A Hybrid Pareto Model for Conditional Density Estimation of Asymmetric Fat-Tail Data

Julie Carreau
Computer Science Dept.
University of Montreal
Julie.Carreau@umontreal.ca

Yoshua Bengio
Computer Science Dept.
University of Montreal
Yoshua.Bengio@umontreal.ca

Abstract

We propose an estimator for the conditional density $p(Y|X)$ that can adapt for asymmetric heavy tails which might depend on X. Such estimators have important applications in finance and insurance. We draw from Extreme Value Theory the tools to build a hybrid unimodal density having a parameter controlling the heaviness of the upper tail. This hybrid is a Gaussian whose upper tail has been replaced by a generalized Pareto tail. We use this hybrid in a multi-modal mixture in order to obtain a nonparametric density estimator that can easily adapt for heavy tailed data. To obtain a conditional density estimator, the parameters of the mixture estimator can be seen as functions of X and these functions learned. We show experimentally that this approach better models the conditional density in terms of likelihood than compared competing algorithms : conditional mixture models with other types of components and multivariate nonparametric models.

1 Introduction

The purpose of this paper is to introduce a new nonparametric model for conditional density estimation when the underlying density $p(Y|X)$ is asymmetric and heavy-tailed. This task is meaningful in a number of application domains where one wishes to make predictions about a random variable Y given an observed X, when the distribution of Y given X can have fat tails, be multimodal or asymmetric.

Practical application domains where such distributions occur include financial and insurance modeling. In finance, estimating the predictive conditional distribution of the profit and loss (P&L) of a portfolio is central for portfolio and risk management. Finance practitioners most often use the so-called Value-at-Risk (VaR) which is a quantile of the P&L distribution. However, the only information carried by the VaR is a bound on the lowest loss attainable with a given probability. It doesn't provide any information as how bad can things go below that bound. The Conditional VaR (CVaR) has been proposed to resolve this issue [1]; it measures the expected loss given that the VaR has been exceeded. A density estimator providing a model of the tail of distribution is thus required to compute the CVaR. Besides, several authors have already provided strong evidence for the presence of fat tails in stock returns data [2]-[3].

In insurance applications, companies are interested in modeling the distribution of the claims given a client profile. Insurers cover losses that fall in a given interval, called the reinsurance layer, by resorting to reinsurance companies. This reinsurance layer is meant to protect against a range of large claims; therefore, good estimates of the tail of the claim distribution are needed to evaluate the probability of losses in the reinsurance layer [4]. In both application domains, the dimension of the input can be as much as hundreds.

In general, if one incurs a loss $l(Y, X, d)$ when decision d is taken and Y and X are realized, then one should choose d to minimize the expected loss:

$$\int l(Y, X, d)p(Y|X)dY.$$

When l is not known precisely ahead of time, it is reasonable to look for an estimator $\phi_\theta(x, y)$ of $p(Y = y|X = x)$ that is close to the true one in the sense of the Kullback-Leibler (KL) divergence. However, since the true conditional density is unknown, one can consider the KL divergence with respect to the empirical distribution, which is equivalent to the conditional log-likelihood.

2 Extreme value theory

The so-called *Peaks over Thresholds* (PoT) method was developed to estimate the tail of a univariate distribution [5]. A suitable threshold needs to be established and the exceedances above that threshold are assumed to follow the Generalized Pareto (GP) distribution. The density function of the GP is given in equation 1 where $y \geq 0$ when $\xi \geq 0$ and $0 \leq y \leq -\beta/\xi$ when $\xi < 0$. The location of the GP can be changed by replacing y by $y - \alpha$ in the equations.

$$g_{\xi;\beta}(y) = \begin{cases} \frac{1}{\beta}(1 + \xi\frac{y}{\beta})^{-1/\xi - 1} & \text{if } \xi \neq 0, \\ \frac{1}{\beta}e^{-\frac{y}{\beta}} & \text{if } \xi = 0. \end{cases} \quad (1)$$

Justification for the use of the GP to model exceedances of a random variable Y comes from Pickands theorem [6]. This theorem states that the distribution of the normalized exceedances of Y over a threshold converges to a GP distribution as the threshold tends to the right endpoint of the distribution if and only if Y is in the maximum domain of attraction of an Extreme-Value distribution. This condition is a fairly general one and encompasses all well-known distributions.

The parameter ξ of the GP controls the thickness of the tail. When $\xi > 0$, the GP can account for heavy tails (e.g. Pareto, α-stable and Student t distributions). When $\xi = 0$, the GP can model exponential tails (e.g. Gaussian, Exponential and Log-Normal distributions). Finally, when $\xi < 0$, the GP has a finite tail (e.g. uniform or Beta distribution). Examples of the GP density for various tail parameters are given in Figures 1 and 2.

