SMAI PROJECT REPORT

```
Team No          : 44
Team Name        : Random_team_1
Project ID       : 26
Project Title    : Implement MULTICLASS SVM WITH DIFFERENT KERNELS FROM SCRATCH
Team Members     :
                   1.2018801010 - Karnati Venkata Kartheek
                   2.2018900061 - Shashikant Ghangare
GitHub Link:
https://github.com/shashikant-ghangare/SMAI-Spring2019-Project
```

Introduction

Multiclass SVM is usually implemented using One vs One or One vs Rest fashion, here in this work an alternative approach is used, optimization formulation is such that at a go it decides the scores for all the classes and assigns highest score class to the test sample.

Problem statement

In this project, we implement simultaneous multiclass SVM with different kernels using CVXOPT and compare their performance (space, time requirements and accuracy) with SCIKIT Learn implementations of One vs One and One vs Rest classifiers.

Overview

Math behind the SVM and multiclass SVM

In Soft margin binary SVM

The optimization function is

$$\min L = \tfrac{1}{2} w'w + C \sum_i^n \xi_i$$

$$\text{such that } y_i f(\mathbf{x}_i) \geq 1 - \xi_i, \text{ for all } i(1 \text{ to } n)$$

$$\xi_i \geq 0$$

Here C is the hyperparameter and ξ_I is the slack variable associated with i^{th} point.

In the Soft Margin, Multiclass SVM

The optimization function is

$$\min L = ½ \sum_t w_t' w_t + C \sum_i \sum_{m(\neq y_i)} \xi_i^m$$

such that for ever point i(1 to n),

$$W_{yi}' \mathbf{x}_i - W_m' \mathbf{x}_i \geq 1 - \xi_i^m, \text{ for } m_{(\neq y_i)} = 1,2 ..k$$

Here C is the hyperparameter and ξ_i^m is the slack variable associated with ith point and mth class.

n is the number of points in the given data and k is the number of classes present in the data.

Here $\mathbf{x}_i$ is a **d+1** dimensional vector, Here each point x_i is appended with 1 as new attribute to account for intercept. and w_k is a **d+1** dimensional vector. the vectors W_k are concatenated to become a single vector

$$W =[w_1\, w_2\, w_3 \ldots\ldots w_k]$$

Thus, W is of dimension **(d+1) *k**

So, the vectorized optimization function with respect to concatenated W is

$$L = ½ W'W + C 1' \xi$$

The constraints $W_{yi}' \mathbf{x}_i - W_m' \mathbf{x}_i \geq 1 - \xi_i^m$, for $m_{(\neq y_i)} = 1,2 ..k$ are converted into the form

$$A*W \geq 1- \xi$$

$$\text{Where} \quad A = \begin{array}{c} \text{Columns} \quad 0 \quad 1 \;\; 2 \ldots\ldots \quad Y_i \ldots\ldots \quad \mathbf{k} \\ \left[\begin{array}{c} -\mathbf{x}_i \;\; 0 \;\; 0 \;\ldots.\;\; 0 \;\; \mathbf{x}_i \;\; 0 \;\ldots.\;\; 0 \\ 0 \;\; -\mathbf{x}_i \;\; 0 \;\ldots.\;\; 0 \;\; \mathbf{x}_i \;\; 0 \;\ldots.\;\; 0 \\ 0 \;\;\; 0 \;\; -\mathbf{x}_i \;\;\ldots.\;\; \mathbf{x}_i \;\; 0 \;\ldots.\;\; 0 \\ \\ \Big\downarrow \\ \\ \end{array}\right] \end{array} \left[\begin{array}{c} w_1 \\ w_2 \\ w_3 \\ \\ w_k \end{array}\right]$$

Here the Dimensions of A = [n(k-1),k(d+1)]

Because for each point there a k-1 constraints and thus n(k-1)

In each row, there are k vectors of dimensions (d+1) thus dimensions k(d+1).

