‘Arrhythmia’ – Heart Disease Detection

A

Summer Training Report

Submitted

In partial fulfilment

For the award of the Degree of

Bachelor of Technology

In Department of Computer Science & Engineering

(With specialization in Computer Science & Engineering)

Submitted to:

Mr. Ankit Kumar

(Associate Professor)

(Department of CSE)

Submitted By:

Rajat Saini

16ESKCS133

Branch: CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SWAMI KESHVANAND INSTITUTE OF TECHNOLOGY, MANAGEMENT & GRAMOTHAN, JAIPUR

RAJASTHAN TECHNICAL UNIVERSITY, KOTA

2019-20

CERTIFICATE

This is to certify that **Mr. Rajat Saini**, a student of B.Tech, (Computer Science and Engineering) **VII semester** has submitted his Practical Summer Training Report entitled **"Arrhythmia – Heart Disease Detection"** under my guidance.

Mr. Ankit Kumar

Associate Professor

(Department of Computer Science & Engineering)

MNIT Innovation and Incubation Centre (MIIC)
(A Technology Business Incubators established by GOI)
Malaviya National Institute of Technology Jaipur

NO-MIIC/MNIT/2019/156

Dt- 29-7-19

TO WHOM SO EVER IT MAY CONCERN

This is to certify that **Mr. Rajat Saini,** IIIrd yr. student of **Swami Keshvanand Institue of Technology** , successfully completed his **Summer Internship Training** with **Bionics and Intelligence Lab**, a company registered at MNIT Innovation And Incubation Centre (MIIC), Jaipur.

The duration of training was 60 days starting from 20/05/2019 to 18/07/2019.

His performance during the training was good and he showed apt interest and keenness to understand the theoretical and practical aspects of the project.

We wish him all the very best for his future endeavors.

23/7/19

Head-MIIC, Jaipur

Head
MNIT Innovation and Incubation Centre
Malaviya National Institute of Technology
Jaipur-302017 (Raj.)

Address: MNIT Innovation and Incubation Centre, (MIIC), MNIT Campus, JLN Marg, Jaipur-302017
Phone: 0141-2529066. Email: miic@mnit.ac.in

Bionics and Intelligence Lab

27th July, 2019

To whomsoever it may Concern

This is an **Internship Completion Certificate** for Mr. Rajat Saini, B-Tech III year from Swami Keshvanand Institute of Technology Management & Gramothan, Jaipur.

We hereby state that Mr. Rajat Saini has successfully completed an internship project in the role of Applied Machine Learning for Intelligent System Design intern at Bionics and Intelligence Lab, MIIC (MNIT Innovation and Incubation center), Jaipur. The internship start date was 20th May and end date was 18th July at MIIC, Jaipur.

During this period, Mr. Rajat Saini worked on various areas of machine learning domain including data visualization, feature engineering, classification, clustering, and regression analysis. Further, Mr. Saini executed the above task with utmost diligence and punctuality.

I wish him the very best for his future.

Feel free to contact me in-case you have questions about his tenure with Bionics and Intelligence lab.

Sidharth Pancholi

Sidharth Pancholi

Proprietor Bionics and Intelligence Lab

MIIC, MNIT, Jaipur

2016rec9543@mnit.ac.in

+91-8764012630

For Bionics and Intelligence Lab

Proprietor

Address: Bionics and Intelligence Lab, MNIT Innovation and Incubation Center, JLN marg, Jaipur 302017, Rajasthan.

CANDIDATE'S DECLARATION

I hereby declare that the work which is being presented in the Seminar Report, entitled **'Arrhythmia' – Heart Disease Detection** in partial fulfilment for the Degree of Bachelor of Technology in Department of Computer Science and Engineering submitted to the Swami Keshvanand Institute of Technology, Management and Gramothan, Jaipur, Rajasthan Technical University is a record of my own investigations carried under the guidance of **Mr. Ankit Kumar**, Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management and Gramothan, Jaipur.

I have not submitted the matter present in this report elsewhere for the award of any other degree.

Rajat Saini

16ESKCS133

ACKNOWLEDGMENT

I feel immense pleasure in expressing my regards to the Chairman **Mr. Surja Ram Meel,** Director **Mr. Jaipal Meel,** Registrar **Mrs. Rachana Meel,** Director (Academics) **Prof.** (**Dr.) S.L. Surana,** Principal **Prof. (Dr.) Ramesh Pachar,** Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur for providing me necessary facilities during the various stages of this work.

I would like to thank **Dr. Mukesh Gupta,** Professor & Head, Department of Computer Science & Engineering for providing me opportunity to work in a consistent direction and providing their valuable suggestions to improve Seminar Report.

