** How do you relate SVM with logistic regression?

They can be related by error function given by

$$E_{SV}(y_n t_n) = (1 - y_n t_n)$$

It can be viewed as an approximation as misclassification errors in SVM.

Then in logistic regression with $p[t = \frac{1}{y}] = \sigma(y)$

$$p[t/y] = \sigma(yt)$$

$$\sum_{n=1}^{N} E_{LR}(y_n t_n) + \lambda \|w\|^2$$

***.
V.V. (20 equations) Explain how SVM can be extended to regression problem.

(or)

Write down the steps involved in converting general SVM to be applied for regression problems.

By preserving the property of one can extend SVM to regression problems.

Step 1 . Introduction of ϵ-intensive error function having a linear cost associated with error cost outside the intensive region.

$$E_\epsilon[y(x) - t] = \begin{cases} 0 & |y(x) - t| < \epsilon \\ |y(x) - t| - \epsilon & \text{otherwise} \end{cases} \quad \text{--- (1)}$$

This is called minimizing the regularization error function.

$$E_r = \frac{1}{2}\sum_{n=1}^{N}(y_n - t_n)^2 + \frac{\lambda}{2}\|w\|^2 \quad \text{--- (2)}$$

Step 2 Reexpressing the optimization function as minimum error of

$$\text{Min } E(w, t) = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N} E_\epsilon(y(x_n) - t_n) \quad \text{--- (3)}$$

subjected to $t_n \leq y(x_n) + \epsilon + \xi_n$ — (4)

$t_n \geq y(x_n) - \epsilon - \hat{\xi}_n$ — (5)

when ξ_n and $\hat{\xi}_n$ are ~~fed~~ slack variable $t_n > y(x_n) + \epsilon$ and $t_n \leq y(x_n) - \epsilon$ are converted to (4), (5)

step3 Modification of error function in view of eq (3) (4) & (5)

$$E_R[w, \xi_n, \hat{\xi}_n] = \frac{1}{2}\|w\|^2 + C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n)$$

where (6) to be minimized subject to. $\xi_n, \hat{\xi}_n > 0$ and eq (4) (5)

step4. & step5. simplification of constraints and setting the derivatives to 0

substitute to $y(x)$ in $w^T\phi(x) + b$ — (*)

$$L[w, b, \xi_n, \hat{\xi}_n] = C\sum_{n=1}^{N}(\xi_n + \hat{\xi}_n) + \frac{1}{2}\|w\|^2 - \sum_{n=1}^{N}(\mu_n\xi_n + \hat{\mu}_n\hat{\xi}_n)$$

$$- \sum_{n=1}^{N} a_n(\epsilon + \xi_n + y_n - t_n) - \sum_{n=1}^{N}\hat{a}_n(\epsilon + \hat{\xi}_n + t_n - y_n)$$ — (7)

$$\frac{\partial L}{\partial w} = 0 \Rightarrow \sum_{n=1}^{N}(a_n - \hat{a}_n)\phi(x_n)$$ — (8)

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N}(a_n - \hat{a}_n) = 0$$ — (9)

$$\frac{\partial L}{\partial \xi_n} = 0 \Rightarrow a_n + \mu_n = C$$ — (10)

$$\frac{\partial L}{\partial \hat{\xi}_n} = 0 \Rightarrow \hat{a}_n + \hat{\mu}_n = C$$ — (11)

Construction of Dual :-

step 6 · because elimination of corresponding values from lagranges

$$\text{maximize} \quad \tilde{L}(a,\hat{a}) = -\frac{1}{2} \sum_{n=1}^{N} \sum_{n=1}^{N} (a_n - \hat{a}_n)(a_m - \hat{a}_m) K(x_n, x_m)$$

$$- \epsilon \sum_{n=1}^{N} (a_n + \hat{a}_n) + \sum_{n=1}^{N} (a_n - \hat{a}_n) t_n \quad \text{—(12)}$$

$$\text{where} \quad K(x, x') = \phi(x)^T \phi(x')$$

This is constraint maximization problem.

