CSE 555: Pattern Recognition: Project Report- Support Vector Machines

Manisha Biswas
mbiswas2@buffalo.edu

Problem Statement 1: Implement support vector classifiers on MNIST dataset with dot-product kernel and 1-norm soft margin, and report accuracy. Also, perform cross-validation for hyper-parameter tuning.

Given Datasets:

Firstly, we imported the MNIST datasets from Yann LeCun's webpage. There are 4 datasets

1. Training Data Sets: Two training datasets -

a. A 600000 x 28 x 28 sized dataset containing 60,000 sample images of handwritten MNIST digits from categories 0,1,2,...,9.
b. A 60,000 x 1 sized dataset containing labels of the 60,000 image samples in the training set, representing the class each image belongs to, from 0,1,...,9.

2. Testing Data Sets: Two testing datasets -

a. A 10000 x 28 x 28 sized dataset containing 10,000 sample images of the handwritten MNIST digits from categories 0,1,...,9 against which the classifier is to be tested.
b. A 10000 x 1 sized dataset containing labels of the 10,000 image samples in a testing set representing the class, each image belongs to, from 0,1,...,9.

Next, we read these datasets and define our train-set input and output variables, and also the test-set input and output variables for developing the model.
The classification model is developed using the SVM library of Scikit Learn and initial accuracy achieved with following parameters

C = 0.01
gamma = 0.1
kernel = linear

was 88.82%.

```
############ Part 2 - Developing the model classifier #################


TIME  =  176.91726851463318


Accuracy =  88.82
############ Part 2 Completed ################
```

Figure 1: Output of SVM classifiers on MNIST dataset with dot-product kernel and 1-norm soft margin

Next, we perform 5-folds cross-validation and tune the hyper-parameters using another library of Scikit Learn - GridSearchCV to tune the hyper-parameters.

The following set of hyper-parameters were used for cross-validation:

C = 0.001, 0.1, 0.5, 10, 100
gamma = 0.01, 0.1, 1, 5, 10

The kernel was kept as **poly**.

After 5-folds cross-validation and hyper-parameter tuning, the best parameters were **C = 0.001, kernel = poly and gamma = 10 with Accuracy = 97.83%.**

```
94 ################ Part 3 - Perform cross-validation and hyper-parameter tuning ###############
95 from sklearn.preprocessing import StandardScaler
96 from sklearn.svm import SVC
97 from sklearn.pipeline import Pipeline
98 from sklearn.model_selection import GridSearchCV
99
100 t=time.time()
101 print("############ Part 3 - Performing cross-validation and hyper-parameter tuning ###############")
102 steps = [('scaler', StandardScaler()), ('SVM', SVC(kernel='poly'))]
103 pipeline = Pipeline(steps)
104
105 C_params = [0.001, 0.1, 0.5, 10, 100]
106 gamma_params = [0.01, 0.1, 1, 5, 10]
107 parameters = {'SVM__C':C_params, 'SVM__gamma':gamma_params}
108
109 classifier = GridSearchCV(pipeline, param_grid=parameters, cv=5)
110
111 classifier.fit(x_train, y_train)
112 y_pred = classifier.predict(x_test)
113
114 print("\n\nTime taken for hyper-parameter tuning : ", time.time() - t)
115 print ("\n\nBest parameters found by GridSearchCV: ", classifier.best_params_)
116 print("\n\nScore  = %3.5f", classifier.score(x_test, y_test))
117
118
119 accuracy = findAccuracy(y_pred, y_test)
120
121 print("\n\nAccuracy calculated by self-defined function = ", accuracy)
122 print("############ Part 3 - Completed ###############")
```

Figure 2: Code for performing 5-folds cross-validation and hyper-parameter tuning using SkLearn.

```
############ Part 3 - Performing cross-validation and hyper-parameter tuning ###############


Time taken for hyper-parameter tuning :  27978.637122631073


Best parameters found by GridSearchCV:  {'SVM__C': 0.001, 'SVM__gamma': 0.1}


Score  = %3.5f 0.9783


Accuracy calculated by self-defined function =  97.83
############ Part 3 - Completed ###############
```

Figure 3: Output after performing 5-folds cross-validation and hyper-parameter tuning using SkLearn.

