Morse_Brobst_sampling

Supervised Machine Learning: Plant Health

Reading in the dataset

```
health_data <- "data/plant_moniter_health_data.csv"
data <- read.csv(health_data,
                 header = TRUE, sep = ",")
```

```
data$Health_Status = as.factor(data$Health_Status)
summary(data)
```

```
##    Plant_ID         Temperature_C     Humidity_.    Soil_Moisture_.
##  Length:1000        Min.   :15.28   Min.   :30.60   Min.   : -0.2927
##  Class :character   1st Qu.:23.06   1st Qu.:53.94   1st Qu.: 35.2800
##  Mode  :character   Median :25.08   Median :60.63   Median : 44.9962
##                     Mean   :25.06   Mean   :60.71   Mean   : 45.0875
##                     3rd Qu.:26.94   3rd Qu.:67.29   3rd Qu.: 54.9137
##                     Max.   :36.56   Max.   :91.93   Max.   :103.8936
##     Soil_pH      Nutrient_Level  Light_Intensity_lux  Health_Score
##  Min.   :5.035   Min.   :18.23   Min.   :11301       Min.   : 52.87
##  1st Qu.:6.131   1st Qu.:43.17   1st Qu.:17919       1st Qu.: 72.45
##  Median :6.500   Median :49.82   Median :19872       Median : 79.45
##  Mean   :6.491   Mean   :49.51   Mean   :19860       Mean   : 79.72
##  3rd Qu.:6.833   3rd Qu.:56.39   3rd Qu.:21837       3rd Qu.: 87.00
##  Max.   :8.122   Max.   :81.13   Max.   :29295       Max.   :115.29
##  Health_Status
##  0:174
##  1:826
##
##
##
##
```

Undersampling and Oversampling to address the unbalanced response

```
# dropping the Plant_ID and renaming columns for simplicity
data <- data[,-c(1)]
data <- data %>%
  rename(
    Humidity = Humidity_.,
    Soil_Moisture = Soil_Moisture_.
    )
```

First, we set up the basic linear model to compare results with.

```
model <- glm(Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level + Ligh
             family='binomial',
             data=data)

summary(model)
```

```
##
## Call:
## glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
##     Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
##     data = data)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          2.200e+00  1.586e+00   1.387   0.1653
## Temperature_C       -5.000e-02  2.847e-02  -1.756   0.0791 .
## Humidity             5.100e-03  8.412e-03   0.606   0.5443
## Soil_Moisture       -6.615e-03  5.667e-03  -1.167   0.2431
## Soil_pH              5.156e-02  1.631e-01   0.316   0.7520
## Nutrient_Level       2.253e-03  8.447e-03   0.267   0.7896
## Light_Intensity_lux  8.306e-06  2.781e-05   0.299   0.7651
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 924.34  on 999  degrees of freedom
## Residual deviance: 919.08  on 993  degrees of freedom
## AIC: 933.08
##
## Number of Fisher Scoring iterations: 4
```

```
'
Call:
glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
    Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
    data = data)

Coefficients:
                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          2.200e+00  1.586e+00   1.387   0.1653
Temperature_C       -5.000e-02  2.847e-02  -1.756   0.0791 .
Humidity             5.100e-03  8.412e-03   0.606   0.5443
Soil_Moisture       -6.615e-03  5.667e-03  -1.167   0.2431
Soil_pH              5.156e-02  1.631e-01   0.316   0.7520
Nutrient_Level       2.253e-03  8.447e-03   0.267   0.7896
Light_Intensity_lux  8.306e-06  2.781e-05   0.299   0.7651
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 924.34  on 999  degrees of freedom
Residual deviance: 919.08  on 993  degrees of freedom
AIC: 933.08

Number of Fisher Scoring iterations: 4
'
```

```
## [1] "\nCall:\nglm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture + \n    Soil_pH
```

```
train_size <- 0.75*nrow(data)
training_sample <- sample(1:nrow(data), train_size)
training_dataset <- data[training_sample, ]
testing_dataset <- data[-training_sample, ]

model_validate <- glm(
  Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level + Light_Intensity
  data=training_dataset,
  family='binomial'
)

y_pred_probs <- predict(
  model_validate,
  newdata=testing_dataset,
  type='response'
)

y_pred <- ifelse(y_pred_probs > 0.5, 1, 0)

table(y_pred, testing_dataset$Health_Status)
```

```
##
## y_pred   0   1
##      1  41 209
```

```
'
y_pred   0   1
     1  31 219
'
```

```
## [1] "\ny_pred   0   1\n     1  31 219\n"
```