We can form the Legrangian

$$L = \tfrac{1}{2} W'W + C\, 1'\xi - \alpha' (A*W-(1-\xi)) - \beta' \xi$$

$$L = \tfrac{1}{2} W'W + C\, 1'\xi - \alpha' (A*W) + \alpha' 1 - \alpha'\xi - \beta' \xi$$

Differentiating w.r.to W = 0

$$\tfrac{1}{2}\, 2W' = \alpha' A$$

$$W = A'\alpha \qquad \text{------} \;\; \underline{1}$$

Differentiating w.r.to ξ = 0

$$C * 1' - \alpha' - \beta' = 0$$

$$C * 1' = \alpha' + \beta' \qquad \text{------} \;\; \underline{2}$$

$$\text{i.e } 0 \le \alpha' \le C$$

Substituting <u>1</u> , <u>2</u> back into legrangian, we get

$$L = \tfrac{1}{2} W'W + C\, 1'\,\xi - \alpha'\,(A*W) + \alpha' 1 - \alpha'\xi - \beta'\,\xi$$

$$L = \tfrac{1}{2}\, \alpha'\, A'\, A\, \alpha + (\alpha' + \beta')\,\xi - \alpha'\, A\, A'\, \alpha + \alpha' 1 - \alpha'\xi - \beta'\,\xi$$

$$\text{Maximize } L = \alpha' 1 - \tfrac{1}{2}\, \alpha'\, A'\, A\, \alpha$$

Or

$$\text{Minimize } L = \tfrac{1}{2}\, \alpha'\, A'\, A\, \alpha - 1'\alpha$$

Such that $0 \leq \alpha' \leq C$ or $\alpha' \leq C$ and $-\alpha' \leq 0$

This is the dual form of the optimization problem

Optimization format accepted by CVXOPT is

$$\text{Minimize } \tfrac{1}{2}\, \alpha' P\, \alpha + q'\, \alpha$$

$$\text{Subject to } G\, \alpha \leq h$$

So in order to solve using CVXOPT

Here $P = A'\, A$ $q = [-1\ -1\ -1\ \ldots.]$

$$\text{Where } G\,\alpha = \begin{bmatrix} \mathbf{-1} & 0 & 0 & \ldots. \\ 0 & \mathbf{-1} & 0 & \ldots. \\ 0 & 0 & \mathbf{-1} & \ldots. \\ & & & \Downarrow \\ 1 & 0 & 0 & \ldots \\ 0 & 1 & 0 & \ldots \\ 0 & 0 & 1 & \ldots \\ & & & \Downarrow \end{bmatrix} \begin{bmatrix} \alpha_i^1 \\ \alpha_i^2 \\ \alpha_i^3 \\ \\ \alpha_n^k \end{bmatrix} \qquad h = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \Downarrow \\ C \\ C \\ C \\ \Downarrow \end{bmatrix}$$

For the linear kernel just the product A' A is enough but for the polynomial kernel and rbf kernel every dot product $x_i'x_j$ involved in A'A calculation is replaced respectively by

$$k(\mathbf{x_i}, \mathbf{x_j}) = (\mathbf{x_i} \cdot \mathbf{x_j} + 1)^d \qquad k(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$$

To predict the label for point $\mathbf{x}_t$ we calculate

argmax $([W_1{}' \mathbf{x}_t, W_2{}' \mathbf{x}_t, \ldots\ldots.., W_k{}' \mathbf{x}_t])$

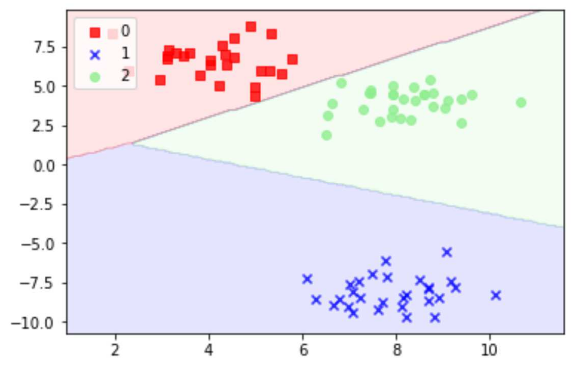
$$[W_1{}' \mathbf{x}_t, W_2{}' \mathbf{x}_t, \ldots\ldots.., W_k{}' \mathbf{x}_t] = W'{*}\text{ValXtMatrix}$$
$$= \alpha' A{*}\text{ValXtMatrix}$$
$$= \alpha'\text{kernel}(A{*}\text{ValXtMatrix})$$

