Table of Content

Note :

In this document for each projects is shown as :

- Link of Jupyter Notebook with implementation code and detail description.
- Overview
- Flowchart of the project.
- Result vedio / gif / graphs.
- The no of star after every project means the level of documentation.

For detail description and Code please go to the Notebook link Provided for every project!!

Kaggle Competetions and Job Entrance Problem :

- Kaggle House Price Prediction :: Data Pre-Processing, ANN with tensorflow low level API and and Hiper-Parameter Tuning. (****)
- Japanese Job Entrance Problem :: Shakura Bloom Prediction (****)

Machine Learning Algorithms from Scratch :

- Neural Network :: Implementation from scratch with raw python (**) (****)
- Decision Tree(ID3) :: Implementation from scratch with continuous feature support. (****)
- Naive Bayes :: Implementation for text classification with text preprocesing from scratch (**)

Reinforcement Learning ALgorithms from scratch :

- DQN(Deep Q Learning) from scratch with Tensorflow-KERAS(**)
- DDPG(Deep Deterministic Policy Gradient) from scratch with Tensorflow(**)

Control Algorithms Implementation from scratch :

- ILQR(Iterative Linear Quadratic Regulator) Implementation from scratch(****)
- MPC(Model Predictive Controller) Implementation from scratch(**)

CNN Projects : (Minimal Doccumentation)

- Yolo with KERAS and Tensorflow for car number plate localization
- Unet with KERAS for City Space Dataset

ROS Project : (Not well doccumented)

- ROS : Simple two linked robot inspired from rrbot(-)
- ROS : Writing a script for driving husky robot and getting feed back

International Robotics Compititions

- University Rover Challenge - 2016

Embedded System Projects for Pi Labs BD Ltd :

- Vault Sequirity : IOT based Vault sequitrity System with AVR Microcontroller
- Safe Box : GPRS based Tracking System with AVR Microcontroller
- Syringe Pump : RTOS Progmable Infusion Pump with AVR Microcontroller
- Digital Weight Machine : Server based digital wight Machine with AVR Microcontroller
- Presentation Link : Embedded System Projects for Pi Labs BD Lt

Academic Project and Thesis (Undergrad) :

- Remote rescue robot with AVR Microcontroller
- Car velocity mseasuring and logging system for genearting drive cycle of Dhaka
- Presentation Link : Academic Project and Thesis

House Price Prediction :: Data Pre-Processing and Hiper-Parameter Tuning

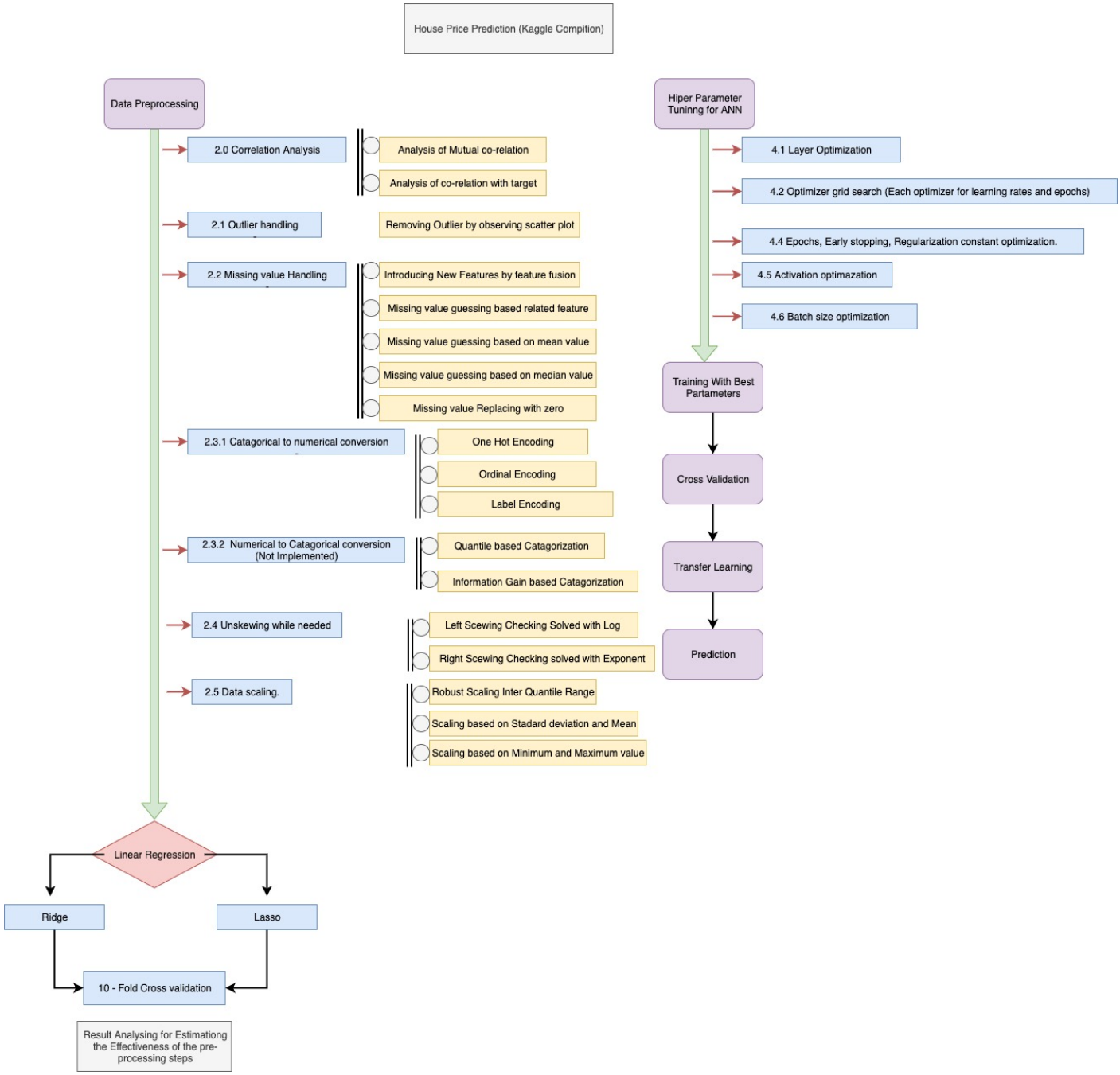
Notebook : House Price Prediction - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Machine-Learning/blob/master/Kaggle/ANN_Tensorflow__Kaggle_House

1. Overview :

- ***House Price Dataset from kaggle.com***
- ***Data Preprocessing***
- ***ANN Class with tensorflow low level API***
- ***Hiperparameter Tuning***
- ***All the graphs of Data preprocessing and Hiperparameter Tuning can be found in Notebook (https://github.com/irfanhasib0/Machine-Learning/blob/master/Kaggle/ANN_Tensorflow__Kaggle_Houseprice_prediction.ipynb)***

