Abstract

A major problem surrounding the release of datasets of experimental studies is how to maintain the privacy of the respondents. The corresponding risk of identity disclosure drastically limits sharing of scientific data. An increasingly plausible solution to reducing identity disclosure risk is to generate synthetic datasets, that is, datasets that are generated artificially to replace some sample data. In this article, CART and parametric methods of generating synthetic data are compared on a chosen dataset by calculating disclosure risks and data utility measures. In addition, the overall quality and usefulness of the synthetic datasets with respect to the original data will be discussed. And finally, the measures will be generated on differing number of synthetic datasets, and the results will be compared.

Introduction

Privacy and identity disclosure risk

A major problem surrounding the release of datasets of experimental studies is how to maintain the privacy of the respondents. In releasing a dataset, there is a risk of a person's identity being identified since certain variables such as gender, ethnicity, and location may, when combined together, cause a person to be identified. This is especially a possibility if the person is part of an underrepresented group. This risk of identity disclosure is problematic because identity disclosure can prove to be a danger to a person's safety, reputation, and mental health. In fact, identity disclosure risk is one of the reasons why public sharing of datasets is incredibly rare: in a sample of published psychological journal articles, only 2.1% provided statements on data availability and only 1.6% provided access to the actual data.[1] This limit of access to data prevents other researchers from being able to check the code and methodology, and reproduce the results in the original paper. Furthermore, it also prevents researchers from further developing and exploring the novel elements in the original research.

Methods of reducing disclosure risk

A few approaches to decrease identity disclosure risk that have developed over time have included 1) using data masking and 2) generation of synthetic data.[1]

Data masking refers to methods of replacing sensitive data with other data; for instance, by substituting, replacing, or swapping data.[4] A major pitfall of data masking, however, is that once the masking is strong enough to lessen the dangers of identity disclosure risk, the relationships between variables are often unable to be preserved, which makes the model and any predictions made inaccurate.[4] On the other hand, if the magnitude of the masking is insufficient, then privacy may be compromised.

The second approach is generating synthetic data, that is, data that is artificially created in order to replace sensitive data. This approach is becoming more prevalent given that it holds distinct advantages over data masking as patterns and relationships in the data may still be preserved without sacrificing too much detail. An important multiple imputation (MI) approach to generating synthetic data was proposed by Rubin in 1993. In this approach, observed data is used to create an imputation model and then samples are drawn from the posterior predictive distribution.[2] Later, a nonparametric approach using classification and regression trees was proposed (CART). This is of particular interest since it no longer requires an imputation model (which often can be very complex) to be specified.[3]

Objective of project

In this project, two approaches to generating synthetic data will be tested: in particular, the CART versus the parametric approach on a new dataset concerning adults with Erosive Esophagitis [5]. Given the nature and rarity of the disease in question, this would be a relevant dataset to test all the synthetic data generation approaches on since it contains extremely sensitive information.
The objective of the project will be to answer a few essential questions:

1. Is either method of generating synthetic data better than the other?
2. Does the synthetic data generated have low disclosure risk?
3. Does the synthetic data generated have high data utility?
4. What happens to disclosure risk and data utility when the number of synthetic datasets generated increases?

While there have been at least several papers published on such comparisons, it is important that the methods and comparisons be performed on additional datasets to confirm conclusions drawn previously. Different datasets may produce different results so it is beneficial to do additional comparisons in order to see if results still align when data is different. In particular, consistent results across multiple papers serves as additional evidence to the effectiveness of certain approaches or methods.

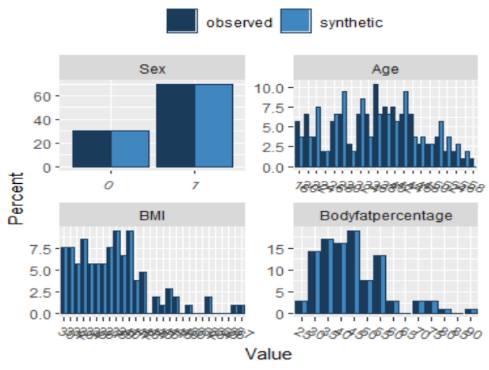
Creating synthetic datasets

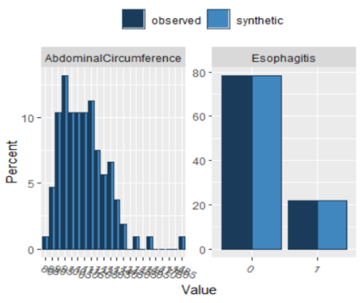
The dataset *Association between the body mass index, waist circumference, and body fat percentage with erosive esophagitis in adults with obesity after sleeve gastrectomy* consists of 106 observations with the following variables:

sex, age, hypertension, blood pressure, glucose, height, weight, body fat percentage, esophagitis, and many others. For this project, we focus only the variables: Sex, Age, BMI, Abdominal Circumference and Esophagitis. The rest of the dataset is removed and we name the new truncated dataset *'data'*.

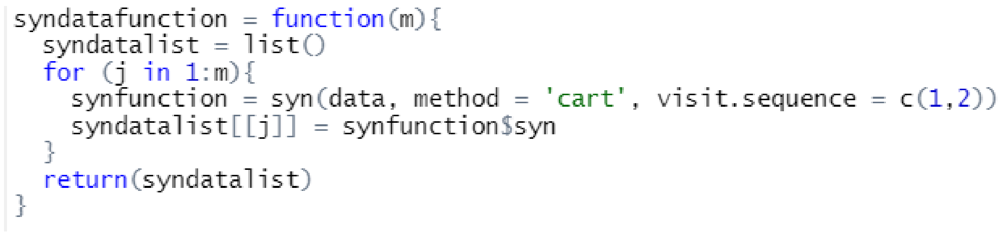
```
library(readxl)
#create data.RData
data = read_excel("/Users/jianf/Documents/winter 2022/stat 499/dataset.xlsx")
data = data[,c(1,2,19,20,21,24)]
colnames(data) = c('Sex','Age','BMI','Bodyfatpercentage','AbdominalCircumference','Esophagitis')
data = data.frame(data)
save(data, file = "/Users/jianf/Documents/winter 2022/stat 499/data.RData")
```

For this dataset, sex and age are chosen as the quasi-identifying variables. This is because sex and age can easily identify an individual as compared with the other variables. Therefore, synthetic data is generated for sex and age only and the other variables are kept the same. The *synthpop* package in R is used to generate the synthetic data. In particular, the function syn(data, method = 'cart', visit.sequence = c(1,2)) generates a synthetic copy of the dataset *data* using the nonparametric cart method. Also, syn(data, method = 'parametric', visit.sequence = c(1,2)) would do the same for the parametric method. Note that the parameter visit.sequence c(1,2) implies that synthetic data only be generated for the variables sex and age.





