**Prediction Cow’s activities into 9 categoriesUsing machine learning**

N Sathish Kumar Reddy

Documentation:

**Section I**

**Objective**

Classify Cow’s activities into 9 labels based on Data collected from IMU Sensors

**Related to work**

presented an advanced methodology of data mining to predict Label for 9 datasets for cow position with 10 features, and a dependent label parameter with nine values.

**Dataset:** 9 datasets having the activities as follows

**Classes and their Encoded values**: -

eating = 1

drinking = 2

walking = 3

standing =4

lying = 5

ruminating standing = 6

ruminating lying = 7

grooming = 8

idle/other = 9

y=Dependent Variable

label

**x= independent variables**

acc\_x, acc\_y,acc\_z: accelerometer output for all 3 dimensions movement.

gyr\_x, gyr\_y, gyr\_z: gyroscope outputs, it measures rotation, rotation rate (angular velocity).

mag\_x, mag\_y,mag\_z: magnetometer outputs, catches magnetic field around the device.

Three machine learning algorithms were used: Decision Tree, Random Forest, K Nearest Neighbors,

**Section II**

Step 1

Importing the required common libraries

Reading the data

**Integrate the data obtained**

**Viewing the data, rows & columns, type of Data, Dimensionality**

**Step 2**

**Data exploration:** to understand the characteristics, format, and quality of data.

plots the most correlated columns to column "label

describe () analyses numeric as well as to object series or series of a Data Frame

**Exploratory Data Analysis** (**EDA**)  Using pandas profiling

Using Auto viz

Using Dataprep

understanding our target variable using count plot

checking for outliers using boxplot

Step 3

**Data Cleaning:**

Plotting box plot for identifying outliers

Replacing of outliers with median using IQR NumPy

Step 4

**Data Transformation**: Normalization - In this method, numerical data is converted into the specified range, i.e., between 0 and one so that scaling of data can be performed. Using minmax scaler

Step 5

Feature Selection using KBest classifier

Mutual Info

Correlation

Dropping the insignificant variables

Step 6

**Data Analysis**

Splitting the data in to train & test

Decision Tree algorithm with SMOTE Oversampling & Under sampling

Random forest Classifier

K nearest Neighbours Classifier

XGBoost Classifier

Train Model with each classifier

Test Prediction with each classifier

Classification Report for each test prediction

**Project Dataset, Code, and Sample Outputs**

**Reading the data from the drive**

eating= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/E1\_train.csv')

drinking= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/D2\_train.csv')

walking= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/W3\_train.csv')

standing= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/S4\_train.csv')

lying= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/L5\_train.csv')

ruminating\_standing= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/RS6\_train.csv')

