Author : Saurabh Kumar

Lets First UnderStand Clustering

What is Clustering?

Definition:

- **Clustering is Unsupervised Machine Learning Technique (Unsupervised – Unlabeled data) where we are trying grouping of data points.These groups are called Clusters.points that are in the same group should have similar properties and/or features, while points in different groups should have highly different properties and/or features.**
- **The aim is to segregate groups with similar traits and assign them into clusters.**

Simple Use Case:

- Suppose you are the head of a retail store and want to understand preferences of your customers to scale up your business. It is possible for you to look at details of each customer and devise a unique business strategy for each one of them. However, what if you have a huge customer base, and it is impossible to look at each individual customer and analyze their behavior. In this case, clustering comes in handy. Clustering allows you to segregate a group of data points into clusters, so that you can then devise a business strategy for each cluster.

Why we use Clustering? or How its helps in Data Science???

- Clustering is used in many fields **Feature engineering + pattern recognition**
- We can use clustering to reveal the structure of the data, explore the data, and gain insight into data patterns.
- Statistical analysis.

Real-time Use cases:

- Since the training dataset is labeled to large processing, detecting identity trends of pattern in large data. This can be very time-consuming and expensive. Clustering helps with this.
- The personal data collected with shopping Scanner, Interest, actions and a limited number of interactions, can be analyzed with clustering, providing new Customer Insights and Trends. Examples of this are the Retail industry, marketing strategies, web analytics, and Social Media.
- Other types of applications based on clustering algorithms are Genetics, chemistry, recommender systems, mathematical and statistical analysis, and many other important operations of artificially.

Possible Applications

- Marketing: Finding groups of customers with similar behavior given a large database of customer data containing their properties and past buying records.
- Biology: Classification of plants and animals given their features.
- Insurance: Identifying groups of motor insurance policy holders with a high average claim cost; identifying frauds.
- City planning: Identifying groups of houses according to their house type, value and geographical location.
- Earthquake studies: Clustering observed earthquake epicentres to identify dangerous zones.
- WWW: document classification; Clustering weblog data to discover groups of similar access patterns.

Clustering Visualization

Types of Clustering

Based on the area of overlap

- **Hard Clustering**
  - In hard clustering, each data point belongs to a cluster. Each data point belongs to a cluster completely or not.
- **Soft Clustering**
  - Instead of assigning a cluster, the data point gives a score to the clusters. The score indicates the probability of belonging or not belonging to a cluster. With probabilities, the data point can belong to more than one cluster. The score can be based on any kind of metric or probability. Fuzzy clustering is a form of soft clustering in which each data point can belong to more than one cluster.

Various Clustering Algorithms:

Lets Discuss different type of method used for clustering

Clustering Algorithm Based on Distribution

The basic idea is that data points that belong to the same distribution are in the same cluster. Gaussian Mixture Models (GMM) are often used for data that is known to contain multivariate Gaussian distributions. Consider the following diagram. It is clear that there are three distributions with different means and different spreads. The goal of the algorithm is to assign each data point to a distribution. The intuition of the partial clustering which is generated from the existing distribution is that the data points that are far from the center of a distribution are less likely to belong to the cluster than those that are close to the center. Clustering based on distribution is a powerful method that gives the flexibility to define clusters of any shape. Although, this method is susceptible to overfitting and has difficulty when a certain threshold is not maintained in the distribution. Therefore, the distribution chosen should be familiar to the underlying data or at least a distribution that is considered to belong to the same cluster.

Analysis

Algorithms:	**DBCLASD**	**GMM**
Complexity:	Out Puts(n)	Out (*Nm)
Comparison	Middle	High

Advantages:

Data scientist is up to the probability of belonging, relatively high scalability by changing the distribution, number of clusters and so on, and supported by the well developed statistical science

Disadvantages:

The process not completely correct, involved in many parameters which have a strong influence on the clustering result and relatively high computational cost

Clustering Based on Density

- see dbscan.gif

Advantages:

Clustering in high efficiency and suitable for data with arbitrary shape

Disadvantages:

Reaching to a clustering result with the quality when the density of data space not even, a memory with big size needed when the data volume is very large, and clustering result highly sensitive to the parameters.

Analysis

Algorithms:	**DBCLASD**	**GMM**
Complexity:	Out*logOut	Out (*Nm)
Comparison	Middle	High

Hierarchical clustering can be based on agglomerative or divisive methods hierarchy of clusters. The agglomerative starts with the data points assigned to a cluster of their own. Then two nearest clusters are merged into the same cluster. In the end, the algorithm terminates when there is only a single cluster left.

Advantages:

Suitable for the data set with arbitrary shape and attribute of arbitrary type, the hierarchical relationship among clusters easily detected and relatively high scalability in general

Disadvantages:

Relatively high in time complexity in general, the number of clusters needed to be precise

Analysis

Algorithms:	BIRCH	CURE	ROCK	Chameleon
Complexity:	Out	Out (*N)	Out (*logOut)	Out (*))
Comparison	Low	Low	High	High

Clustering Algorithm Based on Fuzzy Theory

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters.

