Boosting Based Conditional Quantile Estimation for Regression and Binary Classification

Songfeng Zheng

Department of Mathematics
Missouri State University, Springfield, MO 65897, USA
SongfengZheng@MissouriState.edu

Abstract. We introduce Quantile Boost (QBoost) algorithms which predict conditional quantiles of the interested response for regression and binary classification. Quantile Boost Regression (QBR) performs gradient descent in functional space to minimize the objective function used by quantile regression (QReg). In the classification scenario, the class label is defined via a hidden variable, and the quantiles of the class label are estimated by fitting the corresponding quantiles of the hidden variable. An equivalent form of the definition of quantile is introduced, whose smoothed version is employed as the objective function, which is maximized by gradient ascent in functional space to get the Quantile Boost Classification (QBC) algorithm. Extensive experiments show that QBoost performs better than the original QReg and other alternatives for regression and classification. Furthermore, QBoost is more robust to noisy predictors.

Keywords: Boosting, Quantile Regression, Classification.

1 Introduction

Least square regression aims to estimate the conditional expectation of the response Y given the predictor (vector) $\mathbf{x}$, i.e., $\mathrm{E}(Y|\mathbf{x})$. However, the mean value (or the conditional expectation) is sensitive to the outliers of the data [12]. Therefore, if the data is not homogeneously distributed, we expect the least square regression gives us a poor prediction.

The τ-th quantile of a distribution is defined as the value such that there is $100\tau\%$ of mass on the left side of it. Compared to the mean value, quantiles are more robust to outliers [12]. For a random variable Y, it can be proved [11] that

$$Q_\tau(Y) = \arg\min_c \mathrm{E}_Y[\rho_\tau(Y-c)],$$

where $Q_\tau(Y)$ is the τ-th quantile of Y, $\rho_\tau(r)$ is the "check function" [12] defined by

$$\rho_\tau(r) = rI(r \geq 0) - (1-\tau)r, \tag{1}$$

where $I(\cdot) = 1$ if the condition is true, otherwise $I(\cdot) = 0$.

Given data $\{(\mathbf{x}_i, Y_i), i = 1, \cdots, n\}$, with predictor $\mathbf{x}_i \in \mathbf{R}^d$ and response $Y_i \in \mathbf{R}$, let the τ-th conditional quantile of Y given $\mathbf{x}$ be $f(\mathbf{x})$. Similar to least square regression, quantile regression (QReg) [12] aims at estimating the conditional quantiles of the response given predictor vector $\mathbf{x}$ and can be summarized as

$$f^*(\cdot) = \arg\min_f \frac{1}{n} \sum_{i=1}^{n} \rho_\tau \left(Y_i - f(\mathbf{x}_i)\right), \tag{2}$$

which can be solved by linear programming algorithms [12] or MM algorithms [11]. However, when the predictor $\mathbf{x}$ is in high dimensional space, the aforementioned optimization methods for QReg might be inefficient. High dimension problems are ubiquitous in applications, e.g., image analysis, gene sequence analysis, to name a few. To the best of our knowledge, the problem of high dimensional predictor is not sufficiently addressed in quantile regression literature.

Motivated by the basic idea of gradient boosting algorithms [8], we propose to estimate the quantile regression function by minimizing the objective function in Eqn. (2) with functional gradient descent. In each step, we approximate the negative gradient of the objective function by a base function, and grow the model along that direction. This results Quantile Boost Regression (QBR) algorithm. In the binary classification scenario, we define the class label via a hidden variable, and the quantiles of the class label can be estimated by fitting the corresponding quantiles of the hidden variable. An equivalent form of the definition of quantile is introduced, whose smoothed version is employed as the objective function for classification. Similar to QBR, functional gradient ascent is applied to maximize the objective function, yielding the Quantile Boost Classification (QBC) algorithm. The obtained Quantile Boost (QBoost) algorithms are computationally efficient and converge to a local optimum, more importantly, they enable us to solve high dimensional problems efficiently.

