A

Major Project

On

STRESS DETECTION FOR WEARABLE DEVICES

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

CH. GEETHANJALI	(187R1A0570)
G. ANJANA	(187R1A0583)
R. ABHISHEK	(187R1A05A7)

Under the Guidance of

Mrs. D. SANDHYA RANI

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi)

Recognized Under Section 2(f) & 12(B) of the UGC Act.1956,

Kandlakoya (V), Medchal Road, Hyderabad-501401.

2018-2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR
GROUP OF INSTITUTIONS
EXPLORE TO INVENT

CERTIFICATE

This is to certify that the project entitled "**STRESS DETECTION FOR WEARABLE DEVICES**" being submitted by **CHALLA GEETHANJALI (187R1A0570) , GARREPELLI ANJANA (187R1A0583)** and **RATHOD ABHISHEK (187R1A05A7)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2021-22.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Mrs. D. Sandhya Rani
Assistant Professor
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
HoD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ________________________

ACKNOWLEGDEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **Mrs. D. Sandhya Rani,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work.The blessing, help and guidance given by her shall carryus a long way in the journey of life on which we areabout to embark.

We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. A. Uday Kiran, Mr. J. Narasimha Rao , Dr. T. S. Mastan Rao , Mrs. G. Latha ,Mr. A. Kiran Kumar,** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

CH. GEETHANJALI	**(187R1A0570)**
G. ANJANA	**(187R1A0583)**
R. ABHISHEK	**(187R1A05A7)**

ABSTRACT

In this project, Stress can be recognized by observing changes in physiological responses on the human body. Wearable sensors for stress detection are becoming more prominent in recent years due to their functionality and non-intrusive nature. By utilizing data from wearable sensors, we have developed a personalized stress detection system. Our system performs classification on stress level using multimodal data from wrist-worn device Empatica E4 wearable sensor. We implemented three different classification algorithms: Logistic Regression, Decision Tree, and Random Forest and used four-class classification conditions: baseline, stress, amusement, and meditation. By evaluating the performance of the system, we demonstrate that our system can perform the best and consistent personalized stress detection using T-POT classifier with the accuracy of 88%-99% on 15 subjects.

LIST OF FIGURES/TABLES

LIST OF SCREENSHOTS

TABLE OF CONTENTS

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

This project is titled as “Stress Detection using wearable devices”. This software provides facility to identify how the stress is detected . This project uses machine-learning methods to identify how the stress is detected. First, we use a T-pot classifer to train the dataset. Then we identify how the stress is detected.

1.2 PROJECT PURPOSE

This has been developed to facilitate the identification, retrieval of the items and information. System is built with manually exclusive features. In all cases system will specify object which are physical or on performance characteristics. They are used to give optimal distraction and other information. Data are used for identifying, accessing, storing and identifying fake accounts. The data ensures that only one value of the code with a single meaning is correctly applied to give entity or attribute as described in various ways.

1.3 PROJECT FEATURES

The main features of this project are that the designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user issatisfied with proposal.

2. SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.1 PROBLEM DEFINITION

In today’s fast-paced world, mental stress is very common. Stress can be caused due to situations or events that put pressure on mind and body of a person. Reaction to stress is different for everyone as the capacity of dealing with tough or demanding situations vary for person to person. Some situations may cause stress to one person, while no stress to one altogether. Also, all stress is not bad for health as it can make people more aware of things around them and keep them more cautious about dangers and focused on their goal. A stressor is an event which causes stress to an individual. Many people usually faces stress due to these stressors described

According to American Psychological Association (APA),there are mainly three types of stress which are acute stress, episodic acute stress and chronic stress. Acute stress is short term stress which is least damaging type as compared to the other two. It can be good sometimes as this helps body to deal with the situation. When acute stress occurs frequently then an individual is affected with episodic acute stress. Chronic stress is the most harmful type of stress, if left untreated over a long period of time can damage physical and mental health of a person. Chronic stress puts pressure on the body and mind for an extended period which can cause a range of symptoms and increase the risk of developing certain diseases.

To avoid health problems, people with high risk of getting stressed should be continuously monitored to detect any stress signs. Wearable sensors provide opportunities to monitor stress and can inform people about their stress level which can be useful in order to minimize stress balance before it results into serious health problems. Physical health and mental health are closely connected, hence monitoring and measuring of physiological and physical changes can be used for detecting human stress level. Stress can be detected using physical and physiological measures of body. Physical measures include pulse rate, skin temperature, humidity, Blood pressure and respiration rate whereas physiological measures can be heart rate, heart rate variability, skin conductance. These can be measured using wearable devices made from low-cost sensors although machine learning algorithms can be used to classify and predict stress level of an individual. In this paper, some previous approaches of automatic stress recognition systems who used sensors and machine learning are discussed in detail. In these, physiological data is extracted using some stressor tests on the people. Some common stressor tests includes arithmetic calculations, questionnaire, mental tasks and working out in gym. There are a diversity of machine learning algorithms which are appropriate for stress detection. Among them Support Vector Machines (SVM), Logistic regression, K-Nearest Neighbor, Decision tree and Random forest are most common. In this review, we summarize the various machine learning algorithms available in the literature that aim at detecting state of stress.

2.2 EXISTING SYSTEM

Nowadays, sensors plays a vital role in medical applications. These are generally used for detection and measurement of various diseases and its levels. Stress is usually recognized as one of the major factors leading to various health problems. Therefore, people with high risk of getting stressed should be continuously monitored for detection of any stress signs before it causes health problems. Advances in wearable sensors and mobile computing make it possible to record a variety of physical and physiological signals on a twenty-four hour basis which helps in detection of stress level. Mostly wearable sensor devices like smart band, Chest belts are used for data collection. Some researchers used hardware and software for collection of data through sensors and detection of stress level respectively. A Holster unit was used with LI-PO battery and PC USB Client software for detection of stress. An Amulet wearable platform named StressAware was developed in using SVM. This real time applications classifies the stress level of individuals by continuously monitoring HR and HRV data. Some smart bands can collect and transmit data to users smart phone via Bluetooth and even uploaded to web where it can be accessible by doctor or family members. The overview of few studies are discussed in which shows stressors, subjects, sensors, best accuracy achieved, the classifiers and methods used by various researchers.

2.1.1 LIMITATIONS OF EXISTING SYSTEM

- The methods used for detecting stress are outdated and doesnot give theeffective results.
- Accurate and proper dataset is not available to train the machine.
- To avoid all these limitations and make the working more accurately the system needs to be implemented efficiently.

2.2 PROPOSED SYSTEM

In this paper, some previous approaches of automatic stress recognition systems who used sensors and machine learning are discussed in detail. In these, physiological data is extracted using some stressor tests on the people. Some common stressor tests includes arithmetic calculations, questionnaire, mental tasks and working out in gym. There are a diversity of machine learning algorithms which are appropriate for stress detection. Among them Support Vector Machines (SVM), Logistic regression, K- Nearest Neighbor, Decision tree and Random forest are most common.

In this review, we summarize the various machine learning algorithms available in theliterature that aim at detecting state of stress.

The remaining paper is divided into two sections. In Section II, we describe somemethods to detect and classify the stress levels..

2.1.1 ADVANTAGES OF PROPOSED SYSTEM

- The classifiers used in detecting stress gives us the resultswith highestaccuracy.
- T-pot classifier gives us the best idea on how to solve a machinelearningproblem.
- The dataset used to train the model is selected from kaggle datasets whichgives us the best idea of the values we have used.

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposalis putforth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is toensure that the proposed system is not a burden to the company. Three key considerations involved in the feasibility analysis are

- Economic Feasibility
- Technical Feasibility
- Social Feasibility

2.4.1 ECONOMIC FEASIBILITY

This study is carried out to check the economic impact that the system will haveon the organization.The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified.Thus the developed system as well within the budget and this was achieved because most of thetechnologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand onthe available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user.This includes the process of training the user to use the system efficiently.The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specifies the logical characteristics of each interface between the software product and the hardware components of the system. The following are somehardware requirements.

- Processor : Intel(R) Core(TM) i5-4300U,Pentium IV or higher.
- Harddisk : 500 GB
- RAM : 4GB
- Monitor : 5 inches or above.

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements,

- Operatingsystem : Windows 10
- Languages : Python 3.9.5
- Tool : Anaconda (Jupyter notebook,spyder)

3. ARCHITECTURE

3. ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for fake profile detection using machine learning.

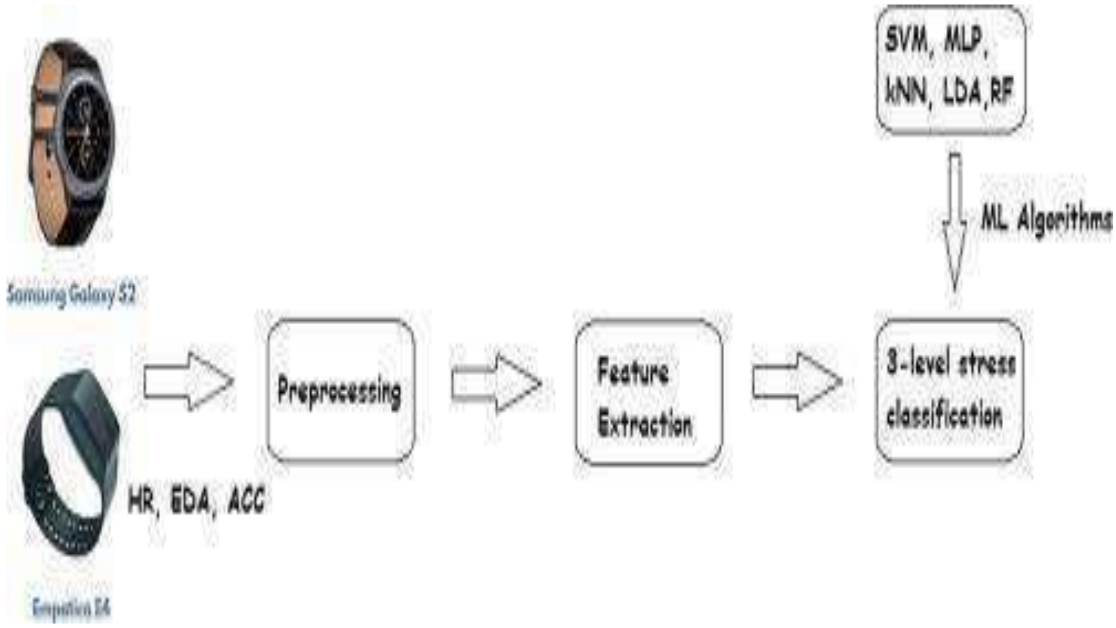


Figure 3.1 Architecture for Stress Detection

3.2 DESCRIPTION

The dataset is collected from the sensor based devices such as smart phones and smart watches.The collected data is preprocessed with the features we need. Here we use T-pot automated classifier to classify the data. TPOT is meant to be an assistant that gives you ideas on how to solve a particular machine learning problem by exploring pipeline configurations that you might have never considered, then leaves the fine-tuning to more constrained parameter tuning techniques such as grid search. TPOT is built on the scikit learn library and follows the scikit learn API closely. It can be used for regression and classification tasks and has special implementations for medical research.TPOT is open source, well documented, and under active development. TPOT has what its developers call a genetic search algorithm to find the best parameters and model ensembles. It could also be thought of as a natural selection or evolutionary algorithm. TPOT tries a pipeline, evaluates its performance, and randomly changes parts of the pipeline in search of better performing algorithms.

3.3 DATA FLOW DIAGRAM

A data flow diagram is a way of representing a flow of datathrough a process or system.The DFD also provides information about the outputs and inputs of each entity and the process itself.A data flowdiagram has no control flow-there are no decision loops and rules.

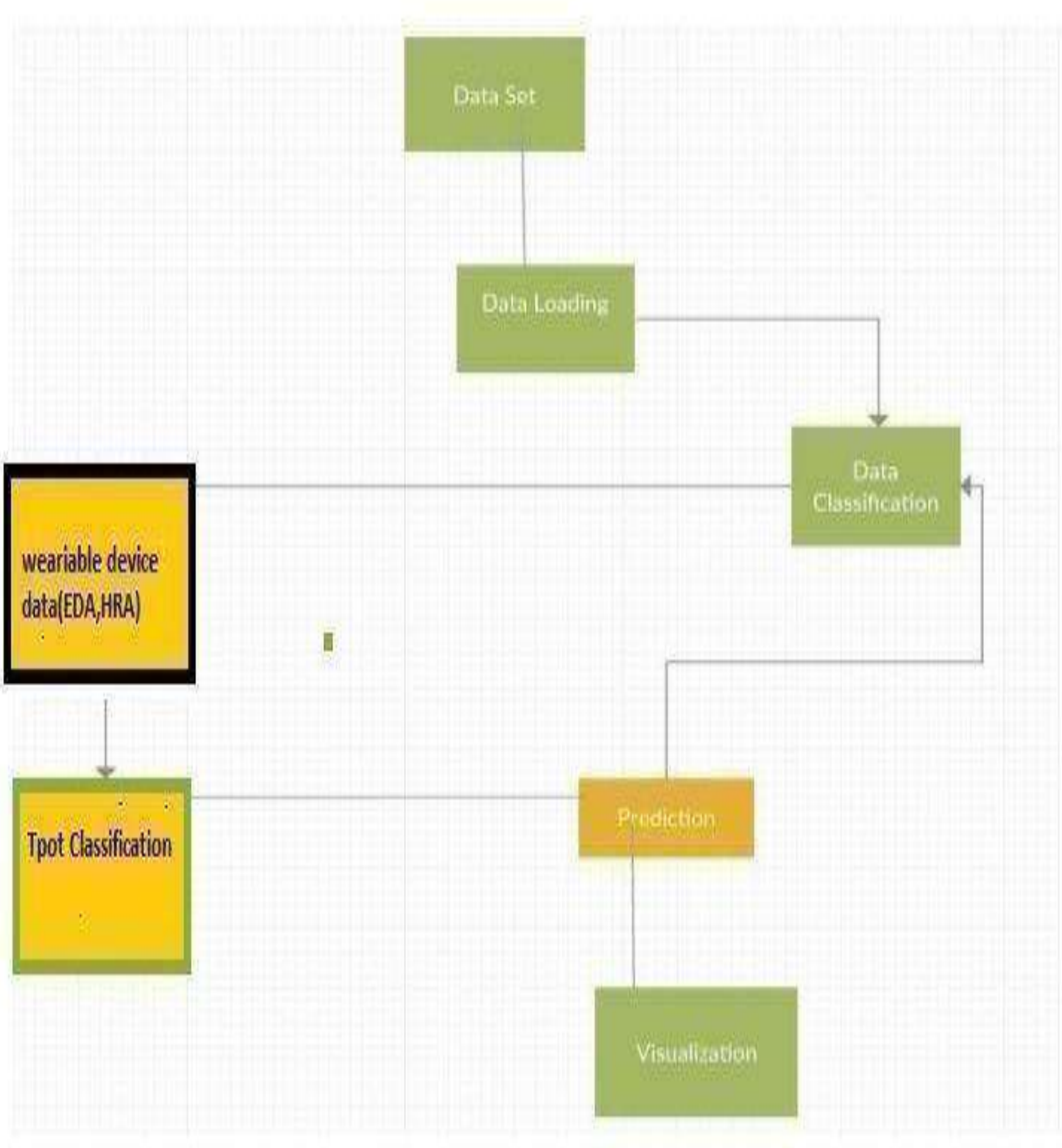


Figure 3.2 Data flow diagram for stress detection

4. IMPLEMENTATION

4. IMPLEMENTATION

4.1 SAMPLE CODE :

K -NEAREST NEIGHBOUR CLASSIFICATION CODE :

```
    {
"nbformat": 4,
"nbformat_minor": 0,
"metadata": {
  "colab": {
    "name": "KNeighbours Classifer.ipynb",
    "provenance": [],
    "collapsed_sections": []
  },
  "kernelspec": {
    "name": "python3",
    "display_name": "Python 3"
  },
  "accelerator": "GPU"
},
"cells": [
  {
    "cell_type": "code",
    "metadata": {
      "id": "xrHoFTwCQtGB",
      "colab_type": "code",
      "colab": {}
    },
    "source": [
      "import numpy as np\n",
      "\n",
      "import pandas as pd\n",
      "from sklearn import datasets, svm, metrics\n",
      "import matplotlib.pyplot as plt"
    ],
    "execution_count": 0,
```

```
  "outputs": []
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "gOQVdHCz0ZQ0",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
   "from google.colab import drive\n",
   "drive.mount('/content/drive')"
  ],
  "execution_count": 0,
  "outputs": []
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "TmHsAC4jSYI9",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
   "trainFile = pd.read_csv('/content/drive/My Drive/TechJam/train.csv').drop(columns=\"datasetId\")\n",
   "testFile = pd.read_csv('/content/drive/My Drive/TechJam/test.csv').drop(columns=\"datasetId\")"
  ],
  "execution_count": 0,
  "outputs": []
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "t3xkR7WAWfVO",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
```

```
    "#features\n",
    "X_train = trainFile.drop(columns='condition')\n",
    "y_train = trainFile['condition']\n",
    "X_test = testFile.drop(columns='condition')\n",
    "y_test = testFile['condition']"
   ],
   "execution_count": 0,
   "outputs": []
  },
  {
   "cell_type": "code",
   "metadata": {
    "id": "INrVmaTjYX1j",
    "colab_type": "code",
    "outputId": "0cf265a2-ffc9-459b-b809-041ca5e208ed",
    "colab": {
     "base_uri": "https://localhost:8080/",
     "height": 51
    }
   },
   "source": [
    "from sklearn.neighbors import KNeighborsClassifier\n",
    "knn = KNeighborsClassifier()\n",
    "knn.fit(X_train, y_train)\n",
    "print('Accuracy of K-NN classifier on training set: {:.2f}'\n",
    "     .format(knn.score(X_train, y_train)))\n",
    "print('Accuracy of K-NN classifier on test set: {:.2f}'\n",
    "     .format(knn.score(X_test, y_test)))"
   ],
   "execution_count": 7,
   "outputs": [
    {
     "output_type": "stream",
     "text": [
      "Accuracy of K-NN classifier on training set: 1.00\n",
      "Accuracy of K-NN classifier on test set: 0.99\n"
     ],
     "name": "stdout"
```

```
    }
  ]
},
{
  "cell_type": "code",
  "metadata": {
    "id": "_EImbKQpKx23",
    "colab_type": "code",
    "outputId": "84a158f1-3092-4b98-9bfa-2da1443653ab",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 34
    }
  },
  "source": [
    "i=1\n",
    "knn.predict([X_test.iloc[i]])"
  ],
  "execution_count": 10,
  "outputs": [
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "array(['time pressure'], dtype=object)"
        ]
      },
      "metadata": {
        "tags": []
      },
      "execution_count": 10
    }
  ]
},
{
  "cell_type": "code",
  "metadata": {
    "id": "yQxLkl7iBEzK",
```

```
  "colab_type": "code",
  "outputId": "e8d6fb66-ef30-4bb5-db66-d2e897bd03bc",
  "colab": {
   "base_uri": "https://localhost:8080/",
   "height": 605
  }
 },
 "source": [
  "print(X_test.iloc[i])"
 ],
 "execution_count": 0,
 "outputs": [
  {
   "output_type": "stream",
   "text": [
    "MEAN_RR             831.632295\n",
    "MEDIAN_RR           836.596795\n",
    "SDRR                 62.069042\n",
    "RMSSD                12.576667\n",
    "SDSD                 12.576557\n",
    "SDRR_RMSSD            4.935254\n",
    "HR                   72.562697\n",
    "pNN25                 4.533333\n",
    "pNN50                 0.266667\n",
    "SD1                   8.895937\n",
    "SD2                  87.326938\n",
    "KURT                 -0.087277\n",
    "SKEW                 -0.226072\n",
    "MEAN_REL_RR           0.000055\n",
    "MEDIAN_REL_RR         0.000707\n",
    "SDRR_REL_RR           0.015505\n",
    "RMSSD_REL_RR          0.008243\n",
    "SDSD_REL_RR           0.008243\n",
    "SDRR_RMSSD_REL_RR     1.881017\n",
    "KURT_REL_RR          -0.087277\n",
    "SKEW_REL_RR          -0.226072\n",
    "VLF                1534.857041\n",
    "VLF_PCT              71.422649\n",
```

```
      "LF              586.603990\n",
      "LF_PCT           27.296882\n",
      "LF_NU            95.519285\n",
      "HF               27.517012\n",
      "HF_PCT            1.280470\n",
      "HF_NU             4.480715\n",
      "TP             2148.978043\n",
      "LF_HF            21.317867\n",
      "HF_LF             0.046909\n",
      "sampen            2.177956\n",
      "higuci            1.202543\n",
      "Name: 5, dtype: float64\n"
     ],
     "name": "stdout"
    }
   ]
  }
 ]
}
```

CODE FOR FEED FORWARD NEURAL NETWORK :

```
    {
"nbformat": 4,
"nbformat_minor": 0,
"metadata": {
 "colab": {
  "name": "Detecting Stress using HRV.ipynb",
  "provenance": [],
  "collapsed_sections": []
 },
 "kernelspec": {
  "name": "python3",
  "display_name": "Python 3"
 },
 "accelerator": "GPU"
},
"cells": [
 {
  "cell_type": "code",
  "metadata": {
   "id": "OncBuCNo-b74",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
   "#math/data libs\n",
   "import numpy as np\n",
   "import pandas as pd\n",
   "from sklearn.preprocessing import MinMaxScaler\n",
   "from sklearn.preprocessing import OneHotEncoder\n",
   "\n",
   "#ml libs\n",
   "import keras\n",
   "from keras import backend as K\n",
   "from keras.models import Sequential\n",
   "from keras. layers import Activation\n",
```

```
    "from keras.layers.core import Dense\n",
    "from keras.optimizers import Adam\n",
    "from keras.metrics import categorical_crossentropy\n",
    "from keras.utils import to_categorical"
   ],
   "execution_count": 0,
   "outputs": []
  },
  {
   "cell_type": "code",
   "metadata": {
    "id": "h9P-G9taAH88",
    "colab_type": "code",
    "colab": {}
   },
   "source": [
    "from google.colab import drive\n",
    "drive.mount('/content/drive')"
   ],
   "execution_count": 0,
   "outputs": []
  },
  {
   "cell_type": "code",
   "metadata": {
    "id": "SqyeyGSuAyLu",
    "colab_type": "code",
    "colab": {}
   },
   "source": [
    "trainFile = pd.read_csv('/content/drive/My Drive/TechJam/train.csv').drop(columns=\"datasetId\")\n",
    "testFile = pd.read_csv('/content/drive/My Drive/TechJam/test.csv').drop(columns=\"datasetId\")"
   ],
   "execution_count": 0,
   "outputs": []
  },
  {
   "cell_type": "code",
```

```
  "metadata": {
    "id": "e-olX2v9BjWz",
    "colab_type": "code",
    "colab": {}
  },
  "source": [
    "#train\n",
    "train_samples = trainFile.drop(columns='condition').to_numpy()\n",
    "train_labels = trainFile['condition'].to_numpy()\n",
    "\n",
    "#test\n",
    "test_samples = testFile.drop(columns='condition').to_numpy()\n",
    "test_labels = testFile['condition'].to_numpy()"
  ],
  "execution_count": 0,
  "outputs": []
},
{
  "cell_type": "code",
  "metadata": {
    "id": "zjKqblmHC8BF",
    "colab_type": "code",
    "colab": {}
  },
  "source": [
    "#normalizing features\n",
    "scaler = MinMaxScaler(feature_range=(0,1))\n",
    "train_samples = scaler.fit_transform(train_samples)\n",
    "test_samples = scaler.fit_transform(test_samples)\n",
    "\n",
    "#one-hot-encoding labels\n",
    "one_hot_encoder = OneHotEncoder(categories='auto')\n",
    "train_labels = one_hot_encoder.fit_transform(train_labels.reshape(-1, 1)).toarray()\n",
    "test_labels = one_hot_encoder.fit_transform(test_labels.reshape(-1, 1)).toarray()"
  ],
  "execution_count": 0,
  "outputs": []
},
```

```
{
 "cell_type": "code",
 "metadata": {
  "id": "mExpPxiO_TuV",
  "colab_type": "code",
  "colab": {}
 },
 "source": [
  "#build the model\n",
  "model = Sequential([\n",
  "    Dense(34, input_shape=[34,], activation='relu'),\n",
  "#     Dense(20, activation='relu'),\n",
  "    Dense(10, activation='relu'),\n",
  "    Dense(3, activation='softmax')\n",
  "])"
 ],
 "execution_count": 0,
 "outputs": []
},
{
 "cell_type": "code",
 "metadata": {
  "id": "zYJPmEjG_TwQ",
  "colab_type": "code",
  "colab": {
   "base_uri": "https://localhost:8080/",
   "height": 257
  },
  "outputId": "b4011586-45a3-417d-b2a3-7704090697ad"
 },
 "source": [
  "model.summary()"
 ],
 "execution_count": 85,
 "outputs": [
  {
   "output_type": "stream",
   "text": [
```

```
      "Model: \"sequential_8\"\n",
      "_________________________________________________________________\n",
      "Layer (type)                 Output Shape              Param#  \n",
      "=================================================================\n",
      "dense_26 (Dense)             (None, 34)                1190      \n",
      "_________________________________________________________________\n",
      "dense_27 (Dense)             (None, 10)                350       \n",
      "_________________________________________________________________\n",
      "dense_28 (Dense)             (None, 3)                 33        \n",
      "=================================================================\n",
      "Total params: 1,573\n",
      "Trainable params: 1,573\n",
      "Non-trainable params: 0\n",
      "_________________________________________________________________\n"
     ],
     "name": "stdout"
    }
   ]
  },
  {
   "cell_type": "code",
   "metadata": {
    "id": "nMq65bPkBP6T",
    "colab_type": "code",
    "colab": {}
   },
   "source": [
    "model.compile(Adam(lr=.0001),\n",
    "              loss='categorical_crossentropy',\n",
    "              metrics=['accuracy'])"
   ],
   "execution_count": 0,
   "outputs": []
  },
  {
   "cell_type": "code",
   "metadata": {
    "id": "saBFc3aVBQIU",
```

```
    "colab_type": "code",
    "colab": {
      "base_uri": "https://localhost:8080/",
      "height": 154
    },
    "outputId": "a163d56d-e621-489e-cf3f-4abaf111385e"
   },
   "source": [
    "model.fit(train_samples, train_labels, validation_split=0.1, batch_size=10, epochs=3, shuffle=True, verbose=2)"
   ],
   "execution_count": 87,
   "outputs": [
    {
      "output_type": "stream",
      "text": [
        "Train on 332360 samples, validate on 36929 samples\n",
        "Epoch 1/3\n",
        " - 112s - loss: 0.8319 - acc: 0.6369 - val_loss: 0.7507 - val_acc: 0.6802\n",
        "Epoch 2/3\n",
        " - 110s - loss: 0.6985 - acc: 0.6984 - val_loss: 0.6563 - val_acc: 0.7144\n",
        "Epoch 3/3\n",
        " - 109s - loss: 0.6208 - acc: 0.7385 - val_loss: 0.5923 - val_acc: 0.7466\n"
      ],
      "name": "stdout"
    },
    {
      "output_type": "execute_result",
      "data": {
        "text/plain": [
          "<keras.callbacks.History at 0x7fba404f0b70>"
        ]
      },
      "metadata": {
        "tags": []
      },
      "execution_count": 87
    }
```

```
  ]
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "ylh_L1FTP-sa",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
   "model.save('model.h5')"
  ],
  "execution_count": 0,
  "outputs": []
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "OBlzeL4Uj4J8",
   "colab_type": "code",
   "colab": {
    "base_uri": "https://localhost:8080/",
    "height": 137
   },
   "outputId": "1ede1afb-35ac-42f9-d33d-dd08a53658e3"
  },
  "source": [
   "model.predict(test_samples)"
  ],
  "execution_count": 92,
  "outputs": [
   {
    "output_type": "execute_result",
    "data": {
     "text/plain": [
      "array([[0.1126608 , 0.8332116 , 0.05412759],\n",
      "       [0.02249658, 0.77812797, 0.19937544],\n",
      "       [0.47015983, 0.48569497, 0.0441452 ],\n",
```

```
          "       ...,\n",
          "       [0.00916936, 0.6517755 , 0.33905518],\n",
          "       [0.08203984, 0.71687603, 0.20108415],\n",
          "       [0.32056192, 0.5739703 , 0.10546778]], dtype=float32)"
         ]
        },
        "metadata": {
         "tags": []
        },
        "execution_count": 92
      }
    ]
  },
  {
    "cell_type": "code",
    "metadata": {
      "id": "8lFvO6kLkjB7",
      "colab_type": "code",
      "colab": {
        "base_uri": "https://localhost:8080/",
        "height": 240
      },
      "outputId": "2ac3fb12-d44c-4e20-829b-9895057a8b3d"
    },
    "source": [
      "test_samples"
    ],
    "execution_count": 94,
    "outputs": [
      {
        "output_type": "execute_result",
        "data": {
          "text/plain": [
            "array([[0.22531332, 0.18486409, 0.08844309, ..., 0.26053516, 0.92367922,\n",
            "        0.64162156],\n",
            "       [0.38244367, 0.28799639, 0.0581626 , ..., 0.04047566, 0.99034621,\n",
            "        0.66112158],\n",
            "       [0.53098161, 0.39563865, 0.19693847, ..., 0.0183135 , 0.99026903,\n",
```

```
      "      0.33885214],\n",
      "     ...,\n",
      "     [0.18331272, 0.12845198, 0.19874266, ..., 0.27243779, 0.91292724,\n",
      "      0.55324208],\n",
      "     [0.27636484, 0.21461972, 0.062443 , ..., 0.17858342, 0.97595153,\n",
      "      0.61357042],\n",
      "     [0.41409514, 0.31321216, 0.06339303, ..., 0.03413039, 0.9491316 ,\n",
      "      0.33185303]])"
     ]
    },
    "metadata": {
     "tags": []
    },
    "execution_count": 94
   }
  ]
 },
 {
  "cell_type": "code",
  "metadata": {
   "id": "WvJQx81npgJ1",
   "colab_type": "code",
   "colab": {}
  },
  "source": [
   ""
  ],
  "execution_count": 0,
  "outputs": []
 }
]
}
```

5. SCREENSHOTS

5.1 IMPORTING THE DATASET TO TRAIN THE MODEL

```
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import GridSearchCV
from tpot import TPOTClassifier
%matplotlib inline
import matplotlib.pyplot as plt
from scipy import signal
import pickle


import sklearn.metrics
from sklearn.model_selection import cross_val_score
from sklearn import svm
import numpy as np
import pandas as pd
from sklearn.metrics import precision_recall_fscore_support

pd.set_option('display.max_columns', None)
pd.set_option('display.max_rows', None)
pd.set_option('display.max_colwidth', -1)
dataframe_hrv = pd.read_csv("hrv dataset/hrv dataset/data/final/train.csv")
dataframe_hrv = dataframe_hrv.reset_index(drop=True)
display(dataframe_hrv.head(5))
```

```
C:\Users\home\anaconda3\lib\site-packages\tpot\builtins\__init__.py:36: UserWarning: Warning: optional dependency `torch` is no
t available. - skipping import of NN models.
  warnings.warn("Warning: optional dependency `torch` is not available. - skipping import of NN models.")
<ipython-input-1-adecb2b93410>:118: FutureWarning: Passing a negative integer is deprecated in version 1.0 and will not be supp
orted in future version. Instead, use None to not limit the column width.
  pd.set_option('display.max_colwidth', -1)
```

	MEAN_RR	MEDIAN_RR	SDRR	RMSSD	SDSD	SDRR_RMSSD	HR	pNN25	pNN50	SD1	SD2	KURT	SKEW	ME
0	885.157845	853.763730	140.972741	15.554505	15.553371	9.063146	69.499952	11.133333	0.533333	11.001565	199.061782	-0.856554	0.335218	
1	939.425371	948.357865	81.317742	12.964439	12.964195	6.272369	64.363150	5.600000	0.000000	9.170129	114.634458	-0.408190	-0.155286	
2	898.186047	907.006860	84.497236	16.305279	16.305274	5.182201	67.450066	13.066667	0.200000	11.533417	118.939253	0.351789	-0.656813	
3	881.757865	893.460030	90.370537	15.720468	15.720068	5.748591	68.809562	11.800000	0.133333	11.119476	127.318597	-0.504947	-0.386138	
4	809.625331	811.184865	62.766242	19.213819	19.213657	3.266724	74.565728	20.200000	0.200000	13.590641	87.718281	-0.548408	-0.154252	

Screenshot 5.1 Importing the dataset to train the model

5.2 CONDITION STATEMENT FOR STRESS DETECTION

```
def fix_stress_labels(df='',label_column='condition'):
    df['condition'] = np.where(df['condition']=='no stress', 0, 1)
    display(df["condition"].unique())
    return df
dataframe_hrv = fix_stress_labels(df=dataframe_hrv)
Y = dataframe_hrv['condition']
display(Y.head(5))
```

```
array([0, 1])

0    0
1    1
2    1
3    0
4    0
Name: condition, dtype: int32
```

Screenshot 5.2 Condition statement for stress detection

5.3 INSERTING THE COLUMNS NEEDED FOR STRESS DETECTION

```
selected_x_columns = ['HR', 'RMSSD', 'pNN50', 'TP', 'VLF', 'LF', 'HF','LF_HF']
X = dataframe_hrv[selected_x_columns]
display(X.head(5))
```

	HR	RMSSD	pNN50	TP	VLF	LF	HF	LF_HF
0	69.499952	15.554505	0.533333	3686.666157	2661.894136	1009.249419	15.522603	65.018055
1	64.363150	12.964439	0.000000	3006.487251	2314.265450	690.113275	2.108525	327.296635
2	67.450066	16.305279	0.200000	2685.879461	1373.887112	1298.222619	13.769729	94.280910
3	68.809562	15.720468	0.133333	3434.520980	2410.357408	1005.981659	18.181913	55.328701
4	74.565728	19.213819	0.200000	2621.175204	1151.177330	1421.782051	48.215822	29.487873

Screenshot 5.3 Inserting the columns needed for stress detection

5.4 ASSIGNING THE MINIMUM VALUES FOR STRESS DETECTION

```
newDataframe_hrv = pd.read_csv("hrv dataset/hrv dataset/data/final/test.csv")
newdataframe_hrv = dataframe_hrv.reset_index(drop=True)
new_selected_x_columns = ['HR', 'RMSSD', 'pNN50', 'TP', 'VLF', 'LF', 'HF','LF_HF']
newX = newDataframe_hrv[selected_x_columns]
display(newX.head(5))
```

	HR	RMSSD	pNN50	TP	VLF	LF	HF	LF_HF
0	84.121868	12.361264	0.000000	1698.605390	1016.073759	615.914573	66.617057	9.245599
1	71.478642	19.298880	0.200000	2358.884694	765.518473	1566.866135	26.500086	59.126832
2	63.874293	21.342715	1.800000	4328.633724	2237.739905	2074.868884	16.024935	129.477524
3	74.330531	11.771814	0.533333	2854.449091	2330.980957	505.886664	17.581470	28.773854
4	82.092049	13.357748	0.666667	5310.027472	4750.624447	524.203971	35.199054	14.892559

Screenshot 5.4 Assigning the minimum values for stress detection

5.5 VALUES OF CV SCORES FOR DATASET

```
def do_tpot(generations=5, population_size=10,X='',y=''):

    X_train, X_test, y_train, y_test = train_test_split(X, y,train_size=0.80,test_size=0.20)
    tpot = TPOTClassifier(generations=generations, population_size=population_size, verbosity=2,cv=3)
    tpot.fit(X_train, y_train)
    print(tpot.score(X_test, y_test))
    tpot.export('tpot_pipeline.py')
    return tpot

tpot_classifer = do_tpot(generations=5, population_size=20,X=X,y=Y)
```

```
Generation 1 - Current best internal CV score: 0.9999593813783929

Generation 2 - Current best internal CV score: 0.9999593813783929

Generation 3 - Current best internal CV score: 0.9999593813783929

Generation 4 - Current best internal CV score: 0.9999593813783929

Generation 5 - Current best internal CV score: 0.9999661511486607

Best pipeline: ExtraTreesClassifier(ZeroCount(input_matrix), bootstrap=False, criterion=entropy, max_features=0.8, min_samples_
leaf=3, min_samples_split=2, n_estimators=100)
0.9999864605052939
```

Screenshot 5.5 Values of cv scores for dataset

5.6 CONDITION VALUES FOR DISPLAYING STRESS CONDITION

```
display(newDataframe_hrv['condition'].head(5))
```

```
0    no stress
1    time pressure
2    no stress
3    no stress
4    interruption
Name: condition, dtype: object
```

Screenshot 5.6 Condition values for displaying stress condition

5.7 STRESS IS DISPLAYED IN BINARY VARIABLES FORMAT

```
pred = tpot_classifer.predict_proba(newX)
dfpred = pd.DataFrame(pred)
display(dfpred.head(5))
```

	0	1
0	1.00000	0.00000
1	0.00000	1.00000
2	1.00000	0.00000
3	0.99993	0.00007
4	0.05469	0.94531

Screenshot 5.7 Stress is displayed in binary variables format

6. TESTING

6. TESTING

6.1 INTRODUCTION TO TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discovere very conceivable fault or weakness in a work product. It provides a way to check the functionality ofcomponents, subassemblies, assemblies and/or a finished product. It is the process of exercisingsoftware with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

6.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actuallyrun as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specificallyaimed at exposing the problems that arise from the combination of components.

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specifiedby the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes.

6.3 TEST CASES

6.3.2 TEST CASE 1

Test case for Training data :

The model is trained using dataset taken from smart watches and kaggle datasets.

Test case ID	Test case name	Test case	Output
1	Uploading correct dataset	User uploads a data which is correct relevant to all values	Upload successful
2	Uploading incorrect dataset	User uploads a data which is incorrectrelevant to all values	Upload successful

6.2.2 TEST CASE 2 :

Test case for Testing data :

The trained models are tested and their results are considered to fine tune the parameters of themodel.

Test case ID	Test case name	Test case	Output
1	Empty fields testing	In the field of datasetif values are not there	It shows an error box showing fields are empty
2	Wrong fields testing	A unique dataset features are given bythe user.By entering other values gives	Testing fails and shows an alert box stating that entered values are incorrect
3	Stress detection fails	If the user doesnot provide the correct informatio or keeps any of the box empty then	Stress detection givesan output as detection failed due to invalid data

7. CONCLUSION

7. CONCLUSION AND FUTURE SCOPE

7.1 PROJECT CONCLUSION

We developed a stress detection scheme to be used in real life.We have collected different parametric values of heart beat from smart watches and obtained the stress resullts. We achievedmaximum 97.92% accuracy for three-level stress detection. The best performing classifiers were the Random Forest and the Multilayer Perceptron algorithms. classification accuracy, whereas this was 86.27% when these modalities were used separately. Finally, we observed that the perceived stress level classification results in lower accuracies than physiological stress level classification. There were up to 15% decrease when compared with physiological stress level classification accuracies.

7.2 FUTURE SCOPE

As a future work, we plan to record data with an increased number of high-quality Empatica E4 devices. We further plan to develop personalized perceived stress level models from ground truth surveys and remove outlier answers to increase the perceived stress level classification accuracies...

8. BIBILOGRAPHY

8.BIBILOGRAPHY

8.1 REFERENCES

[1]R.W. Automating the Recognition of Stress and Emotion: From Lab to Real- World Impact. IEEE MultiMedia. 2016;23:3–7. doi: 10.1109/MMUL.2016.38. [CrossRef] [GoogleScholar]

[2]England M.J., Liverman C.T., Schultz A.M., Strawbridge L.M. Epilepsy across the spectrum: Promoting health and understanding: A summary of the Institute of Medicine report.EpilepsyBehav. 2012;25:266–276 doi: 10.1016/j.yebeh.2012.06.016. [PMC free article] [PubMed] [CrossRef] [Google Scholar]

[3] Ryvlin P., Nashef L., Lhatoo S.D., Bateman L.M., Bird J., Bleasel A., Boon P., Crespel A., Dworetzky B.A., Høgenhaven H., et al. Incidence and mechanisms of cardiorespiratory arrests inepilepsy monitoring units (MORTEMUS): A retrospective study. Lancet Neurol. 2013;12:966– 977. doi: 10.1016/S1474-4422(13)70214-X. [PubMed] [CrossRef] [Google Scholar]

[4]Colligan T.W., Higgins E.M. Workplace stress: Etiology and consequences. J. Workplace Behav. Health. 2006;21:89–97. Doi: 10.1300/J490v21n02_07. [CrossRef] [Google Scholar]

8.2 WEBSITES

[1] Code snippets for any errors http://stackoverflow.com/

[2] Software Testing http://en.wikipedia.org/wiki/Software_testing

8.3 GITHUB LINK

https://github.com/Geetha-96/STRESS-DETECTION-FOR-WEARABLE-DEVICES.git

International Journal of Advances in Engineering and Management (IJAEM)
Volume 4, Issue 6 June 2022, pp: 131-135 www.ijaem.net ISSN: 2395-5252

Stress Detection for Wearable Devices

D.Sandhya Rani[1],Ch.Geethanjali[2],G.Anjana[3],R.Abhishek[4]
[1]Assistant Professor,Dept of CSE ,CMR Technical Campus,Kandlakoya,Medchal.
[2]Student,Dept of CSE, CMR Technical campus,Kandlakoya,Medchal.
[3]Student,Dept of CSE ,CMR Technical campus,Kandlakoya,Medchal.
[4]Student,Dept of CSE ,CMR Technical campus,Kandlakoya,Medchal.

Submitted: 25-05-2022 Revised: 01-06-2022 Accepted: 05-06-2022

ABSTRACT
In this project, Stress can be recognized by observing commute in sensitivity on the human body. The actuators which are apparel are becoming more prominent in recent years due to their functionality and discrete nature. By utilizing data from wearable sensors, we have developed a customizable stress detection system. Our system performs categorization on stress level using cross modal data from wrist-worn device Empatica E4 wearable sensor. We implemented three different classification algorithms: Logistic Regression, Decision Tree, and Random Forest and used four-class categorization conditions: baseline, stress, amusement, and meditation. By estimating the performance of the system, we exhibit that our system can perform the best and consistent customized stress detection using T-POT classifier with the accuracy of 88%-99% on 15 subjects.

KEYWORDS: Stress classification , Preprocessing , random forest ,T-POT Classifier , Wearable devices

I. INTRODUCTION

This project is titled as "Stress Detection using wearable devices". This software provides facility to identify how the stress is detected . This project uses machine-learning methods to identify how the stress is detected. First, we use a T-pot classifer to train the dataset. Then we identify how the stress is detected. This has been developed to facilitate the identification, retrieval of the items and information. System is built with manually exclusive features. In all cases system will specify object which are physical or on performance characteristics. They are used to give optimal distraction and other information. Data are used for identifying, accessing, storing and identifying fake accounts. The data ensures that only one value of the code with a single meaning is correctly applied to give entity or attribute as described in various ways. The main features of this project are that the designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for anendorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user issatisfied with proposal.

II. LITERATURE REVIEW

In this section, we briefly summarize some approaches of stress detection. These approaches vary according to the various stress related factors and measures used. The measures used in this includes physical measures, physiological signals, answering questionnaire, mathematical test, videos, microblog and other techniques, etc. Also, stress detection in various environments is described below. A) Stress Detection using Wearable Sensors and IOT Devices Nowadays, sensors plays a vital role in medical applications. Theseare generally used for detection and measurement of various diseases and its levels. Stress is usually recognizedas one of the major factors leading to various health problems. Therefore, people with high risk of getting stressed should be continuously monitored for detection of any stress signs before it causes health problems[8]. Advances in wearable sensors and mobile computing make it possible to record a variety of physical and physiological signals on a twenty-four hour basis which helps in detection of stress level. Mostly wearable sensor devices like smart band[3], Chest belts[2] are used for data collection. Some researchers used hardware and software for collection of data through sensors and detection ofstress level respectively. A Holster unit was used with LI- PO battery and PC USB Client software for detection of stress[2]. An Amulet wearable platform named StressAware was developed in [7] using SVM. This real time applications classifies the stress level of individuals by continuously monitoring HR and HRV data. Some smart bands can collect and transmit data to users smart phone via Bluetooth and even uploaded to web where it can be accessible by doctor or family members[3]. The overview of few studies are discussed which shows stressors, subjects, sensors, best accuracy achieved, the classifiers and methods used by various researchers.

III. PROPOSED DESIGN

In today's fast-paced world, mental stress is very common. Stress can be originated due to conditions or incidents such put oppression on mind and body of a person. Reaction to stress is different for everyone as the capacity of dealing with tough or demanding situations vary for person to person. Some instances might create stress to one individual, while no stress to one altogether. Also, all stress is not bad to health as it could make people more aware of things around them and keep them more cautious about dangers and focused on their goal. A stressor is an situation which creates stress to an person. Many people generally faces stress due to these stressors described in accord American Psychological Association (APA),there are mainly three types of stress which are short term tension, episodic acute tension and long term tension. Acute stress is short term stress which is least damaging type as compared to the other two. It can be good sometimes as this helps body to communicate with the event. When acute stress occurs frequently then an individual is affected with episodic acute stress. Long term stress is the most harmful type of stress, if left untreated over a long period of time can damage bodily and emotional health of a person. Long term stress puts force on the body and mind for an extended period which can create a range of indications and extend the risk of evolving certain diseases.To avoid health issues, people with high probability of feeling stressed should be continuously monitored to detect any stress signs. Wearable sensors create opportunities to monitor stress and could inform individuals regarding their stress level which can be useful in order to minimize stress balance as it results into severe health issues. Bodily health and emotional health are closely connected, hence monitoring and measuring of physiological and physical changes could be used for detecting human stress level. Stress can be detected using bodily and emotional measures of body. Bodily measures include pulse rate, skin temperature, humidity, Blood pressure and respiration rate whereas physiological measures can be heart rate, heart rate variability, skin conductance. These can be measured using wearable devices made from low-cost sensors although machine learning algorithms can be used to classify and predict stress level of an individual. In this paper, some previous approaches of automatic stress recognition systems who used sensors and machine learning are discussed in detail. In these, emotional data is extracted using some stressor tests on the people. Some common stressor tests includes arithmetic calculations, questionnaire, mental tasks and working out in gym. There are a diversity of machine learning algorithms which are appropriate for stress detection. Among them Support Vector Machines (SVM), Logistic regression, K-Nearest Neighbor, Decision tree and Random forest are most common. In this review, we summarize the various machine learning algorithms available in the writings thataim at perceiving state of stress.

PROPOSED SYSTEM

In this paper, some previous approaches of automatic stress identification systems who used sensors and also machine learning are discussed in detail. In these, physiological data is extracted using some stressor tests on the people. Some common stressor tests includes arithmetic calculations, questionnaire, mental tasks and working out in gym. There are a diversity of machine learning algorithms which are appropriate in stress detection. Among them Support Vector Machines (SVM), Logistic regression, KNearest Neighbor, Decision tree and Random forest are most common. In this review, we summarize the various machine learning algorithms present in the literature that aim at detecting state of stress.

PROJECT ARCHITECTURE

The dataset is collected from the sensor based devices like smart phones and smart watches.The collecteddata is preprocessed with the features we'd like. Here we use T-pot automated classifier to classify the information. TPOT is supposed to be an assistant that provides you ideas on the way to solve a selected machine learning problem by exploring pipeline configurations that you simply may need never considered, then leaves the fine- tuning to more constrained parameter tuning techniques like grid search. TPOT is made on the scikit learn library and follows the scikit learn API closely. It may be used forregression and classification tasks and has special implementations for medical research.TPOT is open source, well documented, and under active development. TPOT has what its developers call a genetic search algorithm to seek out the most effective parameters and model ensembles. It could even be thought of as a selection or evolutionary algorithm. TPOT

International Journal of Advances in Engineering and Management (IJAEM)
Volume 4, Issue 6 June 2022, pp: 131-135 www.ijaem.net ISSN: 2395-5252

tries a pipeline, evaluates its performance, and randomly changes parts of the pipeline in search of higher performing algorithms.

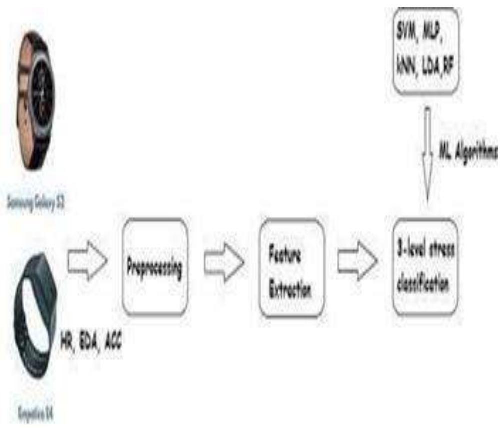
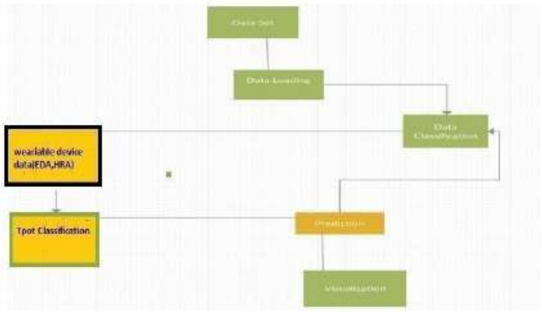


Fig 1. Architecture of stress detection

DATA FLOW DIAGRAM




```
selected_x_columns = ['HR', 'RMSSD', 'pNN50', 'TP', 'VLF', 'LF', 'HF','LF_HF']
X = dataframe_hrv[selected_x_columns]
display(X.head(5))
```

	HR	RMSSD	pNN50	TP	VLF	LF	HF	LF_HF
0	68.499952	15.554505	0.533333	3686.666157	2661.894136	1009.249419	15.522603	65.018055
1	64.363150	12.964439	0.000000	3066.487251	2314.265450	690.113275	2.108525	327.296635
2	67.450066	16.305279	0.200000	2685.879461	1373.887112	1298.222619	13.769729	94.280910
3	68.809562	15.720468	0.133333	3434.523980	2410.357408	1005.981659	18.181913	55.328701
4	74.565726	19.213819	0.200000	2621.175204	1151.177330	1421.782061	48.215822	29.487873

Fig 2. Dataflow diagram of stress detection

IV. RESULTS AND DISCUSSION

Fig 3. Importing the dataset

In the above figure 3,the dataset is imported from various devices such as kaggle datasets and smart devices.

```
def fix_stress_labels(df='',label_column='condition'):
    df['condition'] = np.where(df['condition']=='no stress', 0, 1)
    display(df["condition"].unique())
    return df
dataframe_hrv = fix_stress_labels(df=dataframe_hrv)
Y = dataframe_hrv['condition']
display(Y.head(5))
```

```
array([0, 1])

0    0
1    1
2    1
3    0
4    0
Name: condition, dtype: int32
```

Fig 4. Condition statement for stress detection

```
selected_x_columns = ['HR', 'RMSSD', 'pNN50', 'TP', 'VLF', 'LF', 'HF','LF_HF']
X = dataframe_hrv[selected_x_columns]
display(X.head(5))
```

	HR	RMSSD	pNN50	TP	VLF	LF	HF	LF_HF
0	68.499952	15.554505	0.533333	3686.666157	2661.894136	1009.249419	15.522603	65.018055
1	64.363150	12.964439	0.000000	3066.487251	2314.265450	690.113275	2.108525	327.296635
2	67.450066	16.305279	0.200000	2685.879461	1373.887112	1298.222619	13.769729	94.280910
3	68.809562	15.720468	0.133333	3434.523980	2410.357408	1005.981659	18.181913	55.328701
4	74.565726	19.213819	0.200000	2621.175204	1151.177330	1421.782061	48.215822	29.487873

Fig 5. Inserting columns needed for stress detection

```
newDataframe_hrv = pd.read_csv("hrv dataset/hrv dataset/data/final/test.csv")
newdataframe_hrv = dataframe_hrv.reset_index(drop=True)
new_selected_x_columns = ['HR', 'RMSSD', 'pNN50', 'TP', 'VLF', 'LF', 'HF','LF_HF']
newX = newDataframe_hrv[selected_x_columns]
display(newX.head(5))
```

	HR	RMSSD	pNN50	TP	VLF	LF	HF	LF_HF
0	84.121868	12.361264	0.000000	1698.605390	1016.073759	615.914573	66.617057	9.245599
1	71.478842	19.298880	0.200000	2358.884694	765.518473	1566.866135	26.500086	59.126832
2	63.874293	21.342715	1.800000	4328.633724	2237.739905	2074.868884	16.024935	129.477524
3	74.330531	11.771814	0.533333	2854.449091	2330.980957	505.888664	17.581470	28.773854
4	82.092049	13.357748	0.666667	5310.027472	4750.624447	524.203971	35.199054	14.892559

Fig 6. Assigning the minimum values

```
def do_tpot(generations=5, population_size=10,X='',y=''):

    X_train, X_test, y_train, y_test = train_test_split(X, y,train_size=0.80,test_size=0.20)
    tpot = TPOTClassifier(generations=generations, population_size=population_size, verbosity=2,cv=3)
    tpot.fit(X_train, y_train)
    print(tpot.score(X_test, y_test))
    tpot.export('tpot_pipeline.py')
    return tpot

tpot_classifer = do_tpot(generations=5, population_size=20,X=X,y=Y)
```

```
Generation 1 - Current best internal CV score: 0.9999[illegible]
Generation 2 - Current best internal CV score: 0.9999[illegible]
Generation 3 - Current best internal CV score: 0.9999[illegible]
Generation 4 - Current best internal CV score: 0.9999[illegible]
Generation 5 - Current best internal CV score: 0.9999[illegible]

Best pipeline: ExtraTreesClassifier(ZeroCount(input_matrix), bootstrap=False, criterion=entropy, max_features=0.8, min_samples_
leaf=3, min_samples_split=2, n_estimators=100)
0.9999[illegible]
```

Fig 7. Values of cv scores

```
display(newDataframe_hrv['condition'].head(5))
```

```
0    no stress
1    time pressure
2    no stress
3    no stress
4    interruption
Name: condition, dtype: object
```

Fig 8. Condition values for stress detection

```
pred = tpot_classifer.predict_proba(newX)
dfpred = pd.DataFrame(pred)
display(dfpred.head(5))
```

	0	1
0	1.00000	0.00000
1	0.00000	1.00000
2	1.00000	0.00000
3	0.99993	0.00007
4	0.05469	0.94531

Fig 9. Output in binary variables format

V. CONCLUSION

We developed a stress detection scheme to beutilized in real world.We have collected different parametric values of heart beat from smart watches and obtained the strain resullts. We achieved maximum 97.92% accuracy for three-level stress detection. the most effective performing classifiers were the Random Forest and also the Multilayer Perceptron algorithms. classification accuracy, whereas this was 86.27% when these modalities were used separately. Finally, we observed that the perceived stress level classification leads to lower accuracies than physiological stress level classification. there have been up to fifteen decrease in comparison with physiological stress level classification accuracies.

VI. ACKNOWLEDGMENT

The success of this project includes help from our guide as well.We are grateful to our guide,Mrs.D.Sandhya Rani,Assistant Professor,CMR Technical Campus,for her expertise that guided us in our reasearch.

REFERENCES

[1] Picard R.W. Automating the Recognition of Stress and Emotion: From Lab to Real-World Impact. IEEE MultiMedia.2016;23:3–7.doi: 10.1109/MMUL.2016.38. [CrossRef] [Google Scholar]

[2] England M.J., Liverman C.T., Schultz A.M., Strawbridge L.M. Epilepsy across the spectrum: Promoting health and understanding: A summary of the InstituteofMedicine report. Epilepsy Behav. 2012;25:266–276.doi:10.1016/j.yebeh.2012.06.016. [PMCfree article] [PubMed] [CrossRef] [Google Scholar]

[3] Ryvlin P., Nashef L., Lhatoo S.D., Bateman L.M., Bird J., Bleasel A., Boon P., Crespel A., Dworetzky B.A., Høgenhaven H., et al. Incidence and mechanisms of cardiorespiratory arrests in epilepsy monitoring units (MORTEMUS): A retrospective study. Lancet Neurol. 2013;12:966–977. doi: 10.1016/S1474-4422(13)70214-X. [PubMed] [CrossRef] [Google Scholar]

[4] Colligan T.W., Higgins E.M. Workplace stress: Etiology and consequences. J. Workplace Behav. Health. 2006;21:89–97.doi:10.1300/J490v21n02_07. [CrossRef] [Google Scholar]

[5] American Psychology Association . Stress: The Different Kinds of Stress. American

Psychology Association; Washington, DC, USA: 2019. [Google Scholar]

[6] rantz D.S., Whittaker K.S., Sheps D.S. Heart and Mind: Evolution of Cardiac Psychology. American Psychological Association; Washington, DC, USA: 2011. Psychosocial risk factors for coronary heart disease: Pathophysiologic mechanisms. [Google Scholar]

[7] Pickering T.G. Mental stress as a causal factor in the development of hypertension and cardiovascular disease. Curr. Hypertens. Rep. 2001;3:249–254. doi: 10.1007/s11906-001-0047-1. [PubMed][CrossRef] [Google Scholar]

[8] Mönnikes H., Tebbe J., Hildebrandt M., Arck P., Osmanoglou E., Rose M., Klapp B., Wiedenmann B., Heymann-Mönnikes I. Role of stress in functional gastrointestinal disorders. Dig. Dis. 2001;19:201–211. doi: 10.1159/000050681. [PubMed][CrossRef] [GoogleScholar]

[9] Herbert J. Fortnightly review: Stress, the brain, and mental illness. BMJ. 1997;315:530–535. doi: 10.1136/bmj.315.7107.530. [PMC free article] [PubMed][CrossRef][Google Scholar]

[10] Milczarek M., Elke Schneider E.G. OSH in Figures, Stress at Work, Fact and Figures. European Agency for Safety and Health at Work; Bilbao, Spain: 2009. [Google Scholar]European Agency for Safety and Health at Work . European Opinion Poll on Occupational Safety and Health. European Agency for Safety and Health at Work; Bilbao, Spain: 2013. [CrossRef] [Google Scholar]

[11] Alberdi A., Aztiria A., Basarab A. Towards an automatic early stress recognition system for officeenvironments based on multimodal measurements: A review. J. Biomed. Inform. 2016;59:49–75. doi: 10.1016/j.jbi.2015.11.007. [PubMed] [CrossRef] [GoogleScholar]

[12] Vanitha V., Krishnan P. Real Time Stress Detection System Based on EEG Signals. [(accessed on 18 April 2019)];Biomed. Res. 2016 :S271–S275. Available online: http://www.biomedres.info/biomedical-research/real-time-stress-detection-system-based-on-eeg-signals.html. [Google Scholar]

[13] Costin R., Rotariu C., Pasarica A. Mental stress detection using heart rate variability and morphologic variability of EeG signals; Proceedings of the 2012 International Conference and Exposition on Electrical and Power Engineering; Iaşi, Romania. 25–27 October 2012; pp. 591–596. [Google Scholar]

[14] De Santos Sierra A., Avila C.S., Casanova J.G., del Pozo G.B. A Stress-Detection System Based on Physiological Signals and Fuzzy Logic. IEEE Trans. Ind. Electron. 2011;58:4857–4865. doi: 10.1109/TIE.2010.2103538. [CrossRef] [Google Scholar]

[15] Soury M., Devillers L. Stress Detection from Audio on Multiple Window Analysis Size in a Public Speaking Task; Proceedings of the 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction; Geneva, Switzerland. 2–5 September 2013; pp. 529–533. [Google Scholar]

[16] Wijsman J., Grundlehner B., Liu H., Hermens H., Penders J. Towards mental stress detection using wearablephysiological sensors; Proceedings of the 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society; Boston, MA, USA. 30 August–3 September 2011; pp. 1798–1801. [PubMed] [Google Scholar]

[17] Abouelenien M., Burzo M., Mihalcea R. Human Acute Stress Detection via Integration of Physiological Signals and Thermal Imaging; Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments; Corfu Island, Greece. 29 June–1 July 2016; New York, NY, USA: ACM; 2016. pp. 32:1–32:8. [Google Scholar]

Certificate of Publication

This is to confirm that

D.Sandhya Rani ,Ch.Geethanjali ,G.Anjana, R.Abhishek

Published following article

Stress Detection for Wearable Devices

Volume 4, Issue 6, pp: 131-135

www.ijaem.net

A Peer Reviewed Journal

International journal of Advances in Engineering and Management (IJAEM)

ISSN: 2395-5252

Publication Head