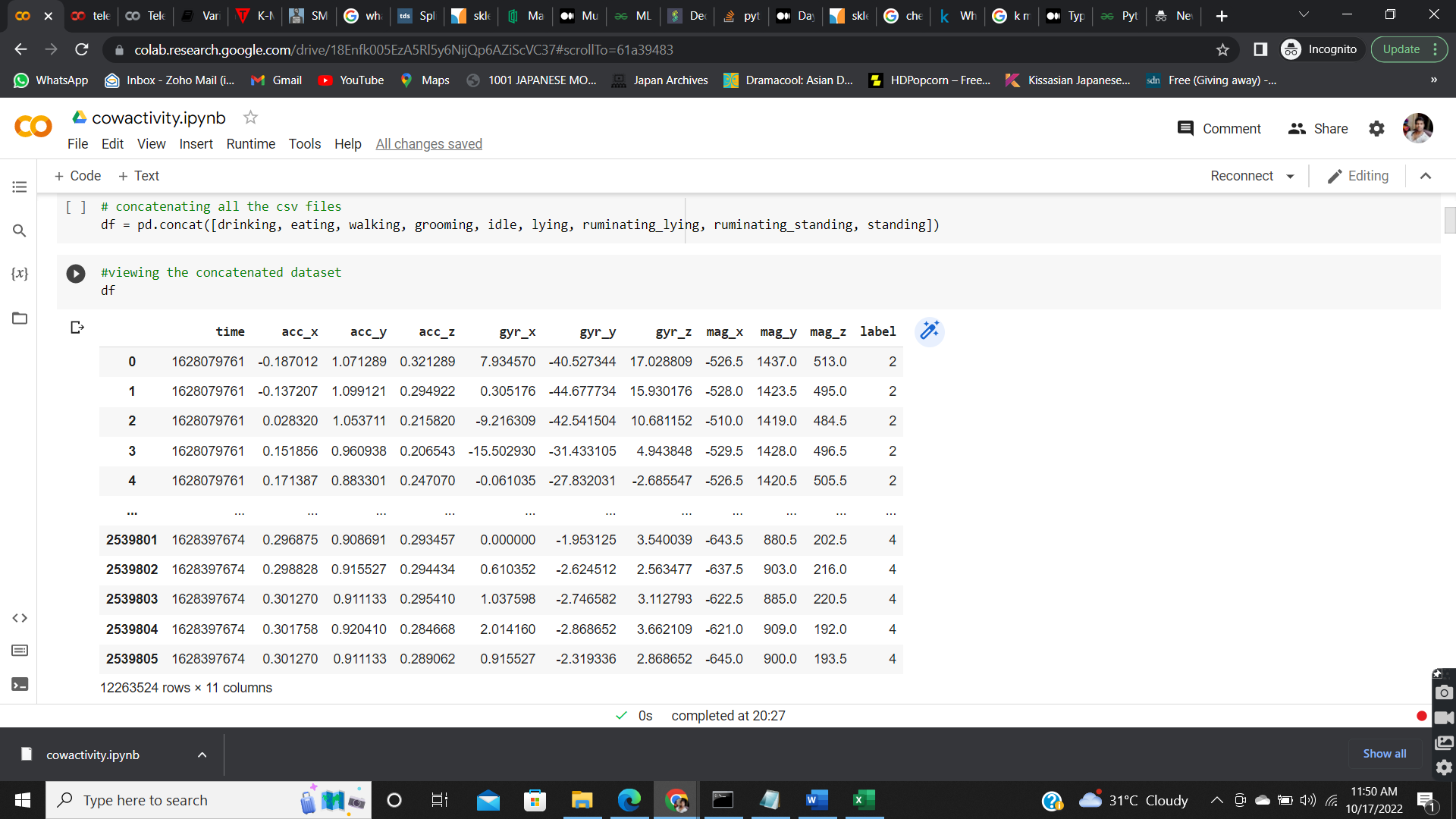
ruminating\_lying= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/RL7\_train.csv')

grooming= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/G8\_train.csv')

idle= pd.read\_csv('/content/drive/MyDrive/Cow\_prediction/I9\_train.csv')