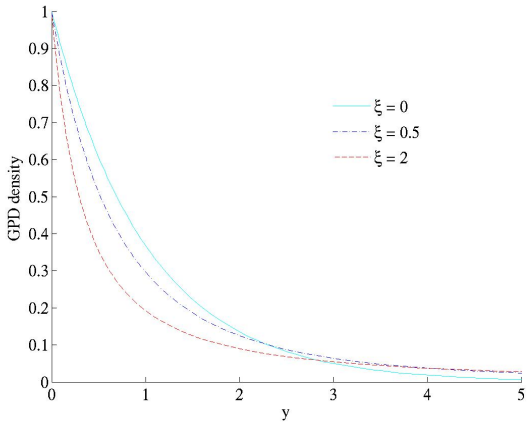

Figure 1: *GP density from light ($\xi = 0$) to heavy tail ($\xi = 2$).*

The choice of threshold above which the exceedances are used for inference of the GP parameters is subject to a bias-variance trade-off. If the threshold is too high, very few points enter in the estimation of the GP parameters, making the estimation subject to high variance. If the threshold is too low, the GP approximation of the tail will have a large bias since, according to Pickands theorem, convergence occurs as the threshold approaches the right endpoint of the distribution. Some methods have been proposed for threshold selection but there is no consensus on which one to use.

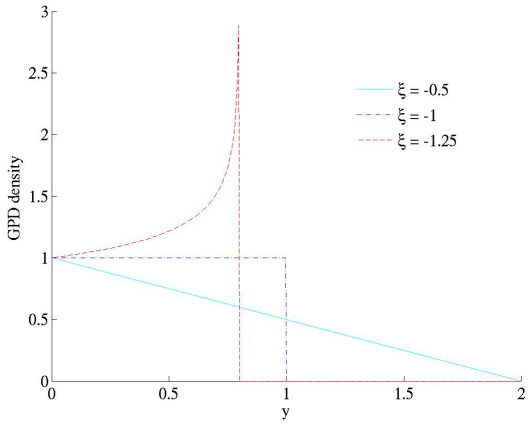

Figure 2: *GP density for finite tails ($\xi < 0$).*

3 Hybrid Pareto distribution

Since the GP is only suited to model the tail of a distribution, we propose the **hybrid Pareto distribution** as a smooth extension of the GP to the whole real axis. This new distribution is built by stitching a GP tail to a Gaussian, while enforcing continuity of the resulting density and of its derivative. The threshold is then defined as the junction point of the Gaussian and the GP and is computed implicitly as a function of the hybrid parameters. Let α be the threshold and let $f_{\mu;\sigma}(y) = 1/(\sqrt{2\pi}\sigma)\exp(-(y-\mu)^2/(2\sigma^2))$ be the Gaussian density function with parameters μ and σ, $g_{\xi;\beta}(y - \alpha)$ be the GP density of equation 1 with parameters ξ and β located above α. The continuity constraint on the density at α means that $f_{\mu;\sigma}(\alpha) = g_{\xi;\beta}(0)$ which gives:

$$\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(\alpha-\mu)^2}{2\sigma^2}\right) = \frac{1}{\beta} \quad (2)$$

Continuity of the derivative of the density at α means that $f'_{\mu;\sigma}(\alpha) = g'_{\xi;\beta}(0)$, which yields:

$$-\frac{(\alpha-\mu)}{\sqrt{2\pi}\sigma^3}\exp\left(-\frac{(\alpha-\mu)^2}{2\sigma^2}\right) = -\frac{(1+\xi)}{\beta^2} \quad (3)$$

Combining equations 2 and 3, we get that:

$$\frac{1+\xi}{\beta} = \frac{\alpha-\mu}{\sigma^2} \quad (4)$$

We set ξ, μ and σ as the free parameters and we let α and β be functions of these free parameters. We then

solve equations 2 and 4 for ξ, μ and σ. For this, we make use of the Lambert W function: given an input z, $w = W(z)$ is such that $z = we^w$. We use a numerical algorithm of order four to find the zero of $z - we^w$ [7]. We let $z = (1+\xi)^2/2\pi$. The dependent parameters α and β are obtained by the following formulae:

$$\beta(\xi,\sigma) = \frac{\sigma(1+\xi)}{\sqrt{W(z)}} \quad \alpha(\xi,\mu,\sigma) = \mu + \sigma\sqrt{W(z)}.$$

For $\xi > -1$, the hybrid Pareto density function is given by:

$$h_{\xi;\mu;\sigma}(y) = \begin{cases} \frac{1}{\gamma} f_{\mu;\sigma}(y) & \text{if } y \leq \alpha, \\ \frac{1}{\gamma} g_{\xi;\beta}(y-\alpha) & \text{if } y > \alpha \end{cases}$$

where γ is the appropriate re-weighting so that the density integrates to one and is given by:

$$\gamma(\xi) = 1 + \frac{1}{2}\left(1 + Erf\left(\sqrt{W\left(z\right)/2}\right)\right),$$

where $Erf(\cdot)$ is the error function $Erf(z) = \frac{2}{\sqrt{\pi}}\int_0^z e^{-t^2}dt$, which can be readily approximated numerically to high precision in standard ways.

