Information Theoretic Clustering using Kernel Density Estimation

Shashank Singh
Department of Statistics
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213
sss1@andrew.cmu.edu

Bryan Hooi
Department of Statistics
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213
bhooi@andrew.cmu.edu

1 Introduction

In recent years, information-theoretic clustering algorithms have been proposed which assign data points to clusters so as to maximize the mutual information between cluster labels and data [1, 2]. Using mutual information for clustering has several attractive properties: it is flexible enough to fit complex patterns in the data, and allows for a principled approach to clustering without assuming an explicit probabilistic generative model for the data.

Recently, [3] showed examples in which maximizing mutual information leads to poor performance when the clusters are unbalanced, even in very simple cases. This occurs because algorithms that maximize mutual information are biased toward splitting the data into equal-sized clusters, and this causes the algorithms to perform poorly in the large data limit. They propose to instead minimize the conditional entropy of the cluster labels given the data, where conditional entropy is estimated using a nearest neighbor-based approach. They show that minimizing this objective instead of maximizing mutual information tends to select clusters with large gaps separating them rather than simply selecting approximately equal-sized clusters.

In terms of computation, [2] formulate the mutual information maximization problem as a semidefinite program, and additionally use a low rank inement heuristic to obtain low rank solutions. [3] apply a similar approach to minimizing conditional entropy, involving a relaxation of the discrete cluster variables resulting in a semidefinite program, which is then converted to a candidate solution by rounding with respect to random projections, taking a similar approach to [4].

The **main contributions** of this work are to:

1. further motivate conditional entropy minimization as a clustering objective using the principle of Minimum Description Length (MDL).
2. propose a principle for estimaing the number of clusters based on MDL.
3. propose a novel approach, CHMin, for minimizing the conditional entropy using kernel density estimation and gradient descent.
4. draw a theoretical connection portraying CHMin as a refinement of the K-means algorithm.
5. empirically study the performance of CHMin on some simple synthetic and real datasets.

2 Definitions and Preliminaries

In the following, $x_1, \ldots, x_n \in \mathbb{R}^d$ are the data points and $y_1, \ldots, y_n \in \{1, \ldots, K\}$ are the associated clustering variables. We first define our kernel-based density estimators. We use the usual density estimator for X:

$$\hat{p}(x) := \frac{1}{hn} \sum_{j=1}^{n} K(\frac{x - x_j}{h})$$

The estimator for joint density of (X, Y) (where Y has support $\{1, \ldots, K\}$) is:

$$\hat{p}(x, y) := \frac{1}{hn} \sum_{j=1}^{n} 1\{y_j = y\} K(\frac{x - x_j}{h})$$

Let $C_k = \{i : Y_i = k\}$, the indices of points assigned to cluster k, and define $n_k = |C_k|$.
We estimate the density of Y by its empirical distribution $\hat{p}(y) := \frac{n_y}{n}$.
The conditional density estimator of X given Y is

$$\hat{p}(x|y) := \frac{\hat{p}(x, y)}{\hat{p}(y)}$$

We then define the estimated entropy of X:

$$\hat{H}(X) := -\frac{1}{n} \sum_{i=1}^{n} \log(\hat{p}(x_i))$$

$\hat{H}(Y)$ and $\hat{H}(X|Y)$ are defined analogously (using $\hat{p}(y)$ and $\hat{p}(x|y)$ respectively).
Finally, we define

$$\begin{aligned}
\hat{H}(Y|X) &:= \hat{H}(Y) + \hat{H}(X|Y) - \hat{H}(X) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(y_i) - \frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(x_i|y_i) + \frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(x_i) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \log \left(\frac{\hat{p}(x_i, y_i)}{\hat{p}(x_i)} \right)
\end{aligned}$$

3 Conditional Entropy Minimization and Minimum Description Length

In this section, we first show that conditional entropy minimization can be seen as a special case of model fitting using Minimimum Description Length (MDL). When using MDL, we select between competing models based on which model allows us to compress the data to the smallest number of bits, providing a trade-off between improving model fit and minimizing model complexity[5].

