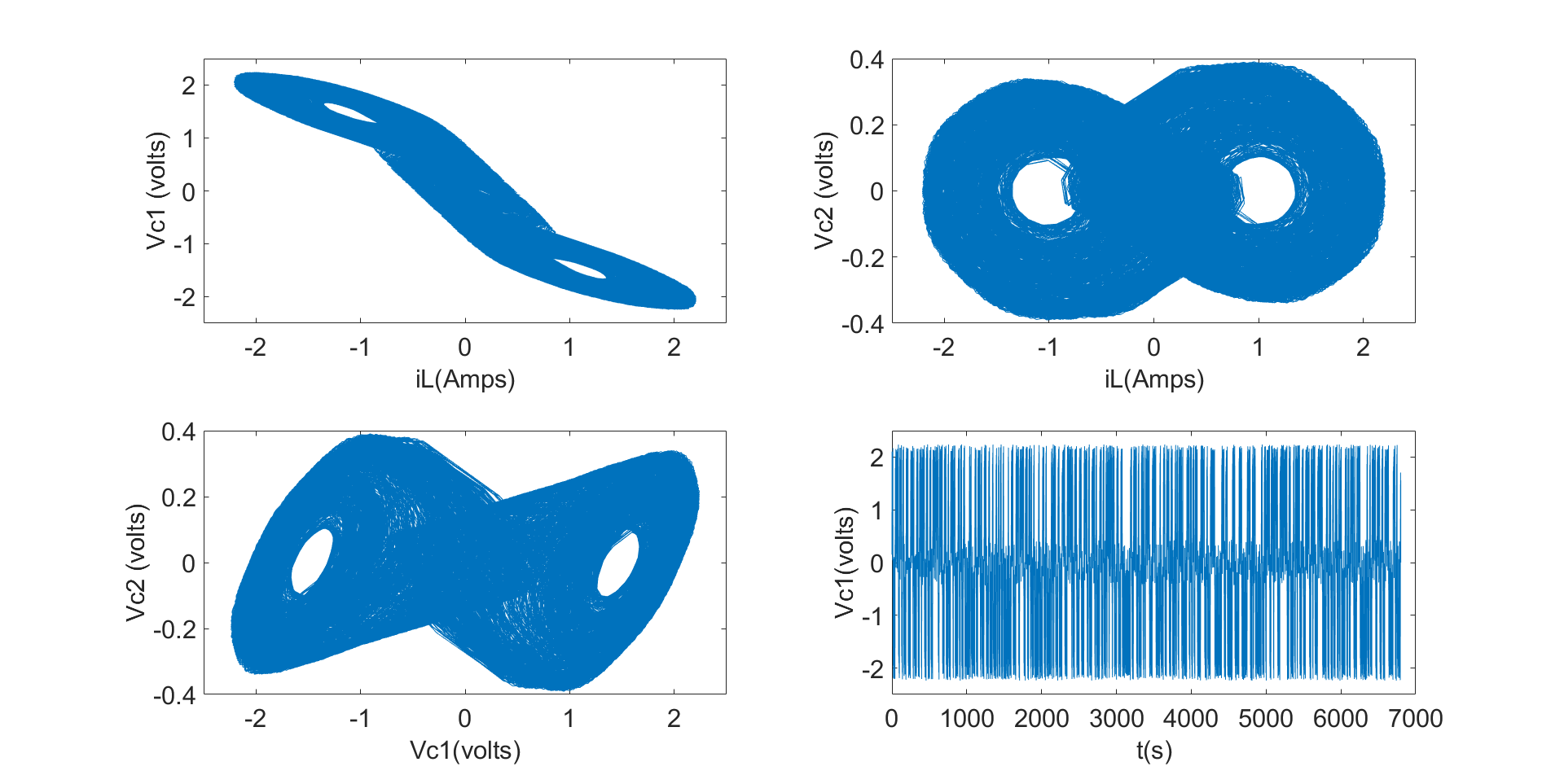
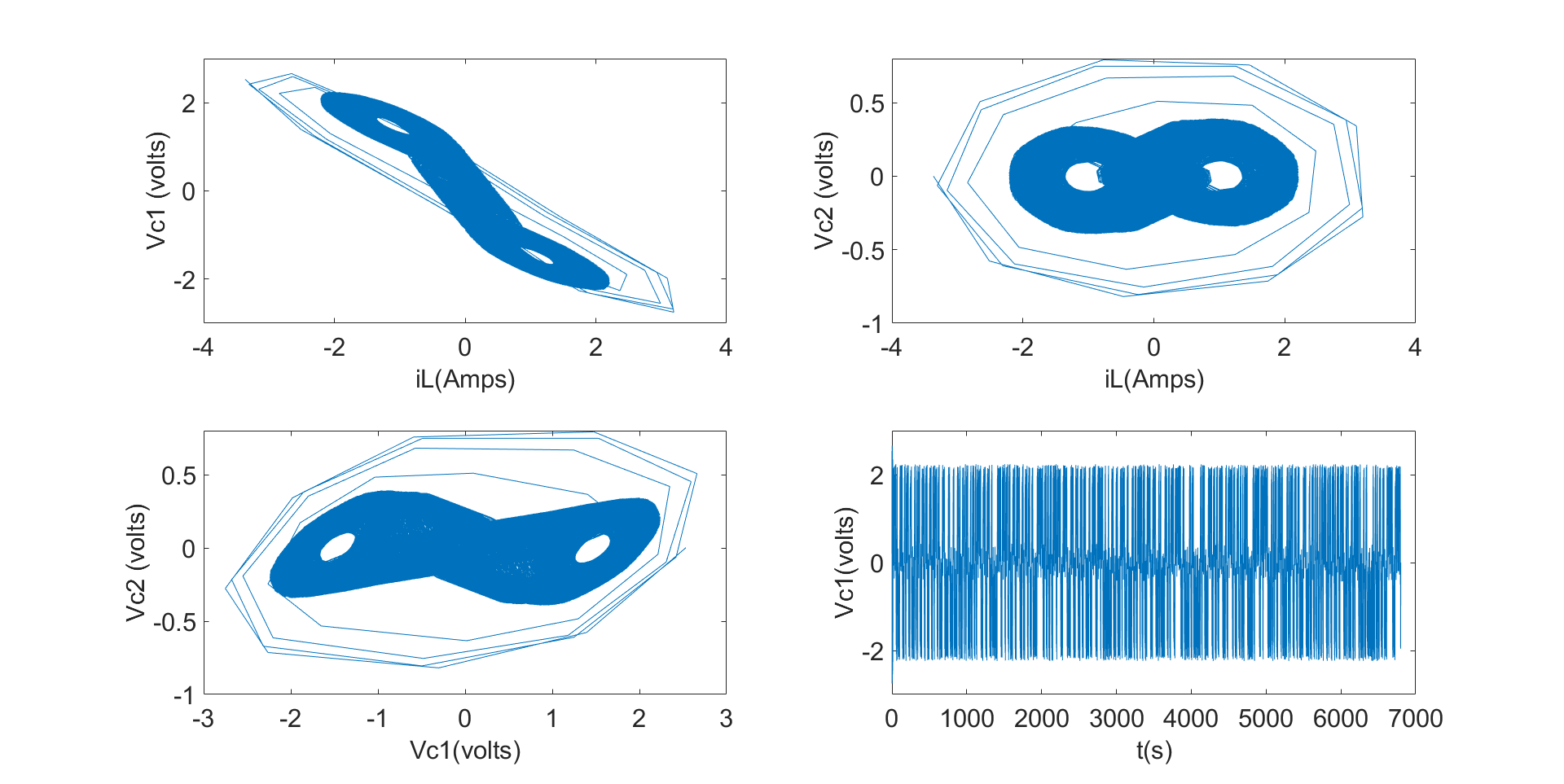
1. Question 1.1
   1. Current-voltage plots, showing scrolls.

