Assignment 2

Convex Optimization SS2022

Martin Zach (martin.zach@icg.tugraz.at)

May 9, 2022

Submission: Upload your report as a single `pdf`–file ($\leq 10\,\text{MB}$) to the TeachCenter and include your implementation (`main.py`) with the auxiliary data files (`w.npy`, `winequality-white.csv`) in your submission.

Deadline: May, 16^{th} 2022, 17:00

Regression (15P)

We aim to predict the quality of white wines using the UCI Wine Quality Data Set [2]. This data set consists of N samples $(\mathbf{x}_i, y_i)_{i=1}^N$, where $\mathbf{x}_i \in \mathbb{R}^n$ are manually determined features such as the hue and alcohol level of a wine, and $y_i \in [0, 10]$ is the corresponding quality level. Here, 0 means that the corresponding wine is of "very bad" quality, while the label 10 is only assigned to "excellent" wines.

Our task is to learn a linear regression model

$$\hat{y}_i = \mathbf{x}_i^\top \widetilde{\mathbf{w}} + b$$

such that $\hat{y}_i$ approximates the true quality y_i. Here, $\widetilde{\mathbf{w}} \in \mathbb{R}^n$ are the weights, which are proportional to the normal vector of the associated hyperplane, and $b \in \mathbb{R}$ is the bias, which is proportional to the distance of the hyperplane to the origin. To simplify notation, let us consider the matrix $X \in \mathbb{R}^{N \times n+1}$ holding all features in the data with an additional bias dimension, and the vector $\mathbf{y} \in \mathbb{R}^N$ denoting all labels, i.e.

$$X = \begin{pmatrix} \mathbf{x}_1^\top & 1 \\ \mathbf{x}_2^\top & 1 \\ \vdots & \vdots \\ \mathbf{x}_N^\top & 1 \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}.$$

Then, we would like to find parameters $\mathbf{w} = (\widetilde{\mathbf{w}}\ b)^\top \in \mathbb{R}^{n+1}$ such that

$$\mathbf{y} \approx X\mathbf{w}.$$

Motivated by the maximum-margin principle employed in support vector machine training, we would like to solve[1]

$$\min_{\mathbf{w} \in \mathbb{R}^{n+1}} \left\{ \mathcal{E}(\mathbf{w}) \coloneqq \tfrac{\zeta}{2} \left\| \mathbf{w} \right\|_Q^2 + \sum_{i=1}^N h_i^\epsilon \left((X\mathbf{w})_i \right) \right\}, \tag{P}$$

where $\zeta \in \mathbb{R}_{++}$ is a balancing hyperparameter and $Q \in \mathbb{R}^{n+1 \times n+1}$ is the matrix

$$Q = \begin{pmatrix} \mathrm{Id} & \mathbf{0} \\ \mathbf{0}^\top & 0 \end{pmatrix}$$

with $\mathrm{Id} \in \mathbb{R}^{n \times n}$ being the identity matrix and $\mathbf{0} \in \mathbb{R}^n$ being the zero vector. The second term of the energy $\mathcal{E}$ is a sum over all samples that applies the ϵ-insensitive loss

$$h_i^\epsilon(x) = \max\{|x - y_i| - \epsilon, 0\}$$

to each element of the vector $X\mathbf{w} \in \mathbb{R}^N$.

[1] For a more detailed motivation for this energy, we redirect the interested reader to the lecture *Machine Learning 2* or [1].

Tasks

Perform the following tasks, to solve this classification problem:

1. Derive a subgradient scheme for the optimization problem (P). State the algorithm and all its requirements in your report.
2. Compute the subdifferential of $\mathcal{E}$.
3. Select a suitable step size scheme.
4. Select a proper stopping criterion.
5. Implement the subgradient descent algorithm using `numpy` in the provided script `main.py`. Do not use any further `python`–packages except those that are already included.
6. For $\zeta \in \{1, 1 \times 10^1, 1 \times 10^3\}$, $\epsilon \in \{0.2, 1, 5\}$, compute the optimal weights $\mathbf{w}$ using your subgradient descent method to solve (P). Plot both the energy $\mathcal{E}$ and the mean absolute error

$$\mathcal{M}(\mathbf{w}) = \frac{1}{N} \sum_{i=1}^{N} |(\mathbf{x}_i^\top \ 1)\mathbf{w} - y_i|$$

   as a function of the iteration index. Use a `loglog` plot for the energy and a `semilogx` plot for the accuracy. How do ζ, ϵ influence the optimal $\mathbf{w}$ and the results?
7. For the same ζ and ϵ, compare the results of your subgradient descent method with the results of the following algorithm:
   (a) Choose the same initial weights $\mathbf{w}^0 \in \mathbb{R}^n$ as before and set $\mathbf{v}^0 = \mathbf{0} \in \mathbb{R}^N$.
   (b) Determine the step sizes $\tau = \sigma = \|X\|_2^{-1}$. *Hint:* Use `np.linalg.norm` to compute the induced matrix norm.
   (c) Iterate until convergence:

