OPT in ML Computational Assignment 2

Brian Lee 1002750855

1 Linear Support Vector Machine

(a) We get that

- The optimal decision boundary is given by the hyperplane $w^T x + b$ where

$$w = \begin{pmatrix} 1.40667 \\ 2.13320 \end{pmatrix} \quad \text{and} \quad b = -10.34500$$

- The optimal support vectors are

$$X_{12} = \begin{pmatrix} 1.9182 \\ 4.0534 \end{pmatrix}, \; X_{19} = \begin{pmatrix} 3.0357 \\ 3.3165 \end{pmatrix}, \; X_{20} = \begin{pmatrix} 1.5841 \\ 3.3575 \end{pmatrix}, \; X_{40} = \begin{pmatrix} 1.3191 \\ 3.5109 \end{pmatrix}$$

- The solution time was 0.037005 seconds

(b)

- The optimal dual solution α is too long to list here cleanly, so I list the first 2 and last 2 elements and refer you to the python code:

$$\alpha = (9.9306 \cdot 10^{-14}, 1.6898 \cdot 10^{-12}, \ldots, 1.644 \cdot 10^{-14}, 1)$$

- The optimal decision bounary hyperplane $w^T x + b$ is given by

$$w = \begin{pmatrix} 1.40667 \\ 2.13320 \end{pmatrix} \quad \text{and} \quad b = -10.17445$$

- The optimal support vectors are

$$X_4 = \begin{pmatrix} 3.5772 \\ 2.856 \end{pmatrix}, \; X_{14} = \begin{pmatrix} 2.6555 \\ 3.5008 \end{pmatrix}, \; X_{26} = \begin{pmatrix} 3.0473 \\ 2.6411 \end{pmatrix}, \; X_{44} = \begin{pmatrix} 2.4482 \\ 2.6411 \end{pmatrix}$$

  I noticed that the support vectors were different from the solutions to the primal problem, and that my bias b was also slightly different, despite w being the same, which I believe caused this discrepancy in optimal vectors. There may be an error in the way I calculated the bias in one of the functions, but I was not able to find and fix the error by the time of submission.

- The solution time was 0.00600 seconds

(c) The decision boundary will change with increased and decreased C value, since our data is not linearly separable. For increased values of C, we increase the penalty of misclassifying thus it will seek to not misclassify data, even if it yields a smaller margin hyperplane. Similarly, for decreased values of C there is less penalty of misclassifying, thus it will yield a larger margin hyperplane that is optimal but may misclassify some points.

(d) I coded a function that uses cvxpy to optimize the hard margin SVM, and returns whether the optimization is feasible or not. The data is linearly separable if and only if this optimization is feasible, as the inequality constraints are precisely the conditions that the data is linearly separable. Hence, the dataset is linearly separable if the optimal solution exists and the optimal objective function value is finite, and not linearly separable if the optimal solution doesn't exist and the optimal objective function is infinity. Using this, we determined that this dataset is not linearly separable.

(e) $\xi_i \geq 0$ is not needed since the objective function contains ξ_i^2 where $(-\xi_i)^2 = \xi_i^2$, thus the condition is superfluous.

(f)
- The optimal decision boundary $w^T x + b$ is given by

$$w = \begin{pmatrix} 0.73873 \\ 1.56042 \end{pmatrix}, \quad b = -6.81400$$

- The optimal support vectors are given by

$$X_6 = \begin{pmatrix} 3.3814 \\ 3.4291 \end{pmatrix}, \quad X_{22} = \begin{pmatrix} 1.9527 \\ 2.7843 \end{pmatrix}, \quad X_{26} = \begin{pmatrix} 3.0473 \\ 2.2931 \end{pmatrix}$$

