Stat365/665 (Spring 2015) Data Mining and Machine Learning

STATS 665 Homework 2

Name: Leon Lixing Yu *In Collaborate with: Faisal Faisal Zaghloul, Junaid Nomani*

1 Problem 1

1.1 closed form of $\hat{\beta}$

$\hat{\beta}$ is given by:

$$\hat{\beta} = \underset{\beta}{\text{argmin}} \frac{1}{2}\|X\beta - y\|_2^2 + \lambda\|\beta\|_2^2$$

In order to find *argmin*, we need to take the derivative of the loss function and make it equal to 0. In this assignment, I will use the notation, $x_j^{(i)}$, to denote that for the given data $(x^{(i)}, y^{(i)})_{i=1}^n$, $x_j^{(i)}$ is the jth element of $x^{(i)}$, where $x^{(i)} \in \mathbb{R}^d, y \in \mathbb{R}$, $X \in \mathbb{R}^{d\times n}, Y \in \mathbb{R}^n$. Therefore, we have:

$$\frac{d}{d\beta_j} l(\beta|X,Y) = ((1/2)\sum_{i=1}^n((\sum_{j=1}^d x_j^{(i)}\beta_j) - y^{(i)})^2 + \lambda\sum_{j=1}^d \beta_j^2)' = 0$$

$$\Rightarrow 2X^T(X\beta - Y) + \lambda(I_d)\beta = 0$$

$$\Rightarrow \hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}X^TY$$

1.2 Find a simple expression for $\|\hat{\beta} - \beta^*\|$

If we plug $Y = X\beta^* + w$ into $\hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}X^TY$, we get:

$$\hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}X^T(X\beta^* + w)$$

$$\Rightarrow \hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}(X^TX\beta^* + X^Tw)$$

$$\Rightarrow \hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}X^TX\beta^*) + (X^TX + 2\lambda(I_d))^{-1}X^Tw$$

$$\Rightarrow \hat{\beta} = (X^TX + 2\lambda(I_d))^{-1}(X^TX\beta^* + 2\lambda(I_d) - 2\lambda(I_d)) + (X^TX + 2\lambda(I_d))^{-1}X^Tw$$

$$\Rightarrow \hat{\beta} = ((X^TX + 2\lambda(I_d))^{-1}(X^TX\beta^* + 2\lambda(I_d)) - ((X^TX + 2\lambda(I_d))^{-1}(2\lambda(I_d)) + (X^TX + 2\lambda(I_d))^{-1}X^Tw$$

$$\Rightarrow \hat{\beta} = ((I_d) - (X^TX + 2\lambda(I_d))^{-1}(2\lambda(I_d)))\beta^* + (X^TX + 2\lambda(I_d))^{-1}X^Tw$$

$$\Rightarrow \hat{\beta} = \beta^* - (X^TX + 2\lambda(I_d))^{-1}(2\lambda(I_d))\beta^* + (X^TX + 2\lambda(I_d))^{-1}X^Tw$$

plug this form back into $\|\hat{\beta} - \beta^*\|$:

$$\|\hat{\beta} - \beta^*\| = \| - (X^TX + 2\lambda(I_d))^{-1}(2\lambda(I_d)))\beta^* + (X^TX + 2\lambda(I_d))^{-1}X^Tw\|$$

$$= \| - (X^TX + 2\lambda(I_d)^{-1})((2\lambda(I_d))\beta^* - X^Tw)\|$$

1.3 Find a closed form of $\hat{f}$

Notation Note: Following the previous convention, I am using $(x^{(i)}, y^{(i)})_{i=1}^n$ instead of $(x_i, y_i)_{i=1}^n$ to represent the dataset. In this question, I use $f(x^{(i)})$ and $< f, \phi(x^{(i)}) >$ interchangably. We know that:

$$f \in \mathcal{H} \Rightarrow \quad f(x^{(i)}) \in \mathcal{H} \Rightarrow \quad < f, \phi(x^{(i)}) > \in \mathcal{H}$$

With the notation given, we have equation:

$$\hat{f} = \underset{f}{\operatorname{argmin}}(1/2n)\sum_{i=1}^{n}(y^{(i)} - < f, \phi(x^{(i)}) >_{\mathcal{H}})^2 + \lambda\|f\|_{\mathcal{H}}^2$$

Using representer theorm, we have:

$$f = \sum_{i=1}^{n}\alpha^{(i)}\phi(x^{(i)}) = \sum_{i=1}^{n}\alpha^{(i)}k(x^{(i)}, \cdot)$$