Advantages:

Clustering in high efficiency and suitable for data with arbitrary shape

Disadvantages:

Reaching to a clustering result with the quality when the density of data space not even, a memory with big size needed when the data volume is very large, and clustering result highly sensitive to the parameters.

Analysis

Algorithms:	DBCLASD	GMM
Complexity:	Out*logOut	Out (*Nm)
Comparison	Middle	High

Advantages:

Suitable for the data set with arbitrary shape and attribute of arbitrary type, the hierarchical relationship among clusters easily detected and relatively high scalability in general

Disadvantages:

Relatively high in time complexity in general, the number of clusters needed to be precise

Analysis

Algorithms:	BIRCH	CURE	ROCK	Chameleon
Complexity:	Out	Out (*N)	Out (*logOut)	Out (*))
Comparison	Low	Low	High	High

Clustering Algorithm Based on Fuzzy Theory

Fuzzy c-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters.

New Lets Jumps to Implementation Section

Famous Clustering Algorithms:

1. **K-means clustering algorithm**
2. **DBSCAN clustering algorithm**
3. **Gaussian Mixture Model algorithm**
4. **BIRCH algorithm**
5. **Affinity Propagation clustering algorithm**
6. **Mean Shift clustering algorithm**
7. **OPTICS algorithm**
8. **Agglomerative Hierarchy clustering algorithm**

K-means clustering algorithm

K-means clustering is the most commonly used clustering algorithm. It's a centroid-based algorithm and the simplest unsupervised learning algorithm.

Its goal is to try to minimize the variance of data points within a cluster. It also has more weight assigned to its unsupervised nature.

K-means is also used as a feature learning method, one way to use the output of the data points. That means it will take more time to classify data points if there are large amounts of them in the data set.

General Implementation

```
# Import libraries.
import numpy as np, pandas as pd, sklearn
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
# Load the data.
df = pd.read_csv('./data/Mall_Customers.csv')
# Relabel the column names for convenience.
df.columns = ['id', 'gender', 'age', 'income', 'score']
# Look at the first 5 rows of the data.
df.head(5)
# Get the shape of the data.
df.shape
# Get a description of the data.
df.describe()
# Get a count of the number of missing values in each column.
df.isnull().sum()
# Get a count of the number of unique values in each column.
df.nunique()
# Get a count of the number of customers by gender.
df['gender'].value_counts()
# Get a count of the number of customers by age.
df['age'].value_counts()
# Create a histogram of the customer ages.
df['age'].hist(bins=df['age'].nunique())
# Create a histogram of the customer incomes.
df['income'].hist(bins=df['income'].nunique())
# Create a histogram of the customer scores.
df['score'].hist(bins=df['score'].nunique())
```

Lets Implement K-means Clustering

```
# Import libraries.
import numpy as np, pandas as pd
import sklearn
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score
# Load the data.
df = pd.read_csv('./data/Mall_Customers.csv')
# Relabel the column names for convenience.
df.columns = ['id', 'gender', 'age', 'income', 'score']
# Create a copy of the dataset.
df_clean = df.copy()
# Create a new column for the gender dummy variable.
df_clean['gender_dummy'] = df_clean['gender'].replace({'Female': 0, 'Male': 1})
# Create a list of the feature column names.
features = ['age', 'income', 'score', 'gender_dummy']
# Import the StandardScaler.
from sklearn.preprocessing import StandardScaler
# Create a new instance of the StandardScaler.
scaler = StandardScaler()
# Create a new dataset for the scaled features.
df_scaled = pd.DataFrame(scaler.fit_transform(df_clean[features]), columns=features)
```

```
# Import the PCA class.
from sklearn.decomposition import PCA
# Create a new instance of the PCA class.
pca = PCA()
# Fit the PCA model.
pca.fit(df_scaled)
# Get the explained variance ratio.
pca.explained_variance_ratio_
# Create a new instance of the PCA class with 3 components.
pca = PCA(n_components=3)
# Fit the PCA model.
pca.fit(df_scaled)
# Transform the data.
df_pca = pd.DataFrame(pca.transform(df_scaled))
# Get the shape of the data.
df_pca.shape
# Create a new instance of the KMeans class.
kmeans = KMeans(n_clusters=5, random_state=1)
# Fit the KMeans model.
kmeans.fit(df_pca)
# Get the cluster labels.
kmeans.labels_
# Create a new column for the cluster labels.
df_clean['cluster'] = kmeans.labels_
# Create a new column for the cluster labels.
df_pca['cluster'] = kmeans.labels_
# Create a new column for the cluster labels.
df_scaled['cluster'] = kmeans.labels_
```

```
# Import the silhouette_score function.
from sklearn.metrics import silhouette_score
# Calculate the silhouette score.
silhouette_score(df_pca, kmeans.labels_)
```

```
# Import the KMeans class.
from sklearn.cluster import KMeans
```

```
sns.pairplot(numerical_cols, dpi=60)
plt.suptitle('Pairwise plots', fontsize=20, y=1.02);
for col in cat_cols:
    sns.boxplot(x=df[col], y=df['Survived'], palette='Paletteer_c')
    plt.title('Survival Rate by {}'.format(col), fontsize=20)
    plt.xticks(rotation=90, fontsize=12)
    plt.show()
```