I would like to thank my Project Training Seminar Report head **Mr. Ankit Kumar, Associate Professor**, Department of Computer Science & Engineering, Swami Keshvanand Institute of Technology, Management & Gramothan, Jaipur for his valuable guidance, constant encouragement, and support throughout this work. Especially I acknowledge his support when I was stuck and he suggested me new ideas to solve the problems. Also, I want to thank **MNIT Innovation and Incubation Centre, Jaipur** for giving me the opportunity to undertake my summer training at their esteemed organization.

I would also like to express my thanks to my parents for their support and blessings. In addition, a very special thanks to all my colleagues and friends for their support in the completion of this work.

Rajat Saini

16ESKCS133

CONTENTS

ABSTRACT

An arrhythmia is a problem with the rate or rhythm of the heartbeat. It means that your heart beats too quickly, too slowly, or with an irregular pattern. Fetal arrhythmia is a term that refers to any abnormality in heart rate of a baby. These can include tachycardia–an increased heart rate–or bradycardia, which is a slowed heartbeat. The normal heart rate for a fetus is anywhere between 120 and 160 beats per minute. But these abnormalities in ECG signal can be found by trained physician only, manually detection of these abnormalities is hard.

So, the analysis of fetal electrocardiogram has an important role during pregnancy, because it provides important information about the condition of the fetus. A continuous monitoring of fECG may increase the early detection of diseases like fetal arrhythmia. But filtering ECG signal obtained from the abdomen of the mother is not an easy task. Because the signal obtained from the abdominal surface of the mother is a collection of noises. It dwells noise generated by fetal brain activity, myographic signals and maternal ECG, etc. The neurokit library is used to preprocess and extract the fetal ECG. The classification is done by using Random Forest. Most of the data used for this study were extracted from the Physionet database. Validation of the proposed extraction and detection framework has been done on publicly available datasets, showing promising results (accuracy = 89.47%).

Chapter 1
Introduction

1.1 OVERVIEW

Machine Learning is the discipline of Artificial Intelligence which is the simulation of human intelligence by computer systems. It is the combination of computer science and statistics. Computer Science mainly focuses on solving the problems and identifying whether the problems are solvable at all stages. The idea of statistics is data modelling, hypothesis and measuring the reliability. Machine learning is a paradigm that learn from past experience for the improvement of future performance. The main aim of this field is automatic learning methodologies. Learning means the modification or improvement to the algorithm based on previous experiences without any involvement of human. ML mainly focuses on the development of the programs that uses data to learn themselves. Machine Learning provides algorithms and tools to make the system work intelligently. It is mainly used for the problems without deterministic solution where there is no specific models for the problem. Algorithms are developed based on diverse disciplines and it is used mainly for accuracy, speed and customizability. Machine Learning also contributes in medical diagnosis for disease prediction, data analysis, therapy planning, etc.

1.2 MOTIVATION

The disease detection we need a large volume of health data associated with each patient. So, Machine Learning plays the best role in predicting a diagnosis of a disease. Since it can handle any type of data associated with a patient like such as X-ray results, vaccinations, blood samples, vital signs, DNA sequences, current medications, past medical history and much more. Furthermore,

ML mainly focuses on the development of the program that uses data to learn themselves. Machine Learning provides algorithms and tools to make the system work intelligently. Machine Learning provide a variety of applications for medical diagnosis like disease prediction, data analysis, therapy planning.

1.3 OBJECTIVES OF TRAINING

Training provide comprehensive learning platform for students where they can enhance their employability skills and become job ready along with real corporate exposure. It enhances students' knowledge in one particular technology. The purpose of this training is to learn various areas of machine learning domain including data visualization, feature engineering, classification, clustering, and regression analysis. This knowledge of machine learning is used to discover patterns in data and then make predictions based on often complex patterns to answer business questions, detect and analyse trends and help solve problems.

Chapter 2
Introduction to Project

2.1 OVERVIEW OF PROJECT/MODULES

The arrhythmia detection project is based on Machine Learning, whose main goal is to detect fetal arrhythmia in patients. The technical overview of our project is-

1. Anaconda Navigator (Jupyter Notebook)- Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda® distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. It is available for Windows, macOS, and Linux.The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

2. Python- It is an interpreted, high-level, general-purpose programming language. It is considered to be in the first place in the list of all AI development languages due to the simplicity. The syntaxes belonging to python are very simple and can be easily learnt. Therefore, many AI algorithms can be easily implemented in it. Python takes short development time in comparison to other languages like Java, C++ or Ruby. Python supports object oriented, functional as well as procedure oriented styles of programming. There are plenty of libraries in python, which make our tasks easier. For example: Numpy is a library for python that helps us to solve many scientific computations. Also, we have Pybrain, which is for using machine learning in Python.

3. Neurokit library- It is most important library which is required for processing of biosignals. It is a Python module that provides high-level integrative functions with good and flexible defaults, allowing users to focus on what's important.

