Exam 2020, STK-IN4300

-

November 25, 2020

Contents

Problem 1: Penalized regression

1. (a)

Ridge regression will shrink all the parameters, shrinking more the parameters for which the eigenvalues of the eigenvectors of their corresponding covariate in X are small. Lasso on the other hand will also tend to set some parameters to zero as a consequence of the shape of its penalty region.
We see from the sampling distribution of β that a beta exactly equal to zero is highly unlikely, making lasso not very well suited for this problem. In addition, we see from the covariance matrix Σ that the parameters are highly correlated. Ridge regression handles correlated covariates better than lasso, which tends to behave erratic in this setting.
For these reasons I would choose ridge regression for this problem

1. (b)

If we ignore the knowledge that few, if any β_i are zero, we still have the problem of correlated covariates, which ridge regression handles better. Ridge regression will still handle the irrelevant β_i relatively well, by shrinking them close to zero, and will also handle the correlation better.
Therefore I would still choose ridge regression.

1. (c)

Elastic net is a compromise between ridge regression and lasso, bringing the variable selection of lasso together with ridge regression superior handling of correlated variables. As we have no information about the sampling distribution of β we will follow the "bet on sparsity" principle, and assume that some covariates are irrelevant for predicting y (because if we are correct we are choosing the superior method, and if they all are relevant both methods will perform badly).
I would therefore choose elastic net in this situation.

2. (a)

The estimator tries to implement some prior knowledge about the data into the estimator, while still letting the data speak loudly. It is based on a classic kernel smoother

$$\frac{1}{N}\sum K_\lambda(x, x_i).$$

Which estimates the density of a point x_0 based on a weighted average of the sample points in a region around this point. The weights and the region is specified by the chosen kernel, e.g. a normal density can be used as weights, with the variance defining the size of the region of high weight, or the k nearest neighbours method can be used, giving equal weight to the k closest sample points, with k choosing the size of the region.
The idea of the Hjort-Glad estimator is to start with a guess for the parametric density of the data $f_0(x, \widehat{\theta})$ and then correct the guess using the data through the kernel smoother including a correction for the parametric estimate. The kernel smoother with correction is given by

$$\frac{1}{N}\sum \frac{K_\lambda(x, x_i)}{f_0(x_i, \widehat{\theta})}$$

Because using a constant C as the initial parametric guess, implying uniform distribution without specifying the limits, leads to only using the kernel estimator

$$\frac{1}{N}\sum K_\lambda(x, x_i)\frac{C}{C} = \frac{1}{N}\sum K_\lambda(x, x_i).$$

Any guess better than this uniform distribution will improve upon the classic kernel estimator. Even a bad initial guess will usually be somewhat better than equal weight for all possible values, and for this reason the method tends to perform better the classic kernel estimator.

2. (b)

For this problem I would choose the gamma distribution as an initial guess for $f_0(x, \widehat{\theta}$.
A problem solved by using this distribution is the estimated positive density for values below zero. As a person cannot drink a negative amount of wine in a year we should not have any positive probability density below zero. The kernel estimator will average points in a region around the negative values close to zero and will be likely to estimate some positive density here. The gamma density is zero for values below zero, and through multiplication with the kernel estimator will force the final estimator to be zero in this region.
In this sense we are correcting the nonparametric estimator with our prior knowledge about the data.

Problem 3: Cross-validation

3. (a)

In the situation described, all the data is used to estimate the tuning parameter λ, if we perform cross validation on the same data in order to test the model the data used for testing will not be independent of the trained parameter. This correlation with the parameter estimate leads to the training error which is too optimistic in estimating the true prediction error.
If estimation of the prediction error is the goal and splitting away a test set is not feasible, then I would recommend performing k-fold cross validation where the fold left out is used to estimate prediction error, and the other folds representing a training set to be used for estimating λ. Each estimation of lambda can be done using cross validation on the "training set"-folds.
Using leave one out (LOO) cross validation for the training and test folds will ensure that the estimated λ for each fold do not differ much from an estimate based on the entire dataset. We will need to estimate a new $\widehat{\lambda}$ for each sample but if there is little data, as in this case, the procedure will not be too computationally intensive.
Because lasso is fit using a smoothing matrix, we could also use *Generalized cross-validation* to estimate LOO cross validation, but as the task is not computationally heavy in this case I would simply recommend using LOO-cross validation.

Cross validation does not estimate the expected error given the training set, but instead estimates expected error over all possible training set, and as such is not as good for our purpose as a test set would have been, but is still a good surrogate. Using LOO cross validation will force the estimate closer to the error incurred by the estimate based on the entire data-set.

