Q1. a) For the i^{th} data point → x_i, y_i obj func:

$$L(x_i, y_i; w) = \frac{1}{2}\|w\|^2 + C\,\Sigma_i \max(0, 1 - y_i(w^T x_i + b))$$

$$\nabla_w L(x_i, y_i) = \nabla \frac{1}{2}\|w\|^2 / \nabla w + C\nabla(\max(0, 1 - y_i(w^T x_i + b)) / \nabla w$$

Computing each gradient:

$$\nabla \frac{1}{2}\|w\|^2 / \nabla w = w$$

$$\nabla(\max(0, 1 - y_i(w^T x_i + b))) / \nabla w = \begin{cases} 0 & \text{if } y_i(w^T x_i + b) \geq 1 \\ -y_i(x_i) & \text{if } y_i(w^T x_i + b) < 1 \end{cases}$$

∴ grad of obj func

$$\nabla_w L(x_i, y_i) = \begin{cases} \lambda w - y_i(x_i) & \text{for } y_i(w^T x_i + b) < 1 \\ \lambda w & \text{o.w} \end{cases}$$

$$L(B;w) = \frac{1}{2}\|w\|^2 + C \sum \max(0, 1 - y_i(w^T x_i + b))$$

$$\nabla_w L(B) = \nabla\left(\frac{1}{2}\|w\|^2\right) / \nabla w + C\,\nabla\left(\sum \max(0, 1 - y_i(w^T x_i + b))\right) / \nabla w$$

Computing separately:

$$\nabla \frac{1}{2}\|w\|^2 / \nabla w = w$$

$$\nabla\left(\sum \max(0, 1 - y_i(w^T x_i + b))\right) / \nabla w = \sum \nabla(\max(0, 1 - y_i(w^T x_i + b))) / \nabla w$$

$\therefore$

$$\nabla_w L(B) = \begin{cases} \lambda w - \frac{1}{|B|} y_i(x_i) & \text{for } y_i(w^T x_i + b) < 1 \\ \lambda w & 0 \cdot w \end{cases}$$

$/B \rightarrow \because$ across batch size.

Expressions ⇒ P, q, G, h, A, b

<u>Q2</u> a)

Dual Problem ⇒ min $\frac{1}{2} x^T P x + q^T x$

Subject to $Gx \leq h$

$Ax = b$

here x = vector with len = no. of training samp.

Obj func ⇒ max dual Lagragian fun.

optimization variable (len = training ex)

L-D(α)

$$L\text{-}D(\alpha) = \text{sum}_i\, \alpha_i - \frac{1}{2}\, \text{sum}_i\, \text{sum}_j\, \alpha_i \alpha_j\, y_i y_j\, K(x_i, x_j)$$

$y_i = +1$ or -1 ; $x_i \rightarrow$ feature vector ; $K(x_i, x_j)$ ⇒ kernel func.

$$\therefore \min \frac{1}{2} \alpha^T P \alpha - q^T \alpha$$

where → $\boxed{P_{ij} = y_i y_j K(x_i, x_j)}$; $\boxed{q_i = -1}$

$\alpha_i \geq 0$ for all i

$\text{sum}_1, \alpha_i, y_i = 0$

$\therefore$ $\boxed{G = [-I_m : I_m]}$ → I_m = identity matrix

$\boxed{h = [0 : C]}$ → C = hyperparameter (trade-off b/w max, min)

$\boxed{A = [y_1, y_2 \cdots y_m]}$; $\boxed{b = 0}$

$\therefore$ min $\frac{1}{2} \alpha^T P \alpha - q^T \alpha$.

subj to $G\alpha \leq h$

$A\alpha = b$

$P_{ij} = y_i y_j K(x_i, x_j)$

$q_i = -1$

$G = [-I_m : I_m]$

$h = [0 : C]$

$A = [y_1, y_2 \cdots y_m]$

$b = 0$

Q1) Code Results:

```
Epoch: 588 | Train acc: 0.448 | Test acc: 0.447
Epoch: 589 | Train acc: 0.552 | Test acc: 0.553
Epoch: 590 | Train acc: 0.448 | Test acc: 0.447
Epoch: 591 | Train acc: 0.552 | Test acc: 0.553
Epoch: 592 | Train acc: 0.448 | Test acc: 0.447
Epoch: 593 | Train acc: 0.552 | Test acc: 0.553
Epoch: 594 | Train acc: 0.448 | Test acc: 0.447
Epoch: 595 | Train acc: 0.552 | Test acc: 0.553
Epoch: 596 | Train acc: 0.449 | Test acc: 0.447
Epoch: 597 | Train acc: 0.552 | Test acc: 0.553
Epoch: 598 | Train acc: 0.448 | Test acc: 0.447
Epoch: 599 | Train acc: 0.552 | Test acc: 0.553
Epoch: 600 | Train acc: 0.448 | Test acc: 0.447
```

Epochs - 600
Lambda - 0.001
Batch size - 50

```
Epoch: 587 | Train acc: 0.460 | Test acc: 0.456
Epoch: 588 | Train acc: 0.454 | Test acc: 0.447
Epoch: 589 | Train acc: 0.485 | Test acc: 0.460
Epoch: 590 | Train acc: 0.553 | Test acc: 0.553
Epoch: 591 | Train acc: 0.552 | Test acc: 0.553
Epoch: 592 | Train acc: 0.568 | Test acc: 0.560
Epoch: 593 | Train acc: 0.568 | Test acc: 0.554
Epoch: 594 | Train acc: 0.552 | Test acc: 0.553
Epoch: 595 | Train acc: 0.616 | Test acc: 0.597
Epoch: 596 | Train acc: 0.631 | Test acc: 0.613
Epoch: 597 | Train acc: 0.552 | Test acc: 0.553
Epoch: 598 | Train acc: 0.559 | Test acc: 0.554
Epoch: 599 | Train acc: 0.555 | Test acc: 0.553
Epoch: 600 | Train acc: 0.575 | Test acc: 0.579
```

