ESE 546, FALL 2023

HOMEWORK 1

KEQI WU [KEQIWU@SEAS.UPENN.EDU],
COLLABORATORS: NONE

Solution 1 (Time spent: 5 hour)**.** Your solution goes here.

(a)

To minimize the violation of the original constraints, we can choose the objective function as:

$$\frac{1}{2}||\theta||^2 + C\sum_{i=1}^{n}\xi_i$$

subject to constraint:

$$y_i(\theta^T x_i + \theta_0) \geq 1 - \xi_i$$

(b)

Support samples are those datasets lying within the margin of the decision hyperplane. They are highly influential and play an important role of defining the boundary and the margin

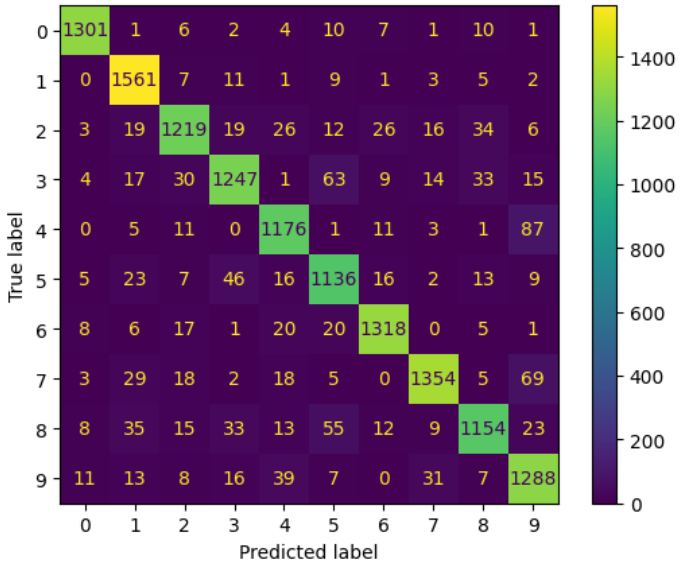
(c)

check code

(d)

check code
C is a regularization parameter (a penalty parameter) that controls the tolerance of the classification error. It is a trade-off between margin width and classification error. Smaller C gives a larger margin, and larger C gives a smaller margin. The default value is 1.0.
γ is a parameter that controls the curvature of the decision boundary and flexibility of the model. It is effectively by mainly using RBF kernel. A higher γ gives more curvature to the boundary. A lower γ gives less curvature to the boundary. The default value is 'scale', which is basically $1/(n_{features} * x_{train}.var())$, for question 1 data after min-max normalization, we have, 0.01334.

The validation error for y_{val} is: 9.6500%
The support sample ratio is: 57.7625%
The classification error for y_{test} is: 8.9000%
Confusion matrix:



The confusion shows that most of data are correctly classified, but still a not too small number of data are misclassified. We also saw that test error is less than the validation error, so this might indicate that the model experiences a little underfitting issue. The model is too simple that can not best identify the underlying pattern of the data. In this question, it probably results from the train-test data points imbalance (the training sample is too small), and poor choices of model parameters.

(e)

Options that I have never seen: coef0, shrinking, probability, cache_size, decision_function_shape, random_state.

Shrinking is a Boolean parameter that controls whether the shrinking heuristic is used or not. Calculating distances and kernel matrices, storing support samples and solving optimization problem require a decent amount of RAM. Shrinking reduces number of kernel values that used to update the decision boundary. It can increase the training speed of the model, but may have some influence on the model accuracy.

SVM in scikit-learn uses SMO-type algorithm (Sequential minimal optimization) to fit the data. SMO breaks a large problem into small pieces. It assigns a Lagrange multiplier to each data point. It iteratively select a pair of Lagrange multipliers and solve the optimization problem, then update their values that satisfy the constraints of the problem. It will stop when the multipliers become sufficiently small or the model reaches the maximum iteration.

(f)

There are two methods that are used to handle the multiclass classification problem:
1. One to One Approach: the model breaks the multi-class classification problem into many binary classification problems. Every pair of classes produces a binary classifier. When we do the multi-class classification, we collect all the results from binary classifiers and determine the right one.
2. One to Rest Approach: We train a binary classifier for each class versus the rest of classes. The binary classifier can identify the differences between that class and the rest of classes. The classifier with the best result is the final prediction.

