cross_validation

First generate simulated training data as before.

```r
f<-function(x){
  return(0.2 + x - 0.5*x^2 + 0.1*x^3 - 0.5*x^4)
}
xv = seq(0,1,0.001)
yv = f(xv)

dx = runif(500)
dy = runif(500)

boundry = f(dx)
label = (dy>boundry)+0

x_value = dx
y_value = dy + rnorm(length(dy),sd=0.1)
training_data = cbind(y_value, x_value, label)
```

Code for K-fold cross-validation

```r
# function for validate a single batch
# the parameters are training data, batch labels and the id of the test batch

validate_batch<-function(data, batch, test_batch_id){
  d = data[batch!=test_batch_id,]

  # fit the probit regression
  y = d[,1]
  x = d[,2]
  l = d[,3]
  fit = glm(l~y+x,family=binomial(link="probit"))
  beta = matrix(ncol=1, fit$coef)
  test_d = data[batch==test_batch_id,1:2]
  if(!is.null(dim(test_d))){
   test_d = matrix(ncol=3,cbind(rep(1,dim(test_d)[1]), test_d))
  }else{
    test_d = matrix(ncol=3,c(1,test_d))
  }

  pred = test_d%*%beta
  pred_label= (pred>=0)+0
  true_label = data[batch==test_batch_id,3]
  return(length(which(true_label!=pred_label)))
}



K_fold_CV<-function(data, K){
  N = dim(data)[1]
  batch = rep(1:K, ceiling(N/K))[1:N]
  total_error = sum(sapply(1:K, function(x) validate_batch(data,batch,x)))
  EPE = total_error/N
  return(EPE)
}
```

Try with $K = 5, 10$

```
K_fold_CV(training_data,5)
```

```
## [1] 0.112
```

```
K_fold_CV(training_data,10)
```

```
## [1] 0.11
```