UNIVERSIDAD POLITECNICA SALESIANA

SEDE CUENCA

CARRERA: *INGENIERIA DE SISTEMAS*

Nombre: Bryam Gabriel Mora Lituma

Materia: Sistemas Expertos

Fecha:10/02/2021

Generar un sistema de recomendación de películas aplicando Logica Difusa y el algoritmo KNN.

Libreria SciKit-Fuzzy.

Scikit-Fuzzy es una colección de algoritmos de lógica difusa destinados a usarse en SciPy Stack , escritos en el lenguaje informático Python.

Este SciKit está desarrollado por la comunidad SciPy.

Algoritmo K-Nearest Neighbors.

El algoritmo de K-Vecinos más cercanos calcula un valor de distancia para todos los pares de nodos en el gráfico y crea nuevas

relaciones entre cada nodo y sus k vecinos más cercanos. La distancia se calcula en función de las propiedades del nodo.

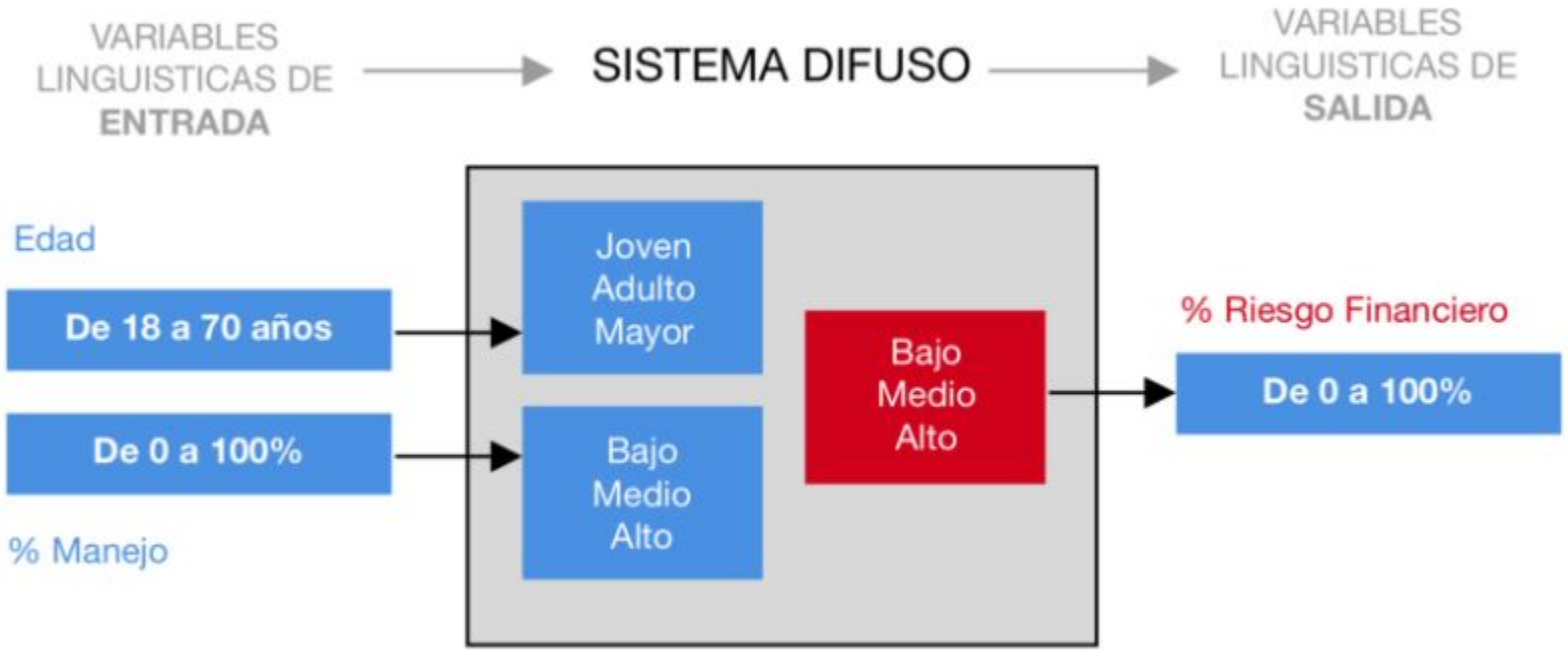
La entrada de este algoritmo es un gráfico monopartito. El gráfico no necesita estar conectado, de hecho, se ignorarán las relaciones existentes entre los nodos. Se crean nuevas relaciones entre cada nodo y sus k vecinos más cercanos.

