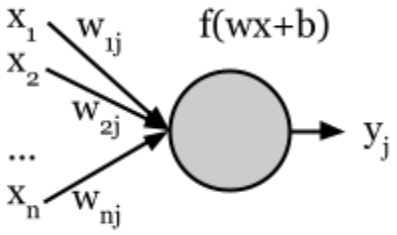
Neural Network	start time:

Before you start, share this document with your team member(s) and then complete the form
below to assign the role of speaker.

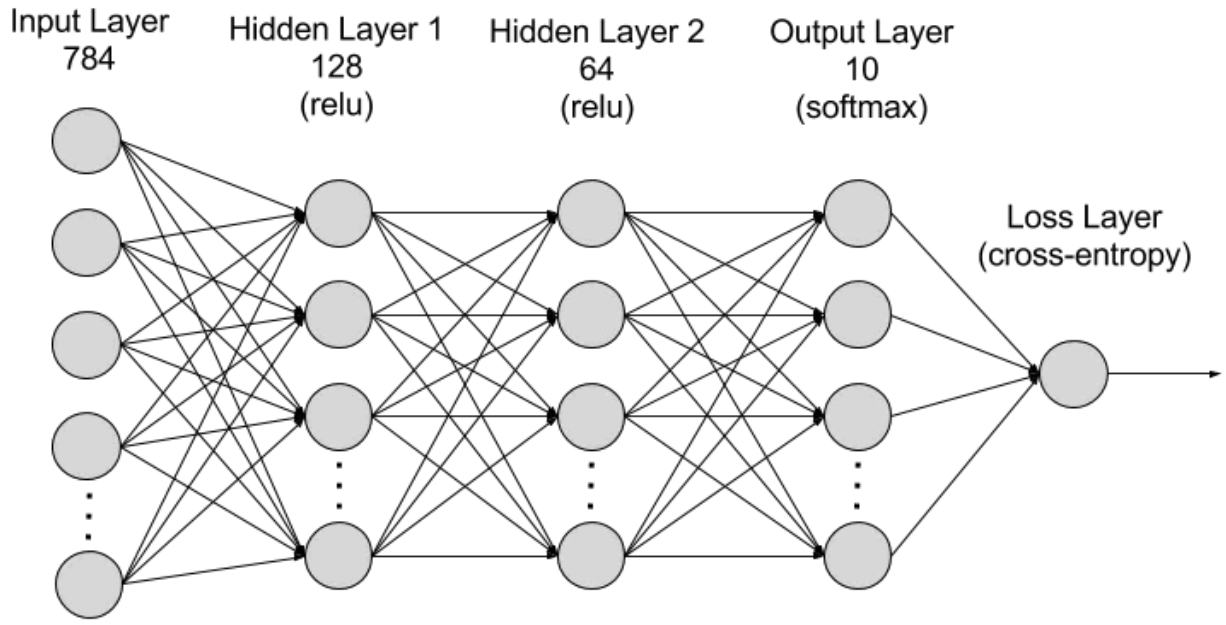
Team Role	Team Member
Speaker: shares your team's ideas with the class.	Arogya, Lauren, Tim

A. Structure of the Network	start time:

A neuron:



A network:



Model: A simple feedforward neural network includes neurons connected through weighted edges, arranged in layers: input, hidden, and output.

Questions:

1. What is a neuron in a neural network?

- Fundamental unit of computation
- A function that takes weighted output from many other neurons and produces an output

2. In the activation function f(wx+b), w is the weight. What is the role of weights in a neural network?

- Determines which output to consider more and which to consider less while deciding output for that neuron

- Prioritizing the inputs
- Summarizing inputs from previous layers

3. In the activation function f(wx+b), what is the purpose of the bias term b?

- Important initial value, shift/adjust the entire input (all input values)
- Improves the flexibility of the network
- So that there is always a default value even if we don't consider the output from previous neuron
- w is for single input b is for all the inputs, therefore we need both w and b

4. How does the input layer differ from the hidden layer?

- Hidden under abstraction?
- Input layer directly receives input from the outside world, hidden layers only receive within the network
- There is only one input layer but as many hidden layers as we want
- All hidden layers receive input data through network/pre-processed data

5. How many neurons are in the input layer for a dataset with 5 features?

- 5: one neuron for each feature

6. What does the output layer represent in a regression problem?

- Only one neuron on output layer
- Regression: Predict singular continuous value
- Equation of regression function or the relation between the dependent and independent variable

