**Hackmasterz Concept Note Problem 1**

**Chosen Columns:** { 'job\_description', 'marital\_status', 'education\_details', 'has\_default',

'balance\_in\_account', 'housing\_status', 'previous\_loan', 'date',

'call\_duration', 'outcome'}

**We have removed the following columns:**

**'serial\_number' – The serial number is dependent on the entries in test.csv**

**'age\_in\_years' – The tendency to give a debit card may not depend on age because there might be young entrepreneurs**

**'phone\_type' – This is another unaffectied column**

**'month\_of\_year' – This rarely depnds on month**

**'campaign\_contacts', 'previous\_contact', 'poutcome\_of\_campaign' – The previous number of participants has no effect on the present campaign**

**'days\_passed'**

For choosing remaining columns I took the **coefficient of correlation** between the columns are dropped those which are closer to zero showing least attachment to outcome column.

I converted the non-numeric data into numeric data with a function which takes all the unique string values in the dataframe and creates a dict giving them a unique index.

Them I remove the rows which have more than two unknown variables

Then I preprocess the data from the functions in scikit learn.

The model I have used is Knearest Neighbours Classification Algorithm with 6 neighbors and n\_jobs as 1 for multiprocessing, tolerance as 0.001.

The accuracy was ranging from 95.1percent to 94.5percent with 80:20 Data Split into Train and Test

I predicted the data from test.csv and found that 69 entries having a Yes

When implemented using Deep learning, the accuracy was stuck at 88.2 percent with all the predictions zero.

When the training data was preprocessed by Quantile Transformer the data was to clustered, so the accuracy was again stuck at 88.2 percent with all zeros

When choosing the neighbors, some columns were still lacking proper correlation, having more outlier neighbors (was sometimes caused by shuffling of test\_train data) which was indicated by fluctuating accuracy.

I found that the optimum number of neighbors is at 10 where I achieved 88.67 accuracy

The training\_data was scaled with zero mean and unit variance.

Unscaled data was creating more spread data decreasing the accuracy by 2 percent.

The snapshots are attached in the zip

Implemented using Google Colab