Chapter 4 Investigation of Recombination Rates Using SVMs

4.1 Introduction

In Chapter 3 attempts were made to divide chromosomes up into blocks with uniform but distinct properties using HMMs. The justification for this was the observation that certain biological processes appear to segregate with such patterns. Another way to look at chromosomes is to consider if there are associated properties that are continuous in nature. One property that appears to have this behaviour is the probability of recombination occurring between any two bases on the sequence.

Recombination events are responsible for the inheritance of a unique, mosaic combination of alleles during sexual reproduction. In humans it has become clear that the single nucleotide polymorphisms observed are grouped into regions bounded by points of recombination, which have recently been mapped for Chromosome 22 (Dawson, Abecasis et al. 2002). How much variation in recombination rates influences the inheritance pattern in organisms including man has to date not been quantified.

The exact mechanisms that drive differences in recombination rate are unknown. It has been shown in some organisms that some recombination hot spots can be directly controlled by very small regions of a chromosome, and that the trait of recombination rate is heritable (Dixon and Kowalczykowski 1991). It is always possible that the heritable component is something other than the genome sequence, such as the methylation pattern. The objective of this chapter is therefore to look for a sequence based signal.

Human Chromosome 22 provides the source of data for this investigation. The rate of recombination has been estimated along a large portion of the q-arm of human chromosome 22 between some 35 genetic markers (Dib, Faure et al. 1996; Dunham, Shimizu et al. 1999). With the advent of a finished sequence for this chromosome, it is possible to compare the genetic and physical distances. As indicated by the blue line in Figure 4-1, the recombination rate is not uniform across the region. By plotting physical position along the x-axis and the ratio of genetic to physical distance on the y-axis, the non-linearity shows up clearly as spikes. There may be recombination rate enhancing and repressing signals within the chromosome that are causing this position-dependant difference in recombination rate.

The process of learning how to predict recombination rate from sequence content can be addressed by a supervised learning approach. The dimensionality of the data is extremely high if we consider all possible sub-sequences within a region of interest. One methodology which has been used successfully for very high dimensional data is the support vector machine (SVM) which is now described in detail.

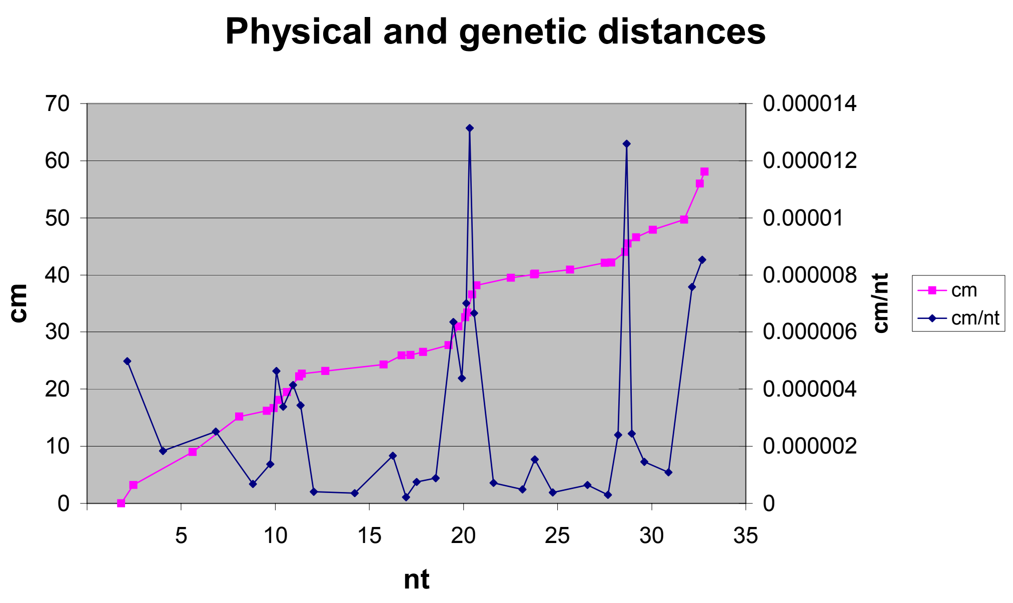

Figure 4-1 Comparison of physical and genetic distances along chromosome 22

The x-axis represents the genetic distance of each marker on the q-arm of chromosome 22 from the centromere, as measured in megabases. The pink line is the distance in centi-morgans of each marker from the first marker. The blue line displays a data-point between each consecutive pair of markers, with a height proportional to the ratio between the difference in the genetic distance between the two markers and their physical distance. Peaks in the blue graph indicate regions of relatively high recombination.