The QBoost algorithms were tested extensively on publicly available datasets for regression and classification. On the regression experiments, QBR performs better than the original QReg in terms of check loss function. Moreover, the comparative experiment on noisy data indicates that QBR is more robust to noise. On classification problems, QBC was compared to binary QReg on a public dataset, the result shows that QBC performs better than binary QReg and is more robust to noisy predictors. On three high dimensional datasets from bioinformatics, binary QReg is not applicable due to its expensive computation, while QBC performs better than or similar to other alternatives in terms of 5 fold cross validation error rates. Furthermore, both QBC and QBR are able to select the most informative variables, inheriting the feature selection ability of boosting algorithm.

2 Boosting as Functional Gradient Descent

Boosting [7] is well known for its simplicity and good performance. The powerful feature selection mechanism of boosting makes it suitable to work in high

Algorithm 1. Generic Functional Gradient Descent

```
0: Initialize $f^{[0]}(\cdot)$ with
       $$f^{[0]}(\cdot) = \arg\min_c \frac{1}{n}\sum_{i=1}^{n} l(Y_i, c),$$
   or set $f^{[0]}(\cdot) = 0$, and set $m = 0$.
1: Increase $m$ by 1. Compute the negative gradient $-\frac{\partial}{\partial f} l(Y, f)$ and evaluate at $f^{[m-1]}(\mathbf{x}_i)$:
       $$U_i = -\left.\frac{\partial l(Y_i, f)}{\partial f}\right|_{f=f^{[m-1]}(\mathbf{x}_i)}, \quad i = 1, \cdots, n.$$
2: Fit the negative gradient vector $U_1, \cdots, U_n$ to $\mathbf{x}_1, \cdots, \mathbf{x}_n$ by the base procedure (e.g. the weak learner in AdaBoost):
       $$\{(\mathbf{x}_i, U_i),\ i = 1, \cdots, n\} \longrightarrow g^{[m]}(\cdot).$$
3: Update the estimation by $f^{[m]}(\cdot) = f^{[m-1]}(\cdot) + \eta g^{[m]}(\cdot)$, where $\eta$ is a step-length factor.
4: Check the stopping criterion, if not satisfied, go to step 1.
```

dimensional spaces. Friedman et al. [8,9] developed a general statistical framework which yields a direct interpretation of boosting as a method for function estimate, which is a "stage-wise, additive model".

Consider the problem of function estimation

$$f^*(\mathbf{x}) = \arg\min_f \mathrm{E}[l(Y, f(\mathbf{x}))|\mathbf{x}],$$

where $l(\cdot,\cdot)$ is a loss function which is typically differentiable and convex with respect to the second argument. Estimating $f^*(\cdot)$ from the given data $\{(\mathbf{x}_i, Y_i),\ i = 1, \cdots, n\}$ can be performed by minimizing the empirical loss $n^{-1}\sum_{i=1}^{n} l(Y_i, f(\mathbf{x}_i))$ and pursuing iterative steepest descent in functional space. This leads us to the generic functional gradient descent algorithm [1,8] as shown in Algorithm 1.

Many boosting algorithms can be understood as functional gradient descent with appropriate loss function. For example, if we choose $l(Y, f) = \exp(-(2Y - 1)f)$, we would recover AdaBoost [9]; if $l(Y, f) = (Y - f)^2/2$ is used, we would result in L_2 Boost [2].

3 Quantile Boost for Regression

We consider the problem of estimating quantile regression function in the general framework of functional gradient descent with the loss function

$$l(y, f) = \rho_\tau(y - f) = (y - f)I(y - f \geq 0) - (1 - \tau)(y - f).$$

A direct application of Algorithm 1 yields the Quantile Boost Regression (QBR) algorithm, which is shown as Algorithm 2.

