Homework 12

Runmin Lu

December 9, 2021

1

(a)

identity:

$$\frac{\delta e}{\delta W_1} \approx \begin{pmatrix} -0.01938197 & -0.01938197 \\ -0.01938197 & -0.01938197 \\ -0.03876394 & -0.03876394 \end{pmatrix}$$

$$\frac{\delta e}{\delta W_2} \approx \begin{pmatrix} -0.18460146 \\ -0.14059139 \\ -0.14059139 \end{pmatrix}$$

tanh:

$$\frac{\delta e}{\delta W_1} \approx \begin{pmatrix} -0.01594156 & -0.01594156 \\ -0.01594156 & -0.01594156 \\ -0.03188311 & -0.03188311 \end{pmatrix}$$

$$\frac{\delta e}{\delta W_2} \approx \begin{pmatrix} -0.15183362 \\ -0.1156356 \\ -0.1156356 \end{pmatrix}$$

(b)

identity:

$$\frac{\delta e}{\delta W_1} \approx \begin{pmatrix} -0.01938022 & -0.01938022 \\ -0.01938022 & -0.01938022 \\ -0.03875693 & -0.03875693 \end{pmatrix}$$

$$\frac{\delta e}{\delta W_2} \approx \begin{pmatrix} -0.18457646 \\ -0.14057689 \\ -0.14057689 \end{pmatrix}$$

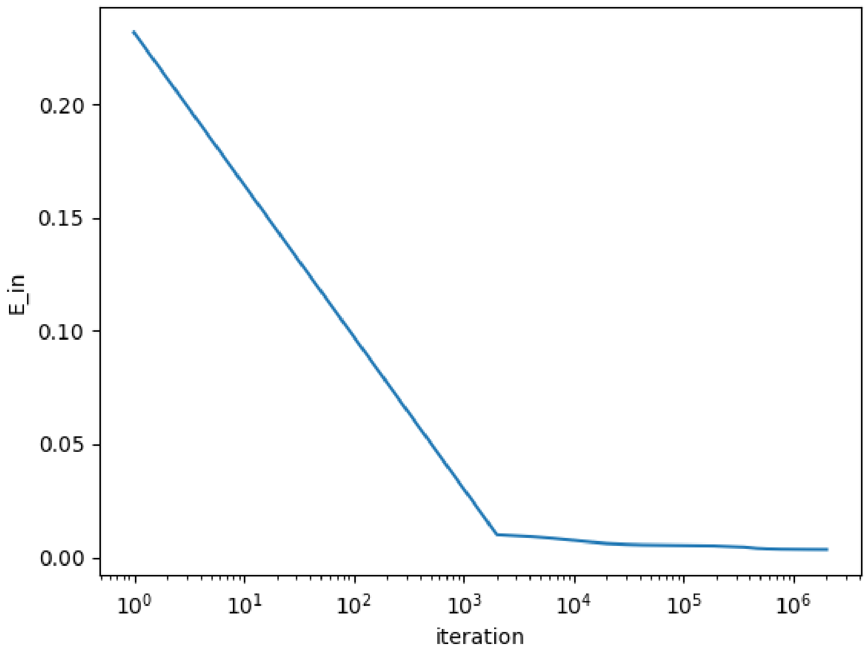
tanh:

$$\frac{\delta e}{\delta W_1} \approx \begin{pmatrix} -0.01594012 & -0.01594012 \\ -0.01594012 & -0.01594012 \\ -0.03187736 & -0.03187736 \end{pmatrix}$$

