TP2 MRR

Noah KWA MOUTOME - Victor TAN

2023-10-31

IV. Cookies Study

```
cookies_data <- read.csv("cookies.csv")
dim(cookies_data)
```

```
## [1]  32 701
```

We see that there are 700 co-variables. We can assume that some of them are less important than the others. To see this, let's do a Ridge regression and look at the coefficient of each co-variables.

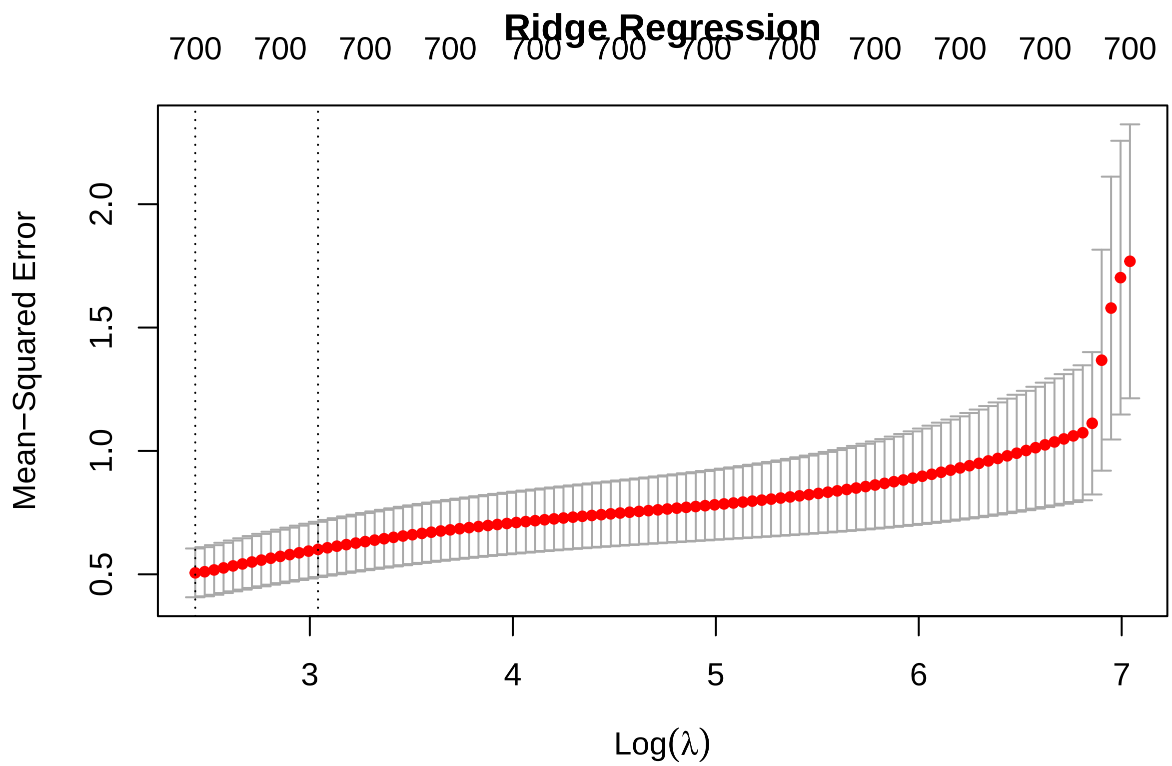
```
library(glmnet)
```

```
## Le chargement a nécessité le package : Matrix
```

```
## Loaded glmnet 4.1-8
```

```
y <- cookies_data[, 1]
X <- cookies_data[, -1]

cv_ridge_model <- cv.glmnet(as.matrix(X), y, alpha=0, standardize = TRUE)
plot(cv_ridge_model, main="Ridge Regression")
```