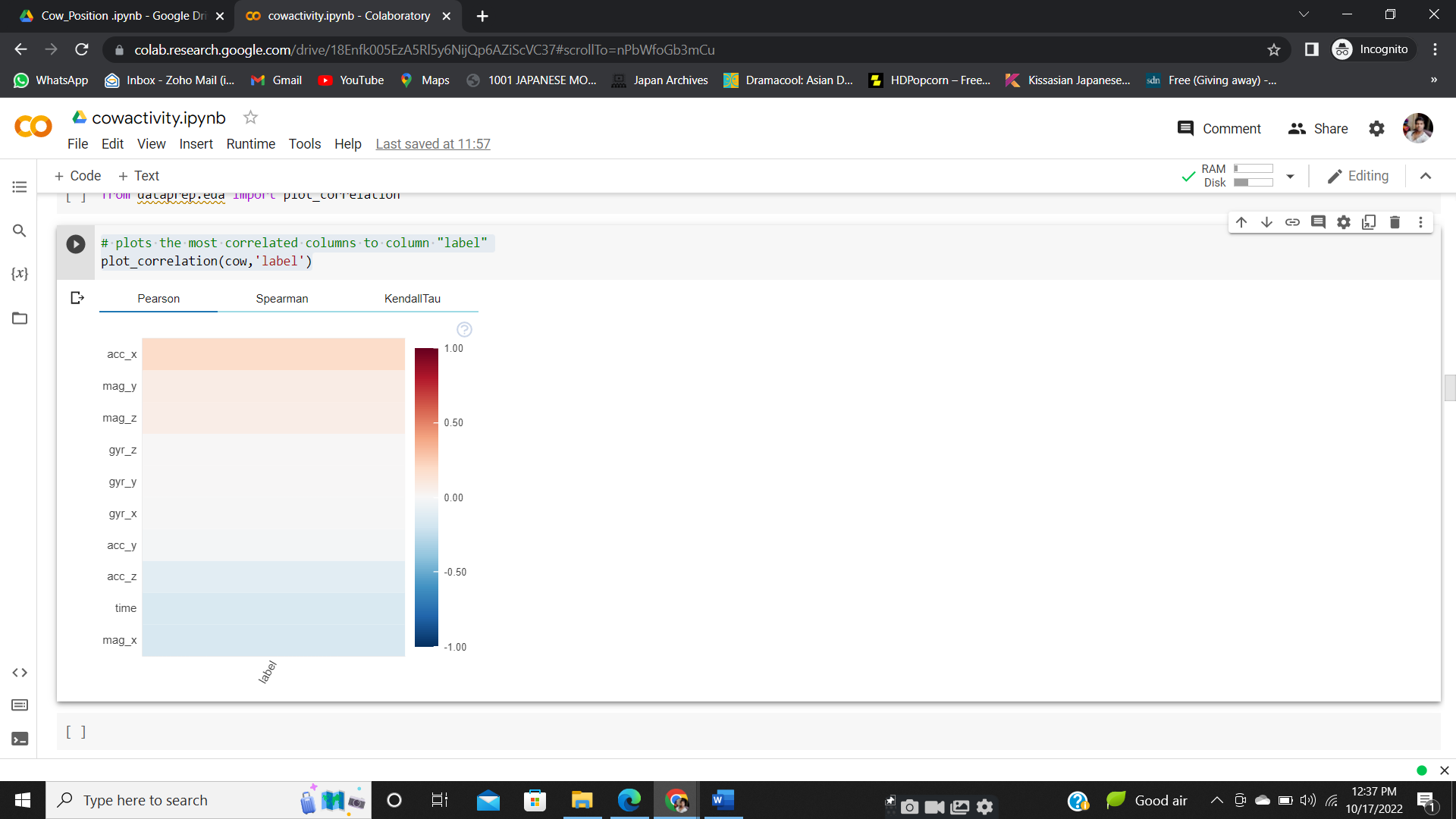
# concatenating all the csv files

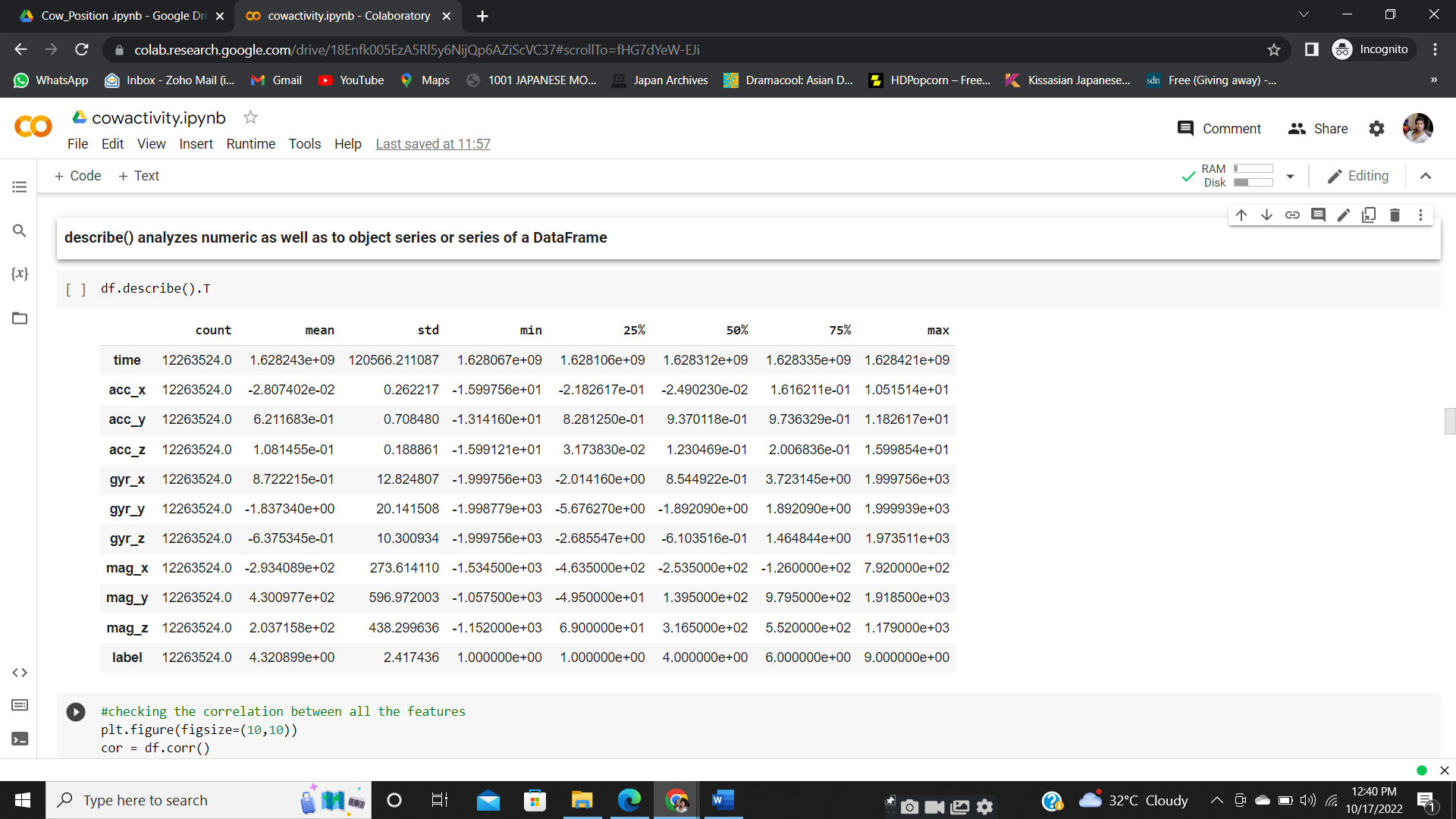
df = pd.concat([drinking, eating, walking, grooming, idle, lying, ruminating\_lying, ruminating\_standing, standing])

