2022/4/5

Section 0

Package

```
library(MASS)
library(rcompanion)
library(randomForest)
library(caret)
library(pROC)
library(faraway)
library(dplyr)
library(ResourceSelection)
library(ggplot2)
library(pscl)
library(nortest)
library(insight)
library(cowplot)
library(glmnet)
```

##Function for Cox & Snell's R2

```
#' @title Cox & Snell's R2
#' @name r2_coxsnell
#'
#' @description
#' Calculates the pseudo-R2 value based on the proposal from *Cox & Sne
ll (1989)*.
#'
#' @param model Model with binary outcome.
#' @param ... Currently not used.
#'
#' @details
#' This index was proposed by *Cox & Snell (1989, pp. 208-9)* and,
#' apparently independently, by *Magee (1990)*; but had been suggested
#' earlier for binary response models by *Maddala (1983)*. However, thi
s
#' index achieves a maximum of less than 1 for discrete models (i.e. mo
dels
#' whose likelihood is a product of probabilities) which have a maximum
 of 1,
#' instead of densities, which can become infinite *(Nagelkerke, 1991)
*.
#'
#' @return A named vector with the R2 value.
#'
#' @examples
#' model <- glm(vs ~ wt + mpg, data = mtcars, family = "binomial")
#' r2_coxsnell(model)
```

```r
#'
#' @references
#' - Cox, D. R., Snell, E. J. (1989). Analysis of binary data (Vol. 3
2).
#'   Monographs on Statistics and Applied Probability.
#' - Magee, L. (1990). R 2 measures based on Wald and likelihood ratio
#'   joint significance tests. The American Statistician, 44(3), 250-25
3.
#' - Maddala, G. S. (1986). Limited-dependent and qualitative variables
 in
#'   econometrics (No. 3). Cambridge university press.
#' - Nagelkerke, N. J. (1991). A note on a general definition of the
#'   coefficient of determination. Biometrika, 78(3), 691-692.
#'
#' @export
r2_coxsnell <- function(model, ...) {
  UseMethod("r2_coxsnell")
}


# helper --------------------------


.r2_coxsnell <- function(model, l_base) {
  l_full <- insight::get_loglikelihood(model)
  G2 <- -2 * (l_base - l_full)

  # Is it still necessary?
  if (inherits(model, c("vglm", "vgam", "clm2"))) {
    n <- suppressWarnings(insight::n_obs(model))
  } else {
    n <- attr(l_full, "nobs")
    if (is.null(n)) n <- suppressWarnings(insight::n_obs(model, disaggr
egate = TRUE))
  }

  r2_coxsnell <- as.vector(1 - exp(-G2 / n))

  names(r2_coxsnell) <- "Cox & Snell's R2"
  r2_coxsnell
}



# r2-coxsnell based on model information --------------------------


#' @export
r2_coxsnell.glm <- function(model, verbose = TRUE, ...) {
```

```r
  if (is.null(info <- list(...)$model_info)) {
    info <- suppressWarnings(insight::model_info(model, verbose = FALS
E))
  }
  if (info$is_binomial && !info$is_bernoulli && class(model)[1] == "glm
") {
    if (verbose) {
      warning(insight::format_message("Can't calculate accurate R2 for
binomial models that are not Bernoulli models."), call. = FALSE)
    }
    return(NULL)
  } else {
    # if no deviance, return NA
    if (is.null(model$deviance)) {
      return(NULL)
    }
    r2_coxsnell <- (1 - exp((model$deviance - model$null.deviance) / in
sight::n_obs(model, disaggregate = TRUE)))
    names(r2_coxsnell) <- "Cox & Snell's R2"
    r2_coxsnell
  }
}

#' @export
r2_coxsnell.BBreg <- r2_coxsnell.glm

#' @export
r2_coxsnell.mclogit <- r2_coxsnell.glm

#' @export
r2_coxsnell.bife <- function(model, ...) {
  r2_coxsnell <- (1 - exp((model$deviance - model$null_deviance) / insi
ght::n_obs(model)))
  names(r2_coxsnell) <- "Cox & Snell's R2"
  r2_coxsnell
}



# mfx models ---------------------


#' @export
r2_coxsnell.logitmfx <- function(model, ...) {
  r2_coxsnell(model$fit, ...)
}

#' @export
r2_coxsnell.logitor <- r2_coxsnell.logitmfx
```

```r
#' @export
r2_coxsnell.poissonirr <- r2_coxsnell.logitmfx

#' @export
r2_coxsnell.poissonmfx <- r2_coxsnell.logitmfx

#' @export
r2_coxsnell.probit <- r2_coxsnell.logitmfx

#' @export
r2_coxsnell.negbinirr <- r2_coxsnell.logitmfx

#' @export
r2_coxsnell.negbinmfx <- r2_coxsnell.logitmfx




# r2-coxsnell based on loglik stored in model object ------------------
---------


#' @export
r2_coxsnell.coxph <- function(model, ...) {
  l_base <- model$loglik[1]
  .r2_coxsnell(model, l_base)
}

#' @export
r2_coxsnell.survreg <- r2_coxsnell.coxph

#' @export
r2_coxsnell.svycoxph <- function(model, ...) {
  l_base <- model$ll[1]
  .r2_coxsnell(model, l_base)
}



# r2-coxsnell based on loglik of null-model (update) ------------------
---------


#' @export
r2_coxsnell.multinom <- function(model, ...) {
  l_base <- insight::get_loglikelihood(stats::update(model, ~1, trace =
 FALSE))
```

```r
  .r2_coxsnell(model, l_base)
}

#' @export
r2_coxsnell.clm2 <- function(model, ...) {
  l_base <- insight::get_loglikelihood(stats::update(model, location =
~1, scale = ~1))
  .r2_coxsnell(model, l_base)
}

#' @export
r2_coxsnell.bayesx <- function(model, ...) {
  junk <- utils::capture.output(l_base <- insight::get_loglikelihood(st
ats::update(model, ~1)))
  .r2_coxsnell(model, l_base)
}

#' @export
r2_coxsnell.clm <- function(model, ...) {
  l_base <- insight::get_loglikelihood(stats::update(model, ~1))
  # if no loglik, return NA
  if (length(as.numeric(l_base)) == 0) {
    return(NULL)
  }
  .r2_coxsnell(model, l_base)
}

#' @export
r2_coxsnell.crch <- r2_coxsnell.clm

#' @export
r2_coxsnell.cpglm <- r2_coxsnell.clm

#' @export
r2_coxsnell.censReg <- r2_coxsnell.clm

#' @export
r2_coxsnell.truncreg <- r2_coxsnell.clm

#' @export
r2_coxsnell.polr <- r2_coxsnell.clm

#' @export
r2_coxsnell.glmx <- r2_coxsnell.clm

#' @export
r2_coxsnell.DirichletRegModel <- r2_coxsnell.clm
```

