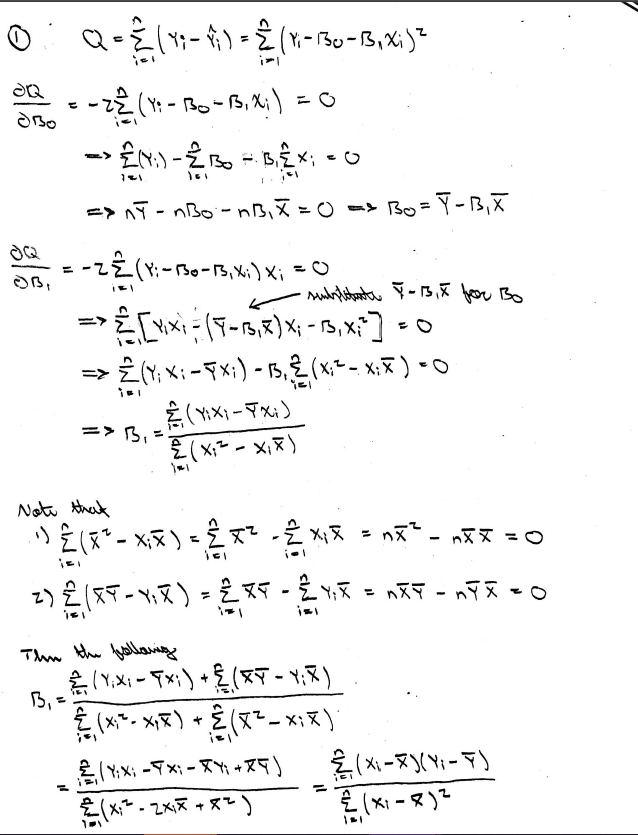
Nicholas Orriols

Statistical Learning

Homework 1

1. The following obtains the values for which minimize the sum of square errors function (denoted by *Q*).



1. i) Because we have the true conditional probability distribution, we know the expected error rate of the Bayes classifier is 1 minus the expectation of the maximum of this probability distribution with respect to the choice of the response class. Since this is the same for both data sets, the expectation of each error rate (i.e. the average error rates) are the same for each data set. Since the data sets are large (*n =* 500), the error rates for each data set will likely be close to this expected rate (which is the irreducible error in the model). However, since the data sets are not equivalent, the sample error rates will likely be slightly different.

ii) After loading in the data, the following code calculates and outputs the training error rate and then the testing error rate.

> prob = function(x1, x2){

+ return (pnorm(.5\*x1 - .4\*x2))

+ }

>

> simpleClassify = function(x1,x2){

+ y=c()

+ for(i in 1:length(x1))

+ y[i] = as.numeric(prob(x1[i],x2[i]) >= .5)

+ return (y)

+ }

>

> trainYhat = simpleClassify(train.x1, train.x2)

> testYhat = simpleClassify(test.x1, test.x2)

>

> mean(trainYhat != train.y)

[1] 0.328

> mean(testYhat != test.y)

[1] 0.312

iii) The following code computes the K-nearest neighbors prediction of the test data for *k = 3* and then outputs the error rate.

> set.seed(777)

> knn3 = knn(train=cbind(train.x1,train.x2), test=cbind(test.x1,test.x2), k=3, cl=train.y)

> knnPred = as.numeric(knn3) - 1

> mean(knnPred != test.y)

[1] 0.386

iv) Since this error rate is significantly larger than the previous error rates which approximate the irreducible error, *k = 3* is not the best choice of *k*: it is likely other values of *k* will produce smaller error rates.

v) The following code computes the error rates for the K-nearest neighbors approximations for each integer between 1 and 100 and creates the following plot of error rates as a function of *k*. It also outputs the 6 smallest error rates and their corresponding *k*s.

> set.seed(777)

> rates=c()

> m = c(1,10)

> for(i in 1:100){

+

+ mod=knn(train=cbind(train.x1,train.x2),

+ test=cbind(test.x1,test.x2), k=i, cl=train.y)

+ pre=as.numeric(mod)-1

+ rates[i] = mean(pre != test.y)

+

+ }

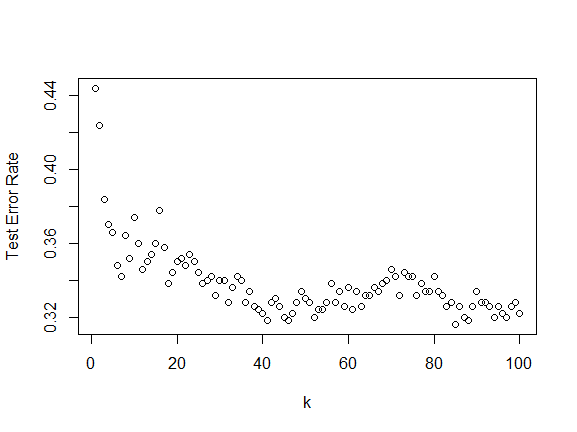
> ks = which(rates %in% sort(rates)[1:5]); ks

[1] 41 45 85 87 97 98

> rates[ks]

[1] 0.318 0.320 0.316 0.320 0.320 0.320

> plot(rates, xlab="k", ylab="Test Error Rate")



I think the best choice of *k* would be 41, since larger *k* values do not decrease the testing error rate by a significant margin, and because the error rate of this choice of *k* (.318) is very close to the approximations of the irreducible error.

vi) Since there are many values of *k* near the approximations for the irreducible error (there are 40 *k*s with an error rate less than or equal to the training error rate using the probability distribution), I think KNN does a good job of approximating the Bayes classifier.

vii) The following code calculates and outputs the KNN test error rate for *k = 40* with the 20 additional covariates (with seed 777).

