Name: Mayank Vilas Chonde

[Emaild = mayankchonde2002@gmail.com](mailto:Emaild=mayankchonde2002@gmail.com)

**DATA ANALYSIS WITH PYTHON:**

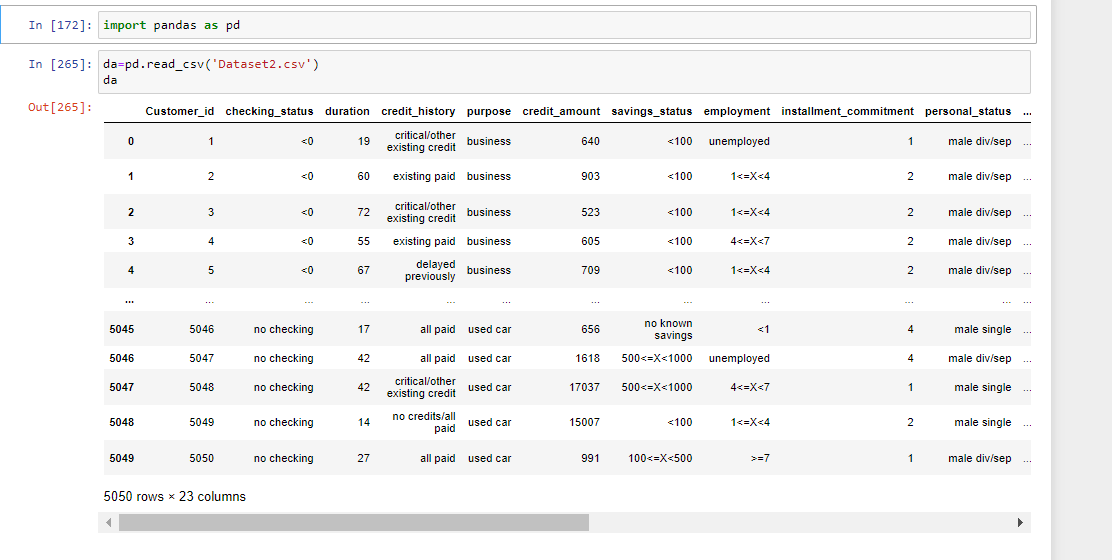
In the given data, we have a list of bank customers who have credit cards. In this model, we will fit a Random Forest Model and see whether a credit card customer is good or bad for a bank.

We have several features in the category and numeric data types. Also, we have an actual dependent variable with ‘good,’ which suggests that they are profitable credit card holders, and ‘bad’ suggests that they are not profitable for a bank.

**EXPLORATARY DATA ANALYSIS:**

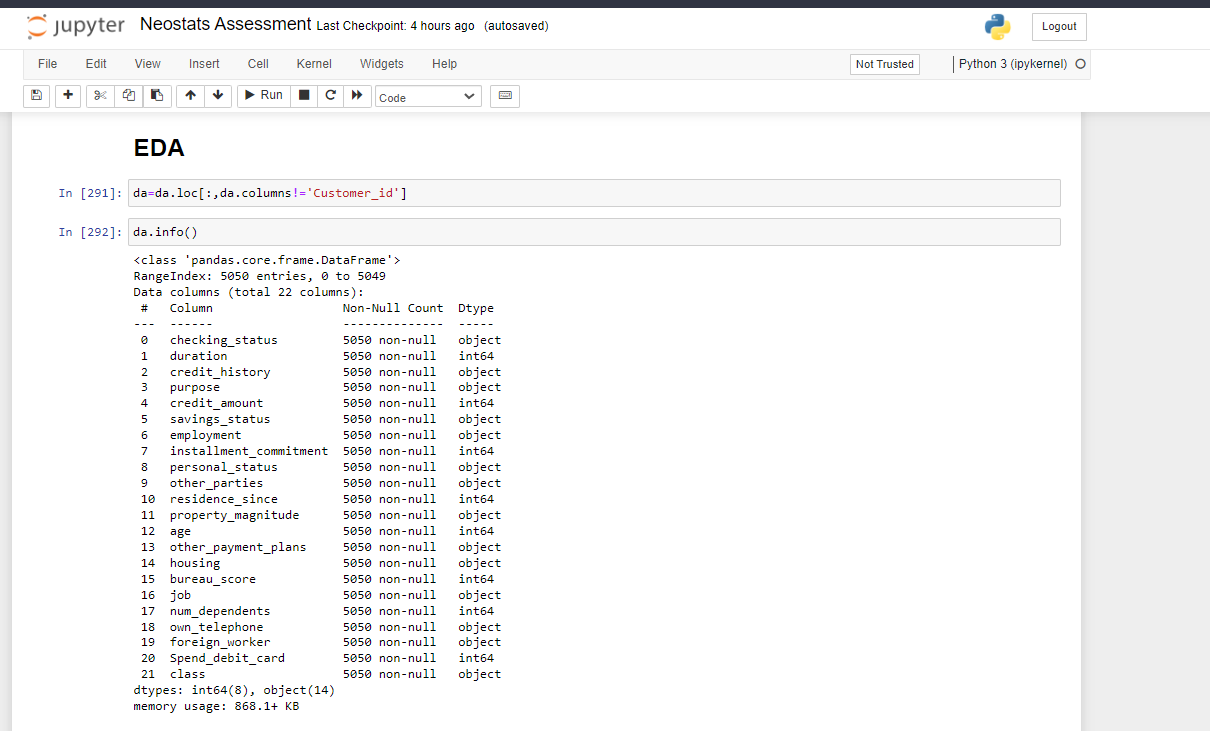
We build a model in Python using a Jupyter environment. Before building, we must analyze the data to check whether we have outliers or missing values.

**OUTPUT:**



We don’t need Customer\_id for further steps because the Customer\_id does not participate much in any model. So we drop the column Customer\_id, and by using info (), we can see what type of variables we have.

