0.1 ~ ".", TRUE ~ "")) %>%  
 dplyr::select(Variavel, Coeficiente\_beta = estimate, std.error, statistic, p.value, Significancia)  
  
contribuicoes\_raca <- as.data.frame(oaxaca\_raca\_result$threefold$variables) %>%  
 mutate(Variavel\_tecnica = rownames(.)) %>%  
 pivot\_longer(cols = -Variavel\_tecnica, names\_to = c(".value", "Componente"), names\_pattern = "(coef|se)\\((.\*)\\)") %>%  
 rename(Contribuicao = coef, Erro\_Padrao = se) %>%  
 left\_join(rotulos\_df, by = "Variavel\_tecnica") %>% mutate(Variavel = coalesce(Label, Variavel\_tecnica)) %>%  
 mutate(Estatistica\_t = Contribuicao / Erro\_Padrao, Valor\_p = 2 \* (1 - pnorm(abs(Estatistica\_t))),  
 Significancia = case\_when(Valor\_p < 0.001 ~ "\*\*\*", Valor\_p < 0.01 ~ "\*\*", Valor\_p < 0.05 ~ "\*", Valor\_p < 0.1 ~ ".", TRUE ~ "")) %>%  
 dplyr::select(Variavel, Componente, Contribuicao, Erro\_Padrao, Estatistica\_t, Valor\_p, Significancia)  
  
ajuste\_brancos <- glance(modelo\_brancos)  
ajuste\_pp <- glance(modelo\_pp)  
  
  
lista\_resultados\_raca <- list(  
 "Resumo\_Geral" = resumo\_geral\_raca,  
 "Decomposicao\_Detalhada" = contribuicoes\_raca,  
 "Regressao\_Grupo\_A\_Brancos" = regressao\_brancos,  
 "Regressao\_Grupo\_B\_PP" = regressao\_pp,  
 "Ajuste\_Modelo\_A\_Brancos" = ajuste\_brancos,  
 "Ajuste\_Modelo\_B\_PP" = ajuste\_pp  
)  
write\_xlsx(lista\_resultados\_raca, path = "Resultados\_Oaxaca\_por\_Raca.xlsx")  
print("Arquivo 'Resultados\_Oaxaca\_por\_Raca.xlsx' foi salvo com sucesso!")  
  
  
# Geração gráficos Raça  
  
# Os outros gráficos já estavam corretos  
# Gráfico 2: Contribuições Detalhadas por Raça  
exclude\_labels <- c("Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021", "Ano\_2022", "Ano\_2023", "Ano\_2024", "Trimestre\_2", "Trimestre\_3", "Trimestre\_4")  
detalhada\_plot\_raca <- contribuicoes\_raca %>%  
 filter(Variavel != "(Intercept)") %>% filter(!Variavel %in% exclude\_labels) %>%  
 mutate(  
 Componente\_Label = recode(Componente, "endowments" = "Dotações (Explicado)", "coefficients" = "Retornos (Não Explicado)", "interaction" = "Interação"),  
 Componente\_Label = factor(Componente\_Label, levels = c("Dotações (Explicado)", "Retornos (Não Explicado)", "Interação"))  
 )  
p2\_raca <- ggplot(detalhada\_plot\_raca, aes(x = Contribuicao, y = reorder(Variavel, Contribuicao))) +  
 geom\_vline(xintercept = 0, color = "grey") +  
 geom\_point(aes(color = Componente\_Label), size = 3, show.legend = FALSE) +  
 geom\_segment(aes(xend = 0, yend = Variavel, color = Componente\_Label), show.legend = FALSE) +  
 geom\_text(aes(label = Significancia), color = "black", hjust = ifelse(detalhada\_plot\_raca$Contribuicao >= 0, -0.5, 1.5), size = 4) +  
 facet\_wrap(~ Componente\_Label, scales = "free\_y") +  
 labs(title = "Decomposição Detalhada por Raça", x = "Contribuição para a Diferença Salarial", y = "") +  
 theme\_minimal()  
print(p2\_raca)  
ggsave("grafico\_2\_decomposicao\_detalhada\_raca.pdf", plot = p2\_raca, width = 12, height = 8)  
  
