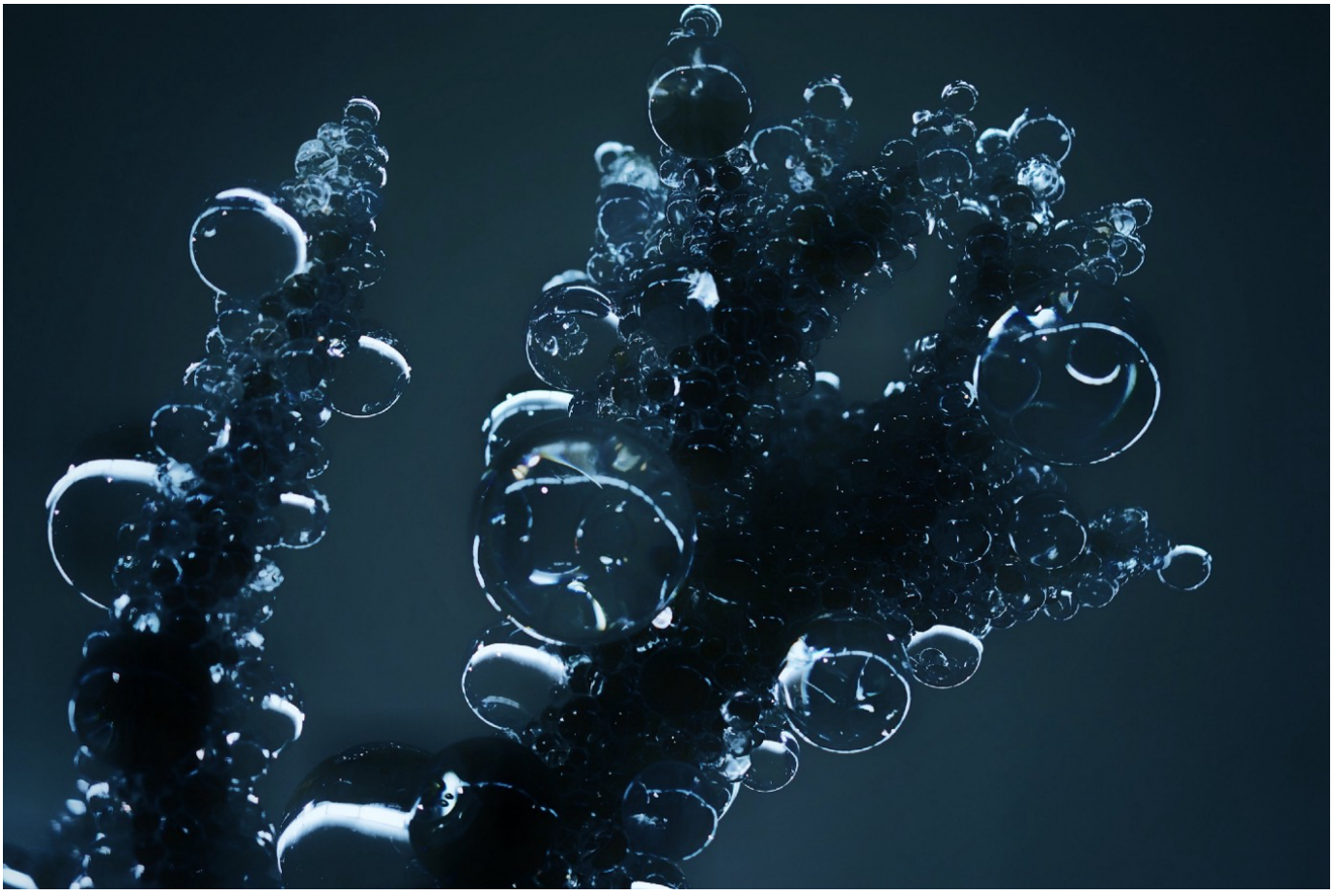

Photo by Fraser Hansen on Unsplash

K-Prototype in Clustering Mixed attributes

Chamani Shiranthika

Feb 4, 2018 · 6 min read

The world is all about data. Wherever our eyes go in, we see data performing marvelous performances in each and every second. Data appears in the form of numerical and also in categorical format. The concept of Machine Learning deals with the semi-automated extraction of knowledge from data. It basically starts with a question that might be answerable using data.

The two main categories of Machine Learning are Supervised Learning and Unsupervised Learning. Here I focus on the unsupervised learning concepts where we extract some structure from data and doing the analysis thereby. Clustering or Cluster Analysis is an Unsupervised Learning technique which bears the task of grouping a set of objects considering their similarity. Objects in the same group or cluster are more similar to each other than to those in other groups or clusters.

