Statistical Methods of Machine Learning
Assignment 2

Martin Grunbaum
Martin Jorgensen
Thomas Barnholdt

March 4, 2014

II.1.1

Linear Discriminant Analysis with $m > 2$ classes is done by creating a discriminant function for each class and running each function on all points we wish to classify. The discimination functions calculates the posterior probability that a point belongs to its class, and so the one with the highest probability is chosen. The discrimination function for a class k looks like

$$\delta_k(x) = x^{\mathrm{T}}\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^{\mathrm{t}}\Sigma^{-1}\mu_k + \ln \Pr(Y = C_k)$$

The prior distribution is calculated like

$$\Pr(Y = C_k) = \ell_k/\ell$$

Where ℓ is the number of elements in the training data, and ℓ_k is the number of elements of class k in the training data. μ_k is the mean of a given class, while Σ as the covariance matrix for each class added together and normalized. They're calculated like so

$$\begin{aligned}\mu_k &= \frac{1}{\ell_k}\sum_{(x,y)\in S_k} x\\ \Sigma &= \frac{1}{\ell - m}\sum_{k=1}^{m}\sum_{(x,y)\in S_k}(x-\mu_k)(x-\mu_k)^{\mathrm{T}}\end{aligned}$$

Where S_k is the points in the training set that corresponds to the class k and m is the number of classes.

Using this method we get an error of 14.00% on the training data and 21.05% on the test data.

II.1.2

The training error on the transformed data is 14% and the test error on the transformed data is 21.05%. This is exactly the same as it is on the un-transformed data, as the standardization of the data makes no difference for Linear Discriminant Analysis.

LDA discovers a number of functions which compute a posterior probability of the input, which in rough terms is equivalent to calculating the probability that a given input comes from that particular distribution (which is Gaussian in our case). In the end all that matters is which of the functions produce the largest posterior probability, which is a relative score compared to the other functions. Standardizing the data has no effect on this decision, as the separating bounds between the probabilities remain the same relative to one another, standardized data or no.

II.1.3

The Bayes optimal classifier is given as the formula

$$c_{\text{BayesOpt}} = \underset{c_j \in C}{\text{argmax}} \sum_{h_i \in H} \text{P}(c_j | h_i) \text{P}(h_i | S)$$

Where H is every hypothesis given and S is the training data.

In this assignment we're given two hypothesis "The label 1 shows up in three fourths of all trials." along with the training data $S = \{(0,0),(0,1),(0,1),(0,1)\}$ which reflects the hypothesis.

In our case, c_{BayesOpt} will always be that there is a bigger propability for the label being 1 than 0.

The Bayes optimal risk for this is 0.25 or 25%. This is the minimum risk over all the measurable hypothesis in H, the hypothesis with the lowest risk is that the label should be 1, which in our training set is true in 75% of the cases and erronous in the remaining 25%.

Now we look at the probabilistic clasifier, that predicts 0 with a probability of 0.25 and 1 with 0.75.

Of the 0.25 guesses on 0, 0.75 will be wrong. Of the 0.75 guesses on 1, 0.25 will be wrong. So the sum of wrong answers is:

$$0.25 \cdot 0.75 + 0.75 \cdot 0.25 = 0.375$$

So it would be better to always guess 1.

II.2.1

The RMS (Root Mean Square) error found on the test set for each of the three selections was:

$$RMS_1 = 35.4651$$
$$RMS_2 = 28.8398$$
$$RMS_3 = 18.7700$$

Of the three models, selection 3 clearly provides the best model, as can be seen both in Figure 4 and by the RMS of 18.7700.

