### Prototypal Analysis and Prototypal Regression

<table>
<thead>
<tr>
<th><strong>Journal:</strong></th>
<th>Journal of the American Statistical Association</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Manuscript ID:</strong></td>
<td>JASA-T&amp;M-2018-0235</td>
</tr>
<tr>
<td><strong>Manuscript Type:</strong></td>
<td>Article – Theory &amp; Methods</td>
</tr>
<tr>
<td><strong>Keywords:</strong></td>
<td>Archetypal Analysis, Distribution Regression, Reproducing Kernel Hilbert Space, Kernel Embedding</td>
</tr>
</tbody>
</table>
Prototypal Analysis and Prototypal Regression

Abstract

Prototypal analysis is introduced to overcome two shortcomings of archetypal analysis: its sensitivity to outliers and its non-locality, which reduces its applicability as a learning tool. Same as archetypal analysis, prototypal analysis finds prototypes through convex combination of the data points and approximates the data through convex combination of the archetypes, but it adds a penalty for using prototypes distant from the data points for their reconstruction. Prototypal analysis can be extended—via kernel embedding—to probability distributions, since the convexity of the prototypes makes them interpretable as mixtures. Finally, prototypal regression is developed, a robust supervised procedure which allows the use of distributions as either features or labels.

Keywords: Archetypal Analysis, Distribution Regression, Reproducing Kernel Hilbert Space, Kernel Embedding

1. INTRODUCTION

Archetypal analysis, an unsupervised learning method introduced by Cutler and Breiman (1994), approximates a set of data points by convex combinations of archetypes, which are themselves convex combinations of the original data. At the cost of introducing convexity constraints into the optimization, archetypal analysis achieves interpretability, as a convex combination can be thought of as a weighted sum of its components—not so a general linear combination, where components can be subtracted as well as added. This extra computational cost can be handled efficiently, as several studies have shown (Bauckhage and Thurau, 2009; Mørup and Hansen, 2012; Chen et al., 2014).
Archetypal analysis has been applied in physics (Stone and Cutler, 1996; Stone, 2002; Chan et al., 2003), biology (Huggins et al., 2007; Römer et al., 2012; Thøgersen et al., 2013), psychology (Thurau and Drachen, 2011; Drachen et al., 2012, 2016; Sifa and Bauckhage, 2013), marketing (Li et al., 2003; DEsposito et al., 2006), performance analysis (Porzio et al., 2006, 2008; Eugster, 2012; Seiler and Wohlrabe, 2013) and computer vision (Marinetti et al., 2006; Thurau and Bauckhage, 2009; Cheema et al., 2011; Asbach et al., 2013; Xiong et al., 2013).

Despite the many positive features of archetypal analysis, one can point out two significant drawbacks. One is its sensitivity to outliers; since the data is approximated by its projection on the convex hull of the archetypes, adding a point outside of the boundary of the data impacts the archetypes to a large degree. Another drawback of the methodology is its non-locality: data points are approximated as convex combinations of archetypes that may be very far away. For many learning tools, such as regression, such representation is of little use.

This paper introduces prototypical analysis as a robust alternative to archetypal analysis without these drawbacks. Prototypical analysis preserves interpretability, as it finds prototypes via convex combinations of the data and reconstructs the data as convex combinations of the prototypes. The difference between archetypal and prototypical analysis is that the former allows arbitrary convex combination of archetypes for representing the data, while the later penalizes the use of prototypes far away from a data point to represent it. Technically, this is achieved by adding a $L_1$ penalty term on the reconstructing coefficients for each point, with weights that depend on the distance between the point and the prototype under consideration. As a consequence, a point far away from the majority of the data would contribute little to the reconstruction and will not be chosen as a prototype.

The locality of the reconstruction by prototypes makes them useful for key learning tasks such as regression. Given training data on predictors and responses, regression concerns inferring the response for new instances of the predictors. We introduce prototypical regression as a new regression method with the advantage of interpretability and robustness. Prototypical regression uses convex combinations to extract prototypes from both the predictors and the response. The regression relationship is built with pairs of one prototype from the predictor and one prototype from the response, i.e. prototypical regression maps each prototype from the predictor to one prototype
from the response and extends to all values of the predictors via local convex combinations. Here convexity is the source of interpretability and, combined with locality, of robustness, as an outlier will only affect the predictions in its immediate neighborhood.

Kernel methods and reproducing kernel Hilbert space (RKHS) are widely used in machine learning to extend algorithms where only inner products among data points are required (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004; Hofmann et al., 2008). This is the case of archetypal analysis, which can therefore be extended via kernels (Mørup and Hansen, 2012). Examples of application can be found in time series clustering (Bauckhage and Manshaei, 2014), behavior analysis (Sifa et al., 2014) and image processing (Zhao et al., 2015; Zhao and Zhao, 2016). Prototypal analysis and prototypical regression can be kernelized as well, enabling in particular the use of probability distributions as either features or outputs, in lieu of the more conventional discrete or real-valued scalars and vectors. This extension is particularly well suited for archetypal and prototypical analysis, as their underlying convex combinations correspond to mixtures of distributions. We adopt kernel embedding (also known as kernel mean embedding) to extend archetypal analysis, prototypical analysis and prototypical regression to handle distributional data. Kernel embedding maps probability distributions or their samples into a RKHS. Using the inner products of the RKHS, one can find archetypes and prototypes of distributions and also perform regression in this infinite dimensional setting. More generally, kernel embedding enables prototypical regression to deal with a blend of categorical, numerical and distributional data.

