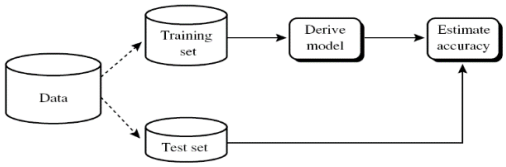
模型选择与评估方法

一、 模型选择

说明：对于同一个问题，可以以有多种模型来解决。如一个分类问题，可以用 logistic 回归、SVM、朴素贝叶斯等，那么如何选择一个最好的模型呢。首先会想到的是，对每个模型，用训练集合去训练他们，取训练误差最小的模型。 但是这有明显的缺陷，因为这将会得到一个最复杂的模型（比如一个 10 次多项式），产生严重的过拟合。给出下面三种方法。

1、 留出法:

数据集划分成互斥的训练集和测试集，随机划分，划分要保持数据分布的一致性。常用做法: 2/3-4/5 样本用作训练，剩余样本用作测试



2、 K-折交叉验证

常用做法：5 折-交叉，10 折-交叉， 20 折-交叉

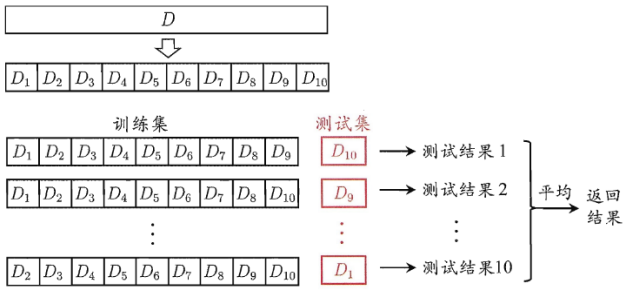


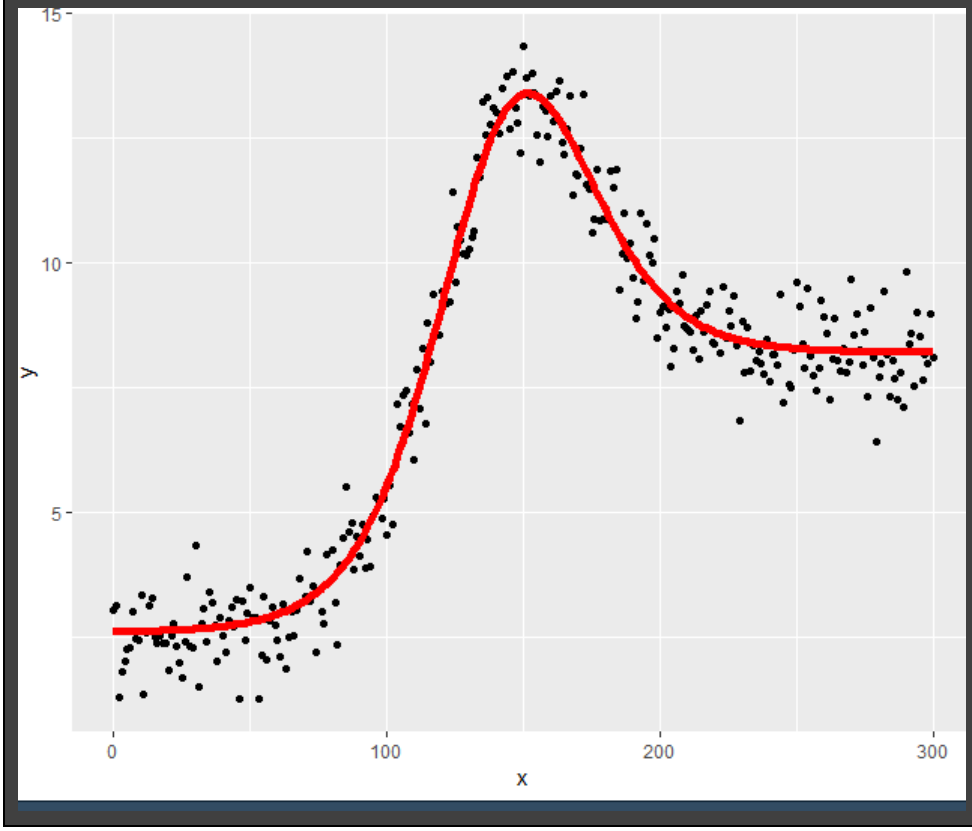
图 2.2 10 折交叉验证示意图

3、 留一法

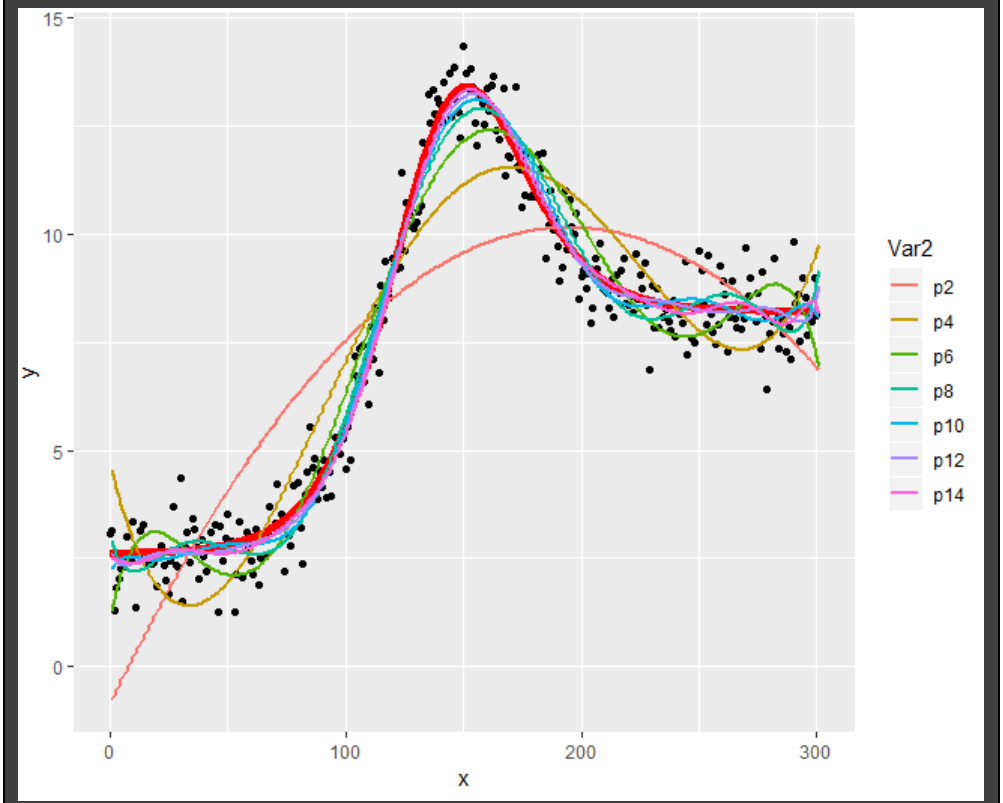
K-折交叉验证的特例，如果样本数为 N，则为 N-折交叉验证

R 程序：

```
library(ggplot2)
#  曲线函数
plusModel<-function(xita,x){
   h0 = xita[1]
   h1 = xita[2]
   h2 = xita[3]
   t1 = xita[4]
   t2 = xita[5] + t1
   beta=xita[6]
   ( h0 + (h1-h0)/(1+exp(-beta*(x-t1))) ) * ( h2 + (h1-h2)/(1+exp(beta*(x-t2))) ) / h1
}
#创建样本数据
x=seq(0,300,by=1)
#添加噪声
par = c(2.6170019, 20.8858480, 8.2183875,128.3199316, 33.8561940,0.0570245,840.6384970)
y=plusModel(par, x) +rnorm(length(x),0,0.6)
data1=data.frame(x,true_y=plusModel(par, x), y=y)
#查看数据
ggplot(data1,aes(x,y))+geom_point()+geom_line(aes(y=true_y), color='red', size=2)
```