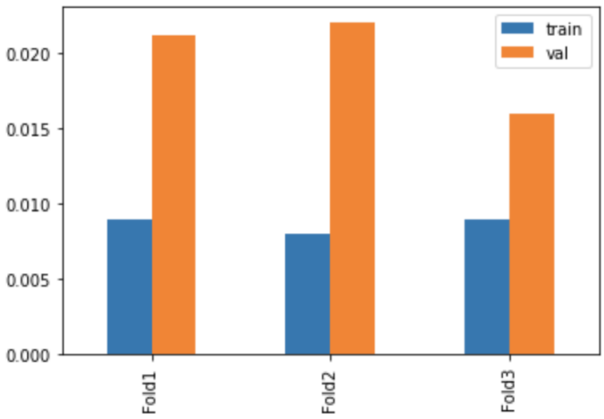
▶ ***Click to expand***

2. Project Flow Chart :

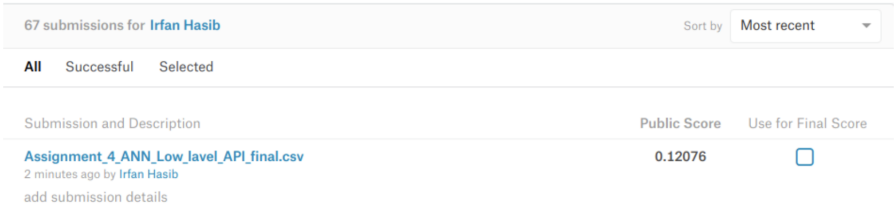


3. Cross validation(MSLE) and Kaggle Result(RMSLE)

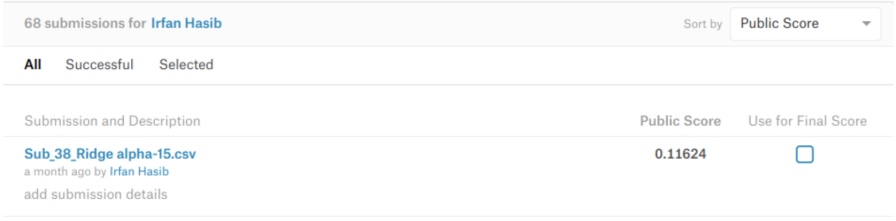
	train	val
Fold1	0.008937	0.021199
Fold2	0.008033	0.022006
Fold3	0.008977	0.016010



Kaggle Score for ANN (trained with 99% train data) : 0.12076 , below is the screenshot ¶



Kaggle Score for Linear regression with l2 regulariaion : 0.11624 , Position 669 (when submitted) ,Screenshot is below



</br>

Go Back to Table of Content

Japanese Job Entrance Problem :: Shakura Bloom Prediction

Notebook : Shakura Bloom Prediction - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Machine-Learning/blob/master/Kaggle/Sakura_TF_NN_Report.ipynb)

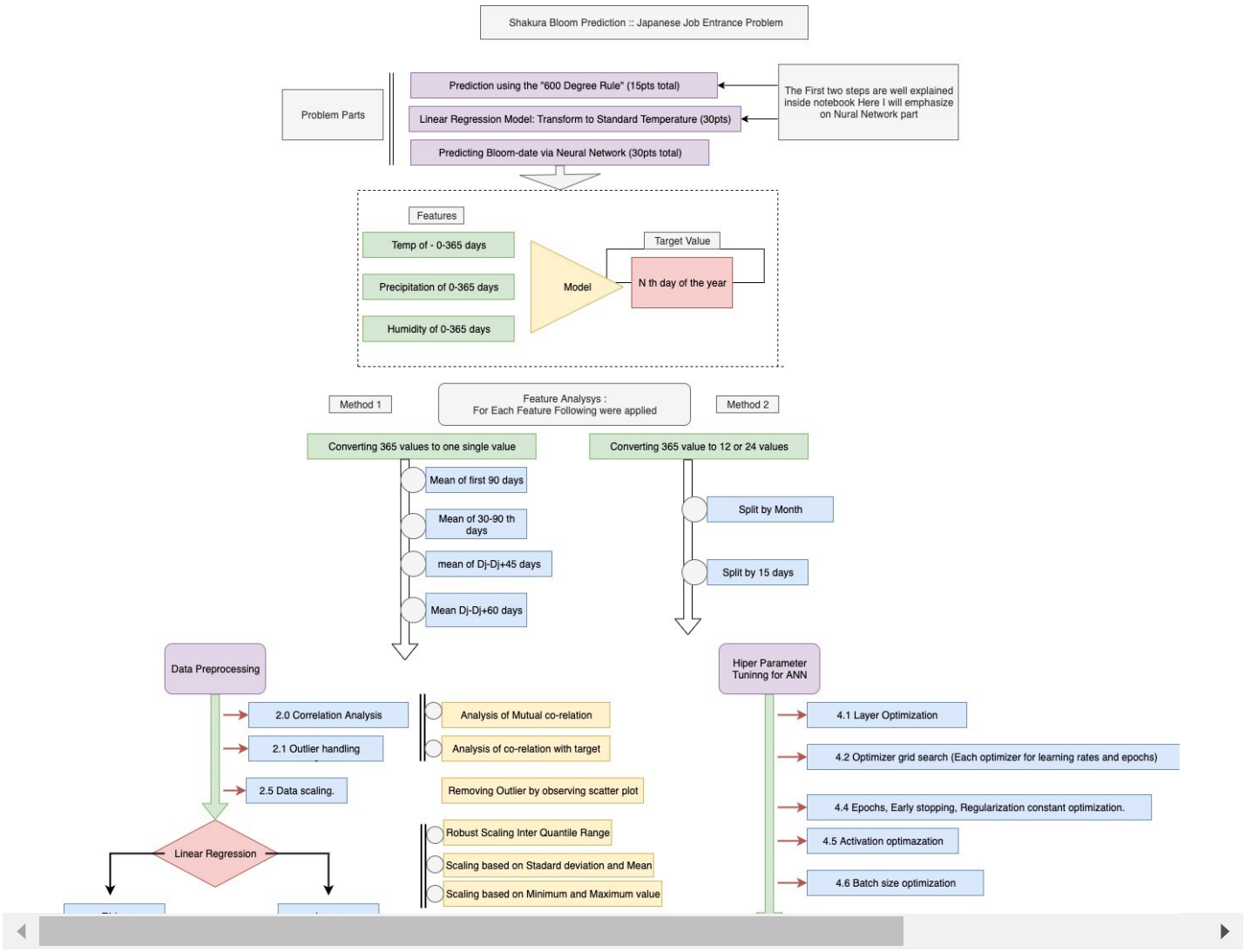
1. Overview

- ***Weather data from japanese meteorological agencym***

- ***Feature Extraction and Data Preprocessing***
- ***ANN Class with tensorflow low level API***
- ***Hiperparameter Tuning***
- ***All the graphs of Data preprocessing and Hiperparameter Tuning can be found in [Notebook (https://github.com/irfanhasib0/Machine-Learning/blob/master/Kaggle/Sakura_TF_NN_Report.ipynb)](https://github.com/irfanhasib0/Machine-Learning/blob/master/Kaggle/Sakura_TF_NN_Report.ipynb)***

▶ ***Click to expand***

2. Project Flow Chart :



Neural Network :: NN Implementation from scratch

[Notebook : NN Implementation from scratch - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Machine-Learning/blob/master/Machine_Learning_Algo_From_Scratch/A

