ARTIFICIAL INTELLIGENCE

**PROJECT**

**REPORT**

**Cancer**

Detection

Group Member: Faculty:

**Kainat Fatima (9413)   
Ali Sher Fazil (9411) Sir Siraj Munir**

Date: 22th April 2021

**Problem description**

In this modern world, we have different types of deceases around us. We hear lots of news about those deceases. In these all deceases, Cancel is very dangerous and many times it couldn’t be diagnosed early so patient have chances to suffer a lot and in the end, God forbid patient could have been died.

Although in this modern world, we have different type of mechanism to detect the cancer types. So I worked on this model and try to figure out that what type of cancer is this.

This assignment is bring us to classify the type of cancer by recognizing the various features and classify it in to 2 classes. I performed classification by using three models Random Forest, Decision Tree and Logistic Regression. For all these working, we use python as language and colab.research.google.com as our tool to classify and get dataset and working on it. I successfully classify the cancer type and successfully train our model to classify it.

**Use Case**

**Admin**

Admin will gather all dataset regarding cancer types and label them. He is responsible to prepare a proper excel or csv file so we can use it on future when it’s needed.

Also he can delete or do feature engineering to maximize the accuracy of dataset and retain all necessary attributes and delete all unnecessary data from dataset.

He can choose any model or machine learning algorithm to train machine by giving prepared dataset and examine that which one is best for this given dataset. He can check the model accuracy as well as check the precision and recall.

**Tester**

Tester can gather any of the data about cancer and choose any of the model which are chosen by Admin and test it that model can predict on new data or that. And if model can predict then what’s the accuracy of it.

**Doctors, Patient**

After developed the system, Doctor or patient use this system to recognize the type of cancer. They can give the new data and machine will tell them that what type of cancer is this.

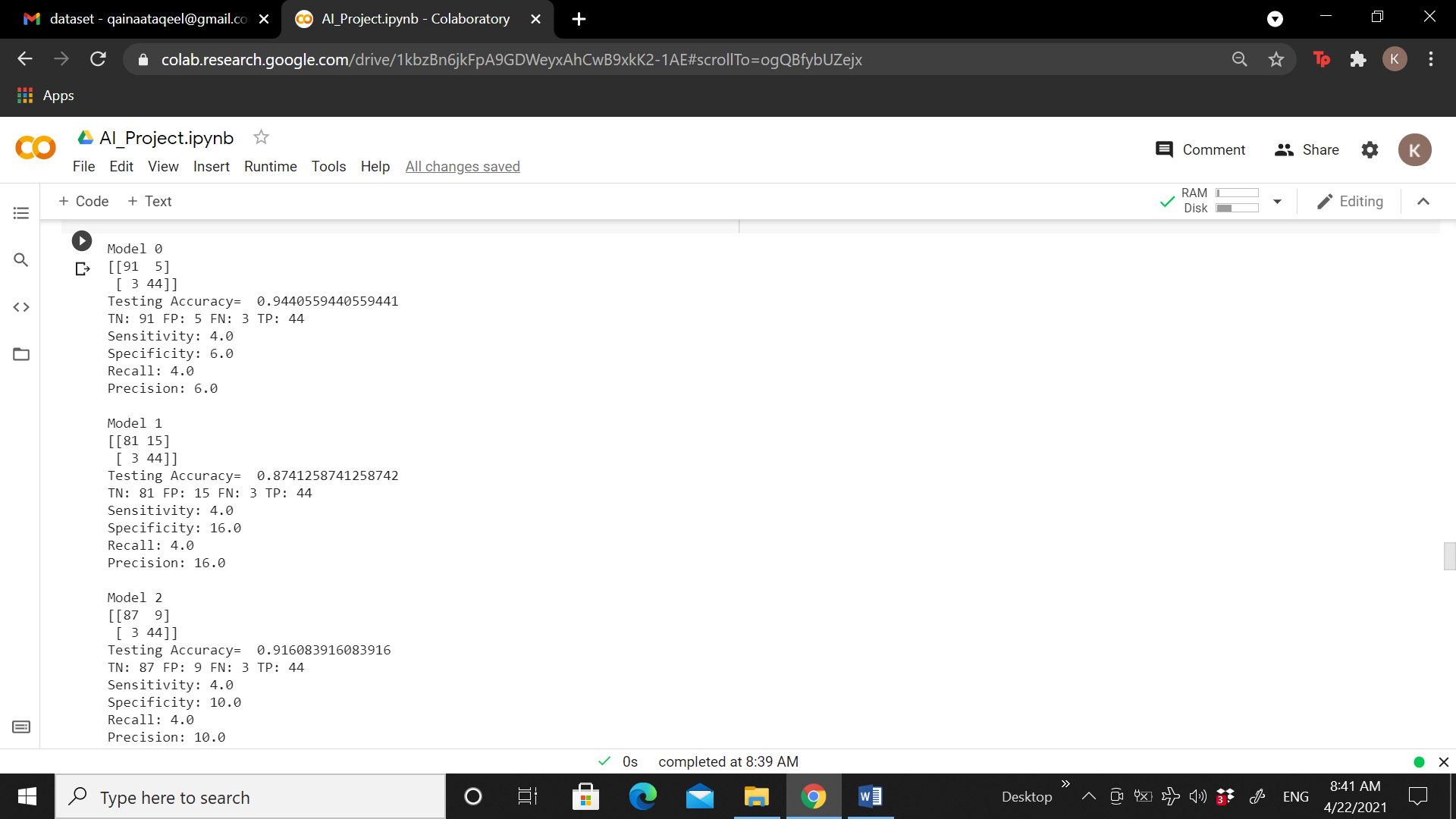
Network Security

Developer

Tester

**Accuracy**

Overall performance of all models which we have used in this system.  
Model 2 is Random forest.



**Methods and Procedures**

1- Import all necessary libraries

2- Import Dataset

3- All necessary steps of features engineering if needed.

4- Split data in to x and y for input and output.

5- Split data for training and testing. (Usually up to 75% of data for training)

6- Train the model on training data using one of the following:

1. **Random Forest**

We have a big dataset so we choose random forest as our training model. Because Random Forest choose the data from data set randomly like some trees in forest and train them and at last, voting will be done that how many result is in maximum number.

1. **Decision Tree**

Decision trees provide an effective method of Decision Making because they: Clearly lay out the problem so that all options can be challenged. Allow us to analyze fully the possible consequences of a decision. Provide a framework to quantify the values of outcomes and the probabilities of achieving them.

1. **Logistic Regression**

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

7- Check the accuracy on test data with actual result and predicted result.

8- Construct the confusion matrix for detailed result and performance.

**Code**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from google.colab import files

upl=files.upload()

data=pd.read\_csv("data.csv")

data.head(5)

data.shape

data.isna().sum()

data=data.dropna(axis=1)

data.shape

data['diagnosis'].value\_counts()

sns.countplot(data['diagnosis'],label='count')

from sklearn.preprocessing import LabelEncoder

labelEncoder\_Y=LabelEncoder()

data.iloc[:,1]=labelEncoder\_Y.fit\_transform(data.iloc[:,1].values)

data.iloc[:,1]

sns.pairplot(data.iloc[:,1:5], hue="diagnosis")

plt.figure(figsize=(10,10))

#sns.heatmap(data.iloc[:,1:12].corr(), annot=True)

sns.heatmap(data.iloc[:,1:12].corr(), annot=True ,fmt='.0%')

X=data.iloc[:,2:31].values

Y=data.iloc[:,1].values

type(data)

**Splitting data into testing and training**

from sklearn.model\_selection import train\_test\_split

trainx, testx ,trainy, testy = train\_test\_split(X,Y,test\_size=0.25)

from sklearn.preprocessing import StandardScaler

sc=StandardScaler()

trainx=sc.fit\_transform(trainx)

testx=sc.fit\_transform(testx)

trainx

**Random Forest**

from sklearn.ensemble import RandomForestClassifier

  RF=RandomForestClassifier(n\_estimators=10,criterion="entropy",random\_state=0)

  RF.fit(trainx,trainy)

**Decision Tree**

from sklearn.tree import DecisionTreeClassifier

  DT=DecisionTreeClassifier(criterion="entropy", random\_state=0)

  DT.fit(trainx,trainy)

**Logistic Regression**

def models(trainx,trainy):

  from sklearn.linear\_model import LogisticRegression

  log=LogisticRegression(random\_state=0)

  log.fit(trainx,trainy)

**Which one is better ?**

print('[0]Logistic Regression Training Accuracy:' ,log.score(trainx,trainy))

  print('[1]Decision Tree Classifier Training Accuracy:' ,DT.score(trainx,trainy))

  print('[2]Random Forest Classifier Training Accuracy:' ,RF.score(trainx,trainy))

  return log, DT , RF

**Confusion Matrix**

from sklearn.metrics import confusion\_matrix

for i in range(len(model)) :

  print('Model', i)

  cm=confusion\_matrix(testy,model[i].predict(testx))

  TP=cm[0][0]

  TN=cm[1][1]

  FN=cm[1][0]

  FP=cm[0][1]

  print(cm)

  test\_accuracy=(TP+TN)/(TP+TN+FN+FP)

  print('Testing Accuracy= ',test\_accuracy)

  tn,fp,fn,tp=confusion\_matrix(testy,model[i].predict(testx)).ravel()

  print("TN:",tn,"FP:",fp,"FN:",fn,"TP:",tp)

  sensitivity=tp/tp+fn

  print("Sensitivity:",sensitivity)

  spec=tn/tn+fp

  print("Specificity:",spec)

  recall=tp/tp+fn

  print("Recall:",recall)

