The Power Landmark Vector Learning Framework

by

Shuo Xiang

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2008

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Kernel methods have recently become popular in bioinformatics machine learning. Kernel methods allow linear algorithms to be applied to non-linear learning situations. By using kernels, non-linear learning problems can benefit from the statistical and runtime stability traditionally enjoyed by linear learning problems. However, traditional kernel learning frameworks use implicit feature spaces whose mathematical properties were hard to characterize. In order to address this problem, recent research has proposed a vector learning framework that uses *landmark* vectors which are unlabeled vectors belonging to the same distribution and the same input space as the training vectors. This thesis introduces an extension to the landmark vector learning framework that allows it to utilize two new classes of landmark vectors in the input space. This augmented learning framework is named the power landmark vector learning framework. A theoretical description of the power landmark vector learning framework is given along with proofs of new theoretical results. Experimental results show that the performance of the power landmark vector learning framework is comparable to traditional kernel learning frameworks.

Acknowledgements

I would like to sincerely thank Dr. Burkowski for the enormous help and encouragement he has generously offered me at each stage of my graduate education. It has been a great honour and privilege to work with Dr. Burkowski. I would also like to thank Dr. to be William Wong for all the help that he has offered me throughout this journey.

I would like to thank, in forward alphabetical order of their last names, Dr. Li and Dr. McConkey for spending their precious time to read and critique this work.

I would like to thank my parents for always being there when I need them the most.

I would like to thank Dr. Cohen, the Bioinformatics Group, the David R. Cheriton School of Computer Science and the University of Waterloo for the spirit and the opportunity.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

Finding patterns out of chaos has been one of our innate abilities ever since our ancestors first walked upon the Earth hundreds of thousands of years ago. The ability to spot the yellow and white patterns of the back of a tiger amid the chaotic background of criss-crossing branches and leaves had been integral to the survival of our primate ancestors in a primeval jungle. Modern quantitative physical sciences such as physics and astronomy also trace their origins to the attempts of Galileo and Kepler to deduce formulas and patterns out of physics experimental data and astronomical observation charts. Recently, humanity has acquired a powerful new tool — the computer — that allows it to deal with massive quantities of high dimensional data whose learning had previously been intractable to the human mind. This marriage of our hundred-thousand year old tradition of learning with our new-found computational capacities resulted in an exciting field called “machine learning”.

Biology started out as a qualitative and taxonomic science. One hundred years ago, when people were using physics to explain everything from the universe to the macroscopic behaviour of atoms, biologists were still asking simple questions such as “how many species” and “what is the difference between fungi and bacteria”. However, with the advent of computers and the discovery of the informational aspects of genes and proteins, biology also became a computational science. The completion of the human genome sequencing project (Venter et al., 2001) — the “Apollo project of life science” — heralds the coming of the age of computation for biology. The marriage of massive computational capabilities of computers and the in-the-large mathematical modeling of life processes and biological data has produced an exciting new field of study known as *bioinformatics*.

1.1 Kernel learning framework

There are two primary types of machine learning — linear machine learning and non-linear machine learning. Linear machine learning has a long history and many efficient algorithms have been proposed and implemented for linear machine learning tasks. The statistical and runtime properties of these algorithms have been thoroughly analyzed by mathematicians and computer scientists and are well understood. The most famous algorithms, such as the perceptron and the support vector machines, have already been in active use for quite a few decades.

Non-linear machine learning, on the other hand, suffers from the lack of a uniform theoretical framework and problem-solving methodology enjoyed by linear machine learning. Past efforts in non-linear machine learning have led to the creation of many *ad-hoc* non-linear learning algorithms, each of which are applicable to only its own narrow band of very specific problems. Furthermore, these algorithms frequently suffered from local-extrema that caused them to return inaccurate or entirely false results, and their statistical and run-time properties were very hard to analyze mathematically. Since useful machine learning in bioinformatics is of a non-linear nature, a new non-linear learning framework with good statistical and run-time behaviours is urgently needed.

The breakthrough came with the introduction of the kernel learning framework by Vapnik et al. at the start of the 1990s. The ingenuity behind the kernel learning framework is the marriage of linear learning algorithms and a non-linear learning framework. A new class of mathematical functions, known as kernels, was employed that could transform non-linear relationships into linear relationships in a higher dimensional space. Thus linear learning algorithms could be easily applied to this higher dimensional space to detect those linear relationships, which can be interpreted as non-linear relationships in the original space. Therefore, when confronted with non-linear learning, rather than allowing decades of research into linear learning algorithms go to waste, the kernel learning framework cleverly recycles linear learning algorithms and allows them to work on non-linear relationships in an almost-transparent way, *almost*, but not totally transparent, since the kernel learning framework still has to modify the linear learning algorithms into a form known as the “dual form” before they can be applied to non-linear learning. A problem can now be posed: is it possible to apply linear learning algorithms, without any modification and in a totally transparent manner, to the set of training data exhibiting non-linear relationships?

1.2 Previous work

In his seminal paper (Balcan and Blum, 2005) mathematician Avrim Blum proposed a novel alternative to traditional kernel learning framework. Blum proposed that unlabeled vectors belonging to the same distribution and the same input space as the training vectors be used as a kind of *landmark* in the generation of a new space in which the non-linear relationships that exist between the training vectors in the original space becomes linear. The difference, compared to traditional kernel learning framework, is that the new space in Blum's framework can be *explicitly* accessed by any linear learning algorithm. Hence there is no need to modify linear learning algorithms and they can be applied, in a totally transparent manner, to the learning of non-linear relationships.

1.3 Research description

This thesis extends Blum's work by introducing additional classes of vectors into the set of landmark vectors that can be used. This thesis also justifies, from a theoretical perspective, why the use of these additional classes of vectors as landmark vectors is mathematically sound. Specifically, this thesis introduces a new learning framework called the Power Landmark Vector Learning Framework that extends Blum's framework by allowing Gaussian random vectors and input-space orthonormal bases vectors, *in addition* to the original unlabeled vectors belonging to the same distribution and the same input space as the training vectors, to be used as landmark vectors in the generation of the new linear learning space. The Power Landmark Vector Learning Framework is flexible and easy to use. A comprehensive theoretical foundation for the various mathematical properties of the framework is established in this thesis to provide some guarantees on the framework's performance. Finally, the framework has potential future applications in dimensionality reduction and data visualization.

1.4 Thesis overview

This thesis presents a comprehensive theoretical formulation of the Power Landmark Vector Learning Framework and applies it to the Gaussian Kernel and finite-dimensional Euclidean input spaces. Experiments are provided to demonstrate the soundness of some of the theoretical arguments and to show Power Landmark Vector Learning Framework working with real bioinformatics data. This chapter has presented a general introduction to the field of machine learning and bioinformatics. Chapter 2 covers basic background materials necessary for an understanding and appreciation of the nature of machine learning and the traditional kernel learning framework and introduces Blum's landmark vector learning framework. Chapter 3 presents the main theoretical contribution of this thesis in the form of a comprehensive theoretical formulation of the Power Landmark Vector Learning Framework with proofs of some key results. Experimental results and discussions are provided in Chapter 4. Finally, conclusions and directions for future research are given in Chapter 5.

Chapter 2
Background

2.1 Overview

The learning of patterns exhibited by massive amounts of numerical data compiled from physical experiments and observations has been as old as natural science itself. Galileo Galilei deduced his physics equations by deriving the numerical relationships between the various time, displacement, initial velocity and final velocity Figures he had recorded for the many experiments he had performed. Johannes Kepler formulated his three laws of planetary motion from years of painstaking work he did on discerning patterns in the numerical data recording the positions of planets compiled by Tycho Brahe. (Shawe-Taylor and Cristianini, 2004). Perhaps it is not an exaggeration to say that one of the major reasons natural science came to its present-day level of sophistication is the human mind's capacity to discern pattern from numbers.

However, human minds are not without their limits. When it comes to certain problems in pattern recognition and data learning, our senses can only take us so far. Imagine the difficulty involved in discerning the 2^{16} = 65,536 vectors in a sixteen-dimensional Euclidean space as belonging to the vertices of a sixteen-dimensional hypercube rotated slightly about the origin. Imagine the difficulty involved in using pencil and paper to find out the differences between microarray readouts for healthy cells and cancerous cells, when the data for each type of cell comprises hundreds of vectors, each of which is further contains thousands of numbers. Clearly, when faced with the daunting task of discovering patterns within data gathered from modern science experiments, numerical data that are both massive in quantity and high in dimensionality, the human mind falls far short of the goal of even comprehending the meaning of the data, let alone finding any patterns hidden among them.

With the advent of modern digital computers, we have a new technology at our disposal to quickly and accurately discover patterns hidden in data. The field of computer science that applies the computer to the task of discerning numerical patterns from massive amounts of data is known as “machine learning”. Its application to bioinformatics data forms the focus of this thesis.

2.2 The perceptron

In the early days of computer science, an efficient algorithm of machine learning known as the “Perceptron” was used (Rosenblatt, 1958) (Minsky and Papert, 1969). The perceptron algorithm learns a decision hyperplane separating the positive examples from the negative examples by starting with a rough estimate of where the decision hyperplane is going to be with one pair of positive and negative examples. It then refines its choice of the decision hyperplane by updating the weight vector of the decision hyperplane as new sample vectors of the positive and negative set become available (Figure 1). The updating is done in such a way that the decision hyperplane remains the dividing line between the positive and negative sample vectors for the currently given set of positive and negative samples. The “decision hyperplane" takes the form of a regular line and a regular plane in $\mathbb{R}^2$ and $\mathbb{R}^3$ spaces respectively.

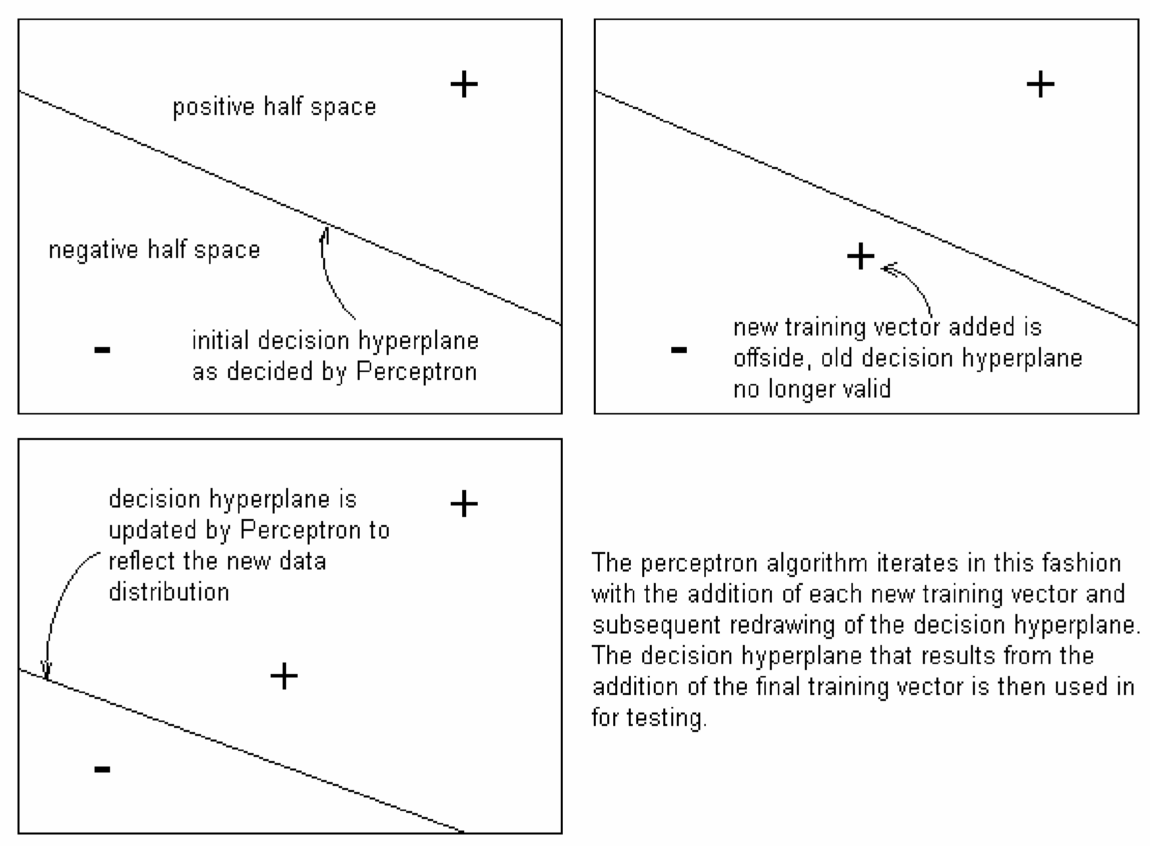
+
positive half space
negative half space
initial decision hyperplane
as decided by Perceptron
-
+
+
new training vector added is
offside, old decision hyperplane
no longer valid
-
decision hyperplane is
updated by Perceptron to
reflect the new data
distribution
+
+
-
The perceptron algorithm iterates in this fashion
with the addition of each new training vector and
subsequent redrawing of the decision hyperplane.
The decision hyperplane that results from the
addition of the final training vector is then used in
for testing.

Figure 1: A simple demonstration of the perceptron algorithm.

However, the perceptron algorithm offers no guarantees on the quality of the separating hyperplane it produces at the end of the learning process. The perceptron algorithm yields correct results as long as the decision hyperplane is able to divide the sample space into two half-spaces, each one of which contains only one class of the training data. It works properly even if the hyperplane is cutting very close to some of the training data vectors. Hence the perceptron algorithm sometimes performed poorly when given actual test data. In machine learning this characteristic of the perceptron algorithm is called its “poor generalization capacity”. The poor generalization capacity of the perceptron algorithm prompted researchers to look for better algorithms that could provide a guarantee for the quality of the decision hyperplane they have learned. The most famous algorithm to accomplish this purpose is the support vector machine (SVM) algorithm (Duda and Hart, 1973; Cover, 1965; Smith, 1968; Vapnik and Lerner 1963; Vapnik and Chervonenkis 1964).

2.3 SVM

As shown in Figure 2, given two classes of training vectors, the SVM algorithm attempts to learn the decision hyperplane that carries the maximum “margin”. Margin is defined here to be the distance from the decision hyperplane to the nearest positive or nearest negative vectors. The vectors that determine the margin of the decision hyperplane are known as “support vectors”, hence the name “support vector machine”. Aside from quantifying the “quality” of the separating hyperplane, the SVM allows the added benefit that the resulting hyperplane maybe expressed as a linear combination of only the support vectors, while the rest of the training set maybe discarded after the learning is done. This allows a significant saving in the storage space requirements of the final decision hyperplane when it is applied to subsequent test data. Note that the decision hyperplane is always positioned such that the distance from the hyperplane to the nearest positive vector is equal to the distance from the hyperplane to the nearest negative vector. In practice, noise present within the training data will often make the data non-linearly separable (Figure 3). Therefore the SVM algorithm has to tolerate the presence of negative training vectors in the positive halfspace and the presence of positive training vectors in the negative halfspace (Figure 3). These vectors, located on the "wrong side" of the decision hyperplane, are referred to as the "slack vectors" and their distances to the decision hyperplane are stored in the "slack variables". The use of the margin concept quantified the quality of the decision hyperplane learned by the SVM algorithm, which makes the SVM one of the best and most-utilized algorithms of machine learning.

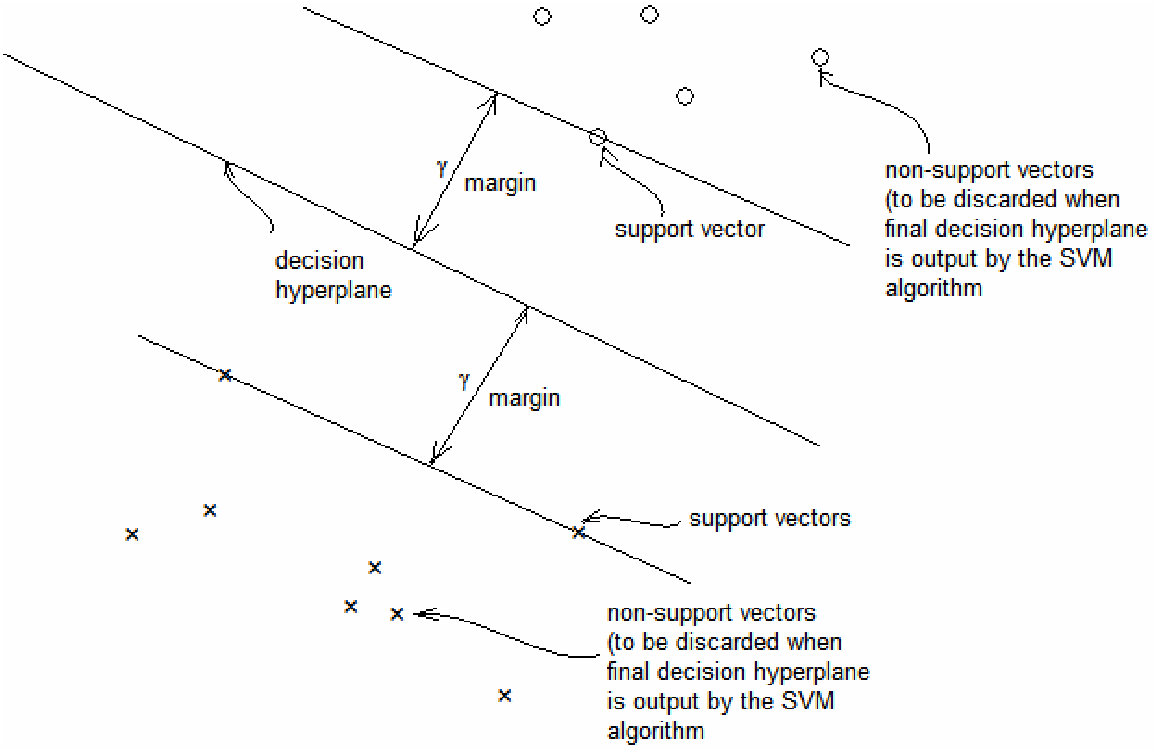

Figure 2: A typical 2-dimensional hard-SVM with the decision hyperplane being a line that divides the plane into two half-planes.

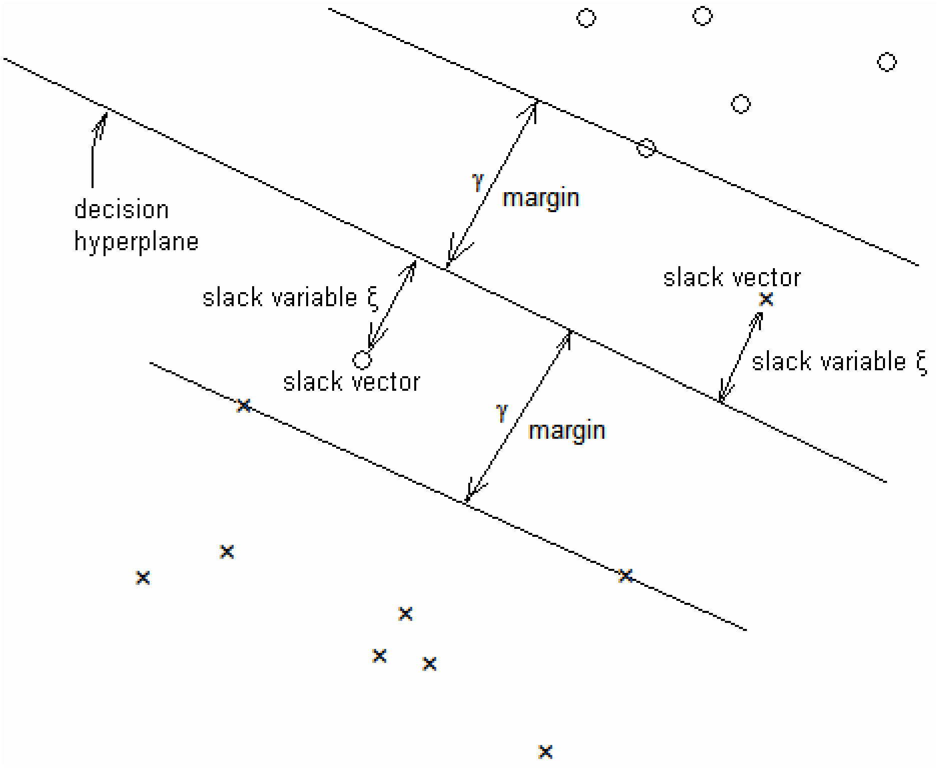

Figure 3: A typical planar soft-SVM with off-side slack vectors and their associated slack variables shown.

Both the perceptron and the SVM suffered from the fact that they were only capable of learning a *linear* decision hyperplane from the dataset they were given. If the positive and negative data were separated by a boundary that does not take the form of a straight line or something that approximates a straight line, then perceptrons and SVMs are unable to learn anything useful from the data. Figure 4 is an example of a non-linear boundary. In Figure 4, the x's (representing positive sample data) are separated from the o's (representing negative sample data) by a non-linear decision boundary. Since most machine-learning problems that arise in natural science are related to learning non-linear decision boundaries separating two or more classes of data, a method of non-linear learning is urgently needed to make machine-learning useful across a broad spectrum of learning tasks.

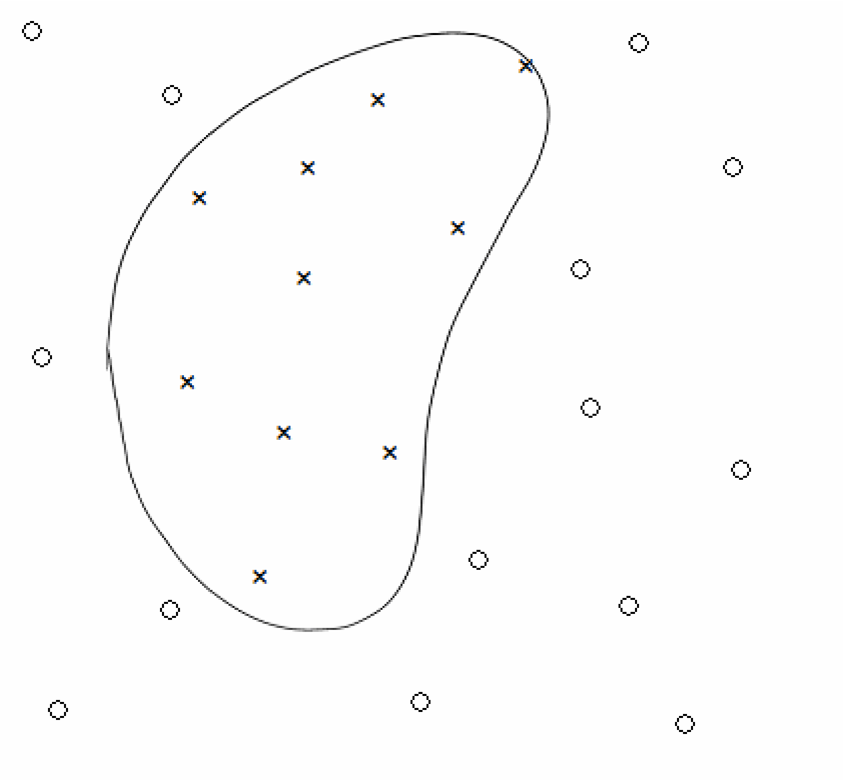

Figure 4: An example of a non-linear decision boundary. [1]

One immediate solution is to propose *ad hoc* non-linear algorithms that are learning problem specific. In this case, one non-linear learning problem would be carefully scrutinized and studied by a group of computer scientists and domain experts, who would then propose one or more non-linear learning algorithms. These non-linear learning algorithms are specific to

[1] This picture is hand-drawn using a tablet computer provided at the courtesy of Chris Long, who is a second year undergraduate computer science student at the University of Waterloo at the time this thesis was written.

one learning problem and could not be applied to any other learning problem. Furthermore, *ad hoc* non-linear learning algorithms often lacked statistical and probabilistic guarantees on their performance and suffered from local minima (Figure 5). This severely curtailed their usefulness in machine learning problems demanding high degrees of optimality and stability. On the other hand, the statistical and computational behaviours of perceptrons and SVMs were well understood (Rosenblatt, 1958; Fisher, 1986), and, being linear (hence convex) algorithms, they suffered no local minima. Therefore, researchers began to wonder if there existed a good way of adapting these linear learning algorithms to the task of non-linear learning.

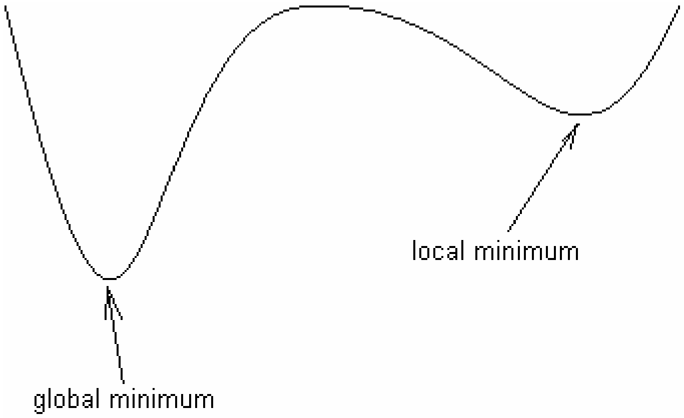

Figure 5: global minimum is the overall lowest point of the curve, while local minimum is lower than all of its immediate surroundings. Note that *ad-hoc* non-linear learning algorithms, not knowing where the global minimum is located, could end up stuck in local minima and obtain sub-optimal results

2.4 Kernels

The breakthrough came in the 1990s with the "kernel trick" (Boser et al. 1992). Kernels were originally classes of symmetric and positive-semi definite functions of two arguments used in functional analysis to facilitate the solution of certain classes of integral equations. They were first mentioned in Mercer's classical paper (Mercer, 1909), and were first formally introduced, in the functional analysis sense, in the seminal paper "Theory of Reproducing Kernels" by Aronszajn in the 1940s (Aronszajn, 1950). The finitely positive semi-definite properties of kernels were expounded upon in Saitoh's theorem (Saitoh, 1988). In modern machine learning, kernels first appeared in the works of Aizermann, Bravermann and Rozoener (Aizermann et al. 1964). The concept was subsequently reintroduced to the modern machine learning community through the works of Boser, Guyon and Vapnik (Boser et al. 1992).

The primary motivation in a kernel's ability to transform a non-linear relationship to a linear relationship lies in its implicit use of a ϕ function that *maps* non-linear data in the input space into a *feature space* with a higher number of dimensions in a process known as *recoding*. To understand why non-linear data will exhibit linear behaviours in higher dimensions, the simple example of a ϕ function given below is examined. The ϕ function below maps vectors in a two dimensional Euclidean input space into a three dimensional Euclidean feature space according to the following explicit form

$$\phi(\rho\cos\theta, \rho\sin\theta) = ((\rho\cos\theta)^2, \sqrt{2}\ \rho^2\cos\theta\sin\theta, (\rho\sin\theta)^2). \qquad (1)$$

As we can see from Figure 6, before the ϕ-map, the vectors in the two circles can only be separated by a non-linear decision boundary in the form of another circle. However, after the ϕ-map, the vectors on the two circles can now be separated by a linear decision boundary in the form of a plane. Therefore, the ϕ-map is capable of transforming a non-linear relationship into a linear relationship.

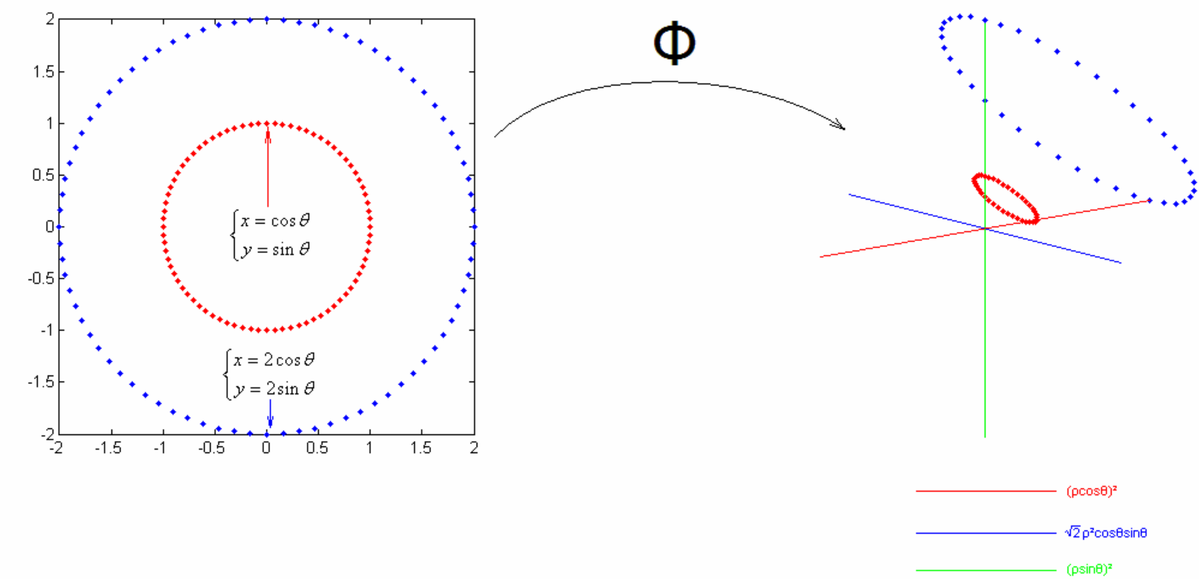

Figure 6: The two circles mapped from input space to feature space, plotted by MATLAB.