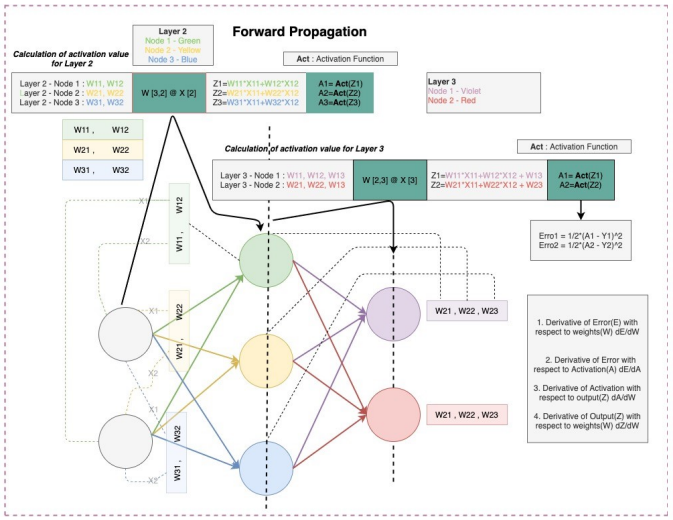
1. Overview :

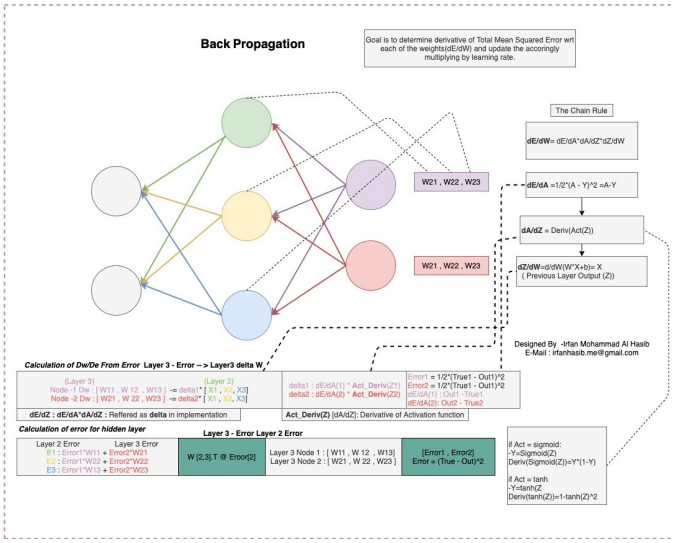
Here I have implemented Neural Network of 3 Layers. I have implemented A Layes Class and functions for Forward propagation,backward propagation and updating weights. I just tested it XOR data. It was fitting good.

- ***XOR Data used for testing***
- ***Forward Propagation***
- ***Backward Propagation***
  - ***Calculating Gradients***
  - ***Updating weights***

▶ ***Click to expand***

2. Process Flow Chart : (Open Image in new tab for full resolution)





Result :

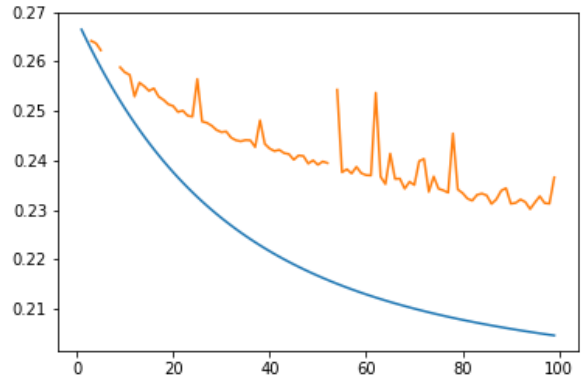
Result of ANN implementation for Bank Note data - mean sqaure error vs epoch -

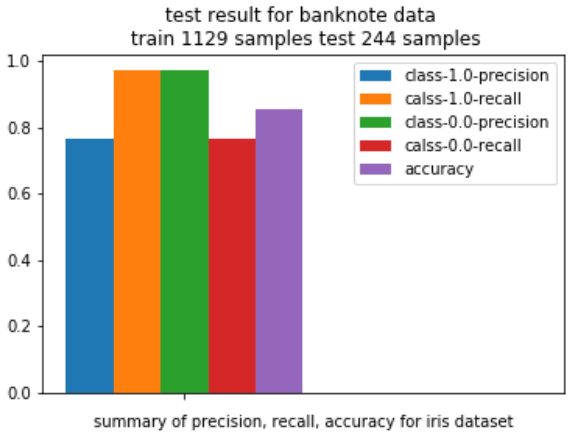
Appendix :

Nodes :

▶ ***Click to toggle expand view***

Go Back to Table of Content





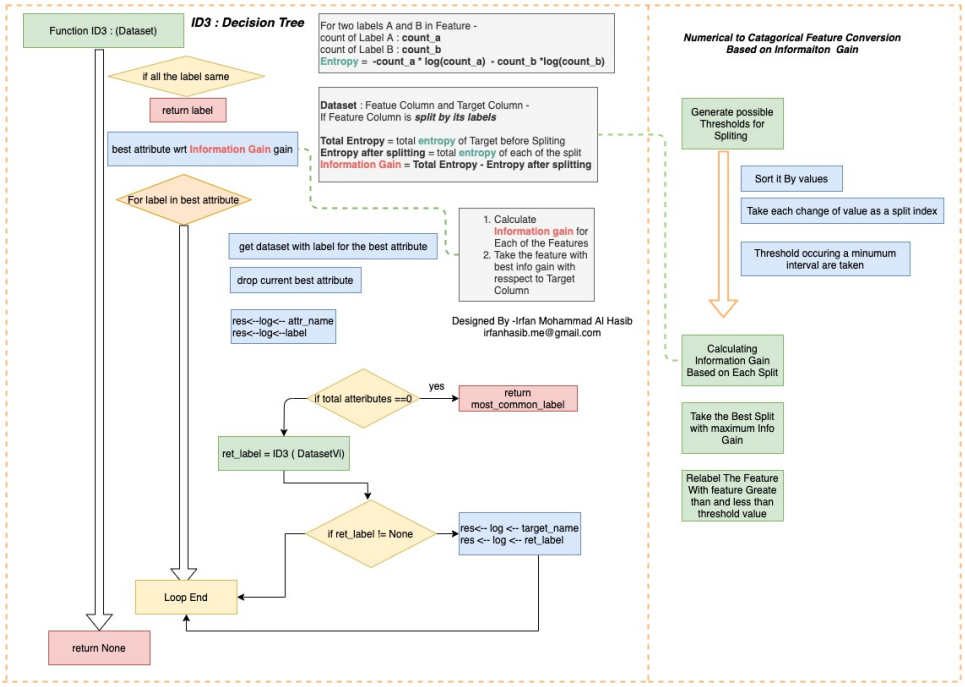
Decision Tree :: ID3 Implementation from scratch

[Notebook : ID3 Implementation from scratch - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Machine-Learning/blob/master/Machine_Learning_Algo_From_Scratch/ID updated.ipynb)]

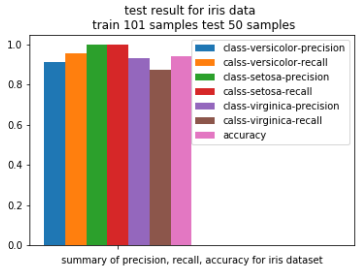
1. Overview

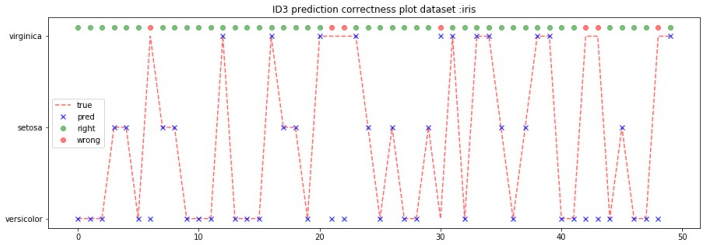
- ***Dataset :***
  Titanic and irish dataset was used for testing ID3.
- ***Steps :***
  - Continuous data spliting based on information gain
  - Information Gain Calculation
  - ID3 Algorithm according to flowchart
- ***Tuning :***
  Reduced Error Pruining.
- ***Prediction :***
  Accuracy,Precision,Recall Reporting.