The equation above means f is a linear combination of feature space, mapping of points. substitute the relation above into the original $\hat{f}$ equation, we have:

$$(1/2n)\sum_{i=1}^{n}(y^{(i)} - < f, \phi(x^{(i)}) >_{\mathcal{H}})^2 + \lambda\|f\|_{\mathcal{H}}^2 = (1/2n)\|Y - K\boldsymbol{\alpha}\|^2 + \lambda\boldsymbol{\alpha}^T K\boldsymbol{\alpha}$$

Taking derivative over α and make it equal to 0 to get $\underset{\alpha}{\operatorname{argmin}}$:

$$\frac{d}{d\alpha}((1/2n)\|Y - K\boldsymbol{\alpha}\|^2 + \lambda\boldsymbol{\alpha}^T K\boldsymbol{\alpha}) = 0$$

$$\Rightarrow (1/2n)2K(Y - K\boldsymbol{\alpha}) + 2\lambda\mathbf{I_d}K\boldsymbol{\alpha} = 0$$

$$\Rightarrow (K + 2n\lambda\mathbf{I_d})\boldsymbol{\alpha} = Y$$

$$\Rightarrow \hat{\boldsymbol{\alpha}} = (K + 2n\lambda\mathbf{I_d})^{-1}Y$$

recall:

$$f = \sum_{i=1}^{n}\alpha^{(i)}\phi(x^{(i)}) = \sum_{i=1}^{n}\alpha^{(i)}k(x^{(i)}, \cdot)$$

our $\hat{f}$ is then:

$$\hat{f} = K^T\hat{\boldsymbol{\alpha}} = K^T(K + 2n\lambda\mathbf{I_d})^{-1}Y$$

In which K is the Kernel matrix W.R.T X

1.4 implement the solution of $\hat{f}$ in matlab

The code for this part is attached in **Appendix A: problem 1 Code**, I use Gaussian Kernel because it is the first one I tried and it worked pretty well. I have attached a few graphs to show the differences between real label value and decision values got from $\hat{f}$.
To have a perfect fit, I adujust the values of λ and σ. Multiple attempts are shown below with descriptions.

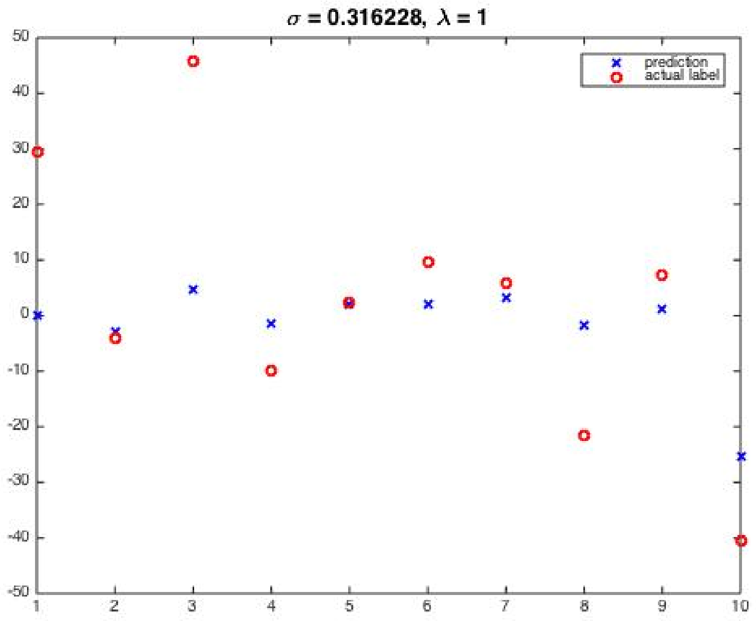

Figure 1: Red circle indicates labels, blue cross is the $\hat{f}$ value. $n = 10, \lambda = 1, \sigma^2 = 0.1$ in this test case. I see that when $x^{(i)}$ is close to decision boundary, 0, it is more accurate in this setting.It is an underfit case.