El algoritmo de K-Neighbours más cercanos compara una propiedad dada de cada nodo. Los knodos donde esta propiedad es más similar son los k vecinos más cercanos.

Evidencias.

1. Aplicacion de la Logica Difusa.

Desarrollo del sistema Difuso.



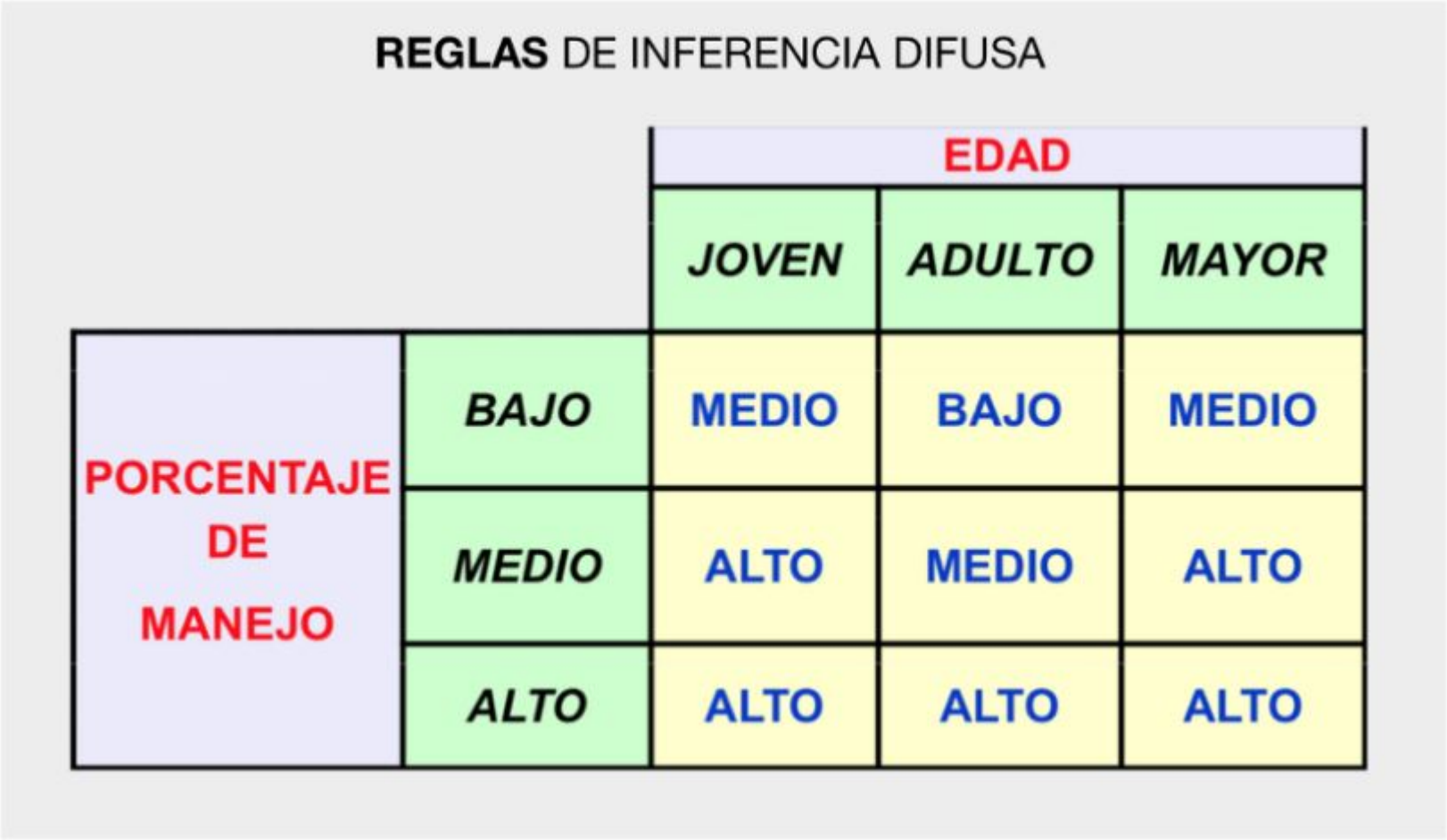
Variables de Entrada → **Sistema Difuso** → Variables de Salida

Codigo para realizar el calculo.