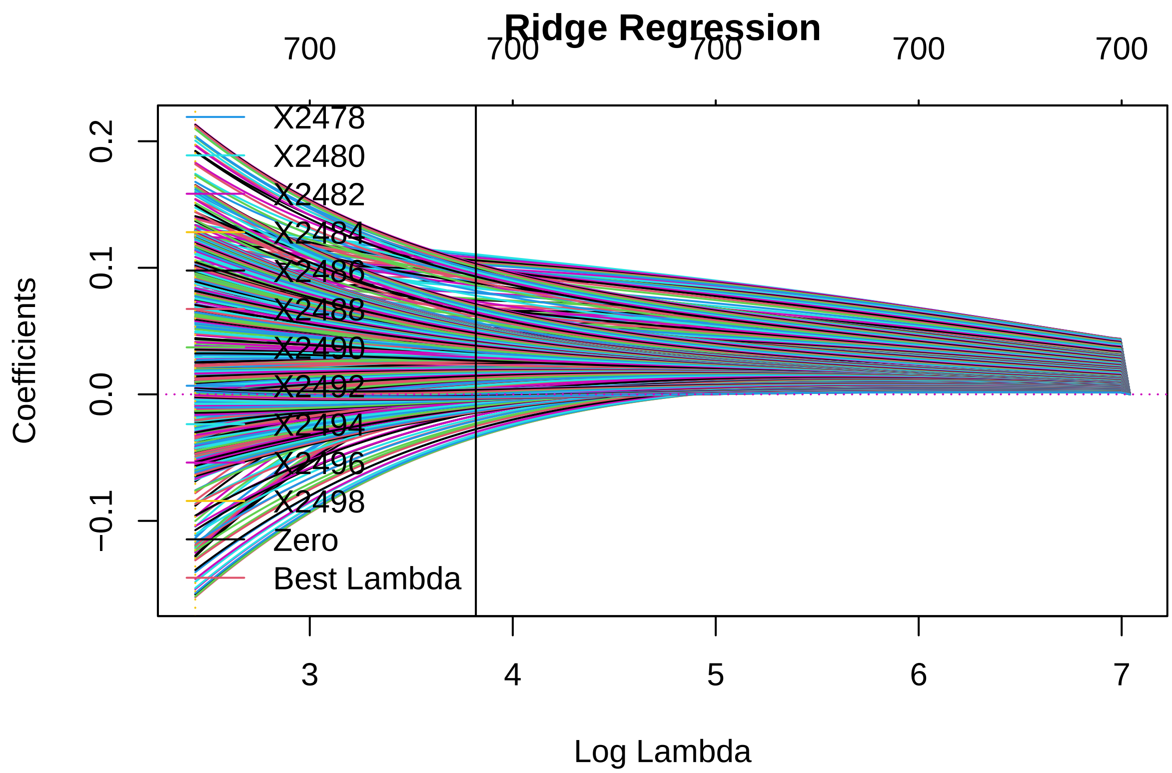

```r
print(cv_ridge_model)
```

```
##
## Call:  cv.glmnet(x = as.matrix(X), y = y, alpha = 0, standardize = TRUE)
##
## Measure: Mean-Squared Error
##
##     Lambda Index Measure      SE Nonzero
## min  11.42   100  0.5058 0.09882     700
## 1se  20.91    87  0.6008 0.11187     700
```

```r
best_lambda <- cv_ridge_model$lambda.min
best_lambda_ridge_model <- best_lambda
print(paste("Best lambda :", best_lambda))
```

```
## [1] "Best lambda : 11.4243191334971"
```

We can also plot the Regularization Path.