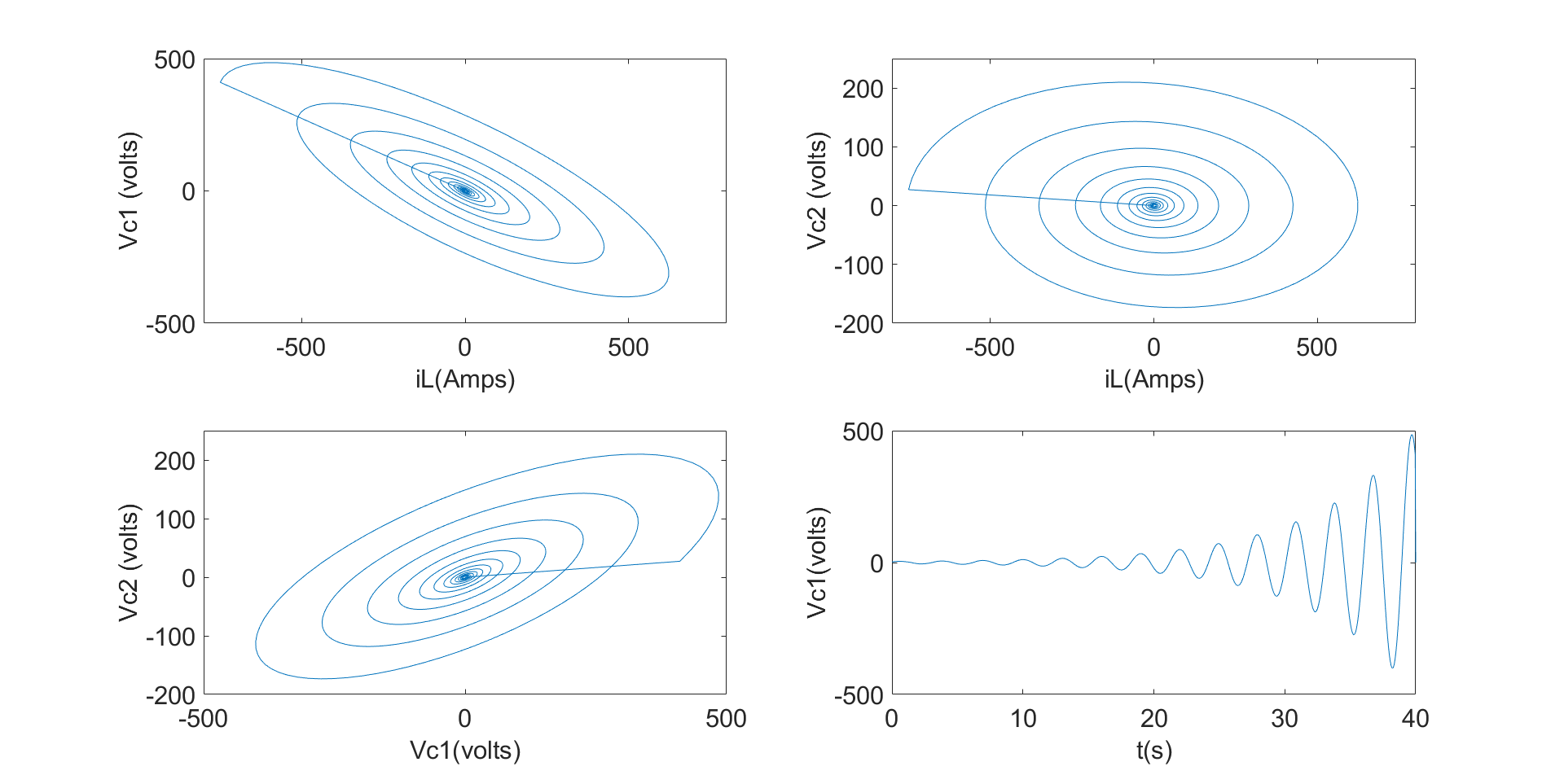
In the following plots, the fourth order Runge Kutta method is used to solve the