4.1.1 Support Vector Machines

Support vector machines (SVM) (Vapnik 1995) are linear models that use dot products and kernel functions to perform classification and regression tasks (see Section 1.3.1). They are designed to estimate an affine transform from an arbitrary dimensional input space to a single dimensional output space. This output space is defined as the distance of a data item from some plane in input space. The distance of a point from a plane can be computed as the dot-product between the point and the normal of the plane, plus the plane's constant (the smallest distance of the plane from the origin). All points lying within a plane satisfy the equation:

Equation 4-1 Equation of a Plane

$$v \cdot x - h = 0$$
$$v = \text{normal to the plane; } x = \text{any point in plane; h} = \text{plane offset}$$

Dot-products of sums can be re-written as sums of dot-products:

Equation 4-2 Normal to a Plane as a Weighted Sum of Vectors

$$\left(\sum_i x_i\right) \cdot x = \sum_i \left(x_i \cdot x\right)$$

It follows that we can represent the normal of the plane by a weighted sum of the training examples. Although most of the problems encountered do not have a good linear solution in the data-space, there is often a linear solution in some alternate 'feature space' that is equivalent to a non-linear solution in the data-space. For example, the space of all polynomial interactions of order two or less between the two components of a vector describes all conics in the data-space. If there is a kernel function that computes dot-products in this feature space, Equation 4-2 shows that the equation of the plane can be implicitly represented as a weighted sum of the kernel functions acting upon each data point and the data-item x. The full equation becomes Equation 4-3 where β_i is the weight of the i th training example in the plane normal.

Equation 4-3 Definition of a Support Vector Machine

$$f(x) = \sum_i \beta_i k(x_i, x) - h$$

This formulation has the interesting property that although the solution is in a potentially large feature space, there is exactly one more parameters than there are training examples. The standard SVM training algorithms introduce a further constraint upon the parameters such that one of them is not free. This means that regardless of the resulting model, it can contain no more information than the original training data.

When using a SVM, it is usual to vary x for some fixed range of values for x_i and β_i. This can be made more explicit by rewriting the kernel function term as a basis function:

Equation 4-4 Basis Functions for Kernel Functions and Data Points

$$\phi_i(x) = k(x_i, x)$$

This allows the SVM equation to be re-written as follows.

Equation 4-5 SVMs in Terms of Basis Functions

$$f(x) = \sum_i \beta_i \phi_i(x) - h$$

If any of the weights β_i are zero, then the associated basis function does not affect the result. If only a few of the weights are non-zero then the resulting function is relatively simple. This property is called sparsity and the data-points associated with the contributing basis functions are called support vectors, as for separable problems, when displayed graphically in the feature-space (and if the feature space is related by a continuous function to the data-space, in that also), they 'support' the learned hyperplane (or decision boundary in the data-space). For the special case of pair-wise classification, the support vectors are the data points that lie on the boundaries of the convex hulls of the two sets of data in feature space.

There are several methods available to estimate the parameters of the hyperplane (β, h) given an error function. We found the most efficient method currently available to be the SMO algorithm (Platt 1998). This method optimises for pairs of support vectors at a time, and eventually this leads to the complete solution being found. For problems with appreciable sparsity, this method takes time approximately proportional to the training set size. Once trained, SVMs are computationally very cheap to apply to new data items, assuming that the kernel function is easy to compute.

4.1.2 BioJava APIs for Support Vector Machines

Support for SVMs is provided by BioJava. Their implementation builds upon the formulism discussed in Section 1.3.1 by providing a Java interface called `KernelFunction` that computes the kernel function for any two Java objects. There are now a number of other publicly available SVM implementations, such as SVM-fu[32]. However, unlike many of the other implementations, BioJava allows arbitrary kernel functions to be used. One of the benefits of this flexibility is that we can manipulate the data inside the kernels, for example normalizing vectors onto a unit sphere or scaling some subspace. We have also observed that for complex kernel functions, the performance of the BioJava implementation does not degrade.

4.2 Methods

4.2.1 Searching for a Signal Affecting Recombination Rates Using a Word-Frequency Kernel Function

Under the assumption that there are sequences within chromosomal DNA that affect recombination rate, and given example sequences known to have high or low rates, it should be possible to discover some metric of the sequence which is predictive of its recombination rate. It is possible that recombination rate is mediated by very simple sequences (e.g. poly-A or poly-GC), or relies upon very complex patterns (e.g. entire promoters or histone-binding regions). Either way, it is likely that part of the signal will correlate with scores collected by counting word frequencies (such as the frequencies of all octamers).

Once the sequence data is transformed into a format that is equivalent to counts over a finite set of properties, it becomes suitable data for processing with a Support Vector Machine using a simple kernel function. The transform from sequence to counts can be considered to be equivalent to a data-to-feature space transform, inducing a new kernel function that both projects sequences to counts and then calculates the dot-product of the counts.

[32] SVM-fu is distributed through the http://www.ai.mit.edu/projects/cbcl/ web site

In previous studies with word-count based kernel functions, greater accuracy has been achieved by normalizing the counts prior to calculating the kernel function as it is usually the relative proportion of the different words and not the absolute count that is informative, and normalization both controls for document size and numerical instabilities introduced by large differences in size between the magnitudes of training data (for example, see example training data accompanying svm-light[33]). However, this normalization can actually be performed within the kernel itself (Equation 4-6) as the process of normalizing each input vector is itself a transform from some data-space to a feature space (projection of all points onto a unit hyper-sphere).

In the cases when the data-space is very large and the cost of normalizing this is prohibitive but the un-normalized kernel is cheap to compute, the normalizing kernel saves both space and time. It potentially makes reading the computer code easier as the entire data-to-feature transform is represented in one place, the kernel, rather than being spread amongst multiple pre-processing steps. If repeatedly calculating the terms $\langle a,a \rangle$ and $\langle b,b \rangle$ is found to be expensive then these values can be cached. By applying an object-oriented design methodology, we can implement a kernel that delegates to an underlying kernel function for all values not known and caches the results for all terms of the form $x \cdot x$ for quick access. This is the approach taken in the BioJava toolkit, and we have found that it drastically decrease the computational load of kernel functions such as the normalizing and radial basis kernels that require some values to be repeatedly calculated.

[33] See http://svmlight.joachims.org/ for an example of using normalised counts for classification

Equation 4-6 The Normalizing Kernel

$$k_{\langle\cdot,\cdot\rangle}^{norm}(a,b)=\langle\hat{a},\hat{b}\rangle=\frac{\langle a,b\rangle}{\sqrt{\langle a,a\rangle\langle b,b\rangle}}$$

The family of kernel functions used in this study can be represented as:

Equation 4-7 SuffixTree Kernel

$$k_{k_{df}^{SuffixTree}}^{norm}(a,b) \text{ where}$$
$$k_{df(\cdot)}^{SuffixTree}(a,b)=\sum_{d\in(0..l)} df(d)\cdot\sum_{i\in\Omega^d} a_i\cdot b_i;$$
df = depth function (scales tree counts by depth);
suffix trees are indexable by string i

Because of how the suffix trees are constructed, they will not contain nodes for zero counts. By definition, if a given sub-string is absent from the entire sequence, then all other sub-strings containing that string will also be absent. For this reason, the nodes of the tree are sparsely populated (only nodes that store non-zero counts are instantiated). Thus, the index i in Equation 4-7 need actually only loop over values of Ω^d that are populated in both a and b.

The depth function term $df(\cdot)$ allows the counts associated with strings of a particular length to be given greater or lesser weight. For example, a depth function that always returns 1 will leave the counts un-scaled (uniform depth function). A depth function that returns non-zero for one value and zero for all others will have the effect of only including words of a single length. A depth function of the form $\left|\Omega^d\right|$ will make longer matches more significant than shorter ones, taking into account the fact that they are less likely by chance (normalizing depth function).

In principal, the depth functions are a subset of functions that return a scale factor for each i that is purely a function of its length. In cases when the sequence bias is significantly divergent from uniform, it may be worth re-defining Equation 4-7 in terms of a per-i scaling function instead of a depth function. However, we considered that in this case any increased accuracy obtained would be off-set by the need to optimize the scaling function, and any associated computational overhead.

4.2.2 Construction and Training of an SVM for Predicting Recombination Rate

The SVM used was constructed with the normalized suffix-tree kernel as described above. Both the uniform and normalizing depth functions were evaluated. The maximum tree depths were fixed from 1 to 9 for the uniform model and 1 to 8 for the normalized model. The models were trained using the SMO method for classifiers as at the time the BioJava implementation of SVM regression that was not numerically stable.

All clones in the partially finished Human chromosome 22 were extracted, together with their approximate coordinates within the chromosome. These were fully repeat-masked for both simple and complex repeats using repeat masker[34]. The chromosomal locations of all 35 markers were used. The high-recombination rate region within approximately 18-21Mb and the low recombination rate region within approximately 21-17Mb were used as the positive and negative training sets respectively. All clones from these regions were used for training. All clones outside of this region were included during the prediction phase.

[34] se here for an online reference for repeatmasker, currently unpublished:

http://ftp.genome.washington.edu/RM/RepeatMasker.html

4.3 Results

4.3.1 Recombination Rates Predictions

In no cases could any of the models with a depth of less than four be trained. This indicates that the information necessary to predict recombination rates relies upon sequences of at least four in length. To bin sequences into two categories should only have required one variable, or the ratio of two variables, so the tree depth of 1 or 2 should have been sufficient to trivially separate them if the signal was purely based upon low-order sequence bias (such as AT/GC ratio).

For the models with maximum depth 4 and upward, the SVM produced an output centred on 0.0, indicating that the model is not consistently predicting items as being positive or negative. During training, the procedure attempts to predict a function such that all negative examples have a value less than –1.0, and all of the positive examples have a value of greater than +1.0. All items that are within the range –1.0 and +1.0 will be support vectors. If unseen data has an output between –1.0 and +1.0, this indicates that the SVM considers this data-point ambiguous, but it still attempts a classification that can be read by looking at the sign of the output. Values of magnitude larger than 1.0 indicate that the SVM was confident in its assignment.

The models trained using the uniform counts (Figure 4-2, Figure 4-3, Figure 4-4) learned functions that model the training data well, giving outputs around +1 for the high recombination regions and -1 for the low recombination regions. The higher depth models (6-9) produce SVM outputs that are noticeably closer to zero and less spread than the lower depth models. However, the general shapes of these curves (as judged by the 10 point moving average) are very similar. Arguably, the output shadows the recombination curve, particularly around the peak at 10Mb, and the dip at 30Mb. However, it also predicts a recombination-poor region at 5Mb. The 5Mb feature may be biologically significant but not visible in the recombination plot due to marker density, or may be an artefact.

The models trained using length-normalized counts, Figure 4-5 Figure 4-6, Figure 4-7) are less prone to wild fluctuations (compare the scattering visible in Figure 4-2 and Figure 4-5), which may indicate that the predictions are more robust. In addition, the dynamics of the predicted recombination frequencies are more similar to the actual rates (Figure 4-6, Figure 4-7). Again, as the depth of the suffix-tree is increased, the resulting function becomes smoother, closer to zero and fluctuates less wildly.

4.3.2 Cross-Validation

To assess how robust the predictions of the SVMs are with respect to the training data, the sequences within the positive and negative training sets were partitioned into three sets randomly. Three models were trained, one for each partition, and then tested by predicting the membership of the other two partitions. This 3-way jack-knifing was performed for all depths in the normalized models using training methods which where otherwise identical to those used above. The best accuracy of 80 % (random 50 %) is achieved for a depth of 5, with accuracy becoming worse for greater depths.

The resulting models appeared to be memorizations of the training data, with very few example sequences not included as support vectors (between 1 and 4 training examples left out). They do seem to be consistent among one another (depth of 5 being the most reproducible), with prediction accuracies that are consistent, both inside the training data (Figure 4-8), and across the entire chromosome (Figure 4-9).

However, over all, the chromosomal predictions are less informative of recombination rate. This is as to be expected with less training data.

Error! Not a valid link.

Figure 4-2 Total Results of Training the SVM using Uniform Counts

SVM predictions are displayed as points on the scatter graph. For comparison, the recombination rate is also displayed.

Error! Not a valid link.

Figure 4-3 Moving Average for Uniform Counts models of Depth 4-6

10 point moving averages of the SVM predictions are displayed for models with depths of four, five and six. For comparison, the rate of recombination is also displayed.

Error! Not a valid link.

Figure 4-4 Moving Average for Uniform Counts models of Depth 7-9

10 point moving averages of the SVM predictions are displayed for models with depths of seven, eight and nine. For comparison, the rate of recombination is also displayed.

Test High Low Test

Test High Low Test

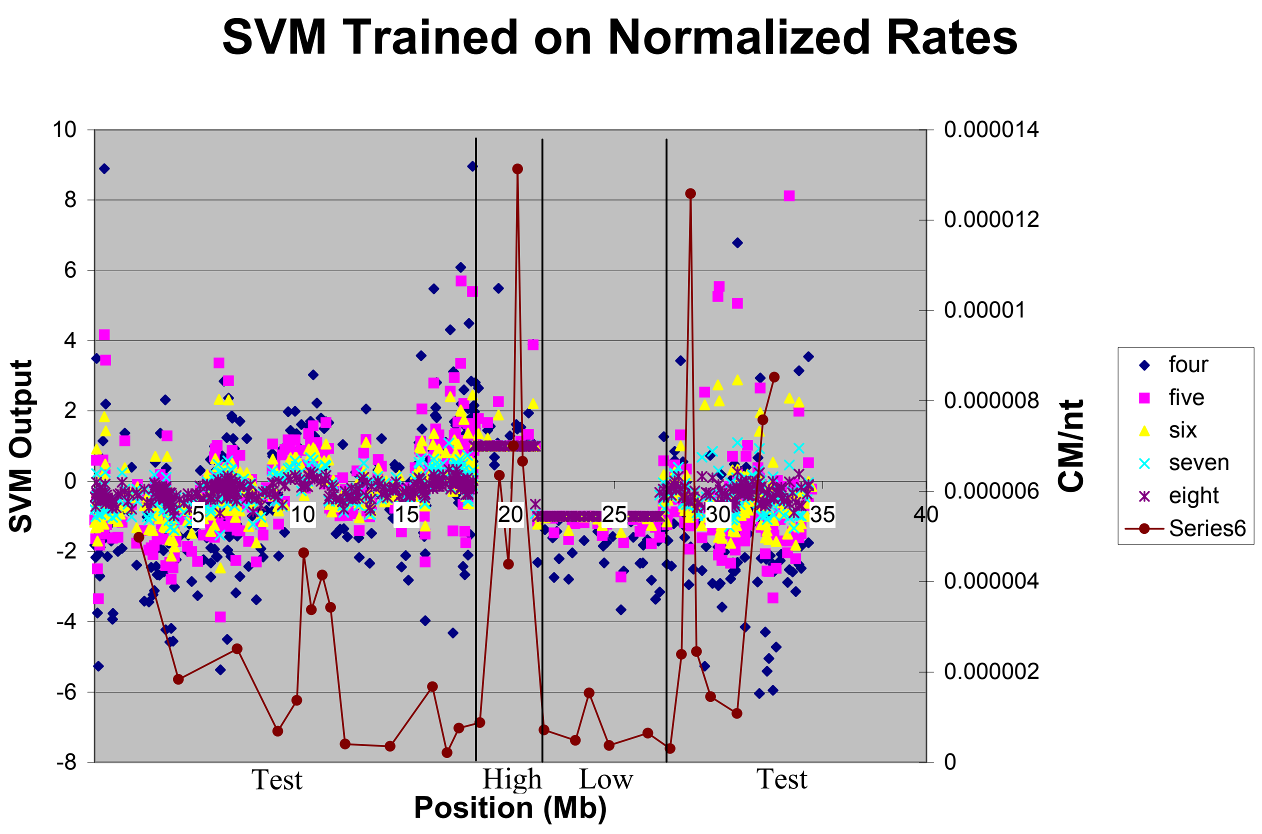

Figure 4-5 Total Results of Training the SVM using Normalized Rates

SVM predictions are displayed as points on the scatter graph. For comparison, the recombination rate is also displayed.

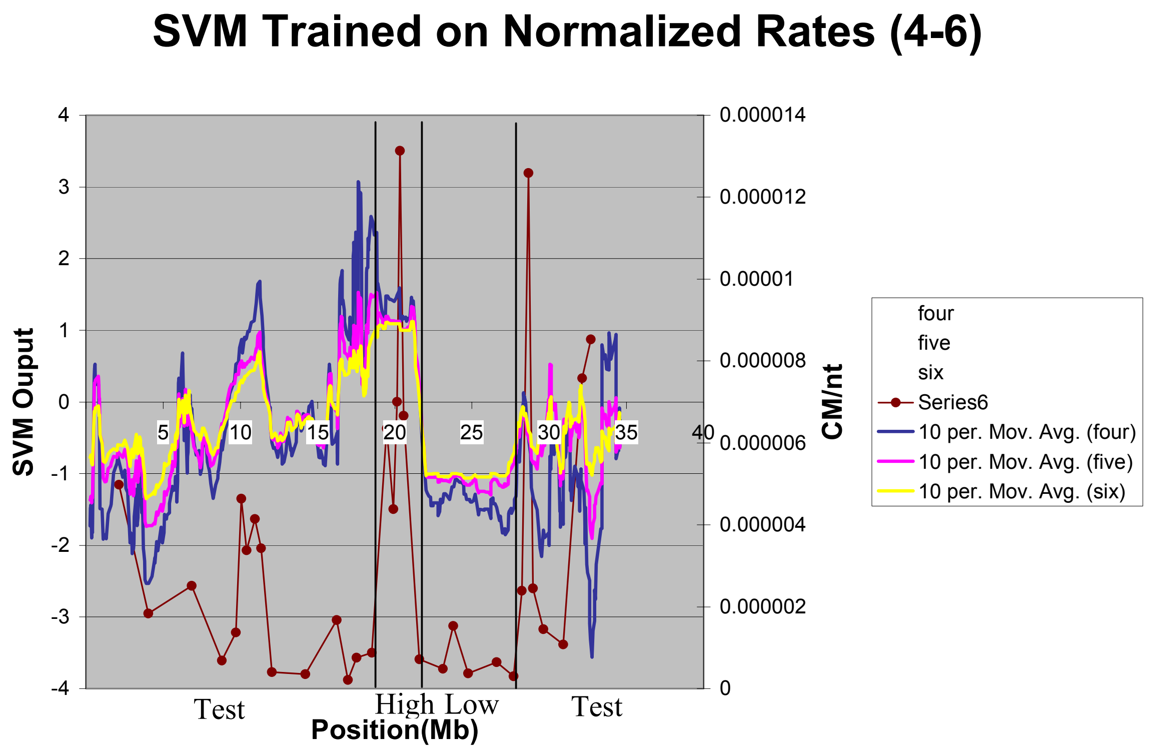

Figure 4-6 Moving Average for Normalized Rates: Depths 4-6

10 point moving averages of the SVM predictions are displayed for models with depths of four, five and six. For comparison, the rate of recombination is also displayed.

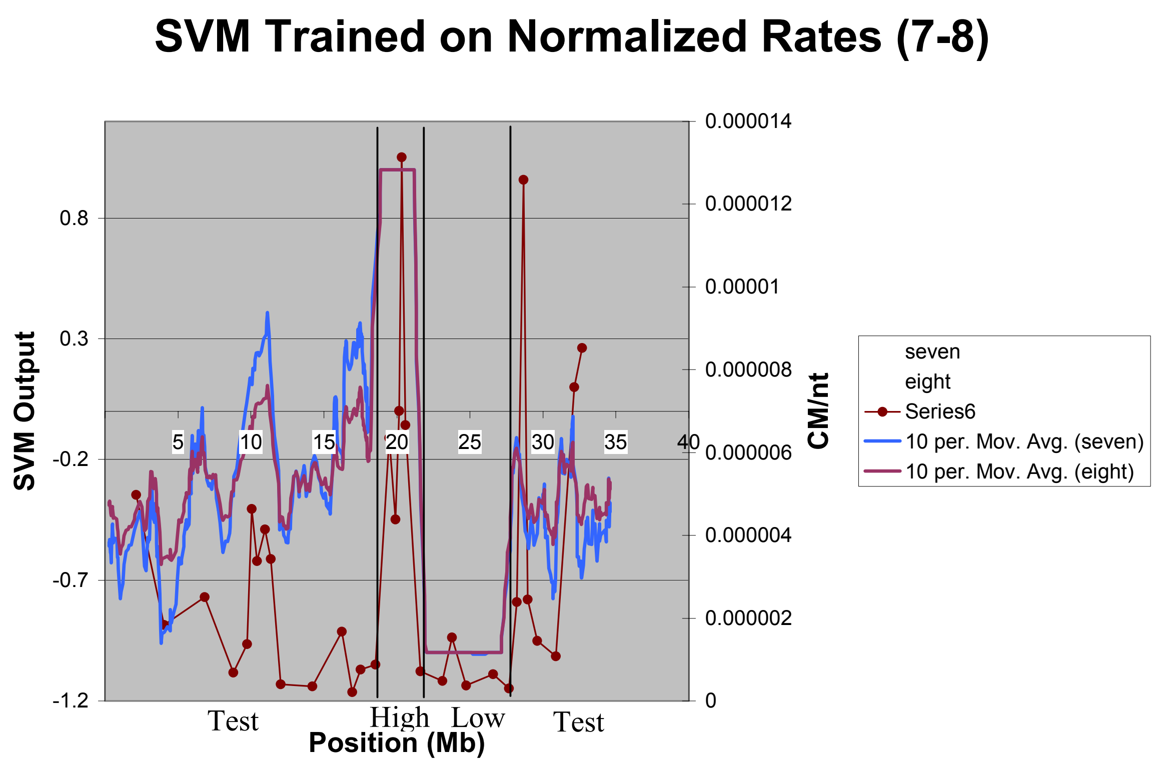

Figure 4-7 Moving Average for Normalized Rates: Depths 7-9

10 point moving averages of the SVM predictions are displayed for models with depths of seven, eight and nine. For comparison, the rate of recombination is also displayed.

