SVM in Analysis of Cross-Sectional Epidemiological Data

Dmitriy Fradkin

Overview

- The goals of analyzing cross-sectional data
- Standard methods used in epidemiology
- Need for model validation and alternative methods
- Support Vector Machines
- Cross-Validation
- Recursive Feature Elimination
- Local models
- Conclusion

Cross-Sectional Studies

In *cross-sectional studies* a random sample from a population is selected and measures of disease occurance and variables of interest are obtained, at a single point in time.

One of the most cost effective data collection methods is via questionnaires. The variable of interest can be numerical, categorical, ordinal, etc. Frequently there is missing data.

Outcome is usually binary.

Goal: To identify a small number of risk factors (by building a model of the data).

Ideally, an "intervention study" follows to determine the effect of manipulating the risk factors.

Model Building in Epidemiology

1. Conduct Univariate Analysis (compute correlation between variables and labels)
2. Select significant variables (or variables believed to be biologically significant)
3. Multivariate Analysis (logistic regression) of the selected variables
4. Choose a final model
5. Analysis of coefficients for significance

Problems with the "Standard" Approach

There are problems with how epidemiologists often analyze the data:

- Interactions between variables are not considered during univariate analysis - important variables can be lost
- Logistic regression model is rarely validated - overfit is possible (and even quality of fit is often not presented)
- The models are taken as the "truth"

The above problems are aggravated by the fact that intervention studies are expensive, require lots of paperwork and are thus frequently not done. The recommendations and policies are frequently set based on the models alone.

Ways to address these issues

There is need for:

- Alternative methods (to be used in addition to logistic regression)
- Model validation methods to obtain estimates of predictive performance and stability of the model
- Multivariate feature selection methods

All of these have already been developed in the field of Machine Learning.

Support Vector Machines (SVM)

Support Vector Machines can be thought of as a method for constructing linear classifiers with theoretical guarantees of good predictive performance (the quality of classification on unseen data). The theoretical foundation of this method is given by statistical learning theory [Vapnik 1998].

While no general method for producing non-linear rules with such properties is known, the so-called "kernel trick" can be used to construct special kinds of non-linear rules using the SVM methodology.

Applications of SVM

Support Vector Machines (SVM) recently became one of the most popular classification methods, applied to

- text classification [Joachims 1998],
- facial expression recognition [Michel et. al 2003],
- gene analysis [Guyon et. al. 2002],
- and many others.

A Linearly Separable Set

In a two class case think of a classifier as a function $f(x) : \Re^d \rightarrow \Re$. A point is assigned to the positive class if $f(x) \geq 0$, and to the negative class otherwise.

A classifier function $f(x)$ is linear if it can be expressed as:

$$f(x; w, b) =< w, x > +b \tag{0.1}$$

where w, b are parameters of the function and $<, >$ denotes the inner product of two vectors.

A set of points (x_i, y_i), $i = \overline{1, l}$, where $y_i \in \{-1, +1\}$ are class labels, is called *linearly separable* if a linear classifier can be found so that $y_i f(x_i) > 0$, $\forall i = 1, \ldots, l$.

The Concept of Margin

A hyperplane

$$< w^*, x > + b^* = 0, \ ||w^*|| = 1 \tag{0.2}$$

is called γ-margin separating hyperplane if

$$y_i(< w^*, x > + b^*) \geq \gamma \tag{0.3}$$

for all (x_i, y_i) in set S. Here γ (clearly $\gamma > 0$) is the margin.

Any separating hyperplane can be converted into this form.