An alternative way is to apply hierarchical classification. We make decisions in a tree structure (like decision tree). Binary classifiers can trained between children, and each time we can discard some of the classes. We can find the final results at the leaves of the tree.

(g)

check code

Hyper Parameters: 0.001, 0.01, 0.1, 1, 10, 100, 1000
Mean Accuracy: 10.2%, 10.2%, 82.9375%, 90.7125%, 93.0625%, 93.425%, 93.1625%

Because GridSearchCV uses cross validation, we can pick the hyper parameter with the best mean accuracy score among those candidates. In this case, hyper parameter 100 has the highest mean score 93.425%. Therefore, we should pick 100 as the best C for model training.

(h)

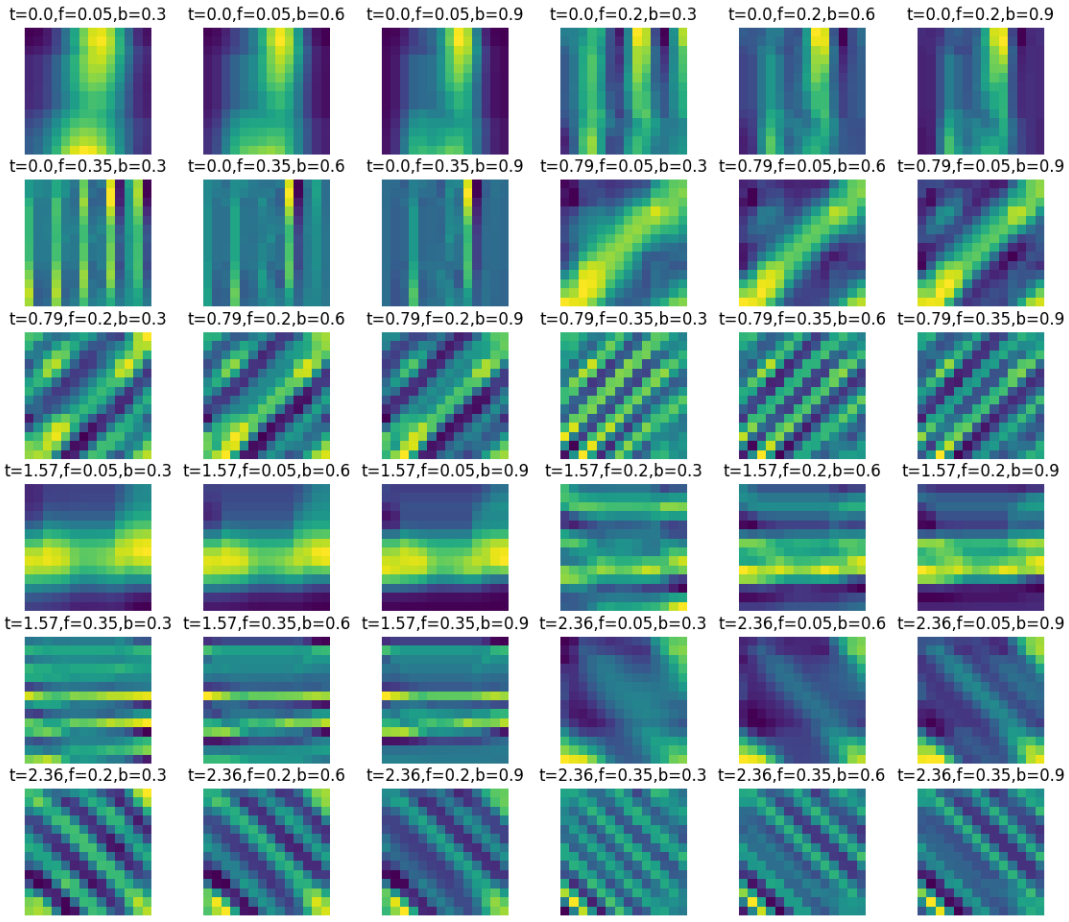
check code

(j)

check code

Frequency F changes the thickness of the strip. With larger F, the smaller of the wavelength, the thickness of the strip decreases.Theta θ changes the orientation of the function. With $theta = 0$, strips are oriented vertically. With $theta = \pi/4$, strips are oriented 45 degrees clockwise. With $theta = \pi/2$, strips are oriented horizontally. Bandwidth controls the overall number of strips. With bandwidth increases, sigma decreases, the number of strips decreases.

Plot of 36 filters:



For 36 filters:
Training score: 96.2000%
Validation score: 89.4000%
For 168 filters:
Training score is: 97.6000%

Validation score is: 88.1000%

From two results, we suspect that the model experiences overfitting as training score gets higher and validation score gets lower. The feature size is significant larger than the training data size. it increases the complexity of the model and bring the curse of dimensionality.

Solution 2 (Time spent: 10 mins)**.** Your solution goes here.

Proof by induction: Let $\mu = P(x_1)x_1 + P(x_2)x_2 + ...P(x_n)x_n$, then:

Base case: $n = 2$,

$\varphi(P(x_1)x_1 + P(x_2)x_2)) \leq P(x_1)\varphi(x_1) + P(x_2)\varphi(x_2)$ by the property of a convex function

Inductive hypothesis:
Assume that

$$\varphi(\sum_{i=1}^{k} P(x_i)x_i) \leq \sum_{i=1}^{k} P(x_i)\varphi(x_i)$$

holds true for all $k = 2, ..., n$

Induction:
When $k = n + 1$,

$$\begin{aligned}
&\varphi(\sum_{i=1}^{n+1} P(x_i)x_i) \\
&= \varphi(P(x_{n+1})x_{n+1} + \sum_{i=1}^{n} P(x_i)x_i) \\
&= \varphi(P(x_{n+1})x_{n+1} + (1 - P(x_{n+1}))\sum_{i=1}^{n} \frac{P(x_i)}{1 - P(x_{n+1})}x_i) \\
&\leq P(x_{n+1})x_{n+1} + (1 - P(x_{n+1}))\varphi(\sum_{i=1}^{n} \frac{P(x_i)}{1 - P(x_{n+1})}x_i) \\
&= P(x_{n+1})x_{n+1} + \varphi(\sum_{i=1}^{n} \frac{P(x_i)}{1 - P(x_{n+1})}x_i) \\
&\leq P(x_{n+1})x_{n+1} + \sum_{i=1}^{n} P(x_i)\varphi(x_i) \text{ by induction hypothesis} \\
&= \sum_{i=1}^{n+1} P(x_i)\varphi(x_i)
\end{aligned}$$

Conclusion:
Since the case works for $k = n+1$, by induction, we can conclude that $E_x[\varphi(X)] = \sum_{i=1}^{n} P(x_i)\varphi(x_i) \geq \varphi(\sum_{i=1}^{n} P(x_i)x_i) = \varphi(\mu)$ as required

Solution 3 (Time spent: 5 hour)**.** Your solution goes here.

(a)

check code

(b)

check code

(c)

check code

(d)

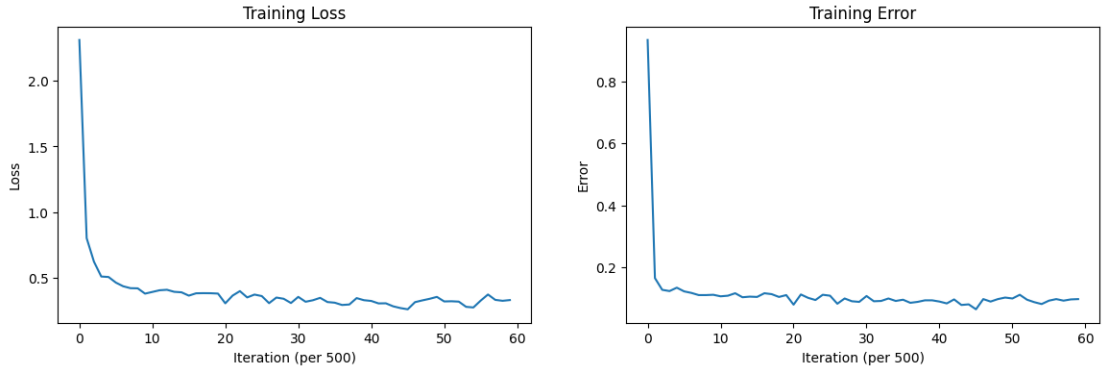
check code

(e)

check code

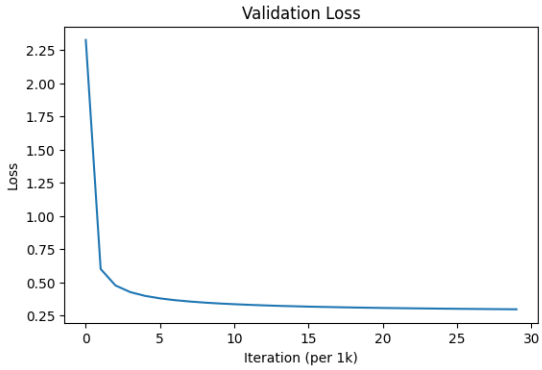
(f)

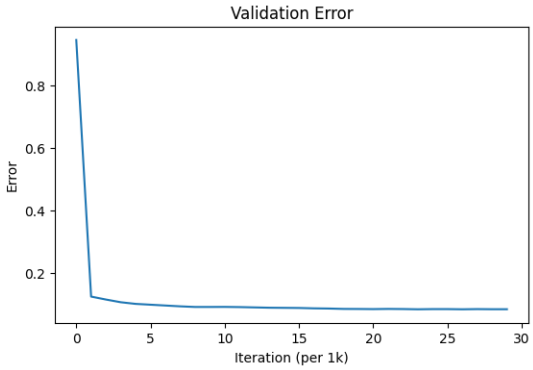
check code


Training Loss
Loss
Iteration (per 500)
Training Error
Error
Iteration (per 500)

(g)

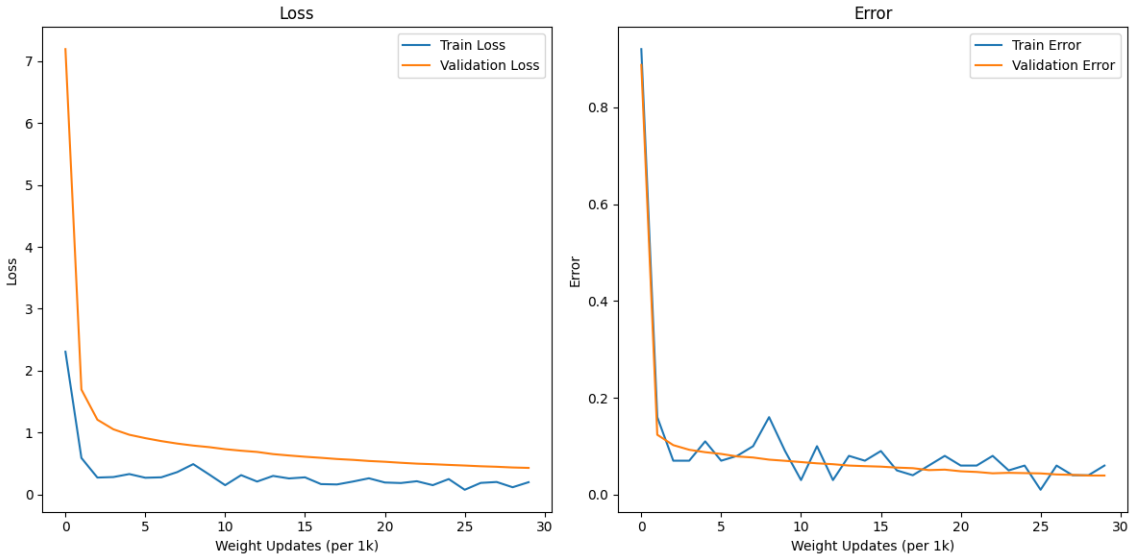
check code


Validation Loss
Loss
Iteration (per 1k)
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0
5
10
15
20
25
30


Validation Error
Error
Iteration (per 1k)
0.8
0.6
0.4
0.2
0
5
10
15
20
25
30

(h)

check code


Loss
Train Loss
Validation Loss
Loss
Weight Updates (per 1k)
Error
Train Error
Validation Error
Error
Weight Updates (per 1k)