**OUTPUT:**

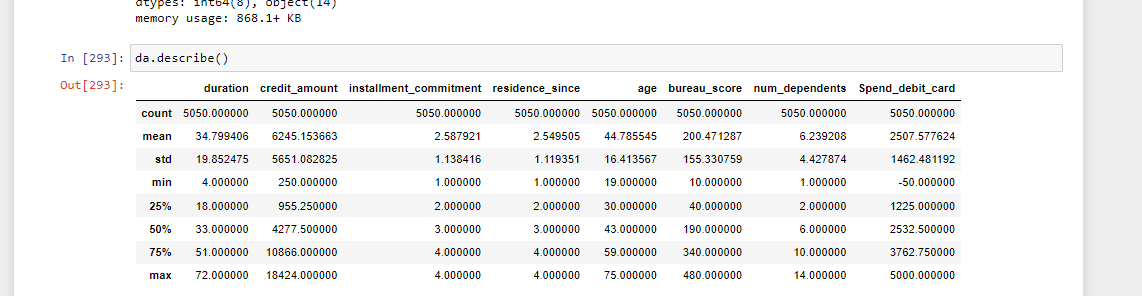


After getting the info output, we can say there are 22 features, from which 8 are integer type, 14 are objects, and 5050 entries in our dataset.

**EXAMINE EACH VARIABLE (EXCEPT CUSTOMER\_ID), SUMMARIZE, AND INTERPRET THE FINDINGS:**

After learning about the variable types, we see their summarization using describe () function.

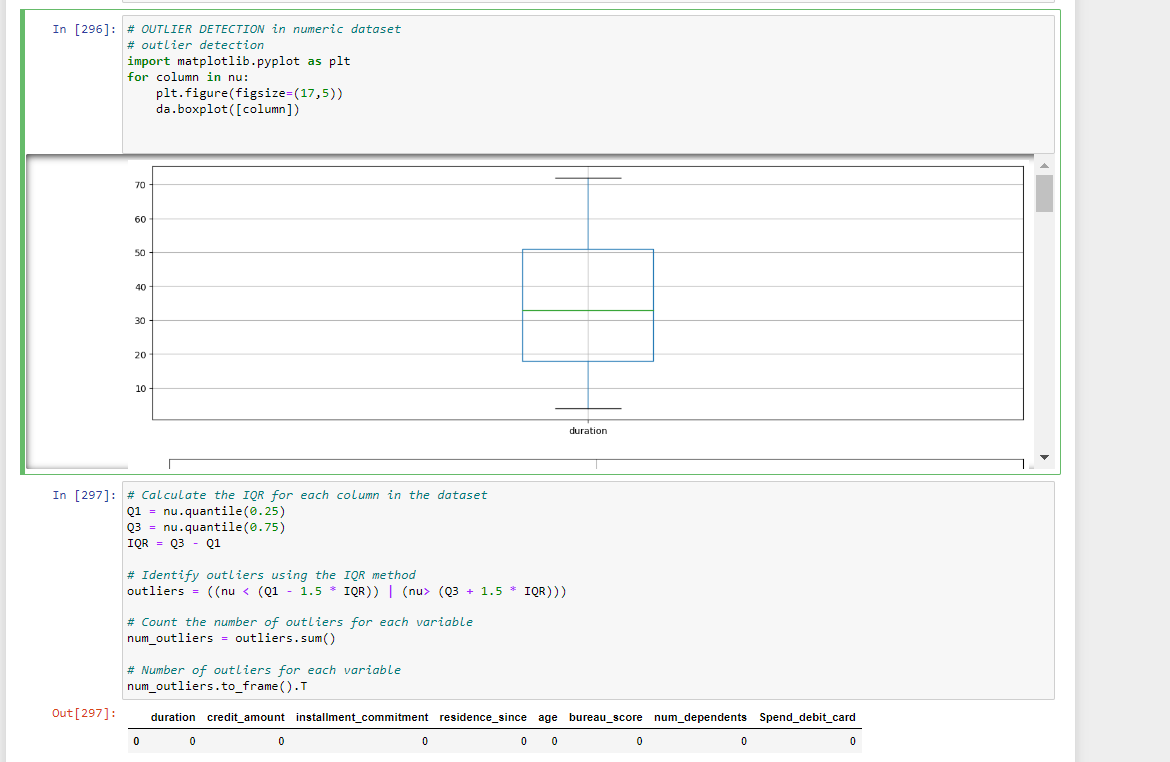
**OUTPUT:**



By getting variable description, we see that describe () only fetches numeric variables which describe their mean, the minimum value in the column, and maximum, the standard deviation of the column. After observing, we can see that in the column of Spend\_debit\_card, their maximum value is 5000, and the minimum value is -50, which may be an outlier we see in the next steps.

**OUTLIER DETECTION:**

**OUTPUT:**



To inform us about outliers in the data, we plot a boxplot where we get the minimum value, maximum value, mean value, and values concerning the quartile. Boxplot is used to see whether we have outliers or not. If the value of outside the box or away from the maximum line, we call that value outlier.

But, after implementing the boxplot and finding the IQR for every numeric variable. We don’t find any outliers in our dataset.

**MISSING VALUES:**

**OUTPUT:**



After getting to know the description of variables, it’s time to look at whether we have missing values in our dataset. To Find the NA values, we use the function null ().sum (), which gives the total NA values in each variable. After implementing this function, we know our dataset has no missing values.

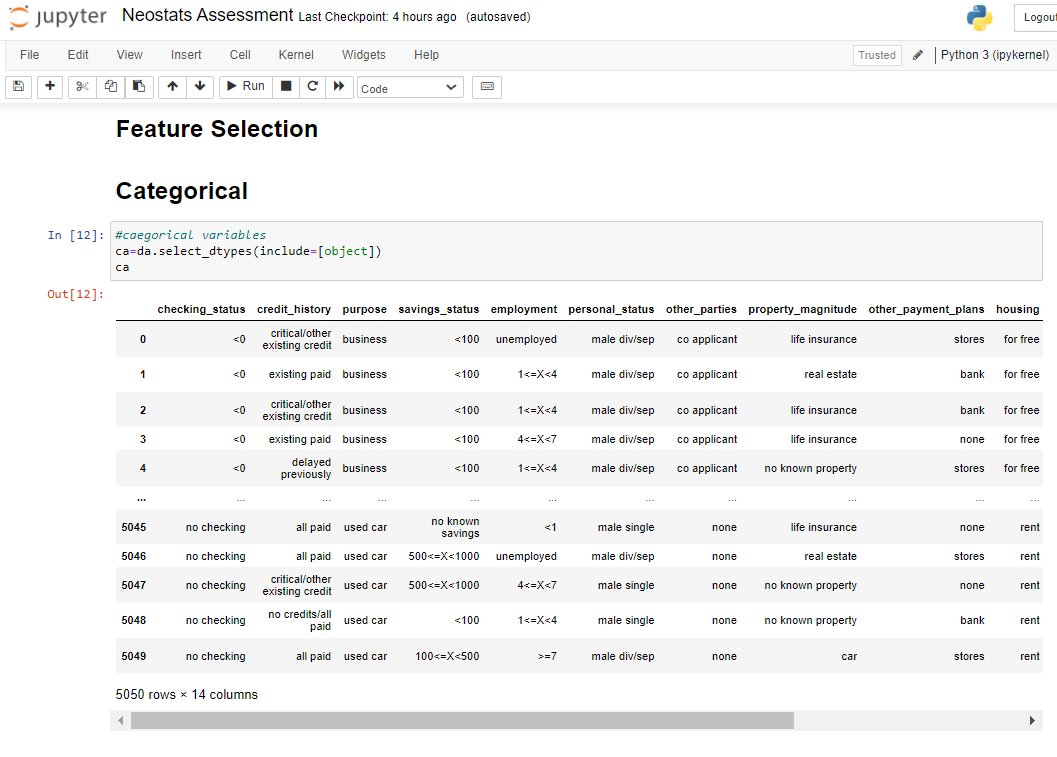
After Interpreting about missing, we will split the variables into two categorical and numeric values, as we can see in the above image.

