Project Report

LassoNet: Neural Network with feature sparsity

Submitted By-
Vipin Kumar
MA20MSCST11002

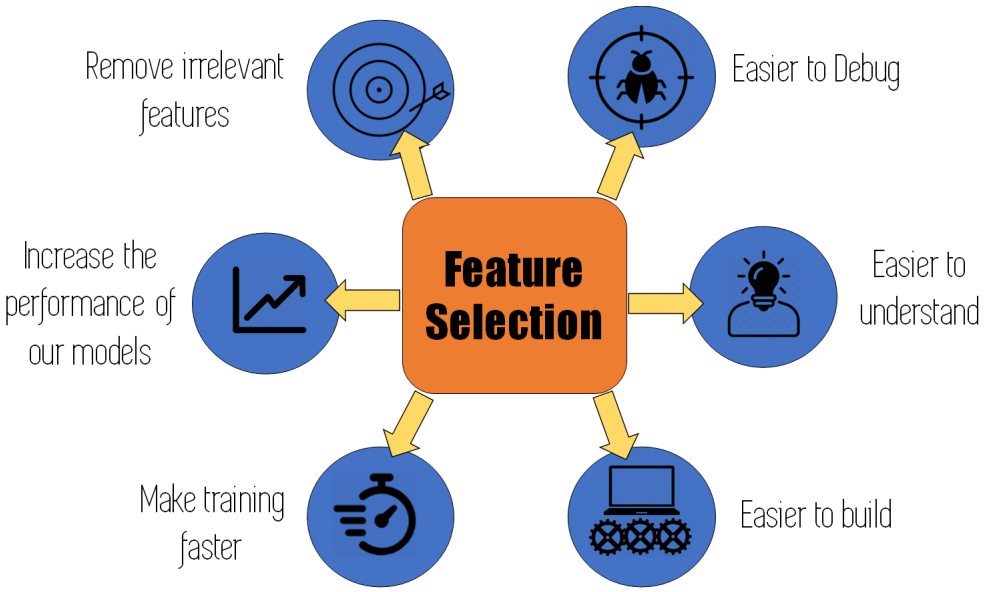
Submitted To-
Dr. Amit Tripathi
Department of Mathematics
IIT Hyderabad

1. Introduction

Features are individual independent variables that act like a input in the machine learning models.

Feature selection is the process of selecting the most important features to input in machine learning algorithms. Feature selection techniques are employed to reduce the number of input variables by eliminating redundant or irrelevant features and narrowing down the set of features to those most relevant to the machine learning model.

Importance of feature selection -



Types of feature selection :

1) Filter Method : Filter methods are generally used as a preprocessing step. The selection of features is independent of any machine learning algorithms. Instead, features are selected on the basis of their scores in various statistical tests for their correlation with the outcome variable.

2) Wrapper Method : The feature selection algorithm exits as a wrapper around the predictive model algorithm and uses the same model to select best features.

3) Embedded Method : It combine the qualities of filter and wrapper methods. It is implemented by algorithms that have their own built-in feature selection methods.

2. Lasso Regression

"LASSO" stands for **L**east **A**bsolute **S**hrinkage and **S**election **O**perator.

Lasso regression is a regularization technique. It is used over regression methods for a more accurate prediction. This model uses shrinkage. The lasso procedure encourages simple and sparse models (i.e. models with fewer parameters).

Lasso regression performs $L1$ **regularization**, which adds a penalty equal to the absolute value of the magnitude of coefficients. This type of regularization can result in sparse models with few coefficients. Some coefficients can become zero and eliminated from the model.

Goal of the algorithm is to minimize :

$$\sum_{i=1}^{n}(y_i - \hat{y_i})^2 + \lambda \sum_{j=1}^{d} |m_j|$$

A tuning parameter, λ controls the strength of the $L1$ penalty. λ is basically the amount of shrinkage:
1) When $\lambda = \mathbf{0}$, no parameters are eliminated. The estimate is equal to the one found with linear regression.
2) As λ increases, more and more coefficients are set to zero and eliminated (theoretically, when $\lambda = \infty$, all coefficients are eliminated).
3) As λ increases, bias increases.
4) As λ decreases, variance increases.

Main **drawback of Lasso regression** is that it can work only with **Linear data**.

So to over come this problem **LassoNet** is introduced.

3. LassoNet

Is an embedded method and the class of residual feed-forward neural network.

$$F = \{\mathrm{f} \equiv \mathrm{f}_{\theta,W} : x \to \theta^T \mathbf{x} + gw(\mathbf{x})\}$$

where **gw** denotes a feed-forward neural network with weights **W**, **d** denotes the data dimension and $\theta \in \mathbf{R}^d$ denotes the weights in the residual layer.

