MIT 9.520/6.860 Project: Feature selection for SVM

Abstract

We consider sparse learning binary classification problems solved with linear support vector machines. We present two popular methods for variable selection: SVM-Recursive Feature algorithm and 1-norm SVM, and propose a third hybrid 1-norm RFE. Finally, we implement this three algorithms and compare their performances on synthetic and microarray datasets. 1-norm SVM gives the lowest test accuracy on synthetic datasets but selects more features. SVM-RFE is the most performant approach on real datasets. 1-norm RFE proves to obtain a good classification accuracy while selecting the smallest support on all kinds of datasets.

1 A brief review of SVM

1.1 Initial motivation of SVM

We consider a set of n pairs of training data $\{(x_i, y_i)\}_{i=1}^n$, $(x_i, y_i) \in \mathbb{R}^p \times \{-1, 1\}$. Our goal is to learn a linear function to obtain a classification rule of our data: $\text{class}(x) = \text{sign}(f(x) = wx + \beta)$. When the data points is linearly separable (the two classes can be separated by an hyperplane), we aim at finding the optimal separating hyperplan that maximizes the margin M between the two classes. Consequently, we consider the following maximization problem:

$$\max_{w \in \mathbb{R}^p, \beta \in \mathbb{R}} M \text{ subject to } \forall i,\ y_i(x_i^T w + \beta) \geq M$$

Using the scability of the solutions, we set $\| w \|_2 = \frac{1}{M}$ and we obtain the equivalent problem:

$$\min_{w \in \mathbb{R}^p, \beta \in \mathbb{R}} \frac{1}{2} \| w \|_2^2 \text{ subject to } \forall i,\ y_i(x_i^T w + \beta) \geq 1$$

1.2 Primal and dual formulations

When the data is not linearly separable, we still want to maximize M by allowing some points to be in the wrong size of the margin. Hence, we define the support vector machine **primal** problem:

$$\min_{w \in \mathbb{R}^p, \beta \in \mathbb{R}} \frac{1}{2} \|w\|_2^2 + C \sum_{i=1}^n \max\left(0, 1 - y_i(x_i^T w + \beta)\right) \tag{1}$$

C is a penalization parameter which controls the trade-off between the classification error and the norm of the estimator. We also consider the **dual** of the SVM problem:

$$\begin{array}{l} \max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{1 \leq i,j \leq n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ \text{subject to } \left\{ \begin{array}{l} \forall i,\ 0 \leq \alpha_i \leq C \\ \sum_{i=1}^n \alpha_i = 0 \end{array} \right. \end{array} \tag{2}$$

The representer theorem guarantees that the solution can be written as a linear combination of the traing data : $w_* = \sum_{i=1}^n \alpha_i y_i x_i$ We call support vectors the vectors such that $\alpha_i \neq 0$.

MIT 9.520/6.860: Statistical Learning Theory and Applications (Fall 2016).

1.3 Motivation for feature selection

In the past twenty years, the avaibility of high scale datasets with hundreds of thousands of variables -such as gene expression microarray or text categorization datasets- has stimulated the interest for research in feature selection and variable ranking algorithms. Guyon and al. (2003) point out in (2) the three advantages of such methods: they improve the speed and performances of the predictor, as well as the understanding of the underlying data.

We now assume our data is in high-dimention: $p \gg n$. The stucture of this paper is as follows. In Section 2 and 3, we present two common approaches for variable ranking using SVM. In Section 4, we propose our implementation of the two methods, as well as a third hybrid one. In Section 5, we compare the classification performances of the three algortihms on synthetic and real datasets.

