Understanding of data:

- The training data consists of 1, 01,180 training samples where each sample consist of 22 features set. Here each sample represent a person and features (like demographics and past activities) corresponding to him/her.
- The target variable is 'C' which of type binary (0 or 1). Here target represent whether the person has brought a product 'Y' (1) or not (0).
- If we look at the data, it is widely distributed. Almost all the features have different units. So Standardization of a dataset is required in this case.
- Also the dataset is imbalanced. For example, we have a 2-class (binary) classification problem with 1, 01,180 instances (rows). A total of 76,353 instances are labeled with Class-0 and the remaining 24,827 instances are labeled with Class-1.
- This is an imbalanced dataset and the ratio of Class-0 to Class-1 instances is 76353:24827 or approximately 3:1.

Pre-processing techniques used:

- There are two feature columns which has data type date. We cannot give this feature directly to the model as it will output error. So I have converted the date to an ordinal i.e. an integer representing the number of days since year 1 day 1. We can do this by a datetime.date's toordinal function.
- As the dataset is imbalanced I have used Up-sampling technique which randomly duplicates observations from the minority class (1) in order to make the observations from both classes qual. The new Data-Frame will have more observations than the original, and the ratio of the two classes will be 1:1.
- I have used StandardScaler function from sklearn.preprocessing to standardize features by removing the mean and scaling to unit variance from entire dataset.
- I have dropped 'Index' column from Data-Frame as it is not needed for training.
- Then I have separated input features(X) and target variable(Y).

Machine Learning model:

- I have used Neural Network for this binary classification problem.
- Neural Network characteristics:
    1. Input layer: Input_dim is set to 22 because each of our input samples has 22 features. So we have 22 input neurons at input layer.
    2. Hidden layer: Next is a Dense layer which is fully connected layer with 1024 neurons.
    3. Dropout layer: Dropout layer which disables fraction of inputs to reduce over fitting. Here I have set dropout to 5%.

4. Activation function: I have used ‘relu’ as a activation function for hidden layer and ‘sigmoid’ at the output layer.
5. Output layer: Output layer is again a dense layer with single neuron as we are dealing with binary classification which has only one output 0 or 1.
6. Model optimizer: It is the search technique used to update weights in the model. I have used RMSprop which is an adaptive learning rate optimization method.
7. Model Loss Function: ‘binary_crossentropy’ as it is binary classification problem.
8. Model training: The model is trained on NumPy arrays using the fit() function.

Evaluation of model:

- Data splitting: Here we separate portion of data into validation dataset and evaluate the performance of model on that validation dataset each epoch. Here I have split 20% data into validation dataset.
- Confusion matrix: It is a table which is used to describe the performance of classification model on set of test data.

```
In [48]: confusion_matrix
    ...:
Out[48]:
array([[13531,  1812],
       [ 2128, 13071]])
```

The classifier made a total of 30542 predictions. Out of those 30542 cases, the classifier predicted "1" 14883 times, and "0" 15659 times. In reality, 15199 persons belong to class “1”, and 15342 to class “0”.

- Classification report:

```
    ...: print(classification_report(y_test, y_pred))
30528/30542 [============================>.] - ETA: 0s    - ETA: 53s
precision    recall  f1-score   support

          0       0.86      0.88      0.87     15343
          1       0.88      0.86      0.87     15199

avg / total       0.87      0.87      0.87     30542
```