**Parameter settings reliability test of a Sensor System for Person Detection in an Office Environment**

Master of Engineering

Information Technology

Machine Learning

Ravikumar Solanki Dipanjan Saha Aishin Abdulla Yoosufali

1324549 1324523 1359022

[ravikumar.solanki@stud.fra-uas.de](mailto:ravikumar.solanki@stud.fra-uas.de) [dipanjan.saha@stud.fra-uas.de](mailto:dipanjan.saha@stud.fra-uas.de) [aishin.abdullayoosufali@stud.fra-uas.de](file:///C:\Users\Ravikumar\Downloads\aishin.abdullayoosufali@stud.fra-uas.de)

***Abstract*—In today’s era, an effective solution for a person detection and recognition in an unsustainable environment is somewhat easy. Nevertheless, there are many algorithms which cannot fulfill the condition for better accuracy. Due to the difficulty and complexity of these assessments, complex machine learning algorithms for this specific application have been established. In our research, we conducted an experiment to detect a person in an office environment, we choose a supervised learning algorithm, called Random Forest algorithm to train our ML model. Simultaneously, in our analysis under the guidance we considered three conditions such as person sitting wearing summer and winter clothes and with an empty chair of objects present in an office space to detect the accuracy and F1 score by creating confusion matrix for all the variations.**

***Keywords—Machine Learning, Supervised Learning, Random Forest, Confusion Matrix***

# Introduction

# Our research is about predicting and testing the reliability of a sensor system for human detection in an automobile while wearing cold clothing. The goal is to identify and enhance the output of a sensor that recognizes if a person is seated in the Office environment, particularly when wearing a certain set of summer and winter clothing such as shawls, jackets, and cap. To record our data, we utilize a sensor and the notion of Fourier transforms. We utilize these measurement datasets to generate numerous confusion matrices for our research once we have adjusted the output readings during the project and got the necessary sets of values and observations. we use these measurement datasets to create several confusion matrices for our analysis and then use the values obtained in these matrices to further train an RF model to increase the reliability and the precision of the output of the sensor.

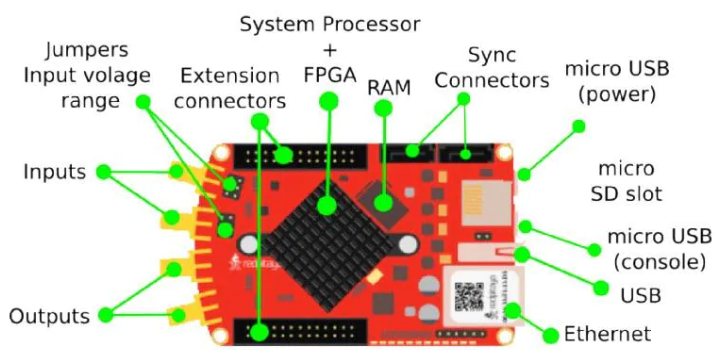
# Theoretical Background

## Red Pitaya

Red Pitaya device was used in our project as the sensor. It's a free and open-source gadget. This means that the sensor's functionality can be coded freely and independently based on our needs, and many previously created programs can be found on the Internet's open-source networks. It gives us a flexible measuring device that can be used with a smartphone, tablet, or PC and one or more probes to replace a variety of much more expensive separate measuring equipment. [1]

It comes in two different configurations. The Xilinx Zynq 7010 SoC, which is a combination of the dual-core ARM Cortex A9 processor and the 28nm Xilinx programmable logic (PL) [8], is included in both versions. Red Pitaya OS, a lightweight version of Linux, is the operating system. SD memory card is used for system storage. Both the processor and the FPGA can be configured using an operating system (OS). [2]

A GNU/Linux operating system is used to run the sensor. Individual programming is possible via several software interfaces. A user interface is provided as an HTML page via a web service for all measurement operations. They can be used to control and document measurements on a computer or mobile device. The core Red Pitaya unit has two analog RF inputs and outputs, as well as 16 general input and output ports. The board also includes a micro-USB connector for the console, a micro-SD card slot, an RJ45 socket for Ethernet, and a USB port. The Red Pitaya can receive as well as transmit radio frequency signals. 50 MHz is the frequency range. [3]



