Applied machine learning

Assignment 3

Names :

Muhammed Mohsen

Phoebe Thabet Wadea

Sandy Adel Latef

Part 1: Calculations

A:

* Centroid 1: (6,3).
* Centroid 2: (2,1).
* By using Euclidean distance (a, b) = √ (a1 - b1)2 + (a2-b2)2

|  |  |  |  |
| --- | --- | --- | --- |
| **Data points** | **Centroid 1** | **Centroid 2** | **Cluster** |
| **A1** | 4.24 | 5.1 | 1 |
| **A2** | 0 | 4.47 | 1 |
| **A3** | 3.61 | 7.8 | 1 |
| **A4** | 4.47 | 0 | 2 |
| **A5** | 6.1 | 8.5 | 1 |

Cluster 1= {A1, A2, A3, A5} Cluster 2={A4}

* **Update centroids:**

Centroid 1= (5.5, 6).

Centroid 2= (2, 1).

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

B:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **10** |  |  |  |  |  |  |  |  |  |  |
| **9** |  |  |  |  | A5 |  |  |  |  |  |
| **8** |  |  |  |  |  |  |  |  |  |  |
| **7** |  |  |  |  |  |  |  |  |  |  |
| **6** |  |  | A1 |  |  |  |  | A3 |  |  |
| **5** |  |  |  |  |  |  |  |  |  |  |
| **4** |  |  |  |  |  |  |  |  |  |  |
| **3** |  |  |  |  |  | A2 |  |  |  |  |
| **2** |  |  |  |  |  |  |  |  |  |  |
| **1** |  | A4 |  |  |  |  |  |  |  |  |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Data points** | **Centroid 1 (5.5, 6)** | **Centroid 2 (2, 1)** | **Cluster** |
| **A1 (3,6)** | 2.5 | 5.1 | 1 |
| **A2 (6,3)** | 3.04 | 4.47 | 1 |
| **A3 (8,6)** | 2.5 | 7.8 | 1 |
| **A4 (2,1)** | 6.1 | 0 | 2 |
| **A5 (5,9)** | 3.04 | 8.5 | 1 |

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

C:

**Silhouette Score:** is calculated by the given equation:

Dissimilarity matrix

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Point** | **A1** | **A2** | **A3** | **A4** | **A5** |
| **A1** | 0 | 4.2 | 5 | 5.1 | 3.6 |
| **A2** |  | 0 | 3.6 | 4.5 | 6.1 |
| **A3** |  |  | 0 | 7.8 | 4.2 |
| **A4** |  |  |  | 0 | 8.5 |
| **A5** |  |  |  |  | 0 |

**A1:**

a = 4.2+5+3.6 / 3 = 4.3 , b= 5.1

SC = b-a / max (b, a) = 5.1 – 4.3 / 5.1 = 0.16

**A2:**

a = 4.2+3.6+6.1 / 3 = 4.6 , b= 4.5

SC = b-a / max (b, a) = 4.5 – 4.6 / 4.6= -0.02

**A3:**

a = 5+3.6+4.2 / 3 = 4.3 , b= 7.8

SC = b-a / max (b, a) = 7.8 – 4.3 / 7.8 = 0.45

**A4:**

a = 0 , b= 5.1+4.5+7.8+8.5 / 4 = 6.5

SC = b-a / max (b, a) = 6.5 – 0 / 6.5 = 1

**A5:**

a = 3.6+6.1+4.2 / 3 = 4.6 , b= 8.5

SC = b-a / max (b, a) = 8.5 – 4.6 / 8.5 = 0.46

**Average silhouette score for cluster 1:**

SC = (0.16 + (-0.02) + 0.45 + 0.46)/4= 0.26

**Average silhouette score for cluster 2:**

SC = 1 / 1= 1

**The overall silhouette score:**

0.26 + 1 / 2= 0.63

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**WSS Score:**

Centroid 1 (c1) = (5.5, 6)

Centroid 2 (c2) = (2, 1)

**Cluster 1:**

(A1, C1): + = 6.25

(A2, C1): + = 9.25

(A3, C1): + = 6.25

(A4, C2): + = 0

(A5, C1): + = 9.25

**WSS score** =

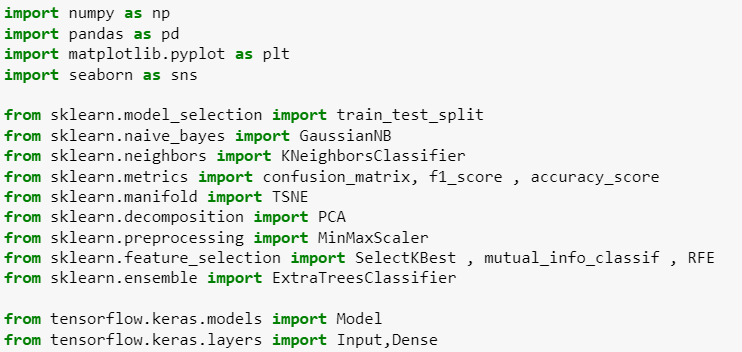
= 6.25 + 9.25 + 6.25 + 0 + 9.25= 31

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Part 2: Programming

1. Use scikit-learn to implement Naive Bayes Classifier (NB) and K-Nearest Neighbor classifiers (KNN) on the provided Mobile Crowd Sensing (MCS) dataset.

* We import some libraries.

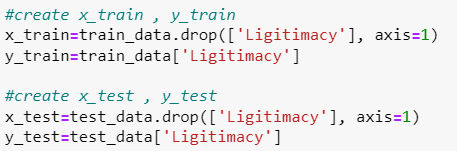


* We make the values 0, 1 and 2 in column day used for training dataset and only the value 3 in column day should be used for test dataset.

A screenshot of a computer code

Description automatically generated

* a: we create training and test datasets for remaining parts.



* b: we provide confusion matrixes and F1 scores of NB and KNN classifier as baseline performances:
* first, we train NB and KNN classifier.
* Then, we predict NB and KNN classifier.
* Then, we calculate confusion matrix and F1 score for NB and KNN classifier.
* Then, we print the results.

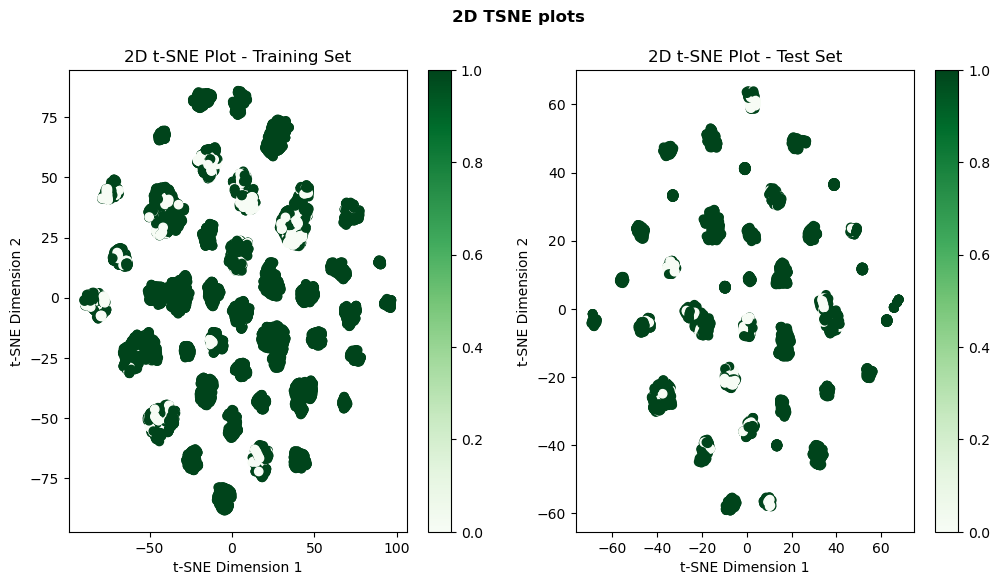
A screenshot of a computer code

Description automatically generated

* A green and white squares

  Description automatically generatedThen, we plot the confusion matrix.
* C: we provide 2D TSNE plots, one for the training set and one for the test set.
* First, we make function train TSNE, and plot 2D TSNE for training & test set.



* Then, we plot 2D TSNE plots, one for the training set and one for the test set.

1. Apply the following Dimensionality Reduction (DR) methods: PCA

(n components=n, random state=0) and Auto Encoder (AE).

* a: we find the best reduced dimensions of PCA and AE based on f1 score of test dataset using both classifiers (NB and KNN).
* First, we scale the data and define no of components.

A screenshot of a computer code

Description automatically generated

* Then, we used this scaled data in PCA method to find the best no of components and print the performance results for each no of components and the best performance for each classifier.

A screenshot of a computer

Description automatically generated

* Then, we created an autoencoder from higher to lower number of neurons.

A screenshot of a computer program

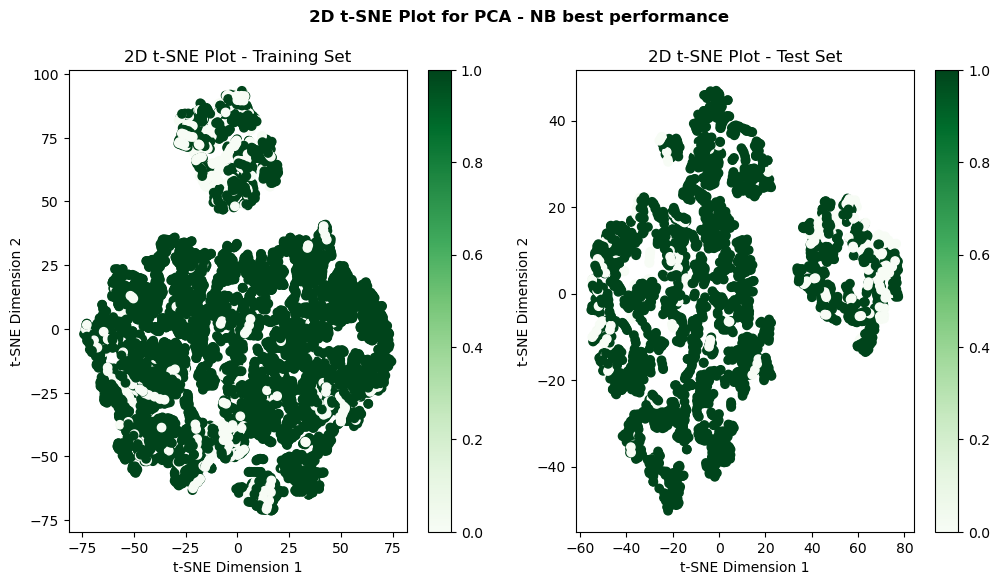
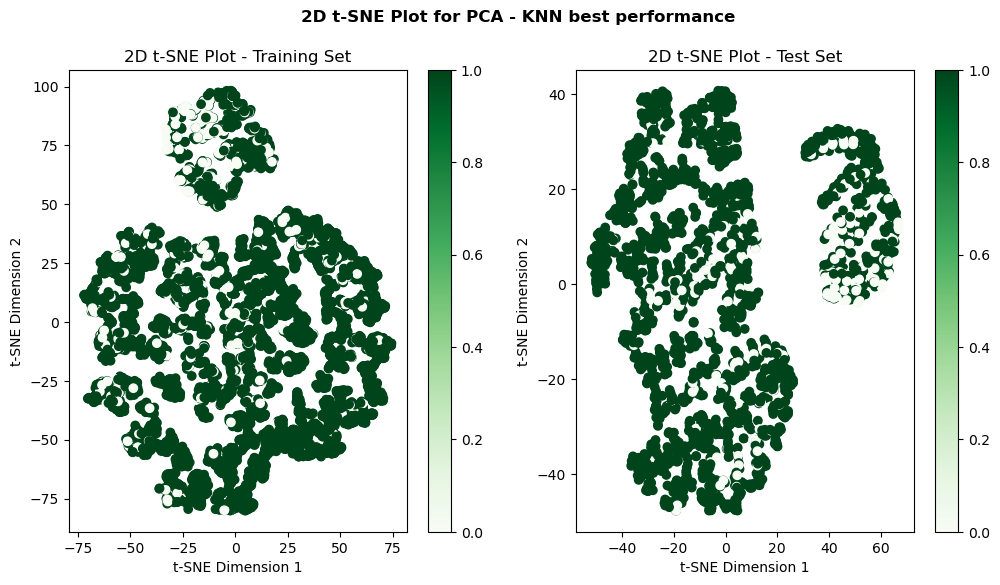
Description automatically generated

* Then, we used this scaled data in PCA method to find the best no of components and print the performance results for each no of components and the best performance for each classifier.

A screenshot of a computer

Description automatically generated

* A graph of different types of data

  Description automatically generatedThen, we plot the number of components (dimension) vs f1 score together with baseline performances for each classifier.
* C: we provide 2D TSNE plots for the best performance in previous part one for the training set and one for the test set.

A screenshot of a graph

Description automatically generated

A green and white graph

Description automatically generated

1. We use Feature Selection methods (one for each method) and find the best number of features based on both the NB and KNN classifiers f1 scores.

* A: we use filter Methods (Information Gain) and print the performance results for each no of components and the best performance for each classifier.

A screenshot of a computer

Description automatically generated

* B: we use wrapper Methods (Recursive Feature Elimination) and print the performance results for each no of components and the best performance for each classifier.

A screenshot of a computer

Description automatically generated

* A group of graphs showing different types of data

  Description automatically generatedThen, we Plot the number of features vs f1 score with the improved baseline performance.
* A graph of a number of different types of data

  Description automatically generatedThen, we Plot the number of features vs accuracy with the improved baseline performance.
* A screenshot of a graph

  Description automatically generatedA screenshot of a graph

  Description automatically generatedC: provide 2D TSNE plots, one for the training set and one for the test set, using only the best method (filter method).

1. We Choose the best number of cluster among 8, 12, 16, 20 and 32 clusters and we make latitude and longitude features be considered for clustering base methods.

* A:
* B:
* C:

1. conclusion part:

* 1- we use part of data not all the data.
* 1- in NB and KNN confusion matrix the TN is very large and F values are very small that make f1-score for both are large, so these classifiers are good for our data.
* 2- in PCA, f1-score for all n-components is large and very close, but the best n-components for NB is 6=0.923, the best n-components for KNN is 3=0.929, so PCA is very suitable for our data.
* 2- in autoencoder, f1-score are changeable in each run, so we prefer PCA.
* 3- we plot f1-score vs n-component then, we plot accuracy vs n-components, we discover that plots are very similar, so accuracy and f1-score are related.
* 3- we discovered that filter method has the best f1-score and accuracy, so the best performance is filter method for KNN and n-components 2=0.983.
* 4-
* 4-