

**Data Mining Project**

**Patient survival**

Sana Mourad 20200946

Ruby Kalha 20200453

Rand AL-Shobaki 20200956

1. **Abstract**

In this study, we present an extensive analysis of a data set, which only included patients who were admitted to an ICU, that was collected and cleaned for the purpose of building predictive models.

Seven different machine learning models were trained and evaluated on the cleaned data set.

The results show that the random forest model achieved the highest overall performance, with an accuracy measure of 92%.

1. **Introduction**

Predictive modeling is a powerful tool for understanding complex systems and making informed decisions. One of the key challenges in building predictive models is obtaining high-quality and relevant data.

## In this study, we present an in-depth analysis of a data set that includes patients who were admitted to an ICU, with a focus on those with the chronic condition of diabetes. The data was collected in 2021 from MIT’s GOSSIS (Global Open-Source Severity of Illness Score) initiative in the US. The dataset contains a variety of features and is pre-processed to ensure the quality and consistency of the data. We will be using the data of patients to predict if they’re going to survive the ICU admission, by applying EDA and feature engineering techniques and using several machine learning models to see which model would best fit the data.

Seven different machine learning models were trained and evaluated on the data, with performance measured using metrics such as accuracy, precision, recall, and F1-score.

1. **Proposed Approach**

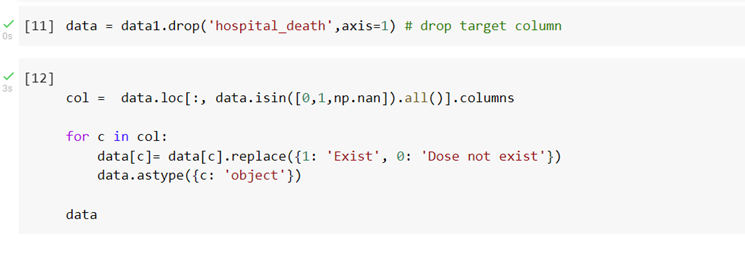
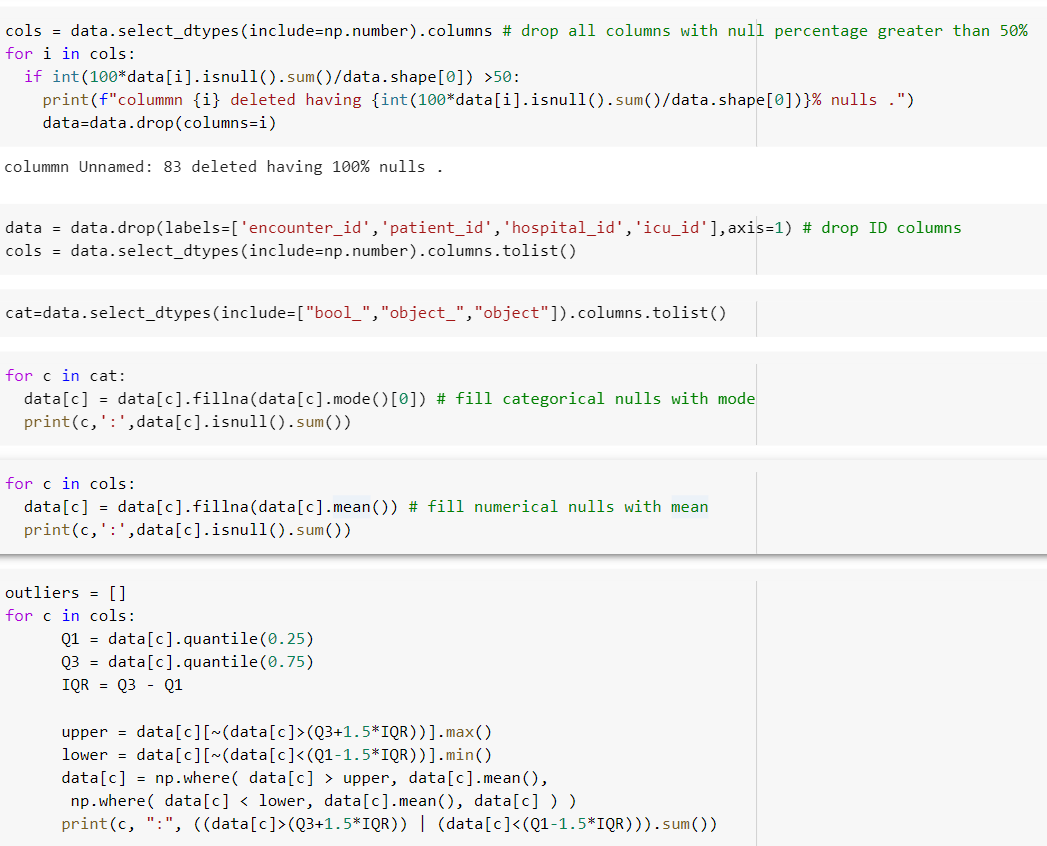
In this dataset, which consists of 91713 records with 85 features, there are various factors given, which are involved when a patient is hospitalized. Based on these factors, we will predict whether the patient will survive or not, by finding relationships, inconsistencies, and most relevant features to the target using some statistical tests.

* 1. **Exploring data**

We start by taking an overview on some of the records and the purpose of each column. Then we start checking the data types for each column and identify the target column to know our problem type (classification or regression problem). Then we check for duplicates, outliers, and missing values.

* 1. **Data Engineering**

First, we transformed all binary classified columns (0,1) from numerical to categorical columns, to better handle them especially when scaling. Then we handled the outliers and nulls of numerical columns by replacing them with their mean, and for the categorical columns with their mode. After that we checked if the labels were imbalanced and applied undersampling for the majority class.



