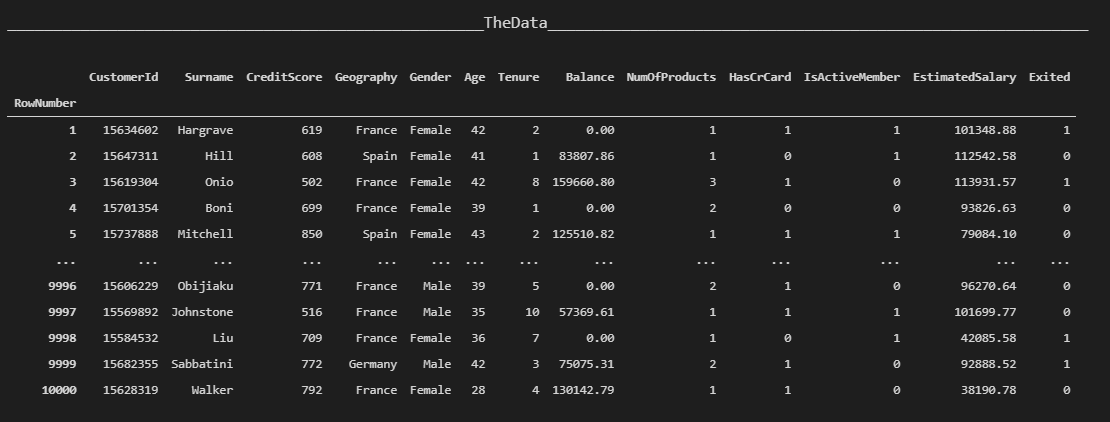
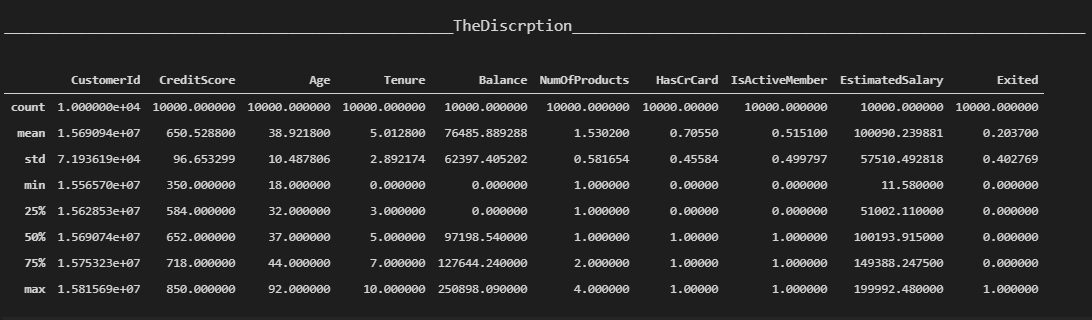
گزارش پروژه یک

سروش حیدری :: 96222031

Main idea and a general explanation:

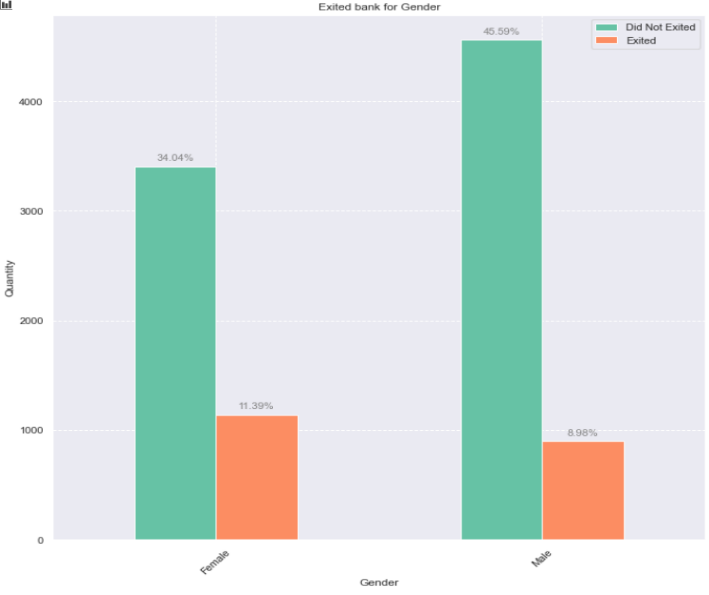
First of all, to get a better sense about out data and the correlations of the columns and of inputs and outputs we’ll use “panda.describe()” function to have a general description of our data, which is shown below

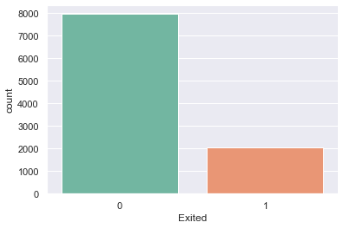


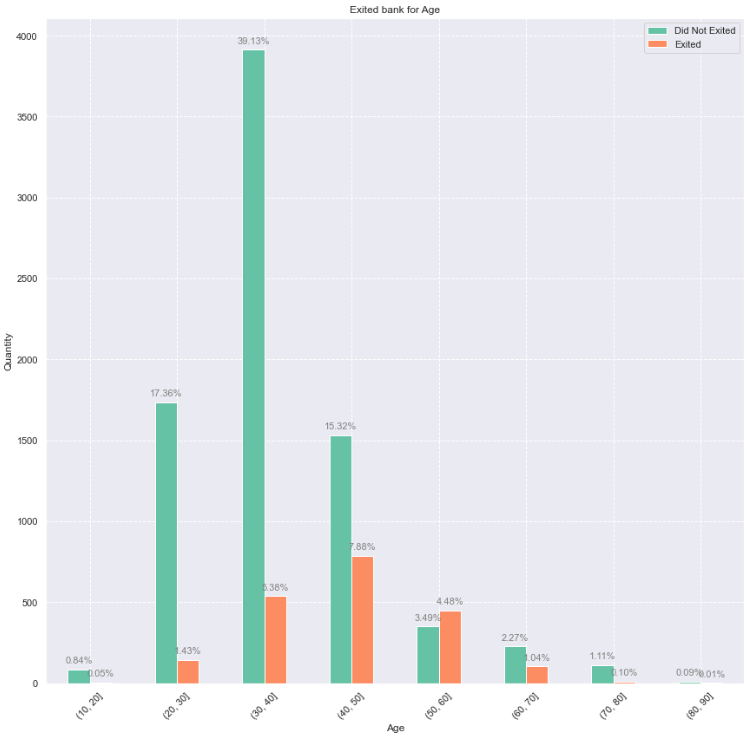
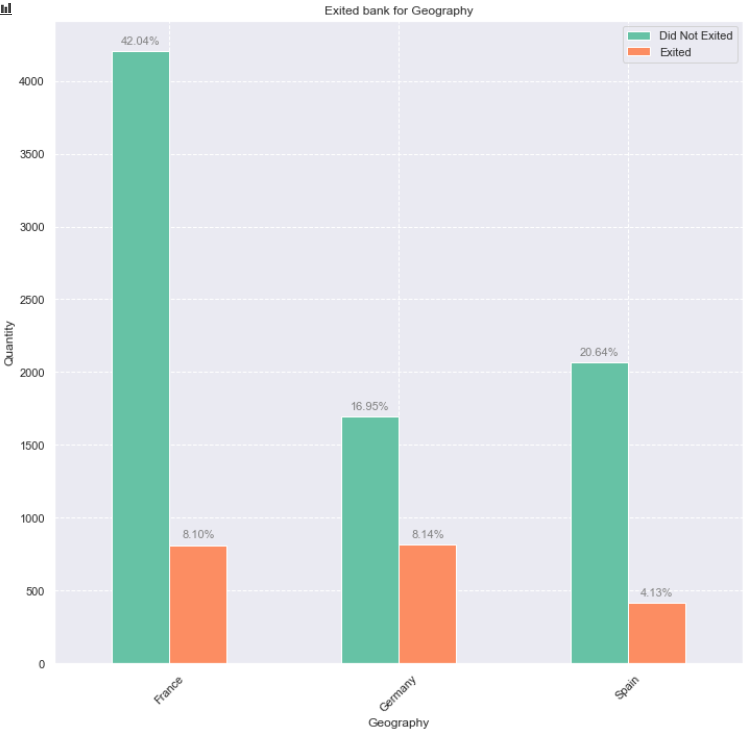


Also to get a better understanding of the correlations we use the library “seaborn” (a part of the functions used for showing the bar\_charts are taken from an external source which I unfortunately lost the link to and I’m NOT to be credited for it! )

The percentage of people exited vs those of whom didn’t and the relation it has with gender :

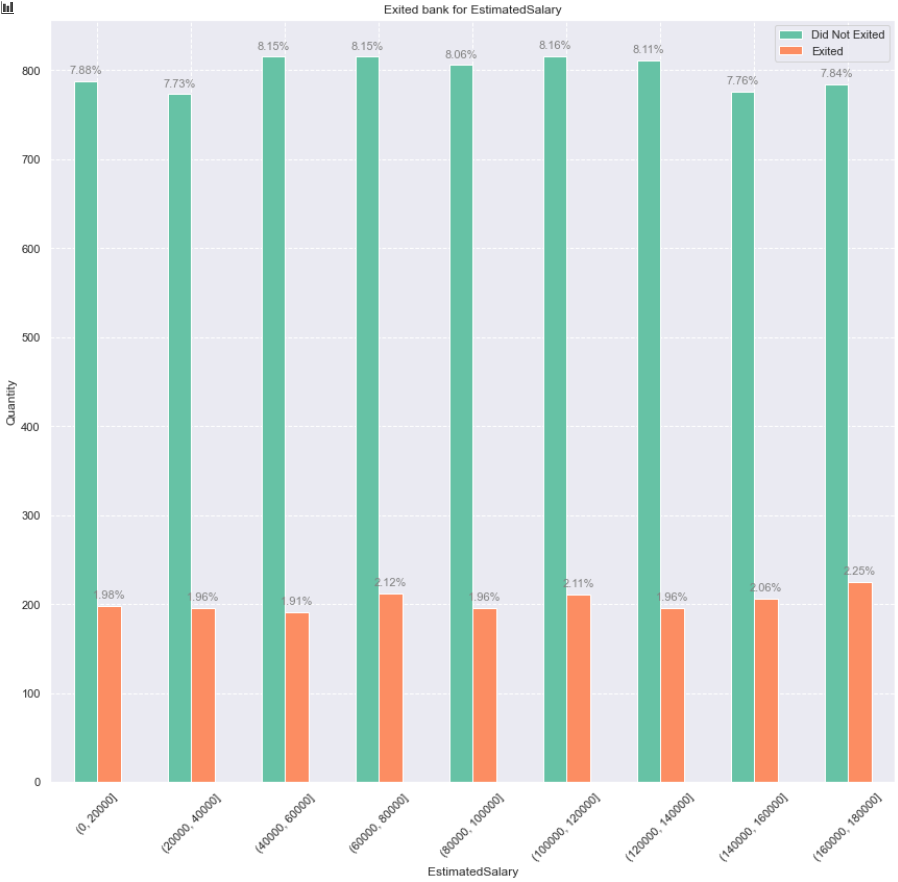
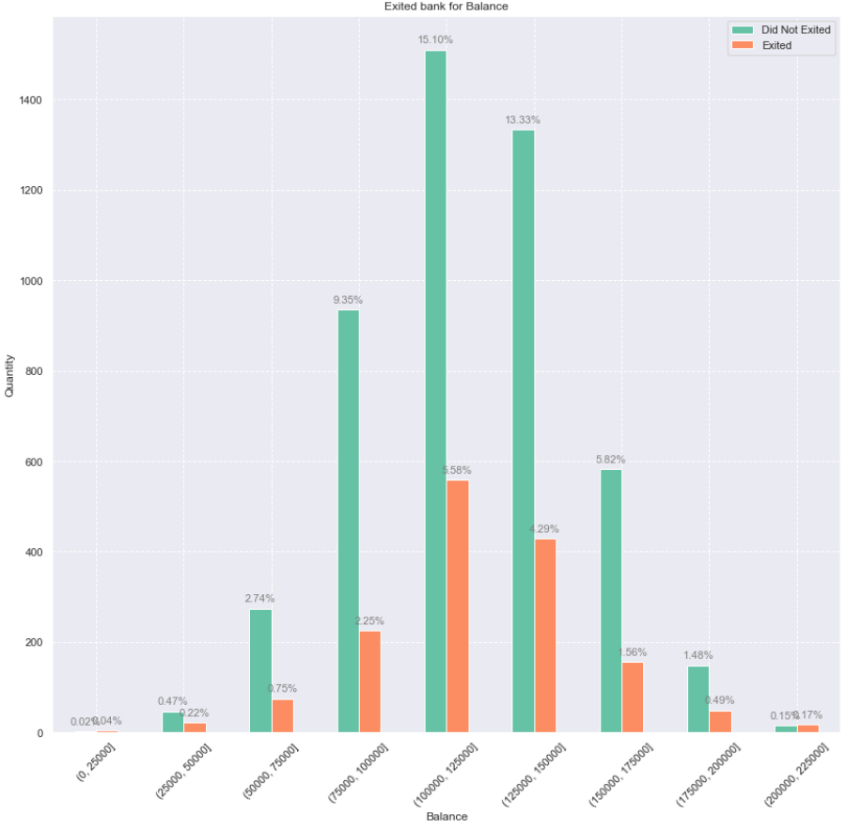
* This can show us the roughly 80 percent of people didn’t quit the bank also females are more likely to quit



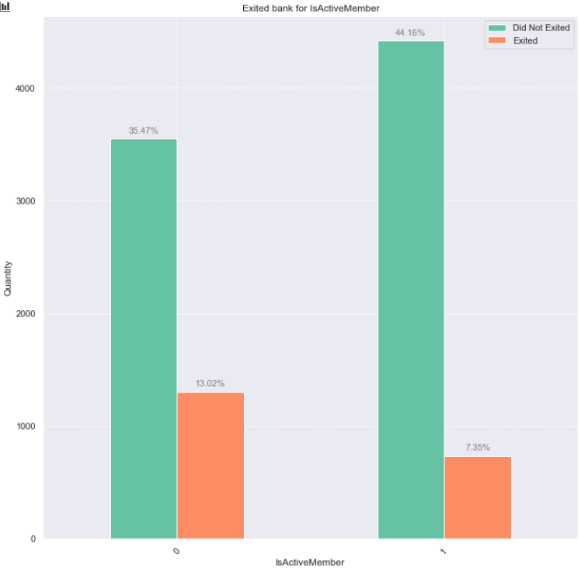
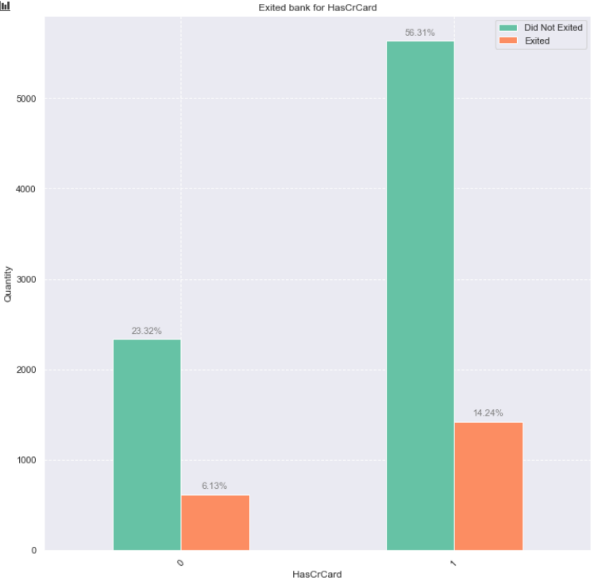
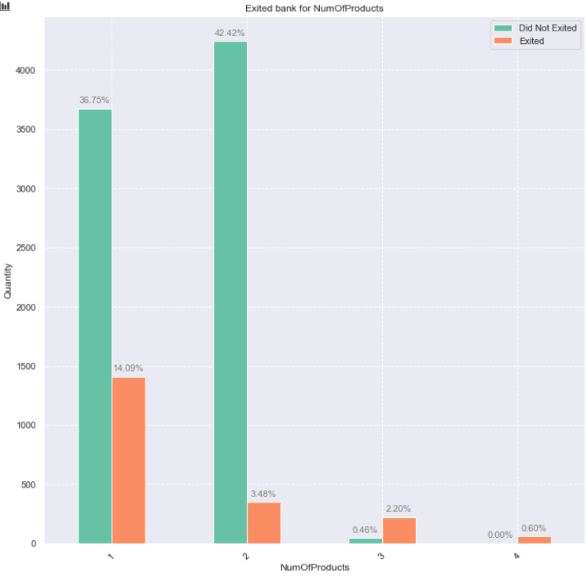
The percentage of people exited the bank in according to their geography and age :

* What these 2 charts can show us is that people in Germany usually quit the more likely to that of France and spain, also that people who are in between 30 and 40 age are very less likely to quit the bank than other age ranges

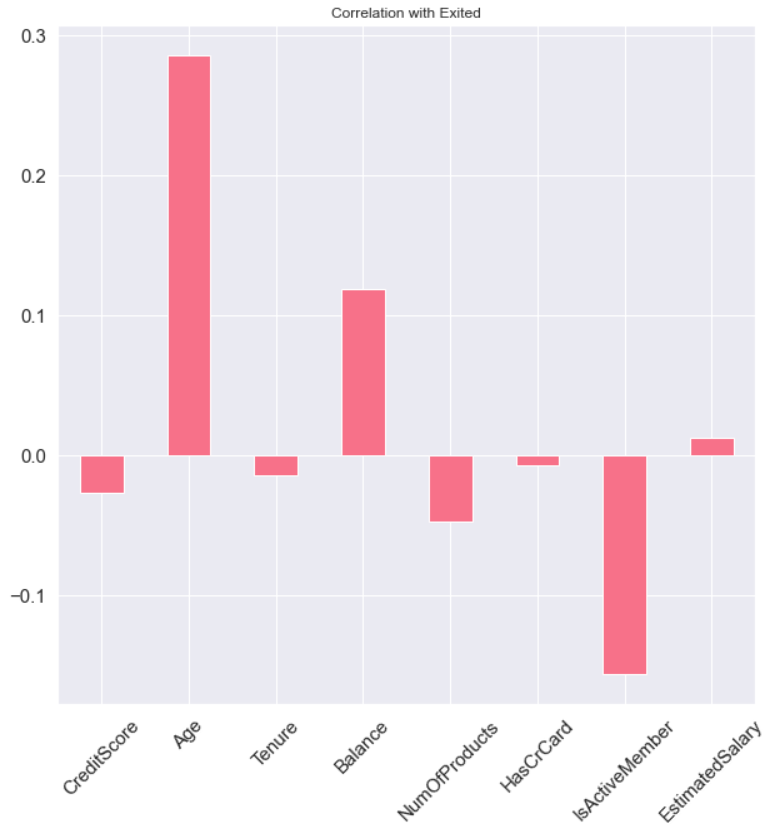
The percentage of people who exited the bank in according to their balance and their salary



What we can perceive from these two charts is that unlike the “Balance” feature, the “Salary” doesn’t really affect the Exit column

The rest of the charts don’t really show us anything, though I put them here just to be more exact!

And a final chart showing the the effect of each feature to predicting the exit column :



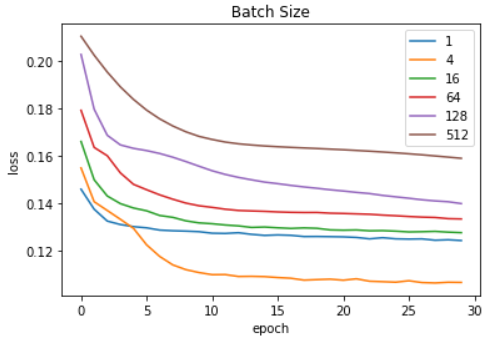
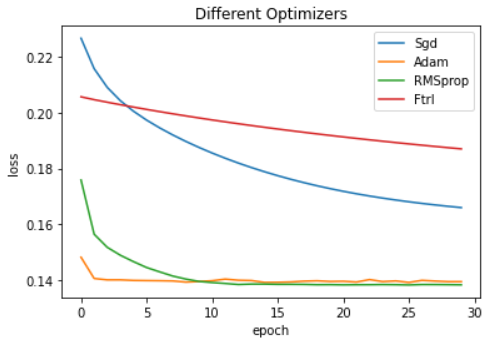
Results analysis :

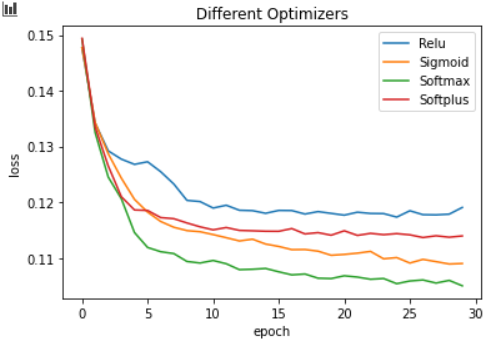
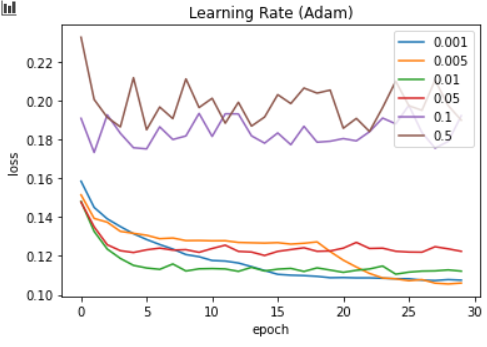
There are a couple of comparisons that I made to show how changing the the hyper-parameters like learning rate and optimizers functions and … could affect the loss function in according to each epoch in progression of our network, keep in mind that these charts are not in the most optimized state only a network so we can compare our desired comparisons

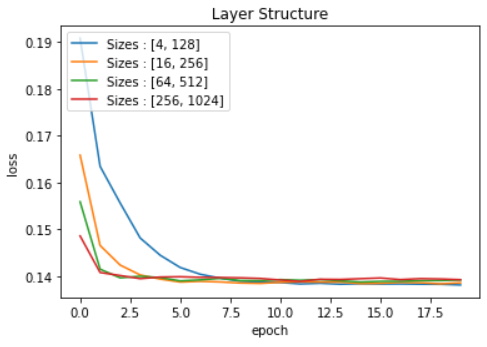
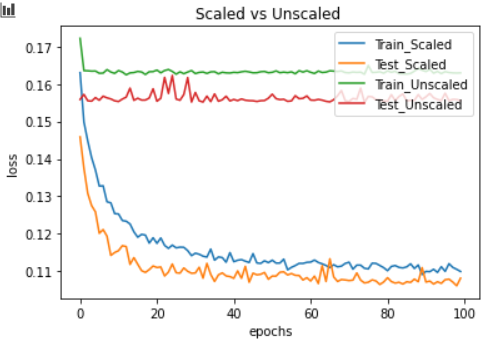
The default of the network properties used in these comparisons are :

* Layer Structure And Sizes : Layer 1 : 4 nodes , Layer 2 : 128 nodes
* Batch sizes = 4
* Optimizer function = Adam
* Learning rate = 0.005
* Activation function = SoftMax
* Loss function = mse(Mean squared error)
* Also I’m using a normalized version of the data and I have brought the unnormalized version in a comparison too and you can see that the normalized version works much much better

Each of the corresponding comparisons defines the mentioned property and the results are as following

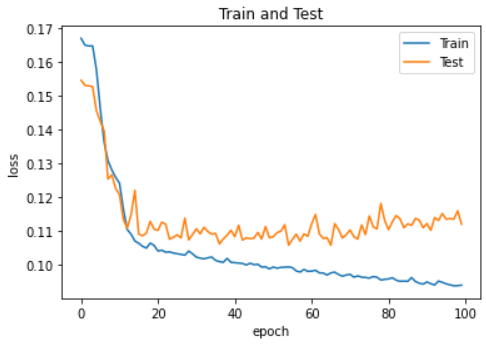




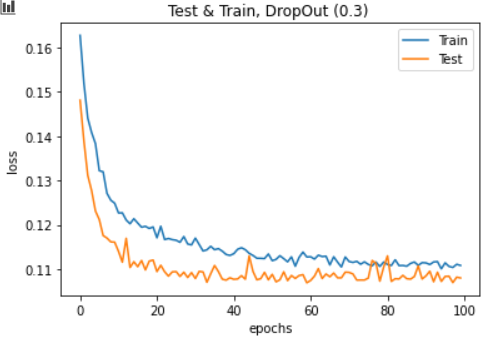


The charts seen above a default version of the network which doesn’t actually lead to overfitting though if we change the network’s properties we might get an overfitted network which is not our ideal

* Here I changed the layer structure(12, 24, 128) to a more complex one and made more epochs for training

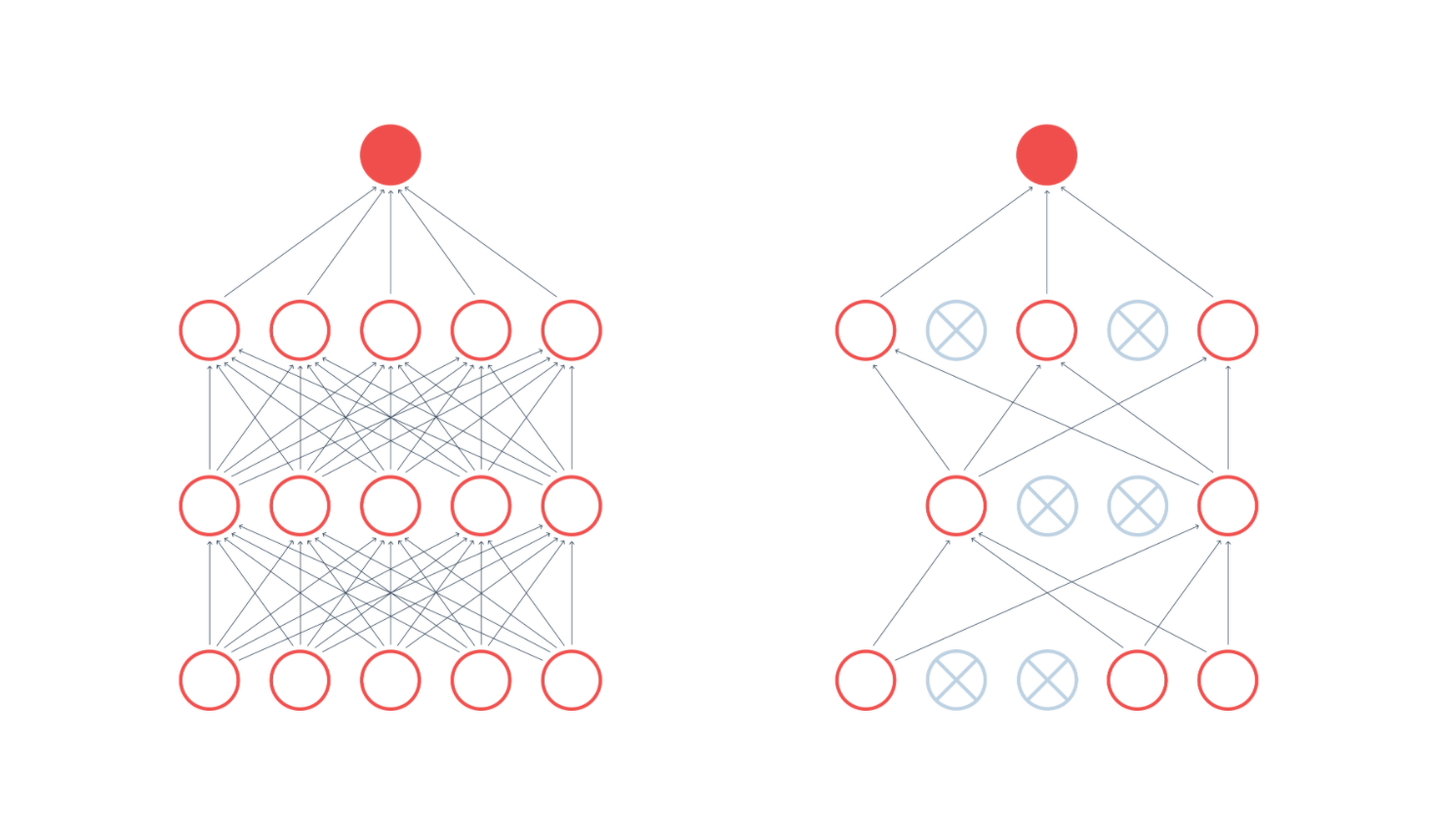


And after applying DropOut technique with 30 percent chance of omitting a node we’ll get this !

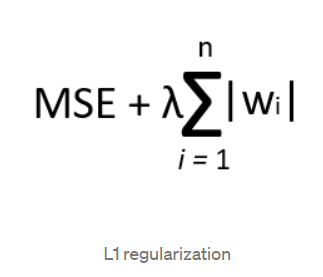
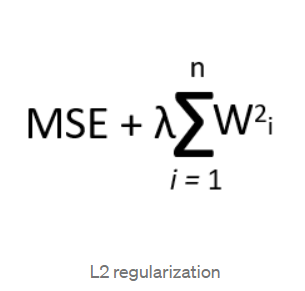


Which is actually quite nice and we can see that now our validation data has error percent of 10 and lower

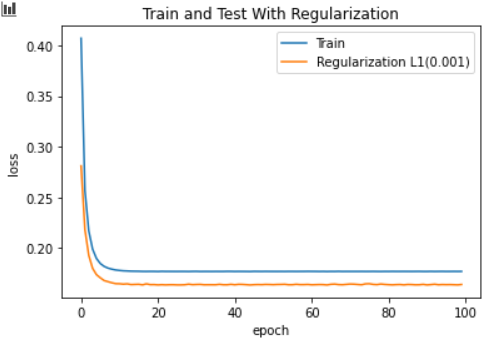
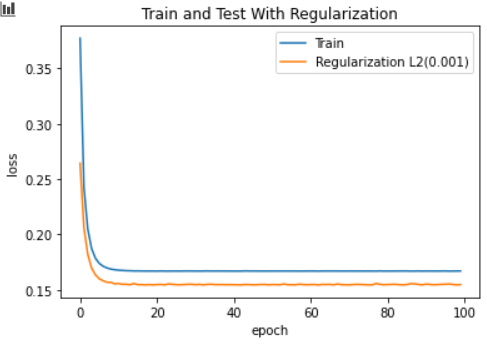
Just as a note in this technique we’re removing some of the nodes and their weights randomly with a pre-defined probability chance on each epoch(?)here’s an abstract demonstration of this technique



Also we’re going to use regularization techniques with the formula below which modifies the loss function before applying it in the backpropagation process the linear sum is then added to loss function



And there’s the examples of using regularization L1(0.001) and L2(0.001) we’d have though its really odd and I don’t really why we get this chart !!!



Final summery :

Data summery :

Not really much to say here except that German women in the age between 50 and 60 years old who have 200k to 250k salary are veeery likely to quit the bank and French males in between 20 and 30 years old who have 0 to 25k salary could prove to be a faithful customer :D

NueralNetwork summery :

As I observed the different variations of the network in this project, generally speaking more simple layer structures work better for this kind of simple and (not big) Data and also have less chance of overfitting

As about the activation functions, SoftMax worked so much better than other activations WITH “ADAM” optimizer, and so I used those two together

And as about the batch sizes, having the batch size of 4 improved the network loss function by a very slight amount when we were using batch size of 1 and also much better than other batch sizes like 8, 16, 32 and so on

And about the learning rates using a rate in between 0.001 to 0.01 worked better and had a more consequent loss improvement and generally better in respect of more epochs (like 100 or more epochs) because it could give us a loss rate of less than 2 percent in 1000th epochs to so ! (Although leading to overfitting)

And at last I observed that having an overfitted network(and the using dropout) could lead to a more precise network than just preventing the overfitting with being careful of epochs and layer-complexity