Advanced Regression Topics

Frank Wood

October 29, 2009

Introduction

- Overview of Topics

Bayesian Regression

- Linear Regression

Poisson Regression

- Neural Modeling Example
- Football Goal Scoring Rate Example

Generalized Linear Models

Advanced Regression Topics

- Bayesian regression
- Weighted / regularized least squares
- Mixtures of Experts
- Gaussian process regression

Some parts of this lecture are taken from Bishop's book *Pattern Recognition and Machine Intelligence*

Advanced Regression Topics

- Bayesian regression
- Weighted / regularized least squares
- Mixtures of Experts
- Gaussian process regression
- Logistic regression
- Poisson regression
- Generalized linear models

Some parts of this lecture are taken from Bishop's book *Pattern Recognition and Machine Intelligence*

Advanced Regression Topics

- Bayesian regression
- Weighted / regularized least squares
- Mixtures of Experts
- Gaussian process regression
- Logistic regression
- Poisson regression
- Generalized linear models
- Neural networks
- Support vector regression

Some parts of this lecture are taken from Bishop's book *Pattern Recognition and Machine Intelligence*

Linear Regression

When output y is related to input $\mathbf{x} = (x_1, \ldots x_D)^T$ through a relationship like

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D$$

then we refer to this as a linear regression relationship.

Key point: this model is a *linear* function of the parameters.

Linear Basis Function Regression

We can generalize this relationship by relating the output to fixed nonlinear functions of the input variables.

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

where $\phi_j(\mathbf{x})$ are known as *basis functions*.

The total number of parameters in this model is M.

Linear Basis Function Regression

One can, of course, set $\phi_0(\mathbf{x}) = 1$ yielding

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \phi_j(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x})$$

where $\mathbf{w} = (w_0, \dots, w_{M-1})$ and $\phi = (\phi_0(\mathbf{x}), \dots \phi_{M-1}(\mathbf{x}))^T$

Note :

- setting $\phi(\mathbf{x}) = \mathbf{x}$ implements standard linear regression.
- setting $\phi_j(x) = x^j$ implements polynomial regression.

Maximum Likelihood and Least Squares

Assume that the output is given by a deterministic function $y(\mathbf{x}, \mathbf{w})$ that has additive Gaussian noise added to it so that

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon$$

where $\epsilon \sim N(0, \beta^{-1})$ (β is simply the inverse variance or “precision”)

In this case we can write

$$P(t|\mathbf{x}, \mathbf{w}) = N(t|y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

where the mean and the variance of t can be read off directly.

Maximum Likelihood and Least Squares

Given a data set $\mathbf{X} = \{\mathbf{x}_1, \dots \mathbf{x}_N\}$ with corresponding target values $\mathbf{t} = \{t_1, \dots, t_N\}$ we can write the likelihood of the data as

$$P(\mathbf{t}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^T\phi(\mathbf{x}_n), \beta^{-1})$$

which can be viewed as a function of $\mathbf{w}$, and β.

Since everything is conditioned on $\mathbf{X}$ we will drop it from the notation.

Maximum Likelihood and Least Squares

The likelihood can be re-expressed in matrix form and solved by taking derivatives w.r.t. the parameters and setting the resulting equations equal to zero

$$\begin{aligned} P(\mathbf{t}|\mathbf{w}, \beta) &= \prod_{n=1}^{N} \mathcal{N}(t_n|\mathbf{w}^T\phi(\mathbf{x}_n), \beta^{-1}) \\ &= \mathcal{N}(\mathbf{t}|\mathbf{\Phi}\mathbf{w}, \beta^{-1}\mathbf{I}) \end{aligned}$$

The solutions being

$$\mathbf{w}_{ML} = (\mathbf{\Phi}^T\mathbf{\Phi})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

and

$$\beta^{-1} = \frac{1}{N}(\mathbf{t} - \mathbf{\Phi}\mathbf{w})^T(\mathbf{t} - \mathbf{\Phi}\mathbf{w})$$

Maximum Likelihood and Least Squares

The matrix $\mathbf{\Phi}$ is composed of all basis functions applied to all $\mathbf{x}$ values.

