

Math Models of Data Science February 6, 2006

Inclass lab. Please hand in table at the end of the lab.

STEP 1:

First you need to fire up matlab and make a directory to work in. Then download the files from www.rpi.edu/~bennek/class/mds/Aqua-all.txt If you are using one of the class computers, do the following.

Login in locally using these credentials:

```
userID: guest
password: bio.info
```

Have the students open a local terminal window by right clicking on the desktop background and selecting "Open Terminal".

Make a directory in your RCS account

```
mkdir mds
cd mds
```

In the terminal window at the prompt enter

```
ssh -X -l <rsc_userid> rcs-sun1.rpi.edu
```

The minus options are an uppercase X and a lowercase L.

Once logged into rcs-sun1.rpi.edu, at a Unix prompt enter

```
/campus/mathworks/matlab/7sp3/bin/matlab
```

Matlab should display on the local machine with the students working on the RCS Sun system from inside their RCS accounts.

Copy

```
/afs/rpi.edu/home/68/bennek/public_html/class/mds/Aqua-all.txt
```

STEP 2:

Load in the data

```
Data=load('Aqua-all.txt');
```

The matrix Data now contains the Aqua Sol Data.

The first column is an id number. The second is the response. The remaining are the 525 descriptors.

First let's break the data up into training and testing sets.

The training data will be the first 100 points. Xtrain contains the descriptors, Ytrain is the response.

```
Xtrain=Data(1:100,3:527);
Ytrain=Data(1:100,2);
```

The remaining points are the testing data:

```
Xtest=Data(101:197, 3:527);
Ytest=Data(101:197,2);
```

Step 3:

Let's duplicate the models we tried in class
First is regular ridge regression with $\lambda = 10$;

$$\min_w L_\lambda(\mathbf{w}, S) = \lambda \|\mathbf{w}\|^2 + \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Write down the optimality condition for \mathbf{w} for this problem?

Now compute the optimal \mathbf{w} using Matlab. Enter the following in Matlab

```
L=10;
```

```
w1 = inv(Xtrain'*Xtrain+ L* eye(525))*(Xtrain'*Ytrain)
```

Let's see how well it works. Calculate the predicted values for the test set.

```
Ypred1=Xtest*w1;
```

```
Ypred1_trn=Xtrain*w1
```

Compute the mean square error of the train and test sets.

```
MSE1=mean((Ypred1-Ytest).^2)
```

```
MSE1_trn=mean((Ypred1_trn-Ytrain).^2)
```

Enter the error in the appropriate entry in the table on the last page.

Step 4:

As discussed in class the model can be improved by taking into account a bias term so $f(x)=xw+b$. For Ridge regression this can be accomplished by centering X and Y . Calculate the mean of X_{train} and Y_{train} .

```
mu=mean(Ytrain);
```

```
Xbar=mean(Xtrain);
```

```
Xtrain2=Xtrain-ones(100,1)*Xbar;
```

```
Ytrain2=Ytrain-mu;
```

```
Xtest2=Xtest-ones(97,1)*Xbar;
```

Create the new model

```
w2 = inv(Xtrain2'*Xtrain2+ L* eye(525))*(Xtrain2'*Ytrain2)
```

To predict the new one, we need to know the bias b .

```
b=mu;
```

The new prediction and the error are:

```
Ypred2=Xtest2*w2+b;
```

```
Ypred2_trn=Xtrain2*w2 +b;
```

```
MSE2=mean((Ypred2-Ytest).^2)
```

```
MSE2_trn=mean((Ypred2_trn-Ytrain).^2)
```

Can you improve it by playing around with L ? Enter your results in the table on the last page.

Step 5:

According to the theory, calculating the function in the dual space should work exactly the same. Let's give it a try. The optimal coefficients in the dual space are

$\alpha = (G + \lambda I_t)^{-1} y$ where $G = \mathbf{X}\mathbf{X}'$ and

$w = \mathbf{X}'\alpha$

In Matlab

```
G=Xtrain2*Xtrain2';
```

```
a=inv(G+L*eye(100))*Ytrain2;
```

```
w3=Xtrain2'*a;
```

Compare w_3 to w_2 ? Are they identical? Why or why not?

Computationally is there any reason to prefer one method over the other?

Step 6:

The advantage of the dual approach is that one can replace the Gram matrix G with any appropriate kernel matrix K . This corresponds to mapping the data to feature space and finding the best predictive model in that space.

The ridge regression algorithm in Steps 4, consisted of the following Matlab commands.

```
%Center the Y data
```

```
mu=mean(Ytrain);
```

```
Ytrain2=Ytrain-mu;
```

```
%Center the X data
```

```
Xbar=mean(Xtrain);
```

```
Xtrain2=Xtrain-ones(100,1)*Xbar;
```

```
Xtest2=Xtest-ones(97,1)*Xbar;
```

```
% Compute W,b
```

```
w2 = inv(Xtrain2'*Xtrain2+ L* eye(525))*(Xtrain2'*Ytrain2)
```

```
b=mu;
```

```
%Compute the predictions
```

```
Ypred2=Xtest2*w2+b;
```

```
Ypred2_trn=Xtrain2*w2 +b;
```

```
MSE2=mean((Ypred2-Ytest).^2)
```

```
MSE2_trn=mean((Ypred2_trn-Ytrain).^2)
```

Now we would like to make this algorithm work in feature space by using the kernel trick.

Let's try this process for a degree d polynomials. Try a quadratic polynomial first.

