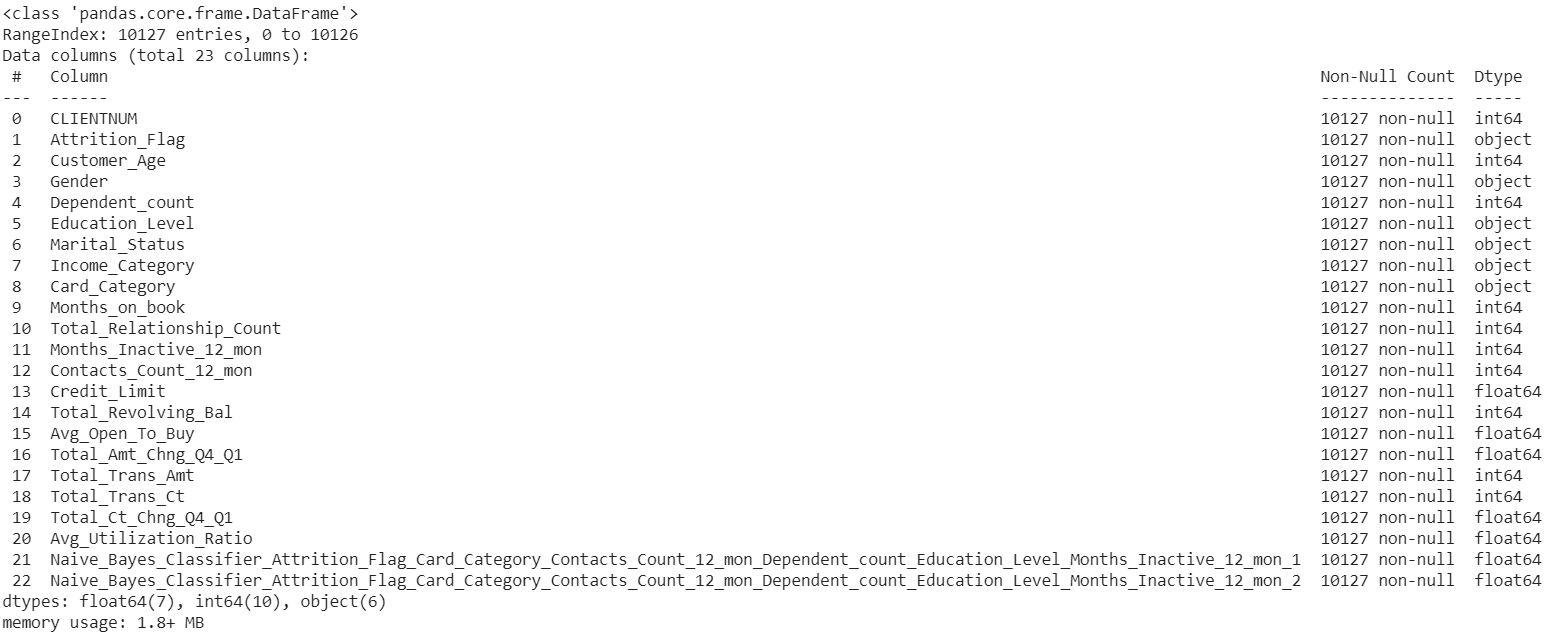
## Credit Card customers Documentation (Predict Churning customers)

1. **Motivation:**
   1. The reason for picking up this problem is to find out the reason for customers leaving credit card services.
   2. To avoid these scenarios we need to provide services based on customer needs and usage.
   3. It is important that credit card companies are able to recognize the pulse of customer usage in credit card services.
2. **Problem Statement:**
   1. A manager at the bank is disturbed with more and more customers leaving their credit card services.
   2. They would really appreciate if one could predict for them who is gonna get churned so they can proactively go to the customer to provide them better services and turn customers' decisions in the opposite direction.
   3. I got this dataset from a website with the URL as <https://leaps.analyttica.com/home>.
   4. Now, this dataset consists of 10,000 customers mentioning their age, salary, marital\_status, credit card limit, credit card category, etc. There are nearly 18 features.
   5. We have only 16.07% of customers who have churned. Thus, it's a bit difficult to train our model to predict churning customers.
3. **Proposed Approach:**
   1. This project **proposes** an idea to identify the reason behind the customer leaving credit card services.
   2. The most commonly techniques **used** methods are Logistic, Decision Tree, Random Forest, XGBoost and DNN.
4. **Database:**
   1. We have fetched the dataset from Kaggle.com as excel sheet.
   2. The link to the dataset is <https://leaps.analyttica.com/home>.
5. **Experiments:**
   1. After receiving the dataset, we have done the basic steps,
      1. Data Wrangling
      2. Data Cleaning & Pre-processing
      3. EDA
      4. Modeling

**Data Wrangling:**

* **Data preprocessing** is a **data** mining **technique** which is used to transform the raw **data** in a useful and efficient format.
* Before we start with anything, I want to see if I can drop any of the columns that are unnecessary. I will look at how some columns effected the important ones. For me some of the important columns are: Income, Credit Limit, Amount of change between Q1 and Q4 and Total transaction amount etc.



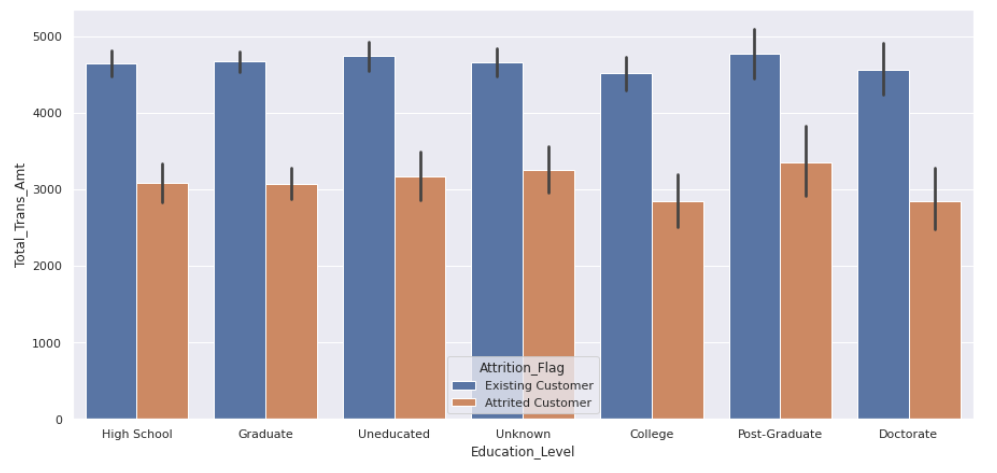
**As this dataset doesn't carry any missing values and null values, so data wrangling looks very easy here. Will proceed with the other findings in the upcoming steps.**

**EDA:**

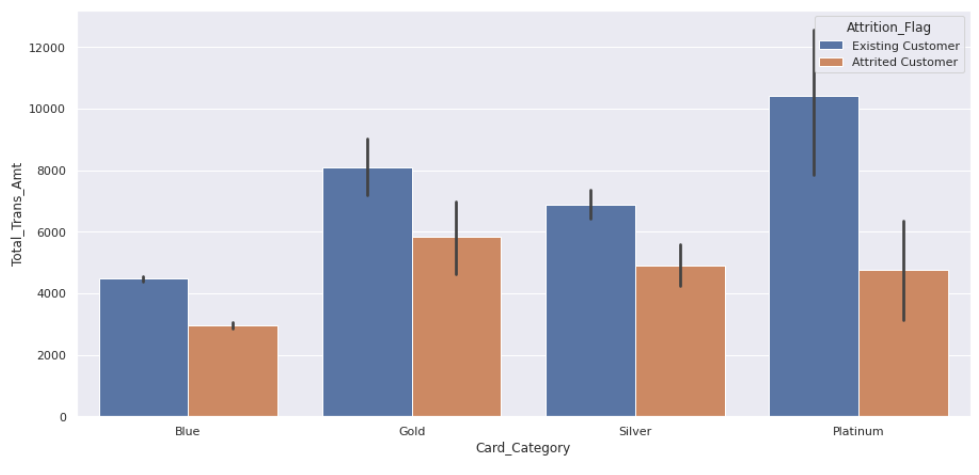
In EDA, the exploratory data analysis will be done,

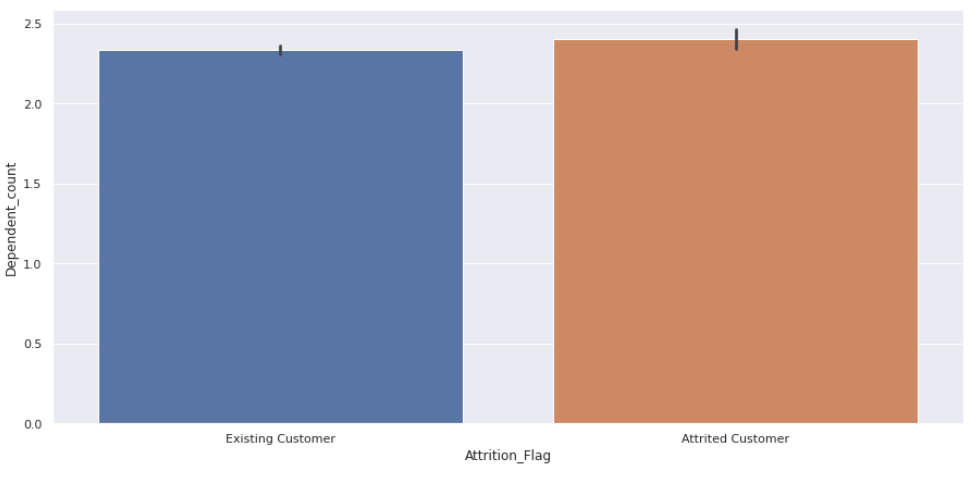
1. Categorical data visualization
2. Numerical data visualization
3. Pearson & Spearman correlation coefficient

Check the data structure of Dataset. There are 20 columns in this dataset. We need to know which data attributes are numerical and which data attributes are categorical.

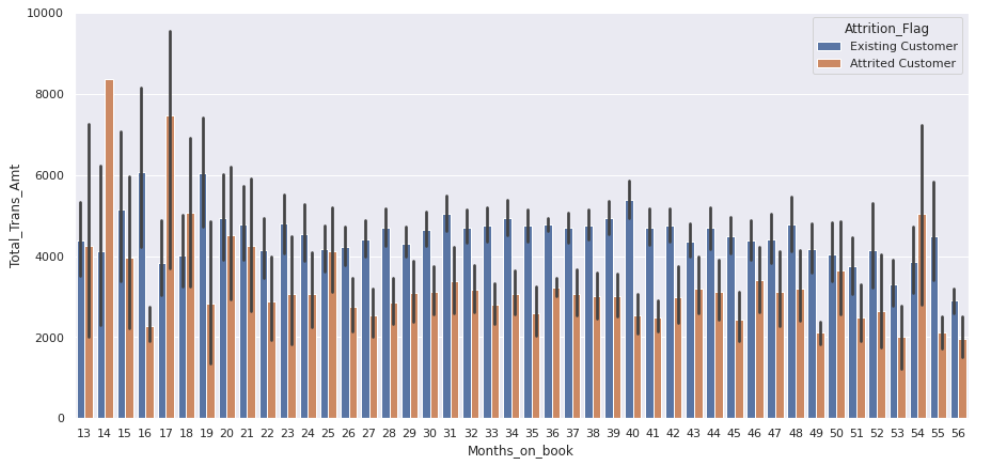


**Here we can see that Education Level did not effected Credit Limit and Attrition.**

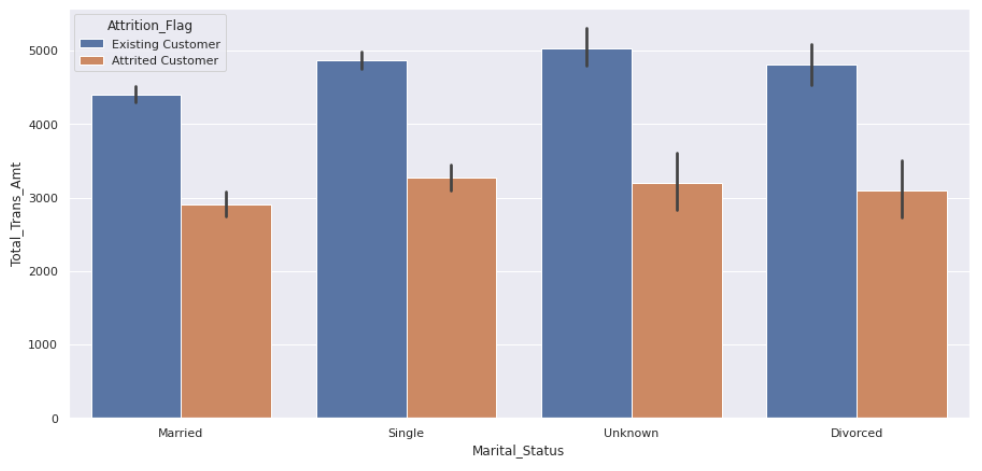


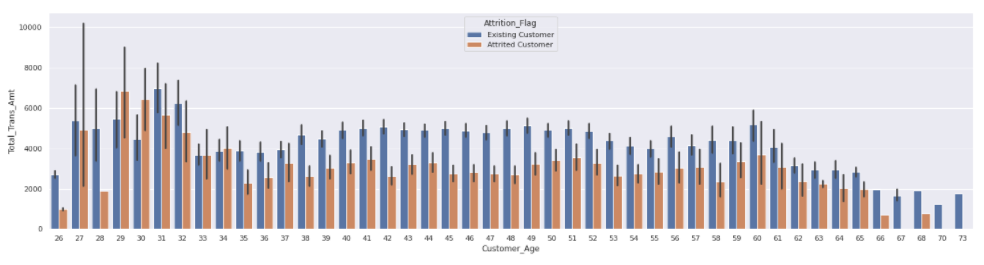


**There is a slight difference within the groups but since we have enough variables, dependent count is droppable.**



**We can say that newly members spend more money. But membership length does not affect the churn. I mean it is inconsistent within the length of the membership.**





**Data Cleaning & Pre-processing**

* So far we are dropping: Client number, Dependent count, Education level, Marital status and Month on book. Because they did not affected the attrition. In addition, obviously Client number is not important for us, either.
* Number of inactive months, total relationship count, contact count, revolving balance, amount of change between quarters, transaction amount&count and average utilization ratio are valuable information for this problem. The rest, we can drop.
* Every variable except Attrition Flag and Card Category are numeric. We need to turn non-numeric categories to numeric categories before the process.

**Handling Class Imbalance**

* Imbalanced data is a problem in supervised learning problems which can result is high bias towards majority class. As we have already seen that this data is severly imbalanced so to balance it we can use various techniques such as:

1. Oversampling
2. Undersampling
3. SMOTE

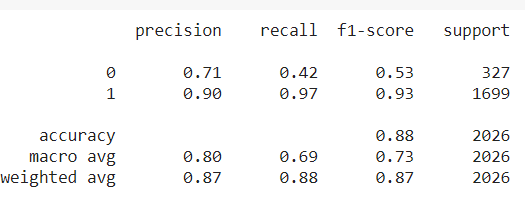
* Out of all these three SMOTE is the most effective so we will go with it, In this technique, instead of simply duplicating data from the minority class, we synthesize new data from the minority class. This is a type of data augmentation for tabular data can be very effective. This approach to synthesizing new data is called the Synthetic Minority Oversampling Technique, or SMOTE for short.
* We can clearly see that now the data is completely balanced so let's use some visualization technique to visualize this data.
* Note that, as our data has a lot of columns and humans can only understand 3D so we will use Dimensionality reduction technique to reduce our data to 3D and then plot it. So, let's get started!

**Modeling:**

* In this section, we will finally apply models

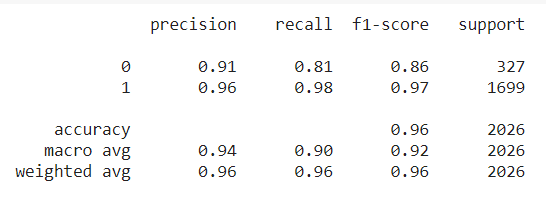
**Logistic Regression:**

* Let's start off with a simple model like Logistic Regression. Note that I will be doing cross validation using Randomized search as the data is very huge and we will do this cross validation after splitting to avoid Data Leakage as discussed above.

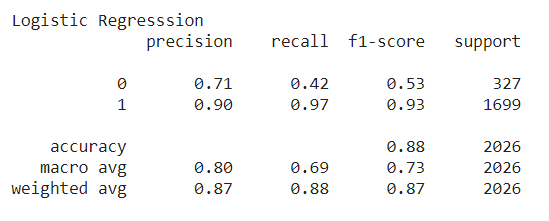


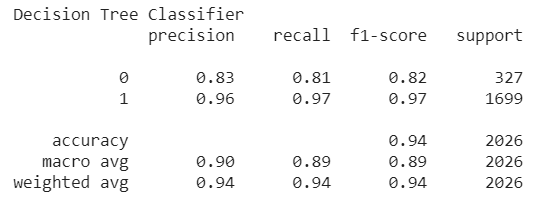
**Random Forest Classifier:**

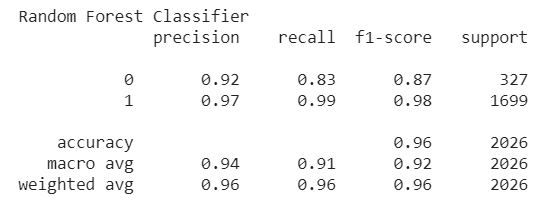
* Now, let's try something which can take account of complex relationships. There are many such models but Random forest is bit better as it is a ensemble model and focuses on reducing variance i.e overfitting without much effecting the bias which is all we want. Also, this algorithm works in time complexity, O(d.n.log(n)) where d is the number of features.
* I have shown the best parameters after GridsearchCV and not the whole process itself as it is very time consuming and takes forever so you can try it yourself.

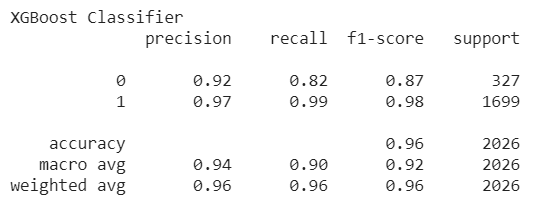


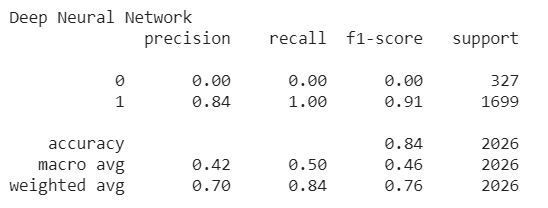
**Results of ML Algorithms Used:**











**Conclusion:**

* We had very successful results with Decision Tree, Random Forest and XGBOOST, up to **%96 accuracy**.
* I have used *regression, random forest, decision tree and neural network* approaches with this dataset.
* I was able to achieve *%96 accuracy* with xgboost which is pretty good.
* Overall, every model performed good but some of them were more suitable for this problem.

**Future Work:**

* Will try implementing some more models to improve the accuracy even more.