EE 559

Midterm Assignment

Sarthak Kumar Maharana

1.

(a) For a 2-class nearest-means classifier, the discriminant function is as follows :

For class 1 : $g_1(\underline{x}) = \|\underline{x} - \mu_1\|_2^2$ ✓

For class 2 : $g_2(\underline{x}) = \|\underline{x} - \mu_2\|_2^2$ ✓

Overall, $g(\underline{x}) = \|\underline{x} - \underline{\mu_1}\|_2^2 - \|\underline{x} - \underline{\mu_2}\|_2^2 = 2(\underline{\mu_1}^T\underline{x} + \underline{\mu_2}^T\underline{x}) - \underline{\mu_1}^T\underline{\mu_1} + \underline{\mu_2}^T\underline{\mu_2}$ //

and so, $\begin{cases} \text{class 1} & g(\underline{x}) \geq 0 \\ \text{class 2} & g(\underline{x}) < 0 \end{cases}$ ← prediction

(b) Let the class under consideration be 'k'.

$\tilde{g}_k(\underline{x}) = \|\underline{x} - \underline{\mu_k}\|_2^2$

$= (\underline{x} - \underline{\mu_k})^T(\underline{x} - \underline{\mu_k}) = \underline{x}^T\underline{x} - \mu_k^T\underline{x} - \underline{x}^T\mu_k + \mu_k^T\mu_k$

$= \underline{x}^T\underline{x} - 2\mu_k^T\underline{x} + \mu_k^T\mu_k$.

Let $g_k(\underline{x}) = 2\mu_k^T\underline{x} - \mu_k^T\mu_k$; $\tilde{g}_k(\underline{x}) = \underline{x}^T\underline{x} - g_k(\underline{x})$

Let's include a quadratic non-linear mapping for two features :

$[1, x_1, x_2, x_1x_2, x_1^2, x_2^2]$. ← augmented notation

$$g_k(\underline{x}) = \frac{2}{N_k}\sum_{i=1}^{N_k} x_i^{(k)T} \cdot \underline{x} - \left(\frac{1}{N_k}\sum_{i=1}^{N_k} x_i^{(k)T}\right)\left(\frac{1}{N_k}\sum_{i=1}^{N} x_i^{(k)}\right)$$

where $k \rightarrow$ current class

$N_k \rightarrow$ total data of class k

$$= \frac{2}{N_k}\sum_{i=1}^{N} x_i^{(k)T}\underline{x} - \frac{1}{N_k^2}\sum_{i=1}^{N_k}\sum_{l=1}^{N_k} x_i^{(k)T}x_l^{(k)} \quad \rightarrow (1)$$

Clearly, we've represented or introduced the inner product notation $\langle \cdot , \cdot \rangle$ for non-linear mapping. And so, this is the dual representation.

$$
\begin{aligned}
g(\underline{x}) &= \| \underline{x} - \underline{\mu_2} \|_2^2 - \| \underline{x} - \underline{\mu_2} \|_2^2 \\
&= \underline{x}^T \underline{x} - 2\underline{\mu_2}^T \underline{x} + \underline{\mu_2}^T \underline{\mu_2} - \underline{x}^T \underline{x} + 2\underline{\mu_2}^T \underline{x} - \underline{\mu_2}^T \underline{\mu_2} \\
&= -2\underline{\mu_2}^T \underline{x} + \underline{\mu_1}^T \underline{\mu_1} + 2\underline{\mu_2}^T \underline{x} - \underline{\mu_2}^T \underline{\mu_2} \\
&= \bullet \left(g_2(\underline{x}) \bullet + g_1(\underline{x}) \right).
\end{aligned}
$$

or $g_D(\underline{x}) = \bullet \left(g_1(\underline{x}) + g_2(\underline{x}) \right)$ //

(c) In equation ①, we can substitute the RBF kernel.

$$k(\underline{x}_1, \underline{x}_2) = \exp\left\{ -\gamma \| \underline{x}_1 - \underline{x}_2 \|_2^2 \right\}$$

∴ ① becomes:

$$\frac{2}{N_K} \sum_{i=1}^{N_K} \underline{x}_i^{(k)T} \underline{x} - \frac{1}{N_K^2} \sum_{i=1}^{N_K} \sum_{l=1}^{N_K} \exp\left\{ -\gamma \| \underline{x}_i^{(k)} - \underline{x}_l^{(k)} \|_2^2 \right\} ;$$

$K \rightarrow$ current class

$N_K \rightarrow$ total points of class K

and so,

$$
\begin{aligned}
g_{rbf}(\underline{x}) &= g_1(\underline{x}) + g_2(\underline{x}) \\
&= \frac{2}{N_1^2} \sum_{i=1}^{N_1} \underline{x}_i^{(1)T} \underline{x} - \frac{1}{N_1^2} \sum_{i=1}^{N_1} \sum_{l=1}^{N_1} \exp\left\{ -\gamma \| \underline{x}_i^{(1)} - \underline{x}_l^{(1)} \|_2^2 \right\} \\
&\quad + \frac{2}{N_2^2} \sum_{i=1}^{N_2} \underline{x}_i^{(2)} \underline{x} - \frac{1}{N_2^2} \sum_{i=1}^{N_2} \sum_{l=1}^{N_2} \exp\left\{ -\gamma \| \underline{x}_i^{(2)} - \underline{x}_l^{(2)} \|_2^2 \right\}.
\end{aligned}
$$
//

The following questions have been answered and solved with respect to a **non-augmented feature space**.

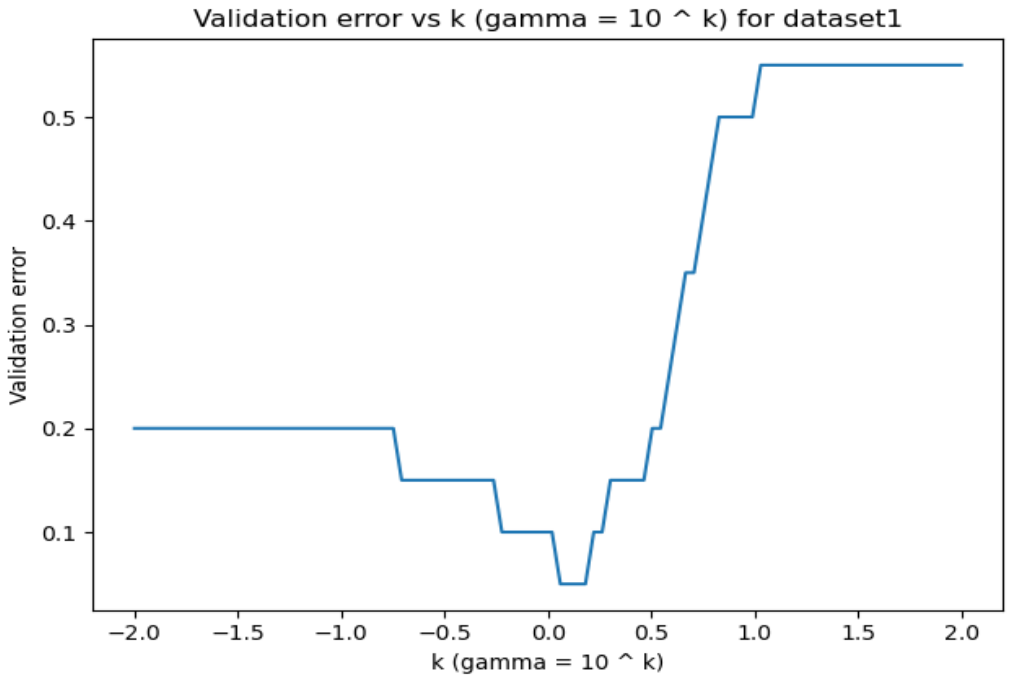
Objective: Code up a 2-class kernel nearest means classifier that can use Radial Basis Function (RBF) or linear kernel.

(d) The validation set, as per model selection, is used to choose the best performing model. In the current problem, the validation set is used to obtain the optimal value of gamma for the RBF kernel. Here, based on the validation set, the value of gamma that gives the lowest classification error, is chosen as the optimal gamma for the rest of the problems that follow.

