Learned-Loss Boosting

Abstract

We consider a generalized problem in regression in which a to goal is not to give a *prediction* for new predictor variables, but a *probability distribution* over possible values of the response. This is done by jointly estimating a residual density for a prediction function at the same time as the prediction function is chosen to maximize the log likelihood of the estimated density.

We demonstrate how this may be done directly in a univariate spline context and make use of the gradient descent view of boosting Hastie et al. (2001) to develop a procedure in which the loss function is updated adaptively as boosting continues.

We demonstrate that our method has a robustification effect, and show its usefulness in diagnosing problems in data. We illustrate our methods with practical examples when likelihood is an appropriate evaluation criterion and we suggest further variations that may provide useful insights into practical problems.

1 Introduction

The task of regression in machine learning is usually presented as the problem of finding a function f of the predictor variables x that minimizes an expected loss between predictions using f and observed values y:

$$EL(f(x), y).$$

We examine a somewhat different problem in which we seek a *distribution* $p(y|x)$ at each value of x that maximizes the density of the outcome y conditional on the predictors x:

$$Ep(y|x).$$

In its full generality problem appears to require insurmountable amounts of data and we restrict ourselves to a class of distributions which vary in *location* with x only: $p(y|x) = p(y - f(x))$. We now want non-parametric estimates of both p and f which we propose to estimate jointly. At the end of the paper, we show how this may be generalized to more flexible models.

We believe that this scheme will have the effect of robustifying the estimate of f. It has long been known that when the object of machine learning is to predict a real-valued quantity, the use of squared-error loss can lead to poor results. Outliers and data which are over-dispersed or censored or otherwise non-gaussian can all cause a learned function to have poor generalization error. In order to deal with this problem various robust loss functions have

been proposed. Friedman (2001) examined both absolute value loss and a Huberized loss function in the context of gradient descent boosting and found that Huberization can significantly improve generalization error in regression.

The choice of loss function makes an implicit assumption about the distribution of residuals. A loss function may be considered to be proportional to the negative log of the assumed residual density. In this context, squared error loss corresponds to a Gaussian density, absolute value a double exponential and Huberization is close (although not identical) to a mixture of these. Huberization, in this context, suggests that there is a higher likelihood of having observations far away from their mean than under a normal model. By estimating a density for the residuals, our methods similarly allow such an affect, but neither demands that the distribution be symmetric nor dictates how heavy the dails of the distribution are.

This paper proposes the use of a negative log-likelihood loss with a density that is estimated directly from the residuals. In this scenario, the prediction function is estimated to optimize log-likelihood of a density which is itself estimated from its residuals. The estimates are made identifiable by smoothness constraints on both the density and the prediction function. In particular, we place a penalty on the third derivative of the log-density. Since this penalty is zero for quadratic functions, we may think of it as penalizing departures from a Gaussian distribution. We view this approach as using squared error loss except where there is good evidence that doing so is not appropriate.

In this paper, §2 outlines the proposed density estimation procedure and its penalty. §3 develops a joint estimation for both the density and the prediction function with particular reference to univariate splines. We use this procedure to demonstrate the qualitative effect of this estimate on Canadian rainfall data in §4. §5 then develops an adaptive negative log-likelihood loss for use in boosting. We demonstrate the use of this loss in terms of diagnostics and robustness for real and simulated observations for baseball salaries from 1991 in §6 and §7 analyzes binned outcomes in the form of the California Income Data. §8 explores a bivariate estimate in which the density is indexed by some other known quantity.

2 Penalized Density Estimation

The central technique in this paper is the estimation of a log density for the residuals of a prediction function. We propose to do this directly through the machinery of functional data analysis Ramsay and Silverman (2005). This estimate was first proposed in Silverman (1982, 1986); Gu and Qiu (1993). The density function $p(r)$ is chosen to minimize the negative log likelihood of the residuals plus a trade-off with a smoothness penalty:

$$p(r) = \underset{p}{\operatorname{argmin}} \sum_{i=1}^{n} -\log(p(r_i)) + \lambda P(p).$$

Here λ controls the trade-off between maximizing the likelihood of the density estimate and the smoothness implied by the penalty P. The penalty we

propose is the integral of the squared third derivative of the log of p:

$$P(p) = \int \left[\frac{d^3}{dr^3} \log(p(r)) \right]^2 dr. \tag{1}$$

If $\log(p(r))$ is quadratic in r – implying that $p(r)$ is Gaussian – this penalty is zero. As $p(r)$ deviates from Gaussianity, the penalty increases. λ therefore dictates the extent to which non-Gaussian residuals influence the fit to the data.

