2.2

1. 37155 low income and 11687 high income earners.
2. No the dataset isn't balanced. There are far more low income earners than high income earners. The model may become biased towards low income classification during training purely because the examples are more likely than not low income earners.

2.3

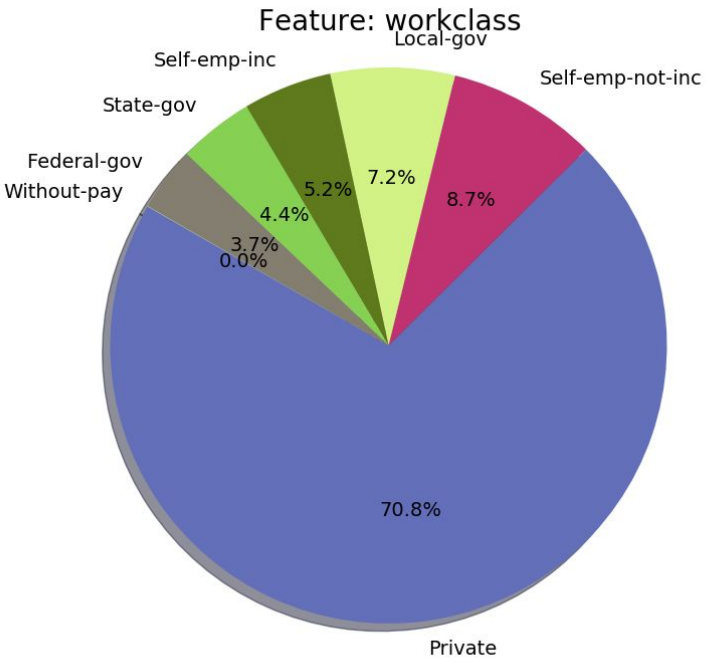
1. 3620 values were thrown out and 45222 are left. This is only a small fraction of the original dataset so it's okay to be throw this many sets out. There may however be a correlation between having missing information in your data and your income that we may lose out on.