In prior work, Muandet et al. (2012) extends support vector machine to support measure machine for classification of distributions using the kernel embedding induced inner product. Szabó et al. (2015, 2016) performs a similar extension for kernel ridge regression. Póczos et al. (2013) regresses numbers from distributions through a kernel-kernel estimator, which involves one kernel for density estimation and another for kernel smoothing, using the distance between the distributions to weight the response variables. Oliva et al. (2013) introduces a distribution to distribution regression model via orthogonal series density estimation on the response distributions and kernel density estimation on the predictor distributions and the new input.

The rest of this paper is organized as follows: Section 2 briefly reviews archetypal analysis and empirically shows that it is not robust to outliers and that, as it concentrates on the boundary of the
data, it does not resolve the underlying space well. Section 3 introduces prototypical analysis as a robust unsupervised method to find prototypes and build data-driven barycentric coordinates system without these two drawbacks. Section 4 introduces simple and multiple prototypical regression—the latter applicable to features of different nature that cannot naturally be regarded as components of a vector. Section 5 extends archetypal and prototypical analysis and prototypical regression via kernels and applies it to the analysis of distributional data.

2. ARCHETYPAL ANALYSIS

Archetypal analysis approximates data points by convex combination of “archetypes”, which are themselves convex combinations of the data points (see Cutler and Breiman, 1994). Given a data set \( \{x_i\}_{i=1}^n \), one seeks archetypes of the form

\[
\begin{align*}
\mathbb{R}^n & \quad b_j x_i, \quad b_j \geq 0, \quad j \in [1, k] \\
\mathbb{R}^n & \quad b_j = 1, \quad j \in [1, k]
\end{align*}
\]

and approximates each data point through

\[
\begin{align*}
x_i & \approx \mathbb{R}^n \quad \sum_{j=1}^k a_{j} b_j x_i, \\
a_{j} & \geq 0, \quad a_{j} = 1, \quad j \in [1, n],
\end{align*}
\]

by solving the following optimization problem:

\[
\begin{align*}
\min \quad & \mathbb{R}^n \quad \sum_{j=1}^k a_{j} b_j x_i - a_{j} b_j x_i \\
\text{subject to} \quad & \sum_{j=1}^k b_j = 1, \quad \sum_{i=1}^n b_j = 1, \quad a_{j} \geq 0, \quad a_{j} = 1
\end{align*}
\]

As archetypal analysis minimizes the distance between the data and the convex hull of the archetypes, it tends to choose as archetypes extreme points among the data in order to enlarge this convex hull. In particular, when the data includes outliers, these are typically chosen as archetypes, as illustrated in Figure 1. As the number \( k \) of archetypes grows, they sit on the boundary of the convex hull of the data, not resolving its interior, as shown in Figure 2. Also, when \( k \) is sufficiently large (typically when \( k > d \), where \( d \) is the number of the vertices of the convex hull spanned by \( \{x_i\}_{i=1}^n \)), the \( a_{j} \) are not uniquely defined.
Algorithm 1 Archetypal Analysis

Input: Data \(\{x_i\}_{i=1}^n\), \(k\): number of archetypes.
Output: Archetypes \(\{u_j\}_{j=1}^k\) and approximation \(\{\hat{x}_i\}_{i=1}^n\) to data by their convex combination.

1. \((a_{ij}, b_j) \leftarrow \arg\min_{a_{ij} \geq 0, b_j \geq 0} \sum_{i=1}^n a_{ij} b_j x_i\) subject to \(a_1 + \cdots + a_n = 1\) and \(b_1 + \cdots + b_n = 1\)
2. for \(j = 1, \ldots, k\) do
3. \(u_j \leftarrow b_j x_1 + \cdots + b_n x_n\)
4. end for
5. for \(i = 1, \ldots, n\) do
6. \(\hat{x}_i \leftarrow a_{i1} u_1 + \cdots + a_{ik} u_k\)
7. end for
8. return \(\{u_j\}_{j=1}^k, \{\hat{x}_i\}_{i=1}^n\)

3. PROTOTYPAL ANALYSIS

Like archetypal analysis, prototypal analysis finds prototypes \(\{u_j\}_{j=1}^k\) as convex combinations of the data points \(\{x_i\}_{i=1}^n\), and approximates the latter as convex combinations of the former, as in Equation 1 and 2. The difference lies in that, when reconstructing each data point, prototypal analysis is biased toward using prototypes near that point. To this end, it adds a penalty term on the distance between points and prototypes, replacing the objective function in Equation 3 by

\[
\min_{a_{ij} \geq 0, b_j \geq 0} \sum_{i=1}^n a_{ij} b_j x_i - \sum_{i=1}^n a_{ij} x_i - b_j x_i + \lambda \sum_{i=1}^n a_{ij} x_i - b_j x_i, \tag{4}
\]

