Model Evaluation: Regression

อ.ดร.สิวะโชติ ศรีสุทธิยากร

9/1/2020

บทเรียนนี้จะกล่าวถึง cross-validation technique ที่ใช้สำหรับตรวจสอบประสิทธิภาพของโมเดลการเรียนรู้ที่พัฒนาขึ้น เนื้อหาในบทเรียนนี้จะกล่าวถึงมโนทัศน์พื้นฐานของการตรวจสอบประสิทธิภาพโมเดลการเรียนรู้ ดัชนีประสิทธิภาพของโมเดล และเทคนิค cross-validation รายละเอียดมีดังนี้

# **มโนทัศน์พื้นฐาน**

ปัญหาหนึ่งที่สำคัญในการพัฒนาโมเดลการเรียนรู้ของเครื่องคือปัญหาโมเดลสอดคล้องกับข้อมูลเกินพอดี (overfitting) ปํญหาดังกล่าวเกิดขึ้นเมื่อโมเดลการเรียนรู้ที่พัฒนาขึ้นไม่ว่าจะเป็น classification model หรือ regression model สามารถเรียนรู้ความสัมพันธ์ระหว่างตัวแปรในชุดข้อมูลฝึกหัดได้ดีมากเกินไปจนทำให้ความสามารถของโมเดลในการทำนายข้อมูลอื่น ๆ ที่อยู่นอกเหนือจากชุดข้อมูลฝึกหัดมีจำกัดหรือต่ำมากจนเกินไป ซึ่งทำให้โมเดลการเรียนรู้ที่พัฒนาขึ้นขาดคุณสมบัติความเป็นนัยทั่วไปของโมเดล (generalizability) วิธีการที่ช่วยลดทอนหรือป้องกันการเกิดปัญหา overfitting ของโมเดลคือการพัฒนาโมเดลการเรียนรู้โดยวิเคราะห์ประสิทธิภาพของโมเดลโดยใช้เทคนิค cross-validation ที่มีการแบ่งข้อมูลที่ผู้วิเคราะห์มีออกเป็นสองส่วน ส่วนแรกคือ **ชุดข้อมูลฝึกหัด (training data)** และส่วนที่สองคือ **ชุดข้อมูลทดสอบ (testing data)**

ชุดข้อมูลฝึกหัดจะใช้สำหรับเป็นตัวอย่างให้โมเดลการเรียนรู้ของเครื่องต่าง ๆ ได้เรียนรู้ความสัมพันธ์ระหว่างตัวแปรและสร้างโมเดลทางคณิตศาสตร์ที่มีความสามารถในการเลียนแบบความสัมพันธ์ในชุดข้อมูลดังกล่าว ประสิทธิภาพของโมเดลการเรียนรู้ที่คำนวณจากชุดข้อมูลฝึกหัดนี้สามารถใช้สะท้อนระดับความลำเอียง (bias) ของโมเดลที่พัฒนาขึ้นได้ อย่างไรก็ตามในบางครั้งโมเดลที่มีระดับความลำเอียงต่ำมาก ๆ ก็ไม่ใช่โมเดลที่ดีเสมอไป ทั้งนี้เป็นเพราะโมเดลดังกล่าวมีความสอดคล้องกับข้อมูลในชุดข้อมูลฝึกหัดมากจนเกินไปจนทำให้ไม่สามารถนำไปทำนายค่าสังเกตอื่นที่อยู่นอกเหนือชุดข้อมูลฝึกหัดได้ การตรวจสอบประสิทธิภาพของโมเดลโดยใช้ชุดข้อมูลทดสอบจึงมีความจำเป็น เนื่องจากข้อมูลในชุดข้อมูลทดสอบนี้เป็นข้อมูลที่โมเดลที่พัฒนาขึ้นยังไม่เคยเรียนรู้มาก่อน ประสิทธิภาพที่คำนวณได้จากชุดข้อมูลทดสอบนี้จึงสะท้อนคุณสมบัติของโมเดลในด้านความเป็นนัยทั่วไปในการทำนาย

