МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ «МИСиС»**

Институт ИТКН

Кафедра инженерной кибернетики

Направление подготовки: «01.03.04 Прикладная математика»

Квалификация: бакалавр

Группа: БПМ-17-2

**ОТЧЕТ**

**ПО КУРСОВОЙ РАБОТЕ**

**«ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ»**

на тему: «Кластерный анализ. Расстояния (Евклида, Хемминга, Минковского и др.)»

**Студент** \_\_\_\_\_\_\_\_\_\_/ Сухарев Р.И.

**Руководитель** \_\_\_\_\_\_\_\_\_\_/ старший преподаватель, Кондыбаева А.Б.

**Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Москва 2020**

**Содержание**

[1. Введение 3](#_Toc59842438)

[2. Теоретическая часть 3](#_Toc59842439)

[2.1. Введение в кластерный анализ 3](#_Toc59842440)

[2.2. Задача кластерного анализа 4](#_Toc59842441)

[2.3. Основные понятия кластерного анализа 5](#_Toc59842442)

[2.4. Методы кластеризации и популярные используемые расстояния 7](#_Toc59842443)

[2.5. Применение кластерного анализа 10](#_Toc59842444)

[3. Реализация модели 11](#_Toc59842445)

[4. Список использованных источников 14](#_Toc59842446)

[5. Приложение 15](#_Toc59842447)

[5.1. Код файла kmeans.py 15](#_Toc59842448)

# Введение

Исследователь часто стоит перед лицом огромной массы индивидуальных наблюдений. Возникает задача сведения множества характеристик к небольшому ряду обобщающих итогов, выражающему действительно существенное для явления. Но пока каждый вовлеченный в анализ признак остается отдельным самостоятельным элементом со своими характеристиками, число параметров, выражающих результаты обработки, не поддается уменьшению. Единственный путь к нему – либо в отсечении большинства признаков и возвращении к малоразмерным классическим задачам, либо в объединении признаков, в замене целых «гроздей» их одним, искусственно построенным на их основе. Так и появилось направление –

«многомерный анализ».

В многомерном статистическом анализе образовались разделы, которые не изолированы, а проникают, переходят один в другой. Это кластерный анализ, метод главных компонент, факторный анализ. Наиболее ярко отражают черты многомерного анализа в классификации объектов кластерный анализ, а в исследовании связей – факторный анализ.

**Кластерный анализ** – это способ группировки многомерных объектов, основанный на представлении результатов отдельных наблюдений точками подходящего геометрического пространства с последующим выделением групп как «сгустков» этих точек (кластеров, таксонов). «Кластер» (cluster) в английском языке означает «сгусток», «гроздь винограда», «скопление звезд» и т.д. Данный метод исследования получил развитие в последние годы в связи с возможностью компьютерной обработки больших баз данных.

# Теоретическая часть

## 2.1. Введение в кластерный анализ

Кластерный анализ предполагает выделение компактных, удаленных друг от друга групп объектов, отыскивает «естественное» разбиение совокупности на области скопления объектов. Он используется, когда исходные данные представлены в виде матриц близости или расстояний между объектами либо в виде точек в многомерном пространстве. Наиболее распространены данные второго вида, для которых кластерный анализ ориентирован на выделение некоторых геометрически удаленных групп, внутри которых объекты близки.

Выбор расстояния между объектами является узловым моментом исследования, от него во многом зависит окончательный вариант разбиения объектов на классы при данном алгоритме разбиения.

Существует большое количество алгоритмов кластерного анализа, их можно разделить по способу построения кластеров на 2 типа: эталонные и неэталонные. В процедурах эталонного типа на множестве объектов задается несколько исходных зон, с которых начинает работу алгоритм. Эталоны могут представлять собой первоначальное разбиение на классы, центр тяжести класса и др. После задания эталонов алгоритм производит классификацию, иногда меняя определенным способом эталоны.

К алгоритмам кластеризации, работающим по иному принципу, относятся иерархические алгоритмы кластерного анализа, процедура разрезания и др.

## Задача кластерного анализа

Пусть множество I={I1,I2,…,In} обозначает n объектов. Результат измерения i-й характеристики Ij объекта обозначают символом xij, а вектор Xj=[xij] отвечает каждому ряду измерений (для j-го объекта). Таким образом, для множества I объектов исследователь располагает множеством векторов измерений X={X1, X2,…,Xn}, которые описывают множество I. Множество X может быть представлено как n точек в p-мерном евклидовом пространстве Ер.