$$\begin{aligned} \mathbf{w}^{k+1} &= \operatorname{prox}_{\tau \frac{\zeta}{2} \|\cdot\|_Q^2} \left(\mathbf{w}^k - \tau X^\top \mathbf{v}^k \right) \\ \mathbf{v}^{k+1} &= \operatorname{prox}_{\sigma (h^\epsilon)^*} \left(\mathbf{v}^k + \sigma X \left(2\mathbf{w}^{k+1} - \mathbf{w}^k \right) \right), \end{aligned}$$

   where

$$\operatorname{prox}_{\tau \frac{\zeta}{2} \|\cdot\|_Q^2}(\mathbf{w}) = (\mathrm{Id} + \tau\zeta Q)^{-1} \mathbf{w}$$

   with $\mathrm{Id} \in \mathbb{R}^{n+1 \times n+1}$ being the identity matrix, and

$$\operatorname{prox}_{\sigma (h^\epsilon)^*}(\mathbf{v}) = \begin{pmatrix} \operatorname{prox}_{\sigma (h_1^\epsilon)^*}(v_1) \\ \operatorname{prox}_{\sigma (h_2^\epsilon)^*}(v_2) \\ \vdots \\ \operatorname{prox}_{\sigma (h_N^\epsilon)^*}(v_N) \end{pmatrix}, \quad \operatorname{prox}_{\sigma (h_i^\epsilon)^*}(v_i) = \begin{cases} -1 & \text{if } \bar{v}_i < -1, \\ \bar{v}_i & \text{if } \bar{v}_i \in [-1, 1], \\ 1 & \text{if } \bar{v}_i > 1. \end{cases}$$

   where $\bar{v}_i = \operatorname{sign}(v_i - \sigma y_i) \max\{|v_i - \sigma y_i| - \sigma\epsilon, 0\}$. For this algorithm, include the same plots as in the previous task in your report.
8. Discuss the differences between both algorithms and the obtained results. Which algorithm converges faster?

Subdifferential (6P)

- Compute the subdifferential of $f : \mathbb{R} \to \mathbb{R}$ wherever possible for the following functions:
  1. $f(x) = \max\{x, 0\}$
  2. $f(x) = |x|$,
  3. $f(x) = \begin{cases} -x + 1 & \text{if } x < 1, \\ (x-1)^2 & \text{if } x \geq 1. \end{cases}$

- Compute the subdifferential of $f : \mathbb{R}^n \to \mathbb{R}$ wherever possible for the following functions:
  1. $f(\mathbf{x}) = \|A\mathbf{x}\|_1$ for $A \in \mathbb{R}^{m \times n}$,
  2. $f(\mathbf{x}) = \|\mathbf{x}\|_2$,
  3. $f(\mathbf{x}) = \|\mathbf{x}\|_2^2$,
  4. $f(\mathbf{x}) = -\log(\|\mathbf{x}\|_\infty)$,
  5. $f(\mathbf{x}) = \sqrt{\mathbf{x}^\top Q\mathbf{x}}$ for a positive definite matrix $Q \in \mathbb{R}^{n \times n}$.
- A function f is homogeneous of degree $p \geq 0$ if $\operatorname{dom} f$ is a cone and $f(\lambda\mathbf{x}) = \lambda^p f(\mathbf{x})$ for all $\mathbf{x} \in \operatorname{dom} f$ and all $\lambda \geq 0$. Let f be convex, subdifferentiable and homogeneous of degree $p \geq 1$. Prove that

  $$\langle s, \mathbf{x} \rangle = pf(\mathbf{x})$$

  for all $\mathbf{x} \in \operatorname{dom} f$ and all $s \in \partial f(\mathbf{x})$.

Convex Functions (4P)

1. Show that $f : \mathbb{R}^n \to \mathbb{R}$, $\mathbf{x} \mapsto -\exp(-g(\mathbf{x}))$ is convex, where $g : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function with a convex domain and the matrix

   $$\begin{pmatrix} \nabla^2 g(\mathbf{x}) & \nabla g(\mathbf{x}) \\ \nabla g(\mathbf{x})^\top & 1 \end{pmatrix}$$

   is positive definite for all $\mathbf{x} \in \operatorname{dom} g$.
2. Show that a continuous function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if and only if for every $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$

   $$\int_0^1 f(\mathbf{x} + t(\mathbf{y} - \mathbf{x}))\mathrm{d}t \leq \frac{f(\mathbf{x}) + f(\mathbf{y})}{2}.$$

References

[1] Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

[2] P. Cortez, A. Cerdeira, F. Almeida, T. Matos and J. Reis. Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier, 2009.