

# Unsupervised Learning

Sanjay Lall and Stephen Boyd

EE104

Stanford University

# Unsupervised learning

## Unsupervised learning

- ▶ in *supervised learning* we deal with pairs of records  $u, v$
- ▶ goal is to predict  $v$  from  $u$  using a prediction model
- ▶ the output records  $v^i$  'supervise' the learning of the model
  
- ▶ in *unsupervised learning*, we deal with only records  $u$
- ▶ goal is to *build a data model* of  $u$ , in order to
  - ▶ *reveal structure* in  $u$
  - ▶ *impute missing entries* (fields) in  $u$
  - ▶ *detect anomalies*
- ▶ yes, the first goal is vague . . .

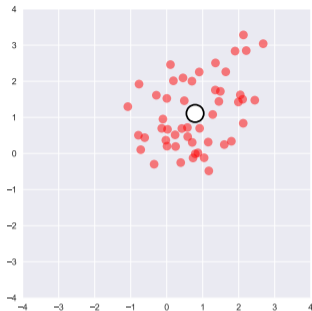
## Embedding

- ▶ as usual we embed raw data  $u$  into a feature vector  $x = \phi(u) \in \mathbf{R}^d$
- ▶ we then build a data model for the feature vectors
- ▶ we un-embed when needed, to go back to the raw vector  $u$
- ▶ so we'll work with feature vectors from now on
- ▶ (embedded) data set has the form  $x^1, \dots, x^n \in \mathbf{R}^d$

## Data model via loss function

- ▶ a *data model* tells us what the vectors in some data set 'look like'
- ▶ can be expressed quantitatively by an *implausibility function* or *loss function*  $\ell : \mathbf{R}^d \rightarrow \mathbf{R}$
- ▶  $\ell(x)$  is how implausible  $x$  is as a data point
  - ▶  $\ell(x)$  small means  $x$  'looks like' our data, or is 'typical'
  - ▶  $\ell(x)$  large means  $x$  does not look like our data
- ▶ if our model is probabilistic, *i.e.*,  $x$  comes from a density  $p(x)$ , we can take  $\ell(x) = -\log p(x)$ , the *negative log density*
- ▶ other names for  $\ell(x)$ : surprise, perplexity, ...
- ▶  $\ell$  is often *parametrized* by a vector or matrix  $\theta$ , and denoted  $\ell_\theta(x)$

## A simple constant model



- ▶ data model:  $x$  is near a fixed vector  $\theta \in \mathbf{R}^d$
- ▶  $\theta \in \mathbf{R}^d$  parametrizes the model
- ▶ some implausibility functions:
  - ▶  $\ell_{\theta}(x) = \|\mathbf{x} - \theta\|_2^2 = \sum_{i=1}^d (x_i - \theta_i)^2$  (square loss)
  - ▶  $\ell_{\theta}(x) = \|\mathbf{x} - \theta\|_1 = \sum_{i=1}^d |x_i - \theta_i|$  (absolute loss)

## $k$ -means data model



► data model:  $x$  is close to one of the  $k$  representatives  $\theta_1, \dots, \theta_k \in \mathbf{R}^d$

► quantitatively: for our data points  $x$ , the quantity

$$\ell_{\theta}(x) = \min_{i=1, \dots, k} \|x - \theta_i\|_2^2$$

i.e., the minimum distance squared to the representatives, is small

►  $d \times k$  matrix  $\theta = [\theta_1 \cdots \theta_k]$  parametrizes the  $k$ -means data model

# Role of loss function in supervised and unsupervised learning

- ▶ in supervised learning

- ▶ a loss function is used to choose a particular predictor from a parameterized family of predictors
- ▶ once we've chosen and validated our predictor, we don't care about the loss function

- ▶ in unsupervised learning

- ▶ a loss function characterizes what the data looks like
- ▶ the loss function is our data model, and is itself our primary goal

