AMAZING ALGORITHMS

FOR SOLVING PROBLEMS IN SOFTWARE

BARRY S. STAHL

SOLUTION ARCHITECT & DEVELOPER

@BSSTAHL@COGNITIVEINHERITANCE.COM

HTTPS://COGNITIVEINHERITANCE.COM

FAVORITE PHYSICISTS & MATHEMAT

FAVORITE PHYSICISTS

1. Harold "Hal" Stahl
2. Carl Sagan
3. Richard Feynman
4. Marie Curie
5. Nikola Tesla
6. Albert Einstein
7. Neil Degrasse Tyson
8. Niels Bohr
9. Galileo Galilei
10. Michael Faraday

Other notables: Stephen Hawking, Edwin Hubble

FAVORITE MATHEMATICIANS

1. Ada Lovelace
2. Alan Turing
3. Johannes Kepler
4. Rene Descartes
5. Isaac Newton
6. Leonardo Fibonacci
7. George Boole
8. Blaise Pascal
9. Johann Gauss
10. Grace Hopper

Other notables: Daphne Koller, Grady B Berezin

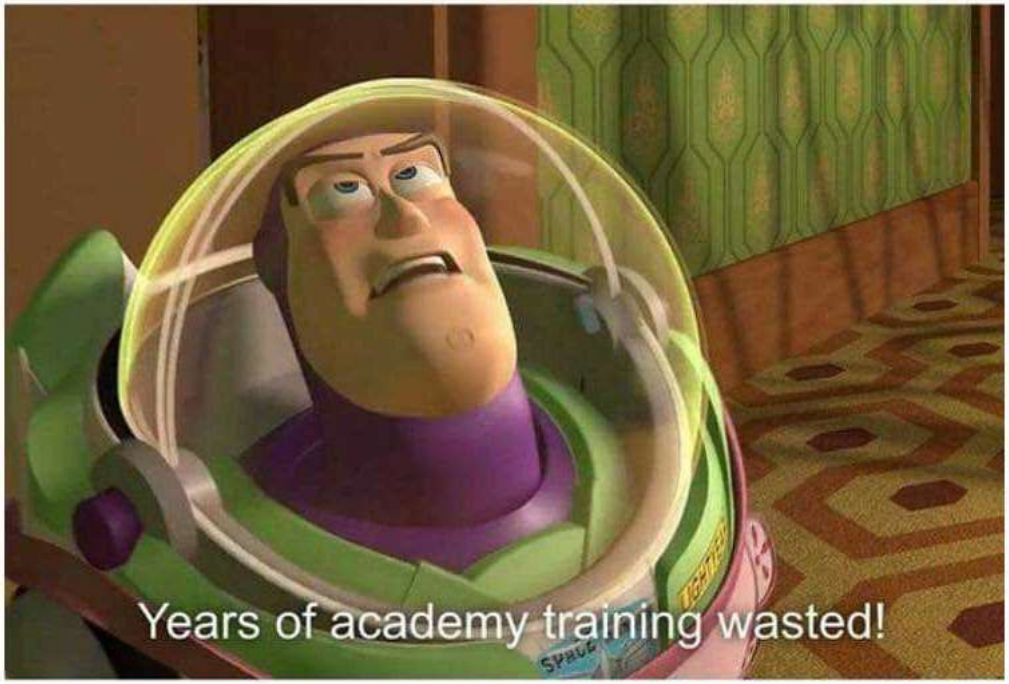
SOME OSS PROJECTS I RUN

1. Liquid Victor : Media tracking and aggregation [used to assemble this presentatio
2. Prehensile Pony-Tail : A static site generator built in c#
3. TestHelperExtensions : A set of extension methods helpful when building unit tes
4. Conference Scheduler : A conference schedule optimizer
5. IntentBot : A microservices framework for creating conversational bots on top of
6. LiquidNun : Library of abstractions and implementations for loosely-coupled appl
7. Toastmasters Agenda : A c# library and website for generating agenda's for Toast
8. ProtoBuf Data Mapper : A c# library for mapping and transforming ProtoBuf mes

HTTP://GIVECAMP.ORG


CELEBRATING BARRY'S 100TH PUBLIC TALK
www.cognitiveinheritance.com
Achievement Unlocked
100G - 100 Public Talks

when you finally get a developer job but they say they don't need you to write sorting algorithms for them because they just use the library functions

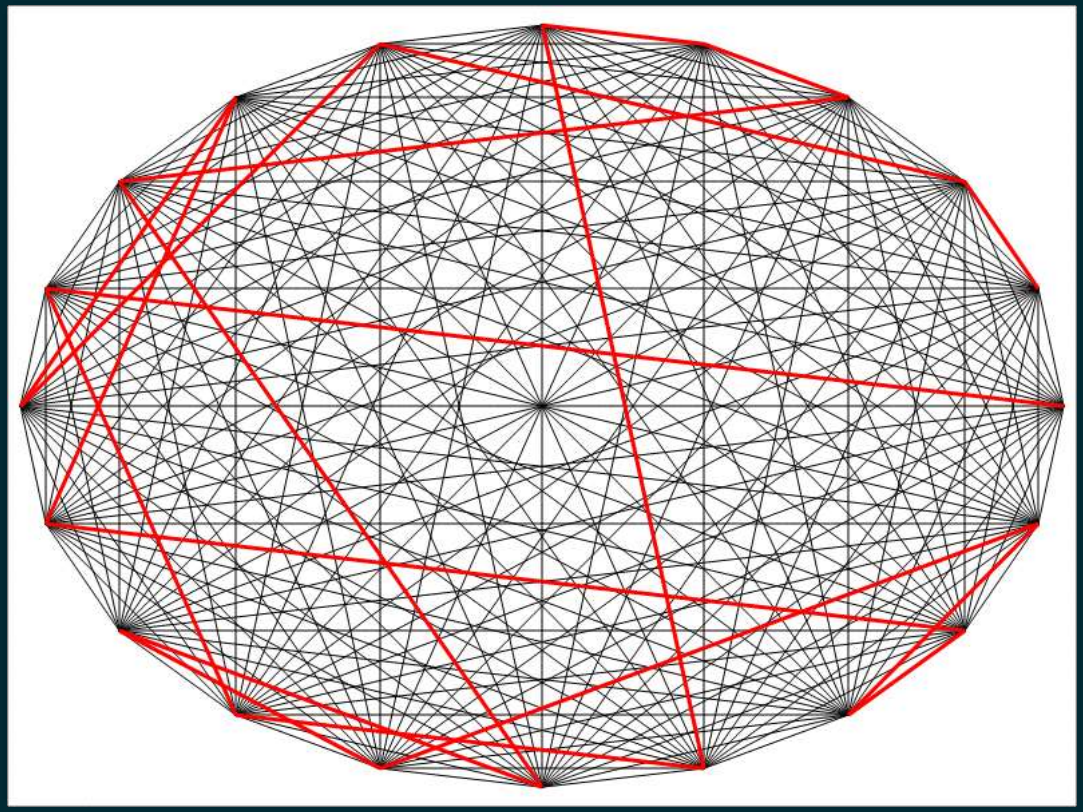


AGENDA

- Best Path Algorithms
  - Bee Colony Optimization
  - Ant Colony Optimization
- Cost Reduction Algorithms
  - Firefly Optimization
  - Amoeba Optimization
- AI Models
  - Intro to Optimizing AI Models
  - Training an AI Model Using Amoebas


