

STATS305A - Lecture

John Duchi
Scribed by Michael Howes

11/09/21

Contents

1	Announcements	1
2	Recap	1
2.1	Ridge regression	1
2.2	PCA	2
3	Principal component regression	2
4	Feature selection methods	4
4.1	All subsets	4
4.2	Forward stepwise	5
4.3	Backwards stepwise	5
5	Boosting	6

1 Announcements

- HW 3 out now.
- An etude coming soon.
- There will be four homeworks and four etudes in total for the course.

2 Recap

2.1 Ridge regression

Last time we saw ridge regression where we use the estimator

$$\hat{\beta}_\lambda = \underset{\lambda}{\operatorname{argmin}} \left\{ \|Xb - Y\|_2^2 + \lambda \|b\|_2^2 \right\}.$$

We saw that if $U\Gamma V^T$ is the SVD of X , then we have

$$\begin{aligned} H_\lambda &:= X(X^T X + \lambda I)^{-1} X^T \\ &= U \operatorname{diag} \left(\frac{\gamma_j^2}{\gamma_j^2 + \lambda} \right) U^T, \end{aligned}$$

and our predictions are given by

$$\widehat{Y}_\lambda = H_\lambda Y = \sum_{j=1}^d \frac{\gamma_j^2}{\gamma_j^2 + \lambda} u_j u_j^T Y.$$

We also saw that when doing ridge regression we often replace Y with $Y - \bar{Y}\mathbf{1}$.

2.2 PCA

We also started talking about principal component analysis (PCA). We wanted to project each $x_i \in \mathbb{R}^d$ to the directions in \mathbb{R}^d capturing the most variance in our data X . Equivalently we want to project onto the subspace closest to all the x'_i 's.

3 Principal component regression

Roughly, if y is related to x we'd hope that the variance in y is explained by the most varying/important direction in X .

Recall that if $X = U\Gamma V^T$ is the SVD of X , then the first k principal components are the first k vectors in $V = [v_1, v_2, \dots, v_k]$. The idea behind principal component regression (pcr) is to replace $x_i \in \mathbb{R}^d$ with the projection of x_i onto $\text{span}(V_k)$ where $V_k = [v_1, \dots, v_k]$. Define $\tilde{x}_i = V_k V_k^T x_i$. We will then run regression on

$$\tilde{X} = \begin{bmatrix} \tilde{x}_1^T \\ \vdots \\ \tilde{x}_n^T \end{bmatrix} \in \mathbb{R}^{n \times d}.$$

Thus our estimate of β is

$$\hat{\beta}_{\text{pcr}(k)} = \underset{b}{\operatorname{argmin}} \|Y - \tilde{X}b\|_2^2.$$

We equivalently formulate this as doing regression in \mathbb{R}^k . Note that $\tilde{X} = X V_k V_k^T$ thus we can let $c = V_k^T b \in \mathbb{R}^k$ so that $\tilde{X}b = X V_k c$. We can this instead solve

$$\hat{c} = \underset{c \in \mathbb{R}^k}{\operatorname{argmin}} \|Y - X V_k c\| = \underset{c \in \mathbb{R}^k}{\operatorname{argmin}} \|Y - U\Gamma V^T V_k c\|.$$

Note that

$$V^T V_k = \begin{bmatrix} I_k \\ 0 \end{bmatrix},$$

and so

$$\hat{c} = \underset{c \in \mathbb{R}^k}{\operatorname{argmin}} \|Y - U\Gamma V^T V_k c\| = \underset{c \in \mathbb{R}^k}{\operatorname{argmin}} \|Y - U_k \Gamma_k c\|,$$

where $U_k = [u_1, \dots, u_k] \in \mathbb{R}^{n \times k}$ and $\Gamma_k = \text{diag}(\gamma_1, \dots, \gamma_k)$. Thus

$$\hat{c} = \Gamma_k^{-1} U_k^T Y,$$

and

$$\hat{\beta}_{\text{pcr}(k)} = V_k \Gamma_k^{-1} U_k Y.$$

Thus in PCR we project X onto the span of the first k left singular vector X and then fit Y as well as possible within that span. The predictions from PCR are thus

$$\begin{aligned} \hat{Y}_{\text{pcr}(k)} &= X \hat{\beta}_{\text{pcr}(k)} \\ &= U\Gamma V^T V_k \Gamma^{-1} U_k^T Y \\ &= U_k U_k^T Y \\ &= H_k Y. \end{aligned}$$

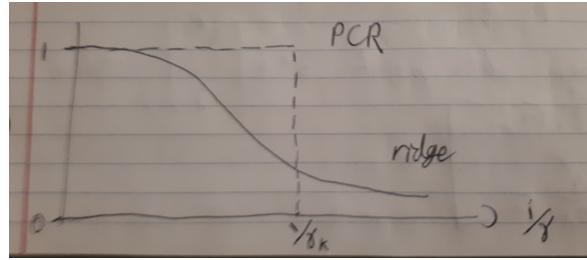
Thus

$$\hat{Y}_{\text{PCR}(k)} = H_k y = \sum_{j=1}^k u_j u_j^T Y.$$

We can compare this to ridge regression

$$\hat{Y}_\lambda = \sum_{j=1}^d \frac{\gamma_j^2}{\gamma_j^2 + \lambda} u_j u_j^T Y.$$