Instead of estimating $p(r)$, we will estimate $l(r) = \log(p(r))$ directly. To do so, we will approximate l by a linear basis expansion:

$$l(r|b) = \sum_{i=1}^{k} b_i \phi_i(r). \tag{2}$$

Here we have chosen the ϕ_i to be B-spline functions. B-splines provide a piecewise polynomial basis that is particularly easy to work with and for which de Boor (2001) provides a useful overview. In choosing this basis, we also ensure that the combined support of the elements ϕ_i cover the range of r. To recover $p(r)$ from $l(r)$ we need to exponentiate and re-normalize:

$$p(r|b) = \frac{e^{l(r|b)}}{\int e^{l(r|b)} dr}.$$

The normalization constant therefore also needs to be included in an optimization criteria for the coefficients in the basis expansion which becomes

$$b = \underset{b}{\operatorname{argmin}} \left\{ -\sum_{i=1}^{N} l(r_i|b) + N \log \int e^{l(r|b)} dr + \lambda \int \left(\frac{d^3}{dr^3} l(r|b) \right)^2 dr \right\}.$$

This problem is convex in b and can be solved efficiently by a conjugate gradient method. An algorithm for this density estimate is already available in the `fda` package for Matlab and R and is described in Ramsay and Silverman (2005).

3 Joint Density and Function Estimation

Once we have a representation of the log likelihood l of the residuals, we can estimate a function f to minimize the negative log-likelihood. This may also incorporate a smoothness penalty $P(f)$ on the prediction function:

$$f(x) = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{N} -l(y_i - f(x_i)) + P(f).$$

Estimating l presupposes that we posses the residuals and therefore f, while estimating f presupposes that we possess l. However, we can combine them into a single optimization to find $\{f, b\}$ to jointly minimize

$$-\sum_{i=1}^{N} l(y_i - f(x_i)|b) + N \log \int e^{l(r|b)} dr + \lambda \int \left(\frac{d^3}{dr^2} l(r) \right)^2 dr + P(f).$$

In particular, in a one-dimensional domain, we can also express f in terms of a basis:

$$f(x|c) = \sum_{j=1}^{m} c_j \psi_j(x) \tag{3}$$

and reduce the problem to estimating two sets of coefficients, $\{c, b\}$ to minimize

$$-\sum_{i=1}^{N} l(y_i - f(x_i|c)|b) + N \log \int e^{l(r|b)} dr + \lambda \int \left(\frac{d^3}{dr^3} l(r|b) \right)^2 dr + P(c). \tag{4}$$

In this context P is typically the integral of squared second derivatives of $f(x|c)$, although more sophisticated choices of penalty are available. In the boosting algorithms below, the penalty will be dropped in favor of direct co-efficient shrinkage.

These equations are now well defined up to an additive constant. Let us fix some $f(x)$ and $l(y)$, then for any number t, (4) takes the same value for the function pair $\{l_t, f_t\} = \{l(y - t), f(x) + t\}$ as it does for $\{l, f\}$. It makes sense to want f to be approximately unbiased and we therefore add an extra constraint that the residuals should sum to zero. Practically, this takes the form of an extra term in (4):

$$\left(\sum_{i=1}^{N} (y_i - f(x_i)) \right)^2.$$

Which penalizes the discrepancy of the *mean error* from zero. Nominally, this can be set to zero without changing the values of any of the terms in (4). In practise, the use of a basis to represent l in (2) may mean that this is only approximately true. We therefore have a final minimization criterion:

$$\begin{aligned} -\sum_{i=1}^{N} l(y_i - f(x_i|c)|b) + N \log \int e^{l(r|b)} dr + \\ \lambda \int \left(\tfrac{d^3}{dr^3} l(r|b) \right)^2 dr + P(c) + \left(\sum_{i=1}^{N} (y_i - f(x_i)) \right)^2 . \end{aligned} \tag{5}$$

As in the density estimate in §2, this problem may be solved with a conjugate gradient method[1]. We note that there may be multiple local minima in the solution space. For example, when the log density l has two local maxima, f may track either one. However, we have never come across an occasion in which this was problematic.

4 Example: Canadian Precipitation

In order to provide an intuitive sense of the effect of this methodology, we will demonstrate it on a univariate example in the form of the daily precipitation in Vancouver averaged over the years 1960 to 1994. Log precipitation has a much higher signal to noise ratio than using precipitation directly and we therefore wish to model this. However, there are two days in which the average recorded precipitation is zero. It is unlikely that this is strictly cor-

[1]The Matlab function `fminunc` in the Optimization package has been used throughout.

rect: rainfall gauges measure zero for any rainfall under 0.5mm. We have therefore added 0.01 to the average precipitation for each day before taking logs.