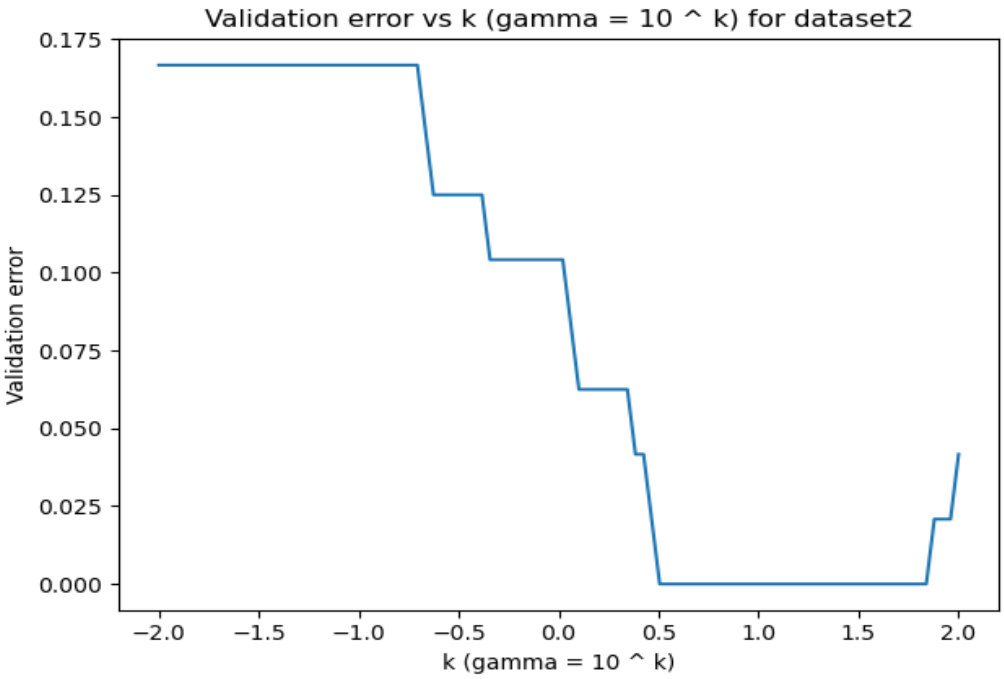
For **dataset 1**: optimal gamma --- **1.1497**
For **dataset 2**: optimal gamma --- **3.1992**

(e) To obtain the optimal gamma, a brute-force search was done with the values of k ranging from **-2 to 2**, for **100 points in between**. The following figures illustrate the validation-set classification error vs k, where gamma = 10^k.

For **dataset 1:**



For **dataset 2:**



(f) For **dataset 1:**

The test set error using the linear kernel is : **0.43 (43%)**

The test set error using the RBF kernel is : **0.23 (23%)**

For **dataset 2:**

The test set error using the linear kernel is : **0.243 (24.3%)**

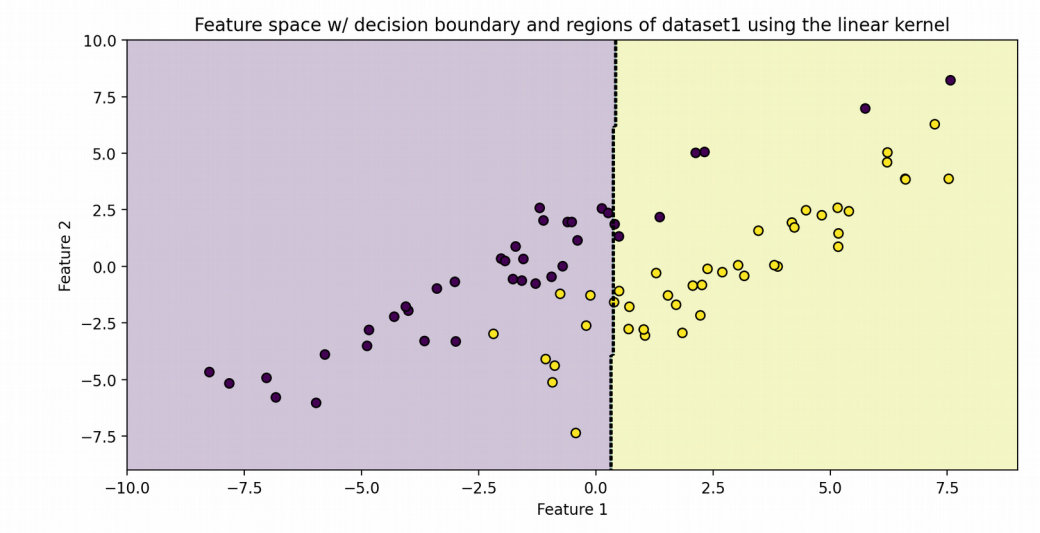
The test set error using the RBF kernel is **: 0.0125 (1.25%)**

An RBF kernel's main objective is used to find a non-linear classifier or regression decision boundary. The gamma used in the kernel is used to control the degree to which two data points are to be similar to each other. This increases the chances that the decision boundary fits well and helps in allowing the dataset to be linearly seperable, in a higher dimensional feature space. A linear class doesn't do well in a higher-dimensional space, and utterly fails in allowing the dataset to be linearly seperable.

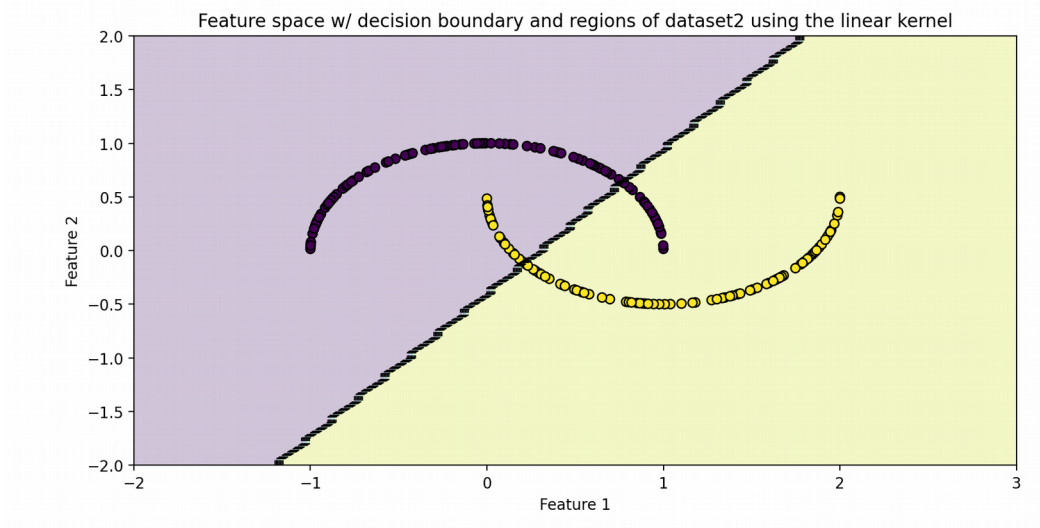
(g) The following figures illustrate the use of the **linear kernel**, for classification, on each dataset. Here, we plot the training data points, decision regions and boundary, in the feature space.

(The linear boundaries look a little rugged because of being stretched.)

