MACHINE LEARNING

In Q1 to Q8, only one option is correct, Choose the correct option:

Q1-The computational complexity of linear regression is:

Answer- B) $O(n)$

Q2- Which of the following can be used to fit non-linear data?

Answer - C) Polynomial Regression

Q3- Which of the following can be used to optimize the cost function of Linear Regression?

Answer- B) Gradient Descent

Q4 - Which of the following method does not have closed form solution for its coefficients?

C) Lasso

Q5- Which gradient descent algorithm always gives optimal solution?

D) All of the above

Q6- Generalization error measures how well a model performs on training data.

A) True

Q7- The cost function of linear regression can be given as $J(w0, w1) = 1\ 2m \sum (w0 + w1x\ (i) - y\ (i)\ )\ m\ 2$ i=1 . The half term at start is due to:

A) scaling cost function by half makes gradient descent converge faster.

Q8- Which of the following will have symmetric relation between dependent variable and independent variable?

A) Regression

In Q9 to Q11, more than one options are correct, Choose all the correct options:

Q9- Which of the following is true about Normal Equation used to compute the coefficient of the Linear Regression?

A) We don't have to choose the learning rate.

B) It becomes slow when number of features are very large.

D) It does not make use of dependent variable.

Q10 - Which of the following statement/s are true if we generated data with the help of polynomial features with 5 degrees of freedom which perfectly fits the data?

A) Linear Regression will have high bias and low variance.

C) Polynomial with degree 5 will have low bias and high variance.

Q11- Which of the following sentence is false regarding regression?

C) It discovers causal relationship.

D) No inference can be made from regression line.

Q12 and Q13 are subjective answer type questions, Answer them briefly.

Q12 - Which Linear Regression training algorithm can we use if we have a training set with millions of features?

Answer-12) We could use batch gradient descent, stochastic gradient descent, or mini-batch gradient descent. SGD and MBGD would work the best because neither of them need to load the entire dataset into memory in order to take 1 step of gradient descent. Batch would be ok with the caveat that you have enough memory to load all the data.

The normal equations method would not be a good choice because it is computationally inefficient. The main cause of the computational complexity comes from inverse operation on an (n x n) matrix.

Q13 - Which algorithms will not suffer or might suffer, if the features in training set have very different scales?

Answer 13) - The normal equations method does not require normalizing the features, so it remains unaffected by features in the training set having very different scales.