**#### Unit 5- 非線性迴歸 (CART, RF, MARS, SVR, BPN, KNN, Poisson)**

Selected performance predictors (Y1= ROE, Y2= EPS)

|  |  |  |
| --- | --- | --- |
| *Financial*  *Perceptive* | A1 | Debt ratio |
| A2 | Current ratio |
| A3 | Quick ratio |
| A4 | Turnover rate of accounts receivable |
| A5 | Turnover rate of inventory |
| A6 | Turnover rate of fixed assets |
| A7 | Ratio of cash flow |
| A8 | Product sale revenue |
| A9 | Turnover rate of total assets |
| A10 | Days of inventory turn |
| *Learning & Growth* | B1 | Number of employees |
| B2 | Turnover rate of employees |
| B3 | Average working years of employees |
| B4 | Average age of employees |
| B5 | Patent term |
| *Customer* | C1 | Product varieties |
| C2 | Profit margin of major products |
| C3 | Proportion of major products |
| C4 | Export ratio of major products |
| *Internal process* | D1 | Ratio of employees’ stock bonus |
| D2 | Ratio of R&D investment |
| D3 | Ratio of operational expense |
| D4 | Growth rate of revenues |
| D5 | Operational efficiency |

, (1)

, (2)

, (3)

**\*\*\*\*\*Three performance measures**

mse <- function(error) #mean-square root error

{

sum(error^2)/(length(ts.id)-1)

}

mad <- function(error) #mean absolute deviation

{

sum(abs(error))/( length(ts.id)-1)

}

mape <- function(errpe) #mean absolute percentage error

{

sum(abs(errpe))/( length(ts.id)-1)

}

install.packages("clusterSim")

library(clusterSim)

IC\_data <- read.csv("IC\_design.csv") # IC\_資料集

IC\_normal <-data.Normalization( IC\_data[,1:24], type ="n4", normalization = "column") #正規化unitization with zero minimum ((x-min)/range)

eps\_data= IC\_normal

eps\_data$EPS= IC\_data$EPS

roe\_data= IC\_normal

roe\_data$ROE= IC\_data$ROE

n=0.33\*nrow(IC\_data) #訓練與測試集切割

ts.id=sample(1:nrow(IC\_data), n)

eps.tr= eps\_data [-ts.id, ]

eps.ts= eps\_data [ts.id, ]

roe.tr= roe\_data [-ts.id, ]

roe.ts= roe\_data [ts.id, ]

**\*\*\*\*\*Linear Multiple Regression**

roe.lm<-lm(ROE~., data = roe.tr)

anova(roe.lm)

summary(step(roe.lm), k=2, method="both")

roe.pre<-predict.lm(roe.lm, roe.ts)

roe.result<- data.frame(realY=roe.ts$ROE, predictY=roe.pre)

roe.er= roe.result$realY - roe.result$predictY

roe.pe=roe.er/roe.result$realY

mse(roe.er)

mad(roe.er)

mape(roe.pe)

eps.lm<-lm(EPS~., data = eps.tr)

anova(eps.lm)

summary(step(eps.lm), k=2, method="both")

eps.pre<-predict.lm(eps.lm, eps.ts)

eps.result<- data.frame(realY=eps.ts$EPS, predictY=eps.pre)

eps.er= eps.result$realY - eps.result$predictY

eps.pe= eps.er/eps.result$realY

eps.pe=subset(eps.pe, eps.pe!=-Inf & eps.pe!=Inf)

mse(eps.er)

mad(eps.er)

mape(eps.pe)

library(car) #共線性檢定

vif(roe.lm)

vif(eps.lm)

**\*\*\*\*\*\*\*\*Regression tree (RT)**

install.packages("rpart")

install.packages("rpart.plot")

library(rpart)

library(rpart.plot)

roe.tree= ROE~A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+B1+B2+B3+B4+B5+C1+C2+C3+C4+D1+D2+D3+D4+D5 ###ROE

roe.RT=rpart(roe.tree, roe.tr, method="anova", minsplit=15, maxdepth=5)

print(roe.RT)

summary(roe.RT)

rpart.plot(roe.RT, type=4, fallen.leaves=TRUE)

roe.pre1<- predict(roe.RT, roe.ts[,-25])

roe.result1<- data.frame(realY=roe.ts$ROE, predictY=roe.pre1)

