Foundation of Machine Learning (IT 582)

Autumn 2022

Pritam Anand

Given the training set $\{(x_i, y_i) : x_i \in \mathbf{R}^n, y_i \in \mathbf{R} \text{ for } i = 1, 2, ..l, \}$, the regularized Least Squares Regression model solves the optimization problem

$$\min_{(w)} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1}^{l} (y_i - (w^T \phi(x_i)))^2 \tag{1}$$

where $\phi(x) = \begin{bmatrix} \phi_1(x) \\ \phi_2(x) \\ .... \\ ..... \\ \phi_m(x) \end{bmatrix}$ and $\phi_j(x)$ for $j = 1, 2, ..., m,$ are m different basis functions.

Though the solution of the regularized Least Squares Regression models is computationally handful, they often obtain poor prediction if $m >> l$ i,e., the number of basis functions used is much more than the number of available training data points. It is often the case when we are working in high dimensions with small sample size data.

LASSO Regression model

This problem with the Least Square Regression model can be solved if we can obtain a solution vector $w \in \mathbb{R}^m$, such that its several components are equal to zero. Obtaining a sparse solution vector w means that we are working with fewer basis functions which can reduce the risk of over-fitting. It allows us to work with a simpler model and decrease the variance of the estimated functions.

The Least Absolute Shrinkage and Selection Operator (LASSO) regression model minimizes the L_1-norm of regularization along with the Least Squares loss function in its optimization problem. The minimization of L_1-norm of regularization helps it to obtain a sparse solution vector. The LASSO regression model solves the problem

$$\min_{(w)} \frac{\lambda}{2} ||w||_1 + \frac{1}{2} \sum_{i=1}^{l} (y_i - (w^T \phi(x_i)))^2, \tag{2}$$

where $w \in \mathbb{R}^m$ and $||w||_1$ is obtained by $\sum_{i=1}^{m} |w_i|$.

We note that objective function of the LASSO problem is convex but, no more differentiable as the $||w||_1$ is not differentiable at $\mathbf{0} \in R^m$. But, we can still compute the subgradient for it. Before we make use of the sub-gradients concept, we need to briefly define it.

Definition (Sub-gradients): A sub-gradient of a convex function f (f may not be differentiable) at $x \in \mathbb{R}^m$ is any $g \in \mathbb{R}^m$ such that the following holds

$$f(y) \geq f(x) + g^T(y - x), \forall y \in \mathbb{R}^m.$$

For example, if we consider a real number t, then $|t|$ is not differentiable at $t = 0$. But, the subgradient g can be obtained as $g = +1$ for $t > 0$, $g = -1$ for $t < 0$. and g is any element in $[-1, 1]$ for $t = 0$.

Subgradient descent method for LASSO regression model

For $w \in \mathbb{R}^m$, we can obtain the subgradients $g = \delta(||w||_1)$ using

$$g[i] = \begin{cases} = \text{sign}(w(i)), & \text{if} \;\; w[i] \neq 0. \\ \text{any element in}[-1, 1], & \text{if} \;\; w[i] = 0. \end{cases} \tag{3}$$

If we deonte the objective functions of the problem (2) using $J(w)$ then we can uniquely obtain the $m \times 1$ subgradient vector $\delta(J(w))$, using

$$\delta(J(w)) = \frac{\lambda}{2} g - \sum_{i=1}^{l} (y_i - (w^T \phi(x_i)))\phi(x_i). \tag{4}$$

Now, we list the steps of the subgradient descent algorithm which can obtain the solution to the LASSO regression problem (2).

Algorithm 1 Subgradient method for LASSO regression model

```
Input :- Training set $T$ and tolerance $\epsilon$
Intailize $w^0 \in \mathbb{R}^m$
Repeat
$w^{k+1} = w^k$ - $\eta\delta(J(w^k))$
until $||\delta(J(w^k)|| \leq \epsilon$.
```

Also, we list the steps of the stochastic subgradient descent algorithm which can obtain the solution of the LASSO regression problem (2) in Algorithm 2.

Algorithm 2 Stochastic Subgradient method for LASSO regression model

```
Input :- Training set $T$ and $max\_iter$
Intailize $w^0 \in \mathbb{R}^m$
for i = 1 to $max\_iter$ do
  Draw a random subset $B$ of size $k$ from Training set $T$.
  $w^{k+1} = w^k$ - $\eta_i\delta(J_B(w^k))$
end for.
```

The $\delta(J_B(w^k))$ used in Algorithm 2 computes the subgradient for only k data points of set B as

$$\delta(J(w)) = \frac{\lambda}{2}g - \sum_{(x_i,y_i)\in B} (y_i - (w^T\phi(x_i)))\phi(x_i).$$

Beyond the Least Squares loss function

A major problem with the Least Squares based Regression models is that the estimates obtained by them can deviate significantly if we change a few of the y_i values with outliers (You can empirically realize this while doing Assignment 3). It is because of that the Least Squares loss function is not a robust loss function as it considers sum of the square of errors.

