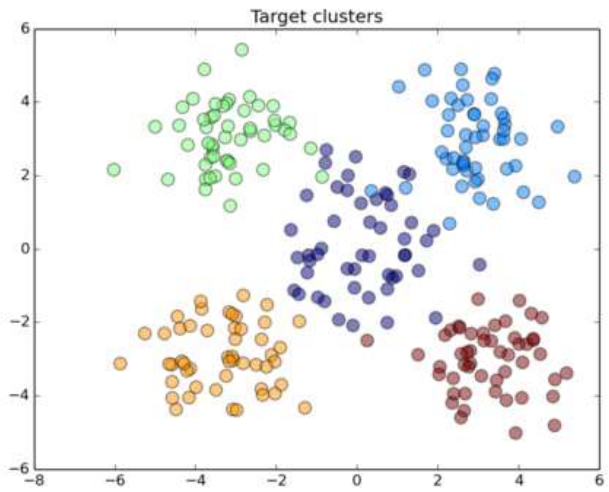
Swarm intelligence

THE PRACTICALITY OF CLASSIFICATION BASED ON BIOLOGICAL SYSTEMS

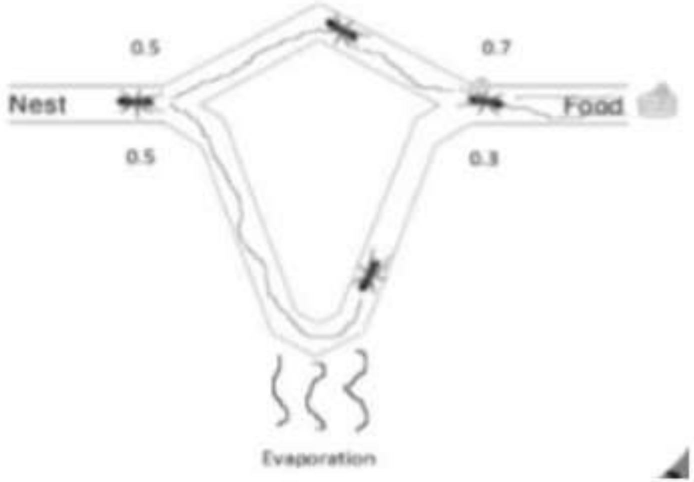
The Clustering Problem

- Clustering – Partitioning datasets into meaningful subclasses
- Unsupervised Learning Problem
- Utilised in many professions from business to medicine
- Clustering is computationally expensive
- Utilises heuristic information, but yet to be a generic solution



Ant Colony Optimisation

- Utilises swarm intelligence for parallelisation
- Each agent is its own entity
- Information passed via pheromone
- Begins with inherent randomness
- Allows for emergent shortest path optimisation
- Randomness is key



Understanding The Equation

- *P denotes the amount of pheromone*
- *i* and *j* denote the nodes for *k* to traverse
- *Pheromone regulation T*
- *Alpha and* Beta *are heuristic values for desirability of* n

$$\rho_{ij}^{k} = \frac{[t_{ij}]^{\alpha} * [n_{ij}]^{\beta}}{\sum l \in N_l^k [t_{il}]^{\alpha} * [n_{il}]^{\beta}}$$

Fig. 3 ACO Solution Equation
(Shekhawat, Poddar & Boswal 2009)

Mathematics To Code

```
for iteration -> n
    set initial population;
    set initial pheromone concentration;
    initialise pheromone matrix;
    calculate initial edge desirability;
    set evaporation rate;

    create_colony;
    calculate_fitness;
    update_elite_fitness;
    update_pheromone_matrix;
end
```

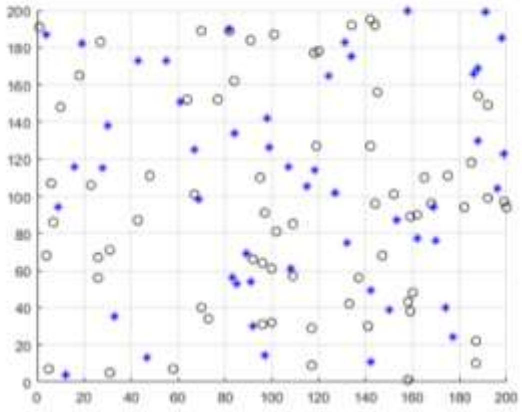
Fig. 6 ACO Compact Pseudocode

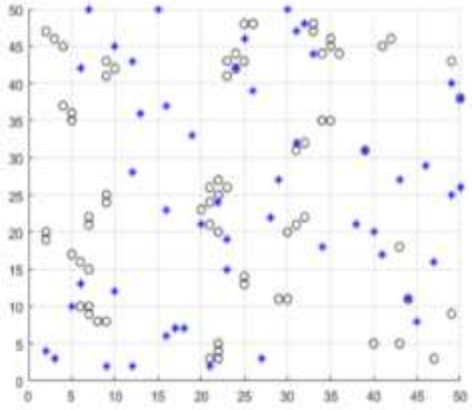
```
% Calc the average of the distances between all edges in graph * #NO edges
% Pheromone concentration
tau0 = 10*1/( graph.n * mean(graph.edges(:) ));

% Create pheramone matrices
tau = tau0 * ones(graph.n, graph.n);

% Edge desirability: shorter is more desirable
% Can be reversed to find the longest path via graph.edges
eta = 1 ./graph.edges;

% Evaporation rate
rho = 0.15;

% Pheromone param
alpha = 1;
% Desirability param
beta = 2;
```