Algorithm 2. Quantile Boost Regression (QBR)

```
0: Given training data $\{(\mathbf{x}_i, Y_i),\ i = 1, \cdots, n\}$ and the desired quantile value $\tau$.
1: Initialize $f^{[0]}(\cdot)$ with
       $f^{[0]}(\cdot) = \tau\text{-th quantile of } (Y_1, \cdots, Y_n)$,
   or set $f^{[0]}(\cdot) = 0$.
2: for $m = 1$ to $M$ do
3:     Compute the negative gradient $-\frac{\partial}{\partial f}\rho_\tau(Y - f)$ and evaluate at $f^{[m-1]}(\mathbf{x}_i)$:
           $U_i = I(Y_i - f^{[m-1]}(\mathbf{x}_i) \geq 0) - (1 - \tau)$.
4:     Fit the negative gradient vector $U_1, \cdots, U_n$ to $\mathbf{x}_1, \cdots, \mathbf{x}_n$ by the base procedure
           $\{(\mathbf{x}_i, U_i),\ i = 1, \cdots, n\} \longrightarrow g^{[m]}(\cdot)$.
5:     Update the estimation by $f^{[m]}(\cdot) = f^{[m-1]}(\cdot) + \eta g^{[m]}(\cdot)$, where $\eta$ is a step-length
       factor.
6: end for
7: Output the estimated $\tau$-th quantile function $f^{[M]}(\mathbf{x})$.
```

Similar to [8], let the base procedure be $h(\mathbf{x}, \mathbf{a})$, where $\mathbf{a}$ is a parameter vector. Then the fourth step can be performed by an ordinary least square regression:

$$\mathbf{a}_m = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} \left[U_i - h(\mathbf{x}_i, \mathbf{a})\right]^2,$$

hence the function $g^{[m]}(\mathbf{x}) = h(\mathbf{x}, \mathbf{a}_m)$ can be regarded as an approximation of the negative gradient by the base procedure. In step 5, the step-length factor η can be determined by line search

$$\eta = \arg\min_{\gamma} \sum_{i=1}^{n} \rho_\tau \left[Y_i - f^{[m-1]}(\mathbf{x}_i) - \gamma g^{[m]}(\mathbf{x}_i)\right].$$

However, line search algorithm is often time consuming, instead, in each iteration, we update the fitted function $f^{[m-1]}(\cdot)$ by a fixed but small step along the negative gradient direction. To guarantee the performance of the resulting model, we fix η at a small value as suggested by [1,8]. Similar to AdaBoost, QBR enables us to select most informative predictors if appropriate base learner is employed, and this will be demonstrated experimentally in Section 5.1.

There is a large volume of literature applying boosting to regression problems, for example, in [5,7,18], among others. However, all these methods estimate mean value of the response, not quantiles. Langford et al. [15] proposed to use classification technique in estimating the conditional quantile. For each given quantile value, their method trains a set of classifiers $\{c_t\}$ for a series of $t \in [0, 1]$, and the testing stage calculates the average of the outputs of the classifiers. Therefore, compared to the proposed QBR algorithm, this method is time consuming. Furthermore, it is not clear how to perform variable selection using the method in [15].

4 Quantile Boost for Classification

This section generalizes the QBR algorithm for classification problems.

4.1 Predicting Quantiles for Classification

Consider the following model:

$$Y^* = h(\mathbf{x}) + \epsilon \quad \text{and} \quad Y = I\{Y^* \geq 0\},$$

where Y^* is a continuous hidden variable, $h(\cdot)$ is the true model for Y^*, ϵ is a disturb, and Y is the observed label of the data. Let $Q_\tau(Y^*|\mathbf{x})$ be the τ-th conditional quantile of Y^* given $\mathbf{x}$, and let $g(\cdot)$ be a real nondecreasing function. Clearly

$$P(Y^* \geq y|\mathbf{x}) = P\left(g(Y^*) \geq g(y)|\mathbf{x}\right),$$

it follows that

$$g\left(Q_\tau(Y^*|\mathbf{x})\right) = Q_\tau(g(Y^*)|\mathbf{x}). \tag{3}$$

Since the indicator function $I(t \geq 0)$ is nondecreasing w.r.t. t, by Eqn. (3), we have

$$I\left(Q_\tau(Y^*|\mathbf{x}) \geq 0\right) = Q_\tau(I(Y^* \geq 0)|\mathbf{x}) = Q_\tau(Y|\mathbf{x}),$$

that is, the conditional quantile function of Y can be obtained by fitting the corresponding conditional quantile of Y^*. If we model the τ-th conditional quantile of the latent variable Y^* by $f(\mathbf{x}, \boldsymbol{\beta})$ with $\boldsymbol{\beta}$ as the parameter vector, i.e.