This shows the problem we need to address. Since class 0 has so few observations the model fails to learn patterns in the data. We need to try both oversampling class 0 and undersampling class 1.

Undersampling class 1

To randomly undersample, we randomly remove observations from the 1 class so that the total observations for class 1 is equal to the total observations for class 0. This is called Random Downsampling.

```r
data_0 <- data[data$Health_Status == 0, ]
n_obs_0 <- nrow(data_0)
data_1 <- data[data$Health_Status == 1, ]
n_obs_1 <- nrow(data_1)

remove_is <- sample(1:n_obs_1, n_obs_1-n_obs_0)
data_1_undersampled <- data_1[-remove_is, ]

data_undersampled <- rbind(data_0, data_1_undersampled)
data_undersampled <- data_undersampled[sample(nrow(data_undersampled)), ]
```

Now, I will retry the logistic regression.

```r
model_under <- glm(Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level +
             family='binomial',
             data=data_undersampled)

summary(model_under)
```

```
##
## Call:
## glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
##     Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
##     data = data_undersampled)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.613e-02  2.017e+00  -0.018    0.986
## Temperature_C       -2.874e-02  3.761e-02  -0.764    0.445
## Humidity             5.339e-03  1.069e-02   0.499    0.617
## Soil_Moisture       -7.316e-03  7.019e-03  -1.042    0.297
## Soil_pH              1.153e-01  2.100e-01   0.549    0.583
## Nutrient_Level      -2.571e-03  1.102e-02  -0.233    0.816
## Light_Intensity_lux  7.549e-06  3.732e-05   0.202    0.840
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 482.43  on 347  degrees of freedom
## Residual deviance: 479.95  on 341  degrees of freedom
## AIC: 493.95
##
## Number of Fisher Scoring iterations: 3
```

```
'
Call:
glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
    Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
    data = data_undersampled)

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          6.233e-01  2.107e+00   0.296    0.767
```

```
Temperature_C       -5.274e-02  3.871e-02  -1.362    0.173
Humidity             8.015e-03  1.065e-02   0.753    0.452
Soil_Moisture       -9.400e-03  7.461e-03  -1.260    0.208
Soil_pH              5.636e-02  2.096e-01   0.269    0.788
Nutrient_Level      -8.063e-04  1.087e-02  -0.074    0.941
Light_Intensity_lux  1.603e-05  3.509e-05   0.457    0.648

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 482.43  on 347  degrees of freedom
Residual deviance: 477.71  on 341  degrees of freedom
AIC: 491.71

Number of Fisher Scoring iterations: 4
'
```

```
## [1] "\nCall:\nglm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture + \n    Soil_pH
```

```
train_size <- 0.75*nrow(data_undersampled)
training_sample <- sample(1:nrow(data_undersampled), train_size)
training_dataset <- data_undersampled[training_sample, ]
testing_dataset <- data_undersampled[-training_sample, ]

model_validate <- glm(
  Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level + Light_Intensity
  data=training_dataset,
  family='binomial'
)

y_pred_probs <- predict(
  model_validate,
  newdata=testing_dataset,
  type='response'
)

y_pred <- ifelse(y_pred_probs > 0.5, 1, 0)

table(y_pred, testing_dataset$Health_Status)
```

```
##
## y_pred  0  1
##      0 17 21
##      1 28 21
```

```
'
y_pred  0  1
     0 25 17
     1 19 26
'
```

```
## [1] "\ny_pred  0  1\n     0 25 17\n     1 19 26\n"
```