> conf.train=cbind(train.x1,train.x2,xrandom[1:500,])

> conf.test=cbind(test.x1,test.x2,xrandom[501:1000,])

>

> knnConf = knn(train=conf.train,test=conf.test,k=40,cl=train.y)

> mean(as.numeric(knnConf)-1 != test.y)

[1] 0.422

viii) The above test error rate being significantly larger than most of the test error rates in the above graph indicates that the efficacy of KNN is negatively affected by covariates which are not predictive of the outcome. This makes sense since KNN classifies points based on the classification of the *k* points nearest to it. When considering many irrelevant covariates, the nearest neighbors of some point will likely be ‘near-by’ because of their distance to the point with respect to the irrelevant covariates. Since these covariates are not predictive of the outcome, this will likely have a negative effect on the testing error rate.

1. ai) I included the year as 4 dummy variables (for each of 2002-2005). If year was included as a continuous random variable, then the associated regression coefficient would be restricted to reflecting a linear trend in the response given the rest of the covariates. For example, given some set of non-year covariates, if the year coefficient was *b > 0* the model would predict 2003 to have a response greater than 2002 by *b* and greater than 2001 by *2b*. However, by using dummy variables, we remove this restriction: for instance, given the rest of the covariates, 2003 could be greater than 2002 by *b* and be less than 2001 by *c* (). The following code fits the model.

> y2002 = rep(c(0,1,0),times=c(242,252,252\*3))

> y2003 = rep(c(0,1,0),times=c(242+252,252,252\*2))

> y2004 = rep(c(0,1,0),times=c(242+252\*2,252,252))

> y2005 = rep(c(0,1),times=c(242+252\*3,252))

>

> lr.mod=lm(Today~Lag1+Lag2+Lag3+

+ Lag4+Lag5+Volume+y2002+y2003+y2004+y2005,

+ data=Smarket)

aii) Whether the covariates are predictive of the response is equivalent to whether all the regression coefficient are equal to 0. The F statistic for *n* observations of *p* covariates is which is distributed as under the null hypothesis (all regression coefficients are 0). The F-statistic and its corresponding p-value for this model are given by the following code and partial output.

> summary(lr.mod)

Residual standard error: 1.137 on 1239 degrees of freedom

Multiple R-squared: 0.006222, Adjusted R-squared: -0.001799

F-statistic: 0.7757 on 10 and 1239 DF, p-value: 0.6525

Since the p-value is large(er than .05), we fail to reject the null hypothesis that the covariates are not predictive of the outcome; that is we do not find sufficient evidence to conclude the covariates are indicative of the outcome.

aiii) The following code fits the same model except with 3 degrees of the Lag1 covariate. Its F-statistic and p-value are included in the partial output below.

> lr2=lm(Today~poly(Lag1, 3)+Lag2+Lag3+

+ Lag4+Lag5+Volume+y2002+y2003+

+ y2004+y2005, data=Smarket)

> summary(lr2)

Residual standard error: 1.138 on 1237 degrees of freedom

Multiple R-squared: 0.006302, Adjusted R-squared: -0.003337

F-statistic: 0.6538 on 12 and 1237 DF, p-value: 0.7966

Since the p-value is large, as above, we fail to reject the hypothesis that the covariates are not predictive of the outcome.

bi) The following code randomly splits the data into a training and testing set. It then calculates the KNN models for *k = 1, … 100*, saves the minimum error rate and its corresponding *k,* and outputs them (note reset of seed 777).

> Smarket$Direction = (as.numeric(Smarket$Direction))-1

> set.seed(777)

> sample=sample(seq\_len(nrow(Smarket)), size=floor(.5\*nrow(Smarket)))

> train.smarket=Smarket[sample,]

> test.smarket=Smarket[-sample,]

>

> min = c(1,10)

> for (i in 1:100){

+

+ mod=knn(train=train.smarket[,1:8], test=test.smarket[,1:8],

+ k=i, cl=train.smarket[,9])

+ rate=mean((as.numeric(mod)-1) != test.smarket[,9])

+

+ if(rate < min[2])

+ min=c(i,rate)

+

+ }

> min

[1] 22.0000 0.0896

So, the smallest testing error rate was .0896 from the KNN model with *k = 22*.

bii) Since the testing error rate is very small, this would indicate the covariates are predictive of the outcome. Although they were not predictive in either regression model, the small test error rate indicates that points with ‘near-by’ covariate vectors are more likely to have the same response.

1. i) Let be the vector of least squares coefficients, let be the average outcome among subjects with predictors and be the response for the subject with predictors . Note that and where is a residual. Both intervals are centered at **.**

The variances of the estimators are

To construct the confidence intervals, we use *RSE* to estimate and take the variances’ square roots. For a given confidence level, the width of each interval is a function of its corresponding standard error (the center plus/minus this standard error times a t-distribution percentile). Since has a larger standard error than , the second interval is wider than the first.

ii) Since each interval is centered at the sample mean of observations with predictors , then increasing n makes this estimate more accurate, so the standard error associated with using it goes to 0 as . Therefore, the confidence interval’s width approaches 0 as n increases. However, since the standard error of is at least *RSE*, the width of the prediction interval only converges to *RSE* times the t-distribution value (i.e. not 0) as .