BEST PATH ALGORITHMS

BEE COLONY OPTIMIZATION



TYPES OF BEES

Active Workers

Forage on their known path and on a **neighboring** path

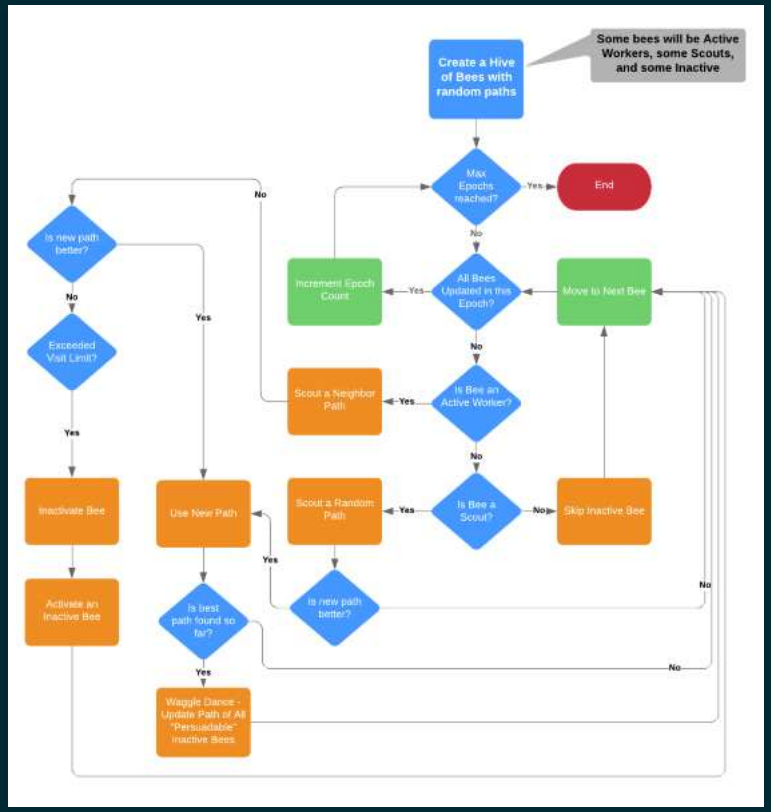
Scouts

Forage on a random path

Inactive Workers

Wait for information from other bees

BEE COLONY OPTIMIZATION



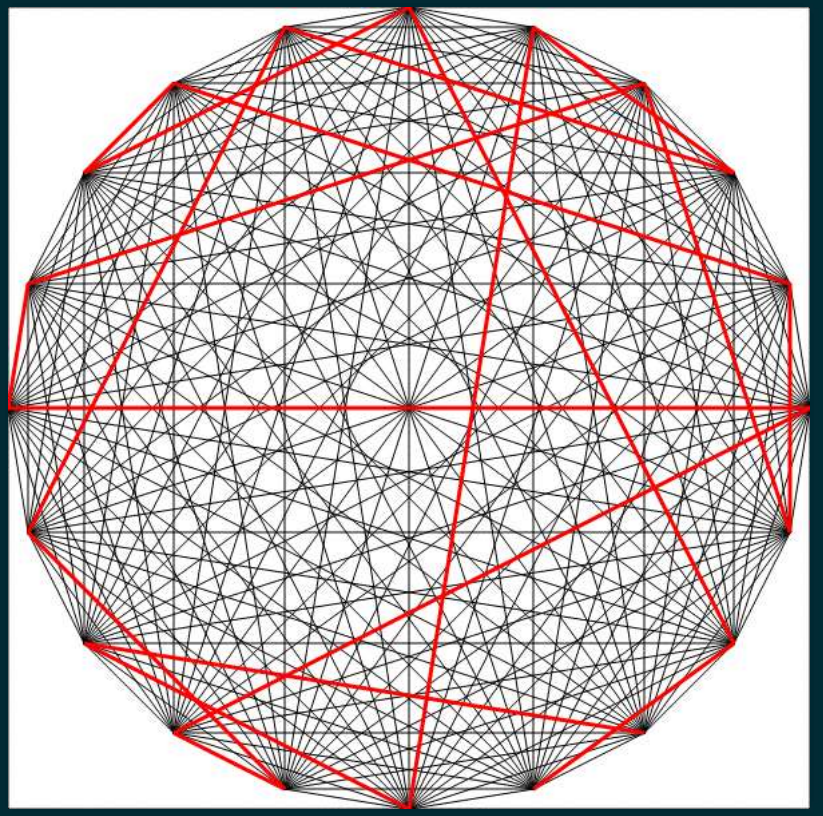
BEE COLONY OPTIMIZATION

```
initialize n bees to random paths
    ~ 75% Active Workers
    ~ 15% Scouts
    ~ 10% Inactive

loop maxEpoch times
    for each bee
        if bee is an Active Worker
            explore a neighbor path
            if new path is shorter than current
                make new path current
                if new path is the best in the hive
                    Waggle Dance - assign ~90% of inactive bees to the new path
            else if visit limit exceeded
                inactivate current bee
                activate a random inactive bee
        else if bee is a Scout
            explore a random alternate path
            if new path is shorter than current
                make new path current
                if new path is the best in the hive
                    Waggle Dance - assign ~90% of inactive bees to the new path
    end for
end loop

return best path found
```

BEE COLONY OPTIMIZATION

- Multiple Search types
  - Active Workers perform neighborhood search
  - Scouts perform random search
  - Best known path propogates
- Tweaks
  - Number of each type of bee
  - Probability of persuasion
  - Visit limit
- Features
  - Less Sensitive to Scale
  - Optimality not guaranteed

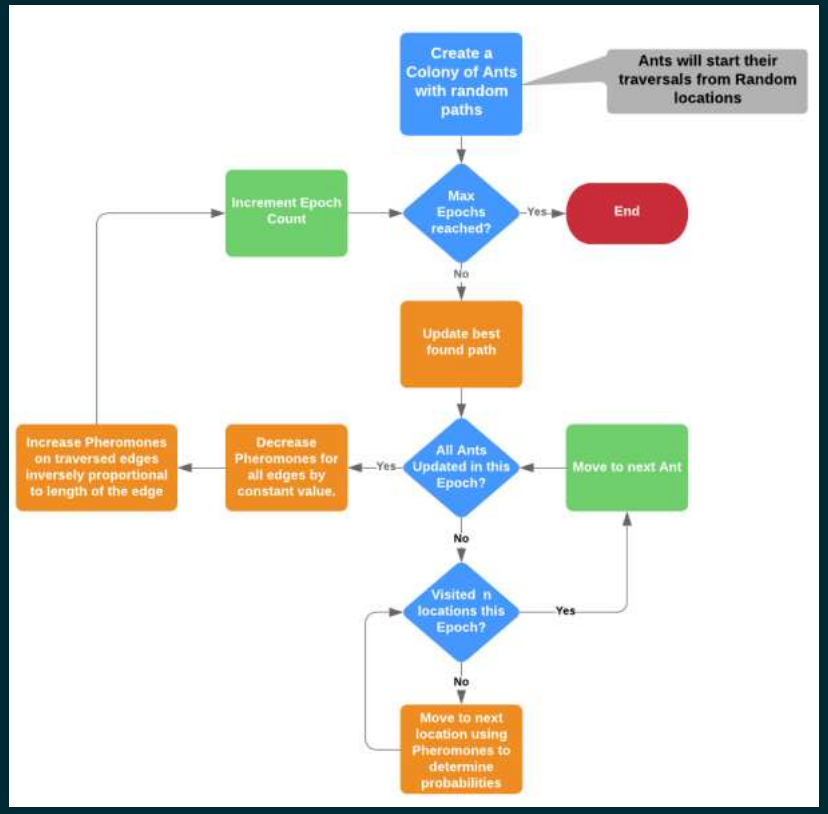
ANT COLONY OPTIMIZATION



PHEROMONES

- The Pheromones are the key
  - Greater Usage => More Pheromones
  - Shorter Path Length => More Pheromones Remain
  - More Pheromones => Higher Probability of Usage

ANT COLONY OPTIMIZATION



ANT COLONY OPTIMIZATION

```
initialize n Ants to random Paths
loop maxEpochs times
  update best Path found
  for each Ant i
    while visits < n
      determine Pheromones on each edge
      calculate % Pheremones / edge
      determine probabilites for each edge
      pick next edge based on probabilies
    end while
  end for
  for each edge between locations
    decrease Pheromones for time passed
    if edge was traversed
      increase Pheromones inversely proportional to length
    end if
  end loop
end loop
return best position found
```

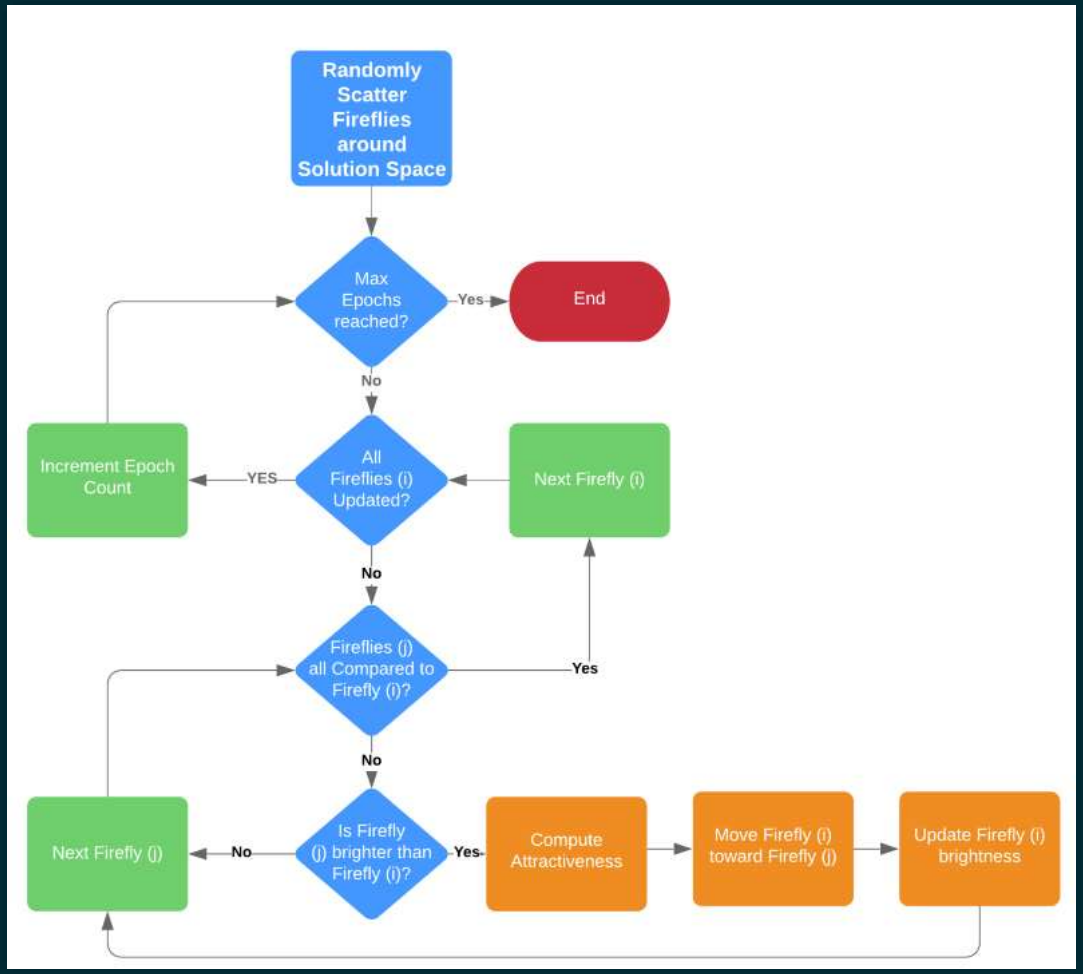
ANT COLONY OPTIMIZATION

- Neighborhood Search
  - Start with random paths
  - Explore neighboring paths based on probability
  - More pheromone = Higher probability of use
  - Can get stuck in a local minima
- Tweaks
  - Number of ants
  - Pheromones to dispense
  - Pheromone dissipation rate
- Features
  - Less sensitive to scale
  - Optimality not guaranteed

REDUCING COSTS

0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
3
2.5
2
1.5
1
0.5
0
0.5
1
1.5
2
2.5
3

