ENDG_510_Lab01_2023-10-27

Cyber-Physical System Engineering

Member 1: John Santos (30087188)
Member 2: Kushal Gadhiya (30094278)
Member 3: Ben Byrne (30113846)

Date: 2023-11-10

Contents

Introduction

The lab session involved two tasks which provides an overview of the key components required for cyber physical systems to capture, process, and transfer the data across networks such that the data can be utilized to generate machine learning models which require machines that can handle stronger processes for training.

The first task was based on temperature and humidity data acquisition from the DHT11 sensor connected to a Raspberry Pi, and transferring the data wireless through LAN to an edge server running on a PC. This task involved many steps that had to be divided equally amongst the group members. Kushal was responsible for the hardware assembly by setting up the connections of the Raspberry Pi and the DHT11 sensors as well as network connections using Hotspot, along with writing the python script for adding noise to the acquired data, and model training using different machine learning algorithms. John was responsible for downloading the Raspbian OS 64-bit image using the Raspberry Pi imager tool and managed the software in a GitHub repository and suggested to use classification algorithms to train the model. Lastly, Ben was involved in formatting the Lab report based on the guidelines and assisted Kushal in adding noise to the data by changing the temperature around the sensor by covering the sensor with his hands and breathing on the sensor. Lastly, everyone in the group also worked on answering the questions provided in each section.

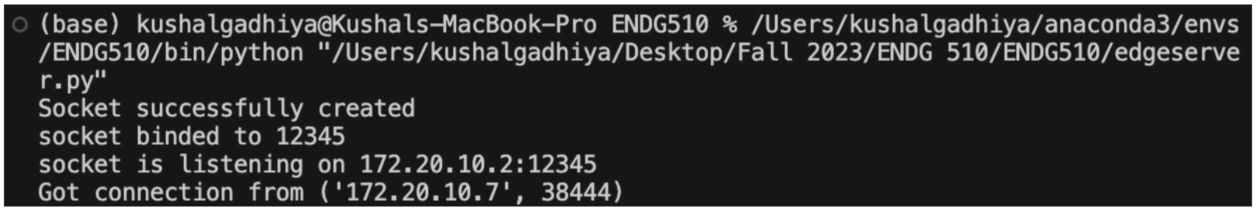
The second task was based on the analysis of the collected IoT data by applying machine learning algorithms to train a model for classifying temperature and humidity data as either valid or invalid. This task involved setting up the Anaconda distribution in our machines following the guidelines provided. Then a Jupyter notebook was created to compose the steps needed to train the supervised classification model such as loading, cleaning, and splitting the dataset into training and test subsets. The final steps were feeding the classification algorithms with the training split to generate the model and using the test split to evaluate the model performance. These are the algorithms used for this task and the reasonings behind their considerations:

- *Gaussian Process Classifier*: Is an algorithm that uses probabilistic prediction to classify labels and assigns a distribution curve over all the labels for the test data point. This algorithm is valuable for seeing the model's performance in detecting false positives and false negatives for crucial tasks in cyber-physical systems.
- *Stochastic Gradient Descent*: Is an algorithm that provides easier model implementation, effectiveness, and efficiency for utilizing larger data sets.
- *Ridge Classification*: Is an algorithm with the ability to treat classification tasks as regression by using linear least square with L2 regularization in the loss function, preventing overfitting and biased predictions.
- *Support Vector Machine Classification*: Is an algorithm with the default radial basis function (RBF) kernel which is particularly effective in classifying clusters with multiple non-linear decision boundaries.

Experiments

Task One

The experiment was performed by following the guidelines provided which included assembling the hardware connections for the sensor and the Raspberry Pi, downloading the Raspbian OS, setting up the network connection between the Raspberry Pi and the edge server on the PC, and providing software to capture data from the sensor and transfer it through the network from the edge device to the server via sockets.



```
○ (base) kushalgadhiya@Kushals-MacBook-Pro ENDG510 % /Users/kushalgadhiya/anaconda3/envs
  /ENDG510/bin/python "/Users/kushalgadhiya/Desktop/Fall 2023/ENDG 510/ENDG510/edgeserve
  r.py"
  Socket successfully created
  socket binded to 12345
  socket is listening on 172.20.10.2:12345
  Got connection from ('172.20.10.7', 38444)
```

Figure 1: Terminal output of edge server(laptop) listening for data from the client (Raspberry Pi)
Note: We forgot to take screenshot of data transfer during lab.

Initially there was a delay when first booting up the Raspberry Pi, the OS U.I. did not display when first powered on but kept looping at boot. This was eventually resolved by ensuring wiring connections, especially the power cable, were secured and that there were no other interferences. Additionally, we also encountered network connectivity issues using the University of Calgary's internal AirUC-Secure network. The issues were based on firewalls in the network which prevented the Raspberry Pi from connecting. This issue was eventually resolved by using a mobile hotspot from Kushal's mobile phone.

Teamwork allowed us to complete the experiment because everyone was involved, not one person did all the work, in this way the workload was spread out to allow the tasks to be completed on time.