$a_n, \hat{a}_n, \mu_n, \hat{\mu}_n \geq 0$ with eq (10) & (11) yields step (7)

~~Step. simplification to programmable form of dual.~~

step 7. construction of box constraint

$$0 \leq a_n \leq C \quad (13)$$

$$0 \leq \hat{a}_n \leq C \quad (14)$$

step 8. simplification to programmable form of dual.

substitute (8) in 7.

$$y(x) = \sum_{n=1}^{N} (a_n - \hat{a}_n) K(x, x_n) + b \quad \text{—(15)}$$

Kuhn-tucker conditions yield.

$$a_n [\epsilon + \xi_n + y_n - t_n] = 0 \quad \text{—(16)}$$

$$\hat{a}_n [\epsilon + \hat{\xi}_n - y_n + t_n] = 0 \quad \text{—(17)}$$

$$(C - a_n) \xi_n = 0 \quad \text{—(18)}$$

$$(C - \hat{a}_n) \hat{\xi}_n = 0 \quad \text{—(19)}$$

where $b = t_n - \epsilon - \sum_{1}^{N} (a_m - \hat{a}_m) K(x_n, x_m)$ —(20)

Q) consider a two input, one output feed forward neural network. Weights from i/p to hidden layer are given by w_{ik}.

$$w_{ik} = \{0.1, 0.4, -0.2, 0.2\}$$

weights from hidden to o/p layer given by

$$W_{kj} = \{0.2, -0.5\}$$

with no bias and sigmoid activation function with slope 0.18.
find the o/p of the neural network for the i/p.

$$I = [(0.4, -0.7)], \text{ (output is 0.1)}$$

or $[I : O] = [\{(0.4, -0.7)\}, 0.1]$

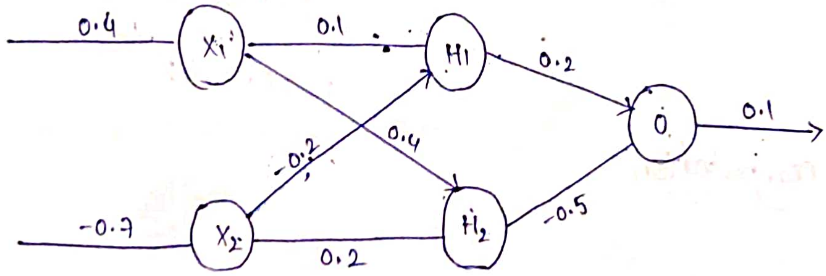
sol Given that training data [Input output] = $[\{(0.4, -0.7)\}, 0.1]$

weight from input to hidden nodes = $w_{11} = 0.1$, $w_{12} = -0.2$, $w_{21} = -0.2$,

$w_{22} = 0.2$,

weight from hidden to output nodes = $w_{11} = V_1 = 0.2$, $w_{22} = V_2 = -0.5$

Architecture.



I. forward computation.

step1 Already we have $-1 < x_1, x_2 < 1$ $\therefore$ No need to normalize

step2 * Input to hidden nodes : $(x_1, x_2)^T = (0.4, -0.7)^T$

step3

$$IH_1 = w_{11} x_1 + w_{21} x_2 = 0.1(0.4) + (-0.2)(-0.7) = 0.18$$

$$IH_2 = w_{12} x_1 + w_{22} x_2 = 0.4(0.4) + (-0.7)(0.2) = 0.02$$

(OR)

$$\begin{bmatrix} IH_1 \\ IH_2 \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} \\ w_{21} & w_{22} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

$$= \begin{bmatrix} 0.1 & -0.2 \\ 0.4 & 0.2 \end{bmatrix} \begin{pmatrix} 0.4 \\ -0.7 \end{pmatrix} = \begin{pmatrix} 0.18 \\ 0.02 \end{pmatrix}$$

<u>Step 4.</u> $OH_1 = (1+e^{-IH_1})^{-1} = (1+e^{-0.18})^{-1} = 0.5448.$

$OH_2 = (1+e^{-IH_2})^{-1} = (1+e^{-0.02})^{-1} = 0.505$

(OR)

$$\begin{bmatrix} OH_1 \\ OH_2 \end{bmatrix} = \begin{bmatrix} (1+e^{-IH_1})^{-1} \\ (1+e^{-IH_2})^{-1} \end{bmatrix} = \begin{bmatrix} (1+e^{-0.18})^{-1} \\ (1+e^{-0.02})^{-1} \end{bmatrix} = \begin{bmatrix} 0.5448 \\ 0.505 \end{bmatrix}$$

* Hidden to output nodes :

