Machine Learning Project 2
Nengyi Fu
2022/2/25

Support Vector Machines

Introduction

A support vector machine (SVM) is a binary classification model whose basic model is a linear classifier with the largest margin defined on the feature space, and the largest margin makes it different from a perceptron; SVM also includes kernel tricks, which make it become an essentially non-linear classifier. The learning strategy of SVM is the interval maximization, which can be formalized as a problem of solving convex quadratic programming, which is also equivalent to the minimization of the regularized hinge loss function. The learning algorithm of SVM is the optimization algorithm for solving convex quadratic programming.

In this project, we implemented a support vector machine (SVM) algorithm, and the optimization method for solving SVM model was sequential Minimal Optimization (SMO). Two implementations regarding SMO were presented, the original one [3] and an improved version of Platt's SMO [4].

Methodology

SVM is to find the line that achieves largest margin, which is the distance from the support vector the decision boundary, we use the equation from slides [6]

$$\rho = \min_{i \in [1,m]} \frac{|\mathbf{w} \cdot \mathbf{x}_i + b|}{\|\mathbf{w}\|} \quad (1)$$

To accommodate outliners, we introduce 'soft marginal' into the algorithm, this is accomplished by including the Lagrange multiplier with the penalty term. The tolerance is a parameter to be determined. The Lagrange multiplier needs to be determined using K-fold cross validation.

The generalized Lagrangian form of the original problem is changed into the corresponding mini-max form, and then for the convenience of solving, the mini-max form can be changed into the minimax form of its dual form, which becomes the original problem the dual problem as in (2).

$$\min_{\mathbf{w},b,\xi} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i$$
$$\text{subject to} \ \ y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \xi_i \ \wedge \ \xi_i \geq 0, i \in [1, m]. \quad (2)$$

We use coordinate ascent to solve this optimization problem, the convergence condition is KKT.

$$\begin{aligned} \alpha_i = 0 \quad &\Rightarrow \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 \\ \alpha_i = C \quad &\Rightarrow \quad y^{(i)}(w^T x^{(i)} + b) \leq 1 \\ 0 < \alpha_i < C \quad &\Rightarrow \quad y^{(i)}(w^T x^{(i)} + b) = 1. \end{aligned} \quad (3)$$

KKT allows the function to be less than zero, that is, outside the support vector The sample points have no effect on the classification results.

SMO

SMO is intrinsically **Coordinate Ascent**, this algorithm achieves the purpose of optimizing the function by updating one dimension in the multivariate function each time, and iterates many times until convergence. Simply put, it is to continuously select a variable for one-dimensional optimization until the function reaches the local optimum point.

Result and Discussion

To determine the best **margin parameter C** (lambda), we applied standard K-fold cross validation, we determine the score use the corrected labelled testing data to divide the total number of testing data.

We choose the K to be the standard value 5, and the lambda was tested from range 1E-6 to 1E6, the train curve for test different lambdas were shown in figure 1 and figure 2. (for artificial data only) The optimal lambda occurs at **0.1** for the blob data, and **1** for Spiral data.