Figures 3 and 4 illustrate the density and the log-density of the Hybrid Pareto distribution.

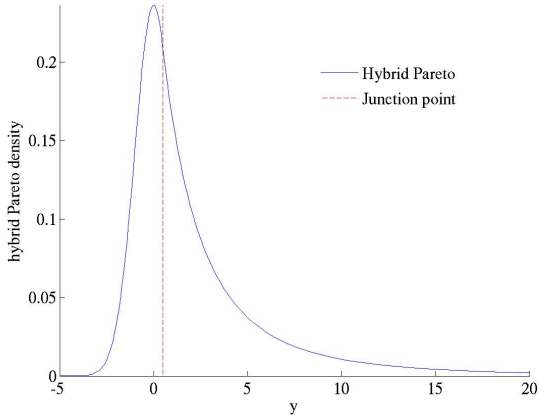

Figure 3: *Hybrid Pareto density with parameters* $\xi = 0.4$, $\mu = 0$ *and* $\sigma = 1$.

4 Mixture models

A popular model in density estimation is the mixture of Gaussians. When the number of components is well chosen according to the number of observations, the mixture of Gaussians has nice convergence properties as a nonparametric estimator (see [8] for instance). However, if the tail of the generative distribution is heavy, i.e. extreme observations can occur far away in the tail, good empirical results can often be obtained by considering a mixture of Gaussians in which one of the Gaussians has a very large σ, that serves to capture the points far away. One disadvantage of this approach

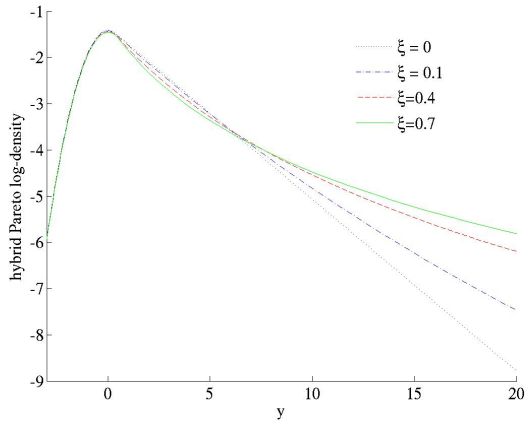

Figure 4: *Hybrid Pareto log-density for various tail parameters and in all cases* $\mu = 0$ *and* $\sigma = 1$.

is that the density model will only account for observed extremes and will still underestimate the density of the upper tail (this is specially true for small training sets). Another drawback is that by using symmetric components in the mixture, the lower tail tends to be overestimated.

We propose to combine the advantages of the mixture model, a flexible nonparametric model, and of the GP, capable of approximating arbitrarily well the tail of most distributions, by using a mixture of hybrid Pareto distributions. Using directly the GP as a component of a mixture model would be impractical since the GP is zero below the threshold that determines its location. Besides, we bypass the need of selecting a threshold inherent in the PoT methodology. The threshold is embedded in the component of the mixture having the heaviest tail and is thus determined implicitly through inference of the whole mixture.

5 Conditional mixture models

We build a conditional density estimator $\phi_\theta(x,y)$ based on the mixture model by modeling the mixture parameters for the density of y as functions of the input x. The estimator is given in equation 5 when using hybrid Pareto components.

$$\phi_\theta(x,y) = \sum_{i=1}^{m} \pi_i(x) h(y; \xi_i(x), \mu_i(x), \sigma_i(x)) \qquad (5)$$

Neural networks and linear or log-linear models are convenient classes of functions to compute the parameters of the output density, that is to implement the functions $\pi_i(\cdot)$, $\xi_i(\cdot)$, $\mu_i(\cdot)$, and $\sigma_i(\cdot)$, given an x, and they have been used successfully for similar tasks [9]. However any parametrized class of functions which can be trained using the gradient with respect to parameters can be used. This is because the estimation of this function is obtained through maximizing the mixture

conditional log-likelihood. By increasing the number of hidden units, neural networks can in principle approximate any continuous function. The neural network output formulae are given in equations 6 and 7 and the resulting architecture is depicted in Figure 5.

