Python/PySpark for Machine Learning

Final Home Project

CYT-180
Professor: Tatiana Outkina

Elaborate by:
Leandro Delgado
Student Number:1144162411

CYT180_ Final Home Project – 15%

Project 1. Python/PySpark for Machine Learning

To start working you need to open your account at geeksforgeeks.org site.

With your account you will have access to tutorials with sample of code and explanations. You are able to run the code on your own computer or in the "Google Colab" cloud environment. Decide for yourself.

In this project you learn how to build application that would be able to recognize fraudulent credit card transactions so that the customers of credit card companies are not charged for items that they did not purchase. It starts from explanation of challenges to be addressed via your application. Then you proceed step-by-step via the process of building application.
Sample of data credit.csv file is enclosed in task description.

https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/

Your tasks:

1. **Read the explanation of challenges and description of methods how the challenges are addressed.**
2. **Run the code in your Python environment – your own or Google Colab.**
3. **Make screenshots and meaningful comments. Your comments must be commensurate with the challenges you have learned in #1.**
4. **Organize your submission.**

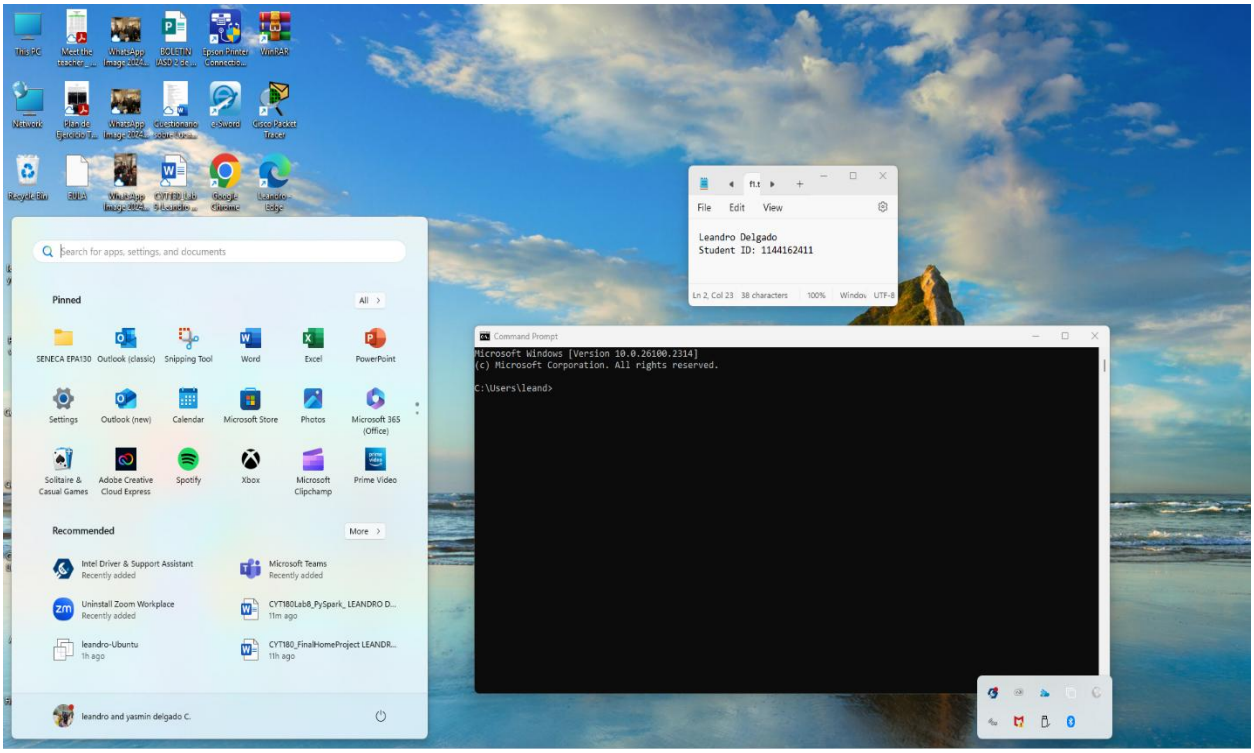
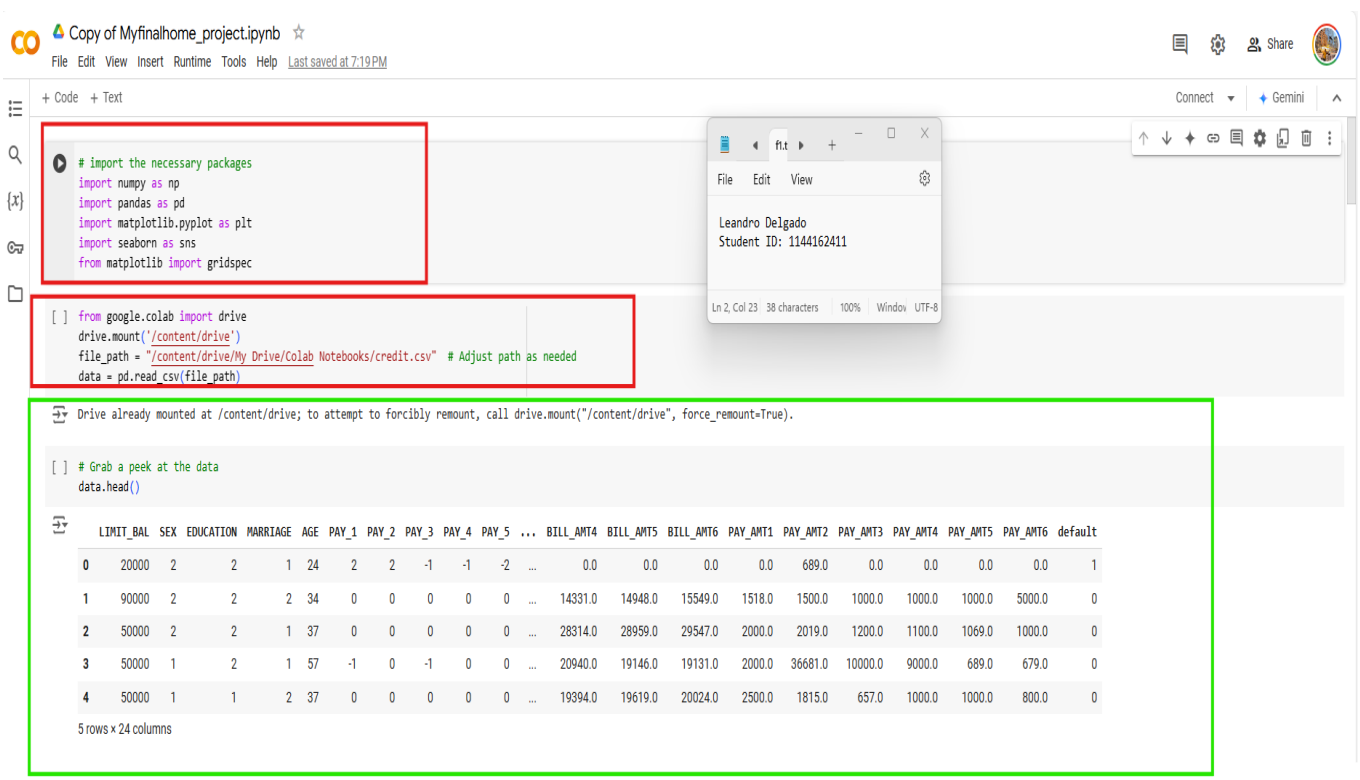