But suppose we would want m synthetic datasets to be generated. The function syndatafunction(m) is created to generate a list of m synthetic datasets for the dataset data using the nonparametric method cart. Note that we are creating a function instead of using the default parameter m in syn() because when $m = 1$, then syn()$syn doesn't return a list as it does when $m > 1$. So a new function was created to make life a little bit easier.



```
syndatafunction = function(m){
  syndatalist = list()
  for (j in 1:m){
    synfunction = syn(data, method = 'cart', visit.sequence = c(1,2))
    syndatalist[[j]] = synfunction$syn
  }
  return(syndatalist)
}
```

On the other hand, a function syndatafunction_p(m) is created in order to generate m synthetic datasets using the parametric method.

Disclosure Risk

Definition

Disclosure risk measures the risk that a person's identity may be identified from the synthetic datasets generated. Clearly, disclosure risk should be minimized as much as possible when generating datasets and should ideally be close to 0%.

Calculating Disclosure Risk

There are several measures of disclosure risk; in particular, we measure disclosure risk through what is known as the *true match rate*. The procedure is listed in 2.5 Disclosure Risk Measures [2], but here written in more accessible language.

We defined a match (between observed and synthetic data value) to occur when sex is equal and the ages are within 1 of each other.

Step 1: Let us consider $i = 1$, the 1st observed value. The 1st observed value from the original dataset is compared with each j-th synthetic data value for $1 \leq j \leq 106$ in the first synthetic dataset. The total number of matches is counted, call this value r_1. Define the probability of matching on the j-th synthetic value as $\frac{1}{r_1}$ if $r_1 > 0$ and 0 if $r_1 = 0$; this yields a vector of match probabilities of length 106. Now, repeat this for the other synthetic datasets (there are m of them). Then, m sets of vectors of matching probabilities are obtained. We take an average entrywise over all m vectors to yield a new single vector of match probabilities. Call the vector of match probabilities s_1.

Step 2: Note: s_1 is a vector of 106 entries. Let c_1 be the number of entries in s_1 that have the highest match probability. Let T_1 be 1 if the 1st synthetic element is one of the elements in c_1 (has the highest match probability) and T_1 be 0 otherwise. Finally, let $I_1 = 1$ if $c_1 T_1 = 1$ and 0 otherwise.

Step 3: Now, the same thing is performed for the $i = 2, 3, \ldots, 106$ observed values. In other words, replace all the 1's in Steps 1 and 2 with i's. This yields values $I_1, I_2, \ldots, I_{106}$.

Step 4: Define the *true match rate* as $\frac{\sum_{i=1}^{n} I_i}{n}$. This is the disclosure risk that is outputted.

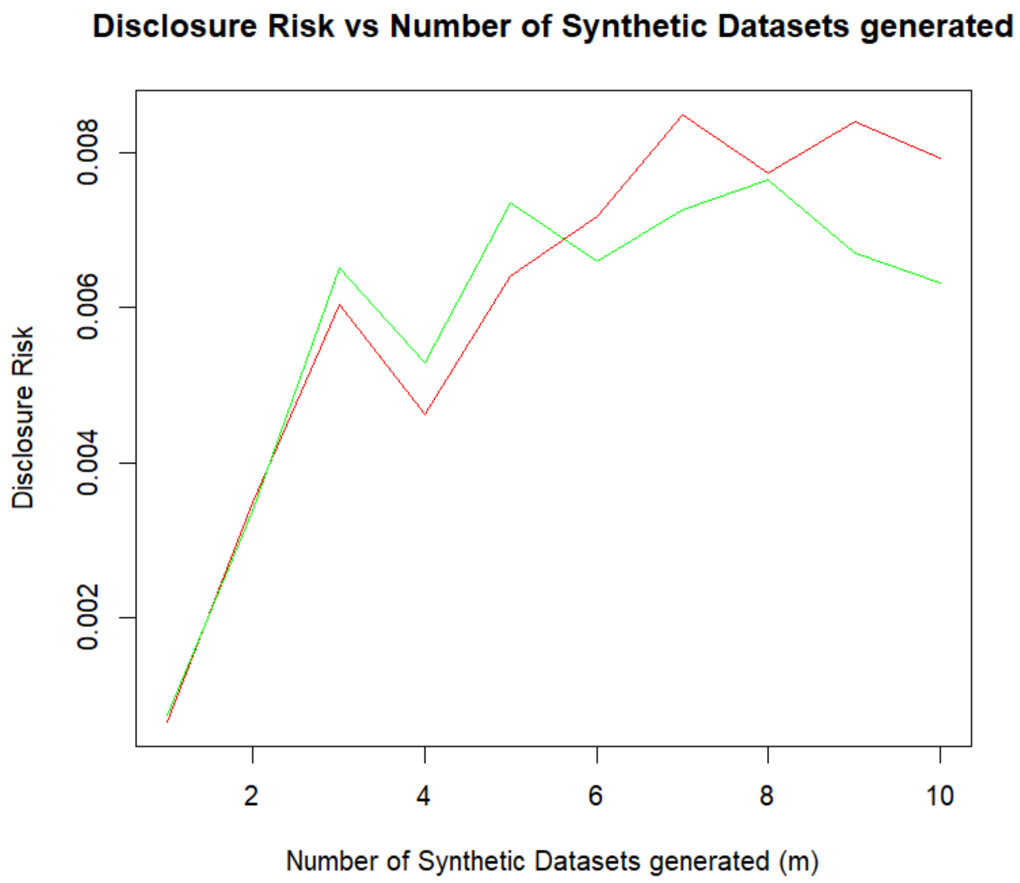
Comparisons

The function disclosure_risk() was called from the file disclosure_risk.R in Supplementary Material [2]. However, this function was modified slightly in order to fit this dataset. In this section, it will be seen how disclosure risk varies when different parameters are changed. In particular, m, the number of synthetic datasets generated and the method of generating the synthetic datasets (CART vs parametric) was varied.
At first, a function was originally created to investigate disclosure risk when varying m for the CART method of generating synthetic data. While calling this function multiple times, it seems like there is no perceivable trend. This is almost certainly due to the small sample size of $n = 106$ which would cause the disclosure risk to vary wildly and unfortunately, this renders it impossible to identify any patterns or relationships when varying the parameters.

