Applying k-means clustering to the Fruits 360 dataset (k=10) and analyzing the clusters and common properties found for each cluster with dataset

Sai Sharanyya Chanda
AITS
9 August 2019
Khammam, India
sharanyyachanda@gmail.com

Uttaran Sarkar
AITS
9 August 2019
Patiala, India
sarkaruttaran@gmail.com

***Abstract*— In this paper we introduce a new, high-quality, dataset of images containing fruits. We also present the results of some numerical experiment for training with k-means clustering algorithm to analyze and find common properties in the dataset. We discuss the reason why we chose to use fruits dataset in this project by proposing a few applications that could use such classifier.**
It's the Assignment-3 given to us as Machine Learning Interns. We have done the Exploratory data Analysis, Preprocessing, modelled with k-means clustering algorithm and have plotted the common properties which are found in a given dataset.

Keywords— K-means clustering; Principal Component Analysis(PCA); Object recognition; Fruits 360 dataset; Image processing

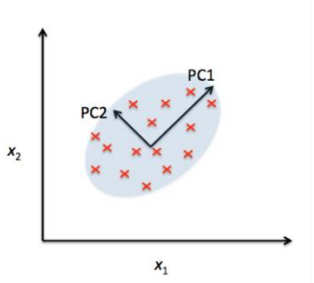
I. INTRODUCTION

Clustering is a technique to categorize the data into groups. Distance metrics plays a very important role in the clustering process. There are number of algorithms which are available for clustering. In general, K-means is a heuristic algorithm that partitions a data set into K clusters by minimizing the sum of squared distance in each cluster. The algorithm consists of three main steps: a) initialization by setting centre points (or initial centroids) with a given K, b) Dividing all data points into K clusters based on K current centroids, and c) updating K centroids based on newly formed clusters. It is clear that the algorithm always converges after several iterations of repeating steps.

In the first part of the paper, we describe the PCA algorithm, the second part details the K-means algorithm, the third presents the scikit library as well as a discussion on clustering. The last part describes the experimental results and conclusions obtained on experimenting with the Fruits 360 data set.

II. PRINCIPAL COMPONENT ANALYSIS

Principal Component Analysis (PCA) is a technique used in order to reduce the dimensionality of a dataset while preserving as much information as possible. Data is projected in a lower dimensional space; in particular we need to find a projection that minimizes the squared error in reconstructing the original data.



There are 3 different technique in order to apply PCA:

1. Sequential
2. Sample Covariance Matrix
3. Singular Value Decomposition (SVD)

We will be dealing with the Sample Covariance Matrix technique:

- The first thing to do is to standardize the data, so for each sample we need to subtract the mean of the full dataset and then divide it by the variance, so as having an unitary variance for each instance. This last process is not completely necessary but it is useful to let the CPU work less.

  $$Z=X-\mu\sigma 2 Z=X-\mu\sigma 2$$

- Then we need to compute Covariance Matrix, given data { x1,x2,...,xnx1,x2,...,xn } with n number of samples, covariance matrix is obtained by:

  $$\Sigma=1n\sum i=1n(xi-x^{-})(x-x^{-})T\Sigma=1n\sum i=1n(xi-x^{-})(x-x^{-})T$$

  where

  $$x^{-}=1n\sum i=inxix^{-}=1n\sum i=inxi$$

  Or simply by multiplying the standardized matrix Z with its transpose:

  $$COV(X)=ZZTCOV(X)=ZZT$$

- Principal Components will be the eigenvectors of the Covariance Matrix sorted in order of importance by the respective eigenvalues.
  Larger eigenvalues ⇒ more important eigenvectors.
  They represent the most of the useful information on the entire dataset in a single vector.