The Concept of Margin

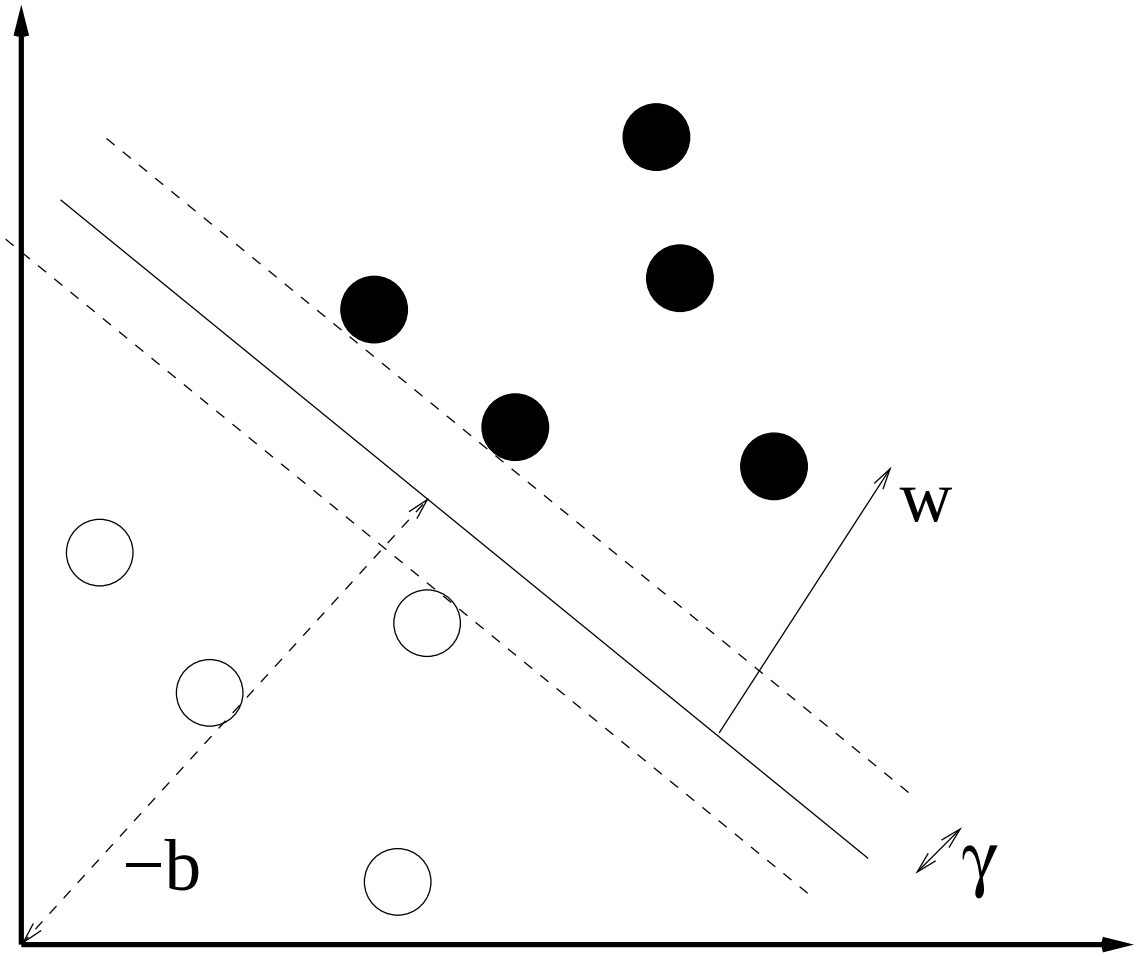


Figure 1: Here $(w, -b)$, $||w|| = 1$, define the separating hyperplane and γ is the size of the margin.

Perceptron Algorithm

Require: A linearly separable set S, learning rate $\eta \in \Re^+$

1: $w_0 = 0; b_0 = 0; k = 0;$
2: $R = \max_{1 \leq i \leq l} ||x_i||$
3: **while** at least one mistake is made in the **for** loop **do**
4: **for** $i = 1, \ldots, l$ **do**
5: **if** $y_i(< w_k, x_i > + b_k) \leq 0$ **then**
6: $w_{k+1} = w_k + \eta y_i x_i$
7: $b_{k+1} = b_k + \eta y_i R^2$
8: $k = k + 1$
9: **end if**
10: **end for**
11: **end while**
12: Return w_k, b_k, where k is the number of mistakes

Novikov's Theorem

THEOREM 0.1. *Novikov 1962. Let S, $|S| = l$, be a training set with*

$$R = \max_{1 \leq i \leq l} ||x_i|| \tag{0.4}$$

Suppose that there exists a γ-separating hyperplane f such that $y_i f(x_i) \geq \gamma$, $\forall\, 1 \leq i \leq l$. Then the number of mistakes made by the on-line perceptron algorithm on S is at most

$$\left(\frac{2R}{\gamma}\right)^2 \tag{0.5}$$

This theorem effectively proves that for a linearly separable set of points a particular method (the perceptron algorithm) finds a separating hyperplane after making a finite number of mistakes. The number of mistakes (in the training stage) is directly proportional to the ratio of the volume of the data to the measure of separation of the classes, γ.

VC Dimension

The *Vapnik-Chervonenkis (VC) dimension* of a set of classifiers is the maximum number h of points that can be separated into all possible 2^h ways using classifiers in this set. If for any n there exists a set of n points that can be separated into two classes in all possible ways, the VC dimension of this set of functions is said to be infinite.

For example, the VC dimension of hyperplanes in R^d is known to be $d + 1$.

The following two results bound the VC dimension of the set of γ-margin separating hyperplanes and the probability of misclassifying an unseen instance with such a hyperplane chosen on the training data.

THEOREM 0.2. ***[Vapnik 1998, Thm. 5.1]*** *Let $x \in X$ belong to sphere of radius R. The the set of γ-margin separating hyperplanes has VC dimension h bounded by:*

$$h \leq min\left(\left(\frac{R}{\gamma}\right)^2, d\right) + 1 \tag{0.6}$$

COROLLARY 0.1. *With probability $1-\eta$ the probability of a test example not being separated correctly by a γ-margin hyperplane has the bound*

$$P_{error} \leq \frac{m}{l} + \frac{E}{2}\left(1 + \sqrt{1 + \frac{4m}{lE}}\right) \tag{0.7}$$

where

$$E = 4\frac{h(\ln\frac{2l}{h} + 1) - \ln\frac{\eta}{4}}{l}, \tag{0.8}$$

m is the number of training examples not separated correctly by γ-margin hyperplane, and h is the bound of the VC dimension given in theorem 0.1.

Interpretation of the Above Bounds

The bound on the probability of making a mistake on unseen data is proportional to the VC dimension of the set of classifiers. In other words, everything else being equal, a classifier with a lower VC dimension is likely to be a better predictor. Notice that this result does not depend on any assumptions about the distribution of the test data — it is a "distribution-free" bound.

Since the upper bound on VC dimension is inversely proportional to the margin, the strategy for building a good classifier is to have as large a margin as possible while keeping the number of errors on the training set low.

This is somewhat similar to the idea regularization but is motivated from the perspective of the statistical learning theory. It can also be seen as a version of Occam's razor: we want to use the simplest classifier that makes no (or fewest) mistakes on the training set.

Maximal Margin Classifier

$$\text{minimize}_{w,b} < w, w > \tag{0.9}$$

subject to:

$$y_i(< w, x_i > +b) \geq 1, \ \forall i = 1, \ldots, l \tag{0.10}$$

Introducing Lagrange multipliers,α_i for the constraints gives the so-called Lagrangian function:

$$L(w, b, \alpha) = \frac{1}{2} < w, w > - \sum_{i=1}^{l} \alpha_i [y_i(< w, x_i > +b) - 1] \tag{0.11}$$

The Dual Optimization Problem

Taking derivatives with respect to w and b gives:

$$w = \sum_{i=1}^{l} y_i \alpha_i x_i \tag{0.12}$$

and

$$0 = \sum_{i=1}^{l} y_i \alpha_i \tag{0.13}$$

Re-substituting these into the primal problem (0.11) gives a dual formulation:

$$\text{maximize } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j < x_i, x_j > \tag{0.14}$$

subject to:

$$\sum_{i=1}^{l} y_i \alpha_i = 0, \ \alpha_i \geq 0, \ \forall i = 1, \ldots, l \tag{0.15}$$

Support Vectors

Let α^* be a vector of parameters optimizing $W(\alpha)$. The the weight vector $w^* = \sum_{i=1}^{l} y_i \alpha_i^* x_i$ is a maximal margin hyperplane, with margin

$$\gamma = \frac{1}{||w^*||_2}. \tag{0.16}$$

The parameter b is not present in the dual formulation and has to be computed from the primal problem.

Notice that the classifier $f(x)$ can be expressed as:

$$f(x) =< w, x > +b = \sum_{i=1}^{l} y_i \alpha_i < x_i, x > +b \tag{0.17}$$

If $\alpha_i = 0$, then x_i is not used in decision rule and can be discarded. Points x_i such that $\alpha_i \neq 0$ lie on the margin and are called support vectors. They determine the decision boundary.

Extension for the non-separable case I

The main idea is that those points that lie on the wrong side of the hyperplane are explicitly penalized by introducing *slack* variables, ξ, that control how far on the wrong side of a hyperplane a point lies.

The optimization problem becomes:

$$\text{minimize}_{w,b} \frac{1}{2} < w, w > + C \sum_{i=1}^{l} \xi_i \tag{0.18}$$

subject to:

$$y_i(< w, x_i > + b) \geq 1 - \xi_i, \ \xi_i \geq 0, \ \forall i = 1, \ldots, l \tag{0.19}$$

where the parameter C, controlling the trade-off between the size of the margin and the training errors, is chosen by the user.

Extension for the non-separable case II

The dual then becomes:

$$\text{maximize } \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j < x_i, x_j > \tag{0.20}$$

subject to:

$$\sum_{i=1}^{l} y_i \alpha_i = 0,\ C \geq \alpha_i \geq 0,\ \forall i = 1, \ldots, l \tag{0.21}$$

Once again the solution is given by a linear combination of inner products with support vectors: $w = \sum_{i=1}^{l} y_i \alpha_i x_i$

The "kernel trick"

The main idea behind the "kernel trick" is to map the data into a different space, called *feature space*, and to construct a linear classifier in this space. It can also be seen as a way to construct non-linear classifiers in the original space.

Notice that in the the dual problem (0.14) the training points are included only via their inner products. Also, as can be seen in (0.17), the classifier function $f(x)$ can be expressed as a sum of inner products with support vectors.

What is a Kernel?

Mercer's theorem states that any symmetric positive semi-definite function $K(x, z)$ is an inner product in some space (and vice-versa). In other words, any such function $K(x, z)$ implicitly defines a mapping into so-called feature space $\phi : x \rightarrow \phi(x)$ such that $K(x, z) =< \phi(x), \phi(z) >$. Such functions K are called kernels.

The feature space can be high-dimensional or even have infinite dimension. However we don't need to know the actual mapping since we can use kernel function to compute similarity in the feature space.

Some examples of kernels include polynomial kernels

$$K(x, z) = (< x, z > +1)^p \tag{0.22}$$

and gaussian kernels

$$K(x, z) = e^{-\frac{||x-z||^2}{2\sigma^2}}. \tag{0.23}$$

Kernel form of SVM

By replacing the inner product in the formulation of SVM by a kernel and solving for Lagrange multipliers α_i, we can obtain via (0.17) a maximal margin separating hyperplane in the feature space defined by this kernel. Thus choosing non-linear kernels allows us to construct classifiers that are linear in the feature space, even though they are non-linear in the original space.

The dual problem in the kernel form is:

$$\text{maximize } W(\alpha) = \sum_{i=1}^{l} \alpha_i - \sum_{i,j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \qquad (0.24)$$

$$\text{subject to } 0 = \sum_{i=1}^{l} y_i \alpha_i, \ \alpha_i \geq 0, \ \forall i = 1, \ldots, l \qquad (0.25)$$

and the classifier is given by:

$$f(x) = \sum_{i=1}^{l} y_i \alpha_i K(x_i, x). \qquad (0.26)$$

SVM Packages

SVM implementations are now available in Matlab and other numeric packages.

Some stand-alone SVM software:

- SVMLight
- LIBSVM

Motivation for Cross-Validation

- Predictions of the model on new data give the strongest statistical estimates of the model validity.
- However, data is "expensive" - we don't want to discard part of the data just to obtain a performance estimate
- Re-sampling schemes of the restricted data can be used to validate a model without use of a hold-out set.

Cross-Validation

One of the most popular re-sampling schemes is k-fold cross-validation [Stone 1977, Efron 1982]. This procedure consists of k identical stages:

- The set of observations is divided into k equal subsets
- Choose one of these to be the "test" set. The remaining $k-1$ subsets are then combined to form a "training" set
- Build a model on the training set, evaluate on the test set
- Repeat for a new test set (i.e. k times)

This results in a prediction for each available observation, and k estimates of model coefficients. If this procedure is repeated t times (with a different partition into k sets every time), we have kt estimates on the coefficients, and t estimates of the predictive accuracy of the model on all observations.

Comments on Cross-Validation

- It is independent of the type of distribution function.
- It allows direct measurement of variance of all the coefficients of the model; this is very convenient from the practical point of view.
- If the test confirms that a model has a high validity, the coefficients can be used to numerically compare the significance of variables. Moreover, because the validity is also measured numerically, it is possible to compute confidence of the model validity estimates.

Stratified Cross-Validation

Stratified cross-validation differs from regular cross-validation in that the test set is required to have the same distribution of classes as the whole set. Another way to think of this is that each class is partitioned into folds separately and the corresponding folds are then combined.

A number of papers compared different accuracy estimation methods (hold-out set, cross-validation, bootstrap). The recommendation of [Kohavi et. al 1995] is to use stratified 10-fold cross-validation.

Feature Selection

There are two general approaches to feature selection: *filter methods* and *wrapper methods* [John, Kohavi and Pfleger 1994].

Filter methods evaluate variables based on some measure of importance, usually individually, without building a predictor.

Wrapper methods evaluate sets of features based on the estimated quality of the model built with these features.

Recursive Feature elimination (RFE)

[Guyon et. al 2002] It is based on the observation that the feature i with the smallest absolute value of weight $|w_i|$ is the one that has the least influence on the decision.

Require: A set of points W in $\Re^d$; f - number of desired features

1: **for** $i = d, \ldots, f+1$ **do**
2: Construct an SVM on set W. Let w^i be the decision hyperplane of that SVM.
3: Let $k_i = \text{argmin}_j |{w_j}^i|$.
4: Remove feature k_i from all points x in W.
5: **end for**
6: Output resulting classifier with f features and hyperplane w^f.

Results on Microarray Data

These results are from [Guyon et. al 2002].

1. Leukemia dataset: 38/34 training/test split, 7129 features (class ratio 2:1).
2. Colon cancer dataset: 31/31 training/test split, 2000 features (class ratio 2:1).

Feature reduction, while giving 100% accuracy:

1. Leukemia: 8, 16
2. Colon cancer: 8

The selected genes were biologically relevant.

Cross-Sectional Data: Wet Litter

Wet litter is a term used when the material covering the floors of poultry houses, usually consisting of wood shavings or chopped straw, reaches its saturation threshold and is incapable of holding more moisture.

It has been shown to be associated with the occurrence of foot-pad, breast and hock lesions, and inducing high levels of stress in the birds.

Results on Cross-Sectional Data

Data on occurance of wet litter in chicken farms in the UK. There are 444 points (248 cases with wet litter and 196 cases without it), 177 features. The model built by an epidemiologist has 12 features. [Hermans et. al. 2005].

Cross-validation sensitivity/specificity of that model: 0.724/0.629
Using RFE:

1. With 6 features, 5 are shared with epidemiological model: 0.679/0.507
2. With 12 features, 6 are shared: 0.753/0.525

Analysis of union of the feature sets suggests presence of correlated variables.

These results are from [Fradkin et. al. 2005].

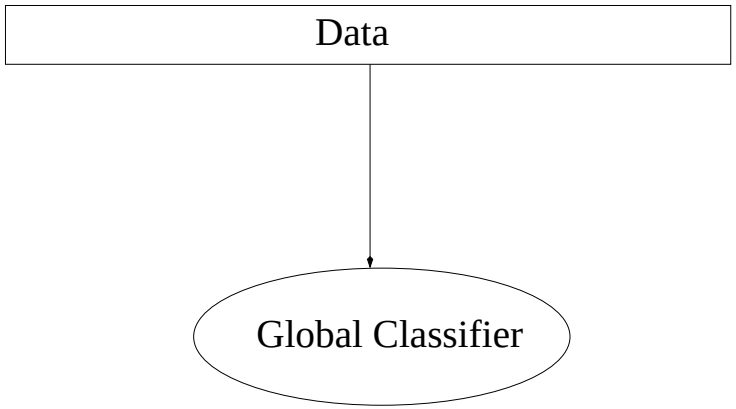
Local Models

- By a local model we mean a model that is was built using a subset of the data
- It may provide better results or a better explanation that a global model (build on the whole data) in a particular region

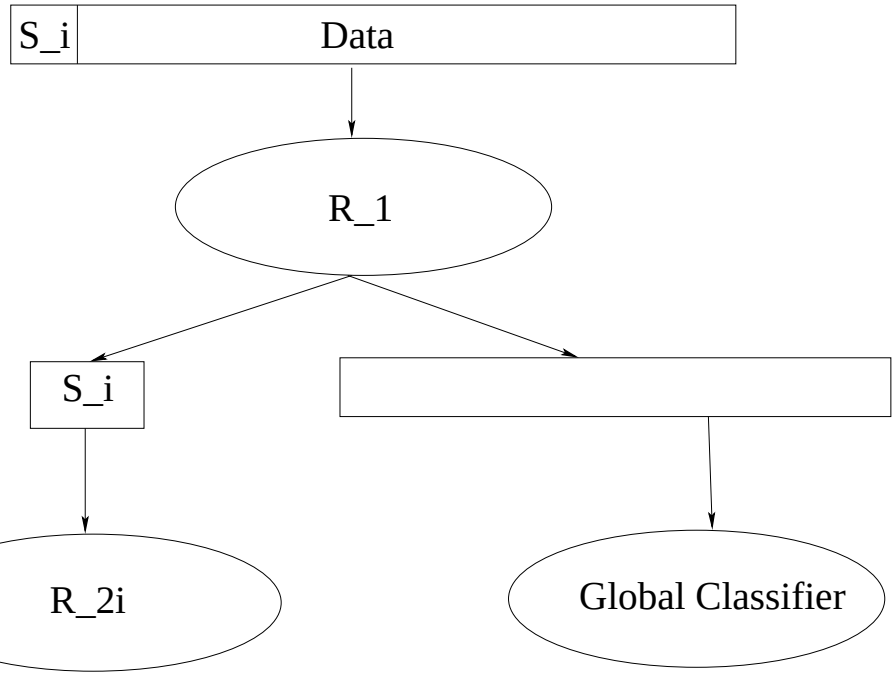
We will discuss an example of how local models can be found and evaluated.

Graphical Representation

Standard Approach



Single Cluster Hierarchy



Single Cluster Hierarchy 1: Training

Require: A set W, cluster $S_i \in W$.

1: Train a global classifier R_0 to distinguish between classes of W.

2: Train classifier R_{1i} to separate S_i (class 1) and W/S_i (class 0).

3: Train classifier R_{2i} on points in S_i.

4: Return R_0,R_{1i} and R_{2i}.

Single Cluster Hierarchy 1: Classification

Require: Classifiers R_0,R_{1i} and R_{2i}; a point x.

1: Let $c = R_{1i}(x)$.
2: **if** $c = 1$ **then**
3: Return $R_{2i}(x)$.
4: **else**
5: Return $R_0(x)$.
6: **end if**

Other Single Cluster Hierarchies

Note that we can have variations on method SCH_1:

- SCH_2: Build local models R_{2i} on S_i and $R_{\overline{2i}}$ on W/S_i and use both instead of R_0
- SCH_3: Use the global model R_0 on the cluster but build a separate local model $R_{\overline{2i}}$ on the complement W/S_i.

Plan: partition the data into k clusters and for each cluster evaluate SCH_1, SCH_2 and SCH_3 schemes. Keep track of the classification rules constructed.

Notation

$\alpha(R|S, L_h)$ - the expected probability of classifier R correctly classifying a point $x \in S$ of class L_h. For classifiers R_0, R_1 and R_{2i} this value can be estimated by cross-validation.

$p(S_i|L_h)$ - the probability of a point of class L_h belonging to cluster S_i. This numbers can be estimated by fractions of points of class L_h that fall into cluster S_i.

Estimation of Performance

We can estimate the accuracy of our Single Cluster Hierarchy 1 as follows:

$$\begin{aligned}\alpha(SCH_1|W, L_h) = p(S_i|L_h)\alpha(R_{1i}|W, S_i)\alpha(R_{2i}|S_i, L_h) + \\ p(W/S_i|L_h)\alpha(R_{1i}|W, W/S_i)\alpha(R_0|W/S_i, L_h) + \\ p(S_i|L_h)(1 - \alpha(R_{1i}|W, S_i))\alpha(R_0|S_i, L_h) + \\ p(W/S_i|L_h)(1 - \alpha(R_{1i}|W, W/S_i))\alpha(R_{2i}|W/S_i, L_h)\end{aligned}$$

Note that if our classifier into clusters (R_{1i}) has high accuracy, the last two terms make very minor contribution.

Experiments

Data: wet litter caused by disease. The epidemiological model was built on 313 farms (1/3 had wet litter) and included 5 features.[Hermans et. al, 2005]

Accuracy Estimates for $k = 2$ (with SVM)

S_i	$\alpha(SCH_1\|W,*)$	$\alpha(SCH_2\|W,*)$	$\alpha(SCH_3\|W,*)$
2-1	$(0.415, 0.907)$	$(0.415, 0.894)$	$(0.318, 0.914)$
2-2	$(0.318, 0.914)$	$(0.415, 0.894)$	$(0.415, 0.907)$

Table 1: SVM results. Global performance is: $(class1, class0)$=$(0.319, 0.926)$.

SVM Feature Weights

Classifier	Features				
	1	2	3	4	5
SVM R_0	0.613	0.278	0.221	-0.223	0.369
	0.0643	0.1975	0.1572	0.1701	0.2618
SVM R_1	0.000	0.000	0.000	0.000	0.955
	0.0001	0.0000	0.0000	0.0001	0.0001
SVM R_{21}	0.797	0.261	0.819	-0.345	0.000
	0.1342	0.2641	0.1989	0.1720	0.0000
SVM R_{22}	0.532	0.064	0.000	-0.066	0.000
	0.0001	0.2331	0.0001	0.2384	0.0000

Table 2: Feature analysis for clusters 2-1 and 2-2, SVM.

Observations on Local Models

- Local models can lead to better or worse performance than global models,
- Local models can have rather different feature weights

Ideas To Take Away

- SVM is a very popular and powerful method.
- Epidemiology is an interesting application area for Machine Learning methods.
- The focus is on finding risk factors - models have to be interpretable.
- It is important to do model validation.