# **ดัชนีวัดประสิทธิภาพของโมเดล (model performance metric)**

บทเรียนนี้จะกล่าวถึงสถิติ 3 ตัวที่มักนิยมใช้เป็นดัชนีวัดประสิทธิภาพในการทำนายของโมเดล ได้แก่

1. **สัมประสิทธิ์การตัดสินใจ (coefficient of determination: )** มีค่าเท่ากับกำลังสองของสัมประสิทธิ์สหสัมพันธ์ระหว่างค่าทำนายกับค่าจริง ทำให้มีค่าที่เป็นไปได้อยู่ในช่วง ซึ่งมีความหมายเป็นร้อยละของปริมาณความผันแปรที่มีร่วมกันระหว่างค่าทำนายที่ได้จากโมเดลกับค่าจริง โมเดลที่มีค่า เข้าใกล้ 1 แสดงว่ามีประสิทธิภาพในการทำนายสูง
2. **รากที่สองของความคลาดเคลื่อนกำลังสอง (root mean squares error: )** มีสูตรการคำนวณดังนี้

เมื่อ คือค่าสังเกตจริงในชุดข้อมูลทดสอบ, คือค่าทำนายของค่าสังเกตในชุดข้อมูลทดสอบที่ได้จากโมเดลการเรียนรู้ และ คือขนาดของชุดข้อมูลทดสอบ ดัชนี สะท้อนปริมาณความคลาดเคลื่อนในการทำนายของโมเดลโดยเฉลี่ย โมเดลที่มีค่า ต่ำแสดงว่ามีประสิทธิภาพในการทำนายสูง

1. **ส่วนเบี่ยงเบนเฉลี่ยของความคลาดเคลื่อน (absolute mean error: )** มีสูตรการคำนวณดังนี้

จากสูตรข้างต้นจะเห็นว่า มีความหมายเหมือนกับ แต่มีการใช้ค่าสัมบูรณ์แทนการยกกำลังสอง จึงทำให้ดัชนี มีความแกร่งคือค่าทำนายที่ผิดปกติมากกว่า

