


JUDGMENT DAY FOR PRICING



Tony Ward provides an algorithmic framework for building generalised linear models using the field of machine learning

“Three billion human lives ended on August 29th, 1997. The survivors of the nuclear fire called the war ‘Judgment Day’. They lived only to face a new nightmare, the war against the machines.”

In the *Terminator* film franchise, an artificial intelligence system designed to protect the US seizes control of the world and launches a global war of extermination against humanity. For the moment, thankfully, this is just science fiction – though it is understandable why, when we hear the term ‘machine learning’, that we immediately think of Arnold Schwarzenegger.

Machine learning is, in fact, a bonafide academic field of study. It is a subfield of computer science, and focuses on the development of computer programmes, or algorithms, that can teach themselves to learn or grow when exposed to new data.

It is a relatively new field, but has rapidly gained in

“Generalised linear models have become the default modelling method for claims costs, and are also widely used in renewal, conversion and lapse modelling”

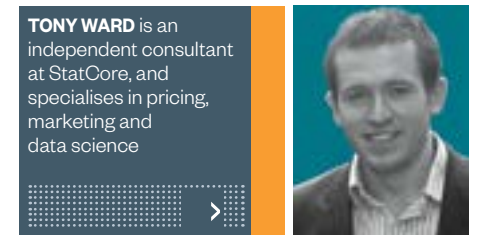


Table 1: Document term matrix for commercial lines claim data

	Damage	Impact	Shop	Water
Impact damage to shop canopy – TP unknown	1	1	1	0
Water leak from bath	0	0	0	1
Impact damage	1	1	0	0
Leakage of water on laminate floor	0	0	0	1

popularity, with applications including: spam detection, speech recognition, recommendation algorithms, such as those used by Amazon, and autonomous driving cars.

Algorithmic pricing

In the 1990s, British actuaries introduced generalised linear models (GLMs) as tools for analysing insurance data. GLMs have now become the default method for modelling claims costs, and are also widely used in renewal, conversion and lapse modelling.

This article presents an algorithmic framework for building GLMs, using Lasso (least absolute shrinkage and selection operator) regression – a predictive modelling technique taken from the machine learning community. The framework retains the familiar components of a GLM (linear predictor, link function, error term), but allows us to fit models in an automated way. This allows us to test unstructured text data such as underwriter notes or social media activity in our pricing models. All code and data used here have been made freely available at www.statcore.co.uk.

Motivating example: salary prediction

TheActuaryJobs.com is the official job board for the actuarial profession. At the time of writing, there were 757 live jobs listed, covering a wide range of experience levels, sectors and locations. Fortunately, the structure of the website meant that it was relatively straightforward to scrape all 27,786 historic job advertisements. Each advert contains information relating to the position, such as www.theactuaryjobs.com/job/39181

Our goal is to use the information provided in the job description to predict the probability that a given job advertisement pays more than £70,000. The information in the advert is a mixture of structured and unstructured text data, and we would like to test both data types in our model.

The first step is to convert the unstructured text into a document term matrix – a tabular form with columns indicating the presence or absence of words. In *The Actuary*, May 2011, my colleague, Alan Chalk and I, presented a

demonstration of this technique on a commercial lines claims dataset, to extract information from the loss adjustor notes (see *Table 1*, above). Once we apply this technique to the ‘job title’ and ‘further info’ variables, we obtain a modelling dataset with the following dimensions (see *Table 2*, below).

Table 2: Modelling dataset

Rating factor	Levels
Experience	7
Sector	14
Country	89
City	99
Job title	189
Further info	691

Model complexity: Man vs machine

During the modelling phase, we must decide on an appropriate degree of model complexity. How many variables should we include in the model? When we include categorical variables, should we group any levels with each other? Should we fit a curve to our numeric variables? If so, what should that curve look like?

Typically, these decisions are taken by an actuary, who uses statistical and consistency tests together with common sense and experience to arrive at a final model. However, this approach simply does not scale very well. As the number of candidate rating factors increases, the number of potential models increases exponentially. This leads to much longer model build-times, and an increased risk of finding a sub-optimal model.

Testing unstructured data presents a new challenge. Since the number of parameters being estimated often runs into the thousands, we need an alternative approach to determine model complexity. That approach is regularisation.

Regularisation and the lasso

When we fit a GLM, the model parameters are calculated by maximum likelihood estimation. In practice, we minimise the negative of the log likelihood, which is equivalent.

$$\min_{\beta_0, \beta} -l(\beta|X, Y)$$

Where β_0 is the intercept term, β is the vector of model relatives and $l(\beta|X, Y)$ is the log likelihood.

The main idea behind regularisation is to penalise complex models. This is achieved by defining a penalty function to quantify the complexity of the model – more complex models will have a greater penalty associated with them. Since the process of fitting a GLM can be considered an optimisation problem where a loss function is minimised, we can add the penalty term and minimise the whole expression together. Lasso regression shrinks the regression coefficients by imposing a L^1 norm penalty on their size.

$$\sum_{j=1}^p |\beta_j| \leq t$$

Models are estimated by penalised maximum likelihood,

$$\min_{\beta_0, \beta} -\frac{1}{N} l(\beta|X, Y) + \lambda \sum_{j=1}^p |\beta_j|$$

where $l(\beta|X, Y)$ is the log likelihood defined previously and $\lambda \geq 0$ is a complexity parameter that controls the amount of shrinkage in the parameter estimates: the larger the value of λ , the greater the amount of shrinkage, which means smaller parameter estimates. Notice that, since β_0 is not present in the penalty function, the intercept term is left unconstrained. The optimal λ is estimated from the training data using k-fold cross validation.

It turns out that making t sufficiently small will cause some coefficients to be exactly zero. So lasso regression performs automatic model selection. It scales well to wide datasets, owing to an optimisation technique called co-ordinate descent, which optimises one model parameter at a time. This allows linear models with arbitrary size p to be fitted, and is why lasso regression is so popular in applications such as genomics, where, for example, $p = 40k$ genes are measured for $N = 100$ subjects.

Application: salary prediction

Recall that our goal is to use the information provided in the job description to predict the probability that a given job advertisement pays more than £70,000. We set this up as a GLM with binomial error distribution, and fitted both a lasso regression and a traditional GLM using two rating factors – experience and sector.

The model parameters shown are on the linear predictor scale – a value of 0 means no effect and a positive value means a relatively high probability (see *Figure 1*).

The red line shows the model parameters for the GLM model. The confidence interval for hedge funds (not shown) indicates that this parameter estimate is not significantly different from zero. We now have a choice – do we leave that parameter estimate ungrouped, or should we group it with another factor level? These decisions are harder when you have limited prior knowledge about the expected relationship.

Lasso regression takes these difficult decisions away from the actuary by automatically deciding which rating factors appear in the model, and which levels should be non-zero. The lasso regression (blue line) has set several levels to zero, and has shrunk the remaining parameter estimates towards zero, especially in areas of low exposure.

Our benchmark tests (statcore.co.uk/case-studies/algorithmic-pricing/) show lasso regression outperforms the traditional approach to fitting GLM models, especially where we have limited prior information on the expected relationship – for example, with geographic or vehicle group data.

Continuing with our modelling problem, we split the data randomly into training and hold out portions. Using the training data, we use lasso regression to fit several candidate models using combinations of the input variables. For example, model ESCICOJF contains variables ‘experience’, ‘sector’, ‘city’, ‘county’, ‘job title’ and ‘further info’. These models are then used to make predictions on the hold out data, and are assessed based on their deviance (see *Figure 2*).

ESCICOJ has the lowest deviance and is therefore selected as the final model. Note that this model includes the unstructured text field ‘job description’. To provide insight into how this increased predictive accuracy, we compare models with (ESCICOJ) and without (ESCICO) this variable, focusing on the highest predictions (see *Figure 3*).