# Gráfico 3: Comparação de Coeficientes por Raça  
reg\_combinada\_raca <- bind\_rows(mutate(regressao\_brancos, Grupo = "Brancos"), mutate(regressao\_pp, Grupo = "Pretos e Pardos")) %>%  
 filter(Variavel != "(Intercept)") %>% filter(!Variavel %in% exclude\_labels)  
p3\_raca <- ggplot(reg\_combinada\_raca, aes(x = Coeficiente\_beta, y = reorder(Variavel, Coeficiente\_beta), color = Grupo)) +  
 geom\_vline(xintercept = 0, color = "grey", linetype = "dashed") +  
 geom\_point(size = 3, alpha = 0.8) +  
 geom\_text(aes(label = Significancia), color = "black", vjust = -1, size = 4, show.legend = FALSE) +  
 labs(title = "Comparação dos Coeficientes da Regressão por Raça", x = "Valor do Coeficiente (β)", y = "", color = "Grupo") +  
 theme\_minimal() + theme(legend.position = "top")  
print(p3\_raca)  
ggsave("grafico\_3\_comparacao\_coeficientes\_raca.pdf", plot = p3\_raca, width = 10, height = 8)  
  
  
# Geração de gráficos barra para Raça  
  
# Definições e Preparação dos Dados (reutilizando 'exclude\_labels' e 'contribuicoes\_raca')  
# Nota: 'exclude\_labels' e 'contribuicoes\_raca' devem ter sido criados nas seções anteriores.  
  
# Preparação do Data Frame para Plotagem (com CIs e filtros)  
exclude\_labels <- c("Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021", "Ano\_2022", "Ano\_2023", "Ano\_2024", "Trimestre\_2", "Trimestre\_3", "Trimestre\_4")  
  
detalhada\_plot\_raca\_barra <- contribuicoes\_raca %>%  
 filter(Variavel != "(Intercept)") %>%  
 filter(!Variavel %in% exclude\_labels) %>%  
 # Calcular o Intervalo de Confiança de 95% (assumindo Z=1.96)  
 mutate(  
 CI\_low = Contribuicao - 1.96 \* Erro\_Padrao,  
 CI\_high = Contribuicao + 1.96 \* Erro\_Padrao,  
 # Ordenar as variáveis pela Contribuição  
 Variavel\_ordenada = forcats::fct\_reorder(Variavel, Contribuicao),  
   
 # Posição e ajuste do texto de significância para o final da barra  
 posicao\_texto = Contribuicao,  
 hjust\_ajustado = ifelse(Contribuicao >= 0, -0.1, 1.1)   
 )  
  
# Função para gerar o gráfico de barras horizontais (Forest Bar Plot)  
criar\_grafico\_oaxaca\_barra <- function(dados, componente\_nome, titulo\_grafico, nome\_arquivo) {  
 # Filtrar dados para o componente específico  
 dados\_componente <- dados %>%  
 filter(Componente == componente\_nome)  
   
 # Altura do gráfico ajustada  
 altura\_plot <- max(6, length(unique(dados\_componente$Variavel)) \* 0.35)  
   
 p <- ggplot(dados\_componente, aes(x = Variavel\_ordenada, y = Contribuicao)) +  
   
 # Gráfico de Barras (geom\_col)  
 geom\_col(fill = "#1FBCB3", color = "#1FBCB3", width = 0.7) +  
   
 # Barras de Erro (Intervalo de Confiança de 95%)  
 geom\_errorbar(aes(ymin = CI\_low, ymax = CI\_high), width = 0.2, color = "black", linewidth = 0.5) +  
   
 # Linha vertical em zero (será horizontal no gráfico final)  
 geom\_hline(yintercept = 0, color = "grey", linetype = "solid") +  
   
 # Inverte os eixos para obter o gráfico de barras horizontais  
 coord\_flip() +  
   
 # Texto de Significância (posicionado no final da barra)  
 geom\_text(  
 aes(label = Significancia, x = Variavel\_ordenada, y = posicao\_texto, hjust = hjust\_ajustado),  
 color = "black",  
 size = 4  
 ) +  
 labs(  
 title = titulo\_grafico,  
 x = "",   
 y = "Coeficiente"  
 ) +  
 theme\_minimal() +  
 theme(  
 plot.title = element\_text(hjust = 0.5, size = 16),  
 panel.grid.major.y = element\_blank(),   
 panel.grid.minor.y = element\_blank(),  
 axis.text.y = element\_text(size = 10)  
 )  
   
