2024

A Guide to Ensemble Styled Machine Learning with JavaScript and Brain.js

Michael Angelo De La Cruz Ortiz
Clevelan State University

A Guide to Ensemble Styled Machine Learning with JavaScript and Brain.js

De La Cruz Ortiz, Michael Angelo; Dr. Eric M Schearer

Center for Human Machine Systems, Cleveland State University, Cleveland, OH

Introduction

This is a guide which details the development of AI (Artificial Intelligence) using brain.js, a library of features which allow for the high-level development of ANNs (Artificial Neural Networks), all purely written in the JavaScript programing language.

Machine Learning with **Brain.js**

&

Goal

This is a comprehensive guide which details an Ensemble Styled ML (Machine Learning) procedure by which brain.js is used to develop of ANN system trained on a sequential, multi-dimensional dataset spanning varying patterns and yet done in a timely and efficient manner while ensuring the product model was neither too under nor overfitted for the task.

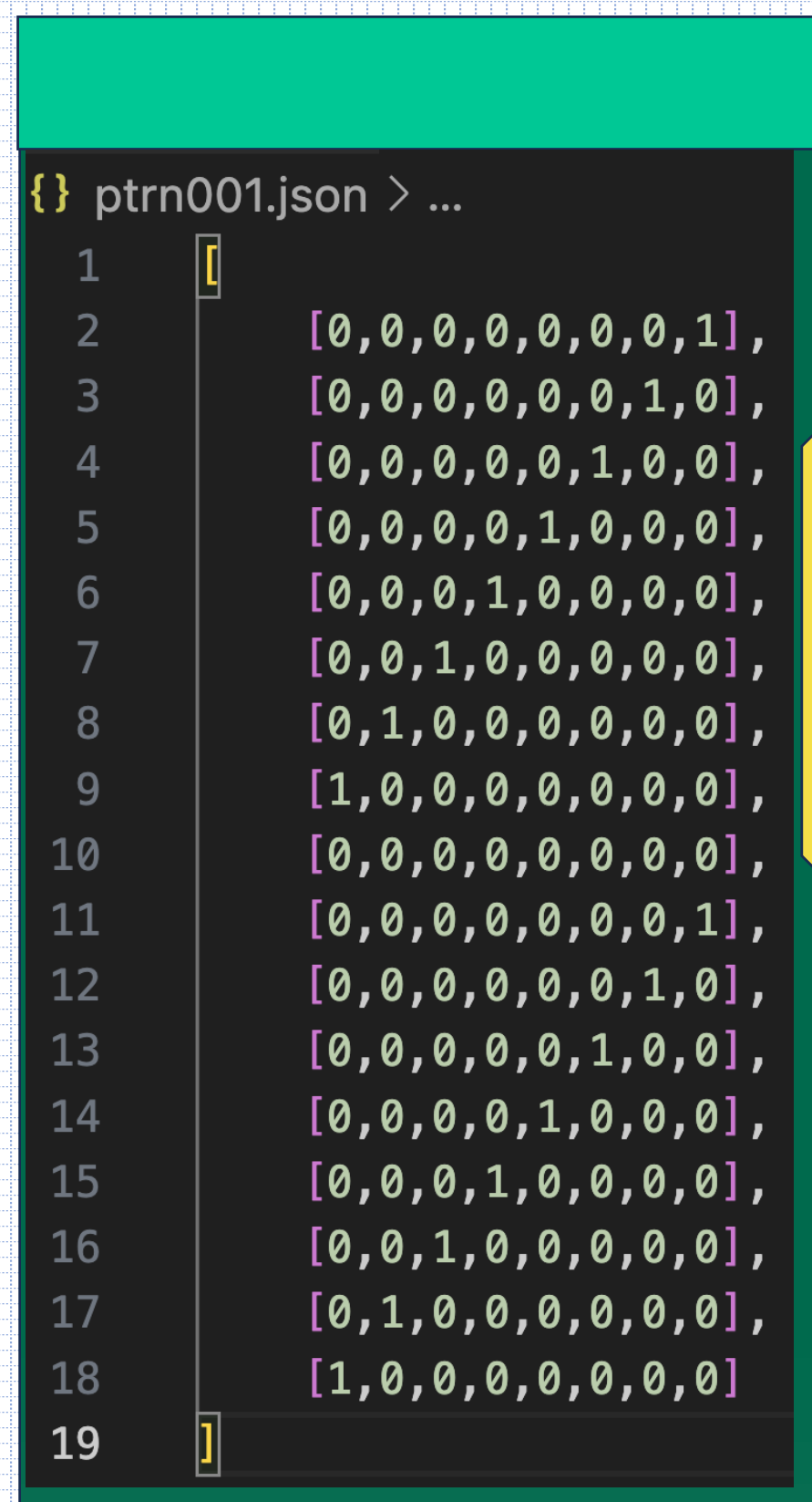
```
{
0, 0, 0, 0, 0, 0, 0, 1
0, 0, 0, 0, 0, 0, 1, 0
0, 0, 0, 0, 0, 1, 0, 0
0, 0, 0, 0, 1, 0, 0, 0
0, 0, 0, 1, 0, 0, 0, 0
0, 0, 1, 0, 0, 0, 0, 0
0, 1, 0, 0, 0, 0, 0, 0
1, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 1
}
```