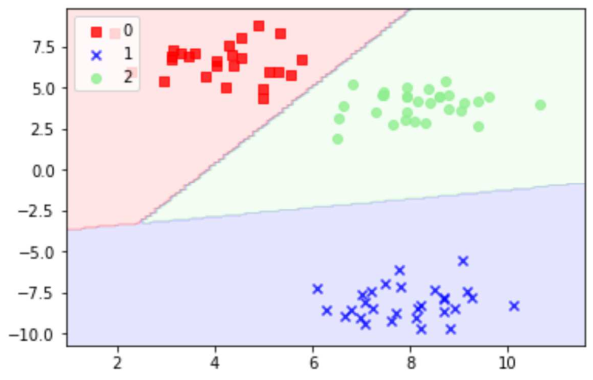
$$\text{Where} \quad A{*}\ \text{ValXtMatrix} = \begin{bmatrix} -\mathbf{x}_i & 0 & 0 & \ldots. & 0 & \mathbf{x}_i & 0 & \ldots. & 0 \\ 0 & -\mathbf{x}_i & 0 & \ldots. & 0 & \mathbf{x}_i & 0 & \ldots. & 0 \\ 0 & 0 & -\mathbf{x}_i & \ldots. & & \mathbf{x}_i & 0 & \ldots. & 0 \\ & & & & \Big\downarrow & & & & \end{bmatrix} \begin{bmatrix} \mathbf{x}_t & 0 & 0 & \ldots & \\ 0 & \mathbf{x}_t & 0 & \ldots & \\ 0 & 0 & \mathbf{x}_t & \ldots & \end{bmatrix}$$

Where $\mathbf{x}_i$'s in the first matrix are the row vectors and $\mathbf{x}_t$'s in the second matrix are the column vectors

Results and Discussion

Visualization on some datasets, different kernels





SMC vs OVO vs OVR Comparison

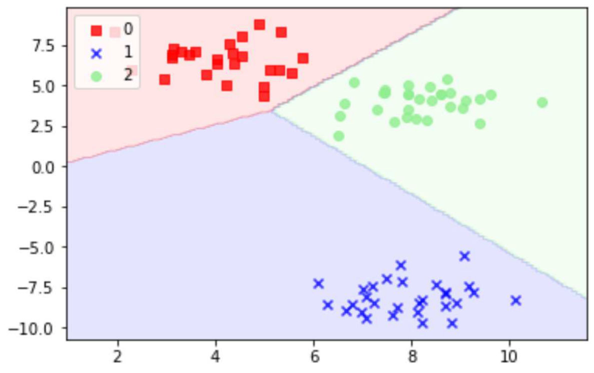
Top Left SMCSVM

Top Right OVO

Bottom Right OVR

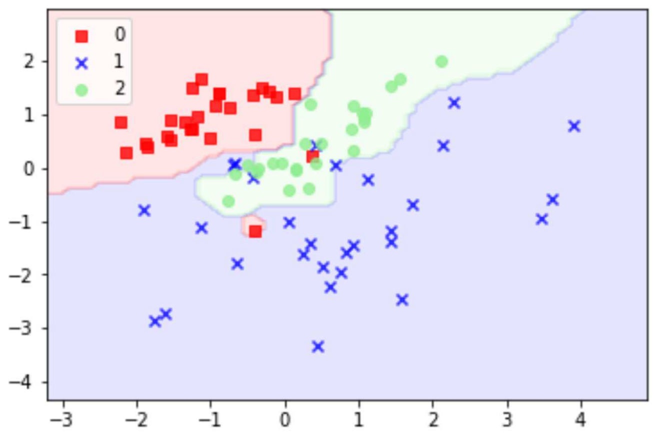
make blobs(n_features=2, centers=3)

