The Pennsylvania State University

The Graduate School

Department of Electrical Engineering

FEATURE SELECTION METHODS FOR SUPPORT VECTOR MACHINES FOR TWO OR MORE CLASSES, WITH APPLICATIONS TO THE ANALYSIS OF ALZHEIMER'S DISEASE AND ITS ONSET WITH MRI BRAIN IMAGE PROCESSING

A Dissertation in

Electrical Engineering

by

Yaman Aksu

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2010

The dissertation of Yaman Aksu was reviewed and approved* by the following:

Kenneth Jenkins
Professor of Electrical Engineering
Head of the Department of Electrical Engineering

David J. Miller
Professor of Electrical Engineering
Dissertation Co-Advisor
Co-Chair of Committee

Qing X. Yang
Professor of Radiology and Neurosurgery
Co-Chair of Committee

George Kesidis
Professor of Electrical Engineering
and Computer Science and Engineering
Dissertation Co-Advisor

Constantino M. Lagoa
Associate Professor of Electrical Engineering

James Wang
Professor of Information Sciences and Technology

*Signatures on file in the Graduate School.

Abstract

Feature selection for classification in high-dimensional spaces can improve generalization, reduce classifier complexity, and identify important, discriminating feature "markers". For support vector machine (SVM) classification, a widely used technique is Recursive Feature Elimination (RFE). We demonstrate RFE is not consistent with margin maximization, central to the SVM learning approach. We thus propose explicit margin-based feature elimination (MFE) for SVMs and show both improved margin and improved generalization, compared with RFE. Moreover, for the case of a nonlinear kernel, we show RFE assumes the squared weight vector 2-norm is strictly decreasing as features are eliminated. We demonstrate this is not true for the Gaussian kernel and, consequently, RFE may give poor results in this case. We show that MFE for nonlinear kernels gives better margin and generalization. We also present an extension which achieves further margin gains, by optimizing only two degrees of freedom – the hyperplane's intercept and its squared 2-norm – with the weight vector orientation fixed. We finally introduce an extension that allows margin slackness. We compare against several alternatives, including RFE and a linear programming method that embeds feature selection within the classifier design. On high-dimensional gene microarray data sets, UC Irvine repository data sets, and Alzheimer's disease brain image data, MFE methods give promising results. We then develop several MFE-based feature elimination methods for the case of more than two classes (the "multiclass" case). We compare against RFE-based multiclass feature elimination and show that our MFE-based methods again consistently achieve better generalization performance. In summary, we identify some difficulties with the well-known RFE method, especially in the kernel case, develop novel, margin-based feature selection methods for linear and kernel-based two-class and multiclass discriminant functions for support vector machines (SVMs) addressing separable and nonseparable contexts, and provide an objective experimental comparison of several feature selection methods, which also evaluates consistency between a classifier's margin and its generalization accuracy.

We then apply our SVM classification and MFE methods to the challenging problem of predicting the onset of Alzheimer's Disease (AD), focusing on predicting conversion from Mild Cognitive Impairment (MCI) to AD using only a single, first-visit MRI for the person so as to aim for early diagnosis. In addition, we apply MFE for selecting brain regions as disease "biomarkers". For these aims, for the pre-classification image data preparation step, we co-develop an MRI brain image processing pipeline system named STAMPS, as well as develop a related system with additional capabilities named STAMPYS. These systems utilize external standard MRI brain image processing tools and generate output image types particularly suitable for detecting (and encoding) brain atrophy for Alzheimer's disease. We identify and remedy some basic MRI image processing problems caused by some limitations of external tools used in STAMPS – i.e. we introduce our basic fv (fill ventricle) algorithm for ventricle segmentation of cerebrospinal fluid (CSF). For prediction of conversion to AD for MCI patients, we demonstrate that our early diagnosis system achieves higher accuracy than similar recently published methods.

Table of Contents

List of Tables

List of Figures

Notation

$\mathbf{R}$	**the set of real numbers**
$\mathbb{N}$	**the set of natural numbers**
M	**number of features** (pre-elimination)
N	**number of training samples**
$\underline{x}$	**training vector**, aka **training sample**
X	**set** $\{\underline{x}_1, \ldots, \underline{x}_N\}$ **of training vectors**
T	**number of support vectors**
$\mathcal{S}$	**set** $\{1, \ldots, T\}$ **of indexes for support vectors**
$\underline{s}$	**support vector** $\in X$, e.g. $\underline{s}_l$ is l^{th} **support vector** with $l \in \mathcal{S}$
k	**number of classes**
y	**class label** $\in \{\pm 1\}$ **for two-class case**
y_n	y **of** $\underline{x}_n$
$y_{\underline{x}}$	y **of** $\underline{x}$, e.g. $y_{\underline{s}_l}$ is y **of** $\underline{s}_l$
p	**class label** $\in \{1, \ldots, k\}$ **for multiclass case**
p_n	p **of** $\underline{x}_n$
$p(\cdot, \cdot)$	**probability density function** (pdf)
c_n	**minimum-signed-distance class** for sample $\underline{x}_n$ in multiclass SVM
$\underline{w}$	**weight vector**
b	**offset**, aka intercept, aka affine parameter
δ or Δ	**delta**, a quantity computed by the MFE method herein
$\langle \cdot, \cdot \rangle$	**inner product**
$\underline{\phi}(\cdot)$	**mapping into feature space**, possibly infinite-dimensional
$K(\cdot, \cdot)$	**kernel function**
$\mathcal{F}$	**space induced by a kernel**
$f(\cdot)$	**discriminant function**
ξ	**slackness variable**
γ	**margin**
λ	**Lagrange multiplier** (uses same subscripting as y above)
$\zeta_{n,p}$	**binary (indicator) variable**; 1 if p_n (the class of $\underline{x}_n$) is p, 0 otherwise.
Λ	**Lagrange coefficient**, calculated from multiple Lagrange multipliers in multiclass SVMs
C	**regularization parameter for SVMs**
$\mathcal{X}$	**space of inputs for a learning machine**
$\mathcal{Y}$	**space of outputs for a learning machine**
$\mathbf{x}$	**input to a learning machine** ($\mathbf{x} \in \mathcal{X}$), such as a **training vector** $\underline{x}$
$\mathbf{y}$	**output of a learning machine** ($\mathbf{y} \in \mathcal{Y}$), such as a **class label** y
h	**VC dimension**
$L(\cdot, \cdot)$	**loss function**
$R(\cdot)$	**expected risk**
$R_{emp}(\cdot)$	**empirical risk**
$\binom{k}{l}$	**number of combinations** (k-choose-l), aka binomial coefficient

Acknowledgments

For all of their help and guidance, I am grateful to my thesis co-advisor and committee co-chair Prof. David Miller, thesis co-advisor Prof. George Kesidis, and committee co-chair Prof. Qing Yang, who were also my three supervisors. I am grateful to my committee members Prof. Constantino Lagoa and Prof. James Wang. Many thanks to Dr. Don Bigler, for our collaboration in creating and using the STAMPS software and for a number of useful discussions on MRI brain images and their processing, and to everyone who helped me at PSU.

Chapter 1

Introduction

1.1 Statistical Learning Theory and learning machines

Statistical learning theory is concerned with producing a function $f : \mathcal{X} \to \mathcal{Y}$ that estimates from an input $\mathbf{x} \in \mathcal{X}$ an output $\mathbf{y} \in \mathcal{Y}$ considered to be the "truth" associated with $\mathbf{x}$. For instance, $\mathbf{x}$ may be a vector $\underline{x} \in \mathbf{R}^M$ ($M \in \mathbb{N}$) of observables and $\mathbf{y}$ the class $y \in \{\pm 1\}$ that $\underline{x}$ belongs to. More precisely, a "learning machine" is defined by a family of possible mappings $\mathbf{x} \mapsto f(\mathbf{x}, \Theta)$ parameterized by Θ, with "learning" (aka training) being the step of employing a "training set" X of input samples in order to choose a particular Θ value[1] whereby one is simply left with the desired mapping $\mathbf{x} \mapsto f(\mathbf{x})$.

"Generalization" refers to the ability to estimate the "true" output for a previously unseen ("test") $\mathbf{x}$ sample (which was not employed by the training). Related to generalization is the notion of "capacity" of the learning machine, an abstract concept that simply refers to the machine's ability to learn any training set without error.[2] For example, trained on data about vehicles, such as images of vehicles, a machine that declares a bicycle is a car because it has wheels is a machine whose capacity is too small, whereas a machine that declares that a two-door red car with round headlights is not a car because of not being white with four doors and rectangular headlights is a machine whose capacity is too large. By finding the right balance between accuracy obtained on a particular training set and capacity, best generalization performance can be achieved. Capacity is made concrete using a measure, a non-negative integer, called the VC dimension (Vapnik Chervonenkis dimension) – the VC dimension is a property of a family of functions and is discussed shortly.

When $\mathbf{y}$ values for the training samples are provided to the learning (as available ground-truth), the learning is called "supervised" – otherwise it is called "unsupervised". The general assumption is that $\mathbf{x}$ and $\mathbf{y}$ are drawn from an unknown cumulative probability distribution $P(\mathbf{x}, \mathbf{y})$. Based on the idea of a loss function $L : \mathcal{Y} \times \mathcal{Y} \to \mathbf{R}$, the loss (i.e. penalty) of making an incorrect decision $f(\mathbf{x}, \Theta)$ when the true output is $\mathbf{y}_{true}$ is $L(f(\mathbf{x}, \Theta), \mathbf{y}_{true})$ – thus the expected risk is stated as:

$$R(\Theta) = \int_{\mathcal{X},\mathcal{Y}} L(f(\mathbf{x}, \Theta), \mathbf{y}) dP(\mathbf{x}, \mathbf{y}). \tag{1.1}$$

Four components (75) are needed for building a learning machine. The first component is the domain, which is the learning problem with its associated loss function $L(\cdot, \cdot)$ – for example, classification, regression (or regression estimation), and probability density estimation are different problems of function estimation i.e. different learning problems. In classification, $\mathcal{Y}$ is a set of k classes (categories), and $L(u, v)$ is 1 if $u = v$, and is 0 otherwise – for example, if $\mathcal{Y} = \{\pm 1\}$, $L(u, v)$ is $\frac{1}{2}|v - u|$. In regression, $\mathcal{Y} = \mathbf{R}$, and one can similarly use $L(u, v) \equiv |v - u|$. In density estimation, $L(f(\mathbf{x}, \Theta), p) \equiv -\log(p(\mathbf{x}))$ can be used, where p is the density being estimated.

---

[1] Θ is a set of parameters called "hyperparameters". A Θ value is a set of hyperparameter values.

[2] Caution: Terminology is not ideal – normally an excess of "ability" or "capacity", such as in humans, would not result in poor performance, whereas in the machine context too much capacity is essentially associated with resulting poor performance e.g. in the form of data overfitting (a type of undesirable "overlearning" of the data).

The second component needed for building a learning machine is an induction principle for defining a decision rule. The optimal f in (1.1) is the one that minimizes the expected risk $R(\Theta)$. As established by Bayesian decision theory, the Bayesian decision minimizes the expected risk $R(\Theta)$ and is thus optimal (20). Thus, in the event one has complete knowledge of the probability density function $p(\mathbf{x}, \mathbf{y})$ (and thus $dP(\mathbf{x}, \mathbf{y})$ which is $p(\mathbf{x}, \mathbf{y})d\mathbf{x}d\mathbf{y}$), using the Bayesian approach one would be able to infer the optimal f (20; 28). However, very often in practice $p(\mathbf{x}, \mathbf{y})$ is not available, and even though one could compute its estimate $\hat{p}(\mathbf{x}, \mathbf{y})$, this estimation is a difficult step and will at best yield a suboptimal result limited by the richness of the training data (20; 28). Requiring neither this estimation nor a priori knowledge about the probability distribution (e.g. the analytic form of $p(\mathbf{x}, \mathbf{y})$), statistical learning theory approaches the problem of analyzing the expected risk via generating bounds for it. The bound analysis is an integral part of the main induction principle in statistical learning theory, named Structural Risk Minimization (SRM), discussed below in Sec. 1.2.

The final two of the four components for building a learning machine (75), which are 3) a set of decision functions, and 4) an algorithm for implementing the first three components (1-3), depend on the particular choice of learning formulation and thus our discussion of them will be addressed throughout this thesis.

1.2 VC dimension and Structural Risk Minimization (SRM)

VC dimension is a property of a parameterized family of functions $\{f(\Theta)\}$ (discussed in Sec. 1.1). Suppose there is a set of N observations $X = \{\underline{x}_1, \ldots, \underline{x}_N\}$ with associated "truth" $Y = \{y_1, \ldots, y_N\}$. Since in this thesis we focus on the classification problem, we now convey a concrete definition for VC dimension by focusing on the two-class classification problem as an example. Given a family of functions $\{f(\Theta)\}$, if one can find, for each of the 2^N possible ways of labeling the N points, a family member able to correctly assign this particular labeling to the N points, we say (using terminology of (6)) that this set of points is "shattered" by this family. The VC dimension is defined as the maximum number of points that the family can shatter (6). The VC dimension of the family being h does not mean that every set of h points can be shattered by this family (6). The partitioning of the data space by a hyperplane into two half-spaces is referred to as a "linear dichotomy", and the VC dimension of a set of hyperplanes in $\mathbf{R}^M$ is $M+1$ (6).

Suppose $L(\cdot, \cdot)$ only takes the values 0 and 1, and $L(u, v)$ is $\frac{1}{2}|v - u|$ as stated above, as this thesis will focus on classification. Given the sets X and Y above, the bound (1.2) holds with probability $1 - \eta$ (where $0 \leq \eta \leq 1$ can be chosen arbitrarily small) for a learning machine with VC dimension h (69):

$$R(\Theta) \leq R_{emp}(\Theta) + \sqrt{\frac{h(log(2N/h) + 1) - log(\eta/4)}{N}}. \tag{1.2}$$

The second term on the right-hand side, which contains the VC dimension, is called the VC confidence. $R_{emp}(\Theta)$, given by (1.3), is the "empirical risk", the mean error rate measured on the training set.

$$R_{emp}(\Theta) = \frac{1}{2N} \sum_{n=1}^{N} |y_n - f(\underline{x}_n, \Theta)| \tag{1.3}$$

Since $R_{emp}(\Theta)$ is a fixed number for a given choice of training set $\{\underline{x}_n, y_n\}$ and the Θ chosen by the training, the right-hand side of (1.2), which is independent of $P(\mathbf{x}, \mathbf{y})$, can be calculated if one knows h – the underlying assumption is that the training data and the test data are drawn independently according to *some* $P(\mathbf{x}, \mathbf{y})$ (6). Thus, a principled way of choosing a learning machine is to choose from among a set of candidates the one with the lowest value for the right-hand side of (1.2). To minimize the right-hand side, there is a tradeoff between minimizing the

training error and the VC confidence depending on the complexity of the machine through the machine capacity measure h. The induction principle Structured Risk Minimization (SRM) has been defined so as to introduce a "structure" into the candidate set F^{cands} by considering nested subsets $F_1 \subset F_2 \subset \ldots \subset F_Z$ (for some integer Z, with $F_Z \subseteq F^{cands}$) with associated known VC dimensions $h_1 < h_2 < \ldots < h_Z$ (or known bounds on the VC dimensions). In this way, for a given subset F_i, the goal would be to minimize the empirical risk R_{emp} among members of that subset. Accordingly, upon training a number of machines across subsets (possibly as few as simply one machine per subset), one can simply choose the machine with the least sum of empirical risk R_{emp} and VC confidence. The bound is guaranteed not tight when the VC confidence, which monotonically increases with h, exceeds a threshold (which, clearly, would be relative to the maximum value chosen for the loss function) – for example, for the 0-1 loss defined above, a reasonable threshold is 1.[3] Overall the bound (1.2) is simply a guide – among two machines achieving zero empirical risk, it is possible for the one with the higher VC dimension to achieve better generalization performance. The bound is not valid for infinite VC dimension, which, by definition, refers to the ability to shatter an arbitrarily large number of points. Note, however, that machines from a family of infinite VC dimension are not guaranteed to generalize poorly – for example, the k-nearest-neighbor classifier, which classifies a sample by voting based on labels of its k nearest neighbors. Along this line, (43; 44) introduced the NNSRM (nearest-neighbor-SRM) classifier which focuses on combining the power of the SRM principle with the versatility of the NN classifier – the classifier is proposed as a guaranteed-to-converge and computationally less expensive alternative to the Support Vector Machine (SVM) which, to be discussed shortly, is *"one of the first practical learning procedures for which useful bounds on the VC dimension could be obtained and hence the SRM program could be carried out"* (39).

Additional examples on the VC dimension, including examples discussing it in the context of a function family's number of parameters, can be found in (6). Next, in Sec. 1.3, we give a brief review of Support Vector Machines (SVMs), which will also serve to lead into our subsequent discussion of the connection between SRM and SVM in Sec. 1.4.

1.3 Brief review of Support Vector Machines (SVMs)

Consider a labeled training set $\{(\underline{x}_n, y_n), n = 1, \ldots, N\}$, where $y_n \in \{\pm 1\}$ is the class label and $\underline{x}_n = (x_{n,1}, \ldots, x_{n,M}) \in \mathbf{R}^M$ is the n-th data sample. A hyperplane acting as a binary (two-class) decision function in this M-dimensional space is defined by $f(\underline{x}) \equiv \underline{w}^{\mathrm{T}}\underline{x} + b = 0$, $\underline{w} \in \mathbf{R}^M$, $b \in \mathbf{R}$. Denoting $g(\underline{x}_n) \equiv y_n f(\underline{x}_n)$, the signed distance from a data point $\underline{x}_n$ to the decision boundary is $\frac{g(\underline{x}_n)}{||\underline{w}||}$. The decision boundary is a *separating* one if it satisfies $g(\underline{x}_n) > 0$ for all $\underline{x}_n$. The margin of the separating decision boundary is thus defined as $\gamma \equiv \frac{\min_n g(\underline{x}_n)}{||\underline{w}||}$. A support vector machine (SVM) is a linear or generalized linear two-class classifier that learns a separator for the training set with *maximum* margin. The "support vectors", used to specify the SVM solution, which we denote by $\{\underline{s}_1, ..., \underline{s}_T\}$ (with index set $\mathcal{S} = \{1, 2, ..., T\}$), are a subset of the training points at margin distance to the decision boundary. In the linear case, the SVM weight vector solution is $\underline{w} \equiv \sum_{k \in \mathcal{S}} \lambda_{\underline{s}_k} y_{\underline{s}_k} \underline{s}_k$, where $\lambda_{\underline{s}_k}$ are scalar (positive) Lagrange multipliers. In the generalized linear (nonlinear) case, $f(\underline{x}) \equiv \underline{w}^{\mathrm{T}}\underline{\phi}(\underline{x}) + b = 0$, $\underline{w} \in \mathbf{R}^L$, $b \in \mathbf{R}$, $\underline{\phi}(\underline{x}) \equiv [\phi_1(\underline{x}),\ \phi_2(\underline{x}),\ \ldots,\ \phi_L(\underline{x})]^{\mathrm{T}}$, with $\phi_i(\underline{x})$ nonlinear functions of the $\underline{x}$ coordinates. Of particular interest is when inner products between $\underline{\phi}(\underline{x})$ and $\underline{\phi}(\underline{u})$ can be efficiently computed via a positive definite kernel function, $K(\underline{x}, \underline{u}) \equiv \underline{\phi}^{\mathrm{T}}(\underline{x})\underline{\phi}(\underline{u})$. In this case, both $\underline{\phi}(\cdot)$ and $\underline{w}$ itself need

[3] For example, for this case, (6) plotted the VC confidence versus h/N for $N = 10,000$ and $\eta = 0.05$ as well as illustrated that the VC confidence exceeds 1 for $h/N > 0.37$.

not be explicitly defined since both the SVM discriminant function $f(\cdot)$ and the SVM weight vector squared 2-norm can be expressed solely in terms of the kernel, *i.e.*:

$$f(\underline{x}) = \sum_{k \in \mathcal{S}} \lambda_{\underline{s}_k} y_{\underline{s}_k} K(\underline{s}_k, \underline{x}) + b, \tag{1.4}$$

$$||\underline{w}||^2 = \sum_{k \in \mathcal{S}} \sum_{l \in \mathcal{S}} \lambda_{\underline{s}_k} y_{\underline{s}_k} \lambda_{\underline{s}_l} y_{\underline{s}_l} K(\underline{s}_k, \underline{s}_l). \tag{1.5}$$

This approach (the "kernel trick"), where $K(\cdot, \cdot)$ is explicitly specified and provided to the SVM training, is referred to as the "nonlinear kernel case".

For both linear or nonlinear kernels, the basic SVM training problem is:

$$\min_{\underline{w}, b} \frac{1}{2} ||\underline{w}||^2 \quad s.t. \quad y_n f(\underline{x}_n) \geq 1, \ n = 1, \ldots, N \tag{1.6}$$

Recall that $f(\underline{x}_n)$ in the constraints in (1.6) simply stands for $\underline{w}^{\mathrm{T}} \underline{x} + b$ in the linear case and $\langle \underline{w}, \underline{\phi}(\underline{x}) \rangle + b$ in the nonlinear kernel case – that is, as (1.6) states the basic SVM training problem conveniently for both cases, it is important to keep in mind $\underline{w}$ and b indeed appear in the constraints. The relationship of (1.6) to margin maximization can be understood as follows (3). Assuming we have a separator (i.e. the training data is separable), the margin is $\gamma = \frac{\min_n g(\underline{x}_n)}{||\underline{w}||}$ and, further, note that $g(\cdot) \equiv y f(\cdot)$ can be amplitude-scaled by an arbitrary nonzero constant ρ without altering the decision boundary. In particular, if we form $\tilde{g} = \rho g$, where $\rho = \frac{1}{\min_n g(\underline{x}_n)}$, then $\min_n g(\underline{x}_n) = 1$ consistent with the constraints and $\gamma = \frac{\min_n g(\underline{x}_n)}{||\underline{w}||} = \frac{1}{||\underline{w}||}$. We thus see the well-known result that, for this special choice of ρ maximizing margin is equivalent to minimizing the squared weight vector 2-norm.

The SVM training problem can alternatively be posed as:

$$\min_{\underline{w}, b, \underline{\xi}} \frac{1}{2} ||\underline{w}||^2 + C \sum_{n=1}^{N} \xi_n \quad s.t. \quad \xi_n \geq 0, \ y_n f(\underline{x}_n) \geq 1 - \xi_n, \ n = 1, \ldots, N \tag{1.7}$$

so as to allow slackness ($\underline{\xi}$) in the margin constraints; (1.7) allows some support vectors to be practically closer than others to the hyperplane (by nonnegative slackness amounts ξ_n), thus handling both margin violations (i.e., $\xi_n > 0$) and nonseparable data (a classification error occurs for sample n if $\xi_n > 1$).[4] For choosing the SVM training parameter C as well as other SVM hyperparameters in the nonlinear kernel case, the standard practice of using a validation or cross-validation procedure (20) can be employed. The relationship of (1.7) to margin maximization can be understood as follows (3). If C is made sufficiently large, no margin slackness will be tolerated and minimizing (1.7) *reduces to* minimizing the squared weight vector 2-norm and, thus, to maximizing margin. We thus see that (1.7) is a generalization of strict margin maximization that *specializes* to strict margin maximization when C is made sufficiently large.

Data dimensionality and separability: Cover's linear dichotomy theorem (12) states that the probability that a training set (with points in general position) is linearly separable is very close to 1 if $N \leq M + 1$. As an example, for the gene microarray domain, it is typical to have *e.g.* $M \approx 7000$ and N no larger than a few hundred patient samples; in this case, it is *highly probable* that the training set will be separable while eliminating all the way down to a few hundred features.

[4] Again, recall from $f(\underline{x}_n)$ in (1.7) that $\underline{w}$ and b indeed appear in the constraints in (1.7).

1.4 SVM and SRM

In Sec. 1.2, we discussed that good generalization performance can be achieved via the SRM induction principle wherein one aims for an effective tradeoff between minimizing the training error and a "VC confidence" quantity that depends on the machine capacity measure h. In Sec. 1.3, we discussed the Support Vector machine in particular, and that the SVM training formulation does indeed achieve SVM's objective of maximizing the margin. We have not, however, yet discussed why maximizing the margin is important, in the context of SRM, for good generalization performance – in this section, we focus on this discussion.

As stated in (75), the following result is given in (69).

Theorem 1: *Let R be the radius of the smallest ball $B_R(\underline{a}) = \{\underline{z} \in \mathcal{F} : |\underline{z} - \underline{a}| \leq R\}$, $\underline{a} \in \mathcal{F}$, containing the points $\underline{x}_1, \ldots, \underline{x}_N$, and let*

$$f_{\underline{w},b} = sign(\underline{w} \cdot \underline{x} + b) \tag{1.8}$$

be canonical hyperplane decision functions defined on these points. Then the set $\{f_{\underline{w},b} : ||\underline{w}|| \leq A\}$ has a VC dimension h satisfying

$$h \leq R^2 A^2 + 1. \tag{1.9}$$

As discussed in (75), first, these hyperplane decision functions being *canonical* is referring to the fact that the minimum distance from some training data to the decision boundary is (normalized to) 1[5], and second, for this set of functions the VC dimension can be bounded, due to the above result, in order to implement the SRM principle. Note, however, that the ball around the data points means that this approach to bounding the VC dimension depends on *observed values* of the features – this is similarly noted by (39) as follows (p. 389): *"The original argument for structural risk minimization for SVMs is known to be flawed, since the structure there is determined by the data (see (Vapnik, 1995), Section 5.11)."* [(Vapnik, 1995) being referred to is (69).] Based on the discussion above, for SVMs one can see that *"in a strict sense, the VC complexity of the class is not fixed a priori, before seeing the features"*, as noted again by (39) (p. 389). It is, however, possible to gain insight as to why maximizing the margin, the SVM objective, is considered important for good generalization performance. To that end, (6) presents a family of SVM-like classifiers named "gap tolerant classifiers" which are based around *both* the idea of putting balls around data points *and* hyperplanes – the classifier is specified by the location of a ball (in $\mathbf{R}^M$) and two parallel hyperplanes with parallel normal vectors (in $\mathbf{R}^M$), and the decision function classifies points as one of two classes so long as these points lie inside the ball but not between the hyperplanes (i.e., so long as the points are not members of the so-called "margin set" of points that may lie between the hyperplanes). The VC dimension of such a family of classifiers can be controlled by controlling both the maximum allowed ball diameter and the minimum allowed perpendicular distance between the two hyperplanes (6) – subsequently discussing along this line with examples, (6) argues that it seems very reasonable to conclude that SVMs too gain a similar kind of capacity control from their training due to their training objectives being very similar to gap tolerant classifiers'.

The discussion above suggests that although a *rigorous* explanation for why SVMs often achieve good generalization performance is not provided by SRM alone there is clearly a theoretical connection between SVMs' objective of margin maximization and SRM. SVMs have become nearly a standard technique in many domains. There are a number of reasons. First, as also discussed above, the SVM objective, maximizing the margin, has a theoretical basis tied to achievement of good generalization accuracy (14). Second, there is a unique, globally optimal solution to the SVM training problem. Third, there are improvements in representation power

[5] Recall that Sec. 1.3 explained this particular normalization concept applicable to hyperplanes.

via nonlinear kernels, which map to a high or even infinite-dimensional feature space and, via the "kernel trick", do so *without* huge increase in decision-making and classifier training complexities. Fourth, SVMs achieve good results on a variety of domains.

1.5 Feature selection in classification

In high-dimensional domains such as image and image sequence classification, text categorization, and gene microarray classification, one often encounters problems where there are very few labeled training samples, or at any rate few samples *relative to* the (high) dimensionality of the feature measurements for each exemplar/sample. In biomedical imaging and bioinformatics in particular, with training databases derived *e.g.* from clinical trials, there may be at most several hundred (patient) samples, each represented by features such as voxels in the hundreds of thousands or gene microarray or text features in the tens of thousands. In these domains, there are compelling reasons for reducing feature dimensionality. First, many features may have at best weak discrimination power. In (68), a type of "curse of dimensionality" (COD) was demonstrated wherein, for fixed sample size, the generalization accuracy may *degrade* as feature dimensionality is increased beyond a certain point. This phenomenon is related to the bias-variance dilemma in statistics (39), which suggests that, for best generalization, model complexity should be matched to available training data resources. More specifically, models with relatively higher complexity (e.g. models with relatively larger number of free parameters) tend to achieve low bias (i.e. low accuracy irrespective of the particular choice of training set) but increased variance (i.e. the output of the learning (e.g. the decision boundary learned in classification) will vary widely from one particular choice of training set to the next). In regression, the estimation error is additive in bias and variance, whereas in (two-class) classification the interaction between bias and variance can be highly nonlinear and multiplicative (20) – more specifically, variance has a nonlinear effect on "boundary error" (deviation from correct estimation of optimal (Bayes) boundary) and this effect depends on a "boundary bias" quantity (20). In classification, since it is important to achieve good generalization accuracy, it is generally more important to achieve low variance than low boundary bias (20). Even in domains where generalization accuracy tends to monotonically improve with feature dimensionality, complexity of the classification operation (both computation and memory storage for decision-making) may outweigh marginal gains in accuracy achieved by using a large number of features. Finally, in some contexts, it is useful to identify a small subset of features necessary for making good predictions – these "markers", *e.g.* anatomical markers in biomedical imaging or gene "biomarkers" in bioinformatics, may shed light on the underlying disease mechanism. Decision-making based on a small set of features is also highly interpretable, which is important for explaining how decisions are reached.

There are several approaches for avoiding model overfitting/COD. One is to fit the original high-dimensional data (with M features) using simple models, *e.g.* naive Bayes models (20; 21). Another is to limit the amount of model training, *e.g.* via regularization or early stopping (20). SVMs attempt to avoid overfitting by finding a discriminant function that maximizes the *margin*, *i.e.* the minimum distance of any sample point to the decision boundary. For a linear SVM, the number of free parameters is upper-bounded by the number of training samples (a subset of which are support vectors at margin distance to the hyperplane), rather than controlled by the feature dimensionality. However, SVMs are *not* immune to the curse of dimensionality (39). Thus, *feature selection*, wherein only a small subset of the original features are retained, or *feature compaction*, wherein linear or nonlinear transformations map the original features to a new, smaller coordinate space, are often essential for achieving good, *generalizeable* classification accuracy.

Consider "backward" recursive feature elimination which starts from a full space (of features) and removes features. For SVMs, for the particular case where $\phi(\cdot)$ is either finite-dimensional (e.g. as in the polynomial kernel case; as opposed to the Gaussian kernel case

where $\phi(\cdot)$ is infinite-dimensional) or there is no $\phi(\cdot)$ used (i.e. the SVM linear case), suppose that the decision boundary (hyperplane) upon feature elimination is chosen to be simply the pre-elimination boundary but with the eliminated coordinates removed (i.e. the boundary after coordinate removal is not altered). In this case, in Theorem 1 (Sec. 1.4), relative to their pre-elimination values R and A are guaranteed to not be larger, and will in fact be smaller (unless the samples or weight vector had a value of 0 at eliminated coordinates), which means that the post-elimination boundary will have a tighter bound (the new R^2A^2+1) for VC dimension h, and thus quite possibly generalization performance not significantly degraded by feature elimination. Thus, since SVMs (i.e. pre-elimination, *trained* SVM classifiers) are known to generalize well in a variety of domains, for the task of feature elimination the use of SVMs in the above fashion would be expected to achieve good generalization accuracy with the retained features (but perhaps only up to a point, at which it no longer becomes possible to remove features without significantly affecting the accuracy). In particular, one can simply aim for as small a post-elimination bound A on $||w||$ as possible (among *canonical* hyperplane candidates) – following our earlier discussion (Sec. 1.3) on the relation between the basic SVM optimization formulation (1.6) and margin maximization for *canonical* hyperplanes, it is easy to see, for SVMs, that eliminating to preserve largest post-elimination margin is consistent with attempting to tighten the post-elimination bound on h (so as to aim for good post-elimination generalization accuracy). In summary, not only there is a clear theoretical connection between SRM and SVMs (as discussed in Sec. 1.4) but also between SRM and margin-maximization-based feature selection with SVMs. The above discussion segues into the statement of the problem that this thesis is concerned with solving (stated below) – shortly afterwards we will elaborate on elimination by margin maximization.

The problem that this thesis is concerned with solving is the problem of finding the minimum feature subset needed to achieve good generalization accuracy. This will be practically addressed in this thesis via "backward" recursive feature elimination methods which start from a full space (of features) and remove features, and the main contribution of this thesis is the development of such methods with superior generalization accuracy compared to past methods such as the widely used Recursive Feature Elimination (RFE) method (37), to be discussed in Sec. 1.6.[6] Feature elimination is clearly feature selection, as opposed to feature compaction which this thesis does not focus on since it loses physical interpretation of the original features (undesirable when aiming to identify feature "markers"). More specifically, in the MRI brain image domain, for analysis of neurodegenerative diseases, in particular Alzheimer's disease, the thesis aims to find anatomical biomarkers (features that do possess a physical interpretation) that achieve good generalization accuracy, as well as aims to aid *early diagnosis* both with and without employing feature selection. Very significant as a first step in the search for a minimum feature subset for good generalization accuracy is domain knowledge (when available) – as such, the thesis has a substantial focus on choosing the original feature space judiciously, via identifying and employing particular MRI brain image analysis methods (and associated MRI-based image types) that produce feature spaces suitable for the problem. In our analysis of AD and its onset in this thesis, in Ch. 4, the features that we utilize are features that are directly associated with specific spatial locations in the brain and furthermore the class definitions involved in all of our SVM classifiers are straightforward and easily interpretable, and thus the results of our AD analyses are easily interpretable.

On the topic of simple rules aiding interpretability of a classifier's solution for biomedical or biological insight, one challenging domain is the domain of gene expression data – although a classifier in this domain may provide good accuracy, interpreting the results for biological insight may be difficult without readily interpretable rules extracted from the results. A notable approach that addresses this challenge is the k-TSP (k-Top Scoring Pairs) classifier (34), an

[6] A list of the main contributions of this thesis is given in Sec. 1.7

extension of TSP (33). It generates simple and easily interpretable rules, involving a small number of gene-to-gene comparisons, that have been shown to be accurate (34) – in binary and multiclass classification experiments, TSP and k-TSP have performed approximately the same as SVM on 19 gene expression data sets involving human cancers (34).

There are different categories of feature selection methods, including the "backward" category that our methods and the abovementioned RFE method (to be discussed in Sec. 1.6) belong to. Given an initial set of M features, unfortunately there are $2^M - 1$ possible feature subsets, with exhaustive search practically prohibited even for modest M, let alone M in the thousands. Practical feature selection techniques are thus heuristic. There are a variety of methods, exercising a large range of tradeoffs between accuracy and complexity (38). "Front-end" (or "filter") methods select features prior to classifier training, based on evaluation of discrimination power for individual features or small feature groups. "Wrapper" methods use classifier training repeatedly to evaluate the classification accuracy of numerous candidate feature subsets. There are also "embedded" methods, e.g. SVM training with a regularization penalty to suppress irrelevant features.

Front-end methods: Eliminating features in this way, i.e. via discrimination power measures *prior to* (and independent of) the subsequent step of learning with the retained features, is robust to overfitting – however, it is well-established that the retained feature set may fail to achieve sufficiently good generalization accuracy. As a first, basic example, a feature completely useless by itself for class separation (i.e. having completely overlapping class conditional densities) can significantly improve generalization accuracy when taken with other features, and two useless features can be useful together, as discussed with examples in (38). As a similar, second example, although the PCA (Principal Components Analysis) front-end method is generally successful in reducing high-d data to a low-d *data representation* via SVD-based projection (which makes the method useful e.g. in data compression), the method does not perform very well in *data classification*, essentially because PCA solely focuses on finding directions in a lower-d subspace, not on class separation. A different front-end approach is to eliminate features based on one or more feature ranking criteria, such as:

1. the (estimate of) the signal-to-noise ratio $SNR_m \equiv \frac{\hat{\mu}_m(-1)-\hat{\mu}_m(+1)}{\hat{\sigma}_m(-1)+\hat{\sigma}_m(+1)}$[7] for feature m, with the two classes being $+1$ and -1, or
2. the (estimate of) the Pearson correlation coefficient e.g. between feature and class variable. PCC can only capture *linear* dependencies between its two variables.

Feature ranking criteria that are able to capture nonlinear dependencies include:

1. the (estimate of) the Mutual-Information-based I_m measure:

$$I_m \equiv \hat{I}(X,Y) = \sum_n \sum_{c\in\{\pm 1\}} \hat{p}_{X,Y}(x_{n,m},c) \log \frac{\hat{p}_{X,Y}(x_{n,m},c)}{\hat{p}_X(x_{n,m})\hat{p}_Y(c)}.$$[8]

Wrapper methods: For wrapper methods, there is greedy forward selection, with "informative" features added, backward feature elimination, which starts from the full space and

[7] For feature m, $\hat{\mu}_m(c)$ is the estimate $\frac{1}{N_c}\sum_{n=1}^{N_c} x_{n,m}$ of the mean (for class c, where N_c is the number of samples in class c), and $\hat{\sigma}_m(c)$ is the estimate $(\frac{1}{N_c-1}\sum_{n=1}^{N_c}(x_{n,m}-\hat{\mu}_m(c))^2)^{1/2}$ of the standard deviation.

[8] The estimates $\hat{p}_{X,Y}(x_{n,m},c)$, $\hat{p}_X(x_{n,m})$, and $\hat{p}_Y(c)$ can be computed from frequency counts – for the first two, the estimation is more difficult for *continuous* X.

removes features, and more complex bidirectional searches such as simulated annealing (38). Backward search starts by assessing joint predictive power of all the features. In principle, one would like to retrain the classifier in conjunction with each backward elimination step that removes a feature subset (optimizing the classifier for the new feature space). However, considering large M and assuming one feature eliminated per step, this requires either M classifier retrainings (if retraining is done after a feature elimination) or $\frac{M(M-1)}{2}$ (if retraining is done after *trial-elimination* of every remaining feature). For SVM-based classifier training, even with recent advances that significantly reduce training time (42), (45), it may not be practically feasible to retrain at each step for M in the tens or hundreds of thousands. Thus, for large M, retraining may only be done periodically, after a "batch" of feature eliminations.

Embedded methods: Finally, before we discuss feature selection for SVMs in the next section, examples for embedded methods in general include those that estimate how a cost function (objective function) will change with movements in a feature subset space – often performed in a greedy (backward or foreword) framework, nested subsets of features can be attained in this way (38). In SVM classification in particular, some embedded methods involve the use of the ℓ_1-norm (instead of the ℓ_2-norm, such as used in equation (1.7)) as well as the "bet on sparsity" principle for high-d data (67) – according to this principle, if the number of features is much larger than the number of samples and the true "weights" associated with the features are Gaussian, neither ℓ_1 nor ℓ_2 will estimate the weights well, due to the data being too little for estimating these nonzero weights due to COD, but solution sparsity can nevertheless be *encouraged* via use of ℓ_1 (whereby numerous coordinates of the weight vector will be driven to zero (or near zero) during the learning algorithm itself, a type of built-in feature selection).

In clustering, for "unsupervised" feature selection, the focus is solely on locality-based separation of training samples (as opposed to separation of *classes*), and thus a main disadvantage is that often there are multiple good ways to cluster the data but *only one*, unknown way that can *also* achieve class-based separation (and good generalization accuracy). Irrelevant or redundant features, especially in high-dimensional data (which are adversely affected by COD), give rise to multiple good ways to cluster – as a toy example, while two well-formed clusters of single-feature (1d) samples will remain separated upon introducing a second, irrelevant feature (with a wide range of irrelevant values), the clusters that the clustering algorithm finds may be different from the above two clusters, e.g. new clusters may have formed due to a gap in values of the irrelevant (second) feature ((48) p.8).

As noted earlier, SVMs have become nearly a standard technique in many domains. There are many reasons, a number of which were discussed in Sec. 1.4. For our aim of practically addressing the problem of finding the minimum feature subset needed to achieve good generalization accuracy, we focus on SVMs and the wrapper setting. For further justification of our approach to the problem, the next section discusses feature selection for SVMs in particular, and the motivation for our methods based on margin maximization.

1.6 Feature selection for SVMs

Front-end feature selection, which, again, alone may fail to select features with sufficiently good generalization accuracy, has been applied for SVMs in numerous prior works, *e.g.* (52),(53). Wrapper-based selection, which, unlike front-end selection, employs a learning algorithm applied (once or repeatedly), has also been applied for SVM-based learning in many prior works. (72) reduced wrapper complexity by replacing the SVM training objective with an upper bound that is less complex to optimize. A widely used method analyzed in this thesis, discussed shortly, is Recursive Feature Elimination (RFE) (37), wherein at each step one removes the feature with least weight magnitude in the SVM solution. This method is computationally very lightweight and thus easily scales to large M. In (55), a wrapper approach was used, with SVM retraining performed after RFE removed a batch of features.

Embedded feature selection methods for SVMs, e.g. (73) and (30), re-formulate SVM training to encourage feature sparsity in the solution. Different norms and optimization approaches have been investigated, e.g. (78), (30), (49), (8). (8) is based on the same standard SVM formulation used in this thesis but builds in feature selection by imposing an upper bound on the number of non-zero weights as an additional constraint on the optimization problem. (78) is based on the Lasso (67) and uses the ℓ_1 norm, instead of the ℓ_2 norm, which encourages feature sparsity (i.e., the "bet on sparsity" principle mentioned in Sec. 1.5) . (30) is based on a hybrid $\ell_1 - \ell_2$ norm minimization, solved by linear programming – this LP-SVM method, which uses Newton-type iterations, is called NLPSVM.

In this thesis, we compare performance of our methods with a representative embedded method – NLPSVM method from (30). Our main focus, however, is wrapper-based feature elimination for linear and nonlinear kernel-based classifiers (including SVMs). In (37), the authors essentially argue that the RFE objective for linear SVMs is consistent with the SVM objective of margin maximization. They note that the SVM primal problem poses minimization of the squared weight vector 2-norm subject to (margin-related) constraints on each training point. Eliminating the feature with smallest weight magnitude has the least effect on the squared weight vector 2-norm and, thus, (37) argues, on the SVM solution. In summary, RFE solely focuses on minimally reducing the norm of the weight vector norm given that this is the weight vector that achieved margin maximization *pre*-elimination. In this thesis, however, we show experimentally that RFE is not in general in close agreement with margin maximization. The reason is that RFE ignores the margin constraints, focusing solely on minimally reducing the squared weight vector 2-norm. In this work, we first develop a method that explicitly performs *margin-based* backward feature elimination (MFE) for linear SVMs. We then consider nonlinear kernels. We show that RFE defined for the kernel case (37) assumes the squared weight vector 2-norm is strictly decreasing as features are eliminated. We demonstrate experimentally that this assumption is not valid for the Gaussian kernel and that, consequently, RFE may give poor results in this case. MFE for the nonlinear kernel case experimentally gives both better margin and generalization accuracy. We then present an MFE extension which stepwisely (greedily) achieves further gains in margin at small additional computational cost. This extension solves an SVM optimization problem to maximize the classifier's margin at each feature elimination step, albeit in a very lightweight fashion by optimizing only over a small set of parameters, very similar to a method suggested in (31). Finally, we develop an MFE extension that allows margin slackness.

While embedded methods do give a potential alternative to our (wrapper-based) approach, previous studies such as (49), (8), (78), (30) have not demonstrated superior performance compared to wrapper approaches such as RFE. Here, we compare MFE with both RFE and the embedded method proposed in (30).

Section 1.3 gave a brief review of SVMs. Section 2.1 discusses limitations of RFE. Sections 2.2 and 2.3 develop our methods, for the *two-class* problem. Section 2.4 gives experimental comparisons. Section 2.5 gives the chapter conclusions. In Ch. 3, we extend our MFE-based methods by formulating and evaluating them for the "multiclass" case (i.e., the case of more than two classes), and show that they perform well, including in comparison with RFE-based multiclass feature elimination. Lastly in Ch. 4, we apply SVM classification and MFE to the analysis of Alzheimer's disease (AD) and its onset, using MRI brain image processing.

1.7 Contributions of this thesis

The main contributions of this thesis are: i) identifying some difficulties with the well-known RFE method, especially in the kernel case; ii) development of novel, margin-based feature selection methods for linear and kernel-based discriminant functions for support vector machines (SVMs) addressing separable and nonseparable contexts; iii) an objective experimental comparison of several feature selection methods, which also evaluates consistency between a classifier's

margin and its generalization accuracy; iv) extending our development of novel, margin-based feature selection methods for linear and kernel-based discriminant functions from the two-class case to the "multiclass" case (i.e., more than two classes); v) experimentally evaluating our several multiclass MFE-based methods, including comparisons with multiclass RFE; vi) applying SVM classification and MFE to MRI brain image data for the analysis of Alzheimer's disease (AD) and its onset, addressing both the diagnosis aim (based on data from a (previously unseen) person's single visit) and the aim of selecting brain regions as disease "biomarkers"; vii) selecting and utilizing leading-edge MRI brain image processing tools in a pipeline fashion to generate output image types particularly suitable for detecting (and encoding) atrophy for Alzheimer's disease; viii) identifying and remedying some basic MRI image processing problems caused by some limitations of these external tools, e.g. introducing our basic *fv (fill ventricle)* algorithm for ventricle segmentation of cerebrospinal fluid (CSF).

Chapter 2

Margin-maximizing feature selection methods for Support Vector Machines: two-class case

2.1 RFE and limitations of RFE

2.1.1 RFE and limitations of RFE: linear kernel case

RFE is a stepwise (greedy), backward feature elimination technique for SVMs. Under RFE (37), the index m^* of the first feature to be eliminated is

$$m^* = \arg \min_{m \in \{1,\ldots,M\}} |w_m|, \tag{2.1}$$

and, more generally, at step i, this same selection rule is applied to the $M-i$ remaining features, with SVM retraining (optionally) applied after a batch of features is eliminated in this way. While (37) does suggest a close tie between the RFE choice and the SVM objective (margin maximization), RFE is equivalent to margin-maximizing feature elimination if and only if (2.2) below is always satisfied, with RFE's margin on the right and the margin achieved by an approach which *explicitly* eliminates the feature that preserves maximum margin on the left:

$$\max_m \min_n \frac{y_n f(\underline{x}_n) - y_n x_{n,m} w_m}{\sqrt{||\underline{w}||^2 - w_m^2}} = \min_n \frac{y_n f(\underline{x}_n) - y_n x_{n,m^*} w_{m^*}}{\sqrt{||\underline{w}||^2 - w_{m^*}^2}}. \tag{2.2}$$

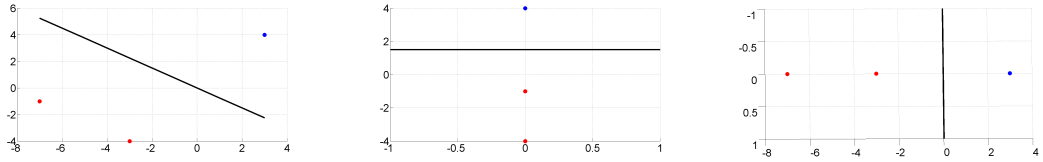
In Fig. 2.1, we prove via a simple 2-d counterexample that eliminating features according to RFE is not equivalent to preserving maximum margin. In general, direct margin maximization, achieved by the margin-based feature elimination method (MFE) we develop in Sec. 2.2, leads to significant gains in margin and may also lead to improved generalization accuracy over RFE, as we demonstrate in the sequel.

2.1.2 RFE and limitations of RFE: nonlinear kernel case

For the case of a nonlinear kernel, a natural extension of RFE was proposed in (37). Of particular interest, for the discussion that follows, is the choice of the Gaussian kernel $K(\underline{u}, \underline{v}) = exp(-\beta||\underline{u} - \underline{v}||^2), \beta > 0$. In (37), it was proposed to evaluate the square of the weight vector 2-norm (1.5) both before and after a candidate feature elimination and, thus, at the i-th stage of feature elimination, to remove the feature that minimizes the difference:

$$\Delta||\underline{w}||^2 = (||\underline{w}||^2)^{i-1,m_{i-1}^*} - (||\underline{w}||^2)^{i,m_i^*}. \tag{2.3}$$

This criterion is the natural extension of the linear RFE criterion and is consistent with the objective of reducing the square of the weight vector 2-norm the least, *assuming that the square of the weight vector 2-norm is in fact monotonically decreasing as the feature dimensionality is reduced.* For example, in the case of the polynomial kernel, $K(\underline{u}, \underline{v}) = (1 + \underline{u}^{\mathrm{T}}\underline{v})^d$, the kernel's mapping function maps an original feature vector $\underline{u}$ to a new, finite-dimensional feature vector $\underline{\phi}(\underline{u})$ whose coordinates $\phi_i(\underline{u})$ are products, raised to powers, of the original feature coordinates. Thus, it is clear for the polynomial case that eliminating an original feature coordinate zeroes out one or more coordinates of $\underline{\phi}(\underline{u})$ while leaving all others unchanged. This *effects* zeroing (removing) the



(a) (b) (c)

Fig. 2.1. When an SVM is trained on the three points in Fig. 2.1(a) ($\underline{x}_1 = (3, 4)$ in class 1, $\underline{x}_2 = (-7, -1)$ and $\underline{x}_3 = (-3, -4)$ in class 2), the decision boundary is $(\underline{w}, b) = ((0.12, 0.16), 0)$ (line shown through origin with slope $-\frac{3}{4}$), with a margin of 5 (to all three points) – neither a vertical separator line through the origin nor a horizontal separator line through 1.5 is the boundary since these achieve only a margin of 3 and 2.5, respectively. Since $w_1 = 0.12 < 0.16 = w_2$, RFE will eliminate the first feature and a threshold of the second feature at zero will become the boundary, resulting in a margin of 1 (distance from $x_{2,2} = -1$ to zero). If instead the second feature is eliminated, a (larger) margin of 3 (distance from $x_{1,1} = 3$ to the boundary at zero) would be obtained. This proves that eliminating features according to RFE is not in general equivalent to eliminating to preserve maximum margin. The latter choice is made by the margin-based feature elimination (MFE) method proposed in this thesis. Fig. 2.1(b) and Fig. 2.1(c) also respectively indicate for the two elimination options that, if a new SVM training were performed in the resulting 1-d space, then the SVM trained after the MFE feature elimination step, rather than RFE, would again have the larger margin ($3 > 2.5$).

associated scalar weights. Thus, the squared weight vector 2-norm *is* monotonically decreasing as original features are eliminated. However, we have also considered the Gaussian kernel. Since it is not so easy to analytically evaluate the Gaussian case (14), we have simply measured the squared weight vector 2-norm experimentally and have found it is *neither* monotonically decreasing *nor* monotonically increasing with feature eliminations. Consider the consequences for the RFE objective (2.3): the RFE-optimal feature elimination (assuming some eliminations increase the weight vector 2-norm) will in fact be the feature whose removal *increases* the squared weight vector 2-norm the most – this is the choice that will decrease (2.3) as much as possible (only, in this case, making $\Delta||\underline{w}||^2$ *negative*).

In Fig. 2.2, on the *UCI arrhythmia* data set, for one "trial"[1] using the Gaussian kernel, we evaluated both RFE and a modified method we dub RFE-abs which eliminates the feature that results in the smallest *change* (either decrease or increase) in the weight vector squared norm (This method is based on (46)). Note that for standard RFE there is initially a significant rise in the squared weight vector 2-norm as features are eliminated, over a range of feature eliminations (i.e., from $225-279$ features retained). Over this range, both the margin and test set error rate of standard RFE are worse than those of RFE-abs. (We give results for RFE and RFE-abs for more UCI data sets in Sec. 2.4.) We further note, however, that RFE-abs is *itself* quite suboptimal with respect to classifier margin – standard RFE becomes nonseparable (with margin going to zero) when 250 features remain, with RFE-abs becoming nonseparable soon afterwards, when 225 features remain. Also shown in Fig. 2.2 are results for a kernel-based version of the MFE method, which achieves both greater margin and lower test set error rates, compared with the RFE methods. This MFE approach is developed in the next section.

2.2 MFE: direct margin-based feature elimination

2.2.1 MFE for the linear kernel case

Since maximizing margin is the (theoretically motivated) goal of SVM training (14), it is expected that eliminating features to preserve the largest (positive) margin should yield classifier solutions that generalize better than solutions with smaller margin. The example in Fig. 2.1 illustrates both maximum margin-preserving feature elimination and the fact that RFE does not in general achieve this elimination. Surprisingly, while there are some related approaches (46)[2], we had not seen direct, margin-based feature elimination (MFE) previously proposed. In recent work (4; 5) we developed just such a technique.

Our MFE method works from a current classifier that is a separator of the training set and, at each step, eliminates the feature m_{MFE} that preserves the largest (positive) training set margin (achieved by sample n_{MFE}), *i.e.*

$$(m_{MFE}, n_{MFE}) = \arg \max_{m \in S \equiv \{m' | y_{n'} f(\underline{x}_{n'}) - y_{n'} x_{n',m'} w_{m'} > 0, \forall n'\}} \min_n \frac{y_n f(\underline{x}_n) - y_n x_{n,m} w_m}{\sqrt{||\underline{w}||^2 - w_m^2}}, \tag{2.4}$$

[1] A "trial", defined in Sec. 2.4.1 and summarized here, is a random $50-50\%$ split of the data set into a held-out set and a non-heldout set. For the trial, we train the SVM on a random 90% subset of the non-heldout set, and evaluate on the remaining 10% (validation set). SVM hyperparameters are then chosen to minimize the average validation error, measured over five such training/validation splits. We then train an SVM on the entire non-heldout set to obtain the initial classifier used both by our methods and RFE. Feature elimination is then performed, with margin measured on this (non-heldout) set and error rate measured on the held-out (test) set.

[2] This reference eliminates features to maximize the average distance to the hyperplane, over all training points, rather than to maximize margin.

with S the set of valid candidate features, whose single eliminations will preserve positive margin. Elimination based on (2.4) is illustrated in the example in Fig. 2.1.

We emphasize that feature selection is most urgently needed when M is very high, and that N can often be *quite* small, such as in medical imaging and bioinformatics (*e.g.* gene microarray) contexts, and, citing (12) in Sec. 1.3, we conveyed it is highly probable separability can be achieved by a linear classifier in such cases. Even for some intermediate-to-low dimensional domains (e.g. UC Irvine data), we will demonstrate that the training set is both initially separable and remains separable while a significant number of features are eliminated (with margin used as the feature elimination criterion). Thus, we argue that the set S in (2.4) is non-empty in many practical cases and, thus, using (positive) margin as the elimination criterion is feasible in practice, especially for high-dimensional domains, where feature selection is also most urgently needed.

On the other hand, to handle the case where the training set is *nonseparable*, we propose (in Sec. 2.3) an extension of MFE that allows for margin slackness and nonseparability.

We next give pseudocode for our basic MFE method.

2.2.1.1 MFE algorithm pseudocode for SVMs: linear kernel case

Notation: $q^{i,m} \equiv$ quantity q at feature elimination step i upon elimination of feature m.

0. *Preprocessing:* Let $\mathcal{M}$ be the set of eliminated features, with $\mathcal{M} = \emptyset$ initially. First run SVM training on the full space to find a separating hyperplane $f(\underline{x}) = 0$ (with f parameterized by $\underline{w}, b$), with weight norm-squared $L^{-1,0} \equiv ||\underline{w}||^2$, where $i = -1$ means before eliminating any features and $m_{-1} = 0$ is a dummy placeholder index value. For each feature m, compute $\delta_n^m = y_n x_{n,m} w_m \ \forall n$. Recall that $g(\underline{x}_n) \equiv y_n f(\underline{x}_n)$ so that δ_n^m is the Δg quantity $\delta_n^{j,m} \equiv (g_n^{j-1,m_{j-1}} - g_n^{j,m})$ whose value is the same at every elimination step for a given (m,n) pair. Compute $g_n^{-1,0} = y_n b + \sum_{m=1}^{M} \delta_n^m \ \forall n$. Set $i \leftarrow 0$. At elimination step i, perform the following operations:

1. For each $m \notin \mathcal{M}$, using recursion, compute $g_n^{i,m} = g_n^{i-1,m_{i-1}} - \delta_n^m \ \forall n$, determine $\mathcal{N}^{i,m} = \min_n g_n^{i,m}$. Determine the candidate feature set $S(i) = \{m \notin \mathcal{M} \mid \mathcal{N}^{i,m} \geq 0\}$. Note that δ_n^m need not be computed in this step if stored for all m and n during preprocessing (step 0). If $S(i)$ is empty (the data is nonseparable) then **stop**.

2. For $m \in S(i)$, using recursion, compute $L^{i,m} = L^{i-1,m_{i-1}} - w_m^2$, determine $\gamma^{i,m} = \max_{m \in S(i)} \frac{\mathcal{N}^{i,m}}{\sqrt{L^{i,m}}}$.

3.1. Eliminate feature $m_i \equiv \arg \max_{m \in S(i)} \gamma^{i,m}$, *i.e.* $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

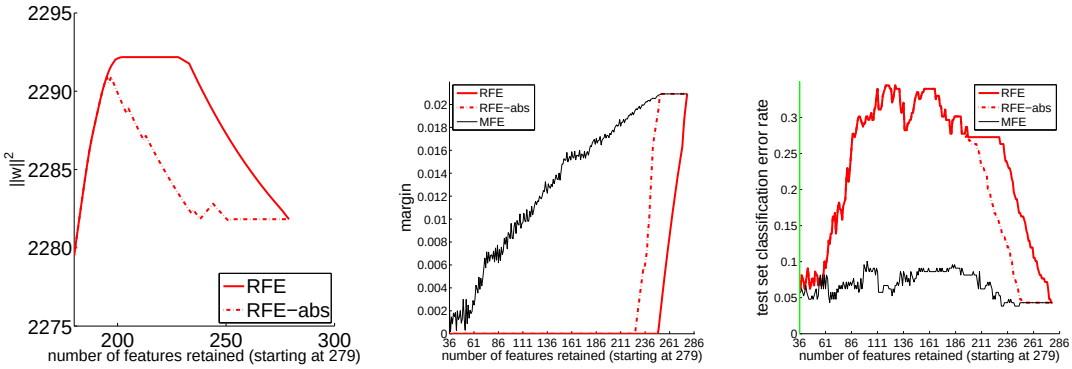
3.2. Keep for the next iteration only the recursive quantities $\{g_n^{i,m_i} \ \forall n\}$, L^{i,m_i}, associated with the eliminated feature.

3.3. $i \rightarrow i + 1$ and go to step 1.

In Fig. 2.3(a), we demonstrate that MFE achieves both larger margins and better overall generalization performance (test set error rate) than RFE on the gene microarray *Leukemia* data set.[3] For this data set, the average number of retained features at which separability was lost

[3]The curves in Fig. 2.3(a) are the result of averaging over 10 "trials" – margin measured on a trial's non-heldout set and error rate measured on the trial's held-out (test) set are averaged over 10 trials.

under MFE was 90, and thus the curves are shown for the range of 90-7129 features retained. Similar results are achieved on other microarray data sets, as well as data sets from the UC Irvine repository. More extensive evaluations are given in Sec. 2.4.



(a) $||\underline{w}||^2$ (b) margin (c) test error rate

Fig. 2.2. Results for the Gaussian kernel on the *UCI arrhythmia* data set with 279 features.

2.2.2 MFE for the nonlinear kernel case

To address the suboptimality of RFE (and RFE-abs) described in Sec. 2.1.2, similar to the pseudocode in Sec. 2.2.1.1 we propose a recursively-implemented margin-optimizing feature elimination (MFE) algorithm, now for *kernel-based* SVMs. In this case, the recursion is on the kernel computation. For example, for the Gaussian kernel, denoting $\mathcal{K}_{k,n}^{i,m} \equiv K(\underline{s}_k^{i,m}, \underline{x}_n^{i,m})$ at elimination step i, we have the recursion:

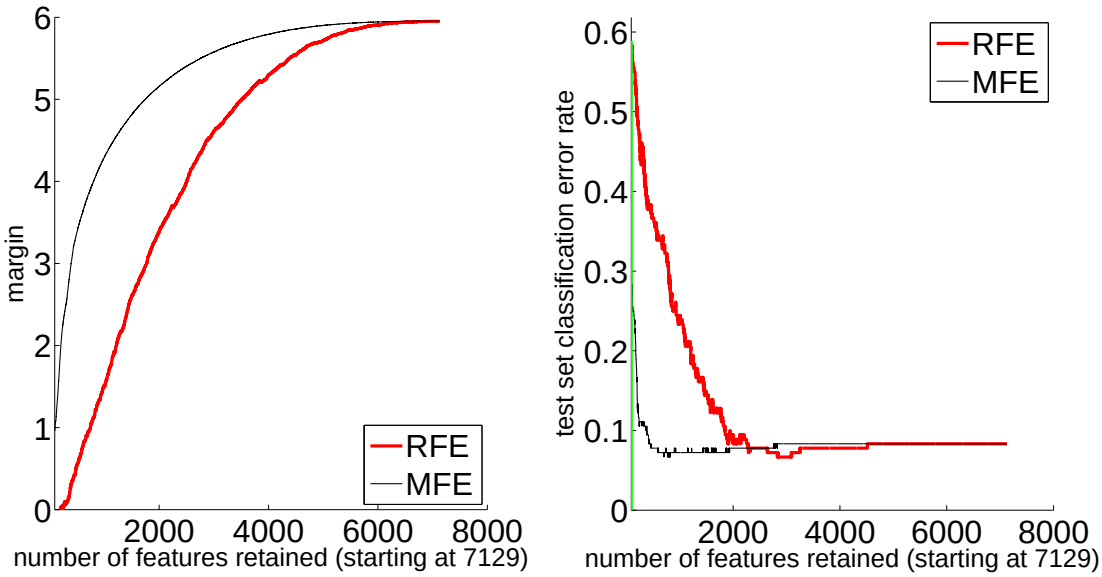
$$\mathcal{K}_{k,n}^{i,m} = \mathcal{K}_{k,n}^{i-1,m_{i-1}} \exp(\beta(s_{k,m} - x_{n,m})^2), \forall k, \forall n \tag{2.5}$$

Likewise, for the polynomial kernel $K(\underline{u}, \underline{v}) = (\beta \underline{u}^{\mathrm{T}} \underline{v} + 1)^d \equiv (H(\underline{u}, \underline{v}))^d$, and denoting $\mathcal{H}_{k,n}^{i,m} \equiv H(\underline{s}_k^{i,m}, \underline{x}_n^{i,m})$ at elimination step i, we have the recursion:

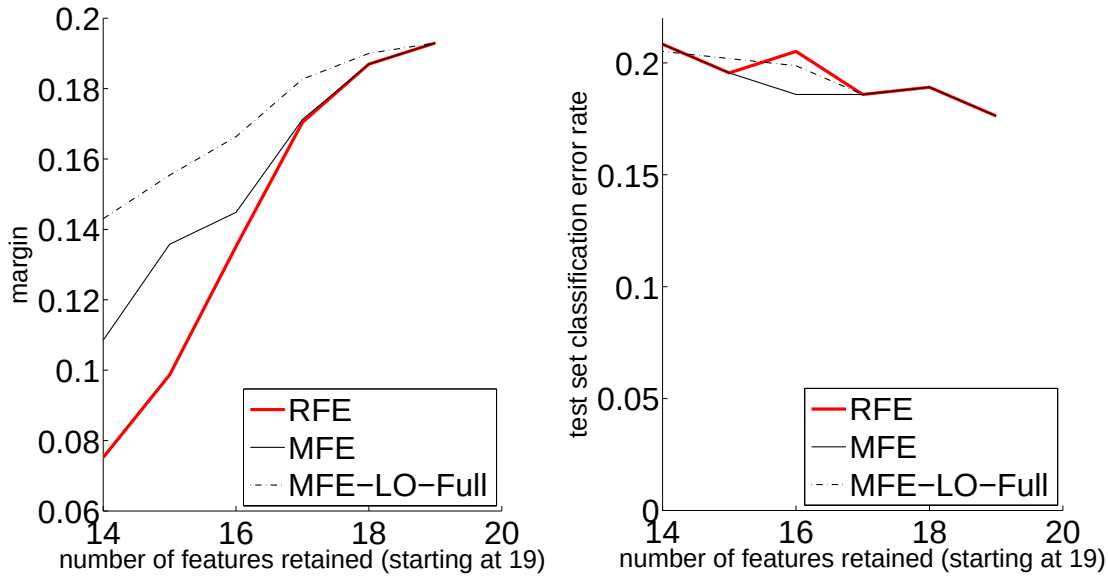
$$\mathcal{H}_{k,n}^{i,m} = \mathcal{H}_{k,n}^{i-1,m_{i-1}} - \beta s_{k,m} x_{n,m}, \forall k, \forall n \tag{2.6}$$

$$\mathcal{K}_{k,n}^{i,m} = (\mathcal{H}_{k,n}^{i,m})^d, \forall k, \forall n \tag{2.7}$$

These recursively computed kernels, which are used to evaluate both the discriminant function $f(\underline{x}_n)$ via (1.4) and the weight vector norm via (1.5), form the basis for a kernel-MFE algorithm whose pseudocode implementation is a simple modification of the linear SVM pseudocode given in Sec. 2.2.1.1. Our MFE method works from a current classifier that is a separator of the training set and, at each step i, eliminates the feature m_{MFE} (below) that



(a) Training set margin, test set error rate, for the *Leukemia* data set, with 7129 features.



(b) Training set margin, test set error rate, for the *UCI hepatitis* data set, with 19 features.

Fig. 2.3.

preserves the largest (positive) training set margin:[4]

$$(m_{MFE}, n_{MFE}) = \arg \max_{m \in S \equiv \{m' | g_{n'}^{i,m'} > 0, \forall n'\}} \min_n \frac{g_n^{i,m}}{||\underline{w}||^{i,m}}. \quad (2.8)$$

Figs. 2.2 (b) and 2.2 (c) demonstrate substantial increases in margin and generalization performance (much lower error rate) achieved by MFE for the Gaussian case over both RFE and RFE-abs on the *UCI arrhythmia* data set. The number of retained features at which separability was lost under MFE was 36, and thus the margin and test set error rate curves are shown for the range of 36-279 features retained – notice from the margin curves that the data remains separable under MFE for much longer than under RFE or RFE-abs. Again, we give more extensive results for these methods in Sec. 2.4.

2.2.3 "Little Optimization" (LO): further increases in margin

For large M, it may not be computationally practical to retrain the SVM in the reduced feature space, in conjunction with each feature elimination step. However, a *type* of classifier retraining at every step that is consistent with margin maximization and yet is exceptionally modest computationally, compared to full SVM retraining, is still possible. The idea is to solve the SVM problem but while optimizing drastically fewer parameters than the full complement of SVM feature weights. Let $(\underline{w}^{-\mathcal{M}}, b)$ denote the linear SVM weight vector (and affine parameter) after a set $\mathcal{M}$ of features are eliminated. Suppose we consider the *new* parameterized weight vector $(A\underline{w}^{-\mathcal{M}}, w_0)$, where A and w_0 are *scalar* parameters to be optimized, with $\underline{w}^{-\mathcal{M}}$ held fixed. That is, we allow adjusting the squared weight vector 2-norm and the affine parameter, with the weight vector orientation fixed. We thus pose the standard SVM training problem, but optimizing only in this *two*-dimensional parameter space: $\min_{A,w_0} A^2 \; s.t. \; y_n(A(\underline{w}^{\mathrm{T}}\underline{\phi}(\underline{x}_n)) + w_0) \geq 1, n = 1, \ldots, N$. In the linear kernel case, this problem is given below by (2.9). In the nonlinear kernel case, it is given by (2.10).

$$\min_{A,w_0} A^2 \; s.t. \; y_n(A(\underline{w}^{-\mathcal{M}^{\mathrm{T}}}\underline{x}_n^{-\mathcal{M}}) + w_0) \geq 1, n = 1, \ldots, N \quad (2.9)$$

$$\min_{A,w_0} A^2 s.t. \; y_n(A(\sum_{k \in \mathcal{S}} \lambda_{\underline{s}_k} y_{\underline{s}_k} K(\underline{s}_k^{-\mathcal{M}}, \underline{x}_n^{-\mathcal{M}})) + w_0) \geq 1, n = 1, \ldots, N \quad (2.10)$$

This problem was previously posed in (31), with the solution achieved by use of Newton iterations. Next, we develop an alternative solution that is advantageous in that it is essentially closed form, requiring very little computation. In particular, the feasible region of the problem is defined by two cones in the (A, w_0) plane (one such cone, $\mathcal{C}^+$, is shown in Fig. 2.4), with the minimum squared weight vector 2-norm (A^2) in each cone achieved at the cone's tip, which is easily found. Thus, the minimization is performed by identifying the tip of each cone and choosing the one with smaller A^2. Referring to Fig. 2.4, we prove this as follows. Denoting a slope $m_n \equiv -\underline{w}^{-\mathcal{M}^{\mathrm{T}}}\underline{x}_n^{-\mathcal{M}}$ in the SVM linear case (or $m_n \equiv -\sum_{k \in \mathcal{S}} \lambda_{\underline{s}_k} y_{\underline{s}_k} K(\underline{s}_k^{-\mathcal{M}}, \underline{x}_n^{-\mathcal{M}})$ in the nonlinear kernel case), we rewrite the constraints (2.9) (or (2.10)) as two sets of inequalities (S_1 for class 1 and S_2 for class 2) in the (A, w_0) plane: $S_1 = \{w_0 \geq m_n A + 1 \mid y_n = 1, n = 1, \ldots, N\}$, $S_2 = \{w_0 \leq m_n A - 1 \mid y_n = -1, n = 1, \ldots, N\}$. In this plane, each inequality in S_1 (one for each data point in class 1) specifies a line. Let L_1 be the set of these lines associated with S_1. Let L_2 be defined similarly. Using the figure, we can show that the feasible region of (2.9) (or (2.10)) in halfspace $A > 0$ is the cone $\mathcal{C}^+$ bounded by the line l_2^+ with maximum slope in L_2 and the line l_1^+ with minimum slope in L_1, and that the (feasible) minimum A^2 in $\mathcal{C}^+$ is at its tip,

[4] (2.8) specializes to (2.4) for the case of a linear kernel.

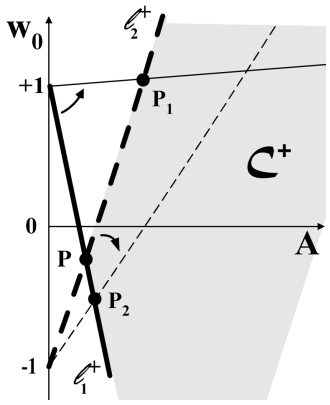


Fig. 2.4. Illustration of $\mathcal{C}^+$, one of two cones used in the solution of the "little optimization (LO)" problem.

P. Specifically, by their definitions, lines l_1^+ and l_2^+ are known. l_1^+ intersects l_2^+ at a lower point (P) along l_2^+ than any other $l_1 \in L_1$ (which all pass through $(0, 1)$ and are oriented away from l_1^+ in the counter-clockwise arrow direction shown – one such line l_1 is shown as a thin solid line in the figure). Similarly, l_2^+ intersects l_1^+ at a higher point along l_1^+ than any other $l_2 \in L_2$ (which all pass through (0,-1) – one such line l_2 is shown as a thin dashed line). Thus, in the halfspace $A > 0$ the feasible region is $\mathcal{C}^+$, and the (feasible) minimum A^2 in $\mathcal{C}^+$ is at its tip, P. Similarly, the minimum (feasible) A^2 in the other halfspace $A < 0$ is at the tip of a corresponding cone $\mathcal{C}^-$ bounded by the line l_1^- with maximum slope in L_1 and the line l_2^- with minimum slope in L_2.

The intersection point $\mathrm{P} = (A_{inter}, w_{0_{inter}})$ is computed as follows: $A_{inter} = \frac{2}{m_{max} - m_{min}}$ (where m_{max} is the slope of l_2^+ and m_{min} is the slope of l_1^+) and $w_{0_{inter}} = m_{min} A_{inter} + 1$. Further, it takes N additional multiplications and additions to scale $f(\underline{x}_n) - w_0$ (i.e. m_n) by A_{inter} and add $w_{0_{inter}}$, creating g_n, $n = 1, \ldots, N$, for use at the next elimination step. This "little optimization" (LO) thus takes only $N + 2$ multiplications and $N + 2$ additions at each elimination step – it just requires first finding the tips of the cones $\mathcal{C}^+$ and $\mathcal{C}^-$, choosing the tip with minimum A^2, and then performing N multiplications (scalings) and adds (shifts).

Since LO requires so little computation, it can be performed in conjunction with each (margin-optimizing) feature elimination step. This can take place *after* eliminating a feature (LO-Lite), or can be embedded into the elimination decision (LO-Full)[5]. In both cases, at each step, LO is guaranteed to increase the margin that would have been achieved by basic MFE during that step. However, since feature elimination is performed within a greedy (stepwise-optimal) framework, there is no theoretical guarantee that the margin curve for LO-Lite or LO-Full will lie strictly above the margin curve for basic MFE – LO will in general alter the (greedily chosen) sequence of margin-maximizing feature eliminations. This lack of guarantee in fact applies even if *full* SVM retraining is coupled to the feature eliminations – we dub such a procedure "MFE-Retrain". In Fig. 2.3(b), we demonstrate a strict increase in margin (averaged over trials) for MFE-LO over basic MFE on the *UCI hepatitis* data set, during the first 5 feature elimination steps (when the data is still separable). In this case, MFE-LO also achieves a modest reduction in average test set error at 14 retained features. We give more detailed results of these methods in Sec. 2.4.

[5] In LO-Full, we perform candidate elimination of each feature and then perform LO in the resulting reduced space. We then pick the candidate elimination that leads to largest post-LO margin.

2.3 MFE-slack: utilizing margin slackness

Recall the two SVM training objectives given in Sec. 1.3, with (1.6) choosing the weight vector to maximize margin while strictly enforcing margin constraints, and with (1.7) allowing for some margin slackness (i.e., $\xi_n > 0$ in (1.7)), including possible misclassifications (i.e., $\xi_n > 1$). Introducing slackness allows classifier design even when the data set is nonseparable. Moreover, strictly satisfying the margin could potentially lead to overfitting when training samples at the margin are outliers or even *mislabeled* samples. Optimizing the amount of slackness (by choosing the parameter C), *e.g.* via cross validation, may yield classifiers with better generalization than those based on strictly maximizing margin. All of these reasons motivate us in this section to propose a feature elimination extension of MFE that allows for margin slackness.

Since the approach we propose uses (1.7) as the feature elimination objective, it is instructive to first discuss in more detail the objectives (1.6) and (1.7) and their relationship to margin maximization – the discussion in this paragraph is partly a repetition from Sec. 1.3. In particular, recall that, assuming we have a separator, the margin is $\gamma = \frac{\min_n g(\underline{x}_n)}{||\underline{w}||}$ and, further, note that $g(\cdot) \equiv yf(\cdot)$ can be amplitude-scaled by an arbitrary nonzero constant ρ without altering the decision boundary. In particular, if we form $\tilde{g} = \rho g$, where $\rho = \frac{1}{\min_n g(\underline{x}_n)}$, then $\gamma = \frac{\min_n g(\underline{x}_n)}{||\underline{w}||} = \frac{1}{||\underline{w}||}$. We thus see the well-known result that, for this special choice of ρ, maximizing margin is equivalent to minimizing the squared weight vector 2-norm. Further, we can define $\xi_n = \max(0, 1 - \tilde{g}(\underline{x}_n))$, *i.e.,* consistent with the constraint $\xi_n \geq 0$ in (1.7), zero slackness for correctly classified samples at greater than margin distance and positive slackness for samples at less than margin distance to the decision boundary (including possibly misclassified samples). The latter samples will be referred to as "margin violators" in the sequel. Now consider the SVM objective function in (1.7) – if C is made sufficiently large, no margin slackness will be tolerated and minimizing (1.7) *reduces to* minimizing the squared weight vector 2-norm and, thus, to maximizing margin. We thus see that (1.7) is a generalization of strict margin maximization that *specializes* to strict margin maximization when C is made sufficiently large.

Now let us relate this to feature elimination algorithms. In Sec. 2.2, at each elimination step we chose the pair (m_{MFE}, n_{MFE}) to strictly maximize margin, with m_{MFE} the eliminated feature and n_{MFE} the sample achieving the post-elimination margin. In this section, we will choose the pair (m_{MFE-S}, n_{MFE-S}) to minimize the objective (1.7), with both the weight vector 2-norm and the slackness values evaluated *post*-feature-elimination. In this discrete optimization, for each *candidate* feature for elimination, m, we must evaluate *every* (correctly classified) sample $\tilde{n}$ as the potential margin-defining sample associated with this elimination (We will dub such margin-defining samples as "anchor" samples). To do so, as discussed above, we find the value $\rho_{\tilde{n}}$ such that, post-elimination of feature m, $\rho_{\tilde{n}} g_{\tilde{n}} = 1$, and we measure the slackness values relative to this anchor sample, *i.e.* $\xi_n = \max(0, 1 - \rho_{\tilde{n}} g_n)$ $\forall n$, and we plug into the objective function (1.7). Candidate anchor samples (and associated induced slackness values) are illustrated by an example in Fig. 2.5. The pair (m_{MFE-S}, n_{MFE-S}) that, post-elimination, minimizes (1.7) over all the discrete choices $\{(m, n)\}$ is selected, which thus determines the eliminated feature, m_{MFE-S}.

From the discussion above on the objective functions (1.6) and (1.7), it should be clear that, if C is made sufficiently large (and assuming the data is separable), nonzero slackness values will again not be tolerated. In this case, choosing (m_{MFE-S}, n_{MFE-S}) to minimize (1.7) *precisely* reduces to choosing the pair to maximize margin, *i.e.* the discrete optimization problem minimizing (1.7) precisely reduces to the problem solved by the standard MFE algorithm from Sec. 2.2. In summary, the method we propose in this section is a natural extension of MFE to incorporate slackness that in fact precisely reduces to MFE for large enough C when the data is separable. We next present pseudocode for this algorithm, which we will (justifiably) refer to as "MFE-slack".

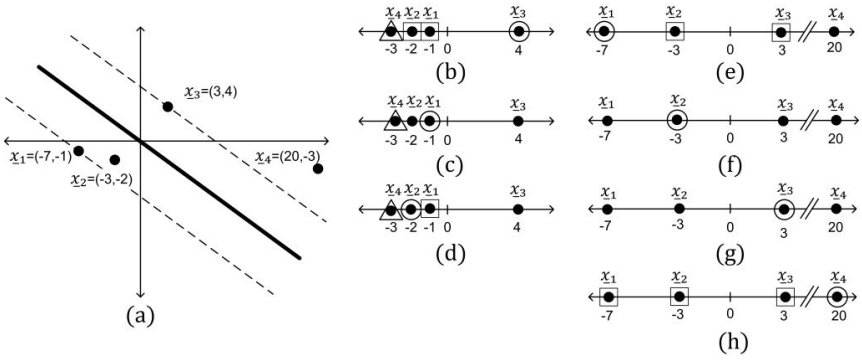


Fig. 2.5. In (a), consider the line through the origin with slope $-\frac{3}{4}$ as a decision boundary obtained by training an SVM on a sample set that includes the four shown samples: $\underline{x}_1$ and $\underline{x}_2$ in class 1, $\underline{x}_3$ and $\underline{x}_4$ in class 2. $\underline{x}_1$ and $\underline{x}_3$ are located at a (violated) margin distance of 5 (shown by dashed lines), with $\underline{x}_2$ the margin violator. For the case where m_c is the second feature, the new boundary is the origin and the candidate anchor $\underline{x}_{n_a}$ in (e)-(h), marked by a circle, is $\underline{x}_1$, $\underline{x}_2$, $\underline{x}_3$, $\underline{x}_4$, respectively. The pseudocode in Sec. 2.3.1 will describe how MFE-slack performs scaling by ρ on all samples to ensure $\xi_{n_a} = 0$ (for anchor $\underline{x}_{n_a}$). In (e)-(h), after scaling, $0 < \xi_n < 1$ for samples marked by a square (i.e., samples closer to the boundary than $\underline{x}_{n_a}$) – there are no misclassified samples (i.e., no samples with $\xi_n > 1$). The case where m_c is the first feature is illustrated in (b)-(d). Here, again after scaling, misclassified samples (with $\xi_n > 1$) are marked by a triangle. Notice there is no diagram with $\underline{x}_4$ as candidate anchor because we only consider as anchors the points that are correctly classified – under elimination of feature 1, $\underline{x}_4$ becomes misclassified.

2.3.1 MFE-slack algorithm pseudocode for linear and nonlinear kernel SVMs

Notation: $q^{i,m_c,n_a} \equiv$ quantity q at elimination step i upon elimination of candidate feature m_c when the candidate anchor is $\underline{x}_{n_a}$.

0. *Preprocessing:* Let $\mathcal{M}$ be the set of eliminated features, with $\mathcal{M} = \emptyset$ initially. First run SVM training on the full space to find the initial hyperplane $f(\underline{x}) = 0$. The hyperplane need not be a separating one; see Sec. 2.4 for our training procedure. Recalling that $g(\underline{x}_n) \equiv y_n f(\underline{x}_n)$, compute $\{g_n^{-1,0}\ \forall n\}$ and $(||\underline{w}||^2)^{-1,0}$, where $i = -1$ means before eliminating any features and $m_c = 0$ is a dummy placeholder index value. Set $i \leftarrow 0$. At elimination step i, perform the following operations:

1. For each $m_c \notin \mathcal{M}$, compute $(||\underline{w}||^2)^{i,m_c}$ using recursion as in basic MFE; for each $m_c \notin \mathcal{M}$ and $\forall n$, compute g_n^{i,m_c} using recursion as in basic MFE.[6] For each candidate feature (m_c) elimination, we consider any data point to be a *valid* candidate anchor $\underline{x}_{n_a}$ if it is not misclassified under that elimination (i.e. if $g_{n_a}^{i,m_c} \geq 0$). For all such *valid* $(\underline{x}_{n_a}, m_c)$ pairs and all candidate features m_c, perform steps 2 and 3.

2. To ensure $\xi_{n_a}^{i,m_c,n_a} = 0$ (for anchor point $\underline{x}_{n_a}$) and to meet all the constraints in (1.7), we perform scaling as follows. Compute $\tilde{g}_n^{i,m_c,n_a} = g_n^{i,m_c}\rho^{i,m_c,n_a}$ for all non-anchors $\underline{x}_n$ $(n \neq n_a,\ n = 1,\ldots,N)$, where $\rho^{i,m_c,n_a} = \frac{1}{g_{n_a}^{i,m_c}}$. Also, set $\tilde{g}_{n_a}^{i,m_c,n_a}$ to 1.

3. Compute $\xi_n^{i,m_c,n_a} = \max(0, 1 - \tilde{g}_n^{i,m_c,n_a}), n = 1,\ldots,N$.

4.1 Perform the discrete double-minimization shown below in (2.11) to determine the feature m_i to be eliminated, and then eliminate it, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.[7]

$$(m_i, n_i) = \arg \min_{m_c \in \bar{\mathcal{M}}} \min_{n_a \in \{n | g_n^{i,m_c} > 0\}} (\frac{1}{2}(||\underline{w}||^2)^{i,m_c}(\rho^{i,m_c,n_a})^2 + C \sum_{n'=1}^{N} \xi_{n'}^{i,m_c,n_a}) \qquad (2.11)$$

4.2 Since the scaling by ρ was only needed for the current elimination step, we do not carry the scaled quantities $\tilde{g}$ to the next elimination step $(i+1)$.

4.3 $i \rightarrow i+1$ and go to step 1.

2.4 Results

We performed experiments to compare our methods with RFE and NLPSVM (30).

2.4.1 Experimental procedure for the initial classifier training

We used the following common experimental procedure both for training of the *initial* classifier used by MFE and RFE and for training of the (*final*) NLPSVM classifier.

Step 1) The data set is randomly split 50-50% into a non-heldout (training) set X and a heldout (test) set $\bar{X}$. Each such split defines one "trial". *Steps 2-4* perform a bootstrap validation procedure to select classifier hyperparameters for each trial, from amongst a candidate set of hyperparameter values.

[6] As a reminder, in the nonlinear kernel case, the recursion is on the kernel computation.

[7] The simple default way to choose C in (2.11) is to set it to the C value used in (1.7) to train the initial (pre-elimination) classifier; however, a different C value may be chosen for (2.11), e.g. based on a cross-validation procedure applied after eliminating a batch of features.

Step 2) Perform five 90-10% random splits of X. For each such split, the large subset X_L will be used for training and the small subset X_S will be used for validation.

Step 3) Perform the following for each candidate for the hyperparameter values: For each bootstrap split, use X_L to train a classifier and evaluate the performance on X_S. Average the validation error rate over the five bootstraps.

Step 4) Select the hyperparameter values, from amongst the set of candidates, that minimize the average validation error rate. Then retrain the classifier for these hyperparameter values using all of X.

To achieve fair comparisons, our methods, RFE, and NLPSVM all shared precisely the same data – for every trial, the training set X, the test set $\bar{X}$, and the multiple X_L and X_S sets (for the trial's bootstrap splits) were the same for all methods. Furthermore, for every trial, our methods and RFE used the same initial classifier (defined by (1.7) and determined in *Step 4*), obtained by training on all of X.

We emphasize that the above procedure is also used for NLPSVM training. Since for NLPSVM the learned classifier is independent of the initial chosen parameters, use of the above procedure for NLPSVM means that the NLPSVM hyperparameters were chosen to minimize the validation error of the (*final, trained*) NLPSVM classifier. By contrast, for MFE and RFE the hyperparameters were only chosen to minimize validation error of the *initial* (pre-elimination) classifier (without accounting for subsequent feature eliminations). In this way, the procedure is somewhat favorably biased toward NLPSVM. While it is possible to modify our procedure to choose best hyperparameters consistent with subsequent feature eliminations for MFE and RFE, we have not done this. Despite this disadvantage, we will show that our basic MFE method typically achieved better or competitive generalization (lower *test set* error rate) compared with NLPSVM (30), in our extensive experiments.

Our set of candidate hyperparameter values for the linear kernel case (i.e. our candidate set for the C parameter) was $\{2^0, 2^1, \ldots, 2^{10}\}$. We also used this set for NLPSVM's first parameter ν (30), a penalty parameter analogous to C in (1.7). For the second NLPSVM parameter δ (30), which is added to the Hessian before matrix inversion (as part of the Newton direction computation), we used the value set from (30): $\{10^{-3}, 10^{-2}, \ldots, 10^3\}$. In this thesis, we compare with NLPSVM for the linear kernel case only. In the nonlinear kernel case, our set of candidate β parameter values was $\{2^{-30}, 2^{-29}, \ldots, 2^{10}\}$ and our C set was the same as in the linear kernel case; thus our hyperparameter grid was of dimension 11×41.

The SVM training (1.7) in all experiments was performed using the LIBSVM software (9). For NLPSVM, we used the code provided in (30), and the following values used therein for additional NLPSVM parameters: $\epsilon = 4 \times 10^{-4}, \alpha = 10^3, tol = 10^{-3}, imax = 50$.

2.4.1.1 Data pre-processing prior to initial classifier training

Categorical features Q whose values are elements of an unordered set of categories (e.g. $Q \in \mathcal{Q} = \{orange, yellow, red\}$) were mapped to $card(\mathcal{Q})$ unit vector features, each indicating one of the categories (e.g. $[0\ 0\ 1]^{\mathrm{T}}$ for *orange*, $[0\ 1\ 0]^{\mathrm{T}}$ for *yellow*, $[1\ 0\ 0]^{\mathrm{T}}$ for *red*). As a second step, we normalized all feature values to the [0,1] range (separately for X and $\bar{X}$).

2.4.2 Experimental procedure for feature elimination

Even for high-dimensional domains, we may eliminate all the way down to a few features, in which case at some point the data typically does become nonseparable. To continue the elimination process after loss of separability, we define the "hybrid" MFE/MFE-slack method, where, simply, MFE is used for the steps where the data is separable (including the initial step) and MFE-slack is used for the other steps.

In the sequel, we have broken up our experimental comparisons into three categories: *i)* high-dimensional data (Sec. 2.4.3 and 2.4.6), *ii)* low-to-intermediate dimensioned domains for which the data *is* initially separable (Sec. 2.4.4), and *iii*) nonseparable domains (Sec. 2.4.5) which require immediate use of our MFE-slack method. For a given (data set, kernel) pairing (where the kernel is linear, polynomial (of degree 3), or Gaussian), if initial classifier separability was obtained in at least 6 out of the 10 trials, we concluded that the pair most suitably fell into the separable category (i.e., Sec. 2.4.3 or Sec. 2.4.4); otherwise the pair fell into the nonseparable category (i.e., Sec. 2.4.5).

2.4.2.1 Stopping criteria

A criterion for stopping the feature elimination process should firstly have a classification error rate at the stopping point not much higher than (and, desirably, lower than) the error rate of the initial (pre-elimination) classifier. Second, in order to identify the important, class-defining feature "markers", it is desirable for the percentage of features eliminated (at the stopping point) to be large[8]. We defined and evaluated the following stopping criterion:

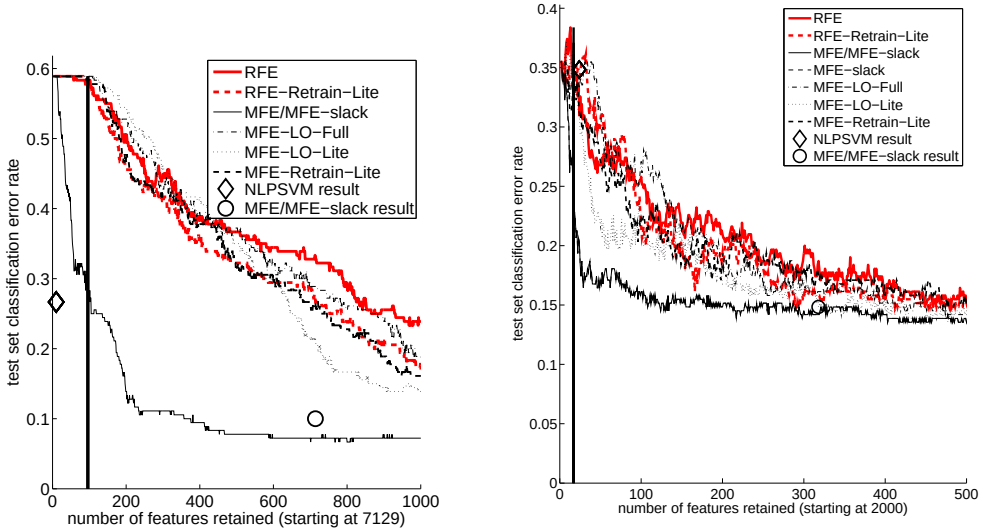
Acceleration-based (*accel*): The criterion is a balance between margin (reward) and the number of remaining features (cost). As we eliminate features, the reward-to-cost ratio increases with the margin staying relatively flat, but at some point with few remaining features the ratio will decrease (or error rate increase). Here we aim to stop. With such a stopping criterion goal, MFE is a better iteratively greedy approach than RFE – MFE also gives better generalization performance than RFE, as our results discussion will demonstrate. More specifically, to robustly detect the "knee" in the validation-set error rate curve for MFE-based methods as features are eliminated, we used the following algorithm to determine a stopping point.

1. On the error sequence $\underline{e} = \{e_i, i = 1, \ldots, M\}$, where e_i is the validation-set error rate after eliminating i features, perform first-order autoregressive smoothing with parameter $\frac{1}{2}$.[9]

2. Compute associated velocity $\underline{v}$ and acceleration $\underline{a}$ sequences: $v_i = e_i - e_{i-1}$ and $a_i = v_i - v_{i-1}, i = 1, \ldots, M$. Compute the running average sequence $\underline{\hat{a}}$ of $\underline{a}$ and the sequence $\underline{d}$ of percent increases in consecutive $\hat{a}_i$ values.

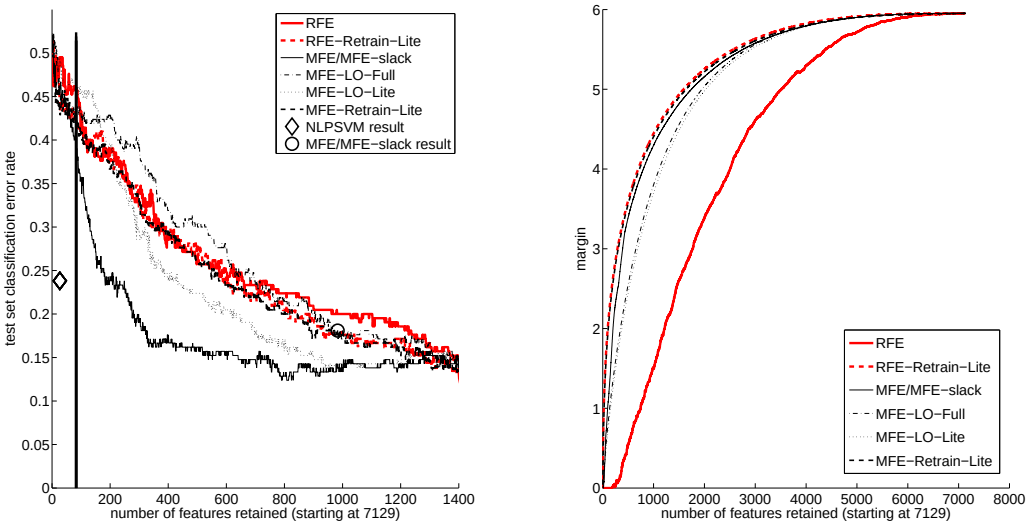
3. Set the percentage threshold t to 200%. Find the first point $\tilde{d}_i$ along the sequence $\underline{d}$ that exceeds t – if such a point does not exist, lower t by 20% and try again. This high-to-low approach avoids stopping before the "knee" in the curve (to avoid false-positive knee detection). Then, to avoid overshooting the knee, based on the elimination sequence, restore $0.01M$ eliminated features.

For the NLPSVM method, we used the stopping criterion from (30): stop when either of two conditions is satisfied: 1) $\|\underline{u}^i - \underline{u}^{i+1}\| \leq tol$, where $\underline{u}^i$ represents an (intermediate) solution for the LP problem at the i-th iteration and tol is a user-selected tolerance value, specified in Sec. 2.4.1. 2) User-selected maximum number of iterations ($imax$, specified in Sec. 2.4.1) is reached.



(a) *Leukemia* data set with 7129 features; linear kernel. The vertical line indicates where the basic MFE method lost separability (on average).

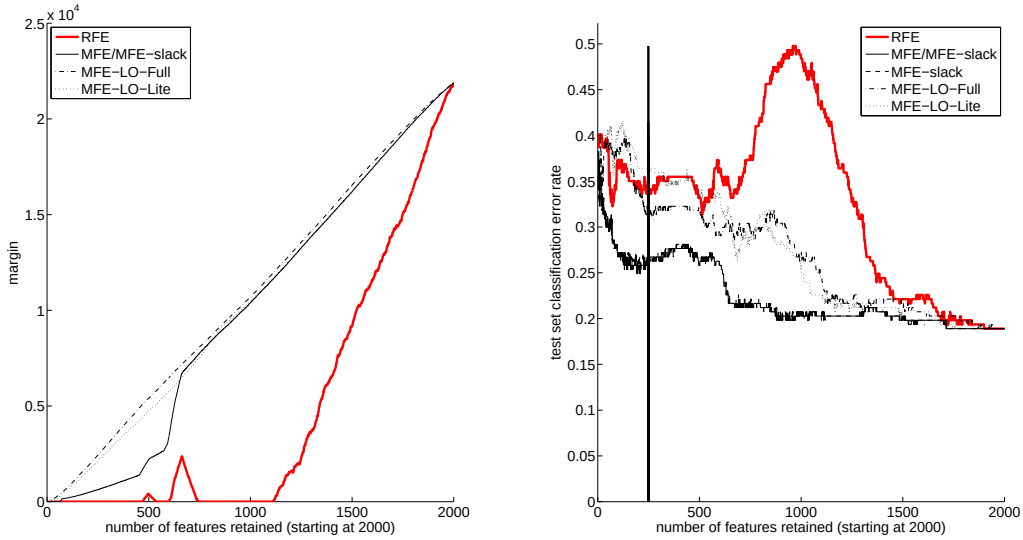
(b) *Colon cancer* data set with 2000 features; linear kernel



(c) *Duke breast cancer* data set with 7129 features; linear kernel

(d) *Leukemia* data set with 7129 features; linear kernel; margin

Fig. 2.6. Average linear SVM test set error rate curves for three gene microarray data sets (Sec. 2.4.3). We truncated the curves from the right to zoom in on the most useful detail in the concluding segment of the feature elimination process – we state the actual "starting number of features" on the x-axis of the plot. In the initial segment (not shown), the curves stay mostly flat, with little or no change in relative performance of the methods. Average training set margin is shown for *Leukemia* in (d).



(a) average training set margin

(b) average test set error rate

Fig. 2.7. Average training set margin and average test set error rate curves on the *Colon cancer* gene microarray data set with 2000 features (Sec. 2.4.3) for the polynomial kernel.

2.4.3 Experiments on high-dimensional separable data

We used three high-d gene microarray data sets which we obtained from the LIBSVM website (9): *Leukemia*, *Duke breast cancer*, *Colon cancer*, respectively with 38, 44, and 62 samples. For the case of the linear kernel, test set classification error rate curves for our methods and RFE are shown in Fig. 2.6. Also shown is training set margin for one of the data sets. All curves shown are averages over 10 trials (and for each trial, the initial SVM was a training set separator). The vertical line indicates the average number of retained features at which separability was lost, over the 10 trials, under our "basic MFE" method. Note that, consistent with theory (12), this is a small number, comparable to the number of training points. Thus, for Figs. 2.6(a) and 2.6(c), on average MFE was able to eliminate from 7129 features down to approximately 100, with MFE-slack applied thereon. For each data set for the linear kernel case, we see that basic MFE achieved much larger training set margins (on average) than RFE and other MFE variants and this was accompanied by much better generalization performance (lower test set error rate curve).

On the 2000-feature *Colon cancer* data, we also eliminated features solely using MFE-slack. For all MFE-slack experiments herein, the MFE-slack C value used (in (2.11)) for each trial was chosen via the procedure given in Sec. 2.4.1 for initial (pre-elimination) classifier design. For each trial, it turned out this C was large enough that there were no margin violations at

[8] If a criterion allows *no* features to remain at the stopping point, this is obviously not a good criterion.

[9] Since the error sequence $\underline{e}$ to be used here must not come from the held-out (test) set $\bar{X}$, we obtain it from the non-heldout set X in a separate experiment accompanying the main experiment – after a training/validation (T/V) split (70-30%) of X, an SVM is trained on set T using the SVM hyperparameters that were earlier selected via the procedure in Sec. 2.4.1 for the main experiment, and the feature elimination order obtained subsequently from set T is applied to set V to obtain $\underline{e}$.

any elimination steps; thus, the MFE/MFE-slack and MFE-slack elimination sequences were identical. However, the two elimination sequences are not always the same in practice when C is chosen using the Sec. 2.4.1 procedure – we will demonstrate this shortly for other data sets.

Several comments should be made at this point. First, although in Sec. 2.4.4 we showed that our novel basic MFE achieved both larger margins and better generalization than RFE for the low-d separable case, we caution that the *theoretical* connection between margin and error rate pertains to an upper bound (39) and does not really tell us how our various methods (with different degrees of margin optimization) will relatively perform (i.e., more margin leading to better error rate may not be an accurate rule-of-thumb). Second, as discussed in Sec. 2.2.3, the greedy (stepwise) nature of feature elimination by our MFE methods and RFE (including variants performing periodic retraining) does not guarantee dominance of one method over another even with respect to margin – it is theoretically possible basic MFE gives a solution sequence with a larger margin curve than either MFE-LO or MFE-Retrain, even though these latter methods adjust the weight vector to increase margin at each step.

Keeping these points in mind, in Fig. 2.6(d), MFE-LO (Full or Lite) did not achieve a higher training set margin curve than basic MFE, unlike our earlier (separable) illustrative example in Fig. 2.3 and unlike our upcoming (separable) example Fig. 2.9(d) for the *UCI mfeat* data set. These examples emphatically illustrate our second point in the preceding paragraph. Note further that RFE-Retrain achieves larger margin than MFE-Retrain. With respect to generalization performance, in Fig. 6, basic MFE and MFE-slack outperformed MFE-LO, as well as MFE-Retrain (the latter method did achieve a higher margin curve than basic MFE, as shown).

The inconsistency between the degree of margin optimization and test error generalization performance can perhaps be understood as a type of "overfitting". This interpretation is also consistent with the good performance of MFE and MFE-slack – the MFE-slack performance will be demonstrated again in Sec. 2.4.4 and extensively in Sec. 2.4.5. We note also that MFE and MFE-slack are the only two among our margin-based methods that, while eliminating features, preserve the initially designed SVM weights. Thus, in this sense, the basic MFE and MFE-slack methods do not "stray" as far from the original SVM solution, and this may have bearing on their performance. Fig. 2.6 also illustrates that MFE-Retrain outperformed RFE-Retrain in generalization in one of the three data sets (*Colon cancer*), and these two methods were competitive for the other two data sets. Recall and notice again that both methods are outperformed by the "basic" MFE method that does not retrain during the elimination steps.

For the case of a polynomial kernel of degree 3, average test set error rate curves for our methods and RFE are shown in Fig. 2.7 for *Colon cancer* (representative of high-d data sets). Also shown is average training set margin. We see that basic MFE again achieved much larger training set margins (on average) than RFE and this was accompanied by much better generalization performance (lower test set error rate curve). In this case, MFE/MFE-slack and MFE-slack produced different elimination sequences; this is hardly discernable in Fig. 2.7, since the generalization performance of the two methods differed only slightly. We also note that basic MFE/MFE-slack and MFE-slack achieved the best generalization performance and were significantly better than the other MFE variants. Comparing Fig. 2.7(b) and 2.6(b), we see that better classification results were obtained for the linear kernel than the polynomial kernel.

Comparison with NLPSVM: Since basic MFE had the best generalization performance for each data set above, we next compared this method to NLPSVM for the linear kernel case. For each trial t, let $M_{t,NLPSVM}$ denote the number of features selected by NLPSVM at its stopping point, and let $E_{t,NLPSVM}$ be the associated test set error rate. For the same trial, let $M_{t,MFE}$ be the number of features selected by MFE/MFE-slack at its stopping point using our *accel* criterion and let $E_{t,MFE}$ be the associated test set error rate. Let M_{NLPSVM} be the average number of features selected by NLPSVM across all trials and let E_{NLPSVM} be the associated average test set error rate, with M_{MFE} and E_{MFE} the corresponding quantities for MFE/MFE-slack. In Fig.

2.6, to compare the generalization performance of MFE/MFE-slack and NLPSVM, we plotted (M_{MFE}, E_{MFE}) as a circle and (M_{NLPSVM}, E_{NLPSVM}) as a diamond. It is important to note that generally $M_{t,MFE}$ will vary across trials and thus the point (M_{MFE}, E_{MFE}), again depicted as a circle, will not necessarily lie on the shown average MFE/MFE-slack curve. Observing these points, notice that the MFE/MFE-slack test set error rate is much lower than the NLPSVM test set error rate for each of the three data sets, albeit achieved with a larger set of selected features than for NLPSVM based on its stopping criterion.

In Fig. 2.8, we illustrate the excursions that the NLPSVM method takes in practice during its iterations (each computing a different decision boundary) towards a final decision boundary. For a typical trial, the dotted path connects (number of features, test set error rate) points for consecutive NLPSVM iterations from its initial iteration (which the "start" arrow points to) to its final iteration (marked by "finish").[10] As demonstrated by the zig-zag movements on the dotted path, the number of features is not monotonic with NLPSVM iterations and different iterations may revisit the same number of features. As demonstrated jointly by the left-right shifts (in number of features) and large up-down shifts (in test set error rate), a wide range of test set error rates is achieved by NLPSVM as it performs consecutive iterations. On the other hand, the MFE/MFE-slack test error rate curve trend is flat or decreasing with increasing number of features (an illustrative MFE single trial is shown in Fig. 2.8(b)). Thus, even though our heuristic stopping criterion is suboptimal[11], if we *restore* eliminated features starting from the MFE/MFE-slack stopping point we are likely to either improve the error rate or leave it unchanged. As indicated by Fig. 2.8, this statement is not true for the NLPSVM method. If the number of NLPSVM features is increased, there is no strong likelihood that the error rate will be better, and going back several iterations from the NLPSVM stopping point gives no guarantee of the same features being retained or of comparable error rate performance. That is, for the NLPSVM method, there is much less predictability of generalization performance than for MFE, in the vicinity of the method's stopping point.

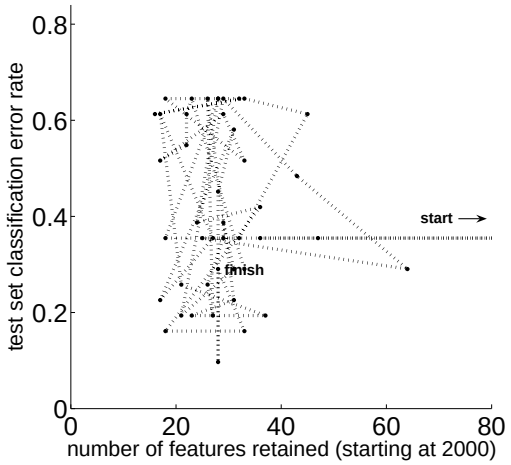
2.4.4 Experiments on low-dimensional separable data

For low-dimensional separable data, average test set error rate curves for three (data set, kernel) pairings are shown in Fig. 2.9. Also shown are average training set margin curves for one of the data sets. As previewed in Sec. 2.4.3, the curves in Fig. 2.9 show that our MFE/MFE-slack method achieved better generalization performance than RFE, and also performed best among all of our methods. These results, like those in the following subsection, are representative of a more extensive study we have conducted involving more data sets and more (data set, kernel) pairings.

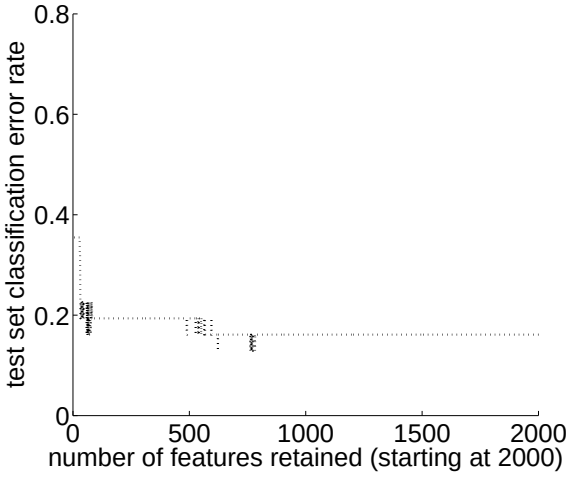
We show the MFE-Retrain and RFE-Retrain generalization performances for the linear kernel case for two data sets in Figs. 2.9(a) and 2.9(b), where we again find that MFE-Retrain is either outperforming or performing competitively with RFE-Retrain. Again, both methods are outperformed by "basic" MFE and MFE-slack (which do not retrain during the elimination steps). As was the case for high-d data sets in Sec. 2.4.3, MFE/MFE-slack and MFE-slack produced the same elimination sequence for one data set (*hepatitis* in Fig. 2.9(b)), whereas for other data sets (*dermatology* in Fig. 2.9(a) and *mfeat* in Fig. 2.9(c)) the two methods produced different sequences and their generalization performances differed very slightly. In Figs. 2.9(a) and 2.9(b) (i.e. the linear kernel case), observing the MFE/MFE-slack and NLPSVM points (circle and diamond, respectively), notice that the NLPSVM test set error rate is slightly higher

[10] An iteration's test set error rate is associated with its (iteration-specific) feature subset. These features correspond to the coordinates not zeroed-out by NLPSVM in the iteration's weight vector, $\underline{w}$.

[11] In Fig. 2.6, for the *Duke breast cancer* data set, for example, note that it may be better to stop basic MFE at approximately 350 features.



(a) NLPSVM



(b) MFE/MFE-slack

Fig. 2.8. Illustrative single trials for the 2000-feature *Colon cancer* data set.

than the MFE/MFE-slack test set error rate for these two data sets, and this is achieved with a slightly smaller number of selected features than MFE/MFE-slack.

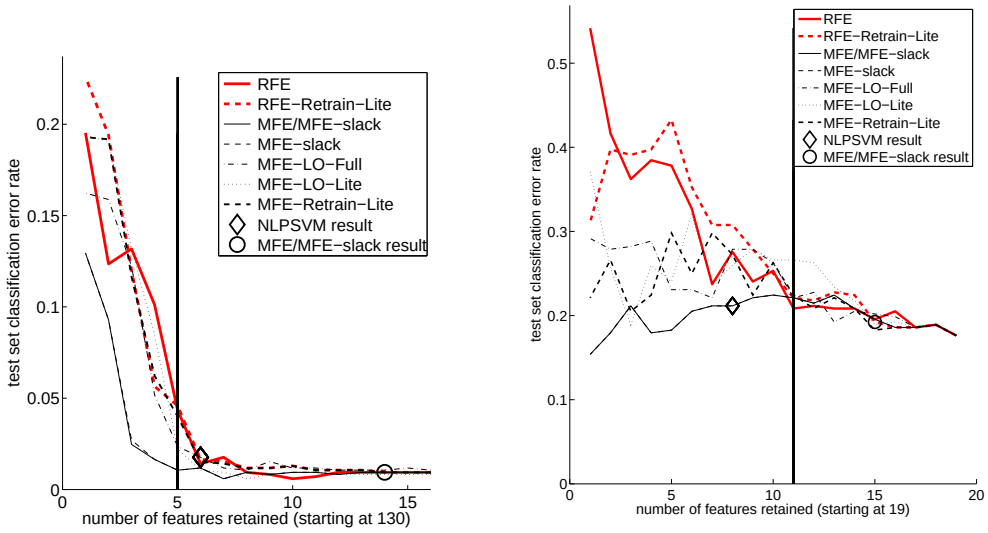
2.4.5 Experiments on low-dimensional nonseparable data

For low-dimensional nonseparable data, average test set error rate curves are shown in Fig. 2.10 for six (data set, kernel) pairings. Excluding Fig. 2.10(d), the curves in Fig. 2.10 (for 5 data sets) are averages over 10, 10, 6, 6, 9 trials, respectively, which were the nonseparable trials among the 10 trials generated. For Fig. 2.10(d), although the curves are averages over 9 *separable* trials among the 10 trials generated, notice from the location of the vertical line that while this *UCI car* data set was initially separable, it immediately lost separability when feature elimination commenced and thus we concluded it is more suitable for discussion in this section than Sec. 2.4.4. The curves in Fig. 2.10 show that MFE-slack consistently achieved better generalization performance than RFE and RFE-abs. Experiments (not shown) found this same result for additional (UCI data set, kernel) pairings. We also note that for *flag* and *ionosphere* in Fig. 2.10 (and e.g. Fig. 2.6(a)), for which we applied RFE retraining, this retraining did improve performance over RFE without retraining. In Figs. 2.10(a) and 2.10(b) (i.e. the linear kernel case), the NLPSVM test set error rate (diamond) is slightly lower than the MFE/MFE-slack test set error rate (circle) albeit achieved with a slightly larger set of selected features.

2.4.6 High-dimensional feature space application: brain images

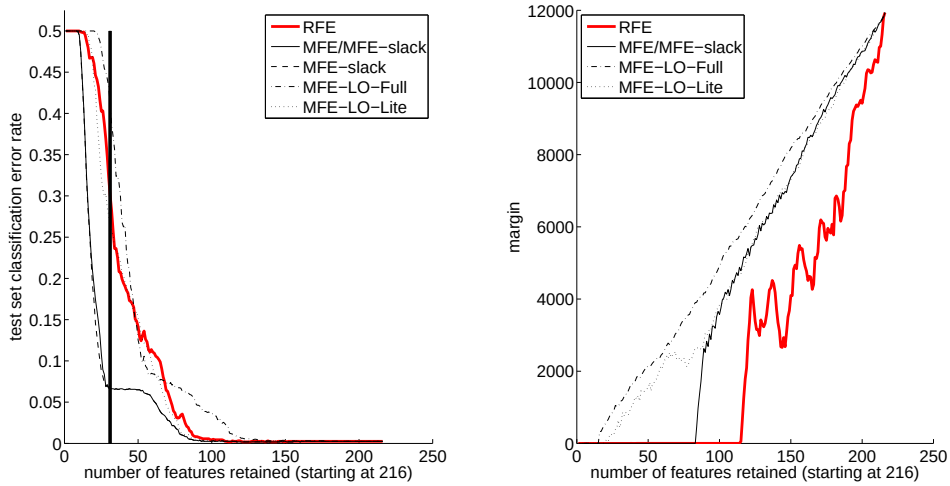
We processed 47 T_1-weighted 3D MRI images (12 Alzheimer's Disease (AD), 35 Control). After segmentation and registration, a Gray Matter tissue density image called "RAVENS" (17; 18; 35; 63) was generated for each by using the HAMMER software (62) and then smoothed by a 5mm Gaussian filter. After identically cropping all (3D) RAVENS images to remove non-brain background, we considered each voxel as a feature and performed SVM classification and feature elimination experiments on a slice-by-slice basis, as follows. First, we split the 47 samples into a training set ($\mathcal{X}$, with 8 AD and 25 Control samples) and a test set ($\bar{\mathcal{X}}$, with 4 AD and 10 Control samples). For *each* of the (approximately) 150 2D (151×186) slices comprising a 3D image, we trained a linear SVM, which perfectly classified all training samples as AD or Control. Next, the RFE and basic MFE methods each eliminated one voxel at a step, generating each method's ordered set of discriminating voxels for that slice. Training set margins and test set classification error rates were averaged over 55 (out of the 150) of these slice-specific classifiers. Fig. 2.11 shows that MFE achieved larger margins and generalized better than RFE. In Fig. 2.12, we show 12 slices that contain the hippocampus[12], with a foreground of overlaid colors. The background image commonly used for such an overlay is the average over the (spatially registered) brain image population used in the experiment – thus, in this case, it is the average of the RAVENS Gray Matter images. In Fig. 2.12(a), the orange regions are those determined by MFE to be the most discriminating regions for AD, *i.e.* these contain the retained voxels for MFE up until the point where the data became nonseparable. Likewise, red indicates the regions found by RFE (with the same number of retained voxels as MFE), and white indicates the regions found by both MFE and RFE. Thus, Fig. 2.12(a) indicates that MFE is detecting the hippocampus, whereas RFE is not. Fig. 2.12(b) shows a confidence ranking among the MFE regions that were displayed (in orange and white) in Fig. 2.12(a). Higher rank for a voxel means it is more discriminating for AD. We indicate the ranking using the following colormap: red=high, yellow=higher, white=highest. The ranking was based simply on the feature elimination order. Notice almost all of the smallest clusters (among the retained voxels which are shown in color) have received the lowest ranking – it is significant that MFE's most discriminating voxels are found in sizeable, spatially compact voxel clusters. It is also

[12] A well-published "marker" brain structure for AD.



(a) *UCI dermatology* data set with 130 features; linear kernel

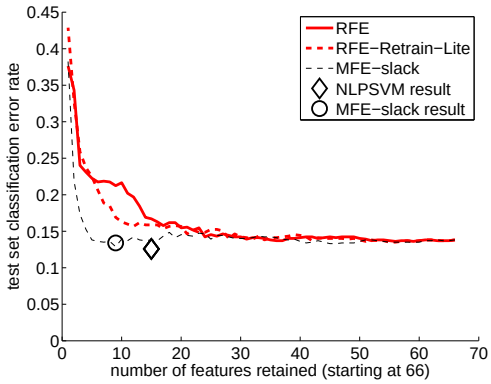
(b) *UCI hepatitis* data set with 19 features; linear kernel



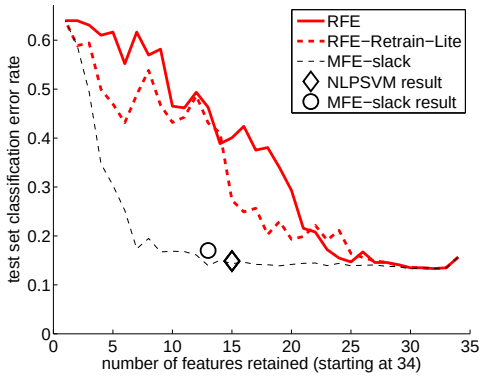
(c) *UCI mfeat* data set with 216 features; polynomial kernel

(d) *UCI mfeat* data set 216 features; polynomial kernel; margin

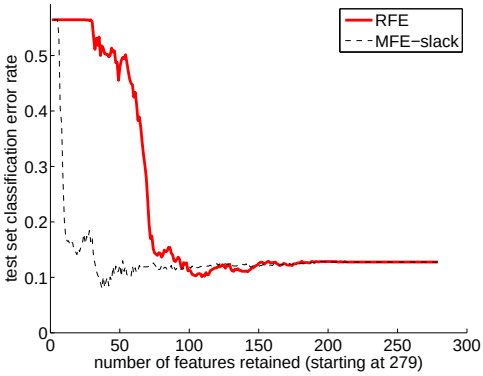
Fig. 2.9. Average test set classification error rate, (a)-(c), for three separable low-dimensional UC Irvine data sets (Sec. 2.4.4). We truncated the *dermatology* graph from the right to zoom in on the most useful detail (see truncation description in Fig. 2.6). The vertical line indicates the average number of retained features at which separability was lost. Average training set margin is shown for one of the data sets in (d). Initial number of features is stated on the x-axis.



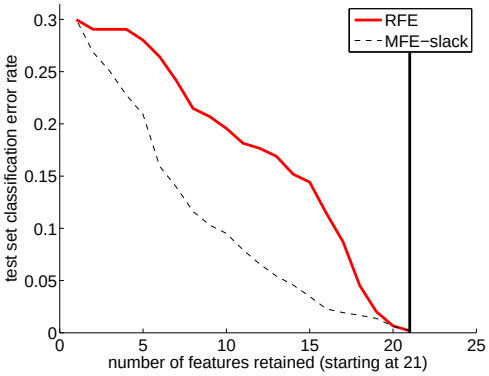
(a) *UCI flag* data set with 66 features; linear kernel



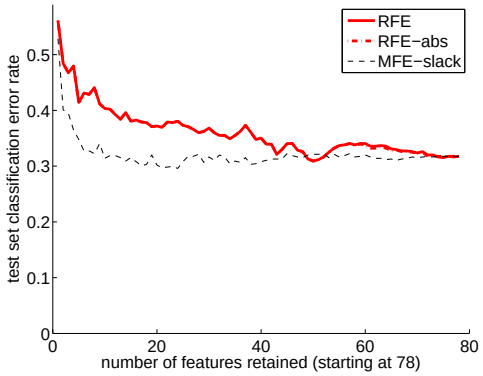
(b) *UCI ionosphere* data set with 34 features; linear kernel



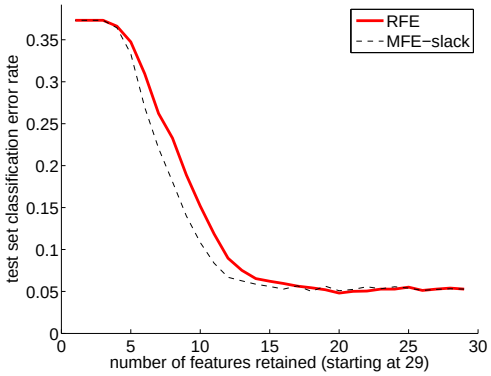
(c) *UCI arrhythmia* data set with 279 features; polynomial kernel



(d) *UCI car* data set with 21 features; polynomial kernel



(e) *UCI cylinder* data set with 78 features; Gaussian kernel



(f) *UCI wisconsin-diagnosis* data set with 29 features; polynomial kernel

Fig. 2.10. Average test set classification error rate on 6 low-dimensional UC Irvine data sets (Sec. 2.4.5). Initial number of features is stated on the x-axis.

significant that, among the retained voxels, those in the hippocampus have almost uniformly received the "higher"-to-"highest" ranking.

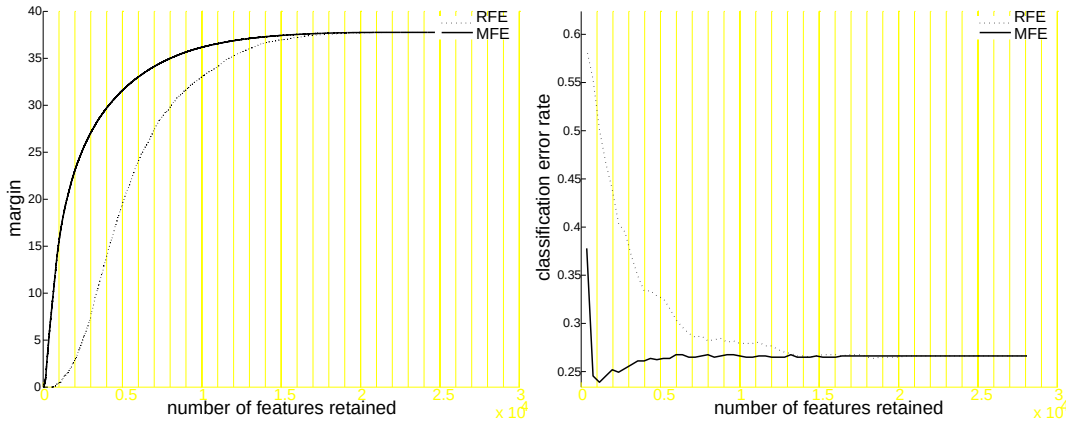
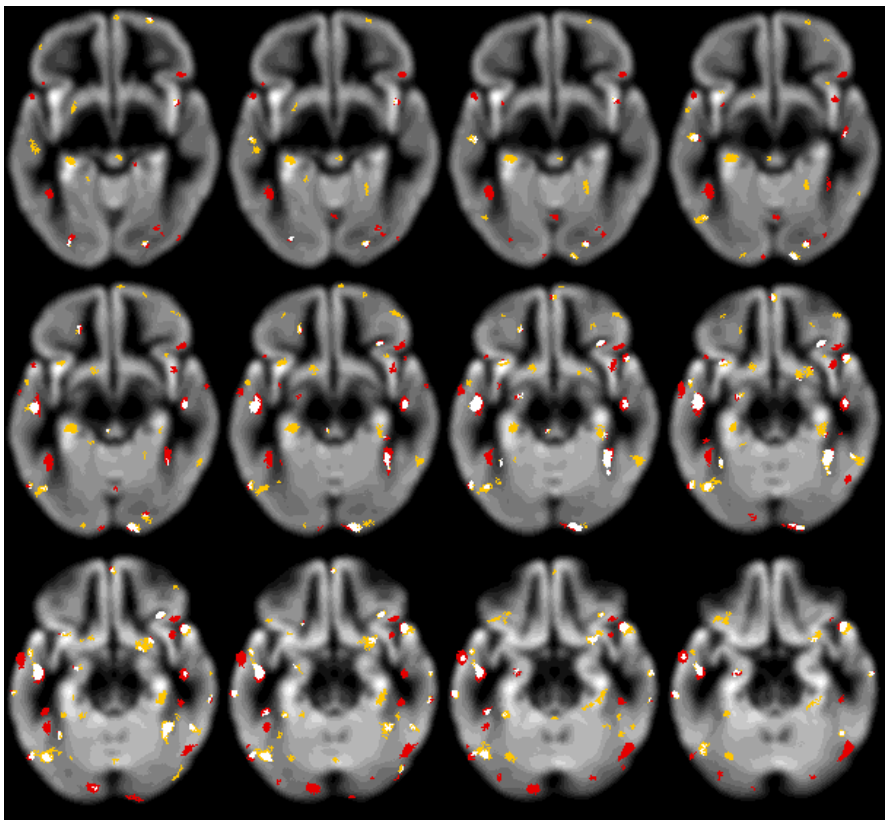


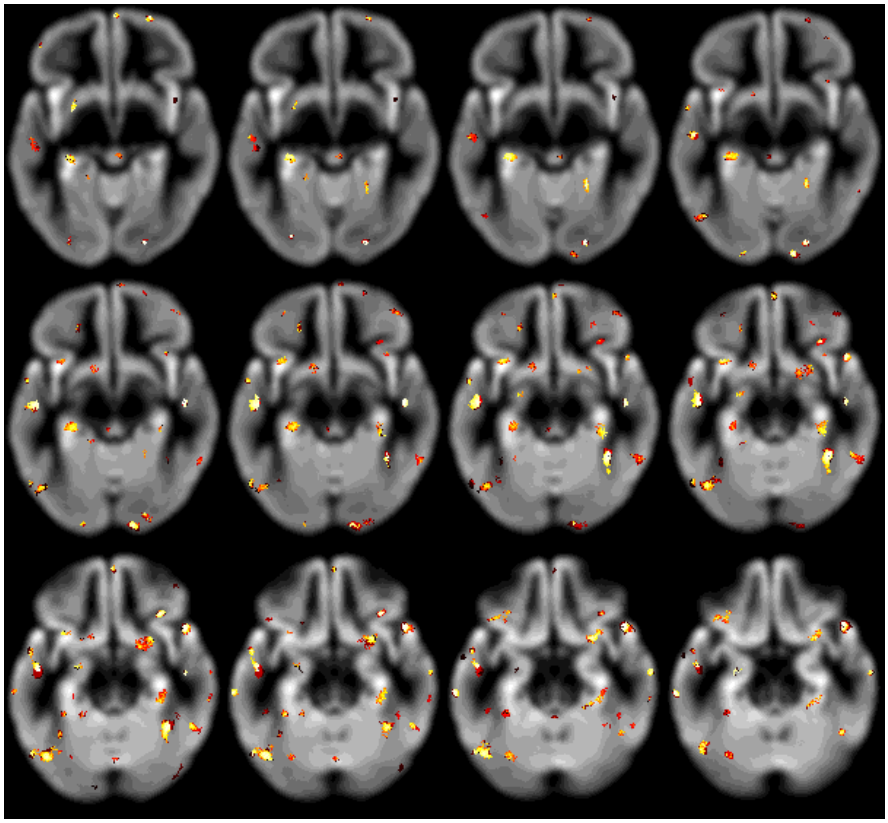
Fig. 2.11. Feature elimination results for high-dimensional brain image data, for the linear SVM case. The initial number of features is 28,086.

2.5 Conclusions

In this chapter, we presented *margin-based feature elimination* and several extensions, applicable to SVMs and other linear and nonlinear, kernel-based discriminant functions. In the nonlinear kernel case, we identified shortcomings of RFE and demonstrated improved margin and accuracy achieved by kernel-based MFE. A second extension performs a lightweight SVM training that adjusts the current solution in the reduced feature space to improve margin. In a third approach, MFE-slack, we formulated a simple extension of MFE to incorporate margin slackness. We evaluated on UCI data sets and gene microarray data sets, toward a comprehensive evaluation of the generalization performances of MFE, RFE, and NLPSVM. We demonstrated that MFE and MFE-slack provide better generalization than RFE, and better or competitive generalization compared with NLPSVM. We found that basic MFE, which requires separable data, was always suitable for high-dimensional data but was also applicable for several low-dimensional UC Irvine data sets. We observed that methods which provide further increases in margin beyond that of basic MFE did not necessarily lead to improved generalization. This was attributed to overfitting and is consistent with the good performance achieved by MFE-slack, which introduces margin slackness. Finally, we also gave illustrative results on brain image data.



(a)



(b)

Fig. 2.12. Retained voxels (i.e. discriminating regions) for Alzheimer's Disease in color (Sec. 2.4.6).

Chapter 3

Margin-maximizing feature selection methods: "multiclass" case ($k > 2$)

3.1 Introduction

Prior works introduced SVMs for the case of more than two classes (aka the "multiclass" case) – we begin this chapter by giving a summary of multiclass SVMs in Sec. 3.2. In Sec. 3.3 we give an overview of RFE-based feature elimination methods for multiclass SVMs (77), where we discuss that these methods have been derived from the suboptimal two-class RFE objective and are thus again suboptimal, motivating us to develop multiclass "basic MFE", i.e. MFE-k, starting with Sec. 3.4. We introduce several MFE-k methods, corresponding to several ways one can define "multiclass margin", and for each method we again introduce a natural extension that incorporates slackness. Lastly in this chapter, in Sec. 3.7, we experimentally evaluate the relative performance of our multiclass MFE-based methods and show that MFE-k outperforms the multiclass RFE-based methods.

3.2 Brief summary of multiclass SVMs

Consider a labeled training set $\{(\underline{x}_n, p_n), n \in \Omega \equiv \{1, \ldots, N\}\}$, where $\underline{x}_n = [x_{n,1}, \ldots, x_{n,M}]^{\mathrm{T}} \in \mathbf{R}^M$ is the n-th data sample and $p_n \in P \equiv \{1, \ldots, k\}$ is its class label. Denote $\bar{p} \equiv P \backslash p$. Let N_p be the number of samples in class $p \in P$, so $N = \sum_{p=1}^{K} N_p$. $\Omega_p \equiv \{n \in \Omega | p_n = p\}$ denotes the set of samples in class p; $\Omega_{p,q} \equiv \Omega_p \cup \Omega_q$. A multiclass support vector machine (SVM) is a linear or generalized linear classifier that achieves maximum margin, i.e. it maximizes the minimum distance of any training sample to the decision boundary. SVM training learns the set of discriminant functions $\{f_p(\cdot)\}$, with the decision function given by:

$$p^* = \arg\max_{p \in P} f_p(\underline{x}). \tag{3.1}$$

The "support vectors", used to specify the SVM solution, are a special subset of the training points, identified below. Denote $g_{n,q} \equiv g_q(\underline{x}_n) \equiv f_{p_n}(\underline{x}_n) - f_q(\underline{x}_n)$ for $q \in \bar{p}_n$, and $\mathcal{N}_{p,q} \equiv \min_{n \in \Omega_{p,q}} g_{n,r_n}$ where $r_n \equiv \{p, q\} \backslash p_n$. Denote $A_p \equiv \{l \in \Omega_p | g_{l,t} > 0 \ \forall t \in \bar{p}\}$, the set of correctly classified samples in class p, and $A \equiv \bigcup_{p=1}^{k} A_p$.

In the linear case, $f_p(\underline{x}) \equiv \underline{w}_p^{\mathrm{T}} \underline{x} + b_p = 0$, $\underline{w}_p = [w_{p,1}, \ldots, w_{p,M}]^{\mathrm{T}} \in \mathbf{R}^M$, $b_p \in \mathbf{R}$, $p \in P$. Denote $L_{p,q} \equiv ||\underline{w}_p - \underline{w}_q||^2$, $d_{n,q} \equiv \frac{g_{n,q}}{\sqrt{L_{p_n,q}}}$ (distance from sample $\underline{x}_n$ to the class-pairwise decision boundary between classes p_n and q), and the *class-pairwise margin* $\gamma_{p,q} \equiv \gamma_{q,p} \equiv \min_{n \in \Omega_{p,q}} d_{n,r_n}$, $r_n = \{p, q\} \backslash p_n$. For a sample $\underline{x}_n$, denote $c_n \equiv \arg\min_{q \in \bar{p}_n} d_{n,q}$, interpreted as follows: if $\underline{x}_n$ is a correctly classified sample, c_n is the class to which $\underline{x}_n$ is closest to being misclassified; else, if $\underline{x}_n$ is a misclassified sample, considering all (class-pairwise) boundaries between class p_n and

another class o that $\underline{x}_n$ is on the o side of (due to being a misclassified sample), c_n is the class from which $\underline{x}_n$ is furthest. In the linear case, $g_{n,q} = \underline{x}_n^{\mathrm{T}}(\underline{w}_{p_n} - \underline{w}_q) + (b_{p_n} - b_q)$ clearly, and the weight vectors of the SVM solution are $\underline{w}_p \equiv \sum_{n \in \Omega} \Lambda_{n,p}\, \underline{x}_n$, where $\Lambda_{n,p} \equiv (\zeta_{n,p} A_n - \lambda_{n,p})$, $0 \leq \lambda_{n,p} \leq C$ are scalar Lagrange multipliers, $A_n \equiv \sum_{p=1}^{k} \lambda_{n,p}$, and $\zeta_{n,p}$ is 1 if $p_n = p$ or is 0 otherwise (74). In the generalized linear (nonlinear) case, $f_p(\underline{x}) \equiv \underline{w}_p^{\mathrm{T}} \underline{\phi}(\underline{x}) + b_p = 0$, $\underline{w}_p \in \mathbf{R}^L$, $b_p \in \mathbf{R}$, $\underline{\phi}(\underline{x}) \equiv [\phi_1(\underline{x}),\ \phi_2(\underline{x}),\ \ldots,\ \phi_L(\underline{x})]^{\mathrm{T}}$, with $\phi_i(\underline{x})$ nonlinear functions of the $\underline{x}$ coordinates. Of particular interest is when inner products between $\underline{\phi}(\underline{x})$ and $\underline{\phi}(\underline{u})$ can be efficiently computed via a positive definite kernel function, $K(\underline{x}, \underline{u}) \equiv \underline{\phi}^{\mathrm{T}}(\underline{x})\underline{\phi}(\underline{u})$. In this case, both $\underline{\phi}(\cdot)$ and $\underline{w}_p$ itself (for any $p \in P$) need not be explicitly defined since the SVM discriminant function $f_p(\cdot)$ and the SVM weight vector squared 2-norm $||\underline{w}_p||^2$ can be expressed solely in terms of the kernel, *i.e.*:

$$f_p(\underline{x}) = \sum_{l \in \Omega} \Lambda_{l,p} K(\underline{x}_l, \underline{x}) + b_p \tag{3.2}$$

$$||\underline{w}_p||^2 = \sum_{n \in \Omega} \sum_{l \in \Omega} \Lambda_{n,p} \Lambda_{l,p} K(\underline{x}_n, \underline{x}_l) \tag{3.3}$$

Thus, $g_{n,q}$ and $L_{p,q}$ are also expressed solely in terms of the kernel, *i.e.*:

$$g_{n,q} = (\sum_{l \in \Omega} \Lambda_{l,p_n} K(\underline{x}_l, \underline{x}_n) + b_{p_n}) - (\sum_{l \in \Omega} \Lambda_{l,q} K(\underline{x}_l, \underline{x}_n) + b_q) \tag{3.4}$$

$$L_{p,q} \equiv ||\underline{w}_p - \underline{w}_q||^2 \equiv ||\underline{w}_p||^2 + ||\underline{w}_q||^2 - 2\underline{w}_p^{\mathrm{T}} \underline{w}_q = \sum_{n \in \Omega} \sum_{l \in \Omega} (\Lambda_{n,p}\Lambda_{l,p} + \Lambda_{n,q}\Lambda_{l,q} - 2\Lambda_{n,p}\Lambda_{l,q}) K(\underline{x}_n, \underline{x}_l) \tag{3.5}$$

This approach (the "kernel trick"), where $K(\cdot, \cdot)$ is chosen and applied within the SVM training, is referred to as the "nonlinear kernel case".

Several distinct multiclass SVM training methods, each learning a multiclass discriminant of the form (3.1), have been proposed in prior works, e.g., (74), (70), (40), (36), (13). Of particular interest is the Weston and Watkins (WW) method (74) which uses an "all-together" design, wherein discriminant functions for all classes are jointly learned to solve the single optimization problem given below by (3.6) (70; 74).

$$\min_{\underline{w}_1, \ldots, \underline{w}_k, b_1, \ldots, b_k, \underline{\xi}} \frac{1}{2} \sum_{p=1}^{k} ||\underline{w}_p||^2 + C \sum_{n=1}^{N} \sum_{q \in \bar{p}_n} \xi_{n,q} \ s.t.\ \xi_{n,q} \geq 0, g_{n,q} \geq 2 - \xi_{n,q}, n = 1, \ldots, N, q \in \bar{p}_n \tag{3.6}$$

Recall that $g_{n,q}$ in the constraints in (3.6) (as well as in the constraints in (3.7) below) simply stands for $\underline{x}_n^{\mathrm{T}}(\underline{w}_{p_n} - \underline{w}_q) + (b_{p_n} - b_q)$ in the linear case and $\langle \underline{\phi}(\underline{x}_n), (\underline{w}_{p_n} - \underline{w}_q) \rangle + (b_{p_n} - b_q)$ in the nonlinear kernel case – that is, as (3.6) states the training problem conveniently for both cases, it is important to keep in mind that "$\underline{w}$" (i.e. $\underline{w}_{p_n}$ and $\underline{w}_q$) and "b" (i.e. b_{p_n} and b_q) indeed appear in the constraints. The design (3.6) allows slackness in the margin constraints, in particular allowing both margin violations (i.e., $\xi_{n,q} > 0$) and misclassifications (a classification error occurs for sample n if $\xi_{n,q} > 2$ for any $q \in \bar{p}_n$).

The BSVM (biased SVM) training method (40), solving the problem given below in (3.7),

$$\min_{\underline{w}_1, \ldots, \underline{w}_k, b_1, \ldots, b_k, \underline{\xi}} \frac{1}{2} \sum_{p=1}^{k} (||\underline{w}_p||^2 + b_p^2) + C \sum_{n=1}^{N} \sum_{q \in \bar{p}_n} \xi_{n,q} \ s.t.\ \xi_{n,q} \geq 0, g_{n,q} \geq 2 - \xi_{n,q}, n = 1, \ldots, N, q \in \bar{p}_n \tag{3.7}$$

adds a bias term of b_p^2 to $||\underline{w}_p||^2$ in the WW objective function in (3.6). Adding the bias term in the objective function was proposed by Mangasarian for the two-class case (26; 50) who showed that the original and the modified SVM training problems differ only slightly (27; 50). Adding the bias term simplifies the corresponding dual problem and makes it faster to solve via decomposition (26; 77), and solutions of the modified formulation mostly coincides with solutions of the original formulation (26). Since BSVM differs only slightly from WW, we use BSVM for training of the initial multiclass classifier in our experiments, as implemented by the BSVM2 tool in the STPR toolkit (65). Moreover, as we will discuss in Sec. 3.6, in one of our multiclass MFE methods, MFE-k-Kesler, we utilize the equivalence that exists via Kesler construction (20) between a one-class SVM problem and the BSVM formulation (as opposed to the WW formulation).

For choosing the SVM training parameter C as well as other SVM hyperparameters in the nonlinear kernel case, we apply the standard practice of using a (bootstrap-based) validation procedure (20).

3.3 Multiclass RFE-based methods

Similar to two-class RFE's feature elimination criteria (equations (2.1) and (2.3) for the linear case and nonlinear case, respectively), $\sum_{p=1}^{k} w_{p,m}^2$ was used by (77) as the ranking criterion for multiclass RFE methods discussed therein. Based on this criterion, (77) introduced and evaluated the feature elimination method MSVM-RFE-WW wherein the $\underline{w}_p$ $\forall p \in P$ are obtained using WW (3.6).

3.4 Multiclass MFE methods

MFE performs margin-maximizing feature elimination (for linear and nonlinear kernel-based binary (two-class) discriminant functions), and MFE-slack is a natural extension of MFE for incorporating slackness and eliminating under nonseparability that reduces to MFE for large enough C when the training data are separable. Aiming to improve on the RFE-based multiclass methods (e.g. MSVM-RFE-WW), we propose margin-optimizing feature elimination for *multiclass* SVMs, i.e. MFE-k. To develop our multiclass MFE approaches, we start by noting that a training sample $\underline{x}_n$ is classified as p_n^* in (3.1) *if and only if* $g_{n,q} > 0 \; \forall q \in \bar{p_n^*}$. Thus, in the event that the training correctly classifies all training samples, the distances $d_{n,q}$ for all training samples $\underline{x}_n$ are positive for all $q \in \bar{p}_n$. Based on this observation, there are several ways to define *multiclass margin*. First, we can define multiclass margin as γ_G, the *global* minimum *class-pairwise margin* $\gamma_{p,q}$ across all class pairs $(p, q \in \bar{p})$ (there are $\binom{k}{2}$ such pairs):

$$\gamma_G \equiv \min_{p,q \in \bar{p}} \gamma_{p,q} \tag{3.8}$$

Alternatively, as a second way, multiclass margin can be defined as γ_{SP}, the sum of all class-pairwise margins $\gamma_{p,q}$:

$$\gamma_{SP} \equiv \sum_{p=1}^{k} \sum_{q<p} \gamma_{p,q} \tag{3.9}$$

Finally, multiclass margin can be defined as γ_{SC}, the sum of all k *classwise (one-versus-rest) margins*, with *classwise margin* γ_p for class p defined as $\gamma_p \equiv \min_{q \in \bar{p}} \gamma_{p,q}$:

$$\gamma_{SC} \equiv \sum_{p=1}^{k} \gamma_p \tag{3.10}$$

For an example separable data with three classes, Fig. 3.1(a) illustrates the $\binom{3}{2}$ class-pairwise margins $\gamma_{p,q}$ and states the resulting γ_G, γ_{SP}, and γ_{SC}.

For separable training data, we propose the margin-maximizing feature elimination methods MFE-k-G, MFE-k-SP, and MFE-k-SC, which eliminate to preserve the largest γ_G, γ_{SP}, and γ_{SC}, respectively. These methods require d_{n,p_n} for all training samples $\underline{x}_n$ to remain positive upon elimination, *i.e.*, m_i, the feature eliminated at step i, must ensure $g_{n,q}^{i,m_i} > 0$[1] for all $(n \in \Omega, q \in \bar{p}_n)$ pairs, *i.e.*, m_i must belong to the candidate feature set $S(i) = \bigcap_{p \in P,\ q \in \bar{p}} S_{p,q}(i)$ where $S_{p,q}(i) \equiv \{m \notin \mathcal{M} \mid \mathcal{N}_{p,q}^{i,m} > 0\}$. The pseudocode for these three methods differ only slightly and are given in Sec. 3.4.1. In Sec. 3.5, we propose natural extensions of these three methods, that allow incorporating slackness and eliminating under nonseparability, which, in the event the data are separable reduce, for large enough C, to these three respective methods.

3.4.1 Pseudocode for algorithms MFE-k-G, MFE-k-SP, and MFE-k-SC, for linear and nonlinear kernel-based multiclass SVMs

0. *Preprocessing:* Let $\mathcal{M}$ be the set of eliminated features, with $\mathcal{M} = \emptyset$ initially. First run SVM training on the full space to find a multiclass discriminant function set $\{f_p(\cdot),\ p = 1, \ldots, k\}$ that correctly classifies each training sample $\underline{x}_n$. As confirmation that the set you found *is* correctly classifying each $\underline{x}_n$, note that all $g_{n,q}^{-1,0}$ values you compute below in the next sentence (for all $(n \in \Omega, q \in \bar{p}_n)$ pairs) must be positive. If linear kernel case, $\forall (n \in \Omega, q \in \bar{p}_n)$, compute $\delta_{n,q}^m \equiv x_{n,m}(w_{p_n,m} - w_{q,m})\ \forall m$, and $g_{n,q}^{-1,0} = (b_{p_n} - b_q) + \sum_{m=1}^{M} \delta_{n,q}^m$, where $i = -1$ means before eliminating any features and $m_{-1} = 0$ is a dummy placeholder index value; if nonlinear kernel case, compute $g_{n,q}^{-1,0}\ \forall (n \in \Omega, q \in \bar{p}_n)$ using (3.4). Compute $L_{p,q}^{-1,0} = ||\underline{w}_p - \underline{w}_q||^2$ for all $(p \in P,\ q \in \bar{p})$. Set $i \leftarrow 0$. At elimination step i, perform the following operations:

1. For each $m \notin \mathcal{M}$, using recursion, compute $g_{n,q}^{i,m}$ for all $(n \in \Omega, q \in \bar{p}_n)$ pairs: if linear kernel case, this computation is $g_{n,q}^{i-1,m_{i-1}} - \delta_{n,q}^m$, where $\delta_{n,q}^m$ need not be computed in this step if stored during preprocessing (step 0); if nonlinear kernel case, compute $g_{n,q}^{i,m}$ using recursion on the kernel computation. Determine $\mathcal{N}_{p,q}^{i,m} = \min_{n \in \Omega_{p,q}} g_{n,r_n}^{i,m}$ for all $(p \in P,\ q \in \bar{p})$ pairs. Determine $S_{p,q}(i) = \{m \notin \mathcal{M} \mid \mathcal{N}_{p,q}^{i,m} > 0\}$ and the candidate feature set $S(i) = \bigcap_{p \in P,\ q \in \bar{p}} S_{p,q}(i)$. If $S(i)$ is empty (the data is nonseparable) then **stop**.[2]

2. For $m \in S(i)$ and each class pair $(p, q < p)$, using recursion compute $L_{p,q}^{i,m}$: if linear kernel case, this computation is $L_{p,q}^{i,m} = L_{p,q}^{i-1,m_{i-1}} - (w_{p,m} - w_{q,m})^2$; if nonlinear kernel case, compute $L_{p,q}^{i,m}$ using recursion on the kernel computation. Compute $\gamma_{p,q}^{i,m} = \frac{\mathcal{N}_{p,q}^{i,m}}{\sqrt{L_{p,q}^{i,m}}}$.

---

[1] $q^{i,m} \equiv$ quantity q at feature elimination step i upon elimination of feature m.

[2] The set $S(i)$ consists of the features at step i that, if singly eliminated, will preserve *class-pairwise* positive margins. Margin will only be evaluated for features in the set $S(i)$.

3.1. If using MFE-k-G, determine feature $m_i = \arg \max_{m \in S(i)} \min_{p \in P, q<p} \gamma_{p,q}^{i,m}$; else if using MFE-k-SP, determine feature $m_i = \arg \max_{m \in S(i)} \sum_{p=1}^{k} \sum_{q<p} \gamma_{p,q}^{i,m}$; else if using MFE-k-SC, determine feature $m_i = \arg \max_{m \in S(i)} \sum_{p=1}^{k} \min_{q \in \bar{p}} \gamma_{p,q}^{i,m}$. Eliminate feature m_i, *i.e.* $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

3.2. Keep for the next iteration only the recursive quantities associated with the eliminated feature – they are $\{g_{n,q}^{i,m_i} \ \forall (n,q)\}$ and $\{L_{p,q}^{i,m_i} \ \forall (p,q<p)\}$ in the linear kernel case, whereas, in the nonlinear kernel case, they are the quantities defined for the recursive kernel computation (e.g., $\{\mathcal{H}_{k,n}^{i,m_i} \ \forall (k,n)\}$, $\{\mathcal{K}_{k,n}^{i,m_i} \ \forall (k,n)\}$).

3.3. $i \rightarrow i+1$ and go to step 1.

3.5 Multiclass MFE-slack: utilizing margin slackness

In MFE-slack for the two-class case, as developed in (3), for each *candidate* feature for elimination, m, we evaluated *every* (correctly classified) training sample $\underline{x}_n$ as the potential margin-defining sample (dubbed "anchor" $\underline{x}_{n_a}$) associated with this elimination, and we chose the pair (m_{MFE-S}, n_{MFE-S}) to minimize (via discrete optimization) the same objective function as used by the pre-elimination SVM training and meet its constraints, with quantities in the objective and constraints, such as slackness $\underline{\xi}$, evaluated *post*-feature-elimination. We showed that, for large enough C (and assuming the data is separable), positive slackness values are not tolerated and minimization of $\frac{1}{2}||\underline{w}||^2 + C \sum_{n=1}^{N} \xi_n$ (*i.e.*, $\frac{1}{2}(1/margin)^2 + C \sum_{n=1}^{N} \xi_n$, where *margin* $1/||\underline{w}||$ was set by the anchor via scaling by a ρ parameter) *reduces* to minimization of $1/margin$. Thus, MFE-slack reduces to strict margin-maximizing elimination (MFE) as a special case for large enough C. In a corresponding fashion, in the multiclass case, based on any one of the above three definitions of *multiclass margin* we can choose the discrete *maximization* objective function as *multiclass margin minus* a (C-weighted) sum of slackness values. Similar to the two-class case, maximizing this objective yields a slackness-based extension of multiclass MFE that reduces to one of our (respective) strict-margin-achieving multiclass MFE methods for large enough C. We next develop these slackness-based MFE-k extensions in subsections 3.5.1, 3.5.2, 3.5.3. We will again denote that $q^{i,m_c,n_a} \equiv$ quantity q at elimination step i upon elimination of candidate feature m_c when the candidate anchor is $\underline{x}_{n_a}$.

3.5.1 MFE-k-SP-slack

Based on the γ_{SP} definition, we choose the discrete *maximization* objective function for MFE-k-SP-slack as:

$$\gamma_{SP} - C \sum_{p=1}^{k} \sum_{q<p} \sum_{n \in \Omega_{p,q}} \psi_{n,r_n} = \sum_{p=1}^{k} \sum_{q<p} (\gamma_{p,q} - C \sum_{n \in \Omega_{p,q}} \psi_{n,r_n}) \tag{3.11}$$

where $r_n \equiv \{p,q\} \backslash p_n$ (cf. Sec. 3.2) and meet the following slackness constraints upon choosing a correctly classified (candidate) *class-pairwise anchor* $n_{a_{p,q}} \in A_p \cup A_q$ for each class pair ($p \in P, q \in \bar{p}$):

$$d_{n,r_n} \geq d_{n_{a_{p,q}}, r_{n_{a_{p,q}}}} - \psi_{n,r_n} \quad \forall n \in \Omega_{p,q}, \ \forall (p \in P, q \in \bar{p}), \ r_n \equiv \{p,q\} \backslash p_n, \ \psi_{n,r_n} \geq 0 \tag{3.12}$$

with slackness ψ_{n,r_n} for $n \in \Omega_{p,q}$ defined as $\max(0, d_{n_{a_{p,q}},r_{n_{a_{p,q}}}} - d_{n,r_n})$, where $d_{n_{a_{p,q}},r_{n_{a_{p,q}}}}$ (which is computed as $\frac{g_{n_{a_{p,q}},r_{n_{a_{p,q}}}}}{||w_p - w_q||}$) sets the (possibly violated) *class-pairwise margin* $\gamma_{p,q}$ in (3.11). In this method, each class pair has its own anchor for a given candidate feature for elimination. Thus, there are $\binom{k}{2}$ anchors working collectively for each candidate, m. In the three-class example illustrated by Fig. 3.1(d) for the scenario where two dimensions remain upon candidate elimination of one of three features, the encircled samples 7, 3, and 15 represent one particular choice of $\binom{3}{2}$ anchors working collectively. These anchors are setting the class-pairwise margins marked as $\gamma_{1,3}$, $\gamma_{2,1}$, $\gamma_{2,3}$ (indicated by solid lines), relative to which the class-pairwise (positive) slacknesses (indicated in matching color, using short dashed lines) are calculated. All anchor-based computations will be specified in detail in Sec. 3.5.1.1 shortly. MFE-k-SP-slack selects the pair $(m^{SP-slack}, \{n^{SP-slack}_{a_{p,q}}, q < p\})$ that, post-elimination, maximizes (3.11) over all the discrete choices $\{(m, \{n_{a_{p,q}}, q < p\})\}$. MFE-k-SP-slack reduces to margin-maximizing (γ_{SP}-maximizing) elimination for large enough C (assuming the data is separable).

3.5.1.1 MFE-k-SP-slack algorithm pseudocode for linear and nonlinear kernel-based SVMs

0. Same as step 0 in Sec. 3.4.1, except the set of discriminating functions need not be a separating one.

1. Using the recursive computation descriptions in steps 1 and 2 in Sec. 3.4.1, compute $L^{i,m}_{p,q}$ for all $(m \notin \mathcal{M}, p \in P, q \in \bar{p})$, and compute $g^{i,m}_{n,q}$ and $d^{i,m}_{n,q} = \frac{g^{i,m}_{n,q}}{\sqrt{L^{i,m}_{p,q}}}$ for all $(m \notin \mathcal{M}, n \in \Omega, q \in \bar{p}_n)$. We consider any data point to be a *valid* candidate anchor for feature elimination candidate m if it is not misclassified under that elimination; for each $(m \notin \mathcal{M}, p \in P)$ pair, determine $A^{i,m}_p = \{l \in \Omega_p | g^{i,m}_{l,t} > 0 \ \forall t \in \bar{p}\}$ (the set of *valid* candidate anchors in class p) and $A^{i,m} = \bigcup_{p=1}^{k} A^{i,m}_p$.

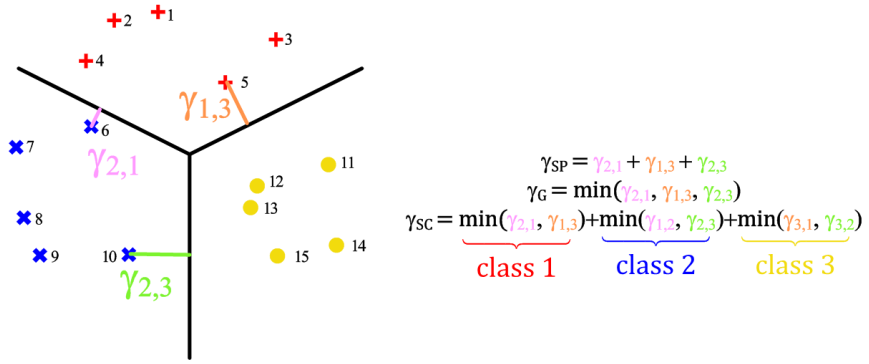
2. For each $(m \notin \mathcal{M}, p \in P, q \in \bar{p}, n_{a_{p,q}} \in A^{i,m}_p \cup A^{i,m}_q)$ 4-tuple, to ensure zero slackness for the *class-pairwise* anchor $n_{a_{p,q}}$ (*i.e.*, $\psi^{i,m,n_{a_{p,q}}}_{n_{a_{p,q}},r_{n_{a_{p,q}}}} = 0$) and to meet all the constraints in (3.12), compute $\psi^{i,m,n_{a_{p,q}}}_{n,r_n} = \max(0, d^{i,m}_{n_{a_{p,q}},r_{n_{a_{p,q}}}} - d^{i,m}_{n,r_n})$ $\forall n \in \Omega_{p,q}$.

3.1 Determine the outcome $(m_i, \{n^i_{a_{p,q}}, p = 1, \ldots, k, q < p\})$ of the discrete maximization shown below in (3.13), and eliminate feature m_i, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.[3]

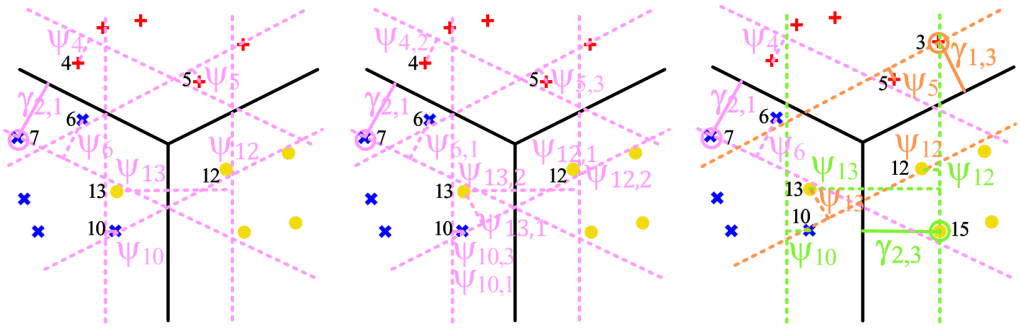
$$\arg\max_{m \in \bar{\mathcal{M}}} \sum_{p=1}^{k} \sum_{q<p} \max_{n_{a_{p,q}} \in A^{i,m}_p \cup A^{i,m}_q} (d^{i,m}_{n_{a_{p,q}},r_{n_{a_{p,q}}}} - C \sum_{n \in \Omega_{p,q}} \psi^{i,m,n_{a_{p,q}}}_{n,r_n}) \tag{3.13}$$

3.2 $i \rightarrow i+1$ and go to step 1.

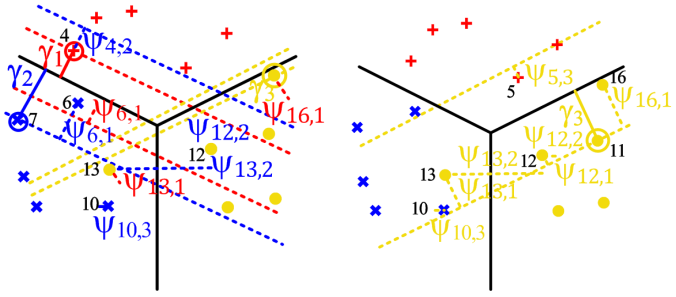
[3] The simple default way to choose C in (3.13) is to set it to the C value used for the pre-elimination SVM training; however, a different C value may be chosen for (3.13), e.g. based on a cross-validation procedure applied after eliminating a batch of features.



(a) Illustration of γ_G, γ_{SP}, and γ_{SC}.



(b) MFE-k-G-slack-s (c) MFE-k-G-slack-m (d) MFE-k-SP-slack



(e) MFE-k-SC-slack-s (f) MFE-k-SC-slack-m

Fig. 3.1. Illustration of MFE-k and MFE-k-slack methods.

3.5.2 MFE-k-G-slack

We now choose the discrete *maximization* objective function for the elimination method MFE-k-G-slack as γ_G (from (3.8)) *minus a (C-weighted) sum of slackness values*. For each training sample $\underline{x}_n$, define a *single* slackness value ψ_n based on the sample's proximity to the class-pairwise boundary between the sample's class p_n and class c_n, or alternatively, we can define *multiple* slackness values $\psi_{n,q}$ based on the sample's proximity to *each* of the class-pairwise boundaries between the sample's class p_n and the $k-1$ other classes $q \neq p_n$. For the first of these two methods, MFE-k-G-slack-s, we choose the objective as:

$$\gamma_G - C \sum_{n \in \Omega} \psi_n \tag{3.14}$$

and meet the following slackness constraints upon choosing a single, correctly classified (candidate) *global anchor* $n_a \in A$:

$$d_{n,c_n} \geq d_{n_a,c_{n_a}} - \psi_n, \quad \psi_n \geq 0, \ \forall n \in \Omega \tag{3.15}$$

with slackness ψ_n defined as $\max(0, d_{n_a,c_{n_a}} - d_{n,c_n})$, where $d_{n_a,c_{n_a}}$ (which is computed as $\frac{g_{n_a,c_{n_a}}}{||w_{p_{n_a}} - w_{c_{n_a}}||}$) sets the (possibly violated) *multiclass margin* γ_G in (3.14). In the three-class example illustrated by Fig. 3.1(b) for the scenario where two dimensions remain upon candidate elimination of one of three features, the encircled sample 7 represents a particular choice for the (single) anchor, which sets γ_G (marked $\gamma_{2,1}$), relative to which the (positive) slacknesses (indicated in matching color) are calculated. All anchor-based computations for this method will be specified in detail, shortly. MFE-k-G-slack-s selects the pair $(m^{G-slack-s}, n_a^{G-slack-s})$ below that, post-elimination, maximizes (3.14) over all the discrete choices $\{(m, n_a)\}$. For the second method, MFE-k-G-slack-m, we choose the objective as:

$$\gamma_G - C \sum_{n \in \Omega} \sum_{q \in \bar{p}_n} \psi_{n,q} \tag{3.16}$$

and meet the following slackness constraints upon choosing the *global anchor* $n_a \in A$:

$$d_{n,q} \geq d_{n_a,c_{n_a}} - \psi_{n,q}, \quad \psi_{n,q} \geq 0, \ \forall n \in \Omega, \ \forall q \in \bar{p}_n \tag{3.17}$$

with slackness $\psi_{n,q}$ defined as $\max(0, d_{n_a,c_{n_a}} - d_{n,q})$, where $d_{n_a,c_{n_a}}$ sets the (possibly violated) *multiclass margin* γ_G in (3.16). Notice in Fig. 3.1(c) (which is for illustrating MFE-k-G-slack-m) that this figure differs from the above-discussed Fig. 3.1(b) in that *multiple* (positive) slacknesses are calculated for a sample for use by the algorithm (such as $\psi_{13,1}$ and $\psi_{13,2}$ for sample 13). All anchor-based computations for this method will be specified in detail in Sec. 3.5.2.2 shortly. MFE-k-G-slack-m selects the pair $(m^{G-slack-m}, n_a^{G-slack-m})$ that, post-elimination, maximizes (3.16) over all the discrete choices $\{(m, n_a)\}$. Notice that each of these two methods reduces to margin-maximizing (γ_G-maximizing) elimination (i.e. MFE-k-G, Sec. 3.4) for large enough C (assuming the data is separable).

3.5.2.1 MFE-k-G-slack-s algorithm pseudocode for linear and nonlinear kernel-based SVMs

0-1. Same as steps 0-1 in Sec. 3.5.1.1.

2. For each $(m \notin \mathcal{M}, n \in \Omega)$ pair, determine $c_n^{i,m} = \arg\min_{q \in \bar{p}_n} d_{n,q}^{i,m}$. For each $(m \notin \mathcal{M}, n_a \in A^{i,m})$ pair, to ensure zero slackness for the *global anchor* n_a (*i.e.*, $\psi_{n_a}^{i,m,n_a} = 0$) and to meet all the constraints in (3.15), compute $\psi_n^{i,m,n_a} = \max(0, d_{n_a,c_{n_a}^{i,m}}^{i,m} - d_{n,c_n^{i,m}}^{i,m})\ \forall n \in \Omega$.

3.1 Determine the outcome (m_i, n_a^i) of the discrete maximization shown below in (3.18), and eliminate feature m_i, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

$$\arg\max_{m \in \bar{\mathcal{M}}} \max_{n_a \in A^{i,m}} (d_{n_a,c_{n_a}^{i,m}}^{i,m} - C \sum_{n \in \Omega} \psi_n^{i,m,n_a}) \tag{3.18}$$

3.2 $i \rightarrow i+1$ and go to step 1.

3.5.2.2 MFE-k-G-slack-m algorithm pseudocode for linear and nonlinear kernel-based SVMs

This only differs from the pseudocode in 3.5.2.1 in steps 2 and 3.1.

2. For each $(m \notin \mathcal{M}, n_a \in A^{i,m})$ pair, determine $c_{n_a}^{i,m} = \arg\min_{q \in \bar{p}_{n_a}} d_{n_a,q}^{i,m}$, and, to ensure zero slackness for the anchor (*i.e.*, $\psi_{n_a,c_{n_a}^{i,m}}^{i,m,n_a} = 0$) and to meet all the constraints in (3.17) compute $\psi_{n,q}^{i,m,n_a} = \max(0, d_{n_a,c_{n_a}^{i,m}}^{i,m} - d_{n,q}^{i,m})\ \forall n \in \Omega\ \forall q \in \bar{p}_n$.

3.1 Determine the outcome (m_i, n_a^i) of the discrete maximization shown below in (3.19), and eliminate feature m_i, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

$$\arg\max_{m \in \bar{\mathcal{M}}} \max_{n_a \in A^{i,m}} (d_{n_a,c_{n_a}^{i,m}}^{i,m} - C \sum_{n \in \Omega} \sum_{q \in \bar{p}_n} \psi_{n,q}^{i,m,n_a}) \tag{3.19}$$

3.5.3 MFE-k-SC-slack

Next, choose the discrete *maximization* objective function for the elimination method MFE-k-SC-slack as γ_{SC} *minus a (C-weighted) sum of slackness values.* For a given class p, for each training sample $\underline{x}_n$ in class p, we can define a *single* slackness value ψ_n only based on the sample's proximity to the class-pairwise boundary between the sample's class p and class c_n (cf. Sec. 3.2), or alternatively, *multiple* slackness values $\psi_{n,q}$ based on the sample's proximity to *each* of the class-pairwise boundaries between the sample's class p and the $k-1$ other classes $q \neq p_n$. For either of these two methods, with each class having its own anchor for a given candidate feature for elimination, note that there are k anchors working collectively for each candidate, m. For the first of the two methods, MFE-k-SC-slack-s, we choose the objective as:

$$\gamma_{SC} - C \sum_{p=1}^{k} (\sum_{n \in \Omega_p} \psi_{n,c_n} + \sum_{n \in \Omega_{\bar{p}}} \psi_{n,p}) = \sum_{p=1}^{k} (\gamma_p - C(\sum_{n \in \Omega_p} \psi_{n,c_n} + \sum_{n \in \Omega_{\bar{p}}} \psi_{n,p})). \tag{3.20}$$

Associated with this objective, we meet the following slackness constraints in choosing a correctly classified (candidate) *classwise (p-versus-rest) anchor* $n_{a_p} \in A$ for each class $p \in P$ (note: n_{a_p} need not be in A_p): define $z_{n_{a_p}}$ for the anchor as $z_{n_{a_p}} \equiv c_{n_{a_p}}$ if the anchor is in p, or $z_{n_{a_p}} \equiv p$ otherwise.

$$d_{n,c_n} \geq d_{n_{a_p},z_{n_{a_p}}} - \psi_{n,c_n} \quad \forall p \in P,\ \forall n \in \Omega_p,\ \psi_{n,c_n} \geq 0 \tag{3.21}$$

$$d_{n,p} \geq d_{n_{a_p},z_{n_{a_p}}} - \psi_{n,p} \quad \forall p \in P,\ \forall n \in \Omega_{\bar{p}},\ \psi_{n,p} \geq 0 \tag{3.22}$$

with slackness ψ_{n,c_n} for $n \in \Omega_p$ defined as $\max(0, d_{n_{a_p},z_{n_{a_p}}} - d_{n,c_n})$ and $\psi_{n,p}$ for $n \in \Omega_{\bar{p}}$ defined as $\max(0, d_{n_{a_p},z_{n_{a_p}}} - d_{n,p})$, where $d_{n_{a_p},z_{n_{a_p}}}$ sets the (possibly violated) *classwise (p-versus-rest) margin* γ_p in (3.20). In the three-class example illustrated by Fig. 3.1(e) for the scenario where two dimensions remain upon candidate elimination of one of three features, the encircled samples 4, 7, and 16 represent a particular choice for the three anchors working collectively, which set the three classwise margins (marked γ_1, γ_2, γ_3, respectively), relative to which the (positive) slacknesses (indicated in matching color) are calculated – in this particular example there are no (positive) slacknesses relative to γ_3. All anchor-based computations for this method will be specified in detail in Sec. 3.5.3.1, shortly. MFE-k-SC-slack-s selects the pair $(m^{SC-slack-s}, n_{a_p}^{SC-slack-s})$ that, post-elimination, maximizes (3.20) over all the discrete choices $\{(m, \{n_{a_p}, p = 1, \ldots, k\})\}$. For the second method, MFE-k-SC-slack-m, we choose the objective as:

$$\gamma_{SC} - C \sum_{p=1}^{k} (\sum_{n \in \Omega_p} \sum_{q \in \bar{p}} \psi_{n,q} + \sum_{n \in \Omega_{\bar{p}}} \psi_{n,p}), \tag{3.23}$$

equivalent to

$$\sum_{p=1}^{k} (\gamma_p - C(\sum_{n \in \Omega_p} \sum_{q \in \bar{p}} \psi_{n,q} + \sum_{n \in \Omega_{\bar{p}}} \psi_{n,p})) \tag{3.24}$$

and meet the following slackness constraints and (3.22) upon choosing the *classwise (p-versus-rest) anchor* $n_{a_p} \in A$ for each class $p \in P$ (note: again, n_{a_p} need not be in A_p):

$$d_{n,q} \geq d_{n_{a_p},z_{n_{a_p}}} - \psi_{n,q} \quad \forall p \in P,\ \forall n \in \Omega_p,\ \forall q \in \bar{p},\ \psi_{n,q} \geq 0 \tag{3.25}$$

with slackness $\psi_{n,q}$ for $n \in \Omega_p$ defined as $\max(0, d_{n_{a_p},z_{n_{a_p}}} - d_{n,q})$, where $d_{n_{a_p},z_{n_{a_p}}}$ sets the (possibly violated) *classwise margin* γ_p in (3.24). Notice in Fig. 3.1(f) (which is for illustrating MFE-k-SC-slack-m) that this figure differs from the above-discussed Fig. 3.1(e) in that *multiple* (positive) slacknesses are calculated for a sample for use by the algorithm (such as $\psi_{13,1}$ and $\psi_{13,2}$ for sample 13) – in order to not crowd the figure, classwise margin and associated slacknesses are being illustrated for class 3 only. All anchor-based computations for this method will be specified in detail in Sec. 3.5.3.2 shortly. MFE-k-SC-slack-m selects the pair $(m^{SC-slack-m}, n_{a_p}^{SC-slack-m})$ that, post-elimination, maximizes (3.24) over all the discrete choices $\{(m, \{n_{a_p}, p = 1, \ldots, k\})\}$. Notice that each of these two methods reduces to margin-maximizing (γ_{SC}-maximizing) elimination (i.e. MFE-k-SC, Sec. 3.4) for large enough C (assuming the data is separable).

3.5.3.1 MFE-k-SC-slack-s algorithm pseudocode for linear and nonlinear kernel-based SVMs

0-1. Same as steps 0-1 in Sec. 3.5.1.1.

2. For each $(m \notin \mathcal{M}, p \in P, n \in \Omega_p)$ triplet, determine $c_n^{i,m} = \arg\min_{q \in \bar{p}} d_{n,q}^{i,m}$. For each $(m \notin \mathcal{M}, p \in P, n_{a_p} \in A^{i,m})$ triplet: first, determine $z_{n_{a_p}}^{i,m}$ (which is $c_{n_{a_p}}^{i,m}$ if the anchor is in p or is p otherwise); second, to ensure zero slackness for the anchor (*i.e.*, $\psi_{n_{a_p},z_{n_{a_p}}^{i,m}}^{i,m,n_{a_p}} = 0$) and to meet all the constraints in (3.21) and (3.22), compute $\psi_{n,c_n}^{i,m,n_{a_p}} = \max(0, d_{n_{a_p},z_{n_{a_p}}^{i,m}}^{i,m} - d_{n,c_n^{i,m}}^{i,m})\ \forall n \in \Omega_p$ and $\psi_{n,p}^{i,m,n_{a_p}} = \max(0, d_{n_{a_p},z_{n_{a_p}}^{i,m}}^{i,m} - d_{n,p}^{i,m})\ \forall n \in \Omega_{\bar{p}}$.

3.1 Determine the outcome $(m_i, \{n^i_{a_p}, p = 1, \ldots, k\})$ of the discrete maximization shown below in (3.26), and eliminate feature m_i, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

$$\arg\max_{m \in \bar{\mathcal{M}}} \sum_{p=1}^{k} \max_{n_{a_p} \in A^{i,m}} (d^{i,m}_{n_{a_p}, z^{i,m}_{n_{a_p}}} - C(\sum_{n \in \Omega_p} \psi^{i,m,n_{a_p}}_{n,c_n} + \sum_{n \in \Omega_{\bar{p}}} \psi^{i,m,n_{a_p}}_{n,p})) \tag{3.26}$$

3.2 $i \rightarrow i+1$ and go to step 1.

3.5.3.2 MFE-k-SC-slack-m algorithm pseudocode for linear and nonlinear kernel-based SVMs

This only differs from the pseudocode in 3.5.3.1 in steps 2 and 3.1.

2. For each $(m \notin \mathcal{M}, p \in P, n_a \in A^{i,m})$ triplet: first, determine $z^{i,m}_{n_{a_p}}$ (which is $c^{i,m}_{n_{a_p}}$ if the anchor is in p or is p otherwise), and, to ensure zero slackness for the anchor (*i.e.*, $\psi^{i,m,n_{a_p}}_{n_{a_p}, z^{i,m}_{n_{a_p}}} = 0$) and to meet all the constraints in (3.25) and (3.22) compute $\psi^{i,m,n_{a_p}}_{n,q} = \max(0, d^{i,m}_{n_{a_p}, z^{i,m}_{n_{a_p}}} - d^{i,m}_{n,q})\ \forall n \in \Omega_p\ \forall q \in \bar{p}$ and $\psi^{i,m,n_{a_p}}_{n,p} = \max(0, d^{i,m}_{n_{a_p}, z^{i,m}_{n_{a_p}}} - d^{i,m}_{n,p})\ \forall n \in \Omega_{\bar{p}}$.

3.1 Determine the outcome $(m_i, \{n^i_{a_p}, p = 1, \ldots, k\})$ of the discrete maximization shown below in (3.27), and eliminate feature m_i, i.e., $\mathcal{M} \rightarrow \mathcal{M} \cup \{m_i\}$.

$$\arg\max_{m \in \bar{\mathcal{M}}} \sum_{p=1}^{k} \max_{n_{a_p} \in A^{i,m}} (d^{i,m}_{n_{a_p}, z^{i,m}_{n_{a_p}}} - C(\sum_{n \in \Omega_p} \sum_{q \in \bar{p}} \psi^{i,m,n_{a_p}}_{n,q} + \sum_{n \in \Omega_{\bar{p}}} \psi^{i,m,n_{a_p}}_{n,p})) \tag{3.27}$$

3.6 MFE-k-Kesler

3.6.1 Kesler construction

The Kesler construction generates a set of higher-dimensional samples starting from the samples of a multiclass classification problem ($k > 2$) so that the multiclass LDF problem can be cast using a (higher-dimensioned) two-class LDF. We begin this section by stating the Kesler construction – additional description can be found in other sources, e.g. (20). For each original $M \times 1$ sample $\underline{x}_n$, $k-1$ Kesler samples $\underline{z}^n_q$ (with $q \in \bar{p}_n$) are constructed, each being a concatenation of k vectors that are each $(M+1)$-dimensional, with augmented vector $\underline{a}_n = [\underline{x}^{\mathrm{T}}_n\ 1]^{\mathrm{T}}$ the p_n-th vector, $-\underline{a}_n$ the q-th vector, and each of the other $k-2$ vectors given by an $(M+1)$-dimensional vector of zeroes. The total number of Kesler samples is thus $(k-1)N$. Denote $\underline{z}^n_q \equiv [z^n_{1,q}, \ldots, z^n_{k(M+1),q}]^{\mathrm{T}}$ (also just denoted $\underline{z}^n_q \equiv [z_1, \ldots, z_J]^{\mathrm{T}}$ with $J \equiv k(M+1)$). For example, if $k = 3$, for each $\underline{x}_n$ in class 2 the following two (i.e. $k-1$) Kesler samples $\underline{z}^2_q,\ q \in \{1, 3\}$ are constructed: $\underline{z}^2_1 = [-\underline{a}^{\mathrm{T}}_n\ \underline{a}^{\mathrm{T}}_n\ \underline{0}^{\mathrm{T}}]^{\mathrm{T}}$, $\underline{z}^2_3 = [\underline{0}^{\mathrm{T}}\ \underline{a}^{\mathrm{T}}_n\ -\underline{a}^{\mathrm{T}}_n]^{\mathrm{T}}$.

As previously shown by e.g. (20), (26), the Kesler construction can be used to express the multiclass problem as a one-class problem – in particular, form:

$$\underline{W} = [W_1, \ldots, W_{k(M+1)}]^{\mathrm{T}} \equiv [\ [\underline{w}^{\mathrm{T}}_1, b_1], \ldots, [\underline{w}^{\mathrm{T}}_k, b_k]\ ]^{\mathrm{T}}. \tag{3.28}$$

Thus let us denote a discriminant function example by $F(\underline{z}) \equiv \underline{W}^{\mathrm{T}}\underline{z}$. Note first that the constraints in (3.7) are captured by $\underline{W}^{\mathrm{T}}\underline{z}_j \geq 1 - \xi_j,\ j = 1, \ldots, (k-1)N$. Second, note that $||\underline{W}||^2 = \sum_{p \in P} ||\underline{w}_p||^2 + b^2_p$. Thus, it is easy to see that the multiclass SVM training problem in

(3.7) can be recast in terms of the augmented weight vector $\underline{W}$ and the Kesler samples via

$$\min_{\underline{W},\underline{\xi}} \frac{1}{2}||\underline{W}||^2 + C\sum_{j=1}^{J}\xi_j \quad s.t. \;\; \xi_j \geq 0, \;\; F(\underline{z}_j) \geq 1-\xi_j, \;\; j=1,\ldots,(k-1)N. \tag{3.29}$$

For a Kesler sample $\underline{z}$, let T_m denote the Kesler feature subset $\{m, m+(M+1), \ldots, m+(k-1)(M+1)\}$ with values determined by the *original* feature m – only two of these k are actual occurrences of original feature m in any given Kesler sample, as the remaining $k-2$ are 0. Thus, denoting:

$$\Delta^t_{n,q} \equiv z^n_{t,q} W_t, \;\; q \in \bar{p}_n, \tag{3.30}$$

[4]

$$\tilde{\Delta}^m_{n,q} \equiv \sum_{t \in T_m} \Delta^t_{n,q}, \;\; q \in \bar{p}_n, \tag{3.31}$$

it becomes obvious that

$$\tilde{\Delta}^m_{n,q} = \delta^m_{n,q}, \;\; q \in \bar{p}_n. \tag{3.32}$$

3.6.2 MFE-k-Kesler

The idea of our MFE-k-Kesler feature elimination method for the multiclass case is to construct Kesler samples $\underline{z}$ from the original samples $\underline{x}$ and utilize the equivalence between (3.7) and (3.29). Specifically, (3.29) is a *one-class* SVM problem (all Kesler samples on the same side of the hyperplane $\underline{W}$). Accordingly, for feature elimination consistent with the optimization in (3.29), we apply two-class MFE (cf. Ch. 2, (3)) on the one-class nature of Kesler samples $\underline{z}$, which is a valid approach because, the one-class problem is a two-class problem where all samples (the Kesler samples) are in class $+1$ (and none in class -1) and lie on a side (the same side) of a hyperplane (i.e. the Kesler hyperplane) $\underline{W}$ given by (3.28). At elimination step i, MFE-k-Kesler eliminates the *Kesler* feature t_i whose elimination yields the largest remaining (positive) *Kesler margin* (i.e. minimum distance from the Kesler samples to $\underline{W}$). After the elimination sequence of Kesler features is obtained using this criterion, the elimination sequence of original features is obtained from it as follows. An original feature m is eliminated when all $t \in T_m$ are eliminated – this generates a complete elimination sequence of original features.

3.7 Results

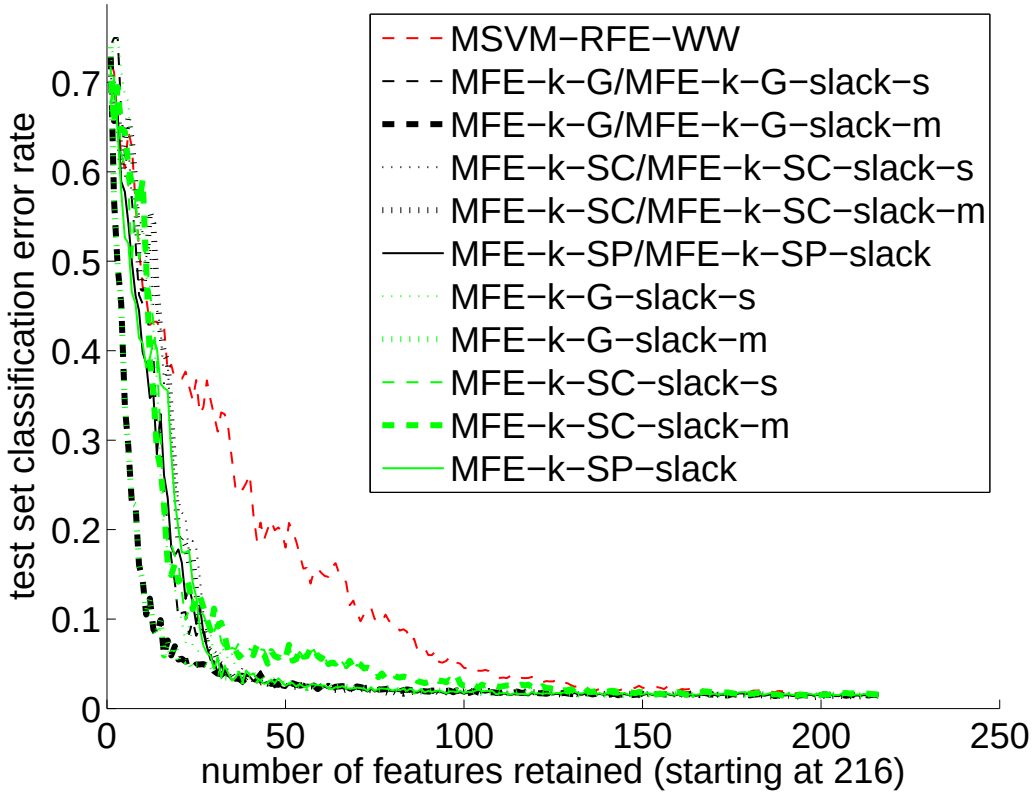
We performed experiments to compare our multiclass MFE-k methods with multiclass RFE-based methods. For experimental procedure for the initial classifier training, and experimental procedure for feature elimination, we followed the approach discussed earlier for the two-class case (cf. Sec. 2.4.1 and 2.4.2). Also as in the two-class case, to continue the elimination process after loss of separability, we define the "hybrid" MFE-k/MFE-k-slack methods (such as MFE-k-G/MFE-k-G-slack-m), where, simply, MFE-k (such as MFE-k-G) is used for the steps where the data is separable (including the initial step) and the corresponding MFE-k-slack method (such as MFE-k-G-slack-m corresponding to MFE-k-G) is used for the other steps.

For the case of the linear kernel, test set classification error rate curves for our MFE-k methods and the RFE-based MSVM-RFE-WW method are shown in Fig. 3.2 for two UC Irvine data sets. All curves shown are averages over 10 trials – and for each trial, the initial SVM was a training set separator. For our three MFE-k methods MFE-k-G, MFE-k-SP, and MFE-k-SC, we show that the first two had a tie for best performance, under our "slack-m" approach for slackness

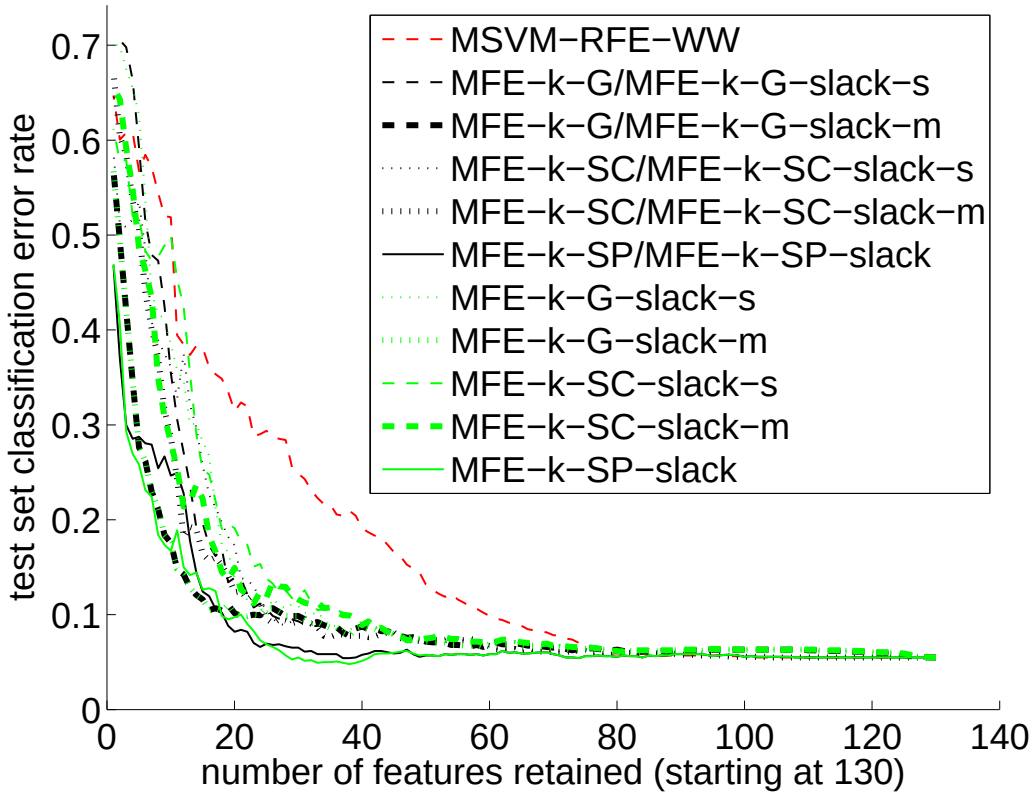
[4] The product on the RHS contains also a multiplication by $y_{\underline{z}^n_q}$, not being shown as it is 1 $\forall n$.

(as opposed to the less rigorous "slack-s") – notice in the figure that this is true for both the hybrid method (e.g. MFE-k-G/MFE-k-G-slack-m) and the solely slackness-based method (e.g. MFE-k-G-slack-m). We also show that our above three "basic MFE-k" methods achieved much better generalization performance (lower test set error rate curve) than MSVM-RFE-WW. A generalization performance comparison between these MFE-k methods and MFE-k-Kesler is not available in this thesis and can be generated in future work – however, in Fig. 3.3(a), we show that MFE-k-Kesler achieved much better generalization performance than MSVM-RFE-WW.

In conclusion, for feature elimination in the multiclass SVM case, our multiclass MFE methods that we advocate are: MFE-k-G/MFE-k-G-slack-m, MFE-k-SP/MFE-k-SP-slack, as well as MFE-k-Kesler. Our multiclass method that we advocate the most at this time is MFE-k-G/MFE-k-G-slack-m, because e.g. 1) it requires less computation than MFE-k-SP/MFE-k-SP-slack which had similar generalization performance, and 2) both the number of samples and the number of features are increased approximately k-fold when MFE-k-Kesler is used.

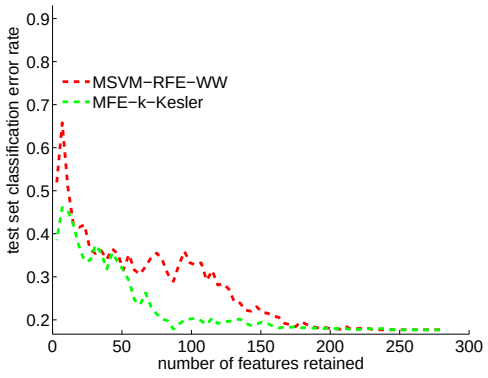


(a) *UCI mfeat* data set with 216 features; linear kernel



(b) *UCI dermatology* data set with 130 features; linear kernel

Fig. 3.2. Average test set classification error rate, (a)-(b), for two separable low-to-intermediate dimensional UC Irvine data sets (Sec. 3.7). Initial number of features is stated on the x-axis.



(a) *UCI arrhythmia* data set; linear kernel

Fig. 3.3. Average test set classification error rate for the *UCI arrhythmia* data set; linear kernel (Sec. 3.7).

Chapter 4

MRI brain image processing and analysis of Alzheimer's disease and its onset

Note for ADNI: *The author, of this thesis chapter i.e. manuscript, is Yaman Aksu, for the Alzheimer's Disease Neuroimaging Initiative*.*

**Data used in preparation of this article were obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database (http://www.loni.ucla.edu/ADNI). As such, the investigators within the ADNI contributed to the design and implementation of ADNI and/or provided data but did not participate in analysis or writing of this report. A complete listing of ADNI investigators can be found at:*
http://www.loni.ucla.edu/ADNI/Collaboration/ADNI_Authorship_list.pdf
The ADNI was launched in 2003 by the National Institute on Aging (NIA), the National Institute of Biomedical Imaging and Bioengineering (NIBIB), the Food and Drug Administration (FDA), private pharmaceutical companies and non-profit organizations, as a $60 million, 5-year public-private partnership. The primary goal of ADNI has been to test whether serial magnetic resonance imaging (MRI), positron emission tomography (PET), other biological markers, and clinical and neuropsychological assessment can be combined to measure the progression of mild cognitive impairment (MCI) and early Alzheimer's disease (AD). Determination of sensitive and specific markers of very early AD progression is intended to aid researchers and clinicians to develop new treatments and monitor their effectiveness, as well as lessen the time and cost of clinical trials. The Principal Investigator of this initiative is Michael W. Weiner, M.D., VA Medical Center and University of California - San Francisco. ADNI is the result of efforts of many co-investigators from a broad range of academic institutions and private corporations, and subjects have been recruited from over 50 sites across the U.S. and Canada. The initial goal of ADNI was to recruit 800 adults, ages 55 to 90, to participate in the research – approximately 200 cognitively normal older individuals to be followed for 3 years, 400 people with MCI to be followed for 3 years, and 200 people with early AD to be followed for 2 years.

4.1 Introduction

The dementing illness Alzheimer's disease (AD), and the state of transition between normal aging and AD referred to as mild cognitive impairment (MCI), continue to be widely studied. MCI includes people with memory impairment without meeting dementia criteria. Annually an estimated 10-15% of people with MCI are diagnosed with AD (54). Moreover, even prior to clinical symptoms, larger increases in structural brain abnormality have been found in people with MCI compared to normal ("Control group") people, via retroactive evaluation of serial (longitudinal) MRI scans (16). Furthermore, AD diagnosis is not considered definitive without confirming AD-characteristic pathologies such as the debated amyloid deposits detectable at autopsy. Motivated by reasons such as the above, there has been much recent analysis of both AD and MCI, to better understand the disease onset and progression to aid diagnosis and effective treatment development. A substantial aim is to develop methods with promising clinical value, which include 1) methods that can effectively aid diagnosis only based on a single (first) visit for the person, 2) methods that can effectively predict which MCI individuals are likely to convert,

especially imminently, to AD, and 3) methods for treatment monitoring. Such methods can be driven by seeking "biomarkers" – these include feature selection methods, as discussed in Ch. 1. Alternatively, methods may attempt to diagnose without seeking biomarkers – these include the use of SVM classification based on the full feature set (without feature selection). In Ch. 4, we experiment with both of these approaches, using SVMs and MFE.

For all aims mentioned above, structural MRI analysis may prove useful as it addresses many complexities of AD and MCI such as the following. First, even at an early, clinically normal stage of AD or MCI, brain atrophy may exhibit complex patterns spatially distributed to many brain regions (10; 23; 25) – due to cell loss there may be cortical thinning (47; 66) and ventricle dilation and gaping (11; 59), volumetric and shape changes in the hippocampus and entorhinal cortex (15; 19; 64), temporal lobe shrinkage (58), and other morphological changes. Second, since atrophy patterns may be more subtle at the early disease stage (11; 16; 25) and may change during disease progression, quantifying patterns early is a big challenge. Third, patterns attributable to normal aging are a confounding factor in the search for patterns that characterize disease (57).

Consider a population of AD/MCI/Control brain MR 3d images segmented into numerous (e.g. 100) intricate regions defined by a brain anatomy atlas – such images would enable numerous region-specific volumetric measurements, useful towards detecting brain atrophy typical for AD/MCI subjects. A second useful image type would be 3d "volumetric density" images – one for gray matter and one for white matter – where, this time we are referring to an image type that both 1) has been considerably registered across the population (i.e. "spatial normalization" into a single spatial frame of reference) and 2) has voxel values that represent how much "local volume" *for the corresponding tissue* (e.g. gray matter) in the original (pre-registration) multi-tissue image had to be dilated or contracted by the registration to fit into that density image voxel. A possible advantage of such density images over the region-segmented image is the much larger number of volumetric measurements (i.e., one per voxel) that it provides per brain. In this thesis we are using the HAMMER registration method and software tool for generating such tissue density images – a separate density image is generated for each of three "tissues" (gray matter, white matter, ventricles), and as mentioned above the voxel values of this set of three images by definition sum to the pre-registration brain volume. Typically, including the case of this thesis, populations of such image types (e.g. region-segmented, or density) are not commonly readily available and instead only the original images, which were generated by the MR scanner itself (e.g. T_1-weighted images), are available as image populations. In this thesis we use HAMMER for generating both of the above two useful image types – we elaborate on this method and tool in Appendix C. Sec. 4.2 discusses our image processing pipeline that includes also non-HAMMER components that are necessary for generating some types of intermediate images from the original (T_1-weighted) MR images, including "tissue-segmented" images. Tissue-segmented images, which are a required HAMMER input, are images that consist of the following segments: gray matter, white matter, ventricles, (non-ventricle) cerebrospinal fluid (CSF), (non-anatomy) background.

As noted by a number of prior works e.g. (11), statistical and predictive power, and validity, of an AD analysis method producing spatio-temporal "patterns" of brain regions, are of much interest. In the domain of AD analysis using MRI, a number of prior works utilized pattern recognition – those that utilized SVM classifiers in particular, as we do in this chapter, include (16; 51; 71). These particular prior studies are significantly notable also because, in order to capture interactions among effects in regions throughout the brain, without making a priori assumptions about specific regions to be measured, each of these methods jointly analyzed voxels (or regions) spanning the entire brain, whereas a number of other studies have focused on specific regions of interest (ROIs), especially the regions mentioned above such as hippocampus and ventricles.

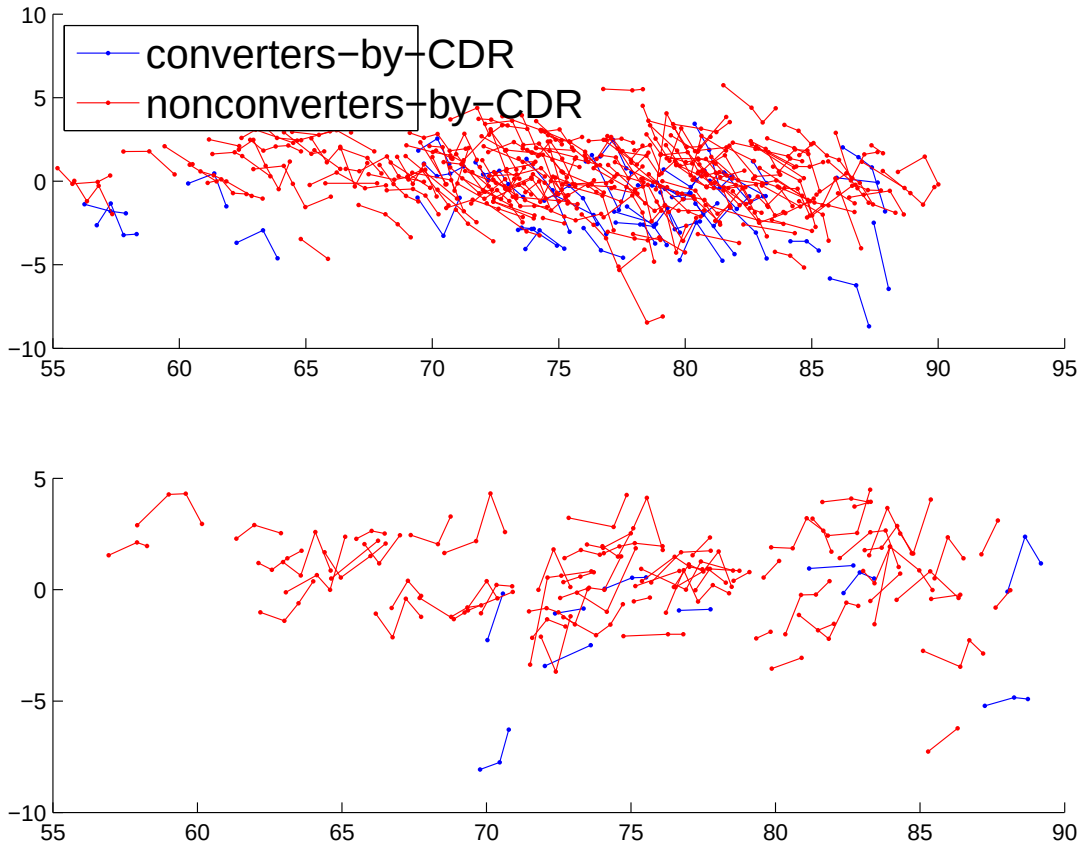
The data used in the experiments in this chapter were obtained from the ADNI database (2), for which three "ground-truth" clinical labels, assigned by clinicians, were available: AD, MCI, Control. This label may be derived from multiple criteria, including CDR (Clinical Dementia Rating) whose possible values are: *0=none, 0.5=questionable, 1=mild, 2=moderate, 3=severe*. ADNI aims to recruit and follow 800 research participants in the 55-90 age range: approximately 200 elderly people to serve as Controls (Normal Controls), 400 people with MCI, and 200 people with AD. The "ground-truth" label (AD, MCI, or Control) was assigned in ADNI at first visit – however, at a later visit there is no stated conversion from one ground-truth label to another, and thus, for example we do not know if or when (at which visit) a participant who is non-AD at first visit converts to AD. On the other hand, available to us are visit-specific numerical data, including structural MR (T_1-weighted) images and non-image data such as CDR score and age, enabling cross-sectional analysis (across a population) and/or longitudinal analysis (across time).

In prior work on AD analysis with MRI data e.g. (51), for some classification experiments, the MCI population was broken up into two subpopulations (classes), according to whether, over time, an MCI participant's CDR score of 0.5 stayed the same or increased. Herein we refer to this subgrouping approach as the "conversion-by-CDR" approach[1]. The concept of conversion-by-CDR should be considered with great caution because there is *very significant* AD-MCI overlap with respect to the CDR value of 1 *and even 0.5* – more specifically, considering the ADNI database as an example (an example with a very large subject population), the majority of the hundreds of AD participants start (at first visit) at 0.5 and stay at 0.5 at the later visits, while at the same time all (or almost all) of the hundreds of MCI participants start at 0.5, with the vast majority of these MCIs staying at 0.5 at the later visits. Very relevant to this overlap issue, a significant contribution of this thesis is our following novel approach for detecting (possible) conversion to AD, driven by image-based information, rather than by a single, non-image-based measurement such as CDR. Upon building a Control-AD classifier (using only AD and Control subjects), we use it to classify all available longitudinal 3d images[2] of the *training MCI subjects*, so as to utilize their distances to this single Control-AD boundary from one visit to the next. An example illustration of these *person-specific* trajectories of longitudinal phenotype scores (positive for Control-like and negative for AD-like) is given by Fig. 4.1(a) – phenotype score vs. age is plotted, with each set of connected dots being a separate person. People who are converters-by-CDR are shown in different color than people who are nonconverters-by-CDR. Also, solely for clearer viewing, the subject population is further broken up in Fig. 4.1(a) into a top and bottom graph, with the top showing people whose score was less for final visit than first visit, and vice versa for the bottom graph – notice that there are many more people who become more AD-like than Control-like over time. As illustrated by Fig. 4.1(b), we then fit a line (using least-squares) on a person-by-person basis. Consider the following four fitted-score (fs) categories:

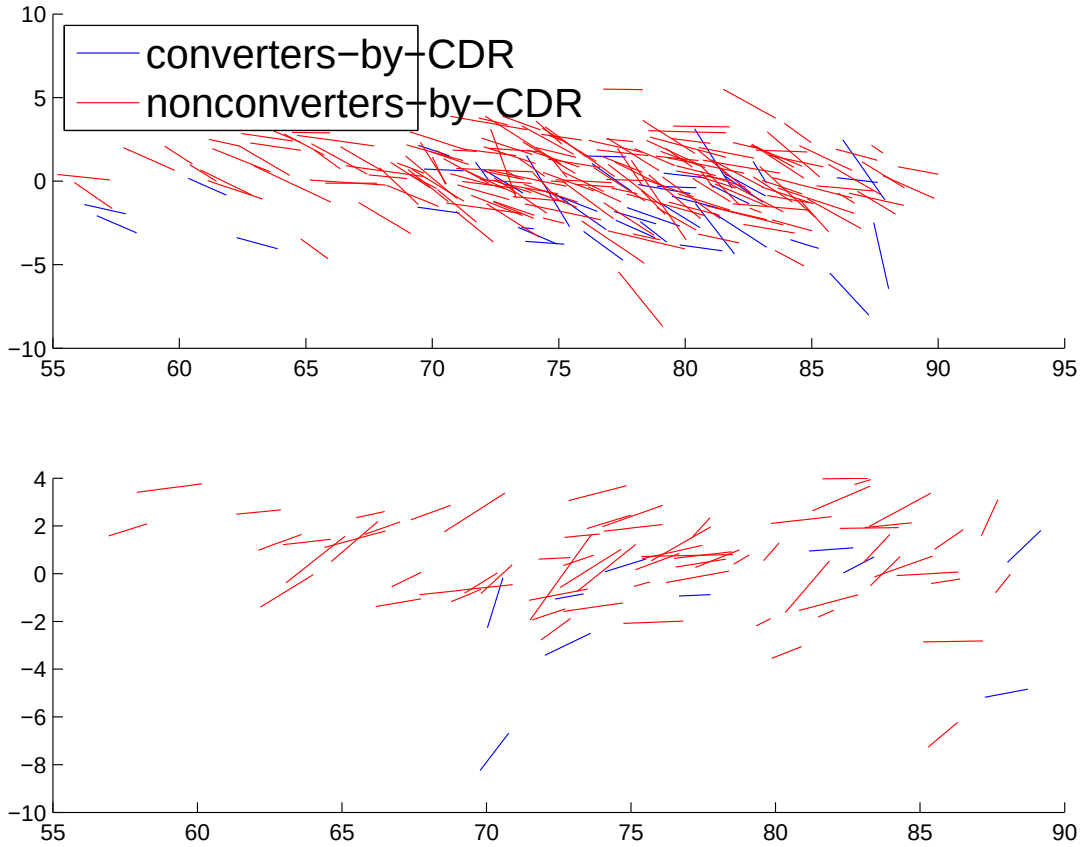
1. fs starts and ends positive.
2. fs starts and ends negative.
3. fs starts positive and ends negative.

[1] While this term says "conversion", note that it does not mean an *actual conversion* to AD which requires assignment of the participant by a clinician to the AD label at the later visit – recall that information is not available. We will refer to the two classes formed by the conversion-by-CDR approach as "converters-by-CDR (cc)" and "nonconverters-by-CDR (ncc)", e.g. the "MCIcc" and "MCIncc" classes.

[2] Here we are simply saying "image" to refer to *our particular feature set for the image* which is not necessarily a set of voxel values, e.g. some experiments in this chapter use volumes of brain regions as features.



(a) Phenotype score vs. age, for MCI converters-by-CDR (MCIcc) and nonconverters-by-CDR (MCIncc).



(b) Fitted phenotype score vs. age, for MCI converters-by-CDR (MCIcc) and nonconverters-by-CDR (MCIncc).

Fig. 4.1. Trajectory results for ROI-based Classifier 1R.

4. fs starts negative and ends positive.

For all types of features in our experiments herein, we consistently found that at least $90-95\%$ of our *training MCI population* fell into the first three of the above four fs categories. Since these three represent almost the entirety of the MCI population, we can legitimately define categories 2 and 3 together as "the first of two classes of MCI subjects" and category 1 as "the second of the two classes of MCI subjects", based on which we can then build a new type of two-class classifier. More specifically, since subjects in category 1 are more Control-like than AD-like *throughout* their visits, we can declare they define (belong to) a "nonconverters-by-trajectory' (MCInct) class, and likewise, since subjects in category 2 and 3 are more AD-like than Control-like by the time of their *final visit*, we declare they define (belong to) a "converters-by-trajectory" (MCIct) class.[3] Note, very importantly, that when building this new classifier, *each training sample would come from the first visit alone, not from multiple visits.* We shall refer to this approach as "conversion-by-trajectory" – clearly, its strength is two-fold:

1. Previously unseen (test) patients would be classified (diagnosed) *based on their first visit alone*, i.e. this is an *early diagnosis* approach.[4]

2. Unlike in the conversion-by-CDR approach, the two MCI subpopulations involved are being defined using image-based information, without using CDR. This is an advantage because, first, a rise in a person's CDR from 0.5 to 1, utilized by the above alternative conversion-by-CDR approach, may not be a strong indicator on which to define MCI subpopulations, and second, assigned CDR values and their change over time may be considerably noisy.

Note that the legitimacy of the conversion-by-trajectory approach stems from the fact that the *test MCI subjects* are not being used when building either of the above two in-tandem classifiers. We will show experimentally that the conversion-by-trajectory approach outperforms the conversion-by-CDR approach.

A second major novel aspect, and significant contribution, of the AD/MCI analysis in this thesis is that MFE is used as the feature selection method, whereas prior works (16; 51) (mentioned above) utilized RFE which was shown in this thesis (Ch. 2) to be consistently outperformed by MFE in a comprehensive evaluation, i.e. for both separable and nonseparable data, for a variety of data set examples for both linear and kernel-based classification.

A third notable aspect for this thesis is that the experiments are utilizing a much larger population of ADNI subjects and their longitudinal images ($\approx$ 2000 images in total) than we have seen in prior works.

Sec. 4.2 discusses our image processing pipeline. Sec. 4.3 discusses our extensive set of experiments and gives experimental results and comparisons. Sec. 4.4 gives the chapter conclusions.

4.2 Brain MR image processing pipeline

For the image processing pipeline used by this thesis, let us begin by giving a short overview of the pipeline steps. The input (3d) image is a T_1-weighted MR image of the head. As first step, this image is coarsely aligned with a reference atlas (aka Atlas1) using rigid-body

[3] Since category 2 by definition starts and ends more AD-like than Control-like, there is no "conversion" taking place for it per se, but we do include this category together with category 3 in our conveniently named "converters"-by-trajectory class because they end AD-like just as category 3 does.

[4] As a reminder, note that the person to be diagnosed must be excluded from the population used to construct (train) the classifier(s) to be used for the diagnosis – this is a pattern classification design requirement.

registration, whereby the AC (Anterior Commisure) landmark in the brain also becomes placed at the origin. Then, non-brain anatomy is removed from the aligned head images. The resulting brain-only 3d image is tissue-segmented into the five segments required by HAMMER: white matter (WM), gray matter (GM), ventricles, (non-ventricle) CSF, (non-anatomy) background. Lastly, a second atlas (aka Atlas2, an atlas by MNI (Montreal Neurological Institute) distributed with HAMMER), serving as the reference atlas to be used by HAMMER registration, is input into HAMMER along with the five-segment image, to generate the above-mentioned density 3d image (RAVENS map) and the 3d region-segmented image – a separate RAVENS map is generated for each of three "tissues": GM, WM, ventricles. The sum of voxel values across these three RAVENS images (cf. Sec. 4.1) is on the order of 10^6 and varies across people – to remove person-specific brain volume as a bias, the set of three RAVENS images is normalized such that each person would have the same volume. As elaborated at the end of this section, the normalized RAVENS images are then smoothed spatially using a 5mm FWHM Gaussian filter.

For Atlas1, we used the MNI/ICBM (International Consortium for Brain Mapping) atlas, resampled to 1mm isotropic voxel dimensions. The alignment to Atlas1 is performed using FSL's linear image registration tool FLIRT (41). The subsequent non-brain removal is performed using FSL's brain extraction tool BET (60). The subsequent tissue-segmentation step consists of three substeps. The first substep is the commonly used approach of tissue segmentation into four (of the above-mentioned five) segments, with ventricles and CSF together labeled as CSF. This segmentation is performed using FSL FAST (76). The second substep is the use of the *labelVN* tool in HAMMER for (automatically) segmenting CSF into ventricles and (non-ventricle) CSF. The third and final substep is to process the resulting five-segment 3d image with our *fv (fill ventricles)* algorithm whose purpose is to automatically remedy the problem where the above ventricle segmentation by HAMMER (which is a deformation process for matching to the ventricles of Atlas2) is often incomplete as a consequence of a subject's anatomy being too different from the atlas. This problem is a common scenario as AD/MCI patients often have very enlarged and distorted ventricles due to atrophy. We describe *fv* in Appendix A.2 – its underlying assumption is that the ventricles that HAMMER does find, which we viewed in numerous brains, are *reliable* despite being incomplete.[5]

HAMMER generates the RAVENS maps by performing a deformation between the subject's tissue-segmented (five-segment, 3d) image and the corresponding tissue-segmented (five-segment, 3d) Atlas2 image[6]. Also available in the Atlas2 image set is the already region-segmented version of this atlas with approximately 100 anatomical regions. This availability is a main reason we are using this particular atlas - using this atlas (i.e., both its region-segmented and five-segment versions) and HAMMER, the pipeline is able to produce a region-segmented image from an individual's five-segment image.

In our above pipeline steps used in this thesis, as a matter of fact the use of the MNI/ICBM atlas (for the initial linear registration), our *fv* algorithm, and the RAVENS normalization and smoothing are the only steps that were not included in our original pipeline which was developed and delivered by us to PSU Hershey CNMRR in 2006 and 2007 which gave rise to our collaboration with Dr. D. Bigler at CNMRR for co-developing a pipeline improved with new capabilities named STAMPS (7) which is discussed in Appendix A.3 – notice in the Appendix that several of the types of output images that STAMPS can readily generate are not used by this thesis. For our above-mentioned pipeline steps in this thesis, we created and used a new and customized version

[5] A known HAMMER property is that it conservatively adjusts the amount of deformation depending on the task so as to control the amount of noise in the result.

[6] Since the region-segmented atlas distributed with HAMMER (Atlas2) had much better segmentation quality than the tissue-segmented (five-segment) atlas distributed with HAMMER (Atlas2), the former was used, by Dr. D. Bigler, to create a replacement for the latter by simply forming region unions. The tissue-segmented Atlas2 that we used was this replacement image.

of STAMP/STAMPS, which we shall simply refer to as "STAMPY/STAMPYS". The differences between STAMPS and STAMPYS can be understood via comparing our above description of our processing steps with STAMPS steps described in (7). In understanding these differences, it is also important to understand that STAMPYS stemmed from the fact that STAMPS has several major limitations. We have met these critical challenges – note some example differences between STAMPS and STAMPYS described below:

1. STAMPS' interface to the cluster (i.e., supercomputer), described in (7), is both complex and assumes that the user has much system administration control, at will, over how the job processes are to be distributed to the cluster nodes. This approach was designed by our pipeline collaborator Dr. D. Bigler specifically with the PSU Hershey CNMRR cluster in mind, and overrode the interface approach of our above-mentioned pre-STAMPS pipeline which is simple so as to make it functional on the PSU UP HPC clusters (on which we do not have system administration control). The UP HPC clusters, together providing at least ten times the processing power of the CNMRR cluster alone, have been essential to our work because, as anticipated, it has been necessary to process more than a thousand ADNI 3d images – not just a single time but rather at will, with each image taking several hours to process. Here, a big challenge was the need to process this many images *and* to do so in a timely manner (again, not once but as many times as the need arises). While more than a thousand *samples* is of course not necessary for a classification experiment, processing $1000 - 2000$ *images* with the pipeline has been necessary, as follows. As Sec. 4.3 will demonstrate concretely, the actual number of samples per class that $1000 - 2000$ *images* provide us can often be as low as 100, or even 50, because the thousand is a union that accounts for i) three subject categories (AD, MCI, Control), ii) up-to-6 visits (i.e. up-to-6 images) per subject[7], and iii) a wide age range whereas we often require 1-to-1 age-matching (cf. Sec. 4.3.3) across classes in our classification experiments herein for AD analysis.

2. As stated in (7), a hard-coded STAMPS step that precedes the rigid-body registration thresholds the input T_1-weighted image. While (7) essentially suggests this hard-coded step is innocuous so long as T_1-weighted images have sufficient signal-to-noise ratio, we have discovered that this step often destroys ventricle voxels by setting them to 0, causing subsequent pipeline images (such as the region-segmented image and the RAVENS ventricle map) to be incorrectly generated. Thus, in our pipeline in this thesis (STAMPY), the thresholding step is removed.

3. While (7) essentially declares that STAMPS has a functioning FSL-based segmentation option apart from its SPM/VBM-based segmentation option, as it turned out FSL-based segmentation was actually not fully implemented into STAMPS even though it was indeed implemented into our original pre-STAMPS pipeline. As we discussed above (Sec. 4.2), STAMPYS implements and uses FSL-based segmentation – this is mainly because SPM/VBM segmentation in the STAMPS design (or a third segmentation option different from SPM/VBM and FSL) was not made feasible on the PSU UP HPC clusters. The infeasibility was due to the fact that it was required in STAMPS to assign the segmentation task to distributed Matlab computing despite the incompatibility of this approach with the restrictions in the UP HPC configuration, such as HPC's *dmatlab* (restrictive, custom, front-end component for controlling distributed Matlab computing) and HPC's particular method of distributing Matlab licenses.

[7] Sec. 4.3 will describe that our diagnosis procedure utilizes *training data* from up-to-6 visits (i.e. up-to-6 images) per subject (rather than relying on a simpler classification experiment procedure) so as to aim for sufficient specificity and sensitivity towards diagnosis.

4. The automated pipeline STAMPYS contains our abovementioned *fv* tool which enables it to remedy the abovementioned ventricle segmentation problem in an automated way, whereas the automated pipeline STAMPS does not contain a similar (automated) tool, as mentioned by footnote 6 in (7). This is a very significant limitation in STAMPS when the image population that needs to be processed is large.

We now elaborate on the RAVENS smoothing mentioned above. The focus of RAVENS is to preserve volume on a tissue-by-tissue basis through the registration process. When creating RAVENS, HAMMER relaxes the amount of warping to combat noise inherent in the nonlinear registration so as to properly detect (and encode) brain atrophy. Consequently, there is considerable inter-person variability in RAVENS images, which is commonly mitigated by smoothing the RAVENS (or the normalized RAVENS) images. Since RAVENS smoothing is not yet made a part of the pipeline (in STAMPS or STAMPYS), we performed the smoothing afterwards, in an automated way.

4.3 Results

4.3.1 ADNI data

Herein we only use "processed" images present in the ADNI database, i.e. images that have undergone ADNI's image correction which is described at the ADNI website.[8] The number of Control, MCI, and AD participants used in our analysis were approximately 180, 300, and 120, respectively. For each Control we used, we only used the participant's initial visit with CDR=0 – the reason is that after the initial visit approximately 10% of these Control participants have CDR=0.5 (just as some MCI and even AD participants do) and thus, these non-initial visits for Controls, if used, would have introduced a confounding effect into the analysis which we choose to simply avoid in this particular case since 180 Control images is sufficient. For MCI and AD participants, from the discussion in Sec. 4.1 on our methods it should be clear that we used multiple visits.

4.3.2 ROI-based and voxel-based experiments

We performed two types of experiments: ROI-based (R) and voxel-based (V). In the R case, the features are 101 normalized region volumes obtained from the region-segmented images, wherein a region's volume is normalized by dividing it by the total volume of all regions. In the V case, which is much higher dimensional with more than 20,000 features, the features are the voxel intensities of a smoothed RAVENS map. As a reminder from Sec. 4.1, while each of these two types of features is volumetric information, V may produce, as this thesis will explore, more accurate generalization due to both the number and finer granularity of its volumetric measurements (i.e., one feature per voxel) that it provides per brain. For both experiment types, we will build several SVM classifiers (named Classifier 1, Classifier 2, etc.) that differ solely by class design (i.e., how the particular population for the classifier is constructed or defined) – we will refer to the ROI-based and voxel-based instances of Classifier 1 as Classifier 1R and Classifier 1V, respectively. For the V case only, we will additionally perform a statistical paired t-test for comparison, using the same sample set as the SVM classifier we are comparing it with.

[8] See www.loni.ucla.edu/ADNI/Data/ADNI_Data.shtml. ADNI image correction steps include Gradwarp, N3, and scaling for gradient drift. As suggested by the above ADNI website, we sought image files having "N3" and "scaled" in the file name.

4.3.3 Procedure for initial classifier training for AD/MCI/Control data

For AD/MCI/Control data, for training of the initial SVM classifier (i.e., the classifier prior to *potentially* subsequently applying feature selection on it), we used the experimental procedure specified below – notice that fundamentally it is an extension of our standard procedure (Sec. 2.4.1) so as to incorporate new aspects appropriate for this data, such as the treatment of age as a confounding variable. For a given participant, image data may be available for multiple visits – however, note that the procedure is restrictive in that every "sample" (exemplar) input for training the SVM classifier shall not combine data from multiple visits (for a given participant) – consequently each sample does only have a single age associated with it. Also note that the procedure does ensure that a given participant contributes a sample (or samples) to either the training set or the test set but not both – there is a *set of training participants*, from which the *set of training images (samples)* are obtained, and likewise there is a *set of test participants*, from which the *set of test images (samples)* are obtained.

Let P_c be the set of participants in class $c \in C \equiv \{1, \ldots, k\}$ where k is the number of classes.[9]

Step 1. Prepare training set and test set: For each $c \in C$, P_c is randomly split 90%-10% into a non-heldout (training) set $P_{c,nh}$ and a heldout (test) set $P_{c,h}$.

Step 2 (optional). Age-match: As per the single-age-per-sample restriction stated above, we produce subsets $P_{c,nh,a} \subset P_{c,nh}$ $\forall c \in C$ such that each sample in $P_{c,nh,a}$ is "age-matched" (i.e. age-separated by no more than one whole year) to a unique sample in each of the other (i.e. $C \backslash c$) classes. Establishing this tight age correspondence between classes in the training set contributes to mitigating the confounding effect of age (with a second contributor being our "adjust feature values for age" step discussed below), as well as makes the training data readily suitable for a statistical paired t-test (with which we compare our methods). Upon matching, to utilize any non-matched (leftover) samples of the non-heldout set, we move them to the heldout set. Subsequently, as in Ch. 2 (and in (3)), let X and $\bar{X}$ denote the non-heldout (training) and heldout (test) set of *samples* (as opposed to *participants*) respectively.

Step 3. Remove constant features: Features constant across the non-heldout samples (in X) are removed from all (non-heldout and heldout) samples.

Step 4. Adjust feature values for age: This step uses a basic approach that aims to remove the confounding effect of age from the data, prior to classification. For each feature m, first, model the feature value as a function of age a by computing the least-squares-fit line $l_m(a) \equiv \alpha_1 a + \alpha_2$ through the points $(a_i, x_{i,m})$ where the index i traverses the non-heldout samples, and for ith such sample (with age a_i) the value of feature m is $x_{i,m}$. Next, using this line (i.e., α_1 and α_2 computed for feature m), for each (heldout or non-heldout) sample adjust $x_{j,m}$ (with j denoting the sample index) by subtracting $l_m(a_j) \equiv \alpha_1 a_j + \alpha_2$.

Step 5. Normalize: For each feature m, compute the minimum value $\min_{nh}$ and the maximum value $\max_{nh}$ for the feature across non-heldout samples, and normalize the feature's value v_j for every (heldout and non-heldout) sample j by setting it to $(v_j - \min_{nh})/(\max_{nh} - \min_{nh})$. This normalizes the non-heldout set's feature values to the [0,1] range, and the heldout set's feature values to the [0,1] range.

Step 6. Train a classifier as described by Steps 2-4 of our "Experimental Procedure for the Initial Classifier Training" in Ch. 2 (and in (3)).

[9] While we are stating the procedure for suitability with any number of classes, note that set C is $\{\pm 1\}$ for our experiments herein.

4.3.4 Classifier 1: Control-AD123 classifier

Using the "initial classifier training procedure" (cf. Sec. 4.3.3), we build the Control-AD classifier discussed in Sec. 4.1, herein referred to as Classifier 1, using only a single, CDR=0 image for Control subjects and all available CDR$\geq$ 1 images for "AD123" subjects (who are, by definition, the AD subjects with at least one visit having a CDR of 1, 2, or 3). Note that AD subjects who start and stay at CDR=0.5, and thus are perhaps too MCI-like *throughout their visits*, are excluded from this Control-AD classifier. Recall from Sec. 4.3.2 that we will refer to distinct classifiers "Classifier 1R" and "Classifier 1V", corresponding to the ROI-based and voxel-based cases for Classifier 1 respectively.

Since all MCI participants in our analyses herein have a CDR of 0.5 at initial visit, one interesting question for early diagnosis of AD is as follows.

Q1: Using (classifying) an MCI participant's initial visit alone, how well does Classifier 1 predict whether this MCI participant will experience a rise in CDR (i.e. a "conversion-by-CDR", suggesting possible conversion to AD)? In our experiments below, this is one of the questions we will address.

4.3.4.1 Classifier 1R: features are ROI-based

Several results are shown for Classifier 1R in Table 4.1. With the test set being somewhat class-balanced with respect to the number of samples (76 Controls, 57 ADs), and the two class-specific test set misclassification rates being close to each other (0.07 for Controls, 0.12 for ADs), in this particular experiment the overall test set misclassification rate is a fairly good indicator *for this entire test population* – note that this rate (0.09) is low, and thus we can say Classifier 1R performs quite well and confidently proceed to characterizing our *MCI data* with this classifier throughout this chapter.

Note that answering Q1, for Classifier 1R, involves a special classification – an AD-Control classifier that was trained without using people with MCI is used to classify people with MCI. Accordingly, for this special classifier, we can define "misprediction" intuitively as an MCIncc person classified as AD or an MCIcc person classified as Control. Then, for Classifier 1R, the answer to Q1 is: not well at all. Table 4.1 states that the "misprediction" rate for 252 MCI participants (209 MCIncc, 43 MCIcc) is high (0.36), and thus a better *early diagnosis system* is needed than 1R alone.

Since 1R gave high accuracy for Control and AD data, we now highlight the brain regions that MFE selected based on this classifier. Listed under 1R in Table 4.1, the circle on the MFE test set error graph indicates that ten out of 101 brain regions were selected using MFE, listed above the graph (in order of elimination). Entorhinal cortex and hippocampus, which are well-published biomarkers for AD, were found by MFE to be the 4th and 5th most discriminating for the disease among the 101 brain regions. As in Ch. 2 (and (3)), the graphs show MFE performed well: the MFE/MFE-slack test set error curve trend is again flat; a large percentage of features is eliminated without a big change in classification accuracy relative to the initial (pre-elimination) classifier; and MFE again outperformed RFE.

4.3.4.2 Classifier 1V: features are voxel-based

Recall that the features used by our voxel-based SVM classifiers herein, including this first voxel-based classifier named Classifier 1V, are the voxel intensities of the three types of smoothed and subsampled RAVENS maps (GM, WM, ventricles). For such classifiers, we created the RAVENS subsample via skipping five RAVENS voxels along each of the three axes – as described in Appendix B.0.1, with the (I, J, K) associated with skips of five being equal to $(6, 6, 6)$, each RAVENS map is exhaustively broken up into 216 subsamples. In conjunction with the exhaustive subsampling approach, in Appendix B.0.1 we introduced the "hierarchical

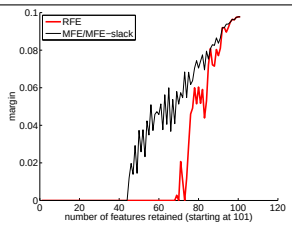
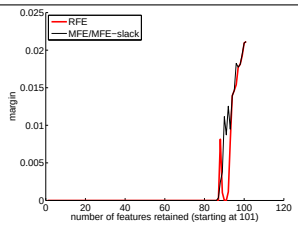
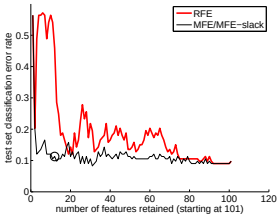
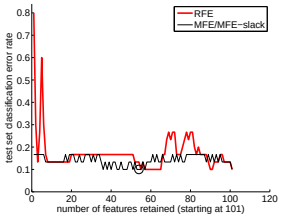
	CLASSIFIER 1R Control-AD123	**CLASSIFIER 2Ra MCIncc-MCIcc**	**CLASSIFIER 2Rb MCIncc-MCIcc**
features	101	101	101
·training	204 (102,102)	273 (228,45)	88 (44,44)
·age-matching	yes	no	yes
·test	(71|5 0.07, 7|50 0.12)	(24|1 0.04, 2|3 0.4)	(140|69 0.33, 3|3 0.5)
·error	0.09	0.13	0.33
MCI (ncc,cc)	(209,43) 0.36	-	-
MFE regions	middle frontal gyrus R lateral front-orbital gyrus R medial frontal gyrus L lingual gyrus L postcentral gyrus L entorhinal cortex L hippocampal formation L thalamus L amygdala L amygdala R		
margin			
error			

Table 4.1. Results for ROI-based Classifiers 1R and 2R.

MFE (H-MFE)" feature elimination procedure that is simply a series of steps where each step simply re-groups multiple feature subsets provided by the previous step (and then applies MFE on each resulting group), with the initial step's feature subsets being the subsamples' individual MFE-selected feature sets. However, herein we do not give results for H-MFE, i.e. the Classifier 1V results we will discuss are only based on a single one subsample among the 216. Several 1V results are shown in Table 4.2. First, the test set misclassification rate (0.08) of the initial (pre-elimination) classifier is low, and very close to the rate found above for the ROI-based case (0.09, cf. Sec. 4.3.4.1), and thus, as in the ROI-based case we are able to confidently proceed to characterizing our *MCI data* with this Control-AD classifier (1V) throughout this chapter. Second, the graphs (Table 4.2) show MFE again performed well: the MFE/MFE-slack test set error curve trend is again flat; a large percentage of features is eliminated without a big drop from the accuracy of the initial classifier. In addition, a slow drop in training set margin during MFE is observed. Third, Fig. 4.2 shows the brain regions selected by MFE (i.e. regions discriminating between AD and Control).

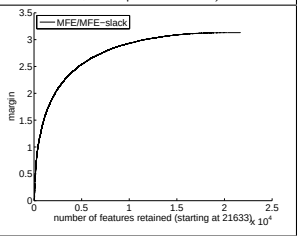
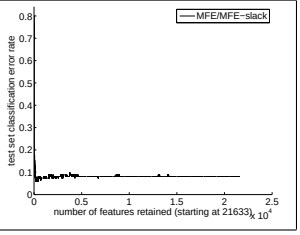
	CLASSIFIER 1V Control-AD123
features	21,633
·training	144 (72,72)
·age-matching	yes
·test	(105|4 0.04, 7|20 0.26)
·error	0.08
MCI (ncc,cc)	(168|62 0.27, 27|22 0.55) 0.32
margin	
error	

Table 4.2. Results for voxel-based Classifier 1V.

Next, we built Classifier 1Vb whose results are shown in Table 4.3. Classifier 1Vb differs from Classifier 1V in that 1Vb only used the gray matter RAVENS image and a single subsample among a total of *eight* (resulting in 84,627 features) whereas 1V had used all three RAVENS images and a single subsample among a total of *216* (resulting in 21,633 features). As shown by the 1Vb results in Table 4.3, in this particular experiment instance the use of a smaller subsampling factor (producing a larger number of features) did not lower the test set misclassification rate.

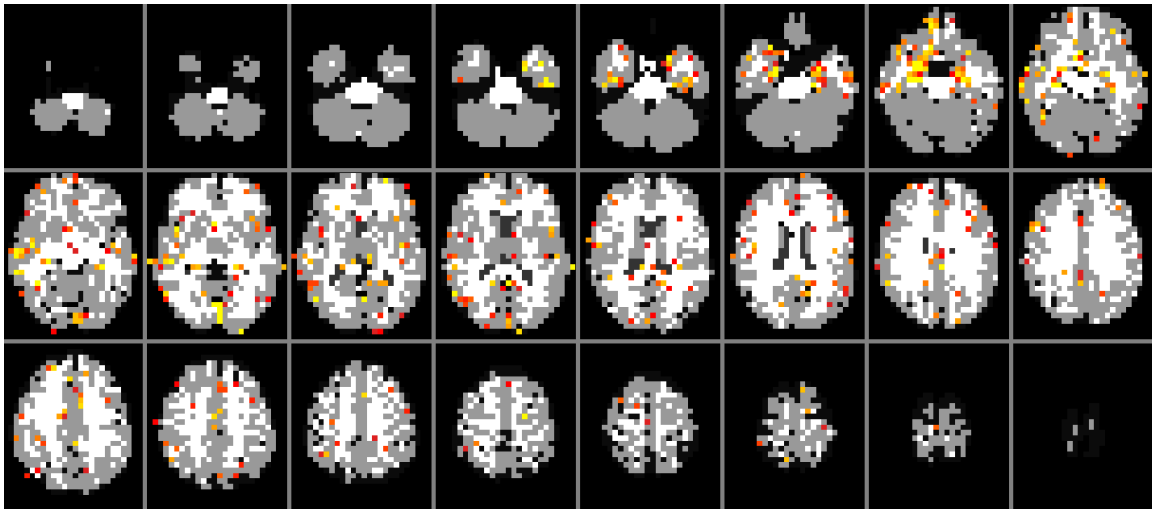
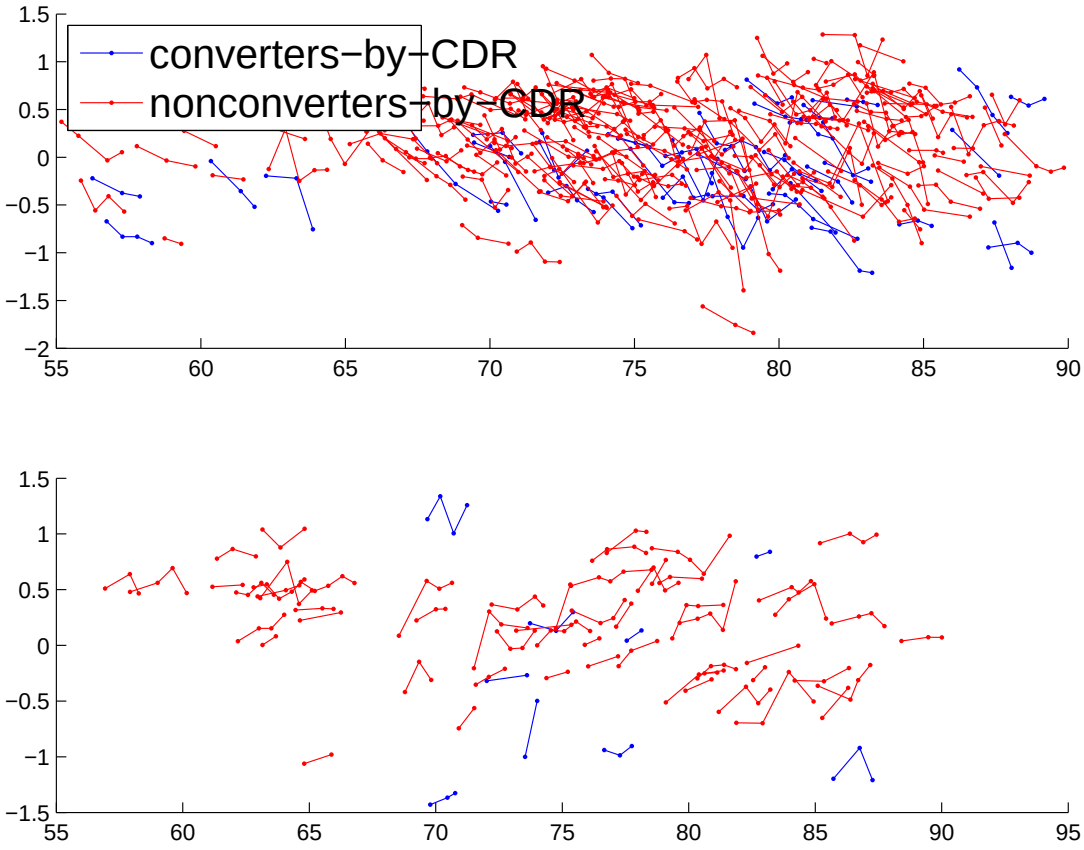


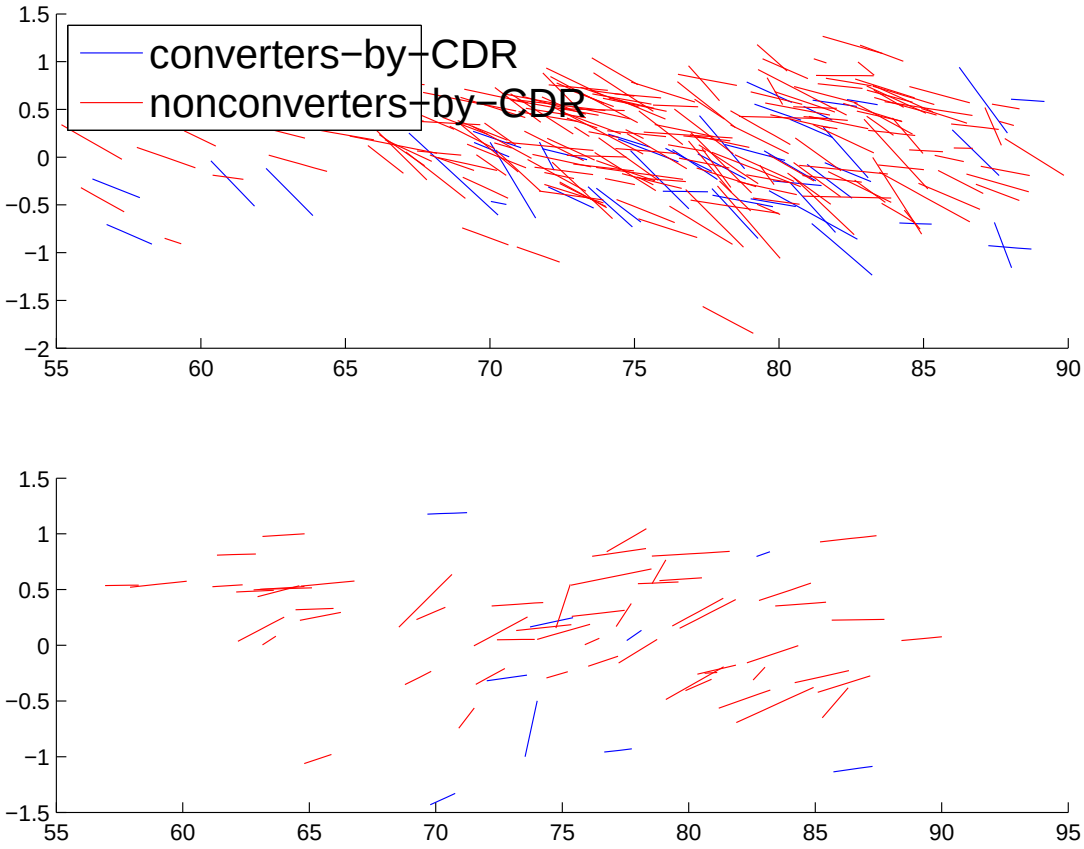
Fig. 4.2. Regions found by MFE with Classifier 1V.

	CLASSIFIER 1Vb Control-AD123
features	84,627
·training	144 (72,72)
·age-matching	yes
·test	(108|1, 10|17)
·error	0.08

Table 4.3. Results for voxel-based Classifier 1Vb.



(a) Phenotype score vs. age, for MCI converters-by-CDR (MCIcc) and nonconverters-by-CDR (MCIncc).



(b) Fitted phenotype score vs. age, for MCI converters-by-CDR (MCIcc) and nonconverters-by-CDR (MCIncc).

Fig. 4.3. Trajectory results for voxel-based Classifier 1V.

4.3.5 Classifier 2: MCIncc-MCIcc (nonconverters-by-CDR vs. converters-by-CDR)

Sec. 4.3.4 showed that Classifier 1 alone is not sufficient for predicting *(possible) conversion from MCI to AD in the conversion-by-CDR sense* using an MCI participant's initial visit alone. However, recall that Classifier 1 did give high accuracy (91% for ROI-based, 92% for voxel-based) for Control and AD data, and thus the poor prediction performance of Sec. 4.3.4 for MCI resulting from *CDR-driven categorization* is attributable to: a) the CDR values being noisy and/or b) a switch for an MCI participant from 0.5 to 1 obviously not necessarily meaning an actual conversion from MCI to AD. Recall that in fact half of all participants clinically labeled as AD at initial visit have a CDR of 0.5 at that visit, just as all MCI participants do at their initial visit, which, again, stresses the inherent complexity of the AD/MCI/Control classification (and diagnosis) problem. Consequently, to more closely investigate the classification problem for MCI participants in particular, and the role of CDR for that task, as well as to evaluate our MFE method for this particular case, first we built an MCI-specific classifier using the conversion-by-CDR approach (cf. 4.1 of prior works such as (16; 51; 71), which we refer to as Classifier 2 – recall that this classifier is a stand-alone classifier not built in-tandem with results from another classifier such as Classifier 1. After evaluating Classifier 2 in this section, in the next section we will build Classifier 3, our novel classifier which, unlike Classifier 2, is built in-tandem with Classifier 1 so as to utilize the concept of conversion-by-trajectory.

4.3.5.1 Classifier 2R: features are ROI-based

As shown in Table 4.1, Classifiers 2Ra and 2Rb were built, without and with age-matching respectively. With age-matching, notice from the figure that the training set for 2Rb is class-balanced but only 44 training subjects per class were available (which is less than half the number used for Classifier 1R), whereas, without age-matching for 2Ra the number of available class 1 subjects (MCI "nonconverters-by-CDR") rises to 228 – notice from the figure that the training set for 2Ra is thus very class-imbalanced. Test misclassification rate was quite high for both of these classifiers – for Classifier 2Rb it was high for *both classes*, whereas in 2Ra it was high only for class 2 (MCI "converters-by-CDR"). Since the features used in Classifier 2R are the same ones used in Classifier 1R which did give high accuracy in that case (for Control and AD participants), the fundamental weakness of Classifier 2R is perhaps not the quality of the features but rather that Classifier 2R is driven by an MCI categorization based on CDR measurements which may be noisy. To address these open questions, and the fact that these results show that Classifier 2R alone, just like Classifier 1R alone, is not sufficient for predicting conversion from MCI to AD using an MCI participant's initial visit alone, we are motivated to proceed to our above-mentioned diagnosis approach wherein, again, we will use Classifier 1R to obtain trajectories to be used to build Classifier 3R (which will thus serve in-tandem with Classifier 1R).

4.3.6 Classifier 3: MCInct-MCIct (nonconverters-by-trajectory vs. converters-by-trajectory)

We would like to only use *a single visit* for a *test MCI individual* in order to diagnose whether or not the individual will convert to AD. Note that this aim is very closely related to Question 1. Based on Sec. 4.1, our procedure for this particular type of diagnosis is as follows:

Step 1. Build Classifier 1 using the procedure in Sec. 4.3.3.

Step 2. Generate two MCI subgroups: Use Classifier 1 to calculate a phenotype score for each of the available longitudinal images of the population of *training* MCI subjects, and break up this population into the MCIct and MCInct subgroups as described by Sec. 4.1.

Step 3. Build Classifier 3: Again using the training procedure of Sec. 4.3.3, build the MCInct-MCIct classifier, herein referred to as Classifier 3.

Step 4. To diagnose a test individual, classify him/her using Classifier 3. If he/she is classified as MCIct, the diagnosis is that he/she will convert to AD; otherwise the diagnosis is that he/she will not convert to AD.

Although this particular diagnosis procedure is clearly suitable for MCI individuals, let us now discuss whether it is also suitable for *any* individual. For people with AD and healthy people, clearly all that the procedure can really declare is whether the person is (a) more like MCI people who will convert to AD or (b) more like MCI people who will not convert to AD. Therefore, clearly it is not a complete procedure for declaring whether a person *will convert to AD* (or *has AD*). However, the procedure *is* useful for also a *healthy* (non-MCI) person because clearly one should certainly expect almost all healthy people to be classified as MCInct rather than MCIct. On the other hand, it is not reasonable to expect *almost all* people with AD to be classified as MCIct by this procedure. One reason is, as discussed in Sec. 4.1 a very large percentage of all AD and MCI subjects (*half* of all AD and *big majority* of all MCI) do exhibit similar characteristics over time such as a non-varying CDR of 0.5 over time. Nevertheless, in Sec. 4.3.6.1 and Sec. 4.3.6.2 we will evaluate the above diagnosis procedure for people with AD (in addition to MCI and healthy people).

Note again that the above diagnosis procedure essentially serves to predict whether a person will be AD *in the future*. To diagnose whether a person has AD *at present*, recall that Classifier 1 can be used but it is, again, by design clearly more reliable for people with AD and healthy people than people with MCI. For an improved diagnosis procedure to address this issue, an additional component can be introduced into our above diagnosis procedure. The component would employ a new classifier, which would be similar to Classifier 3 in that it would be based on the trajectories, but dissimilar in that this time it would produce the two MCI subgroups based on a new criterion that takes into account also the trajectories of the *training* individuals with AD. The new procedure's AD diagnosis can then be combined with the AD diagnosis of Classifier 1 (which classifies an individual as Control or AD) – for this, basic voting or a more sophisticated ensemble learning approach can be employed. Alternatively, *more than two* subgroups can be produced, to then build a multiclass classifier.

4.3.6.1 Classifier 3R: features are ROI-based

Notice from Table 4.4 that we have built Classifiers 3Rc and 3Rd to correspond to Classifiers 2Ra and 2Rb we discussed above in Sec. 4.3.5, using the same features – that is, notice that Classifier 3Rd was trained under age-matching using 44 samples per class (just as in Classifier 2Rb), and Classifier 3Rc due to no age-matching was trained with several times as many class 1 samples as the 45 class 2 samples (just as in Classifier 2Ra). Comparing test set misclassification rates on a class-by-class basis, we see that Classifier 3Rd's rates were 0.28 for class 1 and 0.29 for class 2, whereas the rates of its counterpart Classifier 2Rb were much higher: 0.33 for class 1 and 0.5 for class 2. Similarly comparing Classifier 3Rc to 2Ra, we see that the rates 0.08 and 0.25 of Classifier 3Rc are overall better than the rates of 0.04 and 0.4 of Classifier 2Ra. Thus we can perhaps conclude that our novel in-tandem classification approach, involving two classifiers (1R and 3R), has successfully introduced an improvement towards solving the problem of predicting conversion from MCI to AD using an MCI participant's initial visit alone. In fact, to see that the magnitude of the improvement is perhaps bigger than implied by the above rates alone, let us now consider our Classifiers 3Ra and 3Rb also shown in Table 4.4. These two classifiers, with larger training set sizes than 3Rc and 3Rd, were generated for our in-tandem classification procedure – more specifically: 1) to obtain the 44 age-matched sample pairs with

which to build 3Rd, we simply omitted 54 of the 98 age-matched pairs of 3Rb (cf. Table 4.4)[10], and 2) to obtain the 118 class 1 samples and the 45 class 2 samples with which to build 3Rc, we simply took all 118 class 1 samples of 3Ra (cf. Table 4.4) and omitted 99 of the 144 class 2 samples of 3Ra (cf. Table 4.4). With that said, notice in Table 4.4 that our approach is able to achieve a test classification accuracy of 80%. (51) reported this same number for their methods – this is a remarkable comparison between their methods and our method because their method is actually based on a more sophisticated pre-classification image preparation procedure wherein special features are extracted from RAVENS maps. In other words, while our features may be less powerful, the above-demonstrated strength of our novel in-tandem classification procedure is one way we are overcoming that limitation. Moreover, additionally using MFE, we may achieve further improvements – for example, the test set error graphs for both our Classifiers 3a and 3b show that the error rate drops below 0.2 under MFE/MFE-slack elimination.

In Table 4.4, the "test Control" row indicates that the big majority of Control subjects were consistently classified as MCInct rather than MCIct, as anticipated by the above discussion in Sec. 4.3.6. The "test AD123" row indicates that the big majority of AD123 subjects were consistently classified as MCIct rather than MCInct – an expected result because recall that these particular AD subjects have high CDR *throughout* their visits. Within the group of AD subjects who had a CDR of 0.5 at first visit, the majority was classified the same as the ADs above who had come in with a high CDR (i.e. the majority was classified as the MCI class that is eventually on the AD side of the Control-AD boundary). Within this group of AD subjects, some had gone to a higher CDR at a subsequent visit (i.e. the "test ADcc" row) whereas others had stayed at 0.5 (i.e. the "test ADncc" row). Those with a subsequently high CDR resembled, at first visit, those who had *started* with a high CDR more than did those who stayed at 0.5, with respect to the above classification. In one experimental example (Classifier 3Ra) considering those who had stayed at 0.5, while 72% were classified as the converting MCI class (a relatively small number, perhaps due to it being difficult to discriminate between ADs and MCIs staying at 0.5), the percentage is higher for AD subjects who either start at a high CDR (e.g. 94%) or eventually increase to a high CDR (e.g. 77%).

[10] After sorting the 98 age-matched sample pairs by age, first we omitted every other pair and then we omitted 5 additional pairs so that 44 pairs would remain, so as to maintain the uniformity of the age distribution of the remaining pairs.

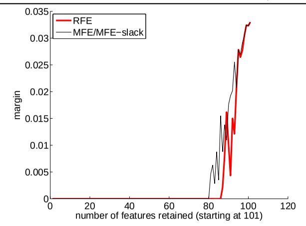
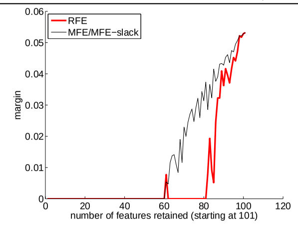
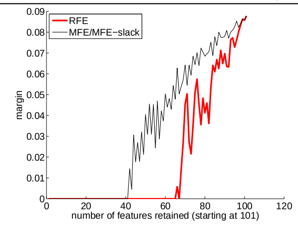
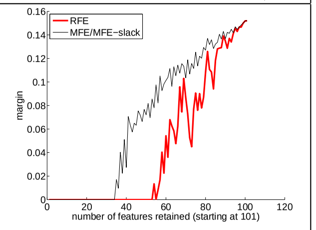
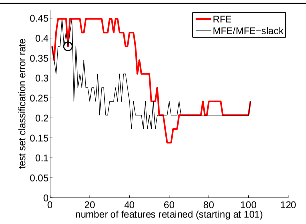
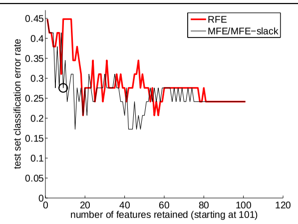
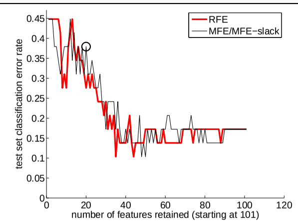
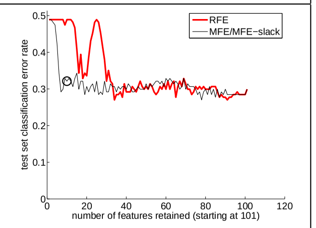
	CLASSIFIER 3Ra MCInct-MCIct	CLASSIFIER 3Rb MCInct-MCIct	CLASSIFIER 3Rc MCInct-MCIct	CLASSIFIER 3Rd MCInct-MCIct
features	101	101	101	101
·training	262 (118,144)	196 (98,98)	163 (118,45)	88 (44,44)
·age-matching	no	yes	no	yes
·test	(9|4 0.3, 2|14 0.13)	(10|3 0.23, 4|12 0.25)	(12|1 0.08, 4|12 0.25)	(48|19 0.28, 20|50 0.29)
·error	0.2	0.24	0.17	0.28
test ADncc	11|29	11|29	18|22	15|25
test AD123	1|15	2|14	3|13	3|13
test ADcc	3|10	4|9	6|7	4|9
test Control	59|17	61|15	68|8	62|14
margin				
error				

Table 4.4. Results for ROI-based Classifier 3R. Part 1 of 2 – see Table 4.5 for part 2.

	CLASSIFIER 3Ra MCInct-MCIct	CLASSIFIER 3Rb MCInct-MCIct	CLASSIFIER 3Rc MCInct-MCIct	CLASSIFIER 3Rd MCInct-MCIct
MFE regions	lateral ventricle L globus palladus L superior frontal gyrus L medial front-orbital gyrus L postcentral gyrus R amygdala R	precentral gyrus R frontal lobe WM R subthalamic nucleus L posterior limb of internal capsule* hippocampal formation L postcentral gyrus R medial occipitotemporal gyrus L corpus callosum amygdala R superior temporal gyrus R angular gyrus L *inc. cerebral peduncle R	globus palladus L putamen R brain stem subthalamic nucleus L posterior limb of internal capsule* middle frontal gyrus L inferior frontal gyrus R medial front-orbital gyrus L superior occipital gyrus L hippocampal formation L lingual gyrus R medial frontal gyrus R medial occipitotemporal gyrus L anterior limb of internal capsule R corpus callosum amygdala R cuneus R *inc. cerebral peduncle L	middle frontal gyrus R superior frontal gyrus R caudate nucleus R postcentral gyrus R amygdala L amygdala R angular gyrus L fornix R

Table 4.5. Results for ROI-based Classifier 3R. Part 2 of 2 – see Table 4.4 for part 1.

4.3.6.2 Classifier 3V: features are voxel-based

We built Classifier 3Vb (with tens of thousands of voxel-based features) which is similar to the above Classifier 3Rb (with 101 ROI-based features) in that both classifiers were built with a similarly sized training set of age-matched pairs (95 pairs (Table 4.6) and 98 pairs, respectively). As shown in Table 4.6, the voxel-based classifier has achieved a much lower test set misclassification rate (0.13) than its ROI-based counterpart (0.2). Note that 0.13 is lower than also the MFE-achieved less-than-0.2 rate of the ROI-based case discussed above in Sec. 4.3.6.1. Thus, the use of voxel-based features along with our trajectory-based approach (i.e. Classifier 3Vb) has successfully introduced further improvements (beyond the improvements in Sec. 4.3.6.1) towards solving the problem of predicting conversion from MCI to AD using an MCI participant's initial visit alone. Moreover, one or both of the following two ways may further reduce 0.13: 1) use of MFE on 3Vb, 2) recall that 3Vb used a single subsample among the 216 RAVENS subsamples we created, and thus, based on our past experimental experience with H-MFE we expect that the utilization of all 216 subsamples via the use of H-MFE (cf. Appendix B.0.1) to either maintain the error rate or lower it further. In this way, the biomarkers would be obtained through a consideration of *all available* brain voxel intensities (i.e. RAVENS voxel-based features).

By contrast, recall that the alternative "conversion-by-CDR" approach for prediction for MCI patients (cf. 4.1), proposed by prior works e.g. (51), is a relatively simplistic approach that considers the group of MCI participants as two subgroups *based only on whether CDR stays as 0.5 over time or increases.* For prediction for MCI patients based on the conversion-by-CDR approach, (51) reported a maximum accuracy of 81.5%, with accuracy fluctuating in the $75-80\%$ range, whereas, for prediction for MCI patients (i.e. the same aim) our system (Classifier 3Vb) achieved 87% accuracy with voxel-based features (and earlier our Classifier 3R achieved 80% accuracy with ROI-based features, which increased to 83% using MFE (cf. Sec. 4.3.6.1 and Table 4.4)).

	CLASSIFIER 3Vb MCInct-MCIct
features	22,063
·training	190 (95,95)
·age-matching	yes
·test	(15|2 0.12, 2|12 0.14)
·error	0.13
test ADncc	19|28
test AD123	0|5
test ADcc	5|8
test Control	94|15

Table 4.6. Results for voxel-based Classifier 3V.

As indicated by the results in the "test AD" and "test Control" rows in Table 4.6, our earlier comments for such rows (cf. Sec. 4.3.6.1) again apply.

4.3.7 Comparison with statistical paired t-test using SPM5

Given two populations (groups) with members (samples) paired with each other, a statistical paired t-test is useful for finding significant statistical differences between (the means of) the two groups. While statistical tests such as the paired t-test are useful for finding *group*

differences, pattern classification methods, such as SVMs herein, have the additional benefit of attempting to help *diagnose a particular individual*. One way that a statistical test can be compared with SVM-based classification is with respect to prediction performance on a test set. One approach would be to build an SVM classifier, say Classifier 4, using the brain voxels (features) that were selected as significant by the paired t-test. Using the same training and test populations for both this t-test and one of our earlier SVM classifiers (in the above sections), the test set misclassification rate of Classifier 4 can be compared with that of (a) one of our classifiers (given in a previous section above) and/or (b) one of our MFE results (given in a previous section above). To that end, we performed a paired t-test on the population of 72 age-matched Control-AD pairs of Table 4.2. Below we will compare this t-test's results with our above pre- and post-elimination results for Classifier 1V. We will perform the voxelwise paired t-test on the entirety of the smoothed RAVENS map without subsampling, whereas recall that our above voxel-based results for our methods were based on a single subsample of the smoothed RAVENS maps. In this way, the comparison is somewhat favorably biased towards the t-test.

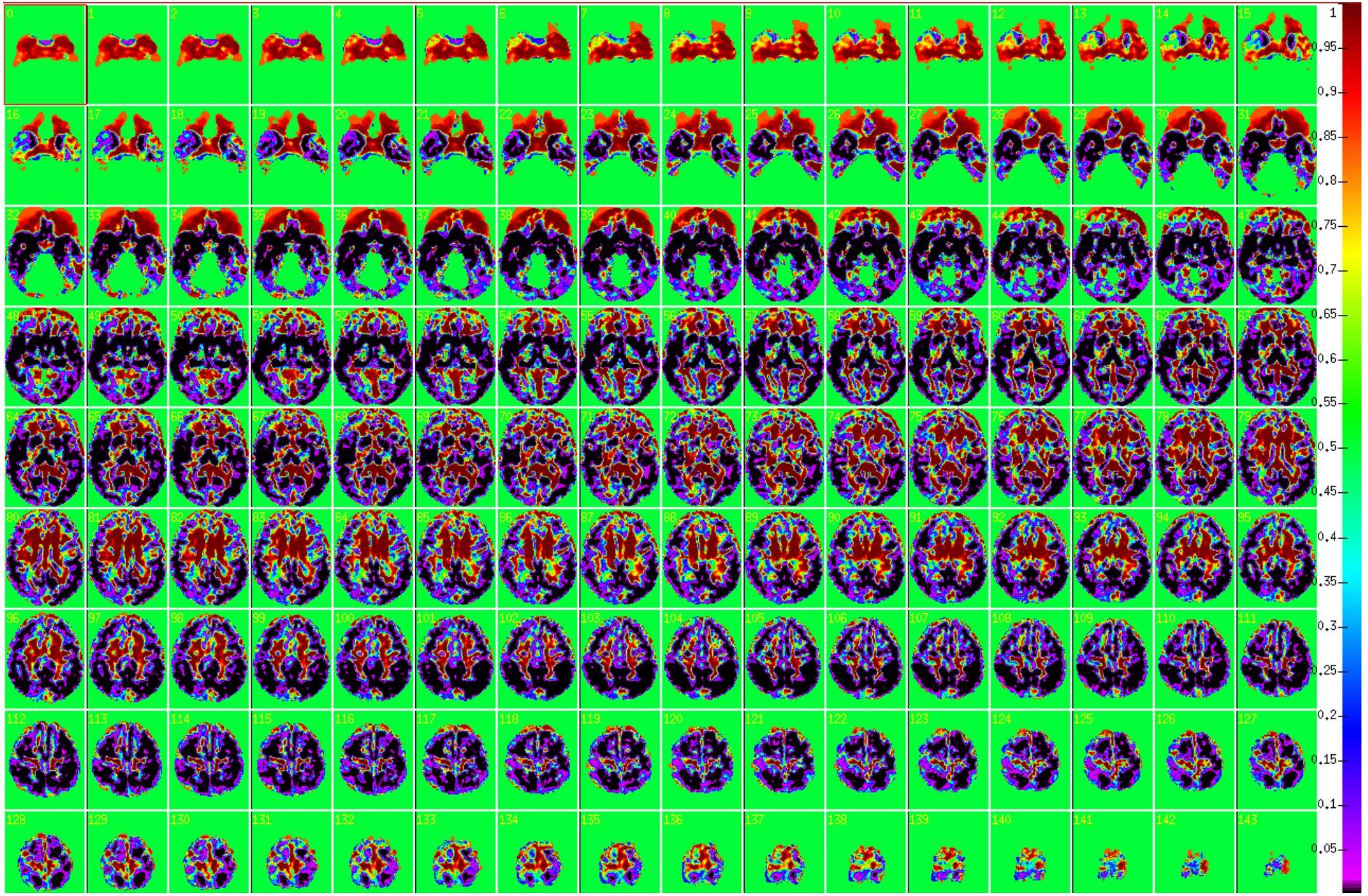


Fig. 4.4. Paired t-test results for AD/Control groups for gray matter: all p values (uncorrected).

The p-values found by the t-test are illustrated by Fig. 4.4. As the colorbar indicates, the voxels with statistical significance level p< 0.05 are the purple and black voxels – there are numerous such voxels. Although a significance level as high as 0.05 is an accepted level in general (for generic data), for spatial data it is common practice to assess significance via applying additional, stringent measures based on the spatial nature of the data, such that the results for the voxel set as a whole (a 3d brain in this case) can be taken into account. We now briefly overview three ways to do so – these approaches are employed by the widely used SPM5 (Statistical Parametric Mapping) method and software which we used for generating our statistical test results: 1) The first approach is to simply declare as significant only the voxels that have a much lower p-value than 0.05, such as p< 0.001. Note that no actual spatial processing/correction of p-values is employed in this case, simply the threshold is lowered – herein we refer to this approach as the "uncorrected p-values" approach. 2) A second approach is based on asking a question about a family of voxelwise statistics and the risk of error that we are prepared to accept referred to as the familywise error (FWE) which is the likelihood that the family of voxels could have arisen by chance. For this approach, to control FWE, SPM5 uses random field theory – i.e. the probability of ever reporting a false positive is controlled. 3) A third approach is to control the false discovery rate (FDR) which is the expected proportion of false positives among those voxels that are declared positive. Note that FDR does not control the probability of ever reporting a false positive (brain) voxel, and false positives will be detected – they are simply controlled so that they do not make up more than a percentage of the discoveries (positives).

Let us now consider the first approach above. Figures 4.5 and 4.6 illustrate in red the voxels with $p < 0.0001$ (uncorrected) for the smoothed RAVENS gray matter maps and the smoothed RAVENS white matter maps, respectively. The background image shown is the above-mentioned segmented Atlas2. We then built an SVM classifier only using as features all of these GM and WM red voxels combined – more specifically, to achieve lower feature dimensionality for this SVM experiment, we only used one of eight subsamples – in this way, the number of features was 34,090. For this classifier, 4 of 109 Control and 7 of 27 AD *test* samples (images) were misclassified, i.e. the misclassification rate was 0.08, same as achieved by SVM and MFE in previous sections on the same training population.

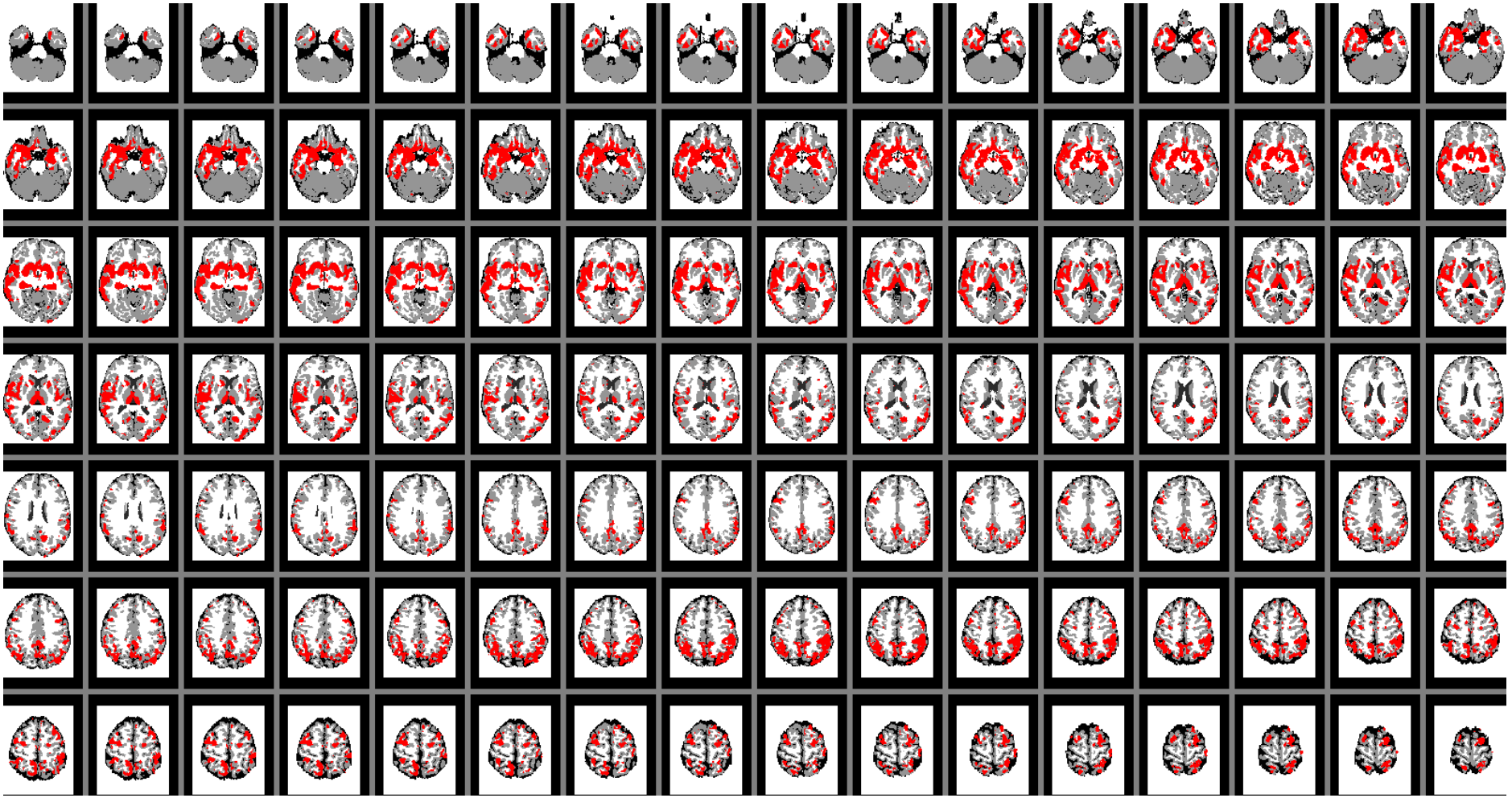


Fig. 4.5. Paired t-test results for AD/Control groups for gray matter: $p < 0.0001$ uncorrected

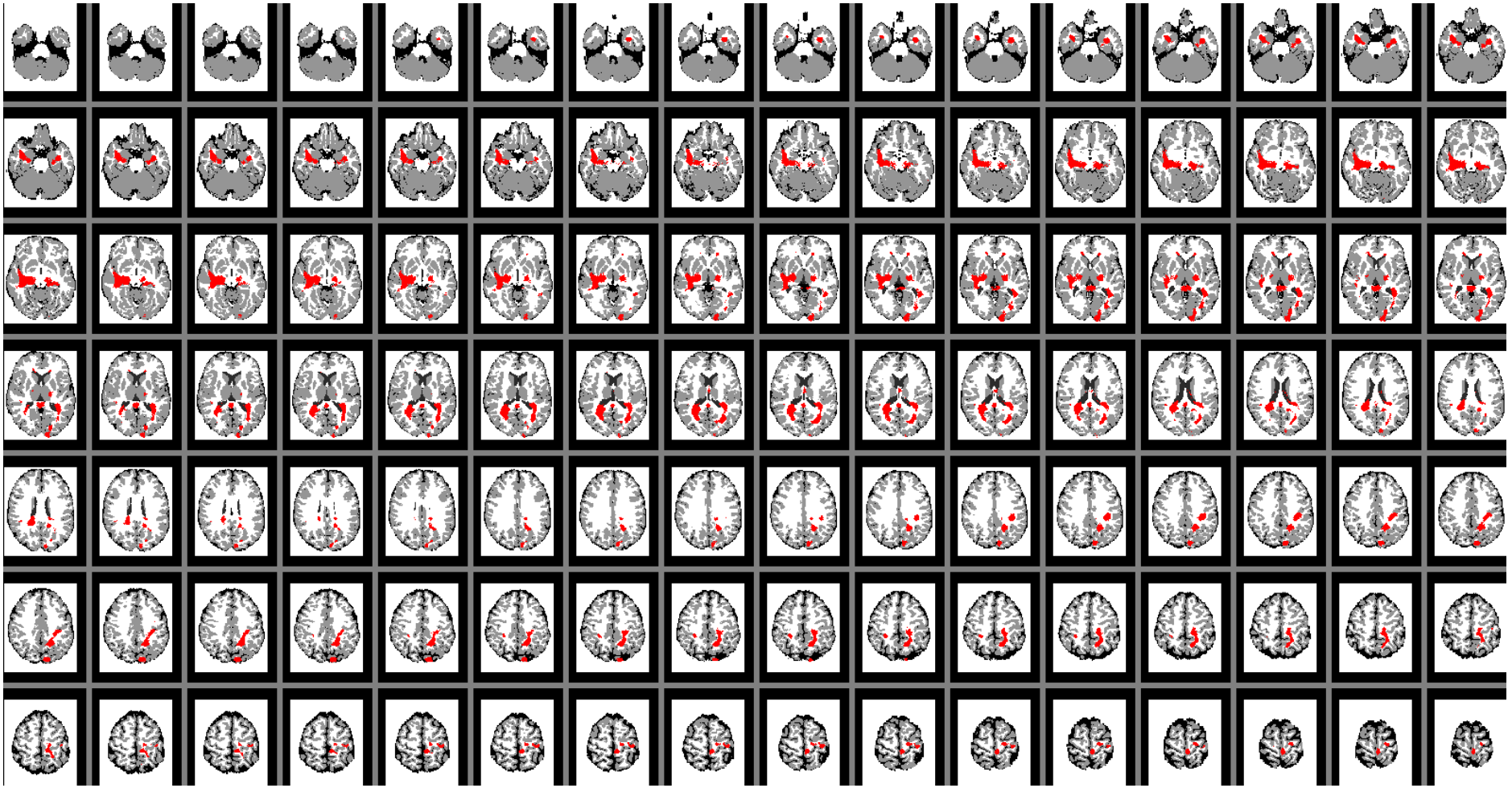


Fig. 4.6. Paired t-test results for AD/Control groups for white matter: $p < 0.0001$ uncorrected

For the third, FDR-based inference approach above, Fig. 4.7 illustrates the voxels for 1) an FDR-corrected p-level of 0.05 and for 2) a voxel extent (i.e. input to FDR for cluster size) of $k = 5$ for the FDR analysis, for the smoothed RAVENS gray matter maps. Yellow indicates higher significance than red - an ortho view of the result is shown in Fig. 4.8. For the AD and Control groups, we thus see that for this particular data (i.e. smoothed RAVENS gray matter maps) the *FDR-corrected* 0.05-level regions (Fig. 4.7) do not differ much from the purple and black *uncorrected* 0.05-level regions shown in Fig. 4.4. In comparison, Fig. 4.5, which, due to no use of correction, properly illustrated the case of a much lower p-threshold than 0.05, is showing fewer brain regions as significant, suggesting that the first inference approach may be more suitable than the third in this case.

As mentioned earlier, it is desirable to perform H-MFE experiments with our voxel-based RAVENS features because classification (or more precisely, a series of classifications and MFE invocations) based on *all available* voxel intensities (rather than a single subsample) would provide a tighter (and thus a more fair) comparison between our methods and other methods that use all voxels. Again, our comparison herein with a statistical test (SPM5) (which, unlike our methods, here *is* using all voxels) is favorably biased towards the statistical test as described above. Note, however, that our methods have a clear general advantage due to the fact that our SVM training (cf. Ch. 4) spans the entire brain, thus capturing voxel correlations throughout the brain at initial analysis (i.e. initial SVM classification prior to potential subsequent feature elimination), whereas the initial step of *voxelwise* statistical tests is to analyze the voxels individually without capturing their correlations. Perhaps this advantage explains partly the fact that our methods were able to achieve the same classification accuracy as SPM5 despite using many fewer voxels overall.

Fig. 4.7. Paired t-test results for AD/Control groups for gray matter: FDR-corrected

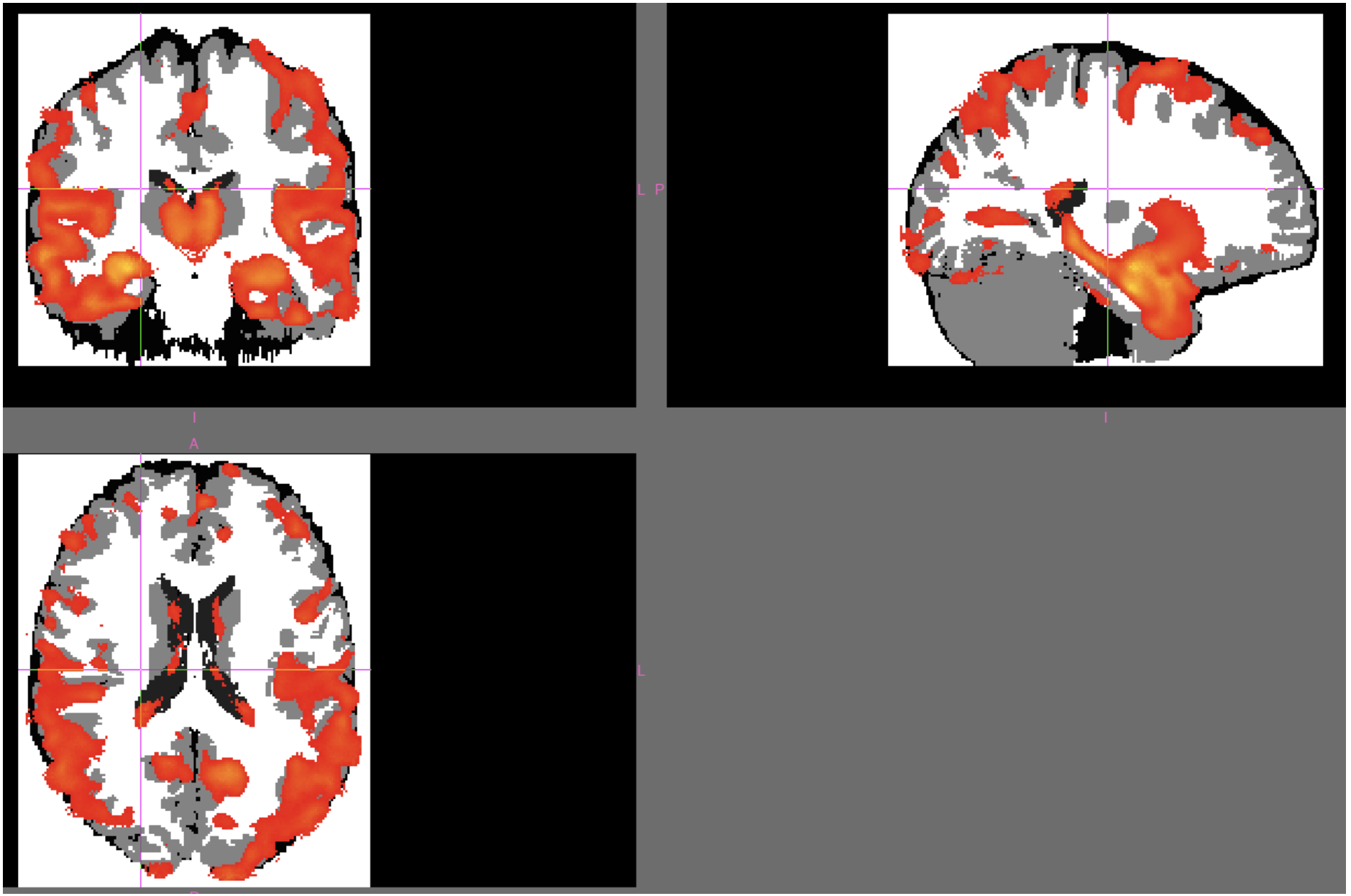


Fig. 4.8. Paired t-test results for AD/Control groups for gray matter: FDR-corrected (ortho)

4.4 Conclusions

Given T_1-weighted MRI brain images for up-to-6 visits of a population of approximately 600 participants clinically labeled as Alzheimer's Disease (AD), Mild Cognitive Impairment (MCI), or neither (i.e. Normal Controls), we have generated region-segmented images and tissue-specific volumetric density images (RAVENS images), which are morphometrically powerful 3d images suitable for analyzing AD and MCI which cause atrophy in potentially numerous regions of the brain. The use of density image voxel intensities as features provided a high-dimensional feature space, and the use of normalized volumes of the brain-atlas-defined regions in the region-segmented image provided a much lower-dimensional feature space. For the case of each space, building a Control-AD SVM classifier, we showed that a previously unseen ("test") AD or Control participant can be correctly classified with 91-92% accuracy. This accuracy is essentially similar to the accuracy reported by prior works for this particular classification task – in the subsequent step (where features are eliminated for AD biomarker selection), clearly we do have one advantage because prior works e.g. (16; 51) selected AD biomarkers based on the RFE method which we showed is outperformed by our MFE method (cf. Ch. 2, Ch. 4, and (3)). For both of our feature categories (i.e. ROI-based and voxel-based), using MFE we were able to eliminate the vast majority of features without significantly lowering the accuracy, thereby identifying brain regions as AD biomarkers with high accuracy. Moreover, we noted that several of the regions we found are well-published biomarkers of the disease.

Next, we designed a novel diagnosis system, suitable mainly for MCI patients, that is able to *only use the first-visit MRI* of the patient for diagnosing with high accuracy whether he/she will convert to AD. This system utilized a novel "conversion-by-trajectory" concept, introduced by this thesis, which is based on the movement, over time, of *training* MCI subjects in relation to a (single, fixed) *Control-AD* classification decision boundary *trained on MRI-based image data*. By contrast, the alternative "conversion-by-CDR" approach (cf. 4.1), proposed by prior works e.g. (51), is a relatively simplistic approach that considers the group of MCI participants as two subgroups *based only on whether CDR stayed as 0.5 over time or increased*. For the above aim of *early diagnosis* at first visit for MCI patients, we generated results for both our system and the alternative conversion-by-CDR approach. For prediction for MCI patients based on the conversion-by-CDR approach, (51) reported a maximum accuracy of 81.5%, with accuracy fluctuating in the $75 - 80\%$ range, whereas, for prediction for MCI patients (i.e. the same aim) our system achieved 87% accuracy with voxel-based features (and 80% accuracy with ROI-based features, which increased to 83% using MFE). We additionally evaluated the diagnosis capability of our system on (the first-visit images of) *AD* and *Control* test participants and found expected results: (1) the vast majority of these first-visit images of Control subjects were classified as the MCI class that stays more Control-like than AD-like over time, (2) the vast majority of the first-visit images of *overall high-CDR AD subjects* (i.e. AD subjects with a CDR of 1, 2, or 3 *even at first visit*) were classified as the MCI class that becomes more AD-like than Control-like over time, and (3) within the group of AD subjects who had a CDR of 0.5 at first visit, the majority was classified the same as the ADs above who had come in with a high CDR (i.e. the majority was classified as the MCI class that is eventually on the AD side of the Control-AD boundary). Within this group of AD subjects who initially had a CDR of 0.5, some went to a higher CDR at a subsequent visit whereas others stayed at 0.5. Those with a subsequently high CDR resembled, at first visit, those who had *started* with a high CDR more than did those who stayed at 0.5, with respect to the above classification. In one example experiment considering those who had stayed at 0.5, while 72% were classified as the converting MCI class (a relatively small number, perhaps due to it being difficult to discriminate between ADs and MCIs staying at 0.5), the percentage is higher for AD subjects who either start at a high CDR (e.g. 94%) or eventually increase to a high CDR (e.g. 77%).

Lastly, we also made some performance comparisons between our methods and the widely used voxelwise statistical testing approach (i.e., the SPM5 paired t-test herein), noting that our methods have a clear general advantage due to the fact that our SVM training (cf. Ch. 4.3.2) spans the entire brain, thus capturing correlations throughout the brain at initial analysis (i.e. initial SVM classification prior to potential subsequent feature elimination), whereas the initial step of *voxelwise* statistical tests is to analyze the voxels individually without capturing their correlations.

4.5 Acknowledgement

Data collection and sharing for this project was funded by the Alzheimer's Disease Neuroimaging Initiative (ADNI) (National Institutes of Health Grant U01 AG024904). ADNI is funded by the National Institute on Aging, the National Institute of Biomedical Imaging and Bioengineering, and through generous contributions from the following: Abbott, AstraZeneca AB, Bayer Schering Pharma AG, Bristol-Myers Squibb, Eisai Global Clinical Development, Elan Corporation, Genentech, GE Healthcare, GlaxoSmithKline, Innogenetics, Johnson and Johnson, Eli Lilly and Co., Medpace, Inc., Merck and Co., Inc., Novartis AG, Pfizer Inc, F. Hoffman-La Roche, Schering-Plough, Synarc, Inc., as well as non-profit partners the Alzheimer's Association and Alzheimer's Drug Discovery Foundation, with participation from the U.S. Food and Drug Administration. Private sector contributions to ADNI are facilitated by the Foundation for the National Institutes of Health (www.fnih.org). The grantee organization is the Northern California Institute for Research and Education, and the study is coordinated by the Alzheimer's Disease Cooperative Study at the University of California, San Diego. ADNI data are disseminated by the Laboratory for Neuro Imaging at the University of California, Los Angeles. This research was also supported by NIH grants P30 AG010129, K01 AG030514, and the Dana Foundation.

Appendix A

Software we created

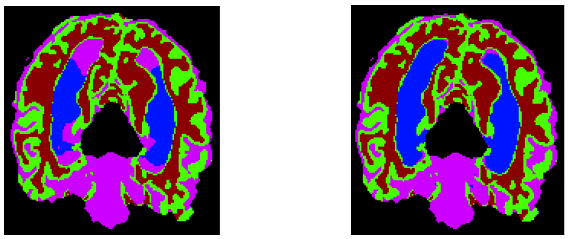
A.1 SVMcatalyst software

SVMcatalyst is our software tool for carrying out SVM classification and SVM-based feature selection experiments that additionally provides the user with several types of convenient and versatile functionality for/during the task of reading, interpreting, and organizing/managing the image data of a large (or small) population of people who may both have multiple images acquired over time (i.e. longitudinally) and multiple image types.

A.2 fv algorithm and tool for ventricle segmentation of CSF

A.2.1 fv algorithm

Illustrated by a single 2d slice in Fig. A.1(a), consider the problematic scenario where a 3d brain image has been segmented into gray matter (green), white matter (brown), CSF (purple), ventricles (blue), and background (black) but with a considerable number of true ventricle voxels mislabeled as CSF.[1]



(a) Before. (b) After.

Fig. A.1. (a) Illustration of problematic scenario with ventricles partly mislabeled as CSF. (b) Result of the *fv* algorithm for the problem.

[1] As discussed in Sec. 4.2, this problematic scenario often arises as a result of using HAMMER to segment CSF into ventricles and non-ventricle CSF.

To remedy this problem (so as to achieve a result as illustrated in Fig. A.1(b)), assuming the voxels labeled as ventricle (denoted by set V_I) have been labeled reliably, we have designed an algorithm based simply on determining which voxels labeled as CSF (denoted by set C_I) should be instead labeled as ventricle – with V_{alg} denoting this set, the *fv (fill ventricle)* algorithm segments C_I into V_{alg} and a second set C_F, and declares the true ventricles as $V_F = V_I \cup V_{alg}$ and the true CSF as C_F. Since V_I is assumed reliable, *fv* utilizes detection theory (hypothesis testing) to detect the C_I voxels that exhibit stronger V_I characteristics than C_I characteristics. V_I serves as the sample set for calculating parameter estimates for the multivariate Gaussian probability density function $f_1(\cdot)$ (associated with the hypothesis H_1 that a voxel labeled as CSF is actually a true ventricle voxel) – C_I similarly serves for calculating parameter estimates for $f_0(\cdot)$ (associated with the competing hypothesis H_0 that a voxel labeled as CSF is indeed a true CSF voxel). Herein, we assumed the variates are uncorrelated with each other, i.e., the covariance matrix is diagonal.

As illustrated by the figure, CSF roughly consists of three regions: 1) r_w: CSF *within* the cortex (i.e., union of ventricles (r_{w_1}) and non-ventricle CSF (r_{w_2})), 2) r_b: CSF *below* the (lower-brain) cortex, 3) r_a: CSF *above* (and surrounding) the cortex. r_w is the most enclosed (by gray and/or white matter, i.e. "GW"), r_b is somewhat enclosed, and r_a is (almost completely) not enclosed. On this highly topological nature we base our sample value construction for the two voxel populations (population 0 and 1) as follows. For a C_I or V_I voxel, the set of 26 rays outward from the voxel can be partitioned, in the following anatomy- and topology-driven way, into subsets R_1, R_2, R_3, R_4, R_5 (in that order): before exiting the 3d image, 1) ray does not encounter GW but encounters ventricle, 2) encounters neither, 3) encounters both but with the restriction that the ventricle encounter precedes the GW encounter, 4) encounters GW but not ventricle, 5) encounters both GW and ventricle but with the restriction that the GW encounter precedes the ventricle encounter[2]. The number of rays in R_2 (attribute a_1) can help distinguish r_w and r_a – we need additional attributes to distinguish r_{w_1} (which we seek) from r_b and r_{w_2}. Since we expect no V_I voxels in r_{w_2} whereas many in r_{w_1} and potentially some in r_b, the number of rays in $R_1 \cup R_3$, $n_{1,3}$, has discrimination power, based on which we designed two attributes (a_2 and a_3) as follows.

First, with a_2, in addition to the use of $n_{1,3}$ we aim to capture how far our sought true ventricle voxels are from their enclosing GW – capturing this combination can be achieved, in different ways for population 0 and 1, as follows. For "distance", we simply use the average of the distances from the voxel to its surrounding GW (note that only rays in $R_3 \cup R_4$ contribute to this average), with the following constraint. Since V_I is our model herein for true ventricle (as per the reliability assumption), we expect a population 1 sample (voxel) to be already surrounded by both other V_I voxels (i.e., $n_{1,3}$ is positive) and GW. Thus, note that the average distance (to GW) for that voxel already achieves the combination, without imposing a special constraint. By contrast, for a population 0 voxel, $n_{1,3}$ is not likely to be positive, so we implement the constraint that we shall artificially set the voxel's average distance (to GW) to 0 in the event that the voxel's $n_{1,3}$ is 0 – the idea is that a C_I voxel v not being surrounded by any V_I (ventricle) voxel is an indicator that v should not contribute an actual positive distance to the average because this number may then have a confounding effect during the hypothesis testing.

Second, with a_3, we aim to capture that many rays through a true ventricle voxel will encounter not only V_I (ventricle) at one end but also GW at the other end. That is, while a_2 was based on distance (to GW) calculated in the same direction the ventricle (V_I) was encountered, a_3 is based on distance (to GW) calculated in the direction opposite to the direction the ventricle (V_I) was encountered. The average distance is used.

[2] Our sole reason for defining R5 is to make the partitioning exhaustive – we do not utilize R5 in the algorithm.

For each voxel $v \in C_I$, using the attribute vector $\underline{x} = [a_1\ a_2\ a_3]^{\mathrm{T}}$ for the voxel the algorithm computes $f_0(\underline{x})$ and $f_1(\underline{x})$, and puts the voxel into V_{alg} if $f_1(\underline{x}) > f_0(\underline{x})$, otherwise puts the voxel into C_F.

A.2.2 fv tool

The fv tool *fv* is a binary executable with usage given below. It can be compiled on multiple platforms – for this thesis, it was only compiled and invoked on Linux.

Apart from the core *fv* algorithm described in Sec. A.2.1, the *fv* tool contains basic 3d connected-components functionality as a user option, whereby the user may remove from the core *fv* algorithm's output ventricles (which is the set V_F in Sec. A.2.1) any connected component no larger than a user-specified threshold (in units of "number of voxels") – see *-r* and *-b* below.

fv - fill ventricles

When a tissue-segmented image (with white matter=250, gray matter=150, ventricles=50, CSF=10, background=0) is problematic due to some of its ventricular voxels having been set to 10 instead of 50, this program attempts to automatically remedy the problem. An output image file will be created - the input image file will not be modified.

To see program usage info, invoke the program name by itself (without its arguments).

usage:

fv [-r threshold] [-b] [-v] [-f] [-c preextension_name_of_comparison_image_file] <number_of_slices> <number_of_rows> <number_of_columns> <directory> <pre_extension_name_of_input_image_file>

- *Returns 0 if successful, 1 otherwise.*
- *User inputs that are optional (indicated above with square brackets [ ]) must precede the user inputs that are required (indicated above with < >). Enter required inputs in the order shown.*
- *Your input image file needs to be in the directory you are specifying. The image must be unsigned 8-bit integer.*
- *Your output image file will be written to that directory - its filename preextension will have a trailing "_fv" appended to the one you are specifying for the input image. Output image will be unsigned 8-bit integer.*
- *Use the -r option if you want to remove from the final ventricles any blob no larger than a size (threshold, in units of "number of voxels") you specify. Irrespective of the -r option, use the -b option if you want to create an output image file in which the blobs that make up the ventricles are enumerated starting at 1, prior to any blob removal.*
- *Use the -v option if you want to create an output image file only indicating the final ventricles (set to 50).*
- *Use the -f option if you want an output image file to be created wherein the CSF voxels switched by this program to become ventricle will be 100 in that image - that is, such image (with "_beforeafter" in its filename) conveys the image appearance before and after.*
- *Use the -c option if you want to provide a reference segmented image (such as your manually segmented image) to this program for a comparison of the program's algorithm's results with this reference image, which is useful for algorithm validation. Requirements for your reference image: 1) Each ventricle voxel value must be 50. 2) The reference image must be unsigned 8-bit integer and in the same orientation as your input image. The following numbers, separated by a single space, will be printed (to stdout) by the program in this order:*

- *B = Number of voxels that are ventricle in both your reference image and the program's output image.*
- *O = Number of voxels that are ventricle in the program's output image but not in your reference image.*
- *R = Number of voxels that are ventricle in your reference image but not in the program's output image.*
- *Sensitivity = B/(B+R).*

A.3 STAMPS software

We created this software in collaboration with Dr. D. Bigler and published (7) which describes STAMP, a software tool for automated MRI post-processing, and STAMPS, a software tool for running STAMP on a supercomputer for drastically higher overall computational efficiency, especially for a large population of subjects and their MR image data.

Appendix B

B.0.1 Exhaustive Subsampling Approach (ESA) and Hierarchical MFE (H-MFE)

ESA: Given a 3d array X, one subsample $S_{i,j,k}$ of X can be obtained as follows (using Matlab syntax): $S_{i,j,k} = X((i+1) : I : end, (j+1) : J : end, (k+1) : K : end)$ where:

- positive integers $I-1$, $J-1$, $K-1$ are respectively the chosen number of skips for first, second, third dimension (i.e. number of X elements skipped along a dimension for obtaining consecutive elements of subsample S).[1]

- nonnegative integers i, j, k are respectively the chosen offsets from which point the skipping will commence for first, second, third dimension.

The number of mutually exclusive subsamples $S_{i,j,k}$ that one would need to create to exhaustively partition X into such subsamples is $I \cdot J \cdot K$. The order in which ESA creates this many subsamples is given by:

```
for i=0 to I-1
for j=0 to J-1
for k=0 to K-1
create S_{i,j,k}
end
end
end
```

H-MFE: The first step of H-MFE is to perform MFE on each subsample, producing feature subsets $F_l^1, l \in S^1 \equiv \{1, \ldots, I \cdot J \cdot K\}$, with the superscript denoting the step index. At the second H-MFE step, based on a user-specified number (K) for exhaustively partitioning S^1 into mutually exclusive subsets $L_1 \subset S^1, \ldots, L_K \subset S^1$, K new sets $F_k^2 = \bigcup_{l \in L_k \subset S^1} F_l^1, \;\; k = 1, \ldots, K$ are formed, on which to perform MFE. The user may choose K such that each MFE instance at the second step would eliminate from roughly as many features as each MFE instance in the first step did, i.e. K may be chosen such that each F_k^2 set (to be automatically formed) would have roughly as many features as each F_k^1 set did. H-MFE simply continues in this iterative fashion until, at some step n, a single F_k^n to be formed for the step (rather than the forming of multiple such sets for the step) suffices in that the number of not-yet-eliminated features during the overall H-MFE process (i.e. the number of features in F_k^n) has now become small enough for a single MFE instance to eliminate from, so as to not employ an additional (subsequent) H-MFE step.

[1] Note that I, J, K need not be equal, because the 3d array need not be a cube and may be a cuboid.

Appendix C

HAMMER method

For morphometric analysis of a population of brain MRIs, due to intra- and inter-person anatomic variations a nonlinear method of image registration, rather than a linear (affine) method, is considered essential. Thus, in our study of AD and MCI using brain MRIs, we utilize the HAMMER nonlinear registration method (62), which we chose for several additional reasons: First, for nonlinear spatial normalization (registration) of each individual to one reference image (3D brain atlas), the HAMMER tool is a freely available method designed for high accuracy in anatomic correspondence. In this method, which is a hierarchical (coarse-to-fine) matching algorithm, the objective function (in the optimization algorithm) is successively approximated by lower-dimensional smooth functions constructed to have significantly fewer local minima. The method constructs them based on selecting driving anatomical features, represented as distinct "attribute" vectors, for the "matching". The aim is to highly reduce ambiguity in finding correspondence. Second, HAMMER registered our data well and was stable and reasonably fast as a rigorous nonlinear method. (i.e. approximately 2 h per 3D image). Third, using a fixed reference 3d brain atlas image that has been intricately pre-segmented into numerous (e.g. 100) anatomical regions, HAMMER is able to segment a subject's MRI brain image into these regions, via HAMMER registration – this enables region-specific volumetric measurements useful for detecting brain atrophy typical for AD/MCI subjects. Fourth, for the brain MRI population in our study, via the nonlinear registration the HAMMER software generates for each individual a *volumetric density map* (a 3D image generated separately for each of three "tissues": 1. gray matter, 2. white matter, 3. ventricles), named "RAVENS tissue density map", which is suitable for overall morphometric analysis (such as atrophy detection for AD/MCI), more specifically as follows. The RAVENS voxel value is a measurement of regional brain volume density *describing the individual's original (pre-registration) 3D image*, in the sense that a sum of an individual's RAVENS voxels in a given region is equivalent to the actual volume one would obtain for that region (by counting the region's pixels) in the individual's pre-registration image. Of particular interest, by adjusting the (volumetric) density of a tissue whenever the underlying nonlinear registration expands or contracts the brain geometry, the RAVENS map preserves the total amount of tissue in any defined brain region, which sets it apart from volumetric density maps of other methods. For example, a brighter RAVENS corpus colossum for person A than B means A has a larger corpus colossum. The RAVENS approach (17; 18; 35; 63) has been validated for voxel-based analysis (17) and applied to various studies e.g. (24). Of particular interest, (17) supported that voxel-based SPM statistical analysis, which we perform herein for comparison with our methods, can be performed on RAVENS maps.

Bibliography

[1] Alzheimer's Disease Neuroimaging Initiative (ADNI), www.adni-info.org

[2] ADNI database, www.loni.ucla.edu/ADNI

[3] Y. Aksu, D. J. Miller, G. Kesidis, and Q. X .Yang, "Margin-Maximizing Feature Elimination Methods for Linear and Nonlinear Kernel-Based Discriminant Functions", *IEEE Transactions on Neural Networks*, vol. 25, no.10, pp. 701-717, 2010.

[4] Y. Aksu, D. J. Miller, and G. Kesidis, "Margin-based feature selection techniques for support vector machine classification," in *Proc. IAPR Workshop Cogn. Inf. Process.*, pp. 176-181, 2008.

[5] Y. Aksu, G. Kesidis, and D. J. Miller, "Scalable, efficient, stepwise-optimal feature elimination in support vector machines," in *Proc. IEEE Workshop Mach. Learn. Signal Process.*, pp.75-80, 2007.

[6] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining Knowledge Discovery* 2 (2) pp.121-167, 1998.

[7] D. C. Bigler, Y. Aksu, D. J. Miller, Q. X. Yang, "STAMPS: Software Tool for Automated MRI Post-processing on a supercomputer", *Computer Methods and Programs in Biomedicine* 95, pp.146-157, 2009.

[8] A. B. Chan, N. Vasconcelos, and G. R. Lanckriet, "Direct convex relaxations of sparse SVM," in *Proc. Int. Conf. Machine Learning*, 2007.

[9] C. Chang and C. Lin, "LIBSVM : a library for support vector machines," software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm, 2001.

[10] G. Chetelat, B. Desgranges, V. de la Sayette, F. Viader, F. Eustache, J-C. Baron, "Mapping gray matter loss with voxel-based morphometry in mild cognitive impairment", *NeuroReport* vol.13. no.15, 28 October 2002.

[11] Y-Y. Chou, N. Lepore, C. Avedissian, S. K. Madsen, X. Hua, C. R. Jack Jr., M. W. Weiner, A. W. Toga, P. M. Thompson, and the Alzheimer's Disease Neuroimaging Initiative, "Mapping Ventricular Expansion and its Clinical Correlates in Alzheimer's Disease and Mild Cognitive Impairment using Multi-Atlas Fluid Image Alignment", *Proc. SPIE*, vol.7259, 725930, 2009.

[12] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Transactions on Electronic Computers*, EC-14(3):326–334, June 1965.

[13] K. Crammer, Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines", *J. Mach. Learn. Res.*, 2, pp. 265-292, 2001.

[14] N. Cristianini and J. Shawe-Taylor, An Introduction to Support Vector Machines. Cambridge University Press, 2000.

[15] J. G. Csernansky, L. Wang, J. Swank, J. P. Miller, M. Gado, D. McKeel, M. I. Miller, J. C. Morris, "Preclinical detection of Alzheimer's disease: hippocampal shape and volume predict dementia onset in the elderly", *NeuroImage* 25, issue 3, 783-792, 2005.

[16] C. Davatzikos, Y. Fan, X. Wu, D. Shen, S. M. Resnick, "Detection of prodromal Alzheimer's disease via pattern classification of magnetic resonance imaging", *Neurobiology of Aging* 29, pp.514-523, 2008.

[17] C. Davatzikos, A. Genc, D. Xu, and S. M. Resnick, "Voxel-Based Morphometry Using the RAVENS Maps: Methods and Validation Using Simulated Longitudinal Atrophy," *NeuroImage* 14, pp.1361-1369, 2001.

[18] C. Davatzikos, "Mapping image data to stereotaxic spaces: Applications to brain mapping," *Hum. Brain Mapp.*, 6:334-338, 1998.

[19] M. J. de Leon, S. DeSanti, R. Zinkowski, P. D. Mehta, D. Pratico, S. Segal, H. Rusinek, J. Li, W. Tsui, L. A. Saint Louis, C. M. Clark, C. Tarshish, Y. Li, L. Lair, E. Javier, K. Rich, P. Lesbre, L. Mosconi, B. Reisberg, M. Sadowski, J. F. DeBernadis, D. J. Kerkman, H. Hampel, L. -O. Wahlund, P. Davies, "Longitudinal CSF and MRI biomarkers improve the diagnosis of mild cognitive impairment," *Neurobiology of Aging* 27, pp. 394-401, 2006.

[20] R. Duda, P. Hart, and G. Stork, Pattern Classification. Second Edition, John Wiley and Sons, New York, 2001.

[21] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization," in *Proc. of the Conf. on Info. and Knowl. Manag.*, 1998.

[22] Y. Fan, D. Shen, C. Davatzikos, "Classification of structural images via high-dimensional image warping, robust feature extraction, and SVM", J. Duncan and G. Gerig (Eds): MICCAI 2005, LNCS 3749, pp. 1-8, 2005.

[23] Y. Fan, N. Batmanghelich, C. M. Clark, C. Davatzikos, "Spatial patterns of brain atrophy in MCI patients, identified via high-dimensional pattern classification, predict subsequent cognitive decline", *NeuroImage* 39, pp.1731-1743, 2008.

[24] Y. Fan, D. Shen, R. C. Gur, R. E. Gur, C. Davatzikos, "COMPARE: Classification of Morphological Patterns Using Adaptive Regional Elements", *IEEE Transactions on Medical Imaging*, vol. 26, no. 1, pp.93-105, 2007.

[25] C. Fennema-Notestine, D. J. Hagler Jr., L. K. McEvoy, A. S. Fleischer, E. H. Wu, D. S. Karow, A. M. Dale, the Alzheimer's Disease Neuroimaging Initiative, "Structural MRI biomarkers for preclinical and mild Alzheimer's disease", *Human Brain Mapping*, vol.30, issue 10, pp.3238-3253.

[26] V. Franc, V. Hlavac, "Multi-class Support Vector Machine", in *ICPR 02: Proceedings 16th International Conference on Pattern Recognition*, vol. 2, pp. 236-239, 2002.

[27] V. Franc, V. Hlavac, "Multi-class Support Vector Machines", presentation, Center for Machine Perception, Czech Technical University, Prague.

[28] V. Franc, "Optimization Algorithms for Kernel Methods" Ph.D. thesis, Czech Technical University, July 29, 2005.

[29] FSL (FMRIB Software Library), http://www.fmrib.ox.ac.uk/fsl

[30] G. Fung, O. L. Mangasarian, "A feature selection Newton method for support vector machine classification," *Computational Optimization and Applications*, vol. 28, no.2:185-202, July 2004.

[31] G. Fung, O. L. Mangasarian, "Multicategory Proximal Support Vector Machine Classifiers," *Machine Learning*, 59:77-97, 2005.

[32] T. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906-914, 2000.

[33] D. Geman, C. d'Avignon, D. Q. Naiman, Raimond L. Winslow, "Classifying gene expression profiles from pairwise mRNA comparisons," *Stat. Appl. Geneti. Mol. Biol.* 3, Article 19, 2004.

[34] A. C. Tan, D. Q. Naiman, L. Xu, R. L. Winslow and D. Geman, "Simple decision rules for classifying human cancers from gene expression profiles", *Bioinformatics* vol. 21, no. 20, pp.3896-3904, 2005.

[35] A. F. Goldszal, C. Davatzikos, D. L. Pham, M. X. H. Yan, R. N. Bryan, S. M. Resnick, "An Image-Processing System for Qualitative and Quantitative Volumetric Analysis of Brain Images," *J. Comput. Assist. Tomogr.*, 22:827-837, 1998.

[36] Y. Guermeur. "Combining Discriminant Models with New Multi-class SVMs", *Pattern Analysis and Applications*, (2002)5:168-179

[37] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, 46(1):389-422, 2002.

[38] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157-1182, 2003.

[39] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, New York: Springer, 2001.

[40] C.-W. Hsu, C.-J. Lin. "A comparison of methods for multi-class support vector machines", *IEEE Transactions on Neural Networks*, 13(2002), 415-425.

[41] M. Jenkinson and S.M. Smith, "A global optimisation method for robust affine registration of brain images," *Med. Image Anal.*, 5(2):143-156, 2001.

[42] T. Joachims, "Training linear SVMs in linear time," *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 217-226, 2006.

[43] B. Karacali, H. Krim, "Fast Minimization of Structural Risk by Nearest Neighbor Rule, *IEEE Transactions on Neural Networks*, vol.14, no.1, 2003.

[44] B. Karacali, R. Ramanath, W. E. Snyder, "A comparative analysis of structural risk minimization by support vector machines and nearest neighbor rule", *Pattern Recognition Letters*, vol. 25, issue 1, pp.63-71, 2004.

[45] S. S. Keerthi, D. DeCoste, "A Modified Finite Newton Method for Fast Solution of Large Scale Linear SVMs," *J. Mach. Learn. Res.*, vol. 6, pp. 341-361, 2005.

[46] M. Kugler, K. Aoki, S. Kuroyanagi, A. Iwata, and A. S. Nugroho, "Feature subset selection for support vector machines using confident margin," *Proc. IJCNN*, pp. 907-912, 2005.

[47] J. P. Lerch, A. C. Evans, "Cortical thickness analysis examined through power analysis and a population simulation", *NeuroImage* 24, pp.163-173, 2005.

[48] Computational Methods of Feature Selection, edited by H. Liu and H. Motoda,Chapman & Hall/CRC, 2008.

[49] O. L. Mangasarian, "Exact 1-norm support vector machines via unconstrained convex differentiable minimization," *J. Mach. Learn. Res.*, vol. 7, pp. 1517-1530, 2006.

[50] O. L. Mangasarian, D. R. Musicant, "Successive overrelaxation for support vector machines," *IEEE Transactions on Neural Networks*, 10(5), 1999.

[51] C. Misra, Y. Fan, C. Davatzikos, "Baseline and longitudinal patterns of brain atrophy in MCI patients, and their use in prediction of short-term conversion to AD: Results from ADNI", *NeuroImage* 44, pp.1415-1422, 2009.

[52] D. Mladenić, J. Brank, M. Grobelnik, N. Milic-Frayling, "Feature selection using linear classifier weights: interaction with classification models," *Proc. ACM SIGIR Conf. on R&D in Info. Retrieval*, pp. 234-241, 2004.

[53] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. PAMI*, vol. 27, no. 8, pp. 1226-1238, Aug. 2005.

[54] R. C. Petersen, Mild cognitive impairment: aging to Alzheimer's Disease, Oxford University Press, 2004.

[55] S. Ramaswamy et al, "Multiclass cancer diagnosis using tumor gene expression signatures," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 98, no. 26, 15149-15154, Dec. 2001.

[56] M. A. Fischler, R. C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Comm. of the ACM 24: 381-395, June 1981.

[57] S. M. Resnick, A. Goldszal, C. Davatzikos, S. Golski, M. A. Kraut, E. J. Metter, R. N. Bryan, A. B. Zonderman, "One-year age changes in MRI brain volumes in older adults", Cereb. Cortex 10, 464-472, 2000.

[58] H. Rusinek, Y. Endo, S. De Santi, D. Frid, W.-H. Tsui, S. Segal, A. Convit, "Atrophy rate in medial temporal lobe during progression of Alzheimer's disease", *Neurology*, vol.63, issue 12, 2354-2359, 2004.

[59] J. M. Schott, S.L. Price, C. Frost, J. L. Whitwell, et al, "Measuring atrophy on Alzheimer disease - A serial MRI study over 6 and 12 months," *Neurology*, 65(1),:119-124, 2005.

[60] S. M. Smith, "Fast robust automated brain extraction," *Human Brain Mapping*, 17(3):143-155, 2002.

[61] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis. Cambridge University Press, 2004.

[62] D. Shen and C. Davatzikos, "HAMMER: hierarchical attribute matching mechanism for elastic registration," *IEEE Trans. Medical Imaging*, 21(11):1421-1439, 2002.

[63] D. Shen, C. Davatzikos, "Very High-resolution Morphometry Using Mass-preserving Deformations and HAMMER Elastic Registration," *NeuroImage* 18, pp.28-41, 2003.

[64] T. R. Stoub, M. Bulgakova, S. Leurgans, D. A. Bennett, D. Fleischman, D. A. Turner, L. deToledo-Morrell, "MRI predictors of risk of incident Alzheimer disease," *Neurology* 64, pp. 1520-1524, 2005.

[65] Statistical Pattern Recognition Toolbox, cmp.felk.cvut.cz/cmp/software/stprtool

[66] P. M. Thompson, K. M. Hayashi, G. de Zubicaray, A. L. Janke, S. E. Rose, et al, "Dynamics of gray matter loss in Alzheimer's disease", *J. Neurosci.* 23, pp.994-1005, 2003.

[67] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Royal Statistical Society*, Series B (Methodological), vol. 58, no. 1, pp. 267-288, 1996.

[68] G. V. Trunk, "A problem of dimensionality: A simple example," *IEEE Trans. PAMI*, vol. 1, issue 3, pp. 306-307, July 1979.

[69] V. Vapnik, The Nature of Statistical Learning Theory. Springer-Verlag, New York, 1995.

[70] V. Vapnik, Statistical Learning Theory. John Wiley & Sons, 1998.

[71] P. Vemuri, J. L. Gunter, M. L. Senjem, J. L. Whitwell, K. Kantarci, D. S. Knopman, B. F. Boeve, R. C. Petersen, C. R. Jack Jr., "Alzheimer's disease diagnosis in individual subjects using structural MR images: Validation studies", *NeuroImage* 39, issue 3, 1186-1197.

[72] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, V. Vapnik, "Feature selection for SVMs," *NIPS 13*, MIT Press, 2001.

[73] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero norm with linear models and kernel methods," *J. Mach. Learn. Res.*, vol. 3, pp. 1439-1461, Mar. 2003.

[74] J. Weston, C. Watkins, "Support vector machines for multiclass pattern recognition", In *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 1999.

[75] J. Weston, "Extensions to the Support Vector Method," Ph.D. thesis, University of London, October 1999.

[76] Y. Zhang, M. Brady, and S. Smith, "Segmentation of brain MR images through a hidden Markov random field model and the expectation maximization algorithm," *IEEE Trans. Medical Imaging*, 20(1):45-57, 2001.

[77] X. Zhou, D.P. Tuck, "MSVM-RFE: extensions of SVM-RFE for multiclass gene selection on DNA microarray data," *Bioinformatics*, vol. 23, no. 9, pp. 1106-1114, 2007.

[78] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani, "1-norm support vector machines," *NIPS*, 2003.

Vita

Yaman Aksu

Education

The Pennsylvania State University, University Park, PA — 2004–present
Ph.D. in Electrical Engineering, expected Summer 2010.

The Johns Hopkins University, Baltimore, MD — 1993–1996
M.S.E. in Electrical and Computer Engineering.

Washington University, St.Louis, MO — 1992–1993
B.S.E. in Electrical Engineering, Summa Cum Laude.

Franklin and Marshall College, Lancaster, PA — 1989–1992
B.A. in Physics, Magna Cum Laude.

Extensive Experience with Biomed. Imaging and Machine Learning Software

BrainWave (GEMS), DIAS, DICOM toolkits, FSL, HAMMER, LibSVM, Matlab, MEDx, NLPSVM (Newton-based Linear Programming SVM), PGPDT (parallel SVM), SPM5 (incl. scripting), SPM99, STAMPS, SVMcatalyst (my software for SVM incl. biomedical image analysis), VolView, etc.

Journal Articles

• Y.Aksu, D.J.Miller, G.Kesidis, Q.X.Yang, "Margin-maximizing feature elimination methods for linear and nonlinear kernel-based discriminant functions", *IEEE Trans. on Neural Networks*, vol.25, no.10, pp.701-717, 2010.

• D.C.Bigler, Y.Aksu, D.J.Miller, Q.X.Yang, "STAMPS: software tool for automated MRI post-processing on a supercomputer", *Comp. Meth. and Prog. in Biomed.* 95, pp.146-157, 2009.

• G.S.Adkins, Y.M.Aksu, M.H.T.Bui, "Calculation of the two-photon-annihilation contribution to the positronium hyperfine interval at order $m\alpha^6$", Physical Review A 47, pp. 2640-2652, 1993.

Conference Papers

• Y.Aksu, D.J.Miller, G.Kesidis, "Margin-based feature selection techniques for support vector machine classification," *Proc. IAPR Workshop Cogn. Inf. Process.*, pp.176-181, 2008.

• Y.Aksu, G.Kesidis, D.J.Miller, "Scalable, efficient, stepwise-optimal feature elimination in support vector machines," *Proc. IEEE Workshop Mach. Learn. Signal Process.*, pp.75-80, 2007.

• Y.Zhang, Y.Aksu, G.Kesidis, D.J.Miller, Y.Wang, "SVM margin-based feature elimination applied to high-dimensional microarray gene expression data," *Proc. IEEE Workshop Mach. Learn. Signal Process.*, 2008.

Refereed Abstracts (One-page ISMRM Conference Papers)

• D. C. Bigler, C. Flaherty-Craig, Y. Aksu, B-Y. Lee, K. R. Scott, H. E. Stephens, J. J. Vesek, J. Wang, M. L. Shaffer, P. J. Eslinger, Z. Simmons, Q. X. Yang. "Cross-sectional and Longitudinal Voxel-Based Relaxometry Study in ALS," Proc. Intl. Soc. Mag. Reson. Med. 18 p.1968, 2010.

• D. C. Bigler, Y. Aksu, H. E. Stephens, J. Vesek, K. R. Scott, C. Flaherty-Craig, J. Wang, P. J. Eslinger, Z. Simmons, Q. X. Yang, "High-resolution Longitudinal Voxel-Based Morphometric Study in ALS," Proc. Intl. Soc. Mag. Reson. Med. 17, p.1113, 2009.

Awards and Honors

• University Graduate Fellowship, Penn State University — 2004-2005
• College of Engineering Supplemental Fellowship, Penn State University — 2004-2007
• Electrical Engineering Dept. Supplemental Fellowship, Penn State University — 2007-2008
• Phi Beta Kappa, and Sigma Pi Sigma National Physics Honor Society — 1992
• Rensselaer Polytechnic Institute Engineering and Science Award for Excellence — 1992
• John Kershner Award in Physics — 1992
• National Dean's List — 1991, 1992
• Hackman undergraduate scholar in telerobotics research — 1991
• Successful Participant in the COMAP Mathematical Contest in Modeling — 1990-1991

Selected Attended Conferences in Biomedical Image Analysis

• NLM ITK Consortium Meeting 2003, NIH, Bethesda, MD.
• fMRI Experience 2002, NIH, Bethesda, MD.
• SPIE Medical Imaging 2000, 2001, San Diego, CA.
• RSNA 1997, 1999, Chicago, IL.

Extensive Industry Experience

In my full-time employment during 1996-2004, serving first as *Systems Engineer (Medical Imaging)* and subsequently *Senior Systems Engineer (Medical Imaging)* and *Director of DICOM and Database Operations*, made significant contributions to major biomedical imaging systems/software developed at Sensor Systems, Inc. and its subsidiary Medical Numerics, such as MEDx (used by over 300 research labs worldwide), BrainWave (FDA-cleared fMRI system on GE Medical Systems MRI scanners), and DIAS/MARS/LARS (Digital Image Archive System at the NIH, for which I was the principal architect).

Other service to the medical imaging community

Contributing member of DICOM Standards Committee Working Group 16 for the Enhanced MR Object, Rosslyn, VA, 1998-2003. Resulting Supplement 49: *ftp://medical.nema.org/medical/dicom/final/sup49_ft.pdf*

Teaching Assistantships

PSU (Computer Vision, Image Processing) 2006-08, JHU (Signals/Systems, Circuits) 1993-95.