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

Regularized Least Squares

Consider the following regression problem and it's second order polynomial solution

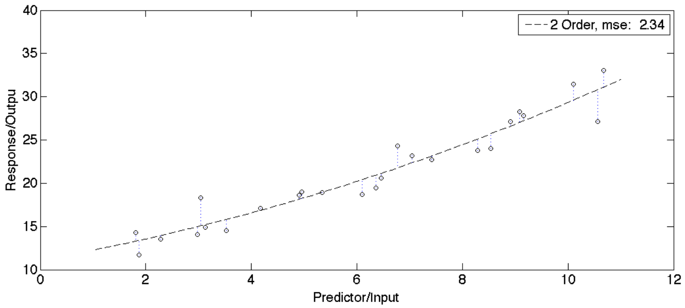

Figure: 2^{nd} order polynomial fit

Regularized Least Squares

It is clear that a better fit to the data can be had by employing a higher order polynomial regression function, for instance, a 17^{th} order polynomial

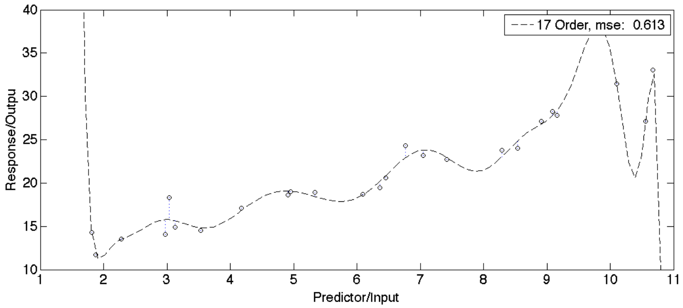

Figure: 17^{th} order polynomial fit

Regularized Least Squares

Problems :

- Too flexible a model (a model with too many parameters) can fit the data too well in the sense that the model doesn't generalize well (this can often be checked and corrected using a cross-validation procedure at often high computational cost)
- Often the "correct" model flexibility isn't known a priori
- Using flexible models can cause numerical instability issues in estimation

Would like a model / estimation procedure that :

- Allows complex, flexible models to be utilized with reduced danger of overfitting
- Avoids numerical instabilities in estimation

Regularized Least Squares

The connection between least squares estimation and maximum likelihood estimation in the multivariate normal regression model arrives from the mechanics of maximizing the log of the regression function likelihood (here we will focus on the weights **w** (by taking the partial derivative w.r.t. **w** and setting the resulting equation equal to 0)).

$$\begin{aligned} \frac{d}{d\mathbf{w}} \log P(\mathbf{t}|\mathbf{w}, \beta) &= 0 \\ \frac{d}{d\mathbf{w}} (\mathbf{t} - \boldsymbol{\Phi}\mathbf{w})^T (\mathbf{t} - \boldsymbol{\Phi}\mathbf{w}) &= 0 \end{aligned}$$

The solution was given before as

$$\mathbf{w}_{ML} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{t}$$

Regularized Least Squares

Regularization adds a penalty for large weights

$$\frac{d}{d\mathbf{w}}\left(\frac{\beta}{2}(\mathbf{t}-\mathbf{\Phi}\mathbf{w})^T(\mathbf{t}-\mathbf{\Phi}\mathbf{w})+\frac{\lambda}{2}\mathbf{w}^T\mathbf{w}\right) \quad = \quad 0$$

The solution to this optimization is

$$\mathbf{w}_{MAP} = (\mathbf{\Phi}^T\mathbf{\Phi}+\frac{\lambda}{\beta}\mathbf{I})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

but $\frac{\lambda}{\beta}$ is just a constant itself so we can write

$$\mathbf{w}_{MAP} = (\mathbf{\Phi}^T\mathbf{\Phi}+\lambda\mathbf{I})^{-1}\mathbf{\Phi}^T\mathbf{t}$$

without loss of generality.

Regularized Least Squares

A solution to our problem?

Regularization : In the linear regression setting regularization

- Penalizes large weights (regression coefficients)
- Helps avoid numerical instabilities in estimation
- Makes it possible to employ complex, flexible models with many parameters with less danger of overfitting

Another possible solution : Bayesian estimation and inference.

Intuition : in a polynomial regression setting, imagine the weights being driven towards zero. If the weights of high order terms in the polynomial are driven all the way to zero then that polynomial is effectively a lower-order polynomial.

