SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

01.112 Machine Learning, Spring 2018
Lecture Notes for Week 2

4. Linear Regression, Ridge Regression

Last update: Thursday 1st February, 2018 11:09

1 Linear Regression

A linear regression function is simply a linear function of the feature vectors, i.e.,

$$f(x;\theta,\theta_0) = \theta \cdot x + \theta_0 = \sum_{i=1}^{d} \theta_i x_i + \theta_0 \tag{1}$$

Each setting of the parameters θ and θ_0 gives rise to a slightly different regression function. Collectively, different parameter choices $\theta \in \mathcal{R}^d$, $\theta_0 \in \mathcal{R}$, give rise to the set of functions $\mathcal{F}$ that we are entertaining. While this set of functions seems quite simple, just linear, the power of $\mathcal{F}$ is hidden in the feature vectors. Indeed, we can often construct different feature representations for objects. For example, there are many ways to map past values of financial assets into a feature vector x, and this mapping is typically completely under our control. This "freedom" gives us a lot of power, and we will discuss how to exploit it later on. For now, we assume that a proper representation has been found, denoting feature vectors simply as x.

Our learning task is to choose one $f \in \mathcal{F}$, i.e., choose parameters $\hat{\theta}$ and $\hat{\theta}_0$, based on the training set $S_n = \{(x^{(t)}, y^{(t)}), t = 1,\ldots,n\}$, where $y^{(t)} \in \mathcal{R}$ (response). As before, our goal is to find $f(x;\hat{\theta},\hat{\theta}_0)$ that would yield accurate predictions on yet unseen examples. There are several problems to address:

1. How do we measure errors? What is the criterion by which we choose $\hat{\theta}$ and $\hat{\theta}_0$ based on the training set?

2. What algorithm can we use to optimize the training criterion? How does the algorithm scale with the dimension (feature vectors may be high dimensional) or the size of the training set (the dataset may be large)?

3. When the size of the training set is not large enough in relation to the number of parameters (dimension) there may be degrees of freedom, i.e., directions in the parameter space, that remain unconstrained by the data. How do we set those degrees of freedom? This is a part of a broader problem known as regularization. The question is how to softly constrain the set of functions $\mathcal{F}$ to achieve better generalization.

1.1 Empirical Risk and the Least Squares Criterion

Similar to the classification setting, we will measure training error in terms of empirical risk

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n} \text{Loss}(y^{(t)} - \theta \cdot x^{(t)}) \tag{2}$$

where, for simplicity, we have dropped the parameter θ_0. It will be easy to add it later on in the appropriate place. Note that, unlike in classification, the loss function now depends on the difference between the real valued target value $y^{(t)}$ and the corresponding linear prediction $\theta \cdot x^{(t)}$. There are many possible ways of defining the loss function. We will use here a simple squared error: $\text{Loss}(z) = z^2/2$. The idea is to permit small discrepancies (we expect the responses to include noise) but heavily penalize large deviations (that typically indicate poor parameter choices). As a result, we have

$$R_n(\theta) = \frac{1}{n}\sum_{t=1}^{n} \text{Loss}(y^{(t)} - \theta \cdot x^{(t)}) = \frac{1}{n}\sum_{t=1}^{n} (y^{(t)} - \theta \cdot x^{(t)})^2/2 \tag{3}$$

Our learning goal is not to minimize $R_n(\theta)$; it is just the best we can do (for now). Minimizing $R_n(\theta)$ is a surrogate criterion since we do not have a direct access to the test or generalization error

$$R^{test}_{n'}(\theta) = \frac{1}{n}\sum_{t=n+1}^{n+n'} (y^{(t)} - \theta \cdot x^{(t)})^2/2 \tag{4}$$

Let us briefly consider how $R_n(\theta)$ and $R^{test}_{n'}(\theta)$ are related. If we select $\hat{\theta}$ by minimizing $R_n(\theta)$, our performance will be measured according to $R^{test}_{n'}(\hat{\theta})$. This test error can be large for two different reasons. First, we may have a large *estimation error*. This means that, even if the true relationship between x and y is linear, it is hard for us to estimate it on the basis of a small (and potentially noisy) training set S_n. Our estimated parameters $\hat{\theta}$ will not be entirely correct. The larger the training set is, the smaller the estimation error will be. The second type of error on the test set is *structural error*. This means that we may be estimating a linear mapping from x to y when the true underlying relationship is highly non-linear. Clearly, we cannot do very well in this case, regardless of how large the training set is. In order to reduce structural error, we would have to use a larger set of functions $\mathcal{F}$. But, given a noisy training set S_n, it will be harder to select the correct function from such larger $\mathcal{F}$, and our estimation error will increase. Finding the balance between the estimation and structural errors lies at the heart of learning problems.