1. Red Pitaya [4]

## First Fourier Transform (FFT)

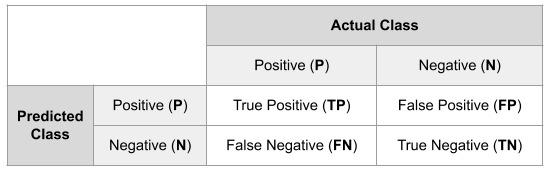
To provide us with output in the desired numerical discrete format, the Red Pitaya server employs the concept of Fast Fourier Transform. To gain a thorough understanding of the project deliverables, a general understanding of the Fast Fourier transform is required. It decomposes a signal into its spectral components and provides frequency information as a result. FFTs are used in defect analysis, quality control, and condition monitoring in machines and systems.

In precise terms, the FFT is the optimum algorithm for accomplishing the "Discrete Fourier Transformation" (DFT). Over time, a signal is sampled and split into frequency components. These components are composed of a single sinusoidal oscillation of varying frequency, within each amplitude and phase.

The diagram below depicts this transition. Throughout the monitored time, the signal has three distinct dominant frequencies. [5]

## Confusion Matrix

A confusion matrix is a common classification output. It specifies the correspondence between the predicted category and the sample's inherent category. For the in-class classification task, the confusion matrix is a n\*n matrix. The degree of polymerization within a category and the degree of dispersion between categories are both represented in the confusion matrix.



1. Confusion Matrix

The above figure consists of several parameters which are calculated based on four core parameters:

1. *True Positive (TP)*

When the expected value matches the actual value, we have a true positive case. This indicates that the model predicted a positive result, and the actual result is positive.

1. *True Negative (TN)*

When the expected value and the real value are the same, a truly negative situation arises. This time, however, the model predicted a negative value, and the actual value is also negative.

1. *False Positive (FP) - Type 1 error*

When the expected value matches the incorrect value, this is known as a false positive. The model anticipated a favorable outcome even though the actual value was negative. This is called the first type of prediction error.

1. *False Negative (FN) - Type 2 error*

When the expected value matches the incorrect value, this is referred to as a false negative case. Even though the actual number was positive, the model predicted a negative result. This is the second type of error in the prediction model, and it assists us in obtaining additional parameters. [6]

*Advantages of a confusion matrix*

* It evaluates the implementation of classification models when predicting the future on test data, allowing us to discover how effective our classification model is. To some extent, it can detect trends and expected values.
* It not only indicates the error type made by the classifiers, as well as the magnitude of the fault. It can, for example, assist us in determining whether the error is type-I or type-II.
* The confusion matrix can be used to manage the model's many properties, such as accuracy and precision. [7]

## Precision Vs Recall

The term 'precision' refers to how many of the cases that were correctly predicted turned out to be positive.

This is a critical metric for determining whether our model is correct.

'Recall,' on the other hand, displays how many of the actual positive cases the model anticipated properly.

(3)

When it comes to analyzing how our model predicts results, both parameters are crucial. Precision is a useful statistic when False Positives are more of a problem than False Negatives. The recall is a useful metric when False Negative outnumbers False Positive.

However, there will be times when determining whether Precision or Recall is more important will be impossible. We'll combine them if that's the case. [8]

## F1 Score

The F1 score is an important parameter that we used as a reference standard for our readings and observations. It is because when we try to improve our model's precision, the recall drops, and vice versa. So, to get the best results from our readings, we use the F1 score, which captures both trends in a single value.

The F1 score provides a composite picture of Precision and Recall because it is a harmonic mean of these two metrics. It reaches its pinnacle when Precision equals Recall. Although the F1 score is an important metric, it is not sufficient for all types of classification. This is due to the F1-score's low interpretability when viewed in isolation. We have no way of knowing whether our classifier is maximizing precision or recall, whether we need to adjust our reading parameters, etc. As a result, we combine its use with other evaluation indicators to obtain a more comprehensive and accurate picture of the outcome. [9]

## Rates

We develop more statistical terms based on the 4 core values to better understand the nature of the readings and observations obtained. The four parameters listed below, all of which are critical, are primarily used to accomplish this. These are called as "rates".

TPR and TNR should be high for better performance, while FNR and FPR should be low. TPR and TNR appear to share a lot of similarities. As the names suggest, we divide the TP and TN into actual positive and negative values. FPR and FNR are in a different set of circumstances. The FP is divided by actual negative values, and the FN is divided by actual positive values for FPR. [7]

## PPV and NPV

Positive Predictive Value (PPV) and Negative Predictive Value (NPV), when the ratio of real negative predictions when all negative predictions are considered, are two other metrics that evaluate the proportion of real positive predictions when all positive forecasts are taken into account. These two statistics account for the condition's prevalence, allowing for precise forecasting of the likelihood of a given outcome.

## Algorithm

Machine Learning provides computers or machines the ability to learn and make decisions using data without any explicit programming. For instance, the collection of Wikipedia entries is classified into different categories. There are three different types of Machine Learning such as

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

To analyze our experiment, we have considered supervised learning and will discuss the chosen algorithm, Random Forest, to train our model.

## Supervised Learning

In Machine learning, uses labeled data (such as data points or samples). Data samples can be either predicted variables or feature variables and a target variable will be there. For the supervised learning structure, a tabular form of data is fed to train the model, which represents the data points and the extracted features. The aim of supervised learning is to predict the target value using predicted variables. For our experiment, we have chosen a random forest algorithm to train the model with the generated dataset using Red pitaya for the person detection in both summer and winter wear in some applied conditions: Movement, No Movement, with Empty Chair.

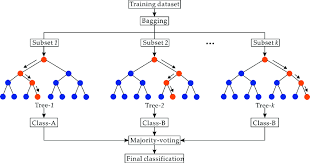
## Random Forest (RF) Algorithm

Random Forest (RF) algorithm constructs decision trees based on the data samples provided, receives predictions from each tree, and votes on the best option. It's also an excellent indicator of how important a feature is.

The Random Forest algorithm integrates numerous decision trees to create a forest of trees. The bigger the frequency of trees in the forest, the higher the accuracy of the random forest classifier. In comparison to other algorithms, the random algorithm requires less training time. It anticipates output with good accuracy, and it runs quickly even with a huge dataset. When a considerable amount of the data is missing, it can still maintain accuracy.

The first part of Random Forest is to generate the random forest by assembling N decision trees, and the second phase is to draw conclusions for each tree generated in the first phase. The following steps and diagram can be used to demonstrate the working process:

1. From the training set, choose K random data points.
2. Create the decision trees that correspond to the data points you've chosen (Subsets).
3. Select the size N for the number of decision trees you wish to create.
4. Repetition of Steps 1 and 2.
5. For new data points, locate each decision tree's forecasts and allocate the datasets to the category with the most votes.



## Fig 3. Random Forest ML Algorithm [10]

The RF algorithm can use both types such as classification and regression problems. The continuous value predicted by a classification algorithm is in the form of a possibility for class labels. Whereas a regression method can estimate a discrete value, perhaps as an integer quantity. Evaluation for classification and regression methods can vary according to some conditions:

* The accuracy of classification predictions can be assessed, but not regression predictions. Although, in the case of evaluation of regression predictions we can use root mean square, whereas classification predictions cannot use that function.
* Because it makes predictions using a huge number of decision trees, it is considered an extremely accurate and resilient model.
* Random forests take an average of all of the decision-tree forecasts, canceling out the biases. As a result, it is free of the problem of overfitting.
* The missing values can be handled by the random forest classifier. There are two approaches to dealing with missing values. Firstly, we can replace the continuous variables with the median values. Secondly, determine the proximity-weighted average of the missing variables.
* For feature selection, a random forest classifier might be utilized. It entails picking the most important characteristics from the training dataset's available features.

# Data Collection & Observation

For a period of more than two months, we generated a dataset using the hardware i.e., Red Pitaya and Ultrasonic sensor to collect the data samples (which produce FFT data) of detected person or object, along with setting up some conditions to observe the differences in all scenarios. We have implemented the methods based on constant analysis and feedback to optimize the solution from the trained model. We would like to discuss more the methodologies and improvements we performed for the optimization of our model and generated output in different phases which is discussed in detail. The data were collected under different circumstances. For instance, we took data in four phases:

* Empty Chair (with/without open Door, with/without open Window) to analyze the differences in all the four alternatives.
* Movement & No Movement measurement using Summer Wear variations. (With & without Threshold value)
* Movement & No Movement measurement using Winter Wear variations. (With Threshold values)

## Measurement Equipment & Methods

During the first phase we got familiar with the Red Pitaya sensor, how to collect samples using the UDP Client Program application to connect with the sensor, and then prepare the data for predicted and target detection.

## Data Collection using Summer Wear

In the second phase, we started by obtaining readings for different people in a set of combinations of clothing and recording their bodily characteristics. We took 3,000 events of 50,000 total measurements per person. We created an excel file using only predicted and target values and the other file using FFT data to train our model with and classification.

A total of 28 separate examples were recorded with three different people, each of whom was recorded with a different outfit each time (for summer and winter wear). The threshold value will be the default value here. In all circumstances, we put the positive event as the classification of a person and the negative event as the classification of an object. In Movement measurement, we continuously were walking under the beam of the sensor to be in the definite range. In non- Movement measurement, we were seated on the chair for a continuous period for the investigation. In both scenarios, we made some variations in the clothing as relatable as summer wear. The output from the sensor was stored to train our model. The result of this investigation is briefly stated in the next section. As a result, each of these cases was unique, as were the clothes or accessories they were wearing, as well as their physical characteristics. The following is an example of a use case:

## Data Collection using Summer Wear with Threshold Variations

At this point, we took the measurement again same as how we performed the last time, however in this phase adjusting the threshold values for the classification process. The threshold value can be changed using the formula (-t X Y). As the default value for the threshold for X is 140 and Y is 10,000. Whenever we restart the red pitaya, it will always be set to the default value as mentioned. In our case, we started to change the threshold values in duration, such as for X of 20 and for Y of 2000. In summer wear data collection, we performed in two different situations like Movement and No Movement. In Movement, a person was continuously moving around for 3000 events. Similarly, for No Movement, a person was kept seated continuously for at least 3000 events. This means that our value for X starts from 80 up to 200 while keeping the Y constant as 10000, similarly for Y from 1000 up to 13,000 while keeping the X constant as 80 as shown in the table below.

1. Threshold VARIATIONS

|  |  |
| --- | --- |
| X = 80 & Y= 10000 | X = 80 & Y = 1000 |
| X = 100 & Y= 10000 | X = 80 & Y = 3000 |
| X = 120 & Y= 10000 | X = 80 & Y = 5000 |
| X = 140 & Y= 10000 | X = 80 & Y = 7000 |
| X = 160 & Y= 10000 | X = 80 & Y = 9000 |
| X = 180 & Y= 10000 | X = 80 & Y = 11000 |
| X = 200 & Y= 10000 | X = 80 & Y = 13000 |

Furthermore, we did the variations with 3 different persons with different height, weight, and clothing style. With all these variations, we can train the model and optimize the output with higher precision as possible. A used case scenario for the variations in person 1 (i.e., Dipanjan Saha) is discussed below.

1. Person 1 Details

|  |  |
| --- | --- |
| **Person Details** | **Parameter** |
| Name | Dipanjan Saha |
| Height | 73kg |
| Weight | 5.4inches |
| Clothing Style | Summer Wear |



Fig. 4. Test case wearing summer wear with closed-door

1. F1 Scores for Threshold values with FFT (movement)

|  |  |
| --- | --- |
| **Parameters** | **F1 Score** |
| **X variates between 80 to 200, Y is constant at 10000** |  |
| 80-10000 | 0.68 |
| 100-10000 | 0.68 |
| 120-10000 | 0.65 |
| 140-10000 | 0.54 |
| 160-10000 | 0.53 |
| 180-10000 | 0.60 |
| 200-10000 | 0.82 |
| **X is constant at 80, Y variates between 1000 to 13000** |  |
|  |  |
| 80-1000 | 0.67 |
| 80-3000 | 0.61 |
| 80-5000 | 0.60 |
| 80-7000 | 0.55 |
| 80-9000 | 0.61 |
| 80-11000 | 0.65 |
| 80-13000 | 0.67 |

## Data Collection using Winter Wear with Threshold Variations

During the time for winter wear data collection, we considered the same parameter variations lie we did in the summer wear data collection. Here also, we kept the threshold variations for X and Y values. The only difference was in the style of clothing such as we wore jackets, gloves, and winter caps. In this stage, the same conditions were applied i.e., Movement and No Movement of a person. In a similar manner, all the data were recorded. Some details of the data samples of person 2 (i.e., Aishin Abdulla Yoosufali) wearing the winter clothing can be found below.

1. Person 2 Details

|  |  |
| --- | --- |
| **Person Details** | **Parameter** |
| Name | Aishin Abdulla Yoosufali |
| Height | 74kg |
| Weight | 5.7inches |
| Clothing Style | Winter Wear |



Fig 5. Test case wearing winter wear with closed-door

1. F1 Scores for Threshold values with FFT (no movement)

|  |  |
| --- | --- |
| **Parameters** | **F1 Score** |
| **X variates between 80 to 200, Y is constant at 10000** |  |
| 80-10000 | 0.65 |
| 100-10000 | 0.63 |
| 120-10000 | 0.62 |
| 140-10000 | 0.61 |
| 160-10000 | 0.54 |
| 180-10000 | 0.54 |
| 200-10000 | 0.64 |
| **X is constant at 80, Y variates between 1000 to 13000** |  |
|  |  |
| 80-1000 | 0.74 |
| 80-3000 | 0.67 |
| 80-5000 | 0.65 |
| 80-7000 | 0.73 |
| 80-9000 | 0.70 |
| 80-11000 | 0.68 |
| 80-13000 | 0.73 |

In all circumstances (Empty chair, Summer and Winter wear with Movement and No Movement) explained above were differentiated in the absence and presence of some noise. These samples were then used to train the RF model and generated the F1 score ad accuracy for all three persons. During our observations, we noticed a person with less height has a low F1 score, and the person with more height has a better F1 score and accuracy. All these measurements and details are explained in detail in the next section.

# Model Analysis

We began building the algorithmic code that was necessary to produce a confusion matrix for the analysis of the measurements and to accomplish the project's goal after we had taken roughly 55000 measurement readings over the course of the first months of the project.

The readings, as well as achieving the project's goal. The code that we created is divided into five sections.

1. Data labeling.
2. Split Data into Separate Training and Test set.
3. Random Forest Classifier Model.
4. Confusion Matrix.

We use the library from python for reading files, changing input files as per required, for ML algorithms, and plotting a chart. In below figure, describes all libraries.

Text

Description automatically generated

Fig 1. Imported libraries

## Data labeling

Data has typically been tagged by hand, which is a time-consuming and resource-intensive procedure. ML models or algorithms, on the other hand, may be used to auto-label data by first learning them on a fraction of manually labeled data.

In the section, we created a Target value column as per the use case and drop the unwanted column and labeled the data as per required.

Case-1: - without FFT file, in this, we only have three columns Dist., Predicated value, and Target Value.

Text

Description automatically generated with medium confidence

Fig 2. Code for Data labeling without FFT file.

Case-2: - for the FFT file in this we have more columns, so we removed unwanted columns and rename columns and add a new column. And declare the feature vector and target variable.

Text

Description automatically generated

Fig.- 3 Data labeling for FFT File.

## Split Data into Separate Training and Test Set.

When machine learning algorithms are used to generate predictions on data that was not used to train the model, the train-test split process is used to measure their performance.

The size of the train and test sets is the procedure's key configurable parameter. For either the train or test datasets, this is usually given as a percentage between 0 and 1. For example, a training set with a size of 0.67 (67%) is given to the test set with the remaining percentage of 0.33 (33%).

Graphical user interface, text, application

Description automatically generated

Fig: - 4 Code for Split data.

## Random Forest Classifier Model.

A random forest is a meta estimation that employs averaging to increase predicted accuracy and control over-fitting by fitting a variety of decision tree classifiers on the different sub of the dataset. If bootstrap=True (default), the sub-sample size is regulated by the max sample’s argument; otherwise, the whole dataset is utilized to create each tree.

Graphical user interface, text, application, email

Description automatically generated

Fig. Case-1: - Random Forest Classifier model with default parameters

We’ve created a Random Forest Classifier model with n estimators = 10 as the default value. As a result, I built the model using ten decision trees. Now I'll try increasing the number of decision trees to see how it affects accuracy.

And in Case-2 With 10 decision trees, the model accuracy score is 0.6847, whereas, with 100 decision trees, it is 0.7057. As a result, as the number of decision trees in the model grows, so does the predicted accuracy.

