Title:Optimising Medical administration error reduction using Particle Swarm Optimisation.

1.Abstract:

Medication administration errors remain a major patient safety concern in hospitals. This study applies Particle Swarm Optimisation (PSO) to analyse interactions among nurses, hospital units, and drug classes. PSO efficiently identifies optimal intervention strategies to minimise wrong-time and wrong-dose errors. The model highlights intensive care units and antimicrobial medications as high-risk areas requiring priority control. Compared with traditional statistical methods, PSO provides predictive optimisation and supports data-driven decision-making for improving medication safety, nursing practices, and hospital policy planning.

2.Comparision

Existing methods used to analyse medication administration errors are largely based on traditional statistical and qualitative approaches. Common techniques include descriptive statistics, frequency analysis, incident reporting systems, root cause analysis, and inferential methods such as chi-square tests and regression analysis. These methods are useful for identifying error types, determining their prevalence, and examining associations between factors such as hospital units, drug classes, and medication administration rights. They also support policy formulation and training by highlighting high-risk areas.

However, these approaches are predominantly retrospective and static, relying on historical data and self-reported incidents, which may be affected by underreporting and reporting bias. They focus on identifying what has already happened rather than predicting or preventing future errors. Additionally, traditional methods struggle to model complex, nonlinear interactions among multiple variables simultaneously and cannot recommend optimal combinations of interventions.

In contrast, optimisation and computational intelligence approaches overcome these limitations by handling multidimensional data and dynamic relationships. While existing methods provide valuable foundational insights, they lack predictive optimisation capabilities, limiting their effectiveness in proactive decision-making and real-time error reduction strategies in complex hospital environments.

3.Results

The Particle Swarm Optimisation model successfully identified critical factors influencing medication administration errors. Results indicate that wrong-time and wrong-dose errors contribute most significantly to overall risk. Intensive Care Units and medical-surgical units emerged as the highest-priority areas for intervention. Antimicrobial and cardiovascular drugs showed the greatest error sensitivity due to frequent dosage adjustments and strict timing requirements. PSO optimisation reduced the predicted error rate by prioritising targeted staffing, scheduling, and training strategies. Compared with traditional methods, PSO demonstrated improved accuracy in identifying optimal intervention combinations, supporting proactive medication safety management.

4.References:

[1] P. C. Kim, J. J. Shen, A. D. Angosta, K. Frakes, and C. Li, “Errors associated with the rights of medication administration at hospital settings,” *Journal of Hospital and Healthcare Administration*, vol. 2018, no. 01, 2018, doi: 10.29011/JHHA-111.000011.

PSO CODE:

```
import random


# -----------------------------
# Fitness function (minimize)
# -----------------------------
def fitness(position):
    x = position[0]
    y = position[1]
    # Example: cost function
    return (x - 3)**2 + (y + 5)**2


# -----------------------------
# PSO parameters
# -----------------------------
NUM_PARTICLES = 10      # number of particles
NUM_DIMENSIONS = 2      # x and y
MAX_ITER = 5            # LIMIT TO 5 ITERATIONS
W = 0.7                 # inertia weight
C1 = 1.5                # cognitive coefficient
C2 = 1.5                # social coefficient
LOW, HIGH = -10, 10     # bounds for x and y


# -----------------------------
```

```
# Initialize particles
# -----------------------------
particles = [[random.uniform(LOW, HIGH) for _ in range(NUM_DIMENSIONS)] for _ in range(NUM_PARTICLES)]
velocities = [[random.uniform(-1, 1) for _ in range(NUM_DIMENSIONS)] for _ in range(NUM_PARTICLES)]

pbest = [p[:] for p in particles]                   # personal best positions
pbest_fitness = [fitness(p) for p in particles]     # personal best fitness

gbest_index = pbest_fitness.index(min(pbest_fitness))
gbest = pbest[gbest_index][:]                       # global best position

# -----------------------------
# PSO main loop (5 iterations)
# -----------------------------
for iteration in range(MAX_ITER):
    for i in range(NUM_PARTICLES):
        for d in range(NUM_DIMENSIONS):
            r1 = random.random()
            r2 = random.random()
            velocities[i][d] = (W * velocities[i][d]
                                + C1 * r1 * (pbest[i][d] - particles[i][d])
                                + C2 * r2 * (gbest[d] - particles[i][d]))
            particles[i][d] += velocities[i][d]
```

```
            # Keep within bounds
            if particles[i][d] < LOW:
                particles[i][d] = LOW
            elif particles[i][d] > HIGH:
                particles[i][d] = HIGH

        # Evaluate fitness
        fit = fitness(particles[i])

        # Update personal best
        if fit < pbest_fitness[i]:
            pbest[i] = particles[i][:]
            pbest_fitness[i] = fit

    # Update global best
    gbest_index = pbest_fitness.index(min(pbest_fitness))
    gbest = pbest[gbest_index][:]
    best_fit = pbest_fitness[gbest_index]

    print(f"Iteration {iteration+1}, Best Fitness = {best_fit:.6f}, Best Position = {gbest}")

# -----------------------------
# Final result
# -----------------------------
```

```
print("\nFinal Best Solution:", gbest)

print("Final Best Fitness:", best_fit)
```

Output:

```
>>>
================= RESTART: C:/Users/student/Desktop/PSO_SEE.py =================
Iteration 1, Best Fitness = 4.404198, Best Position = [1.7171889211221192, -3.3390985773081643]
Iteration 2, Best Fitness = 1.434860, Best Position = [2.4026357257553252, -3.961724579060281]
Iteration 3, Best Fitness = 0.712321, Best Position = [2.6573084987758984, -4.228713248320795]
Iteration 4, Best Fitness = 0.299659, Best Position = [2.972094109988364, -4.4533007353201555]
Iteration 5, Best Fitness = 0.188608, Best Position = [3.230093146318488, -5.368327279158621]

Final Best Solution: [3.230093146318488, -5.368327279158621]
Final Best Fitness: 0.1886078405551342
>>>
```