$$Q_\tau(Y^*|\mathbf{x}) = f(\mathbf{x}, \boldsymbol{\beta}), \tag{4}$$

it follows that the conditional quantile of the binary variable Y can be modeled as

$$Q_\tau(Y|\mathbf{x}) = I\left(f(\mathbf{x}, \boldsymbol{\beta}) \geq 0\right). \tag{5}$$

From the relation $Y = I(Y^* \geq 0)$, it follows that

$$P(Y = 1|\mathbf{x}) = P\left(Y^* \geq 0|\mathbf{x}\right), \tag{6}$$

while Eqn. (4) implies that

$$P(Y^* \geq f(\mathbf{x}, \boldsymbol{\beta})|\mathbf{x}) = 1 - \tau. \tag{7}$$

Thus, if $f(\mathbf{x}, \boldsymbol{\beta}) = 0$, combining Eqn. (6) and (7) yields

$$P(Y = 1|\mathbf{x}) = P(Y^* \geq 0|\mathbf{x}) = P(Y^* \geq f(\mathbf{x}, \boldsymbol{\beta})|\mathbf{x}) = 1 - \tau;$$

if $f(\mathbf{x}, \boldsymbol{\beta}) > 0$,

$$P(Y = 1|\mathbf{x}) = P(Y^* \geq 0|\mathbf{x}) > P(Y^* \geq f(\mathbf{x}, \boldsymbol{\beta})|\mathbf{x}) = 1 - \tau.$$

In summary, we have the following relation:

$$P\left(Y = 1 \middle| f(\mathbf{x}, \boldsymbol{\beta}) \gtreqless 0\right) \gtreqless 1 - \tau, \tag{8}$$

which is an inequality of the posterior probability of response given the predictor vector. Hence, if we make decision with cut-off posterior probability $1 - \tau$, we need to fit a quantile regression model with quantile value τ. Once the model is fit, i.e., the parameter vector $\boldsymbol{\beta}$ is estimated as $\mathbf{b}$, we can make prediction by

$$\hat{Y} = I(f(\mathbf{x}, \mathbf{b}) \geq 0),$$

where $\hat{Y}$ is the predicted label for the predictor vector $\mathbf{x}$.

4.2 Quantile Boost for Classification

To fit the model for the τ-th quantile of Y, i.e., to estimate the parameter vector $\boldsymbol{\beta}$ in Eqn. (5), similar to QBR, the estimated parameter vector $\mathbf{b}$ can be obtained by solving:

$$\mathbf{b} = \arg\min_{\boldsymbol{\beta}} \left\{ L(\boldsymbol{\beta}) = \sum_{i=1}^{n} \rho_\tau \left[Y_i - I(f(\mathbf{x}, \boldsymbol{\beta}) \geq 0) \right] \right\}.$$

We can show (details are omitted due to space limit) that the above minimization problem is equivalent to the following maximization problem [14]:

$$\mathbf{b} = \arg\max_{\boldsymbol{\beta}} \left\{ S(\boldsymbol{\beta}) = \sum_{i=1}^{n} [Y_i - (1 - \tau)] I(f(\mathbf{x}, \boldsymbol{\beta}) \geq 0) \right\}. \tag{9}$$

However, the function $S(\boldsymbol{\beta})$ is not differentiable. To apply gradient based optimization methods, we replace $I(f(\mathbf{x}, \boldsymbol{\beta}) \geq 0)$ by its smoothed version, solving

$$\mathbf{b} = \arg\max_{\boldsymbol{\beta}} \left\{ S(\boldsymbol{\beta}, h) = \sum_{i=1}^{n} [Y_i - (1 - \tau)] K\left(\frac{f(\mathbf{x}, \boldsymbol{\beta})}{h} \right) \right\}, \tag{10}$$

where h is a small positive value, and $K(\cdot)$ is smoothed version of the indicator function $I(t \geq 0)$ with the following properties:

$$K(t) > 0, \ \forall t \in \mathbf{R}, \quad \lim_{t \to \infty} K(t) = 1, \quad \lim_{t \to -\infty} K(t) = 0.$$

In this paper, we take $K(\cdot)$ as the standard normal cumulative distribution function

$$\Phi(z) = \int_{-\infty}^{z} \varphi(t)dt \quad \text{with} \quad \varphi(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}}. \tag{11}$$

Let each term in the objective function (10) be

$$l(Y, f) = [Y - (1 - \tau)] K\left(f/h\right).$$

Following the general steps of functional gradient ascent, we obtain the Quantile Boost Classification (QBC) algorithm as sketched in Algorithm 3. Similar to QBR, the parameter η in QBC can be fixed at a small value instead of performing a line search.