where \(\lambda \geq 0\) is a tuning parameter. In the penalty term, \(a_{ij}\), the weight of the \(j\)-th archetype in the reconstruction of \(x_i\), is multiplied by \(\|x_i - \sum_{j=1}^n b_j x_j\|^2\), the square of distance between data point \(x_i\) and the \(j\)-th prototype \(u_j = \sum_{i=1}^n b_j x_i\). Hence the closer \(x_i\) is to the \(j\)-th prototype, the more weight this prototype will be assigned in the reconstruction. Compared with archetypal analysis, which tends to use extreme points as archetypes, prototypal analysis has prototypes that resemble the original data. Hence it is less sensitive to outliers. Figure 3 shows the prototypes corresponding to the same data of Figure 1. In this case, adding one outlier does not change the archetypes significantly. In the computational procedure we use to minimize Equation 4, we alternate between minimizing over the \(a\) and \(b\), which is also the procedure of choice in archetypal
Figure 1: Archetypal analysis on two dimensional data with 4 archetypes. Data of the right figure contains one more outlier than the left figure. The archetypes are visualized using the '+' sign. Adding one outlier fundamentally changes the location of the archetypes. In addition, the reconstruction of many data-points in terms of the archetypes is not unique.

analysis (Cutler and Breiman, 1994).

Prototypical analysis can be viewed as a mixture of archetypal analysis and k-means clustering. When \( \lambda \) goes to infinity, only the penalty term remains in prototypical analysis, and the problem reduces to

\[
\min_{a_{ij} \geq 0, b_{ij} \geq 0} \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{i=1}^{n} a_{ij} x_{il} - b_{ij} x_{l},
\]

which is equivalent to K-means clustering, with the prototypes \( u_l = \sum_{i=1}^{n} b_{il} x_{i} \) playing the role of barycenters. To see this equivalence, notice two facts about the solution to Equation 5:

1. For each observation \( x_i \), the only nonzero \( a_{ij} \) corresponds to the closest \( u_l \), for which \( a_{ij} = 1 \).

2. For each prototype \( u_l \), the only nonzero \( b_{ij} \) corresponds to those \( i \) such that \( u_l \) is the closest prototype to \( x_i \). Moreover, these \( b_{ij} \) all have the same value, as the barycenter of a set of points is the minimizer of the sum of the square distances to them.
Figure 2: Archetypal analysis on two dimensional data with 3, 4, 5, 6, 7 and 8 archetypes. The archetypes are visualized using the '+' sign. As the number of archetypes grows, they cover just the perimeter of the convex hull of the data.

4. PROTOTYPAL REGRESSION

Given a set of predictor-response pairs \((x_i, y_i)\), regression is the task estimating the response \(y_0\) corresponding to a new value \(x_0\) of the predictor. Performing prototypical analysis on the \(\{x_i\}\) yields the prototypes \(\{u_j\}\) and a rule that approximates \(x_0\) as a convex combination of a local subset of the \(\{u_j\}\). Hence introducing prototypes \(\{v_j\}\) in \(y\)-space that approximate the images of the \(\{u_j\}\), one can estimate \(y_0\) as the corresponding convex combination of the \(\{v_j\}\).

4.1 Simple Prototypical Regression

Simple prototypical regression estimates the response \(y\) from a single predictor \(x\), where both predictor and response can be vectorial, using prototypes of both \(x\) and \(y\). The prototypes of \(x\) come directly from prototypical analysis, i.e. solving Equation 4, while the choice of prototypes of \(y\) takes the regression into account. Denoting by \(u_j\) the prototypes of \(x\) and by \(v_j\) the prototypes of \(y\), the
Algorithm 2 Prototypical Analysis

Input: Data \( \{x_i\}_{i=1}^n \), number of prototypes \( k \), penalty coefficient \( \lambda \).
Output: Prototypes \( \{u_j\}_{j=1}^k \) and reconstruction of data by archetypes \( \{\hat{x}_i\}_{i=1}^n \).

1: \((a_{ij}, b_{ij}) \leftarrow \arg\min_{a_{ij} \geq 0, b_{ij} \geq 0} \sum_{i=1}^n a_{ij} x_i - a_{ij} b_{ij} x_i + \lambda \sum_{i=1}^n a_{ij} \|x_i - b_{ij} x_i\|^2 \)  
2: for \( j = 1, \ldots, k \) do
3: \( u_j \leftarrow b_{ij} x_1 + \cdots + b_{ij} x_n \)
4: end for
5: for \( i = 1, \ldots, n \) do
6: \( \hat{x}_i \leftarrow a_{ij} u_1 + \cdots + a_{ij} u_k \)
7: end for
8: return \( \{u_j\}_{j=1}^k \), \( \{\hat{x}_i\}_{i=1}^n \)

A prototypical pair \((u_j, v_j)\) defines the regression function \( \hat{f} \) via

\[
\hat{f}(x_0) = a_{10} v_1 + \cdots + a_{k0} v_k
\]  

where \( a_{j0} \) are the barycentric coordinates of \( x_0 \) in prototypical analysis:

\[
\min_{a_{j0} \geq 0, \sum_{j=1}^k a_{j0} = 1} \sum_{j=1}^k a_{j0} \|x_0 - a_{j0} u_j\|^2 + \lambda \sum_{j=1}^k a_{j0} \|x_0 - u_j\|^2.
\]

Given the weights \( \{a_{ij}\} \) for reconstructing \( x_i \) in terms of the \( \{u_j\} \), the prototypes \( v_j \) are obtained by minimizing the squared errors of Equation 6 on \((x_i, y_i)\), i.e.

\[
v_j = c_j y_i, \quad c = \arg\min_{c_i \geq 0, \sum_{i=1}^n c_i = 1} \sum_{i=1}^n c_i (y_i - a_{ij} c_j y_i)^2.
\]

Figure 5 illustrates simple prototypical regression, kernel regression, regression tree and k nearest-neighbor regression on a one-dimensional synthetic data set.