The two unusual days now appear as outliers which may strongly affect a smooth of the data. Figure 1 plots the data along with a smooth. In this case we have used a harmonic acceleration penalty in (5):

$$P(f) = \int \left(\frac{d^3}{dt^3} f(t) - \frac{2\pi}{365} \frac{d}{dt} f(t) \right)^2 dt$$

which encourages the smooth toward a sinusoidal function with period 365. A very strong dip is evident around the outliers. A smoothing parameter $\lambda = 10^3$ was chosen by eye.

A joint estimation of the density of the residuals along with the penalized spline smooth provides an exception for the outliers in the form of a "bump" in the residual density estimate and a noticeably less distorted summer dip. This is especially apparent in the plot of log density in Figure 2. It also appears that there is a certain amount of skewness in the residuals that is not necessarily captured by squared error loss.

The lessons from this example are important for machine learning. When there are relatively few degrees of freedom in the model, moderately non-gaussian error distributions do not distort our estimates very much. However, when more flexible learners are used, unusually large errors can pull the prediction function away from its optimal value. When using squared

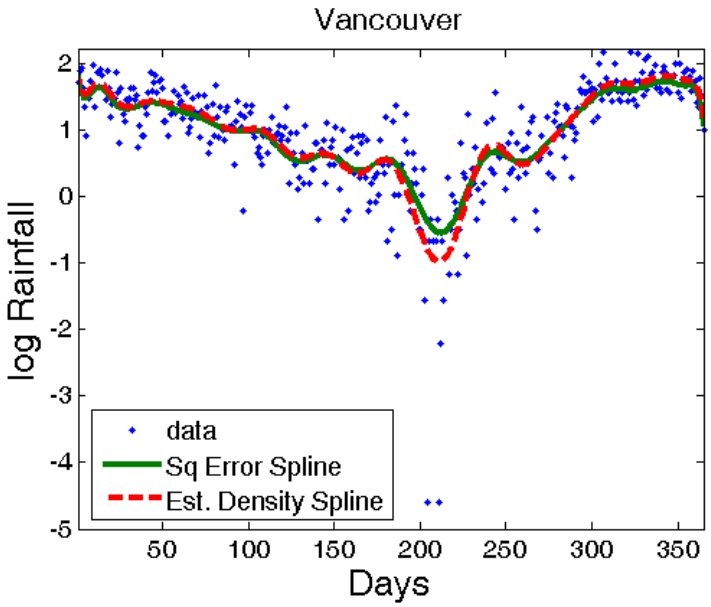

Figure 1: A comparison of fits for the log average precipitation in Vancouver. The dashed line gives the fit estimated with squared-error loss, the solid the fit estimated jointly with the density.

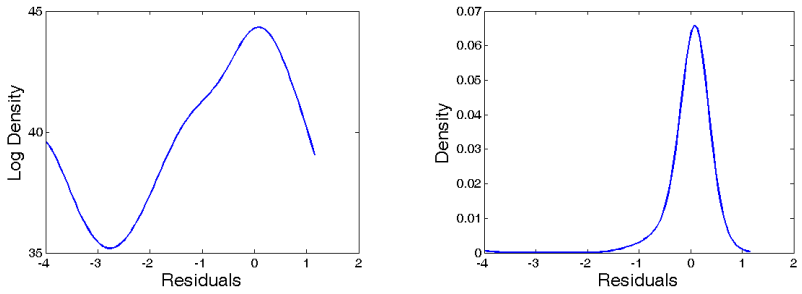

Figure 2: Estimates of log density (left) and density (right) residuals of the Vancouver precipitation data when estimated jointly with the regression function.

error loss, greater smoothing is necessary to counter this tendency, potentially loosing real features in doing so. Fitting according to an appropriate likelihood reduces the consequences of this trade-off.

5 Gradient Boosting for an Adaptive Likelihood

Joint optimization of the form of (5) is only possible where f is expressed as an expansion in terms of known basis functions. This is typically not the case in machine learning. Moreover, many regression techniques assume squared error loss. However, both of these problems may be overcome by following the gradient boosting methodology outlined in Hastie et al. (2001).

5.1 Standard Gradient Boosting

Gradient boosting builds prediction functions that are linear combinations of learners:

$$f(x) = \sum_{j=1}^{K} c_j T_j(x)$$

where T_j is a weak learner; we will employ eight-node binary trees. We attempt to minimize

$$\sum_{i=1}^{n} L(y_i, f(x_i))$$

where L is a loss function indicating the loss of predicting $f(x_i)$ when the truth is y. Boosting proceeds by successively training new prediction functions to predict the gradient of the $L(y_i, f(x_i))$ with respect to $f(x_i)$. Beginning with a prediction function $f_0 = 0$, the jth learner is trained to predict the gradient of the loss for the current function. In this context T_{j+1} is a eight-node tree trained to predict

$$\{-g_{im} = L'(y_i, f_j(x_i))\}_{i=1}^{N}$$

from the x_i where f_j is the current prediction function. The prediction function is then updated

$$f_{j+1}(x) = f_j(x) + c_{j+1}T_{j+1}(x)$$

with c_j chosen to minimize

$$\sum_{i=1}^{N} L(y_i, f_j(x_i) - c_j T_{j+1}(x)).$$

Part of the appeal of this approach to boosting is that it allows us to learn an f to optimize any differentiable loss function using weak learners that are built to minimize squared error loss. In particular, if we knew a log density l for the residuals, we would be able to minimize the negative log likelihood

$$\sum_{i=1}^{N} -l(y_i - f(x_i)).$$

What we will show in the next section is that l may also be estimated within this scheme.

5.2 Learned-Loss Gradient Boosting

The gradient boosting regime outlined above assumes a known loss function L. Our proposal is to learn L using the same joint estimation scheme described in §3. In this scheme both f and L are updated each step. We note that

$$f_j = \sum_{i=1}^{j} c_i T_i$$

is exactly a basis expansion of the form of (3), where the basis is the set of trees learned up until iteration j. We can therefore use the scheme in §3 to re-estimate the c_j along with L. This gives us the following algorithm:

1. Set $f_0 = 0$ and $l_0(r) = r^2$
2. For $j = 1$ to M
   (a) Fit a eight-node tree T_j to predict $\{l_j'(y_i - f_{j-1}(x_i))\}_{i=1}^{N}$ from the training data.
   (b) Estimate $\{c, b\}$ to minimize

$$- \sum_{i=1}^{N} l \left(y_i - \sum_{s=1}^{j} c_s T_s(x_i) | b \right) + \ldots$$

$$N \log \int e^{l(r|b)} dr + \lambda_d \int l''(r|b)^2 dr + \left(\sum_{i=1}^{N} (y_i - f(x_i)) \right)^2$$

with

$$l(r|b) = \sum_{s=1}^{k} b_s \phi_s(r)$$

(c) Set $f_j = \sum_{s=1}^{j} c_s T_s$ and $l_j = \sum_{s=1}^{k} b_s \phi_s$

Since the Newton optimization of $\{c, b\}$ is the main computational bottleneck in the algorithm, we restrict our optimization in Step 2a. Instead, at each iteration, we fix l and build k trees, estimating the co-efficient for each by squared error. We then update l as in Step 2b, re-estimating the coefficients only for the most recent k trees:

1. Set $f_0 = 0$ and $l_0(r) = r^2$

2. For $j = 1$ to M

   (a) For $t = 1$ to k

      i Fit a eight-node tree T_{j+t} to predict $\{l_j'(y_i - f_{j+t-1}(x_i))\}_{i=1}^{N}$ from the training data.

ii Estimate a coefficient for T_{j+t} by squared-error loss for the residuals:

$$\hat{c}_{j+t} = \frac{\sum_{i=1}^{N} T_{j+t}(x_i)(y_i - f_{j+t-1}(x_i))}{\sum_{i=1}^{N} T_{j+t}(x_i)},$$

iii Set $f_{j+t} = f_{j+t-1} + \hat{c}_s T_s$

(b) Estimate $\{c_{j+1}, \ldots, c_{j+k}, b\}$ to minimize

$$-\sum_{i=1}^{N} l\left(y_i - f_{j-1}(x_i) - \sum_{t=1}^{k} c_{j+t} T_{j+t}(x_i)\,|b\right) + \ldots$$
$$N \log \int e^{l(r|b)} dr + \lambda_d \int l''(r|b)^2 dr + \left(\sum_{i=1}^{N} (y_i - f(x_i))\right)^2$$

with

$$l(r|b) = \sum_{s=1}^{k} b_s \phi_s(r)$$

(c) $j = j + k$

(d) Set $f_j = \sum_{s=1}^{j} c_s T_s$ and $l_j = \sum_{s=1}^{k} b_s \phi_s$

The use of squared loss in Step (a)ii is a used as an approximation since l_j is expected to be close to quadratic and the coefficients will be re-estimated the next time we estimate a density.

