Project for Summer Short Course on Data Privacy

Summer 2025

Project Title

Generation of Synthetic Adult Dataset Using Synthpop and the DA-MI Framework

Project Description

This project focuses on synthesizing demographic quasi-identifiers from the UCI Adult Census Income dataset—one of the most commonly used benchmarks in data privacy research. The dataset includes sensitive personal attributes such as age, sex, race, and education level, which are commonly treated as quasi-identifying variables.

The first part of the project involves using the `synthpop` package in R to generate synthetic versions of these demographic variables, based on their statistical relationships with other available covariates (e.g., hours worked per week, occupation, income class). The `synthpop` framework provides flexible model-based synthesis techniques and can be used to fit the association model that estimates $p(X \mid Z)$, where X is a demographic variable and Z includes available non-sensitive covariates.

The second part of the project extends this synthesis process using the Data-Augmented Multiple Imputation (DA-MI) framework (reference paper here). In DA-MI, synthetic values are drawn from the posterior predictive distribution:

$$p(\tilde{X} \mid Z, W) \propto p(\tilde{X} \mid Z; \theta) \cdot p(W \mid \tilde{X}; \lambda),$$

where:

- The association model $p(X \mid Z; \theta)$ specifies a potentially incorrect relationship between X and Z.
- The masking model $p(W \mid X; \lambda)$ encodes a tunable perturbation from X to a pseudo-variable W and this model is always correctly specified.

By combining both models, synthetic values of X are generated in a joint modeling framework as described in Jiang et al. (2022).

Tasks

1. Fit the association model $p(X \mid Z)$ and generate synthetic values for X using the `synthpop` package.

2. (Optional if time permits) Construct pseudo-variables W and generate synthetic values for X using the DA-MI approach.

    - Add Gaussian noise to continuous X (e.g., age) and use a mixture model for categorical X (e.g., race, education).
    - Fit the masking model for $p(W \mid X; \lambda)$ (e.g., by occupation or income class) and the association model for $p(\tilde{X} \mid Z; \theta)$.
    - Generate synthetic X from the DA-MI posterior predictive distribution using both $p(X \mid Z)$ and $p(W \mid X)$.

For both approaches, evaluate synthetic data using utility metrics (e.g., CI overlap, coefficient bias) in a specific regression task and privacy metrics (e.g., re-identification risk).

Sample R code

```
# Install and load required packages
install.packages("synthpop")
install.packages("dplyr")
install.packages("ggplot2")

library(synthpop)
library(dplyr)
library(ggplot2)

# Load dataset
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/adult/adult.data"
col_names <- c("age", "workclass", "fnlwgt", "education", "education_num", "marital_stat
               "occupation", "relationship", "race", "sex", "capital_gain", "capital_los
               "hours_per_week", "native_country", "income")
adult <- read.csv(url, header = FALSE, col.names = col_names, strip.white = TRUE)

# Convert to factors
adult <- adult %>%
  mutate(across(c(sex, race, education, income), as.factor))

# Define variables
vars_to_synthesize <- c("sex", "race", "education", "income")
vars_to_keep <- c("age", "hours_per_week", "capital_gain", "capital_loss")
```

```
# Full set of variables
all_vars <- c(vars_to_keep, vars_to_synthesize)
adult_subset <- adult[, all_vars]

# Specify synthesis methods per variable (optional)
method_vec <- c("","", "", "", "")  # default is CART
names(method_vec) <- all_vars
method_vec["education"] <- "polyreg"
method_vec["income"] <- "logreg"  # logistic regression
method_vec["sex"] <- "cart"
method_vec["race"] <- "cart"

# Specify predictor matrix: only allow synthesis of demo vars conditional on the fixed v
pm <- make.predictorMatrix(adult_subset)
pm[,] <- 0  # start from zero
for (v in vars_to_synthesize) {
  pm[v, vars_to_keep] <- 1  # each demo var predicted by fixed covariates
}

# Optional: if you want synthesized demo vars to condition on other demo vars in sequenc
visit_order <- c("sex", "race", "education", "income")  # sequential synthesis
pm["race", "sex"] <- 1
pm["education", c("sex", "race")] <- 1
pm["income", c("sex", "race", "education")] <- 1

# Run synthesis
syn_result <- syn(adult_subset,
                  method = method_vec,
                  predictor.matrix = pm,
                  visit.sequence = visit_order,
                  m = 1)

# Replace synthesized columns in original dataset
adult_synth <- adult
adult_synth[, vars_to_synthesize] <- syn_result$syn[, vars_to_synthesize]

# Check how well synthesis preserved relationships
model_formula <- income ~ age + education + hours_per_week + sex + race + capital_gain +

fit_real <- glm(model_formula, data = adult, family = binomial)
fit_synth <- glm(model_formula, data = adult_synth, family = binomial)

summary(fit_real)
summary(fit_synth)
```

```
# Compare coefficients
coef_df <- data.frame(
  Variable = names(coef(fit_real)),
  Real = coef(fit_real),
  Synth = coef(fit_synth)
)

print(coef_df)

# Plot coefficient comparison
ggplot(coef_df[-1, ], aes(x = Variable)) +
  geom_point(aes(y = Real, color = "Real")) +
  geom_point(aes(y = Synth, color = "Synthetic")) +
  ylab("Coefficient Estimate") +
  ggtitle("Comparison of Coefficients: Original vs Partially Synthetic") +
  theme_minimal() +
  scale_color_manual(values = c("Real" = "blue", "Synthetic" = "red"))
```