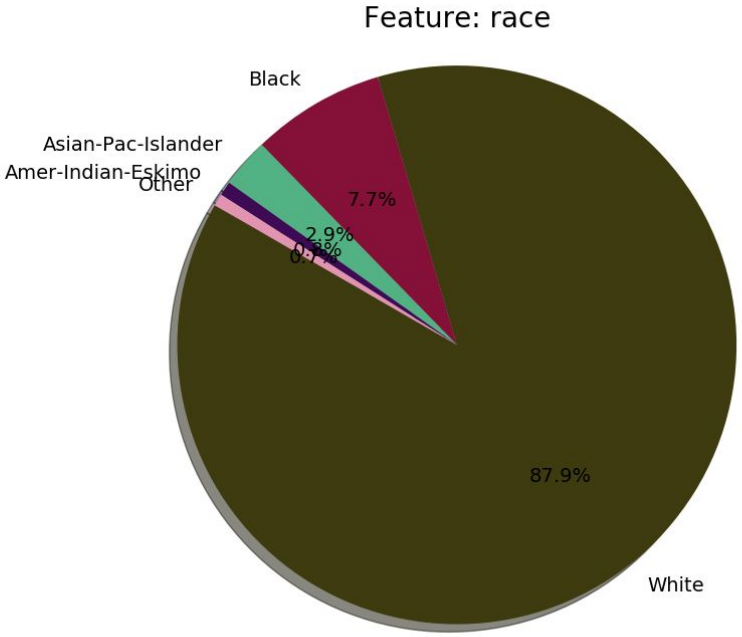
2.5

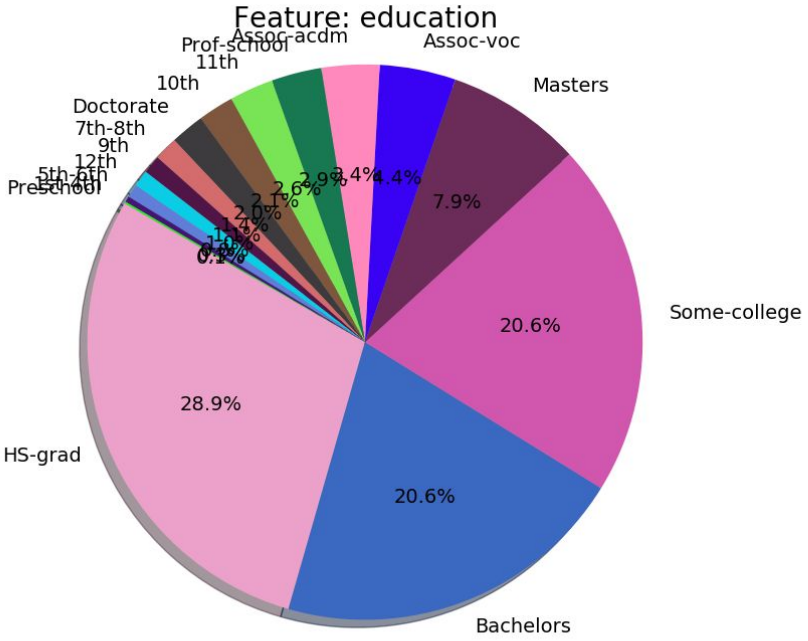
Statistics

1. The minimum age in the dataset is 17 and the minimum number of hours worked per week is 1.

Categorical Features-Pie Charts



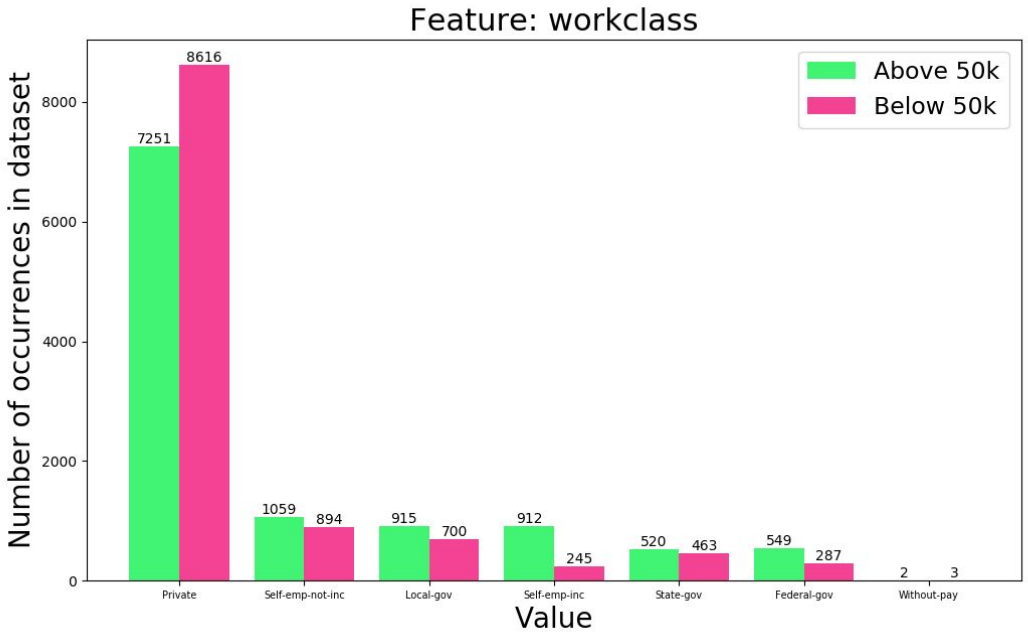


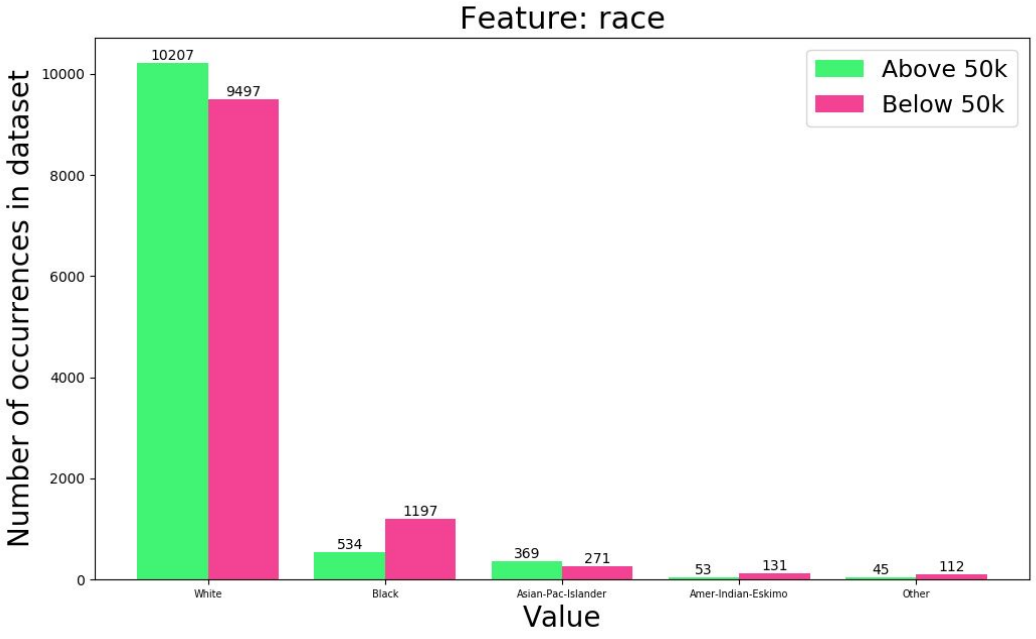


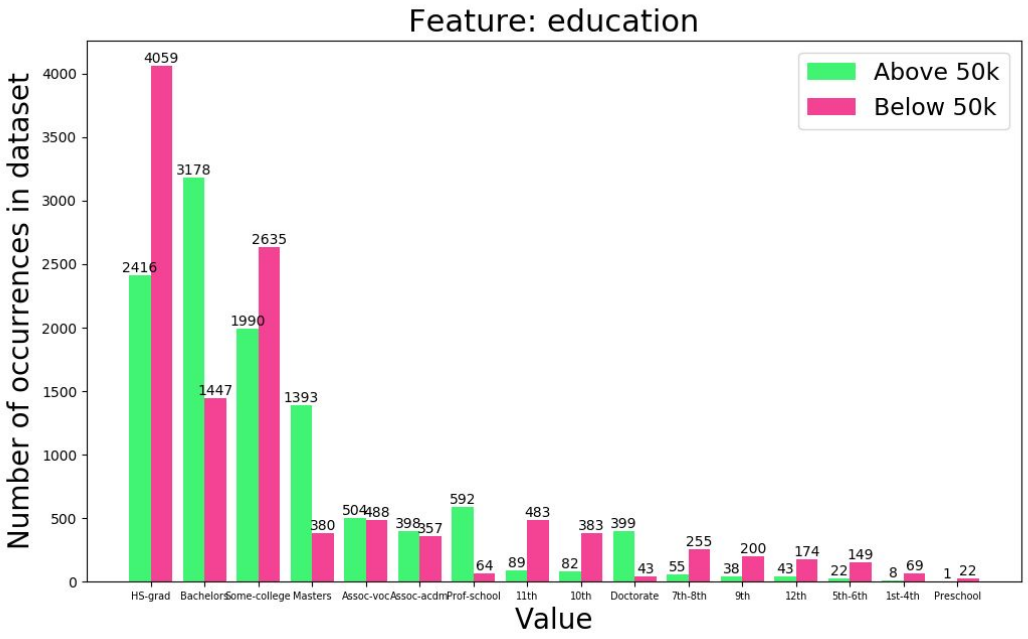
1. White people and private sector workers are overrepresented. Education levels outside of the highschool graduate to bachelors range are also underrepresented. The dataset is unlikely to be able to generalize races.
2. Education levels outside of the highschool graduate to bachelors range are also underrepresented, as are people not in the the private sector. These variations are to

the extreme, there are very few people in preschool in the entirety of the dataset, so the training for that class will most likely be poor.

Categorical Features-Bar Charts







1. Education appears to be a good indicator of salary. People with higher education are much more likely to make above $50 000. Postsecondary degrees are good predictors of being in the higher income bracket while high school and below are good indicators of the low income bracket. Race is also a good indicator of income. White and Asian people are more likely to be in the higher income bracket while Black and Native Americans are likely to be in the lower income bracket.
2. If someone is a high school graduate there is a 37.3% chance that they're in the higher income bracket. It is thus unlikely that they earn over 50K. On the other hand if someone has a Bachelor's degree it's safer to guess that they earn above 50K than not.

2.6

1. Integers apply an arbitrary scale/order to the values that doesn't exist in categorical data. There is no such thing as a race between black and white, but labelling race as black-0, asian-1, white-2 would imply that asian is somehow the race in between white and black.
2. Un-normalized data would make classification all but dependent on large value features. These features would effectively bully neurons into triggering just by being so much larger than smaller valued data, which would no longer have a substantial effect on neuron activation. The larger values also induce an initial bias into the model that the weights will have to adjust for rather than treating them as all initially equal.

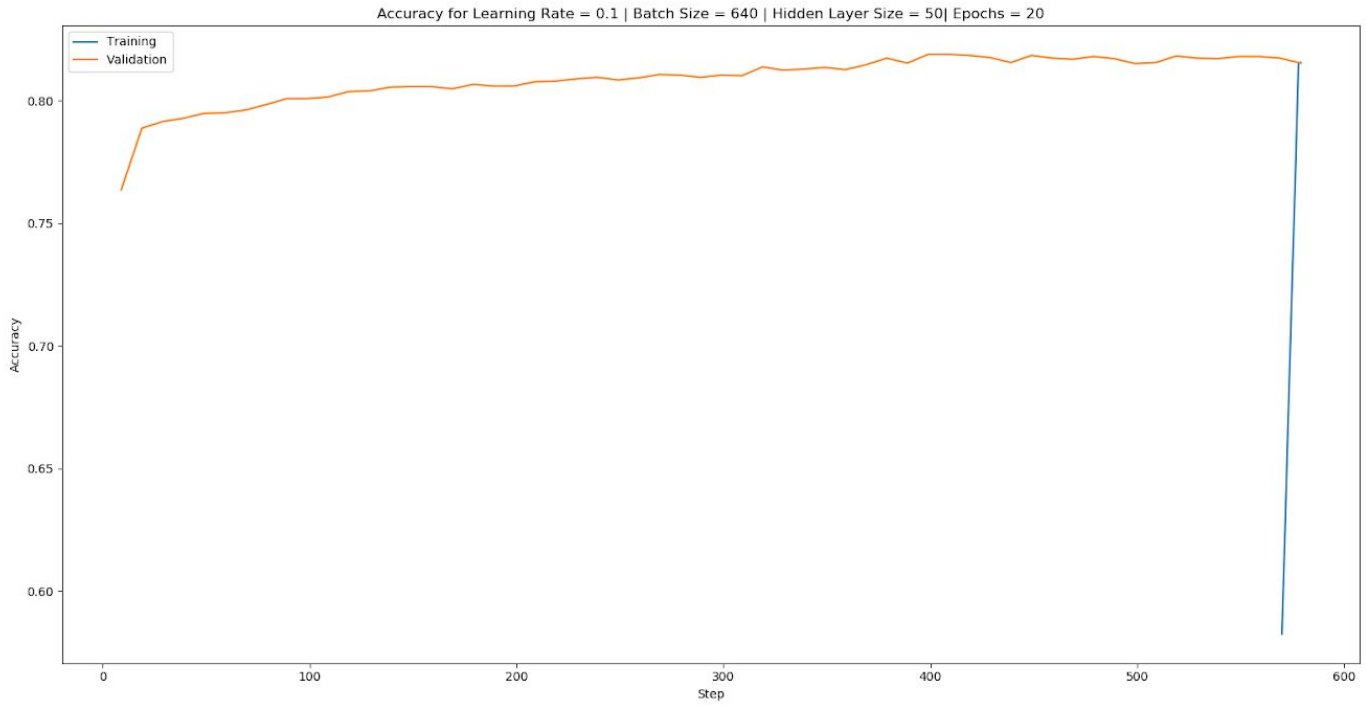
3.2

1. Shuffling the dataset decreases the likelihood that the model will reach a local minimum and halt. Changing the approach each time means that the features are not

static so the value of the loss function and thus the weight adjustments may change drastically. A variant weight adjustment will help it leave a local minimum.

