Assessing the Transportability of Cardiovascular Risk Prediction Models from the Framingham Heart Study to Target Population

Abstract

Objective : This study evaluates the transportability of cardiovascular disease prediction models using simulated target populations and compares the result with the result from the transportability using actual dataset. Utilizing data from the Framingham Heart Study and NHANES (National Health and Nutrition Examination Survey), we explore how variations in data distribution and correlation affect model performance, specifically in terms of MSE/Brier Score, relative bias, and empirical standard error (SE).

Method : Our approach includes a full-factorial simulation study assessing three design factors: the number of correlated columns, correlation scores, and the underlying data distribution. We explore two levels of correlation among continuous data columns and two data distribution shapes – the Framingham distribution and a normal distribution. The study also considers the number of observations and statistical summary from NHANES to simulate datasets, alongside the cardiovascular disease model developed from the Framingham data.

Result Key findings indicate that datasets with minimal correlation and lognormal distributions yield the best outcomes, closely aligning with source population characteristics. This is evidenced by the smallest MSE, low relative bias, and empirical SE, suggesting the model's ability to produce consistent and reliable estimates. The study also reveals that models perform better when adhering to the source population's original distribution, and overestimating correlations tends to worsen performance metrics.

1. Introduction

Predictive models are typically designed to generate predictions for a specific demographic. For instance, a healthcare system may develop a risk prediction model to identify patients at high risk for cardiovascular events among its care recipients. The data informing the development of such models, known as source study data, often come from controlled experiments, extensive observational databases, or longitudinal studies. However, these datasets usually do not represent a random sampling from the intended demographic, resulting in a discrepancy between the target population and the one represented in the source study data. Take, for example, the widely used Framingham ATP-III model from Wilson(1998), which predicts the ten-year risk of cardiovascular events; it was developed using data primarily from white subjects, which may lead to suboptimal predictions for ethnically diverse populations.

Recently, a variety of techniques have been developed to assess the effectiveness of predictive models within a specified target population (or to adapt model performance metrics from the source to the target population) such as transportability analysis in Steingrimsson (2023). Our aim is to apply these methods to a risk score model created with data from the Framingham Heart Study, and then to evaluate the model's performance using a simulated study population drawn from the NHANES (National Health and Nutrition Examination Survey) data. Our objective is to compare the transportability result based on the existng data with result from simulation based approach.

2. Data

n this study, we reference two datasets: 1) The Framingham Heart Study, and 2) The National Health and Nutrition Examination Survey Data (NHANES).

The Framingham Heart Study (Wilson, 1998) is a long-term, prospective investigation into the causes of cardiovascular disease within a cohort of free-living individuals in Framingham, Massachusetts. This dataset includes clinical examination data such as cardiovascular disease risk factors and indicators like blood pressure. Additionally, it documents whether the participants have experienced cardiovascular events, including myocardial infarction (hospitalized and silent or unrecognized), fatal coronary heart disease, atherothrombotic infarction, cerebral embolism, intracerebral hemorrhage, subarachnoid hemorrhage, or fatal cerebrovascular disease. We regard this dataset as the source population, which we use to construct a model to identify individuals at high risk for cardiovascular events.

Conversely, NHANES data (NHANES, 1999-2004) represents a nationally representative sample of adults and children in the United States. Data collection encompasses detailed, face-to-face interviews, physical and physiological examinations, and laboratory tests, some of which overlap with data from the Framingham Heart Study. However, NHANES does not include long-term outcome information such as cardiovascular disease events. We consider this dataset

Table 1: Framingham Statistics Summary Stratified by Sex

	Sex=1(Male)	Sex=2(Female)	P-Value
n	1094	1445	
SEX (mean (SD))	1.00 (0.00)	2.00 (0.00)	<0.001
HDLC (mean (SD))	43.63 (13.37)	53.07 (15.67)	<0.001
TOTCHOL (mean (SD))	226.44 (41.49)	246.32 (45.51)	<0.001
AGE (mean (SD))	60.01 (8.18)	60.55 (8.40)	0.106
SYSBP (mean (SD))	138.94 (20.89)	139.94 (23.71)	0.272
CURSMOKE (mean (SD))	0.39 (0.49)	0.31 (0.46)	<0.001
DIABETES (mean (SD))	0.09 (0.28)	0.07 (0.25)	0.037
BPMEDS (mean (SD))	0.11 (0.32)	0.18 (0.38)	<0.001

the target population and aim to determine how well a model built from the source population performs with this data.

The common variables extracted from both datasets include: 1) TOTCHOL, serum total cholesterol (mg/dL); 2) SYSBP, systolic blood pressure; 3) AGE, age at examination; 4) HDLC, high-density lipoprotein cholesterol (mg/dL); 5) SEX, participant sex; 6) CURSMOKE, current cigarette smoking at examination; 7) DIABETES, diabetic status; 8) BPMEDS, use of anti-hypertensive medication at examination.

In terms of preprocessing, we generate two new variables derived from SYSBP and BPMEDS. The new variables are SYSBP_T, representing the systolic blood pressure for participants using anti-hypertensive medication. For participants on such medication, we retain the SYSBP value; otherwise, we assign a 0 score. The second variable, SYSBP_UT, represents the systolic blood pressure for participants not using anti-hypertensive medication. Here, if the participant does not take the medication, we use the SYSBP value; if they do, we assign a 0 score.

There are some missing data in the NHANES dataset and we decided to use complete dataset going forward because in the transportability analysis we prefer to keep the data as original as possible without introducing some error in the dataset.

Below is the statistics summary of the Framingham and NHANES data (complete case).

3. Method

3.1. Analysis Plan

We begin by constructing a logistic regression model using the Framingham data to predict the occurrence of cardiovascular disease, with the model stratified by sex. We transform the variables TOTCHOL, SYSBP_T, SYSBP_UT, AGE, and HDLC by taking their logarithmic

Table 2: NHANES Statistics Summary Stratified by Sex

	Sex=1(Male)	Sex=2(Female)	P-Value
n	746	760	
SEX (mean (SD))	1.00 (0.00)	2.00 (0.00)	<0.001
HDLC (mean (SD))	47.08 (14.15)	57.43 (16.36)	<0.001
TOTCHOL (mean (SD))	177.14 (42.36)	194.63 (42.21)	<0.001
AGE (mean (SD))	61.88 (13.45)	62.98 (12.87)	0.106
SYSBP (mean (SD))	134.79 (18.90)	138.58 (21.52)	<0.001
CURSMOKE (mean (SD))	0.17 (0.38)	0.14 (0.35)	0.086
DIABETES (mean (SD))	0.35 (0.48)	0.26 (0.44)	<0.001
BPMEDS (mean (SD))	0.85 (0.35)	0.86 (0.35)	0.768

values. Additionally, we incorporate binary predictors, such as CURSMOKE and DIABETES, into the models.

To assess the model's transportability to the target population, we utilize the equation outlined in Steingrimsson (2023) to calculate the mean squared error (MSE) for the target population. This calculation also requires the computation of a weighting estimator, which utilizes inverse odds weights derived from a model that predicts the probability of belonging to the source population based on covariates. We combine data from both the source and target populations to establish these weights. We assign a membership indicator (S), where 1 indicates data from the source population and 0 denotes data from the target population.

The formula for transportability analysis that we use in this study is based on Steingrimsson (2023)

$$\hat{\psi}_{\hat{\beta}} = \frac{\Sigma_{i=1}^{n} I(S_i = 1)\hat{o}(X_i)(Y_i - g_{\hat{\beta}}(X_i))^2}{\Sigma_{i=1}^{n} I(S_i = 0)}$$

The inverse weights we use are

$$\hat{o}(X) = \frac{Pr(S = 0|X)}{Pr(S = 1|X)}$$

Given that we have access to the NHANES data, we conduct two types of evaluations. The first uses the actual, non-simulated NHANES dataset, and the second employs a simulated dataset. For the latter scenario, we simulate a situation where we lack access to the individual records of the NHANES data and have only the dataset's statistical summary. This approach reflects a common real-world constraint where full datasets like NHANES are unavailable, and only summary statistics of the target population are accessible.

3.2 GLM model

Two types of generalized linear models (GLMs) are utilized in this study, and each is constructed separately for men and women.

