Machine Learning Digital Assignment 1

Name: Om Ashish Mishra

Registration Number: 16BCE0789

Slot: A2

The Question:

3. Design a Machine Learning system (using at least two algorithms/techniques) to analyse and classify Network Traffic

The Answer:

I have taken a dataset from kaggle to do my assignment and here are the details:-

Link of dataset: https://www.kaggle.com/crawford/computer-network-traffic

Each row consists of four columns:

1. date: yyyy-mm-dd (from 2006-07-01 through 2006-09-30)
2. l\_ipn: local IP (coded as an integer from 0-9)
3. r\_asn: remote ASN (an integer which identifies the remote ISP)
4. f: flows (count of connections for that day)

Reports of "odd" activity or suspicions about a machine's behavior triggered investigations on the following days (although the machine might have been compromised earlier)

The Algorithms are:

**K Means**

# K-Means Clustering

# Importing the libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Importing the dataset

dataset = pd.read\_csv('da1.csv')

X = dataset.iloc[:, [1,3]].values

# y = dataset.iloc[:, 3].values

# Using the elbow method to find the optimal number of clusters

from sklearn.cluster import KMeans

wcss = []

for i in range(1, 3):

kmeans = KMeans(n\_clusters = i, init = 'k-means++', random\_state = 42)

kmeans.fit(X)

wcss.append(kmeans.inertia\_)

plt.plot(range(1, 3), wcss)

plt.title('The Elbow Method')

plt.xlabel('Number of clusters')

plt.ylabel('WCSS')

plt.show()

# Fitting K-Means to the dataset

kmeans = KMeans(n\_clusters = 2, init = 'k-means++', random\_state = 42)

y\_kmeans = kmeans.fit\_predict(X)

# Visualising the clusters

plt.scatter(X[y\_kmeans == 0, 0], X[y\_kmeans == 0, 1], s = 100, c = 'red', label = 'Cluster 1')

plt.scatter(X[y\_kmeans == 1, 0], X[y\_kmeans == 1, 1], s = 100, c = 'blue', label = 'Cluster 2')

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], s = 300, c = 'yellow', label = 'Centroids')

plt.title('Clusters of Traffic')

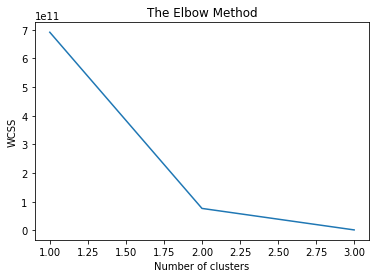
plt.xlabel('Local IP')

plt.ylabel('Remote ASN')

plt.legend()

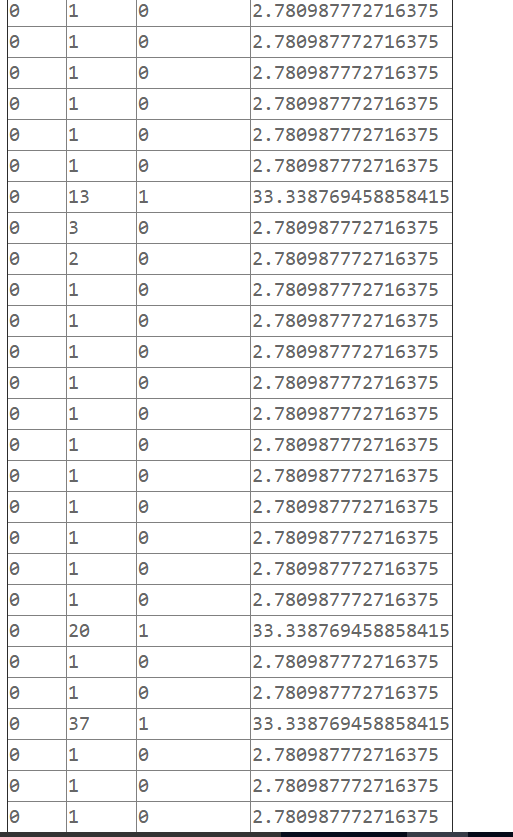
plt.show()

The Output:



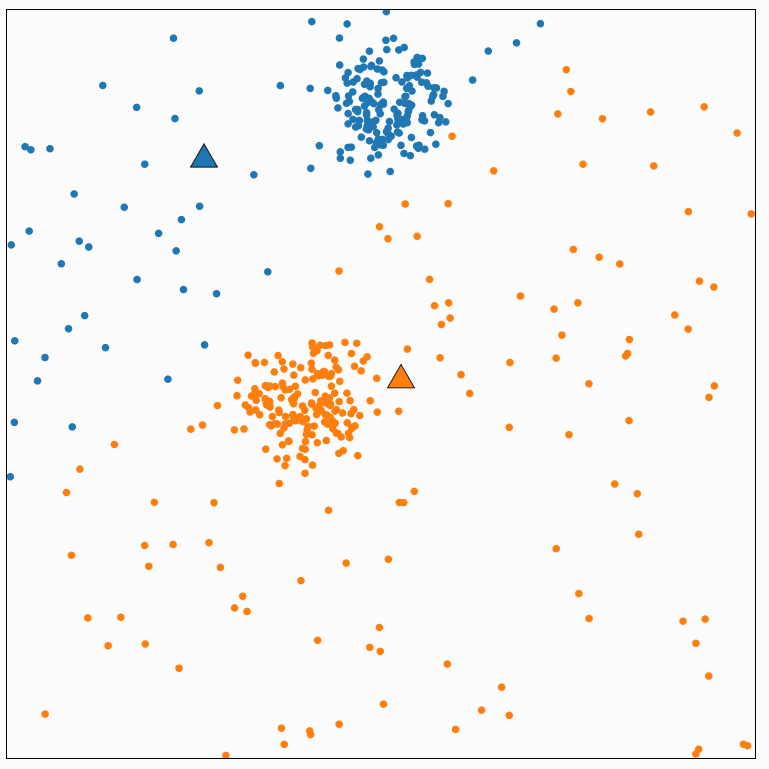
Thus from the elbow graph we conclude 2 clusters are optimal for this data set.

The sample distribution is shown below:



Thus the distribution of the data is shown in 2 centroids one at 33.338769458858415 and other at 2.780987772716375.

Thus I conclude 2 clusters and unsupervised learning taking place in the process.



Classification

Linear Classification

X = connection flow

Y = IP address

The Code:

# -\*- coding: utf-8 -\*-

"""

Created on Mon Jan 7 23:19:01 2019

@author: OM MISHRA

"""

# Logistic Regression

# Importing the libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Importing the dataset

dataset = pd.read\_csv('da1.csv')

X = dataset.iloc[:, [1,2]].values

y = dataset.iloc[:, 3].values

# Splitting the dataset into the Training set and Test set

#from sklearn.cross\_validation import train\_test\_split

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 0)

# Feature Scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

# Fitting Logistic Regression to the Training set

from sklearn.linear\_model import LogisticRegression

classifier = LogisticRegression(random\_state = 0)

classifier.fit(X\_train, y\_train)

# Predicting the Test set results

y\_pred = classifier.predict(X\_test)

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

# Visualising the Training set results

from matplotlib.colors import ListedColormap

X\_set, y\_set = X\_train, y\_train

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 1, stop = X\_set[:, 0].max() + 1, step = 0.01),

np.arange(start = X\_set[:, 1].min() - 1, stop = X\_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),

alpha = 0.75, cmap = ListedColormap(('red', 'green','blue','yellow','pink','orange','brown','white','black')))

plt.xlim(X1.min(), X1.max())

plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y\_set)):

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1],

c = ListedColormap(('red', 'green','blue','yellow','pink','orange','brown','white','black'))(i), label = j)

plt.title('Logistic Regression (Training set)')

plt.xlabel('Connection Flow')

plt.ylabel('IP Address')

plt.legend()

plt.show()

# Visualising the Test set results

from matplotlib.colors import ListedColormap

X\_set, y\_set = X\_test, y\_test

X1, X2 = np.meshgrid(np.arange(start = X\_set[:, 0].min() - 1, stop = X\_set[:, 0].max() + 1, step = 0.01),

np.arange(start = X\_set[:, 1].min() - 1, stop = X\_set[:, 1].max() + 1, step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),

alpha = 0.75, cmap = ListedColormap(('red', 'green','blue','yellow','pink','orange','brown','white','black')))

plt.xlim(X1.min(), X1.max())

plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y\_set)):

plt.scatter(X\_set[y\_set == j, 0], X\_set[y\_set == j, 1],

c = ListedColormap(('red', 'green','blue','yellow','pink','orange','brown','white','black'))(i), label = j)

plt.title('Logistic Regression (Test set)')

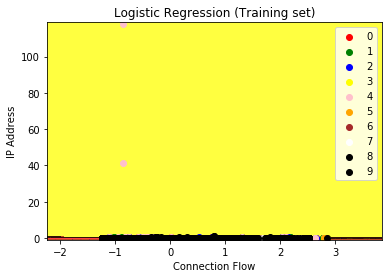
plt.xlabel('Connection Flow')

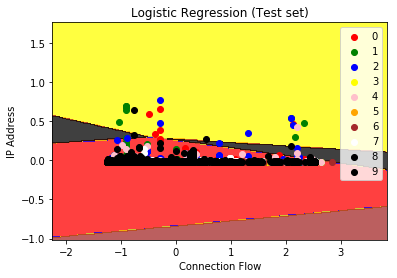
plt.ylabel('IP Address')

plt.legend()

plt.show()

The Answer:





Thus we can see the 3rd IP address is more pre dominating over the other factors in the training set but it was found that the predominance is not true as far as traffic is considered as the data distribution will take different path depending on the need of the production and flow of the data from one point to the other.