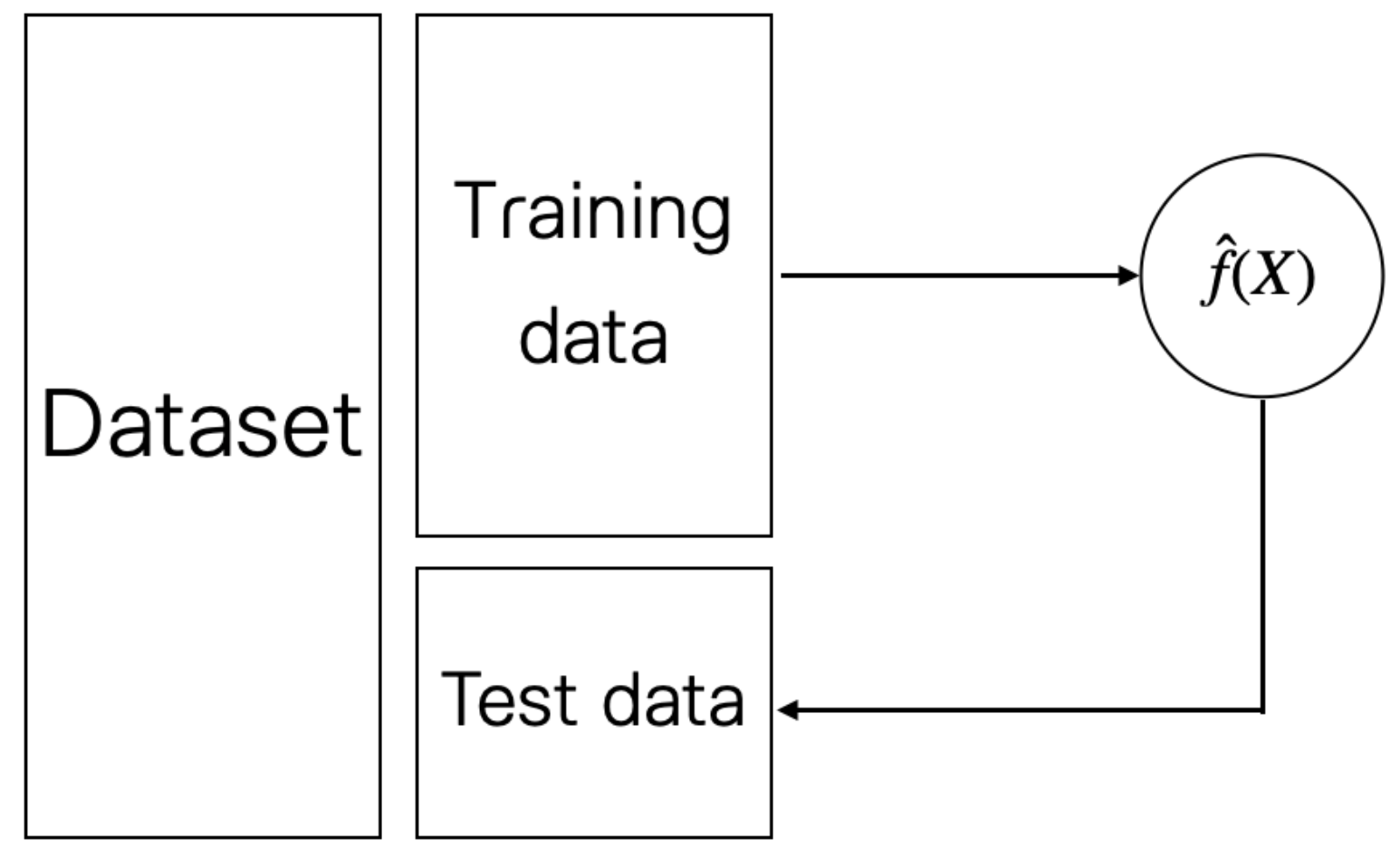
# **Cross-validation**

โดยทั่วไปกระบวนการพัฒนาและตรวจสอบประสิทธิภาพของโมเดลด้วยเทคนิค cross-validation จะประกอบด้วยขั้นตอนใหญ่ ๆ ดังนี้

1. พัฒนาโมเดลโดยใช้ชุดข้อมูลฝึกหัด
2. ใช้โมเดลที่พัฒนาขึ้นเพื่อทำนายค่าสังเกตในชุดข้อมูลทดสอบ
3. คำนวณประสิทธิภาพของโมเดลในชุดข้อมูลทดสอบ

เทคนิค cross-validation ยังอาจจำแนกได้เป็น 4 ประเภท ตามลักษณะของการแบ่งข้อมูล ดังนี้

**(1) train/test method** วิธีการนี้เป็นวิธีการดั้งเดิม กล่าวคือทำการแบ่งชุดข้อมูลที่มีออกเป็น 2 ส่วนได้แก่ ชุดข้อมูลฝึกหัด และชุดข้อมูลทดสอบ ตามอัตราส่วนที่กำหนดเช่น 80:20, 70:30, 60:40 หรือ 50:50 เป็นต้น