Ant Clustering

- Clustering using the ant algorithm begins with population
- The ants and food sources are added to a graph space
- The algorithm utilises random wanderers to discover nodes
- Using the Euclidean distance clusters are determined
- Pheromones remember the utilised paths
- Takes a long time to converge
- Positional values tracked via matrices

0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000
0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000	0.1000

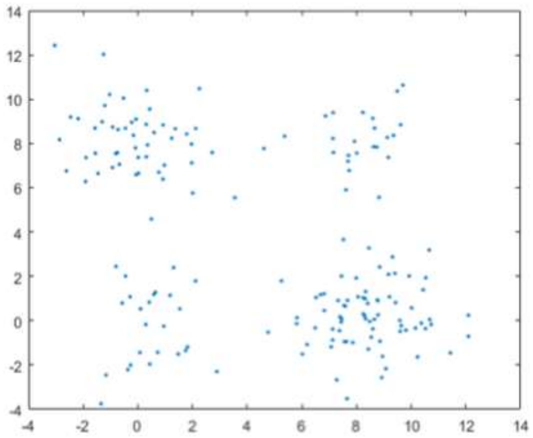


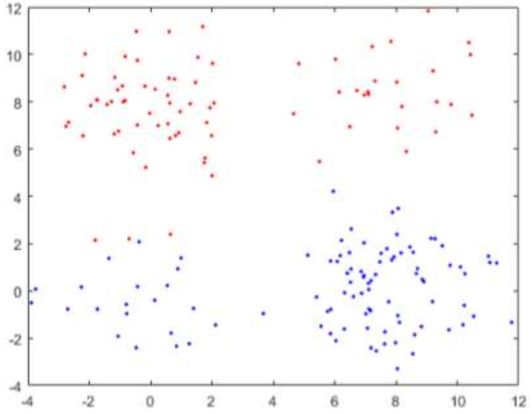


K-means

- Offers Faster Convergence
- Utilises centroid based clustering
- Point k selected
- Nodes (x) calculated distance in regards of *k* and *x*
- Node assignment to cluster *j*

- Not a biological algorithm or generic solution
- Possibility of a hybrid k-means



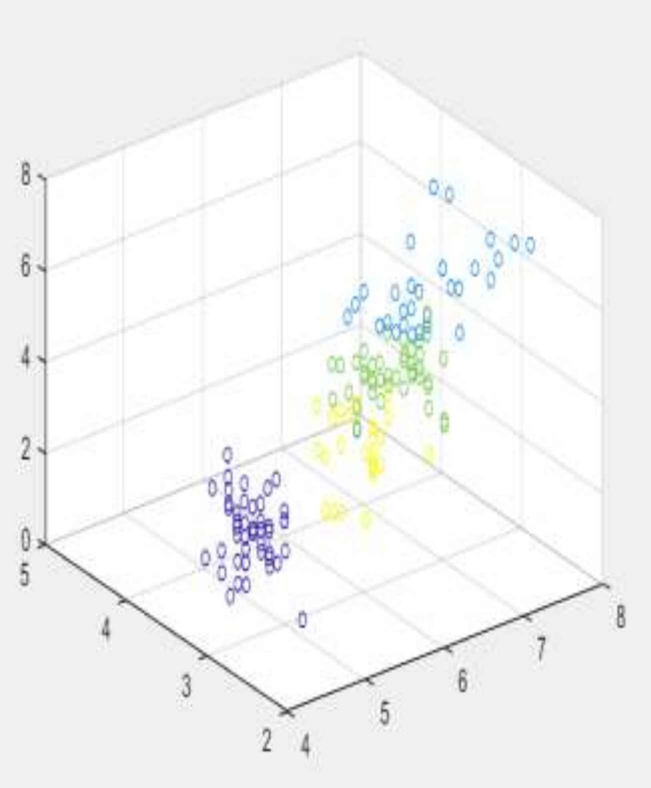


K-means Example

Ant Centroid Clustering

- Superior convergence to Ant Colony
- Utilises ACO methodologies
- Applies K-means style centroids via fitness

- Outperformed by K-means
- Similar Results to K-means
- Some datasets have drastic fluctuations between iterations



Ant Clustering1

```
%% Main ACC

generation_limit = no_generations;

% Set Parameters
dataset_items = size(dataset, 1);
features = size(dataset, 2);

clusters = 2;
population = 10;

eta = .1;
beta = .1;
alpha = .1;
delta = 1;

pd = .9;

% Evaporation Rate
rho = 0.5;
tau0 = ones(dataset_items, clusters) * 0.001;
tau = [];
lsr = 0.2;

% Best Solutions
elite = [];
global_elite = [];
total_elite = round(population*0.2);

% fitness
best_fitness = zeros(generation_limit,1);
current_best_fitness = zeros(generation_limit,1);
```

```
while i <= generation_limit
    tau = tau0./repmat(sum(tau,2),1,clusters);
    tours = zeros(population, dataset_items+delta);
    best_tour = [];

    % Begin Tours
    for ant = 1 : population
        for item = 1 : dataset_items

            r = rand();
            if r > pd
                r2 = rand();
                c = 0;
                t_size = sort(tau(item,:));

                for j = 1: t_size
                    c = c + t_size(j);
                    if r2 < c
                        disc = find(tau(item,:) == t_size(j));
                        % Update Tours
                        if size(disc,2) == 1
                            tours(ant,item) = disc;
                        else
                             tours(ant,item) = disc9floor(rand()*size(disc(2)+delta));
                        end
                        break;
                    end
                end
```