7. What does the output layer represent in a classification problem with 3 classes?

- Classification: grouping of values
- 3 output neurons representing each class and output is probability of the input belonging to each of the classes (sum of the three outputs is 1)

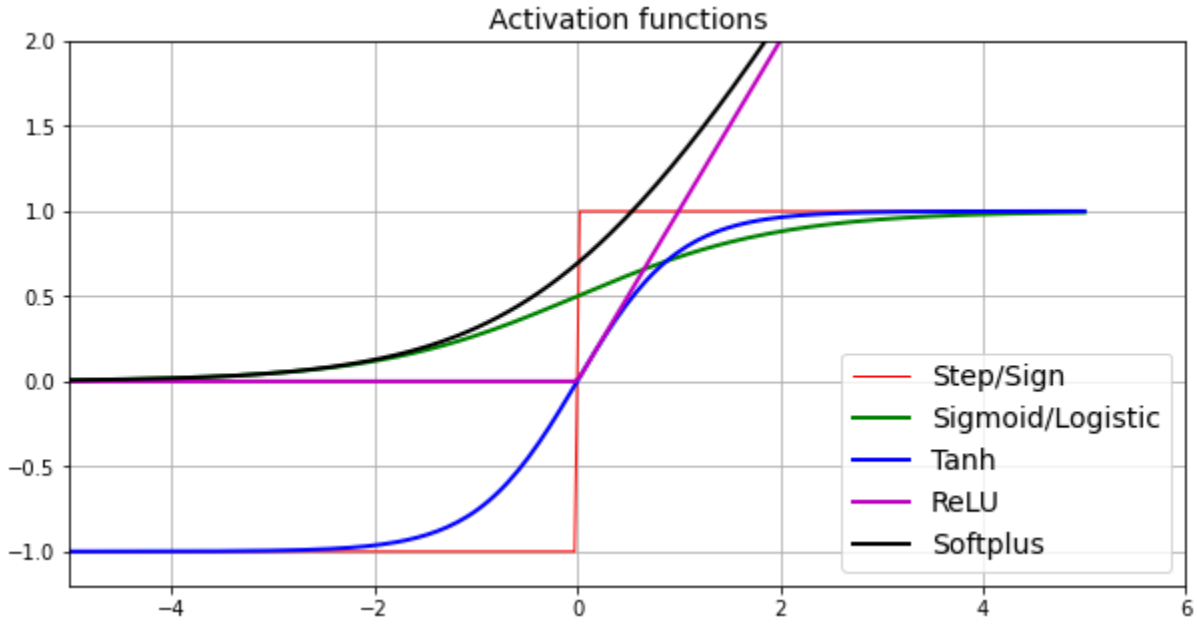
8. What shape are the weights connecting 3 input neurons to 4 hidden neurons?

- 16

9. How is data passed from one layer to the next in the term of calculation?

- Matrix multiplication
- The output is then passed from layer to another layer

B. Activation Functions	start time:



Model: Different activation functions introduce non-linearity: step, sigmoid, ReLU, softplus, softmax.

Questions:

10. What is the purpose of using activation functions in the network?

- Activation functions translate input data into meaning/gives output meaning
- Without activation function, the network would be a linear model that doesn't learn anything
- Improve power of network to handle complex tasks/learn complex patterns and relationships in the data

11. What does the step function output?

- Binary output (0 or 1)
- Outputs true or false

12. Sigmoid function is

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

What is the range of the sigmoid function? What kind of task could be approximated using sigmoid function?

- Range [0, 1]
- Probability for binary classification - to give probability of certain task
- From close to 0 up to close to 1

13. ReLU (Rectified Linear Unit) is max(0,z). What is ReLU and its main advantage?

- f(z) = max(0, z)
- Outputs for positive values linear relationship (same input/output for positive)
- Outputs for negative values 0
- ReLU is a piecewise function, nonlinear
- Main advantage: easy to calculate, learns quickly from fast calculation
    - Gradient = slope of activation function
        - Correlates to the rate of learning
        - Gradient is steep for ReLU means it learns/improves quickly

14. Compare the curve between ReLU and softplus, where the softplus equation is

$$\text{softplus}(x) = \ln(1 + e^x)$$

- Curve is steeper earlier for softplus than for ReLU
- Differentiable at every point
- Gradient is better earlier than ReLU - learns at a faster rate quicker

15. Softmax is

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

where zi is the score of each category. What is the output of softmax if there are 5 categories? (Hint: Compare softmax and sigmoid.)