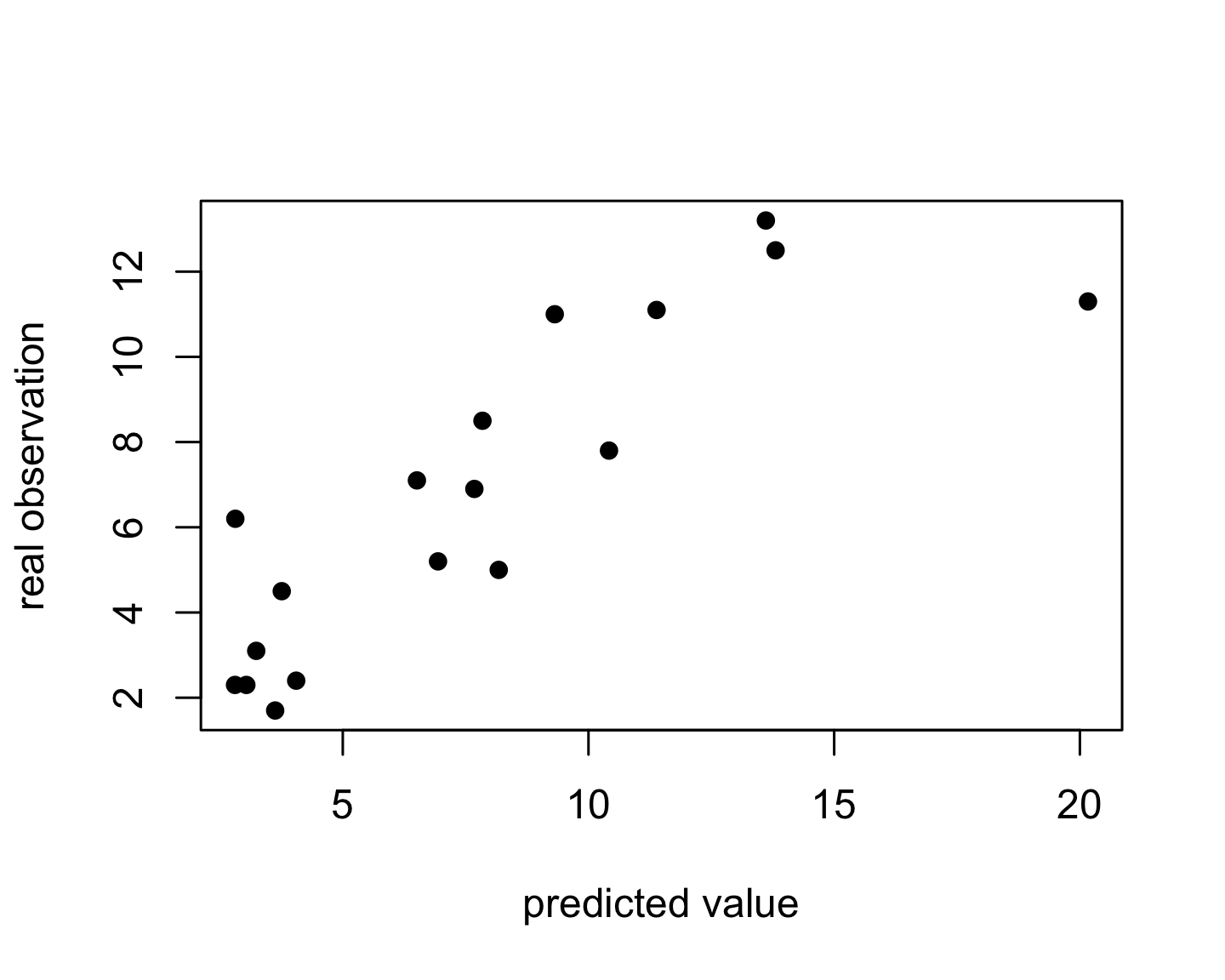


ตัวอย่างต่อไปนี้แสดงการตรวจสอบประสิทธิภาพของโมเดลโดยใช้ train/test method ใช้ข้อมูลที่ใช้เป็นตัวอย่างคือชุดข้อมูล state.x77 ที่เก็บรวบรวมข้อมูลอัตราการฆาตกรรม (Murder) กับสภาพทางเศรษฐกิจ สังคม และภูมิศาสตร์ของแต่ละรัฐในประเทศอเมริกา

dat<-data.frame(state.x77) #import and convert data into data.frame  
str(dat) #exploring dat

## 'data.frame': 50 obs. of 8 variables:  
## $ Population: num 3615 365 2212 2110 21198 ...  
## $ Income : num 3624 6315 4530 3378 5114 ...  
## $ Illiteracy: num 2.1 1.5 1.8 1.9 1.1 0.7 1.1 0.9 1.3 2 ...  
## $ Life.Exp : num 69 69.3 70.5 70.7 71.7 ...  
## $ Murder : num 15.1 11.3 7.8 10.1 10.3 6.8 3.1 6.2 10.7 13.9 ...  
## $ HS.Grad : num 41.3 66.7 58.1 39.9 62.6 63.9 56 54.6 52.6 40.6 ...  
## $ Frost : num 20 152 15 65 20 166 139 103 11 60 ...  
## $ Area : num 50708 566432 113417 51945 156361 ...

