Splines for Regularization and Simple Speech Recognition

Abdul Saleh[*] **Dean Hathout**[*]
Harvard College
`{abdelrhman_saleh, dhathout}@college.harvard.edu`

Abstract

This project explores using smoothing splines for regularizing logistic regression models. We apply spline-regularized logistic regression to a simple speech recognition task. We find that adding spline-regularization improves classification accuracy outperforming all baselines. We also find interesting (empirical) connections between L_2 and spline regularization.

1 Introduction

Many of the models we have encountered in this course have assumed that the response variable is linear in its predictors. These models are convenient and easy to interpret, but in many cases it is highly unlikely that the relationship between the response and predictors is truly linear.

Basis expansions transform the predictors to capture non-linear patterns, achieving more flexible predictive models. Splines are a special type of basis functions that fit piecewise polynomials with smoothness restrictions over different regions of the predictors. Splines enforce smoothness by requiring continuity of derivatives where any two polynomials meet (figure 1).

In addition to their predictive power, splines can also be used for regularization. Fitting splines on the *coefficients* of other predictive models amounts to putting smoothness constraints on how the coefficients vary from one feature to another. Enforcing the smoothness of coefficients can improve model performance if **1)** the features are ordered and **2)** we have reason to believe that the coefficients should not vary significantly from one feature to the next. For example, this applies in speech recognition where it is expected for similar sound frequencies to have similar coefficients.

[*] Equal contribution

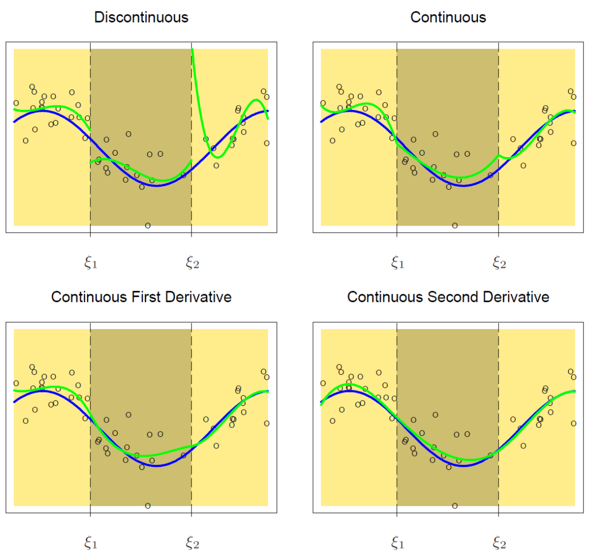

Figure 1: Example of cubic piecewise polynomials with increasing orders of continuity (Hastie et al., 2001).

We demonstrate the regularizing effect of splines by applying spline-regularized logistic regression to distinguish between one-second long utterances of ”Yes” and ”No”. We also evaluate the performance of more complex models such as random forest classifiers and support vector machines on the same task for reference.

We hypothesize that spline-regularized logistic regression will outperform more complex models because of the prior knowledge introduced into the regularized model as restrictions on the coefficients. Although random forests and support vector machines have been shown to outperform logistic regression on many tasks (Couronn et al., 2018; Salazar et al., 2012), we predict that without smooth coefficients, these powerful models will easily overfit the training data hurting their performance.

2 Background

Fitting piecewise polynomials is sometimes preferable to fitting a global polynomial. Low-order polynomials might not have the capacity to fit complex patterns. At the same time, high order polynomials are prone to ”flapping” about outside of the given data range. (figure 2) In this section, we describe how to fit splines which are functions defined by piecewise polynomials.

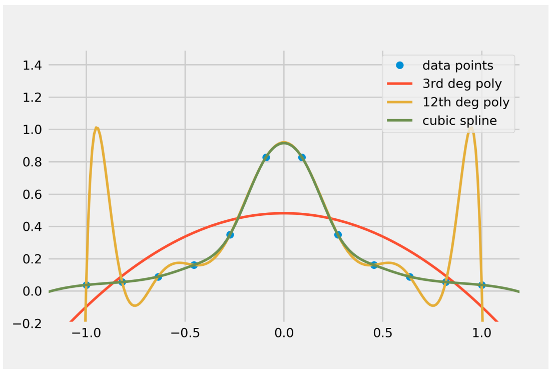

Figure 2: Example where splines are preferable to global polynomials.

2.1 Linear Splines

Suppose we have n data points $\{(x_i, y_i), i = 1, ..., n\}$. Let $X_{N\times 1} = (x_1, ..., x_n)$ be our design matrix and let $Y_{N\times 1} = (y_1, ..., y_n)$ be our response variable. To fit a spline modelling $Y = f(X)$, the domain of X is divided into intervals which meet at the knots $\xi_1, ..., \xi_k$. Within each interval we represent f by the basis functions, $h_i(X)$. For example, in the top left panel of figure 3, we see a piecewise constant polynomial, which has these basis functions:

$$h_1(X) = I(X < \xi_1),\ h_2(X) = I(\xi_1 < X < \xi_2),$$

$$h_3(X) = I(X > \xi_3)$$

where I is constant within the given range and zero otherwise.

A note on basis functions: Basis functions can be considered transformations on the predictors. So in the case above our model takes the form $Y = \sum_{m=1}^{3} \beta_m h_m(X)$ rather than $Y = \beta_1 X$.

To define the piecewise linear function shown in the top right panel of figure 3, three basis functions are added:

$$h_4(X) = h_1(X)X, \quad h_5(X) = h_2(X)X$$

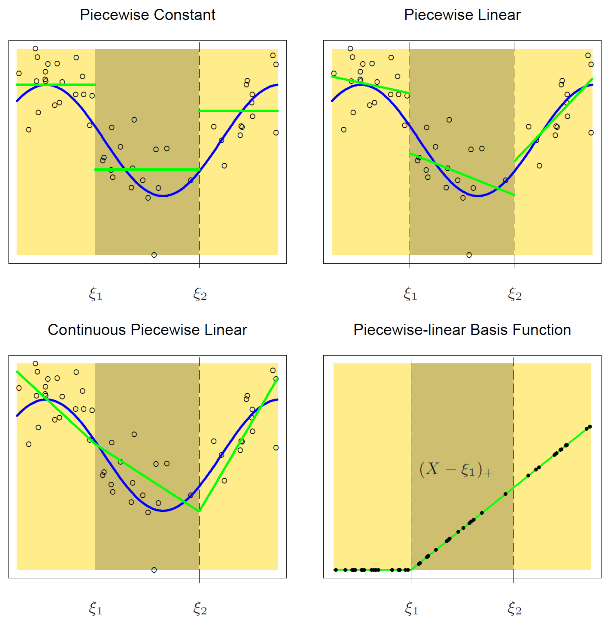

Figure 3: Piecewise Constant and Linear Functions (Hastie et al., 2001).

$$h_6(X) = h_3(X)$$

Each of the new basis functions h_i (corresponding to an additional column of the design matrix) is associated with a β_i so we can model both intercepts and slopes with $\beta_i X$ terms.

2.2 Enforcing Continuity

We can enforce first order continuity with the help of truncated polynomials of degree one represented by $(x - \xi_k)_+^D$, with $D = 1$:

$$(x - \xi_k)_+^D = \begin{cases} 0 & x < \xi_k \\ (x - \xi_k)^D & x \geq \xi_k \end{cases}$$

It follows that we can represent the model in the lower left panel of figure 3 by these basis functions:

$$h_1(X) = 1, \qquad h_2(X) = X,$$

$$h_3(X) = (X - \xi_1)_+,\ h_4(X) = (X - \xi_2)_+$$

Truncated polynomials ensure that polynomials on both sides of ξ_i intersect at $X = \xi_i$. Again, each basis function h_i is associated with a coefficient β_i.

2.3 Cubic Splines

We can extend the ideas presented above to non-linear splines (namely cubic ones). We do this by fitting cubic polynomials between the knots with additional continuity restrictions on the **first** and **second** order derivatives.

We consider the same setup as before. In this case, the truncated polynomial basis for the cubic spline is comprised of the following functions:

$$h_1(X) = 1,\ h_3(X) = X^2,\ h_5(X) = (X - \xi_1)^3_+$$

$$h_2(X) = X,\ h_4(X) = X^3,\ h_6(X) = (X - \xi_2)^3_+,$$

The first four basis functions, which span the entire domain of X, allow any cubic polynomial to be fit in the region to the left of the first knot. Adding the extra cubic terms in $h_5(X)$ and $h_6(X)$ allow polynomials to be fit in the regions to the right of the first and second knots.

Moreover, it is not difficult to see that the truncated polynomial basis above enforces continuity on the first and second order derivatives. Taking first and second derivatives of the truncated polynomial terms yield terms that still meet at the knots ξ_i. As a result, the truncated polynomials in the derivative equations ensure that the derivatives are equal at $X = \xi_i$.

2.4 General Splines

The general non-linear spline follows naturally from the cubic case. In general, a spline of order M with K knots $\xi_1, ..., \xi_K$ is a piecewise polynomial of order M (degree $M - 1$) and continuous derivatives to order $M - 2$. The truncated polynomial basis for the order-M spline is simply:

$$h_j(X) = X^{j-1},\ j = 1, ..., M$$

$$h_{M+l}(X) = (X - \xi_l)^{M-1}_+,\ l = 1, ..., K$$

Continuity up to the first $M-2$ order derivatives follows by the same logic described above in the cubic case.

2.5 Degrees of Freedom

We notice that after enforcing continuity when fitting the linear spline above, we went from needing six basis function to only needing four. Similarly, we only need six basis functions to fit the cubic spline, even though we fit a separate cubic polynomial (which typically takes four parameters) in each of the three regions partitioned by the knots ξ_1 and ξ_2. This is due to the continuity restrictions.

With the linear spline, the slope and intercept of the linear function for the region $X < \xi_1$ determine the intercept for the region $\xi_1 < X < \xi_2$ since the lines meet at the knots. Similarly, the slope and intercept of the linear function in the region $\xi_1 < X < \xi_2$ determine the intercept for the region $\xi_2 < X$. Thus, we are left with only four parameters which is equal to the number of basis functions we found for this spline.

This idea extends to non-linear splines as well, with the general rule being that each continuity restriction at each knot yields one extra degree of freedom (takes away one parameter to be fit). So, for the cubic spline example above, there are only six parameters to fit; there are initially twelve parameters – four for each cubic polynomial in the three regions defined by the knots at ξ_1 and ξ_2 – then at each of the two knots, there are three continuity restrictions imposed: the function itself, its first derivative, and its second derivative are forced to be continuous. Three restrictions at each of two knots takes away 3 x 2 = 6 parameters, yielding 12 - 6 = 6 parameters, which equals the number of basis functions we found for this spline.

Similarly, the general order-M spline has $M + K$ parameters; there are $(K+1)M$ parameters initially – M for each degree $M-1$ polynomial in the $K+1$ regions defined by the K knots. Then, there are $M - 1$ continuity restrictions imposed at each of the k knots: the function itself and its first $M-2$ derivatives are forced to be continuous. $M - 1$ restrictions at each of K knots takes away $(M-1)K$ parameters, yielding $(K + 1)M - (M - 1)K = M + K$ parameters, which equals the number of basis functions we found for the general spline.

2.6 Spline Regression & Design Matrices

With the general truncated polynomial basis functions above, it is straightforward to see that for a spline regression (with only one predictor X), the design matrix is

$$\begin{bmatrix} 1 & X_1 & \dots & X_1^{(M-1)} & (X_1 - \xi_1)_+^{(M-1)} & \dots & (X_1 - \xi_K)_+^{(M-1)} \\ 1 & X_2 & \dots & X_2^{(M-1)} & (X_2 - \xi_1)_+^{(M-1)} & \dots & (X_2 - \xi_K)_+^{(M-1)} \\ & & & \vdots & & & \\ 1 & X_n & \dots & X_n^{(M-1)} & (X_n - \xi_1)_+^{(M-1)} & \dots & (X_n - \xi_K)_+^{(M-1)} \end{bmatrix}$$

The resulting model has coefficients β_i for $i = 0, 1, ...M + K - 1$. These coefficients can be calculated by treating this as an ordinary least squares regression problem while using the transformed design matrix instead of the original features.

2.7 Smoothing Splines & Regularization

At this point, one natural question which arises is that of how many knots to introduce in the domain of X and where to place them. Smoothing splines addresses this problem by simply using a maximal set of knots, i.e. introducing a knot at each data point. Doing so leads to a complex fit. But we can introduce a fixed smoothing parameter, λ, that penalizes curvature to control for this added complexity. In particular, the fit spline f is the one which minimizes a penalized RSS loss function:

$$\text{RSS}(f,\lambda) = \sum_{i=1}^{N}\{y_i - f(x_i)\}^2 + \lambda\int\{f''(t)\}^2 dt$$

2.8 Logistic Regression

Let Y_i be a binary response variable to be predicted from a vector of n features, $X_i = (x_1, x_2, ..., x_n)$ for the i^{th} datapoint. We can fit a logistic regression model to predict the posterior probability of $Y_i = 1$ of this form:

$$\text{P}(Y = 1_i|X_i) = \frac{1}{1 + \exp(-\hat{\theta}^T X_i)} \quad (1)$$

were $\hat{\theta}$ is a vector of n parameters, $\theta_1, ..., \theta_n$, learned through iterative numerical methods maximizing the likelihood function, $p(X;\theta) = \text{P}(Y = 1|X,\theta)$, with respect to θ.

3 Approach

3.1 Spline Regularization

For small training datasets with a large number of features n, the components of θ can fluctuate significantly and have large magnitudes. When this is the case, logistic regression models become too dependent on certain features and make unjustified confident predictions leading to overfitting.

Under some assumptions, we can prevent logistic regression models from overfitting by using smoothing splines. In the following sections, we demonstrate this regularizing effect by training a logistic regression model to distinguish between one-second utterances of ”Yes” and ”No”.

For each one-second clip, the power of the signal was measured at 128 frequencies to generate the input features (figure 4). The input features represent an estimate of the spectral density of the signal. The details of this transformation is beyond the scope of this project. However, it is sufficient to note that each component represents the power of a specific frequency in the sound clip.

For spline regularization to be effective, two conditions need to apply:

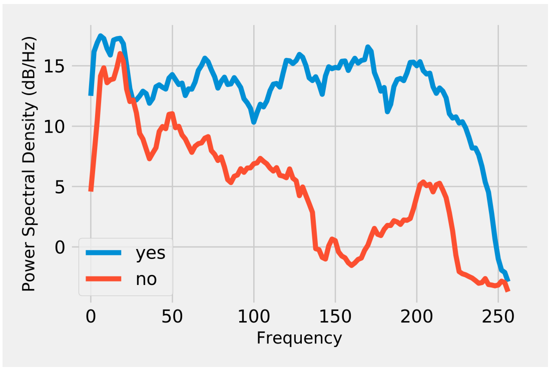

Figure 4: Spectral density plots of ”Yes” and ”No” utterances.

1. The features are ordered

2. It should make sense for the coefficients, θ_i, to not vary significantly from one feature to the next.

Both of these conditions apply when using spectral density estimates for speech recognition. The features in the i^{th} clip, X_i, are in ascending order based on the corresponding frequency, and it makes sense for coefficients of similar frequency to have similar values since they sound the same.

After fitting a regularized regression model to this task, we expect to get θ_i's that fluctuate significantly from one frequency position to the next (blue line in figure 5).

We apply spline-regularization by fitting a smoothing spline on the *coefficients*, θ_i, of the trained logistic regression model (red line in figure 5). This allows us to restrict the coefficients to vary smoothly from one frequency to the next. By restricting the coefficients in this way, we are introducing ”prior” knowledge (in a non-Bayesian sense) we have about the prediction task into the model. This approach also simplifies the model and stabilizes the coefficients mitigating the effect of overfitting.

3.2 Other Models

We also evaluate the performance of support vector machines, random forests, and ridge regression on this task for reference. For more information regarding these algorithms, please refer to Hastie, Tibshirani, and Friedman (2001).

	Test				Train			
Method	Accuracy	Precision	Recall	F1	Accuracy	Precision	Recall	F1
Majority Baseline	50.0	50.0	100.0	66.7	50.0	50.0	100.0	66.7
SVM	91.0	94.2	87.4	90.1	100.0	100.0	100.0	100.0
Random Forest	90.6	92.3	88.6	90.4	99.0	100.0	98.0	99.0
Vanilla Logistic	88.9	88.3	92.0	90.1	100.0	100.0	100.0	100.0
Ridge Logistic	**92.4**	**90.8**	**94.4**	**92.5**	98.8	99.6	98.0	98.8
Spline-Logistic	91.7	90.6	93.0	91.8	94.6	96.3	92.8	94.5

Table 1: Training and test set results summary in %. Best results in **bold**.

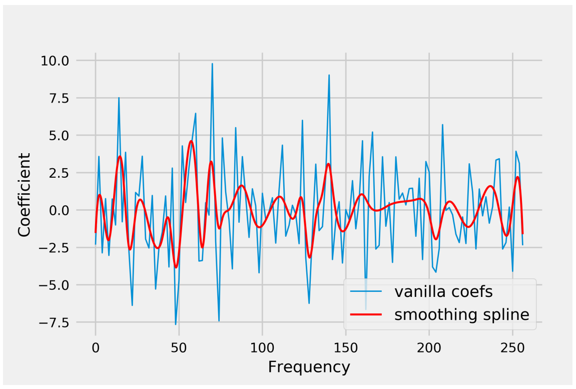

Figure 5: Plot of coefficients vs frequency, before and after spline-regularization.

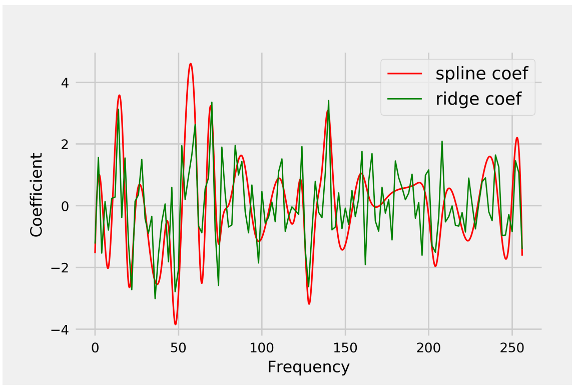

Figure 6: Plot showing similarity between spline-regularized coefficients and ridge regression coefficients.

4 Experiments and Results

4.1 Dataset

We used the Google Speech Commands Dataset (Warden, 2018) for our experiments. We randomly sampled 500, 1000, and 1000 one-second utterances for training, validation, and testing respectively (2500 in total). The datasets were balanced with half of the utterances being ”Yes” and the other half being ”No”. We generated the features using the methods described in Section 3.1.

4.2 Results

The results are summarized in table 1. We chose models based on validation set performance. Some naive hyperparameter tuning was done for all models. We call the unregularized logistic regression model Vanilla Logistic.

The experiments agree with our intuition and we find that SVMs and Random Forests easily overfit to the training set, even after hyperparameter tuning on the validation set. Despite the vanilla model being much simpler than SVMs and Random Forests, it still severely overfit the training set achieving an accuracy, precision, recall, and F1 of 100%, and much significantly worse results on the test set. In other words, it was able to correctly classify all the sound-clips in the training set, but was very sensitive to deviations from the data on which it trained.

We fit a smoothing spline on the vanilla model coefficients and select λ based on validation set performance. As hypothesized, spline-regularization improved absolute validation accuracy by 2.8%, which corresponds to a 25.2% relative reduction in error rate. Spline-regularization also produces a model that outperforms the SVM and Random Forests baselines in terms of accuracy and F1 score.

Ridge logistic regression achieved the best performance in our tests in terms of all accuracy metric considered. This model required applying a reasonably large amount of shrinkage to avoid overfitting and improve validation set performance. We also experimented with applying spline-regularization on the ridge regression coefficients to smooth them. However, this hurt performance so we did not include it in our results.

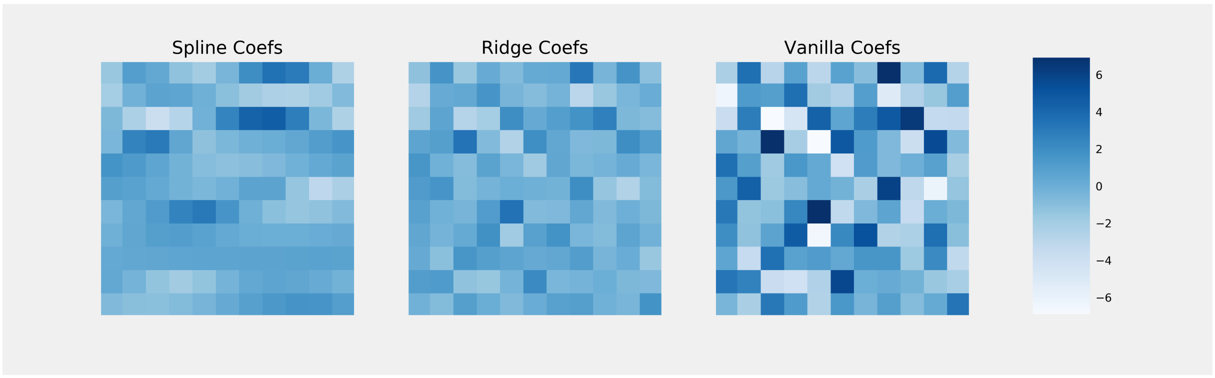

Figure 7: Map of Coefficients in Spline-Regularized, Ridge, and Vanilla Logistic Regression.

5 Analysis

The above results show that fitting a smoothing spline on the coefficients of logistic regression models can serve as an effective regularization technique. We plot the vanilla coefficients against the spline-regularized coefficients and see that they behave as expected in figure 5. The plots show a clear shrinkage in the coefficients compared to the coefficients from the vanilla model. Instead of the counter-intuitive oscillations seen in the vanilla coefficients (we do not expect similar consecutive frequencies to have wildly different coefficients), the spline-regularized model yields a much smoother map of coefficients which do not vary as severely from frequency to frequency.

We also investigate the relationship between ridge and spline regularization further. We do this by directly comparing the spline-regularized coefficients with the ridge coefficients in figures 6 and 7, and observe a high degree of similarity. Overall, it appears that both methods find roughly the same set of significant frequencies and shrink the frequency coefficients by similar degrees, showing that smoothing splines can have a regularizing effect quite similar to ridge.

Spline regularization appears to have a slightly more pronounced smoothing effect over the coefficients (almost making it appear in figure 6 that the spline-regularized coefficients are a smoothed version of the ridge coefficients, though this is not the case), with less drastic oscillation in coefficient values from frequency to frequency. We see this again in figure 7 – the spline coefficients appear to be more smoothly distributed with peaks and lows matching the ridge coefficients. This agrees with our intuition that similar frequencies would have similar coefficients.

6 Conclusion

We explored the use of splines as a regularization method for logistic regression models. Our findings show that spline-regularized logistic regression outperforms vanilla logistic regression under specific conditions. We demonstrate how splines can be used to improve performance in a simple speech recognition tasks and find interesting similarities between ridge regression and spline-regularization in terms of their effects on the coefficients.

Future work could explore theoretical connections between ridge regression and spline-regularization. We have also not studied the robustness of spline-regularization when its assumptions are not satisfied. It would also be interesting to extend this regularization method to a multiple predictor setting and test its effect on performance.

References

Raphael Couronn, Philipp Probst, and Anne-Laure Boulesteix. 2018. Random forest versus logistic regression: a large-scale benchmark experiment. *BMC Bioinformatics*, 19(1).

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2001. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA.

Diego Salazar, Jorge Vlez, and Juan Salazar. 2012. Comparison between svm and logistic regression: Which one is better to discriminate? *Revista Colombiana de Estadstica*, 35(2):223–237.

Pete Warden. 2018. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209.