Oversampling class 0

To oversample class 0, we add "new" class 0 to the data by sampling (with replacement) from the existing class 0 observations. This is also called random Upsampling.

```
data_0_oversampled <- data_0[sample(n_obs_0, n_obs_1, replace = TRUE), ]
data_oversampled <- rbind(data_0_oversampled, data_1)
data_oversampled <- data_oversampled[sample(nrow(data_oversampled)), ]
```

```
model_over <- glm(Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level +
             family='binomial',
             data=data_oversampled)

summary(model_over)
```

```
##
## Call:
## glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
##     Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
##     data = data_oversampled)
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          1.115e+00  9.317e-01   1.196   0.2316
## Temperature_C       -2.899e-02  1.712e-02  -1.693   0.0905 .
## Humidity             1.383e-03  4.763e-03   0.290   0.7715
## Soil_Moisture       -5.197e-03  3.341e-03  -1.555   0.1199
## Soil_pH             -2.311e-02  9.752e-02  -0.237   0.8126
## Nutrient_Level       1.582e-03  5.001e-03   0.316   0.7517
## Light_Intensity_lux -8.164e-06  1.671e-05  -0.488   0.6252
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2290.2  on 1651  degrees of freedom
## Residual deviance: 2284.0  on 1645  degrees of freedom
## AIC: 2298
##
## Number of Fisher Scoring iterations: 3
```

```
'
Call:
glm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture +
    Soil_pH + Nutrient_Level + Light_Intensity_lux, family = "binomial",
    data = data_oversampled)

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          2.354e+00  9.660e-01   2.437 0.014805 *
Temperature_C       -5.864e-02  1.723e-02  -3.403 0.000667 ***
Humidity             9.647e-03  4.893e-03   1.972 0.048624 *
Soil_Moisture       -9.539e-03  3.395e-03  -2.810 0.004960 **
```

```
Soil_pH             -1.700e-01  9.841e-02  -1.727 0.084149 .
Nutrient_Level      -1.671e-03  5.129e-03  -0.326 0.744626
Light_Intensity_lux  8.688e-06  1.706e-05   0.509 0.610614
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2290.2  on 1651  degrees of freedom
Residual deviance: 2261.7  on 1645  degrees of freedom
AIC: 2275.7

Number of Fisher Scoring iterations: 4
'
```

```
## [1] "\nCall:\nglm(formula = Health_Status ~ Temperature_C + Humidity + Soil_Moisture + \n    Soil_pH
```

```
train_size <- 0.75*nrow(data_oversampled)
training_sample <- sample(1:nrow(data_oversampled), train_size)
training_dataset <- data_oversampled[training_sample, ]
testing_dataset <- data_oversampled[-training_sample, ]

model_validate <- glm(
  Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level + Light_Intensity
  data=training_dataset,
  family='binomial'
)

y_pred_probs <- predict(
  model_validate,
  newdata=testing_dataset,
  type='response'
)

y_pred <- ifelse(y_pred_probs > 0.5, 1, 0)

table(y_pred, testing_dataset$Health_Status)
```

```
##
## y_pred   0   1
##      0  79  70
##      1 148 116
```

```
'
y_pred   0   1
     0 100  99
     1 104 110
'
```

```
## [1] "\ny_pred   0   1\n     0 100  99\n     1 104 110\n"
```

The 2 sampling approaches show an improvement on the model compared to using the imbalanced data. However, since the random sampling is either removing most of a class or adding (with replacement) to

a class the models have a lot of variance. This is seen easily by running the above a few times. To get somewhat stable results we will need to consider repeating the process many time over and looking at the variance in accuracy of the models and coefficient estimates.

