Neural Network Report

For this Neural Network project, I used google colab as the environment since it has various built in libraries. I used pandas, numpy, tensorflow/keras. I also imported pyplot but I was unable to use it. I began with reading the csv file and used pd.getdummies function built in with pandas to start 1 hot-code the csv file. Then I divided my set into training set (70), test set (15), and validation set(15). I then popped the scores out of the 1 hot-code data sets, then I put the scores together as their stand alone set call “labels”. I start by building up a linear model for the datas then proceed with the deep neural network. I start with using 1 single input, in this case I used the data for ‘gender\_female’ column. Then I create the training model for multiple inputs. I run my deep neural network for each of the train/test/validation set (70/15/15), then I print there “Margin of error” which is just the improvement of the model depends on the set’s sizes and iterations. The more iteration (larger epochs), the better my training results are (less margin of error). My net work went through 2 layers (“relu”) of training each iteration. The way I display the margin of errors is by saving each of the train or test results into “test\_results” container. Then I simply displayed it. I have to say, I am still clueless on what I am supposed to improve and what I am supposed to display as results, but that was my best guess. Validation strategy, I would say I used my validation set which is the last thing that is displayed on my results table, as a comparison measurement on how well my training and testing sets did. Other than that, I do not understand what I was supposed to use there. I am sure the more I study about tensorflow, the more this will make sense and I can train my network better. Even though I do not understand, I do believe, if I can split the data and manipulate them better, my network will yield better results. Network config CPU times: user 3 µs, sys: 0 ns, total: 3 µs

Wall time: 7.15 µs