For **dataset 1:**

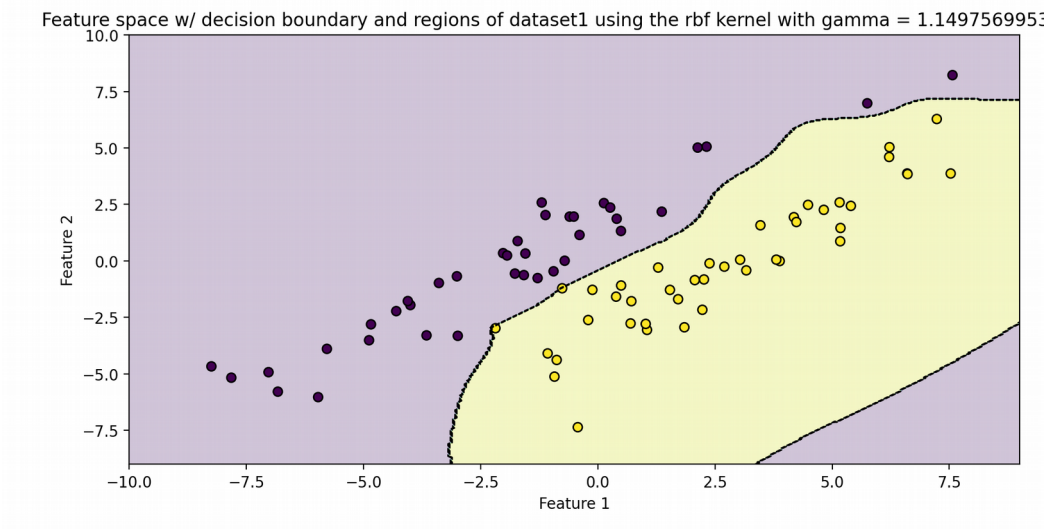


For **dataset 2:**

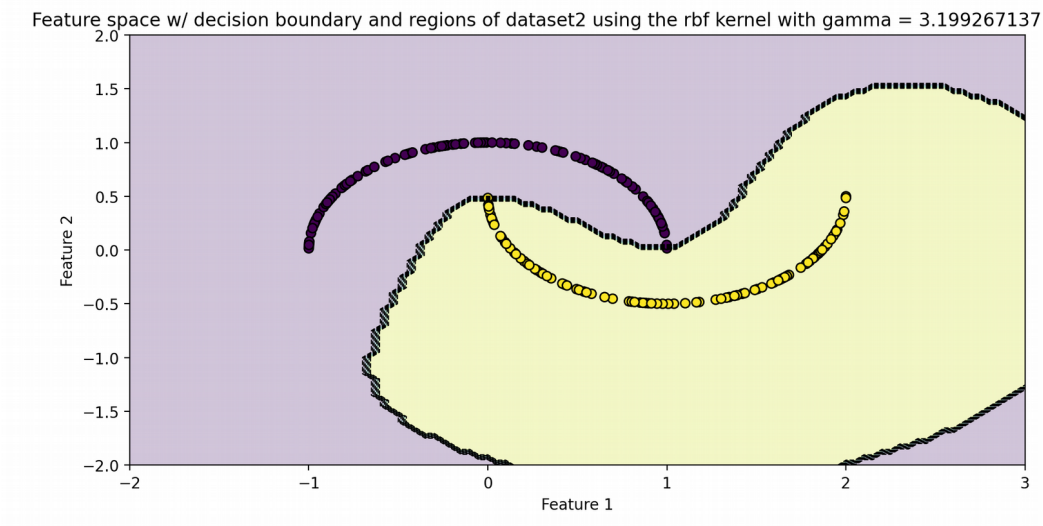


(h) The following figures illustrate the use of the **RBF kernel (w/ optimal gamma)**, for classification, on each dataset.

For **dataset 1:**



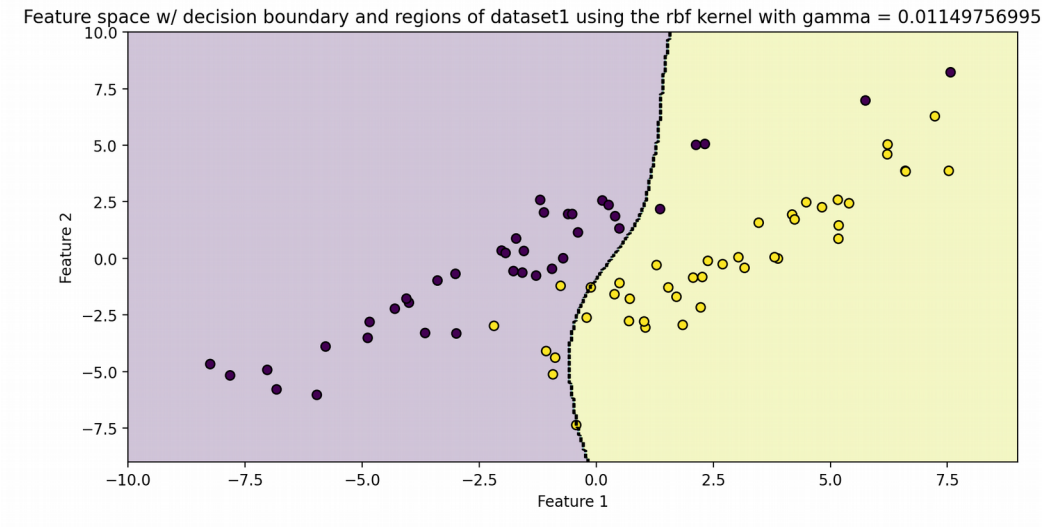
For **dataset 2:**



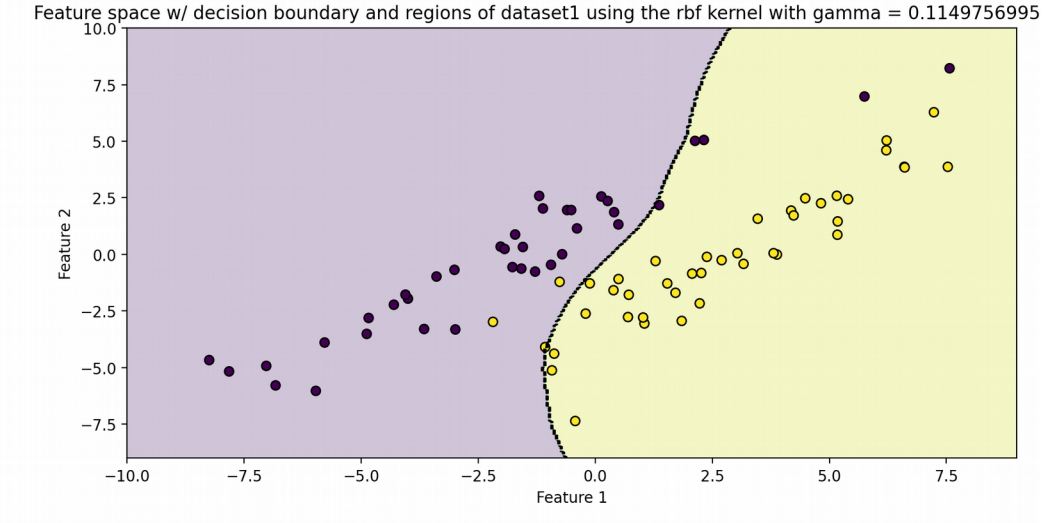
(i) In this subproblem, we repeat (h) but with different values of the multiples of optimal gamma i.e the multiples include 0.01, 0.1, 0.3, 3, 10, 100 for each dataset.