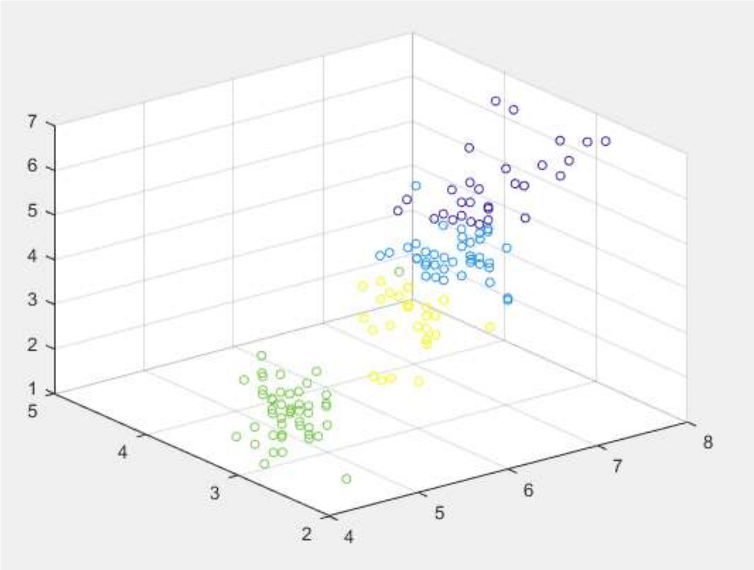
Ant Clustering 2

```
    else
        % Explore Pheromone Trails
        disc = find(tau(item,:) == max(tau(item,:)));
        if size(disc, 2) == 1
            tours(ant,item) = disc;
        else
            tours(ant,item) = disc(floor(rnad()*size(disc,2)+delta));
        end
    end
end
```

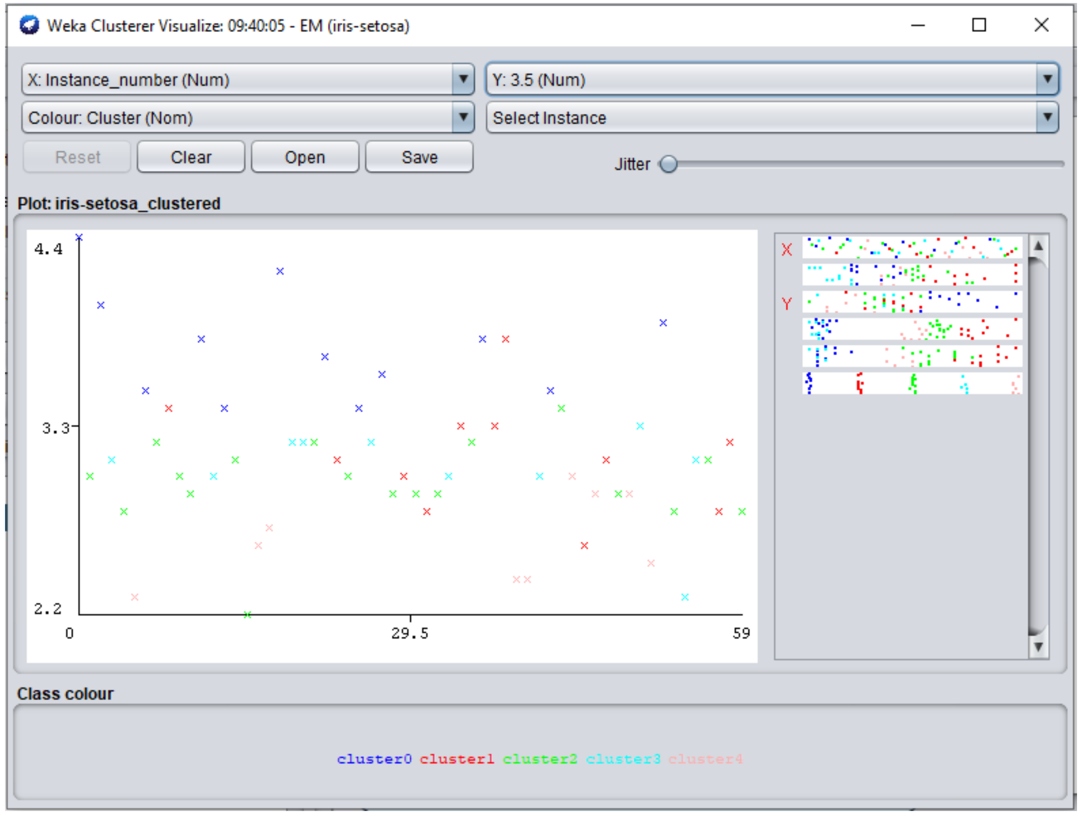
```
    % Calc Fitness with Centroid
    centroid = (weighted_average'*dataset_items)./repmat(sum(weighted_avg,1)',1,features);
    for item = 1 : dataset_items
        ed = sqrt(sum((dataset_items(item, :) - centroid(solutions(ant,item),:)).^2));
        tours(ant,end) = tours(ant,end)+ed;
    end
end
```