The first model is designed to predict the occurrence of cardiovascular disease, with separate models based on sex.

$$\begin{aligned} logit(E[Y]) = {} & \beta_o + \beta_1 log(HDLC) + \beta_2 log(TOTCHOL) + \beta_3 log(AGE) \\ & + \beta_4 log(SYSBP_{UT} + 1) + \beta_5 log(SYSBP_T + 1) + \beta_6 CURSMOKE + \beta_7 DIABETES \end{aligned}$$

The second model aims to predict the probability of membership, again with separate models based on sex. This model is structured similarly to the first but differs in terms of the outcome variable.

$$\begin{aligned} logit(E[S]) = {} & \beta_o + \beta_1 log(HDLC) + \beta_2 log(TOTCHOL) + \beta_3 log(AGE) \\ & + \beta_4 log(SYSBP_{UT} + 1) + \beta_5 log(SYSBP_T + 1) + \beta_6 CURSMOKE + \beta_7 DIABETES \end{aligned}$$

3.1 Transportability Analysis

3.3. Non Simulated Target Population

As previously mentioned, to calculate the mean squared error (MSE) for the target population, we first need to merge our datasets. This is applicable to both the non-simulated and simulated target populations.

In the case of the non-simulated dataset, the process is more straightforward. We need to merge the Framingham and NHANES datasets, filtering to include only complete records. We then assign the membership values accordingly. Once the datasets are combined, we apply the first GLM model to estimate Y (in this case, Y represents CVD). Subsequently, we need to apply the second GLM model to determine \$Pr(S=1|X)\$ and subsequently calculate the inverse weight for each record. We apply both models separately for men and women and then amalgamate the results. Finally, we can estimate the MSE for the target population.

3.4. Simulated Target Population

Factors such as simulation size, random seeds, collected measures, data generation, and simulation parameters are described and justified.

Using the ADEMP framework, below is the design for evaluating the model using simulated target population.

Table 3: NHANES Statistics Summary Stratified by Sex

	Sex=1(Male)	Sex=2(Female)	P-Value
n	746	760	
SEX (mean (SD))	1.00 (0.00)	2.00 (0.00)	<0.001
LOG.HDLC (mean (SD))	3.81 (0.27)	4.01 (0.27)	<0.001
LOG.TOTCHOL (mean (SD))	5.15 (0.24)	5.25 (0.21)	<0.001
LOG.AGE (mean (SD))	4.10 (0.25)	4.12 (0.24)	0.090
LOG.SYSBP (mean (SD))	4.89 (0.14)	4.92 (0.15)	0.001
CURSMOKE (mean (SD))	0.17 (0.38)	0.14 (0.35)	0.086
DIABETES (mean (SD))	0.35 (0.48)	0.26 (0.44)	<0.001
BPMEDS (mean (SD))	0.85 (0.35)	0.86 (0.35)	0.768

3.4.1. Data Generation Method

When considering which parameters to vary, we look to the target population MSE/Brier Score formula. For instance, inaccuracies in specifying the distribution underlying the target population data or introducing correlations in the target population dataset could lead to less accurate predictions of CVD_hat, estimated by a generalized linear model. This, in turn, could result in a larger target population MSE/Brier Score. In light of this, we conduct a full-factorial simulation study examining three design factors: the number of correlated columns, correlation scores for these columns, and the underlying distribution of the data. The values for each factor are as follows:
1) The number of correlated columns (corr_n) is set at two levels: 2 (continuous data partially correlated with each other) and 4 (continuous data full correlated with each other).
2) Correlation scores (corr) for the correlated columns vary across four levels: 0 (no correlation), 0.3 (weak correlation), 0.7 (moderate correlation), and 1 (true correlation).
3) The underlying distribution of the data (shape) is considered at two levels: following the Framingham distribution and using a normal distribution.
Additionally, the shared parameters and models for data generation include:
1) The number of observations (num_obs), set at 2539, matching the source population observations. 2) The statistical summary from the NHANES data, used to simulate all datasets. 3) The model fitted for cardiovascular disease events from the source population.

Below, we present the underlying statistical summary for the NHANES data with log transformation in the Table 3 to generate data with lognormal distribution. We will also refer to Table 2 for the statistical summary for normal distribution.

To determine the underlying distribution of the Framingham continuous data, we refer to the histogram plots in the Figure 1. These plots suggest that the Framingham continuous data closely resemble lognormal distributions and the distribution of the log transformation of the continuous variables resemble normal distribution. For the categorical data, we assume Bernoulli distributions.

