Configuration Manual for A Machine Learning Framework for Prediction of Empathy through Eye-tracking and Speech Analysis

MSc Research Project
Data Analytics

Rahul Badarinath
Student ID: 20247702

School of Computing
National College of Ireland

Supervisor: Dr. Anu Sahni
Supervisor: Dr. Paul Stynes

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rahul Badarinath
Student ID:	20247702
Programme:	Data Analytics
Year:	2022
Module:	MSc Research Project
Supervisor:	Dr. Anu Sahni
Submission Due Date:	15/08/2022
Project Title:	Configuration Manual for A Machine Learning Framework for Prediction of Empathy through Eye-tracking and Speech Analysis
Word Count:	1665
Page Count:	9

Signature:	
Date:	15th August 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual for A Machine Learning Framework for Prediction of Empathy through Eye-tracking and Speech Analysis

Rahul Badarinath
20247702

1 Introduction

This document is to provide details about the steps involved in the implementation of the research project - 'A Machine Learning Framework For Prediction of Empathy Through Eye Tracking and Speech Analysis'.

The configuration manuals provides a detailed step-by-step process of how the research was conducted and how the results were procured. The research aims to determine how efficiently can you measure the level of empathy in humans through Eye-tracking and Speech Analysis?

This implementation involves the use of a YOLOv5 model in conjunction with a ResNet50 model.

The structure of this research is presented in the phases through which this research was conducted. They are:

- Section 2: Hardware and Software requirements - This section comprises of the system specifications, tools and techniques used.
- Section 3 Data Collection - This section will comprise of the process adopted to collect data from the participants.
- Section 4 Data Pre-Processing - This section will comprise of the various ways in which data extraction, data transformations, exploratory data analysis, and modelling was done.
- Section 5 Implementation - This section outlines the implementation overview of this project.
- Section 6 Conclusion - This section will comprise of the findings and the conlcusion of this manual.

2 Hardware and Software Requirements

The research project used multiple devices with various hardware and software configurations. Each of these is mentioned below.

- Eye-tracking: Eye-tracking glasses were provided by the National College of Ireland. This device was manufactured by SensoMotoronic Instruments(SMI) who had also provided a Dell Laptop and a Samsung Note 4 as an interface with the device. The glasses comprised of 3 cameras, two inward of the glasses to track the retina and presence of the eye, on in between the eyes outward of the glasses to track the gaze behavior of the participants. This device is a wired setup and requires to be connected to a smartphone to be used. The use of these glasses is highly popular in the research community and has been used by millions of users. These glasses record the gaze at a sampling rate of 60 hertz. SMI's BeGaze software is used to extract the eye-gaze features and other eye-tracking metrics such as blink, saccades,etc.
- Speech Analysis: The speech samples are recorded using a standard voice recorder embedded within the phone. The recorder in the phone records each of the samples at 44.1Khz. Each of these samples is then processed using Matlab software where they are re-sampled at 16khz for processing. The same software has a wide array of auditory analysis options such as extraction of MFCC features, extraction of logarithmic spectral amplitude (LSA), de-noising using Short-time Spectral Amplitude algorithms (STSA), resampling of the speech signals, etc.
- Storage: The data that was collected, analyzed or procured from various sources were stored in OneDrive by Microsoft. This central data repository was provided by the National College of Ireland.
- Machine Learning (ML): There were precisely two model built, one was the YOLOv5 and the other was the Resnet50 model. THe YOLO model was built on top of GPUs in Google Colab, due to the need for high computing power to train the model. The Resnet model was built on the local machine with Nvidia 1660 TI gpu, and an Intel i7 9th gen processor with python and jupyter as the coding mediums.

