Arda C. Bati
A53284500

ECE 271B HW#4

Problem 1

a) $x \in \mathbb{R}^n \quad \min_x \frac{1}{2}\|x\|^2$ s.t. $\sum_i x_i \leq -3$

We should form the Lagrangian

$$L(x,\lambda) = \frac{1}{2}\|x\|^2 + \lambda\left(\sum_i x_i + 3\right)$$

Setting the gradient to zero

$$\nabla_x L = \nabla_x x + \nabla_x \lambda\left(\sum_i x_i\right) \rightarrow \frac{\partial}{\partial x_i}(\lambda \sum_i x_i) = \lambda$$

$$\nabla_x(\lambda \sum_i x_i) = \lambda$$

$$\forall i, \quad \frac{\partial L}{\partial x_i} = x_i + \lambda = 0$$

$$\nabla_\lambda L = \sum_i x_i + 3 = 0 \quad \Leftrightarrow \quad \text{KKT}$$

Assume $\sum_i x_i \leq -3$ constraint is active, therefore $\lambda > 0$

$$x_i = -\lambda, \quad \sum_i x_i = -3 = n x_i = -3, \quad x_i = -\frac{3}{n}, \quad \lambda = \frac{3}{n}, \forall i$$

(Alternative: If $\sum_i x_i < -3$, means inactive constraint, $\lambda = 0 = x_i$, then $\sum_i x_i > 0$, conflicting)

We should also check the Hessian to be P.S.D.

b) Prove the inequality $(x^T y)^2 \leq (x^T Q x)(y^T Q^{-1} y)$ where Q is a P.S.D symmetric matrix.

Hint: $\max_x y^T x$ subject to $x^T Q x \leq 1$

→ I will use this approach. For this minimization, the first step is to build the Lagrangian.

$L(x,\lambda) = y^T x + \lambda(x^T Q x - 1)$, setting gradient to 0:

$\nabla_x L = y + 2\lambda Q x = 0$, $\nabla_\lambda L = x^T Q x - 1 = 0$

Assume the constraint is inactive, $\lambda = 0$, $y = 2\lambda Q x = 0$.
$y = 0$ doesn't make much sense for this problem.
Therefore I will assume $y \neq 0$, $\lambda > 0$, constraint active.

Then, $0 = y + 2\lambda Q x = x^T y + 2\lambda x^T Q x$ | $x^T Q x = 1$

$0 = x^T y + 2\lambda$, $\lambda = -\frac{1}{2} x^T y$

$\lambda^T \cdot 2$

$y + 2\lambda Q x = 0 = Q^{-1} y + 2\lambda x = y^T Q^{-1} y + 2\lambda y^T x = 0$

$y^T Q^{-1} y = \lambda\lambda^T = \lambda^2$, $\lambda = -\frac{1}{2} x^T y = -\frac{1}{2}\sqrt{y^T Q^{-1} y}$

$x \leftrightarrow y$

$(x^T y)^2 \leq y^T Q^{-1} y$ for $x^T Q x \leq 1$ follows from here

$((y^T x)^2)^T = (x^T y)^2 \leq x^T Q x$, with both of these

$(x^T y)^2 \leq (x^T Q x)(y^T Q^{-1} y)$

Problem 2

First step here is to find the feasible sets:

(1) $f + g = 2x_1 - 1$, $f - g = -x_2 + 1$

$x_1 \geq 0$, $2x_1 - 1 \geq -1$ | $x_2 \geq 0$, $-x_2 + 1 \leq 1$

$f + g \geq -1$ $\quad$ $f - g \leq 1$.

$f \geq -g - 1$ $\quad$ $f \leq 1 + g$

Feasible Sets

$f = 1 + g$

$f = -g - 1$

① $-g - 1 \leq f \leq 1 + g$

② $f = \sqrt{g}$,

③ $f(0) = 0$, $g(0) = -\frac{1}{2}$

$f(1) = -1$, $g(1) = \frac{1}{2}$

$\Big\}$ $f, g = \left\{ (0, -\frac{1}{2}), (-1, \frac{1}{2}) \right\}$

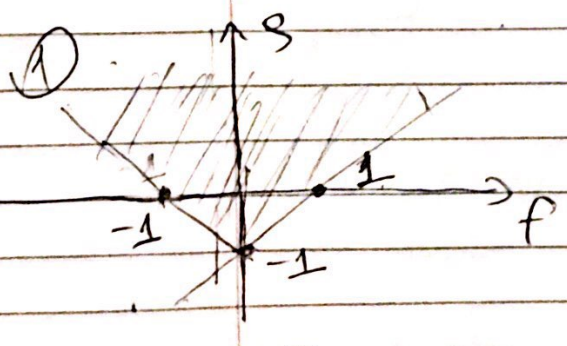
④ Same as above, but a continuous line:

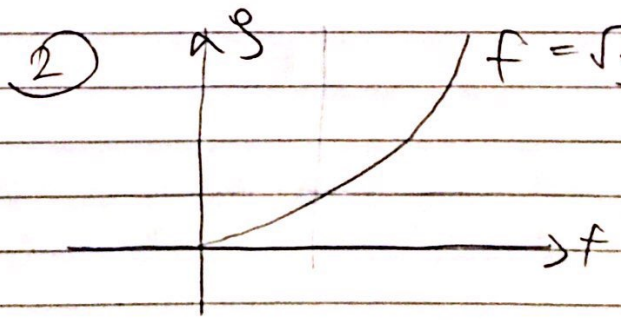
$f, g = \left[(0, -\frac{1}{2}) \quad (-1, \frac{1}{2}) \right]$

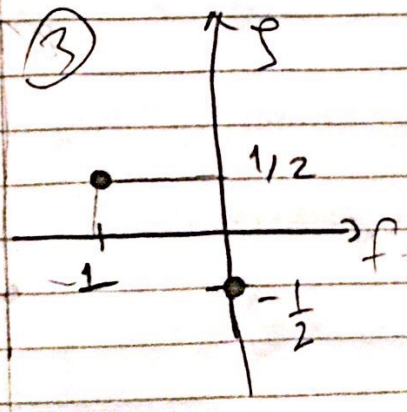
⑤ $f = \left((g+1)^2 + x_2^2 \right) \cdot \frac{1}{2}$, $x_2^2 \geq 0$, $f \geq \frac{(g+1)^2}{2}$

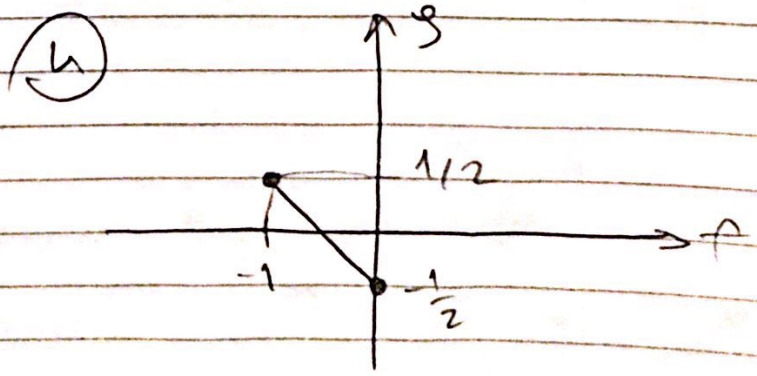
⑥ $f = |g| + x_2$, $x_2 \geq 0$, $f \geq |g|$

