Generating Synthetic Dataset

Technical Document

Table of Contents

R Packages

Library	Use
synthpop	generating synthetic dataset
dplyr	data manipulation
psych	descriptive analysis
tidyverse	collection of R packages which helps data to run faster and human readable
rapportools	column empty space inspection
caret	machine learning workflow
Vcd	categorical data visualization
GGally	data visualization
sampling	draw random sample
car	Applied in Regression
forcats	reordering categorical data

Part A

Check whether the synthetic dataset is still normal distribution compared to the original dataset from generating the data.

1. Generate data

In this part we are interesting to know if the synthpop package generate similar probability distribution by different categories. To check this matter, we generated a dataset with two variables, one categorical with three categories and one numerical variable. Numerical variable has different distribution for each category of the categorical variable.

Category A has a normal distribution with mean and standard deviation of 120 and 5, for 3000 observations. Category B with 2500 observations has normal distribution with mean 500 and standard deviation 50. Category C also has normal distribution with mean 5000 and standard deviation 300, for 4500 observations, Table 1.

```
set.seed(123)
## generate normal distribution for A
x <- rnorm(3000,mean = 120, sd = 5)
y <- 'A'
df <- data.frame(x,y)

## generate normal distribution for B
x1 = rnorm(2500,mean = 500, sd = 50)
y1 <- 'B'
df1 <- data.frame(x1,y1)

## generate normal distribution for C
x2 = rnorm(4500,mean = 5000, sd = 300)
y2 <- 'C'
df2 <- data.frame(x2,y2)
```

Table 1: Distribution of categories

Category	Number of obs.	Distribution	Mean	STD
A	3000	Normal	120	5
B	2500	Normal	500	50
C	4500	Normal	5000	300

Then we combine all three categories in a dataset and run the synthetic package to generate the data.

```
## combine A, B, C together into a data frame
names(df1) <- c("x", "y")
names(df2) <- c("x", "y")
new<-rbind(df,df1,df2)
```

2. Synthetic dataset

We generated synthesized data with 10,000 observations and 2 columns by applying Synthpop package in R. details of all categories for original and synthesized data given in Table 2. We also compared the histogram and normal distribution plot of variable X, for each category in original and synthesized dataset, Figures 1-8.

Table 2: Statistical details of categories

Data	Observations	Mean	SD
A – Original data	3,000	120	5
A – Synthetic data	3,004	120.13	5
B – Original data	2,500	500	50
B – Synthetic data	2,588	500.15	50.26
C – Original data	4,500	5,000	300
C – Synthetic data	4,408	4991,07	298.15

```
myseed=1020
##Apply Synthpop package
library(synthpop)
##Do Synthetic data
synth.obj <- syn(new, method = "cart",seed = myseed)
s_new <- data.frame(synth.obj$syn)
```

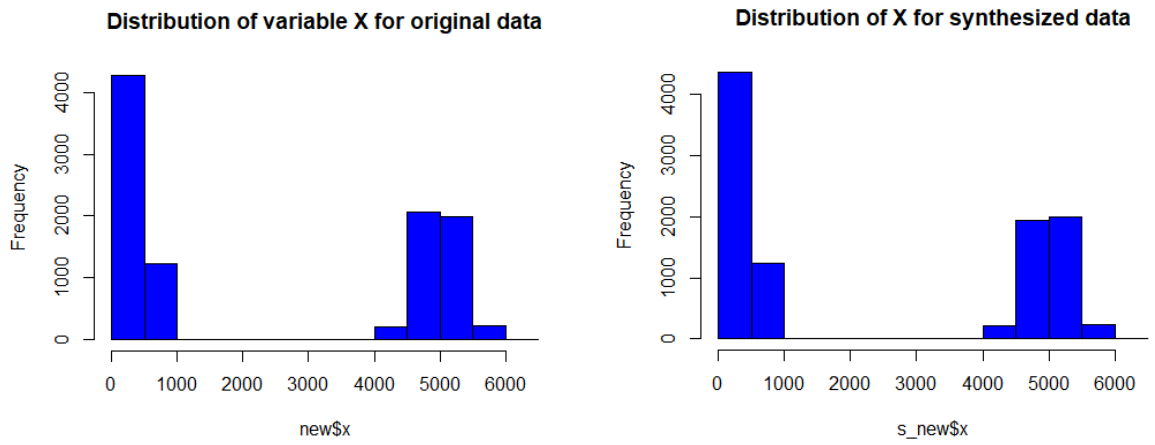


Figure 1: Distribution of variable X for whole datasets

```
## Distribution of X for original data
hist(new$x, main = "Distribution of variable X for original data", col = "Blue"
## Distribution of X for synthesized data
hist(s_new$x, main = "Distribution of X for synthesized data", col="blue")
```

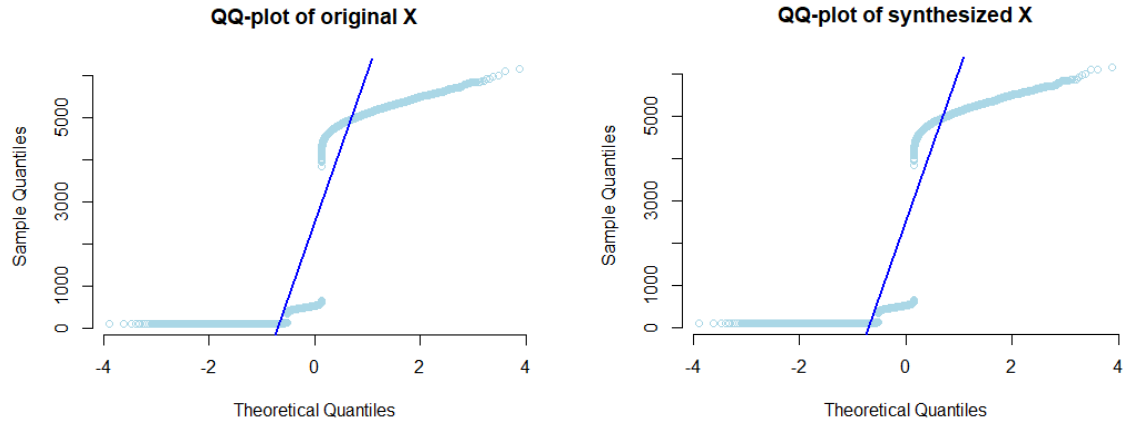


Figure 2: Normal plot of variable X for whole datasets

```
##QQ-plot of original X
qqnorm(new$x, pch = 1, frame = FALSE, main = "QQ-plot of original X", col = "lightblue")
qqline(new$x, col = "blue", lwd = 2)
##QQ-plot of synthesized X
qqnorm(s_new$x, pch = 1, frame = FALSE, main = "QQ-plot of synthesized X", col = "lightblue")
qqline(s_new$x, col = "blue", lwd = 2)
```

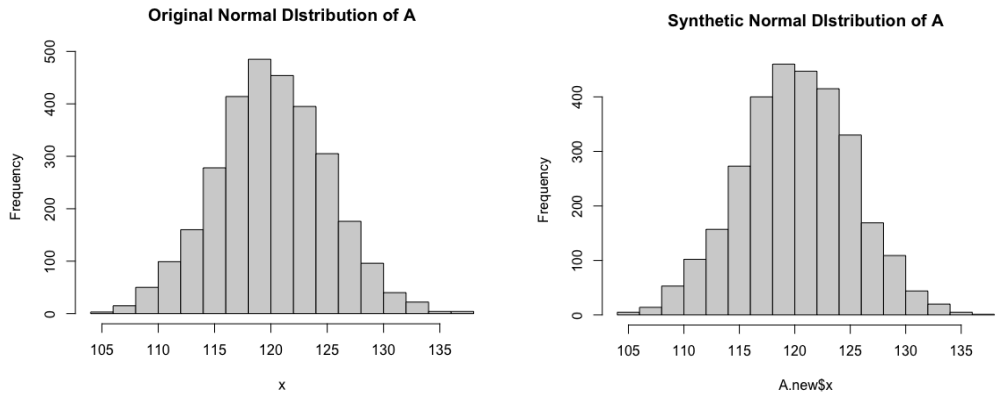


Figure 3: Distribution of variable X for category A

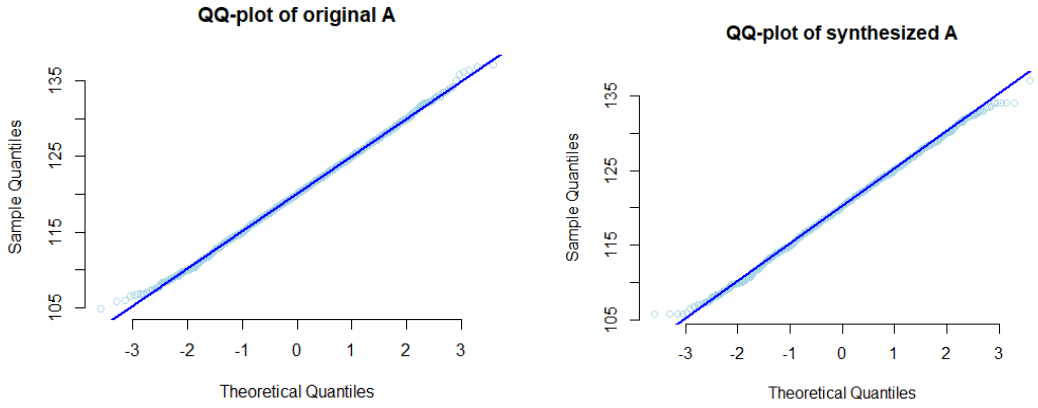


Figure 4: Normal plot of variable X for category A

```
library(dplyr)
library(psych)
## Histograms of A
A.new <- s_new%>% filter(s_new$y == 'A')
hist(A.new$x, main = "Synthetic Normal DIstribution of A")
hist(x, main = "Original Normal DIstribution of A")
dim(A.new)
describe(A.new)
## QQ-plots of A
qqnorm(df$x, pch = 1, frame = FALSE, main = "QQ-plot of original A", col = "lightblue")
qqline(df$x, col = "blue", lwd = 2)
qqnorm(A.new$x, pch = 1, frame = FALSE, main = "QQ-plot of synthesized A", col = "lightblue")
qqline(A.new$x, col = "blue", lwd = 2)
```

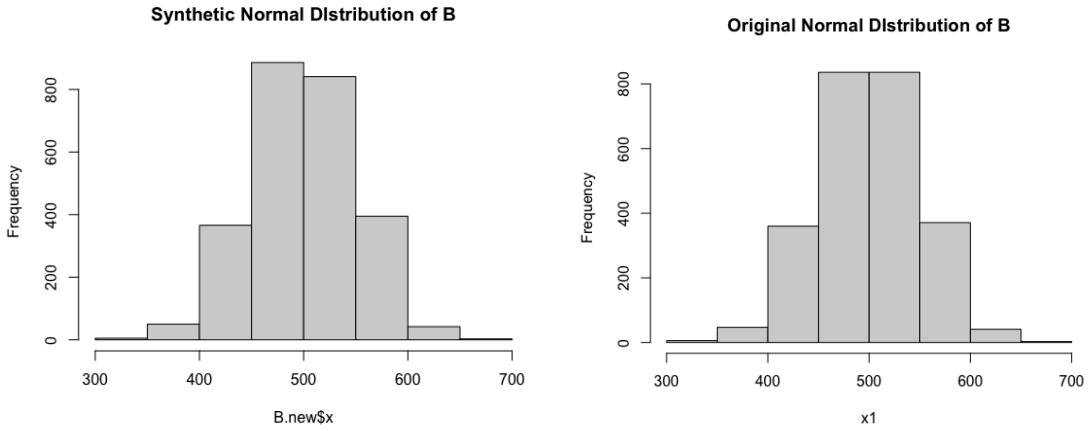


Figure 5: Distribution of variable X for category B

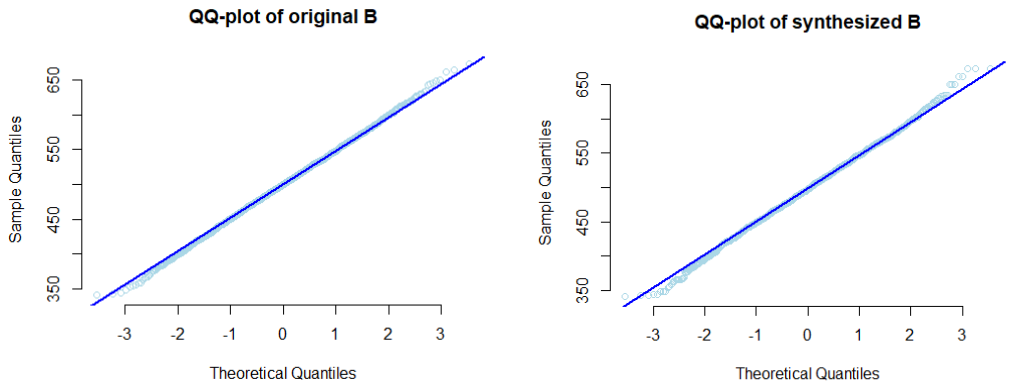


Figure 6: Normal plot of variable X for category B

```
## Histograms of B
B.new <- s_new%>% filter(s_new$y == 'B')
hist(x1, main = "Original Normal Distribution of B")
hist(B.new$x,main = "Synthetic Normal DIstribution of B")
dim(B.new)
describe(B.new)
## QQ-plots of B
qqnorm(df1$x, pch = 1, frame = FALSE, main = "QQ-plot of original B", col = "lightblue")
qqline(df1$x, col = "blue", lwd = 2)
qqnorm(B.new$x, pch = 1, frame = FALSE, main = "QQ-plot of synthesized B", col = "lightblue"
qqline(B.new$x, col = "blue", lwd = 2)
```

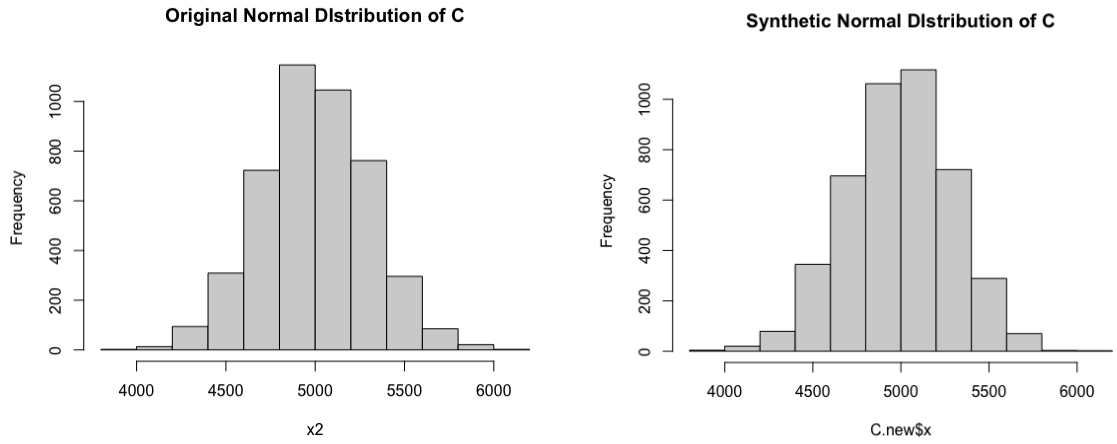


Figure 7: Distribution of variable X for category C

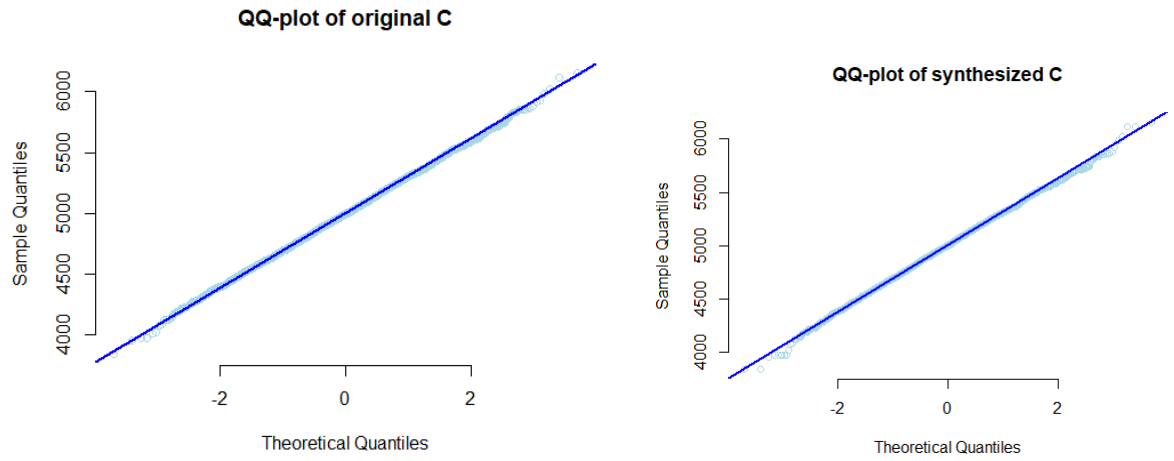


Figure 8: Normal plot of variable X for category C

```
## Histograms of C
C.new <- s_new%>% filter(s_new$y == 'C')
hist(C.new$x, main = "Synthetic Normal DIstribution of C")
hist(x2, main = "Original Normal DIstribution of C")
dim(C.new)
describe(C.new)
##QQ-plots of C
qqnorm(df2$x, pch = 1, frame = FALSE, main = "QQ-plot of original C", col = "lightblue")
qqline(df2$x, col = "blue", lwd = 2)
qqnorm(C.new$x, pch = 1, frame = FALSE, main = "QQ-plot of synthesized C", col = "lightblue")
qqline(C.new$x, col = "blue", lwd = 2)
```

From Table 2 and Figures 1-8, we can see there is not significant difference in terms of observations, Mean, SD, and distribution between the original and the synthesized dataset. Therefore, we can conclude that the synthesized dataset still follows normal distribution like the original dataset.

3. Kolmogorov-Smirnov (KS) Test

Two-sample KS-Test can be used to compare the distribution of the observations in both datasets for each category statistically.

Ho: The two dataset values are from the same continuous distribution.
Ha: The two dataset values are not from the same continuous distribution.

- For category A of the original data and the synthetic data, the test result shows below:

```
        Two-sample Kolmogorov-Smirnov test

data:  df$x and A.new$x
D = 0.018039, p-value = 0.7131
alternative hypothesis: two-sided
```

The output shows p_value > 0.05, so we cannot reject the null hypothesis and it means that the two datasets follow the same continuous distribution.

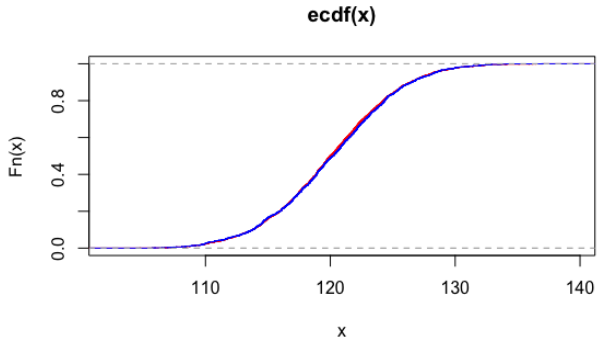


Figure 9: KS-plot for category A

```r
#ks test for A
ks.test(df$x,A.new$x)
plot.ecdf(df$x, verticals=TRUE, do.points=FALSE, col="red")
lines(ecdf(A.new$x), verticals=TRUE, do.points=FALSE, col="blue")
```

- For category B of the original data and the synthetic data, the test result shows below:

```
        Two-sample Kolmogorov-Smirnov test

data:  df1$x and B.new$x
D = 0.027452, p-value = 0.2933
alternative hypothesis: two-sided
```

The result illustrates that p_value is more than 0.05, so there is enough evidence that distribution of variable X for category B in the original and synthesized data are similar. The KS-plot showed in Figure 10.

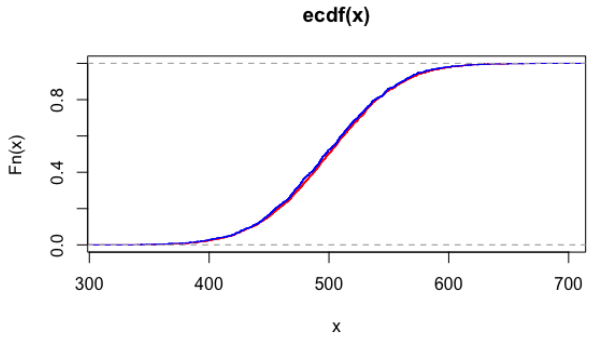


Figure 10: KS-plot for category B

```
#ks test for B
ks.test(df1$x,B.new$x)
plot.ecdf(df1$x, verticals=TRUE, do.points=FALSE, col="red")
lines(ecdf(B.new$x), verticals=TRUE, do.points=FALSE, col="blue")
```

- For category C of the original data and the synthetic data, the test result shows below:

```
        Two-sample Kolmogorov-Smirnov test

data:  df2$x and C.new$x
D = 0.022668, p-value = 0.2026
alternative hypothesis: two-sided
```

Same as categories A and B, the KS-Test shows similarity of distribution for category C. The plot is showed in Figure 11.

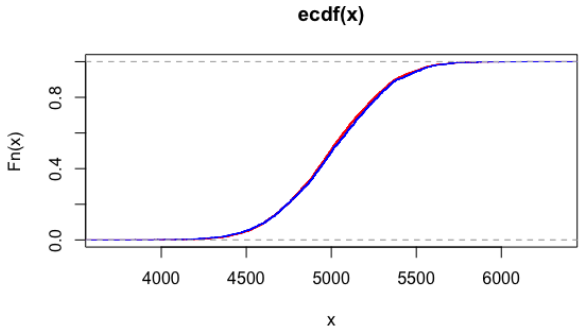


Figure 11: KS-plot for category C

```
#ks test for C
ks.test(df2$x,C.new$x)
plot.ecdf(df2$x, verticals=TRUE, do.points=FALSE, col="red")
lines(ecdf(C.new$x), verticals=TRUE, do.points=FALSE, col="blue")
```

4. Linear regression

We fitted a linear regression on variable X related to the categorical variable for the original dataset and the synthetic dataset. The results show below:

a) The original dataset

```
Call:
lm(formula = x ~ y, data = new)

Residuals:
     Min       1Q   Median       3Q      Max
-1151.60   -41.29    -0.13    40.21  1156.32

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  120.064      3.729   32.20   <2e-16 ***
yB           379.290      5.531   68.58   <2e-16 ***
yC          4877.943      4.814 1013.26   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 204.2 on 9997 degrees of freedom
Multiple R-squared:  0.9925,    Adjusted R-squared:  0.9925
F-statistic: 6.592e+05 on 2 and 9997 DF,  p-value: < 2.2e-16
```

b) The synthetic dataset

```
Call:
lm(formula = x ~ y, data = s_new)

Residuals:
     Min       1Q   Median       3Q      Max
-1159.31   -41.10     0.01    41.99  1148.61

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  120.131      3.778   31.80   <2e-16 ***
yB           377.368      5.553   67.96   <2e-16 ***
yC          4885.588      4.899  997.36   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 207 on 9997 degrees of freedom
Multiple R-squared:  0.9923,    Adjusted R-squared:  0.9923
F-statistic: 6.404e+05 on 2 and 9997 DF,  p-value: < 2.2e-16
```

From the results above, we can see the Estimated values of parameters for two models are similar. The similarity of two models is shown by the residual plots, Figure 12.

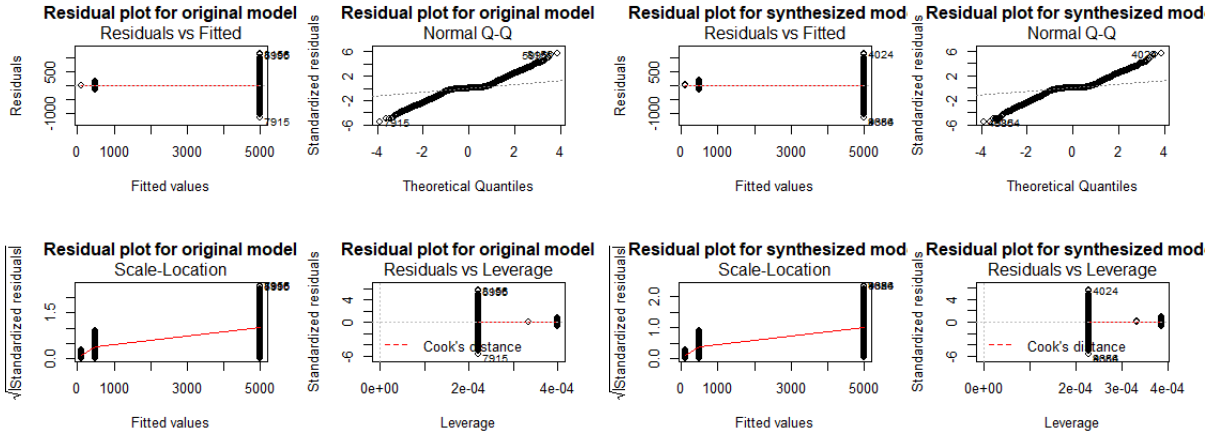


Figure 12: Residual plot of the original model and the synthesized model

```r
### Regression models
##The original data
model = lm(x~y, data = new)
summary(model)

#The synthetic data
model_syn = lm(x~y, data = s_new)
summary(model_syn)

#checking the assumptions
par(mfrow=c(2,2))
plot(model, main= "Residual plot for original model")
par(mfrow=c(1,1))

#checking the assumptions
par(mfrow=c(2,2))
plot(model_syn, main= "Residual plot for synthesized model")
par(mfrow=c(1,1))
```

Part B

Check whether the synthetic dataset is still normal distribution compared to the original dataset from Health Insurance dataset

1. Exploring the dataset

```
> str(HI)
'data.frame':   381109 obs. of  12 variables:
 $ id                  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender              : Factor w/ 2 levels "Female","Male": 2 2 2 2 1 1 2 1 1 1 ...
 $ Age                 : int  44 76 47 21 29 24 23 56 24 32 ...
 $ Driving_License     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Region_Code         : int  28 3 28 11 41 33 11 28 3 6 ...
 $ Previously_Insured  : int  0 0 0 1 1 0 0 0 1 1 ...
 $ Vehicle_Age         : Factor w/ 3 levels "< 1 Year","> 2 Years",..: 2 3 2 1 1 1 1 3 1 1
 ...
 $ Vehicle_Damage      : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 2 2 2 1 1 ...
 $ Annual_Premium      : int  40454 33536 38294 28619 27496 2630 23367 32031 27619 28771
 ...
 $ Policy_Sales_Channel: int  26 26 26 152 152 160 152 26 152 152 ...
 $ Vintage             : int  217 183 27 203 39 176 249 72 28 80 ...
 $ Response            : int  1 0 1 0 0 0 0 1 0 0 ...
>
```

```
##Library Import
library(synthpop)
library(tidyverse)
library(sampling)
library(rapportools)
library(dplyr)
library(caret)
library(vcd)
library(GGally)
library(forcats)
library(car)


##Import Dataset##
HI<-read.csv('health_ insurance.csv')
view(HI)
str(HI)

##Missing value and blank spaces checking##
colSums(is.na(HI))
colSums(is.empty(HI))

##Set up dataset into data frame##
HI<-as.data.frame(HI)
myseed=1337
```

Apply synthpop package to generate the synthetic data from the HI data.

```
##Apply synthpop
library(synthpop)

ptm <- proc.time()   # reading time of generating data
synth.obj <- syn(HI,drop.not.used = TRUE, minnumlevels=3, seed = myseed)
proc.time() - ptm

synth.obj$predictor.matrix
synth.obj$method
syn1<-synth.obj$syn
view(syn1)
```

We tried to check distribution of different categories for real data too. So, from health insurance data we considered variables Age and Vehicle-Age which includes 3 categories: "< 1 year", "1-2 year", and "> 2 year" to check the distribution and frequency of categories for original data and synthesize data.