3.6

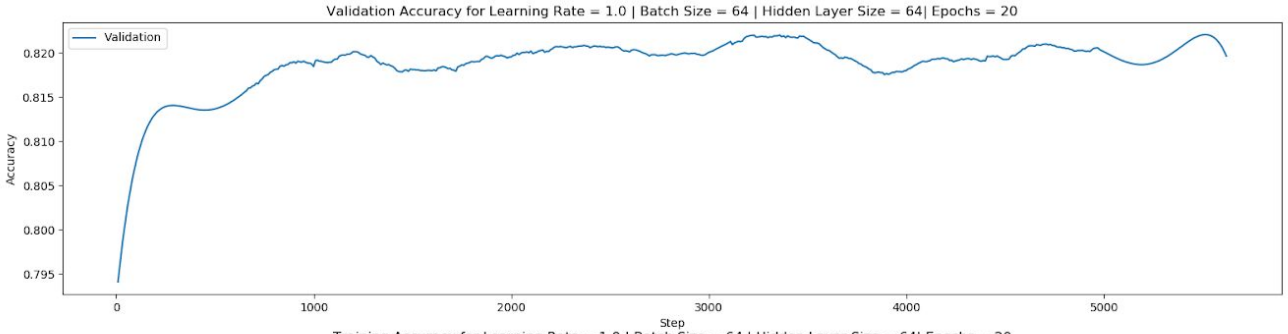
Batch size 640, Hidden layer size 50, Learning Rate 0.1, Accuracy of 0.8153434433541481

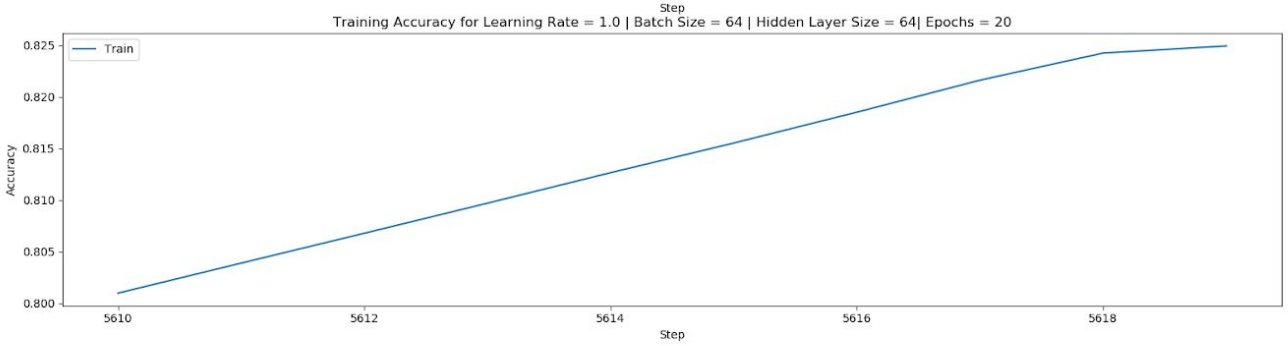


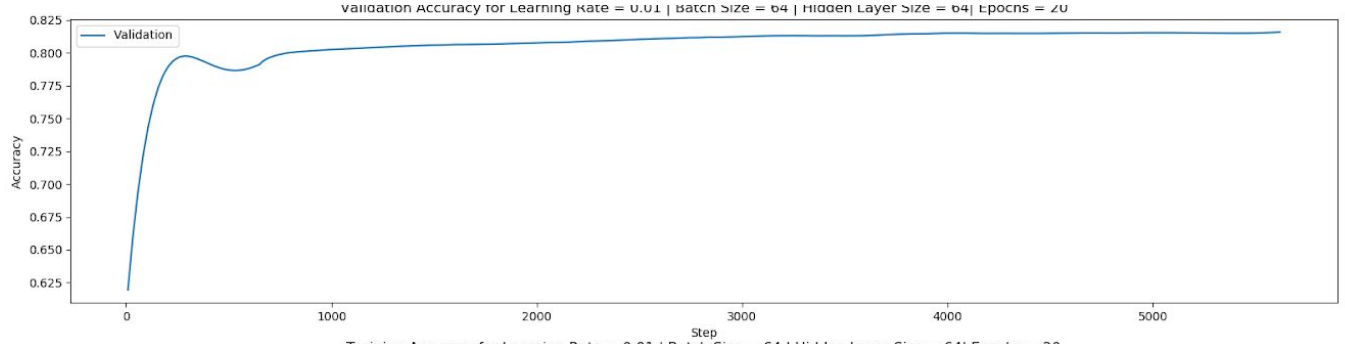
4.1

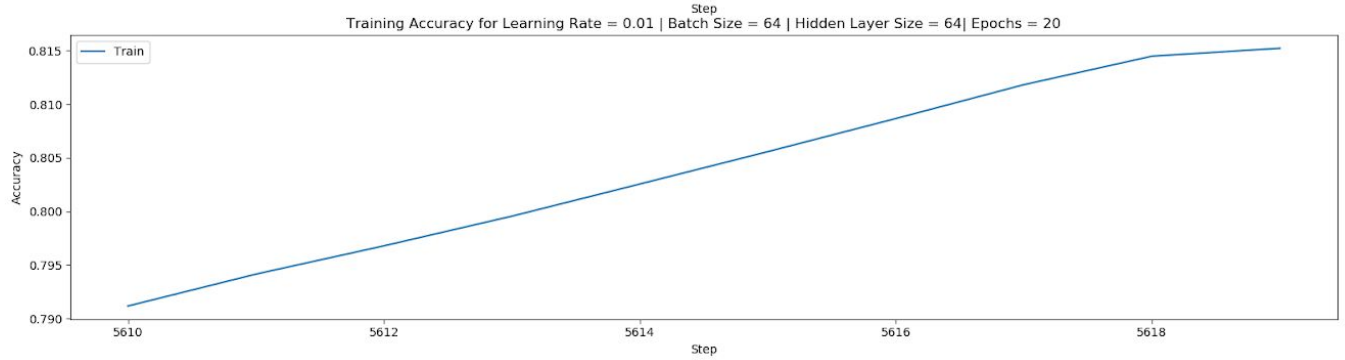
Learning Rate	Validation Accuracy
0.001	0.7954950936663693
0.01	0.8151907204996655
0.1	0.8293933987511151
1.0	0.8282783229259589
10.0	0.8111061552185549
100.0	0.5

