



# Welcome to the **Co**Grammar Tutorial: Neural Networks

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.



## Data Science Session Housekeeping

---

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.  
**(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: [Questions](#)

## Data Science Session Housekeeping cont.

---

- For all **non-academic questions**, please submit a query: [www.hyperiondev.com/support](https://www.hyperiondev.com/support)
- Report a **safeguarding** incident: [www.hyperiondev.com/safeguardreporting](https://www.hyperiondev.com/safeguardreporting)
- We would love your **feedback** on lectures: [Feedback on Lectures](#)

# Skills Bootcamp

## 8-Week Progression Overview

### Fulfil 4 Criteria to Graduation

#### ✓ Criterion 1: Initial Requirements

Timeframe: First 2 Weeks

Guided Learning Hours (GLH):

Minimum of 15 hours

Task Completion: First four tasks

**Due Date: 24 March 2024**

#### ✓ Criterion 2: Mid-Course Progress

**60** Guided Learning Hours

Data Science - **13 tasks**

Software Engineering - **13 tasks**

Web Development - **13 tasks**

**Due Date: 28 April 2024**

# Skills Bootcamp Progression Overview

## ✓ Criterion 3: Course Progress

Completion: All mandatory tasks,  
including Build Your Brand and  
resubmissions by study period end  
Interview Invitation: Within 4 weeks  
post-course  
Guided Learning Hours: Minimum of  
112 hours by support end date  
(10.5 hours average, each week)

## ✓ Criterion 4: Demonstrating Employability

Final Job or Apprenticeship  
Outcome: Document within 12  
weeks post-graduation  
Relevance: Progression to  
employment or related  
opportunity

# CoGrammar

## Tutorial: Neural Networks

May 2024



# Learning Objectives

- ❖ Recap of **neural network** components and layers.
- ❖ Recap of **backpropagation, gradient descent, loss functions**, and **regularisation** techniques.
- ❖ Implementing **Keras** and **TensorFlow** for neural networks.

# Recap of Neural Networks





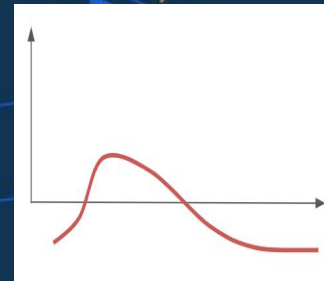
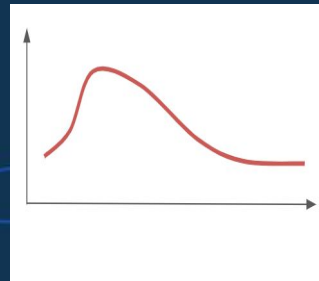
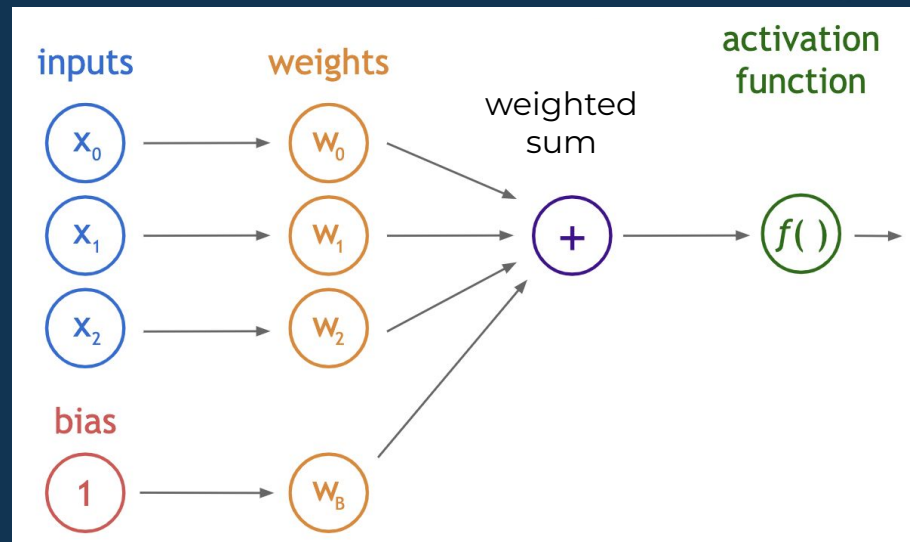
# Neural Network Components

