Machine Learning Driven Network Protection in Cloud Computing Environments

MSc Research Project
Cloud Computing

Sddesh Hiregowja Kumara
Student ID:22160345

School of Computing
National College of Ireland

Supervisor: Dr.Ahmed Makki

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Siddesh Hiregowja Kumara
Student ID:	x22160345
Programme:	M.Sc Cloud Computing
Year:	2023
Module:	MSc Research Project
Supervisor:	Dr.Ahmed Makki
Submission Due Date:	14/12/20203
Project Title:	Configuration Manual
Word Count:	613
Page Count:	10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Siddesh Hiregowja Kumara
Date:	27th January 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Machine Learning Driven Network Protection in Cloud Computing Environments

Siddesh Hiregowja Kumara
22160345

1 Introduction

This document is like a detailed guidebook that aims to help the reader successfully install, set up, and run a project. It provides step-by-step instructions, starting from the installation of different parts of the project and explaining how to configure them. The document also gives an overview of how all the pieces of the system fit together, so the reader understands the big picture. Once the system is set up, it explains how different parts work together when the system is running. Additionally, it includes tips on fixing common problems that may arise during installation or use. If someone follows the instructions in this guide, they should be well-prepared to launch the project in a new environment. It's strongly recommended for anyone planning to use the system to read and understand this document thoroughly.

2 System configuration

This part is about a document that helps you make software by giving you step-by-step instructions. It outlines the application broadly, covering its purpose, parameters, interactions with the environment and users, along with the required hardware and software.

2.1 Hardware Requirements

- **Brand and Model:**Acer Extensa
- **Processor:**11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz (2.42 GHz)
- **Installed RAM:**8.00 GB (7.78 GB usable)
- **Device ID:**6E734670-82CB-4251-B27C-502C03C2F605
- **Product ID:**00356-24573-78890-AAOEM
- **System Type:**64-bit operating system.
- **Processor Architecture:**x64-based processor

2.2 Software Requirements

- **Google Colaboratory:**
  Type:Cloud-based Jupyter notebook
  Python Version:3.8 Radovanovic (2022)

- **Email:**
  Requirement:Gmail account for accessing the drive.

- **Browser:**
  Compatibility:Any web browser.

- **Other Software:**
  Specific Requirement:Word.

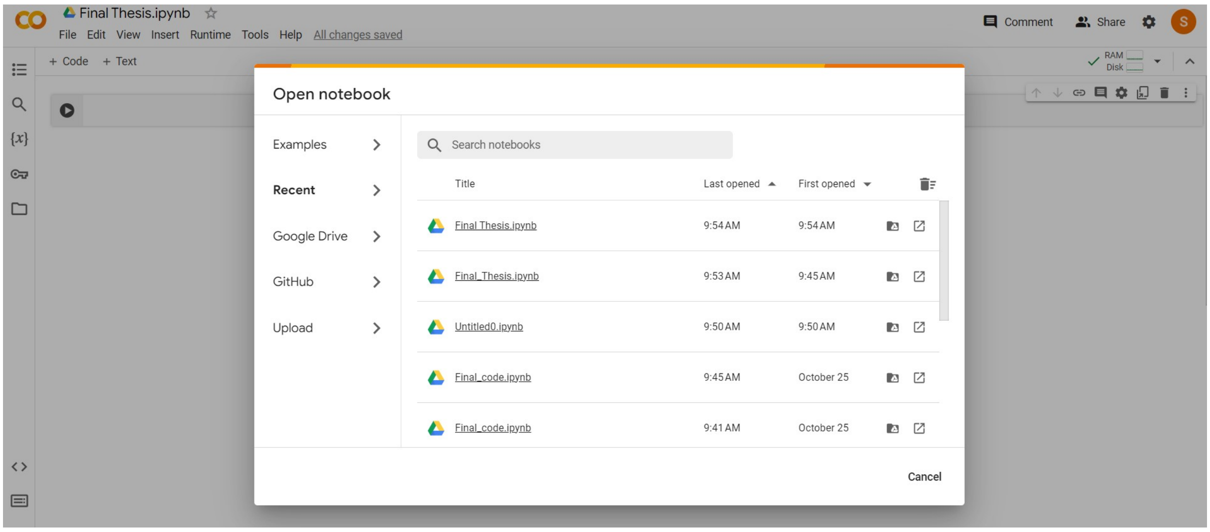
3 Project Execution

3.1 Arranging the environment

Google Colaboratory is the tool we use to create models. You'll need a Gmail account to access it, and we used Python version 3.10.5 for the entire process of making the model.as shown in figure 1

- **Step 1:** Open Google Colaboratory by clicking on the provided link.
  Link:https://research.google.com/colaboratory/

