```
> library(dplyr)
> library(gamlr)
> # Read the dataset
> data <-
read.csv("C:/Users/aksab/Downloads/cleaned_weekly_2017_2019.csv")
> # Compute weekly returns for GSPC and RUA
> data <- data %>%
+   mutate(
+     GSPC_Returns = ((Adj.Close_.GSPC - lag(Adj.Close_.GSPC)) /
lag(Adj.Close_.GSPC)) * 100,
+     RUA_Returns = ((Adj.Close_.RUA - lag(Adj.Close_.RUA)) /
lag(Adj.Close_.RUA)) * 100
+   )
> # Compute the risk-free rate (convert annualized IRX to a weekly rate)
> data <- data %>%
+   mutate(
+     Weekly_Risk_Free_Rate = ((1 + Adj.Close_.IRX / 100)^(1/52) - 1) *
100, # used compounding interest formula for making it more realistic
+   )
> # Compute risk-adjusted returns for GSPC and RUA
> data <- data %>%
+   mutate(
+     GSPC_Risk_Adjusted = GSPC_Returns - Weekly_Risk_Free_Rate,
+     RUA_Risk_Adjusted = RUA_Returns - Weekly_Risk_Free_Rate
+   )
> # Create lagged variables for returns and music sentiment
> data <- data %>%
+   mutate(
+     Lagged_GSPC_Returns = lag(GSPC_Returns),
+     Lagged_RUA_Returns = lag(RUA_Returns),
+     Lagged_Music_Sentiment = lag(Music_Sentiment),
+     Lagged_GSPC_RA = lag(GSPC_Risk_Adjusted),
+     Lagged_RUA_RA = lag(RUA_Risk_Adjusted)
+   ) %>% na.omit()
> dim(data)
[1] 154  16
> model_gspc <- glm(GSPC_Returns ~ Music_Sentiment + Lagged_GSPC_Returns,
data = data)
> summary(model_gspc)

Call:
glm(formula = GSPC_Returns ~ Music_Sentiment + Lagged_GSPC_Returns,
    data = data)

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          0.28978    0.14900   1.945   0.0536 .
Music_Sentiment     -8.91936   10.22095  -0.873   0.3842
Lagged_GSPC_Returns -0.17644    0.08038  -2.195   0.0297 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.355249)

    Null deviance: 524.29  on 153  degrees of freedom
Residual deviance: 506.64  on 151  degrees of freedom
AIC: 628.42

Number of Fisher Scoring iterations: 2

> # Model 2: Russell 3000 Returns
> model_rua <- glm(RUA_Returns ~ Music_Sentiment + Lagged_RUA_Returns,
data = data)
> summary(model_rua)

Call:
glm(formula = RUA_Returns ~ Music_Sentiment + Lagged_RUA_Returns,
    data = data)
```

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         0.27205    0.14901   1.826   0.0699 .
Music_Sentiment   -10.45457   10.22519  -1.022   0.3082
Lagged_RUA_Returns  -0.15055    0.08063  -1.867   0.0638 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.360331)

    Null deviance: 521.56  on 153  degrees of freedom
Residual deviance: 507.41  on 151  degrees of freedom
AIC: 628.66

Number of Fisher Scoring iterations: 2

> # Model 3: S&P 500 Risk-Adjusted Returns
> model_gspc_ra <- glm(GSPC_Risk_Adjusted ~ Music_Sentiment +
Lagged_GSPC_RA, data = data)
> summary(model_gspc_ra)

Call:
glm(formula = GSPC_Risk_Adjusted ~ Music_Sentiment + Lagged_GSPC_RA,
    data = data)

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      0.25290    0.14875   1.700   0.0912 .
Music_Sentiment -8.93661   10.22568  -0.874   0.3835
Lagged_GSPC_RA  -0.17576    0.08039  -2.186   0.0303 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.358253)

    Null deviance: 524.63  on 153  degrees of freedom
Residual deviance: 507.10  on 151  degrees of freedom
AIC: 628.56

Number of Fisher Scoring iterations: 2

> # Model 4: Russell 3000 Risk-Adjusted Returns
> model_rua_ra <- glm(RUA_Risk_Adjusted ~ Music_Sentiment + Lagged_RUA_RA,
data = data)
> summary(model_rua_ra)

Call:
glm(formula = RUA_Risk_Adjusted ~ Music_Sentiment + Lagged_RUA_RA,
    data = data)

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)       0.23598    0.14877   1.586    0.115
Music_Sentiment -10.47156   10.22971  -1.024    0.308
Lagged_RUA_RA    -0.14987    0.08064  -1.859    0.065 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.363203)

    Null deviance: 521.91  on 153  degrees of freedom
Residual deviance: 507.84  on 151  degrees of freedom
AIC: 628.79