5.3 Regularization

So far, we have not penalized the linear combination of base learners. Friedman (2001) found that decreasing the learning rate by multiplying each c_j by some $\nu < 1$ improved performance significantly in reducing the variability of each f_j. This can also be incorporated into our learning process and we will modify Step 2d to

$$f_j = \nu \sum_{s=1}^{j} c_s T_s.$$

When k is large in Step 2b, this shrinkage will tend to slow the learning process and we have found found that $k = 2$ or $k = 3$ provides the fastest implementation when ν takes its traditional value of 0.1.

We have not used the term $P(c)$ from (5) here. It might be natural to include it in the form of a ridge penalty in Step 2b. However, our experiments with such penalties proved unsatisfactory. This is because it is important to have realistic residuals in order to estimate the log density l. Incorporating a large ridge penalty distorts these residuals *before* the density has been estimated, resulting in a density estimate that is broader and flatter than it should be.

6 Example: Baseball Salaries

Watnik (1998) provides data on the salaries of 327 non-pitcher profes-

sional baseball players along with 12 predictor variables containing statistics of the player's performance in the previous year and a further 4 relating to the status of their contract.

6.1 Simulated Responses

We begin by investigating the robustification effect of the methodology. In order to do this, it will be useful to consider simulated data where the true mean response is known. However, for verisimilitude, we have used the predictors from the baseball salary data as predictors for a simulated response. Hooker (2004) observes that machine learning data sets often have predictors whose joint distribution is highly dependent and does not resemble readily simulated data.

We have chosen a linear combination of the predictors as our response function. We added errors distributed according to the slash distribution – a standard normal random variable divided by an independent uniform random variable. The slash distribution was scaled to give an signal to noise ratio of approximately one. This distribution tends to produce very heavy tails, leading to a small number of large outliers.

We employed our algorithm and standard least-squares boosting on this data, leaving out 77 points as a test set and a further 37 points as a validation set used to choose the number of boosting iterations. The performance of the resulting prediction was then evaluated with respect to squared on the test set *without adding errors* so that we measure the distance between the given

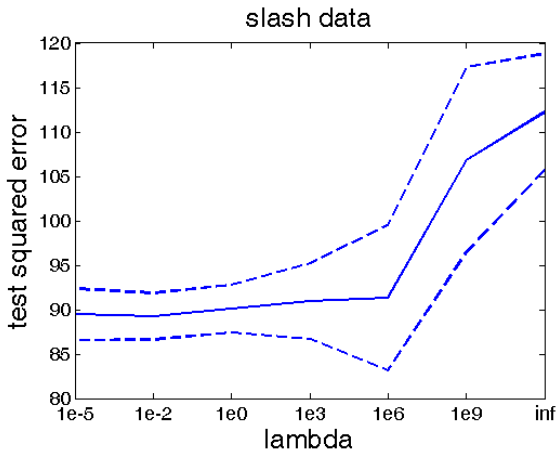

Figure 3: Squared distance from the best prediction for simulated data. Solid line provides the average of 50 simulations, dashed lines represent 95% confidence intervals. 'inf' in the x-axis indicates standard least-squares boosting. Confidence intervals have been calculated taking an additive effect for each training set into account.

solution and a best one. This experiment was repeated 50 times, randomly drawing the test and validation set each time, with 6 different values of λ and with standard least-squares boosting. Figure 3 provides a summary of the responses. For small values of λ, our method produces an approximate 20% improvement in average performance.

6.2 Observed Responses

We now turn to the true responses in the data set. As with simulated responses sets of 77 test and 37 validation points were chosen at random and our method trained with 5 values of λ and using least-squares boosting. In this case, we are interested in measuring the *predicted probability* of the

observed outcome on the test set, given by the negative log-likelihood:

$$-\sum_{i=1}^{N} \log(p(r_i))$$

which was calculated from the estimated density (2) for the proposed methods. We calculated the log-likelihood using a Gaussian density for least-squares boosting, estimating the variance as the mean square of the residuals. In both cases, the number of boosting iterations was chosen based on the likelihood evaluated on a validation set.

Figure 4 summarizes our results in which we find that a medium value of smoothing provides best performance. We have also included an analysis of squared error performance in which a similar improvement is obtained, although the improvement is more modest than for simulated data. Investigating further, Figure 5 compares histograms of training set residuals for the smallest λ with those for least-squares boosting. Here we can see that the estimation of a density has allowed the residuals to take a considerably more skewed distribution while least-squares boosting tries to make them Gaussian. We confirm this with the experiment of performing least-squares boosting and then estimating a density as described in Section 2b. We then compared the test set likelihood of this density with that of the density evaluated at the smallest λ. These results, given in Figure 5, indicate that our method changes the prediction function, providing better estimates of the distribution of residuals than estimating this distribution as a post-processing

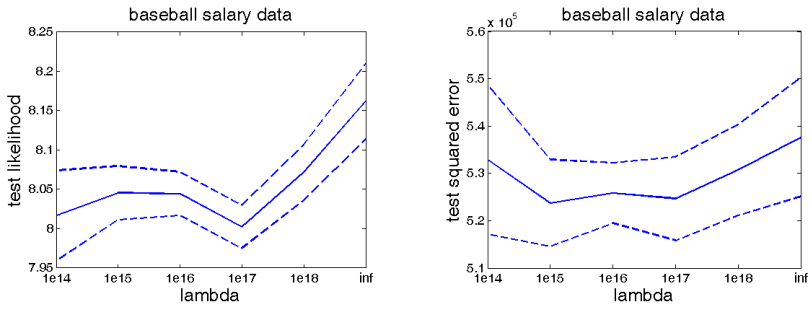

Figure 4: Performance on the baseball salary data with respect to negative log likelihood (left) and squared error (right) over five values of λ and least-squares boosting, designated by 'inf'. Solid lines represent the mean of 20 simulations, dashed a 95% confidence bound. Confidence intervals were calculated taking an additive effect for each training set into account.

step following least-squares boosting.

7 Example: California Income Data

After removing records with missing values, the California Income Data Impact Resources, Inc. (1987) contains 6876 complete records of annual household income along with 13 demographic variables taken from responses to a survey in a San Francisco Bay area shopping mall. In this case income was divided into nine categories which were equally spaced aside from the first and last two. These categories were recorded as numbers from 1 to 9.

In this setting, the discretization of the response has had a number of effects. It has removed outliers and removed a large amount of the skewness commonly observed in income data. It has also changed the appropriate loss for measuring performance. A reasonable loss might be the number of

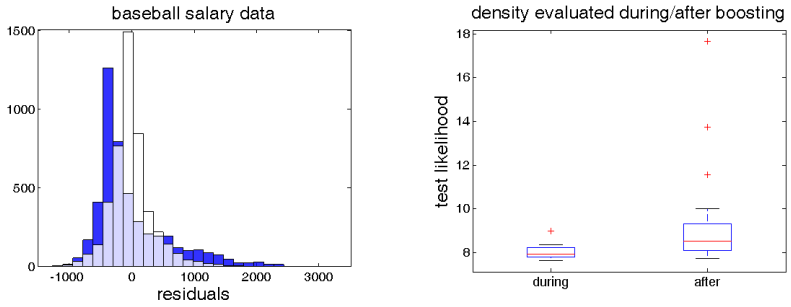

Figure 5: Left: histograms for training set residuals for $\lambda = 10^{14}$ (blue) and least-squares boosting (white), indicating the effect of least-squares boosting in forcing Gaussianity on the residuals. Right: box-plots for test set likelihood of densities estimated with $\lambda = 10^{14}$ during boosting (left) and after least-squares boosting (right).

categories away from the truth - approximately absolute error loss. More appropriate is the predicted probability of the observation. Figure 6 provides statistics on these two measures. Both absolute value and likelihood performance improves with less smoothing. A plot of the estimated density of the residuals shows considerable skewness, indicating that a relatively rough estimate of density is indeed appropriate.

8 Varying Residual Distributions

We can now ask further questions about the distribution of the residuals. It is possible that this distribution varies systematically with some of our measured quantities. First year statistics courses teach that residuals should be analyzed by plotting them against predictors and predicted values. If there

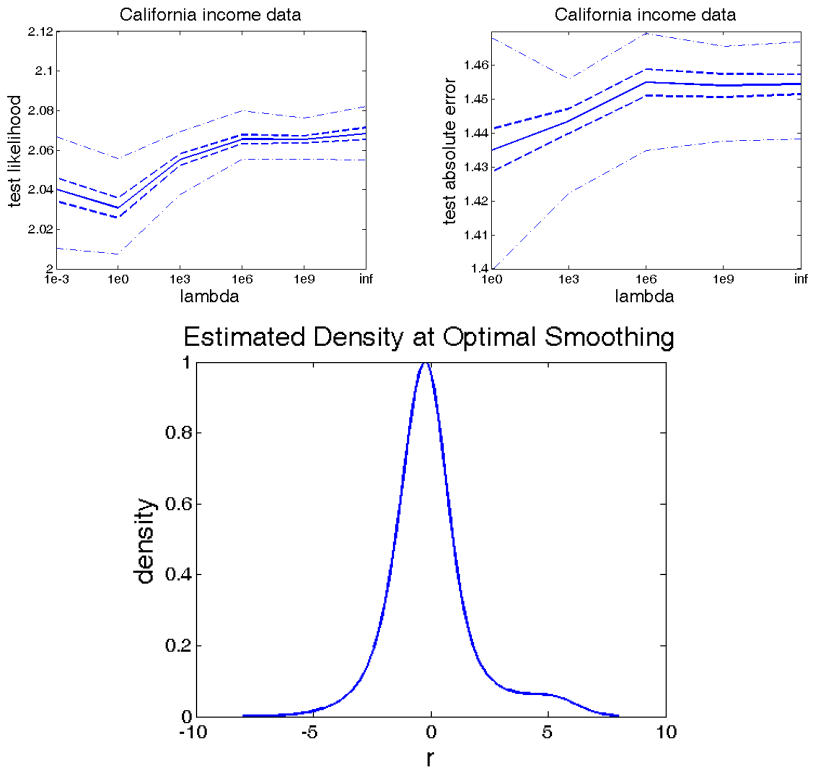

Figure 6: Performance statistics for the California Income Data over 5 values of λ for likelihood (left) and absolute error (right) criteria. Solid lines give the mean of 25 simulations, dashed 95% confidence intervals. Dotted lines provide minimum and maximum values. Confidence intervals were calculated taking account of an additive effect for each training set. The bottom panel provides an estimated density of the residuals at $\lambda = 1$.

is indeed a systematic trend to be found in such plots, this should be taken into account in the loss functions employed. Truncated data, for example, will have error distributions that become increasingly skewed close to the truncation boundary. If this is the case, using the same error distribution across all points would lead to predictions that are further away from the censoring boundary than is optimal.

The density estimation machinery already discussed can be employed to estimate a density that varies over other quantities. Suppose we parameterize the log density by another measured variable t. Typically, t may be the predicted values or the observed values of the response. It may also be one of the predictors if prior knowledge suggests that the error distribution varies with it.

We now have a bivariate function estimation problem and will again fit the parameterized log density, $l(r,t)$, by maximizing the log likelihood. As in univariate density estimation, we will place a penalty on $\frac{d^3}{dr^2}l(r,t)$ to encourage l to be close to squared error loss for each value of t. Since we do not wish to change l across t without good reason, it is appropriate to also penalize $\frac{d}{dt}l(r,t)$. This results in an optimization criterion

$$\sum_{i=1}^{N}\left(-l(r_i,t_i)+\log\int e^{l(r,t_i)}dr\right)+ \\ \lambda_r\int\left(\frac{d^3}{dr^3}l(r,t)\right)^2drdt+\lambda_t\int\left(\frac{d}{dt}l(r,t)\right)^2drdt.$$

Representing $l(r,t)$ by a bivariate basis expansion:

$$l(t,r) = \sum_{i=1}^{k} b_i \phi_i(r,t)$$

we can again employ a conjugate gradient method to solve for the b_i. In our experiments, we have used a tensor-product of the B-spline basis employed for univariate density estimation.

Joint estimation with linear models along the lines of §3 is similarly feasible. If we choose a prediction function of the form (3) then the joint optimization problem becomes

$$\sum_{i=1}^{N} \left(-l(y_i - f(x_i|c), t_i) + \log \int e^{l(r,t_i)} dr \right) + \lambda_r \int \left(\frac{d^3}{dr^3} l(r,t) \right)^2 dr dt + \lambda_t \int \left(\frac{d}{dt} l(r,t) \right)^2 dr dt + P(c).$$

or, if $t_i = f(x_i|c)$,

$$\sum_{i=1}^{N} \left(-l(y_i - f(x_i|c), f(x_i|c)) + \log \int e^{l(r,f(x_i|c))} dr \right) + \lambda_r \int \left(\frac{d^3}{dr^3} l(r,t) \right)^2 dr dt + \lambda_t \int \left(\frac{d}{dt} l(r,t) \right)^2 dr dt + P(c).$$

both of which can also be solved numerically.

We have carried out this scheme with the Vancouver precipitation data,

indexing the density by the observed values. This makes sense when we believe that particular observed values may be systematically biassed, as is the case for zero recorded rainfall. Figure 7 presents a contour plot of density log density. A bump can be seen at the lower corner; effectively creating an exception for zero values.

Here, the fact that outliers are associated with very small values of log precipitation becomes apparent – the density becomes bimodal for small values of t, but for most other values it is close to Gaussian. Using this estimation scheme, truncation boundaries would show up as a line running diagonally across the distribution. Similar diagnostic features to those found in diagnostic scatter plots for curvature or heteroscedasticity can be expected to be apparent in this estimate. Using this form of a density estimate in training a prediction function allows us to control for the adverse effects of these features.

Further indexing dimensions could, of course, be added. However, the cost of the density estimate grows exponentially with dimension. Currently, the conjugate-gradient solver for the bivariate estimate is not efficient enough to be included in an adaptive boosting procedure.

9 Conclusions

This paper demonstrates that it is possible both to learn a regression function for a data set and to learn the density of its residuals at the same

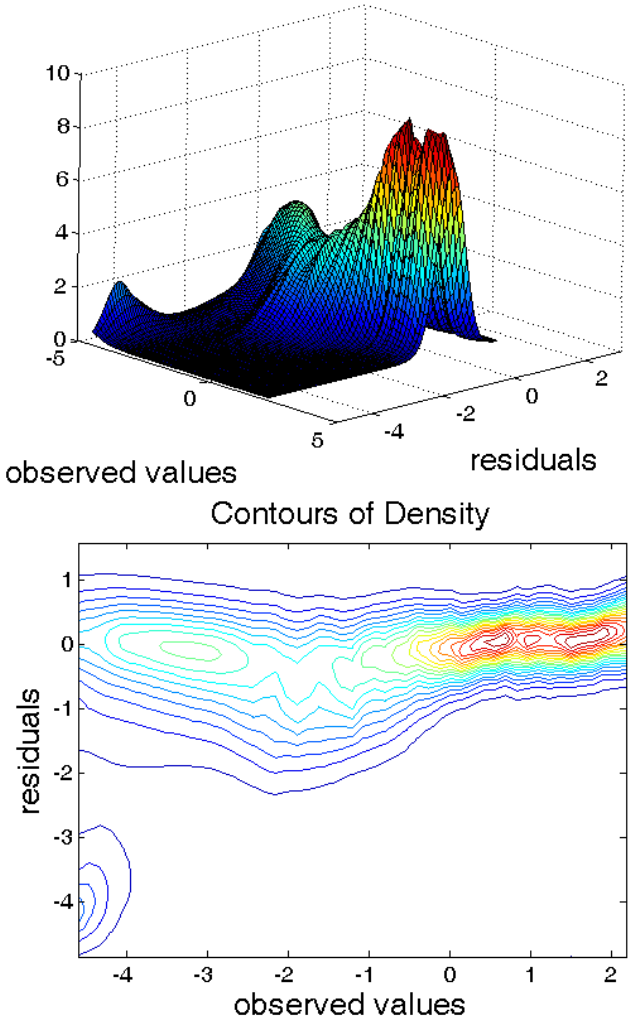

Figure 7: Contour plot of the density of residuals for Vancouver precipitation data. The density has been indexed by observed value.

time. More-over, a good estimate of density requires that this be done jointly. We make use of the gradient descent view of boosting to employ methods from functional data analysis that make this feasible. Our methodology has a natural robusitification effect, allowing the likelihood to automatically discount the influence of outliers. In many settings, likelihood – representing the predicted probability of an observed outcome – is a more natural measure of performance than squared error loss.

There are several open problems for our methods. Foremost among these is computational complexity – the requirement to perform a conjugate-gradient optimization at every k boosting rounds severely slows down the algorithm and a single-step approximation may provide speedup without sacrificing performance. We have also not addressed the choice of λ, which can vary over many orders of magnitude depending on the problem. The use of cross-validation will compound the problems above. However, some experimentation using only a few boosting iterations should allow the user to choose a small set of values to consider.

References

de Boor, C. 2001. *A Practical Guide to Splines*. Springer, New York.

Friedman, J. H. 2001. Greedy function approximation: A gradient boosting machine. *Annals of Statistics 29,* 5, 1189–1232.

Gu, C. and Qiu, C. 1993. Smoothing spline density estimation: theory. *Annals of Statistics 21*, 271–234.

Hastie, T., Tibshirani, R., and Friedman, J. 2001. *The Elements of Machine Learning: Data Mining, Inference and Prediction.* Springer, New York.

Hooker, G. 2004. Diagnosing extrapolation: Tree-based density estimation. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*

Impact Resources, Inc. 1987.

Ramsay, J. O. and Silverman, B. W. 2005. *Functional Data Analysis.* Springer, New York.

Silverman, B. W. 1982. On the estimation of a probability density function by the maximum penalised likelihood method. *Annals of Statistics 10*, 795.

Silverman, B. W. 1986. *Density Estimation for Statistics and Data Analysis.* Chapman and Hall, New York.

Watnik, M. R. 1998. Pay for play: Are baseball salaries based on performance? *Journal of Statistics Education 6,* 2.