We represent the data using a two-part coding scheme, in which we encode a *point hypothesis* (or probability distribution) $H \in \mathcal{H}$ using $L(H)$ bits, and then encode the data D given hypothesis H, requiring $L(D|H)$ bits. We refer to $\mathcal{H}$, the set of hypotheses under consideration, as the *model*. For example, $\mathcal{H}$ may consist of all mixture distributions of K Gaussians. More generally, the following discussion applies to mixtures of any parametric family of distributions where the number of parameters needed to encode each cluster is fixed (for example, for mixtures of Gaussians, each cluster is encoded by its mean and covariance parameters). Moreover, we use the same parametric family when estimating entropy - that is, instead of using kernel density to estimate empirical distributions, we estimate density within each cluster as a distribution from the same parametric family (e.g. a Gaussian distribution), fitted using maximum likelihood. We also assume that the number of clusters K is fixed.

Given data X, under MDL we select the point hypothesis $H = (Y, \mu)$ which minimizes the total description length $L(H) + L(X|H)$, where Y contains the cluster assignments and $\mu = (\mu_1, \ldots, \mu_K)$ contains the cluster parameters, e.g. cluster centers in a Gaussian mixture model.

Theorem 1. *Under the above conditions, minimizing description length $L(H) + L(X|H)$ is equivalent to minimizing estimated conditional entropy $H(Y|X)$.*

Proof. By assumption, encoding the cluster parameters μ requires a fixed number of parameters, so we can ignore this cost. It remains to encode the cluster assignments Y, requiring $L(Y)$ bits, and to encode the data given ths model, requiring $L(X|Y, \mu)$ bits.

We first compute $L(X|Y, \mu)$. By Shannon's source coding theorem, the optimal expected code length is achieved by assigning a code length of $-\log(p)$ to an outcome with probability p (and an analogous fact holds for continuous distributions with p as the density, by discretizing the space of outcomes to small intervals). As such, given hypothesis $H = (Y, \mu)$, the optimal code requires $\hat{p}(x_i|y_i)$ bits to represent x_i. This results in a code length of

$$\begin{aligned} L(X|Y, \mu) &= -\sum_{i=1}^{n} \log \hat{p}(x_i|y_i) \\ &= \hat{H}(X|Y) \end{aligned}$$

Next, we compute $L(Y)$. To encode the cluster assignments Y, we encode each y_i with respect to the empirical distribution of Y, which by Shannon's source coding theorem results in a code length for y_i of $-\log \hat{p}(y_i)$, and summing over i, the code length is

$$-\sum_{i=1}^{n} \log \hat{p}(y_i) = \hat{H}(Y)$$

Note that a recipient of a message with Y encoded in this way can infer which y_i are in the same cluster based on which y_i are encoded using the same code word, but they cannot infer how these code words correspond to the actual clusters parametrized by $\mu_1, \ldots, \mu_k$. As such, we additionally need to send an additional $\log(K!)$ bits to encode which of the $K!$ possible permutations of the code words correspond to $\mu_1, \ldots, \mu_K$ respectively. However, since K is fixed, this is a constant and can be ignored.

Combining the above results, we find that the overall description length is:

$$L(H) + L(X|H) = \hat{H}(Y) + \hat{H}(X|Y) + \text{const}$$

□

One benefit of viewing conditional entropy minimization in the MDL framework is that we can select the number of clusters K in a principled manner based on MDL. We will show that selecting the optimal model in this way amounts to minimizing conditional entropy plus a regularization-like term that penalizes the number of clusters K.

In the MDL approach to model selection, we define models $\mathcal{H}_1, \mathcal{H}_2, \ldots$, each of which is a set of point hypotheses. We select the best model by encoding the data with respect to each model and selecting the model that gives the best compression. In our case $\mathcal{H}_K$ is the set of partitions into K clusters. As before, the description length for model $\mathcal{H}_K$ is given by selecting point hypothesis $H \in \mathcal{H}_K$ to minimize $L(H) + L(X|H)$. However, the number of clusters K is no longer fixed.

Theorem 2. *To select the number of clusters K by MDL, we minimize*

$$\hat{H}(Y|X) + \log^*(K) + Kd(\log(2B) + \frac{1}{2}\log(n)) + \log(K!)$$

Proof. Encoding the data requires encoding the number of clusters K, the cluster parameters μ, the cluster assignments Y, and the data X given Y and μ.

To encode K, we use Rissanen's "universal code for the natural numbers," which was shown by Rissanen to achieve close to the shortest possible code length for large natural numbers [6]. Define the $\log^*$ function as $\log^*(n) = \log_2(n) + \log_2 \log_2(n) + \ldots$, where only the positive terms are to be included in the sum. Then this code length of K is

$$L(K) = \log^*(K) + \log_2(K_0)$$

where K_0 is a constant (equal to $\sum_{n>1} 2^{-\log^*(n)}$).

Next, we compute $L(\mu)$. Since the μ_i are continuous, we need to discretize them before encoding them: following [6], we use a precision of $1/\sqrt{n}$, the asymptotic error of estimating each parameter: this choice was shown to be optimal for regular parametric families in [6]. Assuming that each parameter μ_i has dimension d and that the cluster centers are in a bounded range $|\mu_i| \leq B$, the parameters consist of Kd numbers, each taking one of $2B\sqrt{n}$ possible values after discretization, so the code length is:

$$\begin{aligned} L(\mu) &= Kd \log(2B\sqrt{n}) \\ &= Kd(\log(2B) + \frac{1}{2}\log(n)) \end{aligned}$$

Combining this with the previous theorem, the overall description length is:

$$\begin{aligned} L(X|H) + L(H) &= \hat{H}(Y) + \hat{H}(X|Y) + \log^*(K) + Kd(\log(2B) + \frac{1}{2}\log(n)) + \log(K!) + \text{const} \\ &= \hat{H}(Y|X) + \log^*(K) + Kd(\log(2B) + \frac{1}{2}\log(n)) + \log(K!) + \text{const} \end{aligned}$$

In conclusion, we minimize $\hat{H}(Y|X)$ plus a regularization term, $\log^*(K) + Kd(\log(2B) + \frac{1}{2}\log(n)) + \log(K!)$, which acts as a penalty for larger values of K. By Stirling's formula $\log(K!) = O(K \log K)$, so for $K < n$, the total penalty grows as $O(Kd \log n)$, which is asymptotically the same as the penalty in Bayesian Information Criterion (BIC). □

4 Conditional Entropy and the K-Means Algorithm

In this section, we draw a connection between minimizing conditional entropy and the K-means algorithm. To do this, we construct an upper bound for $\hat{H}(X|Y)$. We then show that when a Gaussian kernel is used, this upper bound becomes the usual K-means objective, and hence the K-means algorithm can be seen as a way to minimize an upper bound on the estimated conditional entropy $\hat{H}(X|Y)$. This will allow us to obtain an upper bound on $\hat{H}(Y|X)$ as well.

Theorem 3. *When using a Gaussian kernel function $K(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$, the estimated conditional entropy $\hat{H}(X|Y)$ satisfies:*

$$\hat{H}(X|Y) \leq \log(h) + \frac{1}{2}\log(2\pi) + \frac{1}{2h^2 n} \sum_{k=1}^{K} \sum_{i \in C_k} (x_i - \mu_k)^2$$

Consequently, minimizing the K-means objective $\sum_{k=1}^{K} \sum_{i \in C_k} (x_i - \mu_k)^2$ is equivalent to minimizing an upper bound for $\hat{H}(X|Y)$.

Proof. We estimate $\hat{H}(X|Y)$ as:

$$\hat{H}(X|Y) = -\frac{1}{n} \sum_{i=1}^{n} \log \hat{p}(x_i|y_i)$$

Estimating $\hat{p}(x_i|y_i)$ using kernel density estimation on the points assigned to the same cluster as y_i:

$$\begin{aligned} \hat{H}(X|Y) &= -\frac{1}{n} \sum_{i=1}^{n} \log\left(\frac{1}{h n_{y_i}} \sum_{j=1}^{n} 1\{y_i = y_j\} K\left(\frac{x_i - x_j}{h}\right)\right) \\ &= \log(h) - \frac{1}{n} \sum_{i=1}^{n} \log\left(\frac{1}{n_{y_i}} \sum_{j=1}^{n} 1\{y_i = y_j\} K\left(\frac{x_i - x_j}{h}\right)\right) \end{aligned}$$

The argument of $\log$ is an average over terms of which n_{y_i} terms are nonzero. Hence, by Jensen's inequality on the convex function $f(x) = -\log(x)$ we interchange the log and the averaging:

$$\hat{H}(X|Y) \leq \log(h) - \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_{y_i}}\sum_{j=1}^{n}1\{y_i = y_j\}\log(K(\frac{x_i - x_j}{h}))$$

Now using the Gaussian kernel $K(x) = \frac{1}{\sqrt{2\pi}}e^{\frac{-x^2}{2}}$:

$$\begin{aligned}\hat{H}(X|Y) &\leq \log(h) - \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_{y_i}}\sum_{j=1}^{n}1\{y_i = y_j\}(-\frac{1}{2}\log(2\pi) - \frac{1}{2}(\frac{x_i - x_j}{h})^2) \\ &= \log(h) + \frac{1}{2}\log(2\pi) + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n_{y_i}}\sum_{j=1}^{n}1\{y_i = y_j\}(\frac{1}{2}(\frac{x_i - x_j}{h})^2) \\ &= \log(h) + \frac{1}{2}\log(2\pi) + \frac{1}{2h^2n}\sum_{i=1}^{n}\frac{1}{n_{y_i}}\sum_{j=1}^{n}1\{y_i = y_j\}(x_i - x_j)^2\end{aligned}$$

The double summation sums $(x_i - x_j)^2$ for exactly the ordered pairs (i, j) where i and j are assigned to the same cluster. Rearranging to sum over clusters:

$$\hat{H}(X|Y) \leq \log(h) + \frac{1}{2}\log(2\pi) + \frac{1}{2h^2n}\sum_{k=1}^{K}\frac{1}{n_k}\sum_{i,j\in C_k}(x_i - x_j)^2 \tag{1}$$

To compare this to the K-means objective, let μ_k be the centroid of cluster k, then:

$$\begin{aligned}\sum_{i,j\in C_k}(x_i - x_j)^2 &= \sum_{i\in C_k}\sum_{j\in C_k}(x_i - \mu_k + \mu_k - x_j)^2 \\ &= \sum_{i\in C_k}\sum_{j\in C_k}((x_i - \mu_k)^2 + (x_j - \mu_k)^2 + 0)\end{aligned}$$

in the above the cross terms sum to 0 when summing over j for any fixed i. Thus this becomes:

$$\sum_{i,j\in C_k}(x_i - x_j)^2 = 2n_k\sum_{i\in C_k}(x_i - \mu_k)^2$$

substituting this into (1):

$$\hat{H}(X|Y) \leq \log(h) + \frac{1}{2}\log(2\pi) + \frac{1}{h^2n}\sum_{k=1}^{K}\sum_{i\in C_k}(x_i - \mu_k)^2$$

We thus see that minimizing the K-means objective $\sum_{k=1}^{K}\sum_{i\in C_k}(x_i - \mu_k)^2$ is equivalent to minimizing an upper bound on the conditional entropy estimator $\hat{H}(X|Y)$. □

An intuitive interpretation of this result comes by considering the equality conditions of the Jensen's inequality step: since $f(x) = -\log(x)$ is strictly convex, equality holds in the Jensen's inequality step iff $(x_i - x_j)^2$ is equal for all i, j in the same cluster. In other words, roughly speaking, when the pairwise distances between points in the same cluster are around equal, minimizing the K-means objective and $\hat{H}(X|Y)$ become roughly equivalent (however, this should be seen as an approximation since the cluster assignments are not fixed). As the within-cluster pairwise distances become more uneven, Jensen's inequality becomes a more strict inequality, so the K-means objective grows faster in relation to conditional entropy, suggesting that the K-means algorithm differs from conditional entropy minimization in that K-means prefers tighter clusters in which the pairwise distances are more evenly distributed.

We now consider minimizing the conditional entropy $\hat{H}(Y|X)$.

Corollary 1. *Minimizing $\hat{H}(Y|X)$ is equivalent to minimizing*

$$\hat{H}(Y) + \frac{1}{h^2n}\sum_{k=1}^{K}\sum_{i\in C_k}(x_i - \mu_k)^2 = -\sum_{k=1}^{K}\frac{n_k}{n}\log\frac{n_k}{n} + \frac{1}{h^2n}\sum_{k=1}^{K}\sum_{i\in C_k}(x_i - \mu_k)^2$$

Proof. By definition:

$$\hat{H}(Y|X) = \hat{H}(Y) + \hat{H}(X|Y) - \hat{H}(X)$$

$\hat{H}(X)$ depends only on the data X and can be ignored. The result then follows from Theorem 3 in addition to writing $\hat{H}(Y) = -\sum_{k=1}^{K} \frac{n_k}{n} \log \frac{n_k}{n}$.

□

5 Gradient Descent for Conditional Entropy Minimization

Here, were derive a simple procedure for minimizing $\hat{H}(Y|X)/\hat{H}(Y)$ by gradient descent. For now, we relax the requirement that each label $y_i \in [K]$, allowing $y_i \in \mathbb{R}^K$. We describe in Section 6.1 how to transform y_i back into a clusters label in $[K]$. Since

$$H(Y|X) = -\mathbb{E}_{X,Y}\left[\log \frac{p_{Y,X}(y,x)}{p_X(x)}\right] \quad \text{and} \quad \hat{H}(Y) = -\mathbb{E}_Y\left[\log p_Y(y)\right],$$

given estimators $\hat{p}_X$ for p_X, $\hat{p}_Y$ for p_Y, and $\hat{p}_{XY}$ for p_{XY}, we estimate

$$\hat{H}(Y|X) = -\frac{1}{n}\sum_{i=1}^{n} \log\left(\frac{\hat{p}_{X,Y}(x_i,y_i)}{\hat{p}_X(x_i)}\right) \quad \text{and} \quad \hat{H}(Y) = -\frac{1}{n}\sum_{i=1}^{n} \log\left(\hat{p}_Y(y_i)\right).$$

Using a kernel density estimates with bandwidth h and a symmetric kernel K_h, $\forall k \in [n]$,

$$\begin{aligned}\frac{d}{dy_i}\hat{H}(Y|X) &= -\frac{1}{n}\sum_{i=1}^{n} \frac{d}{dy_i}\log\left(\frac{\hat{p}_{X,Y}(x_i,y_i)}{\hat{p}_X(x_i)}\right) \\ &= -\frac{1}{n}\sum_{i=1}^{n} \frac{d}{dy_i}\log\left(\hat{p}_{X,Y}(x_i,y_i)\right) = -\frac{1}{n}\sum_{i=1}^{n} \frac{\frac{d}{dy_i}\hat{p}_{X,Y}(x_i,y_i)}{\hat{p}_{X,Y}(x_i,y_i)},\end{aligned}$$

and

$$\frac{d}{dy_i}\hat{H}(Y) = -\frac{1}{n}\sum_{i=1}^{n} \frac{d}{dy_i}\log\left(\hat{p}_Y(y_i)\right) = -\frac{1}{n}\sum_{i=1}^{n} \frac{\sum_{j=1}^{n}\frac{d}{dy_i}K_h(y_i,y_j)}{\sum_{j=1}^{n}K_h(y_i,y_j)}.$$

Hence,

$$\frac{d}{dy_k}\hat{H}(Y|X) = -\frac{1}{n}\sum_{i=1}^{n} \frac{\sum_{j=1}^{n}K_h(x_i,x_j)\frac{d}{dy_k}K_h(y_i,y_j)}{\sum_{j=1}^{n}K_h(x_i,x_j)K_h(y_i,y_j)}. \tag{2}$$

Finally, using the quotient rule, we can compute the gradient as

$$\nabla_y \frac{\hat{H}(Y|X)}{\hat{H}(Y)} = \frac{\hat{H}(Y)\nabla_y\hat{H}(Y|X) - \hat{H}(Y|X)\nabla_y\hat{H}(Y)}{(H(Y))^2}.$$

It should be noted that the majority of summands on the right hand side of Equation (2) are 0 (because they do not contain y_k). Hence, the gradient $\nabla_y\hat{H}(Y|X)$ can be computed using only $O(n^2)$ (rather than $O(n^3)$) estimates of the kernel and its derivative. This can be reduced to $O(n\log n)$ using a kernel of bounded support and a k-d tree representation of the kernel density estimate.

6 Empirical Results

We implemented the gradient descent algorithm described above in MATLAB and tested it on 6 datasets, alongside 5 other clustering algorithms. For all experimental results, it was assumed that the number of clusters was known beforehand.

6.1 Clustering Algorithms

CHMin: Our conditional entropy minimization algorithm is based on the gradient descent steps derived in Section 5. To ensure the values $y_i \in \mathbb{R}^K$ are interpretable as cluster labels, after each iteration of gradient descent, we rescale and project each y_i onto the convex hull of the canonical basis vectors $(0, 0, \ldots, 0, 1), (0, 0, \ldots, 1, 0), \ldots, (1, 0, \ldots, 0, 0)$. In practice, each y_i converges to one of the K vertices of this $(K-1)$-simplex; we cluster each x_i by the vertex to which y_i is nearest.

Other Algorithms: K-means++ and 3 variants of Hierarchical Clustering (HC), with single, complete, and average linkage.

Parameter Selection: The clustering algorithm has 4 free parameters: the kernel and bandwidth for kernel density estimation, and the step size and initialization for the gradient descent.

Kernel: For the labels, we used a Gaussian kernel, which pushed labels towards vertices of the $(K-1)$-simplex to which they were constrained, [1] giving easy interpretation to estimated y values.

For the inputs x, we also generally used a Gaussian kernel which seems to work best for noisy data. However, we found that, when clusters were sufficiently well separated, using bounded kernels (e.g., Epanechnikov, Uniform, etc.) seems to result in very fast convergence.

Bandwidth: We tried several approaches for automatically selecting bandwidth. Specifically, we tried the LCV, HALL, ROT, and LOCAL criteria available in the MATLAB Kernel Density Estimation Toolbox. [2] For our experiments, we used ROT (rule of thumb), a normalization based on the data's covariance matrix, which is very fast and seemed to consistently perform well.

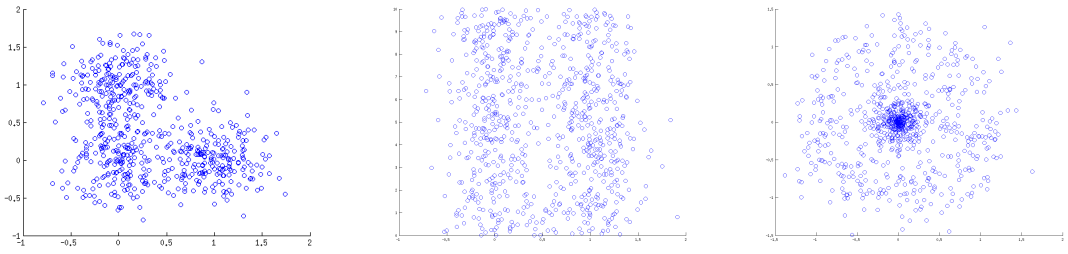
Step Size: We found that most sequences of step sizes decreasing sufficiently slowly to 0 gave comparable results. In our experiments, we used a step size of $\alpha_i = \sqrt{i}$ at iteration i.

Initialization: Theorem 1 suggests using K-means to initialize our labels. Unsurprisingly, in practice, this work well for datasets such as *Iris* where K-means itself is able to coarsely pick out the clusters. In other cases, such as *Bars* or *Circles*, the K-means labeling appears to be a sadde-point and causes gradient descent to converge very slowly. Thus, for experiments, we first initialized with K-means, and then restarted randomly if gradient descent failed to converge within 200 iterations.

6.2 Datasets

We tested CHMin on 3 synthetic datasets and 3 were real datasets from the UCI repository.

6.2.1 Synthetic Datasets



(a) (b) (c)

Figure 1: Realizations of the synthetic datasets (a) *Gaussians*, (b) *Bars*, and (c) *Circles*.

Gaussians: 100 samples from each of 3 Gaussians in $\mathbb{R}^3$, with covariance $0.25I$ and means $(0, 0, 1)$, $(0, 1, 0)$, and $(1, 0, 0)$. All algorithms should perform well given rather distinct, spherical clusters.

[1] Any kernel with unbounded support decreasing monotonically away from 0 should work for this purpose.
[2] Details are available here: `http://www.ics.uci.edu/~ihler/code/kde.html`

Dataset (# clusters)	CHMin	K-means++	HC (single)	HC (complete)	HC (average)
Gaussians (3)	0.991 ± 0.008	**0.998**	0.767	0.984	0.994
Bars (2)	$\mathbf{0.723} \pm 0.013$	0.520	0.502	0.524	0.524
Circles (2)	$\mathbf{0.894} \pm 0.019$	0.671	0.501	0.677	0.605
Iris (3)	$\mathbf{0.929} \pm 0.031$	0.893	0.680	0.840	0.906
Wine (3)	0.675 ± 0.0283	**0.702**	0.427	0.674	0.612
Wine5 (3)	$\mathbf{0.704} \pm 0.044$	0.494	0.387	0.500	0.500

Table 1: Mean ($\pm$ standard deviation) of accuracy, for each algorithm on each *dataset* (# of clusters). Hierarchical clustering algorithms are denoted HC (type), where type is the linkage procedure used. Means and standard deviations were computed from 100 trials, with synthetic datasets resampled in each trial. Boldface numbers indicate the best performing algorithm for each dataset.