3 Data Collection

The Eye-tracking experiment was performed on 50 participants all directly linked to the university. Proper ethical considerations were taken prior to the experiments. Out of the 50 participants, there were a total of 14 female participants, and a total of 36 Male participants between the ages of 19 to 30 years old. The participants were asked to watch video-based stimuli for which their eyes were tracked. Each participant was asked to self-report their sadness level on a scale of 1-10 before the start of the experiment. This data was taken into an excel sheet, along with their Name, Age, and Sex. The participant was then made to sit in front of the screen and a 3-point calibration was performed using the eye-tracking glasses. The participants were then asked to watch the videos was their eye-gaze was tracked. On completion, they were asked to self-report their sadness levels after watching the videos. The participants were then tasked with a 10 questions based Memory Test, which carried 1 mark each. Post which, the participants were asked to answer any one of the two questions asked, to record their speech sample, in under 1 minute. On completion of this, the participants were asked to fill out an empathy-based questionnaire which comprised of 10 questions each marked based on a 7-point likert scale. The total time duration for this experiment was between 25-30 minutes per participant. The list of features extracted is provided below:

- Age - Age of the participant (Self Reported)
- Sex - Gender of the participant (Self Reported)
- Sadness Level Before Experiment - Sadness level before watching the stimuli video (Self Reported)
- Sadness Level After Experiment - Sadness level after watching the stimuli video (Self Reported)
- Difference in Sadness - Differences in self reported sadness level before and after watching the video (Calculated Field)
- Memory test - 10 Multiple Choice-based questions based on the visual stimuli.
- Average Distance from Right Eye - The average distance from right eye of the actor on screen. To derive this, dlib library was used to identify the coordinates of the corner of the eye. Further, the center-points for both eyes (iris) were calculated, and the average distance of the point-of-gaze from the centre of the eye was calculated.
- Average Distance from Left Eye - The same process as above was repeated for the left eye.
- blink mean - Average duration of blinks of the participant (Obtained from SMI BeGaze Analysis Software)
- Blink Std - Standard deviation of blink duration (Obtained from SMI BeGaze Analysis Software)
- Blink Percentage - The percentage of the total number of blinks (Obtained from SMI BeGaze Analysis Software)
- Saccade mean - Average Saccade duration (Obtained from SMI BeGaze Analysis Software). Saccades are defined as the rapid movement of eyes away from a fixed point which is commonly termed as a Distraction. For this experiment, the laptop screen was the field of view, and any gaze pattern outside the screen was deemed as a saccade.
- Saccade standard deviation - Standard deviation of the saccade duration (Obtained from SMI BeGaze Analysis Software)
- Saccade Percentage - The total number of Saccades expressed as a percentage (Obtained from SMI BeGaze Analysis Software)
- Heat Maps - A heatmap essentially highlights the area of higest concentration over a given period of time. For this experiment, the highest concentration of the participant for the entirety of the visual stimuli was drafted in the form of a heatmap. The Heatmap was generated using python based numpy libraries and seaborn for visualization. A YOLO model was trained to depict the circles attributed to the point of gaze and the pattern of occurence of the gaze on to a laptop screen. (Python calculated field)