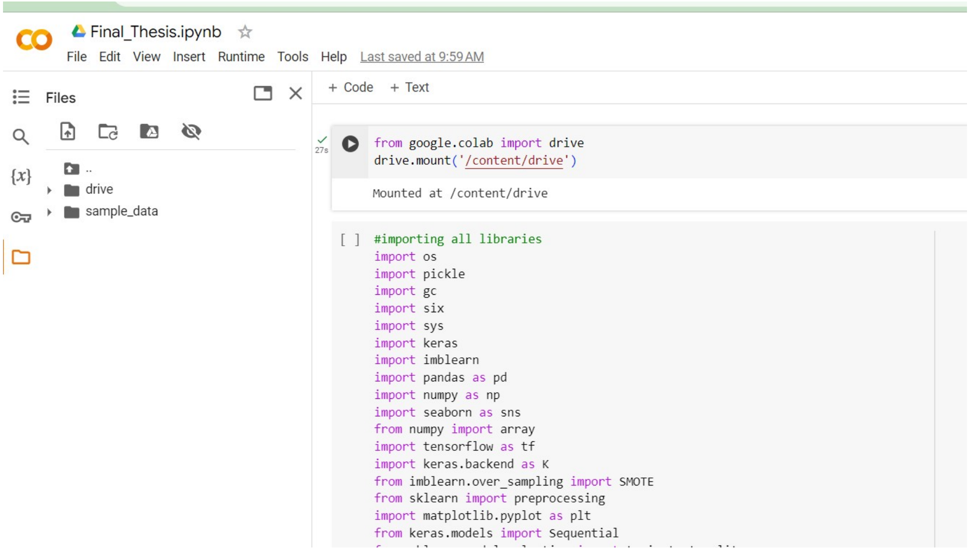
- **Step 2:** Go to the file section and start a new notebook.



Title	Last opened	First opened
Final Thesis.ipynb	9:54 AM	9:54 AM
Final_Thesis.ipynb	9:53 AM	9:45 AM
Untitled0.ipynb	9:50 AM	9:50 AM
Final_code.ipynb	9:45 AM	October 25
Final_code.ipynb	9:41 AM	October 25

Figure 1: Creation a new notebook

- **Step 3:** Go to the "upload" section and select the file you want to upload as shown in figure2


Final_Thesis.ipynb
File Edit View Insert Runtime Tools Help Last saved at 9:59 AM
Files
+ Code + Text
drive
sample_data

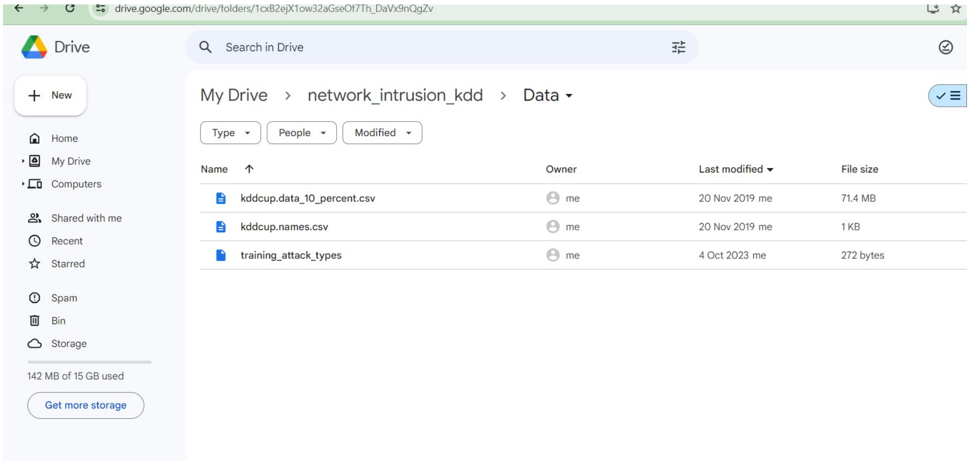
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
#importing all libraries
import os
import pickle
import gc
import six
import sys
import keras
import imblearn
import pandas as pd
import numpy as np
import seaborn as sns
from numpy import array
import tensorflow as tf
import keras.backend as K
from imblearn.over_sampling import SMOTE
from sklearn import preprocessing
import matplotlib.pyplot as plt
from keras.models import Sequential
```

Figure 2: Uploading the files.

- **Step 4:**Importing the Dataset as shown in figure3



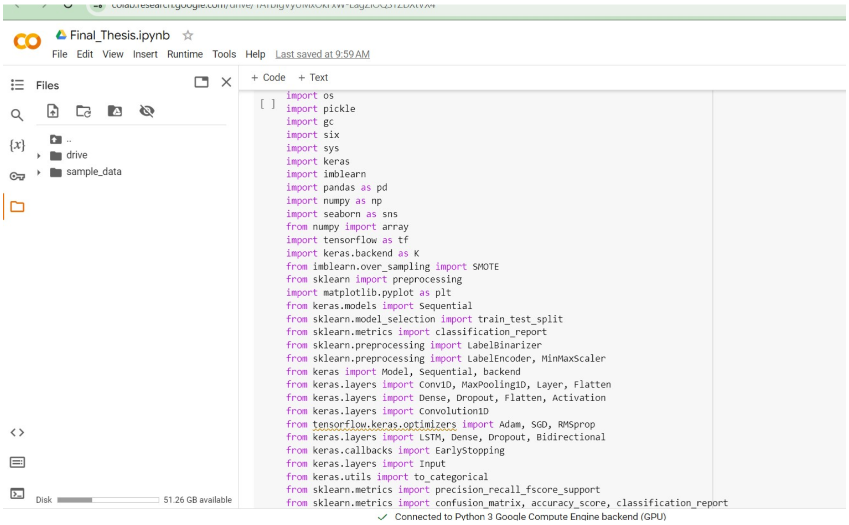
Name	Owner	Last modified	File size
kddcup.data_10_percent.csv	me	20 Nov 2019 me	71.4 MB
kddcup.names.csv	me	20 Nov 2019 me	1 KB
training_attack_types	me	4 Oct 2023 me	272 bytes

Figure 3: Dataset

3.2 Packages and libraries

Once the uploaded dataset is successfully, make sure the following libraries are imported before moving on to the code implementation as shown in figure 4
The following are the libraries required for the task at hand:

os, pickle, gc, six, sys, Keras, imblearn, pandas, numpy, seaborn, tensorflow, SMOTE, preprocessing, matplotlib.pyplot, Sequential, train test split, classification report, LabelBinarizer, LabelEncoder, MinMaxScaler, Model, backend, MaxPooling1D, Layer, Flatten, Dense, Dropout, Activation, Convolution1D, Adam, SGD, RMSprop, LSTM, Bidirectional, EarlyStopping.*Python Tutorial* (n.d.)


Final_Thesis.ipynb
File Edit View Insert Runtime Tools Help Last saved at 9:59 AM
Files
+ Code + Text
drive
sample_data
Disk 51.26 GB available
Connected to Python 3 Google Compute Engine backend (GPU)