Regularization

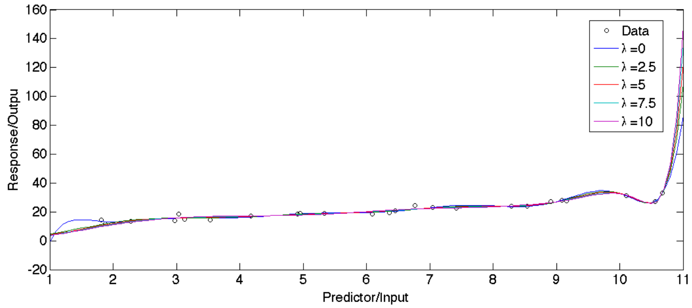

Figure: Regularized 17th order polynomial fit

- Same fit as earlier plot, but different numerical routine results in a different fit even with zero regularization
- Smoothness not apparent without zoom.

Regularization

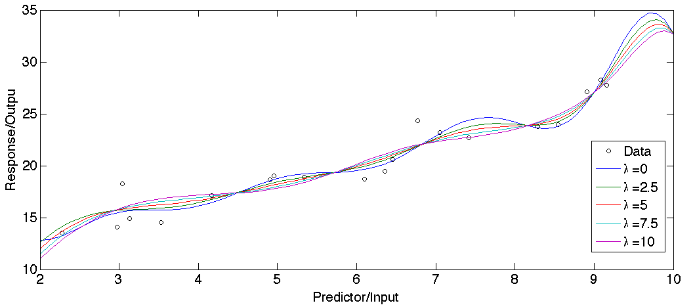

Figure: Zoom of regularized 17^{th} order polynomial fit

- Curves get significantly less wiggly with greater levels of regularization

Regularized Weights

Table: Regularized Weights

λ	0	2.5	5	7.5	10
w_{17}	-754.2	0.77578	0.62928	0.55893	0.51377
w_{16}	2093.3	1.1186	0.9408	0.84949	0.7881
w_{15}	-2332.1	1.3965	1.2306	1.1338	1.0633
w_{14}	1362.2	1.3316	1.2564	1.1899	1.1311
w_{13}	-431.67	0.62999	0.69657	0.69612	0.67641
w_{12}	58.571	-0.4383	-0.35345	-0.31732	-0.30019
w_{11}	6.1042	-0.58128	-0.6995	-0.72647	-0.72499
w_{10}	-3.7669	0.5071	0.58802	0.61917	0.63559
w_9	0.60275	-0.16264	-0.19578	-0.21477	-0.22959
w_8	-0.035517	0.024584	0.032182	0.037576	0.042298
w_7	-0.0012312	-0.0010232	-0.0017861	-0.0024161	-0.0030041
w_6	0.00020767	-0.0002207	-0.00025838	-0.00028164	-0.00029979
w_5	4.896e-06	3.4262e-05	5.1956e-05	6.6306e-05	7.941e-05
w_4	-1.1698e-06	-1.1057e-06	-2.5757e-06	-3.8507e-06	-5.0473e-06
w_3	-2.5692e-08	-1.4743e-07	-1.7262e-07	-1.891e-07	-2.0238e-07
w_2	8.202e-09	1.7064e-08	2.8638e-08	3.8379e-08	4.7368e-08
w_1	-3.6553e-10	-7.1271e-10	-1.4242e-09	-2.0381e-09	-2.6102e-09
w_0	4.8744e-12	1.1292e-11	2.6069e-11	3.8984e-11	5.1072e-11

Regularized Least Squares, Bayesian Interpretation

We optimized a function that looks like this (the negative log likelihood plus some other term) to arrive at regularized least squares

$$\frac{\beta}{2}(\mathbf{t}-\mathbf{\Phi w})^T(\mathbf{t}-\mathbf{\Phi w})+\frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$$

It is reasonable to want to understand more about where this comes from.

Note, this expression looks like the negative log likelihoods from *two* normal distributions, namely $P(\mathbf{t}|\mathbf{w},\beta)=\mathcal{N}(\mathbf{t}|\mathbf{\Phi w},\beta^{-1}\mathbf{I})$ and $P(\mathbf{w}|\mathbf{0},\lambda)=\mathcal{N}(\mathbf{w}|\mathbf{0},\lambda^{-1}\mathbf{I})$

Bayes Rule

Remember Bayes rule?

$$\begin{aligned} P(A|B) &= \frac{P(A,B)}{P(B)} \\ &= \frac{P(A,B)}{\int P(A,B)dA} \\ &= \frac{P(B|A)P(A)}{\int P(A,B)dA} \\ &\propto P(B|A)P(A) \end{aligned}$$