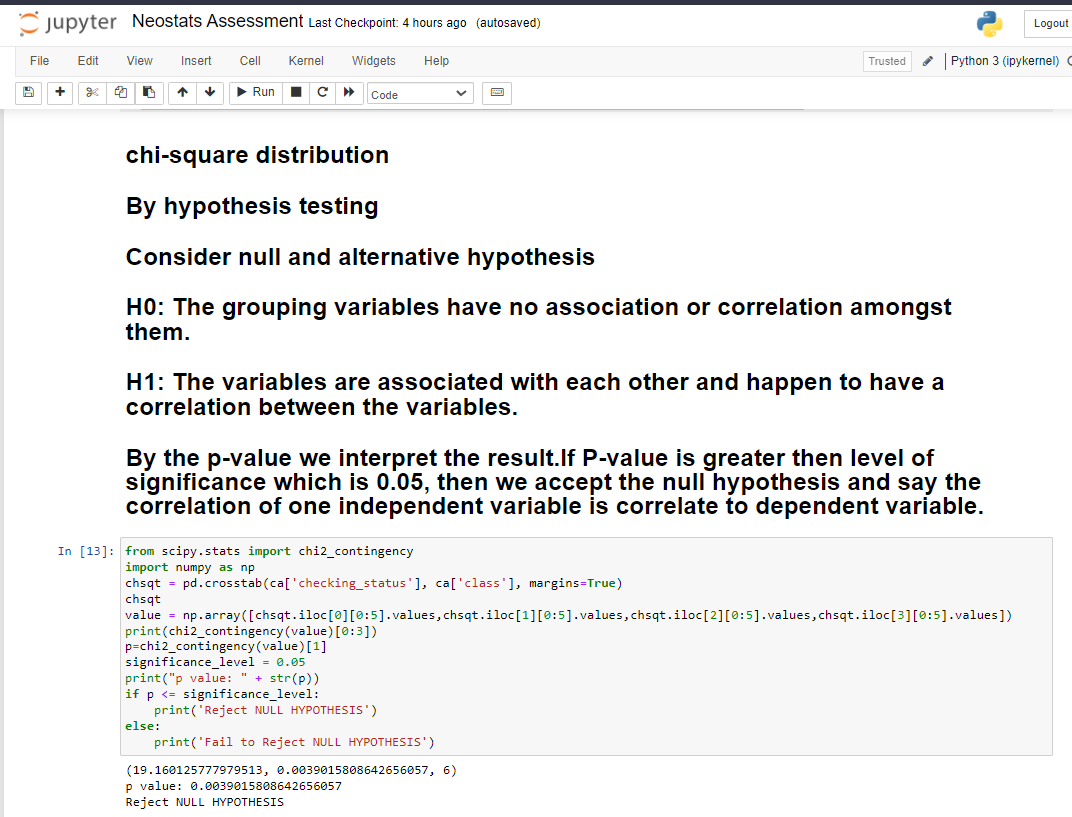
**MODELING IN PYTHON:**

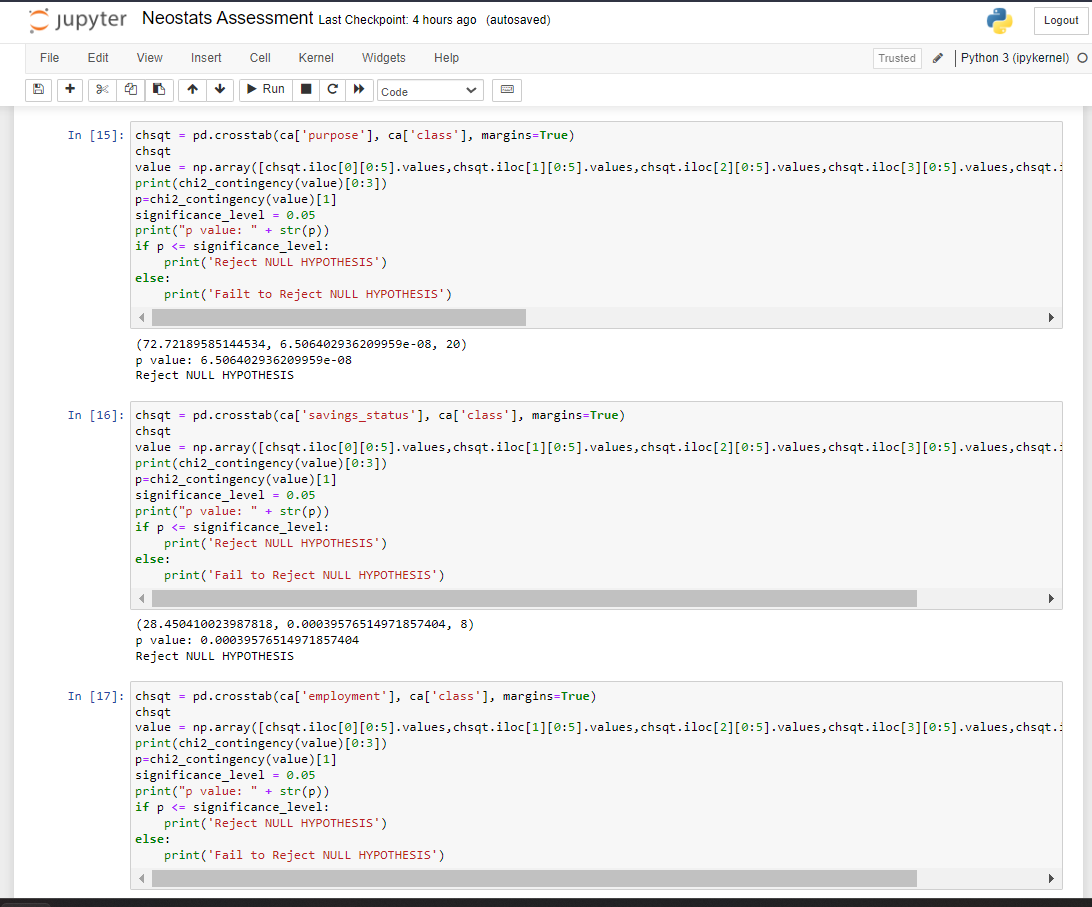
Before Building the model, Feature selection plays a critical role. In Feature selection, we split the data into categorical and numerical data.

