Meta-model Results

Xijin

10/22/2019

Motivation

Based on one of those simulation settings and simulate 1000 times to see the results and the result from meta-model

Simulation settings:

1. Modelling strategies: maximum likelihood
2. Distribution of predictor variables: mulivariate normal distribution
3. Sample size: 400
4. Number of candidate predictors P: 4
5. Events fraction: 0.5
6. Predictor effects: equal effects

functions

```
#predictive error estimation by meta model
mspe_meta <- function(n,ef,P){
  exp( -0.59 - 1.06*log(n) + 0.36*log(ef) + 0.94*log(P))
}

brier_meta <- function(n,ef,P){
  exp(-0.91 - 0.04*log(n) + 0.62*log(ef) + 0.04*log(P) )
}

mape_meta <- function(n,ef,P){
  exp( -0.48 - 0.53*log(n) + 0.31*log(ef) + 0.48*log(P))
}

#predictive error real values
brier_true <- function(p,SIMxy){
  sum((SIMxy[,"y"]-p)^2)/length(p)
}

mspe_true <- function(p,SIMxy,dgm.par){
  design.mat <- as.matrix(cbind(1,SIMxy[,-which(colnames(SIMxy)=="y")]))
  p_true <- exp(design.mat%*%dgm.par)/(1+exp(design.mat%*%dgm.par))
  mean((p_true-p)^2)
}

mape_true <- function(p,SIMxy,dgm.par){
  design.mat <- as.matrix(cbind(1,SIMxy[,-which(colnames(SIMxy)=="y")]))
  p_true <- exp(design.mat%*%dgm.par)/(1+exp(design.mat%*%dgm.par))
  mean(abs(p_true-p))
```

```r
}

#function for prediction

  predict.lrm <- function(lp){
    as.vector(exp(lp)/(1+exp(lp)))
  }

#generate independent variables
gen.MVNX <- function(n,mu,sigma){
  mvrnorm(n=n,mu=mu,Sigma=sigma)
}

#generate dependent variables
gen.binY <- function(SIMx,dgm.par){
  no.X <- ncol(SIMx)
  design.mat <- cbind(1,SIMx)
  p <- exp(design.mat%*%dgm.par)/(1+exp(design.mat%*%dgm.par))
  y <- rbinom(length(p),1,p)
  SIMxy <-data.frame(x=SIMx,y=y)
  colnames(SIMxy)[1:no.X]<-paste("X",1:no.X,sep="")
  SIMxy
}
```

```r
#data generation
args <- list(
  mu = c(0,0,0,0),
  sigma = matrix(c(1,0,0,0,
                   0,1,0,0,
                   0,0,1,0,
                   0,0,0,1),nrow=4),
  n = 400
)
datagen <- "gen.MVNX"
dgm.par <- c(-0.0009684,  0.2832211,  0.2832211,  0.2832211,  0.2832211)
```

Results based on 1000 simulations

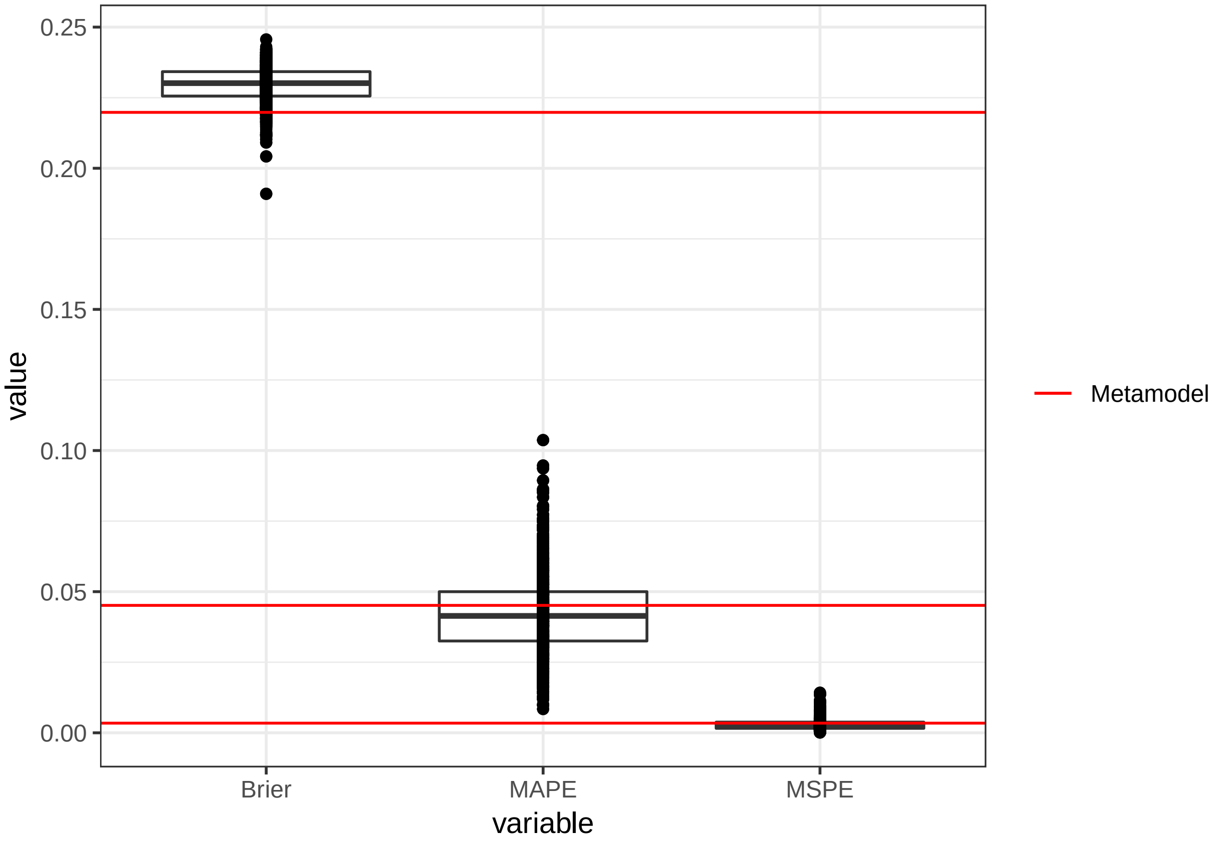
```r
sim.num <- 1000 # 1000 simulations
brier_true_values <- rep(0,sim.num)
mape_true_values <- rep(0,sim.num)
mspe_true_values <- rep(0,sim.num)
ef <- rep(0,sim.num)

for (i in 1:sim.num){
  #data generation, this is original data set
  SIMx <- do.call(what=datagen,args=args)
  SIMxy <- gen.binY(SIMx=SIMx,dgm.par=dgm.par)
  SIMx <- SIMxy[,-ncol(SIMxy)]
  y <- SIMxy[,"y"]
  ef[i] <- sum(y)/length(y)
  #model fitting and corresponding predicted values
  formu <- as.formula(paste("y~",
                            paste(colnames(SIMxy)[-which(colnames(SIMxy)=="y")],collapse="+"),sep=""))
  fit <- lrm(formu,data=SIMxy)
  #predict outcomes
  SIMx <- cbind("Intercept"=1,SIMx)
  lp <- unlist(as.matrix(SIMx[,names(coef(fit))])%*%data.matrix(coef(fit)))
  pred <- predict.lrm(lp)
  #metrics calculation
  brier_true_values[i] <- brier_true(p=pred,SIMxy)
  mape_true_values[i] <- mape_true(p=pred,SIMxy,dgm.par)
  mspe_true_values[i] <- mspe_true(p=pred,SIMxy,dgm.par)
}
```

Results based on meta-model

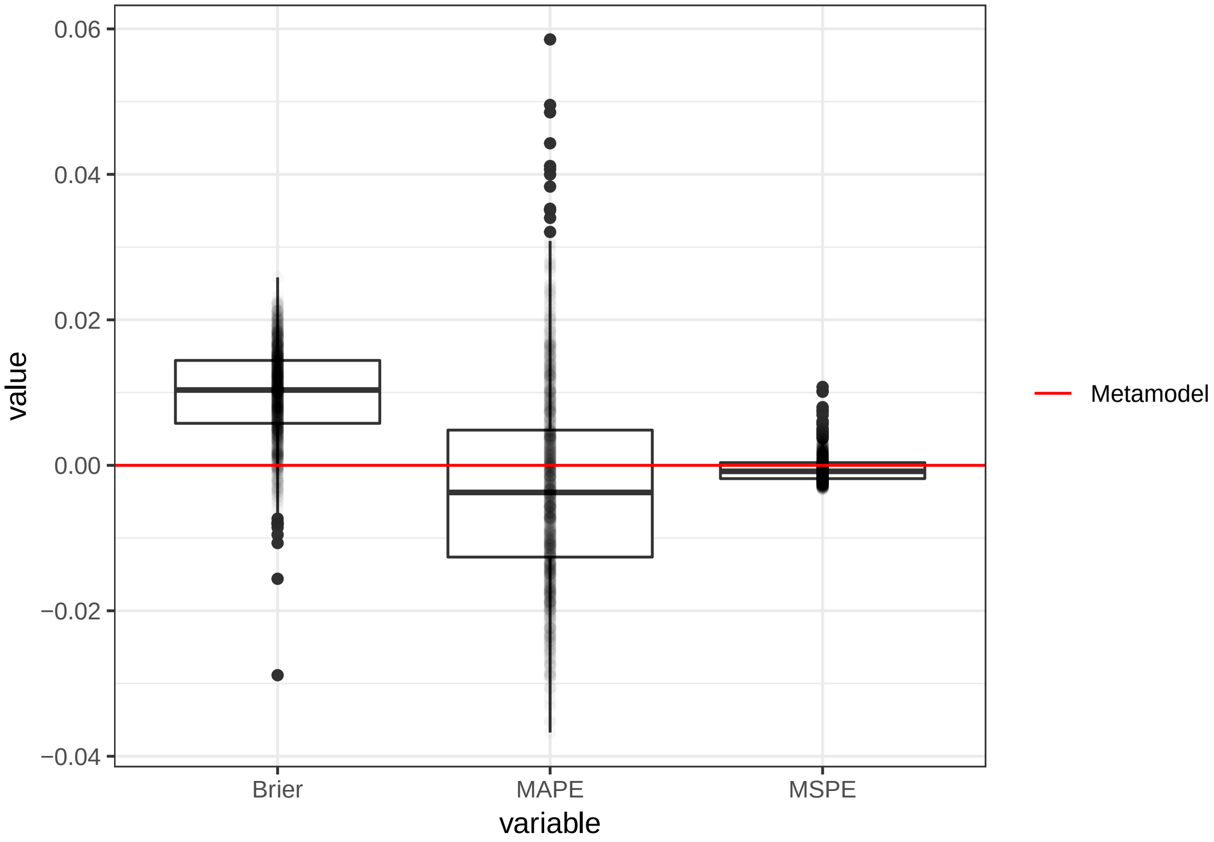
```
brier_meta_values <- brier_meta(n=args$n,ef=0.5,P=length(dgm.par))
mape_meta_values <- mape_meta(n=args$n,ef=0.5,P=length(dgm.par))
mspe_meta_values <- mspe_meta(n=args$n,ef=0.5,P=length(dgm.par))
```

Results comparison

```
dat_sim <- data.frame(cbind(Brier=brier_true_values,
                            MAPE=mape_true_values,
                            MSPE=mspe_true_values))
library(reshape2)
dat_sim <- melt(dat_sim,
              measure.vars=c('Brier',
                             'MAPE',
                             'MSPE'))
dat_sim$variable <- as.factor(dat_sim$variable)
meta_values <- c(brier_meta_values,mape_meta_values,mspe_meta_values)
dat <- cbind(dat_sim,meta=c(rep(brier_meta_values,sim.num),
                            rep(mape_meta_values,sim.num),
                            rep(mspe_meta_values,sim.num)))
ggplot(dat) +
  geom_boxplot(aes(x=variable, y=value))+
  geom_point(aes(x=variable, y=value))+
  geom_hline(aes(yintercept=meta_values[1],col="Metamodel"))+
  geom_hline(aes(yintercept=meta_values[2],col="Metamodel"))+
  geom_hline(aes(yintercept=meta_values[3],col="Metamodel"))+
  theme_bw()+
 # labs(color = "Meta model result",
#       x="Prediction error metrics",
#       ylab="Value")+
  theme(legend.text = element_text(NULL))+
  scale_colour_manual(NULL,values=c('Metamodel' = "red"))
```



```
dat_diff <- dat_sim
meta <- c(rep(brier_meta_values,sim.num),
                            rep(mape_meta_values,sim.num),
                            rep(mspe_meta_values,sim.num))
dat_diff$value <- dat_sim$value-meta
ggplot(dat_diff) +
  geom_boxplot(aes(x=variable, y=value))+
  geom_point(aes(x=variable, y=value),alpha = 0.03)+
  geom_hline(aes(yintercept=0,col="Metamodel"))+
  scale_colour_manual(NULL,values=c('Metamodel' = "red"))+
   theme_bw()+
  theme(legend.text = element_text(NULL))
```


0.06
0.04
0.02
0.00
−0.02
−0.04
value
Brier
MAPE
MSPE
variable
Metamodel