**MINISTRY OF EDUCATION AND TRAINING**

**HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY AND EDUCATION**

# **FALCUTY FOR HIGH QUALITY TRAINING**

# **--------------------**



# **PROJECT REPORT**

**PROJECT 1**

**K-MEANS CLUSTERING**

**GROUP 1:**

**VŨ TUẤN ANH 17110004**

**HỒ VŨ MINH ĐỨC 17110022**

**INSTRUCTOR: LÊ VĂN VINH**

# **HO CHI MINH CITY – 2019**

**CONTENTS**

[**INTRODUCTION** 3](#_Toc27689960)

[**1.** **What is k-means clustering?** 3](#_Toc27689961)

[**2.** **How to perform k-means clustering** 3](#_Toc27689962)

[**3.** **How to determine the optimal value of k?** 5](#_Toc27689963)

[**K-MEANS CLUSTERING ILLUSTRATION ON MATLAB** 7](#_Toc27689964)

[**1.** **Code for K-means Clustering (kmeans.m) which is running on Matlab** 7](#_Toc27689965)

[**2.** **Figures of illustration of data before and after running** 8](#_Toc27689966)

[**PRECISION AND RECALL ON EXCEL** 10](#_Toc27689967)

[**1.** **How to calculate precision and recall** 10](#_Toc27689968)

[**2.** **Result of experience** 11](#_Toc27689969)

# **INTRODUCTION**

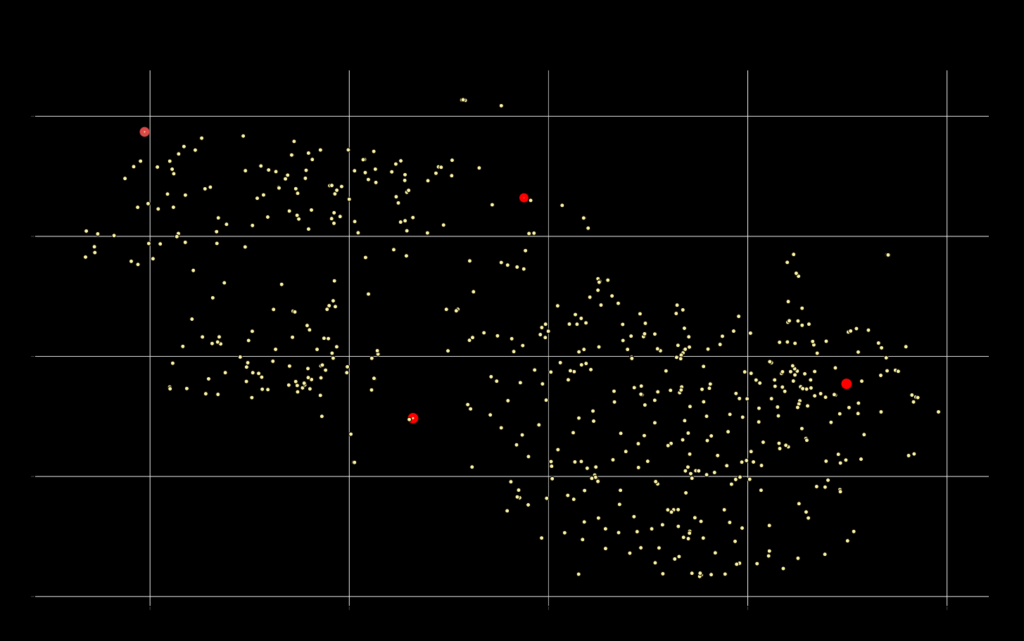
## **What is k-means clustering?**

K-means clustering algorithm is an unsupervised machine learning algorithm for determining which group a certain object really belongs to. What it means by “being unsupervised” is that there are no prescribed labels in the data denoting its structure. The main idea is to assign each observation into the cluster with the nearest mean (centroid), serving as a prototype of the cluster.

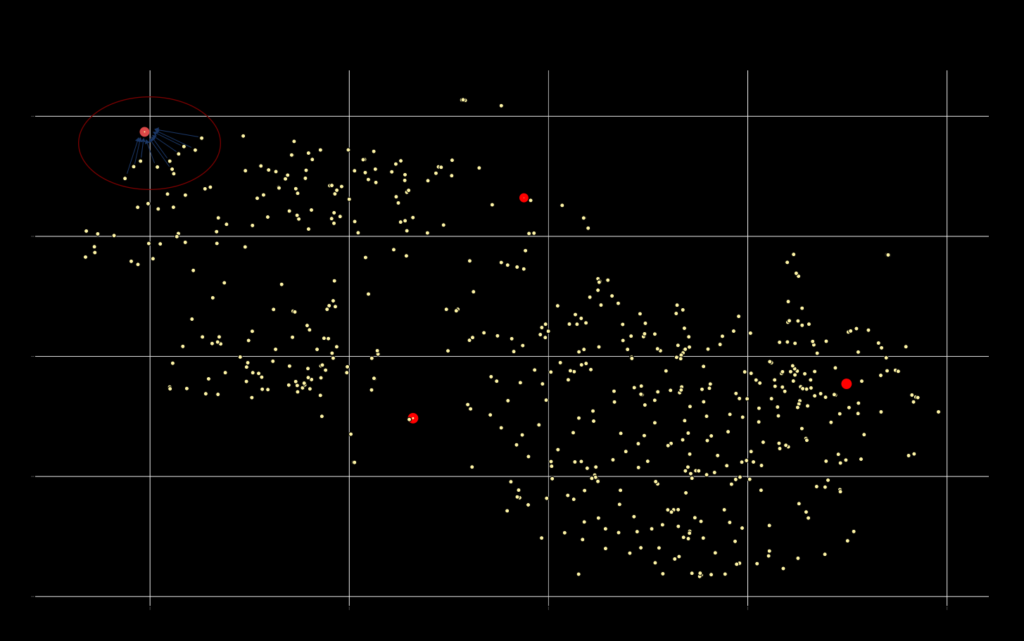
## **How to perform k-means clustering**

Here are five simple steps for the k-means clustering algorithm and an example for illustration:

***Step 1:*** Visualize n data points and decide the number of clusters (k). Choose k random points on the graph as the centroids of each cluster. For this example, we would like to divide the data into 4 clusters, so we pick 4 random centroids.



***Step 2:*** Calculate the Euclidean distance between each data point and chosen clusters’ centroids. A point is considered to be in a particular cluster if it is closer to that cluster's centroid than any other ones.



***Step 3***: After assigning all observations to the clusters, calculate the clustering score, by summing up all the Euclidean distances between each data point and the corresponding centroid.

Where:

k: the number of clusters

n: the number of points belonging to cluster j

cj: the centroid of cluster j

***Step 4:*** Define the new centroid of each cluster by calculating the mean of all points assigned to that cluster. Here’s the formula (n is the number of points assigned to that cluster):

***Step 5:*** Repeat from step 2 until the positions of the centroids no longer move or the assignments stay the same or reach the given number of ilterations (by users). (e.x just run for 1000 loops)

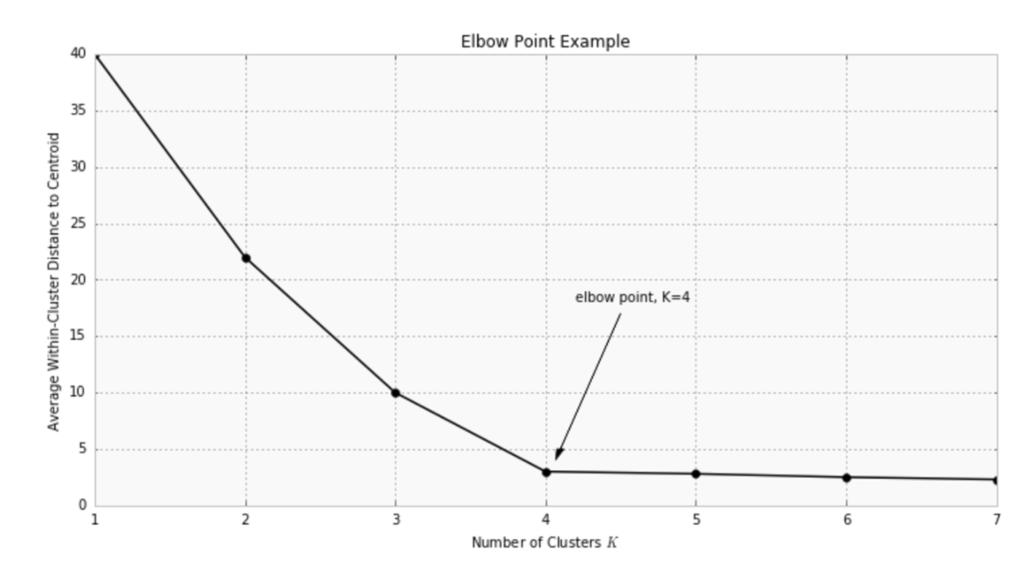
There you go: data points are now grouped into 4 different clusters. Using a simple idea of minimizing distances between data points to group them together, k-means clustering algorithm is extremely helpful for understanding the structure of the data, how observations are classified, and interpreting the story behind. K-means clustering has been widely used in data analysis, especially in life sciences, in analyzing thousands to millions of data points in single-cell RNA-seq and bulk RNA-seq experiments.