ID3 Flow Chart of Implementation (Open Image in new tab for full resolution)

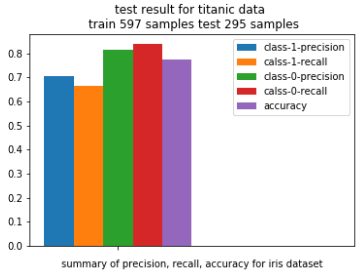


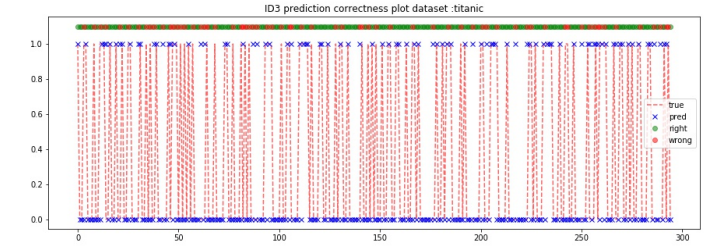
Result of ID3 implementation for iris data - a.Precision Recall and Accuracy and b.True Label vs Prediction -





Result of ID3 implementation for Titanic data - a.Precision Recall and Accuracy and b.True Label vs Prediction





Go Back to Table of Content

Naive Bayes and KNN :: Implementation for text classification

Naive Bayes Implementation from scratch - Notebook (Project Presentation and Code Link)

(https://github.com/irfanhasib0/Machine-Learning/blob/master/Machine_Learning_Algo_From_Scratch/N

1. Overview

- ***Dataset :***

  Archived data from stack exchange is used for classification.

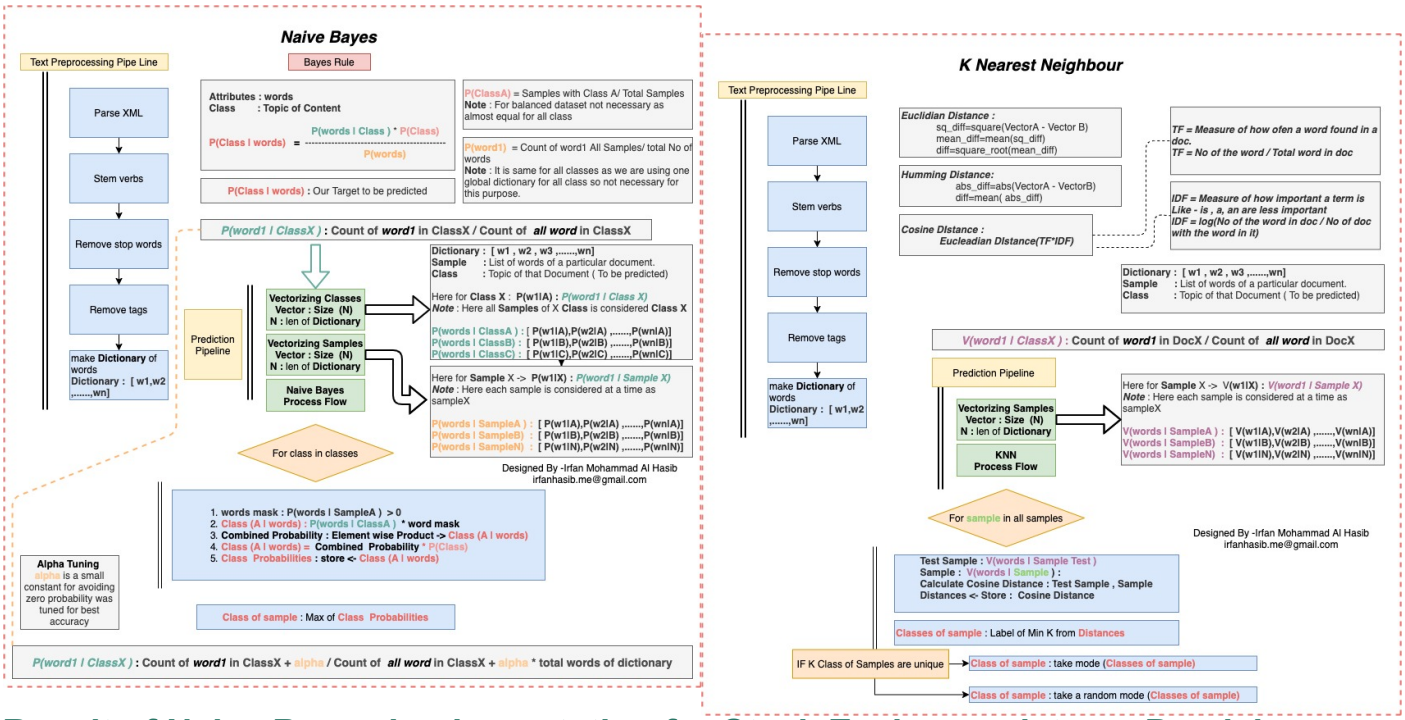
- ***Steps :***
  - Text Preprocessing was done with raw python without nltk.
  - Calculating Attribute(word) Probabilities given each class
  - Calculating Attribute(word) Probabilities given each Samples
  - Applying Bayes Theorem for getting class probalities
  - Max class probality is the predicted label
- ***Steps :***
  - Text Preprocessing was done with raw python without nltk.
  - Calculating Attribute Probability vector for each sample
  - Calculating Cosine/Eucleidian/Humming Distances from test sample to each of train sample
  - Taking labels of K min distance samples from train data
  - Taking mode of K labels as prediction (tie breaks by random choise)
- ***Tuning :***

  The model was Tuned for best alpha values.

- ***Prediction :***

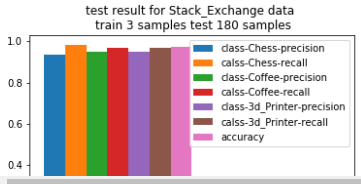
  Accuracy,Precision,Recall Reporting.

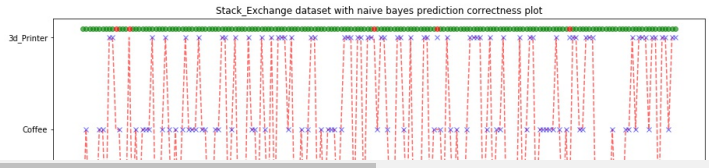
Naive Bayes and K Nearest Neighbour algorithm applied on the procesed text.



Result of Naive Bayes implementation for Stack Exchange data - a.Precision Recall and Accuracy b.True Label vs Prediction

Go Back to Table of Content


test result for Stack_Exchange data
train 3 samples test 180 samples
1.0
0.8
0.6
0.4
class-Chess-precision
calss-Chess-recall
class-Coffee-precision
calss-Coffee-recall
class-3d_Printer-precision
calss-3d_Printer-recall
accuracy


Stack_Exchange dataset with naive bayes prediction correctness plot
3d_Printer
Coffee

ILQR and MPC :: Implementation from scratch for self driving car simulator

Notebook : ILQR Implementation - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Control-algorithms/blob/master/MPC_GYM_CAR_RACING_V0/ilqr_on_
exp.ipynb)
Notebook : MPC Implementation - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/Control-algorithms/blob/master/MPC_GYM_CAR_RACING_V0/mpc_on

1. Overview :

- ***Simulation Platform :***
  - AIRSIM by Microsoft Inc.
  - OpenAI GYM - Car Environment was tested
- ***IO :***

  Input --> Map Points , Output --> Steering Angle, Acclelation, Brake
- ***MAP to Trajectory : Steps :***
  - Environment Module
  - Map Tracker Module
  - Data Preprcessor Module
- ***Optimaization Algorithm : Trajectory to Optimal Steering,Accleration,Brake :***
  - iLQR using Raw Python and Numpy
  - MPC using Python , Numpy and CVXPY

▶ ***Click to expand***

MPC modeling

$State\ Space : z = [x, y, v, \phi] \qquad where, x : x - position, y : y - position, v : velocity, \varphi : yawan$

$Action\ Space : u = [a, \delta] \qquad where, a : accellation, \delta : steeringan$

Cost and Constraints :

$$Cost:$$

$$min\ Q_f(z_{T,ref} - z_T)^2 + Q\Sigma(z_{t,ref} - z_t)^2 + R\Sigma {u_t}^2 + R_d\Sigma(u_{t+1} - u_t)^2$$

$$z_{ref}\ :\ target\ states$$

$$Constraints:$$

$$z_{t+1} = Az_t + Bu + C$$

$$Maximum\ steering\ speed = abs[u_{t+1} - u_t] < du_{max}$$

$$Maximum\ steering\ angle = u_t < u_{max}$$

$$Initial\ state = z_0 = z_{0,ob}$$

$$Maximum\ and\ minimum\ speed = v_{min} < v_t < v_{max}$$

$$Maximum\ andminimum\ input = u_{min} < u_t < u_{max}$$

State Space model for Car system

$$z_{t+1} = Az_t + Bu + C\ where,\ A = \begin{bmatrix} 1 & 0 & cos(\bar{\phi})dt & -\bar{v}sin(\bar{\phi})dt \\ 0 & 1 & sin(\bar{\phi})dt & \bar{v}cos(\bar{\phi})dt \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{tan(\bar{\delta})}{L}dt & 1 \end{bmatrix}$$

$$B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ dt & 0 \\ 0 & \frac{\bar{v}}{Lcos^2(\bar{\delta})}dt \end{bmatrix} C = \begin{bmatrix} \bar{v}sin(\bar{\phi})\bar{\phi}dt \\ -\bar{v}cos(\bar{\phi})\bar{\phi}dt \\ 0 \\ -\frac{\bar{v}\bar{\delta}}{Lcos^2(\bar{\delta})}dt \end{bmatrix}$$

▶

Expand to see derivation

Iterative Linear Quadratic Regulator :

1. U as a function of Previous U , X and Previous X :

$$u'(i) = u(i) + k(i) + K(i)(x'(i) - x(i))$$

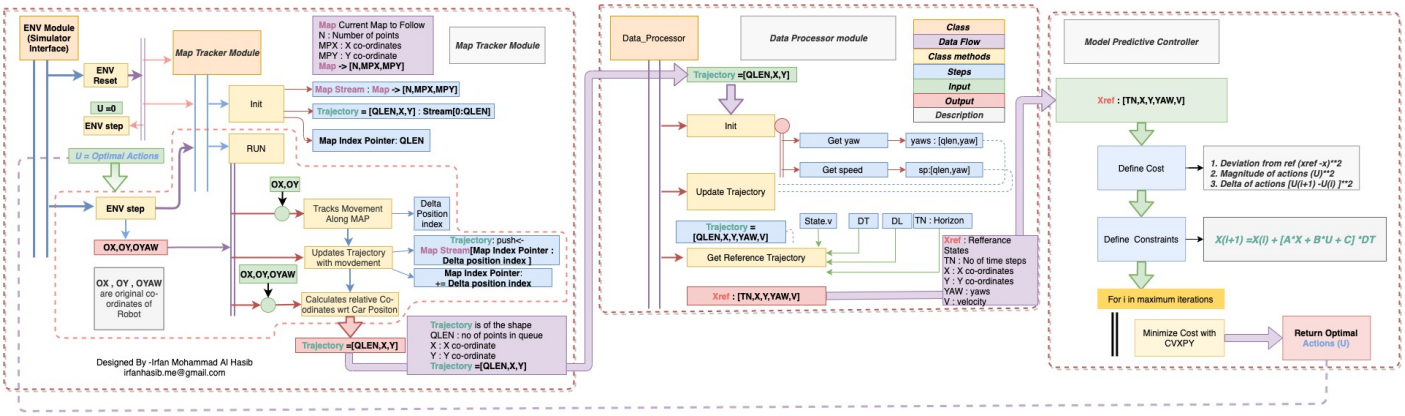
▶

2. Calcualting K and k : (Expand to see)

4. For detail derivation please see the paper :

Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization by -Yuval Tassa citations(369) according to February 27 2020, Published on - 2012

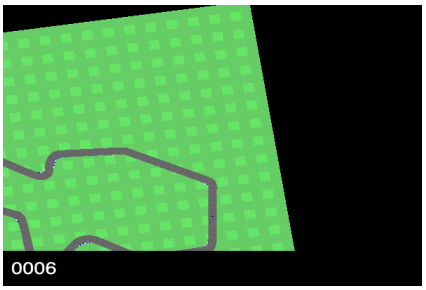
2. Project Flow Chart :

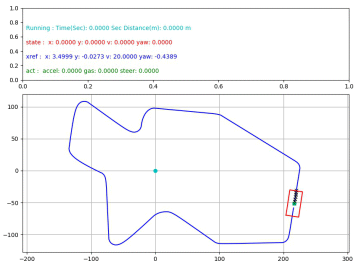


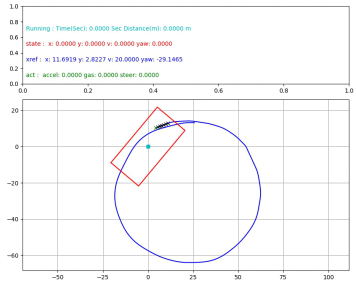
▶ ***Click to expand***

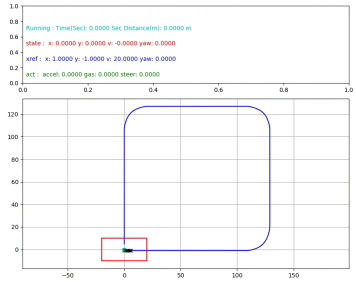
3. Results (ILQR) :

* OpenAI Gym Car Environment * Airsim City Space Environment * Airsim Neighbourhood Environment