**Inputs:** set of features that are fed into the model for learning process.

**Weights:** give importance to those features that contribute more towards learning

**Activation function:** introduces non-linearity

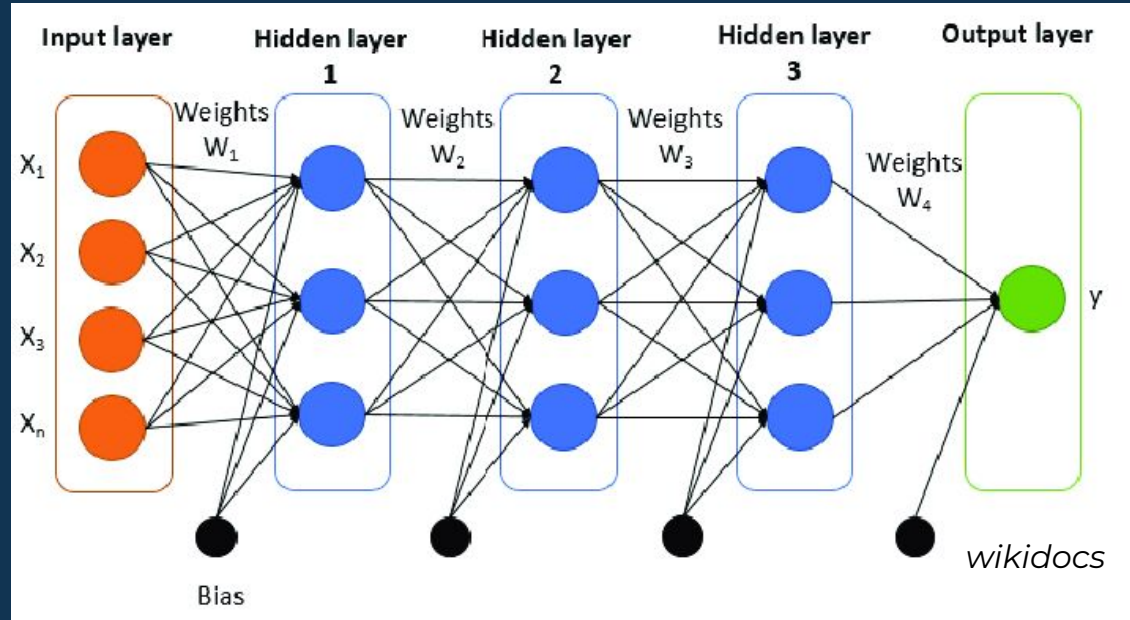
**Bias:** shift the value produced by the activation function



# Neural Network Layers

**Input layer:** sends the data to subsequent layers, no. of nodes depend on features

**Hidden layer/s:** weighted inputs fed into these intermediate layers for computations; extracts features from the data.



