3 Introduction to regression

3.1 The regression problem

We will be dealing with what Seber & Wild (1989) call a "fixed regressor model". That is, we have a data set $\mathcal{X} = \{(\boldsymbol{x}(n), y(n))\}_{n=1,\ldots,N}$ of observation pairs, where $\boldsymbol{x}(n)$ is the input and $y(n)$ is the corresponding output. We assume that the output is generated by the following process

$$y(n) = g(\boldsymbol{x}(n)) + \varepsilon(n) \tag{3.1}$$

where $\varepsilon(n)$ is a zero mean noise process with constant variance σ_ε^2. We refer to g as the "underlying function".

We search for g by picking candidate functions $f(\boldsymbol{x}; \boldsymbol{w})$ from a model family $\mathcal{F}$, where $\boldsymbol{w}$ denotes the parameters of the function. We select from the model family $\mathcal{F}$ the function $f(\boldsymbol{x}; \boldsymbol{w})$ that has the minimum "distance" $E(f(\boldsymbol{x}; \boldsymbol{w}); y)$ to our observed data y. We will below refer to the distance E as the error function.

The modeling process consists of selecting both an appropriate model family $\mathcal{F}$ and the best function in this family.

3.1.1 Examples of model families

Some examples of model families are:

$\mathcal{F}$ = all **linear models**:

$$f(\boldsymbol{x}; \boldsymbol{w}) = w_0 + \sum_{k=1}^{D} w_k x_k = \sum_{k=0}^{D} w_k x_k = \boldsymbol{w}^T \boldsymbol{x} \tag{3.2}$$

where we have added the extra input x_0, which is always equal to one, to allow a simpler notation.

$\mathcal{F}$ = all **polynomial models** of order p:

$$f(\boldsymbol{x}; \boldsymbol{w}) = w_0 + \sum_{k=1}^{p} w_k x^k = \sum_{k=0}^{p} w_k x^k \tag{3.3}$$

where we have used a single variable example, but this could of course be extended to the multivariate case as well.

$\mathcal{F}$ = all **generalized linear models**:

$$f(\boldsymbol{x};\boldsymbol{w}) = w_0 + \sum_{k=1}^{D} w_k h_k(\boldsymbol{x}) \tag{3.4}$$

where $h_k(\cdot)$ are prespecified, fixed, functions.

3.1.2 Examples of error functions (distance measures)

The summed square error (SSE):

$$E = \text{SSE} = \sum_{n=1}^{N} \left(f(\boldsymbol{x}(n);\boldsymbol{w}) - y(n)\right)^2 = \sum_{n=1}^{N} e^2(n) \tag{3.5}$$

For instance, if $\mathcal{F}$ is the set of all linear functions and we use SSE as our error measure, then we have the standard linear regression case. When the SSE is used, we assume that the Gauss-Markov conditions are fulfilled ($\langle\rangle$ stands for the expectation operator):

1. $\langle\varepsilon(n)\rangle = 0$, and
2. $\langle\varepsilon^2(n)\rangle = \sigma_\varepsilon^2$, which is a constant.

The "maximum likelihood" (ML) measure: (we use the negative log likelihood because it is more convenient to work with)

$$E = -\ln\mathcal{L}(\mathcal{X}|\boldsymbol{w}) = -\ln\left(\prod_{n=1}^{N} p(\boldsymbol{x}(n), y(n)|\boldsymbol{w})\right) = -\sum_{n=1}^{N} \ln p(\boldsymbol{x}(n), y(n)|\boldsymbol{w}) \tag{3.6}$$

where $p(\boldsymbol{x}(n), y(n)|\boldsymbol{w})$ is the likelihood for the observation $\{\boldsymbol{x}(n), y(n)\}$ given the parameter values $\boldsymbol{w}$. The most common assumption is the Gaussian likelihood

$$p(\boldsymbol{x}(n), y(n)|\boldsymbol{w}) = \frac{1}{\sigma_\varepsilon\sqrt{2\pi}} \exp\left[-\frac{\left(f(\boldsymbol{x}(n);\boldsymbol{w}) - y(n)\right)^2}{2\sigma_\varepsilon^2}\right] \tag{3.7}$$

in which case the negative log likelihood is

$$\begin{aligned} E = -\ln\mathcal{L}(\mathcal{X}|\boldsymbol{w}) &= \ln\sigma_\varepsilon + 0.5\ln 2\pi + \frac{1}{2\sigma_\varepsilon^2}\sum_{n=1}^{N}\left(f(\boldsymbol{x}(n);\boldsymbol{w}) - y(n)\right)^2 \\ &\rightarrow \sum_{n=1}^{N}\left(f(\boldsymbol{x}(n);\boldsymbol{w}) - y(n)\right)^2 \end{aligned} \tag{3.8}$$

i.e. equal to the SSE, since we can ignore all terms that do not depend on $\boldsymbol{w}$. However, the ML cost is more general than SSE and easily adjusted to new conditions, e.g. when the Gauss-Markov conditions are violated.