<u>Problem Statement 2</u>: Identify the Lagrange dual problem of the following primal problem. Point out what is the "margin" in both the primal formulation and the dual formulation, what are the benefits of maximizing the margin. Characterize the support vectors. Point out the benefit of solving the dual problem instead of the primal problem.

1. Primal Problem for SVM

Given features

$$(x_1, y_1), \ldots, (x_N, y_N), \text{ where } y_1, ..., y_N \in [-1, 1]$$

Thus, the classifications are based on a **decision boundary** represented by the hyperplane *w* For binary classification, *w* points towards the positive class.

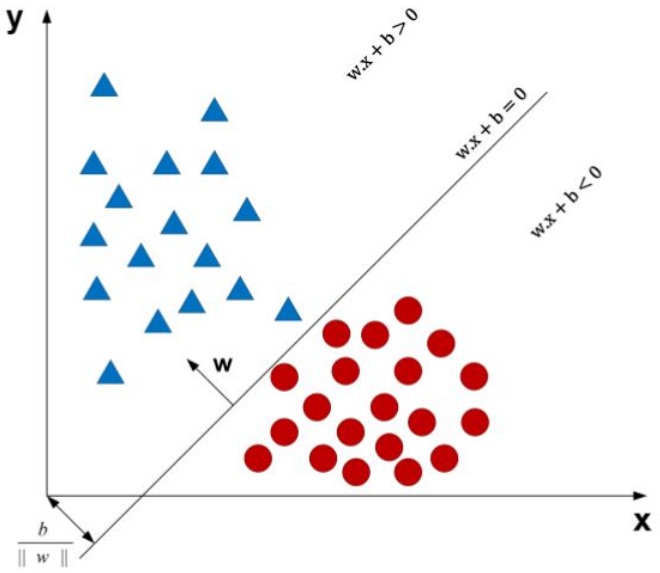

Figure 4: SVM for linear classification

$$Margin = \frac{2}{||w||}$$

∴ The Decision Rule for y can be given as -

$$y = sign(w^T x + b), \; w \text{ being the weights and } b \text{ is the bias.}$$
$$\text{where, } w^T x + b > 0 \Rightarrow y = +1$$
$$w^T x + b < 0 \Rightarrow y = -1$$

We are supposed to minimize -

$$w^T.w + C \sum_{i=1}^{N} \xi_i, \qquad \text{..... (1)}$$

$$\text{subject to}$$

$$y_i(w^T x_i) \geq 1 - \xi_i$$

$$x_i \geq 0 \text{ for } i = 1, ..., N$$

First, let us now define the expression for the primal problem.

