nlpndecsionbinary

April 11, 2024

[3]:
```
import pandas as pd
import requests

url = "http://192.168.0.222:3000/csv/
 ↪detecting_testing_applied_rules_3ad8690c-0f99-4f1e-9e91-8e905206be26.csv"

response = requests.get(url)
with open("network_data.csv", "wb") as file:
    file.write(response.content)
```

[4]:
```
pip install tensorflow
```

```
Requirement already satisfied: tensorflow in ./venv/lib/python3.11/site-packages
(2.16.1)
Requirement already satisfied: absl-py>=1.0.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in
./venv/lib/python3.11/site-packages (from tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
./venv/lib/python3.11/site-packages (from tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in
./venv/lib/python3.11/site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=3.10.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (3.11.0)
Requirement already satisfied: libclang>=13.0.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (18.1.1)
Requirement already satisfied: ml-dtypes~=0.3.1 in ./venv/lib/python3.11/site-
packages (from tensorflow) (0.3.2)
Requirement already satisfied: opt-einsum>=2.3.2 in ./venv/lib/python3.11/site-
packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in ./venv/lib/python3.11/site-packages
(from tensorflow) (24.0)
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3
in ./venv/lib/python3.11/site-packages (from tensorflow) (4.25.3)
Requirement already satisfied: requests<3,>=2.21.0 in
```

```
./venv/lib/python3.11/site-packages (from tensorflow) (2.31.0)
Requirement already satisfied: setuptools in ./venv/lib/python3.11/site-packages
(from tensorflow) (66.1.1)
Requirement already satisfied: six>=1.12.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in
./venv/lib/python3.11/site-packages (from tensorflow) (4.11.0)
Requirement already satisfied: wrapt>=1.11.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
./venv/lib/python3.11/site-packages (from tensorflow) (1.62.1)
Requirement already satisfied: tensorboard<2.17,>=2.16 in
./venv/lib/python3.11/site-packages (from tensorflow) (2.16.2)
Requirement already satisfied: keras>=3.0.0 in ./venv/lib/python3.11/site-
packages (from tensorflow) (3.2.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
./venv/lib/python3.11/site-packages (from tensorflow) (0.36.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in
./venv/lib/python3.11/site-packages (from tensorflow) (1.26.4)
Requirement already satisfied: wheel<1.0,>=0.23.0 in ./venv/lib/python3.11/site-
packages (from astunparse>=1.6.0->tensorflow) (0.43.0)
Requirement already satisfied: rich in ./venv/lib/python3.11/site-packages (from
keras>=3.0.0->tensorflow) (13.7.1)
Requirement already satisfied: namex in ./venv/lib/python3.11/site-packages
(from keras>=3.0.0->tensorflow) (0.0.7)
Requirement already satisfied: optree in ./venv/lib/python3.11/site-packages
(from keras>=3.0.0->tensorflow) (0.11.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
./venv/lib/python3.11/site-packages (from requests<3,>=2.21.0->tensorflow)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in ./venv/lib/python3.11/site-
packages (from requests<3,>=2.21.0->tensorflow) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in ./venv/lib/python3.11/site-
packages (from requests<3,>=2.21.0->tensorflow) (2.2.1)
Requirement already satisfied: certifi>=2017.4.17 in ./venv/lib/python3.11/site-
packages (from requests<3,>=2.21.0->tensorflow) (2024.2.2)
Requirement already satisfied: markdown>=2.6.8 in ./venv/lib/python3.11/site-
packages (from tensorboard<2.17,>=2.16->tensorflow) (3.6)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
./venv/lib/python3.11/site-packages (from tensorboard<2.17,>=2.16->tensorflow)
(0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in ./venv/lib/python3.11/site-
packages (from tensorboard<2.17,>=2.16->tensorflow) (3.0.2)
Requirement already satisfied: MarkupSafe>=2.1.1 in ./venv/lib/python3.11/site-
packages (from werkzeug>=1.0.1->tensorboard<2.17,>=2.16->tensorflow) (2.1.5)
Requirement already satisfied: markdown-it-py>=2.2.0 in
```

```
./venv/lib/python3.11/site-packages (from rich->keras>=3.0.0->tensorflow)
(3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
./venv/lib/python3.11/site-packages (from rich->keras>=3.0.0->tensorflow)
(2.17.2)
Requirement already satisfied: mdurl~=0.1 in ./venv/lib/python3.11/site-packages
(from markdown-it-py>=2.2.0->rich->keras>=3.0.0->tensorflow) (0.1.2)
Note: you may need to restart the kernel to use updated packages.
```

[5]:
```python
import numpy as np
import tensorflow as tf
from sklearn.model_selection import train_test_split

# Load the data from a CSV file
data = pd.read_csv("network_data.csv")

# Preprocess the payload data
def transform_payload(payload):
    if pd.isna(payload):
        return np.zeros(3000)
    numbers = payload.split(" ")
    payload_array = np.zeros(3000)
    for i, num in enumerate(numbers):
        if num != "" and i < 3000:
            payload_array[i] = float(num)
    return payload_array

payload_columns = ["payload" + str(i) for i in range(3000)]
payload_data = pd.DataFrame(data["payload"].apply(transform_payload).tolist(),␣
 ↪columns=payload_columns)
data = pd.concat([data, payload_data], axis=1)
data = data.drop("payload", axis=1)
```

[7]:
```python
print("rfrr?")
```

```
rfrr?
```

[ ]:
```python

```

[9]:
```python
# Split the data into features and target
X = data[payload_columns].values
y = data[["http_scan", "ping_scan", "nmap_scan", "unmalicious",␣
 ↪"maybemalicious", "malicious"]].values

# Split the data into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

```
# Build the TensorFlow model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation="relu", input_shape=(3000,)),
    tf.keras.layers.Dense(64, activation="relu"),
    tf.keras.layers.Dense(6, activation="sigmoid")
])

model.compile(optimizer="adam", loss="binary_crossentropy",␣
 ↪metrics=["accuracy"])

# Train the model
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test,␣
 ↪y_test))

# Save the model
model.save("network_model.h5")

# Function to make predictions
def predict(data):
    payload_data = pd.DataFrame(data["payload"].apply(transform_payload).
 ↪tolist(), columns=payload_columns)
    predictions = model.predict(payload_data.values)
    return predictions

# Example usage
input_data = pd.DataFrame({
    "timestamp": ["2024-04-09T04:26:01+10:00"],
    "source_ip": ["192.168.0.222"],
    "destination_ip": ["192.168.0.148"],
    "protocol": ["TCP"],
    "length": [206],
    "payload": ["72 84 84 80 47 49 46 49 32 50 48 48 32 79 75 13 10 68 97 116␣
 ↪101 58 32 77 111 110 44 32 48 56 32 65 112 114 32 50 48 50 52 32 49 56 58 50␣
 ↪54 58 48 49 32 71 77 84 13 10 67 111 110 116 101 110 116 45 84 121 112 101␣
 ↪58 32 116 101 120 116 47 112 108 97 105 110 59 32 99 104 97 114 115 101 116␣
 ↪61 117 116 102 45 56 13 10 67 111 110 116 101 110 116 45 76 101 110 103 116␣
 ↪104 58 32 50 13 10 65 99 99 101 115 115 45 67 111 110 116 114 111 108 45 65␣
 ↪108 108 111 119 45 79 114 105 103 105 110 58 32 42 13 10 13 10 79 75"]
})

predictions = predict(input_data)

output_data = input_data[["timestamp", "source_ip", "destination_ip",␣
 ↪"protocol", "length"]].copy()
output_data["http_scan"] = predictions[:, 0]
output_data["ping_scan"] = predictions[:, 1]
output_data["nmap_scan"] = predictions[:, 2]
```

```
output_data["unmalicious"] = predictions[:, 3]
output_data["maybemalicious"] = predictions[:, 4]
output_data["malicious"] = predictions[:, 5]

print(output_data.to_string(index=False))
```

```
/home/jupyter/venv/lib/python3.11/site-
packages/keras/src/layers/core/dense.py:86: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models,
prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)

Epoch 1/10

2024-04-10 22:48:44.971394: W
external/local_tsl/tsl/framework/cpu_allocator_impl.cc:83] Allocation of
1416276000 exceeds 10% of free system memory.

3689/3689             9s 2ms/step -
accuracy: 0.2772 - loss: 0.3673 - val_accuracy: 0.0000e+00 - val_loss: 0.2242
Epoch 2/10
3689/3689             9s 2ms/step -
accuracy: 0.2344 - loss: 0.2206 - val_accuracy: 0.4518 - val_loss: 0.2258
Epoch 3/10
3689/3689             9s 2ms/step -
accuracy: 0.2277 - loss: 0.2208 - val_accuracy: 0.4518 - val_loss: 0.2266
Epoch 4/10
3689/3689             9s 2ms/step -
accuracy: 0.2287 - loss: 0.2209 - val_accuracy: 0.4518 - val_loss: 0.2297
Epoch 5/10
3689/3689             9s 2ms/step -
accuracy: 0.2253 - loss: 0.2203 - val_accuracy: 0.4653 - val_loss: 0.2344
Epoch 6/10
3689/3689             9s 2ms/step -
accuracy: 0.2322 - loss: 0.2202 - val_accuracy: 0.0068 - val_loss: 0.2377
Epoch 7/10
3689/3689             9s 2ms/step -
accuracy: 0.2154 - loss: 0.2203 - val_accuracy: 0.0068 - val_loss: 0.2402
Epoch 8/10
3689/3689             9s 2ms/step -
accuracy: 0.2103 - loss: 0.2204 - val_accuracy: 0.0135 - val_loss: 0.2421
Epoch 9/10
3689/3689             9s 2ms/step -
accuracy: 0.2127 - loss: 0.2202 - val_accuracy: 0.0067 - val_loss: 0.2436
Epoch 10/10
3689/3689             9s 2ms/step -
accuracy: 0.2244 - loss: 0.2205 - val_accuracy: 0.0068 - val_loss: 0.2459

WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or
`keras.saving.save_model(model)`. This file format is considered legacy. We
```

```
recommend using instead the native Keras format, e.g.
`model.save('my_model.keras')` or `keras.saving.save_model(model,
'my_model.keras')`.

1/1             0s 31ms/step
                timestamp     source_ip destination_ip protocol  length
http_scan  ping_scan    nmap_scan  unmalicious  maybemalicious  malicious
2024-04-09T04:26:01+10:00 192.168.0.222  192.168.0.148      TCP     206
0.0        0.0 1.716450e-34          0.0             0.0        0.0
```

[14]:
```
tf.saved_model.save(model, "network_model_savedmodel")
```

```
INFO:tensorflow:Assets written to: network_model_savedmodel/assets

INFO:tensorflow:Assets written to: network_model_savedmodel/assets
```

[ ]:
```

```