Synthetic Data

MATH 301 Data Confidentiality

Henrik Olsson

February 11, 2020

```
CEdata<- read.csv("CEdata.csv")
head(CEdata)
```

	UrbanRural <int>	Income <int>	Race <int>	Expenditure <dbl>
1	1	98600	1	5972.167
2	1	24360	1	5854.500
3	1	80200	1	5506.667
4	1	150500	1	8968.891
5	1	130000	1	10092.833
6	1	32836	1	5520.267

6 rows

1) Use your own synthesis model (different from the simple linear regression we covered in class) to synthesize m = 1 synthetic dataset for the CE sample.

Our goal is to generate synthetic data from the estimated Bayesian synthesizer from the posterior predictive distribution. To produce a good synthesizer, there will be trade-offs between utility and risks.

The most sensitive variable is Income, which is a continuous variable. If an intruder were to know one's income then they can obtain the person's information with much greater probability than if they had access to another variable. The total income is based on the past 12 months, which is a greater time span, and thus, a greater range than the Expenditure variable.

Instead of building a synthesis model of simple linear regression between Income and Expenditure, we can also create a hierachical model with UrbanRural, Race, or multiple linear regression.

```
CEdata$LogExp <- log(CEdata$Expenditure)
CEdata$LogIncome <- log(CEdata$Income)

## create indicator variable for Rural (2)
CEdata$Rural = fastDummies::dummy_cols(CEdata$UrbanRural)[,names(fastDummi
es::dummy_cols(CEdata$UrbanRural))
== ".data_1"]

## create indicator variables for Black (3), Native American (4),
## Asian (5), Pacific Islander (6), and Multi-race (7)
CEdata$Race_Black = fastDummies::dummy_cols(CEdata$Race)[,names(fastDummie
s::dummy_cols(CEdata$Race)) == ".data_2"]
CEdata$Race_NA = fastDummies::dummy_cols(CEdata$Race)[,names(fastDummies::
dummy_cols(CEdata$Race)) == ".data_3"]
CEdata$Race_Asian = fastDummies::dummy_cols(CEdata$Race)[,names(fastDummie
s::dummy_cols(CEdata$Race)) == ".data_4"]
CEdata$Race_PI = fastDummies::dummy_cols(CEdata$Race)[,names(fastDummies::
dummy_cols(CEdata$Race)) == ".data_5"]
CEdata$Race_M = fastDummies::dummy_cols(CEdata$Race)[,names(fastDummies::d
ummy_cols(CEdata$Race)) == ".data_6"]
```

```
## JAGS script
modelString <-"
model {
## sampling
for (i in 1:N){
y[i] ~ dnorm(beta0 + beta1*x_income[i] + beta2*x_rural[i] +
beta3*x_race_B[i] + beta4*x_race_N[i] +
beta5*x_race_A[i] + beta6*x_race_P[i] +
beta7*x_race_M[i], invsigma2)
}
## priors
beta0 ~ dnorm(mu0, g0)
beta1 ~ dnorm(mu1, g1)
beta2 ~ dnorm(mu2, g2)
beta3 ~ dnorm(mu3, g3)
beta4 ~ dnorm(mu4, g4)
beta5 ~ dnorm(mu5, g5)
beta6 ~ dnorm(mu6, g6)
beta7 ~ dnorm(mu7, g7)
invsigma2 ~ dgamma(a, b)
sigma <- sqrt(pow(invsigma2, -1))
}"
```

```r
y = as.vector(CEdata$LogExp)
x_income = as.vector(CEdata$LogIncome)
x_rural = as.vector(CEdata$Rural)
x_race_B = as.vector(CEdata$Race_Black)
x_race_N = as.vector(CEdata$Race_NA)
x_race_A = as.vector(CEdata$Race_Asian)
x_race_P = as.vector(CEdata$Race_PI)
x_race_M = as.vector(CEdata$Race_M)
N = length(y) # Compute the number of observations


## Pass the data and hyperparameter values to JAGS
the_data <- list("y" = y, "x_income" = x_income,
"x_rural" = x_rural, "x_race_B" = x_race_B,
"x_race_N" = x_race_N, "x_race_A" = x_race_A,
"x_race_P" = x_race_P, "x_race_M" = x_race_M,
"N" = N,
"mu0" = 0, "g0" = 1, "mu1" = 0, "g1" = 1,
"mu2" = 0, "g2" = 1, "mu3" = 0, "g3" = 1,
"mu4" = 0, "g4" = 1, "mu5" = 0, "g5" = 1,
"mu6" = 0, "g6" = 1, "mu7" = 0, "g7" = 1,
"a" = 1, "b" = 1)
```

```r
initsfunction <- function(chain){
.RNG.seed <- c(1,2)[chain]
.RNG.name <- c("base::Super-Duper",
"base::Wichmann-Hill")[chain]
return(list(.RNG.seed=.RNG.seed,
.RNG.name=.RNG.name))
}
```

```r
## Run the JAGS code for this model:
posterior_MLR <- run.jags(modelString,
n.chains = 1,
data = the_data,
monitor = c("beta0", "beta1", "beta2",
"beta3", "beta4", "beta5",
"beta6", "beta7", "sigma"),
adapt = 1000,
burnin = 5000,
sample = 5000,
thin = 1,
inits = initsfunction)
```

```
## Loading required namespace: rjags
```

```
## Compiling rjags model...
## Calling the simulation using the rjags method...
## Note: the model did not require adaptation
## Burning in the model for 5000 iterations...
## Running the model for 5000 iterations...
## Simulation complete
## Calculating summary statistics...
```

```
## Warning: Convergence cannot be assessed with only 1 chain
```

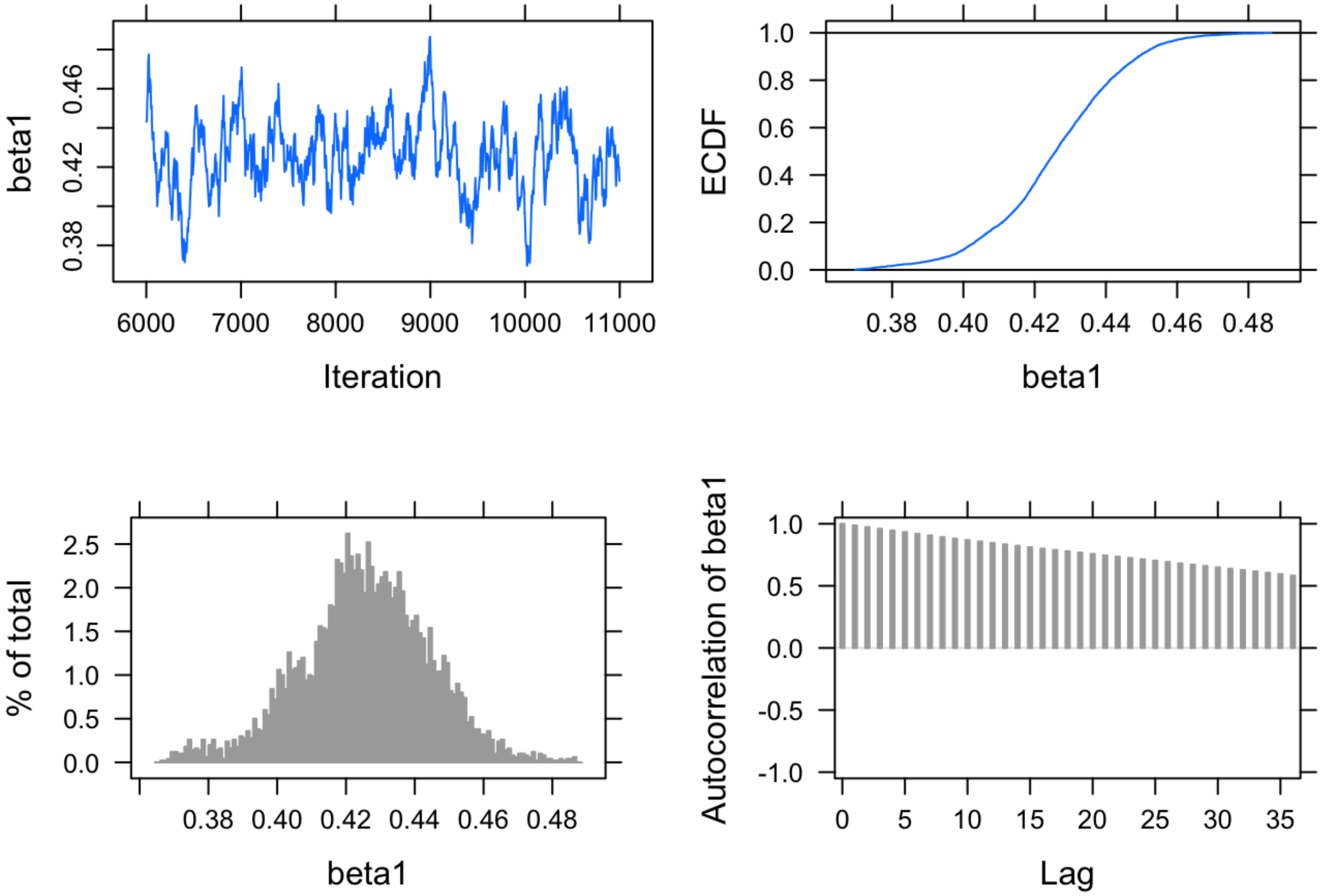
```
## Finished running the simulation
```

```
## JAGS output
summary(posterior_MLR)
```

```
##           Lower95      Median     Upper95        Mean         SD Mode
## beta0  3.54580441  4.00080795  4.47511996  4.02342538 0.22801936   NA
## beta1  0.38724270  0.42589245  0.46270522  0.42560569 0.01888028   NA
## beta2  0.07556203  0.27567761  0.49356550  0.27739517 0.10683751   NA
## beta3 -0.33286463 -0.19636237 -0.05011323 -0.19589145 0.07350734   NA
## beta4 -0.49856355  0.01200176  0.52491192  0.01108006 0.26200777   NA
## beta5 -0.07838912  0.15751196  0.38365788  0.15652442 0.11925047   NA
## beta6 -0.47113608  0.08692820  0.60972710  0.08885212 0.28043794   NA
## beta7 -0.31549244  0.04217888  0.37819450  0.04125956 0.17844949   NA
## sigma  0.69161484  0.72115468  0.75539675  0.72161423 0.01621386   NA
##              MCerr MC%ofSD SSeff       AC.10 psrf
## beta0 0.0438707646    19.2    27  0.894522661   NA
## beta1 0.0032115062    17.0    35  0.868794694   NA
## beta2 0.0093583674     8.8   130  0.595800945   NA
## beta3 0.0012361528     1.7  3536 -0.000743349   NA
## beta4 0.0037053495     1.4  5000 -0.008922402   NA
## beta5 0.0017885674     1.5  4445 -0.018979087   NA
## beta6 0.0039659914     1.4  5000  0.007764935   NA
## beta7 0.0026293845     1.5  4606  0.017692787   NA
## sigma 0.0002292986     1.4  5000  0.010989815   NA
```

```
plot(posterior_MLR, vars = "beta1")
```

```
## Generating plots...
```



```
## Saving posterior parameter draws
post <- as.mcmc(posterior_MLR)

## Generating one set of sythetic data
synthesize <- function(X, index, n){
  mean_Y <- post[index, "beta0"] +  X$x_income * post[index, "beta1"] +  X
$x_rural * post[index, "beta2"] +  X$x_race_B * post[index, "beta3"] +  X
$x_race_N * post[index, "beta4"] +  X$x_race_A * post[index, "beta5"] +  X
$x_race_P * post[index, "beta6"] +  X$x_race_M * post[index, "beta7"]
  synthetic_Y <- rnorm(n, mean_Y, post[index, "sigma"])
  data.frame(X$x_income, synthetic_Y)
}
n <- dim(CEdata)[1]
new <- data.frame(x_income, x_rural, x_race_B, x_race_N, x_race_A, x_race_
P, x_race_M)
synthetic_one <- synthesize(new, 1, n)
names(synthetic_one) <- c("OrigLogIncome", "SynLogIncome")
synthetic_one
```