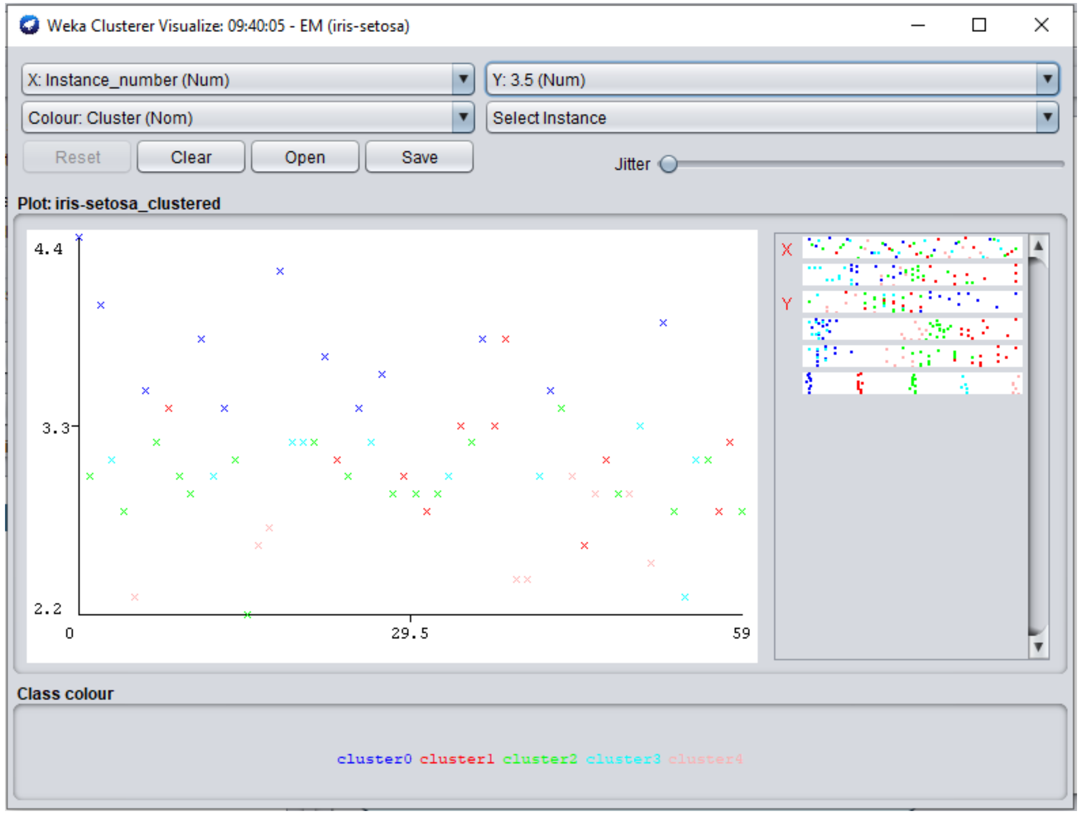
Ant Clustering 3

- Iterates over the dataset
- Creates path solutions utilising ACO
- Explore Pheromone Trails
- Sets a weighted average dependent on tour data
- Randomly selects centroids within data
- Partitions the data calculating the Euclidean distance



Weka Iris-setosa





Datasets

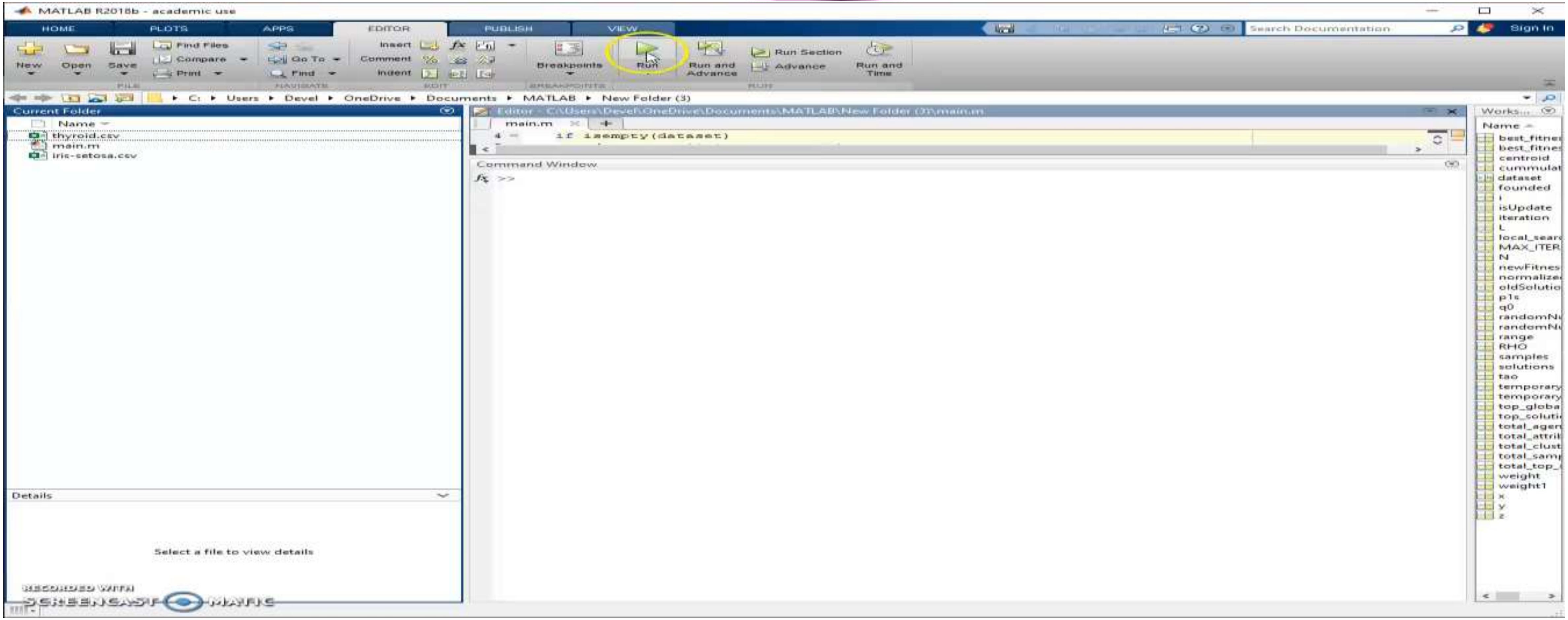
Iris-Setosa

	A	B	C	D
1	5.1	3.5	1.4	0.2
2	4.9	3	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5	3.6	1.4	0.2
6	5.4	3.9	1.7	0.4
7	4.6	3.4	1.4	0.3
8	5	3.4	1.5	0.2
9	4.4	2.9	1.4	0.2
10	4.9	3.1	1.5	0.1
11	5.4	3.7	1.5	0.2
12	4.8	3.4	1.6	0.2
13	4.8	3	1.4	0.1
14	4.3	3	1.1	0.1
15	5.8	4	1.2	0.2
16	5.7	4.4	1.5	0.4
17	5.4	3.9	1.3	0.4
18	5.1	3.5	1.4	0.3
19	5.7	3.8	1.7	0.3
20	5.1	3.8	1.5	0.3
21	5.4	3.4	1.7	0.2
22	5.1	3.7	1.5	0.4
23	4.6	3.6	1	0.2
24	5.1	3.3	1.7	0.5
25	4.8	3.4	1.9	0.2
26	5	3	1.6	0.2
27	5	3.4	1.6	0.4
28	5.2	3.5	1.5	0.2
29	5.2	3.4	1.4	0.2
30	4.7	3.2	1.6	0.2
31	4.8	3.1	1.6	0.2
32	5.4	3.4	1.5	0.4
33	5.2	4.1	1.5	0.1
34	5.5	4.2	1.4	0.2