Now let's take a look at the coefficients of the best model we've managed to get.

```
final_ridge_model <- glmnet(as.matrix(X), y, alpha=0, lambda=best_lambda)

abs_coef <- abs(coef(final_ridge_model))
```

```
## [1] 2.232542e-05
```

```
## [1] "Number of value higher than 10^-1 :  179"
```

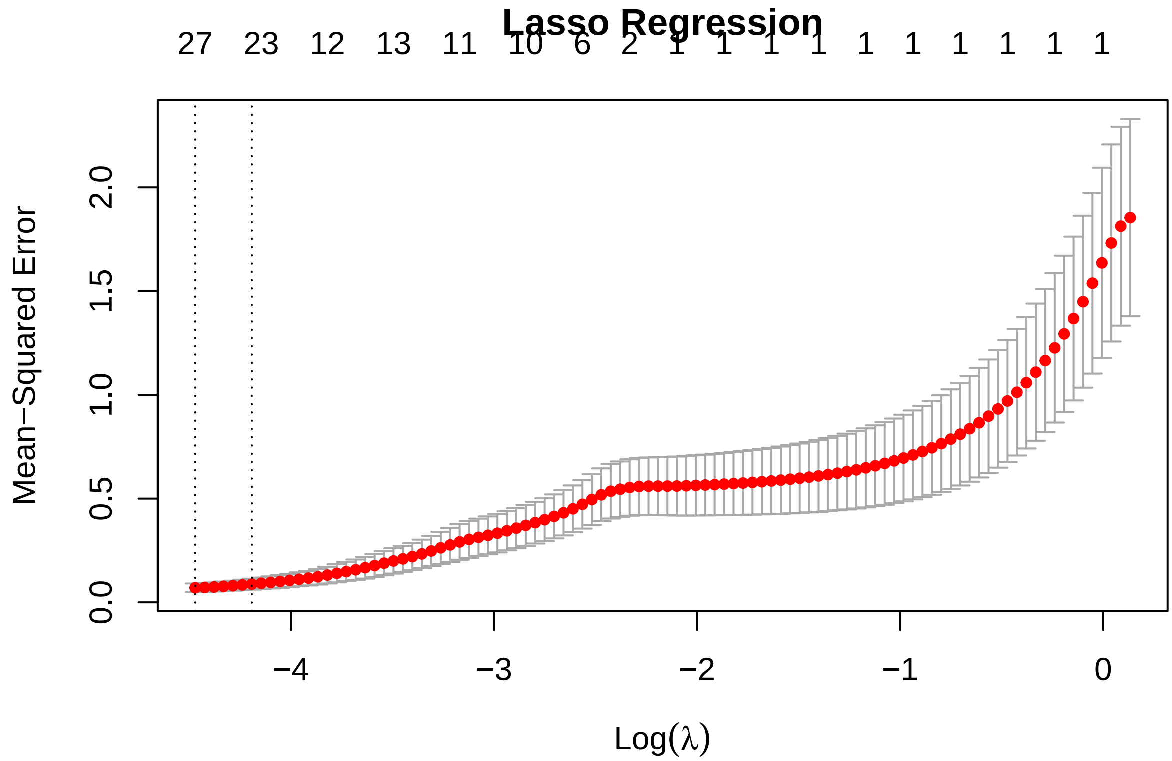
```
## [1] "Number of value higher than 10^-2 :  630"
```

```
## [1] "Number of value higher than 10^-3 :  694"
```

```
## [1] "Number of value higher than 10^-4 :  700"
```

We can see that the majority of the coefficients are lower than 10^{-1}. Then, we could think that a lot of our co-variables are useless to predict the target variable.