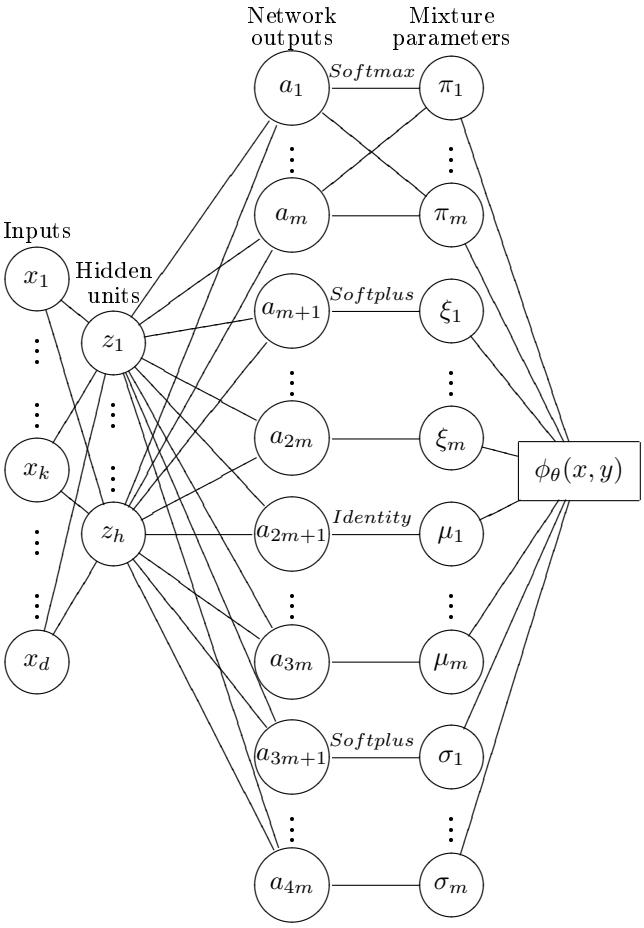

Figure 5: *Conditional Mixture Model: A feed-forward neural network with one hidden layer and hyperbolic tangent activation function is used to predict input dependent mixture parameters. Appropriate transfer functions at the network outputs are used to impose range constraints.*

The outputs of the neural network, the a_j in Figure 5, are linear combinations of the hidden unit outputs:

$$a_j(x) = b_j + \sum_{i=1}^{h} w_{ji} z_i(x), \tag{6}$$

where

$$z_i(x) = \tanh\left(c_i + \sum_{k=1}^{d} v_{ik} x_k\right) \tag{7}$$

are hidden unit activations. The number of hidden units h controls the capacity (the number of parameters and the flexibility of the model) and d is the dimension of the input x. By convention, setting $h = 0$ results in a linear model ($a_j(x) = b_j + \sum_{k=1}^{d} w_{ji} x_k$).

The transfer function at the output of the neural network is chosen so as to impose range constraints on the parameters of the mixture. The mixture weight $\pi_i(x) = P(i|X = x)$ is the probability that the i^{th} component is responsible for generating y given x. It must therefore be positive and all the $\pi_i(\cdot)$'s must sum to one. This is ensured by a *softmax* function: $\pi_i(x) = \exp(a_i(x))/\sum_j \exp(a_j(x))$, where the $a_j(\cdot)$'s, $j = 1 \ldots m$ are the neural network outputs dedicated to the priors $\pi_j(\cdot)$'s. A *softplus* function is used to guarantee the positivity of the $\sigma_i(\cdot)$'s:

$$\text{softplus}(x) = \log(1 + e^x).$$

The *softplus* has been introduced by [10]; like the exponential, the *softplus* has a positive range but it grows slower than the exponential which makes numerical optimization more stable[1]. In the experiments we have also constrained the $\xi_i(\cdot)$'s to be positive with a *softplus*. The $\mu_j(\cdot)$'s are unconstrained $a_j(\cdot)$'s.

The free parameters of the conditional mixture model are thus the neural network parameters $\theta = (b, c, v, w)$. These are determined by minimizing the empirical negative log-likelihood:

$$l(\theta) = -\sum_{i}^{n} \log \phi_\theta(x_i, y_i). \tag{8}$$

We use a conjugate gradient descent algorithm for the optimization. For each example, we obtain the gradient of the empirical negative log-likelihood with respect to θ in two steps:

1. First compute derivatives of l with respect to a_j, $j = 1, \ldots, 4m$, (the outputs of the neural network before the output transfer function, see Figure 5).

2. Back-propagate gradients as usual through the neural network in order to obtain $\frac{\partial l}{\partial \theta}$. Implicitly, the resulting gradient is therefore obtained through
$$\frac{\partial l}{\partial \theta} = \sum_j \frac{\partial l}{\partial a_j} \frac{\partial a_j}{\partial \theta}.$$

Since the derivative in step 2 is standard in neural network applications (see [9]), we describe only the derivative in step 1. Let $l = -\log(\phi_\theta(x, y)) = -\log(\phi_{\theta(x)}(y))$ be the value of the error function for example (x, y) where $\phi_\theta(x, y)$ is given in equation 5 and $\theta(x) = (\boldsymbol{\pi}(x), \boldsymbol{\xi}(x), \boldsymbol{\mu}(x), \boldsymbol{\sigma}(x))$ are the mixture

[1] Note that if $x > 0$, we have $\text{softplus}(x) = x + \log(1 + e^{-x})$ and asymptotically we have that $\lim_{x \to \pm\infty} \text{softplus}(x) \to x^+$ where x^+ denotes the positive part of x

parameters for input x where, for instance, $\boldsymbol{\pi}(x)$ denotes the vector of length m of mixture weights. From Figure 5, we see that the derivative $\frac{\partial l}{\partial a_j}$ can be separated into two different cases depending on the value of j:

- If $1 \leq j \leq m$, a_j is one of the outputs controlling the priors and its derivative can be expressed as:

$$\frac{\partial l}{\partial a_j} = \frac{\partial l}{\partial \phi_{\theta(x)}(y)} \sum_{k=1}^{m} \frac{\partial \phi_{\theta(x)}(y)}{\partial \pi_k} \frac{\partial \pi_k}{\partial a_j}. \quad (9)$$

- On the other hand, if $m+1 \leq j \leq 4m$, a_j governs one of the hybrid Pareto component parameter $\theta_j(x)$ and its derivative is simpler:

$$\frac{\partial l}{\partial a_j} = \frac{\partial l}{\partial \phi_{\theta(x)}(y)} \frac{\partial \phi_{\theta(x)}(y)}{\partial \theta_j(x)} \frac{\partial \theta_j(x)}{\partial a_j}. \quad (10)$$

Each partial derivative in equations 9 and 10 is developed next. In both equations we have $\frac{\partial l}{\partial \phi_{\theta(x)}(y)} = -\frac{1}{\phi_{\theta(x)}(y)}$. When the derivative is taken with respect to one of the mixture weights, we have, for $1 \leq j \leq m$:

$$\frac{\partial \phi_{\theta(x)}(y)}{\partial \pi_j} = h(y; \xi_j(x), \mu_j(x), \sigma_j(x)).$$

Differentiating with respect to the hybrid Pareto parameters $\theta_j(x)$, for $m+1 \leq j \leq 4m$ and $i = j \mod 3$,

$$\frac{\partial \phi_{\theta(x)}(y)}{\partial \theta_j(x)} = \pi_i(x) \frac{\partial}{\partial \theta_j(x)} h(y; \xi_j(x), \mu_j(x), \sigma_j(x)).$$

The derivative of the priors and of the mixture parameters with respect to the network outputs are

$$\begin{aligned}
\frac{\partial \pi_k}{\partial a_j} &= \begin{cases} \pi_j(1-\pi_j) & \text{if } j = k \\ -\pi_k \pi_j & \text{if } j \neq k \end{cases} \\
\frac{\partial \theta_j(x)}{\partial a_j} &= 1 - \exp(-\xi_i(x)) \quad \text{if } \theta_j(x) = \xi_i(x) \\
\frac{\partial \theta_j(x)}{\partial a_j} &= 1 \quad \text{if } \theta_j(x) = \mu_i(x) \\
\frac{\partial \theta_j(x)}{\partial a_j} &= 1 - \exp(-\sigma_i(x)) \quad \text{if } \theta_j(x) = \sigma_i(x).
\end{aligned}$$

6 Experiments

The Student t and Log-Normal distributions are robust alternatives to the Gaussian distribution. The Student t distribution is a symmetric distribution whose tail heaviness is controlled by a parameter ν called the number of degrees of freedom. The tail of the Student t can be modeled by a GP with tail parameter $\xi = 1/\nu$. The Log-Normal distribution is asymmetric with an upper tail slightly heavier than the Gaussian tail. However, the Log-Normal tail, like the Gaussian tail, can be modeled by a GP with tail parameter $\xi = 0$. Based on those considerations, we compared our conditional mixture of hybrid Paretos **CMM-H** with conditional mixture models (also with a neural network to predict parameters) with different types of components **CMM-G** for Gaussian,**CMM-T** for Student t and **CMM-L** for Log-Normal. We also compared **CMM-H** with multivariate models on (X, Y) : Gaussian mixture **MM-G** and Parzen window estimator **MM-P** , transformed to estimate $p(Y|X) = p(X, Y)/p(X)$.

The conditional mixture models are all trained by conjugate gradient descent to minimize the negative log-likelihood. The multivariate mixture of Gaussians is trained by the EM algorithm. Since the optimization for all models may lead to local minima, during learning, each model is re-initialized randomly 5 times and the optimization is re-started accordingly (this is done only on the real data sets). We keep the parameters that gave the smallest training error.

The conditional mixture models (regardless of the type of components), have two hyper-parameters: the number of hidden units n_h for the neural network and the number of components m of the mixture. The multivariate mixture of Gaussians has one hyper-parameter, the number of components. The variance-covariance matrix is chosen to be diagonal except for the experiments on the artificial data sets where the matrix is full. The variance-covariance matrix for the multivariate Parzen window estimator has two hyper-parameters, λ_x which controls the input variance and λ_y which controls the target variance. We use a validation set, independent of the training set, to select the optimal values for the hyper-parameters of each model. The complexity of each model is thus optimized fairly.

6.1 Artificial data sets

We generated data from a conditional Fréchet distribution whose parameters are made conditionally dependent on the input by using either a linear ($f(X) = aX + b$) or a sine-shaped ($f(X) = c\sin(aX + b) + d$) functional. To easily control the range of the Fréchet parameters, we chose X scalar and uniform on $[0, 1]$. The tail index of the Fréchet was chosen to be either in the interval $[1/6, 1/4]$ to allow for moderately heavy tails or in the interval $[1/2, 2]$ to allow for heavier tails. We have thus four distinct generative models.

