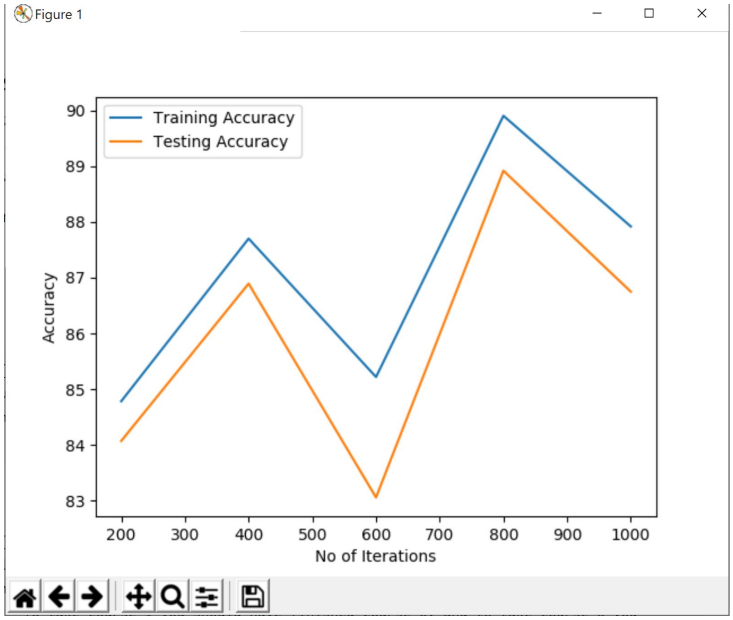
Results for Question 3:

Training and Testing accuracies for different learning rates:

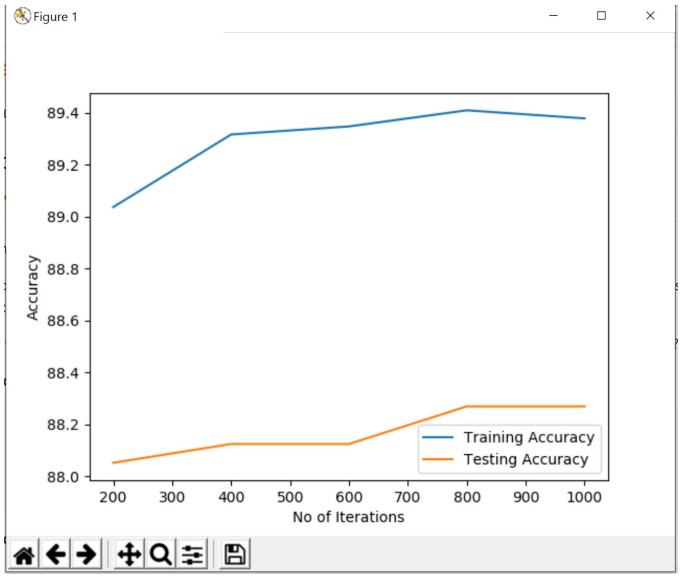
```
Iteration: 1000, mean abs gradient = 54.160718965339626
train accuracy = 87.9192546583851, test_accuracy = 86.74873280231716 for Learning Rate = 1.0
train accuracy = 89.37888198757764, test_accuracy = 88.26937002172339 for Learning Rate = 0.01
train accuracy = 92.14285714285714, test_accuracy = 92.75887038377986 for Learning Rate = 0.0001
train accuracy = 90.3416149068323, test_accuracy = 90.87617668356263 for Learning Rate = 1e-06
Iteration: 50, mean abs gradient = 2.0470165685036563
```

Plots for different Learning Rates:

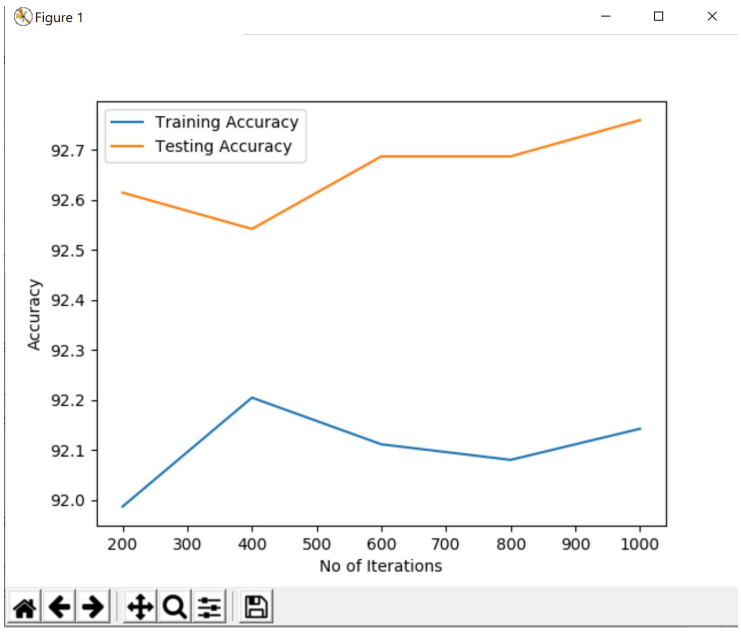
Learning Rate: 1e-0



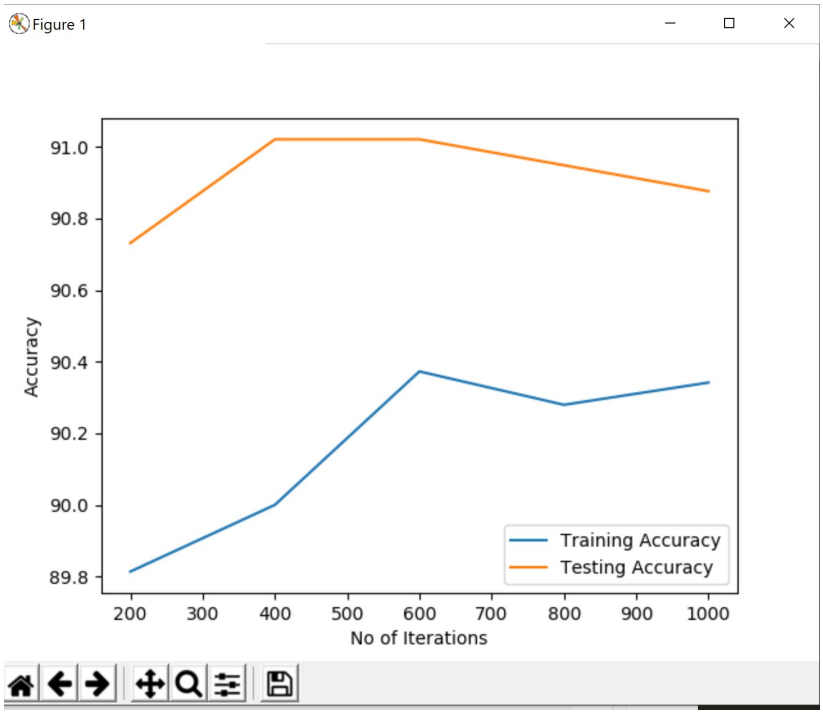
Learning Rate: 1e-2



Learning Rate:1e-4



Learning Rate: 1e-6



Explanation for the effect of choosing different learning rate:

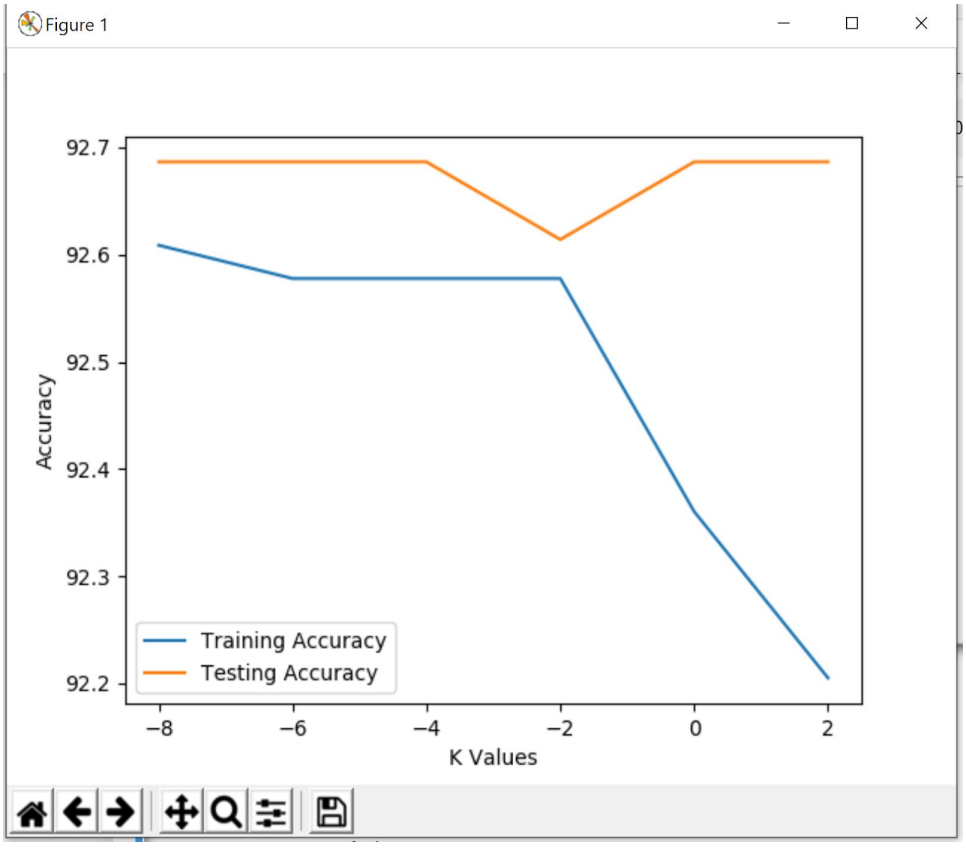
For choosing learning rate as 1e-0 as the learning rate is higher than compared to other's. The jump from one position to a different position in gradient descent is higher in between It may skip the optimal values which were actually to be checked. So, the training accuracy and test accuracy got decreased when compared to other's. where as, when we go to decreasing the learning rate at 1e-2 and 1e-3 gets optimal and the training, testing accuracy is increased. When the learning rate decreased that is at 1e-6 it gets overfitted and the training, testing accuracies got dispersed.In 1e-0 and 1e-2 the training accuracy is better than testing accuracy whereas in 1e-4, 1e-6 the testing accuracy is greater than training accuracy. In 1e-2 as the iteration increases the training and testing accuracy increased because it gets time to train.

Training and testing accuracies for different lambda values:

Training and testing accuracies for different regularized coefficient values:

```
train accuracy = 92.6086956521739, test_accuracy = 92.68645908761766 for Lambda value = 0.00390625
train accuracy = 92.5776397515528, test_accuracy = 92.68645908761766 for Lambda value = 0.015625
train accuracy = 92.5776397515528, test_accuracy = 92.68645908761766 for Lambda value = 0.0625
train accuracy = 92.5776397515528, test_accuracy = 92.61404779145546 for Lambda value = 0.25
train accuracy = 92.36024844720497, test_accuracy = 92.68645908761766 for Lambda value = 1
train accuracy = 92.20496894409938, test_accuracy = 92.68645908761766 for Lambda value = 4
```

Training and Testing Accuracy Curves for different regularized coefficients:



Performance of regularized logistic regression model using different regularization coefficients:

When the regularized coefficient value is very low. The accuracy is very high and it doesn't change as the K values increased it is almost the same till K value reached -2 but then after as the regularized coefficient increases by large value, the accuracies got decreased.Regularized coefficient is basically done to avoid overfitting and to balance the bias and variance. As we increase the regularization parameter, optimization function will have to choose a smaller

theta in order to minimize the total cost in the process of increasing regularization parameter it is losing some constraint satisfaction so the training accuracy and testing accuracy both were decreasing.

Comparing training/testing accuracy with non-regularized logistic regression model:

When we observe carefully non regularized and regularized parameter the trend in non regularized increases from low value to high value and decreases again due to overfitting or losing trend whereas in regularized during the initial stages for small values of regularized value the accuracies are very similar which says the maximum accuracy that we can obtain Then after due to loss of balance between bias and variance the trend loses and accuracies got decreased. Basically what regularization has done when compared to normal model is it tries to fit the pattern and ignore the noise.