FIREFLY OPTIMIZATION

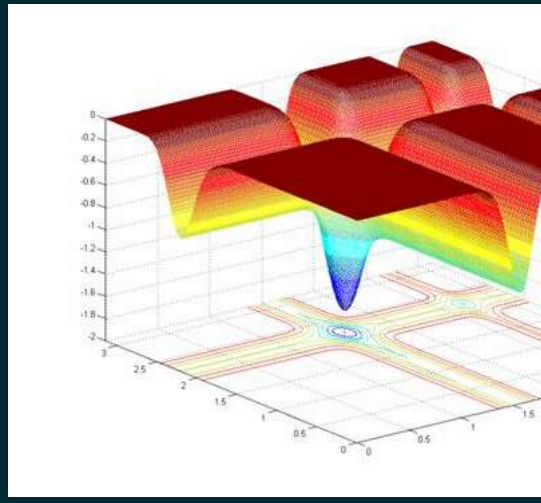


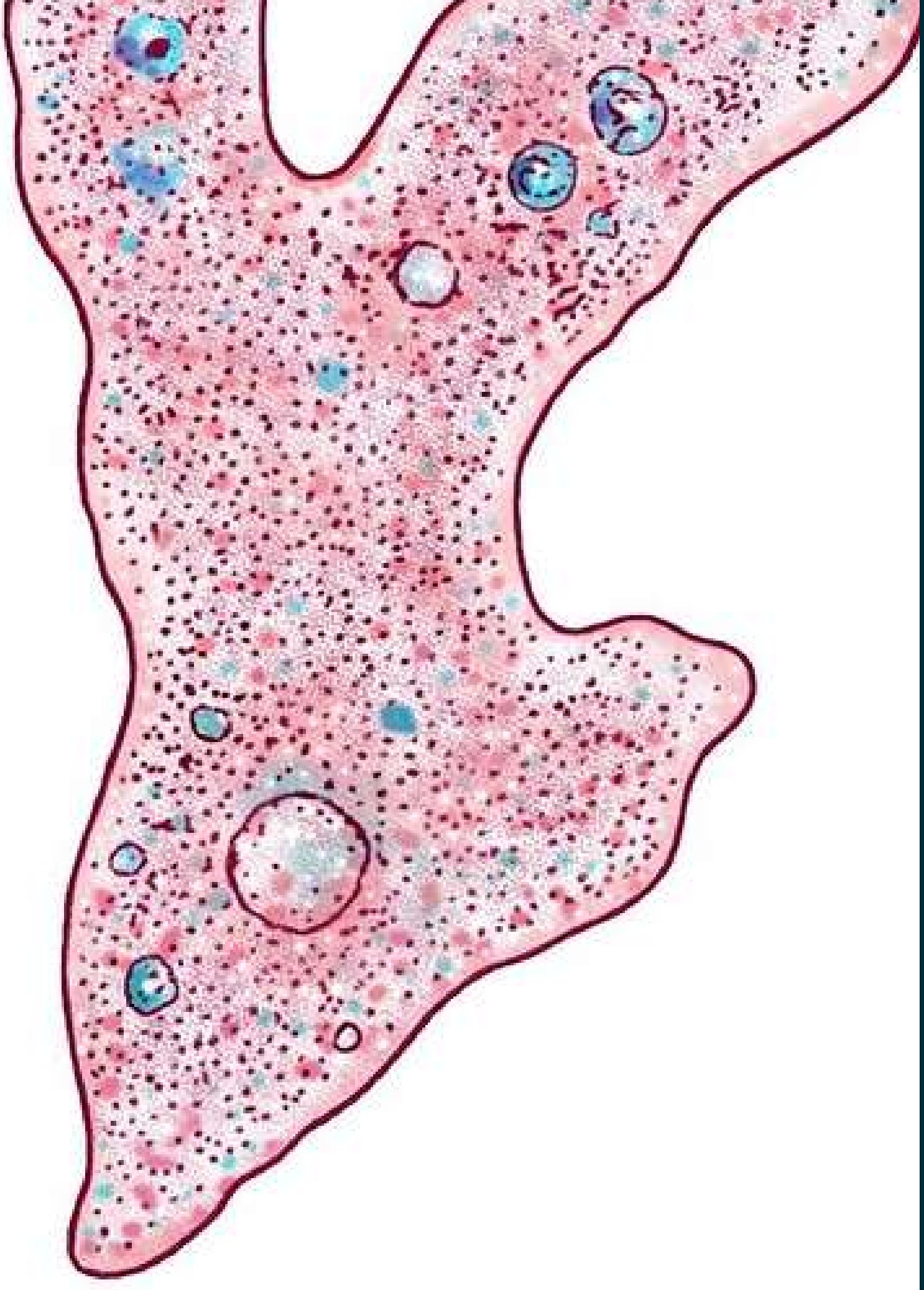
FIREFLY OPTIMIZATION

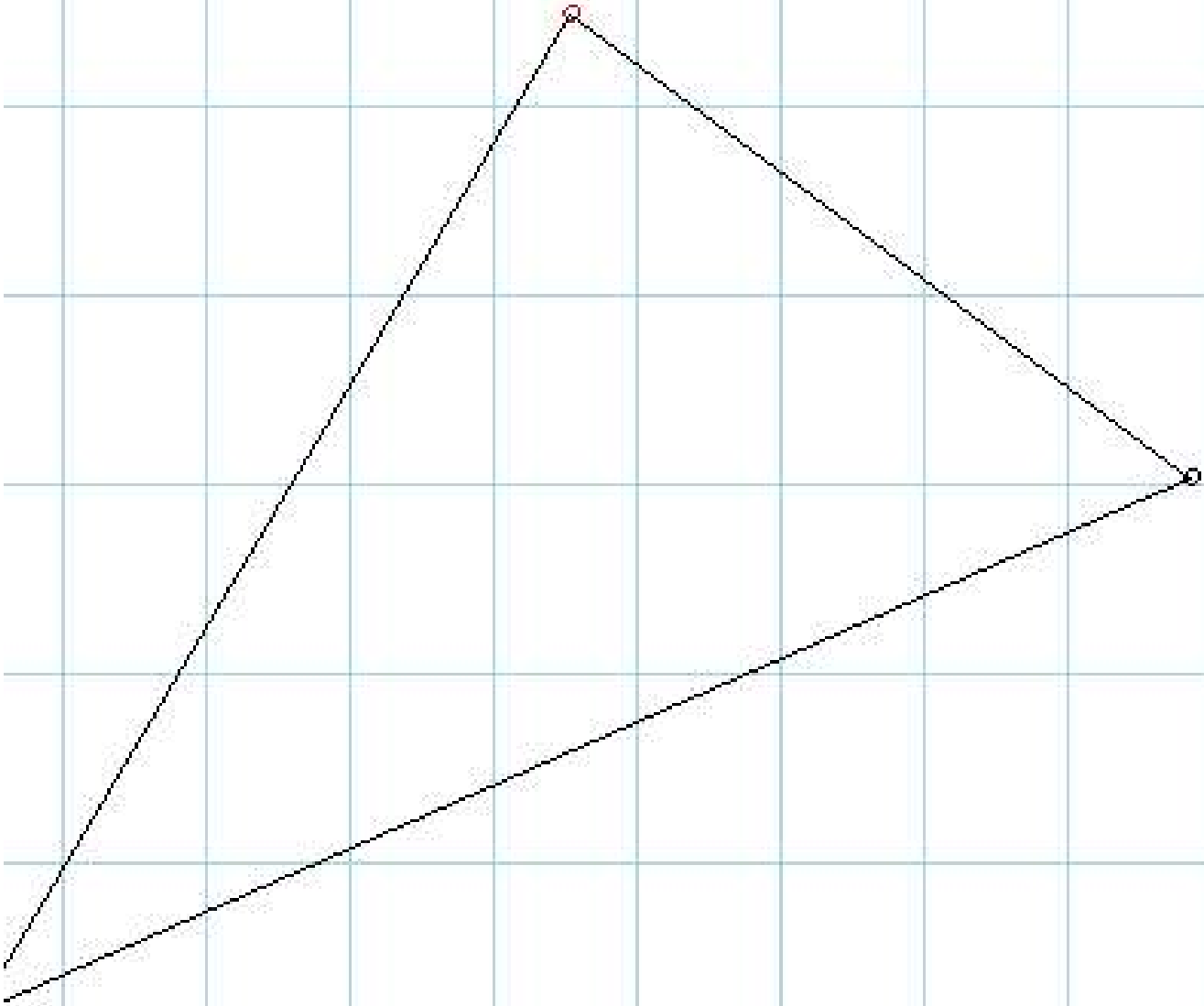
```
initialize n fireflies to random positions
loop maxEpochs times
  for each firefly i
    for each firefly j
      if intensity(i) < intensity(j)
        compute attractiveness
        move firefly(i) toward firefly(j)
        update firefly(i) intensity
      end for
    end for
  sort fireflies
end loop
return best position found
```

FIREFLY OPTIMIZATION

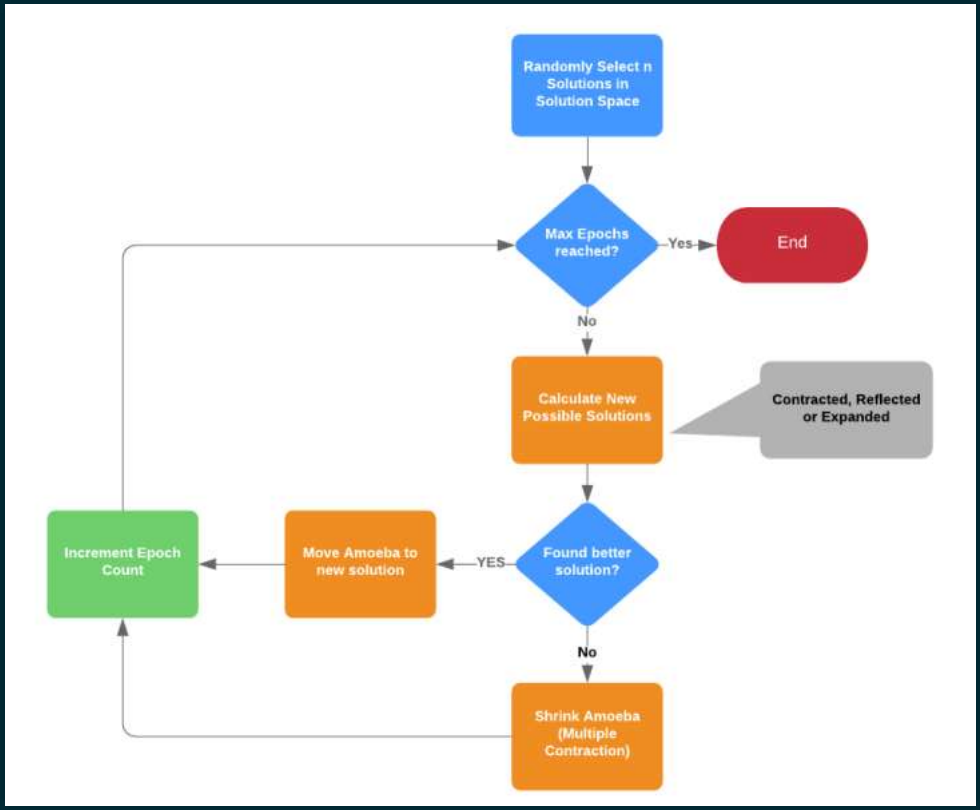
- Neighborhood Search
  - Search the area toward best known solution
  - Closer and Brighter: More Attractive
    - Similar to gravity
- Tweaks
  - Number of fireflies
  - Range of possible values
  - How fast fireflies move
- Features
  - Works better for linear problems
  - Optimality not guaranteed
  - Can suffer from sparseness problems