Set the degree

```
d=2
```

Compute the polynomial

```
K=(Xtrain2*Xtrain2'+1).^d;
```

Now we need to replace each of the steps with the kernel space.

Centering the Y data is not changed.

But now we cannot center the X data directly in feature space.

We cannot center X data directly in feature space. We must instead figure out the equivalent kernel operations.

$$\text{Note } \bar{\mathbf{x}} = \frac{1}{\ell} \sum_{i=1}^{\ell} \mathbf{x}_i = \frac{1}{\ell} \mathbf{e}' \mathbf{X}$$

$$\mathbf{X}_c = \mathbf{X} - \mathbf{e}\bar{\mathbf{x}} = \mathbf{X} - \mathbf{e}\left(\frac{1}{\ell} \mathbf{e}' \mathbf{X}\right) = \left(\mathbf{I} - \frac{1}{\ell} \mathbf{e}\mathbf{e}'\right) \mathbf{X}$$

$$\begin{aligned} \mathbf{K}_c &= \mathbf{X}_c \mathbf{X}_c' = \left(\mathbf{I} - \frac{1}{\ell} \mathbf{e}\mathbf{e}'\right) \mathbf{X} * \mathbf{X}' \left(\mathbf{I} - \frac{1}{\ell} \mathbf{e}\mathbf{e}'\right) \\ &= \left(\mathbf{I} - \frac{1}{\ell} \mathbf{e}\mathbf{e}'\right) \mathbf{K} \left(\mathbf{I} - \frac{1}{\ell} \mathbf{e}\mathbf{e}'\right) \end{aligned}$$

\mathbf{K}_c is now the centered kernel

Create a matlab expression that "centers" the kernel.

$\mathbf{K}_c = ?$

The next is to compute \mathbf{a} . The only change is now we use the dual version.

Now that we have the centered kernel we can go ahead and fit a degree d polynomial.

$$\mathbf{a} = \text{inv}(\mathbf{K}_c + \mathbf{L} * \text{eye}(100)) * \mathbf{Y}_{\text{train2}};$$

We can't compute \mathbf{w} , but we can predict the test points. . Note we have to be careful to center the testing kernel before we predict. Note that since we centered the training data before computing the kernel, we center the testing data in the same way. The uncentered test kernel is

$$\mathbf{K}_{\text{test}} = (\mathbf{X}_{\text{test2}} * \mathbf{X}_{\text{train2}}' + 1).^d;$$

Before we used the matlab command $\mathbf{X}_{\text{test2}} = \mathbf{X}_{\text{test}} - \text{ones}(97,1) * \mathbf{X}_{\text{bar}}$, to center the test data. Now we need to compute the kernel equivalent of $\mathbf{K}_{\text{test}} = \mathbf{X}_{\text{test2}} * \mathbf{X}_{\text{train2}}'$;

Note that for prediction, for the case of training with 100 points and testing with 97, the test kernel centering becomes:

$$\begin{aligned} \mathbf{K}_{\text{testc}} &= (\mathbf{X}_{\text{tst}} - e_{97} \bar{\mathbf{x}}) * (\mathbf{X}_{\text{trn}} - e_{100} \bar{\mathbf{x}}) \quad \text{where } \bar{\mathbf{x}} = \frac{1}{\ell} e_{100}' \mathbf{X}_{\text{trn}} \\ &= \left(\mathbf{X}_{\text{tst}} - \frac{1}{\ell} e_{97} e_{100}' \mathbf{X}_{\text{trn}}\right) * \mathbf{X}_{\text{trn}}' \left(\mathbf{I} - \frac{1}{\ell} e_{100} e_{100}'\right) \\ &= \mathbf{X}_{\text{tst}} \mathbf{X}_{\text{trn}}' \left(\mathbf{I} - \frac{1}{\ell} e_{100} e_{100}'\right) - \frac{1}{\ell} e_{97} e_{100}' \mathbf{X}_{\text{trn}} \mathbf{X}_{\text{trn}}' \left(\mathbf{I} - \frac{1}{\ell} e_{100} e_{100}'\right) \\ &= \left(\mathbf{K}_{\text{tst}} - \frac{1}{100} e_{97} (e_{100}' \mathbf{K})\right) \left(\mathbf{I} - 1/100\right) \end{aligned}$$

Multiplying out and putting it in Matlab:

$$\mathbf{K}_{\text{testc}} = (\mathbf{K}_{\text{test}} - 1/100 * \text{ones}(97,1) * \mathbf{K}_{\text{bar}}') * (\text{eye}(100) - 1/100);$$

The prediction is then

$Y_{pred6} = K_{testc} * a + b;$
 $Y_{pred6_trn} = K_c * a + b;$
 $MSE6 = \text{mean}((Y_{pred6} - Y_{test}).^2)$
 $MSE6_trn = \text{mean}((Y_{pred6_trn} - Y_{train}).^2)$

Repeat this experiment with $L=50$. Repeat this experiment for $d=3$.

Method	Parameter	MSE train	MSE test
Ridge Regression	$L=10$		
Ridge Regression	$L=$		
Ridge Regression with bias	$L=10$		
Ridge Regression with bias	$L=$		
Dual Ridge Regression with bias	$L=10$		
Kernel Ridge Regression	$L=10$ $d=2$		
Kernel Ridge Regression	$L=50$ $d=2$		
Kernel Ridge Regression	$L=50$ $d=3$		

HINT: Here is one way to calculate the centered training kernel

$K_{bar} = K * \text{ones}(100,1);$ $K_c = (K - 1/100 * \text{ones}(100,1) * K_{bar}') * (\text{eye}(100) - 1/100);$