For this experiment, the training and the test set both had 500 observations. To capture the performance relative to the generative model, we measure the performance with the out-of-sample relative log-likelihood:

$$\mathcal{RLL}(\mathcal{D}) = -\sum_{i=1}^{n} \log\left(\frac{p(y_i|x_i)}{\phi_\theta(x_i, y_i)}\right),$$

where $p(\cdot)$ is the density function of the generative model, $\phi_\theta(\cdot)$ is the density function of the estimator and the sum is over the test set $\mathcal{D}$. The smaller the RLL criterion is, the better the estimator is per-

forming. We generated 20 pairs of training and test sets. Table 1 presents a sample overview of the results; it shows the average out-of-sample RLL along with its standard error over the 20 test sets. The generative model is the conditional Fréchet with linearly dependent parameters and moderately heavy tail ($\xi \in [1/6, 1/4]$). All conditional mixture models have one hidden unit (which theoretically should be sufficient since the functional dependence is linear) and the number of components is allowed to increase.

Table 1: *Average out-of-sample RLL (standard err.) between predicted density of the estimators and the generative model - conditional Fréchet distribution. Smaller values mean better estimators.*

m	CMM-H	CMM-G	
1	10.0 (6.6)	161.6 (28.3)	
2	11.0 (6.0)	53.9 (20.7)	
4	9.8 (5.4)	36.8 (21.7)	
8	11.9 (5.3)	29.6 (13.5)	
m	CMM-T	CMM-L	
1	134.9 (36.8)	112.7 (18.1)	
2	37.3 (14.5)	37.6 (25.9)	
4	30.4 (16.8)	19.3 (9.0)	
8	30.8 (15.8)	20.1 (8.3)	
m	MM-G	(λ_x, λ_y)	MM-P
1	179.1 (28.9)	$(10^{-3}, 10^{-3})$	292.3 (53.8)
2	202.8 (96.8)	$(10^{-3}, 10^{-2})$	130.8 (37.1)
4	187.9 (68.1)	$(10^{-2}, 10^{-3})$	307.0 (52.4)
8	221.0 (86.0)	$(10^{-2}, 10^{-2})$	228.2 (34.7)

Table 1 shows that the conditional mixture with hybrid Pareto components has the smallest RLL even with only one component in the mixture. The complete results for all four data sets give a similar insight.

6.2 Insurance data set

In a second set of experiments we used real insurance data graciously provided by an anonymous insurance company. The complete distribution of the claims include a mass point at 0. One way to deal with this is to use a probabilistic classifier that predicts, given a client profile X, the most probable class (claim $= 0$, claim > 0). For the second class, we need to estimate $p(\text{claim}|X, \text{claim} > 0)$ and this is the part of the problem we addressed here. This is why the records used in the experiments are only for policies that had a non-zero claim. Data from one year, containing 54119 records with positive claims, were used for training, hyper-parameter selection, testing and model comparison. The dependent variable Y is the claim amount divided by the duration of the policy. The input variable X is a vector of 140 numbers, mostly binary indicators, describing the client profile. The numeric inputs have been standardized. Principal component analysis has been applied on the input variable to reduce dimensionality: enough components (between 61 and 69 depending on the training set size) were retained to explain 90% of input variance. The histogram of the positive claims smaller than 5000$ of Figure 6 illustrates the unconditional distribution; it shows that the distribution has at least two modes. This distribution is strongly skewed: more than 75% of the claim amounts are smaller than the average claim amount.

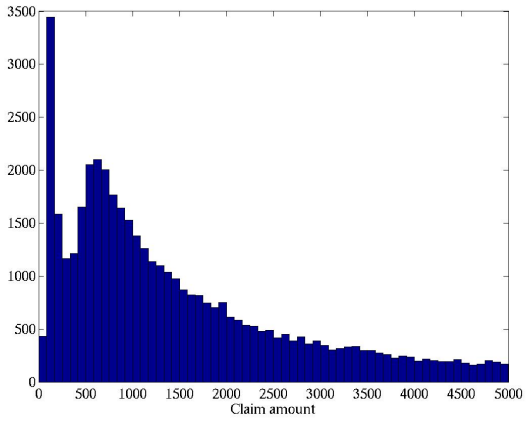

Figure 6: *Histogram of the positive claims smaller than 5000$ for the insurance data set.*

The performance is measured by comparing how a competing estimator performs relative to the proposed conditional mixture of hybrid Paretos. Let (x, y) be a particular data point, the relative performance measure is then written as:

$$\mathcal{R}(x, y) = \log(\phi_\theta(x, y)) - \log(\hat{\phi}_\theta(x, y)),$$

where $\phi_\theta(\cdot)$ is the density function of the conditional mixture of hybrid Paretos and $\hat{\phi}_\theta(\cdot)$ is the density function of a competing estimator. Positive values indicate that the conditional mixture of hybrid Paretos performed better than the competing estimator.

