3 Gradient Calculation

Suppose $\boldsymbol{x}$ and y are known, $\boldsymbol{w} \in \mathbb{R}^d$ is a column vector. Consider the following functions that have been broadly used in machine learning:

- Sigmoid function $F(\boldsymbol{w}) = 1/(1 + e^{-\boldsymbol{x}\cdot\boldsymbol{w}})$;
- Hinge loss $F(\boldsymbol{w}) = \max\{1 - y\boldsymbol{x} \cdot \boldsymbol{w}, 0\}$;
- ℓ_1-norm $F(\boldsymbol{w}) = \|\boldsymbol{w}\|_1$.

1. Use python to plot their curves for the case $d = 1$. You can set $x = y = 1$;
2. Derive their gradient or subgradients for a general $d > 0$.

3.1 Implementation

Gradient descent is typically used to solve a general optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} F(\boldsymbol{w}).$$

It starts from an arbitrary point $\boldsymbol{w}^0$ and gradually refines the solution as

$$\boldsymbol{w}^t \leftarrow \boldsymbol{w}^{t-1} - \eta \cdot \nabla F(\boldsymbol{w}^{t-1}).$$

Fix $d = 1$, i.e., the variable $\boldsymbol{w}$ is a scalar. Further fix $w^0 = 1$.

1. Consider $F(w) = \frac{1}{2}w^2$. For each learning rate
$$\eta \in \{10^{-4}, 10^{-3}, 0.01, 0.1, 0.5, 1, 2, 5, 10, 100\},$$
calculate the sequence $\{w^t\}_{t=1}^{1000}$ generated by GD and plot the curve "$\left|w^t\right|$ v.s. t".
2. Consider $F(w) = \frac{1}{2}w^4$. For each learning rate
$$\eta \in \{10^{-4}, 10^{-3}, 0.01, 0.1, 0.5, 1, 2, 5, 10, 100\},$$
calculate the sequence $\{w^t\}_{t=1}^{1000}$ generated by GD and plot the curve "$\left|w^t\right|$ v.s. t".

4 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:

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} F(\boldsymbol{w}) := \frac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{w}\right\|^2, \tag{4.1}$$

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 (6.4) is a convex program.

2. Note that (6.4) is equivalent to the following:

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

in the sense that any minimizer of (6.4) is also an optimum of the above, and vice versa. State why we stick with the least-squares formulation.

3. State when the objective function is strongly-convex and when it is not.

4. Fix $n = 1000$ and increase d from 20 to 500, with a step size 20. For each problem size (n, d), 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 $\boldsymbol{w}^* = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{y}$ of (6.4) and record the computation time. Plot the curve of "time v.s. d" and summarize your observation.

5. Consider $n = 100$ and $d = 40$. Again, generate $\boldsymbol{X}$ and $\boldsymbol{y}$, and calculate the optimal solution. 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 (see the slides for the bound). 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 "$\|\boldsymbol{w}^t - \boldsymbol{w}^*\|$ v.s. t" for $1 \le t \le 100$ and summarize your observation. Note that you can start GD with $\boldsymbol{w}^0 = \boldsymbol{0}$.

6. 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 $1 \le t \le 100$ and summarize your observation.