In this blog, we will analyse the Insurance Claims Fraud Dataset, Insurance fraud is a huge problem in the industry. It's difficult to identify fraud claims. Machine Learning is in a unique position to help the Auto Insurance industry with this problem.

**Defining the problem statement**

In this project, we are provided a dataset which has the details of the insurance policy along with the customer details. It also has the details of the accident on the basis of which the claims have been made.

In this example, we will be working with some auto insurance data to demonstrate how we can create a predictive model that predicts if an insurance claim is fraudulent or not.

Link to the dataset is here: <https://github.com/dsrscientist/Data-Science-ML-Capstone-Projects/blob/master/Automobile_insurance_fraud.csv>

First, we will import the required libraries for reading and EDA(Exploratory data analysis)

import pandas as pd

import numpy as np

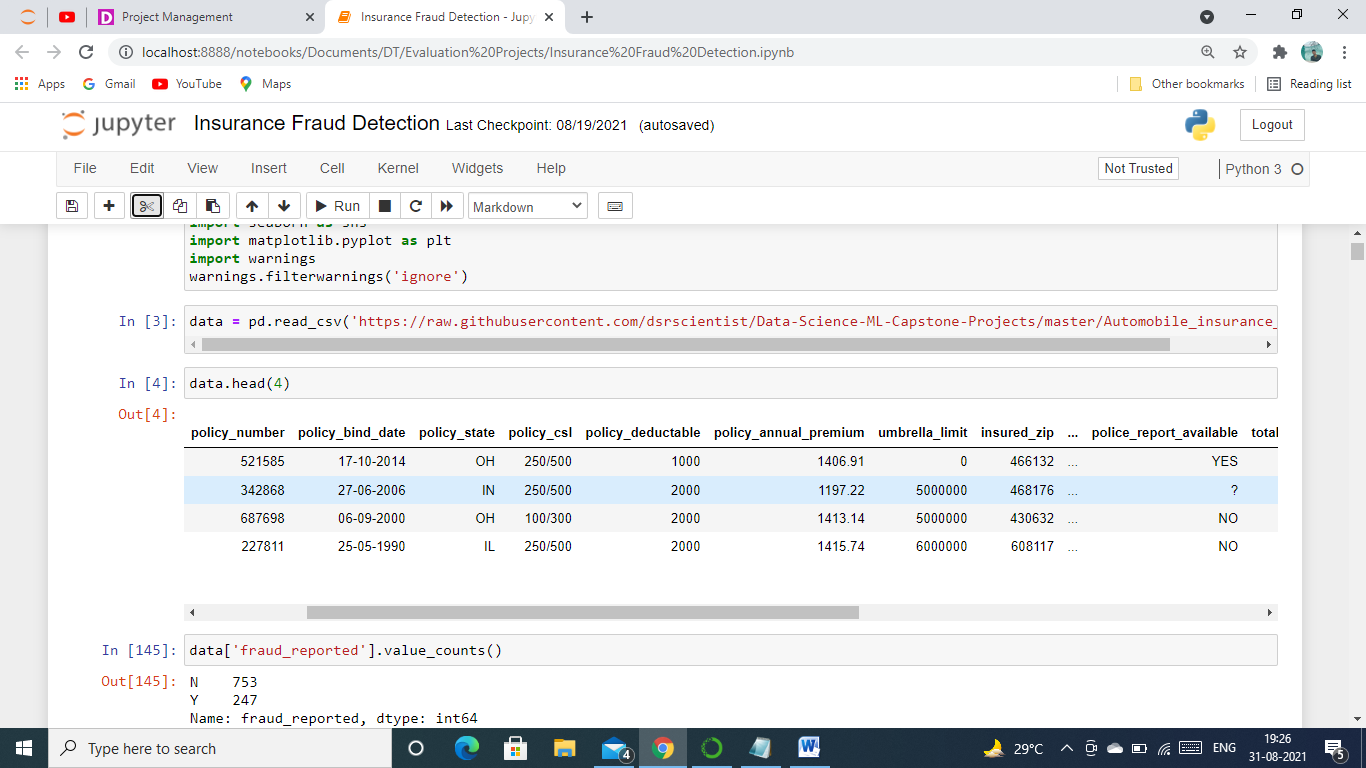
import seaborn as sns

import matplotlib.pyplot as plt

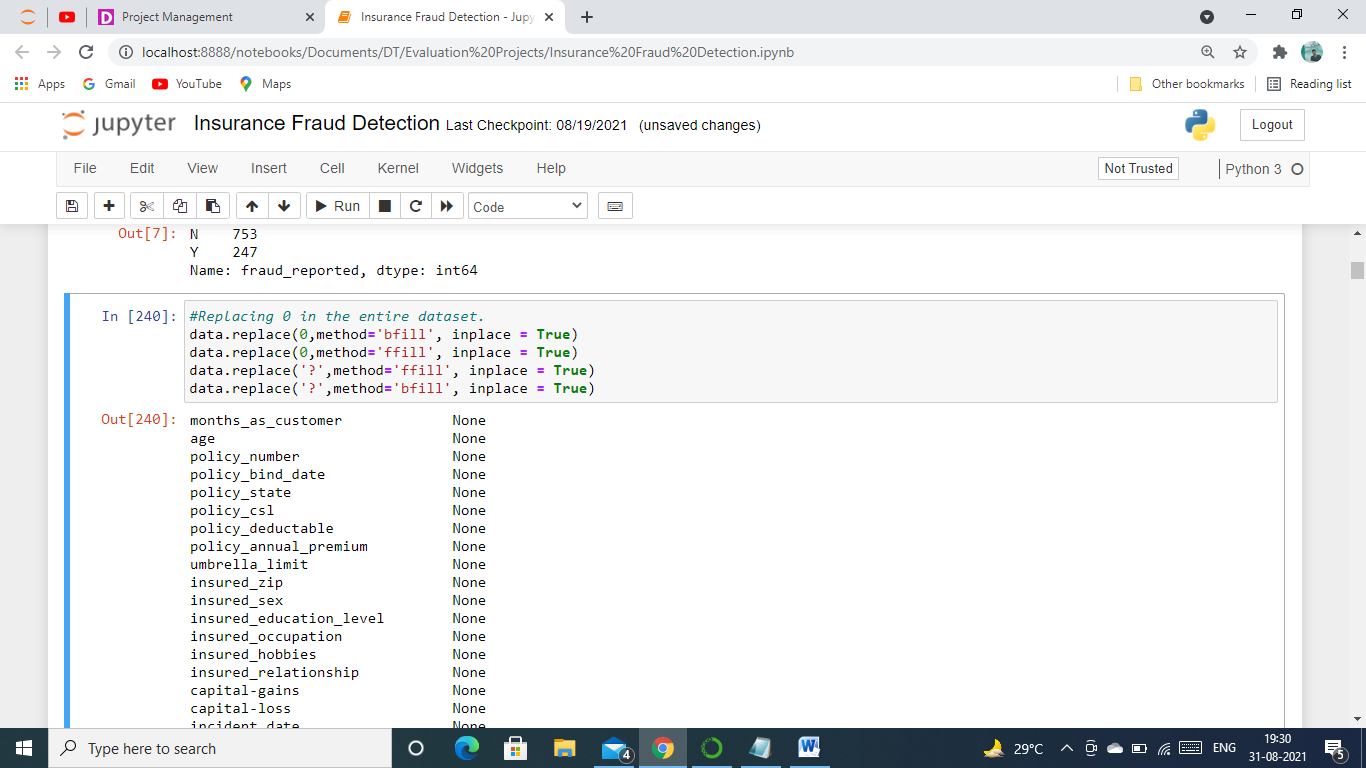
import warnings

warnings.filterwarnings('ignore')

Next, we load the data into a **pandas** dataframe using the **read\_csv** function



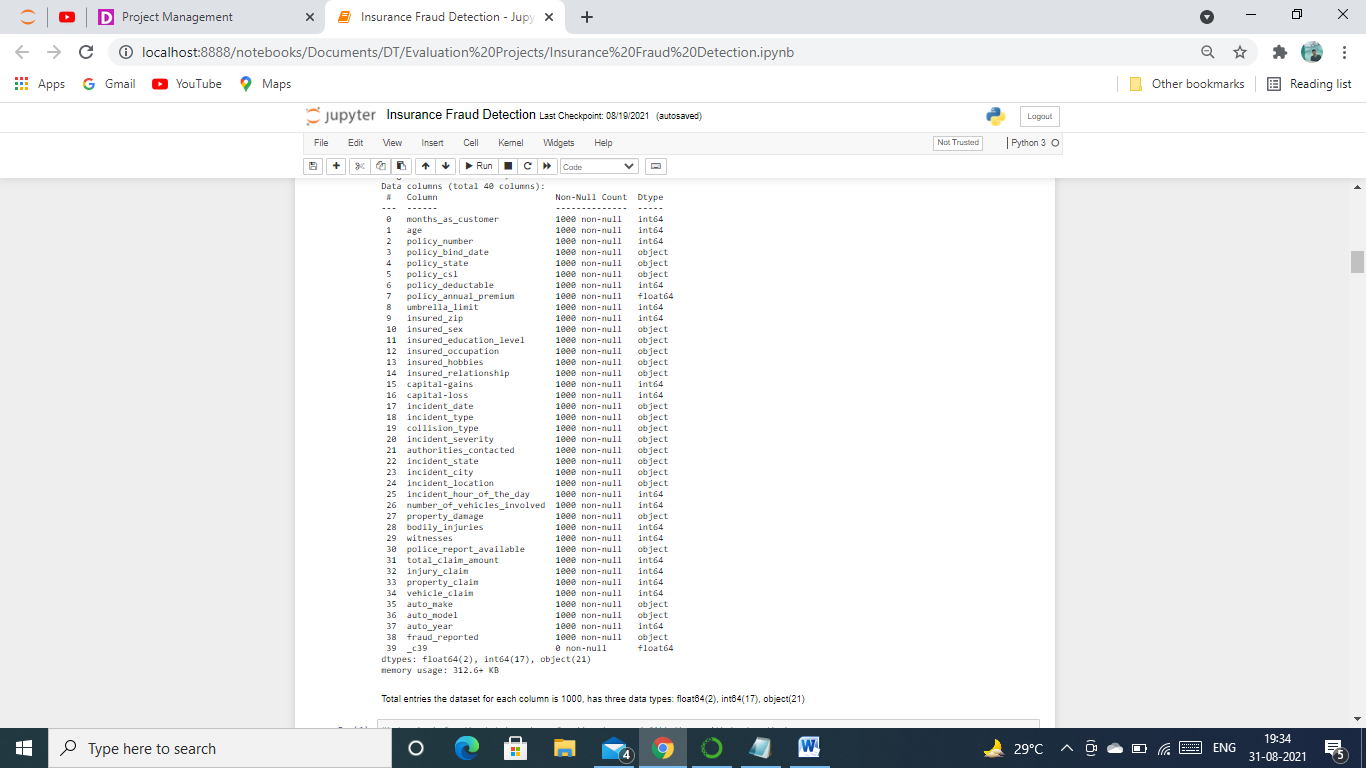
We can see above that data is successfully loaded into pandas dataframe, it is stored in variable data which we will use further to do EDA, Cleaning and Model Building. Also we found out that there are some of the columns which has 0 and ? as values in them so we are first going to replace these using data.replace().



Now we will proceed further and check do EDA(Exploratory Data Analysis) to analyse and understand the data.

**Exploratory Data Analysis**

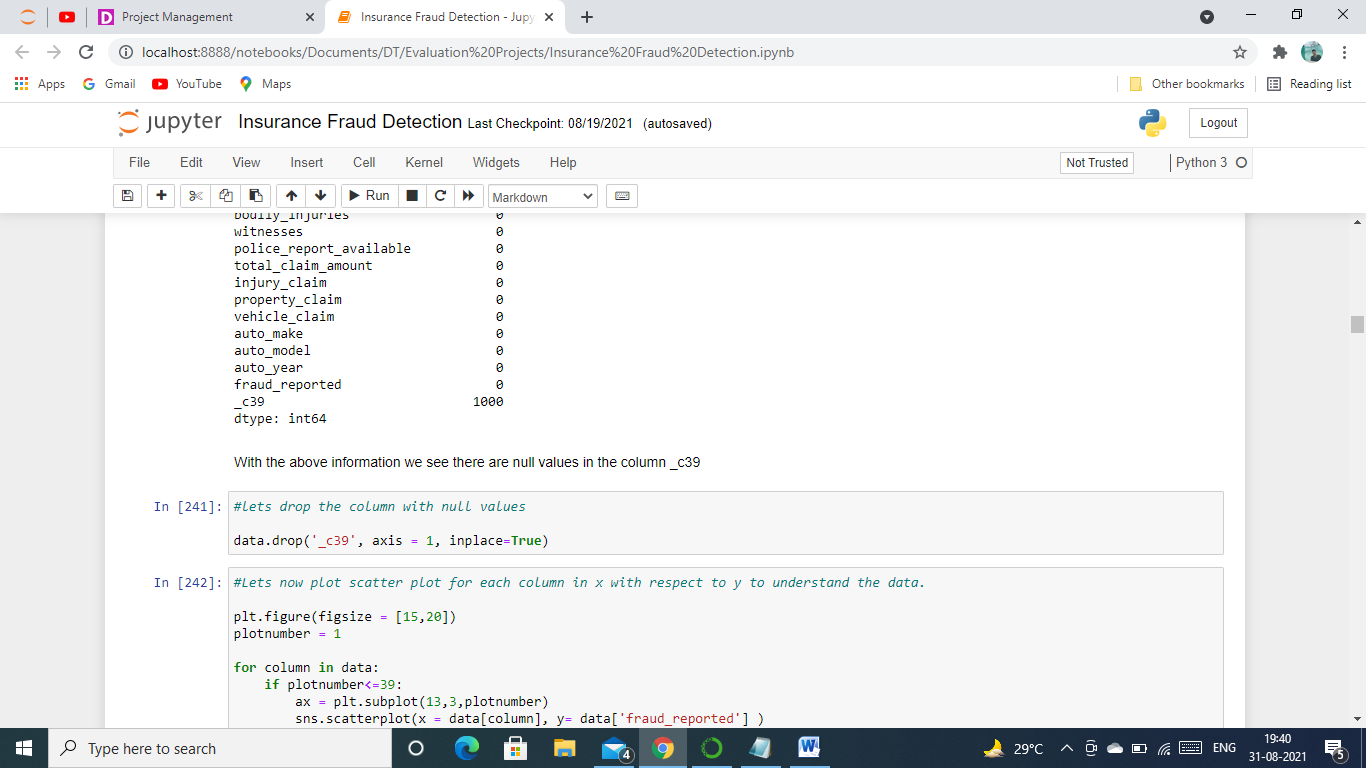
Let’s get more information about the data using  data.info()



Above figure gave us the detailed information of our dataset. Our Observations are as follows.

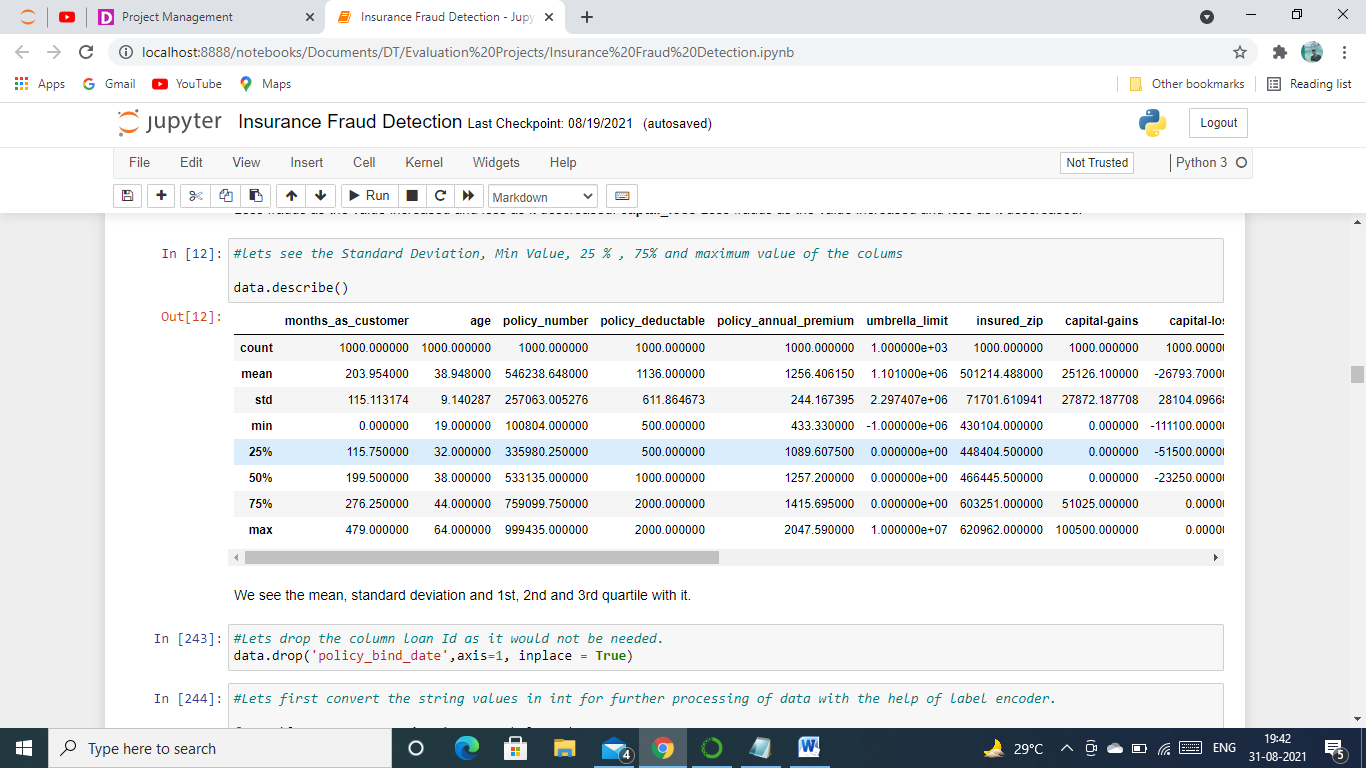
* Total entries in the dataset for each column is 1000, has three data types: float64(2), int64(17), object(21)
* There are 21 categorical columns in the dataset including our label and 19 numerical features.
* There are 40 columns out of which 39 are feature and one is label, fraud\_reported.
* \_c39 column has 0 entries in it.
* There are no null values in the dataset.

As we saw above that in the feature \_c39 there are no entries, we can remove it from the dataset for which we will use the below code.



The column \_c39 has been successfully now removed from our dataset.

More information about the data can be gathered by using data.describe()

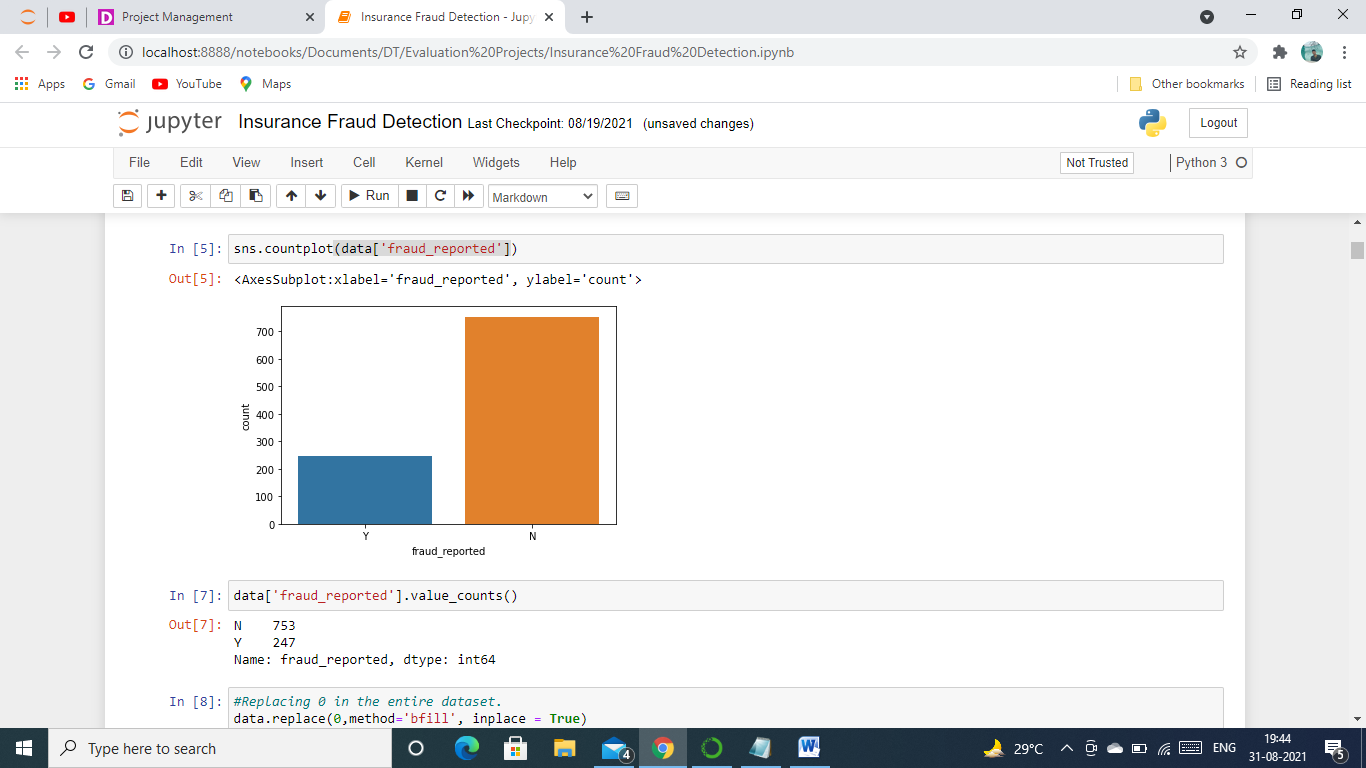


As we can see above it gives information regarding only the numerical data. Let see what we get from it.

* We can see the mean, standard deviation and 1st, 2nd and 3rd quartile of each feature.
* The values are on different scales. Many machine learning models require the values to be on the same scale. To scale the data we will use StandardScaler after cleaning it.

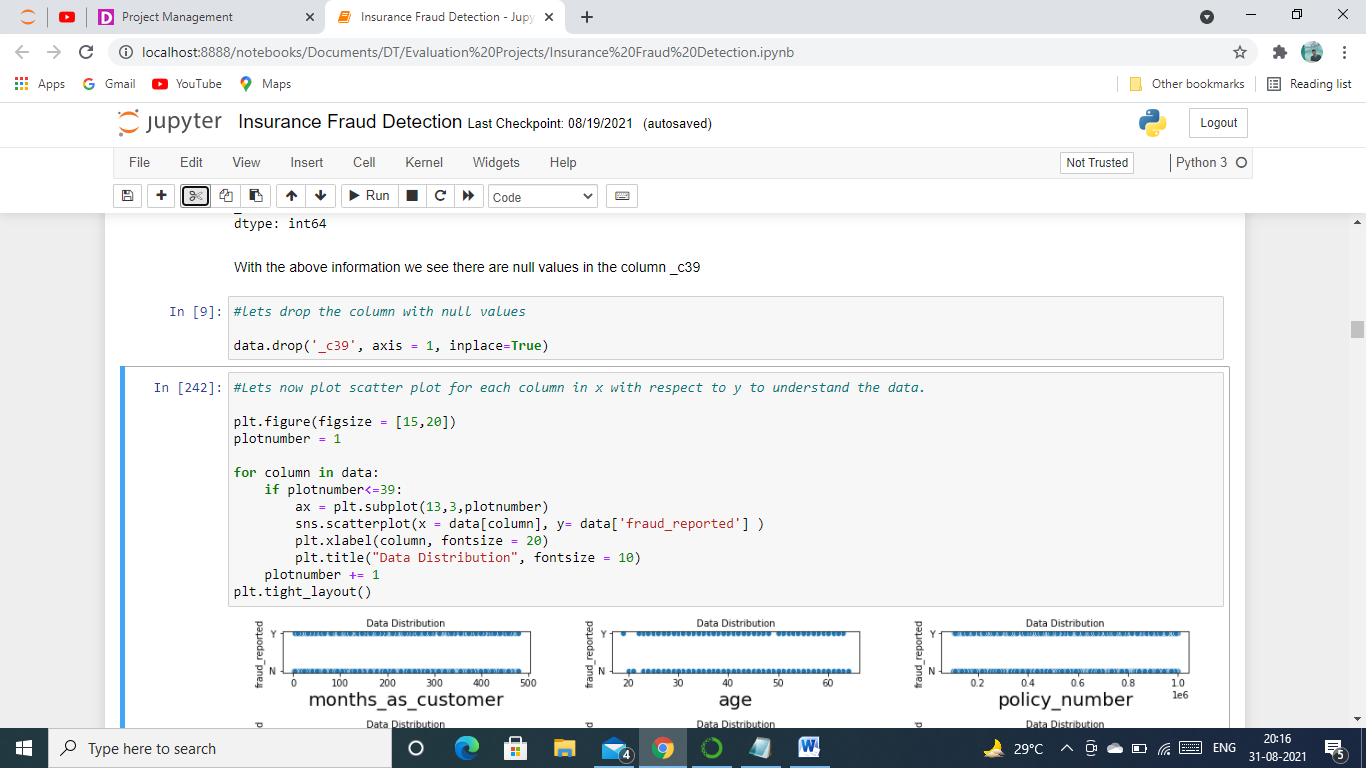
**Data Visualization**

**Let’s check our label(fraud detected first) we will use countplot which is available in seaborn library to analyse it.**



**We can see from the above figure that there are two values Y( stands for fraud detected) and N( stands for fraud not detected. Number of Y is more than N which means there is imbalance in the label and to avoid the imbalance for building our model we are going to use Stratify parameter in train test split.**

**We will now plot scatter plot from Matplot library for each column in x with respect to y (Income) to understand the data. The code for which is below.**



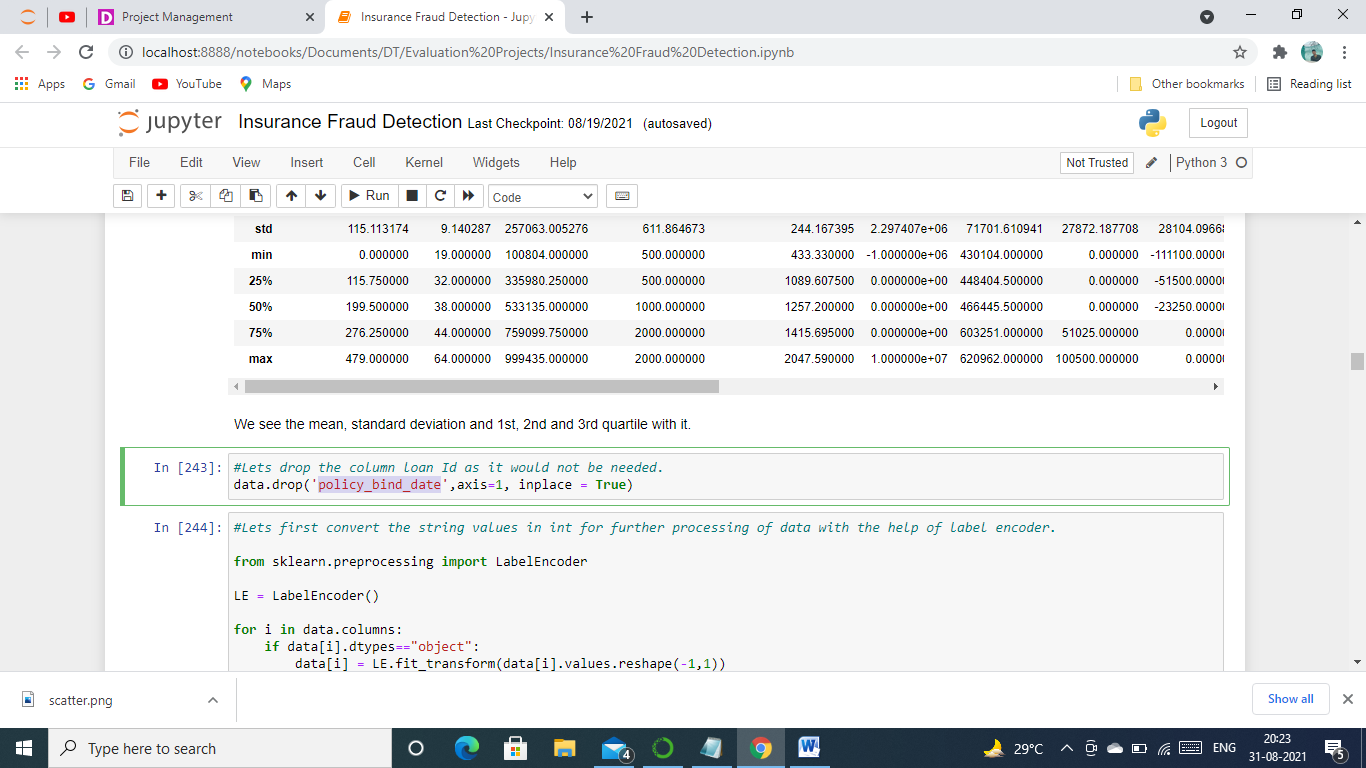
**Now let’s see the plot and gather some information from it.**



**Below are the observations from scatterplot.**

* **Vehicle Claim** Less frauds detected in it as it increased.
* **property\_claim** More frauds as the value increased and less as it descreased.
* **injury\_claim** More frauds as the value increased and less as it descreased.
* **total\_claim\_amount** Less frauds as the value increased and less as it descreased.
* **captal\_gains** Less frauds as the value increased and less as it descreased.
* **captal\_loss** Less frauds as the value increased and less as it descreased.

**We will now remove the feature policy\_bind\_date as it will hardly play any role in building model.**



**Since there are 21 categorical columns in the data we will first encode them for this we are going to use Label Encoder, we have to first import it and then use it for categorical columns.**

#Lets first convert the string values in int for further processing of data with the help of label encoder.

from sklearn.preprocessing import LabelEncoder

LE = LabelEncoder()

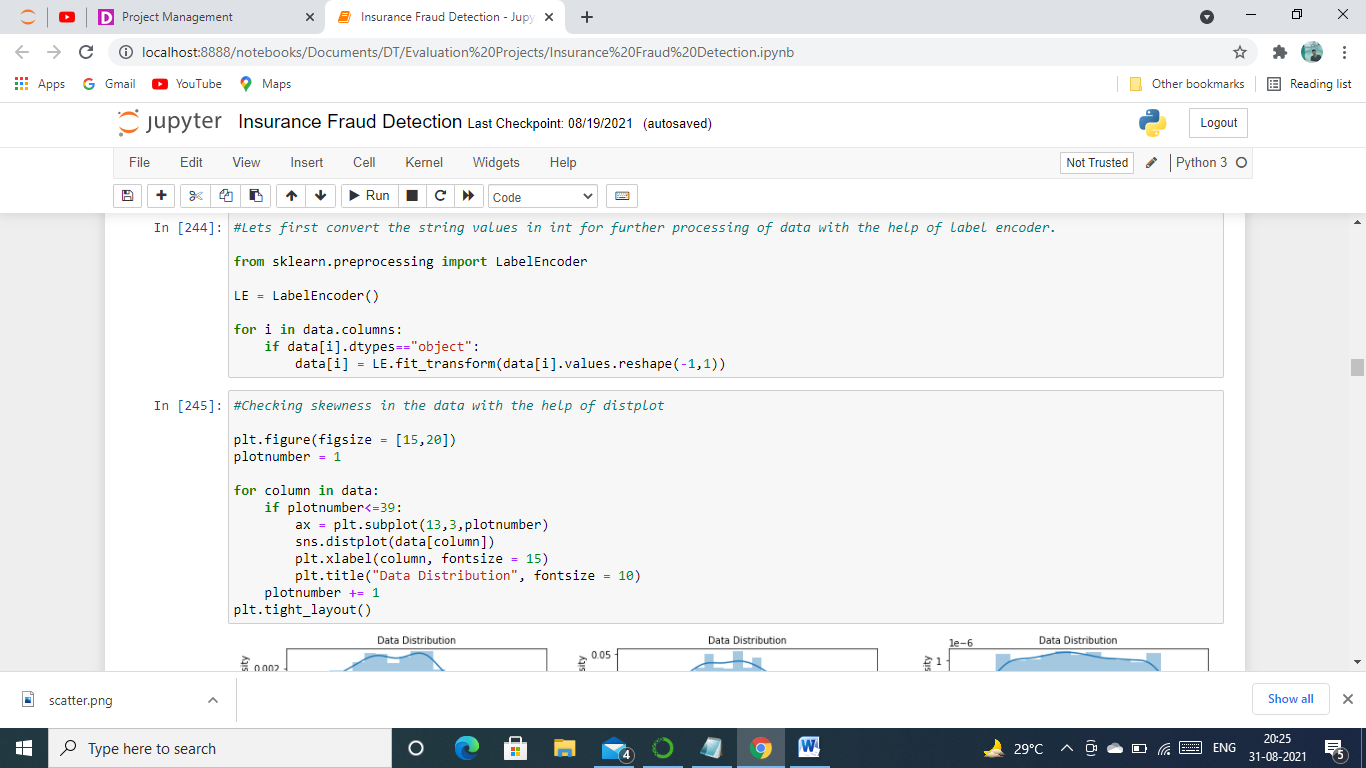
for i in data.columns:

if data[i].dtypes=="object":

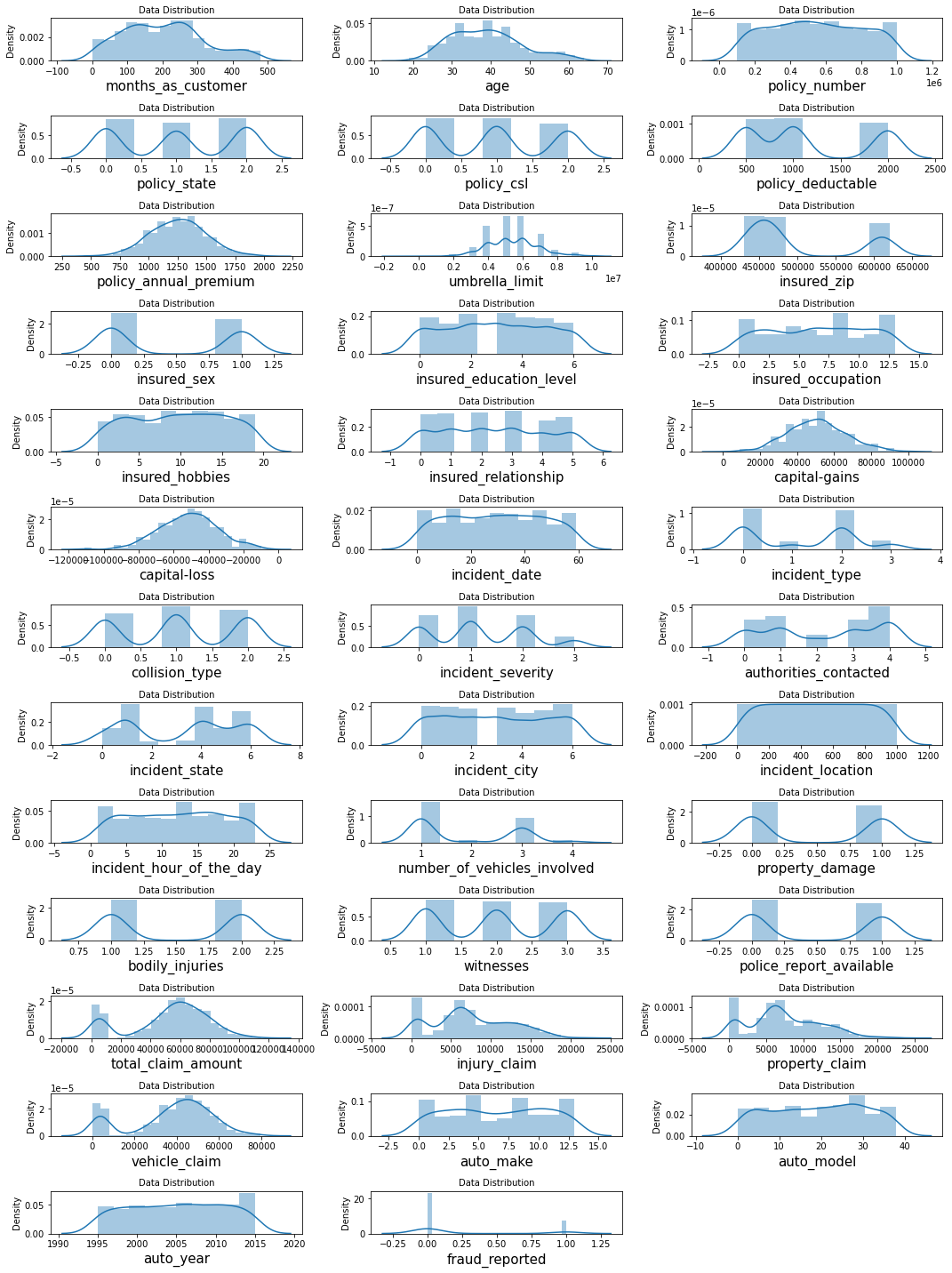
data[i] = LE.fit\_transform(data[i].values.reshape(-1,1))

This code encoded our categorical columns into numericals which will help us to do further analysis and model building.

Let’s us now check for Skewness in the dataset, what is it? Skew is the degree of distortion from a normal distribution, normal distribution forms bell curve and is consider good for research. We will use distplot in seaborn library to plot it for the dataset. Below is the code for it.



Here is the output of our above code.



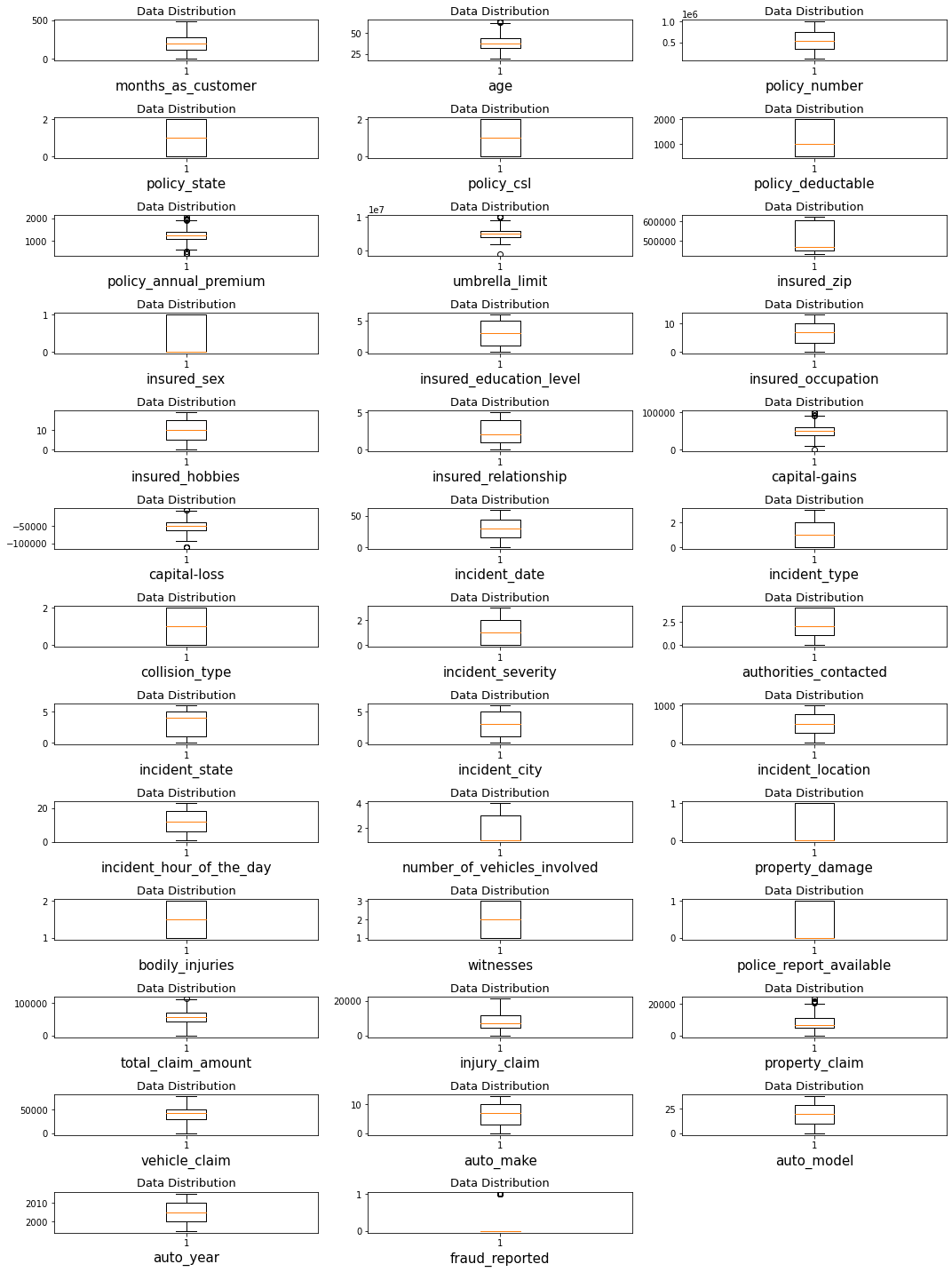
From the above figure we can find the below observations.

**This shows that there is skewness in the columns** 1- insured\_zip 2- total\_claim\_amount 3- vehicle\_claim and are continuous data which means not normal and is not good for our model, these are skewed towards right as we can see in the figure. Others also have but are categorical data which we do not consider for building a model.

The next step is to check for outliers, what are these? An outlier is an object that deviates significantly from the rest of the objects. They can be caused by measurement or execution error. Outliers are not good for model hence we will check it by plotting boxplot which is in Matplot library with the below code.



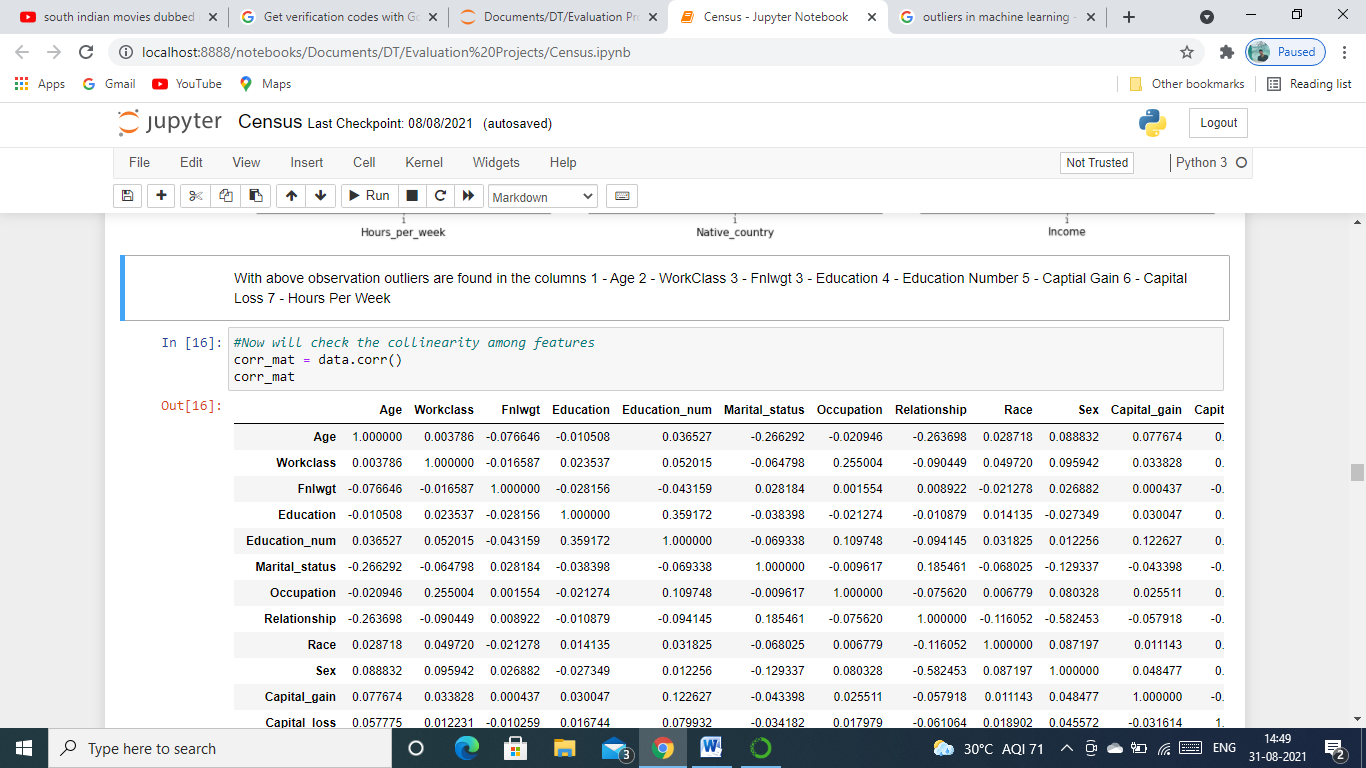
This code will give us the below output.

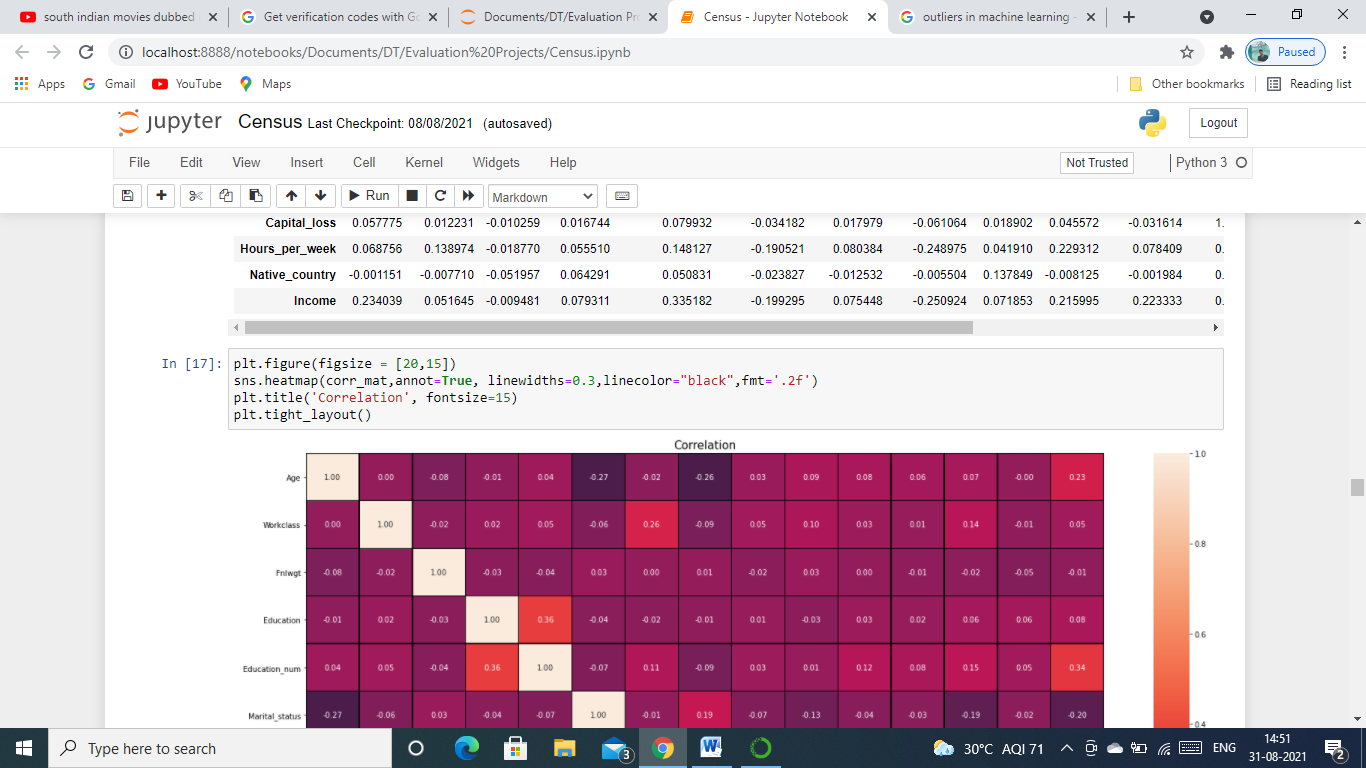


What are our observations?

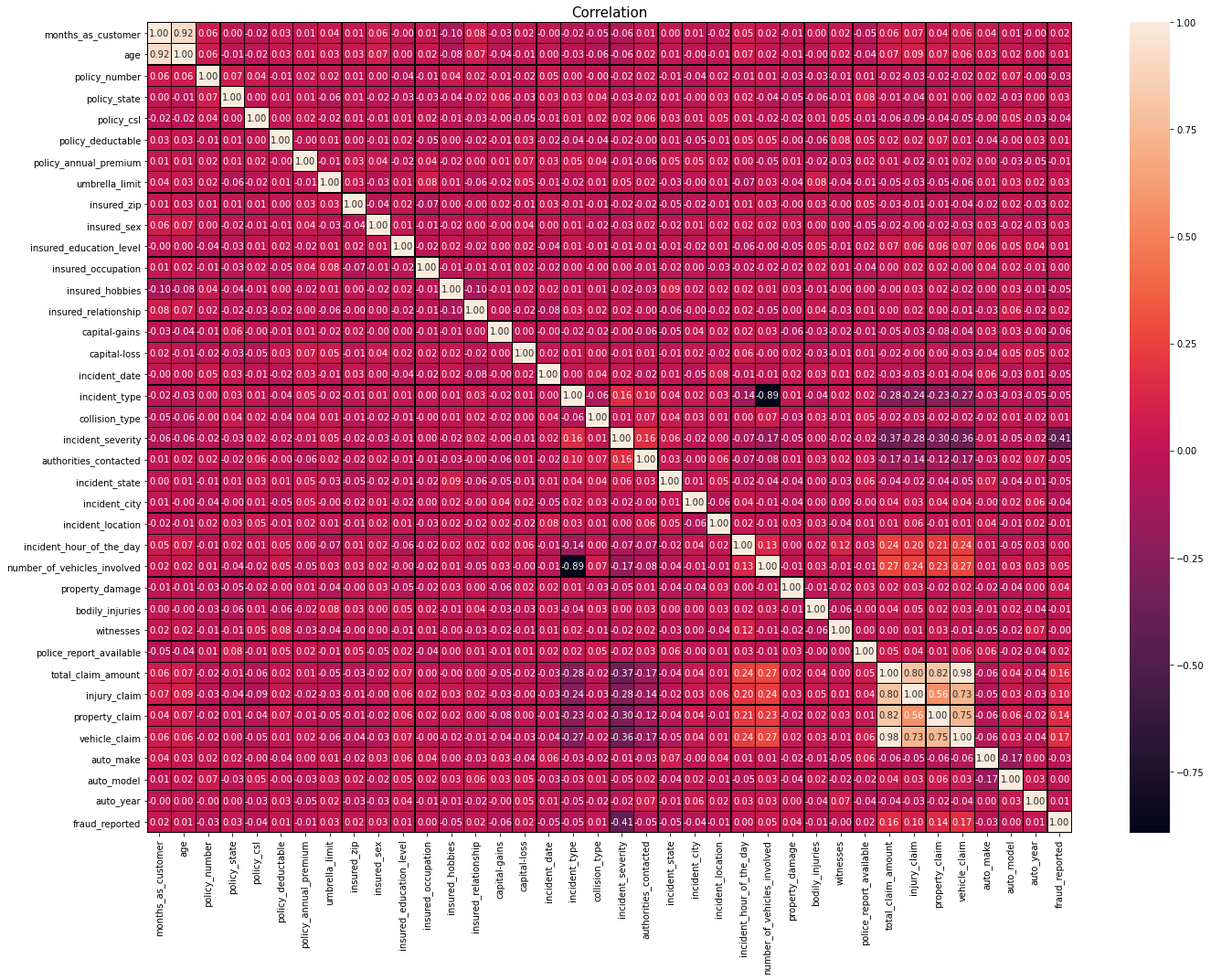
With above observation outliers are found in the columns with above observation we see there are lot of outliers in the column policy\_annual\_premium,age, property\_claim, capital\_gains. These outliers needs to be removed from entire dataset so that we can build a good model for this problem statement.

Now we will look for co-linearity among features, how can we do that? Well for this we will plot heatmap which is also available in seaborn library and is only used to plot the co-linearity to understand it better. Below is the code for it.





This code will give us the below heatmap for us to understand the co-linearity among features in our dataset.



What can be observe from the above figure?

* Max Correlation of the output with the features columns is vehicle\_claim.
* Min Correlation of the output with the features columns is with incident\_severity.
* Also we can see that there is too much corelation between age and months\_as\_customer, vehicle\_claim and total\_claim\_amount.

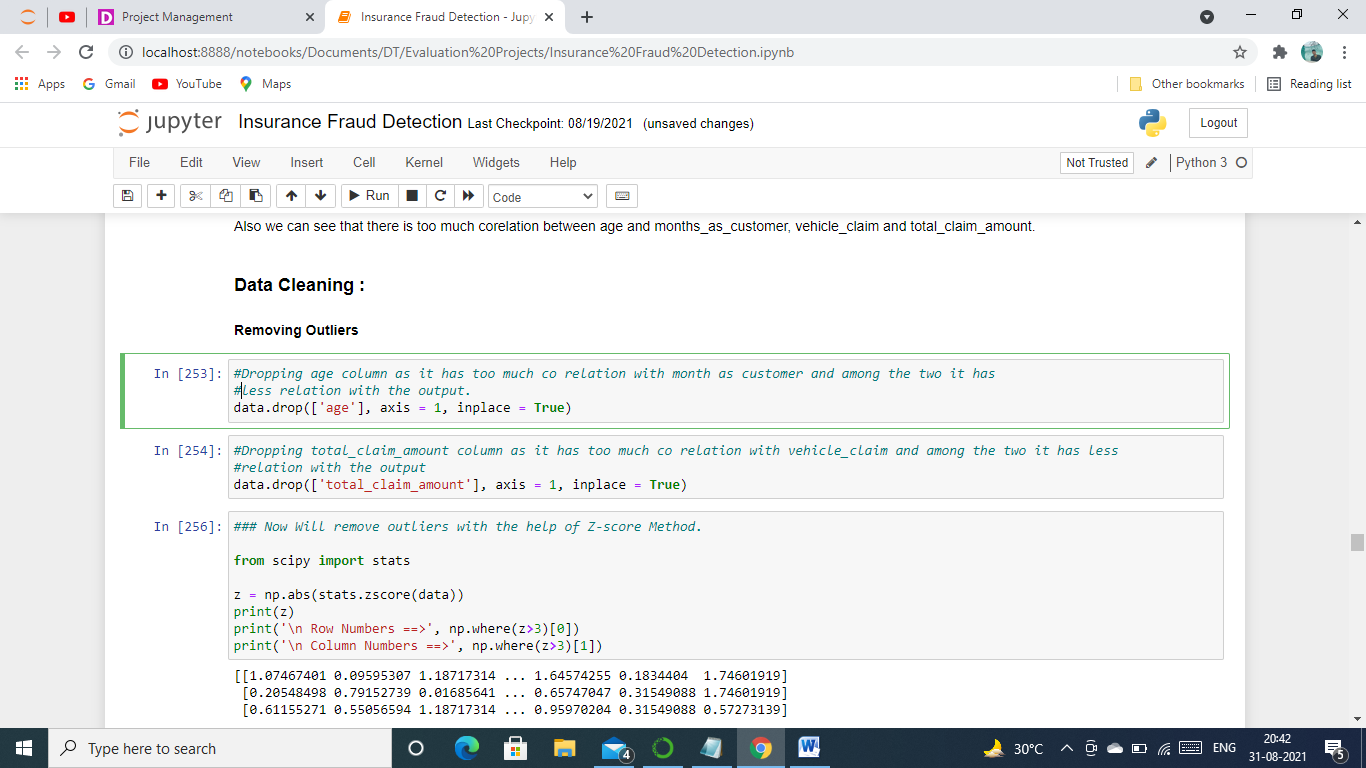
Since there are not much co-linearity among few features we will remove some of them from the dataset depending upon which has least co-relation with the label feature for building our model.

Now since we visualized all the important factors, we will clean it for building a good model.

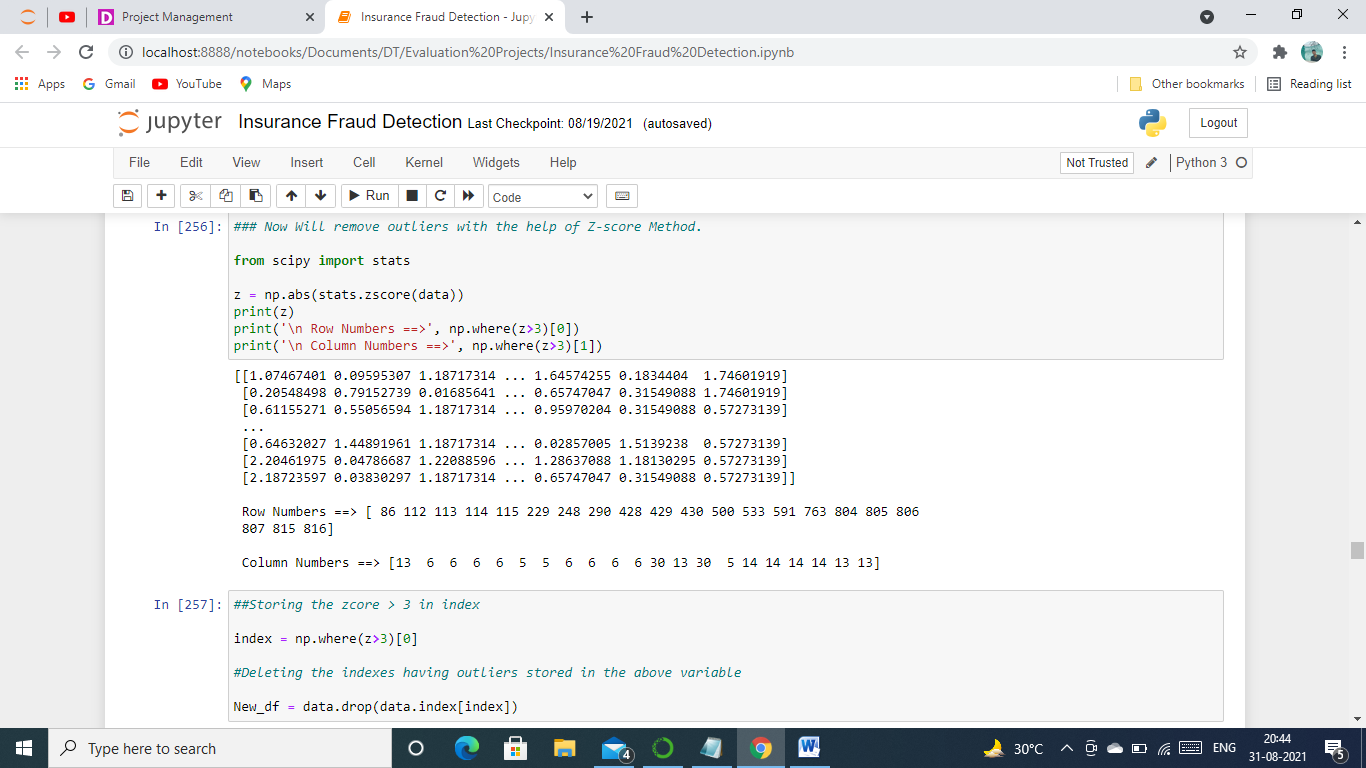
**Data Cleaning**

We will first delete the columns which has co-linearity using drop() function.

Below is the code for it.

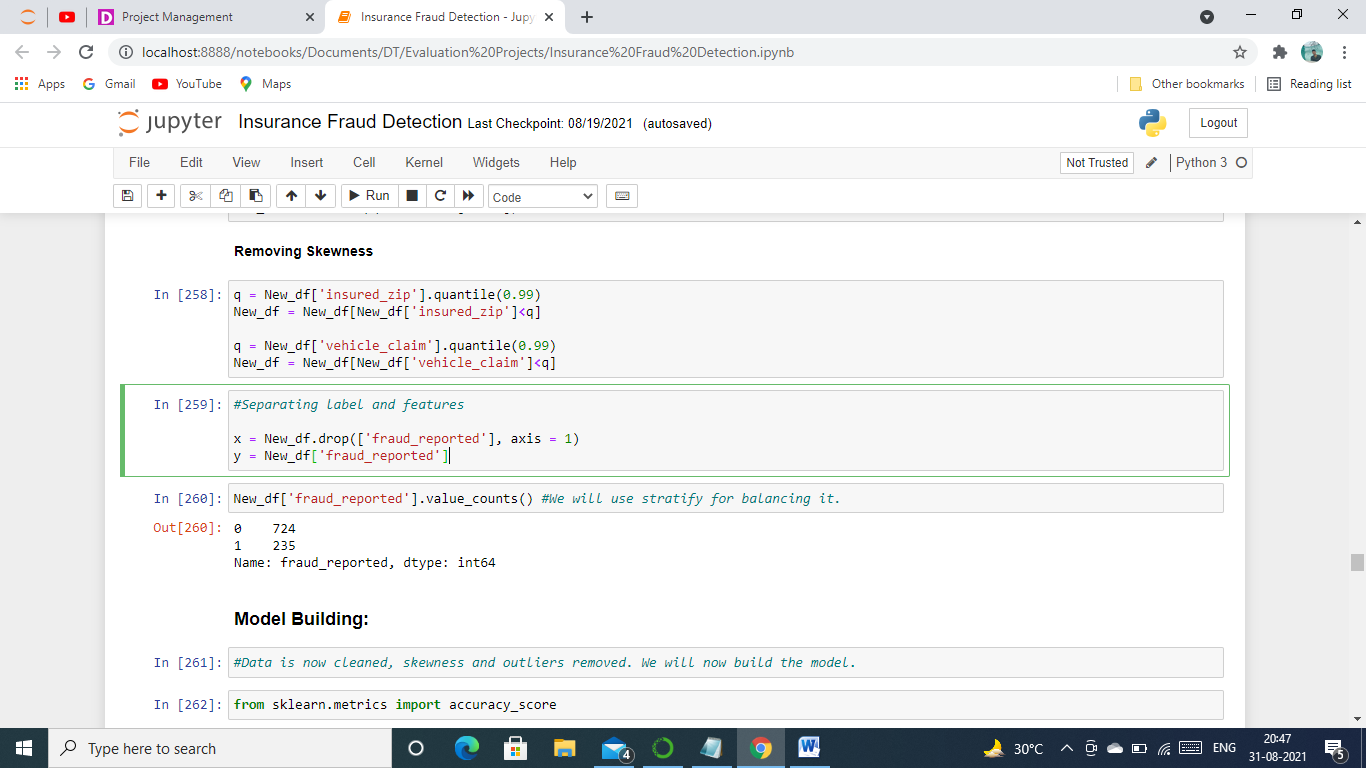


As we saw above that there were outliers in the dataset, we will remove these with the help of zscore method , we will import the necessary library required for it and then we will apply zscore on the dataset to remove using data.drop() and store it in a new variable New\_df. Below is the code and output for removing the outliers.



Since we have stored the data without outliers in a different variable New\_df, the next step for us would be to remove the skewness from it, as we observed before that we had skewness in two of the features, insured\_zip and vehicle\_claim. We will remove skewness with the help of quantile method.

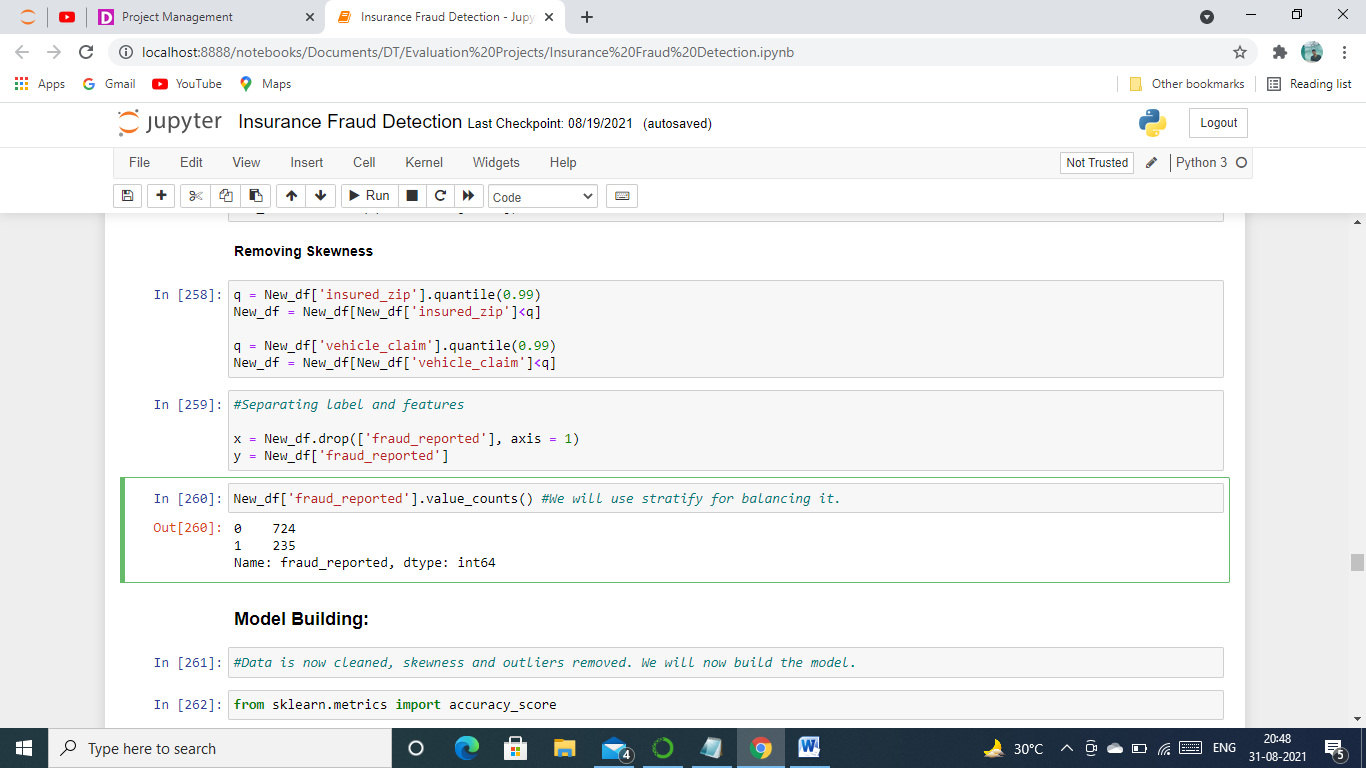
Code’s are as follows.



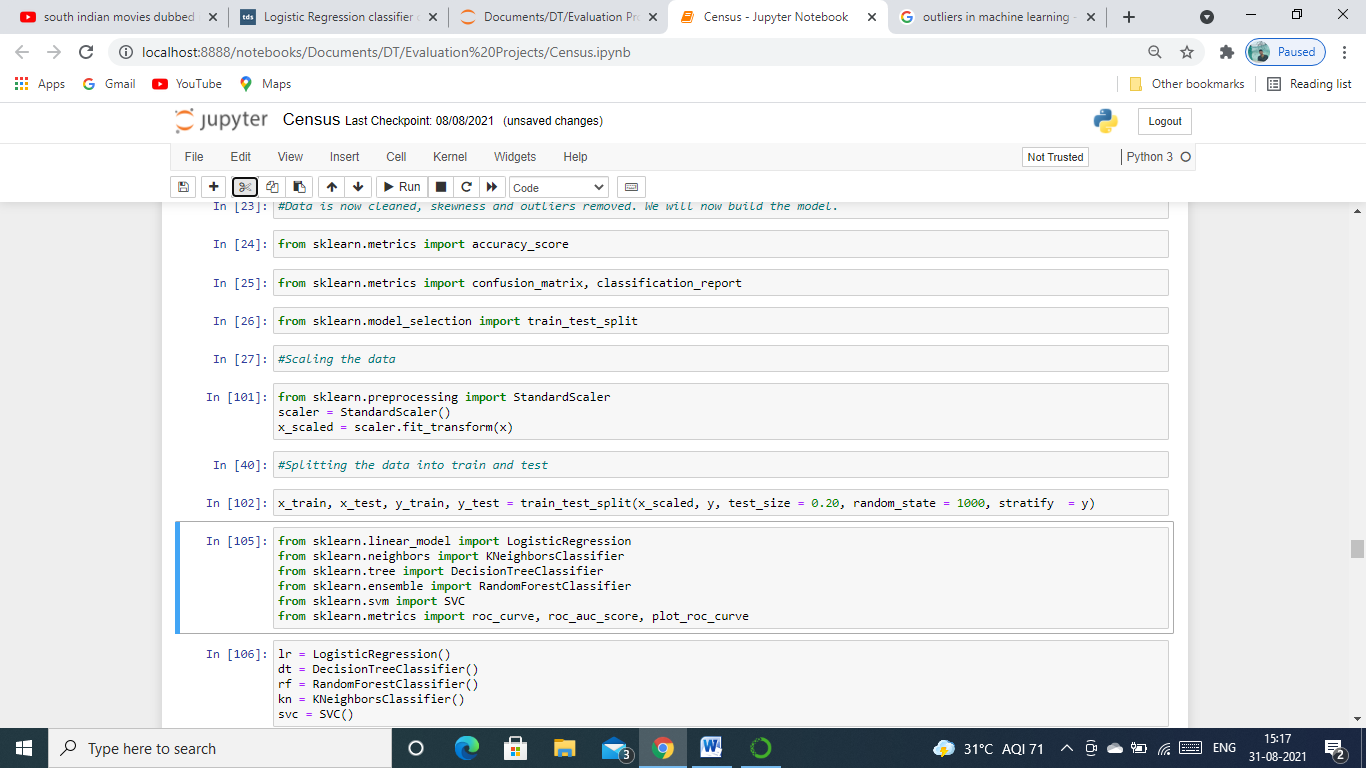
Now that our data is clean, removed all the outliers and skewness from it, Next step we finally can say that our data is good for building model.

**Model Building:**

Let us now first separate our features and label into X and Y variable to build our model.

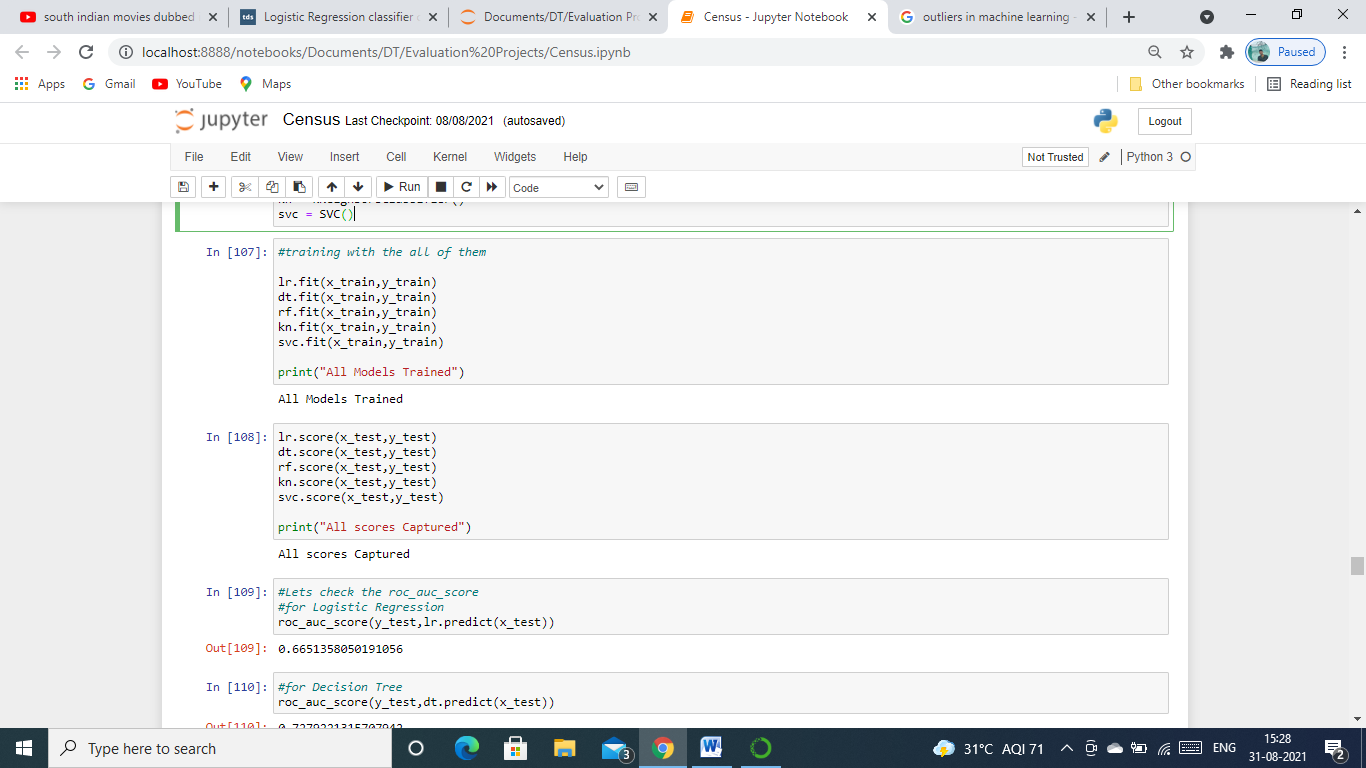


Now we are going to import the required algorithms and metrics which we will use to train and test the model and store them in a variable and then scale our features with the help of StandardScaler as we saw previously that they were on different scale. Below are the codes for all this.

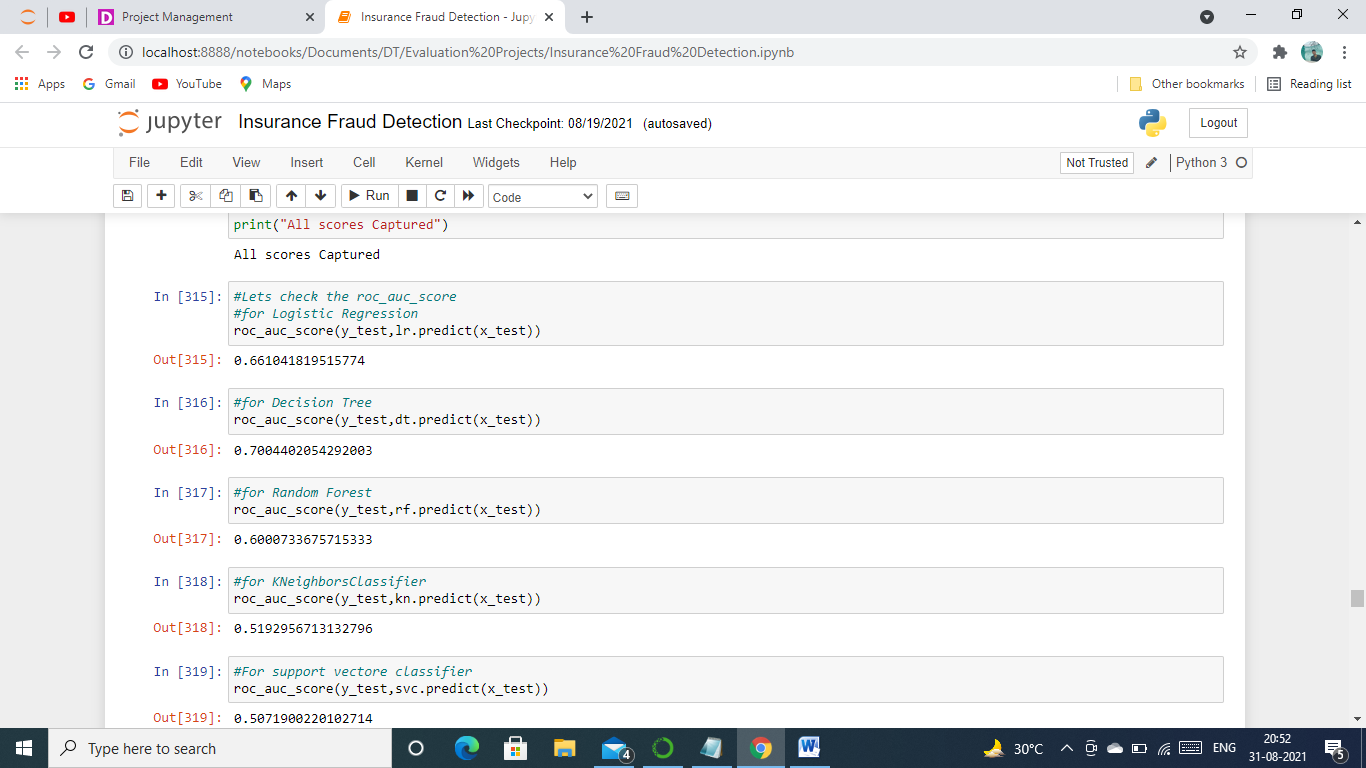


Above we can see that dataset into train and test sets using train\_test\_split method, kept the test size to 20% and we used Stratify=y to balance the label as we saw above that there was imbalance in it.

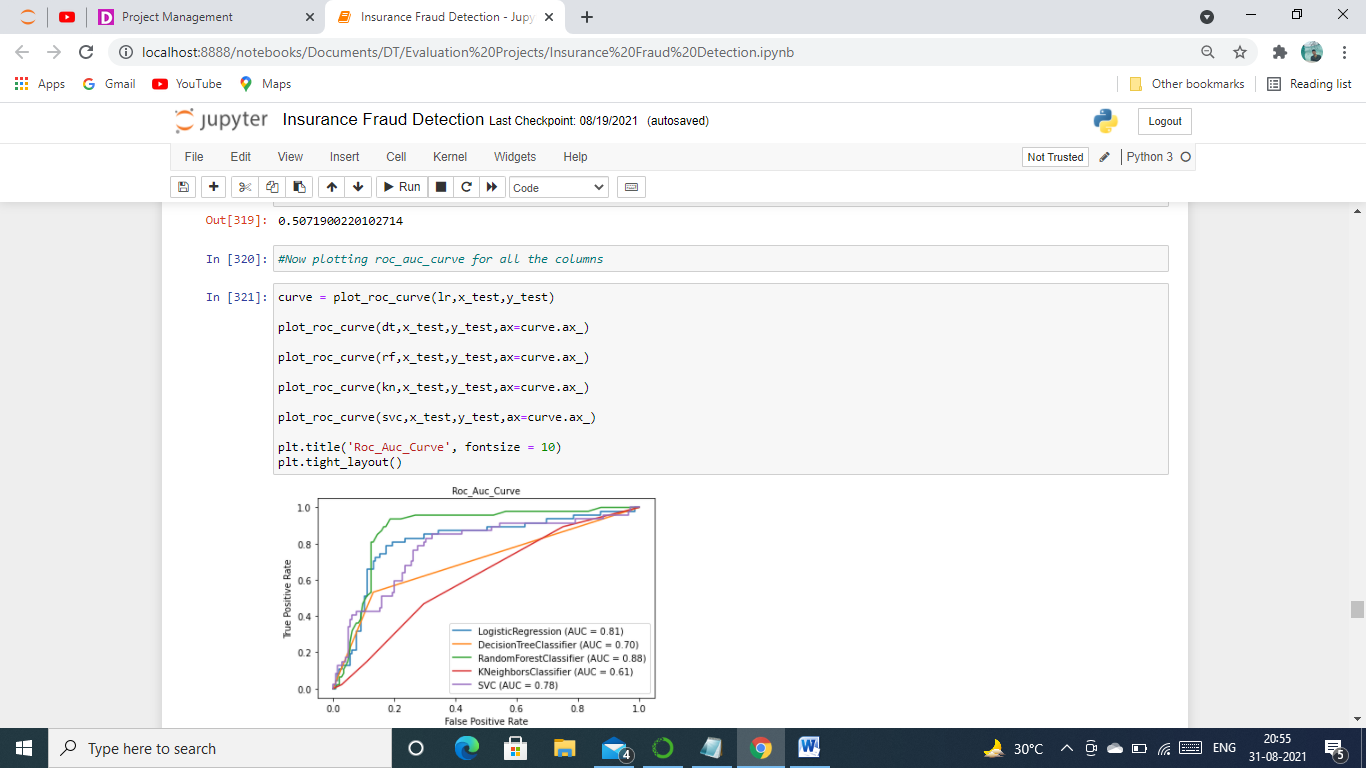
Let’s train the model and capture all the scores. We can do that using below codes.



Now that our model is trained using all the all the algorithms, we will now see how good it performs using roc\_auc metric which we imported previously. Below are the codes and output for each algorithm.

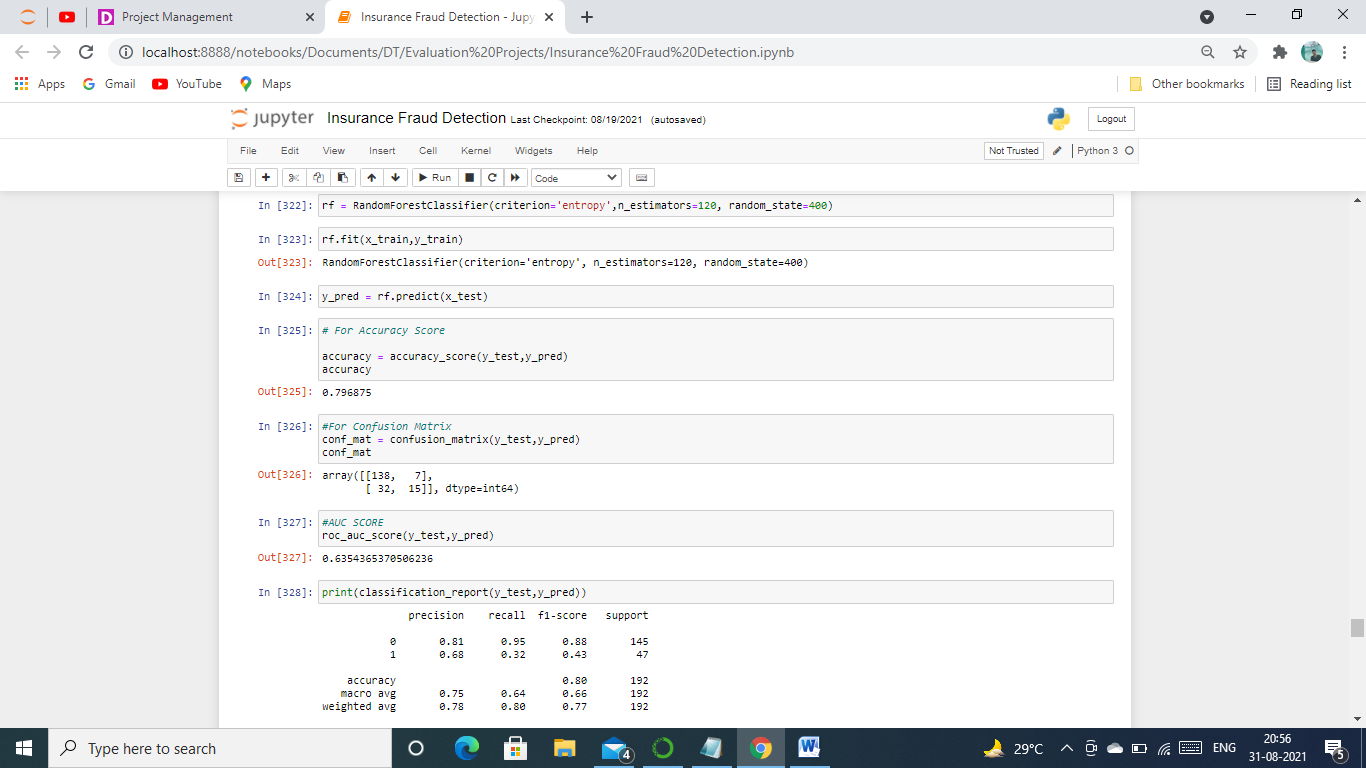


We see that Decision Tree model is giving us the best predictions which is of 70.04% on this dataset. We will now plot an roc\_auc curve for all the algorithms and see which of them is covering most of the area, model which covers most of the area is considered to be good for classification problems. Below is the code for plotting it and the outcome.

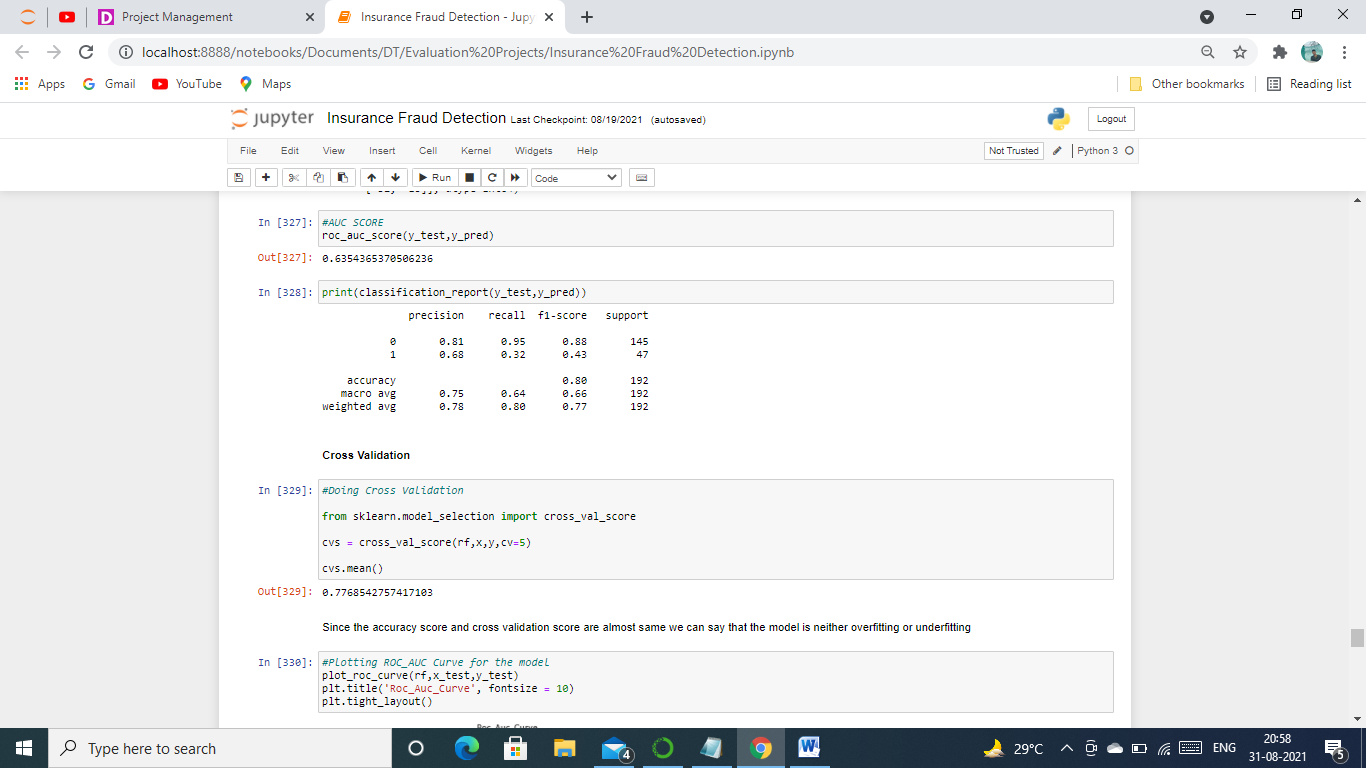


With above plotting we can see area covered by Random Forest Model is the most which is 88% hence he will chose it for further evaluation and tuning it.

Let’s now see all the scores of our RandomForest model, below are the codes and scores for it.



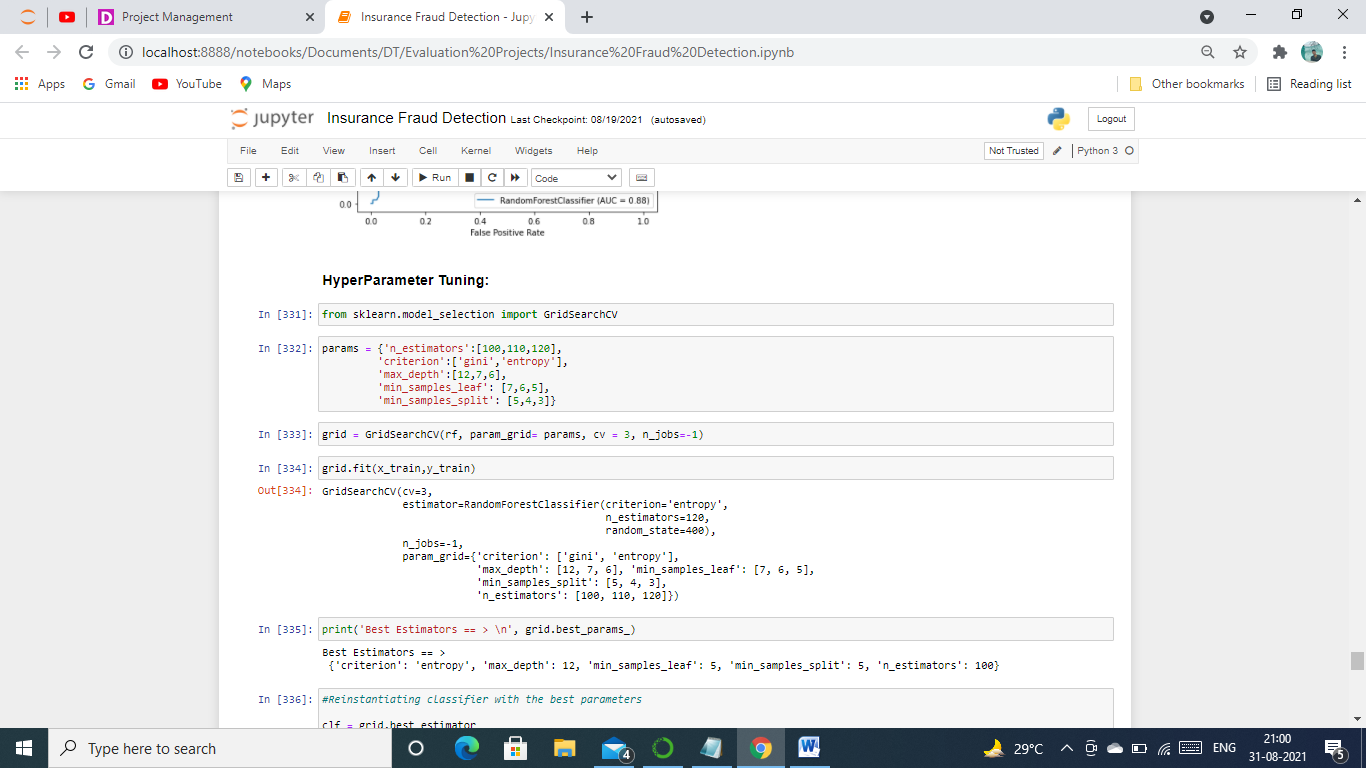
After this we will cross validate our model to see if it is overfitting or underfitting, we will import required library and then apply it on the features and label to get the mean of the score, We can do this with the following code.



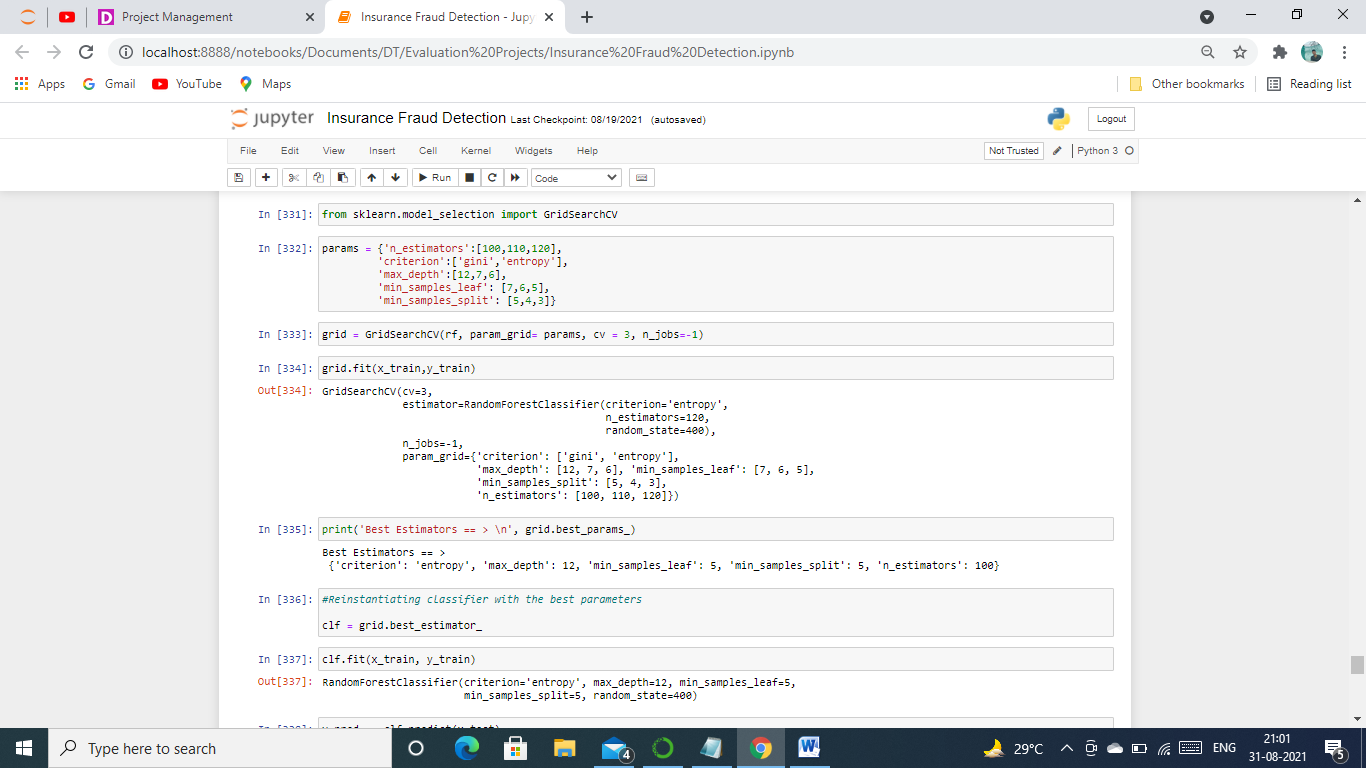
As we saw that there is hardly any difference between our cross validation score and accuracy score we can now say that our model is good, it is neither overfitting nor underfitting.

Next step would be to do Hyperparameter tuning to improve our scores. For this we will use GridSearchCV from Sciket-Learn library to do an exhaustive search over specified parameter values for an estimator and playing around with the parameters which we can find from the documentation of [Random Forest Classifier](sklearn.ensemble.RandomForestClassifier%20—%20scikit-learn%200.24.2%20documentation.html) available on Scikit Learn Website.

Code for importing GridSearch is below and we will use the following parameters.

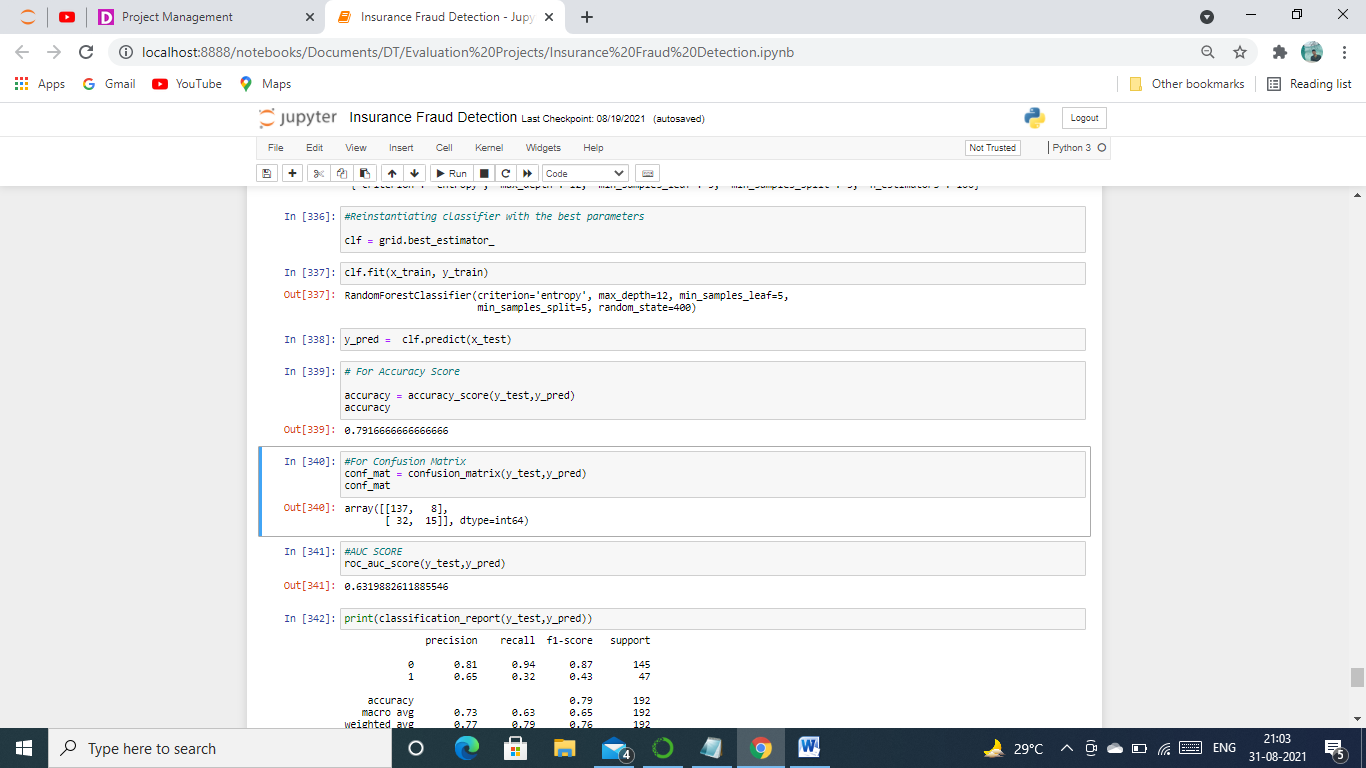


Using GridSearchCV to find the optimal parameters, below are the codes for it and the best parameters which we got out of the above taken parameters for this dataset.



Above we can see the optimal parameters for this dataset. We will now re-instantiate with the best parameters that we got and find the all the scores using metrics imported previously.

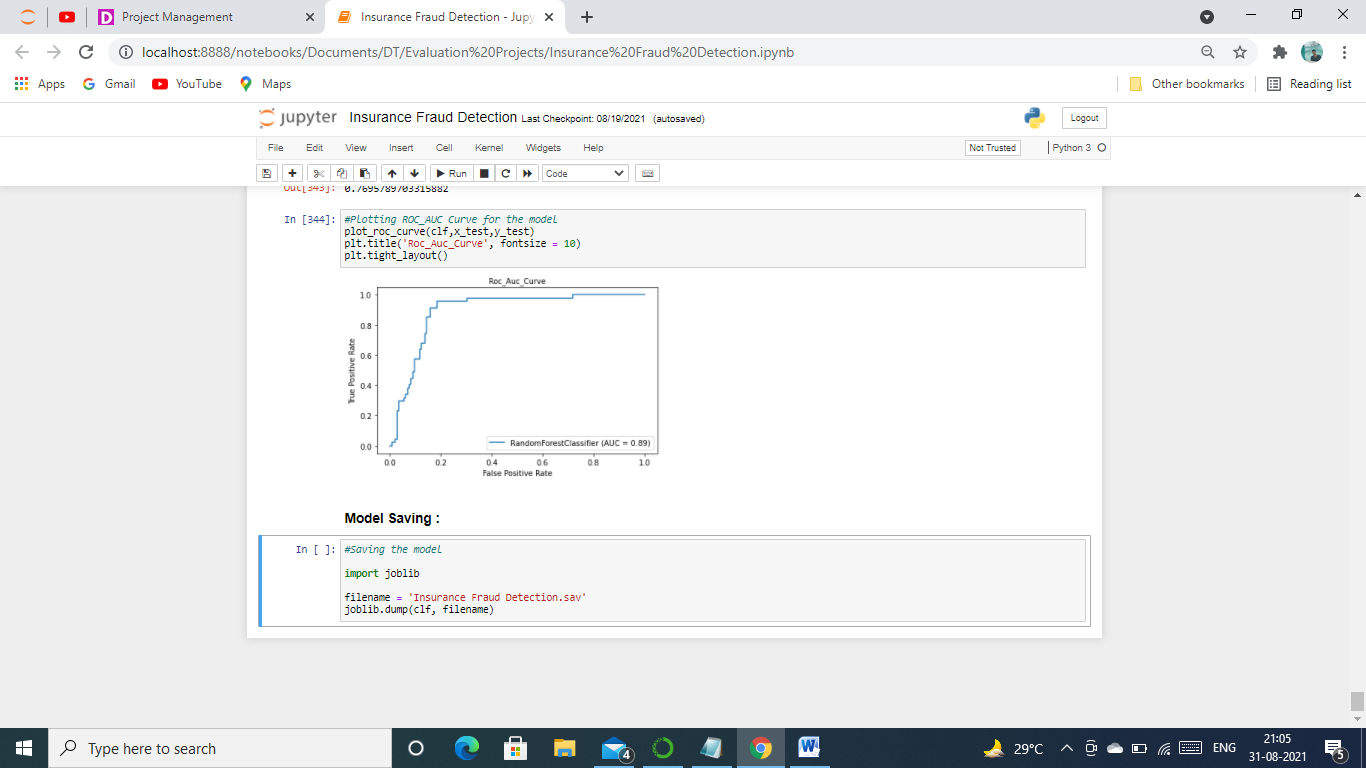
Below are the codes and scores for using all the metrics.



After tuning it with the best parameters, we didn’t get much difference between the scores.

**Saving the model**

We have done all the hard work of creating and testing the model. It would be good if we could save the model for future use rather than retrain it, for this we will first import [Joblib](https://joblib.readthedocs.io/en/latest/) and then save our model with the name Insurance Fraud Detection.



# Final Remarks

We have learned to build a complete machine learning model. In the process, We visualised, cleaned and built a model which will detect fraud in auto-insurance industry with an accuracy of approximately 80%, taking various independent variables or we can say features as input . We also learned to improve the accuracy score by tuning our model using various parameters and save it for further use.

I hope this blog helped you in somewaty and feel free to let me know if there are other changes that could be done to improve the scores and building a better model.

Thank you for reading the blog!