  Prec=tp/tp+fp

  print("Precision:",Prec)

  print()

**Model 2(Random forest) better working:**

pred=model[2].predict(testx)

print(pred)

print()

print(testy)

for i in range(len(model)) :

  print('Model', i)

  y\_pred = model[i].predict(testx)

  cm = confusion\_matrix(testy, y\_pred)

  df\_cm = pd.DataFrame(cm, range(2),

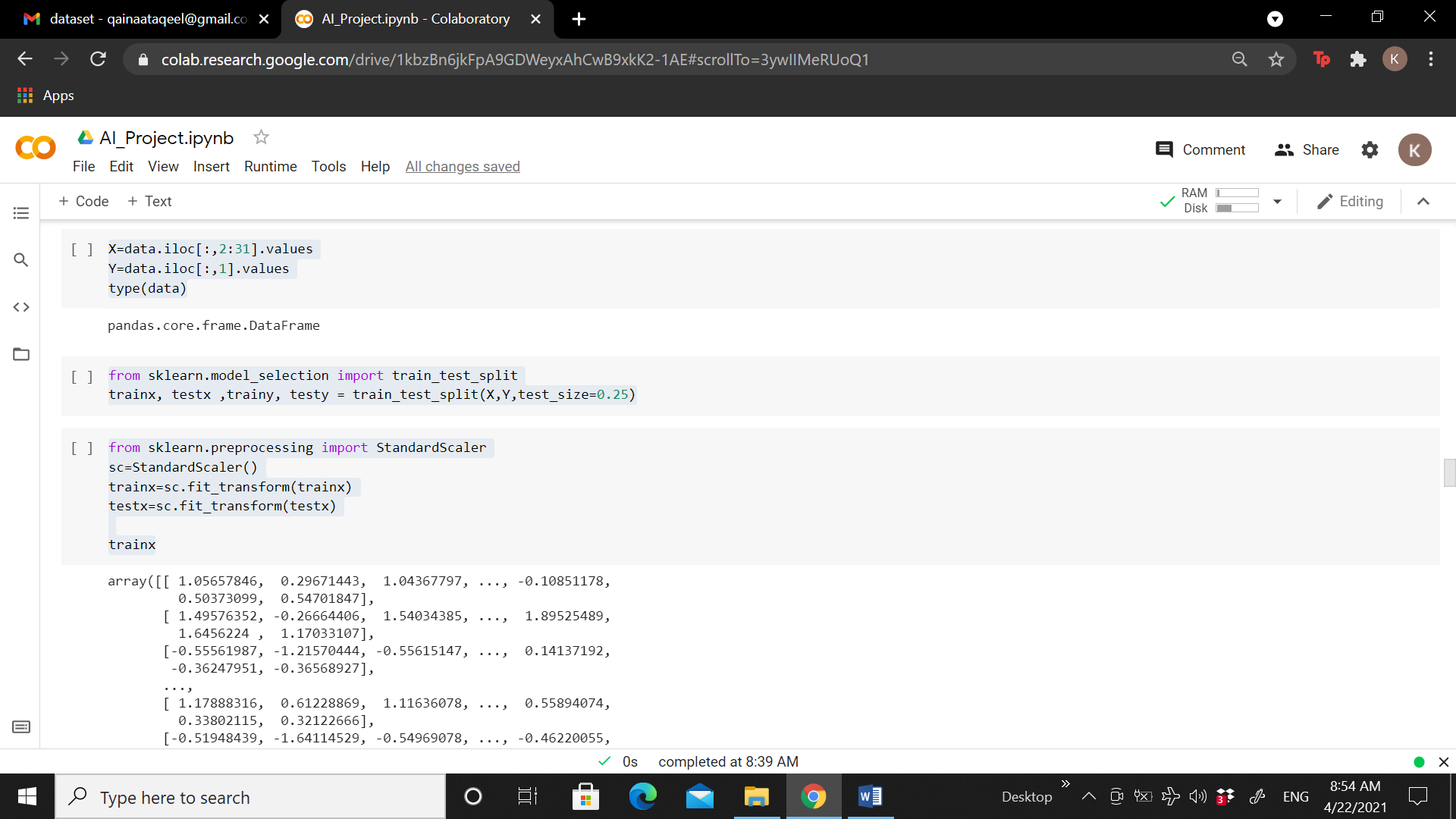
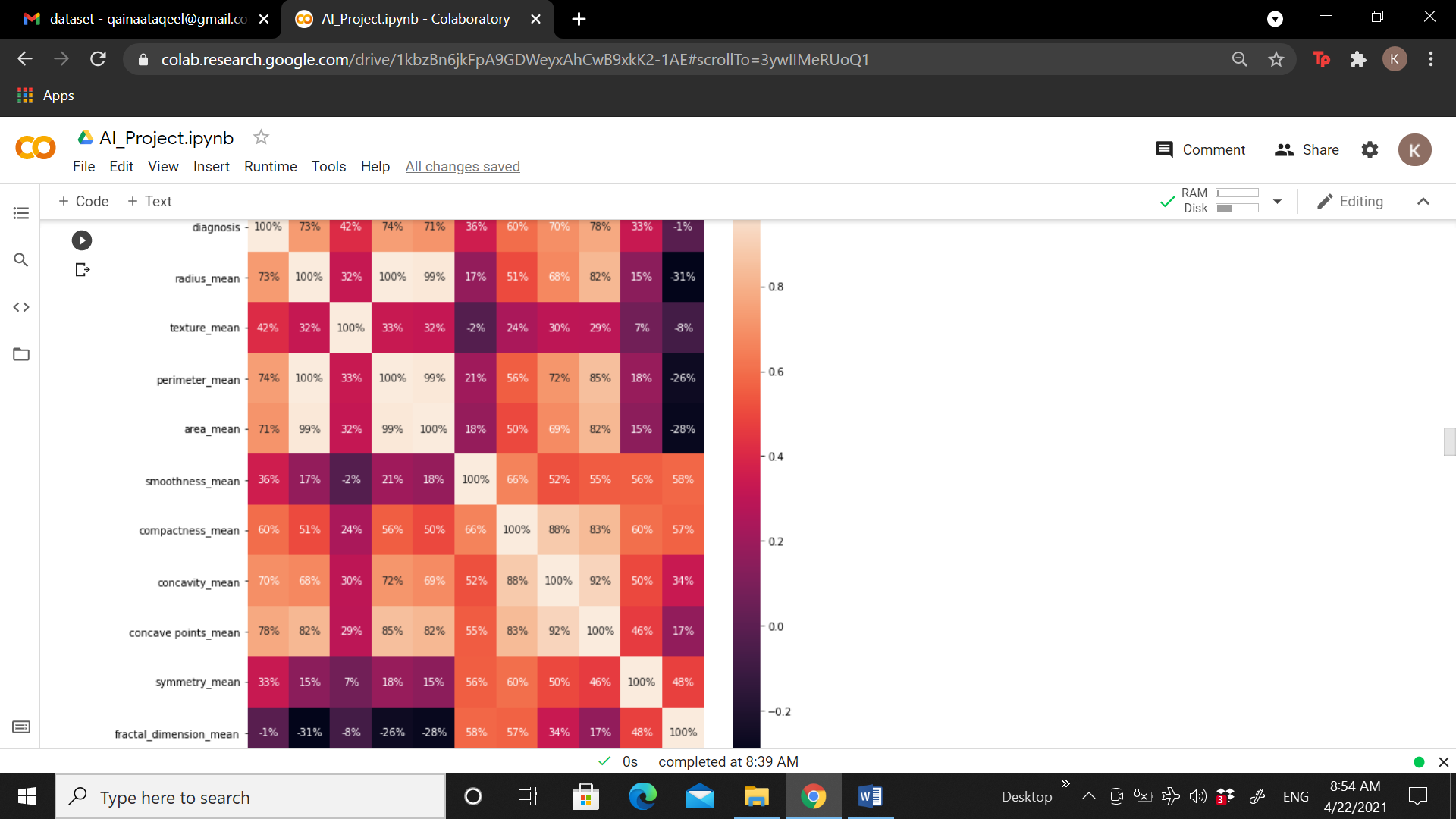
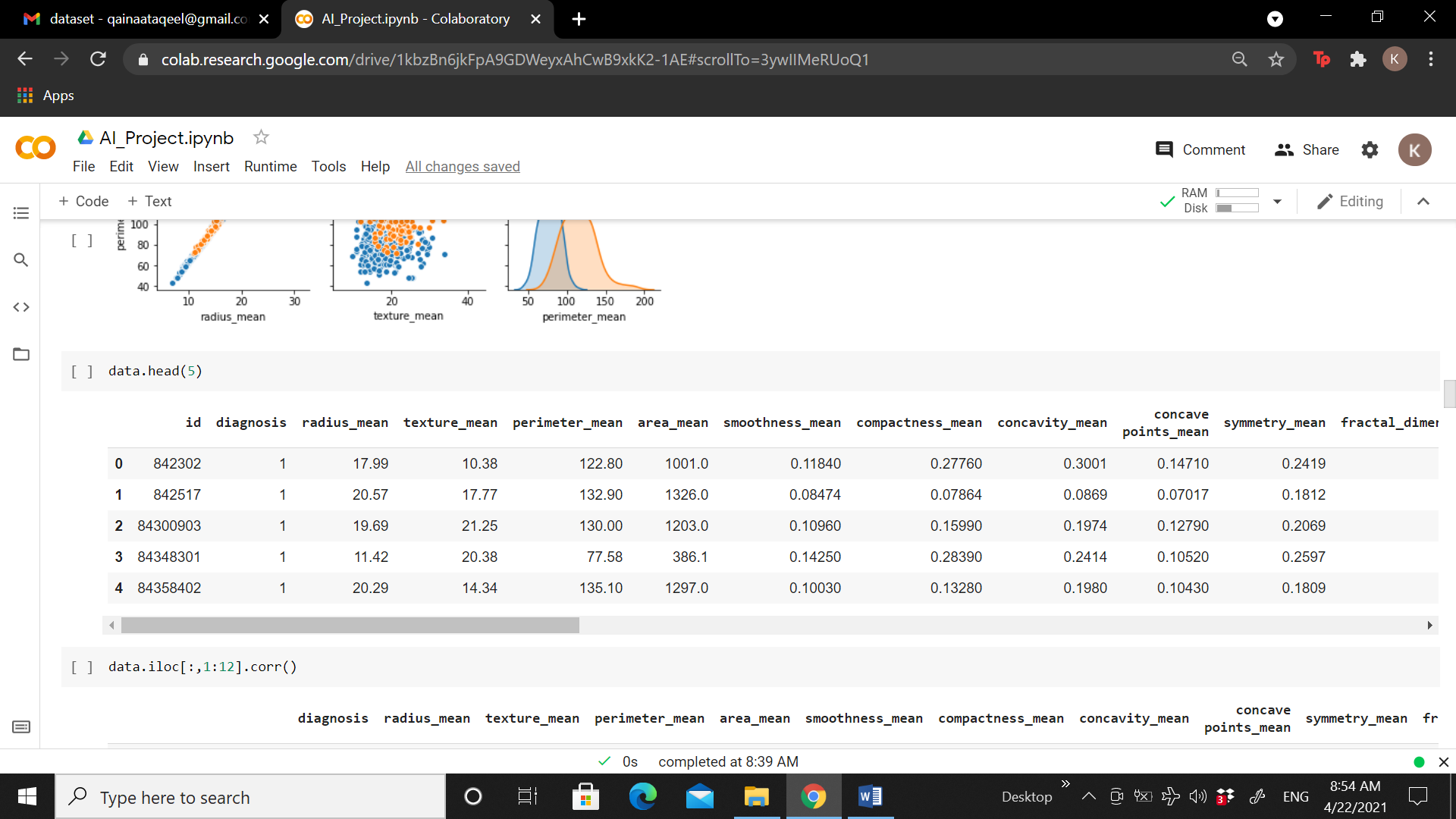
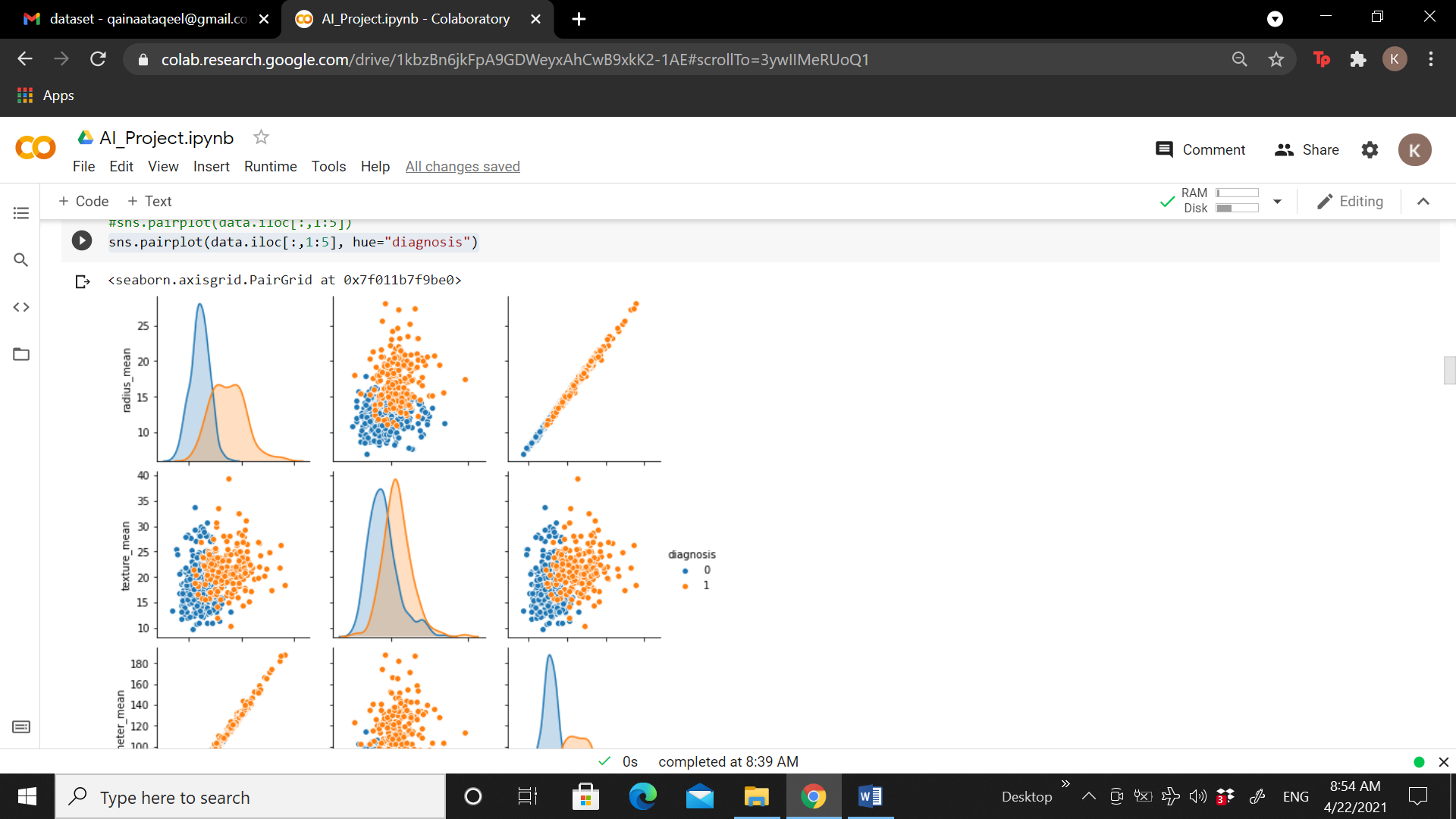
