HW 6

黃熙漢

5.

Augmented Primal problem :

$$\min_{\xi, \tilde{w}} \frac{1}{2} \tilde{w}^T \tilde{w} + C \sum_{n=1}^{N} \xi_n$$

s.t $y_n z_n^T \tilde{w} \geq 1 - \xi_n, \ \forall n$

$\xi_n \geq 0, \ \forall n$

where $z_n = [1, x_n^T]^T$

(b^*, w^*, α^*) is an optimal solution to the original problem P_1 only if $b^* = 0$ is indeed the optimal bias for (P_1).

counter example :

feature space $(d = 1)$

$x_1 = 1, \ y_1 = +1$

$x_2 = -1, \ y_2 = -1$

Primal Prob :

$$\min_{b, \xi, w} \frac{1}{2} w^2 + C(\xi_1 + \xi_2)$$

s.t $y_1(wx_1 + b) \geq 1 - \xi_1 \quad , \xi_1 \geq 0$

$y_2(wx_2 + b) \geq 1 - \xi_2 \quad , \xi_2 \geq 0$

to satisfy $y_1(wx_1 + b) \geq 1 \quad y_2(wx_2 + b) \geq 1$

$w + b \geq 1 \quad , \quad -w + b \geq 1$

$2b \geq 2 \Rightarrow b \geq 1$

$2w \geq 0 \Rightarrow w \geq 0$

minimiz $\frac{1}{2} w^2$, let $w = 0, \ b = 1$

opt sol : $w^* = 0, \ b^* = 1 \ \xi_1 = 0 \ \xi_2 = 0$

Augmented Primal Prob

$z_n = [1, x_n]^T$, $\tilde{w} = [b, w]^T$

Dr. Threshold Sol : $\tilde{w}_0^* = 0$

Correspond w^*; $w^* = [0]^T$

Original Opt Primal sol : $b^* = 1$, $w^* = 0$

Dr. Threshold's solution: $b^T = 0$, $w^T = 0$

$y_1(0 \cdot 1 + 0) = 0 \geq 1 - \xi_1 \Rightarrow \xi_1 \geq 1$

$y_2(0 \cdot (-1) + 0) = 0 \geq 1 - \xi_2 \Rightarrow \xi_2 \geq 1$

objective $(P_1) = 2C$

objective (Dr. Threshold) $= 2C$

- same but constraints are violated for Dr Threshold Solution

infeasble to Primal Problem

∴ Dr. Threshold solution ($b^* = 0$, $w^* = 0$) does not satisfy the original primal problem's constraints, thereby not being optimal for Primal Problem.

6. $y_0(w^T \Phi(x_0) - b) \geq 1$, $y_0 = -1$, $x_0 = 0$, $b \geq 1$

Langrange

$L(w, \xi_n, b, \alpha, \beta, \gamma)$

$$= \frac{1}{2} w^T w + C \sum_{n=1}^{N} \xi_n + \alpha \sum_{n=1}^{N} (1 - \xi_n - y_n(w^T \Phi(x_n) + b)) - \beta \sum_{n=1}^{N} (-\xi_n) - \gamma(y_0(w^T \Phi(x_0) + b - 1]$$

$$\max_{\beta \geq 0, \alpha \geq 0, \gamma \geq 0} (\min_{w, \xi_n, b} (L(w, \xi_n, b, \alpha, \beta, \gamma))))$$

$\frac{\partial L}{\partial b} = 0$, $-\sum_{n=1}^{N} \alpha_n y_n \gamma y_0 n = 0$, $\gamma_n = \sum_{n=1}^{N} \alpha_n y_n$, $y_0 = -1$

$\frac{\partial L}{\partial w} = 0$, $w - \alpha \sum_{n=1}^{N} y_n \Phi(x_n) + \beta \Phi(0)$, $w = \sum_{n=1}^{N} \alpha_n y_n \Phi(x_n) - \beta \Phi(0)$

$\frac{\partial L}{\partial \xi} = 0$ $C \sum_{n=1}^{N} 1 - \alpha \sum_{n=1}^{N} 1 = \sum_{n=1}^{N} \beta$, $C - \alpha_n = \beta_n$

$\therefore M_n \geq 0$

$0 \leq \alpha_n \leq C$

$$L_D(\alpha) = \frac{1}{2} w^T w + C \sum_{n=1}^{N} \xi_n - \sum_{n=1}^{N} \alpha_n [y_n(w^T \Phi(x_n) + b) - 1 + \xi_n] - \gamma [y_0(w^T \Phi(x_0) + b) - 1]$$

$$L_D(\alpha) = -\frac{1}{2} \alpha^T M \alpha + 2 \sum_{n=1}^{N} \alpha_n$$

$$M_{ij} = K(x_i, x_j) - K(x_i, 0) - K(x_j, 0) + K(0, 0)$$

$\therefore$ Dual prob

$$\max -\frac{1}{2} \alpha^T N \alpha + 2 \sum_{n=1}^{N} \alpha_n$$

s.t. $0 \le \alpha_n \le C, \; n = 1, 2 \dots N$

7.