(g) We construct the Lagrangian of the ℓ_2 norm soft margin SVM:

$$\begin{aligned}
\mathcal{L}(w, b, \xi, \alpha) &= \frac{1}{2} \|w\|_2^2 + \frac{C}{2} \sum_{i=1}^{n} \xi_i^2 + \sum_{i=1}^{n} \alpha_i (y_i (w^T x_i + b) - 1 + \xi) \\
&= \frac{1}{2} w^T w + \frac{C}{2} \xi^T \xi - w^T \sum_{i=1}^{n} \alpha_i y_i x_i - b\alpha^T y + \mathbf{1}_n^T \alpha - \xi^T \alpha \\
&= \frac{1}{2} w^T w + \frac{C}{2} \xi^T \xi - w^T z^T \alpha - b\alpha^T y + \mathbf{1}_n^T \alpha - \xi^T \alpha \qquad \text{where } z_i = y_i x_i
\end{aligned}$$

Taking partial derivatives, we get

$$\frac{\partial \mathcal{L}}{\partial w} = w - z^T \alpha = 0 \quad \Rightarrow \quad w = z^T \alpha$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\alpha^T y = 0$$

$$\frac{\partial \mathcal{L}}{\partial \xi} = C\xi - \alpha = 0 \quad \Rightarrow \quad \xi = \frac{\alpha}{C}$$

Plugging these conditions into the Lagrangian, we get

$$\begin{aligned}
\mathcal{L}(w, b, \xi, \alpha) &= \frac{1}{2} w^T w + \frac{1}{2C} \alpha^T \alpha - w^T w + \mathbf{1}_n^T \alpha - \frac{1}{C} \alpha^T \alpha \\
&= -\frac{1}{2} w^T w - \frac{1}{2C} \alpha^T \alpha + \mathbf{1}_n^T \alpha \\
&= -\frac{1}{2} \alpha^T z z^T \alpha - \frac{1}{2C} \alpha^T \alpha + \mathbf{1}_n^T \alpha \\
&= -\frac{1}{2} \alpha^T \left(z z^T + \frac{1}{C} I \right) \alpha + \mathbf{1}_n^T \alpha
\end{aligned}$$

Hence we obtain the dual problem:

$$\begin{aligned}
&\text{minimize } \ \mathcal{L}(\alpha) = \frac{1}{2} \alpha^T \left(z z^T + \frac{1}{C} I \right) \alpha - \mathbf{1}_n^T \alpha \\
&\text{subject to } \ \alpha^T y = 0 \ \text{ and } \ \alpha_i \geq 0, \ 1 \leq i \leq n
\end{aligned}$$

Implementing this, we get

- The optimal dual solution α, which is too large so again I refer you to the code and list the first and last two components:

$$\alpha = (7.5917 \cdot 10^{-14}, 0.11166 \cdot 10^{-1}, \ldots, 4.2408 \cdot 10^{-14}, 1.3454)$$

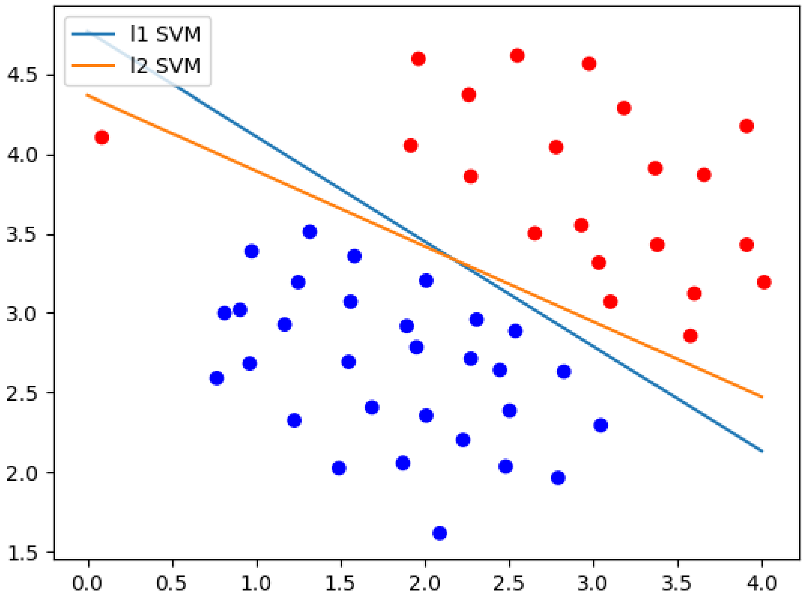
- Recovering w and b from α, we get

$$w = \begin{pmatrix} 0.73873 \\ 1.56042 \end{pmatrix} \quad \text{and} \quad b = -6.81400$$

- The optimal support vectors are

$$X_6 = \begin{pmatrix} 3.3814 \\ 3.4291 \end{pmatrix},\ X_{22} = \begin{pmatrix} 1.9527 \\ 2.7843 \end{pmatrix},\ X_{26} = \begin{pmatrix} 3.0473 \\ 2.2931 \end{pmatrix}$$

(h) The plot is as follows:



We see that the ℓ_1 SVM punishes misclassification less, hence is less affected by the sole red point on its own in the upper left, and instead opts for better separating the two clusters instead (not caring as much about the misclassification). In comparison, $l2$ SVM punishes misclassification more, hence we see the orange decision line is more angled to try and include the outlier point.

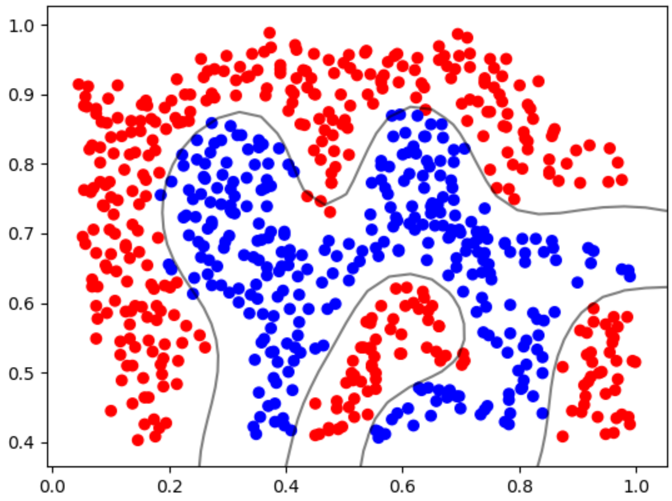
(i) One can look at a scatterplot and determine outliers/patterns, and see whether you want to punish misclassifying more (increasing C value) or the opposite (it is really a case-by-case basis depending on the distribution of the points). Since you could also see this as a hyperparameter, you could use cross-validation as well.

2 Kernel Support Vector Machine and Application

(a) Completed in the code.

(b)
- The number of supopoort vectors is $(80, 82)$ meaning 80 for one group and 82 for the other.
- The prediction error rate for X_{test} is 0.00578.

- The approximate decision boundary plot is



(c) Completed in the code

(d)
```
Degree  1
Number of support vectors:  [294 295]
The training set error rate is:  0.4217391304347826
The test set error rate is:  0.4335260115606936
Degree  2
Number of support vectors:  [256 255]
The training set error rate is:  0.2927536231884058
The test set error rate is:  0.2774566473988439
Degree  3
Number of support vectors:  [250 252]
The training set error rate is:  0.2855072463768116
The test set error rate is:  0.2832369942196532
Degree  4
Number of support vectors:  [228 229]
The training set error rate is:  0.26956521739130435
The test set error rate is:  0.2890173410404624
Degree  5
Number of support vectors:  [224 224]
The training set error rate is:  0.26956521739130435
The test set error rate is:  0.2890173410404624
```

(e) Based on the 5 models trained above, the predictive error decreases as the degree increases. This is because the higher degree polynomial allows more 'curves' to fit the curves in the data, leading to a more robust decision boundary. However, this may lead to overfitting the data. The number of support vectors also decreases as the degree of the polynomial kernel increases, as the higher order terms gives the boundary more flexibility, thus the likelihood individual vectors having an influence on the shape of the curve is low.