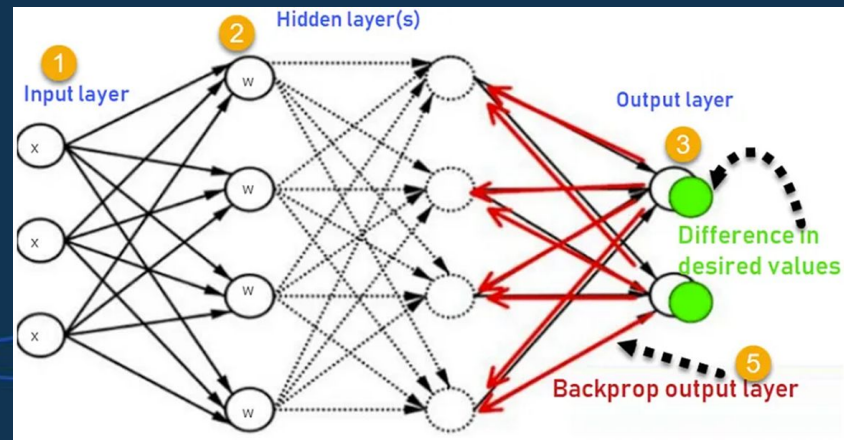
**Output layer:** takes input from preceding hidden layers and comes to a final prediction based on the model's learnings

# Backpropagation

- ❖ Computes the **gradient** of the **loss function (cross-entropy, MSE, MAPE)** with respect to the network weights layer by layer, and iterating backward from the last layer.
- ❖ Efficiently compute the gradient concerning each weight, to train multi-layer networks and **update weights** to **minimise loss**.
- ❖ **Gradient descent** or **stochastic gradient descent** estimation method.

Weights updated using **optimal learning rates** and the **gradients**.

$$w_{\text{new}} = w_{\text{old}} - \alpha \frac{\delta J}{\delta w}$$

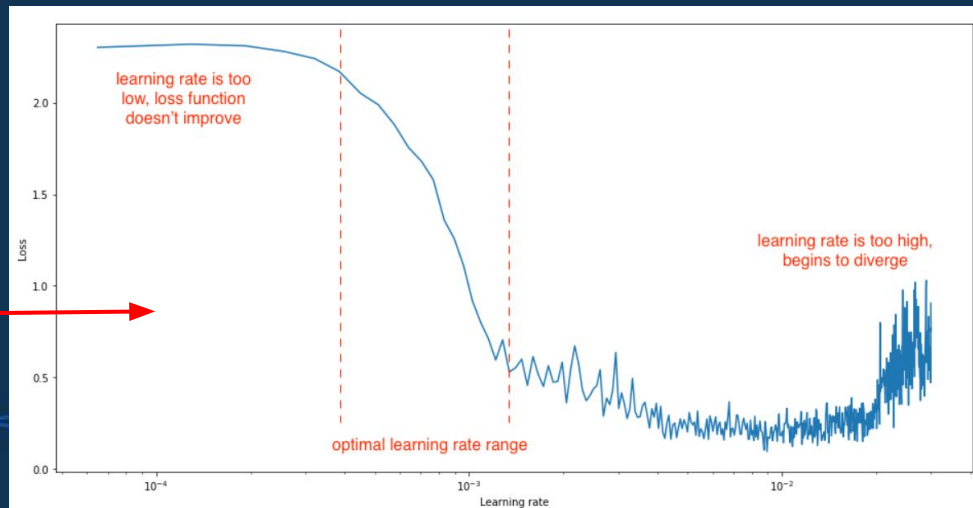


# Learning Rates

**High learning rates:** result in larger steps but risks overshooting the minimum.

**Low learning rate:** small step sizes, more precision, but compromises overall efficiency, more iterations takes more time and computations to reach the minimum.

Set learning rate bounds to observe all three phases, making the optimal range trivial to identify.



# Regularisation Techniques

Regularisation techniques improve neural network's generalisation ability by reducing overfitting.

- ❖ **Early Stopping:** Prevent training loss from becoming arbitrarily low, model is less likely to overfit on training dataset, and will generalise better.
- ❖ **Data Augmentation:** Increase the training data sample or more diverse training examples to prevent overfitting.
- ❖ **L1 and L2 regularisation:** update the general cost function by the regularisation term.
- ❖ **Dropout:** removing random nodes at each iteration.