However, the ϕ function will often map input-space vectors into a feature space with an extremely large number of dimensions. Sometimes the feature space will even reach an infinite number dimensions. Under these circumstances, the direct application of existing linear learning algorithms to mapped vectors in the feature space quickly becomes unwieldy.

Fortunately, such direct applications are not necessary. The ϕ mapping function and the feature space vectors are not actually used explicitly in the learning process. Instead, all learning is done based on computing the inner-products between pairs of feature vectors which are the ϕ-mappings of input space vectors into the feature space. Existing linear learning algorithms such as the perceptron and the SVM can be rewritten in such a way that they do not directly access the input vectors themselves but rather use only the inner products between pairs of input vectors to do their learning. The inner products between pairs of feature space input vectors are known as *kernels*. As an example of kernels, consider the mapping in equation (1). This mapping has the effect of transforming a quadratic, and hence non-linear, relationship in an $\mathbb{R}^2$ input space to a linear relationship in an $\mathbb{R}^3$ feature space. Now consider calculating the inner product of the ϕ mapped three dimensional vectors of two vectors x and z in the feature space.

$$\langle \phi(\rho_1\cos\theta, \rho_1\sin\theta), \phi(\rho_2\cos\theta, \rho_2\sin\theta)\rangle$$

$$= \langle((\rho_1\cos\theta)^2, \sqrt{2}\ \rho_1\cos\theta\, \rho_1\sin\theta, (\rho_1\sin\theta)^2), ((\rho_2\cos\theta)^2, \sqrt{2}\ \rho_2\cos\theta\rho_2\sin\theta, (\rho_2\sin\theta)^2)\rangle$$

$$= (\rho_1\cos\theta)^2(\rho_2\cos\theta)^2 + 2\rho_1^2\cos\theta\sin\theta\rho_2^2\cos\theta\sin\theta + (\rho_1\sin\theta)^2(\rho_2\sin\theta)^2$$

$$= (\rho_1\rho_2\cos\theta\cos\theta + \rho_1\rho_2\sin\theta\sin\theta)^2$$

$$= \langle(\rho_1\cos\theta, \rho_1\sin\theta), (\rho_2\cos\theta, \rho_2\sin\theta)\rangle^2$$

We come to the surprising discovery that the complicated inner product of two vectors mapped into a three-dimensional feature space turned out to be simply the square of the inner product of the same two vectors calculated directly in the input space. Thus, kernels have very convenient representations in the input space. In fact, it is these input space representations of kernels that allow for very simple calculations in the transformation of non-linear relationships into linear relationships. Given linear algorithms that have been modified so that they only work with inner-products between input vectors, these inner products could then be trivially recoded with the kernel to immediately and transparently enable the modified linear algorithms to work on non-linear learning with all the statistical guarantees of their performance and accuracy in place and with no local minima problems. Moreover, the calculation of kernel values is often orders of magnitudes less complex than directly calculating the feature vectors using the ϕ-map and finding their inner product in a naïve algebraic way in the feature space. Thus the kernel method allows for a method of doing non-linear learning with a linear learning algorithm in a high-dimensional space without having to pay the corresponding price of a full ϕ-mapping of low-dimensional input-space vectors into the high-dimensional feature space. It is this saving in computational costs that makes kernel methods the enabling tool in non-linear machine learning.

2.5 The Blum landmark vector learning framework

Under traditional kernel learning frameworks, only the set of labeled training vectors are available to a kernel learning algorithm. Therefore, the weight vector of the decision hyperplane that is ultimately learned by the kernel learning algorithm in the feature space is the linear combination of only the set of ϕ-mapped feature space vectors corresponding to the set of labeled training vectors in the input space. This traditional framework suffers from a drawback: the feature space is kept implicit behind the kernel, so the user has no idea what the training vectors would look like after they have been mapped into the feature space of the kernel. To put this problem in a mathematical perspective, consider the quadratic kernel $K(x, y) = \langle x, y\rangle^2$.

From the previous section we know that the quadratic kernel maps vectors in an $\mathbb{R}^2$ input space into an $\mathbb{R}^3$ feature space. The corresponding mapping function ϕ is defined as follows:

$$\phi(x_1, x_2) = (x_1^2, \sqrt{2}\ x_1x_2, x_2^2) \qquad (2)$$

However, a linear classification algorithm working under traditional kernel learning framework only works with vectors in the $\mathbb{R}^2$ input space by taking the input space inner-product between two vectors and squaring that inner-product as per the quadratic kernel function. The linear classification algorithm does not know about what the vectors will be after they have been mapped into the feature space according to equation (2). In the example we have used here it is still possible, if we really desired it, to find out how the input space vectors are actually distributed in the feature space. On the other hand, if we are working with kernels whose feature space has a large or infinite number of dimensions, then it is no longer possible to calculate the explicit coordinates of vectors in the feature space. The incomputable feature space vectors make traditional kernel framework unwieldy for visualizing the linear distribution of the input-space vectors in the feature space. It also prevents further analyses of particular mathematical or statistical properties of the feature space beyond those provided by the kernel. More importantly, it requires existing linear learning algorithms to be rewritten in the “dual-form” since the linear feature space is not explicitly available for the primal forms of the linear learning algorithms. A mathematical example of this recoding will be given in the next remark. Thus a more transparent framework employing the kernel function is needed.

In (Blum, 2006) the authors proposed one such novel kernel learning framework that allows for transparent learning and has an explicit and computable feature space. This novel framework makes use of the potentially vast number of unlabeled vectors belonging to the same distribution and the same input space as the training vectors in the input space. Given a set of labeled training vectors $\{x^{(1)}, x^{(2)}, \ldots, x^{(\mathrm{n})}\}$ and a set of unlabeled training vectors $\{z^{(1)}, z^{(2)}, \ldots, z^{(d)}\}$ both of which belongs to a probability distribution called P, let a function $F{:}P \rightarrow FP$ be defined in the following way:

$$F(x) = \begin{pmatrix} K(x, z^{(1)}) \\ \vdots \\ K(x, z^{(d)}) \end{pmatrix} \tag{3}$$

where $x \in \{x^{(1)}, x^{(2)}, \ldots, x^{(n)}\}$ is a labeled training vector. We can see that the function F maps a labeled training vector from the original input space and probability distribution into an $\mathbb{R}^d$ feature space called the FP space. Furthermore, the mapped vectors in the FP feature space are explicitly accessible as their components are computed as the kernel evaluations between a training vector and each of the unlabeled vectors belonging to the same distribution and the same input space as the training vectors. Since the margin between the feature space vectors in the FP feature space is now made linear as proven by (Blum, 2006), one may now trivially apply any ordinary linear learning algorithm such as the perceptron and SVM *in their primal forms without any modification* to the FP feature space to do the needed learning. The linear learning done by these algorithms in the FP feature space is then transparently translated back into non-linear learning done in the input space by Blum's landmark vector learning framework.

Remark Notice that Blum's landmark vector learning framework acts as a *preprocessor* that hides the complexities of the kernel learning framework from linear learning algorithms that were invented before kernels. The traditional kernel learning framework, as illustrated in (Shawe-Taylor and Cristianini, 2004), demands that traditional linear learning algorithms be *modified* to use only inner-products between pairs of input vectors during their execution. For example, consider the ridge regression algorithm discussed in (Shawe-Taylor and Cristianini, 2004).

The problem that the ridge regression algorithm sets out to solve is the finding of a linear function

$$g(x) = \langle w, x \rangle = \sum_{i=1}^{n} w_i x_i \tag{4}$$

that would best interpolate a given training set $S = \{(x^{(1)}, y_1), \ldots, (x^{(l)}, y_l)\}$ of vectors $x^{(i)} = (x_1{}^{(i)}, x_2{}^{(i)}, \ldots, x_n{}^{(i)})$ from set $X \subseteq \mathbb{R}^n$, where $x_j{}^{(i)}$ denotes the j^{th} component of vector $x^{(i)}$, with corresponding labels y_i from set $Y \subseteq \mathbb{R}$. Arrange the set of vectors x_i into the following matrix form:

$$X = \begin{pmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(l)} \end{pmatrix} = \begin{pmatrix} x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{(l)} & x_2^{(l)} & \cdots & x_n^{(l)} \end{pmatrix}$$

It becomes apparent that in order for equation (4) to best interpolate S it is necessary to minimize the following difference:

$$L(w, S) = \| y - Xw \|^2 = (y - Xw)^{\mathrm{T}}(y - Xw) \tag{5}$$

where the function $L(w, S)$ is called the “loss function” and serves as a measure of the amount that the predicted labels generated by equation (4) deviates from the true set of labels Y.

In order to find the weight vector *w* that will minimize equation (5) it is necessary to differentiate equation (5) with respect to *w*, we have

$$\frac{\partial L(w,S)}{\partial w} = -2X^{\mathrm{T}}y + 2X^{T}Xw = 0$$

moving -2$X^{\mathrm{T}}y$ to the right hand side of the above equation and canceling out the coefficient, we have

$$X^{\mathrm{T}}Xw = X^{\mathrm{T}}y \tag{6}$$

Solving for w in the above equation will yield the linear function of equation (4)'s form that will best interpolate set S. However, the inverse of $X^{\mathrm{T}}X$ does not always exist or may be ill-conditioned. Therefore it is necessary to introduce a "regularizer" to $X^{\mathrm{T}}X$ to ensure that it will always have a well-behaved inverse and at the same time it will closely approximates the original $X^{\mathrm{T}}X$ matrix itself. We introduce this "regularizer" by adding a constant λ to each entry on the main diagonal of $X^{\mathrm{T}}X$ and rewrite equation (6) as

$$(X^{T}X + \lambda I)w = X^{\mathrm{T}}y \tag{7}$$

where *I* is an identity matrix whose dimension matches that of $X^{T}X$. The final weight vector may now be computed as:

$$w = (X^T X + \lambda I)^{-1} X^{\mathrm{T}} y \qquad (8)$$

This is the "primal form" for the ridge regression algorithm because in order to calculate *w* it is necessary to explicitly access the training vectors themselves due to the presence of terms in equation (8) such as $X^T X$ and X^{T} which are not related to inner-products between pairs of training vectors. In order to enable ridge regression algorithm to work under traditional kernel learning frameworks, it is necessary to spend additional effort to rewrite the ridge regression algorithm so that it works with *only* the inner products between pairs of training vectors.

To transform the ridge regression algorithm into a dual-form, we go back to equation (7) and start from there

$$(X^T X + \lambda I) w = X^{\mathrm{T}} y$$

Solving for *w*, we get:

$$w = \lambda^{-1} X^{\mathrm{T}} (y - Xw) = X^{\mathrm{T}} \alpha \qquad (9)$$

The last expression shows that *w* can be written as a linear combination of training vectors with the weight vector being $\alpha = \lambda^{-1} (y - Xw)$. Hence, we have

$$\alpha = \lambda^{-1} (y - Xw)$$

$$\lambda\alpha = y - Xw$$

$$\lambda\alpha = y - XX^{\mathrm{T}}\alpha$$

$$\lambda\alpha + XX^{\mathrm{T}}\alpha = y$$

$$(\lambda I + XX^{\mathrm{T}})\alpha = y$$

$$(\lambda I + \mathrm{G})\alpha = y$$

$$\alpha = (\mathrm{G} + \lambda I)^{-1} y \tag{10}$$

The matrix G is called Gram's matrix, and its components are defined as $G_j^{(i)} = \langle x^{(i)}, x^{(j)} \rangle$. Therefore Gram's matrix consists purely of inner-products between pairs of training vectors. Since equation (10) accesses training vectors only through a Gram matrix, the ridge regression algorithm has now been rewritten in the dual form.

The need to rewrite linear learning algorithms in dual forms is inconvenient. Blum's landmark vector learning framework obviates the need of such rewrites by preprocessing, using a kernel, the training vectors into a new linear feature space that can be explicitly accessed by the unmodified primal forms of linear learning algorithms. If the ridge regression algorithm is applied under Blum's landmark vector learning framework, then all the work that goes into the derivation of equations (9) and (10) is unnecessary. The primal form of ridge regression algorithm in equation (8) can be directly applied to non-linear data by defining the training matrix X as

$$X = \begin{pmatrix} F(x^{(1)})^T \\ F(x^{(2)})^T \\ \vdots \\ F(x^{(l)})^T \end{pmatrix} = \begin{pmatrix} K(x^{(1)}, z^{(1)}) & K(x^{(1)}, z^{(2)}) & \cdots & K(x^{(1)}, z^{(d)}) \\ K(x^{(2)}, z^{(1)}) & K(x^{(2)}, z^{(2)}) & \cdots & K(x^{(2)}, z^{(d)}) \\ \vdots & \vdots & \ddots & \vdots \\ K(x^{(l)}, z^{(1)}) & K(x^{(l)}, z^{(2)}) & \cdots & K(x^{(l)}, z^{(d)}) \end{pmatrix}$$

A final concept that we need to cover before introducing the core results from (Blum, 2006) is the PAC learning model. The PAC learning model is used throughout (Blum, 2006). A good and easy-to-understand introduction to the PAC learning model is found in (Rathi, 2005).

PAC Learning Model (Rathi) Let *C* be a class of boolean functions $f: X \rightarrow$ {-1, +1}. We say *C* is *PAC-learnable* if there exists an algorithm *L* such that for every $f \in$ *C*, any probability distribution *D*, any ε where 0 ≤ ε < ½ and any δ where 0 ≤ δ < 1, algorithm *L* on input ε and δ and a set of random examples picked from any probability distribution *D* outputs at least with a probability 1 – δ, a function *h* : $X \rightarrow$ {-1, +1} such that error(*h*, *f*) ≤ ε.

Efficient PAC Learning Model (Rathi) We say that *C* is *efficiently PAC-learnable* if *C* is *PAC-learnable* and

- the number of examples that *L* takes is bounded by some polynomial in *n*, $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$
- *L* runs in time asymptotically bounded by some polynomial in *n*, $\frac{1}{\varepsilon}$ and $\frac{1}{\delta}$.

$$d \geq \frac{8}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln\frac{1}{\delta}\right]$$

used throughout (Blum, 2006) to express the number of unlabeled examples drawn i.i.d. from the distribution underlying the set of labeled training examples.

With the required preliminary concepts and definitions covered in this and previous sections of this chapter, we now present two fundamental results from (Blum, 2006)

Lemma (Blum) Consider a set S of labeled examples in Euclidean space such that there exists a linear separator defined by the weight vector w that separates the set S with margin γ and error 0. If we draw $d \geq \frac{8}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln\frac{1}{\delta}\right]$ examples $z^{(1)}, \ldots, z^{(d)}$ i.i.d. from the distribution underlying S, then with probability ≥ 1 – δ, there exists a vector w' in span($z^{(1)}, \ldots, z^{(d)}$) that defines a hyperplane that separates the set S with margin γ/2 and error at most ε.

This lemma implies that if a set of training vectors is linearly separable with margin γ under a kernel, and we draw $d = \frac{8}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln\frac{1}{\delta}\right]$ unlabeled vectors $z^{(1)}, \ldots, z^{(d)}$ from the same distribution as the labeled training vectors we started out with, then with probability at least 1 – δ there is a linear decision hyperplane, with w as its weight vector, in the feature space with error rate at most ε that can be written as the linear combination of the ϕ-mapped feature space vectors of the chosen unlabeled vectors $z^{(1)}, \ldots, z^{(d)}$ in the form below:

$$w = \sum_{i=1}^{d} \alpha_i \phi(z^{(i)}) .$$

According to definition of kernels, $\langle w, \phi(x) \rangle = \alpha_1 K(x, z^{(1)}) + \ldots + \alpha_d K(x, z^{(d)})$. An immediate implication is that if we treat $K(x, z^{(i)})$ as the i^{th} "feature" of x in the FP feature space constructed by equation (3), then with high probability the vector $(\alpha_1, \ldots, \alpha_d)$ is an approximate weight vector for a decision hyperplane in the FP space that will linearly discriminate between the ϕ-mapped vectors of the original training vectors. This leads to the following result.

Corollary (Blum) If a set S of labeled training vectors becomes linearly separable after being mapped into the feature space associated with a kernel function K, then with probability at least $1 - \delta$, if $d = \frac{8}{\varepsilon}\left[\frac{1}{\gamma^2} + \ln\frac{1}{\delta}\right]$ unlabeled vectors $z^{(1)}, \ldots, z^{(d)}$ are drawn from the distribution underlying S, the mapping

$$F(x) = \begin{pmatrix} K(x, z^{(1)}) \\ \vdots \\ K(x, z^{(d)}) \end{pmatrix}$$

applied to the set S will produce a new feature space FP in which the set S becomes linearly separable with error at most ε.

2.6 Random projection

Random projection is a mathematical technique commonly used for dimensionality reduction. A very famous lemma by Johnson and Lindenstrauss (Johnson and Lindenstrauss, 1984) states that when vectors from a high-dimensional space are projected into a low-dimensional space, distances and angles between all pairs of vectors are approximately preserved (Figure 7). The Johnson-Lindenstrauss lemma quantified the degree to which the approximation took place and fixed it in relation to the other parameters that characterized the original high-dimensional vectors. Numerous proofs have since been given for this observation (Frankl and Maehara, 1988; Indyk and Motwani, 1998; Dasgupta and Gupta, 1999), with at least one of them being elementary (Dasgupta and Gupta, 1999).

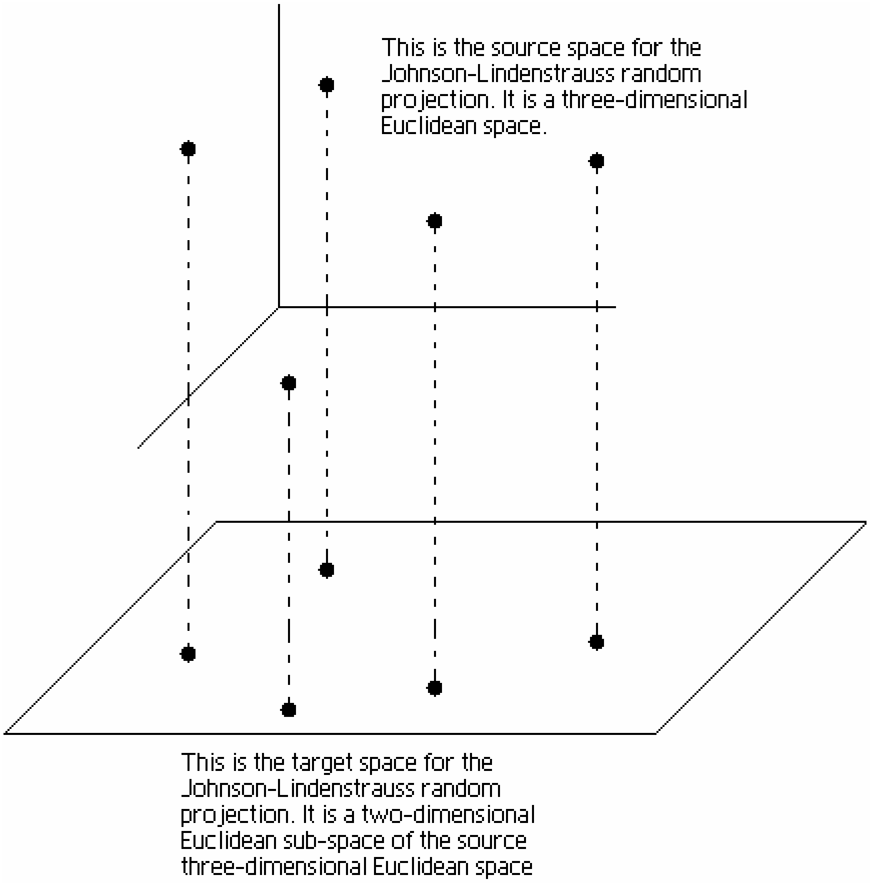
This is the source space for the Johnson-Lindenstrauss random projection. It is a three-dimensional Euclidean space.
This is the target space for the Johnson-Lindenstrauss random projection. It is a two-dimensional Euclidean sub-space of the source three-dimensional Euclidean space

Figure 7: A Johnson-Lindenstrauss random projection from three-dimensional space onto a plane. Note that the relative distances and orientations between the vectors are approximately preserved in the random projection.

2.7 Chapter summary

This chapter introduced various background materials that are necessary for a good understanding of the next chapter. The origins of learning and machine learning were discussed. Various classical machine learning algorithms such as the Perceptron and SVM are introduced and their operating principles discussed and illustrated. The rationale for a uniform non-linear learning framework was introduced at the end of the discussions on SVM.

Kernels were presented immediately afterwards as a viable framework for uniform non-linear learning. A detailed examination of a simple kernel was presented to clarify the various concepts associated with kernel learning inner products and the ϕ-mapping function. All of these ideas lead to the introduction of Blum's landmark vector learning framework that presents a novel alternative to traditional kernel learning frameworks. Finally the idea of random projection is introduced that forms the prerequisite knowledge for the linear kernel proof discussed in Chapter 3.

Chapter 3
New Theories and Designs

3.1 The power landmark vector learning framework

This section describes an original extension, proposed by this thesis, to Blum's landmark vector learning framework. The new learning framework is called "the Power Landmark Vector Learning Framework".

Before proceeding further the concept of "landmark vectors" needs to be defined as it is very important to the main contribution of this thesis. Simply put, landmark vectors are vectors that belong to the same input space as the set of labeled training vectors given for a learning problem, but are distinct from any vectors in the training set.

Example The set of unlabeled vectors $\{z^{(1)}, \ldots, z^{(d)}\}$ are landmark vectors belonging to a probability distribution P used in Section 2.5.

In the previous section we looked at Blum's landmark vector learning framework. We learned that we could construct an FP feature space for a given training set X using the following function F : $P \rightarrow FP$

$$F(x) = \begin{pmatrix} K(x, z^{(1)}) \\ \vdots \\ K(x, z^{(d)}) \end{pmatrix}$$

where $x \in X$, K is a kernel function and $\{z^{(1)}, \ldots, z^{(d)}\}$ is the set of landmark vectors. In the new feature space the coordinates for the feature vectors could be explicitly computed and accessed.

We can then proceed to run an ordinary linear learning algorithm in this new feature space to directly learn the linear decision hyperplane separating the positive and negative vectors, perform dimensionality reduction to visualize the feature space itself, or study other mathematical and statistical properties of the feature space.

A key concept underlying Blum’s landmark vector learning framework is the ability of the unlabeled vectors to express the linear combination needed to characterize the weight vector of the decision hyperplane properly. This argument is intuitively sound. Blum and Balcan further utilized probabilistic arguments to argue that this is in fact the case. However, a question was left at the end of Blum’s work concerning the necessity of having the landmark vectors belong to the same probability distribution as the training vectors. This section addresses this question and extends Blum’s landmark vector learning framework so that the resulting new framework is capable of accepting other types of landmark vectors that are not dependent on the probability distribution of the landmark vectors.

Key Concept If the weight vector of a decision hyperplane in the feature space induced by a kernel is expressed as

$$w = \sum_{i=1}^{n} \alpha_i \phi(x^{(i)})$$

where $\alpha_i \in \mathbb{R}$ and $\{x^{(1)}, \ldots, x^{(n)}\}$ is the training set, then with high probability the weight vector of the decision hyperplane can be equivalently expressed as

$$w = \sum_{i=1}^{d} \beta_i \phi(z^{(i)})$$

where $\beta_i \in \mathbb{R}$ and $\{z^{(1)}, \ldots, z^{(n)}\}$ is the set of unlabeled landmark vectors. The β_i's are related to the α_i's through the following equation:

$$\beta_i = \left\langle \sum_{i=1}^{n} \alpha_i \phi(x^{(i)}), \phi(z^{(i)}) \right\rangle \tag{11}$$

A key insight is that the weight vector in the feature space can be expressed as the linear combination of *more than one* set of feature space vectors. In traditional kernel learning problems, only the set of labeled training vectors is available to the kernel algorithm, thus the resulting decision hyperplane that is learned must *necessarily* be the linear combination of the ϕ-mapped feature space vectors of the labeled training vectors. This is due to the "what we learn can only come from what we already know" principle and the fact that the set of labeled training vectors is all that we know. If other types of vectors from the input space of the training vectors are used, then it would be natural for one to question whether those types of vectors are capable of spanning a subspace in the kernel feature space the approximates the weight vector spanned by the linear combination of ϕ-mapped training vectors. (Blum, 2006) has already shown that the ϕ-mapped vectors of the unlabeled vectors belonging to the same distribution and the same input space as the training vectors in the input space in the input space are capable of spanning such a subspace. We now propose the key question that extends Blum's argument and gives rise to the power landmark vector learning framework.

Key question proposed by this thesis If the weight vector of a decision hyperplane in the feature space induced by a kernel can be expressed as the linear combination of the set of ϕ-mapped feature space vectors that are the images of a set of labeled training vectors in the input space, then is it possible, with high probability, that the weight vector of the same decision hyperplane can be equivalently expressed as the linear combination of a set of ϕ-mapped feature space vectors that correspond to input-space Gaussian-random or orthogonal vectors that are independent of the training vector's distribution?

For kernels with a finite-dimensional feature space such as the quadratic kernel, one obvious strategy to guarantee the capture of the subspace containing the decision hyperplane by alternative sets of vectors other than the labeled training vectors is to find the vectors in the input space that, when mapped into the feature space through the ϕ-map, would yield a set of orthonormal bases for the feature space of the form (1, 0, 0, …. 0), (0, 1, …, 0), …, (0, 0, …, 1). As these artificial vectors do not exist naturally in the input space, we must start with the orthonormal bases in the feature space and "back-map" them back into the input space. This problem of back-mapping is known as the *kernel pre-image* problem. Research in the kernel pre-image problem has not advanced sufficiently to readily allow one to perform this type of computation (Schölkopf and Smola, 2002; Kwok and Tsang, 2003). Furthermore, for kernels with infinite dimensional feature spaces, such as the Gaussian kernel, this strategy simply fails to work. Therefore alternatives must be sought.

We will now present several theorems, corollaries, remarks and discussions that constitute a comprehensive theoretical description of the power landmark vector learning framework and some of the rationales behind its operation.

A baseline argument is that the set of landmark vectors, upon being mapped into the feature space through the ϕ-map, should not all be mapped to the same vector in the feature space, as is the case with the feature space of a pathological kernel we will examine later in this chapter. For this argument, the thesis introduces a novel result called the *Non-collapsible Principle*, which states that a set of vectors spanning an n-dimensional Euclidean space should be able to span a subspace of at least n-dimensions in the feature space of a kernel.

Non-collapsible principle A kernel that is good for the power landmark vector learning framework should preserve the dimensionality of the training set from the input space into the feature space.

Non-collapsible characterization If the ϕ-map of a given kernel is such that it contains, in its feature list, a repetition of all the dimensions of the input space, then the kernel is said to possess the *Non-collapsible characterization*.

Remark The non-collapsible characterization is a sufficient, but *not* necessary, condition for a kernel to satisfy the non-collapsible principle. This means that there will exist other ways for a kernel to satisfy the non-collapsible principle. The investigation of these alternative venues will be a piece of future work for future graduate students in computer science and mathematics.

Non-collapsible corollary for the Gaussian kernel If a set of *n* vectors span an *n*-dimensional space, then the ϕ-mapped feature space vectors of those *n* vectors will still span an *n*-dimensional subspace in the feature space of the Gaussian kernel.

Proof: If the *n* vectors are capable of spanning an *n*-dimensional space, then there exist no linear dependencies in this set of *n* vectors. That is, we let the set of *n* vectors be $\{x^{(1)}, \ldots, x^{(n)}\}$. Then there exists no sequence of real numbers $c_1, \ldots, c_{\mathrm{m}}$ $(1 \leq m < n)$ such that $c_1 x^{(i_1)} + \ldots + c_{\mathrm{m}} x^{(i_m)} = x^{(i')}$, where $i_1, \ldots, i_{\mathrm{m}}, i' \in \{1, 2, \ldots, n\}$ and $i_1 \neq \ldots \neq i_{\mathrm{m}} \neq i'$. We now show that no new linear dependencies will arise among this set of *n* vectors after they have been mapped into the feature space via the ϕ-map.

Let us suppose that a new linear dependency did arise in the set $\{\phi(x^{(1)}), \ldots, \phi(x^{(n)})\}$, that is, there now exists a sequence of real numbers $c_1, \ldots, c_m$ ($1 \leq m < n$) such that $c_1 \phi\left(x^{(i_1)}\right) + \ldots + c_m \phi\left(x^{(i_m)}\right) = \phi\left(x^{(i')}\right)$, we now examine these ϕ-mapped feature space vectors in detail.

Shawe-Taylor and Cristianini showed that the features of a vector in the feature space of the Gaussian kernel consisted of "all possible monomials of input features with no restriction placed on the degrees" (Shawe-Taylor and Cristianini, 2004). In the context of this proof, this means that the features of a ϕ-mapped vector will contain at least a repeat of all the individual input vector space coordinates of an x vector as its monomials with degree 1. That is, if the x vectors reside in a d-dimensional space, then we take an x vector out of the set, call it x^*, then $x^* = (x^*_1, x^*_2, \ldots, x^*_d)$, where x^*_i is the i^{th} coordinate of the vector x^* in the input space. According to Shawe-Taylor and Cristianini, $\phi(x^*)$ looks like the following:

$$\phi(x^*) = (1, x^*_1, x^*_2, \ldots, x^*_d, x^*_1 x^*_2, x^*_2 x^*_3, \ldots, x^*_d x^*_1,$$

$$x^{*2}_1, x^{*2}_2, \ldots, x^{*2}_d, x^*_1 x^*_2 x^*_3, \ldots, \prod_{k=1}^{j} x^*_{i_k}, \ldots)$$

Notice that the set of components of x^*: $x^*_1, x^*_2, \ldots, x^*_d$, are repeated in the components of $\phi(x^*)$. It is this region in the components of $\phi(x^*)$, which repeats the components of x^*, that we are interested in and will focus on. Since $c_1 \phi\left(x^{(i_1)}\right) + \ldots + c_m \phi\left(x^{(i_m)}\right) = \phi\left(x^{(i')}\right)$, we have:

$$\sum_{j=1}^{m} c_j \begin{pmatrix} \vdots \\ x_1^{(i_j)} \\ x_2^{(i_j)} \\ \vdots \\ x_d^{(i_j)} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ x_1^{(i')} \\ x_2^{(i')} \\ \vdots \\ x_d^{(i')} \\ \vdots \end{pmatrix}$$

where $x_l^{(i_j)}$ denotes the l^{th} feature of vector $x^{(i_j)}$ in the input space. Thus we may obtain, from the above equation, the following set of equations:

$$\sum_{j=1}^{m} c_j x_l^{(i_j)} = x_l^{(i')}$$

where $l = 1, \ldots, d$. Recombining these equations back into column vector notation, we have:

$$\sum_{j=1}^{m} c_j \begin{pmatrix} x_1^{(i_j)} \\ x_2^{(i_j)} \\ \vdots \\ x_d^{(i_j)} \end{pmatrix} = \begin{pmatrix} x_1^{(i')} \\ x_2^{(i')} \\ \vdots \\ x_d^{(i')} \end{pmatrix}$$

which gives us:

$$\sum_{j=1}^{m} c_j x^{(i_j)} = x^{(i')} \tag{12}$$

Hence, there exists a sequence of real numbers $c_1, \ldots, c_m$ such that equation (12) holds. However, the premise of the proof stated that there exists no sequence of real numbers $c_1, \ldots, c_m$ $(1 \leq m < n)$ such that (12) holds. Thus we have arrived at a contradiction. Therefore there will arise no new linear dependencies after the projection of the x vectors into feature space and the set of $\phi(x)$'s will span a full n-dimensional subspace inside the Gaussian kernel feature space and the original theorem holds. □

Corollary It is immediately evident that the non-collapsible theorem can be readily applied to the quadratic kernel. □

Now that we have some guarantees on the kind of subspace that can be formed by a set of landmark vectors mapped into the Gaussian kernel feature space, we want to know if this subspace can in fact span the actual weight vector, for all would be in vain if this were not the case. From linear algebra we know that vectors cannot span directions that are perpendicular, or *orthogonal*, to it. Thus, a good indicator of a vector's ability to characterize a vector space is whether the vector is *orthogonal* to that vector space. Mathematically, a vector x is orthogonal to a vector space V if for all vectors $v \in V$, $\langle x, v \rangle = 0$.

If a vector x is orthogonal to a vector space V, then x is useless in characterizing V because it will not add to any linear combination characterizing V. For the power landmark vector learning framework we would like the set of landmark vectors, upon being mapped into the feature space by the ϕ-map, to be *non-orthogonal* to the subspace containing the decision hyperplane spanned by the ϕ-mapped labeled training vectors. This way, the ϕ-mapped landmark vectors may span a subspace in feature space that could adequately contain the subspace of the decision hyperplane. For this purpose, this thesis introduces another novel result, the *Non-orthogonal Principle*.

Non-orthogonal Principle In other words, a kernel that is good for the power landmark vector learning framework should prevent two vectors that are non-orthogonal in the input space from becoming orthogonal in the feature space.

Non-orthogonal Characterization If a kernel returns large values when it is evaluated on a landmark vector and a training vector, or, barring that, if the kernel will not return 0 for valuation between any pair of landmark vector and training vector, then the kernel is said to possess the *non-orthogonal characterization*.

Remark As in the non-collapsible case, the non-orthogonal characterization is a sufficient, but *not* necessary, condition for a kernel to obey the non-orthogonal characterization. This means that there will exist other ways for a kernel to satisfy the non-orthogonal principle. The investigation of these alternative venues will be a piece of future work for future graduate students in computer science and mathematics.

Non-orthogonal Corollary for the Gaussian Kernel No two vectors in the input space may, after being mapped into the feature space of the Gaussian kernel, become orthogonal to each other in the Gaussian kernel feature space.

Before we delve into a proof for the above theorem, one definition and two remarks are in order.

Definition of l^2 space If $x = (x_1, x_2, \ldots, x_\mathrm{n}, \ldots)$ is a vector with a countably infinite number of components such that the sum

$$\sum_{i=1}^{\infty} x_i^2 < \infty$$

and y is another vector of the same type as x, let the inner product between x and y be defined as

$$\langle x, y \rangle = \sum_{i=1}^{\infty} x_i y_i$$

It is easy to show that the above inner product also converges. The resulting inner-product space is known as l^2. (Shawe-Taylor and Cristianini, 2004)

Remark The feature space of the Gaussian kernel is an l^2 space.

Remark Naturally one would want to re-cycle the strategy employed in the proof of the non-collapsible theorem. That is, assuming that

$$\langle \phi(x), \phi(z) \rangle = 0$$

when

$$\langle x, z \rangle \neq 0$$

and show that this entails that

$$\langle (1, x_1, \ldots, x_n, \ldots), (1, z_1, \ldots, z_n, \ldots) \rangle = 0$$
$$1 + x_1 z_1 + \ldots + x_n z_n + \ldots = 0$$

which implies that

$$x_1 z_1 + \ldots + x_n z_n = 0$$

contradicting with the fact that *x* and *z* are non-orthogonal.

However, for this theorem there is an added complication from the fact that the individual components in the ϕ-mapped vector can now no longer be kept independent of each other as in the proof for the non-collapsible theorem. This is because the inner-product demands that the components be *added* to each other. Since the feature space of the Gaussian kernel are infinite dimensional, just because

$$x_1z_1 + \ldots + x_nz_n \neq 0 \qquad (13)$$

does not imply that

$$1 + x_1z_1 + \ldots + x_nz_n + \ldots \neq 0$$

since other terms in the above converging infinite sum might cancel out with the non-zero value of equation (13).

However, we should not be so preoccupied with the feature space mappings, because it worked so well for the non-collapsible theorem case, that we lose the overall vision of the kernel itself. The insight is that the Gaussian kernel itself *is* the inner-product of two vectors mapped into the Gaussian feature space, and we know that *that* kernel cannot yield any zero values.

Non-orthogonal Theorem (restated) No two vectors in the input space may, after being mapped into the feature space of the Gaussian kernel, become orthogonal to each other in the Gaussian kernel feature space.

Proof According to Shawe-Taylor and Cristianini, the form of the Gaussian kernel is $\exp\left(-\frac{\|\vec{x}-\vec{z}\|^2}{2\sigma^2}\right)$ (Shawe-Taylor and Cristianini, 2004). Since the exponential function does not yield any 0 or negative values, the inner product of any two input space vectors ϕ-mapped into the Gaussian feature space cannot be 0 and hence the ϕ-mapped feature space projection vectors of any two input-space vectors cannot be orthogonal to each other. □

Remark An examination of equation (11) also shows that large values for the inner-products between landmark vectors and training vectors can help the β_i's to better approximate the α_i's.

The non-collapsible theorem establishes a guarantee that a set of *n* landmark vectors, if they are capable of spanning an *n*-dimensional space, will still be capable of spanning an *n*-dimensional subspace in the Gaussian feature space after mapping via the ϕ-map. This theorem prevents *n* input space linearly independent landmark vectors from being mapped into the same vector in the feature space. This is necessary in order for the landmark vectors to span an *n*-dimensional subspace required to span the weight vector.

Despite the seemingly infinite dimension of the Gaussian feature space, a collection of *n* labeled training vectors, even after being mapped into this infinite dimensional feature space through the ϕ-map, can still span at most an *n*-dimensional subspace in the feature space. Since the weight vector is expressed as a *linear combination* of the ϕ-mapped labeled training vectors, we have that the decision hyperplane itself is an *n*-1 dimensional object contained within the *n*-dimensional subspace spanned by the set of *n* ϕ-mapped labeled training vectors.

Now we have two facts:

1. A set of *n* random vectors that can span an *n*-dimensional space in the input space can likewise span an *n*-dimensional subspace in the feature space after ϕ-map.
2. The decision hyperplane itself is contained in an *n*-dimensional subspace spanned by the *n* labeled training vectors in the feature space.

It remains to find out how well the *n*-dimensional subspace in point 1 could characterize the *n*-dimensional subspace in point 2. In other words how well could the *n* random vectors span the weight vector that is formed from the linear combination of training vectors?

To this question the non-orthogonal theorem provides a satisfactory lower-bound answer. As shown in the proof of the non-orthogonal theorem for the Gaussian kernel, in a Gaussian kernel feature space it is impossible for orthogonal vectors to exist. Thus we have that for any pair of input-space vectors x and z

$$\langle \phi(x), \phi(z) \rangle \neq 0$$

where ϕ is the mapping function underlying the Gaussian kernel. The non-orthogonal theorem guarantees that it is impossible for any ϕ-mapped landmark vector to exist entirely outside of the subspace of the decision hyperplane. Thus the non-orthogonal theorem prevents the occurrence of landmark vectors whose ϕ-mappings are totally orthogonal to the decision hyperplane and cannot be used in the formation of the final landmark vector linear combination of the weight vector of the decision hyperplane. Furthermore, the non-orthogonality experiments in Chapter 4 will demonstrate that the ϕ-mapped landmark vectors are actually capable of providing good characterizations of the subspace spanned by the ϕ-mapped training vectors.

The linear independence of random vectors experiments performed in Chapter 4 demonstrates that, with high probability, the generated random vectors in an input vector space will form a linearly independent set. We could also deliberately construct the set of *orthonormal basis* vectors for the input vector space to ensure the linear independence of the landmark vectors. Together, these two types of vectors allow us to apply the non-collapsible theorem, the non-orthogonal theorem and previous discussions to provide an answer to the key question posed at the start of this section in the affirmative.

Answer to Key Question Yes, If the weight vector of a decision hyperplane in the feature space induced by a kernel can be expressed as the linear combination of the set of ϕ-mapped feature space vectors that are the images of a set of labeled training vectors in the input space, then is it possible, with high probability, that the same decision hyperplane can be equivalently expressed as the linear combination of a set ϕ-mapped feature space vectors that are of the following types:

1. Gaussian normal random vectors
2. Input space orthonormal basis vectors

This is because the feature space *does not distinguish between* feature vectors that are of the random vector type, input-space orthonormal basis vector type, unlabeled data type or labeled training data type. Once mapped into the feature space *all vectors are equal in significance*. The different types of input space vectors lose their own unique input-space properties, and in the case of the labeled training vectors, their labels, as soon as they are mapped into the feature space. To the Gaussian kernel, all feature vectors are infinite-dimensional l^2 space vectors without any labels, distributions or any other properties attached. As long as the underlying kernel-induced feature space and the set of ϕ-mapped landmark vectors satisfy the linear algebraic properties outlined in the non-collapsible theorem and the non-orthogonal theorem, then the power landmark vector learning framework is capable of generating, through the particular linear learning algorithm employed in the feature space, the specific linear combination of the landmark vectors, in the form of the α sequence, that are needed to give rise to a weight vector that closely approximates the original weight vector learned using labeled training data only.

3.2 A retrospective on Section 4.6 of (Blum, 2006)

In Section 4.6 of (Blum, 2006) Blum gave a kernel for which the power landmark vector learning framework developed in this thesis would fail to work. Paraphrasing Blum's pathological kernel, the definition of the ϕ-map for the pathological kernel is as follows:

$$\phi(x) = \begin{cases} \text{The Gaussian feature space } \phi\text{-map if } x \text{ belongs to the distribution} \\ \text{of the learning problem} \\ \\ (1,\ 1,\ 1) \text{ if } x \text{ does not belong to the distribution of the learning} \\ \text{problem} \end{cases}$$

Blum employed a complicated and abstract argument involving half-spaces to demonstrate the necessity of having the distribution of the randomly chosen unlabeled vectors belonging to the same distribution and the same input space as the training vectors in the input space. His conclusion also demonstrated why the power landmark learning framework will fail for the above feature space. Using the theories developed for the power landmark learning framework, we can now more easily see why power landmark learning will fail for the above pathological feature space and kernel. The Gaussian normal random vectors, the input-space orthonormal bases vectors, and most types of landmark vectors other than the unlabeled vectors belonging to the same distribution and the same input space as the training vectors in the input space would most likely not belong to the distribution of the learning problem. This pathological kernel, when applied under the power landmark vector learning framework, will *violate* the non-collapsible theorem by collapsing all landmark vectors into a single (1, 1, 1) vector. Hence, the ϕ-mapped landmark vectors in this pathological feature space would only be able to span a vector in the direction of the (1, 1, 1) vector. This vector would deviate from most weight vectors spanned by training examples under the Gaussian kernel. The deviations would become so great that the power landmark vector learning framework is made ineffective.

This example demonstrates the fact that the utility of the power landmark vector learning framework is not universal. The validity of the power landmark vector learning framework must be justified on a per-kernel basis. This example also demonstrates that the non-collapsible theorem and the non-orthogonal theorem may be used for identifying the kernels that will work under the power landmark vector learning framework.

Stated simply, the two theorems hold for the Gaussian kernel because the feature vector space of the Gaussian kernel *subsumes* the input vector space. These properties of the Gaussian kernel allow it to meet the non-collapsible and non-orthogonal principles, hence making it a good kernel for use by the power landmark vector learning framework. Such properties, however, do not apply to all kernels and all input spaces, the pathological kernel discussed in this section being the best example. Thus, the power landmark vector learning framework must be justified on a per-kernel and per-input space basis.

3.3 Benefits of the power landmark vector learning framework

The power landmark vector learning framework naturally inherits the benefits of Blum's landmark vector learning framework. The power landmark vector learning framework offers an explicit and easy way to access the feature space. It also preprocesses the non-linear input space based training vectors in a manner similar to Blum's framework and produces a feature space in which traditional linear learning algorithms such as the Perceptron and the SVM can be applied without any modification. More importantly, the power landmark vector learning framework improves upon Blum's landmark vector learning framework by allowing additional classes of landmark vectors from the input space to be utilized for the learning process. As experiments in Chapter 4 will show, the power landmark vector learning framework has potential applications in dimensionality reduction and data visualization. Finally, the power landmark vector learning framework is an alternative framework of learning using kernel theory, thus by learning and understanding the principles of power landmark learning framework one could obtain a better grasp and a clearer conceptual understanding of the various theoretical underpinnings of kernel methods in general.

3.4 Relevance to bioinformatics

The power landmark learning framework provides an alternative to traditional kernel learning frameworks by making linear feature spaces explicit. Since many modern bioinformatics learning problems involve the use of kernels, the power landmark learning framework could be applied to these bioinformatics learning problems. Due to the explicit feature space involved in power landmark learning, the feature vectors corresponding to the training vectors in the input space could be explicitly examined and used for other purposes. Potentially, a fewer number of landmark vectors are needed than the total number of all input training vectors. This observation implies that power landmark learning may be used to reduce the runtime of bioinformatics learning problems from $O(n^2)$ to $O(nl)$, where l is the number of landmark vectors. Power landmark learning also has potential applications to dimensionality reduction and data visualization tasks in bioinformatics machine learning. All of these serve to make power landmark learning highly relevant and suitable for non-linear machine learning problems that arise in bioinformatics. In fact, the entire Chapter 4 involves the application of the power landmark vector learning framework to the discovery of non-linear relationships existing within each of the three bioinformatics machine-learning datasets.

3.5 The linear kernel

The linear kernel is simply the Euclidean inner product (dot product) between two vectors in $\mathbb{R}^n$. Let $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$ be two vectors in $\mathbb{R}^n$, with $x_1, \ldots, x_n \in \mathbb{R}$ and $y_1, \ldots, y_n \in \mathbb{R}$, then the linear kernel $K(x, y)$ is defined as:

$$K(x,y) = \langle x,y \rangle = \langle (x_1,x_2,\cdots,x_n),(y_1,y_2,\cdots,y_n) \rangle = x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

The linear kernel is the most basic and simple kernel. In fact, other kernels could be seen as generalizations of the basic linear kernel into higher-dimensional Euclidean space or the infinite-dimensional l^2 space. Historically the field of kernel methods also traces its origins to a generalization of the concept of the regular dot-product in Euclidean space.

Before the question is addressed in the present context, one more study done by (Vempala, 2004) needs to be mentioned. It is the lemma 1.3 shown on page 2 of (Vempala, 2004).

Lemma (Santosh) Let each entry of an $n \times d$ matrix R be chosen independently from $N(0, 1)$. Let $v = \frac{1}{\sqrt{k}} R^{\mathrm{T}} u$ for $u \in \mathbb{R}^{\mathrm{n}}$. Then for any $\varepsilon > 0$,

1. $\mathbb{E}(||v||^2) = ||u||^2$.
2. $\mathrm{P}(|||v||^2 - ||u||^2| \geq \varepsilon ||u||^2) < 2e^{-(\varepsilon^2 - \varepsilon^3)(k/4)}$

where $||v||$ denotes the Euclidean norm of a vector v and $\mathrm{P}(x)$ denotes the probability of event x.

Note that point 2 is essentially saying that with high probability, the norm of u will be approximately preserved to a high degree through the selection of a suitable pair of (ε, k) values when it has been projected into a lower dimensional space in the form of vector v.

Since the pair-wise distances between pairs of data vectors in a high dimensional space is the norm of the vector formed by joining the starting data vector to the ending data vector, that norm will also be preserved when the data vectors themselves are projected into a lower dimensional space. This is the theoretical basis underlying much of (Blum, 2006)'s work and is also the theoretical basis for some of the arguments in this section.

My theorem: If P is linearly separable in the ϕ-space induced by a linear kernel K, let $x^{(1)}, \ldots, x^{(d)}$ be d uniform random vectors whose components are generated in an independent and identically distributed manner, let $F(x) = \frac{1}{\sqrt{d}}(K(x, x^{(1)}), \ldots, K(x, x^{(\mathrm{d})}))^{\mathrm{T}}$, then $F(P)$ is linearly separable over $\mathbb{R}^{\mathrm{d}}$.

Proof: Take the random vectors $\{x^{(i)}\}$ and form the following matrix:

$$R = \begin{pmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(d)} \\ x_2^{(1)} & x_2^{(2)} & \cdots & x_2^{(d)} \\ \vdots & \vdots & \ddots & \vdots \\ x_n^{(1)} & x_n^{(2)} & \cdots & x_n^{(d)} \end{pmatrix}$$

where $x_j^{(i)}$ denotes the j^{th} component of vector i.

For any vector u in the input space $\mathbb{R}^{\mathrm{n}}$,

$$F(u) = \frac{1}{\sqrt{d}}(K(u, x^{(1)}) \quad K(u, x^{(2)}) \quad \dots \quad K(u, x^{(d)}))^{\mathrm{T}}$$

$$= \frac{1}{\sqrt{d}}(x^{(1)\mathrm{T}}u \quad x^{(2)\mathrm{T}}u \quad \dots \quad x^{(\mathrm{d})\mathrm{T}}u)$$

$$= \frac{1}{\sqrt{d}}(x^{(1)\mathrm{T}} \quad x^{(2)\mathrm{T}} \quad \dots \quad x^{(\mathrm{d})\mathrm{T}})\, u$$

$$= \frac{1}{\sqrt{d}}\begin{pmatrix} x_1^{(1)} & \cdots & x_1^{(d)} \\ \vdots & \ddots & \vdots \\ x_n^{(1)} & \cdots & x_n^{(d)} \end{pmatrix}^T \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix}$$

$$= \frac{1}{\sqrt{d}} R^T u$$

This is essentially the same form of Johnson-Lindenstrauss style random projection as seen in lemma (Santosh). If the ability of Johnson-Lindenstrauss style random projection to approximately preserve linear margins can be ascertained (see Section 5.2.4), then we can conclude that *P* remains linearly separable in *F*(*P*) in the sense of lemma (Santosh). □

3.6 The positive semi-definite kernel

The positive semi-definite (PSD) kernel is defined in proposition 3.22 of (Shawe-Taylor and Cristianini, 2004). If *A* is an *n* × *n* symmetric and positive semi-definite matrix, and *x* and *z* are *n*-dimensional vectors, then the PSD kernel is defined as: $K(x, z) = x^{\mathbf{T}}Az$. The reason the PSD kernel can be considered a kernel is because the matrix *A*, due to its symmetry and positive semi-definiteness, can be expressed as the product V^TDV. The matrix *D* is a diagonal matrix containing the non-negative eigenvalues of the matrix A and V is the orthogonal matrix containing eigenvectors corresponding to the eigenvalues in D. Let $\sqrt{D}$ be the matrix that results from taking the square root of each entry in D. Note that $\sqrt{D}\,\sqrt{D} = D$. Thus, $A = V^T \sqrt{D}\,\sqrt{D}\, V$ and

$$
\begin{aligned}
K(x, z) &= x^T A z \\
&= x^T V^T D V z \\
&= x^T V^T \sqrt{D}^{\mathbf{T}} \sqrt{D}\, V z \\
&= (\sqrt{D}\, V x)^T \sqrt{D}\, V z \\
&= \langle \sqrt{D}\, V x,\ \sqrt{D}\, V z \rangle
\end{aligned}
$$

which is a Euclidean inner product for the vectors $\sqrt{D}\, Vx$ and $\sqrt{D}\, Vz$. These two products are essentially projections of the vectors x and z into a new space spanned by the vectors of $(V\sqrt{D}\,)^{\mathrm{T}}$. Since matrices defined on the pair-wise Euclidean inner products are always symmetric and positive semi-definite, the PSD kernel is likewise symmetric and positive semi-definite.

In a sense, the PSD kernel can be viewed as an extension of the linear kernel. The proof of its validity under the uniform random landmark vectors learning framework will likewise follow the proof for the linear kernel case.

My Theorem: Let $x^{(1)}, \ldots, x^{(d)}$ be d uniform random vectors whose components are generated independently and identically from the $N(0, 1)$ distribution. Let

$F(x) = \frac{1}{\sqrt{d}}(K(x, x^{(1)}), \ldots, K(x, x^{(\mathrm{d})}))^{\mathrm{T}}$, where K is a PSD kernel. If P is linearly separable in

the ϕ-space induced by K, then $F(P)$ is linearly separable over $\mathbb{R}^{\mathrm{d}}$.

Proof: Let $\{x^{(1)}, \ldots, x^{(d)}\}$ be a set of d n-dimensional random vectors whose components are generated independently and identically from the $N(0, 1)$ distribution.

Also, let A be any symmetric and positive semi-definite matrix:

$$A = \begin{pmatrix} a_1^{(1)} & a_2^{(1)} & \cdots & a_n^{(1)} \\ a_2^{(1)} & a_2^{(2)} & \cdots & a_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ a_n^{(1)} & a_n^{(2)} & \cdots & a_n^{(d)} \end{pmatrix}$$

For any vector u in the input space $\mathbb{R}^{\mathrm{n}}$,

$$F(u) = \frac{1}{\sqrt{d}}(K(u, x^{(1)}) \quad K(u, x^{(2)}) \quad \ldots \quad K(u, x^{(d)}))^{\mathrm{T}}$$

$$= \frac{1}{\sqrt{d}}(x^{(1)\mathrm{T}}Au \quad x^{(2)\mathrm{T}}Au \quad \ldots \quad x^{(d)\mathrm{T}}Au)^{\mathrm{T}}$$

$$= \frac{1}{\sqrt{d}}((\sum_{i=1}^{n} a_i^{(1)} x_i^{(1)} \quad \cdots \quad \sum_{i=1}^{n} a_n^{(i)} x_i^{(1)}) \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}$$

$$(\sum_{i=1}^{n} a_i^{(1)} x_i^{(2)} \quad \cdots \quad \sum_{i=1}^{n} a_n^{(i)} x_i^{(2)}) \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}$$

$$\cdots$$

$$(\sum_{i=1}^{n} a_i^{(1)} x_i^{(d)} \quad \cdots \quad \sum_{i=1}^{n} a_n^{(i)} x_i^{(d)}) \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix})^{\mathrm{T}} \tag{14}$$

From statistics (Feller, 1968) we know that a linear combination of *N*(0, 1) random variables is itself another *N*(0, 1) random variable. Hence we could treat the symmetric and positive semi-definite matrix *A* as supplying a matrix of weights to the random landmark vectors to form the linear combination that will give rise to a new *N*(0, 1) random variable. By renaming these new random variables using the following scheme:

$$R_i^{(j)} = \sum_{k=1}^{j} a_j^{(k)} x_k^{(i)} + \sum_{k=j+1}^{n} a_k^{(j)} x_k^{(i)}$$

we could simplify expression (14) into:

$$(14) = \frac{1}{\sqrt{d}}((R_1{}^{(1)} \quad \dots \quad R_1{}^{(n)})\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} \quad (R_2{}^{(1)} \quad \dots \quad R_2{}^{(n)})\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} \quad \dots \quad (R_d{}^{(1)} \quad \dots \quad R_d{}^{(n)})\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix})^{\mathrm{T}}$$

$$= \frac{1}{\sqrt{d}}\begin{pmatrix} R_1^{(1)} & R_1^{(2)} & \cdots & R_1^{(n)} \\ R_2^{(1)} & R_2^{(2)} & \cdots & R_2^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ R_d^{(1)} & R_d^{(2)} & \cdots & R_d^{(n)} \end{pmatrix}\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}$$

$$= \frac{1}{\sqrt{d}} R^{\mathrm{T}} u$$

where

$$R^T = \begin{pmatrix} R_1^{(1)} & R_1^{(2)} & \cdots & R_1^{(n)} \\ R_2^{(1)} & R_2^{(2)} & \cdots & R_2^{(n)} \\ \vdots & \vdots & \ddots & \vdots \\ R_d^{(1)} & R_d^{(2)} & \cdots & R_d^{(n)} \end{pmatrix}$$

This is essentially the same form of Johnson-Lindenstrauss style random projection as seen in lemma (Santosh). If the ability of Johnson-Lindenstrauss style random projection to approximately preserve linear margins can be ascertained (see Section 5.2.4), then we can conclude that *P* remains linearly separable in *F*(*P*) in the sense of lemma (Santosh). □

3.7 Experimental procedures

The algorithms that allow the power landmark vector learning framework to be applied to the learning experiments done in Chapter 4 are presented here.

Algorithm 1

Given:

- X an $l \times n$ matrix representing the input training set to the power landmark vector learning framework, consisting of l n-dimensional training vectors.
- Y an l dimensional vector representing the labels for each of the l n-dimensional training vectors.
- Z a $d \times n$ matrix representing the set of d n-dimensional landmark vectors that are either Gaussian random or input-space orthogonal.
- K the Gaussian kernel

Find:

- r prediction accuracy under the power landmark vector learning framework

Procedure:

1. For each vector $x \in X$, construct the FP feature space vector $F(x) = (K(x, z^{(1)}), \ldots, K(x, z^{(d)}))^{\mathrm{T}}$, where $z^{(1)}, \ldots, z^{(d)} \in Z$.
2. Compile all the FP feature space vectors constructed in step 1 into a new matrix $F(X)$.
3. Feed $F(X)$ and Y as input to the unmodified, primal form of a classical linear learning algorithm.
4. Obtain prediction accuracy result r after algorithm in step 3 completes execution.

5. Compare with prediction accuracies obtained under traditional kernel learning frameworks.

Algorithm 2

Given:

- *X* an $l \times n$ matrix representing the input training set to the power landmark vector learning framework, consisting of l n-dimensional training vectors.
- *Y* an l dimensional vector representing the labels for each of the l n-dimensional training vectors.
- *Z* a $3 \times n$ matrix representing the 3 n-dimensional landmark vectors that are either Gaussian random or randomized input-space orthogonal that will form the bases for the three dimensional Euclidean space used for data visualization
- *K* the Gaussian kernel

Find:

- Three dimensional plots in which the two classes of training vectors are represented by vectors in space of two different colors.

Procedure:

1. For each vector $x \in X$, construct the FP feature space vector $F(x) = (K(x, z^{(1)}), K(x, z^{(2)}), K(x, z^{(3)}))^{T}$, where $z^{(1)}, z^{(2)}, z^{(3)} \in Z$.

2. For each FP feature space vector constructed in step 1, plot it into a three dimensional Euclidean space, with one color used for feature vectors with label “-1” and another color used for feature vectors with label “+1”.
3. For the actual experiments that were performed in Chapter 4, vectors indicating healthy condition are colored green and vectors indicating ill condition are colored red.
4. Output the three dimensional plot. Assess quality of plot by examining how well vectors of the same color have clustered in the plot.

Algorithm 3

Given:

- X an $l \times n$ matrix representing the input training set to the power landmark vector learning framework, consisting of l n-dimensional training vectors.
- Y an l dimensional vector representing the labels for each of the l n-dimensional training vectors.
- Z a $d \times n$ matrix representing the set of d n-dimensional landmark vectors that are either Gaussian random or input-space orthogonal.
- K the Gaussian kernel

Find:

- v a 8-dimensional vector that maintain the eight tally counts for the number of kernel evaluations, x, that fall within each of the eight ranges:

$x > 1$, $1 > x > 10^{-1}$, $10^{-1} > x > 10^{-2}$, $10^{-2} > x > 10^{-3}$, $10^{-3} > x > 10^{-4}$,

$10^{-4} > x > 10^{-5}$, $10^{-5} > x > 10^{-6}$, $10^{-6} > x$.

Procedure:

1. Initialize v as the zero-vector.
2. For each vector $x \in X$, construct the FP feature space vector

   $F(x) = (K(x, z^{(1)}), \ldots, K(x, z^{(d)}))^{T}$, where $z^{(1)}, \ldots, z^{(d)} \in Z$.
3. Compile all the FP feature space vectors constructed in step 1 into a new matrix $F(X)$.
4. Examine the value of each entry, x, in the matrix $F(X)$ to determine which of the eight ranges $x > 1$, $1 > x > 10^{-1}$, $10^{-1} > x > 10^{-2}$, $10^{-2} > x > 10^{-3}$, $10^{-3} > x > 10^{-4}$, $10^{-4} > x > 10^{-5}$, $10^{-5} > x > 10^{-6}$, $10^{-6} > x$ it falls within.
5. Increment the appropriate entry in vector v as per result of step 4 for each entry in the matrix $F(X)$
6. Repeat steps 4 and 5 for all entries in the matrix $F(X)$
7. Output vector v, plot into bar graph if desired.

Algorithm 4

Given:

- Z a $d \times n$ matrix representing the set of d n-dimensional Gaussian-random landmark vectors.

Find:

- b the boolean that answers whether or not no vector in Z maybe expressed as a linear combination of other vectors in Z

Procedure:

1. Perform Gauss-Jordan elimination on Z.
2. Note number of non-zero rows in Z after step 1 completes. Assign this number to r.
3. If $r = d$ or $r = n$, then $b \leftarrow$ true, else $b \leftarrow$ false.
4. Plot r versus d as a bar graph.

3.8 Chapter summary

This chapter presents the Power Landmark Vector Learning Framework as the major theoretical contribution of this thesis. The chapter started with several observations of some of the powerful ideas behind Blum's work which was not explicitly stated by Blum. It then proposes a key question concerning the possibility of extending Blum's framework to two new classes of vectors and proceeds to answer it. An entire framework for a style of kernel learning based on two new classes of vectors was formulated in the process of answering the key question. This thesis's work culminates in the introduction of the *non-collapsible* and *non-orthogonal* theorems and the two proofs proposed to show that these theorems work for the feature space of the Gaussian kernel. Together, the two theorems also provide a way of quickly identifying kernels that are not viable under the Power Landmark Vector Learning Framework. Finally the answer to the key question is given in the affirmative. Some follow-up discussions on the benefits of the Power Landmark Vector Learning Framework, its relevance to bioinformatics, and a retrospect on the pathological kernel discussed in Blum's paper were given. Two more proofs were given which show the suitability of applying the linear kernel and the PSD kernel under the Power Landmark Vector Learning Framework using the technique of Johnson-Lindenstrauss random projection. Finally, algorithmic procedures were given for all the major tasks that are to be accomplished in Chapter 4.

Chapter 4
Experiments, Results and Discussions

4.1 Experimental setup

4.1.1 Gaussian-random landmark vector mapping

This experiment tests the ability of Gaussian random landmark vectors to generate FP spaces that could adequately capture or approximate the feature space in which the original decision hyperplane resides. The results of this experiment will be in the form of a graph. The graph is a line graph documenting the experimentally observed relationship between the accuracy of learning and the number of input-space Gaussian Random landmark vectors used to generate the FP space. This experiment uses algorithm 1 from Section 3.7.

4.1.2 Input-space orthonormal landmark vector mapping

This experiment tests the ability of input-space orthonormal landmark vectors to generate FP spaces that could adequately capture or approximate the feature space in which the original decision hyperplane resides. The results of this experiment will be in the form of a graph. The graph is a line graph documenting the experimentally observed relationship between the accuracy of learning and the number of input-space orthonormal landmark vectors used to generate the FP space. This experiment uses algorithm 1 from Section 3.7.

4.1.3 Data visualization

This is where we push the power landmark vector learning framework to the limit and see how it performs with just three landmark vectors. Two graphs will be shown in this section. The first graph shows how well vectors belonging to a same class cluster with each other in the three dimensional space generated by applying the power landmark vector on the training set with three Gaussian random landmark vectors. The second graph shows how well vectors belonging to a same class cluster with each other in the three dimensional space generated by applying the power landmark vector on the training set with three *randomized* input-space orthonormal bases landmark vectors. This experiment uses algorithm 2 from Section 3.7.

4.1.4 Non-orthogonality experiment

This experiments test whether the Gaussian kernel is "well-behaved" in terms of the non-orthogonal principle. In other words, this experiment will record the values of all Gaussian kernel valuations from a trial and sort the values according to their magnitude into eight logarithmic categories. The results of this experiment will be in the form of a graph. The graph is a bar graph. The graph depicts the number of kernel evaluations, x, that fall into each of the eight logarithmic ranges. The ranges are: $x > 1$, $1 > x > 10^{-1}$, $10^{-1} > x > 10^{-2}$, $10^{-2} > x > 10^{-3}$, $10^{-3} > x > 10^{-4}$, $10^{-4} > x > 10^{-5}$, $10^{-5} > x > 10^{-6}$, $10^{-6} > x$. This experiment uses algorithm 3 from Section 3.7.

4.1.5 Linear independence of random vectors

A quick way of showing the mutual linear independence of a group of random vectors is to put them into a matrix and perform Gaussian-Jordon elimination on the matrix and demonstrate that the rank of the resulting matrix is equal to the number of random vectors in the group. This section will perform such an experiment for each type of input vector space associated with each learning problem by formatting them into a matrix and apply Matlab's `rank()` function to calculate the rank. It will contain one double-bar graph. The *x* axis of that bar graph will indicate the number of Gaussian random vectors used to form the matrix from a moderately small number to a number that moderately exceeds the dimensionality of the input space. For each set of random vectors generated, a pair of bars will appear above the corresponding *x*-position, the first bar, which is blue, will be the number of mutually linearly independent random vectors, the second bar, which is yellow, will be the number of random vectors that are linearly dependent on the other vectors. This experiment uses algorithm 4 from Section 3.7.

4.2 Protein subcellular localization dataset

The dataset used in this section of the experiments is supplied at the courtesy of Cory Spender from Simon Fraser University, Canada. The dataset was used in a study done in (Gardy et al. 2003). The dataset concerns the prediction of subcellular localization sites for proteins found in Gram-negative bacteria. The dataset consists of 391 input vectors. Each input vector has 20 features. Therefore the input space is $\mathbb{R}^{20}$. There are 3 different labels representing the three different localization sites for subcellular proteins. Hence the training vectors could fall into three classes. Five-fold cross validation was done in Sections 4.2.1and 4.2.2 to derive the final classification accuracy results.

Canonical kernel learning using traditional kernel learning framework is performed at the beginning of this section of experiments by libSVM. The results of the learning are shown in Figure 8. The canonical cross-validated prediction accuracy for this section is established by libSVM to be 83.12%

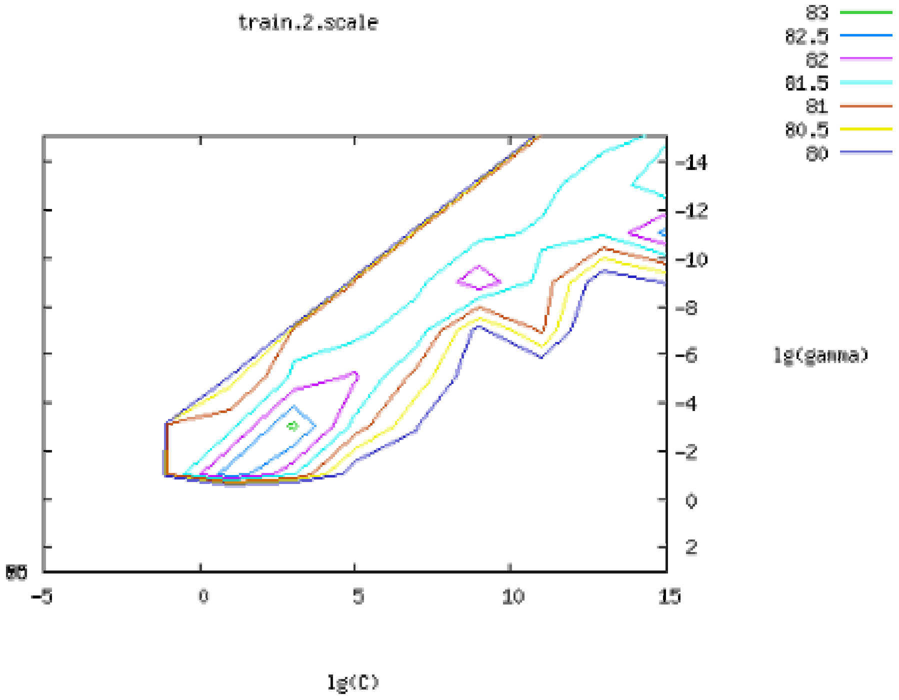

Figure 8: results of canonical kernel learning of Section 4.2 dataset by libSVM.

4.2.1 Gaussian-random landmark vector mapping

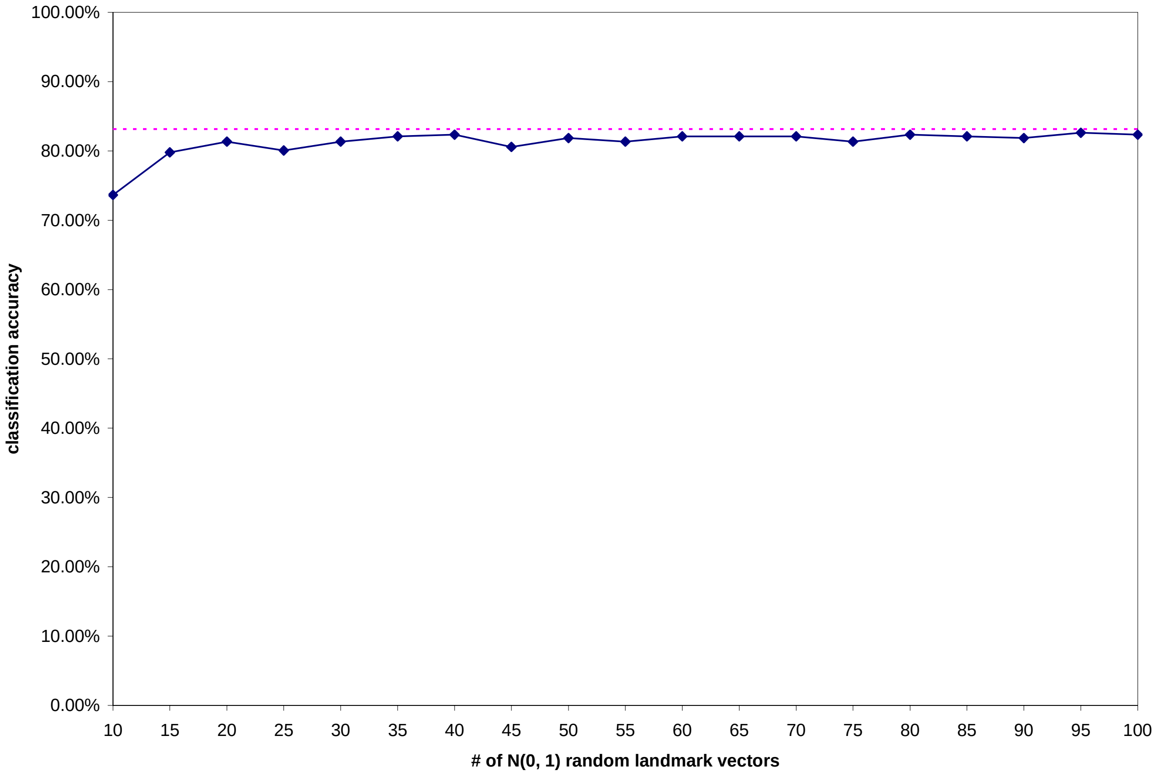

Figure 9: Results of the Gaussian random landmark vector projection experiments for Section 4.2. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark For sufficient number of random vectors the performance of the Power Landmark Framework rivals that of tradition kernel learning framework. This demonstrates the validity of the Power Landmark Vector Learning Framework. Also note that far less than 391 random vectors, which is the total size of the training set, were needed in order to achieve performance comparable with traditional kernel learning framework. This demonstrates the potential for dimensionality reduction that exists within the Power Landmark Vector Learning Framework.

4.2.2 Input-space orthonormal landmark vector mapping

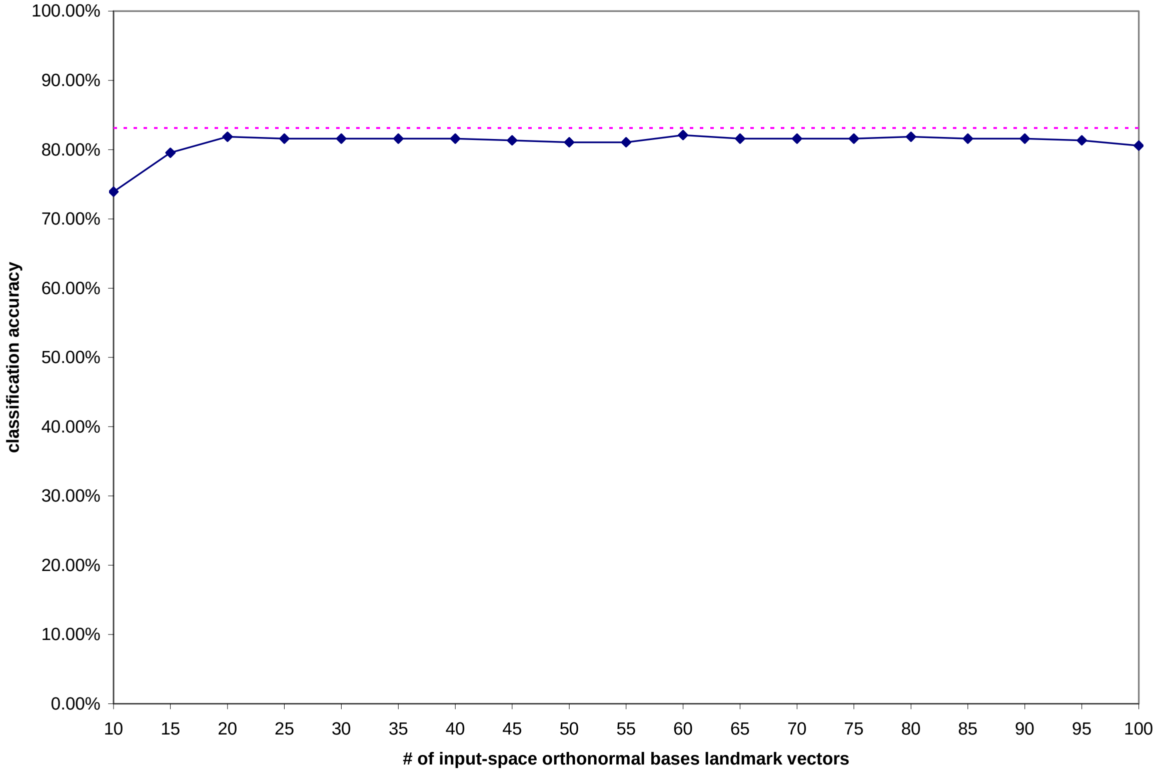

Figure 10: Results of the input-space orthonormal bases landmark vector projection experiments for Section 4.2. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark These results confirm the ability of input-space orthonormal bases landmark vectors to approximate the decision hyperplane, albeit with a slightly less degree of accuracy compared to the Gaussian normal random landmark vectors if one were to examine the numerical data.

4.2.3 Data visualization

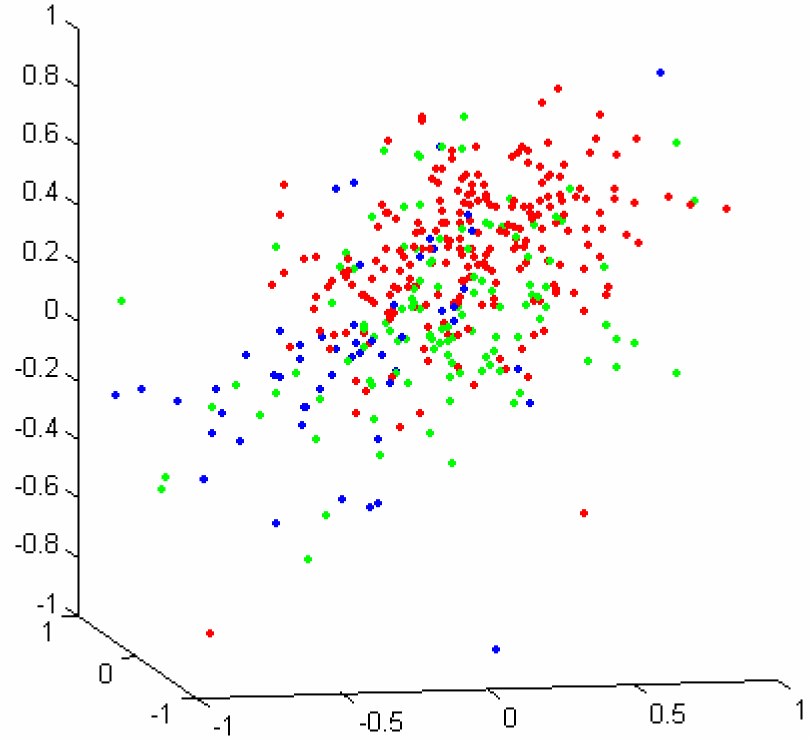

Figure 11: Results of data visualization using Gaussian normal landmark random vectors for Section 4.2.

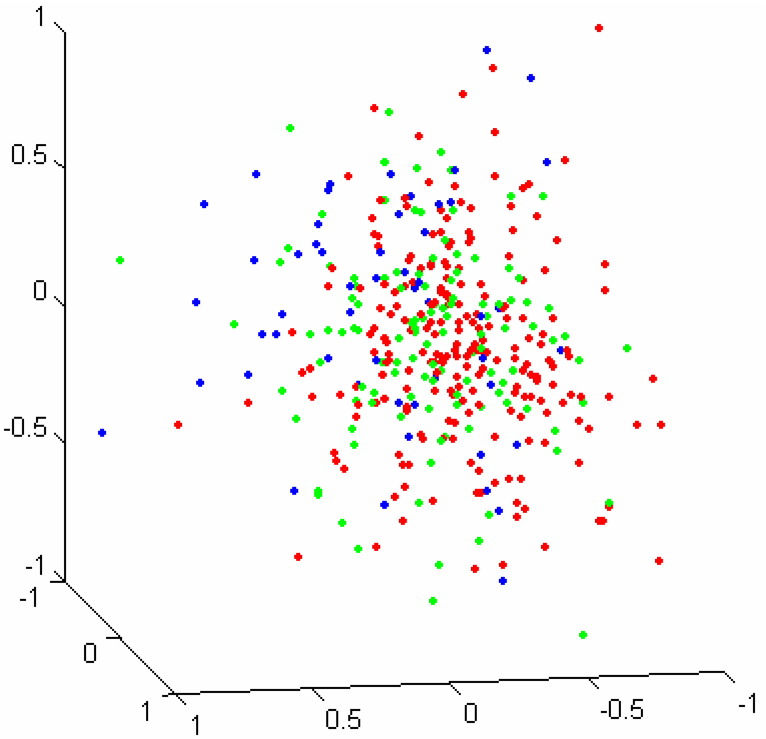

Figure 12: Results of data visualization using randomized input-space orthonormal bases landmark vectors for Section 4.2.

Remark: The green, red and blue plot vectors represent the three different localization sites of subcellular proteins. The two data visualizations all indicate that the training vectors for these three different classes have clustered into three layers in the FP space. The green layer is at the very front relative to the reader. The red layer is in the middle. The blue layer is at the back.

4.2.4 Non-orthogonality test

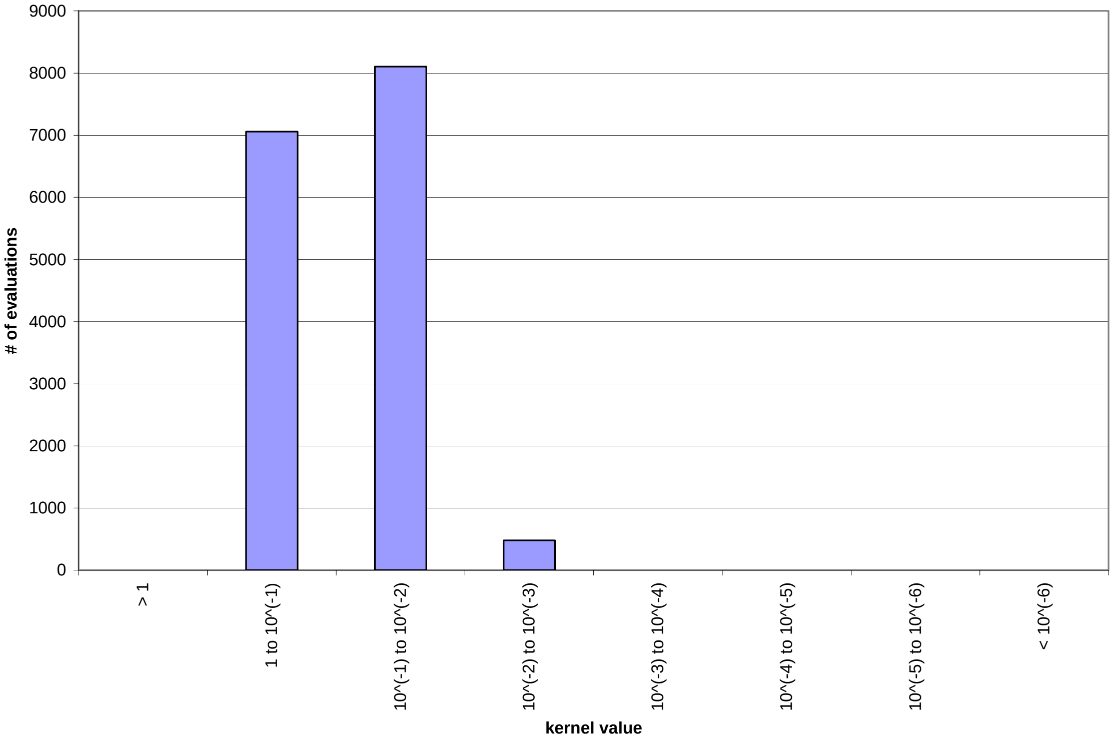

Figure 13: Results of the non-orthogonality test experiment for Section 4.2.

Remark This experiment demonstrates that most inner-products under the Gaussian kernel tend to be “well-behaved” in that their values lie well away from small values toward the right-side of the *x*-axis in Figure 13. Since small kernel values indicate near-orthogonality, these “well-behaved” values mean that Gaussian random vectors tend to be rather related to most training vectors.

4.2.5 Linear independence of random vectors

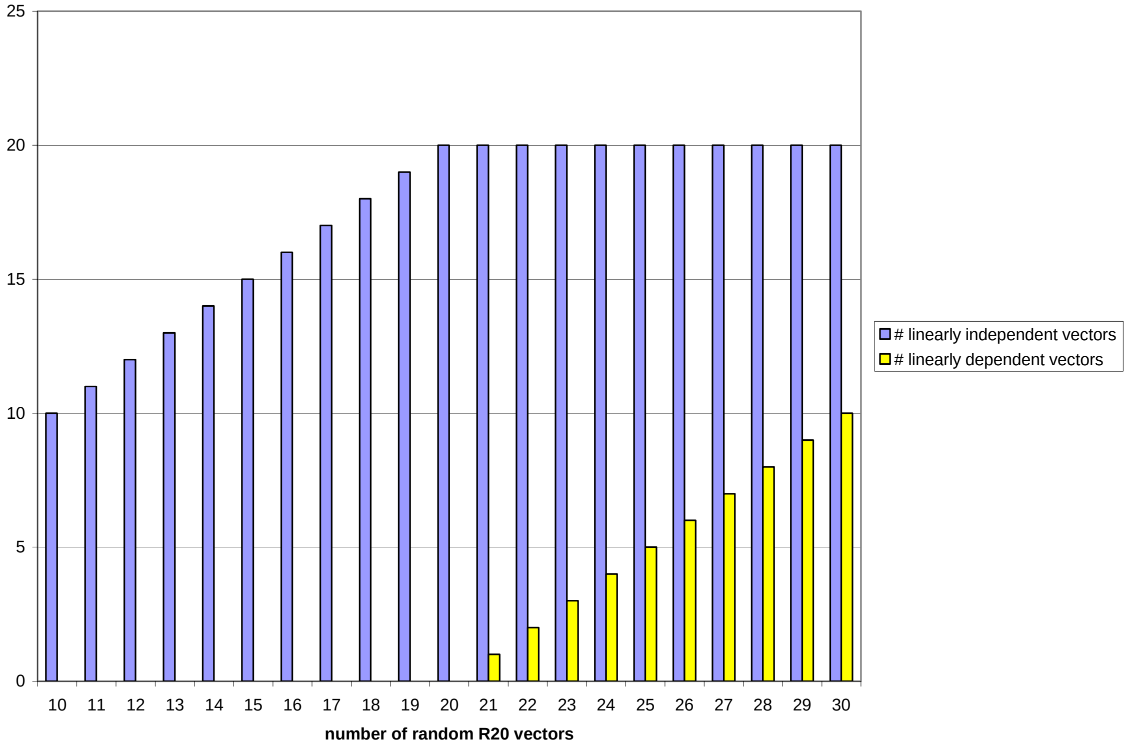

Figure 14: Results of the linear independence of random vectors experiment for the input space of Section 4.2.

Remark As expected, the generated $\mathbb{R}^{20}$ random vectors form linearly independent sets. The presence of linearly dependent random vectors in the later trials is due to the number of random vectors generated exceeding the dimensionality of the input space of the training vectors.

4.3 Pima aboriginals diabetes dataset

The Pima aboriginals diabetes dataset is provided at the courtesy of National Institute of Diabetes and Digestive and Kidney Diseases and Vincent Sigillito of the Applied Physics Laboratory of the Johns Hopkins University who was the original donor of the dataset. The actual data itself is obtained by the author of this thesis from the FTP site of the University of California at Irvine.

This data had been used in the past by (Smith et al., 1988) to investigate possible vital signs that may be used to indicate the presence of diabetes within patients according to World Health Organization (WHO) standards. The aboriginal patients who volunteered for the study conducted to gather the original data live near Phoenix, Arizona, US. Smith et al. were able to achieve a classification accuracy of 76% on the dataset. This accuracy closely mirrored the accuracies achieved under the power landmarks vector learning framework in this thesis.

There are a total of 768 training instances included in this dataset. Out of these training instances, 759 are actually used. The 7 remaining training instances had missing attributes after they have been processed by libSVM and hence had to be discarded. Each training instance has 8 features and class variable that provides the label for that training instance. The features have to do with the various measurable vital signs pertinent to the diabetic condition such as plasma glucose concentration, diastolic blood pressure and body mass index (BMI) that are measured for each patient. The class variable takes on the binary value of 0 or 1 with 0 indicating a healthy person and 1 indicating a diabetic patient.

Canonical kernel learning using traditional kernel learning framework is performed at the beginning of this section of experiments by libSVM. The results of the learning are shown in Figure 15. The canonical cross-validated prediction accuracy for this section is established by libSVM to be 77.474%

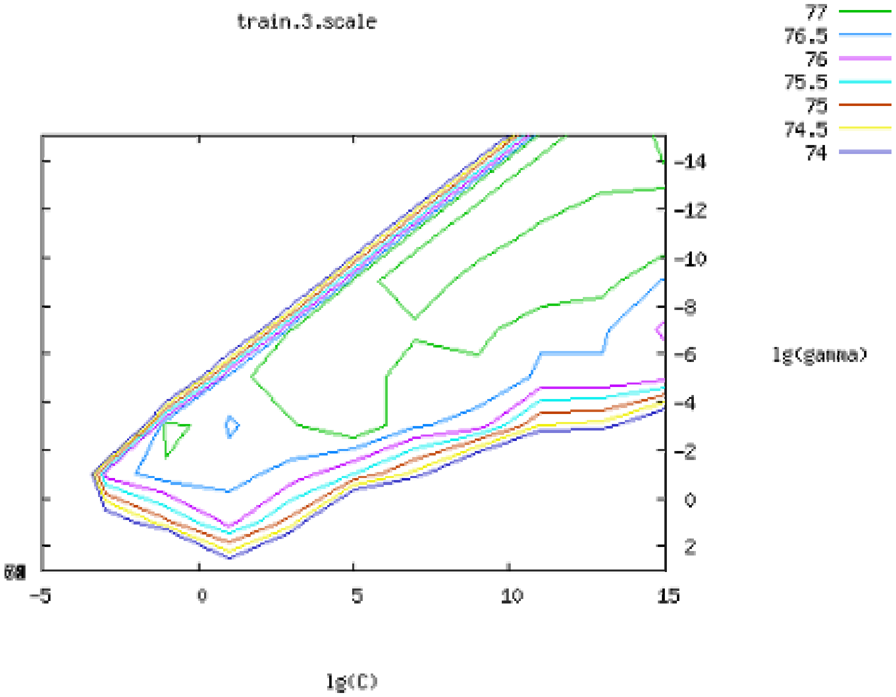

Figure 15: results of canonical kernel learning of Section 4.3 dataset by libSVM.

4.3.1 Gaussian-random landmark vector mapping

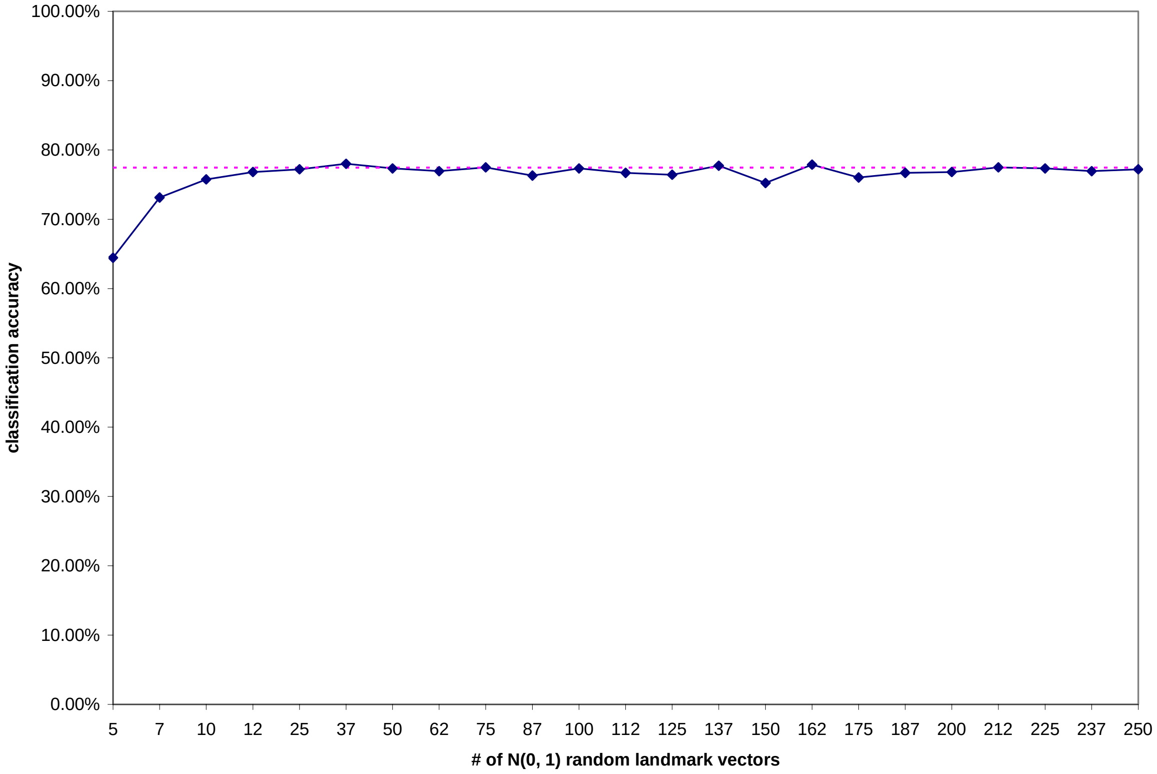

Figure 16: Results of the Gaussian random landmark vector projection experiments for Section 4.3. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark For sufficient number of random vectors the performance of the Power Landmark Framework rivals that of tradition kernel learning framework. This demonstrates the validity of the Power Landmark Vector Learning Framework. Also note that far less than 759 random vectors, which is the total size of the training set for Section 4.3, were needed in order to achieve performance comparable with traditional kernel learning framework. This demonstrates the potential for dimensionality reduction that exists within the Power Landmark Vector Learning Framework.

4.3.2 Input-space orthonormal landmark vector mapping

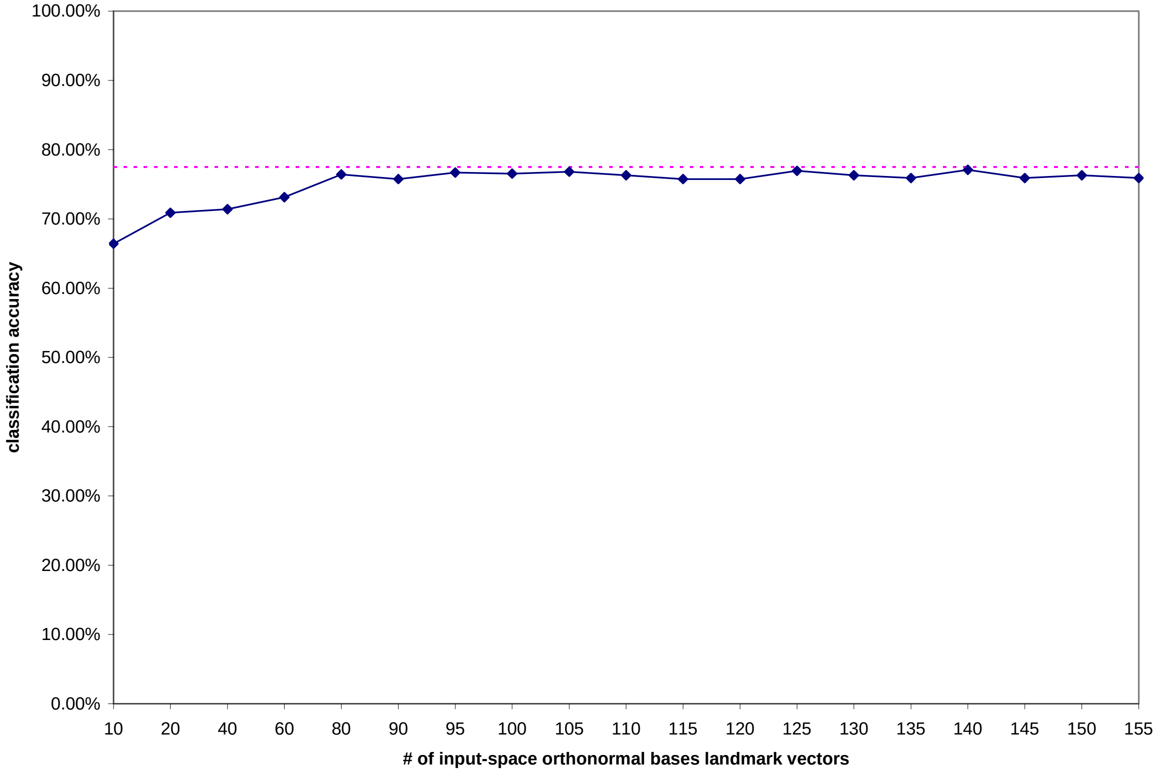

Figure 17: Results of the input-space orthonormal bases landmark vector projection experiments for Section 4.3. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark These results confirm the ability of input-space orthonormal bases landmark vectors to approximate the decision hyperplane, albeit with a slightly less degree of accuracy compared to the Gaussian normal random landmark vectors if one were to examine the numerical data.

4.3.3 Data visualization

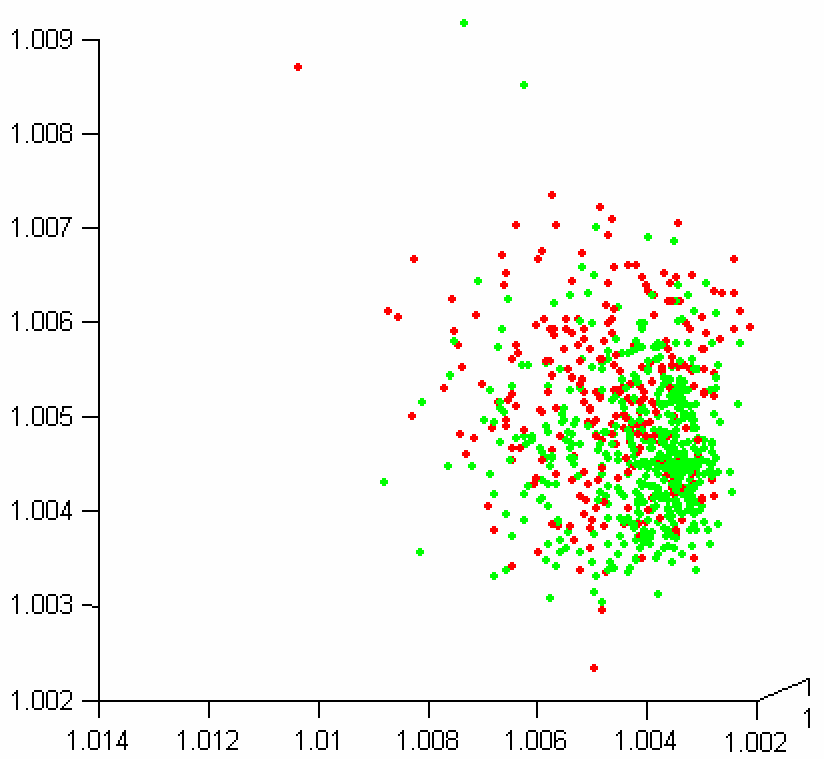

Figure 18: Results of data visualization experiment using Gaussian normal landmark random vectors for Section 4.3.

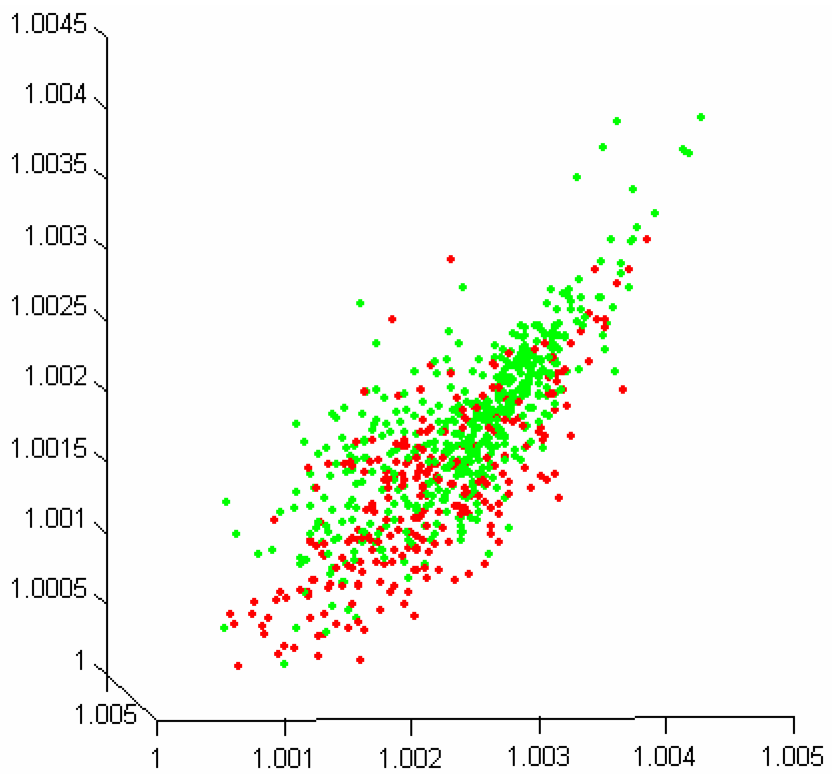

Figure 19: Results of data visualization experiment using randomized input-space orthonormal bases landmark vectors for Section 4.3.

Remark Each green vector indicates a patient who is tested and found to be negative for diabetes. Each red vector indicates a patient who is tested and found to be positive for diabetes. The data visualization results for random landmark vectors in this section seem to cluster better than the data visualization results for the input-space orthonormal bases landmark vectors. The green vectors are seen tightly clustered together in Figure 18 with the red vectors appearing in a layer in front of the cluster from the vantage point of the reader. These visualization results seen in Figure 18 seem to resonate with the case seen in Figure 18, where red vectors indicating a diseased condition are also spread out more than green vectors indicating a relatively healthy condition. The visualization results in Figure 18 also means that patients with diabetes will have more varied vital sign readings than patients without diabetes, making a diabetic condition potentially harder to diagnose than non-diabetic conditions because the vital sign characteristics for the diabetic conditions are spread out into a broader spectrum.

4.3.4 Non-orthogonality test

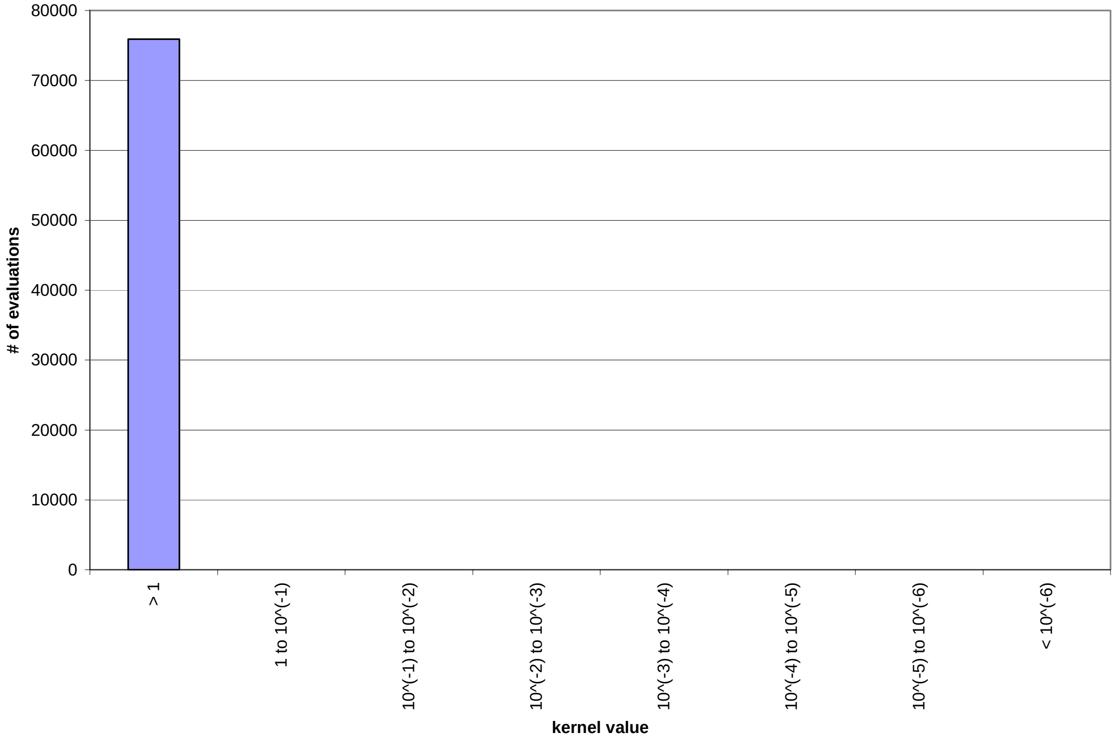

Figure 20: Results of the non-orthogonality test experiment for Section 4.3.

Remark This experiment demonstrates that all inner-products under the Gaussian kernel for Section 4.3 tend to be “well-behaved” in that their values lie well away from small values toward the right-side of the *x*-axis in Figure 20. Since small kernel values indicate near-orthogonality, these “well-behaved” values mean that Gaussian random vectors generated for Section 4.3 tend to be rather related to most training vectors in Section 4.3.

4.3.5 Linear independence of random vectors

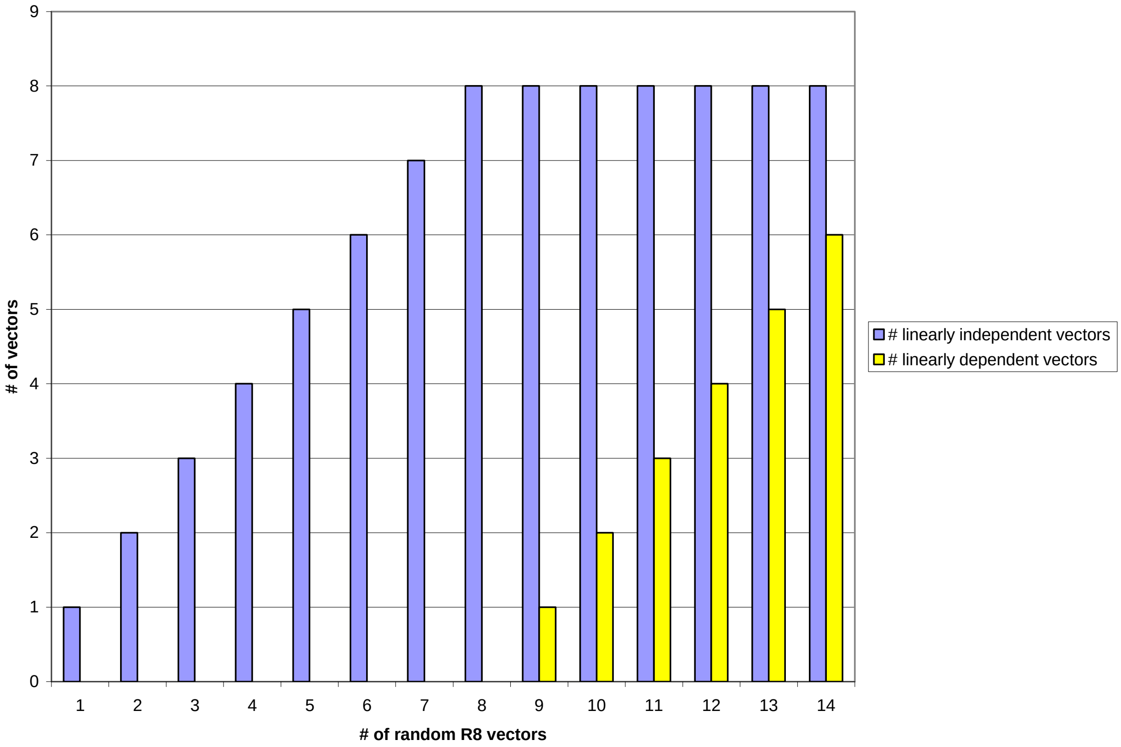

Figure 21: Results of the linear independence of random vectors experiment for the input space of Section 4.3.

Remark As expected, the generated $\mathbb{R}^8$ random vectors form linearly independent sets. The presence of linearly dependent random vectors in the later trials is due to the number of random vectors generated exceeding the dimensionality of the input space of the training vectors.

4.4 Wisconsin breast cancer dataset

The Wisconsin breast cancer dataset is provided at the courtesy of Dr. William H. Wolberg of the University of Wisconsin Hospitals, Madison (Mangasarian and Wolberg, 1990; Wolberg and Mangasarian, 1990; Mangasarian et al., 1990; Bennett and Mangasarian, 1992). The actual data itself is obtained by the author of this thesis from the FTP site of the University of California at Irvine.

This data has been used in the past by (Wolberg and Mangasarian, 1990) and (Zhang, 1992) to investigate the possibility of applying machine learning to the task of learning to distinguish between benign outgrowth and malignant outgrowth. The patients who volunteered for this study are divided into 8 groups that participated in the study separately in a period of time spanning from January 1989 to November 1991. Wolberg and Zhang's algorithms achieved prediction accuracies of 92.2% to 95.9%. These accuracies are surpassed slightly by the power landmark vector learning framework which had achieved prediction accuracies ranging from 96.2% to 97.8%.

There are a total of 699 training instances in this dataset. Out of these training instances, only 683 are actually used. The 16 remaining training instances had missing attributes, in the form of '?' features, in the original dataset. These 16 instances were thus unusable to power landmark learning and had to be discarded. Each training instance has 9 features, one of which is the patient id number and is disregarded by the power landmark learning framework because it has no relationship to the patient's cancer conditions. The 8 remaining features used by the learning framework have to do with the various biological properties of the outgrowth such as clump thickness, cell size, cell shape, nucleus state and marginal adhesion. A class variable comes at the end of each training instance that provides a label for that training instance. The class variable takes on the binary value of 2 and 4 with 2 indicating a benign outgrowth and 4 indicating a malignant outgrowth.

Canonical kernel learning using traditional kernel learning framework is performed at the beginning of this section of experiments by libSVM. The results of the learning are shown in Figure 22. The canonical cross-validated prediction accuracy for this section is established by libSVM to be 97.22%.

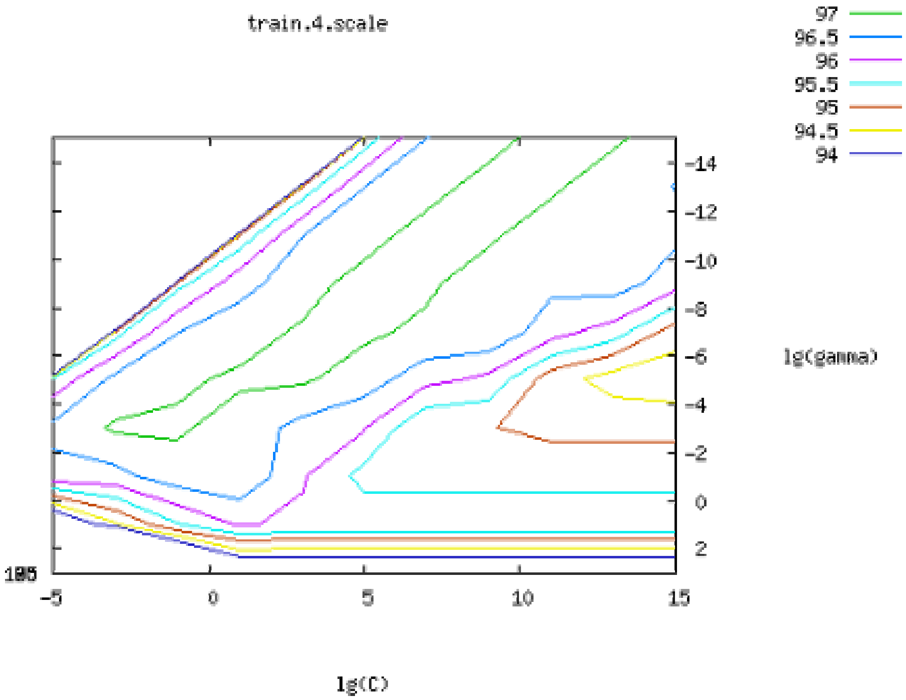

Figure 22: results of canonical kernel learning of Section 4.4 dataset by libSVM.

4.4.1 Gaussian-random landmark vector mapping

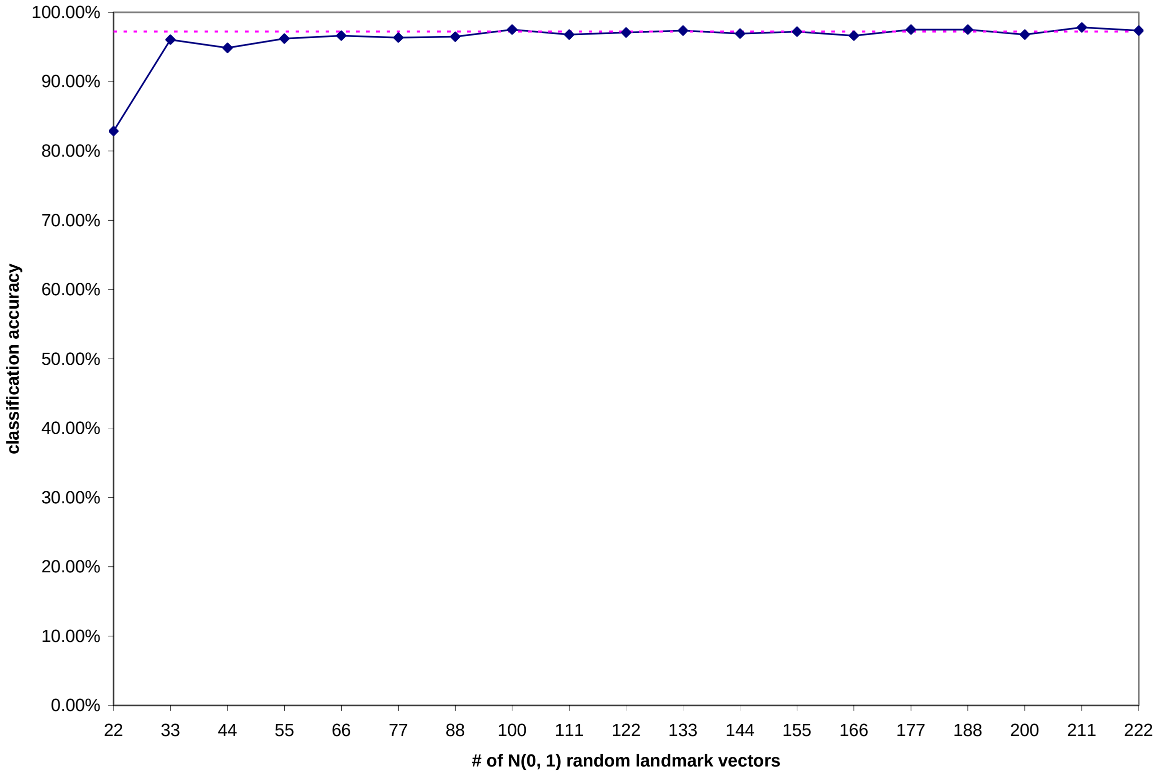

Figure 23: Results of the Gaussian random landmark vector projection experiments for Section 4.4. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark For sufficient number of random vectors the performance of the Power Landmark Framework rivals, and even exceeds on several occasions, that of tradition kernel learning framework. This demonstrates the validity of the Power Landmark Vector Learning Framework. Also note that far less than 683 random vectors, which is the total size of the training set for Section 4.4, were needed in order to achieve performance comparable with traditional kernel learning framework. This demonstrates the potential for dimensionality reduction that exists within the Power Landmark Vector Learning Framework.

4.4.2 Input-space orthonormal landmark vector mapping

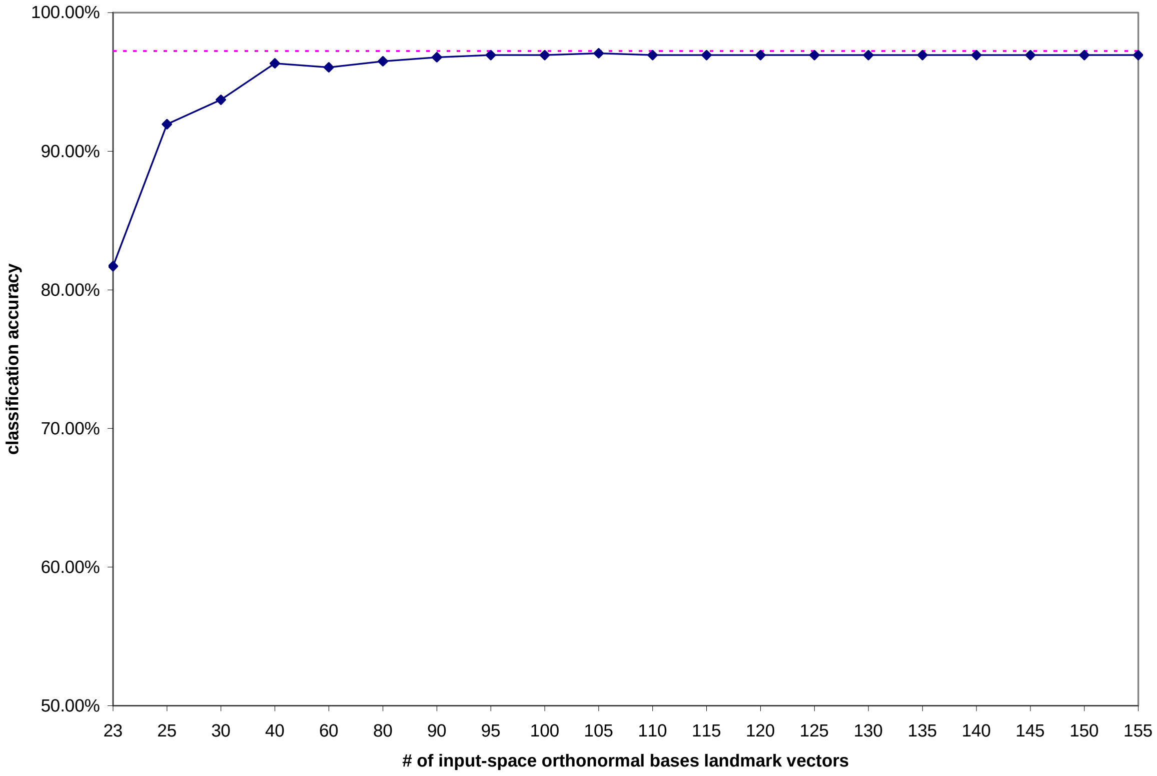

Figure 24: Results of the input-space orthonormal bases landmark vector projection experiments for Section 4.4. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark The performance of the input-space orthonormal bases landmark vectors in this experiment has been surprisingly good. Although these landmark vectors consistently deliver a slightly less degree of accuracy compared to the canonical kernel learning framework, they have been consistently more accurate than most corresponding sets of random landmark vectors used in Section 4.4.1. Readers should note that the *y*-axis of Figure 24 has to be magnified to the 50% to 100% range in order to sufficiently delineate the difference between the landmark vector performance plot and the canonical kernel learning performance denoted to by the dashed line, whereas such a magnification was not necessary for Figure 23. This is a phenomenon that is not observed in any other experiments of the same type.

4.4.3 Data visualization

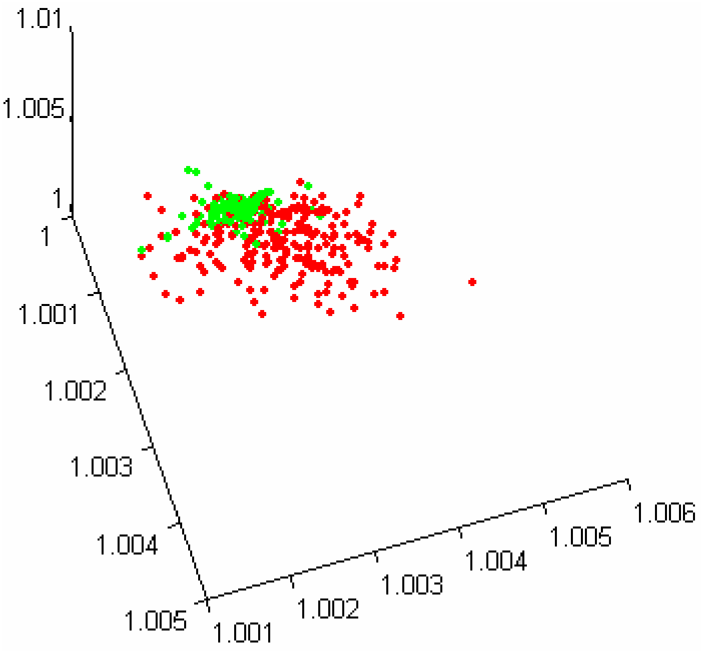

Figure 25: Results of data visualization experiment using Gaussian normal landmark random vectors for Section 4.4.

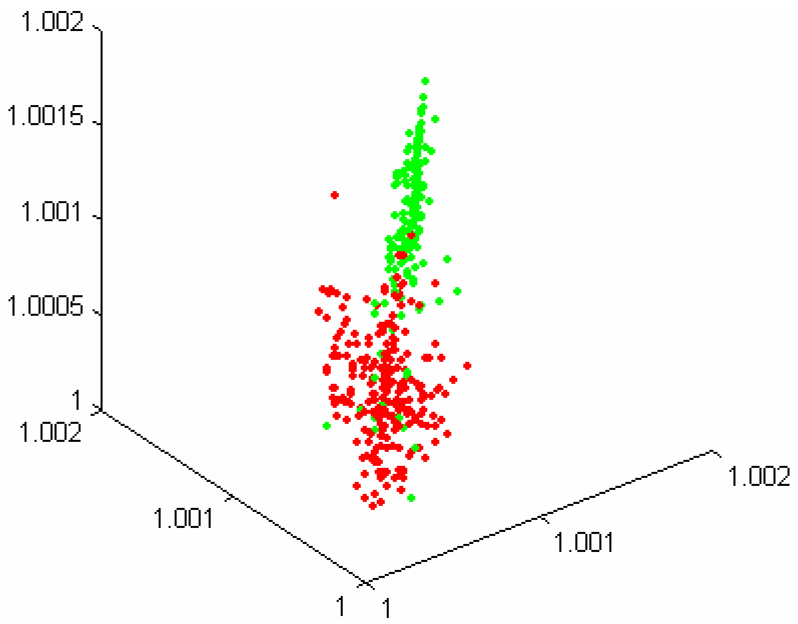

Figure 26: Results of data visualization experiment using randomized input-space orthonormal bases landmark vectors for Section 4.4.

Remark Each green vector indicates a patient who is tested and found to have benign outgrowth on the breast. Each red vector indicates a patient who is tested and found to have malignant outgrowth on the breast. The data visualization for input-space orthonormal bases landmark vectors in this section gives particularly excellent visualization results of the two classes of vectors. The green vectors are seen tightly clustered into a narrow long band in Figure 26 that is well separated from the red vectors, which also appear well clustered in a layer that appears in front of the narrow green band from the vantage vector of the reader. Also of note from Figure 25 is the fact that the green vectors representing benign outgrowth are more clustered than the red vectors representing malevolent outgrowth, this might indicate that benign outgrowths have more in common with each other than malignant outgrowth. It also indicates that malignant outgrowths, which represent the onset of true instances breast cancer, might be more difficult to diagnose than benign outgrowths due to their more varied characteristics that cause the red vectors to spread out more. The data visualization contained in Figure 26 represents the best visualization achieved out of all three sets of experiments and is a demonstration of the data visualization utility of the power landmark vector learning framework.

4.4.4 Non-orthogonality test

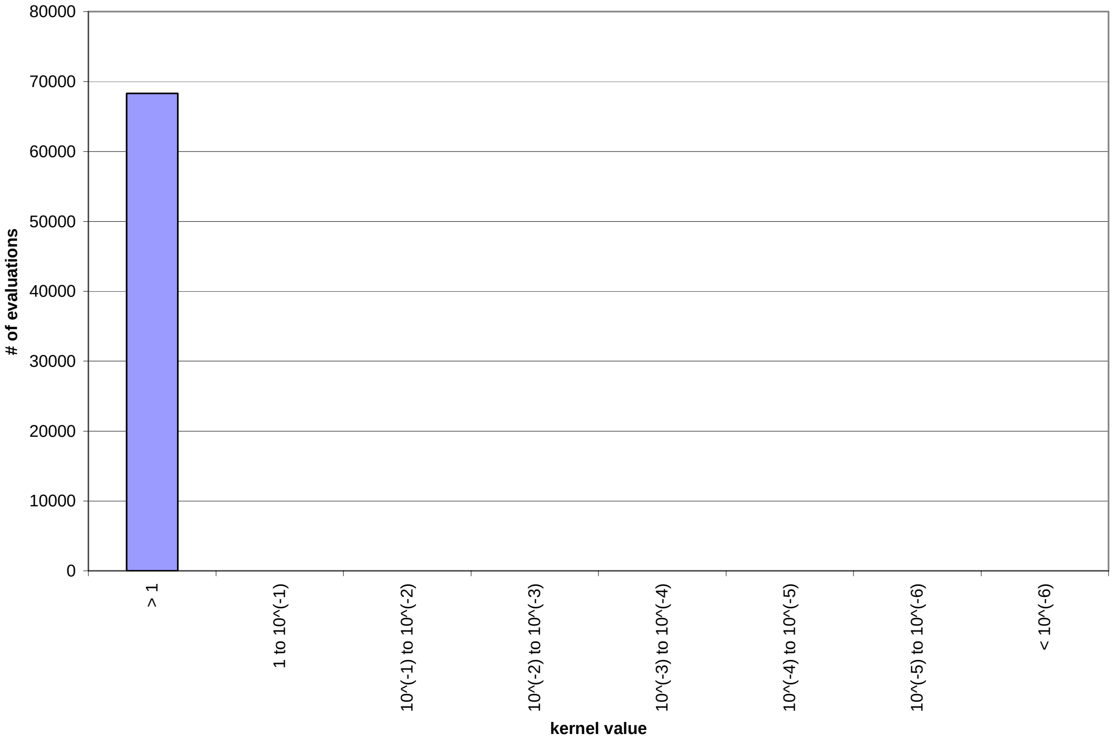

Figure 27: Results of the non-orthogonality test experiment for Section 4.4.

Remark This experiment demonstrates that all inner-products under the Gaussian kernel for Section 4.4 tend to be "well-behaved" in that their values lie well away from small values toward the right-hand side of the *x*-axis in Figure 27. Since small kernel values indicate near-orthogonality, these "well-behaved" values mean that Gaussian random vectors generated for Section 4.4 tend to be rather related to most training vectors in Section 4.4.

4.4.5 Linear independence of random vectors

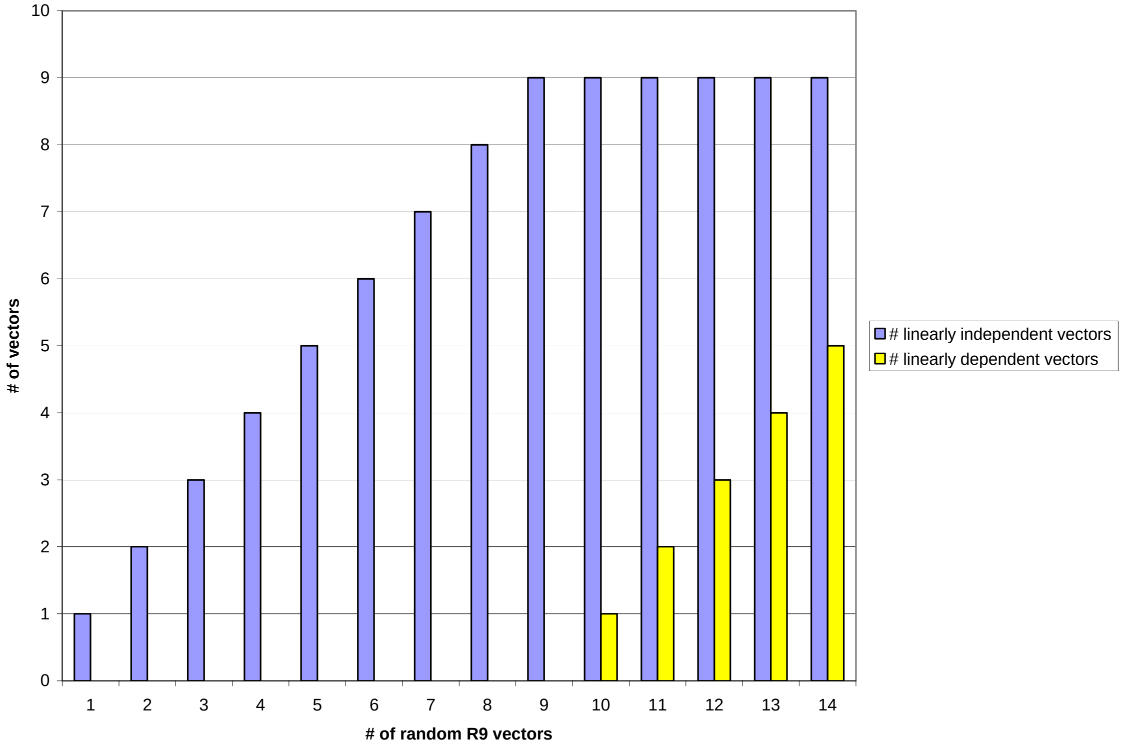

Figure 28: Results of the linear independence of random vectors experiment for the input space of experiment 4.4

Remark As expected, the generated $\mathbb{R}^9$ random vectors form a linearly independent set. The presence of linearly dependent random vectors in the later trials is due to the number of random vectors generated exceeding the dimensionality of the input space of the training vectors.

4.5 Mushrooms Dataset

In order to test the robustness of the power landmark vector learning framework, the mushrooms dataset is used in this final experiment. The mushrooms dataset is a classical dataset renowned for the difficulty of producing a good classifier for it. This dataset is not directly related to bioinformatics proper, but is included here in order to demonstrate the universality of the power landmark vector learning framework for kernels that are good such as the Gaussian kernel. Thanks go to Dr. Li for pointing out this dataset to me.

The mushrooms dataset is also known as the “agaricus-lepiota” dataset which reflects the genus and species names for the particular family of mushrooms being investigated. The dataset is provided at the courtesy of Dr. Jeff Schlimmer of Carnegie Mellon University. Schlimmer extracted the data out of (Lincoff, 1981) The actual data itself is obtained by the author from a mirror of the FTP site of the University of California at Irvine.

This data has been investigated in the past by (Schlimmer, 1987) using the STAGGER program to achieve an asymptotical classification accuracy of 95% after reviewing 1000 labeled training vectors. This data has also been investigated in the past by (Iba et al., 1988) using the HILLARY program to achieve approximately the same accuracy.

There are a total of 8124 labeled training vectors in this dataset. Out of these training vectors, only 4257 are actually used. The remaining training vectors had missing attributes after scaling by libSVM, thus preventing their being used properly by the FP space generating program. Each training vector has 22 features, one of which is the same for all training vectors and is disregarded by the power landmark learning framework because it has no discriminative power. The 21 remaining features used by the learning framework have to do with the various shapes, sizes and colors of various parts of the mushroom and its odor, population and habitat. A class variable comes at the end of each training vector that provides a label for that training vector. The class variable takes on the binary value of “p” and “e” with “p” indicating poisonous and “e” indicating edible.

Canonical kernel learning using traditional kernel learning framework is performed at the beginning of this section of experiments by libSVM. The results of the learning are shown in Figure 29. The canonical cross-validated prediction accuracy for this section is established by libSVM to be 100%.

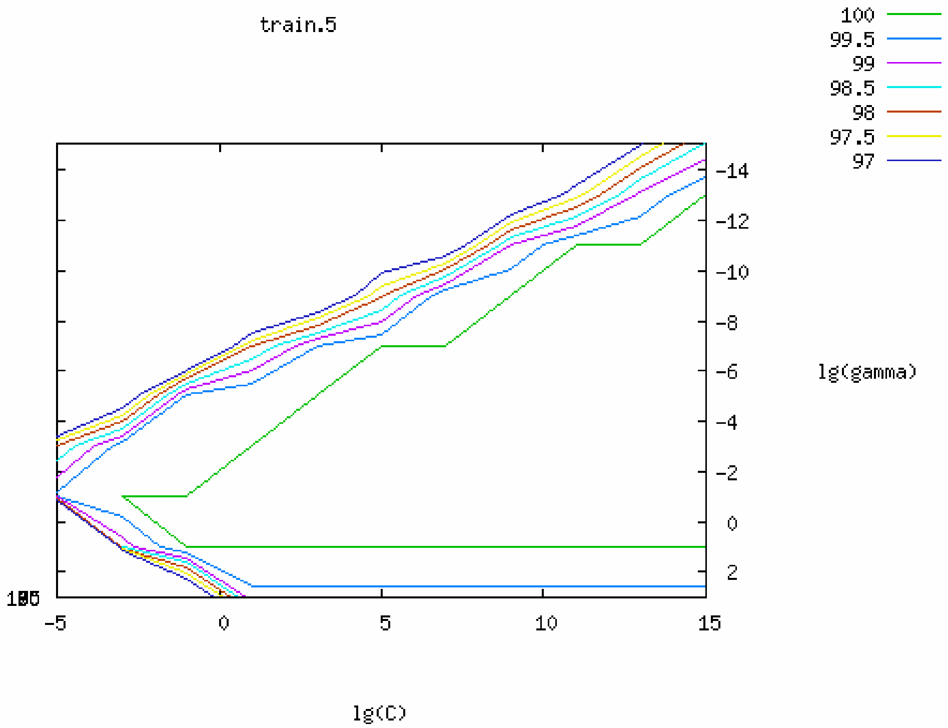

Figure 29: results of canonical kernel learning of Section 4.5 dataset by libSVM.

4.5.1 Gaussian-random landmark vector mapping

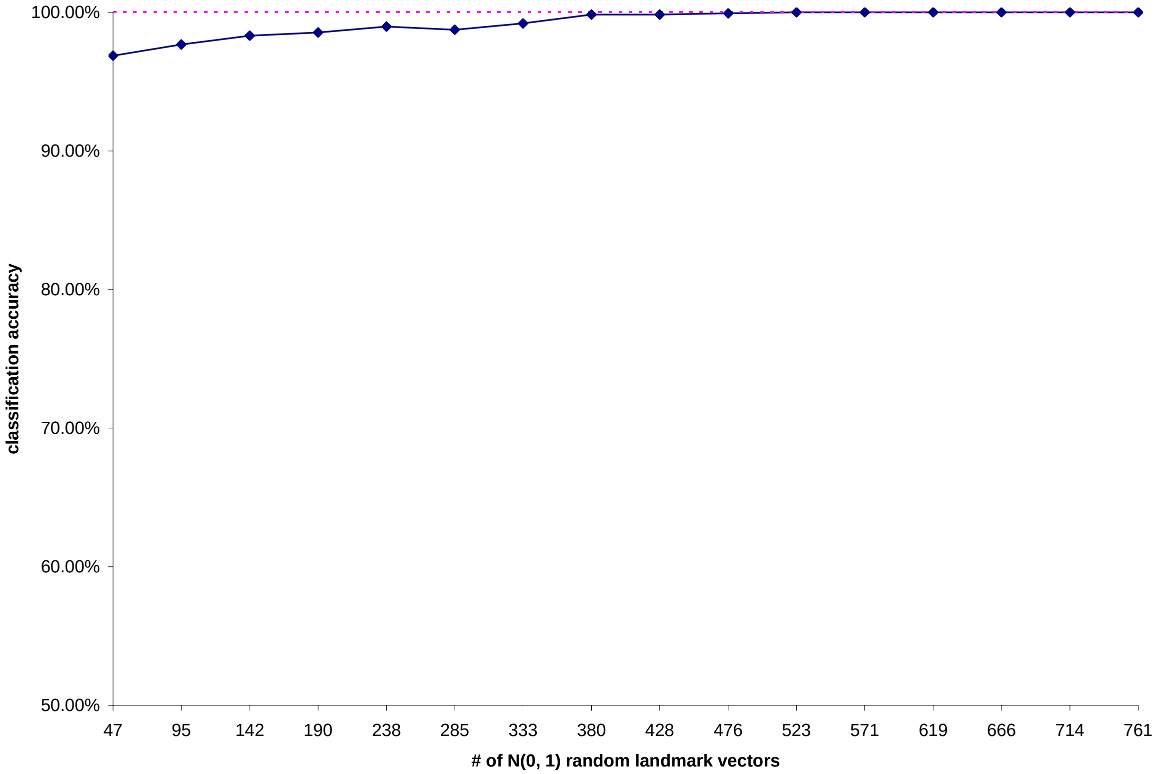

Figure 30: Results of the Gaussian random landmark vector projection experiments for Section 4.5. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark For sufficient number of random vectors the performance of the Power Landmark Framework rivals that of the tradition kernel learning framework. This demonstrates the validity of the Power Landmark Vector Learning Framework. Also note that far less than 4257 random vectors, which is the total size of the training set for Section 4.5, were needed in order to achieve performance comparable with traditional kernel learning framework. This demonstrates the potential for dimensionality reduction that exists within the Power Landmark Vector Learning Framework.

4.5.2 Input-space orthonormal landmark vector mapping

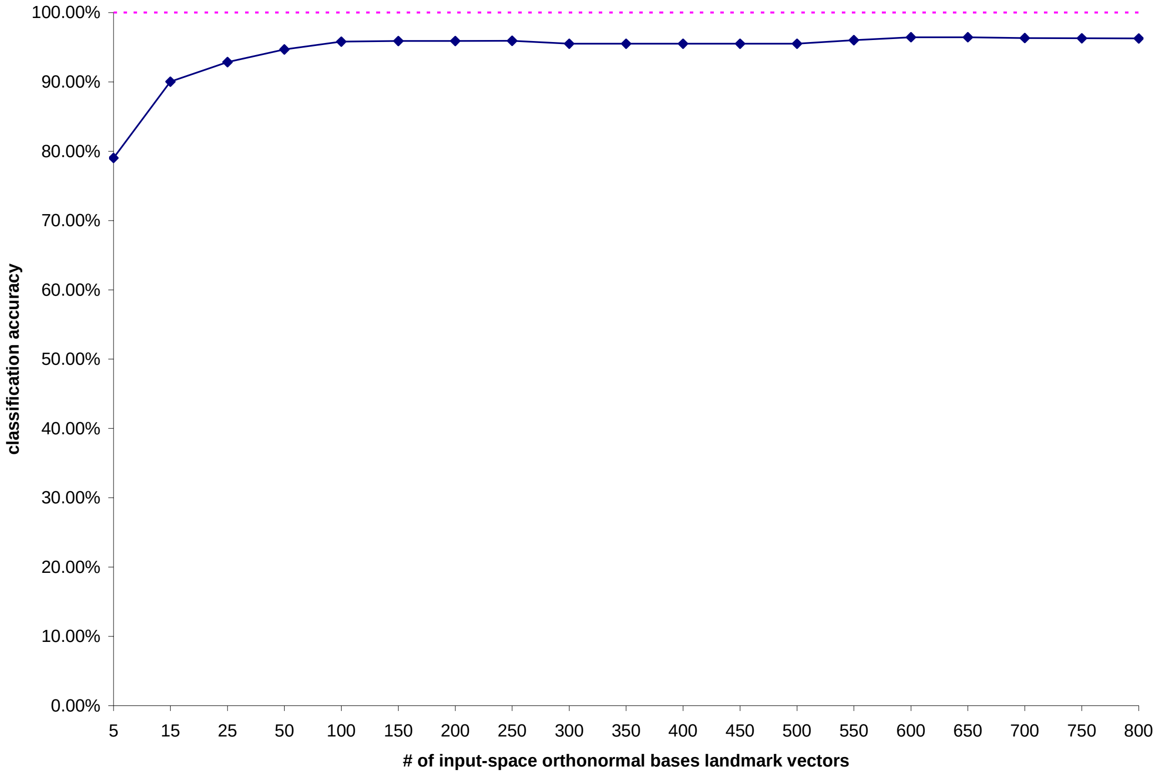

Figure 31: Results of the input-space orthonormal bases landmark vector projection experiments for Section 4.5. The horizontal dashed line indicates canonical classification accuracy under traditional kernel learning framework.

Remark These results confirm the ability of input-space orthonormal bases landmark vectors to approximate the decision hyperplane, albeit with a 5% decrease of accuracy compared to the best Gaussian normal random landmark vectors if one were to examine the details.

4.5.3 Data visualization

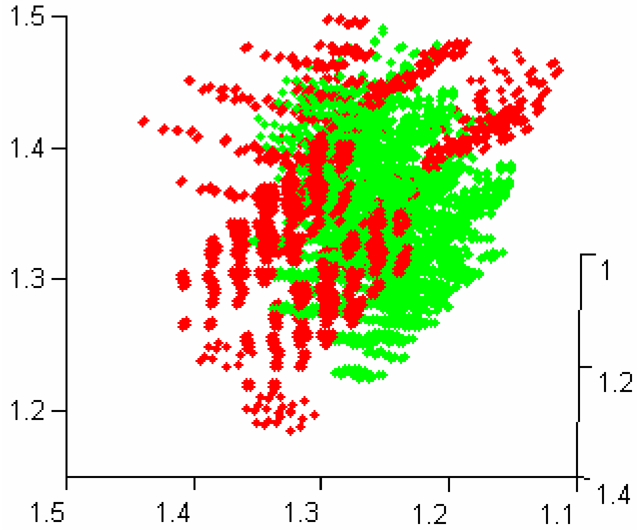

Figure 32: Results of data visualization experiment using Gaussian normal landmark random vectors for Section 4.5.

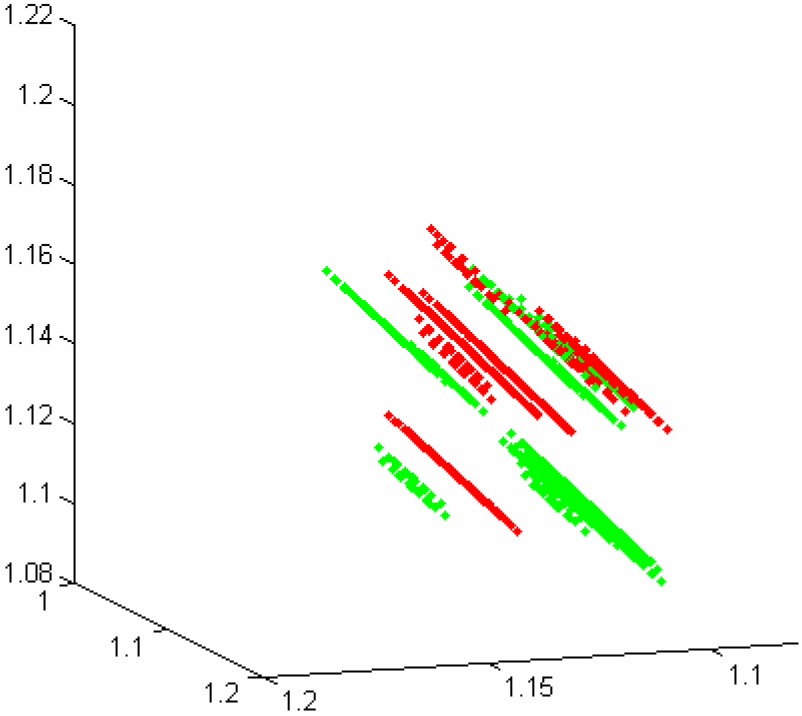

Figure 33: Results of data visualization experiment using randomized input-space orthonormal bases landmark vectors for Section 4.5.

Remark Each green vector indicates a mushroom that is tested and found to be edible. Each red vector indicates a mushroom that is tested and found to be poisonous. Similar to the data visualization experiment in Section 4.4, the data visualization for input-space orthonormal bases landmark vectors in this section gives particularly excellent visualization results of the two classes of vectors. The green vectors are seen tightly clustered into several narrow bands in Figure 33. The red vectors are also sent tightly clustered into several narrow bands in Figure 33. The green bands and the red bands are well separated from each other.

4.5.4 Non-orthogonality test

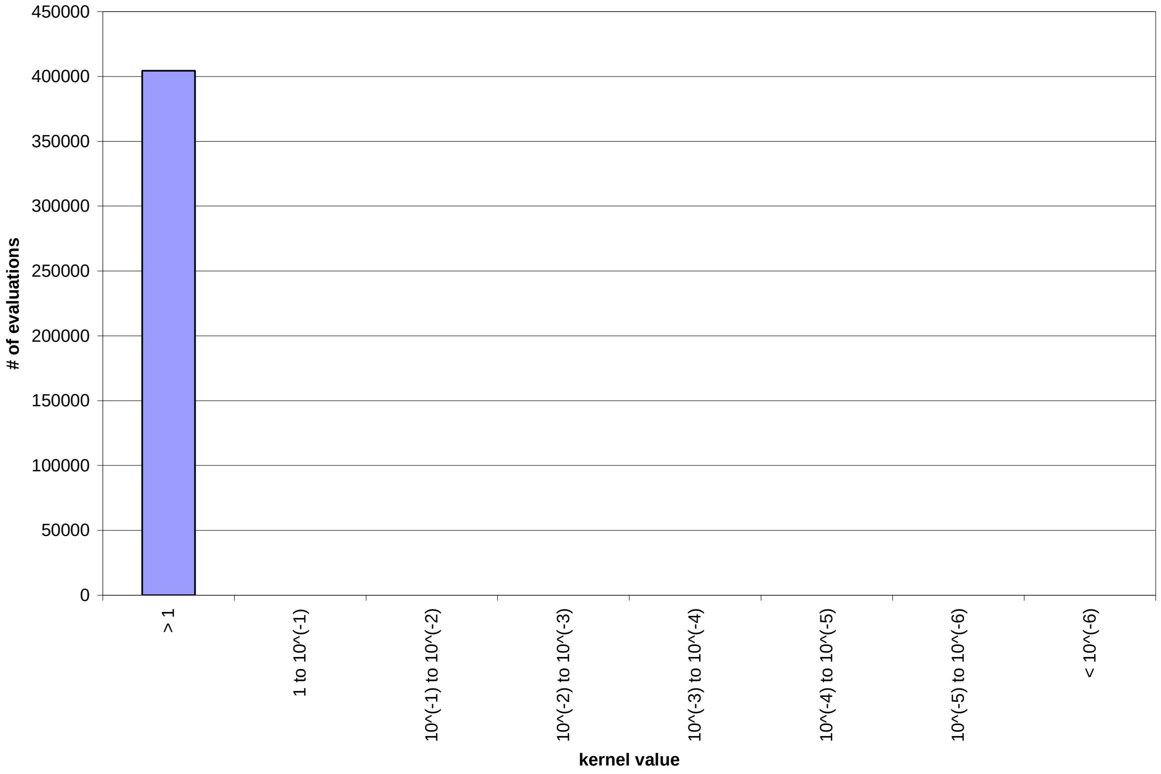

Figure 34: Results of the non-orthogonality test experiment for Section 4.5.

Remark This experiment demonstrates that most inner-products under the Gaussian kernel tend to be "well-behaved" in that their values lie well away from small values toward the right-side of the *x*-axis in Figure 34. Since small kernel values indicate near-orthogonality, these "well-behaved" values mean that Gaussian random vectors tend to be rather related to most training vectors.

4.5.5 Linear independence of random vectors

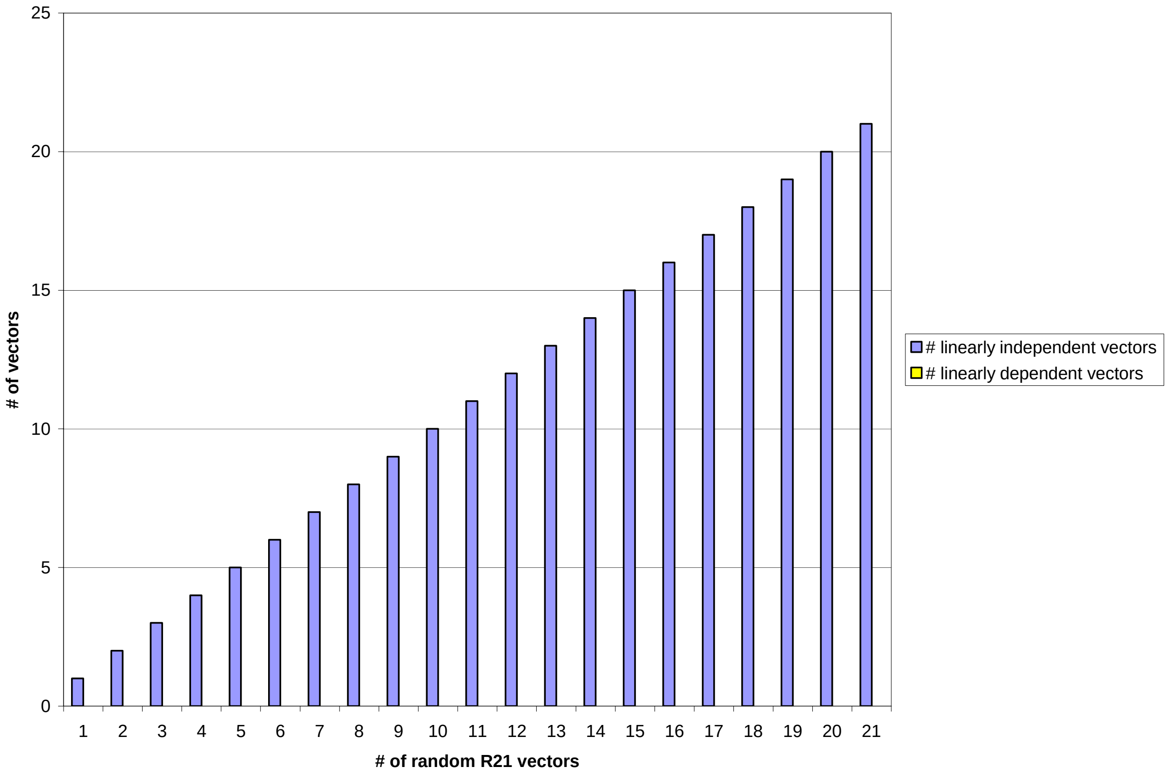

Figure 35: Results of the linear independence of random vectors experiment for the input space of experiment 4.5

Remark As expected, the generated $\mathbb{R}^{21}$ random vectors form a linearly independent set.

4.6 Chapter summary

The Power Landmark Vector Learning Framework introduced in Chapter 3 was applied to a variety of bioinformatics learning situations that have been previously studied in literature. The objective of these experiments is to verify the theoretical arguments made in Chapter 3 and also to compare the performance of the power landmark vector learning framework against more traditional kernel learning framework. The results derived are all very satisfactory. They indicate that the performance of the power landmark vector learning framework on all the bioinformatics learning situations investigated in this chapter are comparable to that of traditional kernel learning framework. In particular, the power landmark vector learning framework was able to provide good data visualization results for the Pima aboriginal diabetes dataset and the Wisconsin breast cancer dataset with particularly good results for the latter dataset under the orthonormal bases landmark vectors. The experiments also confirmed that randomly generated Gaussian normal vectors tend to form linearly independent sets and that the kernel evaluations under most learning situations in the framework tend to be “well-behaved”.

Chapter 5
Conclusions and Future Work

In this thesis a framework called the Power Landmark Vector Learning Framework has been introduced. The Power Landmark Vector Learning Framework uses Gaussian random vectors or orthonormal bases vectors in the input space as “landmark vectors” to perform learning of actual training data. An entire theoretical framework has been established in this thesis to justify the validity of applying the Power Landmark Vector Learning Framework to linear, quadratic and Gaussian kernels and their associated finite-dimensional Euclidean input spaces. Subsequently, several experiments were designed to test the power landmark vector learning framework on several bioinformatics datasets that have been previously studied in the literature to establish final and conclusive evidence for the efficacy of the power landmark vector learning framework.

5.1 Contributions

The main contribution of this thesis has been the extension of Blum’s landmark vector learning framework into a more complete and general framework capable of using Gaussian normal random vectors and orthonormal bases vectors as landmark vectors in generating the linear feature space required for the final learning. The most significant theoretical contribution of this thesis lies in the proving of two original theoretical results — the non-collapsible theorem and the non-orthogonal theorem — for the Gaussian kernel. This thesis has also established the non-collapsible principle and non-orthogonal principle as two necessary, but not sufficient, conditions to test whether the power landmark vector learning framework will work successfully with a given kernel or not. Sufficiency has subsequently been established from an empirical perspective. Finally, this thesis' explanations of the relevant background mathematical materials and concepts, mostly from the area of advanced linear algebra and functional analysis, that is required for the readers to understand both Blum's framework and the power framework of this thesis, serve to further educate the readers and provide them with a clearer and firmer grasp of the theories and mathematical beauty of the field of kernel-enabled machine learning.

5.2 Future research

The works of this thesis opens up an extremely rich and fertile field for future investigations and research. Listed below are several main areas of future endeavour that may bear fruitful findings.

5.2.1 Extension into other kernels and non-vectorial input spaces

Perhaps the most urgently required work is to extend the utility of the Power Landmark Vector Learning Framework into other kernels and input spaces. In this thesis, the Power Landmark Vector Learning Framework has been shown to work for the Gaussian kernel coupled with the finite-dimensional ordinary Euclidean $\mathbb{R}^n$ vectorial input space only. However, there exists far greater number of other kernels than the Gaussian kernel, and there also exist a whole variety of possible input spaces, some of them not vectorial at all. A good example of a popular non-Gaussian kernel with a non-vectorial input space is the class of kernels that work with textual strings that are described in detail in Chapter 10 of (Shawe-Taylor and Cristianini, 2004). The input spaces for these kernels are the Σ^n spaces, where Σ is the set of basic symbols for the language of the text. For example, for the English language Σ = {‘A’, …, ‘Z’, ‘a’, …, ‘z’, punctuation marks, space, ‘0’, …, ‘9’}. The training set for these textual kernels might be the concatenated texts of paragraphs, entire articles or even short books. The landmarks needed for these kernels to work with might be short, randomly generated text fragments such as ‘ca’, ‘dbd’, ‘sad’, ‘dam’, etc. It could also be possible that more robust landmarks are required for text learning to work with the power landmark vector learning framework. The extension into text kernels could have ready applications in terms of genome and proteome related learning tasks, where long strings of DNA and amino acid molecules are abstracted into long strings of letters and kernels are employed to find patterns and similarities exhibited by massive quantities of those strings. The landmarks used in these circumstances could be of the form ‘A’, ‘T’, ‘G’, ‘GATC’, ‘Q’, ‘QVED’, etc.

5.2.2 Further work on dimensionality reduction

Together, the non-collapsible theorem and the non-orthogonal theorem provide a solid foundation for justifying the validity of the power landmark vector framework on a basic level. They also provide a basic guarantee that at most the same number of landmarks as the number of training vectors is needed. Experiments in Chapter 4, however, show that far less landmark vectors are needed than the total number of training vectors. This implies that the power landmark vector learning framework could also be used for dimensionality reduction.

However, theoretical work is lacking on the exact extent to which this dimensionality reduction is possible under the power landmark vector learning framework, and the qualities of the dataset itself that would make this dimensionality reduction possible. This thesis proposes a tentative direction for future research where the kernel principle component analysis (KPCA) algorithm is used to determine the actual dimensionality of the training data and the landmark vector projections of the principle components be used to generate the FP space for linear learning. It is also conceivable that there exist better theoretical frameworks capable of justifying the ability of most data sets to be dimensionally reduced in the power landmark vector learning framework. These frameworks should also be investigated.

5.2.3 Random vectors and non-orthogonality

In this thesis experiments were used to demonstrate that, with high probability, identically and independently generated random Gaussian normal vectors are linearly independent. Further probabilistic theoretical arguments are needed to conclusively show that this is indeed the case. The orthogonality argument whose validity is vindicated by the experiments would also benefit from more theoretical studies to justify why, from the perspective of actual machine learning data, would the non-orthogonality argument work out so well with the Gaussian kernel returning well-behaved values most of the time. Furthermore, the non-orthogonal argument, by itself, seems to be a slightly weak argument in justifying the superior performance observed with power landmark vector learning problems observed in the experiments as it only characterizes the impossibility of the lower-bound worst-case problems. Thus, the power landmark vector learning framework would greatly benefit from a stronger theoretical framework able to fully provide probabilistic measures on the performance of power landmark vector learning framework in real machine learning situations.

5.2.4 Johnson-Lindenstrauss random projection

As mentioned towards the end of Sections 3.5 and 3.6, it is necessary to show that the Johnson-Lindenstrauss random projection would approximately preserve linear margins before we could conclude that the FP space remains linearly separable under the linear and PSD kernels. A possible starting point for showing that the Johnson-Lindenstrauss random projection approximately preserves linear margins is the fact that pair-wise distances and angles between the linearly separable vectors in distribution P is approximately preserved in the $F(P)$ space. This preservation of distances and angles would preserve the orientation of the high dimensional vectors to each other in the low dimensional subspace, hence preserving the linear quality of the margin.

5.2.5 Better Data Visualization

Each data visualization three-dimensional MATLAB plot has been rotated and oriented in a way that attempts to delineate the separation between vectors with different labels as good as possible as per suggestions from Dr. McConkey. However, as earlier experiments have shown, small numbers of landmark vectors tend to corrupt the linear margin that exists between data of differing classes in the feature space. Since data visualization is about pushing the power landmark vector learning framework to the limit of using only *three* landmark vectors, that the resulting data visualizations are less than idea is thus understandable. In the future, it would be good to investigate alternative ways of performing the data visualization under the power landmark vector learning framework such that the linear margin that exists between data of differing classes in the feature space gets preserved to as great an extent as possible.

Appendix A
Pima Aboriginals Dataset in Detail

The information found in tabular forms in this and the next appendix that are meant to fully describe the nature and the content of two of the datasets used for the thesis lab are taken (almost) verbatim from the "names" file that accompanies each dataset.

The dataset used in the second experiment is the Pima aboriginals dataset. The original name of this dataset is Pima Indians dataset, but the use of the word "Indians" to refer to the aboriginal people of the Americas maybe politically incorrect in the modern age. Therefore the name "Pima aboriginals" is used throughout thesis proper to refer to this dataset. If the words "Pima Indians" are seen anywhere in this appendix they are intended to mean "Pima aboriginals" by the thesis author.

Table 1: Detailed summary of Pima aboriginals dataset.

Original Owners	National Institute of Diabetes and Digestive and Kidney Diseases
Donor of database	Vincent Sigillito (vgs@aplcen.apl.jhu.edu) Research Center, RMI Group Leader Applied Physics Laboratory The Johns Hopkins University

	Johns Hopkins Road Laurel, MD 20707 (301) 953-6231
Past Usage	Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., and Johannes, R. S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Symposium on Computer Applications and Medical Care* (pp. 261--265). IEEE Computer Society Press. The diagnostic, binary-valued variable investigated is whether the patient shows signs of diabetes according to World Health Organization criteria (i.e., if the 2 hour post-load plasma glucose was at least 200 mg/dl at any survey examination or if found during routine medical care). The population lives near Phoenix, Arizona, USA. Results: Their ADAP algorithm makes a real-valued prediction between 0 and 1. This was transformed into a binary decision using a cutoff of 0.448. Using 576 training instances, the sensitivity and specificity of their algorithm was 76% on the remaining 192 instances.

Relevant Information	Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage. ADAP is an adaptive learning routine that generates and executes digital analogs of perceptron-like devices. It is a unique algorithm; see the paper for details.
Size of training set	768 labeled training vectors
Number of features	8
Number of training vectors with missing features	0

Also note that the terms "feature" and "attribute" are equivalent. The term "feature" is used more frequently in machine learning while the term "attribute" is used more frequently in natural science. This thesis belongs in the field of machine learning. Thus, the term "feature" is used throughout this thesis.

Table 2: List of features of the Pima aboriginals dataset

#	**Feature description**	**μ**	**σ**

1	Number of times pregnant	3.8	3.4
2	Plasma glucose concentration a 2 hours in an oral glucose tolerance test	120.9	32.0
3	Diastolic blood pressure (mm Hg)	69.1	19.4
4	Triceps skin fold thickness (mm)	20.5	16.0
5	2-Hour serum insulin (μU/ml)	79.8	115.2
6	Body mass index (weight in kg/(height in m)^2)	32.0	7.9
7	Diabetes pedigree function	0.5	0.3
8	Age (years)	33.2	11.8

Again, the terms “label” and “class” are equivalent. The term “label” is used more frequently in machine learning while the term “class” is used more frequently in natural science. This thesis belongs in the field of machine learning. Thus, the term “label” is used throughout this thesis.

Table 3: List of labels of the Pima aboriginals dataset

Label Value	**Label description**	**# of instances with label**
0	Tested negative for diabetes	500
1	Tested positive for diabetes	268

Based on the above output model, two more experiments were done for this appendix on the Pima aboriginals dataset. The first experiment extracts the α weights of all the support vectors and produces a bar plot of the magnitudes of the weights. The resulting bar plots are shown in Figure 36 and Figure 37. The second experiment performs a data visualization, much like the data visualizations shown in the main thesis, on the set of support vectors. The result of this data visualization is shown in Figure 38.

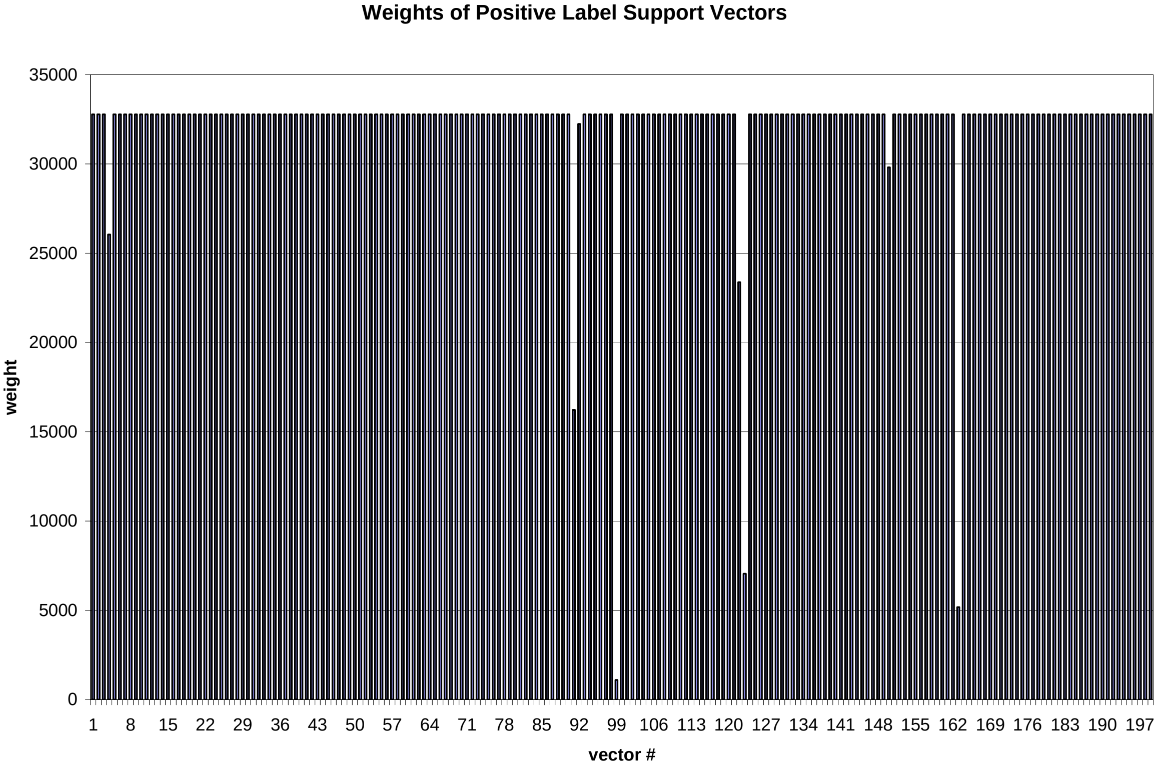

Figure 36: Weights of positive label support vectors for the Pima aboriginals dataset.

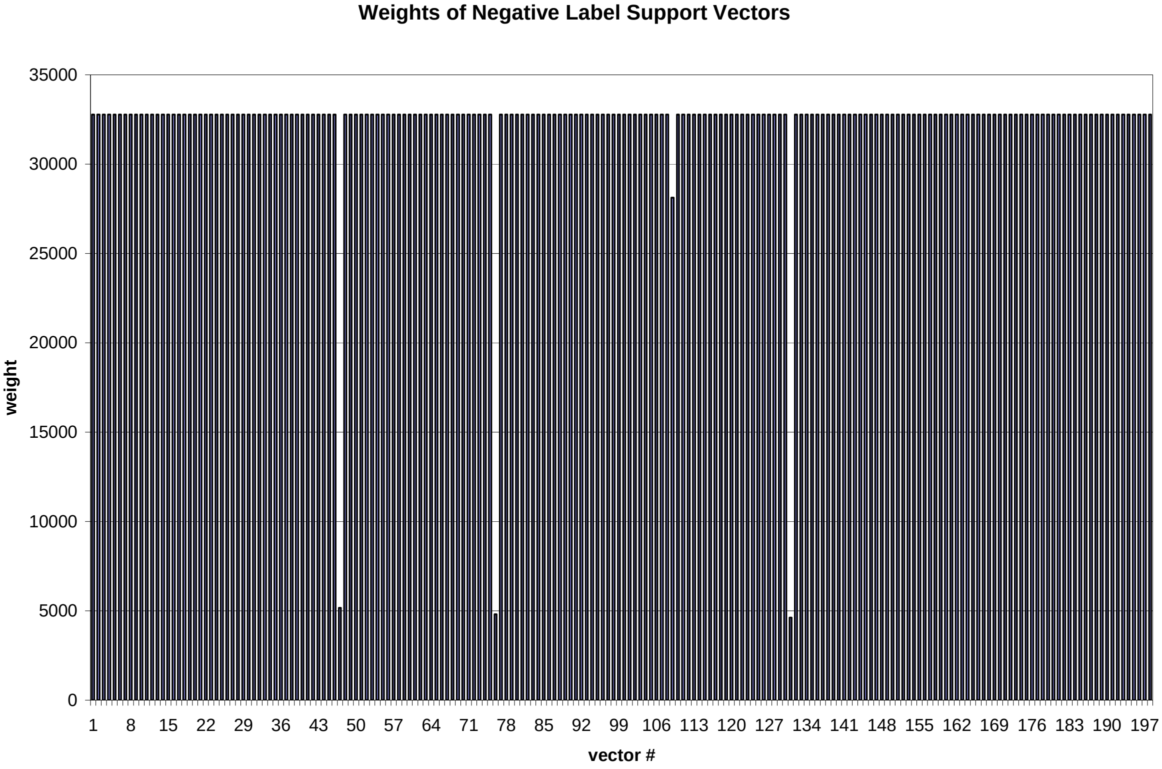

Figure 37: Weights of negative label support vectors for the Pima aboriginals dataset.

As we can see from Figure 36 and Figure 37, the weights of all support vectors of both label types are fairly consistent with the exception of a few support vectors which are not weighed as heavily as the other weight vectors. With Figure 36 and Figure 37, a biologist may notice the vectors with the smaller weights and isolate them into another file and do something interesting with them. This is because in machine learning any vectors with a weight that deviates significantly from other weights may hold further information relating to the underlying distribution of the Pima aboriginal diabetes biological problem at hand.

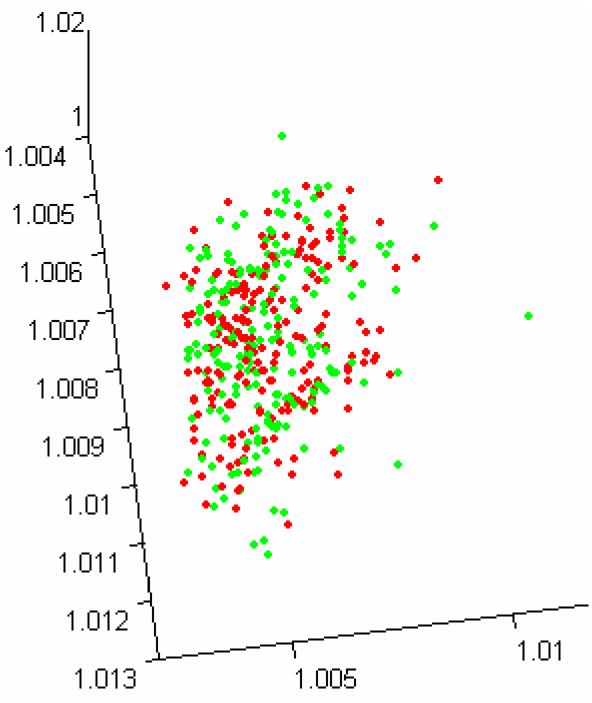

Figure 38: Data visualization of the support vectors of the Pima aboriginals dataset.

As we can see from Figure 38, the green support vectors are situated roughly in the layer above the red support vectors. However, since large number of support vectors are needed to form the resulting hyperplane, the data visualization of this appendix still looks cluttered and less than ideal. Finally, as noted in Figure 18, red vectors indicating a diseased condition are spread out more than green vectors indicating a relatively healthy condition. The visualization results in Figure 18 means that patients with diabetes will have more varied vital sign readings than patients without diabetes, making a diabetic condition potentially harder to diagnose than non-diabetic conditions because the vital sign characteristics for the diabetic conditions are spread out into a broader spectrum.

Future work

During the question and answer phase of this thesis the author responded to Dr. McConkey's question about the transparency of support vector machines with the answer that weights will be given at the end of the training phase that will delineate the importance of the features with respect to one another. Upon close examination of the output model file it was realized that the weights are *not* assigned to the input space features, but rather to each of the training vectors. In the future it would be helpful to find a way of training the vectors under the power landmark vector learning framework such that weights underlining the importance of each feature/attribute in the input file could be given at the end of the training phase. Having this information will greatly aid the biologist in weighing the relative importance of each feature in order to decide how the list of features should be narrow down so that the biologist can concentrate on those features that play the most roles in determining the biological label of the training and testing set.

Appendix B
Wisconsin Breast Cancer Dataset in Detail

Table 4: Detailed summary of the Wisconsin breast cancer dataset.