kernel = 'linear' C = 100000



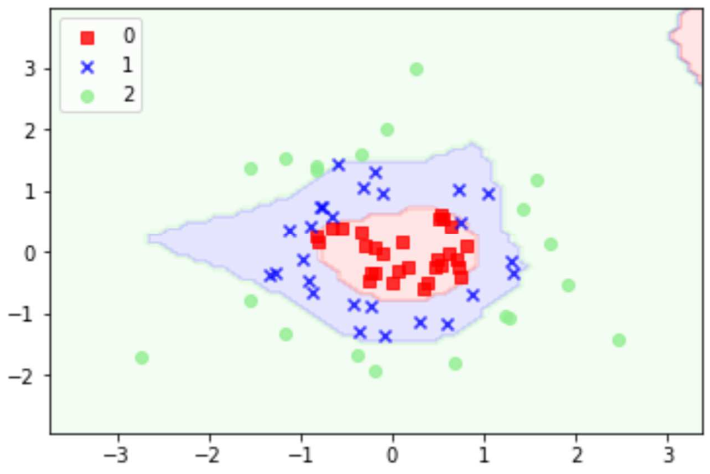
(n_features=2 ,n_clusters_per_class=1, n_classes=3)(overfitted)

kernel = 'rbf' degree = 4 sigma = 0.4 C = 100



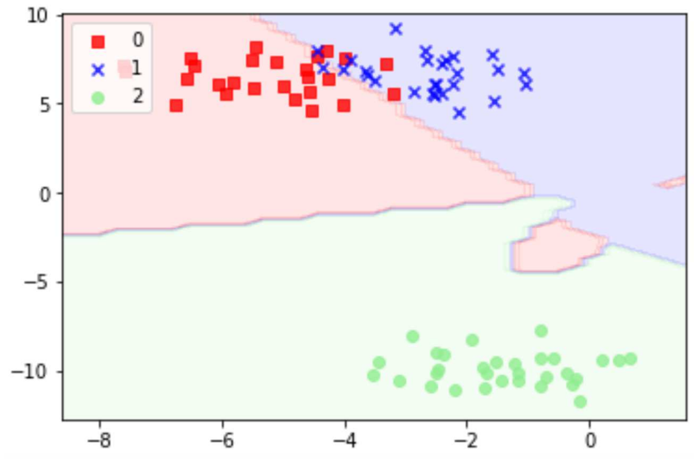
Gaussian quantiles(n_features=2, n_classes=3)(overfitted)

kernel = 'rbf'   degree = 4 sigma = 0.4 C = 100



make blobs (n_features=2, centers=3)

kernel = 'polynomial'   degree = 3 C = 100



From the visualizations, we can see that all the kernels are working and from the comparison of OVO, OVR and MCSVM we can see that in OVO classifier only cares about that particular pair on which it is trained and in OVR classifier considers a single class and rest all other classes as one other class and trains, but in MCSVM we can see that it differs from in view that while deciding on one classifier considers all other classes these can be seen from the boundaries in the figure.

Error rates (Best hyperparameter values set from cross Validation) comparison for 3 classifiers on different datasets

Below are the Avg time Taken by the classifiers(OVO,OVR,SMCSVM) And error obtained for best hyperparameter values obtained from cross validation we can see that SMCSVM perform comparably with others

Dataset = IRIS

Method - SMCSVM

c=1,10,50 kernel='rbf' sigma='0.7' degree=1

mean_error= 0.0608

Avg Time Taken = 6.32 secs

Method – LinearSVC(OVR)

C=1,10 kernel='linear'

mean_error = 0.0605

Avg Time Taken = 0.006 secs

method – SVC(OVO)

C=50 kernel = 'rbf', 'linear', poly sigma = 0.2,0.7,1.2 degree = 2,3,4

mean_error = 0.1078

Avg Time Taken = 1.9 secs

Dataset = Wine

Method – SMCSVM

c=1 kernel='linear' mean_error= 0.208 Avg Time Taken = 0.65 secs

Note: Using a higher sigma value above 10 increases accuracy of RBF kernel

Method – LinearSVC(OVR)

C=1,10,50 kernel='poly', 'linear' degree = 2,3,4

mean_error = 0.2254

Avg Time Taken = 0.38 secs

method - SVC(OVO)