Number of Fisher Scoring iterations: 2

> # Run regression models for Panel B: Lagged Effects
> # Model 5: S&P 500 Returns with Lagged Music Sentiment
```

```
> model_lag_gspc <- glm(GSPC_Returns ~ Lagged_Music_Sentiment +
Lagged_GSPC_Returns, data = data)
> summary(model_lag_gspc)

Call:
glm(formula = GSPC_Returns ~ Lagged_Music_Sentiment + Lagged_GSPC_Returns,
    data = data)

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)             0.2854     0.1493   1.912   0.0578 .
Lagged_Music_Sentiment -1.7980    10.2475  -0.175   0.8610
Lagged_GSPC_Returns    -0.1704     0.0803  -2.122   0.0355 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.371483)

    Null deviance: 524.29  on 153  degrees of freedom
Residual deviance: 509.09  on 151  degrees of freedom
AIC: 629.17

Number of Fisher Scoring iterations: 2

> # Model 6: Russell 3000 Returns with Lagged Music Sentiment
> model_lag_rua <- glm(RUA_Returns ~ Lagged_Music_Sentiment +
Lagged_RUA_Returns, data = data)
> summary(model_lag_rua)

Call:
glm(formula = RUA_Returns ~ Lagged_Music_Sentiment + Lagged_RUA_Returns,
    data = data)

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.26706    0.14946   1.787   0.0760 .
Lagged_Music_Sentiment -1.97045   10.27400  -0.192   0.8482
Lagged_RUA_Returns     -0.14389    0.08072  -1.782   0.0767 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.382771)

    Null deviance: 521.56  on 153  degrees of freedom
Residual deviance: 510.80  on 151  degrees of freedom
AIC: 629.68

Number of Fisher Scoring iterations: 2

> # Model 7: S&P 500 Risk-Adjusted Returns with Lagged Music Sentiment
> model_lag_gspc_ra <- glm(GSPC_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_GSPC_RA, data = data)
> summary(model_lag_gspc_ra)

Call:
glm(formula = GSPC_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_GSPC_RA,
    data = data)

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)            0.24872    0.14906   1.669   0.0973 .
Lagged_Music_Sentiment -1.81739   10.25226  -0.177   0.8595
Lagged_GSPC_RA         -0.16968    0.08031  -2.113   0.0363 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.374537)
```

```
    Null deviance: 524.63  on 153  degrees of freedom
Residual deviance: 509.56  on 151  degrees of freedom
AIC: 629.31

Number of Fisher Scoring iterations: 2

> # Model 8: Russell 3000 Risk-Adjusted Returns with Lagged Music
Sentiment
> model_lag_rua_ra <- glm(RUA_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_RUA_RA, data = data)
> summary(model_lag_rua_ra)

Call:
glm(formula = RUA_Risk_Adjusted ~ Lagged_Music_Sentiment + Lagged_RUA_RA,
    data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)             0.23119    0.14923   1.549   0.1234
Lagged_Music_Sentiment -1.98796   10.27855  -0.193   0.8469
Lagged_RUA_RA          -0.14319    0.08073  -1.774   0.0781 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.385703)

    Null deviance: 521.91  on 153  degrees of freedom
Residual deviance: 511.24  on 151  degrees of freedom
AIC: 629.82

Number of Fisher Scoring iterations: 2

> # Extract standard errors and store them in a list
> standard_errors_list <- list(
+   model_gspc_se = summary(model_gspc)$coefficients[, "Std. Error"],
+   model_rua_se = summary(model_rua)$coefficients[, "Std. Error"],
+   model_gspc_ra_se = summary(model_gspc_ra)$coefficients[, "Std.
Error"],
+   model_rua_ra_se = summary(model_rua_ra)$coefficients[, "Std. Error"],
+   model_lag_gspc_se = summary(model_lag_gspc)$coefficients[, "Std.
Error"],
+   model_lag_rua_se = summary(model_lag_rua)$coefficients[, "Std.
Error"],
+   model_lag_gspc_ra_se = summary(model_lag_gspc_ra)$coefficients[, "Std.
Error"],
+   model_lag_rua_ra_se = summary(model_lag_rua_ra)$coefficients[, "Std.
Error"]
+ )
> # Print the list of standard errors
> print(standard_errors_list)
$model_gspc_se
        (Intercept)     Music_Sentiment Lagged_GSPC_Returns
          0.1489970          10.2209543           0.0803763

$model_rua_se
       (Intercept)    Music_Sentiment Lagged_RUA_Returns
         0.14901259        10.22519458         0.08062682

$model_gspc_ra_se
    (Intercept) Music_Sentiment  Lagged_GSPC_RA
     0.14874883     10.22567546      0.08038769

$model_rua_ra_se
    (Intercept) Music_Sentiment   Lagged_RUA_RA
     0.14877281     10.22970508      0.08063615