## Anomaly detector

- ▶ a data model allows us to identify *suspicious* or *anomalous* feature vectors
- ▶ first choose or fit a data model, with loss function  $\ell$
- ▶ find the 99th (say) percentile  $t$  of  $\ell(x^i)$  on some test data
- ▶ flag a feature vector  $x$  as anomalous if  $\ell(x) > t$

# Imputing missing entries

## Imputing missing entries

- ▶ suppose  $x$  has some entries missing, denoted  $?$  or  $NA$  or  $NaN$
- ▶  $\mathcal{K} \subseteq \{1, \dots, d\}$  is the set of *known entries*
- ▶ we use our data model to guess or *impute* the missing entries
- ▶ we'll denote the imputed vector as  $\hat{x}$
- ▶  $\hat{x}_i = x_i$  for  $i \in \mathcal{K}$
- ▶ imputation example, with  $\mathcal{K} = \{1, 3\}$

$$x = \begin{bmatrix} 12.1 \\ ? \\ -2.3 \\ ? \end{bmatrix} \implies \hat{x} = \begin{bmatrix} 12.1 \\ -1.5 \\ -2.3 \\ 3.4 \end{bmatrix}$$

- ▶ we are imputing or guessing  $\hat{x}_2 = -1.5$ ,  $\hat{x}_4 = 3.4$
- ▶ the other entries we know:  $\hat{x}_1 = x_1 = 12.1$ ,  $\hat{x}_3 = x_3 = -2.3$

## Application: Recommendation system

- ▶ features are movies; examples are customer ratings or *?* if the customer has not rated that movie
- ▶ imputed entries are our guess of *what rating the customer would give, if they rated that movie*
- ▶ we *recommend* movies to a customer
  - ▶ that they have not rated
  - ▶ and for which the imputed rating is large

## Application: Filling in missing features for supervised learning

- ▶ setting: data set in supervised learning problem contains some missing features
- ▶ common approach: ignore any record that has any feature missing
- ▶ in some cases, you'll lose much of your data (and therefore do poorly at fitting a prediction model)
- ▶ alternative approach:
  - ▶ use imputation to fill in (presumably few) missing feature entries
  - ▶ then proceed with supervised learning

## Application: Detecting anomalous entries

- ▶ goal is to flag suspicious or anomalous *entries* in  $x$
- ▶ method based on imputation: for each  $i$ 
  - ▶ pretend entry  $x_i$  in  $x$  is ?
  - ▶ find its imputed value  $\hat{x}_i$  (based on all other entries of  $x$ )
  - ▶ if  $x_i$  and  $\hat{x}_i$  are very different, flag  $x_i$  as anomalous

## Supervised learning as special case of imputation

- ▶ suppose we wish to predict  $y \in \mathbf{R}^m$  based on  $x \in \mathbf{R}^d$
- ▶ we have some training data  $x^1, \dots, x^n, y^1, \dots, y^n$
- ▶ define  $(d + m)$ -vector  $\tilde{x} = (x, y)$
- ▶ build data model for  $\tilde{x}$  using training data  $\tilde{x}^1, \dots, \tilde{x}^n$
- ▶ to predict  $y$  given  $x$ , impute last  $m$  entries of  $\tilde{x} = (x, ?)$

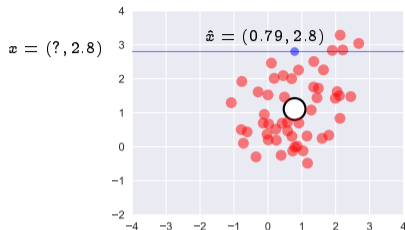
## Imputation using a data model

- ▶ given partially specified vector  $x$  we minimize over the unknown entries:

$$\begin{aligned} & \text{minimize} && \ell_{\theta}(\hat{x}) \\ & \text{subject to} && \hat{x}_i = x_i, \quad i \in \mathcal{K} \end{aligned}$$

- ▶ *i.e.*, impute the unknown entries to minimize the implausibility, subject to the given known entries
- ▶ ... a simple and natural method

## Imputing with constant data model



- ▶ given  $x$  with some entries unknown
- ▶ constant data model with implausibility function  $\ell_{\theta}(x) = \|x - \theta\|_2^2$
- ▶ we minimize  $(\hat{x}_1 - \theta_1)^2 + \dots + (\hat{x}_d - \theta_d)^2$  subject to  $\hat{x}_i = x_i$  for  $i \in \mathcal{K}$
- ▶ so  $\hat{x}_i = x_i$  for  $i \in \mathcal{K}$
- ▶ for  $i \notin \mathcal{K}$ , we take  $\hat{x}_i = \theta_i$
- ▶ *i.e.*, for the unknown entries, guess the model parameter entries
- ▶ example has  $\theta = (0.79, 1.11)$

## Imputing with $k$ -means data model

- ▶ given  $x$  with some entries unknown
- ▶  $k$ -means data model with implausibility function  $\ell_{\theta}(x) = \min_{i=1, \dots, k} \|x - \theta_i\|_2^2$
- ▶ find nearest representative  $\theta_j$  to  $x$ , using only known entries
- ▶ *i.e.*, find  $j$  that minimizes  $\sum_{i \in \mathcal{K}} (x_i - (\theta_j)_i)^2$
- ▶ guess  $\hat{x}_i = (\theta_j)_i$  for  $i \notin \mathcal{K}$
- ▶ *i.e.*, for the unknown entries, guess the entries of the closest representative

## Validating imputation

we can validate a proposed data model (and imputation method):

- ▶ divide data into a training and a test set
- ▶ build data model on the training set
- ▶ mask some entries in the vectors in the test set (*i.e.*, replace them with ?)
- ▶ impute these entries and evaluate the average error of the imputed values, *e.g.*, the RMSE

# Fitting data models

## Generic fitting method

- ▶ given data  $x^1, \dots, x^n$  (with no missing entries), and parametrized implausibility function  $\ell_\theta(x)$
- ▶ how do we choose the parameter  $\theta$ ?
  
- ▶ average implausibility or *empirical loss* is

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n \ell_\theta(x^i)$$

- ▶ choose  $\theta$  to minimize  $\mathcal{L}(\theta)$ , (possibly) subject to  $\theta \in \Theta$ , the set of acceptable parameters
- ▶ *i.e.*, choose parameter  $\theta$  so the observed data is least implausible

## Fitting a constant model with sum squares loss

▶ sum squares implausibility function  $\ell_{\theta}(x) = \|x - \theta\|^2$

▶ empirical loss is

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n \|x^i - \theta\|_2^2$$

▶ minimizing over  $\theta$  yields

$$\theta = \frac{1}{n} \sum_{i=1}^n x^i$$

the mean of the data vectors

## Fitting a constant model with sum absolute loss

▶ sum absolute implausibility function  $\ell_{\theta}(x) = \|x - \theta\|_1$

▶ empirical loss is

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n \|x^i - \theta\|_1$$

▶ minimizing over  $\theta$  yields

$$\theta = \text{median}(x^1, \dots, x^n)$$

the elementwise median of the data vectors

## Fitting a $k$ -means model

- ▶ implausibility function  $\ell_\theta(x) = \min_{j=1, \dots, k} \|x - \theta_j\|^2$
- ▶ parameter is  $d \times k$  matrix with columns  $\theta_1, \dots, \theta_k$
- ▶ empirical loss is

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n \min_{j=1, \dots, k} \|x^i - \theta_j\|^2$$

- ▶ this is the  $k$ -means objective function!
- ▶ we can use the  $k$ -means algorithm to (approximately) minimize  $\mathcal{L}(\theta)$ , *i.e.*, fit a  $k$ -means model