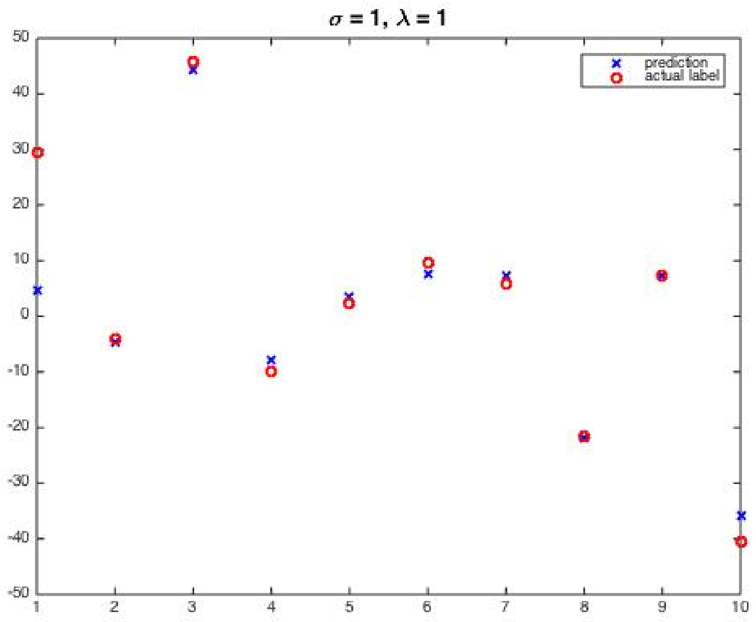

Figure 2: $\lambda = 1, \sigma = 1$ in this test case. I see that most of the $x^{(i)}$ can be predicted accurately with two exceptions at the end of the decision boundary

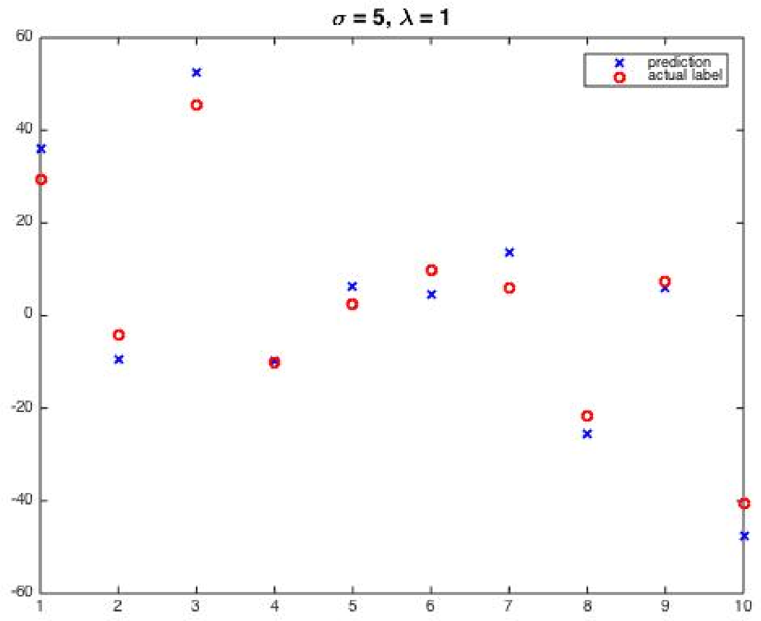

Figure 3: $\lambda = 1, \sigma = 5$ in this test case. Though the $x^{(i)}$ at both ends are better predicted, the average accuracy has dropped.

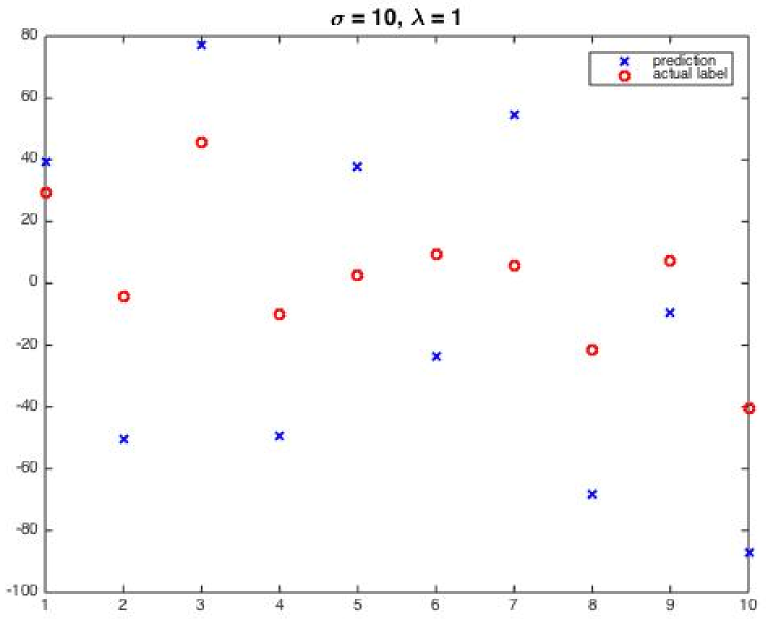

Figure 4: $\lambda = 1, \sigma = 10$. All $x^{(i)}$ are fitted badly. It is clearly a case of over fitting.

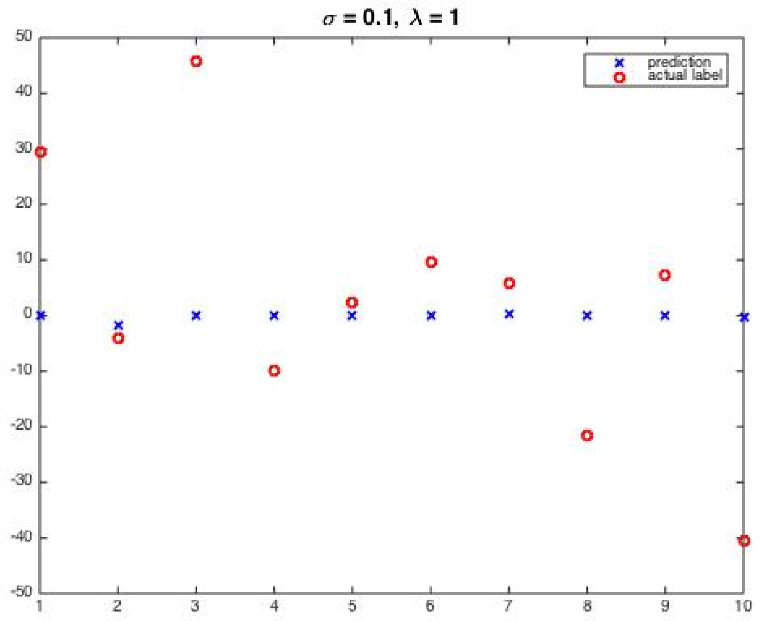

Figure 5: $\lambda = 1, \sigma = 0.1$. It is clearly a case of under fit. It is way under fit so that the decision boundary looks like a straight line.

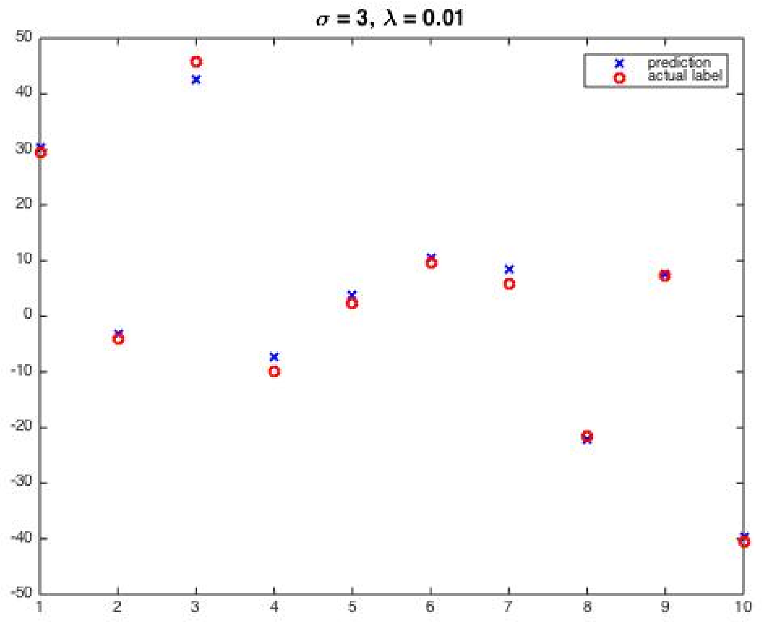

Figure 6: $\lambda = 0.01, \sigma = 3$. Most of the predictions for $x^{(i)}$ tends to overlap with their labels or be really close to their actual labels. I consider this is a good fit for the test data.

2 Problem 2

2.1 Heatmap of learned function

The Matlab code for this problem is attached in **Appendix B: problem 2 code**. I use *libsvm* library for this problem. I fixed the σ to 1. The heatmap is attached below for both training dataset and testing dataset.

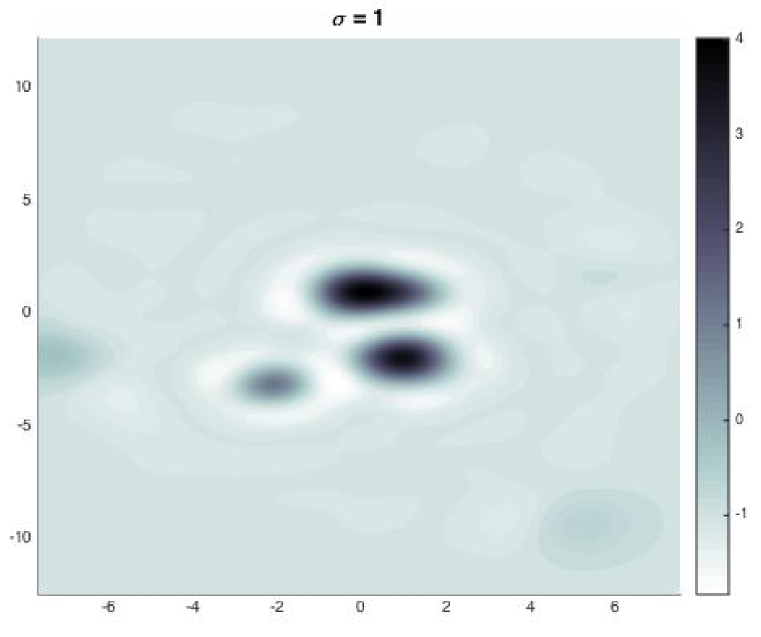

Figure 7: $\sigma = 1$. This heatmap is for training dataset,we can see the blurred decision boundary based on the heatmap, in which the color band spans from lightest to darkest with respective to decsision value spanning from -1 to 1 f value. The same convention applies to the rest of the heat map

2.2 Level curves

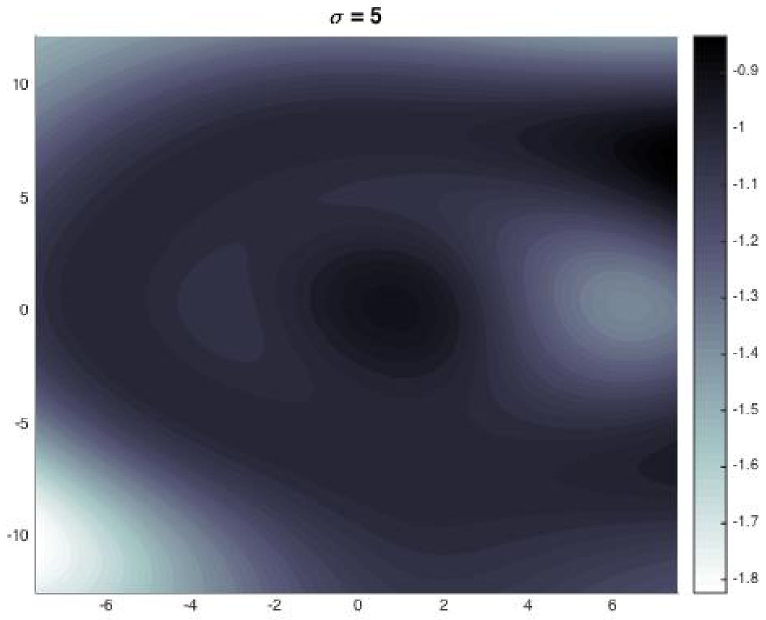

Figure 8: When $\sigma = 5$. we can see that most of the regions are way dark, meaning most of the decision values are positive and rather close to 1. It is clearly a case of underfitting

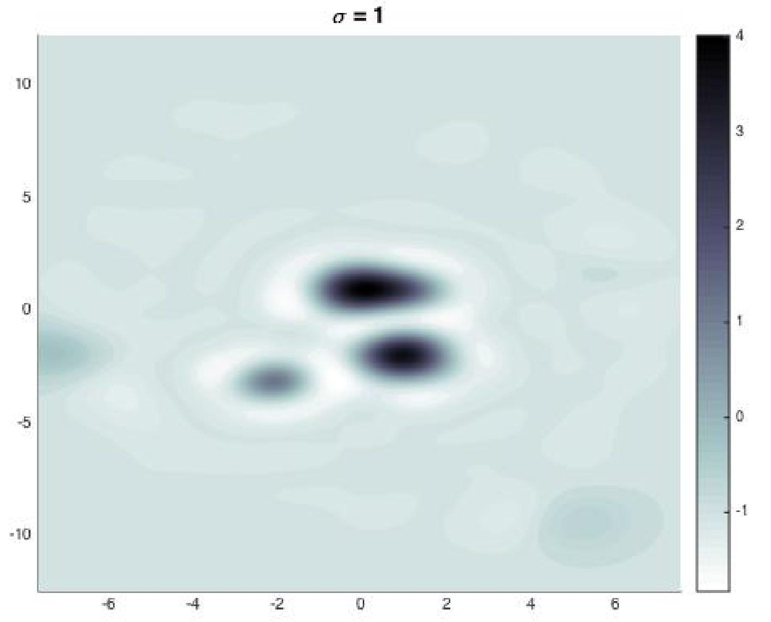

Figure 9: When $\sigma = 1$. we can see the blurred decision boundary based on the heatmap

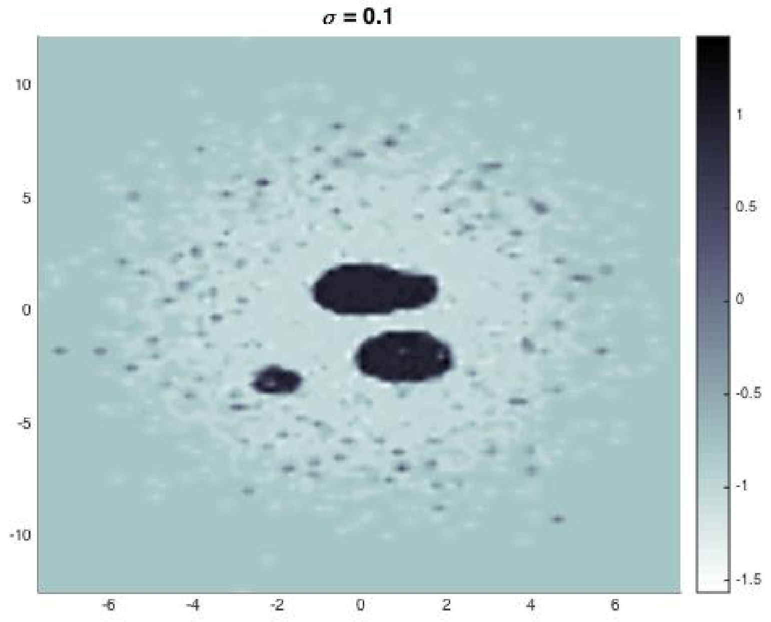

Figure 10: When $\sigma = 0.1$. The boundary between dark and light color became clear and the decision values are most accurate

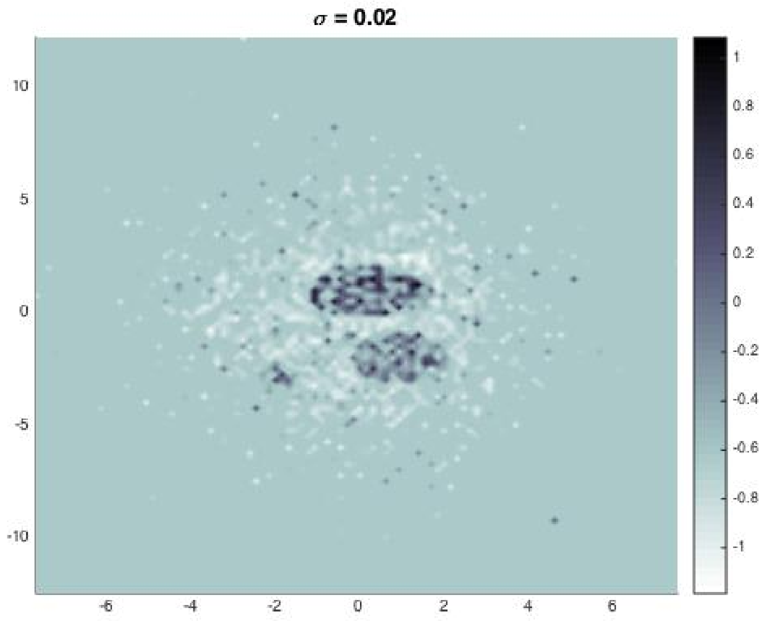

Figure 11: When $\sigma = 0.02$. The darker region previously seen became a collection of darker dots, and the lighter regions shows lighter dots. This can be catagorized as overfitting since the f values are rather training set specific.

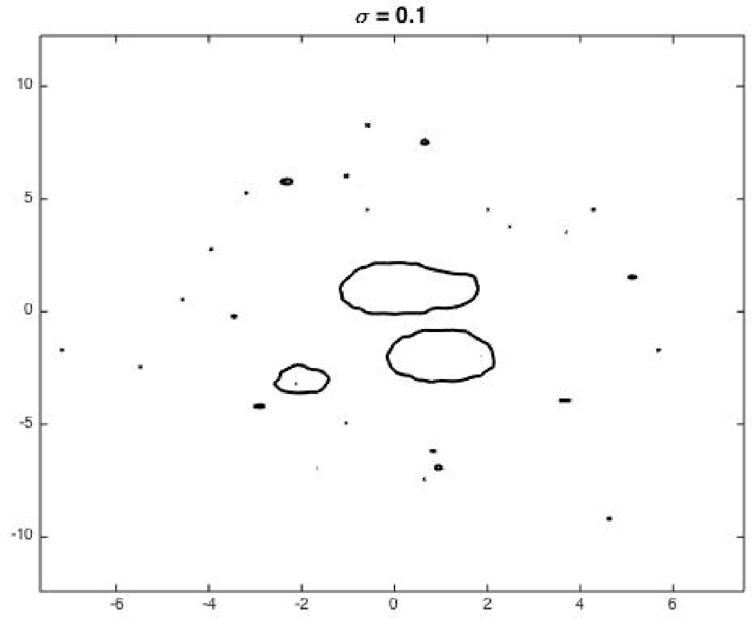

Figure 12: $\sigma = 0.1$ level curve of $\hat{f} = 0$, we can see that this setting is properly fitted with neighbourhood clearly defined.

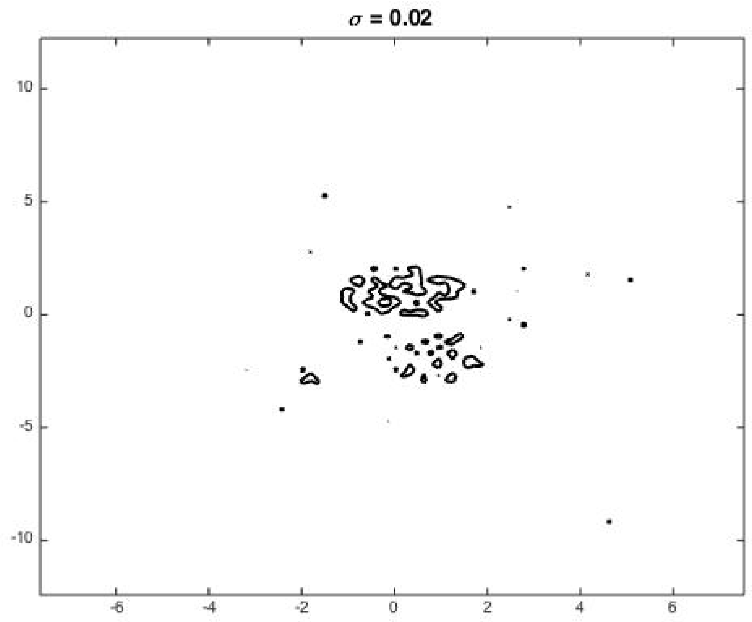

Figure 13: $\sigma = 0.02$ level curve of $\hat{f} = 0$,we see that the it is clearly a case of overftting as the neighbourhood became smaller and only is tailered to the training dataset.

The level curve for the underfitting case cannot be displayed since all of the prediction are positive number. When $\sigma = 5$, level curve$\hat{f} = 0$, all of the decision values are on one size of $\hat{f} = 0$ curve.

2.3 Plot the training and testing error vs $1/\sigma$

The plot is shown in the figure below.

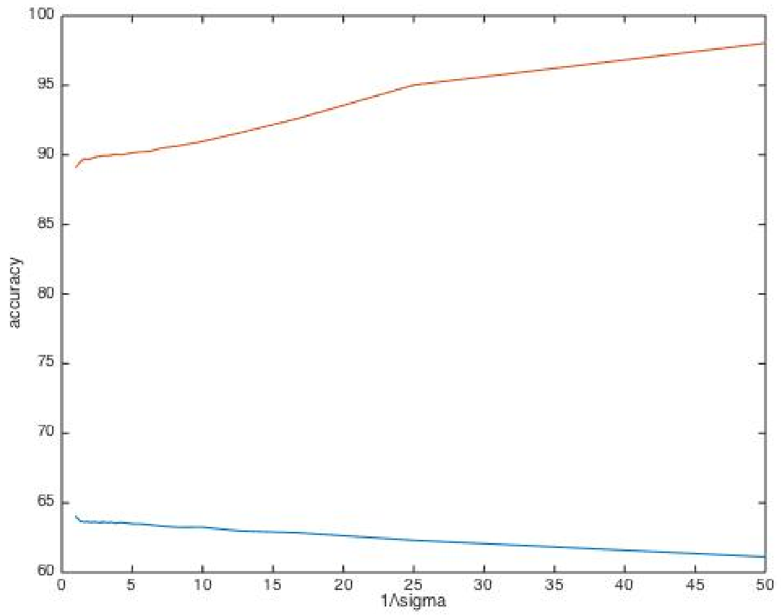

Figure 14: $\sigma \in [1, 0.01)$ with step size -0.02. X-axis shows $1/\sigma$, and y-axis shows the accuracy in %, we see that even though the training error keeps reducing with smaller σ value, the testing results is not improving. This is because while $\sigma \to 0$, the decision boundary is overfitted onto each data point, $x_{i}{}_{i=1}^{n}, n = 10,000$. The testing dataset cannot benefit from the overfitting of training dataset.

3 Problem 3

3.1 Show $k(x, y)$ is a valid kernel

We know that kernel is a function that maps $\chi \times \chi \to \mathbb{R}$. We also know that kernel is valid if and only if for $x_{i}{}_{i=1}^{n} \in \chi, K_{ij} = k(x_i, x_j)$ is PSD. We need to prove these two statements.
Proving $k(x, y)$ maps $\chi \times \chi \to \mathbb{R}$:
Assuming:

$$k_1(x, y) =< \Phi_1(x), \Phi_1(y) >$$

$$k_2(x, y) =< \Phi_2(x), \Phi_2(y) >$$

Since $k_1(x, y)$ and $k_2(x, y)$ are vaild kernel, they both maps $\chi \times \chi \to \mathbb{R}$. we now have

$$k(x, y) =< \Phi_1(x), \Phi_1(y) > \cdot < \Phi_2(x), \Phi_2(y) >$$

$$\Rightarrow k(x,y) = (\Phi_1(x)^T \Phi_1(y)) \cdot (\Phi_2(x)^T \Phi_2(y))$$

$$\Rightarrow k(x,y) \in \mathbb{R}$$

We also know that $(x,y) \in \mathbb{R}^\chi$, so $k(x,y)$ maps $\chi \times \chi \to \mathbb{R}$.
Proving $k(x,y)$ is PSD:

$$K_{ij} = k(x_i, x_j) =< \Phi(x_i), \Phi(x_j) >= \Phi(x_i)^T \Phi(x_j)$$

$$\upsilon^T K \upsilon = \sum_i^n \sum_j^n \upsilon_i K_{ij} \upsilon_j$$

$$= \sum_i^n \sum_j^n \upsilon_i \Phi(x_i)^T \Phi(x_j) \upsilon_j$$

$$= \sum_i^n \sum_j^n \upsilon_i \sum_l^n \phi_l(x_i) \phi_l(x_j) \upsilon_j$$

$$= \sum_l^n \sum_i^n \sum_j^n \upsilon_i \phi_l(x_i) \phi_l(x_j) \upsilon_j$$

$$= \sum_l^n (\sum_i^n \upsilon_i \phi_l(x_i))^2 \geq 0$$

3.2 Prove RKHS

Based on fundamental theorem of Calculus, we have Taylor series formula:

$$g(x) = \sum_{l=0}^{k-1} \frac{g^{(l)}(0) x^l}{l!} + \int_0^x \frac{(x-y)^{k-1}}{(k-1)!} g^{(x)}(y) dy$$

With property of RKHS and the taylor series above, we have:

$$g(x) = k(x,y)$$

note that the k here stands for kernel instead of degree of deratives. and the kernel function must fit:

$$(\frac{d}{dy})^l k(x,y) = \frac{x^l}{l!} \quad while \quad y = 0, \quad and \quad l \in [1, 2, ...n-1]$$

$$(\frac{d}{dy})^n k(x,y) = \frac{(x-y)^{n-1}}{(k-1)!} \quad while \quad y \leq x$$

$$(\frac{d}{dy})^n k(x,y) = 0 \quad while \quad y > x$$

The function that meets these constraints is the function is the kernel function. This kernel function is solverable as polynomial function of n degrees is n times differentiable.

4 Problem 4

The code for this section is attached in **Appendix C: problem 4 code**.

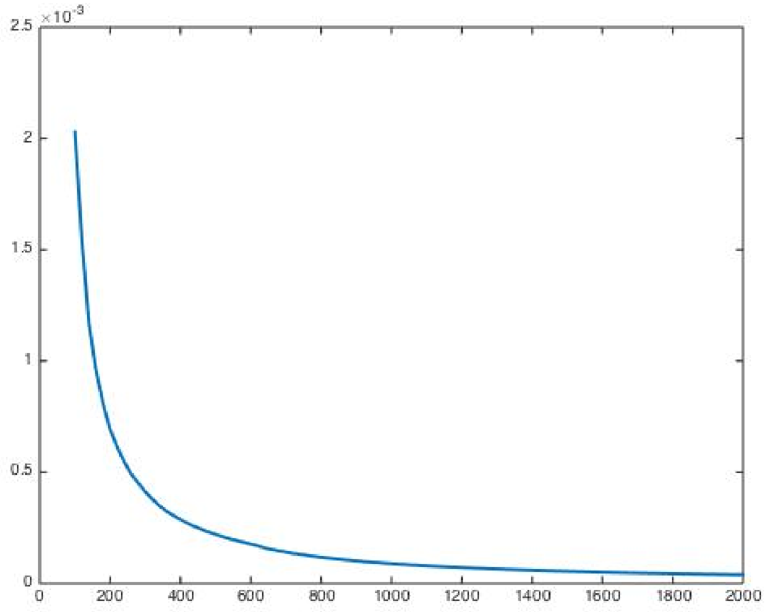

Figure 15: The $y - axis$ shows mean square error between $\hat{f}$ and f plot with the testing dataset. The $x - axis$ is the number of n. n is set from 100 to 2000 with step size of 20. The plot shows that when n is very small, the error is minimum. The MSE declines exponentially when we increate the training size n.

end of the story