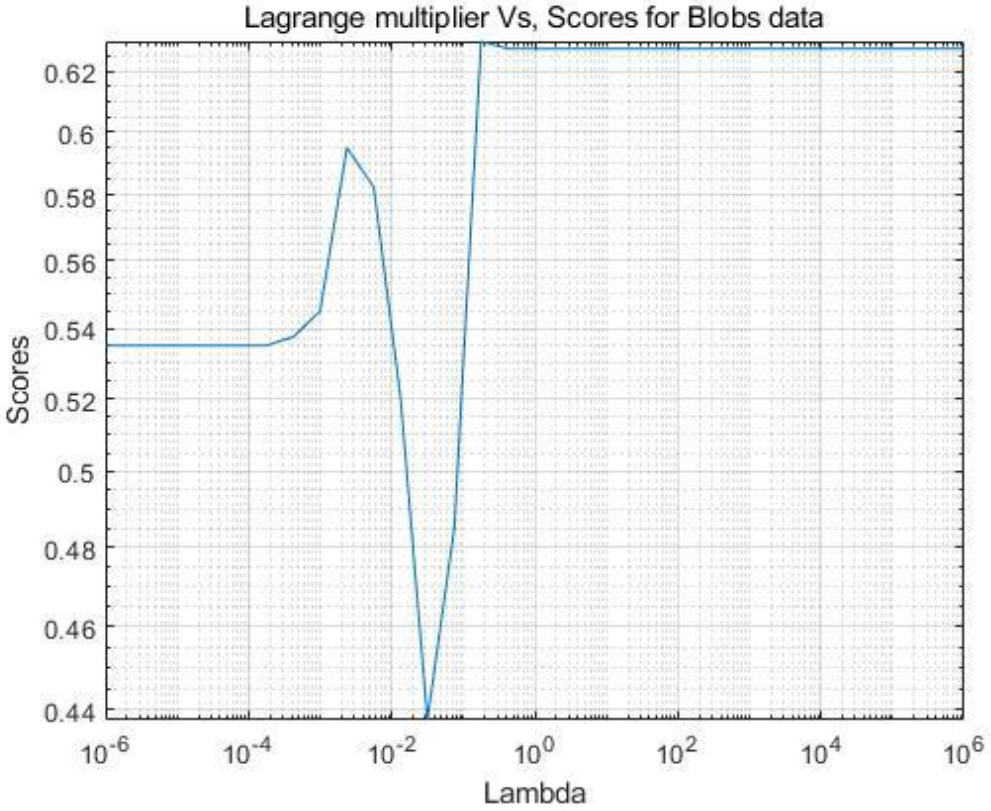

Figure 1. Lagrange multiplier score for Blobs data, 0.1 is the optimal value

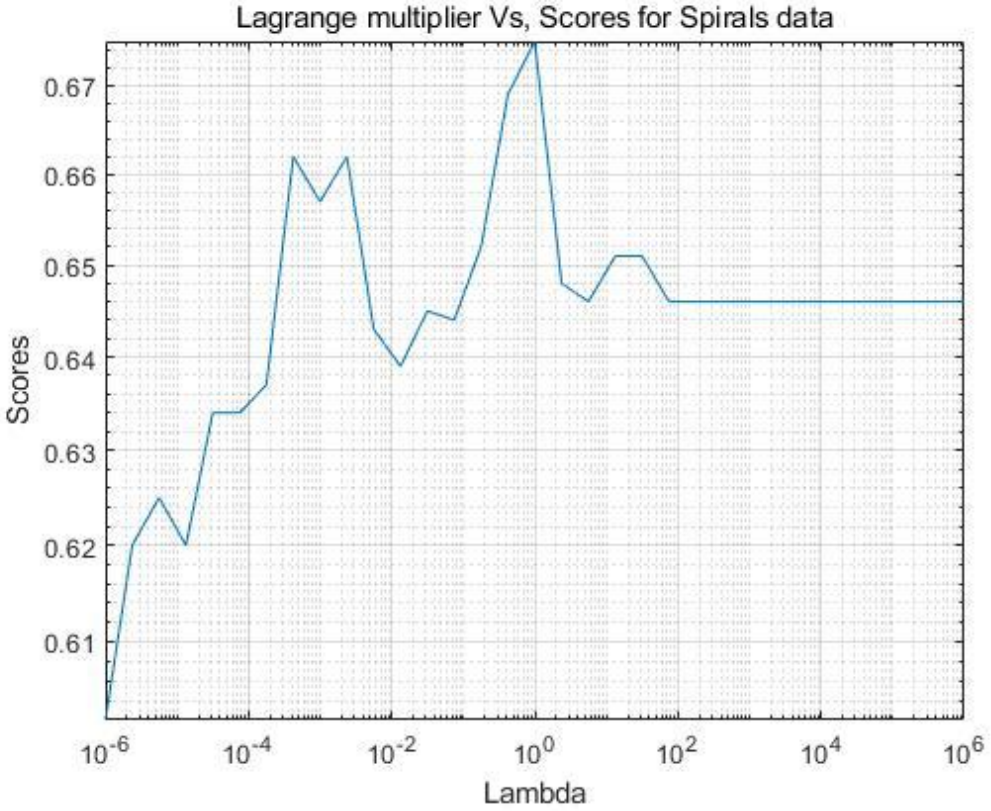

Figure 2. Lagrange multiplier score for spirals data, 1 is the optimal value

After we have determined the optimal lambda, we use it to re-train the data and plot the results in figure 3 and figure 4. We use polynomial degree of 1 and 9 to train the blob and spiral data.

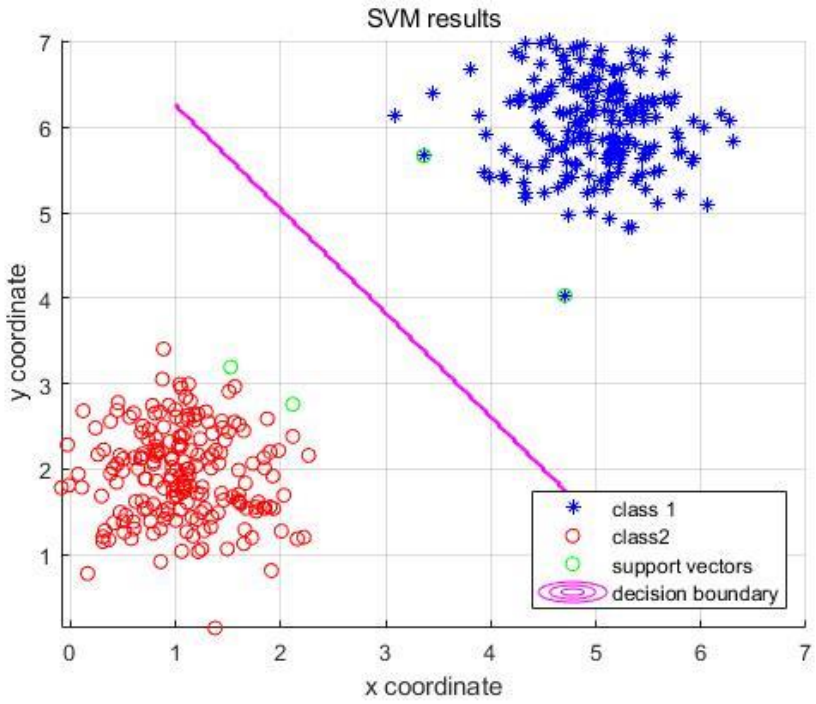

Figure 3. SVM result for Blobs, two class were indicated by blue and red colors, support vectors were labels with green, and the pink boundary is the decision boundary.

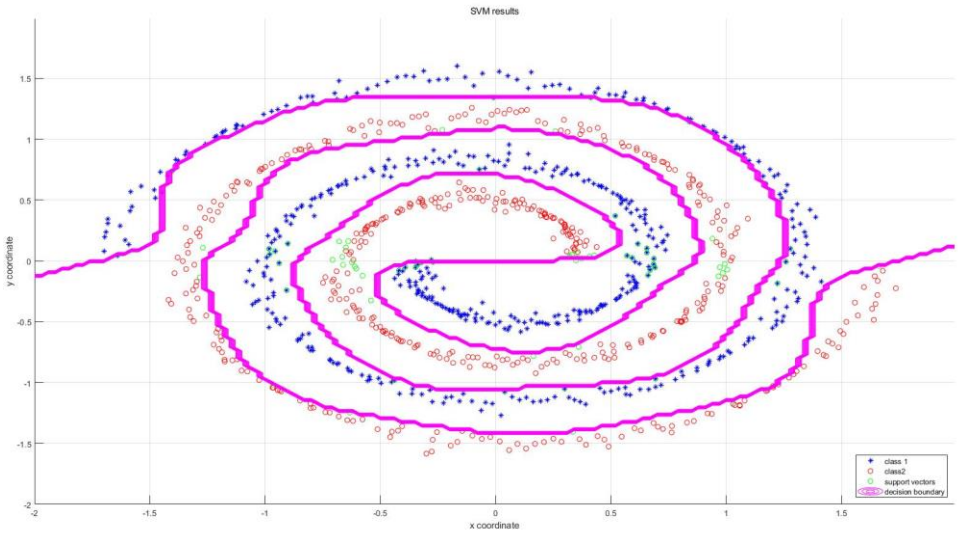

Figure 4. SVM result for Spirals, two class were indicated by blue and red colors, support vectors were labels with green, and the pink boundary is the decision boundary.

We had carried out SVM for 3 other data set, after we plotted each data, after plotted, they are separable, therefore we applied polynomial degree of one for the kernel.

(Setosa, Versicolor) [data 1],
(sunflower, rose) [data 2],
(low income, High income) [data 3]

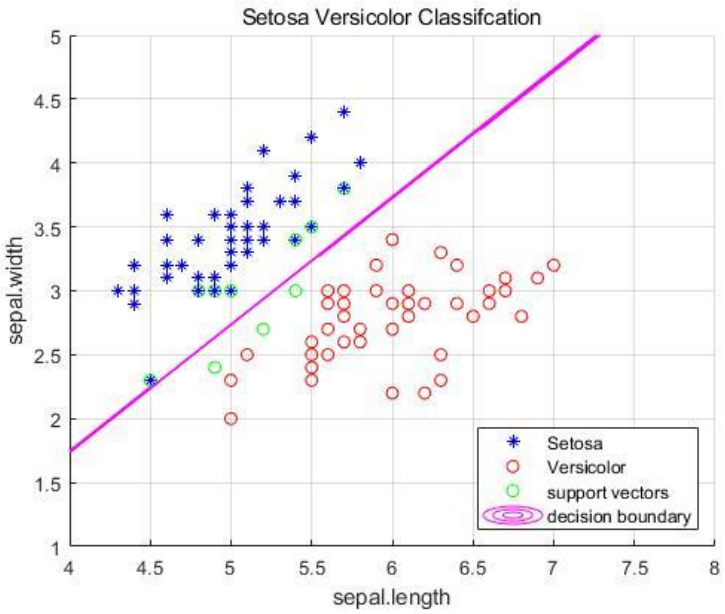

Figure 5. SVM result for data 1, two class were indicated by blue and red colors, support vectors were labels with green, and the pink boundary is the decision boundary..

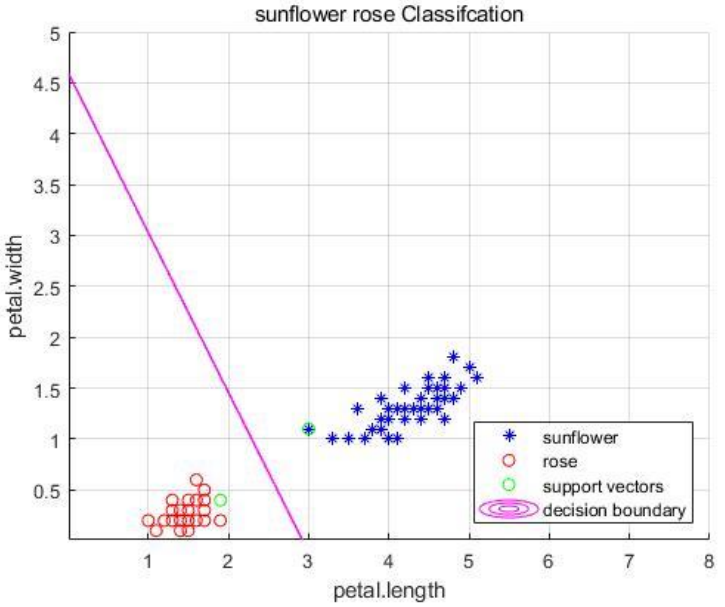

Figure 6. SVM result for data 2, two class were indicated by blue and red colors, support vectors were labels with green, and the pink boundary is the decision boundary..