Figure 13 and 14 shows histogram for the Age variable based on Vehicle-Age categories for original and synthesized data. We can see that both have similar distribution.

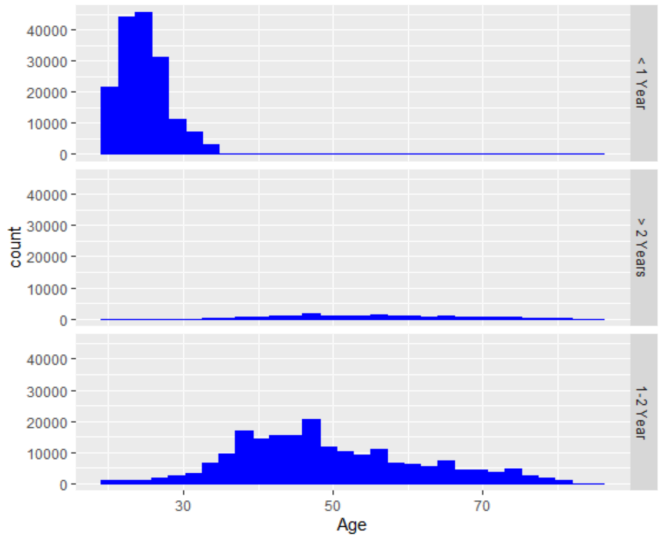


Figure 13: Histogram for Age based on Vehicle-Age categories of the original data

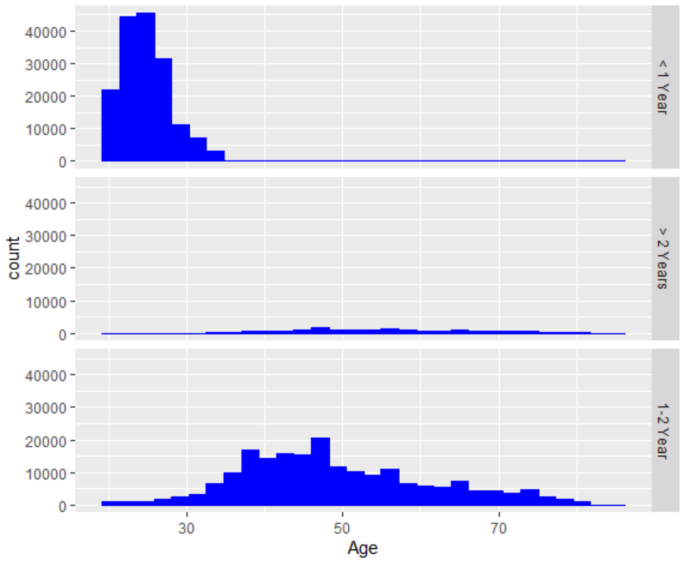


Figure 14: Histogram for Age based on Vehicle-Age categories of the synthetic data

```
## statistical summary of Age based on Vehicle_age categories
HI %>%
        group_by(Vehicle_Age) %>%
        summarise(across(Age, mean,na.rm=TRUE)) %>%
        summaries(count())

HI %>%
        group_by(Vehicle_Age) %>%
        summaries()

library(MASS)
head(birthwt)
# Use vehicle_age as the faceting variable
ggplot(HI, aes(x = Age)) +
        geom_histogram(fill = "Blue", colour = "Blue") +
        facet_grid(Vehicle_Age ~ . )

# plot for synthesized data
ggplot(syn1, aes(x = Age)) +
        geom_histogram(fill = "Blue", colour = "Blue") +
        facet_grid(Vehicle_Age ~ .)
```

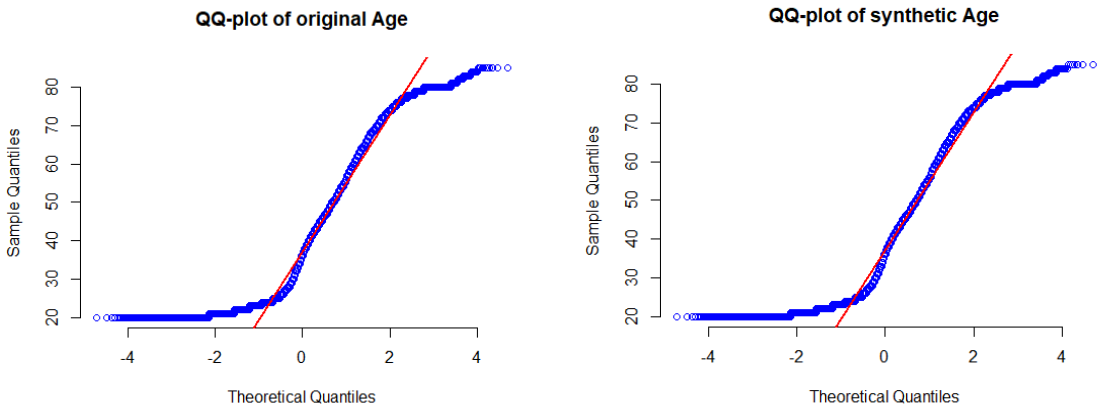
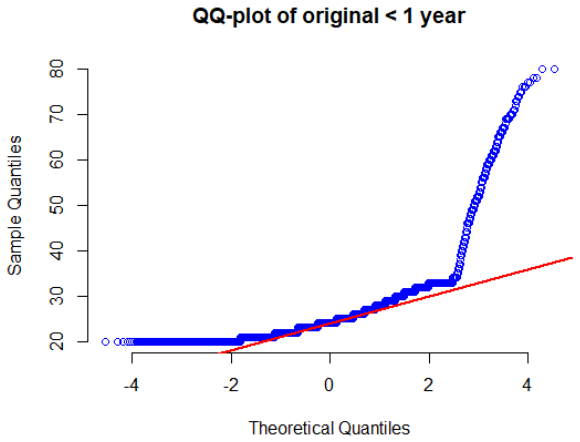


Figure 15: QQ-plot of the original Age the synthetic Age

```
##for normality comparison between synthetic data and original data
##QQ-plot for the original Age and the synthetic Age
qqnorm(HI$Age, pch = 1, frame = FALSE, main = "QQ-plot of original Age", col = "blue")
qqline(HI$Age, col = "red", lwd = 2)
qqnorm(syn1$Age, pch = 1, frame = FALSE, main = "QQ-plot of synthetic Age", col = "blue")
qqline(syn1$Age, col = "red", lwd = 2)
```



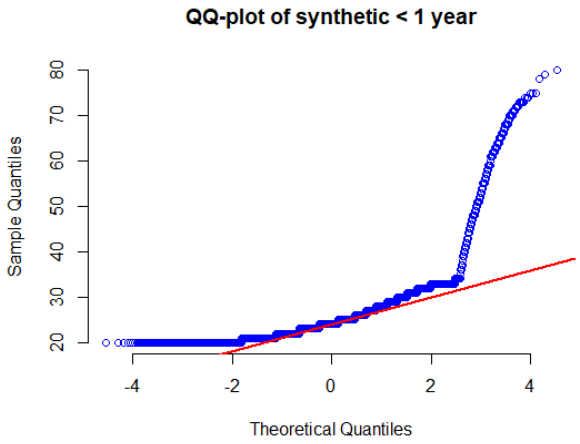


Figure 16: QQ-plot of the original the synthetic < 1 year

```
levels(year_1$Vehicle_Age)
# "< 1 Year"  "> 2 Years" "1-2 Year"
##QQ-plot for the original Age and the synthetic Age < 1 year
year_1 = HI %>% filter(Vehicle_Age =="< 1 Year")
year_1syn = syn1 %>% filter(Vehicle_Age =="< 1 Year")
str(year_1syn)
qqnorm(year_1$Age, pch = 1, frame = FALSE, main = "QQ-plot of original < 1 year", col = "blue")
qqline(year_1$Age, col = "red", lwd = 2)
qqnorm(year_1syn$Age, pch = 1, frame = FALSE, main = "QQ-plot of synthetic < 1 year", col = "blue")
qqline(year_1syn$Age, col = "red", lwd = 2)
```

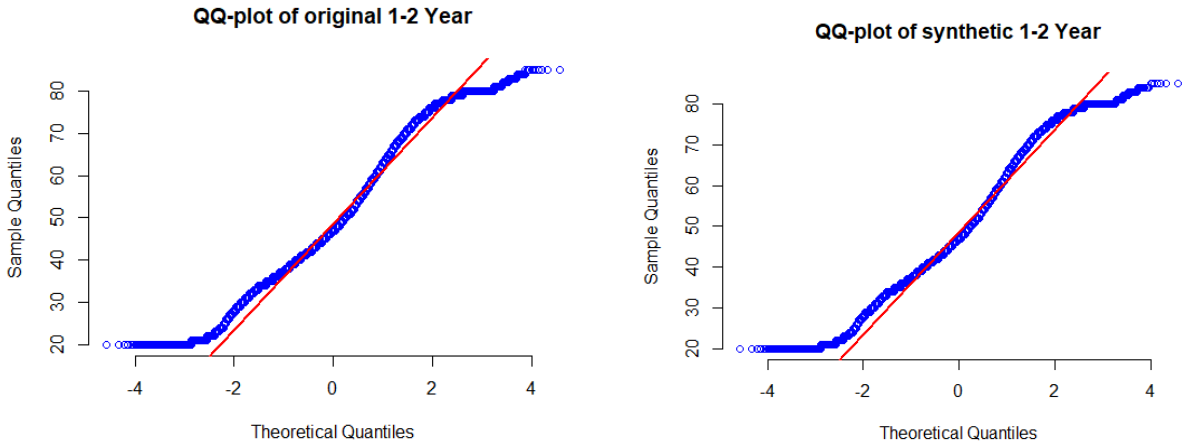


Figure 17: QQ-plot of the original the synthetic 1-2 year

```
##QQ-plot for the original Age and the synthetic Age < 2 year
year_2 = HI %>% filter(Vehicle_Age =="1-2 Year")
year_2syn = syn1 %>% filter(Vehicle_Age =="1-2 Year")
str(year_2syn)
qqnorm(year_2$Age, pch = 1, frame = FALSE, main = "QQ-plot of original 1-2 Year", col = "blue")
qqline(year_2$Age, col = "red", lwd = 2)
qqnorm(year_2syn$Age, pch = 1, frame = FALSE, main = "QQ-plot of synthetic 1-2 Year", col = "blue")
qqline(year_2syn$Age, col = "red", lwd = 2)
```

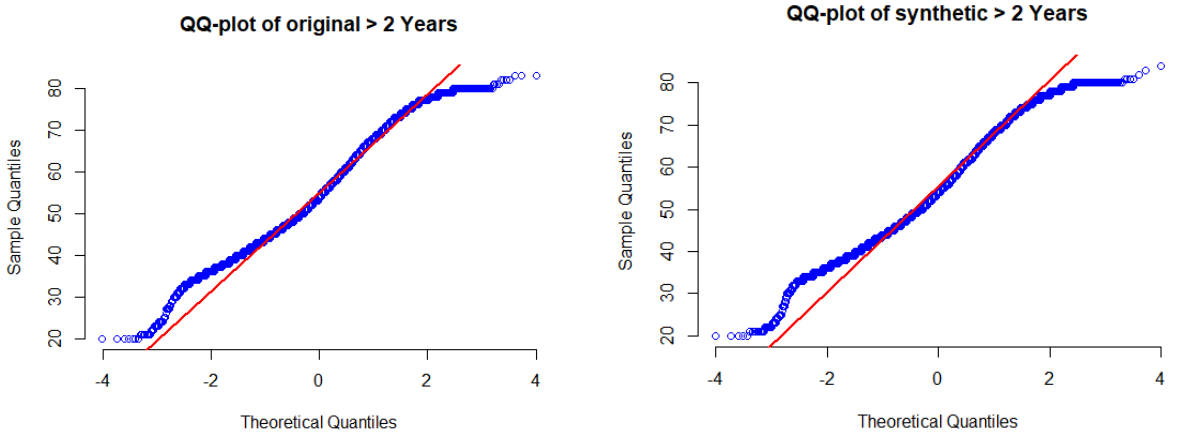


Figure 18: QQ-plot of the original the synthetic > 2 year

```
##QQ-plot for the original Age and the synthetic Age > 2 year
year_3 = HI %>% filter(Vehicle_Age =="> 2 Years")
year_3syn = syn1 %>% filter(Vehicle_Age =="> 2 Years")
str(year_3syn)
qqnorm(year_3$Age, pch = 1, frame = FALSE, main = "QQ-plot of original > 2 Years", col = "blue")
qqline(year_3$Age, col = "red", lwd = 2)
qqnorm(year_3syn$Age, pch = 1, frame = FALSE, main = "QQ-plot of synthetic > 2 Years", col = "blue")
qqline(year_3syn$Age, col = "red", lwd = 2)
```

2. Kolmogorov-Smirnov (KS) test

Two-sample KS test can be used to compare the distributions of the observations from the two datasets.

H_o: The two dataset values are from the same continuous distribution.

H_a: The two dataset values are not from the same continuous distribution.

```
> ks.test(HI$Age, syn1$Age)

        Two-sample Kolmogorov-Smirnov test

data:  HI$Age and syn1$Age
D = 0.00254, p-value = 0.171
alternative hypothesis: two-sided
```

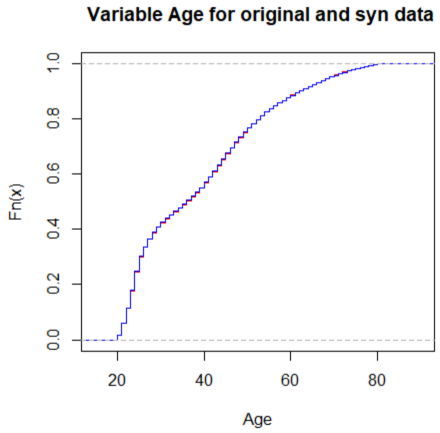


Figure 19: KS-plot of the original Age and the synthesized Age

```
############## KS test ##############
#ks test for Age
ks.test(HI$Age, syn1$Age)
plot.ecdf(HI$Age, verticals=TRUE, do.points=FALSE, col="red",
          xlab="Age", main=" Variable Age for original and syn data")
lines(ecdf(syn1$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

- Test for < 1 year of Vehicle-Age

```
> ks.test(year_1$Age,year_1syn$Age)

        Two-sample Kolmogorov-Smirnov test

data:  year_1$Age and year_1syn$Age
D = 0.0015861, p-value = 0.9855
alternative hypothesis: two-sided
```

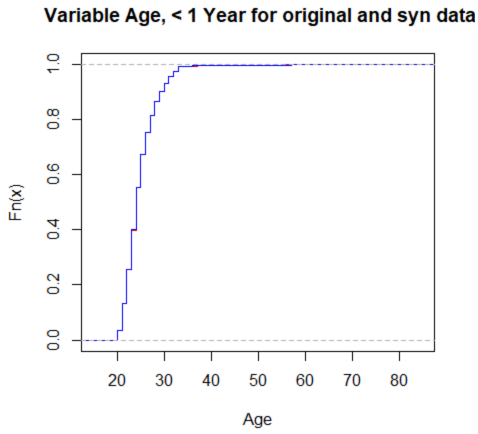


Figure 20: KS-plot of Vehicle-Age < 1 year of the original and the synthetic data

```
#ks test for < 1 year
ks.test(year_1$Age,year_1syn$Age)
plot.ecdf(year_1$Age, verticals=TRUE, do.points=FALSE, col="red",
          xlab="Age", main=" Variable Age, < 1 Year for original and syn data")
lines(ecdf(year_1syn$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

- Test for 1-2 years of Vehicle-Age

```
> ks.test(year_2$Age,year_2syn$Age)

        Two-sample Kolmogorov-Smirnov test

data:  year_2$Age and year_2syn$Age
D = 0.0015317, p-value = 0.973
alternative hypothesis: two-sided
```

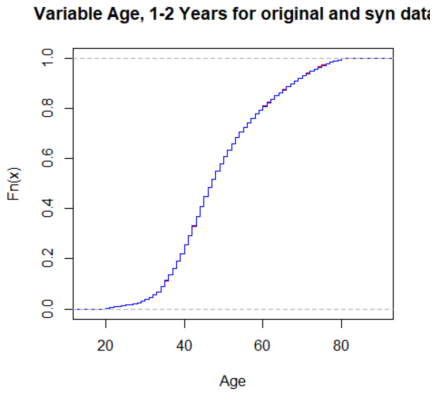


Figure 21: KS-plot of Vehicle-Age 1-2 year of the original and the synthetic data

```
#ks test for 1-2 years
ks.test(year_2$Age,year_2syn$Age)
plot.ecdf(year_2$Age, verticals=TRUE, do.points=FALSE, col="red",
          xlab="Age", main=" Variable Age, 1-2 Years for original and syn data")
lines(ecdf(year_2syn$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

- Test for > 2 years of Vehicle-Age

```
> ks.test(year_3$Age,year_3syn$Age)

        Two-sample Kolmogorov-Smirnov test

data:  year_3$Age and year_3syn$Age
D = 0.0050276, p-value = 0.9884
alternative hypothesis: two-sided
```

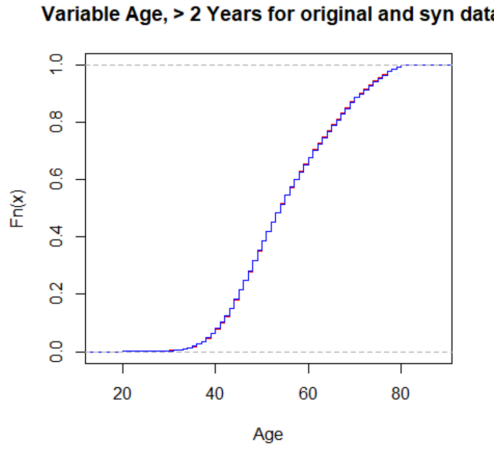


Figure 22: KS-plot of Vehicle-Age > 2 years of the original and the synthetic data

```
#ks test for >2 years
ks.test(year_3$Age,year_3syn$Age)
plot.ecdf(year_3$Age, verticals=TRUE, do.points=FALSE, col="red",
          xlab="Age", main=" Variable Age, > 2 Years for original and syn data")
lines(ecdf(year_3syn$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

For the Health Insurance dataset also, Kolmogorov-Smirnov (KS) test shows that for all data and for each category of Vehicle-Age variable the distribution of Age is similar for original and synthesized data.

3. Multinomial Logistic Regression

Then we run multinomial logistic regression for these variables by adding two more variables of Gender and Vintage and compared the output. The fitted model for both original and synthesized data are similar.

	Estimate		Std. Error		z-value		Pr(>|z|)	
Parameters	**Original**	**Synth**	**Original**	**Synth**	**Original**	**Synth**	**Original**	**Synth**
(Intercept):1	14.8700	14.8200	0.0588	0.0585	253.018	253.346	2.00E-16	2.00E-16
(Intercept):2	-4.6160	-4.6360	0.0409	0.0412	-112.869	-112.523	2.00E-16	2.00E-16
Age:1	-0.4602	-0.4602	0.0018	0.0018	-253.652	-253.807	2.00E-16	2.00E-16
Age:2	0.0384	0.0382	0.0006	0.0007	59.519	58.757	2.00E-16	2.00E-16
Vintage:1	0.0001	0.0000	0.0001	0.0001	0.751	0.387	0.452	0.699
Vintage:2	0.0001	0.0001	0.0001	0.0001	0.575	1.364	0.565	0.172
(Gender)Male:1	-0.3225	-0.2291	0.0164	0.0164	-19.641	-13.993	2.00E-16	2.00E-16
(Gender)Male:2	0.1297	0.1161	0.0173	0.0175	7.505	6.655	6.12E-14	2.84E-11

	Residual deviance	Log-likelihood	df
Original Data	214989.7	-107495	762210
Synthesized Data	213352.4	-106676	762210

Before synthetic

```
> #fit1<- vglm(Vehicle_Age~ Age+Vintage+factor(Gender),  family=multinomial, data=HI)
> summary(fit1)

Call:
vglm(formula = Vehicle_Age ~ Age + Vintage + factor(Gender),
    family = multinomial, data = HI)

Coefficients:
                      Estimate Std. Error  z value Pr(>|z|)
(Intercept):1         1.487e+01  5.875e-02  253.018  < 2e-16 ***
(Intercept):2        -4.616e+00  4.090e-02 -112.869  < 2e-16 ***
Age:1                -4.602e-01  1.814e-03 -253.652  < 2e-16 ***
Age:2                 3.841e-02  6.454e-04   59.519  < 2e-16 ***
Vintage:1             7.368e-05  9.806e-05    0.751    0.452
Vintage:2             5.692e-05  9.903e-05    0.575    0.565
factor(Gender)Male:1 -3.225e-01  1.642e-02  -19.641  < 2e-16 ***
factor(Gender)Male:2  1.297e-01  1.728e-02    7.505 6.12e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3])

Residual deviance: 214989.7 on 762210 degrees of freedom

Log-likelihood: -107494.8 on 762210 degrees of freedom

Number of Fisher scoring iterations: 8

Warning: Hauck-Donner effect detected in the following estimate(s):
'(Intercept):1', '(Intercept):2', 'Age:1'
```

After synthetic

```
> summary(fit1_syn)

Call:
vglm(formula = Vehicle_Age ~ Age + Vintage + factor(Gender),
    family = multinomial, data = syn1)

Coefficients:
                      Estimate Std. Error  z value Pr(>|z|)
(Intercept):1         1.482e+01  5.851e-02  253.346  < 2e-16 ***
(Intercept):2        -4.636e+00  4.120e-02 -112.523  < 2e-16 ***
Age:1                -4.602e-01  1.813e-03 -253.807  < 2e-16 ***
Age:2                 3.824e-02  6.508e-04   58.757  < 2e-16 ***
Vintage:1             3.782e-05  9.779e-05    0.387    0.699
Vintage:2             1.369e-04  1.004e-04    1.364    0.172
factor(Gender)Male:1 -2.291e-01  1.638e-02  -13.993  < 2e-16 ***
factor(Gender)Male:2  1.161e-01  1.745e-02    6.655 2.84e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Names of linear predictors: log(mu[,1]/mu[,3]), log(mu[,2]/mu[,3])

Residual deviance: 213352.4 on 762210 degrees of freedom

Log-likelihood: -106676.2 on 762210 degrees of freedom

Number of Fisher scoring iterations: 8

Warning: Hauck-Donner effect detected in the following estimate(s):
'(Intercept):1', '(Intercept):2', 'Age:1'
```

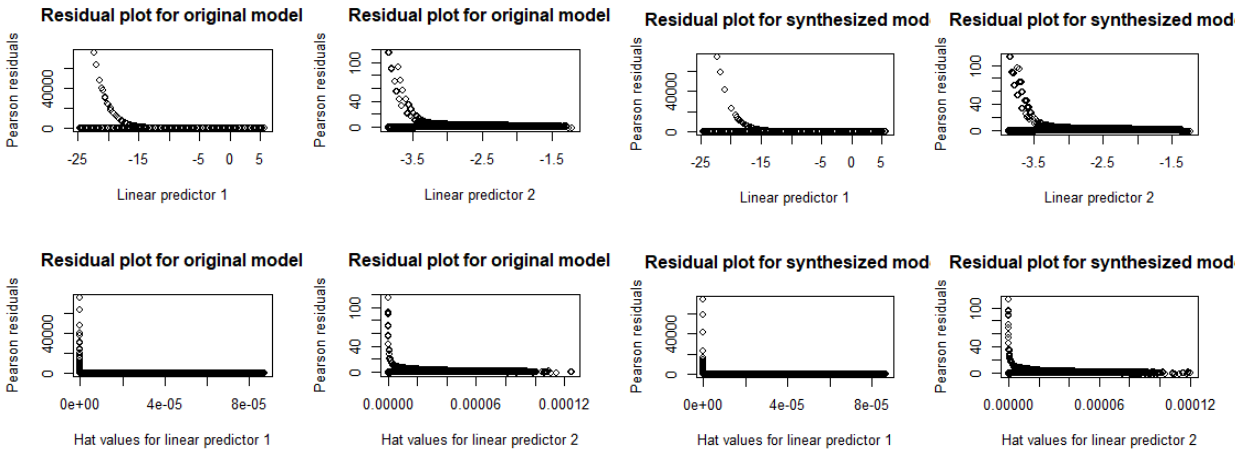


Figure 23: Residual plot of the original and the synthetic data

To confirm the similarity of frequency we have the pivot table for Vehicle-Age and Gender. The number of observations for each category in the original data and synthesized data are similar, tables below.