<u>Step 5.</u>

$(OH_1, OH_2)^T = (0.5448 \quad 0.505)^T$

Given that $(v_1 \; v_2) = (0.2, -0.5) = V$

$$I^Y = V(OH)^T = (0.2, -0.5)_{r\times c} \begin{bmatrix} 0.5448 \\ 0.505 \end{bmatrix}_{2\times 1} = -0.14354$$

$$OY = (1+e^{-IY})^{-1} = (1-0.14354)^{-1}$$

$$= 0.4642$$

II. Error computation.

$$\text{Errors} = (T_0 - O_0)^2$$

$$= (0.1 - 0.4642)^2$$

$$= 0.13264$$

<u>Error output layer :</u>

$$\delta = (0 - OY)(1 - OY)\, OY$$

$$\left[y'(x) = \frac{-1}{(1+e^{-x})^2} (-e^{-x}) = \frac{e^{-y}}{(1+e^{-x})^2} \right.$$

$$\frac{1}{(1+e^{-x})}\left[1-\frac{1}{1+e^{-x}}\right] = \frac{(1+e^{-x})-1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2}$$

$$= y(x)\,[1-y(x)]$$ Mol sum

$$\delta = (O - O_Y)(1-O_Y)O_Y$$

$$= (0.1-0.4642)(1-0.4642)(0.4642)$$

$$\delta = -0.09058$$

$$\Delta U_1 = \delta\, OH_1 = -0.09058(0.5448) = -0.0493$$

$$\Delta U_2 = \delta\, OH_2 = -0.09058\,(0.505) = -0.0457$$

Assuming learning rate $\boxed{\eta = 0.6}$

$$\Delta U_1 = -0.0493 \times 0.6 = -0.02958$$

$$\Delta U_2 = -0.0457 \times 0.6 = -0.02742$$

III. Back propagation.

* New weights for output layers.

$$U_1 + \Delta U_1 \rightarrow U_1 = 0.2 - 0.02958 = 0.17042$$

$$U_2 + \Delta U_2 \rightarrow U_2 = -0.5 - 0.02742 = -0.52742$$

* Errors for hidden layers

$$\delta_1 = \delta \times U_1 \times \phi'(OH_1) = \delta \times U_1 \times OH_1(1-OH_1)$$

$$= (-0.09058)(0.2)(0.5448)(1-0.5448)$$

$$\delta_1 = -0.00449$$

$$\delta_2 = \delta \times U_2 \times \phi'(OH_2) = \delta \times U_2 \times OH_2(1-OH_2)$$

$$= (-0.09058)(-0.5)(0.505)(1-0.505)$$

$$\delta_2 = 0.01132$$

* Increment computation for weights (Input to hidden layer)

$\Delta w_{11} = \delta_1 \times x_1 = -0.00449 \times 0.4 = -0.001796$

$\Delta w_{12} = \delta_2 \times x_1 = 0.01132 \times 0.4 = 0.004528$

$\Delta w_{21} = \delta_1 \times x_2 = -0.00449 \times (-0.7) = 0.003143$

$\Delta w_{22} = \delta_2 \times x_2 = 0.01132 \times (-0.7) = -0.007924$

* New hidden layer weight updation.

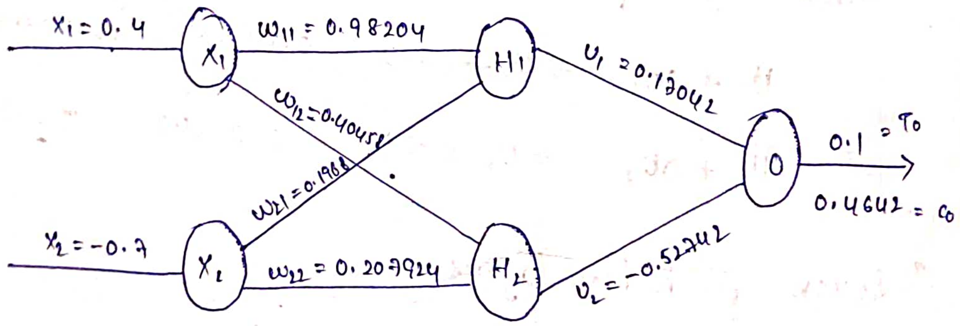
$w_{11} = w_{11} + \Delta w_{11} = 0.1 + (-0.001796) = 0.098204$

$w_{12} = w_{12} + \Delta w_{12} = 0.4 + (0.004528) = 0.404528$

$w_{21} = w_{21} + \Delta w_{21} = -0.2 + (0.003143) = 0.19686$

$w_{22} = w_{22} + \Delta w_{22} = 0.2 + (-0.007924) = 0.207924$

Revised architecture after I Iteration :



16/05/22. Q. Explain how relevance vector machine can be built for solving regression problems with modified prior resulting in sparse solutions

Step 1. RVM model defines a condition distribution for a real variable target variable t, given an input variable x.