When we formulate linear regression problem as a statistical problem, we can talk about the structural error as "bias", while the estimation error corresponds to the "variance" of the estimator $\hat{\theta}(S_n)$. The parameters $\hat{\theta}$ are obtained with the help of training data S_n and thus can be viewed as functions of S_n.

1.2 Optimizing the Least Squares Criterion

Perhaps the simplest way to optimize the least squares objective $R_n(\theta)$ is to use the stochastic gradient descent method discussed earlier in the classification context. Our case here is easier, in fact, since $R_n(\theta)$ is everywhere differentiable. At each step of the algorithm, we select one training example at random, and nudge parameters in the opposite direction of the gradient

$$\nabla_\theta (y^{(t)} - \theta \cdot x^{(t)})^2/2 = (y^{(t)} - \theta \cdot x^{(t)}) \nabla_\theta (y^{(t)} - \theta \cdot x^{(t)}) = -(y^{(t)} - \theta \cdot x^{(t)}) x^{(t)} \tag{5}$$

As a result, the algorithm can be written as

$$\begin{aligned} &\text{set } \theta^{(0)} = 0 \\ &\quad \text{randomly select } t \in \{1, \ldots, n\} \\ &\quad\quad \theta^{(k+1)} = \theta^{(k)} + \eta_k (y^{(t)} - \theta \cdot x^{(t)}) x^{(t)} \end{aligned} \tag{6}$$

where η_k is the learning rate (e.g., $\eta_k = 1/(k+1)$). Recall that in classification the update was performed only if we made a mistake. Now the update is proportional to discrepancy $(y^{(t)} - \theta \cdot x^{(t)})$ so that even small mistakes count, just less. As in the classification context, the update is "self-correcting". For example, if our prediction is lower than the target, i.e., $y(t) > \theta \cdot x^{(t)}$, we would move the parameter vector in the positive direction of $x^{(t)}$ so as to increase the prediction next time $x^{(t)}$ is considered. This would happen in the absence of updates based on other examples.

1.3 Closed Form Solution

We can also try to minimize $R_n(\theta)$ directly by setting the gradient to zero. Indeed, since $R_n(\theta)$ is a convex function of the parameters, the minimum value is obtained at a point (or a set of points) where the gradient is zero. So, formally, we find $\hat{\theta}$ for which $\nabla R_n(\theta)_{\theta=\hat{\theta}} = 0$. More specifically,

$$\nabla R_n(\theta)_{\theta=\hat{\theta}} = \frac{1}{n} \sum_{t=1}^{n} \nabla_\theta \left\{ (y^{(t)} - \theta \cdot x^{(t)})^2/2 \right\}_{|\theta=\hat{\theta}} \tag{7}$$

$$= \frac{1}{n} \sum_{t=1}^{n} \{ -(y^{(t)} - \hat{\theta} \cdot x^{(t)}) x^{(t)} \} \tag{8}$$

$$= -\frac{1}{n} \sum_{t=1}^{n} y^{(t)} x^{(t)} + \frac{1}{n} \sum_{t=1}^{n} (\hat{\theta} \cdot x^{(t)}) x^{(t)} \tag{9}$$

$$= \underbrace{-\frac{1}{n} \sum_{t=1}^{n} y^{(t)} x^{(t)}}_{=b} + \underbrace{\frac{1}{n} \sum_{t=1}^{n} x^{(t)} (x^{(t)})^T}_{=A} \hat{\theta} \tag{10}$$

$$= -b + A\hat{\theta} = 0 \tag{11}$$

where we have used the fact that $\hat{\theta} \cdot x^{(t)}$ is a scalar and can be moved to the right of vector $x^{(t)}$. We have also subsequently rewritten the inner product as $\hat{\theta} \cdot x^{(t)} = (x^{(t)})^T \hat{\theta}$. As a result, the equation

for the parameters can be expressed in terms of a $d \times 1$ column vector b and a $d \times d$ matrix A as $A\theta = b$. When the matrix A is invertible, we can solve for the parameters directly: $\hat{\theta} = A^{-1}b$. In order for A to be invertible, the training points $x^{(1)}, \ldots, x^{(n)}$ must *span* $\mathcal{R}^d$. Naturally, this can happen only if $n \geq d$, and is therefore more likely to be the case when the dimension d is small in relation to the size of the training set n. Another consideration is the cost of actually inverting A. Roughly speaking, you will need $\mathcal{O}(d^3)$ operations for this. If $d = 10,000$, this can take a while, making the stochastic gradient updates more attractive.

In solving linear regression problems, the matrix A and vector b are often written in a slightly different way. Specifically, define $X = [x^{(1)}, \ldots, x^{(n)}]^T$. In other words, X^T has each training feature vector as a column; X has them stacked as rows. If we also define $\vec{y} = [y^{(1)}, \ldots, y^{(n)}]^T$ (column vector), then you can easily verify that

$$b = \frac{1}{n} X^T \vec{y}, \quad A = \frac{1}{n} X^T X \tag{12}$$

2 Ridge Regression

What happens when A is not invertible? In this case the training data provide no guidance about how to set some of the parameter directions. In other words, the learning problem is ill-posed. The same issue inflicts the stochastic gradient method as well though the initialization $\theta^{(0)} = 0$ helps set the parameters to zero for directions outside the span of the training examples. The simple fix does not solve the broader problem, however. How should we set the parameters when we have insufficient training data?

We will modify the estimation criterion, the mean squared error, by adding a *regularization term*. The purpose of this term is to bias the parameters towards a default answer such as zero. The regularization term will "resist" setting parameters away from zero, even when the training data may weakly tell us otherwise. This resistance is very helpful in order to ensure that our predictions generalize well. The intuition is that we opt for the "simplest answer" when the evidence is absent or weak.

There are many possible regularization terms that fit the above description. In order to keep the resulting optimization problem easily solvable, we will use $\|\theta\|^2/2$ as the penalty. Specifically, we will minimize

$$J_{n,\lambda}(\theta) = \frac{\lambda}{2}\|\theta\|^2 + R_n(\theta) = \frac{\lambda}{2}\|\theta\|^2 + \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot x^{(t)})^2/2 \tag{13}$$

where the *regularization parameter* $\lambda \geq 0$ quantifies the trade-of between keeping the parameters small – minimizing the squared norm $\|\theta\|^2/2$ – and fitting to the training data – minimizing the empirical risk $R_n(\theta)$. The use of this modified objective leads to *Ridge regression*.

While important, the regularization term introduces only small changes to the two estimation algorithms. For example, in the stochastic gradient descent algorithm, in each step, we will now move in the reverse direction of the gradient.

$$\nabla_\theta \left\{ \frac{\lambda}{2}\|\theta\|^2 + (y^{(t)} - \theta \cdot x^{(t)})^2/2 \right\}_{|\theta=\theta^{(k)}} = \lambda\theta^{(k)} - (y^{(t)} - \theta^{(k)} \cdot x^{(t)})x^{(t)} \tag{14}$$

As a result, the algorithm can be rewritten as

$$\begin{aligned} &\text{set } \theta^{(0)} = 0 \\ &\quad \text{randomly select } t \in \{1, \ldots, n\} \\ &\quad\quad \theta^{(k+1)} = (1 - \lambda\eta_k)\theta^{(k)} + \eta_k(y^{(t)} - \theta \cdot x^{(t)})x^{(t)} \end{aligned} \tag{15}$$

As you might expect, there's now a new factor $(1 - \lambda\eta_k)$ multiplying the current parameters $\theta^{(k)}$, shrinking them towards zero during each update.

When solving for the parameters directly, the regularization term only modifies the $d \times d$ matrix $A = \lambda I + (1/n)X^T X$, where I is the identity matrix. The resulting matrix is *always* invertible so long as $\lambda > 0$. The cost of inverting it remains the same, however.