Algorithm 3. Quantile Boost Classification (QBC)

```
0: Given training data $\{(\mathbf{x}_i, Y_i), i = 1, \cdots, n\}$ with $Y_i \in \{0, 1\}$, and the desired quantile value $\tau$.
1: Initialize $f^{[0]}(\cdot)$ with $f^{[0]}(\cdot) = 0$.
2: for $m = 1$ to $M$ do
3:    Compute the gradient $\frac{\partial}{\partial f} l(Y, f)$ and evaluate at $f^{[m-1]}(\mathbf{x}_i)$:
          $$U_i = \frac{Y_i - (1 - \tau)}{h} K'\left(\frac{f^{[m-1]}(\mathbf{x}_i)}{h}\right)$$
4:    Fit the gradient vector $U_1, \cdots, U_n$ to $\mathbf{x}_1, \cdots, \mathbf{x}_n$ by the base procedure:
          $$\{(\mathbf{x}_i, U_i),\ i = 1, \cdots, n\} \longrightarrow g^{[m]}(\cdot).$$
5:    Update the estimation by $f^{[m]}(\cdot) = f^{[m-1]}(\cdot) + \eta g^{[m]}(\cdot)$, where $\eta$ is a step-length factor.
6: end for
7: Output the obtained classifier $I(f^{[M]}(\mathbf{x}) \geq 0)$.
```

4.3 Related Works

Kordas et al. [13,14] proposed binary quantile regression to predict quantiles for classification tasks. However, in binary QReg, simulated annealing algorithm is employed to perform the optimization task. Although a local optimum is guaranteed, simulated annealing is well known for its expensive computation. While QBC is based on gradient ascent, which yields a local maximum and converges fast. Due to the expensive computation of simulated annealing, binary QReg can only work in very low dimensional spaces. However, in applications, we frequently face hundreds of, even thousands of predictors, and it is often desired to find out the informative predictors. Clearly, in this case, binary QReg is not applicable. On the contrary, QBC is designed to work in high dimensional spaces, and it enables us to select the most informative variables by using certain types of base learner.

Hall et al. [10] proposed a median based classifier which works in high dimensional space. For a given predictor vector, Hall et al. compares the L_1 distances from the new predictor vector to the component-wise medians of the positive and negative examples in the training set, and assigns class label as the class with the smaller L_1 distance. Although computationally efficient, this simple nearest neighbor like algorithm cannot perform variable selection as the proposed QBC algorithm. The method proposed in [15] can also be applied to classification tasks, however it is computationally more expensive than QBC, and it is not clear how to select most informative predictors as well.

5 Experiments

This section tests the proposed QBoost algorithms on various regression and classification problems. In the generic gradient descent/ascent algorithm, the step size factor is of minor importance as long as it is small [1]. Thus, in all the following experiments, the step size parameter of QBR and QBC is fixed at $\eta = 0.1$.

5.1 Results of QBR

QBR was tested on five datasets from UCI machine learning repository: White Wine Quality (size: 4898, dimension: 11), Red Wine Quality (size: 1599, dimension: 11), Forest Fire (size: 517, dimension: 12), Concrete Slump (size: 103, dimension: 10), and Concrete Compressive (size: 1030, dimension: 9). The predictor variables and the response variables were normalized to be in $[-1, 1]$. The original QReg was also tested on these datasets. To make the comparison between QBR and QReg fair, we used simple linear regression as base procedure in QBR.

Table 1. The comparison between QReg and QBR on the 5 datasets from UCI machine learning repository. The listed are the mean values of the check losses of the 500 runs, and the standard deviations are listed in parentheses.