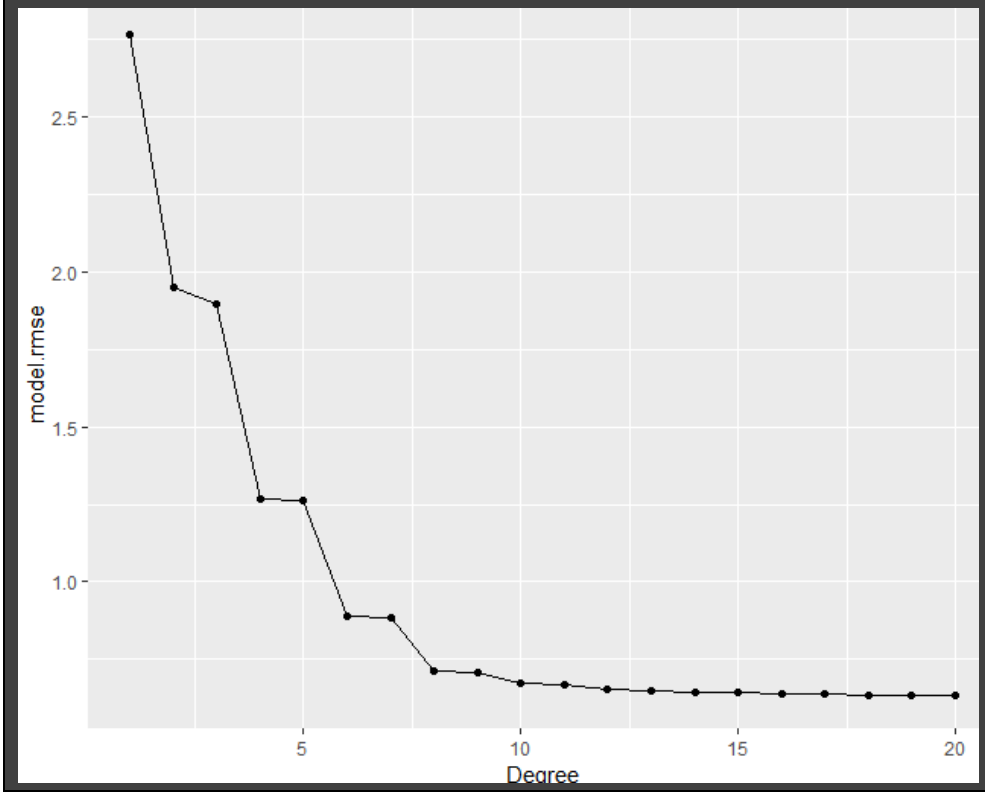


```
# 这里采用多种模型
# y=a+bx+cx^2+dx^3+ex^4···
#曲线拟合
poly3=lm(y~poly(x,degree = 3),data=data1)
polydata = cbind(p2=predict(lm(y~poly(x,degree = 2),data=data1)),
                 p4=predict(lm(y~poly(x,degree = 4),data=data1)),
                 p6=predict(lm(y~poly(x,degree = 6),data=data1)),
                 p8=predict(lm(y~poly(x,degree = 8),data=data1)),
                 p10=predict(lm(y~poly(x,degree = 10),data=data1)),
                 p12=predict(lm(y~poly(x,degree = 12),data=data1)),
                 p14=predict(lm(y~poly(x,degree = 14),data=data1)))
polydata = melt(polydata)
ggplot(data1,aes(x,y))+geom_point()+
  geom_line(aes(y=true_y), color='red', size=2)+
  geom_line(data=polydata, aes(x=Var1,y=value, color=Var2),   size=1)
```




```
#计算均方误差  RMSE
RMSE=function(t,p){
  return(sqrt(mean((t-p)^2)))
}
#随着自由度的增加，  查看均方误差的变化
```