2.1 The Effect of Regularization

The regularization term shifts emphasis away from the training data. As a result, we should expect that larger values of λ will have a negative impact on the training error. Specifically, let $\hat{\theta} = \hat{\theta}(\lambda)$ denote the parameters that we would find by minimizing the regularized objective $J_{n,\lambda}(\theta)$. We view $\hat{\theta}(\lambda)$ here as a function of λ. We claim then that $R_n(\hat{\theta}(\lambda))$, i.e., mean squared training error, increases as λ increases. If the training error increases, where's the benefit? Larger values of λ actually often lead to lower generalization error as we are no longer easily swayed by noisy data. Put another way, it becomes harder to over-fit to the training data. This benefit accrues for a while as λ increases, then turns to hurt us. Biasing the parameters towards zero too strongly, even when the data tells us otherwise, will eventually hurt generalization performance. As a result, you will see a typical U-shaped curve in terms of how the generalization error depends on the regularization parameter λ.

→ part of validation data

Learning Objective

You need to know:

1. What is linear regression and how to learn a linear regression model?
2. What is ridge regression and how is it different from linear regression?
3. What is regularization and why do we need to do regularization?

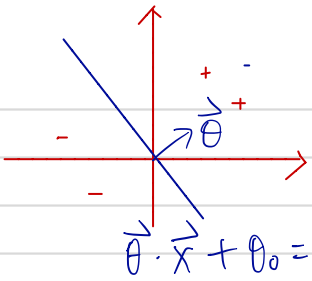
W2 Recap

Problem
→ No Linear Classifier

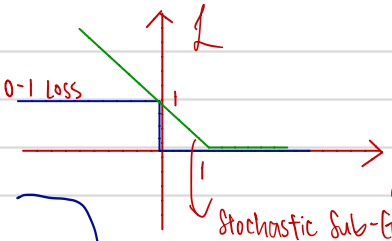
Soln (Stochastic Gradient Descent - Random Sampling)

Linear Classificatn : Functn

$f: R^n \to \{-1, +1\}$



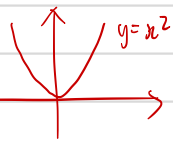
$\vec{\theta} \cdot \vec{x} + \theta_0 = 0$



Stochastic Sub-Gradient Descent.

Mtd 1
Analytical Approach

$\frac{dy}{dx} = 0$



$2(\vec{\theta} \cdot \vec{x} + \theta_0) = 0$ (Any diff acceptable soln)
(Non-unique soln)

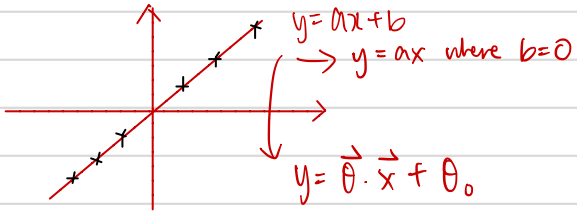
Mtd 2
Numerical Approach

Is it still same line in diagram?

W3

Linear Regression

Functn $f: R^n \to R^m$
(lower dimen)



$(\vec{x}^{(1)}, y^{(1)})$
$(\vec{x}^{(2)}, y^{(2)})$
$(\vec{x}^{(n)}, y^{(n)})$

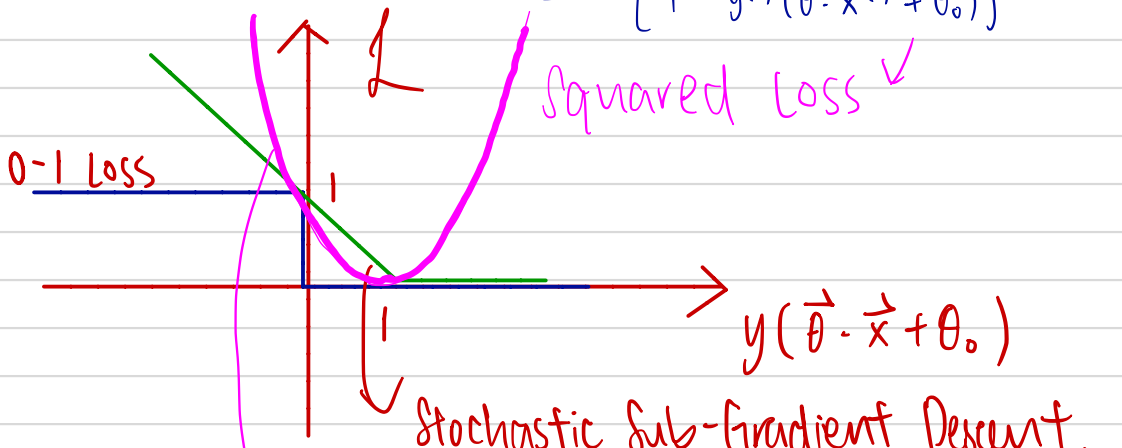
Least Squares Error :

$$\sum_{i=1}^{n} [y^{(i)} - (\vec{\theta} \cdot \vec{x}^{(i)} + \theta_0)]^2 \cdot y^{(i)2}$$

