Automotive Sensor for Object Recognition using Red Pitaya and Raspberry Pi

Masters of IT  
Sehrish Fahim Limon  
[sfahim@stud.fra-uas.de](mailto:sfahim@stud.fra-uas.de) limon@stud.fra-uas.de

*Abstract*— **Objective is to design software for receiving, displaying and saving the acquired ultrasonic signals from Red Pitaya with Raspberry Pi. The objective is to make the system stable. First is to acquire the data from the ultrasonic sensor with the range of 1 meter. taking the data with least noise ratio by capturing the time when object data is available using external trigger condition.**

**When we have desired object before working on feature part we have cut down the input part to make it more stable and accurate data for each object.**

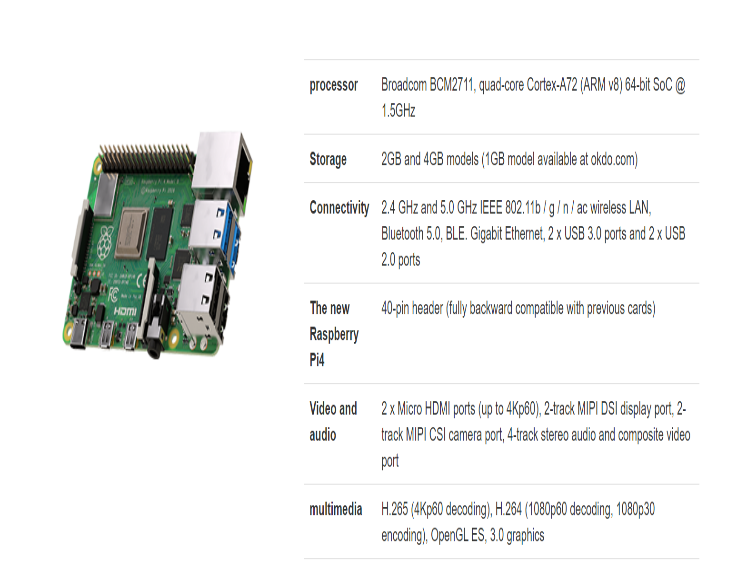
Keywords—component, formatting, style, styling, insert (key words)

Introduction (Red Pitaya)

The Raspberry Pi is a low cost computer that plugs into a computer monitor or TV, and uses a standard keyboard and mouse. We can use it to design own software and we can use program in languages like Scratch and Python. It’s capable of doing everything you’d expect a desktop computer to do, from browsing the internet and coding for software.

The reason to use this is to able to connect it with Red pitaya to show how one device which is connected to the sensor can send data and we can use raspberry as a receiver for that information and code to make it intelligent to recognize different objects precisely.

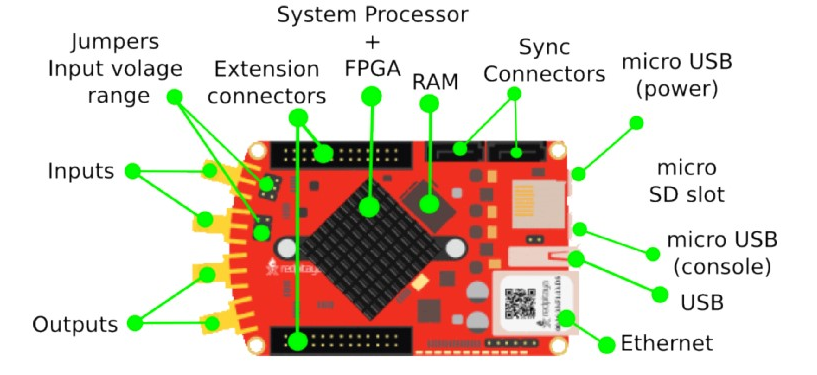
Like a normal computer raspberry pi contains processor, storage,connectivity, video audio support, multimedia



Introduction (Raspberry Pi)

Red Pitaya is an open-source hardware made to alternate for many expensive laboratory measurement and control instruments.

Red Pitaya offers open-source-software measurement and control tools that consists of easy-to-use visual programming software and free of charge, ready-to-use open-source, web-based test and measurement instruments running on powerful, credit card-sized boards. With a single click, the board can transform into a web-based oscilloscope, spectrum analyzer, signal generator, LCR meter, Bode analyzer, or one of many other applications. Red Pitaya can be controlled by using Matlab, LabVIEW, Python & Scilab.



*Key Features*

* High performance hardware
* Replaces most essential instruments like Oscilloscope, Spectrum analyzer, Signal generator, LCR meter
* LAN or wireless access from any WEB browser via tablet or a PC regardless of the OS (MAC, Linux, Windows, Android, iOS) LabVIEW and MATLAB® interface.
* Possibility to make your own application and share it with others
* Open source software

*Software*

Red Pitaya is based on GNU/Linux operating system and can be customized at different programming levels. Available software interfaces include: HDL, C/C++, scripting languages, MATLAB and HTML based web interfaces.Red Pitaya software is open source and can be downloaded from GitHub. All development tools are free.

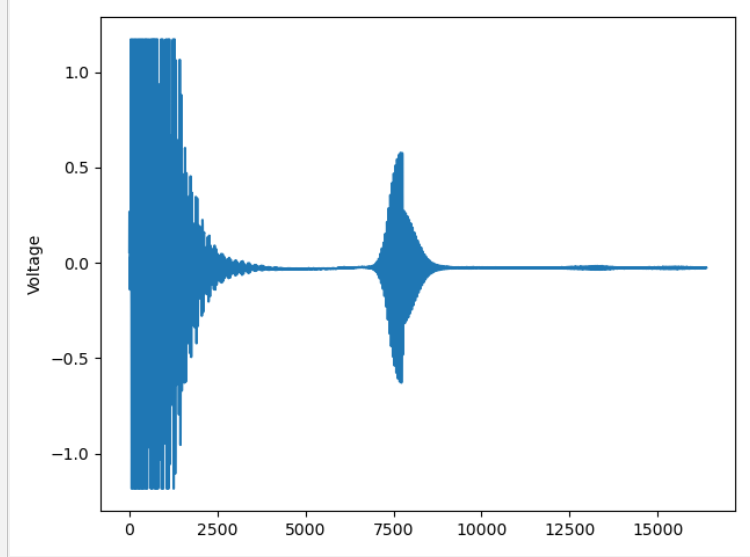
# Overview

## Data acquizition

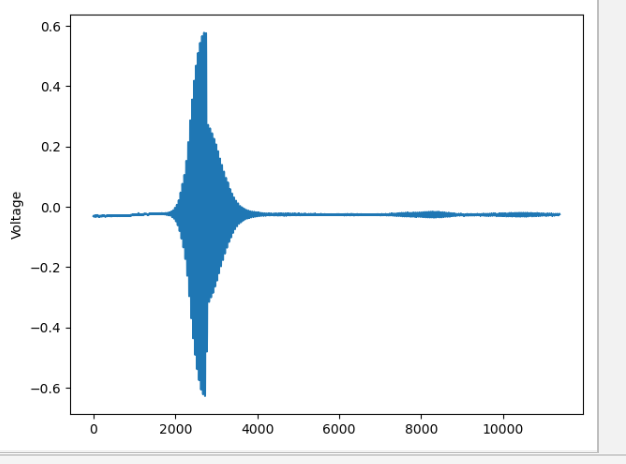
When we have desired object before working on feature part we have cut down the input part to make it more stable and accurate data for each object.

then after analyzing the data, develop feature extraction and classification. We have chosen Bayes Classifier for Machine learning part and trained our model. We have used frequency domain for this purpose.

Signal input with object reflection:



After elimination of input signal:



## Feature Extracion

In order to do this which give us more precise look to object to extract features.

We have applied Fast Fourier transform fft in order to have frequency domain for captured signal,

After that we have done power spectrum in order to have

A Power Spectral Density (PSD) is the measure of signal's power content versus frequency. A PSD is typically used to characterize broadband random signals. It can be looked upon as a frequency-domain plot of power per unit Hz vs. frequency.

Therefore, while the power spectrum calculates the area under the signal plot using the discrete Fourier Transform, the power spectrum density assigns unit of frequency and thus, enhances periodicities. By doing this it will give us more detailed signal to extract features. By this data we have taken peaks, shape, mean and variance as features to proceed with chosen Bayes Classification.

# Machine Learning

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience.

## Selection of Naïve Bayes as Machine Learning

Naive Bayes comes under the class of generative models for classification. It models the posterior probability from the class conditional densities. So the output is a probability of belonging to a class.

SVM on the other hand is based on a discriminant function given by y = w.x+b. Here the weights w and bias parameter b are estimated from the training data. It tries to find a hyperplane that maximizes the margin and there is optimization function in this regard. Performance wise SVMs using the radial basis function kernel are more likely to perform better as they can handle non-linearity’s in the data. Naive Bayes performs best when the features are independent of each other which often does not happen in real. Having said that it still performs good even when the features are not independent. We have chosen naïve Naive Bayes as our features are independent of each other and we have three objects to identify (class) Class 1 Wall, Class 2 Human, Class 3 Car. By using Naive Bayes, we will train and create these three type of classes and train our model.

## How Naïve Bayes Work(Implementation)

We can describe this in 5 part:

Step 1: Separate by Class.

Step 2: Summarize Dataset.

Step 3: Summarize Data by Class.

Step 4: Gaussian Probability Density Function.

Step 5: Class Probabilities

## Separate By Class

We will need to calculate the probability of data by the class they belong to, the so-called base rate.This means that we will first need to separate our training data by class. A relatively straightforward operation.We can create a dictionary object where each key is the class value and then add a list of all the records as the value in the dictionary. Below is a function named separate\_by\_class()

## Summrize Dataset

We need two statistics from a given set of data.

We’ll see how these statistics are used in the calculation of probabilities in a few steps. The two statistics we require from a given dataset are the mean and the standard deviation (average deviation from the mean).The mean is the average value and can be calculated as:

• mean = sum(x)/n \* count(x)

Where x is the list of values or a column we are looking.

Below is a small function named mean() that calculates the mean of a list of numbers. We require the mean and standard deviation statistics to be calculated for each input attribute or each column of our data.We can do that by gathering all of the values for each column into a list and calculating the mean and standard deviation on that list. Once calculated, we can gather the statistics together into a list or tuple of statistics. Then, repeat this operation for each column in the dataset and return a list of tuples of statistics.

## Summarize Data By Class

We require statistics from our training dataset organized by class. Above, we have developed the separate\_by\_class () function to separate a dataset into rows by class. And we have developed summarize\_dataset() function to calculate summary statistics for each column. We can put all of this together and summarize the columns in the dataset organized by class values. Below is a function named summarize\_by\_class() that implements this operation. The dataset is first split by class, then statistics are calculated on each subset. The results in the form of a list of tuples of statistics are then stored in a dictionary by their class value.

## Gaussian Probability Density Function

Calculating the probability or likelihood of observing a given real-value like X1 is difficult. One way we can do this is to assume that X1 values are drawn from a distribution, such as a bell curve or Gaussian distribution. A Gaussian distribution can be summarized using only two numbers: the mean and the standard deviation. Therefore, with a little math, we can estimate the probability of a given value. This piece of math is called a Gaussian Probability Distribution Function (or Gaussian PDF) and can be calculated as:

• f(x) = (1 / sqrt(2 \* PI) \* sigma) \* exp(-((x-mean)^2 / (2 \* sigma^2)))

Where sigma is the standard deviation for x, mean is the mean for x and PI is the value of pi.

## Class Probablities

Now it is time to use the statistics calculated from our training data to calculate probabilities for new data.

Probabilities are calculated separately for each class. This means that we first calculate the probability that a new piece of data belongs to the first class, then calculate probabilities that it belongs to the second class, and so on for all the classes. The probability that a piece of data belongs to a class is calculated as follows:

• P(class|data) = P(X|class) \* P(class)

You may note that this is different from the Bayes Theorem described above. The division has been removed to simplify the calculation.

This means that the result is no longer strictly a probability of the data belonging to a class. The value is still maximized, meaning that the calculation for the class that results in the largest value is taken as the prediction. This is a common implementation simplification as we are often more interested in the class prediction rather than the probability.

The input variables are treated separately, giving the technique its name “naive“. For the above example where we have 2 input variables, the calculation of the probability that a row belongs to the first class 0 can be calculated as:

• P(class=0|X1,X2) = P(X1|class=0) \* P(X2|class=0) \* P(class=0)

Now you can see why we need to separate the data by class value. The Gaussian Probability Density function in the previous step is how we calculate the probability of a real value like X1 and the statistics we prepared are used in this calculation. We created a function named calculate\_class\_probabilities() that ties all of this together.

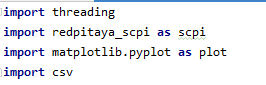
It takes a set of prepared summaries and a new row as input arguments. First the total number of training records is calculated from the counts stored in the summary statistics. This is used in the calculation of the probability of a given class or P(class) as the ratio of rows with a given class of all rows in the training data.

Next, probabilities are calculated for each input value in the row using the Gaussian probability density function and the statistics for that column and of that class. Probabilities are multiplied together as they accumulated. This process is repeated for each class in the dataset. Finally a dictionary of probabilities is returned with one entry for each class.

## Code Explanation: (Data Acquisation Part)

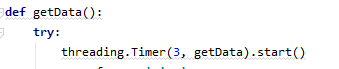
Data acquire from red pitaya: For supervise learning we first need to collect data from red pitaya to raspberry pie. Our main target is to recognize, human wall and car. To collect data from red pitaya which will feed in machine learning part we have to run our data\_get\_common\_code.py file. This file will continuously receive signal data from red pitaya and store into a csv format file for future use. Let’s see how the data acquisition works.

In our script we have imported all the necessary library which will use while the script run. Figure 1 shoes all the necessary library



Now we will initialize scpi Object “rp\_s = scpi.scpi('192.168.128.1')” for sending scpi command to red pitaya. 192.168.128.1 is our red pitaya ip address.

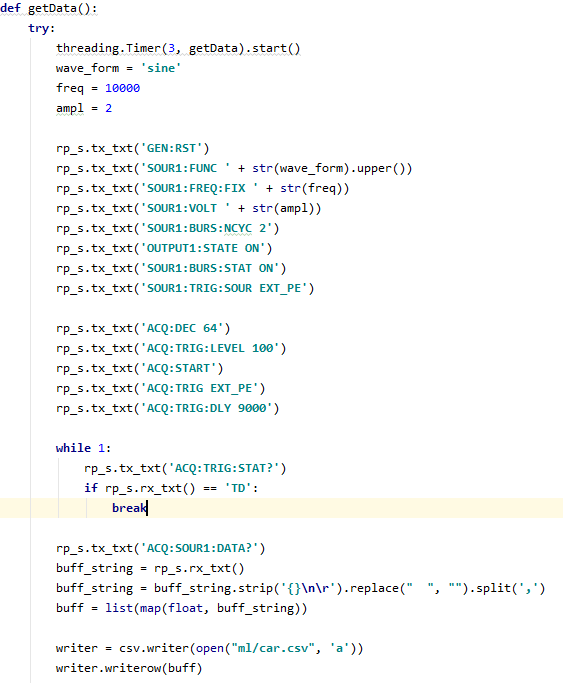
The Standard Commands for Programmable Instruments (SCPI) defines a standard for syntax and commands to use in controlling programmable test and measurement devices, such as automatic test equipment and electronic test equipment



In this script out getData() function is call in every 3 second to get or acquire the signals from red pitaya. We used thread timer so that the method automatically in very

given seconds.

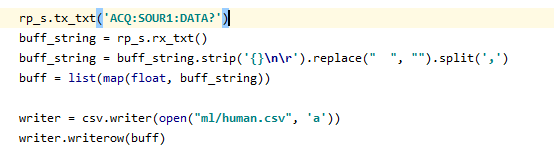
In this function we have used different scpi commands to send to the red pitaya. Let’s see what are the commands we run sequentially to acquire data. The total function is given below figure.

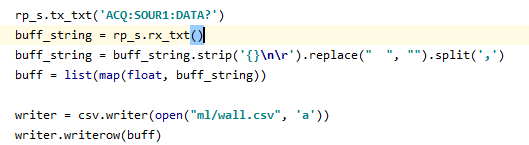


|  |  |
| --- | --- |
| Commands | What it does |
| GEN:RST | Reset generator to default settings. |
| SOUR1:FUNC | Set waveform of fast analog outputs. |
| SOUR1:FREQ:FIX | Set frequency |
| SOUR1:VOLT | Set amplitude voltage of fast analog outputs.Amplitude + offset value must be less than maximum output range ± 1V |
| SOUR1:BURS:NCYC 2 | Set N number of periods in one burst. |
| OUTPUT1:STATE ON | Enable fast analog outputs. |
| SOUR1:BURS:STAT ON | Enable burst (pulse) mode. Red Pitaya will generate R number of N periods of signal and then stop. Time between bursts is P. |
| SOUR1:TRIG:SOUR EXT\_PE | Set trigger source for selected signal. |
| ACQ:DEC 64 | Set decimation factor |
| ACQ:TRIG:LEVEL 100 | Set trigger level |
| ACQ:START | Starts acquisition |
| ACQ:TRIG EXT\_PE | Disable triggering, trigger immediately or set trigger source & edge. |
| ACQ:TRIG:DLY 9000 | Set trigger delay in samples |
| ACQ:TRIG:STAT? | Get trigger status |
| ACQ:SOUR1:DATA? | Read full buf |

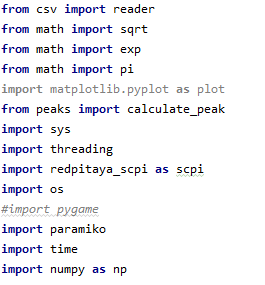
Running these scpi command we continuously get buffered string data and store these in car.csv

We kept run the process until we received at least 100 data for car and then we continued the process to collect data for human and wall also. Code will be same accept for human data file will be human.csv and for wall file will be wall.csv





## Code Explanation: (All Used Libraries)



1. In our programming we have to use different library to get advantage of some complex mechanism.
2. Csv, library to read data and write from csv files.
3. Math for mathematical squat root, exponential, pi and other operations in machine learning.
4. Matplotlib for plot the signal for more visualization.
5. Peaks in our own written python script.
6. Threading for running asynchronous process continuously. As we need to acquire signal from redpitaya in every n seconds.
7. redpitaya\_scpi to run all the scpi commands into red pitaya
8. pygame to play sound on the background
9. Finally, numpy for doing more complex stochastic operation for example Fourier Transform of a signal, mean, variance.

## Required Rechnologies

1. Python (Language)is an interpreted, high-level, general-purpose programming language. Created by Guido van Rossum and first released in 1991. This scripting language is amazing to use for machine learning purpose as it has many machine learning library / api support. To write code in python we choose python version 3.7.3.

To make our solution running in python version 3.7.3 we have made our raspberry pi default python version from 2.7 to 3.7.3.

1. CSV file to store data getting from Red pitaya
2. Paramiko install for ssh connection. As red pitaya starts with ssh connection from red pitaya and we can do it either manually connecting ssh connection or can make it automated in programming. To make it automated we installed paramiko library for establishing ssh connection between red pitaya and raspberry pie.
3. Matplotlib install to plot the signal coming from red pitaya. While doing the research with the signal or data we needed to plot to visualize the process.
4. Pygame install to play sound. Pip install pygame
5. Numpy is the fundamental package for scientific computing with Python. For example, we did fft for the signal and also got the peaks of the fft. These are the functionality were achieved by numpy library.

## Code Explanation: (Feature Extraction Part)

Below Fig-1 code explain how to get feature for each file which we save after data acquisition. Read the data from “Car csv file” then for each data we calculated peaks, mean and variance We have set the label for each object here for Car label is 3.



Fig-1

Similarly, we have done for data file human and wall as shown in Fig 2 and Fig 3 respectively.



Fig 2



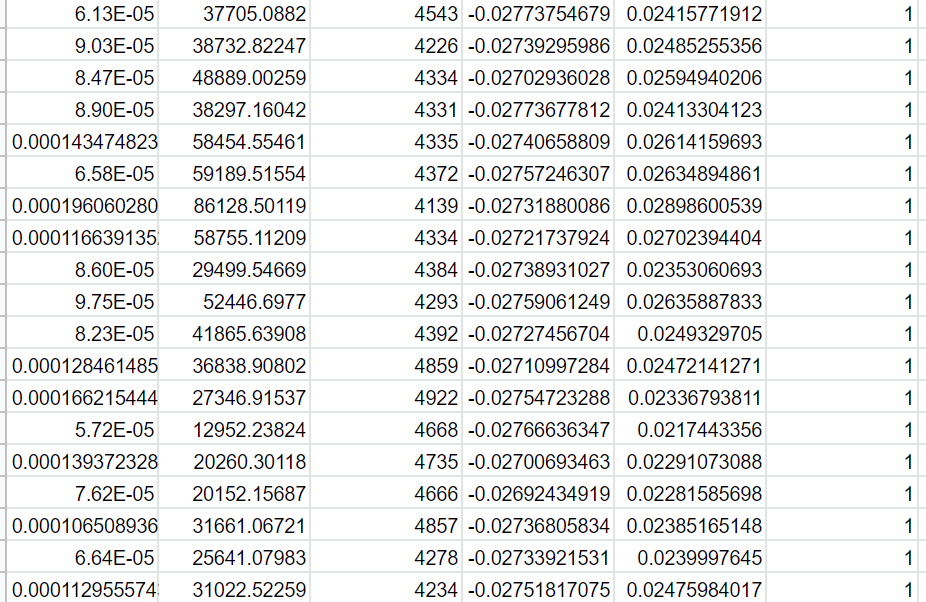
Fig 3

We have save data in csv file, here is separate file which calculate peaks shown in Fig-4



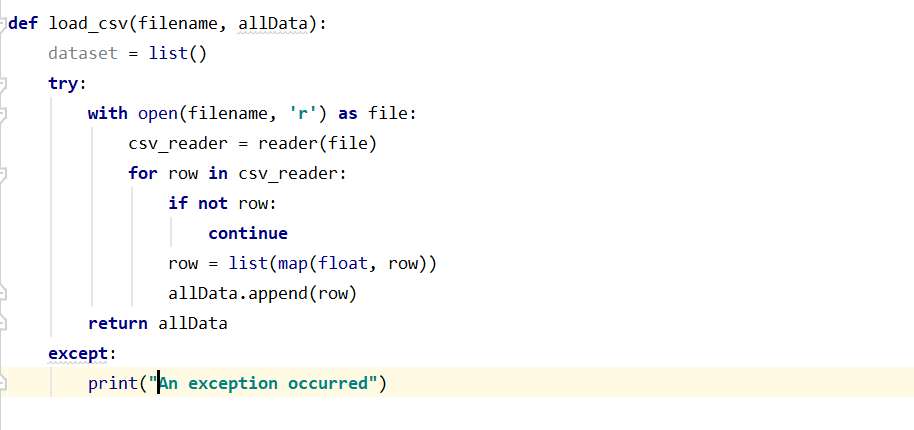
Fig-4

Below Fig-5 are the features extracted file Following are the features, Min peak, max peak,shape,mean,variance and label

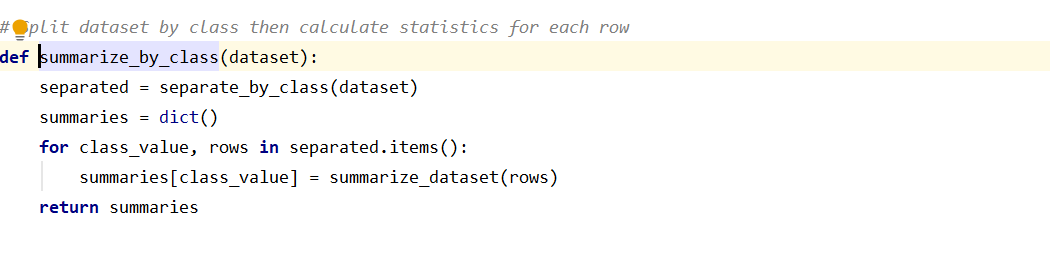


## Code Explanation: (Machine Learning Part)

**Load data:**First we load data from feature file in which we save all features.

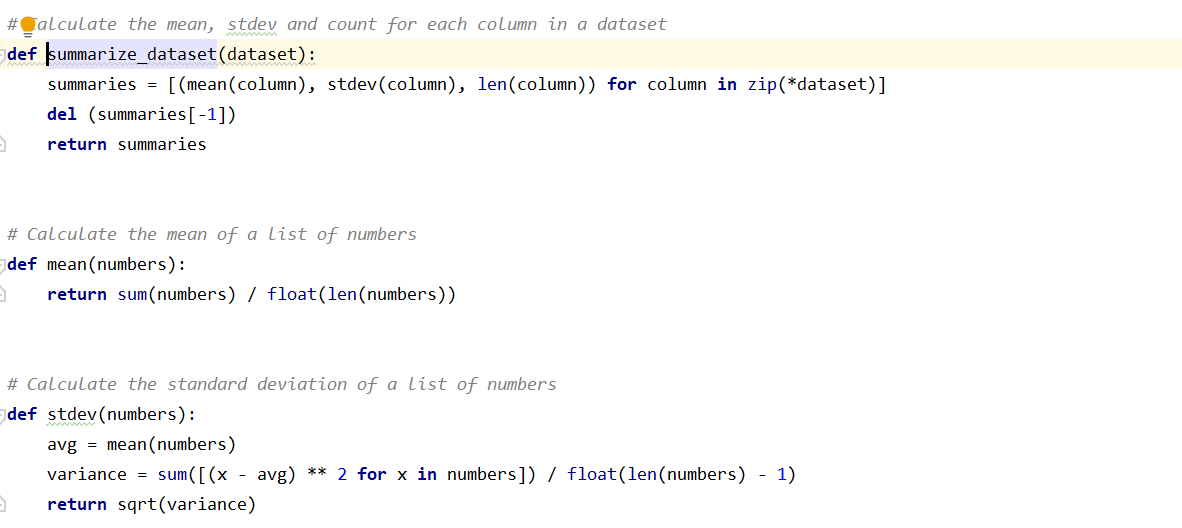


Then we separate data by class and save in dictionary:

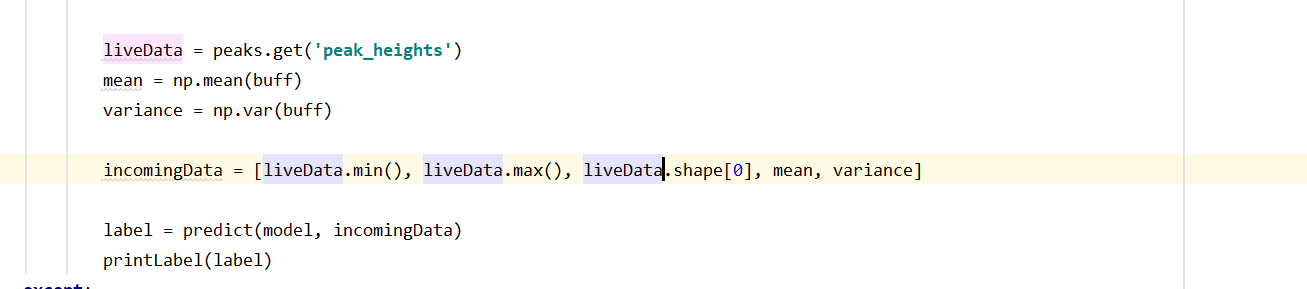


##### 

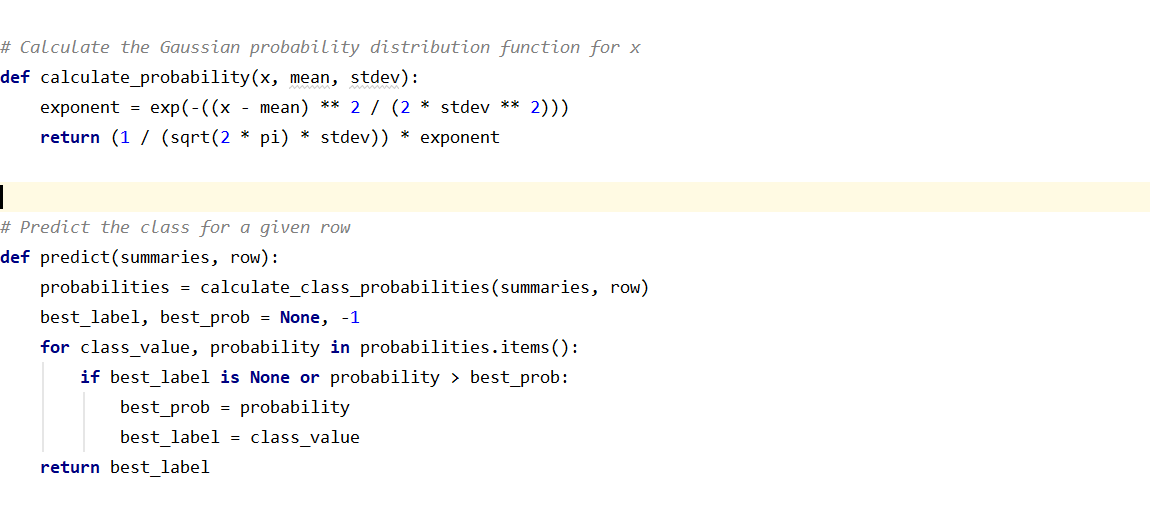
Then for each column we calculated mean, variance and standard deviation:



Till here we have trained out system, now we will call predict function to predict single data.



In predict function we will calculate Gaussian probability of this input data that how likely it Is belongs to anyone of the trained class.



To show the result We have use two approaches.1 to print label on screen 2 to play audio for object name



## Testing: (Plug & Play)

To test our project just power raspberry pie and red pitaya and wait for 40 seconds. Waiting for 40 second is important because raspberry pie files, folder, services, Wi-Fi service all important services should be ready. We have made our script wait to raspberry pie get ready. Point the red pitaya to a wall or human or a car. A monitor can be connected with raspberry pie but it is not mandatory. We have 2 kind of output one is sound and other is printed name of the object. If we have any monitor connected with raspberry pie just open the terminal that’s it, otherwise just connect a headphone or a speaker with raspberry pie to listen which object is detecting.

Our object detection python code is completely automated and enough stable to predict Human, Wall and Car. To make the object detection code automated we added the object\_detection.py path in “/home/pi/.bashrc”

In bashrc last line we added sudo python /home/pi/object\_detection.py

Now whenever the raspberry pie will boot or the terminal open the object\_detection python script will run. As we continuously acquire data from red pitaya we must have a stable ssh connection between raspberry pie and red pitaya. Instead of manually doing it paramiko library helps us to make an automated stable ssh connection



We initially try to make ssh connection using ssh\_connection() function but if it fails because it is still not connected with red pitaya Wi-Fi, it will wait and continue to try in every 5 seconds. Finally, if it will success to make ssh connection it will never try again.

##### Acknowledgment *(Heading 5)*

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

##### References

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first ...”

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors’ names; do not use “et al.”. Papers that have not been published, even if they have been submitted for publication, should be cited as “unpublished” [4]. Papers that have been accepted for publication should be cited as “in press” [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

**This template contains guidance text for composing and formatting seminar paper. Please ensure that all template text is removed from your paper prior to**

**submission to the examination office. Failure to remove**

**template text from your paper may result in your paper being devaluated.**

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an MSW document, this method is somewhat more stable than directly inserting a picture.

To have non-visible rules on your frame, use the MSWord “Format” pull-down menu, select Text Box > Colors and Lines to choose No Fill and No Line.