library(caret)  
train.id<-createDataPartition(dat$Murder,p=0.6,list=F) #split the data  
train.dat<-dat[train.id,] #training dataset  
test.dat<-dat[-train.id,] #testing dataset  
  
#trainthe regression model using lm() function  
fit<-lm(Murder~., data=train.dat)  
  
#calculate predictions and compute performance metric from testing data  
pred<-predict(fit,test.dat)  
# plot scatter plot between prediction and real value  
plot(pred,test.dat$Murder,xlab="predicted value",ylab="real observation",pch=16)



**Performance metric**

# performance metric  
performance<-data.frame(R2(pred,test.dat$Murder),  
 RMSE(pred,test.dat$Murder),  
 MAE(pred,test.dat$Murder))  
names(performance)<-c("R2","RMSE","MAE")  
performance

## R2 RMSE MAE  
## 1 0.7257354 2.624652 1.733352

วิธี train/test เป็นวิธีที่มีประสิทธิภาพดีในกรณีที่ผู้วิเคราะห์มีข้อมูลจำนวนมาก อย่างไรก็ตามวิธีการนี้ก็มีข้อสังเกตคือ โมเดลทำนายที่พัฒนาขึ้นพัฒนาจากข้อมูลในชุดข้อมูลฝึกหัดซึ่งเป็นข้อมูลเพียงบางส่วนของข้อมูลทั้งหมด จึงอาจทำให้โมเดลเกิดความลำเอียงได้ โดยเฉพาะในกรณีที่ข้อมูลมีจำนวนไม่มาก

**(2) Leave one out cross validation (LOOCV)** วิธีการนี้แบ่งข้อมูลออกเป็นสองส่วนเช่นเดียวกับวิธีการ train/test แต่มีความแตกต่างตรงลักษณะของการแบ่ง กล่าวคือวิธีการนี้จะแบ่งข้อมูล 1 ค่าไว้เป็นข้อมูลทดสอบ จากนั้นใช้ข้อมูลที่เหลือทั้งหมดเพื่อพัฒนาโมเดลสำหรับทำนายข้อมูล 1 ค่าในข้างต้น และเก็บผลประสิทธิภาพการทำนายข้อมูลดังกล่าวไว้ จากนั้นดำเนินกระบวนการข้างต้นซ้ำจนกว่าจะทดสอบข้อมูลครบทุกค่า และแล้วจึงคำนวณประสิทธิภาพของโมเดลจากค่าประสิทธิภาพการทำนายของข้อมูลที่เก็บไว้

ตัวอย่างต่อไปนี้แสดงการวิเคราะห์ประสิทธิภาพของโมเดลแบบ Leave one out cross validation โดยใช้ฟังก์ชัน train() ใน package-caret โดยใช้ข้อมูล state.x77 ในข้างต้น

# specify training control  
train.clt<-trainControl(method="LOOCV")  
  
# train the regression model using train() function  
fit<-train(Murder~., data=dat,method="lm", trControl=train.clt)  
  
#print the LOOCV results  
fit

## Linear Regression   
##   
## 50 samples  
## 7 predictor  
##   
## No pre-processing  
## Resampling: Leave-One-Out Cross-Validation   
## Summary of sample sizes: 49, 49, 49, 49, 49, 49, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 2.004621 0.7115175 1.621311  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE

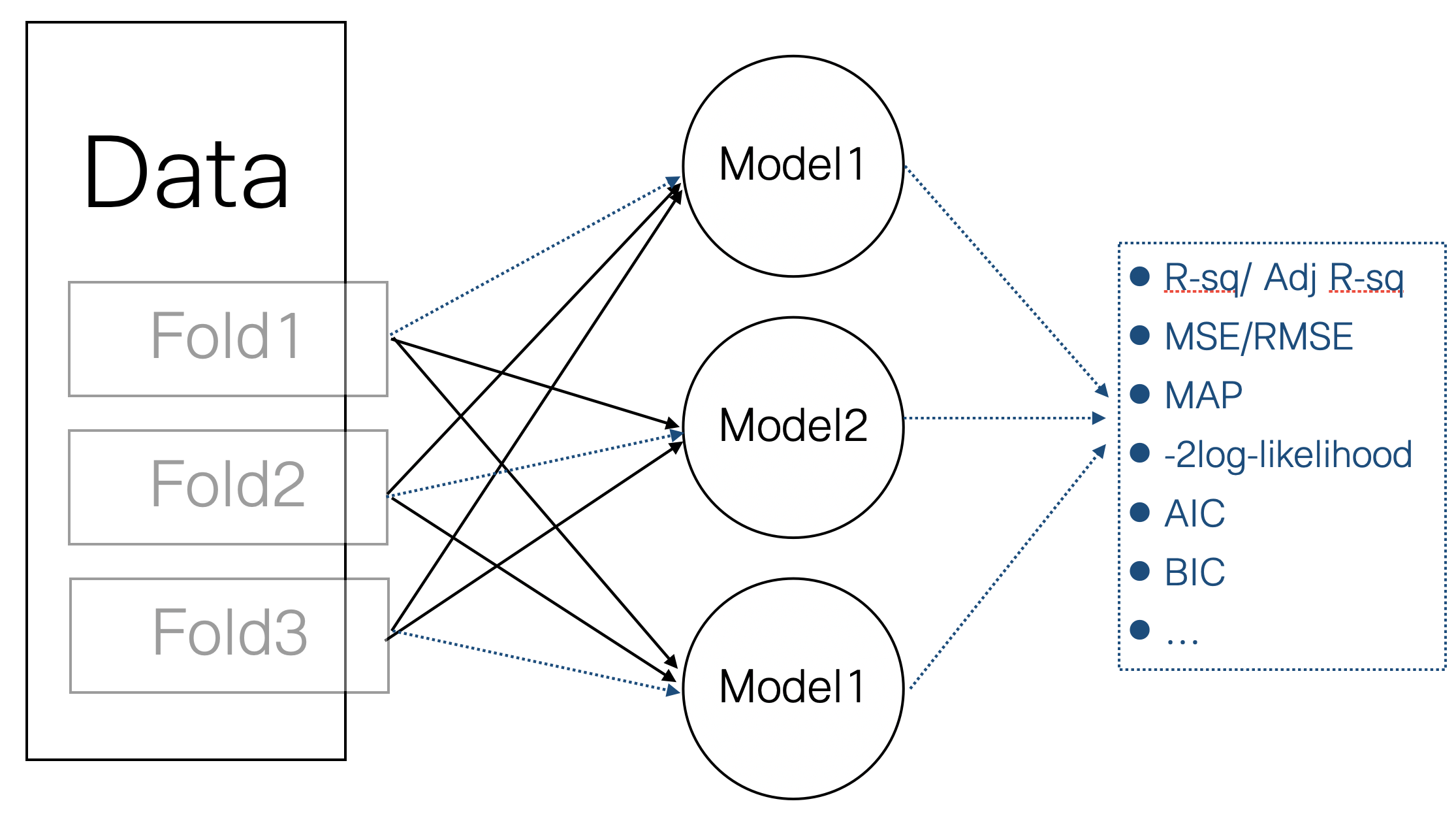
#regression output  
summary(fit)

##   
## Call:  
## lm(formula = .outcome ~ ., data = dat)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -3.4452 -1.1016 -0.0598 1.1758 3.2355   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 1.222e+02 1.789e+01 6.831 2.54e-08 \*\*\*  
## Population 1.880e-04 6.474e-05 2.905 0.00584 \*\*   
## Income -1.592e-04 5.725e-04 -0.278 0.78232   
## Illiteracy 1.373e+00 8.322e-01 1.650 0.10641   
## Life.Exp -1.655e+00 2.562e-01 -6.459 8.68e-08 \*\*\*  
## HS.Grad 3.234e-02 5.725e-02 0.565 0.57519   
## Frost -1.288e-02 7.392e-03 -1.743 0.08867 .   
## Area 5.967e-06 3.801e-06 1.570 0.12391   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.746 on 42 degrees of freedom  
## Multiple R-squared: 0.8083, Adjusted R-squared: 0.7763   
## F-statistic: 25.29 on 7 and 42 DF, p-value: 3.872e-13