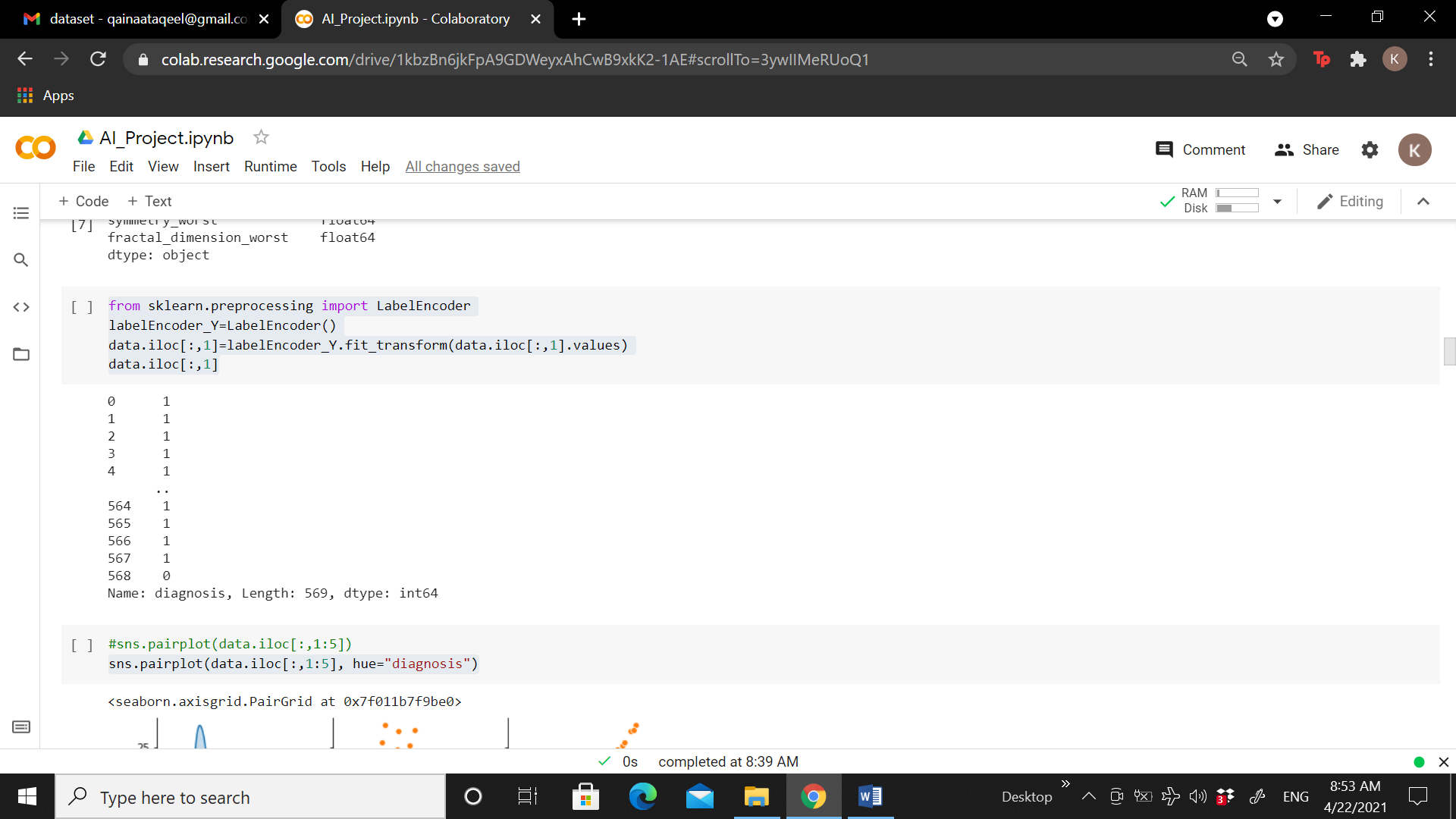
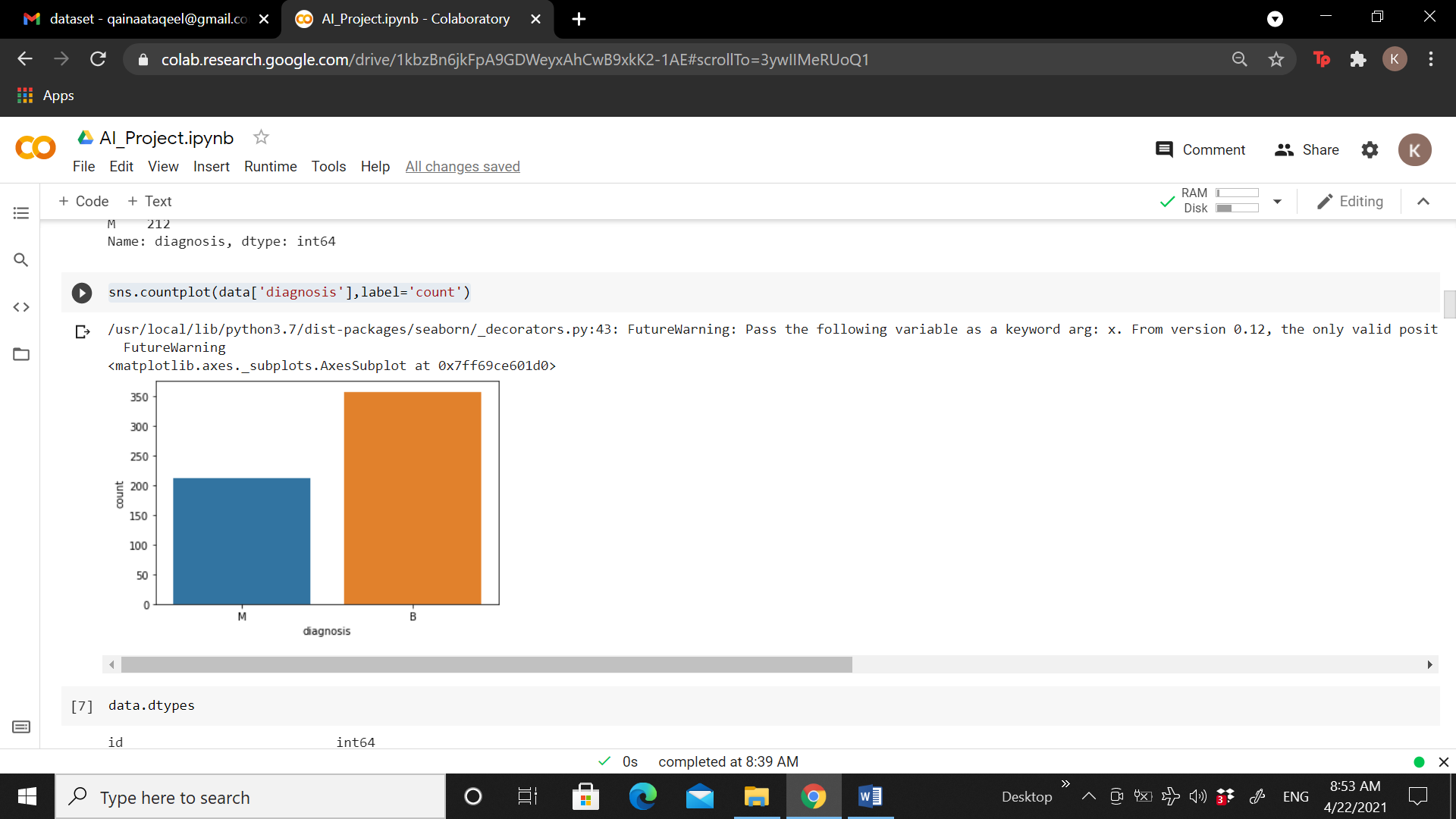
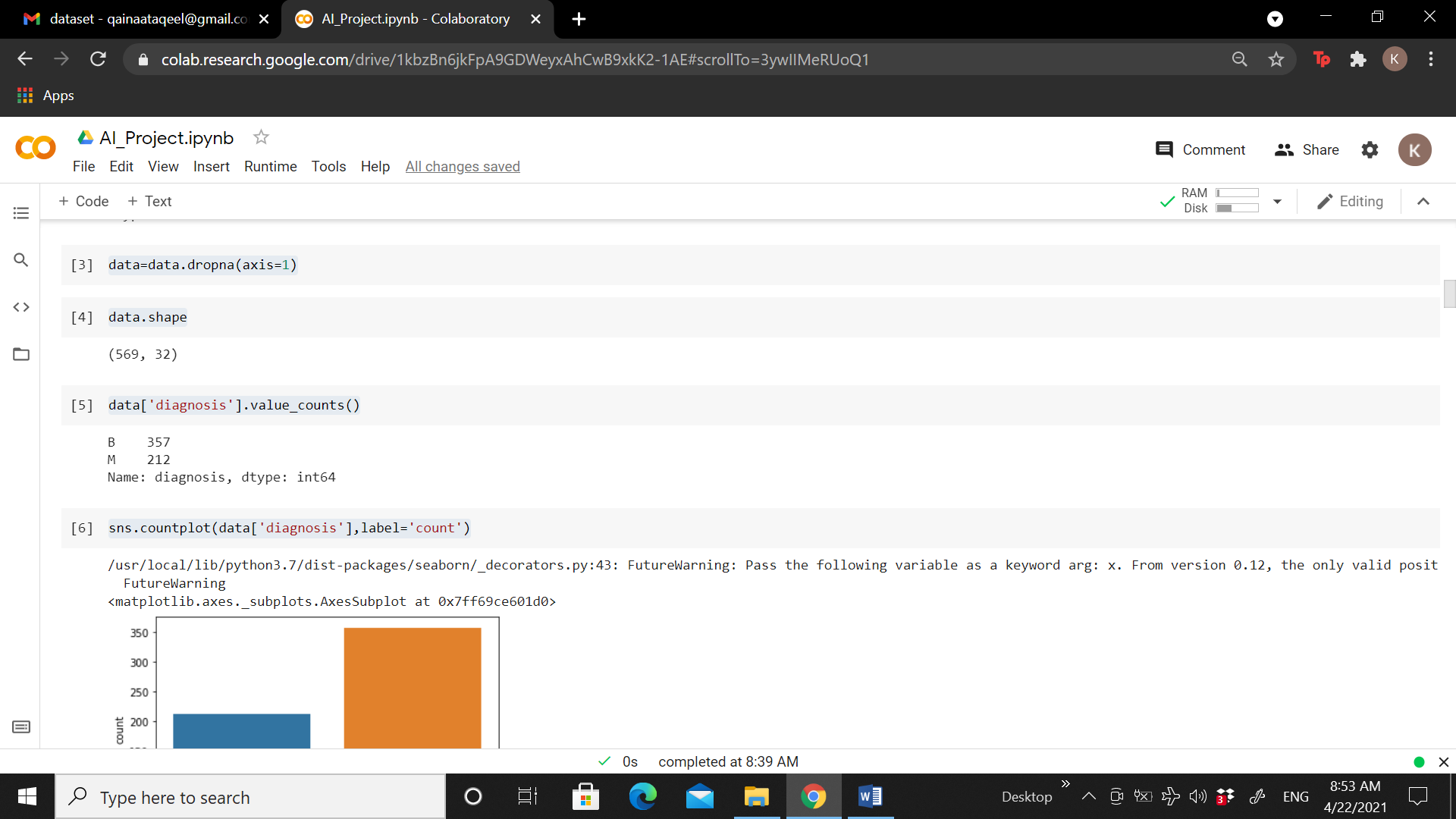
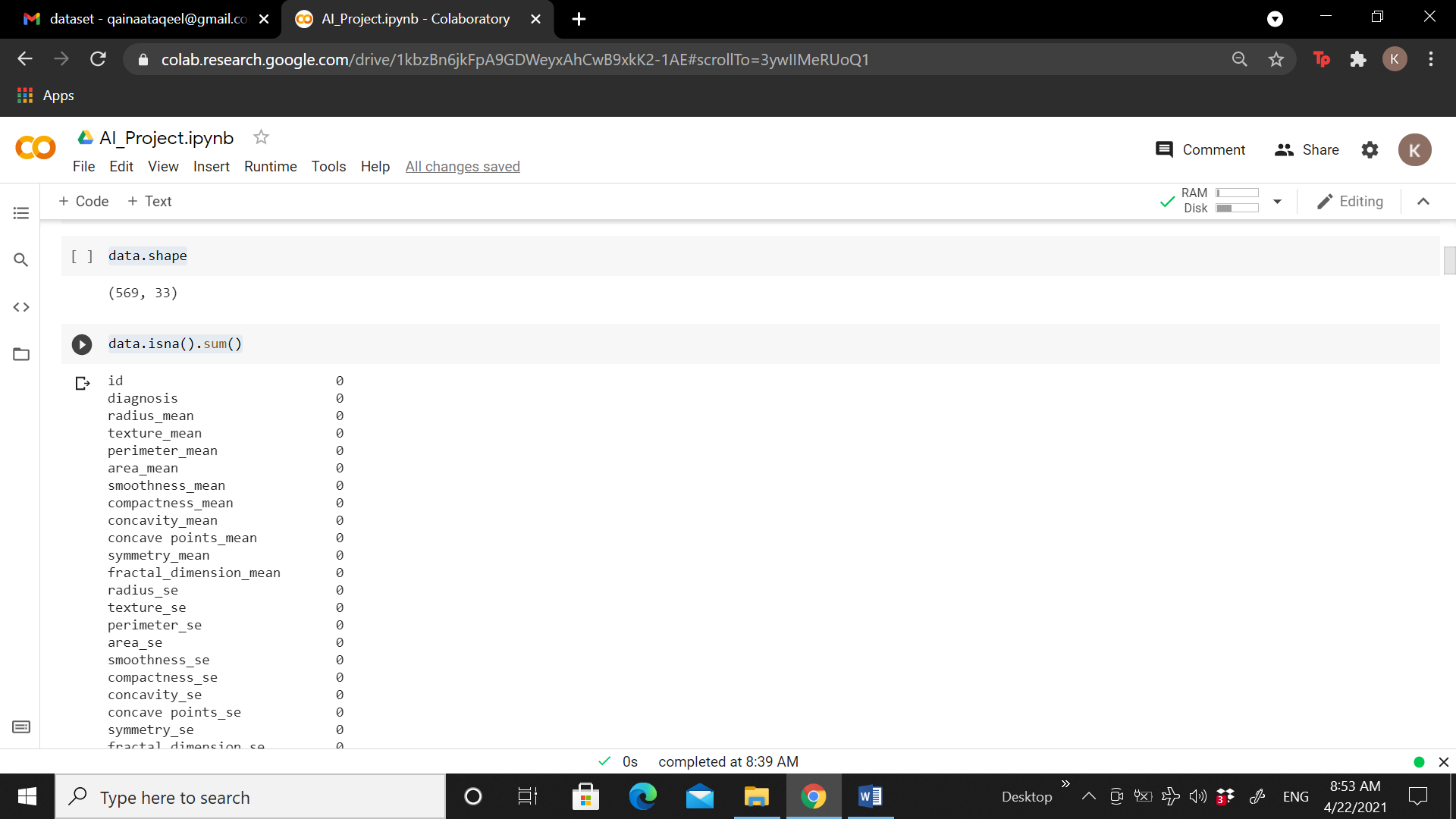
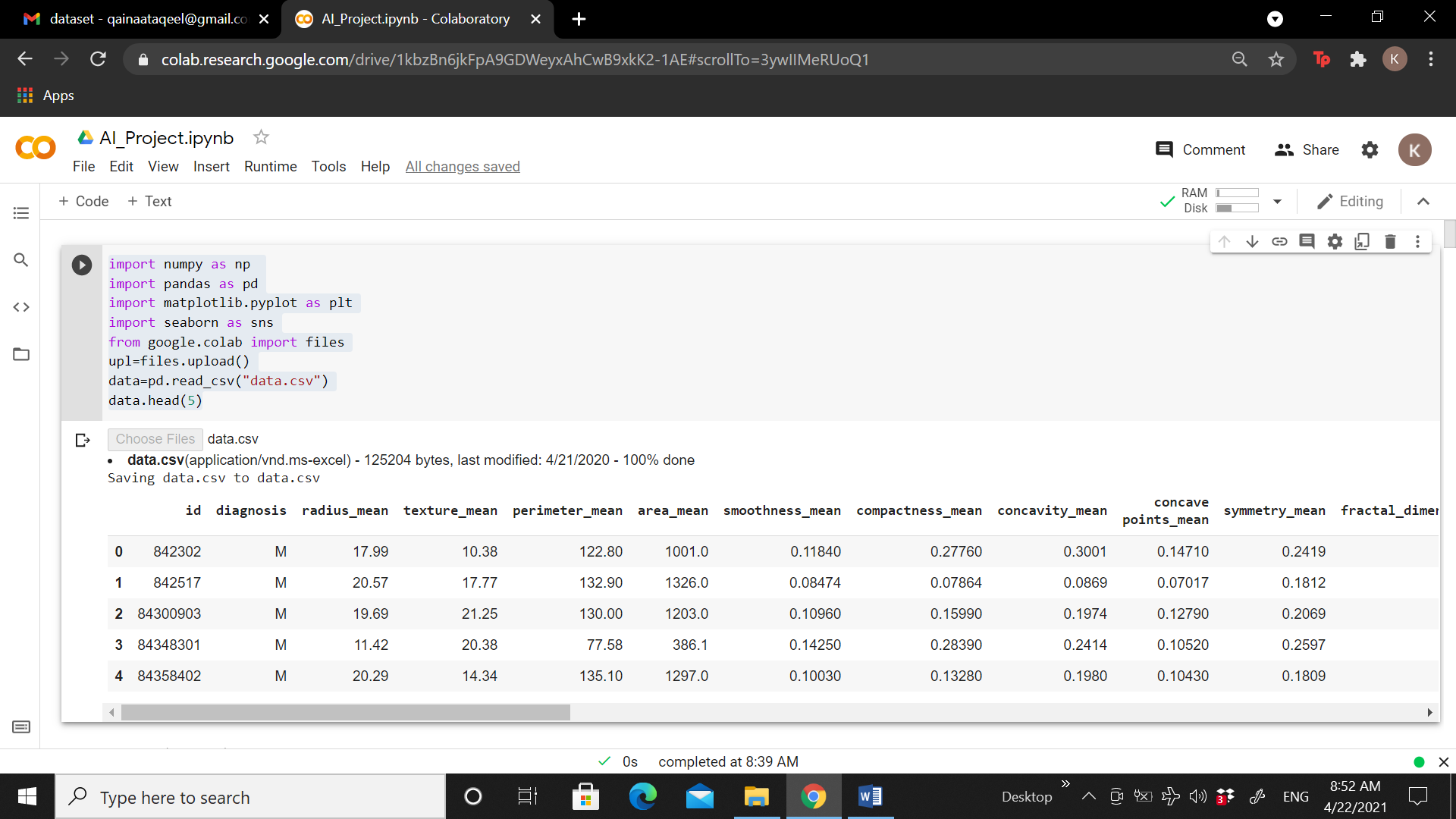
                  range(2))