Dataset	$\tau = 0.25$		$\tau = 0.50$		$\tau = 0.75$	
	QReg	QBR	QReg	QBR	QReg	QBR
Red Wine	19.71(1.11)	19.41(1.11)	24.30(1.06)	24.07(1.09)	19.40(0.81)	19.16(0.79)
White Wine	62.40(1.84)	62.23(1.91)	78.69(1.78)	78.60(1.92)	62.03(1.50)	61.78(1.64)
Forest Fire	4.97(0.54)	4.93(0.54)	10.04(0.82)	9.62(0.78)	9.56(0.92)	9.14(0.73)
concrete slump	0.74(0.14)	0.71(0.12)	1.00(0.17)	0.92(0.16)	0.95(0.19)	0.84(0.17)
concrete comp.	15.02(0.77)	14.91(0.79)	18.21(1.08)	18.17(1.00)	13.63(0.83)	13.52(0.77)

To numerically evaluate the performances, in lack of the true quantile functions for these datasets, we adopt the sum of the "check loss" on the testing set as the error measure, which is defined as

$$L(\tau) = \sum_{i=1}^{N_{\text{test}}} \rho_\tau(Y_i - \hat{f}_\tau(\mathbf{x}_i)),$$

where N_{test} is the size of the testing set, and $\hat{f}_\tau(\mathbf{x}_i)$ is the predicted τ-th quantile at $\mathbf{x}_i$. By the definition of quantile, the smaller the value of $L(\tau)$ is, the closer the estimated quantile to the true value of quantile.

For each dataset, we randomly select 80% of the examples as training set, and the remaining 20% as testing set. QBR and QReg are separately trained and evaluated on these subsets. The partition-training-testing process is repeated 500 times. The means and the standard deviations of the 500 check losses are calculated, as shown in Table 1. It is readily observed that in all experiments, QBR uniformly achieves smaller average check loss compared to QReg, which indicates that QBR estimates more accurately.

In our experiments, both QReg and QBR obtain a linear function of the predictors. Since the predictors and response are normalized to be in $[-1, 1]$, it makes sense if we delete the variables with too small coefficients, thus performing variable selection. 20 noisy predictors are added to the Concrete Slump data, each is generated uniformly at random from $[-1, 1]$. These noisy predictors are considered as noninformative, and the original predictors are informative to the problem. Then we repeat the above experiment. In the obtained models, we calculate the sum of the absolute values of all the coefficients. For any predictor, if its coefficient absolute value is less than 1% of that sum, it is trimmed out. We calculate the average numbers of the selected noisy predictors of QReg and QBR over the 500 runs, the means and standard deviations of the check losses on testing set are also calculated. Table 2 summarizes the results, from which it is readily observed that for each quantile value, compared to QReg, QBR selects far fewer noisy predictors while achieves smaller mean error. This experiment shows that QBR is more robust to noisy predictors and keeps the variable selection ability of boosting algorithm.

Table 2. Comparison between QReg and QBR on variable selection: see text for details

Method	$\tau = 0.25$		$\tau = 0.50$		$\tau = 0.75$	
	# of N. V.	Error	# of N. V.	Error	# of N. V.	Error
QReg	8.13	0.99 (0.16)	6.97	1.38 (0.21)	9.69	1.28 (0.24)
QBR	1.04	0.97 (0.19)	0.63	1.14 (0.21)	0.98	1.27 (0.30)

The procedure of QBR enables us to select other forms of base learner, for example, regression trees [8,9], and this provides us flexibility in certain applications. While it is not clear how to make use of other forms of regression in the framework of the original QReg.

5.2 Results of QBC

In our experiments for classification, we made decision at the cut-off posterior probability 0.5, therefore we fit QBC with $\tau = 0.5$. The standard normal cumulative distribution function in Eqn. (11) was used as approximation to the indicator function with $h = 0.1$. From our experience, QBC is not sensitive to the value of h as long as $h < 0.5$. For the comparative purpose, since QBC is a boosting based procedure, we mainly considered several popular boosting based classifiers which include L_2-Boost [2], AdaBoost [7], and LogitBoost [9]. We also tested the Median Classifier [10] and compared the performance.

Results on Credit Score Data. We compare the result of QBC to that of binary QReg [13] on the German bank credit score dataset which is available from UCI machine learning repository. The dataset is of size 1000 with 300 positive examples, each example has 20 predictors normalized to be in $[-1, 1]$. In [13], only 8 predictors are preselected to fit the binary QReg due to the expensive simulated annealing algorithm it employees in the optimization procedure.