Citation Request	This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg. If you publish results when using this database, then please include this information in your acknowledgements. Also, please cite one or more of: 1. O. L. Mangasarian and W. H. Wolberg: "Cancer diagnosis via linear programming", SIAM News, Volume 23, Number 5, September 1990, pp 1 & 18. 2. William H. Wolberg and O.L. Mangasarian: "Multisurface method of pattern separation for medical diagnosis applied to breast cytology", Proceedings of the National Academy of Sciences, U.S.A., Volume 87, December 1990, pp 9193-9196. 3. O. L. Mangasarian, R. Setiono, and W.H. Wolberg: "Pattern recognition via linear programming: Theory and application to medical diagnosis", in: "Large-scale numerical optimization",

	Thomas F. Coleman and Yuying Li, editors, SIAM Publications, Philadelphia 1990, pp 22-30. 4. K. P. Bennett & O. L. Mangasarian: "Robust linear programming discrimination of two linearly inseparable sets", Optimization Methods and Software 1, 1992, 23-34 (Gordon & Breach Science Publishers).
Title	Wisconsin Breast Cancer Database (January 8, 1991)
Sources	Dr. WIlliam H. Wolberg (physician) University of Wisconsin Hospitals Madison, Wisconsin USA Donor: Olvi Mangasarian (mangasarian@cs.wisc.edu) Received by David W. Aha (aha@cs.jhu.edu) Date: 15 July 1992
Past Usage	Attributes 2 through 10 have been used to represent instances. Each instance has one of 2 possible classes: benign or malignant. 1. Wolberg, W. H. and Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to

	breast cytology. In *Proceedings of the National Academy of Sciences*, 87: 9193—9196. • Size of data set: only 369 instances (at that point in time) • Collected classification results: 1 trial only • Two pairs of parallel hyperplanes were found to be consistent with 50% of the data • Accuracy on remaining 50% of dataset: 93.5% • Three pairs of parallel hyperplanes were found to be consistent with 67% of data • Accuracy on remaining 33% of dataset: 95.9% 2. Zhang, J. (1992). Selecting typical instances in instance-based learning. In *Proceedings of the Ninth International Machine Learning Conference* (pp. 470--479). Aberdeen, Scotland: Morgan Kaufmann. • Size of data set: only 369 instances (at that point in time) • Applied 4 instance-based learning algorithms • Collected classification results averaged over 10 trials • Best accuracy result: 1. 1-nearest neighbor: 93.7% 2. Trained on 200 instances, tested on the other 169

	• Also of interest: 1. Using only typical instances: 92.2% (storing only 23.1 instances) 2. Trained on 200 instances, tested on the other 169
Relevant Information	Samples arrive periodically as Dr. Wolberg reports his clinical cases. The database therefore reflects this chronological grouping of the data. This grouping information appears immediately below, having been removed from the data itself: Group 1: 367 instances (January 1989) Group 2: 70 instances (October 1989) Group 3: 31 instances (February 1990) Group 4: 17 instances (April 1990) Group 5: 48 instances (August 1990) Group 6: 49 instances (Updated January 1991) Group 7: 31 instances (June 1991) Group 8: 86 instances (November 1991) Total: 699 vectors (as of the donated database on 15 July 1992) Note that the results summarized above in Past Usage refer to a

	dataset of size 369, while Group 1 has only 367 instances. This is because it originally contained 369 instances; 2 were removed. The following statements summarizes changes to the original Group 1's set of data: ##### Group 1 : 367 vectors: 200B 167M (January 1989) ##### Revised Jan 10, 1991: Replaced zero bare nuclei in 1080185 & 1187805 ##### Revised Nov 22,1991: Removed 765878,4,5,9,7,10,10,10,3,8,1 no record ##### Removed 484201,2,7,8,8,4,3,10,3,4,1 zero epithelial ##### Changed 0 to 1 in field 6 of sample 1219406 ##### Changed 0 to 1 in field 8 of following sample: ##### 1182404,2,3,1,1,1,2,0,1,1,1
Size of training set	699 labeled training vectors (as of 15 July 1992)
Number of features	10
Number of training vectors with missing features	16 Comment: There are 16 instances in Groups 1 to 6 that contain a single missing (i.e., unavailable) attribute value, now denoted by

	"?".

Table 5: List of features of the Wisconsin breast cancer dataset

#	**Feature description**	**Domain**	**Comment**
1	Sample code number	Id number	This field was irrelevant and was taken out.
2	Clump Thickness	1 – 10	
3	Uniformity of Cell Size	1 – 10	
4	Uniformity of Cell Shape	1 – 10	
5	Marginal Adhesion	1 – 10	
6	Single Epithelial Cell Size	1 – 10	
7	Bare Nuclei	1 – 10	
8	Bland Chromatin	1 – 10	
9	Normal Nucleoli	1 – 10	
10	Mitoses	1 – 10	

Table 6: List of labels of the Wisconsin breast cancer dataset

Label Value	**Label description**	**# of instances with label**
2	Benign	458 (65.5%)

4	Malignant	241 (34.5%)

Based on the above output model, two more experiments were done for this appendix on the Wisconsin breast cancer dataset. The first experiment extracts the α weights of all the support vectors and produces a bar plot of the magnitudes of the weights. The resulting bar plots are shown in Figure 39 and Figure 40. The second experiment performs a data visualization, much like the data visualizations shown in the main thesis, on the set of support vectors. The result of this data visualization is shown in .

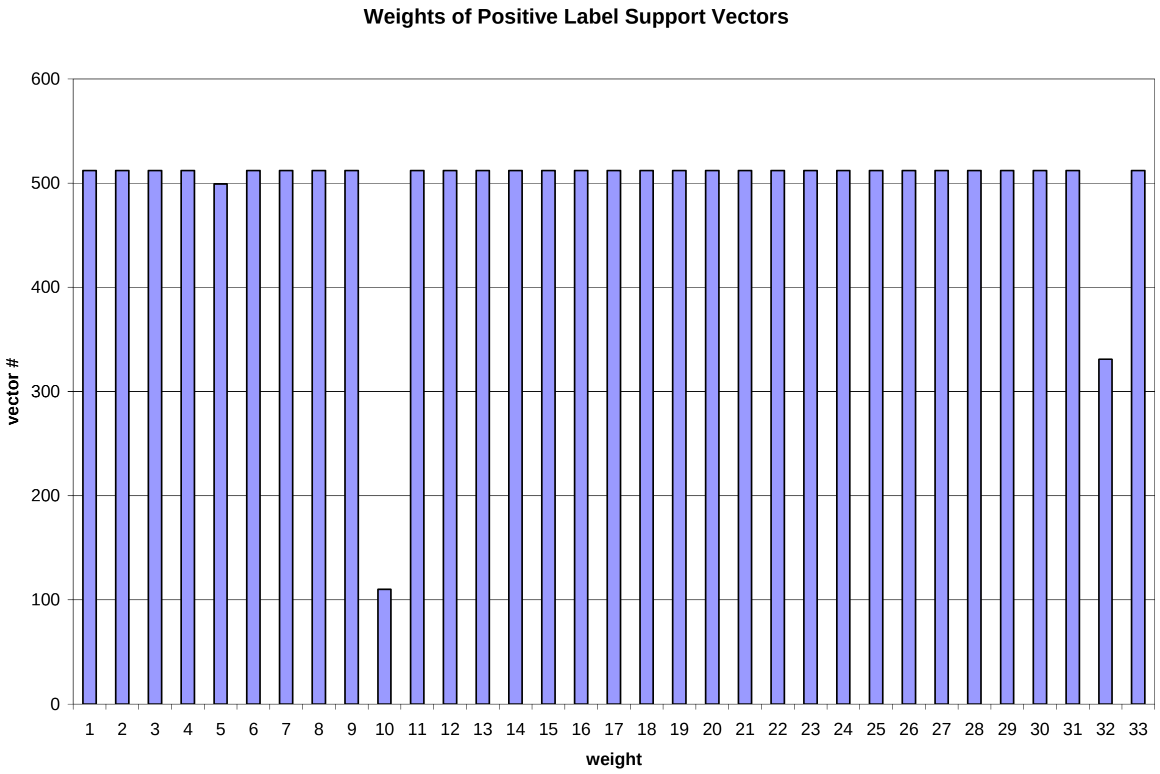

Figure 39: Weights of positive label support vectors for the Wisconsin breast cancer dataset.

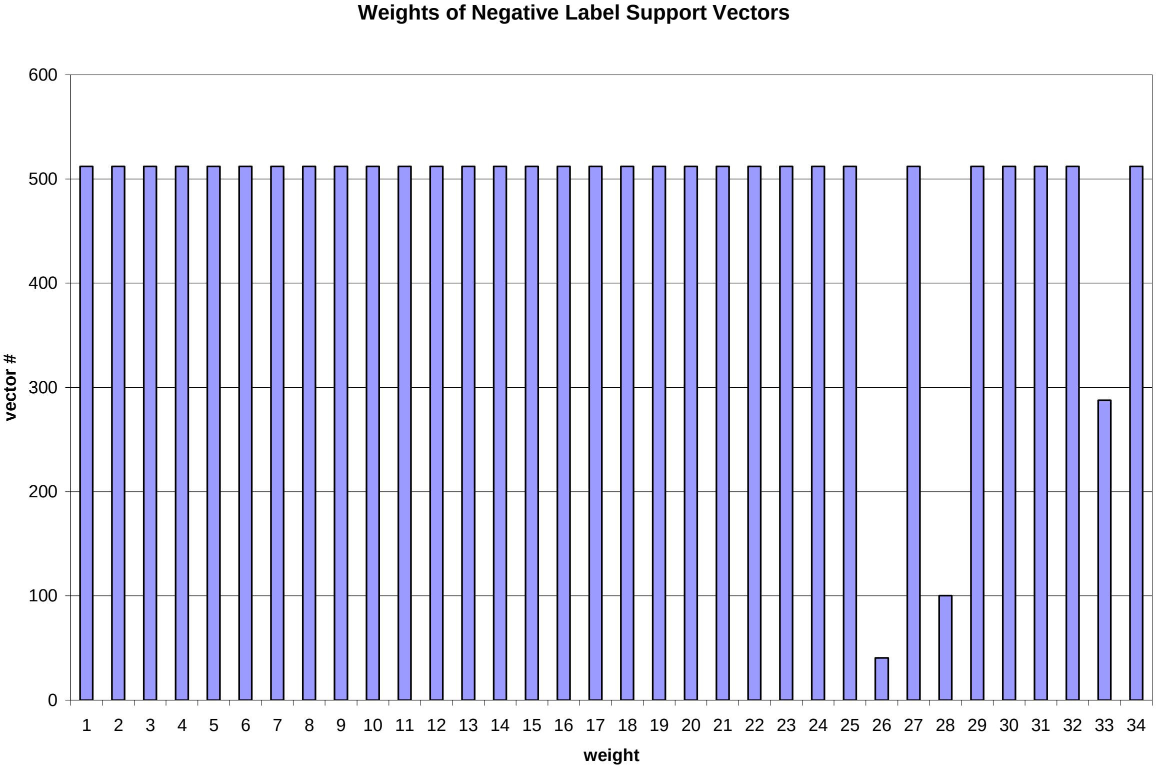

Figure 40: Weights of negative label support vectors for the Wisconsin breast cancer dataset.

As we can see from Figure 39 and Figure 40, the weights of all support vectors of both label types are fairly consistent with the exception of a few support vectors (support vectors 5, 10 and 32 of the positive set of support vectors, and support vectors 26, 28 and 33 of the negative set of support vectors) which are not weighed as heavily as the other weight vectors. With Figure 39 and Figure 40, a biologist may notice the vectors with the smaller weights and isolate them into another file and do something interesting with them. This is because in machine learning any vectors with a weight that deviates significantly from other weights may hold further information relating to the underlying distribution of the Wisconsin breast cancer biological problem at hand.

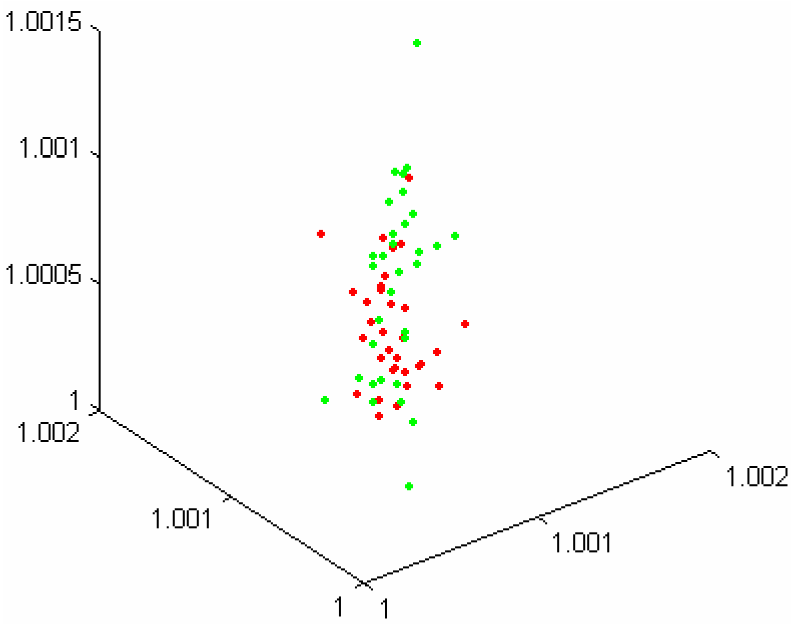

Figure 41: Data visualization of the support vectors of the Wisconsin breast cancer dataset.

This is essentially **Figure 26** with all non-support vectors taken out. The clarity of this data visualization, compared to the one seen in figure **Figure 38**, owes a great deal to the small number of support vectors that are needed to form the decision hyperplane. The green support vectors indicating patients with benign outgrowth and the red support vectors indicating patients with malignant outgrowth are still seen to fall roughly on two planes. But the vectors are not as tightly clustered as they were in **Figure 26** since we only have the support vectors this time. With the added extra clarity of **Figure 41** we may actually see that some of the green support vectors do fall in front of the red support vectors at this time. Finally, as noted in Figure 25 and **Figure 26**, the green vectors representing benign outgrowth are more clustered than the red vectors representing malevolent outgrowth, this might indicate that benign outgrowths have more in common with each other than malignant outgrowth. It also indicates that malignant outgrowths, which represent the onset of true instances breast cancer, might be more difficult to diagnose than benign outgrowths due to their more varied characteristics that cause the red vectors to spread out more.

Future work

During the question and answer phase of this thesis the author responded to Dr. McConkey's question about the transparency of support vector machines with the answer that weights will be given at the end of the training phase that will delineate the importance of the features with respect to one another. Upon close examination of the output model file it was realized that the weights are *not* assigned to the input space features, but rather to each of the training vectors. In the future it would be helpful to find a way of training the vectors under the power landmark vector learning framework such that weights underlining the importance of each feature/attribute in the input file could be given at the end of the training phase. Having this information will greatly aid the biologist in weighing the relative importance of each feature in order to decide how the list of features should be narrow down so that the biologist can concentrate on those features that play the most important roles in determining the biological label of the training and testing set.

Bibliography

Aizerman M., Braverman E. and Rozonoer L. (1964) Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821-837.

Aronszajn N. (1950) Theory of Reproducing Kernels. *Transactions of the American Mathematical Society*, 68(3):337-404.

Balcan M. F., Blum A., and Vempala S. S. (2004) Kernels as features: On kernels, margins, and low-dimensional mappings. In *15th International Conference on Algorithmic Learning Theory (ALT '04)*, pages 194-205. An extended version is available at http://www.cs.cmu.edu/~avrim/Papers/.

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988), The New S Language, Wadsworth & Brooks/Cole.

Blum A. (2006) Random Projection, Margins, Kernels, and Feature-Selection. LNCS 3940, pp. 52-68. URL http://www.cs.cmu.edu/~avrim/Papers/randomproj.pdf

Balcan M.F. and Blum A. (2005) A PAC-style model for learning from labeled and unlabeled data. In *Proceedings of the 18th Annual Conference on Computational Learning Theory* (*COLT*), pages 111-126.

Bennett, K. P. and O. L. Mangasarian (1992): "Robust linear programming discrimination of two linearly inseparable sets", Optimization Methods and Software 1, 1992, 23-34 (Gordon & Breach Science Publishers).

Boser B. E., Guyon I. M. and Vapnik V. N. (1992) A training algorithm for optional margin classifiers. In D. Haussler, editor, *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory (COLT)*, ACM Press. Pp. 144-152.

Chang C.-C. and Lin C.-J. (2001) LIBSVM : a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

Cover T. M. (1965) Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Trans. Elect. Comp.,* 14:326-334.

Dasgupta S. and Gupta A. (1999) An elementary proof of the Johnson-Lindenstrauss Lemma. *International Computer Science Institute, Technical Report TR-99-006*, UC Berkeley.

Duda R. O., Hart P. E., and Stork D. G. (2000) Pattern Classification. Wiley-interscience, 2nd ed.

Feller W. (1968) An Introduction to Probability Theory and Its Applications, Volumes I & II. New York, NY: John Wiley.

Fisher R. A. (1986) The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2):179-188.

Frankl P. and Maehara H. (1988) The Johnson-Lindenstrauss lemma and the sphericity of some graphs. *J. Combin. Theory Ser. B* 44(3):355-362.

Gardy, J. L., Spencer C., Wang K., Ester M., Tusnady G. E., Simon I., Hua S., deFays K., Lambert C., Nakai K., and Brinkman F. S. (2003). PSORT-B: improving protein subcellular localization prediction for gram-negative bacteria. *Nucleic Acids Research* 31(13): 3613–3617.

Hsu C. W., Chang C.-C. and Lin, C.-J. (2007). A Practical Guide to Support Vector Classification. URL http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

Iba,W., Wogulis,J., & Langley,P. (1988). Trading off Simplicity and Coverage in Incremental Concept Learning. In Proceedings of the 5th International Conference on Machine Learning, 73-79. Ann Arbor, Michigan: Morgan Kaufmann.

Indyk P. and Motwani R. (1998) Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. *Proc. 30th Symposium on Theory of Computing*, pp. 604-613.

Johnson W. and Lindenstrauss J. (1984) Extensions of Lipschitz maps into a Hilbert space. *Contemp. Math.* 26:189-206.

Kaku M. (2007) Visions of the future 2: the biotechnological revolution. BBC 4 documentary.

Kwok J. and Tsang I. (2003) The pre-image problem in kernel methods. *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*. Washington, D.C., USA. Pp 408-415

Lincoff G. H. (1981) The Audubon Society Field Guide to North American Mushrooms. New York: Alfred A. Knopf

Mangasarian O. L. and W. H. Wolberg (1990): "Cancer diagnosis via linear programming", SIAM News, Volume 23, Number 5, September 1990, pp 1 & 18.

Mangasarian, O. L., R. Setiono, and W.H. Wolberg (1990): "Pattern recognition via linear programming: Theory and application to medical diagnosis", in: "Large-scale numerical optimization", Thomas F. Coleman and Yuying Li, editors, SIAM Publications, Philadelphia 1990, pp 22-30.

Mercer J. (1909) Functions of positive and negative type and their connection with the theory of integral equations. *Philosophical Transactions of the Royal Society of London*, Series A 209:416-446.

Mika S., Schoelköpf B., Smola A., Müller K. R., Schölz M. and Ratsch G. (1999). Kernel PCA and de-noising in feature spaces. In M. S. Kearns, S. A. Solla and D. A. Cohn (Eds.), *Advances in neural information processing systems* 11:536-542. Cambridge, MA: MIT Press.

Minsky M. and Papert S. (1969) Perceptrons: An Introduction to Computational Geometry. MIT Press.

Nicholson W. K. (2002) Linear Algebra with Applications, 5th edition. McGraw-Hill Publishing Co.

Rathi A. (2005) CS 395T Computational Learning Theory, Lecture 9: September 27, 2005. University of Texas Department of Computer Science. http://www.cs.utexas.edu/~klivans/f06lec2.pdf

Rosenblatt F. (1958) The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386-408.

Saitoh S. (1988) Theory of Reproducing Kernels and its Applications. Longman Scientific & Technical.

Schlimmer,J.S. (1987). Concept Acquisition Through Representational Adjustment (Technical Report 87-19). Doctoral disseration, Department of Information and Computer Science, University of California, Irvine.

Shawe-Taylor J. and Cristianini N. (2004) Kernel Methods for Pattern Analysis. Cambridge, UK: Cambridge University Press.

Smith F. W. (1968) Pattern classifier design by linear programming. *IEEE Transactions on Computers,* C-17:367-372.

Smith, J. W., Everhart, J. E., Dickson, W. C., Knowler, W. C., & Johannes, R. S. (1988). Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the Symposium on Computer Applications and Medical Care* (pp. 261--265). IEEE Computer Society Press.

Vapnik V. and Chervonenkis A. (1964) A note on one class of perceptrons. *Automation and Remote Control*, 25.

Vapnik V. and Lerner A. (1963) Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24.

Vempala, S. S. (2004) The random projection method. In *DIMACS series in discrete mathematics and theoretical computer science, ISSN 1052-1798,* Volume 65. American Mathematical Society.

J. Craig Venter,[1*] Mark D. Adams,[1] Eugene W. Myers,[1] Peter W. Li,[1] Richard J. Mural,[1] Granger G. Sutton,[1] Hamilton O. Smith,[1] Mark Yandell,[1] Cheryl A. Evans,[1] Robert A. Holt,[1] Jeannine D. Gocayne,[1] Peter Amanatides,[1] Richard M. Ballew,[1] Daniel H. Huson,[1] Jennifer Russo Wortman,[1] Qing Zhang,[1] Chinnappa D. Kodira,[1] Xiangqun H. Zheng,[1] Lin Chen,[1] Marian Skupski,[1] Gangadharan Subramanian,[1] Paul D. Thomas,[1] Jinghui Zhang,[1] George L. Gabor Miklos,[2] Catherine Nelson,[3] Samuel Broder,[1] Andrew G. Clark,[4] Joe Nadeau,[5] Victor A. McKusick,[6] Norton Zinder,[7] Arnold J. Levine,[7] Richard J. Roberts,[8] Mel Simon,[9] Carolyn Slayman,[10] Michael Hunkapiller,[11] Randall Bolanos,[1] Arthur Delcher,[1] Ian Dew,[1] Daniel Fasulo,[1] Michael Flanigan,[1] Liliana Florea,[1] Aaron Halpern,[1] Sridhar Hannenhalli,[1] Saul Kravitz,[1] Samuel Levy,[1] Clark Mobarry,[1] Knut Reinert,[1] Karin Remington,[1] Jane Abu-Threideh,[1] Ellen Beasley,[1] Kendra Biddick,[1] Vivien Bonazzi,[1] Rhonda Brandon,[1] Michele Cargill,[1] Ishwar Chandramouliswaran,[1] Rosane Charlab,[1]

Kabir Chaturvedi,[1] Zuoming Deng,[1] Valentina Di Francesco,[1] Patrick Dunn,[1] Karen Eilbeck,[1] Carlos Evangelista,[1] Andrei E. Gabrielian,[1] Weiniu Gan,[1] Wangmao Ge,[1] Fangcheng Gong,[1] Zhiping Gu,[1] Ping Guan,[1] Thomas J. Heiman,[1] Maureen E. Higgins,[1] Rui-Ru Ji,[1] Zhaoxi Ke,[1] Karen A. Ketchum,[1] Zhongwu Lai,[1] Yiding Lei,[1] Zhenya Li,[1] Jiayin Li,[1] Yong Liang,[1] Xiaoying Lin,[1] Fu Lu,[1] Gennady V. Merkulov,[1] Natalia Milshina,[1] Helen M. Moore,[1] Ashwinikumar K Naik,[1] Vaibhav A. Narayan,[1] Beena Neelam,[1] Deborah Nusskern,[1] Douglas B. Rusch,[1] Steven Salzberg,[12] Wei Shao,[1] Bixiong Shue,[1] Jingtao Sun,[1] Zhen Yuan Wang,[1] Aihui Wang,[1] Xin Wang,[1] Jian Wang,[1] Ming-Hui Wei,[1] Ron Wides,[13] Chunlin Xiao,[1] Chunhua Yan,[1] Alison Yao,[1] Jane Ye,[1] Ming Zhan,[1] Weiqing Zhang,[1] Hongyu Zhang,[1] Qi Zhao,[1] Liansheng Zheng,[1] Fei Zhong,[1] Wenyan Zhong,[1] Shiaoping C. Zhu,[1] Shaying Zhao,[12] Dennis Gilbert,[1] Suzanna Baumhueter,[1] Gene Spier,[1] Christine Carter,[1] Anibal Cravchik,[1] Trevor Woodage,[1] Feroze Ali,[1] Huijin An,[1] Aderonke Awe,[1] Danita Baldwin,[1] Holly Baden,[1] Mary Barnstead,[1] Ian Barrow,[1] Karen Beeson,[1] Dana Busam,[1] Amy Carver,[1] Angela Center,[1] Ming Lai Cheng,[1] Liz Curry,[1] Steve Danaher,[1] Lionel Davenport,[1] Raymond Desilets,[1] Susanne Dietz,[1] Kristina Dodson,[1] Lisa Doup,[1] Steven Ferriera,[1] Neha Garg,[1] Andres Gluecksmann,[1] Brit Hart,[1] Jason Haynes,[1] Charles Haynes,[1] Cheryl Heiner,[1] Suzanne Hladun,[1] Damon Hostin,[1] Jarrett Houck,[1] Timothy Howland,[1] Chinyere Ibegwam,[1] Jeffery Johnson,[1] Francis Kalush,[1] Lesley Kline,[1] Shashi Koduru,[1] Amy Love,[1] Felecia Mann,[1] David May,[1] Steven McCawley,[1] Tina McIntosh,[1] Ivy McMullen,[1] Mee Moy,[1] Linda Moy,[1] Brian Murphy,[1] Keith Nelson,[1] Cynthia Pfannkoch,[1] Eric Pratts,[1] Vinita Puri,[1] Hina Qureshi,[1] Matthew Reardon,[1] Robert Rodriguez,[1] Yu-Hui Rogers,[1] Deanna Romblad,[1] Bob Ruhfel,[1] Richard Scott,[1] Cynthia Sitter,[1] Michelle Smallwood,[1] Erin Stewart,[1] Renee Strong,[1] Ellen Suh,[1] Reginald Thomas,[1] Ni Ni Tint,[1] Sukyee Tse,[1] Claire Vech,[1] Gary Wang,[1] Jeremy Wetter,[1] Sherita Williams,[1] Monica Williams,[1] Sandra Windsor,[1] Emily Winn-Deen,[1] Keriellen Wolfe,[1] Jayshree Zaveri,[1] Karena Zaveri,[1] Josep F. Abril,[14] Roderic Guigó,[14] Michael J. Campbell,[1] Kimmen V. Sjolander,[1] Brian Karlak,[1] Anish Kejariwal,[1] Huaiyu Mi,[1] Betty Lazareva,[1] Thomas Hatton,[1] Apurva Narechania,[1] Karen Diemer,[1] Anushya Muruganujan,[1] Nan Guo,[1] Shinji Sato,[1] Vineet Bafna,[1] Sorin Istrail,[1] Ross Lippert,[1]

Russell Schwartz,[1] Brian Walenz,[1] Shibu Yooseph,[1] David Allen,[1] Anand Basu,[1] James Baxendale,[1] Louis Blick,[1] Marcelo Caminha,[1] John Carnes-Stine,[1] Parris Caulk,[1] Yen-Hui Chiang,[1] My Coyne,[1] Carl Dahlke,[1] Anne Deslattes Mays,[1] Maria Dombroski,[1] Michael Donnelly,[1] Dale Ely,[1] Shiva Esparham,[1] Carl Fosler,[1] Harold Gire,[1] Stephen Glanowski,[1] Kenneth Glasser,[1] Anna Glodek,[1] Mark Gorokhov,[1] Ken Graham,[1] Barry Gropman,[1] Michael Harris,[1] Jeremy Heil,[1] Scott Henderson,[1] Jeffrey Hoover,[1] Donald Jennings,[1] Catherine Jordan,[1] James Jordan,[1] John Kasha,[1] Leonid Kagan,[1] Cheryl Kraft,[1] Alexander Levitsky,[1] Mark Lewis,[1] Xiangjun Liu,[1] John Lopez,[1] Daniel Ma,[1] William Majoros,[1] Joe McDaniel,[1] Sean Murphy,[1] Matthew Newman,[1] Trung Nguyen,[1] Ngoc Nguyen,[1] Marc Nodell,[1] Sue Pan,[1] Jim Peck,[1] Marshall Peterson,[1] William Rowe,[1] Robert Sanders,[1] John Scott,[1] Michael Simpson,[1] Thomas Smith,[1] Arlan Sprague,[1] Timothy Stockwell,[1] Russell Turner,[1] Eli Venter,[1] Mei Wang,[1] Meiyuan Wen,[1] David Wu,[1] Mitchell Wu,[1] Ashley Xia,[1] Ali Zandieh,[1] Xiaohong Zhu[1], (2001) The Sequence of the Human Genome. *Science* 291(5507): 1304 – 1351. DOI: 10.1126/science.1058040.

Wessa P., (2007) Random Number Generator for the Normal Distribution (v1.0.4) in Free Statistics Software (v1.1.22-r4), Office for Research, Development and Education, URL http://www.wessa.net/rwasp_rngnorm.wasp/

Wichura, M. J. (1988) Algorithm AS 241: The Percentage Points of the Normal Distribution, Applied Statistics, nr. 37, 477-484.

Wolberg, William H. and O.L. Mangasarian (1990): "Multisurface method of pattern separation for medical diagnosis applied to breast cytology", Proceedings of the National Academy of Sciences, U.S.A., Volume 87, December 1990, pp 9193-9196.