1) Explain the formulation of linear regression as maximization of a likelihood function. Explain why classification algorithms such as logistic regression learn a probability distribution over classes conditioned on the input instead of directly producing an output y (for eg:- y would be a binary output taking values 0 or 1 for the case of binary classification) with an appropriate loss function defined on the output y instead of on the output probabilities. Give at least two points to support your answer.

Let $y = wx + b + \varepsilon$

be the linear regression model where ε is the random noise of data. We assume the noise is distributed from a Gaussian distribution with a mean of 0 and some unknown variance σ^2.

$$\varepsilon \sim N(0, \sigma^2)$$

Now, we can write the conditional distribution of y given x in terms of Gaussian.

$$p(y|x; w, b, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-(wx+b))^2}{2\sigma^2}}$$

Each point is independent and identically distributed (iid), so we can write the likelihood function with respect to all of our observed points as the product of each individual probability density. Since σ^2 is the same for each data point, we can write it as:

$$L_x(w, b, \sigma^2) = \prod_{(x,y)\in X} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(y-(wx+b))^2}{2\sigma^2}}$$

Now take log and simplify it and then multiply it with -1. We need to minimize the negative log likelihood and here we can observe the second term on the right hand side. It is the MSE term.

$$-l_x(w, b, \sigma^2) = \log\left(\sqrt{2\pi\sigma^2}\right) + \frac{1}{2\sigma^2} \Sigma(y-\hat{y})^2$$

The reason why classification algorithms learn probabilities instead of binary outputs is because the notion on which we are training our network on is probability of Y happening in given conditions X. Which is P(Y|X) a conditional probability. And hence it learns a probability instead of binary labels.

2) Explain the algorithm for Stochastic Gradient Descent with Momentum. Provide some intuitive justifications/ analogies with momentum in physics to motivate the algorithm.

   The need for Stochastic Gradient Descent with Momentum comes from the fact that SGD faces issues in moving quickly in areas where slope is steep in one direction. What SGD with Momentum does is average last few steps and move in the relevant direction.
   The formulas are –
   $VdW = \beta VdW + (1-\beta)dW$
   $Vdb = \beta Vdb + (1-\beta)db$
   $W = W - \alpha VdW$, $b = b - \alpha Vdb$

   A physics analogy is a ball rolling down a hill. The gradient term adds acceleration to it and the momentum term(VdW and Vdb) adds velocity and the term β is like friction.

3) Answer the following part briefly:

   1) Explain why mini-batches are used. Also, comment on the efficiency and robustness of using mini-batch as compared to not using mini-batch (or mini-batch with size 1)

      As the size of data increases the time taken by optimization algorithm also increases and if the size is too big then training time will make training the network too difficult. To tackle this problem mini-batches are used and the size of batches is neither too big(about the size of data) nor too small(each example is a batch). The batch size is taken to be something in power of 2(eg – 64, 128, 256…). The issue with using batch size of one example is that we lose the benefit of vectorisation.

   2) Weight Initialization is one of the crucial element while training. So, explain why initializing weights symmetrically can be problematic.

      If weights are initialised symmetrically then during training all the neurons will be equivalent and they will learn the same parameters. In the end the whole neural network will become equivalent to one linear step.

   3) Why regularization reduces overfitting and what are other ways to tackle the problem of overfitting.

Regularization aims at keeping weights small. The more you train your network the more its weights become complex. Larger weights are associated with overfitting and regularization aims at precisely that issue.
Some methods to tackle overfitting are –

1) Dropout
2) Using cross-validation sets or development sets
3) Early stop

4) Explain Batch Normalization in NN and why it stabilizes the training of NN.

   We know that gradient descent works faster in linear regression when features are normalised. The idea behind Batch Normalization is that in neural networks we go back and update values during backpropagation and in a way we are chasing ever moving targets. Batch normalization deals with this and helps in training faster by normalising values after applying activation function in every layer.

4) Consider a Conv layer with kernel size (or window size) as 3x3 and input channel as 3, output channel as 8, padding P equals to 1 and stride equals to 2. Calculate the number of parameter of this conv layer (don't consider bias terms) and given an input of image of N1xN2x3 (where N1 is the height, N2 is the width of image and 3 is the number of channels in images) calculate the dimension of output when input is passed to this conv layer.

   No. of parameters = 3*3*3*8 = 216.
   Dimensions of output = ((N1-3)/2 + 1), (N2-3)/2+1, 8)