Bayesian Inference

Let's give some names to A and B now. Let's let $A = \{\mathbf{w}\}$ be the collection of parameters in our model (let's assume that β is a fixed parameter given to us at the moment – same for λ). Let's also let $B = \{\mathbf{t}\}$ be the data we are given (remember that everything is conditioned on $\mathbf{X}$). Then

$$P(\mathbf{w}|\mathbf{t}) \quad \propto \quad P(\mathbf{t}|\mathbf{w})P(\mathbf{w})$$

This relationship, and it's obvious more general form, is so important that the individual elements of the equation are given names. They are

$$\mathit{POSTERIOR} \quad \propto \quad \mathit{LIKELIHOOD} \times \mathit{PRIOR}$$

Regularization as MAP Estimation

Finding the set of parameters that maximizes the posterior is called maximum a posteriori (MAP) estimation. When we regularized the linear regression solution we were implicitly imposing a multivariate Gaussian prior on $\mathbf{w}$.

When we maximize the posterior the proportionality becomes equality because the denominator is not a function of $\mathbf{w}$.

$$\frac{d}{d\mathbf{w}} \log P(\mathbf{w}|\mathbf{t}) \quad = \quad \frac{d}{d\mathbf{w}} (\log(P(\mathbf{t}|\mathbf{w})P(\mathbf{w}))) = 0$$

Note that the regularization we chose gives us the distributional form of both the likelihood and the prior:
$P(\mathbf{t}|\mathbf{w}, \beta) = \mathcal{N}(\mathbf{t}|\mathbf{\Phi w}, \beta^{-1}\mathbf{I})$ and $P(\mathbf{w}|\mathbf{0}, \lambda) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \lambda^{-1}\mathbf{I})$ – both are Normal.

MAP Inference

When we do inference, we simply use the MAP optimal value of the parameters

$$\mathbf{w}_{MAP} = (\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \lambda\mathbf{I})^{-1}\boldsymbol{\Phi}^T\mathbf{t}$$

For instance, if we are given a new observation $\mathbf{x}_{pred}$ that we'd like to produce a prediction for, our prediction is simply given by

$$\mathbf{y}_{pred} = \phi(\mathbf{x}_{pred})^T\mathbf{w}_{MAP}$$

Note that the variance of this prediction is just β^{-1} because, for now, we're treating β as known.

Bayesian Inference

It is sometimes the case that we can analytically derive the distributional form of the posterior distribution, i.e. instead of doing MAP inference we can do posterior predictive inference

$$P(\mathbf{w}|\mathbf{t}) \quad \propto \quad P(\mathbf{t}|\mathbf{w})P(\mathbf{w})$$

Namely, if we know $P(\mathbf{w}|\mathbf{t})$ then we can compute or analytically derive the posterior predictive distribution

$$P(t|\mathbf{t}, \beta, \lambda) = \int P(t|\mathbf{w}, \beta)P(\mathbf{w}|\mathbf{t}, \beta, \lambda)d\mathbf{w}$$

In the Gaussian linear regression case we can (we'll derive that result later in the class). For now, let's develop some intuition about what's going on.

Discrete Prior

For pedagogical purposes, let us consider a discrete prior

$$\{\mathbf{w}_\ell\}_{\ell=1}^{L}, \mathbf{w}_\ell \sim \mathcal{N}(\mathbf{0}, \lambda^{-1}\mathbf{I})$$

This can be expressed as a uniformly weighted mixture

$$P(\mathbf{w}) = \frac{1}{L}\sum_{\ell} \delta_{\mathbf{w}_\ell}$$

Posterior Estimation, Discrete Prior

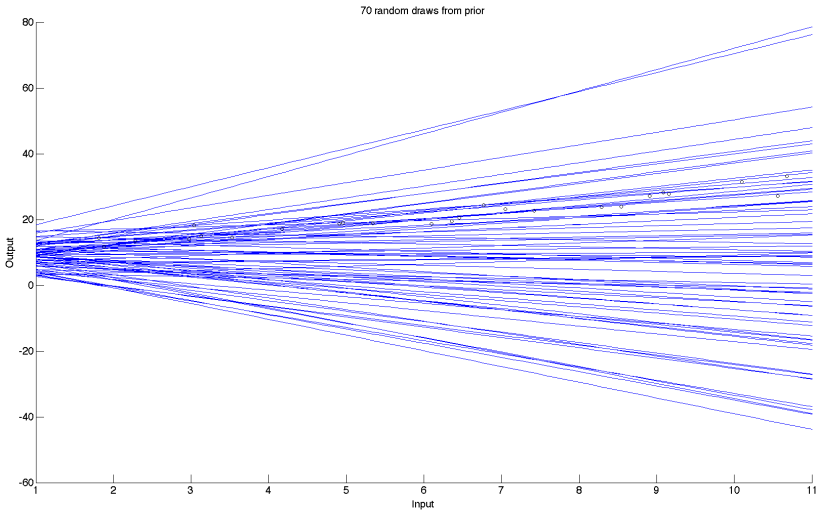

Figure: Random draws from the prior

Discrete Posterior

Given a discrete prior, the posterior can likewise be expressed as a (non-uniformly) weighted discrete mixture

$$P(\mathbf{w}|\mathbf{t}) \propto \sum_{\ell} P(\mathbf{t}|\mathbf{w})\delta(\mathbf{w} - \mathbf{w}_{\ell})$$

Where the weights for the discrete posterior are proportional to the likelihood of the data under a regression model with corresponding parameter

$$P(\mathbf{w}|\mathbf{t}) = \sum_{\ell} \pi_{\ell}\delta_{\mathbf{w}_{\ell}}, \pi_{\ell} = \frac{P(\mathbf{t}|\mathbf{w}_{\ell})}{\sum_j P(\mathbf{t}|\mathbf{w}_j)}$$

Weighted Posterior Distribution Over Model Parameters

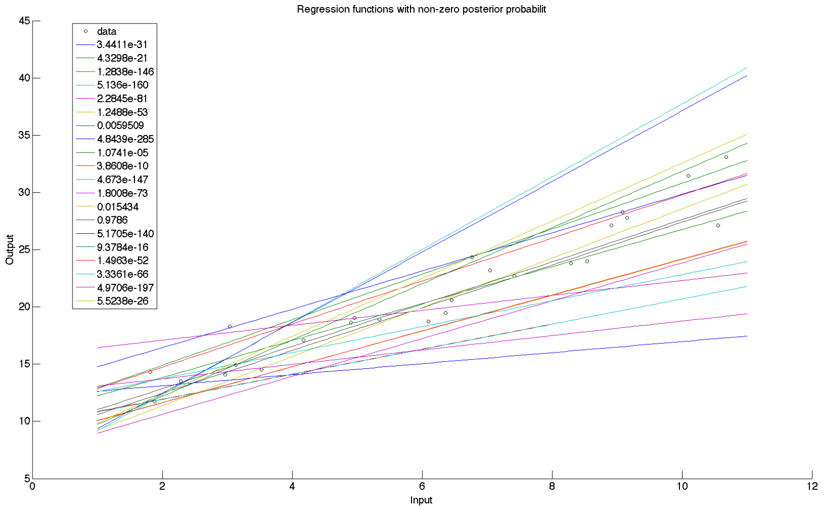

Figure: Weight vectors with high posterior probability.

Posterior Predictive Calculation with Discrete Posterior

Given the simple form of a discrete posterior distribution, the posterior predictive distribution can also be straightforwardly calculated.

$$P(t|\mathbf{t}, \beta, \lambda) = \int P(t|\mathbf{w}, \beta)P(\mathbf{w}|\mathbf{t}, \beta, \lambda)d\mathbf{w}$$

Becomes

$$P(t|\mathbf{t}, \beta, \lambda) = \sum_{\ell} \pi_{\ell} P(t|\mathbf{w}_{\ell}, \beta)$$

Weighted Posterior Distribution Over Model Parameters

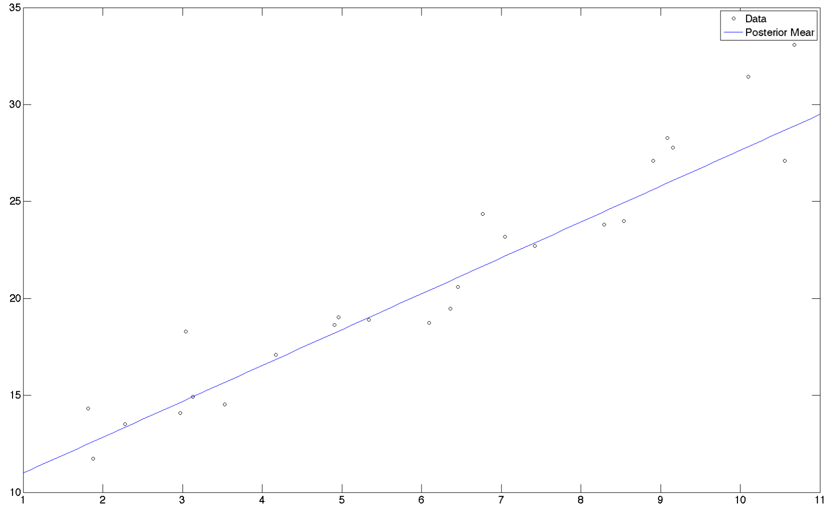

Figure: Posterior predictive distribution.

Weighted Posterior Distribution Over Model Parameters

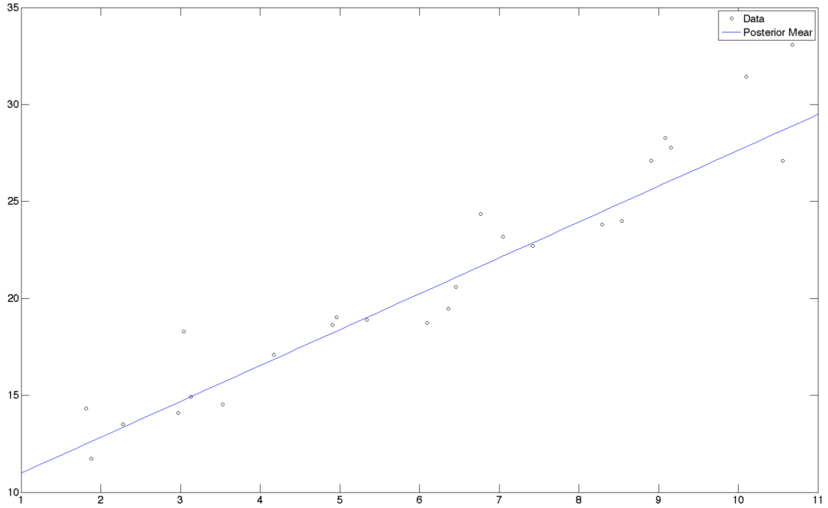

Figure: Posterior predictive distribution.

Bayesian Regression Wrap-Up

Bayesian regression allows you to

- Include prior information (or regularization)
- Express uncertainty in the model parameters
- Etc.

In this family of regression models we find the Gaussian process regressor, mixture of experts regressors, and others.

Bayesian estimation and inference is *far* more general than this setting and worth significant exploration.

Other Advance Regression Techniques

A little closer to home we have

- Poisson regression
- Logistic regression
- Generalized linear models
- Neural networks
- Etc.