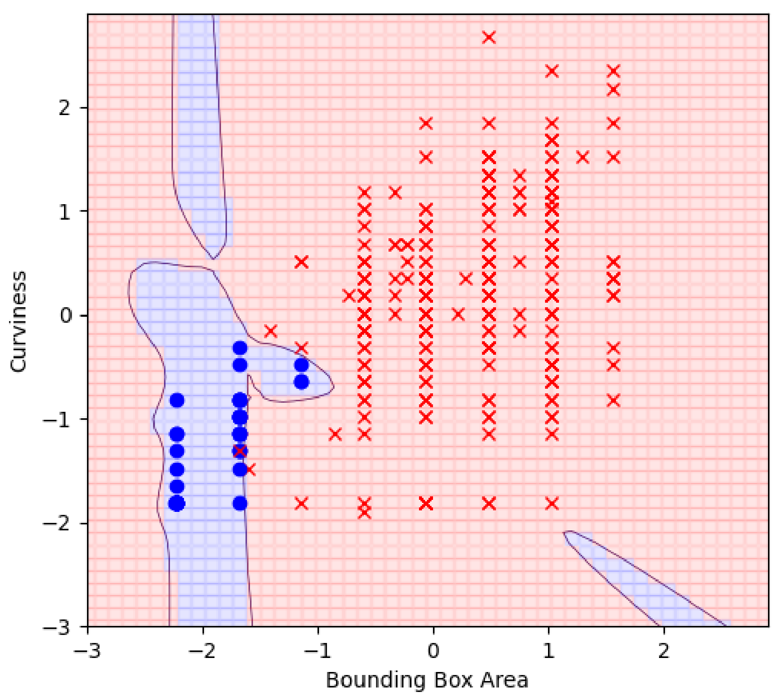
$$\frac{\delta e}{\delta W_2} \approx \begin{pmatrix} -0.15181331 \\ -0.11562382 \\ -0.11562382 \end{pmatrix}$$

2

(a)

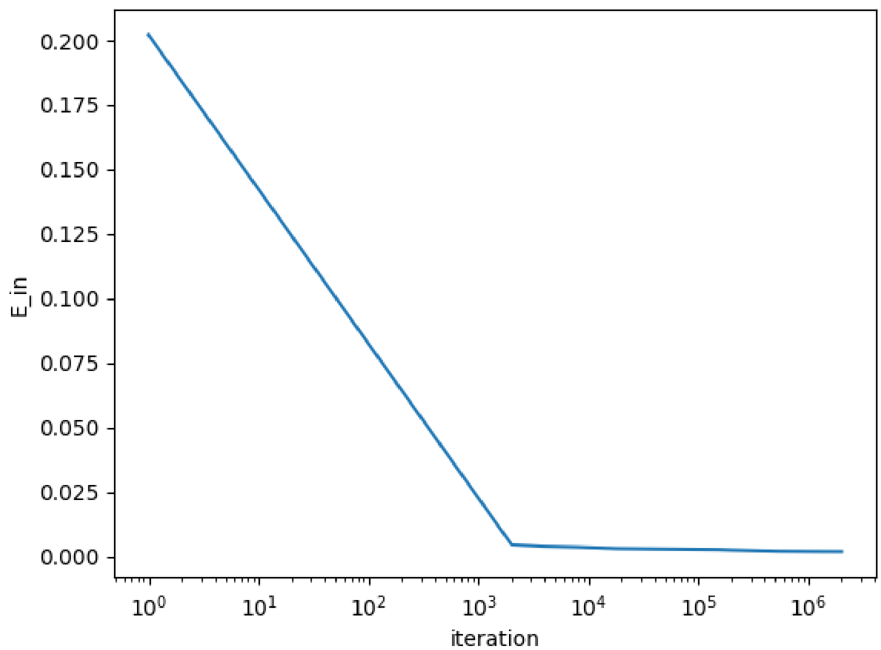


E_in
0.20
0.15
0.10
0.05
0.00
10^0
10^1
10^2
10^3
10^4
10^5
10^6
iteration

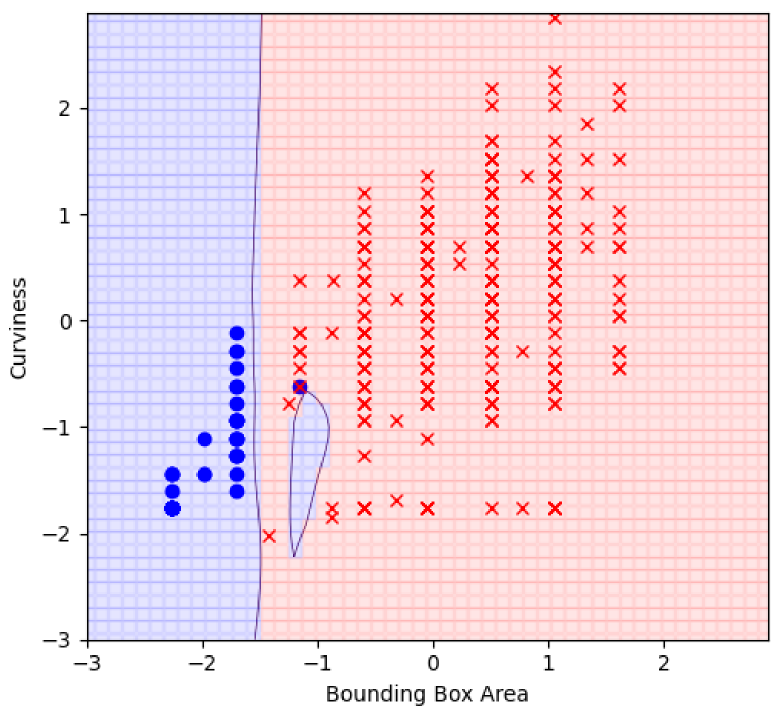


Curviness
2
1
0
−1
−2
−3
−3
−2
−1
0
1
2
Bounding Box Area

(b)



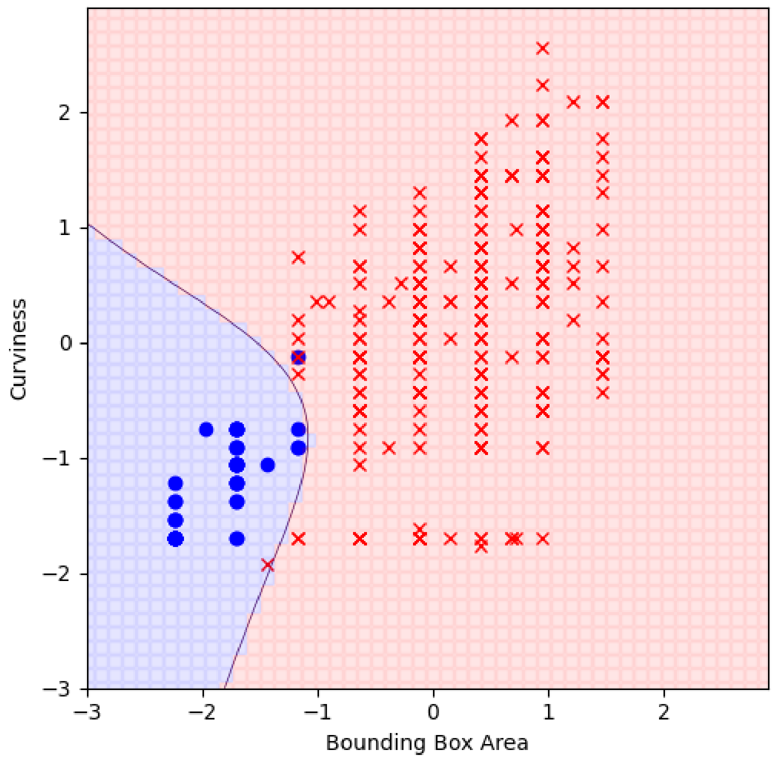
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
E_in
10^0 10^1 10^2 10^3 10^4 10^5 10^6
iteration



Curviness
2
1
0
−1
−2
−3
−3 −2 −1 0 1 2
Bounding Box Area

(c)

Since E_{in} reduces much more slowly after about the 2000th iteration, I inferred that iterations after that will overfit. Therefore, from a list of $100x$ for integer $x \in [1, 20]$, the optimal number of iterations is 100 with $E_{\text{test}} \approx 0.0139$.



3

(a)

Let the optimal hyperplane be $ax + by + c = 0$. Then the distance to $\mathbf{x}_1$ is $\frac{|a+c|}{\sqrt{a^2+b^2}}$ and the distance to $\mathbf{x}_2$ is $\frac{|-a+c|}{\sqrt{a^2+b^2}}$. To maximize those values, we want to minimize the denominator, so we set $b = 0$ and the hyperplane becomes a vertical line.

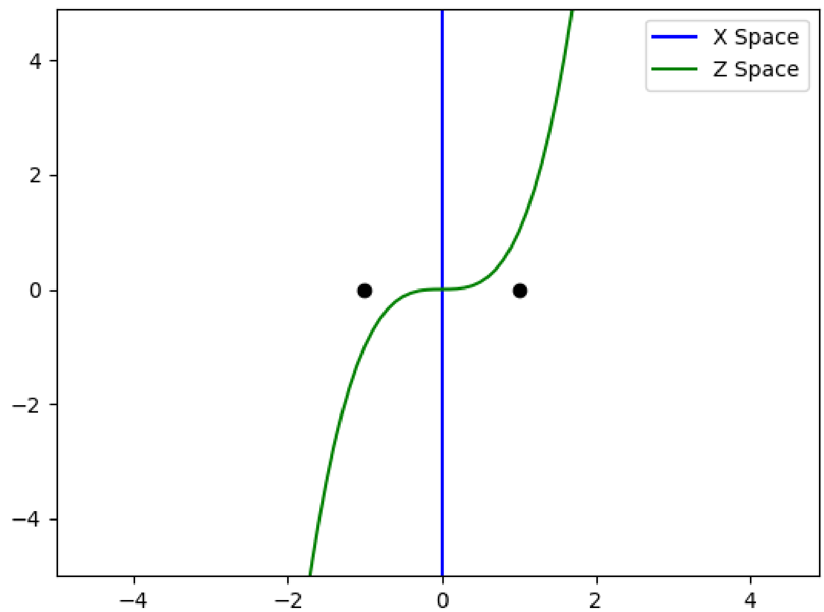
Since $\mathbf{x}_1$ and $\mathbf{x}_2$ must be on different sides of the line, to maximize the minimum of the distances, we set $\frac{c}{a} = 0$. Therefore, $c = 0$ and $a \neq 0$. WLOG let $a = 1$ and the corresponding line becomes $x_1 = 0$, which is in fact their perpendicular bisector.

To make the right side $+1$, we have $\mathbf{w} = (1, 0), b = 0$.

(b)

$\mathbf{z}_1 = (1, 0), \mathbf{z}_2 = (-1, 0)$. Since the coordinates are the same, the perpendicular bisector is the same: $z_1 = 0$. Optimal hyperplane: $\mathbf{w} = (1, 0), b = 0$.

(c)



(d)

$$
\begin{aligned}
K(\mathbf{x}, \mathbf{y}) &= \mathbf{z}(\mathbf{x}) \cdot \mathbf{z}(\mathbf{y}) \\
&= \begin{pmatrix} x_1^3 - x_2 \\ x_1 x_2 \end{pmatrix} \cdot \begin{pmatrix} y_1^3 - y_2 \\ y_1 y_2 \end{pmatrix} \\
&= \boxed{(x_1^3 - x_2)(y_1^3 - y_2) + x_1 x_2 y_1 y_2}
\end{aligned}
$$

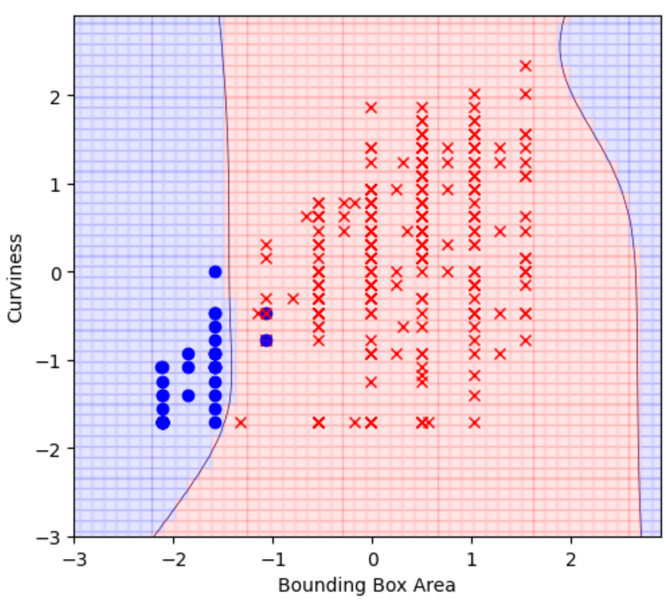
(e)

$$
\begin{aligned}
g(\mathbf{x}) &= \text{sign}(\mathbf{w} \cdot \mathbf{z}(\mathbf{x}) + b) \\
\mathbf{w} &= \alpha_1 y_1 \mathbf{z}(\mathbf{x}_1) + \alpha_2 y_2 \mathbf{z}(\mathbf{x}_2) \\
(1, 0) &= \alpha_1 (1, 0) - \alpha_2(-1, 0) \\
1 &= \alpha_1 + \alpha_2 \\
\text{Dual Constraint: } 0 &= \alpha_1 y_1 + \alpha_2 y_2 \\
&= \alpha_1 - \alpha_2 \\
\alpha_1 &= \frac{1}{2} \\
\alpha_2 &= \frac{1}{2} \\
g(\mathbf{x}) &= \text{sign}((\frac{1}{2}\mathbf{z}(\mathbf{x}_1) - \frac{1}{2}\mathbf{z}(\mathbf{x}_2)) \cdot \mathbf{z}(\mathbf{x}) + b) \\
&= \boxed{\text{sign}(\frac{1}{2}K(\mathbf{x}_1, \mathbf{x}) - \frac{1}{2}K(\mathbf{x}_2, \mathbf{x}))}
\end{aligned}
$$

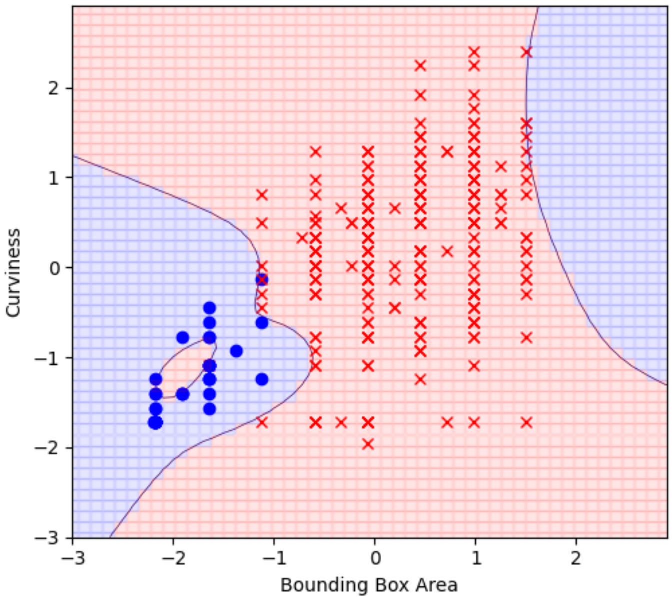
4

(a)

Small $C = 0.0001$:



Large $C = 1000$:



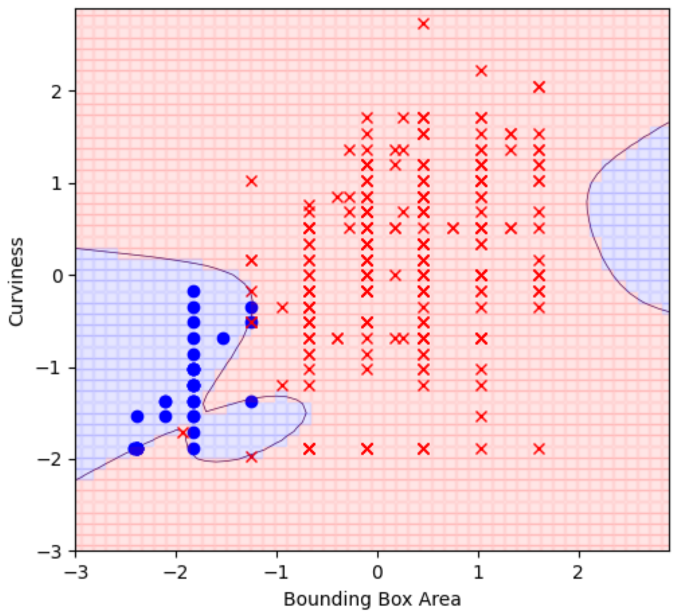
(b)