$model_lag_gspc_se
           (Intercept) Lagged_Music_Sentiment    Lagged_GSPC_Returns
            0.14929787            10.24753222             0.08029944
```

```
$model_lag_rua_se
           (Intercept) Lagged_Music_Sentiment     Lagged_RUA_Returns
             0.14946249             10.27399586              0.08072425

$model_lag_gspc_ra_se
           (Intercept) Lagged_Music_Sentiment         Lagged_GSPC_RA
             0.14905559             10.25225575              0.08031074

$model_lag_rua_ra_se
           (Intercept) Lagged_Music_Sentiment          Lagged_RUA_RA
             0.14922672             10.27855194              0.08073378


> #Bootstrapping
> library(boot)
> library(sandwich)
Warning message:
package 'sandwich' was built under R version 4.3.3
> library(lmtest)
Loading required package: zoo

Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric

> library(parallel)
> detectCores()
[1] 8
> # Define the model formulas and coefficient names in a list
> model_formulas <- list(
+   GSPC_Contemp = glm(GSPC_Returns ~ Music_Sentiment +
Lagged_GSPC_Returns, data = data),
+   RUA_Contemp = glm(RUA_Returns ~ Music_Sentiment + Lagged_RUA_Returns,
data = data),
+   GSPC_RA_Contemp = glm(GSPC_Risk_Adjusted ~ Music_Sentiment +
Lagged_GSPC_RA, data = data),
+   RUA_RA_Contemp = glm(RUA_Risk_Adjusted ~ Music_Sentiment +
Lagged_RUA_RA, data = data),
+   GSPC_Lagged = glm(GSPC_Returns ~ Lagged_Music_Sentiment +
Lagged_GSPC_Returns, data = data),
+   RUA_Lagged = glm(RUA_Returns ~ Lagged_Music_Sentiment +
Lagged_RUA_Returns, data = data),
+   GSPC_RA_Lagged = glm(GSPC_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_GSPC_RA, data = data),
+   RUA_RA_Lagged = glm(RUA_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_RUA_RA, data = data)
+ )
> # Function to extract the coefficient
> getBeta <- function(data, indices, formula, coef_name){
+   model <- glm(formula = formula, data = data[indices, ])
+   return(model$coef[coef_name])
+ }
> # Empty list to store bootstrapping results
> bootstrap_results <- list()
>set.seed(44)
> # Loop through each model and perform bootstrapping
> for (name in names(model_formulas)) {
+   formula <- model_formulas[[name]]
+   coef_name <- if (grepl("Lagged", name)) "Lagged_Music_Sentiment" else
"Music_Sentiment"
+
+   # Perform the bootstrapping
+   bootstrap_results[[name]] <- boot(data, getBeta, R = 2000, formula =
formula, coef_name = coef_name,
+                                     parallel = "snow", ncpus =
detectCores())
```

```
+ }
> # Print the results
> print(bootstrap_results)
$GSPC_Contemp

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -8.919361 -4.831239    15.71031

$RUA_Contemp

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -10.45457 -5.407749    15.74979

$GSPC_RA_Contemp

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -8.936609 -4.681999     15.4046

$RUA_RA_Contemp

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -10.47156 -5.177806    15.84201

$GSPC_Lagged

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())
```

```
Bootstrap Statistics :
     original    bias    std. error
t1* -1.797976 -3.257163    11.23782

$RUA_Lagged

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -1.970451 -3.377556    11.17908

$GSPC_RA_Lagged

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
     original    bias    std. error
t1* -1.817387 -3.070022    10.82511

$RUA_RA_Lagged

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = data, statistic = getBeta, R = 2000, formula = formula,
    coef_name = coef_name, parallel = "snow", ncpus = detectCores())


Bootstrap Statistics :
    original    bias    std. error
t1* -1.98796 -2.873702    10.99513

> models <- list(
+   GSPC_Contemp = model_gspc,
+   RUA_Contemp = model_rua,
+   GSPC_RA_Contemp = model_gspc_ra,
+   RUA_RA_Contemp = model_rua_ra,
+   GSPC_Lagged = model_lag_gspc,
+   RUA_Lagged = model_lag_rua,
+   GSPC_RA_Lagged = model_lag_gspc_ra,
+   RUA_RA_Lagged = model_lag_rua_ra
+ )
> # Empty list
> hc_results <- list()
> # Loop through each model and compute HC0 standard errors
> for (name in names(models)) {
+   model <- models[[name]]
+
+   # Compute heteroskedasticity-consistent covariance matrix
+   VHC <- vcovHC(model, type = "HC0")
+
+   # Test coefficients with robust standard errors
+   hcstats <- coeftest(model, vcov = VHC)
+
```