- n = 5
- Sum of 5 outputs is 1
- Each output is a probability
- Calculate probabilities using softmax

16. Which activation is used for binary classification output?
    - sigmoid

17. Which activation is used for multiple classification output?
    - softmax

C. Datasets and Batching	start time:

Model:

- Training set: used to fit model (input data)
- Validation set: used to tune hyperparameters (learning rate, size of network)
- Test set: used to evaluate final performance (apply network on testing set to evaluate model)
- Batch: subset of training data (divide into batches to decide how large of a set to run model on)
- Epoch: one full pass through training data

Questions:

18. Why do we split data into training, validation, and test sets?

- If we train, validate and test on same data then it will always be 100% accurate and we will never know if it actually works on other datasets.
- To remove bias in model and improve overfitting

19. Does the model see all training samples in each epoch?

- Yes each epoch will see all samples in batch

20. If batch size is 100 and dataset has 1000 samples in the training set, how many batches per epoch?

- 10 batches
- Size of batches 2^n

21. Why use batches?

- You can update more frequently in batches
- Smoother updates and smaller jumps in resulting values
- Parallel computation

22. If validation accuracy drops while training accuracy improves, what's happening?

- The model could be overfitting so the training must be stopped

23. What's the danger of not using a test set?

- Overfitting, will not perform well on unseen/new data if not receiving the proper evaluation of

the model’s performance through a test set

24. Early stopping means stopping training early. When could we use early stopping?

- Prevent overfitting
- Save computational resources
- Stop before model’s performance on test set starts to decrease

D. Loss Functions	start time:

Model:

A loss function is a mathematical function that measures how wrong a model's predictions are compared to the true labels.

Loss=Difference between predicted output and actual label

- For a single example → we get a loss
- For the whole dataset → we compute average loss (cost)

The goal of training is to minimize the loss by adjusting the model's weights using backpropagation and an optimizer.

Here are common loss functions:

- Regression:
  - MSE: Mean squared error.
  - MAE: Mean absolute error.
- Classification:
  - For binary: Binary Cross-entropy.
  - For multi-class: Categorical Cross-Entropy.

Questions:

25. MSE is

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

where y_i is the actual value and y_i_hat is the predicted value.

What does MSE penalize in a regression task?

- It is the average of square of distance between the prediction data and the actual data
- the square helps us remove the negative numbers
- Square also penalizes larger errors more severely, sensitive to outliers

26. MAE is

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Which is more sensitive to outliers: MSE or MAE?

MSE is more sensitive to outliers because it is taking square which increases the sensitivity by power of 2

27. Which one is more common in the regression tasks, MSE or MAE?

MSE - squared nature of equation helps to not have outliers mess up your calculation of the average or estimation

MSE has more smoother function

MAE is not differentiable

28. In following regression tasks, which one may work better, MSE or MAE?
    - General regression like house prices
        - MSE
    - Noisy data with lots of outliers
        - MAE
    - Robust models(e.g. using median)
        - MAE

29. For a prediction y and true value y_{true}, the Huber loss is defined as:

$$L(y, y_{\text{true}}) = \begin{cases} \frac{1}{2}(y - y_{\text{true}})^2 & \text{if } |y - y_{\text{true}}| \leq 1 \\ |y - y_{\text{true}}| - \frac{1}{2} & \text{otherwise} \end{cases}$$

    a. What does Huber loss look like when the error is small, MSE or MAE?
        i. MSE (follows top part of L(y, y_{true}) equation)
    b. What does Huber loss look like when the error is big, MSE or MAE?
        i. MAE(follows bottom part of equation)
    c. What is the benefit when using Huber loss for regression tasks?

i. Differentiable at every point, not sensitive to outliers AND not sensitive to small differences

30. What does classification error measure? Is it a good idea of using classification error as the loss function for classification tasks?

- Classification error just gives yes or no
- The output should be a probability instead of directly saying 100% chance of this or that
- Therefore, it is not a good idea to use classification error
- The design of the loss function will be very simple

31. Cross-entropy loss measures the difference between predicted and actual probability distributions. For binary classification, we often use binary cross-entropy:

$$-[y \cdot \log(\hat{y}) + (1-y) \cdot \log(1-\hat{y})]$$

where y is the actual label 0 or 1 and y_hat is the predicted probability.

Suppose the true label is y=[0,1,0] and the predicted probabilities is y_hat=[0.1,0.8,0.1]. What is the loss?

- - [y * log(y^) + (1 - y) * log (1 - y^)]
- - [0 * log(0.1) + (1 - 0) * log (1 - 0.1)] = -log(0.9)
- - [1 * log(0.8) + (1 - 1) * log (1 - 0.8)] = -log(0.8)
- - [0 * log(0.1) + (1 - 0) * log (1 - 0.1)] = -log(0.9)
- (-log(0.9) - log(0.8) - log(0.9))/3 = **0.063**
- Prediction right -> loss small
- Prediction wrong -> loss large

32. For multi-class classification task, we often use Categorical Cross-Entropy:

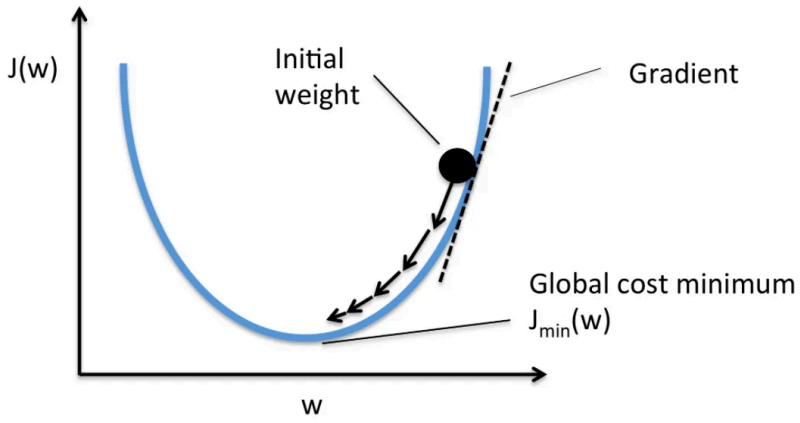
$$-\sum_{i=1}^{n} y_i \cdot \log(\hat{y}_i)$$

where y_i is the true label with one-hot coding and y_i hat is the predicted probability for class i. Is the binary cross entropy a special case of multi-class formula when n is 2?

- yes.

33. What is the ideal value for loss when the network model is perfect?
    0. loss cannot be negative. the smaller the loss better the model.

E. Training the Network - Gradient Descent and Backpropagation	start time:



(https://medium.com/@ayushch612/gradient-descent-slope-3949df5dab5b)

Model: Weights are updated through backpropagation using the gradient of the loss and the learning rate.

A **gradient** is the vector of partial derivatives of the loss function with respect to each weight. It tells us the direction and rate of change of the loss if we adjust that weight.

- If the gradient is large → the weight has a big influence on the loss.
- If the gradient is small → the weight has little effect, or we are near a minimum.

Backpropagation uses the chain rule of calculus to compute gradients layer by layer from the output layer backward to the input layer.

Weight update rule (gradient descent): New weight = old weight - learning rate × gradient

$$w := w - \alpha \cdot \frac{\partial \mathcal{L}}{\partial w}$$

Where:

- w is the weight
- alpha is the learning rate
- L is the loss
- div L/W is gradient

Questions:

34. In your own words, explain what the gradient is?

- A measure of how the neural network is performing
- Gradient is slope of the function
    - The higher the slope, the faster the network can learn

- Gradient line has a direction, points in the direction of the steepest increase of cost function
    - Goal during training is to minimize the cost function
    - Follow the direction of the gradient in order to do so

35. What does it mean when the gradient is 0?

- Model's loss function is fully optimized, no more adjustments to the weights can be made
- No further improvements
- Gradient is 0, model stops learning

36. Explain the basic idea of backpropagation in your own words?

- Computing the gradients of tangents starting from output layer to input layer backwards to reduce the loss for next time
- Adjusting the weight layer to layer to remediate loss
- Practice of fine-tuning the neural network's weights based on error rate/loss obtained from the previous iteration (epoch)

37. What does the learning rate control?

- Adjust/governs the rate at which the algorithm learns/updates
- Controls how much to change the model in response to the estimated error/loss
- Key parameter on how fast you want to update your network

38. What happens if the learning rate is too large or too small?
- Learning rate is 0 means it only takes old weight, Learning rate is high means it highly considers the new gradient
- if learning rate is too large, the jump is very big from one state to other and therefore the transition will be less smooth; the weight changes too slow if learning rate is very low.

