1. Explain the Activation Functions in your own language

a) sigmoid

b) tanh

c) ReLU

d) ELU

e) LeakyReLU

f) swish

Answer:----> a.Sigmoid---->The sigmoid function is also called a squashing function as its domain is the set of all real numbers, and its range is (0, 1). Hence, if the input to the function is either a very large negative number or a very large positive number, the output is always between 0 and 1. Same goes for any number between -∞ and +∞. The Sigmoid function performs the role of an activation function in machine learning which is used to add non-linearity in a machine learning model. Basically, the function determines which value to pass as output and what not to pass as output. symbol σ The symbol σ represents sigmoid function with the form σ(x) = 1.

b. tanh---->Tanh is the hyperbolic tangent function, which is the hyperbolic analogue of the Tan circular function used throughout trigonometry. Tanh[α] is defined as the ratio of the corresponding hyperbolic sine and hyperbolic cosine functions via . Tanh may also be defined as , where is the base of the natural logarithm Log.'tansh', or sometimes as 'than'. The function is defined by the formula. tanhx = sinh x cosh x .

c. ReLU---->A rectified linear unit (ReLU) is an activation function that introduces the property of non-linearity to a deep learning model and solves the vanishing gradients issue. "It interprets the positive part of its argument. It is one of the most popular activation functions in deep learning.ReLU stands for Rectified Linear Unit. ReLU activation function is one of the most used activation functions in the deep learning models. ReLU function is used in almost all convolutional neural ne tworks or deep learning models.

d.ELU---->The Exponential Linear Unit (ELU) is an activation function for neural networks. In contrast to ReLUs, ELUs have negative values which allows them to push mean unit activations closer to zero like batch normalization but with lower computational complexity. Mean shifts toward zero speed up learning by bringing the normal gradient closer to the unit natural gradient because of a reduced bias shift effect. While LReLUs and PReLUs have negative values, too, they do not ensure a noise-robust deactivation state. ELUs saturate to a negative value with smaller inputs and thereby decrease the forward propagated variation and information.

e. Leakly ReLU----->The rectified linear unit (ReLU) is one of the most common activation functions in machine learning models. As a component of an artificial neuron in artificial neural networks (ANN), the activation function is responsible for processing weighted inputs and helping to deliver an output.

2. What happens when you increase or decrease the optimizer learning rate?

Answer:---->A learning rate that is too large can cause the model to converge too quickly to a suboptimal solution, whereas a learning rate that is too small can cause the process to get stuck. Both low and high learning rates results in wasted time and resources. A lower learning rate means more training time. more time results in increased cloud GPU costs. a higher rate could result in a model that might not be able to predict anything accurately.If your learning rate is set too low, training will progress very slowly as you are making very tiny updates to the weights in your network. However, if your learning rate is set too high, it can cause undesirable divergent behavior in your loss function. Generally, a large learning rate allows the model to learn faster, at the cost of arriving on a sub-optimal final set of weights. A smaller learning rate may allow the model to learn a more optimal or even globally optimal set of weights but may take significantly longer to train.

3. What happens when you increase the number of internal hidden neurons?

Answer:---->We will see that increasing the number of hidden neurons increases the performance of a model using the MNIST dataset. The MNIST dataset is a common standard dataset used to evaluate machine learning models performance, which is just a task of recognizing digits from 0 to 9.On the other hand when the number of hidden layers cross the optimal number of hidden layers (three layers), time complexity increases in orders of magnitude as compared to the accuracy gain.Adding a hidden layer between the input and output layers turns the Perceptron into a universal approximator, which essentially means that it is capable of capturing and reproducing extremely complex input–output relationships.Neurons in the output layers have direct relation with the complexity of the problem or the work given to the network [1]. Hidden layer is basically an intermediate layer between the input and the output layer of the network. Number of hidden layers within the network changes from problem to problem.

