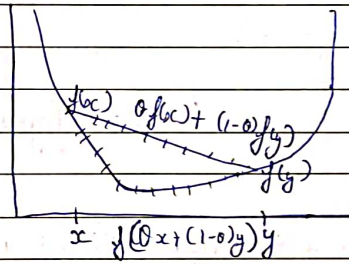
Date 25-01-2022
Page No.

PRML

- **Convex Sets** – $C \subseteq R^n$ is convex, if $\theta x + (1-\theta)y \in C$ for any $x, y \in C$ and $0 \leq \theta \leq 1$.

  i.e. a set is convex if the line connecting any 2 points in the set is entirely inside the set

- **Convex Functions** – $f: R^n \to R$ is convex if $dom(f)$ (the domain of f) is a convex set and if $f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y)$ for any $x, y \in dom(f)$ and $0 \leq \theta \leq 1$



- finding minima of convex function is easy.
- If $f_1(x)$ is convex and f_2 is convex –

  then – (i) $f_1(x) + f_2(x)$ is also convex

  (ii) $\alpha f_1 x$ is also convex

  (iii) $\max(f_1(x) + f_2(x))$ is also convex

- If a function is convex function and it is also smooth, then it is differentiable through out the domain.
- convexity means when it will decrease it will decrease only and when it will increase it will increase only.

- there is a point where rate of change in every direction is zero (i.e. gradient), that will give the minimum value
- If function → convex & smooth, then necessary and sufficient condition for optimal solution x_0 is $\nabla f(x) = 0$ at $x = x_0$

e.g. if function is from $R^2 \to R$, then the gradient will also be in $R^2 \to \begin{bmatrix} a \\ b \end{bmatrix}$ → gradient vector, direction of gradient is always direction of ascent (moving up)

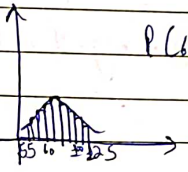
* $y_i = \underbrace{E(Y/x_i)}_{\hookrightarrow f(x_i)} + \epsilon_i$, $E(\epsilon_i) = 0$

↳ Mean Regression Model

* $y_i = \underbrace{f(x_i)}_{\hookrightarrow \text{if it tells, suppose lowest 10\% of values of } y_i \text{ for given } x, \text{ then this type of model is Quantile Regression Model [it gives more information than mean regression model]}} + \epsilon_i$

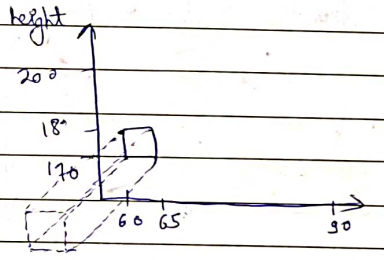
* while plotting histogram if we reduce the size of bin then we can join the points and make a curve and then find area of curve of the region we want to evaluate.
- when bin is large, much data is lost because of approximation.

$P(60 \le x \le 120) = \int_{60}^{120} f(x)\,dx$ → telling what is the density around certain point



[1- dimension]

- for 2- dimension –

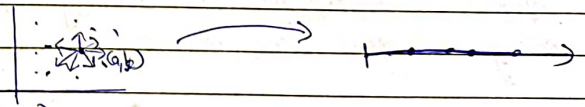


bin will be of pair $[(60,65), (170-180)]$

$f(x,y) =$ ------

- gradient comes when we are dealing with 2 or more variables otherwise for 1 variable we will restrict to the concept of differentiation.

e.g. $f: R^2 \to R$, $f(x,y) = x^2 + y^2$



$\vec{u} \cdot \nabla f(a,b)$ (X)

$\arg\max_u \left(\underbrace{\nabla f(x)^T u}_{\hookrightarrow \text{rate of change in } (u) \text{ direction}} \right)$ [where $u = \nabla f(x) / \|\nabla f(x)\|$] where 'u' is unit vector

$\nabla f(x)^T \nabla f(x) = \|\nabla f(x)\|^2$

$\frac{\nabla f(x)^T \nabla f(x)}{\|\nabla f(x)\|} = \frac{\|\nabla f(x)\|^2}{\|\nabla f(x)\|} = \|\nabla f(x)\|$

Max. rate of change we can attain is the norm of the gradient of $f(x)$

$u = \|\nabla f(x)\|$ → this is direction of maximum ascent

$u = -\|\nabla f(x)\|$ → direction of maximum descent

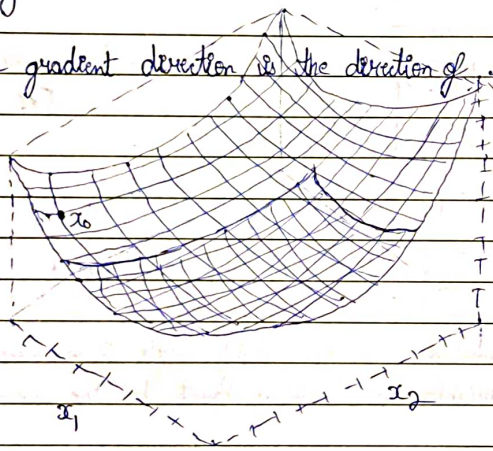
* Gradient descent algorithm – [an Iterative algorithm]

Algo ⇒ Initialize $x^0 = x^{start} \in R^n$

repeat $x^{(k+1)} := x^{(k)} - \gamma_k \nabla f(x^{(k)})$

Until $\|\nabla f(x^{(k)})\| \leq \varepsilon$

The negative gradient direction is the direction of steepest descent.



and gradient is laplace continuous

- If our function is convex, then with this algo we are guaranteed to converge on global local minima

$f: R^n \to R$, $\|\nabla f(x) - \nabla f(y)\|_2 \leq \mathcal{L} \|x - y\|_2$ → L2 - norm

$\mathcal{L} \geq 0$

After kth step,

$$(f(x^k) - f(x^*)) \leq \frac{\|(x)^{(0)} - x^*\|_2^2}{2tk}$$ → L2 norm

where, t is gamma (γ) (step length)

as we increase the value of k, $f(x^{(k)})$ converges to $f(x^*)$

$f(x^{(k)}) \to f(x^*)$

here, x^k → at kth step

x^* → optimal solution (desired)

x^0 → initial position

* Loss function – It is perception that what do we mean by error there are many types of loss function ($\mathcal{L}(f(x_i) - Y_i)$)

e.g.

(i) $(f(x_i) - Y_i)^2$

(ii) $|f(x_i) - Y_i|$

* Emperical Risk Minimization – $\frac{1}{\mathcal{L}} \sum_{i=1}^{\mathcal{L}} \mathcal{L}(Y_i - f(x_i))$ → f''

* Structured Risk Minimization – $\int \mathcal{L}(Y_i - f(x_i)) \, dP(x,y)$ → f'

as, $\mathcal{L} \to \infty$ then $f'' \to f'$

* F.f can be anything, lets start with F.f as a linear function

where $x \in R^n$, $Y \in R$

$= w_1 x_1 + w_2 x_2 + w_n x_n + b$

$= w^T x + b$

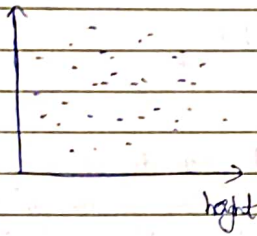
$F = \{ w^T x + b : x, w \in R^n, b \in R \}$

$L = (Y - f(x))^2 \Rightarrow u^2$ → least square loss function, it is smooth and convex, so differentiable at every given point