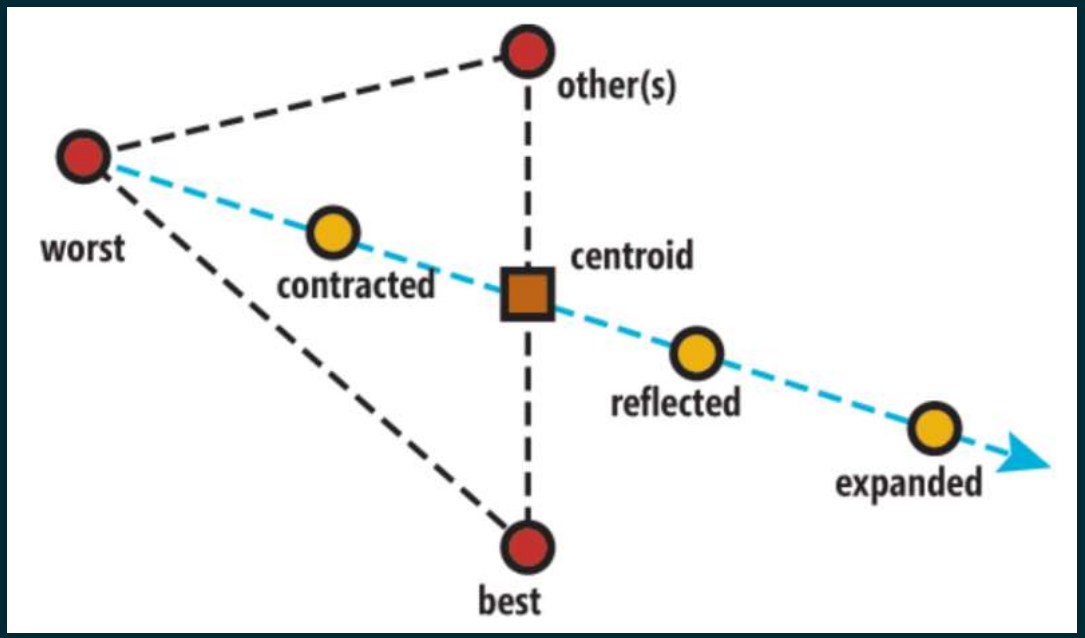
AMOEBA OPTIMIZATION



AMOEBA OPTIMIZATION

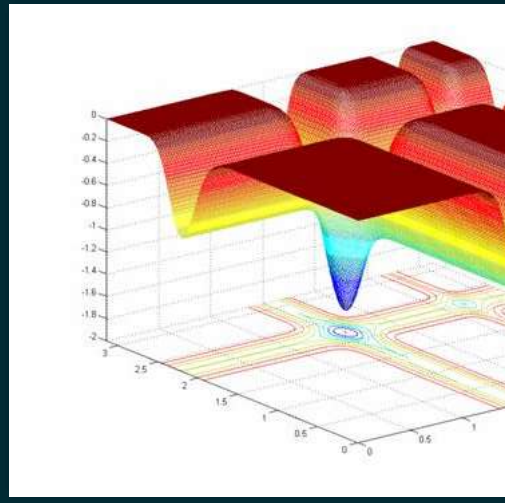
```
initialize the amoeba with n (size) locations
loop maxEpochs times
    calculate new possible solutions
        contracted - midway between centroid and worst
        reflected - contracted point reflected across centroid
        expanded - beyond reflected point by a constant factor
    if any solution is better than the current
        replace worst value with best value from new solution
    else
        shrink (multiple contract) all lesser nodes toward the best
    increment epoch count
end loop
return best position found
```

AMOEBA OPTIMIZATION



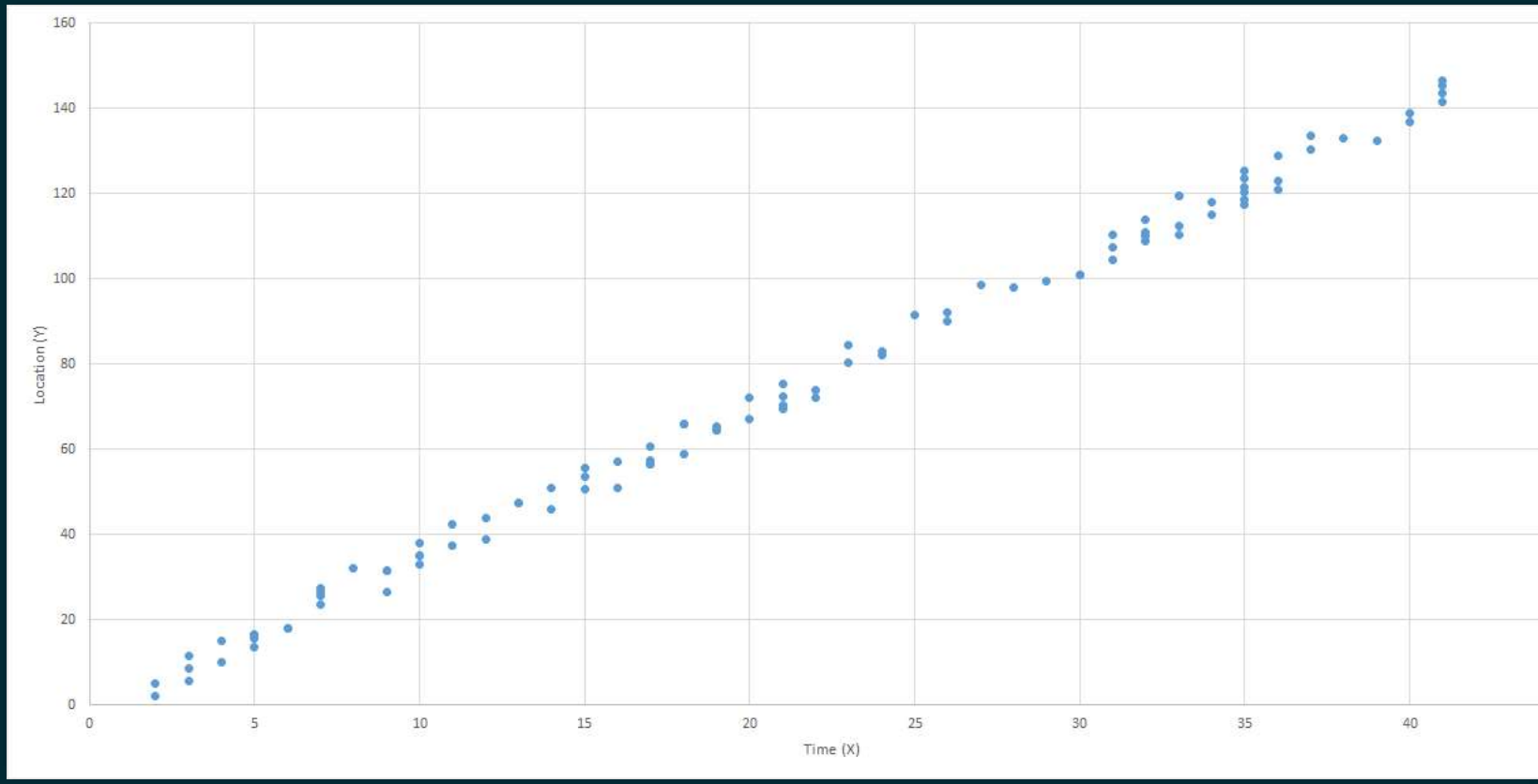
AMOEBA OPTIMIZATION

- Neighborhood Search
  - Start with random locations
  - Explore neighbors based on amoeba movement
  - Surround the solution, then contract to it
- Tweaks
  - Size of the amoeba
    - > # of search dimensions
  - Number of executions
    - Handle local minima
- Features
  - Optimality no guaranteed
  - Can suffer from sparseness problems




WHAT CAN WE DO WITH THIS STU

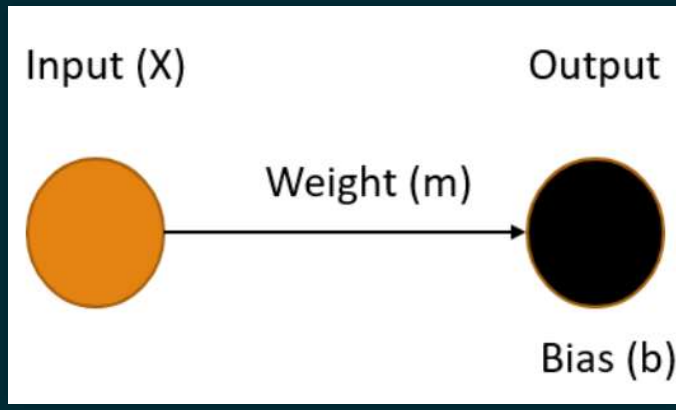
LINEAR DATA


160
140
120
100
80
60
40
20
0
Location (Y)
0
5
10
15
20
25
30
35
40
Time (X)

