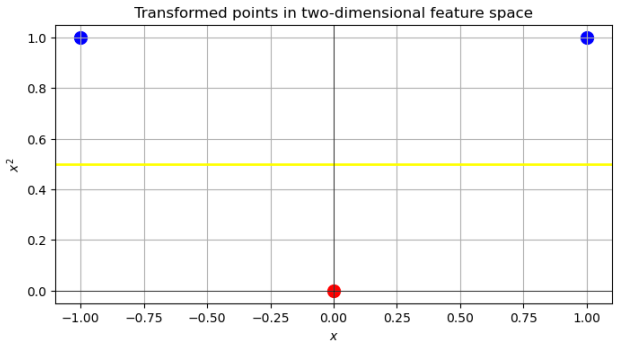
1 Support Vector Machines

1.1

No.

In one-dimensional feature space, the hyperplane of the linear classifier will be a point. x_3 is between x_1 and x_2, but x_3 has a different label than x_1 and x_2. No matter where we place a threshold, we can't separate x_3 from both x_1 and x_2.

1.2



There is a linear model that can correctly separate the three points. The yellow line is a linear classifier.

1.3

$$\begin{pmatrix} \phi(x_1)^T\phi(x_1) & \phi(x_1)^T\phi(x_2) & \phi(x_1)^T\phi(x_3) \\ \phi(x_2)^T\phi(x_1) & \phi(x_2)^T\phi(x_2) & \phi(x_2)^T\phi(x_3) \\ \phi(x_3)^T\phi(x_1) & \phi(x_3)^T\phi(x_2) & \phi(x_3)^T\phi(x_3) \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

1.4

Primal Formulations

$$\min_{\omega,b} \frac{1}{2} \| w \|_2^2$$
$$s.t.\ y_i(\omega^T\phi(x_i) + b) \geq 1, \forall i \in [1,2,3]$$

Dual Formulation

$$\max_{\alpha_i} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \phi(x_i)^T \phi(x_j)$$
$$s.t.\ \sum_i \alpha_i y_i = 0, \alpha_i \geq 0, \forall i \in [1,2,3]$$