In literature, you can find several ideas that attempt to bring robustness back to the Least Square Regression models. A most popular class of them consider different weighting mechanisms, which assign higher weights to dense data points and assign lower weights to outliers. Can you think of a simple and effective version of the Weighted Least Square Regression model?

L_1-norm loss function based regression model

The L_1-norm loss function is one of the popular robust loss functions which considers the sum of absolute values of errors. For u in $\mathbb{R}^l$, it is given by $L(u) = ||u||_1 = \sum_{i=1}^{l}(|u_i|)$. For our regression problem , we shall consider $u_i = y_i - (w^T\phi(x_i) + b)$ for $i = 1, 2, .., l$.

The L_1-norm loss regression model solves the following optmization problem

$$\min_{(w)} \frac{\lambda}{2}w^Tw + \frac{1}{2}\sum_{i=1}^{l}|y_i - (w^T\phi(x_i))| \tag{5}$$

where $\phi(x) = \begin{bmatrix}\phi_1(x)\\ \phi_2(x)\\ ....\\ .....\\ \phi_m(x)\end{bmatrix}$ and $\phi_j(x)$ for $j = 1, 2, ..., m$, are m different basis functions. The optimization problem (5) is a convex but, not a smooth function of w as $\sum\limits_{i=1}^{l}|y_i - (w^T\phi(x_i))|$ is not differentiable .

Subgradient descent method for L_1-norm loss regression model

At first, let us obtain the sub-gradeints of L_1-norm loss for an any arbitrary data point (x_j, y_j) i,e., $|y_j - (w^T\phi(x_j))|$. We can obtain the m-dimensional subgradient vectors $g(x_j, y_j) = \delta(|y_j - (w^T\phi(x_j))|)$ using

$$g(x_j, y_j) = \begin{cases} -\phi(x_j), & \text{if } \; y_j - (w^T\phi(x_j)) > 0. \\ \phi(x_j), & \text{if } \; y_j - (w^T\phi(x_j)) < 0. \\ r\phi(x_j), & \text{if } \; y_j - (w^T\phi(x_j)) = 0, \; \text{where r is any element in [-1,1].} \end{cases}$$

$$= \begin{cases} -\text{sign}(y_j - (w^T\phi(x_j))) \; \phi(x_j) \\ r\phi(x_j), & \text{if } \; y_j - (w^T\phi(x_j)) = 0, \; \text{where r is any element in [-1,1].} \end{cases}$$

If we deonte the objective function of the problem (5) using $J(w)$ then we can uniquely obtain the $m \times 1$ subgradient vector $\delta(J(w_2))$, using

$$\delta(J(w)) = \lambda w + \frac{1}{2}\sum_{i=1}^{l} g(x_i, y_i). \tag{6}$$

Now, we present the steps of the subgradient descent algorithm which can obtain the solution for the L_1-nrom loss regression model (5).

Algorithm 3 Subgradient method for L_1-nrom loss regression model

```
Input :- Training set $T$ and tolerance $\epsilon$
Intailize $w^0 \in \mathbb{R}^m$
Repeat
$w^{k+1} = w^k$ - $\eta(\delta(J(w^k))$
until $||\delta(J(w^k)|| \leq \epsilon$.
```

Also, we list the steps of the stochastic subgradient descent algorithm which can obtain the solution of the L_1-nrom loss regression model (5) in Algorithm 4. The $\delta(J_B(w^k))$ used in Algorithm 4 computes the subgradient for only k data

Algorithm 4 Stochastic Subgradient method for L_1-nrom loss regression model

```
Input :- Training set $T$ and $max\_iter$
Intailize $w^0 \in \mathbb{R}^m$
for i = 1 to $max\_iter$ do
  Draw a random subset $B$ of size $k$ from Training set $T$.
  $w^{k+1} = w^k$ - $\eta_i\delta(J_B(w^k))$
end for.
```

points of set B as

$$\delta(J(w)) = \lambda w - \frac{1}{2}\sum_{(x_i,y_i)\in B} g(x_i, y_i).$$

Thanks
Pritam.

.