Figure 1- O screeshots.



	LIMIT_BAL	SEX	EDUCATION	MARRIAGE	AGE	PAY_1	PAY_2	PAY_3	PAY_4	PAY_5	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5	PAY_AMT6	default
0	20000	2	2	1	24	2	2	-1	-1	-2	...	0.0	0.0	0.0	0.0	689.0	0.0	0.0	0.0	0.0	1
1	90000	2	2	2	34	0	0	0	0	0	...	14331.0	14948.0	15549.0	1518.0	1500.0	1000.0	1000.0	1000.0	5000.0	0
2	50000	2	2	1	37	0	0	0	0	0	...	28314.0	28959.0	29547.0	2000.0	2019.0	1200.0	1100.0	1069.0	1000.0	0
3	50000	1	2	1	57	-1	0	-1	0	0	...	20940.0	19146.0	19131.0	2000.0	36681.0	10000.0	9000.0	689.0	679.0	0
4	50000	1	1	2	37	0	0	0	0	0	...	19394.0	19619.0	20024.0	2500.0	1815.0	657.0	1000.0	1000.0	800.0	0

5 rows × 24 columns

*Figure 2-**Importing all the necessary Libraries/Loading the Data/Understanding the Data.***

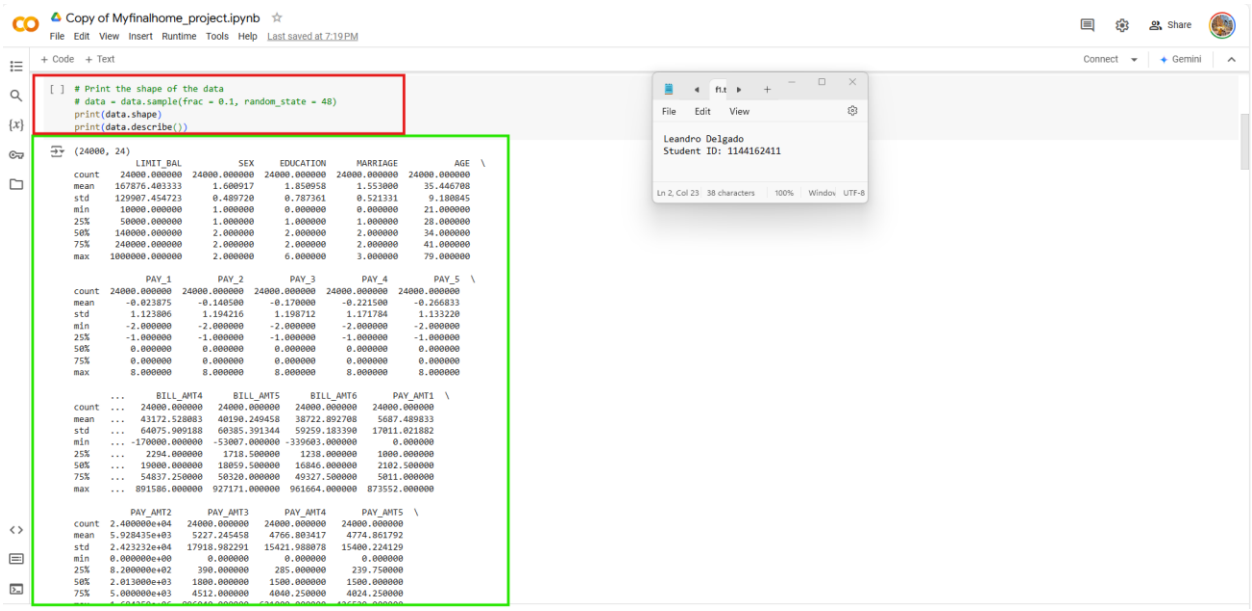