$b=0, \ \alpha=1$

$\hat{h}(x) = \text{sign}(\sum_{n=1}^{N} y_n \cdot \exp(-\gamma ||x-x'||^2))$

consider two condition

$n=m$ $\quad y_n \cdot \exp(-\gamma || x_n - x_n ||^2)) \Rightarrow y_n$

$n \neq m \sum_{m\neq n} y_n \cdot k(x, x')$

So

$\hat{h}(x) = \text{sign}(y_n + \sum_{m\neq n} y_m k(x_m, x_n))$

for $m \neq n$ $\quad ||x_n - x_m|| \geq \varepsilon$

$k(x_m,x_n) = \exp(-\gamma ||x_m - x_n||^2) \leq \exp(-\gamma\varepsilon^2)$

$|y_m k(x_m,x_n)| \leq \exp(-\gamma\varepsilon^2)$

$|\sum_{m\neq n} y_m k(x_m,x_n)| \leq \exp(-\gamma\varepsilon^2)(N-1))$

$\hat{h}(x_n) = y_n$:

$y_n = \text{sign}(y_n + \sum_{m\neq n} y_m k(x_m,x_n))$, valid if $|\sum_{m\neq n} y_m k(x_m,x_n)| < 1$

$\exp(-\gamma\varepsilon^2) < \frac{1}{N-1}$

$-\gamma\varepsilon^2 < \ln(\frac{1}{N-1}) = -\ln(N-1) \Rightarrow \gamma > \ln\frac{(N-1)}{\varepsilon^2}$ (proved.)

∴ When γ satisfies above inequality, $\hat{h}$ correctly classifies all training data, means larger γ kernel can seperate give data set perfectly

8. $k(x,x') = \exp(2\cos(x-x')-2)$

Trigonometric:

$$k(x,x') = \exp(2(\cos x \cos x' + \sin x \sin x') - 2)$$

feature map

$$\phi = \begin{bmatrix} \sqrt{2}\cos x \\ \sqrt{2}\sin x \end{bmatrix}$$

$$\phi(x)^T \phi(x') = 2\cos x \cos x' + 2\sin x \sin x'$$

Replace

$$k(x,x') = \exp(\phi(x)^T \phi(x') - 2)$$

$$= \exp(-2)\exp(\phi(x)^T \phi(x'))$$

Proof its semi-positive definite

$$k'(x,x) = \exp(\phi(x)^T(x'))$$

Taylor expand

$$k'(x,x') = \sum_{n=1}^{\infty} \frac{(\phi(x)^T \phi(x'))^k}{k!}$$

, valid! inner product with power and positive const scaled

∴ $k(x,x') = e^{-2} \cdot \exp(\phi(x)^T \phi(x'))$ scale by $e^{-2} > 0$ (positive scalar

∴ $k(x,x')$ is a valid kernel for all $x, x' \in R$

9. $k_{ds}(x, x') = (\phi_{ds}(x))^T \Phi_{ds}(x') = \sum_{i=1}^{d} \sum_{\theta} g_{i,\theta}(x) \cdot g_{i,\theta}(x')$

$g_{i,\theta}(x) = \mathbb{I}(x_i > \theta)$

feature map : $\Phi_{ds}(x) = [g_{1,\theta_1}(x), g_{1,\theta_2}(x), \ldots, g_{d,\theta_n}(x)]^T$

each feature has differenc threshold, exp: feature 1 $[\mathbb{I}(150 > 140.5), \mathbb{I}(150 > 141 \ldots]$

so kernel reflect how many times both points x, x' agree on being above thresholds across all feature. more agree, higher similar

we need to find all $g_{i,\theta}(x) = 1$ and $g_{i,\theta}(x') = 1$, occur it both x_i, x'_i greater θ

so, $\theta < \min(x_i, x'_i)$

$n_i = \min(x_i, x'_i) - L$, stare with $L + 0.5$

$k_{ds}(x, x') = \sum_{i=1}^{d} n_i = \sum^{d} [\min(x_i, x'_i) - L]$ #

13.

Primal

minimize over $w \in R^d$, $b \in R$, $\xi \in R^n$

$$\min_{w,b,\xi} \frac{1}{2}|w|^2 + C\sum_{i=1}^{n} \xi_i$$

$$y_i(w^T x_i + b) \quad 1-\xi_i \quad , \xi_i \geq 0, \forall i$$

Dual Problem

$\alpha \in R^n$

$$\max_{\alpha} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

$$0 \leq \alpha_i \leq C, \sum_{i=1}^{n} \alpha_i y_i = 0 \quad , \forall i$$

Solve Dual (Dual Problem)

$$L(\alpha, w, b, \xi) = \frac{1}{2}\alpha^T Q \alpha - \sum_{i=1}^{n} \alpha_i + \sum_{i=1}^{n} \lambda(\alpha_i - C) - \sum_{i=1}^{n} \mu_i \alpha_i$$

$$+ v\left(\sum_{i=1}^{n} \alpha_i y_i\right) \quad , \text{with } Q_{ji} = y_i y_j x_i^T x_j$$

$$\frac{\partial L}{\partial \alpha_i} = [Q\alpha]_i - 1 + \lambda_i - \mu_i + v y_i = 0$$

i. $\lambda_i, \mu_i \geq 0$

ii. $\mu_i \alpha_i = 0$

iii $\lambda_i(\alpha_i - C) = 0$

$\alpha = Q^{-1}(1 - \lambda + \mu - \nu y))$

Derive to Primal Variable

if $0 < \alpha_i < C$, $\mu_i = 0$, $\lambda_i = 0$

$\alpha_i = 0$, $\mu_i \geq 0$, $\lambda_i = 0$

$\alpha_i = C$, $\lambda_i \geq 0$, $\mu_i = 0$

∴ Yes, when derive simplified lagrange dual problem of the soft margin SVM dual, we essentially retrieve the original soft margin SVM primal problem. This happens because, under strong duality condition which hold for convex optimize problem

They are similar in mathematic, just approached from the diff perspective, one from primal weight and bias, the other from dual variables, Lagrange multiplier associated with the constraints, so we can just using dual safely.

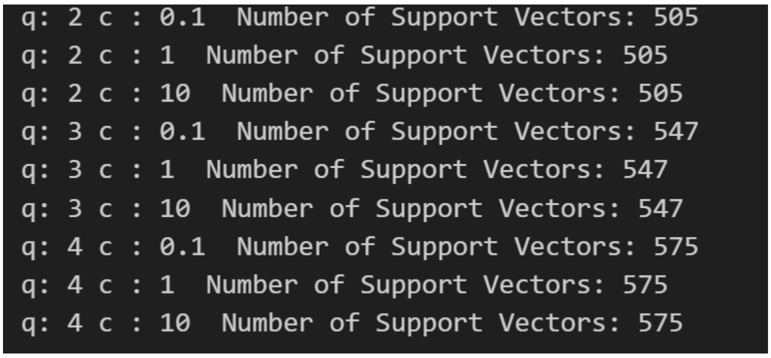
10.

Phenomenon Descripe:

Based on the results, increasing the degree of the polynomial term leads to a higher number of support vectors, indicating that the polynomial term has a greater impact on the model's performance than the regularization parameter

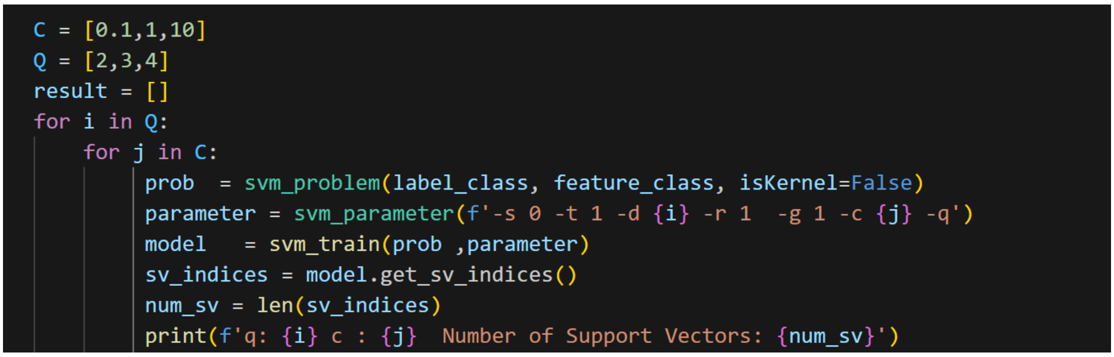
C. Specifically, as the polynomial degree increases, the model becomes more complex, allowing it to capture intricate patterns within the data. This increased complexity results in the formation of more support vectors, which are critical in defining the decision boundary of the Support Vector Machine (SVM).

Graph:



```
q: 2 c : 0.1  Number of Support Vectors: 505
q: 2 c : 1  Number of Support Vectors: 505
q: 2 c : 10  Number of Support Vectors: 505
q: 3 c : 0.1  Number of Support Vectors: 547
q: 3 c : 1  Number of Support Vectors: 547
q: 3 c : 10  Number of Support Vectors: 547
q: 4 c : 0.1  Number of Support Vectors: 575
q: 4 c : 1  Number of Support Vectors: 575
q: 4 c : 10  Number of Support Vectors: 575
```

Code:

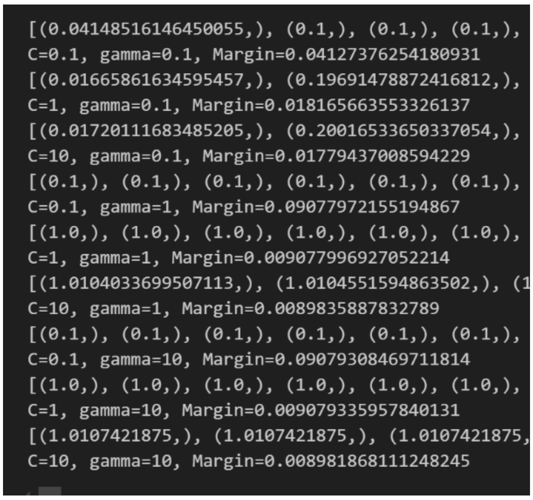


```python
C = [0.1,1,10]
Q = [2,3,4]
result = []
for i in Q:
    for j in C:
        prob  = svm_problem(label_class, feature_class, isKernel=False)
        parameter = svm_parameter(f'-s 0 -t 1 -d {i} -r 1  -g 1 -c {j} -q')
        model    = svm_train(prob ,parameter)
        sv_indices = model.get_sv_indices()
        num_sv = len(sv_indices)
        print(f'q: {i} c : {j}  Number of Support Vectors: {num_sv}')
```

11.

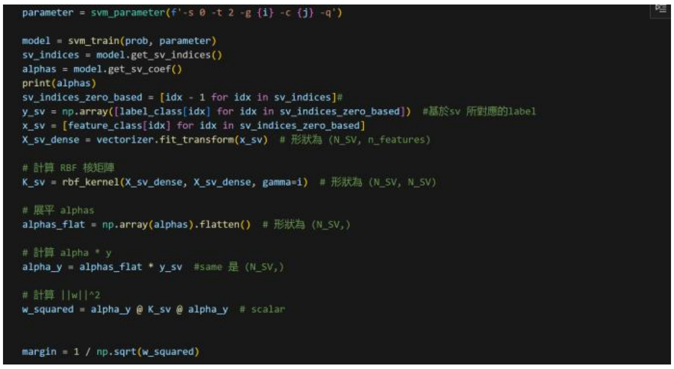
Phenomenon Description:Based on the experimental results, when both the hyperparameters γ and C are set to values smaller than 1, the SVM model achieves a larger margin. This behavior aligns with the properties of Gaussian distributions, where a higher γ corresponds to a smaller standard deviation. Consequently, a higher γ results in a more localized influence of each support vector, leading to a more complex decision boundary with a smaller margin. Conversely, lower γ values spread the influence of support vectors over a larger area, contributing to a smoother decision boundary with a larger margin. Additionally, the regularization parameter C acts as an upper bound for the Lagrange multipliers (α), controlling the trade-off between maximizing the margin and minimizing classification errors. A smaller C allows for a larger margin by permitting some misclassifications, thereby enhancing the model's generalization capability. In contrast, a higher C enforces stricter classification rules, resulting in a narrower margin as the model strives to minimize training errors.