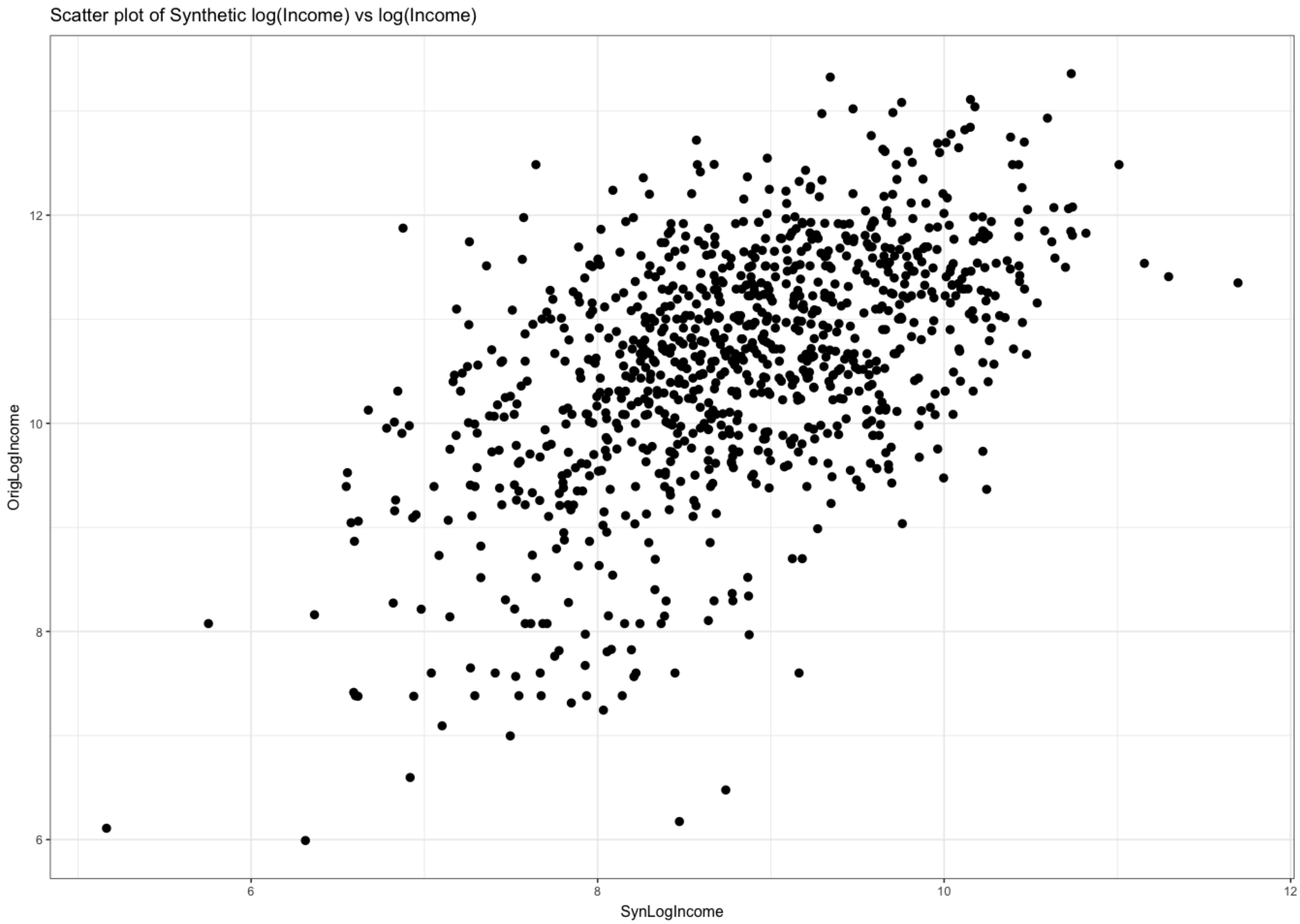
OrigLogIncome <dbl>	SynLogIncome <dbl>
11.498827	8.872622
10.100698	8.046865
11.292279	10.096460
11.921718	9.311092
11.775290	9.063257
10.399281	10.254284
7.414573	6.592130
11.624538	8.890004
8.732950	7.623198
11.571194	8.002598

1-10 of 994 rows Previous **1** 2 3 4 5 6 ... 100 Next

I preserved relationships by having inferences done on synthetic data that are “close” to those done on confidential data. I attempted to preseve the relationships between Income and Expenditure, UrbanRural, Race using Multiple Linear Regression.

2) Make a scatter plot of the synthesized log(Income) against the original log(Income), and see what you find.

```
ggplot(synthetic_one, aes(x = SynLogIncome, y = OrigLogIncome)) +
  geom_point(size = 1) +
  labs(title = "Scatter plot of Synthetic log(Income) vs log(Income)") +
  theme_bw(base_size = 6, base_family = "")
```



From the scatter plot of the sythesized log(Income) against the original log(income), we see that there is a positive linear relationship.

3) Compare the mean and median of log(Income), in the original dataset and the confidential dataset. Are they close to each other?

```
##synthesized log(Income)
mean(synthetic_one$SynLogIncome)
```

```
## [1] 8.824479
```

```
median(synthetic_one$SynLogIncome)
```

```
## [1] 8.831394
```

```
##original log(Income)
mean(synthetic_one$OrigLogIncome)
```

```
## [1] 10.59507
```

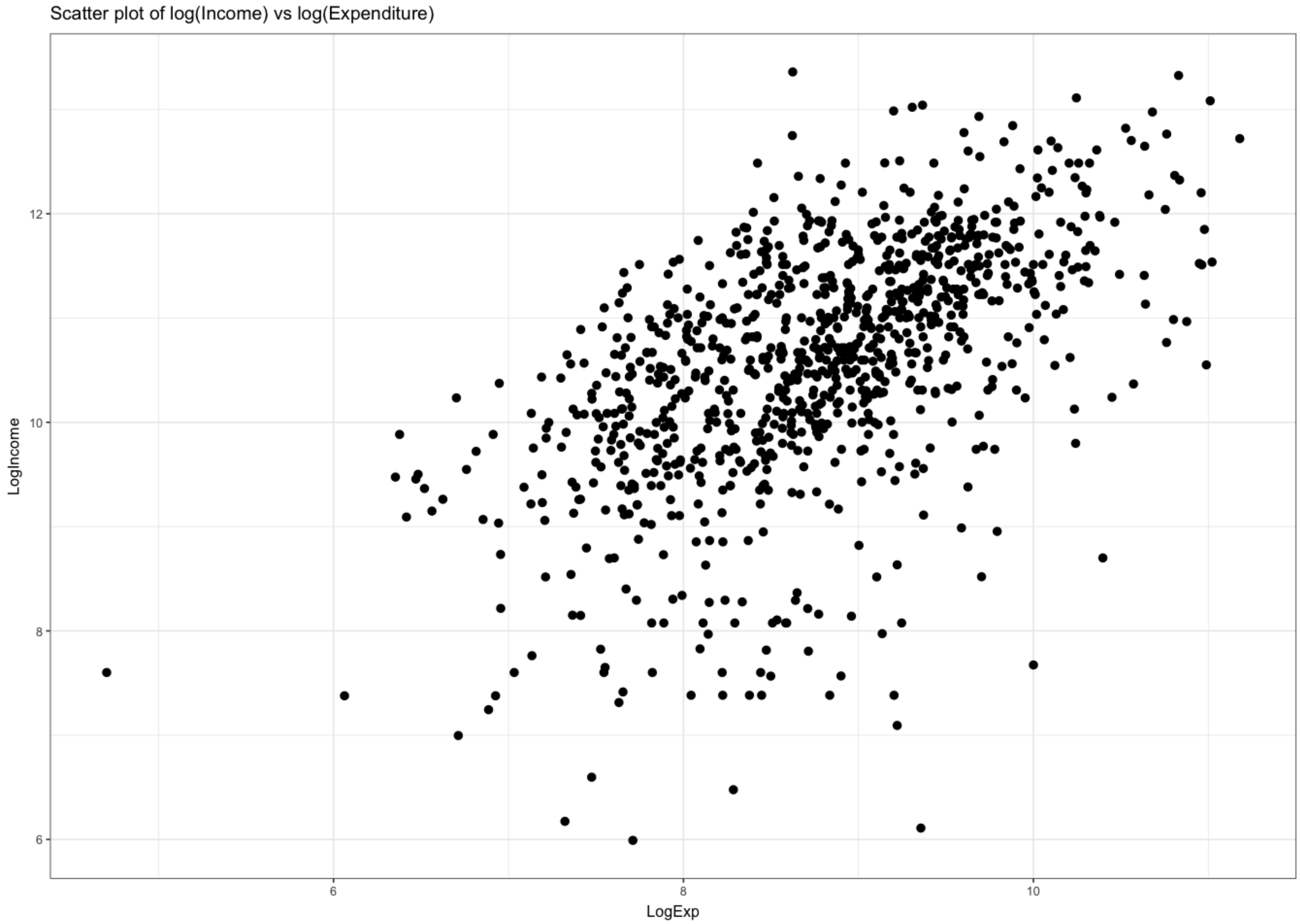
```
median(synthetic_one$OrigLogIncome)
```

```
## [1] 10.70574
```

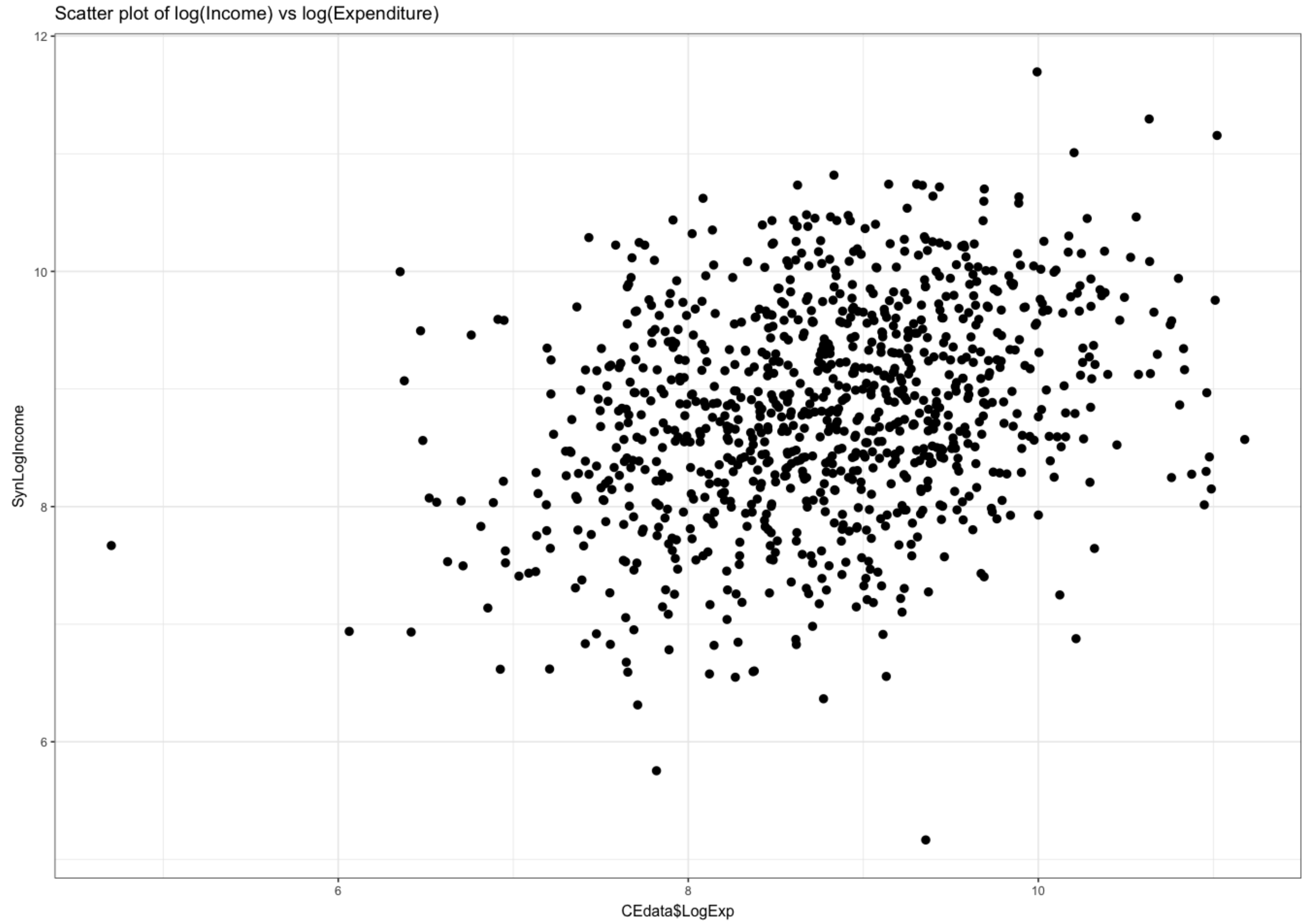
The mean and median of the synthesized log(Income) is approximately 2 units below the mean and median of the original log(Income), respectively.

4) Compare the point estimate of the regression coefficients of log(Income) on log(Expenditure), in the original dataset and the confidential dataset. Are they close to each other?

```
ggplot(CEdata, aes(x = LogExp, y = LogIncome)) +
  geom_point(size = 1) +
  labs(title = "Scatter plot of log(Income) vs log(Expenditure)") +
  theme_bw(base_size = 6, base_family = "")
```




```
ggplot(synthetic_one, aes(x = CEdata$LogExp, y = SynLogIncome)) +
  geom_point(size = 1) +
  labs(title = "Scatter plot of log(Income) vs log(Expenditure)") +
  theme_bw(base_size = 6, base_family = "")
```



The point estimate of the regression coeffecients between the two graphs are very close.