For **dataset 1:**

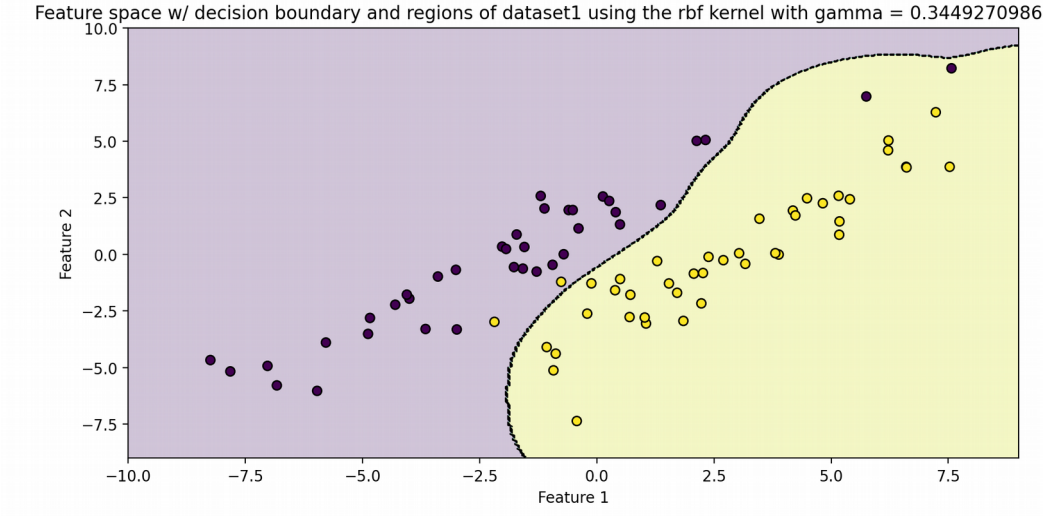
gamma = 0.01 * 1.1497 = 0.011497



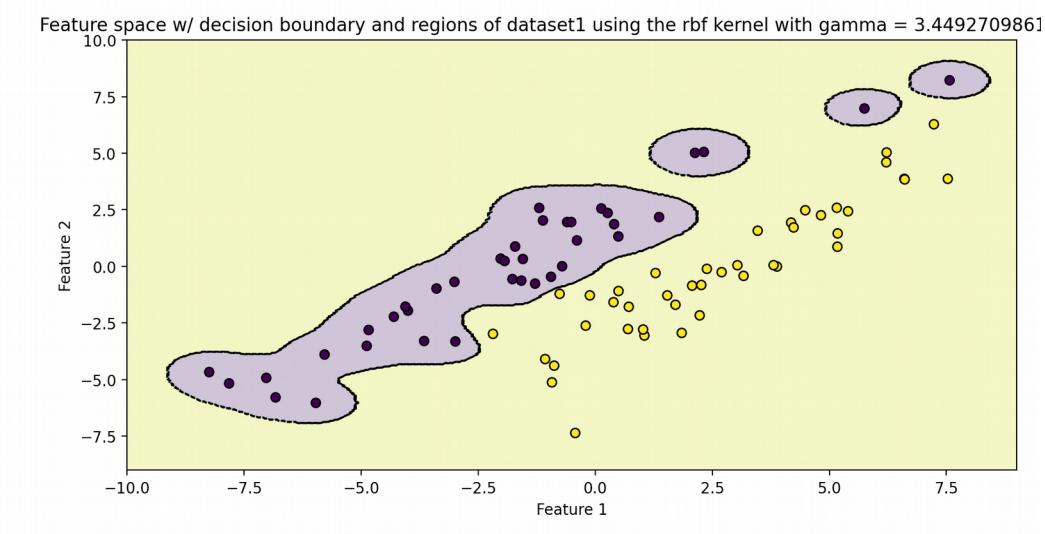
gamma = 0.1 * 1.14967 = 0.11497



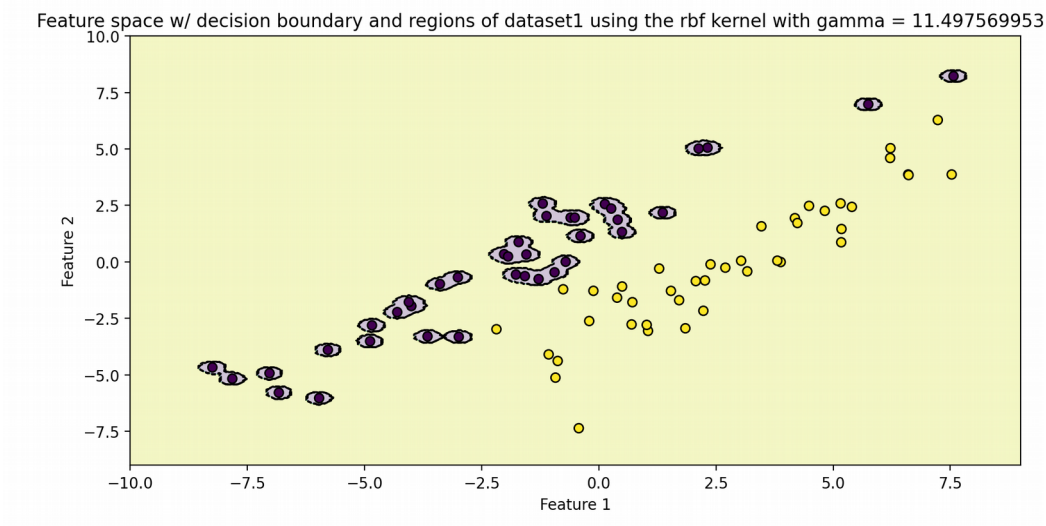
gamma = 0.3 * 1.14967 = 0.3449



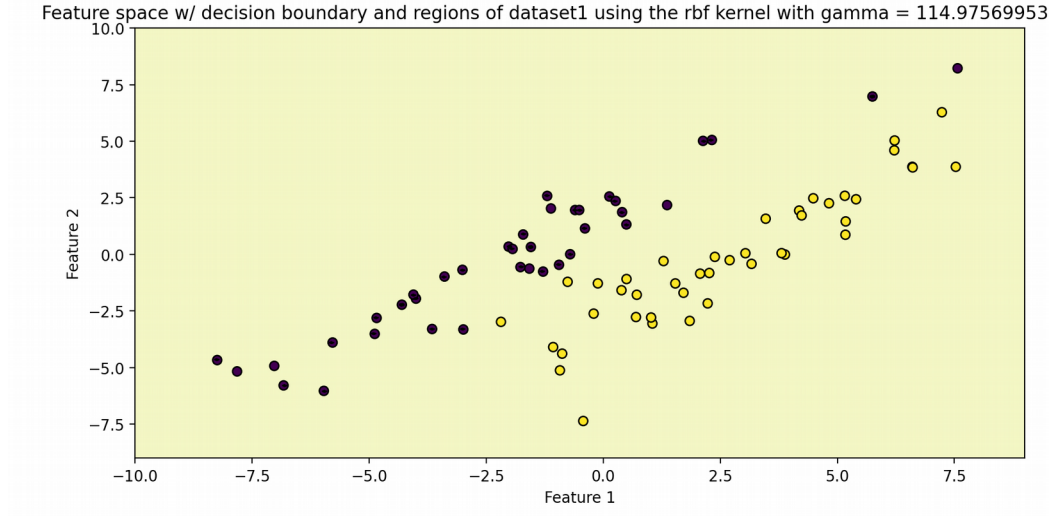
gamma = 3 * 1.14967 = 3.4492



gamma = 10 * 1.14967 = 11.4967

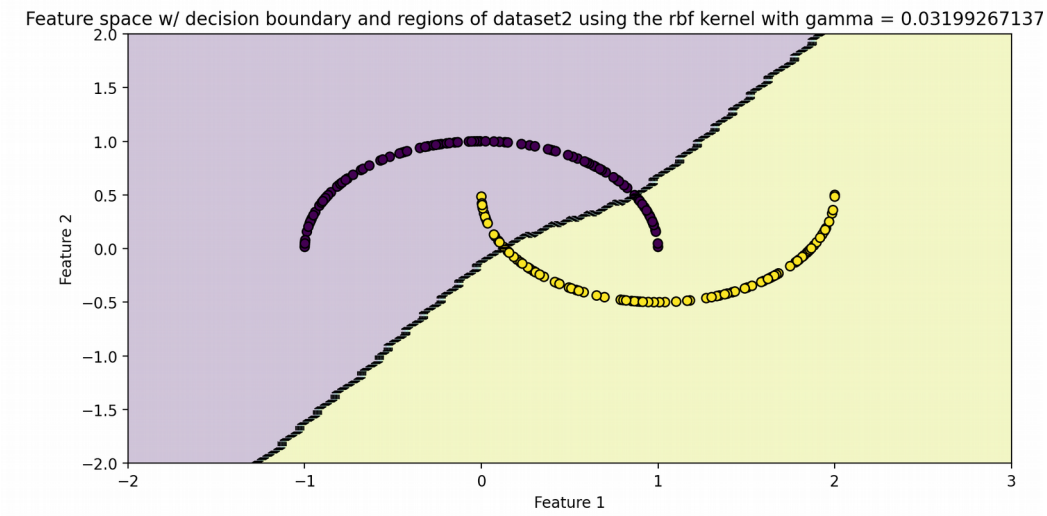


gamma = 100 * 1.14967 = 114.967

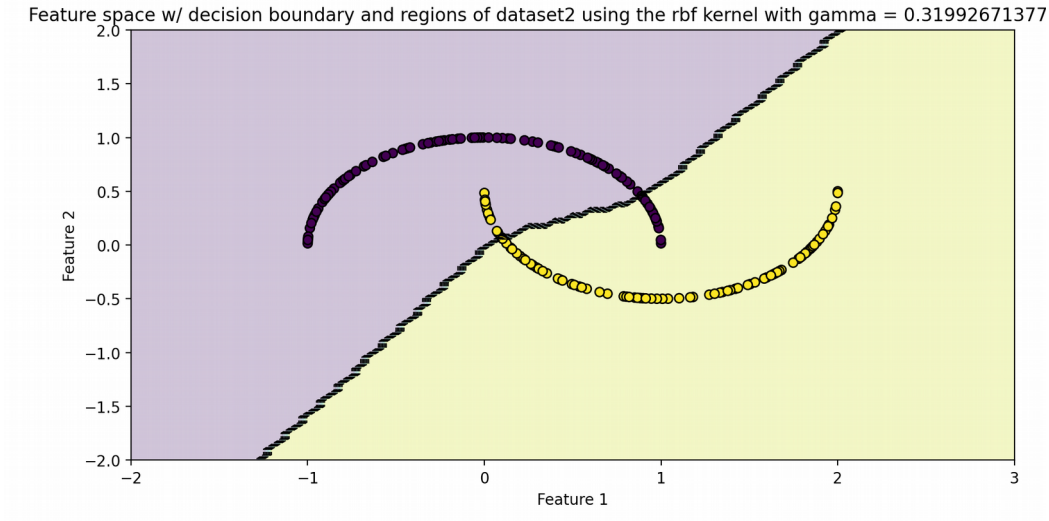


For **dataset 2:**

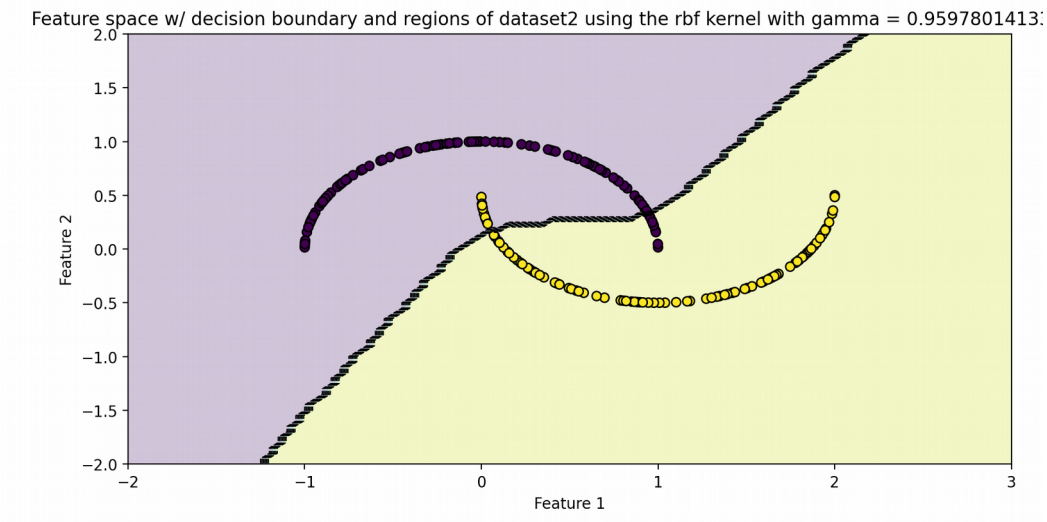
gamma = 0.01 * 3.1992 = 0.031992



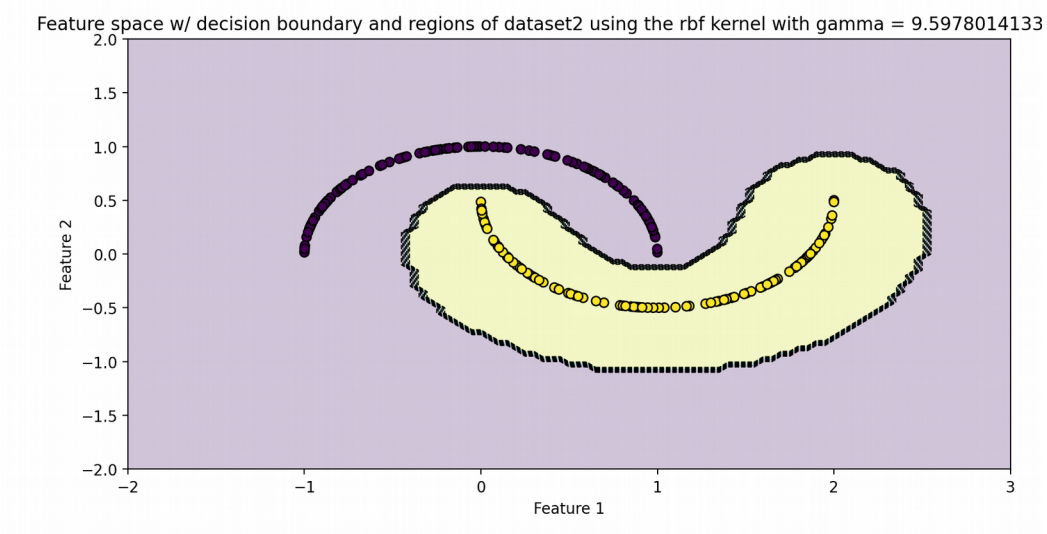
gamma = 0.1 * 3.1992 = 0.31992



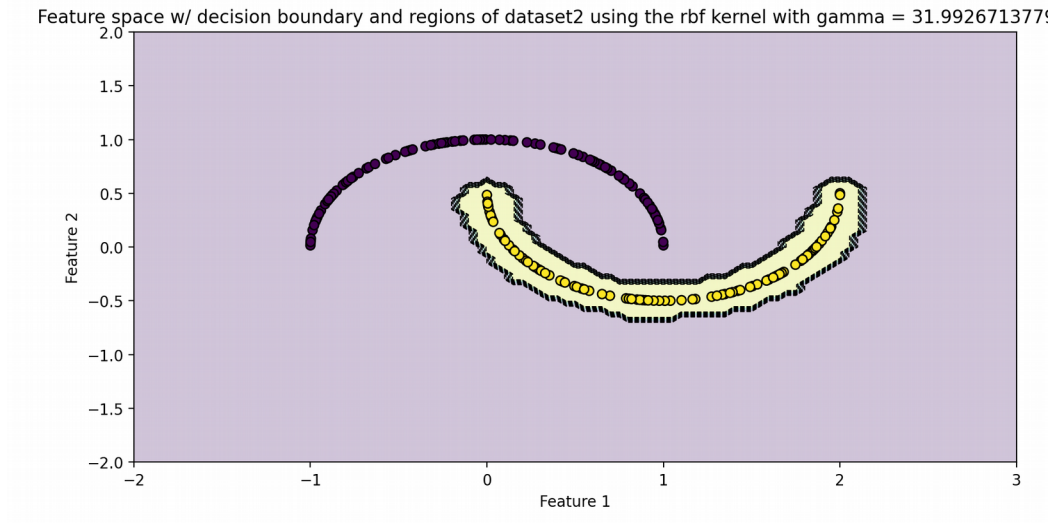
gamma = 0.3 * 3.1992 = 0.9597



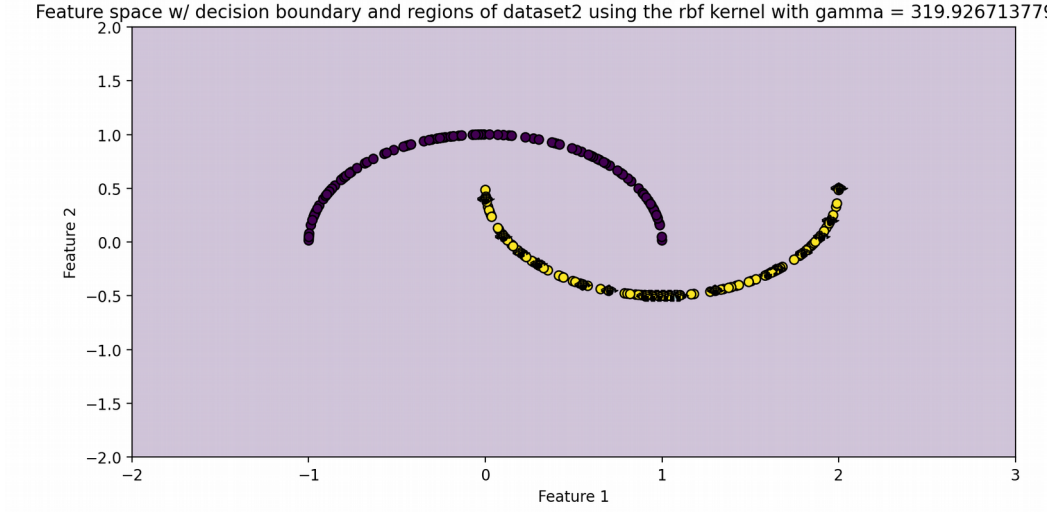
gamma = 3 * 3.1992 = 9.597



gamma = 10 * 3.1992 = 31.992



gamma = 100 * 3.1992 = 319.92



We observe that as the value of *gamma* is increased beyond the optimal gamma (for each dataset), the performance of non-linear classification begins to get better. As said earlier, gamma parameterizes and models the similarity between two data points. And so, a higher value *tries* to bring a higher similarity between the training and unknown (validation/test) data points. In a standard RBF kernel, gamma is defined as the inverse of the variance of the Gaussian/normal distribution (that the RBF's probability distribution underlies). And so, larger values of gamma ensure that the variance between the two data points is as low as possible, resulting in high similarity and better model performance on unknowns. At higher values of the optimal gamma, the decision boundary performs extremely well in classifying between the two classes of data, and so the error rate reduces massively (expected).

2. Given criterion function of a 2-class problem (using augmented notation) :

$$J(\underline{w}) = -\sum_{n=1}^{N} [[g(z_n \underline{x}_n) \leq 0)]] \frac{\underline{w}^T z_n \underline{x}_n}{\|\underline{w}\|_2}$$

(a) Based on the indicator function, $J(\underline{w})$ can be rewritten as :

$$J(\underline{w}) = \begin{cases} -\sum_{n=1}^{N} \frac{\underline{w}^T z_n \underline{x}_n}{\|\underline{w}\|_2} & g(z_n x_n) \leq 0 \rightarrow \text{misclassified} \\ 0 & \text{else} \rightarrow \text{correct classification.} \end{cases}$$

Now, $\frac{w^T z_n x_n}{\|w\|_2}$ can be interpreted as the distance of the point x_n from the hyperplane H.

or $J(\underline{w})$ = - (Sum of all distances of the data points from the hyperplane) ; given the current weight vector.

And so, when there are correct classifications, $J(\underline{w}) = 0$.
If the misclassified points are close to each other, the magnitude of $J(\underline{w})$ can be brought down.

(b) $$J(\underline{w}) = -\sum_{n=1}^{N} [[g(z_n \underline{x}_n) \leq 0)] \frac{\underline{w}^T z_n \underline{x}_n}{\|w\|_2}$$

$$= \begin{cases} -\sum_{n=1}^{N} \frac{\underline{w}^T z_n \underline{x}_n}{\|w\|_2} & \text{if } g(z_n x_n) \leq 0 \\ 0 & g(z_n \underline{x}_n) > 0. \end{cases}$$

Taking the gradient of $J(\underline{w})$ wrt $\underline{w}$:

$$\nabla_{\underline{w}} J(\underline{w}) = -\sum_{n=1}^{N} \frac{\|w\|_2 z_n x_n - \underline{w}^T z_n x_n \cdot \frac{1}{2}(\underline{w}^T\underline{w})^{-1/2} 2\underline{w}}{\|w\|_2^2}$$

$$= -\sum_{n=1}^{N} \frac{\|\underline{w}\|_2 z_n \underline{x}_n - \underline{w}^T z_n x_n \cdot \frac{\underline{w}}{\|w\|_2}}{\|w\|_2^2}$$

$$= -\sum_{n=1}^{N} \frac{z_n \underline{x}_n}{\|w\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|w\|_2^3}$$

or $\nabla_{\underline{w}} J(\underline{w}) = \begin{cases} -\sum_{n=1}^{N} \frac{z_n \underline{x}_n}{\|\underline{w}\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|\underline{w}\|_2^3} & \text{if } g(z_n x_n) \le 0 \\ 0 & \text{else.} \end{cases}$

Batch gradient descent

The weight update rule is as follows :

$$\underline{w}(i+1) = \underline{w}(i) - \eta(i) \nabla_{\underline{w}} J(\underline{w}) \qquad \eta(i) \rightarrow \text{learning rate}$$

$$= \underline{w}(i) + \eta(i) \sum_{n=1}^{N} \left(\frac{z_n \underline{x}_n}{\|\underline{w}\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|\underline{w}\|_2^3} \right).$$

Stop when $\sum_{n=1}^{N} \frac{z_n \underline{x}_n}{\|\underline{w}\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|\underline{w}\|_2^3} = 0$ ✓

$$w(i+1) = w(i) + \eta(i) \sum_{n=1}^{N} [[\, g(z_n \underline{x}_n \le 0)]] \left(\frac{z_n \underline{x}_n}{\|\underline{w}\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|\underline{w}\|_2^3} \right).$$

Stochastic gradient descent - variant 1 :

(Randomly shuffle before each epoch)

Iterate over $m = 1, 2, 3, \ldots$ until no change in $\underline{w}$ over 1 full epoch.

(i) Randomly shuffle dataset

(ii) Define the sequential epoch variable n as :

for n in range (len (N)) : $N \rightarrow$ data points

$i = (m-1)\,N + (n+1)$.

$$\begin{cases} \underline{w}(i+1) = \underline{w}(i) + \eta(i)\, \nabla_{\underline{w}} J(\underline{w}) & \text{if } \underline{w}^T z_n \underline{x}_n \le 0 \\ \underline{w}(i+1) = \underline{w}(i) . & \text{if } \underline{w}^T z_n \underline{x}_n > 0 \end{cases}$$

$\nabla_{\underline{w}} J(\underline{w}) \rightarrow$ gradient of $J(\underline{w})$ defined earlier

$$= \sum_{n=1}^{N} \frac{z_n \underline{x}_n}{\|\underline{w}\|_2} - \frac{\underline{w}^T z_n \underline{x}_n \underline{w}}{\|\underline{w}\|_2^3} \quad \text{if } g(z_n \underline{x}_n) \le 0$$

(c) Total features $= D = 2$.

$\eta(i) = \frac{1}{1+i}$.

For epoch 1 : $(\underline{x}_1, \underline{x}_2, \underline{x}_3, \underline{x}_4)$

(i) $S_1 : \underline{x}_1 = [-4, 1]^T$, $\underline{x}_2 = [-3, -2]^T$

$S_2 : \underline{x}_3 = [-1, 3]^T$, $\underline{x}_4 = [3, -2]^T$

Since we're dealing w/ a non-augmented feature space

(ii) $\underline{w}(1) = [-1, 4]^T$ $\quad w_1 = -1$; $w_2 = 4$. (since non-aug)

(iii) As found above,

$$\underline{w}(i+1) = \underline{w}(i) + \eta(i)\, \nabla_{\underline{w}} J(\underline{w})$$

$$\text{or } \underline{w}(2) = \underline{w}(1) + \eta(1)\left[\frac{z_2 x_2}{\|w(i)\|_2} - \frac{w(1)^T z_2 x_2 w(1)}{\|w(i)\|_2^3}\right].$$

$$\eta(1) = \frac{1}{2}.$$

(iv) $$\underline{w}(2) = [-1\ 4]^T + \frac{1}{2}\left[\frac{\begin{bmatrix}-3\\-2\end{bmatrix}}{\|[-1,4]^T\|_2} - \frac{[-1,4]^T[-3,-2]^T[-1,4]^T}{\|[-1,4]\|_2^3}\right]$$

$$= [-1,4]^T + \frac{1}{2}\left[\frac{\begin{bmatrix}3\\-2\end{bmatrix}}{\sqrt{1^2+4^2}} - \frac{\begin{bmatrix}5\\-20\end{bmatrix}}{(1^2+4^2)^{3/2}}\right].$$

$$= [-1\ 4]^T + \frac{1}{2}\left[\frac{\begin{bmatrix}3\\-2\end{bmatrix}}{\sqrt{17}} - \frac{\begin{bmatrix}5\\-20\end{bmatrix}}{(\sqrt{17})^3}\right].$$