2 SVM-RFE algorithm

2.1 Correlation criteria

(2) opposes *ranking variable* and *variable subset selection* methods for feature selection. *Ranking variable* algorithms rank each feature by measuring how it contributes individually to class distinction. The simplest idea is to use a correlation ranking: several coefficients can be computed. For instance, one could use a ranking according to the absolute value of the Pearson coefficient of the variables with the output, or to the coefficient Golub et al. (1999) propose in (3)

The predicition classifier returned by such a method is a combination of univariate classifiers. It can only detect linear dependencies between variable and target. Redundancy among features will not be eliminated. However, in practice, algorithms using mutual information between features have proved to obtain better classification performances. Consequently, we now focus on *variable subset selection* methods, where predictors are trained on several features to compute a ranking.

2.2 The Recursive Feature Elimination algorithm

Let us start by presenting the popular Recursive Feature Elimination algorithm for SVM, introduced by Guyon et al. (2000) in (4):

Initialization : Ranked = [], Var = [1,...,N];
while Var is not empty **do**

1. Train a SVM classifier on the whole training set for the variables in Var
2. Compute the weight and ranking vectors w and r
3. Eliminate the feature $r_{\min}$ with lowest weight: Var = Var-$\{r_{\min}\}$ with $r_{\min} = \arg\min r$
4. Update the ranking list Ranked = [Ranked $r_{\min}$]

end

SVM-RFE algortihm rank each feature according to its position of elimination, while handling mutliple features over training. Note that now, the top ranked features are not necesary the most relevant ones: only a subset of features is optimal. The authors proved that RFE eliminates gene redundancy, and then leads to more compact subset of features with better classification performances.

2.3 A coordinate descent algorithm on the dual problem

Step1 requires to solve one of the primal or dual SVM problems on the training set. For large-scale datasets, the most common current algorithms are respectively subgradient descent algorithms on the primal problem or coordinate descent algorithms on the dual problem. Some of the first class of algorithms -see Leon Bottou's SGD in (5) or Pegasos mini-batch iterations in (6)- prove to outperform every other solver for $n > 100K$.

However in our hypothesis $p \gg n$ we have a smaller number of features than this threshold. In this regime, a coordinate descent algortihm on the dual has proved to be faster than the state of art of solvers: see (10) for further details. Consequently we will use it for step 1.

2.4 A ranking criterion for RFE

Step 2 requires the existence of a ranking criterion. Rakotomamonjy (2003) in (7) compares two ranking approaches for SVM: a zero-order method where the removed variable is the one whose elimination minimizes the variation of the objective value, and a first order method which measures the sensitivity of a given criterion with respect to each variable. He shows that the derivative ranking criterion gives good performances and stability among datasets. Note that this two methods are equivalent for linear SVM: the feature to be removed is the one with the lowest absolute value in the weight vector: $r_{\min} = \arg\min_i |w_i|$.

2.5 Computational time

It is computationally more efficient to reuse part of the calculations in order to speed up the while loop: the current solution can serve as a warm start to the next step. Besides, when the number of features is large, eliminating one feature at each step is very time consuming. Several features can be removed at a time, at the expense of possible classification performances degradation.

In (4), the authors claim that only for small subset of features (less than 100), better results are obtained by eliminating one feature at a time. Consequently we start by removing a bunch of features at every iteration, and then only one when the reduced dataset is of size of hundreds.

2.6 Cross-validation and choice of the hyperparameter

Two control parameters appear in RFE: the SVM parameter C and the number of feature to be removed. We use cross-validation across a grid of hyperparameters to select our model.

Hastie and al. (2004) in (8) compute the entire regularization path for SVM. They start with the smallest value of C, and then increase it. When C is small, w tend to be small, hence the margin $\frac{1}{\|w\|}$ is large: all the vectors are support vectors. As suggested in (9), the smallest C should verify $\forall i, y_i x^T w < 1$. Then a choice can be: $C_{\min} = (n \max \|x_i\|_2^2)^{-1}$.

When we increase C, the number of support vectors decreases.