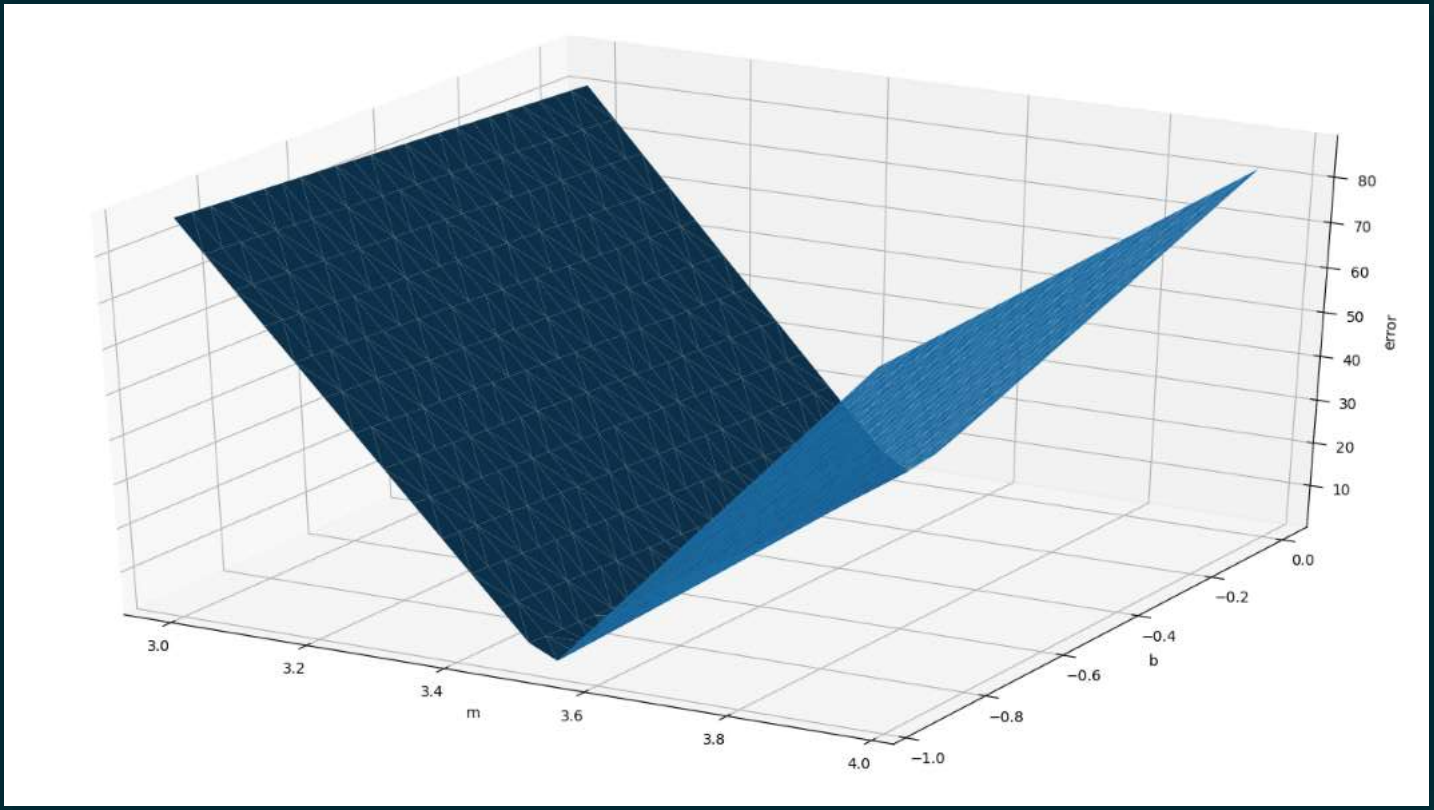
LINEAR REGRESSION MODEL

PREDICT THE UNKNOWN VALUES IN A LINEAR EQUATION

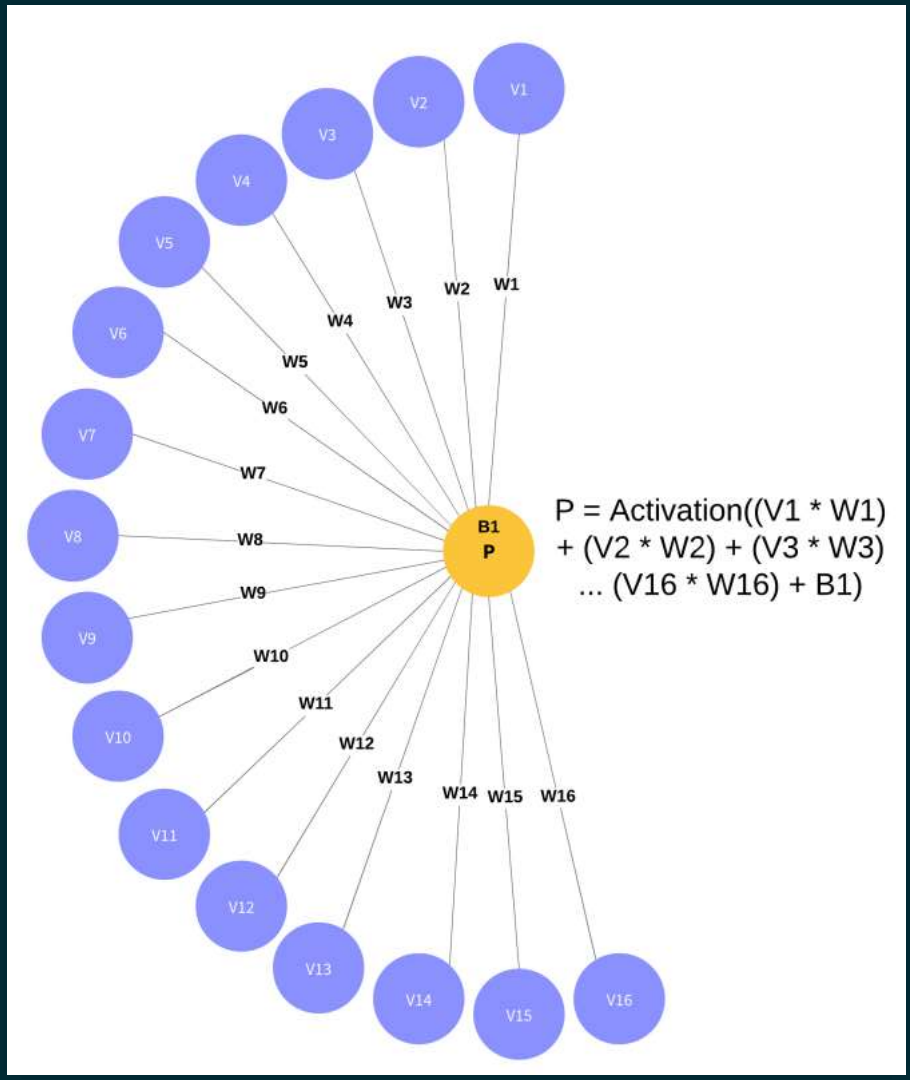
- Given X (time), predict Y (location)
  - `Y = mX + b`
- Find the best values for **m** and **b**
  - Minimize total error

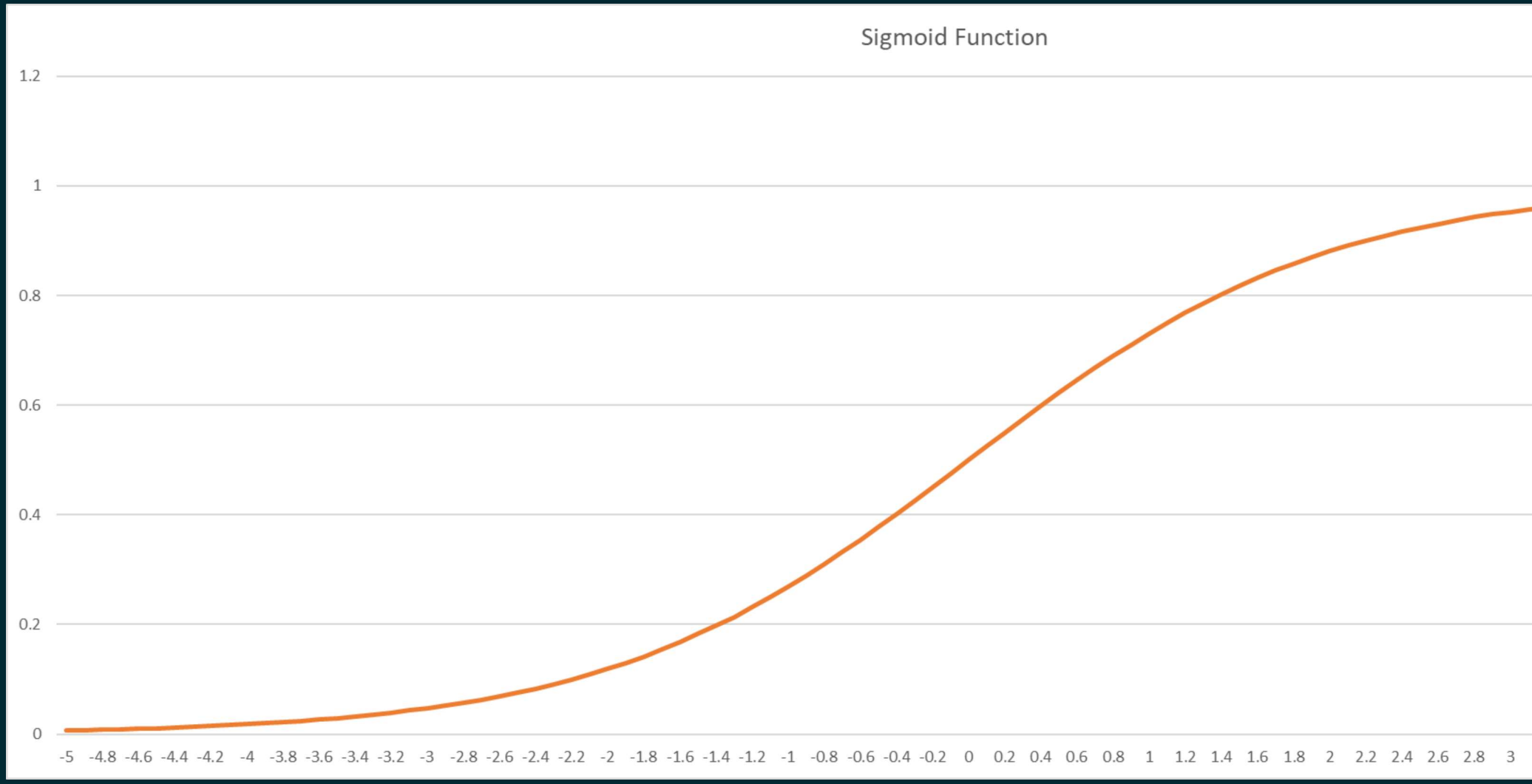


ERROR FUNCTION



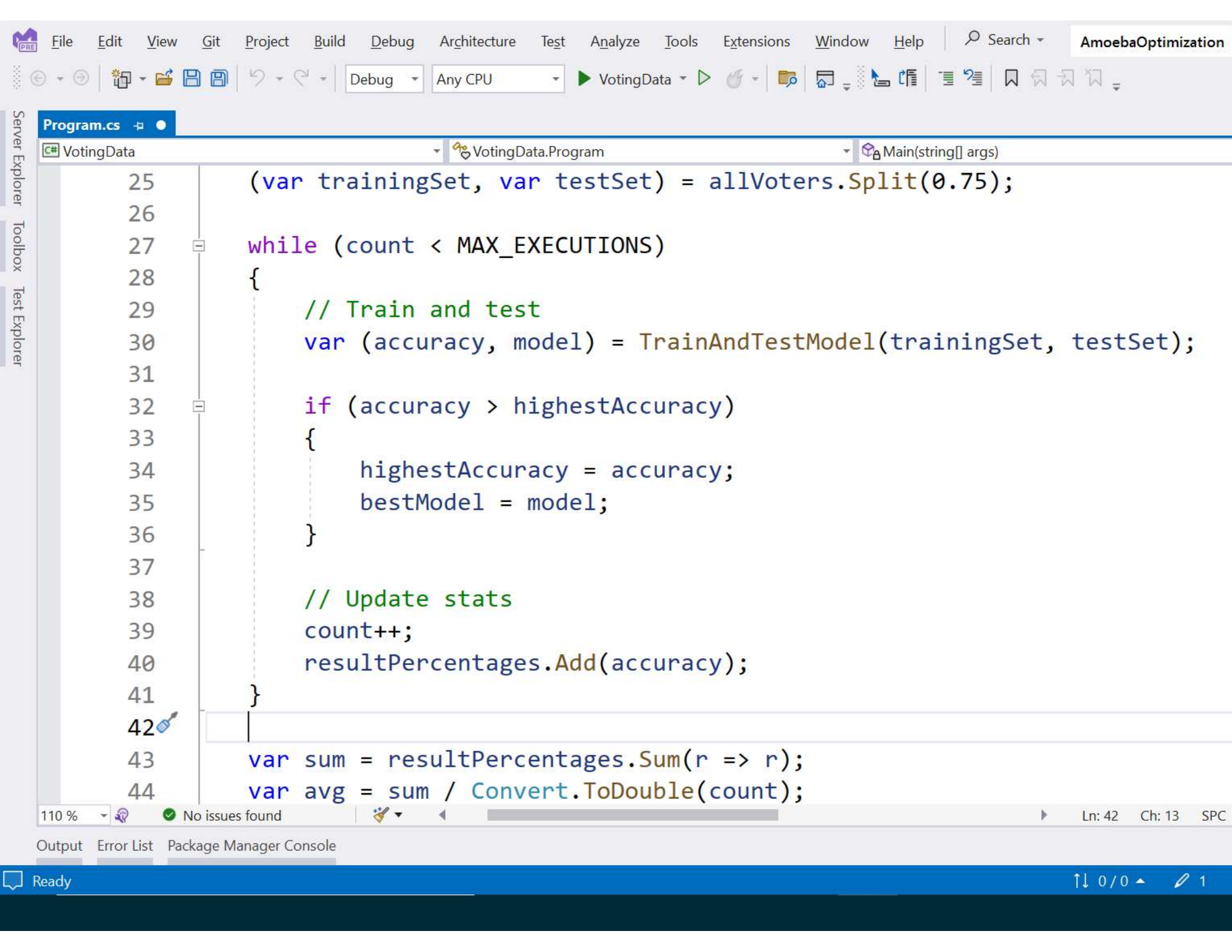
VOTER MODEL

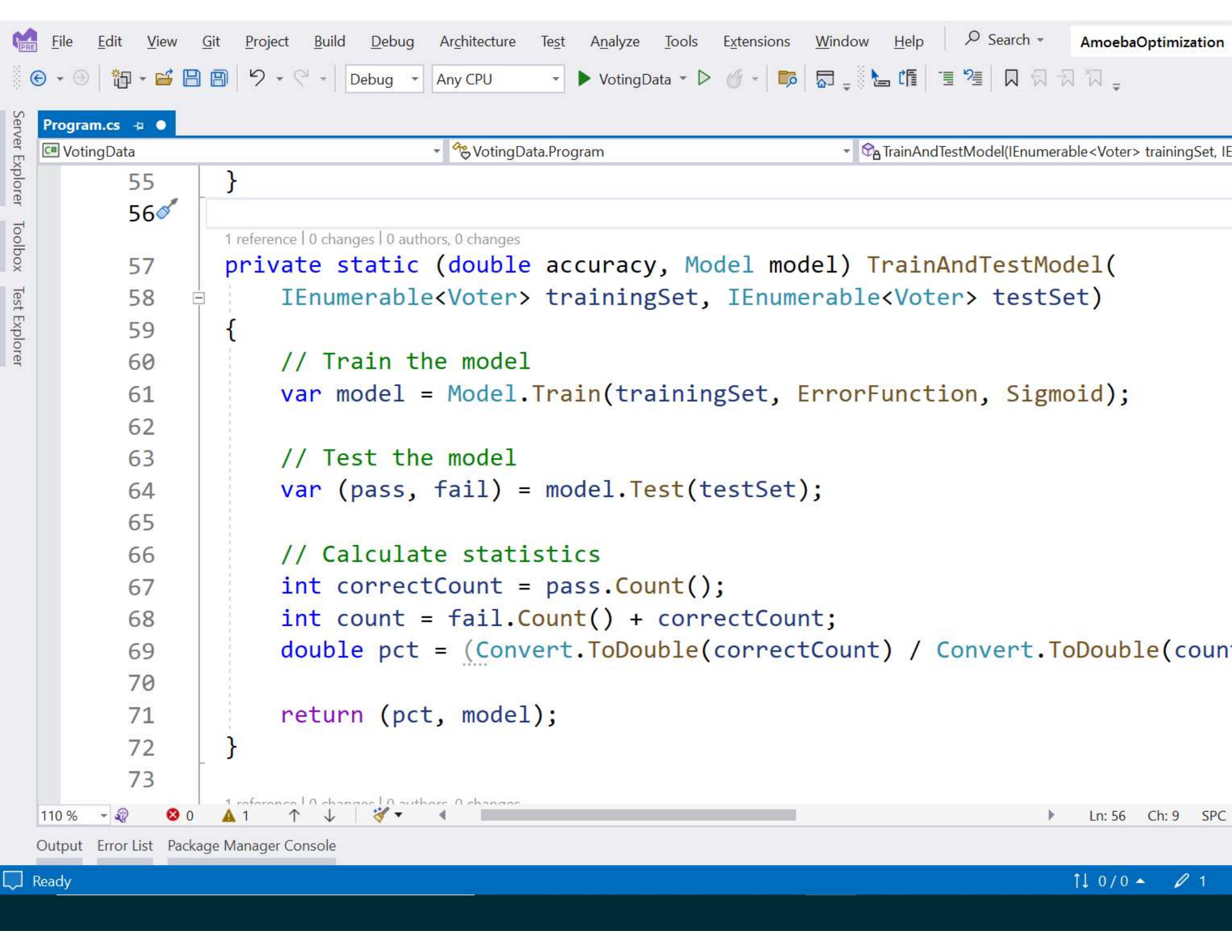

V1
V2
V3
V4
V5
V6
V7
V8
V9
V10
V11
V12
V13
V14
V15
V16
W1
W2
W3
W4
W5
W6
W7
W8
W9
W10
W11
W12
W13
W14
W15
W16
B1
P
P = Activation((V1 * W1)
+ (V2 * W2) + (V3 * W3)
... (V16 * W16) + B1)

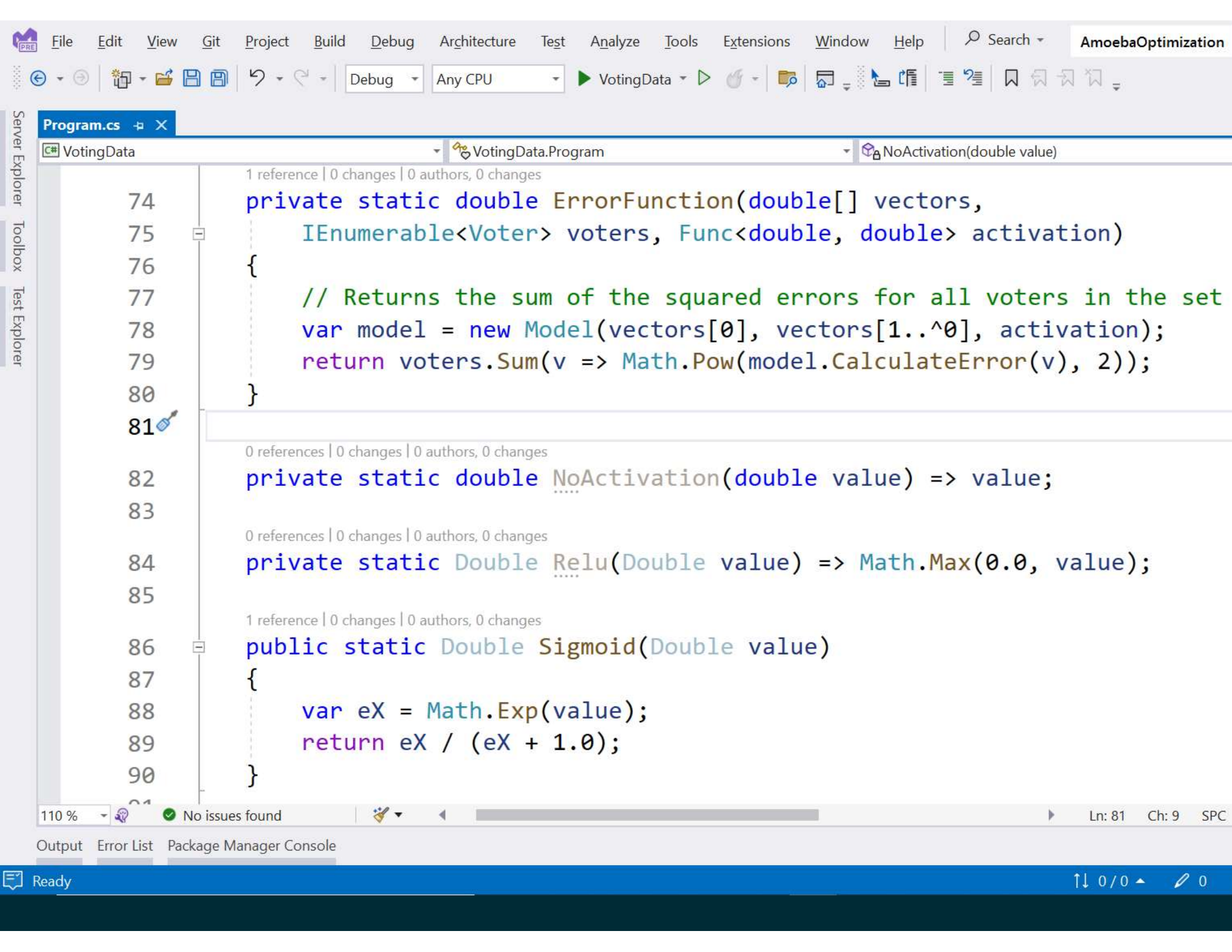

Sigmoid Function
1.2
1
0.8
0.6
0.4
0.2
0
-5 -4.8 -4.6 -4.4 -4.2 -4 -3.8 -3.6 -3.4 -3.2 -3 -2.8 -2.6 -2.4 -2.2 -2 -1.8 -1.6 -1.4 -1.2 -1 -0.8 -0.6 -0.4 -0.2 0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6 1.8 2 2.2 2.4 2.6 2.8 3

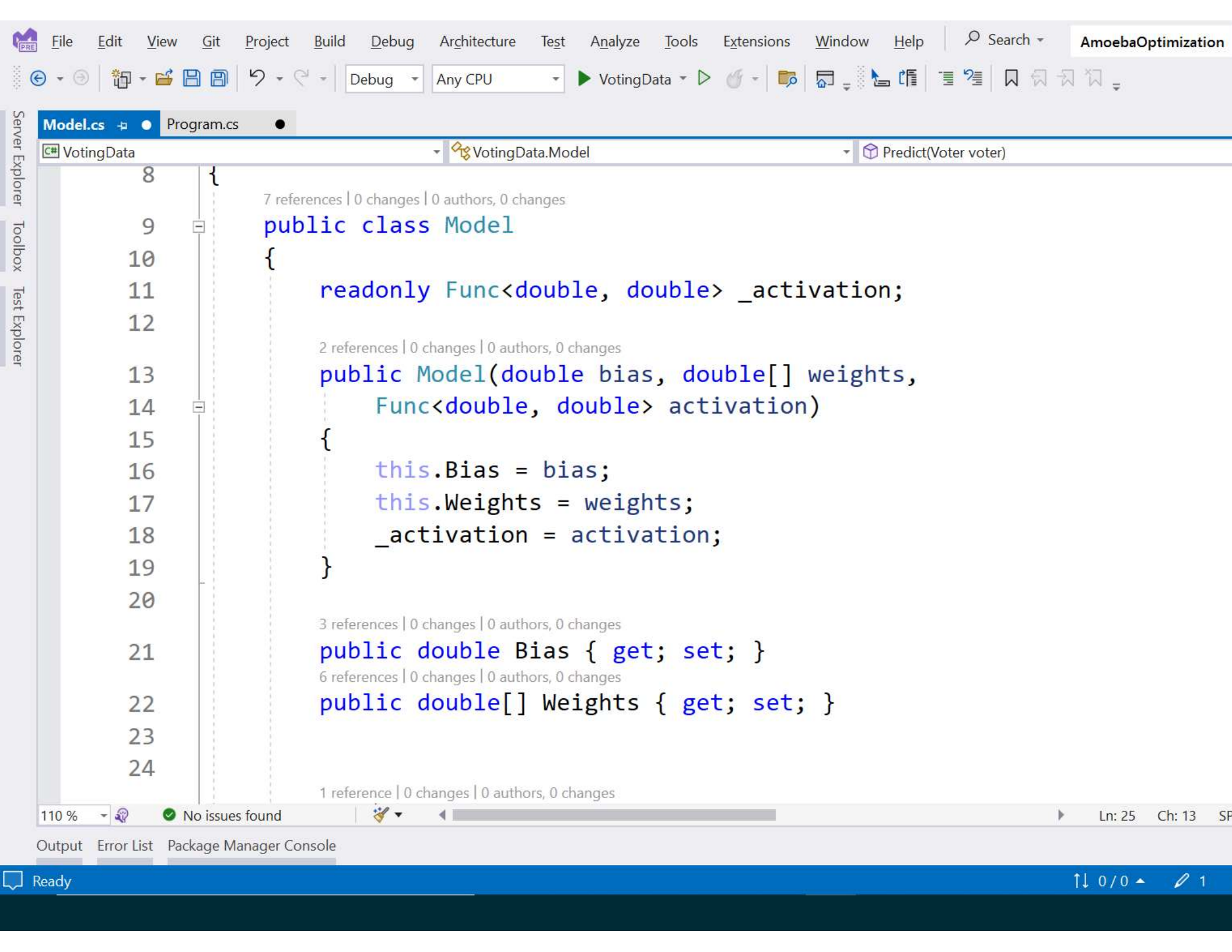
ML DEMO

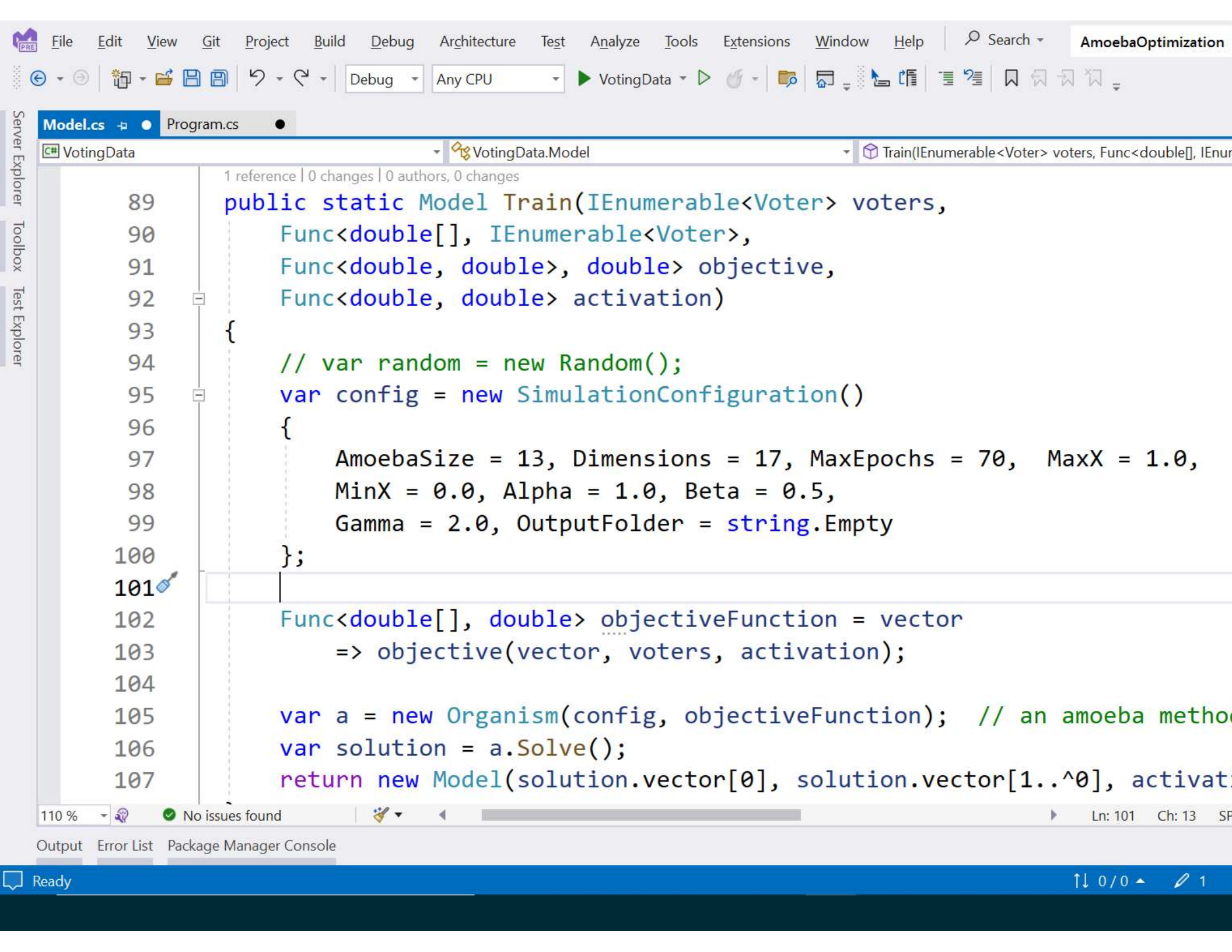
USING AMOEBA OPTIMZATION TO TRAIN AN ML MODEL

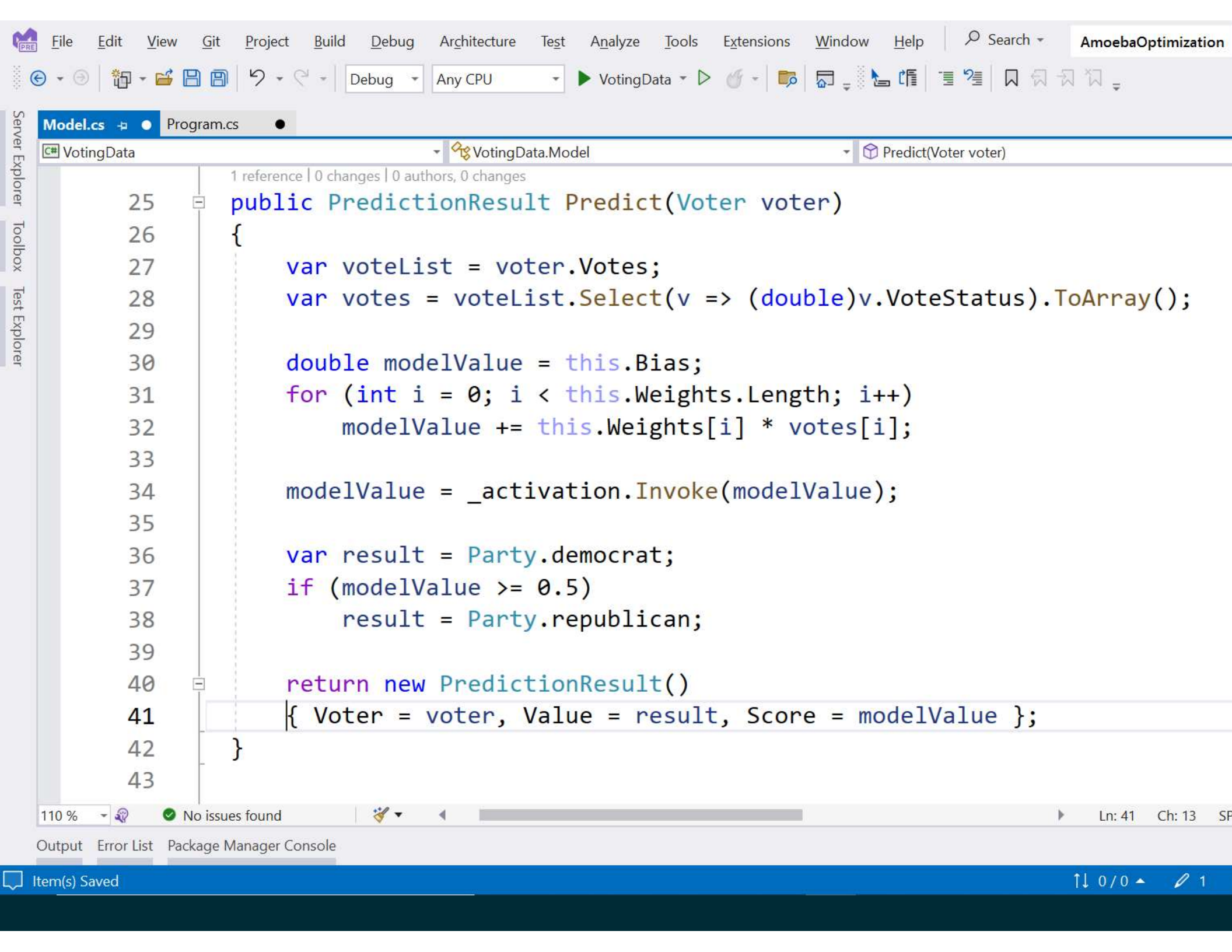

File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Server Explorer
Toolbox
Test Explorer
Program.cs
VotingData
VotingData.Program
Main(string[] args)
25 (var trainingSet, var testSet) = allVoters.Split(0.75);
26
27 while (count < MAX_EXECUTIONS)
28 {
29 // Train and test
30 var (accuracy, model) = TrainAndTestModel(trainingSet, testSet);
31
32 if (accuracy > highestAccuracy)
33 {
34 highestAccuracy = accuracy;
35 bestModel = model;
36 }
37
38 // Update stats
39 count++;
40 resultPercentages.Add(accuracy);
41 }
42
43 var sum = resultPercentages.Sum(r => r);
44 var avg = sum / Convert.ToDouble(count);
110 %
No issues found
Ln: 42
Ch: 13
SPC
Output
Error List
Package Manager Console
Ready
0 / 0
1


File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Server Explorer
Toolbox
Test Explorer
Program.cs
VotingData
VotingData.Program
TrainAndTestModel(IEnumerable<Voter> trainingSet, IE
55 }
56
1 reference | 0 changes | 0 authors, 0 changes
57 private static (double accuracy, Model model) TrainAndTestModel(
58 IEnumerable<Voter> trainingSet, IEnumerable<Voter> testSet)
59 {
60 // Train the model
61 var model = Model.Train(trainingSet, ErrorFunction, Sigmoid);
62
63 // Test the model
64 var (pass, fail) = model.Test(testSet);
65
66 // Calculate statistics
67 int correctCount = pass.Count();
68 int count = fail.Count() + correctCount;
69 double pct = (Convert.ToDouble(correctCount) / Convert.ToDouble(coun
70
71 return (pct, model);
72 }
73
110 %
0
1
Ln: 56
Ch: 9
SPC
Output
Error List
Package Manager Console
Ready
0 / 0
1


File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Program.cs
VotingData
VotingData.Program
NoActivation(double value)
1 reference | 0 changes | 0 authors, 0 changes
private static double ErrorFunction(double[] vectors,
IEnumerable<Voter> voters, Func<double, double> activation)
{
// Returns the sum of the squared errors for all voters in the set
var model = new Model(vectors[0], vectors[1..^0], activation);
return voters.Sum(v => Math.Pow(model.CalculateError(v), 2));
}
0 references | 0 changes | 0 authors, 0 changes
private static double NoActivation(double value) => value;
0 references | 0 changes | 0 authors, 0 changes
private static Double Relu(Double value) => Math.Max(0.0, value);
1 reference | 0 changes | 0 authors, 0 changes
public static Double Sigmoid(Double value)
{
var eX = Math.Exp(value);
return eX / (eX + 1.0);
}
110 %
No issues found
Ln: 81
Ch: 9
SPC
Output
Error List
Package Manager Console
Ready
0 / 0
0


File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Model.cs
Program.cs
VotingData
VotingData.Model
Predict(Voter voter)
{
7 references | 0 changes | 0 authors, 0 changes
public class Model
{
readonly Func<double, double> _activation;
2 references | 0 changes | 0 authors, 0 changes
public Model(double bias, double[] weights,
Func<double, double> activation)
{
this.Bias = bias;
this.Weights = weights;
_activation = activation;
}
3 references | 0 changes | 0 authors, 0 changes
public double Bias { get; set; }
6 references | 0 changes | 0 authors, 0 changes
public double[] Weights { get; set; }
1 reference | 0 changes | 0 authors, 0 changes
110 %
No issues found
Ln: 25
Ch: 13
Output
Error List
Package Manager Console
Ready
0 / 0
1


File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Model.cs
Program.cs
VotingData
VotingData.Model
Train(IEnumerable<Voter> voters, Func<double[], IEnur
1 reference | 0 changes | 0 authors, 0 changes
public static Model Train(IEnumerable<Voter> voters,
Func<double[], IEnumerable<Voter>,
Func<double, double>, double> objective,
Func<double, double> activation)
{
// var random = new Random();
var config = new SimulationConfiguration()
{
AmoebaSize = 13, Dimensions = 17, MaxEpochs = 70, MaxX = 1.0,
MinX = 0.0, Alpha = 1.0, Beta = 0.5,
Gamma = 2.0, OutputFolder = string.Empty
};
Func<double[], double> objectiveFunction = vector
=> objective(vector, voters, activation);
var a = new Organism(config, objectiveFunction); // an amoeba metho
var solution = a.Solve();
return new Model(solution.vector[0], solution.vector[1..^0], activat
110 %
No issues found
Ln: 101
Ch: 13
Output
Error List
Package Manager Console
Ready
0 / 0
1
Server Explorer
Toolbox
Test Explorer


File Edit View Git Project Build Debug Architecture Test Analyze Tools Extensions Window Help
Search
AmoebaOptimization
Debug
Any CPU
VotingData
Model.cs
Program.cs
VotingData
VotingData.Model
Predict(Voter voter)
1 reference | 0 changes | 0 authors, 0 changes
public PredictionResult Predict(Voter voter)
{
var voteList = voter.Votes;
var votes = voteList.Select(v => (double)v.VoteStatus).ToArray();
double modelValue = this.Bias;
for (int i = 0; i < this.Weights.Length; i++)
modelValue += this.Weights[i] * votes[i];
modelValue = _activation.Invoke(modelValue);
var result = Party.democrat;
if (modelValue >= 0.5)
result = Party.republican;
return new PredictionResult()
{ Voter = voter, Value = result, Score = modelValue };
}
110 %
No issues found
Ln: 41
Ch: 13
Output
Error List
Package Manager Console
Item(s) Saved
0 / 0
1

```
Training - Mean: 89.02222222222218  Min: 76.06837606837607  Max 95.72649572649573  StDe
Incorrect Predictions:
Voter 00104 - Party:   democrat (0.9998)
Voter 00107 - Party: republican (0.0412)
Voter 00144 - Party:   democrat (0.9867)
Voter 00147 - Party:   democrat (0.6808)
Voter 00151 - Party:   democrat (0.9845)
Voter 00161 - Party:   democrat (0.9960)
Voter 00166 - Party: republican (0.1047)
Voter 00167 - Party: republican (0.0322)
Voter 00176 - Party: republican (0.1021)
Voter 00197 - Party: republican (0.0001)
Voter 00242 - Party: republican (0.0002)
Voter 00248 - Party: republican (0.0010)
Voter 00267 - Party: republican (0.2057)
Voter 00296 - Party: republican (0.0000)
Voter 00313 - Party: republican (0.3075)
Voter 00315 - Party: republican (0.0349)
Voter 00326 - Party:   democrat (1.0000)
Voter 00352 - Party:   democrat (0.9998)
Voter 00355 - Party: republican (0.0000)
Voter 00375 - Party:   democrat (0.9999)
Voter 00382 - Party:   democrat (0.9616)
Voter 00393 - Party: republican (0.0002)
Voter 00407 - Party:   democrat (1.0000)
Voter 00430 - Party: republican (0.4934)
```

MORE BIO-INSPIRED

- Roach Infestation Optimization
- Particle Swarm Optimization
- Multi-Swarm (Birds) Optimization
- Bacterial Foraging Optimization
- Genetic Algorithms

SUMMARY

- **Minimum Cost Optimization**
  - Firefly Optimization
  - Amoeba Method Optimization
- **Best Path Optimization**
  - Bee Colony Optimization
  - Ant Colony Optimization

RESOURCES

- This slide deck
  - https://amazingalgorithms.azurewebsites.net/
- My Blog
  - http://www.CognitiveInheritance.com
- Building AI Solutions with Google OR-Tools
  - https://www.youtube.com/watch?v=zZAobExOMB0&list=FLq-iLd7rfmqSIFiu
- Articles
  - https://msdn.microsoft.com/en-us/magazine/mt147244.aspx
  - https://msdn.microsoft.com/magazine/gg983491
  - https://msdn.microsoft.com/magazine/dn201752
  - https://msdn.microsoft.com/magazine/hh781027
- Code (many languages)
  - https://github.com/TheAlgorithms
  - https://github.com/bsstahl/AIDemos