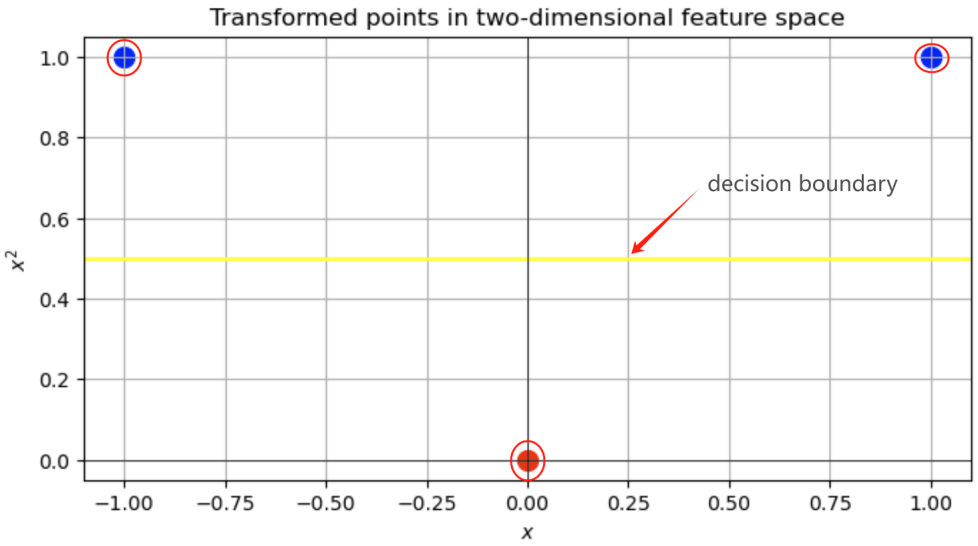
1.5

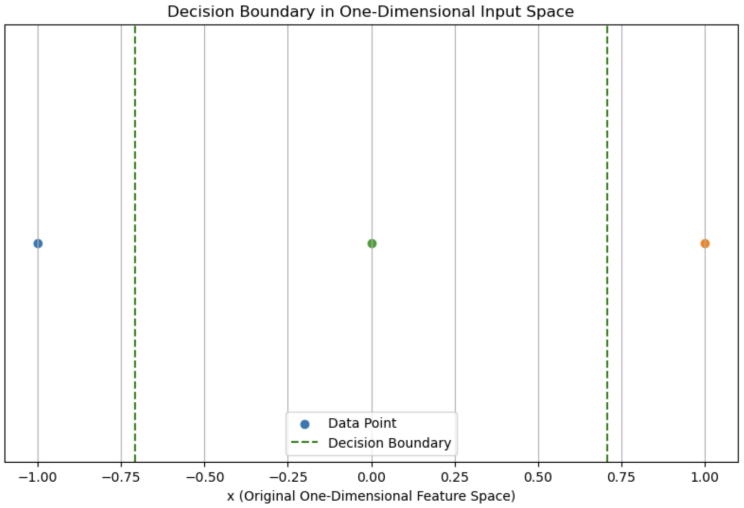
$$\begin{aligned} W(\alpha) &= \max_{\alpha_i} \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} y_i y_j \alpha_i \alpha_j \phi(x_i)^T \phi(x_j) \\ &= \alpha_1 + \alpha_2 + \alpha_3 - \frac{1}{2}(y_1^2\alpha_1^2 K_{11} + y_2^2\alpha_2^2 K_{22}) \\ &= \alpha_1 + \alpha_2 + \alpha_3 - (\alpha_1^2 + \alpha_2^2) \\ &= \alpha_1 + \alpha_2 + (\alpha_1 + \alpha_2) - (\alpha_1^2 + \alpha_2^2) \\ &= 2\alpha_1 + 2\alpha_2 - (\alpha_1^2 + \alpha_2^2) \\ &= 2\alpha_1 - \alpha_1^2 + 2\alpha_2 - \alpha_2^2 \end{aligned}$$

$W(\alpha) = 2$ when $\alpha_1 = 1, \alpha_2 = 1, \alpha_3 = 2$.

$$\omega = \sum_{i=1}^{3} \alpha_i y_i \phi(x_i) = [0, -2]^T$$
$$b = y_i - \sum_{j=1}^{n} \alpha_j^* y_j k(x_j, x_i) = 1$$
$$primal\ solution:\ f(x) = \omega^T\phi(x) + b = -2x^2 + 1$$

1.6





2 Kernel Composition

- $k_1(x, x') = \phi_1(x)^T \phi_1(x') = \sum_{i=1}^{m_1} \phi_{1(i)}(x)\phi_{1(i)}(x')$
- $k_1(x, x') = \phi_2(x)^T \phi_2(x') = \sum_{j=1}^{m_2} \phi_{2(j)}(x)\phi_{2(j)}(x')$
- denote $\phi_{1(i)}(x)$ as i th coordinate of $\phi_1(x)$
- denote $\phi_{2(j)}(x)$ as j th coordinate of $\phi_2(x)$
- $k(x, x') = \phi(x)^T \phi(x') = \sum_{j=1}^{m_2} \sum_{i=1}^{m_1} \phi_{1(i)}(x)\phi_{1(i)}(x')\phi_{2(j)}(x)\phi_{2(j)}(x')$
- $\phi(x) = [\phi_{1(1)}(x)\phi_{2(1)}(x), \phi_{1(2)}(x)\phi_{2(1)}(x), \phi_{1(3)}(x)\phi_{2(1)}(x), \ldots, \phi_{1(m1)}(x)\phi_{2(1)}(x), \phi_{1(1)}(x)\phi_{2(2)}(x), \ldots, \phi_{1(1)}(x)\phi_{2(m_2)}(x), \phi_{1(2)}(x)\phi_{2(m_2)}(x),$
- If we can construct the mapping ϕ, we can know $k(x, x')$ is a kernel