Initial conditions



Initial conditions



* 1. Are there other conditions for which the solution diverges to infinity? If yes, identify them, simulate and show the results. The following initial conditions result in divergence initialCond = [2.532735,1.29,0.2227]; with 1000 iterations to show the divergence clearly
  2. How can we modify to ensure the solution does not diverge with any initial condition? Run the solver for different values of and see how the solution changes. For my case, I found that with a the solution did not diverge, irrespective of the initial conditions.

1. The finite difference approximated solution is given by

With and plot of for and t = 0, t = 0.02 and t = 0.04 are as follows:



With and plot of for and t = 0, t = 0.025 and t = 0.05 are as follows:



Assume that the solution is a pure exponential: substitute into the parabolic PDE above to show that where

Finite Difference gives:

Let

If

From the half-angle formula

Which is equivalent to:

The biggest that U can get is therefore determined by

Whose maximum is

*Prove that the stability criterion (von Neumann) requires that*

This scheme is stable if

* 1. K Nearest Neighbors Algorithm

The ideal value of k = k\* is 5 whose classification rate is 0.86. The classification rate for k = k\*+2 = 7 is 0.86 and that for k = k\*-2 = 3 is 0.86.

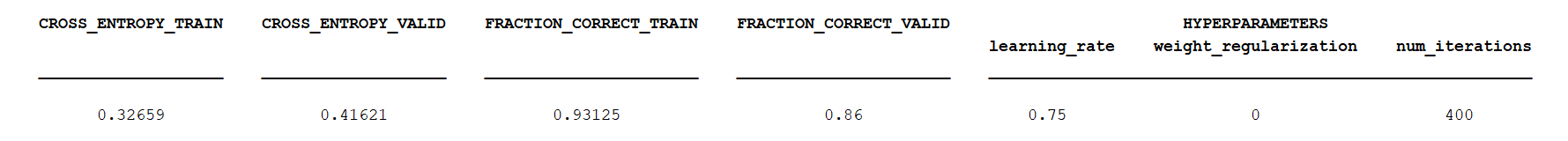




The classification rate for the test data also gives the same value of k = k\* = 5 as the optimal. However, on the test data, the classification rate is higher. Test classification rate at k = k\* = 5 is 0.94 versus 0.86 for the validation set. Better test performance is encouraging because it shows that the model does not overfit the training data. We do not expect test performance and validation performance to always be equal since the model trained

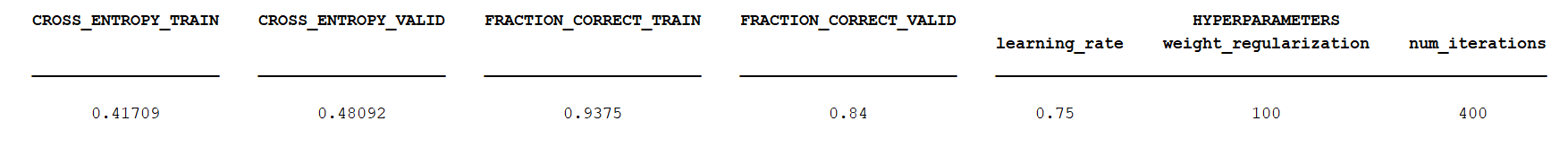
* 1. Logistic Regression

Using big training set (80 examples)





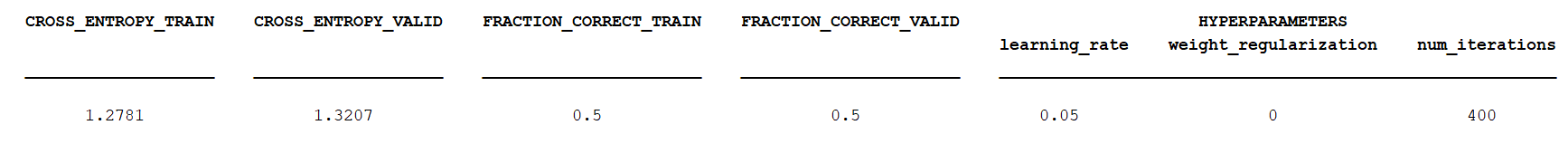
**General discussion**: we expect the training accuracy (93.125%) to be higher than validation (86%) since in training we are able to adjust the weights while in the validation stage we do not adjust the weights. In the case shown there is no regularization so we expect a very good fit with the data. However, the fit with the validation set is compromised.