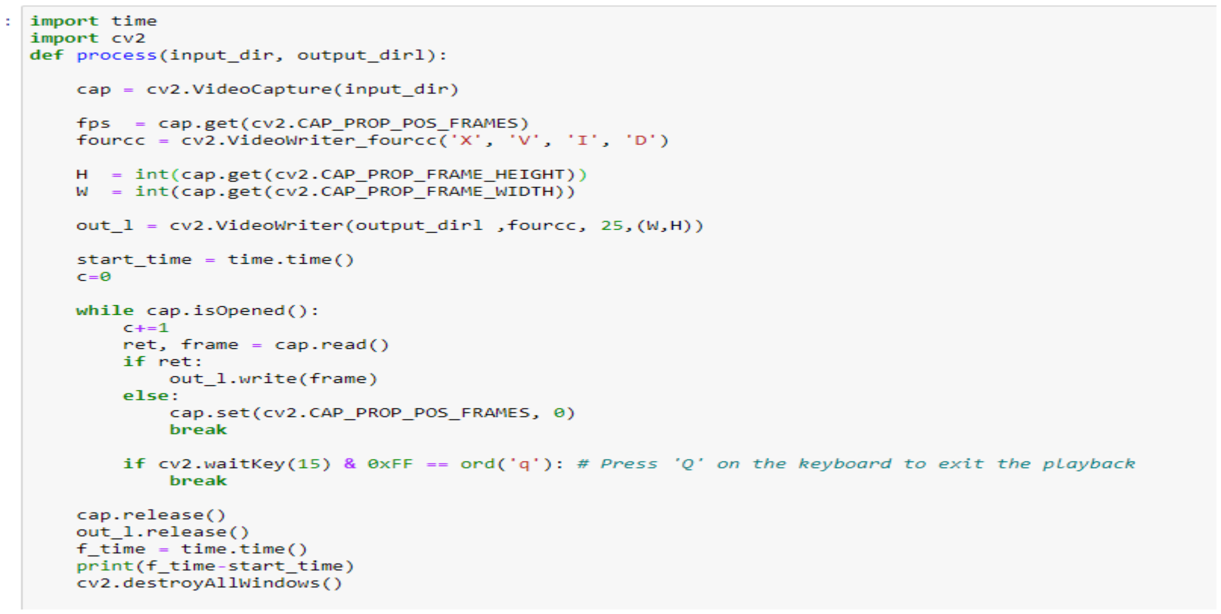
- Most Frequent Emotion : The speech sample was processed to depict the most frequent emotion. Each speech sample was split into 25ms chunks and a speech emotion recognition model was deployed to predict the emotion in that chunk. On completion of the entire sample, the most frequent emotion predicted was taken. (CNN based calculated feature)

4 Data Pre-Processing

This section comprises of various methods deployed for data pre-processing, extraction, and exploratory data analysis.

4.1 Data Preparation

The eye-tracked videos were fed into BeGaze software and extracted. These extracted videos were sampled at 1000 frames per second. To convert it to 25 frames per second, a python script was deployed, as shown in Figure 2. The name of the python file containing the code for extraction is *convert_25fps.py*. The obtained file is in Mp4 format.



```
import time
import cv2
def process(input_dir, output_dirl):

    cap = cv2.VideoCapture(input_dir)

    fps   = cap.get(cv2.CAP_PROP_POS_FRAMES)
    fourcc = cv2.VideoWriter_fourcc('X', 'V', 'I', 'D')

    H   = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
    W   = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))

    out_l = cv2.VideoWriter(output_dirl ,fourcc, 25,(W,H))

    start_time = time.time()
    c=0

    while cap.isOpened():
        c+=1
        ret, frame = cap.read()
        if ret:
            out_l.write(frame)
        else:
            cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
            break

        if cv2.waitKey(15) & 0xFF == ord('q'): # Press 'Q' on the keyboard to exit the playback
            break

    cap.release()
    out_l.release()
    f_time = time.time()
    print(f_time-start_time)
    cv2.destroyAllWindows()
```

Figure 1: Script to convert 1000fps to 50fps

The speech samples recorded were all under 1 minute and hence, a maximum buffer of 60 seconds was created to store the speech sample during analysis.

4.2 Data Extraction

The feature extraction methodologies for this research is discussed below. For the extraction of the heatmap of point of gaze of the eye, two YOLOv5 models were deployed, one fore determining the point of gaze circle and the other for detecting the laptop screen coordinates. First, the YOLOv5's folder present on Github, hosted by Ultralytics was cloned on the local system. The data required to train this model was acquired from taking screenshots of the frames of raw data present (visual stimuli). From the raw data (without eye-gaze), the coordinates of the laptop screen were determined, and for the gaze data, the 1000fps data was leveraged. A total of 124 images were used to train the YOLO model for circle detection, and a total of 215 images were used to train the YOLO model for the laptop screen detection. These data points were annotated using the *labelimg* software, present in the open source domain. The trained YOLO model would provide weights in the form of a*.pt* format. The training was simultaneously performed for a total of 300 epochs with a batch size of 8.

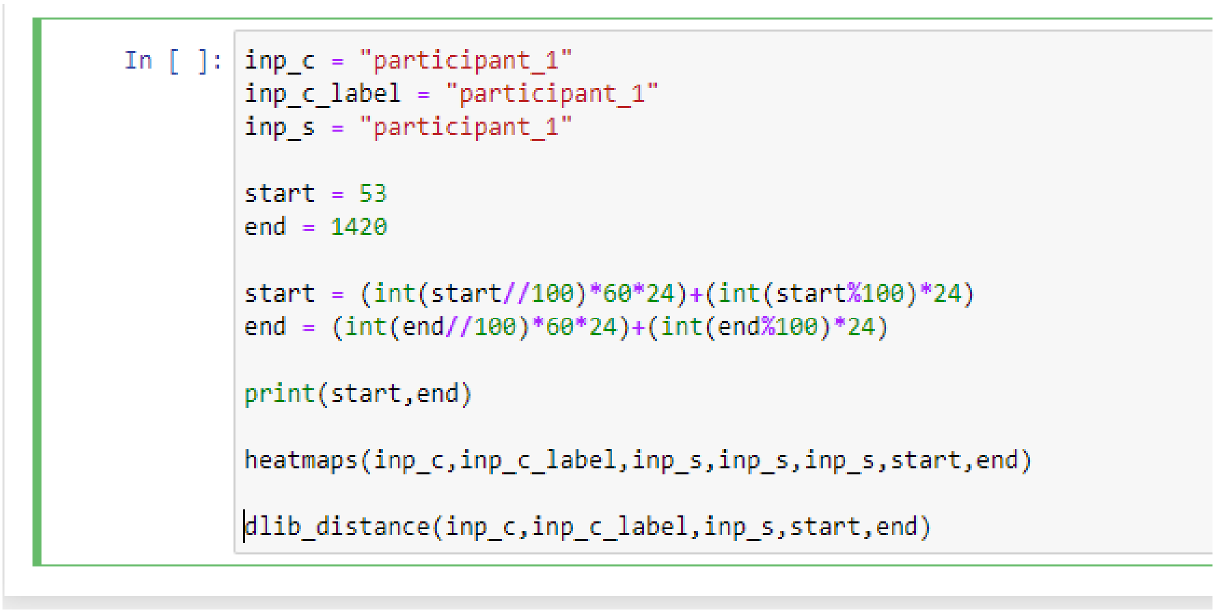
A python command which was :

python detect.py –weights
path to model weights
best circle weight.pt –img 640 –conf 0.25 –source
path to video
participant 1.avi –save-txt –nosave –save-conf –name participant 1 –project circle or screen/

was passed to detect the point of circle or screen. The output of this command would be the generation of a text file in the same folder as the sample (unless explicitly changed) with the coordinates of the circle or screen for each frame of the video.

Next, the generation of the average distance from the eye. To achieve this, the dlib library was used. This library provides a coordinates of the face of the actors narrating the story on screen. A jupyter notebook was created which contains the script to generate the data points. The name of the file is *"face_data_generation.ipynb"*. The weights for running this model is in the file called as *"shape_predictor_68_face_landmarks.dat"*. The obtained results is in the form of a comma seperated file (csv) comprising of the coordinates for the eyes of the actors on screen.

Since, there exists multiple files, a single jupyter notebook was created to call of these commands in one file. The only dependency is to ensure the weights and other files for the models are present in the respective directories.



```
In [ ]: inp_c = "participant_1"
        inp_c_label = "participant_1"
        inp_s = "participant_1"

        start = 53
        end = 1420

        start = (int(start//100)*60*24)+(int(start%100)*24)
        end = (int(end//100)*60*24)+(int(end%100)*24)

        print(start,end)

        heatmaps(inp_c,inp_c_label,inp_s,inp_s,inp_s,start,end)

        dlib_distance(inp_c,inp_c_label,inp_s,start,end)
```

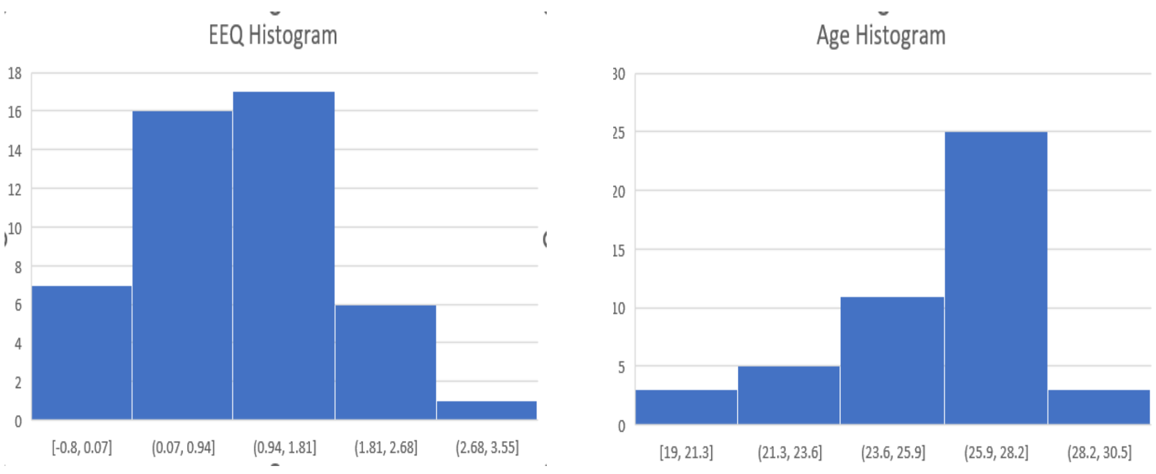
Figure 2: Script to Generate HeatMaps

A Speech Emotion Recognition (SER)-based model was procured from Github, which was used to derive the Most Frequent Emotion for the entire speech sample. The commands used to derive the speech emotion are :

python model_inference.py
path to the pre-trained model
option for how many features to extract (3,5,7)
path to audio file

4.3 Exploratory Data Analysis

The exploratory data analysis performed indicated certain attributes of the data such as the number of female participants were 14, the number of male participants were 36, the total range of their ages were between 19 to 30 years, of which majorly, the participants were between 25-28 years of age. The average level of sadness for participants before watching the videos was 2.5, while the level of sadness after watching increased to 4.06, indicating that on an average the participants felt sad after watching the videos.The average score secured for the memory test was 8 out of 10.



(a) Empathy-Score distribution (b) Distribution of Ages

Figure 3: Exploratory Data Analysis

5 Implementation

The final approach was to implement 3 different machine-learning models. All of these model contained hyper-parameter tuning, randomized search and a 3-fold cross-validation parameters. The 3 model deployed were Random Forest, Gradient Boosting, and Logistic Regression. The research was conducted with 3 different machine learning models used. All models were built using 3-fold cross validation and hyper-parameters were tuned using randomized-search. Logistic Regression, Random Forest and Gradient Boosting were used as part of this research.

Principal Component Analysis was performed on the heatmap data to depict the correlated and necessary features to be used for the final prediction. The results showed that 34 featuress of all the flattened pixels explained about 95% of the variance for the heatmap, as shown in Fig 4 PCA was done to reduce the number of features from the heatmap. it showed that 34 features from heatmap flattened pixels explain 95% of the variance as shown in Fig.

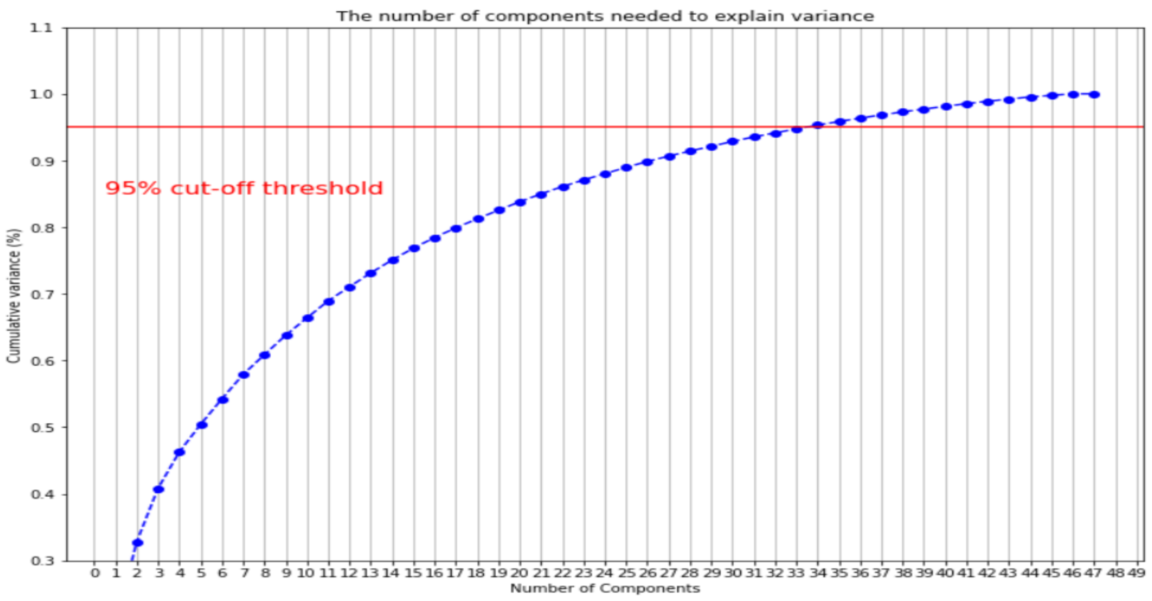
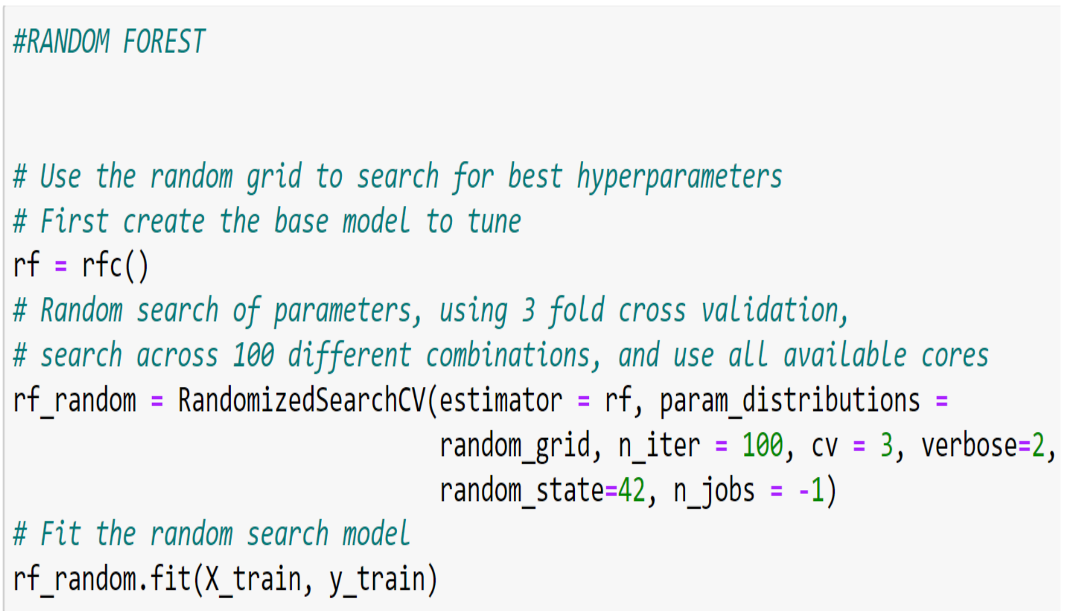


Figure 4: PCA Threshold for HeatMap features

5.1 Model 1 : Random Forest

A Random Forest model was built with hyper-parameter tuning, and cross-validation. The cross-validation was kept to 3-fold, the hyper-parameters that were tuned were n-estimators, maximum-depth, minimum-samples-leaf, minimum-samples-split. The model was built using Random Search.



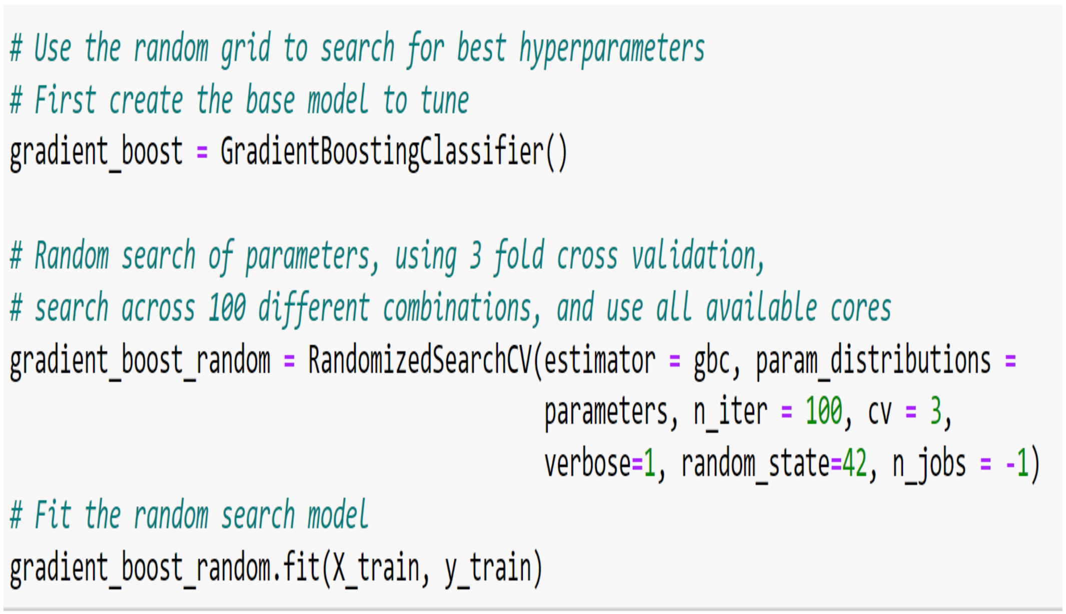
```
#RANDOM FOREST


# Use the random grid to search for best hyperparameters
# First create the base model to tune
rf = rfc()
# Random search of parameters, using 3 fold cross validation,
# search across 100 different combinations, and use all available cores
rf_random = RandomizedSearchCV(estimator = rf, param_distributions =
                               random_grid, n_iter = 100, cv = 3, verbose=2,
                               random_state=42, n_jobs = -1)
# Fit the random search model
rf_random.fit(X_train, y_train)
```

Figure 5: Code to initiate Random Forest mode

5.2 Model 2 : Gradient Boosting

A Gradient Boosting model was built with hyper-parameter tuning, and cross-validation. The cross-validation was kept to 3-fold, the hyper-parameters that were tuned were n-estimators, maximum-depth, minimum-samples-leaf, minimum-samples-split, and learning rate . The model was built using Random Search.



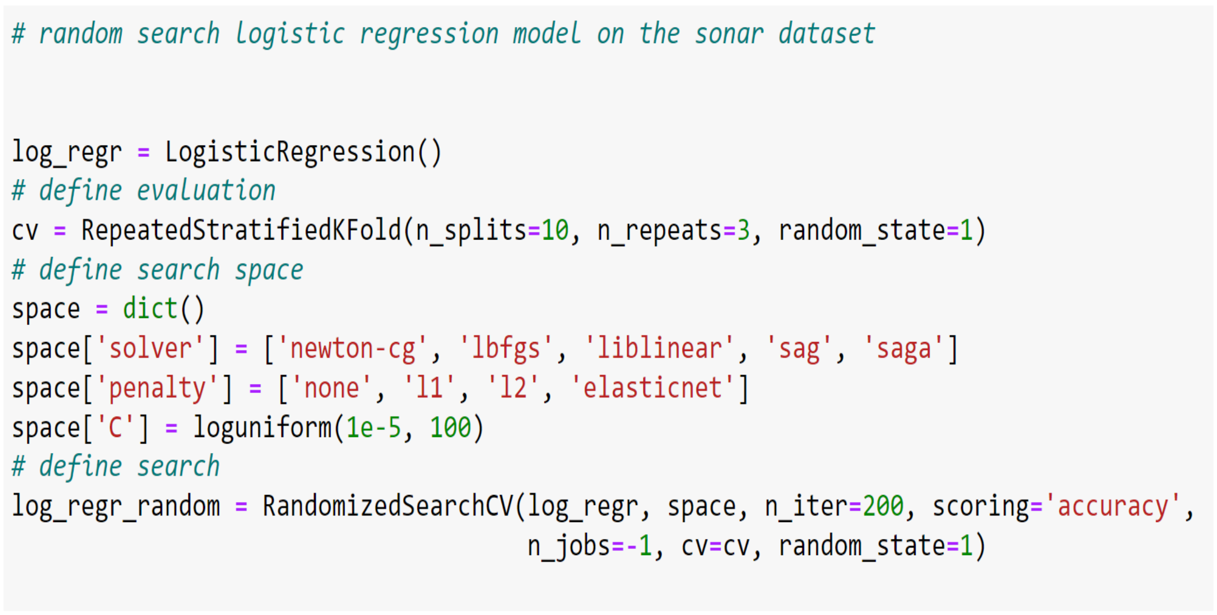
```python
# Use the random grid to search for best hyperparameters
# First create the base model to tune
gradient_boost = GradientBoostingClassifier()

# Random search of parameters, using 3 fold cross validation,
# search across 100 different combinations, and use all available cores
gradient_boost_random = RandomizedSearchCV(estimator = gbc, param_distributions =
                                            parameters, n_iter = 100, cv = 3,
                                            verbose=1, random_state=42, n_jobs = -1)
# Fit the random search model
gradient_boost_random.fit(X_train, y_train)
```

Figure 6: Code to initiate Gradient Boosting model

5.3 Model 3 : Logistic Regression

A Gradient Boosting model was built with hyper-parameter tuning, and cross-validation. The cross-validation was kept to 3-fold, the hyper-parameters that were tuned were *solver, penalty and C*. The model was built using Random Search.



```python
# random search logistic regression model on the sonar dataset


log_regr = LogisticRegression()
# define evaluation
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
# define search space
space = dict()
space['solver'] = ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']
space['penalty'] = ['none', 'l1', 'l2', 'elasticnet']
space['C'] = loguniform(1e-5, 100)
# define search
log_regr_random = RandomizedSearchCV(log_regr, space, n_iter=200, scoring='accuracy',
                                     n_jobs=-1, cv=cv, random_state=1)
```

Figure 7: Code to initiate Logistic Regression

6 Conclusion

In conclusion, this report contains all the information pertaining to the procedures followed in this research project. This ensures that the code is reusable. Everything performed in this project is in the open-source domain, that can be reproduced. This report is structured by the phases in which the research ensuring that the information is in sequential format.