		Gender		
	Vehicle-Age	**Female**	**Male**	**Total**
Original Data	< 1 Year	91321	73465	164786
	> 2 Years	5707	10300	16007
	1-2 Year	77992	122324	200316
Total		**175020**	**206089**	**381109**
Synthesized Data	< 1 Year	91423	74166	165589
	> 2 Years	5631	9958	15589
	1-2 Year	78303	121628	199931
Total		**175357**	**205752**	**381109**

Figure 24: The pivot table for Vehicle-Age and Gender

```r
####### multinomial Logistic regression for Vehicle_age and age
library(nnet)
library(VGAM)

fit1<- vglm(Vehicle_Age~ Age+Vintage+factor(Gender),  family=multinomial, data=HI)
summary(fit1)

#checking the assumptions
par(mfrow=c(2,2))
plot(fit1, main= "Residual plot for original model")
par(mfrow=c(1,1))

# fit the model for synthetic data
fit1_syn<- vglm(Vehicle_Age~ Age+Vintage+factor(Gender),  family=multinomial, data=syn1)
summary(fit1_syn)
anova(fit1_syn)
AIC(fit1_syn)

#checking the assumptions
par(mfrow=c(2,2))
plot(fit1_syn, main= "Residual plot for synthesized model")
par(mfrow=c(1,1))
```

Part C

Headcheck Company

1. Objectives

- Check distribution of a continues variable for different categories of a categorical variable after synthesizing the dataset.
- Check the estimated value of the regression model fitted to the dataset before and after synthesizing dataset.

2. Headcheck dataset

The data contains 20 variables, and 29,355 observations as follows:

```
> str(data)
spec_tbl_df [29,355 x 20] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ Test.Type                    : chr [1:29355] "Baseline" "Baseline" "Baseline" "Baseline" ...
 $ Org.ID                       : num [1:29355] 36 23 4 32 24 7 16 56 27 27 ...
 $ Team.ID                      : num [1:29355] 850 528 109 789 552 ...
 $ Sport                        : num [1:29355] 3 4 3 3 3 10 4 11 3 3 ...
 $ Athlete.ID                   : num [1:29355] 18146 12079 2576 17288 12542 ...
 $ Birthdate                    : Date[1:29355], format: "2006-03-01" "2000-01-01" "2000-04-01" ..
 $ Gender                       : chr [1:29355] "Male" "Male" "Male" NA ...
 $ Height                       : num [1:29355] NA NA NA NA 172 NA NA NA 190 190 ...
 $ Position                     : num [1:29355] 1 1 1 1 1 1 1 3 2 8 ...
 $ Examiner.ID                  : num [1:29355] 973 630 3 883 3 ...
 $ Examiner.Role                : num [1:29355] 5 5 2 83 2 2 5 5 5 5 ...
 $ Assessment.Date              : POSIXct[1:29355], format: "2019-08-29 10:19:00" "2018-11-23 07:(
 "2019-09-04 05:42:00" ...
 $ Injury.Date                  : POSIXct[1:29355], format: NA NA NA ...
 $ Concussion.Diagnosed.Decision: chr [1:29355] NA NA NA NA ...
 $ Removed.from.Play            : chr [1:29355] NA NA NA NA ...
 $ Assessment.ID                : num [1:29355] 23481 16438 4393 22487 16996 ...
 $ Fatigue                      : num [1:29355] NA NA 40 NA NA NA 50 NA NA NA ...
 $ Activity.Type                : num [1:29355] 1 1 1 1 1 1 1 1 1 1 ...
 $ Test.collection              : num [1:29355] 4 1 141 4 39 43 3 17 5 5 ...
 $ RTP.ID                       : num [1:29355] 1 1 1 1 1 1 1 1 1 1 ...
  attr(*, "spec")=
```

In this part we are interested to know if the synthpop package generates a similar probability distribution by different categories. To check this matter, we synthesized a dataset from the original dataset and then we used two variables (Sport and Age) to check. The reason we chose the Sport variable because it has many categories compared to the rest.

```
> unique(data$Sport)
 [1]  3  4 10 11  6  5  9  2  7 19  8  1 23 15 22 21 20 12 16 17 13 14
> unique(data$Test.Type)
 [1] "Baseline"             NA                     "SCAT5 Baseline"       "Locker Room"
 [5] "Athlete Self-Test"    "Side Line"            "SCAT5"                "RTP"
 [9] "Incident"             "On-Field"             "Lay"                  "Child SCAT5 Baseline"
> unique(data$Examiner.Role)
  [1]   5   2  83   4   8   1  62  59  17   3  26  18  82  39  31  28  43  58  89  27  11  12  32  51
 [25]  45  65  55  19  44   9   6 105  61  88  78  49  76  96  22  79  54  23   7  34  87  16 102  42
 [49]  35 100  73  21  48  72  80  10  75  63  86  70  77  25  29  67  57  14  64  85  24  60  33  46
 [73]  50  13 104  68  38  94  69  84 101  91  92  52  56  81  37  66  71  53  36  41  30  47  40  95
 [97]  74 103  98  15  99
```

- Sport has 22 categories: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 19, 20, 21, 22, 23. Every number is equivalent with the name of the Sport.
- Age (numerical variable) has different distributions for each category of Sport variable. Age variable comes from Birthday variable. It has the range from 11 to 54.

```r
## import libraries
library(readr)
library(readxl)
library(eeptools)
library(dplyr)
library(openxlsx)
library(ggplot2)
library(ggpubr)
library(tidyverse)
library(outliers)
library(psych)
library(MASS)
library(psych)

##read and explore the data
data <- read_csv("Documents/Langara/Term 4/Capstone/Synthesized_Data.csv")
View(data)
str(data)
dim(data)
unique(data$Sport)
unique(data$Test.Type)
unique(data$Age)
unique(data$Examiner.Role)
```

We apply Synthpop package in R to do synthetic data.

```r
##synthetic data
##Convert Injury date to numeric
data$Injury.Date<-strptime(paste0(as.character(data$Injury.Date)," 00:00"), format = "%d/%m/%Y %H:%M")
str(data$Injury.Date)
data$Injury.Date<-as.numeric(data$Injury.Date)

##create Age variable from the Birthdate variable
data$Age <- as.Date(data$Birthdate, format = "%Y-%m-%d")
data = filter(data, Age >= '1920-01-01' & Age <= '2021-01-01')
data$Age <- floor(age_calc(data$Age, units = "years"))
unique(sort(data$Age))

#Sorting and dropping value out of range
data = filter(data, Age >= 11 & Age <= 60)
unique(sort(syn$Age))
df_new <- data[,-c(6,12)]
head(df_new)
rules.list <- list(
  Injury.Date = "Test.Type ==  'Baseline'",
  Concussion.Diagnosed.Decision = "Test.Type ==  'Baseline' || Test.Type ==  'RTP' || Test.Type ==  'Athlete Self-Test'",
  Removed.from.Play = "Test.Type ==  'Baseline'")
##apply rules for synthesizing
rules.value.list <- list(
  Injury.Date = NA,
  Concussion.Diagnosed.Decision = NA,
  Removed.from.Play = NA)
```

```r
myseed=1020
##Apply synthpop
library(synthpop)
synth.obj <- syn(df_new, method = "cart",rules = rules.list, rvalues = rules.value.list, seed = myseed, maxfaclevels = 430)
##the synthetic data
df_syn <- data.frame(synth.obj$syn)
head(df_syn)
```

Let's check the observations, the mean, the standard deviation, and the histograms of Age of the original data and the synthetic data below:

Age	Observation	Mean	STD	Median	Min	Max
Original data	29,022	21.42	3.68	21	11	54
Synthetic data	29,022	21.42	3.71	21	11	54

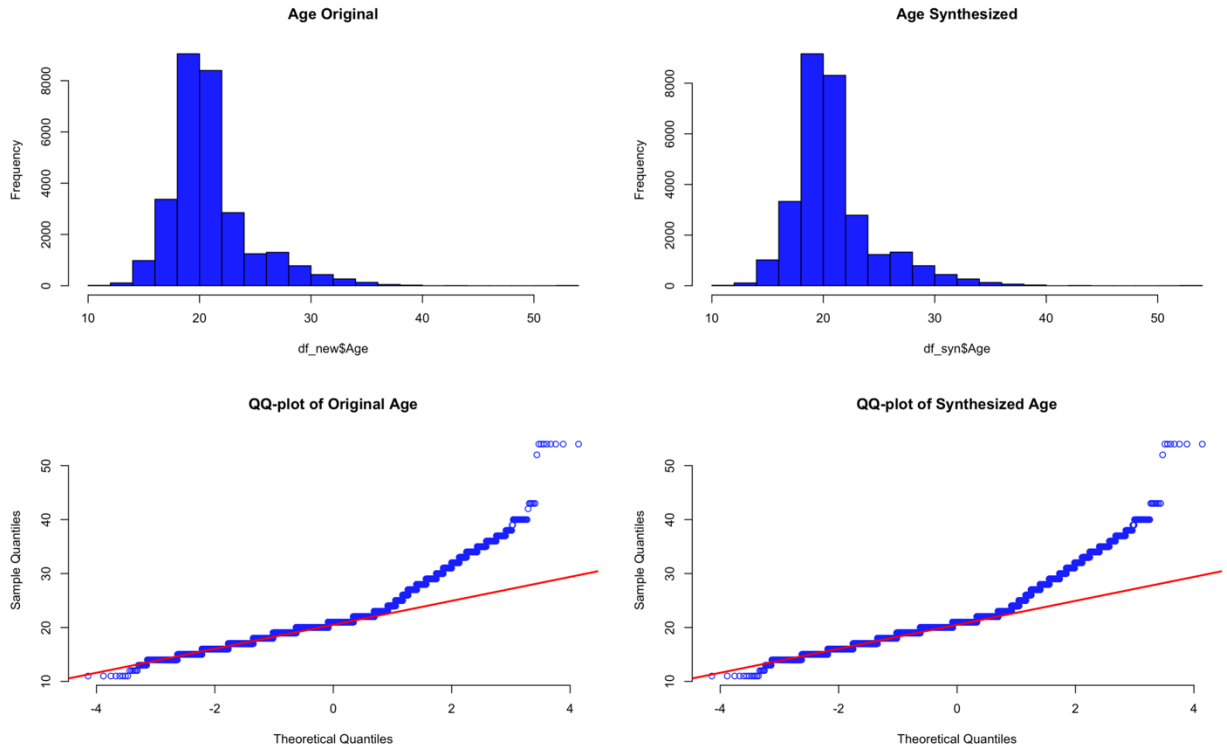


Figure 25: Histogram and QQ-plot of Age original and Age Synthetic

```
##histogram of Age Original and Age Synthesized
hist(df_new$Age,main = "Age Original",col = 'blue')
hist(df_syn$Age,main = "Age Synthesized",col = 'blue')

##QQ plot of Age Original and Age Synthesized
plot(as.factor(df_new$Age),main = "Age Original",col = 'blue')
plot(as.factor(df_syn$Age),main = "Age Synthesized",col = 'blue')

##mean,median,std,min,max
describe(df_new$Age)
describe(df_syn$Age)
```

The frequency table of Sport

- The original data

```
> table(df_new$Sport)

   1    2    3    4    5    6    7    8    9   10   11   12
 246  407 16882 2434 2536  583   94  223 3420  656 1264   13
  13   14   15   16   17   19   20   21   22   23
   4    4   28   25    5   88   17   32   32   29
```

- The synthetic data

```
> table(df_syn$Sport)

   1    2    3    4    5    6    7    8    9   10   11   12
 234  387 16817 2466 2572  578   89  231 3444  659 1257   16
  13   14   15   16   17   19   20   21   22   23
   3    7   27   29    3   98   18   33   29   25
```

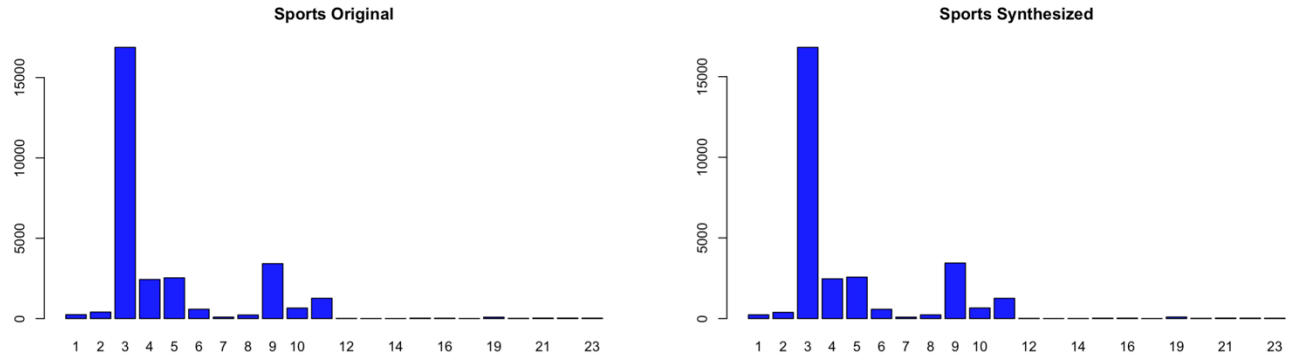


Figure 26: Bar plot of the Original Sports and Synthetic Sports

```
##frequency table for Sports
table(df_new$Sport)
table(df_syn$Sport)

##bar chart of the Sports Original and the Sports Synthesized
plot(as.factor(df_new$Sport),col="blue", main = "Sports Original")
plot(as.factor(df_syn$Sport),col="blue", main = "Sports Synthesized")
```

After checking the distribution of the Age variable of different Sport categories between the original data and the synthetic data, we see most of them have the same distribution. However, there are some special cases that have different distributions so let check some special cases with different distributions between the original data and the synthetic data and some cases just have a few observations. And for most of cases having the same distribution, they are enclosed in the Appendix part for reference.

3. Comparison between the original data and the synthetic data

- Different distribution cases:

For Sport 1:

Sport 1	Observation	Mean	STD	Median	Min	Max
Original data	246	21.7	2.71	21	11	30
Synthetic data	234	21.44	2.48	21	18	29

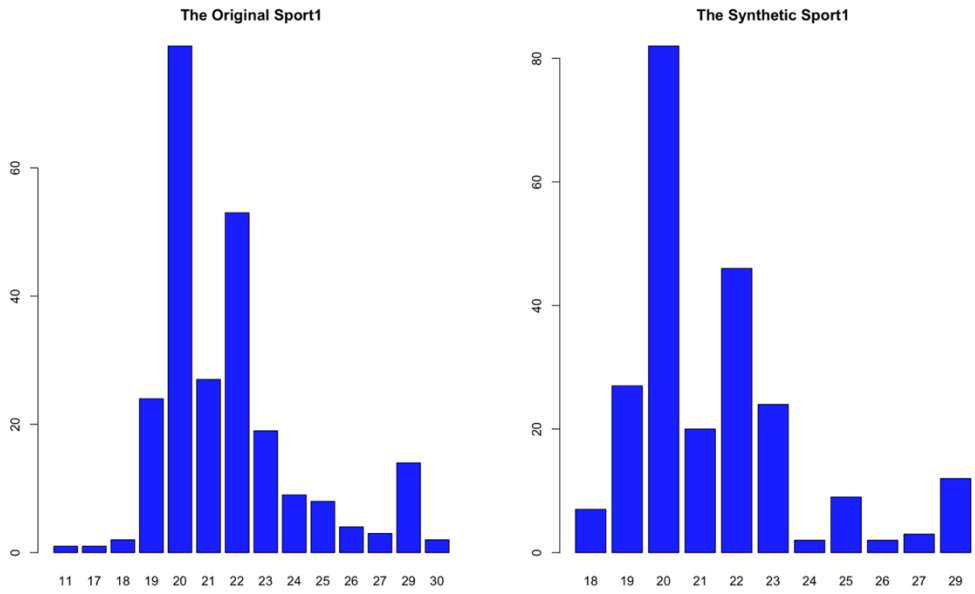


Figure 26: Bar plot of the Original Sport11 and the Synthetic Sport11

We can see that the observation, the mean, the standard deviation, and the median of Age of the original data and the synthetic data are very similar but the min are different. From the original data, we can see the people who are 11-year-old in the Sport 1, but they aren't in the Sport 1 of the synthetic data. Therefore, we can see that with this synthesized method, it

couldn't create the new data with the same range as the original data. The problem is that some people are at 11-year-old they cannot appear in the Sport 1 of the synthetic dataset.

```
###Comparison between the original data and the synthetic data
####sport 1####
par(mfrow = c(1,2))
sport1.df_new <- df_new%>% filter(df_new$Sport == 1)

plot(as.factor(sport1.df_new$Age),col="blue", main = "The Original Sport1")
hist(sport1.df_new$Age,main = "Sport1 The Original Data",col = 'blue')
dim(sport1.df_new)
describe(sport1.df_new$Age)
sport1.df_syn <- df_syn%>% filter(df_syn$Sport == 1)

plot(as.factor(sport1.df_syn$Age),col="blue", main = "The Synthetic Sport1")
hist(sport1.df_syn$Age,main = "Sport1 The Synthesized Data",col = 'blue')
dim(sport1.df_syn)
describe(sport1.df_syn$Age)
```

For Sport 10:

Sport 10	Observation	Mean	STD	Median	Min	Max
Original data	656	18.26	2.55	18	14	54
Synthetic data	659	18.01	1.87	18	11	30

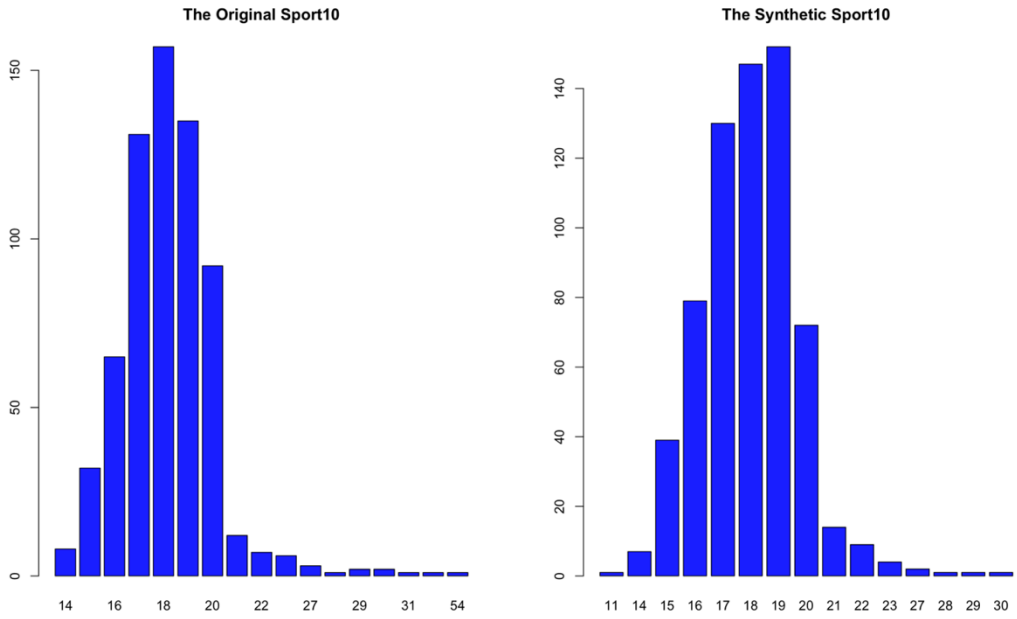


Figure 27: Bar plot of the Original Sport10 and the Synthetic Sport10

```
####sport 10####
sport10.df_new <- df_new%>% filter(df_new$Sport == 10)

plot(as.factor(sport10.df_new$Age),col="blue", main = "The Original Sport10")
hist(sport10.df_new$Age,main = "Sport10 Distribution of The Original Data",col = 'blue')
dim(sport10.df_new)
describe(sport10.df_new)

sport10.df_syn <- df_syn%>% filter(df_syn$Sport == 10)

plot(as.factor(sport10.df_syn$Age),col="blue", main = "The Synthetic Sport10")
hist(sport10.df_syn$Age,main = "Sport10 Distribution of The Synthesized Data",col = 'blue')
dim(sport10.df_syn)
describe(sport10.df_syn)
```

For this case, we can see that the observation, the mean, the standard deviation, and the median of Age of the original data and the synthetic data are very similar but the min are different. From the synthetic data, we can see the people who are 11-year-old in the Sport 1, but they aren't in the Sport 1 of the original data. Age variables have different range of both data so we can see the problem might be that some people who are 11-year-old not qualify for this category Sport 10. And people are older than 31, they are not in the synthetic data.

Let's check to see which categories that the young people (11-year-old) and the old people (54-year-old) belong to.

Young people

- The original data

```
> table(Age11.df_new$Sport)

 1  3 11
 1  5  2
```

The young people (11-year-old) of the original data belong to Sport1, Sport3 and Sport11.

- The synthetic data

```
> table(Age11.df_syn$Sport)

 3  6 10
 6  5  1
```

The young people (11-year-old) of the synthetic data belong to Sport3, Sport6 and Sport10.

From two table above, we can only see the Sport3 belongs to both data, so the problem is the young people don't belong to the Sport6 and Sport10, but the synthetic is created that containing Sport6 and Sport10.

Old people

- The original data

```
> table(Age54.df_new$Sport)

 8  9 10
 5  2  1
```

The old people (54-year-old) of the original data belong to Sport8, Sport9 and Sport10.

- The synthetic data

```
> table(Age54.df_syn$Sport)

 8  9
 4  3
```

The old people (54-year-old) of the synthetic data belong to Sport8, Sport9.

From two table above, we can see the Sport8 and Sport9 belongs to both data so the problem is the old people with Sport10 don't belong to the synthetic data.

```
##young people
Age11.df_new <- df_new%>% filter(df_new$Age == 11)
table(Age11.df_new$Sport)
Age11.df_syn <- df_syn%>% filter(df_syn$Age == 11)
table(Age11.df_syn$Sport)

##old people
Age54.df_new <- df_new%>% filter(df_new$Age == 54)
table(Age54.df_new$Sport)
Age54.df_syn <- df_syn%>% filter(df_syn$Age == 54)
table(Age54.df_syn$Sport)
```

The categories with very small observations:

We check some categories with very small observations to see whether the synthpop package can create the same data as the original one.

- **Sport 13**

Sport 13	Observation	Mean	STD	Median	Min	Max
Original data	4	19.75	0.5	20	19	20
Synthetic data	3	29.33	8.08	34	20	34

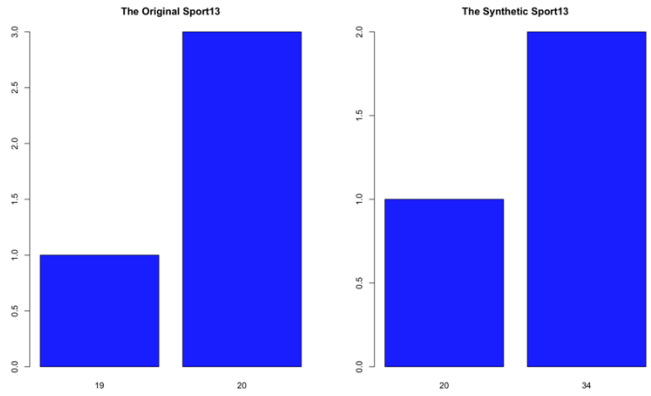


Figure 28: Bar plot of the Original Sport13 and the Synthetic Sport13

```
##The categories with very small observations
####sport 13####
sport13.df_new <- df_new%>% filter(df_new$Sport == 13)
plot(as.factor(sport13.df_new$Age),col="blue", main = "The Original Sport13")
hist(sport13.df_new$Age,main = "Sport13 Distribution of The Original Data",col = 'blue')
dim(sport13.df_new)
describe(sport13.df_new$Age)

sport13.df_syn <- df_syn%>% filter(df_syn$Sport == 13)
plot(as.factor(sport13.df_syn$Age),col="blue", main = "The Synthetic Sport13")
hist(sport13.df_syn$Age,main = "Sport13 Distribution of The Synthesized Data",col = 'blue')
dim(sport13.df_syn)
describe(sport13.df_syn$Age)
```

- **Sport 14**

Sport 14	Observation	Mean	STD	Median	Min	Max
Original data	4	23.25	7.18	20	19	34
Synthetic data	7	22	5.29	20	20	34

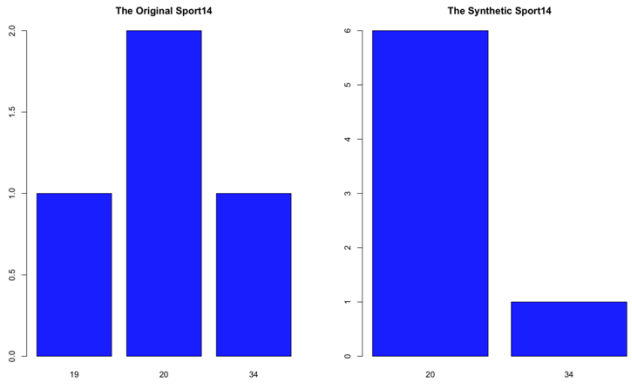


Figure 29: Bar plot of the Original Sport14 and the Synthetic Sport14

```
####sport 14####
sport14.df_new <- df_new%>% filter(df_new$Sport == 14)
plot(as.factor(sport14.df_new$Age),col="blue", main = "The Original Sport14")
hist(sport14.df_new$Age,main = "Sport14 Distribution of The Original Data",col = 'blue')
dim(sport14.df_new)
describe(sport14.df_new$Age)

sport14.df_syn <- df_syn%>% filter(df_syn$Sport == 14)
plot(as.factor(sport14.df_syn$Age),col="blue", main = "The Synthetic Sport14")
hist(sport14.df_syn$Age,main = "Sport14 Distribution of The Synthesized Data",col = 'blue')
dim(sport14.df_syn)
describe(sport14.df_syn$Age)
```

- **Sport 17**

Sport 17	Observation	Mean	STD	Median	Min	Max
Original data	5	22.2	2.17	21	20	25
Synthetic data	3	22	2.65	21	20	25

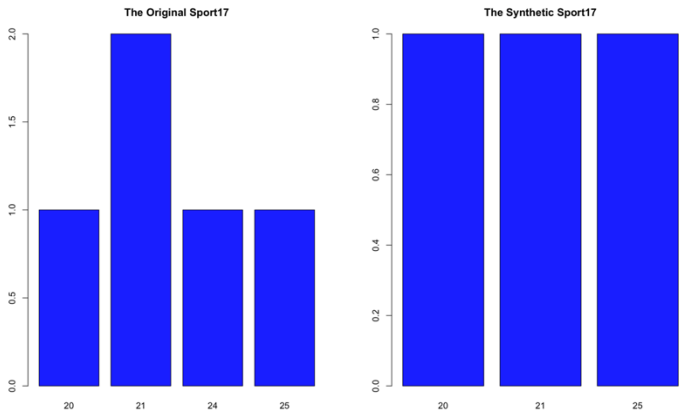


Figure 29: Bar plot of the Original Sport17 and the Synthetic Sport17

```
####sport 17####
sport17.df_new <- df_new%>% filter(df_new$Sport == 17)
plot(as.factor(sport17.df_new$Age),col="blue", main = "The Original Sport17")
hist(sport17.df_new$Age,main = "Sport17 Distribution of The Original Data",col = 'blue')
dim(sport17.df_new)
describe(sport17.df_new$Age)

sport17.df_syn <- df_syn%>% filter(df_syn$Sport == 17)
plot(as.factor(sport17.df_syn$Age),col="blue", main = "The Synthetic Sport17")
hist(sport17.df_syn$Age,main = "Sport17 Distribution of The Synthesized Data",col = 'blue')
dim(sport17.df_syn)
describe(sport17.df_syn$Age)
```

From the above results, we can see the Sport13, the range of Age of the original data is different from the synthetic data. For Sport14 and Sport17, the ranges of Age of both data are similar. Therefore, we can see that the synthpop package cannot create the data has the same range for all categories.

4. Kolmogorov-Smirnov (KS) test

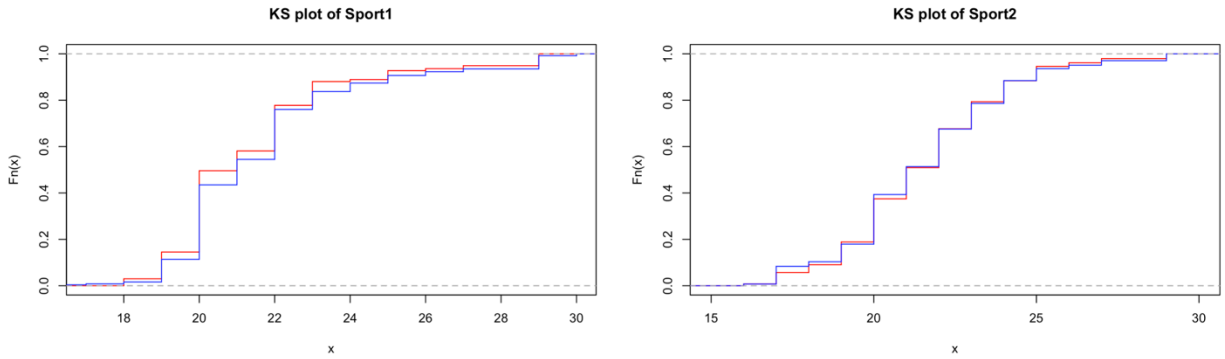
Two-sample KS test can be used to compare the distributions of the observations from the two datasets.
H_o: The two dataset values are from the same continuous distribution.
H_a: The two dataset values are not from the same continuous distribution.

The test result shows below:\

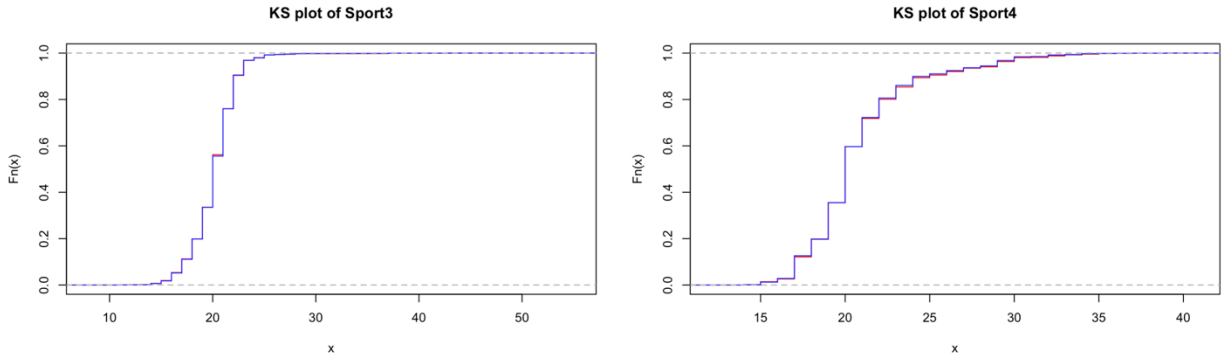
Categories	p-values
Sport 1	0.7677
Sport 2	0.9989
Sport 3	0.9458
Sport 4	1
Sport 5	0.9769
Sport 6	0.4447
Sport 7	0.9998
Sport 8	0.9562
Sport 9	0.9797
Sport 10	0.75
Sport 11	1
Sport 12	0.09417
Sport 13	0.4313
Sport 14	0.9973
Sport 15	0.8729
Sport 16	0.9167
Sport 17	1
Sport 19	1
Sport 20	1
Sport 21	0.9306
Sport 22	1
Sport 23	0.9318

We can see all p_values > 0.05, the significant level so we cannot reject the null hypothesis and conclude that the two dataset values are from the same continuous distribution.

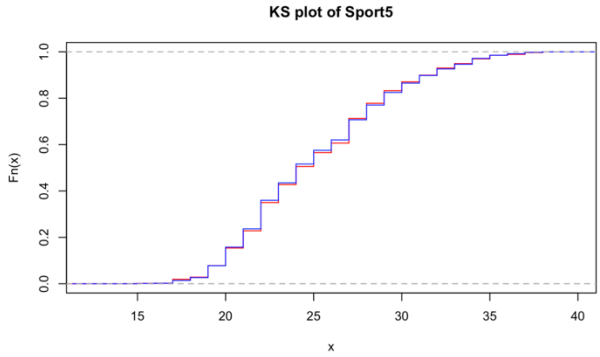
KS plot for Sport Category

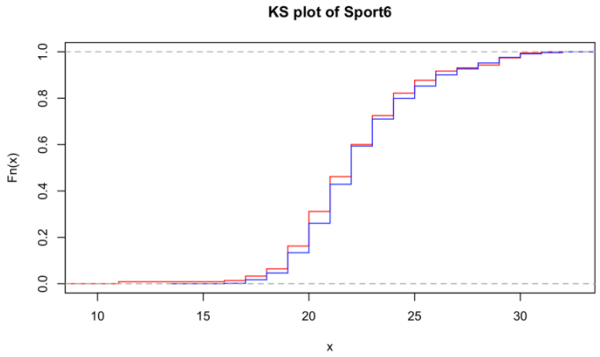

KS plot of Sport1
Fn(x)
x

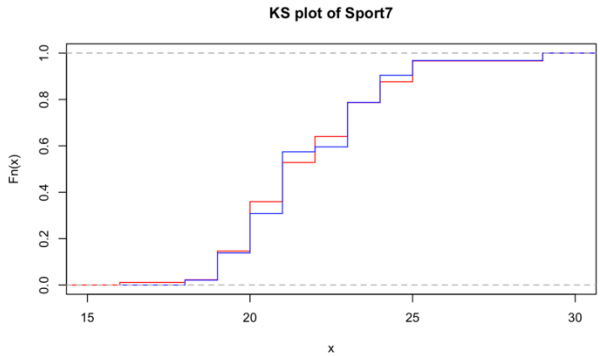
KS plot of Sport2
Fn(x)
x

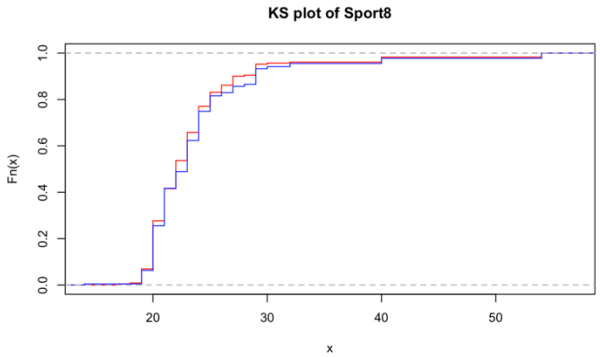

KS plot of Sport3
Fn(x)
x

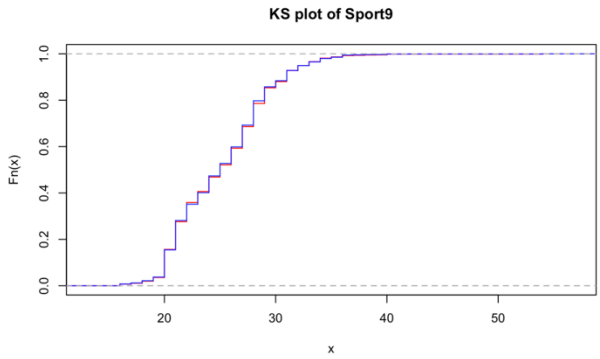
KS plot of Sport4
Fn(x)
x

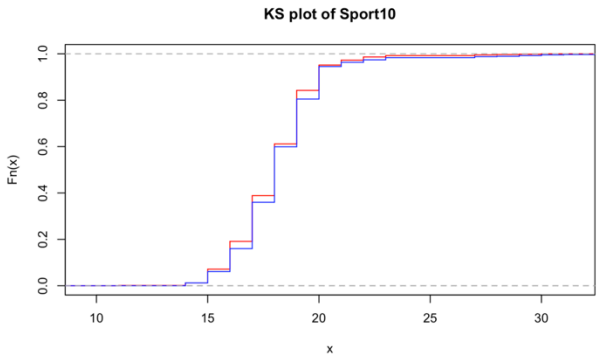

KS plot of Sport5
Fn(x)
x

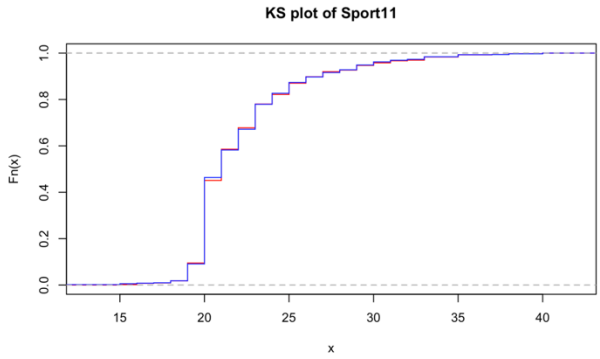

KS plot of Sport6
Fn(x)
x

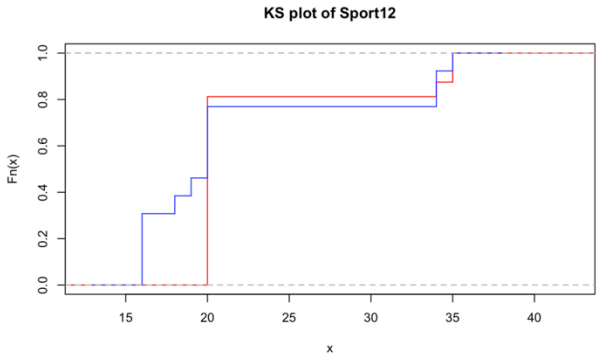

KS plot of Sport7
Fn(x)
x

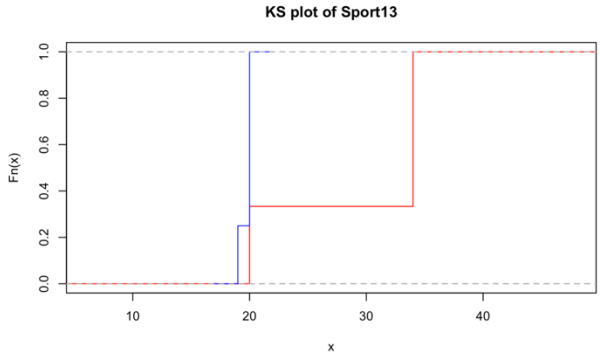

KS plot of Sport8
Fn(x)
x

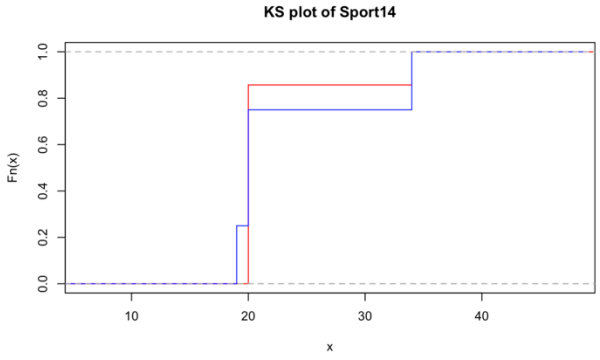

KS plot of Sport9
Fn(x)
x

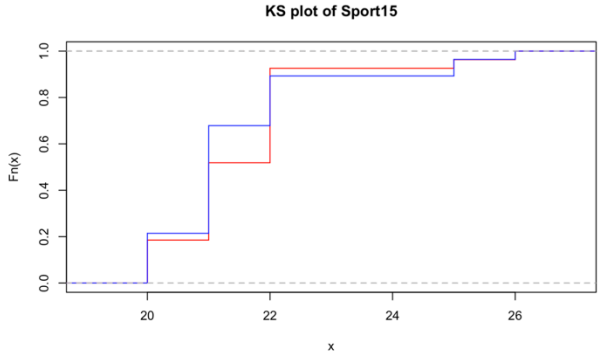

KS plot of Sport10
Fn(x)
x

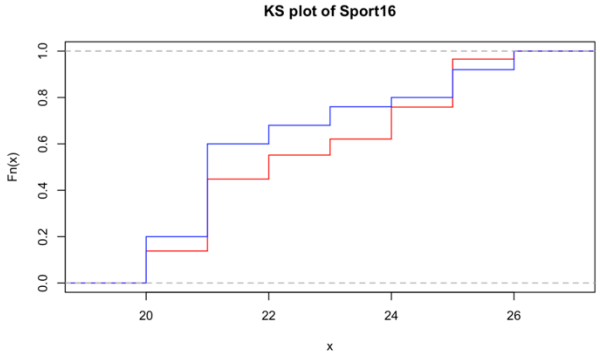

KS plot of Sport11
Fn(x)
x

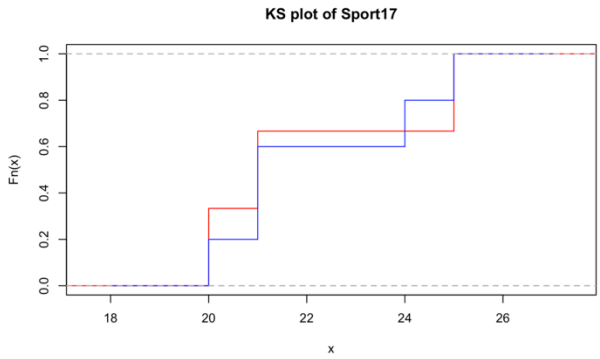

KS plot of Sport12
Fn(x)
x

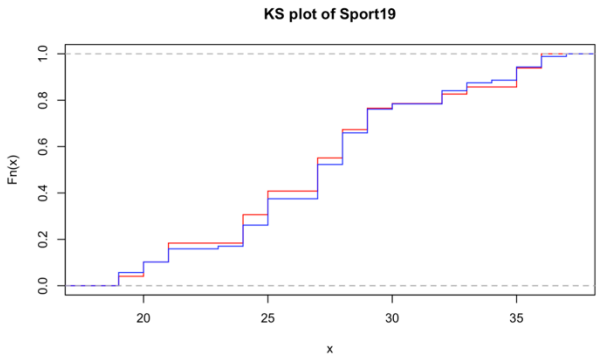

KS plot of Sport13
Fn(x)
x

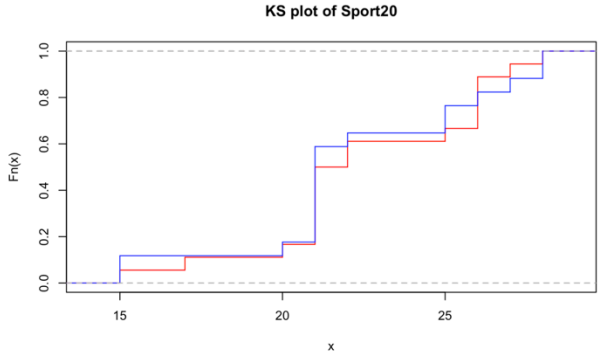

KS plot of Sport14
Fn(x)
x

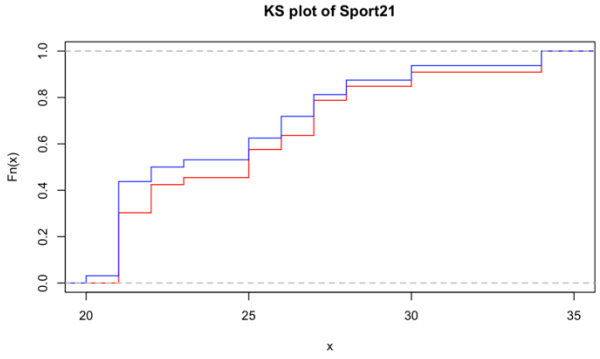

KS plot of Sport15
Fn(x)
x

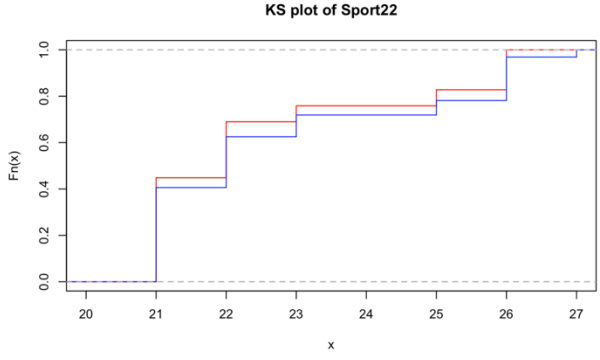

KS plot of Sport16
Fn(x)
x

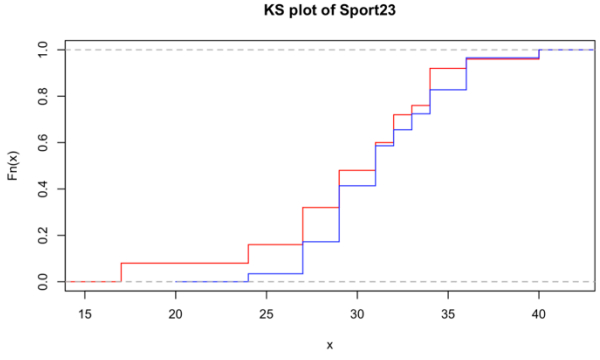

KS plot of Sport17
Fn(x)
x


KS plot of Sport19
Fn(x)
x

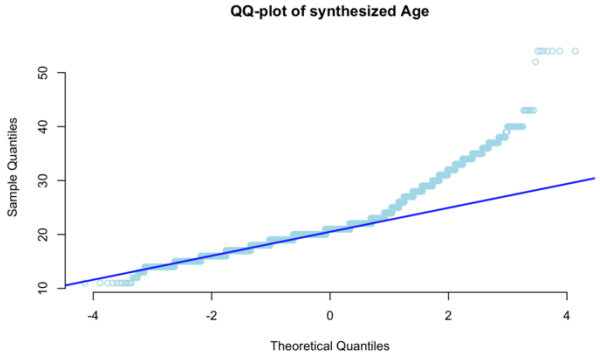

KS plot of Sport20
Fn(x)
x


KS plot of Sport21
Fn(x)
x


KS plot of Sport22
Fn(x)
x


KS plot of Sport23
Fn(x)
x


QQ-plot of original Age
Sample Quantiles
Theoretical Quantiles


QQ-plot of synthesized Age
Sample Quantiles
Theoretical Quantiles

```r
#######ks test######
par(mfrow = c(2,2))
#ks test for sport1
ks.test(sport1.df_syn$Age,sport1.df_new$Age)
plot.ecdf(sport1.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport1")
lines(ecdf(sport1.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport2
ks.test(sport2.df_syn$Age,sport2.df_new$Age)
plot.ecdf(sport2.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport2")
lines(ecdf(sport2.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport3
ks.test(sport3.df_syn$Age,sport3.df_new$Age)
plot.ecdf(sport3.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport3")
lines(ecdf(sport3.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport4
ks.test(sport4.df_syn$Age,sport4.df_new$Age)
plot.ecdf(sport4.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport4")
lines(ecdf(sport4.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport5
ks.test(sport5.df_syn$Age,sport5.df_new$Age)
plot.ecdf(sport5.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport5")
lines(ecdf(sport5.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport6
ks.test(sport6.df_syn$Age,sport6.df_new$Age)
plot.ecdf(sport6.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport6")
lines(ecdf(sport6.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

```r
#ks test for sport7
ks.test(sport7.df_syn$Age,sport7.df_new$Age)
plot.ecdf(sport7.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport7")
lines(ecdf(sport7.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport8
ks.test(sport8.df_syn$Age,sport8.df_new$Age)
plot.ecdf(sport8.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport8")
lines(ecdf(sport8.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport9
ks.test(sport9.df_syn$Age,sport9.df_new$Age)
plot.ecdf(sport9.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport9")
lines(ecdf(sport9.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport10
ks.test(sport10.df_syn$Age,sport10.df_new$Age)
plot.ecdf(sport10.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport10")
lines(ecdf(sport10.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport11
ks.test(sport11.df_syn$Age,sport11.df_new$Age)
plot.ecdf(sport11.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport11")
lines(ecdf(sport11.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport12
ks.test(sport12.df_syn$Age,sport12.df_new$Age)
plot.ecdf(sport12.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport12")
lines(ecdf(sport12.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

```r
#ks test for sport13
ks.test(sport13.df_syn$Age,sport13.df_new$Age)
plot.ecdf(sport13.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport13")
lines(ecdf(sport13.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport14
ks.test(sport14.df_syn$Age,sport14.df_new$Age)
plot.ecdf(sport14.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport14")
lines(ecdf(sport14.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport15
ks.test(sport15.df_syn$Age,sport15.df_new$Age)
plot.ecdf(sport15.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport15")
lines(ecdf(sport15.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport16
ks.test(sport16.df_syn$Age,sport16.df_new$Age)
plot.ecdf(sport16.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport16")
lines(ecdf(sport16.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport17
ks.test(sport17.df_syn$Age,sport17.df_new$Age)
plot.ecdf(sport17.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport17")
lines(ecdf(sport17.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport19
ks.test(sport19.df_syn$Age,sport19.df_new$Age)
plot.ecdf(sport19.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport19")
lines(ecdf(sport19.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport20
ks.test(sport20.df_syn$Age,sport20.df_new$Age)
plot.ecdf(sport20.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport20")
lines(ecdf(sport20.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport21
ks.test(sport21.df_syn$Age,sport21.df_new$Age)
plot.ecdf(sport21.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport21")
lines(ecdf(sport21.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport22
ks.test(sport22.df_syn$Age,sport22.df_new$Age)
plot.ecdf(sport22.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport22")
lines(ecdf(sport22.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport23
ks.test(sport23.df_syn$Age,sport23.df_new$Age)
plot.ecdf(sport23.df_syn$Age, verticals=TRUE, do.points=FALSE, col="red",main = "KS plot of Sport23")
lines(ecdf(sport23.df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")
```

Summary

- Synthesized data is generated based on the overall distribution of each variable across the entire dataset, without considering the distribution within specific categories
- Synthpop handles large datasets without issues, but it may not recognize specific categories within large datasets.
- While synthesized data addresses missing values and outliers, they may be assigned to incorrect categories or variables.
- Some issues can be resolved by applying specific rules, which requires access to the original dataset and detailed descriptions of its variables.

Part D

Generating datasets with different sample size by applying synthpop to check the time of generating and comparing the datasets

1. Introduction

In this section, we tried to generate synthetic data with different sizes and monitored the time of generating data. We recorded time for three different datasets including Health Insurance, Heart Disease, and Headcheck. In addition, we check the distribution and statistical inference of synthesized data with small size from Heahcheck whether it is like the original dataset or not.

2. Time of generating data

- Heart Disease dataset

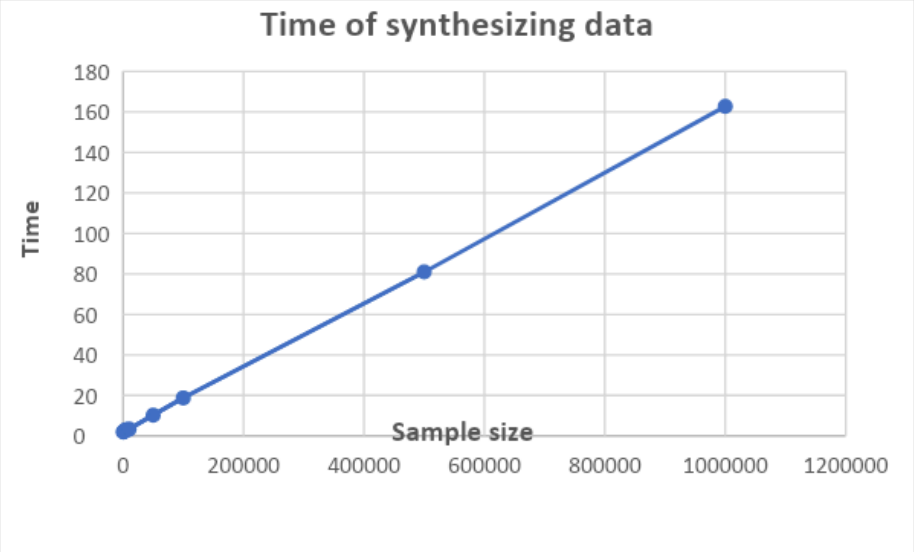
```
> str(Framingham_df)
'data.frame':   4238 obs. of  16 variables:
 $ male           : Factor w/ 2 levels "0","1": 2 1 2 1 1 1 1 1 2 2 ...
 $ age            : num  39 46 48 61 46 43 63 45 52 43 ...
 $ education      : Factor w/ 4 levels "1","2","3","4": 4 2 1 3 3 2 1 2 1 1 ...
 $ currentSmoker  : Factor w/ 2 levels "0","1": 1 1 2 2 2 1 1 2 1 2 ...
 $ cigsPerDay     : num  0 0 20 30 23 0 0 20 0 30 ...
 $ BPMeds         : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ prevalentStroke: Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ prevalentHyp   : Factor w/ 2 levels "0","1": 1 1 1 2 1 2 1 1 2 2 ...
 $ diabetes       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ totChol        : num  195 250 245 225 285 228 205 313 260 225 ...
 $ sysBP          : num  106 121 128 150 130 ...
 $ diaBP          : num  70 81 80 95 84 110 71 71 89 107 ...
 $ BMI            : num  27 28.7 25.3 28.6 23.1 ...
 $ heartRate      : num  80 95 75 65 85 77 60 79 76 93 ...
 $ glucose        : num  77 76 70 103 85 99 85 78 79 88 ...
 $ TenYearCHD     : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 2 1 1 1 ...
```

Heart Disease included 16 variables and 4238 observations. Table xxx shows the generated datasets with different sizes and duration of generating.

Table xxx

Size	Time (Second)
500	1.92
1000	2.10
2000	2.23
4238	3.05
10000	3.33

50000	10.23
100000	18.77
500000	80.94
1000000	162.72



The graph in Figure xxx shows that there is a linear relationship between sample size and time of generating data.

```
myseed=1020
##Applying synthpop
library(synthpop)

ptm <- proc.time()
synth.obj <- syn(Framingham_df, method = "cart",rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram <- data.frame(synth.obj$syn)
# generate 500
ptm <- proc.time()
synth.obj1 <- syn(Framingham_df, method = "cart", k=500, rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram1 <- data.frame(synth.obj1$syn)
# generate 1000
ptm <- proc.time()
synth.obj2 <- syn(Framingham_df, method = "cart", k=1000, rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram2 <- data.frame(synth.obj2$syn)
# generate 2000
ptm <- proc.time()
synth.obj3 <- syn(Framingham_df, method = "cart", k=2000, rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram3 <- data.frame(synth.obj3$syn)
# generate 10000
ptm <- proc.time()
synth.obj4 <- syn(Framingham_df, method = "cart", k=10000, rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram4 <- data.frame(synth.obj4$syn)
# generate 50000
ptm <- proc.time()
synth.obj5 <- syn(Framingham_df, method = "cart", k=50000, rules = rules.list, rvalues = rules.value.list, seed = myseed)
proc.time() - ptm
s_fram5 <- data.frame(synth.obj5$syn)
# generate 100000
ptm <- proc.time()
synth.obj6 <- syn(Framingham_df, method = "cart", k=100000, rules = rules.list, rvalues = rules.value.list, seed = myseed)
```

- Health Insurance dataset

```
> str(HI)
'data.frame':   381109 obs. of  12 variables:
 $ id                  : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender              : chr  "Male" "Male" "Male" "Male" ...
 $ Age                 : int  44 76 47 21 29 24 23 56 24 32 ...
 $ Driving_License     : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Region_Code         : int  28 3 28 11 41 33 11 28 3 6 ...
 $ Previously_Insured  : int  0 0 0 1 1 0 0 0 1 1 ...
 $ Vehicle_Age         : chr  "> 2 Years" "1-2 Year" "> 2 Years" "< 1 Year" ...
 $ Vehicle_Damage      : chr  "Yes" "No" "Yes" "No" ...
 $ Annual_Premium      : int  40454 33536 38294 28619 27496 2630 23367 32031 27619 28771
 ...
 $ Policy_Sales_Channel: int  26 26 26 152 152 160 152 26 152 152 ...
 $ Vintage             : int  217 183 27 203 39 176 249 72 28 80 ...
 $ Response            : int  1 0 1 0 0 0 0 1 0 0 ...
```

Health insurance has 12 variables and 381109 observations. For generating a dataset same as original dataset, 381109, it took minimum 28 minutes by using different machines, hence we just generated a sample size of one million which it took 49 minutes on the same machine.

Table xx1

Size	Time (Minute)
381,109	28
1,000,000	49

- Headcheck dataset

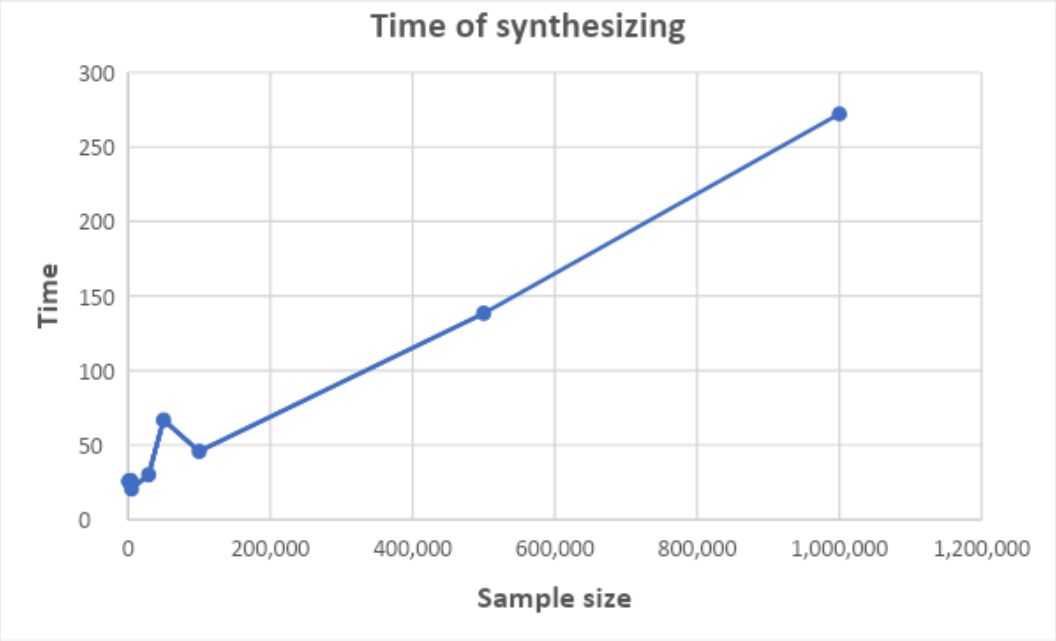
```
> str(df_new)
'data.frame':   29355 obs. of  19 variables:
 $ Test.Type                    : chr  "Baseline" "Baseline" "Baseline" "Baseline" ...
 $ Org.ID                       : int  36 23 4 32 24 7 16 56 27 27 ...
 $ Team.ID                      : int  850 528 109 789 552 168 430 1067 630 640 ...
 $ Sport                        : int  3 4 3 3 3 10 4 11 3 3 ...
 $ Athlete.ID                   : int  18146 12079 2576 17288 12542 4093 9346 21566 14025 14224 ...
 $ Gender                       : chr  "Male" "Male" "Male" NA ...
 $ Height                       : num  NA NA NA NA 172 NA NA NA 190 190 ...
 $ Position                     : int  1 1 1 1 1 1 1 3 2 8 ...
 $ Examiner.ID                  : int  973 630 3 883 3 3 445 1234 715 715 ...
 $ Examiner.Role                : int  5 5 2 83 2 2 5 5 5 5 ...
 $ Injury.Date                  : num  NA NA NA NA NA NA NA NA NA NA ...
 $ Concussion.Diagnosed.Decision: chr  NA NA NA NA ...
 $ Removed.from.Play            : chr  NA NA NA NA ...
 $ Assessment.ID                : int  23481 16438 4393 22487 16996 6292 12791 27997 18633 18855 ...
 $ Fatigue                      : int  NA NA 40 NA NA NA 50 NA NA NA ...
 $ Activity.Type                : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Test.collection              : int  4 1 141 4 39 43 3 17 5 5 ...
 $ RTP.ID                       : int  1 1 1 1 1 1 1 1 1 1 ...
 $ Age                          : Date, format: "2006-03-01" "2000-01-01" "2000-04-01" ...
```

Headcheck dataset included 19 variables and 29022 observations. In this dataset, we created variable Age based on date of birth for further analysis. Table xx2 shows the result of time of generating datasets. The result shows that the relationship between sample size and duration of generating for this dataset is not linear.for example time of generating 1000 data is slightly

higher than 2000 observations and time for 4000 is higher than 5000. Time of generation for 5 million observations is much higher compared to other sizes.

Table xx2

Size	Time (Minute)
1,000	25.76
2,000	25.20
4,000	26.33
5,000	20.57
29,022	30.03
40,000	66.70
100,000	45.75
500,000	138.37
1,000,000	272.168
5,000,000	1281.15



According to the result for Headcheck dataset, it illustrated that the time of generating different sample size depends on the size of original dataset and the type of variables. It is also related to the configuration of machine which is used for running codes. We tried with different machins and we got different time for each level.

3. Compare statistical distribution of variables for original and synthesized data

Since generating a large dataset takes long time, therefore we tried to generate a dataset with smaller sizes of original size of Headcheck and compared the synthesized datasets whether they transfer the complete information from original dataset or not.

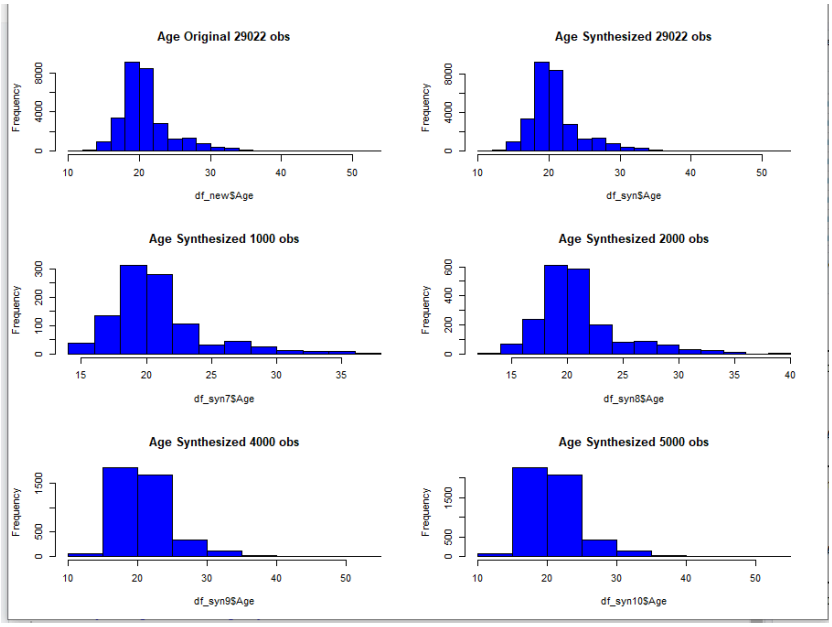
All the plots and tables show that in general small sample size of synthesized data have the same statistical distribution with original dataset.

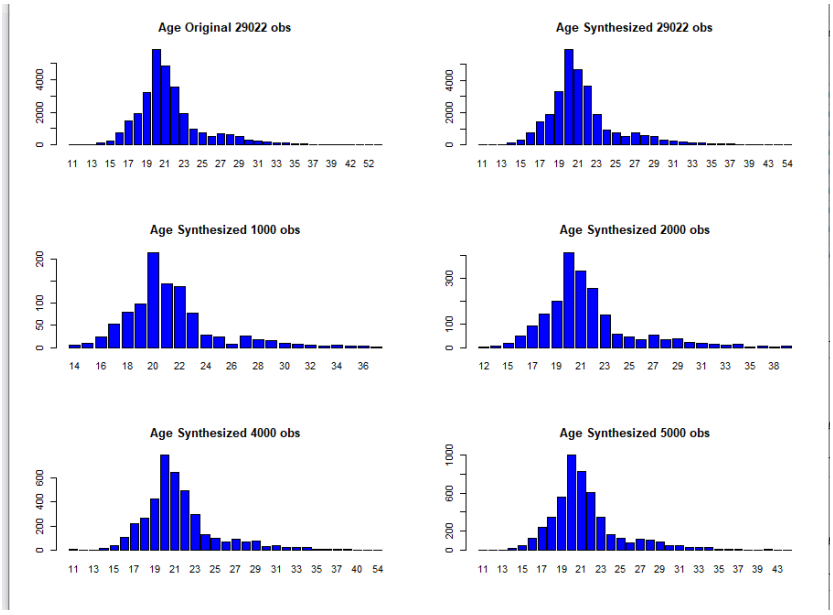
```
### describe data sets
describe(df_new$Age)
describe(df_syn$Age)
describe(df_syn7$Age)
describe(df_syn8$Age)
describe(df_syn9$Age)
describe(df_syn10$Age)

table(df_new$Age)
table(df_syn10$Age)
```

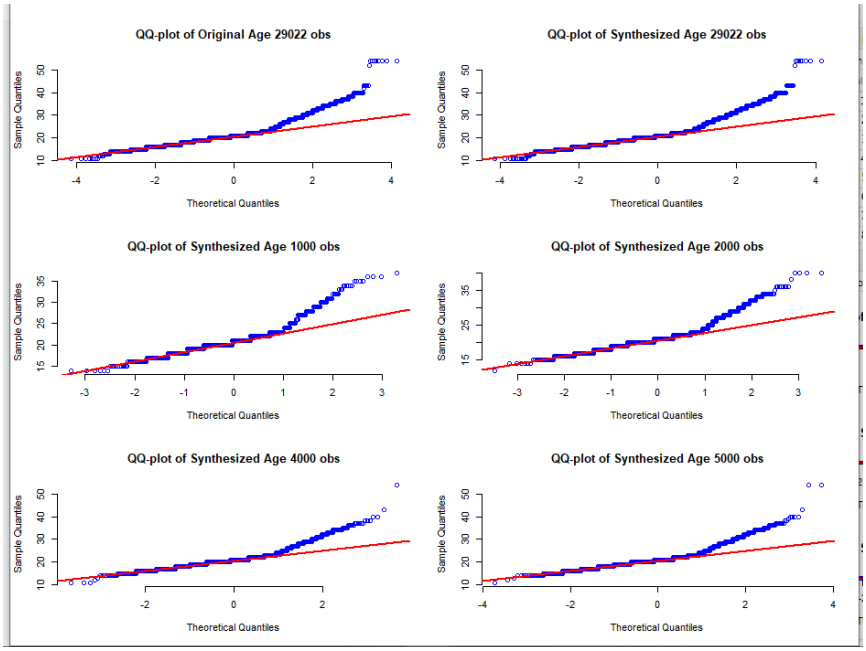
```
> ### describe data sets
> describe(df_new$Age)
   vars     n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 29022 21.42 3.68     21   20.97 2.97  11  54    43 1.57     4.56 0.02
> describe(df_syn$Age)
   vars     n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 29022 21.42 3.71     21   20.97 2.97  11  54    43 1.57     4.42 0.02
> describe(df_syn7$Age)
   vars    n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 1000 21.28 3.57     21   20.84 2.97  14  37    23 1.42     2.94 0.11
> describe(df_syn8$Age)
   vars    n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 2000 21.44 3.67     21   20.98 2.97  12  40    28 1.48      3.2 0.08
> describe(df_syn9$Age)
   vars    n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 4000 21.38 3.68     21   20.93 2.97  11  54    43 1.55     4.43 0.06
> describe(df_syn10$Age)
   vars    n  mean   sd median trimmed  mad min max range skew kurtosis   se
X1    1 5000 21.44 3.75     21   20.97 2.97  11  54    43 1.65     5.03 0.05
>
```

Dataset	n	Mean	SD	Median	Min	Max
Original	29022	21.42	3.68	21	11	54
Syn1	29022	21.42	3.71	21	11	54
Syn2	1000	21.28	3.57	21	14	37
Syn3	2000	21.44	3.67	21	12	40
Syn4	4000	21.38	3.68	21	11	54
Syn5	5000	21.44	3.75	21	11	54

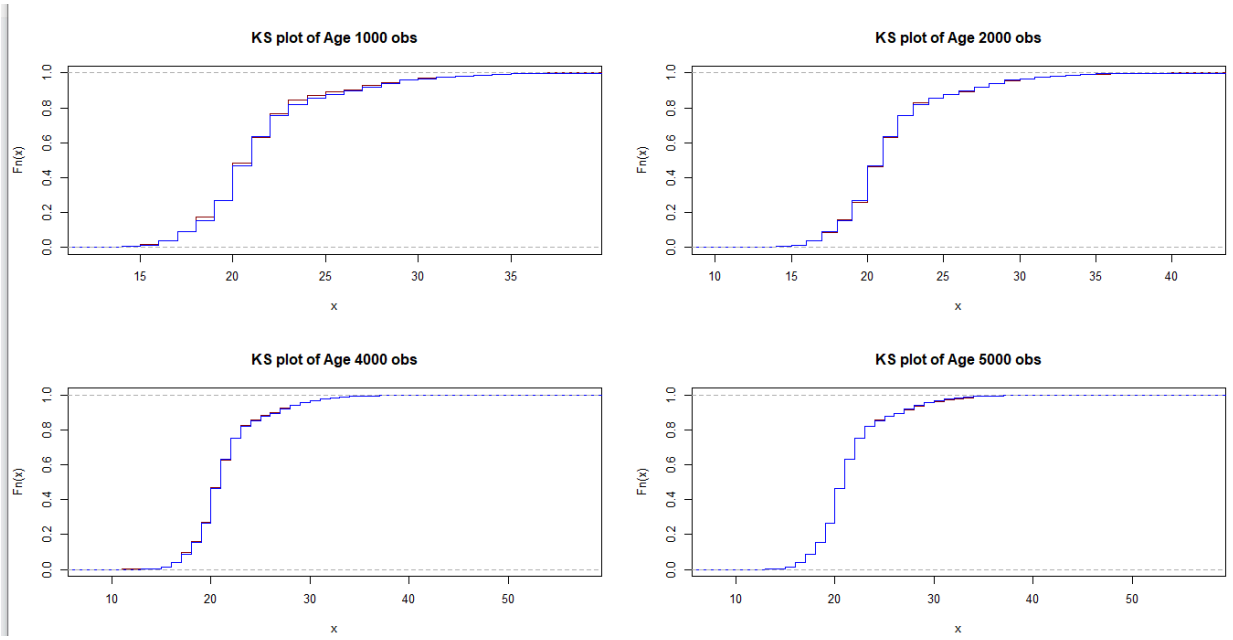
```r
#compare(df_syn1, df_new)$plot
# Age
plot(as.factor(df_new$Age),main = "Age Original 29022 obs",col = 'blue')
plot(as.factor(df_syn$Age),main = "Age Synthesized 29022 obs",col = 'blue')
plot(as.factor(df_syn7$Age),main = "Age Synthesized 1000 obs",col = 'blue')
plot(as.factor(df_syn8$Age),main = "Age Synthesized 2000 obs",col = 'blue')
plot(as.factor(df_syn9$Age),main = "Age Synthesized 4000 obs",col = 'blue')
plot(as.factor(df_syn10$Age),main = "Age Synthesized 5000 obs",col = 'blue')


qqnorm(df_new$Age, pch = 1, frame = FALSE, main = "QQ-plot of Original Age 29022 obs", col = "blue")
qqline(df_new$Age, col = "red", lwd = 2)
qqnorm(df_syn$Age, pch = 1, frame = FALSE, main = "QQ-plot of Synthesized Age 29022 obs", col = "blue")
qqline(df_syn$Age, col = "red", lwd = 2)
qqnorm(df_syn7$Age, pch = 1, frame = FALSE, main = "QQ-plot of Synthesized Age 1000 obs", col = "blue")
qqline(df_syn7$Age, col = "red", lwd = 2)
qqnorm(df_syn8$Age, pch = 1, frame = FALSE, main = "QQ-plot of Synthesized Age 2000 obs", col = "blue")
qqline(df_syn8$Age, col = "red", lwd = 2)
qqnorm(df_syn9$Age, pch = 1, frame = FALSE, main = "QQ-plot of Synthesized Age 4000 obs", col = "blue")
qqline(df_syn9$Age, col = "red", lwd = 2)
qqnorm(df_syn10$Age, pch = 1, frame = FALSE, main = "QQ-plot of Synthesized Age 5000 obs", col = "blue")
qqline(df_syn10$Age, col = "red", lwd = 2)
```




Age Original 29022 obs
Age Synthesized 29022 obs
Age Synthesized 1000 obs
Age Synthesized 2000 obs
Age Synthesized 4000 obs
Age Synthesized 5000 obs


QQ-plot of Original Age 29022 obs
QQ-plot of Synthesized Age 29022 obs
QQ-plot of Synthesized Age 1000 obs
QQ-plot of Synthesized Age 2000 obs
QQ-plot of Synthesized Age 4000 obs
QQ-plot of Synthesized Age 5000 obs
Sample Quantiles
Theoretical Quantiles

```r
par(mfrow = c(2,2))
#ks test for sport1
ks.test(df_syn7$Age,df_new$Age)
plot.ecdf(df_syn7$Age, verticals=TRUE, do.points=FALSE, col="dark red",main = "KS plot of Age 1000 obs")
lines(ecdf(df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport1
ks.test(df_syn8$Age,df_new$Age)
plot.ecdf(df_syn8$Age, verticals=TRUE, do.points=FALSE, col="dark red",main = "KS plot of Age 2000 obs")
lines(ecdf(df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport1
ks.test(df_syn9$Age,df_new$Age)
plot.ecdf(df_syn9$Age, verticals=TRUE, do.points=FALSE, col="dark red",main = "KS plot of Age 4000 obs")
lines(ecdf(df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")

#ks test for sport1
ks.test(df_syn10$Age,df_new$Age)
plot.ecdf(df_syn10$Age, verticals=TRUE, do.points=FALSE, col="dark red",main = "KS plot of Age 5000 obs")
lines(ecdf(df_new$Age), verticals=TRUE, do.points=FALSE, col="blue")
```



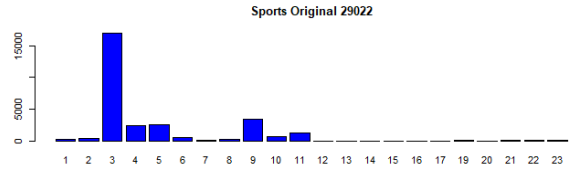

```
        Two-sample Kolmogorov-Smirnov test

data:  df_syn7$Age and df_new$Age
D = 0.021726, p-value = 0.7515
alternative hypothesis: two-sided
```

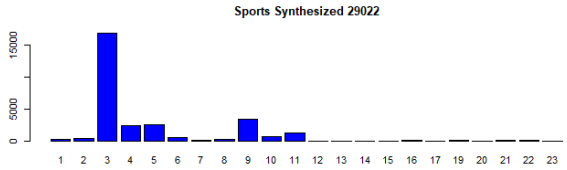
```
        Two-sample Kolmogorov-Smirnov test

data:  df_syn8$Age and df_new$Age
D = 0.0084883, p-value = 0.9993
alternative hypothesis: two-sided
```

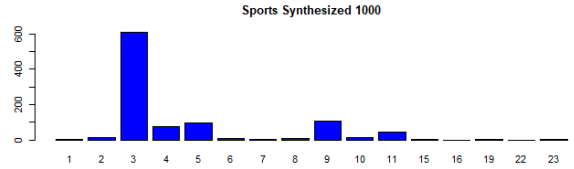
```
        Two-sample Kolmogorov-Smirnov test

data:  df_syn9$Age and df_new$Age
D = 0.0082377, p-value = 0.9709
alternative hypothesis: two-sided
```

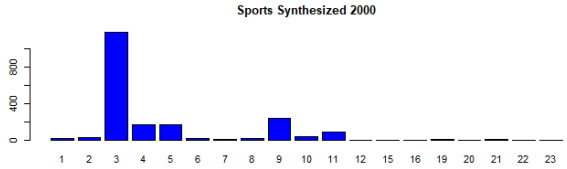
```
        Two-sample Kolmogorov-Smirnov test

data:  df_syn10$Age and df_new$Age
D = 0.0036999, p-value = 1
alternative hypothesis: two-sided
```

