Кузиной Екатерины

Обзор на статью NGBoost: Natural Gradient Boosting for Probabilistic Prediction

https://arxiv.org/pdf/1910.03225v1.pdf

Методы машинного обучения применяются для решения самых разнообразных задач. Например, для прогнозирования погоды на следующий день или наличия заболевания в будущем на основе медицинских записей пациента. В таких задачах важно уметь оценивать прогнозную оценку неопределенности. Для решения данной задачи был предложен Natural Gradient Boosting (NGBoost) - алгоритм предсказания на основе бустинга, который является гибким, простым в использовании и быстрым по сравнению с существующими методами.

Рассмотрим прогноз с плотностью вероятности $P_\theta(y|x)$, где x - вектор признаков, y - целевое значение, а θ - параметры распределения. Обозначим функцию распределения как F_θ.

Для оценивания восстановленного распределения P вводится оценка $S(P, y)$ такая, чтобы выполнялось условие:

$$E_{y\sim Q}[S(Q,y)] \le E_{y\sim Q}[S(P,y)] \quad \forall P, Q, \ (1)$$

где Q - истинное распределение y, P - любое другое распределение. Авторы статьи берут P и Q из одного распределения и находят оптимальный параметр θ.

Наиболее широко для оценивания неизвестного параметра используется оценка максимального правдоподобия (maximum likelihood estimation, MLE), рассмотрим оценку $-\log$ likelihood:

$$L(\theta, y) = -\log P_\theta(y).$$

Также можно рассматривать альтернативу MLE - оценку CRPS:

$$C(\theta, y) = \int_{-\infty}^{y} F_\theta(z)^2 dz + \int_{y}^{\infty} (1 - F_\theta(z))^2 dz$$

Для подсчета расхождения между распределениями вводится дивергенциия исходя из (1):

$$D_S(Q||P) = E_{y\sim Q}[S(P,y)] - E_{y\sim Q}[S(Q,y)],$$

которое принимает не отрицательные значения. Используя выше введенные оценки для распределения, получаем дивергенцию Кульбака-Лейблера и L^2 дивергенцию:

$$D_L(Q||P) = E_{y\sim Q}[L(P,y)] - E_{y\sim Q}[L(Q,y)] = E_{y\sim Q}\left[\log\frac{Q(y)}{P(y)}\right] =: D_{KL}(Q||P),$$

$$D_C(Q||P) = E_{y\sim Q}[C(P,y)] - E_{y\sim Q}[C(Q,y)] = \int_{-\infty}^{\infty}(F_Q(z) - F_P(z))^2 dz =: D_{L^2}(Q||P)$$

Для максимизации оценки рассматривается обобщенный естественный градиент (the generalized natural gradient) - направление наибольшего возрастания в римановом пространстве, которое определяется:

$$\hat{\nabla} S(\theta, y) \propto \lim_{\epsilon\to 0} \arg \max_{d:D_s(P_\theta||P_{theta+d})=\epsilon} S(\theta + d, y).$$

Решив задачу оптимизации, получим, что естественный градиент имеет вид:

$$\hat{\nabla} S(\theta, y) \propto I_S(\theta)^{-1} \nabla S(\theta, y),$$

где $I_S(\theta)$ - римановская метрика статистического многообразия при θ, определяемая оценкой S, $\nabla S(\theta, y)$ - обычный градиент оценки S.

Используя $S = L$, получаем информацию Фишера для распределения P_θ:

$$I_L(\theta) = E_{y\sim P_\theta}[-\nabla_\theta^2 L(\theta, y)].$$

Аналогично при S=C получим римановскую метрику статистического многообразия $I_C(\theta)$, использующую D_{L^2} в качестве меры расстояния:

$$I_C(\theta) = 2\int_{-\infty}^{\infty} \nabla_\theta F_\theta(z) \nabla_\theta F_\theta(z)^T dz.$$