# Keras Architecture

## Model, Layer, Core Modules



- ❖ **Sequential model** (Functional API for more complex models)
- ❖ Add **dense layer** (Dense API) with **relu activation** (using Activation module).
- ❖ Add a **dropout layer** (Dropout API) to handle over-fitting.
- ❖ Add final dense layer (Dense API) with **softmax activation** (using Activation module) function.

```
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout

model = Sequential()
model.add(Dense(512, activation = 'relu', input_shape = (784,)))
model.add(Dropout(0.2))
model.add(Dense(num_classes, activation = 'softmax'))
```

# Recap Q&A







**A 4-input neuron has weights 1, 2, 3 and 4. The inputs are 4, 10, 5 and 20 respectively. The output will be:**


1. 238
2. 119
3. 42
4. 123






A 4-input neuron has weights 1, 2, 3 and 4. The inputs are 4, 10, 5 and 20 respectively. The output will be:

1. 238
  2. **119** ( $1*4 + 2*10 + 3*5 + 4*20 = 4 + 20 + 15 + 80 = 119$ )
  3. 42
  4. 123
- 




**Can the equation  $y = ax^2 + bx + c$  be represented by a neural network of single hidden layer with linear threshold?**

1. Yes, hidden layers can help in incorporating non-linearity.
  2. Yes, a linear threshold within a hidden layer can represent higher order polynomials.
  3. No, having a linear threshold restricts the neural network to a linear transformation function.
  4. No, multiple hidden layers are needed for this.
- 





Can the equation  $y = ax^2 + bx + c$  be represented by a neural network of single hidden layer with linear threshold?

1. Yes, hidden layers can help in incorporating non-linearity.
  2. Yes, a linear threshold within a hidden layer can represent higher order polynomials.
  - 3. No, having a linear threshold restricts the neural network to a linear transformation function.**
  4. No, multiple hidden layers are needed for this.
- 



# Which of the following incorporates non-linearity to a neural network?

1. Stochastic Gradient Descent
2. Rectified Linear Unit
3. Convolution function
4. Data Augmentation





# Which of the following incorporates non-linearity to a neural network?

1. Stochastic Gradient Descent
- 2. Rectified Linear Unit**
3. Convolution function
4. Data Augmentation





# Which of the following is not the promise of an artificial neural network?


1. It can explain the result
2. It can survive the failure of some nodes
3. It has inherent parallelism
4. It can handle noise