In [ ]:

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
import matplotlib.pyplot as plt
from random import randint, uniform,random

edad = ctrl.Antecedent(np.arange(18, 71, 1), 'edad')
manejo = ctrl.Antecedent(np.arange(0, 101, 1), 'manejo')
riesgo = ctrl.Consequent(np.arange(0, 101, 1), 'riesgo')

riesgo['bajo'] = fuzz.trimf(riesgo.universe, [0, 10, 20])
riesgo['medio'] = fuzz.trimf(riesgo.universe, [10, 30, 45])
riesgo['alto'] = fuzz.trimf(riesgo.universe, [40, 55, 100])

manejo['bajo'] = fuzz.trimf(manejo.universe, [0, 10, 20])
manejo['medio'] = fuzz.trimf(manejo.universe, [10, 40, 60])
manejo['alto'] = fuzz.trimf(manejo.universe, [50, 70, 100])

edad['joven'] = fuzz.trimf(edad.universe, [18, 25, 30])
edad['adulto'] = fuzz.trimf(edad.universe, [20, 35, 50])
edad['mayor'] = fuzz.trimf(edad.universe, [40, 60, 70])
```

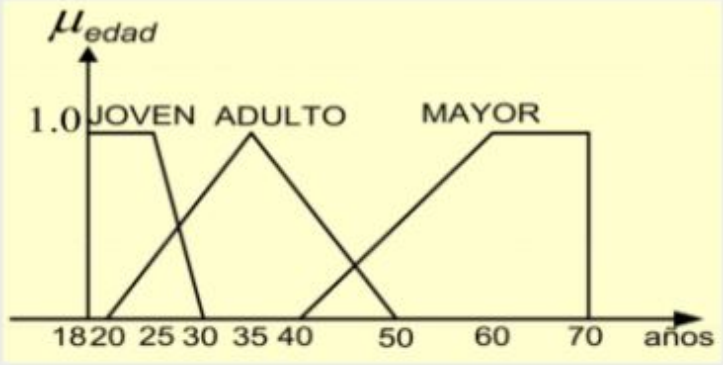
Reglas.



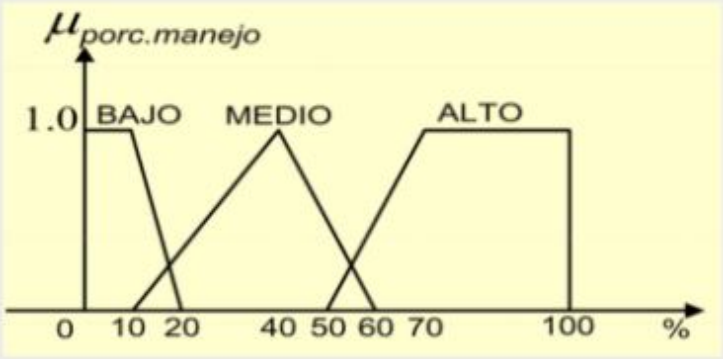
Rangos para realizar el calculo.

FUNCIONES DE PERTENENCIA DE
ENTRADA

Edad

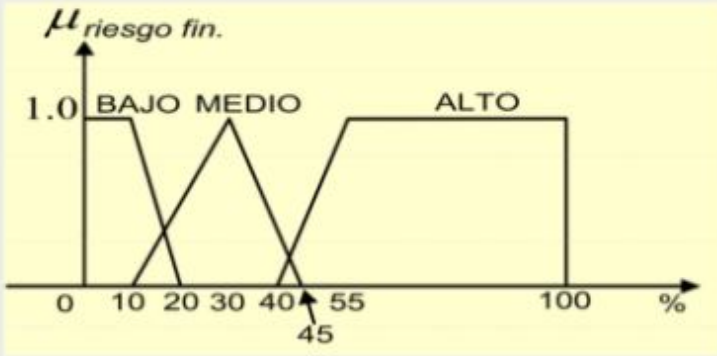


% Manejo

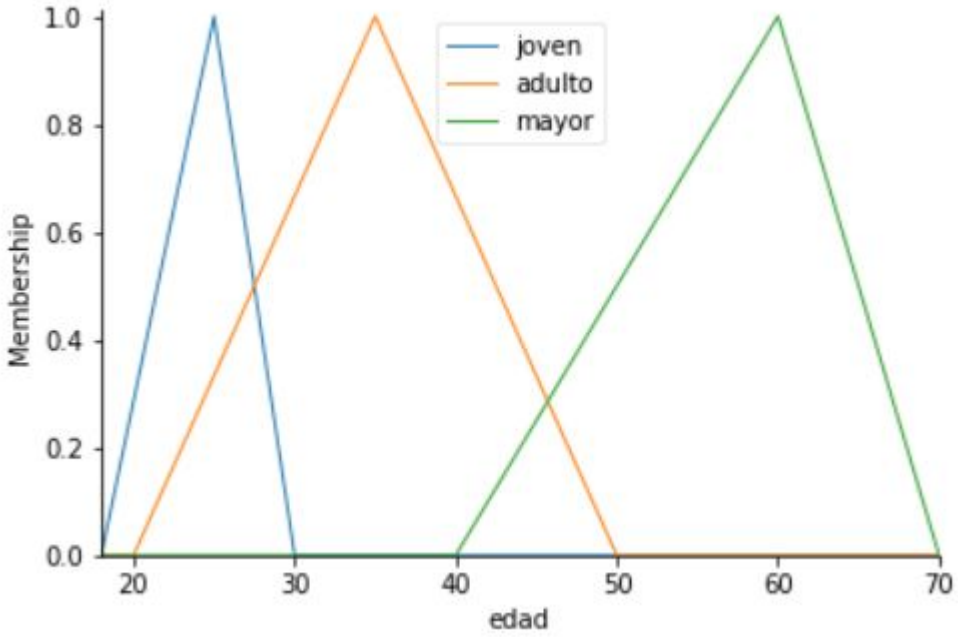


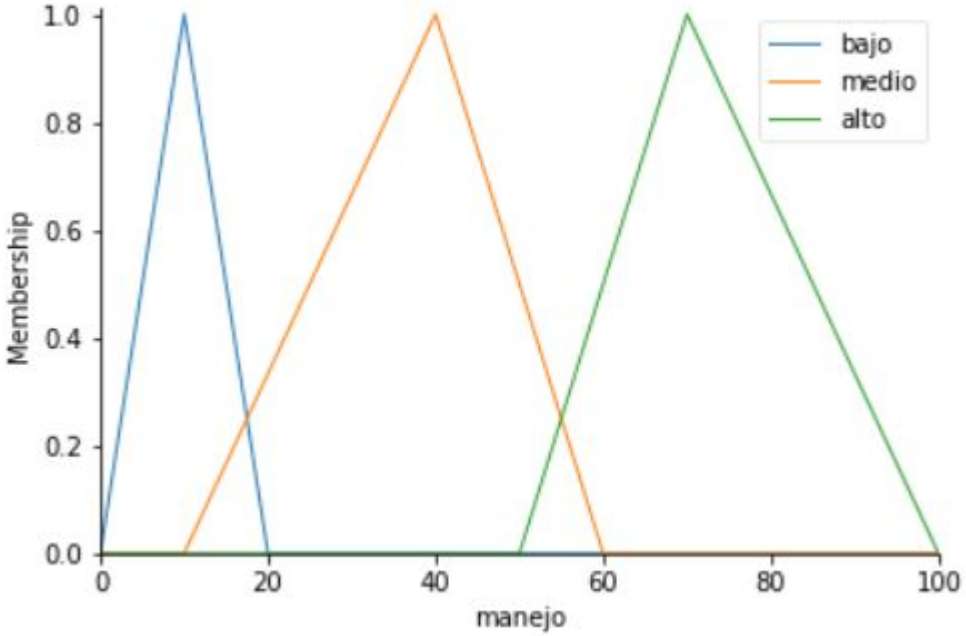
FUNCIONES DE PERTENENCIA DE
SALIDA

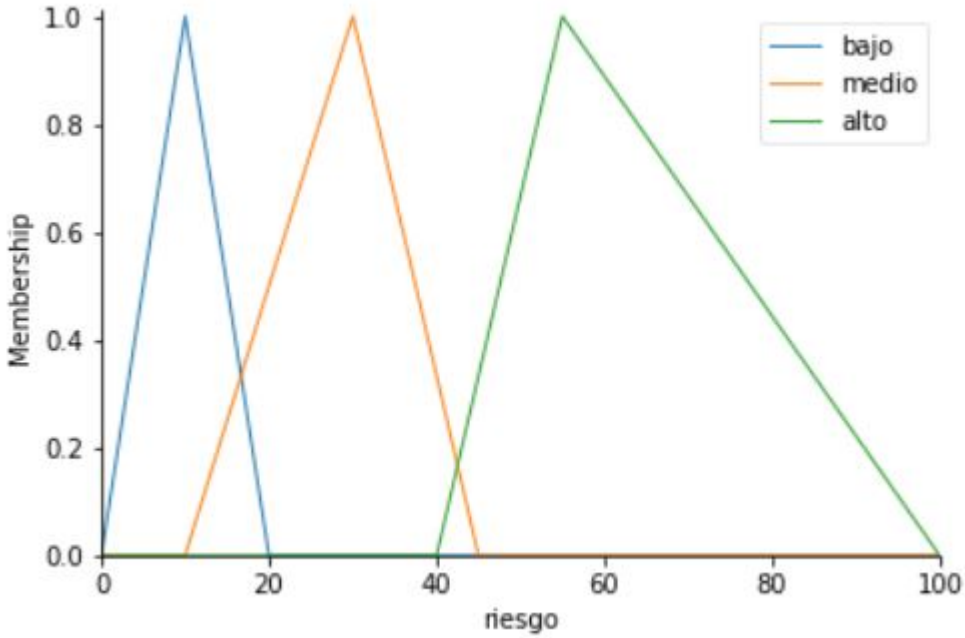
% Riesgo Financiero



Rangos establecidos mediante la libreria.


joven
adulto
mayor
Membership
edad
1.0
0.8
0.6
0.4
0.2
0.0
20
30
40
50
60
70


bajo
medio
alto
Membership
manejo
1.0
0.8
0.6
0.4
0.2
0.0
0
20
40
60
80
100



Creacion de las reglas.

In [ ]:

```
regla1 = ctrl.Rule(manejo['bajo'] and edad['joven'], riesgo['medio'])
regla2 = ctrl.Rule(manejo['medio'] and edad['joven'], riesgo['alto'])
regla3 = ctrl.Rule(manejo['alto'] and edad['joven'], riesgo['alto'])

regla4 = ctrl.Rule(manejo['bajo'] and edad['adulto'], riesgo['bajo'])
regla5 = ctrl.Rule(manejo['medio'] and edad['adulto'], riesgo['medio'])
regla6 = ctrl.Rule(manejo['alto'] and edad['adulto'], riesgo['alto'])

regla7 = ctrl.Rule(manejo['bajo'] and edad['mayor'], riesgo['medio'])
regla8 = ctrl.Rule(manejo['medio'] and edad['mayor'], riesgo['alto'])
regla9 = ctrl.Rule(manejo['alto'] and edad['mayor'], riesgo['alto'])

regla10 = ctrl.Rule(edad['joven'] and manejo['bajo'], riesgo['medio'])
regla11 = ctrl.Rule(edad['joven'] and manejo['medio'], riesgo['alto'])
regla12 = ctrl.Rule(edad['joven'] and manejo['alto'], riesgo['alto'])

regla13 = ctrl.Rule(edad['adulto'] and manejo['bajo'], riesgo['bajo'])
regla14 = ctrl.Rule(edad['adulto'] and manejo['medio'], riesgo['medio'])
regla15 = ctrl.Rule(edad['adulto'] and manejo['alto'], riesgo['alto'])

regla16 = ctrl.Rule(edad['mayor'] and manejo['bajo'], riesgo['medio'])
regla17 = ctrl.Rule(edad['mayor'] and manejo['medio'], riesgo['alto'])
regla18 = ctrl.Rule(edad['mayor'] and manejo['alto'], riesgo['alto'])
```

Creancion de las 100 personas y aplicaion de la logica difusa.

In [ ]:

```
p=1
neo4j = Neo4jService('bolt://localhost:7687', 'neo4j', 'expertos')
with neo4j._driver.session() as session:
    for i in (np.arange(0, 100, 1)):
        nom="persona"+str(p)
        riesgos.input['edad'] = randint(18,70)
        riesgos.input['manejo'] = randint(1,100)
        riesgos.compute()
        s=float(riesgos.output['riesgo'])
        session.write_transaction(neo4j.crear_nodo , nom,s)
        p=p+1
```

Match de los Nodos.

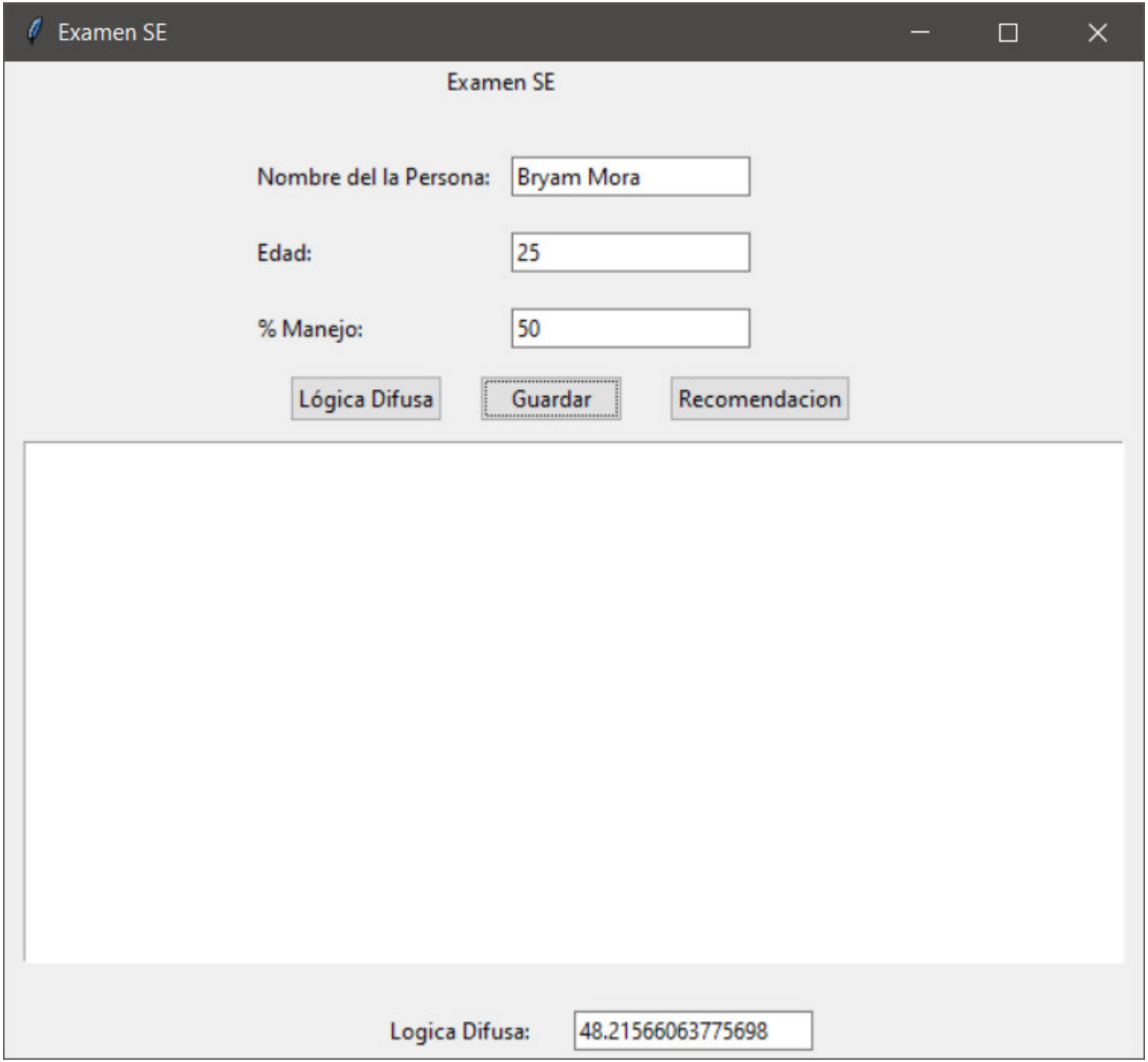
Creacion de los Nodos Peliculas.

In [ ]:

```
neo4j = Neo4jService('bolt://localhost:7687', 'neo4j', 'expertos')
with neo4j._driver.session() as session:
    for i in peliculas:
        peli=i
        riesgo = float(randint(1,100))
        session.write_transaction(neo4j.crear_pelicula , peli,riesgo)
```

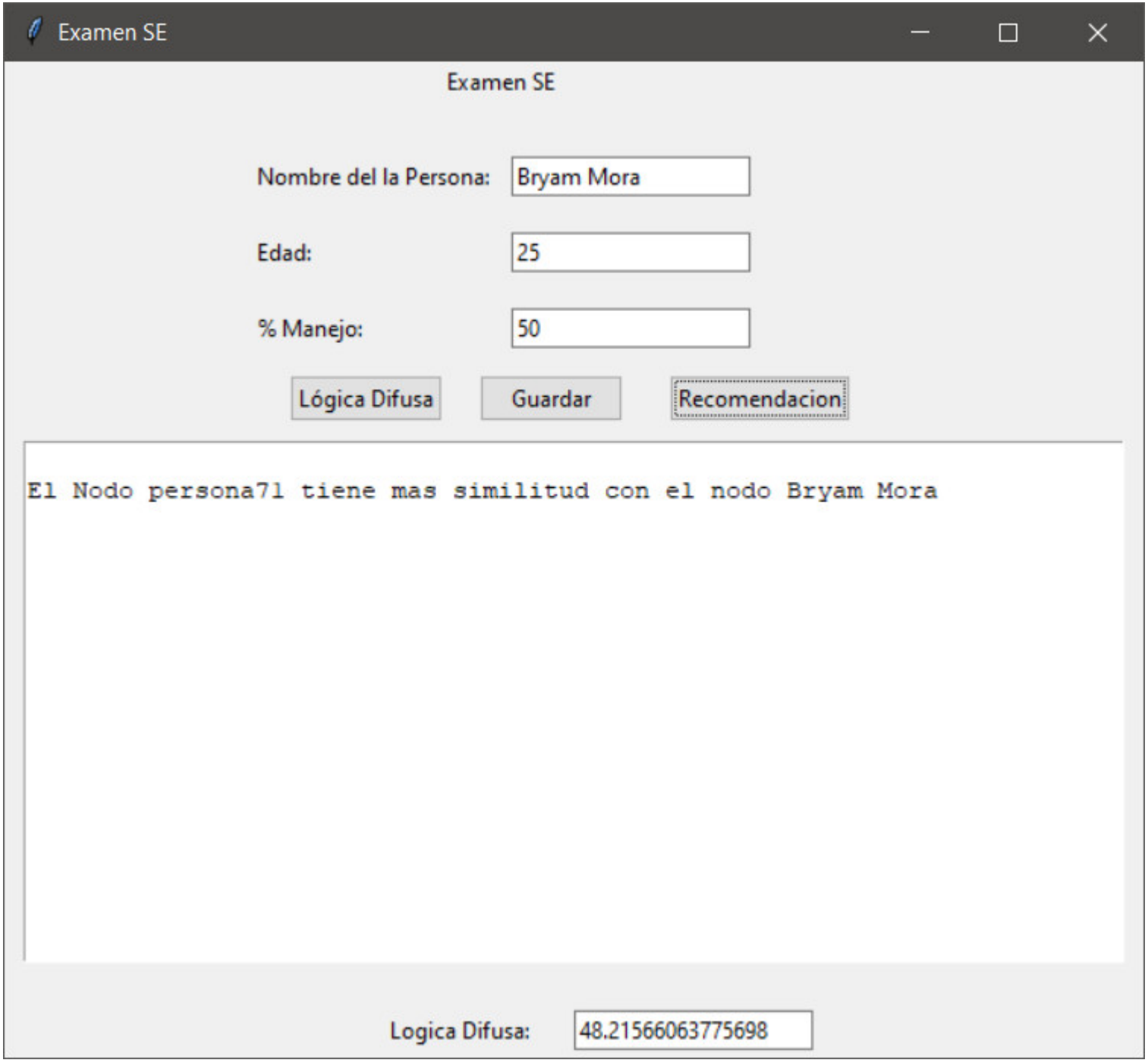
Match de los Nodos.

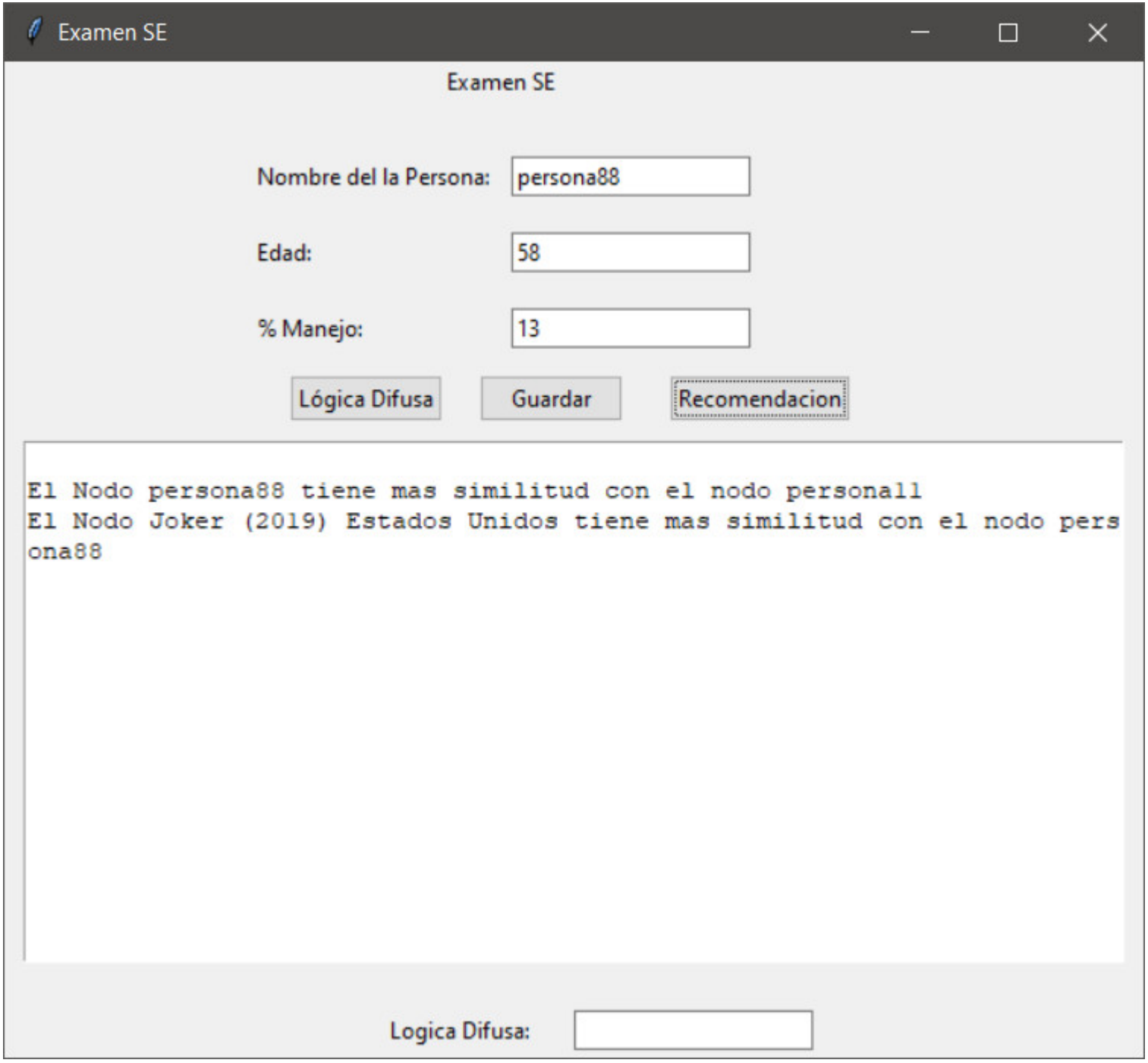
Interfaz Grafica.


Examen SE
Examen SE
Nombre del la Persona:
Bryam Mora
Edad:
25
% Manejo:
50
Lógica Difusa
Guardar
Recomendacion
Logica Difusa:
48.21566063775698

Match de los Nuevo Nodo.

Similitud de los Nodos.


Examen SE
Examen SE
Nombre del la Persona:
Bryam Mora
Edad:
25
% Manejo:
50
Lógica Difusa
Guardar
Recomendacion
El Nodo persona71 tiene mas similitud con el nodo Bryam Mora
Logica Difusa:
48.21566063775698


Examen SE
Examen SE
Nombre del la Persona:
persona88
Edad:
58
% Manejo:
13
Lógica Difusa
Guardar
Recomendacion
El Nodo persona88 tiene mas similitud con el nodo persona11
El Nodo Joker (2019) Estados Unidos tiene mas similitud con el nodo pers
ona88
Logica Difusa: