Final project and assignments

Paulina Stochel

Multi-Layer Neural Network

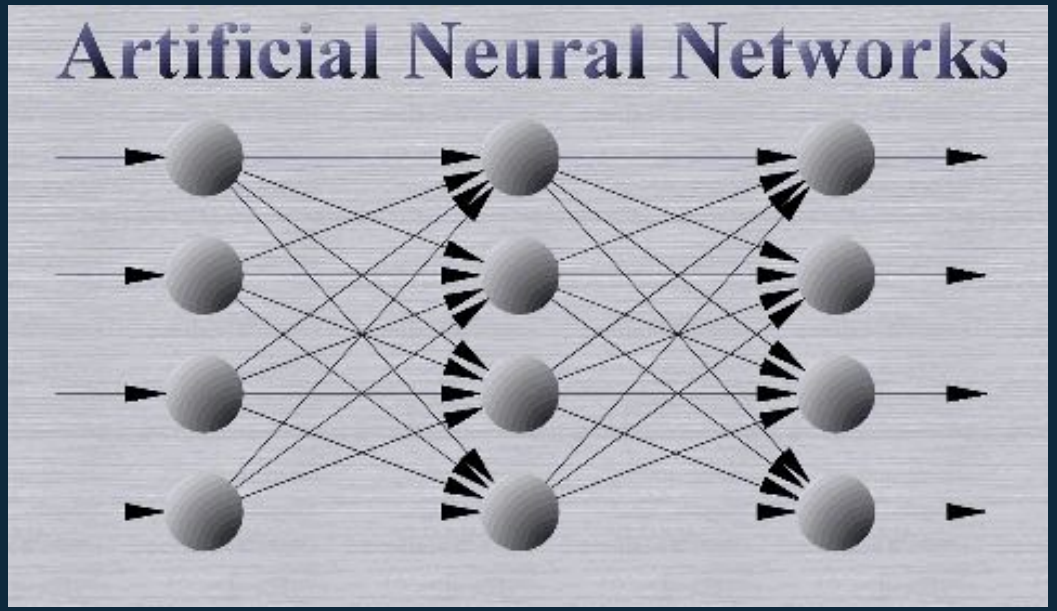
Algorithm:

- Initialize Network.
- Forward Propagate.
- Back Propagate Error.
- Train Network.
- Predict.

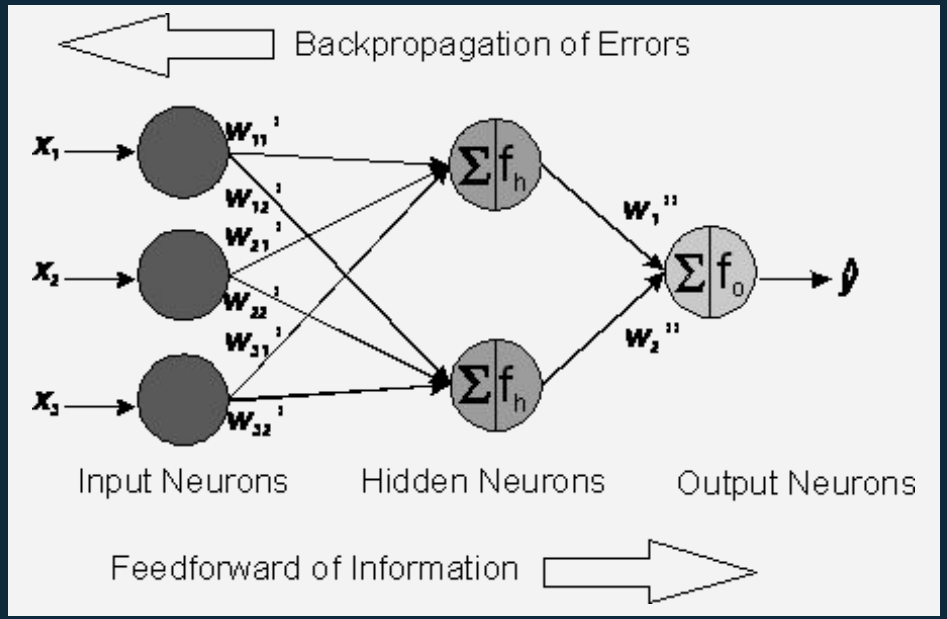
Multi-Layer Neural Network

- class Neuron
  - bias
  - weights
  - delta
- layers
  - input layer
  - hidden layers
  - output layer

Multi-Layer Neural Network



Multi-Layer Neural Network



Example output

Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0
Expected=0, Got=0

Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1
Expected=1, Got=1

Expected=2, Got=1
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2
Expected=2, Got=2

Multi-Layer Neural Network

K - fold cross validation

96% -> 98%

Scores:

[96.66666666666667, 96.66666666666667, 96.66666666666667, 100.0, 100.0]

Mean Accuracy:

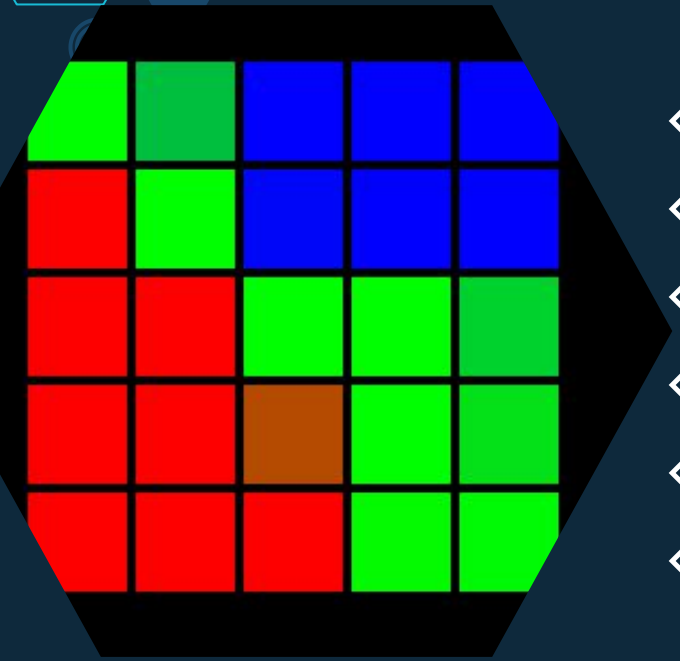
98.000%

	98%
Folds number	5
Learning rate	0.01
Iterations number	1000
Neurons in hidden layer	15

K – fold cross validation parameters and accuracy

	97.33%	95.33%	97.33%	98%	97.33%
Folds number	5	5	5	5	6
Learning rate	0.1	0.1	0.05	0.01	0.01
Iterations number	500	1000	1000	1000	1000
Neurons in hidden layer	5	15	15	15	15

Self-Organizing Map



Algorithm:

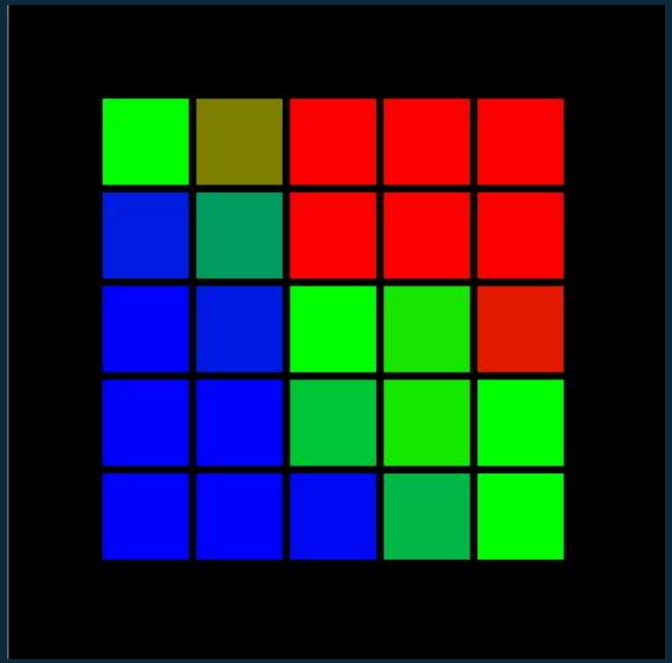
- Initialize node’s weights.
- Randomly chosen vector.
- Finding Best Matching Unit.
- Calculating radius.
- Adjusting node’s weights.
- Repeating for N iterations.

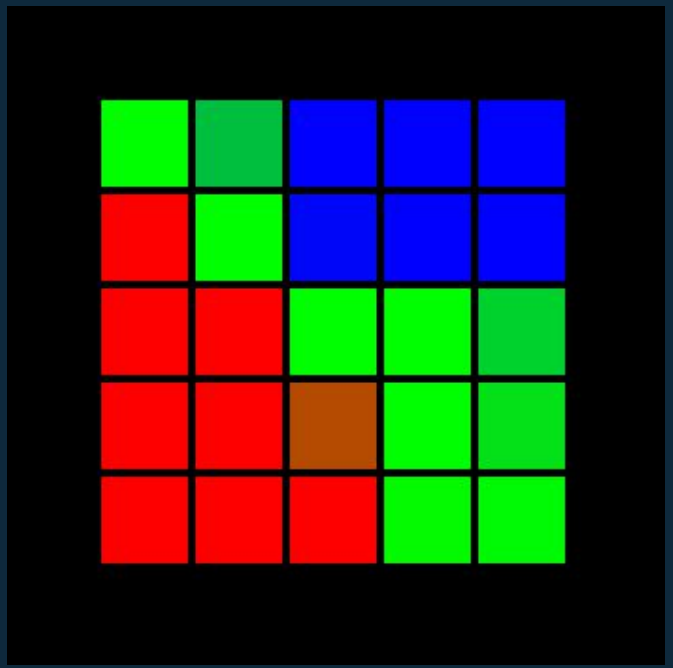
SOM

```
SOM = np.random.random((som_dimensions[0], som_dimensions[1], m))

for i in range(n_iterations):
...
    for x in range(SOM.shape[0]):
        for y in range(SOM.shape[1]):
            w = SOM[x, y, :].reshape(m, 1)
            w_dist = np.sum((np.array([x, y]) - bmu_idx) ** 2)
            # if the distance is within the current neighbourhood radius
            if w_dist <= r**2:
                infl = calculate_influence(w_dist, r)
                new_w = w + (l * infl * (t - w))
                SOM[x, y, :] = new_w.reshape(1, m)
```

Self-Organizing Map





Self-Organizing Map

Using SOM as an
input of neural network

96% -> 97.333%

Parameter	value
Learning rate	0.5
Iterations number	1000
Grid dimension	5

Final project

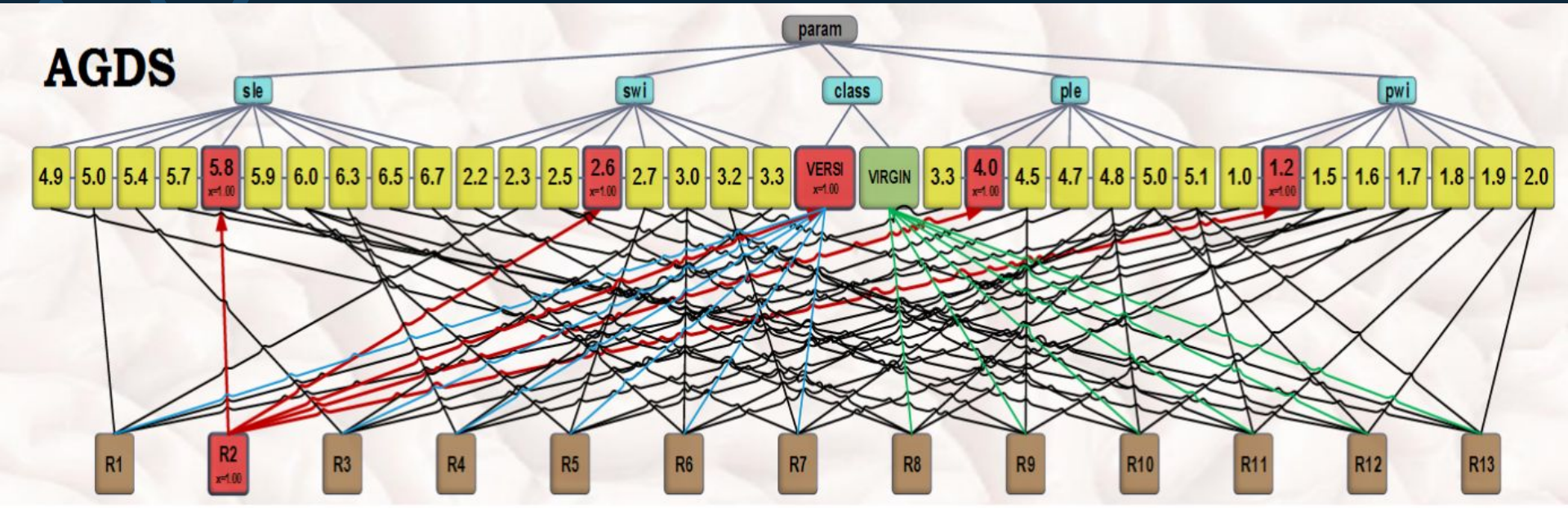
Associative Graph Data Structure (AGDS) Used for Data Representation and Reasoning

AGDS

Algorithm:

- Initialize structure of attributes and values.
- Sorting values in each attribute.
- Calculating weight for connections.
- Connecting objects to corresponding values.
- Assigning value x in value nodes.
- Counting similarities to the chosen object.

AGDS



Source: lectures of dr hab. Adrian Horzyk, http://home.agh.edu.pl/~horzyk/

Final project

House Sales in King County, USA

- date,
- price,
- bedrooms,
- bathrooms,
- square footage of the home,
- floors,
- House which has a view to a waterfront,
- condition,
- grade of the house,
- square footage of house apart from basement,
- Built Year,
- Latitude coordinate,
- Longitude coordinate

Final project

Example:

price, bedroom, bathroom, living, lot, floors, view, waterfront, condition, grade, above, basem, built, renovated, zipcode, lat, long, living, lot

180000, 3, 2, 1000, 1000, 1, 0, 0, 4, 10, 1200, 200, 1995, 0, 98178, 47.3097, -122.327, 1500, 1300

#1 - 96,51%

149900, 2, 1.75, 1090, 1950, 1, 0, 0, 4, 8, 1090, 0, 1982, 0, 98198, 47.3782, -122.319, 1360, 3426

#2 - 96,13%

202200, 2, 1.75, 1330, 2159, 1, 0, 0, 4, 8, 1330, 0, 1979, 0, 98198, 47.3822, -122.32, 1220, 3679

#3 - 95,86%

176000, 2, 1, 920, 2332, 1, 0, 0, 4, 8, 920, 0, 1980, 0, 98198, 47.3779, -122.32, 1310, 2853

Thank you for your attention!