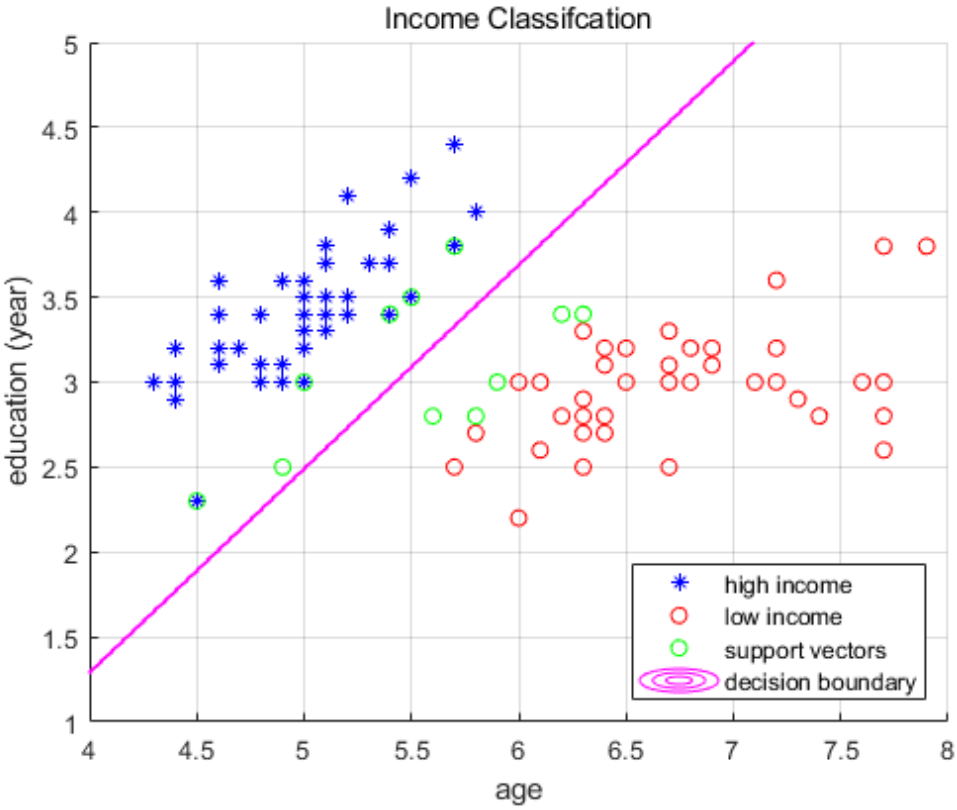

Figure 6. SVM result for data 3, two class were indicated by blue and red colors, support vectors were labels with green, and the pink boundary is the decision boundary..

Inner Product Kernel Method

For the nonlinear classification problem in the input space, it can be transformed into a linear classification problem in a certain dimensional feature space through nonlinear transformation, and a linear support vector machine can be learned in a high-dimensional feature space. Since in the dual problem of linear support vector machine learning, the objective function and the classification decision function only involve the inner product between the instance and the instance, so there is no need to explicitly specify the nonlinear transformation, but to replace the inner product with the kernel function. product. The kernel function represents the inner product between two instances after a nonlinear transformation.

Consider a second order polynomial kernel:

$P_2(\mathbf{x},\mathbf{y}) = (x'y + c)^2$

can be rewritten as the product of two vectors:

$P_2'(\mathbf{x},\mathbf{y}) = [\text{sqrt}(2c)\mathbf{x}, \mathbf{x}^2, c] \cdot [\text{sqrt}(2c)\mathbf{y}, \mathbf{y}^2, c]$

As for the Guassian Kernel, for the sake for simplicity, we take a logarithm of the kernel than we would have a second order polynomial, than we can reuse the method above to compute the inner product. After we obtained the inner product, we than need to take an exponential of it to convert it back.

We compute the kernel and dot product results, and substrate the two to find the absolute difference, as shown in figure 8, the difference is almost zero.
After testing, the kernel trick is about twice faster than computing the dot product, since dot product method divide it into two parts.