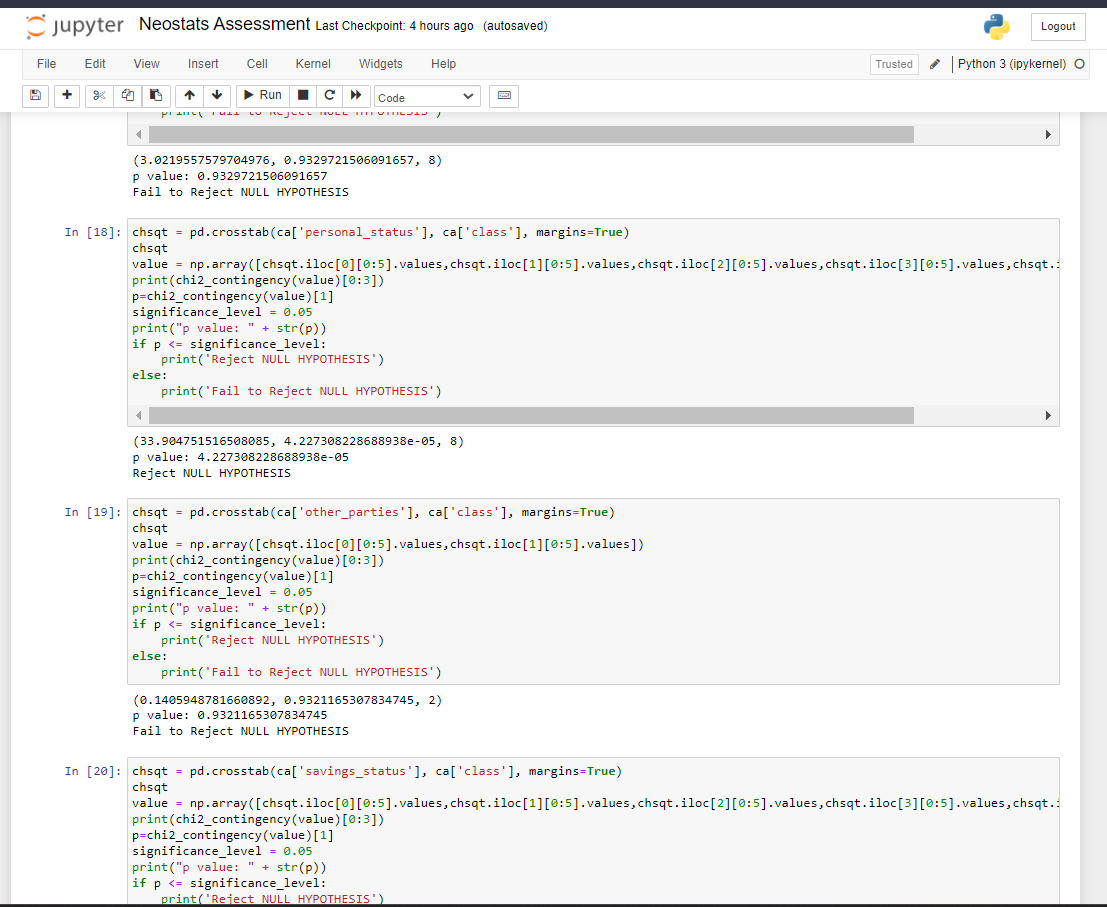
To implement Feature selection in categorical data, we use Chi-square distribution which shows us how our variable is significant to our dependent variable. From the chi-square distribution, we get the p-value for each variable; with this p-value, we can say whether the variable is independent. Suppose the p-value is greater than the level of significance. In that case, we have enough evidence to show that the independent variable is not associated with the dependent variable. It can take as the independent variable for building a model.

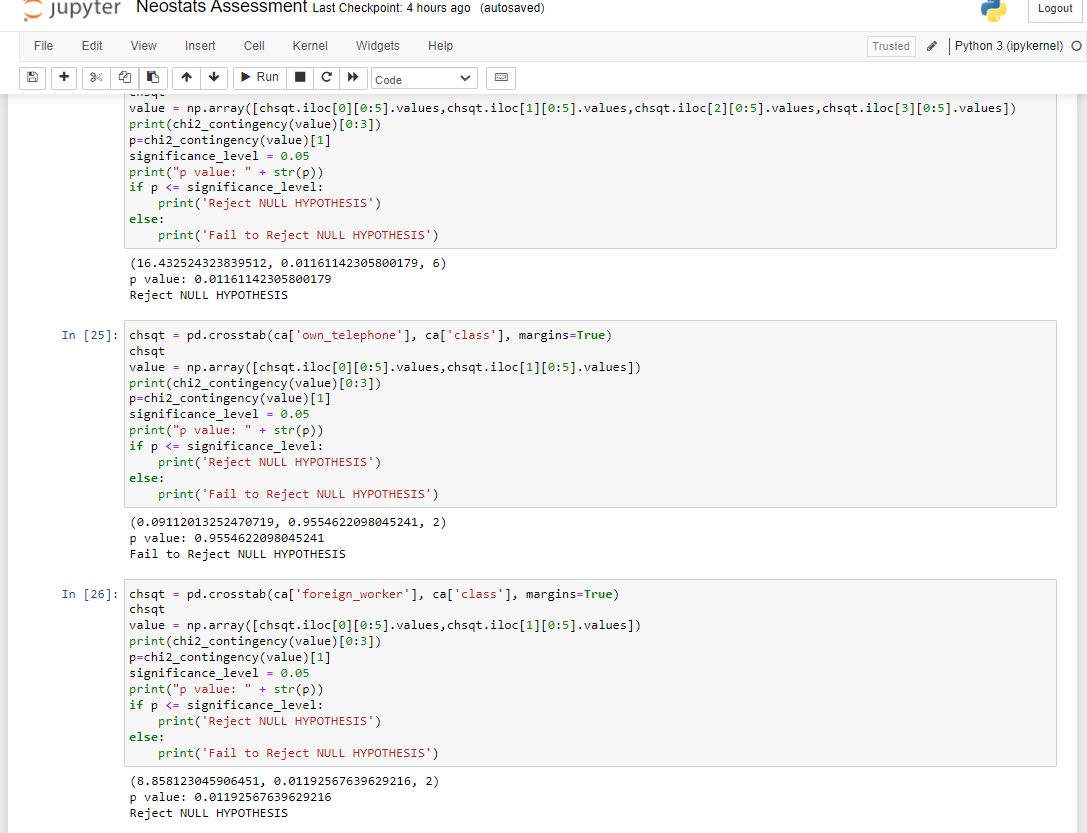
**OUTPUT:**





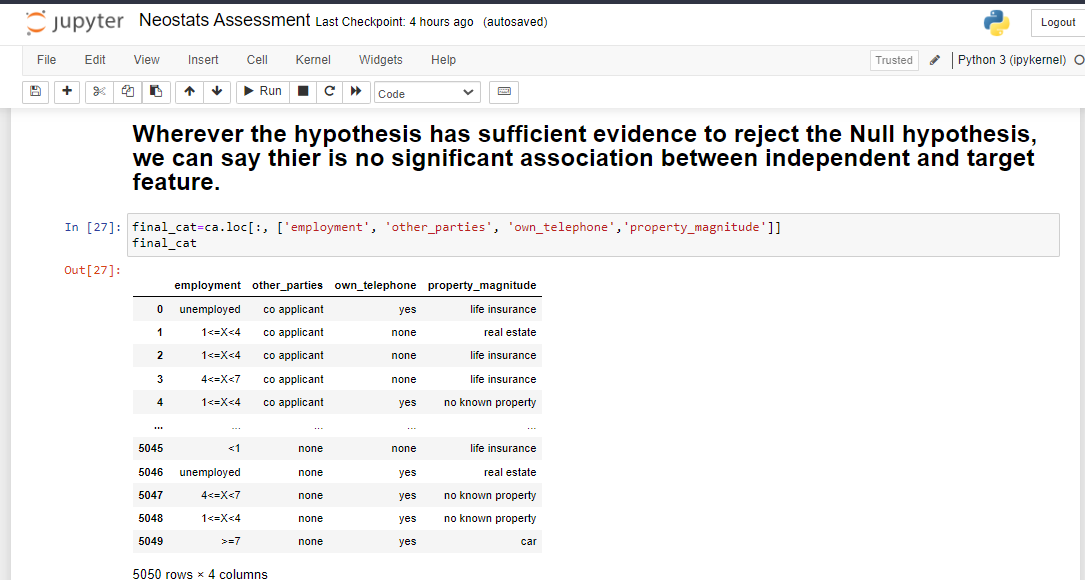




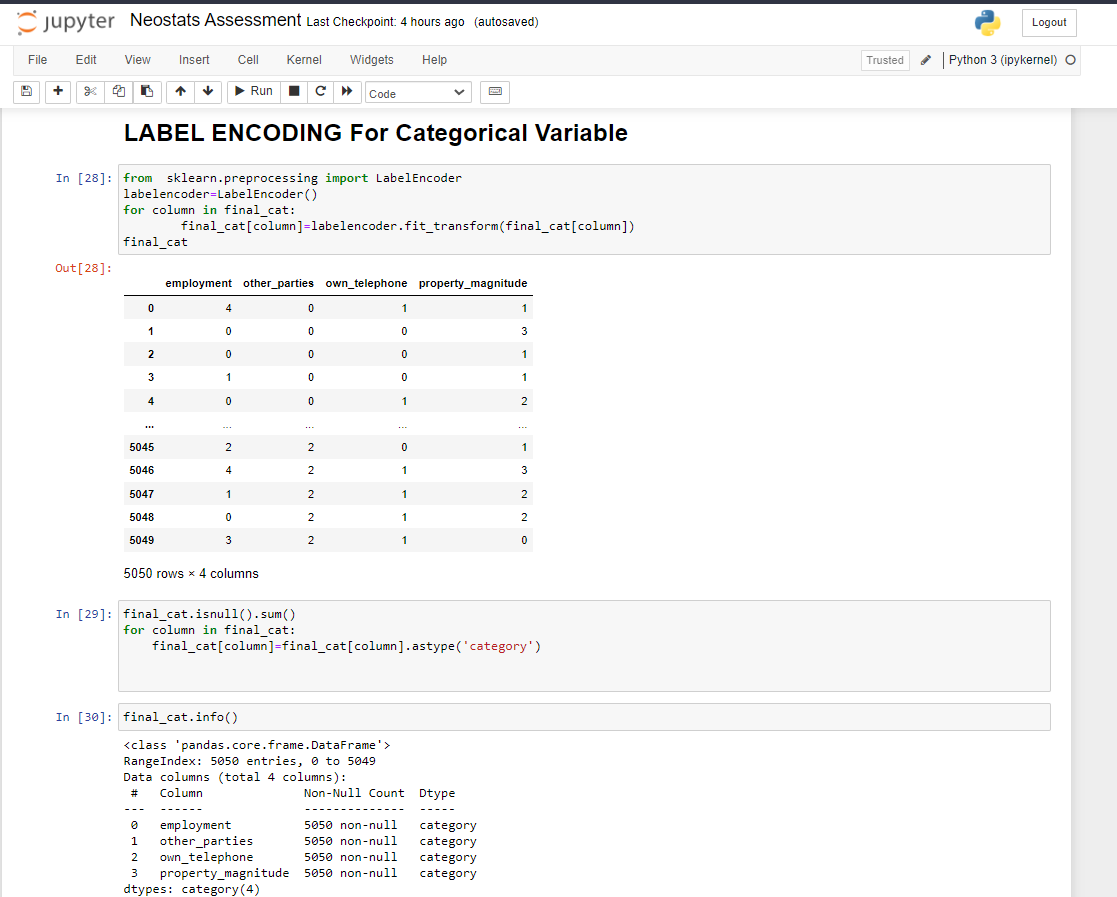


After learning the p-value for that feature, we take hypothesis testing to reject the null hypothesis and fail to reject the null hypothesis.