4. Matplotlib library- It is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc.with just a few lines of code.

5. Scikit-learn library- Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language.[3] It features various classification, regression and clustering algorithms including support

vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

6. Numpy library- It stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. Using NumPy, a developer can perform the following operations –
   1) Mathematical and logical operations on arrays.
   2) Fourier transforms and routines for shape manipulation.
   3) Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.

7. Pandas library- It is a software library written for Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. Features-
   1) DataFrame object for data manipulation with integrated indexing.
   2) Tools for reading and writing data between in-memory data structures and different file formats.
   3) Data alignment and integrated handling of missing data.
   4) Data structure column insertion and deletion.
   5) Data set merging and joining.

2.2 METHODOLOGY OF PROJECT

The structure of fetal arrhythmia detection and classification in my project consists of three main phases: Preprocessing, ECG feature extraction and classification. The proposed block diagram is shown in figure:

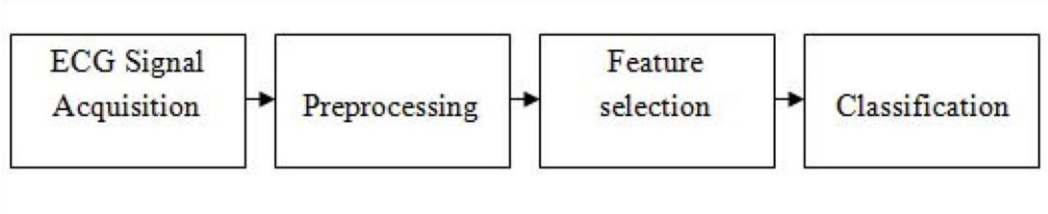


Figure 1: Block diagram

1. ECG Signal Acquisition

This phase consists of a collection of one-minute fetal ECG recordings. Each recording includes fetal ECG obtained from the abdomen of the mother. The sampling frequency was 500 Hz or 1 kHz. The Non-Invasive Fetal ECG Arrhythmia Database (NIFEA DB) provides a series of fetal arrhythmias recordings (n=12) and a number of control normal rhythm recordings (n=14) performed using the non-invasive fetal electrocardiography (NI-FECG) technique. Recordings are named with the following convention: i) ARR: arrhythmia fetus. ii) NR: normal rhythm fetus.

2. Preprocessing

The noisy raw input collected from mother is mixed with respiratory as well as muscular signals are denoised using neurokit library. The classification of arrhythmia is done using random forest. A Preprocessing shows the input raw ECG signal obtained from the abdominal surface of the mother.

Since it consists of so many noises like respiratory as well as muscle contraction noises, it needs filtering for accurately analyzing fetal ECG to find arrhythmia.

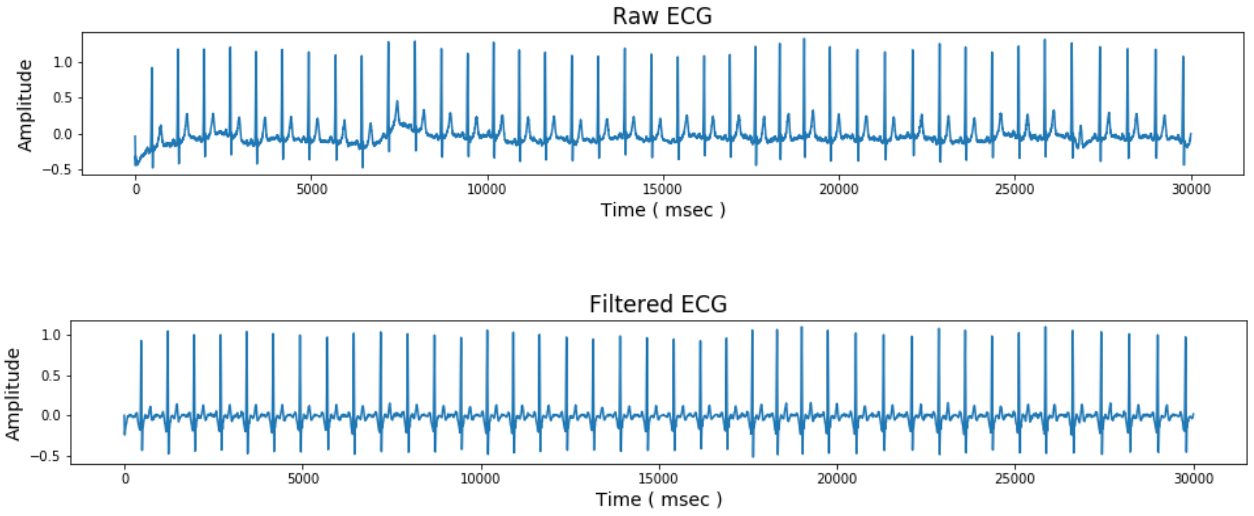


Figure 3: Raw and Filtered ECG Signals

3. Feature Selection

Biomedical signal means a collective electrical signal acquired from any organ that represents a physical variable of interest. FECG is a biomedical signal that gives electrical representation of FHR. Sometimes the FECG is the only information source in early stage diagnostic of fetal health. Because one important factor is that, heart is one of the first organs formed in the very early stages of pregnancy. The fetal ECG consists of mainly three parts:

i) The P-wave: which reflects the contraction of the atriales.ie, The first segment is P-wave which indicates the depolarized wave that distributes from the SA node to the atria

ii) The QRS-complex is associated with the contraction of the ventricles. The QRS complex is a combination of three of the graphical deflections in the ECG waveforms. It is one of the most important segments of ECG waveforms, which represents ventricular depolarization.

iii) The T-wave- It is the last part which indicates ventricular repolarization and its time is larger than depolarization.

The feature selection of the ECG is done by analyzing certain parameters; these are the RR interval, the PQ interval, the QT interval, the QRS interval, the P amplitude, the R amplitude and the T amplitude etc. The characteristic points P, Q, R, S and T peaks are detected using neurokit library. Important features selected for classification are- R-R

interval, P-amplitude, P-R interval, Q-R interval, P-amplitude, QRS complex duration etc. The features extracted for classification consist of both domain time domain and amplitude. These features gives more accurate results for classification of fetal arrhythmia because the fetal arrhythmia is completely dependent on these parameters.

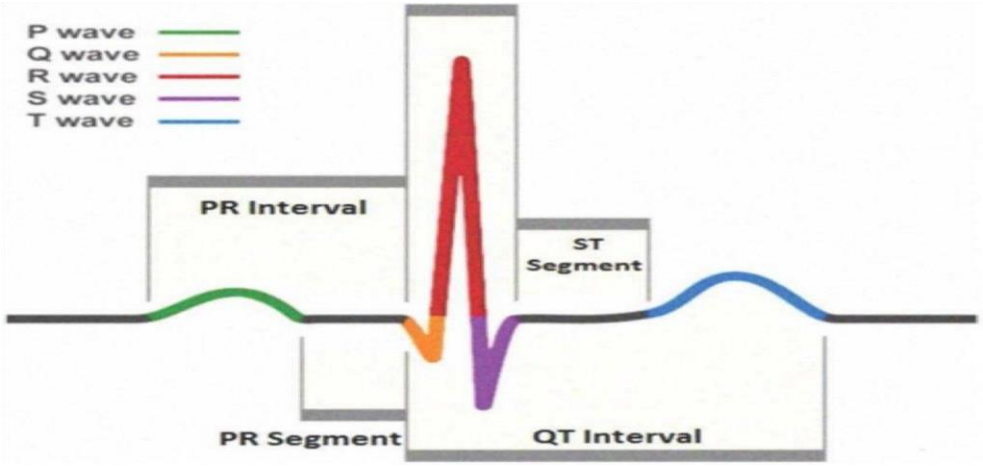


Figure 3: The ECG Waveform

4. Classification

Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y). For example, spam detection in email service providers can be identified as a classification problem. This is s binary classification since there are only 2 classes as spam and not spam. A classifier utilizes some training data to understand how given input variables relate to the class. Classification belongs to the category of supervised learning where the targets also provided with the input data. There are many applications in classification in many domains such as in credit approval, medical diagnosis, target marketing etc. It is a supervised **learning** approach in which the computer program learns from the data input given to it and then uses this learning to classify new observations. This phase involves two main steps Training and Testing. In Machine Learning, we basically try to create a model to predict on the test data. So, we use the training data to fit the model and testing data to test it. The models generated are to predict the results unknown which is named as the test set. As pointed out, the dataset is divided into train and test set in order to check accuracy, precision by training and testing it on it.

The proportion to be divided is completely up to you and the task you face. It is not essential that 70% of the data has to be for training and rest for testing. It completely depends on the dataset being used and the task to be accomplished. This is mostly because

in Machine Learning, the bigger the dataset to train is better. The three sets in which dataset is divided:

1. **Training Set**: Here, you have the complete training dataset. You can extract features and train to fit a model and so on.
2. **Validation Set**: This is crucial to choose the right parameters for your estimator. We can divide the training set into train set and validation set. Based on the validation test results, the model can be trained (for instance, changing parameters, classifiers). This will help us get the most optimized model.

3. **Testing Set**: Here, once the model is obtained, you can predict using the model obtained on the training set.

For training of dataset there are several Machine learning algorithms that can be used. Here, we trained model using following algorithms- 1) k-nearest neighbours 2) Decision Tree classifier 3) Random Forest Classifier.

1. k-nearest neighbours classifier-

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. To evaluate any technique we generally look at 3 important aspects:

1. Ease to interpret output
2. Calculation time
3. Predictive Power

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor. The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

2. Decision Tree Classifier-

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control

statements. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods. Tree based methods empower predictive models with high accuracy, stability and ease of interpretation. Unlike linear models, they map nonlinear relationships quite well. They are adaptable at solving any kind of problem at hand (classification or regression).

Root Node: It represents entire population or sample and this further gets divided into two or more homogeneous sets.

1. **Splitting:** It is a process of dividing a node into two or more sub-nodes.
2. **Decision Node:** When a sub-node splits into further sub-nodes, then it is called decision node.
3. **Leaf/ Terminal Node:** Nodes do not split is called Leaf or Terminal node.
4. **Pruning:** When we remove sub-nodes of a decision node, this process is called pruning. You can say opposite process of splitting.
5. **Branch / Sub-Tree:** A subsection of entire tree is called branch or sub-tree.
6. **Parent and Child Node:** A node, which is divided into sub-nodes is called parent node of sub-nodes whereas sub-nodes are the child of parent node.

3. Random Forest Classifier

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds. In data science speak, the reason that the random forest model works so well is: A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models. The low correlation between models is the key. Just like how investments with low correlations (like stocks and bonds) come together to form a portfolio that is greater than the sum of its parts, uncorrelated models can produce ensemble predictions that are more accurate than any of the individual predictions. The reason for this wonderful effect is that the trees protect each other from their individual errors (as long as they don't constantly all err in the same direction).

While some trees may be wrong, many other trees will be right, so as a group the trees are able to move in the correct direction. So the prerequisites for random forest to perform well are: There needs to be some actual signal in our features so that models built using those features do better than random guessing and The predictions (and therefore the errors) made by the individual trees need to have low correlations with each other. Random forests overcome several problems with decision trees, including:

- Reduction in overfitting: By averaging several trees, there is a significantly lower risk of overfitting.

- Less variance: By using multiple trees, you reduce the chance of stumbling across a classifier that doesn't perform well because of the relationship between the train and test data. As a consequence, in almost all cases, random forests are more accurate than decision trees.

- Random Forest can automatically handle **missing values**.

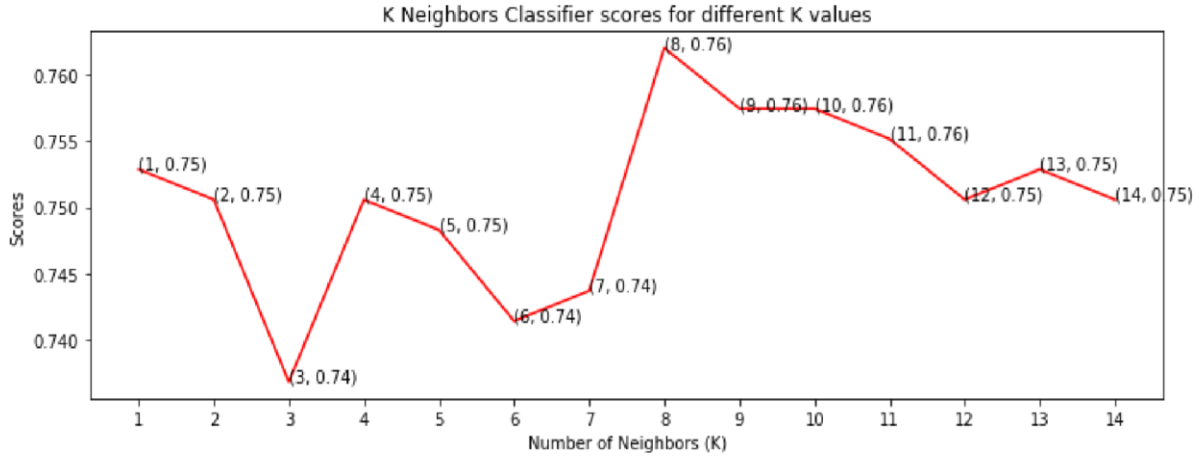
- **No feature scaling required:** No feature scaling (standardization and normalization) required in case of Random Forest as it uses rule based approach instead of distance calculation.