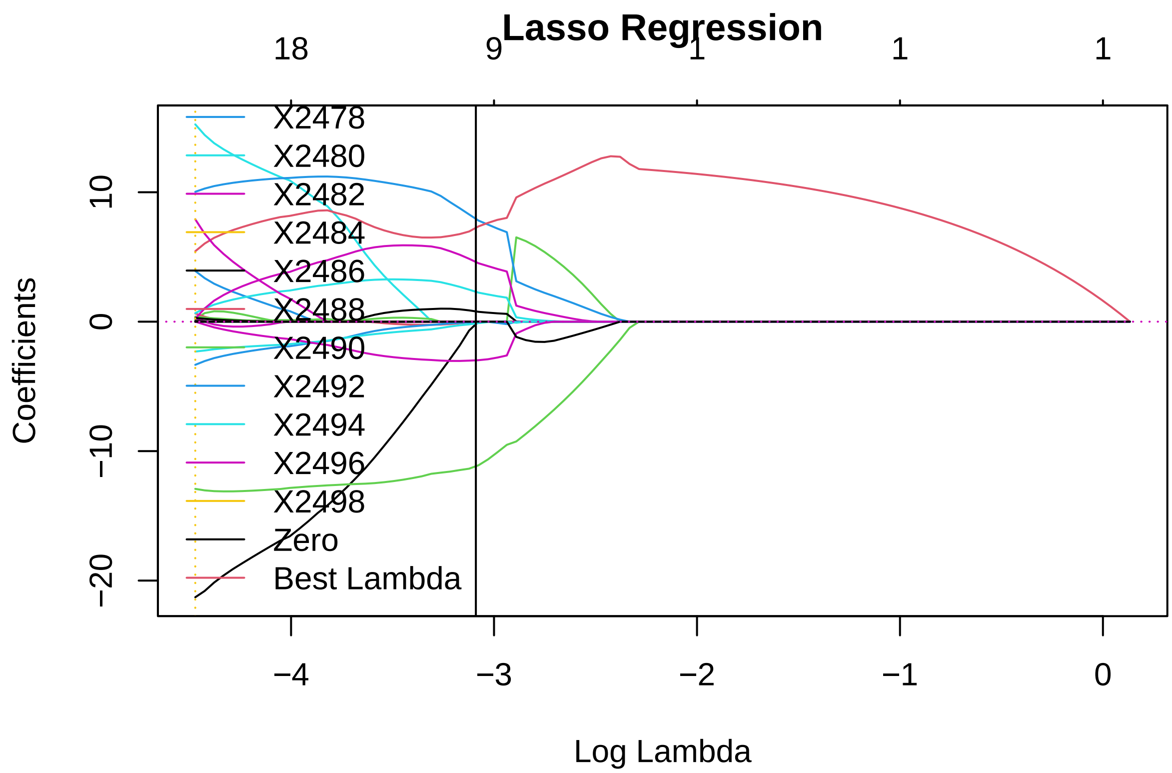
Let's do a Lasso regression to see if there are less co-variables that are actually useful to predict the fat :




```
##
## Call:  cv.glmnet(x = as.matrix(X), y = y, alpha = 1)
##
## Measure: Mean-Squared Error
##
##      Lambda Index Measure      SE Nonzero
## min 0.01142   100 0.07035 0.02055      27
## 1se 0.01510    94 0.08768 0.02637      23
```

Again, here's the regularization path :

```
plot(cv_lasso_model$glmnet.fit, xvar = "lambda", main="Lasso Regression")
abline(h = 0, col = 6, lty = 3)
abline(v = log(best_lambda_lasso), col = 7, lty = 3)
legend("bottomleft", legend = c(colnames(X), "Zero", "Best Lambda"), col = 1:7, lty = 1)
```




```
print(log(best_lambda_lasso))
```

```
## [1] -4.472011
```

Now, let's see how co-variables we have left :

```
##
## Call:  glmnet(x = as.matrix(X), y = y, alpha = 1, lambda = best_lambda_lasso)
##
##   Df  %Dev  Lambda
## 1 31 98.09 0.01142
```

We can see that our model is pretty accurate (deviance of 98.09%) with only 31 co-variables used among the 700 existant.

Actually, we've shown that even less co-variables are useless than what we thought.

