CS 541 Artificial Intelligence: Homework 3

Instructor: Jie Shen

Due: 11/08/2020, 20:00 pm EST

Gradient Calculation

Suppose $\boldsymbol{x} \in \mathbb{R}^d$ and $y \in \mathbb{R}$ are known. Calculate the gradient of the following functions:

- Sigmoid function $F(\boldsymbol{w}) = 1/(1 + e^{-\boldsymbol{x} \cdot \boldsymbol{w}})$;
- Logistic loss $F(\boldsymbol{w}) = \log(1 + e^{-y\boldsymbol{x} \cdot \boldsymbol{w}})$.

Note: we take both $\boldsymbol{x}$ and $\boldsymbol{w}$ as column vectors.

Linear Regression

Suppose we are given a data set $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$, where each $\boldsymbol{x}_i \in \mathbb{R}^d \times \mathbb{R}$ is a row vector. We hope to learn a mapping f such that each y_i is approximated by $f(\boldsymbol{x}_i)$. Then a popular approach is to fit the data with *linear regression* – it assumes there exists $\boldsymbol{w} \in \mathbb{R}^d$ such that $y_i \approx \boldsymbol{w} \cdot \boldsymbol{x}_i$. In order to learn $\boldsymbol{w}$ from the data, it typically boils down to solving the following *least-squares* program:

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} F(\boldsymbol{w}) := \frac{1}{2} \left\| \boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} \right\|_2^2, \tag{1}$$

where $\boldsymbol{X}$ is the data matrix with the ith row being $\boldsymbol{x}_i$, and $\boldsymbol{y} = (y_1, y_2, \ldots, y_n)^\top$.

1. Compute the gradient and the Hessian matrix of $F(\boldsymbol{w})$, and show that (1) is a convex program.
2. Note that (1) is equivalent to the following:

   $$\min_{\boldsymbol{w} \in \mathbb{R}^d} \left\| \boldsymbol{y} - \boldsymbol{X}\boldsymbol{w} \right\|_2^{100},$$

   in the sense that any minimizer of (1) is also an optimum of the above, and vice versa. State why we stick with the least-squares formulation.
3. State a possible condition on $\boldsymbol{X}$ such that $F(\boldsymbol{w})$ is strongly-convex. Under which condition on $\boldsymbol{X}$ the objective function is not strongly-convex.
4. Consider $n = 100$ and $d = 40$. Generate the data matrix $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ and the response $\boldsymbol{y} \in \mathbb{R}^n$, for example, using the python API `numpy.random.randn`. Then calculate the exact solution

$\boldsymbol{w}^* = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{y}$ of (1). Use python API to calculate the minimum and maximum eigenvalue of the Hessian matrix, and derive the upper bound on the learning rate η in gradient descent. Let us denote this theoretical bound by η_0. Run GD on the data set with 6 choices of learning rate: $\eta \in \{0.01\eta_0, 0.1\eta_0, \eta_0, 2\eta_0, 20\eta_0, 100\eta_0\}$. Plot the curve of "$\left\|\boldsymbol{w}^t - \boldsymbol{w}^*\right\|_2$ v.s. t" for $1 \leq t \leq 100$ and summarize your observation. Note that you can start GD with $\boldsymbol{w}^0 = \boldsymbol{0}$.

5. Consider $n = 100$ and $d = 200$, and generate $\boldsymbol{X}$ and $\boldsymbol{y}$. What happens when you are trying to calculate the closed-form solution $\boldsymbol{w}^*$? In this case, can we still apply GD? If yes, derive the theoretical bound η_0 and run GD with 6 different η as before. Plot the curve of "$F(\boldsymbol{w}^t)$ v.s. t" for $1 \leq t \leq 100$ and summarize your observation.