Chapter 3
Assignments/Results

3.1 RESULTS ACHIEVED

1. k-Nearest Neighbour Classifier

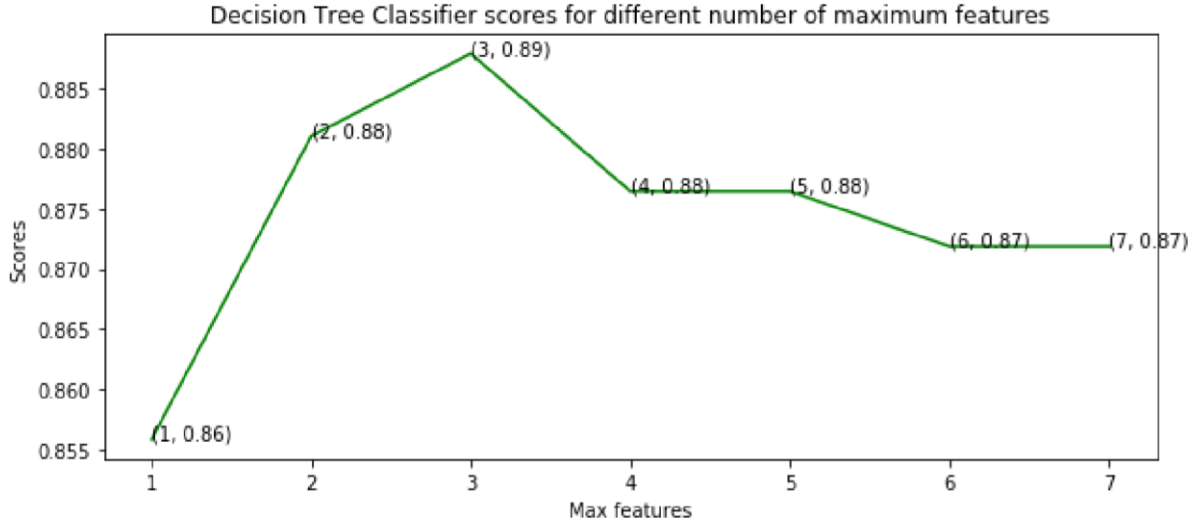
A line graph has been plotted of the number of neighbours and the test scores achieved in each case.



The maximum score 76% has been achieved when number of neighbours is 8.

2. Decision Tree Classifier

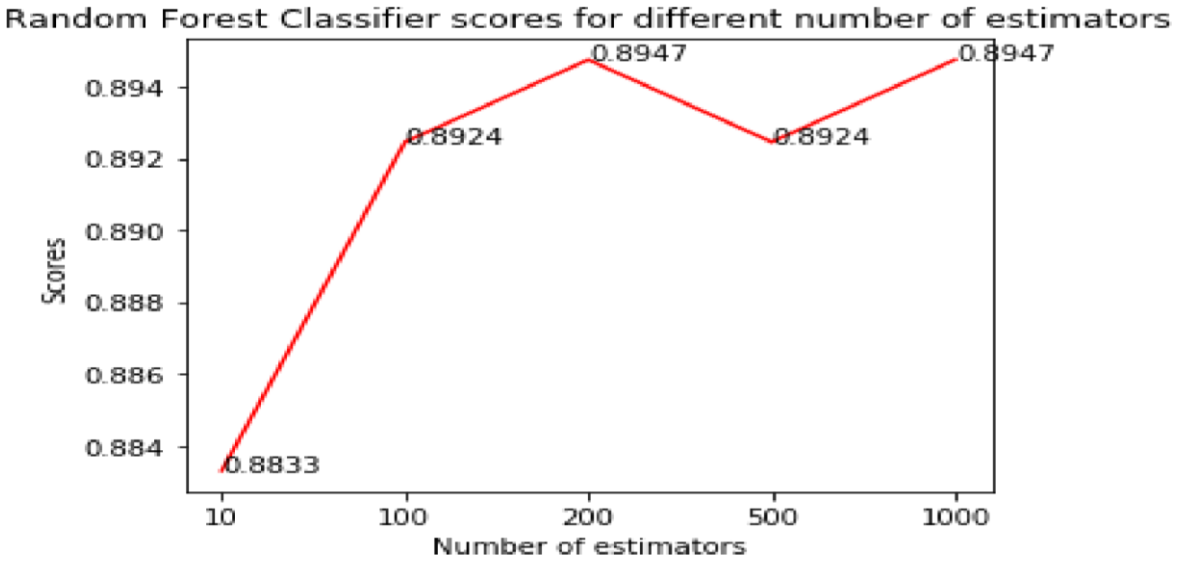
A line graph has been plotted to see the effect of the number of features on the model scores.



From the above line graph, we can clearly see that the maximum score is 89% and is achieved for maximum features being selected.

3. **Random Forest Classifier**

A line graph has been plotted to see the effect of the number of estimators on the model scores.



Taking a look at the bar graph, we can see that the maximum score of 89.47% was achieved for both 200 and 1000 trees.