Section 2.1

CODE A

```
#Modified factor
#select train and test sample
#train set : test set = 7:3
set.seed(123)
seq <- rnorm(4920)
train <- Cleaned_Data[sample(nrow(Cleaned_Data),4920), ]
test <- Cleaned_Data[-sample(nrow(Cleaned_Data),4920), ]

#Modified factor
as.factor(Cleaned_Data$Industry)

train$UrbanRural <- as.factor(train$UrbanRural)
train$NewExist <- as.factor(train$NewExist)
train$MIS_Status <- ifelse(train$MIS_Status == 'CHGOFF', 1,0)
train$Industry <- as.factor(train$Industry)
train$MIS_Status <- as.factor(train$MIS_Status)

test$UrbanRural <- as.factor(test$UrbanRural)
test$NewExist <- as.factor(test$NewExist)
test$MIS_Status <- ifelse(test$MIS_Status == 'CHGOFF', 1,0)
test$Industry <- as.factor(test$Industry)
test$MIS_Status <- as.factor(test$MIS_Status)

#check each variables is in the appropriate form
summary(train)
```

Section 4.1

##CODE B

```
#CODE B
#Appendix C
summary(train)
summary(test)
```

##CODE C

```
#Figure 4.3
#left graph
x <- TTold$Term
h<-hist(x, breaks=100, col="brown", xlab="Disbursement",
   main="Frequency of Disbursion")
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
```

```
lines(xfit, yfit, col="blue", lwd=2)

#right graph
x <- train$Term
h<-hist(x, breaks=100, col="blue", xlab="Term",
   main="Fequcey of Term")
xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)
```