roe.er1= roe.result1$realY - roe.result1$predictY

roe.pe1= roe.er1/ roe.result1$realY

mse(roe.er1)

mad(roe.er1)

mape(roe.pe1)

eps.tree=EPS~A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+B1+B2+B3+B4+B5+C1+C2+C3+ C4+D1+D2+D3+D4+D5 ###EPS

eps.RT=rpart(eps.tree, eps.tr, method="anova", minsplit=15, maxdepth=5)

print(eps.RT)

summary(eps.RT)

rpart.plot(eps.RT, type=4, fallen.leaves=TRUE)

eps.pre1<- predict(eps.RT, eps.ts[,-25])

eps.result1<- data.frame(realY=eps.ts$EPS, predictY=eps.pre1)

eps.er1= eps.result1$realY - eps.result1$predictY

eps.pe1= eps.er1/ roe.result1$realY

mse(eps.er1)

mad(eps.er1)

mape(eps.pe1)

**\*\*\*\*\*\*\*\*Random Forest (RF)**

install.packages("randomForest")

library(randomForest)

roe.RF<-randomForest(ROE~., data=roe.tr, ntree=100, mtry=10, importance=T, proximity=T, na.action=na.omit) ###ROE

importance(roe.RF)

plot(roe.RF)

print(roe.RF)

roe.pre2<- predict(roe.RF, roe.ts[,-25])

roe.result2<- data.frame(realY=roe.ts$ROE, predictY =roe.pre2)

roe.er2= roe.result2$realY - roe.result2$predictY

roe.pe2=roe.er2/ roe.result2$realY

mse(roe.er2)

mad(roe.er2)

mape(roe.pe2)

eps.RF<-randomForest(EPS~., data=eps.tr, ntree=100, mtry=10, importance=T, proximity=T, na.action=na.omit) ###EPS

importance(eps.RF)

plot(eps.RF)

print(eps.RF)

eps.pre2<- predict(eps.RF, eps.ts[,-25])

eps.result2<- data.frame(realY=eps.ts$EPS, predictY =eps.pre2)

eps.er2= eps.result2$realY - eps.result2$predictY

eps.pe2=eps.er2/eps.result2$realY

mse(eps.er2)

mad(eps.er2)

mape(eps.pe2)

\*\*\*\*\*\*\*\*\*\*MARS

install.packages("earth")

library(earth)

#Trace earth’s execution. Default is 0

1 overview

2 forward pass

3 pruning

4 model mats summary, pruning details

roe.MARS<-earth( ROE~., degree = 2, trace=1, roe.tr) #degree考慮交互項

summary(roe.MARS) ###ROE

roe.var<-evimp(roe.MARS, trim=FALSE)

roe.var

roe.pre3<- predict(roe.MARS, roe.ts[,-25])

roe.result3<- data.frame(realY=roe.ts$ROE, predictY =roe.pre3)

roe.er3= roe.result3$realY - roe.result3$ROE

roe.pe3= roe.er3/roe.result3$realY

mse(roe.er3)

mad(roe.er3)

mape(roe.pe3)

eps.MARS<-earth( EPS~., degree = 2, trace=1, eps.tr)

summary(eps.MARS) ###EPS

eps.var<-evimp(eps.MARS, trim=FALSE) #挑選重要變數

eps.var

eps.pre3<- predict(eps.MARS, eps.ts[, -25])

eps.result3<- data.frame(realY= eps.ts$EPS, predictY= eps.pre3)

eps.er3= eps.result3$realY - eps.result3$EPS

eps.pe3= eps.er3/eps.result3$realY

mse(eps.er3)

mad(eps.er3)

mape(eps.pe3)

\*\*\*\*\*\*\*\*\*\*\*SVR\_ROE (用前面MARS選到的變數)

install.packages("e1071")

library(e1071)

MARS\_data1<- data.frame(D5= IC\_normal$D5, D1= IC\_normal$D1, C3= IC\_normal$C3, A1= IC\_normal$A1, A9= IC\_normal$A9, ROE=IC\_data$ROE)

roe.tr= MARS\_data1 [-ts.id, ]

roe.ts= MARS\_data1 [ts.id, ]

tuneSVR1 <- tune(svm, ROE ~., data = roe.tr, ranges = list(roeilon = seq(0,1,0.1), cost = 2^(2:9)))

print(tuneSVR1)

roe.SVR <- tuneSVR1$best.model #模型建立

predictY1<- predict(roe.SVR, roe.ts[,1:5]) #預測測試集

roe.result4 <- data.frame(Y1=roe.ts$ROE, predictY1)

roe.er4= roe.result4$Y1 - roe.result4$predictY1

roe.pe4=roe.er4/ roe.result4$Y1

mse(roe.er4)

mad(roe.er4)

mape(roe.pe4)

\*\*\*\*\*\*\*\*\*\*\*SVR\_EPS (用前面MARS選到的變數)

MARS\_data2 <- data.frame(D1= IC\_normal$D1, A6= IC\_normal$A6, B3= IC\_normal $B3, D5= IC\_normal$D5, C3= IC\_normal$C3, B1= IC\_normal$B1, A4= IC\_normal$A4, EPS=IC\_data$EPS)

eps.tr= MARS\_data2 [-ts.id, ]

eps.ts= MARS\_data2 [ts.id, ]

tuneSVR2 <- tune(svm, EPS ~., data=eps.tr, ranges = list(epsilon = seq(0,1,0.1),cost =2^(2:9)))

print(tuneSVR2)

eps.SVR <- tuneSVR2$best.model #訓練集建立模型

predictY2<- predict(eps.SVR, eps.ts[, 1:7]) #測試集預測效果

eps.result4 <- data.frame(Y2=eps.ts$EPS, predictY2)

eps.er4= eps.result4$Y2 - eps.result4$predictY2

eps.pe4=eps.er4/ eps.result4$Y2

eps.pe4=subset(eps.pe4,eps.pe4!=-Inf & eps.pe4!=Inf)

mse(eps.er4)

mad(eps.er4)

mape(eps.pe4)

\*\*\*\*\*\*\*\*\*\*BPN\_EPS (用前面MARS選到的變數)

library(nnet)

r1=1/max(abs(eps.tr[,1:7])) #初始隨機權數的範圍

mse1=0

num=15 #尋找隱藏層適合的神經元個數

for (i in 1:num)

{

model1=nnet(EPS~., data=eps.tr, linout = TRUE, maxit=2000, rang=r1, size=i, decay=5e-4) #建立模型,並計算隱藏層在不同神經元個數下的mse

predict.tr1=predict(model1, eps.tr[,1:7])

result1<- data.frame(epsY=eps.tr$EPS, predict1 = predict.tr1)

err1=result1$epsY - result1$predict1

mse1[i]= sqrt( sum(err1^2)/(nrow(eps.tr)-1) ) #訓練集誤差

}

plot(1:num, mse1, 'l', col=1, lty=1, ylab="Mean-square root error", xlab="# of hidden nodes")

points(1:num, mse1, col=2, pch="+")

eps.BPN=nnet(EPS~., data=eps.tr, linout = TRUE, maxit=2000, rang=r1, size=15,decay=5e-4)

predictY1=predict(eps.BPN, eps.ts[,1:7])

eps.result5<- data.frame(Y1=eps.ts$EPS, predictY1)

eps.er5= eps.result5$Y1 - eps.result5$predictY1

eps.pe5= eps.er5/eps.result5$Y1

mse(eps.er5)

mad(eps.er5)

mape(eps.pe5)

\*\*\*\*\*\*\*\*\*BPN\_ROE (用前面MARS選到的變數)

library(nnet)

r2=1/max(abs(roe.tr[,1:5])) #初始隨機權數的範圍

mse2=0

num=15 #尋找隱藏層適合的神經元個數

for (i in 1:num)

{

model2=nnet(ROE~., data=roe.tr, linout = TRUE, maxit=2000, rang=r2, size=i, decay=5e-4) #建立模型,並計算隱藏層在不同神經元個數下的mse

predict.tr2=predict(model2, roe.tr[,1:5])

result2<- data.frame(roeY=roe.tr$ROE, predict2 = predict.tr2)

err2=result2$roeY - result2$predict2

mse2[i]= sqrt( sum(err2^2)/(nrow(roe.tr)-1) ) #訓練集誤差

}

plot(1:num, mse2, 'l', col=1, lty=1, ylab="Mean-square root error", xlab="# of hidden nodes")

points(1:num, mse2, col=2, pch="o")

roe.BPN=nnet(ROE~., data=roe.tr, linout = TRUE, maxit=2000, rang=r2, size=13, decay=5e-4)

predictY2=predict(roe.BPN, roe.ts[,1:5])

roe.result5<- data.frame(Y2=roe.ts$ROE, predictY2)

roe.er5= roe.result5$Y2 - roe.result5$predictY2

roe.pe5=roe.er5/ roe.result5$Y2

mse(roe.er5)

mad(roe.er5)

mape(roe.pe5)

**\*\*\*\*\*\*\*\**K*-nearest neighbor (KNN) regression**

install.packages("FNN")

library(FNN)

roe.pre6<- knn.reg(roe.tr[,-25], roe.ts[,-25], roe.tr[,25], k = 3, algorithm= "brute")

roe.result6<- data.frame(realY=roe.ts$ROE, predictY=roe.pre6$pred)

roe.er6= roe.result6$realY - roe.result6$predictY

roe.pe6= roe.er6/ roe.result6$realY

mse(roe.er6)

mad(roe.er6)

mape(roe.pe6)

err\_knn=rep(0, 5) #針對EPS選取合適的k值

for (i in 1:7)

{

eps.pre6<- knn.reg(eps.tr[,-25], eps.ts[,-25], eps.tr[,25], k=i, algorithm= "brute")

eps.result6<- data.frame(realY=eps.ts$EPS, predictY=eps.pre6$pred)

eps.er6= eps.result6$realY - eps.result6$predictY

err\_knn[i]= mse(eps.er6)

}

err\_knn

plot(err\_knn, type="l", xlab="k", main="Selection of k=1,2,3,3,5,6,7")

eps.pre6<- knn.reg(eps.tr[,-25], eps.ts[,-25], eps.tr[,25], k = 2, algorithm= "brute")

eps.result6<- data.frame(realY=eps.ts$EPS, predictY=eps.pre6$pred)

eps.er6= eps.result6$realY - eps.result6$predictY

eps.pe6= eps.er6 / eps.result6$realY

eps.pe6=subset(eps.pe6, eps.pe6!=-Inf & eps.pe6!=Inf)

mse(eps.er6)

mad(eps.er6)

mape(eps.pe6)

**\*\*\*\*\*Linear Multiple Regression**

install.packages("MASS") #保險資料集

library(MASS)

data(Insurance)

attach(Insurance)

summary(Insurance)

cor(Holders, Claims)

tree1= Holders~District+Group+Age

rp\_DT1=rpart(tree1, Insurance, method="anova", minsplit=6)

print(rp\_DT1)

summary(rp\_DT1)

rpart.plot(rp\_DT1, type=4, fallen.leaves=TRUE)

tree2= Claims~District+Group+Age

rp\_DT2=rpart(tree2, Insurance, method="anova", minsplit=6)

print(rp\_DT2)

summary(rp\_DT2)

rpart.plot(rp\_DT2, type=4, fallen.leaves=FALSE)

**\*\*\*\*\*Poisson Regression (預測一段時間內的罕見事件的發生次數)**

**#####http://www.az-sportsnet.com/sports/?bID=mlb&pID=techterm (職棒術語)**

Barry <-read.csv("BarryBonds.csv", header=T, sep=",")

attach(Barry)

BarryIBB<-glm(formula=IBB~HR+H+SO+SB+TB, data=Barry, family=poisson(link="log"))

summary(BarryIBB)

IBB\_model<- BarryIBB<-glm(formula=IBB~H+SO+SB, data=Barry, family=poisson(link="log"))

names(IBB\_model)

exp(IBB\_model$coefficients)

predict.glm(IBB\_model, type="response", newdata=data.frame(H=100, SO=50, SB=20))

IBB\_model$deviance  #deviance統計量

1-pchisq(q=IBB\_model$deviance, df= IBB\_model$df.residual) #模型配適度衡量

**\*\*\*\*\*Unsupervised feature selection**

install.packages("caret")

library(caret)

IC.Corr = cor(IC\_normal)

IC.Redun = findCorrelation(IC.Corr, 0.90, verbose = T)

IC.Redun #pairwise correlation comparison

IC.Comba= findLinearCombos(IC\_normal)

IC.Comba #linear combination

IC\_reduce=IC\_normal[, -c(IC.Redun)]

IC\_reduce$ROE=IC\_data[,25]

IC\_reduce$EPS=IC\_data[,26]