To run QBC, we randomly split the whole dataset without variable preselection into training set of size 800, and evaluate the learned QBC classifier on the other 200 examples. The QBC training algorithm was ran for 100 iterations using simple linear regression with only one predictor as base learner, by this way, it is fair to compare the performance of QBC to that of binary QReg. The splitting-training-testing process was repeated 500 times, and we report the mean training and testing error rates in Table 5.2, which also lists the performance of binary QReg from [13]. We see that QBC performs better than binary QReg on both the training and testing set. This is due to the efficient computation of QBC which allows the algorithm to explore more predictors and thus selects more informative ones.

In order to test the variable selection ability of QBC, 20 noisy predictors generated from uniform distribution on $[-1, 1]$ are added to the original dataset,

Table 3. Comparison between binary QReg and QBC on credit dataset. Clean means the original dataset, and Noisy means 20 noisy predictors are added to the dataset.

Dataset	QBC		Binary QReg	
	Training Error	Testing Error	Training Error	Testing Error
Clean	19.84%	24.47%	21.9%	26.5%
Noisy	22.53%	25.64%	NA	NA

Table 4. The testing errors of L_2 Boost, AdaBoost, LogitBoost, QBC, and Median Classifier on the credit dataset

Data	L_2 Boost	AdaBoost	LogitBoost	QBC	Median Classifier
Clean	28.68%	29.99%	28.81%	**28.50%**	35.04%
Noisy	31.92%	30.05%	32.68%	**28.55%**	38.94%

Table 5. The 5-fold cross validation mean error rates of the considered algorithms on the three bioinformatics datasets. All the algorithms were ran for 100 iterations. The best mean error rates for a dataset are displayed in **bold**.

Dataset	Size	dim	L_2-Boost	AdaBoost	LogitBoost	QBC	Median Classifier
Estrogen	49	7,129	21.11%	17.11%	15.11%	**13.33%**	13.38%
Colon	62	2,000	24.62%	21.41%	21.41%	21.67%	**14.58%**
Nodal	49	7,129	31.78%	24.89%	20.44%	**20.00%**	42.84%

and the above procedure was repeated 500 times. Table 5.2 also lists the average training and testing errors of QBC with only 1 noisy variable selected in average. Our result demonstrates that QBC performs only slightly worse on noisy data, which indicates that QBC is robust to noise. Due to the expensive computation of binary QReg, its performance on noisy data is not provided.

Compared to binary QReg, QBC enjoys the flexibility to choose other forms of weak learner. We also compared QBC to the alternatives mentioned at the beginning of Section 5.2 on the credit dataset with and without noisy predictors. All the boosting based algorithms used stump as base procedure for fair comparison. All algorithms were ran for 500 times with randomly selected 800 training examples and 200 testing examples, and the average testing error rates are listed in Table 5.2, from which we can see that compared to the alternative methods, QBC achieves the best performance on both clean and noisy data. Again, we observe that QBC deteriorates only slightly on the noisy data, which verifies its robustness to noisy predictors and is able to select informative variables.

Results on Bioinformatics Data. We compare QBC to the alternative methods on 3 publicly available datasets in bioinformatics [3]: Estrogen, Nodal, and Colon. All of the datasets are of very high dimension (see Table 5), and this makes binary QReg [13,14] not affordable. All the boosting-based algorithms used decision stump as base learner for fair comparison.

We have conducted 5-fold cross validation (5-CV), and Table 5 lists the average testing error rates for the 5 runs of each algorithm on every dataset. We observe that QBC yields the best performance on 2 out of the 3 datasets. On the Colon dataset, QBC performs better than L_2 Boost, similar to LogitBoost and AdaBoost, and worse than Median Classifier. It can also be observed that the Median Classifier [10] is not stable – sometimes the performance is very good, sometimes the performance is very poor.