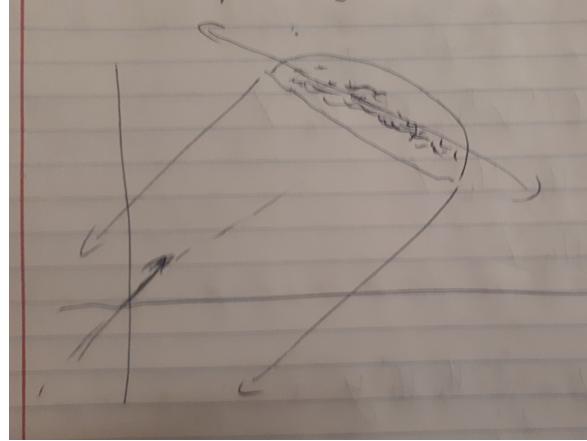
In ridge regression we apply a soft threshold to each of the singular direction u_j where we shrink by the factor $\frac{\gamma_j^2}{\gamma_j^2 + \lambda}$ which increases with j . In PCR we have a hard threshold where we cut off the singular directions u_j for $j > k$. The thresholds look something like this:



Remark 1. A good question from the audience: what about the scale of the columns in

$$X = [x^{(1)}, \dots, x^{(d)}]?$$

In PCR we always standardize data so that $\mathbb{E}[x^{(j)}] = 0$ and $\text{Var}(x^{(j)}) = 1$. This is to avoid situations like this



To do this we set $\tilde{\tau}_j^2 = \|x^{(j)}\|_2^2$ and replace $x^{(j)}$ with $\sqrt{n} \frac{x^{(j)}}{\tilde{\tau}_j}$. If we see a new data point $x^* = [x_1^*, \dots, x_d^*]$, to make predictions we have to first perform the transformation

$$x^* \mapsto \begin{bmatrix} \sqrt{n} \frac{x_1^*}{\tilde{\tau}_1} \\ \vdots \\ \sqrt{n} \frac{x_d^*}{\tilde{\tau}_d} \end{bmatrix}.$$

But sometimes we care about the scale of the variables and wouldn't want to standardize. This is a problem specific issue but boosting is a technique we will see later that can help. Boosting allows this process to be automated.

Remark 2. On the homework we are asked to compute the optimism/bias in PCR and ridge regressions. That is, what are the effective degrees of freedom?

4 Feature selection methods

Given $X = [x^{(1)}, \dots, x^{(d)}] \in \mathbb{R}^{n \times d}$, which features should we include? That is which columns of X should we include in our regression?

A common approach is to pick different index subsets $J \subseteq [d] = \{1, \dots, d\}$. We then fit a model on $X_J = [x^{(j)}]_{j \in J} \in \mathbb{R}^{n \times |J|}$. We then ask if the fitted model is “good enough”. Our goal is to choose H to minimize errors on future data. There are two approaches for deciding which subsets are “good enough”

- Hold out some data in a test set $(x^{\text{test}}, y^{\text{test}})$ and evaluate the model on this test set. We will talk about this and variations such as permutation tests and cross validation in a later lecture.
- Penalization approaches where we penalize the complexity of J using something like Mallows’s C_p statistic.

The notation we will use is

$$\widehat{\text{Risk}}(\beta) = \text{estimated error/risk on future data.}$$

The quantity $\widehat{\text{Risk}}(\beta)$ will be the basis for which we select models. For example we may have

$$\widehat{\text{Risk}}(\beta) = \frac{1}{n} \|X\beta - Y\|_2^2 + \frac{2\hat{\sigma}^2 p}{n},$$

where $p = \|\beta\|_0 = \# \text{ of non-zero entries}$. Or we might have

$$\widehat{\text{Risk}}(\beta) = \frac{1}{n_{\text{test}}} \|X^{\text{test}}\beta - Y^{\text{test}}\|_2^2.$$

4.1 All subsets

In all subsets regression, we do the following:

- Look at all subsets $J \subseteq [d]$.
- Compute $\widehat{\beta}_J = \arg\min_b \|X_J b - Y\|_2^2$.
- Choose $\widehat{\beta}$ to be the minimizer of $\widehat{\text{Risk}}(\widehat{\beta}_J)$.

There are some challenges to all subset regression.

- There are 2^d such subsets which means it can be very expensive. (In practice, branch and bound methods can allow $d \sim 1000$).
- Often the results are optimistic. Since we are searching over so many possible models, the models might overfit even with a good choice of $\widehat{\text{Risk}}$.
- The effectiveness of methods (especially when compared to convex optimization based approaches like ridge regression) depends a lot on $\sigma^2 = \text{Var}(y|x)$. If

$$\frac{\text{Var}(x^T \beta)}{\sigma^2},$$

is larger (most of the variance is explained by x), then all subsets is good otherwise all subsets is not so good.