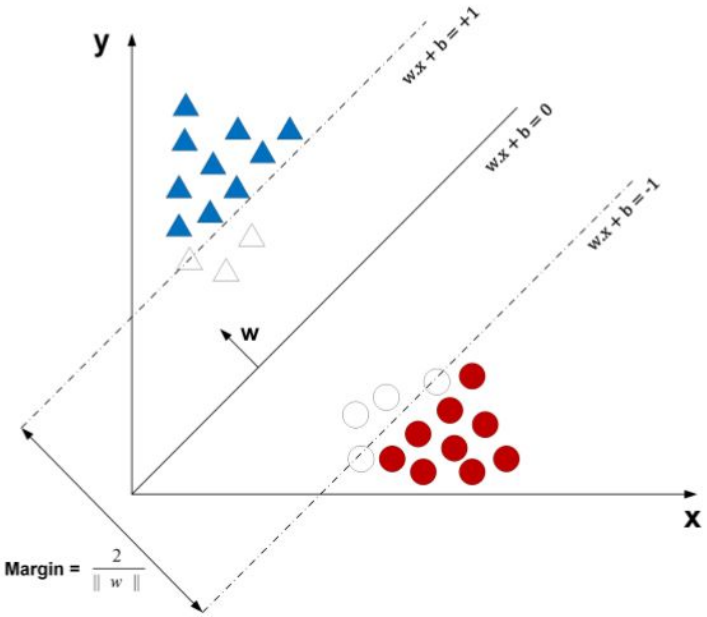

Figure 5: SVM for linear classification with margins

SVM problem with hard constraints is given as -

$$\min_{w} \left[\frac{1}{2} w^T w\right]$$

such that

$$y_i(w^T x_i) \geq 1, i = 1, ..., N$$

However, for handling the case of non-linearly separable classes, we relax the constraint by introducing a **slack variable**, ξi.

$$\therefore \text{New constraint: } y_i(w^T x_i + b) \geq 1 - \xi_i, \forall\ i = 1, ..., N \text{ and } \xi_i \geq 0$$

$$\text{For misclassification, } \xi_i > 1$$

1.1 Significance of C

In equation (1), C controls the impact of margin and the margin error.
It is okay to have some misclassified training examples, i.e., some ξi will be non-zero.
Thus, we minimize the number of such examples, i.e

$$minimize \sum_{i=1}^{N} \xi_i$$

Hence, the **optimization problem** OR the primal problem for non-separable case becomes -

$$\underset{w,b}{minimize} f(w,b) = ||w||^2 + C\sum_{i=1}^{N}\xi_i$$

..... (2)

subject to $y_i(w^T x_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0\ \forall i = 1, ..., N$
where the weights, w are given as -

$$w = \sum_{i=1}^{N}\alpha_i y_i x_i$$

2. Deriving the Dual Problem

The Dual problem basically maps the data to a higher dimensional space to tackle the problem of non-linearly separable data points.
Thus, equation (2) can also be expressed as -

$$\min_{w,\xi}\ \max_{\alpha\geq 0,\beta\geq 0}[\frac{1}{2}||w^2|| + C\sum_i \xi_i - \sum_i \alpha_i(y_i w^T x_i - 1 + \xi_i) - \sum\sum_i \beta_i\xi_i$$

..... (3)

Certain conditions like Slater's condition allow exchanging *min, max* without changing the optimal solution.

∴ (3) becomes,

$$\max_{\alpha\geq 0,\beta\geq 0}\ \min_{w,\xi}\left[\frac{1}{2}||w^2|| + C\sum_i \xi_i - \sum_i \alpha_i(y_i w^T x_i - 1 + \xi_i) - \sum\sum_i \beta_i\xi_i\right]$$

..... (4)

$$= \max_{\alpha\geq 0,\beta\geq 0}\ \min_{w,\xi}\left[\frac{1}{2}||w^2|| - \sum_i \alpha_i y_i w^T x_i + \sum_i \xi_i(C - \alpha_i - \beta_i) + \sum_i \alpha_i\right]$$

..... (5)

For any given $\alpha, \beta,$ minimizer of w will satisfy-

$$\frac{\partial L}{\partial w} = w - \sum_i \alpha_i y_i x_i = 0$$

$$\Rightarrow w^* = \sum_i y_i\alpha_i x_i$$

..... (6)

Also, since C = $\alpha i + \beta i,$ ξi will never make the objective function equal to $-\infty$.
Substituting equation (6) back in equation (5), we obtain:

$$\max_{\alpha \geq 0, \beta \geq 0, C = \alpha + \beta} \left[-\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i \right]$$

..... (7)

Thus, the **dual problem** is given as follows -

$$\max_{C \geq \alpha \geq 0} \left[-\frac{1}{2} \alpha^T Q x + e^T \alpha \right] := D(\alpha)$$

..... (8)

where, Q is and NxN matrix with $Q_{i,j} = y_i y_j x_i^T x_j$.

