Interior-Point Methods for Logistic Regression

Rubing Shen, François Pacaud

December 4th, 2019

Artelys | OPTIMIZATION SOLUTIONS

Who are we ?

- This work was part of a summer internship in the team developing the non-linear optimization solver Artelys Knitro
- Knitro is able to solve generic problems of the form

$$\min_{x \in \mathbb{R}^d} f(x)$$
$$\text{s.t. } c_L \leq c(x) \leq c_U$$

with $f : \mathbb{R}^d \to \mathbb{R}$ and $c : \mathbb{R}^d \to \mathbb{R}^m$ *smooth* functions

- Knitro has four algorithms implemented :
  - Direct interior point method
  - Trust-region
  - Active-set (Sequential Linear Quadratic Programming)
  - Sequential Quadratic Programming
- How does Knitro perform when applied to logistic regression problems ?

Formulating a logistic regression problem

Settings

- Data : n observations *i.i.d.*

$$\mathcal{D} = \left\{(x_i, y_i) \in \mathbb{R}^d \times \{-1, 1\} \mid i = 1, .., n\right\}$$

- Goal : classify $x \in \mathbb{R}^d$ in -1 or 1

– Generalized linear model : binary random variable $\mathbf{y} : \mathcal{Y} \to \{-1, 1\}$ s.t.

$$\mathbb{P}\left(y = 1|x\right) = \frac{1}{1 + \exp(-\theta^\top x)}$$

– Formulate as a maximum (log-)likelihood estimation

$$\max_{\theta \in \mathbb{R}^d} \sum_{i=1}^{n} \log\left(\frac{1}{1 + e^{-y_i \theta^\top x_i}}\right)$$

Finding the optimal parameter

Finding the optimal regression parameter θ resumes to solve the optimization problem

Optimization problem

$$\min_{\theta \in \mathbb{R}^d} \underbrace{\frac{1}{n} \sum_{i=1}^{n} \log\left[1 + \exp\left(-y_i \theta^\top x_i\right)\right]}_{\text{Training loss}} + \underbrace{\lambda \Omega\left(\theta\right)}_{\text{Regularization}}$$

with difference choices of regularization functions :

- ℓ_2 : $\Omega(\theta) = \|\theta\|_2^2 = \sum_{i=1}^{d} \theta_i^2$
- ℓ_1 : $\Omega(\theta) = \|\theta\|_1 = \sum_{i=1}^{d} |\theta_i|$
- Elastic-net : $\Omega(\theta) = \beta \|\theta\|_1 + (1-\beta)\|\theta\|_2^2$, with $\beta \in [0, 1]$

Solving the logistic regression problem

Logistic regression is a well-known problem

- It formulates as a non-linear problem
- Already studied in (Lin et al., 2008; Friedman et al., 2009)
- Solved with mature solvers
  - L-BFGS-B (Zhu et al., 1997)
  - LIBLINEAR (Fan et al., 2008; Chang and Lin, 2011)
  - `glmnet` (Friedman et al., 2009)
  - And others...
- Currently, more efforts devoted to ℓ_1 regularization

Here, we follow a two step procedure

We suppose given a regularization function Ω (ℓ_1 or ℓ_2). Let

$$\mathcal{L}(\theta, \lambda) = \frac{1}{n} \sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i \theta^\top X_i\right)\right) + \lambda \Omega(\theta)$$

Inner problem : finding optimal parameter θ

Let $\lambda \in \mathbb{R}$ be a regularization parameter. Solve iteratively with Knitro the logistic problem

$$\min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta, \lambda)$$

Outer problem : finding optimal penalization λ

Solve the *bilevel* program

$$\min_{\lambda \in \mathbb{R}_+} \mathcal{L}(\theta^\sharp(\lambda), \lambda)$$

$$\text{s.t. } \theta^\sharp(\lambda) \in \arg\min_{\theta \in \mathbb{R}^d} \mathcal{L}(\theta, \lambda)$$

Plan

Plan

Solving the inner problem with Knitro

In all this section, we suppose given the regularization parameter $\lambda \in \mathbb{R}$
Let

$$f_\lambda(\theta) = \mathcal{L}(\theta, \lambda)$$

We derive the analytical expression of the *gradient* ∇f_λ and *Hessian* $\nabla^2_\lambda f$

Procedure : we rely on scikit-learn Pedregosa et al. (2011)

