Approach for the Problem

Dataset:

The dataset is given in a comma separated value format. The *Train* and *Test* are given separately. There were 11 feature column given in the dataset. We have to predict ***Is_Lead*** feature column for the test dataset which represents whether the Customer is interested for the Credit Card or not. The ***ID*** column is for customer and is unique for each one, so it will not contribute in predictions and we can drop that feature.

ID	Gender	Age	Region_Code	Occupation	Channel_Code	Vintage	Credit_Product	Avg_Account_Balance	Is_Active	Is_Lead
NNVBBKZB	Female	73	RG268	Other	X3	43	No	1045696	No	0.0
IDD62UNG	Female	30	RG277	Salaried	X1	32	No	581988	No	0.0
HD3DSEMC	Female	56	RG268	Self_Employed	X3	26	No	1484315	Yes	0.0

Data Preparation:

Starting with combining both the train and test dataset and create one *DataFrame* and do all the operations and later split it into it original distribution. We can see that few feature column has categorical values and string type values in them. Machine Learning algorithms doesn't understand strings or categorical or any other data types, it only understands numbers.

So firstly, we will use ***One-Hot-Encoding*** from Scikit-Learn library for columns ***Gender, Region_Code, Occupation, Channel_Code, Is_active*** and we will use ***Label-Encoding*** from pandas library for columns ***Age, Vintage, Avg_Balance.***

We choose different encoding techniques for different columns because for the categorical feature which are not ordinal, we will apply *One-Hot-Encoding* and for ordinal feature columns we will use *Label-Encoding*.

For the columns where *Label-Encoding* is applied, I group them into smaller category e.g. for *Age* column, the min value is 23 and max value 85, I converted them into 4 groups. 1st from 23-34 age, 2nd from 35-45, 3rd from 46-60 and 4th rest till 85 and label them 0 to 3 respectively.

Gender_Female	Gender_Male
1	0
1	0
1	0

One-Hot-Encoding

Age	Vintage
3	1
0	0
2	0

Label-Encoding

Missing Data Handling:

The only feature column with missing data is ***Credit_Product.*** This data tells whether the customer has already an active loan or not.

```
df.isnull().sum()
```

ID	0
Gender	0
Age	0
Region_Code	0
Occupation	0
Channel_Code	0
Vintage	0
Credit_Product	41847
Avg_Account_Balance	0
Is_Active	0

This column has *41847* null values in train and test combined. As data is most precious in data science, we can not drop the null values. I used a *classifier algorithm* for predicting the missing values. I used ***xgboost XGBRegressor()*** for this predictions. I created the *Credit_Product* rows as test dataset and remaining as the train dataset. I set the *Credit_Product* column as target column and rest as feature columns.
After predicting the missing values, replaced that missing values with the predicted values from the classifier predictions.
After that we will split the data into it's original train and test set.

Model Training:

After preparing the data, it's time to choose an algorithm for predictions. I will first split the train dataset into train and test for validating the model accuracy with the ***sklearn's*** method ***train_set_split()*** and set the *test_size* as *30%* of the total train dataset.

Feature Selection:

For feature selection I will use *sklearn's* ***SelectKBest*** method which will give me score for a specific feature. More the score of the feature, more the predictions depends on it. I will select top 14 features from the *SelectKBest* list and use those feature for predictions.