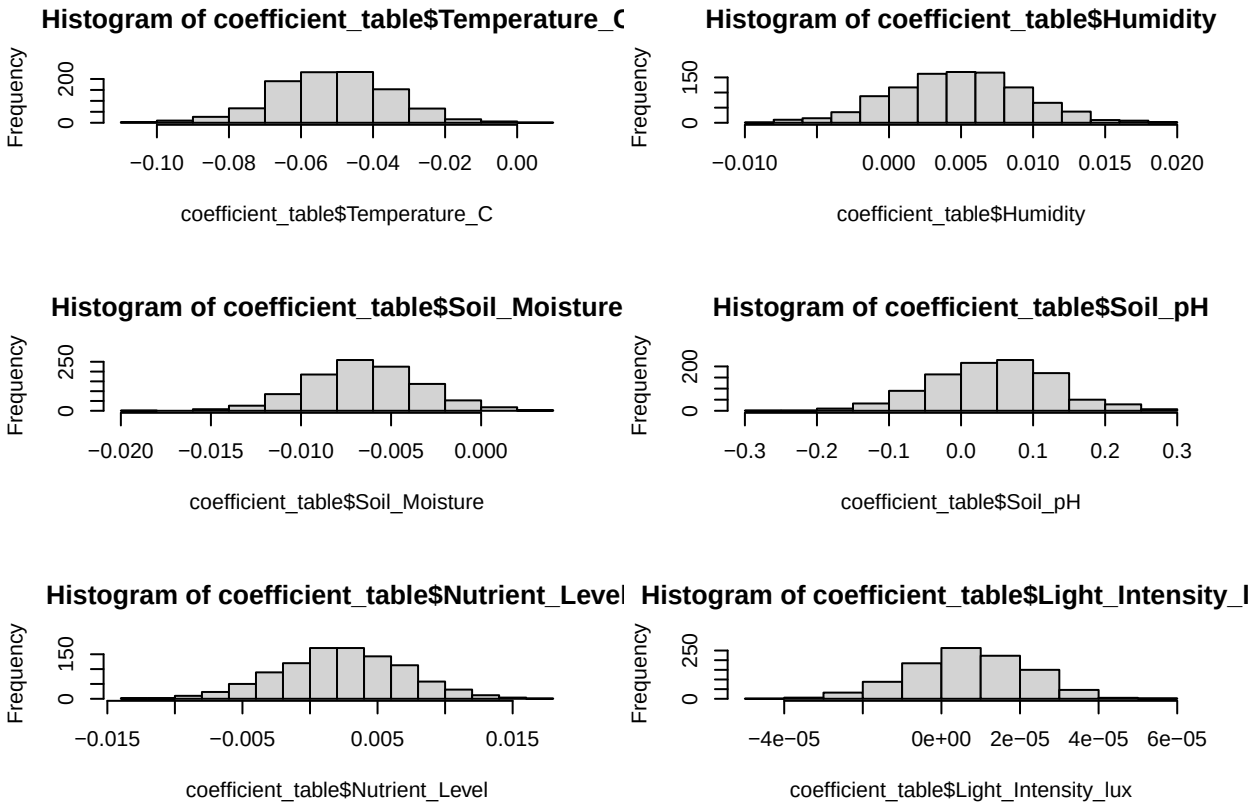
Variance of over sampling

We created a function to return the oversampled dataset without a seed set. This function can be called repetitively to get different oversamples.