2.7 RFE and boostrap

In (12), Duan et al. (2005) apply a boostrap resampling of the reduced dataset at every iteration of the RFE algorithm. Then they train one SVM on each subsample, normalize the predictor, and propose a weight for each variable as the ratio between the mean and the standard deviation of the list of the coefficient associated to this variable for every weight vectors. Their approach proves to reduce the test error on four real datasets (among which the leukemia and the lung cancer we later use).

3 1-norm SVM

3.1 Linear programming formulation

A sparse SVM classifer can also be obtained by using an l_1 penalization in the primal problem:

$$\min_{w \in \mathbb{R}^p, \beta \in \mathbb{R}} |w|_1 + C \sum_{i=1}^{n} \max\left(0, 1 - y_i(x_i^T w + \beta)\right) \tag{3}$$

This new formulation has two nice properties. First, similarly to LASSO for the regression problem, it uses the l_1 penalization to enforce the sparsity of the solution and provide feature selection. Second, it transforms the original Quadratic programming problem into a Linear programming, and makes it computationally faster to solve. Consequently, let us consider the LP reformulation of 3:

$$\min_{z \in \mathbb{R}^p, \xi \in \mathbb{R}^n} \sum_{i=1}^{p} z_i + C \sum_{i=1}^{n} \xi_i$$
$$\text{subject to } \begin{cases} \forall i \leq p,\ z_i \geq w_i,\ z_i \geq -w_i \\ \forall i \leq n,\ \xi_i \geq 0,\ \xi_i \geq 1 - y_i(x_i^T w + \beta) \end{cases}$$

Note that Zhu et al. (2003) proposes in (14) an algorithm to compute the whole solution path, and make easier the selection of the tuning parameter. They also present results where 1-norm-SVM improve the classical SVM for classification performances with redundant noise features.

3.2 1-norm RFE: polishing 1-norm SVM using SVM-RFE

For regression problems, LASSO can be seen as a feature selection algorithm. A better estimator may be obtained by running a Least Squares on the support of the LASSO estimator.

Similarly, for classification problems, we can try to improve 1-norm SVM and select less variables. Hence a natural idea is to reduce the input matrix to the support of the 1-norm SVM estimator, and then run the RFE algorithm on this restricted space. If the new estimator is different to the previous one, it has necesary a smaller support. We call this approach 1-norm RFE and will compare it to the two previous algorithms in Section 5.

3.3 Other approaches

Obtaining sparse solutions for SVM has been a widely studied problem, and an impressive benchmark of methods and heuristics has been proposed. A similar popular approach, refered as l_1-SVM in (4), (10) and (11) is to use an l1-penalization, but a square hinge loss function.

Alternatively, Tan et al. (2010) propose in (13) an l_0 penalization to control the sparsity of the solution. They obtain an Mixed Integer Programming formulation and solve its relaxation with a cutting plane algorithm. They proved good performances of their algorithm in very high dimensional problems.

4 Description of the methods compared

4.1 Presentation of the methods

We compare three methods on both experimental and real datasets:

1. SVM RFE solved with coordinate descent on the dual
2. 1-norm-SVM solved with Gurobi
3. 1-norm RFE by polishing the 1-norm SVM solution on its support with SVM-RFE

Our goal is to train the first two algorithms on several dataset for a similar time in order to compare their test accuracy performances and the number of features selected.

For every method, the optimal model will be chosen by 5-fold stratify cross-validation. Folds are obtained by randomly splitting the training set, so that the proportion of the two classes in each fold is the same as the one in the whole training set. As recomanded in (15) we try an exponentially growing sequence for the hyperpameter C (from 2^{-10} to 2^{10}, and use the same grid for all three methods. We resolve ties in cross-validation by computing the estimator on the whole training set and keeping the one with smallest support. Note that from our later observations, the solution is rather insensitive to the value of C because the training data sets are linearly separable down to just a few features.

4.2 Implementing the SVM RFE algorithm

The authors of the dual coordinate descent algortihm in (10) released a package called LIBLINEAR to implement their solution - presented in (11) - which is an extension of LIBSVM. Besides, Scikit learn library proposes a Recursive Feature Elimination algorithm for any given estimator (in our case an SVM with an hinge loss and an l2-penalization solved with LIBLINEAR). The number or fraction of features to be removed at every iteration is selected by cross-validation.