Since the quadratic program has constraints: $\forall i : 0 \leq \alpha_i \leq C$, as C gets smaller, each α_i is more constrained and the feasible regions becomes smaller, which corresponds to more regularization.

It can be seen from the plots that the decision boundary resulted from large C is much more curvy and complex. Although both models overfit with the right blue region, the one with large C has a really unnecessary red hole on the lower left.

(c)

From a list of 2^x for integer $x \in [-10, 10]$, the optimal $C = 1$ with $E_{\text{test}} \approx 0.0180$



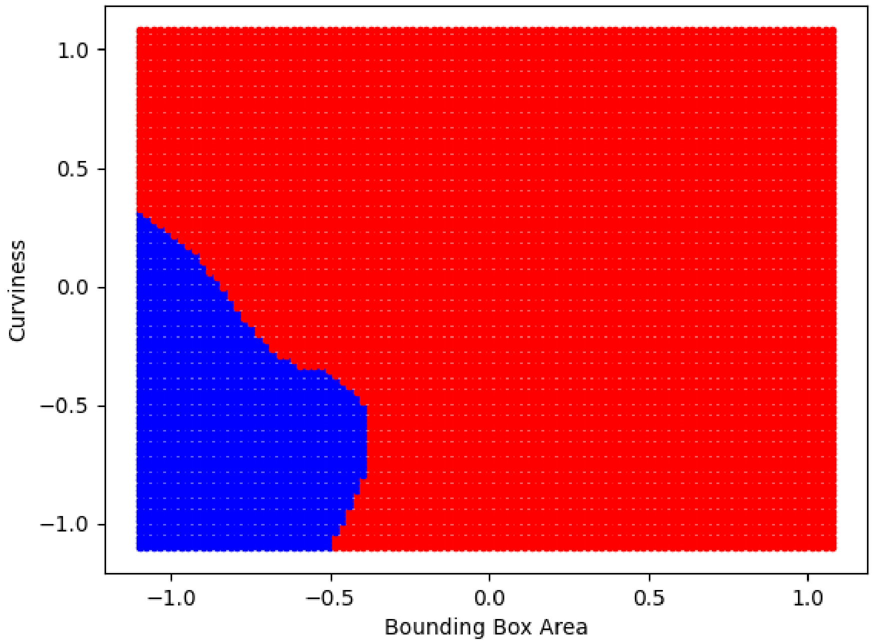
5

Model	E_{test}
Linear	0.0111
k-NN	0.0127
RBF	0.0139
Neural Network	0.0139
SVM	0.0180

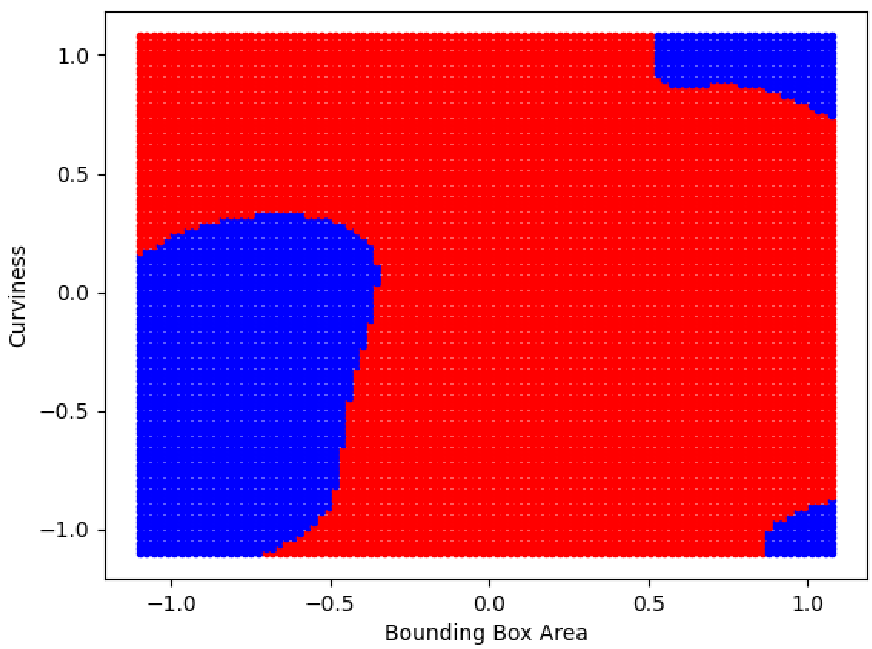
If we compare models only based on E_{test}, then we would pick the linear model as the best one. However, that shouldn't be the only factor, especially when these E_{test}'s are close because they can be above or below E_{out} by a factor of $\frac{1}{\sqrt{K}}$ where K is the number of test points. If we examine more, we can see that the decision boundaries of k-NN (shown below) and neural network are much less complex than other models (shown below), which all have some blue regions on the right. Therefore, k-NN and neural network overfit the least and are two of the best models for this problem.

In addition, if we think intuitively, SVM is generally not a good model for inseparable data, which defeats the whole purpose of SVM (maximum margin). This explains why it has the largest E_{test} among all the models.

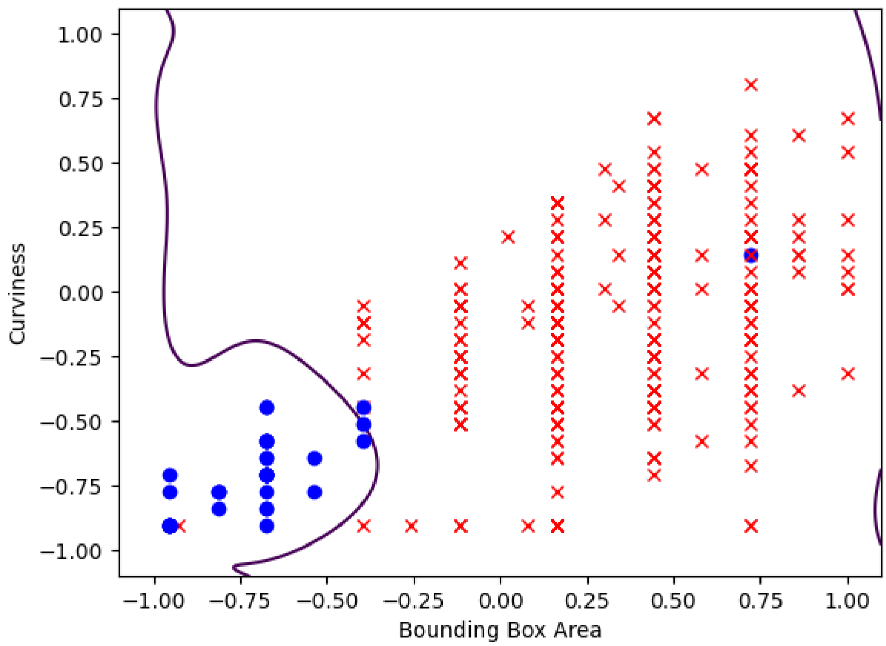
k-NN:


1.0
0.5
0.0
−0.5
−1.0
Curviness
−1.0
−0.5
0.0
0.5
1.0
Bounding Box Area

RBF:


1.0
0.5
0.0
−0.5
−1.0
Curviness
−1.0
−0.5
0.0
0.5
1.0
Bounding Box Area

Linear:


1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
Curviness
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Bounding Box Area