If the p-value is greater than the significance level, then we fail to reject the null hypothesis and consider that feature for the model.

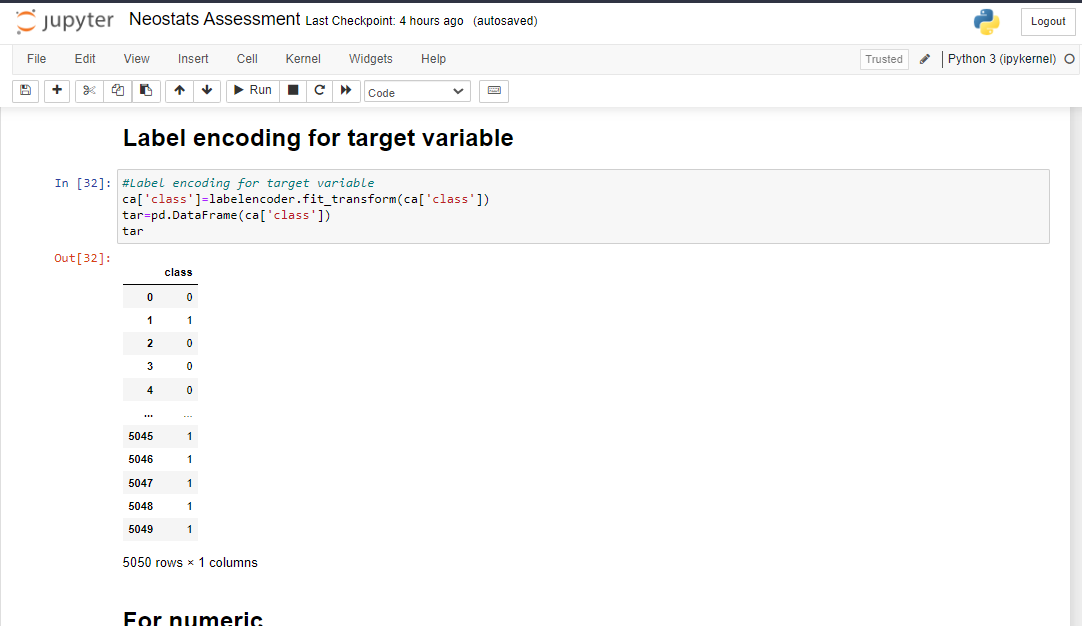


After the Feature selection of Categorical features, it’s time to label them using the Label Encoder () function.

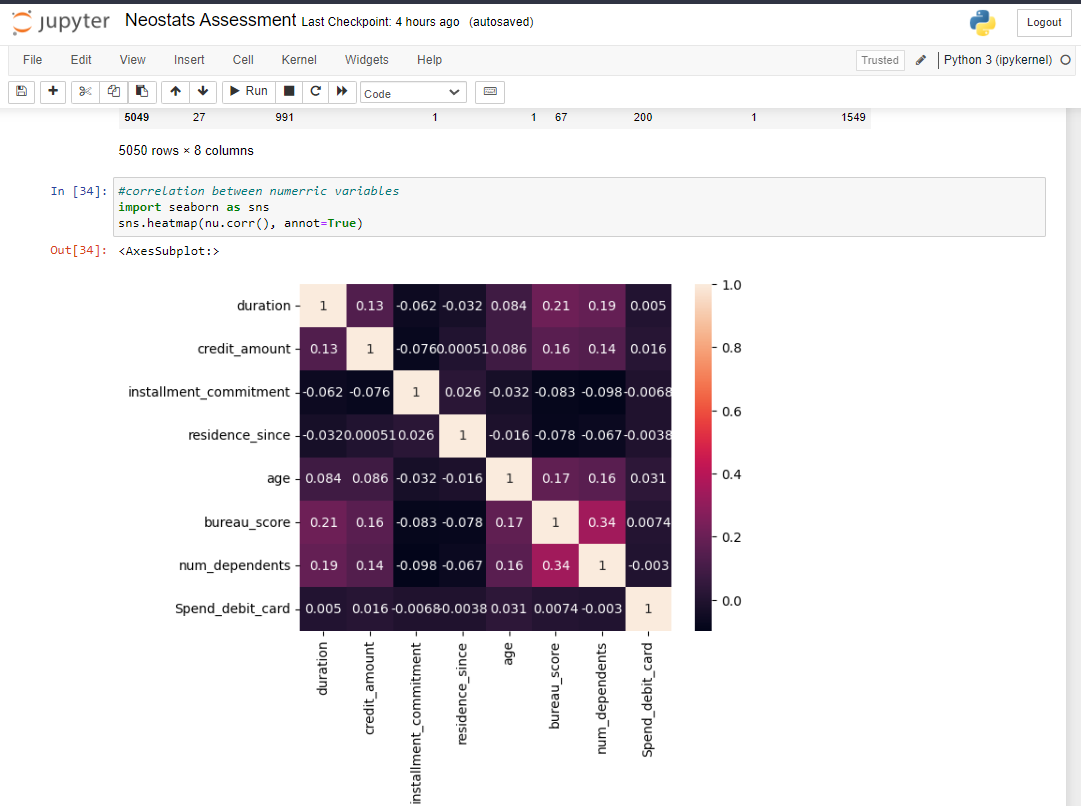


In this step, firstly, we give labels to each variable and convert them into categorical variables. And by using info (), we see column types as categories.

Also, our dependent feature, ‘class,’ is a classification type. We encode them into 0 and 1. Bad=0 and good=1.