- Recommended reading: this issue of Statistical Science on all subsets regression. See in particular the parts written by
 - Bertsimas,
 - Mazumdar,
 - Hastie/Tibshirani/Tibshirani.

4.2 Forward stepwise

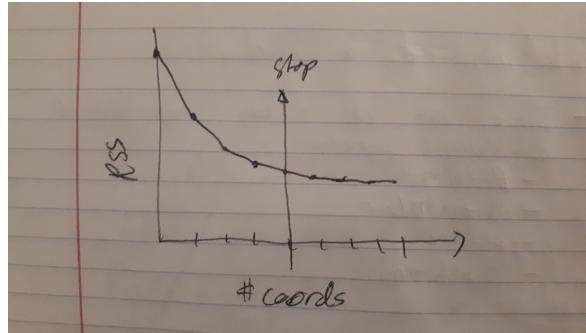
Consider the following greedy algorithm.

- Pick the feature j not in the model that makes the most progress. Add feature j to the model.
- Repeat.

For formally, Set $J = \emptyset$. Then for $p = 1, 2, \dots$:

- Define $J_{+j} = J \cup \{j\}$ and $r_j^2 = \min_b \|X_{J+j}b - y\|_2^2$ for $j \notin J$.
- Set $j^* = \operatorname{argmin}_j r_j^2$.
- Update J to $J \cup \{j^*\}$.

This is a reasonable approach and can be computed quickly. It is often effective even when d is large. The idea is that we would run this until the improvements get small. The plot of the residuals against $|J|$ would look something like this:



When doing the forward stepwise algorithm we will need to make approximate $d^2 \ll 2^d$ computations of r_j^2 . Thus it is much better than a naive implementation of all subsets.

4.3 Backwards stepwise

Unfortunately the forward stepwise algorithm is a greedy algorithm and we might end up with some junk. This gives us the backwards stepwise algorithm.

- First fit using the full model (or the model produced by the forwards stepwise algorithm).
- Iteratively remove coordinates that increase RSS the least.

5 Boosting

This technique was originally developed in 1995 by Freund and Schapire. It is designed to solve the challenge that we often do not know any of the following:

- What scales to use.
- What non-linear transformation to use.

Suppose we have a tool that can fit a small model well (a weak-learner). We then have the procedure:

- Start with raw data $(x, y) \in \mathcal{X} \times \mathbb{R}$.
- Iteratively
 - Find some feature mapping $\phi : \mathcal{X} \rightarrow \mathbb{R}$ capturing some signal in x about y .
 - Add this feature to the model we're building.

More formally, for $j = 1, 2, \dots$, pick $\phi^{(j)} : \mathcal{X} \rightarrow \mathbb{R}$ and add $\phi^{(j)}$ to the model

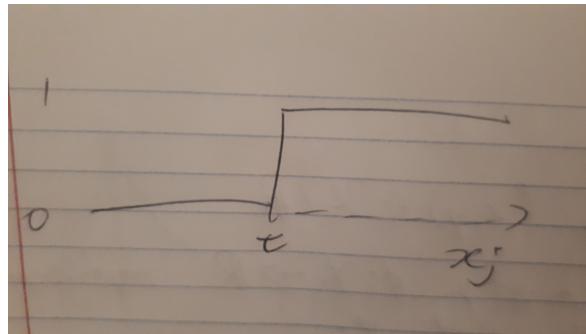
$$M(X) = \begin{bmatrix} \phi^{(1)}(x_1) & \phi^{(2)}(x_1) & \dots & \phi^{(j)}(x_1) \\ \phi^{(1)}(x_2) & \phi^{(2)}(x_2) & \dots & \phi^{(j)}(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi^{(1)}(x_n) & \phi^{(2)}(x_n) & \dots & \phi^{(j)}(x_n) \end{bmatrix} \in \mathbb{R}^{n \times d}.$$

Fit a model and repeat.

How do we actually do this? Roughly: coordinate descent. We first fix a family Φ of feature mappings. For example we may use decision stumps for $x \in \mathbb{R}^d$

$$\phi(x) = \begin{cases} 1 & \text{if } x_j \geq \tau, \\ 0 & \text{if } x_j < \tau. \end{cases}$$

That is Φ consists of 0-1 thresholds on individual coordinates of raw data. The function ϕ looks like this



The idea is that we start with $M(X)$ and ask which feature improves the risk the most. That is we define

$$R(\delta, \phi; M) = \left\| [M, \phi] \begin{bmatrix} \beta \\ \delta \end{bmatrix} - y \right\|_2^2,$$

where β is the minimizer of $\|Mb - y\|_2^2$. There are then two approaches:

- Ask for

$$\hat{\delta} = \underset{\delta}{\operatorname{argmin}} R(\delta, \phi; M).$$

- Or ask what gives the most local progress

$$R'(\delta, \phi; M)|_{\delta=0}.$$

We will discuss this more on Thursday.