Note that the Euclidean metric measures the distance based on the vector connecting two points, and will cause some biases for data with different scales. For example, in RNA-seq data, gene expression values can range from as little as 0.001 to a thousand, stretching the data points along an axis. That is, the variable with the smaller scale will be easily dominated and play little in the convergence, as clusters will scatter along an axis only. For this reason, it is necessary to make sure that the variables are at the same scale before using k-means clustering.

## **How to determine the optimal value of k?**

Note that before determining the number of clusters to assign the data into (the variable k), you should have an overview of the data and on what basis you want to group them. You can even apply a hierarchical clustering on the data first to briefly understand the structure of the data before choosing k by hand.

A well-known method to validate the number of clusters is the Elbow method, that is to run k-means clustering several times for a range of values of k (usually from 2 to 10) and pick out the value of k that causes sudden drop in the sum of squared distances. More specifically, for each value of k, we calculate the sum of squared distances (between each point and the corresponding centroid) and graph the results on a line chart. Choose the value where the sum of squares drops, giving an angle in the graph (a.k.a an elbow) - that is the optimal value of k.



# **K-MEANS CLUSTERING ILLUSTRATION ON MATLAB**

## **Code for K-means Clustering (kmeans.m) which is running on Matlab**

%% Summary code: openExample('stats/PartitionDataIntoTwoClustersExample')

%% Partition Data into Two Clusters

%%

% Randomly generate the sample data.

rng default; % For reproducibility

X = [randn(100,2)\*0.75+ones(100,2);

randn(100,2)\*0.5-ones(100,2)];

figure;

plot(X(:,1),X(:,2),'.');

title 'Randomly Generated Data';

%%

% There appears to be two clusters in the data.

%%

% Partition the data into two clusters, and choose the best arrangement out of

% five initializations. Display the final output.

opts = statset('Display','final');

[idx,C] = kmeans(X,2,'Distance','cityblock',...

'Replicates',5,'Options',opts);

%%

% By default, the software initializes the replicates separately using

% \_k\_-means++.

%%

% Plot the clusters and the cluster centroids.

figure;

plot(X(idx==1,1),X(idx==1,2),'r.','MarkerSize',12)

hold on

plot(X(idx==2,1),X(idx==2,2),'b.','MarkerSize',12)

plot(C(:,1),C(:,2),'kx',...

'MarkerSize',15,'LineWidth',3)

legend('Cluster 1','Cluster 2','Centroids',...

'Location','NW')

title 'Cluster Assignments and Centroids'

hold off

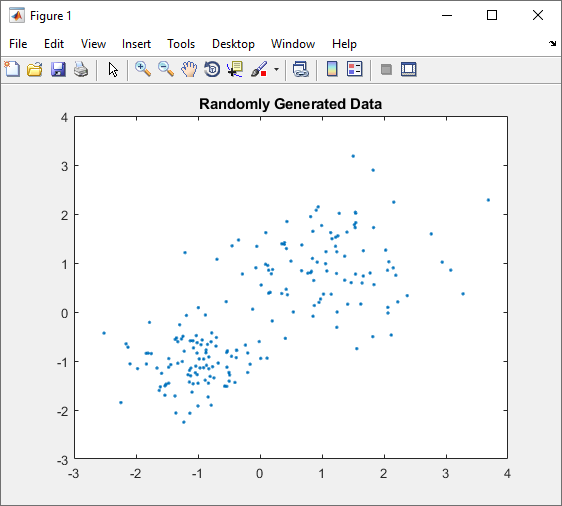
%%

% You can determine how well separated the clusters are by passing |idx| to

% <docid:stats\_ug.f3984482>. for K-means Clustering

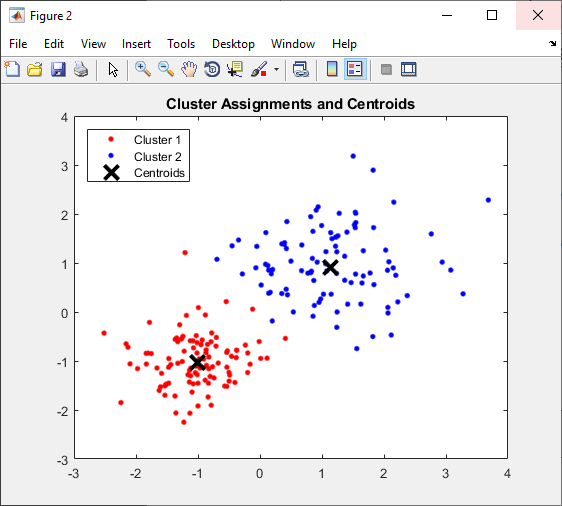
## **Figures of illustration of data before and after running**

**Step 1. Create the data**



*Figure 1. Illustration of data before running (step1)*

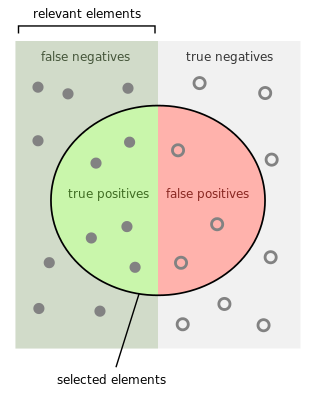
**Step 2. Set up 2 clusters and run the algorithm**

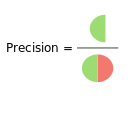
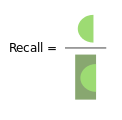


*Figure 2. Illustration of data after running*

# **PRECISION AND RECALL ON EXCEL**

1. **How to calculate precision and recall**



* **Calculate the precision**
* **Calculate the recall**

1. **Result of experience**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Predicted | Actual | TN | FN | FP | TP |
| 1.40325 | 1.630282 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.375414 | 0.333976 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.69414 | 1.07507 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.64663 | 0.591603 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.239074 | 1.227641 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.019234 | 0.549755 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.674806 | 1.367474 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.256968 | 1.554522 | 2 | 2 | 0 | 0 | 0 | 1 |
| 3.683798 | 2.283916 | 2 | 2 | 0 | 0 | 0 | 1 |
| 3.077078 | 0.854407 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.01242 | -0.60377 | 1 | 1 | 1 | 0 | 0 | 0 |
| 3.276193 | 0.370308 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.544053 | 2.015946 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.952709 | 0.195884 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.536057 | 1.720715 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.846275 | 1.093037 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.906892 | 2.077522 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.117273 | -0.47067 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.056776 | 0.851726 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.062894 | 0.094116 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.503623 | 3.181006 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.094385 | 1.618914 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.537929 | 2.034229 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.222676 | 0.206365 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.36667 | 0.648538 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.77602 | 0.795648 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.545164 | 1.823818 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.772419 | 0.791596 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.220404 | 1.526156 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.409538 | -0.53886 | 1 | 1 | 1 | 0 | 0 | 0 |
| 1.666297 | 0.734613 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.139697 | 0.38231 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.198347 | -0.18279 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.392876 | 1.380981 | 2 | 2 | 0 | 0 | 0 | 1 |
| -1.20821 | 1.211488 | 2 | 1 | 0 | 0 | 1 | 0 |
| 2.078785 | 1.02511 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.243893 | -0.00026 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.433804 | 1.845619 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.027724 | 1.262635 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.28364 | 0.7757 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.923318 | 1.017167 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.818915 | 0.803503 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.239405 | -0.31266 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.234644 | 0.785762 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.35134 | 0.376475 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.977462 | 0.265595 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.876341 | 0.132699 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.47078 | 0.599832 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.819949 | -0.50198 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.831955 | 1.723172 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.35226 | 1.390045 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.058019 | 0.984979 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.089412 | 0.973922 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.164874 | 0.401377 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.994863 | 1.764014 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.149473 | 0.900087 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.422751 | 0.464102 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.278534 | 2.013539 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.830812 | 0.831422 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.838017 | 0.558228 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.183202 | 0.779685 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.024418 | 0.364055 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.414395 | 0.159904 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.825458 | 2.8945 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.158159 | 2.241623 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.064448 | 1.230651 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.11869 | 0.057161 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0.443274 | 0.350899 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.203814 | 0.867599 | 1 | 2 | 0 | 1 | 0 | 0 |
| 2.762843 | 1.593562 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.538299 | 0.000997 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.561058 | -0.7474 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.855686 | -0.08682 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.666458 | 1.250133 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.426363 | 1.293515 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.0517 | 1.33876 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.06678 | 0.902287 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.366145 | 1.137767 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.866969 | 0.642885 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.85296 | 1.646516 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.064483 | -0.02127 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.218688 | 1.341272 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.148358 | 0.363468 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.190774 | 0.748835 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.396651 | 1.414588 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.522468 | 1.779318 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.626316 | 0.161771 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.817214 | 1.945494 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.161753 | 1.495107 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.125617 | 0.949101 | 1 | 2 | 0 | 1 | 0 | 0 |
| 0.139035 | 0.853584 | 1 | 2 | 0 | 1 | 0 | 0 |
| 1.078656 | 0.836795 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.541691 | 0.772669 | 2 | 2 | 0 | 0 | 0 | 1 |
| 2.939118 | 1.017284 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.499832 | 1.038468 | 2 | 2 | 0 | 0 | 0 | 1 |
| 1.140498 | 1.619547 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.938129 | 2.145233 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.44977 | 1.350186 | 2 | 2 | 0 | 0 | 0 | 1 |
| 0.670775 | 0.842715 | 1 | 2 | 0 | 1 | 0 | 0 |
| -0.34601 | 1.468893 | 2 | 2 | 0 | 0 | 0 | 1 |
| -0.90839 | -0.96003 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.51488 | -1.47424 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.52539 | -0.79425 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.84647 | -0.66151 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.93241 | -0.57113 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.74238 | -1.34558 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.8693 | -0.77531 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.47074 | -0.94968 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.08117 | -0.58697 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.07303 | -0.73192 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.26601 | -0.55106 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.15895 | -1.06597 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.43786 | -1.0736 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.24191 | -0.49611 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.356 | -2.06183 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.58711 | -1.25229 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.09612 | -1.6353 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.13704 | -1.19129 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.23496 | -0.67566 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.12451 | -0.58714 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.53211 | -1.50747 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.19827 | -1.23553 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.38266 | -0.93149 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.11481 | -1.14593 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.75308 | -0.84909 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.22231 | -0.80003 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.07797 | -1.46498 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.86197 | -1.08842 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.13058 | -2.06605 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.77829 | -0.42732 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.80405 | -1.31455 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.62534 | -1.60192 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.47398 | -1.12697 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.37055 | -1.71432 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.25391 | -1.01043 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.16029 | -1.28033 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.99377 | 0.088889 | 1 | 1 | 1 | 0 | 0 | 0 |
| -2.51459 | -0.43077 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.22851 | -2.24844 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.37878 | -0.77934 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.53335 | -1.69907 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.53314 | -1.12753 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.82484 | -0.9178 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.0145 | -0.62613 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.90877 | -1.13652 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.78253 | -0.21185 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.04227 | -1.24047 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.19803 | -0.83624 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.95083 | -0.66763 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.97931 | -0.95741 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.36708 | -0.55952 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.01541 | -0.83839 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.88383 | -1.39207 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.78681 | -1.90269 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.1864 | -0.0707 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.11823 | -1.30227 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0.011845 | -0.94832 | 1 | 1 | 1 | 0 | 0 | 0 |
| -2.12918 | -0.71842 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0.114723 | -0.9432 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.83122 | -1.45236 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.49997 | -1.23386 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.83208 | -1.06244 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.29502 | -0.26052 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.13903 | -1.43041 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.78864 | -0.60767 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.8351 | -0.84569 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.76418 | -1.11693 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.60642 | -1.52849 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.9669 | -1.14207 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.67382 | -1.04335 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.83647 | -1.7347 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.45868 | -0.90391 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.49696 | -1.41115 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.32545 | -1.04712 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.87147 | -0.83189 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.47219 | -1.45233 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.66089 | -1.14413 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.53759 | -0.82497 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.99998 | -1.91793 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.02746 | -0.48201 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.54444 | 0.212231 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.70271 | -0.5203 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.8249 | -1.15789 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.37487 | -0.78569 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.53511 | -1.51799 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.88012 | -0.06107 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.34518 | -0.52965 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.32578 | -0.60633 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.40395 | -1.43794 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.80592 | -0.84003 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.01223 | -1.27915 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.97442 | -1.15571 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.48975 | -1.285 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.56914 | -1.51287 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.99942 | -1.45437 | 1 | 1 | 1 | 0 | 0 | 0 |
| -1.03542 | -1.10495 | 1 | 1 | 1 | 0 | 0 | 0 |
| -2.24314 | -1.84943 | 1 | 1 | 1 | 0 | 0 | 0 |
| -0.70941 | -0.6962 | 1 | 1 | 1 | 0 | 0 | 0 |
| -2.09622 | -1.0589 | 1 | 1 | 1 | 0 | 0 | 0 |
| -2.15964 | -0.65042 | 1 | 1 | 1 | 0 | 0 | 0 |
| Total | | | | 103 | 24 | 1 | 72 |

|  |  |  |
| --- | --- | --- |
| **Actual class** | **Predicted class** | |
|  | Negative | Positive |
| Negative | TN | FP |
| Positive | FN | TP |

* Precision’s result
* Recall’s result