- Minimization of least square loss function –

$$\min_{w,b} \frac{1}{2} \sum_{i=1}^{2} (y_i - (w^T x_i + b))^2$$

*



$X = (x, y)$ → height, width

$T = \{(x_1, y_1), (x_2, y_2), \ldots\ldots (x_n, y_n)\}$

Unbiased Estimate of mean $= \bar{X} = (\bar{x}, \bar{y}) = \frac{1}{n} \sum_{i=1}^{n} (x_i, y_i)$ [in 2-Dimension]

Unbiased Estimate of variance $= \frac{1}{n} \sum_{i=1}^{2} (\bar{x} - x_i)^2$ [in 1-Dimension]

If we have large no of data points, then this will converge to variance

In 2-D, we have co-variance

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{bmatrix}$$

(σ_x^2: variance along x-axis; σ_y^2: variance along y-axis; σ_{xy}: co-variance; matrix → Co-variance matrix (Σ))

(Symmetric & positive semi-definite matrix)

$$Cov(x, y) = \frac{1}{n-1} \sum_{i=1}^{2} (x_i - \bar{x})(y_i - \bar{y})$$

Variance along a particular direction can be obtained by – $\hat{a}^T \Sigma \hat{a}$

*/ In 2-D, if we want to know that our data follows Normal Distribution or not, then we need to project it to a axis (say 'w') and then on that axis we will check if 68.3% of data values are falling under $(u-\sigma)$ to $(u+\sigma)$ and 95.4% are falling under $(u-2\sigma)$ to $(u+2\sigma)$ and 99.7% are falling under $(u-3\sigma)$ to $(u+3\sigma)$ then it is normal distribution

* $X = (x, y)$, if, $X \sim N(u, \Sigma)$ [Density function, if x & y are independent]
then $w^T X \sim N(w^T u, w^T \Sigma w)$

then we can say x & y are jointly normal

$X = (x, y)$

$X = (x_1, x_2)$

$f(x) = f(x_1) f(x_2)$

$$= \frac{1}{\sqrt{2\pi}\sigma_1} \exp^{-\frac{1}{2}\left(\frac{x-u_1}{\sigma_1^2}\right)^2} * \frac{1}{\sqrt{2\pi}\sigma_2} \exp^{-\frac{1}{2}\left(\frac{x-u_2}{\sigma_2^2}\right)^2}$$

$$f(x) = \frac{1}{(\sqrt{2\pi})^2 |\Sigma|^{\frac{1}{2}}} \exp^{-\frac{1}{2}(x-u)^T \Sigma^{-1} (x-u)}, \quad X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix}$$

↓ density function of gaussian or normal distribution in n-dimension

[since x & y are independent, covariance of them is zero]

- Contour points are the points where value of $f(x_1, x_2)$ is same, i.e. $f(x_1, x_2) = c$

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}, \quad A = \begin{bmatrix} x_{11}, x_{12}, \dots, x_{1m}, 1 \\ x_{21}, \dots, 1 \\ \vdots \\ x_{n1}, \dots, x_{nm}, 1 \end{bmatrix}, \quad w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ b \end{bmatrix}$$

$$\min_{w} \underbrace{(Y - Aw)^T (Y - Aw)}_{h(w)}$$

$$\nabla_w h(w) = 0$$

* **Least squares linear regression problem**

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ b \end{bmatrix} = (A^T A)^{-1} A^T Y$$

* **Least square quadratic regression model** — Refer screenshot & assignment

$$\min_{(w)} f(w) = \min \left((Y - Aw)^T (Y - Aw) \right)$$

$$\nabla_w (2(Y - Aw))$$

$$2A^T (Y - Aw) = 0$$

$$A^T Y - A^T A w = 0$$

$$A^T A w = A^T Y$$

$$w = (A^T A)^{-1} A^T Y$$

* **Polynomial basis functions** — [Non-linear estimate]

$$M=2 \left\{ 1, x_1, x_2, x_3, x_1 x_2, x_1 x_3, x_2 x_3, x_1^2, x_2^2, x_3^2 \right\}$$

