IITM-CS5691 : Pattern Recognition and Machine Learning
Assignment 3

Release Date: May 18, 2021
Due Date: Jun 8, 2021 (11.59pm)

Roll No: EE19B132
Name: Sagnik Ghosh
Collaborators (if any):
References (if any):

- Use LaTeX to write-up your solutions (in the solution blocks of the source LaTeX file of this assignment), and submit the resulting single pdf file at GradeScope by the due date. (Note: **No late submissions** will be allowed, other than one-day late submission with 10% penalty or four-day late submission with 30% penalty! Within GradeScope, indicate the page number where your solution to each question starts, else we won't be able to grade it! You can join GradeScope using course entry code **5VDNKV**).
- For the programming question, please submit your code (rollno.ipynb file and rollno.py file in rollno.zip) directly in moodle, but provide your results/answers in the pdf file you upload to GradeScope.
- Collaboration is encouraged, but all write-ups must be done individually and independently, and mention your collaborator(s) if any. Same rules apply for codes written for any programming assignments (i.e., write your own code; we will run plagiarism checks on codes).
- If you have referred a book or any other online material for obtaining a solution, please cite the source. Again don't copy the source *as is* - you may use the source to understand the solution, but write-up the solution in your own words.
- Points will be awarded based on how clear, concise and rigorous your solutions are, and how correct your code is. Overall points for this assignment would be **min**(your score including bonus points scored, 50).

1. (10 points) [HYPER-TUNE, SET, GO...] Hyperparameters have a crucial impact on the performance of machine learning models, and there are several techniques to perform hyperparameter tuning.
   (a) (4 points) Cross-validation (CV) with grid search is a predominant technique for hyperparameter tuning that involves building models for each of the possible combinations of hyperparameter values, and the combination that gives the best performance in a CV setting is chosen.
      i. (3 points) Specify the hyperparameter(s) of any three regression/classification models seen in our class.

      Solution:

      - Linear Regression: Regularisation Parameter (λ)
      - Soft Margin SVM: Parameter C which controls the slackness or regularisation
      - Decision Trees: Tree depth

ii. (1 point) For any one of the hyperparameters from previous question, specify what criteria could be used to choose its range and values (over which to do grid-search on).

> **Solution:** The criteria that can be used to choose the regularisation parameter λ in Linear Regression is as follows:
> Choose a larger value of λ if the training error is low but the test error is comparatively high.
> Choose a lower value of λ if both train and test errors are high.

(b) (6 points) An alternate approach called Empirical Bayes (EB) can help estimate hyperparameters from the training data (without having to do CV). Let θ be the parameters of a model with prior distribution $p(\theta|\alpha)$ and X denote the training data. EB estimate of the hyperparameter α is simply the MLE that maximizes the marginal likelihood $p(X|\alpha) = \int_\theta p(X|\theta)p(\theta|\alpha)d\theta$. Let's look at toy problems that illustrates EB, which you can then take forward to understand EB of Bayesian logistic/linear regression beyond the course.

Consider a generative classification model with m classes, with our training data $X = \{X_i\}_{i=1,\dots,m}$ comprising one observation per class. That is,

$$X_i \mid \mu_i \overset{\text{indep}}{\sim} N(\mu_i, \sigma^2), \quad i = 1, \dots, m$$
$$\mu_i \overset{\text{iid}}{\sim} N(\phi, \tau^2), \quad \sigma^2, \tau^2 \text{ known.}$$

Now, answer the questions below.

[Hint: While you can derive the results from scratch, try using known results about marginal/-conditional (multivariate) Gaussians to avoid derivations and cite which result you are using.]

i. (2 points) Write down the marginal likelihood $p(X|\phi)$.

> **Solution:** We know,
> $$X_i \mid \mu_i \overset{\text{indep}}{\sim} N(\mu_i, \sigma^2), \quad i = 1, \dots, m$$
> $$X_i \mid \mu_i \overset{\text{indep}}{\sim} N(0, \sigma^2) + \mu_i$$
> and
> $$\mu_i \overset{\text{iid}}{\sim} N(\phi, \tau^2)$$
> Hence,
> $$X_i \mid \phi \overset{\text{indep}}{\sim} N(0, \sigma^2) + N(\phi, \tau^2)$$
> The above expression can be simplified using a result for Gaussians.
> $$X_i \mid \phi \overset{\text{indep}}{\sim} N(\phi, \sigma^2 + \tau^2)$$

ii. (2 points) Use the computed marginal likelihood to derive the hyperparameter's EB estimate, denoted $\widehat{\phi}$.

Solution: We know from the previous section that,

$$X_i \mid \phi \overset{\text{indep}}{\sim} N(\phi, \sigma^2 + \tau^2)$$

We also know that $\widehat{\phi}$ is the MLE of the above Likelihood. The MLE of the above Likelihood when σ and τ are known is:

$$\widehat{\phi} = \frac{1}{m} \sum_{i=1}^{m} X_i$$

iii. (2 points) What is the posterior mean of μ_i at this EB estimate (i.e., $E[\mu_i|X, \widehat{\phi}]$)? Show that it has an intuitive interpretation as the weighted average of X_i and $\bar{X}$.

Solution:

(c) (3 points) [OPTIONAL BONUS] Answer the same three questions above when you've n observations (instead of one) per class for a total of nm observations.

(d) (2 points) [OPTIONAL BONUS] How will your answer to the above question simplify if you've a density estimation problem, where there is only one class with n observations? That is, $X_j \mid \mu \overset{\text{iid}}{\sim} N(\mu, \sigma^2)$, $j = 1, ..., n$ with σ^2 known, and a prior distribution $\mu \sim N(\phi, \tau^2)$ with hyperparameter ϕ unknown and τ^2 known. Do you see any issues with EB in this question here, and can it be overcome with historical data (i.e., data collected prior to current data X)?

2. (10 points) [(LOGISTICALLY) CLASSIFIED INFORMATION]

(a) (3 points) Consider the following 2-dimensional (2D) binary classification dataset with 8 points given by:

$$X^T = \begin{bmatrix} -2 & -2 & -1 & -1 & 1 & 1 & 2 & 3 \\ -1 & 2 & 1 & 2 & 1 & 3 & 3 & 2 \end{bmatrix}$$

$$y^T = \begin{bmatrix} 1 & 1 & 1 & -1 & -1 & 1 & -1 & -1 \end{bmatrix}$$

Run one iteration of gradient descent with the logistic regression objective by hand (pen-and-paper). No bias required, only the 2D weight vector is to be optimized. Choose the step size $\eta = 1$. Initialise at $w = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$.

Solution: We know that the update step for logistic regression is as follows:

$$w^{(2)} = w^{(1)} + \eta \sum_{i=1}^{N} y_i x_i \sigma(-y_i w^T x_i)$$

where $w^{(1)}$ is the initialised weight vector, $w^{(2)}$ is the updated weight vector, η is the learning rate and $\sigma(.)$ is the sigmoid function. We are given that $w = [0\ 0]^T$, hence the term inside the sigmoid function for each term in the summation is zero and sigmoid of zero is 0.5. Hence,

$$w^{(2)} = 0.5 \sum_{i=1}^{N} y_i x_i$$

Simplifying this for the given values, we get,

$$w^{(2)} = [-\tfrac{9}{2} \ -\tfrac{3}{2}]^T$$

(b) (2 points) Plot the sigmoid function $\frac{1}{1+e^{-wX}}$ vs $X \in \mathbb{R}$ for $w \in [1, 5, 100]$. A qualitative sketch would also work. From these plots infer why logistic regression can lead to overfitting if the the weights are high.

Solution:

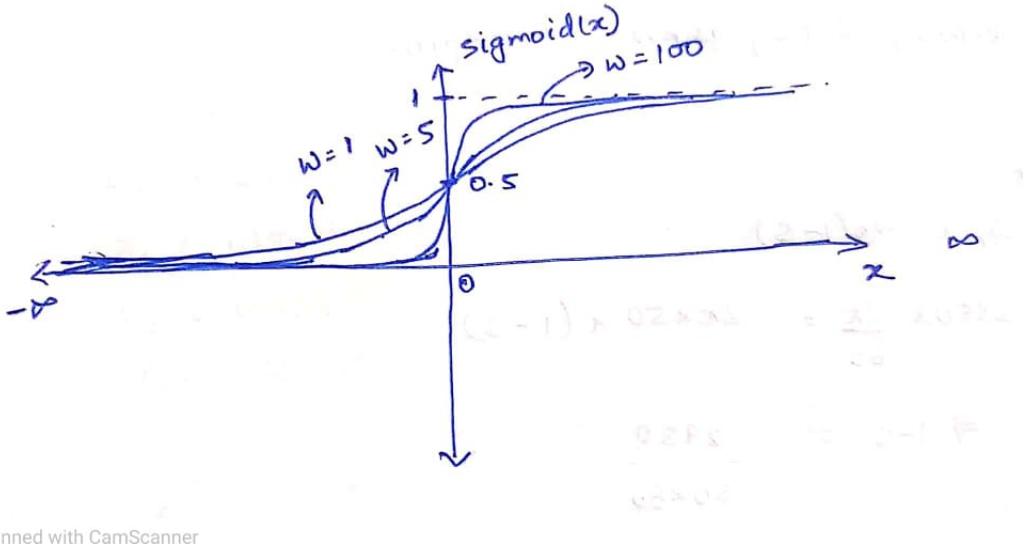

Figure 1: Sigmoid

As it can be seen from the above figure, for higher value of weights, the sigmoid function saturates faster and hence its gradient becomes very close to zero. During gradient descent, smaller gradients would lead to smaller update steps and that would increase the probability of the objective function getting stuck at a sub optimal minima, i.e., it is more prone to overfitting.

(c) (5 points) Let us look at multi-class logistic regression. Say we have K classes and each input x is a d-dimensional vector. The posterior probability (after ignoring the bias term) is given by:

$$P(Y = k \mid X = x) = \frac{e^{w_k^T x}}{\sum_{l=1}^{K} e^{w_l^T x}} \quad k = 1, 2 \cdots, K$$

i. (1 point) What are the parameters to estimate and how many are there?

Solution: The parameters to estimate are $\{w_k\}_{k=1,2,\ldots,K}$. There are K parameters to be estimated.

ii. (2 points) Given n training samples $\{(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)\}$, write down the log likelihood function, and simplify.

Solution: Log Likelihood = $\ln\left(\prod_{i=1}^{N} \frac{e^{w_{y_i}^T x_i}}{\sum_{l=1}^{K} e^{w_l^T x_i}}\right)$

Log Likelihood = $\sum_{i=1}^{n}[w_{y_i}^T x_i - \ln(\sum_{l=1}^{K} e^{w_l^T x_i})]$

iii. (2 points) Compute the gradient of the log likelihood with respect to w_k, and simplify.

Solution: $\nabla_{w_k} \ln(\text{Likelihood}) = \sum_{\{j:y_j=k\}} x_j - \sum_{i=1}^{n} \frac{x_i e^{w_k^T x_i}}{\sum_{i=1}^{K} e^{w_l^T x_i}}$

3. (10 points) [KERNELIZE...] Let K_1 and K_2 be a valid kernel functions, with feature mapping $\varphi_1 : \mathbb{R}^d \to \mathbb{R}^{d_1}$ and $\varphi_2 : \mathbb{R}^d \to \mathbb{R}^{d_2}$.

(a) (2 points) Show that $K_3 = K_1 + K_2$ is also a valid kernel. Give the feature mapping φ_3 corresponding to K_3 in terms of φ_1 and φ_2.

Solution: Let us construct a mapping φ_3 such that $K_3 = K_1 + K_2$ and $K_3 = \varphi_3(u)^T\varphi_3(v)$.
Let us claim that mapping φ_3 is such that it is $d_1 + d_2$ dimensions long and its first d_1 dimensions are equal to the mapping φ_1 and the remaining d_2 dimensions are equal to the mapping φ_2.
If we take the dot product $\varphi_3(u)^T\varphi_3(v)$ it is obvious that it is equal to $\varphi_1(u)^T\varphi_1(v) + \varphi_2(u)^T\varphi_2(v)$ or $K_3 = K_1 + K_2$.
As we have found a feature mapping corresponding to K_3, K_3 is a valid kernel.

(b) (2 points) Show that $K_4 = K_1 \cdot K_2$ is also a valid kernel. Give the feature mapping φ_4 corresponding to K_4 in terms of φ_1 and φ_2.

Solution: To show that K_4 is a valid kernel, let us find a feature mapping φ_4 such that $K_4 = \varphi_4^T\varphi_4$.
Let us assume that $\varphi_1(u) = [a_1(u)\ a_2(u)...a_{d_1}(u)]$ and $\varphi_2(u) = [b_1(u)\ b_2(u)...b_{d_2}(u)]$.
We need to find φ_4 suct that

$$\varphi_4(u)^T\varphi_4(v) = \varphi_1(u)^T\varphi_1(v).\varphi_2(u)^T\varphi_2(v)$$

If we multiply out the above expression we get,

$$\varphi_4(u)^T\varphi_4(v) = \sum_{i\in[1\ldots d_1], j\in[1\ldots d_2]} a_i(u)a_i(v)b_j(u)b_j(v)$$

The above expression can be obtained if we take $\varphi_4(u)$ as a vector containing all the elements in the set $\{a_i(u)b_j(u)\}_{\{i\in[1\ldots d_1],j\in[1\ldots d_2]\}}$.
As we have found a corresponding feature mapping $\varphi_4(u)$ for K_4, K_4 is a valid kernel.

(c) (2 points) Show that $K_5 = f(u)K_1(u,v)f(v)$ is also a valid kernel for any function $f : \mathbb{R}^d \to \mathbb{R}$. Give the feature mapping φ_5 corresponding to K_5 in terms of φ_1 and f.

Solution: We are given, $K_5 = f(u)K_1(u,v)f(v)$. This means that,

$$K_5 = f(u)\varphi_1(u)^T\varphi_1(v)f(v) = (f(u)\varphi_1(u))^T\varphi_1(v)f(v)$$

If we take the feature mapping corresponding to K_5, $\varphi_5(u) = f(u)\varphi_1(u)$, we get $K_5 = \varphi_5(u)^T\varphi_5(v)$. As we have found a corresponding feature mapping, K_5 is a valid kernel.

(d) (2 points) Show that a Kernel given by $K(u,v) = \exp(2u^Tv)$ is a valid kernel. [Hint: Use the results above on a polynomial expansion of $exp(t)$.]

Solution: We are given that $K(u,v) = \exp(2u^Tv)$. This can be written as follows:

$$K(u,v) = 1 + (2u^Tv) + \tfrac{(2u^Tv)^2}{2!} + \ldots$$

We know that $(2u^Tv)$ is a valid kernel by taking its feature map as $\sqrt{2}u$.
We also know that the product of any two kernels is a valid kernel (proved in a previous section). This result can be extended by induction to the case of product of multiple kernels. Hence, $(2u^Tv)^n$ is also a valid kernel for any natural number n.
We also know that the sum of any two kernels is a valid kernel. This can also be extended to the case of sum of multiple kernels.
Using the above results, it is clear that K is a valid kernel.

(e) (2 points) Show that a Kernel given by $K(u,v) = \exp(-\|u-v\|^2)$ is a valid kernel. [Hint: Use last two parts' results.]

Solution: We are given $K(u,v) = \exp(-\|u-v\|^2)$. This can be written as:

$$K(u,v) = \exp(-u^Tu + 2u^Tv - v^Tv) = \exp(-u^Tu)\exp(2u^Tv)\exp(-v^Tv)$$

The above equation can be written in the form $K(u,v) = f(u)\exp(2u^Tv)f(v)$.
We know $\exp(2u^Tv)$ is a valid kernel from the previous section and we also know that $f(u)Kf(v)$ is a valid kernel from a previous section.
Hence, $K(u,v) = \exp(-u^Tu)\exp(2u^Tv)\exp(-v^Tv)$ is a valid kernel.

4. (10 points) [YESS, VEE, EMM...] Consider a soft margin SVM with the regularization parameter C. For the dataset $\{x_i, y_i\}_{i=1}^n$ such that $x_i \in \mathbb{R}^d$ and $y_i \in \{-1,+1\}$, let $\alpha^* \in \mathbb{R}^n$ be the optimal dual

solution and $w^* \in \mathbb{R}^d, b^* \in \mathbb{R}, \epsilon^* \in \mathbb{R}^n$ be the optimal primal solution.

(a) (3 points) Compute the range of α_i. Say each $\alpha_i \in [l_i, u_i]$. Compute the range of ${w^*}^T x_i + b^*$ when 1) $\alpha_i = l_i$, 2) $\alpha_i = u_i$, or 3) $l_i < \alpha_i < u_i$.

Solution: Each α_i belongs to $[0, C]$.
When $\alpha_i = l_i$, ${w^*}^T x_i + b^* >= 1$ when $y_i = +1$ and ${w^*}^T x_i + b^* <= -1$ when $y_i = -1$.
When $\alpha_i = u_i$, ${w^*}^T x_i + b^* <= 1$ when $y_i = +1$ and ${w^*}^T x_i + b^* >= -1$ when $y_i = -1$.
When $< l_i \alpha_i < u_i$, ${w^*}^T x_i + b^* = 1$ when $y_i = +1$ and ${w^*}^T x_i + b^* = -1$ when $y_i = -1$.

(b) (3 points) Explain what would happen if C is 1) negative, 2) zero, or 3) infinity.

Solution: When C is negative, the slack variables can be made as large as possible and hence the objective function can be made as negative as possible. Hence, the term in the objective function corresponding to w is not relevant. Hence, any finite w will be optimal.
When C is zero, the objective function reduces to $\frac{1}{2} w^T w$. This can be minimized by setting $w = 0$. We can be sure that $w = 0$ will satisfy the constraints of the optimisation problem because we can adjust the slack variables accordingly.
When C is infinity, each α_i can take any non-negative value and hence, the optimization problem is equivalent to the Hard Margin SVM.

(c) (4 points) Construct a Kernel SVM for the XOR function with inputs x_1 and x_2 where $x_1, x_2 \in \{0, 1\}$. Visualize the decision boundary back in Euclidean space. Would it be possible to construct the function without using a kernel? If yes, how? If not, why?

Solution: The XOR function with inputs a and b can be represented as $ab' + a'b$ in boolean algebra. Taking inspiration from that, let us consider the feature mapping corresponding to the kernel as $\varphi(x) = x_1(1 - x_2) + x_2(1 - x_1)$.
Using the described feature map, points (0,0) and (1,1) will be mapped to 0 and points (0,1) and (1,0) will be mapped to 1. Hence the optimal decision boundary in the transformed space will be $x_1(1-x_2) + x_2(1-x_1) = 0.5$. This expression can be factored as $(x_1 - 0.5)(x_2 - 0.5) = 0$. This represents a pair of straight lines, i.e., $x_1 = 0.5$ and $x_2 = 0.5$. This can be visualised as a horizontal line and a vertical line which exactly separates the points in the original space.
It will not be possible to construct the function without using a Kernel because in the original space, the 4 points are not linearly separable.

(d) (3 points) [OPTIONAL BONUS] Consider an n dimensional input dataset X with m tuples. Derive an expression in terms of a Kernel to calculate the average distance of center of mass from each $\phi(x_i)$ in the feature space, where center of mass is just the average of all $\phi(x_i)$ where

$x_i \in X$.

5. (10 points) [SVM LIFE IN HIGHER DIMENSIONS] Let's learn some SVM models in this question. You are required to choose the best hyperparameters for three kernel types, and may use `sklearn.svm` for this purpose (using a part of the training data as a validation set). You will report the best regularisation parameter for each kernel type, and the overall best among these three kernel types:

   1. Linear,
   2. RBF, and
   3. Polynomial kernels.

   Note: Linear Kernel has no kernel parameter.

   Please use the template `.ipynb` file in this folder to prepare your solution. Run your SVM algorithm on the classification datasets A,B,C in this folder; report the training and test zero-one error for the different hyperparameter settings; and illustrate the learned classifier for each kernel type for datasets A,B.

   Provide your results/answers in the pdf file you upload to GradeScope, and submit your code separately in this moodle link. The code submitted should be a `rollno_name.zip` file containing two files: `rollno_name.ipynb` file (including your code as well as the exact same results/plots uploaded to Gradescope) and the associated `rollno_name.py` file.

   Please note that you are not allowed to add any import statements in the code (given template). All the required libraries have already been added to the template.

   (a) (3 points) Plot the decision boundary of SVM (3 learned kernels on 2 datasets {A,B} = 6 plots) using a 2D plot similar to Fig 4.5 in Bishop's book (i.e., add the training data points to the plots; color the positively classified area light green and negatively classified area light red, etc.).

   Solution: Please note that the below plots are of the scaled versions of the original datasets. sklearn.MinMaxScaler() was applied on the datasets so that faster training and hyperparameter tuning could be acheived.

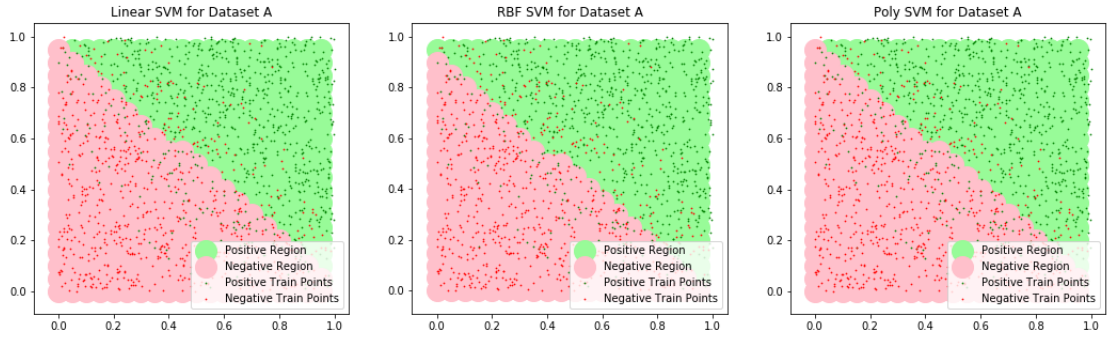

Figure 2: Plot of Dataset A

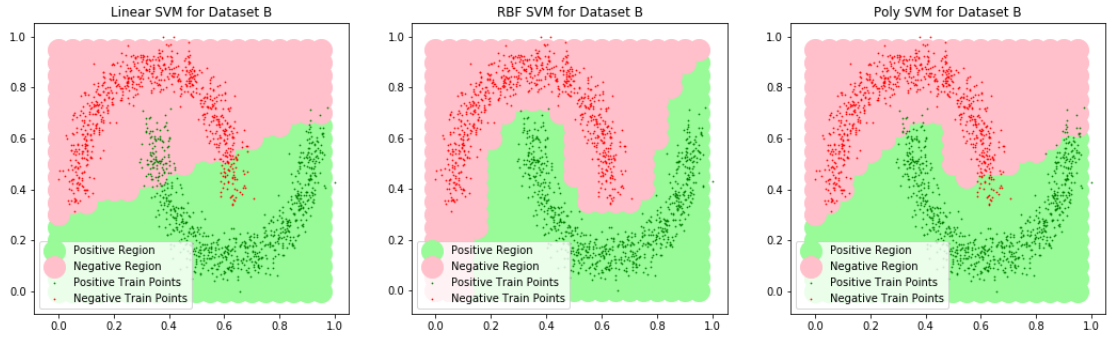

Figure 3: Plot of Dataset B

(b) (4 points) For all the three datasets {A,B,C}, provide the training and test zero-one error rates for all three kernels, along with various choices of hyperparameters (in tables with appropriately named rows and columns). Also highlight your optimal choice of hyperparameters.

Solution:

Dataset	Kernel	Optimal Hyperparameters	Train Error	Test Error
A	Linear	C = 10	0.176	0.21
A	RBF	C = 1, gamma = 0.01	0.172	0.18
A	Poly	C = 1, degree = 1, gamma = 10	0.176	0.21
B	Linear	C = 10	0.129	0.132
B	RBF	C = 10, gamma = 10	0	0.002
B	Poly	C = 10, degree = 3, gamma = 10	0.043	0.056
C	Linear	C = 100	0.073	0.103
C	RBF	C = 10, gamma = 0.1	0.124	0.157
C	Poly	C = 1, degree = 2, gamma = 10	0.047	0.085

(c) (3 points) Summarise and explain your observations based on your plots and the assumptions given in the problem.

Solution:

- Dataset A inherently has a Linear Decision Boundary and hence the classifier with linear kernel performs similarly to the other two kernels. While both RBF and Polynomial Kernels can learn non-linear decision boundaries, it is not possible to compare their performance on dataset A which has a linear decision boundary.
- By looking at the plots of dataset B, it is easy to notice that the decision boundary is non-linear in nature (two crescent moons) and the two classes are nearly seperable.
- As expected, the linear kernel performs very poorly on dataset B because it is inherently non-linear. It is interesting to note the difference in performance of RBF and Polynomial kernels. It is seen that RBF Kernel outperforms the Polynomial Kernel. This can be explained as follows. The RBF Kernel is essentially the exponential of a square. If we consider the polynomial expansion for the exponential function, there are essentially infinite number of terms which makes it infinite dimensional. Whereas, the Polynomial Kernel, for a particular degree has only a fixed number of terms, reducing its flexibility.
- For the case of dataset C, we notice that the optimal degree of the polynomial kernel is 2 and the polynomial kernel performs better than the other kernels, this indicates that the decision boundary for dataset C is quadratic in nature.