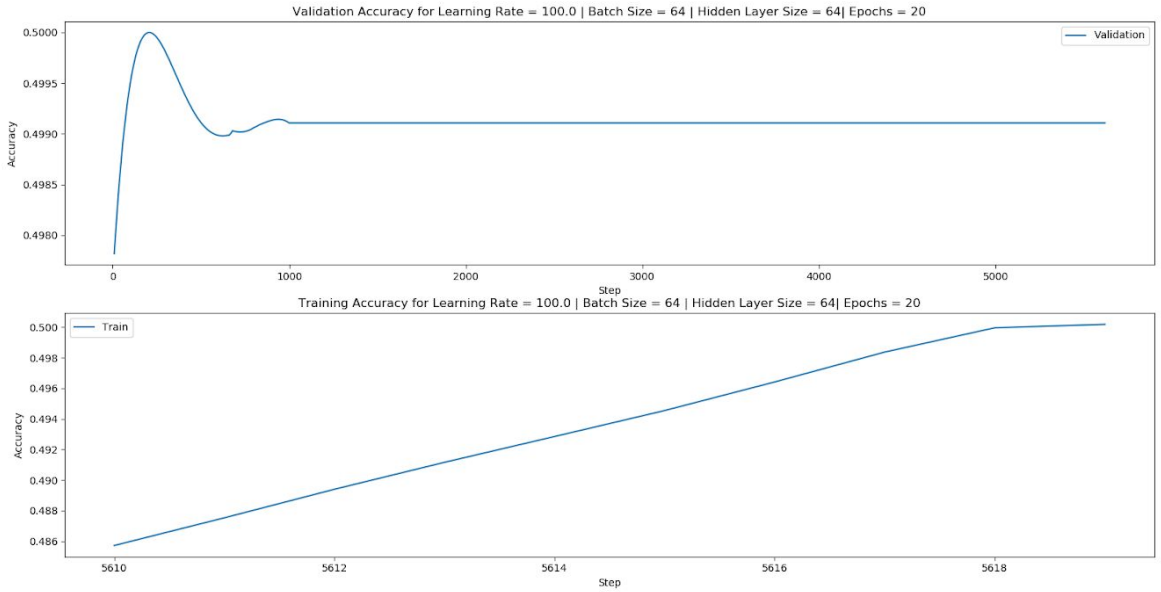
1000.0	0.7827832292595897





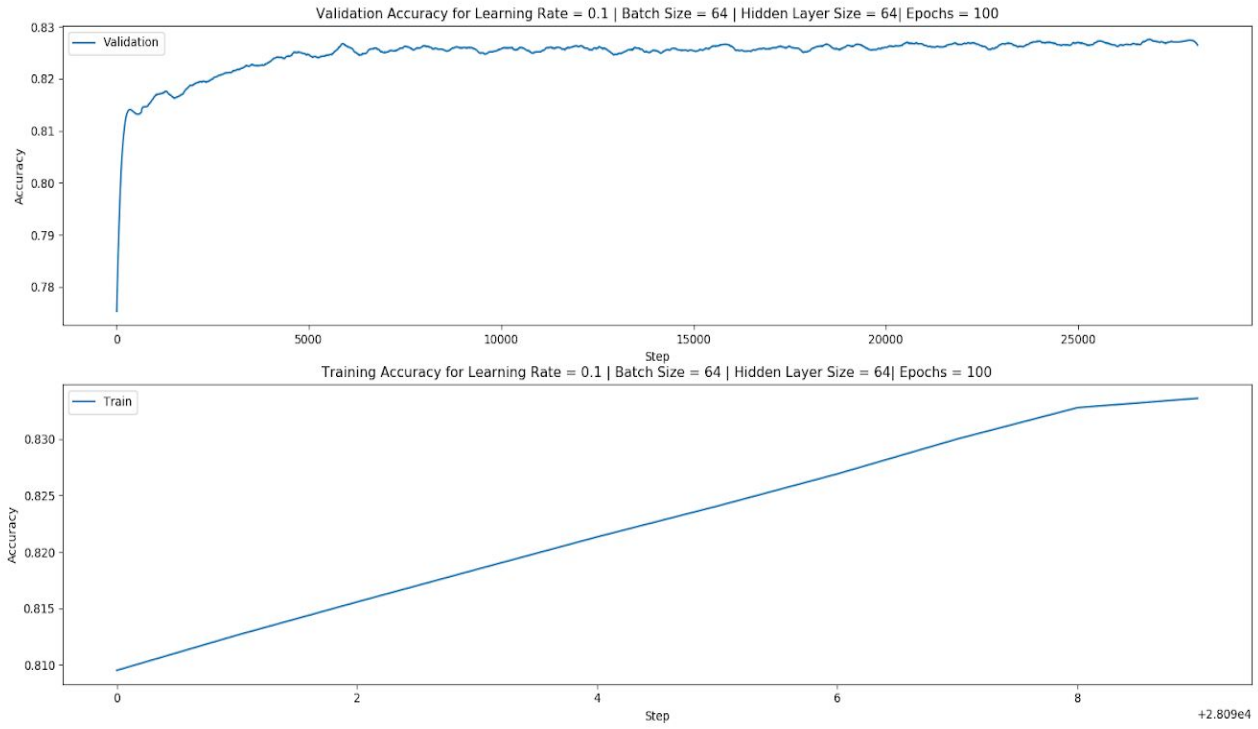






1. A learning rate of 0.1 was the best performing learning rate
2. A low learning rate fails to converge to the minima as fast as higher learning rates. High learning rates being to bounce around the minima far too much resulting in erratic, and ultimately random behaviour. Both the 100 and 1000 learning rate ended at an accuracy of around 50% indicating random classification.

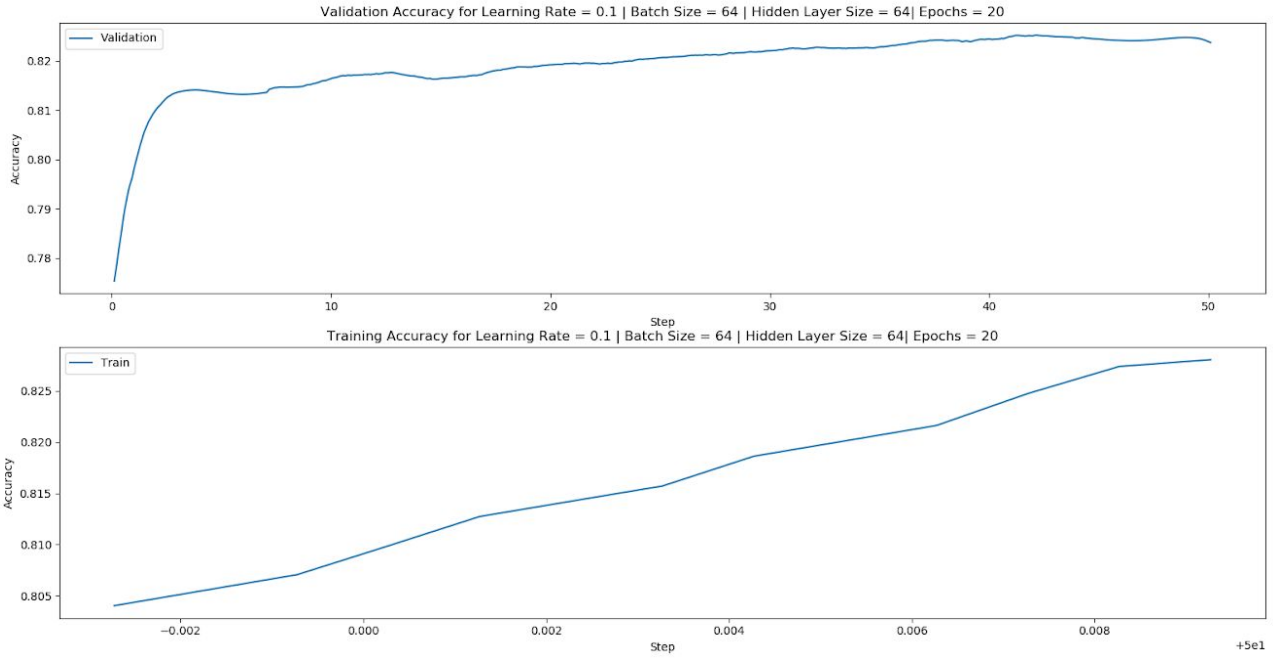
4.2

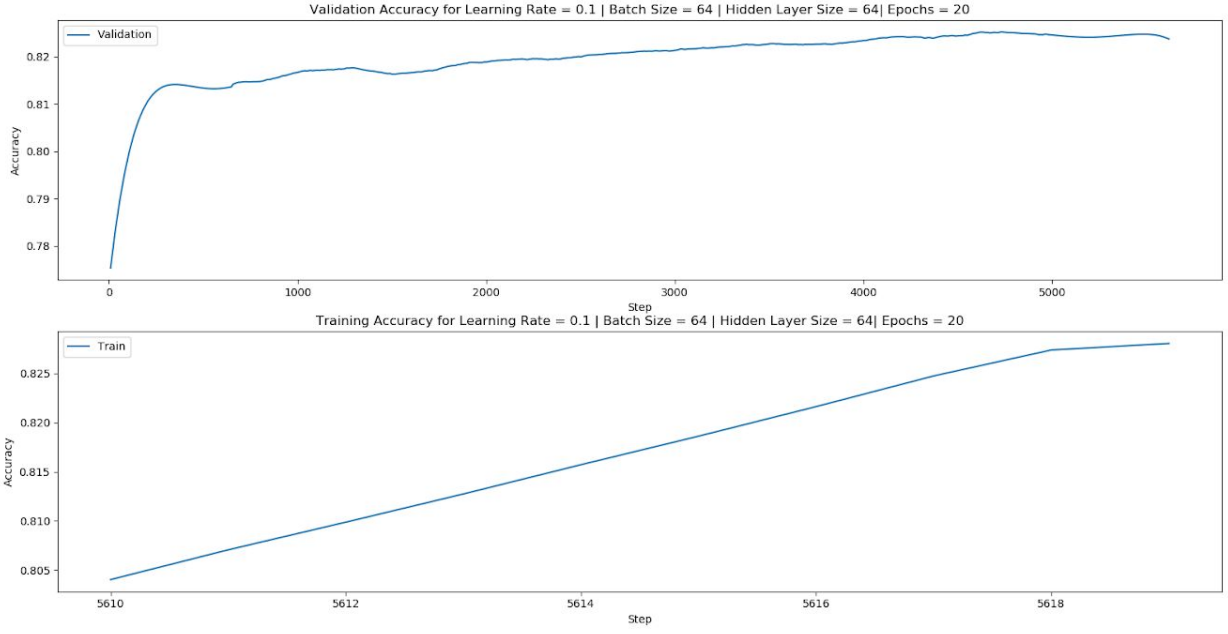


Increasing the number of epochs changes the max validation accuracy to 0.8320695807314897 at around 21190 (76 epochs) but further training appears to have no substantial effect (at up to 100 epochs/28100 steps).

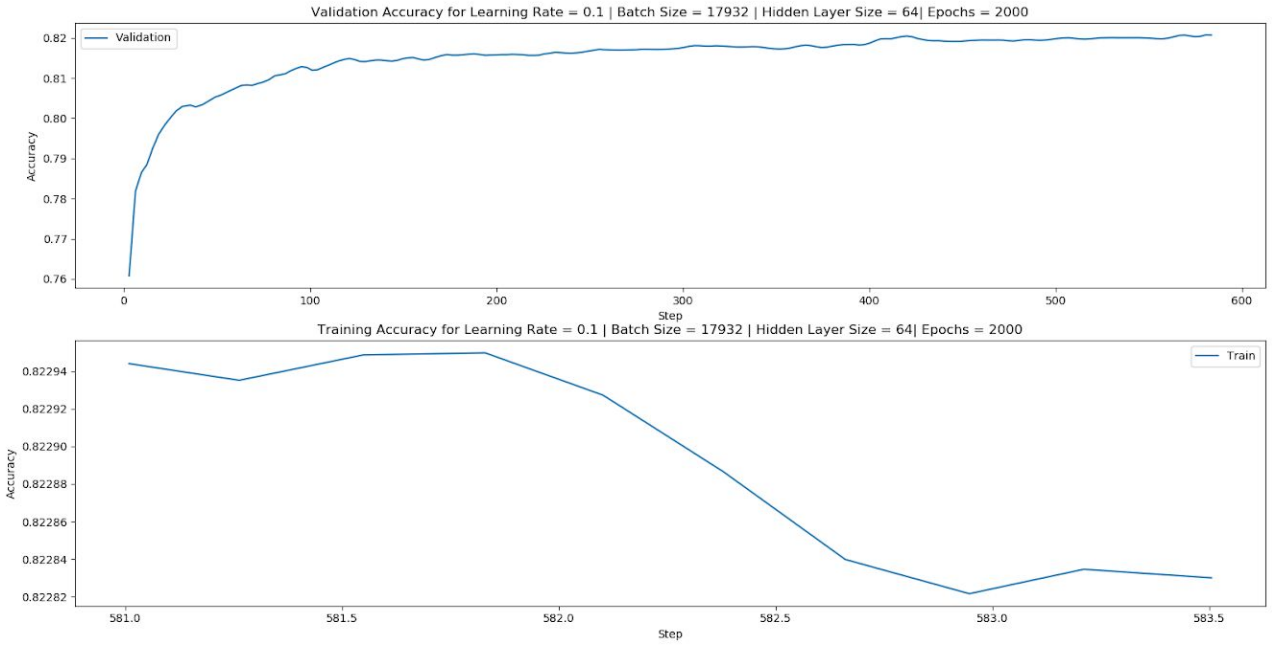
4.3

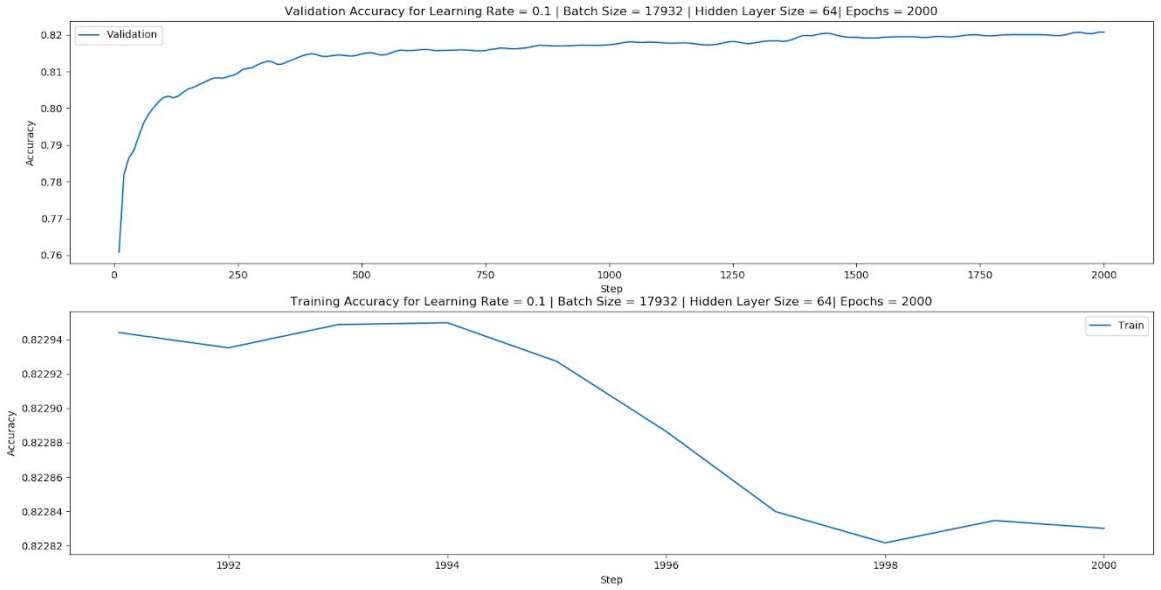
Batch size 64, Learning rate 0.1, 20 epochs (Time vs Accuracy)


Validation Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = 64| Epochs = 20
Validation
Accuracy
0.82
0.81
0.80
0.79
0.78
0
10
20
30
40
50
Step
Training Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = 64| Epochs = 20
Train
Accuracy
0.825
0.820
0.815
0.810
0.805
−0.002
0.000
0.002
0.004
0.006
0.008
Step
+5e1

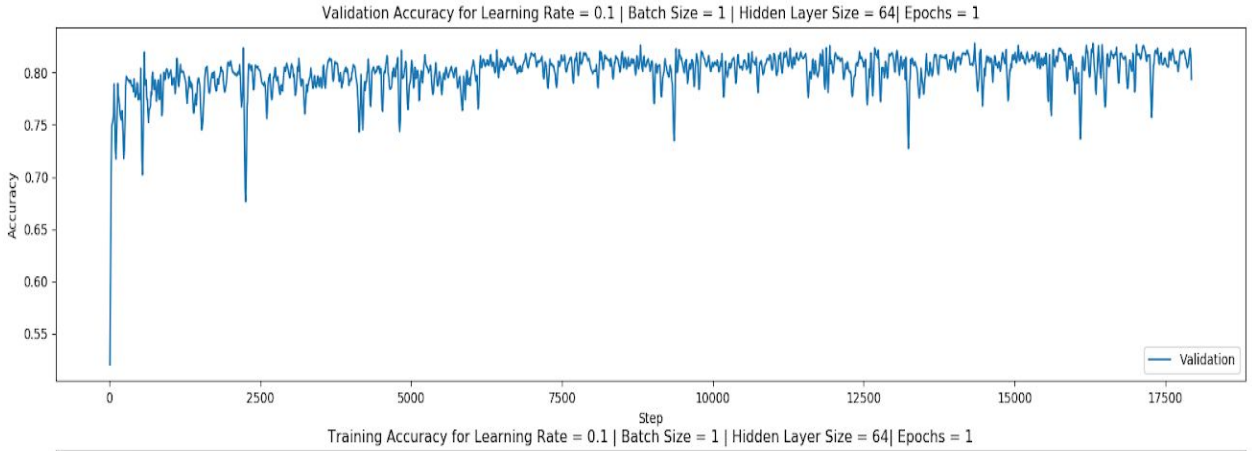

Validation Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = 64| Epochs = 20
Validation
Accuracy
0.82
0.81
0.80
0.79
0.78
0
1000
2000
3000
4000
5000
Step
Training Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = 64| Epochs = 20
Train
Accuracy
0.825
0.820
0.815
0.810
0.805
5610
5612
5614
5616
5618
Step

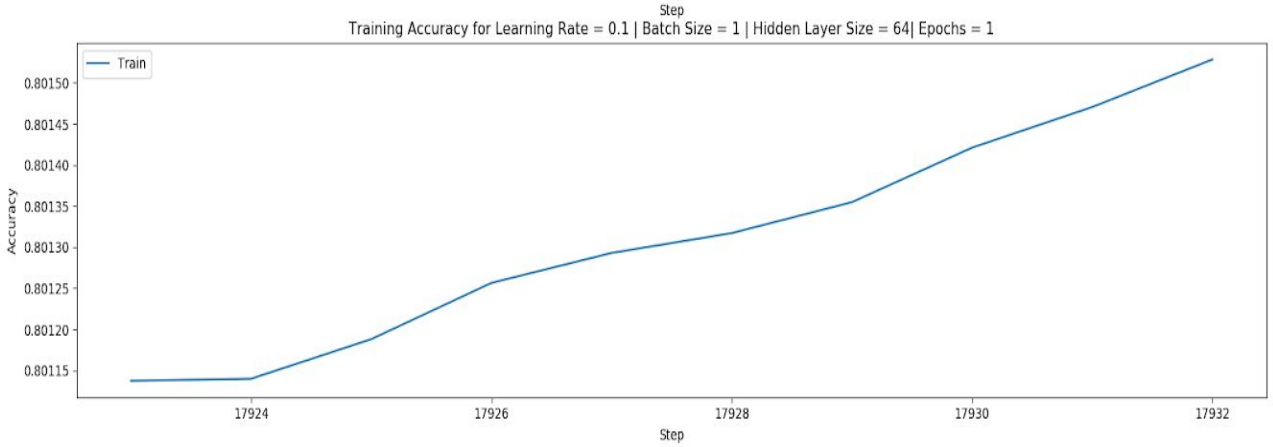
Batch size 17632, Learning rate 0.1, 20 epochs (Time vs Accuracy)