Poisson Regression Example : Neural Modeling

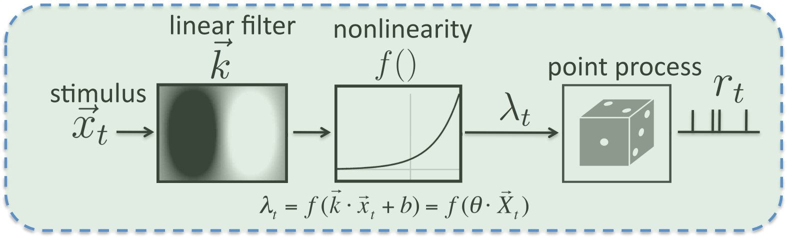

Figure: Schematic for neural response model

Neural Modeling : Input

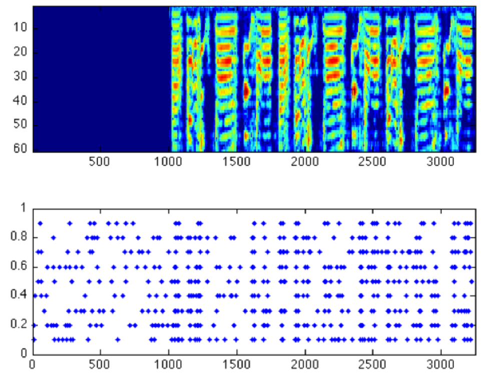

Figure: Input to neural response model

Neural Modeling : Fit

Learned filter for single cell

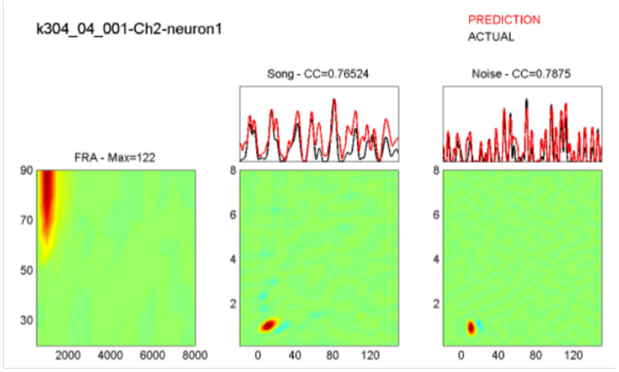

Figure: Learned filter and PSTH predictions from Poisson regression model.

(English) Premier League football goal scoring

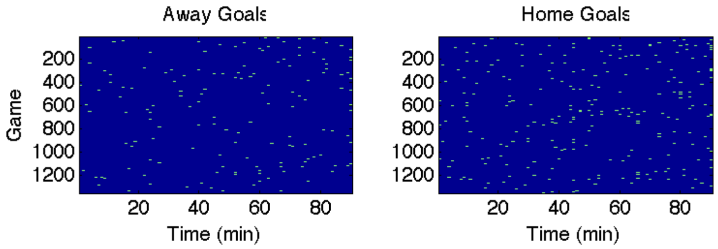

Figure: All goals scored in premier league between 2005-2008 inclusive.

Goal : predict when the next goal is will be scored based on current game state. Available data :

- Current score, home and away
- Yellow cards
- Red card
- Goals in last minute

Another Poisson Regression Example : Football Modeling

Poisson regression can be used to model the probability of a goal (home or away) occurring in any given minute as a function of current game state.

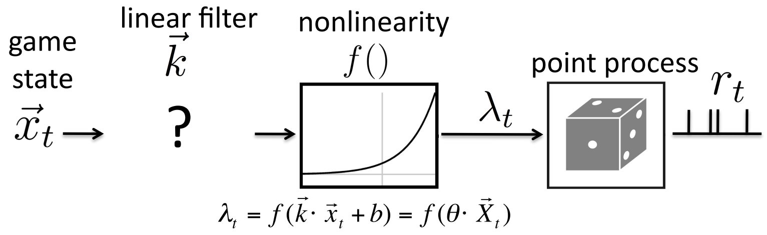

Figure: Schematic for football goal prediction GLM

Poisson regression : a generalized linear model (GLM)

- For count observations (spikes or goals per dt), Poisson likelihood is natural.

$$P(r_t|\mathbf{X}_t,\Theta) = \frac{e^{-\lambda_t dt}(\lambda_t dt)^{r_t}}{r_t}$$

  Here r_t is single element of response vector $\mathbf{r}$ (single team or neuron).

- Strictly positive link function is natural as well, for instance

$$\lambda_t = e^{\mathbf{x}_t'\mathbf{k}+b} = e^{\mathbf{x}_t^{*\prime}\theta}$$

  where $\mathbf{x}_t^*$ is the input at time t, $\mathbf{x}_t$ with an added row/column of ones and θ encompasses both the filter $\mathbf{k}$ and the offset b.

Fitting the parameters of a GLM

A generalized linear model is characterized by

- Observations are independent responses (from an exponential family with mean parameter $\mathrm{E}\left[Y_i\right] = \mu_i$)
- Linear predictor $\mathbf{X}_i'\theta = \theta_0 + \theta_1 X_{i,1} + \cdots + \theta_D X_{i,D}$
- Link function $\mathbf{X}_i'\theta = g(\mu_i)$

  Examples

  - $e^{\mathbf{x}_t^{*\prime}\theta} = \mu_i \implies g(\mu_i) = \log(\mu_i) = \mathbf{x}_t^{*\prime}\theta$ (Poisson)
  - $\mathbf{x}_t^{*\prime}\theta = \mu_i \implies g(\mu_i) = \mu_i = \mathbf{x}_t^{*\prime}\theta$ (Gaussian)
  - etc.

For learning, it helps if the link function is monotonic and differentiable. Iteratively re-weighted least squares can be used to learn the parameters of such a model.