Learning the Semantics of Discrete Random Variables: Ordinal or Categorical?

José Miguel Hernández-Lobato[1]
Harvard University
jmh@seas.harvard.edu

James Robert Lloyd[1]
Cambridge University
jrl44@cam.ac.uk

Daniel Hernández-Lobato
Universidad Autónoma de Madrid
daniel.hernandez@uam.es

Zoubin Ghahramani
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

When specifying a probabilistic model of data, the form of the model will typically depend on the spaces in which random variables take their values. In particular, different probability distributions are appropriate for continuous, discrete and binary data. As we respond to ever increasing quantities of data, with increasingly more automatic data analysis techniques, it is necessary to identify these different types of data automatically. While it is trivial to create concise logical rules to distinguish between many different types of data, this cannot be said for choosing between categorical and ordinal data, let alone inferring the ordering. We present some first attempts at this problem and evaluate their performance empirically.

1 Introduction

Many data analytic procedures depend on whether data are binary, categorical, ordinal, continuous or otherwise. On small datasets this information is typically 'obvious' or known. However, as we respond to ever increasing quantities of data with increasingly sophisticated automatic data analysis techniques [e.g. 15, 11, 3, 14, 6, 10] it will be necessary to automatically identify different data types. While it is trivial to distinguish between binary and other discrete variables automatically (by counting the unique values) it is impossible to tell the difference between categorical and ordinal data in isolation. Knowing when data is ordinal allows one to develop data analysis techniques with improved performance over their categorical counterparts [e.g. 2].

We propose two Bayesian methods for identifying whether data is categorical or ordinal and to infer the true ordering of the variables when the data is ordinal. This latter operation is performed by an exhaustive search algorithm that attempts to find a particular permutation of the variables that maximizes the Bayesian model evidence. After this, our approach for discriminating between ordinal and categorical data is based on comparing the model evidence and the predictive test log-likelihood of ordinal and categorical models.

2 Probabilistic models for discrete random variables

Assume a dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ of input vectors $\mathbf{x}_i$ and corresponding discrete labels y_i, which may take L different values, *i.e.*, $y_i \in \mathcal{L} = \{v_1, \ldots, v_L\}$. We describe two probabilistic models for the conditional distribution $p(y_i|\mathbf{x}_i)$. Each model is based on a different semantic interpretation of the labels in $\mathcal{L}$. The first one assumes an ordering relation for the labels. The second one does not.

[1]Both authors contributed equally.

2.1 An ordinal regression model

We assume that there is an ordering among the labels, *i.e.*, they can be ranked according to some criterion. For example, each data point in $\mathcal{D}$ could represent a different patient in a hospital, with $\mathbf{x}_i$ encoding the patient's symptoms and y_i encoding the severity of the patient's condition, with $\mathcal{L} = \{v_1 = \text{mild}, v_2 = \text{severe}, v_3 = \text{very mild}, v_4 = \text{very severe}\}$. In this case, there is a permutation σ of 1,…,L such that $v_{\sigma(1)}$,…,$v_{\sigma(L)}$ is a sequence of labels correctly ordered according to their semantic meaning. Two valid permutations are then $\sigma_1 = (2, 3, 1, 4)$ and $\sigma_2 = (3, 2, 4, 1)$ that rank the labels from *very mild* to *very severe* and from *very severe* to *very mild*, respectively.

Given a valid permutation σ, we can then learn $p(y_i|\mathbf{x}_i, \sigma)$ under the above setting using an ordinal regression model [2, 13]. Under this model, the labels y_i are generated from the corresponding input vectors $\mathbf{x}_i$ as follows: First, a function f is used to map the feature vectors $\mathbf{x}_i$ to the real line, which is partitioned into L contiguous intervals. Assume that $f(\mathbf{x}_i)$ falls in the k-th interval. Then y_i is fixed to take value $v_{\sigma(k)}$. Let $b_0 < \ldots < b_L$ be the boundaries of the L intervals, where $b_0 = -\infty$ and $b_L = \infty$, and let $f_i = f(\mathbf{x}_i)$. The likelihood for f_i and $\mathbf{b} = (b_1, \ldots, b_{L-1})$ given y_i is then

$$p(y_i|f_i, \mathbf{b}, \sigma) = \prod_{l=1}^{L-1} \Theta\left[\text{sign}(g(y_i) - l - 0.5)(f_i - b_l)\right] , \tag{1}$$

where Θ is the Heaviside step function and $g(y_i)$ returns the index of the value of y_i in the ranking of the entries of $\mathcal{L}$ specified by the permutation σ, *i.e.*, if $y_i = v_j$ then $g(y_i) = \sigma(j)$. (1) takes value 1 when the following conditions are all met: a) f_i falls in the k-th contiguous interval, b) y_i takes vale $v_{\sigma(k)}$, c) b_0 to $b_{g(y_i)}$ are all smaller than f_i and c) $b_{g(y_i)+1}$ to b_L are all larger than f_i. Thus, (1) allows to learn the boundary vector $\mathbf{b}$ from the data [13]. The prior for $\mathbf{b}$ is $p(\mathbf{b}) = \prod_{l=1}^{L-1} \mathcal{N}(b_l|m_l^0, v_l^0)$, where we fix the mean parameters $m_1^0, \ldots, m_L^0$ to be on an uniform grid in the interval $[-6, 6]$, that is, if $L = 5$, we have that $(m_1^0, \ldots, m_4^0) = (-4, -2, 2, 4)$, as recommended in [8]. We fix a Gaussian process (GP) prior for f, *i.e.*, *a priori* $\mathbf{f} = (f_1, \ldots, f_n)$ follows the multivariate Gaussian distribution $p(\mathbf{f}) = \mathcal{N}(\mathbf{f}|\mathbf{m}, \mathbf{K})$, where the mean vector $\mathbf{m}$ and the covariance matrix $\mathbf{K}$ are obtained by evaluating the mean function and the covariance function of the GP on the feature vectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ [9]. The posterior distribution for $\mathbf{f}$ and $\mathbf{b}$, $p(\mathbf{f}, \mathbf{b}|\mathcal{D}, \sigma)$, is in general intractable. However, we can use expectation propagation (EP) to obtain an efficient Gaussian approximation [7]. Let $q(\mathbf{f}, \mathbf{b})$ be this approximation. The predictive distribution for the label $y_\star$ of a new vector $\mathbf{x}_\star$ is approximated as

$$p(y_\star|\mathbf{x}_\star, \mathcal{D}) = \int p(y_\star|f_\star, \mathbf{b})p(f_\star|\mathbf{f})p(\mathbf{f}, \mathbf{b}|\mathcal{D}) \approx \int p(y_\star|f_\star, \mathbf{b})p(f_\star|\mathbf{f})q(\mathbf{f}, \mathbf{b})d\mathbf{f}\,\mathbf{b} , \tag{2}$$

where $f_\star = f(\mathbf{x}_\star)$ and $p(f_\star|\mathbf{f})$ is the Gaussian process predictive distribution for $f_\star$ given $\mathbf{f}$ [9].

2.1.1 Learning the permutation σ

EP also approximates the model evidence, *i.e.*, the normalization constant in $p(\mathbf{f}, \mathbf{b}|\mathcal{D})$ [7]. For a fixed σ, we can then maximize this approximation with respect to the GP hyper-parameters (*e.g.*, the length-scales and amplitudes of the covariance function). This can be done using a gradient ascent algorithm since EP also approximates the gradients of $p(\mathcal{D}, \sigma)$ with respect to the GP hyper-parameters [12]. This allows to select the optimal GP hyper-parameters [1]. Let $\tilde{z}(\sigma)$ be the value of the maximized approximation of the model evidence with respect to the GP hyper-parameters for a given permutation σ. We can infer the value of σ that generates a valid ranking of the labels in $\mathcal{L}$ by further maximizing $\tilde{z}(\sigma)$ with respect to σ.

