MULTICOLINEARITY

1. Introduction

This exercise gives an opportunity to fit a ridge regression model.

Note: Much of the material here can be found in the textbook *An Introduction to Statistical Learning*, by James, Witten, Hastie, and Tibshirani. The electronic version of the text can be downloaded for free while on UO campus: `http://alliance-primo.hosted.exlibrisgroup.com/UO:CP71188145890001451`

This is a very readible reference on the *learning* approach to statistical modeling, which focuses on prediction.

2. Cross-Validation

We first discuss k-fold *cross validation*. In this case, the data is divided into k equal portions, at random. Call these subsets $D_1, \ldots, D_k$.

There are then k performance measures: If j ranges in $1, 2, \ldots, k$, the j-th measure fits the model on all the data but D_j, and then tests the model on D_j. Performance is measured by, say, mean squared error between predicted and observed values on D_j. Call this MSE_j.

The k-fold cross validation measure is then defined as

$$\mathrm{CV}_{(k)} = \frac{1}{k} \sum \mathrm{MSE}_j$$

2.1. Ridge Regression.

There are several functions available to perform ridge regression. The `lm.ridge` function from `MASS` package is one; we use here `glmnet` from `glmnet` package.

We look at the `Hitters` data:

`http://pages.uoregon.edu/dlevin/DATA/Hitters.txt`

Use

```
> hiturl = url("http://pages.uoregon.edu/dlevin/DATA/Hitters.txt")
> hit = na.omit(read.table(hiturl))
```

Set aside a third of the data to later use to test the predictive ability of the fitted model. Use mean squared error of prediction to evaluate the predictive ability. Use cross validuation (on the training data only) to pick the parameter λ.

Below is some helpful code. Note that it is run for the entire data set.

```
> library(glmnet)
> x = model.matrix(Salary~., hit)
> y = hit$Salary
> grid=10^seq(10,-2,length=100)
> #alpha = 0 does ridge regression
> ridge.mod = glmnet(x,y,alpha=0,lambda=grid)
> set.seed (1)
> train=sample(1:nrow(x), nrow(x)/2)
> test=(-train)
```

```
> y.test=y[test]
> ridge.mod=glmnet(x[train,],y[train],alpha=0,lambda=grid, thresh =1e-12)
> ridge.pred=predict(ridge.mod,s=4,newx=x[test,])
> mean((ridge.pred-y.test)^2)
[1] 101036.8
> set.seed(1)
> cv.out=cv.glmnet(x[train ,],y[train],alpha=0)
> #plot(cv.out)
> bestlam=cv.out$lambda.min
> bestlam
[1] 211.7416
>
```

3. Principle Components

Do the same as above, but use principle components. Pick the number of components via cross validation.

```
> #need pls package
> library(pls)
> set.seed(2)
> pcr.fit=pcr(Salary~., data=hit ,scale=TRUE, validation ="CV",x=TRUE,y=TRUE)
> validationplot(pcr.fit,val.type="MSEP")
```

4. Homework Assignment

In this exercise, we will predict the number of applications received using the other variables in the College data set:

pages.uoreogn.edu/dlevin/DATA/College.txt

(a) Split the data set into a training set and a test set.

(b) Fit a linear model using least squares on the training set, and report the test error obtained.

(c) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

(d) Fit a PCR model on the training set, with M chosen by cross- validation. Report the test error obtained, along with the value of M selected by cross-validation.

(e) Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these three approaches?