Case-2: - Random Forest Classifier model with parameter n\_estimators=100

Graphical user interface, text, application, email

Description automatically generated

**So, we use 100 decision-tree in all other experiments.**

## Confusion Matrix.

A confusion matrix is a technique for summarizing a classification algorithm's performance. A confusion matrix will provide us with a clear image of the classification model's performance as well as the sorts of mistakes it generates. It provides a breakdown of right and wrong predictions for each category.

Text

Description automatically generated

Fig 5. Code for Confusion Matrix.

To plot the Confusion Matrix in good view we use the Seaborn library.

Text

Description automatically generated

Fig. 6. Code for plotting Confusion Matrix.

Another technique to assess the classification model's performance is to generate a classification report. It shows the model's precision, recall, f1, and support scores.

Text

Description automatically generated with medium confidence

Fig 7. Code for Classification Report.

# Results and Trends

As previously stated, we used a well-defined, statistically supported, and regularly updated method to record, evaluate, and extract findings from the readings during the length of the project.

## The First Phase.

We were able to establish which range of physical features and combination of winter clothing produced the intended or expected outcomes in phase 1 when we were obtaining readings from different persons in the early stages when we were taking readings from different individuals.

A person sitting at a desk

Description automatically generated with low confidence

Fig 8. Use-case Summer Wear non-movement with Cap.

A picture containing indoor, wall, floor, person

Description automatically generated

Fig 9. Use-Case Summer Wear Movement with Cap.

A picture containing text, desk, floor, indoor

Description automatically generated

Fig 10. Use-case Empty Chair.

## The Second Phase

Following the learnings from the first phase, we used 76 different combinations of Empty Chair with the door close, Empty Chair with the door open, Empty Chair with Window open, Empty Chair with Window and Door open, Movement and no-Movement for Summer Wear, Door open, Movement and no-Movement for Winter Wear, Person with Summer Jacket, Person with Winter Jacket, Person with Winter Cap for each person in the 2nd phase of the experiment.

The confusion matrix recorded in the 2nd phase is as follows:

Chart, treemap chart

Description automatically generated

Fig 11. Confusion matrix without FFT file in Phase 2

## The Third Phase

Following the learnings of the 2nd Phase, we observed all readings with respect to possible parameters followed previously and identified the threshold value representing the best possible observations at the value of “-t 180 10000”. Therefore, we have taken a total of 30000 readings at this value for different use cases at "-t 180 10000". Cases used are

Empty Chair with the door close, Empty Chair with the door open, Empty Chair with Window open, Empty Chair with Window and Door open, Movement and no-Movement for Summer Wear, Door open, Movement and no-Movement for Winter Wear, Person with Summer Jacket, Person with Winter Jacket, Person with Winter Cap for each person in the 2nd phase of the experiment.

We then made a small variation in the X and Y parameters of the threshold to verify our initial threshold values. We thus recorded the new readings at these values for **Summer Wear.**

For No Movement

1. "-t 80 7000 "   No of readings: 5000
2. "-t 85 90000"   No of readings: 5000
3. "-t 80 10000"   No of readings: 5000
4. "-t 100 10000"   No of readings: 5000

For Movement

1. "-t 80 7000 "   No of readings: 5000
2. "-t 85 90000"   No of readings: 5000
3. "-t 80 10000"   No of readings: 5000
4. "-t 100 10000"   No of readings: 5000

Chart, waterfall chart, treemap chart

Description automatically generated

Fig. 12. Summer Wear No Movement Confusion matrix in 80-7000

Chart, treemap chart

Description automatically generated

Fig. 13. Summer Wear No Movement Confusion matrix in 80-9000

Chart, treemap chart

Description automatically generated

Fig. 14. Summer Wear No Movement Confusion matrix in 80-10000

Chart, treemap chart

Description automatically generated

Fig. 15. Summer Wear No Movement Confusion matrix in 100-10000

## The Four-Phase Readings and F1 Score analysis

For analysis F1 score with divide the use-case in the FFT file and without the FFT file and find out the confusion matrix, and then divide into Movement and no movement mode for each person. We did this for Summer Wear and Winter Wear with different variations.

In each case, there is an input file with and without FFT file, in without FFT file we calculated without any classifier for the confusion matrix, and with FFT File we use Random Forest with 100 decision tree ML algorithm.

And then we created three different sections.

1. Empty chair – in this section we use it for different cases. For with FFT and without FFT file.
2. Empty chair with the door closed.
3. Empty chair with the door open.
4. Empty chair with Window and door open.
5. Empty chair with the window open.

Chart, bar chart

Description automatically generated

Fig. 16 F1 score for Empty Chair

We notice from the figure that if we open the window and door together then we are getting a bad F1 score, because of noise.

1. Summer Wear in this we created 14 combinations with respect to x and y values. For both with FFT and without FFT for movement and no movement. For variation, we use “Cap” and “Glasses”
2. Movement with FFT (14 Combinations)
3. Movement without FFT (14 Combinations)
4. No Movement with FFT (14 Combinations)
5. No Movement without FFT (14 Combinations)

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

1. Winter Wear in this we created 14 combinations with respect to x and y values. For both with FFT and without FFT for movement and no movement.

For variation we use “Winter Cap” and “Winter hand cloves”

1. Movement with FFT (14 Combinations)
2. Movement without FFT (14 Combinations)
3. No Movement with FFT (14 Combinations)
4. No Movement without FFT (14 Combinations)

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

We used the FFT to be collected dataset from the Red Pitaya as an input for our final acquired threshold of "-t 80 10000" in this project:

Table

Description automatically generated

Fig. 17. Dataset of Movement Seat case.

Table

Description automatically generated

Fig. 18. Dataset of Empty Chair case.

# Conclusion

Our demonstration presents an integrated framework for person detection in an office environment, learning and recognizing with a set of circumstances. Our analysis of detection varied with summer and winter wear. Random forests are generally both type homogenously and classical decision tree, which generates the random decision criteria. We implemented our framework to compare the performance of a person or object detection. In this experiment, we built two models, one with 10 decision trees and the other one with 100 decision trees. The observation for the F1 score of the model. Besides that, we noticed the variations with Empty chair, where the window and door were kept closed, then the F1 score was very low due to the sound effects. In both summer and winter wear results, a different set of confusion matrices were developed including the FFT and non-FFT file information. The observation from summer wear (no movement) shows the F1 score as 0.97, in comparison with winter wear, where the F1 score shows 0.89, where the person is sitting under the sensor for a continuous period. Similarly, the summer wear for movement shows the F1 score of 0.93, and the winter wear shows 0.84. To conclude we can say that our RF model that we used for a person detection in an office space environment, predicted very well for a person wearing summer wear rather than a person wearing winter wear.

# Acknowledgement

We would like to take this opportunity to express our gratitude to our professor, Dr. Andreas Pech, for their unwavering support, guidance, and assistance in allowing us to work on this project. It would not have been possible without the generous support and assistance of our mentor, Prof Umansky, who was always willing to offer advice on the issues we encountered throughout the project.

# References

|  |  |
| --- | --- |
| [1] | M. Richards, Pi Updates and Red Pitaya, Radio User. Bournemouth, UK: PW Publishing Ltd. 11 , 2016. |
| [2] | P. M. a. T. H. P. Dejdar, High-speed Data Acquisition and Signal Processing Using Cost Effective ARM + FPGA Processors, 42nd International Conference on Telecommunications and Signal Processing (TSP), 2019, pp. 593-596. |
| [3] | Red Pitaya, "<https://redpitaya.com/>" |
| [4] | Red Pitaya,  "https://www.mouser.in/new/red-pitaya/red-pitaya-starter-kit/", 2015 |
| [5] | C. Loan, Computational Frameworks for the Fast Fourier Transform, 1992. |
| [6] | K. M. Y Ting, Sammut, Claude; Webb, Geoffrey I. (eds.). Encyclopedia of machine learning., 2011. |
| [7] | D. M. W. Powers, Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation, Journal of Machine Learning Technologies. 2 (1):, 2011, pp. 37-63. |
| [8] | JavaPoint, " <https://www.javatpoint.com/confusion-matrix-in-machine-learning>" |
| [9] | Y. Sasaki, The truth of the F-measure, 2007. |
| [10] | Ernest Yeboah Boateng, Joseph Otoo, Daniel "Abaye, Basic Tenets of Classification Algorithms K-Nearest-Neighbor, Support Vector Machine, Random Forest and Neural Network: A Review", Journal of Data Analysis and Information Processing, 2020 |