 # Salvar o gráfico  
 ggsave(nome\_arquivo, plot = p, width = 9, height = altura\_plot)  
 return(p)  
}  
  
# Gerar e salvar os 3 gráficos separados para RAÇA  
  
# Gráfico 1: Endowments (Dotação / Explicado)  
p\_endowments\_raca\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_raca\_barra,  
 componente\_nome = "endowments",  
 titulo\_grafico = "Endowments (Dotação / Explicado) - Raça",  
 nome\_arquivo = "grafico\_oaxaca\_endowments\_raca\_barra.pdf"  
)  
print(p\_endowments\_raca\_barra)  
  
# Gráfico 2: Coefficients (Coeficientes / Não-explicado)  
p\_coefficients\_raca\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_raca\_barra,  
 componente\_nome = "coefficients",  
 titulo\_grafico = "Coefficients (Coeficientes / Não-explicado) - Raça",  
 nome\_arquivo = "grafico\_oaxaca\_coefficients\_raca\_barra.pdf"  
)  
print(p\_coefficients\_raca\_barra)  
  
# Gráfico 3: Interaction (Interação)  
p\_interaction\_raca\_barra <- criar\_grafico\_oaxaca\_barra(  
 detalhada\_plot\_raca\_barra,  
 componente\_nome = "interaction",  
 titulo\_grafico = "Interaction (Interação) - Raça",  
 nome\_arquivo = "grafico\_oaxaca\_interaction\_raca\_barra.pdf"  
)  
print(p\_interaction\_raca\_barra)  
  
  
## REGRESSAO QUANTILICA DE TODOS OS MODELOS COM PARALELIZAÇÃO PARA OTIMIZAÇÃO--------------------------------  
   
library(quantreg)  
library(dplyr)  
library(tidyr)  
library(broom)  
library(car)  
library(lmtest)  
library(writexl)  
library(ggplot2)  
library(stringr)  
library(doParallel) # Novo! Para paralelização  
library(foreach) # Novo! Para paralelização  
  
# Variável dependente e quantis  
Y <- "VD4016\_log\_hora"  
quantis <- c(0.10, 0.25, 0.50, 0.75, 0.90)  
meu\_azul <- "#00ACC1"  
  
# Rótulos Completos   
rotulos <- c(  
 # Termos individuais  
 "(Intercept)" = "Intercepto",  
 "V2007\_Mulher" = "Mulher",  
 "V2010\_grupo\_PP" = "Raça (PP)",  
 "V2009" = "Idade",  
 "V2009\_quad" = "Idade²",  
 "VD3005\_cont" = "Escolaridade (anos)",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior" = "Sem Ensino Superior",  
 "V1023\_recode3\_Resto\_UF" = "Não Capital ou Metropolitana",  
 "Regiao\_Nordeste" = "Nordeste",  
 "Regiao\_Centro\_Oeste" = "Centro Oeste",  
 "Regiao\_Sul" = "Sul",  
 "Regiao\_Norte" = "Norte",  
 "Regiao\_Sudeste" = "Sudeste",  
 "V4029\_Sim" = "Carteira Assinada (Sim)",  
 "`V4025\_Não`" = "Contrato Temporário (Não)",   
 "V4025\_Não" = "Contrato Temporário (Não)", # Versão sem crase  
 "V4039" = "Horas Semanais",  
 "Horas\_trabalhadas2\_Integral" = "Regime Integral",  
 "V4040\_recode\_Mais\_de\_2\_anos" = "Mais de 2 anos de Emprego",  
 "Ano\_2017" = "2017", "Ano\_2018" = "2018", "Ano\_2019" = "2019",  
 "Ano\_2020" = "2020", "Ano\_2021" = "2021", "Ano\_2022" = "2022",  
 "Ano\_2023" = "2023", "Ano\_2024" = "2024",  
 "Trimestre\_2" = "2º Tri", "Trimestre\_3" = "3º Tri", "Trimestre\_4" = "4º Tri",  
 "Pandemia\_Durante\_Pandemia" = "Durante a Pandemia (20-21)",  
 "Pandemia\_Apos\_Pandemia" = "Após a Pandemia (22-24)",  
   
 # Rótulos para Interações (Usando a ordem alfabética para robustez)  
 "V2007\_Mulher:V2010\_grupo\_PP" = "Mulher x Raça (PP)",  
 "V2007\_Mulher:V2009" = "Mulher x Idade",  
 "V2007\_Mulher:VD3005\_cont" = "Mulher x Escolaridade (anos)",  
 "V2007\_Mulher:VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior" = "Mulher x Sem Ensino Superior",  
 "V2007\_Mulher:Regiao\_Centro\_Oeste" = "Mulher x Centro Oeste",  
 "V2007\_Mulher:Regiao\_Nordeste" = "Mulher x Nordeste",  
 "V2007\_Mulher:Regiao\_Norte" = "Mulher x Norte",  
 "V2007\_Mulher:Regiao\_Sul" = "Mulher x Sul",  
 "V2007\_Mulher:V1023\_recode3\_Resto\_UF" = "Mulher x Não Capital ou Metropolitana",  
 "V2007\_Mulher:V4029\_Sim" = "Mulher x Carteira Assinada (Sim)",  
 "V4025\_Não:V2007\_Mulher" = "Mulher x Contrato Temporário (Não)", # Corrigido: Ordem alfabética  
 "V2007\_Mulher:V4039" = "Mulher x Horas Semanais",  
 "V2007\_Mulher:Horas\_trabalhadas2\_Integral" = "Mulher x Regime Integral",  
 "V2007\_Mulher:V4040\_recode\_Mais\_de\_2\_anos" = "Mulher x Mais de 2 anos de Emprego",  
   
 # Interações com Raça (V2010\_grupo\_PP)  
 "V2010\_grupo\_PP:V2009" = "Raça (PP) x Idade",  
 "V2010\_grupo\_PP:VD3005\_cont" = "Raça (PP) x Escolaridade (anos)",  
 "V2010\_grupo\_PP:VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior" = "Raça (PP) x Sem Ensino Superior",  
 "V2010\_grupo\_PP:Regiao\_Centro\_Oeste" = "Raça (PP) x Centro Oeste",  
 "V2010\_grupo\_PP:Regiao\_Nordeste" = "Raça (PP) x Nordeste",  
 "V2010\_grupo\_PP:Regiao\_Norte" = "Raça (PP) x Norte",  
 "V2010\_grupo\_PP:Regiao\_Sul" = "Raça (PP) x Sul",  
 "V2010\_grupo\_PP:V1023\_recode3\_Resto\_UF" = "Raça (PP) x Não Capital ou Metropolitana",  
 "V2010\_grupo\_PP:V4029\_Sim" = "Raça (PP) x Carteira Assinada (Sim)",  
 "V4025\_Não:V2010\_grupo\_PP" = "Raça x Contrato Temporário (Não)", # Corrigido  
 "V2010\_grupo\_PP:V4039" = "Raça (PP) x Horas Semanais",  
 "V2010\_grupo\_PP:Horas\_trabalhadas2\_Integral" = "Raça (PP) x Regime Integral",  
 "V2010\_grupo\_PP:V4040\_recode\_Mais\_de\_2\_anos" = "Raça (PP) x Mais de 2 anos de Emprego",  
   
 # Interações com Escolaridade (VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior)  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:V2009" = "Sem Ensino Superior x Idade",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:Regiao\_Centro\_Oeste" = "Sem Ensino Superior x Centro Oeste",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:Regiao\_Nordeste" = "Sem Ensino Superior x Nordeste",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:Regiao\_Norte" = "Sem Ensino Superior x Norte",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:Regiao\_Sul" = "Sem Ensino Superior x Sul",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:V1023\_recode3\_Resto\_UF" = "Sem Ensino Superior x Não Capital ou Metropolitana",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:V4029\_Sim" = "Sem Ensino Superior x Carteira Assinada (Sim)",  
 "V4025\_Não:VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior" = "Sem Ensino Superior x Contrato Temporário (Não)", # Corrigido  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:V4039" = "Sem Ensino Superior x Horas Semanais",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:Horas\_trabalhadas2\_Integral" = "Sem Ensino Superior x Regime Integral",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior:V4040\_recode\_Mais\_de\_2\_anos" = "Sem Ensino Superior x Mais de 2 anos de Emprego",  
   