Increasing sizes of training sets were used; the validation set was fixed as the quarter of the size of the training set and was used for hyper-parameters selection. The remaining data was used for testing and model comparison. The average relative performance on the test set with its standard error in parentheses are given in table 2 for all training set sizes. The hyper-parameters selected on the validation set are given in table 3.

We see in table 2 that the performance of the conditional mixture with Gaussian components and of the multivariate Parzen window estimator is really poor in two instances. This is because these two algorithms are greatly affected by the presence of previously unseen extremes in the test set (target values in the range

Table 2: *Average* **relative** *performance (standard err.) in test with respect to CMM-H for the insurance data set, n being the training set size. Positive values indicate that the CMM-H performed better.*

n	CMM-G	CMM-T	CMM-L
400	93 (43)	0.73 (0.0057)	0.037 (0.018)
800	246 (90)	0.58 (0.005)	0.021 (0.0043)
1600	46 (19)	0.68 (0.0043)	0.0014 (0.0068)
3200	21 (9.8)	0.59 (0.0039)	0.059 (0.011)
6400	72 (30)	0.44 (0.0034)	0.027 (0.013)
n	MM-G	MM-P	
400	0.67 (0.081)	67 (58)	
800	0.69 (0.069)	67 (58)	
1600	0.85 (0.082)	69 (59)	
3200	0.92 (0.082)	698 (594)	
6400	0.96 (0.063)	5.3 (4.5)	

Table 3: *Hyper-parameters selected on the validation set for the insurance data set.*

n	CMM-H (n_h, m)	CMM-G (n_h, m)	CMM-T (n_h, m)
400	(1, 14)	(1, 10)	(1, 14)
800	(1, 10)	(1, 12)	(1, 12)
1600	(1, 16)	(1, 10)	(1, 14)
3200	(1, 12)	(1, 8)	(1, 18)
6400	(1, 18)	(1, 10)	(1, 8)
n	CMM-L (n_h, m)	MM-G m	MM-P (λ_x, λ_y)
400	(1, 14)	4	(100, 1000000)
800	(1, 18)	4	(100, 1000000)
1600	(1, 14)	4	(100, 1000000)
3200	(1, 16)	6	(100, 100000)
6400	(1, 14)	4	(100, 10000000)

of 10^6 whereas the maximum value seen in training is in the range of 10^5). The conditional mixture with hybrid Pareto components is steadily outperforming the competing algorithms although, in some cases, its performance is not significantly better than the conditional mixture with Log-normal components. This could be explained by the fact that the Log-normal tail is a particular case of generalized Pareto tail.

6.3 KDD cup 98 data set

In the last set of experiments, we used the data set provided by the Fourth International Conference on Knowledge Discovery and Data Mining (KDD Cup 98). The dependent variable Y is the amount donated to a national veterans organization. The input variable X has 479 fields describing the donor profile. A binary variable indicates whether or not a person responded to the promotion; the donation amount is only observed when this variable is on. Just like for the insurance data set, a probabilistic classifier could be used to predict, given X, the probability that the person will make a positive donation. However, we addressed only the problem of estimating $p(Y|X, Y > 0)$. We thus have a total of 9716 positive records. We note that 75% of the donations are less or equal to 20$ although some donations go all the way up to 500$. The target variable takes value in a discrete set containing mainly integer numbers; the amounts corresponding to multiple of 5$ are especially frequent as can be seen from the bar plot of Figure 7.

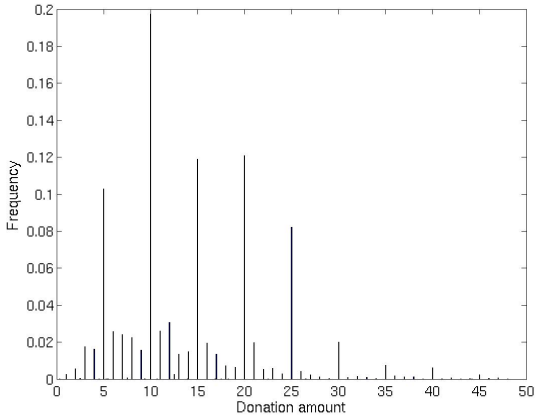

Figure 7: *KDD cup 98 data set: bar plot displaying the frequency of each donation amount smaller than 50$.*

We followed [11] for preprocessing, yielding five input variables. We used the same procedure as for the insurance data set regarding the performance criterion, the training, validation and test sets. The average performance on the test set along with its standard error is given in table 4 for each competing algorithm and each training set size. The selected hyper-parameters are given in table 5.

Table 4: *Average* **relative** *performance (standard err.) in test with respect to CMM-H for the KDD cup 98 data set, n being the training set size. Positive values indicate that the CMM-H performed better.*

n	CMM-G	CMM-T	CMM-L
400	2.3 (0.2)	0.5 (0.021)	1.2 (0.048)
800	1.5 (0.56)	0.37 (0.055)	1.1 (0.056)
1600	0.88 (0.45)	0.26 (0.029)	0.35 (0.043)
3200	0.37 (0.092)	0.31 (0.052)	1 (0.052)
6400	0.015 (0.1)	0.24 (0.08)	0.72 (0.099)
n	MM-G	MM-P	
400	3.1 (0.2)	1515 (2068)	
800	2.4 (0.13)	8.3 (12)	
1600	1.9 (0.14)	3.1 (0.25)	
3200	2.3 (0.098)	3.9 (2.5)	
6400	2.2 (0.11)	1.2 (0.17)	

The results in table 4 show that for this data set as

Table 5: *Hyper-parameters selected on the validation set for the KDD cup 98 data set.*

n	CMM-H (n_h, m)	CMM-G (n_h, m)	CMM-T (n_h, m)
400	(1, 18)	(1, 8)	(1, 4)
800	(1, 18)	(1, 18)	(1, 8)
1600	(1, 4)	(2, 16)	(1, 8)
3200	(1, 4)	(1, 18)	(1, 18)
6400	(1, 16)	(2, 16)	(1, 8)
n	CMM-L (n_h, m)	MM-G m	MM-P (λ_x, λ_y)
400	(1, 16)	16	(100, 0.01)
800	(2, 16)	18	(100, 1)
1600	(1, 18)	20	(1, 100)
3200	(4, 18)	22	(1, 0.01)
6400	(1, 8)	20	(100, 0.01)

well, the conditional mixture with hybrid Pareto components outperforms the other algorithms consistently. However, in this case, the closest competitor is the conditional mixture of Gaussians which gives a performance not significantly distinguishable from the conditional mixture of hybrid Paretos when the training set gets larger. This could also be explained by the fact that the Gaussian tail is well approximated by a generalized Pareto tail of index $\xi = 0$.

7 Conclusion

Fat tailed data are prominent in several commercial applications of statistical machine learning, such as finance and insurance. Research on extreme events has been mainly concerned with unconditional density estimation whereas in many such applications it is required to consider a conditioning variable, which can be very high dimensional. On the other hand, existing tools for representing conditional densities are not always appropriate in the presence of fat tail variations, multi-modal and asymmetric conditional densities. The main contributions of this paper are thus the following.

1. We have introduced a new fat-tailed density, the **hybrid Pareto**, which combines the generalized Pareto with the Gaussian distribution. This density can be used within a mixture model that allows for multi-modal and arbitrarily shaped conditional densities. Such a mixture circumvents the classical problem with the *Peaks over Thresholds* methodology of determining the appropriate threshold above which the samples are considered to be in the tail.

The hybrid Pareto tail includes as particular cases, the tail of the Gaussian, the Log-Normal and the Student t. By using the hybrid Pareto, we avoid making a specific assumption regarding the heaviness of the tail of the underlying distribution. Also, the asymmetric shape of the hybrid might be more suited for the shape of the generative distribution we are looking at.

2. Second, we have proposed a conditional density model based on such a mixture, whose parameters are learned functions of the conditioning variable. These functions can be parametric or nonparametric and we have worked with a simple neural network formulation, which is flexible enough for many applications.

3. Third, we have shown through a series of experiments on artificial and real data sets that the proposed conditional density modelling provides significant advantages over competing algorithms based on mixtures and nonparametric density estimation.

Acknowledgments

The authors thank the following funding organizations: NSERC, MITACS, and the Canada Research Chairs.

References

[1] R. T. Rockafellar and S. Uryasev. Conditional value-at-risk for general loss distributions. *Journal of Banking & Finance*, 26:1443–1471, 2002.

[2] E.F. Fama. The behavior of stock market prices. *Journal of Business*, 38:34–105, 1965.

[3] B. Mandelbrot. The variation of certain speculative prices. *Journal of Business*, 36:394–419, 1963.

[4] A. J. McNeil. Estimating the tails of loss severity distributions using extreme value theory. *Astin Bulletin*, 27:117–137, 1997.

[5] P. Embrechts, C. Kluppelberg., and T. Mikosch. *Modelling Extremal Events.* Applications of Mathematics, Stochastic Modelling and Applied Probability. Springer, 1997.

[6] J Pickands. Statistical inference using extreme order statistics. *Annals of Statistics*, 3:119–131, 1975.

[7] R.M. Corless, G.H. Gonnet, D.E.G. Hare, D.J. Jeffrey, and D.E. Knuth. On the lambert w function. *Advances in Computational Mathematics*, 5:329–359, 1996.

[8] C.E. Priebe. Adaptive mixtures. *Journal of the American Statistical Association*, 89:796–806, 1994.

[9] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford, 1995.

[10] C. Dugas, Y. Bengio, F. Bélisle, C. Nadeau, and R. Garcia. A universal approximator of convex functions applied to option pricing. In *Advances in Neural Information Processing Systems*, volume 13, 2001.

[11] J. Georges and A.H. Milley. Kdd'99 competition: Knowledge discovery contest. *SIGKDD Explorations*, 1(2), January 2000.