Convex Optimization and Gradient Descent in Data Science

1 Introduction to Convex Optimization

Convex optimization is a fundamental topic in data science, enabling efficient solutions to many optimization problems. The concepts of convexity and convex functions are crucial because convex problems guarantee that any local minimum is also a global minimum, simplifying the search for optimal solutions.

1.1 Key Concepts

1. **Convex Set**: A set $S \subseteq \mathbb{R}^n$ is convex if, for any $x, y \in S$ and $\theta \in [0, 1]$:

$$\theta x + (1 - \theta)y \in S$$

(Rationale: This definition ensures that every point on the line segment connecting any two points in the set also lies within the set, reflecting the "non-curving inward" property of convex sets.)

2. **Convex Function**: A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if, for any $x, y \in \mathbb{R}^n$ and $\theta \in [0, 1]$:

$$f(\theta x + (1 - \theta)y) \leq \theta f(x) + (1 - \theta)f(y)$$

(Rationale: This inequality guarantees that the function lies below or on the straight line connecting $f(x)$ and $f(y)$, indicating no local dips or peaks between x and y.)

3. **Objective Function**: The function to be minimized or maximized, denoted $f(x)$.

4. **Constraints**: Additional conditions that the solution must satisfy, often expressed as inequalities or equalities.

1.2 Convex Optimization Problem

A general convex optimization problem can be formulated as:

$$\text{Minimize: } f(x) \quad \text{subject to: } g_i(x) \leq 0,\ h_j(x) = 0$$

where $f(x)$ and $g_i(x)$ are convex functions, and $h_j(x)$ is affine.

2 Gradient Descent

Gradient Descent is an iterative optimization algorithm used to minimize an objective function. It is widely applied in machine learning, deep learning, and data science to find the optimal parameters of models.

2.1 Key Idea

The algorithm adjusts the parameters x in the opposite direction of the gradient of the objective function $f(x)$, which points toward the steepest ascent. By moving opposite to the gradient, $f(x)$ is minimized.

2.2 Algorithm

1. **Initialization**: Start with an initial guess x_0 for the parameters.
2. **Gradient Computation**: Compute the gradient of the objective function, $\nabla f(x_k)$, at the current parameter x_k.
3. **Update Rule**: Update the parameters using:

   $$x_{k+1} = x_k - learning\_rate \nabla f(x_k)$$

   where $learning\_rate > 0$ is the learning rate parameter.
4. **Stopping Criterion**: Stop the algorithm when the gradient magnitude is sufficiently small or the change in the objective function is below a threshold.

3 Example: Least Squares Regression

Objective: Minimize the Mean Squared Error (MSE) for a linear regression model.

1. **Objective Function**:

$$f(w) = \frac{1}{2m} \sum_{i=1}^{m} \left(y_i - x_i^T w\right)^2$$

where:

- w: Parameter vector
- x_i: Feature vector for the i-th example
- y_i: Observed value for the i-th example
- m: Number of training examples

2. **Gradient**:

$$\nabla f(w) = -\frac{1}{m} \sum_{i=1}^{m} \left(y_i - x_i^T w\right) x_i$$

3. **Update Rule**:

$$w_{k+1} = w_k - learning\_rate \nabla f(w_k)$$

4 Key Properties and Insights

1. **Convexity**: Gradient descent is guaranteed to converge to the global minimum for convex functions, provided the learning rate is appropriate.

2. **Learning Rate Parameter**:

   - If $learning\_rate$ is too large: The algorithm may diverge.
   - If $learning\_rate$ is too small: The algorithm converges slowly.

3. **Applications**:

   - Logistic Regression
   - Support Vector Machines (SVM)
   - Neural Networks (for non-convex functions)

5 Pseudocode for Gradient Descent

```
# Gradient Descent Algorithm
Input: Objective function f(x), gradient function grad_f(x),
    initial guess x_0, learning_rate, tolerance tol
Output: Optimal parameters x_opt

# Initialization
x = x_0

# Iterative updates
while True:
    grad = grad_f(x)                  # Compute gradient
    x_new = x - learning_rate * grad # Update parameters
    if abs(f(x_new) - f(x)) < tol:   # Check convergence
        break
    x = x_new                         # Update for next
    iteration

return x
```

6 Practical Considerations

1. **Batch Gradient Descent**: Uses all training data in each iteration.
2. **Stochastic Gradient Descent (SGD)**: Uses one training example per iteration; faster but noisier.
3. **Mini-batch Gradient Descent**: A compromise between batch and stochastic, using subsets of training data.

This summary provides the essential understanding of convex optimization and gradient descent, equipping you with theoretical and practical knowledge for applications in data science.