Labs
Machine Learning Course
Fall 2022

EPFL
School of Computer and Communication Sciences
Nicolas Flammarion & Martin Jaggi
www.epfl.ch/labs/mlo/machine-learning-cs-433

Problem Set 3, Oct 6, 2022 (Theory Questions Part)

1. Warmup :

   (a) We want to show that the sum of two convex functions is convex as well.
   Let $f, g, h \colon X \to \mathbb{R}$ such that $\forall x \in X \quad h(x) = f(x) + g(x)$ and f, g are convex. Then $\forall x \in X \quad \lambda \in [0, 1]$:

$$\begin{aligned} h(\lambda x + (1-\lambda)y) &= f(\lambda x + (1-\lambda)y) + g(\lambda x + (1-\lambda)y) \\ &\leq \lambda f(x) + (1-\lambda) f(y) + \lambda g(x) + (1-\lambda) g(y) \\ &= \lambda \left[f(x) + g(x) \right] + (1-\lambda) \left[f(y) + g(y) \right] \\ &= \lambda h(x) + (1-\lambda) h(y) \end{aligned}$$

   Therefore the sum of two convex fonction is also convex.

   (b) In order to see whether we can solve $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ we want to look at the relative rank of $\mathbf{A}$ and $[\mathbf{A}|\boldsymbol{b}]$ (the extended matrix). Suppose that $\mathbf{A}$ and $\boldsymbol{b}$ have size respectively $m \times n$ and m then $[\mathbf{A}|\boldsymbol{b}]$ has size $m \times (n+1)$. Then :

   - $\mathbf{A}$ is a square matrix s.t. $rank(\mathbf{A}) = m$: the system has a **unique solution**
   - $rank(\mathbf{A}) < rank([\mathbf{A}|\boldsymbol{b}])$: the system has **no solution**
   - $rank(\mathbf{A}) = rank([\mathbf{A}|\boldsymbol{b}]) < n$: the system has **infinitely many solutions**

   (c) The computational complexity of (supposing that we start with $\boldsymbol{w}$ for GD and SGD):

   - Grid Search : Let $|W|_i$ be the number of different values we consider for the i^{th} dimension of $\boldsymbol{w}$ then $O((\prod_i |W|_i) \times N)$. (if we consider the same number of steps for each dimension then $O(|W|^D \times N)$.
   - GD for Linear regression with MSE : $O(N.D)$
   - SGD for Linear regression with MSE : Let B be the batch. We have the MSE loss function :

$$\mathcal{L}_B(\boldsymbol{w}) = \frac{1}{2|B|} \sum_{n \in B} (y_n - \sum_{l=1}^{D} x_{nl} w_l)^2 \tag{1}$$

     Then if we compute the partial differential w.r.t. $\boldsymbol{w}_j$ we have :

$$\frac{\partial \mathcal{L}_B}{\partial w_j}(\boldsymbol{w}) = -\frac{1}{B} \underbrace{\sum_{n \in B} (y_n - \underbrace{\sum_{l=1}^{D} x_{nl} w_l}_{\alpha_n}) \times x_{nj}}_{O(|B|) \text{ considering } \alpha_n \text{ a constant}} \tag{2}$$

     Notice that α_n doesn't depend on j, its computation takes $O(D)$ and is computed only once for all dimensions per sample. Therefore computing $\{\alpha_n : n \in B\}$ takes $O(|B|.D)$ and can be considered as constant for the rest of the analysis.
     Then to compute all the dimensions of the gradient, we will need all the partial derivatives. Therefore the total complexity is $O(|B|D + |B|D) = O(|B|D)$

   (d) We wish to find $\boldsymbol{w} = (w_1, w_2)$ such that it satisfies $\boldsymbol{x}^\top \boldsymbol{w} = w_1 x_1 + w_2 x_2 = y$. To do so we solve the systems (computation behind fairly easy we won't give any detail here) :
   For the first case :

$$\begin{cases} w_1 = -100 \\ w_2 = -200 \end{cases}$$

For the second case :

$$\begin{cases} w_1 = 40'000 \\ w_2 = 79'800 \end{cases}$$

Let's have a better look at what happened, we define the matrix

$$X = \begin{bmatrix} 400 & -201 \\ -800 & 401 \end{bmatrix} \tag{3}$$

which corresponds to the matrix of variables in the first case. This matrix has a condition number $cond(X) \approx 2,503$.

Now from wikipedia, "A problem with a low condition number is said to be well-conditioned, while a problem with a high condition number is said to be ill-conditioned. In non-mathematical terms, an ill-conditioned problem is one where, for a small change in the inputs (the independent variables) there is a large change in the answer or dependent variable". And indeed, we can see in our case that a small change, here of X, has a huge impact on the solution.

Note also that this high condition number is related to the fact that the two columns of A are "nearly" multiples of each other.

2. Cost Functions :

   (a) No need for correction (remember that the cost function is in a 3-dimensional space)

   (b) We want to compute the gradient of

$$\mathcal{L}(\boldsymbol{w}) = \frac{1}{N}\sum_{n=1}^{N}\frac{[\boldsymbol{x}_n^\top \boldsymbol{w} - y_n]^2}{y_n^2 + \epsilon}.$$

   A good first approach to compute a gradient is to compute one of its component

$$\frac{\partial \mathcal{L}}{\partial w_i} = \frac{1}{N}\sum_{n=1}^{N}\frac{2x_{ni}[\boldsymbol{x}_n^\top \boldsymbol{w} - y_n]}{y_n^2 + \epsilon}$$

   (c) Then we can identify

$$\nabla_{\boldsymbol{w}}\mathcal{L} = \tfrac{2}{N}\mathbf{X}^\top\mathbf{D}(\mathbf{X}\boldsymbol{w} - \boldsymbol{y}) = \tfrac{2}{N}\mathbf{X}^\top\mathbf{D}(-\boldsymbol{e}) \tag{4}$$

   with $\mathbf{D} = diag(\frac{1}{y_1^2+\epsilon}, ..., \frac{1}{y_N^2+\epsilon})$ a diagonal matrix.

   Finally,

$$\nabla_{\boldsymbol{w}}\mathcal{L} = -\frac{2}{N}\mathbf{X}^\top\mathbf{D}\boldsymbol{e} \tag{5}$$

   (d) You can see that the function is very sensitive to outliers, as the relative error is extremely high. If we call $\mathcal{L}_1$ the cost function we've been working with for the whole exercise and $\mathcal{L}_2$ the one that we want to compare it with we have :

	$\mathcal{L}_1$	$\mathcal{L}_2$
$y = 1$ and $\hat{y} = 10$	40.5	≈ 0.54814
$y = 1$ and $\hat{y} = 100$	$4'900.5$	≈ 2.90120