

The Minimax Probability Machine Regression (MPMR) Algorithm

1 Building the MPMR Model

We assume the following Training data:

$$\begin{pmatrix} x_{11} & x_{12} & \dots & x_{1d} & y_1 \\ x_{21} & x_{22} & \dots & x_{2d} & y_2 \\ \dots & \dots & \dots & \dots & \dots \\ x_{N1} & x_{N2} & \dots & x_{Nd} & y_N \end{pmatrix} \quad (1)$$

This data is assumed to be generated randomly from an unknown and possibly noisy function:

$$y_i = f(x_{i1}, \dots, x_{id}) + \rho \quad (2)$$

for $i = 1, \dots, N$ and where ρ is a random variable with mean 0 and finite variance. For regression problems we assume $y \in \mathfrak{R}$ and for classification problems we assume a binary problem where $y \in \{-1, +1\}$.

The MPMR Learning Algorithm learns an approximation $\hat{y} = \hat{f}(x_1, \dots, x_d)$ from the training data. This approximation is constructed by projecting the training data inputs into a Kernel function space. Letting $\mathbf{u}, \mathbf{v} \in \mathfrak{R}^d$, three commonly used Kernel functions are:

1. Linear Kernel: $K(\mathbf{u}, \mathbf{v}) = \mathbf{u}\mathbf{v}^T$
2. Polynomial Kernel: $K(\mathbf{u}, \mathbf{v}) = (1 + \mathbf{u}\mathbf{v}^T)^p$, where $p = 1, 2, \dots$
3. Gaussian Kernel: $K(\mathbf{u}, \mathbf{v}) = \exp\left(\frac{-(\mathbf{u}-\mathbf{v})(\mathbf{u}-\mathbf{v})^T}{2\sigma^2}\right)$, where $\sigma > 0$.

The model (i.e. hypothesis) constructed by MPMR has the following form:

$$\hat{y} = \hat{f}(\mathbf{x}) = \sum_{i=1}^N \beta_i K(\mathbf{x}_i, \mathbf{x}) + b \quad (3)$$

where $\mathbf{x} = (x_1, \dots, x_d)$ is some new input vector, \hat{y} is the approximated output, $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$ is training instance i , and the coefficients β_1, \dots, β_N and offset b are derived as follows.

First, choosing one of the above Kernel functions, we project the training data inputs into kernel function space by defining the variable z_{ij} as:

$$z_{ij} := K(\mathbf{x}_i, \mathbf{x}_j) \quad (4)$$

where $\mathbf{x}_i = (x_{i1}, \dots, x_{id})$ is training instance i and $\mathbf{x}_j = (x_{j1}, \dots, x_{jd})$ is training instance j , for all $i, j \in \{1, \dots, N\}$. Computing $K(\mathbf{x}_i, \mathbf{x}_j)$ for all N^2 pairs gives us the Gram Matrix:

$$\begin{pmatrix} z_{11} & \dots & z_{1N} \\ \vdots & \ddots & \vdots \\ z_{N1} & \dots & z_{NN} \end{pmatrix}$$

For the MPMR algorithm we interpret the columns $z_{\cdot j}$ of this matrix as random variables and calculate the covariances between them as well as the covariances to the output column y of the training data:

$$\sigma_{z_{\cdot i} z_{\cdot j}} := \frac{1}{N-1} \sum_{k=1}^N (z_{ki} - \bar{z}_{\cdot i})(z_{kj} - \bar{z}_{\cdot j}) \quad (5)$$

where

$$\bar{z}_{\cdot i} := \frac{1}{N} \sum_{k=1}^N z_{ki} \quad (6)$$

The same formula is used to calculate the covariances $\sigma_{y z_{\cdot i}}$ for $i = 1, \dots, N$:

$$\sigma_{y z_{\cdot j}} := \frac{1}{N-1} \sum_{k=1}^N (y_k - \bar{y})(z_{kj} - \bar{z}_{\cdot j}) \quad (7)$$

where

$$\bar{y} := \frac{1}{N} \sum_{k=1}^N y_k \quad (8)$$

Given these covariances, we solve the following linear system to obtain the coefficients β_1, \dots, β_N :

$$\begin{pmatrix} \sigma_{z_{\cdot 1} z_{\cdot 1}} & \sigma_{z_{\cdot 1} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot 1} z_{\cdot N}} \\ \sigma_{z_{\cdot 2} z_{\cdot 1}} & \sigma_{z_{\cdot 2} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot 2} z_{\cdot N}} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{z_{\cdot N} z_{\cdot 1}} & \sigma_{z_{\cdot N} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot N} z_{\cdot N}} \end{pmatrix} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \cdots \\ \beta_N \end{pmatrix} = \begin{pmatrix} \sigma_{y z_{\cdot 1}} \\ \sigma_{y z_{\cdot 2}} \\ \cdots \\ \sigma_{y z_{\cdot N}} \end{pmatrix}$$