C=10 kernel = 'rbf' sigma= 0.2 mean_error = 0.1682 Avg Time Taken = 3 secs

Comparison w.r.to Training Time taken by 3 classifiers

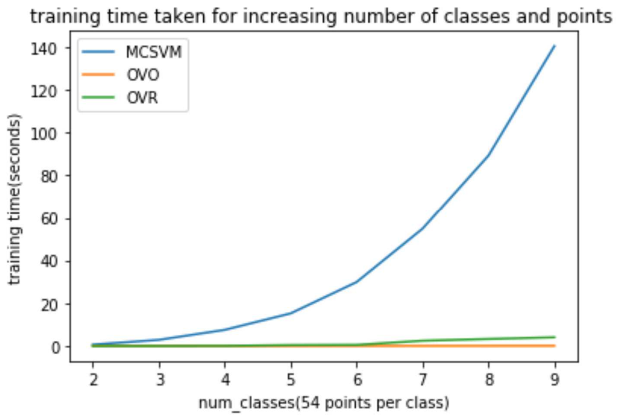
Space Requirements

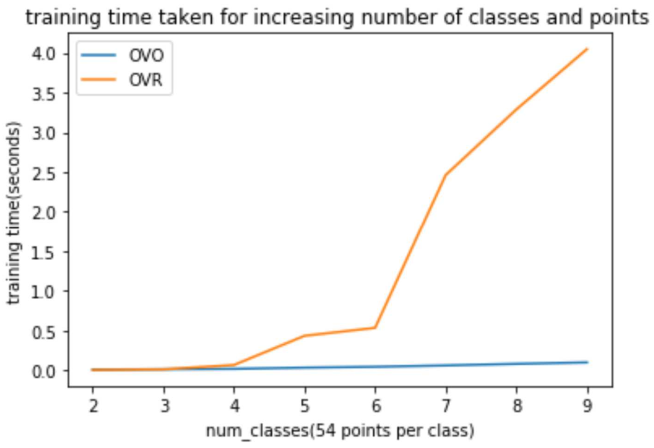
In the OVO(one vs one) and OVR(one vs rest) cases always binary classifiers will be trained (kC2 classifiers in OVO case and k classifiers in OVR case) we can see these trends more clearly in next comparison but in Multiclass classifier the classifier isn't binary and the constraints increase with increase in classes, The A matrix show before is an indicative of this and

dim(A'A) = [n(k-1), n(k-1)]

A'A has to be supplied to the classifier and with increase in k and n the space requirements will increase in multiclass classifier

Comparison of Time taken while increasing number of points and number of classes





Here we can see multiclass SVM training time is increasing as expected and also OVR training time is more than that of OVO because in OVR number of points are increasing as the classes are increases and in OVO the classifiers is always trained on two classes with fixed number of points (54 in this case). Thus we can see that if number of points per class is less then OVO is at advantage

Failure cases

When the number of classes and number of data points increase the matrices involved in optimization are of dimensions [n(k-1),k(d+1)], [n(k-1), n(k-1)] thus quickly scale up with n*k and also takes time to constructed kernelized product of those matrices. When the matrices dimensions are of 12000*12000 CVXOPT giving memory error.

task assignment

The tasks performed by

KARNATI VENKATA KARTHEEK (2018801010)

1.Mathematical Formulation of Simultaneous multiclass SVM
2. Coding of the multiclass SVM classifier (SMCSVM) with kernels
3.Comparision between OVO, OVR and SMCSVM decision boundaries
4.Visualization on some datasets, using different kernels
5.Space requirements
6.Comparison while increasing number of points and number of classes
7.Report and Presentation

The tasks performed by

SHASHIKANT GHANGARE (2018900061)

1.Setting up and maintenance of github repository
2.Coding of cross validation rotinue
3.Cross Validated Error rates comparison on different datasets
4.Avg Time requirements comparison from cross validation
5.Debugging and Making the entire code object oriented.

Some part of visualization code was obtained from internet.
Apparel dataset used in Comparison while increasing number of points and number of classes is the one given for assignment(PCA).