วิธี LOOCV มีข้อดีคือโมเดลทำนายถูกพัฒนาข้อมูลทั้งหมดที่มี จึงทำให้โมเดลทำนายที่ได้มีความลำเอียงต่ำ อย่างไรก็ตามวิธีการนี้มีข้อจำกัดคือใช้ทรัพยากรในการคำนวณที่ค่อนข้างมาก โดยเฉพาะในกรณีที่ชุดข้อมูลมีขนาดใหญ่ นอกจากนี้วิธีการนี้ยังมีความไวต่อค่าทำนายที่ผิดปกติอีกด้วย

**(3) K-fold cross-validation**

วิธีการนี้แตกต่างจาก LOOCV ตรงทีมีการแบ่งชุดข้อมูลออกเป็น k ส่วน ที่เรียกว่า k-fold จากนั้นเก็บข้อมูล 1 ชุดไว้เพื่อใช้ทดสอบประสิทธิภาพของโมเดล และใช้ชุดข้อมูล k-1 ชุดที่เหลือเพื่อพัฒนาโมเดลทำนายสำหรับทำนายค่าสังเกตในชุดข้อมูลที่เก็บไว้ในข้างต้น จากนั้นดำเนินกระบวนการข้างต้นซ้ำเพื่อสลับระหว่างชุดข้อมูลทดสอบ และชุดข้อมูลฝึกหัดจนครบ แล้วจึงคำนวณค่าประสิทธิภาพของโมเดลจากดัชนีวัดประสิทธิภาพจำนวน k ค่า รูปต่อไปนี้แสดงกระบวนการ 3-fold cross validation



# specify the train control  
train.clt<-trainControl(method="cv", number=10)  
  
# train the regression model using train() function  
fit<-train(Murder~., data=dat, method="lm", trControl=train.clt)  
  
# print the CV results  
fit

## Linear Regression   
##   
## 50 samples  
## 7 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 44, 45, 45, 45, 45, 45, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 1.945005 0.7938908 1.594328  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE

วิธีการนี้หากกำหนดให้จำนวน k มีค่ามากขึ้นเรื่อย ๆ ผลการวิเคราะห์ที่ได้จะมีค่าลู่เข้าหาวิธีการ LOOCV นอกจากนี้โมเดลทำนายที่ได้จะมีความลำเอียงลดลงตามจำนวน k ที่เพิ่มขึ้น อย่างไรก็ตามจากการวิจัยที่ผ่านมาพบว่า การกำหนดจำนวน k ที่เหมาะสมอาจกำหนดให้อยู่ในช่วงตั้งแต่ 5 ถึง 10 (ไม่ได้มีกฎเกณฑ์ตายตัว) Molinaro (2005) ทำการเปรียบเทียบประสิทธิภาพระหว่างเทคนิค LOOCV กับ 10-fold CV พบว่าให้ผลที่ไม่แตกต่างกัน แต่ประสิทธิภาพในการคำนวณของเทคนิค CV สามารถทำได้ไวกว่า นอกจากนี้ยังพบว่าการกำหนด k=2 หรือ k=3 ก่อให้เกิด bias ในการประมาณค่าประสิทธิภาพของโมเดลสูงมาก

**(4) Repeated K-fold cross-validation**

เทคนิค repeated k-fold CV สามารถใช้เพื่อเสริมประสิทธิภาพให้กับเทคนิค CV ได้ เทคนิคนี้เป็นการทำ k-fold cross validation ซ้ำได้หลาย ๆ ครั้ง ซึ่งช่วยลดความลำเอียงของเทคนิค CV ในกรณีที่กำหนดให้ k มีจำนวนน้อย ๆ

train.clt<-trainControl(method="repeatedcv", number=5, repeats=5)  
fit<-train(Murder~., data=dat, method="lm", trControl=train.clt)  
fit

## Linear Regression   
##   
## 50 samples  
## 7 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 5 times)   
## Summary of sample sizes: 40, 42, 39, 39, 40, 41, ...   
## Resampling results:  
##   
## RMSE Rsquared MAE   
## 1.99268 0.7646133 1.623016  
##   
## Tuning parameter 'intercept' was held constant at a value of TRUE