3. (b)

With LOO cross-validation we are estimating our parameter on the largest possible data sets, and the bias will be reduced accordingly. But the data-sets are very similar to each other and the resulting averaged estimate is averaged over very similar values.
With larger folds we will have smaller data-sets with less information for estimation and the bias increases, but we are averaging errors over more different data-sets and this averaging effect will reduce the variance of the estimate.

Problem 4: Wisdom of the Crowd and Bagging

4. (a)

When $P \in [0; 0.25)$ the 15 voters with "knowledge" performs worse than random guessing and the expected performance of an individual will continue to decrease in the same linear fashion. While the consensus vote will decrease more drastically, the voters with (erroneous) "knowledge" will through the help of the random votes draw the consensus vote to be incorrect towards 100% of the time as P decreases, in the same way they drew the vote to 100% correct in the other extreme.
For bagging this means that it is essential for the success of the method that we aggregate learners that perform better than random guessing. The knowledge of the voters is here represented by the randomly chosen bootstrap samples which in each split, represented by the vote, may or may not be informative.

4. (b)

The probability of a randomly chosen individual being correct when voting is

$$\begin{aligned}\text{Pr("Knowledge" individual is chosen)} \cdot \text{Pr("Knowledge" individual individual votes correct)} \\ +\text{Pr(Random guess individual chosen)} \cdot \text{Pr(Random vote is correct)}\end{aligned}$$

$$= \frac{15}{50} * 0 + \frac{35}{50} * 0.25 = 0.175$$

And thus the expected number of correct out of 10 for an individual when $p = 0$ will be 1.75.

4. (c)

The variance of the average of B identically distributed random variables is

$$\frac{1}{B^2}\text{Cov}(\sum_{i=1}^{B} X_i) = \frac{1}{B^2}\text{Var}(\sum_{i=1}^{B} X_i) + 2\frac{1}{B^2}\sum_{i=1}^{B}\sum_{j<i}^{B}\text{Cov}(X_i, X_j) = \frac{\sigma^2}{B} + 2\frac{B-1}{B}\rho$$

Which is not the desired result. I assume I have looked at the wrong problem. From the correct expression we see that by averaging a large number of estimators (incresing B) we reduce the variance of the estmiator.
By reducing the correlation between the variables we can further reduce the variance of the estimator (random forest).

Problem 5: Boosting

5. (a)

In boosting the two tuning parameters are the step size ν and the number of steps m. If the step size is constant, then with enough steps we will eventually reach the unrestricted estimate, in this case the estimate that minimizes the mean squared error on the training data. In order to avoid over-fitting we would like to stop at a point before this occurs, as we expect the data to not be a complete representation of the sampling distribution, and a smoothed estimate is more likely to have better predictive properties. We would still like to take enough steps that we reach a point where we have used the information in the sample to better our predictions, but not so many that we have over-fitted. Of course with a smaller step size we will need to take more steps to reach an equivalent estimate, and vice versa.
A large step size reaches the optimal estimate in fewer computations, which is desirable from a computational standpoint but risks stepping over optimal values.

The curves represent this effect in the following ways.

- The top left graph shows a boosting procedure with very small step size, the procedure makes small methodical steps towards a better estimate, but never reaches the low error estimates attained in the other curves, because the method has not taken enough steps.
- The top right graph shows usage of the recommended step size of 0.01 and steadily reaches a low value before increasing as a result of over-fitting.
- The bottom left graph uses a higher step size of $\nu = 0.1$ and reaches a low value very fast, before over-fitting.
- The bottom right panel shows a procedure takes too large steps with $\nu = 1$ and completely steps over the low values attained by the previously mentioned procedures. attained by the

In this case I would choose a step size of 0.1 and approximately 5 steps, as this attains the lowest mean square prediction error of the alternatives, and has the (not very important) advantage of requiring the fewest computations as well.

5. (b)

The component-wise version of boosting is different in that at each "step" of the algorithm it chooses only one covariate to change.
An advantage of this algorithm is that it can force the estimated effect of some parameters to be zero, in the same manner as lasso, and thus performs automatic variable selection. In fact if all of the basis learners used as a dictionary are mutually uncorrelated (not the case for trees) the resulting estimates will be approximately the same as the lasso estimates for an equivalent lasso penalty λ and for sufficiently small step sizes ν and large number of steps M (requires $\nu \to 0$ and $M \to \infty$ such that $\nu M \to t$ for some constant t).