## $k$ -means algorithm

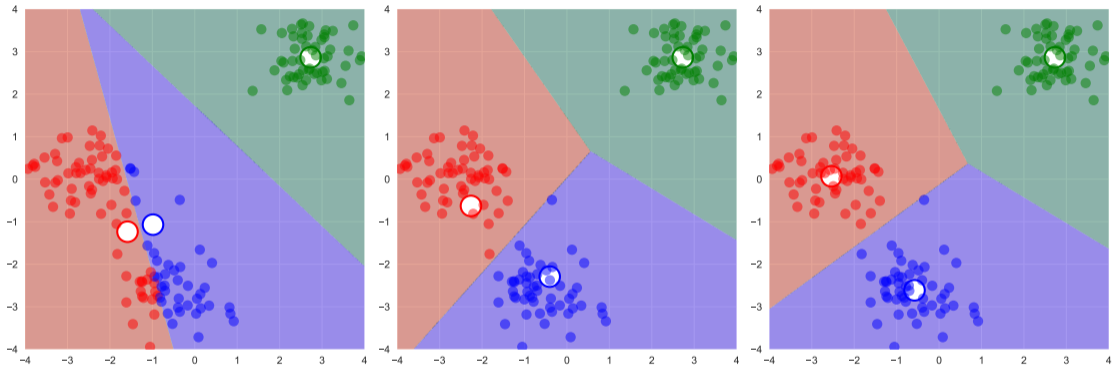
- ▶ define the *assignment* or *clustering* vector  $c \in \mathbb{R}^n$
- ▶  $c_i$  is the cluster that data vector  $x^i$  is in (so  $c_i \in \{1, \dots, k\}$ )
- ▶ to minimize

$$\mathcal{L}(\theta) = \frac{1}{n} \sum_{i=1}^n \min_{j=1, \dots, k} \|x^i - \theta_j\|^2$$

we minimize  $\frac{1}{n} \sum_{i=1}^n \|x^i - \theta_{c_i}\|^2$  over both  $c$  and  $\theta_1, \dots, \theta_k$

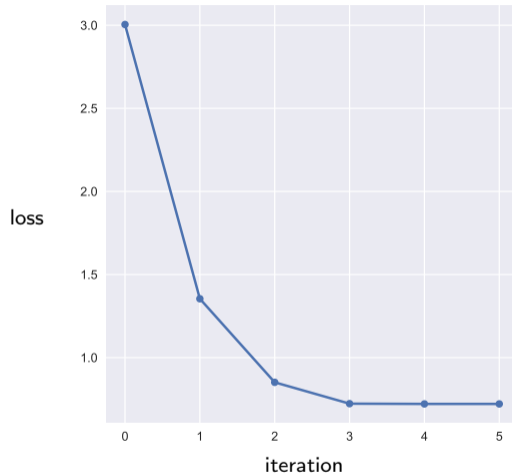
- ▶ we can minimize over  $c$  using  $c_i = \operatorname{argmin}_j \|x^i - \theta_j\|^2$
- ▶ we can minimize over  $\theta_1, \dots, \theta_k$  using  $\theta_i$  as the average of  $\{x^j \mid c_j = i\}$
- ▶  $k$ -means algorithm alternates between these two steps
- ▶ it is a heuristic for (approximately) minimizing  $\mathcal{L}(\theta)$

## $k$ -means example



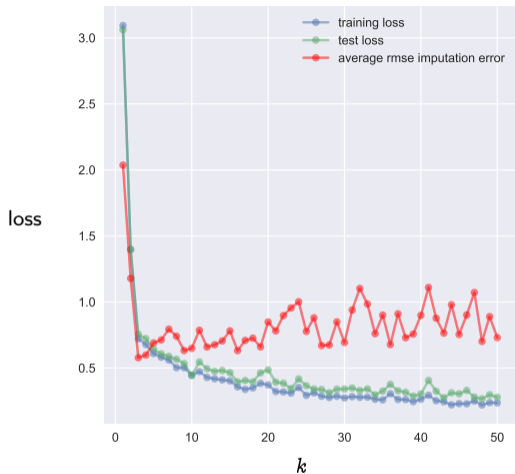
► 200 data points; reserve 40 for test

## $k$ -means example



► convergence after 4 iterations

## $k$ -means example



- ▶ fit  $k$ -mean data model for  $k = 1, 2, \dots, 50$
- ▶ validate by removing randomly either  $u_1$  or  $u_2$  from each record in test set