Thesis Simulation Data Generation Document for Chapter 4

Dasha Asienga

2024-03-14

Contents

This file is intended to try out and test different data generation mechanisms.

We're interested in creating a data set that has 50-50 class balance, even across the demographic group, and also has better predictive performance than the COMPAS tool. For this set-up, we will only use 2 variables from the COMPAS data set: 1 continuous variable and 1 categorical variable.

Reading in the Data

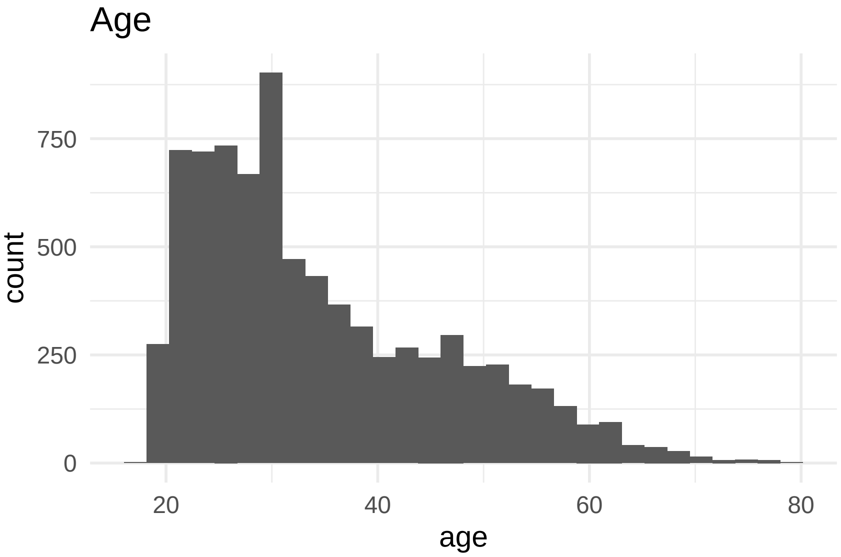
First, let's read in the data.

```
compas_path <- "/home/dasienga24/Statistics-Senior-Honors-Thesis/Data Sets/COMPAS/compas_seldonian_bw.c
compas_sim <- read.csv(compas_path)
```

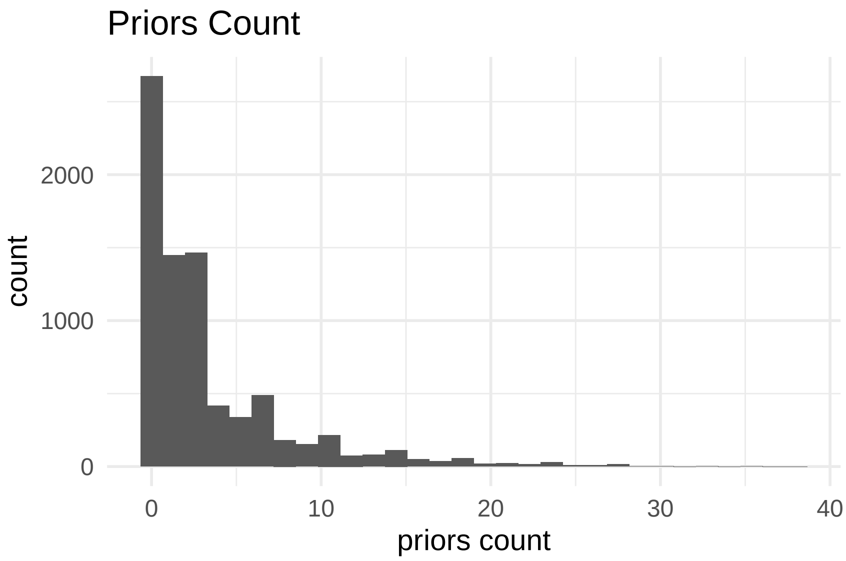
Data Subsetting

Next, let's plot the distributions of the continuous variables to choose which one we'll proceed with.

```
compas_sim %>%
  ggplot(mapping = aes(x = age)) +
  geom_histogram() +
  theme_minimal() +
  labs(title = "Age")
```



```
compas_sim %>%
  ggplot(mapping = aes(x = priors_count)) +
  geom_histogram() +
  theme_minimal() +
  labs(title = "Priors Count",
       x = "priors count")
```



Because age has more variation, we'll use it as our continuous variable. We'll convert `priors_count` into a categorical variable.

```
compas_sim <- compas_sim %>%
  mutate(prior_offense = ifelse(priors_count > 0, 1, 0)) %>%
  dplyr::select(c(race, prior_offense, age, is_recid))
```

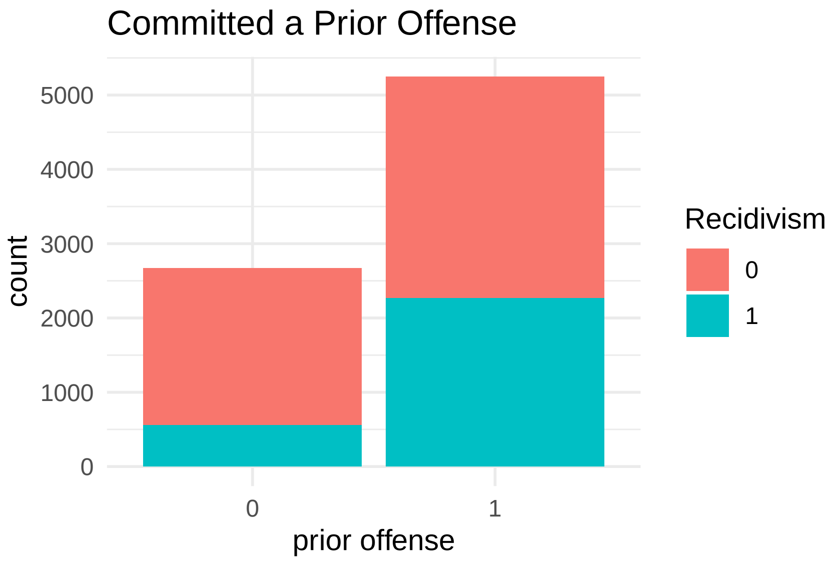
`age` seems to be a useful predictor for recidivism.

```
favstats(age ~ is_recid, data = compas_sim)
```

```
##   is_recid min Q1 median Q3 max     mean       sd    n missing
## 1        0  18 26     33 44  80 35.92591 12.24397 5102       0
## 2        1  18 24     29 37  78 32.25797 10.57842 2822       0
```