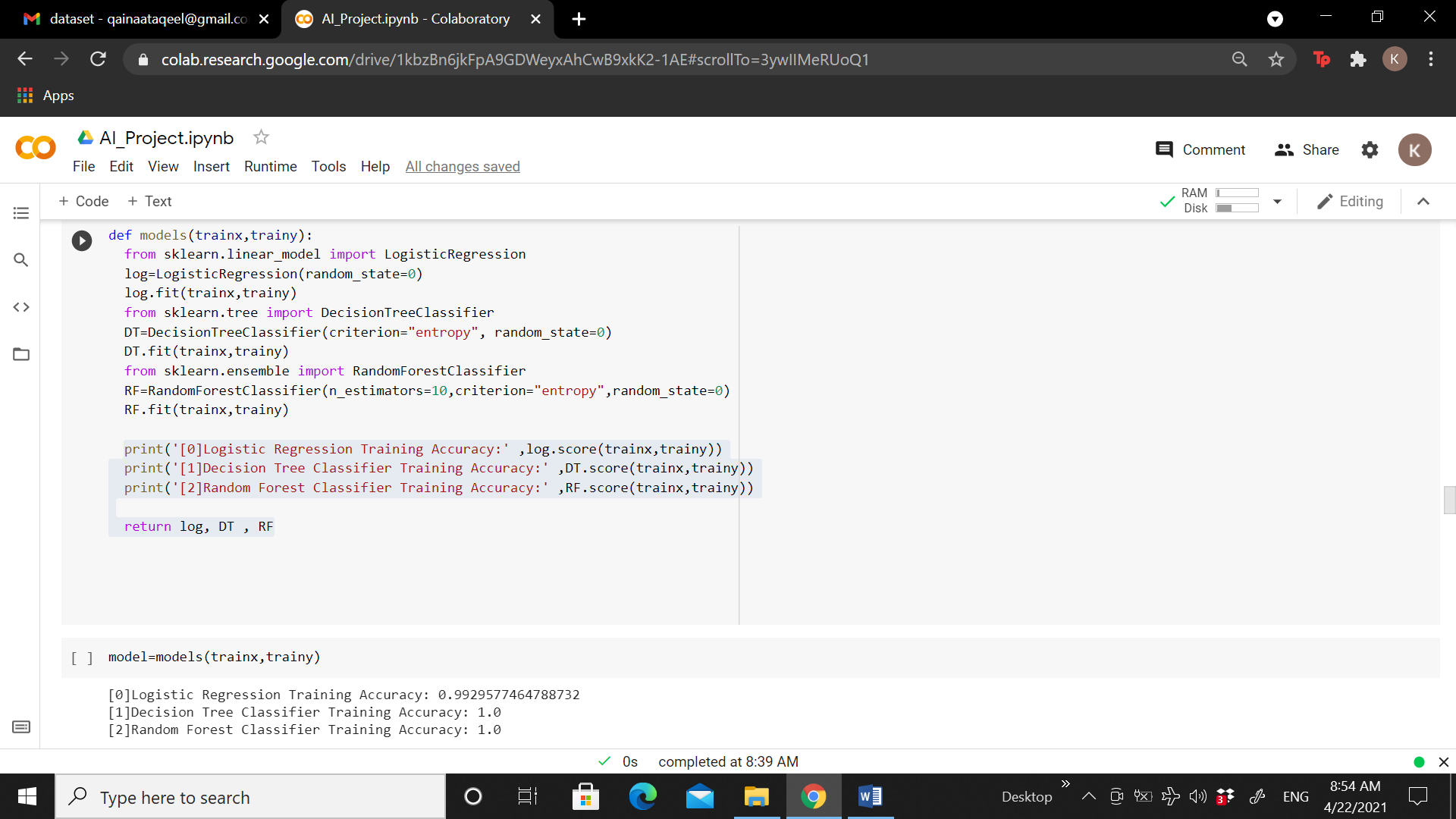
  plt.figure(figsize=(10,7))

  sns.set(font\_scale=1.4)#for label size

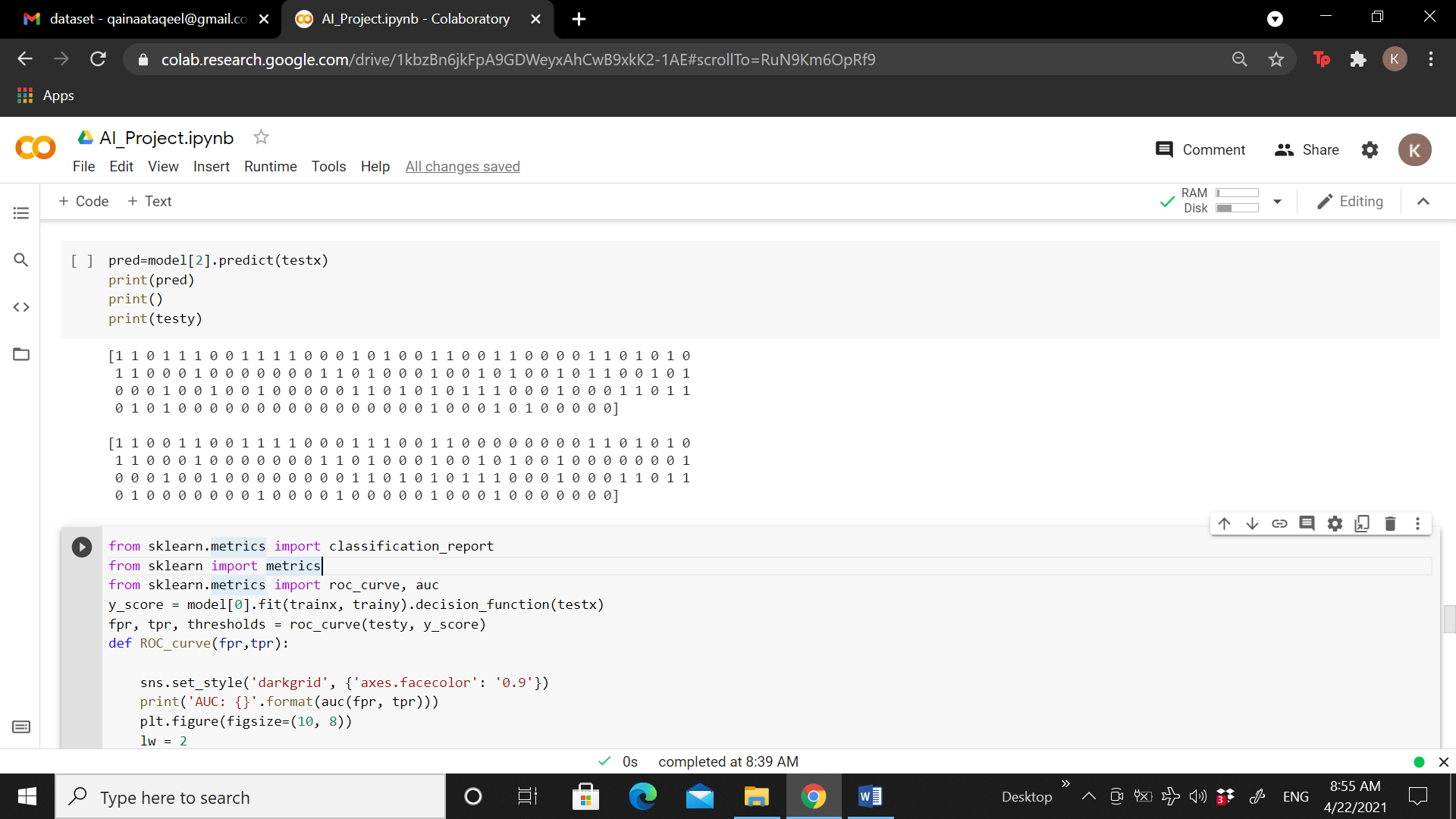
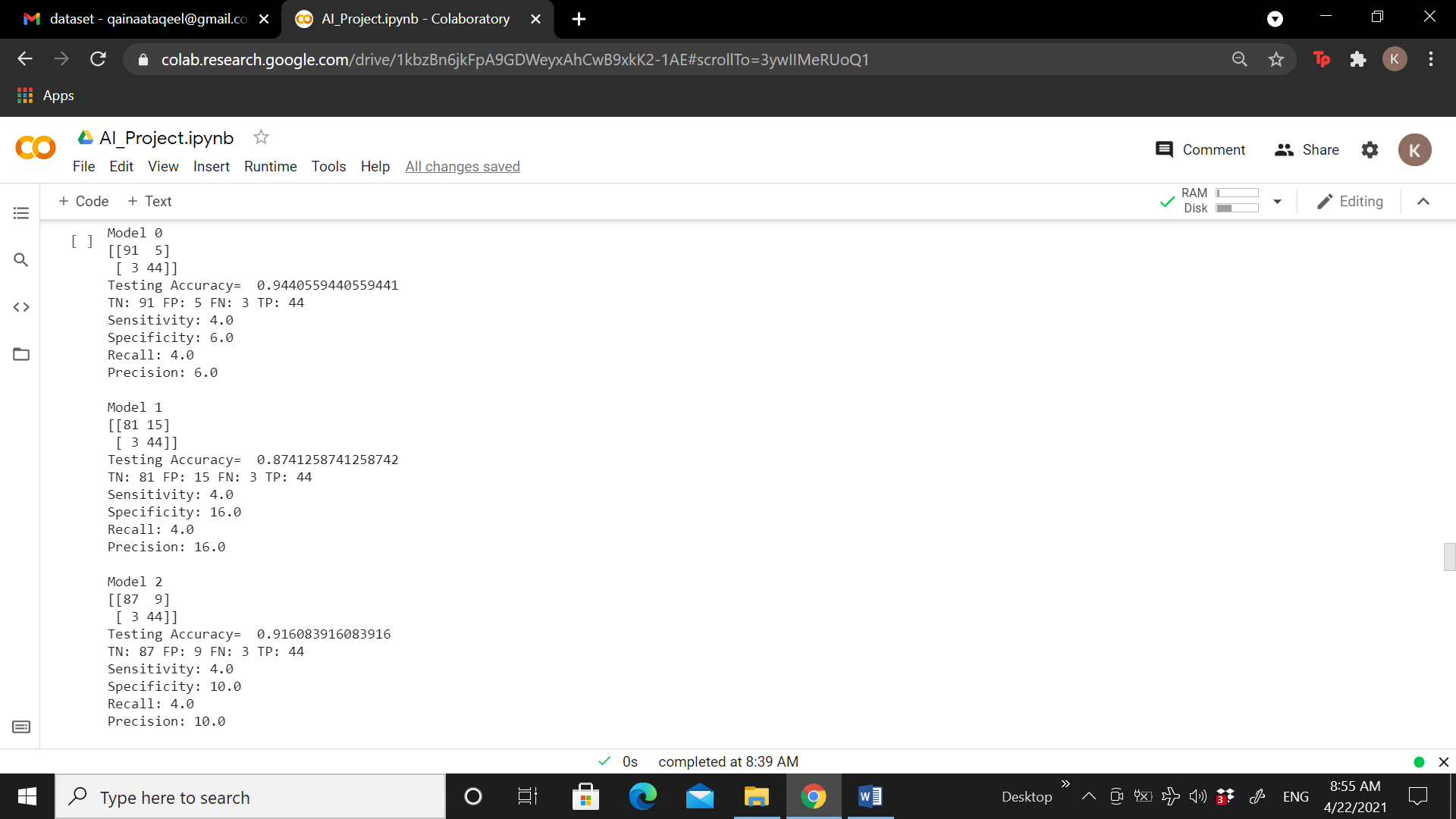
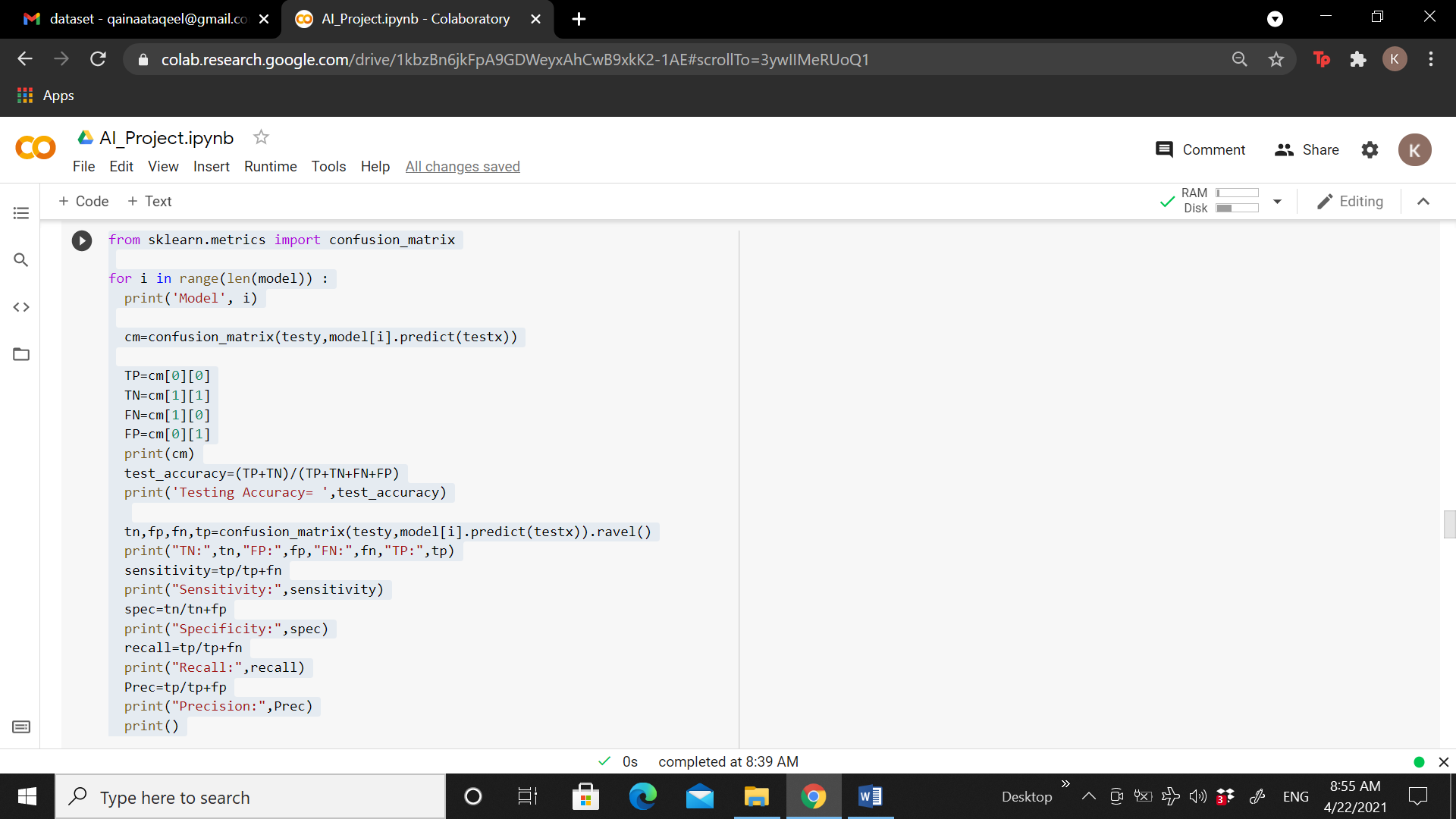
  cm\_plot = sns.heatmap(df\_cm, annot=True, fmt='n', annot\_kws={"size": 10})

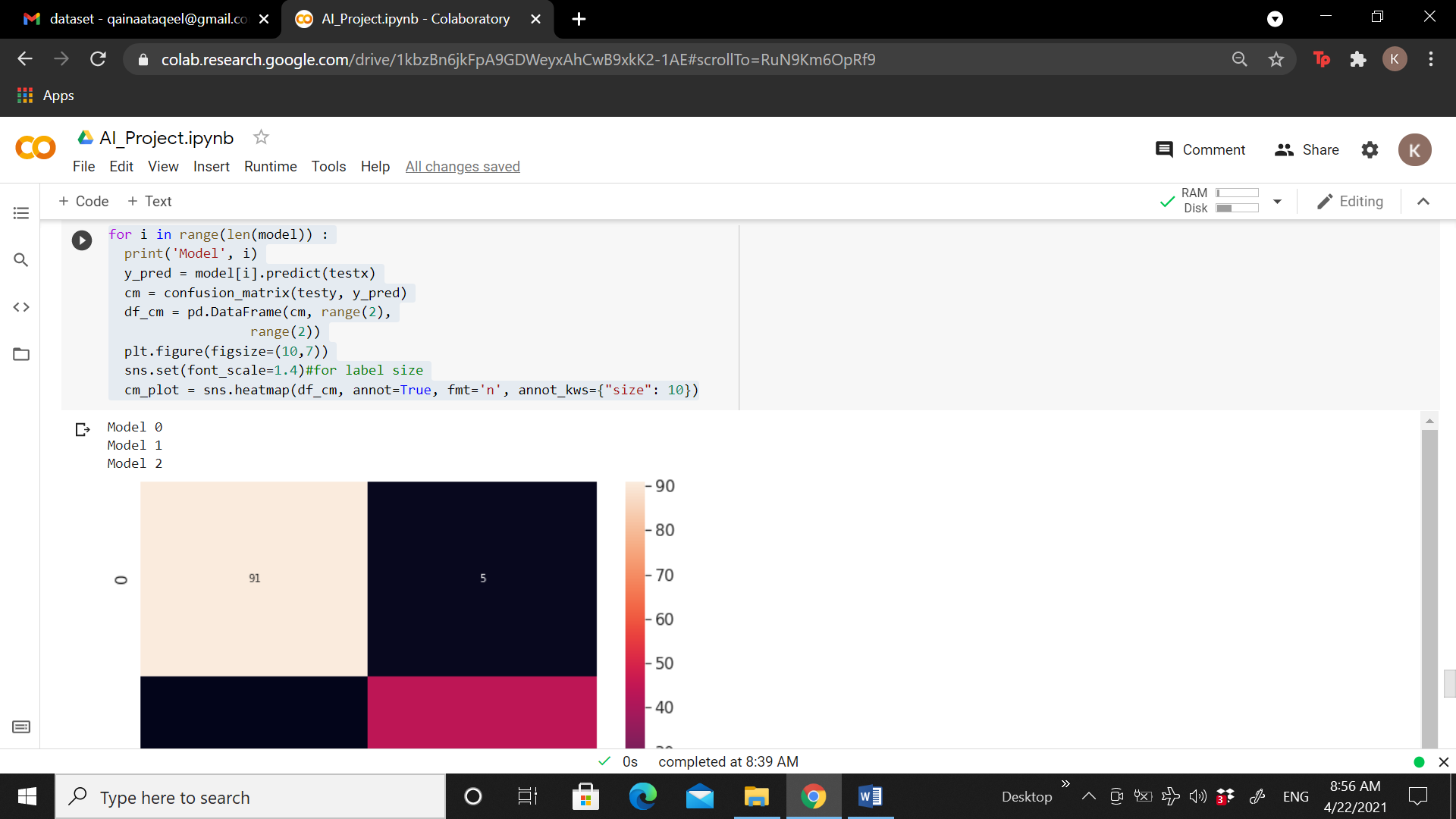
**Output Screenshots**

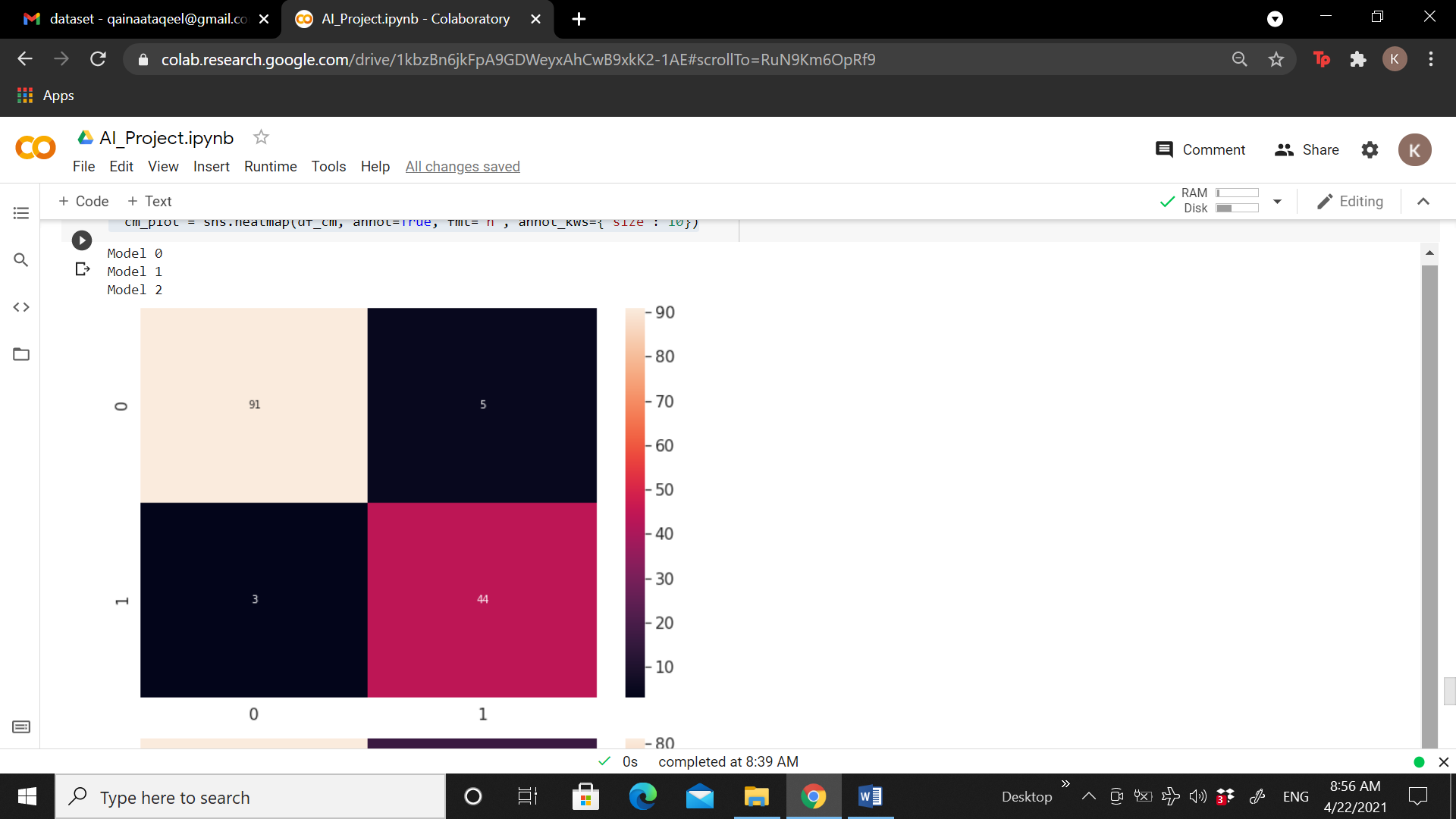


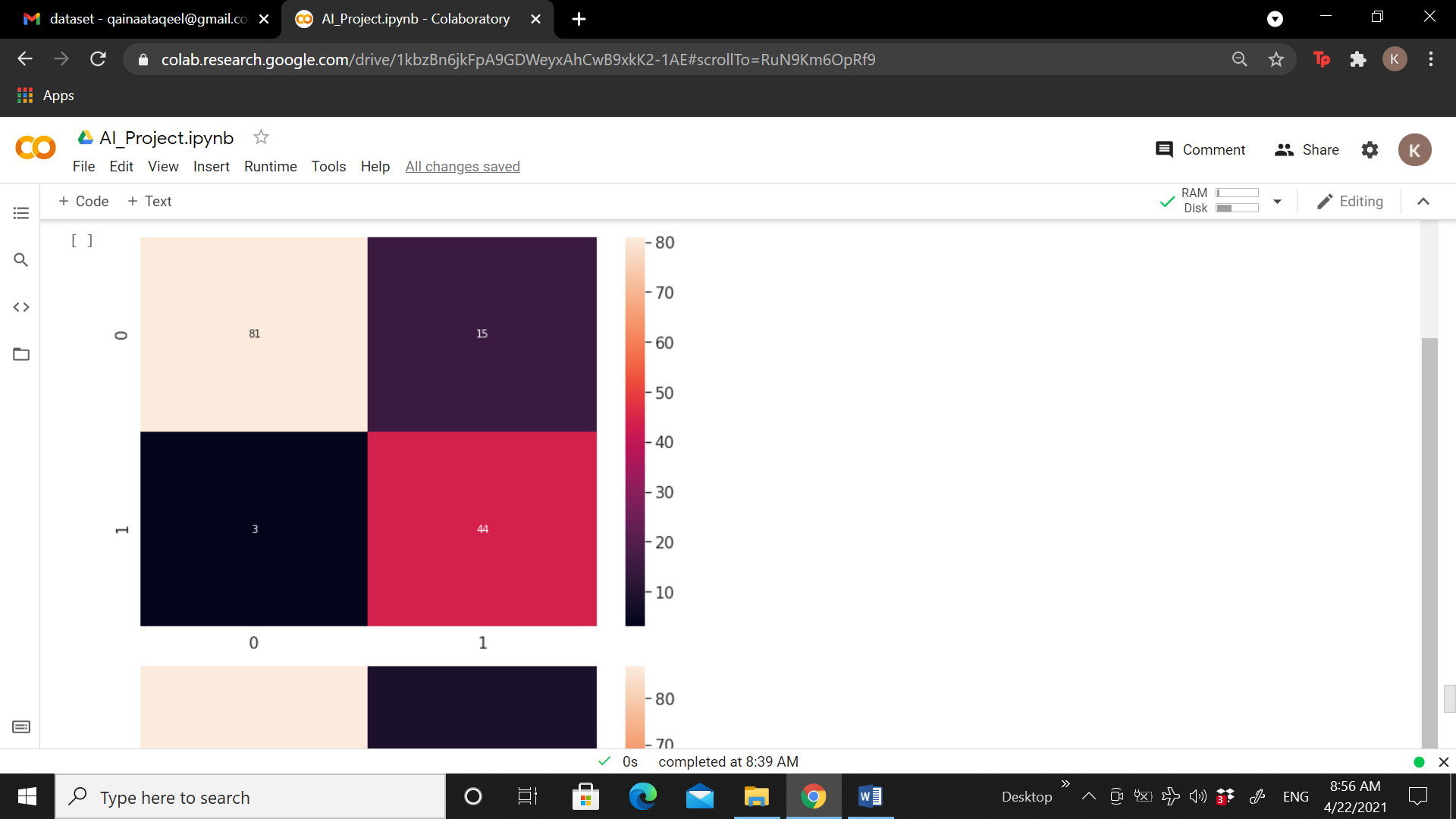
**Model Trainings**

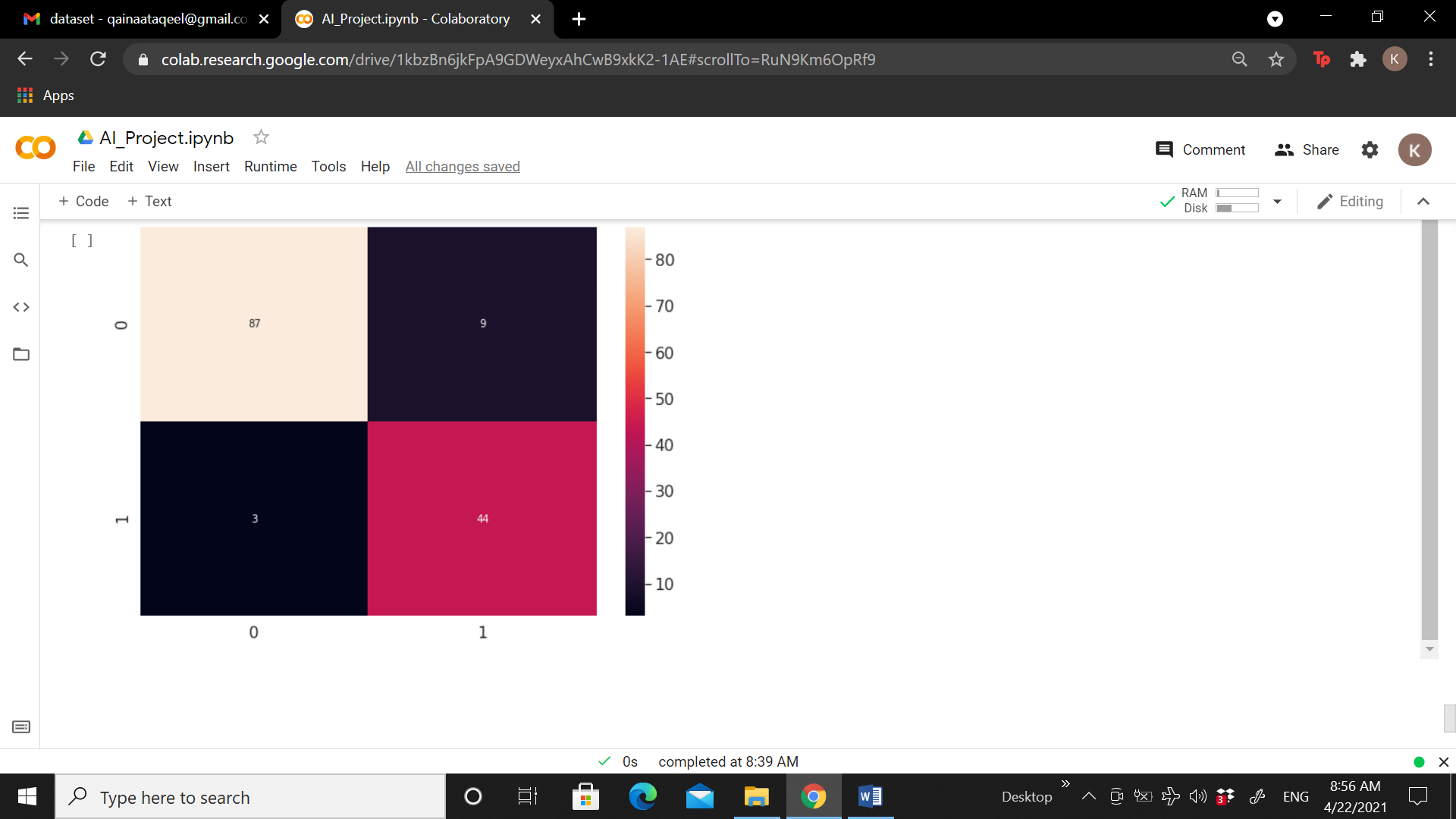
Confusion Matrix











**Supporting Tools and Techniques**

**Basic Library**

numpy

pandas

**Graph Library**

matplotlib.pyplot

seaborn

**Models and Split data Library**

sklearn.model\_selection import train\_test\_split

sklearn.ensemble import RandomForestClassifier

sklearn.linear\_model import LogisticRegression

sklearn.tree import DecisionTreeClassifier

**Models Performances Library**

from sklearn.metrics import confusion\_matrix,

from sklearn.metrics import accuracy\_score

from sklearn.metrics import precision\_recall\_fscore\_support

**Python Version**

3.8.2

**Tool**

Colaboratory by google

**References**

Dr. Affan’s Machine Learning Lecture where he gave students this dataset to classify that.