 # Interações com Carteira Assinada (V4029\_Sim)  
 "V4029\_Sim:V2009" = "Carteira Assinada (Sim) x Idade",  
 "V4029\_Sim:VD3005\_cont" = "Carteira Assinada (Sim) x Escolaridade (anos)",  
 "V4029\_Sim:Regiao\_Centro\_Oeste" = "Carteira Assinada (Sim) x Centro Oeste",  
 "V4029\_Sim:Regiao\_Nordeste" = "Carteira Assinada (Sim) x Nordeste",  
 "V4029\_Sim:Regiao\_Norte" = "Carteira Assinada (Sim) x Norte",  
 "V4029\_Sim:Regiao\_Sul" = "Carteira Assinada (Sim) x Sul",  
 "V4029\_Sim:V1023\_recode3\_Resto\_UF" = "Carteira Assinada (Sim) x Não Capital ou Metropolitana",  
 "V4025\_Não:V4029\_Sim" = "Carteira Assinada x Contrato Temporário (Não)", # Corrigido  
 "V4029\_Sim:V4039" = "Carteira Assinada (Sim) x Horas Semanais",  
 "V4029\_Sim:Horas\_trabalhadas2\_Integral" = "Carteira Assinada (Sim) x Regime Integral",  
 "V4029\_Sim:V4040\_recode\_Mais\_de\_2\_anos" = "Carteira Assinada (Sim) x Mais de 2 anos de Emprego",  
   
 # Interações com Idade (V2009)  
 "V2009:VD3005\_cont" = "Idade x Escolaridade (anos)",  
 "V2009:Regiao\_Centro\_Oeste" = "Idade x Centro Oeste",  
 "V2009:Regiao\_Nordeste" = "Idade x Nordeste",  
 "V2009:Regiao\_Norte" = "Idade x Norte",  
 "V2009:Regiao\_Sul" = "Idade x Sul",  
 "V2009:V1023\_recode3\_Resto\_UF" = "Idade x Não Capital ou Metropolitana",  
 "V2009:V4039" = "Idade x Horas Semanais",  
 "V4025\_Não:V2009" = "Idade x Contrato Temporário (Não)", # Corrigido  
 "V2009:Horas\_trabalhadas2\_Integral" = "Idade x Regime Integral",  
 "V2009:V4040\_recode\_Mais\_de\_2\_anos" = "Idade x Mais de 2 anos de Emprego"  
)  
  
# Vetor para fixar a ordem canônica dos fatores no gráfico  
ordem\_canonica <- names(rotulos)  
  
  
# Variáveis base (sem termos de tempo)  
X\_base <- c("V2007\_Mulher", "V2010\_grupo\_PP", "V2009", "V2009\_quad", "VD3005\_cont",  
 "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior",  
 "V1023\_recode3\_Resto\_UF", "Regiao\_Nordeste", "Regiao\_Centro\_Oeste", "Regiao\_Sul", "Regiao\_Norte",  
 "V4029\_Sim", "`V4025\_Não`", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4040\_recode\_Mais\_de\_2\_anos")  
  
# Termos de tempo  
termos\_tempo\_base <- c("Ano\_2017", "Ano\_2018", "Ano\_2019", "Ano\_2020", "Ano\_2021", "Ano\_2022", "Ano\_2023", "Ano\_2024",  
 "Trimestre\_2", "Trimestre\_3", "Trimestre\_4")  
termos\_pandemia <- c("Pandemia\_Durante\_Pandemia", "Pandemia\_Apos\_Pandemia")  
  
  
# Definição das interações agrupadas por variável base  
vars\_para\_interagir <- c("V2010\_grupo\_PP", "V2009", "VD3005\_cont", "VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior",  
 "Regiao\_Nordeste", "Regiao\_Centro\_Oeste", "Regiao\_Sul", "Regiao\_Norte",  
 "V1023\_recode3\_Resto\_UF", "V4029\_Sim", "`V4025\_Não`", "V4039", "Horas\_trabalhadas2\_Integral",  
 "V4040\_recode\_Mais\_de\_2\_anos")  
  