4. What happens when you increase the size of batch computation?

Answer:---->larger batch sizes make larger gradient steps than smaller batch sizes for the same number of samples seen. for the same average Euclidean norm distance from the initial weights of the model, larger batch sizes have larger variance in the distance.Batch size, risk and cycle time are all directly proportional. As any grows larger, so do the others. The amount of risk we create is equal to our batch size and cycle time, i.e. our total work in process.The batch size affects some indicators such as overall training time, training time per epoch, quality of the model, and similar. Usually, we chose the batch size as a power of two, in the range between 16 and 512. But generally, the size of 32 is a rule of thumb and a good initial choice.Large batch sizes may cause bad generalization (or even get stuck in a local minimum). Generalization means that the neural network will perform quite well on samples outside of the training set.

5. Why we adopt regularization to avoid overfitting?

Answer:----> Regularization is the process of regularizing the parameters that constrain, regularizes, or shrinks the coefficient estimates towards zero. In other words, this technique discourages learning a more complex or flexible model, avoiding the risk of Overfitting. We adopt regularization to avoid overfitting because overfitting occurs when a model is too complex and captures not only the underlying patterns in the data, but also the noise and random fluctuations. This leads to a model that performs very well on the training data but generalizes poorly to new data.

Regularization helps prevent overfitting by adding a penalty term to the loss function that the optimizer is trying to minimize. This penalty term discourages the model from assigning too much importance to any single feature or combination of features, which can lead to overfitting. Regularization effectively reduces the complexity of the model and makes it more generalizable.

Common regularization techniques include L1 and L2 regularization (also known as weight decay), dropout, and early stopping. L1 and L2 regularization add a penalty term to the loss function based on the magnitude of the model's weights, while dropout randomly drops out some of the neurons during training to prevent them from relying too heavily on any single input feature. Early stopping stops training when the performance on a validation set starts to degrade, preventing the model from overfitting to the training data.

6. What are loss and cost functions in deep learning?

Answer:---->The loss function is to capture the difference between the actual and predicted values for a single record whereas cost functions aggregate the difference for the entire training dataset. The Most commonly used loss functions are Mean-squared error and Hinge loss.A loss function is a function that compares the target and predicted output values; measures how well the neural network models the training data. When training, we aim to minimize this loss between the predicted and target outputs.While the loss function calculates the error for a single data point (sample), the cost function calculates the loss for the entire dataset. The cost of a neural network is nothing but the sum of losses on individual training samples.The cost function is ½ of the difference of the predicted value and the accurate value squared. The cost function's purpose is to calculate the error we get from our prediction. Our goal is to minimize the cost function. The smaller the output of the cost function, the closer the predicted value is to the actual value.

7. What do ou mean by underfitting in neural networks?

Answer:----->Underfitting is a scenario in data science where a data model is unable to capture the relationship between the input and output variables accurately, generating a high error rate on both the training set and unseen data.It occurs when a model is too simple, which can be a result of a model needing more training time, more input features, or less regularization. Like overfitting, when a model is underfitted, it cannot establish the dominant trend within the data, resulting in training errors and poor performance of the model.Underfitting happens when the network can neither model the training or test data which results in overall bad performance. Underfitting happens in the first diagram i.e, on the left shown in the above picture. In that diagram, the model doesn't cover all the data points & has a high error on both training & test data.Underfitting, the counterpart of overfitting, happens when a machine learning model is not complex enough to accurately capture relationships between a dataset's features and a target variable.

8. Why we use Dropout in Neural Networks?

Answer:---->Dropout layers have been the go-to method to reduce the overfitting of neural networks. It is the underworld king of regularisation in the modern era of deep learning. In this era of deep learning, almost every data scientist must have used the dropout layer at some moment in their career of building neural networks.Dropout refers to data, or noise, that's intentionally dropped from a neural network to improve processing and time to results. A neural network is software attempting to emulate the actions of the human brain.The main advantage of dropout is of course that it prevents overfitting. It is great for bigger networks where neurons are all connected. The regularization process of dropout makes the neurons independent of each other so that all of them can perform better with less noise.