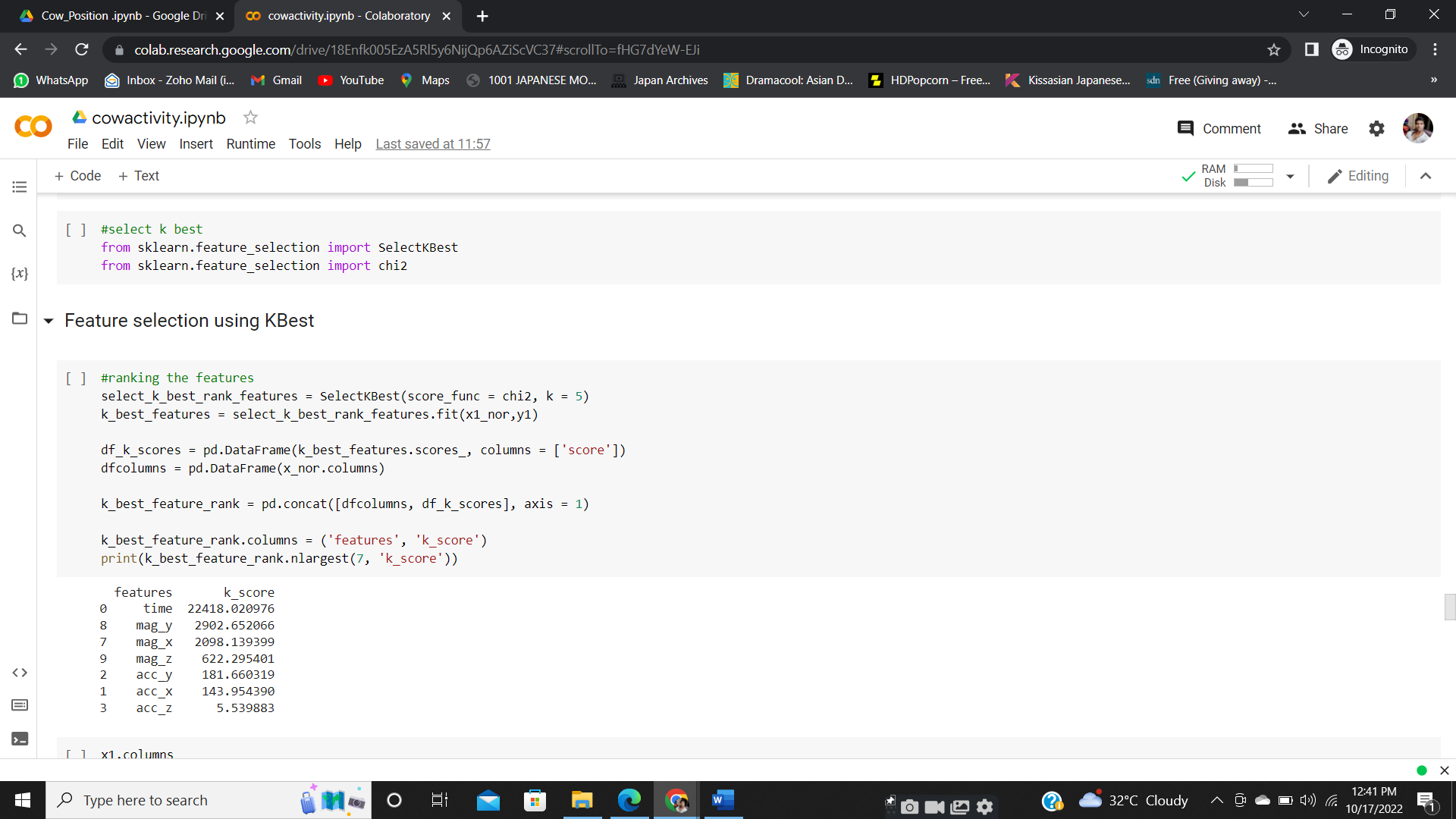


# plots the most correlated columns to column "label"

plot\_correlation(cow,'label')









**Decision Tree algorithm with SMOTE Oversampling & Under sampling**

print("Before OverSampling, counts of label '1': {}".format(sum(y1\_train == 1)))

print("Before OverSampling, counts of label '0': {} \n".format(sum(y1\_train == 0)))

# import SMOTE module from imblearn library

# pip install imblearn (if you don't have imblearn in your system)

from imblearn.over\_sampling import SMOTE

sm = SMOTE(random\_state = 2)

X\_train\_res, y\_train\_res = sm.fit\_resample(x1\_train, y1\_train.ravel())

print('After OverSampling, the shape of train\_X: {}'.format(X\_train\_res.shape))

print('After OverSampling, the shape of train\_y: {} \n'.format(y\_train\_res.shape))

print("After OverSampling, counts of label '1': {}".format(sum(y\_train\_res == 1)))

print("After OverSampling, counts of label '0': {}".format(sum(y\_train\_res == 0)))

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score

from sklearn.metrics import classification\_report

# Create Decision Tree classifier object

clf = DecisionTreeClassifier()