# Interações para cada modelo  
interacoes\_mulher <- paste0("V2007\_Mulher \* ", vars\_para\_interagir)  
interacoes\_raca <- paste0("V2010\_grupo\_PP \* ", vars\_para\_interagir)  
interacoes\_escolaridade <- paste0("VD3004\_nivel\_Escol3\_Sem\_Ensino\_Superior \* ", vars\_para\_interagir)  
interacoes\_cartassinada <- paste0("V4029\_Sim \* ", vars\_para\_interagir)  
interacoes\_idade <- paste0("V2009 \* ", vars\_para\_interagir)  
  
  
### Lista Mestra de Modelos Regressão Quantílica--------------------------------  
lista\_modelos <- list(  
 "Modelo\_Basico" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = NULL,  
 nome\_arquivo = "basico"  
 ),  
 "Modelo\_Pandemia" = list(  
 termos = c(X\_base, termos\_pandemia),  
 interacoes = NULL,  
 nome\_arquivo = "pandemia"  
 ),  
 "Modelo\_Int\_Mulher" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = interacoes\_mulher,  
 nome\_arquivo = "int\_mulher"  
 ),  
 "Modelo\_Int\_Raca" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = interacoes\_raca,  
 nome\_arquivo = "int\_raca"  
 ),  
 "Modelo\_Int\_Escolaridade" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = interacoes\_escolaridade,  
 nome\_arquivo = "int\_escol"  
 ),  
 "Modelo\_Int\_CarteiraAssinada" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = interacoes\_cartassinada,  
 nome\_arquivo = "int\_cartAss"  
 ),  
 "Modelo\_Int\_Idade" = list(  
 termos = c(X\_base, termos\_tempo\_base),  
 interacoes = interacoes\_idade,  
 nome\_arquivo = "int\_idade"  
 )  
)  
  
  
  
cria\_formula <- function(y, termos, interacoes = NULL) {  
 if (is.null(interacoes)) {  
 formula\_str <- paste(y, "~", paste(termos, collapse = " + "))  
 } else {  
 termos\_full <- unique(c(termos, interacoes))  
 formula\_str <- paste(y, "~", paste(termos\_full, collapse = " + "))  
 }  
 return(as.formula(formula\_str))  
}  
  
  
processa\_modelo\_quantilica <- function(formula, data, taus, nome\_modelo, rotulos\_map) {  
 cat(paste0("Ajustando: ", nome\_modelo, " (Bootstrap B=500)\n"))  
   
 # Passo 1: Ajuste da Regressão Quantílica   
 p.erq <- rq(  
 formula = formula,  
 tau = taus,  
 method = "br",  
 data = data  
 )  
   
 # Defina o número de repetições do Bootstrap aqui  
 NUM\_BOOTSTRAP <- 500   
   
 # Passo 2: Resumo Tidy usando Bootstrap   
 resultados\_tidy <- tidy(  
 p.erq,   
 se.type = "boot", # Usa o método Bootstrap  
 B = NUM\_BOOTSTRAP # Define o número de repetições  
 ) %>%  
 mutate(  
 significancia = case\_when(  
 p.value < 0.001 ~ "\*\*\*",  
 p.value < 0.01 ~ "\*\*",  
 p.value < 0.05 ~ "\*",  
 TRUE ~ ""  
 ),  
 term\_label = dplyr::recode(term, !!!rotulos\_map, .default = term)   
 )  
   
 # Passo 3: Filtro e Tabela Wide   
 resultados\_tidy <- resultados\_tidy %>%  
 filter(term\_label != "NA" & !is.na(term\_label))  
   
 coeficientes\_wide <- resultados\_tidy %>%  
 dplyr::select(tau, term\_label, estimate, significancia) %>%  
 dplyr::mutate(  
 estimate = paste0(format(round(estimate, 4), nsmall = 4), significancia)  
 ) %>%  
 dplyr::select(tau, term\_label, estimate) %>%  
 tidyr::pivot\_wider(names\_from = tau, values\_from = estimate, names\_prefix = "Quantil\_")  
   
 return(list(  
 p.erq = p.erq,  
 resultados\_tidy = resultados\_tidy,  
 coeficientes\_wide = coeficientes\_wide  
 ))  
}  
  
  
calcula\_ajuste <- function(formula, data, taus, Y) {  
 n <- nrow(data)  
 resultados\_list <- list()  
 for(t in taus){  
 modelo <- rq(formula, tau = t, data = data, method = "br")  
 modelo\_nulo <- rq(as.formula(paste(Y, "~ 1")), tau = t, data = data, method = "br")  
   
 pseudo\_R2 <- 1 - modelo$rho / modelo\_nulo$rho  
 k <- length(coef(modelo))  
 AIC\_pseudo <- n \* log(modelo$rho / n) + 2 \* k   
   
 resultados\_list[[as.character(t)]] <- data.frame(  
 Quantil = t,  
 Pseudo\_R2 = pseudo\_R2,  
 AIC = AIC\_pseudo  
 )  
 }  
 return(bind\_rows(resultados\_list))  
}  
  
  
gera\_grafico\_main <- function(df\_tidy, nome\_modelo, rotulos\_map, ordem\_cana, color, filtro\_termos\_excluir = "Ano|Trimestre") {  
   
 df\_tidy <- df\_tidy %>% filter(!is.na(term\_label))  
   
 term\_col\_name <- "term"  
 niveis\_presentes <- intersect(ordem\_cana, unique(df\_tidy[[term\_col\_name]]))  
   
 df\_plot <- df\_tidy %>%  
 filter(term %in% niveis\_presentes) %>%  
 mutate(!!term\_col\_name := factor(.data[[term\_col\_name]], levels = niveis\_presentes)) %>%  
 filter(!grepl(filtro\_termos\_excluir, term))  
   
 grafico <- df\_plot %>%  
 ggplot(aes(x = tau, y = estimate)) +  
 geom\_hline(yintercept = 0, linetype = "dashed", color = "gray") +  
 geom\_line(size = 1, color = color) +  
 geom\_point(aes(shape = significancia), size = 3, color = color) +  
 facet\_wrap(~ term, scales = "free\_y", labeller = as\_labeller(rotulos\_map)) +  
 labs(  
 title = paste("Coeficientes por Quantil (Modelo: ", nome\_modelo, ")", sep=""),  
 x = "Quantil",  
 y = "Coeficiente Estimado",  
 shape = "Significância"  
 ) +  
 scale\_shape\_manual(  
 values = c(" " = 1, "\*" = 16, "\*\*" = 17, "\*\*\*" = 18),  
 labels = c(" " = "Sem significância", "\*" = "p < 0.05", "\*\*" = "p < 0.01", "\*\*\*" = "p < 0.001"),  
 drop = FALSE  
 ) +  
 theme\_minimal() +  
 theme(  
 text = element\_text(size = 12),  
 strip.text = element\_text(size = 10, face = "bold"),  
 legend.position = "bottom",  
 plot.title = element\_text(hjust = 0.5)  
 )  
 return(grafico)  
}  
  
  
  
# Configurar o Cluster de Paralelização  
# Tente usar todos os núcleos menos um para evitar travar o PC  
num\_cores <- detectCores() - 1   
if (num\_cores < 1) num\_cores <- 1 # Garante pelo menos um núcleo  
  
cat(paste("Iniciando paralelização em", num\_cores, " núcleos...\n"))  
cl <- makeCluster(num\_cores)  
registerDoParallel(cl)  
  
# Loop paralelo   
# Exporta todas as funções e variáveis necessárias para os núcleos  
# O retorno é suprimido, pois salvamos os arquivos diretamente no loop  
# O dataframe pnad\_tur\_Filtrada deve estar carregado globalmente.  
tempo\_inicio <- Sys.time()  
  
resultados\_paralelos <- foreach(  
 nome = names(lista\_modelos),   
 .packages = c("quantreg", "dplyr", "tidyr", "broom", "stringr", "writexl", "ggplot2"),  
 .export = c("Y", "quantis", "meu\_azul", "rotulos", "ordem\_canonica", "lista\_modelos",   
 "cria\_formula", "processa\_modelo\_quantilica", "calcula\_ajuste",   
 "gera\_grafico\_main", "pnad\_tur\_Filtrada"),  
 .errorhandling = "stop"  
) %dopar% {  
   
 # O corpo loop original (que agora roda em paralelo)  
   
 modelo\_info <- lista\_modelos[[nome]]  
 arquivo\_base <- modelo\_info$nome\_arquivo   
   
