Machine Learning Method Project

Loyalty of a bank client

Thomas Le Page

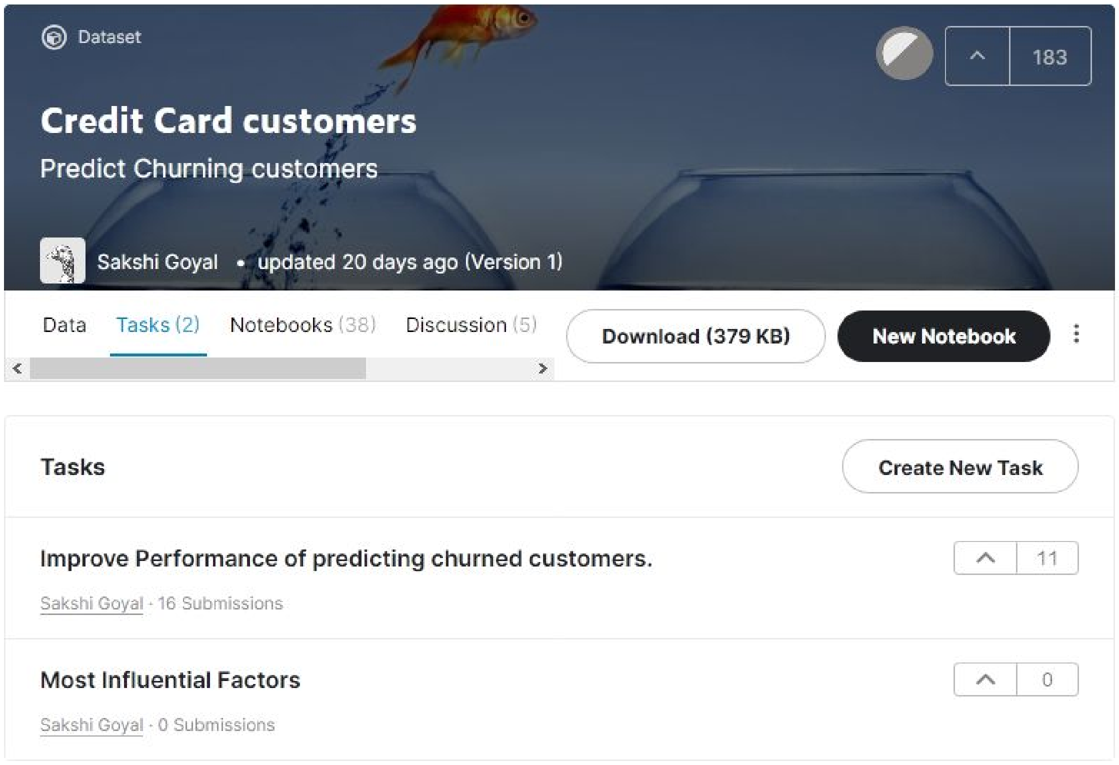
Autumn 2020

1. Problem description

Marketing is trying to get more and more clients with special offers. Their goal is to target the group which would be the most loyal when coming to their company. The financial sector is starting to change with the growing up of technology with some companies like N26 and Revolut for banks. A suitable dataset was released on Kaggle to define the factors which influence loyalty. The project would follow 2 axes :

  a. Is it possible to anticipate the loyalty to his bank ?
  b. Which group has the best loyalty ?

Those axes also refer to the tasks of the Kaggle topic :


Dataset

Credit Card customers

Predict Churning customers

Sakshi Goyal • updated 20 days ago (Version 1)

Tasks

Improve Performance of predicting churned customers. 11

Sakshi Goyal · 16 Submissions

Most Influential Factors 0

Sakshi Goyal · 0 Submissions

The topic is available at :
https://www.kaggle.com/sakshigoyal7/credit-card-customers

2. Dataset

The dataset contains more than 10 000 rows, which are unique clients, and 23 features such as id, age, educational level, marital status, attrition flag … All the variables have valid values. This paper goal is to define when a client is going to leave the bank, this will be defined by the *attrition flag* (1 : account is closed | 0 : account still opened) and the duration of the relationship by the *total relationship count*.
Some features might be not used, and referred to the topic, some are depreciated.

3. Methodology

Through supervised learning, I will teach the computer to anticipate the duration of the relationship between a client and the bank by using the other features available. I will firstly need to clean the dataset. The dataset will be divided into 2 parts randomly : the training set and the test set. I will use the knn method, a clustering algorithm, and the random forest classifier method, to define the most relevant features. These relevant features goals are to enhance the knn model by deleting the irrelevant ones. The error will be calculated with the MAE of the model used.

4. Realisation
    a. Preparation

Firstly, I separated the dataset in two : the features set and the flag set given by the column 'Attrition_Flag'. Then, I separated the two datasets in two : the training set and the test set. The splitting has been done in the proportion 0.8 for training and 0.2 for testing because the whole dataset is huge enough.

```python
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

FLAG_COL = 'Attrition_Flag'

x = data.drop([FLAG_COL], axis=1)
y = data[FLAG_COL]
```

```python
# Split the dataset into
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.8, test_size=0.2, random_state=3)
```

Secondly, I cleaned and prepared the data for the model. I decided to use a LabelEncoder for the flag set to keep a 1 column set. For the features set, I used a SimpleImputer for numerical columns, and a SimpleImputer and OneHotEncoder for objets columns. The preparation of the features set was hidden in a pipeline to enhance the performance of the code.

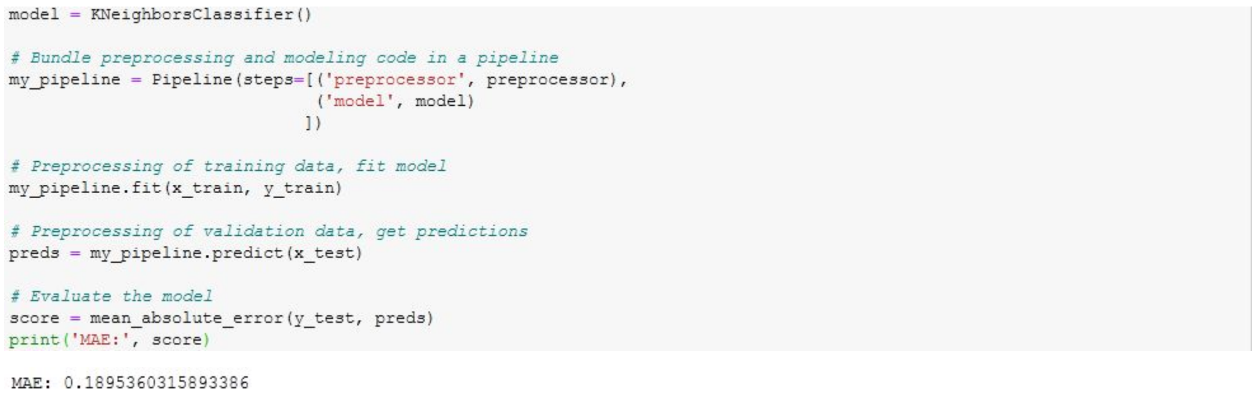
```python
# Transform the DataFrame y with a LabelEncoder ('Existing Customer' -> 1 ; Attrited Customer -> 0)
flag_transformer = LabelEncoder()
y = flag_transformer.fit_transform(y)
```

```python
# the dataset has been posted as cleaned, but I still make a SimpleImputer
numerical_transformer = SimpleImputer(strategy='constant')

categorical_transformer = Pipeline(steps=[
    ('imputer', SimpleImputer(strategy='most_frequent')),
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
    ])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numerical_transformer, numerical_cols),
        ('cat', categorical_transformer, categorical_cols)
    ])
```

I need to notice that I needed to drop the 2 last columns which result from a research on the subject. The KNN MAE score of the 2 features only gave 0.

b. KNN

The KNN method has been modeling in the pipeline, with the data preparation as shown. The error used was given by the MAE score, and in this case the score given was : 0.1895360315893386.

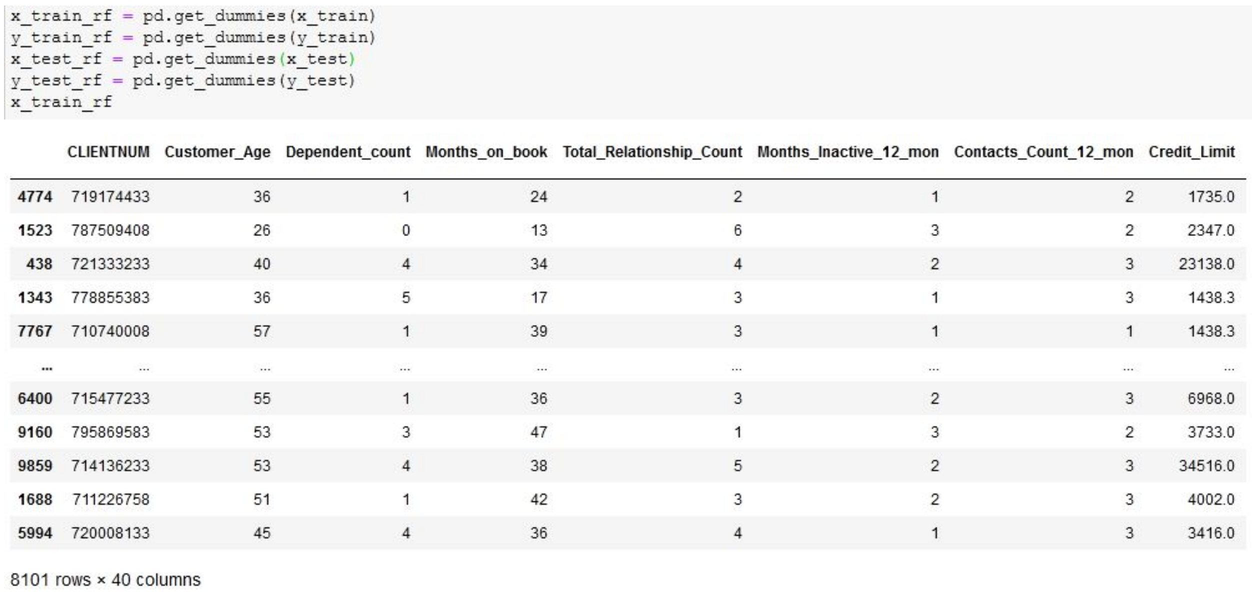


```
model = KNeighborsClassifier()

# Bundle preprocessing and modeling code in a pipeline
my_pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                              ('model', model)
                             ])

# Preprocessing of training data, fit model
my_pipeline.fit(x_train, y_train)

# Preprocessing of validation data, get predictions
preds = my_pipeline.predict(x_test)

# Evaluate the model
score = mean_absolute_error(y_test, preds)
print('MAE:', score)
```

```
MAE: 0.1895360315893386
```

c. Random Forest Classifier

The random forest classifier method also needed a preparation to be modeling. To simplify the code, the preparation was made with the function ‘pandas.get_dummies’.



```
x_train_rf = pd.get_dummies(x_train)
y_train_rf = pd.get_dummies(y_train)
x_test_rf = pd.get_dummies(x_test)
y_test_rf = pd.get_dummies(y_test)
x_train_rf
```

	CLIENTNUM	Customer_Age	Dependent_count	Months_on_book	Total_Relationship_Count	Months_Inactive_12_mon	Contacts_Count_12_mon	Credit_Limit
4774	719174433	36	1	24	2	1	2	1735.0
1523	787509408	26	0	13	6	3	2	2347.0
438	721333233	40	4	34	4	2	3	23138.0
1343	778855383	36	5	17	3	1	3	1438.3
7767	710740008	57	1	39	3	1	1	1438.3
...	...	...	...	...	...	...	...	...
6400	715477233	55	1	36	3	2	3	6968.0
9160	795869583	53	3	47	1	3	2	3733.0
9859	714136233	53	4	38	5	2	3	34516.0
1688	711226758	51	1	42	3	2	3	4002.0
5994	720008133	45	4	36	4	1	3	3416.0

8101 rows × 40 columns

Then, I could use the model to define the most relevant features in my case with the function ‘feature_importance’.

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=10)

rf.fit(x_train_rf, y_train_rf)
```

```
feature_importances = pd.DataFrame(rf.feature_importances_, index= x_train_rf.columns, columns=['importance']).sort_values

feature_importances
```

	importance
Total_Trans_Ct	0.192343
Total_Trans_Amt	0.176785
Total_Revolving_Bal	0.107084
Total_Ct_Chng_Q4_Q1	0.081799
Avg_Utilization_Ratio	0.071144
Total_Amt_Chng_Q4_Q1	0.065202
Total_Relationship_Count	0.044932
Customer_Age	0.032664
Avg_Open_To_Buy	0.029064
Credit_Limit	0.027881
CLIENTNUM	0.024851

The first feature has an importance of 0.19, the second of 0.18 and the third of 0.11 which define that they have a big weight on the model (48% of the importance). We could suppose that we could only use these three features to define our KNN model.

d. KNN optimisation

With the given features from the part above, we will try to define a sufficient model. So, I isolate these features for the new KNN model.

```
FEATURE_1 = 'Total_Trans_Amt'
FEATURE_2 = 'Total_Trans_Ct'
FEATURE_3 = 'Total_Ct_Chng_Q4_Q1'

x_train_2 = x_train[[FEATURE_1,FEATURE_2, FEATURE_3]]
y_train_2 = y_train
x_test_2 = x_test[[FEATURE_1,FEATURE_2, FEATURE_3]]
y_test_2 = y_test
x_train_2
```

Then, I repeated the same KNN method as above with the new features. The MAE score resulting from the model is : 0.10562685093780849.

```
MAE: 0.10562685093780849
```

5. Conclusion

The model given by the KNN method is relevant since the MAE is relatively low. With the Random Forest Classifier method I could enhance the model with only 3 features which could be more relevant to determine easily if a client would stay or not. The MAE went from 0.19 to 0.10 ! The final model answers the question in the way that with a given client it could anticipate his ‘loyalty’ and so on for the whole clients of the bank.