Bars: 400 noisy samples (x_i, y_i) from each of two long, vertical lines, where each $y_i \sim \text{Unif}(0, 10)$, $x_1, \ldots, x_{200} \sim \mathcal{N}(0, 0.3)$, and $x_{201}, \ldots, x_{400} \sim \mathcal{N}(1, 0.3)$. We expect K-means to perform poorly here because the variance in y is much greater than in x, while only x depends on the cluster.

Circles: 600 noisy samples $(x_i, y_i) = (r_i \cos\theta_i, r_i \sin\theta_i)$ from two concentric circles, where $\theta_i \sim \text{Unif}(0, 2\pi)$, $r_1, \ldots, r_{400} \sim \mathcal{N}(1, 0.25)$, and $r_{401}, \ldots, r_{600} \sim \mathcal{N}(2, 0.25)$. Since the clusters are not linearly separable, we expect K-means to perform poorly. In order to show that, unlike clustering via mutual information maximization, CHMin does not force clusters to be of equals size, we drew $2/3$ of our samples from the cluster of smaller radius.

6.2.2 Real Datasets

Iris: 50 samples of flower measurements from each of 3 species of irises.

Wine: 178 samples of each of 13 chemical properties from wines from 3 Italian wine cultivars. Due to the large number of features relative to the sample size (for a nonparametric method), we expect the kernel density estimation step may cause CHMin may perform poorly.

Wine5: A subset of *Wine* using the (arbitrarily chosen) first 5 features and all 178 data points. This subset is used to study how the performance of CHMin scales with the dimension of the data.

6.3 Results

Of the five algorithms tested, CHMin performed best on 4 of 6 datasets. On the remaining two datasets (*Gaussians* and *Wine*), K-means++ performed best. Since *Gaussians* is precisely the scenario for which K-means is designed, this is unsurprising. We suspect the poor behavior of CHMin on *Wine* is due to the poor dependence of kernel density estimation on dimension (the curse of dimensionality). This is supported by the fact that the performance of CHMin improved when the number of features was reduced to 5, while performance of other algorithms fell sharply. For both *Gaussians* and *Wine*, CHMin still performed competitively (within 3% of K-means).

7 Conclusions

In this work, we motivated estimated conditional entropy minimization as a clustering objective using MDL, proposed a principle for estimaing the number of clusters based on MDL, proposed CHMin for minimizing the conditional entropy objective via kernel density estimation and gradient descent, related CHMin to the K-means algorithm, and tested CHMin on real and synthetic data.

Avenues for empirical future work include testing CHMin on larger datasets, combining CHMin with dimension reduction techniques for high-dimensional datasets, testing how well our MDL criterion (Theorem 3) can estimate the number of clusters, and comparing CHMin to more modern nonparametric clustering approaches such as mutual information maximization, the mean shift algorithm, the k-nearest neighbor approach of [3], and spectral clustering. Theoretical extensions might include weakening the assumptions of Theorem 2 to cover the nonparametric case and trying to adapt error bounds for entropy estimation (see, e.g., [7]) to study the performance of CHMin.

References

[1] Lev Faivishevsky and Jacob Goldberger. Nonparametric information theoretic clustering algorithm. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 351–358, 2010.

[2] Meihong Wang and Fei Sha. Information theoretical clustering via semidefinite programming. In *International Conference on Artificial Intelligence and Statistics*, pages 761–769, 2011.

[3] Greg Ver Steeg, Aram Galstyan, Fei Sha, and Simon DeDeo. Demystifying information-theoretic clustering. *arXiv preprint arXiv:1310.4210*, 2013.

[4] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[5] Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

[6] Jorma Rissanen. A universal prior for integers and estimation by minimum description length. *The Annals of statistics*, pages 416–431, 1983.

[7] S. Singh and B. Poczos. Exponential concentration of a density functional estimator. In *Neural Information Processing Systems (NIPS)*, 2014.