Problem 10

Consider a linear regression problem in which we want to weight different training examples differently. Specifically, suppose we want to minimize:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{m} w^{(i)}(\theta^T x^{(i)} - y^{(i)})^2$$

Part A
Show that $J(\theta)$ can be written in the form:

$$J(\theta) = (X\theta - y)^T W(X\theta - y)$$

for an appropriate diagonal matrix W, where X is the $m \times d$ input matrix and y is a $m \times 1$ vector denoting the associated outputs. State clearly what W is.

Solution

We can express $X\theta - y$:

$$X\theta - y = \begin{bmatrix} \theta^T x^{(1)} - y^{(1)} \\ \theta^T x^{(2)} - y^{(2)} \\ \vdots \\ \theta^T x^{(m)} - y^{(m)} \end{bmatrix} = \begin{bmatrix} \theta^T x^{(1)} \\ \theta^T x^{(2)} \\ \vdots \\ \theta^T x^{(m)} \end{bmatrix} - \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

We can rewrite $J(\theta)$ as follows:

$$J(\theta) = \frac{1}{2} (X\theta - y)^T \operatorname{diag}(w) (X\theta - y)$$

Here, $\operatorname{diag}(w)$ represents a diagonal matrix with the weights $w^{(i)}$ on the diagonal:

$$\operatorname{diag}(w) = \begin{bmatrix} w^{(1)} & 0 & \cdots & 0 \\ 0 & w^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w^{(m)} \end{bmatrix}$$

Now, let's perform the matrix multiplications and transpose:

$$\begin{aligned} J(\theta) &= \frac{1}{2} (X\theta - y)^T \operatorname{diag}(w) (X\theta - y) \\ &= \frac{1}{2} \left(\begin{bmatrix} \theta^T x^{(1)} - y^{(1)} \\ \theta^T x^{(2)} - y^{(2)} \\ \vdots \\ \theta^T x^{(m)} - y^{(m)} \end{bmatrix} \right)^T \begin{bmatrix} w^{(1)} & 0 & \cdots & 0 \\ 0 & w^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w^{(m)} \end{bmatrix} \begin{bmatrix} \theta^T x^{(1)} - y^{(1)} \\ \theta^T x^{(2)} - y^{(2)} \\ \vdots \\ \theta^T x^{(m)} - y^{(m)} \end{bmatrix} \end{aligned}$$

Now, we have the expression in the desired form, where W is a diagonal matrix:

$$J(\theta) = (X\theta - y)^T \operatorname{diag}(w) (X\theta - y)$$

So, W is a diagonal matrix where the diagonal elements are the weights $w^{(i)}$:

$$W = \operatorname{diag}(w) = \begin{bmatrix} w^{(1)} & 0 & \cdots & 0 \\ 0 & w^{(2)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w^{(m)} \end{bmatrix}$$

Part B

If all the $w^{(i)}$'s are equal to 1, the normal equation to solve for the parameter θ is:

$$X^T X\theta = X^T y$$

and the values of θ that minimizes $J(\theta)$ is $(X^T X)^{-1} X^T y$. By computing the derivative of the weighted $J(\theta)$ and setting it equal to zero, generalize the normal equation to the weighted setting and solve for θ in closed form in terms of W, X, and y.

Solution

To generalize the normal equation to the weighted setting and solve for θ in closed form in terms of W, X, and y, we will compute the derivative of $J(\theta)$ with respect to θ and set it equal to zero:

$$\begin{aligned}\frac{\partial J(\theta)}{\partial \theta} &= \frac{\partial}{\partial \theta}\left(\frac{1}{2}\sum_{i=1}^{m} w^{(i)}(\theta^T x^{(i)} - y^{(i)})^2\right) \\ &= \frac{1}{2}\sum_{i=1}^{m} 2w^{(i)}(\theta^T x^{(i)} - y^{(i)})\frac{\partial}{\partial \theta}(\theta^T x^{(i)} - y^{(i)}) \\ &= \sum_{i=1}^{m} w^{(i)}(\theta^T x^{(i)} - y^{(i)})x^{(i)}\end{aligned}$$

Setting the derivative equal to zero and solve for θ:

$$\begin{aligned}0 &= \sum_{i=1}^{m} w^{(i)}(\theta^T x^{(i)} - y^{(i)})x^{(i)} \\ 0 &= \sum_{i=1}^{m} w^{(i)}(\theta^T x^{(i)} x^{(i)} - y^{(i)}x^{(i)}) \\ 0 &= \sum_{i=1}^{m} w^{(i)}\theta^T x^{(i)}x^{(i)} - \sum_{i=1}^{m} w^{(i)}y^{(i)}x^{(i)} \\ 0 &= X^T W X\theta - X^T W y \\ \theta &= (X^T W X)^{-1}X^T W y\end{aligned}$$

Part C

To predict the target value for an input vector x, one choice for the weighting functions $w^{(i)}$ is:

$$w^{(i)} = exp\left(-\frac{(x - x^{(i)})^T(x - x^{(i)})}{2\tau^2}\right)$$

Points near x are weighted more heavily than points far away from x. The parameter τ is a bandwidth defining the sphere of influence around x. Note how the weights are defined by the input x. Write down an algorithm for calculating θ by gradient descent for locally weighted linear regression. Is locally weighted linear regression a parametric or a non-parametric method?

Solution

Locally weighted linear regression is a non-parametric method since it does not assume a set relationship between the input and the target variable. It adapts the model based on the structure of the data, weighting data points that are closer more heavily.

To use locallly weighted lienar regression with gradient descent, coefficient vector θ needs to be adjusted based on minimizing the cost function $J(\theta)$:

1. Initialize θ to random values
2. Choose learning rate α and a convergence criteria
3. Repeat the following until convergence:

```
1: function GRADIENT-DESCENT-ON-LWLR(initial, α, iterations)
2:     Initialize θ to initial                          ▷ Initialize model parameters
3:     m ← number of training examples
4:     repeat
5:         for i from 1 to m do                         ▷ For each training example
6:             w^(i) = exp(−(x−x^(i))^T(x−x^(i)) / 2τ^2)
7:             δ^(i) = w^(i)(θ^T x^(i) − y^(i))
8:             θ_j = θ_j − (α/m) Σ_{i=1}^{m} δ^(i) x_j^(i)
```

```
        end for
    until convergence or iterations reached
    return θ                                   ▷ Final model parameters
end function
```