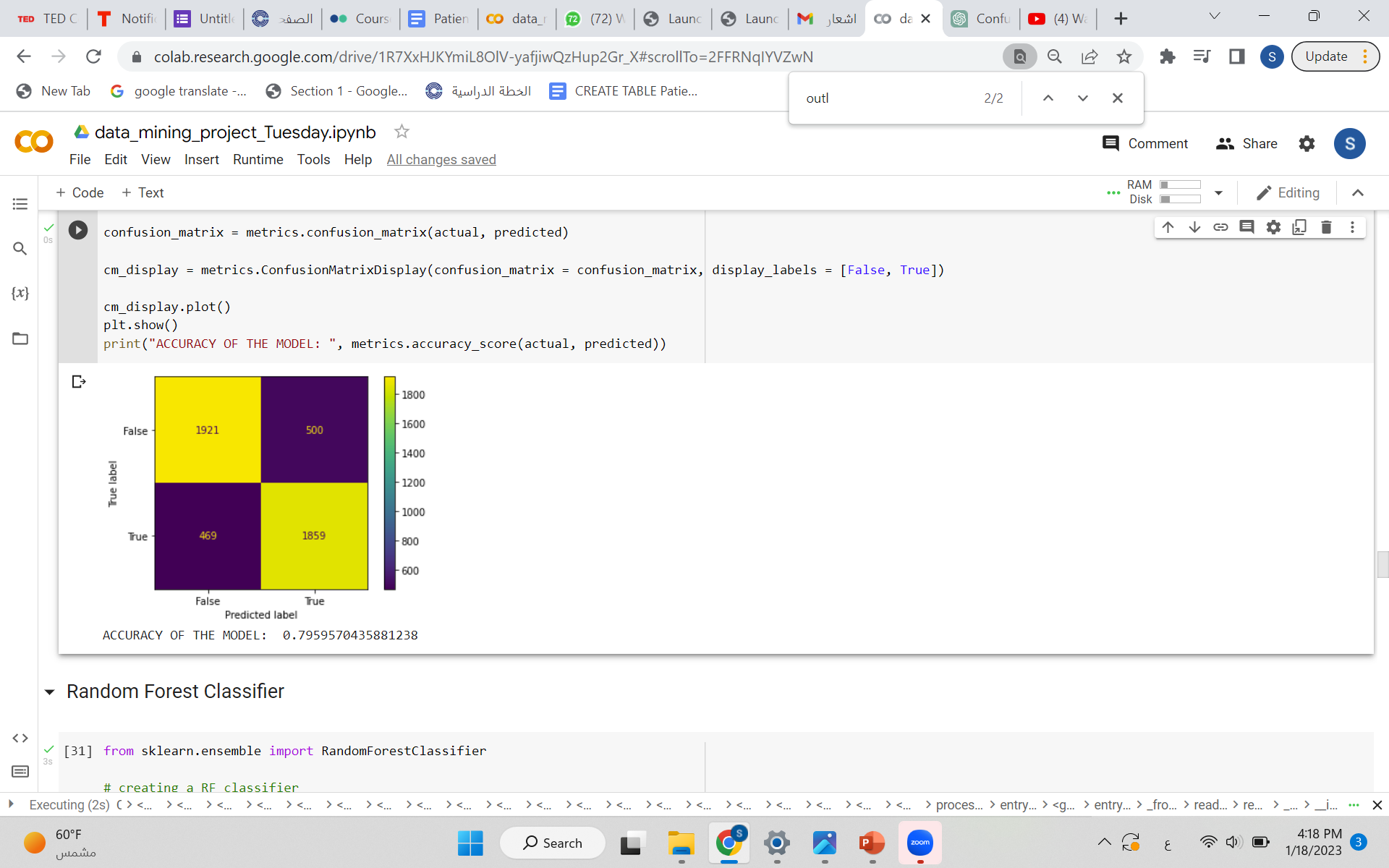
## 

* 1. **Feature Engineering**

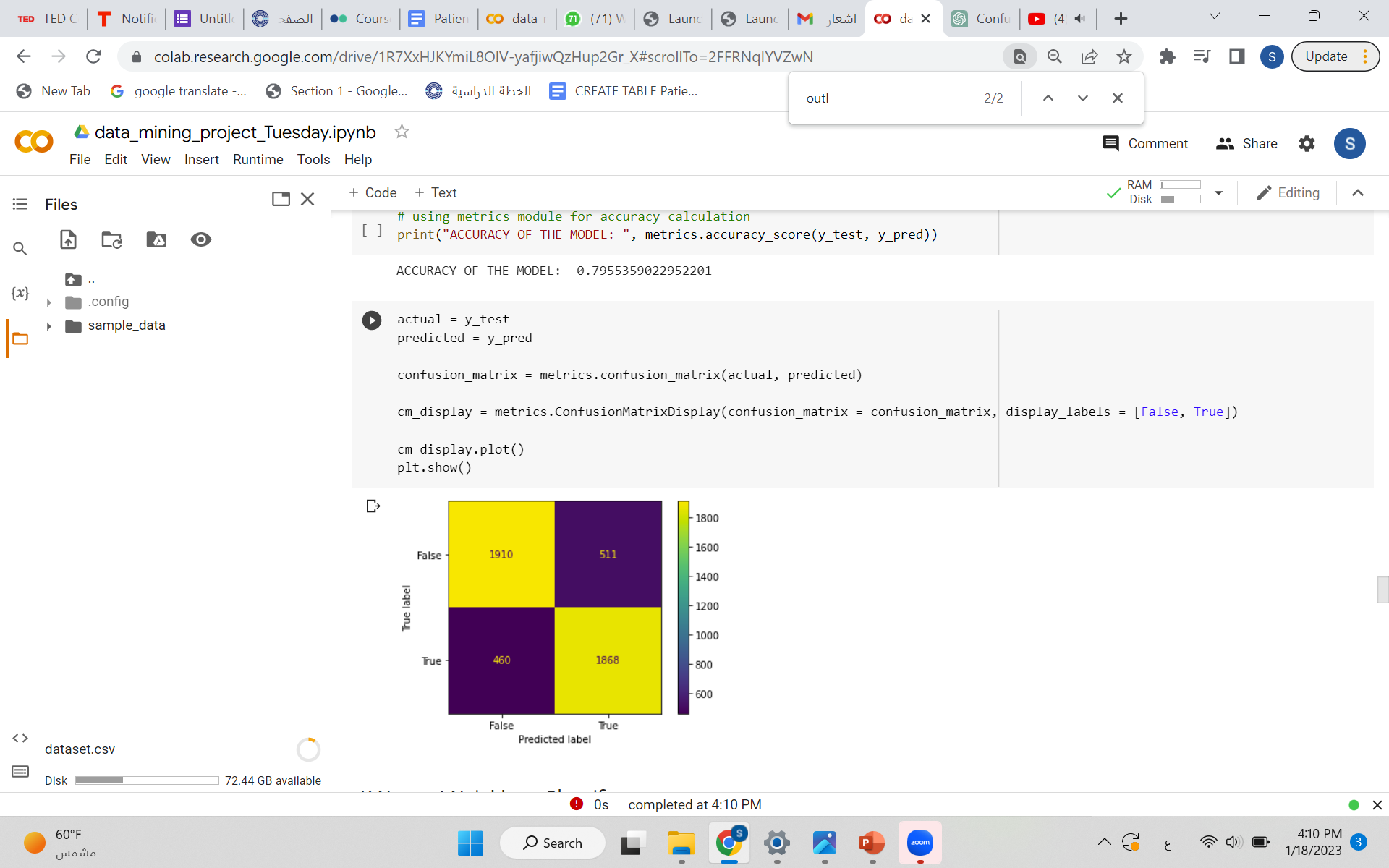
We applied feature scaling for numerical (standardization) and categorical (labelEncoder) columns. Then we did feature selection by choosing the most relevant features, which had high correlation with the target column. By that we ended with 40 columns instead of 84, and that’s how we applied dimensionality reduction.

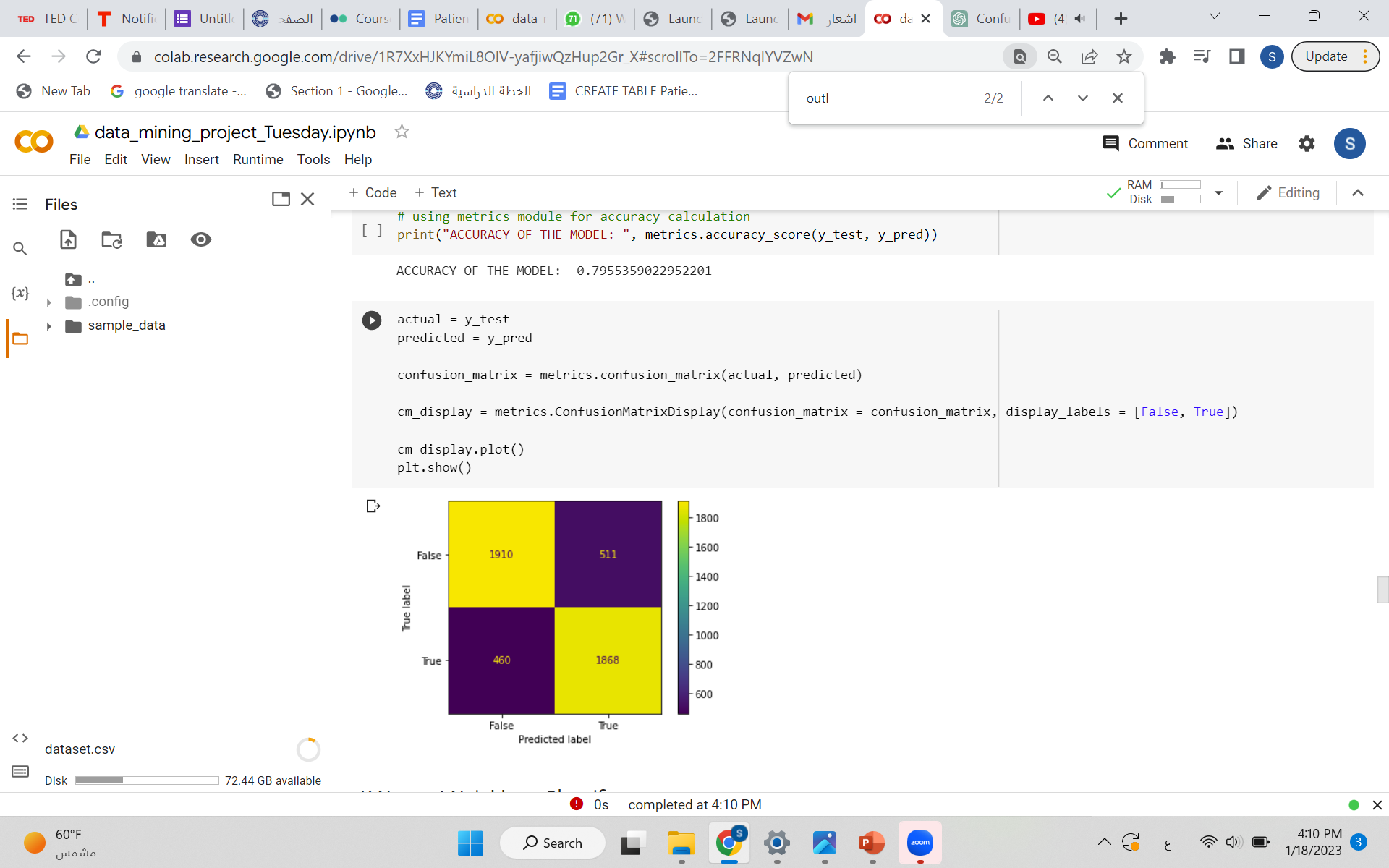
1. **Modeling**
   1. **Model performance evaluation measures table below:**

• XGBoost Classifier:

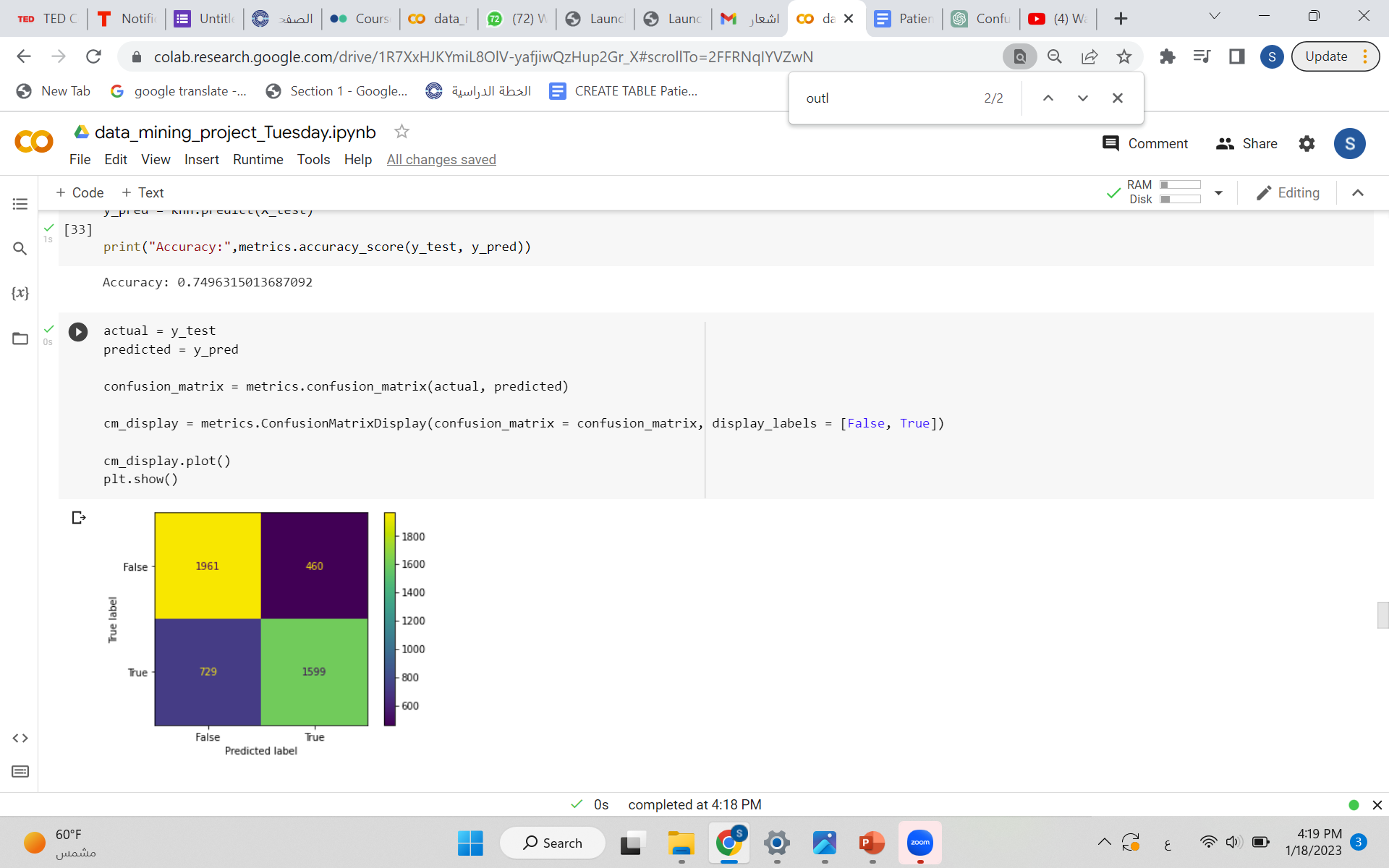


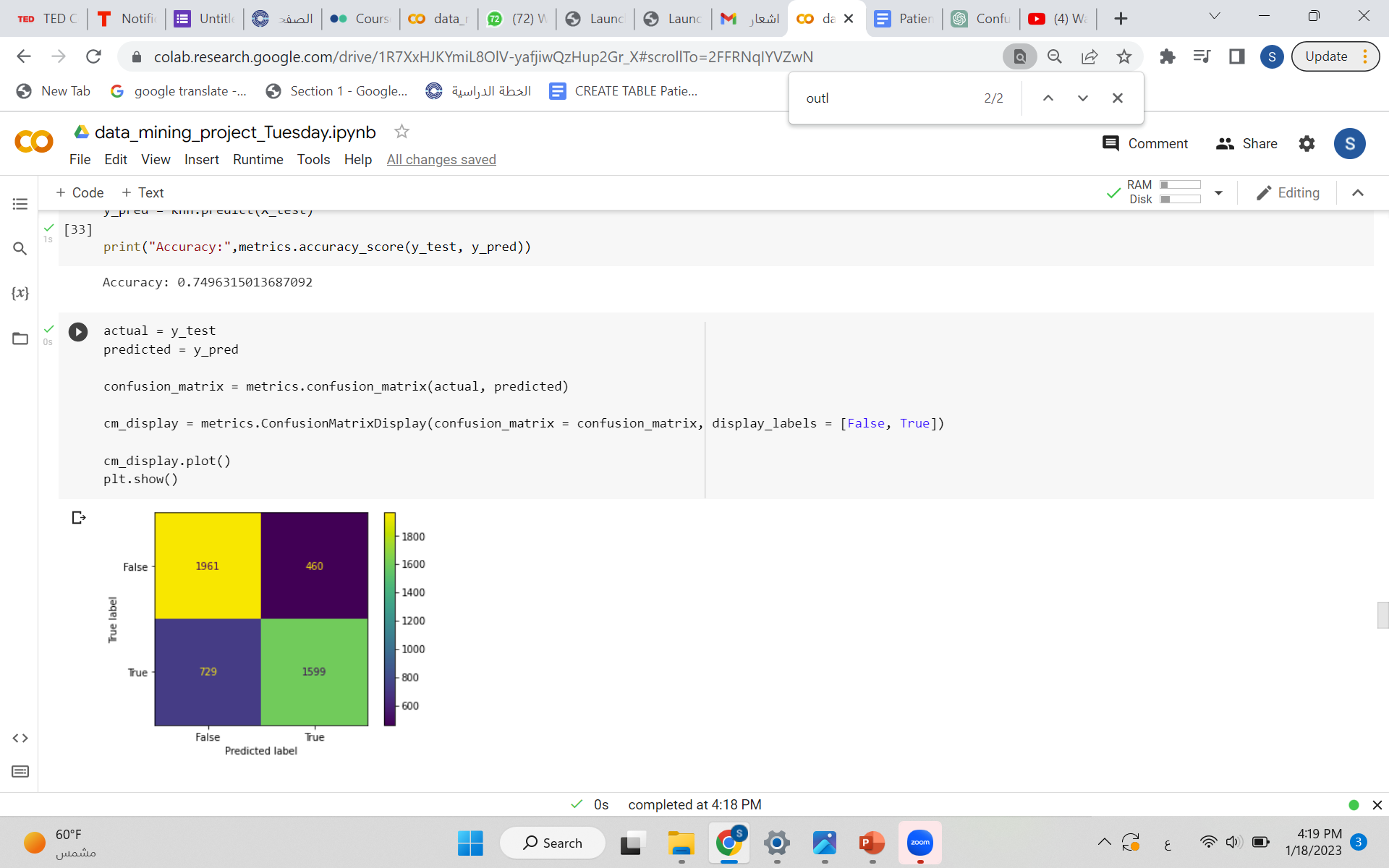
• Random Forest Classifier:



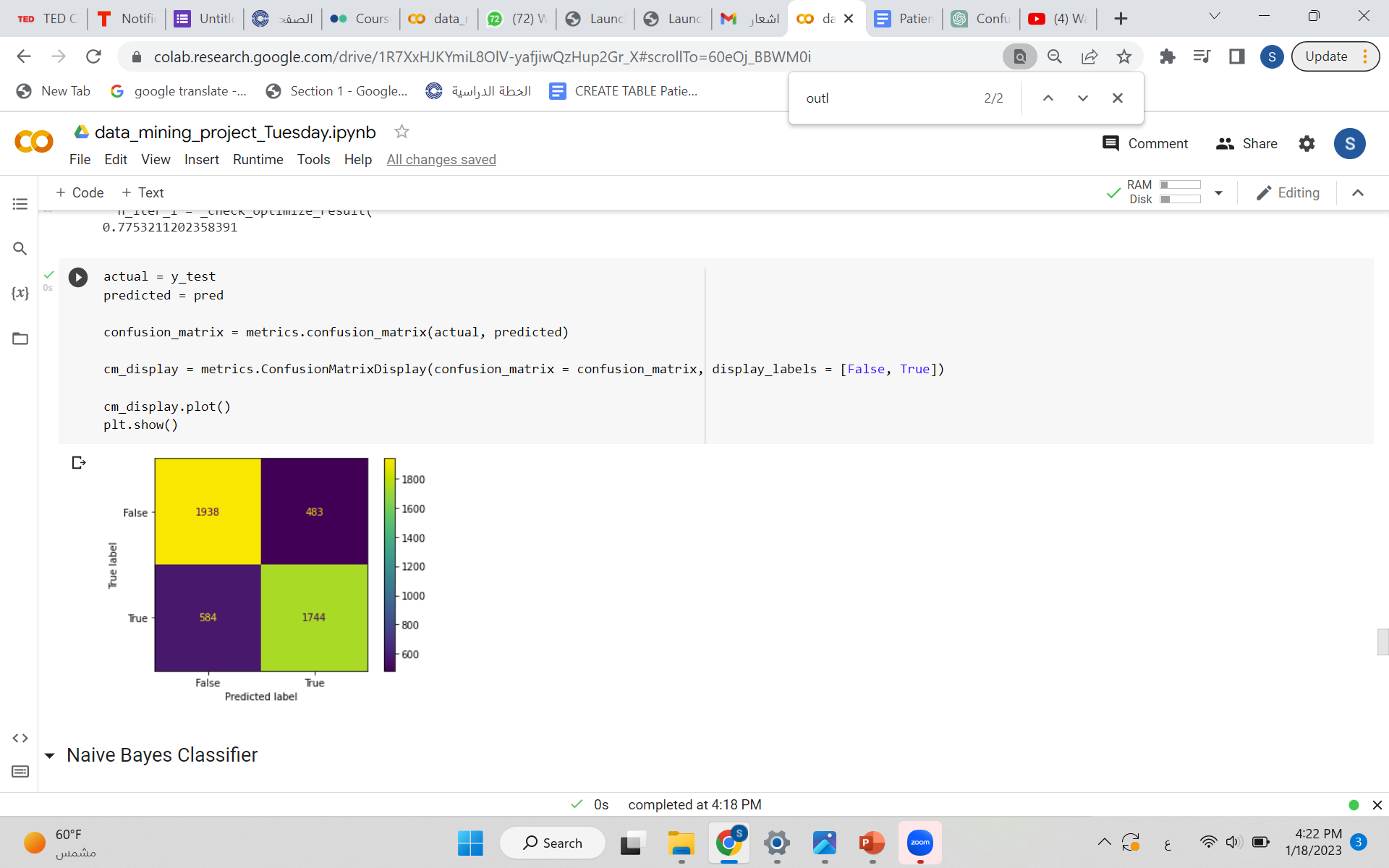


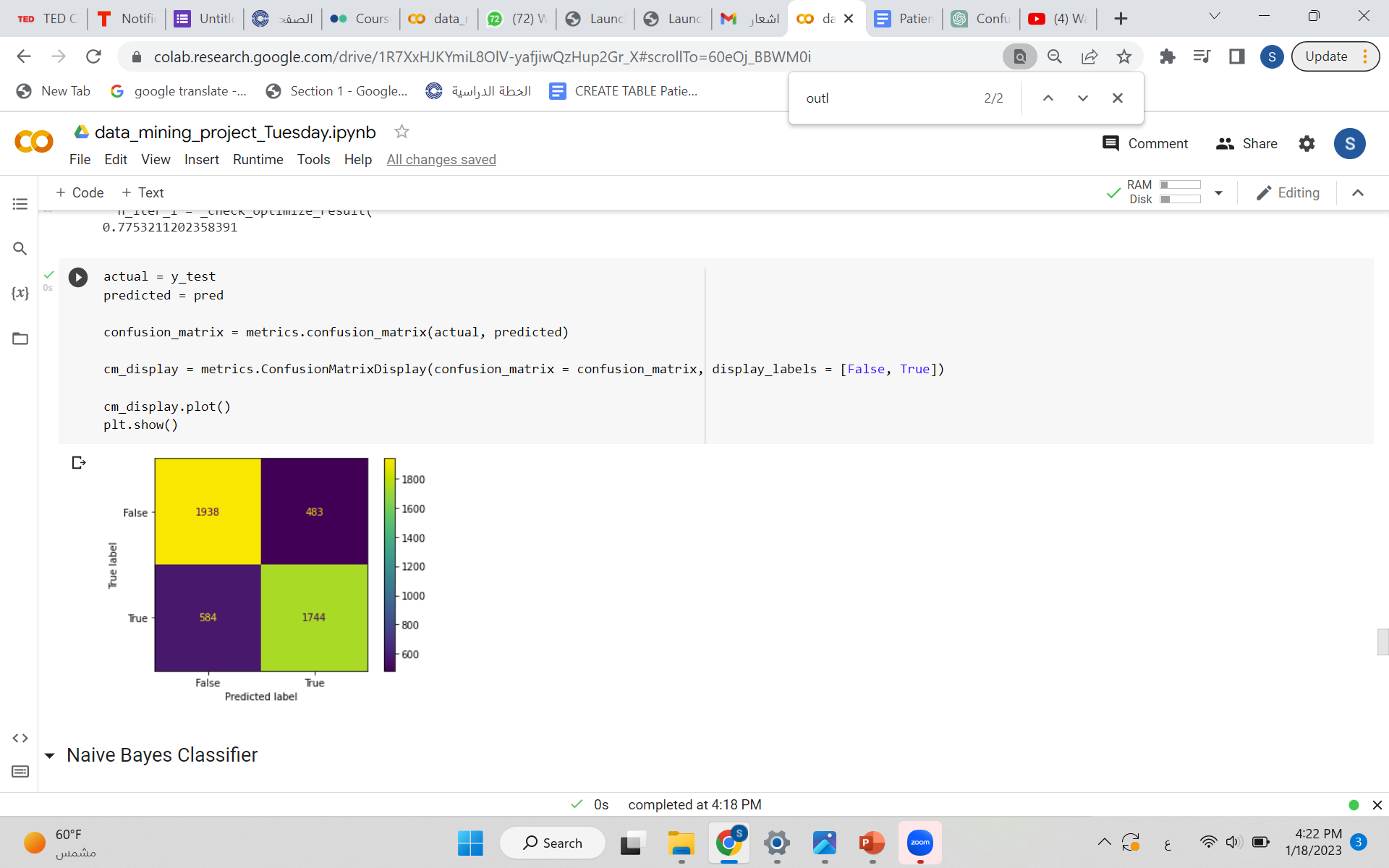
• k-Nearest Neighbors (KNN) Classifier:



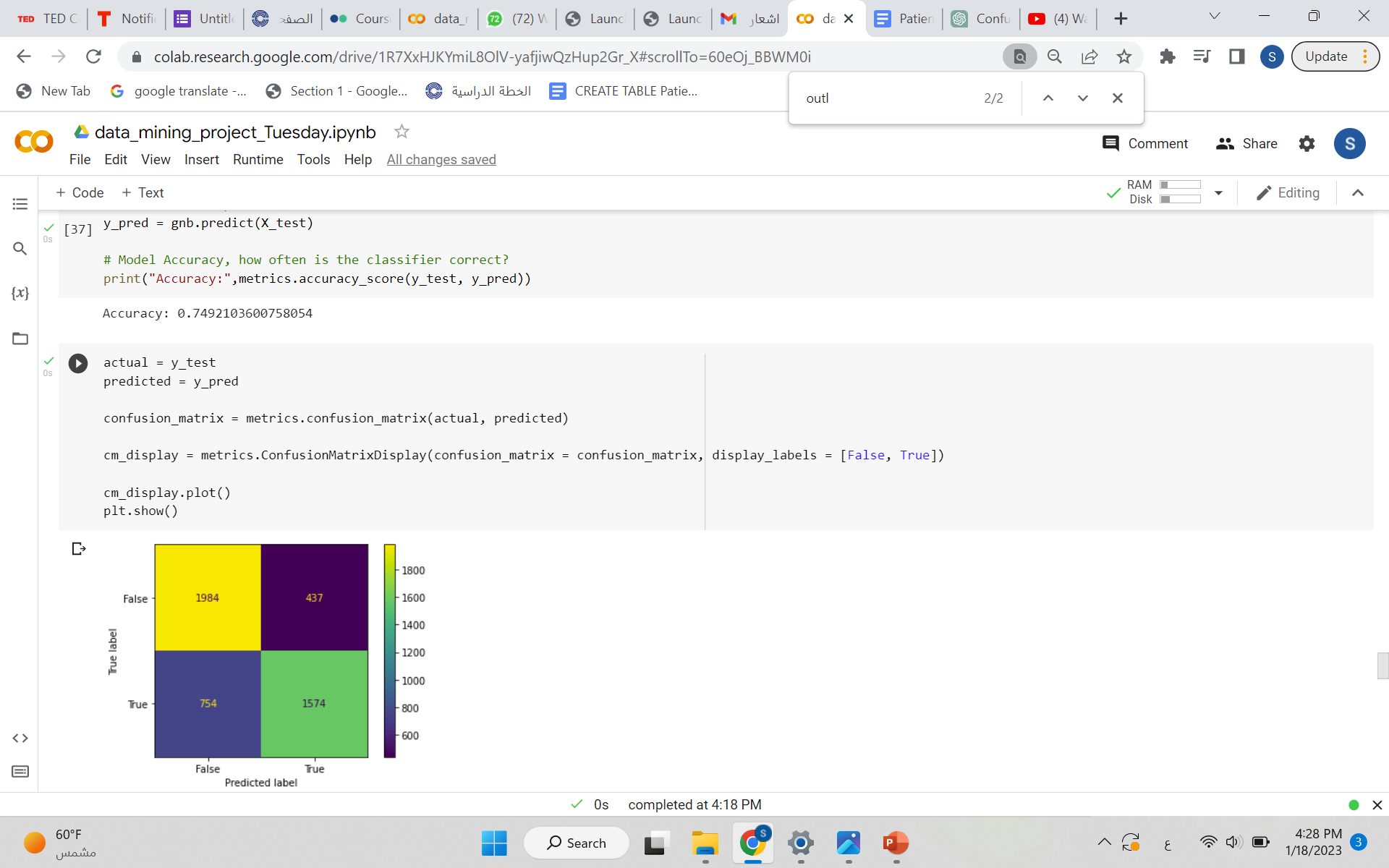


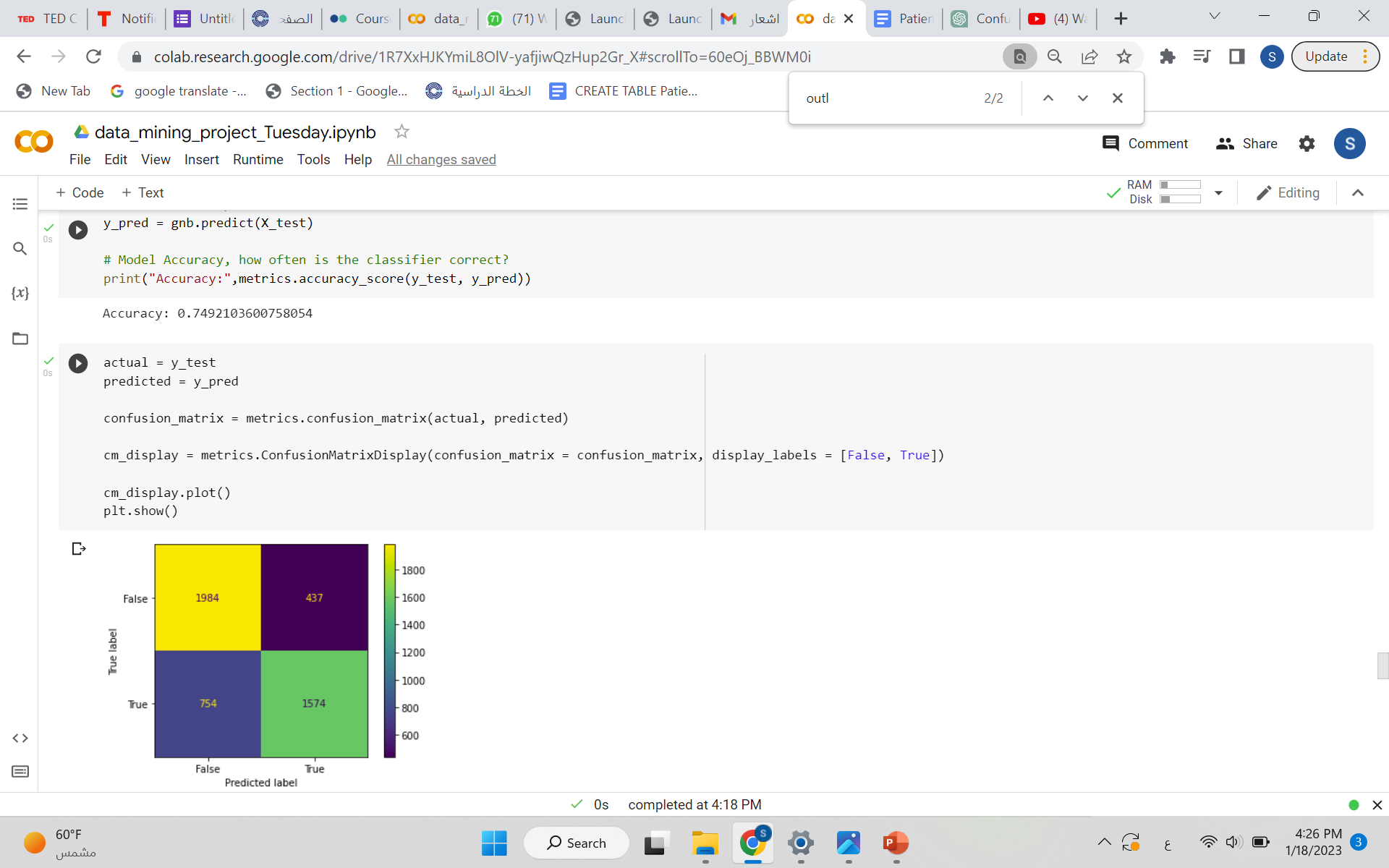
• Logistic Regression Classifier:



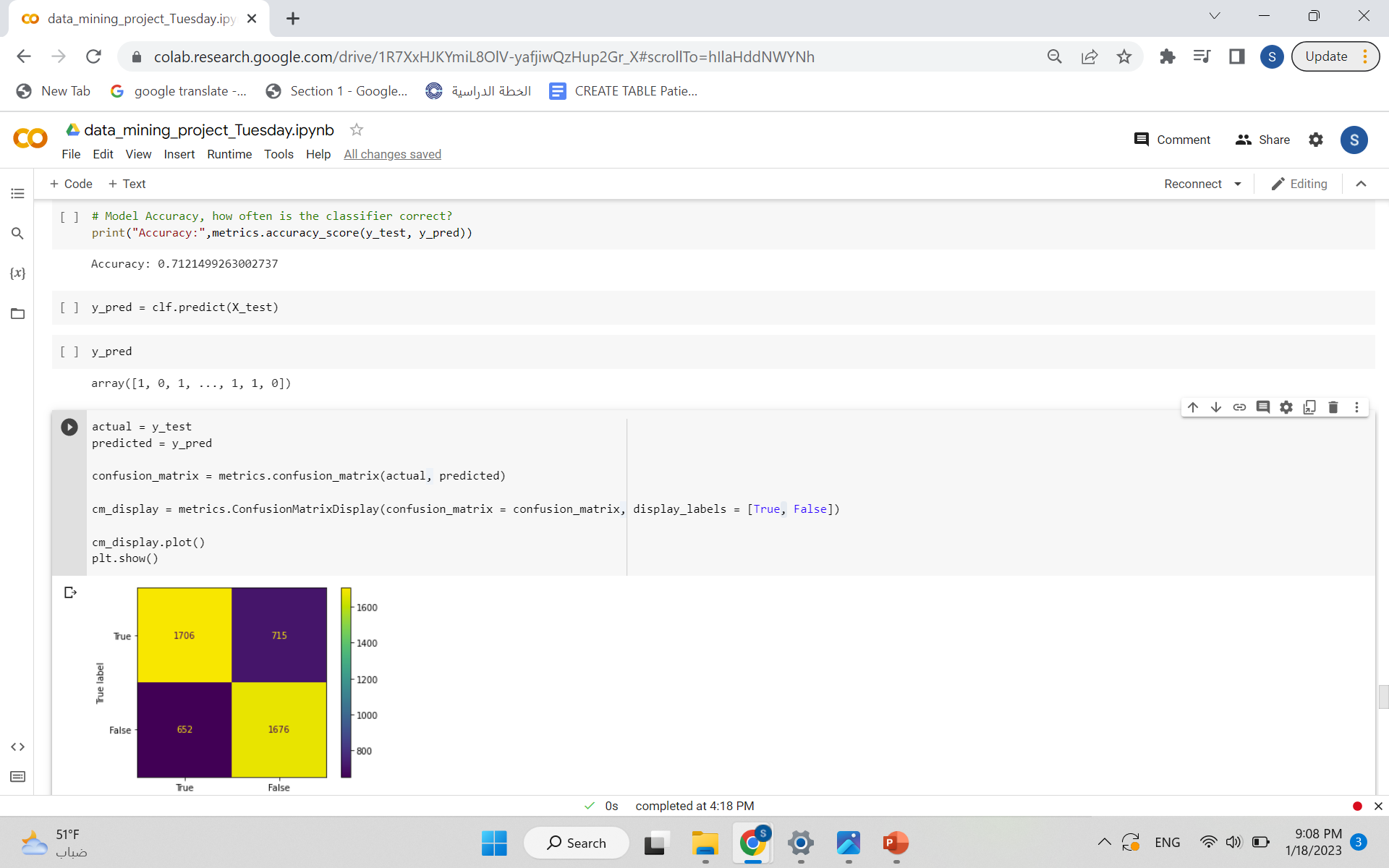


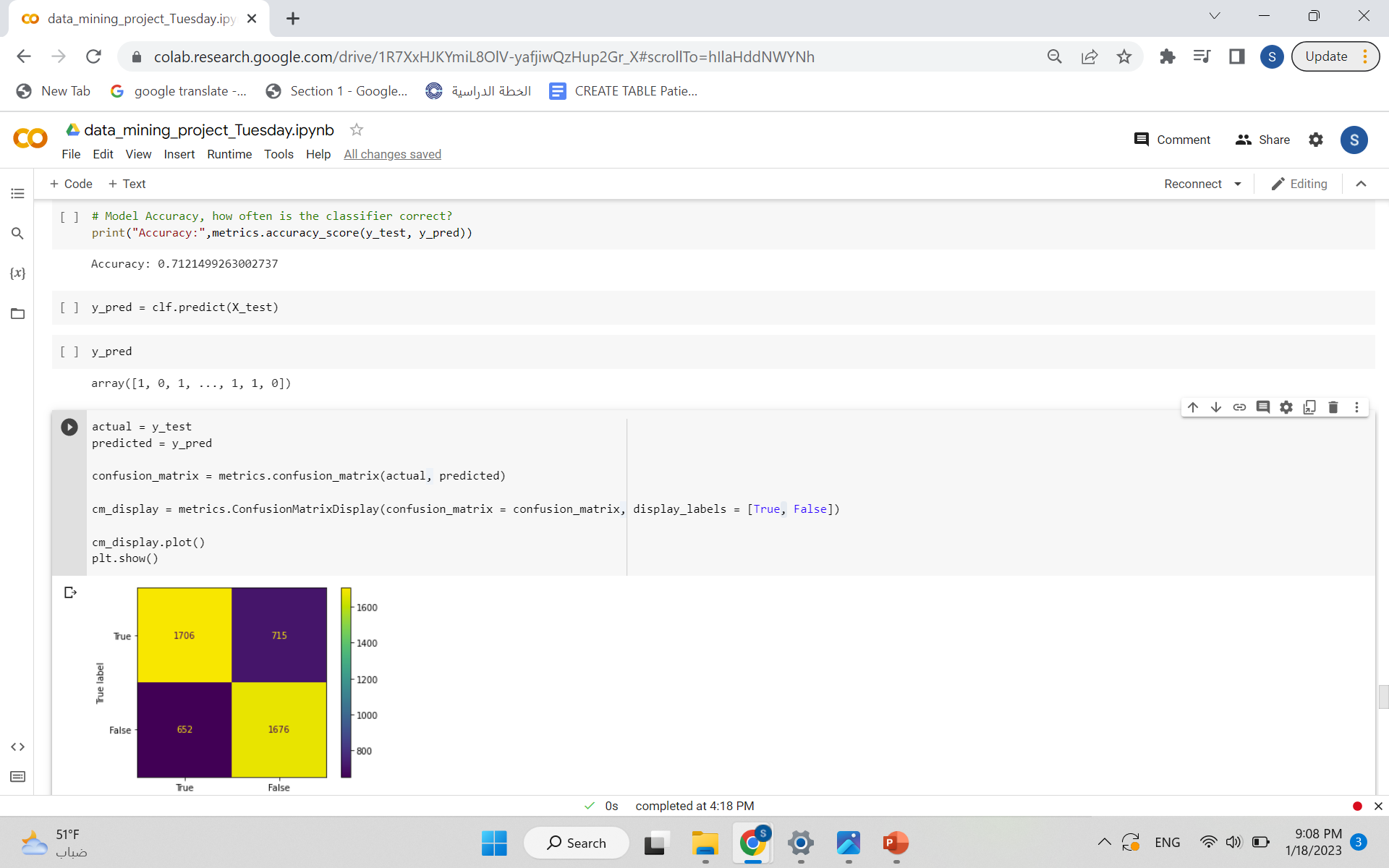
• Naive Bayes Classifier:



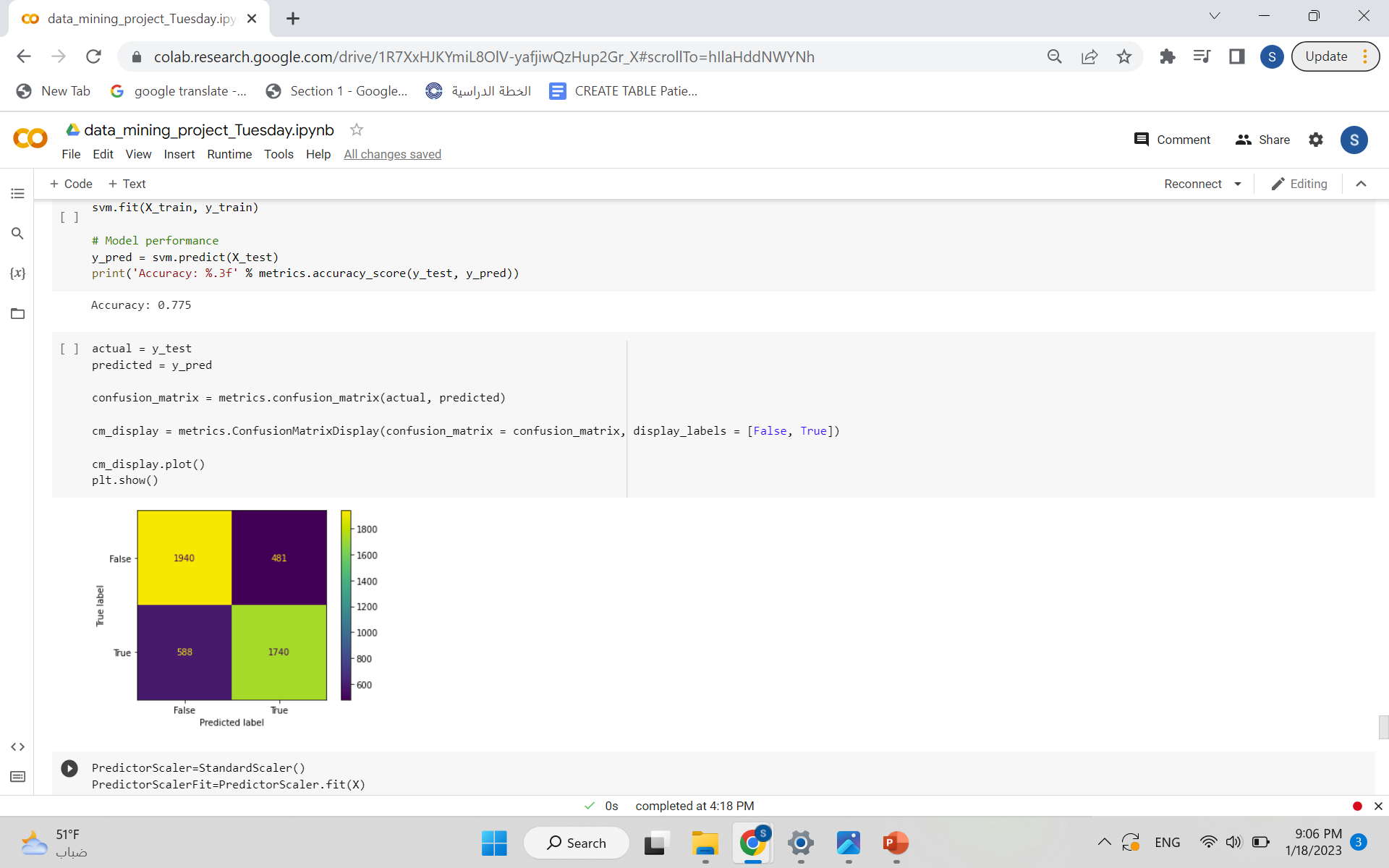


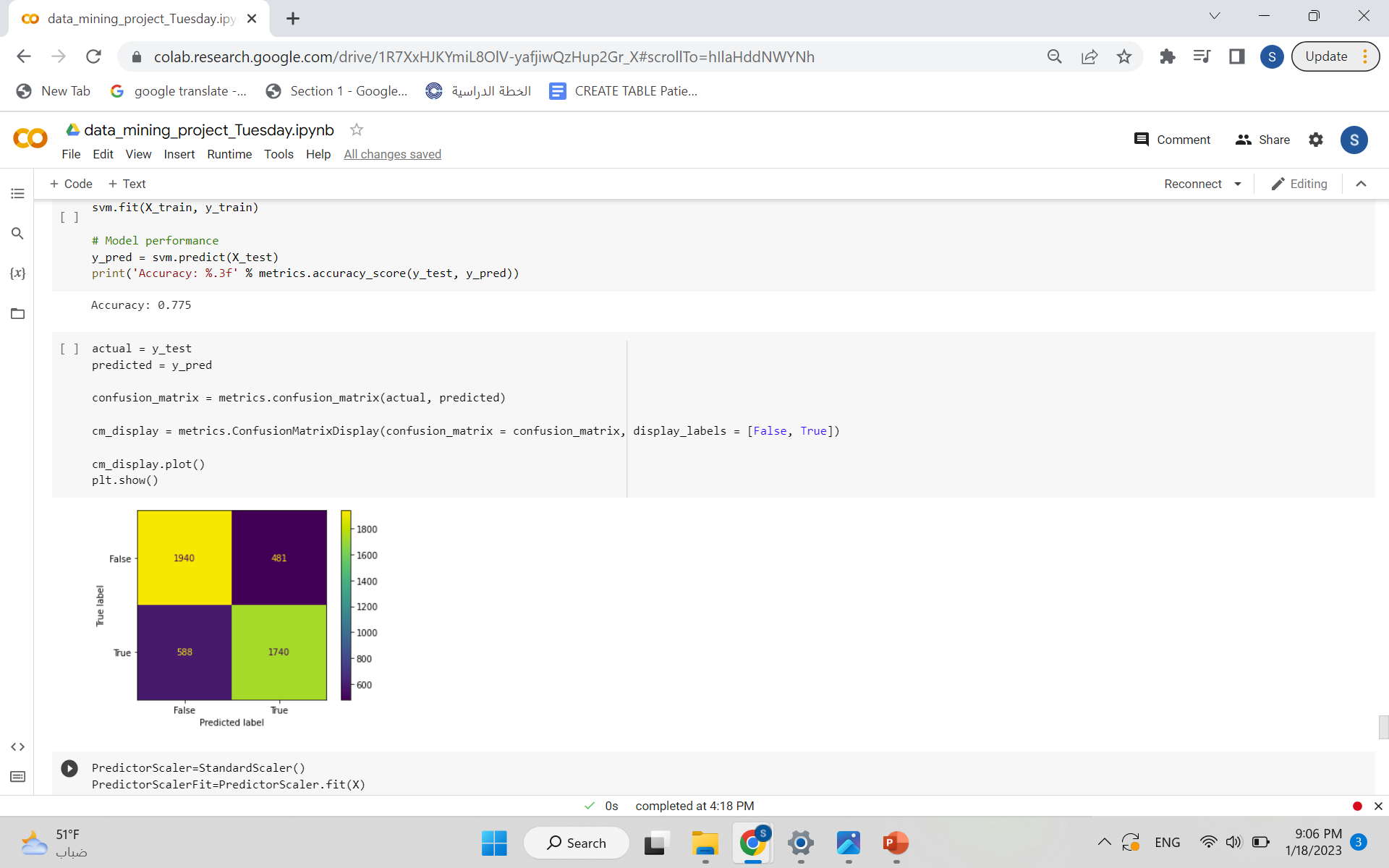
• Decision Tree Classifier:

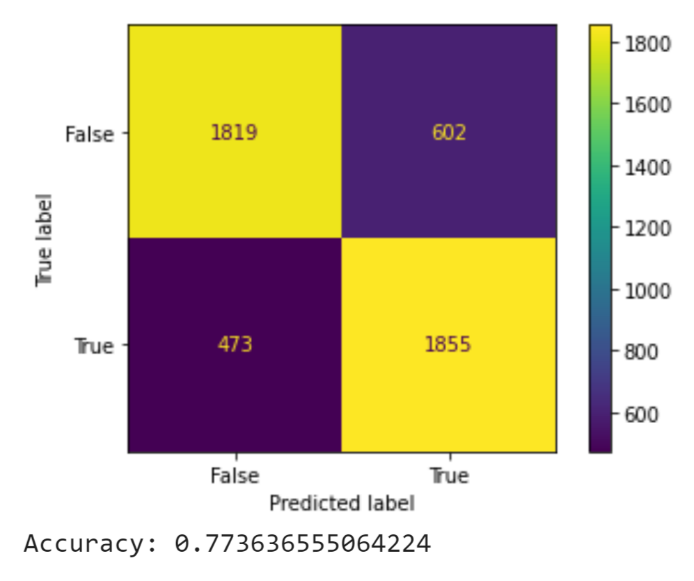




• Support Vector Machine Classifier:





* Artificial Neural Networks Classifier ANN

**Conclusion**

The results show that the XGBoost Classifier model achieved the highest overall performance among the eight models. As shown below:

ACCURACY OF THE MODEL: 0.7904822067803748

The lowest overall performance was the Decision Tree Classifier with an accuracy of: 0.7068856601389766

**References**

Dataset from Kaggle: <https://www.kaggle.com/datasets/mitishaagarwal/patient?resource=download>

<https://medium.com/grabngoinfo/four-oversampling-and-under-sampling-methods-for-imbalanced-classification-using-python-7304aedf9037>

<https://thinkingneuron.com/how-to-use-artificial-neural-networks-for-classification-in-python/>

<https://www.w3schools.com/python/python_ml_confusion_matrix.asp>

<https://www.w3schools.com/>

AI lab (PSUT).

ML lab (PSUT)