Words such as ‘partner’, ‘director’ ‘head’ and ‘chief’ all indicate seniority, and words ‘expert’, ‘stochastic’, ‘remetrica’ and ‘esg’ indicate a niche specialism, which is why these adverts receive very high predictions.

We have presented a new framework for building GLM models in an automated way, using lasso regression, one of many powerful machine learning algorithms. Leo Breiman, the statistician and inventor of the Random Forest algorithm, said: “If our goal as a field is to use data to solve problems, then we need to move away from exclusive dependence on data models and adopt a more diverse set of tools.”

Thanks to Redactive Media for permission to use data from TheActuaryJobs.com

Figure 1: Model parameters: lasso regression vs GLM

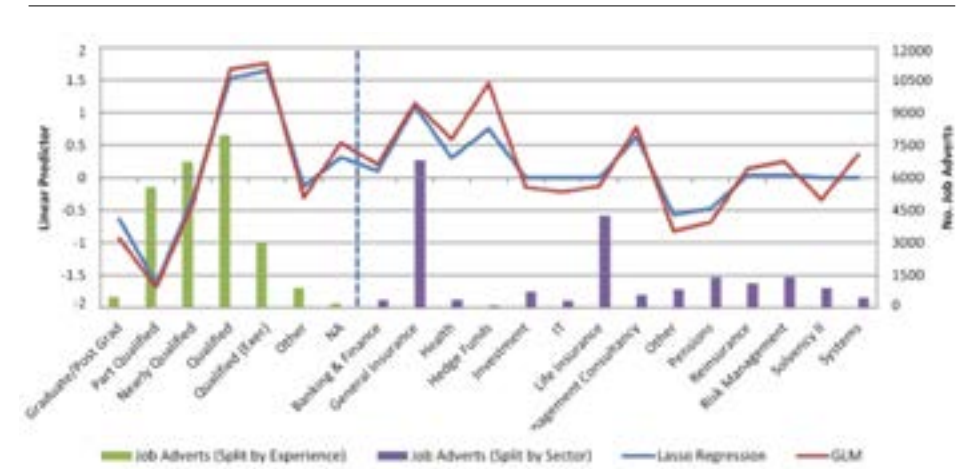


Figure 2: Hold out performance: deviance

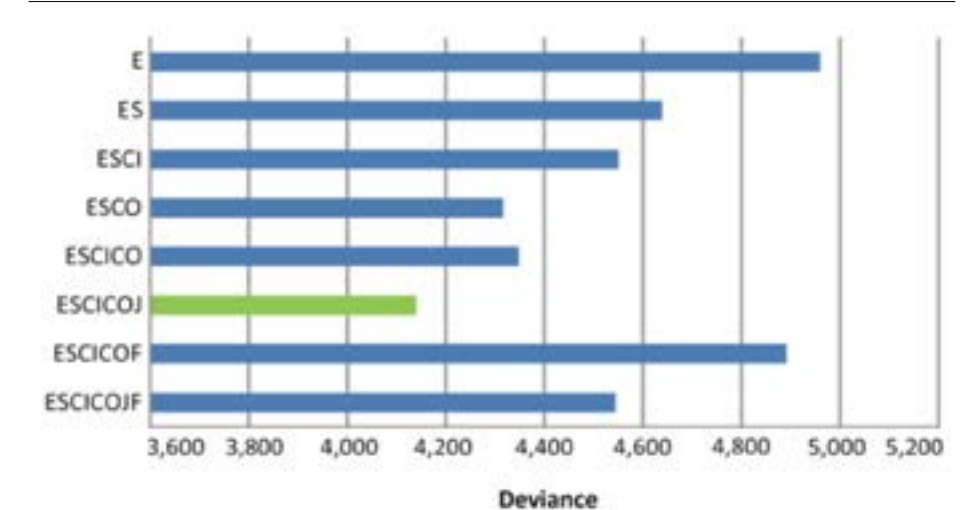


Figure 3: Predicted values: job description