Пусть m – целое число, меньшее чем n. Задача кластерного анализа заключается в том, чтобы на основании данных, содержащихся во множестве Х, разбить множество объектов I на m кластеров (подмножеств) 1,2,…, m так, чтобы каждый объект Ij принадлежал одному и только одному подмножеству разбиения и чтобы объекты, принадлежащие разным кластерам, были разнородными (несходными).

**Решением задачи кластерного анализа** является разбиение, удовлетворяющее некоторому условию оптимальности. Этот критерий может представлять собой некоторый функционал, выражающий уровни желательности различных разбиений и группировок. Этот функционал часто называют целевой функцией. Задачей кластерного анализа является задача оптимизации, т.е. нахождение минимума целевой функции при некотором заданном наборе ограничений. Примером целевой функции может служить, в частности, сумма квадратов внутригрупповых отклонений по всем кластерам.

## Основные понятия кластерного анализа

N измерений X1, X2,…,Xn могут быть представлены в виде матрицы

X=[X1, X2,…,Xn]=

 *x*11

*x*21



*x*12 *x*22

...

...

*x*1*N* 

*x*2 *N* 

 .

 ...

...

...

... 

 *xn*1



*xn*2

...

*xnN* 

Аналогичным образом расстояния между парами векторов d(Xi,Xj) могут быть представлены в виде матрицы расстояний:



 0 *d*12

...

*d*1*n* 



*d*21

= 

 ...

0

...

... *d* 2*n* 

... ... 

 ,

*dn*1



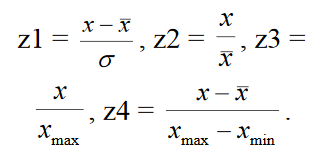


*dn*2

... 0 

dii=0 для i=1,2,…,n.

Если признаки измерены в разных единицах измерения, то определить расстояние между объектами нельзя. Тогда применяется нормировка показателей, переводящая их в безразмерные величины. Наиболее распространенные способы нормирования следующие:



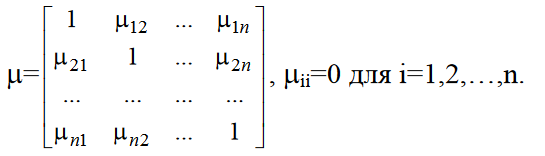
Понятием, противоположным понятию расстояния между объектами Xi и Xj, является понятие близости (сходства) между Xi и Xj. Точнее, мера близости между объектами Xi и Xj – это вещественная функция (Xi,Xj)=ij со свойствами:

0(Xi,Xj)<1 для XiXj;

(Xi,Xi)=1;

(Xi,Xj)=(Xj,Xi).

Пары значений мер близости можно объединить в матрицу близости:



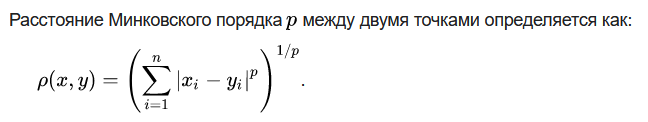
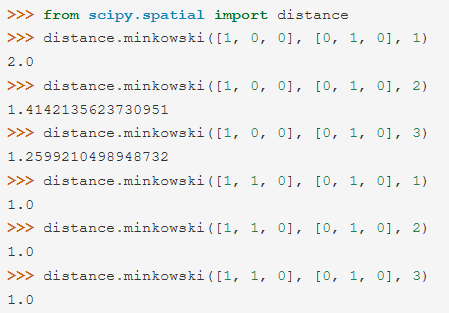
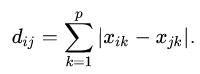
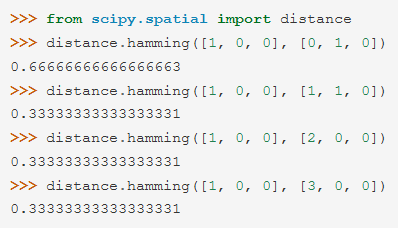
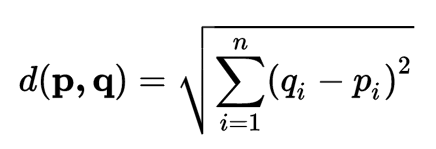
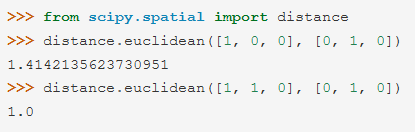
Величину ij называют коэффициентом близости. Примером линейной близости является коэффициент корреляции.