3.2 SCREENSHOTS OF CODE & RESULTS

```
In [1]: import neurokit as nk
        import pandas as pd
        import numpy as np
        import os
        import matplotlib.pyplot as plt
        from sklearn.model_selection import train_test_split
        import seaborn as sns
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.svm import SVC
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.ensemble import RandomForestClassifier
```

Figure 4. : Importing libraries

```
In [5]: print(len(df.index))
        df=df.dropna()
        print(len(df.index))
        df.head()
```

```
2210
2184
```

Out[5]:

	QRS_interval	PQ_interval	QT_interval	RR_interval	P_amplitude	R_amplitude	T_amplitude	class
1	57	94	288	742.0	0.029543	0.999226	0.126534	1
2	58	57	292	746.0	-0.052132	0.999169	0.131391	1
3	57	49	292	730.0	-0.096451	1.036440	0.130586	1
4	57	103	286	742.0	0.028471	1.012081	0.130808	1
5	56	45	290	754.0	-0.101115	0.994287	0.119405	1

Figure 5: Data frame with class labelled attributes

```
In [6]: df.corr()
```

Out[6]:

	QRS_interval	PQ_interval	QT_interval	RR_interval	P_amplitude	R_amplitude	T_amplitude	class
QRS_interval	1.000000	-0.015112	0.275003	0.126350	-0.023560	0.080882	0.033371	-0.224261
PQ_interval	-0.015112	1.000000	0.287249	0.435740	-0.150959	-0.063386	0.093899	-0.123550
QT_interval	0.275003	0.287249	1.000000	0.600814	-0.224378	-0.102746	-0.049305	-0.037195
RR_interval	0.126350	0.435740	0.600814	1.000000	-0.079844	-0.119021	-0.010191	-0.171625
P_amplitude	-0.023560	-0.150959	-0.224378	-0.079844	1.000000	0.075516	0.007732	0.070257
R_amplitude	0.080882	-0.063386	-0.102746	-0.119021	0.075516	1.000000	0.539064	0.014014
T_amplitude	0.033371	0.093899	-0.049305	-0.010191	0.007732	0.539064	1.000000	0.052970
class	-0.224261	-0.123550	-0.037195	-0.171625	0.070257	0.014014	0.052970	1.000000

Figure 6: Correlation among all columns

KNN Classifier

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
knn_scores = []
for k in range(1,15):
    knn_classifier = KNeighborsClassifier(n_neighbors = k)
    knn_classifier.fit(x_train, y_train)
    knn_scores.append(knn_classifier.score(x_test, y_test))
figure= plt.subplots(figsize=(12, 4))
plt.plot([k for k in range(1, 15)], knn_scores, color = 'red')
for i in range(1,15):
    plt.text(i, knn_scores[i-1], (i, round(knn_scores[i-1],2)))

plt.xticks([i for i in range(1, 15)])
plt.xlabel('Number of Neighbors (K)')
plt.ylabel('Scores')
plt.title('K Neighbors Classifier scores for different K values')
```

```
Text(0.5, 1.0, 'K Neighbors Classifier scores for different K values')
```

Figure 7: KNN Classifier

Decision Tree

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
dt_scores = []
for i in range(1, len(x.columns) + 1):
    dt_classifier = DecisionTreeClassifier(max_features = i, random_state = 0)
    dt_classifier.fit(x_train, y_train)
    dt_scores.append(dt_classifier.score(x_test, y_test))
figure= plt.subplots(figsize=(10, 4))
plt.plot([i for i in range(1, len(x.columns) + 1)], dt_scores, color = 'green')
for i in range(1, len(x.columns) + 1):
    plt.text(i, dt_scores[i-1], (i,round(dt_scores[i-1],2)))
plt.xticks([i for i in range(1, len(x.columns) + 1)])
plt.xlabel('Max features')
plt.ylabel('Scores')
plt.title('Decision Tree Classifier scores for different number of maximum features')
```

```
Text(0.5, 1.0, 'Decision Tree Classifier scores for different number of maximum features')
```

Figure 8: Decision Tree

Random Forest

```
from matplotlib.cm import rainbow
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
rf_scores = []
estimators = [10, 100, 200, 500, 1000]
for i in estimators:
    rf_classifier = RandomForestClassifier(n_estimators = i, random_state = 0)
    rf_classifier.fit(x_train, y_train)
    rf_scores.append(rf_classifier.score(x_test, y_test))
colors = rainbow(np.linspace(0, 1, len(estimators)))
figure= plt.subplots(figsize=(5, 4))
plt.plot([i for i in range(len(estimators))], rf_scores, color = 'red')
for i in range(len(estimators)):
    plt.text(i, rf_scores[i], round(rf_scores[i],4))
plt.xticks(ticks = [i for i in range(len(estimators))], labels = [str(estimator) for estimator in estimators])
plt.xlabel('Number of estimators')
plt.ylabel('Scores')
plt.title('Random Forest Classifier scores for different number of estimators')
```

```
Text(0.5, 1.0, 'Random Forest Classifier scores for different number of estimators')
```