To maximize $\tilde{z}(\sigma)$ we start with an initial permutation selected uniformly at random. After this, we generate all the possible $L(L-1)/2$ sets containing two elements from the set of integers $\{1, \ldots, L\}$ forming σ and iterate as follows: First, for each set $\{i, j\}$ previously generated, we create a new permutation $\sigma_{i,j}$ by swapping the i-th and the j-th elements in σ. For each $\sigma_{i,j}$ we then compute the value of $\tilde{z}(\sigma_{i,j})$. This step involves solving a maximization problem with respect to the GP hyper-parameters, where each step of the maximization process requires to run EP. After computing $\tilde{z}(\sigma_{i,j})$ for all the possible pairs $\{i, j\}$, we set σ to be the permutation $\sigma_{i,j}$ with the highest $\tilde{z}(\sigma_{i,j})$ that is also larger than $\tilde{z}(\sigma)$. When this condition is not met the algorithm stops. The corresponding pseudocode is shown in Algorithm 1. Its cost is determined by the $L(L-1)/2$ iterations performed by the inner *for* loop. In practice, however, we expect L to be small. Furthermore, the inner *for* loop can be trivially parallelized as shown in the pseudocode.

```
Input: Dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^n$ with $y_i \in \mathcal{L} = \{v_1, \ldots, v_L\}$.
 1: Select $\sigma$ at uniformly at random and compute $\tilde{z}(\sigma)$.
 2: Generate set $\mathcal{P}$ with all the 2-element subsets of $\{1, \ldots, L\}$.
 3: finished ← False.
 4: while Not finished do
 5:     finished ← True.
 6:     for every subset $\{i, j\}$ contained in $\mathcal{P}$ do
 7:         Generate $\sigma_{i,j}$ by swapping the elements $i$ and $j$ in $\sigma$.   parallel
 8:         Compute $\tilde{z}(\sigma_{i,j})$.
 9:     end for
10:     Find indexes $\{k, l\}$ such that $\tilde{z}(\sigma_{k,l}) \geq \tilde{z}(\sigma_{i,j})$ for any $i, j$.
11:     if $\tilde{z}(\sigma_{k,l}) > \tilde{z}(\sigma)$ then
12:         finished ← FALSE, $\sigma \leftarrow \sigma_{k,l}$, $\tilde{z}(\sigma) \leftarrow \tilde{z}(\sigma_{k,l})$
13:     end if
14: end while
15: return $\sigma$
```

Algorithm 1: Exhaustive search for learning σ

Method	**Auto**	**Boston**	**Fires**	**Yacht**
OR-L	1.000	1.000	0.333	1.000
OR-SE	0.840	0.968	0.427	1.000

Table 1: Average Kendall's tau.

Method	**OR-L**	**OR-SE**	**MC-L**	**MC-SE**
Auto	**-0.679**	-0.726	-0.874	-0.706
Boston	-0.901	**-0.795**	-0.957	-0.856
Fires	**-1.044**	-1.070	-1.050	-1.084
Yacht	-0.181	**-0.180**	-0.897	-0.207
Wins	4		0	

Table 2: Avg. Test LL. Ordinal.

Method	**OR-L**	**OR-SE**	**MC-L**	**MC-SE**
Glass	-0.224	-0.133	-1.264	**-0.096**
Iris	**-0.079**	-0.092	-0.331	-0.112
Thyroid	**-0.065**	-0.077	-0.187	-0.066
Wine	-0.205	-0.113	**-0.076**	-0.103
Wins	1.5		2.5	

Table 3: Avg. Test LL. Multi-class.

2.2 A multi-class classifier model

When there is no ordering among the entries in $\mathcal{L}$, the data can be described by a multi-class Gaussian process classifier [5]. In this case y_i is given by $y_i = \arg\max_{k \in \mathcal{L}} f_k(\mathbf{x}_i)$, where $f_{v_1}, \ldots, f_{v_L}$ are unknown noisy latent functions that need to be estimated. Define $\mathbf{f} = (f_{v_1}(\mathbf{x}_1), f_{v_1}(\mathbf{x}_2), \ldots, f_{v_1}(\mathbf{x}_n), \ldots, f_{v_L}(\mathbf{x}_1), f_{v_L}(\mathbf{x}_2), \ldots, f_{v_L}(\mathbf{x}_n))^{\mathrm{T}}$. The likelihood of $\mathbf{f}$ given $\mathbf{y} = (y_1, \ldots, y_n)^{\mathrm{T}}$ and $\mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^{\mathrm{T}}$ is $p(\mathbf{y}|\mathbf{X}, \mathbf{f}) = \prod_{i=1}^n [\prod_{k \neq y_i} \Theta(f_{y_i}(\mathbf{x}_i) - f_k(\mathbf{x}_i))]$. The prior for $\mathbf{f}$ is set as a product of zero-mean GPs with covariance matrices $\mathbf{K}_{v_1}, \ldots, \mathbf{K}_{v_L}$, which are obtained by evaluating the covariance function of each GP on $\mathbf{x}_1, \ldots, \mathbf{x}_n$ [5]. Define $\mathbf{f}_{v_l} = (f_{v_l}(\mathbf{x}_1), \ldots, f_{v_l}(\mathbf{x}_n))^{\mathrm{T}}$ for each $v_l \in \mathcal{L}$. The prior for $\mathbf{f}$ is $p(\mathbf{f}) = \prod_{l=1}^L \mathcal{N}(\mathbf{f}_{v_l}|\mathbf{0}, \mathbf{K}_{v_l})$. The prior and the likelihood are combined to get the posterior $p(\mathbf{f}|\mathcal{D}) = p(\mathbf{y}|\mathbf{X}, \mathbf{f})p(\mathbf{f})/p(\mathbf{y}|\mathbf{X})$. As in the previous model, this distribution is in general intractable. In [5, 4] EP is used to obtain a Gaussian approximation. Let $q(\mathbf{f})$ be this approximation. The predictive distribution for $y_\star$ is approximated as $p(y_\star|\mathbf{x}_\star, \mathcal{D}) \approx \int p(y_\star|f_\star)p(f_\star|\mathbf{f})q(\mathbf{f})d\mathbf{f}$, where $f_\star = (f_{v_1}(\mathbf{x}_\star), \ldots, f_{v_L}(\mathbf{x}_\star))^{\mathrm{T}}$, $p(f_\star|\mathbf{f})$ is a Gaussian conditional distribution and $p(y_\star|f_\star) = \prod_{k \neq y_\star} \Theta(f_{y_\star}(\mathbf{x}_i) - f_k(\mathbf{x}_i))$. This approximate distribution can be computed by solving a one-dimensional numerical integral.

The hyper-parameters of each GP under a Gaussian covariance function, *i.e.*, the length-scale, the amplitude and the additive Gaussian noise, can be found using gradient ascent. Specifically, EP also approximates in this case the gradients of $p(\mathbf{y}|\mathbf{X})$ with respect to these parameters. This allows to select optimal hyper-parameters via type-II maximum likelihood, as in the previous model.

3 Experiments

In this section the models and algorithms described in Section 2 are evaluated on real-world data.

3.1 Accuracy of the exhaustive search algorithm

The accuracy of the search algorithm from Section 2.1.1 is evaluated in different ordinal regression problems in which the correct ordering is known beforehand. We generate several of these problems by taking standard regression problems from the UCI repository and then discretizing the target variables using equal-probability bining. In this case, the bins divide the range of target values into a number of contiguous intervals with the same empirical probabilities. We considered the datasets Boston Housing, Forest Fires, Auto MPG and Yatch Hydrodynamics. In all cases, we fix the number of labels L to 5, except in Forest Fires, where we fix $L = 3$.

The accuracy of each method is computed in terms of the absolute value of Kendall's tau correlation coefficient between the true ranking of the labels and the ranking discovered by our algorithm. Kendall's tau is equal to the difference between the fractions of concordant and discordant pairs of labels, where two labels are concordant if the predicted ordering of the labels is the same as the correct ordering. A Kendall's tau value of 1 means that the predicted ranking is the same as the original one, while a value of -1 means that the predicted ranking is the opposite as the original one. Both results, 1 and -1, are equivalently good, as described in Section 2.1.

We consider ordinal regression models based on linear (OR-L) and squared exponential (OR-SE) covariance functions. Table 1 shows the average absolute value of Kendall's tau obtained by each method across 50 random partitions of the data in training sets with 2/3 of the instances and tests sets with the remaining 1/3. OR-L correctly identifies the true ordering in all cases except in the Forest Fires dataset. OR-SE is slightly less accurate in the Auto MPG dataset.

3.2 Accuracy in the identification of ordinal or categorical data

We also compare OR-L and OR-SE with the multi-class clasiffier (MC) described in Section 2.2 based on linear (MC-L) and squared exponential (MC-SE) covariance functions. We consider the ordinal regression tasks with known ordering described in Section 3.1, and four additional multi-class datasets from the UCI repository: Glass, Iris, New Thyroid, and Wine with 6, 3, 2 and 3 class labels, respectively. For each dataset, the data is split into training and test sets as in Section 3.1.

Method	**OR-L**	**OR-SE**	**MC-L**	**MC-SE**
Auto	-185.616	-172.205	-249.040	**-171.312**
Boston	-309.426	**-240.459**	-347.371	-241.394
Fires	-350.198	**-339.915**	-363.892	-340.411
Yacht	-58.853	-51.904	-213.102	**-51.737**
Wins	3		3	

Table 4: Average Evidence Ordinal.

Method	**OR-L**	**OR-SE**	**MC-L**	**MC-SE**
Glass	-75.219	-34.141	-93.092	**-26.971**
Iris	**-13.067**	-14.223	-37.993	-15.554
Thyroid	**-17.445**	-19.005	-41.358	-18.466
Wine	-31.217	-19.006	-20.652	**-15.172**
Wins	2		2	

Table 5: Average Evidence Multi-class.

Tables 2 and 3 show the average test log-likelihood obtained by each method on the ordinal and multi-class datasets, respectively. The results of the best method on each dataset are displayed in bold. We have also underlined those results that are statistically equivalent to the best result according to a paired t-test. The bottom rows in these tables show the number of times that an ordinal regression model or a multi-class classifier obtains the best performance, where ties count a 1/2. Table 2 shows that the ordinal regression models obtain the best predictive performance on the ordinal regression datasets, as one would expect. Table 3 shows that the multi-class models perform best in the multi-class datasets Glass and Wine and are equivalent to the best solution in New Thyroid. By contrast, Iris is better described by the ordinal regression models. Iris is a classic dataset for linear-discriminant analysis, which works by projecting the data on the real line. Therefore, it is unsurprising that ordinal regression models perform better in this case.

Finally, tables 4 and 5 show the average EP estimate of the model evidence obtained by each method on the different ordinal and multi-class datasets, respectively. Again, the best results have been highlighted in bold and those statistically equivalent have been underlined. In this case, the EP estimate of the model evidence seems to be a less accurate metric for distinguishing the type of discrete labels (ordinal or multi-class) than the average test log-likelihood reported in tables 2 and 3. The reason for this is probably the fact that we are already maximizing this estimate with respect to the permutation σ and the GP hyper-parameters, which can lead to biased estimates that are incorrectly too high.

4 Conclusions

Identifying the type of data variables in arbitrary datasets is required for automatic statistical data analysis and model building [6]. As an initial approach to this problem, we have focused on distinguishing categorical data from ordinal data. Our solution fits ordinal regression and multi-class classification models to the data and then evaluates their quality of fit. However, a ranking of the class labels has to be specified in advance in standard ordinal regression models. To avoid this, we propose an exhaustive search procedure that automatically selects an optimal ranking of the available labels. Our experiments show that, when linear models are used, we can correctly identify the true ranking most of the times. However, the ranking recovery process is less accurate when non-linear models are used. These flexible models can learn complex functions that compensate for an incorrect ordering, decreasing the ability to identify the actual ranking. Two performance metrics to discriminate between ordinal and multi-class models are used: The average test log-likelihood (TLL) and the approximation of the model evidence (AME) returned by EP. While TLL can successfully identify the correct type (ordinal or categorical) of the data, AME fails to achieve this.

Acknowledgement. J.M.H.L acknowledges support from the Rafael del Pino Foundation. D.H.L. is supported by MCyT and by CAM (projects TIN2010-21575-C02-02, TIN2013-42351-P and S2013/ICE-2845).

References

[1] C. M. Bishop. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

[2] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. In *Journal of Machine Learning Research*, pages 1019–1041, 2005.

[3] R. B. Grosse, R. Salakhutdinov, W. T. Freeman, and J. B. Tenenbaum. Exploiting compositionality to explore a large space of model structures. In *Conf. on Unc. in Art. Int. (UAI)*, 2012.

[4] D. Hernández-Lobato, J. M. Hernández-Lobato, and P. Dupont. Robust multi-class Gaussian process classification. In *Advances in neural information processing systems*, pages 280–288, 2011.

[5] H.-C. Kim and Z. Ghahramani. Bayesian Gaussian process classification with the EM-EP algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12):1948–1959, 2006.

[6] J. R. Lloyd, D. Duvenaud, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic construction and Natural-Language description of nonparametric regression models. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2014.

[7] T. P. Minka. *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology, 2001.

[8] U. Paquet, B. Thomson, and O. Winther. A hierarchical model for ordinal matrix factorization. *Statistics and Computing*, 22(4):945–957, 2012.

[9] C. Rasmussen and C. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[10] I. Sassoon, J. Keppens, and P. McBurney. Towards argumentation for statistical model selection. *comma2014.arg.dundee.ac.uk*, 2014.

[11] M. Schmidt and H. Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, Apr. 2009.

[12] M. Seeger. Expectation propagation for exponential families. Technical report, 2005.

[13] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *Proceedings of the 18th international conference on World wide web*, pages 111–120. ACM, 2009.

[14] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proc. Int. Conf. on Knowledge Discovery and Data Mining*, KDD '13, pages 847–855, New York, NY, USA, 2013. ACM.

[15] L. Todorovski and S. Dzeroski. Declarative bias in equation discovery. In *International Conference on Machine Learning*, pages 376–384, 1997.