III. K-MEANS CLUSTERING

K-means clustering is one of the simplest and popular unsupervised machine learning algorithms.
Typically, unsupervised algorithms make inferences from datasets using only input vectors without referring to known, or labelled, outcomes.
Andrey Bu, who has more than 5 years of machine learning experience and currently teaches people his skills, says that "the objective of K-means is simple: group similar data points together and discover underlying patterns. To achieve this objective, K-means looks for a fixed number (*k*) of clusters in a dataset."
A cluster refers to a collection of data points aggregated together because of certain similarities.
You'll define a target number *k*, which refers to the number of centroids you need in the dataset. A centroid is the imaginary or real location representing the center of the cluster.
Every data point is allocated to each of the clusters through reducing the in-cluster sum of squares.
In other words, the K-means algorithm identifies *k* number of centroids, and then allocates every data point to the nearest cluster, while keeping the centroids as small as possible. The *'means'* in the K-means refers to averaging of the data; that is, finding the centroid.

A. How the K-Means algorithm works

To process the learning data, the K-means algorithm in data mining starts with a first group of randomly selected centroids, which are used as the beginning points for every cluster, and then performs iterative (repetitive) calculations to optimize the positions of the centroids
It halts creating and optimizing clusters when either:

- The centroids have stabilized — there is no change in their values because the clustering has been successful.
- The defined number of iterations has been achieved.

B. K-Means algorithm example problem

Let's see the steps on how the K-means machine learning algorithm works using the Python programming language.

We'll use the Scikit-learn library and some random data to illustrate a K-means clustering simple explanation.

Step 1: Import libraries

As you can see from the above code, we'll import the following libraries in our project:

- Pandas for reading and writing spreadsheets
- Numpy for carrying out efficient computations
- Matplotlib for visualization of data

Step 2: Generate random data

Generating random data on which K-Means is applied.

Step 3: Use Scikit-Learn

We'll use some of the available functions in the Scikit-learn library to process the randomly generated data.

```
from sklearn.cluster import KMeans
Kmean=KMeans(n_clusters=2)
Kmean.fit(X)
```

Step 4: Finding the centroid

Here is the code for finding the center of the clusters:

```
Kmean.cluster_centers_
```

Step 5: Testing the algorithm

Here is the code for getting the labels property of the K-means clustering example dataset; that is, how the data points are categorized into the two clusters.

```
Kmean.labels_
```

K-means clustering is an extensively used technique for data cluster analysis.
It is easy to understand, especially if you accelerate your learning using a K-means clustering tutorial. Furthermore, it delivers training results quickly.
However, its performance is usually not as competitive as those of the other sophisticated clustering techniques because slight variations in the data could lead to high variance.
Furthermore, clusters are assumed to be spherical and evenly sized, something which may reduce the accuracy of the K-means clustering Python results.

IV. SCIKIT-LEARN LIBRARY

If you are a Python programmer or you are looking for a robust library you can use to bring machine learning into a production system then a library that you will want to seriously consider is scikit-learn.
In this part, you will get an overview of the scikit-learn library and useful references of where you can learn more. scikit-learn was initially developed by David Cournapeau as a Google summer of code project in 2007. Later Matthieu

Brucher joined the project and started to use it as a part of his thesis work. In 2010 INRIA got involved and the first public release (v0.1 beta) was published in late January 2010.
The project now has more than 30 active contributors and has had paid sponsorship from INRIA, Google, Tinyclues and the Python Software Foundation.

A. *What is scikit-learn*

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python.
It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.
The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:
NumPy: Base n-dimensional array package
SciPy: Fundamental library for scientific computing
Matplotlib: Comprehensive 2D/3D plotting
IPython: Enhanced interactive console
Sympy: Symbolic mathematics
Pandas: Data structures and analysis

Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.
The vision for the library is a level of robustness and support required for use in production systems. This means a deep focus on concerns such as easy of use, code quality, collaboration, documentation and performance.
Although the interface is Python, c-libraries are leverage for performance such as numpy for arrays and matrix operations, LAPACK, LibSVM and the careful use of cython.

B. *Features of scikit-learn*

Some popular groups of models provided by scikit-learn include:
Clustering: for grouping unlabeled data such as KMeans.
Cross Validation: for estimating the performance of supervised models on unseen data.
Datasets: for test datasets and for generating datasets with specific properties for investigating model behavior.
Dimensionality Reduction: for reducing the number of attributes in data for summarization, visualization and feature selection such as Principal component analysis.
Ensemble methods: for combining the predictions of multiple supervised models.
Feature extraction: for defining attributes in image and text data.
Feature selection: for identifying meaningful attributes from which to create supervised models.
Parameter Tuning: for getting the most out of supervised models.
Manifold Learning: For summarizing and depicting complex multi-dimensional data.
Supervised Models: a vast array not limited to generalized linear models, discriminate analysis, naïve-Bayes, lazy methods, neural networks, support vector machines and decision tree.

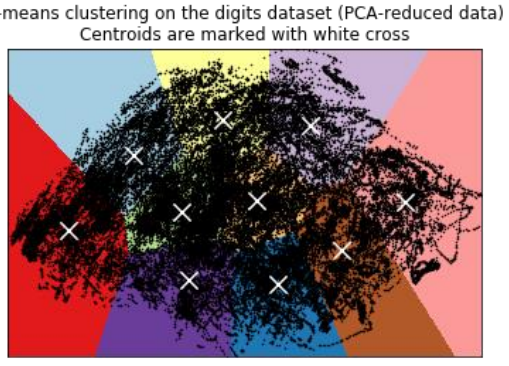
C. *CLUSTERING*

- Clustering of unlabeled data can be performed with the module **sklearn.cluster**.
- Each clustering algorithm comes in two variants: a class, that implements the fit method to learn the clusters on train data, and a function, that, given train data, returns an array of integer labels corresponding to the different clusters. For the class, the labels over the training data can be found in the labels_ attribute.

Input data:- One important thing to note is that the algorithms implemented in this module can take different kinds of matrix as input. All the methods accept standard data matrices of shape [n_samples, n_features]. These can be obtained from the classes in the sklearn.feature_extraction module. For Affinity Propagation, Spectral Clustering and DBSCAN one can also input similarity matrices of shape [n_samples, n_samples]. These can be obtained from the functions in the sklearn.metrics.pairwise module.
Non-flat geometry clustering is useful when the clusters have a specific shape, i.e. a non-flat manifold, and the standard Euclidean distance is not the right metric. This case arises in the two top rows of the figure above.
Gaussian mixture models, useful for clustering, are described in another chapter of the documentation dedicated to mixture models. KMeans can be seen as a special case of Gaussian mixture model with equal covariance per component.

V. Experimental Results

We have applied K-Means clustering algorithm on the training set. Accordingly, the data was neatly grouped into 10 clusters on the basis of common properties found for each cluster. The K-Means Clustering Algorithm giving reasonably good performance for the given dataset.



CONCLUSION

K-means is a typical clustering algorithm and it is widely used for clustering large sets of data. Also we made some numerical experiments by using Scikit-learn library in

order to classify and cluster the same features of the images according to their content. From our point of view one of the main objectives for the future is to improve the accuracy of the k-means.

REFERENCES

[1] Sun Jigui, Liu Jie, Zhao Lianyu, "Clustering algorithms Research",Journal of Software ,Vol 19,No 1, pp.48-61,January 2008.

[2] Fahim A M,Salem A M,Torkey F A, "An efficient enhanced k-means clustering algorithm" Journal of Zhejiang University Science A, Vol.10, pp:1626-1633,July 2006.

[3] Jain, A.K., Murty, M.N., Flynn, P.J., 1999, "Data clustering: A review", ACM Comput. Surveys 31 (3), 264-323.

[4] J.M. Pena, J.A. Lozano, P. Larranaga, "An empirical comparison of four initialization methods for the K-Means algorithm", Pattern Recognition Letters 20 (1999) 1020-1040.

[5] Sergios Theodoridis and Konstantinos Koutroumbas, "Pattern Recognition", 4th edition, Academic Press Publications, Canada, 2009.