Epochs - 600
Lambda - 0.1
Batch size - 50

```
Epoch: 565 | Train acc: 0.520 | Test acc: 0.462
Epoch: 566 | Train acc: 0.603 | Test acc: 0.590
Epoch: 567 | Train acc: 0.578 | Test acc: 0.553
Epoch: 568 | Train acc: 0.562 | Test acc: 0.553
Epoch: 569 | Train acc: 0.564 | Test acc: 0.554
Epoch: 589 | Train acc: 0.533 | Test acc: 0.465
Epoch: 590 | Train acc: 0.505 | Test acc: 0.463
Epoch: 591 | Train acc: 0.719 | Test acc: 0.651
Epoch: 592 | Train acc: 0.557 | Test acc: 0.549
Epoch: 593 | Train acc: 0.553 | Test acc: 0.553
Epoch: 594 | Train acc: 0.556 | Test acc: 0.553
Epoch: 595 | Train acc: 0.613 | Test acc: 0.602
Epoch: 596 | Train acc: 0.555 | Test acc: 0.554
Epoch: 597 | Train acc: 0.552 | Test acc: 0.553
Epoch: 598 | Train acc: 0.565 | Test acc: 0.504
Epoch: 599 | Train acc: 0.700 | Test acc: 0.654
```

Epochs - 600
Lambda - 1.0
Batch size - 50

```
Epoch: 587 | Train acc: 0.552 | Test acc: 0.553
Epoch: 588 | Train acc: 0.552 | Test acc: 0.553
Epoch: 589 | Train acc: 0.552 | Test acc: 0.553
Epoch: 590 | Train acc: 0.552 | Test acc: 0.553
Epoch: 591 | Train acc: 0.552 | Test acc: 0.553
Epoch: 592 | Train acc: 0.552 | Test acc: 0.553
Epoch: 593 | Train acc: 0.552 | Test acc: 0.553
Epoch: 594 | Train acc: 0.552 | Test acc: 0.553
Epoch: 595 | Train acc: 0.552 | Test acc: 0.553
Epoch: 596 | Train acc: 0.552 | Test acc: 0.553
Epoch: 597 | Train acc: 0.552 | Test acc: 0.553
Epoch: 598 | Train acc: 0.552 | Test acc: 0.553
Epoch: 599 | Train acc: 0.552 | Test acc: 0.553
Epoch: 600 | Train acc: 0.552 | Test_acc: 0.553
```

Epochs - 600
Lambda - 2.0
Batch size - 50

```
Epoch: 587 | Train acc: 0.552 | Test acc: 0.553
Epoch: 588 | Train acc: 0.552 | Test acc: 0.553
Epoch: 589 | Train acc: 0.552 | Test acc: 0.553
Epoch: 590 | Train acc: 0.552 | Test acc: 0.553
Epoch: 591 | Train acc: 0.552 | Test acc: 0.553
Epoch: 592 | Train acc: 0.552 | Test acc: 0.553
Epoch: 593 | Train acc: 0.552 | Test acc: 0.553
Epoch: 594 | Train acc: 0.552 | Test acc: 0.553
Epoch: 595 | Train acc: 0.552 | Test acc: 0.553
Epoch: 596 | Train acc: 0.552 | Test acc: 0.553
Epoch: 597 | Train acc: 0.552 | Test acc: 0.553
Epoch: 598 | Train acc: 0.552 | Test acc: 0.553
Epoch: 599 | Train acc: 0.552 | Test acc: 0.553
Epoch: 600 | Train acc: 0.552 | Test_acc: 0.553
```

Epochs - 600
Lambda - 10.0
Batch size - 50

Q2) Dual Problem:

These are the results that the code produced for different C values.

```
Optimal solution found.
C value: 0.001
Train acc: 0.762
Test acc: 0.642
```

```
Optimal solution found.
C value: 0.01
Train acc: 0.941
Test acc: 0.713
```

```
Optimal solution found.
C value: 0.1
Train acc: 0.979
Test acc: 0.709
```

```
Optimal solution found.
C value: 1
Train acc: 0.985
Test acc: 0.683
```

```
Optimal solution found.
C value: 10
Train acc: 0.985
Test acc: 0.683
```

```
Optimal solution found.
C value: 100
Train acc: 0.985
Test acc: 0.683
```

d) Dual shows more accuracy compared to Pegasos. One possible reason is that the dual SVM can better handle non-linearly separable datasets by using a kernel trick to map the data into a higher-dimensional space where the data becomes separable. In contrast, Pegasos uses a linear classifier that works well only when the data is linearly separable. Therefore, if the data is not linearly separable, dual SVM can achieve better accuracy than Pegasos.

Q3. $\because$ Feature space is not linearly separable → we use kernel trick to map the data.

To find a suitable kernel func:

$$[x_1^3, x_1^2 x_2, x_1^2, x_1 x_2, x_1, x_2]$$

$$\therefore \; K(x, y) = (x_1^3)(y_1^3) + (x_1^2 x_2)(y_1^2 y_2) + (x_1^2)(y_1^2) + (x_1 x_2)(y_1 y_2) + (x_1 y_1) + (x_2 y_2)$$

↳ Kernel func.