6 Conclusion and Future Works

The original quantile regression (QReg) is inefficient when the predictor is in high dimensional space. This paper applies functional gradient descent to fit QReg, resulting Quantile Boost Regression (QBR). In the case of binary classification, the class label is defined via a hidden variable, and predicting the quantiles of the binary response is reduced to predicting the corresponding quantiles of the hidden variable. An equivalent form of the definition of quantile is introduced, whose smoothed version is employed as the objective function. Functional gradient ascent is used to maximize the objective function for fitting the classifier, yielding Quantile Boost Classification (QBC) algorithm. In QBR (QBC), the gradient of the objective function is approximated by a base procedure and the model grows along the negative gradient (gradient) direction. The proposed algorithms yield a local optimum, and converge fast, more importantly, they enable us to solve problems in high dimensional spaces.

Extensive experiments were conducted for regression and binary classification. Detailed analysis of the results show that QBR/QBC performs better than the original/binary QReg. Moreover, QBR and QBC are more robust to noisy predictors and able to select informative variables. On high dimensional datasets in bioinformatics, binary QReg is not applicable due to its expensive computation, while QBC performs better when compared to other alternatives.

QBR belongs to stage-wise additive model, which has a close connection to L_1 constrained models [6]. Recently, L_1 quantile regression models [16,17] were proposed which imposes L_1 constraint on the coefficient vector in quantile regression. Therefore, it is natural to investigate the relation between QBR and L_1 QReg in the aspects of performance and variable selection ability. The current version of QBC is for two-class problems, and we plan to develop the multi-class version of QBC by reducing it to a series of two-class tasks, for example, by one-vs-all [7] or Error-Correcting Output Codes [4].

Acknowledgement

This work was supported by CNAS Summer Research Fellowship of Missouri State University.

References

1. Bühlmann, P., Hothorn, T.: Boosting Algorithms: Regularization, Prediction and Model Fitting. Statistical Science 22, 477–505 (2007)
2. Bühlmann, P., Yu, B.: Boosting with the L2 Loss: Regression and Classification. Journal of the American Statistical Association 98, 324–340 (2003)
3. Dettling, M., Bühlmann, P.: Boosting for Tumor Classification with Gene Expression Data. Bioinformatics 19(9), 1061–1069 (2003)
4. Dietterich, T.G., Bakiri, G.: Solving Multiclass Learning Problems via Error-Correcting Output Codes. Journal of Artificial Intelligence Research 2, 263–286 (1995)
5. Duffy, N., Helmbold, D.: Boosting Methods for Regression. Machine Learning 47(2-3), 153–200 (2002)
6. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least Angle Regression. Annals of Statistics 32(2), 407–499 (2004)
7. Freund, Y., Schapire, R.: A Decision-Theoretic Generalization of On-line Learning and An Application to Boosting. Journal of Computer and System Sciences 55(1), 119–139 (1997)

8. Friedman, J.: Greedy Function Approximation: A Gradient Boosting Machine. Annals of Statistics 29, 1189–1232 (2001)
9. Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: a Statistical View of Boosting. Annal of Statistics 28(2), 337–407 (2000)
10. Hall, P., Titterington, D.M., Xue, J.H.: Median-based Classifiers for High-Dimensional Data (2008), `http://www.gla.ac.uk/media/media_71170_en.pdf`
11. Hunter, D.R., Lange, K.: Quantile Regression via an MM Algorithm. Journal of Computational & Graphical Statistics (2000)
12. Koenker, R.: Quantile Regression. Cambridge University Press, Cambridge (2005)
13. Kordas, G.: Credit Scoring Using Binary Quantile Regression. In: Statistical Data Analysis Based on the L_1-Norm and Related Methods (2002)
14. Kordas, G.: Smoothed Binary Regression Quantiles. Journal of Applied Econometrics 21(3), 387–407 (2006)
15. Langford, J., Oliveira, R., Zadrozny, B.: Predicting Conditional Quantiles via Reduction to Classification. In: In Proc. of Uncertainty in Artifical Intelligence (2006)
16. Li, Y., Zhu, J.: L_1-Norm Quantile Regression. Journal of Computational & Graphical Statistics 17(1), 163–185 (2008)
17. Wu, Y., Liu, Y.: Variable Selection in Quantile Regression. Statistica Sinica 19, 801–817 (2009)
18. Zemel, R., Pitassi, T.: A Gradient-based Boosting Algorithm for Regression Problems. In: Proc. of Advances in Neural Information Processing Systems (2001)