```
+   # Extracting the results for 'Music_Sentiment' or
'Lagged_Music_Sentiment'
+   coef_name <- if (grepl("Lagged", name)) "Lagged_Music_Sentiment" else
"Music_Sentiment"
+
+   # Store rounded results in the list
+   hc_results[[name]] <- round(hcstats[coef_name,], 5)
+ }
> # Display results
> print(hc_results)
$GSPC_Contemp
  Estimate Std. Error    z value   Pr(>|z|)
  -8.91936   13.64994   -0.65344    0.51348

$RUA_Contemp
  Estimate Std. Error    z value   Pr(>|z|)
 -10.45457   13.87782   -0.75333    0.45125

$GSPC_RA_Contemp
  Estimate Std. Error    z value   Pr(>|z|)
  -8.93661   13.67312   -0.65359    0.51338

$RUA_RA_Contemp
  Estimate Std. Error    z value   Pr(>|z|)
 -10.47156   13.90015   -0.75334    0.45124

$GSPC_Lagged
  Estimate Std. Error    z value   Pr(>|z|)
  -1.79798    9.43220   -0.19062    0.84882

$RUA_Lagged
  Estimate Std. Error    z value   Pr(>|z|)
  -1.97045    9.40695   -0.20947    0.83408

$GSPC_RA_Lagged
  Estimate Std. Error    z value   Pr(>|z|)
  -1.81739    9.44853   -0.19235    0.84747

$RUA_RA_Lagged
  Estimate Std. Error    z value   Pr(>|z|)
  -1.98796    9.42257   -0.21098    0.83290



> # LASSO
>
> library(gamlr)
> library(Matrix)
> x_GSPC = model.matrix(GSPC_Returns ~ Music_Sentiment +
Lagged_GSPC_Returns, data = data )[,-1]
> y_GSPC = data$GSPC_Returns
> cv_fit_GSPC <- cv.gamlr(x=x_GSPC, y=y_GSPC,lmr=1e-4, standardize =T)
> plot(cv_fit_GSPC)
> (lambda_min_GSPC <- cv_fit_GSPC$lambda.min)
[1] 0.05865118
> (lasso_coef_GSPC <- coef(cv_fit_GSPC, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                          seg19
intercept            0.2794998
Music_Sentiment     -4.4428067
Lagged_GSPC_Returns -0.1412406

> (lasso_coef_GSPC_1 <- coef(cv_fit_GSPC, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                         seg1
intercept           0.243428
Music_Sentiment     .
Lagged_GSPC_Returns .
```

```
> #RUA
> x_RUA = model.matrix(RUA_Returns ~ Music_Sentiment + Lagged_RUA_Returns,
data = data)[, -1]
> y_RUA = data$RUA_Returns
> cv_fit_RUA <- cv.gamlr(x=x_RUA, y=y_RUA, lmr=1e-4, standardize=TRUE)
> plot(cv_fit_RUA)
> (lambda_min_RUA <- cv_fit_RUA$lambda.min)
[1] 0.2628563
> (lasso_coef_RUA <- coef(cv_fit_RUA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                       seg1
intercept          0.2328423
Music_Sentiment    .
Lagged_RUA_Returns .
> (lasso_coef_RUA_1 <- coef(cv_fit_RUA, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                       seg1
intercept          0.2328423
Music_Sentiment    .
Lagged_RUA_Returns .
> x_GSPC_RA = model.matrix(GSPC_Risk_Adjusted ~ Music_Sentiment +
Lagged_GSPC_RA, data = data)[, -1]
> y_GSPC_RA = data$GSPC_Risk_Adjusted
> cv_fit_GSPC_RA <- cv.gamlr(x=x_GSPC_RA, y=y_GSPC_RA, lmr=1e-4,
standardize=TRUE)
> plot(cv_fit_GSPC_RA)
> (lambda_min_GSPC_RA <- cv_fit_GSPC_RA$lambda.min)
[1] 0.3118075
> (lasso_coef_GSPC_RA <- coef(cv_fit_GSPC_RA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                      seg1
intercept       0.2121753
Music_Sentiment .
Lagged_GSPC_RA  .
> (lasso_coef_GSPC_RA_1 <- coef(cv_fit_GSPC_RA, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                      seg1
intercept       0.2121753
Music_Sentiment .
Lagged_GSPC_RA  .
> x_RUA_RA = model.matrix(RUA_Risk_Adjusted ~ Music_Sentiment +
Lagged_RUA_RA, data = data)[, -1]
> y_RUA_RA = data$RUA_Risk_Adjusted
> cv_fit_RUA_RA <- cv.gamlr(x=x_RUA_RA, y=y_RUA_RA, lmr=1e-4,
standardize=TRUE)
> plot(cv_fit_RUA_RA)
> (lambda_min_RUA_RA <- cv_fit_RUA_RA$lambda.min)
[1] 0.1243036
> (lasso_coef_RUA_RA <- coef(cv_fit_RUA_RA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                      seg9
intercept        0.21725051
Music_Sentiment -1.01977600
Lagged_RUA_RA   -0.07536483
> x_lag_GSPC = model.matrix(GSPC_Returns ~ Lagged_Music_Sentiment +
Lagged_GSPC_Returns, data = data)[, -1]
> y_lag_GSPC = data$GSPC_Returns
> cv_fit_lag_GSPC <- cv.gamlr(x=x_lag_GSPC, y=y_lag_GSPC, lmr=1e-4,
standardize=TRUE)
> plot(cv_fit_lag_GSPC)
> (lambda_min_lag_GSPC <- cv_fit_lag_GSPC$lambda.min)
[1] 0.04436746
> (lasso_coef_lag_GSPC <- coef(cv_fit_lag_GSPC, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                           seg22
intercept                0.2788516
Lagged_Music_Sentiment   .
Lagged_GSPC_Returns     -0.1455915
> (lasso_coef_lag_GSPC_1 <- coef(cv_fit_lag_GSPC, s = "1se"))
```

```
3 x 1 sparse Matrix of class "dgCMatrix"
                              seg1
intercept                0.243428
Lagged_Music_Sentiment   .
Lagged_GSPC_Returns      .
> x_lag_RUA = model.matrix(RUA_Returns ~ Lagged_Music_Sentiment +
Lagged_RUA_Returns, data = data)[, -1]
> y_lag_RUA = data$RUA_Returns
> cv_fit_lag_RUA <- cv.gamlr(x=x_lag_RUA, y=y_lag_RUA, lmr=1e-4,
standardize=TRUE)
> plot(cv_fit_lag_RUA)
> (lambda_min_lag_RUA <- cv_fit_lag_RUA$lambda.min)
[1] 0.1370535
> (lasso_coef_lag_RUA <- coef(cv_fit_lag_RUA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                               seg8
intercept                0.24882159
Lagged_Music_Sentiment   .
Lagged_RUA_Returns      -0.06836488
> (lasso_coef_lag_RUA_1 <- coef(cv_fit_lag_RUA, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                              seg1
intercept                0.2328423
Lagged_Music_Sentiment   .
Lagged_RUA_Returns       .
> x_lag_GSPC_RA = model.matrix(GSPC_Risk_Adjusted ~ Lagged_Music_Sentiment
+ Lagged_GSPC_RA, data = data)[, -1]
> y_lag_GSPC_RA = data$GSPC_Risk_Adjusted
> cv_fit_lag_GSPC_RA <- cv.gamlr(x=x_lag_GSPC_RA, y=y_lag_GSPC_RA, lmr=1e-
4, standardize=TRUE)
> plot(cv_fit_lag_GSPC_RA)
> (lambda_min_lag_GSPC_RA <- cv_fit_lag_GSPC_RA$lambda.min)
[1] 0.134974
> (lasso_coef_lag_GSPC_RA <- coef(cv_fit_lag_GSPC_RA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                              seg10
intercept                0.23250326
Lagged_Music_Sentiment   .
Lagged_GSPC_RA          -0.09580692

> (lasso_coef_lag_GSPC_RA_1 <- coef(cv_fit_lag_GSPC_RA, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                              seg1
intercept                0.2121753
Lagged_Music_Sentiment   .
Lagged_GSPC_RA           .


> x_lag_RUA_RA = model.matrix(RUA_Risk_Adjusted ~ Lagged_Music_Sentiment +
Lagged_RUA_RA, data = data)[, -1]
> y_lag_RUA_RA = data$RUA_Risk_Adjusted
> cv_fit_lag_RUA_RA <- cv.gamlr(x=x_lag_RUA_RA, y=y_lag_RUA_RA, lmr=1e-4,
standardize=TRUE)
> plot(cv_fit_lag_RUA_RA)
> (lambda_min_lag_RUA_RA <- cv_fit_lag_RUA_RA$lambda.min)
[1] 0.164322
> (lasso_coef_lag_RUA_RA <- coef(cv_fit_lag_RUA_RA, s = "min"))
3 x 1 sparse Matrix of class "dgCMatrix"
                               seg6
intercept                0.21230163
Lagged_Music_Sentiment   .
Lagged_RUA_RA           -0.05287207

> (lasso_coef_lag_RUA_RA_1 <- coef(cv_fit_lag_RUA_RA, s = "1se"))
3 x 1 sparse Matrix of class "dgCMatrix"
                              seg1
intercept                0.2015895
Lagged_Music_Sentiment   .
Lagged_RUA_RA            .
```

```
> # Robustness checks
> model_GSPC_rc_with_control <- glm(GSPC_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment + Lagged_GSPC_Returns, data = data)
> summary(model_GSPC_rc_with_control)

Call:
glm(formula = GSPC_Returns ~ Music_Sentiment + Lagged_Music_Sentiment +
    Lagged_GSPC_Returns, data = data)

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.29080    0.14953   1.945   0.0537 .
Music_Sentiment         -9.06618   10.27304  -0.883   0.3789
Lagged_Music_Sentiment -2.36179   10.27493  -0.230   0.8185
Lagged_GSPC_Returns     -0.17752    0.08077  -2.198   0.0295 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.376428)

    Null deviance: 524.29  on 153  degrees of freedom
Residual deviance: 506.46  on 150  degrees of freedom
AIC: 630.37

Number of Fisher Scoring iterations: 2

> model_GSPC_rc_without_control <- glm(GSPC_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment, data = data)
> summary(model_GSPC_rc_without_control)

Call:
glm(formula = GSPC_Returns ~ Music_Sentiment + Lagged_Music_Sentiment,
    data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.2464     0.1500   1.642    0.103
Music_Sentiment         -6.8015    10.3501  -0.657    0.512
Lagged_Music_Sentiment  -1.0518    10.3869  -0.101    0.919

(Dispersion parameter for gaussian family taken to be 3.462093)

    Null deviance: 524.29  on 153  degrees of freedom
Residual deviance: 522.78  on 151  degrees of freedom
AIC: 633.25

Number of Fisher Scoring iterations: 2

> model_RUA_rc_with_control <- glm(RUA_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment + Lagged_RUA_Returns, data = data)
> summary(model_RUA_rc_with_control)

Call:
glm(formula = RUA_Returns ~ Music_Sentiment + Lagged_Music_Sentiment +
    Lagged_RUA_Returns, data = data)

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.27325    0.14955   1.827   0.0697 .
Music_Sentiment        -10.62219   10.27767  -1.034   0.3030
Lagged_Music_Sentiment  -2.64541   10.29242  -0.257   0.7975
Lagged_RUA_Returns      -0.15207    0.08109  -1.875   0.0627 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.381244)

    Null deviance: 521.56  on 153  degrees of freedom
```

```
Residual deviance: 507.19  on 150  degrees of freedom
AIC: 630.59

Number of Fisher Scoring iterations: 2

> model_RUA_rc_without_control <- glm(RUA_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment, data = data)
> summary(model_RUA_rc_without_control)

Call:
glm(formula = RUA_Returns ~ Music_Sentiment + Lagged_Music_Sentiment,
    data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.2366     0.1495   1.582    0.116
Music_Sentiment         -8.7396    10.3134  -0.847    0.398
Lagged_Music_Sentiment  -1.2359    10.3501  -0.119    0.905

(Dispersion parameter for gaussian family taken to be 3.437597)

    Null deviance: 521.56  on 153  degrees of freedom
Residual deviance: 519.08  on 151  degrees of freedom
AIC: 632.16

Number of Fisher Scoring iterations: 2

> model_GSPC_ra_rc_with_control <- glm(GSPC_Risk_Adjusted ~
Music_Sentiment + Lagged_Music_Sentiment + Lagged_GSPC_RA, data = data)
> summary(model_GSPC_ra_rc_with_control)

Call:
glm(formula = GSPC_Risk_Adjusted ~ Music_Sentiment +
Lagged_Music_Sentiment +
    Lagged_GSPC_RA, data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)             0.25389    0.14928   1.701   0.0911 .
Music_Sentiment        -9.08478   10.27777  -0.884   0.3782
Lagged_Music_Sentiment -2.38257   10.27959  -0.232   0.8170
Lagged_GSPC_RA         -0.17685    0.08078  -2.189   0.0301 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.379431)

    Null deviance: 524.63  on 153  degrees of freedom
Residual deviance: 506.91  on 150  degrees of freedom
AIC: 630.51

Number of Fisher Scoring iterations: 2

> model_GSPC_ra_rc_without_control <- glm(GSPC_Risk_Adjusted ~
Music_Sentiment + Lagged_Music_Sentiment, data = data)
> summary(model_GSPC_ra_rc_without_control)

Call:
glm(formula = GSPC_Risk_Adjusted ~ Music_Sentiment +
Lagged_Music_Sentiment,
    data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.2151     0.1501   1.433    0.154
Music_Sentiment         -6.8244    10.3534  -0.659    0.511
Lagged_Music_Sentiment  -1.0734    10.3903  -0.103    0.918

(Dispersion parameter for gaussian family taken to be 3.464327)
```

```
    Null deviance: 524.63  on 153  degrees of freedom
Residual deviance: 523.11  on 151  degrees of freedom
AIC: 633.35

Number of Fisher Scoring iterations: 2

> model_RUA_ra_rc_with_control <- glm(RUA_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment + Lagged_RUA_RA, data = data)
> summary(model_RUA_ra_rc_with_control)

Call:
glm(formula = RUA_Returns ~ Music_Sentiment + Lagged_Music_Sentiment +
    Lagged_RUA_RA, data = data)

Coefficients:
                        Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.26844    0.14925   1.799   0.0741 .
Music_Sentiment        -10.62102   10.27838  -1.033   0.3031
Lagged_Music_Sentiment  -2.64536   10.29308  -0.257   0.7975
Lagged_RUA_RA           -0.15168    0.08107  -1.871   0.0633 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 3.381601)

    Null deviance: 521.56  on 153  degrees of freedom
Residual deviance: 507.24  on 150  degrees of freedom
AIC: 630.61

Number of Fisher Scoring iterations: 2

> model_RUA_ra_rc_without_control <- glm(RUA_Returns ~ Music_Sentiment +
Lagged_Music_Sentiment, data = data)
> summary(model_RUA_ra_rc_without_control)

Call:
glm(formula = RUA_Returns ~ Music_Sentiment + Lagged_Music_Sentiment,
    data = data)

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)              0.2366     0.1495   1.582    0.116
Music_Sentiment         -8.7396    10.3134  -0.847    0.398
Lagged_Music_Sentiment  -1.2359    10.3501  -0.119    0.905

(Dispersion parameter for gaussian family taken to be 3.437597)

    Null deviance: 521.56  on 153  degrees of freedom
Residual deviance: 519.08  on 151  degrees of freedom
AIC: 632.16

Number of Fisher Scoring iterations: 2

> outcome_var <- "GSPC_Returns"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_GSPC_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_GSPC_returns_1c <- doubleML(X,D,Y, nfold=10)
> summary(dml_GSPC_returns_1c)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
```

```
    Min      1Q  Median      3Q     Max
-7.4839 -0.5666  0.2725  1.0247  4.6794

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -8.441     10.258  -0.823    0.412

Residual standard error: 1.858 on 153 degrees of freedom
Multiple R-squared:  0.004406, Adjusted R-squared:  -0.002101
F-statistic: 0.6771 on 1 and 153 DF,  p-value: 0.4119

> outcome_var <- "RUA_Returns"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_RUA_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_RU_returns_1c <- doubleML(X,D,Y, nfold=10)
> summary(dml_RU_returns_1c)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.8445 -0.6882  0.2760  1.0527  4.6301

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -10.03      10.41  -0.964    0.337

Residual standard error: 1.893 on 153 degrees of freedom
Multiple R-squared:  0.006037, Adjusted R-squared:  -0.0004596
F-statistic: 0.9292 on 1 and 153 DF,  p-value: 0.3366

> outcome_var <- "GSPC_Risk_Adjusted"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_GSPC_RA")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_GSPC_RA_returns_1c <- doubleML(X,D,Y, nfold=10)
> summary(dml_GSPC_RA_returns_1c)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.4768 -0.6092  0.2106  0.9933  4.3899

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -9.049     10.196  -0.888    0.376

Residual standard error: 1.847 on 153 degrees of freedom
Multiple R-squared:  0.005122, Adjusted R-squared:  -0.001381
F-statistic: 0.7877 on 1 and 153 DF,  p-value: 0.3762

> outcome_var <- "RUA_Risk_Adjusted"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_RUA_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
```

```
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_RU_RA_returns_1c <- doubleML(X,D,Y, nfold=10)
> summary(dml_RU_RA_returns_1c)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.8380 -0.6616  0.2788  1.0469  4.5516

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -11.20      10.46  -1.071    0.286

Residual standard error: 1.9 on 153 degrees of freedom
Multiple R-squared:  0.007436, Adjusted R-squared:  0.0009491
F-statistic: 1.146 on 1 and 153 DF,  p-value: 0.286

> outcome_var <- "GSPC_Returns"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_Music_Sentiment", "Lagged_GSPC_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_GSPC_returns <- doubleML(X,D,Y, nfold=10)
> summary(dml_GSPC_returns)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.5703 -0.6066  0.2443  0.9681  4.9052

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -10.10      10.18  -0.991    0.323

Residual standard error: 1.864 on 153 degrees of freedom
Multiple R-squared:  0.006381, Adjusted R-squared:  -0.0001128
F-statistic: 0.9826 on 1 and 153 DF,  p-value: 0.3231

> # Setup for doubleML - choosing an outcome and treatment
> outcome_var <- "RUA_Returns"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_Music_Sentiment", "Lagged_RUA_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
>   # Using doubleML to estimate the effects
> dml_RU_returns <- doubleML(X,D,Y, nfold=10)
> summary(dml_RU_returns)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.7318 -0.6409  0.2038  0.9562  4.8697

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -9.764     10.371  -0.941    0.348
```

```
Residual standard error: 1.874 on 153 degrees of freedom
Multiple R-squared:  0.00576,  Adjusted R-squared:  -0.0007384
F-statistic: 0.8864 on 1 and 153 DF,  p-value: 0.3479

> outcome_var <- "GSPC_Risk_Adjusted"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_Music_Sentiment", "Lagged_GSPC_RA")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_GSPC_RA_returns <- doubleML(X,D,Y, nfold=10)
> summary(dml_GSPC_RA_returns)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.5591 -0.5973  0.2029  0.9630  4.6816

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -6.871     10.290  -0.668    0.505

Residual standard error: 1.863 on 153 degrees of freedom
Multiple R-squared:  0.002906, Adjusted R-squared:  -0.003611
F-statistic: 0.4459 on 1 and 153 DF,  p-value: 0.5053

> outcome_var <- "RUA_Risk_Adjusted"
> treatment_var <- "Music_Sentiment"
> covariates <- c("Lagged_Music_Sentiment", "Lagged_RUA_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_RU_RA_returns <- doubleML(X,D,Y, nfold=10)
> summary(dml_RU_RA_returns)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.6483 -0.6425  0.2797  1.0306  4.5038

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -9.535     10.291  -0.927    0.356

Residual standard error: 1.87 on 153 degrees of freedom
Multiple R-squared:  0.00558,  Adjusted R-squared:  -0.0009199
F-statistic: 0.8585 on 1 and 153 DF,  p-value: 0.3556

> outcome_var <- "GSPC_Returns"
> treatment_var <- "Lagged_Music_Sentiment"
> covariates <- c("Lagged_GSPC_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_lag_GSPC_returns <- doubleML(X,D,Y, nfold=10)
> # Print the summary of the double machine learning results
> summary(dml_lag_GSPC_returns)

Call:
```

```
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.5579 -0.5777  0.2515  1.0345  4.3845

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -1.786     10.222  -0.175    0.862

Residual standard error: 1.861 on 153 degrees of freedom
Multiple R-squared:  0.0001995,       Adjusted R-squared:  -0.006335
F-statistic: 0.03053 on 1 and 153 DF,  p-value: 0.8615

> # Setup for doubleML - choosing an outcome and treatment
> outcome_var <- "RUA_Returns"
> treatment_var <- "Lagged_Music_Sentiment"
> covariates <- c("Lagged_RUA_Returns")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_lag_RUA_Returns <- doubleML(X,D,Y, nfold=10)
> # Print the summary of the double machine learning results
> summary(dml_lag_RUA_Returns)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.7410 -0.6646  0.1818  1.1043  4.8142

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -2.165     10.466  -0.207    0.836

Residual standard error: 1.903 on 153 degrees of freedom
Multiple R-squared:  0.0002796,       Adjusted R-squared:  -0.006255
F-statistic: 0.04279 on 1 and 153 DF,  p-value: 0.8364

> # Setup for doubleML - choosing an outcome and treatment
> outcome_var <- "GSPC_Risk_Adjusted"
> treatment_var <- "Lagged_Music_Sentiment"
> covariates <- c("Lagged_GSPC_RA")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_lag_GSPC_RA <- doubleML(X,D,Y, nfold=10)
> # Print the summary of the double machine learning results
> summary(dml_lag_GSPC_RA)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.6292 -0.6442  0.2400  1.0455  5.0126

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -1.554     10.243  -0.152     0.88

Residual standard error: 1.868 on 153 degrees of freedom
Multiple R-squared:  0.0001504,       Adjusted R-squared:  -0.006385
F-statistic: 0.02301 on 1 and 153 DF,  p-value: 0.8796
```

```
> # Setup for doubleML - choosing an outcome and treatment
> outcome_var <- "RUA_Risk_Adjusted"
> treatment_var <- "Lagged_Music_Sentiment"
> covariates <- c("Lagged_RUA_RA")
> # Preparing the model matrix for doubleML
> X <- model.matrix(~ ., data = data[covariates])
> Y <- data[[outcome_var]]
> D <- data[[treatment_var]]
> # Using doubleML to estimate the effects
> dml_lag_RUA_RA <- doubleML(X,D,Y, nfold=10)
> # Print the summary of the double machine learning results
> summary(dml_lag_RUA_RA)

Call:
lm(formula = y ~ d - 1, x = TRUE, y = TRUE)

Residuals:
    Min      1Q  Median      3Q     Max
-7.7917 -0.6285  0.2281  1.0531  4.8578

Coefficients:
  Estimate Std. Error t value Pr(>|t|)
d   -1.636     10.367  -0.158    0.875

Residual standard error: 1.878 on 153 degrees of freedom
Multiple R-squared:  0.0001628,       Adjusted R-squared:  -0.006372
F-statistic: 0.02492 on 1 and 153 DF,  p-value: 0.8748
```