Basis functions $\{\phi_m(x)\}_{m=1}^{M}$

$$\phi_1(x) = 1, \quad \phi_2(x) = x_1, \quad \phi_3(x) = x_2, \quad \dots \quad \phi_6(x) = x_1 x_3, \dots$$

Independent of choice of basis functions, the regression parameters were calculated using the well-known equations for linear regression

$$\sum_{i=1}^{M\phi} w_i \phi_i(x)$$

$$f(x) = \sum_{i=1}^{M-1} w_i \phi_i(x) + b \Rightarrow w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x)$$

'M' is the total no. of parameters in the model

$$\min_{w_0, w_1, \dots w_m} \sum_{i=1} \left(Y_i - \left(w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) \right) \right)^2$$

$$\min_{w} \sum_{i=1} (Y_i - w^T \phi(x_i))^2$$

$$\min_{w} (Y - w^T \phi)^T (Y - w^T \phi)$$

$$w = (\phi^T \phi)^{-1} \phi^T Y$$

where, $\phi = \begin{pmatrix} \phi_1(x_1) & \dots & \phi_{M\phi}(x_1) \\ \phi_1(x_2) & \dots & \phi_{M\phi}(x_2) \end{pmatrix}$

* **Linear**

- $f(x) = w^T x$
- $w = (x^T x)^{-1} x^T Y$
- $x = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} & 1 \\ \vdots & & & & \vdots \\ x_{t1} & x_{t2} & \dots & x_{tn} & 1 \end{bmatrix}$

Non-linear

$$f(x) = w^T \phi(x)$$

$$w = ((\phi^T \phi)^{-1} \phi^T Y)$$

$$\phi = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & \dots & \phi_m(x_1) \\ \vdots & & & \vdots \\ \phi_1(x_t) & \phi_2(x_t) & \dots & \phi_m(x_t) \end{bmatrix}$$

* Maximum likelihood estimate — It estimates the parameter (θ) of a distribution, suppose we have normal distribution then our parameter (θ) to be estimated will be (μ, σ)

• Sample — $\{x_1, x_2, \ldots x_n\}$

$$P(\theta/T) = \frac{P(T/\theta)\,P(\theta)}{P(T)}$$

$\text{Max } P(\theta/T) = \text{Max } P(T/\theta)$ [because we don't have info about $P(\theta)$ & $P(T)$]

$$= P(x_1, x_2, \ldots x_n/\theta)$$
$$= \prod_{i=1}^{n} P(x_i/\theta)$$

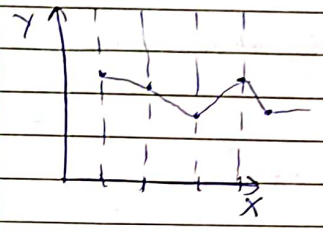
Let us suppose θ is coming from normal distribution

$$\prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

$$\text{Max } \log P(\theta/T) = \text{Max } \log \prod_{i=1}^{n} \exp^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Now, we can take derivative with respect to μ & σ to maximize μ & σ

after maximizing, $\mu = \frac{\sum_{i=1}^{n} x_i}{n}$, $\sigma = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n-1}$.



$$Y = f(x) + \epsilon \quad \hookrightarrow E(\epsilon) = 0$$
$$Y \sim N(f(x), \beta)$$
$$Y/X \sim N(f(x), \beta) \quad \hookrightarrow \omega^T \phi(x) \text{ or } \omega^T x$$

$$P(Y/X, \omega) = N(\omega^T \phi(x), \beta)$$

$x_1, y_1, \; x_2, y_2, \; \ldots \; x_n, y_n$

$$\text{Max } P(Y_1/X_1, \omega, \beta) = N(Y_1/\omega^T \phi(x), \beta)$$

$$\text{Max } \prod_{i=1}^{n} N(Y_i/\omega^T \phi(x_i), \beta)$$

$$\text{Max } \prod_{i=1}^{n} \frac{1}{\beta\sqrt{2\pi}} \exp^{-\frac{(x-\omega^T\phi(x))^2}{2\beta^2}}$$