Primal Minimum = **Dual Maximum** (under Slater's condition), i.e

If p^* is the primal solution and d^* is the dual solution, then

$$p^* = \sum_i y_i d_i^* x_i$$

Thus, we can solve the dual problem instead of the primal problem.

3. Finding the Support Vectors

In **linear** case,

if x_i does not lie on margin, then

$$y_i(w^T x_i + b) > 1$$

$$\Rightarrow \alpha_i = 0$$

$\therefore \alpha_i \neq 0$ only for x_i on margin. These are the **support vectors** as shown in the figure.

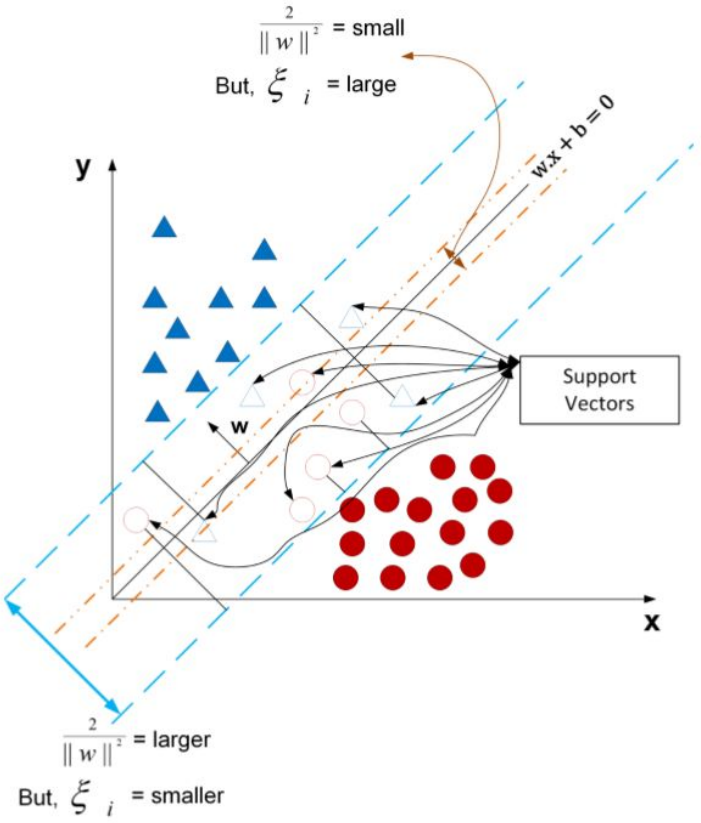

Figure 6: SVM for linear classification with two margins with support vectors.

However, in **non-linear** cases, support vectors are also those that lie within the margin on the *incorrect side of the separating hyperplane.*

Thus, *number of support vectors(non-linear) > number of support vectors(linear)*

When the input is mapped to a higher dimensional space, the **"Kernel Trick"** helps in easier optimization of the margin. Basically, we do the following -

$$x \to \phi(x), \text{ where } \phi : R^d \to R^D$$

such that, the data is linearly separable in the mapped feature space.

Hence, **the dual problem for SVM is easier to solve than the primal problem.**

4. Margins

Margin in primal problem:

$$\frac{2}{||w||}$$

Margin in dual problem:

$$\frac{2}{||w||}$$

$$\text{where, } w = \alpha_i y_i$$

5. Questions

Q1. Point out what is the margin in the dual and primal problem.

Ans: The margins for both the solutions are mentioned in Section 4. Margins as above.

Q2. What are the benefits of maximizing the margin?

Ans: Let's assume the data is linearly separable and we are fitting the SVM to the data. Our primary aim is to alter the direction of the hyperplane to maximize the geometric distance of the closest training point to the plane. However, directly optimizing the geometric distances is computationally infeasible, so we instead maximize the functional margin, which is inversely proportional to $||w||^2$. Hence, we indirectly minimize the error. Additionally, maximizing the margin reduces overfitting error and increases generalization. So, chances of getting high testing errors and low training errors are very less.