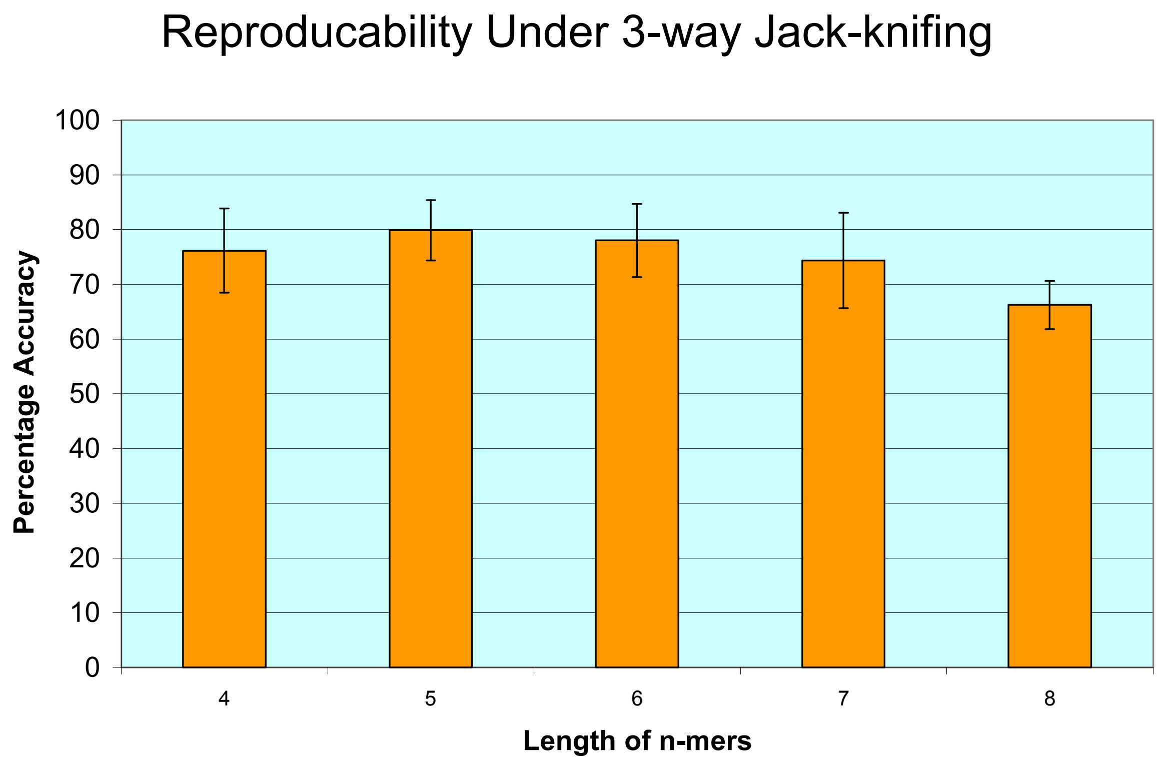

Figure 4-8 Accuracy for Recombination SVMs Under 3-Way Jack-knifing

The percentage accuracy for each group of jack-knifed models is displayed as a bar. The y-axis displays the percentage accuracy of each group of jack-knifed models. Error bars represent two standard deviations. The bar representing 5-mers has the greatest height.

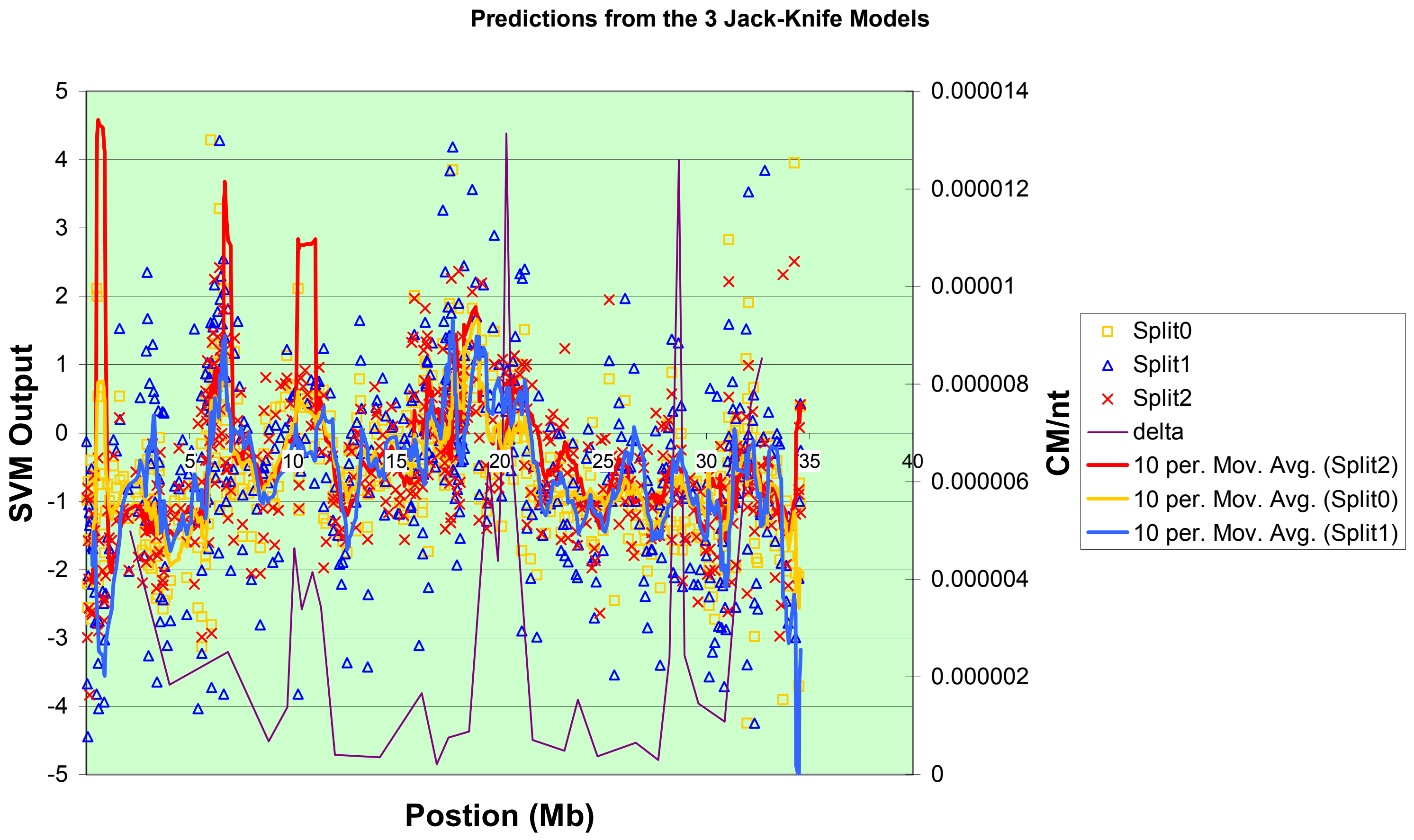

Figure 4-9 Predictions Across the Entire Chromosome from the 3 Jack-knife Models for Depth of 5

4.4 Discussion

It is clear that the normalized rates kernel appears to be more robustly predicting recombination rates than the uniform counts kernel. This exposes one of the interesting properties of SVMs when dealing with data containing errors. The relative magnitude of each data-space basis (here each word) acts in the spirit of a prior over how likely this basis is to form part of the normal to the hyper-plane. In theory, the training method for SVMs should allow these affects to be removed. In practice, with incomplete and stochastically sampled data sets it seems to be important to try to make sure that the raw data is presented in such a way as to normalize out any biases (such as normalizing vectors, pre-normalizing each dimension and the like). It is possible that the RVM methodology (explored in Chapter 5 for a different type of problem) would be more robust for this kind of data as during training it makes an estimate of the degree of certainty with which parameters can be set. This would have the effect of removing dimensions that sharply but inconsistently affect the predicted value regardless of their magnitude. SVMs will tend to be locked into local minima associated with these sharply varying dimensions, as they are greedy algorithms (they search for a global maximum by hill climbing).

The jack-knife results indicate that the signal being learned is not strongly dependant on the training data. This is evidence that these models are learning real signals. It is interesting that the 3 jack-knife models appear to be memorizing their training sets but still generalize in comparable ways to each other. Because the function learned is represented as a sum of normals to a plane (each support vector representing one of these normals), different combinations of support vectors and associated weights may be constructing a very similar hyper-plane. It would be interesting to attempt to calculate the total normal to the hyper-plane in each case and in the case where all training data is used to find the short sequences that are actually informative. This has not yet been attempted, as the raw data has gone though several data-to-feature space projections before arriving in the space for which the hyper-plane is constructed and it is not clear how to represent these transforms in terms of raw word counts.

Recombination hot spots are still not, to our knowledge, fully explained or predictable by any method available. However, recently there has been evidence that poly GT/AC tracts may contribute to differences in recombination rates (Gendrel, Boulet et al. 2000; Majewski and Ott 2000). In the light of this, it would be worth re-running the analysis with the now finished and un-masked chromosome 22 sequence. The masked sequences used here almost certainly had most of this signal screened out. An important message to this story is that 'junk' DNA may well have functionality within the genome, and should not be discarded out-of-hand. The genome contains sequences with many functions, many of them which are not directly related to coding for valid proteins.