Figure 9: Random Forest

```
figure= plt.subplots(figsize=(16, 2))
d1['ECG'].iloc[0:30000].plot()
plt.xlabel("Time ( msec )",fontsize=14)
plt.ylabel("Amplitude",fontsize=14)
plt.tick_params(axis='both', which='major', labelsize=10)
plt.tick_params(axis='both', which='minor', labelsize=8)
plt.title("Raw ECG",fontsize=17)
```

```
Text(0.5, 1.0, 'Raw ECG')
```

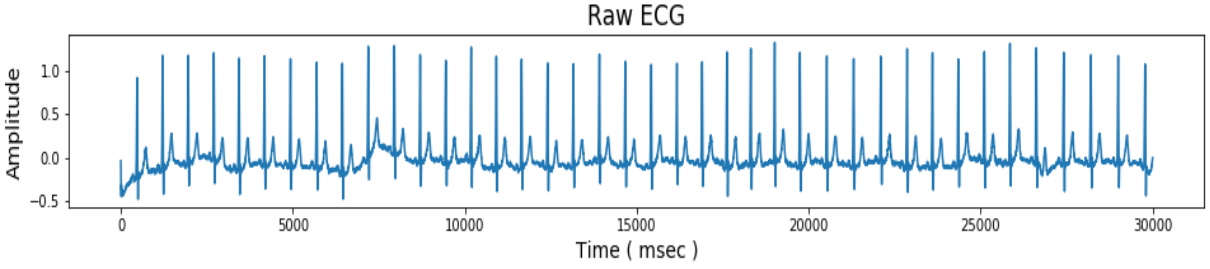


Figure 10: Raw ECG Signal

```
figure= plt.subplots(figsize=(16, 2))
ecg.iloc[0:30000].plot()
plt.xlabel("Time ( msec )",fontsize=14)
plt.ylabel("Amplitude",fontsize=14)
plt.tick_params(axis='both', which='major', labelsize=10)
plt.tick_params(axis='both', which='minor', labelsize=8)
plt.title("Filtered ECG",fontsize=17)
```

```
Text(0.5, 1.0, 'Filtered ECG')
```

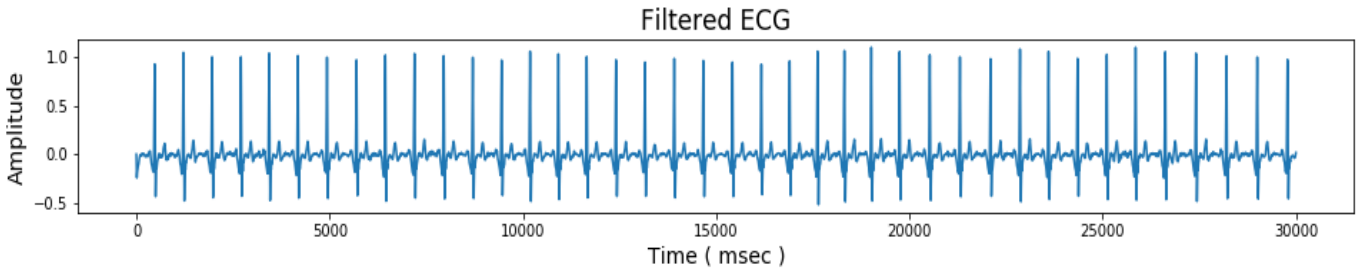


Figure 11: Filtered ECG Signal

Chapter 4
Conclusion

4.1 TAKEAWAYS OF TRAINING

Some of the takeaways of the training were:

- Insights regarding how solutions are formulated and structured for a given requirement while considering various approaches and adopting the most suitable one by weighing in various parameters and production level scenarios..
- Better approach and understanding of Data Analysis and Machine learning. Also, more acquainted with the software needed to build such machine learning algorithms.
- Helped in building a more impressive and strong resume for the future purpose while applying for companies.
- Worked on various areas of machine learning domain including data visualization, feature engineering, classification, clustering, and regression analysis.
- Exposure to neural networks and libraries like Tensor Flow.
- Learned to work in team while collaborating with my peers.

4.2 PROJECT CONCLUSION

The goal of this project is to detect fetal arrhythmia in an earlier stage of pregnancy from the raw ECG obtained from the abdominal surface of mother which is equipped by respiratory and muscular noises. The challenge is to extract ECG accurately from this mixed signal and to classify exactly. This preprocessing is effectively implemented by using Neurokit library of python and ecg_preprocess() function to determine different peaks in an ECG signal. The, 3 classifier models were trained and tested with maximum scores as follows:

1. K Neighbors Classifier: 76%
2. Decision Tree Classifier: 89%
3. Random Forest Classifier: 89.47%

Random Forest Classifier scored the best score of 89.47% with both 200 and 1000 estimators.