Some of popular numerical data clustering methods and algorithms are as follows.

· Representative based clustering-> K Means clustering algorithm

· Hierarchical clustering ->Agglomerative clustering

· Density-based clustering->DBSCAN

· Spectral and graph clustering->Spectral clustering

· Gaussian Mixtures

Some of popular categorical data clustering methods and algorithms are as follows.

· K-modes algorithm (Gower's similarity coefficient)

· Squeezer

· LIMBO

· GAClast

· Cobweb algorithm

· STIRR , ROCK, CLICK

· CACTUS,COOLCAT, CLOPE

In the popular K-modes algorithm distance is measured by the number of common categorical attributes shared by the two data points.

So what do you think about a data set considering both numerical and categorical values? Will the above methods do the job? How does the machine get to know about the grouping of "mixed" attributes? Here comes the K-Prototype. That's the simple combination of K-Means and K-Modes in clustering mixed attributes.

Here are the simple steps of the K-prototype algorithm

1. Select k initial prototypes from the dataset X. It must be one for each cluster.

2. Allocate each object in X to a cluster whose prototype is the nearest to it. This allocation is done with considering the **dissimilarity measure** which is described next.

3. After all, objects have been allocated to a cluster, retest the similarity of objects against the current prototypes. If you find that an object is found such that its nearest to another cluster prototype, update the prototypes of both the clusters.

4. Repeat 3, until no object changes its cluster after fully testing X.

How the objects are assigned into the clusters? It's by considering the dissimilarity measure.

Dissimilarity measure

Let X={x1, x2,…, xn} denotes a set of n objects

Xi = [xi1, xi2,…, xim] each object is represented by m attribute values

First objects in X are divided into k disjoint clusters.

So what is this **prototype**? It's the center of the cluster.

$$E_l = \sum^{n} y_{il} \sum^{m_r} (x_{ij}^r - q_{lj}^r)^2 + \gamma_l \sum^{n} y_{il} \sum^{m_c} \delta(x_{ij}^c, q_{lj}^c)$$

$$E_l = E_l^r + E_l^c$$

1–1st cluster

r — Numerical attributes

c — Categorical attributes

yi1 — Object Xi belongs to cluster 1

xi1 — jth attribute

qij — jth attribute of the prototype in cluster 1

mr — No.of Numerical attributes

mc — No.of categorical attributes

m = mr + mc

r1 — Weight of the categorical attribute in cluster 1

If r1 is small, it indicates that the clustering is dominated by numerical attributes

If r1 is large, it indicates that the clustering is dominated by categorical attributes

E1= Minimum sum of the difference of all the elements and the prototypes in cluster 1

E1r= Minimum sum of the difference of the numerical attributes

of all the elements and the prototypes in cluster 1

E1c= Minimum sum of the difference of the categorical attributes of all the elements and the prototypes in cluster 1

Implementation of the algorithm

Here are the simple 5 steps in implementing the K-Prototype algorithm

1. Read parameter

2. Initial prototypes

3. Initial allocation

4. Reallocation

5. Program output

Let's dig bit into detail.

1. Read parameter

Here read various parameters of the given database. Such as

*Total record number n

*Maximum cluster number k

*No.of categories for each categorical attributes

*Name and type of each attribute

*Sequence of attributes in the database

2. Initial prototypes selection

Here select k objects as the initial prototypes for k clusters at random.

For example, if X[i] denotes object i

X [i, j] value of jth attribute for object i

Prototype_N[i] — Is the numerical element of prototype for cluster i

Prototype_C[i] — Is the categorical element of prototype for cluster i

3. Initial allocation

Each object of the data set x is assigned to a cluster which has the minimum difference with its prototype with the previous method, dissimilarity measure. After cluster prototype is updated accordingly after each assignment.

Some functions available in the algorithm are as follows.

Distance () — Square Euclidean distance function for the numeric attributes

Sigma () — function with the minimum difference between the categorical attribute and its prototype

Clustership [] — Cluster membership of objects

Clustercount [] — No.of objects in cluster[i]

SumInCluster[i] — Sums up numeric attributes of objects in cluster [i] and used to update values of numeric attributes of the cluster prototypes

FrequencyInCluster[i] — Records frequencies of different values of categorical attributes

HighestFreq () — Is used to obtain which categorical value has the highest frequency and is used to update the value of categorical attributes of prototypes

4. Reallocation

Here the prototypes for the previous and current clusters of the objects must be updated. When we running the algorithm console shows the variable “moves” which records the number of objects which have changed clusters in the process. If the moves =0, it indicates that the algorithm has obtained the best result.

Simple python implementation of the K prototype clustering is as follows.

Here I have used a simple data set which has been extracted from Facebook using graph API. Details regarding the implementations carried out there will be discussed separately. Here the following is a snapshot of the data set which contains both categorical and numerical attributes. Comma separated values include the publisher name, category score, category type, and place name separately.

240,Ransika Fernando,0.59375,plant,No Data
240,Ransika Fernando,0.04296875,outdoor_,No Data
240,Ransika Fernando,0.26953125,outdoor_road,No Data
241,Sachini Jagodaarachchi,0.98046875,outdoor_mountain,Manigala Mountain
242,Chathuri Senanayake,0.96484375,outdoor_mountain,Adara Kanda
242,Chathuri Senanayake,0.1953125,building_,No Data
242,Chathuri Senanayake,0.00390625,outdoor_,No Data
242,Chathuri Senanayake,0.23046875,building_,Kuwait
242,Chathuri Senanayake,0.2578125,building_street,Kuwait
242,Chathuri Senanayake,0.015625,outdoor_,Kuwait
243,Nilantha Premakumara,0.9453125,sky_sun,No Data
243,Nilantha Premakumara,0.75,outdoor_mountain,No Data
244,Chathuri Senanayake,0.00390625,outdoor_,Trincomalee
244,Chathuri Senanayake,0.6328125,outdoor_oceanbeach,Trincomalee
245,Surangani Bandara,0.7734375,plant_tree,No Data
246,Hasitha Lakmal,0.4140625,people_many,No Data
246,Hasitha Lakmal,0.0078125,outdoor_,No Data
247,Pradeep Kalansooriya,0.40234375,building_,No Data
247,Pradeep Kalansooriya,0.0078125,outdoor_,No Data
248,Dilini Wijesinghe,0.07421875,outdoor_,Victoria Dam
248,Dilini Wijesinghe,0.0078125,others_,Victoria Dam
249,Chiranthi Vinghghani,0.015625,outdoor_,No Data
249,Chiranthi Vinghghani,0.6484375,outdoor_waterside,No Data
250,Janindu Praneeth Weerawarnakula,0.671875,outdoor_oceanbeach,Galle Fort
251,Chathurangi Shyalika,0.00390625,outdoor_,No Data
252,Chathurangi Shyalika,0.9296875,trans_trainstation,No Data
253,Surangani Bandara,0.625,outdoor_field,No Data
253,Surangani Bandara,0.01171875,outdoor_,No Data

```
254,Surangani Bandara,0.99609375,sky_object,No Data
255,Chathurangi Shyalika,0.00390625,outdoor_,No
Data
256,Chathurangi
Shyalika,0.33984375,outdoor_field,No Data
```

Below given is the categorization of the above data set by using the k prototype algorithm.

```
#!/usr/bin/env python
import numpy as np
from kmodes.kprototypes import KPrototypes
import matplotlib.pyplot as plt
from matplotlib import style
style.use("ggplot")
colors = ['b', 'orange', 'g', 'r', 'c', 'm', 'y',
'k', 'Brown', 'ForestGreen']

#Data points with their publisher name,category
score, category name, place name
syms = np.genfromtxt('travel.csv', dtype=str,
delimiter=',')[:, 1]
X = np.genfromtxt('travel.csv', dtype=object,
delimiter=',')[:, 2:]
X[:, 0] = X[:, 0].astype(float)

kproto = KPrototypes(n_clusters=15, init='Cao',
verbose=2)
clusters = kproto.fit_predict(X, categorical=[1,
2])

# Print cluster centroids of the trained model.
print(kproto.cluster_centroids_)
# Print training statistics
print(kproto.cost_)
print(kproto.n_iter_)

for s, c in zip(syms, clusters):
    print("Result: {}, cluster:{}".format(s, c))
```

```
# Plot the results
for i in set(kproto.labels_):
    index = kproto.labels_ == i
    plt.plot(X[index, 0], X[index, 1], 'o')
    plt.suptitle('Data points categorized with
category score', fontsize=18)
    plt.xlabel('Category Score', fontsize=16)
    plt.ylabel('Category Type', fontsize=16)
plt.show()

# Clustered result
fig1, ax3 = plt.subplots()
scatter = ax3.scatter(syms, clusters, c=clusters,
s=50)
ax3.set_xlabel('Data points')
ax3.set_ylabel('Cluster')
plt.colorbar(scatter)
ax3.set_title('Data points classifed according to
known centers')
plt.show()

result = zip(syms, kproto.labels_)
sortedR = sorted(result, key=lambda x: x[1])
print(sortedR)
```

Hope you got a brief knowledge on clustering of mixed attributes.

Machine Learning Clustering