- we can also prove this in high level:
- $a_i = [\phi_{1(i)}(x_1), \phi_{1(i)}(x_2), \phi_{1(i)}(x_3), \ldots, \phi_{1(i)}(x_n)]^T$
- $b_j = [\phi_{2(j)}(x_1), \phi_{2(j)}(x_2), \phi_{2(j)}(x_3), \ldots, \phi_{2(j)}(x_n)]^T$
- $c_{ij} = a_i \circ b_j = [\phi_{1(i)}(x_1)\phi_{2(j)}(x_1), \phi_{1(i)}(x_2)\phi_{2(j)}(x_2), \phi_{1(i)}(x_3)\phi_{2(j)}(x_3), \ldots, \phi_{1(i)}(x_n)\phi_{2(j)}(x_n)]^T$
- k_1 and k_2 are kernel functions, so K_1 and K_2 are positive semidefinite. Because $k = k_1 * k_2$, we can know that $K = K_1 \circ K_2$. Let $K_1 = \sum_i a_i a_i^T$ and $K_2 = \sum_j b_j b_j^T$, following derivation can be make.

$$
\begin{aligned}
K &= K_1 \circ K_2 \\
&= \sum_i a_i a_i^T \circ \sum_j b_j b_j^T \\
&= \sum_{i,j} (a_i \circ b_j)(a_i \circ b_j)^T \\
&= \sum_{i,j} c_{ij} c_{ij}^T \ \ (c_{ij} = a_i \circ b_j) \\
&= CC^T
\end{aligned}
$$

- So K is also positive semidefinite and k is kernel function.

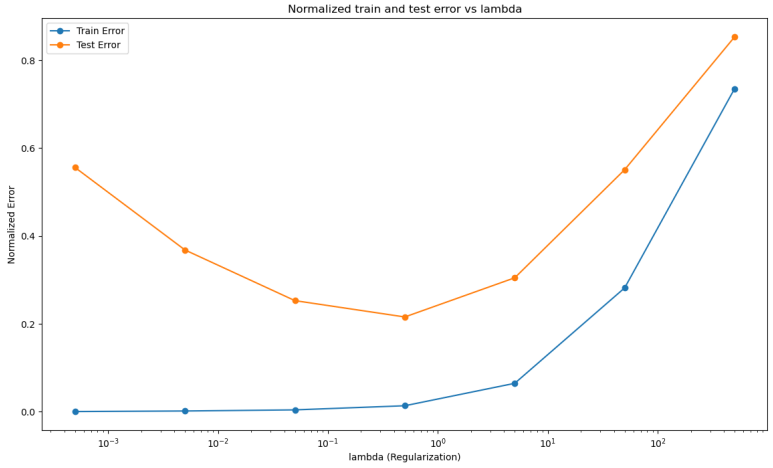
3

3.1

Normalized train error (linalg soln): 7.37206e-14

Normalized test error (linalg soln): 2.03376e+00

3.2



Normalized train error (L2 linalg soln): [3.88951187e-04 1.65428837e-03 4.18708652e-03 1.35805258e-02 6.46753694e-02 2.81939965e-01 7.35487767e-01]

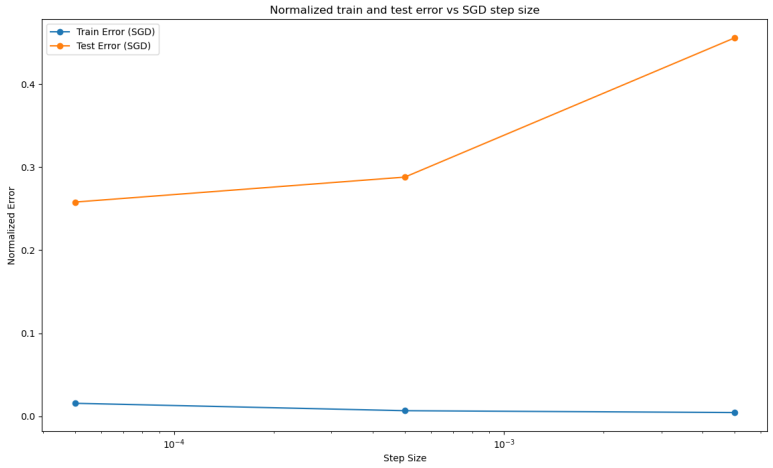
Normalized test error (L2 linalg soln): [0.55604423 0.36827449 0.25291799 0.21570124 0.30491481 0.55105611 0.85336329]

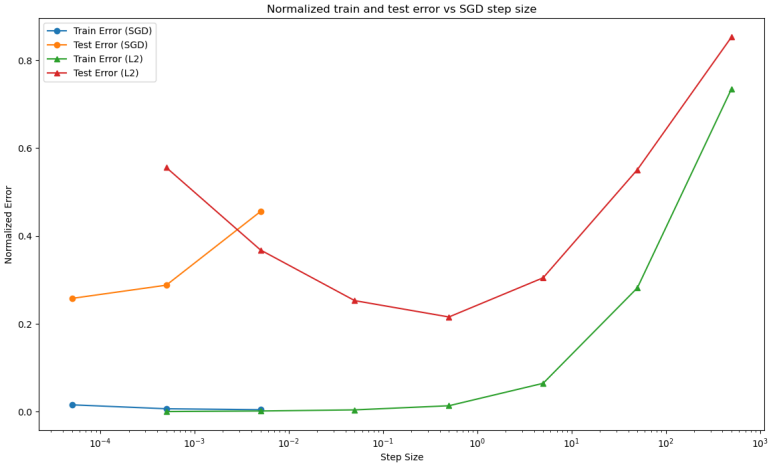
- The characteristics of the plot:
  - Normalized error of train dataset grows when λ grows.
  - Normalized error of test dataset VS λ plot has a "U" shape.
  - Normalized error of test dataset is bigger than Normalized error of train dataset for all the λ.
  - When λ = 0.5, Normalized error of test dataset becomes smallest among all λs.
  - When Normalized error of test dataset is smallest, the gap between Normalized error of train dataset and Normalized error of test dataset is relatively small compared with small λs.
  - When λ grows after 0.5, the gap between Normalized error of train dataset and Normalized error of test dataset getting smaller and smaller, but at the same time, the Normalized error of both train dataset and test data set become very big.
- Compare with result of 3.1
  - 3.1 shows when λ = 0, Normalized train error (linalg soln)= 7.37206e-14 and Normalized test error (linalg soln)= 2.03376e+00
  - After we deploy l_2 regularization, we can find, the Normalized error of train dataset become bigger compared with 3.1. But the gap between Normalized error of train dataset and Normalized error of test dataset getting smaller.
  - The test error of L2 regularization outperforms the test error without any regularization, but the train error is reversed. Without any regularization, the model is at risk of overfitting.

3.3

Normalized train error (SGD): [0.01566128 0.00673015 0.00451215]

Normalized test error (SGD): [0.25799275 0.28805797 0.45570946]

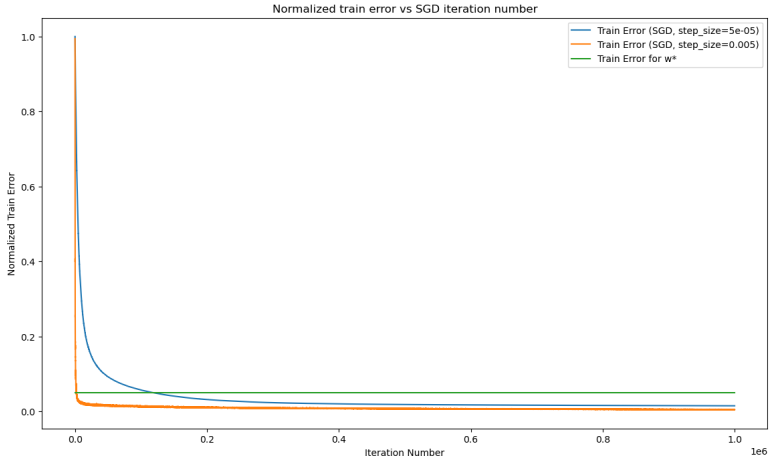




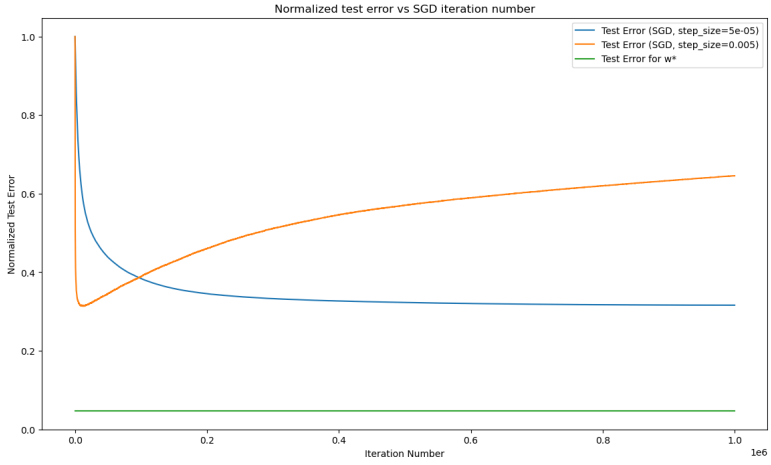
- How does SGD compare with l_2 regularization Best normalized error for SGD and l_2 regularization nearly have the same performance.
  - The performance of SGD is nearly as good as l_2 regularization when we select a good parameter lambda and step size.
  - This indicates that SGD has an implicit effect of regularization. SGD act like regularizers because of noisy update and learning rate decay.
- Training and test error of SGD compared with closed form solution in 3.1
  - From the above normalized error data, we can see that closed form solution in 3.1 and SGD both tend to kind of overfit the training data, which means both of them have a relative small train error but big test error.
  - The gap between train error and test error is bigger than the gap of w_true. Closed form solution in 3.1 should have small generalization gap.
  - However, due to the size of data size is relatively small, we can not expect small generalization gap. The performance of SGD is nearly as good as closed form solution in 3.1 when we select a good step size.

3.4

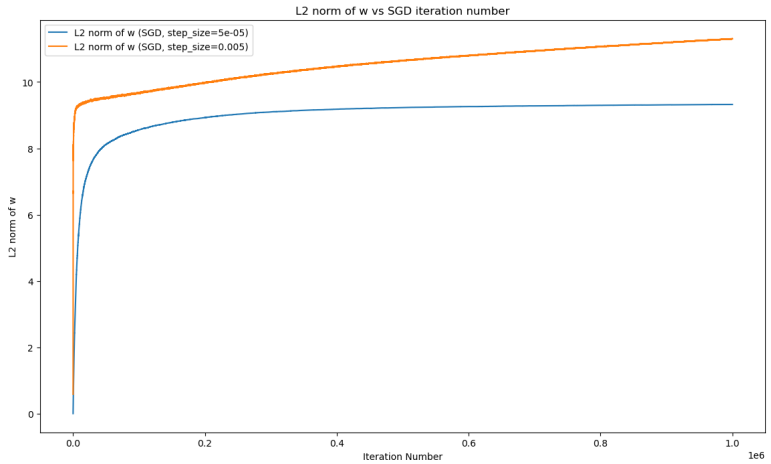
3.4.1

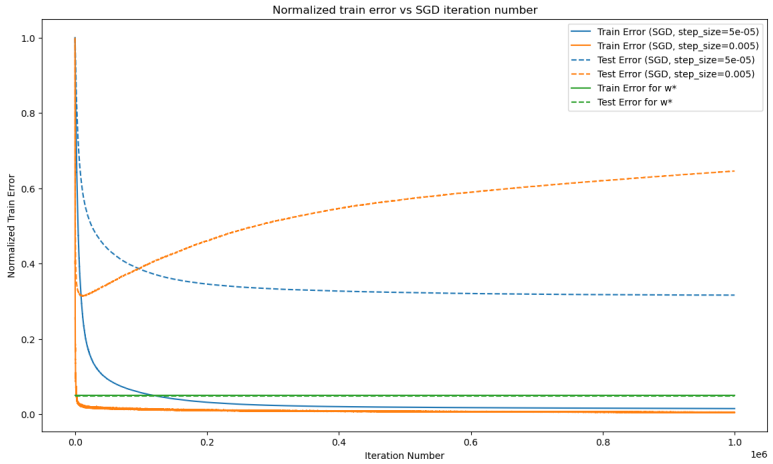


3.4.2



3.4.3



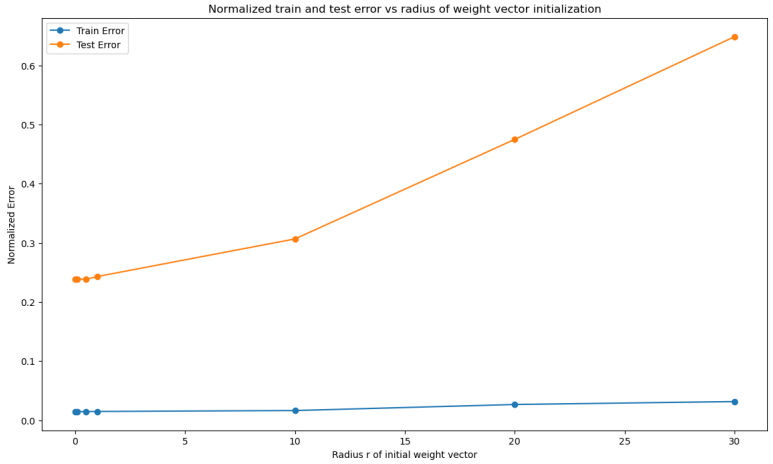


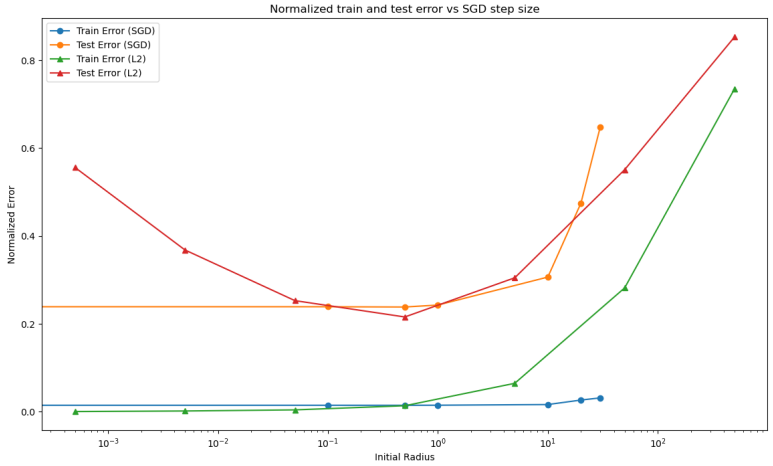
- The generalization ability of SGD with different step size:
  - SGD with right step size will converge for both training error and test error. It means that the generalization gap will become small for this model with right step size.
  - However, SGD with improper step size will converge for training error and diverge for test error. It means that the generalization gap will become big when we do more iteration. The generalization ability of SGD with improper step size is weak.
- Does the plot correspond to the intuition that a learning algorithm starts to overfit when the training error becomes too small
  - Yes
  - The training error of $\lambda = 0.005$ is smaller than that of $\lambda = 0.00005$. And the test error of $\lambda = 0.005$ is bigger than that of $\lambda = 0.00005$. Therefore, the generalization gap of model with small training error is bigger than model with big training error, which indicates the model with too small training error is overfitting.
- How does the generalization ability of the final solution depend on the l2 norm of the final solution?
  - The generalization ability of the final solution depends on the l2 norm of the final solution. The SGD with good generalization ability has a smaller l2 norm of w.

3.5

Normalized train error (SGD): [0.01464884 0.01465723 0.0145712 0.01478888 0.01632941 0.0265503 0.03151314]

Normalized test error (SGD): [0.23883863 0.23896153 0.23836105 0.24288567 0.30653049 0.47488753 0.64864869]

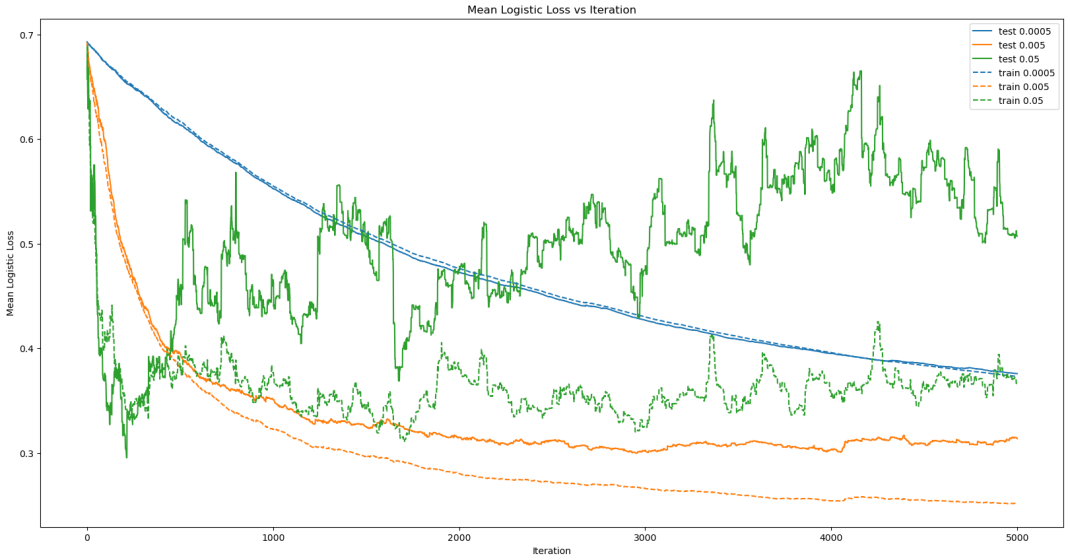




- comments on the average normalized training/test error vs r
  - the average normalized test error will grow very fast when r becomes bigger than 10.
  - However, if r is smaller than 10, the average normalized training/test is kind of insensitive to the value of r.
  - Conclusion: the initial norm of w should be smaller than or be close to the norm of optimal w. If the norm of initial w is too bigger than the normal of optimal w, both of final training and test error will become bad.
- Explanation for the plot behavior:
  - The performance of l_2 regularization solution with too big regularization coefficient is similar with that of SGD with too big norm of initial w. This behavior may indicate that the norm of initial w will result in big norm of final w in SGD. Therefore, we should pay attention to select a proper initial norm of initial w.

4

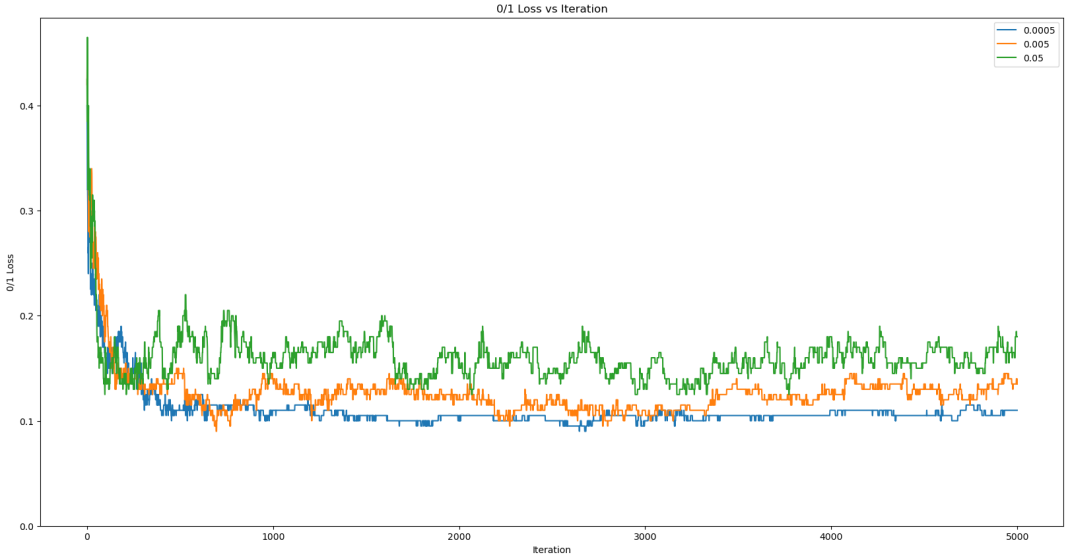
4.1



- For 0.005 and 0.05 step sizes, test errors are all higher than training error.
- we can see that the gap between the objective function values on the train and test data of the best step sizes, which is 0.0005 is the smallest among all three step size

- For step size 0.005 and 0.05, the final mean logistic loss can not converge. The gap bewteen train and test loss will become larger and larger when we do more iterations
- A proper increase in the step size will speed up convergence thus obtaining a lower error rate, but when the step size is too large, the error rate rises instead.

4.2



- step size is 0.0005 and the corresponding value is 0.11

4.3

- (0.11, 0.135, 0.18)
- Logistic loss can act well as a surrogate for the 0-1 loss
- Good results, but accuracy still needs to be improved
- Even if the final mean logistic loss has trend of divergence, the 0-1 loss for all the three SGD logistic regression of different step size is very similar. It indicates all of them do well on binary classification

5

5.1

- Linear classifiers do not perform well on the Moon and Circle dataset, but RBF SVM does a good job.
- The decision boundaries of linear classifiers are straight lines, which can't capture the complexity of the data's distribution. The use of RBF kernel allows the SVM to find a nonlinear decision boundary that better captures the distribution of the data. SVM with RBF can let data points become separable by mapping them into high-dimensional feature space, thus do well on nonlinear classification.

5.2

C	**0.00025**	**0.0025**	**0.025**	**0.25**	**2.5**	**25**	**250**	**2500**
Train accuracy(Linearly)	0.53	0.53	0.80	0.93	0.93	0.93	0.93	0.93
Test accuracy(Linearly)	0.45	0.45	0.85	0.90	0.80	0.85	0.85	0.85

- As C increases, training accuracy increases and then remain constant, test accuracy increases and then fall back slightly. SVM transitions from underfitting to a good fit and potentially to overfitting.
- As C increases, decision boundary changes from slanting down to slanting up.
- We also notice the distance between adjacent contour lines becomes widest when C = 0.25, which has the best train accuracy and test accuracy. It means the best C = 0.25 has the largest margin width among all Cs.

5.3

SVM with RBF kernel For MOON dataset

C	**0.025**	**0.1**	**0.25**	**0.1**	**2.5**	**10**	**50**	**100**
Train accuracy(MOON)	0.52	0.75	0.93	0.93	0.97	0.98	0.98	0.98

C	0.025	0.1	0.25	0.1	2.5	10	50	100
Test accuracy(MOON)	0.47	0.60	0.95	0.97	0.97	0.97	0.95	0.93
Train accuracy(Circle)	0.57	0.57	0.87	0.97	0.97	0.98	1.0	1.0
Test accuracy(Circle)	0.40	0.40	0.68	0.88	0.90	0.88	0.85	0.85
Train accuracy(Linearly)	0.53	0.53	0.97	0.93	0.97	1.0	1.0	1.0
Test accuracy(Linearly)	0.45	0.45	0.90	0.85	0.85	0.85	0.85	0.85

- As C increases, training accuracy increases and even up to 1, test accuracy increases and then fall back slightly.
- As C increases, the decision boundary becomes more flexible, fitting closer to the training data, which usually increases both training and test accuracy. However, there is a risk of overfitting.

5.4

Reg_lambda	0	0.5	1	1.5	2	2.5	5	10
Train accuracy(linearly)	0.93	0.93	0.90	0.87	0.87	0.87	0.87	0.83
Test accuracy(Linearly)	0.90	0.95	0.90	0.90	0.90	0.90	0.90	0.90

Yes

- For Linear separable dataset, as reg-lambda increases, the train accuracy become a little lower.
- Although applying a little regularization (reg-lambda = 0.5) on logistic regression didn’t improve train accuracy, it can improve test accuracy, indicating good generalization.
- If there is a lot of outliers, we can expect that regularization will have some effect to control model being overfitting.
- For other two dataset, weak regularization still can't capture the non-linear boundaries, so accuracy doesn't change when reg_lambda changes.