Whether a defendant has committed a prior offense or not appears to be a useful predictor for recidivism as well.

```
compas_sim %>%
  ggplot(mapping = aes(x = as.factor(prior_offense), fill = as.factor(is_recid))) +
  geom_bar() +
  theme_minimal() +
  labs(title = "Committed a Prior Offense",
       fill = "Recidivism",
       x = "prior offense")
```



We'll proceed with these 2 variables – `age` and `prior_offense` – for the simulation study. A glimpse of the data is shown below.

```r
compas_sim <- compas_sim %>%
  mutate(prior_offense = as.factor(prior_offense),
         is_recid = as.factor(is_recid))

glimpse(compas_sim)
```

```
## Rows: 7,924
## Columns: 4
## $ race          <chr> "African-American", "African-American", "Caucasian", "Ca~
## $ prior_offense <fct> 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0,~
## $ age           <int> 34, 24, 41, 39, 20, 26, 27, 23, 37, 22, 41, 47, 31, 25, ~
## $ is_recid      <fct> 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1,~
```

```r
head(compas_sim)
```

```
##                race prior_offense age is_recid
## 1 African-American             0  34        1
## 2 African-American             1  24        1
## 3        Caucasian             1  41        1
## 4        Caucasian             0  39        0
## 5        Caucasian             0  20        0
## 6        Caucasian             0  26        0
```

Data Generation Mechanism #1 (class balance)

Generating the Parent Simulation Data Set

We want a setting with 50-50 class balance for each combination of race and recidivism status. To achieve that, we'll perform sample observations with replacement. Let's create a data set with 1250 observations in each of these 4 groups, hence, 5000 observations total.

First, let's subset these 4 groups.

```
compas_b_y <- compas_sim %>%
  filter(race == "African-American" & is_recid == 1)

compas_b_n <- compas_sim %>%
  filter(race == "African-American" & is_recid == 0)

compas_w_y <- compas_sim %>%
  filter(race == "Caucasian" & is_recid == 1)

compas_w_n <- compas_sim %>%
  filter(race == "Caucasian" & is_recid == 0)
```

Next, let's randomly sample 1250 observations from each of these groups.

```
set.seed(93)
compas_b_y_balanced <- compas_b_y[sample(nrow(compas_b_y), 1250, replace = TRUE),]
compas_b_n_balanced <- compas_b_n[sample(nrow(compas_b_n), 1250, replace = TRUE),]
compas_w_y_balanced <- compas_w_y[sample(nrow(compas_w_y), 1250, replace = TRUE),]
compas_w_n_balanced <- compas_w_n[sample(nrow(compas_w_n), 1250, replace = TRUE),]
```

Finally, let's union all these together into a single data set.

```
compas_sim_balanced <- rbind(compas_b_y_balanced,
                             compas_b_n_balanced,
                             compas_w_y_balanced,
                             compas_w_n_balanced)
```

Let's also shuffle the data set row orderings to aid the machine learning algorithms later. Some algorithms face convergence problems, especially in Python, when the classes are ordered.

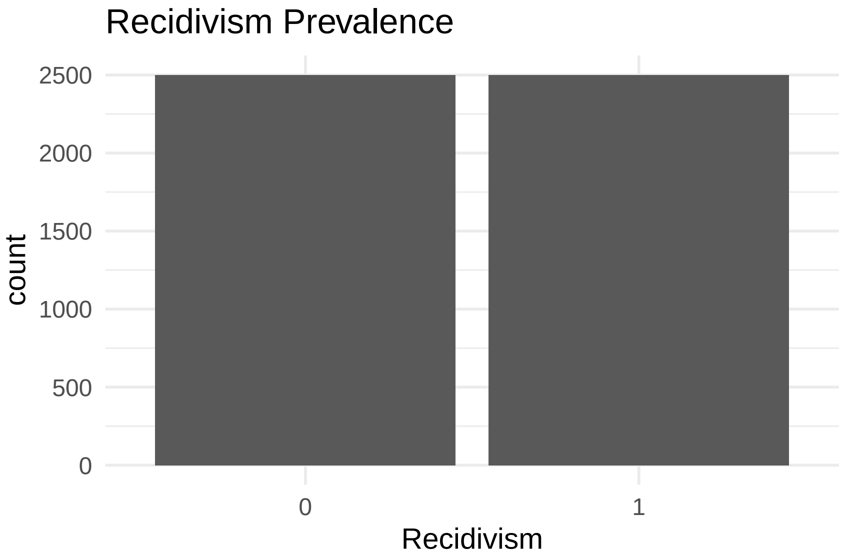
```
set.seed(123)
compas_sim_balanced <- compas_sim_balanced[sample(nrow(compas_sim_balanced),
                                                  5000,
                                                  replace = FALSE),]
```

The parent data set is now ready.