```
import os
import pickle
import gc
import six
import sys
import keras
import imblearn
import pandas as pd
import numpy as np
import seaborn as sns
from numpy import array
import tensorflow as tf
import keras.backend as K
from imblearn.over_sampling import SMOTE
from sklearn import preprocessing
import matplotlib.pyplot as plt
from keras.models import Sequential
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.preprocessing import LabelBinarizer
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
from keras import Model, Sequential, backend
from keras.layers import Conv1D, MaxPooling1D, Layer, Flatten
from keras.layers import Dense, Dropout, Flatten, Activation
from keras.layers import Convolution1D
from tensorflow.keras.optimizers import Adam, SGD, RMSprop
from keras.layers import LSTM, Dense, Dropout, Bidirectional
from keras.callbacks import EarlyStopping
from keras.layers import Input
from keras.utils import to_categorical
from sklearn.metrics import precision_recall_fscore_support
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
```

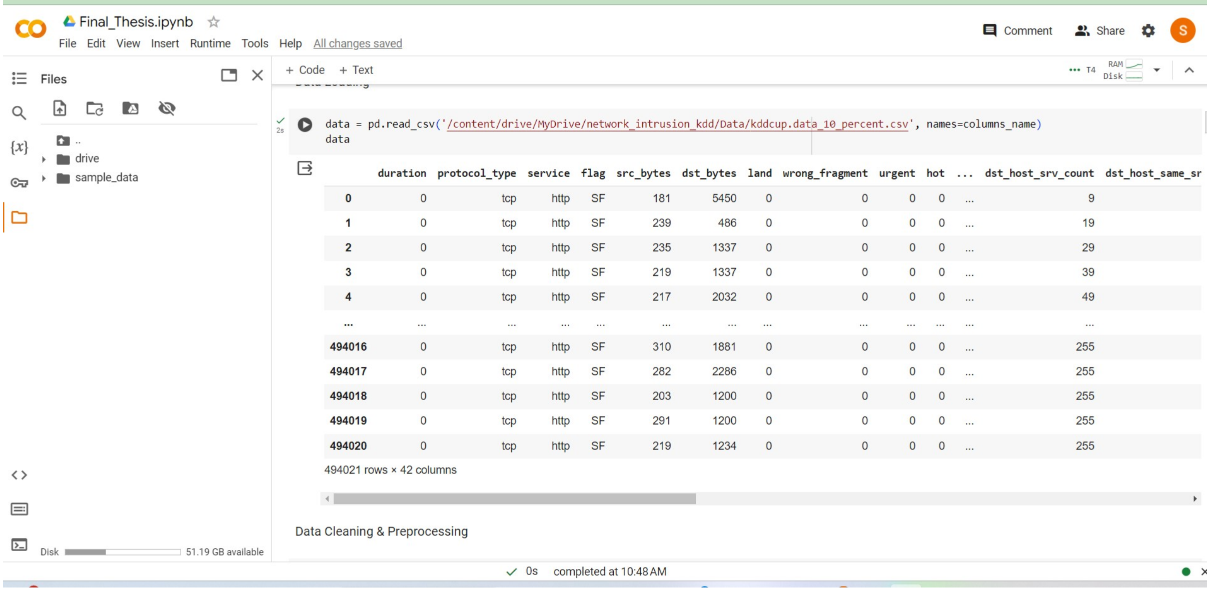
Figure 4: Required Libraries

4 Phases

The paragraphs that follow detail the entire research process, covering everything from collecting data to Exploratory Data Analysis.

4.1 Collecting data

- Reading a dataset as show in figure 5



```
data = pd.read_csv('/content/drive/MyDrive/network_intrusion_kdd/Data/kddcup.data_10_percent.csv', names=columns_name)
data
```

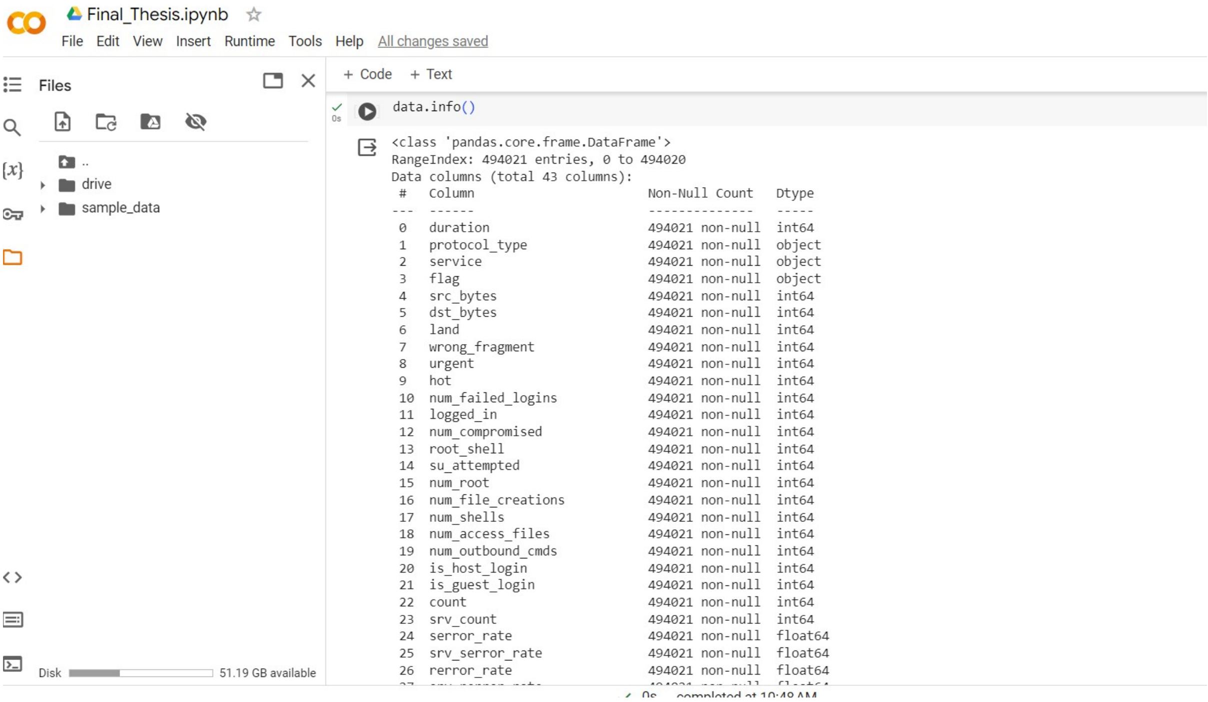
	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count	dst_host_same_sr
0	0	tcp	http	SF	181	5450	0	0	0	0	...	9	
1	0	tcp	http	SF	239	486	0	0	0	0	...	19	
2	0	tcp	http	SF	235	1337	0	0	0	0	...	29	
3	0	tcp	http	SF	219	1337	0	0	0	0	...	39	
4	0	tcp	http	SF	217	2032	0	0	0	0	...	49	
...	...	...	...	...	...	...	...	...	...	...	...	...	
494016	0	tcp	http	SF	310	1881	0	0	0	0	...	255	
494017	0	tcp	http	SF	282	2286	0	0	0	0	...	255	
494018	0	tcp	http	SF	203	1200	0	0	0	0	...	255	
494019	0	tcp	http	SF	291	1200	0	0	0	0	...	255	
494020	0	tcp	http	SF	219	1234	0	0	0	0	...	255	

494021 rows × 42 columns

Data Cleaning & Preprocessing

Figure 5: Reading the Data

- Display the details of the dataset.figure 6



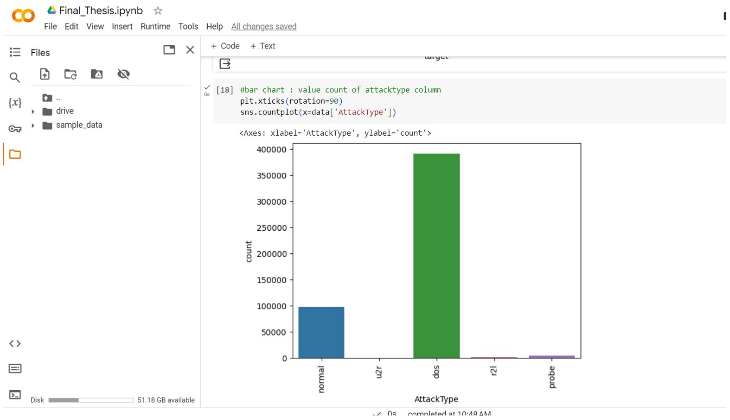
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 494021 entries, 0 to 494020
Data columns (total 43 columns):
 #   Column                      Non-Null Count   Dtype
---  ------                      --------------   -----
 0   duration                    494021 non-null  int64
 1   protocol_type               494021 non-null  object
 2   service                     494021 non-null  object
 3   flag                        494021 non-null  object
 4   src_bytes                   494021 non-null  int64
 5   dst_bytes                   494021 non-null  int64
 6   land                        494021 non-null  int64
 7   wrong_fragment              494021 non-null  int64
 8   urgent                      494021 non-null  int64
 9   hot                         494021 non-null  int64
 10  num_failed_logins           494021 non-null  int64
 11  logged_in                   494021 non-null  int64
 12  num_compromised             494021 non-null  int64
 13  root_shell                  494021 non-null  int64
 14  su_attempted                494021 non-null  int64
 15  num_root                    494021 non-null  int64
 16  num_file_creations          494021 non-null  int64
 17  num_shells                  494021 non-null  int64
 18  num_access_files            494021 non-null  int64
 19  num_outbound_cmds           494021 non-null  int64
 20  is_host_login               494021 non-null  int64
 21  is_guest_login              494021 non-null  int64
 22  count                       494021 non-null  int64
 23  srv_count                   494021 non-null  int64
 24  serror_rate                 494021 non-null  float64
 25  srv_serror_rate             494021 non-null  float64
 26  rerror_rate                 494021 non-null  float64
```

Figure 6: Show details about the dataset.

4.2 Data Exploration and Analysis

- Showing a bar graph that categorizes attacks as shown in figure 7 and figure 8



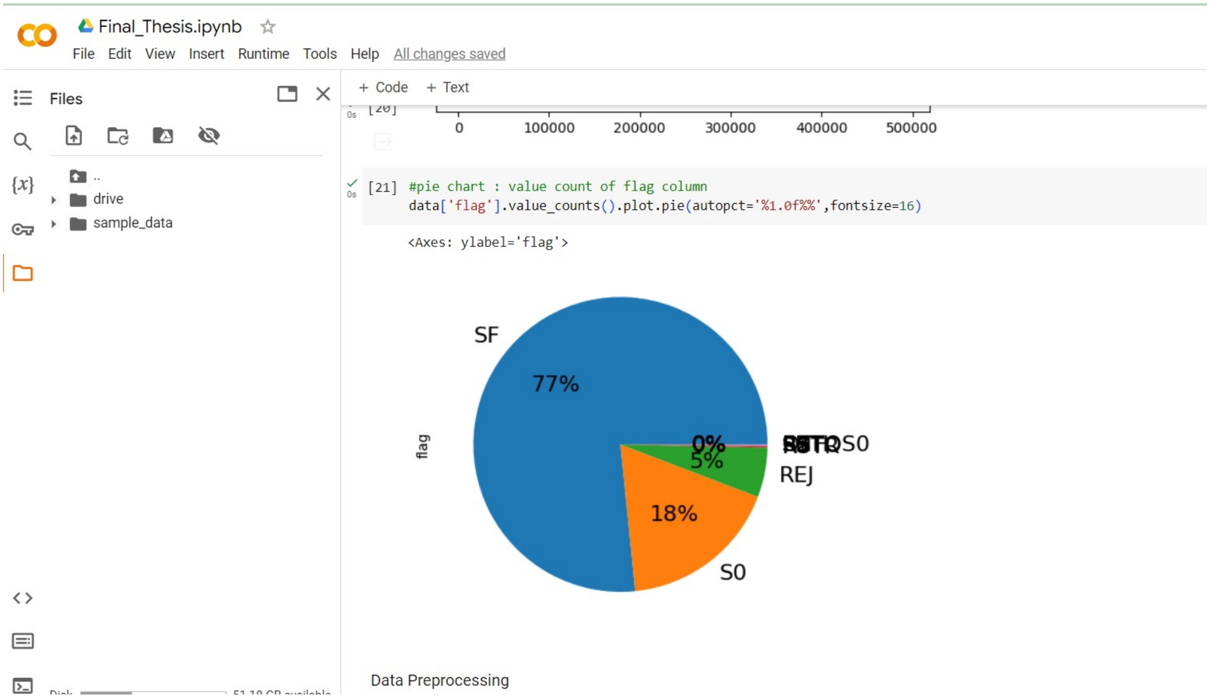
```
[18] #bar chart : value count of attacktype column
plt.xticks(rotation=90)
sns.countplot(x=data['AttackType'])
```

```
<Axes: xlabel='AttackType', ylabel='count'>
```

Figure 7: Number of Attacks

5 Data Pre-processing

The process of using deep learning to protect computer networks in the cloud. It starts by collecting information on network behavior, categorizing activities into five groups, and using the KDD Cup 1999 dataset. Visual charts are created to understand the distribution of connections and attacks, revealing a class imbalance. and introduce SMOTE

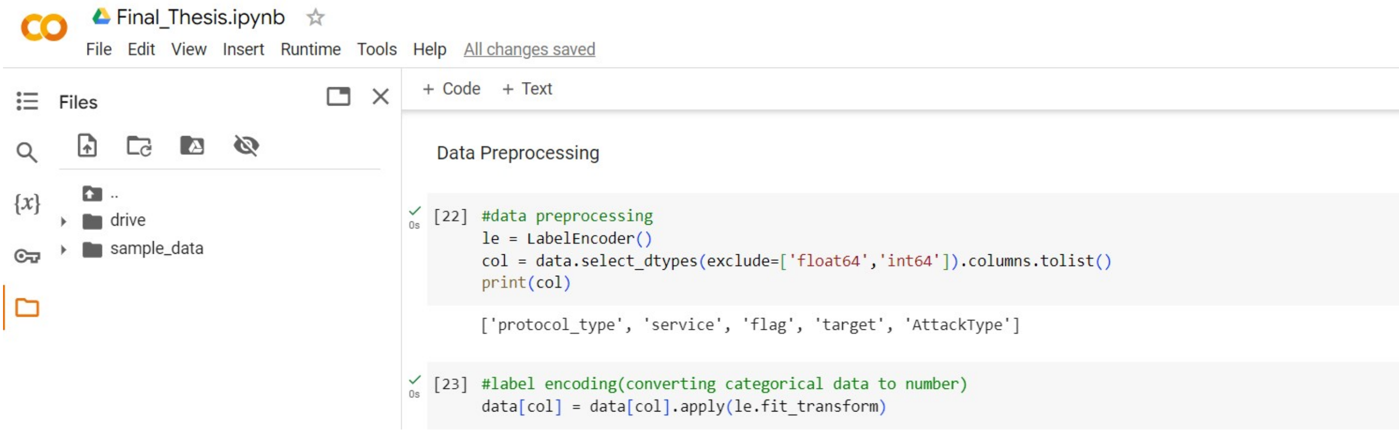


```
[21] #pie chart : value count of flag column
     data['flag'].value_counts().plot.pie(autopct='%1.0f%%',fontsize=16)

     <Axes: ylabel='flag'>
```

Data Preprocessing

Figure 8: Percentage of attacks

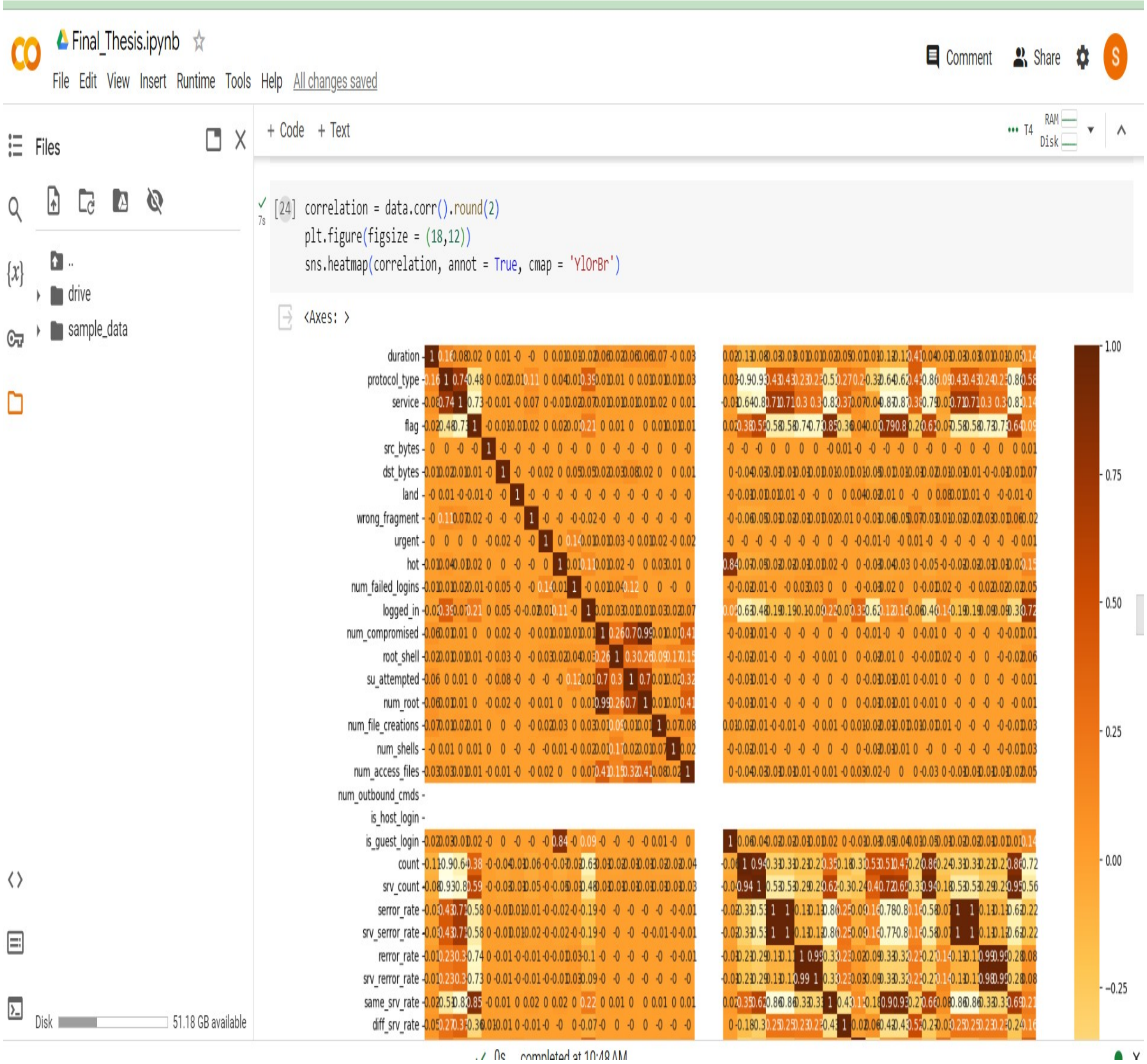
to address this imbalance. The ultimate goal is to create a system using a well-prepared dataset to predict and distinguish between normal and harmful network connections. The additional context on data label encoding and heatmap visualization highlights their crucial roles in preparing the dataset for machine learning. Data label encoding involves converting categorical labels into numerical representations for easier processing, while a heatmap visually represents correlations between different features in the dataset,to the identification of patterns and relationships.shown in figure 9 and figure 10



Data Preprocessing

```
[22] #data preprocessing
     le = LabelEncoder()
     col = data.select_dtypes(exclude=['float64','int64']).columns.tolist()
     print(col)

     ['protocol_type', 'service', 'flag', 'target', 'AttackType']

[23] #label encoding(converting categorical data to number)
     data[col] = data[col].apply(le.fit_transform)
```

Figure 9: label encoding



```
[24] correlation = data.corr().round(2)
     plt.figure(figsize = (18,12))
     sns.heatmap(correlation, annot = True, cmap = 'YlOrBr')
```

```
<Axes: >
```