4.3 Implementing the 1-norm-SVM

For the 1-norm SVM, we code our own appraoch, using a GUROBI solver along a given regularization path. Let us note that classical libraries do not propose this implementation. We also implement a stratify k-fold cross validation. The speed of computing the path for one fold is highly increased by keeping the Gurobi model along the path and simply updating its penalizer C at every iteration.

5 Simulation

5.1 Synthetic datasets simulations

Our simulations examples are inspired from (16). We consider an input matrix, with rows being independant realizations of a p dimentional multivariate normal centered distribution. Only k_0 dimensions are relevant for classification. We propose three different settings:

Example 1 and 2: We generate n samples, half from each class. Let $\mu_+ \in \{0,1\}^n$ and $\mu_- = -\mu_+$: then the data from the ± 1 class respectively have the distribution: $\forall i,\ x_i^{\pm} \sim \mathbb{N}(\mu_{\pm}, \Sigma)$

Example 1: $\mu_+^i = 1$ for k_0 equi-spaced values (or nearest rounded integer), $\mu_+^i = 0$ otherwise. We introduce a correlation ρ and consider: $\forall i, \Sigma_{i,i} = 1$ and $\forall i \neq j, \Sigma_{i,j} = \rho^{|i-j|}$. Note that for $\rho = 0$, the variables are independent.

Example 2: Similar to Example 1 except that: $\forall i \neq j, \Sigma_{i,j} = \rho$ and $\mu_+ = (\frac{1}{k_0}, \frac{2}{k_0}, ..., 1, \mathbf{0}_{p-k_0})$,

Example 3: $\forall i,\ x_i \sim \mathbb{N}(\mu, \Sigma)$ with $\mu = (\mathbf{1}_{k_0}, \mathbf{0}_{p-k_0})$, Σ is the same as in Example1. We now use a probit model: $\mathbb{P}(Y = 1) = \Phi(X\mu)$ where Φ is the CDF of the standard distribution. Note that the training set is not necesary balanced.

For all three examples, we will use different kind of settings for ρ under the assumption that p is large. All computations were carried out with processor 1,6 GHz Intel Core i5 and 8GB of RAM.

5.2 Synthetic datasets results

For each of the three examples, we present one relevant configuration of (n, p, k_0). We consider every value of ρ in the range $\{0, 0.2, 0.5, 0.8\}$ and run 5 simulations for each value. We present the averaged test errors and number of features selected. Standard deviations appear in brackets.

Example 1: the average training time was less than 3 minutes for each of the two first methods.

Table 1: Test errors and number of features selected for Example 1 with $n = 100, P = 2000, k_0 = 10$

ρ	1-norm-SVM		SVM RFE		1-norm RFE	
	Test	Features	Test	Features	Test	Features
0	.6(1.2)	8.6(1.9)	3(2.3)	5.6(2.4)	2.6(1.8)	4.8(.7)
.2	.6(.8)	10.5(2)	2.6(1.4)	4.8(1.3)	2.8(1.4)	4.4(1.1)
.5	1(.6)	8.6(.8)	1.8(1.3)	5(1.3)	1.8(1.2)	4.8(1.2)
.8	0(0)	13(6.5)	.8(.4)	5(.9)	.8(.4)	5(.9)

For every coefficient ρ, 1-norm-SVM gives the lowest test misclassifcation error and selects a support close to the optimal one: for 17 out of the 20 iterations, it gets a support of size 8, 9 or 10 only with true positive coefficients. The other two approaches select a smaller support, with only true positive coefficients, but get a lower test accuracy. 1-norm RFE tends to select slightly less variables.

Example 2: average training time was between 2 and 10 min.

Table 2: Test errors and number of features selected for Example 2 with $n = 100, p = 2000, k_0 = 10$

ρ	1-norm-SVM		SVM RFE		1-norm RFE	
	Test	Features	Test	Features	Test	Features
.2	1.8(1.5)	35.2(9)	4.2(2)	21(9.5)	4.4(1.5)	9.2(1.3)
.5	.4(.5)	44(6)	3(1.4)	12(3.5)	2.8(1.6)	9.8(1.2)
.8	.2(.4)	28(5.3)	.8(1.2)	5.4(1.4)	2.6(1.8)	3.8(.4)

1-norm-SVM clearly outperforms the two other methods for test accuracy. However it selects a very high number of variables. 1-norm RFE improves this first step for feature selection. It has a misclassification error similar to SVM-RFE but it selects a more sparse support. Note that surprisingly, we have better results for higher correlations: we do not have any explanation so far.

Example 3: the number of features selected is higher than for the two previous cases: consequently we restrict p to 500. Average training time was less than 30 seconds.

Table 3: Test errors and number of features selected for Example 3 with $n = 100, p = 500, k_0 = 10$

ρ	1-norm-SVM		SVM RFE		1-norm RFE	
	Test	**Features**	**Test**	**Features**	**Test**	**Features**
.2	22.6(6)	33.2(19)	27.4(4)	30.6(11)	26.4(3)	25(9)
.5	15.6(5)	39.6(20)	20.6(6)	45.8(18)	18.6(3)	30.8(13)
.8	12.8(4)	26(12)	15.4(4)	27(11)	13.4(3)	14.4(3)

First note that the standard deviations are high (we round them to the nearest integer): for different simulation with the same parameters, there is a big disparity in the models returned. Second, note that again, the model selected tends to be more accurate and less dense for higher correlation.

SVM-RFE performs very badly for this kind of dataset: it is always but once ranked third for both metrics. 1-norm RFE is not so far from 1-norm-SVM for test accuracy, and gives the smallest support.

Consequently, whereas there were no real clear winner between SVM-RFE and 1-norm RFE on the two previous cases, our new approach offers more robustness to this third synthetic datasets. It appears as a good trade-off between a good test accuracy and a sparse support in all three simulations.

5.3 Real microrarray datasets

We now compare our three methods on real microarray datasets. Microarray datasets contain the expressions of thousands of hundreds of genes collected from a relatively small number of samples. Hence it is particularly relevant to select a very few number of genes for later research.

The first dataset classifies two kinds of lung cancers. The defaut training set contains 32 samples, 16 of each class. The defaut test set contains 149 samples: 134 from one class and 15 from the other. Each sample is patient by 12533 genes. The dataset is available at: http://datam.i2r.a-star.edu.sg/datasets/krbd/LungCancer/LungCancer-Harvard2.html

The second dataset divides patients with Leukemia in two categories. The defaut training set contains 27 features from one class and 11 from the other. The test set contain 20 patients from the first class and 14 from the other. Each patient is described by 6817 genes. The dataset is available at: http://datam.i2r.a-star.edu.sg/datasets/krbd/Leukemia/ALLAML.html

5.4 Microrarray datasets results

We use a two-step preprocessing method. First, we mix the defaut train and test set, and we center and standardize each column by substracting its mean and dividing the result by its standard deviation. Then we shuffle this new dataset and split it randomly into a new train and test sets with a same ratio between the two classes. We run 5 iterations for all three methods and keep the best result:

Table 4: Results for the Lung cancer dataset

Methods	Test error	Features
1-norm SVM	4/91	26
SVM-RFE	3/91	3
1-norm RFE	5/91	2

Table 5: Results for the Leukemia dataset

Methods	Test error	Features
1-norm SVM	4/37	24
SVM-RFE	1/37	15
1-norm RFE	4/37	4

For the Lung cancer dataset, 1-norm SVM selects a very high number of features, but doesn't get the best test accuracy. SVM-RFE algorithm performs extremely well: it gets the lowest misclassification error by only selecting 3 genes. Interestingly, the support of size 2 returned by our 1-norm RFE is included in the support of size 3 of SVM-RFE.

For the Leukemia dataset, the observations are quite similar: SVM-RFE algorithm performs the best, and SVM-RFE support is the largest. However 1-norm RFE greatly improves the performances of 1-norm SVM: it keeps the same accuracy but selects 6 times less genes and 3 times less than SVM-RFE. Note that our results are slightly different from Hastie et al. in (14).

6 Conclusion

We have focused on two very popular methods for feature selection and variable ranking using SVM, and mixed them to propose a new one. Curious reader may deepen its understanding of the existing work and solutions using the references this paper provides.

In our simulations, we have shown that 1-norm SVM selects more features than the other two methods. It is the most accurate algorithm on all three synthetic datasets we have proposed. On the two real datasets, SVM-RFE gives the best results. Note that in both cases, our 1-norm RFE approach leads to great imporvements for variable selection: without degradating a lot 1-norm SVM classification performances, it selects the smallest support of all three methods. It is also more robust to different synthetic datasets than SVM-RFE.

References

[1] J. Friedman, R. Tibshirani, T. Hastie, The elements of statistical learning: data mining, inference, and prediction, 2nd edn. Springer, NewYork, 2009

[2] I. Guyon and A. Elisseeff, An introduction to variable and feature selection, JMLR, 3:1157–1182, 2003

[3] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, E. S. Lander, Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring, Science, 286, 531–537, 1999

[4] I. Guyon, J. Wetson, S. Barnhill, V. Vapnik, Gene Selection for Cancer Classification using Support Vector Machines, Machine Learning, 46:389422, 2002

[5] L. Bottou, Large-scale machine learning with stochastic gradient descent, International Conference on Computational Statistics,177–187, 2010

[6] S. Shalev-Shwartz, Y. Singer, N., Srebro, Pegasos: Primal estimate sub-gradient solver for SVM, Mathematical Programming, 127:3-30, 2011

[7] A. Rakotomamonjy. Variable selection using SVM-based criteria. JMLR, 3:1357–1370, 2003

[8] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, The entire regularization path for the support vector machine. Journal of Machine Learning Research, JMLR 5:1391–1415, 2004

[9] B.-Y. Chu, C.-H. Ho, C.-H. Tsai, C.-Y. Lin, C.-J. Lin, Warm Start for Parameter Selection of Linear Classifiers, ACM KDD, 2015

[10] C. Hsieh, K. Chang, C. Lin, S. Keerthi, S. Sundararajan: A dual coordinate descent method for large-scale linear SVM, ICML, 408–415, 2008

[11] R. E. Fan, K. W. Chang, C. J. Hsieh, X. R. Wang, C. J. Lin, LIBLINEAR: A Library for Large Linear Classification, Journal of Machine Learning Research 9:1871-1874, 2008

[12] K. B. Duan, J. C. Rajapakse, H. Wang, F. Azuaje, Multiple SVM–RFE for Gene Selection in Cancer Classification Witn Expression Data, IEEE Transactions on Nanobioscience, 4, 228–234, 2005

[13] M. Tan, L. Wang, and I. W. Tsang. Learning sparse svm for feature selection on very high dimensional datasets, ICML, 1047–1054, 2010

[14] J. Zhu, S. Rosset, T. Hastie, R. Tibshirani, 1- norm support vector machines, Advances in Neural Information Processing Systems, 2003

[15] C. W. Hsu, C. C. Chang, C. J. Lin, A practical guide to support vector classification, C. W. Hsu, C. C. Chang, C. J. Lin, A practical guide to support vector classification, 2003

[16] G.-B. Ye, Y. Chen, X. Xie, Efficient variable selection in support vector machines via the alternating direction method of multipliers, International Conference on Artificial Intelligence and Statistics, pp. 832–840, 2011