39. What happens if the gradient is too small or too big?

- Gradient near 0 means the learning is very slow
- Gradient very high means the change is very big jump from one state to another

40. What is the limitation of backpropagation?

- Training through backpropagation can be computationally complex aka time and compute power consuming
- Overfitting can occur - need to be precise when setting your hyperparameter for the learning rate in order to accurately assess what you want the network to learn vs when you want the already/preprocessed training data to take precedence
- Time consuming if learning rate is too small
- We need to make choice on many parameters to make sure it works

41. How do we initialize the weights?

- We initialize weights through
- random weight for each neuron

F. Optimizers - SGD and more	start time:

Model: Optimizers guide weight updates using gradients.

An optimizer is an algorithm used in neural networks to update the model's weights in order to minimize the loss function during training.

After computing how wrong the model is (via the loss function) and how each weight contributes to that error (via the gradient), the optimizer decides how to adjust the weights to reduce the error on the next training step.

Think of it like a GPS:

- The gradient tells you which direction is downhill (toward lower loss).
- The optimizer decides how big a step to take and how to use past steps to guide you.

SGD stands for Stochastic Gradient Descent, and it is one of the most fundamental and widely used optimizers in machine learning and neural networks.

- Gradient Descent is an algorithm that minimizes a loss function by moving in the direction of the negative gradient of the loss with respect to the model's parameters (weights). This is what we learned in the previous section.

$$w := w - \alpha \cdot \frac{\partial \mathcal{L}}{\partial w}$$

- Stochastic Gradient Descent (SGD) is a variant where you update the weights after evaluating the gradient on a single data point or a small batch instead of the whole dataset.

$$w := w - \alpha \cdot \nabla \mathcal{L}(w; x_i, y_i)$$

where we only use a single sample or a small batch of samples to calculate the gradient.

Questions:

42. In your own words, summarize the biggest difference between gradient descent and SGD.

- In GD(gradient descent) we use all data to calculate the gradient, stochastic means we use randomly, and do not use the whole data to calculate the direction and gradient.

43. What is the advantage of using SGD for training?

- We can save time if we are not using the whole dataset it is faster
- Improved memory efficiency since it doesn't require the entire data set for each iteration

44. What is the disadvantage of using SGD for training?

- Since there is less data to look at, the network will not be finely tuned for different type of data
- Noisy gradient estimates will potentially cause the cost function to fluctuate rather than steadily decrease, erratic behavior possible

Note:

Here are some other popular optimizers:

Optimizer	When to Use
SGD	When training large models with lots of data and you care about **generalization** (e.g., ResNet on ImageNet). Use with momentum.
SGD + Momentum	When SGD is too slow or oscillates — helps smooth and speed up convergence.
RMSProp	Good for **non-stationary**, noisy, or reinforcement learning environments (e.g., RL agents).
Adam	Great **default** choice: fast convergence, good with **sparse gradients** (e.g., NLP, embeddings).
AdamW	Use this instead of Adam when training **deep neural nets** with **weight decay** (e.g., Transformers, BERT, vision models).

G. Evaluation Metrics	start time:

Model:

An evaluation metric measures how well the trained model performs — typically on validation or test data.

It provides a human-interpretable performance score to assess the model's effectiveness for the specific task.

- Regression: MSE, MAE, R^2
- Classification: Accuracy, Precision, Recall, F1

Confusion Table:

		Prediction Results		
		Positive (PP)	Negative (PN)	
Actual Observations	Positive (P)	True Positive (TP)	False Negative (FN, **Type II Error**)	**Recall, TPR = TP/P**
	Negative (N)	False Positive (FP, **Type I Error**)	True Negative (TN)	**FPR = FP/N**
		Precision = TP/PP		**Accuracy = (TP+TN)/(P+N)**

Questions:

45. What is the difference between evaluation metrics and loss function?

- Loss function is used during the training of the neural network to determine the model's learning and minimize loss/error rate
- Evaluation metrics are used after training to assess how well the model performs on unseen/test data

46. When is MAE better than RMSE?

- RMSE enhances error exponentially. Therefore if there are more outliers then we consider MAE over RMSE
- MAE is better than RMSE when extreme values for outliers are likely present but all outliers are of equal importance

47. R^2 is the proportion of variance explained. What does a high R^2 indicate?

- High R^2 indicates that the model's predictions are very close to the actual values
    - Model explains a large proportion of the
- High R^2 means model fits but lower means model doesn't fit

48. Recall that the accuracy or classification error is not a good choice of loss function for classification tasks. Is it a good choice for evaluating the classification network?

- It is not a good choice for evaluating the network
- We use the confusion table to better evaluate the classification network

49. In which situation, simply using classification error to evaluate the model is not a good idea?

- If the data is biased. for example if the data has 99% dog and 1% cat and we give 100 images. if the network is dumb and says dog to everything, we get 99% accuracy in this case. Therefore if the data is imbalance we cannot use classification error as a mean of evaluating the classification network

50. Precision is TP / (TP + FP). What does precision measure?

- Ratio of true positive to number of total positives predictions
- If the precision is high, the model is more accurate. example: if a model says someone is sick, then there is a high chance the person is sick.
- False Positive(FP): Instances of a negative class incorrectly predicted as positive

51. Recall is TP / (TP + FN). What does recall measure?

- Recall measures the ability of the model finds all relevant instances of a positive class
- True Positive(TP): Instances of a positive class that are correctly identified
- False Negative(FN): Instances of a positive class that are incorrectly identified as as negative
- Quantifies the proportion of actual positive cases that the model correctly identifies
- If we want to find all positives, then high recall, may have lower precision

52. Which one should you choose if you focus on false negative, precision or recall?

- Recall

53. Which one should you choose if you focus on false positive, precision or recall?

- Precision

54. How do we balance precision and recall?

- Precision is how likely it is to predict positive
- Recall is the ability to find positive
- We balance by using F1 score

55. How do we balance type 1 and type 2 errors?

    Type 1 error = false positive

    - Telling someone they have cancer when they don't have cancer

    Type 2 error = false negative

    - Telling someone they don't have cancer when they do have cancer

- Adjust parameters/costs to account for miscalculations based on potential consequences
- Dependent on situation (ex: medical diagnostics type 2 error is more severe and wants to be noted more than type 1 errors)
- Adjust the decision threshold
    - Depends on the relative cost of each type of error in the specific application

56. F1 score is 2*Precision*Recall/(Precision+Recall). What does F1 score measure?

- Balance/harmonic mean
- Model's accuracy when balancing both precision and recall
- Tries to give equal importance to both, used when both false positives and false negatives carry equal importance

57. Which metric helps with imbalanced datasets, classification accuracy or F1 Score?

- For imbalanced datasets, F1 score is the best metric
    - Considers both precision and recall
    - Better understanding of a model's performance on minority classes
    - More reliable than accuracy alone for imbalanced datasets

Dummy Model always predicts positive

99% Negative

1% Positive

TP / (TP + FP) = 0.01/(0.01 + 0.99) = 0.01 = Precision

TP / (TP + FN) = 0.01/(0.01) = 1 = Recall

H. Regularization	start time:

Model: Regularization helps prevent overfitting. Common types include L2, dropout, and early stopping. **Regularization** in neural networks is a set of techniques used to **prevent overfitting** — that is, when a model learns the training data too well (including noise), but performs poorly on new, unseen data.

Questions:

58. **L2** means

$$\mathcal{L}_{\text{reg}} = \text{CrossEntropy} + \lambda \sum w^2$$

What does L2 regularization penalize? How does it help to prevent overfitting?

- L2 is the squared sum of the model's coefficients/weights
- Penalty term proportional to the square of the coefficients/weights meaning it penalizes large weights
- Prevents overfitting by encouraging the model to distribute influence more evenly across features rather than depend heavily on small set of features

59. **Dropout** randomly turns off neurons during training. How does this help the training?

- It means dropping some of the neurons to prevent overfitting
- Less reliant on a specific neuron, adds randomness and makes the network better at generalization on new data

60. **Early stopping** means stopping training early. When could we use early stopping?

- Validation is stable, no improvement on the testing
- Training keeps going down, you get overfitting, should early stop

61. **Data augmentation** means generating data from itself to enlarge the size of the training set. Why is this helpful?

- This is helpful because there are only limited number of data but we need more to complete the training. A model will be better one if there are more number of training data.

62. If the dataset is images, what kinds of data augmentation could we use to generate more images?

- changes in saturation, contrast, sharpness, orientation, etc

63. **Batch normalization** is a step to standardize the inputs to a layer so that they have mean as 0 and standard deviation as 1. Why this step could be helpful?

- To improve training speed and stability