Statistical Learning, Spring 2017

Homework exercise 4

Due date: 26 June 2017

(* Note: another problem will be added to the final version of this homework, probably on boosting *)

1. **Categorical splitting algorithm for CART**
   Prove the favorable property mentioned in ESL 9.2.4: if we are splitting on a categorical variable $X_j$ with $q$ values and looking for the optimal split in terms of either squared error reduction (for regression) or Gini index (for 2-class classification), then an optimal split out of the possible $2^{q-1}$ splits is always one of the $q-1$ splits defined by:
   - Sort the $q$ groups by their average response: $\bar{y}(1) \leq \bar{y}(2) \leq ... \leq \bar{y}(q)$.
   - Consider only splits along this sequence, i.e., ones which divide according to whether $X_j \in \{g(1), g(2), ..., g(k)\}$ for some $k < q$.

   Hint: Prove this by negation, showing that you can improve a split that does not comply with this condition by “switching” values of $X_j$ between the splits.

2. **Playing around with trees**
   Run a variety of tree-based algorithms on our competition data and show their performance. Compare:
   - Small tree without pruning
   - Large tree without pruning
   - Large tree after pruning with 1-SE rule
   - Bagging small trees (100 iterations)
   - Bagging large trees (100 iterations)

   Do this under five-fold cross validation on our competition training set, and use the results of the five different folds to calculate confidence intervals for performance. Plot all the results in a reasonable way (e.g. using `boxplot()`) and comment on them. Explain your choices of “small” and “large”.

   Hints: a. Start early since bagging may take a while to run. b. Use as a basis the code from class which implements much of this.

3. **ESL 7.5 (7.6 in 2nd edition): Degrees of freedom of Nearest Neighbors**
   Prove that in the standard i.i.d error model (which the book calls “additive error”), the effective degrees of freedom of $k$-NN with $N$ observations is $N/k$.

4. **ESL 7.8 (7.9 in 2nd edition): Trying out model selection methods**
   The use of BIC is optional.
   Tip: for all-subset modeling in R, you can use the function `leaps()` in the package of the same name, which you may need to download from the CRAN repository\(^1\) and install.

---

\(^1\)http://cran.r-project.org/web/packages/
5. “Kernel” representation of regular ridge regression

Consider the ridge regression model in the $p > n$ scenario (assume full rank, so $\text{Rank}(X) = n$).

(a) $(\star + 5)$ Show that the ridge regression solution $\hat{\beta} = (X^T X + \lambda I_p)^{-1} X^T Y$ can be written as $\hat{\beta} = X^T \hat{\alpha}$ where $\hat{\alpha} \in \mathbb{R}^n$ is given by:

$$\hat{\beta} = (X X^T + \lambda I_n)^{-1} Y.$$

Suggested approach (there are other approaches based on algebra identities or optimization theory):

i. Consider the “wasteful” SVD $X = U_{n \times p} D_{p \times p} V_{p \times p}^T$, where the last $p - n$ columns of $U$ and the last $p - n$ rows and columns of $D$ are all zeros. The non-zero columns of $U$ are of course orthonormal (as are all columns of $V$).

ii. Write the ridge solution $\hat{\beta}$ in terms of this SVD: $\hat{\beta} = V D (D^2 + \lambda I_p)^{-1} U^T Y$ (show the derivation).

iii. Notice that $U^T U$ is a matrix with $I_n$ at the top left and zeros elsewhere. Use this to argue that $VD = VDU^T U$.

iv. Now argue that $U(D^2 + \lambda I_p)^{-1} U^T = (X X^T + \lambda I_n)^{-1}$ (explain why).

(b) Since we now know the optimal $\hat{\beta} \in \mathbb{R}^p$ has a representation as $X^T \hat{\alpha}$ with $\hat{\alpha} \in \mathbb{R}^n$, we can now plug the representation $\beta = X \alpha$ into the ridge criterion $\|Y - X \beta\|^2_2 + \lambda \|\beta\|^2_2$.

Do this, and rewrite this criterion as a function of only $Y$, $K = XX^T$ and $\alpha$.

(c) Explain why this means that if we are given $K$ we can solve the problem in a complexity that does not depend on $p$. If we have to calculate $K$ as well from $X$, does this representation still help to solve the problem more efficiently than calculating $\hat{\beta}$ directly? Explain briefly.

(d) If we want to predict at a new point $x_0$, can we do it based on its $n$ inner products $K_{0i} = <x_0, X_i>$ without needing to represent $X$ or $x_0$ explicitly?