Non-Informative Title
Fully Bayesian Analysis of RVMs

Dan Murphy and Sean Abreau

May 2019

1 Model Description

Bayesian methods are extremely useful in describing the underlying mechanisms that result in certain sets of observations. Even if we have little information regarding the prior distributions of the parameters of our models, we can often employ non-informative priors to allow the likelihood of our observed data (given our model) to heavily influence the posterior distributions of our parameters.

Unfortunately, the success of this procedure relies on our ability to choose reasonable models, given our expectations of how our data is distributed. We may not always have correct intuition in this respect, especially if we are dealing with irregular or high-dimensional data. In the case of low-dimensional irregular data, we have seen transformations (like the Box-Cox transformation) that attempt to regularize our data's distribution to allow for easier Bayesian inference, but such methods still often require explicit tuning.

This may explain the popularity of discriminative methods when trying to perform tasks like classification. We specifically look to SVMs, which use kernel transformations to separate a given set of data while making very few assumptions about its initial distribution (other than that it be reasonably linearly separable after projecting it into a higher-dimension determined by the choice of kernel). This makes them widely applicable to various data distributions, as they assume little prior knowledge about the distributions. But their loss functions must still be manually tuned to be more or less sensitive to outliers. A lot of effort may also be expended to detect and remove such outliers using other discriminative heuristics, like random forests or KNN models.

A Relevance Vector Machine (RVM) is a fairly new model, proposed in 1999, that has recently gained popularity, which is very similar to an SVM except with a Bayesian formulation. Such a formulation may allow us to better reason about what it does, by working with distributions over parameters rather than heuristics. It achieves comparable recognition accuracy to the SVM, yet provides a full predictive distribution, and also requires substantially fewer kernel functions. Like an SVM, it has advantages for high dimensional or irregularly distributed data, since it again assumes very little about the underlying data distribution.

1.1 Classification RVM

In Michael Tipping's paper, he formulates an RVM for classification as follows:

Given:

- $X \in \mathbb{R}^{N \times D}$ is our data
- $Y \in \{0,1\}^N$ is a vector of binary labels

Let:

- $\Phi \in \mathbb{R}^{N \times N+1}$ be a kernel matrix, where $\Phi_{i,j+1}$ represents $\texttt{Kernel}(X_i, X_j)$ and $\Phi_{\cdot,0}$ is a columns of 1s
- $w \in \mathbb{R}^{N+1}$ be a vector of weights
- $\alpha \in \mathbb{R}^{N+1}$ be a vector of parameters

The RVM model now assumes:

$$\begin{aligned} Y &\sim \texttt{Bernoulli}(\texttt{logit}^{-1}(\Phi w)) \\ w &\sim \texttt{Norm}(0, \alpha^{-1}) \\ \alpha &\sim \texttt{Unif}(-\infty, \infty) \end{aligned}$$

The model is therefore a form of hierarchical model, in which the posterior distribution is:

$$p(w, \alpha | Y, \Phi) \propto \texttt{Bernoulli}(\texttt{logit}^{-1}(\Phi w)) \cdot \texttt{Norm}(0, \alpha^{-1})$$

To classify a new datapoint, $\hat{x}$, we take

$$\hat{y} \sim \texttt{Bernoulli}(\texttt{logit}^{-1}(w_0 + \sum_{i=1}^{N} \texttt{Kernel}(X_i, \hat{x}) w_i)))$$

Each w_i essentially corresponds to an element of our dataset X_i, and it relates that datapoint to the distribution of our observed labels.

In the original paper, the authors note that by giving each weight w_i a unique variance α_i^{-1} that is independent of all others, and by using a diffuse prior on α, the results of this model are such that most elements of w are 0, while a few take on larger values. They deem the vectors that correspond to these large values "relevance vectors."

As previously stated, this model makes few assumptions regarding the underlying data, except possibly through the choice of kernel function. The authors note that unlike SVMs, an RVM can take an arbitrary kernel function, so if we are able to reason about our data distributions, we could incorporate this knowledge through a choice of kernel. The main explicit assumption that this model makes is that the distribution of our observations, Y, can be summarized by choosing only a few of our datapoints, which is a criterion that we would hope a sufficiently large dataset would satisfy.

1.2 Reformulated Classification RVM

Though the original RVM model has excellent flexibility, it does have several downsides, which make it intractible to sample from in a fully Bayesian manner. This led us to derive a reformulated variant of the original RVM model.

