LINK TO COLAB SUBMISSION:
https://colab.research.google.com/drive/1uhumttN1LRDFStj6fML58tWLXU8T0JwE?usp=sharing

Question-1:

a) The performance did not improve drastically when regularization was used along with normal equations of linear regression. There was no drastic change in error because regularization works better with gradient descent as seen further in this assignment.
I also tried with various values of lambda as can be seen in the implementation of this assignment. Among values of [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 100, 1000, 10000], values around lambda = 10 were giving better performance.

b) The training was faster with gradient descent than normal equations as the time complexity of implemented code in (a) was in the order of O(n^3). So, it was training slower.
Without scaling, a problem I encountered was that due to the unscaled feature values being large, matrix multiplication was throwing overflow warning.
Thus, scaling seems to be better also in terms of not causing overflows.
The learning curves also converged faster with feature scaling than without feature scaling. So, for the same number of epochs (in my case, I set epochs as 200), the prediction error was lesser with scaling.

I also experimented with different values of batch sizes like 2, 4, 16 and 32 which can be observed in the implementation. Batch_size = 32 was giving better results compared to the rest of the values mentioned above.
With respect to stochastic, batch and mini batch gradient, stochastic gradient gave lesser prediction and mini batch gradient gave similar results and batch gave slightly higher error. This could have also been possible due to epoch values set which in my implementation, I set it 200 as it was giving the least error.

c) The model performed better with regularization than without regularization with respect to batch gradient descent, mini batch gradient descent and stochastic gradient descent.

Regularization with lambda = 20 was giving the best results. So, this value has been used for all three types of descents.

I also experimented with different values of batch sizes like 2, 4, 16 and 32 which can be observed in the implementation. Batch_size = 32 was giving better results compared to the rest of the value mentioned above.

With respect to stochastic, batch and mini batch gradient, stochastic gradient gave lesser prediction and mini batch gradient gave similar results and batch gave slightly higher error. This could have also been possible due to epoch values set which in my implementation, I set it 200 as it was giving the least error.

Question-2:

LWR was performing similarly to the linear regression model implemented in a), b) and c).
This could be due to the fact that the dataset is not large, henceforth the number of local points are lesser.

I also experimented with various values of tau and exhibited the following results:
Higher values of tau was causing an underfitting problem.
Lower values of tau was causing an overfitting problem.