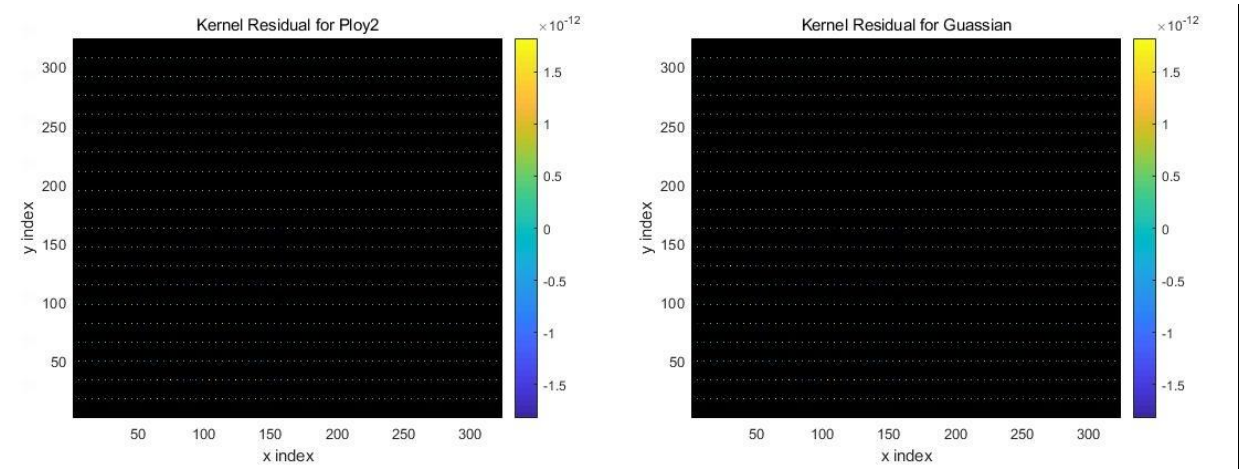

Figure 8, Kernel and dot product residual for polynomial and Gaussian

Improvements to Platt's SMO Algorithm [4]

Looking at the SMO algorithm, its core is how to select the two Lagrange multipliers for each round of optimization. The standard SMO algorithm selects the multiplier to be optimized by judging whether the multiplier violates the KKT condition of the original problem. There may be a question, review the KKT condition, whether it violates it is related to these factors: Lagrange multiplier, sample label, bias b. The update of b depends on two optimized Lagrangian multipliers, which may occur: the Lagrange multiplier has already made the objective function optimal, and the SMO algorithm itself cannot determine the current situation due to the two the b calculated by the optimized Lagrangian multiplier the one that makes the objective function optimal. They violated the KKT conditions, resulting in some time-consuming and useless searches. In view of the shortcomings of the standard SMO, the following improved methods have appeared. They also use the KKT conditions to select the multipliers. The difference is that the dual problem is used in KKT condition.
To realize the improved version of SMO, we simply add a tolerance to KKT condition. It took 70 seconds to iterate through lambdas (CV) for the Blob data, with the improved version, it only took 50 seconds, it achieved 28% speed up.

Conclusion

The nonlinear mapping is the theoretical basis of the SVM method, and the SVM uses the inner product kernel function to replace the nonlinear mapping to the high-dimensional space; the optimal hyperplane for dividing the feature space is the goal of the SVM, and the idea of maximizing the classification margin is the SVM method. Support vector is the training result of SVM, and it is the support vector that plays a decisive role in SVM classification decision; a small number of support vectors determine the sample results, which makes the algorithm have high "robustness". It is a convex optimization problem, so the local optimal solution must be the global optimal solution.

SVM is difficult to implement for large sample data training, because SVM uses quadratic programming to solve the support vector, which involves the calculation of the m-order matrix, which consumes a lot of space and time. An improved method for the above problems is the SMO algorithm.

The classic SVM only gives the algorithm of binary classification, but in practice, it is more of a multi-classification problem, so we generally solve it by constructing a combination of multiple binary support vector machines. There are mainly one-to-many patterns, one-to-one patterns and SVM decision trees.

For a practical problem, the selection of kernel functions is empirical and does not have a reasonable explanation, so there is still no good method in different fields. to solve the problem of kernel function selection.

Reference

Matlab Code:
https://github.com/nengyifu/ML-project-2

[1] Leo Breiman. Bagging predictors. Machine Learning, 24(2):123–140, Aug 1996.
[2] Leo Breiman. Random forests. Machine Learning, 45(1):5–32, Oct 2001.
[3] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. IEEE Intelligent Systems and their applications, 13(4):18–28, 1998.
[4] S. Sathiya Keerthi, Shirish Krishnaj Shevade, Chiranjib Bhattacharyya, and Karuturi Radha Krishna Murthy. Improvements to platt's smo algorithm for svm classifier design. Neural computation, 13(3):637–649, 2001.
[5] John Platt. Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges, and A. Smola, editors, Advances in Kernel Methods: Support Vector Learning, pages 185–208. MIT Press, Cambridge, MA, 1998.
[6] Machine Learning CS7335 Slides, Supervised Learning: Support Vector Machines, Sandip Sen.

[7] Data 3: https://archive.ics.uci.edu/ml/datasets/census+income
[8] Data 1: https://archive.ics.uci.edu/ml/datasets/iris
[9] Data 2: https://archive.ics.uci.edu/ml/datasets/leaf