$y^{(i)}$: Expected y; $(\vec{\theta} \cdot \vec{x}^{(i)} + \theta_0)$: predicted y.

$$= [1 - y^{(i)}(\vec{\theta} \cdot \vec{x}^{(i)} + \theta_0)]^2$$

Squared Loss



Stochastic Sub-Gradient Descent.

$$\frac{1}{n} \sum_{i=1}^{n} [y^{(i)} - (\vec{\theta} \cdot \vec{x}^{(i)})]^2 / 2$$

$$L = \frac{1}{n}\sum_{i=1}^{n}\left[y^{(i)} - (\vec{\theta}\cdot\vec{x}^{(i)})\right]^2/2 + \frac{\lambda}{2}\|\vec{\theta}\|^2$$

Purpose: Minimise this loss as well too
↳ If $\frac{\lambda}{2}\|\vec{\theta}\|^2 = 0$ for regularizatn
Solve overfitting issue.

$$\frac{\partial L}{\partial \theta} = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - \vec{\theta}\cdot\vec{x}^{(i)})\cdot(-\vec{x}^{(i)})\ \frac{2}{2} + \lambda\vec{\theta}$$

(scalar, scalar, vector)

$$0 = -\frac{1}{n}\sum_{i=1}^{n} y^{(i)}\vec{x}^{(i)} + \frac{1}{n}\sum_{i=1}^{n}(\vec{\theta}\cdot\vec{x}^{(i)})\vec{x}^{(i)} + \lambda\vec{\theta}$$

(Given; unknown; matrix product)

$$= -\frac{1}{n}\sum_{i=1}^{n} y^{(i)}\vec{x}^{(i)} + \frac{1}{n}\sum_{i=1}^{n}(\vec{x}^{(i)}\vec{x}^{T(i)})\vec{\theta} + \lambda\vec{\theta}$$

$\underbrace{\quad}_{=\vec{b}}$ $\underbrace{\quad}_{=A}$

$$= -b + (A+\lambda I)\vec{\theta} \rightarrow \vec{\theta} =$$

$$\vec{\theta} = A^{-1}b$$

$(\vec{\theta}\cdot\vec{x}^{(i)})\vec{x}^{(i)}$ (dx1)
$= \vec{x}^{(i)}(\vec{\theta}\cdot\vec{x}^{(i)})$
$= \vec{x}^{(i)}(\vec{x}^{(i)}\cdot\vec{\theta})$
$= \vec{x}^{(i)}\vec{x}^{(i)T}\vec{\theta}$ (1xd)

$u\cdot v = u^T v$ (d, d)

Numerical Approach

↳ >1 soln if A is non-invertible (non-unique)
(1 soln vs ∞ soln)

Prove positive definite to invert ⟶ $v^T(XX^T + \lambda I)v > 0$

$v^T XX^T v + \lambda v^T v > 0$

Check ≥ 0 ; > 0

- Error w training set ↓ as set progress → use to explain noise in traing set.
- Error w test set may ↑ as set progress → Add Development / Validatn set (Held out in Data set & not used in traing set)

Stop here to prevent overfitting set.

D^3 operatn (expensive)

$(AI)\ A^{-1}$ → use Gaussian Elimination

→ Hence use Numerical Approach.

Stochastic Gradient Descent

① $\vec{\theta}^{(0)} = \vec{0}$

② Randomly pick $t \in \{0, \ldots, n\}$ (0-1 Loss)

$\vec{\theta}^{(k)} \leftarrow \vec{\theta}^{(k-1)} + \eta_k(y^{(t)} - \vec{\theta}^{(k-1)}\cdot\vec{x}^{(t)})\vec{x}^{(t)}$ $\quad \vec{\theta} \leftarrow \vec{\theta} - \alpha\vec{\nabla}$ (gradient)

③ Repeat ②

Make use of $L = [y^{(i)} - (\vec{\theta}\cdot\vec{x}^{(i)})]^2/2 + \frac{\lambda}{2}\|\theta\|^2$

$\frac{\partial L}{\partial \vec{\theta}} = (y^{(i)} - \vec{\theta}\cdot\vec{x}^{(i)})(-\vec{x}^{(i)})$

Variance Bias Decompositn

→ Collect many high quality data (Estimatn Error)
→ Improve Data
→ Improve Model. (Structural Error)

$(\hat{y} - y)^2$ → True y.
↳ Linear Regressor

$\rightarrow \text{Var}(\hat{y} - y) = E[(\hat{y}-y)^2] - E[\hat{y}-y]^2$

If all points shift left

$\rightarrow \text{Var}(\hat{y}) = E[(\hat{y}-y)^2] - E[\hat{y}-y]^2$

MSE $= \text{Var}(\hat{y}) + (E[\hat{y}-y])^2$ → Variance Bias Decompositn

$\text{Var}(X) = E[X^2] - E[X]^2$

Clustering
- k-Means
- LVQ

import Statistical Clustering.k-Means .*;

- View Java code
- View Python code

public void k-Means: Step-By-Step Example

{

As a simple illustration of a k-means algorithm, consider the following data set consisting of the scores of two variables on each of seven individuals:

Subject	A	B
1	1.0	1.0
2	1.5	2.0
3	3.0	4.0
4	5.0	7.0
5	3.5	5.0
6	4.5	5.0
7	3.5	4.5

This data set is to be grouped into two clusters. As a first step in finding a sensible initial partition, let the A & B values of the two individuals furthest apart (using the Euclidean distance measure), define the initial cluster means, giving:

	Individual	Mean Vector (centroid)
Group 1	1	(1.0, 1.0)
Group 2	4	(5.0, 7.0)

The remaining individuals are now examined in sequence and allocated to the cluster to which they are closest, in terms of Euclidean distance to the cluster mean. The mean vector is recalculated each time a new member is added. This leads to the following series of steps:

	Cluster 1		Cluster 2	
Step	Individual	Mean Vector (centroid)	Individual	Mean Vector (centroid)
1	1	(1.0, 1.0)	4	(5.0, 7.0)
2	1, 2	(1.2, 1.5)	4	(5.0, 7.0)
3	1, 2, 3	(1.8, 2.3)	4	(5.0, 7.0)
4	1, 2, 3	(1.8, 2.3)	4, 5	(4.2, 6.0)
5	1, 2, 3	(1.8, 2.3)	4, 5, 6	(4.3, 5.7)
6	1, 2, 3	(1.8, 2.3)	4, 5, 6, 7	(4.1, 5.4)

Now the initial partition has changed, and the two clusters at this stage having the following characteristics:

	Individual	Mean Vector (centroid)
Cluster 1	1, 2, 3	(1.8, 2.3)
Cluster 2	4, 5, 6, 7	(4.1, 5.4)

But we cannot yet be sure that each individual has been assigned to the right cluster. So, we compare each individual's distance to its own cluster mean and to that of the opposite cluster. And we find:

Individual	Distance to mean (centroid) of Cluster 1	Distance to mean (centroid) of Cluster 2
1	1.5	5.4
2	0.4	4.3
3	2.1	1.8
4	5.7	1.8
5	3.2	0.7
6	3.8	0.6
7	2.8	1.1

Only individual 3 is nearer to the mean of the opposite cluster (Cluster 2) than its own (Cluster 1). In other words, each individual's distance to its own cluster mean should be smaller that the distance to the other cluster's mean (which is not the case with individual 3). Thus, individual 3 is relocated to Cluster 2 resulting in the new partition:

	Individual	Mean Vector (centroid)
Cluster 1	1, 2	(1.3, 1.5)
Cluster 2	3, 4, 5, 6, 7	(3.9, 5.1)

The iterative relocation would now continue from this new partition until no more relocations occur. However, in this example each individual is now nearer its own cluster mean than that of the other cluster and the iteration stops, choosing the latest partitioning as the final cluster solution.

Also, it is possible that the k-means algorithm won't find a final solution. In this case it would be a good idea to consider stopping the algorithm after a pre-chosen maximum of iterations.

}

News Links:

2018 (0)
February (0)
January (0)
2017 (0)
2016 (0)
2015 (0)
2014 (0)
2013 (32)
2012 (242)
2011 (217)
2010 (185)
2009 (20)

Search News Links:
Search News

}