4.2 Multiple Prototypical Regression

Multiple prototypical regression estimates the response \( y \) using \( m \) predictors \( \{x^{(l)}\}_{l=1}^m \) (again, both the response and each of the predictors can be vectorial.) As in simple prototypical regression, it
Figure 3: Prototypical analysis on two dimensional data with 4 prototypes and penalty 0.05. The data of the right figure contains one more outlier than the left figure, but this affects the location of the prototypes only minimally. The prototypes are visualized using ‘+’ signs.

finds prototypes for \( x^{(l)} \) and \( y \) and builds the regression function on prototypes.

The prototypes of \( x^{(l)} \) still come from direct prototypical analysis, i.e. solving Equation 4 for each \( \{x_i^{(l)}\}_{i=1}^n \). Each predictor has \( k_i \) prototypes and penalty coefficient \( \lambda_i \), these need not be the same across predictors. When finding prototypes for \( y \), we weight the prototypes of each \( x^{(l)} \) by an importance coefficient. Denoting by \( u_j^{(l)} \) the prototypes of \( x^{(l)} \), by \( v_j^{(l)} \) the prototypes of \( y \) and by \( \tau_l \) the importance coefficient corresponding to the \( l \)-th predictor, the regression function \( \hat{f} \) in multiple prototypical regression is given by

\[
\hat{f}(x_0) = \sum_{l=1}^{\#l} \frac{\sum_{j=1}^{\#j} \tau_l a_j^{(l)} v_j^{(l)}}{\sum_{j=1}^{\#j} \tau_l a_j^{(l)}}, \tag{9}
\]

where \( a_{j}^{(l)} \) are the barycentric coordinates of \( x_0^{(l)} \) in prototypical analysis as in Equation 7.

The importance coefficients \( \tau_l \) in Equation 9 are non-negative and add up to one. Both the importance coefficients and the prototypes of \( y \) are obtained by minimizing the squared errors of
Figure 4: Prototypical analysis on two dimensional data with penalty 0.05. The number $k$ of prototypes is set to 3, 4, 5, 6, 7 and 8. The prototypes are visualized using ‘+’ signs. Unlike archetypes, as the number of prototypes grows, they populate all data-rich areas.

Equation 9 on the data: denoting by $a^{(l)}_{ij}$ the weight of $v^{(l)}_j$ for reconstructing $x^{(l)}_i$,

$$v^{(l)}_j = \sum_{i=1}^{n} a^{(l)}_{ij} y_i \ , \ c, \tau = \arg \min_{c_{ij}, \tau_i \geq 0} \sum_{i=1}^{n} \left( \tau_i - \sum_{j=1}^{m} c_{ij} \right)^2 \, y_i - \sum_{j=1}^{m} \sum_{h=1}^{n} c^{(l)}_{ij} c^{(l)}_{j} y_{h} \right).$$

Here the optimization is carried out through the alternate minimization over the $c$ and $\tau$.

4.3 Applications

Iris Flowers We apply multiple prototypical regression to the data set for classification of Iris into species introduced by Fisher (1936). This includes three Iris species with four features for each flower: sepal length, sepal width, petal length and petal width. In this example, we treat the sepal and petal dimensions as two two-dimensional predictors and one-hot encode the three species as
Figure 5: 100 pairs $x_i, y_i$ are sampled from a Gaussian conditional distribution with conditional mean $\bar{y} = \sin(x) - x^3$ (the black curve), $x \sim U[0, 1]$, $y = \bar{y} + \varepsilon \sim N(0, 0.1)$. The red curves arise from regression. Top left panel: prototypical regression with 6 prototypes and penalty 0.01 (The prototypes of $x_i$ and $y_i$ are visualized using ' + ' signs.) Top right panel: kernel regression with Epanechnikov kernel with (half) window width $\lambda = 0.15$. Lower left panel: regression tree. Lower right panel: 10-nearest-neighbor regression.
Algorithm 3 Simple Prototypical Regression - Fitting

Input: Predictor data \( \{x_i\}_{i=1}^{n} \), response data \( \{y_i\}_{i=1}^{n} \), number of prototypes \( k \), penalty coefficient \( \lambda \).

Output: Prototypes \( \{u_j\}_{j=1}^{k} \) and \( \{v_j\}_{j=1}^{k} \) for predictor and response respectively.