Examining Distributions of the Recidivism in the Parent Data Set

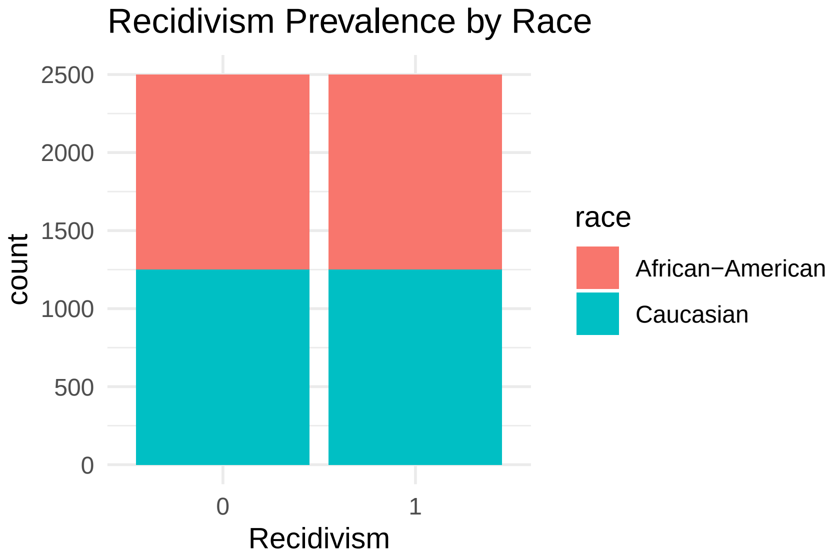
The bar plot below shows that we've achieve perfect class balance.

```
compas_sim_balanced %>%
  ggplot(mapping = aes(x = is_recid)) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Recidivism",
       title = "Recidivism Prevalence")
```



The bar plot below reveals that the balance is preserved by race as well.

```
compas_sim_balanced %>%
  ggplot(mapping = aes(x = is_recid, fill = race)) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Recidivism",
       title = "Recidivism Prevalence by Race")
```



Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```
glm1 <- glm(is_recid ~ age + prior_offense,
            data = compas_sim_balanced,
            family = binomial(logit))

msummary(glm1)
```

```
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.14520    0.10075   1.441     0.15
## age             -0.02636    0.00264  -9.985   <2e-16 ***
## prior_offense1   1.09204    0.06494  16.816   <2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6931.5  on 4999  degrees of freedom
## Residual deviance: 6547.6  on 4997  degrees of freedom
## AIC: 6553.6
##
## Number of Fisher Scoring iterations: 4
```

```
glm1augment <- glm1 %>%
  broom::augment(type.predict = "response")
glm1augment <- mutate(glm1augment, binprediction = round(.fitted, 0))
with(glm1augment, table(is_recid, binprediction))
```

```
##         binprediction
## is_recid    0    1
##        0 1339 1161
##        1  801 1699
```

```
(1331 + 1673)/5000
```

```
## [1] 0.6008
```

This model has 60.08% accuracy – this is 10% higher than a random model would achieve. Our previous data was only 4% higher in accuracy than what a blind model would achieve, so in some sense, this is better and provides more room for the Seldonian results to vary.

Discrepancy across race is also preserved, signifying that there is room to demonstrate the Seldonian algorithm's ability to satisfy fairness constraints.

```
preds <- predict(glm1, newdata = compas_sim_balanced, type="response")

compas_sim_balanced <- compas_sim_balanced %>%
  mutate(preds = preds,
         prediction = round(preds, 0),
         pred_risk = ifelse(prediction == 0, 'Low', 'High'))

compas_sim_balanced %>%
  dplyr::select(race, pred_risk, is_recid) %>%
  rename("Risk" = pred_risk,
         "Race" = race) %>%
  group_by(Race, is_recid) %>%
  mutate(Total = n()) %>%
  group_by(Risk, Race, Total) %>%
  summarise("Reoffended" = count(is_recid == 1),
            "Did Not Reoffend" = count(is_recid == 0)) %>%
  pivot_longer(cols = c("Reoffended", "Did Not Reoffend"),
               names_to = "Recidivism") %>%
  pivot_wider(
    id_cols = c("Risk", "Recidivism", "Total"),
    names_from = "Race",
    values_from = value
  ) %>%
  rename("Black" = `African-American`,
         "White" = `Caucasian`) %>%
  mutate(Black = round(100 * Black / Total, 2),
         White = round(100 * White / Total, 2)) %>%
  dplyr::select(-Total) %>%
  group_by(Risk, Recidivism) %>%
  summarize(Black = max(Black, na.rm = TRUE),
            White = max(White, na.rm = TRUE)) %>%
  filter((Risk == "High" & Recidivism == "Did Not Reoffend") |
           (Risk == "Low" & Recidivism == "Reoffended")
  ) %>%
  kable(booktabs = TRUE)
```

Risk	Recidivism	Black	White
High	Did Not Reoffend	54.16	38.72
Low	Reoffended	25.84	38.24

Data Generation Mechanism #2 (better predictive performance)

Using the balanced data set created above, we'll simulate Y values that have a stronger correlation with the 2 predictors. This is in an aim to hopefully ge better predictive performance, in addition to the class balance.

Generating the Parent Simulation Data Set

```
set.seed(123)
# define a linear combination of predictors as desired
linear_combination = - 15 - 24 * compas_sim_balanced$age + ifelse(compas_sim_balanced$prior_offense == 1

# pass through an inverse-logit function
probs = exp(linear_combination) / (1 + exp(linear_combination))

# generate 5000 Bernoulli RVs for y
is_recid_sim = rbinom(5000, 1, probs)

# join to original data frame
compas_sim_balanced_2 <- cbind(compas_sim_balanced, is_recid_sim)
```

There are 2444 defendants who did not recidivate in this data set.

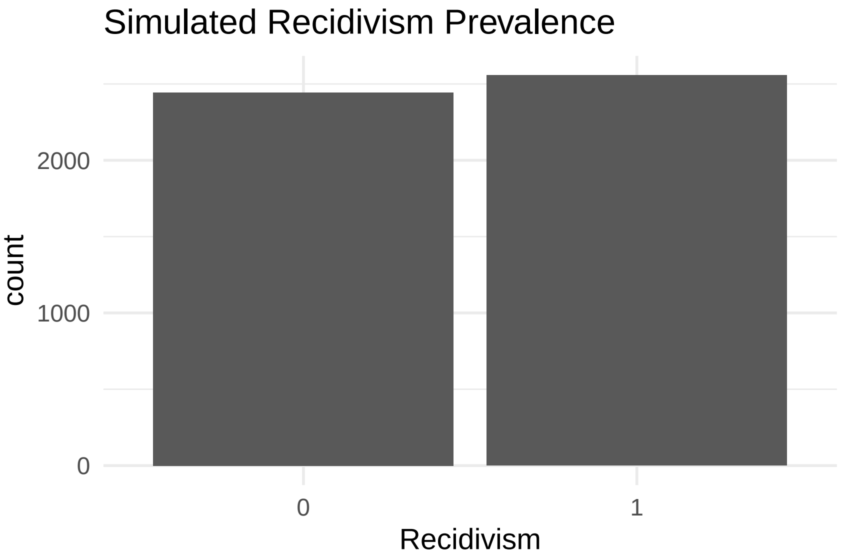
There are 2556 defendants who recidivated in this data set.

The data set seems pretty balanced. Let's look at this distribution in more detail, though.

Examining Distributions of the Recidivism in the Parent Data Set

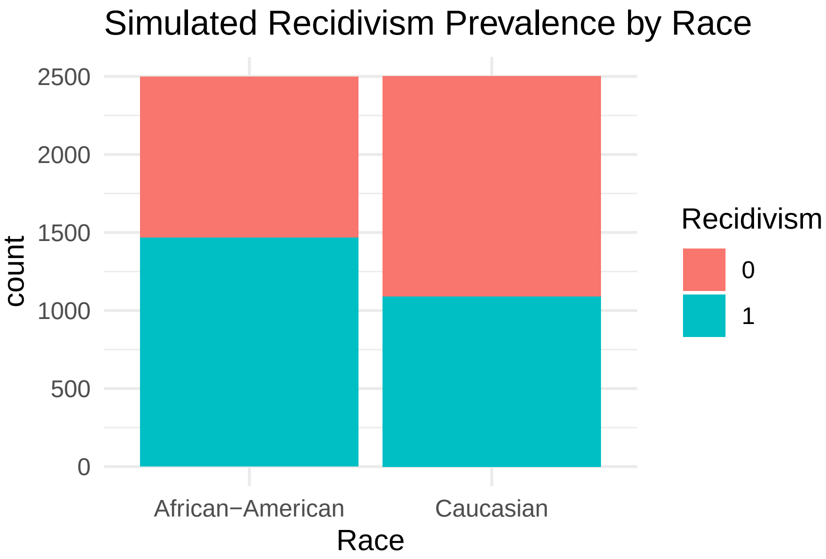
The bar plot below confirms the balanced nature of this new data set.

```r
compas_sim_balanced_2 %>%
  ggplot(mapping = aes(x = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Recidivism",
       title = "Simulated Recidivism Prevalence")
```



However, the balance is not perfectly achieved by race. Some of the proxy relationships present in the predictor variables have influenced the distribution of the Y variable by race.

```r
compas_sim_balanced_2 %>%
  ggplot(mapping = aes(x = race, fill = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Race",
       title = "Simulated Recidivism Prevalence by Race",
       fill = "Recidivism")
```



Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```
glm2 <- glm(is_recid_sim ~ age + prior_offense,
            data = compas_sim_balanced_2,
            family = binomial(logit))

msummary(glm2)
```

```
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)        337.31   12936.20   0.026    0.979
## age                -18.84     632.65  -0.030    0.976
## prior_offense1     435.61   14716.38   0.030    0.976
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6928.963  on 4999  degrees of freedom
## Residual deviance:   90.468  on 4997  degrees of freedom
## AIC: 96.468
##
## Number of Fisher Scoring iterations: 25
```

```
glm2augment <- glm2 %>%
  broom::augment(type.predict = "response")
glm2augment <- mutate(glm2augment, binprediction = round(.fitted, 0))
with(glm2augment, table(is_recid_sim, binprediction))
```

```
##             binprediction
## is_recid_sim    0    1
##            0 2418   26
##            1    0 2556
```

```
(2458 + 2529)/ 5000
```

```
## [1] 0.9974
```

This data set has 99.74% accuracy, although the distribution of race is affected. There is barely any error, so this may not be too useful for our fairness definition.

Data Generation Mechanism #3 (class balance and better predictive performance)

Let's introduce some balance across racial lines.

Generating the Parent Data Set

First, let's subset these 4 groups.

```r
compas_b_y <- compas_sim_balanced_2 %>%
  filter(race == "African-American" & is_recid_sim == 1)

compas_b_n <- compas_sim_balanced_2 %>%
  filter(race == "African-American" & is_recid_sim == 0)

compas_w_y <- compas_sim_balanced_2 %>%
  filter(race == "Caucasian" & is_recid_sim == 1)

compas_w_n <- compas_sim_balanced_2 %>%
  filter(race == "Caucasian" & is_recid_sim == 0)
```

Next, let's randomly sample 1250 observations with replacement from each of the 4 groups in the new data set.

```r
set.seed(123)
compas_b_y_balanced <- compas_b_y[sample(nrow(compas_b_y), 1250, replace = TRUE),]
compas_b_n_balanced <- compas_b_n[sample(nrow(compas_b_n), 1250, replace = TRUE),]
compas_w_y_balanced <- compas_w_y[sample(nrow(compas_w_y), 1250, replace = TRUE),]
compas_w_n_balanced <- compas_w_n[sample(nrow(compas_w_n), 1250, replace = TRUE),]
```

Finally, let's union all these together into a single data set.

```r
compas_sim_balanced_3 <- rbind(compas_b_y_balanced,
                               compas_b_n_balanced,
                               compas_w_y_balanced,
                               compas_w_n_balanced)
```

Let's also shuffle the data set row orderings to aid the machine learning algorithms later.