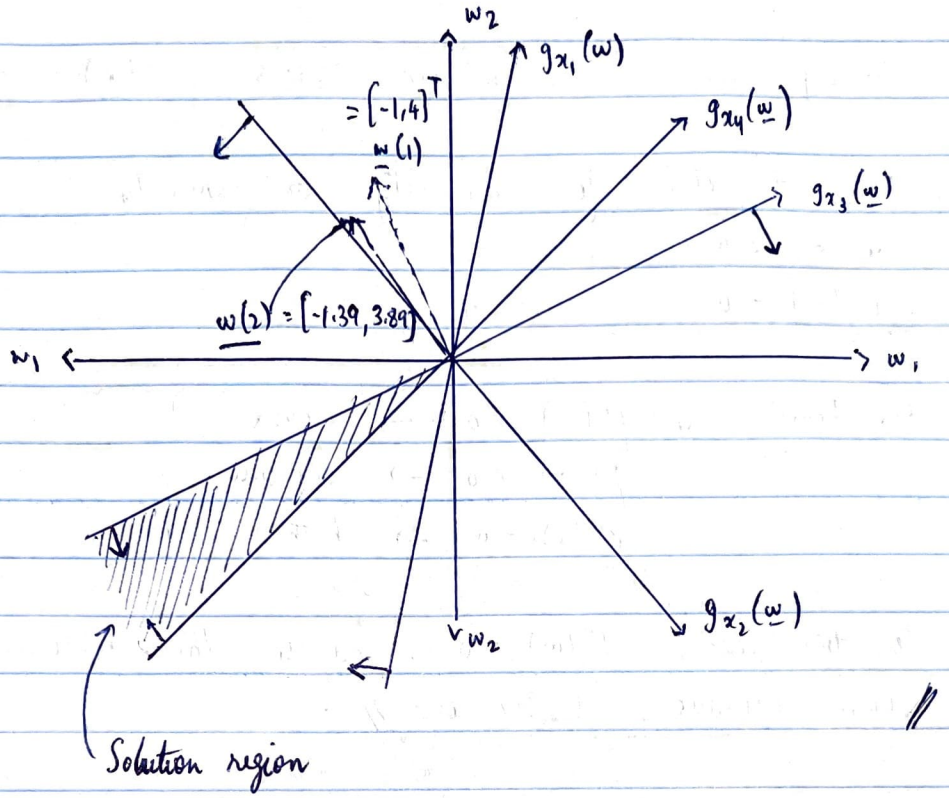
$$= \begin{bmatrix}-1\\4\end{bmatrix} + \frac{1}{2}\left[\begin{bmatrix}-0.727\\-0.485\end{bmatrix} - \begin{bmatrix}-0.071\\-0.98\end{bmatrix}\right]$$

$$\text{or } \underline{w}(2) = \begin{bmatrix}-1.399\\3.897\end{bmatrix} \begin{matrix}\leftarrow w_1\\\leftarrow w_2\end{matrix}$$

Or, at iteration 2, $\left.\begin{matrix}w_1 = -1.399\\w_2 = 3.897\end{matrix}\right\} = w(2)$ //

Plot for 2

(c)



(d) We're given, $\underline{w} = w$, $\underline{x_n} = x$ and $w_0 = 0$ } scalars.

$$J(w) = \begin{cases} -\sum_{n=1}^{N} [[\, g(z_n x_n) \le 0 \,]] \, \dfrac{w z_n x_n}{\|w\|_2} & g(z_n x_n) \le 0 \\ 0 \quad \text{else} \end{cases}$$

The L2 norm of a scalar is its absolute value.

$$= \begin{cases} -\sum_{n=1}^{N} \dfrac{w z_n x_n}{|w|} & w z_n x_n \le 0 \\ 0 & \text{else} \end{cases}$$

$$= \begin{cases} \sum_{n=1}^{N} z_n x_n & w \le 0 \\ -\sum_{n=1}^{N} z_n x_n & w > 0 \\ 0 \quad \text{else} \end{cases} \Bigg\} \; w z_n x_n \le 0 .$$

Clearly, since z_n & x_n are scalars, $J(\underline{w})$ = constant = λ

Now, to check for convexity or concavity,
$J'(w) = 0$.
$J''(w) = 0$

We know, if $f''(x) > 0 \longrightarrow$ convex
$f''(x) \leq 0 \longrightarrow$ concave
$f''(x) = 0 \longrightarrow$ both

In this case, $J''(w) = 0$ and so the criterion function is both concave and convex. //.

This clearly states that the weight update will happen for an infinite amount of time and won't reach convergence.

y_N

3. Total no. of training data points = N.
No. of features = $D = 3$.

(a) In approach A1, we do a degree 3 or cubic polynomial mapping from $\underline{x}$ to $\underline{u}$ (expanded feature space) and then use a linear regressor in $\underline{u}$ space. Let the features be (x_1, x_2, x_3).

$$\therefore \phi(\underline{x}) = \underline{u} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \\ x_1^2 \\ x_2^2 \\ x_3^2 \\ x_1x_2 \\ x_1x_3 \\ x_2x_3 \\ x_1^3 \\ x_2^3 \\ x_3^3 \\ x_1x_2^2 \\ x_1x_3^2 \\ x_2x_3^2 \\ x_1^2x_2 \\ x_1^2x_3 \\ x_2^2x_3 \\ x_1x_2x_3 \end{bmatrix} \quad w' = \begin{bmatrix} w_0 \\ w_{01} \\ w_{02} \\ w_{03} \\ w_{01}^2 \\ w_{02}^2 \\ w_{03}^2 \\ w_{01}w_{02} \\ w_{01}w_{03} \\ w_{02}w_{03} \\ w_{01}w_{01}w_{01} \\ w_{02}^2 \\ w_{03}^2 \\ w_{01}w_{02}^2 \\ w_{01}w_{03}^2 \\ w_{02}w_{03}^2 \\ w_{01}^2w_{02} \\ w_{01}^2w_{03} \\ w_{02}^2w_{03} \\ w_{01}w_{02}w_{03} \end{bmatrix}$$

(augmented notation)

20×1 $\qquad$ 20×1

$y_N \rightarrow$ target values

The expanded feature space & the weight vector have a total dimension of 20. If the labels are denoted by y_N in the new space, $J_{A1}(\underline{w}') = \frac{1}{N} \| \phi(\underline{x})\underline{w}' - \underline{y}_N \|_2^2$

$= \frac{1}{N} \| \underline{u}\underline{w}' - \underline{y}_N \|_2^2$. //

(b) Degrees of freedom is defined as the total number of parameters that have to be fit to the data. Clearly, the total parameters = 20. And so, dof = 20 //.

(c) In approach A_1, $J(w') = \frac{1}{N} \| \underline{u}\underline{w}' - \underline{y}_N \|_2^2$.

To obtain the optimal weight vector, we use the Moore-Penrose method and so,

$\hat{\underline{w}}' = \underline{\underline{U}}^{-1} \underline{y}_N$ where $\underline{\underline{U}}^{-1} = (\underline{\underline{U}}^T \underline{\underline{U}})^{-1} \underline{\underline{U}}^T$

$\underline{\underline{U}} \rightarrow$ New feature space of dimension = 20.
$\underline{y}_N \rightarrow$ Labels in the new space.

(d) As per approach 2, in the original feature space, $\hat{f}(\underline{x}) = \underline{w}^T \underline{x}$.

In the nonlinear space, $\hat{y}(\underline{x}) = w_1' (\hat{f}(\underline{x})) + w_2' (\hat{f}(\underline{x}))^2 + w_3' (\hat{f}(\underline{x}))^3$.

Let $\underline{w}' = [w_1', w_2', w_3']$.

$\therefore J(\underline{w}'\underline{w}) = \frac{1}{N} \| \hat{\underline{y}}(\underline{x}) - \underline{y}_N \|_2^2$ $N \rightarrow$ total data points

$= \frac{1}{N} \| w_1' \hat{f}(\underline{x}) + w_2' \hat{f}(\underline{x})^2 + w_3' \hat{f}(\underline{x})^3 - \underline{y}_N \|_2^2$

$J(\underline{w}, \underline{w}') = \frac{1}{N} \| [\hat{f}(\underline{x}) \ \hat{f}(\underline{x})^2 \ \hat{f}(\underline{x})^3]^T \underline{w}' - \underline{y}_N \|_2^2$ $y_N \rightarrow$ training labels or target values

$= \frac{1}{N} \| [\underline{w}^T \underline{x} \ (\underline{w}^T \underline{x})^2 \ (\underline{w}^T \underline{x})^3]^T \underline{w}' - \underline{y}_N \|_2^2$ //.

(e) Since, it's a 3 feature training matrix

$\underline{w} \rightarrow$ 4 dimensional (since augmented)
In the expanded feature space, $\underline{w}' = (w_1' \; w_2' \; w_3') = 3$.

Total dof $\rightarrow 4+3 = 7$ //.

(f) $J(\underline{w}') = \frac{1}{N} \| \hat{y}(\underline{x}) - y_N \|_2^2$

but $\hat{y}(\underline{x}) = w_1' (\hat{f}(\underline{x})) + w_2' (\hat{f}(\underline{x}))^2 + w_3' (\hat{f}(\underline{x}))^3$.
$\hat{f}(\underline{x}) = \underline{w}^T \underline{x}$

Clearly, $\hat{y}(\underline{x})$ is **non-linear** wrt to $\underline{x}$ (since it's a polynomial)

$\hat{y}(\underline{x})$ is **non-linear** w.r.t to $\underline{w}$ and **linear** w.r.t $\underline{w}'$.

$\hat{y}(x)$ is directly parameterised as a linear function of $\underline{w}' = [w_1', w_2', w_3']$.

(g) Criterion function of A2 $\rightarrow$ $J(\underline{w}, \underline{w}')$ ~~$J(\underline{w}')$~~ $= \frac{1}{N} \| \hat{y}(\underline{x}) - y_N \|_2^2$.

$= \frac{1}{N} \| [\hat{f}(\underline{x}) \; \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T \underline{w}' - y_N \|_2^2$.

Every convex function is continuous and differentiable. Clearly, $J(\underline{w}')$ is convex since it satisfies the Hessian double differential check. The Hessian is positive semi-definite everywhere.

So, $J(\underline{w}, \underline{w}')$ ~~$J(\underline{w}')$~~ is continuous and differentiable in terms of $\underline{w}'$ and $\underline{w}$. //

(h) Given, $\hat{f}(\underline{x}) = \underline{w}^T \underline{x}$.

~~$J(\underline{w}) = \frac{1}{N} \| \underline{w}^T \underline{x} - y_N \|_2^2$~~ .

$$J(\underline{w}, \underline{w}') = \frac{1}{N} \| [\hat{f}(\underline{x}), \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T \underline{w}' - y_N \|_2^2 .$$

Taking the gradient with respect to $\underline{w}'$,

$$\nabla_{\underline{w}'} J(\underline{w}, \underline{w}') = \frac{2}{N} \left([\hat{f}(\underline{x}) \; \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T \underline{w}' - y_N \right) \cdot [\hat{f}(\underline{x}) \; \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T$$

The weight update formula is as follows ; at iteration i :

$$\underline{w}'(i+1) = \underline{w}'(i) - \rho(i) \nabla_{\underline{w}'} J(\underline{w}, \underline{w}')$$

$$= \underline{w}'(i) - \rho(i) \cdot \frac{2}{N} \left([\hat{f}(\underline{x}) \; \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T \underline{w}' - y_N \right) \cdot [\hat{f}(\underline{x}) \; \hat{f}(\underline{x})^2 \; \hat{f}(\underline{x})^3]^T$$

where $\rho(i) \rightarrow$ learning rate update of $\underline{w}'$.

Taking the gradient wrt $\underline{w}$,

~~$\nabla_{\underline{w}} J(\underline{w}, \underline{w}') = \frac{1}{N} \| w_1' \underline{w}^T \underline{x} + w_2' (\underline{w}^T \underline{x})^2 + w_3' (\underline{w}^T \underline{x})^3 - y_N \|_2^2$.~~

$$\nabla_{\underline{w}} J(\underline{w}, \underline{w}') = \frac{2}{N} \left(w_1' \underline{w}^T \underline{x} + w_2' (\underline{w}^T \underline{x})^2 + w_3' (\underline{w}^T \underline{x})^3 - y_N \right) \times \left(w_1' \underline{x} + 2 w_2' (\underline{w}^T \underline{x}) \underline{x} + 3 w_3' (\underline{w}^T \underline{x})^2 \cdot \underline{x} \right)$$

$$= \frac{2}{N} \left(w_1' \hat{f}(\underline{x}) + w_2' (\hat{f}(\underline{x}))^2 + w_3' (\hat{f}(\underline{x}))^3 - y_N \right) \left(w_1' \underline{x} + 2 w_2' \hat{f}(\underline{x}) \underline{x} + 3 w_3' \hat{f}(\underline{x})^2 \underline{x} \right)$$

The weight update formula is as follow at iteration i:

$$\underline{w}(i+1) = \underline{w}(i) - \eta(i)\, \nabla_{\underline{w}} J(\underline{w}).$$

$$= w(i) - \eta(i)\frac{2}{N}\left(w_1' \hat{f}(\underline{x}) + w_2'(\hat{f}(\underline{x}))^2 + w_3'(\hat{f}(\underline{x}))^3 - y_n \right)\left(w_1'\underline{x} + 2w_2'\hat{f}(\underline{x})\,\underline{x} + 3w_3'\hat{f}(\underline{x})^2\,\underline{x} \right) //$$

(i) For approach A1, $J(\underline{w}') = \frac{1}{N} \| \underline{u}\underline{w}' - y_{\underline{N}} \|_2^2$

$$= \| \phi(\underline{x})\,\underline{w}' - y_{\underline{N}} \|_2^2$$

With L2 regularisation or by ridge ~~regression~~, regularization:

$$J(\underline{w}') = \frac{1}{N} \| \underline{u}\underline{w}' - y_{\underline{N}} \|_2^2 + \lambda \| \underline{w}' \|_2^2$$ where λ is the regularisation parameter.

λ is a penalty factor that controls the range of values of the weights. In this way, it prevents overfitting on the training set.

(j) For approach A2, $J(\underline{w}, \underline{w}') = \frac{1}{N} \| \hat{y}(\underline{x}) - y_N \|_2^2$

With L2 regularisation or ridge regularization:

$$J(\underline{w}, \underline{w}') = \frac{1}{N} \| \hat{y}(\underline{x}) - y_N \|_2^2 + \lambda \| \underline{w} \|_2^2 + \lambda' \| \underline{w}' \|_2^2 . //$$

→ λ' & λ' are the regularization parameters, to prevent overfitting.

(K) When there's plenty of data i.e N is more, approach A1 will be more suitable. The degrees of freedom in this case is roughly $D+1 = 4$. And when $N > 00f$, we obtain an over determined case and so, the error will be less on the unseen data. We will obtain a better fitted polynomial for the entire data.

(d) When the training data is limited, A2 is preferable. Here, the amount of data i.e N is less. The degrees of freedom of such a model is 7. Clearly, during training of the regression model, the no. of parameters to parameterize and tune are more and as gradient descent happens, the regressor begins to fit well to the data. This will result in the model to perform fairly well on the unseen data set. So, A2 is preferable even though it's an underdetermined case.