CREDIT CARD LEAD PREDICTION

(Analytics Vidhya Jobathon)

Approach report

NOTE: Please check .ipynb for a detailed Analysis of data.

Here I am defining my approach for solving this problem in steps

Step 1: First I read the problem statement carefully 2 times.

Step 2: Then read about the dataset and about each column/feature

Step 3: Collect the data from the Analytics Vidhya Jobathon page and load it into memory with the help of pandas.

Step 4: Data Analysis

After loading the dataset I did data analysis like:

- How dataset looks like with (*data.head()*)
- What is the shape of the dataset (how many rows it has and how many columns it has)

- Check for duplicates in it then I found 21 and I dropped them.
- To check datatypes of each column and check how many non-null values have each column (After applying data.info() I found that in "*Credit_Product*" columns we have *29325* missing values)
- After that, I plot some plots to get insights and to know which columns are more important for classification, for each column one by one with respect to the "*Is_Lead*" column with this analysis I found
    1. *Vintage*
    2. *X2, X3 channel codes*
    3. *Credit product*
    4. *All types of occupation*
    5. *4 region codes*
    6. *And Average Account balance*

Now we have to find the most important features to get maximum accuracy for our model.

Step 5: To achieve this task I used selectkbest with ***Chi-square*** from sklearn's feature_selection module because we have categorical values here.
But before apply this method we need to do ***feature engineering*** because we have some features in numeric form and some are in categorical form.

Step 6: Missing Value Handling
(And here we also need to take care of missing values)
We have various techniques to deal with the missing values some of them are given below.
1. Drop the rows which have missing values.
2. Drop the columns which contain missing values.
3. Replace missing values with the most frequent ones.
4. Make a model to predict missing values.
5. Use clustering to fill missing values.
6. Make a new category for missing values.

"*Credit_Product"* have some missing values
In this category, we have 2 values "Yes" and "No"
Here I try 2 methods one is

1. *"Put value which occurs most of the time"*

2. *“Make a new category with missing values”*

With the first method, I get low accuracy then I apply the second one.

Step 7: Feature Engineering

1. **One-hot Encoding**
    - Gender (map{Female=0,Male=1})
    - Occupation
    - Channel code
    - Credit product
    - Is Active (map{No=0,Yes=1})
    - Region Code

For all these above columns I apply one-hot encoding

NOTE: To get rid of the Dummy variable trap I drop Columns “X4”, “Other occupation”, “nan credit product”,

Step 8: Apply SelectKBest with Chi-Square and I pick the best 10 (without considering Region code) features.

Step 9: Split data into 80:20 ratio to test models locally

Step 10: Modeling

- Here I apply various algorithms
    - Logistic regression
    - KNN
    - Decision Tree
    - Random Forest
    - XGBOOST

After apply XGBoost i got 85.02% accuracy then I submit my solution to the portal and I got 73.9% (I submit actual values 0 or 1).
Then I read the problem statement again then I submit probabilities and got 86.80%.

Step 11: Hyperparameter tunning

- Then I did some Hyperparameter tuning but the score does not go up and taking too much time.

Step 12: Search for a better solution
Now I search on the internet how we can improve accuracy than I found CatboostClasssifier Algorithms and LGBMClassifier.

Then I train the model with both of them and I got 85.29% with LGBMClassifer and my score on the portal is 87.28%.

Step 13: My own Second Approach for missing values
- Now I try my own method to deal with missing values
  Methods:
    - First count all null values
    - Then check in which ratio train data have credit product for Yes and No then I found (No-2:yes-1) ratio approx.
    - Then I the same ratio I create a list with yes and no values and shuffled it.
    - Then add this list to the missing values.

  Outcomes: Accuracy was decreased to 79.80%

Step 14: Again Feature Selection
- This time I select the top 20 features with Region code (Having One-hot Encoding)
- Apply LGBM then I got 87.30%

NOTE: I try lots of techniques and approaches in between which are not mentioned here. That will take more to write down.