Рассмотрим основные способы определения расстояний между объектами.

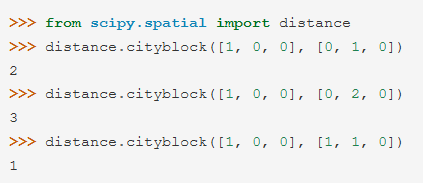
## Методы кластеризации и популярные используемые расстояния

1. *Вероятностный подход*. Предполагается, что каждый рассматриваемый объект относится к одному из k классов. Некоторые авторы (например, А. И. Орлов) считают, что данная группа вовсе не относится к кластеризации и противопоставляют её под названием «дискриминация», то есть выбор отнесения объектов к одной из известных групп (обучающих выборок).
   * K-средних
   * К-медиан
   * EM-алгоритм
   * Алгоритмы семейства FOREL
   * Дискриминантный анализ
2. Подходы на основе систем искусственного интеллекта: весьма условная группа, так как методов очень много и методически они весьма различны.
   * Метод нечеткой кластеризации C-средних (C-means)
   * Нейронная сеть Кохонена
   * Генетический алгоритм
3. Логический подход. Построение дендрограммы осуществляется с помощью дерева решений.
4. Теоретико-графовый подход.
   * Графовые алгоритмы кластеризации
5. Иерархический подход. Предполагается наличие вложенных групп (кластеров различного порядка). Алгоритмы в свою очередь подразделяются на агломеративные (объединительные) и дивизивные (разделяющие). По количеству признаков иногда выделяют монотетические и политетические методы классификации.
   * Иерархическая дивизивная кластеризация или таксономия. Задачи кластеризации рассматриваются в количественной таксономии.
6. Другие методы. Не вошедшие в предыдущие группы.
   * Статистические алгоритмы кластеризации
   * Ансамбль кластеризаторов
   * Алгоритмы семейства KRAB
   * Алгоритм, основанный на методе просеивания
   * DBSCAN и др.

А также в рамках кластерного анализа используются следующие расстояния:

* Расстояние Минковского   
  
* Расстояние Хэмминга  
   
* Расстояние Евклида  
    
  
* Расстояние городских кварталов





## Применение кластерного анализа

Кластерный анализ применяется для решения задач в следующих направлениях:

* Биология
* Социология
* Информатика
* Кластеризация результатов поиска
* Сегментация изображений
* Интеллектуальный анализ данных

# Реализация модели

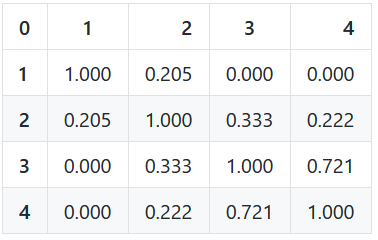
Для демонстрации и реализации кластерного анализа рассмотрим решение простой задачи.

Данные неориентированного графа используются в этой задаче. Он представляет связанные блоги с помеченными двумя классами. В этом проекте метод К-средних используется для кластеризации этих данных, и вычисления были выполнены для F-меры и чистоты.

Данные предварительно обрабатываются для создания матрицы связи, а затем создается матрица подобия с функциями подобия. Конкретно в этой задаче использовалось Манхэттенское расстояние.