# Train Decision Tree Classifier

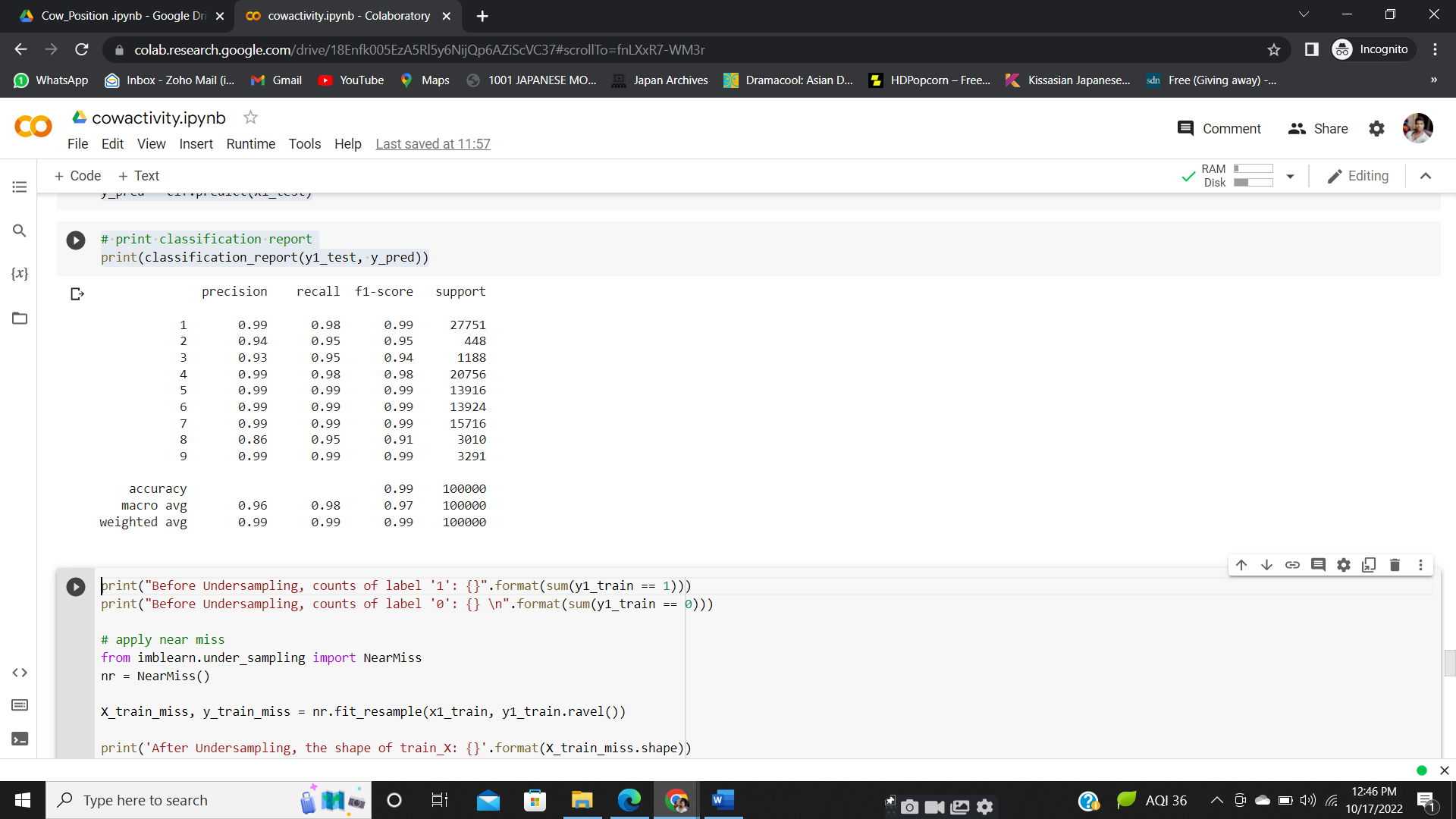
clf = clf.fit(X\_train\_res, y\_train\_res)

#Predict the response for test dataset

y\_pred = clf.predict(x1\_test)

# print classification report

print(classification\_report(y1\_test, y\_pred))



# Summary

Cow Activities dataset we started by cleaning the data and analyzing it with visualization. Then, to be able to build a machine learning model, we tried 6 different machine learning algorithms using default parameters. Finally, we tuned the hyperparameters of the SMOTE with decision Tree  (best performance model) for model optimization, obtaining an**accuracy of nearly 99%**