Histogram of Framingham Continuous Variables

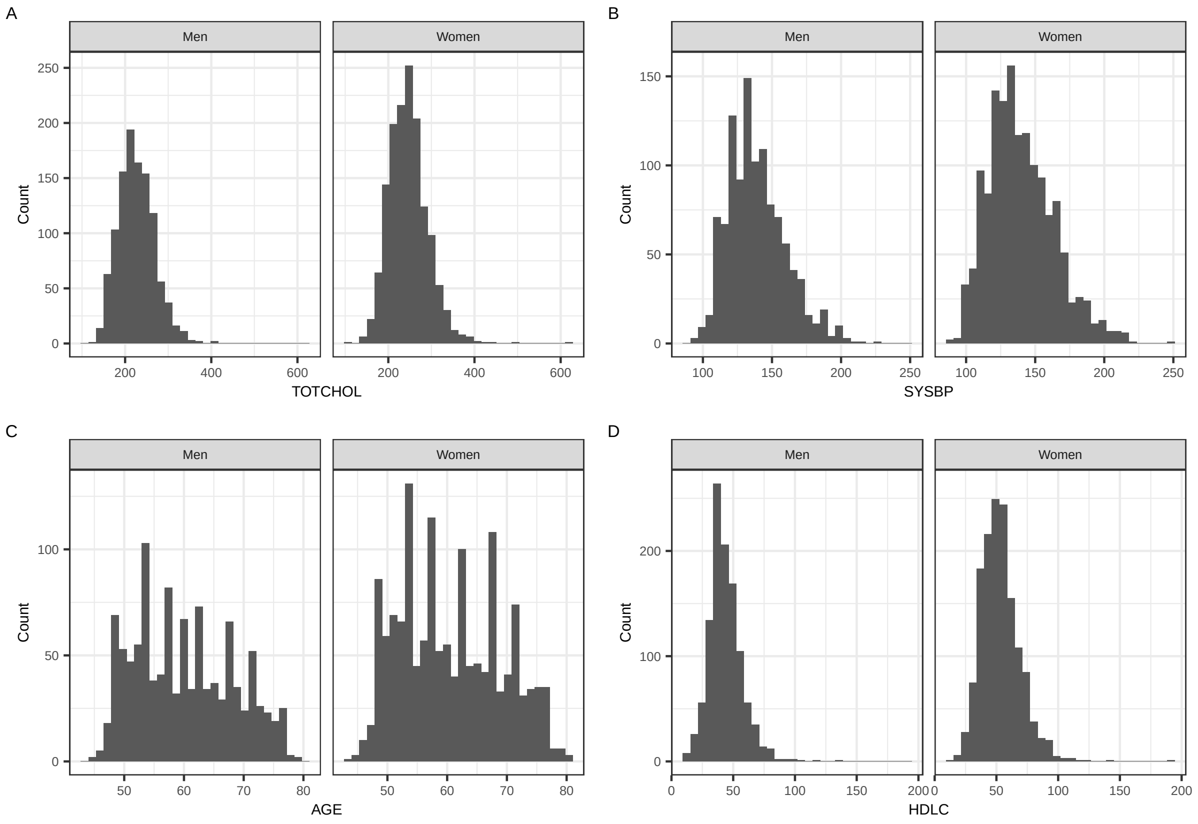

Figure 1. Histogram of Framingham Continuous Variables

Histogram of Framingham Log Continuous Variables

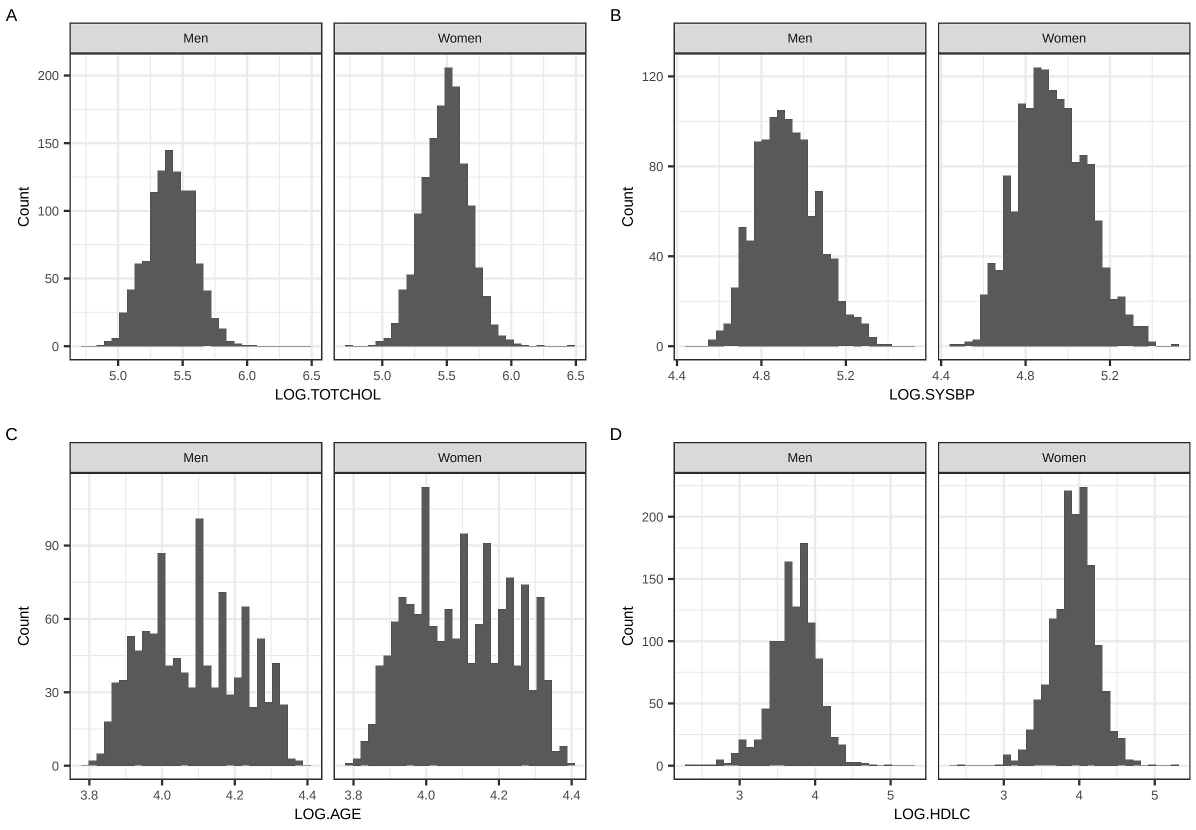

Figure 2. Histogram of Framingham Log Continuous Variables

We could also see from the Table 2 that the continuous data from source population has moderate correlation among each other.

Table 4: Correlation of Source Population Continuous Data

	TOTCHOL	HDLC	AGE	SYSBP
TOTCHOL	1.0000000	0.1816020	0.0890339	0.1025918
HDLC	0.1816020	1.0000000	-0.0072341	-0.0122821
AGE	0.0890339	-0.0072341	1.0000000	0.3341253
SYSBP	0.1025918	-0.0122821	0.3341253	1.0000000

3.4.2. Estimand

The estimand in this simulation study is the target population MSE/Brier Score.

3.4.3. Method

In each simulated target population dataset, we generate data using combinations of correlation values, the number of correlated columns, and different shapes, as previously specified. We then merge the source population data with the simulated target population dataset to perform the transportability analysis and obtain the estimand.

3.4.4. Performance Measure

In this simulation study, we do not have access to the true parameter, so we focus on the relative bias and empirical standard error (SE), with relative bias being the key performance measure of interest. However, as we have the target population data from the non-simulated dataset, we can treat it as the true estimand for comparison purposes. This allows us to compare results from both non-simulated and simulated data. Therefore, mean squared error (MSE) as additional performance measures, assuming that the transportability results from the non-simulated data represent the true estimand.

Regarding the number of repetitions for each simulated dataset, we based our assumptions on the variance of the estimand (determined from an initial small simulation run) is 0.002 (men) and 0.079 (women). We aimed for the required Monte Carlo SE of bias to be lower than 0.005 since the estimates are quite small. Using this information, we calculated that $n_{iter} = 3154$ is necessary.

4. Result and Discussion

The result from the non simulated dataset (NHANES) 0.1267 for men model and 0.0957 for women model. It suggests that the model's predictions in the target population have a mean squared error of 0.1267 for men model 0.0957 for women model. The women model performs better compared to the men model probably because the women in the NHANES data has fewer comorbidities (e.g. lower diabetes proportion in population). But both indicates a reasonable level of accuracy and the original models could be reliably used in the target population.

Transportability Analysis Performance Measures for Men

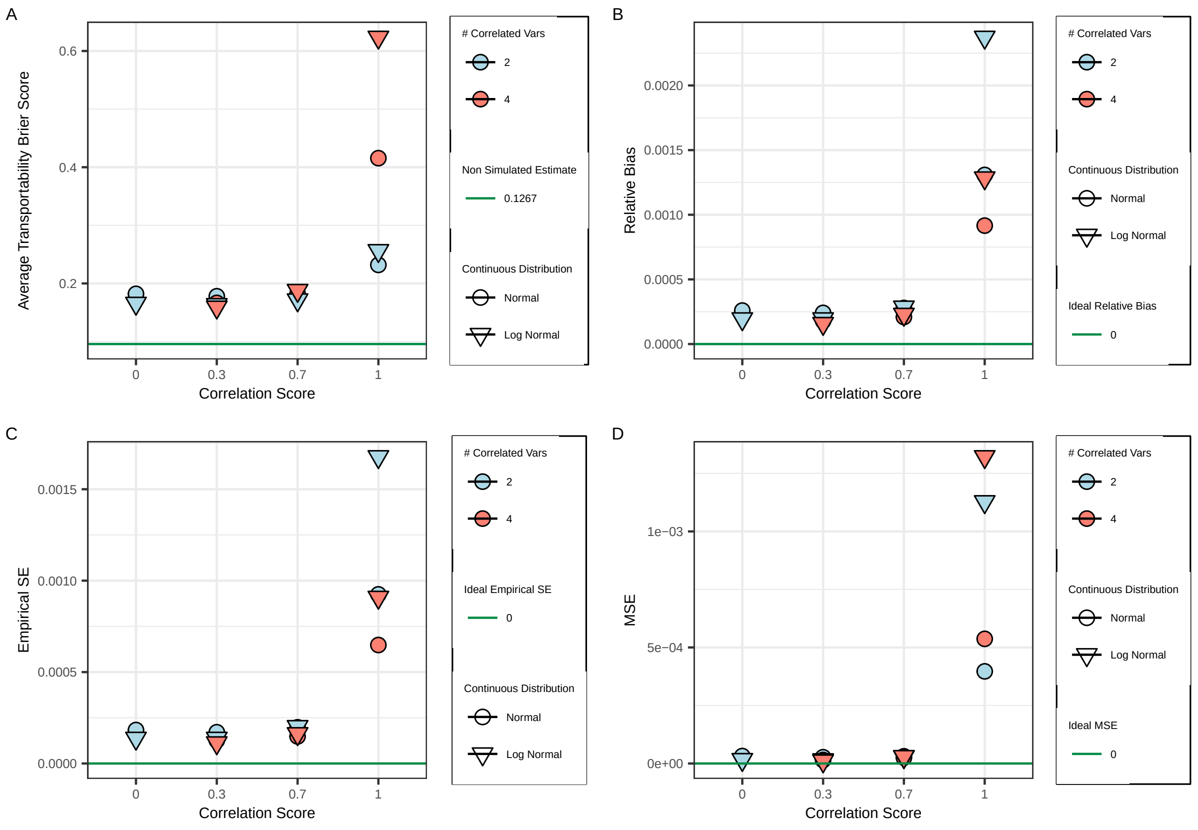

Figure 3. Transportability Analysis Performance Measures for Men

The results in Figure 3 and Figure 4 showed that models for women still perform better, and the best performing combinations for both men and women were those with distribution shapes similar to the original Framingham dataset and with no or moderate correlation. In particular, the men model has the lowest relative bias, empirical SE, MSE, and average brier score (0.00016, 0.00011, 0.00001, and 0.1592) when the dataset has moderate correlation (0.3) among all the continuous distribution and the continuous distribution follows the framingham's distribution shape (lognormal). In comparison, the women model perform best (0.00007 relative bias, 0.00005 empirical SE, 0.0000 MSE, 0.0974 average brier score) when the continuous variables do not have any correlation among each other and the continuous distribution follows the framingham's distribution shape. However the difference with the dataset that follows normal distribution and has no correlation among the continuous variables is close.

In general, both models tend to perform better (with smaller MSE, relative bias, and empirical SE) when they adhere to the original distribution of the source population and has minimal correlation among the continuous columns. This makes sense, as the dataset closely mimics the

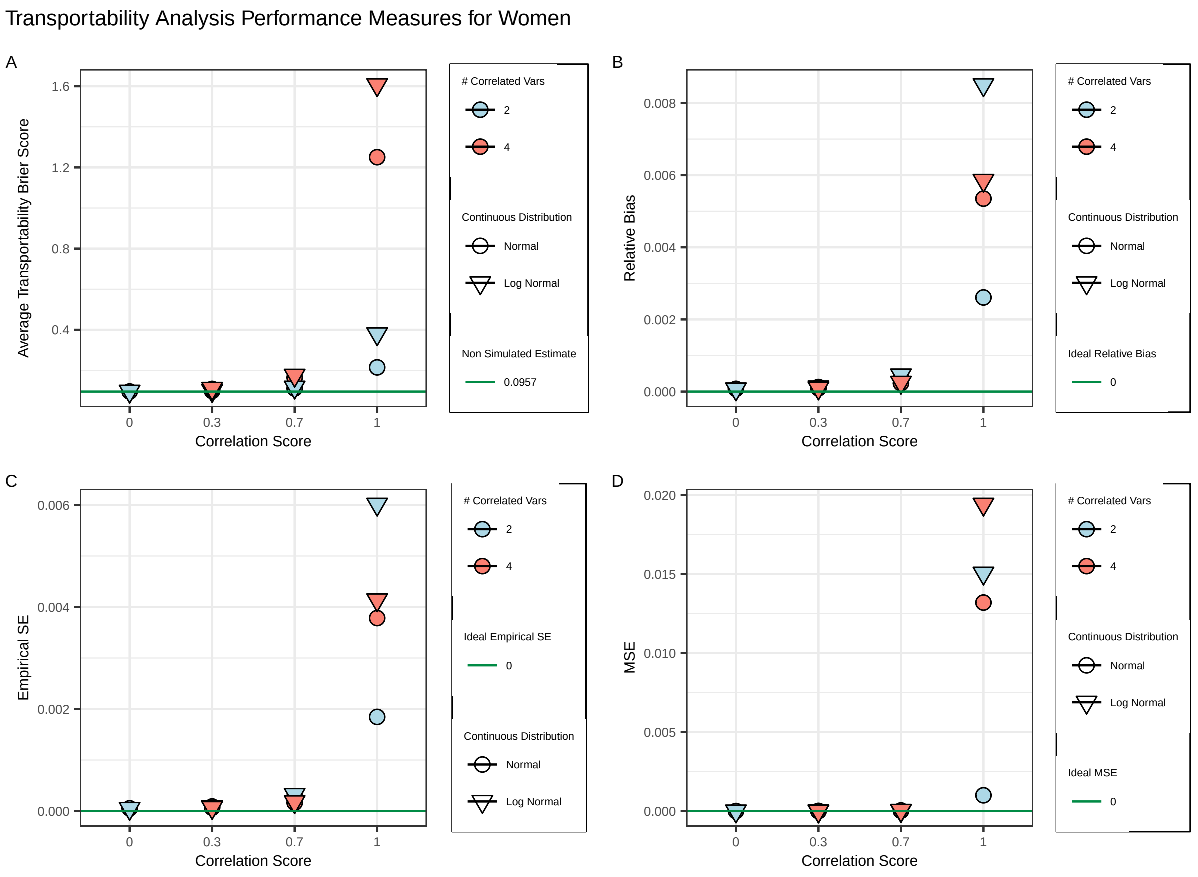

Figure 4. Transportability Analysis Performance Measures for Women

distribution and correlation patterns found in the source population. It also suggests that by closely following the source population's distribution and correlation, we can achieve estimates that are quite close to those of the non-simulated target population MSE/Brier Score. It is also evident that overestimating the correlation in the data leads to poorer outcomes across all metrics compared to underestimating the correlation.

In conclusion, to simulate a target population effectively for a cardiovascular disease prediction model, it's best to mimic the source population as closely as possible, both in terms of the continuous distribution's shape and the correlation patterns.

5. Limitation

This study is subject to several limitations. Firstly, our analysis is limited to assessing whether the continuous distribution of the data corresponds with that of the source population or adheres to a normal distribution. Secondly, there is an underlying assumption that all continuous variables in the Framingham data follow a lognormal distribution, based on the apparent closeness of fit. However, it's important to note that the age data might not strictly conform to a lognormal distribution in reality. Thirdly, in our approach, we uniformly apply the same correlation values across all correlated columns, a scenario that is highly improbable in real-world data. Lastly, we only incorporate positive correlations. Improvement in these areas could could be crucial for a more accurate and comprehensive analysis.

Supplemental Material

Supplemental material can be seen in this github page

Reference

Wilson, P. W., D'Agostino, R. B., Levy, D., Belanger, A. M., Silbershatz, H., & Kannel, W. B. (1998). Prediction of coronary heart disease using risk factor categories. Circulation, 97(18), 1837-1847. https://doi.org/10.1161/01.cir.97.18.1837

Steingrimsson, J. A., Gatsonis, C., Li, B., & Dahabreh, I. J. (2023). Transporting a prediction model for use in a new target population. American Journal of Epidemiology, 192(2), 296-304. https://doi.org/10.1093/aje/kwac128

Centers for Disease Control and Prevention. (1999-2004). National Center for Health Statistics. National Health and Nutrition Examination Survey Data. U.S. Department of Health and Human Services. http://www.cdc.gov/nchs/nhanes.htm

Code Appendix

```
#############
### SETUP ###
#############

library(formatR)

knitr::opts_chunk$set(echo = TRUE)
knitr::opts_chunk$set(message = F)
knitr::opts_chunk$set(warning = F)
knitr::opts_chunk$set(fig.align="center")
knitr::opts_chunk$set(fig.width=8, fig.height=6)

###############
### LIBRARY ###
###############
library(riskCommunicator)
library(tidyverse)
library(tableone)
library(MASS)
library(truncnorm)
library(ggplot2)
library(kableExtra)
library(patchwork)
# The NHANES data here finds the same covariates among this national survey data
library(nhanesA)
############
### SEED ###
############

set.seed(7)
########################
### DEFINE FUNCTIONS ###
########################

brier_transport = function(df) {
  #' Calculate the target population MSE/Brier Score based on Steingrimsson (2023)
  #' @param df, dataset including source population and target population
  #' @param return, MSE/Brier Score from df
  # Separate Source and Target Population
  s1 = df %>% filter(S==1)
  s0 = df %>% filter(S==0)

  # Calculate the nominator from the target population MSE/Brier Score
  s1 = s1 %>%
    mutate(noms = WEIGHT * (CVD - CVD_HAT)^2)
  nom = sum(s1$noms)

  # Calculate the denominator from the target population MSE/Brier Score
  denom = nrow(s0)

  # Calculate target population MSE/Brier Score
  result = nom/denom
  return(result)
}

transportability_analysis <- function(source_pop, target_pop, dist_shape='normal') {
  #' Calculate the target population MSE/Brier Score from specified source
  #'    population dataset and target population dataset for men and women
  #' @param source_pop, dataset of source population
  #' @param target_pop, dataset of target population
  #' @param return, brier score for target population for men and women

  # Combine source and target population
  combined_df = rbind(source_pop, target_pop)
  # Separate men and women data from the combined dastaset
  combined_df_men = combined_df %>% filter(SEX == 1)
  combined_df_women = combined_df %>% filter(SEX == 2)

  # Make the membership probability model for bot men and women
  pr_s_mod_men <- glm(S ~ log(HDLC)+log(TOTCHOL)+log(AGE)+log(SYSBP_UT+1)
                     +log(SYSBP_T+1)+CURSMOKE+DIABETES
                     , data = combined_df_men, family= "binomial")

  pr_s_mod_women <- glm(S ~ log(HDLC)+log(TOTCHOL)+log(AGE)+log(SYSBP_UT+1)
                       +log(SYSBP_T+1)+CURSMOKE+DIABETES
                       , data = combined_df_women, family= "binomial")

  # Only take relevant columns to be modeled
  combined_df_mod = combined_df[,c('HDLC','TOTCHOL','AGE','SYSBP_UT'
```

```r
                                    ,'SYSBP_T','CURSMOKE','DIABETES')]

  # Get the Pr(S=1|X) for men and women
  pr_S_m = predict(pr_s_mod_men, newdata = combined_df_mod, type = 'response')
  pr_S_f = predict(pr_s_mod_women, newdata = combined_df_mod, type = 'response')
  # Get the inverse weight for men and women
  weight_m =  (1-pr_S_m) / pr_S_m
  weight_f =  (1-pr_S_f) / pr_S_f

  # Get the CVD_hat (y_hat) for men and women
  cvd_hat_m = predict(mod_men, newdata = combined_df_mod , type = 'response')
  cvd_hat_f = predict(mod_women, newdata = combined_df_mod , type = 'response')

  # Put the inverse weight and CVD_hat to the combined dataset
  combined_df = combined_df %>%
    mutate(WEIGHT_M = weight_m
           ,WEIGHT_F = weight_f
           ,CVD_HAT_M = cvd_hat_m
           ,CVD_HAT_F = cvd_hat_f) %>%
    mutate(WEIGHT = ifelse(SEX == 1, WEIGHT_M, WEIGHT_F)
           ,CVD_HAT = ifelse(SEX == 1, CVD_HAT_M, CVD_HAT_F)) %>%
    dplyr::select(-c(WEIGHT_M,WEIGHT_F,CVD_HAT_M,CVD_HAT_F))


  # Calculate Target Population MSE/Brier Score
  result_m = brier_transport(combined_df %>% filter(SEX == 1))
  result_f = brier_transport(combined_df %>% filter(SEX == 2))
  return(c(result_m,result_f))
}

data_gen <- function(n, corr_any, corr_n, corr, dist_shape) {
  #' Generate dataset based on NHANES statistics summary
  #'    and use Framingham continuous distribution shape (ori) or use normal dist
  #' @param n, number of observations in dataset
  #' @param corr_any, apply correlation in continuous dataset, Yes or No
  #' @param corr_n, how many continuous columns that have correlation
  #' @param corr, correlation magnitude
  #' @param dist_shape, distribution shape for the continuous columns
  #' @param return, final dataset

  # Stopping Criterions
```

```r
if(!corr_any %in% c(TRUE, FALSE)) {
  stop("corr_any must be TRUE/FALSE")
}

if((corr_n == FALSE) & (corr_n !=0)) {
  stop("corr_n must be 0 if corr_any is FALSE")
}

if((corr_n == FALSE) & (!corr_n %in% c(2,3,4))) {
  stop("corr_n must be 2/3/4 if corr_any is TRUE")
}

if(abs(corr) > 1) {
  stop("Corr cannot be more than 1 or less than -1")
}

if(!dist_shape %in% c('normal', 'ori')) {
  stop("corr_any must be ori/normal")
}

# Determine the columns that has correlation
if(corr_any == TRUE) {
  corr_idxs = sample(1:4, corr_n)
}

# Initialize final dataset
final_df = data.frame()
# Generate data for men (SEX = 1) and women (SEX = 2)
for (i in 1:2) {
  if (i == 1) {
    # Generate categorical columns for men bases on NHANES statistics summary
    n_sex = round(n * 0.50)
    sex = rep(i, n_sex)
    bpmeds = rbinom(n_sex, 1, 0.85)
    cursmoke = rbinom(n_sex, 1, 0.17)
    diabetes = rbinom(n_sex, 1, 0.35)

    # Specify mean and sd for men bases on NHANES statistics summary
    # 1)TOTCHOL, 2)SYSBP, 3)AGE, 4)HDLC
    # If distribution shape is similar to framingham/lognormal
    if(dist_shape == 'ori') {
      means = c(5.15, 4.89, 4.10, 3.81)
      sds = c(0.24, 0.14, 0.25, 0.27)
    }

    # If distribution shape is normal
    if(dist_shape == 'normal') {
      means = c(177.14, 134.79, 61.88, 47.08)
      sds = c(42.36, 18.90, 13.45, 14.15)
    }

  }
  else {
    # Generate categorical columns for women bases on NHANES statistics summary
    n_sex = n - round(n * 0.49)
    sex = rep(i, n_sex)
    bpmeds = rbinom(n_sex, 1, 0.86)
    cursmoke = rbinom(n_sex, 1, 0.14)
    diabetes = rbinom(n_sex, 1, 0.09)

    # Specify mean and sd for women bases on NHANES statistics summary
    # 1)TOTCHOL, 2)SYSBP, 3)AGE, 4)HDLC
    # If distribution shape is similar to framingham/lognormal
    if(dist_shape == 'ori') {
      means = c(5.25, 4.92, 4.12, 4.01)
      sds = c(0.21, 0.15, 0.24, 0.27)
    }

    # If distribution shape is normal
    if(dist_shape == 'normal') {
      means = c(194.63,138.58,62.98,57.43)
      sds = c(42.21,21.52,12.87,16.36)
    }
  }

  # Generate continuous columns based on determined dist_shape
  continuous_list = vector('list',4)
  for (j in 1:4) {
    # Get mean and sd for the corresponding continuous columns
    # 1)TOTCHOL, 2)SYSBP, 3)AGE, 4)HDLC
    m = means[j]
    s = sds[j]
    # If dist_shape == 'ori' then use framingham distribution shape, lognormal
    if (dist_shape == 'ori') {
      dat = rnorm(n_sex, m, s)
    }

    # If dist_shape == 'normal' then use normal distribution shape
    if (dist_shape == 'normal') {
      dat = rnorm(n_sex, m, s)
      # For age, truncate the data, min = 1, max = 81 (based on framingham)
      if(j == 3) {dat = rtruncnorm(n = n_sex,a = 1,b = 81,mean = m,sd = s)}
    }

    # Combine all continuous columns
    continuous_list[[j]] = dat
  }

  # Make the continuous columns correlated if corr_any == TRUE
  if (corr_any == TRUE) {

    # Generate multivariate normal dataset that correspond with the corr value
    corr_mat = matrix(corr, ncol = corr_n, nrow = corr_n)
    diag(corr_mat) = 1
    mvdat = mvrnorm(n = n_sex, mu = rep(0,corr_n), Sigma = corr_mat, empirical = TRUE)

    # Apply the correlation to the existing continuous columns by using the
    #   sorted position of the multivariate normal dataset
    k = 1
    for (corr_idx in corr_idxs) {

      # Sort multivariate normal dataset
      rnk = rank(mvdat[ , k], ties.method = "first")
      # Arrange the continuous columns based on the previous sorting
      dat_sorted = sort(continuous_list[[corr_idx]])
      # Replace the original continuous columns
      continuous_list[[corr_idx]] = dat_sorted[rnk]
      k = k + 1
    }

    # Remove mvdat var to avoid mvrnomr overwrite error
    rm(mvdat)
  }
```

```r
    # Make dataset based on the continuous and categorical columns
    continuous_df = data.frame(TOTCHOL = continuous_list[[1]]
                               , SYSBP = continuous_list[[2]]
                               , AGE = continuous_list[[3]]
                               , HDLC = continuous_list[[4]])
    if(dist_shape == 'ori') {
      continuous_df = continuous_df %>%
        mutate(TOTCHOL = exp(TOTCHOL)
               , SYSBP = exp(SYSBP)
               , AGE = exp(AGE)
               , HDLC = exp(HDLC))
    }
    continuous_df = continuous_df[sample(nrow(continuous_df)),]

    df_tmp = data.frame(SEX = sex
                        , BPMEDS = bpmeds
                        , CURSMOKE = cursmoke
                        , DIABETES = diabetes
                        , continuous_df)

    # Append to final dataset
    final_df = rbind(final_df, df_tmp)
  }

  # Preprocess dataset so the structure is accepted by the models
  final_df =  final_df %>%
    mutate(S = 0
           , CVD = NA
           , SYSBP_UT = ifelse(BPMEDS == 0, SYSBP, 0)
           , SYSBP_T = ifelse(BPMEDS == 1, SYSBP, 0)
    ) %>%
    dplyr::select(HDLC, TOTCHOL, AGE, SYSBP, BPMEDS, SYSBP_UT, SYSBP_T
                  , CURSMOKE, DIABETES, SEX, S, CVD)

  # Return dataset
  return(final_df)

}

# save(brier_transport, file = 'brier_transport.Rda')
# save(transportability_analysis, file = 'transportability_analysis.Rda')
# save(data_gen, file = 'data_gen.Rda')
load('brier_transport.Rda')
load('transportability_analysis.Rda')
load('data_gen.Rda')
#######################
### PRERPOCESS DATA ###
#######################

data("framingham")

# The Framingham data has been used to create models for cardiovascular risk.
# The variable selection and model below are designed to mimic the models used
# in the paper General Cardiovascular Risk Profile for Use in Primary Care
# This paper is available (cvd_risk_profile.pdf) on Canvas.

framingham_df <- framingham %>% dplyr::select(c(CVD, TIMECVD, SEX, TOTCHOL, AGE,
                                      SYSBP, DIABP, CURSMOKE, DIABETES, BPMEDS,
                                      HDLC, BMI))
framingham_df <- na.omit(framingham_df)

# CreateTableOne(data=framingham_df, strata = c("SEX"))

# Get blood pressure based on whether or not on BPMEDS
framingham_df$SYSBP_UT <- ifelse(framingham_df$BPMEDS == 0,
                                 framingham_df$SYSBP, 0)
framingham_df$SYSBP_T <- ifelse(framingham_df$BPMEDS == 1,
                                framingham_df$SYSBP, 0)

# Looking at risk within 15 years - remove censored data
# dim(framingham_df)
framingham_df <- framingham_df %>%
  filter(!(CVD == 0 & TIMECVD <= 365*15)) %>%
  dplyr::select(-c(TIMECVD))
# dim(framingham_df)

save(framingham_df, file = 'framingham_df.Rda')

# Filter to each sex
framingham_df_men <- framingham_df %>% filter(SEX == 1)
framingham_df_women <- framingham_df %>% filter(SEX == 2)
```

```r
save(framingham_df_men, file = 'framingham_df_men.Rda')
save(framingham_df_women, file = 'framingham_df_women.Rda')

# blood pressure, demographic, bmi, smoking, and hypertension info
bpx_2017 <- nhanes("BPX_J") %>%
  dplyr::select(SEQN, BPXSY1 ) %>%
  rename(SYSBP = BPXSY1)
demo_2017 <- nhanes("DEMO_J") %>%
  dplyr::select(SEQN, RIAGENDR, RIDAGEYR) %>%
  rename(SEX = RIAGENDR, AGE = RIDAGEYR)
bmx_2017 <- nhanes("BMX_J") %>%
  dplyr::select(SEQN, BMXBMI) %>%
  rename(BMI = BMXBMI)
smq_2017 <- nhanes("SMQ_J") %>%
  mutate(CURSMOKE = case_when(SMQ040 %in% c(1,2) ~ 1,
                              SMQ040 == 3 ~ 0,
                              SMQ020 == 2 ~ 0)) %>%
  dplyr::select(SEQN, CURSMOKE)
bpq_2017 <- nhanes("BPQ_J") %>%
  mutate(BPMEDS = ifelse(BPQ050A == 1, 1, 0)) %>%
  dplyr::select(SEQN, BPMEDS)
tchol_2017 <- nhanes("TCHOL_J") %>%
  dplyr::select(SEQN, LBXTC) %>%
  rename(TOTCHOL = LBXTC)
hdl_2017 <- nhanes("HDL_J") %>%
  dplyr::select(SEQN, LBDHDD) %>%
  rename(HDLC = LBDHDD)
diq_2017 <- nhanes("DIQ_J") %>%
  mutate(DIABETES = case_when(DIQ010 == 1 ~ 1,
                              DIQ010 %in% c(2,3) ~ 0,
                              TRUE ~ NA)) %>%
  dplyr::select(SEQN, DIABETES)

# Join data from different tables
df_2017 <- bpx_2017 %>%
  full_join(demo_2017, by = "SEQN") %>%
  full_join(bmx_2017, by = "SEQN") %>%
  full_join(hdl_2017, by = "SEQN") %>%
  full_join(smq_2017, by = "SEQN") %>%
  full_join(bpq_2017, by = "SEQN") %>%
  full_join(tchol_2017, by = "SEQN") %>%
  full_join(diq_2017, by = "SEQN")

save(df_2017, file = 'df_2017.Rda')
# CreateTableOne(data = df_2017, strata = c("SEX"))

#################
### FIT MODEL ###
#################

# Fit models with log transforms for all continuous variables
mod_men <- glm(CVD~log(HDLC)+log(TOTCHOL)+log(AGE)+log(SYSBP_UT+1)+
                 log(SYSBP_T+1)+CURSMOKE+DIABETES,
      data= framingham_df_men, family= "binomial")

mod_men <- glm(CVD~log(HDLC)+log(TOTCHOL)+log(AGE)+log(SYSBP_UT+1)+
                 log(SYSBP_T+1)+CURSMOKE+DIABETES,
      data= framingham_df_men, family= "binomial")

save(mod_men, file = 'mod_men.Rda')

mod_women <- glm(CVD~log(HDLC)+log(TOTCHOL)+log(AGE)+log(SYSBP_UT+1)+
                   log(SYSBP_T+1)+CURSMOKE+DIABETES,
               data= framingham_df_women, family= "binomial")

save(mod_men, file = 'mod_women.Rda')

load('framingham_df.Rda')
load('framingham_df_men.Rda')
load('framingham_df_women.Rda')
load('mod_men.Rda')
load('mod_women.Rda')
load('df_2017.Rda')
#########################################
### PREPARE SOURCE POP DATA FOR MODEL ###
#########################################

# Prepare the source population dataset for the models
framingham_prep = framingham_df %>%
  dplyr::select(c(HDLC,TOTCHOL,AGE,SYSBP,BPMEDS,CURSMOKE,DIABETES,SEX,CVD)) %>%
  # Generate SYSBP_UT and SYSBP_T, combination of SYSBP and BPMEDS
  # Generate S, the membership indication
```

```r
  mutate(SYSBP_UT = ifelse(BPMEDS == 0, SYSBP, 0)
         ,SYSBP_T = ifelse(BPMEDS == 1, SYSBP, 0)
         ,S = 1) %>%
  dplyr::select(c(HDLC,TOTCHOL,AGE,SYSBP,BPMEDS,SYSBP_UT,SYSBP_T,CURSMOKE
                  ,DIABETES,SEX,S,CVD))

# save(framingham_prep, file = 'framingham_prep.Rda')
###################################################
### PREPARE TARGET POP (NON SIM) DATA FOR MODEL ###
###################################################

# Prepare the non simulated target population dataset for the models
nhanes_prep = df_2017 %>%
  dplyr::select(c(HDLC,TOTCHOL,AGE,SYSBP,BPMEDS,CURSMOKE,DIABETES,SEX)) %>%
  # Take complete data only
  na.omit() %>%
  # Generate SYSBP_UT and SYSBP_T, combination of SYSBP and BPMEDS
  # Generate S, the membership indication
  mutate(SYSBP_UT = ifelse(BPMEDS == 0, SYSBP, 0)
         ,SYSBP_T = ifelse(BPMEDS == 1, SYSBP, 0)
         ,S = 0) %>%
  mutate(CVD = NA) %>%
  dplyr::select(c(HDLC,TOTCHOL,AGE,SYSBP,BPMEDS,SYSBP_UT,SYSBP_T,CURSMOKE
                  ,DIABETES,SEX,S,CVD))

# save(nhanes_prep, file = 'nhanes_prep.Rda')
load('framingham_stat_tb.Rda')
load('nhanes_stat_tb.Rda')

# Print statistics summary of Framingham
framingham_stat_tb %>%
  kableExtra::kbl(caption = 'Framingham Statistics Summary Stratified by Sex'
                  , booktabs = T
                  , escape = T
                  , align = 'c'
                  , col.names = c('Sex=1(Male)','Sex=2(Female)', 'P-Value','')) %>%
  kableExtra::kable_classic(full_width = F
                            , html_font = 'Cambria'
                            , font_size = 5
                            # , position = 'float_left'
                            , latex_options = 'scale_down')
```

```
# Print statistics summary of NHANES
nhanes_stat_tb %>%
  kableExtra::kbl(caption = 'NHANES Statistics Summary Stratified by Sex'
                  , booktabs = T
                  , escape = T
                  , align = 'c'
                  , col.names = c('Sex=1(Male)','Sex=2(Female)', 'P-Value','')) %>%
  kableExtra::kable_classic(full_width = F
                            , html_font = 'Cambria'
                            , font_size = 5
                            # , position = 'right'
                            , latex_options = 'scale_down')
##############################
### GET NHANES LOG SUMMARY ###
##############################

nhanes_log = nhanes_prep %>%
  mutate(LOG.TOTCHOL = log(TOTCHOL)
         ,LOG.SYSBP = log(SYSBP)
         ,LOG.AGE = log(AGE)
         ,LOG.HDLC = log(HDLC)) %>%
  dplyr::select(SEX,LOG.HDLC,LOG.TOTCHOL,LOG.AGE,LOG.SYSBP,CURSMOKE,DIABETES,BPMEDS)

lognhanes_stat = CreateTableOne(data = nhanes_log, strata = c("SEX"))
lognhanes_stat_tb = print(lognhanes_stat$ContTable)
# save(lognhanes_stat_tb,file='lognhanes_stat_tb.Rda')
load('lognhanes_stat_tb.Rda')

# Print statistics summary of NHANES
lognhanes_stat_tb %>%
  kableExtra::kbl(caption = 'NHANES Statistics Summary Stratified by Sex'
                  , booktabs = T
                  , escape = T
                  , align = 'c'
                  , col.names = c('Sex=1(Male)','Sex=2(Female)', 'P-Value','')) %>%
  kableExtra::kable_classic(full_width = F
                            , html_font = 'Cambria'
                            , font_size = 5
                            , latex_options = 'scale_down')
##################################################
### OBSERVE DISTRIBUTIONS FROM SOURCE POP DATA ###
```

```
##################################################

# Non Log
# Get histogram from the framingham dataset continuous columns
fram_hist = vector('list',4)
cont_cols = c('TOTCHOL', 'SYSBP', 'AGE', 'HDLC')
it = 1
sex_label = c(`1` = 'Men',`2`='Women')

for (colname in cont_cols) {
  pl = ggplot(framingham_df) +
    geom_histogram(aes_string(x = colname)) +
    facet_grid(~SEX, labeller = as_labeller(sex_label)) +
    theme_bw() +
    labs(x = colname
         ,y = 'Count') +
    theme(text = element_text(size=7))

  fram_hist[[it]] = pl
  it = it + 1
}

patchworkHist = (fram_hist[[1]] + fram_hist[[2]] + fram_hist[[3]]
                 + fram_hist[[4]]) + plot_layout(nrow = 2, ncol = 2)
save(patchworkHist, file='patchworkHist.Rda')

# Log
# Get histogram from the framingham dataset continuous columns
fram_hist2 = vector('list',4)
cont_cols2 = c('LOG.TOTCHOL', 'LOG.SYSBP', 'LOG.AGE', 'LOG.HDLC')
it = 1
sex_label = c(`1` = 'Men',`2`='Women')

for (colname in cont_cols2) {
  pl = ggplot(framingham_log) +
    geom_histogram(aes_string(x = colname)) +
    facet_grid(~SEX, labeller = as_labeller(sex_label)) +
    theme_bw() +
    labs(x = colname
         ,y = 'Count') +
    theme(text = element_text(size=7))
  fram_hist2[[it]] = pl
  it = it + 1
}

patchworkHist2 = (fram_hist2[[1]] + fram_hist2[[2]] + fram_hist2[[3]]
                  + fram_hist2[[4]]) + plot_layout(nrow = 2, ncol = 2)
# save(patchworkHist2, file='patchworkLogHist2.Rda')
load('patchworkHist.Rda')
patchworkHist +
  plot_annotation(tag_levels = 'A'
                  ,title = "Histogram of Framingham Continuous Variables"
                  ,caption = 'Figure 1. Histogram of Framingham Continuous Variables'
                  ,theme = theme(plot.title = element_text(size = 10)
                                 ,plot.tag = element_text(size = 3)))

load('patchworkHist2.Rda')
patchworkHist2 +
  plot_annotation(tag_levels = 'A'
                  ,title = "Histogram of Framingham Log Continuous Variables"
                  ,caption = 'Figure 2. Histogram of Framingham Log Continuous Variables'
                  ,theme = theme(plot.title = element_text(size = 10)
                                 ,plot.tag = element_text(size = 3)))
#########################################
### OBSERVE CORR FROM SOURCE POP DATA ###
#########################################

cor(framingham_df[,c('TOTCHOL','HDLC','AGE','SYSBP')]) %>%
  kableExtra::kbl(caption = 'Correlation of Source Population Continuous Data'
                  , booktabs = T
                  , escape = T
                  , align = 'c') %>%
  kableExtra::kable_classic(full_width = F
                            , font_size = 5
                            , html_font = 'Cambria'
                            , latex_options = 'HOLD_position')
####################################################
### GENERATE INITIAL ESTIMATES FOR EXTREME CASES ###
####################################################

init_briers = replicate(2000, {
  dataa = data_gen(n=2359, corr_any=TRUE, corr_n=4, corr=0.99, dist_shape='normal')
```

```r
  transportability_analysis(framingham_prep, dataa)
})

# save(init_briers, file = 'init_briers.Rda')
load('init_briers.Rda')

# Calculate number of iterations for each simulation
n_iters = var(init_briers)/(0.001)^2
load('framingham_prep.Rda')
load('nhanes_prep.Rda')

##################################################
### TRANSPORTABILITY ANALYSIS FOR NON SIM DATA ###
##################################################

# Apply transportability analysis to the non simulated target population dataset
nonsimulated_result = transportability_analysis(framingham_prep, nhanes_prep)
# nonsimulated_result # men 0.12673996 women 0.09572181
##############################################
### TRANSPORTABILITY ANALYSIS FOR SIM DATA ###
##############################################

corr_ns = c(2,4)
corrs = c(0, 0.3, 0.7, 1)
shapes = c('ori','normal')
num_rep = 3154
num_obs = 2539

# Simulation to get the target population MSE/Brier score based on the
#   specified values of shapes and corrs
outputs = data.frame()
for (shape in shapes) {
  for (corr in corrs) {
    for (corr_n in corr_ns) {

      if (corr == 0) {corr_any = FALSE}
      else {corr_any = TRUE}

      results = replicate(num_rep, {
        dataa = data_gen(n = num_obs
                         , corr_any = corr_any, corr_n = corr_n, corr = corr
```

```
                          , dist_shape = shape)
        transportability_analysis(framingham_prep, dataa)
      })
      outputs_tmp <- data.frame(ta = results, shape = shape, corr = corr, corr_n = corr_n)
      outputs = rbind(outputs, outputs_tmp)
    }
  }
}

# save(outputs, file = 'outputs2.Rda')
load('outputs2.Rda')
mc_bias <- function(estimates) {
  #' Calculate monte carlo relative bias using the mean estimates instead of
  #'    true estimates
  #' @param estimates, set of estimates
  #' @param return, monte carlo relative bias of the estimates
  nsim = length(estimates)
  mean_estimates = mean(estimates)
  est_diff_squared = (estimates - mean_estimates)^2
  result = sqrt(1/(nsim *(nsim-1)) * sum(est_diff_squared))
  return(result)
}

mc_empSE <- function(estimates) {
  #' Calculate monte carlo empirical SE
  #' @param estimates, set of estimates
  #' @param return, monte carlo empirical SE
  nsim = length(estimates)
  mean_estimates = mean(estimates)
  est_diff_squared = (estimates - mean_estimates)^2
  est_emp_SE = sqrt(1/(nsim-1) * sum(est_diff_squared))
  result = est_emp_SE / sqrt(2*(nsim-1))
  return(result)
}

mc_MSE <- function(estimates, true_estimate) {
  #' Calculate monte carlo MSE
  #' @param estimates, set of estimates
  #' @param return, monte carlo MSE
  nsim = length(estimates)
  est_diff_squared = (estimates - true_estimate)^2
```

```r
  est_MSE = (1/(nsim) * sum(est_diff_squared))
  result = sqrt( sum((est_diff_squared - est_MSE)^2) / ((nsim)*(nsim-1)))
  return(result)
}

# Calculate the performance measures
eval_summary <- outputs %>%
  group_by(shape, corr, corr_n) %>%
  mutate(relative_bias_m = mc_bias(ta_m)
         ,empSE_m = mc_empSE(ta_m)
         ,MSE_m = mc_MSE(ta_m, nonsimulated_result[1])
         ,avg_ta_m = mean(ta_m)
         ,relative_bias_f = mc_bias(ta_f)
         ,empSE_f = mc_empSE(ta_f)
         ,MSE_f = mc_MSE(ta_f, nonsimulated_result[2])
         ,avg_ta_f = mean(ta_f)
  )%>%
  ungroup() %>%
  dplyr::select(-c('ta_m','ta_f')) %>%
  group_by(shape, corr, corr_n) %>%
  slice(1) %>%
  mutate_if(is.numeric, round, 6)
load('eval_summary.Rda')
# Initialization
all_plot_men = vector('list',4)
all_plot_women = vector('list',4)
iter = 1

for (i in c('M','F')) {
  if(i == 'M') {
    metrics_col = c('relative_bias_m', 'empSE_m', 'MSE_m', 'avg_ta_m')
    nonsim_res = 0.1267

    # Avg TA Brier Score
    plot1 = ggplot(eval_summary) +
      geom_point(aes_string(x = 'factor(corr)', y = metrics_col[4]
                            , fill = 'factor(corr_n)', shape = 'shape')
                 , size = 3) +
      geom_hline(data = eval_summary, aes(yintercept = nonsim_res
                                          , colour = "0.1267"), show_guide=TRUE) +
      scale_shape_manual(name= 'Continuous Distribution'
                         ,values = c(21,25),labels=c('Normal','Log Normal')) +
      scale_fill_manual(name= '# Correlated Vars'
                        ,values = c('lightblue','salmon')) +
      scale_color_manual(name = 'Non Simulated Estimate'
                         , values = c("0.1267" = "springgreen4"))+
      theme_bw() +
      labs(x = 'Correlation Score'
           ,y = 'Average Transportability Brier Score') +
      theme(text = element_text(size=7)
            ,legend.title=element_text(size=5)
            ,legend.text=element_text(size=5)
            ,legend.box.background = element_rect(color = "black")) +
      guides(fill = guide_legend("# Correlated Vars"
                                 , override.aes = list(shape = 21)))
  }

  if(i == 'F') {
    metrics_col = c('relative_bias_f', 'empSE_f', 'MSE_f', 'avg_ta_f')
    nonsim_res = 0.0957

    # Avg TA Brier Score
    plot1 = ggplot(eval_summary) +
      geom_point(aes_string(x = 'factor(corr)', y = metrics_col[4]
                            , fill = 'factor(corr_n)', shape = 'shape')
                 , size = 3) +
      geom_hline(data = eval_summary, aes(yintercept = nonsim_res
                                          , colour = "0.0957"), show_guide=TRUE) +
      scale_shape_manual(name= 'Continuous Distribution'
                         ,values = c(21,25),labels=c('Normal','Log Normal')) +
      scale_fill_manual(name= '# Correlated Vars'
                        ,values = c('lightblue','salmon')) +
      scale_color_manual(name = 'Non Simulated Estimate'
                         , values = c("0.0957" = "springgreen4"))+
      theme_bw() +
      labs(x = 'Correlation Score'
           ,y = 'Average Transportability Brier Score') +
      theme(text = element_text(size=7)
            ,legend.title=element_text(size=5)
            ,legend.text=element_text(size=5)
            ,legend.box.background = element_rect(color = "black")) +
      guides(fill = guide_legend("# Correlated Vars"
                                 , override.aes = list(shape = 21)))
  }

  # Avg Relative Bias
  plot2 = ggplot(eval_summary) +
    geom_point(aes_string(x = 'factor(corr)', y = metrics_col[1]
                          , fill = 'factor(corr_n)', shape = 'shape')
               , size = 3) +
    geom_hline(data = eval_summary, aes(yintercept = 0
                                        , colour = "0"), show_guide=TRUE) +
    scale_shape_manual(name= 'Continuous Distribution'
                       ,values = c(21,25),labels=c('Normal','Log Normal')) +
    scale_fill_manual(name= '# Correlated Vars'
                      ,values = c('lightblue','salmon')) +
    scale_color_manual(name = 'Ideal Relative Bias'
                       , values = c("0" = "springgreen4"))+
    theme_bw() +
    labs(x = 'Correlation Score'
         ,y = 'Relative Bias') +
    theme(text = element_text(size=7)
          ,legend.title=element_text(size=5)
          ,legend.text=element_text(size=5)
          ,legend.box.background = element_rect(color = "black")) +
    guides(fill = guide_legend("# Correlated Vars"
                               , override.aes = list(shape = 21)))

  # Avg EmpSE
  plot3 = ggplot(eval_summary) +
    geom_point(aes_string(x = 'factor(corr)', y = metrics_col[2]
                          , fill = 'factor(corr_n)', shape = 'shape')
               , size = 3) +
    geom_hline(data = eval_summary, aes(yintercept = 0
                                        , colour = "0"), show_guide=TRUE) +
    scale_shape_manual(name= 'Continuous Distribution'
                       ,values = c(21,25),labels=c('Normal','Log Normal')) +
    scale_fill_manual(name= '# Correlated Vars'
                      ,values = c('lightblue','salmon')) +
    scale_color_manual(name = 'Ideal Empirical SE'
                       , values = c("0" = "springgreen4"))+
    theme_bw() +
    labs(x = 'Correlation Score'
         ,y = 'Empirical SE') +
    theme(text = element_text(size=7)
          ,legend.title=element_text(size=5)
          ,legend.text=element_text(size=5)
          ,legend.box.background = element_rect(color = "black")) +
    guides(fill = guide_legend("# Correlated Vars"
                               , override.aes = list(shape = 21)))

  # Avg MSE
  plot4 = ggplot(eval_summary) +
    geom_point(aes_string(x = 'factor(corr)', y = metrics_col[3]
                          , fill = 'factor(corr_n)', shape = 'shape')
               , size = 3) +
    geom_hline(data = eval_summary, aes(yintercept = 0
                                        , colour = "0"), show_guide=TRUE) +
    scale_shape_manual(name= 'Continuous Distribution'
                       ,values = c(21,25),labels=c('Normal','Log Normal')) +
    scale_fill_manual(name= '# Correlated Vars'
                      ,values = c('lightblue','salmon')) +
    scale_color_manual(name = 'Ideal MSE'
                       , values = c("0" = "springgreen4"))+
    theme_bw() +
    labs(x = 'Correlation Score'
         ,y = 'MSE') +
    theme(text = element_text(size=7)
          ,legend.title=element_text(size=5)
          ,legend.text=element_text(size=5)
          ,legend.box.background = element_rect(color = "black")) +
    guides(fill = guide_legend("# Correlated Vars"
                               , override.aes = list(shape = 21)))


  if (i == 'M') {
    all_plot_men[[1]] = plot1
    all_plot_men[[2]] = plot2
    all_plot_men[[3]] = plot3
    all_plot_men[[4]] = plot4
  }
  if (i == 'F') {
    all_plot_women[[1]] = plot1
    all_plot_women[[2]] = plot2
```

```
    all_plot_women[[3]] = plot3
    all_plot_women[[4]] = plot4
  }
}

# Combine plots for men
patchwork_men = (all_plot_men[[1]] + all_plot_men[[2]] + all_plot_men[[3]]
                 + all_plot_men[[4]]) + plot_layout(nrow = 2, ncol = 2)
# save(patchwork_men, file='patchwork_men.Rda')

# Combine plots for men
patchwork_women = (all_plot_women[[1]] + all_plot_women[[2]] + all_plot_women[[3]]
                   + all_plot_women[[4]]) + plot_layout(nrow = 2, ncol = 2)
# save(patchwork_women, file='patchwork_women.Rda')
load('patchwork_men.Rda')
load('patchwork_women.Rda')

options(repr.plot.width=5, repr.plot.height=3)
patchwork_men +
  plot_annotation(tag_levels = 'A'
                  ,title = "Transportability Analysis Performance Measures for Men"
                  ,caption = 'Figure 3. Transportability Analysis Performance Measures for
                  ,theme = theme(plot.title = element_text(size = 10)
                                 ,plot.tag = element_text(size = 3)))


options(repr.plot.width=5, repr.plot.height=3)
patchwork_women +
  plot_annotation(tag_levels = 'A'
                  ,title = "Transportability Analysis Performance Measures for Women"
                  ,caption = 'Figure 4. Transportability Analysis Performance Measures for
                  ,theme = theme(plot.title = element_text(size = 10)
                                 ,plot.tag = element_text(size = 3)))
```