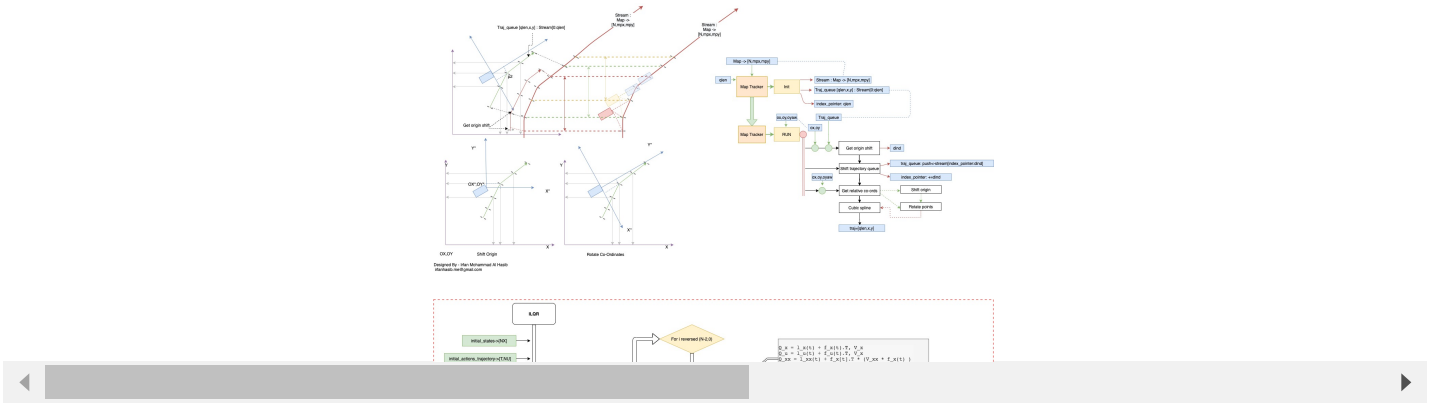
Inspired from -

-AtsushiSakai/PythonRobotics (https://github.com/AtsushiSakai/PythonRobotics/blob/f51a73f47cb922a12659f8ce2d544c347a2a8156/PathTracki L301)

Reference

- AtsushiSakai/PythonRobotics (https://github.com/AtsushiSakai/PythonRobotics/blob/eb6d1cbe6fc90c7be9210bf153b3a04f177cc138/PathTr
  L102)
- Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization by -Yuval Tassa

Appendix : Map Tracker and iLQR



DQN and DDPG:: Implementation from scratch

Notebook : Mountain Car with DQN - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/RL-Algorithms/blob/master/Deep_Q_Learning_mc.ipynb) Notebook : Pendulum with DDPG - Notebook (Project Presentation and Code Link) (https://github.com/irfanhasib0/RL-Algorithms/blob/master/Deep_Q_Learning_mc.ipynb)

1. Overview

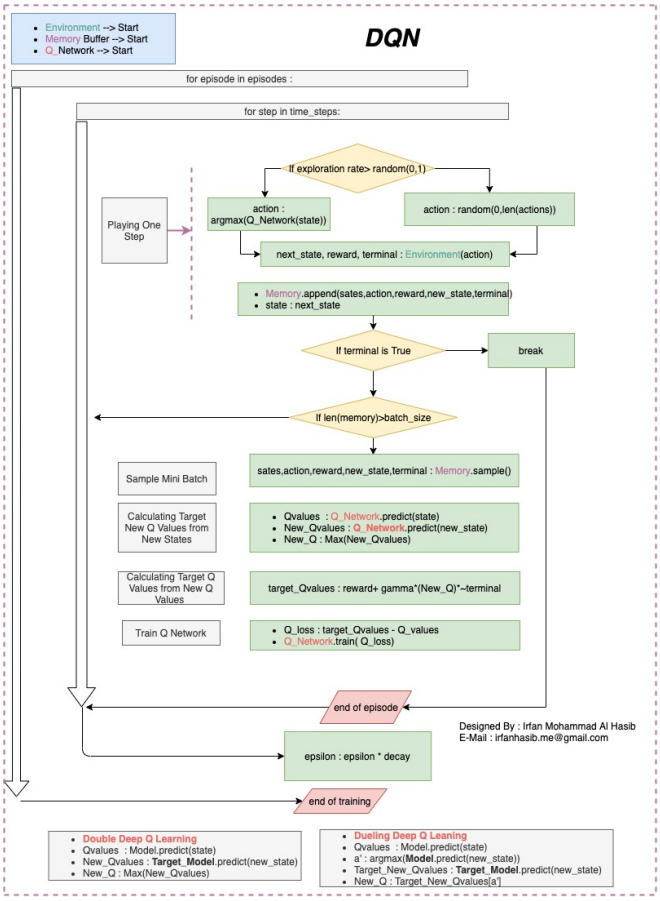
- ***DQN Environments***
  OpenAI gym --> Mountain Car ENvironment
- ***DQN Environments***
  OpenAI gym --> Pendulumn Environment
- ***DQN Steps***
  For each time step of a episode
  - action : explore -> random | or exploit -> max Q value based on epsilon decay
  - play one step -> store experience
  - sample minibatch
  - target Q values : reward+gamma*Q_network(new_state,new_actions)
  - train Q network
- ***DDPG Steps***
  For each time step of a episode
  - actor(state): action + OU Noise
  - play one step -> store experience
  - sample minibatch
  - target Q values : reward+gamma*target_critic(new_state,new_actions)
  - actor loss from critic(state,action)
  - train actor and critic
  - target networks : networks*tau + (1-tau)*target_networks

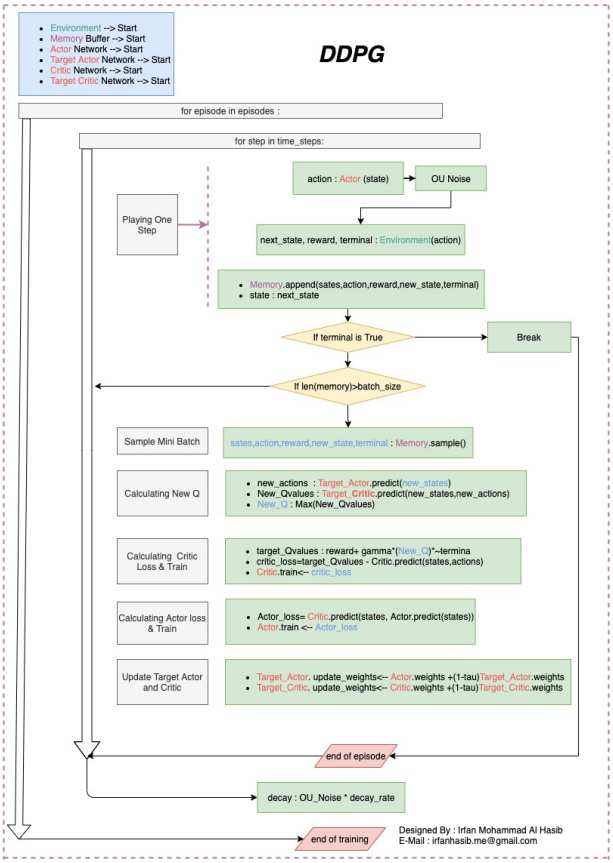
2. Detailed Flow Chart for DQN and DDPG : (Please open in New tab for proper resolution)

</br>

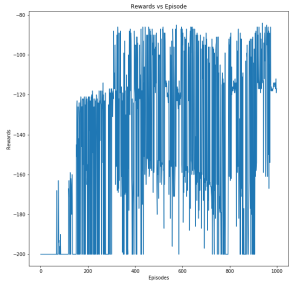
3. Results

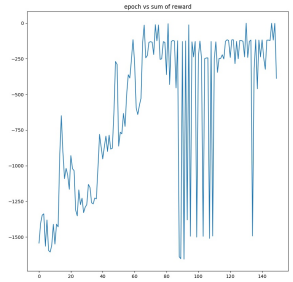
- a. Results DQN on Mountain Car (Left):
- b. Results DDPG on Pendulum (Right):



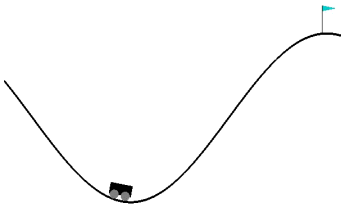


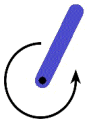
- c. Tset DQN on Mountain Car (Left):
- d. Test DDPG on Pendulum (Right):





Irfan Mohammad AL Hasib
irfanhasib.me@gmail.com



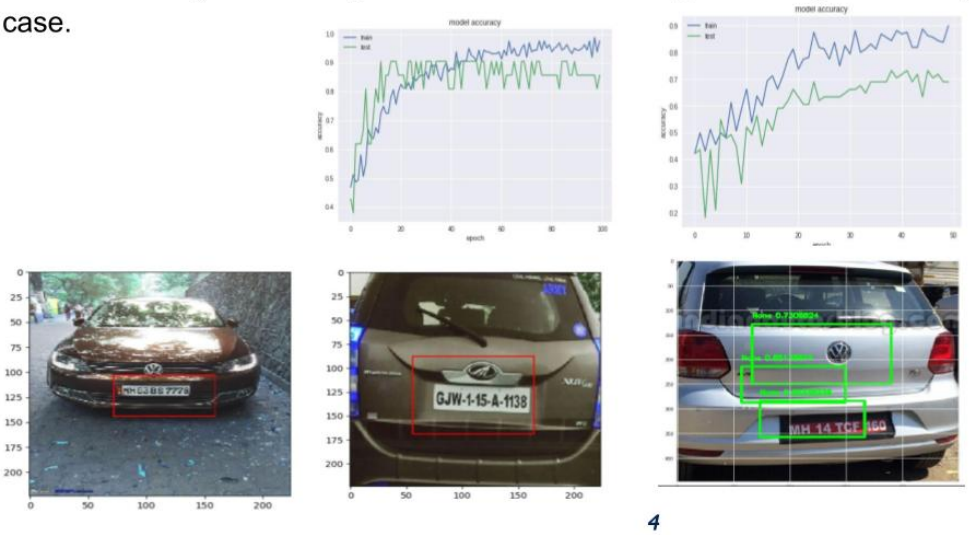


Yolo with KERAS and Tensorflow for numberplate detection

[Notebook : Yolo-V3 (https://github.com/irfanhasib0/CNN-Projects/blob/master/Yolo_NET_V_1.ipynb)](https://github.com/irfanhasib0/CNN-Projects/blob/master/Yolo_NET_V_1.ipynb)
[Notebook : Yolo with VGG16 (https://github.com/irfanhasib0/CNN-Projects/blob/master/VGG_NET_V_1.ipynb)](https://github.com/irfanhasib0/CNN-Projects/blob/master/VGG_NET_V_1.ipynb)

Number plate localization with VGG net and YOlO

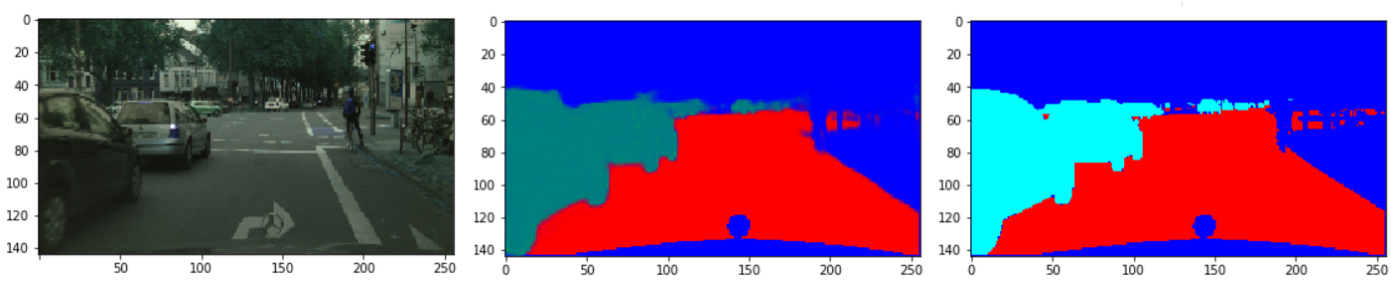
I have implemented VGG 16 and YOLO with KERAS and tensorflow with help of some Git repositories and tutorials, accuracy on train data was about 90% and test data was about 75-80%, I have used Indian dataset available on kaggle and searching for a rich dataset for the problem. YOLO accuracy is not that good till now. I am working on finding the best hypothesis still now for this case.

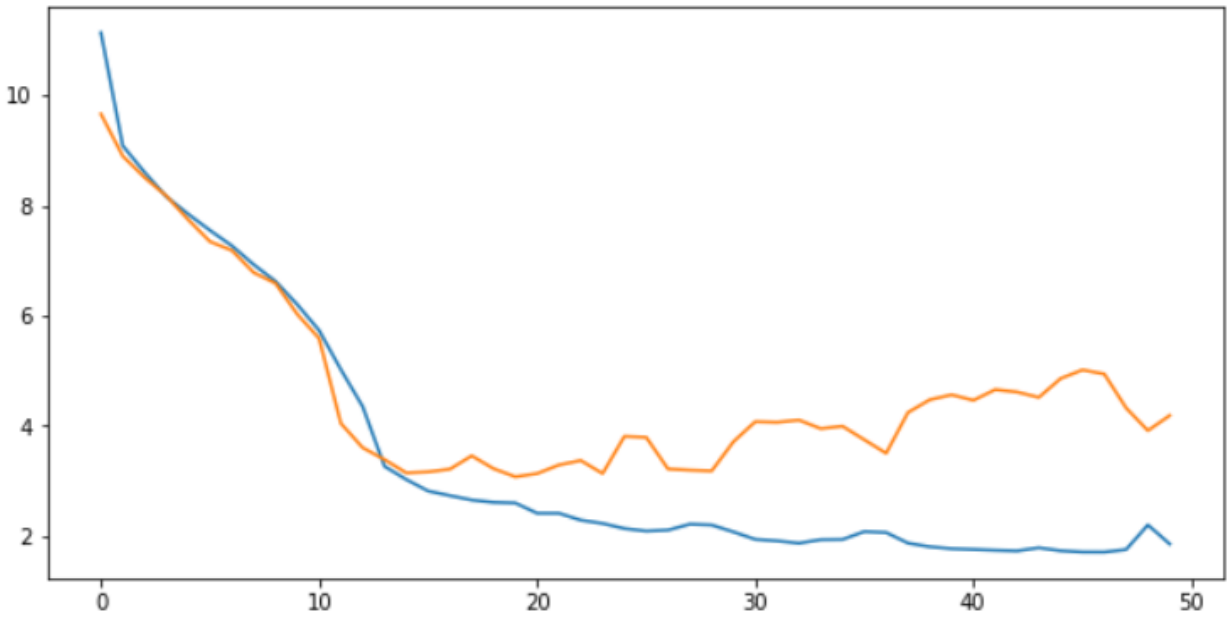


4

Unet with KERAS for City Space Dataset

[Notebook :Unet for segmenting City Space Dataset (https://github.com/irfanhasib0/CNN-Projects/blob/master/as_unet_seg-cs.ipynb)](https://github.com/irfanhasib0/CNN-Projects/blob/master/as_unet_seg-cs.ipynb)





Go Back to Table of Content

ROS : Simple two linked robot inspired from rrbot

- URDF Link(https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_description (https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_description))
- Controller Link(https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_control (https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_control))
- Gazebo Link(https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_gazebo (https://github.com/irfanhasib0/ros_ws/tree/master/src/rrbot/rrbot_gazebo))
- Vedio Link (https://youtu.be/lJbyy89X7gM (https://youtu.be/lJbyy89X7gM))

Embedded System Projects : Pi Labs BD LTD

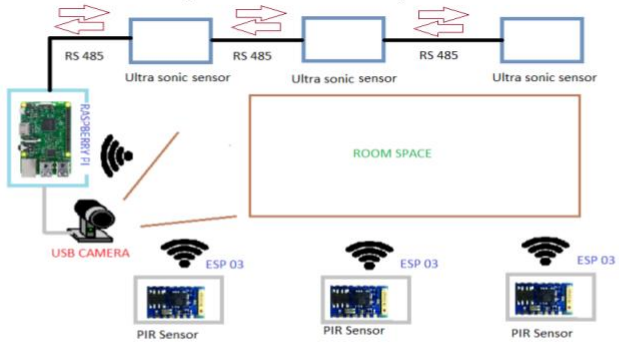
All these projects I did as an employee of Pi Labs BD Ltd. www.pilabsbd.com

VAULT SECURITY SYSTEM - PI Labs BD Ltd.

I was co-developer of vault security system v-1.0

I have developed -

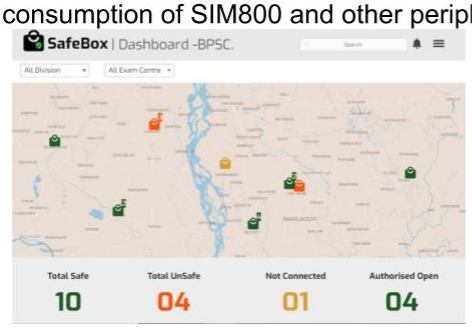
- The wireless PIR (passive infrared sensor) sensor nodes with ESP 03 (ESP 8266 based Wi-Fi transceiver module) for sending data to raspberry pi server.
- Motion detection with USB camera on raspberry pi
- Sending Ultra sonic sensor data to raspberry pi via RS485 protocol.
- Worked on low power modes of operation of ESP 03.



SafeBox – Box tracking Device for Pi Labs BD Ltd.

I was co-developer of vault Safe Box v-1.0 a box tracking and box safety status reporting device.

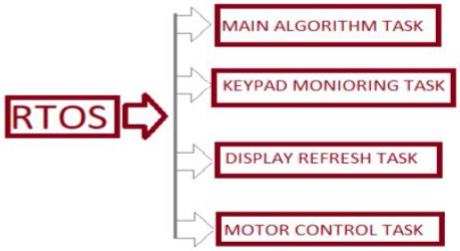
- I have worked on sending data packet to the server at regular interval via GSM network with SIM800 with least data loss.
- I have developed an algorithm for keeping back up of box status data on SD card (specially when GSM network strength is low) with FATFS on AVR platform by SPI interfacing with SD Card and reporting backup to server when network is back again.
- I have also worked on the battery backup time enhancement and researched on power consumption of SIM800 and other peripherals.



SYRINGE INFUSION PUMP Pi Labs Bangladesh Ltd.

I was main developer of syringe infusion pump.

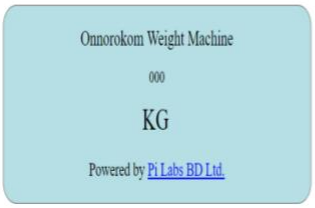
- The pump was programmable for pumping liquid at desired rate for desired time period.
- The system was designed on AVR platform using RTOS (Real time operating system).
- Keypad and Display for programming the pump.
- Stepper motor controlling with TB6560 driver with 1/32 micro step controlling.
- 3 modes of operation. They are – Automatic Mode, Manual Mode and Refill Mode.
- Pumping volume precision up to 0.01 cc/s

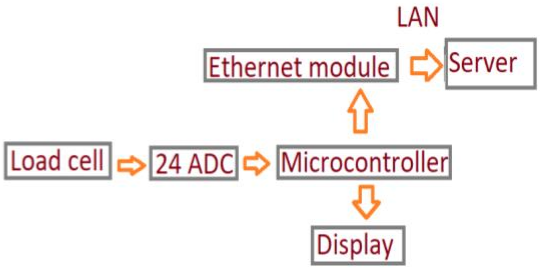


6

Online weight machine for Pi Labs Bangladesh Ltd.

- The weight machine was developed on AVR microcontroller platform.
- The weight machine capable of showing up to +/- 3 g precision using 24 bit ADC(Analog to digital conversion).
- The weight machine by e capable of sending measured weight directly to server by using LAN connection.
- I have also developed prototype of version two with native raspberry pi server with python-flask and Wi-Fi support.





7

Academic Project and Thesis:

- My undergrad project of intrumentation and measurement course
- My undergrad thesis

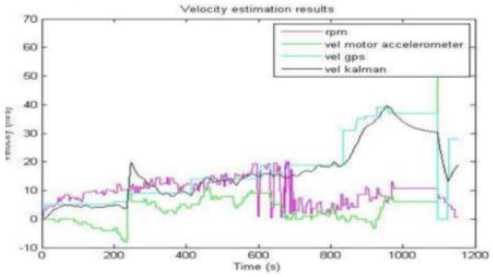
Academic Project and Thesis

Level-3 Term-1 project-Remote Controlled Surveillance Robot:

❖ Development of a Remote c ontrol Surveillance bot Hardware and Software Platform for soil sample collection and data acquisition and transmission.

Level 4 -Thesis

❖Development of a precision vehicle speed measurement and data logging system for making a Drive Cycle of Bangladesh. (A study of precision velocity measurement system including Inertial Measurement system (IMU), GPS and sensor fusion with kalman filter.)



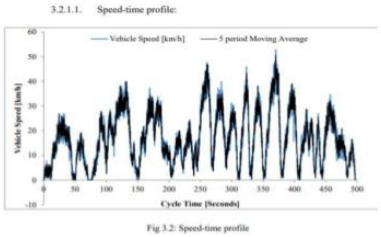


Fig 3.2: Speed-time profile

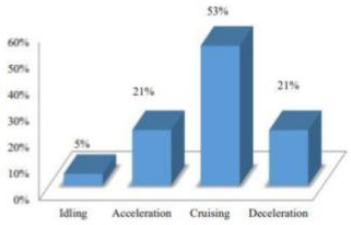


Fig 3.8: Standard Modes Distribution

8

Go Back to Table of Content

University Rover Challenge - 2016

Critical Design Rivew : Video Link (https://www.youtube.com/watch?v=MIN-VFj14LE)

University Rover Challenge -2016 , Utah ,USA

Along with my team **Interplanetar** I have participated and attained 5th position in Phobos final in University Rover challenge 2016 at Utah, USA. He competition was organized by mars society ,USA. I have developed the Robotic Arm section and Object griping , Griping and rotating wrist simultaneously for knob opening and soil sample collection task.

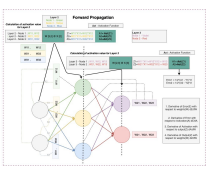
2

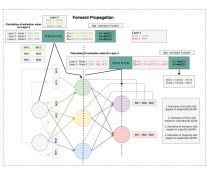
Go Back to Table of Content

<!DOCTYPE html>

Images Side by Side

How to create side-by-side images with the CSS float property:





In [ ]: