LEARNING WITH RANDOM INPUTS

ROBERT WOLSTENHOLME

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

Given some observations of dependent and independent and independent variables, there are many ways to learn some relationship between them. This essentially falls under the category of supervised learning. However, if we do not directly observe the independent variables but instead simply the distribution they have come from we have to modify some techniques so that they become tractable. This can arise when the true values of the independent variables are essentially hidden and we only have an estimate of their distribution.

We will examine the case for a softmax regression with a multivariate normal distribution for the independent varaibles. Two different approximations are used to the objective function to make the optimisation tractable. Then we look to performing the optimisation in the Scikit-Learn and Tensorflow libraries.

* *Department of Biology, University of Examples, London, United Kingdom*
[1] *Department of Chemistry, University of Examples, London, United Kingdom*

INTRODUCTION

Consider the known multivariate normal parameters at time t

$$\mu_t \in \mathbb{R}^n \text{ and } \Sigma_t \in \mathbb{R}^{n\times n}$$

such that we have a random variable

$$\mathbf{X}_t \sim \text{MVN}(\mu_t, \Sigma_t).$$

Remark 1. The actual realisations $\mathbf{x}_t$ from $\mathbf{X}_t$ will never be observed and essentially represent independent variables.

Also consider an observation vector at time t,

$$\mathbf{y}_t \in \{0,1\}^k$$

such that $\sum_{i=0}^{k} y_{i,t} = 1$ i.e. $\mathbf{y}_t$ contains a unique entry equal to 1 and the rest are 0. It therefore represents our dependent variable and is a realisation from a multinomial distribution conditional on $\mathbf{X}_t$,

$$\mathbf{Y}_t|\mathbf{X}_t = \mathbf{x}_t \sim \text{Multinomial}(g_W(\mathbf{x}_t)),$$

where the ith component of $g_W(\mathbf{x}_t)$ is

$$[g_W(\mathbf{x}_t)]_i = \frac{\exp(\mathbf{w}_i^T\mathbf{x}_t)}{\sum_{j=1}^{k}\exp(\mathbf{w}_j^T\mathbf{x}_t)}.$$

Hence $g_W(\cdot)$ is a softmax transformation with weight vectors $\mathbf{w}_i \in \mathbb{R}^n$ which we write as a matrix

$$W = [\mathbf{w}_1, \ldots, \mathbf{w}_k] \in \mathbb{R}^{n\times k}.$$

Our goal is that given some sequence of observations $\{\mathbf{y}_t\}$ and observed distribution parameters $(\{\mu_t\},\{\Sigma_t\})$, to find optimal weight matrix W.

Remark 2. While we think of all components of $\mathbf{X}_t$ being stochastic, non-random components are of course dealt with by 0s in the covariance matrix.

OPTIMISATION

In order to write the optimisation, we must first write the likelihood function, given some values for W, of observing the $\mathbf{y}_t$ value.

The conditional probability is (from a multinomial distribution),

$$\Pr(\mathbf{Y}_t = \mathbf{y}_t|\mathbf{X}_t = \mathbf{x}_t; W) = g_W(\mathbf{x}_t)^T\mathbf{y}_t$$

and so the likelihood is

$$\Pr(\mathbf{Y}_t = \mathbf{y}_t; W) = \int_{\mathbf{x}_t}\left[g_W(\mathbf{x}_t)^T\mathbf{y}_t\right] f_{\mathbf{X}_t}(\mathbf{x}_t)\,d\mathbf{x}_t$$

where $f_{\mathbf{X}_t}(\mathbf{x}_t)$ is a multivariate normal pdf with parameters μ_t and Σ_t and the '$;W$' represents the fact the probability depends on deterministic matrix W.

Expanding this over all values of t $(1,\ldots,T)$, we have likelihood

$$L(W) = \prod_{t=1}^{T} \Pr(Y_t = y_t; W)$$

$$= \int_{x_1} \cdots \int_{x_T} \prod_{t=1}^{T} \left[g_W(x_t)^T y_t \right] f_{X_1,\ldots,X_T}(x_1, \ldots, x_T) dx_1 \ldots dx_T.$$

Under the assumption of independence between the $\{X_t\}$ (which in reality is very often not going to be true) this can be written

$$L(W) = \prod_{t=1}^{T} \int_{x_t} \left[g_W(x_t)^T y_t \right] f_{X_t}(x_t) dx_t$$

and the log likelihood can be written

$$\log(L(W)) = LL(W) = \sum_{t=1}^{T} \log \left[\int_{x_t} \left[g_W(x_t)^T y_t \right] f_{X_t}(x_t) dx_t \right].$$

The optimisation to solve can then be written as

$$\hat{W} = \arg\max_W LL(W) + \lambda r(W)$$

for some $\lambda \in \mathbb{R}$ and regularisation function $r : \mathbb{R}^{n \times k} \to \mathbb{R}$.

Remark 3.