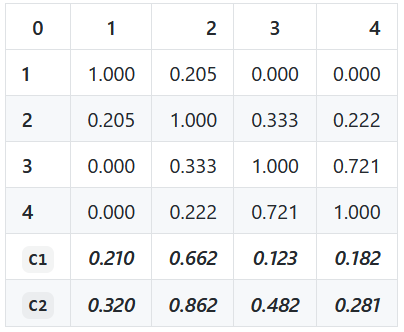


Пример матрицы связей



Пример матрицы расстояний

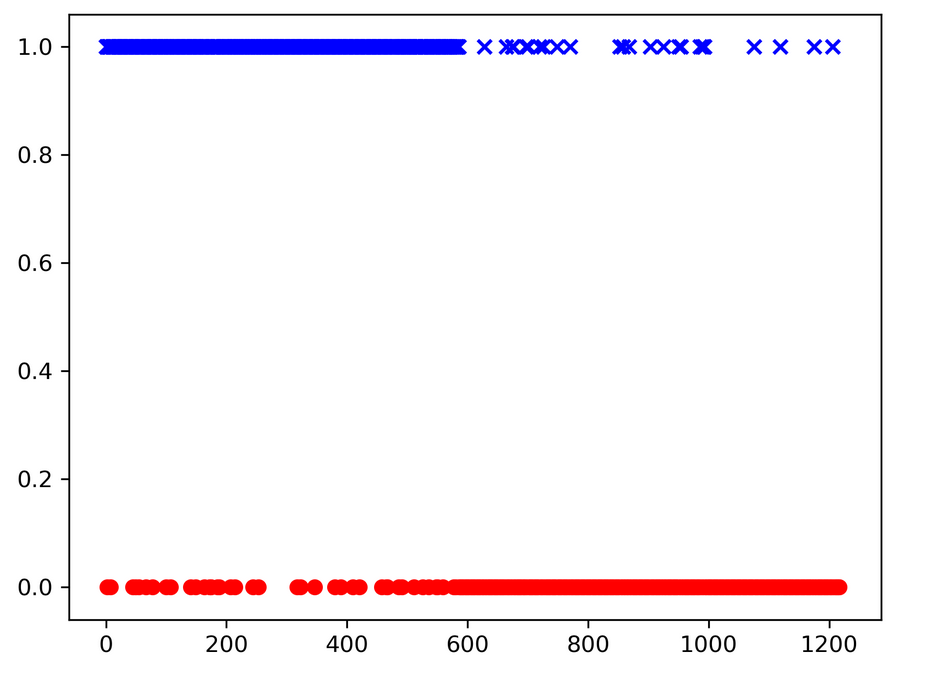
Число K случайных центроидов с добавленными N случайными атрибутами (т. е. подобием по отношению к каждому узлу)



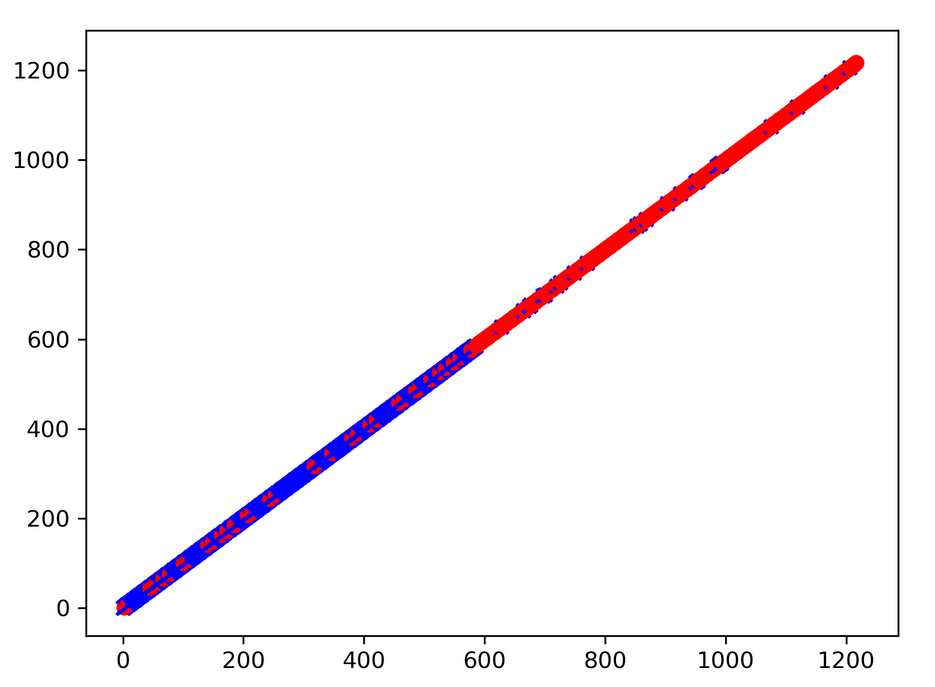
Далее метод К-средних применяется на матрице расстояний.

* Какой центроид каждого из N узлов ближе всего к найденному (расстояние каждого узла относительно каждого центроида)
* Центроиды обновляются после каждой кластеризации (среднее значение каждого атрибута - строк - для кластера в рассчитанной матрице)

Рассчитываются точность, отзывчивость и чистота (с энтропией).



Граф кластеризации Nx(кол-во классов)



Граф кластеризации NxN

# Список использованных источников

1. *Хайдуков Д. С.* Применение кластерного анализа в государственном управлении// Философия математики: актуальные проблемы. — М.: МАКС Пресс, 2009. — 287 с.
2. *Бериков В. С., Лбов Г. С.* Современные тенденции в кластерном анализе Архивная копия от 10 августа 2013 на Wayback Machine // Всероссийский конкурсный отбор обзорно-аналитических статей по приоритетному направлению.
3. *Вятченин Д. А.* Нечёткие методы автоматической классификации. — Минск: Технопринт, 2004. — 219 с.

# Приложение

## 5.1. Код файла kmeans.py

import numpy as np

import matplotlib.pyplot as plt

import os

import math

DECIMAL\_POINT = 3

MAX\_ITERATION = 500

CLUSTER\_FOLDER\_NAME = "kmeans"

CLUSTER\_FILE\_NAME = "clusters"

DATA\_PATH = "polblogs.txt"

DATA\_LABELS\_PATH = "polblogs-labels.txt"

SIMILARITY\_MATRIX = "blogsSimilarityMatrix.txt"

def manhattan\_distance(first\_row, second\_row):

if np.array\_equal(first\_row,second\_row):

return 1

counter\_first = 0

counter\_second = 0

counter\_mutual = 0

for i in range(len(first\_row)):

if first\_row[i] == 1 and second\_row[i] == 1:

counter\_mutual = counter\_mutual + 1

elif first\_row[i] == 1 and second\_row[i] == 0:

counter\_first += 1

elif first\_row[i] == 0 and second\_row[i] == 1:

counter\_second += 1

if(counter\_first + counter\_second - counter\_mutual) == 0:

return 0

else:

return counter\_mutual / (counter\_first + counter\_second - counter\_mutual)

def pre\_process\_data():

print("Pre-processing data started...")

data = np.loadtxt(DATA\_PATH)

x, y = data[:, 0], data[:,1]

blogs = np.zeros((1218, 1218),dtype=int)

for i in range(len(x)):

a, b = int(x[i]), int(y[i])

blogs[a,b] = 1

blogs[b,a] = 1

print("Connected adjency mtrix created!")

print(blogs)

similarity\_matrix = np.zeros((1218, 1218),dtype=float)

print("Manhattan distance calculations started...")

for i in range(1218):

for j in range(1218):

similarity = manhattan\_distance(blogs[i,:], blogs[j,:])

similarity\_matrix[i, j] = similarity

print("Similarity matrix created!")

np.savetxt(SIMILARITY\_MATRIX, similarity\_matrix, fmt="%.3f", delimiter="\t")

print("Pre processing finished, all files saved!!")

def kmeans(data, k, tour):

print("K-Means Started...")

not\_converged = True

centroids = np.zeros((len(data), k), dtype=float)

for i in range(2):

for j in range(len(centroids)):

centroids[j,i] = np.around(np.random.uniform(0.0, 1.0),DECIMAL\_POINT)

cluster0 = []

cluster1 = []

counter = 0

while not\_converged:

counter += 1

if counter > MAX\_ITERATION:

not\_converged = False

tmp\_cluster0 = cluster0[:]

tmp\_cluster1 = cluster1[:]

for i in range(len(data)):

sumC1 = 0

sumC2 = 0

for j in range(len(data)):

sumC1 = sumC1 + abs(centroids[i,1] - data[i,j])

sumC2 = sumC2 + abs(centroids[i,0] - data[i,j])

if sumC1 < sumC2:

if i not in cluster0:

cluster0.append(i)

if i in cluster1:

cluster1.remove(i)

else:

if i not in cluster1:

cluster1.append(i)

if i in cluster0:

cluster0.remove(i)

if tmp\_cluster0 == cluster0 and tmp\_cluster1 == cluster1:

print("Clusters Same Thats finished!")

not\_converged = False

else:

oldCentroids = np.copy(centroids)

centroids = update\_centroids(centroids,cluster0,cluster1,data)

if np.array\_equal(oldCentroids,centroids):

print("Centroids same, Finished!")

not\_converged = False

print("Final Iteration number = ", counter)

print("cluster0 = ", cluster0)

print("cluster1 = ", cluster1)

labels = np.empty((len(data),2), dtype=int)

for i in range(len(cluster0)):

labels[cluster0[i],0] = cluster0[i]

labels[cluster0[i],1] = 0

for i in range(len(cluster1)):

labels[cluster1[i],0] = cluster1[i]

labels[cluster1[i],1] = 1

if not os.path.exists(CLUSTER\_FOLDER\_NAME):

os.makedirs(CLUSTER\_FOLDER\_NAME)

print("Clusters folder created.")

file\_path = CLUSTER\_FOLDER\_NAME

file\_path += "/"

filename = CLUSTER\_FILE\_NAME

file\_path += filename

file\_path += str(tour)

file\_path += ".txt"

np.savetxt(file\_path, np.int\_(labels), fmt="%d")

print("K-Means clustering finished. Files are ready!")

def update\_centroids(centroids,cluster0,cluster1,data):

for i in range(len(data)):

sums = 0

for j in range(len(cluster0)):

sums = sums + data[cluster0[j],i]

centroids[i,0] = np.around(sums / len(cluster0),DECIMAL\_POINT)

for i in range(len(data)):

sums = 0

for j in range(len(cluster1)):

sums = sums + data[cluster1[j],i]

centroids[i,1] = np.around(sums / len(cluster1),DECIMAL\_POINT)

return centroids

def plot\_kmeans():

print("Plot Clusters...")

polblogs\_labels = np.int\_(np.loadtxt(DATA\_LABELS\_PATH))

folder\_path = CLUSTER\_FOLDER\_NAME

folder\_path += "/"

files = os.listdir(folder\_path)

accuracy\_list = []

for name in files:

preLabel = folder\_path

preLabel += name

cluster\_labels = np.int\_(np.loadtxt(preLabel))

cluster\_labels\_only = cluster\_labels[:,1]

matched = 0

for i in range(len(polblogs\_labels)):

if polblogs\_labels[i] != cluster\_labels\_only[i]:

matched += 1

if(len(polblogs\_labels) - matched) > matched:

accuracy = (len(polblogs\_labels) - matched) / len(polblogs\_labels)

accuracy\_list.append(accuracy)

else:

accuracy = matched/len(polblogs\_labels)

accuracy\_list.append(accuracy)

best\_accuracy\_file = files[accuracy\_list.index(max(accuracy\_list))]

folder\_path = CLUSTER\_FOLDER\_NAME

folder\_path += "/"

folder\_path += best\_accuracy\_file

cluster\_labels1 = np.int\_(np.loadtxt(folder\_path))

list\_of\_symbols = ["o", "x", "v", "^", "<", "H", "8", "|", "\_", "P"]

for i in range(len(cluster\_labels1)):

if cluster\_labels1[i,1] == 0:

plt.scatter(cluster\_labels1[i, 0], cluster\_labels1[i, 1], color="red", marker=list\_of\_symbols[0])

if cluster\_labels1[i,1] == 1:

plt.scatter(cluster\_labels1[i, 0], cluster\_labels1[i, 1], color="blue", marker=list\_of\_symbols[1])

plt.show()

plt.clf()

return cluster\_labels1

def f\_measure(cluster):

tp = 0

tn = 0

fp = 0

fn = 0

polblogs\_labels = np.int\_(np.loadtxt(DATA\_LABELS\_PATH))

classlabels = cluster[:,1]

for i in range(len(polblogs\_labels)):

if polblogs\_labels[i] == 1 and polblogs\_labels[i] == classlabels[i]:

tp += 1

elif polblogs\_labels[i] == 0 and polblogs\_labels[i] == classlabels[i]:

tn += 1

elif polblogs\_labels[i] == 1 and polblogs\_labels[i] != classlabels[i]:

fn += 1

elif polblogs\_labels[i] == 0 and polblogs\_labels[i] != classlabels[i]:

fp += 1

if tp < fp:

tmp = tp

tp = fp

fp =tmp

if tn < fn:

tmp = tn

tn = fn

fn = tmp

precision = tp / (tp + fp)

recall = tp / (tp + fn)

print("True positive counts = ", tp)

print("True negative counts = ", tn)

print("False positive counts = ", fp)

print("False negative counts = ", fn)

print("Precision = ", precision)

print("Recall = ", recall)

print("F-Measure = ", 2 \* (precision \* recall) / (precision + recall))

entropy = purity((tp+fp)/len(polblogs\_labels), (tn+fn)/len(polblogs\_labels))

print("Purity = ", entropy)

def purity(pos, neg):

print("Calculating entropy")

entropy = (-pos\*math.log(pos)) - (-neg \* math.log(neg))

return entropy

def find\_clusters():

data = np.loadtxt(SIMILARITY\_MATRIX)

kmeans(data, 2,1)

pre\_process\_data()

find\_clusters()

cluster = plot\_kmeans()

f\_measure(cluster)

print("Finised")