Now let's try to split our dataset into train and test dataset:

Ridge :

```
# We split into 2 dataframe randomly
indice_train <- sample(1:nrow(cookies_data), 0.8 * nrow(cookies_data))
train_data <- cookies_data[indice_train, ]
test_data <- cookies_data[-indice_train, ]
```

```r
# We define X & y for both dataframe
y_train <- train_data[, 1]
X_train <- train_data[, -1]
y_test <- test_data[, 1]
X_test <- test_data[, -1]

# We train the model with the best value for lambda
cv_ridge_model <- cv.glmnet(as.matrix(X_train), y_train, alpha = 0, grouped = FALSE)
best_lambda_ridge <- cv_ridge_model$lambda.min
ridge_model <- glmnet(as.matrix(X_train), y_train, lambda = best_lambda_ridge, alpha = 0)

predictions_ridge <- predict(ridge_model, s = best_lambda_ridge, newx = as.matrix(X_test))

error_ridge <- sqrt(mean((predictions_ridge - y_test)^2))
print(paste("RMSE Ridge :", round(error_ridge, 2)))
```

```
## [1] "RMSE Ridge : 0.56"
```

Lasso :

```r
# We split into 2 dataframe randomly
indice_train <- sample(1:nrow(cookies_data), 0.8 * nrow(cookies_data))
train_data <- cookies_data[indice_train, ]
test_data <- cookies_data[-indice_train, ]

# We define X & y for both dataframe
y_train <- train_data[, 1]
X_train <- train_data[, -1]
y_test <- test_data[, 1]
X_test <- test_data[, -1]

# We train the model with the best value for lambda
cv_lasso_model <- cv.glmnet(as.matrix(X_train), y_train, alpha = 1, grouped = FALSE)
best_lambda <- cv_lasso_model$lambda.min
lasso_model <- glmnet(as.matrix(X_train), y_train, lambda = best_lambda, alpha = 1)

# We make prediction on the X_test
predictions_lasso <- predict(lasso_model, s = best_lambda, newx = as.matrix(X_test))

# We compute the RMSE
error_lasso  <- sqrt(mean((predictions_lasso - y_test)^2))
print(paste("RMSE :", round(error_lasso, 2)))
```

```
## [1] "RMSE : 0.23"
```

We see that the RMSE for the Lasso regression is way better than for the Ridge one.

Finally, let's verify if only 31 co-variables are useful for our prediction by using Step forward selection :

```
##
## Call:  glm(formula = fat ~ X1980 + X2128 + X1416 + X1716 + X2200 + X1428 +
##     X1976 + X2224 + X1978 + X2202 + X1468 + X2252 + X2018 + X2242 +
```

```
##     X2192 + X1564 + X2164 + X1718 + X2176 + X2216 + X1878 + X1144 +
##     X1566 + X2186 + X2244 + X2196 + X2346 + X1502 + X2246 + X1346 +
##     X2398, family = gaussian, data = cookies_data)
##
## Coefficients:
## (Intercept)        X1980        X2128        X1416        X1716        X2200
##   1.554e+01   -1.658e+02    2.891e+01    8.449e+01   -4.707e+02    5.583e+02
##       X1428        X1976        X2224        X1978        X2202        X1468
##  -8.572e+01   -3.285e+02   -9.854e+01    4.724e+02   -3.391e+02    3.288e+00
##       X2252        X2018        X2242        X2192        X1564        X2164
##   1.213e+02    4.101e+01   -1.714e+02    4.322e+01    1.270e+02   -1.991e+02
##       X1718        X2176        X2216        X1878        X1144        X1566
##   4.175e+02    1.244e+02   -6.102e+01    4.573e+01   -5.264e+00   -9.188e+01
##       X2186        X2244        X2196        X2346        X1502        X2246
##  -2.245e+01    1.672e+01   -4.137e+01    4.972e-01    2.369e-01    2.554e-01
##       X1346        X2398
##  -5.883e-03    3.412e-06
##
## Degrees of Freedom: 31 Total (i.e. Null);  0 Residual
## Null Deviance:       56.37
## Residual Deviance: 1.382e-23     AIC: -1638
```

We see that we also get only 31 degrees of freedom, the same as the lasso regression.