1. \((a_{ij}), (b_{ij}) \leftarrow \arg\min_{a_{ij} \geq 0, b_{ij} \geq 0} \sum_{i=1}^{n} (x_i - a_{ij} b_{ij} x_i)^2 + \lambda \sum_{i=1}^{n} a_{ij} \) \( i = 1 \rightarrow 1, j = 1 \rightarrow 1 \)
2. for \( j = 1, \ldots, k \) do
3. \( u_j \leftarrow b_{ij} x_1 + \ldots + b_{ij} x_n \)
4. end for
5. \((c_{ij}) \leftarrow \arg\min_{c_{ij} \geq 0} \sum_{i=1}^{n} (y_i - a_{ij} c_{ij} y_i)^2 \) \( i = 1 \rightarrow 1, j = 1 \rightarrow 1 \)
6. for \( j = 1, \ldots, k \) do
7. \( v_j \leftarrow c_{ij} y_1 + \ldots + c_{ij} y_n \)
8. end for
9. return \( \{u_j\}_{j=1}^{k}, \{v_j\}_{j=1}^{k} \)

Algorithm 4 Simple Prototypical Regression - Prediction

Input: Value \( x_0 \) of the predictor, prototypes \( \{u_j\}_{j=1}^{k} \) and \( \{v_j\}_{j=1}^{k} \) for predictor and response respectively, penalty coefficient \( \lambda \).

Output: Predicted \( \hat{y}_0 \)

1. \((a_j) \leftarrow \arg\min_{a_j \geq 0} \|x_0 - a_j u_j\|^2 + \lambda \|x_0 - u_j\|^2 \) \( i = 1 \rightarrow 1 \)
2. \( \hat{y}_0 \leftarrow a_j v_1 + \ldots + a_j v_n \)
3. return \( \hat{y}_0 \)

\((1, 0, 0), (0, 1, 0)\) and \((0, 0, 1)\). Multiple prototypical regression predicts a probability vector given the sepal and petal features. The species with highest probability is then adopted as predicted label.

There are 150 samples in the Iris data set with 50 samples for each species. Using stratified sampling, we randomly split the samples into a training set of 105 samples and a test set of 45 samples. By grid search with cross validation on the training data, we pick the number of prototypes to be 11 and the penalty coefficient to be 0.1 for both features. The accuracy scores on the training and testing sets are shown in Table 1.

The Iris data set and the prototypes of the sepal and petal dimensions are shown in Figure 6. Figure 6 suggests the petal dimensions are more informative than the sepal’s for the classification task. This agrees with the importance coefficients of prototypical regression, which are \(3 \times 10^{-7}\) and
Algorithm 5 Multiple Prototypical Regression - Fitting

Input: Predictor data \( \{ x_i^{(1)} \}_{i=1}^{n}, \ldots, \{ x_i^{(m)} \}_{i=1}^{n} \), response data \( \{ y_i \}_{i=1}^{n} \), number of prototypes \( k_1, \ldots, k_m \), penalty coefficient \( \lambda_1, \ldots, \lambda_m \).

Output: Prototypes \( \{ u_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ u_j^{(m)} \}_{j=1}^{k_m} \) for predictors and \( \{ v_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ v_j^{(m)} \}_{j=1}^{k_m} \) for response, importance coefficients \( \tau_1, \ldots, \tau_m \).

1: for \( l = 1, \ldots, m \) do
2: \( (a_l^{(1)}), (b_{lj}^{(l)}) \leftarrow \arg \min_{a_l^{(1)} \geq 0} \sum_{i=1}^{n} \left( a_l^{(1)} b_{lj}^{(l)} x_{ih} - a_l^{(1)} b_{lj}^{(l)} x_{ih} + \lambda_l \right) + \lambda_l \sum_{i=1}^{n} a_l^{(1)} x_{ih} - \sum_{i=1}^{n} b_{lj}^{(l)} x_{ih} \)
3: for \( j = 1, \ldots, k_l \) do
4: \( u_j^{(l)} \leftarrow \sum_{l=1}^{m} b_{lj}^{(l)} x_{1h} \)
5: end for
6: end for

7: \( (c_{lj}^{(l)}), (\tau_l) \leftarrow \arg \min_{c_{lj}^{(l)} \geq 0, \tau_l \geq 0} \sum_{i=1}^{n} \left( c_{lj}^{(l)} y_{ih} - \tau_l \right) + \lambda_l \sum_{i=1}^{n} c_{lj}^{(l)} y_{ih} \)
8: for \( l = 1, \ldots, m \) do
9: for \( j = 1, \ldots, k_l \) do
10: \( v_j^{(l)} \leftarrow \sum_{l=1}^{m} c_{lj}^{(l)} y_{n} \)
11: end for
12: end for
13: return \( \{ u_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ u_j^{(m)} \}_{j=1}^{k_m} \), \( \{ v_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ v_j^{(m)} \}_{j=1}^{k_m} \), \( \tau_1, \ldots, \tau_m \)

0.9999997 for the sepal and petal dimensions respectively. Figure 7 shows the responses of this classification problem and the prototypes of the responses corresponding to the petal dimensions.