Graph:



```
[(0.04148516146450055,), (0.1,), (0.1,), (0.1,),
C=0.1, gamma=0.1, Margin=0.04127376254180931
[(0.01665861634595457,), (0.19691478872416812,),
C=1, gamma=0.1, Margin=0.018165663553326137
[(0.01720111683485205,), (0.20016533650337054,),
C=10, gamma=0.1, Margin=0.01779437008594229
[(0.1,), (0.1,), (0.1,), (0.1,), (0.1,), (0.1,),
C=0.1, gamma=1, Margin=0.09077972155194867
[(1.0,), (1.0,), (1.0,), (1.0,), (1.0,), (1.0,),
C=1, gamma=1, Margin=0.009077996927052214
[(1.0104033699507113,), (1.0104551594863502,), (1
C=10, gamma=1, Margin=0.0089835887832789
[(0.1,), (0.1,), (0.1,), (0.1,), (0.1,), (0.1,),
C=0.1, gamma=10, Margin=0.09079308469711814
[(1.0,), (1.0,), (1.0,), (1.0,), (1.0,), (1.0,),
C=1, gamma=10, Margin=0.009079335957840131
[(1.0107421875,), (1.0107421875,), (1.0107421875,
C=10, gamma=10, Margin=0.008981868111248245
```

Code:

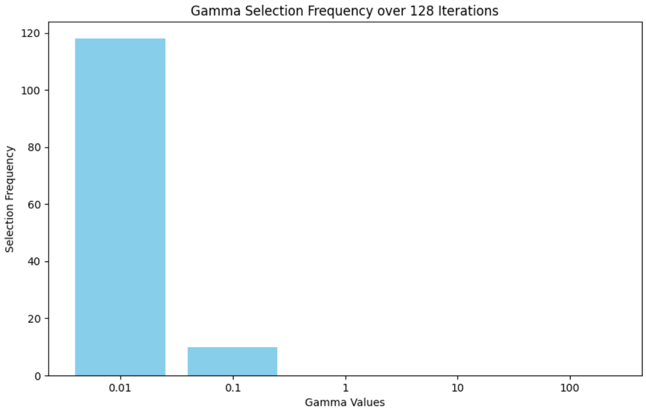


```
parameter = svm_parameter(f'-s 0 -t 2 -g {i} -c {j} -q')

model = svm_train(prob, parameter)
sv_indices = model.get_sv_indices()
alphas = model.get_sv_coef()
print(alphas)
sv_indices_zero_based = [idx - 1 for idx in sv_indices]#
y_sv = np.array([label_class[idx] for idx in sv_indices_zero_based])  #基於sv 所對應的label
x_sv = [feature_class[idx] for idx in sv_indices_zero_based]
X_sv_dense = vectorizer.fit_transform(x_sv)  # 形狀為 (N_SV, n_features)

# 計算 RBF 核矩陣
K_sv = rbf_kernel(X_sv_dense, X_sv_dense, gamma=i)  # 形狀為 (N_SV, N_SV)

# 展平 alphas
alphas_flat = np.array(alphas).flatten()  # 形狀為 (N_SV,)

# 計算 alpha * y
alpha_y = alphas_flat * y_sv  #same 是 (N_SV,)

# 計算 ||w||^2
w_squared = alpha_y @ K_sv @ alpha_y  # scalar


margin = 1 / np.sqrt(w_squared)
```