 # Cria a fórmula  
 formula\_modelo <- cria\_formula(Y, modelo\_info$termos, modelo\_info$interacoes)  
   
 # Processa (ajusta, resume e cria tabela wide)  
 resultados <- processa\_modelo\_quantilica(  
 formula = formula\_modelo,  
 data = pnad\_tur\_Filtrada,   
 taus = quantis,  
 nome\_modelo = nome,  
 rotulos\_map = rotulos  
 )  
   
 # Gera e salva a Tabela Wide (Coeficientes com Significância)  
 nome\_arquivo\_excel\_coef <- paste0("coeficientes\_quantil\_significancia\_", arquivo\_base, ".xlsx")  
 writexl::write\_xlsx(resultados$coeficientes\_wide, nome\_arquivo\_excel\_coef)  
   
 # Calcula e Salva Pseudo R² e AIC  
 ajuste\_df <- calcula\_ajuste(formula\_modelo, pnad\_tur\_Filtrada, quantis, Y)  
 nome\_arquivo\_excel\_ajuste <- paste0("resultados\_modelo\_quantis\_", arquivo\_base, ".xlsx")  
 writexl::write\_xlsx(ajuste\_df, nome\_arquivo\_excel\_ajuste)  
   
 # Gera e Salva o Gráfico Principal  
 grafico\_main <- gera\_grafico\_main(  
 df\_tidy = resultados$resultados\_tidy,  
 nome\_modelo = nome,  
 rotulos\_map = rotulos,  
 ordem\_cana = ordem\_canonica,  
 color = meu\_azul  
 )  
 nome\_arquivo\_grafico\_pdf <- paste0("grafico\_coeficientes\_", arquivo\_base, "\_filtrado.pdf")  
 ggplot2::ggsave(nome\_arquivo\_grafico\_pdf, grafico\_main, width = 20, height = 8)  
   
 # Gera e Salva o Gráfico Apenas de Interações  
 if (!is.null(modelo\_info$interacoes)) {  
 df\_interacoes <- resultados$resultados\_tidy %>% filter(str\_detect(term, ":"))  
   
 grafico\_interacoes <- gera\_grafico\_main(  
 df\_tidy = df\_interacoes,  
 nome\_modelo = paste(nome, " (Somente Interações)"),  
 rotulos\_map = rotulos,  
 ordem\_cana = ordem\_canonica,  
 color = meu\_azul,  
 filtro\_termos\_excluir = "NENHUM\_TERMO\_ASSIM\_ESPERO"  
 )  
   
 nome\_arquivo\_grafico\_int\_pdf <- paste0("grafico\_coeficientes\_somente\_int\_", arquivo\_base, ".pdf")  
 ggplot2::ggsave(nome\_arquivo\_grafico\_int\_pdf, grafico\_interacoes, width = 20, height = 8)  
 }  
   
 # Retorna o nome do modelo para visualização do progresso, se necessário  
 return(nome)  
}  
  
# Parar o Cluster (MUITO IMPORTANTE!) ---  
stopCluster(cl)  
tempo\_fim <- Sys.time()  
tempo\_total <- tempo\_fim - tempo\_inicio  
  
cat("\n----------------------------------------------\n")  
cat("PROCESSAMENTO PARALELO CONCLUÍDO.\n")  
cat(paste("Modelos executados:", paste(unlist(resultados\_paralelos), collapse = ", "), "\n"))  
cat(paste("Tempo total gasto: ", format(tempo\_total), "\n"))  
cat("----------------------------------------------\n")  
  
  
# Após rodar todo o código da regressão quantílica veja o seu diretório de dados para acessar as saídas da análise.  
  
  
## SALVANDO O CÓDIGO--------------------------------  
  
setwd("C:/Users/Ana Oliveira/Desktop/pnadc 2023")   
# Carregue os Pacotes  
library(knitr)  
library(rmarkdown)  
  
# Defina o Nome do Seu Arquivo R (NOME EXATO)  
NOME\_DO\_SEU\_CODIGO <- "Analises Tese Junho com Seleção Amostral e correções 03-10-25.R"  
  
# Execução  
  
# Gerando o R Markdown (.Rmd)  
# knit=FALSE para garantir que a execução não comece nesta fase.  
knitr::spin(NOME\_DO\_SEU\_CODIGO, knit = FALSE)