Figure 1. An example of one of nine patterns which the ANN must be trained to recognize and predict.

```
{
0, 0, 0, 0, 0, 0, 1, 1
0, 0, 0, 0, 1, 1, 0, 0
0, 0, 1, 1, 0, 0, 0, 0
1, 1, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 1, 1
0, 0, 0, 0, 1, 1, 0, 0
0, 0, 1, 1, 0, 0, 0, 0
1, 1, 0, 0, 0, 0, 0, 0
0, 0, 0, 0, 0, 0, 0, 0
}
```

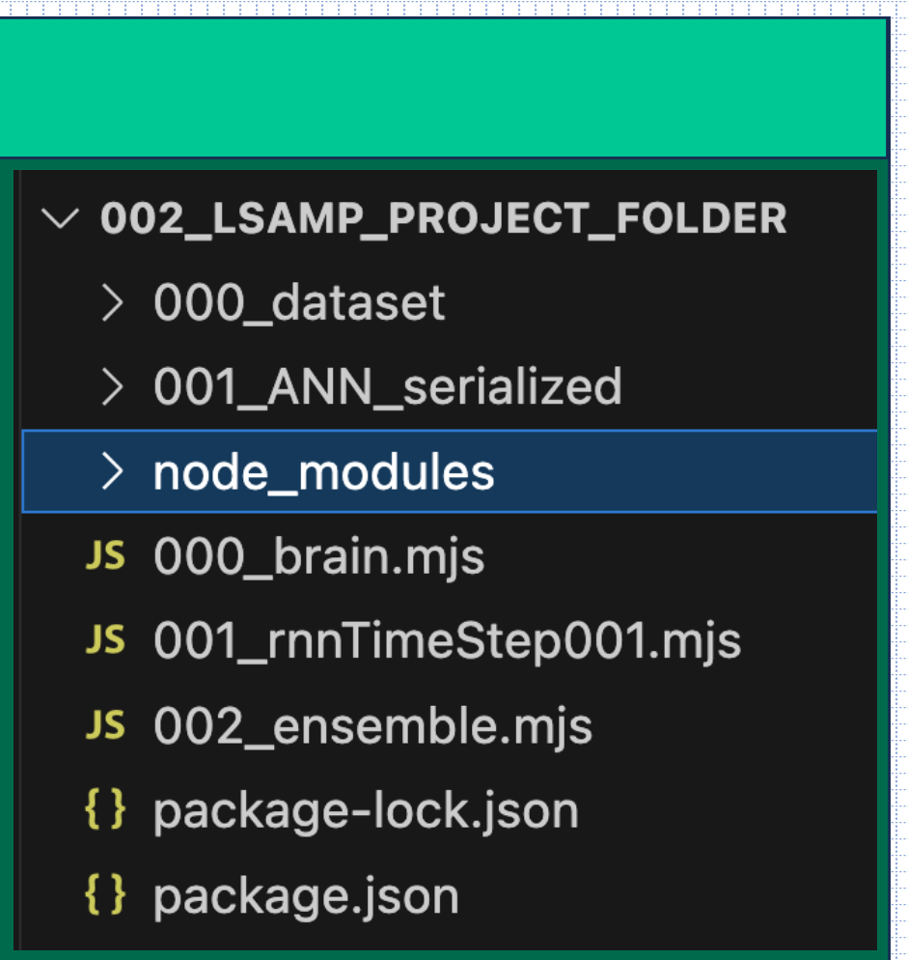
Figure 2. An example of one of nine patterns which the ANN must be trained to recognize and predict.

Methodology

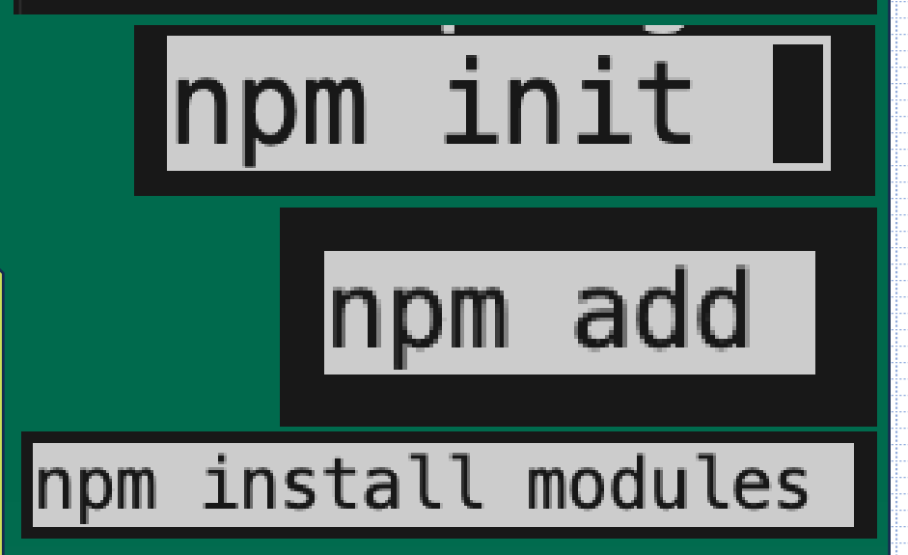


```
{} ptrn001.json > ...
[
    [0,0,0,0,0,0,0,1],
    [0,0,0,0,0,0,1,0],
    [0,0,0,0,0,1,0,0],
    [0,0,0,0,1,0,0,0],
    [0,0,0,1,0,0,0,0],
    [0,0,1,0,0,0,0,0],
    [0,1,0,0,0,0,0,0],
    [1,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,1],
    [0,0,0,0,0,0,1,0],
    [0,0,0,0,0,1,0,0],
    [0,0,0,0,1,0,0,0],
    [0,0,0,1,0,0,0,0],
    [0,0,1,0,0,0,0,0],
    [0,1,0,0,0,0,0,0],
    [1,0,0,0,0,0,0,0]
]
```

Converting data into a JSON format dataset and separating the dataset into two data set partitions.



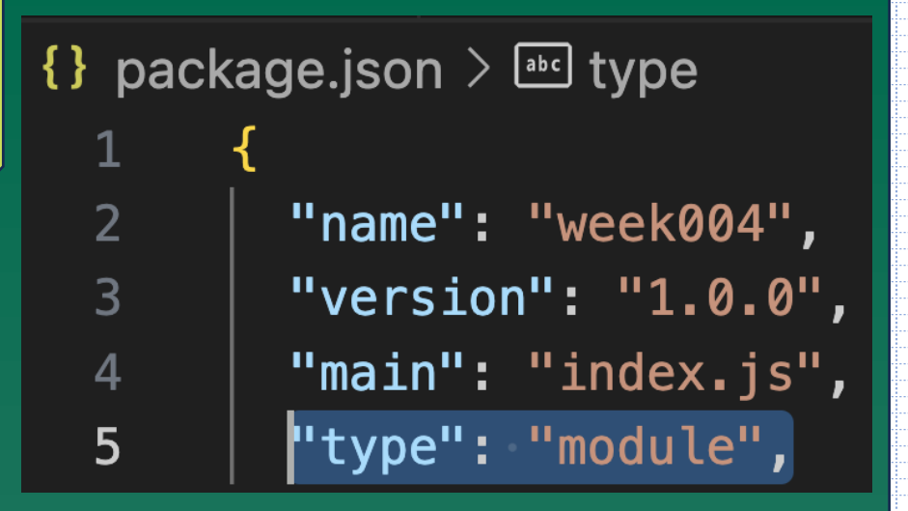
```
002_LSAMP_PROJECT_FOLDER
  > 000_dataset
  > 001_ANN_serialized
  > node_modules
  JS 000_brain.mjs
  JS 001_rnnTimeStep001.mjs
  JS 002_ensemble.mjs
  {} package-lock.json
  {} package.json
```



```
npm init
```

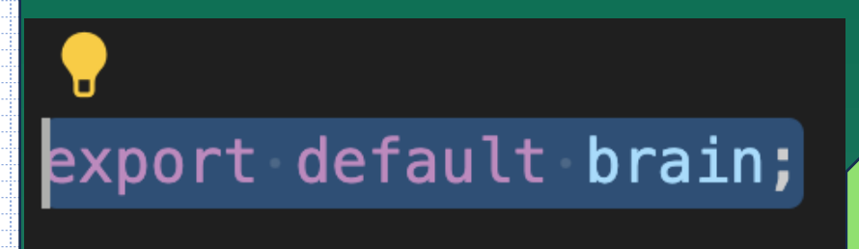
```
npm add
```

```
npm install modules
```



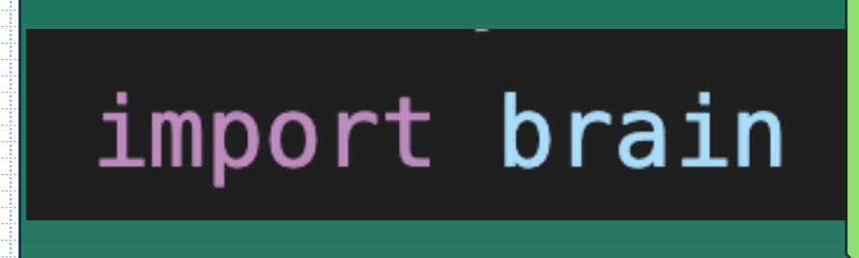
```
{} package.json > type
{
  "name": "week004",
  "version": "1.0.0",
  "main": "index.js",
  "type": "module",
```

Setting up a server side node.js JavaScript environment on the terminal with the ES6 modules importing system.



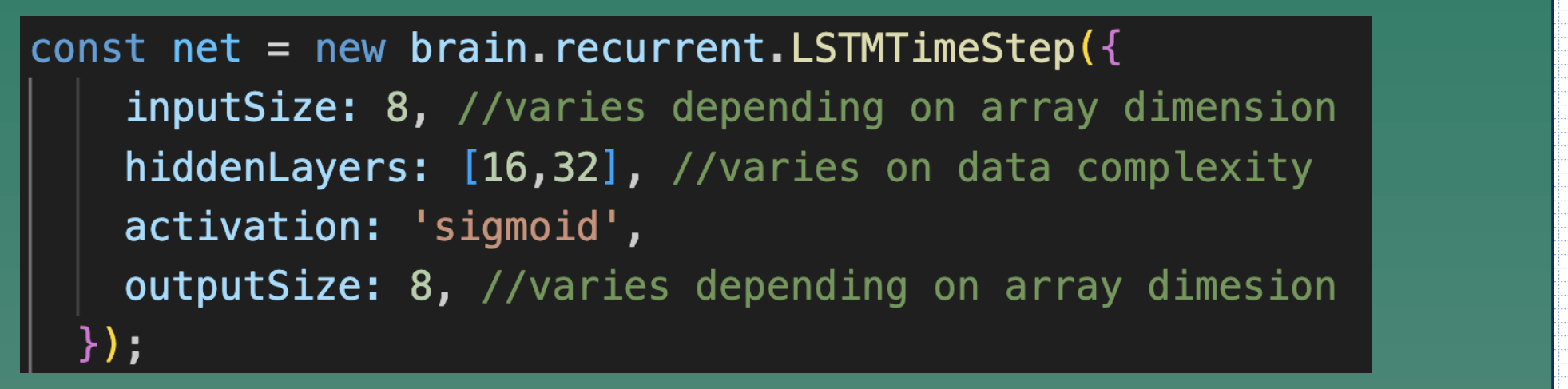
```
export default brain;
```

Including the brain.js library in the environment. Adding "export default brain" line of code to the brain.js library source code.



```
import brain
```

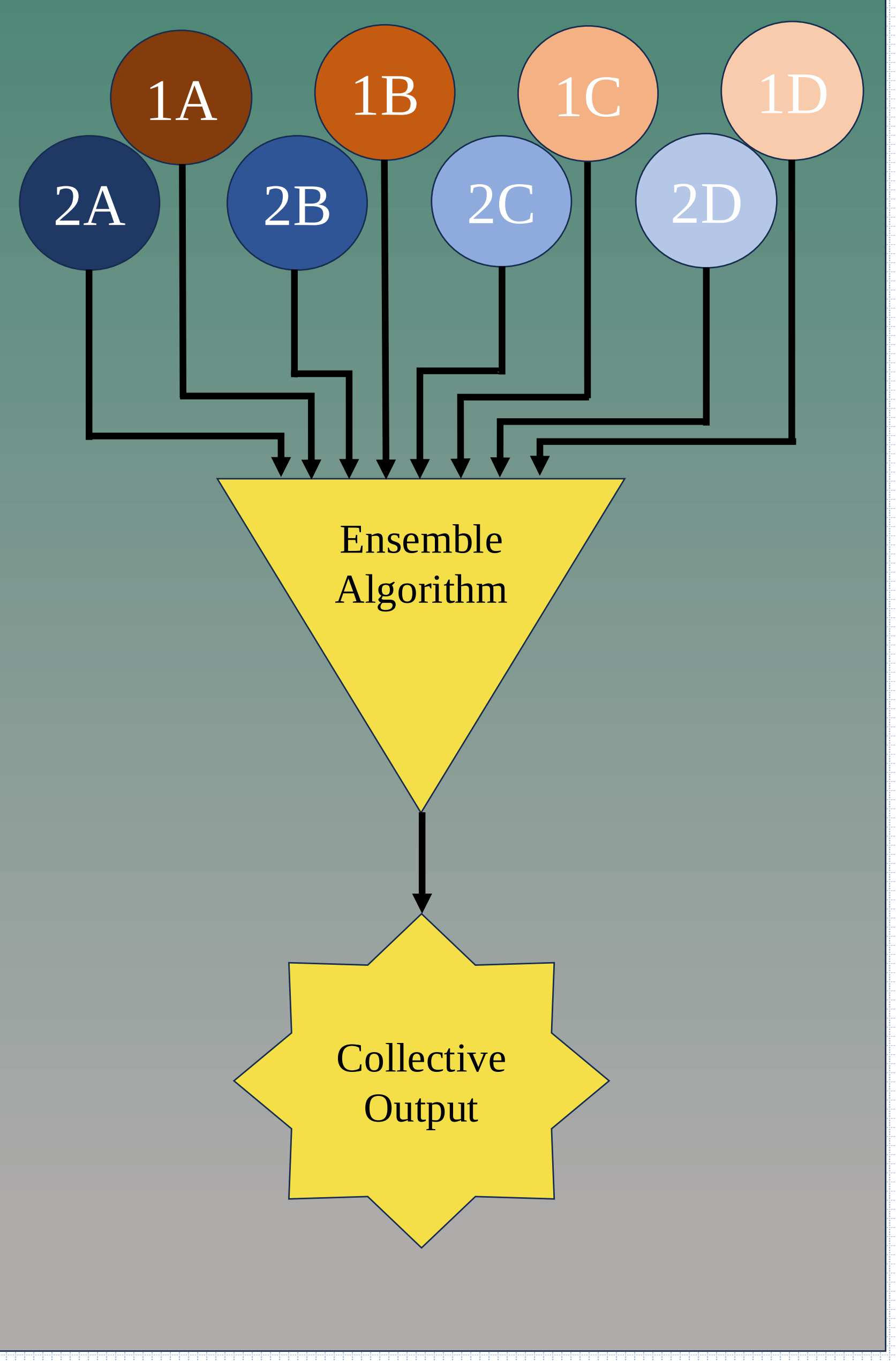
Importing brain.js to implement LSTM (Long-Short-Term-Memory) Time Step ANN models in JavaScript.



```
const net = new brain.recurrent.LSTMTimeStep({
    inputSize: 8, //varies depending on array dimension
    hiddenLayers: [16,32], //varies on data complexity
    activation: 'sigmoid',
    outputSize: 8, //varies depending on array dimesion
});
```

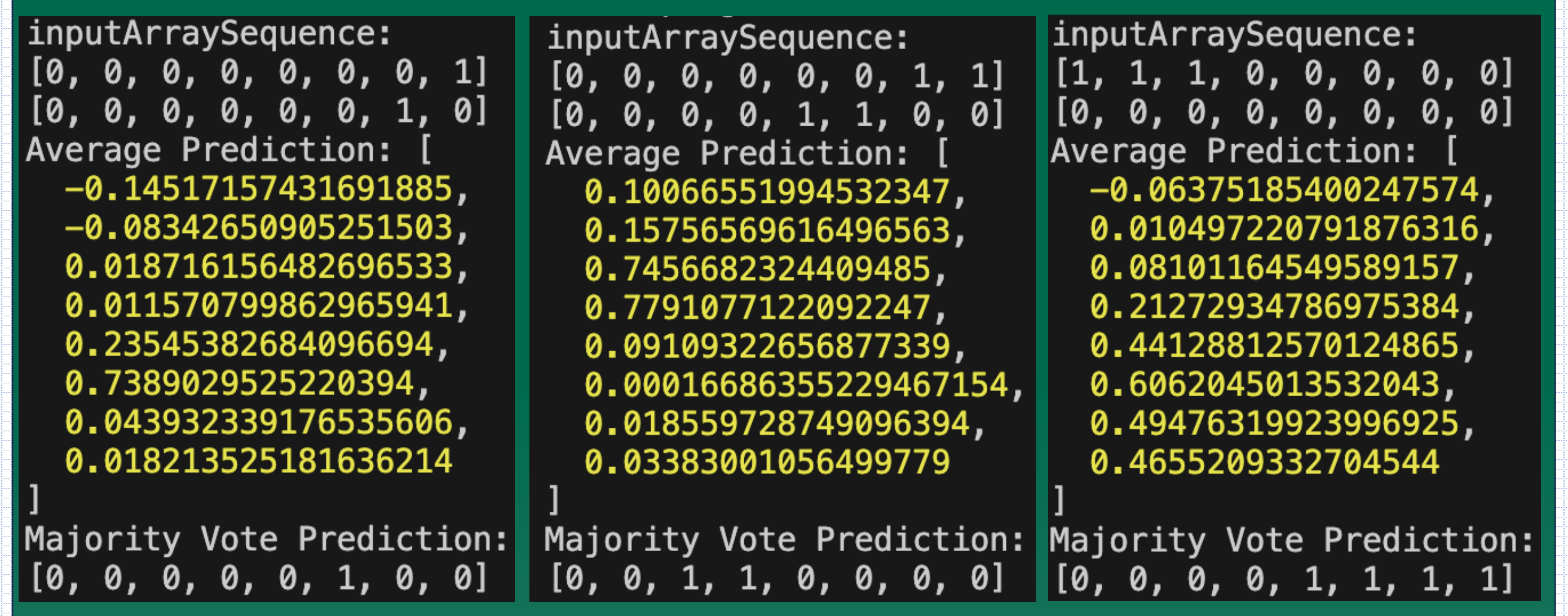
Individually training a set of ANNs at different levels of convergence and serializing variant models (1A, 1B…2D) on each dataset partition.

Creating an averaging and voting ensemble algorithm to group and analyze the collective output of all serialized models upon input.



Analyzing the ANN system's performance upon input of trained and untrained data.

Examples of Results on Trained Data

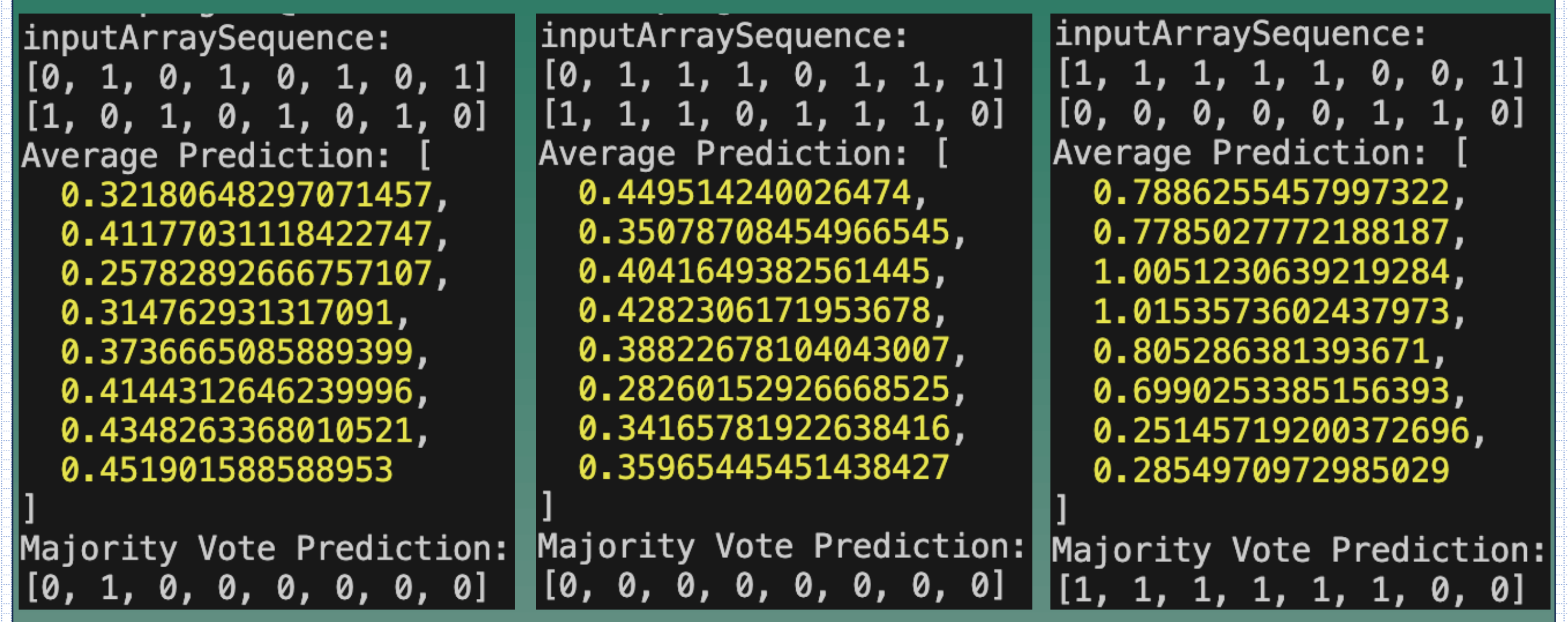


```
inputArraySequence:
[0, 0, 0, 0, 0, 0, 0, 1]
[0, 0, 0, 0, 0, 0, 1, 0]
Average Prediction: [
  -0.14517157431691885,
  -0.08342650905251503,
  0.018716156482696533,
  0.011570799862965941,
  0.23545382684096694,
  0.7389029525220394,
  0.043932339176535606,
  0.018213525181636214
]
Majority Vote Prediction:
[0, 0, 0, 0, 0, 1, 0, 0]
```

```
inputArraySequence:
[0, 0, 0, 0, 0, 0, 1, 1]
[0, 0, 0, 0, 1, 1, 0, 0]
Average Prediction: [
  0.10066551994532347,
  0.15756569616496563,
  0.7456682324409485,
  0.7791077122092247,
  0.09109322656877339,
  0.00016686355229467154,
  0.018559728749096394,
  0.03383001056499779
]
Majority Vote Prediction:
[0, 0, 1, 1, 0, 0, 0, 0]
```

```
inputArraySequence:
[1, 1, 1, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0]
Average Prediction: [
  -0.06375185400247574,
  0.010497220791876316,
  0.08101164549589157,
  0.21272934786975384,
  0.44128812570124865,
  0.6062045013532043,
  0.49476319923996925,
  0.4655209332704544
]
Majority Vote Prediction:
[0, 0, 0, 0, 1, 1, 1, 1]
```

Examples of Results on Un-Trained Data



```
inputArraySequence:
[0, 1, 0, 1, 0, 1, 0, 1]
[1, 0, 1, 0, 1, 0, 1, 0]
Average Prediction: [
  0.32180648297071457,
  0.41177031118422747,
  0.25782892666757107,
  0.314762931317091,
  0.3736665085889399,
  0.4144312646239996,
  0.4348263368010521,
  0.451901588588953
]
Majority Vote Prediction:
[0, 1, 0, 0, 0, 0, 0, 0]
```

```
inputArraySequence:
[0, 1, 1, 1, 0, 1, 1, 1]
[1, 1, 1, 0, 1, 1, 1, 0]
Average Prediction: [
  0.449514240026474,
  0.35078708454966545,
  0.4041649382561445,
  0.4282306171953678,
  0.38822678104043007,
  0.28260152926668525,
  0.34165781922638416,
  0.35965445451438427
]
Majority Vote Prediction:
[0, 0, 0, 0, 0, 0, 0, 0]
```

```
inputArraySequence:
[1, 1, 1, 1, 1, 0, 0, 1]
[0, 0, 0, 0, 0, 1, 1, 0]
Average Prediction: [
  0.7886255457997322,
  0.7785027772188187,
  1.0051230639219284,
  1.0153573602437973,
  0.805286381393671,
  0.6990253385156393,
  0.25145719200372696,
  0.2854970972985029
]
Majority Vote Prediction:
[1, 1, 1, 1, 1, 1, 0, 0]
```

Conclusion

Upon input of sequence data which the model had been trained upon, the model predicted mostly correct values with medium certainty. Upon input of sequence data which the model had not be trained upon, the model predicted values which held partial binary identity (right amount of 0s and 1s or right placements yet not aligning with new input pattern) meaning that the model in this example is more underfit than overfit and yet had the ability recognize its own training data set and not break or output noise upon input of untrained data and output creative predictions. These are characteristics of a generalized model.

References

1. What is ensemble learning? | IBM.www.ibm.com. Published February 9, 2024. https://www.ibm.com/topics/ensemble-learning
2. BrainJS/brain.js. GitHub. Published July 14, 2024. https://github.com/BrainJS/brain.js#brainjs

Acknowledgements

A special thanks to:
Theresa M. Nawalaniec, Justin K. Simmerer, Dr. Ángel L. Reyes-Rodriguez, Amber Finley, Dr. Adam T Sonstegard,

And my Mom.

Funding: Louis Stokes Alliance for Minority Participation at Cleveland State University