- We write callbacks for f_λ, ∇f_λ and $\nabla^2 f_\lambda$ with `numpy`
- We write a class inheriting from the class `sklearn.LogisticRegression` (to gain access to the methods `predict` and `score` implemented in Scikit-Learn)
- Overwrite the method `fit` to wrap the solver Knitro

Benchmarks

Comparison procedure

- Computation time before convergence (`logit.fit`)
- Accuracy of the prediction (`logit.predict`)
- Evaluation with *cross validation*

We use the following datasets from `LIBSVM`[1]

Colon-cancer	62	2,000
Covtype.binary	581,012	54
SUSY	5,000,000	18

with all features normalized during the preprocessing

1. `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

We first focus on ℓ_2 regularization

When choosing a ℓ_2 regularization, f_λ writes

$$f_\lambda(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log\left(1 + \exp\left(-y_i\theta^\top x_i\right)\right) + \lambda\|\theta\|_2^2$$

Properties

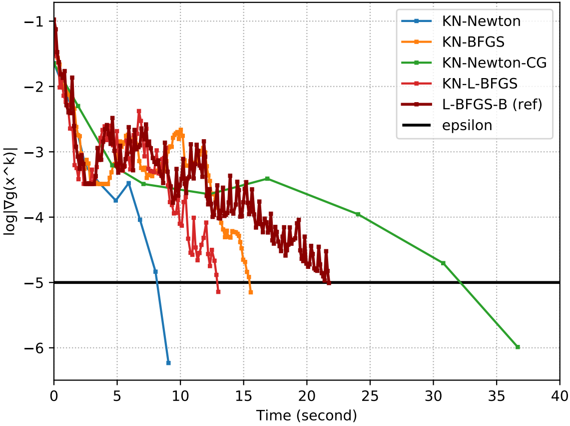
- $f_\lambda : \mathbb{R}^d \to \mathbb{R}$ is convex smooth
- The problem is *unconstrained*

L-BFGS is a well-known algorithm to solve $\min_\theta f_\lambda(\theta)$
Algorithm is mature enough, so benchmarks sum up to

- the linear algebra library used in backend (OpenBLAS, MKL,...)
- the difference in the line-search algorithm

Covtype dataset, ℓ_2 regularization

We compare Knitro with L-BFGS-B (Zhu et al., 1997)



Tackling non-smoothness in ℓ_1 regularization

We now consider a ℓ_1 regularization and rewrite f_λ as

$$f_\lambda(\theta) = \frac{1}{n}\sum_{i=1}^{n} \log\left[1 + \exp\left(-y_i\theta^\top x_i\right)\right] + \lambda\|\theta\|_1$$

- f_λ is a *non-smooth* function !
- We reformulate it to obtain a *constrained* smooth optimization problem

Property

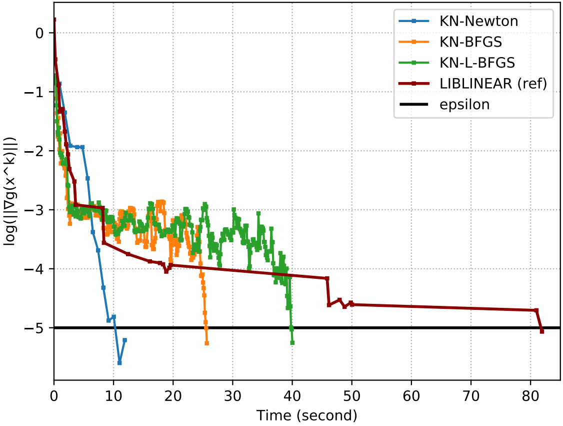
The problem $\min_\theta f_\lambda(\theta)$ is equivalent to

$$\min_{\theta, z \in \mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \log\left[1 + \exp\left(-y_i\theta^\top x_i\right)\right] + \lambda\sum_{j=1}^{d} z_j$$

$$\text{s.t. } z_j \geq -\theta_j, \quad z_j \geq \theta_j \quad \forall j = 1, \cdots, d$$

Covtype, ℓ_1 regularization

We compare Knitro (+crossover mode) with LIBLINEAR (Fan et al., 2008)



Plan

Hyperparameters optimization as a bilevel program

- We now aim at optimizing a given cross-validation score
  For $X \in \mathbb{R}^{n \times d}$, $y \in \mathbb{R}^{n}$, $\theta \in \mathbb{R}^{d}$, let

$$\mathcal{C}(\theta, X, y) = \frac{1}{n} \sum_{i=1}^{n} \log\left[1 + \exp\left(-y_i \theta^\top X_i\right)\right]$$

- Let $V_1, \cdots, V_K$ be K testing sets and $\mathcal{T}_1, \cdots, \mathcal{T}_K$ K training sets
  We define the cross validation loss as

$$\mathcal{C}_{cv}(\lambda) = \frac{1}{K} \sum_{j=1}^{K} \frac{1}{|V_j|} \sum_{(X_i, y_i) \in V_j} \mathcal{C}\left(\theta_j^\sharp(\lambda), X_i, y_i\right)$$

where

$$\theta_j^\sharp(\lambda) \in \underset{\theta \in \mathbb{R}^n}{\arg\min} \left\{ \frac{1}{|T_j|} \sum_{(X_i, y_i) \in T_j} \mathcal{C}(\theta, X_i, y_i) + \lambda \Omega(\theta) \right\}$$

Hyperparameters optimization as a bilevel program

We follow a similar idea as in (Bengio, 2000; Barratt and Sharma, 2018) to compute

$$\min_{\lambda \in \mathbb{R}_+} \mathcal{C}_{cv}(\lambda)$$

by using a dedicated formula to compute $\nabla_\lambda \theta^\sharp(\lambda)$

Theorem

Let $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ a C^2-smooth mapping, strictly convex with respect to the first variable. Then $\theta^* : \mathbb{R}^m \to \mathbb{R}^n$ defined by :

$$\theta^\sharp(\lambda) = \arg\min_{\theta \in \mathbb{R}^n} F(\theta, \lambda)$$

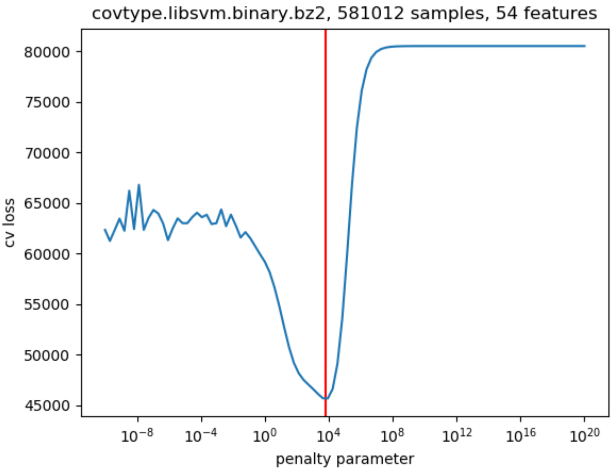
is differentiable and its derivative is given by

$$\nabla_\lambda \theta^\sharp(\lambda) = -\left[\nabla^2_\theta F\left(\theta^\sharp(\lambda), \lambda\right)\right]^{-1} \times \nabla_\lambda \nabla_\theta F\left(\theta^*(\lambda), \lambda\right)$$

Proof : Implicit Function Theorem

Results

Optimizing the ℓ_2 regularization parameter. For `covtype` we get



Plan

Conclusion

- *Inner problem* :
  - We get same performance as L-BFGS-B when using ℓ_2 regularization
  - Knitro gives promising results when solving ℓ_1 regularization
  - Lot of room for improvement (Byrd et al., 2016)
- *Outer problem* :
  - Knitro allows to optimize the regularization hyperparameters

More about Knitro on

`www.artelys.com/docs/knitro`

Barratt, S. and Sharma, R. (2018). Optimizing for generalization in machine learning with cross-validation gradients. *arXiv preprint arXiv :1805.07072*.

Bengio, Y. (2000). Gradient-based optimization of hyperparameters. *Neural computation*, 12(8) :1889–1900.

Byrd, R. H., Chin, G. M., Nocedal, J., and Oztoprak, F. (2016). A family of second-order methods for convex l1-regularized optimization. *Mathematical Programming*, 159(1-2) :435–467.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm : A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3) :27.

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., and Lin, C.-J. (2008). Liblinear : A library for large linear classification. *Journal of machine learning research*, 9(Aug) :1871–1874.

Friedman, J., Hastie, T., and Tibshirani, R. (2009). glmnet : Lasso and elastic-net regularized generalized linear models. *R package version*, 1(4).

Lin, C.-J., Weng, R. C., and Keerthi, S. S. (2008). Trust region newton method for logistic regression. *Journal of Machine Learning Research*, 9(Apr) :627–650.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn : Machine learning in python. *Journal of machine learning research*, 12(Oct) :2825–2830.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. (1997). Algorithm 778 : L-BFGS-B : Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4) :550–560.