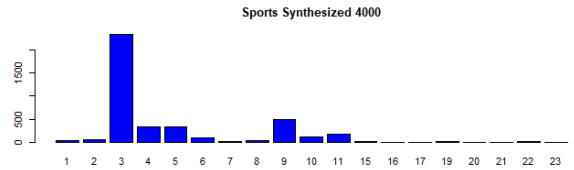
```r
## plots for variables sport
plot(as.factor(df_new$Sport),col="blue", main = "Sports Original 29022")
plot(as.factor(df_syn$Sport),col="blue", main = "Sports Synthesized 29022")
plot(as.factor(df_syn7$Sport),col="blue", main = "Sports Synthesized 1000")
plot(as.factor(df_syn8$Sport),col="blue", main = "Sports Synthesized 2000")
plot(as.factor(df_syn9$Sport),col="blue", main = "Sports Synthesized 4000")
plot(as.factor(df_syn10$Sport),col="blue", main = "Sports Synthesized 5000")
```

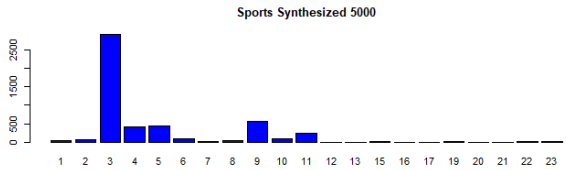




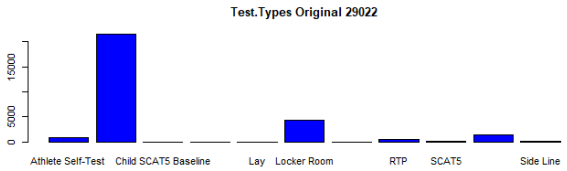


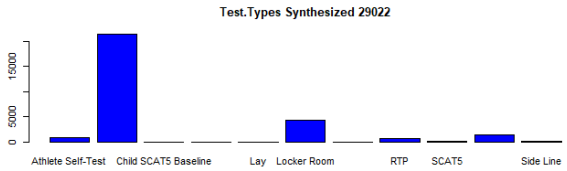


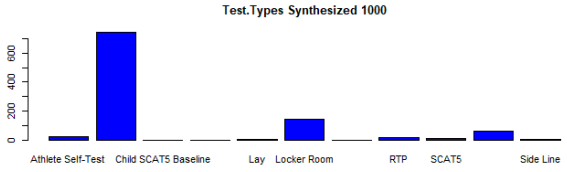


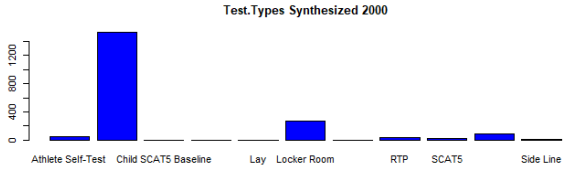


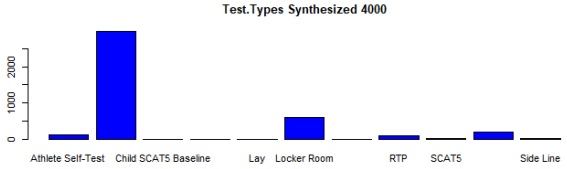

```r
## plots for variables test.type
plot(as.factor(df_new$Test.Type),col="blue", main = "Test.Types Original 29022")
plot(as.factor(df_syn$Test.Type),col="blue", main = "Test.Types Synthesized 29022")
plot(as.factor(df_syn7$Test.Type),col="blue", main = "Test.Types Synthesized 1000")
plot(as.factor(df_syn8$Test.Type),col="blue", main = "Test.Types Synthesized 2000")
plot(as.factor(df_syn9$Test.Type),col="blue", main = "Test.Types Synthesized 4000")
plot(as.factor(df_syn10$Test.Type),col="blue", main = "Test.Types Synthesized 5000")
```

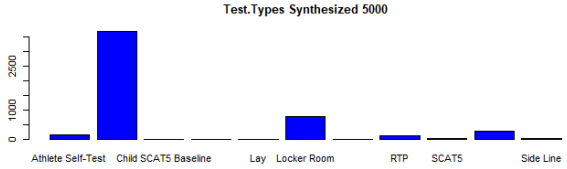




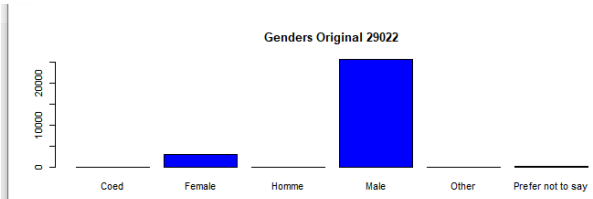


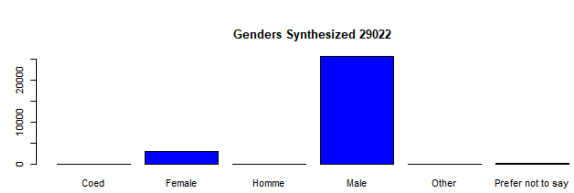


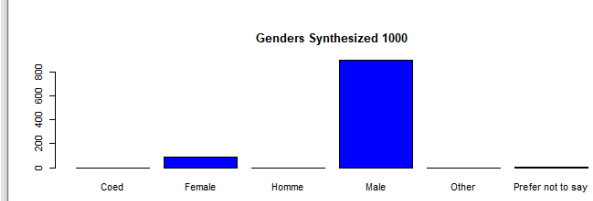


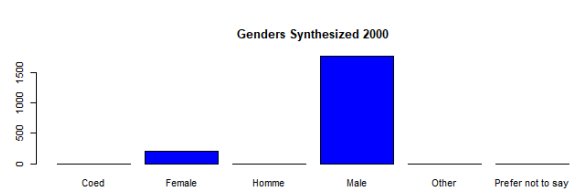


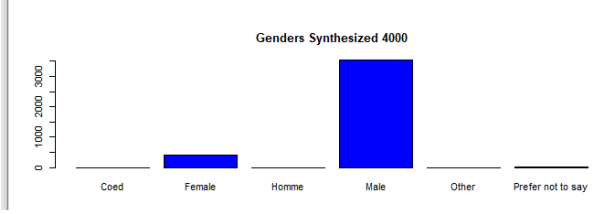

```r
## plots for variables Gender
plot(as.factor(df_new$Gender),col="blue", main = "Genders Original 29022")
plot(as.factor(df_syn$Gender),col="blue", main = "Genders Synthesized 29022")
plot(as.factor(df_syn7$Gender),col="blue", main = "Genders Synthesized 1000")
plot(as.factor(df_syn8$Gender),col="blue", main = "Genders Synthesized 2000")
plot(as.factor(df_syn9$Gender),col="blue", main = "Genders Synthesized 4000")
plot(as.factor(df_syn10$Gender),col="blue", main = "Genders Synthesized 5000")
```

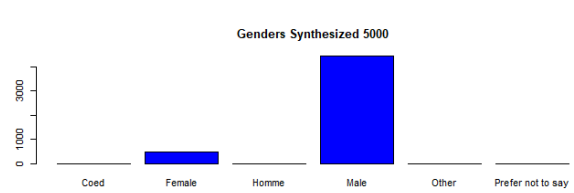




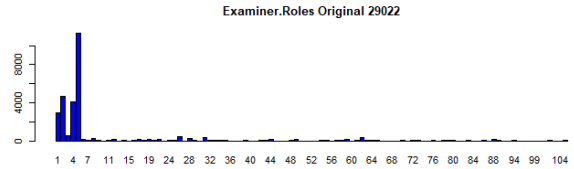


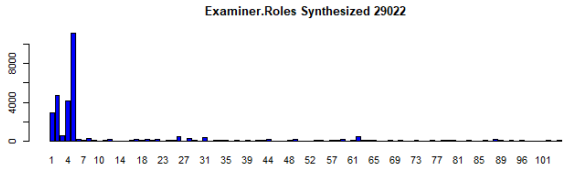


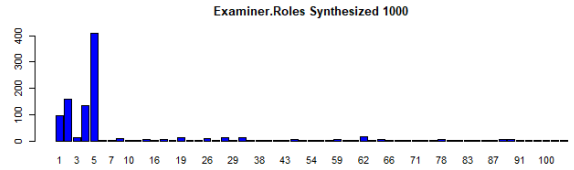


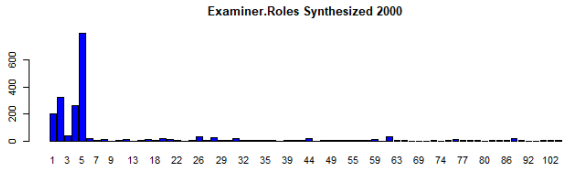


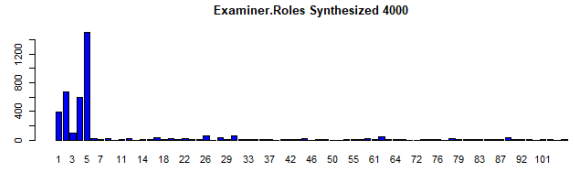
```
## plots for variables Examiner.Role
plot(as.factor(df_new$Examiner.Role),col="blue", main = "Examiner.Roles Original 29022")
plot(as.factor(df_syn$Examiner.Role),col="blue", main = "Examiner.Roles Synthesized 29022")
plot(as.factor(df_syn7$Examiner.Role),col="blue", main = "Examiner.Roles Synthesized 1000")
plot(as.factor(df_syn8$Examiner.Role),col="blue", main = "Examiner.Roles Synthesized 2000")
plot(as.factor(df_syn9$Examiner.Role),col="blue", main = "Examiner.Roles Synthesized 4000")
plot(as.factor(df_syn10$Examiner.Role),col="blue", main = "Examiner.Roles Synthesized 5000")
```

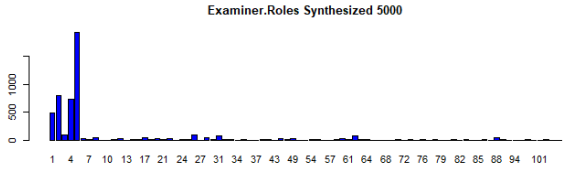




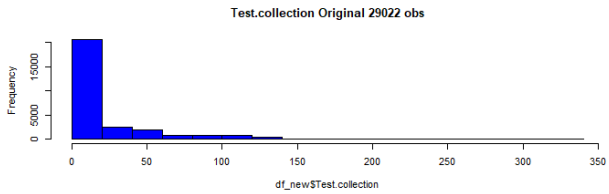


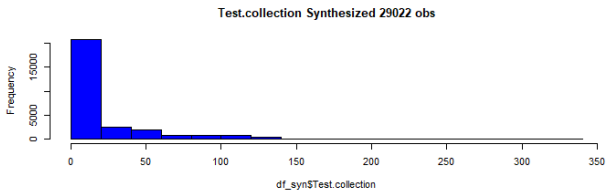


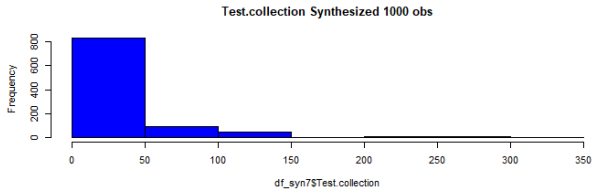


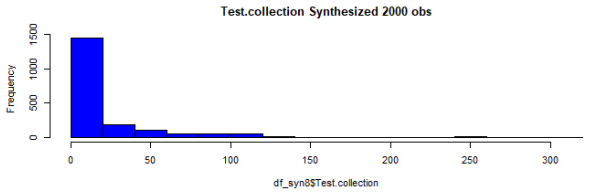


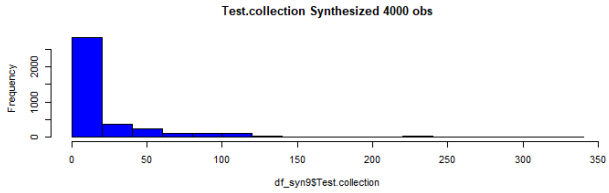

```
### hist for small samples Test collections
hist(df_new$Test.collection,main = "Test.collection Original 29022 obs",col = 'blue')
hist(df_syn$Test.collection,main = "Test.collection Synthesized 29022 obs",col = 'blue')
hist(df_syn7$Test.collection,main = "Test.collection Synthesized 1000 obs",col = 'blue')
hist(df_syn8$Test.collection,main = "Test.collection Synthesized 2000 obs",col = 'blue')
hist(df_syn9$Test.collection,main = "Test.collection Synthesized 4000 obs",col = 'blue')
hist(df_syn10$Test.collection,main = "Test.collection Synthesized 5000 obs",col = 'blue')

## plots for variables Test.collection
plot(as.factor(df_new$Test.collection),col="blue", main = "Test.collections Original 29022")
plot(as.factor(df_syn$Test.collection),col="blue", main = "Test.collections Synthesized 29022")
plot(as.factor(df_syn7$Test.collection),col="blue", main = "Test.collections Synthesized 1000")
plot(as.factor(df_syn8$Test.collection),col="blue", main = "Test.collections Synthesized 2000")
plot(as.factor(df_syn9$Test.collection),col="blue", main = "Test.collections Synthesized 4000")
plot(as.factor(df_syn10$Test.collection),col="blue", main = "Test.collections Synthesized 5000")
```

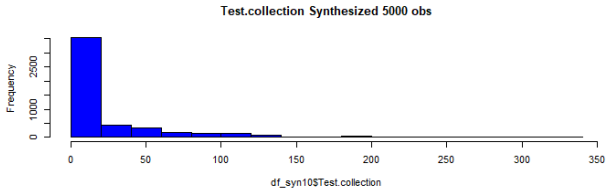


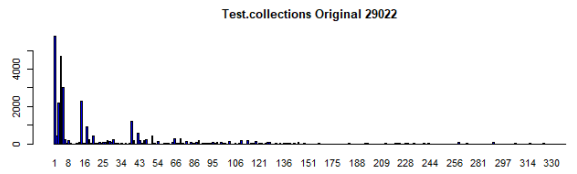


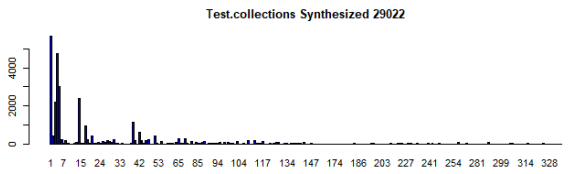


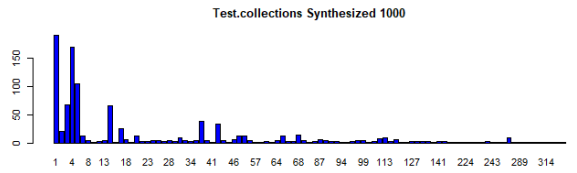


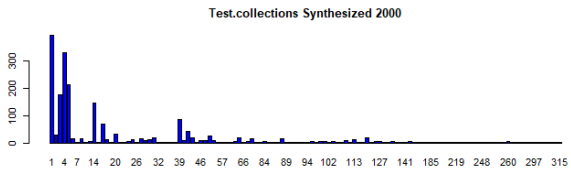


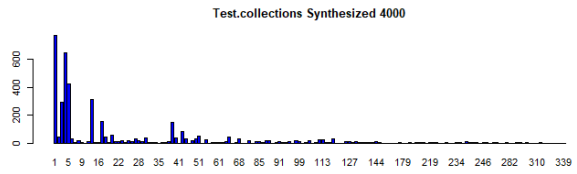


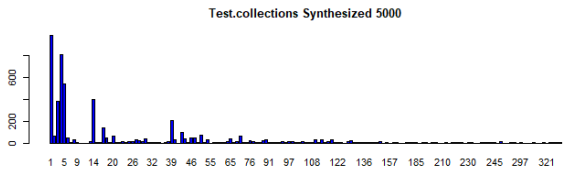










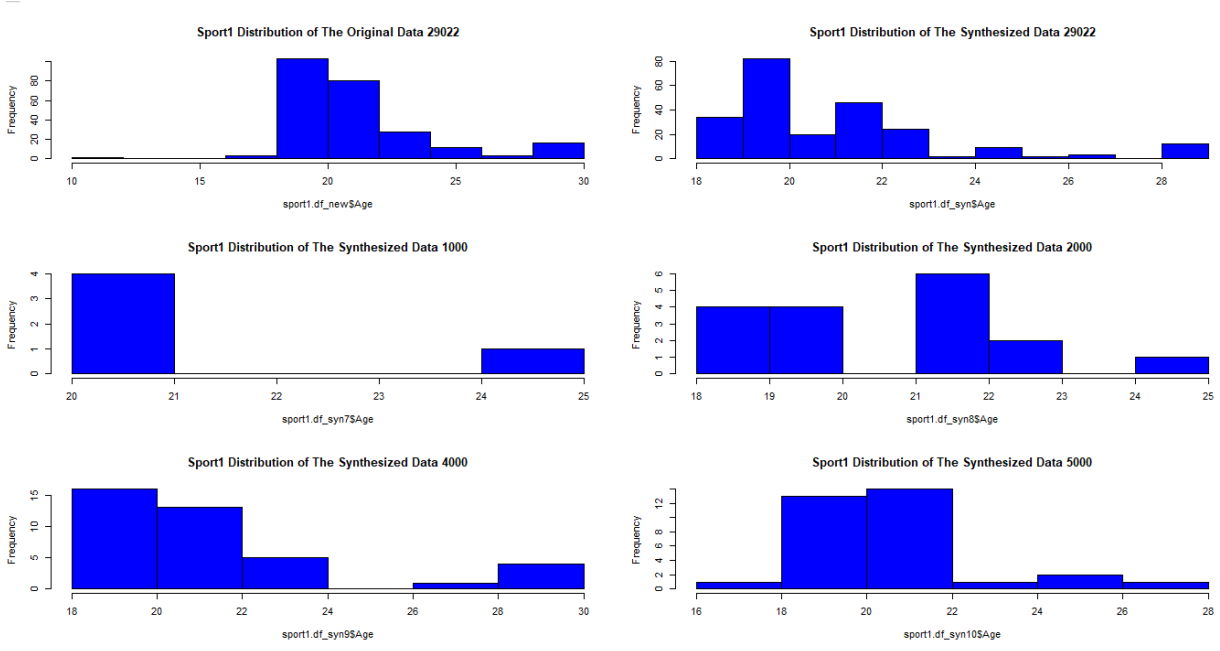


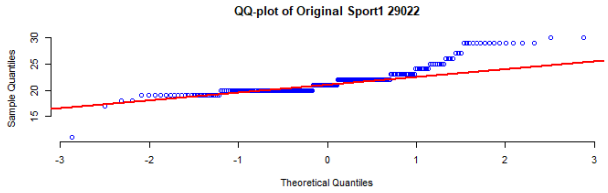
Furthermore, we checked the distribution of Variable Age for different categories of variable Sport with 22 categories. We noticed that the categories with small numbers will not appear in small dataset with 1000 observations. Figures xxx show the distribution for some categories.

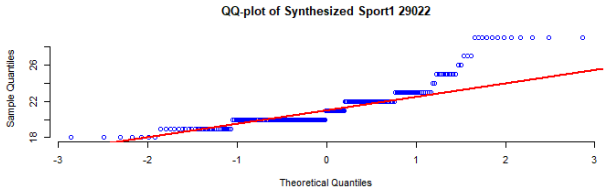
Dataset Sport	1	2	3	4	5	6	7	8	9	10	11

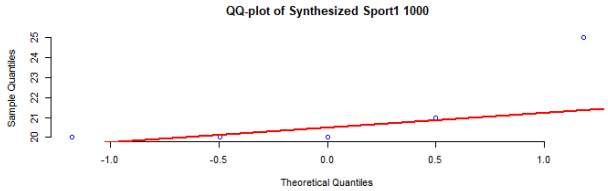
Original, n=29022	2466	407	16882	2434	2536	583	94	223	3420	656	1264
Synthesized, n= 29022	2347	387	16817	2466	2572	578	89	231	3444	659	1257
Synthesized, n= 1000	5	14	606	77	98	11	5	7	107	15	46
Synthesized, n= 2000	17	31	1178	166	169	22	11	18	234	42	90
Synthesized, n= 4000	39	48	2328	330	340	84	14	26	481	104	165
Synthesized, n= 5000	32	71	2914	423	450	101	9	34	564	100	241

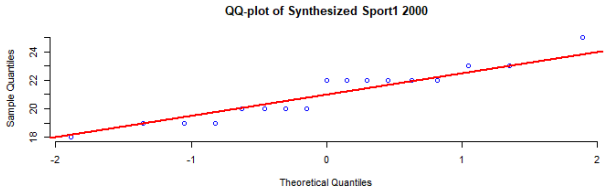
Dataset Sport	**12**	**13**	**14**	**15**	**16**	**17**	**19**	**20**	**21**	**22**	**23**
Original, n=29022	13	4	4	28	25	5	88	17	32	32	29
Synthesized, n= 29022	16	3	7	27	29	3	98	18	33	29	25
Synthesized, n= 1000				2	1		3			1	2
Synthesized, n= 2000	1			1	2		9	2	4	1	2
Synthesized, n= 4000				4	3	1	21	3	3	4	2
Synthesized, n= 5000	2	1		9	4	1	18	3	4	13	6

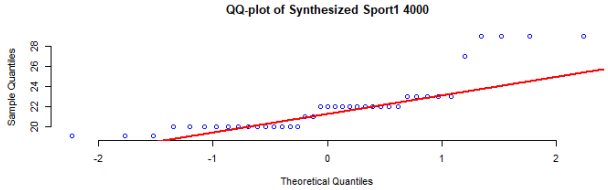


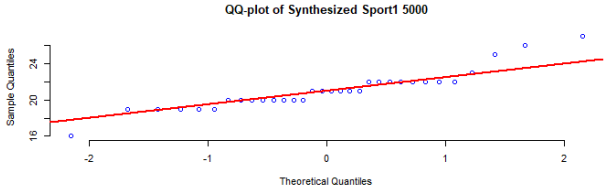

QQ-plot of Original Sport1 29022
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport1 29022
Sample Quantiles
Theoretical Quantiles

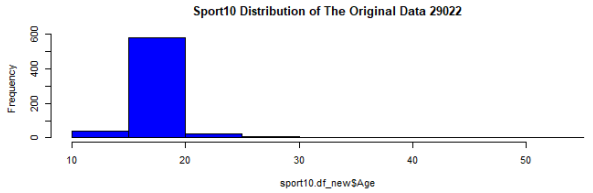

QQ-plot of Synthesized Sport1 1000
Sample Quantiles
Theoretical Quantiles

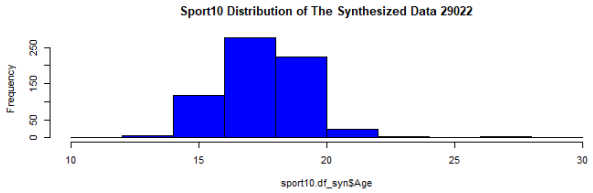

QQ-plot of Synthesized Sport1 2000
Sample Quantiles
Theoretical Quantiles

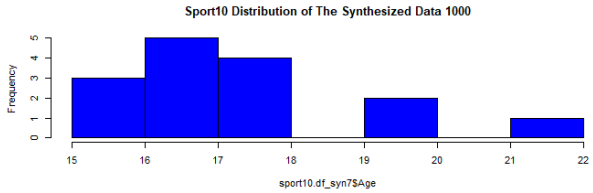

QQ-plot of Synthesized Sport1 4000
Sample Quantiles
Theoretical Quantiles

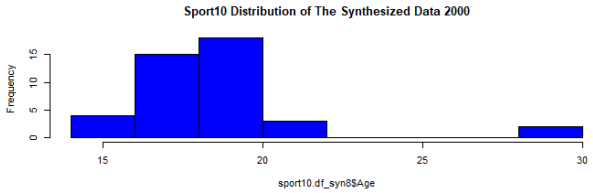

QQ-plot of Synthesized Sport1 5000
Sample Quantiles
Theoretical Quantiles

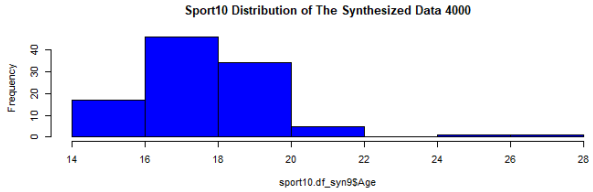
Sport 10

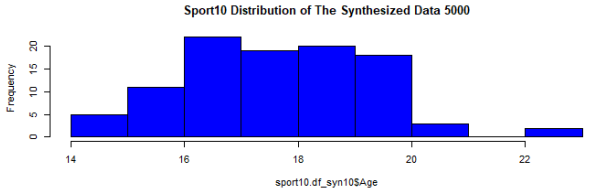

Sport10 Distribution of The Original Data 29022
Frequency
sport10.df_new$Age

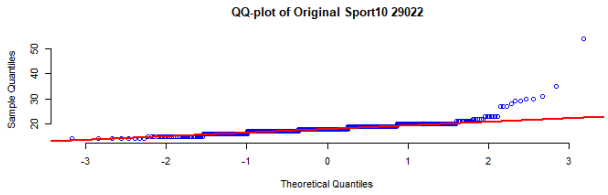

Sport10 Distribution of The Synthesized Data 29022
Frequency
sport10.df_syn$Age

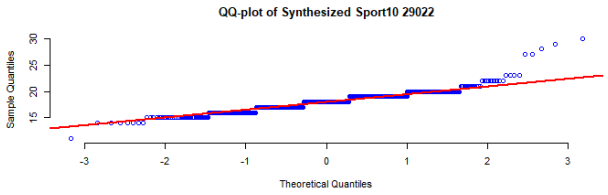

Sport10 Distribution of The Synthesized Data 1000
Frequency
sport10.df_syn7$Age

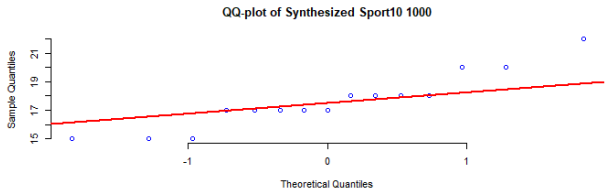

Sport10 Distribution of The Synthesized Data 2000
Frequency
sport10.df_syn8$Age

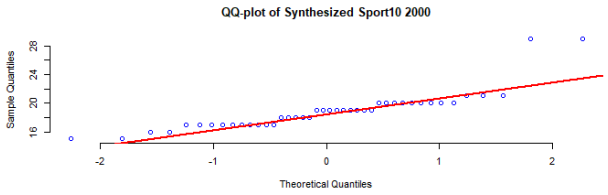

Sport10 Distribution of The Synthesized Data 4000
Frequency
sport10.df_syn9$Age

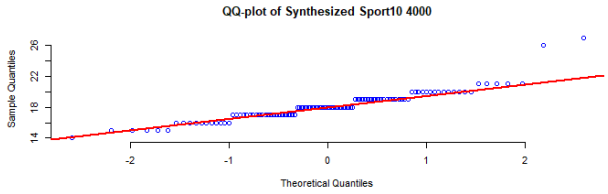

Sport10 Distribution of The Synthesized Data 5000
Frequency
sport10.df_syn10$Age

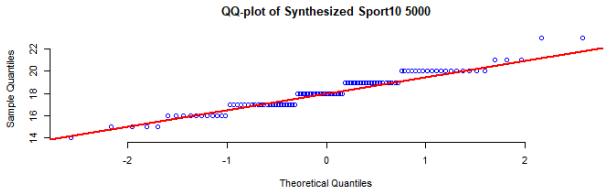

QQ-plot of Original Sport10 29022
Sample Quantiles
Theoretical Quantiles

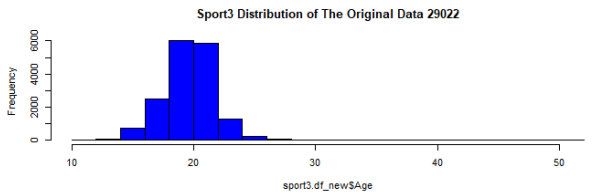

QQ-plot of Synthesized Sport10 29022
Sample Quantiles
Theoretical Quantiles

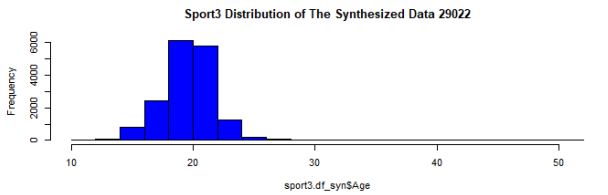

QQ-plot of Synthesized Sport10 1000
Sample Quantiles
Theoretical Quantiles

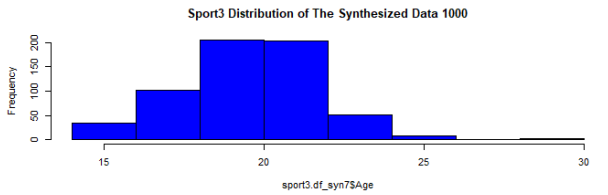

QQ-plot of Synthesized Sport10 2000
Sample Quantiles
Theoretical Quantiles

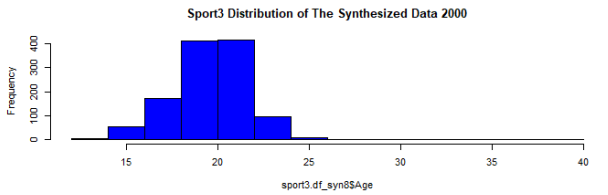

QQ-plot of Synthesized Sport10 4000
Sample Quantiles
Theoretical Quantiles

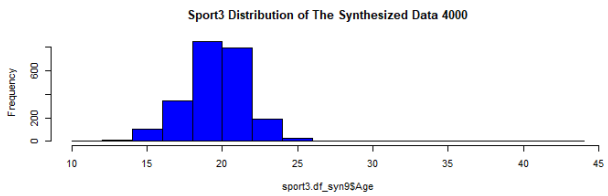

QQ-plot of Synthesized Sport10 5000
Sample Quantiles
Theoretical Quantiles

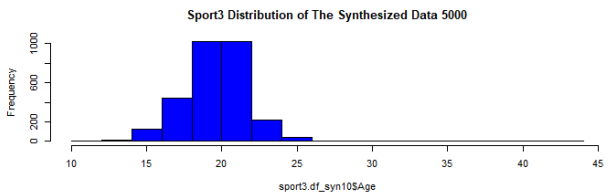

Sport3 Distribution of The Original Data 29022
Frequency
sport3.df_new$Age

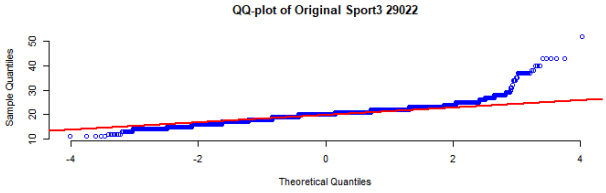

Sport3 Distribution of The Synthesized Data 29022
Frequency
sport3.df_syn$Age

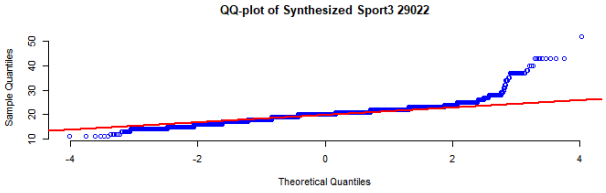

Sport3 Distribution of The Synthesized Data 1000
Frequency
sport3.df_syn7$Age

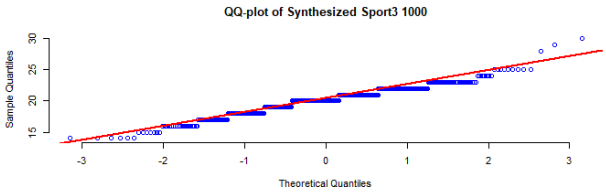

Sport3 Distribution of The Synthesized Data 2000
Frequency
sport3.df_syn8$Age

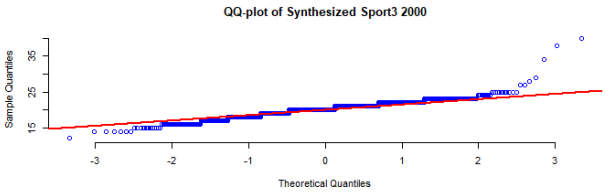

Sport3 Distribution of The Synthesized Data 4000
Frequency
sport3.df_syn9$Age

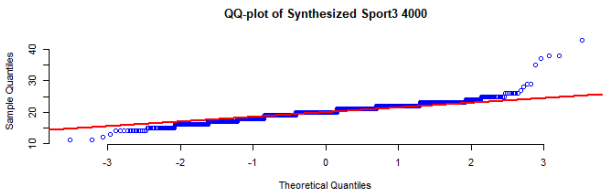

Sport3 Distribution of The Synthesized Data 5000
Frequency
sport3.df_syn10$Age

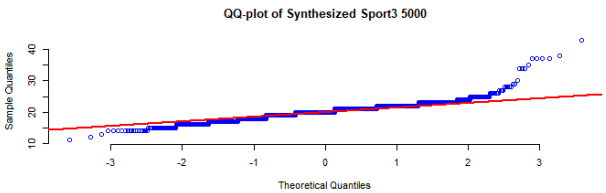

QQ-plot of Original Sport3 29022
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport3 29022
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport3 1000
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport3 2000
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport3 4000
Sample Quantiles
Theoretical Quantiles


QQ-plot of Synthesized Sport3 5000
Sample Quantiles
Theoretical Quantiles

Conclusion

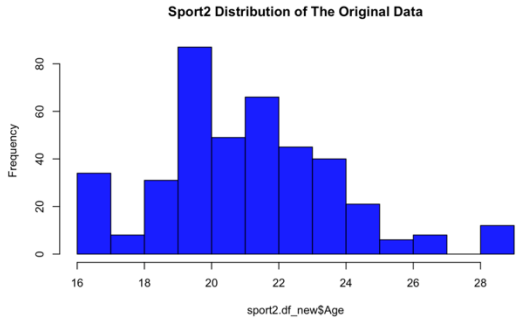
The synthetic dataset generated using the Synthpop package exhibits a similar statistical distribution to the original dataset. Through various tests and comparisons, we found that:

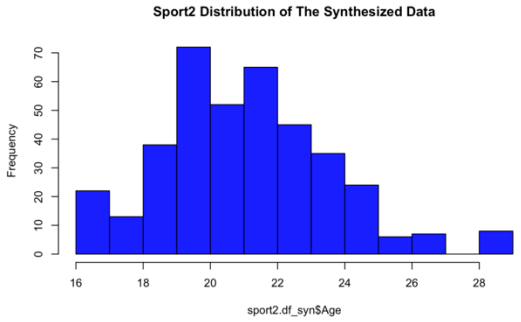
- The synthetic dataset retains the distribution characteristics for both categorical and numerical variables, as validated by the KS test.
- Logistic and linear regression models fitted on synthetic data provide results comparable to those obtained from the original data.
- The time required to generate larger synthetic datasets increases significantly, particularly with datasets exceeding 1,000,000 observations and numerous variables.
- The synthetic data effectively covers missing values and outliers, maintaining the integrity of the analysis.

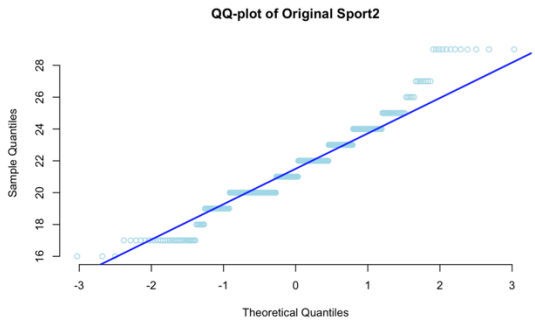
Overall, synthetic data proves to be a valuable tool for preserving data confidentiality while ensuring that statistical analyses remain accurate and reliable. However, addressing potential issues may require additional rules and access to the original dataset and its variable descriptions.

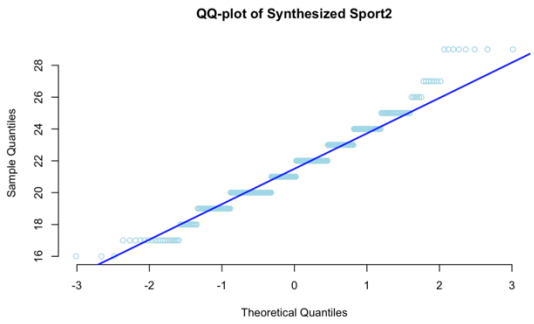
Appendix

- ## Sport 2

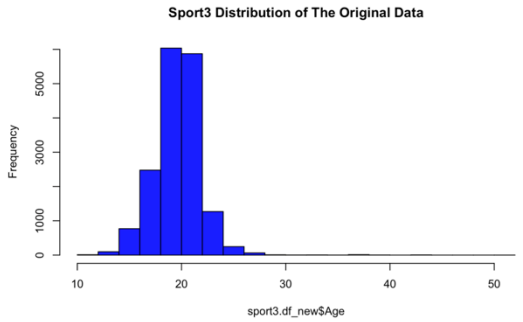


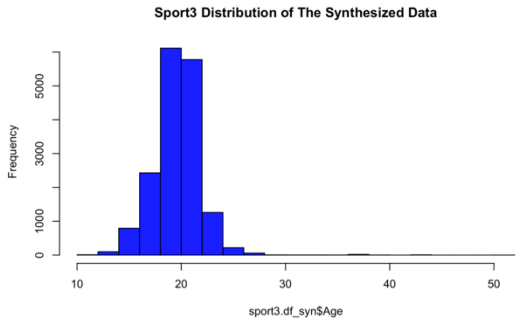


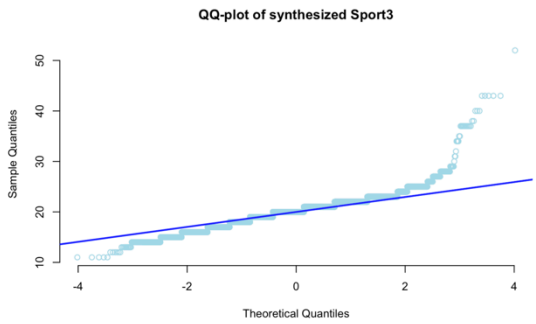


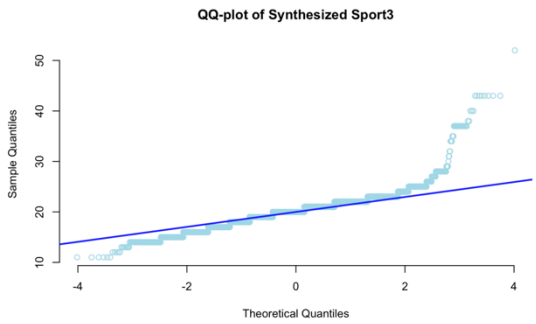


- ## Sport 3

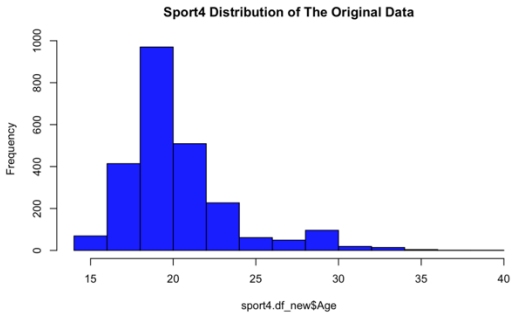


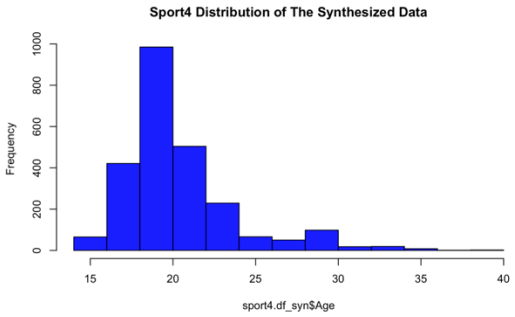


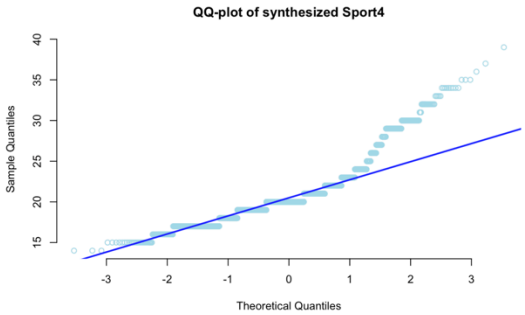


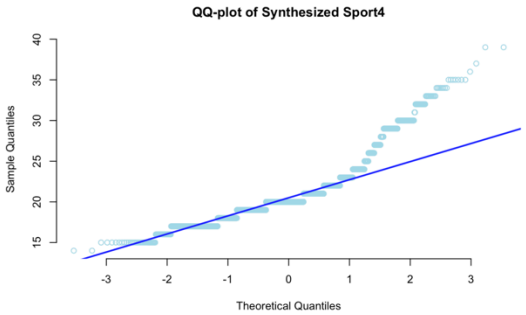


- **Sport 4**

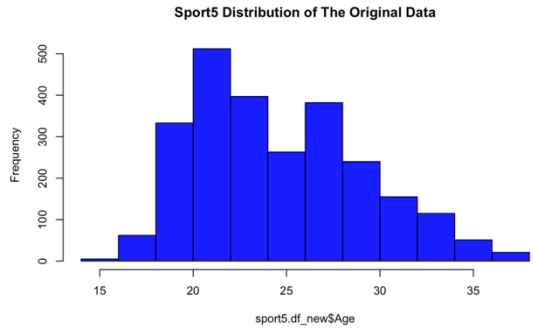


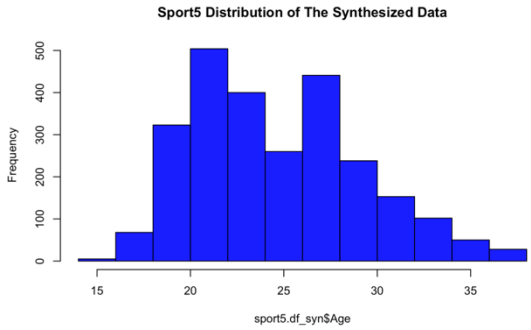


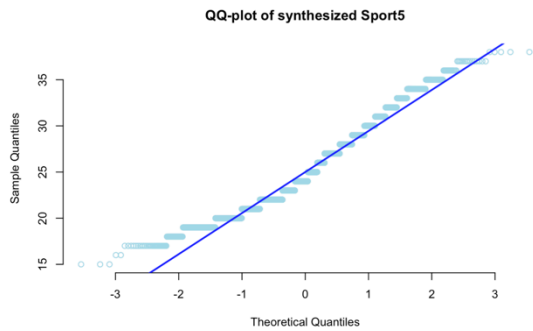


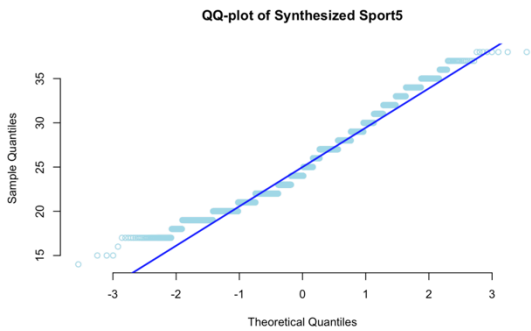


- **Sport 5**

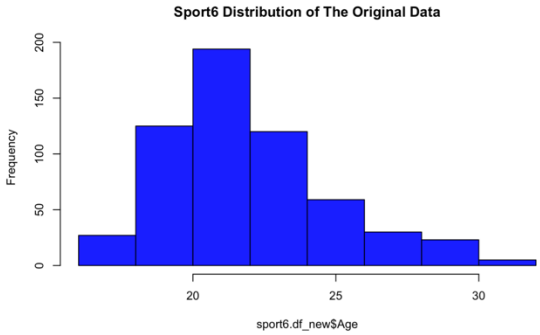


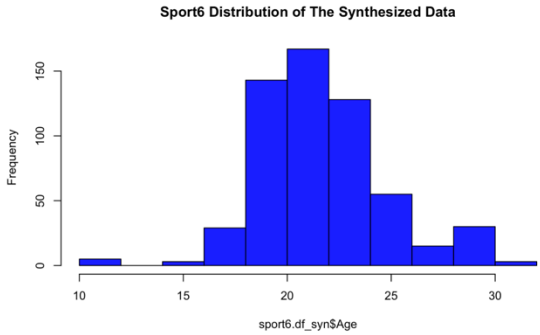


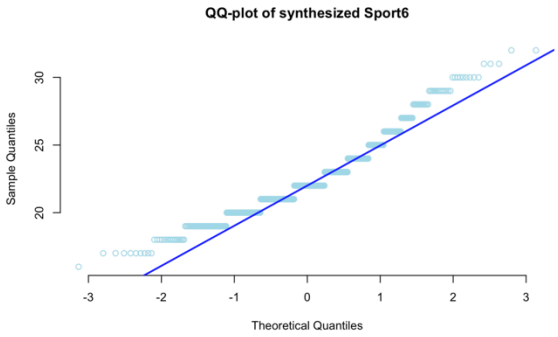


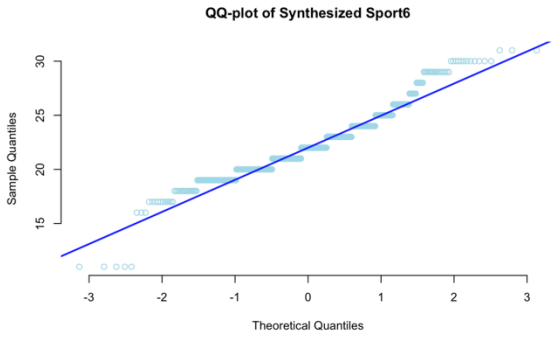


- **Sport 6**

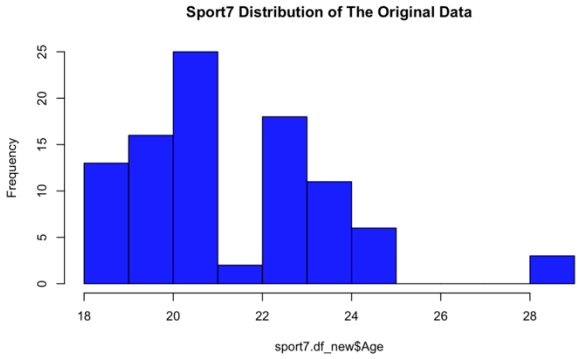


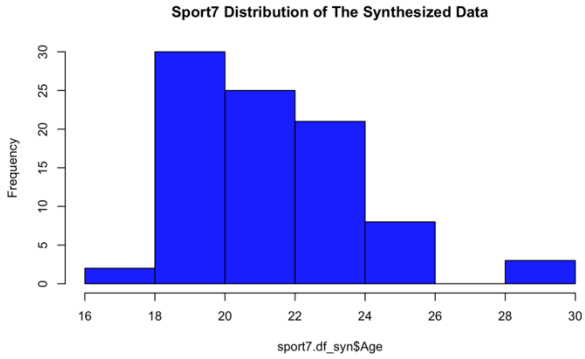


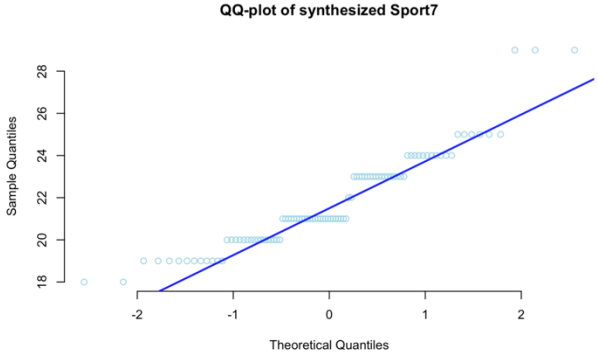


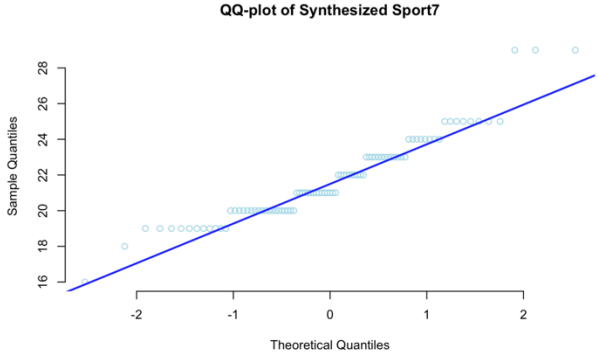


- **Sport 7**

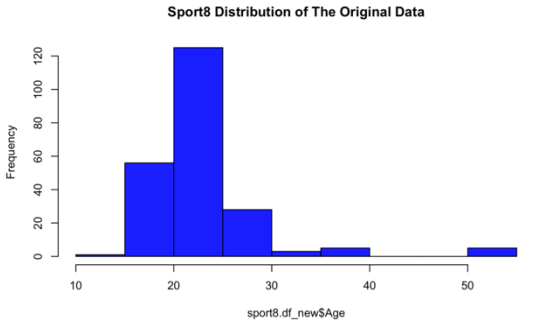


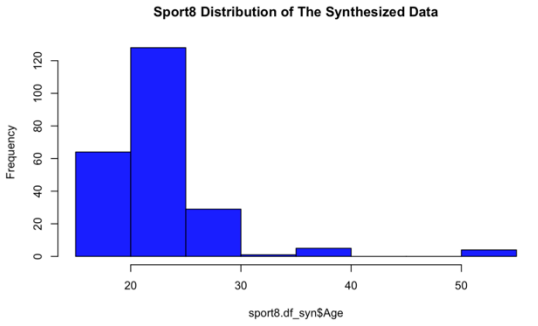


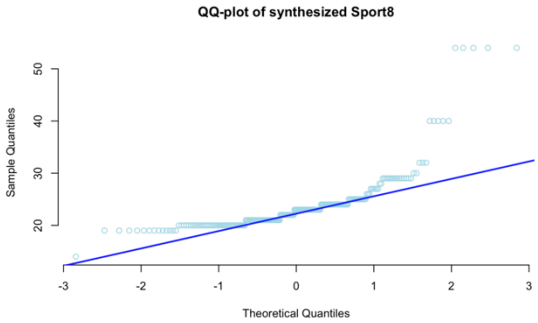


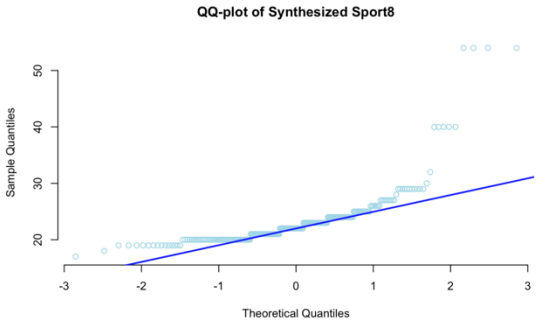


- **Sport 8**

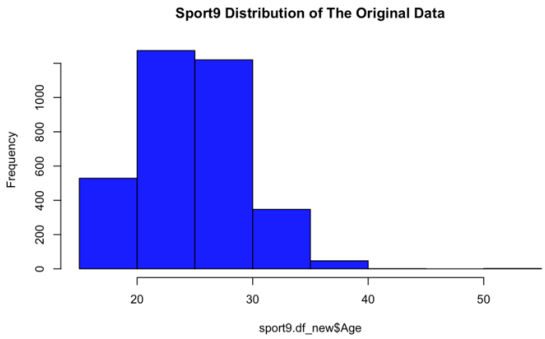


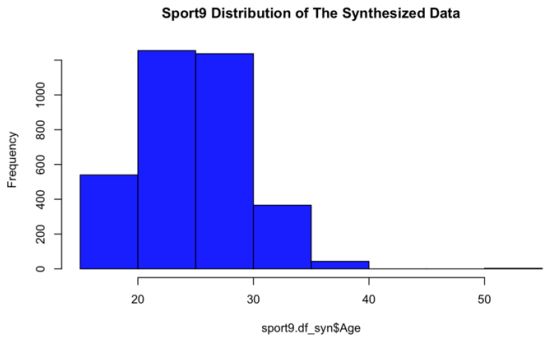


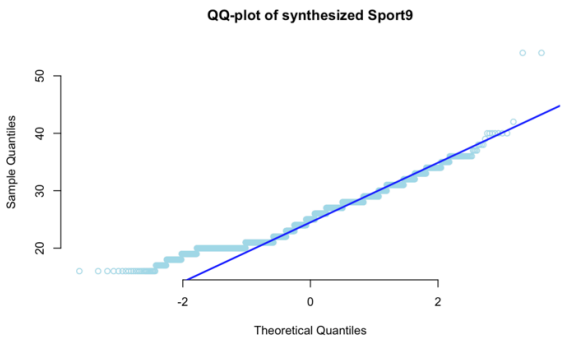


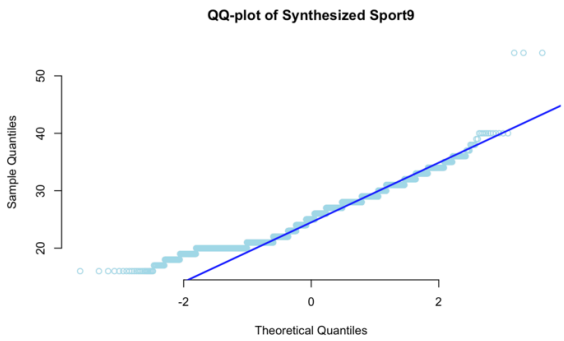


- **Sport 9**

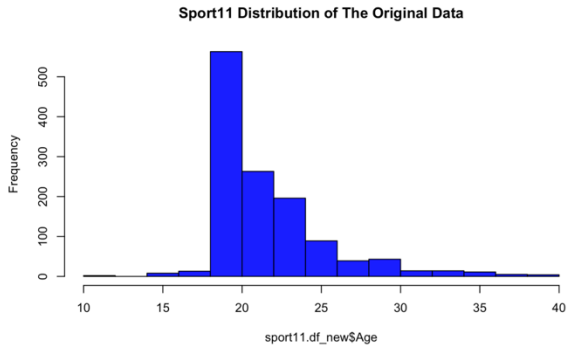


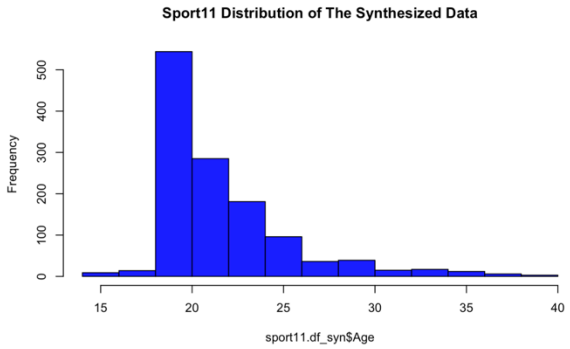


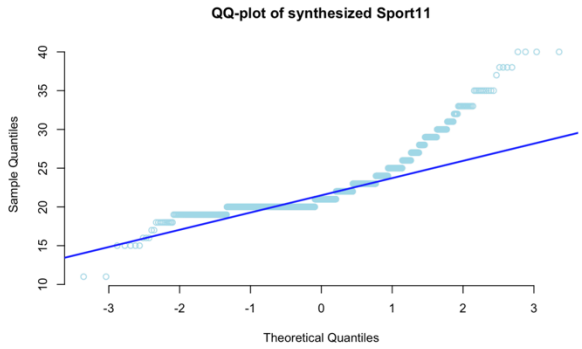


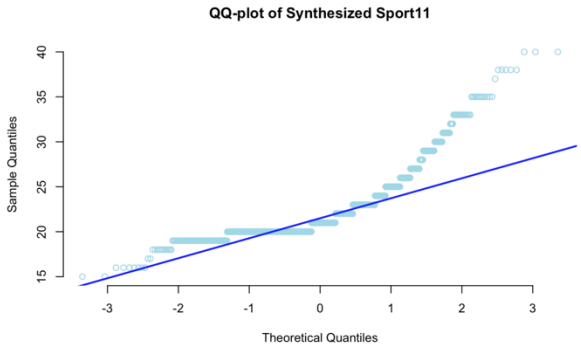


- **Sport 11**

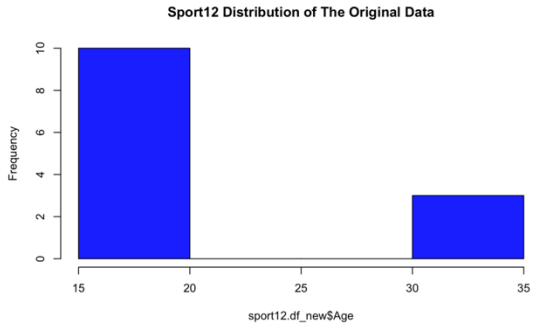


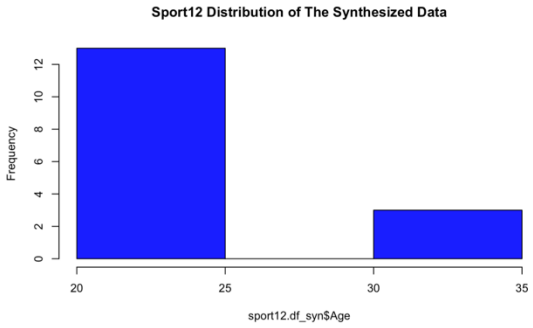


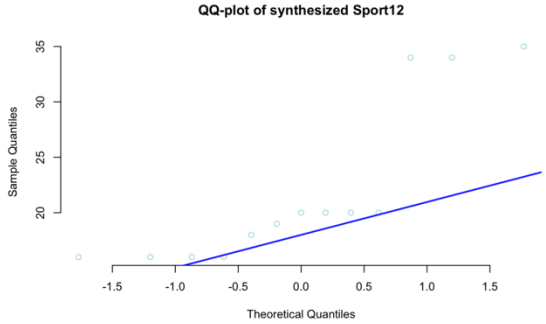


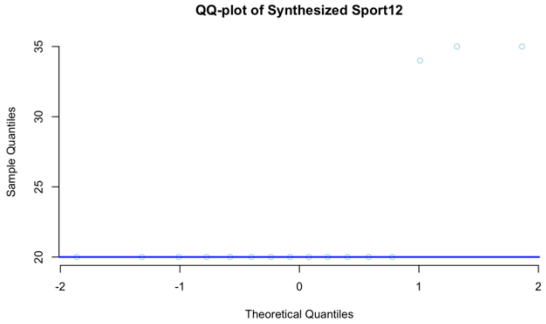


- **Sport 12**

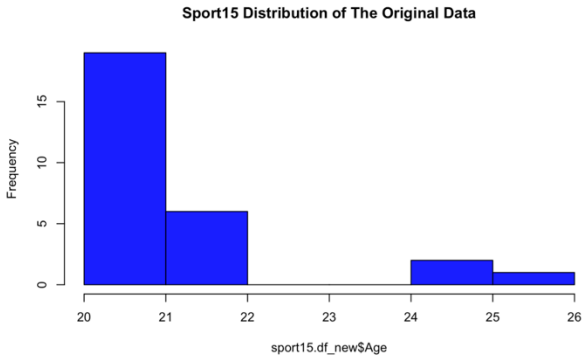


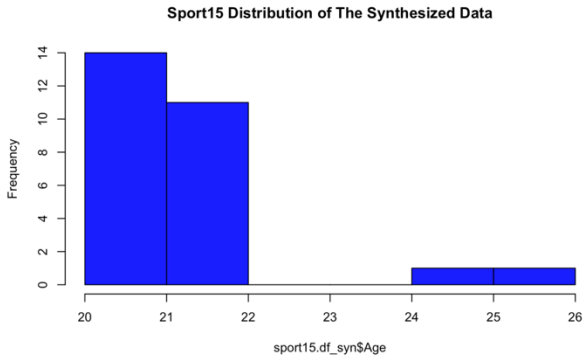


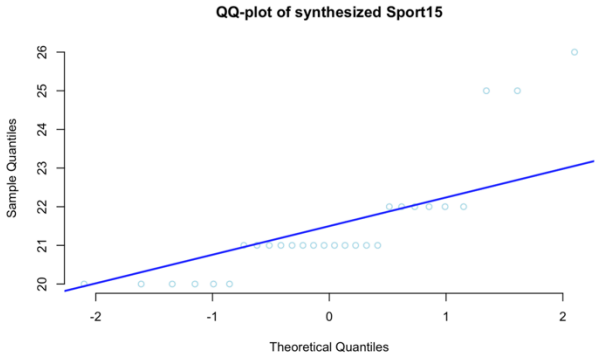


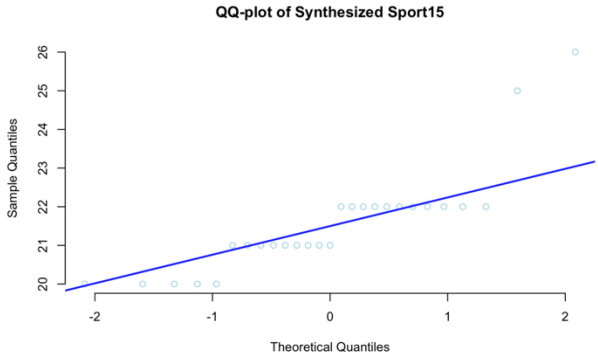


- **Sport 15**

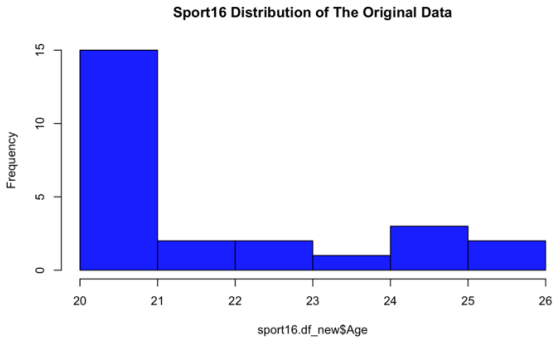


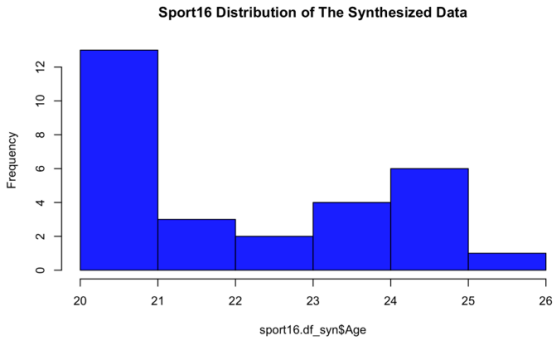


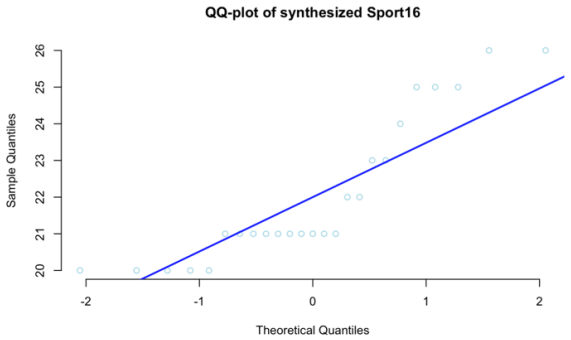


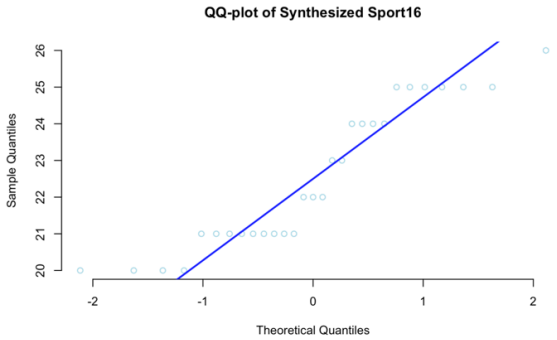


- **Sport 16**

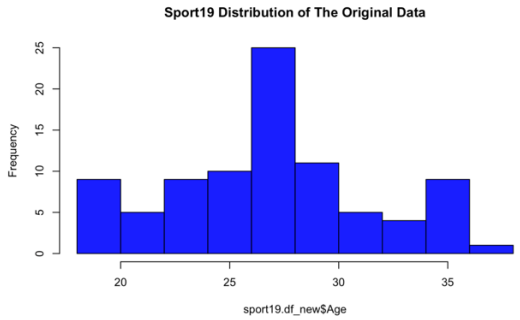


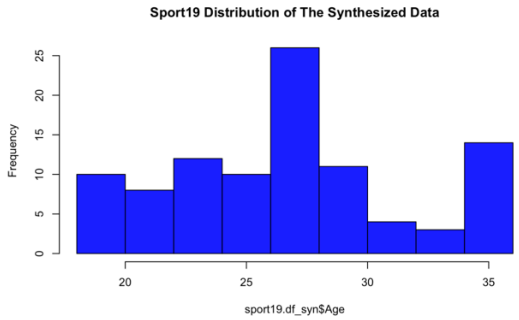


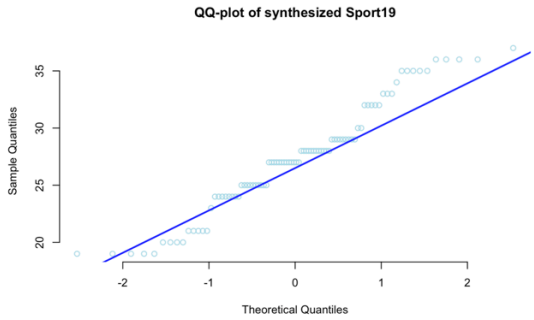


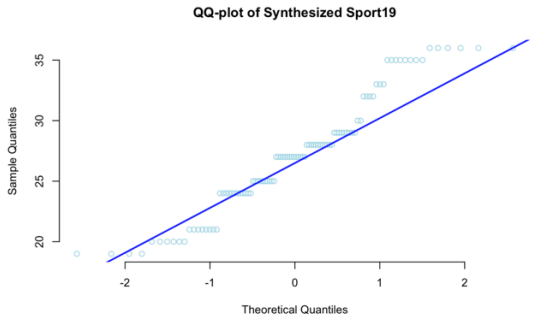


- **Sport 19**

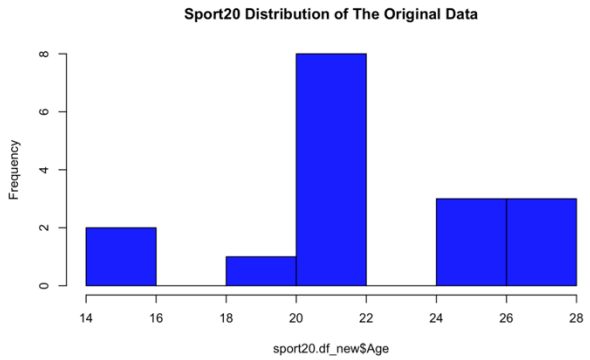


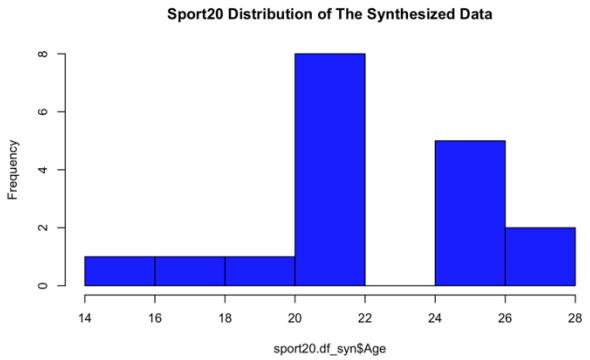


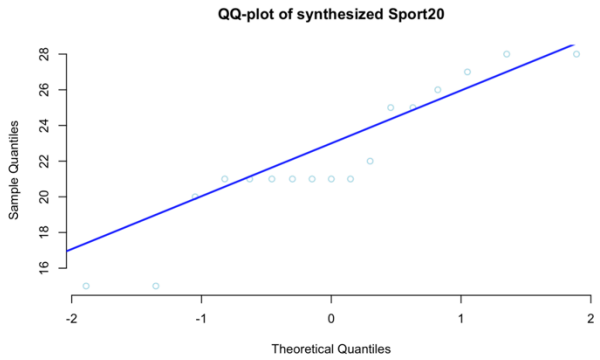


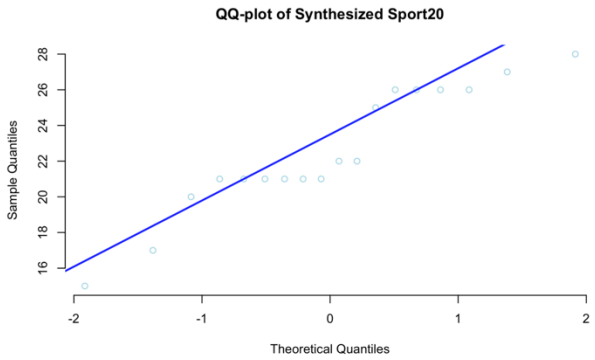


- **Sport 20**

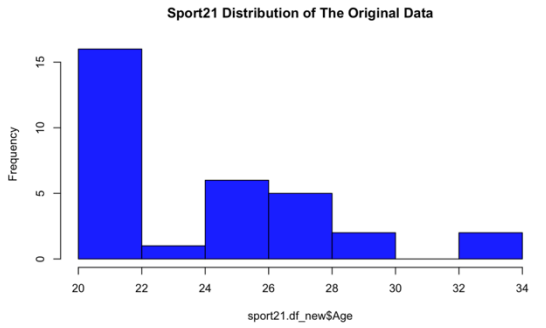


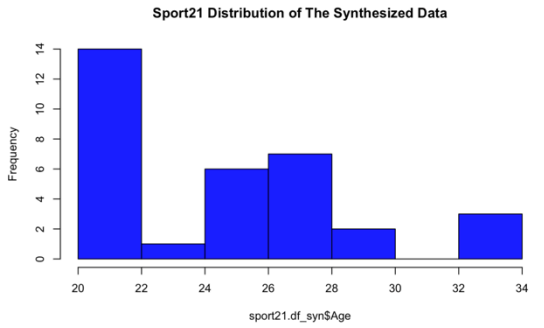


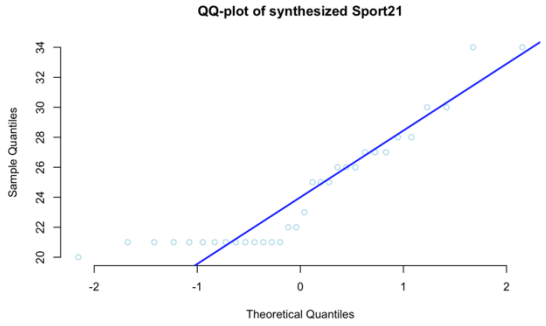


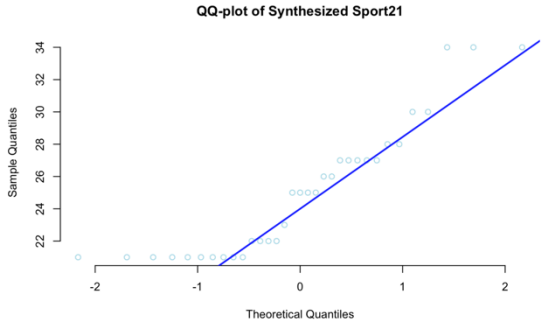


- **Sport 21**

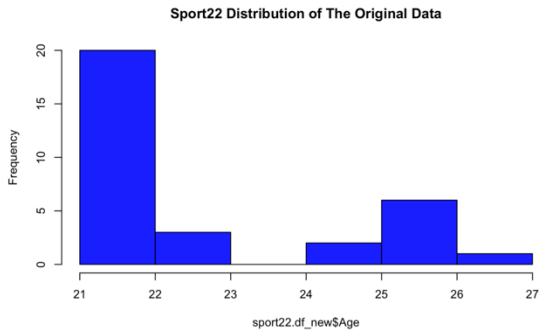


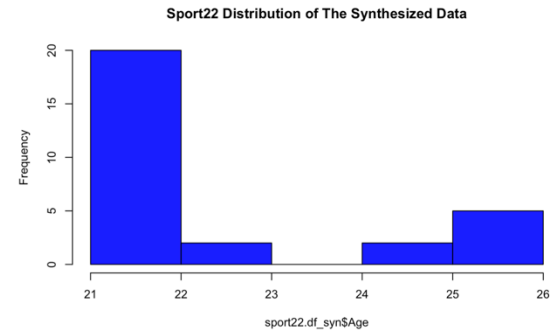


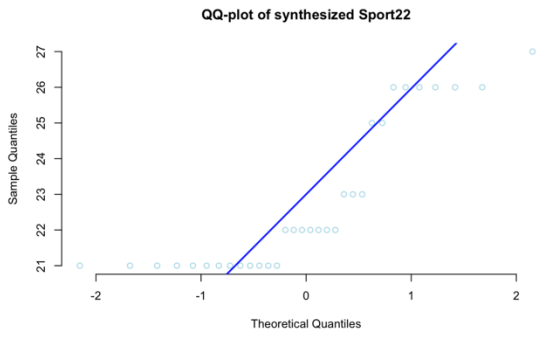


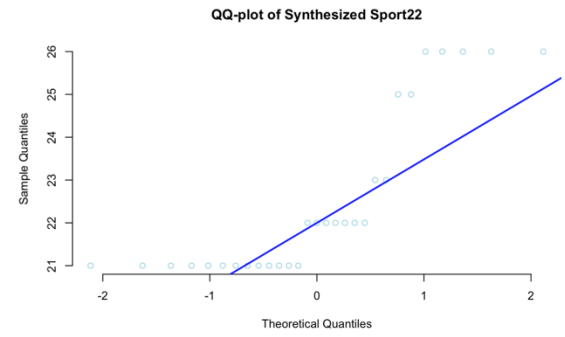


- **Sport 22**

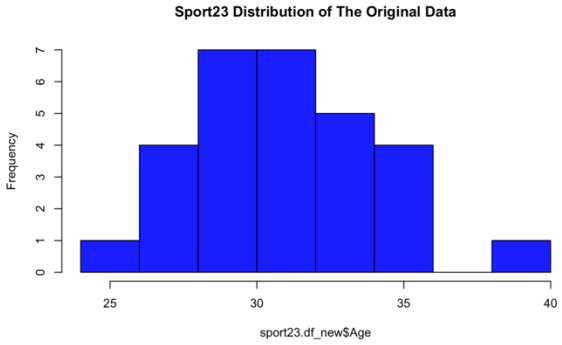


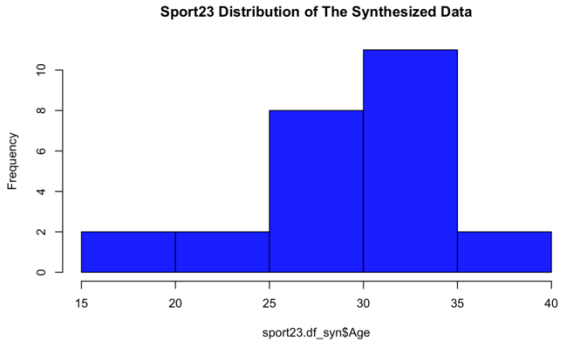


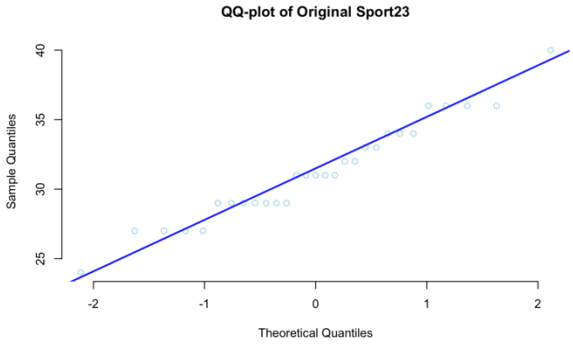


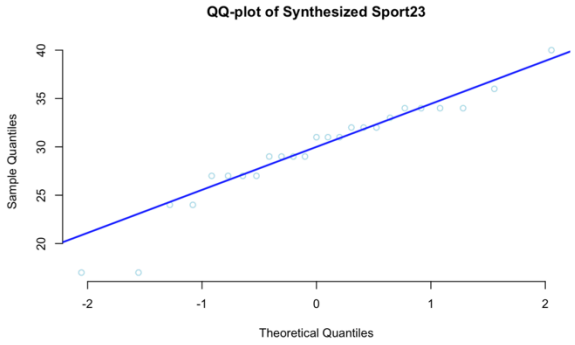


- **Sport 23**


Sport23 Distribution of The Original Data
Frequency
0
1
2
3
4
5
6
7
25
30
35
40
sport23.df_new$Age


Sport23 Distribution of The Synthesized Data
Frequency
0
2
4
6
8
10
15
20
25
30
35
40
sport23.df_syn$Age


QQ-plot of Original Sport23
Sample Quantiles
25
30
35
40
-2
-1
0
1
2
Theoretical Quantiles


QQ-plot of Synthesized Sport23
Sample Quantiles
20
25
30
35
40
-2
-1
0
1
2
Theoretical Quantiles