Note:
The maximum likelihood measure is based on the following: We assume a distribution for the noise ε, e.g. Gaussian. We then assume that our model is equal to the true underlying function, in which case the residuals $e(n) = y(n) - f(\boldsymbol{x}(n); \boldsymbol{w})$ become the noise term $\varepsilon(n)$, c.f. expression (3.1), and we can compute the likelihood that the observed data $\mathcal{X}$ was generated by our model $f(\boldsymbol{x}(n); \boldsymbol{w})$. We then vary the parameters $\boldsymbol{w}$ and choose the parameter values that maximize this likelihood.

This picture of ML may feel "backward", assuming that our model is correct and then computing likelihood etc., and the Bayesian measure described below actually clarifies things quite a bit. What the ML measure actually does is to maximize the probability for $\boldsymbol{w}$ given the data $\mathcal{X}$ under a flat prior assumption.

The Bayesian measure:
Maximizing the likelihood is somewhat strange. Why maximize the likelihood for the observations given the model parameters (although we do this by changing the model parameters)? What we really would like to do is to maximize the model parameters, given the observations. Bayes' theorem tells us how we should do. The probability for the model parameters, given the observations, is expressed as

$$p(\boldsymbol{w}|\mathcal{X}) = \frac{p(\mathcal{X}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{X})} = \frac{\mathcal{L}(\mathcal{X}|\boldsymbol{w})p(\boldsymbol{w})}{p(\mathcal{X})} \tag{3.9}$$

where $p(\boldsymbol{w})$ is our "prior" for the model parameters $\boldsymbol{w}$. Just as in the case of the ML cost, it is more convenient to minimize the negative likelihood, which gives us

$$\begin{aligned} E = -\ln p(\boldsymbol{w}|\mathcal{X}) &= -\ln \mathcal{L}(\mathcal{X}|\boldsymbol{w}) - \ln p(\boldsymbol{w}) + \ln p(\mathcal{X}) \\ &\rightarrow -\ln \mathcal{L}(\mathcal{X}|\boldsymbol{w}) - \ln p(\boldsymbol{w}) \end{aligned} \tag{3.10}$$

since the third term does not depend on the model parameters $\boldsymbol{w}$.

The Bayesian error measure is even more general than the ML error. The ML error is equal to the special case of a uniform prior in the Bayesian picture. The Bayesian error is very important to avoid overfitting, which is a significant problem with flexible nonlinear regression models.

3.2 Generalized linear models vs. nonlinear ones

Doing linear regression amounts to solving a set of linear equations, which is lightning fast in a numerical analysis package like MATLAB. Therefore, one can afford a lot more experimenting if linear models are used. Furthermore, the solution to the linear regression problem is, in most practical cases, unique (i.e there exists only one solution). Of course, for generalized linear regression to work well it is necessary that the underlying function g is close to linear.

Nonlinear regression, on the other hand, requires iterative search methods to optimize the model parameters. Also, there can be (and often is) many local minima, i.e. suboptimal solutions to the problem. It therefore takes considerably longer time to fit nonlinear models, which leaves less time for experimenting. However, there is no need for the underlying function g to be linear.

3.3 Should all variables be included?

Often, the naive modeller will think that "the more inputs I add to the model, the better it should become". Well, this may be true for the error on the training set, but it definitely is not true for the generalization (test) error. For example, if data is generated by a linear process with noise, and used to train a linear model, and then new data is generated from the same process, with noise, then the relationship between the training and the test mean square errors is

$$\langle \mathrm{MSE}_{test} \rangle = \langle \mathrm{MSE}_{train} \rangle + \frac{\sigma_\varepsilon^2 (D+1)}{N} \tag{3.11}$$

where D is the dimension of the inputs (excluding the constant bias term), N is the number of observations, and σ_ε^2 the variance of the noise. The operator $\langle \cdot \rangle$ denotes taking averages over both different training sets and over different test sets.

This means that we pay a penalty equal to $\frac{\sigma_\varepsilon^2}{N}$ for each input we add to the model. Thus, if the information content in a new input variable is less than the noise penalty, then the result will be a worse test error than if this input had been left out of the regression!

This is an example of overfitting and the so-called "bias vs. variance" trade-off. The more inputs we add, the better a linear model can model the

noise (because the data set becomes less and less dense if we add dimensions while the number of observations remains the same), with the consequence that model variance increases. Model variance is a measure of how much a model varies when we train it with different training sets. At the same time, the bias may decrease (it can also remain constant). The generalization performance is a sum of bias and variance (see below) and one must therefore find the optimal combination of the two.

3.4 Overfitting and the bias vs. variance trade-off

This section treats one of the most fundamental insights a nonlinear modeller needs. Namely that the training data is only a sample of the real world and that it is surprisingly easy to overemphasize the importance of the training data, at the cost of worse performance on new test data (i.e. worse generalization performance).

To understand how this can be, we introduce the concepts of "model bias" and "model variance". Model bias is a measure of how well we can model the underlying function g with our model family $\mathcal{F}$. If the underlying function can be modeled perfectly with a model from our family, i.e. if the underlying model is a member of our family, $g \in \mathcal{F}$, then we say that our model family has zero model bias. If the underlying function is not a member of the model family, $g \notin \mathcal{F}$, then we say that our model family is biased. Model variance is a measure of how much our models vary when we train them with different training sets. If the model family $\mathcal{F}$ is very small then there will be small differences between models trained with different training sets and we say that the model variance is small. On the other hand, if the model family is large then there can be (will be by Murphy's law) large differences between models trained with different training sets and we say that the model variance is large.

Examples:

Suppose that the underlying function g is linear.
$\mathcal{F} =$ { all linear models } has zero model bias and small model variance.
$\mathcal{F} =$ { all polynomial models of order 3 } also has zero model bias, but a significantly larger model variance.

Suppose that the underlying function g is cubic.
$\mathcal{F} =$ { all linear models } has a significant model bias and small model variance.

$\mathcal{F} = \{$ all polynomial models of order 3 $\}$ has zero model bias and a large model variance.

3.4.1 The bias and variance decomposition of generalization error

When we construct models we are always interested in models that are able to generalize, i.e. that would produce good predictions/classifications on out-of-sample data. We are, in mathematical terms, interested in minimizing the generalization error

$$\begin{aligned} E_{gen}(\boldsymbol{w}) &= \int\int \left(f(\boldsymbol{x};\boldsymbol{w}) - y(\boldsymbol{x})\right)^2 p(\boldsymbol{x})p(y|\boldsymbol{x})d^D x dy \\ &= \int\int \left(f(\boldsymbol{x};\boldsymbol{w}) - g(\boldsymbol{x}) - \varepsilon\right)^2 p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon \end{aligned} \qquad (3.12)$$

where $p(\boldsymbol{x})$ is the probability density function for the inputs $\boldsymbol{x}$, and $p(\varepsilon)$ is the probability density function for the noise.

Now, the generalization error $E_{gen}(\boldsymbol{w})$ will be different for models trained on different training data, since $\boldsymbol{w}$ varies. We are interested in the expected performance from models in our model family $\mathcal{F}$ and we therefore average $E_{gen}(\boldsymbol{w})$ over all training sets $\mathcal{X}$

$$\langle E_{gen}\rangle_w = \int E_{gen}(\boldsymbol{w})p(\boldsymbol{w}|\mathcal{X})p(\mathcal{X})d\mathcal{X} \qquad (3.13)$$

where $p(\mathcal{X})$ is the probability density function for picking training set $\mathcal{X}$, and $p(\boldsymbol{w}|\mathcal{X})$ is the conditional probability for ending up with parameters $\boldsymbol{w}$ when we train with training data set $\mathcal{X}$. The latter is a distribution since our final set of parameters can depend on the initial conditions for our training.

Let's do some algebra...

$$\begin{aligned} \langle E_{gen}\rangle_w &= \langle \int\int \left(f(\boldsymbol{x};\boldsymbol{w}) - g(\boldsymbol{x}) - \varepsilon\right)^2 p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w \\ &= \langle \int\int f^2(\boldsymbol{x};\boldsymbol{w})p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w + \langle \int\int g^2(\boldsymbol{x})p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w \\ &+ \langle \int\int \varepsilon^2 p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w - 2\langle \int\int f(\boldsymbol{x};\boldsymbol{w})g(\boldsymbol{x})p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w \\ &- 2\langle \int\int f(\boldsymbol{x};\boldsymbol{w})\varepsilon p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w - 2\langle \int\int g(\boldsymbol{x})\varepsilon p(\boldsymbol{x})p(\varepsilon)d^D x d\varepsilon\rangle_w \\ &= \langle \int f^2(\boldsymbol{x};\boldsymbol{w})p(\boldsymbol{x})d^D x\rangle_w + \int g^2(\boldsymbol{x})p(\boldsymbol{x})d^D x + \sigma_\varepsilon^2 \\ &- 2\langle \int f(\boldsymbol{x};\boldsymbol{w})g(\boldsymbol{x})p(\boldsymbol{x})d^D x\rangle_w \end{aligned} \qquad (3.14)$$

where we have made use of the facts that $\int p(\varepsilon)d\varepsilon = 1$, $\int p(\boldsymbol{x})d^D x = 1$, $\int \varepsilon^2 p(\varepsilon)d\varepsilon = \sigma_\varepsilon^2$, that the noise ε is uncorrelated with both $g(\boldsymbol{x})$ and $f(\boldsymbol{x};\boldsymbol{w})$, and that $g(\boldsymbol{x})$ does not depend on the training set (only $\boldsymbol{w}$ depends on the training set $\mathcal{X}$).

Averaging over the training sets $\mathcal{X}$ and integrating over $\boldsymbol{x}$ are independent operations so we can interchange the order of these, which results in

$$\begin{aligned}
\langle E_{gen}\rangle_w &= \int \langle f^2(\boldsymbol{x};\boldsymbol{w})\rangle_w p(\boldsymbol{x})d^D x + \int g^2(\boldsymbol{x})p(\boldsymbol{x})d^D x + \sigma_\varepsilon^2 \\
&- 2\int \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w g(\boldsymbol{x})p(\boldsymbol{x})d^D x \\
&= \int \langle f^2(\boldsymbol{x};\boldsymbol{w})\rangle_w p(\boldsymbol{x})d^D x + \int g^2(\boldsymbol{x})p(\boldsymbol{x})d^D x + \sigma_\varepsilon^2 \\
&- 2\int \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w g(\boldsymbol{x})p(\boldsymbol{x})d^D x + \int \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w^2 p(\boldsymbol{x})d^D x \\
&- \int \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w^2 p(\boldsymbol{x})d^D x \\
&= \int \langle [f(\boldsymbol{x};\boldsymbol{w}) - \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w]^2\rangle_w p(\boldsymbol{x})d^D x \\
&+ \int [g(\boldsymbol{x}) - \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w]^2 p(\boldsymbol{x})d^D x + \sigma_\varepsilon^2 \\
&= \text{Variance} + \text{Bias}^2 + \text{noise variance} \qquad (3.15)
\end{aligned}$$

That is, the expected generalization error $\langle E_{gen}\rangle_w$ can be decomposed into three terms which we denote "the model variance", "the model bias" (squared), and "the noise variance". The model bias (squared) is

$$\text{Bias}^2 = \int [g(\boldsymbol{x}) - \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w]^2 p(\boldsymbol{x})d^D x \qquad (3.16)$$

which is the squared difference between the average model $\langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w$ and the underlying function $g(\boldsymbol{x})$. The average model equals the model we would get if we had infinitely many observations in our training set. Thus, if $g \in \mathcal{F}$ then $\langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w = g(\boldsymbol{x})$ and we have zero bias.

The model variance is

$$\text{Variance} = \int \langle [f(\boldsymbol{x};\boldsymbol{w}) - \langle f(\boldsymbol{x};\boldsymbol{w})\rangle_w]^2\rangle_w p(\boldsymbol{x})d^D x \qquad (3.17)$$

which is the variance of the models in $\mathcal{F}$ when they are trained with different training sets $\mathcal{X}$.

In general, the model bias will increase when the model variance decreases, and vice versa.

It should not be surprising that the noise variance also enters into the expected generalization error, since the noise variance represents a lower limit on the generalization error. We cannot do anything about the noise.

The bottom line is that whenever we are constructing a model, we should remember that the ultimate goal is to minimize $\langle E_{gen} \rangle$, which entails weighting the model bias against the model variance. This may mean that choosing a model family $\mathcal{F}$ such that $g \in \mathcal{F}$ is bad, because the accompanying model variance cancels the benefits from having zero bias. Doing this bias vs. variance trade-off well is very much the trade-mark of the experienced modeler.

References

Seber, G. A. F. & Wild, C. J. (1989), *Nonlinear Regression*, John Wiley & Sons, New York.

Wetherill, G. B. (1986), *Regression Analysis with Applications*, Monographs on Statistics and Applied Probability, Chapman and Hall, London.