

# Comparing the Performance of Several Popular Machine Learning Algorithms on Classifying TATA-box from putative TATA box

Raja Loganantharaj  
Center for Advanced Computer Studies  
University of Louisiana at Lafayette  
logan@cacs.louisiana.edu

## Abstract

*A TATA box is a common transcription binding site that occurs in the upstream of a transcription start site of many genes. Identifying a TATA box accurately is important since it has been shown empirically that a transcription start site (TSS) occurs in the downstream of a TATA box after a fixed distance that is only dependent on the species. Unfortunately, many substrings of a DNA sequence fit to the profile of a TATA box and such substrings are called putative TATA boxes. Identification of a TATA box among putative TATA boxes will improve the accuracy of determining a TSS and hence the detection of a promoter. In this paper we investigate the effectiveness of several popular machine learning algorithms for discriminating TATA box from putative TATA boxes. These algorithms include naïve Bayes, artificial neural network, decision tree C4.5, random forest and support vector machine. To compare the effectiveness, we use a metric of prediction accuracy, true and false positive. Empirically we have shown that tuned support vector machine has outperformed all other machine learning algorithms*

**Keywords:** Positional weighted matrix, neural network, prediction estimates, binding sites

## 1. Introduction

To understand a regulatory mechanism, it is important to detect all the binding sites in a promoter region. A TATA box is a common transcription binding site and it is usually detected by matching the profile 5'-TATAWAW-3' where W is either A or T. Unfortunately there are several substrings in the neighborhood of a TATA box fit to the profile which make the problem of recognizing the real TATA box from putative TATA boxes hard.

Our previous work [1, 2] on discriminating TATA box from putative TATA boxes has revealed that the neighborhood around a TATA box carries the information required to distinguish TATA box from putative TATA boxes. We have investigated which among the upstream or the downstream neighboring

strings of a core TATA promoter is influential in discriminating a TATA box from a putative TATA box. In this paper, we are investigating the effectiveness of some popular machine learning algorithms in recognizing a TATA box from putative ones.

This paper is organized as follows. After the introduction, we describe our approach. In section 3, we describe the experiments and their results. It is followed by a summary and conclusion in section 4.

## 2. Our Approach

The consensus sequence of a core TATA motif is 5' – TATAWAW – 3' and any substring of a DNA having the similar profile of a core TATA box is identified as a putative TATA box. Among many methods, a position specific weighted matrix (PSWM) has been used very successfully to detect a motif in a sequence knowing the profiles of the motif. We, therefore, use a PSWM to detect all the putative TATA boxes from a given set of promoter sequences.

The problem of discriminating a TATA box from a set of putative TATA boxes becomes a problem of a binary classification. A machine learning algorithm can be applied to learn the patterns from the known set of TATA and non TATA boxes, and then be used to classify an unknown putative TATA box into a TATA or a non TATA box.

### 2.1 Detection of TATA Box

The consensus sequence of a TATA core alone does not help to identify a putative TATA box. The TATA box is usually detected by position specific weighted matrix constructed from a profile. The profile of a TATA box is obtained by finding the positional probability distribution of the nucleotides {a,c,g,t} from a set of annotated set of TATA boxes. We have used the profile of a TATA box of plant genome provided in [3] and for the convenience of the reader we have reproduced the profile in the Table 1. The positional weight at position k of a nucleotide  $e_k$  of

TATA box is given by  $\log_2(p(e_k)/bgp(e_k))$  where  $bgp(e_k)$  is the background probability of nucleotide  $e_k$ .

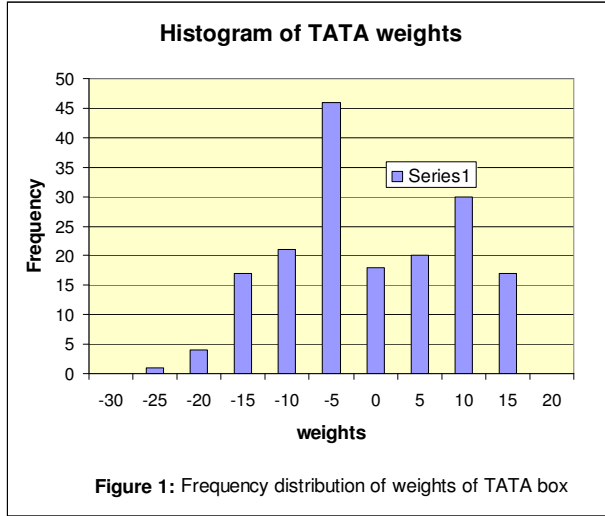


Figure 1: Frequency distribution of weights of TATA box

By scanning the entire promoter regions from -200 1 of all the TATA and TATA-less promoters, the background probability of the nucleotide  $a, c, g$  and  $t$  are obtained as 0.310377, 0.231451, 0.164437 and 0.293735. To allow for the possibility of a nucleotide that did not appear in the profile, we replace 0 in the profile with 0.001 for the computation of PSWM. The computed PSWM of a TATA box is given in Table 2.

	-2	-1	1	2	3	4	5	6	7	+1	+2
A	0.28	0.16	0.03	0.95	0	1	0.62	0.97	0.38	0.73	0.13
C	0.27	0.63	0.01	0	0.04	0	0	0	0.01	0.08	0.42
G	0.17	0.05	0	0	0	0	0	0.02	0	0.1	0.28
T	0.28	0.16	0.96	0.05	0.96	0	0.38	0.01	0.61	0.09	0.18
			T	A	T	A	W	A	W		

Table 1: Profile of a TATA box of plant genome

sigmoid activation for the networks used here.

During the training phase, the input is propagated to the

	-2	-1	1	2	3	4	5	6	7	+1	+2
A	-0.15	-0.96	-3.37	1.61	-8.28	1.69	1.00	1.64	0.29	1.23	-1.26
C	0.22	1.44	-4.53	-7.85	-2.53	-7.85	-7.85	-7.85	-4.53	-1.53	0.86
G	0.05	-1.72	-7.36	-7.36	-7.36	-7.36	-7.36	-3.04	-7.36	-0.72	0.77
T	-0.07	-0.88	1.71	-2.55	1.71	-8.20	0.37	-4.88	1.05	-1.71	-0.71

Table 2: PSWM of a TATA box of plant genome

A TATA box is detected when the summation of the weights in PSWM corresponding to that of the nucleotides in a substring of length 7, which is the length of a core TATA box, exceed an appropriate threshold value. If the threshold value is low, it may accept many subsequences as putative TATA box while higher value will ignore genuine TATA box. It is not

easy to determine the optimal value of a threshold and there is no guideline to determine such value. From the set of known TATA boxes we have created a histogram of their weight distribution and it is shown in Figure 1 and we have selected a threshold -15.65 that allows over 90% of known TATA boxes. Using this threshold, we have detected all the putative TATA boxes in a given DNA sequence.

## 2.2 Artificial Neural Networks

A feed forward multi-layer back propagating neural network, which is referred to as ANN [4, 5] in this sequel, have been used for many applications involving pattern recognition and prediction purposes in bioinformatics [6]. We have used 3 layer networks consisting of input layer, a hidden layer and an out layer. Since the network is used for binary classification, the output layer has a single node. Each node in the input layer is connected to all the nodes in the hidden layer. Similarly, all the nodes in the hidden layer are connected to all the nodes in the output layer. During a forward propagation, all the nodes except those at the input layer accumulate (sum) the weighted output of each node from the previous layer and from a bias node connected to the layer. The output of a node becomes the results of applying an activation function to the accumulated inputs of a node. We have used a

output layer and the error, which the difference between the expected output and the current output, is computed. If the sum of the square of all the errors is within the tolerable limit it is accepted, otherwise the error is propagated backward. The weight of the connected layer is adjusted proportional to the learning rate and to the partial derivative of the error with respect to the weight. The forward propagation of signal and the backward propagation of error and the

subsequent adjustments to the weights are continued until the outcome is acceptable.

### 2.2.1 The model

The training and the test sequence consists of nucleotides around TATA box and each of these sequences is modeled as a string of length  $2k$  with alphabets  $a, c, g$  and  $t$  representing possible nucleotides. Each position of the string is represented by 4 distinct input nodes one for each of the nucleotides. The total number of input node is  $8k$  for the string of length  $2k$ . For example, if the input string is  $a, c, g, t$  then the input will be  $[1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1]$

## 2.3 Decision Tree C4.5

A decision tree is a simple, but a powerful machine learning algorithm that has been used successfully for classification problems. Each leaf node represents a class, while each internal node represents an attribute. When classifying an instance, a series of decisions was made during the traversal from root to a leaf node and the instance was classified to the one associated with the leaf node at the end of the traversal. Each internal node is a decision node and a value of a given instance is compared to the decision function to decide which branch to follow. A decision tree is build using a training data set so as to reduce the average depth of each path from root to leaf node and to be flexible enough to avoid data over fitting. The popular algorithms for decision tree include id3 [5, 7] and its successor C4.5 [5, 8]. Both algorithms are using changes in entropy to achieve overall shorter depth of all the paths from root to leaf nodes. The algorithm C4.5 is an improved version of ID3 by allowing continuous variable and partition optimization for them and as well providing tree pruning techniques to avoid data over fitting.

The attributives in our case is the nucleotides in the sequences flanking a putative TATA box in position  $-k$  through  $k$  and each attribute will take one of the four values  $[a, c, g, t]$ .

## 2.4 Random Forest

Breiman [9] has proposed random forest that can be used as a classifier and as well as a clustering algorithm. From each training example, the algorithm builds a tree starting with a feature that is selected randomly and grow to its maximum depth. When classifying, the decision is made by aggregating (majority vote for classification and average for

regression) the prediction of all the trees built during training. Since each tree in the ensemble is built non-deterministically, the classifier is not sensitive to noise and it seems to be outperforming single tree classifier such as CART and C4.5. Random forest has a built in out of bag estimator that enables to perform cross validation during the forest building phase which in turn makes the classifier to improve its accuracy without over fitting, which makes the system better than any other decision system based on single tree.

## 2.5 Support vector machine

Support vector machine [10, 11] is a linear binary classifier in feature space that maximizes the separation between decision boundaries of two classes in the feature space. Several kernel functions including the following ones are used to map the data point indirectly into the feature space: Linear, quadratic and radial bias kernels. The performance in terms of prediction accuracy is dependent on the kernel and the parameters; therefore, one should pay close attention to selecting an appropriate kernel and its parameters. It has been suggested [12] that radial bias function has performed well in several areas of classification besides the obvious advantage of having only two parameters namely the cost  $C$  and the gamma.

## 2.6 Naïve Bayes

Naïve Bayes have been successfully used to many classification problems in various domains including bioinformatics. Given a surrounding sequences, say  $e_{-k}, e_{-k+1}, \dots, e_{-1}$  and  $e_1, \dots, e_k$ , The probability of a class  $C$  is given by

$$P(C | e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k) =$$

$$P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k | C) \cdot P(C) / P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k)$$

Using chain rule and applying positional independence

$$= \prod_{m=-k}^k P(e_m | C) \cdot P(C) / P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k)$$

$$P(\neg C | e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k) = P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k | \neg C) \cdot P(\neg C) / P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k)$$

$$= \prod_{m=-k}^k P(e_m | \neg C) \cdot P(\neg C) / P(e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k)$$

Given the flanking sequence, say  $e_{-k}, e_{-k+1}, \dots, e_{-1}$  and  $e_1, \dots, e_k$ , a putative TATA box is classified as a TATA box if  $P(\text{total } e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k) > P(\neg \text{total } e_{-k}, e_{-k+1}, \dots, e_{-1}, e_1, \dots, e_k)$ . The conditional probability is computed using naïve Bayes simplification as shown above. Make a note that the prior probability of the positive and the negative classes are computed from the

training instances, which often do not reflect the prior probabilities correctly.

### 3. Experiments

We have downloaded promoter sequences of plant genome from PlantProm DB (<http://mendel.cs.rhul.ac.uk/mendel.php?topic=plantprom>), an annotated non-redundant collection of proximal promoter sequences for RNA polymerase II with experimentally determined transcription start site(s) (TSS) from various plant species. The current release of PlantProm DB contains 305 entries, of which 71 are monocot, 220 are dicot and 14 are other plants.

We started the experiment with the detection of all TATA boxes in TATA promoters using PSWM as we have described in the previous section. The putative TATA box found in the TATA promoters region between -40bp to -10bp from the TSS is indeed the TATA box. We have collected the substrings of length 15bp in the downstream and in the upstream from the core TATA box (TATAWAW) and these strings form the positive instances. We have scanned the promoter regions from -200bp to -40bp for TATA box and the substrings flanking these putative TATA boxes become the negative instances. The probability of finding a

16,384 bp by chance. We found abundance of putative TATA boxes in the promoter region, about 2 or 3 within 160bp length. Since negative instances were over 2 times of that of positive instances, we have created 3 data sets for each offset with equal positive and negative instances in each set.

To find the relationship between the length of the surrounding TATA box and its influence on discriminating TATA from putative TATA boxes, we have collected positive and negative instances of the data for substrings of length 3, 6, 9, 12 and 15.

Weka [13] is a popular data mining workbench written in Java and is available for free downloading. Weka consists of many popular machine learning algorithms and it provides several options for testing data sets. We have used the latest version of Weka to compare the performance of Naïve Bayes, artificial neural network, random forest, Decision tree C4.5 and Support Vector Machine. Each of these algorithms was tested with 10 fold cross validation, that is, the data set was randomly divided into 10 equal parts and the algorithm was trained with all but one partition, which was used for testing. The training and testing were repeated 10 times so that each partition was tested exactly once. It has been empirically shown [14] that 10 fold cross

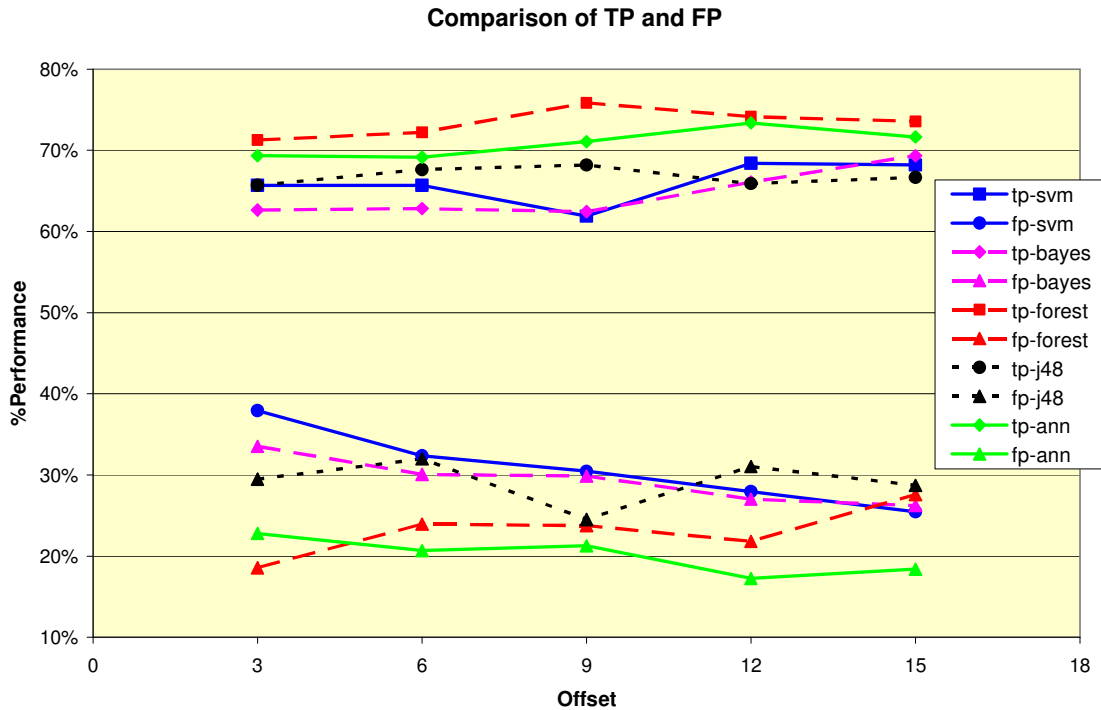


Figure 2: TP and FP of different classifiers VS offset

TATA box in a genomic sequence is  $(1/4)^7$ , that is, one can find a TATA box in every sequence of length

validation gets the best estimate of error.

### 3.1 Results

Suppose, the positive and negative instances in the test set are respectively are  $P_i$  and  $N_i$

For the comparison, we have used the following metric.

$$\text{True Positive (TP)} = \text{TP}_{pi} / P_i$$

$$\text{False Positive (FP)} = 1 - \text{TN}_{ni} / N_i$$

$$\text{Prediction accuracy (PA)} = (\text{TP}_{pi} + \text{TN}_{ni}) / (P_i + N_i)$$

For each offset from 3 to 15 in step of 3, we have

of large number of negative instances. A data set, say offset15\_1.arff denoting set 1 with offset 15, was loaded into WEKA. Each of the following classifiers was run with the data set with 10 fold validation and the results were recorded: artificial neural network (ANN), decision tree C4.5 (J48), Naïve Bayes, Support vector machine (SVM) and Random forest. Similarly other two sets with the same offset were loaded and the test was conducted. The average results of true positive, false positive and prediction accuracy are computed and are shown Table 3.

The variation of true positive and false positive of different classification algorithms over the surrounding

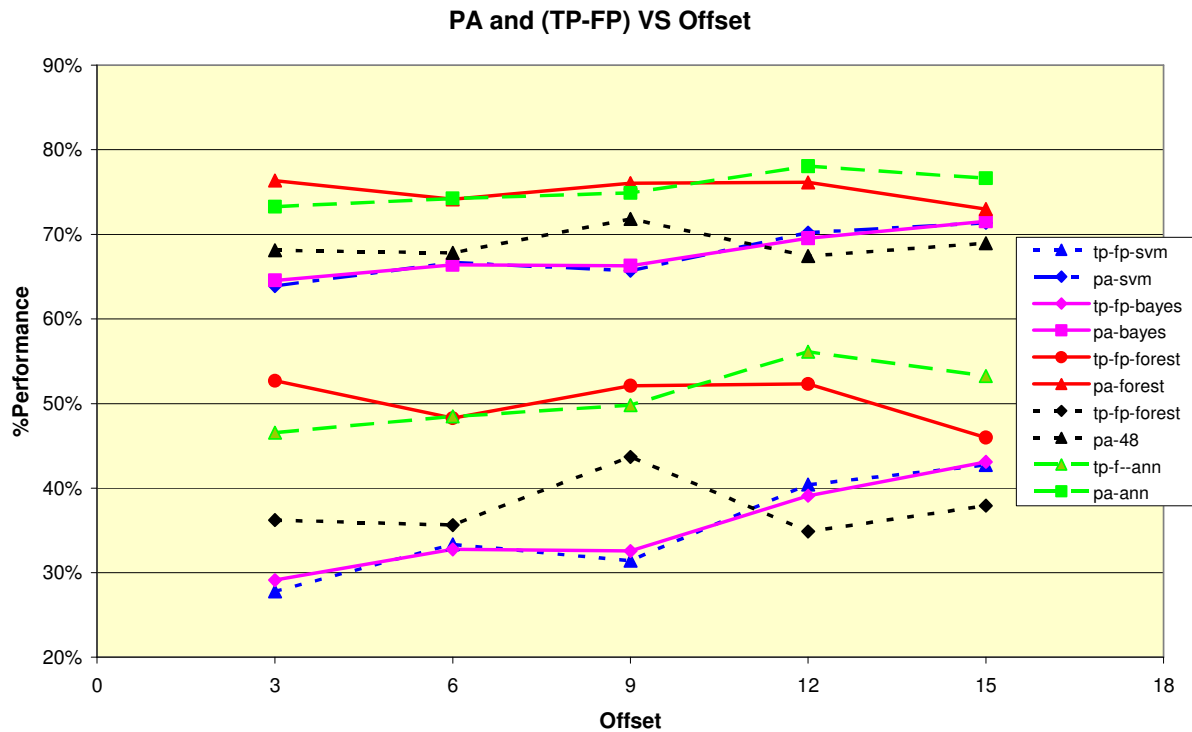


Figure 3: PA and the differences between TP and FP against Offset

created 3 separate data sets to avoid biasing as a result sequence length is plotted as shown in Figure 2. The

Length around TATA box	SVM			Naïve Bayes			Random Forest			J48			ANN		
	TP	FP	PA	TP	FP	PA	TP	FP	PA	TP	FP	PA	TP	FP	PA
3	66%	38%	64%	63%	34%	65%	71%	19%	76%	66%	30%	68%	69%	23%	73%
6	66%	32%	67%	63%	30%	66%	72%	24%	74%	68%	32%	68%	69%	21%	74%
9	62%	30%	66%	62%	30%	66%	76%	24%	76%	68%	25%	72%	71%	21%	75%
12	68%	28%	70%	66%	27%	70%	74%	22%	76%	66%	31%	67%	73%	17%	78%
15	68%	25%	71%	69%	26%	72%	74%	28%	73%	67%	29%	69%	72%	18%	77%

Table 3: Comparison of performance of different algorithms

best classification algorithms should have the highest prediction accuracy and true positive rate while having the lowest false positive rate. To visualize the comparison of algorithm, we have plotted a graph in Figure 3 with total prediction accuracy with the difference between true positive and false positive rate. The best classifier should have the highest prediction accuracy in addition to having the largest difference between the true positive and false positive rate.

The artificial neural network outperformed the rest of the algorithms and the best results occurred at the offset 12. It has the highest prediction accuracy of 78% with the true positive rate of 73% and false positive rate of 17%. The performance of random forest is very close to that of a ANN. The support vector machine showed dismal performance and it very closely resembles that of a naïve Bayes classifier. The results surprised us since we were expecting a better performance from a support vector machine. WEKA provides little or no control over selecting parameters for a SVM whose performance is sensitive to parameter selection. To explore the entire parametric space of a SVM we have used libSVM [15] with binary encoding for the nucleotide input. We have used radial bias kernel. To explore a wide range of cost parameter C and gamma

space with 10 fold cross validation. The results are shown in Figure 4.

#### 4. Summary and conclusion

Accurate detection of transcription binding sites plays an important role in studying regulatory mechanism. The binding sites are often detected by applying position specific weighted matrix computed from the profiles of the motif of interest. When a profile fits to more than one site, it becomes a difficult problem for discriminating putative binding sites from the real one.

TATA box is a common transcription binding site for many genes. Unfortunately, there are many substrings in the neighborhood of TATA box that fit to the profile of TATA box. The occurrence of a core TATA box in a genomic sequence by chance is one in 16K bp. But, we found over abundance of TATA boxes in the promoter regions (about 2 to 3 in each sequence in the promoter region of length 160bp).

Our previous work on discriminating TATA box from putative TATA boxes has revealed that the neighborhood around TATA box and putative TATA box have different composition. In this paper we have

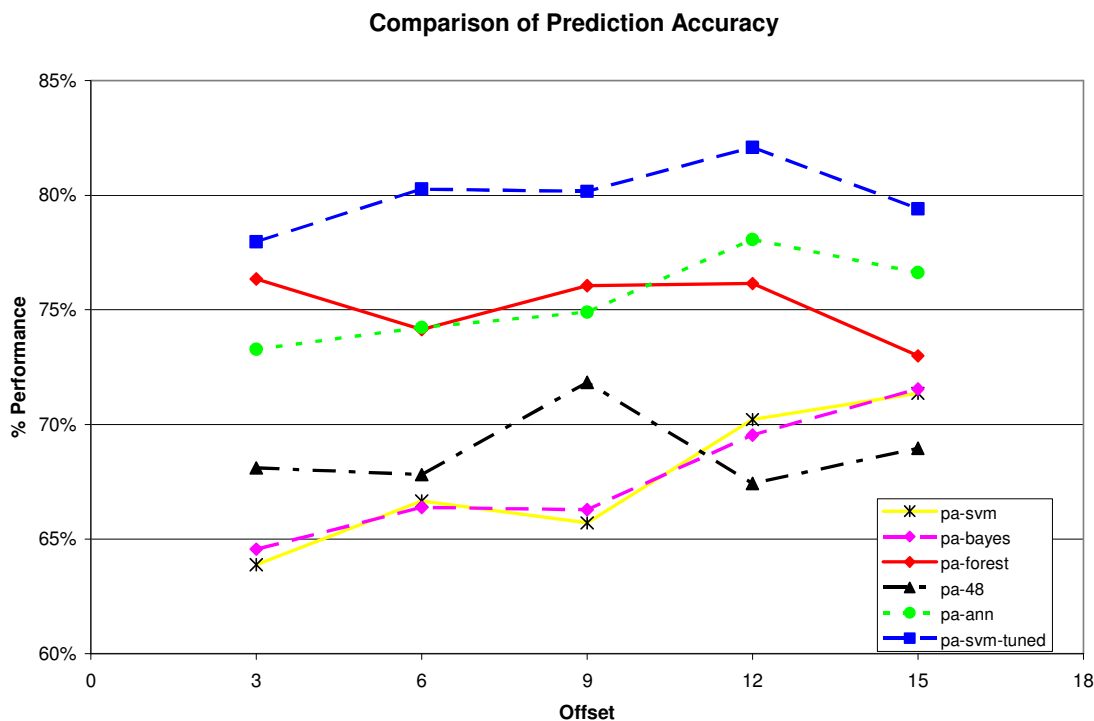


Figure 4: Comparison of prediction accuracy

we have used the Python script called *grid* and we have obtained the best result from SVM over the parametric

further investigated the effects of the neighboring strings in discriminating TATA box from putative

TATA boxes. We have used the PSWM to detect putative TATA boxes. From the annotated sequences of the plant promoters, we have selected the subsequences in the upstream and in the downstream of a TATA promoter and these substrings formed positive instances. The problem of discrimination becomes a problem of binary classification. Several machine learning algorithms have been used in Bioinformatics successfully for classifying purposes and their effectiveness is very much dependent on the data set. We have selected the following popular machine learning algorithms for comparing their effectiveness for this particular problem: artificial neural network (ANN), decision tree C4.5 (J48), Naïve Bayes, Support vector machine (SVM) and Random forest. We have used Weka, a data mining workbench, for testing these algorithms on this data set. Each algorithm was run with 10 fold cross validation and the results such as true positive, false positive and total prediction accuracy are shown in Table 3. For visual comparison, we have plotted the results in Figure 2 and 3.

The best classifier should have the highest prediction accuracy and true positive rate while having the lowest false negative rate. Based on the results from WEKA, the artificial neural network outperformed the rest of the algorithms and the best results occurred at the offset 12. It has the highest prediction accuracy of 78% with the true positive rate of 73% and false positive rate of 17. The performance of SVM was very disappointing and it was at the same level of performance as that of a Naïve Bayes.

Since WEKA does not provide any options to control the parameter of kernel function of a SVM, we have used libSVM with binary encoding for the nucleotides input. We have used radial bias kernel and explored the parameter space of C and gamma using Python script called grid. The exploration over the parametric space found the best combination of C and gamma that maximizes the prediction accuracy with 10 fold cross validation. The results of the tuned SVM are shown in Figure 4 and it is compared with the performance of other algorithms. The a tuned SVM showed a remarkable prediction accuracy of 83% which support our hypothesis that a promoter region can be detected with high degree of certainty when it is combined with other detection of binding sites in the promoter region.

The performance of a Naïve Bayes can also be improved by adjusting the prior probabilities so as to maximize the prediction accuracy. We are making progress on automatically finding appropriate threshold for decision in a naïve Bayes classifier.

**5. Acknowledgments:** This work is partially supported by the Governor's IT initiatives and the author would like to acknowledge the grant given to the Bioinformatics Research laboratory.

## 6. Reference

1. Loganantharaj, R., M.E. Karim, and A. Lakhotia. *Recognizing TATA promoters based on discriminating frequency analysis of neighborhood tuples*. in *Biot-04: First Biotechnology and Bioinformatics Symposium: A Community and Academic Forum*. 2004, September. Colorado Springs, Colorado.
2. Loganantharaj, R., *Discriminating TATA-box from putative TATA box in plant genome*. International Journal of Bioinformatics Research and Applications, 2006. **2**(1): p. 36-51.
3. Shahmuradov, I.A., et al., *PlantProm: a database of plant promoter sequences*. Nucleic Acids Res, 2003. **31**(1): p. 114-7.
4. Russell, S.J. and P. Norvig, *Artificial intelligence: a modern approach*. 2nd ed. Prentice Hall series in artificial intelligence. 2003, Upper Saddle River, N.J.: Prentice Hall. xxviii, 1080 p.
5. Winston, P.H., *Artificial intelligence*. 3rd ed. 1992, Reading, Mass.: Addison-Wesley Pub. Co. xxv, 737 p.
6. Baldi, P. and S. Brunak, *Bioinformatics: the machine learning approach*. 2nd ed. Adaptive computation and machine learning. 2001, Cambridge, Mass.: MIT Press. xxi, 452 p.
7. Quinlan, J.R., *Simplifying decision trees*. International Journal of Man-Machine Studies, 1987. **27**: p. 221-234.
8. Quinlan, J.R., *C4.5: Programs for Machine Learning*. 1993: Morgan Kaufman.
9. Breiman, L., *Random Forest*. Machine Learning, 2001. **45**: p. 5-32.
10. Cristianini, N. and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. 2000, Cambridge University Press: Cambridge; New York.

11. Saxena, V. and Massachusetts Institute of Technology. Dept. of Civil and Environmental Engineering., *Support vector machine and its applications in information processing*. 2004. p. 72 leaves.
12. Hsu, C.-W., C.-C. Chang, and C.-J. Lin, *A practical guide to support vector classification*. 2003.
13. *Weka*. 2004. p. Data Mining Software in Java, <http://www.cs.waikato.ac.nz/ml/weka/>.
14. Witten, I.H. and E. Frank, *Data mining: practical machine learning tools and techniques*. second ed. 2005, San Francisco, Calif.: Morgan Kaufmann. xxv, 524.
15. Chang, C.-C. and C.-J. Lin, *LIBSVM: a library for support vector machines*. 2001.