Figure 3- ***Describing the Data.***

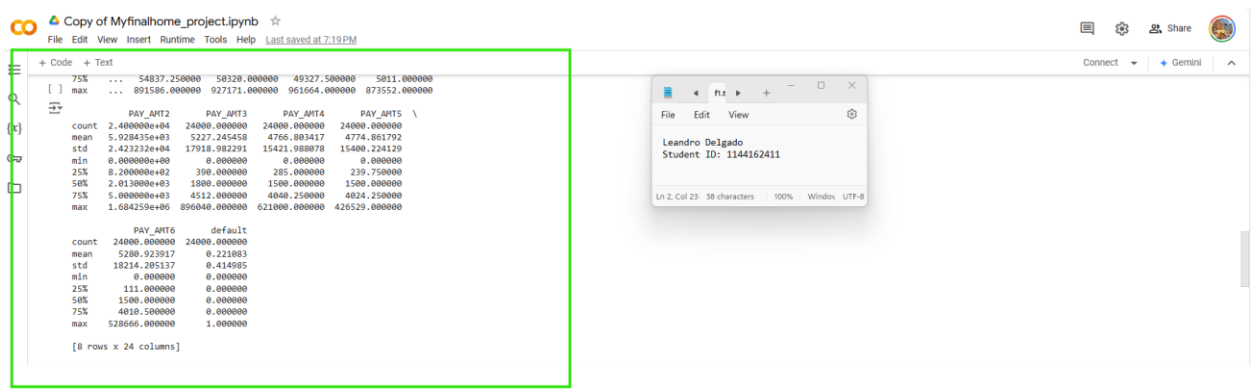


*Figure 4-****Describing the Data.***

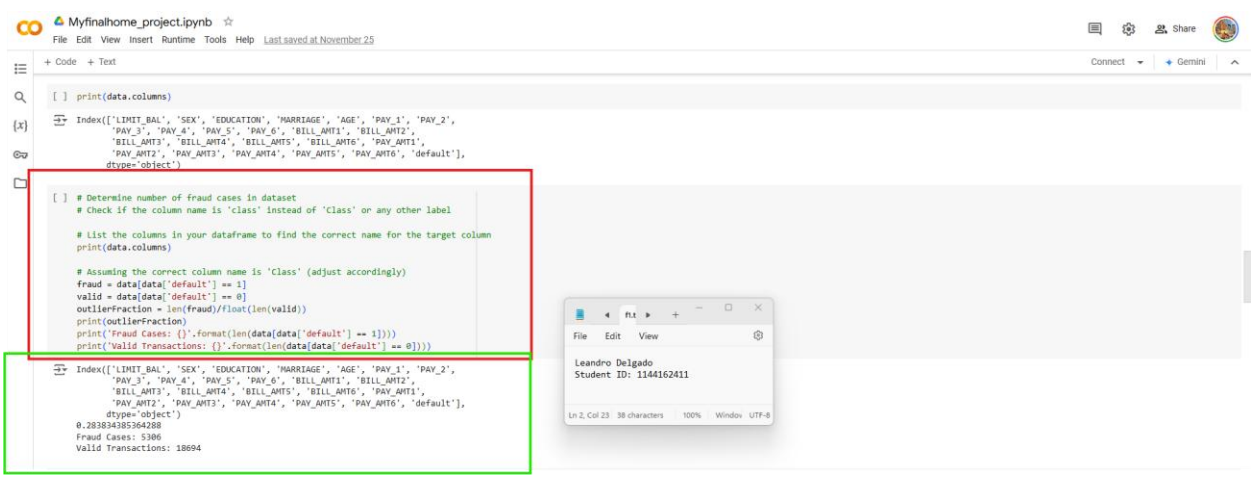


Figure 5- ***Imbalance in the data.***

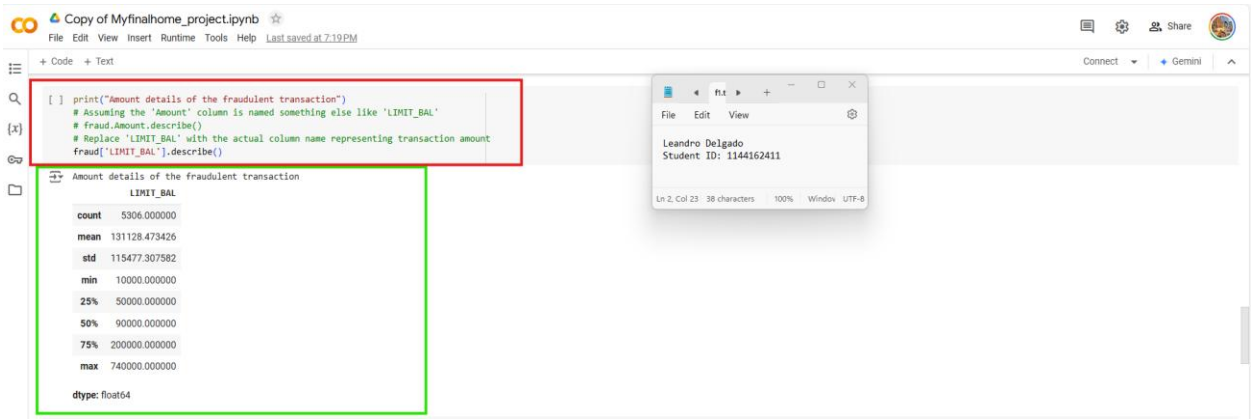


Figure 6- ***Print the amount details for Fraudulent Transaction.***

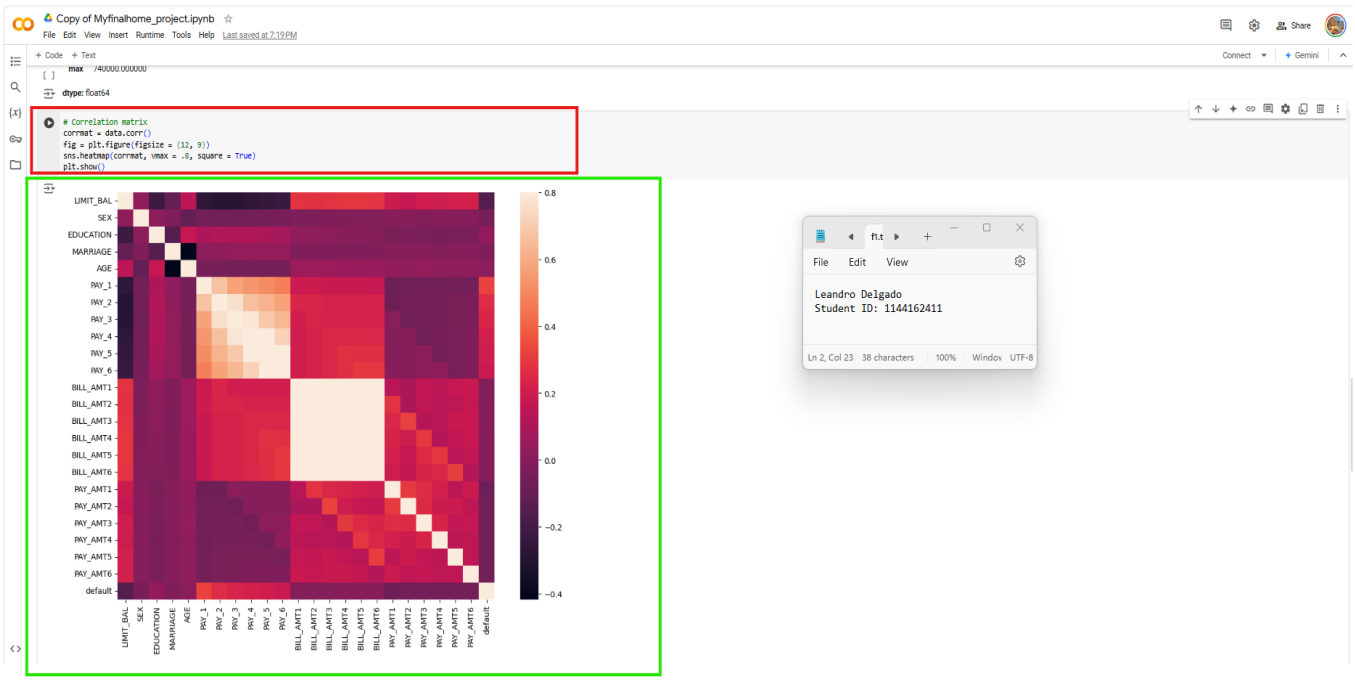


Figure 7- ***Plotting the Correlation Matrix.***

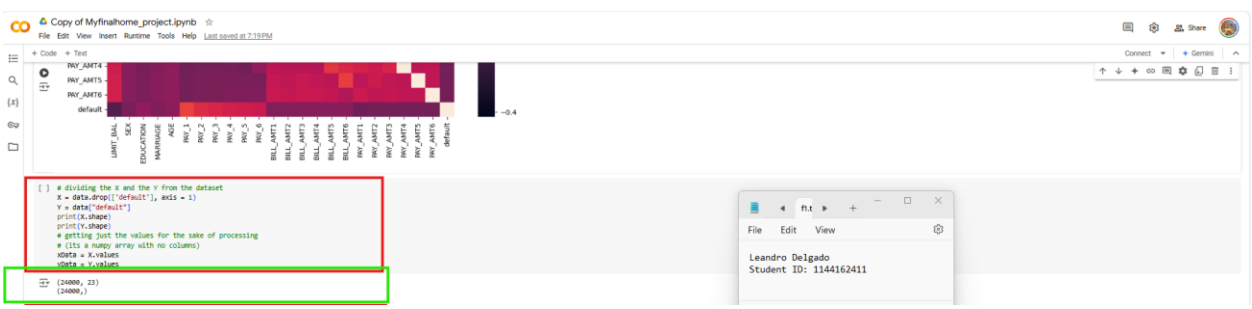


Figure 8- ***Separating the X and the Y values.***

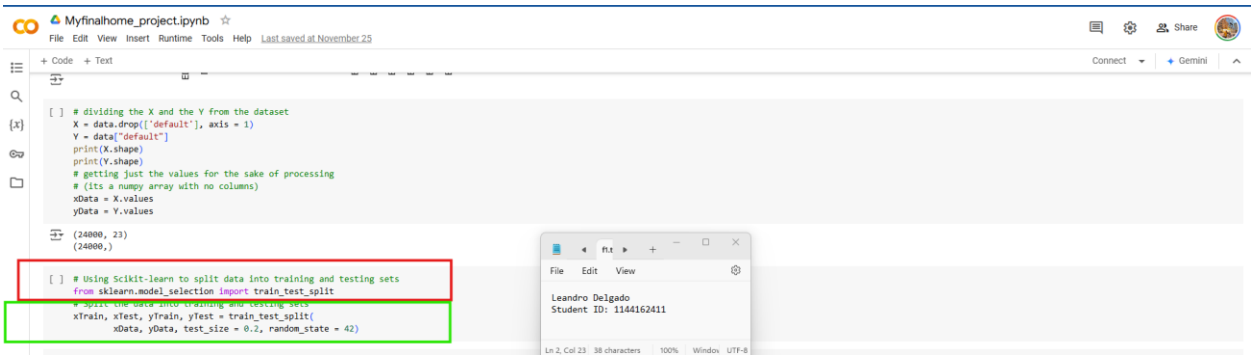


Figure 9- ***Training and Testing Data Bifurcation.***

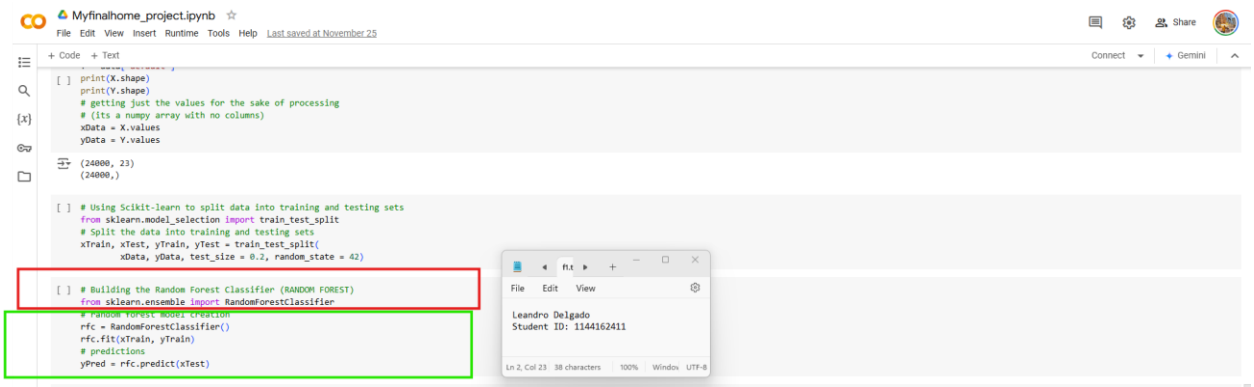


Figure 10- Figure 10- ***Building a Random Forest Model using scikit learn.***

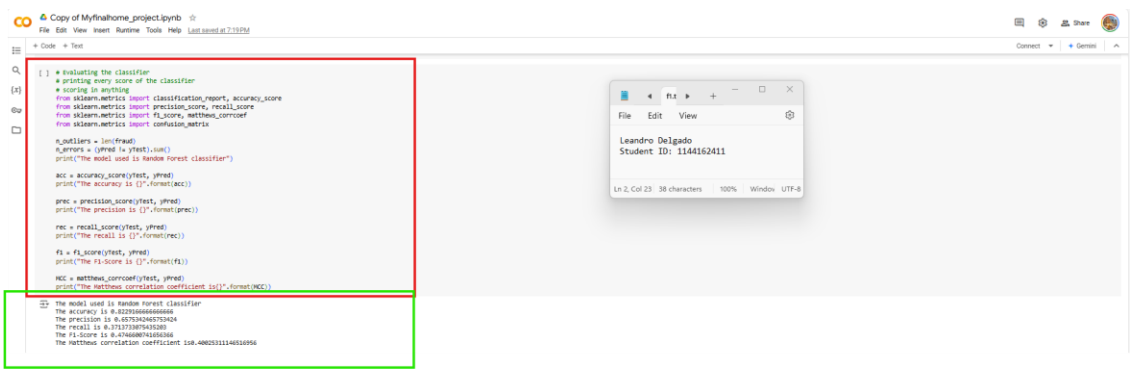


Figure 11- ***Building all kinds of evaluating parameters.***

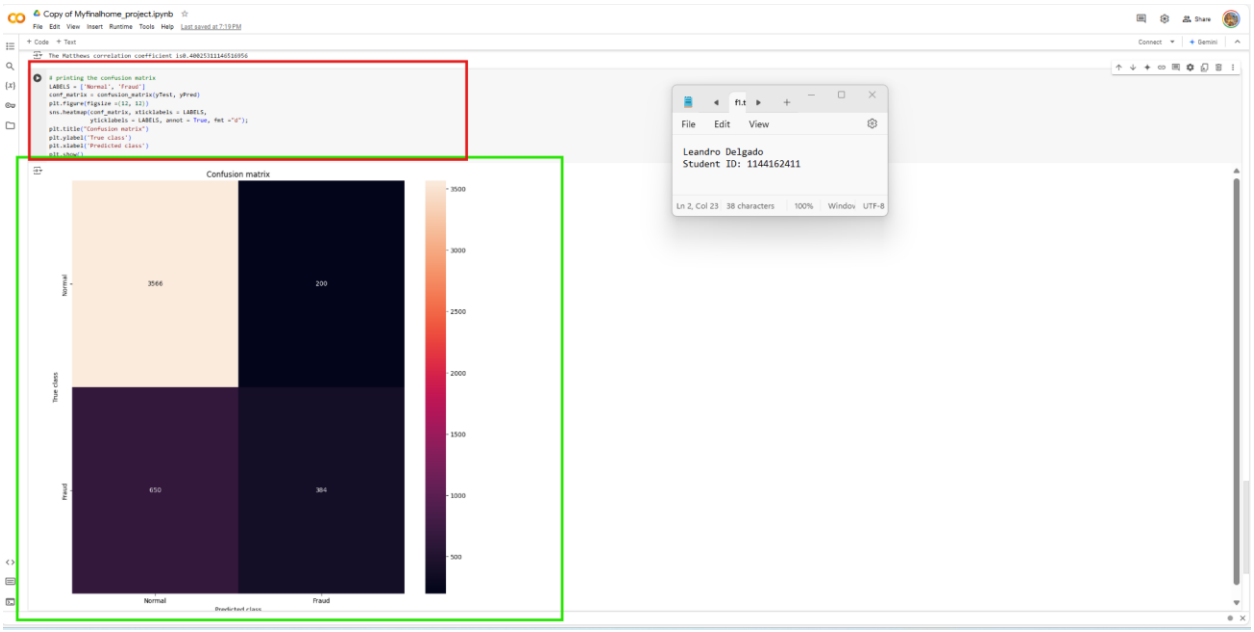


Figure 12- ***Visualizing the Confusion Matrix.***

Summary:

Is important to understand how serious and vital credit card fraud is nowadays. My project identified normal and fraudulent transactions through data provided by the professor, but it also missed some cases of fraud, which shows that there is room for improvement in the process.

1. The **confusion matrix** generated in your code showcases the performance of the trained model. It indicates how well the model can distinguish between normal and fraudulent transactions.

    - **True Positives (Fraud correctly detected):** 384
    - **False Positives (Normal misclassified as Fraud):** 200
    - **True Negatives (Normal correctly detected):** 3566
    - **False Negatives (Fraud missed):** 630

- This suggests that the model is detecting fraudulent transactions with some misses (False Negatives), which is critical to improving in fraud detection tasks.

If I am comparing with the next table of algorithms without dealing with the imbalancing of the data.:

What other Data Scientists got

Method Used	Frauds	Genuines	MCC
Naïve Bayes	83.130	97.730	0.219
Decision Tree	81.098	99.951	0.775
Random Forest	42.683	99.988	0.604
Gradient Boosted Tree	81.098	99.936	0.746
Decision Stump	66.870	99.963	0.711
Random Tree	32.520	99.982	0.497
Deep Learning	81.504	99.956	0.787
Neural Network	82.317	99.966	0.812
Multi Layer Perceptron	80.894	99.966	0.806
Linear Regression	54.065	99.985	0.683
Logistic Regression	79.065	99.962	0.786
Support Vector Machine	79.878	99.972	0.813

Comparison with the other Algorithm.

Figure 13- ***Comparing with the other Algorithm.***

I got that the MCC score of 0.4003, the model is effective, but not essentially robust as other advanced methods used such as support vector machines or multilayer perceptron. While the accuracy is okay, the recall needs work since missing fraudulent cases can be costly. This project provides important lessons to consider and that by trying different approaches, adjustments can be made to improve the results. It has been a great step to start and understand the complexities of fraud detection and what actions can be taken to make these models more reliable for use in the real world.