We can solve for β_1, \dots, β_N :

$$\begin{pmatrix} \beta_1 \\ \beta_2 \\ \cdots \\ \beta_N \end{pmatrix} = \begin{pmatrix} \sigma_{z_{\cdot 1} z_{\cdot 1}} & \sigma_{z_{\cdot 1} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot 1} z_{\cdot N}} \\ \sigma_{z_{\cdot 2} z_{\cdot 1}} & \sigma_{z_{\cdot 2} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot 2} z_{\cdot N}} \\ \cdots & \cdots & \cdots & \cdots \\ \sigma_{z_{\cdot N} z_{\cdot 1}} & \sigma_{z_{\cdot N} z_{\cdot 2}} & \cdots & \sigma_{z_{\cdot N} z_{\cdot N}} \end{pmatrix}^{-1} \begin{pmatrix} \sigma_{y z_{\cdot 1}} \\ \sigma_{y z_{\cdot 2}} \\ \cdots \\ \sigma_{y z_{\cdot N}} \end{pmatrix}$$

In practice this linear system has an infinite number of solutions, hence methods like singular value decomposition are required to solve the system.

The β_i 's are the coefficients of an N -dimensional hyperplane, the offset can be calculated as: $b = \bar{y} - \sum_{i=1}^N \beta_i \bar{z}_{\cdot i}$

2 How Good is the MPMR Model?

The MPMR algorithm also computes a lower bound on the probability Ω that the predicted output will be within a $\pm\epsilon$ margin accurate:

$$\Omega = \frac{1}{4\mathbf{a}^T \Sigma \mathbf{a} + 1} \quad (9)$$

where

$$\mathbf{a} = \frac{1}{2\epsilon} \begin{pmatrix} 1 \\ -\beta_1 \\ -\beta_2 \\ \dots \\ -\beta_N \end{pmatrix} \quad (10)$$

and

$$\Sigma = \begin{pmatrix} \sigma_{yy} & \sigma_{yz_1} & \dots & \sigma_{yz_N} \\ \sigma_{z_1y} & \sigma_{z_1z_1} & \dots & \sigma_{z_1z_N} \\ \dots & \dots & \dots & \dots \\ \sigma_{z_Ny} & \sigma_{z_Nz_1} & \dots & \sigma_{z_Nz_N} \end{pmatrix} \quad (11)$$

By doing some transformations we can obtain an equivalent formula for Ω :

$$\Omega = \frac{1}{(\sigma_{yy} - \sum_{i=1}^N \beta_i \sigma_{yz_i})/\epsilon^2 + 1} \quad (12)$$

Computing the confidence in our model and our estimate of Ω (i.e. detecting overfitting of the training data) can be done in the following way:

Given a fraction α of rows to be taken out of the Gram matrix and the output vector, we compute the new covariance matrix Σ_α (which is still of size $(N + 1)^2$). From the new Σ_α we calculate β_α and a_α . The confidence in our model and in our Ω estimate is then defined as the ratio:

$$R = \begin{cases} \frac{\alpha}{\mathbf{a}_\alpha^T \Sigma_\alpha \mathbf{a}_\alpha}, & \text{iff } \mathbf{a}_\alpha^T \Sigma_\alpha \mathbf{a}_\alpha < \alpha \\ \frac{\alpha \mathbf{a}_\alpha^T \Sigma_\alpha \mathbf{a}_\alpha}{\mathbf{a}_\alpha^T \Sigma_\alpha \mathbf{a}_\alpha - \mathbf{a}^T \Sigma \mathbf{a}}, & \text{otherwise} \end{cases} \quad (13)$$

If $R \geq 1$ we are confident in our model. If $R < 1$ we reject it.