# Which of the following is not the promise of artificial neural network?

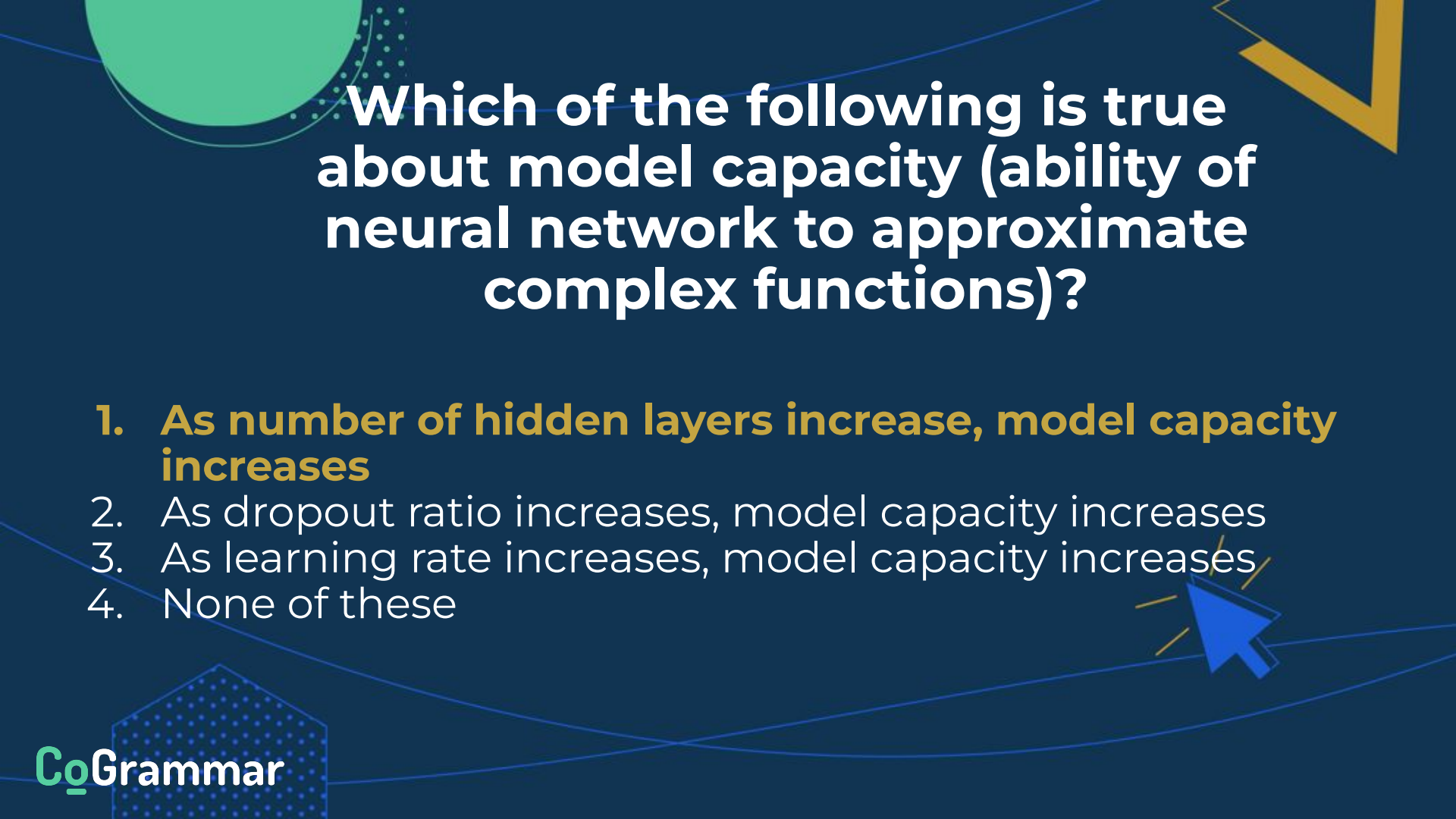
1. **It can explain the result**
2. It can survive the failure of some nodes
3. It has inherent parallelism
4. It can handle noise





# Which of the following is true about model capacity (ability of neural network to approximate complex functions)?

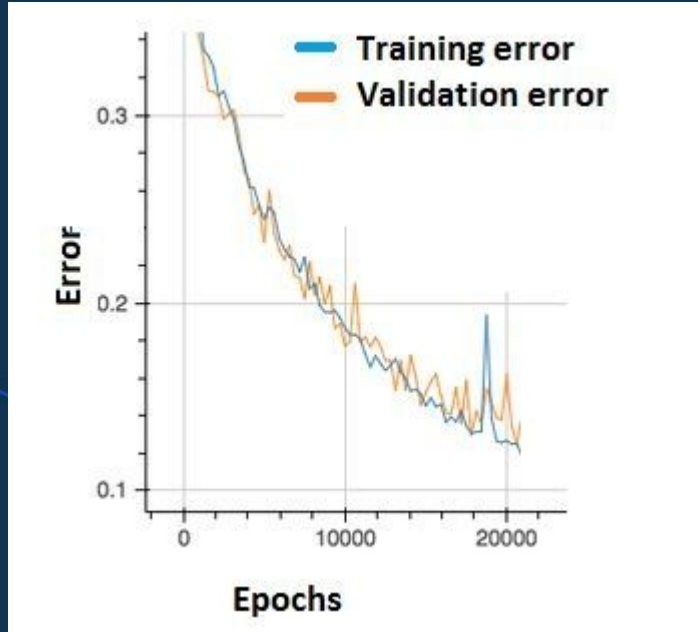
1. As number of hidden layers increase, model capacity increases
  2. As dropout ratio increases, model capacity increases
  3. As learning rate increases, model capacity increases
  4. None of these
- 



# Which of the following is true about model capacity (ability of neural network to approximate complex functions)?

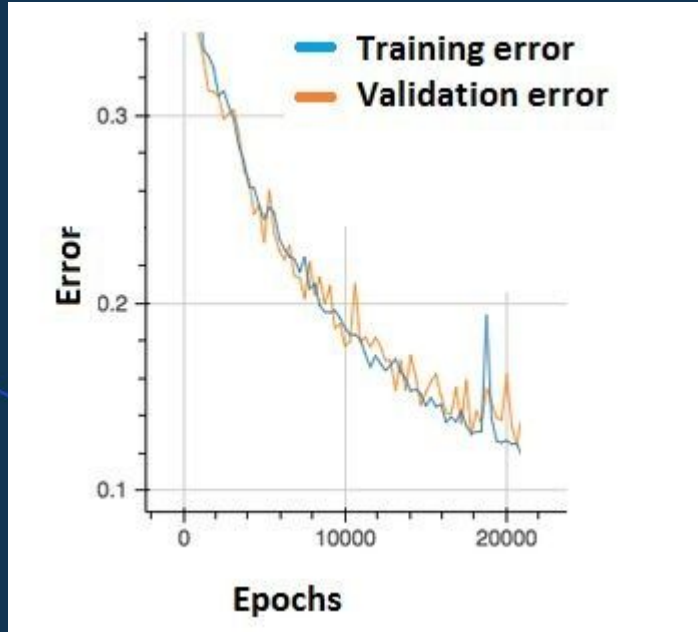
1. **As number of hidden layers increase, model capacity increases**
2. As dropout ratio increases, model capacity increases
3. As learning rate increases, model capacity increases
4. None of these

In the plot, we observe that the error has many “ups and downs”. Is it a problem?



1. Yes, because this means there is a problem with the learning rate of neural network.
2. No, as long as there is a cumulative decrease in both training and validation error, we don't need to worry.

In the plot, we observe that the error has many “ups and downs”. Is it a problem?



1. Yes, because this means there is a problem with the learning rate of neural network.
2. **No, as long as there is a cumulative decrease in both training and validation error, we don't need to worry.**

Increasing the batch size will reduce these.

# Summary





# Key Takeaways

Architecture of a neural network is challenging, time complexity of neural networks is very high to iterate over all combinations of **hyperparameters**.

- ❖ **Number of Layers** increase the depth of the neural network, and also the ability to learn more complex features.
- ❖ **Number of Nodes** in the **first hidden layer** and the **last hidden layer** must be equal to the number of input **features** and **classes** to predict. **For the hidden layer**, convention to use 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 nodes (hardware performs efficiently when numbers stored in powers of two), although no proof that this is the most optimal way.
- ❖ **Activation Function:** introduce nonlinearity on each node, E.g. ReLU, Sigmoid, and Leaky ReLU.

# Key Takeaways

- ❖ **Batch Size:** optimal subsample of the dataset to calculate the gradient and update the weights, iterate over all to cover the whole dataset. Lesser batch size causes more fluctuations while reaching the minima, and a greater value can cause memory errors.
- ❖ **Loss Function:** Cross-entropy loss function for multi-class classification and MSE, MAPE for regression and predictive analysis, each with internal hyperparameters to be tuned.
- ❖ **Optimiser:** Apart from gradient descent method, Adagrad, Adam Optimizer exists with various hyperparameters.
- ❖ **Regularisation:** hyperparameters for early stopping, data augmentation, L1 and L2 regularisation, dropout, and batch normalisation.

# Questions and Answers



# Thank you for attending



Department  
for Education

CoGrammar