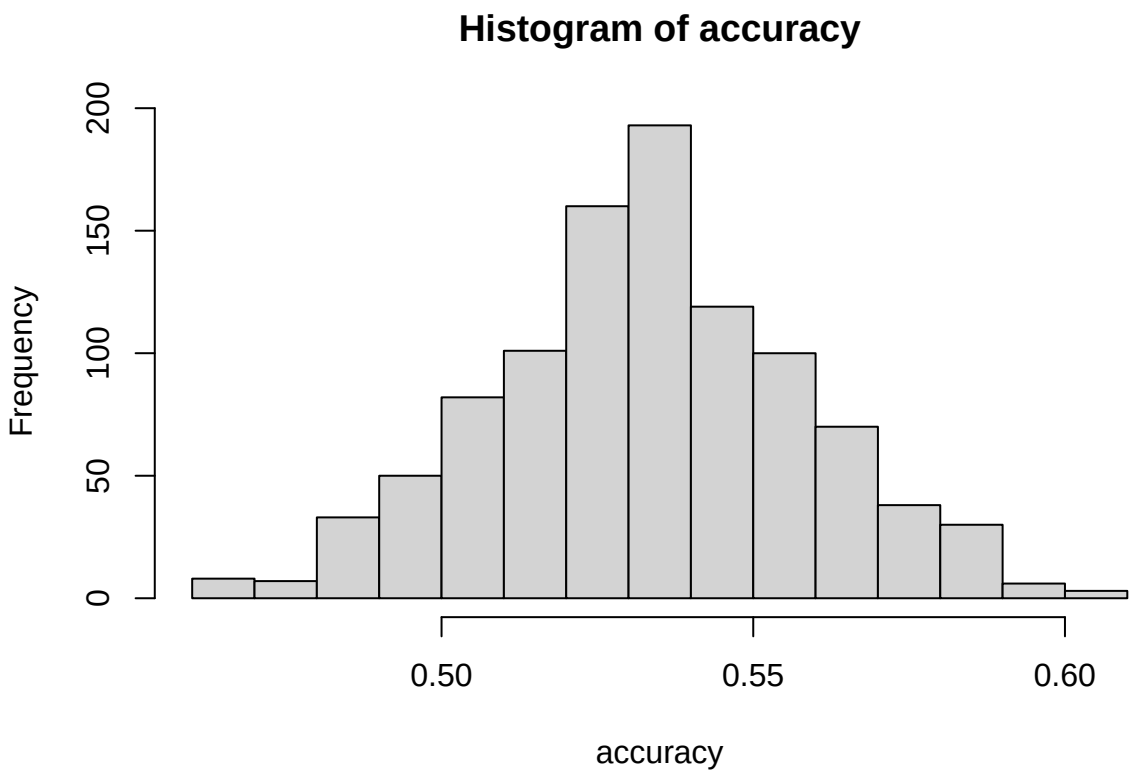
```
oversample <- function(data){
  data_0 <- data[data$Health_Status == 0, ]
  n_obs_0 <- nrow(data_0)
  data_1 <- data[data$Health_Status == 1, ]
  n_obs_1 <- nrow(data_1)

  data_0_oversampled <- data_0[sample(n_obs_0, n_obs_1, replace = TRUE), ]
  data_oversampled <- rbind(data_0_oversampled, data_1)
  data_oversampled <- data_oversampled[sample(nrow(data_oversampled)), ]

  data_oversampled
}
```

```
B <- 1000
coefficient_table <- NULL
accuracy <- seq(B)

for(i in 1:B){
  data_sample <- oversample(data)
  train_size <- 0.75*nrow(data_sample)
  training_sample <- sample(1:nrow(data_sample), train_size)
  training_dataset <- data_sample[training_sample, ]
  testing_dataset <- data_sample[-training_sample, ]

  model_validate <- glm(
    Health_Status ~ Temperature_C + Humidity + Soil_Moisture + Soil_pH + Nutrient_Level + Light_Intensi
    data=training_dataset,
    family='binomial'
  )

  y_pred_probs <- predict(model_validate, newdata=testing_dataset,type='response')
  y_pred <- ifelse(y_pred_probs > 0.5, 1, 0)
  correct <- ifelse(y_pred == testing_dataset$Health_Status, 1, 0)
  accuracy[i] <- mean(correct)

  if(is.null(coefficient_table)){
    coefficient_table <- model_validate$coefficients
  } else {
    coefficient_table <- rbind(coefficient_table, model_validate$coefficients)

  }
}
coefficient_table <- as.data.frame(coefficient_table)
```

```r
par(mfrow=c(3, 2))
hist(coefficient_table$Temperature_C)
hist(coefficient_table$Humidity)
hist(coefficient_table$Soil_Moisture)
hist(coefficient_table$Soil_pH)
hist(coefficient_table$Nutrient_Level)
hist(coefficient_table$Light_Intensity_lux)
```




```r
hist(accuracy)
```



The Variance in the model's performance is alarming however, with enough iterations we see that the mean accuracy is greater than 50% so some information about the population and the relationship is represented in the sample.

package imbalance

This package provides other strategies for oversampling. This could be interesting to play with but unfortunately RStudio struggled to download the package

```
#install.packages("devtools")
#devtools::install_github("ncordon/imbalance")
```