5. KERNE LS AND EXTENSION TO PROBABILITY DISTRIBUTIONS

5.1 Prototypical Learning with Kernels

Archetypal analysis, prototypical analysis and prototypical regression involve the data only through the pairwise inner products

\[ x_i, x_j \] and \( y_i, y_j \]

as follows from expanding the squared norms in Equation 3, 4, 8 and 10. Hence we can extend all three to reproducing kernel Hilbert spaces. Choosing a symmetric and positive semidefinite kernel
Figure 6: Sepal dimensions and petal dimensions of Iris flowers and their prototypes.

Figure 7: Species of Iris flowers and prototypes corresponding to petal dimensions. This plot of the three-dimensional object \((P_1, P_2, P_3)\) is represented here in barycentric coordinates, where the three vertices of the triangle correspond to the three species.
Algorithm 6 Multiple Prototypical Regression - Prediction

Input: Values $x_0 = x_0^{(1)}, \ldots, x_0^{(m)}$ of the predictors, prototypes $\{ u_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ u_j^{(m)} \}_{j=1}^{k_m}$ for predictors and $\{ v_j^{(1)} \}_{j=1}^{k_1}, \ldots, \{ v_j^{(m)} \}_{j=1}^{k_m}$ for response, importance coefficients $\tau_1, \ldots, \tau_m$, penalty coefficients $\lambda_1, \ldots, \lambda_m$.

Output: Predicted $\hat{y}_0$.

1: for $l = 1, \ldots, m$ do
2: \[ (a_j^{(l)}) \leftarrow \arg \min_{a_j^{(l)} \geq 0} \sum_{j=1}^{k_j} a_j^{(l)} u_j^{(l)} + \lambda_l \sum_{j=1}^{k_j} a_j^{(l)} x_0^{(l)} - u_j^{(l)} \] \[ \text{subject to } a_1^{(l)} + \cdots + a_{k_j}^{(l)} = 1 \]
3: end for
4: $\hat{y}_0 \leftarrow \sum_{l=1}^{m} \tau_l a_j^{(l)} v_j^{(l)}$
5: return $\hat{y}_0$

<table>
<thead>
<tr>
<th>prototypical regression</th>
<th>training score</th>
<th>test score</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.96</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Table 1: Accuracy score on Iris flowers data set.

function $K$, the map from $x_i$ to $h(x_i) = K(\cdot, x_i)$ yields the inner product

\[ K(x_i), h(x_j) = K(x_i, x_j), \]

which replaces the inner products in Equation 3, 4, 8 and 10, and extends archetypal analysis, prototypical analysis and prototypical regression to a (potentially infinite-dimensional) reproducing kernel Hilbert space.

5.2 Prototypical Learning on Distributions through Kernel Embedding

Probability distributions or samples thereof can also be mapped to a reproducing kernel Hilbert space via kernel embedding (see Berlinet and Thomas-Agnan, 2004; Gretton et al., 2006; Smola et al., 2007; Sriperumbudur et al., 2010; Sejdinovic et al., 2012; Muandet et al., 2017). With a symmetric, positive semidefinite kernel function $K(\cdot, \cdot)$ on $X \times X$, the kernel embedding $g$ maps a probability measure $\mu(\cdot)$ on $X$ to a reproducing kernel Hilbert space through

\[ \mu(\cdot) \rightarrow g(\mu(\cdot)) = \int_X K(\cdot, x) d\mu(x), \tag{11} \]
with induced inner product given by
\[
g(\mu_1(\cdot),\mu_2(\cdot)) = \int_{\mathbb{R}^d} K(x_1, x_2) d\mu_1(x_1) d\mu_2(x_2). \tag{12}
\]

Kernel embedding does not necessarily yield an injective map; Sriperumbudur et al. (2010) give several criteria for whether a kernel induces an injective embedding for distributions on \(\mathbb{R}^d\) and \(\mathbb{T}^d\). Some commonly used kernels on \(\mathbb{R}^d\) for injective kernel embeddings are listed in Table 2. The Gaussian, Laplacian and \(B_{2n+1}\)-spline kernels are shown to induce injective embeddings in Sriperumbudur et al. (2010). The energy distance kernel induces an embedding well-defined on distributions with finite first moment. The energy distance \(D_{\text{ED}}\) (Székely and Rizzo, 2013; Rizzo and Székely, 2016):
\[
D_{\text{ED}}^2(\mu_1(\cdot),\mu_2(\cdot)) = 2 \int_{\mathbb{R}^d} \|x_1 - x_2\| d\mu_1(x_1) d\mu_2(x_2) - \int_{\mathbb{R}^d} \|x_1 - x_2\| d\mu_1(x_1) d\mu_2(x_2)
\]
is proved in Klebanov (2002) to yield a metric, implying that the energy distance kernel induces an injective embedding.