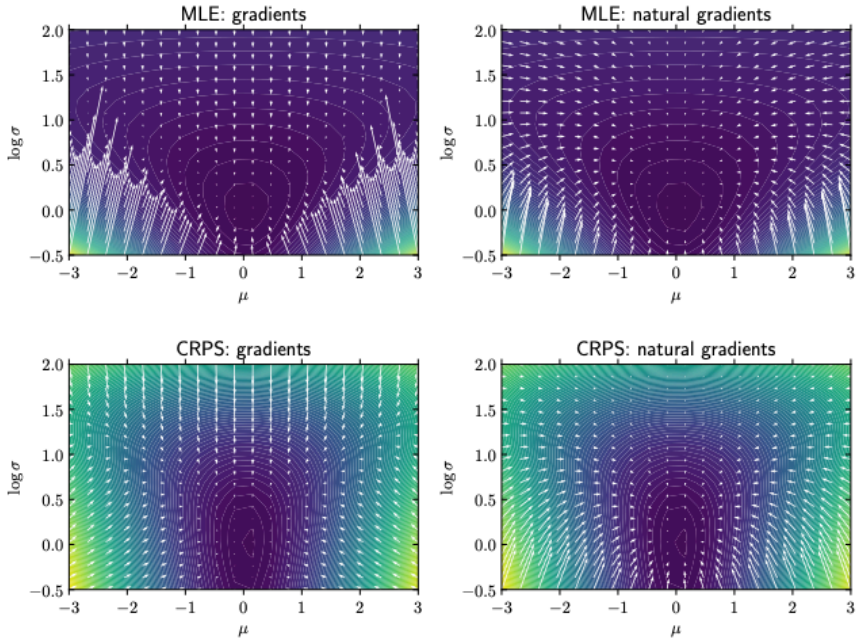

Figure 3: Proper scoring rule loss functions and corresponding gradients for fitting a Normal distribution on samples $\sim N(0, 1)$. For each scoring rule, the landscape of the loss (colors and contours) is identical, but the gradient fields (arrows) are markedly different depending on which kind of gradient is used.

Задача оптимизации с использованием естественного градиента имеет гораздо более эффективную и стабильную динамику обучения, чем при использовании только обычных градиентов. На Figure 3 показано векторное поле градиентов и естественных градиентов для MLE и CRPS в параметрическом пространстве нормального распределения со средним μ и логарифмом стандартного отклонения $\log\sigma$.

NGBoost - это алгоритм обучения с учителем для предсказания, который использует градиентный бустинг для нахождения параметров условного распределения $y|x$, где y может принимать одно из значений (±1,R,1,…, K, R_+, N и др) и $x \in R^d$.

Алгоритм (Figure 2) состоит из 3 модулей, которые выбираются заранее:

- базового алгоритма (решающее дерево)
- параметрического распределения (нормальное, Бернулли и др.)
- оценки распределения (MLE, CRPS и др.)

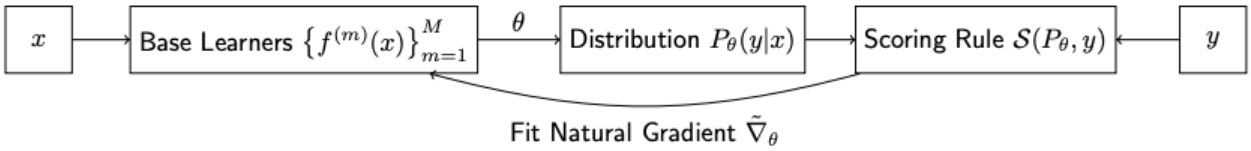

Figure 2: NGBoost is modular with respect to choice of base learner, distribution, and scoring rule.

Предсказание $y|x$ для нового объекта x порождается из условного распределения P_θ. Параметр θ находим с помощью градиентного бустинга с M базовыми алгоритмами (base estimator). Например, для нормального распределения с параметром $\theta = (\mu, \log\sigma)$ каждый базовый алгоритм должен уметь находить оба параметра, т.е. $f^{(m)} = \left(f_\mu^{(m)}, f_{\log\sigma}^{(m)}\right)$.

На каждой итерации алгоритма m вычисляются естественные градиенты каждого прецедента, на которых затем обучается базовый алгоритм. Результаты базового алгоритма масштабируются с учетом коэффициента масштабирования $\rho^{(m)}$ и скорости обучения η:

$$\theta^{(m)} = \theta^{(m-1)} - \eta(\rho^{(m)} f^{(m)}(x)).$$

Коэффициент масштабирования выбирается таким образом, чтобы минимизировать потерю по направлению проекции градиента вдоль линии поиска.

Algorithm 1 NGBoost for probabilistic prediction

Data: Dataset $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$.
Input: Boosting iterations M, Learning rate η, Probability distribution with parameter θ, Proper scoring rule $\mathcal{S}$, Base learner f.
Output: Scalings and base learners $\{\rho^{(m)}, f^{(m)}\}_{m=1}^M$.

$\theta^{(0)} \leftarrow \arg\min_\theta \sum_{i=1}^n \mathcal{S}(\theta, y_i)$ ▷ initialize to marginal
for $m \leftarrow 1, \ldots, M$ **do**
  for $i \leftarrow 1, \ldots, n$ **do**
    $g_i^{(m)} \leftarrow \mathcal{I}_\mathcal{S}\left(\theta_i^{(m-1)}\right)^{-1} \nabla_\theta \mathcal{S}\left(\theta_i^{(m-1)}, y_i\right)$
  end
  $f^{(m)} \leftarrow \text{fit}\left(\left\{x_i, g_i^{(m)}\right\}_{i=1}^n\right)$
  $\rho^{(m)} \leftarrow \arg\min_\rho \sum_{i=1}^n \mathcal{S}\left(\theta_i^{(m-1)} + \rho \cdot f^{(m)}(x_i), y_i\right)$
  for $i \leftarrow 1, \ldots, n$ **do**
    $\theta_i^{(m)} \leftarrow \theta_i^{(m-1)} - \eta\left(\rho^{(m)} \cdot f^{(m)}(x_i)\right)$
  end
end

Для экспериментального исследования были взяты наборы данных из UCI Machine Learning Repository. Качество предсказаний определяется среднеквадратичным отклонением (RMSE), а качество прогнозной оценки неопределенности - отрицательным логарифмом правдоподобия (NLL). Полученные в ходе экспериментов результаты (Table 1) сравниваются с MC dropout и Deep Ensembles, поскольку выбранные алгоритмы сопоставимы по простоте и подходу. Для обучения NGBoost использовалось нормальное распределение и решающее дерево с максимальной глубиной 3.

Dataset	N	RMSE			NLL		
		MC dropout	Deep Ensembles	NGBoost	MC dropout	Deep Ensembles	NGBoost
Boston	506	**2.97 ± 0.85**	**3.28 ± 1.00**	**2.94 ± 0.53**	**2.46 ± 0.25**	**2.41 ± 0.25**	**2.43 ± 0.15**
Concrete	1030	**5.23 ± 0.53**	**6.03 ± 0.58**	**5.06 ± 0.61**	**3.04 ± 0.09**	**3.06 ± 0.18**	**3.04 ± 0.17**
Energy	768	1.66 ± 0.19	2.09 ± 0.29	**0.46 ± 0.06**	1.99 ± 0.09	1.38 ± 0.22	**0.60 ± 0.45**
Kin8nm	8192	0.10 ± 0.00	**0.09 ± 0.00**	0.16 ± 0.00	-0.95 ± 0.03	**-1.20 ± 0.02**	-0.49 ± 0.02
Naval	11934	0.01 ± 0.00	**0.00 ± 0.00**	**0.00 ± 0.00**	-3.80 ± 0.05	**-5.63 ± 0.05**	-5.34 ± 0.04
Power	9568	**4.02 ± 0.18**	4.11 ± 0.17	**3.79 ± 0.18**	**2.80 ± 0.05**	**2.79 ± 0.04**	**2.79 ± 0.11**
Protein	45730	**4.36 ± 0.04**	4.71 ± 0.06	**4.33 ± 0.03**	2.89 ± 0.01	**2.83 ± 0.02**	**2.81 ± 0.03**
Wine	1588	**0.62 ± 0.04**	**0.64 ± 0.04**	**0.63 ± 0.04**	**0.93 ± 0.06**	**0.94 ± 0.12**	**0.91 ± 0.06**
Yacht	308	1.11 ± 0.38	1.58 ± 0.48	**0.50 ± 0.20**	1.55 ± 0.12	1.18 ± 0.21	**0.20 ± 0.26**
Year MSD	515345	**8.85 ± NA**	8.89 ± NA	8.94 ± NA	3.59 ± NA	**3.35 ± NA**	3.43 ± NA

Table 1: Comparison of performance on regression benchmark UCI datasets. Results for MC dropout and Deep Ensembles are reported from Gal and Ghahramani (2016) and Lakshminarayanan et al. (2017) respectively. NGBoost offers competitive performance in terms of RMSE and NLL, especially on smaller datasets.

NGBoost - отличный алгоритм для прогнозной оценки неопределенности, а его производительность, по мнению пользователей, конкурентоспособна по сравнению с современными подходами, такими как LightGBM или RandomForest.