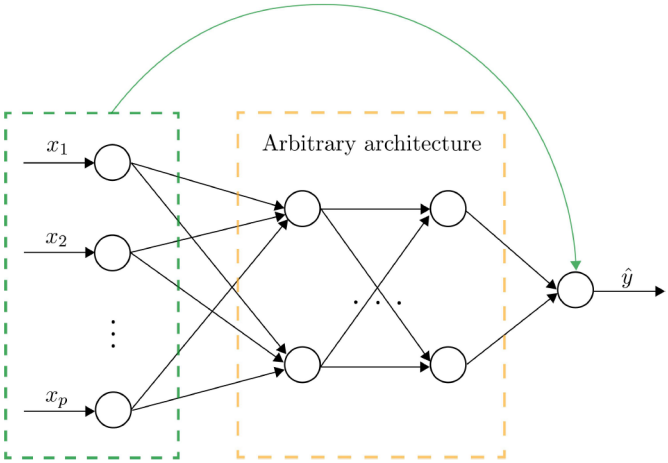


Figure 1: LassoNet architecture

The architecture of LassoNet consists of a single residual connection, shown in green and an arbitrary feed-forward neural network, shown in black.

LassoNet objective function is defined as :

$$minimize_{\theta,W}\ L(\theta, W) + \lambda\ ||\theta||_1$$

$$\text{subject to } ||W_j^{(1)}||_\infty \leq M|\theta_j|,\ j = 1, 2, ..., d$$

where $L(\theta, W) = \frac{1}{n}\sum_{i=1}^{n} l(f_{\theta,W}(x_i), y_i)$ is the **empirical loss**[2] on the training data set, consisting of samples and l is the loss function (i.e. squared-error).
M is the **hierarchy coefficient**, it controls the relative strength of the linear and non-linear components. l_1 **penalty**, λ. Higher values of λ encourage sparser models.

Theorem (The Karush-Kuhn-Tucker conditions (KKT)) Assuming that the convex optimization problem stated above satisfies the Slater condition. Then the optimal solution x^* for the problem satisfies the following conditions:
1) Stationarity: $0 \in \partial\left(f(x^*) + \sum_{i=1}^{m} s_i^* f_i(x^*) + \sum_j v_j A_j x^*\right)$
2) Complementary slackness: $s_i^* f_i(x^*) = 0$ for all $i = 1, \cdots, m$.
3) Primal feasibility: $f_i(x^*) \leq 0$ and $Ax^* = b$,
4) Dual feasibility: $s_i^* \geq 0$ for all $i = 1, \cdots, m$.

Conversely, any solution of the above equations, is the optimal solution for the optimization problem considered above.

Theorem (Lasso) - Let $u \in \mathbb{R}^k$ and $V \in \mathbb{R}^K$. Consider the problem

$$\begin{aligned} \text{minimize}_{b\in\mathbb{R}^k, W\in\mathbb{R}^K} \ & \frac{1}{2}\left(||u-b||_2^2 + ||V-W||_2^2\right) + \lambda||b||_2 + \mu||W||_1 \\ \text{subject to} \ & ||W||_\infty \leq M||b||_2. \end{aligned}$$

The sufficient and necessary condition for characterizing the global optimum (b^*, W^*) of the above optimization problem is described as follows:
1) Re-arrange the index set of the coordinates $\{U_i\}$ as $\{U_{(i)}\}$ such that

$$0 = |U_{(K+1)}| \leq |U_{(K)}| \leq \cdots \leq |U_{(1)}| \leq |U_{(0)}| = \infty$$

and define for all $s \in [K]$

$$b_s = \frac{1}{1+sM^2}\left(1 - \frac{a_s}{||v||_2}\right)_+ v, \text{ where } a_s = \lambda - M\sum_{i=1}^{s}\left(|U_{(i)}| - \mu\right).$$

2) Then $b^* = b_{s^*}$ where s^* is the unique s such that

$$s^* \in \left[S_\mu|U_{(s^*+1)}|, S_\mu|U_{(s^*)}|\right].$$

3) The W^* must satisfy $W^* = sign(U^*)\min\{M||b^*||_2, S_\mu(|U|)\}$.

Outline of the proof

The proof is divided into 4 main steps:
Step 1: Define a map $\Theta : \mathbb{R}^k \to \mathbb{R}^K$, as

$$\Theta(b) = (\Theta_1(b), \cdots, \Theta_K(b))$$

where $\Theta_j(b) = sign(U_j)\min\{M||b^*||_2, S_\mu(|U_j|)\}$. It is shown that the pair (b^*, W^*) that minimizes the optimization problem, satisfies:

$$\begin{aligned} W_j^* =& \Theta_j(b^*) = sign(U_j)\min\{M||b^*||_2, S_\mu(|U_j|)\}. \\ b^* =& \text{argmin}_{b\in\mathbb{R}^k} F(b), \text{ where} \\ F(b) =& \frac{1}{2}\left(||v-b||_2^2 + ||U-\Theta(b)||_2^2\right) + \lambda||b||_2 + \mu||\Theta(b)||_1. \end{aligned}$$

Step 2: Re-arrange the index set of the coordinates $\{U_i\}$ as $\{U_{(i)}\}$ such that

$$0 = |U_{(K+1)}| \leq |U_{(K)}| \leq \cdots \leq |U_{(1)}| \leq |U_{(0)}| = \infty.$$

The intervals $\left[S_\mu(|U_{(j+1)}|), S_\mu(|U_{(j)}|)\right]$ cover the half-line $\mathbb{R}^{\geq 0}$, thus for any $b \in \mathbb{R}^k$ there is some $s \in [K+1] \cup \{0\}$ such that $M||b||_2 \in \left[S_\mu(|U_{(s+1)}|), S_\mu(|U_{(s)}|)\right]$.

This defines a composition map

$$\mathbb{R}^k \xrightarrow{\delta} \mathbb{R}^{\geq 0} \xrightarrow{\gamma} [K+1] \cup \{0\}, \text{ where } \delta(b) := M||b||_2,$$
$$\text{and } \gamma(w) = s \text{ such that } w \in \left[S_\mu(|U_{(s+1)}|), S_\mu(|U_{(s)}|)\right).$$

For any $b \in \mathbb{R}^k$, define $s_b = \gamma \circ \delta(b)$, it is shown that, one can write $F(b) = F_{s_b}(b) + r_{s_b}$ where r_{s_b} is independent of b (although it may depend on $s = \alpha(b)$ and hence indirectly on b) and

$$F_s(b) := \frac{1}{2}(1 + sM^2)\|b - \frac{1}{1+sM^2}v\|_2^2 + \left(\lambda - M\sum_{i=1}^{s}(|U_{(i)}| - \mu)\right)||b||_2.$$

Define a map

$$\beta : [K+1] \cup \{0\} \to \mathbb{R}^k, \text{ as } s \mapsto \text{argmin}_{b \in \mathbb{R}^k} F_s(b)$$

It can be shown using extremal theory that

$$\beta(s) = \frac{1}{1+sM^2}\left(1 - \frac{a_s}{||v||_2}\right)_+ v, \text{ where } a_s = \lambda - M\sum_{i=1}^{s}\left(|U_{(i)}| - \mu\right).$$

Step 3: There is a unique $s^* \in [K]$ such that $\alpha \circ \beta(s^*) = s^*$.
Step 4: The corresponding b_{s^*} is the global minimum of $F(b)$ thus solving the original minimization problem. One can recover the corresponding W^* using step 1.

4. Algorithms

Algorithm 1 Training LassoNet

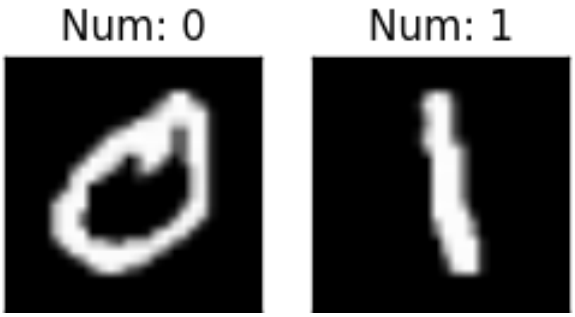
Input: training dataset $X \in \mathbb{R}^{n \times d}$, training labels Y, feed-forward neural network $g_W(\cdot)$, number of epochs B, hierarchy multiplier M, path multiplier ϵ, learning rate α
Initialize and train the feed-forward network on the loss $L(\theta, W)$
Initialize the penalty, $\lambda = \lambda_0$, and the number of active features, $k = d$
while $k > 0$ **do**
    Update $\lambda \leftarrow (1+\epsilon)\lambda$
    for $b \in \{1 \dots B\}$ **do**
        Compute gradient of the loss w.r.t to (θ, W) using back-propagation
        Update $\theta \leftarrow \theta - \alpha\nabla_\theta L$ and $W \leftarrow W - \alpha\nabla_W L$
        Update $(\theta, W^{(1)}) \leftarrow$ HIER-PROX$(\theta, W^{(1)}, \alpha\lambda, M)$
    end for
    Update k to be the number of non-zero coordinates of θ
end while
where HIER-PROX is defined in Alg. 2

Algorithm 2 Hierarchical Proximal Operator

procedure HIER-PROX($\theta, W^{(1)}; \lambda, M$)
  for $j \in \{1, \ldots, d\}$ **do**
    Sort the entries of $W_j^{(1)}$ into $|W_{(j,1)}^{(1)}| \geq \ldots \geq |W_{(j,K)}^{(1)}|$
    for $m \in \{0, \ldots, K\}$ **do**
      Compute $w_m := \frac{M}{1+mM^2} \cdot \mathcal{S}_\lambda\left(|\theta_j| + M \cdot \sum_{i=1}^{m} |W_{(j,i)}^{(1)}|\right)$
    end for
    Find $\tilde{m}$, the first $m \in \{0, \ldots, K\}$ such that $|W_{(j,m+1)}^{(1)}| \leq w_m \leq |W_{(j,m)}^{(1)}|$
    $\tilde{\theta}_j \leftarrow \frac{1}{M} \cdot \text{sign}(\theta_j) \cdot w_{\tilde{m}}$
    $\tilde{W}_j^{(1)} \leftarrow \text{sign}(W_j^{(1)}) \cdot \min(w_{\tilde{m}}, |W_j^{(1)}|)$
  end for
  return $(\tilde{\theta}, \tilde{W}^{(1)})$
end procedure
Notation: d denotes the number of features; K denotes the size of the first hidden layer.
Conventions: Ln. 6, $W_{(j,K+1)}^{(1)} = 0$, $W_{(j,0)}^{(1)} = +\infty$; Ln. 9, minimum and absolute value are applied coordinate-wise.

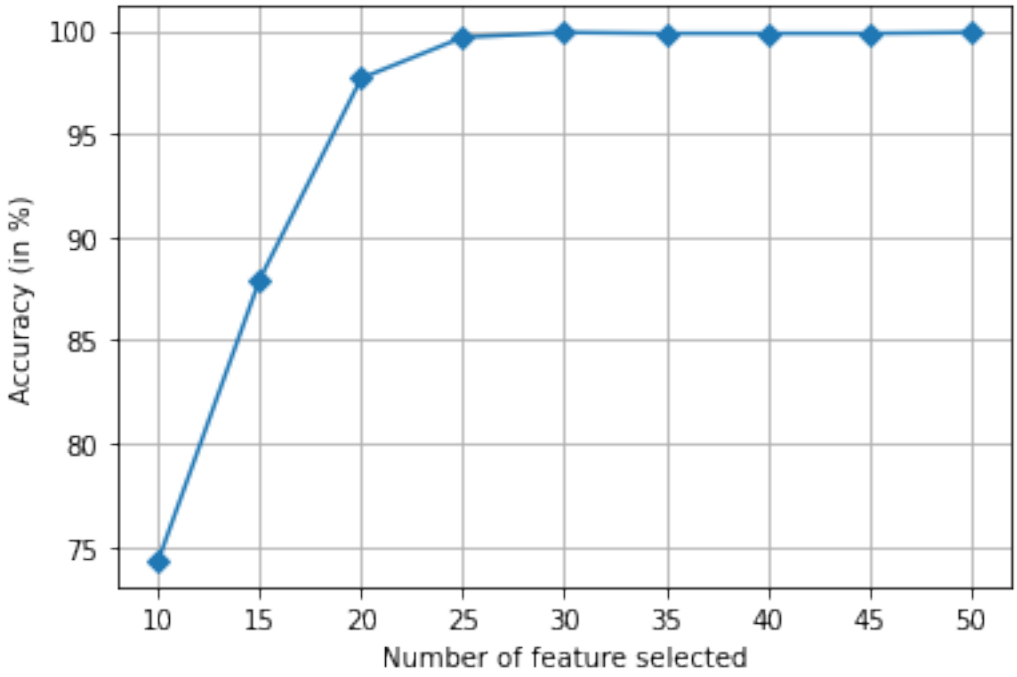
5. Experiment

We performed the experiment on the **MNIST** consist of 28-by-28 grayscale images of hand-written digits. We choose these datasets because they are widely known in the machine learning community. Although these are image datasets, the objects in each image are centered, which means we can meaningfully treat each 784 pixels in the image as a separate feature.
We used only **0** and **1** digit out of 10 digits.



Size of a train data set = (12665, 784)
Size of a test data set = (2115, 784)

6. Result



As we can see that our method is giving **99.669%** accuracy even when we have **25 features** out of 784, So we can say that the LassoNet is strong performer. The accuracy might fall if we consider all the 10 digit of MNIST dataset to train and test the model.