Replacing the integrals in Equation 11 and 12 by the corresponding empirical means gives the kernel embedding and induced inner product for samples of distributions. Given samples \(\{x_i\}_{i=1}^n\) of \(\mu\), the kernel embedding for the empirical distribution \(\hat{\mu}\) is
\[
\hat{\mu}(\cdot) \mapsto g(\hat{\mu}(\cdot)) = \frac{1}{n} \sum_{i=1}^n K(\cdot, x_i),
\]
and given samples \(\{x_i^{(1)}\}_{i=1}^{n_1}, \{x_i^{(2)}\}_{i=1}^{n_2}\) of \(\mu_1\) and \(\mu_2\), the induced inner product of the empirical distributions \(\hat{\mu}_1\) and \(\hat{\mu}_2\) is
\[
g(\hat{\mu}_1(\cdot),\hat{\mu}_2(\cdot)) = \frac{1}{n_1 n_2} \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} K(x_i^{(1)}, x_i^{(2)}).
\]

In general, the time complexity of evaluating the inner product is \(O(n_1 n_2)\). For the Gaussian kernel, the time complexity for the inner product can be reduced to \(O(n_1 + n_2)\) via the fast Gauss transform.
Table 2: Some commonly used kernels on $\mathbb{R}^d$ for injective kernel embeddings. For $B_{2n+1}$-spline, $B_{2n+1}(x) = \ast_{1}^{(2n+2)} 1_{[-\frac{1}{2}, \frac{1}{2}]}(x)$, where the symbol $\ast_{1}^{(2n+2)}$ represents the $(2n + 2)$-fold convolution. (Greengard and Strain, 1991) or the improved fast Gauss transform (Yang et al., 2003). For the energy distance kernel on sorted samples of one-dimensional distributions, the time complexity of evaluating the inner product is $O(n_1 + n_2)$, as shown in Appendix A.

We can extend archetypal analysis, prototypal analysis and prototypal regression to distributions with the inner products induced by kernel embedding. In archetypal/prototypal analysis, the archetypes/prototypes are mixtures of the input distributions and their mixtures are used to reconstruct the input distributions. In prototypal regression, we can have distributions as predictors, responses or both. In multiple prototypal regression, we can blend numerical, categorical and distributional predictors.

5.3 Applications

Smartphone-based Human Activities Recognition Data Set The smartphone-based human activities recognition data set from Anguita et al. (2013) and Reyes-Ortiz et al. (2016) contains activity data collected by smartphone's inertial sensors. In their experiments, 30 volunteers conducted 6 activities: walking, walking upstairs, walking downstairs, sitting, standing and laying while wearing a wrist-mounted smartphone. The data set contains raw and processed data. The raw data are the triaxial signals from the accelerometer and the gyroscope of smartphones at a constant rate of 50Hz for each activity. The processed data include statistics, such as the mean, standard deviation and auto correlation of the raw signals, and other data, such as the magnitude and the fast Fourier transform of the raw signals.

Anguita et al. (2013) and Reyes-Ortiz et al. (2016) use the processed data to classify the activities. We use the raw data instead, i.e. the triaxial signals from the accelerometer and gyroscope.
Table 3: Confusion matrix of multiple prototypical regression on smartphone-based human activities recognition data set. The rows are the actual classes and the columns are the predicted classes.

<table>
<thead>
<tr>
<th></th>
<th>walk</th>
<th>upstairs</th>
<th>downstairs</th>
<th>sit</th>
<th>stand</th>
<th>lay</th>
</tr>
</thead>
<tbody>
<tr>
<td>walk</td>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>upstairs</td>
<td>1</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>downstairs</td>
<td>0</td>
<td>0</td>
<td>18</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>sit</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>stand</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>lay</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 4: Importance coefficients of multiple prototypical regression on smartphone-based human activities recognition data set.

<table>
<thead>
<tr>
<th></th>
<th>accelerometer</th>
<th>gyroscope</th>
</tr>
</thead>
<tbody>
<tr>
<td>importance coeffs</td>
<td>0.44</td>
<td>0.56</td>
</tr>
</tbody>
</table>

Each trial in the raw data set contains two three-dimensional time series of the accelerometer and the gyroscope respectively and a label of the activity. We divide the data set into a training data set of 772 trials and a test data set of 84 trials. Multiple prototypical regression is applied for this classification task. The samples of triaxial signals from the accelerometer and the gyroscope are the two predictors in multiple prototypical regression and energy distance kernel is used for kernel embedding. The labels are binarized via one-hot encoding. The number of prototypes is set to be 70 and the penalty coefficient is set to be 1 for both predictors. We achieve a 97.62% accuracy on the testing subset. The confusion matrix for the test data is shown in Table 3, the importance coefficients are listed in Table 4.

EPA Outdoor Air Quality Data Set The EPA Outdoor Air Quality Data (US Environmental Protection Agency, 2017) collects pollutant and meteorological data at outdoor monitors across the United States, Puerto Rico, and the U. S. Virgin Islands. This data set contains hourly data of criteria gases (Ozone, SO₂, CO and NO₂), toxics and precursors (HAPs, VOCs, NONOxNOy and lead), particulates (PM2.5 FRM/ FEM Mass, PM2.5 non FRM/ FEM Mass, PM10 Mass and PM2.5 Speciation) and meteorological data (winds, temperature, barometric pressure, relative humidity and dew point).

We use multiple prototypical regression to estimate the distributions of the nitrogen dioxide
Table 5: Importance coefficients of multiply prototypical regression on EPA outdoor air quality data set.

(NO₂) density from the geophysical locations (the latitude and longitude of the stations) and the distributions of the meteorological data. The meteorological data that we use are the one-dimensional distribution of wind speed, the one-dimensional distribution of wind direction and one-dimensional distribution of outdoor temperature. The training data set contains the data collected in the year 2016 at 200 stations and the test data set contains the data collected in the same year at 23 other stations. We use the energy distance kernel for embedding. The number of prototypes is set to 40 and the penalty coefficient to 0.1 for all predictors. The importance coefficients are listed in Table 5 and the out-of-sample predictions are illustrated in Figure 8.

6. CONCLUSIONS

We have proposed and developed prototypical analysis and regression, two robust extensions of archetypal analysis. In addition, we have shown how these methodologies can be extended via kernel embedding to handle learning problems where the data points are probability distributions known through samples. Here the interpretability associated with the convex combinations involved is clearest, as these combinations can be interpreted as mixtures of distributions.

Prototypical analysis adds to the objective function of archetypal analysis a term that penalizes the use of distant prototypes for the reconstruction of data points. It can be regarded of as an interpolation between archetypal analysis—corresponding to a zero value of the penalization parameter λ—and k-means, which arises as λ → ∞. This adds robustness to outliers and a sense of locality, which becomes particularly useful when the methodology is used for regression.

We illustrate through real-life examples the applicability of the procedure, particularly to scenarios that blend numerical and distributional features or that have probability distributions as labels to predict.
Figure 8: Out-of-sample prediction of NO$_2$ density distribution. The black curves are the true NO$_2$ distributions at each station and the red curves are the predicted NO$_2$ distributions by multiple prototypical regression.

A. ENERGY DISTANCE KERNEL OF ONE-DIMENSIONAL DISTRIBUTIONS

The energy distance kernel on distributions $\mu$, $\nu$ can be estimated using their samples $\{x_i\}_{i=1}^{n_x}$, $\{y_j\}_{j=1}^{n_y}$ through the empirical mean:

$$k_{ED}(\mu, \nu) \approx \frac{1}{n_x} \sum_{i=1}^{n_x} ||x_i|| + \frac{1}{n_y} \sum_{j=1}^{n_y} ||y_j|| - \frac{1}{n_x n_y} \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} ||x_i - y_j||.$$  \hspace{1cm} \text{(A.1)}

The time complexity of evaluating Equation A.1 is $O(n_x n_y)$.

For one-dimensional distributions, the time complexity of evaluating Equation A.1 can be reduced to the linear $O(n_x + n_y)$ when the samples $\{x_i\}_{i=1}^{n_x}, \{y_j\}_{j=1}^{n_y}$ are sorted, as illustrated in Algorithm 7. The intuition behind is that each term in $\sum_{i=1}^{n_x} \sum_{j=1}^{n_y} ||x_i - y_j||$ can be expanded into

$$||x_i - y_j|| = 1_{x_i > y_j} (x_i - y_j) - 1_{x_i \leq y_j} (x_i - y_j) = (1_{x_i > y_j} - 1_{x_i \leq y_j}) x_i + (1_{x_i \leq y_j} - 1_{x_i > y_j}) y_j,$$
Algorithm 7 Energy Distance Kernel of 1D distributions

Input: Sorted samples \( \{x_i\}_{i=1}^{n_x}, \{y_j\}_{j=1}^{n_y} \) of 1D distributions \( \mu, \nu \).
Output: Empirical estimation of energy distance kernel \( k_{ED}(\mu, \nu) \)

1: \( \text{sum}_x \leftarrow 0, \text{sum}_y \leftarrow 0, i \leftarrow 1, j \leftarrow 1 \)
2: while \( i \leq n_x \) and \( j \leq n_y \) do
3:  if \( x_i \leq y_j \) then
4:   \( \text{sum}_x \leftarrow \text{sum}_x + \{(j - 1) - [n_y - (j - 1)]\} x_i \)
5:   \( i \leftarrow i + 1 \)
6:  else
7:   \( \text{sum}_y \leftarrow \text{sum}_y + \{(i - 1) - [n_x - (i - 1)]\} y_j \)
8:   \( j \leftarrow j + 1 \)
9: end if
10: end while
11: if \( i > n_x \) then
12:  \( \text{sum}_x \leftarrow \text{sum}_x + n_x \sum_{k=i}^{y_x} y_k \)
13: else
14:  \( \text{sum}_x \leftarrow \text{sum}_x + n_x \sum_{k=i}^{y_x} y_k \)
15: end if
16: \( k_{ED}(\mu, \nu) \leftarrow \frac{\sum_{k=1}^{x} x_k / n_x + \sum_{k=1}^{y} y_k / n_y - (\text{sum}_x + \text{sum}_y) / (n_x n_y)}{n_x n_y} \)
17: return \( k_{ED}(\mu, \nu) \)

yielding

\[
\|x_i - y_j\| = \sum_{i=1}^{x} \sum_{j=1}^{y} \left(1_{x_i > y_j} - 1_{x_i \leq y_j}\right) x_i + \left(1_{x_i \leq y_j} - 1_{x_i > y_j}\right) y_j . \tag{A.2}
\]

Equation (A.2) implies that we only need to count how many \( y_j \)'s are smaller than each \( x_i \) and how many \( x_i \)'s are smaller than each \( y_j \). If the samples are sorted, this counting can be done in linear time.

REFERENCES