Questions

1. **What is a Raspberry Pi? Why did you use it? Which version did you use?**

   A Raspberry Pi is an edge device with a microprocessor. A Raspberry Pi is used because it provides GPIO pins to directly connect IoT devices such as DHT sensors. It also provides connection to public networks allowing the transfer of data from the Raspberry Pi to other edge device using socket programming. The Raspberry Pi 4 is used in this experiment.

2. **What is socket programming, a server, and a client?**

   Socket programming is a type of wireless connection between two or more edge devices that allows a transfer of data. A socket establishes the connection between two devices connected under the same network and discoverable via the IP address and the designated port as the endpoint. A server establishes the connection and the port for which the edge devices can connect to that either sends or receives data. A client discovers the server by the server's IP address and the designated port to either send or receive the data.

3. **Which protocol did it use in socket programming? What are the benefits of using that protocol, and what are other protocols that can also be used?**

Transmission Control Protocol (TCP) was used in the socket programming. This protocol is more secure as it involves handshaking with the server to establish connection and to ensure the sent packets are received in order. An alternate protocol that could be used is User Datagram Protocol, but it is less reliable than TCP since packets may be lost or arrive out-of-order and does not involve handshaking.

4. **Which sensor did you use for the experiments, and what does it do? What are GPIO pins? What is "ground" in GPIO pins?**

The sensor that was used in the experiment was a DHT11 sensor which measures the temperature and humidity in the immediate environment. GPIO pins are access points to the Raspberry PI aside from the established ports that can either act as inputs or outputs. In the case of the experiment, they are configured as inputs and outputs which provide the sensor with the input voltage and the Raspberry Pi with the temperature and the humidity data that was measured by the sensor. A ground pin provides a reference voltage of 0V and assigned to pin 5 as an example for the Raspberry Pi 4.

5. **What do "encode" and "decode" mean in the provided code? Why are they used here?**

To “encode” messages means to translate them to a proper format such that they can be sent properly over the defined medium; in our case it was a wireless medium using a TCP protocol. Communication and message transfers over a medium are subject to noise which can interfere with the signal being transmitted. Encoding ensures that the effects of noise are minimized and that the signal is in the proper bandwidth such that they can be decoded and transmitted properly. To “decode” messages is to convert the signal either back to the original format or to another format which can be processed or understood by the user or the client/server.

6. **Apart from socket programming, what are other ways to transmit data?**

There are also wired communication protocols such as serial and UART which provide connections between devices and transfer of data using physical wires.

7. **Why is 'Label' added in the data? What do 'valid' and 'invalid' mean? Which value represents 'valid' and 'invalid'?**

A label is added because to train a model or to interpret the data, we need to differentiate between valid data versus poor data or noise. Valid data means that it can be used for training or provides an accurate representation of the environment. Invalid data are the noise or the corruption of data which does not conform to the temperatures and humidity values captured by the sensor. Valid data is represented as a 1 and invalid data is represented as a 0.

8. **What is an edge server and what are advantages of using an edge server?**

An edge server is a type of device which provides an entry point to a network along with computing resources for data processing. The advantage of using an edge server is that the processing of the data happens near the edge device reducing the propagation delay through the channel which is beneficial for real-time critical CPS application.

Task Two

The experiment was performed by following the guidelines provided to set up Anaconda in our machines and to train the model using a Jupyter notebook which involved many steps such as loading the data, cleaning the data by removing invalid, repeated, and missing values, splitting the data into train and test subsets, and finally training the model based on the training split and evaluating the model based on the test split.

The difficulty experienced in this task was the decision to either use classification or regression algorithms. There was a minor disagreement between the choices because the data was continuous, thus leading to the belief that regression algorithms should be used to predict the next trend in the values. On the other hand, the y-values were distinguished between valid or invalid attributes, thus classification algorithms can be used to allow the model to predict subsequent data as either valid or invalid. This dispute was eventually resolved by examining the steps provided which shows classification algorithms being used. Furthermore, regression algorithms produced continuous values which could not be analyzed nor compared to the y-values of the test split which comprised of binary valid or invalid labels.

Another difficulty was that initially, the algorithms produced models with perfect performance in accuracy, precision, and recall. It was later observed that the data used to train the model did not have enough noise and there wasn't a clear distinction between valid and invalid data because only 20% of the data was invalid data which caused the model to overfit on the valid data and that the noise data did not have enough variations for the model to consider many cases of noisy data. This issue was resolved by reducing the training split by 10% and adding 20% more noise with more variations following the conditions below.

$$[1]\ -50℃ < temperature_{noise} < 15℃,\ 40℃ < temperature_{noise} < 100℃$$

$$[2]\ -100\% < humidity_{noise} < 0\%,\ 200\% < humidity_{noise} < 500\%$$

Questions

1. **What is Anaconda? What benefits does Anaconda provide?**

Anaconda is a distribution of Python and R languages primarily used in data science, data processing, or machine learning applications. As part of the benefits, Anaconda provides a python package management tool that allows seamless integration of different packages in the application by ensuring that the appropriate, compatible versions are installed. For example, certain TensorFlow versions are compatible with certain NumPy versions, so Anaconda installs the specific versions to allow both libraries to be compatible. There are more functionalities such as `conda build` which allows users to build and share custom packages to the Anaconda cloud, PyPI, or other repositories.