In order to remedy this, a function generate_disclosure_risk_R() was created to create R = 100 replications of *each* of the combinations of m and method of synthetic generation. It then outputs a table of the mean of the R = 100 disclosure risks calculated for each of the combinations. The function calls syndatafunction(m) and disclosure_risk(n,m,syndata) over and over again in a *for loop*.

```r
generate_disclosure_risk_R = function(){
  set.seed(001)
  mean_list = mean_list_p = rep(NA,10)
  for (i in 1:10){
    mean_list[i] = mean(replicate(100,disclosure_risk(n,i,syndatafunction(i))))
    mean_list_p[i] = mean(replicate(100,disclosure_risk(n,i,syndatafunction_p(i))))
  }
  plot(1:10,mean_list,type='l',col='red',
       main='Disclosure Risk vs Number of Synthetic Datasets generated',
       xlab='Number of Synthetic Datasets generated (m)',
       ylab='Disclosure Risk')
  lines(1:10,mean_list_p,col='green')
}
```

After running the function, it yielded this plot:

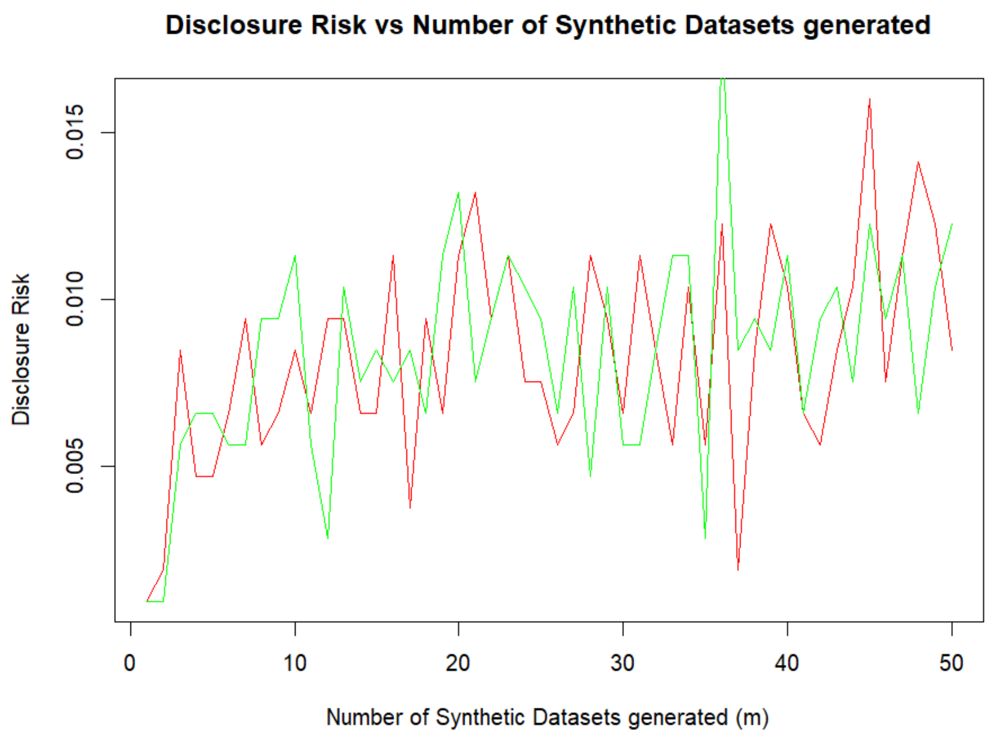


In the plot, the red colored line represents the CART method and the green colored line represents the parametric method.

From the replications above, it is clear that disclosure risk increases as the number of synthetic datasets generated increases. This is not surprising since as the number of synthetic datasets generated increases, the information obtained increases as well which leads to an increase in disclosure risk. Note that both methods have very similar disclosure risks for nearly all of the values of m and also follow an almost identical trend. It can be concluded that both methods perform nearly identically well with respect to minimizing disclosure risk.

What about if we also consider $m = 11, 12, 13, \ldots, 50$? Then, by modifying the above equation so that it calculates up to $m = 50$ and also changing the number of replications R to 10 (otherwise, there would be too many computations for the computer to handle!), this yields:

```
generate_disclosure_risk_R2 = function(){
  set.seed(001)
  mean_list = mean_list_p = rep(NA,50)
  for (i in 1:50){
    mean_list[i] = mean(replicate(10,disclosure_risk(n,i,syndatafunction(i))))
    mean_list_p[i] = mean(replicate(10,disclosure_risk(n,i,syndatafunction_p(i))))
  }
  plot(1:50,mean_list,type='l',col='red',
       main='Disclosure Risk vs Number of Synthetic Datasets generated',
       xlab='Number of Synthetic Datasets generated (m)',
       ylab='Disclosure Risk')
  lines(1:50,mean_list_p,col='green')
}
```



This does seem a bit disappointing as there does not seem to be a clear trend or a clear method that performs better than the others. But this may also be due to the low number of replications. Nevertheless, it can be seen that as m increases until around 10 that disclosure risks for both methods are increasing. Then, there is no clear trend as m increases further.

Also, taking the data from the first plot, if we average over $m = 1, 2, \ldots, 10$, then disclosure risk for CART is 0.006094340 and for parametric is 0.005783019. Now, taking the data from the second plot, if we average over $m = 1, 2, \ldots, 50$, then disclosure risk for CART is 0.008301887 and for parametric is 0.008433962. It can be seen that the syn() function generates low disclosure risk for both methods for this dataset. All the averages of the disclosure risks are well under 1%, which show that individuals in the dataset are unlikely to be identified, given that synthetic data is generated. Therefore, we conclude that generating synthetic data from either method would protect privacy of the respondents of this dataset quite well.

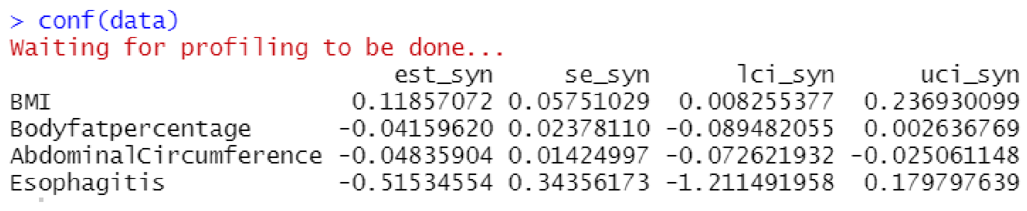
Data Utility

Confidence Intervals for original dataset

Let Y = Sex and define A,B,C,D be BMI, Body fat percentage, Abdominal Circumference, and Esophagitis respectively. Suppose a binary probit regression model of Y is fitted against A,B,C,D. Let a, b, c, d be the coefficients for A,B,C,D for the fit. Let $V(a), V(b), V(c), V(d)$ be the variance of the coefficients a, b, c, d respectively. Of course, the coefficients and variance of the coefficients are unknown parameters.
Let a', b', c', d' be the estimates of a, b, c, d respectively and let $v(a'), v(b'), v(c'), v(d')$ be estimates of $V(a'), V(b'), V(c'), V(d')$ respectively. The function conf(data) was created to generate 95% Confidence Intervals for any dataset.

```r
conf = function(data){
  fit = glm(data[,1] ~ ., data = data[,3:ncol(data)], family = binomial(probit))
  conf = data.frame(summary(fit)$coefficients[-1,1:2],confint(fit)[-1,1:2])
  colnames(conf) = c('est_syn','se_syn','lci_syn','uci_syn')
  return(conf)
}
```

Note that the lower bound of the confidence interval is lci_syn and the upper bound is uci_syn. Applying conf(data) on the original dataset yields us:



```
> conf(data)
Waiting for profiling to be done...
                          est_syn     se_syn      lci_syn      uci_syn
BMI                    0.11857072 0.05751029  0.008255377  0.236930099
Bodyfatpercentage     -0.04159620 0.02378110 -0.089482055  0.002636769
AbdominalCircumference -0.04835904 0.01424997 -0.072621932 -0.025061148
Esophagitis           -0.51534554 0.34356173 -1.211491958  0.179797639
```

Confidence Intervals for synthetic datasets

This procedure outlined below is explained in Section 2.3 Analysis of Synthetic Datasets [2]. However, the language used is very technical so it is explained here in more intuitive and accessible language. Let m be the number of synthetic datasets produced. Let Y_j denote the Sex variable data for the j-th synthetic dataset where $j = 1, 2, \ldots, m$. For each $j = 1, 2, \ldots, m$, a binary probit regression model of Y_j is fit against A,B,C,D. Note here that A,B,C,D are all original data since synthetic data was produced only for Sex and Age. Then for each $j = 1, 2, \ldots, m$, the estimated coefficients a_j, b_j, c_j, d_j and its corresponding estimated variances $v(a_j), v(b_j), v(c_j), v(d_j)$ are extracted for A,B,C,D.

Consider $\overline{a_j} = \frac{\sum_{j=1}^{m} a_j}{m}$, our estimator of a; $\overline{b_j} = \frac{\sum_{j=1}^{m} b_j}{m}$, our estimator of b; $\overline{c_j} = \frac{\sum_{j=1}^{m} c_j}{m}$, our estimator of c; $\overline{d_j} = \frac{\sum_{j=1}^{m} d_j}{m}$, our estimator of d. Note that for each of A,B,C,D, this is taking the mean of the coefficients of the variables over the m synthetic datasets.

Consider $\overline{v(a_j)} = \frac{\sum_{j=1}^{m} v(a_j)}{m}$, $\overline{v(b_j)} = \frac{\sum_{j=1}^{m} v(b_j)}{m}$, $\overline{v(c_j)} = \frac{\sum_{j=1}^{m} v(c_j)}{m}$, $\overline{v(d_j)} = \frac{\sum_{j=1}^{m} v(d_j)}{m}$. Note that for each of A,B,C,D, this is taking the mean of the variance of the coefficients over the m synthetic datasets.

Define

$$\hat{V}(\overline{a_j}) = \frac{(a_j - \bar{a_j})^2 + (b_j - \bar{a_j})^2 + (c_j - \bar{a_j})^2 + (d_j - \bar{a_j})^2}{m(m-1)} + \overline{v(a_j)}$$

as our estimate of $V(\overline{a_j})$;

$$\hat{V}(\overline{b_j}) = \frac{(a_j - \bar{b_j})^2 + (b_j - \bar{b_j})^2 + (c_j - \bar{b_j})^2 + (d_j - \bar{b_j})^2}{m(m-1)} + \overline{v(b_j)}$$

as our estimate of $V(\overline{b_j})$;

$$\hat{V}(\overline{c_j}) = \frac{(a_j - \bar{c_j})^2 + (b_j - \bar{c_j})^2 + (c_j - \bar{c_j})^2 + (d_j - \bar{c_j})^2}{m(m-1)} + \overline{v(c_j)}$$

as our estimate of $V(\overline{c_j})$;

$$\hat{V}(\overline{d_j}) = \frac{(a_j - \bar{d_j})^2 + (b_j - \bar{d_j})^2 + (c_j - \bar{d_j})^2 + (d_j - \bar{d_j})^2}{m(m-1)} + \overline{v(d_j)}$$

as our estimate of $V(\overline{d_j})$.

Furthermore, if n is large, the distributions of $\overline{a_j}, \overline{b_j}, \overline{c_j}, \overline{d_j}$ approximate to a t-distribution with respective degrees of freedom:

$$df_a = (m-1) + \frac{m\overline{v(a_j)}}{(a_j - \overline{a_j})^2 + ((b_j - \overline{a_j})^2 + (c_j - \overline{a_j})^2 + (d_j - \overline{a_j})^2})^2$$

$$df_b = (m-1) + \frac{m\overline{v(b_j)}}{(a_j - \overline{b_j})^2 + ((b_j - \overline{b_j})^2 + (c_j - \overline{b_j})^2 + (d_j - \overline{b_j})^2})^2$$

$$df_c = (m-1) + \frac{m\overline{v(c_j)}}{(a_j - \overline{c_j})^2 + ((b_j - \overline{c_j})^2 + (c_j - \overline{c_j})^2 + (d_j - \overline{c_j})^2})^2$$

$$df_d = (m-1) + \frac{m\overline{v(d_j)}}{(a_j - \overline{d_j})^2 + ((b_j - \overline{d_j})^2 + (c_j - \overline{d_j})^2 + (d_j - \overline{d_j})^2})^2$$

We note that we have the estimated coefficients, the estimated variances of the coefficients and the distribution with the estimates with associated degrees of freedom. Therefore, we can obtain confidence intervals for $\overline{a_j}, \overline{b_j}, \overline{c_j}, \overline{d_j}$.

Note that the identical methods and calculations can also be done when Y = Age. In this case, we fit a linear model of Y = Age against A,B,C,D and continue as above.

Consider the function combine_syn() in combine.R in Supplementary Material [2]. This function takes synthetic datasets and outputs coefficients, standard errors, and 95% Confidence Intervals based on the above methodology. Note that the function was modified accordingly to fit this dataset. Note that as before, a significant problem remains that n here is small so the approximation to the t-distribution is likely somewhat inaccurate. It is also important to note that it is necessary for $m > 1$ for combine_syn() to output results since the sample variance is calculated which results in NAs if $m = 1$.

Note that for this project, we are only considering Y = Sex (and not Y = Age) due to time constraints.

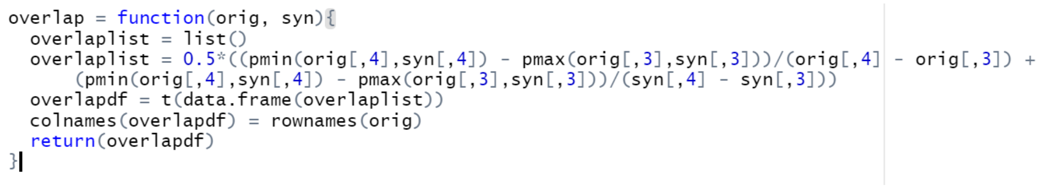
Data Utility Measures

Here we define the concept of data utility. Data utility is defined as the usefulness of the synthetic data generated. An ideal synthetic dataset should of course, have low disclosure risk but high data utility. A few measures of data utility include the *Confidence Interval Overlap Measure* and the *Standardized Difference*.

Confidence Interval Overlap Measure

The 95% confidence interval overlap measure is defined as the percentage overlap of original data confidence intervals and synthetic data confidence intervals. Note that Overlap Measure equals 1 when there is complete overlap (ie. the confidence interval endpoints are identical), while if Overlap Measure is less or equal to 0, then there is no overlap. Clearly, we would want the Overlap Measure to be as high as possible in order to increase data utility. Confidence Interval Overlap Measure is defined by:

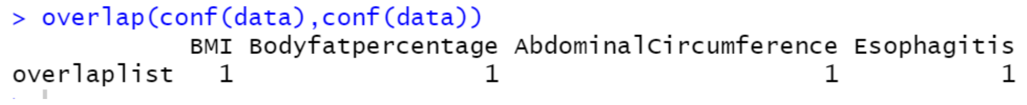
$$0.5[\frac{\min(U_{orig}, U_{syn}) \text{ - } \max(L_{orig}, L_{syn})}{U_{orig} - L_{orig}} + \frac{\min(U_{orig}, U_{syn}) \text{ - } \max(L_{orig}, L_{syn})}{U_{syn} - L_{syn}}]$$



```
overlap = function(orig, syn){
  overlaplist = list()
  overlaplist = 0.5*((pmin(orig[,4],syn[,4]) - pmax(orig[,3],syn[,3]))/(orig[,4] - orig[,3]) +
      (pmin(orig[,4],syn[,4]) - pmax(orig[,3],syn[,3]))/(syn[,4] - syn[,3]))
  overlapdf = t(data.frame(overlaplist))
  colnames(overlapdf) = rownames(orig)
  return(overlapdf)
}
```

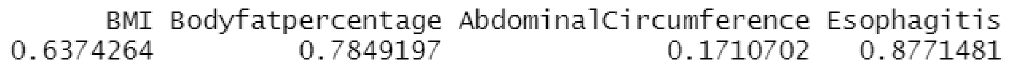
A function overlap(orig,syn) was created to calculate the overlap measure from the original data confidence intervals and synthetic data confidence intervals. Note that overlap(orig,syn) takes a summary of coefficients, standard errors, and confidence interval bounds of original data (orig) and the synthetic data (syn) as arguments.

By definition, if overlap() is defined on identical confidence intervals, then it would return 1. As expected, overlap(conf(data),conf(data)) returns 1 for all the variables.



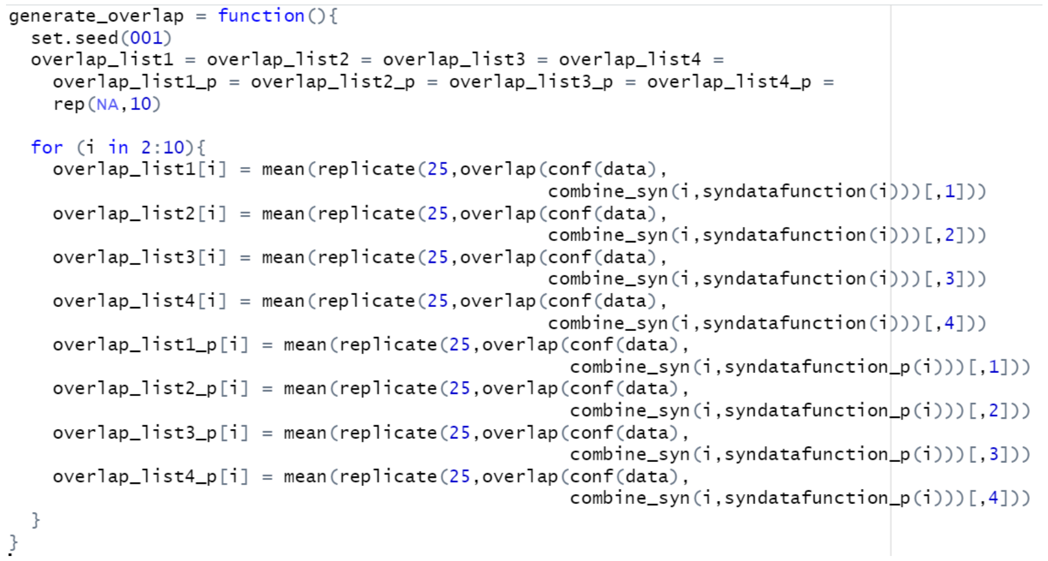
```
> overlap(conf(data),conf(data))
            BMI Bodyfatpercentage AbdominalCircumference Esophagitis
overlaplist   1                 1                      1           1
```

On the other hand, running overlap(conf(data), combine_syn(2,syndatafunction(2))) yields:



```
      BMI Bodyfatpercentage AbdominalCircumference Esophagitis
0.6374264         0.7849197              0.1710702   0.8771481
```

As before, the small sample size $n = 106$ poses a problem, so R=25 replications are done through a function generate_overlap().



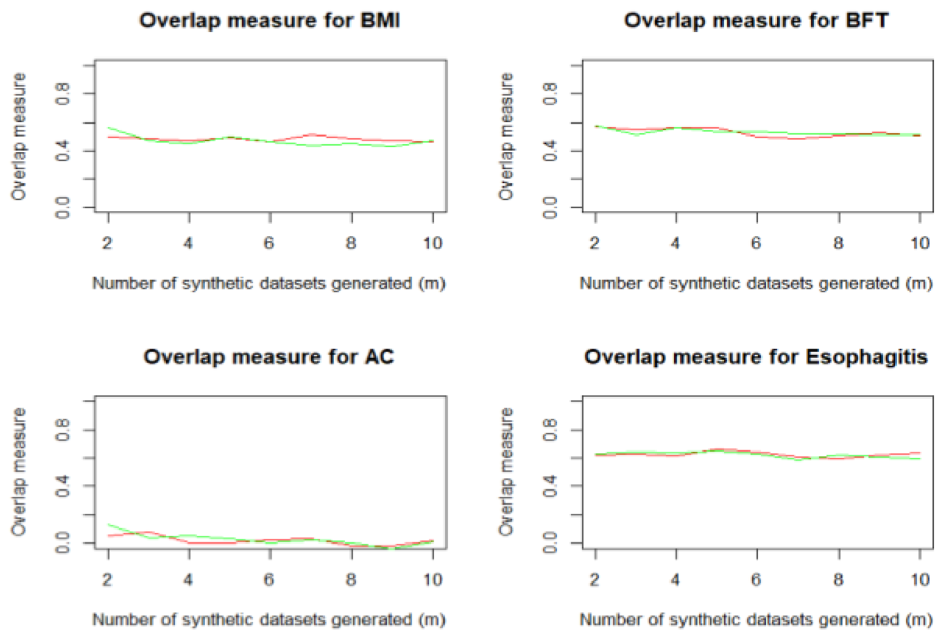
```
generate_overlap = function(){
  set.seed(001)
  overlap_list1 = overlap_list2 = overlap_list3 = overlap_list4 =
    overlap_list1_p = overlap_list2_p = overlap_list3_p = overlap_list4_p =
    rep(NA,10)

  for (i in 2:10){
    overlap_list1[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction(i)))[,1]))
    overlap_list2[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction(i)))[,2]))
    overlap_list3[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction(i)))[,3]))
    overlap_list4[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction(i)))[,4]))
    overlap_list1_p[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction_p(i)))[,1]))
    overlap_list2_p[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction_p(i)))[,2]))
    overlap_list3_p[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction_p(i)))[,3]))
    overlap_list4_p[i] = mean(replicate(25,overlap(conf(data),
                                          combine_syn(i,syndatafunction_p(i)))[,4]))
  }
}
```

Then, the overlap measure is plotted against Number of Synthetic Datasets generated (m) for each of BMI, BFT, AC, and Esophagitis by running plot() and lines() commands in R:

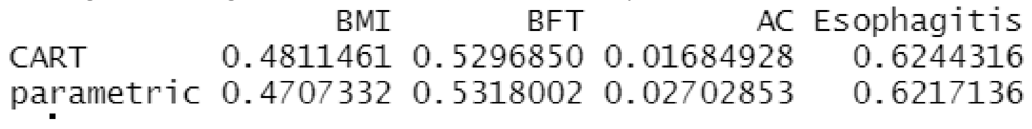
```r
par(mfrow = c(2,2))
plot(2:10, overlap_list1[2:10], type = 'l', col='red',
     main = 'Overlap measure for BMI',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Overlap measure',
     ylim = c(0,1))
lines(2:10, overlap_list1_p[2:10], col='green')
plot(2:10, overlap_list2[2:10], type = 'l', col='red',
     main = 'Overlap measure for BFT',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Overlap measure',
     ylim = c(0,1))
lines(2:10, overlap_list2_p[2:10], col='green')
plot(2:10, overlap_list3[2:10], type = 'l', col='red',
     main = 'Overlap measure for AC',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Overlap measure',
     ylim = c(0,1))
lines(2:10, overlap_list3_p[2:10], col='green')
plot(2:10, overlap_list4[2:10], type = 'l', col='red',
     main = 'Overlap measure for Esophagitis',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Overlap measure',
     ylim = c(0,1))
lines(2:10, overlap_list4_p[2:10], col='green')
```

This yields the following plots:



Again, the red colored line represents the CART method while the green colored line represents the parametric method.

As can be seen from the plots, both methods produce very similar results. Taking an average over $m = 1, 2, \ldots, 10$, this yields the table below:



```
                 BMI       BFT         AC Esophagitis
CART       0.4811461 0.5296850 0.01684928   0.6244316
parametric 0.4707332 0.5318002 0.02702853   0.6217136
```

Here it can be seen that the parametric and CART methods yield almost identical overlap measures across all of the variables. Looking at the data, the coefficients BMI, BFT, and Esophagitis have some confidence interval overlap; however, at roughly 0.47 - 0.63, it is less than ideal. However, the major problem is the variable Abdominal Circumference, which has a value of approximately 0.02 which is quite problematic since it implies that there is nearly zero overlap in Confidence Intervals. This is somewhat surprising and shows that there may be problems with data utility for both methods of generating synthetic data.

From the plots, it appears that as m (the number of synthetic datasets generated) increases, the overlap measure stays approximately constant for all the variables. This is a bit surprising since one would expect overlap measure to increase as m increases since there is additional information gained from the extra synthetic datasets. However, this may just be because of the fact that only $m = 1, 2, \ldots, 10$ is considered; this trend may be apparent if we consider $m = 50$ or $m = 100$.

Standardized Difference

The standardized difference is another measure of data utility. It is defined as the absolute difference of estimates of the original data and synthetic data coefficient divided by the standard error of the estimate of the original data coefficient. The mathematical equation is:

$$\frac{|\hat{\beta}_{orig} - \hat{\beta}_{syn}|}{SE(\hat{\beta}_{orig})}$$

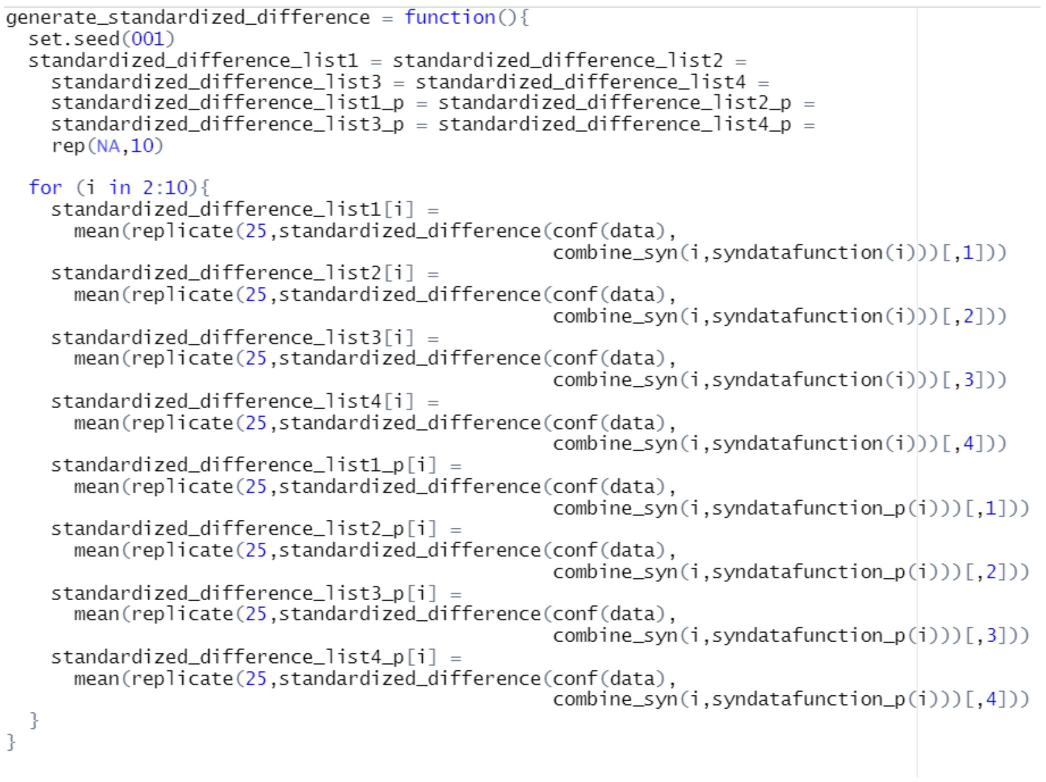
Clearly, standardized difference is equal to 0 when the estimated original and synthetic coefficients are equivalent. On the other hand, standardized difference is large when the absolute difference between estimated original and synthetic coefficients is large with respect to standard error (of estimated original coefficient). The function standardized_difference() is created to generated standardized differences of the coefficients. Note that overlap(orig,syn) takes a summary of coefficients, standard errors, and confidence interval bounds of original data (orig) and the synthetic data (syn) as arguments.

```
standardized_difference = function(orig, syn){
  sd_list = (abs(orig[,1] - syn[,1]))/orig[,2]
  sd_df = t(data.frame(sd_list))
  colnames(sd_df) = rownames(orig)
  return(sd_df)
}
```

In order to verify the function works correctly, standardized_difference()is tried on the coefficients of the original data fit against itself (which should equal 0).

```
BMI Bodyfatpercentage AbdominalCircumference Esophagitis
  0                 0                      0           0
```

Since the small sample size $n = 106$ poses a problem, so R=25 replications are done through a function generate_standardized_difference().



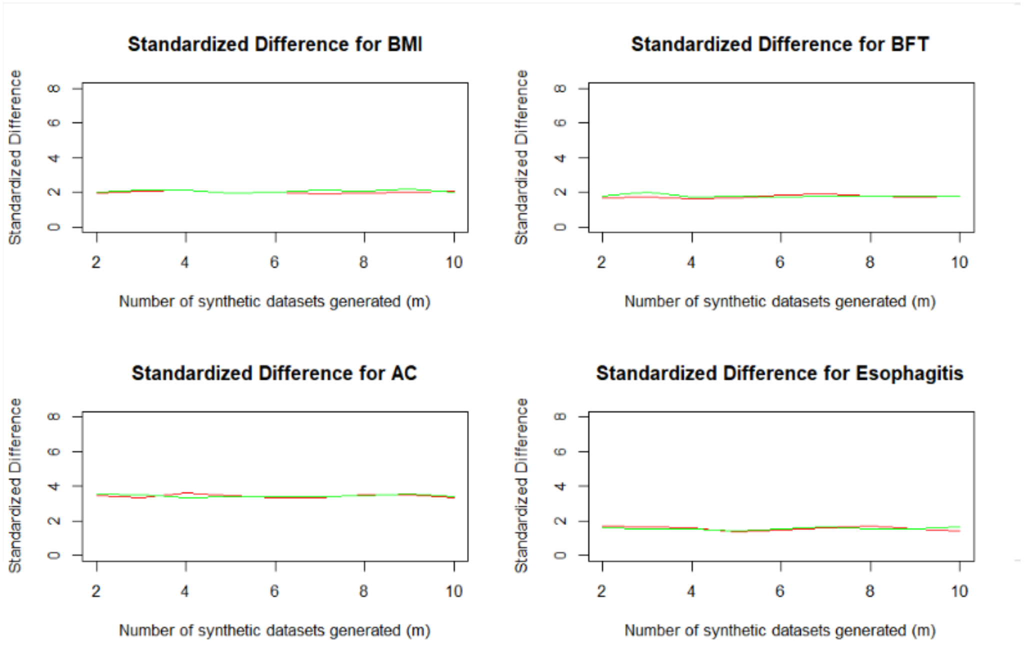
```
generate_standardized_difference = function(){
  set.seed(001)
  standardized_difference_list1 = standardized_difference_list2 =
    standardized_difference_list3 = standardized_difference_list4 =
    standardized_difference_list1_p = standardized_difference_list2_p =
    standardized_difference_list3_p = standardized_difference_list4_p =
    rep(NA,10)

  for (i in 2:10){
    standardized_difference_list1[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction(i)))[,1]))
    standardized_difference_list2[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction(i)))[,2]))
    standardized_difference_list3[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction(i)))[,3]))
    standardized_difference_list4[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction(i)))[,4]))
    standardized_difference_list1_p[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction_p(i)))[,1]))
    standardized_difference_list2_p[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction_p(i)))[,2]))
    standardized_difference_list3_p[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction_p(i)))[,3]))
    standardized_difference_list4_p[i] =
      mean(replicate(25,standardized_difference(conf(data),
                                                combine_syn(i,syndatafunction_p(i)))[,4]))
  }
}
```

Then, the standardized difference is plotted against Number of Synthetic Datasets generated (m) for each of BMI, BFT, AC, and Esophagitis by running plot() and lines() commands in R:

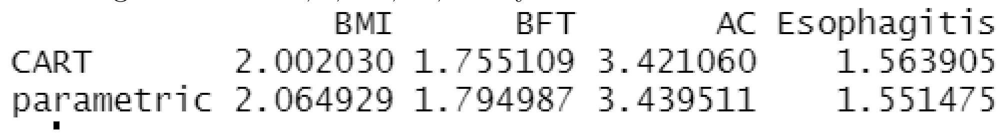
```
par(mfrow = c(2,2))
plot(2:10, standardized_difference_list1[2:10], type = 'l', col='red',
     main = 'Standardized Difference for BMI',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Standardized Difference',
     ylim = c(0,8))
lines(2:10, standardized_difference_list1_p[2:10], col='green')
plot(2:10, standardized_difference_list2[2:10], type = 'l', col='red',
     main = 'Standardized Difference for BFT',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Standardized Difference',
     ylim = c(0,8))
lines(2:10, standardized_difference_list2_p[2:10], col='green')
plot(2:10, standardized_difference_list3[2:10], type = 'l', col='red',
     main = 'Standardized Difference for AC',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Standardized Difference',
     ylim = c(0,8))
lines(2:10, standardized_difference_list3_p[2:10], col='green')
plot(2:10, standardized_difference_list4[2:10], type = 'l', col='red',
     main = 'Standardized Difference for Esophagitis',
     xlab = 'Number of synthetic datasets generated (m)',
     ylab = 'Standardized Difference',
     ylim = c(0,8))
lines(2:10, standardized_difference_list4_p[2:10], col='green')
```

This yields the following plots:



Again, the red colored line represents the CART method while the green colored line represents the parametric method.

It appears as well that both methods produce very similar results. Taking an average over $m = 1, 2, \ldots, 10$, this yields the table below:



```
                BMI      BFT       AC Esophagitis
CART       2.002030 1.755109 3.421060    1.563905
parametric 2.064929 1.794987 3.439511    1.551475
```

Here it can be seen that the parametric and CART methods yield almost identical overlap measures across all of the variables. Looking at the data, the standardized difference appears to range from 1.5-2.1 for BMI, BFT, and Esophagitis which is not that close to 0 and therefore not ideal. However, standardized difference is at around 3.4 for Abdominal Circumference which proves to be quite problematic. Therefore, we conclude that there may be problems with data utility for both methods of generating synthetic data.

From the plots, it appears that as m (the number of synthetic datasets generated) increases, the standardized difference stays approximately constant for all the variables. This is a bit surprising since one would expect standardized difference to decrease as m increases since there is additional information gained from the extra synthetic datasets. However, this may just be because of the fact that only $m = 1, 2, \ldots, 10$ is considered; this trend may be apparent if we consider $m = 50$ or $m = 100$.

Conclusion

Comparison of CART and parametric methods

In this experiment, the parametric and CART methods performed nearly identically well. In particular, there were no substantial differences in disclosure risks, overlap measure, and standardized difference for any number of synthetic datasets generated.

Disclosure Risk and Data Utility

In this section, the quality of the synthetic data generated with respect to this dataset will be discussed. The disclosure risk was under 1% for either method and for any average of m tested so the synthetic data generated minimizes privacy risk very well. On the other hand, the overlap measures range from around 0.47-0.63 for all of the predictor variables except for AC, which yielded overlap measure of around 0.02. None of the overlap measures are very close to 1, especially not that of AC. Furthermore, standardized differences range from around 1.5-2.1 for all the predictor variables except for AC, which yielded a standardized difference of 3.4. None of the standardized differences are very close to 0, especially not that of AC. This shows that the CART and parametric methods may have problems with data utility and that a better model is likely needed.

Trends as m increases

As m increases up until around $m = 10$, the disclosure is seen to increase for both methods which is expected. However, after as m increases further, there is no clear trend. Here, it is important to note that the sample size n is small and the number of replications is likewise small. Perhaps if those values were much larger, then it may be the case that disclosure risk increases as well as m increases.

As m increases, the Confidence Interval Overlap Measure and the Standardized Difference for this dataset appears to be constant. This is quite surprising; although, it is important to note that the sample size n is small and the number of replications is likewise small. Again, if those values were much larger, then it may be the case that when m increases, then Overlap Measure increases and Standardized Difference decreases.

Limitations and Improvements

A few limitations include that the number of methods that were explored was small. Ideally, more (maybe 6 or 7) would have been explored and compared but due to time constraints, only 2 were chosen here. In retrospect, a larger dataset should also have been chosen since $n = 106$ is much too small an amount of observations, especially considering that there were 6 variables in total. However, we have compensated for that somewhat with taking means of a large number of replications. Even so, it would have been beneficial if R=1000 or perhaps R=10000+ replications were performed. Although it would likely take weeks to run on this inefficient code, it is likely possible with optimized code.

Another limitation is that only $m = 1, 2, \ldots, 10$ were tested for the data utility measures. A possible improvement would be to see what would happen if $m = 20, 30, 50, 100$ for instance were tested as well. That way, long term trends would be apparent and the question of whether or not Confidence Interval Overlap Measure increases and standardized difference decreases when m increases will be solved for this dataset.

References

[1] Grund, S., Lüdtke, O., & Robitzsch, A. (2021, July 23). *Using synthetic data to improve the reproducibility of statistical results in psychological research.* https://doi.org/10.31234/osf.io/d7zwj

[2] Jiang, B., Raftery, A. E., Steele, R. J., & Wang, N. (2021, March 12). *Balancing Inferential Integrity and Disclosure Risk Via Model Targeted Masking and Multiple Imputation* Taylor and Francis Online. Retrieved March 4, 2022, from https://doi.org/10.1080/01621459.2021.1909597

[3] Reiter, Jerome P.. "Using CART to generate partially synthetic public use microdata." *Journal of Official Statistics* 21 (2005): 441-462.

[4] Shaked, D. S. (2021, February 18). Synthetic test data vs. data masking. Datomize. Retrieved March 4, 2022, from https://www.datomize.com/synthetic-test-data-vs-data-masking-what-are-the-main-differences/

[5] Zevallos Ventura, Alba Sofía. *Association between the Body Mass Index, Waist Circumference, and Body Fat Percentage with Erosive Esophagitis in Adults with Obesity after Sleeve Gastrectomy.* Jan. 2022. *dataverse.harvard.edu*, https://doi.org/10.7910/DVN/ZBVTY4.