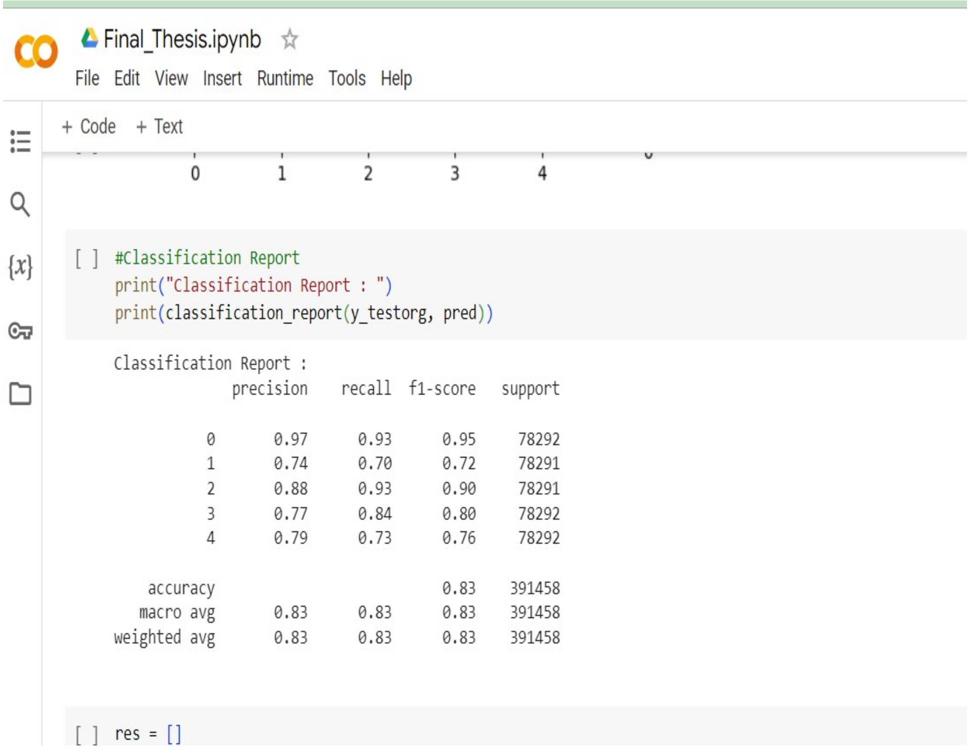
Figure 10: HeatMap

6 Deep learning models with Results

LSTM (Long Short-Term Memory) model:is a type of recurrent neural network (RNN) designed to capture and remember long-term dependencies in data.involves creating a neural network architecture with LSTM layers that process sequential data while minimizing the risk of copying existing work. This originality in the model's predictions.the LSTM model consistently displayed its capabilities, securing a validation accuracy of 82.77%.shown in figure 11,

Bidirectional Long Short-Term Memory (BILSTM):network operates by processing sequential data bidirectionally—forward and backward. It employs Long Short-Term Memory (LSTM) units to capture long-term dependencies in the input sequences. During the forward pass, the network processes the input data from the beginning to the end, while simultaneously, the backward pass processes the data in reverse. The outputs from both passes are then combined to create a comprehensive representation that incorporates information from both past and future contexts. This bidirectional approach is particularly valuable in tasks like network security, enabling the model to effectively capture patterns and dependencies over time for accurate predictions.The BILSTM model,in a parallel accomplished a commendable validation accuracy of 90.16%..shown in figure 12

Bidirectional Long Short-Term Memory (BILSTM) with Attention Mechanism model: operates by processing input sequences in both forward and backward directions, capturing contextual information from both past and future elements. The Long Short-Term Memory (LSTM) component, a type of recurrent neural network, is adept at preserving long-term dependencies. Simultaneously, the Attention Mechanism enhances the model's focus on specific elements within the sequence, assigning weights to highlight their relevance during predictions. This combination allows the model to make informed and contextually rich predictions, making it particularly effective in tasks involving sequential data, such as natural language processing, where understanding context and dependencies is crucial for accurate predictions.In contrast, the BILSTM with Attention Mechanism outshone both, attaining a remarkable validation accuracy of 99.47%. It's worth highlighting that the standout performer, boasting the highest accuracy, is BiLSTM with Attention.shown in figure 13

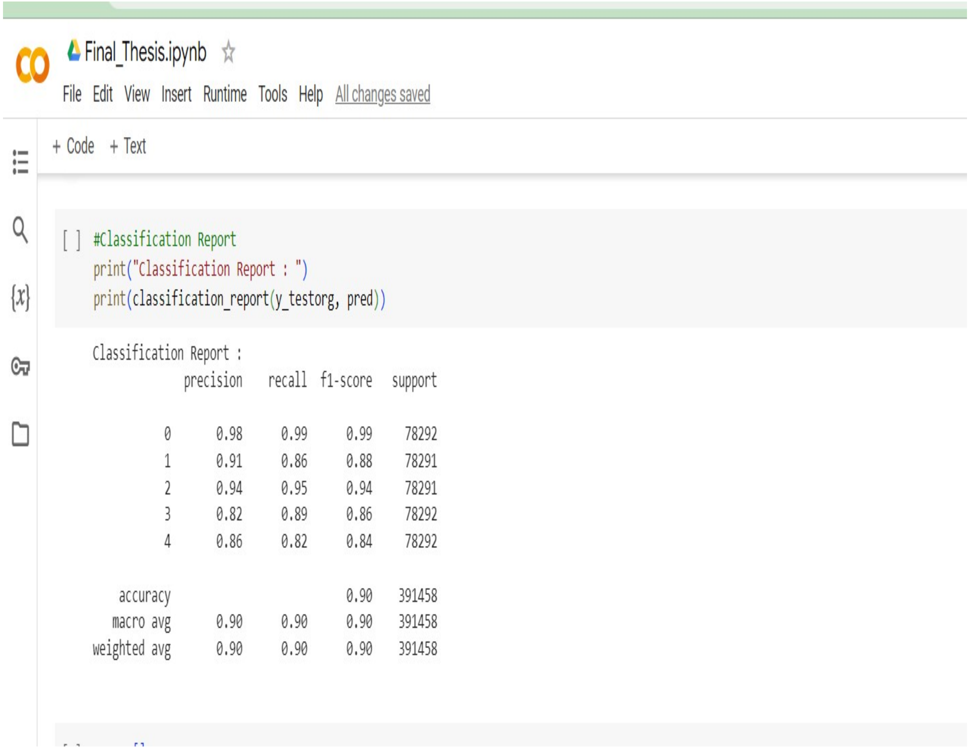


```
[ ] #Classification Report
    print("Classification Report : ")
    print(classification_report(y_testorg, pred))

Classification Report :
              precision    recall  f1-score   support

           0       0.97      0.93      0.95     78292
           1       0.74      0.70      0.72     78291
           2       0.88      0.93      0.90     78291
           3       0.77      0.84      0.80     78292
           4       0.79      0.73      0.76     78292

    accuracy                           0.83    391458
   macro avg       0.83      0.83      0.83    391458
weighted avg       0.83      0.83      0.83    391458

[ ] res = []
```

Figure 11: LSTM model accuracy score

References

Python Tutorial (n.d.).
URL: *https://www.tutorialspoint.com/python/index.htm*

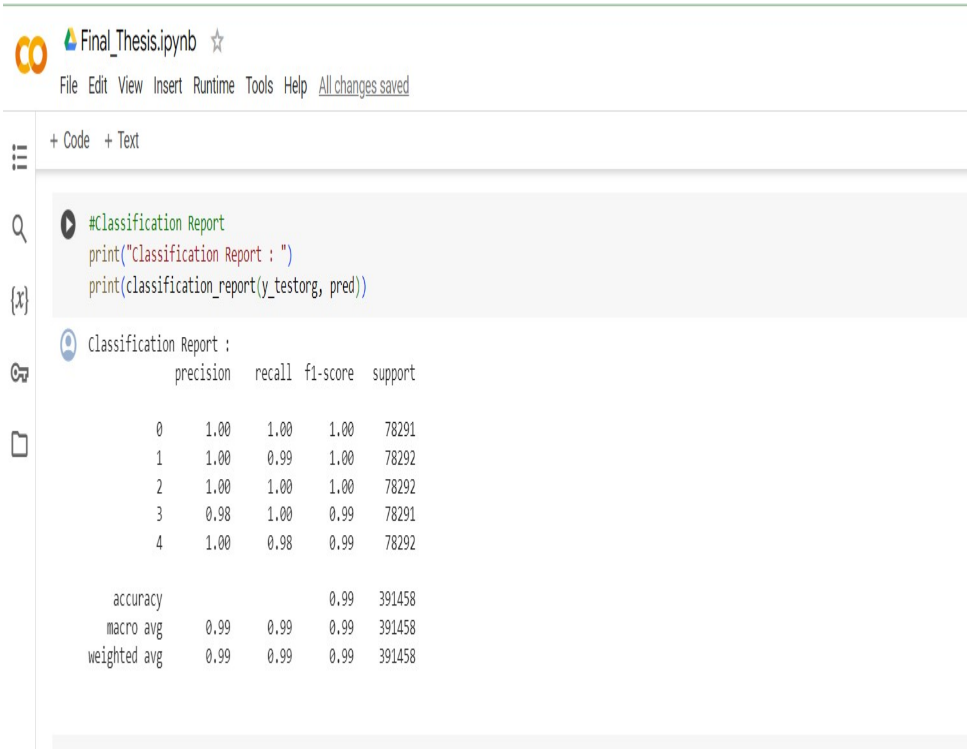


```
#Classification Report
print("Classification Report : ")
print(classification_report(y_testorg, pred))
```

```
Classification Report :
              precision    recall  f1-score   support

           0       0.98      0.99      0.99     78292
           1       0.91      0.86      0.88     78291
           2       0.94      0.95      0.94     78291
           3       0.82      0.89      0.86     78292
           4       0.86      0.82      0.84     78292

    accuracy                           0.90    391458
   macro avg       0.90      0.90      0.90    391458
weighted avg       0.90      0.90      0.90    391458
```

Figure 12: BILSTM model accuracy score

Radovanovic, I. (2022). Google colab – a step-by-step guide.
URL: *https://algotrading101.com/learn/google-colab-guide/*



```
#Classification Report
print("Classification Report : ")
print(classification_report(y_testorg, pred))
```

```
Classification Report :
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     78291
           1       1.00      0.99      1.00     78292
           2       1.00      1.00      1.00     78292
           3       0.98      1.00      0.99     78291
           4       1.00      0.98      0.99     78292

    accuracy                           0.99    391458
   macro avg       0.99      0.99      0.99    391458
weighted avg       0.99      0.99      0.99    391458
```

Figure 13: BILSTM with Attention Mechanism Model accuracy score