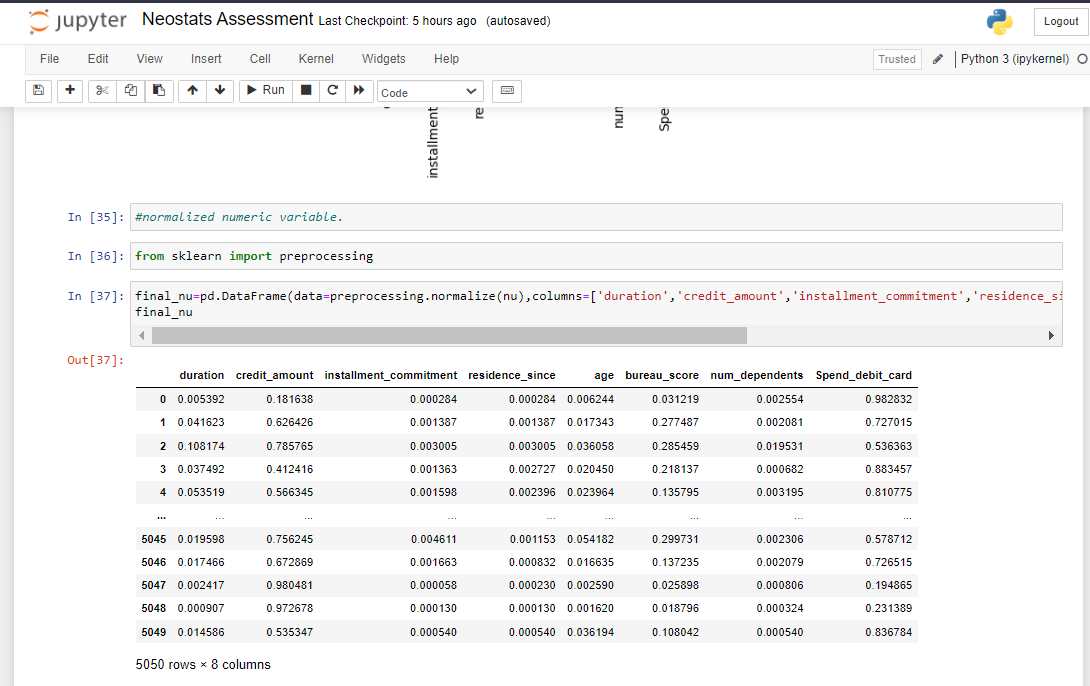
Our Categorical Features and target variable are ready for the model; now it’s time for numeric feature selection. To do this, we use a correlation matrix; by using a heat map, we get each value of how much the variables correlate. If the numeric features are not independent, our model can cause a multi-collinearity factor. For this, our independent variables should be independent of each other.



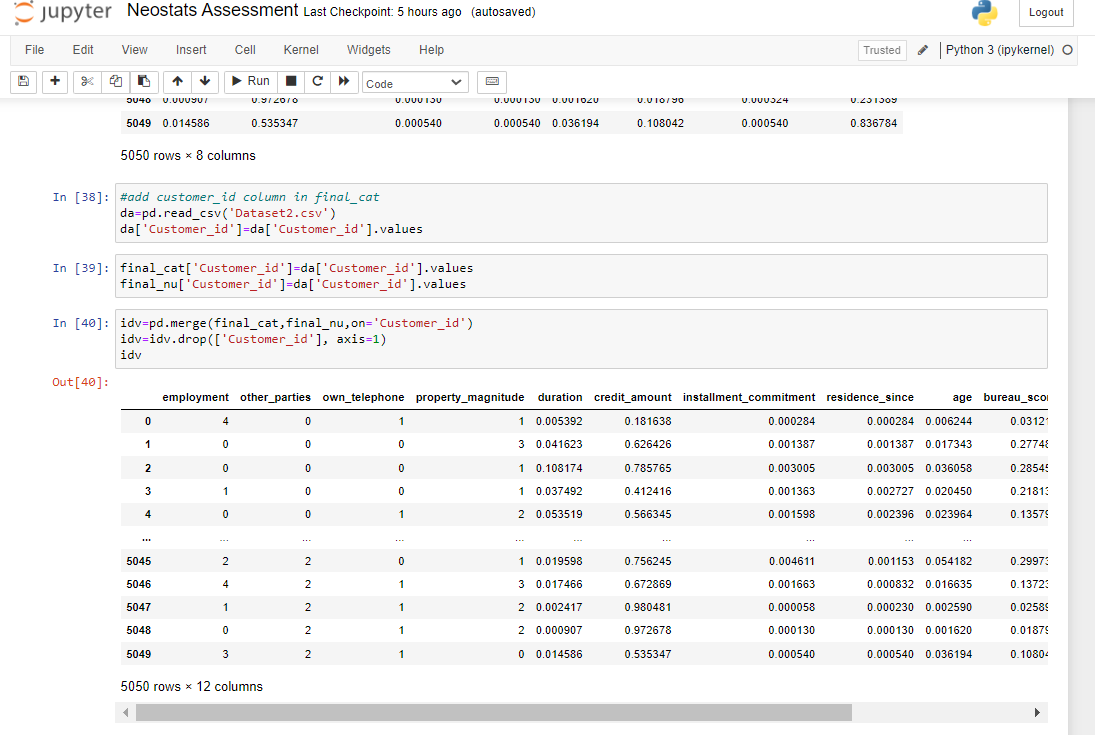
After implementing the heat map, we can see that there is very less correlation between variables. And from this, we interpret that is no multi-collinearity among the variables.

So, we can take all variables to fit the model after normalizing them.