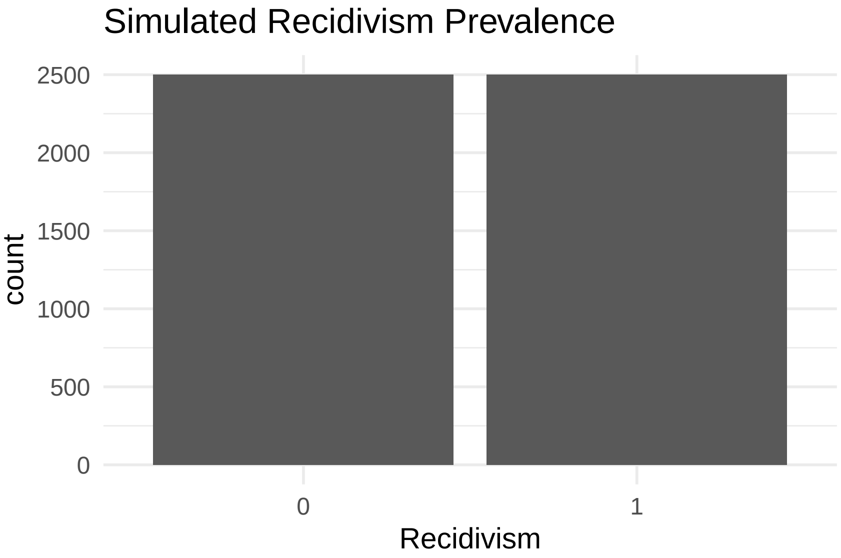
```r
set.seed(123)
compas_sim_balanced_3 <- compas_sim_balanced_3[sample(nrow(compas_sim_balanced_3),
                                                      5000,
                                                      replace = FALSE),]
```

The data set is now ready.

Examining Distributions of the Recidivism in the Parent Data Set

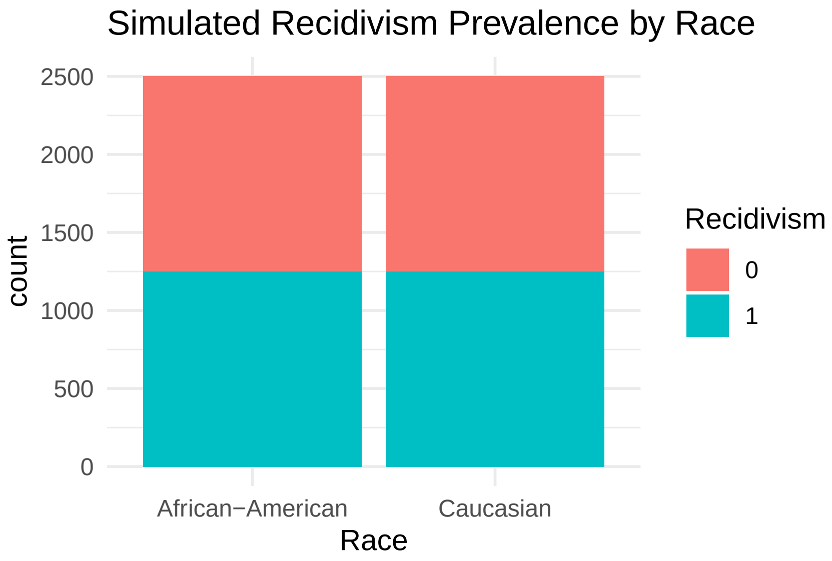
The bar plot below confirms the balanced nature of this new data set.

```
compas_sim_balanced_3 %>%
  ggplot(mapping = aes(x = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Recidivism",
       title = "Simulated Recidivism Prevalence")
```



The balance is also preserved by race.

```
compas_sim_balanced_3 %>%
  ggplot(mapping = aes(x = race, fill = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Race",
       title = "Simulated Recidivism Prevalence by Race",
       fill = "Recidivism")
```



Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good, but not perfect, predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```
glm3 <- glm(is_recid_sim ~ age + prior_offense,
            data = compas_sim_balanced_3,
            family = binomial(logit))

msummary(glm3)
```

```
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)        337.79   13087.54   0.026    0.979
## age                -18.86     642.09  -0.029    0.977
## prior_offense1     435.89   14899.32   0.029    0.977
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6931.47  on 4999  degrees of freedom
## Residual deviance:   93.19  on 4997  degrees of freedom
## AIC: 99.19
##
## Number of Fisher Scoring iterations: 25
```

```
glm3augment <- glm3 %>%
  broom::augment(type.predict = "response")
glm3augment <- mutate(glm3augment, binprediction = round(.fitted, 0))
with(glm3augment, table(is_recid_sim, binprediction))
```

```
##             binprediction
## is_recid_sim    0    1
##            0 2472   28
##            1    0 2500
```

```
(2478 + 2500)/5000
```

```
## [1] 0.9956
```

The accuracy is a little bit too good. For the final mechanism, we will follow this framework, but weaken the strengths of the correlations.

Data Generation Mechanism #4 (adjusting coefficients of Method #2)

Generating the Parent Data Set

```
set.seed(123)
# define a linear combination of predictors as desired
linear_combination = - 0.34 - 16 * compas_sim_balanced$age + ifelse(compas_sim_balanced$prior_offense ==

# pass through an inverse-logit function
probs = exp(linear_combination) / (1 + exp(linear_combination))

# generate 5000 Bernoulli RVs for y
is_recid_sim = rbinom(5000, 1, probs)

# join to original data frame
compas_sim_balanced_4 <- cbind(compas_sim_balanced, is_recid_sim)
```

There are 2689 defendants who did not recidivate in this data set.

There are 2311 defendants who recidivated in this data set.

Next, let's induce balance into this data set. First, let's subset these 4 groups.

```
compas_b_y <- compas_sim_balanced_4 %>%
  filter(race == "African-American" & is_recid_sim == 1)

compas_b_n <- compas_sim_balanced_4 %>%
  filter(race == "African-American" & is_recid_sim == 0)

compas_w_y <- compas_sim_balanced_4 %>%
  filter(race == "Caucasian" & is_recid_sim == 1)

compas_w_n <- compas_sim_balanced_4 %>%
  filter(race == "Caucasian" & is_recid_sim == 0)
```

Next, let's randomly sample 1250 observations with replacement from each of the 4 groups in the new data set.

```r
set.seed(123)
compas_b_y_balanced <- compas_b_y[sample(nrow(compas_b_y), 1250, replace = TRUE),]
compas_b_n_balanced <- compas_b_n[sample(nrow(compas_b_n), 1250, replace = TRUE),]
compas_w_y_balanced <- compas_w_y[sample(nrow(compas_w_y), 1250, replace = TRUE),]
compas_w_n_balanced <- compas_w_n[sample(nrow(compas_w_n), 1250, replace = TRUE),]
```

Finally, let's union all these together into a single data set.

```r
compas_sim_balanced_4 <- rbind(compas_b_y_balanced,
                               compas_b_n_balanced,
                               compas_w_y_balanced,
                               compas_w_n_balanced)
```

Let's also shuffle the data set row orderings to aid the machine learning algorithms later.

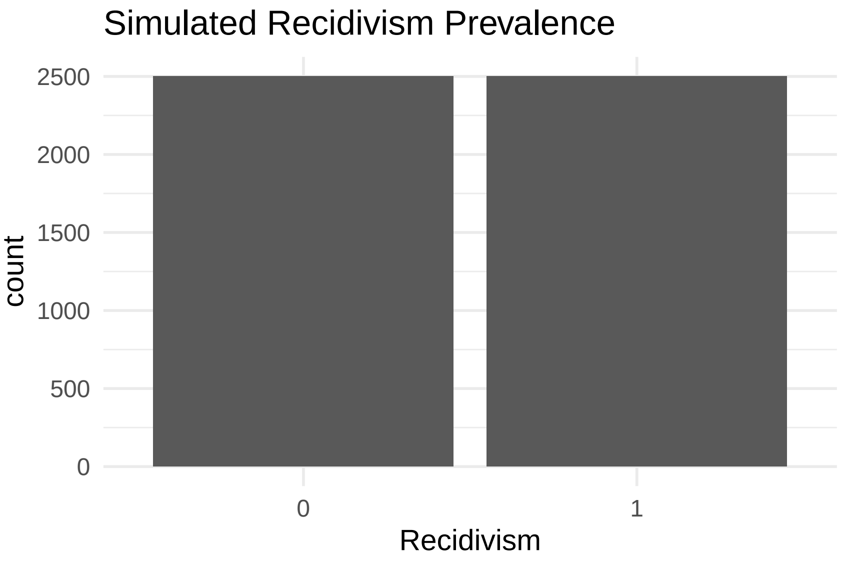
```r
set.seed(123)
compas_sim_balanced_4 <- compas_sim_balanced_4[sample(nrow(compas_sim_balanced_4),
                                                      5000,
                                                      replace = FALSE),]
```

The data set is now ready.

Examining Distributions of the Recidivism in the Parent Data Set

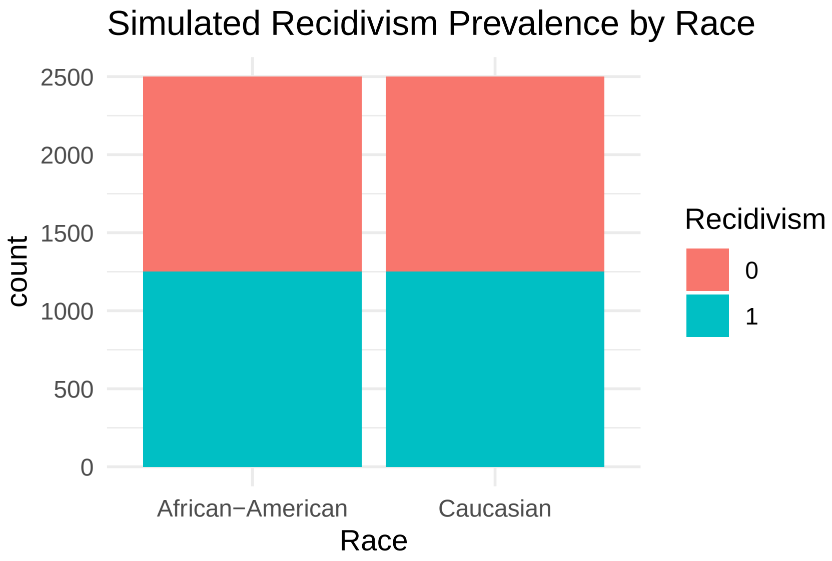
The bar plot below confirms the balanced nature of this new data set.

```r
compas_sim_balanced_4 %>%
  ggplot(mapping = aes(x = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Recidivism",
       title = "Simulated Recidivism Prevalence")
```



The balance is also preserved by race.

```r
compas_sim_balanced_4 %>%
  ggplot(mapping = aes(x = race, fill = as.factor(is_recid_sim))) +
  geom_bar() +
  theme_minimal() +
  labs(x = "Race",
       title = "Simulated Recidivism Prevalence by Race",
       fill = "Recidivism")
```



Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good, but not perfect, predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```r
glm4 <- glm(is_recid_sim ~ age + prior_offense,
            data = compas_sim_balanced_4,
            family = binomial(logit))

msummary(glm4)
```

```
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)       674.97   16198.13   0.042    0.967
## age               -36.59     822.94  -0.044    0.965
## prior_offense1    697.35   15794.28   0.044    0.965
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6.9315e+03  on 4999  degrees of freedom
## Residual deviance: 4.3909e-06  on 4997  degrees of freedom
## AIC: 6
##
## Number of Fisher Scoring iterations: 25
```

```
glm4augment <- glm4 %>%
  broom::augment(type.predict = "response")
glm4augment <- mutate(glm4augment, binprediction = round(.fitted, 0))
with(glm4augment, table(is_recid_sim, binprediction))
```

```
##             binprediction
## is_recid_sim    0    1
##            0 2500    0
##            1    0 2500
```

Hmm, maybe we need to introduce an error term.

```
mean(linear_combination)
```

```
## [1] -137.7448
```

```
sd(linear_combination)
```

```
## [1] 328.2499
```

Maybe it's okay to not have class balance? Think about this some more.

Data Generation Mechanism #5 (class balance, but with some error)

Generating the Parent Data Set

```
set.seed(93)
# define a linear combination of predictors as desired
error <- rnorm(n = 5000, mean = 100, sd = 100)
linear_combination = - 0.34 - 16 * compas_sim_balanced$age + ifelse(compas_sim_balanced$prior_offense ==

# pass through an inverse-logit function
```

```r
probs = exp(linear_combination) / (1 + exp(linear_combination))

# generate 5000 Bernoulli RVs for y
is_recid_sim = rbinom(5000, 1, probs)

# join to original data frame
compas_sim_balanced_5 <- cbind(compas_sim_balanced, is_recid_sim)
```

There are 2344 defendants who did not recidivate in this data set.

There are 2656 defendants who recidivated in this data set.

Next, let's induce balance into this data set. First, let's subset these 4 groups.

```r
compas_b_y <- compas_sim_balanced_5 %>%
  filter(race == "African-American" & is_recid_sim == 1)

compas_b_n <- compas_sim_balanced_5 %>%
  filter(race == "African-American" & is_recid_sim == 0)

compas_w_y <- compas_sim_balanced_5 %>%
  filter(race == "Caucasian" & is_recid_sim == 1)

compas_w_n <- compas_sim_balanced_5 %>%
  filter(race == "Caucasian" & is_recid_sim == 0)
```

Next, let's randomly sample 1250 observations with replacement from each of the 4 groups in the new data set.

```r
set.seed(93)
compas_b_y_balanced <- compas_b_y[sample(nrow(compas_b_y), 1250, replace = TRUE),]
compas_b_n_balanced <- compas_b_n[sample(nrow(compas_b_n), 1250, replace = TRUE),]
compas_w_y_balanced <- compas_w_y[sample(nrow(compas_w_y), 1250, replace = TRUE),]
compas_w_n_balanced <- compas_w_n[sample(nrow(compas_w_n), 1250, replace = TRUE),]
```

Finally, let's union all these together into a single data set.

```r
compas_sim_balanced_5 <- rbind(compas_b_y_balanced,
                               compas_b_n_balanced,
                               compas_w_y_balanced,
                               compas_w_n_balanced)
```

Let's also shuffle the data set row orderings to aid the machine learning algorithms later.

```
set.seed(93)
compas_sim_balanced_5 <- compas_sim_balanced_5[sample(nrow(compas_sim_balanced_5),
                                                       5000,
                                                       replace = FALSE),]
```

The data set is now ready.

Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good, but not perfect, predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```
glm5 <- glm(is_recid_sim ~ age + prior_offense,
            data = compas_sim_balanced_5,
            family = binomial(logit))

msummary(glm5)
```

```
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)        0.2681     0.7451    0.36    0.719
## age               -0.2785     0.0104  -26.78   <2e-16 ***
## prior_offense1    11.8683     0.7380   16.08   <2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6931.5  on 4999  degrees of freedom
## Residual deviance: 1493.3  on 4997  degrees of freedom
## AIC: 1499.3
##
## Number of Fisher Scoring iterations: 9
```

```
glm5augment <- glm5 %>%
  broom::augment(type.predict = "response")
glm5augment <- mutate(glm5augment, binprediction = round(.fitted, 0))
with(glm5augment, table(is_recid_sim, binprediction))
```

```
##             binprediction
## is_recid_sim    0    1
##            0 2310  190
##            1  118 2382
```

```
(2324 + 2370)/ 5000
```

```
## [1] 0.9388
```

This model has 93.88% accuracy.

```
preds <- predict(glm5, newdata = compas_sim_balanced_5, type="response")

compas_sim_balanced_5 <- compas_sim_balanced_5 %>%
  mutate(preds = preds,
         prediction = round(preds, 0),
         pred_risk = ifelse(prediction == 0, 'Low', 'High'))
```

Barely discrepancy is observed.

```
compas_sim_balanced_5 %>%
  dplyr::select(race, pred_risk, is_recid) %>%
  rename("Risk" = pred_risk,
         "Race" = race) %>%
  group_by(Race, is_recid) %>%
  mutate(Total = n()) %>%
  group_by(Risk, Race, Total) %>%
  summarise("Reoffended" = count(is_recid == 1),
            "Did Not Reoffend" = count(is_recid == 0)) %>%
  pivot_longer(cols = c("Reoffended", "Did Not Reoffend"),
               names_to = "Recidivism") %>%
  pivot_wider(
    id_cols = c("Risk", "Recidivism", "Total"),
    names_from = "Race",
    values_from = value
  ) %>%
  rename("Black" = `African-American`,
         "White" = `Caucasian`) %>%
  mutate(Black = round(100 * Black / Total, 2),
         White = round(100 * White / Total, 2)) %>%
  dplyr::select(-Total) %>%
  group_by(Risk, Recidivism) %>%
  summarize(Black = max(Black, na.rm = TRUE),
            White = max(White, na.rm = TRUE)) %>%
  filter((Risk == "High" & Recidivism == "Did Not Reoffend") |
           (Risk == "Low" & Recidivism == "Reoffended")
  ) %>%
  kable(booktabs = TRUE)
```

Risk	Recidivism	Black	White
High	Did Not Reoffend	47.06	42.79
Low	Reoffended	34.80	36.06

Data Generation Mechanism #6 (class balance, but with more error)

Generating the Parent Data Set

```
set.seed(93)
# define a linear combination of predictors as desired
error <- rnorm(n = 5000, mean = 50, sd = 75)
error2 <- rnorm(n = 5000, mean = 50, sd = 75)
linear_combination = - 0.34 - 16 * compas_sim_balanced$age + ifelse(compas_sim_balanced$prior_offense ==

# pass through an inverse-logit function
probs = exp(linear_combination) / (1 + exp(linear_combination))
```

```
# generate 5000 Bernoulli RVs for y
is_recid_sim = rbinom(5000, 1, probs)

# join to original data frame
compas_sim_balanced_6 <- cbind(compas_sim_balanced, is_recid_sim)
```

There are 2362 defendants who did not recidivate in this data set.

There are 2638 defendants who recidivated in this data set.

Next, let's induce balance into this data set. First, let's subset these 4 groups.

```
compas_b_y <- compas_sim_balanced_6 %>%
  filter(race == "African-American" & is_recid_sim == 1)

compas_b_n <- compas_sim_balanced_6 %>%
  filter(race == "African-American" & is_recid_sim == 0)

compas_w_y <- compas_sim_balanced_6 %>%
  filter(race == "Caucasian" & is_recid_sim == 1)

compas_w_n <- compas_sim_balanced_6 %>%
  filter(race == "Caucasian" & is_recid_sim == 0)
```

Next, let's randomly sample 1250 observations with replacement from each of the 4 groups in the new data set.

```
set.seed(93)
compas_b_y_balanced <- compas_b_y[sample(nrow(compas_b_y), 1250, replace = TRUE),]
compas_b_n_balanced <- compas_b_n[sample(nrow(compas_b_n), 1250, replace = TRUE),]
compas_w_y_balanced <- compas_w_y[sample(nrow(compas_w_y), 1250, replace = TRUE),]
compas_w_n_balanced <- compas_w_n[sample(nrow(compas_w_n), 1250, replace = TRUE),]
```

Finally, let's union all these together into a single data set.

```
compas_sim_balanced_6 <- rbind(compas_b_y_balanced,
                               compas_b_n_balanced,
                               compas_w_y_balanced,
                               compas_w_n_balanced)
```

Let's also shuffle the data set row orderings to aid the machine learning algorithms later.

```r
set.seed(93)
compas_sim_balanced_6 <- compas_sim_balanced_6[sample(nrow(compas_sim_balanced_6),
                                                       5000,
                                                       replace = FALSE),]
```

The data set is now ready.

Assessing Baseline Predictive Performance of the Parent Data Set

We want to make sure that our data set also has good, but not perfect, predictive performance. We'll fit a logistic regression and assess baseline accuracy.

```r
glm6 <- glm(is_recid_sim ~ age + prior_offense,
            data = compas_sim_balanced_6,
            family = binomial(logit))

msummary(glm6)
```

```
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     0.223212   0.613920   0.364    0.716
## age            -0.255416   0.009064 -28.178   <2e-16 ***
## prior_offense1 10.899868   0.606450  17.973   <2e-16 ***
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6931.5  on 4999  degrees of freedom
## Residual deviance: 1689.6  on 4997  degrees of freedom
## AIC: 1695.6
##
## Number of Fisher Scoring iterations: 9
```

```r
glm6augment <- glm6 %>%
  broom::augment(type.predict = "response")
glm6augment <- mutate(glm6augment, binprediction = round(.fitted, 0))
with(glm6augment, table(is_recid_sim, binprediction))
```

```
##             binprediction
## is_recid_sim    0    1
##            0 2289  211
##            1  147 2353
```

```r
(2309 + 2373)/5000
```

```
## [1] 0.9364
```

This data set has 93.64% accuracy.

```r
preds <- predict(glm6, newdata=compas_sim_balanced_6, type="response")

compas_sim_balanced_6 <- compas_sim_balanced_6 %>%
  mutate(preds = preds,
         prediction = round(preds, 0),
         pred_risk = ifelse(prediction == 0, 'Low', 'High'))
```

Some discrepancy is preserved, and in the same direction as previous analyses, although not of the same magnitude. The discrimination statistic is ~0.09.

```
compas_sim_balanced_6 %>%
  dplyr::select(race, pred_risk, is_recid) %>%
  rename("Risk" = pred_risk,
         "Race" = race) %>%
  group_by(Race, is_recid) %>%
  mutate(Total = n()) %>%
  group_by(Risk, Race, Total) %>%
  summarise("Reoffended" = count(is_recid == 1),
            "Did Not Reoffend" = count(is_recid == 0)) %>%
  pivot_longer(cols = c("Reoffended", "Did Not Reoffend"),
               names_to = "Recidivism") %>%
  pivot_wider(
    id_cols = c("Risk", "Recidivism", "Total"),
    names_from = "Race",
    values_from = value
  ) %>%
  rename("Black" = `African-American`,
         "White" = `Caucasian`) %>%
  mutate(Black = round(100 * Black / Total, 2),
         White = round(100 * White / Total, 2)) %>%
  dplyr::select(-Total) %>%
  group_by(Risk, Recidivism) %>%
  summarize(Black = max(Black, na.rm = TRUE),
            White = max(White, na.rm = TRUE)) %>%
  filter((Risk == "High" & Recidivism == "Did Not Reoffend") |
           (Risk == "Low" & Recidivism == "Reoffended")
  ) %>%
  kable(booktabs = TRUE)
```

Risk	Recidivism	Black	White
High	Did Not Reoffend	46.22	40.94
Low	Reoffended	33.17	34.39