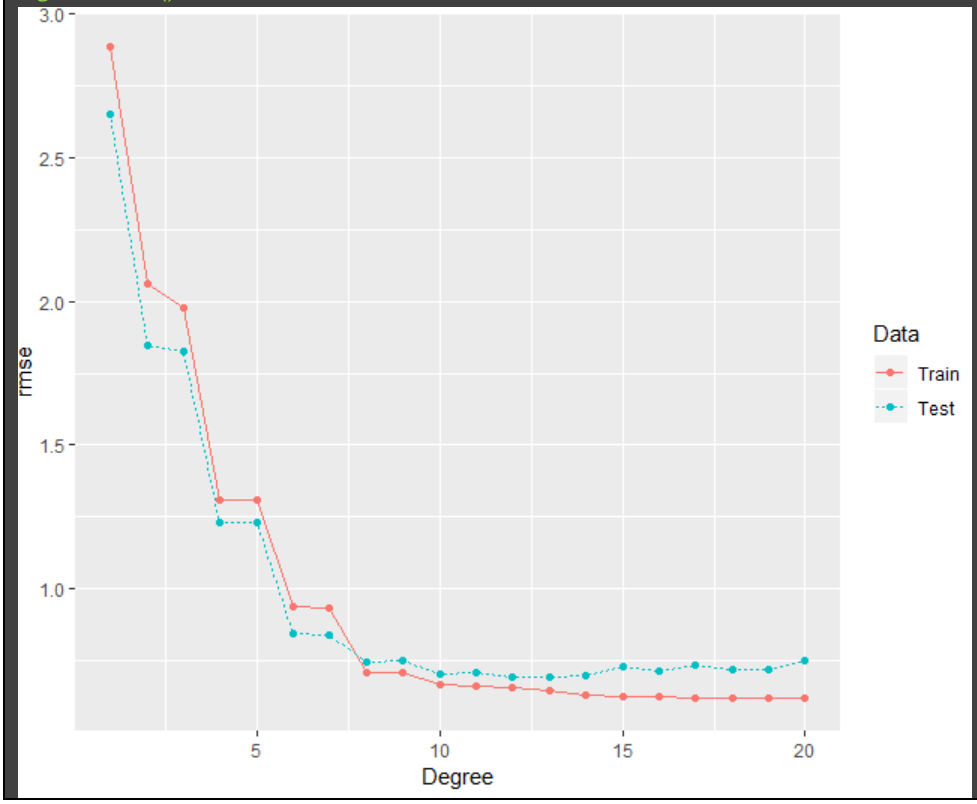
```
Performance=data.frame()
for (d in 1:20) {
   polyfit=lm(y~poly(x,degree = d),data=data1)
   mean.rmse=RMSE(data1$y,mean(data1$y))
   model.rmse=RMSE(data1$y,predict(polyfit))
   Performance=rbind(Performance,  data.frame(Degree=d,model.rmse,Rsqr=1-model.rmse/mean.rmse))
}
ggplot(Performance,aes(Degree,model.rmse))+geom_line()+geom_point()
```




```
#1.交叉验证法
# 所谓交叉验证方法即把数据集分为两部分，Training data 和 testing data.用 Training data 建模，
# 用 testing data 来验证模型的泛化能力。从而避免过拟合。
#1.把数据分为 trainingdata and testingdata
index=nrow(data1)
#随机抽取一部分为训练样本，一部分为测试样本
index1=sample(index,round(0.5*index))
trainingdata=data1[index1,]
testingdata=data1[-index1,]

#做一个循环得到 traindata 和 testdata 的 rmse.
Performance=data.frame()
for(d in 1:20){
```

```
polyfit=lm(y~poly(x,degree = d),data=trainingdata)
Performance=rbind(Performance,data.frame(Degree=d,                    Data='Train',
rmse=RMSE(trainingdata$y,predict(polyfit))))
Performance=rbind(Performance,data.frame(Degree=d,                    Data='Test',
rmse=RMSE(testingdata$y,predict(polyfit,newdata = testingdata))))
}
ggplot(Performance,aes(Degree,rmse,linetype=Data,color=Data))+geom_point()
+geom_line()
```




```
#10 折交叉验证法

library(caret)
folds<-createFolds(y=data1$y,k=10) #根据 data1 的 y 把数据集切分成 10 等份
re<-{}
Performance=data.frame()
for(i in 1:10){
   traindata<-data1[-folds[[i]],]
   testdata<-data1[folds[[i]],]
   p6=lm(y~poly(x,degree = 6),data=traindata)
   p8=lm(y~poly(x,degree = 8),data=traindata)
   p10=lm(y~poly(x,degree = 10),data=traindata)
   pre6 = predict(p6,newdata=testdata)
   pre8 = predict(p8,newdata=testdata)
   pre10 = predict(p10,newdata=testdata)
   Performance<- rbind(Performance,c(RMSE(as.numeric(testdata$y),pre6),
```

```
                                        RMSE(as.numeric(testdata$y),pre8),
                                        RMSE(as.numeric(testdata$y),pre10)))
}
apply(Performance,2,mean)

>0.8934863          0.7272538          0.6921159
```

二、 评估方法

1、 计算准确率，错误率，精度，召回率，F1 度量，加权 F1 度量， 宏查准率，宏查全率， 微查准率，微查全率

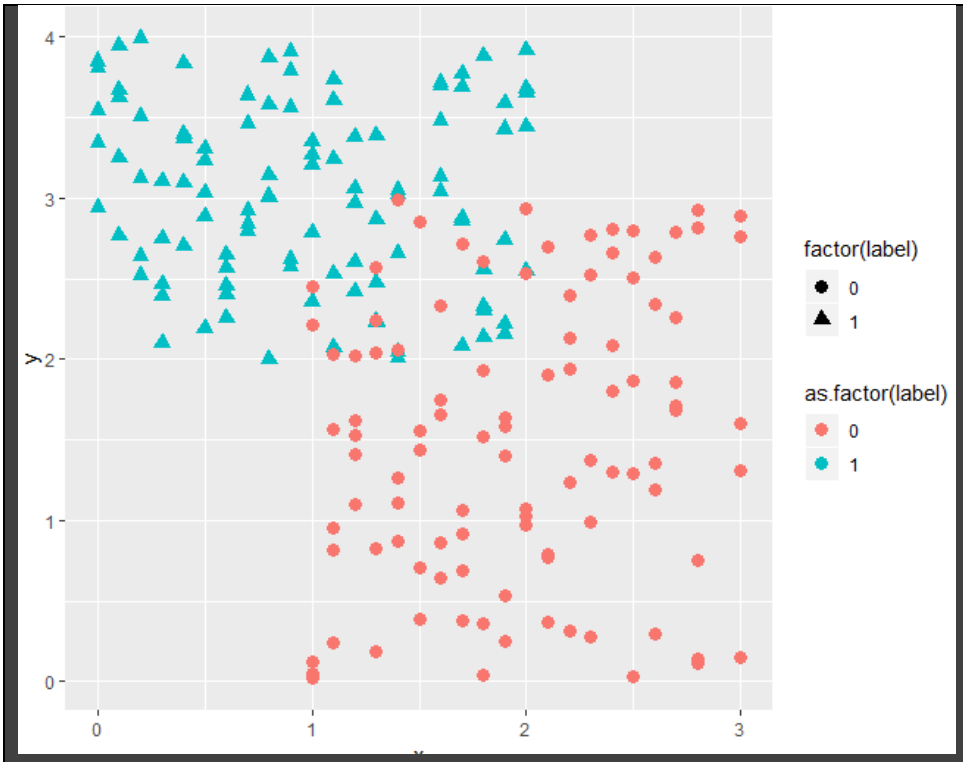
2、 P-R 图

3、 ROC-AUC

4、 统计检验

5、 偏差方差分解

```
#
class1 = data.frame(x=sample(seq(0,2,0.1),20),y=sample(seq(2,4,0.01),100), label=1)
class2 = data.frame(x=sample(seq(1,3,0.1),20),y=sample(seq(0,3,0.01),100), label=0)
data = rbind(class1,class2)
#md = melt(data,id=c,measure=c("y1","y2","x1","x2"))
ggplot(data=data,          aes(x,y,          color=as.factor(label)))+geom_point(aes(shape          =
factor(label)),size=3)
```




```
########
index=nrow(data)
#随机抽取一部分为训练样本，一部分为测试样本
index1=sample(index,round(0.6*index))
trainingdata=data[index1,]
testingdata=data[-index1,]

logi <- glm(label ~ x+y, family= binomial(link="logit"), trainingdata)
fitt.pi<-predict(glm.safe1,testingdata[,1:2],type="resp")#fitted(logi)
#ypred<-1*(fitt.pi>0.5) #1*逻辑变量就变成了 0 和 1 变量
library(plotROC)
r1=plot.roc(testingdata[,3],fitt.pi, col='red', lwd=4)

library(e1071)
svm1 <- svm(label ~ x+y, data=data )
svm.pi<-predict(svm1,testingdata[,1:2],type="resp")
r2=plot.roc(testingdata[,3],svm.pi, col='deepskyblue', lwd=4, add=TRUE)

legend('bottomright',
       legend= paste(c('logit', 'svm'),
                     ' - AUC=',signif(c(as.numeric(r1$auc), as.numeric(r2$auc))),
                     sep=''),
```

```
col=c('red', 'deepskyblue'), lty=1, lwd=
```