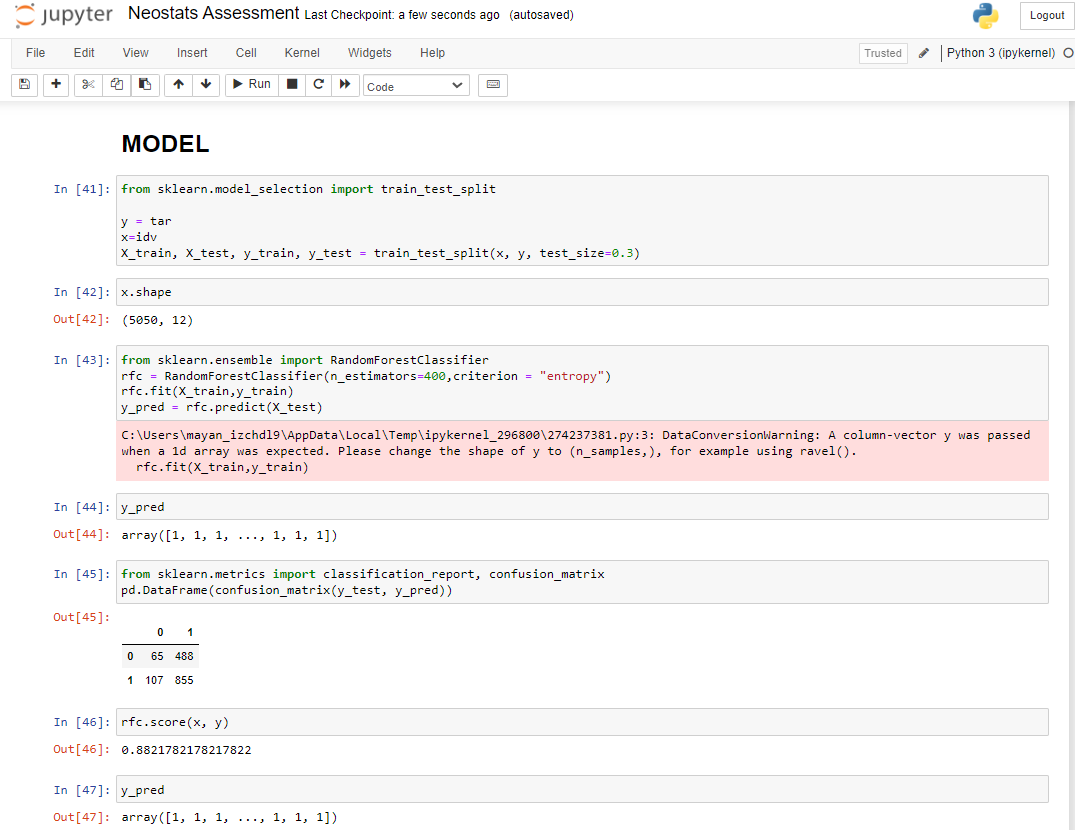
The normalization technique is a very common technique used in data mining to transform the values of a dataset into a common scale. After the data points Normalize, it ranges from 0 to 1 only, which can help to remove unnecessary data points and convert them between values 0 and 1.



After normalizing, we merge the categorical and numerical data frames into one by taking Customer\_id as a common field.



After normalizing the data and merging independent features into one variable, now it’s time to build a model.



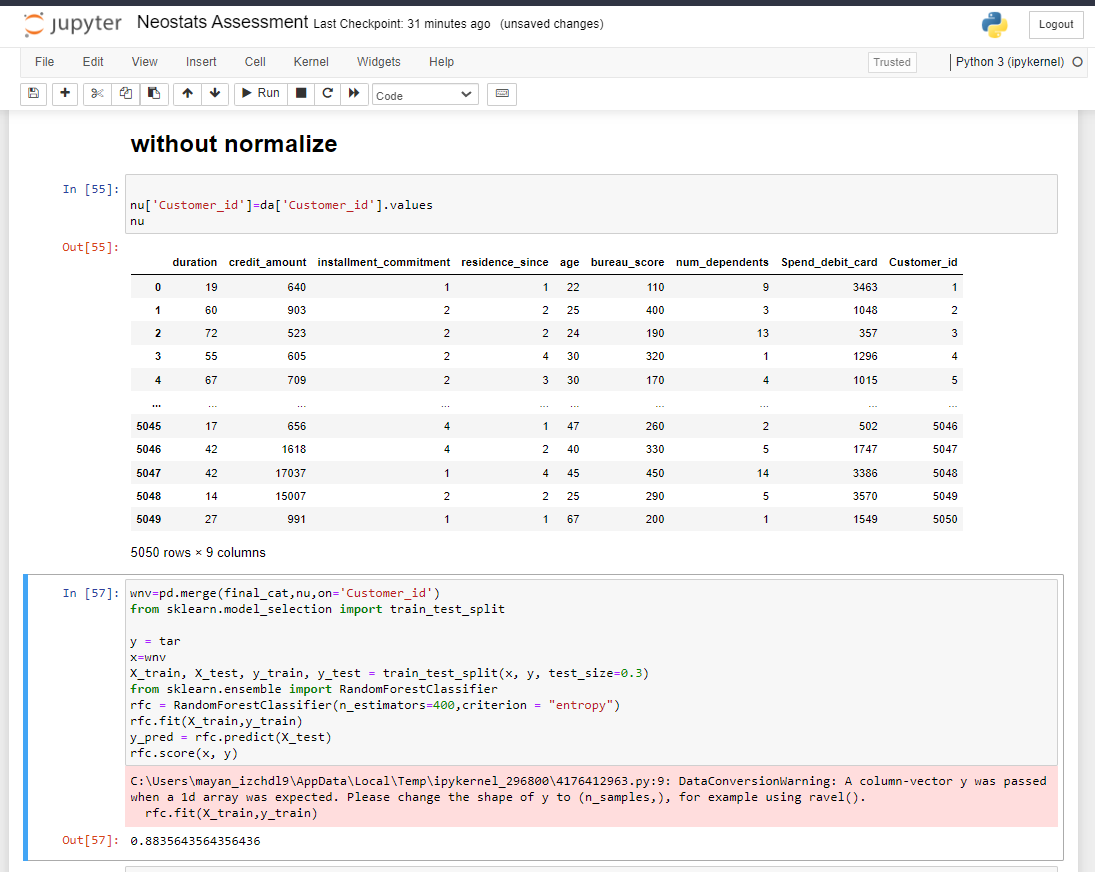
Here, we use Random Forest to build a model and split our data into training and testing by 0.7 and 0.3, respectively.

As we can see, the model score we are getting is 0.8821, which means 88.21% of the model is a good fit with the actual value of the dependent feature.

In the confusion matrix, we can say that 855 is a higher number which says that whatever our predicted value matches the actual values, or we can say the predicted class matches the actual class. The actual was negative, and the model also predicted a negative value.

107, called type-II Error, means the actual value is positive, but the model predicted a negative value. Also, our Type-I error is 488, which says that the actual value is negative, but the model gives us positive values. Comparably our True Negative is higher than type-I and type-II, which shows that the model gives us significant values.

We also implement the model without normalizing the numerical data.



After implementing the model without normalization, we can compare accuracy, which is not much to vary if we normalize the data. This means that we can build a model without normalizing the numerical dataset in this dataset.