1. Despite the fact the independence between the $\{X_t\}$ is often not true, there is very little we can do. If we don't know the joint distribution of course there is nothing we can do, but even if it is known, the already intractable computation becomes even harder to approximate.
2. The integrals within the likelihood are expectations over X_t i.e.

$$\mathbb{E}_{X_t} \left[g_W(x_t)^T y_t \right] = \int_{x_t} \left[g_W(x_t)^T y_t \right] f_{X_t}(x_t) dx_t.$$

3. In general the integral is intractable and we have to use numerical approximations to evaluate and get a gradient for $LL(W)$.
4. For the regularisation, we will use the L2 norm i.e.

$$r(W) = \|W\|_2 = \sum_{i,j} W_{ij}^2.$$

Approximation 1

First write the log likelihood as

$$LL(W) = \sum_{t=1}^{T} \log \mathbb{E}_{X_t} \left[g_W(x_t)^T y_t \right].$$

Now, $g_W(x_t)^T y_t$ is not convex or concave. This can be seen by considering the sigmoid function $y = \frac{e^x}{1+e^x}$ which has second derivative $\frac{\partial^2 y}{\partial x^2} = (1-y)(1-2y)y$. This is negative for $x > 0$ and positive for $x < 0$.

We cannot therefore use Jensen's inequality to get a lower/upper bound but we still attempt the approximation by swapping the expectation into the function

$$\mathrm{LL}^{(1)}(W) = \sum_{t=1}^{T} \log\left[g_W(\mathbb{E}_{X_t}[x_t])^T y_t\right] = \sum_{t=1}^{T} \log\left[g_W(\mu_t)^T y_t\right].$$

The optimsation

$$\hat{W}^{(1)} = \arg\max_W \mathrm{LL}^{(1)}(W) + \lambda r(W)$$

is simply a standard constrained softmax regression and we solve it using:

1. Scikit-Learn: sklearn.linear_model.LogisticRegression class with multi_class = 'multinomial'.
2. Tensorflow: Coded constrained softmax regression.

Remark 4. The above approximation completely discard the information provided by the $\{\Sigma_t\}$ covariance matrices.

Approximation 2

For our second approximation, we approximate the integrals as finite sums by samping from the f_{X_t} distribution i.e.

$$\int_{x_t}\left[g_W(x_t)^T y_t\right] f_{X_t}(x_t)\,dx_t \approx \frac{1}{|\mathbb{X}_t|}\sum_{x\in\mathbb{X}_t} g_W(x)^T y_t$$

where $\mathbb{X}_t$ is a set of samples from the distribution with pdf f_{X_t}.
Therefore we write

$$\mathrm{LL}^{(2)}(W) = \sum_{t=1}^{T} \log\left[\frac{1}{|\mathbb{X}_t|}\sum_{x\in\mathbb{X}_t} g_W(x)^T y_t\right].$$

Note that this has gradient

$$\frac{\partial \mathrm{LL}^{(2)}(W)}{\partial W} = \sum_{t=1}^{T} \frac{\sum_{x\in\mathbb{X}_t} \frac{\partial g_W(x)^T y_t}{\partial W}}{\sum_{x\in\mathbb{X}_t} g_W(x)^T y_t}.$$

Remark 5.

1. The size of the sets $\mathbb{X}_t$ no longer matters once we consider the derivative of $\mathrm{LL}^{(2)}(W)$ (which is what will be used in gradient ascent/descent). Hence all that matters is we have enough samples for a 'good' approximation to each integral. Even then the definition of good really is whether the optimisation converges to the correct values and if we have enough observations, it may be enough to simply have $|\mathbb{X}_t| = 1$!
2. In some border cases however we must make very sure we have enough observations in our set for a good prediction. Consider $n = 1$, $k = 2$ and $g_W(x) = [I(Wx > 2), I(Wx \leqslant 2)]^T$. Then for $x \leqslant W/2$ we predict class 1 and for $x > W/2$ we predict class 0. Hence if the true x value caused a class 0 prediction but a single sampled value caused a class 1 prediction, then the true parameter value W will cause an estimated log likelihood of $-\infty$! Even worse, if this happened multiple times, it may be impossible to have any non degenerate value for the estimated value of W. Of course the model we use for g_W is a multinomial

model so the above definition is impossible. However we can get similar situation arising if the distribution of x has a very large variance. For example, the observed class may be class 0, but the single sampled value may have a very large magnitude making the prediction of class 1 occur with very high probability. This can be enough to significantly affect the overall log likelihood and hence cause a bad W estimation.

3. If we have $|\mathbb{X}_t| = 1$ for all t, then $LL^{(2)}(W)$ is in the form of a standard softmax regression, like $LL^{(1)}(W)$, with the only difference being that $LL^{(1)}(W)$ uses the mean value of the observations μ_t and $LL^{(2)}(W)$ uses a sample from $MVN(\mu_t, \Sigma_t)$.

Hence, we solve $LL^{(2)}(W)$ using:

1. Scikit-Learn: sklearn.linear_model.LogisticRegression class with multi_class = 'multinomial' only when $|\mathbb{X}_t| = 1$ for all t.
2. Tensorflow: For varying sizes of the $\mathbb{X}_t$.