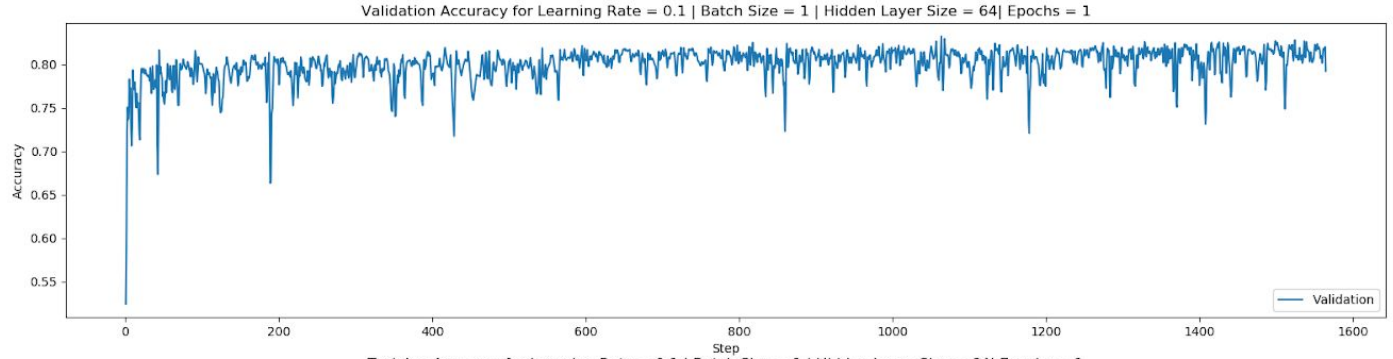


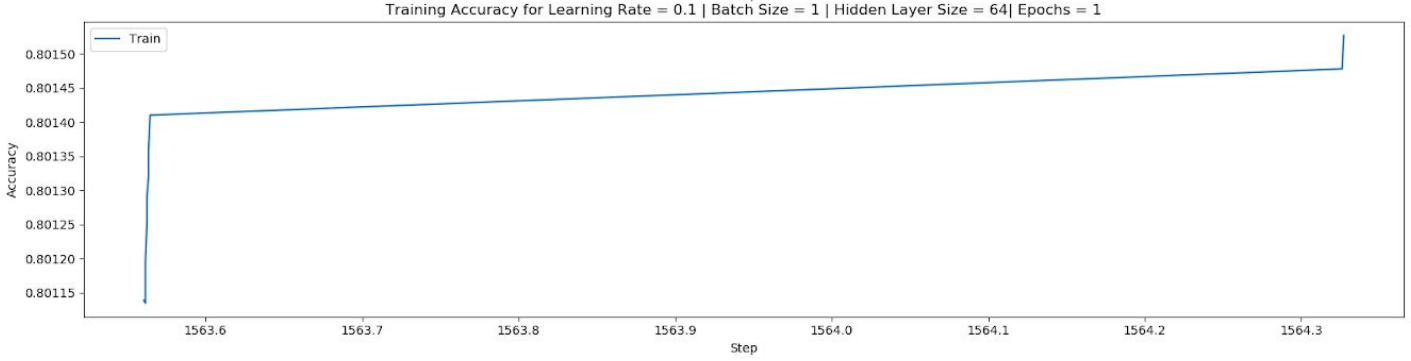


Batch Size 1, Learning Rate 0.1, 20 Epochs


Validation Accuracy for Learning Rate = 0.1 | Batch Size = 1 | Hidden Layer Size = 64| Epochs = 1
Accuracy
Step
Validation


Training Accuracy for Learning Rate = 0.1 | Batch Size = 1 | Hidden Layer Size = 64| Epochs = 1
Train
Accuracy
Step

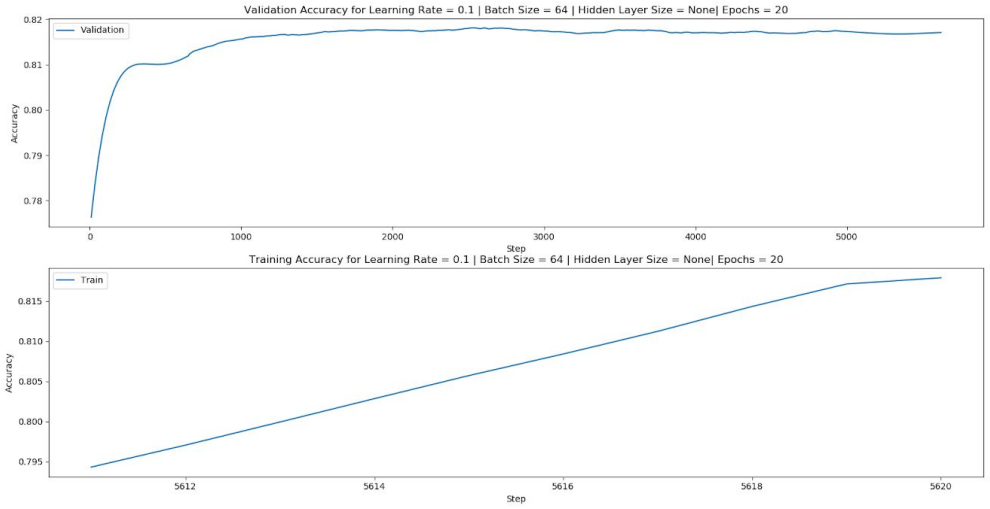

Validation Accuracy for Learning Rate = 0.1 | Batch Size = 1 | Hidden Layer Size = 64| Epochs = 1
Accuracy
Step
Validation


Training Accuracy for Learning Rate = 0.1 | Batch Size = 1 | Hidden Layer Size = 64| Epochs = 1
Train
Accuracy
Step

Batch size	Time to max	Total time (s)	Max	Steps to High Accuracy
1	1506.2285103797913	1560	0.8273862622658341	>400 for 80% (max at 17210)
64	42.584532260894775	49	0.8293933987511151	~80 for 80%
17932	566.45165848732	583.5608313083649	0.8206958073148974	~90 for 80% (1940 steps for max)

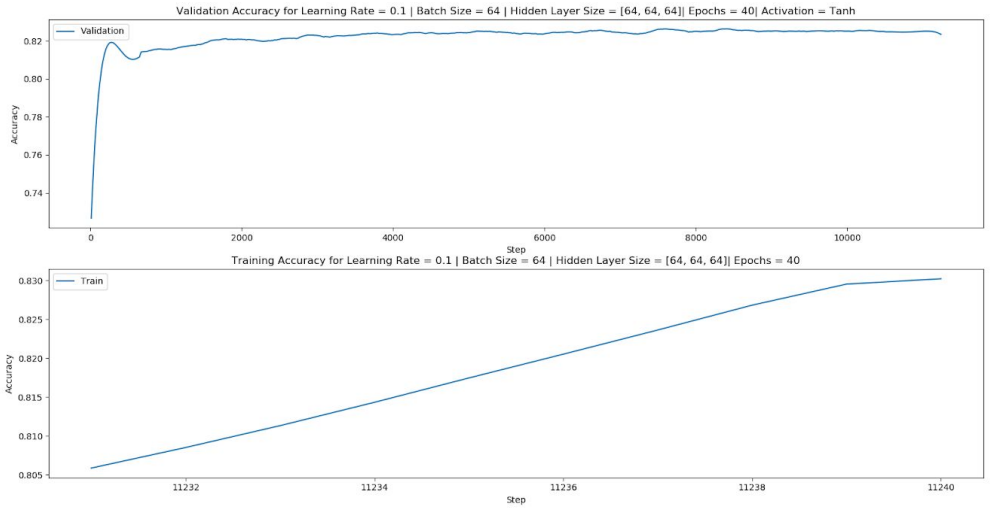
1. Batch size 64 gives the highest accuracy.
2. Batch size 64 reaches a high accuracy at very few steps, with 17932 at a very similar speed.
3. If the batch size is too low the network converges at an extremely low rate in terms of time with extremely erratic behaviour. While the rate of convergence per epoch increases as batch size decreases, the overall program slows down. On the other hand having a higher batch size requires significantly more epochs to reach the same accuracy. Additionally overall accuracy appears to decrease when the batch size is too high.
4. A small batch size leads to high computational cost but it generally allows for greater accuracy in the end. On the other hand while a larger batch size may be result in faster program time, it seems to lead to a decrease in accuracy. Intuitively this makes sense because small batches account for each example individually while large batch sizes take the gradient with respect to a larger amount of the examples, so it makes sense that it will fail at the individual classification level.

4.4



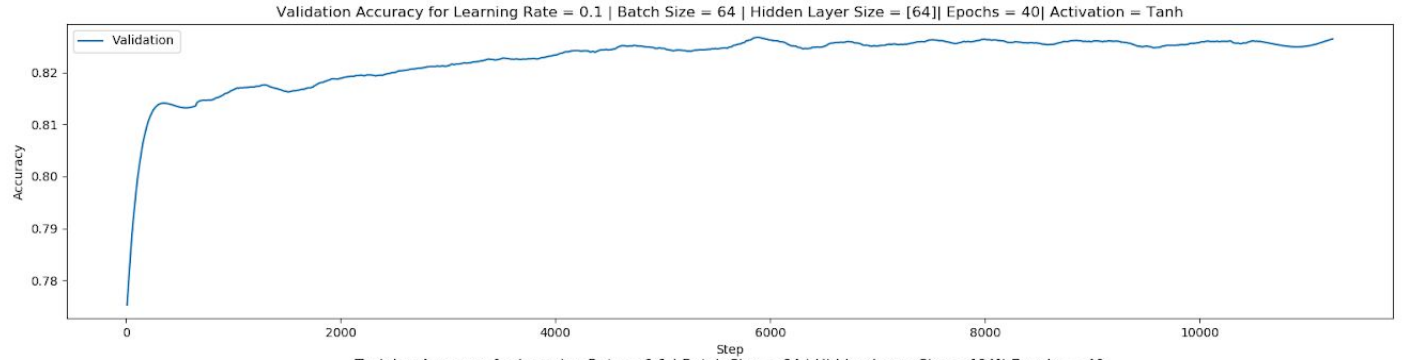
1. The accuracy peaks at 0.8178674994423377 and the model fails to continue generalizing and learning the data. The accuracy levels off at this point, falling short of 82.9% with at least one hidden layer, indicating underfitting.

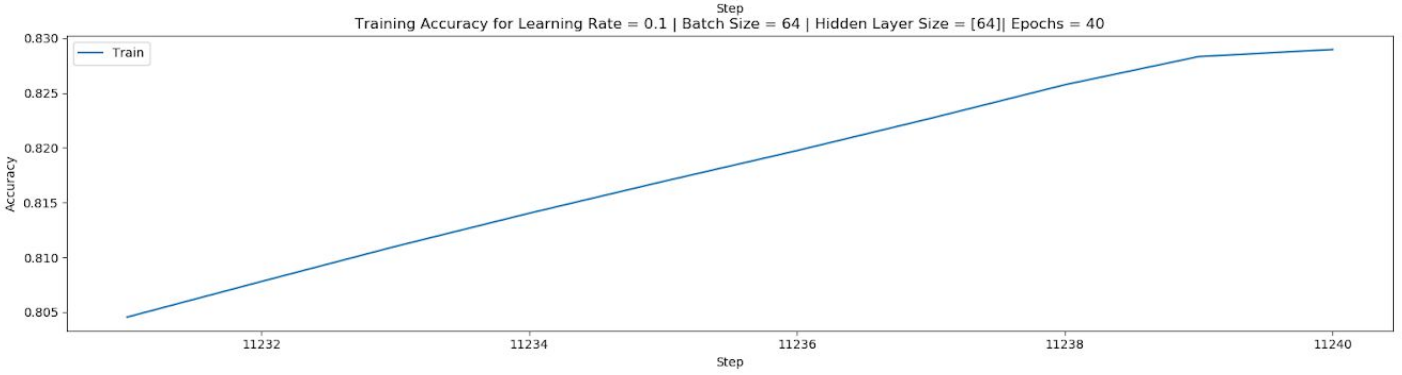
4.5

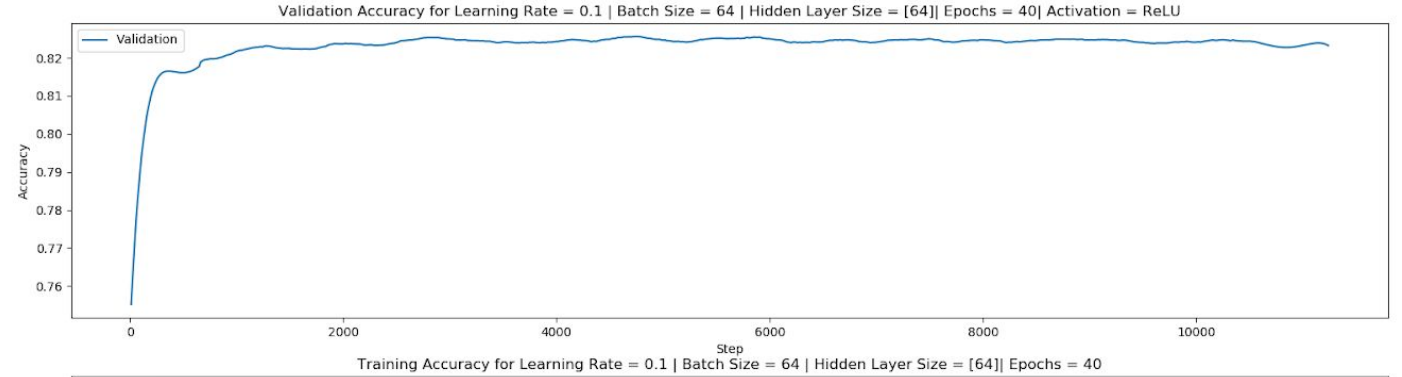


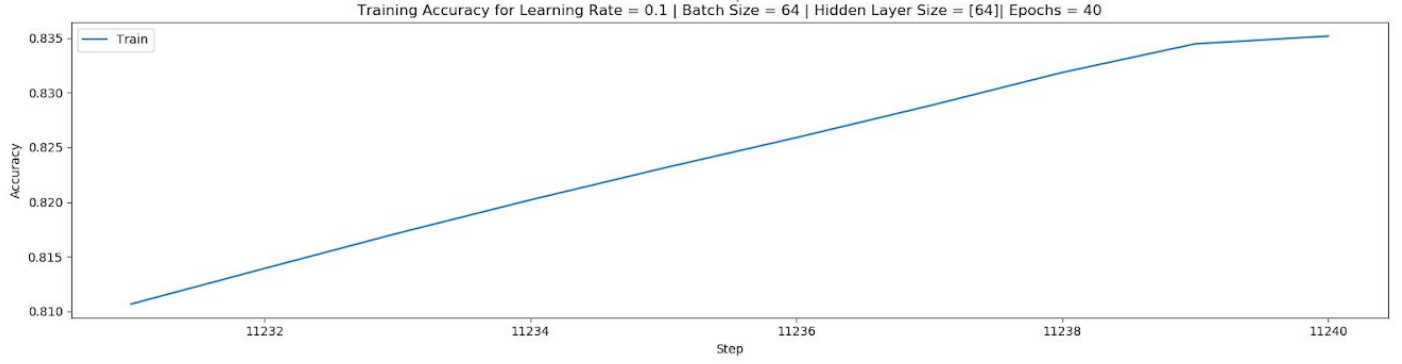
1. At 40 epochs (11240 steps) this reaches a maximum of 82.96% validation accuracy and a training accuracy of 82.95%. On the other hand the one layer model achieved a validation accuracy of 83.07% at the same number of epochs. The model achieves less accuracy with even more parameters, and it is thus overfitting.

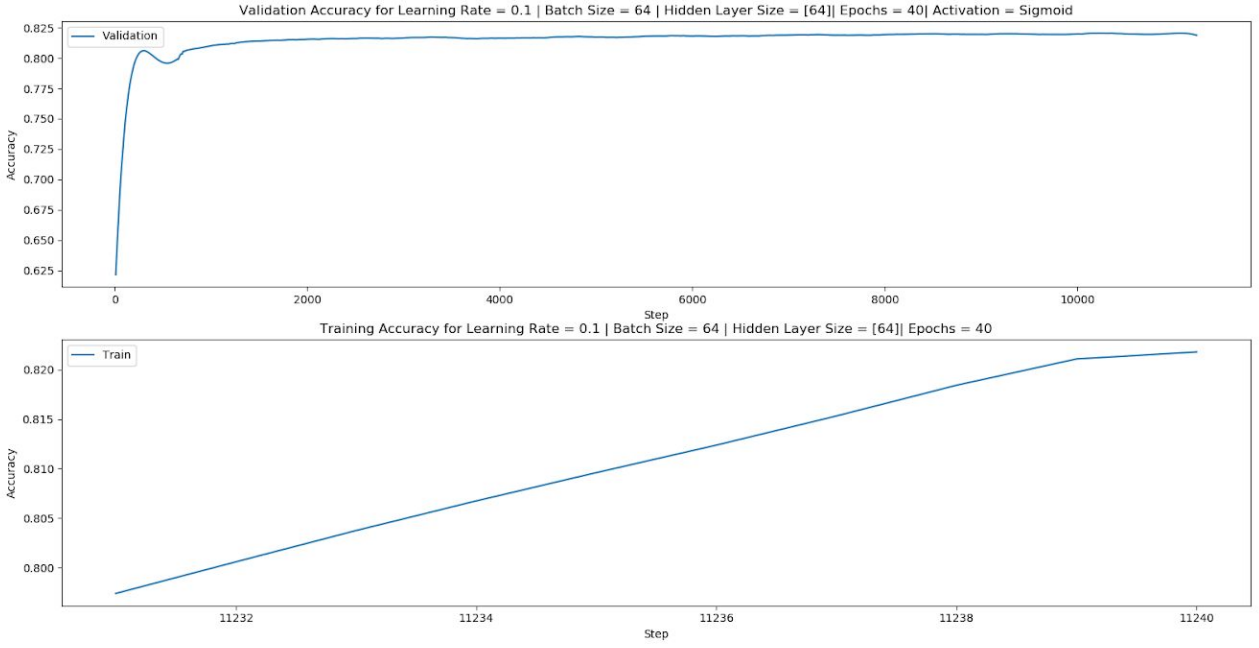
4.6


Validation Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = [64]| Epochs = 40| Activation = Tanh
Validation
Accuracy
Step


Training Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = [64]| Epochs = 40
Train
Accuracy
Step


Validation Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = [64]| Epochs = 40| Activation = ReLU
Validation
Accuracy
Step

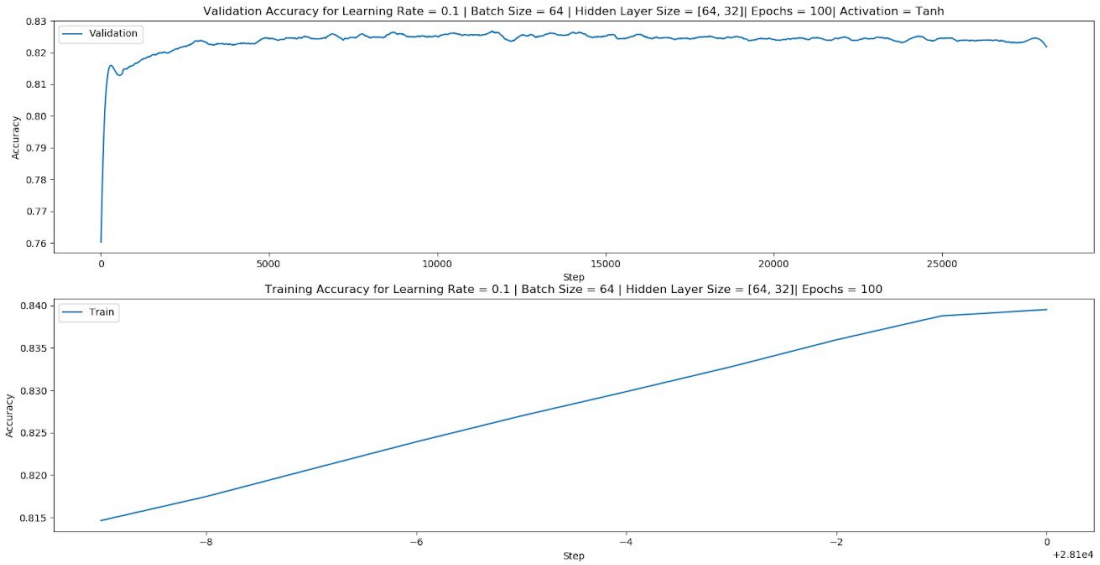

Training Accuracy for Learning Rate = 0.1 | Batch Size = 64 | Hidden Layer Size = [64]| Epochs = 40
Train
Accuracy
Step



Function	Time (s)	Accuracy
Tanh	92.59180474281311	0.8307314897413024
ReLU	97.06292676925659	0.8298394290811775
Sigmoid	101.64418625831604	0.8240410347903657

It appears that the functions with faster run times reach greater accuracy. Tanh has the best accuracy but seems to converge in the least amount of steps. It has a lower initial slope than the other two functions but peaks higher.

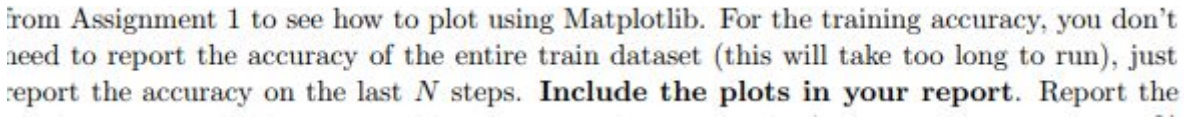
5



At 50 epochs, a batch size of 64 and two hidden layers with sizes 64 and 32 respectively, it was found that the validation accuracy for these parameters was 0.8314005352363961.

6

1. Feedback
    a. The assignment took far too much time. It spanned about 20 hours of work, 14 of which were just spent on graphing the same stuff over and over again. It was an extremely repetitive task and it felt difficult to distinguish this from assignment 1. I felt that cycling through 7 learning rates, 3 batch sizes (plotting against time and steps), 3 activation functions, and various other hyperparameters was a tedious and completely uninspiring task with little to no educational reward considering the previous assignment. I felt unengaged as I sat waiting for my computer to run code with parameter changes that I fundamentally don't understand. The fact that we've learned little to nothing about the actual statistical theory and mathematics behind these hyperparameters meant that testing hyperparameter changes felt akin to poking the body of a dead specimen with a stick and seeing what happens to learn its biology.
    b. The instructions were extremely challenging to understand. In particular:


rom Assignment 1 to see how to plot using Matplotlib. For the training accuracy, you don't
need to report the accuracy of the entire train dataset (this will take too long to run), just
report the accuracy on the last N steps. **Include the plots in your report**. Report the

This section in particular was extremely unclear. The instructions say to plot the accuracy on the last N steps. This could very easily be interpreted as the very last N steps of the whole training set. Of 5000+ steps this would be a measly final 10 steps of the whole training loop. On the other hand, it was made clear far too late that this is actually in reference to an average across every N steps (as opposed to even a sample every 10 steps). It was made even more confusing when we were asked to sample the validation accuracy instead of merely evaluate that for the last N steps. Why we would only sample the validation accuracy, whose max is asked for multiple times but average the training accuracy across those same N steps is beyond my comprehension. We were asked numerous times for a max validation accuracy, yet we were expected to sample this and not the training set. It would have made far more sense to evaluate the validation accuracy every step, and the training accuracy to be sampled, or maybe create some consistency between the two and just have both be sampled. I hand in this report knowing full well that I did not follow the instructions "properly," however I reserve all blame to the lack of clarity in the instructions. I do not have any interest in redoing 14 or so graphs over the course of 2 days. The instructions itself were rife with mistakes, even referencing its own code incorrectly several times, such as when they claim that the default value for the epochs parameter is 100 when it's 20.
    c. I enjoyed the part of the assignment in which I actually built the neural network. However even this was flawed. My code relied almost entirely on the Lecture 10 code, which was just given to us. This undermined my learning significantly seeing as I was essentially given the code on a silver platter and

not forced to actually learn the material itself. The nature of a neural network still remains a black box to me, and I expect this to continue to be true for the remainder of the course.

d. The instructions were poorly written and clarified. I was forced to redo some of my graphs multiple times and still seem to have ended up with the “incorrect” answer.
e. The Lecture 10 and 9 code were pretty useful seeing as they did the fun part of the assignment for me. Otherwise, I think learning pandas and torch will be extremely useful for me in the future.