Q3. Characterize the support vectors.

Ans: The support vectors have been shown in Figure 4.
In Linear classification, using SVM, if x_n is **not** on margin, we get $\alpha_n = 0$. Thus, $\alpha_n \neq 0$ only for those x_n that lie on the margin. Such x_n are called the **support vectors** in that case.

However, for non-linearly separable classification using SVM, there are some x_n that get misclassified and lie away from the margin on the incorrect side. For these, $\alpha_n = 0$, but they are still considered as the support vectors. Therefore, **in non-linear classification, the number of support vectors is greater than that in linear classification using SVM.**

Q4. Point out the benefit of solving the dual problem instead of the primal problem.

Ans: Dual problem solution helps in the following ways over that of the primal problem -

a. We can easily use Kernel Methods in dual problems to classify data that is non-linearly separable in the original feature space.

b. When the number of data points is lower than the number of dimensions, it's easier to optimize in the dual than in the primal irrespective of how many dimensions are there. Dual representation will have only as many parameters as the number of data points.

c. **Regularizing the sparse support vector** in dual problem is sometimes **more intuitive** than regularizing the vector of regression coefficients. Dual problems always adapt to the amount of available data.
d. It is known that convex problems always converge. The primal problem being a non-convex problem may not always converge, however the dual problem which is a convex problem will always converge.
e. The dual problem helps in unconstrained optimization, especially when we need to find the optimum value at a point in the convex function where the slope is zero. Applying the KKT conditions on the primal problem for the same case leads to the dual problem solution eventually, which can be used in the first place.

<u>Problem Statement 3</u>: Formulate the primal problem and derive the dual problem if there are multiple classes.

Task 3 :-

for a multiclass classification problem a new multiclass classifier called NHCMC non-parallel hyperplane classifier for multi-class classification.

Consider the multiple classification problem with the training set : $T = \{(x_1, y_1) \dots (x_i, y_i)\}$ where $x_i \in R^n, i = 1 \dots k, \dots$ ①

$y_i \in \{1 \dots k\}$ is the corresponding pattern of x_i.

for multiple classification, we seek K non-parallel hyperplanes

$$(w_k \cdot x) + b_k = 0 . b = 1 \dots K \quad \dots\dots\dots\dots ②$$

for convenience, we denote the number of each class of the training set (1) as l_k and the points belonging to k-th class as $A_k \in R^{l_k \times n}$, $k = 1 \dots k$. Besides, we define the matrix B_k.

$$B_k = [A_1^T, \dots, A_{k-1}^T, A_{k+1}^T, \dots A_k^T]^T$$

As all the points except for the points belonging to the kth class.

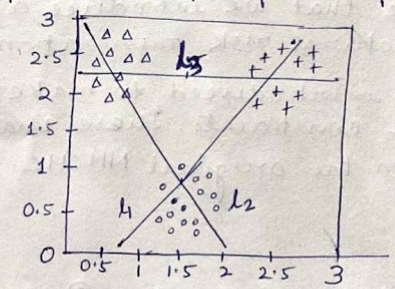

fig 1 :- An example of linear NHCMC learning

<u>Linear case</u> :-

<u>The Primal Problem</u> :-

we seek to construct K-non parallel hyperplanes eq(2) by solving the following convex quadratic programming problems [QPPS]

$$\min_{w_k, b_k, \eta_k, \epsilon_k} \quad \frac{1}{2} c_1 \|w_k\|^2 + \frac{1}{2} \eta_k^T \eta_k + c_2 c_k^T \epsilon_k \quad \dots\dots ③$$

$$s.t \quad B_k \cdot w_k + c_{k1} \cdot b_k \cdot e \eta_k$$

$$(A_k \cdot w_k + c_{k2} \cdot b_k) + \epsilon_k \geq c_{k2}$$

$$\epsilon_k \geq 0$$

where $\eta_K \in R^{(l-l_K)}$ is a variable and ϵ_K is a slack variable, $c_{K1} \in R^{(l-l_K)}$ and $c_{K2} \in R^{l_K}$ are the vectors of. $C_1 \geq 0$ and $C_2 \geq 0$ are penality parameters.