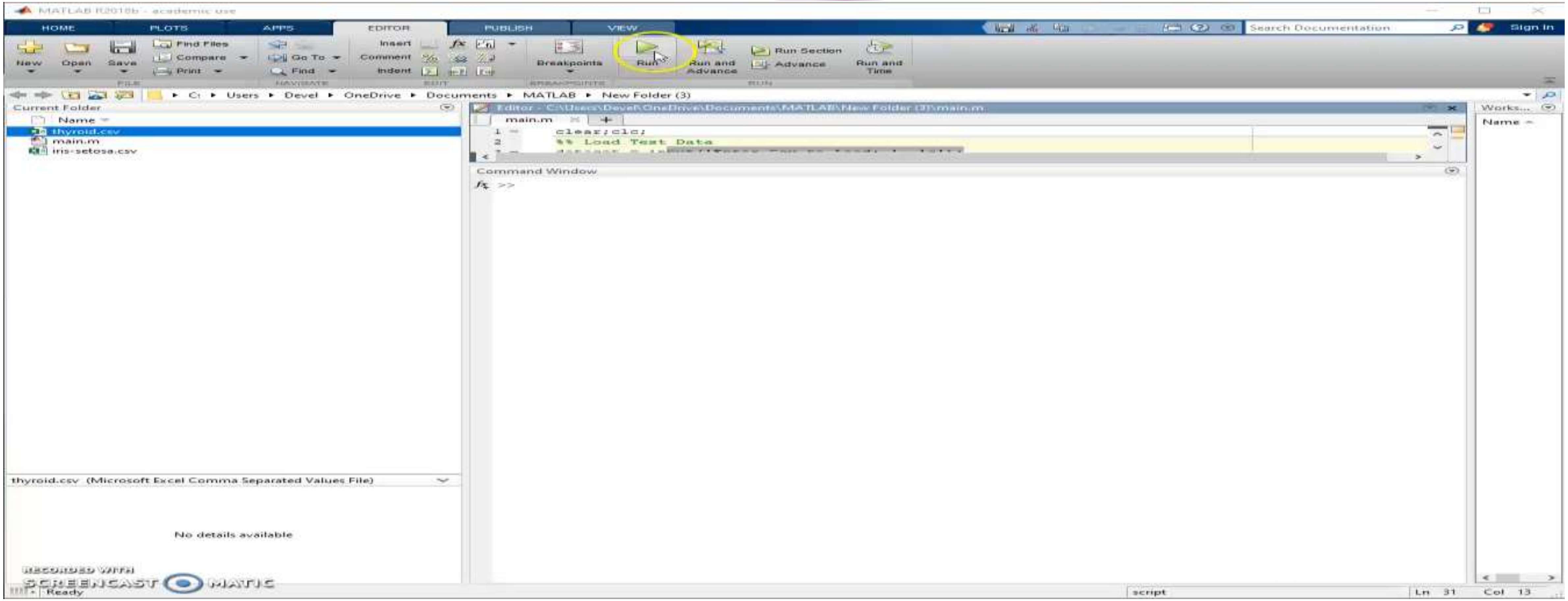
Thyroid

	A	B	C	D	E
1	107	10.1	2.2	0.9	2.7
2	113	9.9	3.1	2	5.9
3	127	12.9	2.4	1.4	0.6
4	109	5.3	1.6	1.4	1.5
5	105	7.3	1.5	1.5	-0.1
6	105	6.1	2.1	1.4	7
7	110	10.4	1.6	1.6	2.7
8	114	9.9	2.4	1.5	5.7
9	106	9.4	2.2	1.5	0
10	107	13	1.1	0.9	3.1
11	106	4.2	1.2	1.6	1.4
12	110	11.3	2.3	0.9	3.3
13	116	9.2	2.7	1	4.2
14	112	8.1	1.9	3.7	2
15	122	9.7	1.6	0.9	2.2
16	109	8.4	2.1	1.1	3.6
17	111	8.4	1.5	0.8	1.2
18	114	6.7	1.5	1	3.5
19	119	10.6	2.1	1.3	1.1
20	115	7.1	1.3	1.3	2
21	101	7.8	1.2	1	1.7
22	103	10.1	1.3	0.7	0.1
23	109	10.4	1.9	0.4	-0.1
24	102	7.6	1.8	2	2.5
25	121	10.1	1.7	1.3	0.1
26	100	6.1	2.4	1.8	3.8
27	106	9.6	2.4	1	1.3
28	116	10.1	2.2	1.6	0.8
29	105	11.1	2	1	1
30	110	10.4	1.8	1	2.3
31	120	8.4	1.1	1.4	1.4
32	116	11.1	2	1.2	2.3
33	110	7.8	1.9	2.1	6.4
34	90	8.1	1.6	1.4	1.1

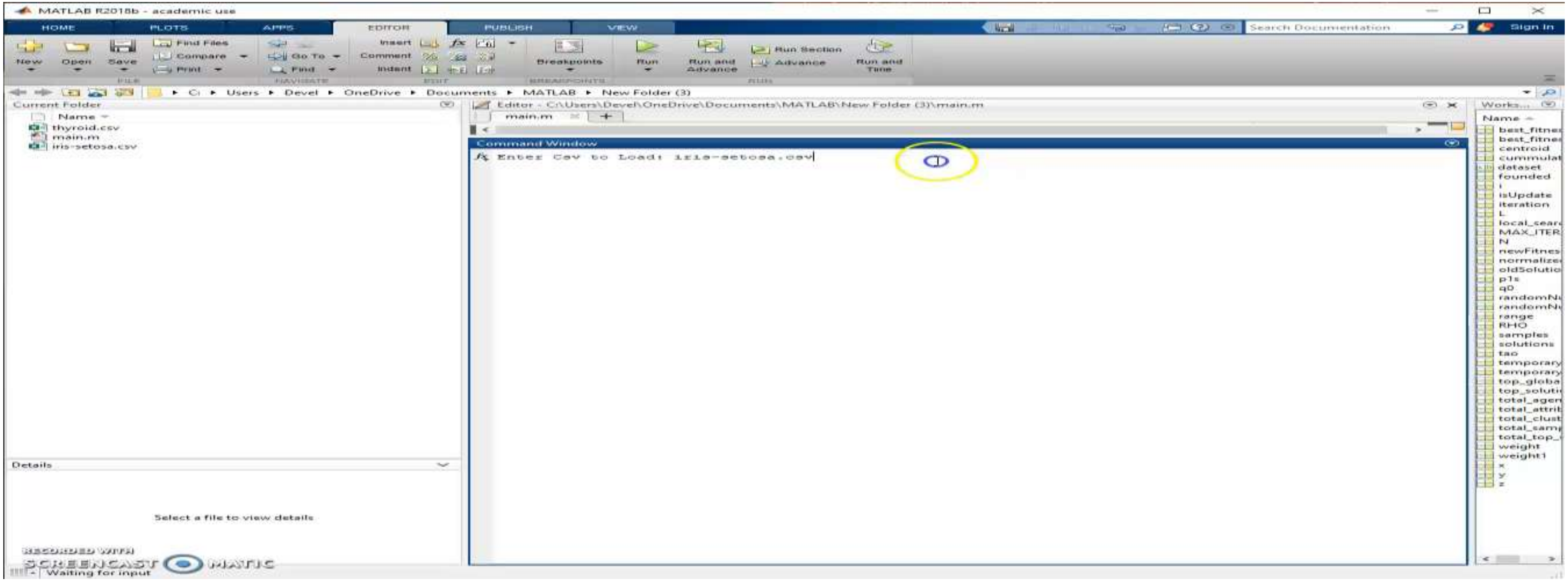
Iris-Setosa: Ants 20 Generation 100



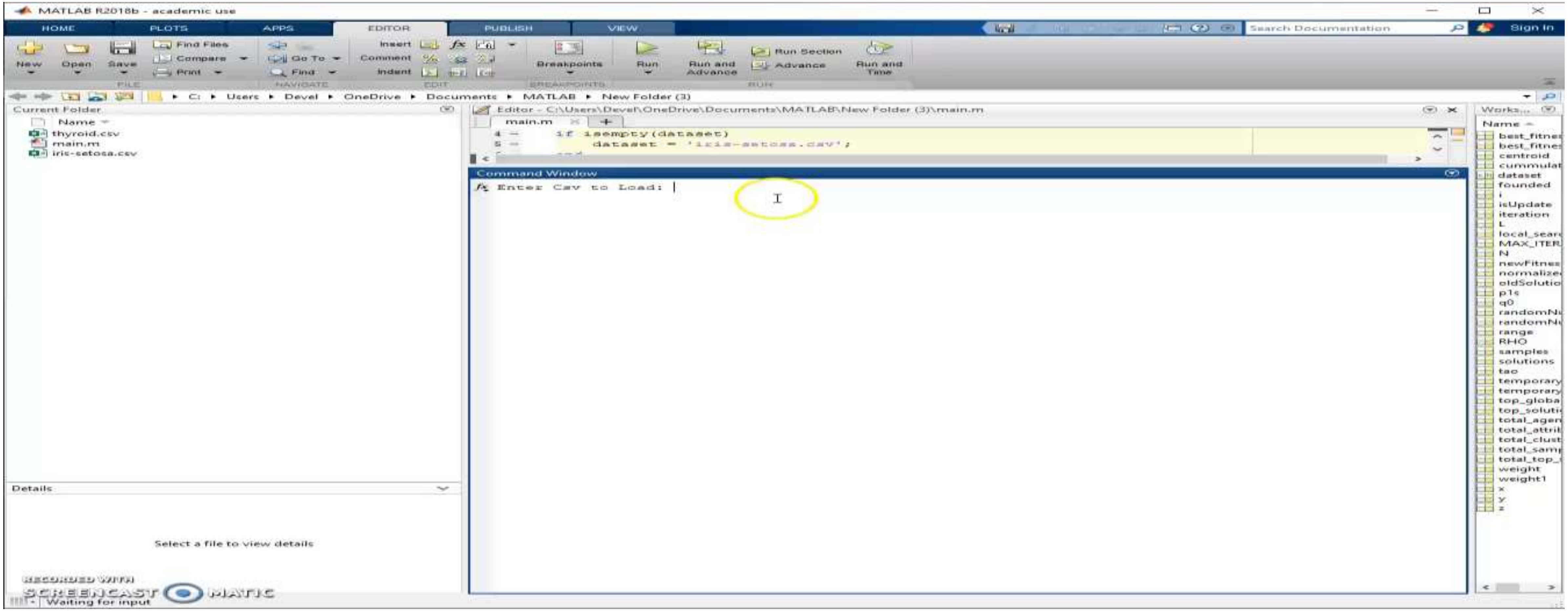
Iris-Setosa: 100 Ants Gen 500



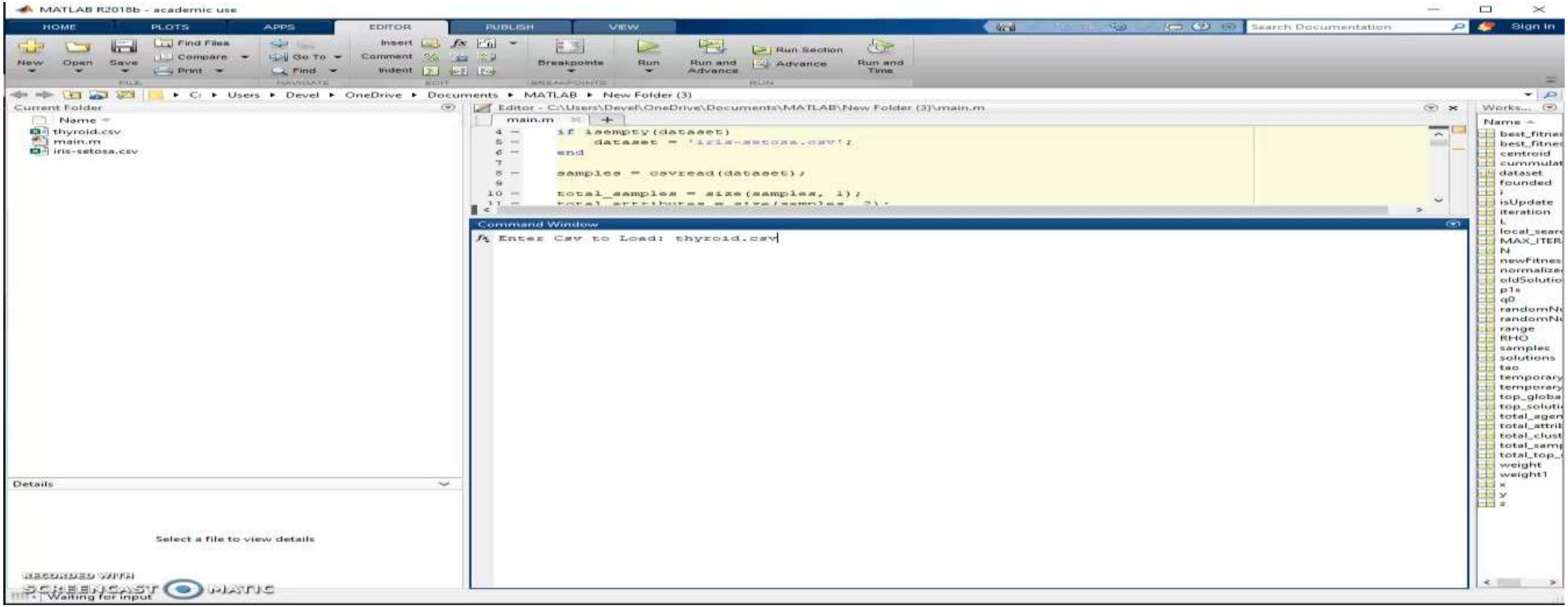
Iris-Setosa: Ants 200 Gen 2000: 372907



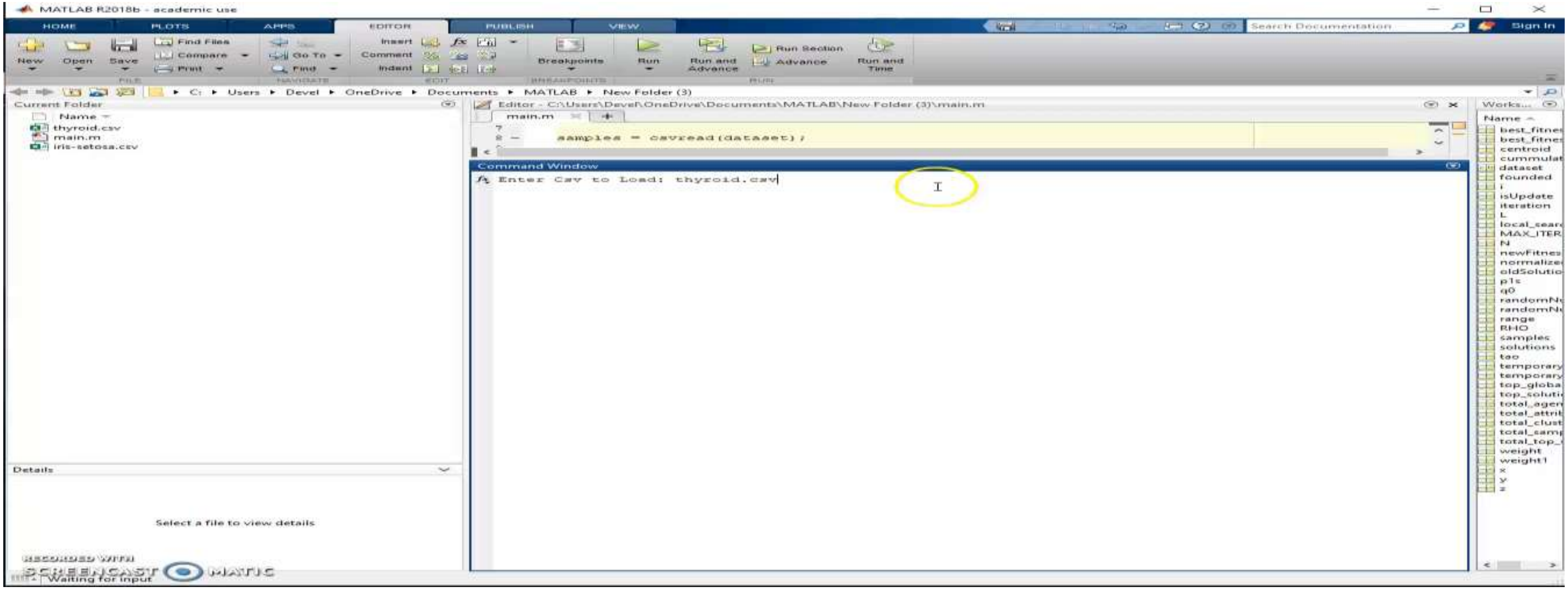
Thyroid: Ants 10 gen 500



Thyroid: Ants 100 Gen 1000



Thyroid: Ants 25 Gen 100



Fuzzy Logic

- Based on natural thinking processes
- Allows for definitive circumstances with intermediate responses

- C-means is an example of Fuzzy Logic in a clustering algorithm
- Fuzzy Logic could be applied to ACC

- K-means ACO provides adequate solution in reasonable time frame

Next Steps

- Research into Fuzzy Implementation
- Comparison of Fuzzy Logic to the base algorithms
- Improve centroid based solution via Fuzzy Logic

- Continuation of the implementation within java for clustering/feature selection

Conclusion

- K-means remains the best for simplicity and reliability
- Nature inspired algorithm cluster data effectively
- AC is the best for time-constrained solutions

- Fuzzy Logic could improve performance and reliability as seen in c-means
- Alternative solution could be to investigate the use of biological inspired algorithms for training neural networks weighting