Final Assignment - C01 - IBM ml professional certificate

Brief description

We have found in Kaggle a data set on churn in the banking sector with multiple variables. Studying when a client is about to leave the services of a company is vital to anticipate actions to keep them as customers.

The data set can be found at **https://www.kaggle.com/sakshigoyal7/credit-card-customers?select=BankChurners.csv**, which is quite complete with **10127** different **rows**, each one representing a different client. Of the **23** existing **columns**, the author recommends to drop **Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_1** and **Naive_Bayes_Classifier_Attrition_Flag_Card_Category_Contacts_Count_12_mon_Dependent_count_Education_Level_Months_Inactive_12_mon_2** since they do not provide any useful information for the study. The rest are the following:

CLIENTNUM, unique identifier for the customer holding the account.
Attrition_Flag, if the account is closed then 1 else 0.
Customer_Age, Customer's Age in Years.
Gender, M if male and F if female.
Dependent_count, number of dependents.
Education_Level, educational qualification of the account holder
Marital_Status, married, single, divorced or unknown
Income_Category, annual income category of the account holder
Card_Category, type of card
Months_on_book, period of relationship with bank
Total_Relationship_Count, total no. of products held by the customer
Months_Inactive_12_mon, No. of months inactive in the last 12 months
Contacts_Count_12_mon, No. of contacts in the last 12 months
Credit_Limit, credit limit on the credit card
Total_Revolving_Bal, total revolving balance on the credit card
Avg_Open_To_Buy, open to buy credit Line (average of last 12 months)
Total_Amt_Chng_Q4_Q1, change in transaction amount (Q4 over Q1)
Total_Trans_Amt, total transaction amount (Last 12 months)
Total_Trans_Ct, total transaction count (Last 12 months)
Total_Ct_Chng_Q4_Q1, change in transaction Count (Q4 over Q1)
Avg_Utilization_Ratio, average card utilization ratio

Initial plan

The objective is to create a model that is capable of predicting when a client is about to leave the service or not. We can use a logistic regression classifier, which is the most common used in these cases, from the data set to this purpose. To carry it out successfully we must make sure that the data set that we are going to use is completely adequate and that it will not generate erroneous predictions. This means that we must ensure that the data set used is composed of numeric values, that it does not have outliers or missing values. **The plan is the following**:

1) Drop unnecessary columns
2) Look for missing values and deal with them if needed.
3) Look for outliers and deal with them if needed
4) Correct feature distributions when needed (skew)
5) Transform categorical to numeric variables
6) Scale variables
7) Find correlations

Actions taken

We identify the variable **Attrition_Flag** as the **target** variable since it is the one that contains the information about the client's continuity in the service.

As a first action we drop column **CLIENTNUM** since it does not give useful information to create the model. Using the **info()** method of the **pandas** dataframe we can see that there are **no null values** in the data set so we will not need to deal with missing values. We also see that there are some columns with type object, including the target variable. **These columns are categorical ones**.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10127 entries, 0 to 10126
Data columns (total 20 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Attrition_Flag            10127 non-null  object 
 1   Customer_Age              10127 non-null  int64  
 2   Gender                    10127 non-null  object 
 3   Dependent_count           10127 non-null  int64  
 4   Education_Level           10127 non-null  object 
 5   Marital_Status            10127 non-null  object 
 6   Income_Category           10127 non-null  object 
 7   Card_Category             10127 non-null  object 
 8   Months_on_book            10127 non-null  int64  
 9   Total_Relationship_Count  10127 non-null  int64  
 10  Months_Inactive_12_mon    10127 non-null  int64  
 11  Contacts_Count_12_mon     10127 non-null  int64  
 12  Credit_Limit              10127 non-null  float64
 13  Total_Revolving_Bal       10127 non-null  int64  
 14  Avg_Open_To_Buy           10127 non-null  float64
 15  Total_Amt_Chng_Q4_Q1      10127 non-null  float64
 16  Total_Trans_Amt           10127 non-null  int64  
 17  Total_Trans_Ct            10127 non-null  int64  
 18  Total_Ct_Chng_Q4_Q1       10127 non-null  float64
 19  Avg_Utilization_Ratio     10127 non-null  float64
dtypes: float64(5), int64(9), object(6)
memory usage: 1.5+ MB
```

Some of them only have two possible values then we can make a **binary encoding**, such as Attrition_Flag and Gender, other some are **ordinal categories** such as Educational_Level, Card_Category and Income_category so we can perform on them an ordinal encoding and for the Marital_status which has multiple possible values we can perform a **one hot encoding**. After performing the different encoding,

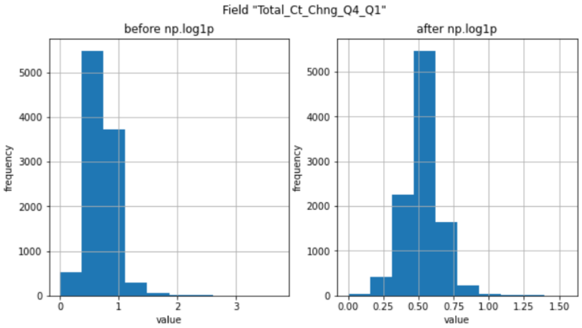
```
updated_01_df.info()
```

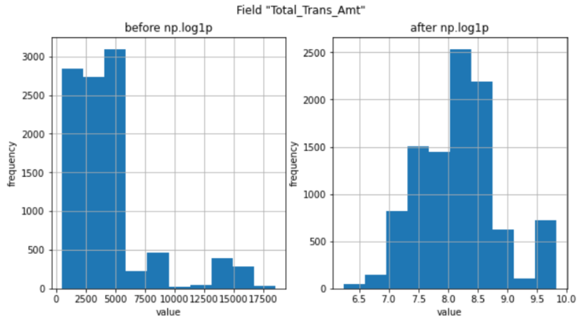
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10127 entries, 0 to 10126
Data columns (total 24 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Attrition_Flag            10127 non-null  int32  
 1   Customer_Age              10127 non-null  int64  
 2   Gender                    10127 non-null  int32  
 3   Dependent_count           10127 non-null  int64  
 4   Education_Level           10127 non-null  int64  
 5   Marital_Status            10127 non-null  object 
 6   Income_Category           10127 non-null  int64  
 7   Card_Category             10127 non-null  int64  
 8   Months_on_book            10127 non-null  int64  
 9   Total_Relationship_Count  10127 non-null  int64  
 10  Months_Inactive_12_mon    10127 non-null  int64  
 11  Contacts_Count_12_mon     10127 non-null  int64  
 12  Credit_Limit              10127 non-null  float64
 13  Total_Revolving_Bal       10127 non-null  int64  
 14  Avg_Open_To_Buy           10127 non-null  float64
 15  Total_Amt_Chng_Q4_Q1      10127 non-null  float64
 16  Total_Trans_Amt           10127 non-null  int64  
 17  Total_Trans_Ct            10127 non-null  int64  
 18  Total_Ct_Chng_Q4_Q1       10127 non-null  float64
 19  Avg_Utilization_Ratio     10127 non-null  float64
 20  Divorced                  10127 non-null  uint8  
 21  Married                   10127 non-null  uint8  
 22  Single                    10127 non-null  uint8  
 23  Unknown                   10127 non-null  uint8  
dtypes: float64(5), int32(2), int64(12), object(1), uint8(4)
memory usage: 1.5+ MB
```

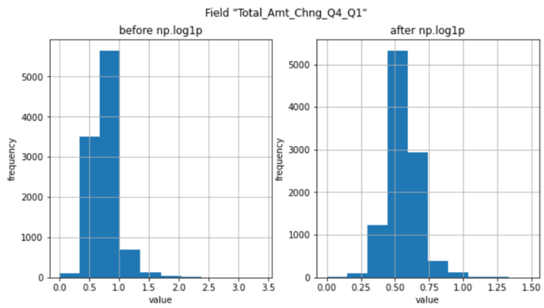
Once we have transformed the Marital_status into different new columns we can drop it.
Then, we continue studying the distribution of the numeric variables, the skew, to see if any transformation is needed. We can use the **skew() method** from pandas dataframe. We have selected a skew equal or greater than 0.75. We check by plotting the result before and after the transformation. The columns that need a look are,

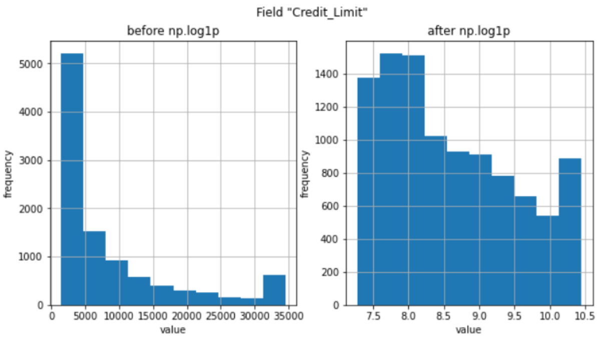
	Skew
Total_Ct_Chng_Q4_Q1	2.064031
Total_Trans_Amt	2.041003
Total_Amt_Chng_Q4_Q1	1.732063
Credit_Limit	1.666726
Avg_Open_To_Buy	1.661697

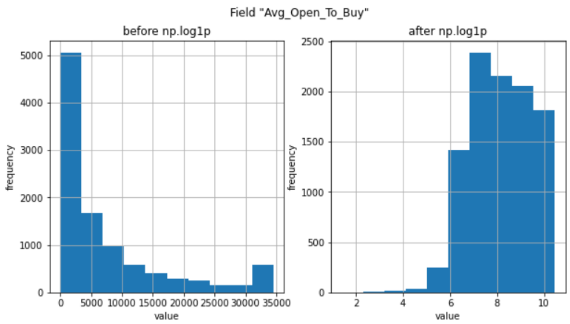
We plot the histogram for these variables before and after the transformation,









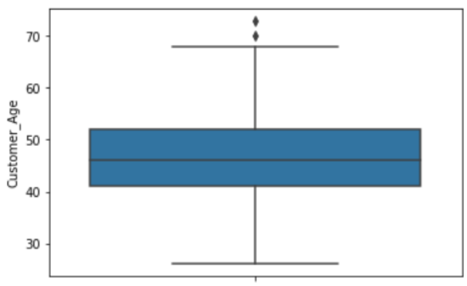


As we can see the logarithmic transformation **of Credit_Limit and Avg_Open_To_Buy does not solved any problem so we will not keep the transformation for these two columns**.

Using the boxplot we can study the outliers. Many of them seems to has ouliers such as for Customer_Age,

```
sns.boxplot(y="Customer_Age", data=df)
```

<AxesSubplot:ylabel='Customer_Age'>



But after having a look to the different values of each column **I diceded to keep these "outliers"** since seems to have normal values.

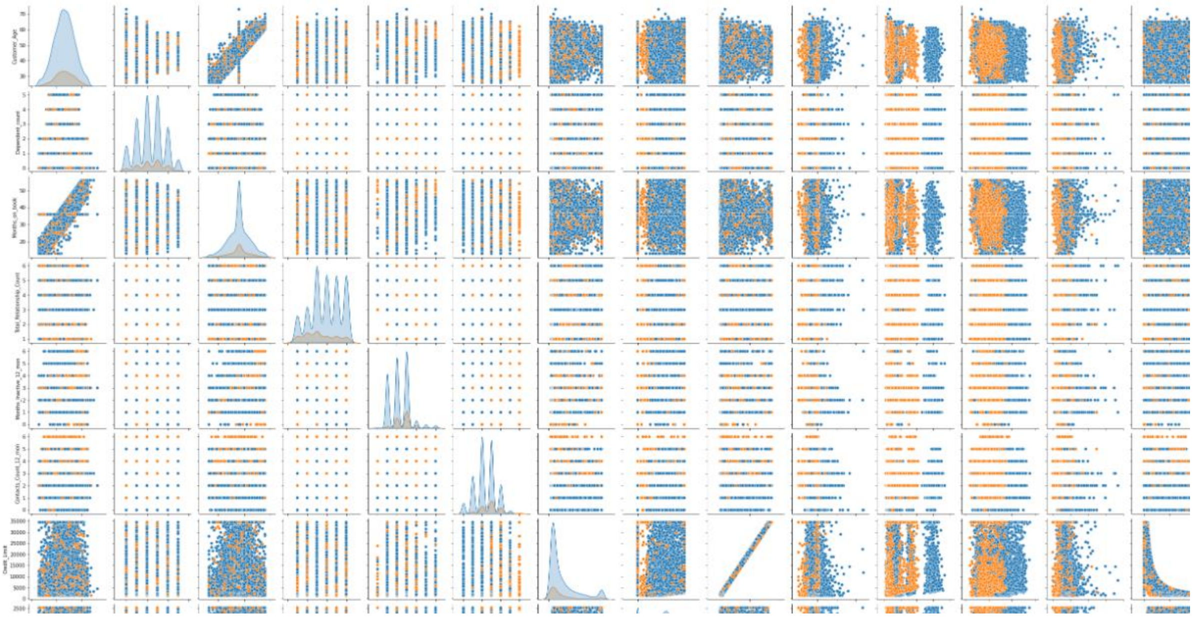
```
df['Customer_Age'].describe()
```

```
count    10127.000000
mean        46.325960
std          8.016814
min         26.000000
25%         41.000000
50%         46.000000
75%         52.000000
max         73.000000
Name: Customer_Age, dtype: float64
```

We can study the **correlation between the different variables by using a pairplot** from the seaborn library. In an single image we have all the needed information. **A detail of this plot is shown below**,

```
sns.pairplot(df, hue = 'Attrition_Flag')
```

```
<seaborn.axisgrid.PairGrid at 0x167d8208438>
```



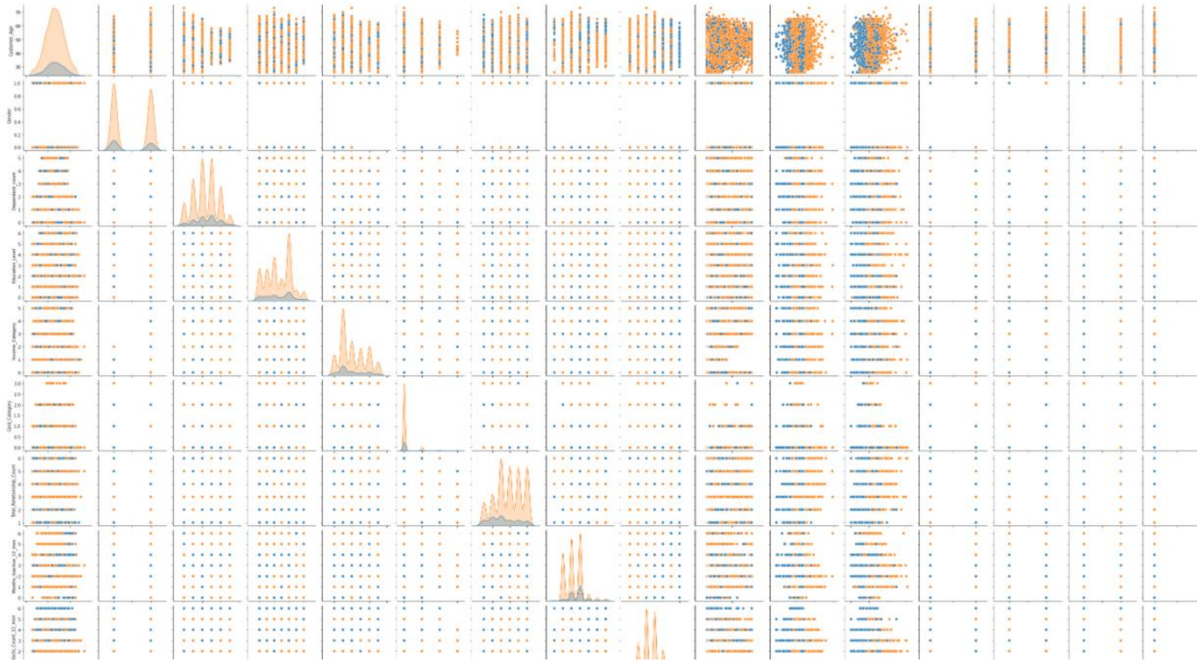
Looking into the detail we found some **correlations between these variables which are the following**,

Customer_Age is correlated with Months_on_book, which can be normal. Credit_Limit is correlated with Avg_Open_To_Buy and both are also correlated with Avg_Utilization_Ratio, also Avg_Utilization_Ratio is correlated with Total_Revolving_Bal . There is also a correlation between Total_Amt_Chng_Q4_Q1 and Total_Trans_Amt, and also between Total_Ct_Chng_Q4_Q1 and Total_Trans_Ct. So, we decided to **drop** the following columns, **Months_on_book**, **Avg_Open_To_Buy**, **Total_Trans_Ct**, **Total_Trans_Amt**, **Total_Revolving_Bal** and **Avg_Utilization_Ratio**.

After this action we have a look again on the pairplot

```
sns.pairplot(updated_02_df, hue = 'Attrition_Flag')
```

```
<seaborn.axisgrid.PairGrid at 0x167cc12c1d0>
```



The **last action that we take is the scaling ot the data set**. We use a **Min Max scaler since we are sure that we don't have any real outlier** on our data, so it will not create any problem whn applying the scaling. Thus, we will have **values between 0 and 1**.

```
updated_03_df.describe()
```

	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Income_Category	Card_Category	Total_Relationship_Count	Months_In
count	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	
mean	0.839340	0.432467	0.470919	0.469241	0.433659	0.417142	0.027879	0.562516	
std	0.367235	0.170571	0.499178	0.259782	0.283403	0.294928	0.111261	0.310882	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	0.319149	0.000000	0.200000	0.166667	0.200000	0.000000	0.400000	
50%	1.000000	0.425532	0.000000	0.400000	0.500000	0.400000	0.000000	0.600000	
75%	1.000000	0.553191	1.000000	0.600000	0.666667	0.600000	0.000000	0.800000	
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	

Key Finding

The data set was initially quite good, with no missing values and few outliers. Some categorical variables have been found that were transformed into numeric and some logarithmic transformations were carried out to correct the distribution of the variables. The vast majority of the variables are not correlated, then they are very useful to create the model. The correlated ones were drop, keeping only those that had a direct meaning for the problem.

Formulate 3 hypothesis

Hipotesis 1, it is seems that the customer that **leave the service are for equal distributed between fameles and males**. So if I pick one random customer from the ones that have left the service the possibility that I found ahead if it is a male or a female is in principle the same. Such in the coin tose example I can claim that **I can guess correclty the gender of 55%** of the cases. This is a big words but it could be possible. If I manga to do it, Can we said that I have an special intuition or just lbeing lucky?. The null hypothesis will be that I have an special intuition and **the alternative hypothis is that I don't have it**.

Hipothesis 2, again we can do a similar thing but instead of using the gender qe can do it by education level grouping first the education into two simple categories, high or low. Again the probability of leaving the service and being in one of these categories are equal.

Hypothesis 3, it is also similar but we will use the income_level. After grouping into High income and low income it seems that there is a different probability for the two categories. **Being the low income more promt to leave the service with a 60% of the cases**.

Conducting a significance test

We perfomr the test for hypothesis 1. In this case the customers that have left the service are 8500, where 4428 are females and 4072 males so we can say that both categories has the same probability of churn. If the null hypothesis is correct, the test statistic is binomial distributed with parameters n = 8500 and p = 0.5. That is, if we repeated the whole experiment many times, we would see such a distribution for all the results. The choice of a cutoff at 5% probability is common. That is, if we would only see data as extreme as we've seen less than 5% of the time, we'll say that seems too unlikely and we will conclude that we don't think the null hypothesis is true. In the case of the binomial distribution, which is discrete and not too complicated mathematically, we could just work out the probability. The 55% of the cases is 4675.

```
from scipy.stats import binom
prob = 1 - binom.cdf(4675, 8500, 0.5)

print(str(round(prob*100, 1))+"%")
```

```
0.0%
```

As we see the probability of guessing correctly the 55% is 0.0%, in other words imposssible. So if I claim that I did it only if I have an special intuition it is possible (or I'm cheating)

Unlike the example with the coin toss, now we have a large sample size. We see during the course that increasing the sample size was the best scenario to get close to the real distribution.

We can find at which number of picks can be considered as a lucky guess,

```
print(binom.ppf(0.95,8500,0.5)+1)
```

```
4327.0
```

```
prob = 1 - binom.cdf(4326, 8500, 0.5)
print(str(round(prob*100, 1))+"%")
```

```
4.9%
```

For number of correct gusses larger than 4326 we can consider that you need an special intuition.

Suggestion next steps

A next step prior to create the model it's to split the data set into two different sets, the train and the test ones. The train set we will use to train the model, and the second one to evaluate it.

As I suggest in the bigining, with this data set we want to create a model that can predict when a customer is about to leave the company. This is a binary classification problem and can be achieve with a logistic regression.

After the training the testing will said us how good is our model. If it is good enough we can deploy the model and start predicting on new data. If not good enough we can study if more or different data are needed and go back and start again the same process until we get a model that can solve our problem.

Summary

First we select the target variable, that is **Attrition_Flag**, so all the other columns are possible features that can be used to create the model. So, then we start to have a look on the data to clean it from columns that are not necessary or are correlated. We also transform the categorical columns into numeric to let the model used them properly. Also some of the columns did need to be transformed to correct the their distribution. After all this process we have a data set with **10127 rows (different customers)** and **16 features** that are useful to create the model.

Regarding the **significance test**, we find that for this **large amount of data** that to corecly guess more than 51% of the gender of random customer that have left the service is **almost impossible so only an special ability could lead to a beter guess.**