Our initial reformulated version is the probit model:

$$\begin{aligned} Y &\sim \texttt{Bernoulli}(\mathit{\Phi}(\Phi w, 1)) \\ w &\sim \texttt{Norm}(0, \alpha^{-1}) \\ \alpha &\sim \texttt{Unif}(-\infty, \infty) \end{aligned}$$

We discuss the advantages of this model in the sampling section below.

Our initial reformulation still had undesirable convergence properties, however. This led us to propose an improved model:

$$\begin{aligned} Y &\sim \texttt{Bernoulli}(\mathit{\Phi}(\Phi w, \sigma^2)) \\ w &\sim \texttt{Norm}(0, \alpha^{-1}) \\ \alpha &\sim \texttt{Unif}(a, \infty) \\ \sigma^2 &\sim \sigma^{-2} \end{aligned}$$

where $\mathit{\Phi}(\mu, \sigma^2)$ is the cumulative normal distribution centered at μ with variance σ^2, and a is a constant.

The motivation and analysis for this model are discussed in the model verification section below.

2 Methods

The original RVM classification model does not have a nice posterior distribution for the weights, since it involves the product of Bernoulli and Normal distributions, which cannot be directly sampled.

2.1 Maximum A Posteriori Estimates

The original paper ignores this sampling issue by instead focusing on maximizing the posterior distribution of the parameters. An iterative procedure is used to accomplish this:

Given a set of weights, it would first calculate the most likely α values. Then fixing these α values, it would use the gradient of the conditional posterior for $w|\alpha, Y, Phi$ to shift the weights closer to a local maximum of their posterior distribution. The details of this algorithm are described in full in Michael Tipping's paper.

This procedure is efficient; however, the use of the gradient does not guarantee that it will find a global maximum - it is subject to converging to only a local maximum. Furthermore, it only provides point estimates for our parameters; we cannot analyze the entire posterior distribution to make inferences about our final sets of weights in a true Bayesian manner. We focused much of our efforts into deriving methods for sampling the full posterior distribution, to make sense of what this model is doing.

2.2 MCMC Sampling

If we wish to sample the full distribution, we again have to deal with the posterior distribution for w. Using MCMC sampling, we can attempt to overcome irregular distributions such as this. We implemented this model in `Stan`, which even compiles python code to C++, in order to sample most efficiently (see code appendix). Nonetheless, after 7 minutes, `Stan` had collected fewer than 200 samples.

We believe the bottleneck to be the large number of parameters in the model. There is a weight for every datapoint, and they cannot be independently sampled. In such a high dimensional space, the sampling distribution for all parameters becomes sharply peaked, so any proposal for the Markov chain has a high chance of being rejected (since a poor draw for any single parameter would cause the entire sample to be rejected).

2.3 Gibbs Sampling

When dealing with high dimensional distributions, Gibbs sampling can make things much easier by allowing us to sample entire blocks of parameters at once directly from their conditional posterior distributions. However, the posterior conditional distribution for the weights in the original logistic regression RVM model cannot be directly sampled. This motivates our first model reformulation.

As described above, we can see that this model should perform fairly similarly to the logistic regression RVM, since the cumulative normal function will be peaked in the same area and merely have slightly wider tails. However, we can introduce latent variables, $z \in \mathbb{R}^{N+1}$ that allow us to directly sample all parameters from their conditional posterior distributions in this model.

We let:

$$\begin{aligned} Y &= \texttt{sign}(z) \\ z &\sim \texttt{Norm}(\Phi w, 1) \\ w &\sim \texttt{Norm}(0, \alpha^{-1}) \\ \alpha &\sim \texttt{Unif}(-\infty, \infty) \end{aligned}$$

We now obtain regular conditional posterior distributions:

$$\begin{aligned} p(\alpha_i|w, z, Y) &\propto \alpha^{\frac{1}{2}} e^{\frac{-w_i^2}{2}\alpha} \propto \texttt{Gamma}(\frac{3}{2}, \frac{w_i^2}{2}) \\ p(z_i|w, \alpha, Y, \Phi) &\propto \texttt{TruncNorm}_{Y_i}(\Phi w, 1) \end{aligned}$$

Where $\texttt{TruncNorm}_{Y_i}$ is the truncated normal distribution, which is truncated above 0 if Y_i is 1 and below 0 if Y_i is 0.

The conditional posterior distribution for w is a bit more involved. We first note that $\texttt{Norm}(w; 0, \alpha^{-1}) = \texttt{Norm}(0; w, \alpha^{-1})$. Now we can combine this value with $\texttt{Norm}(z; \Phi w, 1)$ by expressing it as

$$z_0 \sim \texttt{Norm}(\Phi_{Aug} w, \Sigma)$$

where z_0 is z concatenated with a vector of $N+1$ 0s, Φ_{Aug} is Φ concatenated with the $N+1 \times N+1$ identity matrix below it, and Σ is $\begin{bmatrix} \mathrm{I}_{N\times N} & 0 \\ 0 & \texttt{diag}(\alpha^{-1}) \end{bmatrix}$.

This is a familiar regression problem, and from *Bayesian Data Analysis* we see that

$$w \sim \texttt{Norm}(\hat{\beta}, \hat{\Sigma})$$

where

$$\begin{aligned} \hat{\beta} &= (\Phi_{Aug}^T \Sigma^{-1} \Phi_{Aug})^{-1} \Phi_{Aug}^T \Sigma^{-1} z_0 \\ \hat{\Sigma} &= (\Phi_{Aug}^T \Sigma^{-1} \Phi_{Aug})^{-1} \end{aligned}$$

We can exploit the presence of 0s and identity matrices to simplify this linear algebra to be more computationally efficient, deriving

$$\begin{aligned} \hat{\Sigma} &= (\Phi^T \Phi + \texttt{diag}(\alpha))^{-1} \\ \hat{\beta} &= \hat{\Sigma} \Phi^T z \end{aligned}$$

We can thus sample the conditional posterior distributions directly in our model. To further increase the sampling speed, we take advantage of positive semi-definite covariance matrices to sample the high-dimensional multivariate normal distribution of w, using the Cholesky factorization of Σ to transform independent draws from the standard normal into a draw from the multivariate normal. The result is an efficient Gibbs sampler for our model.

3 Model Verification

3.1 Data

To verify the performance of our model, we chose a fairly simple dataset, so that we could best interpret the significance of our results. To accomplish this, we selected the standard Skin Segmentation Dataset (see link in citations). This dataset was sampled from other datasets that were produced specifically to help researchers fit models for general skin detection in images, so it is fairly extensive, and there is little missing data. The actual datapoints were random pixels sampled from images in the Color FERET Image Database and PAL Face Database. They incorporated multiple races, ages, and genders, and photographs of subjects were taken in controlled environments. There were a total of 245057 samples, with 50859 samples of skin and 194198 of non-skin, and each sample had 3 dimensions, for its R, G, and B values. The number of weights to sample in our model increases linearly with the number of observations, so we additionally randomly sampled uniformly from our dataset, thinning the number of observations we were working with down to 490, with 385 pixels of skin and 105 pixels of no skin.

3.2 Initial Model Assessment

We ran 5 different chains, for 5000 iterations each. Our data was not clearly linearly separable, but even using a linear kernel, the results initially looked promising. We indeed observed that most of our weights went to 0, except for a few 'relevant' weights. Selecting samples of our weights, we evaluated the probabilities of our model predicting our observed labels - that is,

$$\Phi(\texttt{logit}^{-1}(w_0 + \sum_{i=1}^{N} \texttt{Kernel}(X_i, \hat{x}) w_i))$$

- and found that all of these probabilities were almost exactly 1 or 0, indicating very steep prediction boundaries. Furthermore, evaluating on new data produced predictions, 86% of which matched the true labels.

Looking further, we found that these results were not actually an effective fit of our model. Looking at the chains for the weights that did not converge to 0, we found that they actually *diverged.* Furthermore, while such steep decision boundaries did not hurt our accuracy much in this case, they may not be a good predictor for other data distributions. They suggest that our formulated RVM model may be prone to overfitting, and that its weights may be underconstrained.

3.3 Stabilization Techniques

Though the predictions of our model were fairly accurate, and perhaps sufficient for frequentist analysis, we cannot summarize the distributions of our parameters (and therefore our predictions) if our model has not converged.

3.3.1 Alpha Mixture Model

Upon seeing that our model was underconstrained, our first attempt to offer more regularization was to cluster our weights by creating a new hierarchical distribution for them.

Rather than giving each weight a unique element of α, we introduced a new latent variable for each weight, $r \in \{0, 1\}^{N+1}$ that indicates whether a weight is 'relevant' or not. We also introduced a parameter, p, denoting the likelihood that any weight is 'relevant', and we only had 2 elements in α, one for the 'relevant' cluster of weights, and one for the rest.

We therefore created a mixture model: our weights were either drawn from the 'relevant' distribution or the 'non-relevant' distribution. We hoped that this would impose more structure on the weights and prevent them from diverging.

We used a noninformative Beta(0.5,0.5) prior on p, since we did not want to constrain the optimal number of relevant vectors (though we expected it to be small). The model was thus:

$$\begin{aligned}
Y &= \texttt{sign}(z) \\
z &\sim \texttt{Norm}(\Phi w, 1) \\
w &\sim p\texttt{Norm}(0, \alpha_1^{-1}) + (1-p)\texttt{Norm}(0, \alpha_0^{-1}) \\
\alpha &\sim \texttt{Unif}(-\infty, \infty) \\
p &\sim \texttt{Beta}(0.5, 0.5)
\end{aligned}$$

with conditional posterior distributions:

$$\begin{aligned}
p(\alpha_i|w, z, r, p, Y, \Phi) &\propto \texttt{Gamma}(\frac{3}{2}, \sum_{j=0}^{N} \mathbb{1}_{r_i=j} \frac{w_i^2}{2}) \\
p(z_i|w, \alpha, r, p, Y, \Phi) &\propto \texttt{TruncNorm}_{Y_i}(\Phi w, 1) \\
p(w|\alpha, z, r, p, Y, \Phi) &\propto \texttt{Norm}(\hat{\beta}, \hat{\Sigma}) \\
p(r_i|w, \alpha, z, p, Y, \Phi) &\propto \frac{\texttt{Norm}(w_i; 0, \alpha_{r_i}^{-1})}{\sum_{j=0}^{1} \texttt{Norm}(w_i; 0, \alpha_j^{-1})} \\
p(p|w, \alpha, z, r, Y, \Phi) &\propto \texttt{Beta}(\sum_{i=0}^{N} r_i + 0.5, \sum_{i=0}^{N} (1 - r_i) + 0.5)
\end{aligned}$$

Where $\hat{\beta}$ and $\hat{\Sigma}$ are computed as before, except in place of our previous vector of $N+1$ elements for α, we use a vector of $N+1$ elements where element i equals α_0 if $Y_i = 0$ and α_1 if $Y_i = 1$.

Even without incorporating priors that would strongly differentiate the two values in α, we did observe that one value was very low while the other was very high, so it was interesting to note that the weights naturally clustered themselves into high and low variance 'relevance' clusters. For example, one of our chains produced alpha values of 1e-4, 6e+4. We also observed that weights were no longer diverging in a single direction.

However, we rejected this model, since weights still seemed to be diverging by oscillating at greater and greater magnitudes.

3.3.2 Stronger Alpha Priors

Looking more closely at the distribution for our α values, we noted that if a weight already had a high value, this would skew the distribution of its corresponding element of α closer to zero, meaning that the next draw of α would allow our weight to have even greater variance and stray even farther from 0.

We thus tried to incorporate stronger priors on our α values to skew them farther from 0 and reduce the variance in our weights. In order to continue to efficiently sample and avoid resorting to MCMC methods, we constrained our search to Gamma priors on α, which is a conjugate distribution. To offset the positive feedback loop of high-valued weights resulting in high variances, we increased the scale parameter to add more mass higher in the distribution. We also increased the shape parameter, but this was less effective.

Changing this prior did not influence the posterior distributions of any other parameters.

3.3.3 Truncated Alpha Prior

Using stronger Gamma priors still had limited success in shifting our α values away from 0, so we resorted to the even stronger strategy of truncating their prior distribution. This has the advantage of only scaling the supported region by a constant, so it does not change our equations for α significantly.

To allow our model to continue to make accurate predictions, we had to also slightly change the probit model. Since our weights are more strongly bounded, the values of Φw, which represent the mean of the distribution for z, are also more strongly bounded. We now allow the variance of z to change instead of keeping it fixed at 1, so that more sharper boundaries can be drawn between classes by decreasing the variance of z rather than increasing our weights to shift its mean. We use a noninformative prior $p(\sigma) \propto \sigma^{-1}$, which, though improper, we know will produce a familiar, proper posterior.