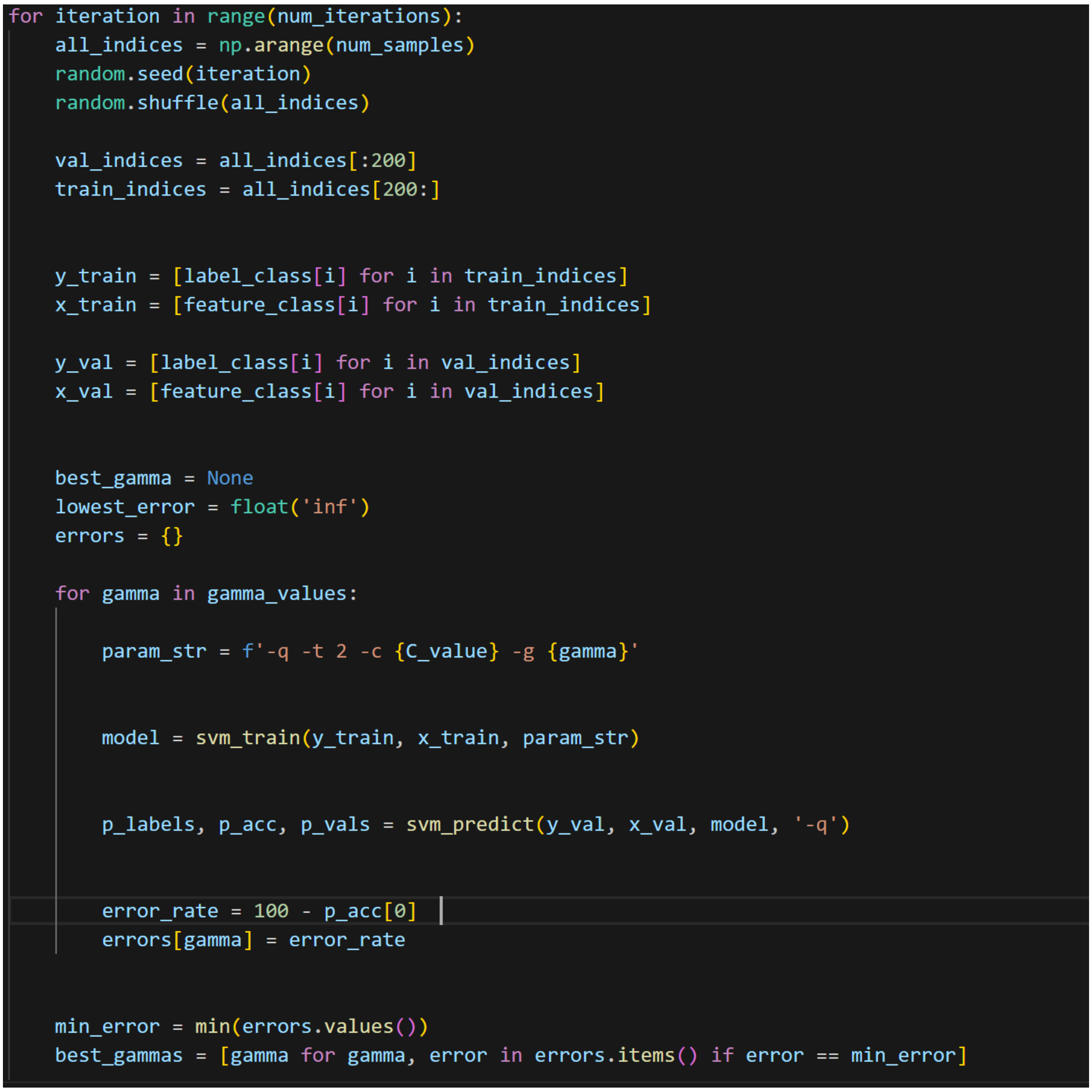
12

Phenomenon Describe : When using validation to select the hyperparameter γ, it is observed that smaller γ values, such as γ=0.01, often result in better performance. This behavior reflects the principle that a more complex model, while achieving a lowerEin , does not necessarily generalize well to unseen data, leading to a suboptimal Eout . A γ produces a smoother decision boundary by spreading the influence of each support vector over a larger area, reducing the risk of overfitting. This is particularly important in datasets with noise, where a complex model induced by a larger γ might overfit the noise instead of capturing the underlying patterns. Similarly, when the dataset is small or lacks sufficient representation of the full data distribution, a simpler model generalizes better by avoiding overfitting to limited data points. Thus, the observation that smaller γ values yield better results can be attributed to their ability to mitigate overfitting, especially in noisy or insufficiently large datasets.

Graph:



Code:



```
for iteration in range(num_iterations):
    all_indices = np.arange(num_samples)
    random.seed(iteration)
    random.shuffle(all_indices)

    val_indices = all_indices[:200]
    train_indices = all_indices[200:]


    y_train = [label_class[i] for i in train_indices]
    x_train = [feature_class[i] for i in train_indices]

    y_val = [label_class[i] for i in val_indices]
    x_val = [feature_class[i] for i in val_indices]


    best_gamma = None
    lowest_error = float('inf')
    errors = {}

    for gamma in gamma_values:

        param_str = f'-q -t 2 -c {C_value} -g {gamma}'


        model = svm_train(y_train, x_train, param_str)


        p_labels, p_acc, p_vals = svm_predict(y_val, x_val, model, '-q')


        error_rate = 100 - p_acc[0]
        errors[gamma] = error_rate


    min_error = min(errors.values())
    best_gammas = [gamma for gamma, error in errors.items() if error == min_error]
```