CODE D

```
#Figure 4.4
#Plot of NewExit
plot(train$NewExist,
     main = 'NewExit')

#Plot of UrbanRural
plot(train$UrbanRural,
     main = 'UrbanRural')

#Plot of Industry
plot(train$Industry,
     main = 'Industry')
```

#CODE E

```
#Table 4.1
xtabs(~ Industry + NewExist + UrbanRural, train)


Cor_Ind1 <- xtabs(~ NewExist + Industry, train)
cramerV(Cor_Ind1, ci = TRUE)

Cor_In2 <- xtabs(~ Industry + UrbanRural, train)
cramerV(Cor_In2, ci = TRUE)

Cor_In3 <- xtabs(~ NewExist + UrbanRural, train)
cramerV(Cor_In3, ci = TRUE)

Cor_Mis4 <- xtabs(~ MIS_Status + Industry, train)
cramerV(Cor_Mis4)

Cor_Mis5 <- xtabs(~ MIS_Status + NewExist, train)
cramerV(Cor_Mis5)

Cor_Mis6 <- xtabs(~ NewExist + UrbanRural, train)
cramerV(Cor_Mis6)
```

CODE F

```
#Pearson Correlation
#Figure 4.1
pairs(train)

panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...) {
    usr <- par("usr")
    on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    Cor <- abs(cor(x, y)) # Remove abs function if desired
    txt <- paste0(prefix, format(c(Cor, 0.123456789), digits = digits)
[1])
    if(missing(cex.cor)) {
        cex.cor <- 0.4 / strwidth(txt)
    }
    text(0.5, 0.5, txt,
         cex = 1 + cex.cor * Cor) # Resize the text by level of correla
tion
}

pairs(train,
      upper.panel = panel.cor,    # Correlation panel
      lower.panel = panel.smooth) # Smoothed regression lines
```

CODE G

```
#Figure 4.2

plot(train$RetainedJob,train$NoEmp,
     main = 'relationship between RetainedJob & Noemp')
boxplot(train$Term,train$MIS_Status,
     main = 'relationship between Term & MIS_Status')


filter = which(train$MIS_Status == 0)
filter2 = which(train$MIS_Status == 1)
TTnew <- train[filter,]
TTold <- train[filter2,]
summary(TTnew$Term)



xfit<-seq(min(x),max(x),length=40)
yfit<-dnorm(xfit,mean=mean(x),sd=sd(x))
yfit <- yfit*diff(h$mids[1:2])*length(x)
lines(xfit, yfit, col="blue", lwd=2)

plot(train$NoEmp,train$DisbursementGross,
     main = 'relationship between Disbursion & Noemp')
```

```
plot(train$RetainedJob,train$DisbursementGross,
     main = 'relationship between Disbursion & RetainedJob')

plot(train$CreateJob,train$RetainedJob)




#create scatterplot of x1 vs. y1
plot(train$NoEmp, train$RetainedJob, col='red', pch=19)


#add scatterplot of x2 vs. y2
points(train$NoEmp, train$DisbursementGross/10000, col='brown', pch=19)

#add scatterplot of x2 vs. y2
points(train$NoEmp, train$CreateJob, col='darkblue', pch=19)

#add legend
legend('topleft', legend=c('RetainedJob', 'CreateJob','Disbursemen'), p
ch=c(19, 19,19), col=c('red', 'blue','brown'))
```

#Section 4.2.1 #Code H

```
#Create Logistic Model
glm <- glm(MIS_Status ~ ., family = binomial, train)
sumary(glm)
beta <- coef(glm)
drop1(glm,test = 'Chi')
confint(glm_r)

linpred <- predict(glm)
predprob<- predict(glm, type = 'response')
head(predprob)

rawres <- train$MIS_Status - predprob
```

CODE I

```
#Figure 4.5

qqnorm(residuals(glm))
halfnorm(hatvalues(glm))

filter(train, hatvalues(glm) > 0.015) %>% select (Term,NoEmp,NewExist,C
reateJob,RetainedJob,UrbanRural,DisbursementGross,MIS_Status,Industry)
```

Stepwise

##CODE J

```
glm_r <- step(glm, trace = 0)
sumary(glm_r)

#Note that we have excluded variables with high correlation
```

##Using HL Test ##CODE K

```
#For glm
predict_y <- predict(glm,x_test,type = 'response')
predict_y <- unname(predict_y)
preY <- ifelse(predict_y >= 0.1968882   , 1,0)
hoslem.test(exaY, preY)

#For glm_r
predict_y_r <- predict(glm_r,x_test,type = 'response')
predict_y_r <- unname(predict_y_r)
preY_r <- ifelse(predict_y_r >= 0.1982326   , 1,0)
hoslem.test(exaY, preY_r)
```

Graph

CODE L

```
#Figure 4.6
linpred <- predict(glm,train)
wcgsm <- na.omit(train)
wcgsm <- mutate(train,residuals=residuals(glm),predprob=predict(glm,tra
in,type = 'response'))
gdf <- group_by(wcgsm, cut(linpred, breaks=unique(quantile(linpred,(1:5
0)/51))))
hldf <- summarise(gdf, y=sum(residuals), ppred=mean(predprob), count=n
())

hldf <- mutate(hldf, se.fit=sqrt(ppred*(1-ppred)/count))
ggplot(hldf,aes(x=ppred,y=y/count,ymin=y/count-2*se.fit,ymax=y/count+2*
se.fit))+geom_point()+geom_linerange(color=grey(0.75))+geom_abline(inte
rcept=0,slope=1)+xlab("Predicted Probability")+ylab("Observed Proportio
n")
```

Using McFadden's Pseudo - R^2 & Cox Snell

CODE M

```
#McFadden's

pR2(glm_r)
pR2(glm)
#values of 0.2 to 0.4 for ρ2 represent EXCELLENT fit."
```

```
#Cox Snell

r2_coxsnell(glm_r)
r2_coxsnell(glm)
```

Calculate Accuracy and Specificity

CODE N

```
#For glm
confusionMatrix( as.factor(exaY) , as.factor(preY), positive = "1")

#For glm_r
confusionMatrix( as.factor(exaY) , as.factor(preY_r), positive = "1")
```

#Random Forest # CODE O

```
set.seed(123)
#Random Forest (default mtry = 4, trees = 500)

rf <- randomForest(MIS_Status~.,data = train)
print(rf)

preY_RF <- predict(rf,test)

confusionMatrix(preY_RF,exaY, positive = "1")


#Find the suitable No. of variables tried at each split

##Tree = 500
oob.values <- vector(length=10)
for(i in 1:10) {
  temp.model <- randomForest(MIS_Status ~ ., data = train, mtry=i, ntre
e=500)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate),1]
}
oob.values

##Tree = 1000
oob.values <- vector(length=10)
for(i in 1:10) {
  temp.model <- randomForest(MIS_Status ~ ., data = train, mtry=i, ntre
e=1000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate),1]
}
oob.values

##Tree = 5000
```

```r
oob.values <- vector(length=10)
for(i in 1:10) {
  temp.model <- randomForest(MIS_Status ~ ., data = train, mtry=i, ntre
e=5000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate),1]
}
oob.values

##Tree = 7000
oob.values <- vector(length=10)
for(i in 1:10) {
  temp.model <- randomForest(MIS_Status ~ ., data = train, mtry=i, ntre
e=7000)
  oob.values[i] <- temp.model$err.rate[nrow(temp.model$err.rate),1]
}
oob.values

###So we should choose no. = 4 with 5000 trees



#Modify the No. of variables tried at each split

##Explore more trees (Try 5000 Trees)
rf_new <- randomForest(MIS_Status ~.,data = train,mtry = 4, ntree = 500
0)
print(rf_new)

preY_RF_new <- predict(rf_new,test)

confusionMatrix(preY_RF_new,exaY, positive = "1")

##graph
obb.error.data <- data.frame(
  Trees = rep(1:nrow(rf_new$err.rate), times = 3),
  Type = rep(c('OOB','UNDEFAULT','DEFAULT'), each = nrow(rf_new$err.rat
e)),
  Error = c(rf_new$err.rate[,'OOB'],
    rf_new$err.rate[,'0'],
    rf_new$err.rate[,'1'])
)

ggplot(data = obb.error.data, aes(x = Trees, y = Error)) + geom_line(ae
s(color = Type))
```