Sketches

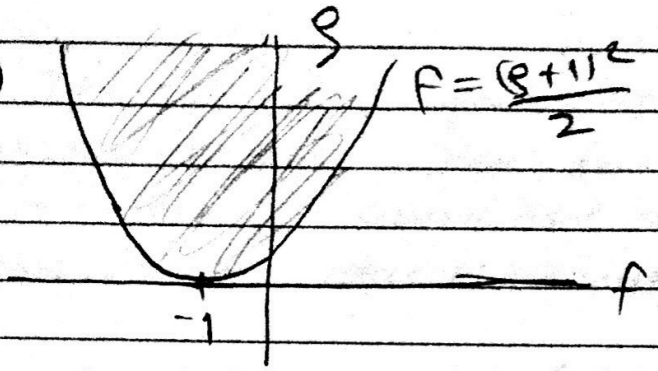




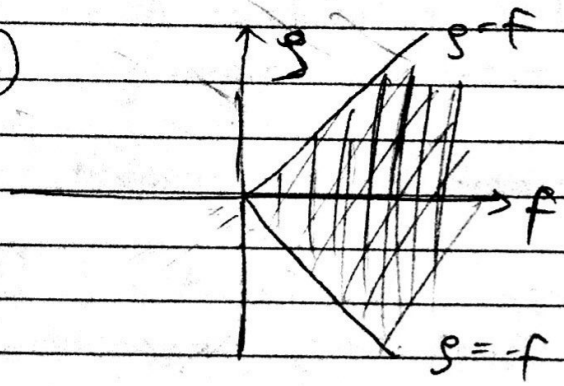




(5)



(6)



b) From the lecture slides, for $w^* = \begin{bmatrix} 1 \\ \mu^* \end{bmatrix}$, we have

two cases: $g(x^*)$ support point $\begin{cases} g(x^*) < 0, & w^* \text{ horizontal} \quad \boxed{C1} \\ g(x^*) = 0, & w^* \text{ is in the first quadrant} \quad \boxed{C2} \end{cases}$

$w^* > 0, \ \mu^* > 0$

One of these conditions must hold for the feasible sets outlined before. For each feasible set:

(1) C1 doesn't hold. C2 holds, $f = -g - 1 = 0$

supporting point $(f, g) = (-1, 0)$

In the lecture slides, we have seen that

$(x \neq x^*)$ $f(x) + \mu^* g(x) - L^* \geq 0$, $f + g + 1 = 0$, $\mu^* = 1$, $f^* = -1$

(2) The supporting plane here would be on the f-axis. However w^* can't be just on the f-axis, it should have some component on the g axis. Therefore there is no suitable Lagrange multiplier

(3) Both conditions C1, C2 don't hold. Therefore no Lagrange multiplier

(4) Again both conditions do not hold.

(5) C1 holds, the support plane is horizontal with support point $(g, f) = (-1, 0)$, $f^* = 0$

As $g(x^*) < 0$, w horizontal, $\mu^* = 0$, $w^* = (1, 0)^T$
C2 doesn't hold.

(6) C1 doesn't hold, no $g(x) < 0$ support point

C2 holds for support point $f, g = (0, 0)$

For the support plane, there can be multiple lines. They are between the horizontal and $f = -g$ lines. μ can go down until 0, and up to 1
$\mu \in (0, 1]$. $f^* = 0$.

c) For the dual problems, $q(\mu)$ is the Lagrangian dual function.

$$q(\mu) = \min_x [L(x, \mu)] = \min_x [f(x) + \mu^T g(x)]$$

↳

(1) $q(\mu) = \min_{x_1 \geq 0,\, x_2 \geq 0} x_1 - x_2 + \mu(x_1 + x_2 - 1)$

$$= \min_{x_1, x_2 \geq 0} x_1(\mu+1) + x_2(\mu-1) - \mu, \quad \mu \geq 0$$

for $\mu \geq 1$, $q(\mu) = -\mu$

$1 > \mu \geq 0$, $q(\mu) = -\infty$ $(x_2 \to \infty)$

q^* when $\mu = 1$, $q^* = -1$, $f^* = -1$ (from before)

$q^* = f^*$, no duality gap.

(2) $q(\mu) = \min_x (x + \mu x^2)$

for $\mu > 0$, $q(\mu) = -\frac{1}{4\mu}$

$\mu \leq 0$, $q(\mu) = -\infty$

$f^* = 0$ and $q^* \to 0$ for $\mu \to \infty$ therefore, eventually $f^* = q^*$ no duality gap. However as $\mu \to \infty$ is not a valid Lagrange multiplier, we aren't actually getting a sensible solution.

(3) $q(\mu) = \min_{x \in \{0,1\}} -x + \mu(x - \frac{1}{2})$

$$= \min\left(-\frac{\mu}{2}, -1 + \frac{\mu}{2}\right), \mu \geq 0$$

$q^* = -\frac{1}{2}$ but $f^* = 0$ which means duality gap therefore the dual is not useful.

(5) $q(\mu) = \min_{x_1, x_2} \frac{1}{2}(x_1^2 + x_2^2) + \mu(x_1 - 1)$

$$q(\mu) = -\frac{1}{2}\mu^2 - \mu$$

$q^* = 0$, $f^* = 0$, no duality gap, dual solutions viable.

(6) $q(\mu) = \min_{x_1 \geq 0, x_2 \geq 0} |x_1| + x_2 + \mu x_1$

for $|\mu| \leq 1$, $q(\mu) = 0$

$|\mu| > 1$, $q(\mu) = -\infty$

$q^* = 0$, $f^* = 0$, no duality gap, dual solutions are viable.

Problem 3

a) The first step is building the Lagrangian

$$L(\alpha, \beta, \gamma) = \frac{1}{2}\|w\|^2 - \nu\rho + \frac{1}{n}\sum_i \xi_i - \sum_i \alpha_i [y_i(\langle x_i, w\rangle + b) - \rho + \xi_i] - \sum_i \beta_i \xi_i - \gamma\rho$$

Setting the derivatives to 0:

$$\nabla_w L = 0, \quad w - \sum_i \alpha_i y_i x_i = 0, \quad w = \sum_i \alpha_i y_i x_i$$

$$\frac{\partial L}{\partial \xi_i} = \beta_i + \alpha_i - \frac{1}{n} = 0, \quad \beta_i + \alpha_i = \frac{1}{n}$$

$$\frac{\partial L}{\partial \rho} = \sum_i \alpha_i - \gamma - \nu = 0, \quad \sum_i \alpha_i - \gamma = \nu$$

$$\frac{\partial L}{\partial b} = \sum_i \alpha_i y_i = 0$$

Now we should put these results back into the Lagrangian.

$$L(\alpha, \beta, \gamma) = -\frac{1}{2}\|w\|^2 - \nu\rho + \frac{1}{n}\sum_i \xi_i - \sum_i \alpha_i[-\rho + \xi_i] - \sum_i \beta_i \xi_i - \gamma\rho$$

$$= -\frac{1}{2}\|w\|^2 + \frac{1}{n}\sum_i \xi_i - \sum_i \alpha_i \xi_i - \sum_i \beta_i \xi_i$$

$$= -\frac{1}{2}\sum_{i,j} \langle x_i, x_j \rangle \alpha_i \alpha_j y_i y_j$$

We have $\gamma \geq 0$, then $\sum_i \alpha_i \geq \nu$

$\beta_i \geq 0$, then $\alpha_i \leq \frac{1}{n}$

we have maximum with negative sign ↑

The resulting dual will be the minimization

$$\min_{\alpha} \frac{1}{2}\sum_{i,j} \langle x_i, x_j \rangle \alpha_i \alpha_j y_i y_j \quad \text{subject to} \quad \begin{cases} \sum_i \alpha_i \geq \nu \\ \sum_i \alpha_i y_i = 0 \\ \frac{1}{n} \geq \alpha_i \geq 0 \end{cases}$$

For the resulting decision function we will have

$$f(x) = \text{sign}\left[\sum_i \alpha_i y_i \langle x, x_i \rangle + b\right]$$

c) We have $\rho > 0$, Then $\gamma = 0$ (KKT)

$v = \sum_i \alpha_i$, Suppose there are C α_i's such that $\alpha_i = 1/n$. Then we will have $k \cdot \frac{1}{n} \leq v$

$\alpha_i = \frac{1}{n}$ is a condition for i's that have $\xi_i > 0$

$(\#i\text{'s such that } \xi_i > 0) \cdot \frac{1}{n} \leq v$

v is the upper bound for the points that cause margin error. (The first condition to prove)

$(\#S.V.) \cdot \frac{1}{n} \geq \sum_i \alpha_i = v$

↳ The highest a SV can contribute

$\frac{(\#S.V.)}{n} \geq v$, where $\frac{\#S.V}{n}$ is the fraction of the vectors that are support vectors

the previous part ↵

b) The support vectors are either on the margin or elsewhere. If they are on the margin $\xi_i = 0$.

We have $y_i(\langle x_i, w \rangle + b) = \rho - \xi_i$ in general

so specifically $y_i(\langle x_i, w \rangle + b) = \rho$ for S.V. on the margin

Summing over all S.Vs on the margin we can get $\rho \times (\#S.V \text{ on margin})$

$$\sum_{x_i \mid y_i = 1} (\langle x_i, w \rangle + b) - \sum_{x_i \mid y_i = -1} (\langle x_i, w \rangle + b) = \rho(C_1 + C_{-1})$$

where C_1 = # S.V. on the margin with $y_i = 1$, Set S_1
C_{-1} = # S.V on the margin with $y_i = -1$, Set S_{-1}

b's can be grouped the same way

$$\sum_{x \in S_1} \langle x_i, w \rangle - \sum_{x \in S_{-1}} \langle x_i, w \rangle + C_1 b - C_{-1} b = C_1 \rho + C_{-1} \rho$$

to drop the b's, we can use S_1' and S_{-1}' which are subsets of S_1 & S_{-1} such that $C_1' = C_{-1}'$, S_1', S_{-1}'

$$= \underline{\Sigma} \langle \rangle - \underline{\Sigma} \langle \rangle = 2 \rho C_1'$$

$$\rho = \frac{1}{2C_1'} \left[\sum_{x \in S_1'} \langle x_i, w \rangle - \sum_{x \in S_{-1}'} \langle x_i, w \rangle \right]$$

$$\sum_{x \in S_1'} \langle x_i, w \rangle = C_1' (\rho - b), \quad b = \rho - \frac{1}{C_1'} \sum \langle x_i, w \rangle$$

$$b = -\frac{1}{2C_1} \sum_{x \in S_1' \cup S_{-1}'} \langle x_i, w \rangle$$

d) We can minimize the 2nd problem and fix ρ to a fixed value, the optimal when minimizing over the other variables w, ξ, b.

For $C = 1$, this will give the optimal w and ξ that are applicable to the first problem.

The only difference left is the constraint difference:

$y_i (\langle x_i, w \rangle + b) \geq 1 - \xi_i$ vs $y_i (\langle x_i, w \rangle + b) \geq \rho - \xi_i$

Changing the variables as: $w_{new} = \frac{w}{\rho}$, $\xi_{i\,new} = \frac{\xi_i}{\rho}$, $b_{new} = \frac{b}{\rho}$

solves the difference.

The first problem $\min\left(\frac{1}{2}\|w\|^2 + \frac{c}{n}\sum_{i=1}^{n}\xi_i\right)$

becomes: $\min\left(\frac{1}{2}\|w_{new}\|^2 + \frac{c}{n}\sum_{i=1}^{n}\xi_{i\,new}\right)$

where $c = \frac{1}{\rho}$, as required

Programming Assignment

For this report I will first provide the written answers, then include all the results at the end.

5) a)

Accuracy results:

From the linear SVM results, we can see that the results can change significantly between different classifiers. Some digits are harder to classify than others, especially 8 and 9. Therefore their respective classifiers make more errors compared to others. Digit 1's classifier is the most successful, which can be expected as digit 1 has a very simple shape. 0 and 4's classifiers are also more successful than most others, perhaps because of their unique shapes. I couldn't notice an important difference made by the C values.

Support Vector counts:

The number of support vectors required for each classifier is closely correlated to the accuracy. Harder digits, that lead to lower accuracy require typically more support vectors. For example, to classify 8 and 9 correctly, their respective classifiers have to use more support vectors compared to others.

Looking at the support vectors with highest lagrange multipliers:

For different C values, we see slightly different support vectors, but there many same ones as well. As expected, vectors with high Lagrange multipliers are the ones that are hardest to classify. This can be seen from the digits themselves. For the digits of the correct type, they are hard to recognize and have similarities to other digits. For the digits of wrong type, they are quite close to the original digit the classifier is looking for. These support vectors are very close to the decision boundary, which makes them harder to classify, from a geometric perspective.

5) b)

We see quite similar CDF graphs for different C values. There are minor differences between each classifier. The CDF's mostly have a sigmoid like structure, including both negative and positive margins. The points with positive margin are more than the ones with negative, as expected from the accuracy results. Classifiers that make more errors (harder digits) have more negative margin values, and a higher minimum margin value.

5) c)

The best C and gamma pair I obtained was C = 4, gamma = 0.015625. The first observation for the RBF Kernel is that the accuracy results are overall better than the Linear SVM. This is expected, as we are using a more sophisticated method. This is also represented by the increase in the number of support vectors. The gamma value chosen would be quite important in the SV counts as well. There is a tradeoff here between better accuracy results and a simpler classifier. If we want to go for the accuracy, we can use rbf instead of linear svm, accepting the more complex classifiers.

The rest of the results are similar to Linear SVM. Support vectors with highest Lagrangian values change, but the overall idea is the same. For the margin cdf's the overall structure of the graphs are mostly similar. There are lesser points with negative margin, compared to the Linear SVM. This is expected as we are getting better accuracy values compared to Linear SVM.

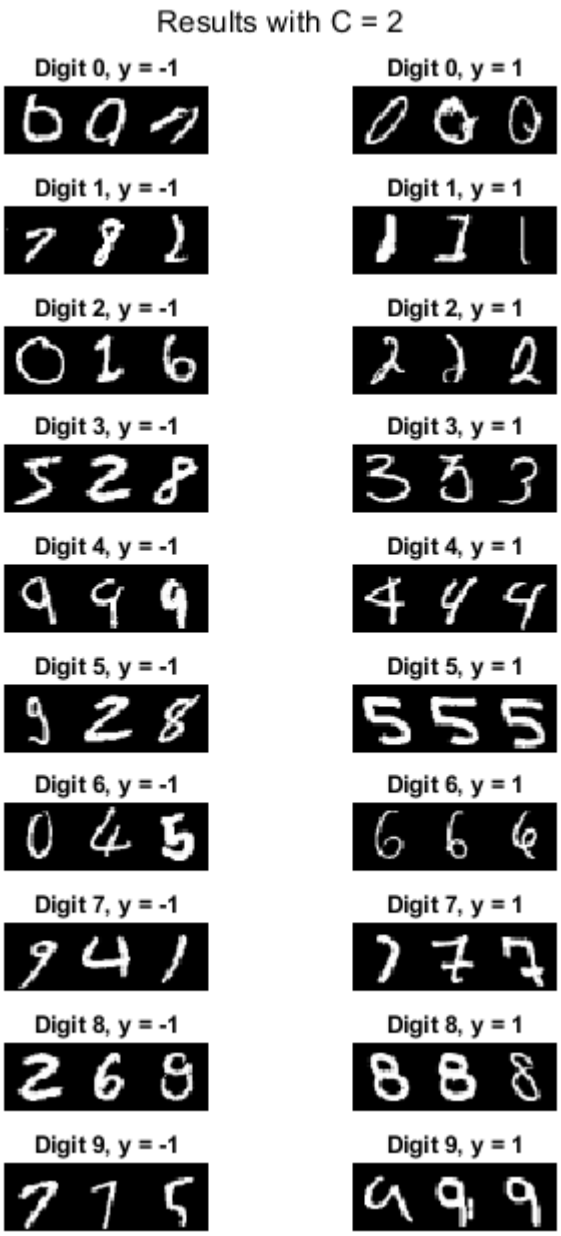
Linear SVM accuracy results: Overall: 97.21%

	0	1	2	3	4	5	6	7	8	9	Overall
C = 2	98.73	99.32	97.96	97.47	98.14	97.60	98.07	98.28	95.71	96.31	90.76%
C = 4	98.68	99.22	97.92	97.42	98.02	97.49	98.04	98.21	95.59	96.31	90.43%
C = 5	98.54	99.13	97.87	97.48	97.96	97.36	97.92	98.08	95.55	96.36	90.13%

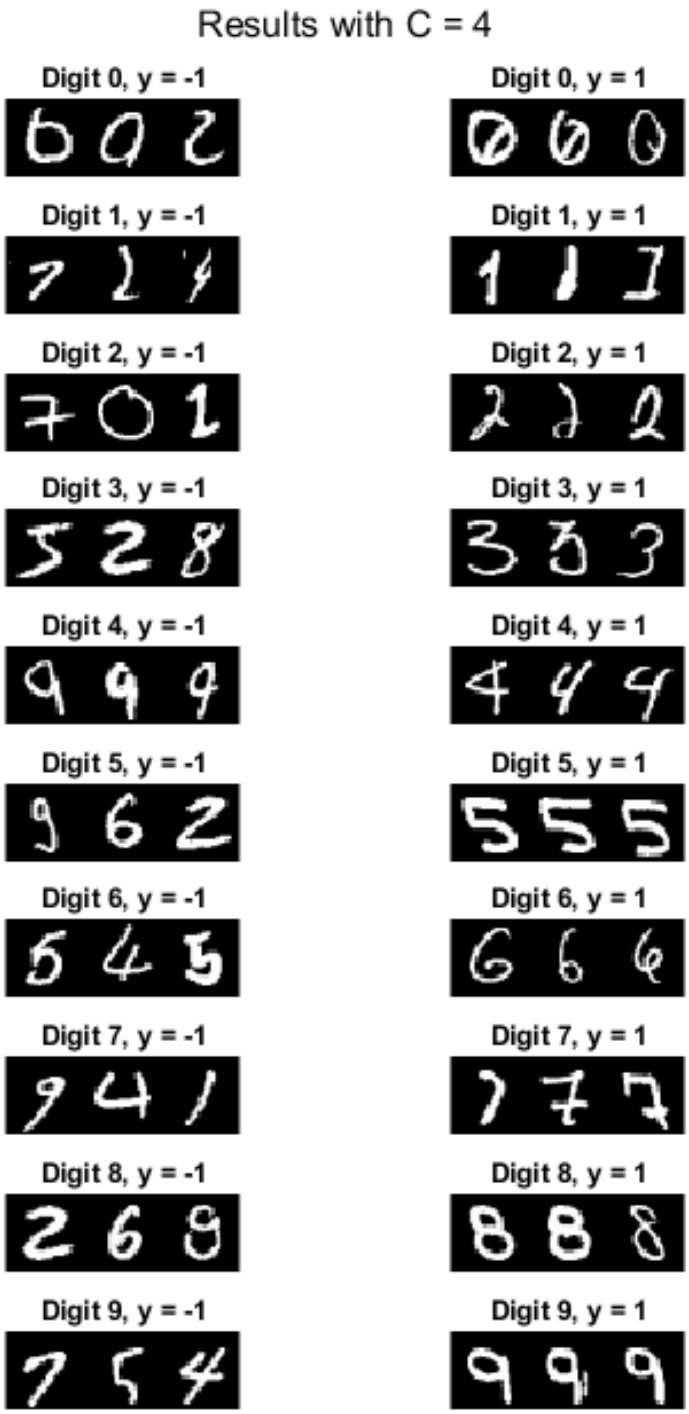
Linear SVM Support Vector counts:

	0	1	2	3	4	5	6	7	8	9
C = 2	464	505	1211	1422	900	1326	686	779	2093	1848
C = 4	455	486	1201	1416	880	1292	671	770	2097	1836
C = 5	436	455	1189	1400	869	1253	651	744	2077	1816

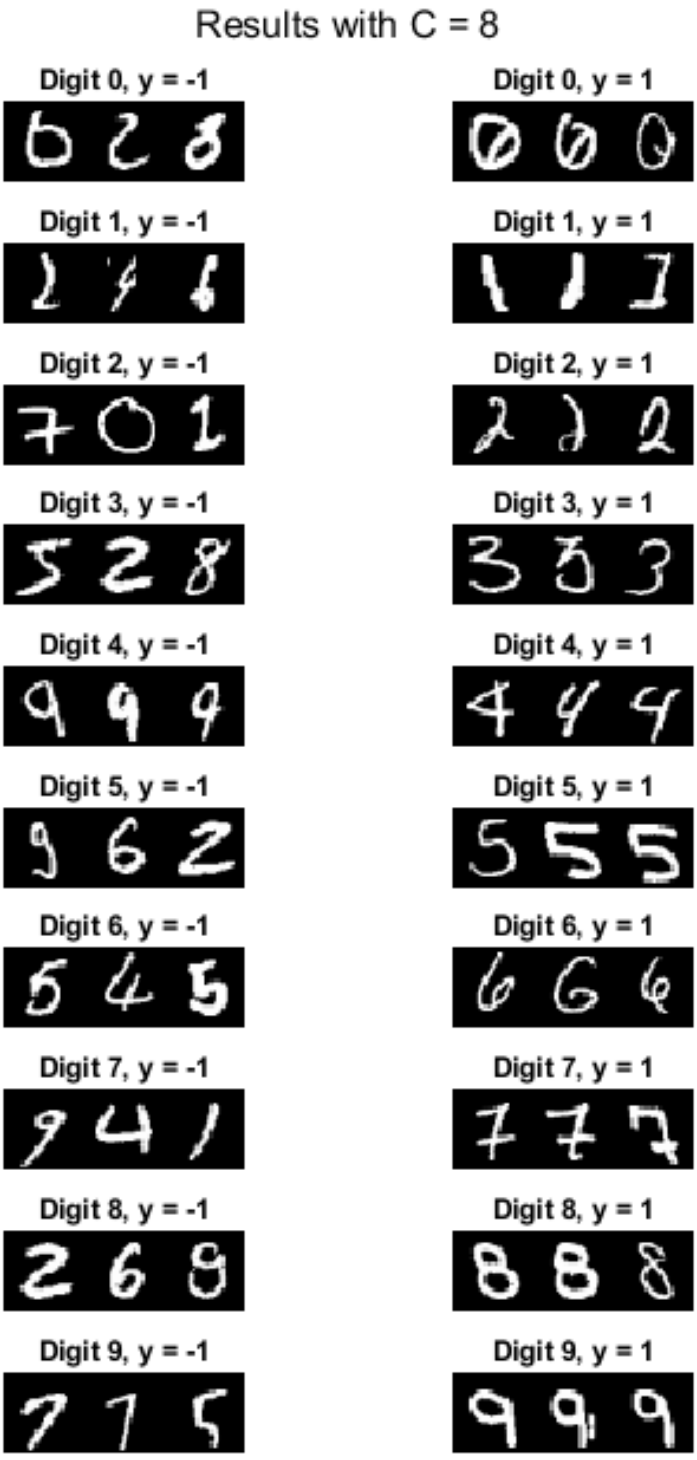
Linear SVM, 3 examples of largest Lagrange multiplier on both sides



Linear SVM, 3 examples of largest Lagrange multiplier on both sides

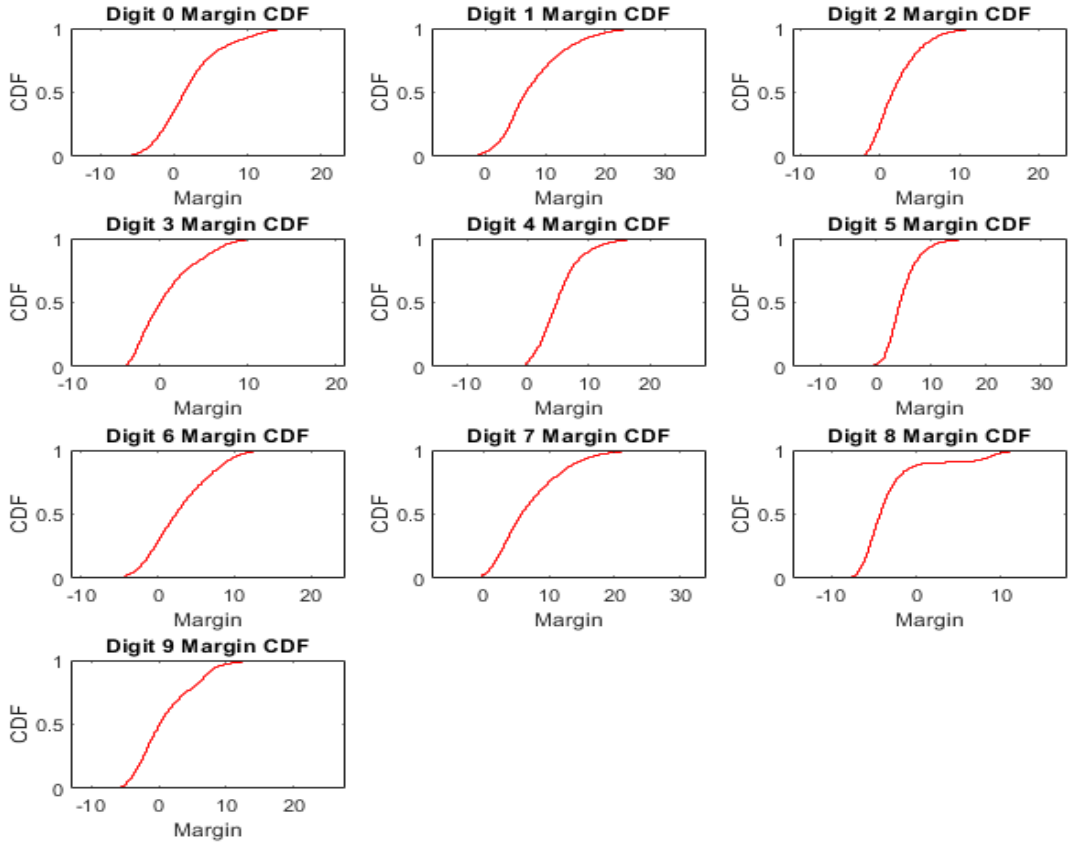


Linear SVM, 3 examples of largest Lagrange multiplier on both sides

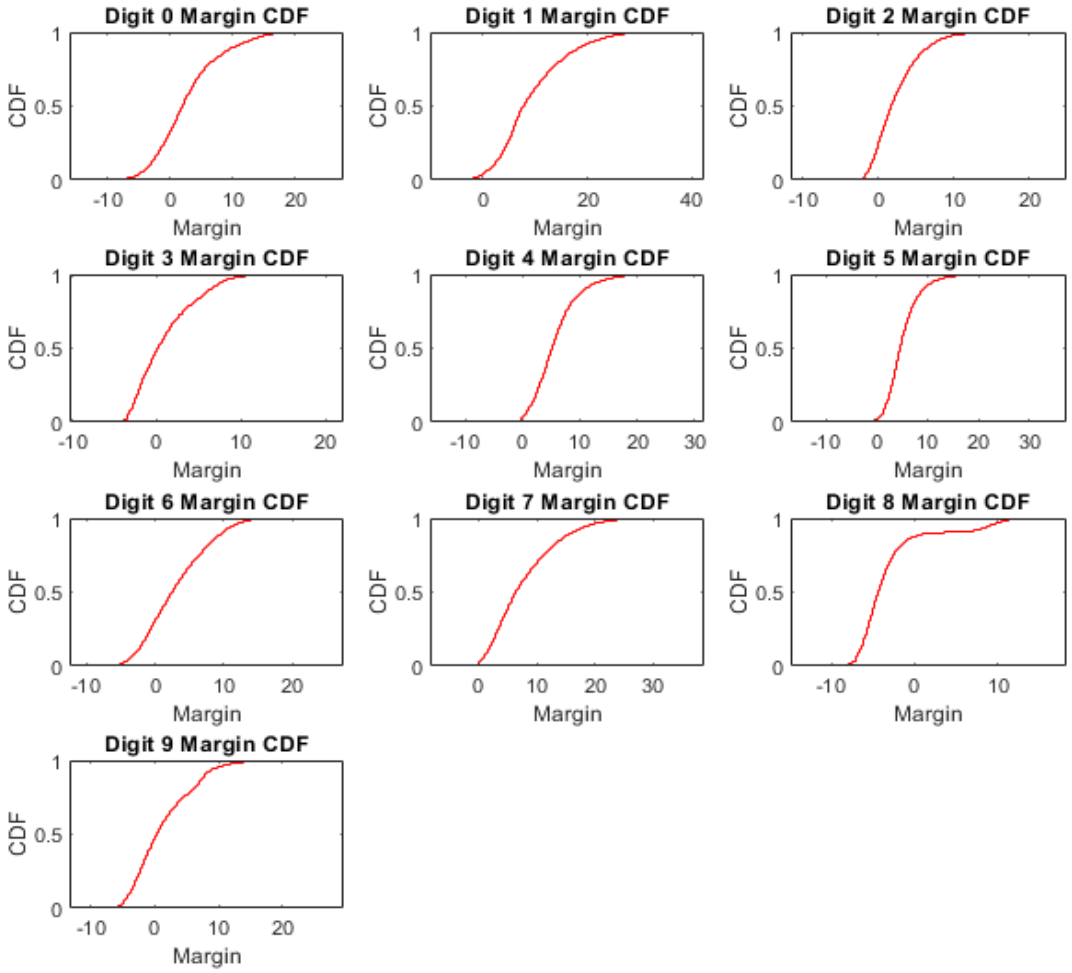


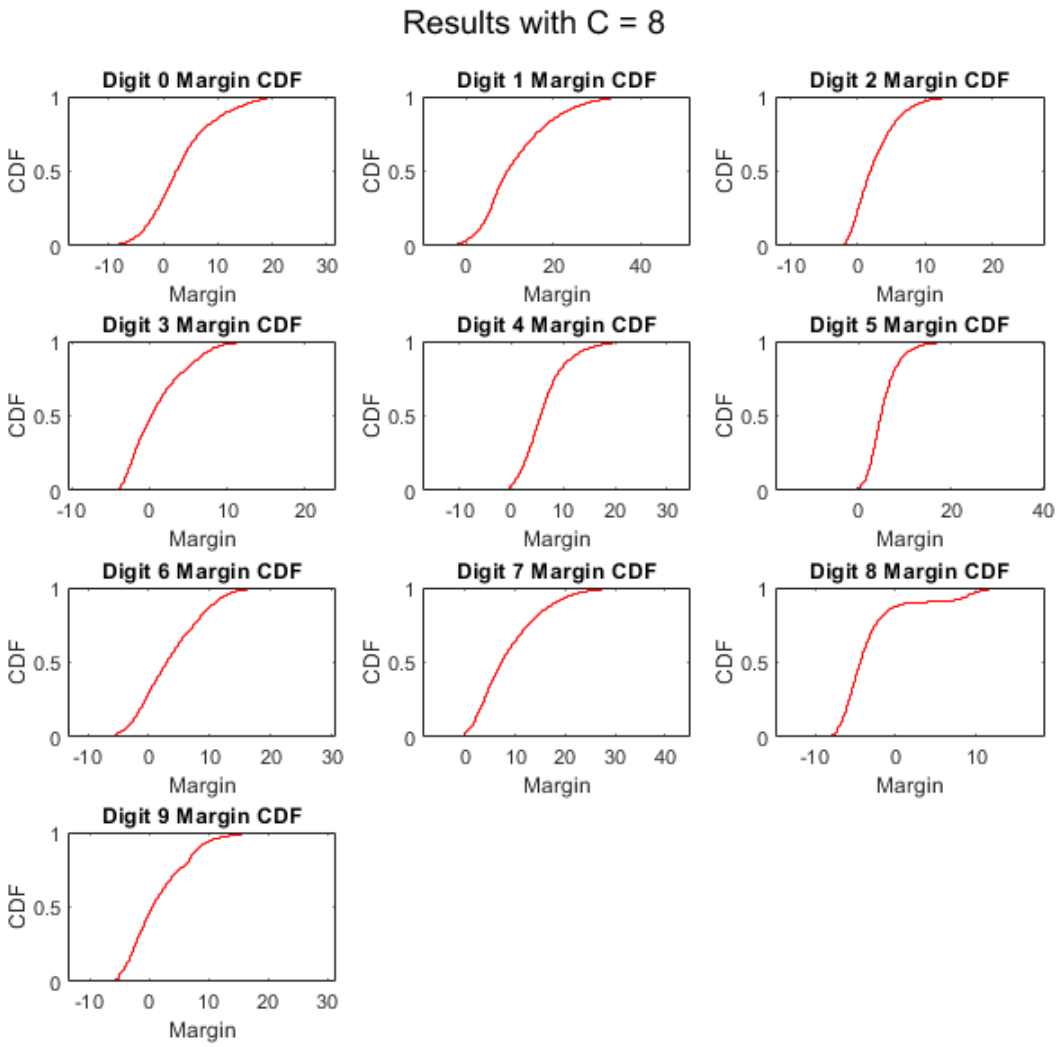
Below Margin CDF's for linear SVM are given:

Results with C = 2



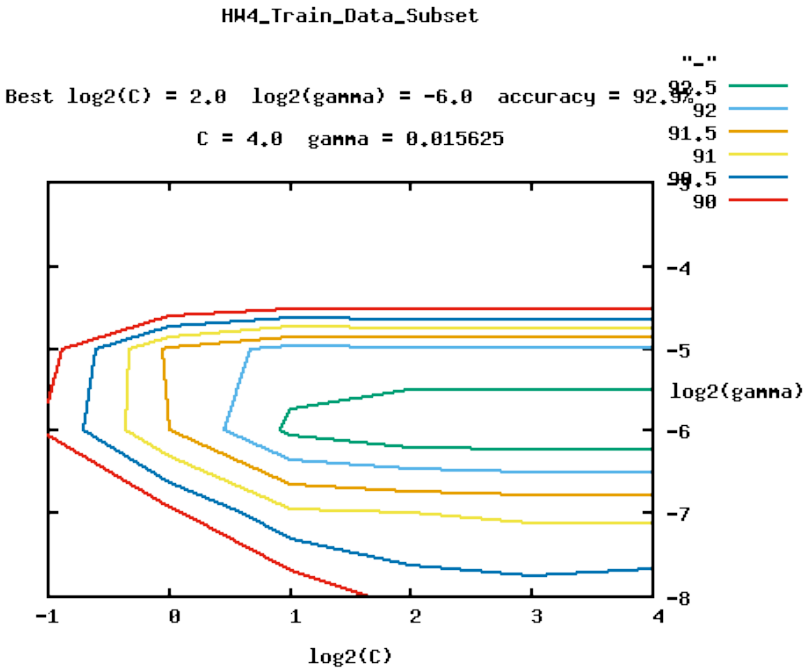
Results with C = 4





RBF Kernel

Best C, gamma pair: C = 4, gamma = 0.015625



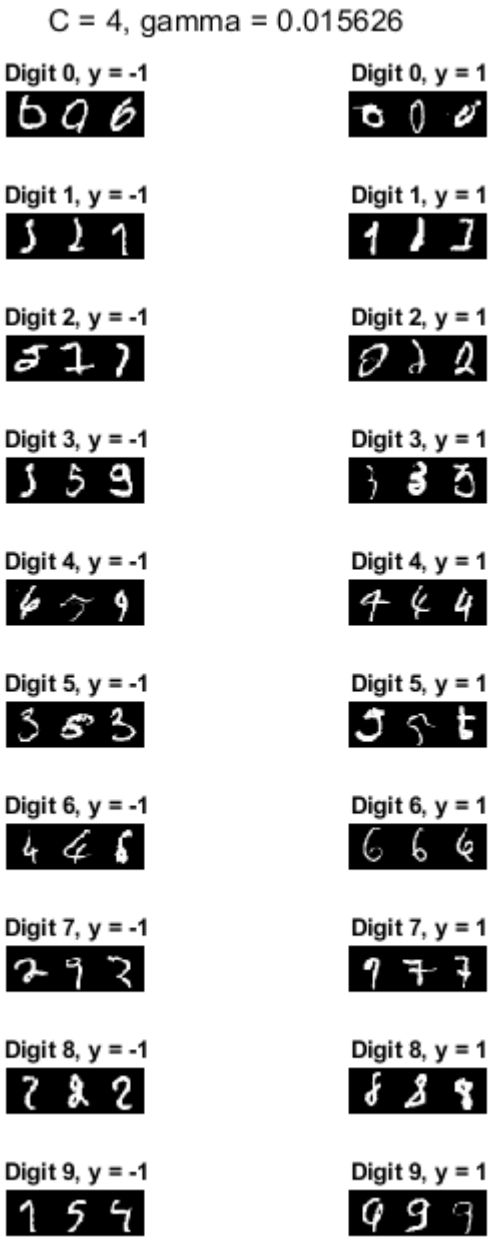
RBF Kernel accuracy results: (Overall Accuracy: 97.21%)

	0	1	2	3	4	5	6	7	8	9
C = 4, gamma = 0.15625	99.71	99.77	99.41	99.48	99.48	99.50	99.55	99.32	99.30	99.11

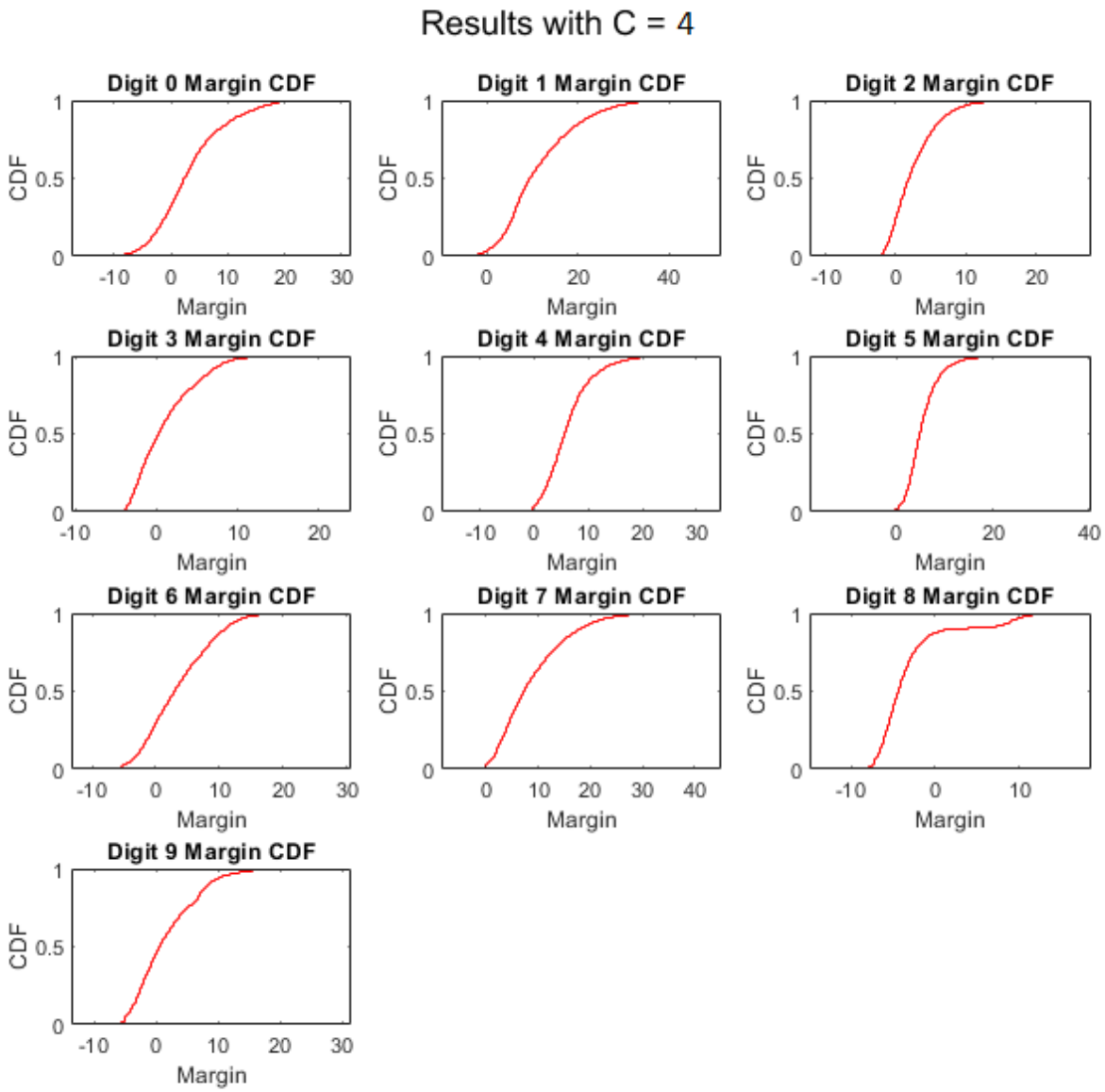
RBF Kernel SVM Counts

	0	1	2	3	4	5	6	7	8	9
C = 4, gamma = 0.15625	3293	2083	4925	4732	3846	3487	3958	4836	5983	4986

RBF Kernel, 3 examples of largest Lagrange multiplier on both sides



Below Margin CDF's for RBF Kernel are given:



```
clc;
clear all;
% trainImg = 'train-images.idx3-ubyte';
% trainLabel = 'train-labels.idx1-ubyte';
%
% testImg = 't10k-images.idx3-ubyte';
% testLabel = 't10k-labels.idx1-ubyte';
%
% [train_imgs, train_labels] = readMNIST(trainImg, trainLabel, 20000,
 0);
% [test_imgs, test_labels] = readMNIST(testImg, testLabel, 10000, 0);
%
% save HW4_Data

load HW4_Data

accuracies = zeros(3,10);
sv_counts = zeros(3,10);
% 3 different C values
margins_linear = zeros(30,20000);

loop = 0;
C_count = 0;
tic
for C = [2,4,8]
    C_count = C_count + 1;
    figure()
    sgtitle(sprintf('Results with C = %d',C));
    [ha, ~] =  tight_subplot(10,2,[0.04 0.001],[.01 .1],[.01 .01]);
    inner_loop = -1;
    for classifier = 1:10     % 10 different classifiers
        loop = loop + 1
        inner_loop = inner_loop + 2;
        tr_labels = train_labels;
        tr_labels(tr_labels ~= classifier - 1) = -1;
        tr_labels(tr_labels ~= -1) = 1;

        %Linear SVM
        svm_model = svmtrain(tr_labels, sparse(train_imgs), sprintf('-
q -t 0 -c %d',C));

        te_labels = test_labels;
        te_labels(te_labels ~= classifier - 1) = -1;
        te_labels(te_labels ~= -1) = 1;

        [~, accuracy, ~] = svmpredict(te_labels, sparse(test_imgs),
 svm_model, '-q');
        sv_counts(C_count, classifier) = svm_model.totalSV;
        accuracies(C_count, classifier) = accuracy(1);

        [coefficients, indices] = sort(svm_model.sv_coef);
        svms = svm_model.sv_indices;
```

```
        negative_coefs = svms(indices(1:3));
        positive_coefs = svms(indices(size(indices)-2 :
 size(indices)));

        loop_image1 = [];
        loop_image2 = [];

        for i = 1:3
            loop_image1 = [loop_image1;
 reshape(train_imgs(negative_coefs(i),:),[28 28])];
            loop_image2 = [loop_image2;
 reshape(train_imgs(positive_coefs(i),:),[28 28])];
        end

%        axes(ha(inner_loop)); imshow(loop_image1');
%        title(sprintf('Digit %d, y = -1',classifier-1));
%        axes(ha(inner_loop+1)); imshow(loop_image2');
%        title(sprintf('Digit %d, y = 1',classifier-1));

        w = (svm_model.sv_coef)'*(svm_model.SVs);
        b = svm_model.rho;
        margins_linear(loop,:) = tr_labels' .* (w*(train_imgs') + b);
    end

%     svm_model = svmtrain(train_labels, sparse(train_imgs),
 sprintf('-q -t 0 -c %d',C));
%     [~, accuracy, ~] = svmpredict(test_labels, sparse(test_imgs),
 svm_model, '-q');
end
toc

save HW4_Linear_Results
load HW4_Linear_Results

no_points = 1000;
loop = 0;
for C = [2,4,8]
    figure();
    sgtitle(sprintf('Results with C = %d',C));
    for i = 1:10
        index = loop*10 + i;
        lower = min(margins_linear(index,:))-3;
        upper = max(margins_linear(index,:))+3;
        edges = linspace(lower,upper,no_points);
        subplot(4,3,i);
        h1 =
 histogram(margins_linear(index,:),no_points,'BinEdges',edges,'Normalization','cdf
        xlim([lower upper-0.01])
        title(sprintf('Digit %d Margin CDF',i-1)); xlabel('Margin');
 ylabel('CDF');
    end
    loop = loop + 1;
end
% [ha, pos] = tight_subplot(3,2,[.01 .03],[.1 .01],[.01 .01])
```

```
% for ii = 1:6; axes(ha(ii)); plot(randn(10,ii)); end
% set(ha(1:4),'XTickLabel',''); set(ha,'YTickLabel','')

libsvmwrite('HW4_Train_Data', train_labels, sparse(train_imgs));

data_size = 20000;
sample_size = 1000;
random_indices = randperm(data_size,sample_size);
%random_indices = sort(random_indices);
libsvmwrite('HW4_Train_Data_Subset', train_labels(random_indices),
 sparse(train_imgs(random_indices,:)));

% grid.py used to determine C, y
% Anaconda prompt
% cd C:\Users\Asus\Desktop\WI19 Courses\ECE 271B\MATLAB\MATLAB_HW4
% python grid.py -svmtrain "C:\Users\Asus\Desktop\WI19 Courses\ECE
 271B\MATLAB\libsvm-3.23\libsvm-3.23\windows\svm-train.exe" -gnuplot
 "C:\Program Files\gnuplot\bin\gnuplot.exe" -log2c 0,4,1 -log2g
 -8,-3,1 -v 5 -m 300 HW4_Train_Data_Subset
% Command:
% Train Subset size 200:  Output: C=2.0, gamma=0.0625,   Accuracy=54.0
% Train Subset size 1000: Output: C=4.0, gamma=0.015625, Accuracy=92.7
% Values to be chosen C = 4, gamma = 0.015625

C = 4; gamma = 0.015626;

margins_rbf = zeros(30,20000);

accuracies_rbf = zeros(10,1);
sv_counts_rbf = zeros(10,1);

figure()
sgtitle(sprintf('RBF Kernel, C = %d, gamma = %f',C,gamma));
[ha, pos] =  tight_subplot(10,2,[0.06 0.001],[.01 .1],[.01 .01]);
inner_loop = -1;
tic
for classifier = 1:10
    % 10 different classifiers
        classifier
        inner_loop = inner_loop + 2;
        tr_labels = train_labels;
        tr_labels(tr_labels ~= classifier - 1) = -1;
        tr_labels(tr_labels ~= -1) = 1;

        %Linear SVM
        svm_model = svmtrain(tr_labels, sparse(train_imgs), sprintf('-
q -t 2 -c %d -g %f',C,gamma));

        te_labels = test_labels;
        te_labels(te_labels ~= classifier - 1) = -1;
        te_labels(te_labels ~= -1) = 1;

        [~, accuracy, ~] = svmpredict(te_labels, sparse(test_imgs),
 svm_model, '-q');
```

```
        sv_counts_rbf(classifier) = svm_model.totalSV;
        accuracies_rbf(classifier) = accuracy(1);

        [coefficients, indices] = sort(svm_model.sv_coef);
        svms = svm_model.sv_indices;
        negative_coefs = svms(indices(1:3));
        positive_coefs = svms(indices(size(indices)-2 :
 size(indices)));

        loop_image1 = [];
        loop_image2 = [];

        for i = 1:3
            loop_image1 = [loop_image1;
 reshape(train_imgs(negative_coefs(i),:),[28 28])];
            loop_image2 = [loop_image2;
 reshape(train_imgs(positive_coefs(i),:),[28 28])];
        end

        axes(ha(inner_loop)); imshow(loop_image1');
        title(sprintf('Digit %d, y = -1',classifier-1));
        axes(ha(inner_loop+1)); imshow(loop_image2');
        title(sprintf('Digit %d, y = 1',classifier-1));

        w = (svm_model.sv_coef)'*(svm_model.SVs);
        b = svm_model.rho;

        margins_rbf(classifier,:) = tr_labels' .* (w*(train_imgs') +
 b);
end
%     svm_model = svmtrain(train_labels, sparse(train_imgs),
 sprintf('-q -t 0 -c %d',C));
%     [~, accuracy, ~] = svmpredict(test_labels, sparse(test_imgs),
 svm_model, '-q');
toc

save HW4_RBF_Results
load HW4_RBF_Results

no_points = 1000;
loop = 0;
figure();
C = 4;
sgtitle(sprintf('RBF Kernel, C = %d, gamma = %f',C,gamma));
for i = 1:10
    index = loop*10 + i;
    lower = min(margins_rbf(index,:))-3;
    upper = max(margins_rbf(index,:))+3;
    edges = linspace(lower,upper,no_points);
    subplot(4,3,i);
    h1 =
 histogram(margins_rbf(index,:),no_points,'BinEdges',edges,'Normalization','cdf','
    xlim([lower upper-0.01])
```

```
    title(sprintf('Digit %d Margin CDF',i-1)); xlabel('Margin');
 ylabel('CDF');
end
loop = loop + 1;
```

Published with MATLAB® R2018b