2. What is a missing value, null, or duplicate? Why do we need to clean them?

Missing values are unpaired values such as feature values without a label or a label without a feature value. These values need to be cleaned or removed because they don't have sufficient information to aid in the model's training. A null value is not a number. These values should be cleaned or removed because they are not useful for training a model due to being inconsistent with the rest of the provided data. A duplicate is a value that already exists. These values should be cleaned or removed because if the labels are different between duplicates, this can potentially confuse the model and alter training results. Or, if the labels are the same, this could potentially cause overfitting in the model.

3. Why do we need to split our data into training and testing sets?

Data needs to be split into training and testing sets because training a model also pertains to the testing the models after training. We cannot use the same data used for training because the model is biased to the training data. To test that the model can make generally accurate predictions, it needs to be tested on data it has not already seen; this is done by splitting the data into training and testing sets.

4. Why do we need scaling? What are the existing scaling techniques? What are the drawbacks of the min-max scaling technique?

Scaling will improve the algorithm performance and prevent a single feature from dominating in calculation sometime in the future.

Existing scaling techniques:

- *Standardization*: Converts the mean and standard deviation of the feature value to be 0 and 1, respectively.
- *Normalization (Min-Max Scaling)*: Ensures data features is between a definite range, typically the max value of 1 and min value of 0.

Drawbacks of the min-max scaling technique:

- It is sensitive to the outliers in the data.
- Unable to preserve the relationship between data points, and in case of skewed distribution, alteration of the original data is highly possible.
- Less effective if the range of data is anything different than 0 to 1 for analysis done by some algorithms.

5. Which machine-learning model did you choose? Why?

We chose supervised machine-learning models and classification models in particular, as the primary goal was to distinguish between two sets of classes: noise and actual values (temperature and humidity). The output is also explicitly defined as the label.

6. **How did you evaluate your machine-learning model? Present your results in figure or table.**

We split the data into 70% training and 30% testing and used the accuracy and precision metrics provided by the scikit learn library to evaluate the model performance.

The metrics are calculated following the definitions for true positives, false positives, and false negatives.
[1] True positive: A correct prediction {prediction: 0, label: 0} or {prediction: 1, label: 1}
[2] False positive: Incorrectly predicted invalid data: {prediction: 1, label: 0}
[3] False negative: Incorrectly predicted valid data (missed): {prediction: 0, label: 1}

Accuracy was used to measure the overall performance of the model to see how often the model was able to correctly classify noise and the actual values.

$$Accuracy = \frac{true\ positives}{true\ positives + false\ positives + false\ negatives}$$

Precision was used to measure how effective the model is in making the correct predictions. In other words, in all the model's predictions, how many were correct?

$$Precision = \frac{true\ positives}{true\ positives + false\ positives}$$

Recall was used to measure how effective the model is in finding valid data. In other words, in all the ground truth test split, how many was correctly identified by the model?

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives}$$

See *Table 1.0 Classification Algorithms Performance Evaluation* under the section *Performance Evaluation* for the comparative analysis of the metrics used to evaluate the machine learning model.

Performance Evaluation

The following table shows the performance of the classification model trained on the same dataset but using four distinct algorithms. The classification algorithms that were evaluated are as follows: *Gaussian Process Classifier, Stochastic Gradient Descent, Ridge Classification, and Support Vector Machine Classification*.
Note: Recall was not used to measure the model's performance because a score of 1.0 was generated for all the algorithms used which is not useful for comparing the algorithm that produced the best results.

Algorithm	Accuracy	Precision
Gaussian Process Classifier	0.9904988123515439	0.9875
Stochastic Gradient Descent	0.7505938242280285	0.7505938242280285
Ridge Classification	0.8669833729216152	0.8494623655913979
Support Vector Machine	0.997624703087886	0.9968454258675079

Table 1.0 Classification Algorithmes Performance Evaluation

The results show that the **Support Vector Machine Classification algorithm** with RBF kernel produces superior results (i.e. higher accuracy and precision) compared to the rest of the algorithms.

Code & Dataset

E. (n.d.). *GitHub - ENELEngineering/ENDG510 at 510_LabOne*. GitHub.
https://github.com/ENELEngineering/ENDG510/tree/510_LabOne

Conclusion

This experiment demonstrates the key concepts and components in how cyber physical systems collect, process, and transfer data from one device to another over the network such that the data can be utilized to train machine learning models to perform the desired tasks which is only possible in machines that can handle higher processes. The first task of the experiment involved setting up the Raspberry Pi and DHT11 sensor hardware, the network, and the software to collect the temperature and humidity data to be transferred over the network to the server hosted by our PC. The second task involved utilizing the collected data to train a supervised classification machine learning model based on the *Support Vector Machine Classification* algorithm which generated the model with the best performance using the accuracy and the precision metric.