Q1)

1. The **bias** adds the ability to intercept the Y-axis at different points while the **activation function** adds non-linearity to the model's decision boundaries
2. MLP, since it's able to learn non-linear functions from data while LMS is only suitable for linearly separable problems
3. Since after each iteration the model's generalization can be assessed on validation data and if the model performance on it starts dropping then the training should stop.
4. Because the backpropagation algorithm requires a differentiable activation function while the Step function is not differentiable

Q2)

1. The momentum parameters prevent the gradient descent from oscillating which provides faster convergence
2. The model has overfit on the training data and failed to generalize
3. Using a high learning rate will cause the weights to explode which will cause divergence of the model, this can be detected by watching the cost increase after each iteration and the weights magnitude being very large

Q3)

1. Yes, $\gamma \; and \; \beta$ which control the mean and variance of the normalized data, they allow the backpropagation to control the scale and shift of the output to best represent the activations
2. Batch GD converges steadily towards the local optima however if the data is large the updating steps can take long.
   while the SGD suffers from oscillations in the descent and is more prone to noise in data since weights are updated after each sample.
   A moderate approach is the Mini-Batch which updates weights after each batch of samples
3. Since the Tanh has steeper gradients thus it can take bigger steps towards the minima

Q4)

$Input: 32 \times 32 \times 3$
$Neurons: 5 \qquad Kernel: 3 \times 3 \qquad stride: 1$

1)

$$Output = \frac{Input - K + 2P}{S} + 1$$

Since padding is set to “same”, $Output = Input = 32$, replace in the equation to get the pad

$$32 = \frac{32 - 3 + 2P}{1} + 1$$

$$P = 1$$

2)

$Output = 32 \times 32 \times 5$
(32 because it’s mentioned in the Q that the output have the same volume as input, 5 because 1 output for each neuron)

3)

Weights per neuron: $kernel\ size \times input\ depth + bias = 3 \times 3 \times 3 + 1 = 28$
Total weights = $28 \times 5\ (filters) = 140$

4)

To produce the same output volume:
$neurons = 32 \times 32 \times 5 = 5120$

Weights: input image pixels count x output neurons count + biases
$Weights = (32 \times 32 \times 3) \times 5120 + 5120 = 15,733,760$

5)

The weights in the fully connected are larger

- Since in convolution layers the neuron will connect to 3x3x3 chunk and have 3x3x3 weights
- Convolutional layers allow the sharing of parameters between neurons

Q5)

1. ReLU
2. Fully connected layer
3. One neuron or Two Neurons(must use softmax)
4. Sigmoid or Softmax
5. LMS or Binary Cross Entropy