This takes us to our final model, as described above. We provide the new conditional distributions for w and z and a conditional distribution for σ^2:

$$\begin{aligned} p(z|w, \alpha, \sigma, Y, \Phi) &\propto \texttt{TruncNorm}_{Y_i}(\Phi w, \sigma^2) \\ p(w|\alpha, z, \sigma, Y, \Phi) &\propto \texttt{Norm}(\hat{\beta}, \hat{\Sigma}) \\ p(\sigma^2|w, \alpha, z, Y, \Phi) &\propto \texttt{Inv-ChiSq}(N, \tau^2) \end{aligned}$$

Where tau^2 is $\frac{1}{N}\sum_{i=1}^{N}(z_i - \Phi_{i,}.w)^2$, and

$$\begin{aligned} \hat{\Sigma} &= (\sigma^{-2}\Phi^T\Phi + \texttt{diag}(\alpha))^{-1} \\ \hat{\beta} &= \hat{\Sigma}\Phi^T\sigma^{-2}z \end{aligned}$$

The advantage of this was that we were primarily interested in preventing our weights from increasing without bound, so we strongly discouraged extremely high weights. This finally produced more reasonable weight values, which appeared to converge. We still observed that despite converging, the weights still did appear to switch between values, which we interpret to be an indication that our distribution has multiple modes. Given that values in Φ can be arbitrary, we thought this was reasonable, and it further justifies our approach of sampling the entire distribution, rather than using gradient-based approaches that could get stuck in local maxima.

Below, we provide histograms of the values of our weights, after truncating α to be above 10.

As a predictive check, we evaluated our model on several new datapoints and summarized the accuracy in the above table. This shows that our model makes accurate predictions even on new data. We also analyzed the probabilities that a label was a 1 or 0 on our points. We still mostly obtained values that were nearly 1 or 0; however, unlike the original model, we also obtained several predictions that were around 0.1 when predicting the red class. This suggests that the extra constraints of our model may allow it to better represent uncertainty in our data.

4 Conclusions

One limitation of our final model is the use of a strictly truncated prior. Future directions may investigate more priors to place over the α parameters in our model to better constrain it, perhaps in a more natural manner.

We have developed a framework for the Bayesian treatment of Relevance Vector Machines. Our model converges with little variance and high sparsity, and allows for feasible sampling. We achieved classification results beating the standard RVM and competitive with SVM. One further advantage of RVMs that was discussed in the original paper is the fact that they can use arbitrary kernel functions, unlike SVMs. In future work we would like to make our model kernel adjustable such that we would be able to choose kernel functions that better suit any expectations we have about the underlying data distributions we are working with.

The sampling solution for the Relevance Vector Machine is computationally more expensive than the type-II maximum likelihood approach. However, the advantages of a fully Bayesian approach may outweigh the costs where there is limited data, as they allow us to better assess model convergence, and they can allow us to better asses the confidence we should place in our predictions.

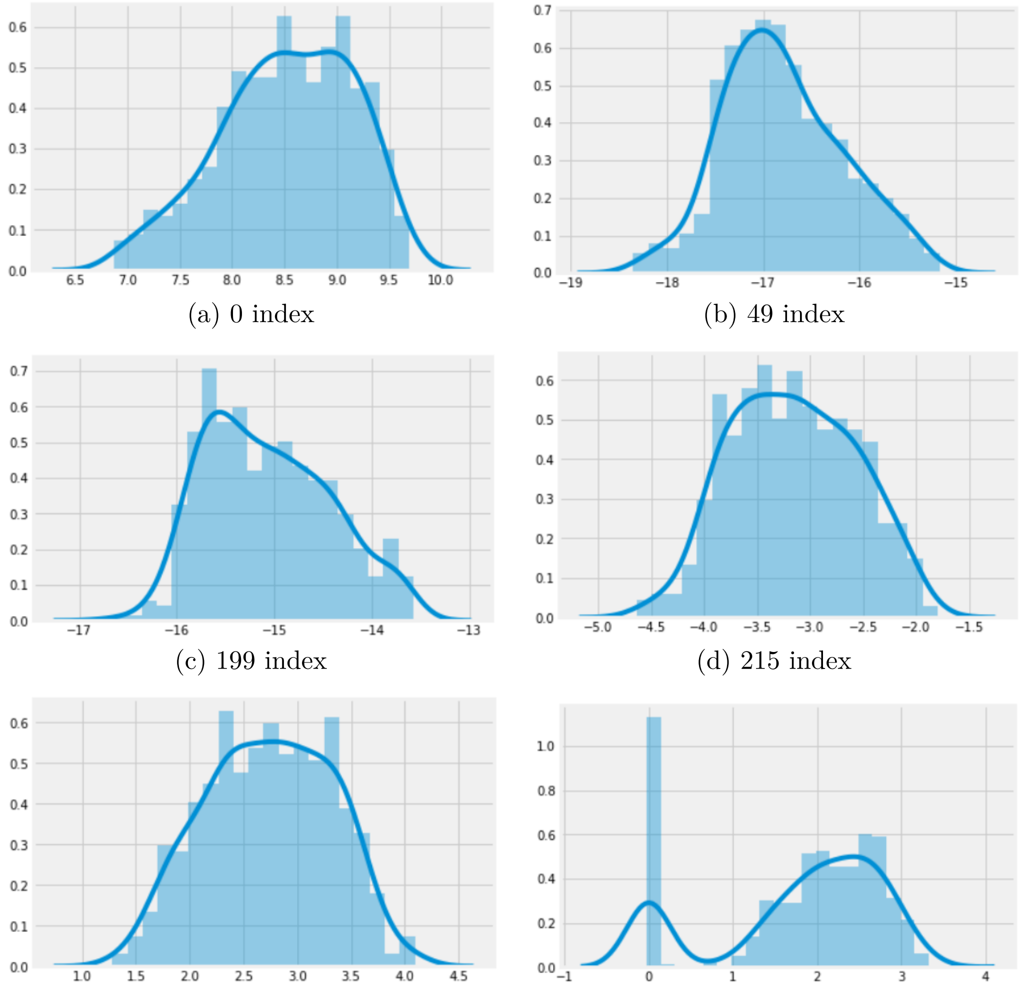
5 Citations

Tipping, Michael E. "The relevance vector machine." Advances in neural information processing systems. 2000.

Gelman, Andrew, et al. Bayesian data analysis. Chapman and Hall/CRC, 2013.

G. Tzikas, Dimitris Wei, Liyang Likas, Aristidis Yang, Yongyi. (2006). A *tutorial on relevance vector machines for regression and classification with applications. Eurosip. 17.*

https://archive.ics.uci.edu/ml/datasets/skin+segmentation



(a) 0 index

(b) 49 index

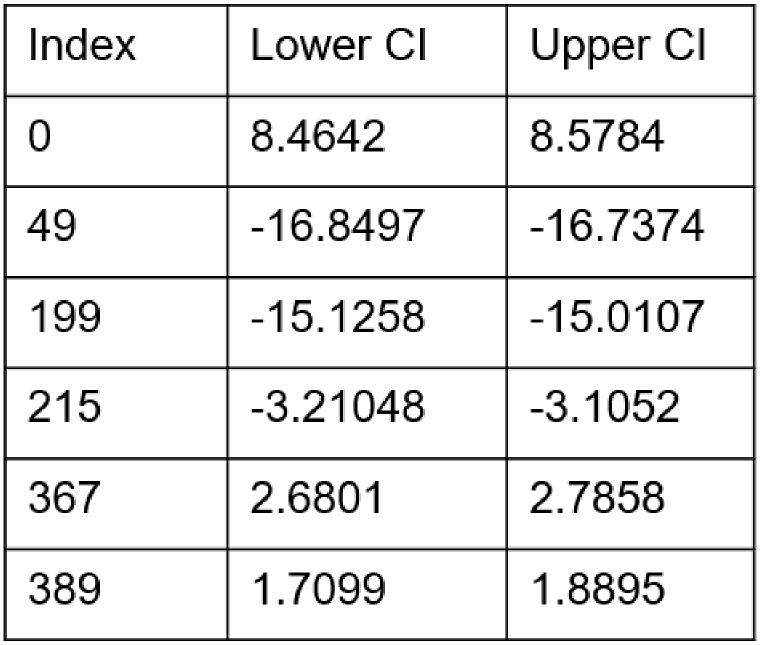
(c) 199 index

(d) 215 index

(e) 367 index

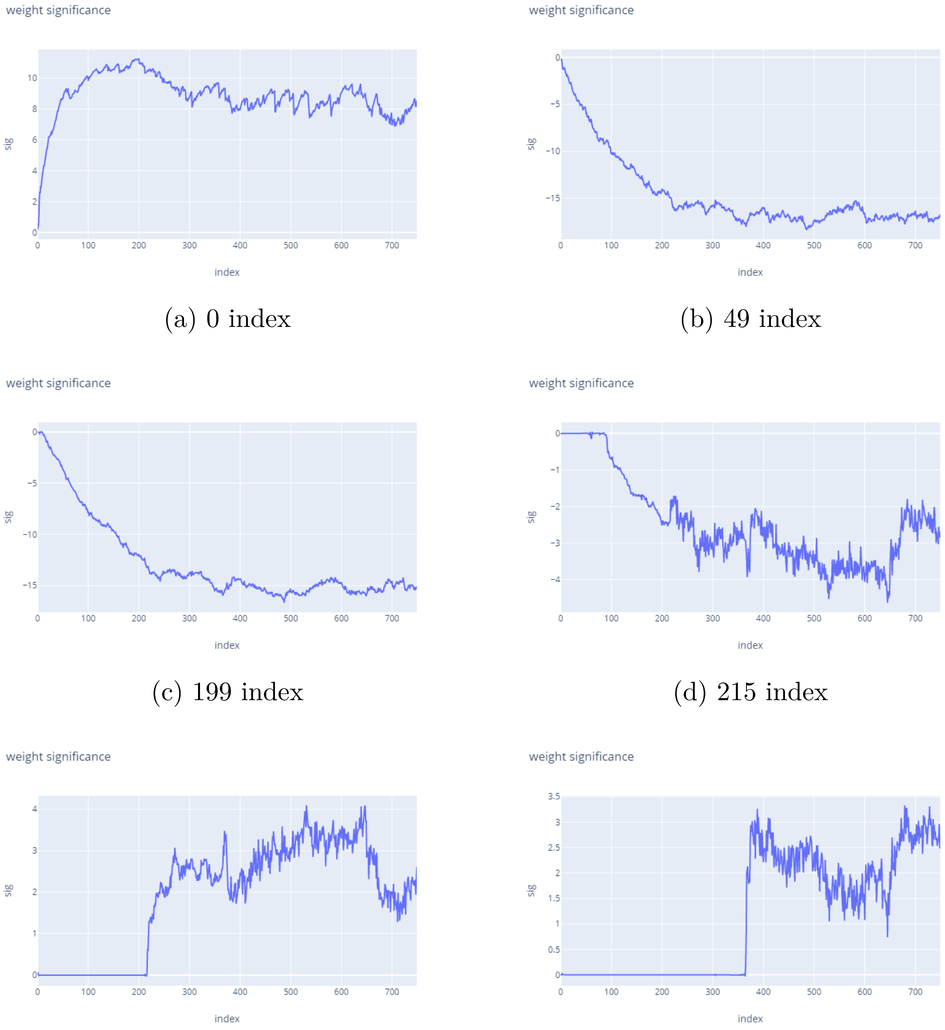
(f) 389 index

Figure 1: Histograms of variance of our weights past the 275 iteration



Index	Lower CI	Upper CI
0	8.4642	8.5784
49	-16.8497	-16.7374
199	-15.1258	-15.0107
215	-3.21048	-3.1052
367	2.6801	2.7858
389	1.7099	1.8895

Figure 2: Significant weights confidence intervals of weight distributions past 275 iteration



(a) 0 index

(b) 49 index

(c) 199 index

(d) 215 index

(e) 367 index

(f) 389 index

Figure 3: Significant weights convergence plots for truncated alpha

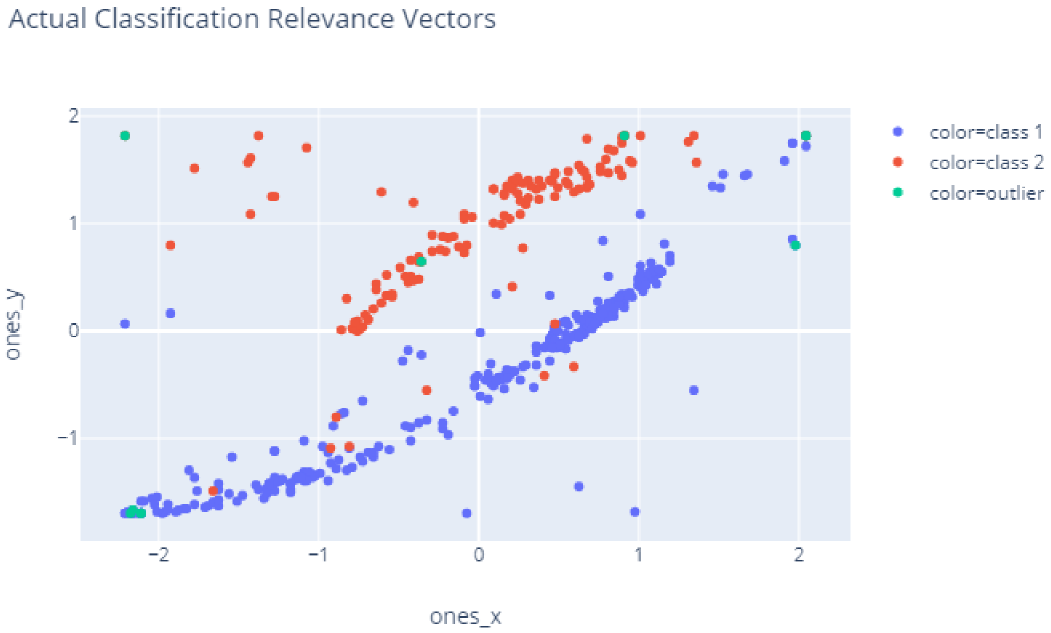


Figure 4: relevance vectors given significant weight overlayed on actual values

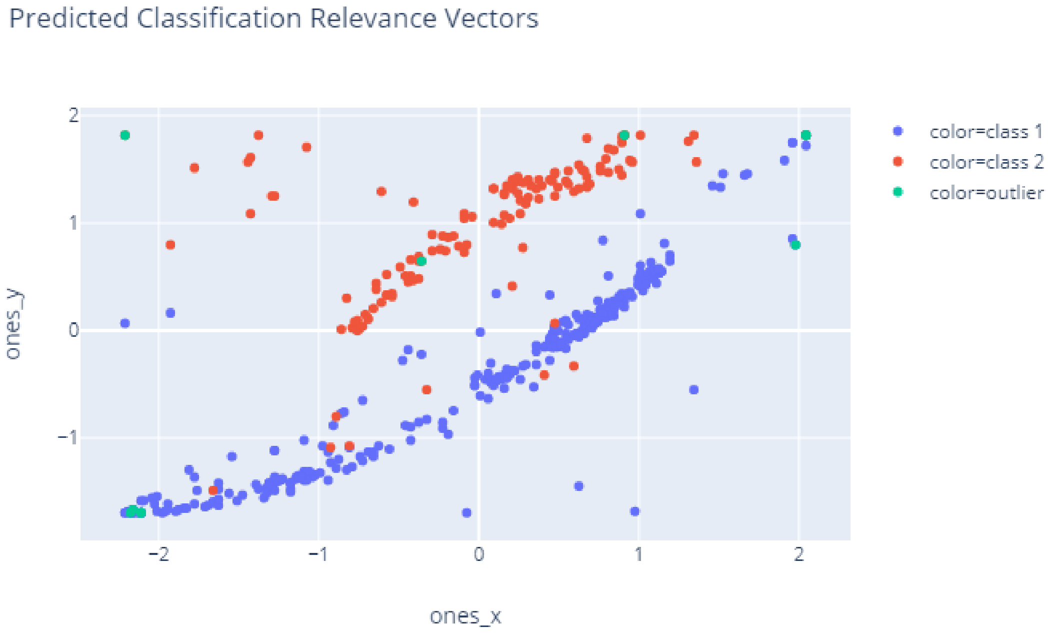
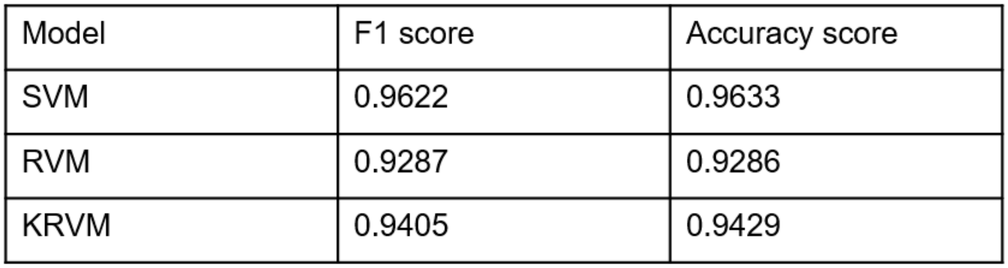


Figure 5: relevance vectors given significant weights overlayed on prediction values



Model	F1 score	Accuracy score
SVM	0.9622	0.9633
RVM	0.9287	0.9286
KRVM	0.9405	0.9429

Figure 6: Classification results achieved on Skin dataset using truncated alpha prior