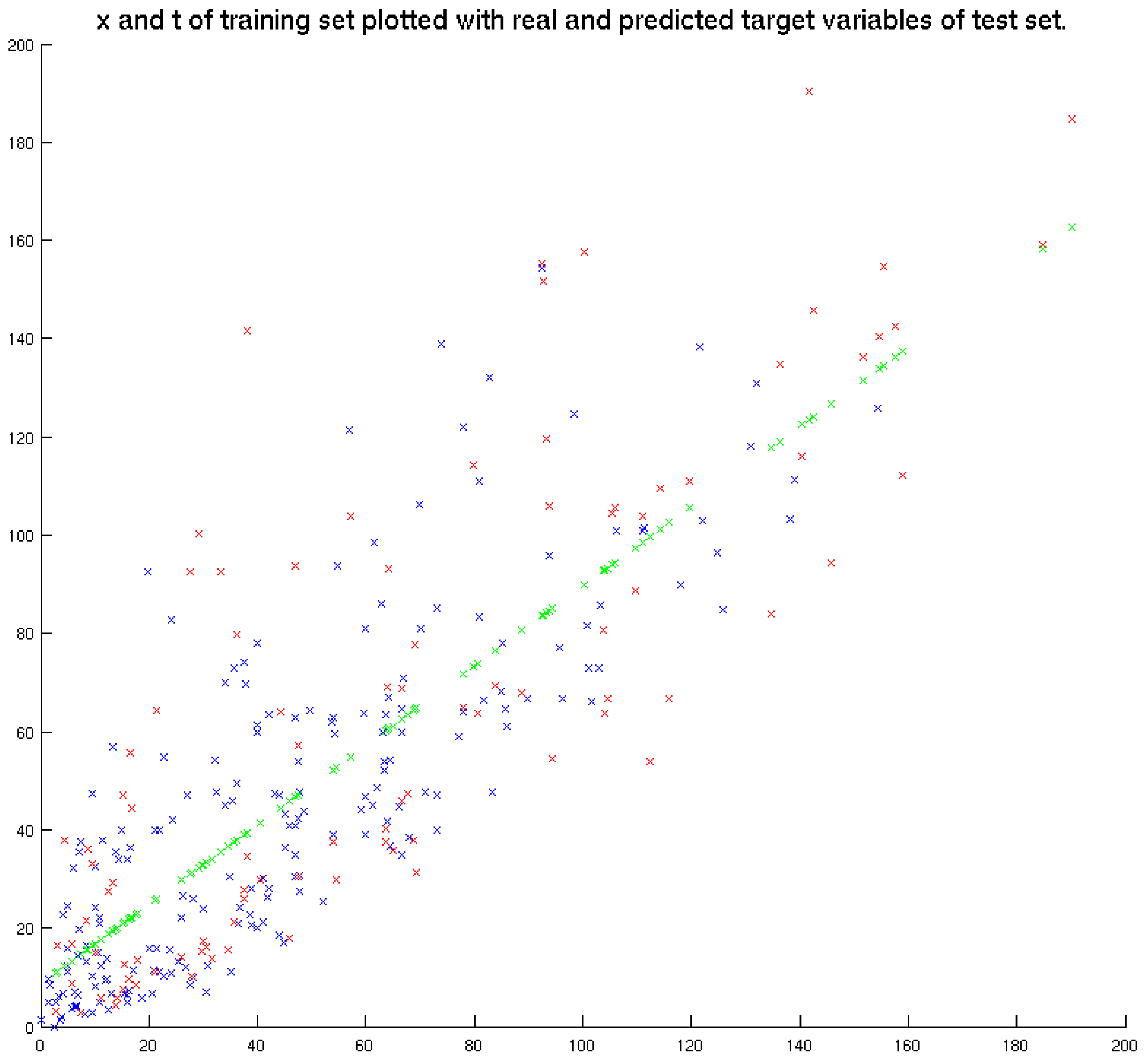

Figure 1: x and t of training set, plotted with real and predicted target variables.