$$\text{Max } \log \prod_{i=1}^{n} \frac{1}{\beta\sqrt{2\pi}} \exp^{-\frac{(x-\omega^T\phi(x))^2}{2\beta^2}}$$

$$\text{Max } -\frac{1}{2}\log\beta - \frac{1}{2}\log(2\pi) - \frac{1}{2\beta^2}\sum_{i=1}^{n} (Y_i - \omega^T\phi(x_i))^2$$

$$\text{Max } -\sum_{i=1}^{n} (Y_i - \omega^T\phi(x_i))^2$$

$$\text{Min } \sum_{i=1}^{n} (Y_i - \omega^T\phi(x_i))^2$$

If sample follows normal distribution then maximization will be equal to the minimization of loss least square loss function.

• Disadvantage of least square loss function is that it is sensitive to presence of the outliers

- Overfitting occurs mostly when our training set is small.
- Introducing Regularization —

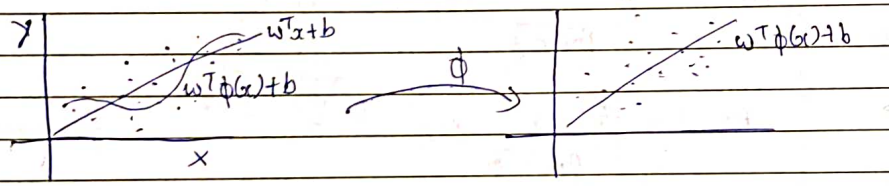
$$\text{Min } \frac{\lambda}{2}\|w\|_2 + \sum_{i=1}^{n}(y_i - (w^T\phi(x_i)+b))^2$$

$$\frac{\partial L}{\partial w} = 0 \qquad \frac{\partial L}{\partial b} = 0$$

$$\begin{bmatrix} w \\ b \end{bmatrix} = (\phi^T\phi + \lambda I)^{-1}\phi^T Y$$

* Read lecture slide-9 (17-02-2022) — (bias, variance, etc.)
- Chances of overfitting decreases if working with large data sets.

* Kernel



mapping to higher dimensional space

x_1 → $\phi_1(x)$
x_2 → $\phi_2(x)$

$$\underbrace{K(x_1,x_2)}_{\text{positive semi-definite}} = \phi(x_1)^T\phi(x_2)$$

$$\boxed{w = \sum_{i=1}^{n} u_i\phi(x_i)}$$

$$f(x) = w^T\phi(x)+b = \left(\sum_{i=1}^{n} u_i\phi(x_i)\right)^T\phi(x)+b$$

$$= \sum_{i=1}^{n} u_i\phi(x_i)^T\phi(x)+b = \sum_{i=1}^{n} K(x_i,x)u_i + b$$

- Least square regularized kernel regression —

$$J(w,b) = \min_{w,b} \frac{\lambda}{2}\|w\|_2 + \sum_{i=1}^{n}(y_i - (\sum_{i=1}^{n} K(x_i,x)u_i + b))^2$$

$$\frac{\partial L}{\partial u} = 0 \quad , u = (X^TX)^{-1}X^TY \quad , X = \begin{bmatrix} K(x_1,x_1) & \cdots & K(x_1,x_n) & 1 \\ \vdots & & & \vdots \\ K(x_n,x_1) & \cdots & K(x_n,x_n) & 1 \end{bmatrix}$$

* - We can also call this as least square support vector regression because it is all about working with kernel generated surfaces.

$$\|w\|^2 = w^Tw$$

$$\nabla_w w^Tw = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \vdots \\ \frac{\partial f}{\partial w_n} \end{bmatrix} = 2\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = 2w$$

$$\nabla_w J(w) = \frac{\lambda}{2}\times 2w + \frac{1}{2}\sum_{i=1}^{n}(y_i - (w^T\phi(x_i)+b))^2$$

$$= \lambda w - \sum_{i=1}^{n}(y_i - (w^T\phi(x_i)+b)\phi(x_i)$$

$$\nabla_b J(w) = 0 - \sum_{i=1}^{n}(y_i - (w^T\phi(x_i)+b))$$

Gradient-descent → do while

$$x^{k+1} = x^k - \eta\nabla f(x^k)$$

until $\|\nabla f(x^k)\|_2 \le \epsilon$

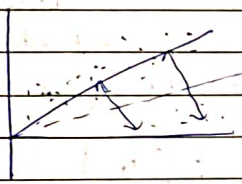
$$\begin{bmatrix} w^{k+1} \\ b^{k+1} \end{bmatrix} = \begin{bmatrix} w^k \\ b^k \end{bmatrix} - \eta\left(\nabla J(w,b)\right)$$

* Stochastic Gradient – we take K-[illegible] data points instead of whole data set, that's why it is also known as K-mini batch stochastic gradient.

* Bringing back robustness in least square estimate by introducing weights, dense points will have more weightage than the sparse ones

$$\min_{w,b} \frac{\lambda}{2} w^T w + \sum_{i=1}^{n} m_i \|Y_i - (w^T \phi(x_i) + b)\|_2^2$$

(m_i : weight)



K-NN can also be used for assigning weightage, as dense points will have lot of neighbours than the sparse ones

* However, we need a new loss function – (using L_1-norm)

not sparse

$$\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \cdot \sum_{i=1}^{n} \|(Y_i - (w^T \phi(x_i) + b)\|_1$$

In L_1-norm, we don't take squares, so it is less sensitive in presence of outliers

- Disadvantage of L_1-norm – function is not smooth, we can not differentiate at every point

* $\min_{w,b} \frac{\lambda}{2} \|w\|_1 + \frac{1}{2} \sum_{i=1}^{n} (Y_i - (w^T \phi(x_i) + b))^2$ — Lasso Regression (sparse sol^n)

In this we are considering L_1-norm for regularization and L_2-norm for loss function to overcome the problem of sensitivity of outliers

not sparse

* $\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1}^{n} (Y_i - (w^T \phi(x_i) + b))^2$ — Ridge Regression

- $w = w^T x + b$

If we have sparse vector (w), then we are dropping some features

$$[w_1, 0, w_2, 0, 0, \ldots w_n] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} + b$$

thus, a simpler model because of usage of less features, so Lasso Regression model will perform better when we will have large data points. [simpler function generalizes well] as variances of unwanted features does not effect our estimate

* $\min_{w,b} \frac{\lambda}{2} \|w\|_1 + \frac{1}{2} \sum_{i=1}^{n} \left(\left| (Y_i - (w^T \phi(x_i) + b)) \right| \right)$ sparse sol^n

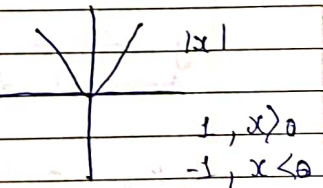
- most convient way to estimate is by using Ridge regression by taking gradient and equating it to zero

* Solution of L_1-norm Regression model – (Optimization problem)

$$\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1}^{n} \left| Y_i - (w^T \phi(x_i) + b) \right|$$

We will solve optimization problem by sub-gradient

we have 2 tangents, so we can not use normal gradient



$$\nabla_w \left(\left| Y_j - (w^T \phi(x_j) + b) \right| \right) = t(\phi(x_j))$$

$$= -\phi(x_j), \text{ if } (Y_j - (w^T \phi(x) + b)) > 0$$

$$= \phi(x_j), \text{ if } (Y_j - (w^T \phi(x) + b)) < 0$$

for kernel regression –

$$\left| y_i - \left(\sum_{i=1} K(x_i, x_j) w_i + b\right) \right| = -\sum_{i=1} K(x_i, x_j), \text{ if } \left(y_i - \left(\sum K(x_i, x_j) w_i + b\right)\right) < 0$$

$$= \sum_{i=1} K(x_i, x_j), \text{ if } \left(y_i - \left(\sum K(x_i, x_j) w_i + b\right)\right) > 0$$

- Laplace Support Vector Regression model –

$$\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1} \left| y_i - \left(\sum_{j=1} K(x_j, x_i) w_j + b\right) \right|$$

- L_1-norm loss is also known as laplace loss

* ϵ-insensitive loss functions –

L_1-norm may be robust but may not be sparse, to overcome this we have ϵ-insensitive loss function

$$\mathcal{L}_\epsilon(u) = \begin{cases} 0, & \text{if } |u| < \epsilon \\ |u| - \epsilon, & \text{otherwise} \end{cases}$$

→ absolute error

if absolute error is less than ϵ, then it is insensitive, if it crosses $|\epsilon|$, then it gets converted to L_1-norm loss

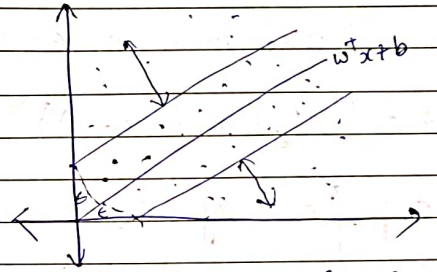
$$\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1} \mathcal{L}_\epsilon \left(y_i - \left(\sum_{j=1} K(x_j, x_i) w_j + b\right)\right)$$

$$\mathcal{L}_\epsilon \left(y_k - \underbrace{\left(\sum_{j=1} K(x_j, x_k) w_j + b\right)}_{w^T \phi(x_j) + b}\right) = \begin{cases} 0, & \text{if } \left|y_k - \left(\sum K(x_j, x_k) w_j + b\right)\right| < \epsilon \\ \left|y_k - \left(\sum_{j=1} K(x_j, x_k) w_j + b\right)\right| - \epsilon, & \text{otherwise} \end{cases}$$

* linear case –

$$\min_{w,b} \frac{\lambda}{2} w^T w + \frac{1}{2} \sum_{i=1} \mathcal{L}_\epsilon (y_k - (w^T x + b))$$

$$\mathcal{L}_\epsilon (y_k - (w^T x_k + b)) = \begin{cases} 0, & \text{if } |y_k - (w^T \phi_x + b)| < \epsilon \\ |y_k - (w^T x_k + b)| - \epsilon, & \text{otherwise} \end{cases}$$



- minimization of ϵ-insensitive loss function is equal to drawing a ϵ-insensitive tube around the data point and ignoring all the data points which are lying inside the ϵ-insensitive tube.
- It is not sensitive to outliers and solution is going to be sparse

* non-linear case –

for non-linear

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{l} |y_i - (w^T \phi(x_i) + b)|_\epsilon \quad [\text{parameters } C, \epsilon, l]$$

$$|y_i - (w^T \phi(x_i) + b)|_\epsilon \rightarrow L_\epsilon(w) = |w|_\epsilon$$

for linear

$$\min_{w,b} \frac{1}{2} w^T w + C \sum_{i=1}^{l} L_\epsilon (y_i - (w^T x_i + b)) \quad [\text{parameters } C, \epsilon]$$

Support Vector regression model is also known as ϵ-SVM.

$$f(w,b) = \frac{1}{2} w^T w + C \sum_{i=1}^{l} L_\epsilon (y_i - (w^T x_i + b))$$

$$\nabla_w f(w,b) = w + C \sum_{i=1}^{l} \begin{cases} 0, & \text{if } -\epsilon \le y_i - (w^T x_i + b) \le \epsilon \\ -x_i, & \text{if } y_i - (w^T x_i + b) \ge \epsilon \\ x_i, & \text{if } y_i - (w^T x_i + b) \le -\epsilon \end{cases}$$

$$\nabla_b f(w,b) = \sum_{i=1}^{l} \begin{cases} 0, & \text{if } -\epsilon \le y_i - (w^T x_i + b) \le \epsilon \\ -1, & \text{if } y_i - (w^T x_i + b) \ge \epsilon \\ 1, & \text{if } y_i - (w^T x_i + b) \le -\epsilon \end{cases}$$

$$\begin{bmatrix} w^{k+1} \\ b^{k+1} \end{bmatrix} = \begin{bmatrix} w^k \\ b^k \end{bmatrix} - \eta \begin{bmatrix} \nabla_w f(w,b) \\ \nabla_b f(w,b) \end{bmatrix}$$

until $\left\| \begin{matrix} \nabla_w f(w,b) \\ \nabla_b f(w,b) \end{matrix} \right\| \le$ some value

for non-linear

$$f(w,b) = \frac{1}{2} w^T w + C \sum_{i=1}^{l} L_\epsilon \left(y_i - \left(\sum_{j=1}^{l} K(x_j, x_i) w_j + b\right)\right)$$

$$\nabla_w f(w,b) = w + C \sum_{i=1}^{l} \begin{cases} 0, & \text{if } -\epsilon \le y_i - \left(\sum_{j=1}^{l} K(x_j, x_i) w_j + b\right) \le \epsilon \\ \begin{bmatrix} K(x_i, x_1) \\ K(x_i, x_2) \\ \vdots \\ K(x_i, x_l) \end{bmatrix}, & \text{if } y_i - \sum K(x_j, x_i) w_j + b \ge \epsilon \\ -\begin{bmatrix} K(x_i, x_1) \\ K(x_i, x_2) \\ \vdots \\ K(x_i, x_l) \end{bmatrix}, & \text{if } y_i - \sum K(x_j, x_i) w_j + b \le -\epsilon \end{cases}$$

* Resampling method —

Data

- Training set (used for tuning parameters)
- Validation set
- Testing set (unknown)

- K-fold cross validation — will give k RMSE values and we will take mean of that along with ± standard deviation. It is good for large dataset

RMSE = ___ ± S.D

A good regression model will obtain lower mean RMSE value and lower variance of RMSE vales.

- Leave-one-out validation – Here we leave one data point for testing and rest are taken for training.
  e.g. 90 data points → 89 for training & 1 for testing
  So, we will have 89 RMSE's & 89 variances.

  In k-fold if k→1, then it becomes leave-one-out. Variance will be less for leave-one-out, but will be high for low value of k in k-fold.

- $RMSE = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(f(x_i)-y_i)^2}$

- SSE (Sum of square of error) $= \sum_{i=1}^{k}(f(x_i)-y_i)^2$

  $SST = \sum_{i=1}^{k}(y_i-\bar{y_i})^2$ [Variance in Y]

- $NMSE = \frac{SSE}{SST} = \frac{\sum_{i=1}^{k}(f(x_i)-y_i)^2}{\sum_{i=1}^{k}(y_i-\bar{y_i})^2}$

- $MAE = \frac{1}{k}\sum_{i=1}^{k}|(f(x_i)-y_i)|$

- $R^2 = \frac{SSR}{SST}$, $SSR = \sum_{i=1}^{k}(f(x_i)-f(\bar{x}))^2$

  $R^2 = \frac{\sum_{i=1}^{k}(f(x_i)-f(\bar{x}))^2}{\sum_{i=1}^{k}(y_i-\bar{y})^2}$, if $R^2 \simeq 1$, that means model is good

For model to be good — $R^2 \simeq 1$
$RMSE \simeq 0$
$NMSE \simeq 0$
$MAE \simeq 0$

- Sparsity – also a measure of evaluation, we see how many values are '0' of solution
  $\Rightarrow \frac{\text{No. of } (w_i=0)}{\text{Total No of } w_i}$