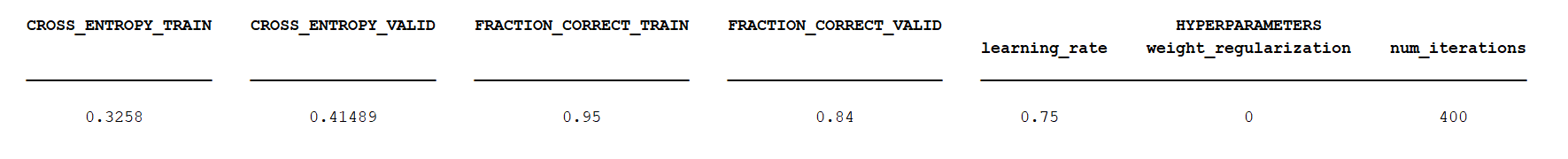
Adding in regularization increases the error for the training set because we do not fit the training set as precisely as before. However, we expected the regularization to lead to a gain in validation accuracy, which we do not see. So we play around with hyperparameters some more.

We remove regularization and instead lower the learning rate.



Reducing the learning rate has led to an increase in error both from the training and the validation set. I expected a lower learning rate to simply increase computational time rather than increase error like so.

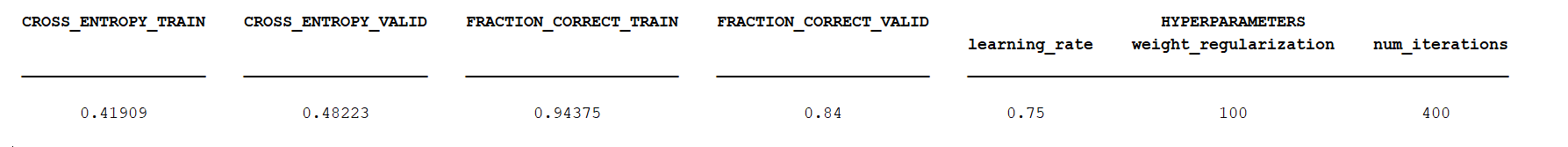
Next, I used initial weights that are 10 times smaller than before as well as using the earlier learning rate of 0.75





As shown the accuracy increases: the training set (50% to 95%) and validation set (50 to 84%). So reducing weights (at least when holding other parameters constant) is good for the model.

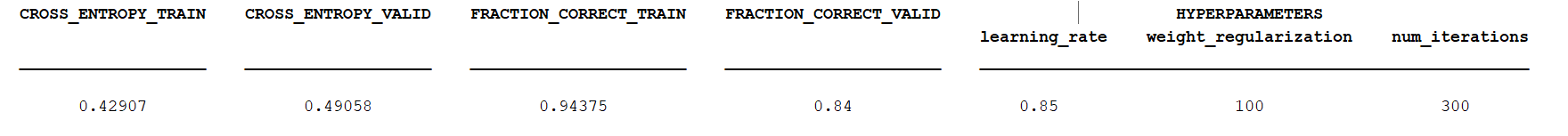
I then added regularization to reduce bias towards training set and better fit the validation data.



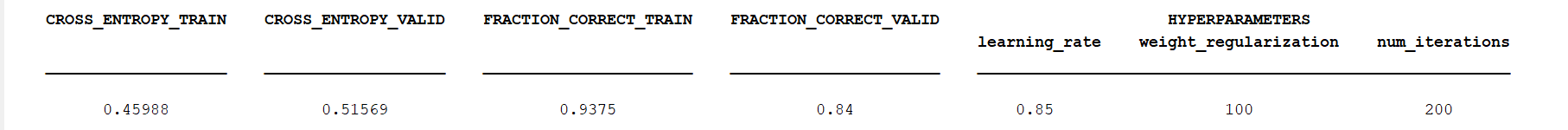


As expected, regularization reduces training set accuracy. But here it did not significantly boost accuracy of validation set.

Further changes showed that reducing iteration to 300 then 200 does not appreciably reduce accuracy of the learner. Few iterations are computationally desirable; changing from 300 to 200 iterations reduced accuracy of training set by less than 1%. We could thus settle for 200 iterations.



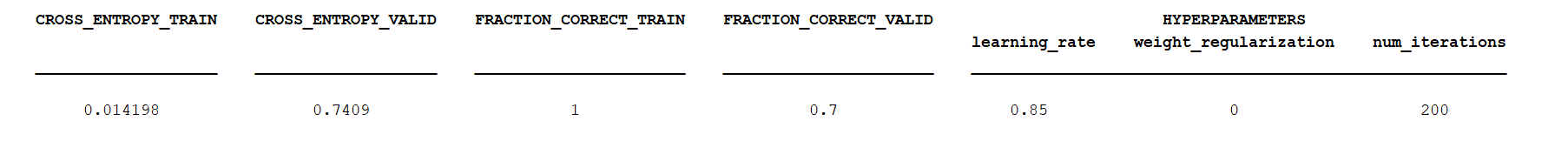






Next, a comparison of performance between the model based on the smaller training set and the one based on the larger set when tested on the training set provided. We will without further optimization of the hyperparameters (because at this point, we would already have developed the two models. The training has been done):

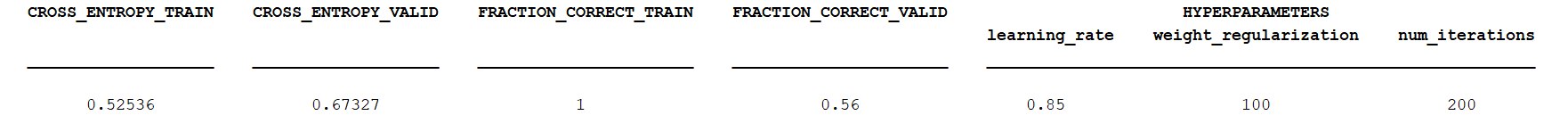
Results:

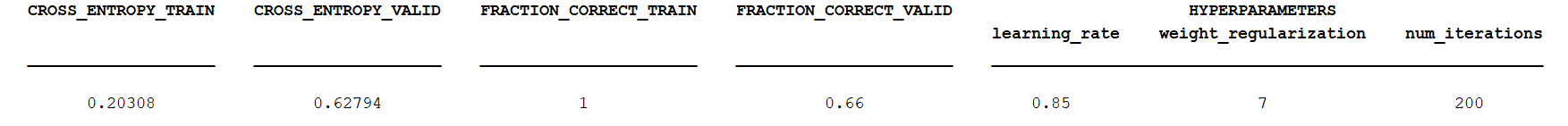


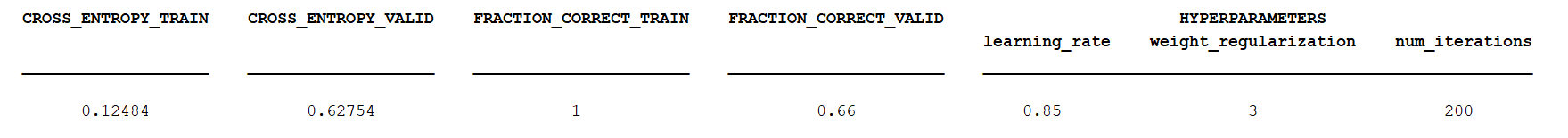


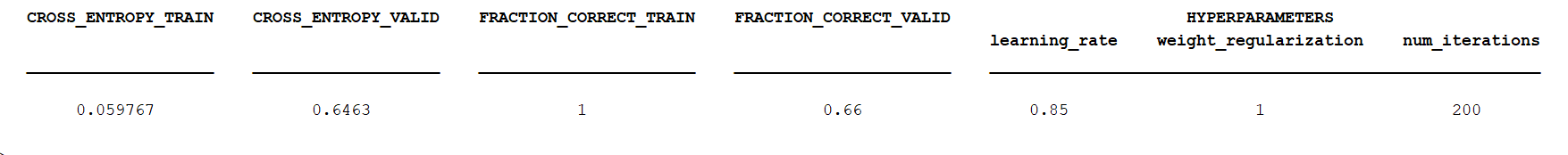
As expected, the smaller training set gives a higher validation error (0.7409) as compared to the larger training set (0.51569). Equivalently, validation set accuracy of model trained on smaller set (5 examples) in 14% lower than that trained on larger sample (80 examples).

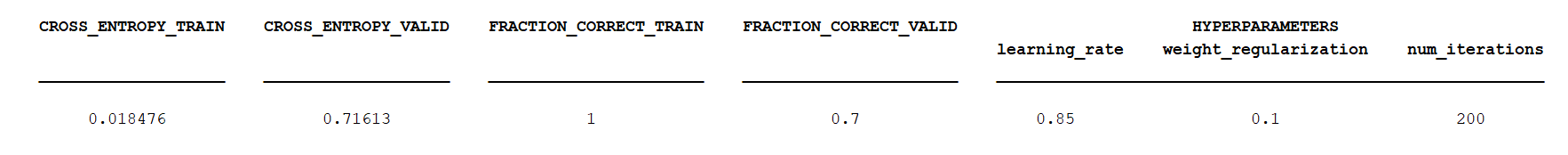
With regularization





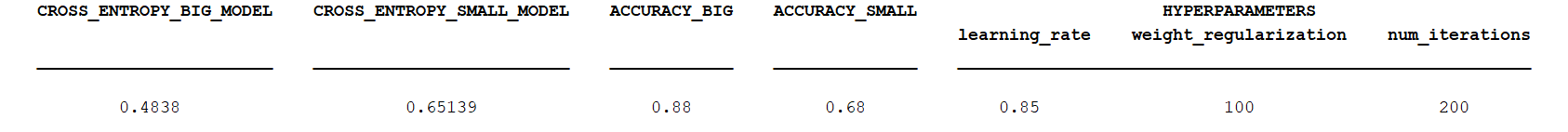






In general I note that regularization does not always improve the model. It in fact can degrades it. So it would be wise to implement the model without regularization in some instances. On the other hand, I note that introducing the regularization term does reduce validation error (for example, changing regularization from 7 to 3 reduces error). But the validation accuracy can also suffer in introducing the regularization term.

**Performance of models on test data:**





As expected, the big model has greater accuracy than the small model. This is because bigger model had more training samples.

1. Regression on Abalone Data

Ridge Regression: Choosing the best ridge parameter



We seek a smaller standard coefficient for all the features, so in the case shown above I chose the value of k = 1;

Lasso Model



The ideal coefficient for the lasso model is around as shown above.

Comparison between Ordinary Least Squares, Lasso and Ridge Regression



The plot above shows the variances of each model. For example, with a target value of about 10, we see that Ordinary Least Squares Method makes some very inaccurate predictions, as large as 16.

e. Comparison between linear model and ridge regression model



The graph above shows that for a given target value the Ridge model gives a more precise prediction; that is, its prediction has less variance. The Ridge model used has which was found to be produce the least mean of sum of squares of error. However, the overall error associated with the linear model is 4.7516 versus 5.5182 for the Ridge model. This shows that overall one could simply use the linear model and still get good enough results.

Principal Component Analysis is useful for transforming a correlated set of features into a set in which the features are no longer correlated. This results in a more accurate model because features that are dependent tend to create singularities. LASSO (Least absolute shrinkage and selection operator) is useful for selecting appropriate features, as well as for regularization to avoid overfitting the training data set. Reducing overfitting is desirable because we want our model to not just fit training data, but to be useful for new samples.

# Appendix