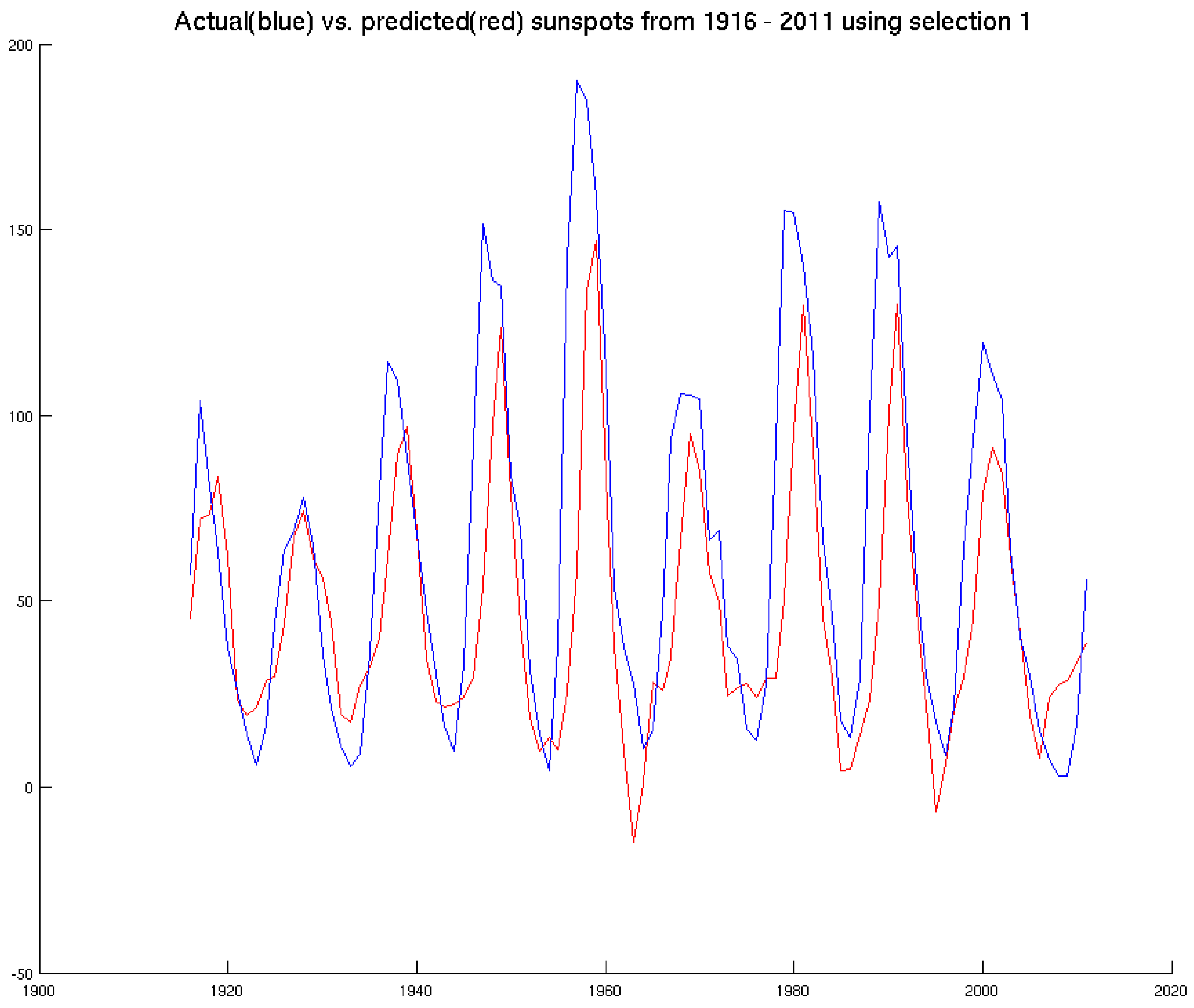

Figure 2: Years vs. predicted sunspot numbers, plotted with the true target values.

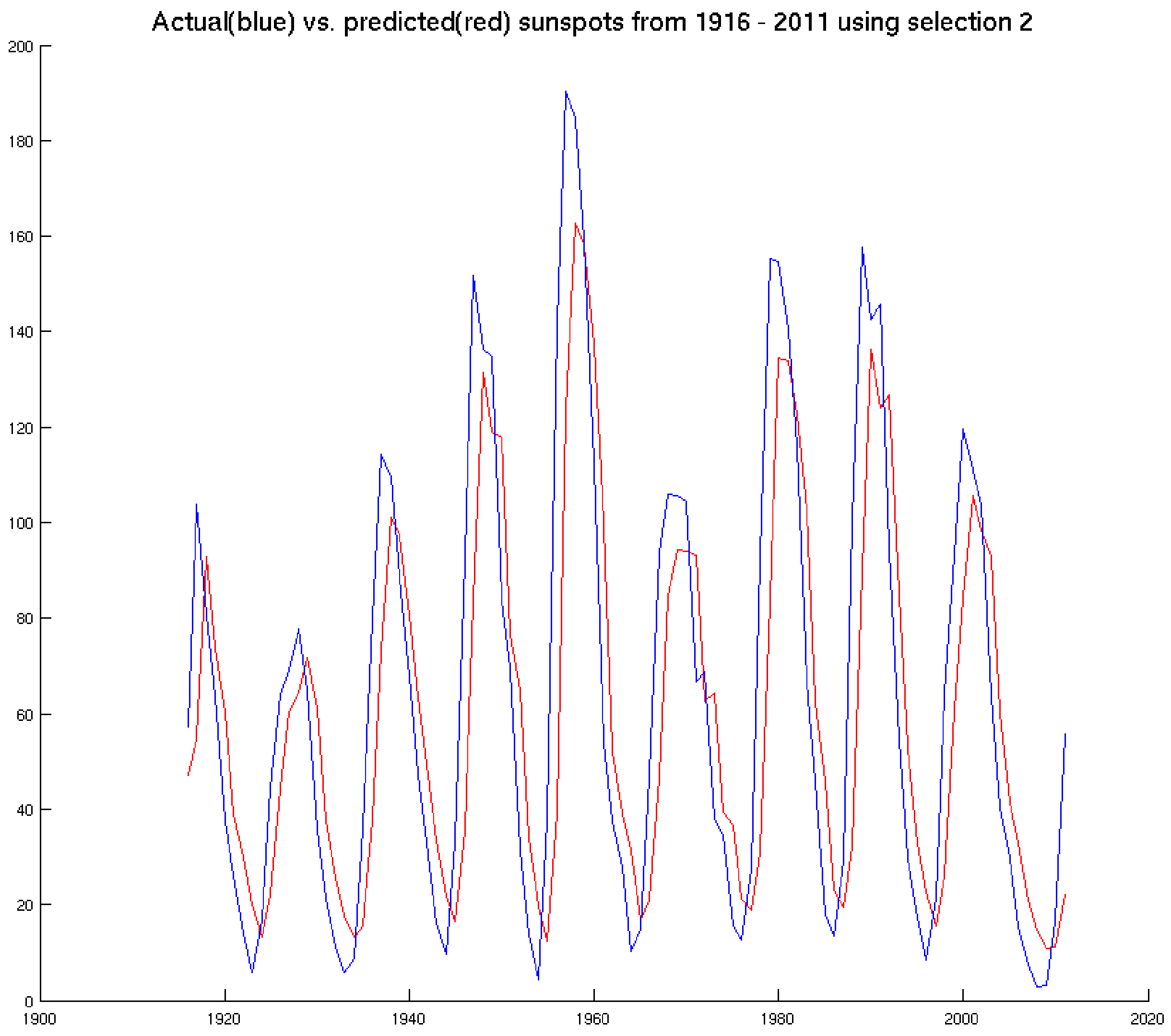

Figure 3: Years vs. predicted sunspot numbers, plotted with the true target values.

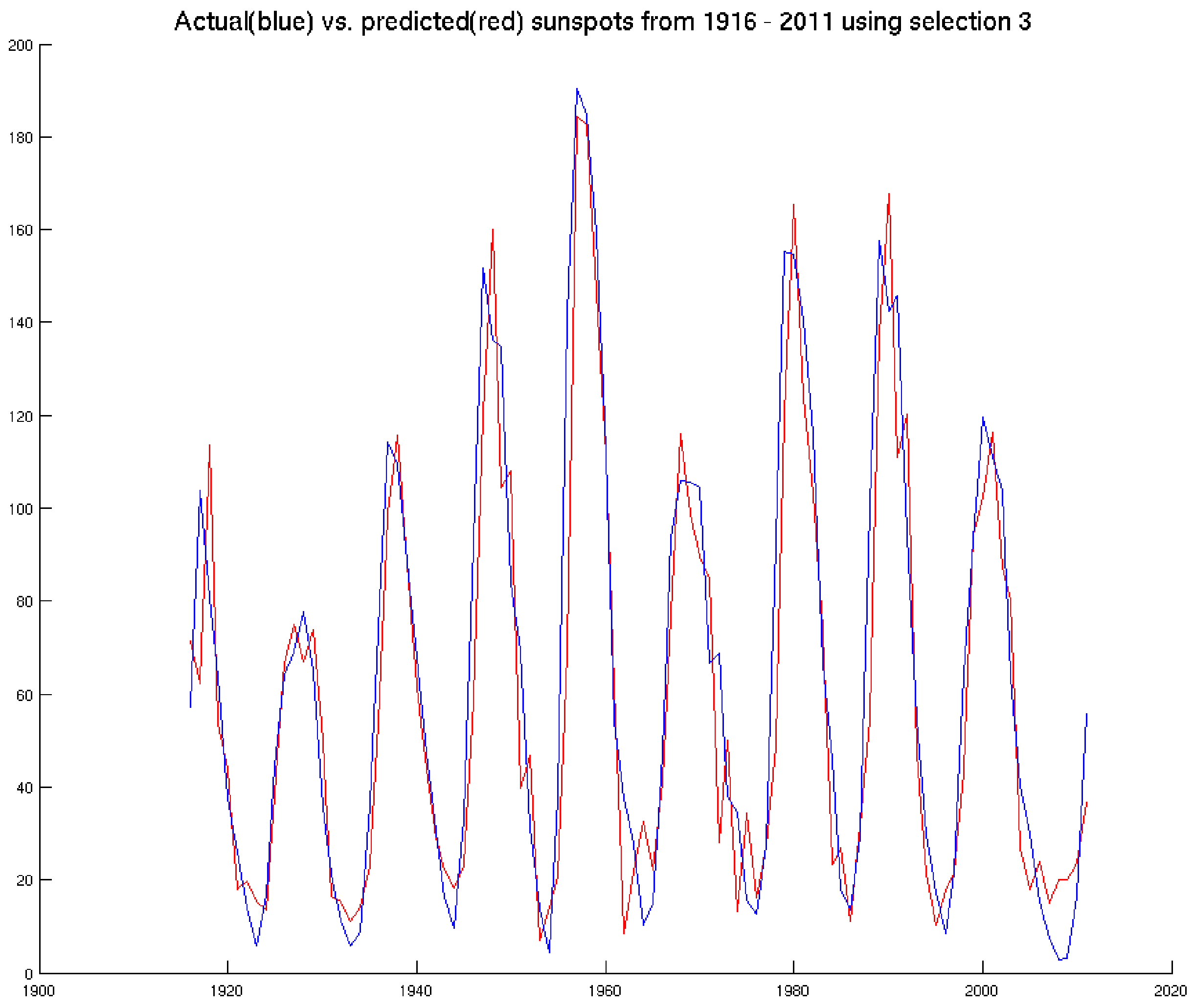

Figure 4: Years vs. predicted sunspot numbers, plotted with the true target values.

II.2.2

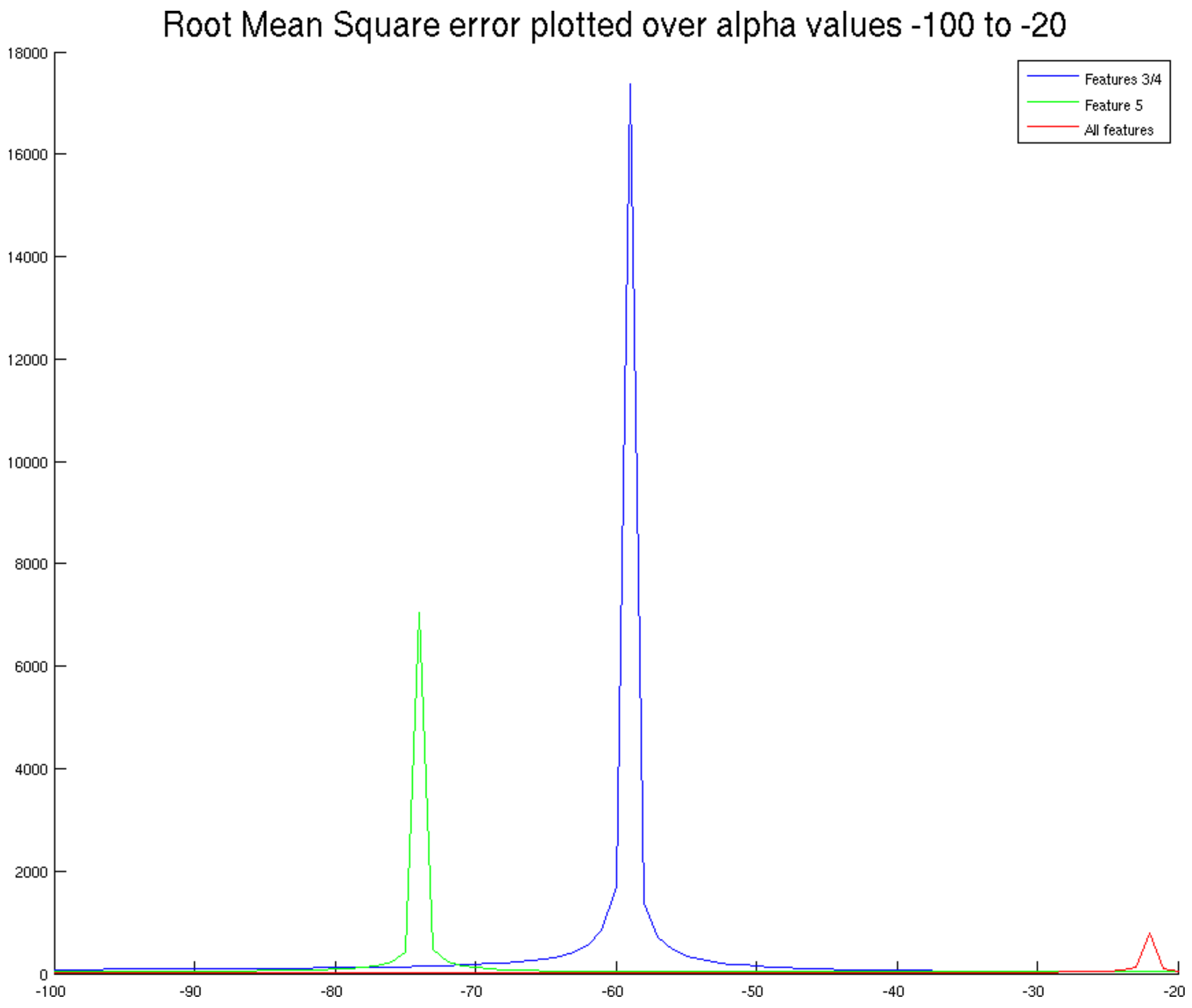

Figure 5: RMS of each selection plotted against particularly badly chosen α values.

The model for selection 3 clearly performs best again, for most values of α. We have been unsure of what range of values to try with α, and so we have tested all the way from -10000 to 10000. There are some particularly bad α-values in the negative range, at different points for the three different selections. Figure 5 shows these peaks specifically, while Figure 6 shows all 3 selections against the RMS obtained for that selection in II.2.1, for α values 1 to 500. For selection 3, testing for α-values between 0 and 10000, the RMS of selection 3 becomes lower for α values 0 to 7540.

In general, adjusting the alpha value seems to be a viable way to lower the prediction error; however, as Figure 5 shows, selecting bad α values can also result in disastrous prediction error.

II.2.3

We look at:

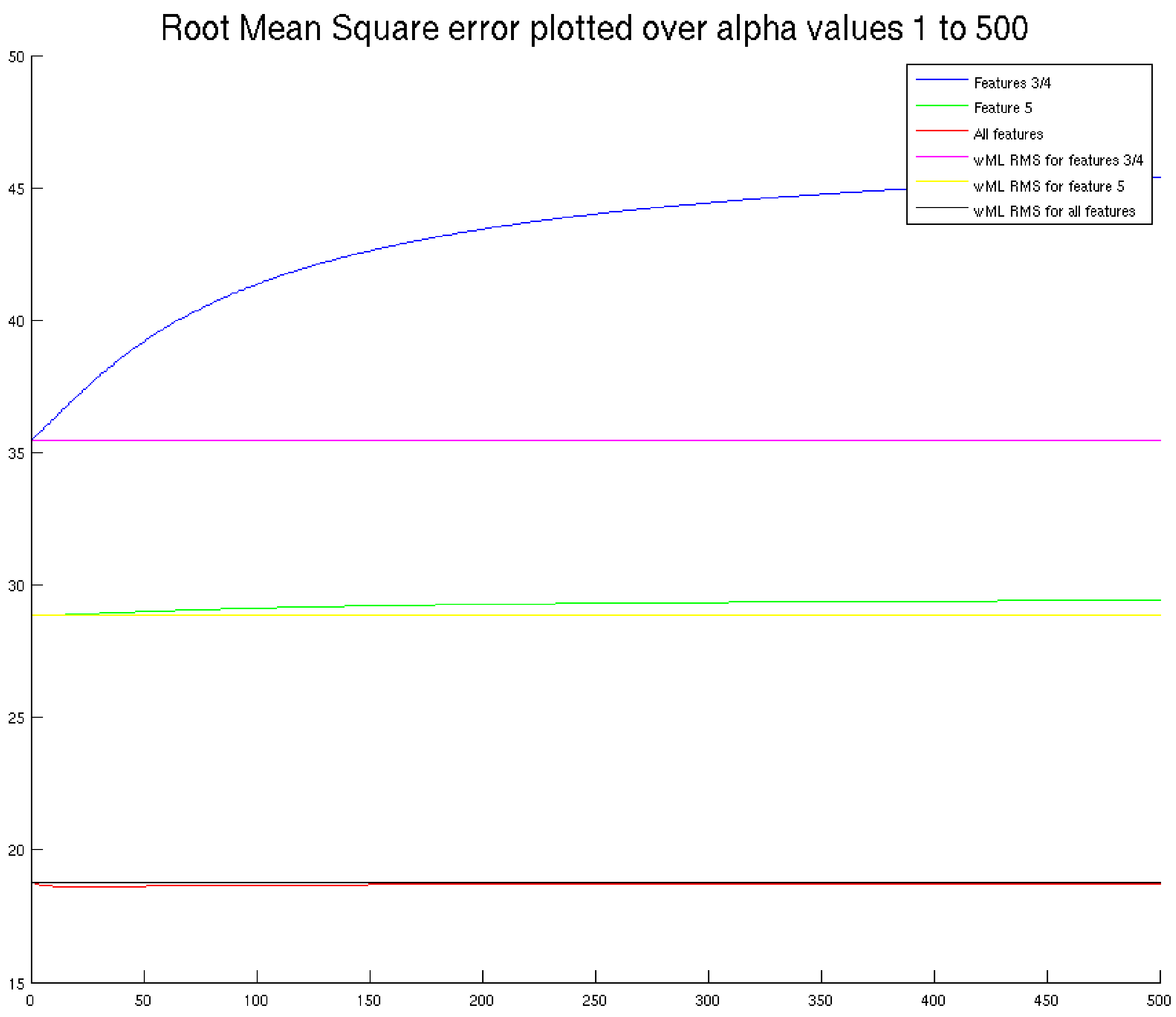

Figure 6: RMS of each selection and its corresponding w_{ML}-based model plotted against α values 1 to 500.

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} r_n\{t_n - \mathbf{w}^T\phi(\mathbf{x}_n)\}^2$$

$$= \frac{1}{2}\sum_{n=1}^{N} r_n \left\{ t_n - \sum_{m=1}^{|\mathbf{w}|} w_m\phi_m(\mathbf{x}_n) \right\}^2$$

To find $\mathbf{w}^*$ that minimizes this we find the partiel diffirentiated, and set it equal 0. Here we diffirentiate with respect to w_i, $0 < i < |\mathbf{w}|$:

$$\begin{aligned}(E_D(\mathbf{w}))' &= \left(\frac{1}{2}\sum_{n=1}^{N} r_n \left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\}^2\right)' \\ &= \frac{1}{2}\sum_{n=1}^{N} r_n \left(\left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\}^2\right)'\end{aligned}$$

Chain rule where:

$$\begin{aligned}f(x) &= x^2 \\ f'(x) &= 2x \\ g(\mathbf{w}) &= \left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\} \\ g'(\mathbf{w}) &= \left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\}' \\ &= -\phi_i(\mathbf{x}_n)\end{aligned}$$

So:

$$\begin{aligned}(E_D(\mathbf{w}))' &= \frac{1}{2}\sum_{n=1}^{N} r_n \left(2\left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\} \cdot -\phi_i(\mathbf{x}_n)\right) \\ &= \sum_{n=1}^{N} r_n \left(\left\{t_n - \sum_{m=1}^{|\mathbf{w}|} w_m \phi_m(\mathbf{x}_n)\right\} \cdot -\phi_i(\mathbf{x}_n)\right) \\ &= \sum_{n=1}^{N} \left\{-r_n t_n \phi_i(\mathbf{x}_n) + \sum_{m=1}^{|\mathbf{w}|} r_n w_m \phi_m(\mathbf{x}_n)\phi_i(\mathbf{x}_n)\right\}\end{aligned}$$

Setting equal to 0:

$$\begin{aligned}(E_D(\mathbf{w}))' &= 0 \\ \sum_{n=1}^{N} \left\{-r_n t_n \phi_i(\mathbf{x}_n) + \sum_{m=1}^{|\mathbf{w}|} r_n w_m \phi_m(\mathbf{x}_n)\phi_i(\mathbf{x}_n)\right\} &= 0 \\ \sum_{n=1}^{N}\sum_{m=1}^{|\mathbf{w}|} r_n w_m \phi_m(\mathbf{x}_n)\phi_i(\mathbf{x}_n) &= \sum_{n=1}^{N} r_n t_n \phi_i(\mathbf{x}_n) \\ \sum_{n=1}^{N} r_n \mathbf{w}^T \phi(\mathbf{x}_n)\phi_i(\mathbf{x}_n) &= \sum_{n=1}^{N} r_n t_n \phi_i(\mathbf{x}_n)\end{aligned}$$

As this count for every $0 < i < |\mathbf{w}|$, we can sum over every i:

$$\begin{aligned}
\sum_{i=1}^{|\mathbf{w}|}\sum_{n=1}^{N} r_n \mathbf{w}^T\phi(\mathbf{x}_n)\phi_i(\mathbf{x}_n) &= \sum_{i=1}^{|\mathbf{w}|}\sum_{n=1}^{N} r_n t_n \phi_i(\mathbf{x}_n) \\
\sum_{n=1}^{N} r_n \mathbf{w}^T\phi(\mathbf{x}_n)\sum_{i=1}^{|\mathbf{w}|}\phi_i(\mathbf{x}_n) &= \sum_{n=1}^{N} r_n t_n \sum_{i=1}^{|\mathbf{w}|}\phi_i(\mathbf{x}_n) \\
\sum_{n=1}^{N} r_n \mathbf{w}^T\phi(\mathbf{x}_n)\mathbf{1}^T\phi(\mathbf{x}_n) &= \sum_{n=1}^{N} r_n t_n \mathbf{1}^T\phi(\mathbf{x}_n)
\end{aligned}$$

Which is something of the form:

$$w_1 \cdot C_1 + w_2 \cdot C_2 + \ldots + w_{|\mathbf{w}|} \cdot C_{|\mathbf{w}|} = K$$

which yields a family of vectors $\mathbf{w}$ minimizing the error function.

Intuition about weighted sum-of-squares:

(i) Data dependent noise variance: If we have domain knowledge then we can encode the domain knowledge into our error function by assuming some data-dependent level of noise for each individual sample.

(ii) Replicated data points: If the data contains replicated data points, these risk becoming overly important for regular sum-of-squares. With weighted sum-of-squares we can proportionally reduce the importance of replicated points.

References