CODE P

```
deviance(lasso.model)
deviance(ridge.model)
deviance(glm_r)
```

IMPORTANCE

CODE Q

```
varImpPlot(rf_new)
```

LASSO, RIDGE & ELASTIC NET

CODE R

LASSO

```
set.seed(123)
# Predictor variables
x <- model.matrix(MIS_Status~., train)[,-1]
# Outcome variable
y <- train$MIS_Status

cv.lasso <- cv.glmnet(x, y, alpha = 1, type.measure = 'deviance', famil
y = "binomial")
plot(cv.lasso)

cv.lasso$lambda.min
coef(cv.lasso, cv.lasso$lambda.min)
coef(cv.lasso, cv.lasso$lambda.1se)

# Final model with lambda.min
lasso.model <- glmnet(x, y, alpha = 1,type.measure = 'deviance', family
 = "binomial",
                      lambda = cv.lasso$lambda.1se)
# Make prediction on test data
x.test <- model.matrix(MIS_Status ~., test)[,-1]
pre_lasso <- lasso.model %>% predict(newx = x.test, type = 'response')
preL <- ifelse(pre_lasso >= 0.4, 1,0)
# Model accuracy
confusionMatrix(as.factor(exaY) , as.factor(preL), positive = "1")
```

RIDGE

```
cv.ridge <- cv.glmnet(x, y, alpha = 0, type.measure = 'deviance', famil
y = "binomial")
plot(cv.ridge)

cv.ridge$lambda.min
coef(cv.ridge, cv.ridge$lambda.min)
```

```
coef(cv.ridge, cv.ridge$lambda.1se)

# Final model with lambda.min
ridge.model <- glmnet(x, y, alpha = 0,type.measure = 'deviance', family
 = "binomial",
                      lambda = cv.ridge$lambda.1se)
# Make prediction on test data
x.test <- model.matrix(MIS_Status ~., test)[,-1]
pre_ridge <- ridge.model %>% predict(newx = x.test, type = 'response')
preL <- ifelse(pre_ridge >= 0.4, 1,0)
# Model accuracy
confusionMatrix(as.factor(exaY) , as.factor(preL), positive = "1")
```

ROC

CODE S

```
set.seed(123)

pre_rf_pro <- predict(rf,test,type = 'prob')
pre_rf_pro_new <- predict(rf_new,test,type = 'prob')

#For Random Forest
RF.testing.ROC <- roc(test$MIS_Status ~ pre_rf_pro[, 1], quiet = TRUE)
plot(RF.testing.ROC)

#For Random Forest (NEW)
RF.testing.ROC_new <- roc(test$MIS_Status ~ p1_new[, 1], quiet = TRUE)
plot(RF.testing.ROC_new)

#For glm (NOT glm_r)
GLM.testing.ROC <- roc(test$MIS_Status ~ predict_y,quiet = TRUE)
plot(GLM.testing.ROC)
coords(GLM.testing.ROC, "best", "threshold")

#For glm_r
GLM_R.testing.ROC <- roc(test$MIS_Status ~ predict_y_r,quiet = TRUE)
plot(GLM_R.testing.ROC)
coords(GLM_R.testing.ROC, "best", "threshold")

#For LASSO
GLM_LASSO.testing.ROC <- roc(test$MIS_Status ~ pre_lasso[,1],quiet = TR
UE)
plot(GLM_LASSO.testing.ROC)
coords(GLM_LASSO.testing.ROC, "best", "threshold")

#For Ridge
GLM_ridge.testing.ROC <- roc(test$MIS_Status ~ pre_ridge[,1],quiet = TR
UE)
```

```r
plot(GLM_ridge.testing.ROC)
coords(GLM_ridge.testing.ROC, "best", "threshold")


##Put glm_r, glm, And LASSO together
plot(GLM.testing.ROC,type="l",col="red")
lines(GLM_R.testing.ROC,col="blue")
lines(GLM_LASSO.testing.ROC,col="blue")


#Notebook refer a graph with threshold and Proportion
#It also mention about R2 attributed to Nagelkerke, we can refer it and
 do a comparation

#Should be 1 - specificity (remember to correct it)



#Calculate AUC
auc(GLM_R.testing.ROC)
auc(GLM.testing.ROC)
auc(RF.testing.ROC)
auc(RF.testing.ROC)
auc(GLM_LASSO.testing.ROC)
auc(GLM_ridge.testing.ROC)
```