In order to illustrate the prinal problem of NHCMC, we generated an artificial two dimensional 3-class dataset. The geometric interpretion of the above problem is i^{th} α belongs to R^2 as shown in figure above. where α minimizes the sum of the square distance from the hyperplane of k-1 classes. i.e all classes accept for those of the k^{th} class, and the points of the k^{th} class are far from the i^{th} hyperplane. Take the +class in figure for an example. we hope the hyperplane of the O class l_1 is far from the + points and closest to the O, and Δ points. In order to minimize the classification the points of the k^{th} class are at a distance from the hyperplane and we minimize the sum of error variables with soft margin SVM.

The differences between multiple birth support vector machines (MBSVM) and NHCMC are that we introduce a regularization form to implement structural risk minization (SRM). principal and a variable is introduced to make a set of objective functions to be constraints. These changes have many positive effects on the original NHCMC.

The Dual Problem :-

In order to get the solution of problem ③ we need to derive its dual problem. The lagrangian of the problem ③ is given by :

$$L(w_K, b_K, \eta_K, \epsilon_K, \alpha, \beta, \lambda) = \frac{1}{2} c_1 \|w_K\|^2 + \frac{1}{2} \eta_K^T \eta_K + c_2 e_{k_2}^T \epsilon_K + \lambda^T (B_K w_K + e_{k_2} b_K - \eta_K) - \alpha^T (A_K w_K + e_{k_2} b_K + \epsilon_K - e_{k_2}) - \beta^T \epsilon_K \quad \text{---- ④}$$

where $\alpha = (\alpha_1, \ldots, \alpha_{l_K})^T$,

$\beta = (\beta_1, \ldots, \beta_{l_K})^T$, $\lambda = (\lambda_1, \ldots, \lambda_{l-l_K})^T$

are the lagrange multiplier vectors. The KKT conditions for w_K, b_K, η_K, ϵ_K and α, β, λ are given by

$$\nabla_{w_k} L = C_1 w_k + B_k^T \cdot \lambda - A b^T \alpha = 0 \quad \text{---} \quad (5)$$

$$\nabla_{b_k} L = e_{k_1}^T \lambda - e_{k_2}^T \alpha = 0 \quad \text{---} \quad (6)$$

$$\nabla_{\eta_k} L = \eta_k - \lambda = 0 \quad \text{---} \quad (7)$$

$$\nabla_{\epsilon_k} L = C_2 e_{k_2} - \alpha - \beta = 0 \quad \text{---} \quad (8)$$

$$B_k w_k + e_{k_1} b_k = \eta_{k_1} \quad \text{---} \quad (9)$$

$$(\Delta_k w_k + e_{k_2} b_k) + \epsilon_k \geq e_{k_2}, \epsilon_k \geq 0 \quad \text{---} \quad (10)$$

$$\alpha^T ((\Delta_k w_k + e_{k_2} b_k) + \epsilon_k) = 0, \; B^T E_k = 0, \; \alpha \geq 0, \beta \geq 0 \quad \text{---} \quad (11)$$

Since, $\beta \geq 0$ from eq (8) we have $0 \leq \alpha \leq C_2 e_{k_2}$ ---- (12)

and from eq (5) we have $w_k = -\frac{1}{C_1} \left[B_k^T \lambda - A_k^T \alpha \right]$ --- (13)

Then put in eq (13) and eq (7) into Lagrangian and using eq (11) we obtain the dual problem of eq (4).