Cross Validated

SVD in linear regression [duplicate]

Asked 8 years, 3 months ago Active 8 years, 3 months ago Viewed 9k times

4

4

This question already has an answer here:

Definition of orthogonal matrix (1 answer)

Closed 6 years ago.

I was reading the book Elements of Statistical Learning and came across the section that tried to interpret ridge regression using singular value decomposition (SVD) of the design matrix, X. Specifically, I found the following:

$X = UDV^T$, where matrix U is $N \times p$, V is a $p \times p$ orthogonal matrix, and D is a $p \times p$ diagonal matrix.

I am confused because from Wikipedia, the orthogonal matrix has to be a square matrix. In this case matrix U does not qualify. Later I tend to believe that U contains orthogonal columns only, and that results in $U^TU = I$, but $UU^T \neq I$. This seems to make sense because I found in the book

$X\hat{\beta} = X(X^TX)^{-1}X^TY = UU^TY$, and UU^TY should not be equal to Y

So my question becomes: are there two versions of SVD I can do? One results in both U and V being orthogonal and square matrix, and the other like this? Or is there anything wrong with my argument?

Any guidance is appreciated.

Update after receiving initial answer:

After reading @BabakP 's answer, I thought testing the algorithm using software is a good idea. So I tried svd function in Matlab. The result shows a square U matrix in dimension NxN, a diagonal matrix D in dimension Nxp, and a square V matrix in dimension pxp. Example below:

```
A=[ones(10,1) randn(10,1)];
[U,S,V]=svd(A);
>> size(U)

ans =

10    10

>> size(S)

ans =
```

```
10     2

>> size(V)

ans =

 2     2
```

So does this mean R and Matlab give two different versions?

regression machine-learning svd

Share Cite Edit Follow Flag

edited Sep 9 '13 at 23:51
Glen_b
255k 30 545 926

asked Sep 9 '13 at 22:08
Jerry
368 1 4 11

2 Answers

Active Oldest Votes

3

As far as I know there is only one version of SVD. The correct dimensions for an SVD decomposition are $N \times p_1$, $p_1 \times p_2$ and $M \times p_2$, this makes sense because you want the product of the three matrices to be (a reconstruction of) the original matrix. So if X is $N \times M$, so should the reconstruction be, or to put it differently:

$$N \times M = (N \times p_1) \times (p_1 \times p_2) \times (M \times p_2)^T$$

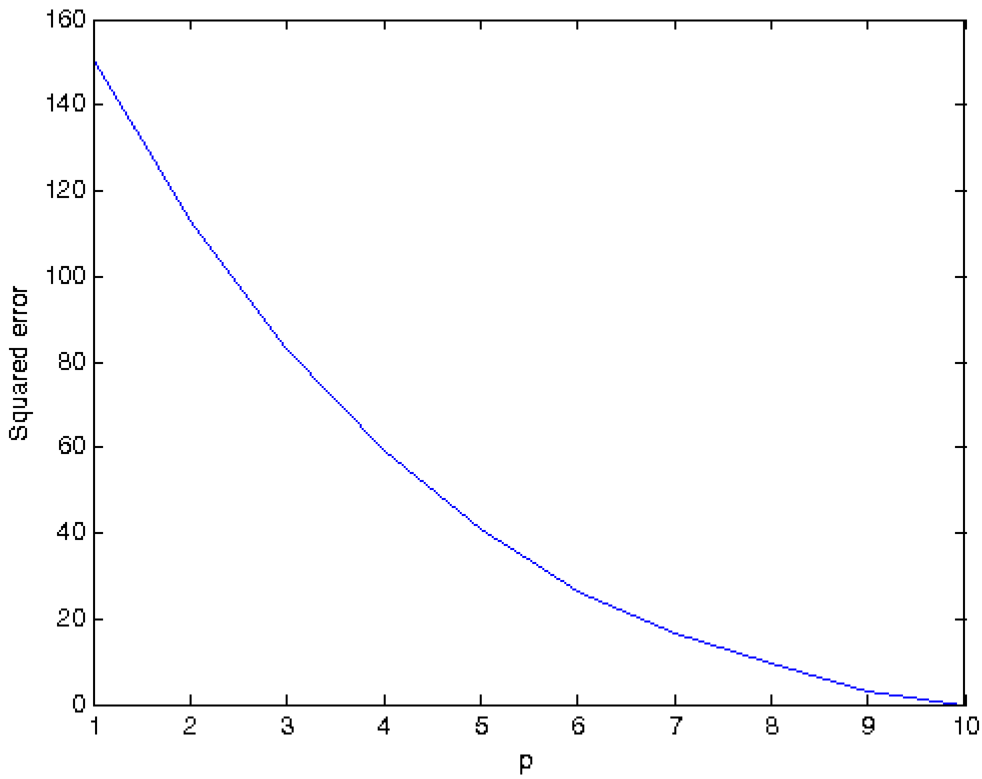
Edit: usually, $p_1 = p_2 = p$, resulting in a square matrix (like in Matlab)

The orthogonal, rectangular matrices contain left and right singular vectors respectively and the middle, rectangular matrix contains the singular values on the diagonal.

Edit2: (see comments)

```
A=[ones(10,1) randn(10,20)];

[U,S,V] = svd(A);
errors = zeros(10,1);
for p = 10:-1:1
    err = U(:,1:p) * S(1:p,1:p) * V(:,1:p)' - A;
    errors(p) = sum(sum(err.*err));
end
plot(errors);
ylabel('Squared error');
xlabel('p');
```



Share Cite Edit Follow Flag

edited Sep 10 '13 at 17:56

answered Sep 9 '13 at 22:22

ciri
1,113 9 20

Do we need symbol 'p' in the example? I thought one version is (if X is NxM), U: NxM, D: MxM, V: MxM. I have found references associated with this version, but still references with alternative explanation like this (web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm). Do you happen to know why? – Jerry Sep 9 '13 at 23:06

1 SVD is used (amongst other uses) as a preprocessing step to reduce the amount of dimensions for your learning algorithm. This why you would introduce a choice of $p << M$, which basically allows you to learn in the reduced p-dimensional space. Here, p is a design choice. If you are familiar with PCA (which I recommend you should be), this would be the equivalent of dropping the $M - k$ least important eigenvectors. Setting p equal to the original dimensions asin your example, allows for a flawless reconstruction, but no dimensionality reduction. – ciri Sep 9 '13 at 23:29

yes, I understood your argument. – Jerry Sep 10 '13 at 1:23

Great, could you mark this question as answered/closed? – ciri Sep 10 '13 at 10:17

I agree there is link between SVD and PCA. But both versions I found allows flawless reconstruction, and they output U matrix of different dimensions. By the way, in your equation, shouldn't p1 be set equal to N rather than p2 to make U as a square matrix like the Matlab example I posted? That's a typo, right? I haven't closed the question as I am still not 100% clear on it. – Jerry Sep 10 '13 at 16:28

Here is some R code that validates your formulas given above:

1

```
 [6,]    1 -0.31559482
 [7,]    1  0.20561526
 [8,]    1  0.55152336
 [9,]    1 -0.69396930
[10,]    1 -1.21970880
> X
      [,1]        [,2]
 [1,]    1 -0.20283033
 [2,]    1 -0.85846798
 [3,]    1  0.07970559
 [4,]    1 -0.28254373
 [5,]    1  0.39261439
 [6,]    1 -0.31559482
 [7,]    1  0.20561526
 [8,]    1  0.55152336
 [9,]    1 -0.69396930
[10,]    1 -1.21970880


#Calculate UU'Y
U = svd(X)$u
XB2 = U%*%t(U)%*%Y

#Check to see if they return the same thing
cbind(XB1,XB2)

> cbind(XB1,XB2)
            [,1]       [,2]
 [1,] -0.4644321 -0.4644321
 [2,] -0.7215807 -0.7215807
 [3,] -0.3536183 -0.3536183
 [4,] -0.4956966 -0.4956966
 [5,] -0.2308919 -0.2308919
 [6,] -0.5086596 -0.5086596
 [7,] -0.3042351 -0.3042351
 [8,] -0.1685660 -0.1685660
 [9,] -0.6570624 -0.6570624
[10,] -0.8632634 -0.8632634
```

So as you can see from the output above, for sure one decomposition of X is $X = UDV^T$. Likewise, calculating UU^TY is equivalent to calculating $X\hat{\beta}$ where $\hat{\beta} = (X^TX)^{-1}X^TY$. So this solution really just pertains to validating your second question about whether or not what you are doing is correct.

Share Cite Edit Follow Flag edited Sep 9 '13 at 22:31 answered Sep 9 '13 at 22:23

user25658

Thanks for this example. I checked your code and it does indicate that U matrix is Nxp, and V matrix is pxp and D matrix is pxp, where N=10, and p=2. However, since I find reference about getting U as square matrix, such as here (web.mit.edu/be.400/www/SVD/Singular_Value_Decomposition.htm) I am still a bit confused. – Jerry Sep 9 '13 at 22:55