$p[t/x, w, \beta]$

where $\beta = \frac{1}{\sigma^2}$ being the noise precision and mean given in the linear form as

$$y(x) = \sum_{i=1}^{N} w_i \phi_i(x) = w^T \phi(x) \quad \text{--- (2)}$$

It is a non linear bias function $\phi_i(x)$ including a constant term as bias of eq (2)

2) General expression of (1) takes the SVM like form

$$y(x) = \sum_{n=1}^{N} w_n K(x, x_n) + b \quad \text{--- (3)}$$

where b is bias term

Here no. of parameters is $M = N+1$

unlike SVM.

a) There is no restriction in +ve definite kernel.

b) The basis functions are not tied in either in no. or variance to their training data point.

Q. Distinguish RVM for regression from SVM for regression ⇒ a, b answer
(2M)

Step-2. learning choices of α and β

3) let there be n observations of input vector x.

The data matrix X have n^{th} row as x_n^T; $n = 1$ to N

Target values $t = (t_1, t_2, \ldots\ldots t_n)^T$

4) Then likelihood function is given by

$$P[t \mid X, w, \beta] = \prod_{n=1}^{N} P[t_n \mid x_n, w, \beta^{-1}] \quad \text{--- (4)}$$

Introducing seperate hyperparameter α_i for each of the weight parameter w_i (instead of single shared hyper parameter as in SVM)

The weight prior is given by

$$P[w|\alpha] = \prod_{n=1}^{M} N[w_i | 0, \alpha_i^{-1}] \quad —⑤$$

where α is perception of w_i and $\alpha = (\alpha_1, \alpha_2, \cdots \alpha_M)^T$ —⑥

using linear regression models, posterior distribution for weights is again guassian and is given by

$$P[w|t, x, \alpha, \beta] = N[w|m, \Sigma] \quad —⑦$$

step 3.

4) Evaluating mean and co-variance of the posterior. where mean and covariance are given by

$$m = \beta \Sigma \phi^T t \quad —⑧$$

$$\Sigma = (A + \beta \cdot \phi^T \phi)^{-1} \quad —⑨$$

where ϕ is $N \times M$ matrix, with $\phi_{ni} = \phi_i(x_n)$

$A = \mathrm{diag}(\alpha_i)$

and k is eq ③ is the symmetric $(N+1)(M+1)$ Kernel matrix with elements from $k(x_n, x_m)$

The values of α and β in eq ⑦ are determited using type 2 maximum likelihood (cenknown as evident subtraction)

Thus $p[t|x,\alpha,\beta] = \int p(t|x,\omega,\beta)\, p(\omega|\alpha)\, d\omega$.— (10)

It is type 2 maximum likelihood that maximizes the marginal likelihood function obtained by integrating weight parameters.

As eq (10) is convolution of 2 guassian, it can be evaluated to log likelihood in the form.

$$\log p[t|x,\alpha,\beta] = \log N[t| \quad ,c]$$

$$= -\frac{1}{2}\left[N\log 2\pi + \log |C| + t^T C^{-1} t\right] \quad — (11)$$

where $t = (t_1, t_2 \cdots t_N)^T$ and $C = \beta^{-1} I + \phi A^{-1} \phi^T$ — (12)

For maximizing eq (11) setting a derivatives of marginal likelihood to zero yields point 5 step 4

Step 4. Reestimating of hyper parameters

5)
$$\alpha_i^{new} = \frac{\gamma_i}{m_i^2} \quad — (13)$$

$$(\beta^{new})^{-1} = \|t - \phi m\|$$

where m_i is i^{th} component of posterior mean 'm' as in eq (8)

γ_i – a measure corresponding to ω_i determined by the data.

defined as $\gamma_i = 1 - \alpha_i \Sigma_{ii}$

where Σ_{ii} is the diagonal component of the posterior covariance given by eq (9) step 5 · Reestimating Mean and covariance.

6) until suitable convergence criterion is satisfied.

Q. What do you mean relavant vectors in RVM ?

sol In relavant vector machine in general expression :

$$y(x) = \sum_{n=1}^{N} w_n k(x, x_n) + b$$

The input $x(n)$ corresponding to remain in non zero weights are called relavant vectors they are identified through mechanism of Automatic relevant determination.

Q. What is alternate procedure for improving training speed of RVM

sol ~~positive~~ Sparsity mechanism.

Q. Explain the mathematical analysis of mechanism of sparsity in ~~relevance~~ of context of Relevance Vector Machnie

Q. Write down sequential sparse bayesian learning algorithm

Q. Build a relevant vector machine for classification problem.