Tutorial-Section A

Question 1 [24 marks]

An IT team of an organization was tasked to assist in analyzing the cyber security posture of an organization. Among the tasks involved, the team needs to build a classification model using Logistic Regression to detect and classify network traffic instances as normal or malicious.

The dataset includes the following attributes.

- Source IP address
- Destination IP address
- Protocol (TCP, UDP, ICMP, etc.)
- Source port
- Destination port
- Packet length
- Time of day
- Flag (SYN, ACK, RST)
- Type of service
- Label (N - Normal or I - Intrusion)

The following code is executed that shows the first few rows of the data.

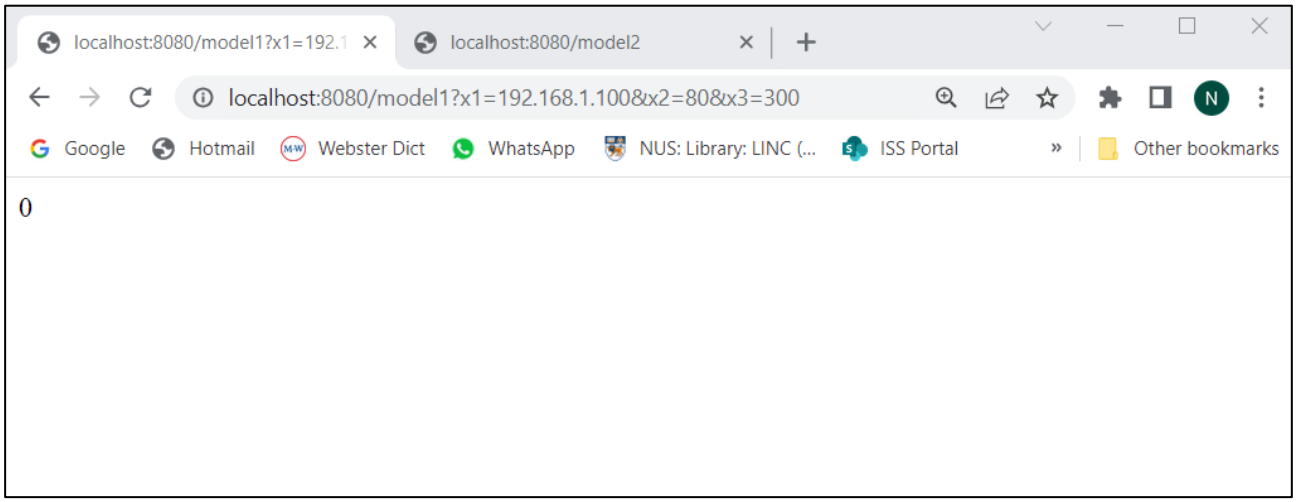
```
df.head()
```

	source_ip	source_port	destination_ip	destination_port	protocol	packet_length	time_of_incidence	tcp_flag	label
0	192.168.1.100	8081	203.0.113.10	80	TCP	300	9:30AM	SYN	N
1	10.0.0.1	8081	203.0.113.10	51	UDP	200	10:30AM	SYN	N
2	10.0.0.2	8080	203.0.113.10	80	TCP	100	11:30AM	SYN	N
3	10.0.0.3	5678	203.0.113.10	51	TCP	120	11:40PM	SYN	N
4	10.0.0.4	5679	203.0.113.10	50	TCP	240	10:00PM	SYN	N

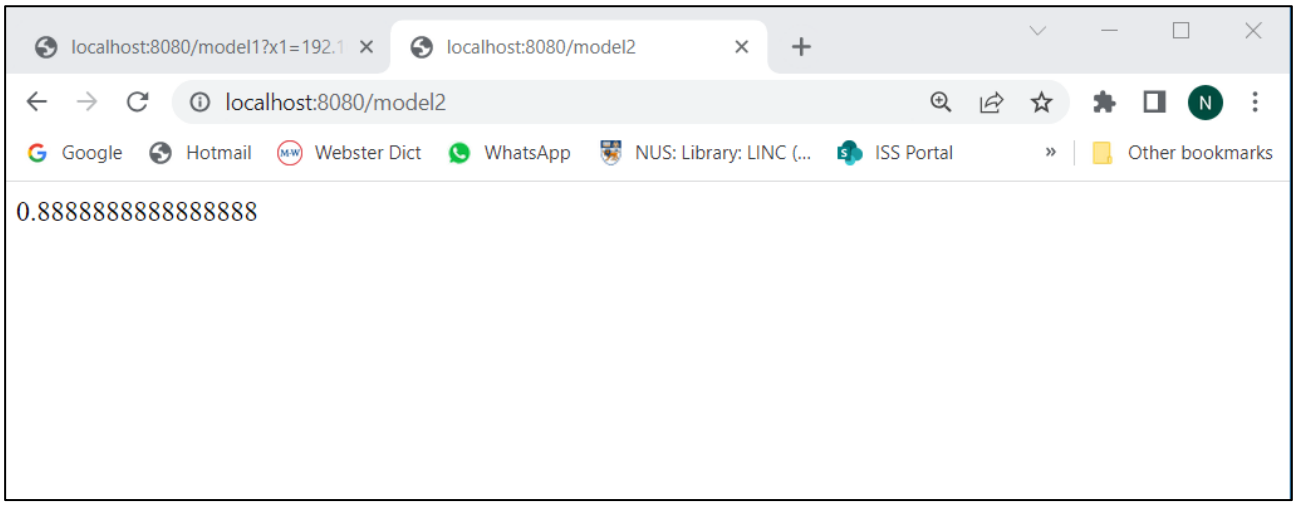
In the first phase (and also for the scope of this question), the following features are selected to train a Logistic Regression model.

- Source IP address
- Source Port
- Packet length

The team shall train the model, load the model into a Python Web server, expose the services to allow the user to provide the data of the three features and return the outcome. The deployed model shall also enable the user to enquire the accuracy of the model.



Calling the services to classify a set of values with 3 features (Source IP address, Source port and Packet length)



Calling the services to retrieve the accuracy of the model

Answer the following question. You are required to write Python code.

a) Train a Logistic Regression model with the input dataset 'NID.csv'.

(4 marks)

b) The data are defined in string format. For example, the feature *source_ip* may contain a value such as '192.168.1.100', and the *source_port* may have a value such as '8080'. Essential encoding shall be performed on the data so that the model will handle numerical values. You may utilize the LabelEncoder class provided by the Sklearn library. Refer below for the documentation of the LabelEncoder class.

(6 marks)

	Sklearn.preprocessing.**LabelEncoder**
Methods	
Fit *(Purpose - Fit the label encoder)*	Returns an instance of itself. Example- `le = preprocessing.LabelEncoder()` `le.fit(["paris", "paris", "tokyo", "amsterdam"])` *`(Output)>>>LabelEncoder()`*
Transform (Purpose - Transform labels to normalized encoding)	Returns encoded values. `le.transform(["tokyo", "tokyo", "paris"])` *`(Output)>>>array([2, 2, 1])`* *Note that the labelEncoder need to be fitted first before calling the transform method.*

c) Your model shall be deployed to a Flask Waitress server.

(4 marks)

d) Code the server that exposed the following two services. The following <u>two</u> services shall be exposed to the web client (as "Get" and "Post").

i. Prediction.

Given the values of the three features, return a predicted label value.

ii. Accuracy.

Allow the user to retrieve the model's accuracy based on the current training data provided in the input dataset.

(10 marks)