- **Logistic Regression:**
    - For three of the four variations tested, performance of the Logistic Regression model peaks at C = 0.1(equivalent as lambda = 10)
    - For the model using Ridge(L2) regularization and polynomially transformed features(degree = 2), performance peaks at C = 0.01(equivalent to lambda = 100)
    - We can see that after a certain point, increasing the value of C(adjusting the penalty/regularization) doesn't improve performance anymore
    - According to the definition of C, we can conclude that a hard classifier doesn't fit this classification problem since large C values yield worse results
- **Support Vector Machines**
    - Unfortunately, three models using three different kernels obtain the best performance at three different C values
    - No obvious pattern can be observed on how to choose appropriate C value
    - But luckily, the model with RBF kernel has a monotonously

increasing performance while C increases(lambda decreases)

- We can see that SVM with Linear Kernel has the highest average accuracy and the highest maximum accuracy among all three kernels. Thus, the original data points(without any feature transformation) are already linearly separable. This also explains why SVM with RBF Kernel and Polynomial Kernel both yield much lower average accuracies when using the same C values.
- We can see that all three SVM models obtain peak performance at a relatively small C value(smaller than or equal to 1), which means a soft-margin classifier works better for this problem rather than a hard-margin one.

- **Neural Network:**
  - Setup:
    - Ridge Regularization is used along with 1 hidden layer. The goal is to improve test accuracy by varying number of neurons on the single hidden layer and the regularization parameter used. A threshold of 0.5 is also used.

- After applying the ridge regularization, the test accuracy did not improve significantly from the no regularization version.
- Large number of neurons used in a relatively small NN (with about 300 training examples and only 13 features) doesn't seem to help to improve the accuracy. In fact, from the results of test accuracy obtained, using small number of neurons, eg. 3 neurons on the hidden layer, seems to do slightly better than using larger number of neurons.
- Though not included as part of our code, we tried to use other activation functions, eg. ReLU and Leaky ReLU, it seems that they help to decrease the time needed to train the NN, but the accuracy decreases significantly to around 50%.
- We concluded that the best test accuracy can be achieved with 3 neurons used on a single hidden layer, lambda = 0.00140 or 0.01, number of iterations = 3000, learning rate = 0.5. (Note, this is just one of the combinations among others that can achieve same test accuracy. Given that fact, a NN that consumes less computational power is preferred.)