<Title: - DETECTION OF STRESS IN IT EMPLOYEES AND STUDENTS USING MACHINE LEARNING TECHNIQUES>

A Project Synopsis

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE WITH SPECIALIZATION IN

INFORMATION SECURITY AIT APEX

Submitted by:

<Student UID: - 22BIS80001> <Student Name: - Amritpal Kaur Dhillon>

Under the Supervision of:

<Project Supervisor Name: - Geeta Rani Mam>

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,
PUNJAB

January 2025

ABSTRACT

BASE PAPER ABSTRACT

The objective of this paper is to apply machine learning and visual processing to identify overworked IT employees. Our technology is an improved version of older stress detection systems that did not include live detection or personal counseling. Stress detection methods that don't include real-time monitoring or individual counseling are being updated in this research. A survey is used to collect data on employees' mental stress levels to provide effective stress management solutions. To get the most out of your employees, this paper will look at stress management and how to create a healthy, spontaneous work environment.

OUR PROPOSED ABSTRACT

In the modern world with the latest technology gadgets, Stress is rising most for everyone. Because of this, despite their affluence, people are not satisfied. A pressured feeling is stress. Pressure may be mental, emotional, or even physical. Systems for managing stress are essential for identifying the stress levels that disturb our socioeconomic way of life. According to the World Health Organization (WHO), one in four people suffer from the mental health issue of stress. Human stress causes mental and socioeconomic issues, loss of focus at work, strained relationships with coworkers, despair, and in the worst circumstances, suicide. This requires the provision of counseling to help those under stress manage their stress. While it is impossible to completely avoid stress, taking preventive measures can help you manage it. Only medical and physiological professionals can now assess whether a person is depressed or stressed. A questionnaire-based approach is one of the more established ways to identify stress. Our project's primary goal is to identify signs of stress in IT professionals utilizing sophisticated machine learning and image processing

methods. Our technology is an improved version of the previous stress detection technologies, which did not take into account the employee's emotions or live detection. However, this system includes both periodic and live employee emotion detection. Automatic detection of stress minimizes the risk of health issues and improves the welfare of the IT employee and the company. Knowing the IT employee's emotions allows the business to provide the right guidance and obtain better results from them. The accuracy of our suggested system model, which is developed using CNN Model Architecture, is 87.34% during training and 98.45% during validation.

INDEX

INTRODUCTION

- **Deep Learning:** Deep learning allows computational models composed of multiple processing layers to learn representations of data with various levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection, and many other domains, such as drug discovery and genomics. Deep learning discovers intricate structures in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer.
- **Supervised learning:** The most common form of machine learning, deep or not, is supervised learning. Imagine that we want to build a system that can classify images as containing, say, a house, a car, a person, or a pet. We first collect a large data set of images of houses, cars, people, and pets, each labeled with its category. During training, the machine is shown an image and produces an output in the form of a vector of scores, one for each category. We want the desired category to have the highest score of all categories, but this is unlikely to happen before training.
- **Backpropagation to train multilayer architectures:** From the earliest days of pattern recognition, researchers have aimed to replace hand-engineered features with trainable multilayer networks, but despite its simplicity, the solution was not widely understood until the mid-1980s. As it turns out, multilayer architectures can be trained by simple stochastic gradient descent.
- **Convolutional neural networks:** ConvNets are designed to process data that come in the form of multiple arrays, for example, a color image composed of three 2D arrays containing

pixel intensities in the three-color channels. Many data modalities are in the form of multiple arrays: 1D for signals and sequences, including language; 2D for images or audio spectrograms; and 3D for video or volumetric images.

Image understanding with deep convolutional networks

- Since the early 2000s, ConvNets have been applied with great success to the detection, segmentation, and recognition of objects and regions in images. These were all tasks in which labelled data was relatively abundant, such as traffic sign recognition, the segmentation of biological images, particularly for connection to mics, and the detection of faces, text, pedestrians, and human bodies in natural images. In the first layer, each word creates a different pattern of activations or word vectors. In a language model, the other layers of the network learn to convert the input word vectors into an output word vector for the predicted next word, which can be used to predict the probability for any word in the vocabulary to appear as the next word.

Recurrent neural networks

- When backpropagation was first introduced, its most exciting use was for training recurrent neural networks (RNNs). For tasks that involve sequential inputs, such as speech and language, it is often better to use RNNs. RNNs process an input sequence one element at a time, maintaining in their hidden units a ‘state vector’ that implicitly contains information about the history of all the past elements of the sequence.
- Memory networks can be trained to keep track of the state of the world in a setting similar to a text adventure game, and after reading a story, they can answer questions that require complex inference. In one test example, the network is shown a 15-sentence version of the Lord of the Rings and correctly answers questions such as “Where is Frodo now?”.

The future of deep learning

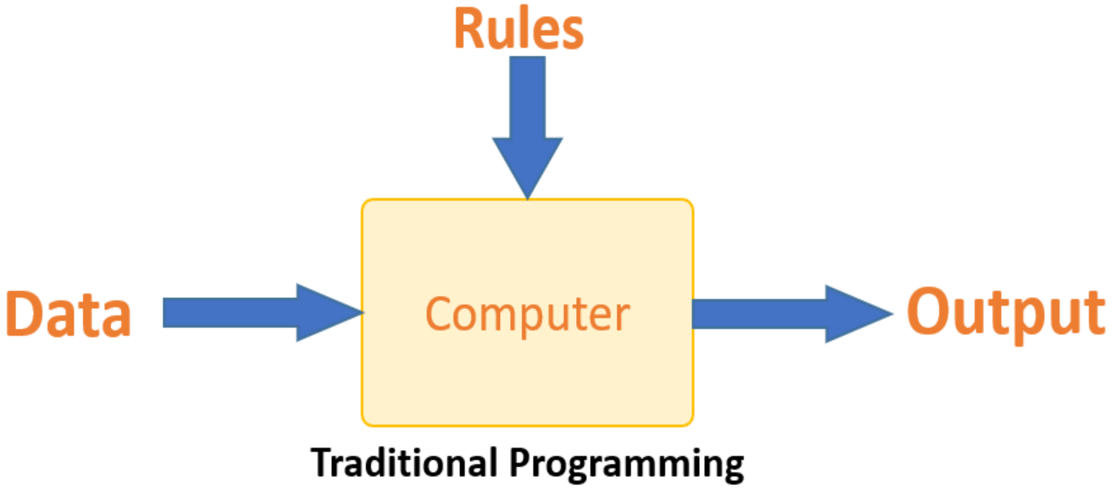
- Unsupervised learning had a catalytic effect in reviving interest in deep learning but has since been overshadowed by the successes of purely supervised learning. Although we have not focused on it in this Review, we expect unsupervised learning to become far more important in the longer term. Human and animal learning is largely unsupervised: we discover the structure of the world by observing it, not by being told the name of every object. Natural language understanding is another area in which deep learning is poised to make a large impact over the next few years. We expect systems that use RNNs to understand sentences or whole documents will become much better when they learn strategies for selectively attending to one part at a time.

❖ What is Machine Learning?

❖ Machine Learning is a system of computer algorithms that can learn from examples through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial intelligence that combines data with statistical tools to predict an output that can be used to make actionable insights. Machine learning is also used for a variety of tasks like fraud detection, predictive maintenance, portfolio optimization, automated tasks, and so on.

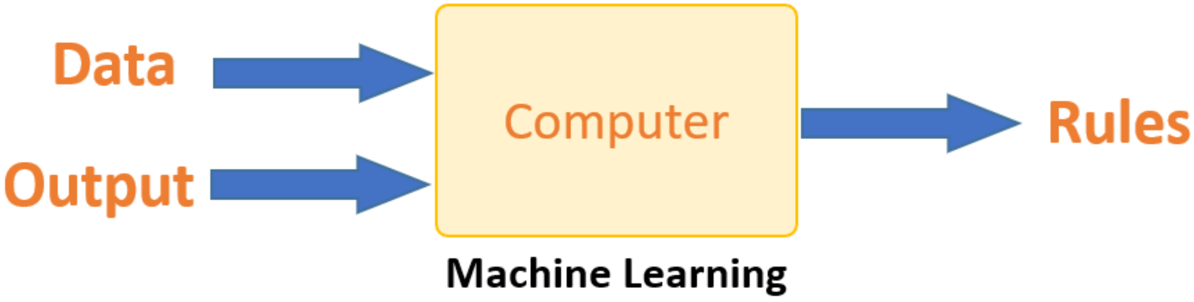
Machine Learning vs. Traditional Programming

❖ Traditional programming differs significantly from machine learning. In traditional programming, a programmer codes all the rules in consultation with an expert in the industry for which the software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain. Traditional programming differs significantly from machine learning. In traditional programming, a programmer codes all the rules in consultation with an expert in the industry for which the software is being developed.



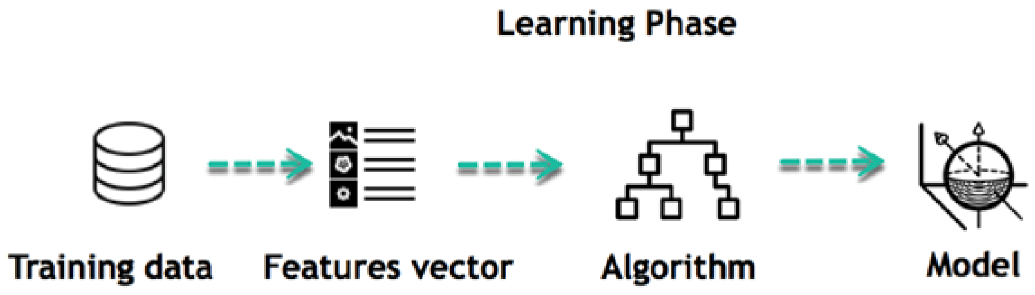
Traditional Programming

❖ Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated, and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.



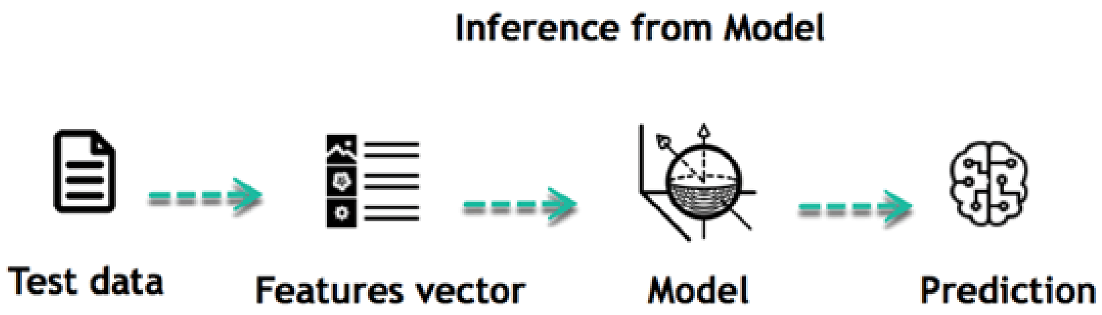
How does Machine Learning Work?

- Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if it feeds a previously unseen example, the machine has difficulties predicting. The machine uses some fancy algorithms to simplify reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.



Inferring

- When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model, and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or retrain the model. You can use the previously trained model to make inferences about new data.

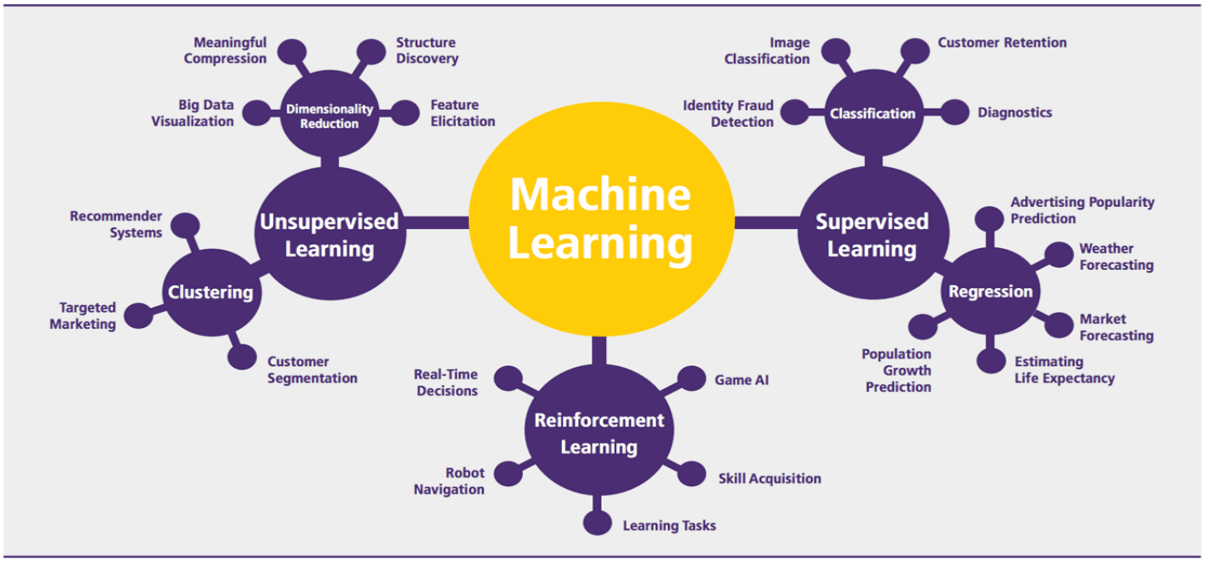


❖ The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

❖ Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data.

Machine Learning Algorithms and Where They Are Used?



Machine learning Algorithms

- Machine learning can be grouped into two broad learning tasks: Supervised and Unsupervised. There are many other algorithms

Supervised learning

- An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expenses and weather forecasts as input data to predict the sales of cans.
- You can use supervised learning when the output data is known. The algorithm will predict new data.
- There are two categories of supervised learning:
- Classification task
- Regression task

Classification

- Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc., from your customer database. You know the gender of each of your customers; it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected).

Regression

- When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of features like equity, previous stock performances, and macroeconomic indices. The system will be trained to estimate the price of the stocks with the lowest possible error.

Algorithm Name	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. Th output variable 3 is binary (e.g., only black or white) rather tha continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	A highly interpretable classification or regression model that splits dat feature values into branches at decision nodes (e.g., if a feature is color, each possible color becomes a new branch) until a final decisio output is made	Regression Classification
Naive Bayes	The Bayesian method is a classification method that makes use of th Bayesian theorem. The theorem updates the prior knowledge of a event with the independent probability of each feature that can affec the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for th classification task. The SVM algorithm finds a hyperplane tha optimally divides the classes. It is best used with a non-linear solver.	Regression (nc very commor Classification
Random forest	The algorithm is built upon a decision tree to improve the accurac drastically. Random forest generates many times simple decision tree and uses the 'majority vote' method to decide on which label to retur For the classification task, the final prediction will be the one with th most votes, while for the regression task, the average prediction of a the trees is the final prediction.	Regression Classification
AdaBoost	Classification or regression technique that uses a multitude of model to come up with a decision but weighs them based on their accuracy i predicting the outcome	Regression Classification

Algorithm Name	Description	Type
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regressio technique. It is focusing on the error committed by the previous tree and tries to correct it.	Regression Classification

Unsupervised learning

- ❖ In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns)

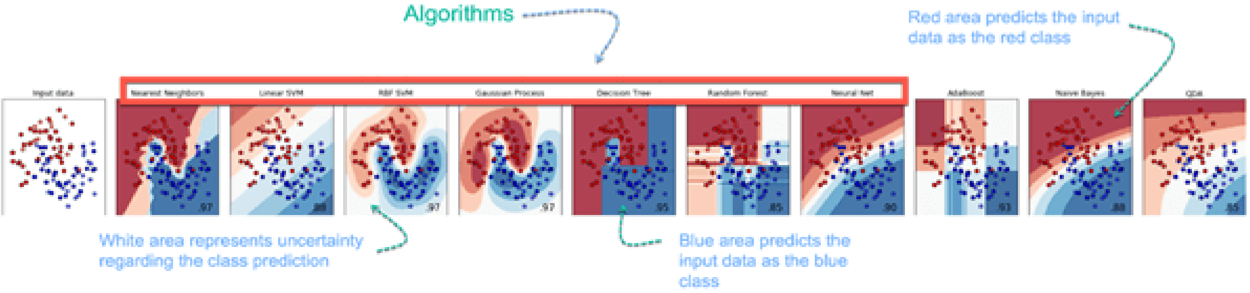
Algorithm	Description	Type
K-means clustering	Puts data into some groups (k) that each contain data with similar characteristics (as determined by the model, not in advance by humans)	Clustering
Gaussian mixture model	A generalization of k-means clustering that provides more flexibility in the size and shape of groups (clusters)	Clustering
Hierarchical clustering	Splits clusters along a hierarchical tree to form a classification system. Can be used for Cluster loyalty-card customer	Clustering
Recommender system	Help to define the relevant data for making a recommendation.	Clustering

Algorithm	Description	Type
PCA/T-SNE	Mostly used to decrease the dimensionality of the data. The algorithms reduce the number of features to 3 or 4 vectors with the highest variances.	Dimension Reduction

How to Choose a Machine Learning Algorithm

Machine Learning (ML) algorithm:

❖ There are plenty of machine learning algorithms. The choice of the algorithm is based on the objective. In the Machine learning example below, the task is to predict the type of flower among the three varieties. The predictions are based on the length and the width of the petal. The picture depicts the results of ten different algorithms. The picture on the top left is the dataset. The data is classified into three categories: red, light blue, and dark blue. There are some groupings.



Challenges and Limitations of Machine Learning

❖ The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight.

Application of Machine Learning

Augmentation:

❖ Machine learning assists humans with their day-to-day tasks, personally or commercially, without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistants, Data analysis, and software solutions. The primary user is to reduce errors due to human bias.

Automation:

- Machine learning works entirely autonomously in any field without the need for any human intervention. For example, robots perform the essential process steps in manufacturing plants.

Finance Industry

- Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization

- The government makes use of ML to manage public safety and utilities. Take the example of China with its massive face recognition. The government uses Artificial intelligence to prevent jaywalking.

Healthcare industry

- Healthcare was one of the first industries to use machine learning with image detection.

Marketing

- Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers developed advanced mathematical tools like Bayesian analysis to estimate the value of a customer.

Example of application of Machine Learning in Supply Chain

- Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network. For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report, and make recommendations in real time.

Example of Machine Learning Google Car

- For example, everybody knows the Google car. The car is full of lasers on the roof, which are telling it where it is regarding the surrounding area. It has radar in the front, which informs the car of the speed and motion of all the cars around it.

Why is Machine Learning Important?

- Machine learning is the best tool so far to analyze, understand, and identify a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can make decisions with minimal human intervention. Take the following example for this ML tutorial: A retail agent can estimate the price of a house based on his own experience and his knowledge of the market.

Overview

- Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. In practice, it can turn out to be more effective to help the machine develop its algorithm rather than have human programmers specify every needed step.

Machine learning approaches

- Machine learning approaches are traditionally divided into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system:
- **Supervised learning:** The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.
- **Unsupervised learning:** No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means toward an end (feature learning).
- **Reinforcement learning:** A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program provides feedback that's analogous to rewards, which it tries to maximize.

History and relationships to other fields

- The term machine learning was coined in 1959 by Arthur Samuel, an American IBMer and pioneer in the field of computer gaming and artificial intelligence. A representative book of machine learning research during the 1960s was Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. In 1981, a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal.

Artificial intelligence

- Machine Learning as a subfield of AI. Part of Machine Learning as a subfield of AI or part of AI as a subfield of Machine Learning. As a scientific endeavour, machine learning grew out of the quest for artificial intelligence. In the early days of AI as an academic discipline, some researchers were interested in having machines learn from data. This line, too, was continued outside the AI/CS field as "connectionism" by researchers from other disciplines, including Hopfield, Rumelhart, and Hinton. Their main success came in the mid-1980s with the reinvention of backpropagation. As of 2020, many sources continue to assert that machine learning remains a subfield of AI. The main disagreement is whether all of ML is part of AI, as this would mean that anyone using ML could claim they are using AI. Others have the view that not all of ML is part of AI, where only an 'intelligent' subset of ML is part of AI.

Data mining

- Machine learning and data mining often employ the same methods and overlap significantly, but while machine learning focuses on prediction, based on known properties learned from the training data, data mining focuses on the discovery of (previously) unknown properties in the data (this is the analysis step of knowledge discovery in databases). Data mining uses many machine learning methods but with different goals; on the other hand, machine learning also employs data mining methods as "unsupervised learning" or as a preprocessing step to improve learner accuracy.

Optimization

- Machine learning also has intimate ties to optimization: Many learning problems are formulated as the minimization of some loss function on a training set of examples. Loss functions express the discrepancy between the predictions of the model being trained and the actual problem instances (for example, in classification, one wants to assign a label to instances, and models are trained to correctly predict the pre-assigned labels of a set of examples).

Generalization

- The difference between optimization and machine learning arises from the goal of generalization: While optimization algorithms can minimize the loss on a training set, machine learning is concerned with minimizing the loss on unseen samples. Characterizing the generalization of various learning algorithms is an active topic of current research, especially for deep learning algorithms.

Statistics

- Machine learning and statistics are closely related fields in terms of methods, but they are distinct in their principal goal: statistics draws population inferences from a sample, while machine learning finds generalizable predictive patterns. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long pre-history in statistics. He also suggested the term data science as a placeholder to call the overall field.

Theory

- A core objective of a learner is to generalize from their experience. Generalization in this context is the ability of a learning machine to perform accurately on new, unseen examples/tasks after having experienced a learning data set. In addition to performance bounds, learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned in polynomial time. Negative results show that certain classes cannot be learned in polynomial time.

Approaches

Types of learning algorithms

- The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

Supervised learning

- A support vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white. Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal.

Unsupervised learning

- Unsupervised learning algorithms take a set of data that contains only inputs and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labelled, classified, or categorized.
- Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar.

Semi-supervised learning

- Semi-supervised learning falls between unsupervised learning (without any labelled training data) and supervised learning (with completely labelled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabelled data, when used in conjunction with a small amount of labelled data, can produce a considerable improvement in learning accuracy.

Reinforcement learning

- Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics, and genetic algorithms.

Self-learning

- Self-learning as a machine learning paradigm was introduced in 1982, along with a neural network capable of self-learning named crossbar adaptive array (CAA). It is a learning with no external rewards and no external teacher advice.
- In situation s act a;
- Receive consequence situation s’;
- Compute emotion of being, in consequence, situation v(s’);
- Update crossbar memory w’ (a, s) = w (a, s) + v(s’).
- It is a system with only one input, situations, and only one output, action (or behavior) a. There is neither a separate reinforcement input nor an advice input from the environment.

The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The

Feature learning

- Several learning algorithms aim at discovering better representations of the inputs provided during training. Classic examples include principal components analysis and cluster analysis. Feature learning algorithms, also called representation learning algorithms, often attempt to preserve the information in their input but also transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions.

Sparse dictionary learning

- Sparse dictionary learning is a feature learning method where a training example is represented as a linear combination of basic functions and is assumed to be a sparse matrix. The method is strongly NP-hard and difficult to solve approximately. A popular heuristic method for sparse dictionary learning is the K-SVD algorithm. Sparse dictionary learning has been applied in several contexts.

Anomaly detection

- In data mining, anomaly detection, also known as outlier detection, is the identification of rare items, events, or observations that raise suspicions by differing significantly from the majority of the data. Typically, the anomalous items represent an issue such as bank fraud, a structural defect, medical problems, or errors in a text. Anomalies are referred to as outliers, novelties, noise, deviations, and exceptions. Semi-supervised anomaly detection techniques construct a model representing normal behavior from a given normal training data set and then test the likelihood of a test instance to be generated by the model.

Robot learning

- In developmental robotics, robot learning algorithms generate their sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies, and imitation.

Association rules

- Association rule learning is a rule-based machine learning method for discovering relationships between variables in large databases. It is intended to identify strong rules discovered in databases using some measure of "interestingness". Rule-based machine learning approaches include learning classifier systems, association rule learning, and artificial immune systems.

Models

- ❖ Performing machine learning involves creating a model that is trained on some training data and then can process additional data to make predictions. Various types of models have been used and researched for machine learning systems.

Artificial neural networks

- ❖ An artificial neural network is an interconnected group of nodes akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron, and an arrow represents a connection from the output of one artificial neuron to the input of another.

Decision trees

- ❖ Decision tree learning uses a decision tree as a predictive model to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modeling approaches used in statistics, data mining, and machine learning.

Support vector machines

- ❖ Support vector machines (SVMs), also known as support vector networks, are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other.

Regression analysis

- ❖ Regression analysis encompasses a large variety of statistical methods to estimate the relationship between input variables and their associated features. Its most common form is linear regression, where a single line is drawn to best fit the given data according to a mathematical criterion such as ordinary least squares.

Bayesian networks

- ❖ A simple Bayesian network. Rain influences whether the sprinkler is activated, and both rain and the sprinkler influence whether the grass is wet. A Bayesian network, belief network, or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independence with a directed acyclic graph (DAG).

Genetic algorithms

- A genetic algorithm (GA) is a search algorithm and heuristic technique that mimics the process of natural selection, using methods such as mutation and crossover to generate new genotypes in the hope of finding good solutions to a given problem.

Training models

- Usually, machine learning models require a lot of data for them to perform well. Usually, when training a machine learning model, one needs to collect a large, representative sample of data from a training set. Data from the training set can be as varied as a corpus of text, a collection of images, and data collected from individual users of a service. Overfitting is something to watch out for when training a machine learning model.

Federated learning

- Federated learning is an adapted form of distributed artificial intelligence for training machine learning models that decentralizes the training process, allowing for users' privacy to be maintained by not needing to send their data to a centralized server.

Detection of Stress in IT Employees And Students Using Machine Learning Techniques

BASE PAPER ABSTRACT

- The objective of this paper is to apply machine learning and visual processing to identify overworked IT employees.
- Our technology is an improved version of older stress detection systems that did not include live detection or personal counseling.

OUR PROPOSED ABSTRACT

- In the modern world with the latest technology gadgets, Stress is rising most for everyone.
- Because of this, despite their affluence, people are not satisfied.
- A pressured feeling is stress.
- Pressure may be mental, emotional, or even physical.

- Systems for managing stress are essential for identifying the stress levels that disturb our socioeconomic way of life.
- Only medical and physiological professionals can now assess whether a person is depressed or stressed.
- A questionnaire-based approach is one of the more established ways to identify stress.
- Our project's primary goal is to identify signs of stress in IT professionals utilizing sophisticated machine learning and image processing methods.
- Our technology is an improved version of the previous stress detection technologies, which did not take into account the employee's emotions or live detection.

EXISTING SYSTEM:

- In the existing system work on stress detection is based on digital signal processing, taking into consideration Galvanic skin response, blood volume, pupil dilation, and skin temperature.
- The other work on this issue is based on several physiological signals and visual features (eye closure, head movement) to monitor the stress in a person while he is working.
- However, these measurements are intrusive and are less comfortable in real applications.

DISADVANTAGES OF THE EXISTING SYSTEM:

- Different people may behave or express differently under stress and it is hard to find a universal pattern to define the stress emotion.
- The existing system model does not work well with large dataset as calculating distances between each data instance would be very costly.
- The existing system model does not work well with high dimensionality as this will complicate the distance calculating process to calculate distance for each dimension.

PROPOSED SYSTEM:

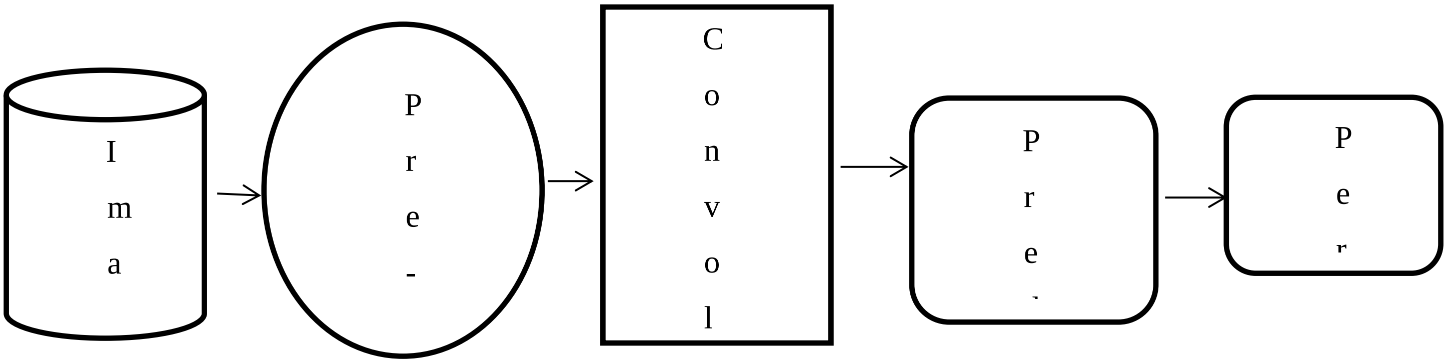
- Our proposed system is an updated version of prior stress detection systems that did not include live analysis or individual employee emotional analysis.
- Our proposed system model is developed using the CNN Model Architecture.
- By taking a picture as input and returning characteristics associated with those images as output.

- We use a bounded box to show the employee's feelings, and also the emotions are shown at the top of the bounded box.
- The proposed system uses emotional classes such as: Angry, Disgusted, Fearful, Happy, Neutral, Sad, and Surprised to detect the emotion of the IT employee.
- To check the efficiency of the stability, we gathered information and executed a quantitative experimental investigation.
- Firstly, the User has to register with the required details user name, login ID, password, mobile, email, Locality, Address, city, state, etc.

ADVANTAGES OF PROPOSED SYSTEM:

- The Stress Detection System helps workers in managing difficulties that add to stress by giving proactive stress management solutions.
- In the proposed system the employee can also use a live cam to detect stress, which shows their emotional characteristics of them.
- Stress Detection System enables employees to cope with their issues leading to stress through preventative stress management solutions.
- The proposed system model has very High accuracy.

SYSTEM ARCHITECTURE:



SYSTEM REQUIREMENTS:

HARDWARE REQUIREMENTS:

- System: Pentium i3 Processor.
- Hard Disk : 500 GB.
- Monitor: 15’’ LED
- Input Devices: Keyboard, Mouse
- Ram: 4 GB

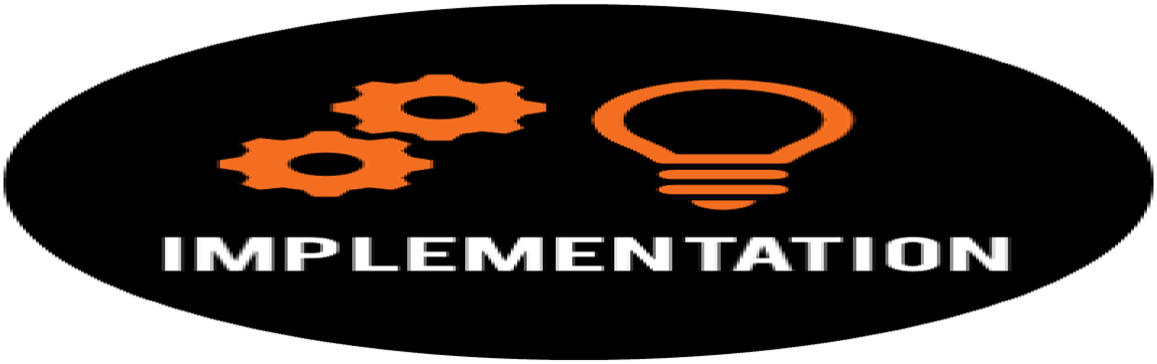
SOFTWARE REQUIREMENTS:

- Operating system: Windows 10.
- Coding Language: Python 3.8
- Web Framework: Flask
- Database: MYSQL.

REFERENCE:

Suresh Kumar Kanaparthi; Surekha P; Lakshmi Priya Bellamkonda; Bhavya Kadiam; Beulah Mungara, “Detection of Stress in IT Employees using Machine Learning Technique”, 2022 International Conference on Applied Artificial Intelligence and Computing (ICAAIC), IEEE Conference, 2022.

IMPLEMENTATION



MODULES:

- Dataset

- Importing the necessary libraries
- Retrieving the images
- Splitting the dataset
- Building the model
- Apply the model and plot the graphs for accuracy and loss
- Accuracy on test set
- Saving the Trained Model
- Database connecting using MySQL

MODULES DESCRIPTION:

- **Dataset: -** In the first module, we developed the system to get the input dataset for training and testing purposes. The dataset is given in the model folder. The dataset consists of 49,543 Facial Expression images. The dataset is referred from the Kaggle website. The link to the dataset is given below.
- Kaggle link: https://www.kaggle.com/datasets/jayaprakashpondy/emotion-dataset
- **Importing the necessary libraries: -** We will be using Python language for this. First, we will import the necessary libraries such as keras for building the main model, sklearn for splitting the training and test data, PIL for converting the images into an array of numbers, and other libraries such as pandas, numpy, matplotlib, and TensorFlow.

- **Retrieving the images: -** We will retrieve the images and their labels. Then resize the images to (48,48) as all images should have the same size for recognition. Then convert the images into a NumPy array.

- **Splitting the dataset: -** Split the dataset into train and test. 80% train data and 20% test data.

✹ Convolutional Neural Networks

The objectives are:

- To understand the convolution operation
- To understand the pooling operation

✹ Computer Vision

Some of the computer vision problems which we will be solving in this article are:

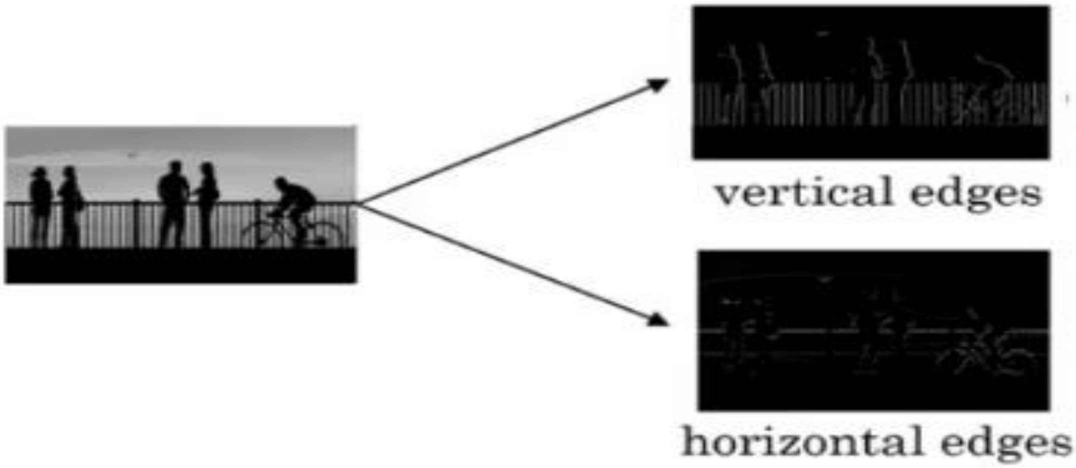
- Image classification
- Object detection
- Neural style transfer

ℵ One major problem with computer vision problems is that the input data can get big. Suppose an image is of the size 68 X 68 X 3. The input feature dimension then becomes 12,288. This will be even bigger if we have larger images (say, of size 720 X 720 X 3).

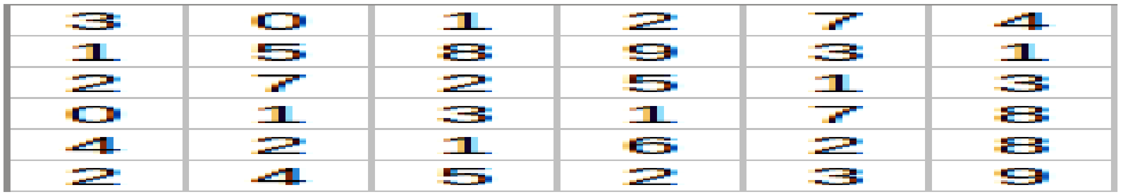
Edge Detection Example

- In the previous article, we saw that the early layers of a neural network detect edges from an image. Deeper layers might be able to detect the cause of the objects and even more deeper layers might detect the cause of complete objects (like a person's face).

- **As you can see, there are many vertical and horizontal edges in the image.**
- **The first thing to do is to detect these edges:**

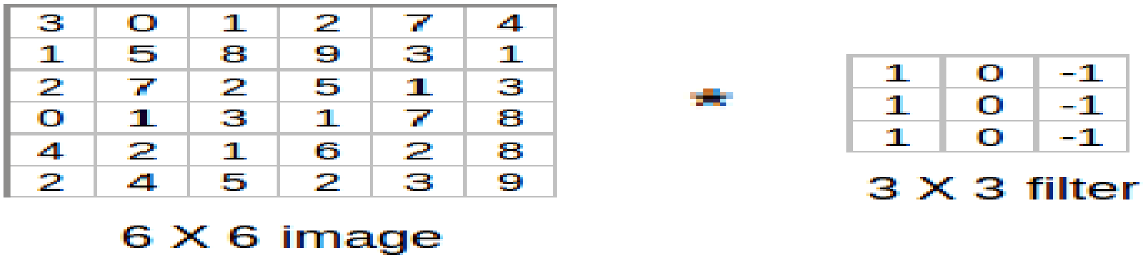


- But how do we detect these edges? To illustrate this, let's take a 6 X 6 grayscale image (i.e. only one channel):



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

- ❖ Next, we convolve this 6 X 6 matrix with a 3 X 3 filter:



3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

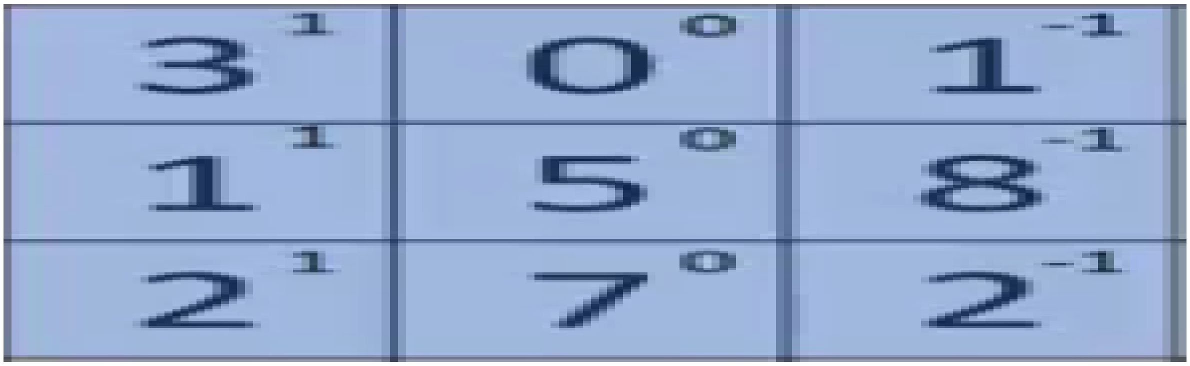
6 X 6 image

*

1	0	-1
1	0	-1
1	0	-1

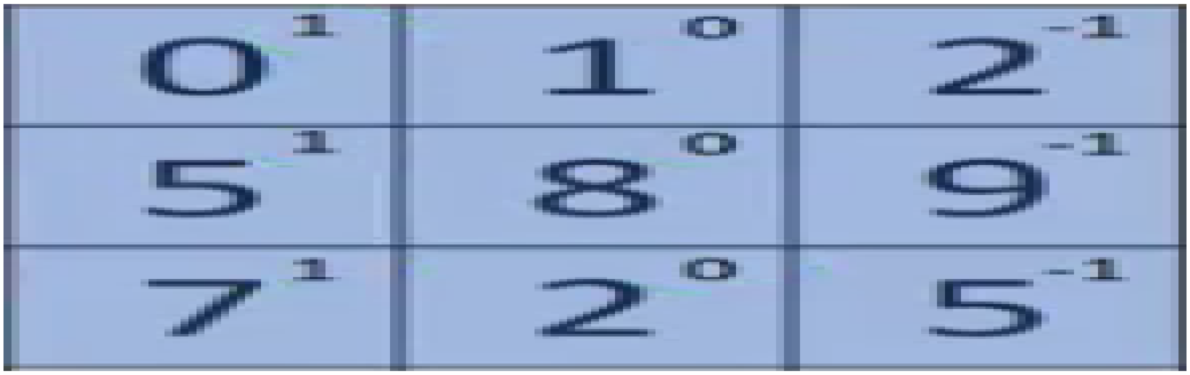
3 X 3 filter

- ❖ After the convolution, we will get a 4 X 4 image. The first element of the 4 X 4 matrix will be calculated as:



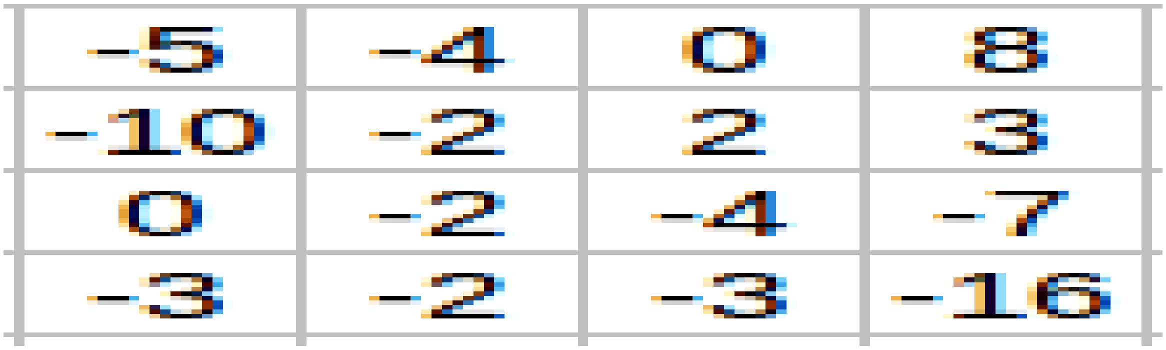
3^{1}	0^{0}	1^{-1}
1^{1}	5^{0}	8^{-1}
2^{1}	7^{0}	2^{-1}

- ❖ So, we take the first 3 X 3 matrix from the 6 X 6 image and multiply it with the filter. Now, the first element of the 4 X 4 output will be the sum of the element-wise product of these values, i.e. 3*1 + 0 + 1*-1 + 1*1 + 5*0 + 8*-1 + 2*1 + 7*0 + 2*-1 = -5.
- ❖ To calculate the second element of the 4 X 4 output, we will shift our filter one step towards the right and again get the sum of the element-wise product:



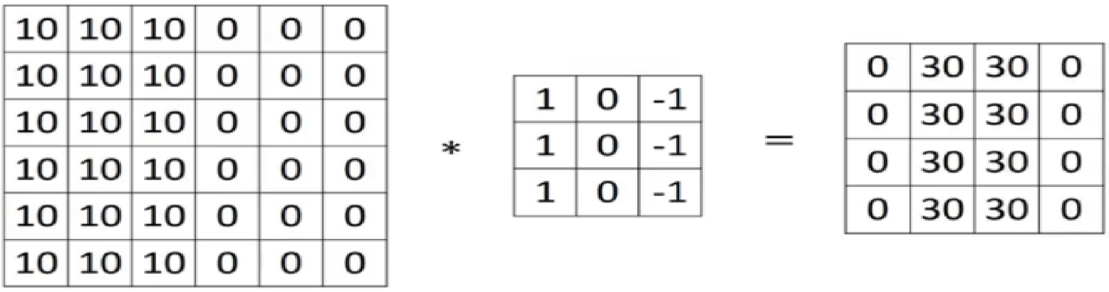
0^{1}	1^{0}	2^{-1}
5^{1}	8^{0}	9^{-1}
7^{1}	2^{0}	5^{-1}

- ❖ Similarly, we will convolve over the entire image and get a 4 X 4 output:

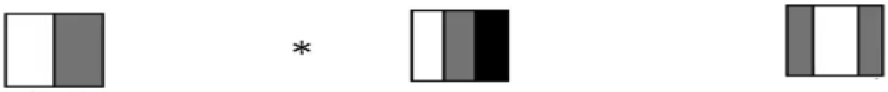


-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

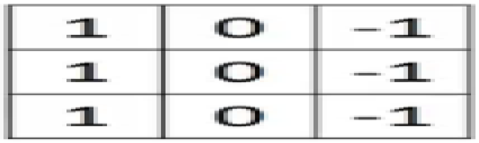
- So, convolving a 6 X 6 input with a 3 X 3 filter gave us an output of 4 X 4. Consider one more example:



$$\begin{array}{|c|c|c|c|c|c|}\hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline 10 & 10 & 10 & 0 & 0 & 0 \\ \hline \end{array} * \begin{array}{|c|c|c|}\hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline 1 & 0 & -1 \\ \hline \end{array} = \begin{array}{|c|c|c|c|}\hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline 0 & 30 & 30 & 0 \\ \hline \end{array}$$

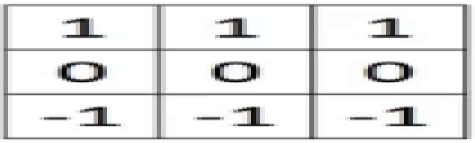


- **Note**: Higher pixel values represent the brighter portion of the image and the lower pixel values represent the darker portions. This is how we can detect a vertical edge in an image.
- **More Edge Detection: -** The type of filter that we choose helps to detect the vertical or horizontal edges. We can use the following filters to detect different edges:
- Some of the commonly used filters are:



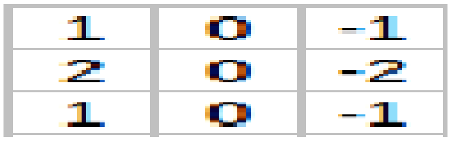
1	0	-1
1	0	-1
1	0	-1

Vertical



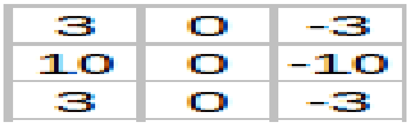
1	1	1
0	0	0
-1	-1	-1

Horizontal



1	0	-1
2	0	-2
1	0	-1

Sobel filter



3	0	-3
10	0	-10
3	0	-3

Scharr filter

❖ The Sobel filter puts a little bit more weight on the central pixels. Instead of using these filters, we can create our own as well and treat them as a parameter which the model will learn using backpropagation.

❖ **Padding: -** We have seen that convolving an input of 6 X 6 dimension with a 3 X 3 filter results in 4 X 4 output. We can generalize it and say that if the input is n X n and the filter size is f X f, then the output size will be (n-f+1) X (n-f+1):

❖ **Input:** n X n

❖ **Filter size:** f X f

❖ **Output:** (n-f+1) X (n-f+1)

♣ **There are primarily two disadvantages here:**

    ☹ Every time we apply a convolutional operation, the size of the image shrinks

    ☹ Pixels present in the corner of the image are used only a few times during convolution as compared to the central pixels. Hence, we do not focus too much on the corners since that can lead to information loss

* To overcome these issues, we can pad the image with an additional border, i.e., we add one pixel all around the edges. This means that the input will be an 8 X 8 matrix (instead of a 6 X 6 matrix). Applying convolution of 3 X 3 on it will result in a 6 X 6 matrix which is the original shape of the image. This is where padding comes to the fore:

⌘ **Input:** n X n

⌘ **Padding:** p

⌘ **Filter size:** f X f

⌘ **Output:** (n+2p-f+1) X (n+2p-f+1)

♣ **There are two common choices for padding:**

♣ **Valid:** It means no padding. If we are using valid padding, the output will be (n-f+1) X (n-f+1)

♣ **Same:** Here, we apply padding so that the output size is the same as the input size, i.e., n+2p-f+1 = n. So, p = (f-1)/2

♣ We now know how to use padded convolution. This way we don't lose a lot of information and the image does not shrink either. Next, we will look at how to implement strided convolutions.

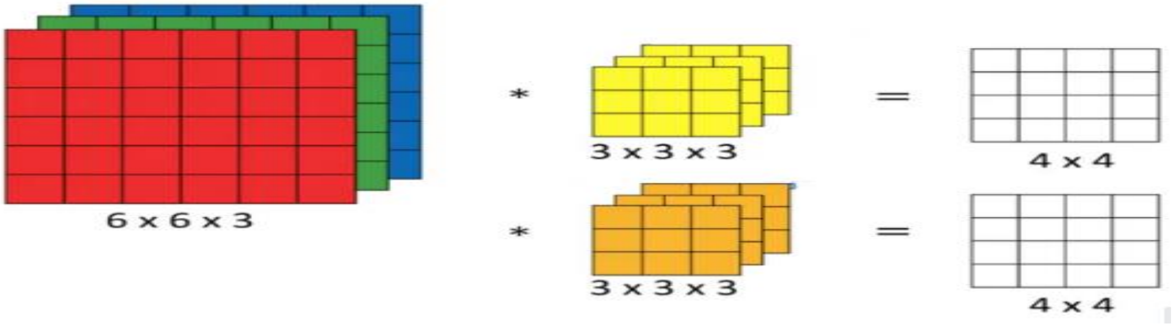
♣ **Strided Convolutions: -** Suppose we choose a stride of 2. So, while convoluting through the image, we will take two steps – both in the horizontal and vertical directions separately. The dimensions for strides will be:

♣ **Input:** n X n

♣ **Padding:** p

♣ **Stride:** s

- **Filter size:** f X f
- **Output:** [(n+2p-f)/s+1] X [(n+2p-f)/s+1]
- Stride helps to reduce the size of the image, a particularly useful feature.
- **Convolutions Over Volume: -** Suppose, instead of a 2-D image, we have a 3-D input image of shape 6 X 6 X 3. How will we apply convolution to this image? We will use a 3 X 3 X 3 filter instead of a 3 X 3 filter. Let's look at an example:
- **Input:** 6 X 6 X 3
- **Filter:** 3 X 3 X 3
- The dimensions above represent the height, width, and channels in the input and filter. Keep in mind that the number of channels in the input and filter should be the same**.** This will result in an output of 4 X 4. Let's understand it visually:
- Since there are three channels in the input, the filter will consequently also have three channels. After convolution, the output shape is a 4 X 4 matrix. So, the first element of the output is the sum of the element-wise product of the first 27 values from the input (9 values from each channel) and the 27 values from the filter. After that, we convolve over the entire image.
- Instead of using just a single filter, we can use multiple filters as well. How do we do that? Let's say the first filter will detect vertical edges and the second filter will detect horizontal edges from the image. If we use multiple filters, the output dimension will change. So, instead of having a 4 X 4 output as in the above example, we would have a 4 X 4 X 2 output (if we have used 2 filters):



- **<u>Generalized dimensions can be given as:</u>**

- **Input:** n X n X n_c
- **Filter:** f X f X n_c
- **Padding:** p
- **Stride:** s
- **Output:** [(n+2p-f)/s+1] X [(n+2p-f)/s+1] X n_c'
- Here, n_c is the number of channels in the input and filter, while n_c' is the number of filters.

- **One Layer of a Convolutional Network: -** Once we get an output after convolving over the entire image using a filter, we add a bias term to those outputs and finally apply an activation function to generate activations. *This is one layer of a convolutional network*. Recall that the equation for one forward pass is given by: $z^{[1]} = w^{[1]} * a^{[0]} + b^{[1]}$ $a^{[1]} = g(z^{[1]})$.
- In our case, input (6 X 6 X 3) is $a^{[0]}$and filters (3 X 3 X 3) are the weights $w^{[1]}$. These activations from layer 1 act as the input for layer 2, and so on. The number of parameters in the case of convolutional neural networks is independent of the size of the image. It essentially depends on the filter size. Suppose we have 10 filters, each of shape 3 X 3 X 3. What will be the number of parameters in that layer? Let's try to solve this:
- Number of parameters for each filter = 3*3*3 = 27
- There will be a bias term for each filter, so total parameters per filter = 28
- As there are 10 filters, the total parameters for that layer = 28*10 = 280
- No matter how big the image is, the parameters only depend on the filter size. Awesome, isn't it? Let's have a look at the summary of notations for a convolution layer:

¥ $f^{[l]}$ = filter size

¥ $p^{[l]}$ = padding

¥ $s^{[l]}$ = stride

¥ $n_{[c]}^{[l]}$ = number of filters

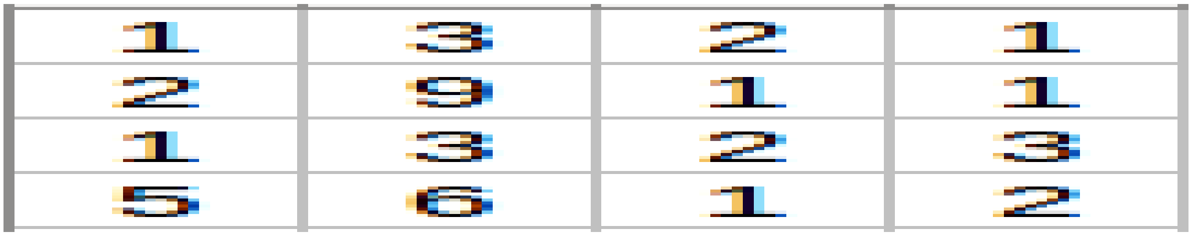
¥ Let's combine all the concepts we have learned so far and look at a convolutional network example.

➢ Simple Convolutional Network Example

- We'll take things up a notch now. Let's look at how a convolution neural network with convolutional and pooling layers works. Suppose we have an input of shape 32 X 32 X 3:
- We take an input image (size = 39 X 39 X 3 in our case), convolve it with 10 filters of size 3 X 3, and take the stride as 1 and no padding. This will give us an output of 37 X 37 X 10. We convolve this output further and get an output of 7 X 7 X 40 as shown above. Finally, we take all these numbers (7 X 7 X 40 = 1960), unroll them into a large vector, and pass them to a classifier that will make predictions. This is a microcosm of how a convolutional network works.
- There are several hyperparameters that we can tweak while building a convolutional network. These include the number of filters, size of filters, stride to be used, padding, etc. We will look at each of these in detail later in this article. Just keep in mind that as we go deeper into the network, the size of the image shrinks whereas the number of channels usually increases.
- In a convolutional network (ConvNet), there are three types of layers:
    - Convolution layer

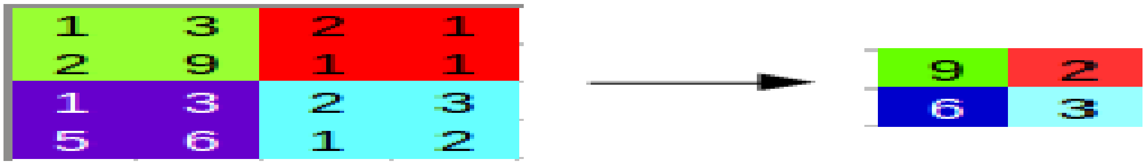
- Pooling layer
- Fully connected layer

- Let’s understand the pooling layer in the next section.
- **Pooling Layers: -** Pooling layers are generally used to reduce the size of the inputs and hence speed up the computation. Consider a 4 X 4 matrix as shown below:



1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

- Applying max pooling on this matrix will result in a 2 X 2 output:



1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

9	2
6	3

- For every consecutive 2 X 2 block, we take the max number. Here, we have applied a filter of size 2 and a stride of 2. These are the hyperparameters for the pooling layer. Apart from max pooling, we can also apply average pooling where, instead of taking the max of the numbers, we take their average. In summary, the hyperparameters for a pooling layer are:
- Filter size
- Stride
- Max or average pooling
- If the input of the pooling layer is n_h X n_w X n_c, then the output will be [{(n_h – f) / s + 1} X {(n_w – f) / s + 1} X n_c].

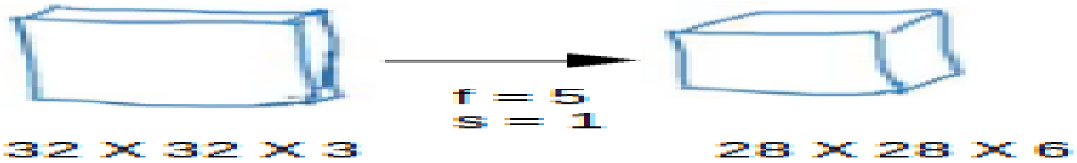
- **CNN Example: -** We’ll take things up a notch now. Let’s look at how a convolution neural network with convolutional and pooling layers works. Suppose we have an input of shape 32 X 32 X 3:
- There are a combination of convolution and pooling layers at the beginning, a few fully connected layers at the end, and finally a SoftMax classifier to classify the input into various categories. There are a lot of hyperparameters in this network which we have to specify as well.
- Generally, we take the set of hyperparameters that have been used in proven research and they end up doing well. As seen in the above example, the height and width of the input shrinks as we go deeper into the network (from 32 X 32 to 5 X 5), and the number of channels increases (from 3 to 10).

- All of these concepts and techniques bring up a very fundamental question – why convolutions? Why not something else?

Why Convolutions?

- There are primarily two major advantages of using convolutional layers over using just fully connected layers:
- Parameter sharing
- Sparsity of connections

Consider the below example:



- If we had used just the fully connected layer, the number of parameters would be = 32*32*3*28*28*6, which is nearly equal to 14 million! Makes no sense, right?
- If we see the number of parameters in the case of a convolutional layer, it will be = (5*5 + 1) * 6 (if there are 6 filters), which is equal to 156. Convolutional layers reduce the number of parameters and speed up the training of the model significantly.
- In convolutions, we share the parameters while convolving through the input. The intuition behind this is that a feature detector, which is helpful in one part of the image, is probably also useful in another part of the image. So, a single filter is convolved over the entire input and hence the parameters are shared.
- The second advantage of convolution is the sparsity of connections. For each layer, each output value depends on a small number of inputs, instead of taking into account all the inputs.

Building the model:

- For building the model we will use a sequential model from the keras library. Then we will add the layers to make a convolutional neural network. In the first 2 Conv2D layers we have used 32 filters and the kernel size is (5,5).
- In the MaxPool2D layer, we have kept the pool size (2,2) which means it will select the maximum value of every 2 x 2 area of the image. By doing this dimension of the image will be reduced by a factor of 2. In the dropout layer, we have kept the dropout rate = 0.25 which means 25% of neurons are removed randomly.
- We apply these 3 layers again with some changes in parameters. Then we apply a flattened layer to convert 2-D data to 1-D vector. This layer is followed by a dense layer, a dropout layer, and a dense layer again. The last dense layer outputs 7 nodes as the Facial Expression Detection. This layer uses

the SoftMax activation function which gives probability value and predicts which of the 7 options has the highest probability.

Apply the model and plot the graphs for accuracy and loss:

- ☯ We will compile the model and apply it using the fit function. The batch size will be 10. Then we will plot the graphs for accuracy and loss. We got the training Accuracy of 87.34%
- ☯ **Accuracy on test set: -** We got an accuracy of 98.45%. on the test set.

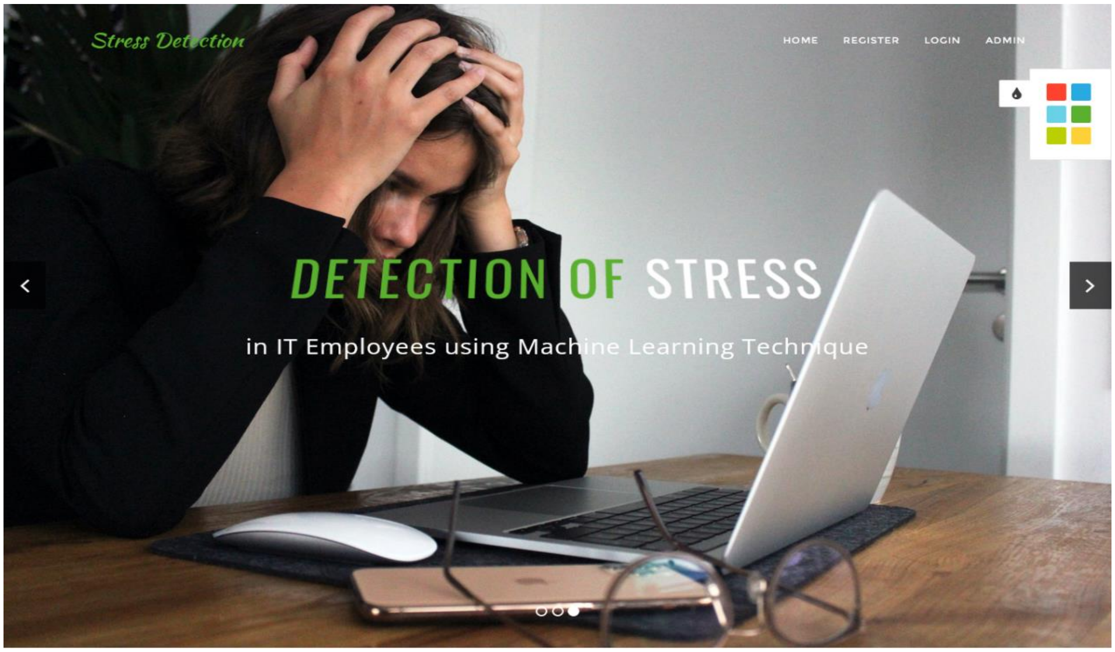
- ☯ **Saving the Trained Model: -** Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like Pickle. Make sure you have a pickle installed in your environment. Next, let's import the module and dump the model into the .pkl file

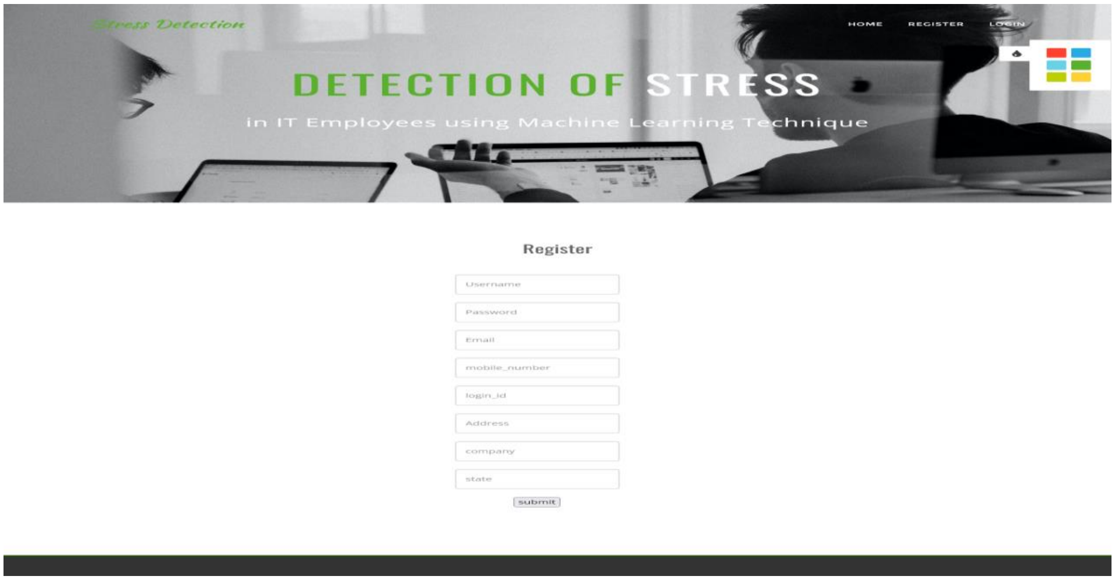
Import MYSQLdb

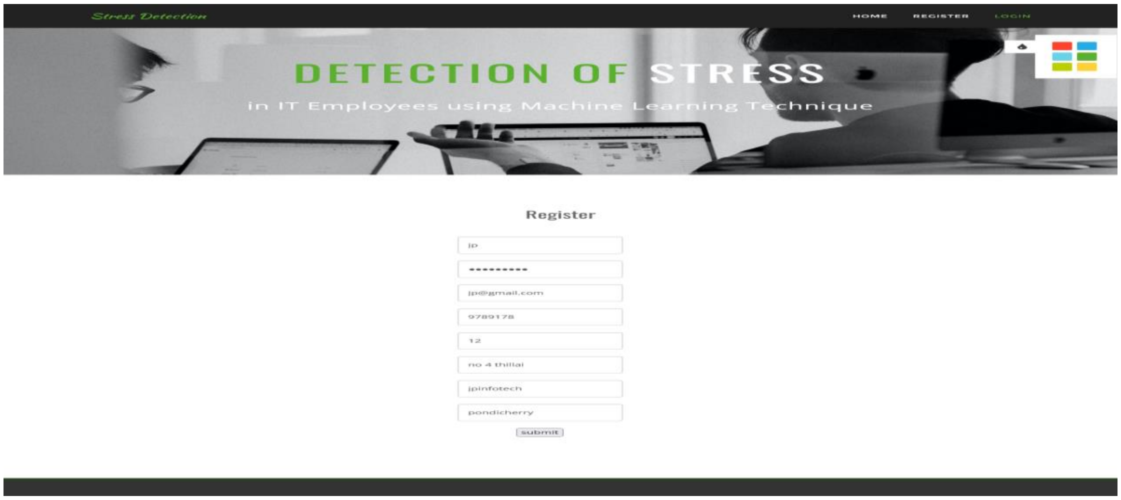
- ☯ So long as that works, do a quick control+d to exit the Python instance.
- ☯ Next, we want to make a Python file that can connect to the database. Generally, you will have a separate "connect" file, outside of any main files you may have.
- ☯ This is usually true across languages, and here's why. Initially, you may have just a simple __init__.py, or app.py, or whatever, and that file does all of your operations.
- ☯ What can happen in time, however, is that your website does other things. For example, with one of my websites, Sentdex.com, I perform a lot of analysis and store that analysis in a database, and I also operate a website for users to use. Generally, for tasks, you will use what is called a "cron."
- ☯ A cron is a scheduled task that runs when you program it to run. Generally, this runs another file, almost certain to not be your website's file. So then, to connect to a database, you'd have to write the database connecting code again in the file being run by your cron.
- ☯ As time goes on, these sorts of needs stack up where you have some files modifying the database, but you still want the website to be able to access it, and maybe modify it too.
- ☯ Then, consider what might happen if you change your database password. You'd then need to go to every single file that connects to the database and change that too. So, usually, you will find the smartest thing to do is to just create one file, which houses the connection code.
- ☯ **Import the module: -** Create a connection function to run our code. Here we specify where we're connecting to, the user, the user's password, and then the database that we want to connect to.
- ☯ As a note, we use "localhost" as our host. This just means we'll use the same server that this code is running on. You can connect to databases remotely as well, which can be pretty neat.
- ☯ To do that, you would connect to a host by their IP or their domain. To connect to a database remotely, you will need to first allow it from the remote database that will be accessed/modified.

- Next, let's go ahead and edit our __init__.py file, adding a register function. For now, we'll keep it simple, mostly just to test our connection functionality.
- We allow for GET and POST but aren't handling it just yet.
- We're going to just try to run the imported connection function, which returns c and conn (cursor and connection objects).
- If the connection is successful, we just have the page say okay, otherwise it will output the error.

SCREENSHOTS







DETECTION OF STRESS

in IT Employees using Machine Learning Technique

Register

jp

****

jp@gmail.com

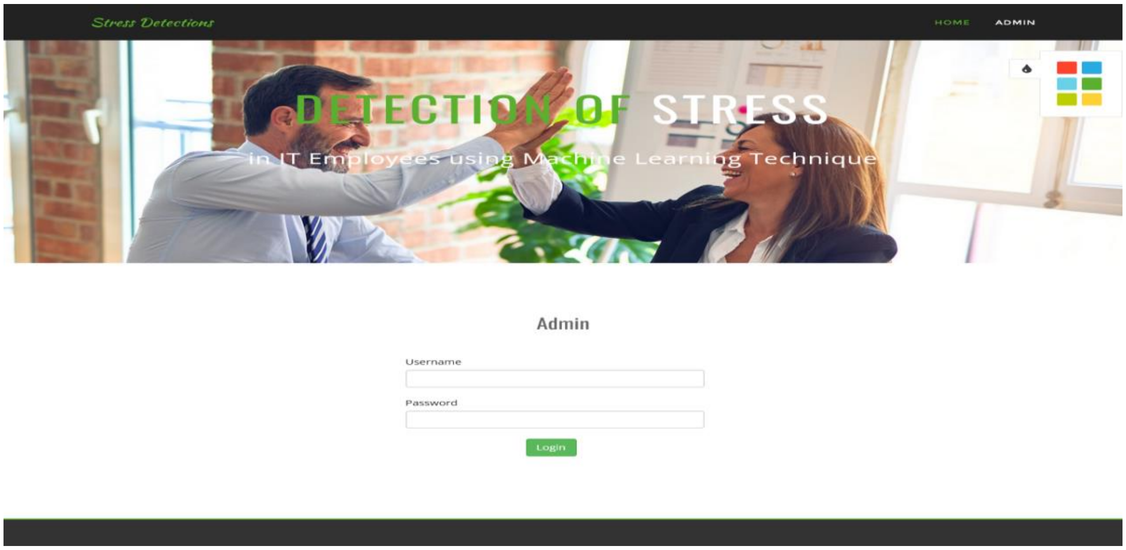
9789178

12

no 4 thillai

jpinfotech

pondicherry



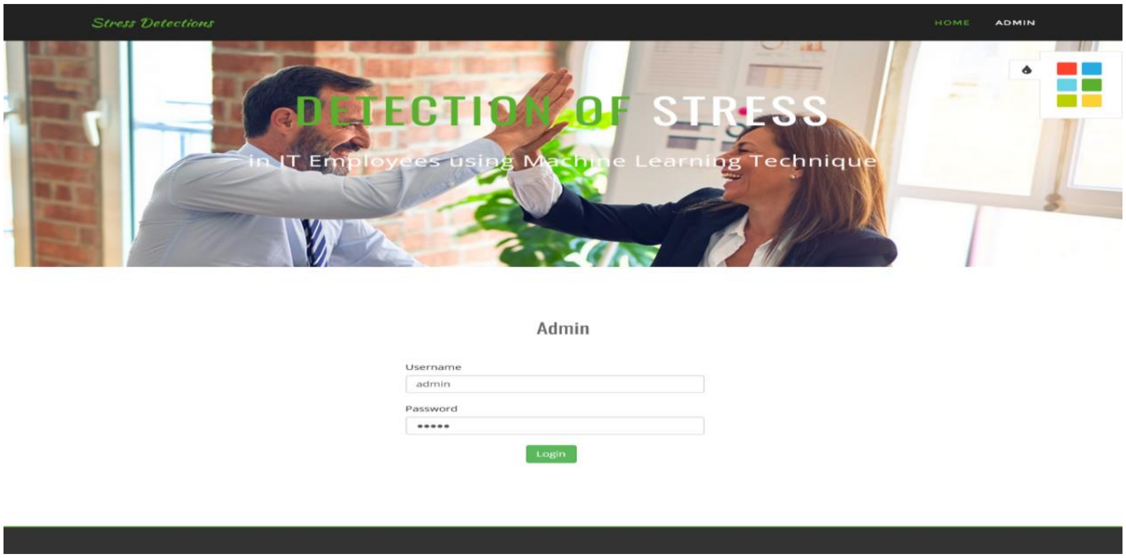
DETECTION OF STRESS

in IT Employees using Machine Learning Technique

Admin

Username

Password



DETECTION OF STRESS

in IT Employees using Machine Learning Technique

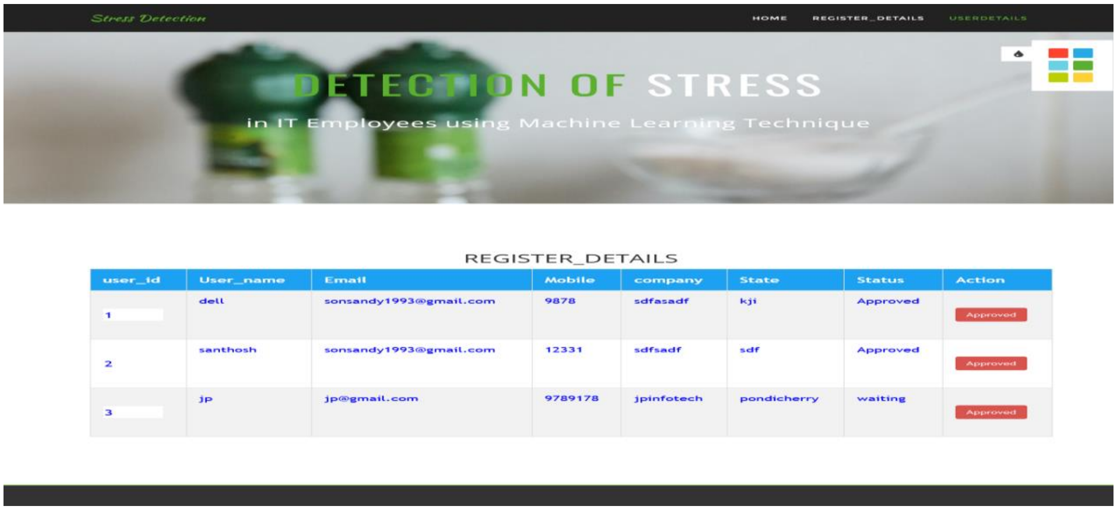
Admin

Username

admin

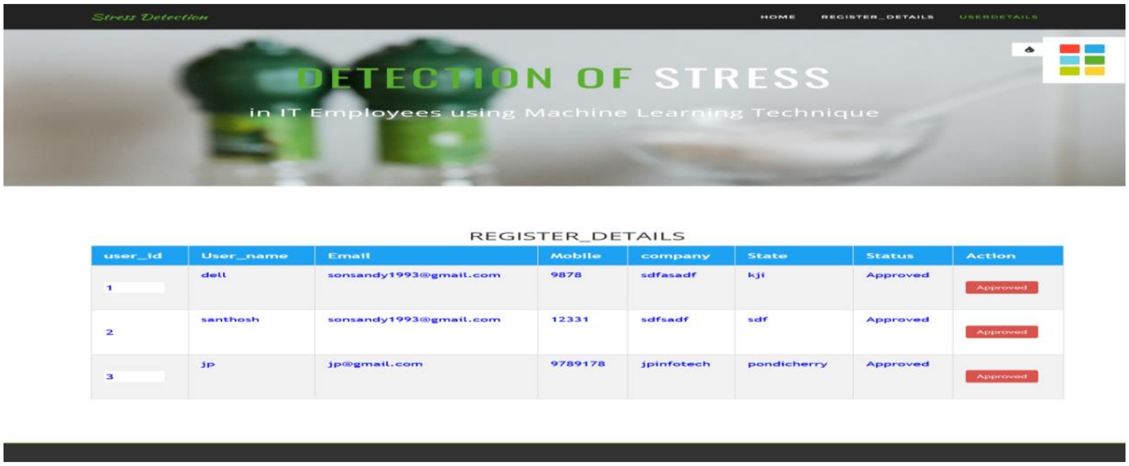
Password

*****



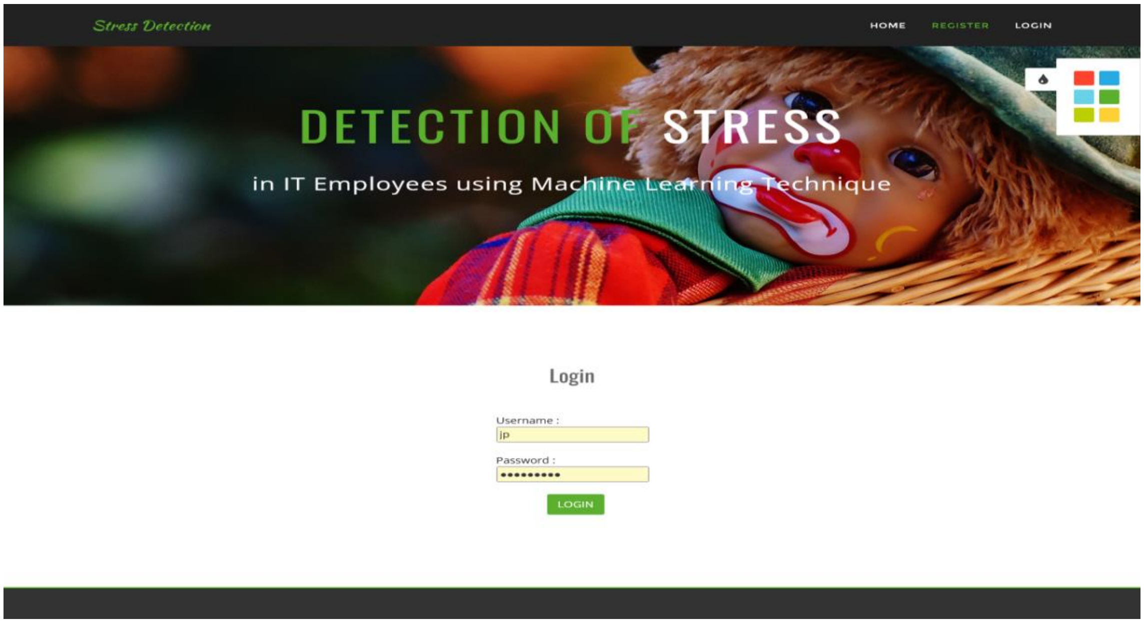
REGISTER_DETAILS

user_id	User_name	Email	Mobile	company	State	Status	Action
1	dell	sonsandy1993@gmail.com	9878	sdfasadf	kji	Approved	Approved
2	santhosh	sonsandy1993@gmail.com	12331	sdfsadf	sdf	Approved	Approved
3	jp	jp@gmail.com	9789178	jpinfotech	pondicherry	waiting	Approved



REGISTER_DETAILS

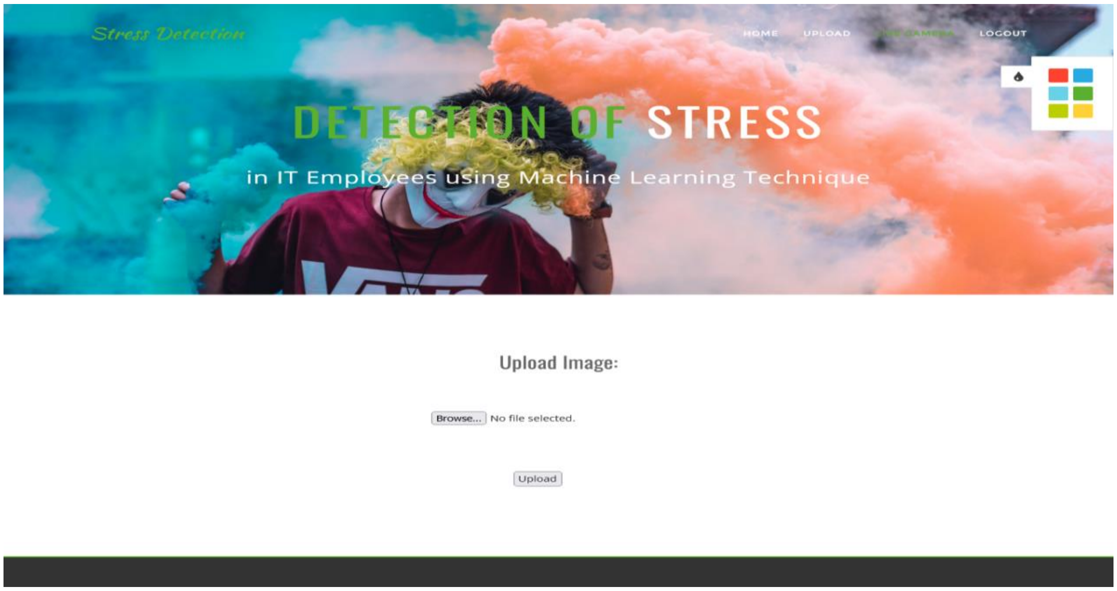
user_id	User_name	Email	Mobile	company	State	Status	Action
1	dell	sonsandy1993@gmail.com	9878	sdfasadf	kji	Approved	Approved
2	santhosh	sonsandy1993@gmail.com	12331	sdfsadf	sdf	Approved	Approved
3	jp	jp@gmail.com	9789178	jpinfotech	pondicherry	Approved	Approved



Login

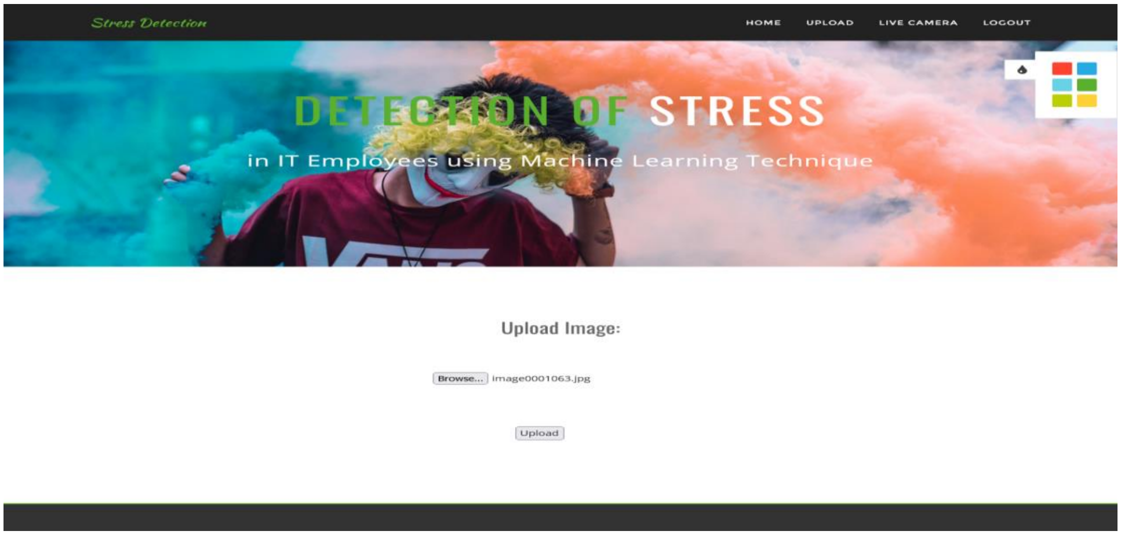
Username : jp

Password : ••••••••••



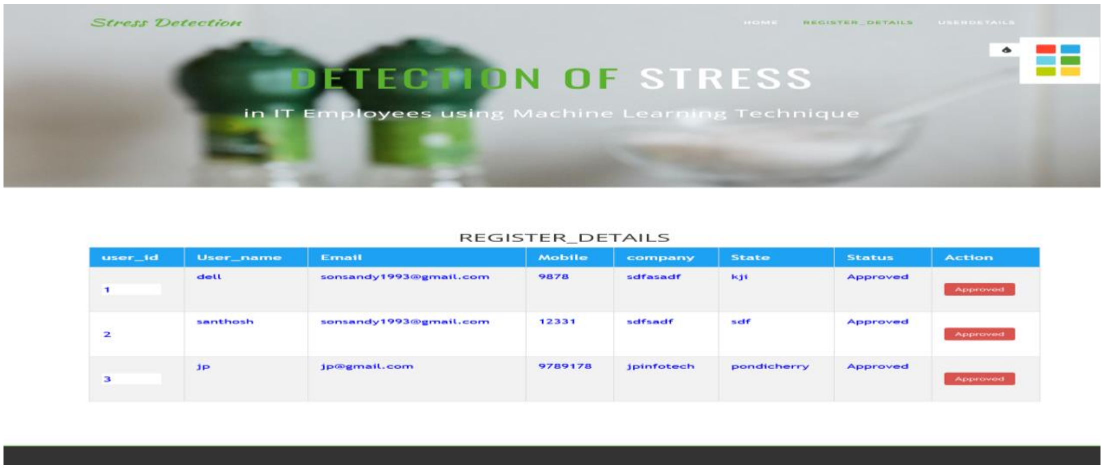
Upload Image:

No file selected.



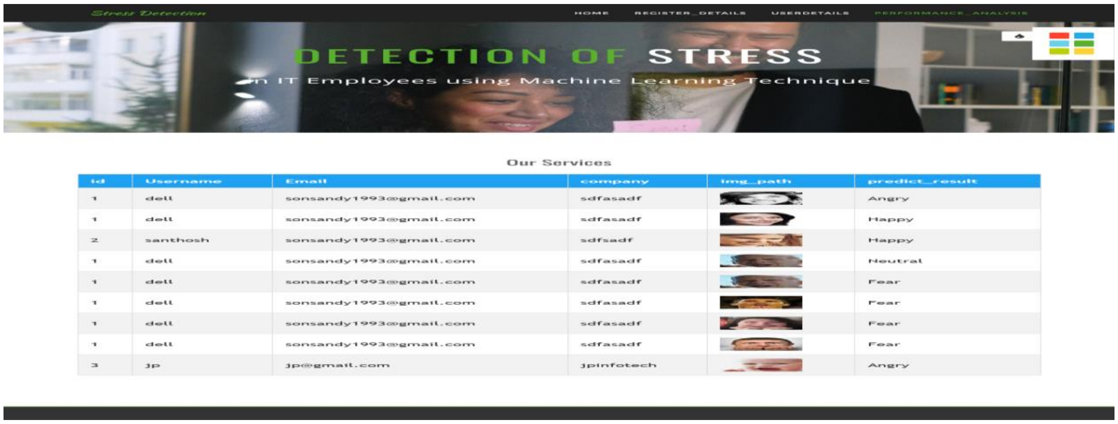
Upload Image:

image0001063.jpg



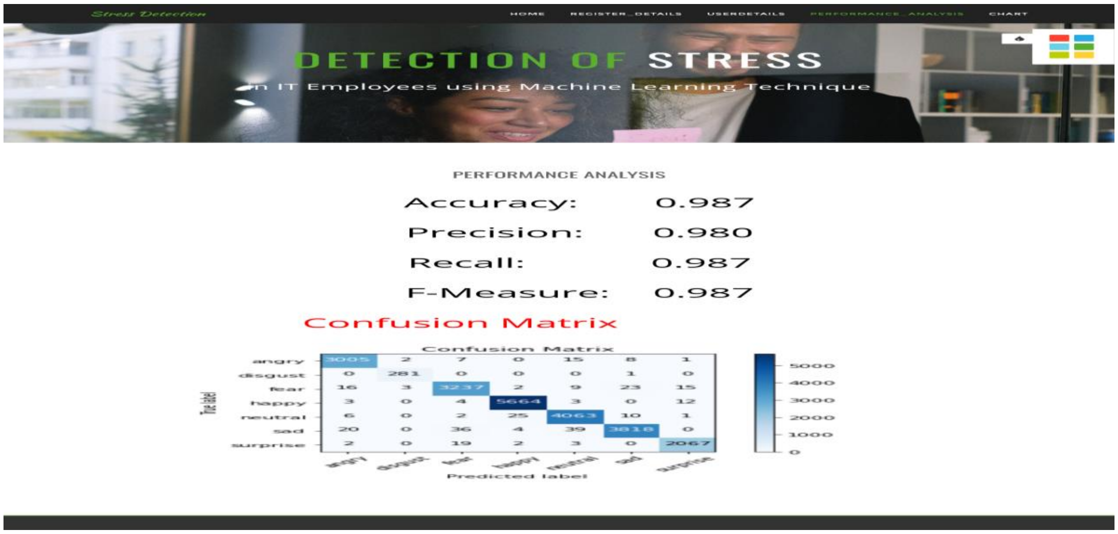
REGISTER_DETAILS

user_id	User_name	Email	Mobile	company	State	Status	Action
1	dell	sonsandy1993@gmail.com	9878	sdfasadf	kji	Approved	Approved
2	santhosh	sonsandy1993@gmail.com	12331	sdfsadf	sdf	Approved	Approved
3	jp	jp@gmail.com	9789178	jpinfotech	pondicherry	Approved	Approved



Our Services

id	Username	Email	company	img_path	predict_result
1	dell	sonsandy1993@gmail.com	sdfasadf		Angry
1	dell	sonsandy1993@gmail.com	sdfasadf		Happy
2	santhosh	sonsandy1993@gmail.com	sdfsadf		Happy
1	dell	sonsandy1993@gmail.com	sdfasadf		Neutral
1	dell	sonsandy1993@gmail.com	sdfasadf		Fear
1	dell	sonsandy1993@gmail.com	sdfasadf		Fear
1	dell	sonsandy1993@gmail.com	sdfasadf		Fear
1	dell	sonsandy1993@gmail.com	sdfasadf		Fear
3	jp	jp@gmail.com	jpinfotech		Angry



PERFORMANCE ANALYSIS

Accuracy: 0.987

Precision: 0.980

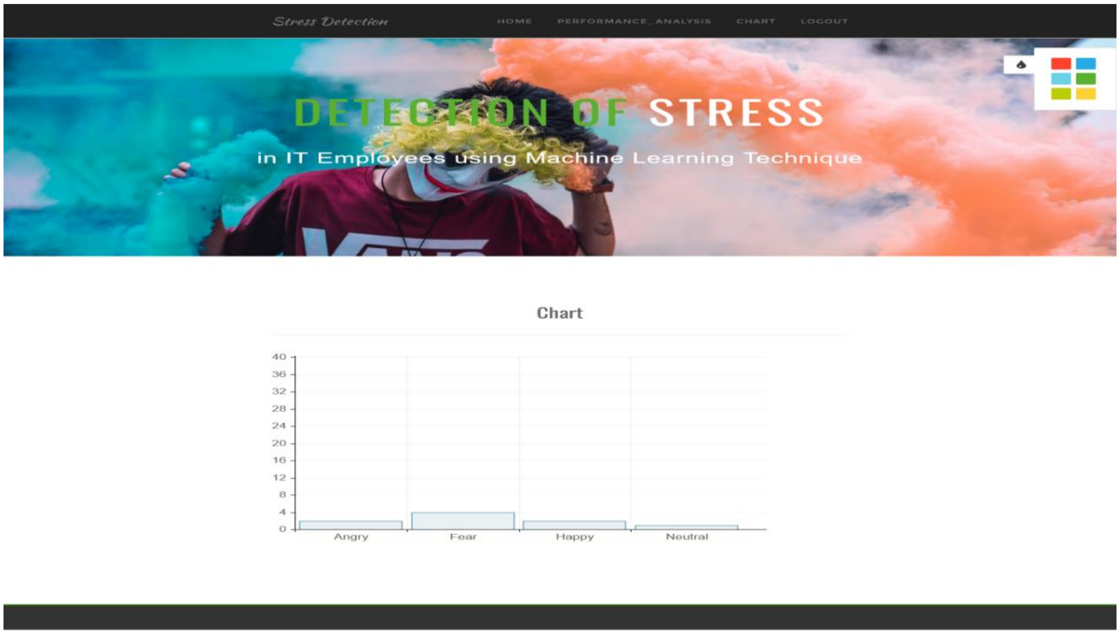
Recall: 0.987

F-Measure: 0.987

Confusion Matrix

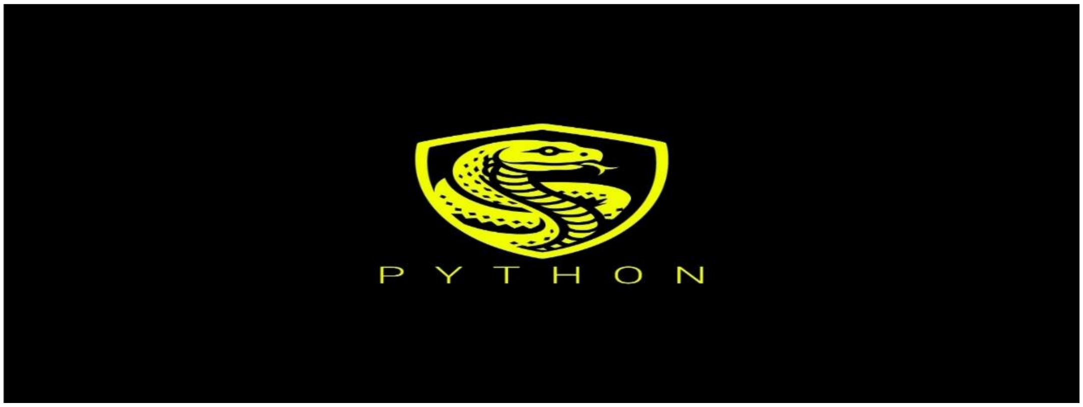
Confusion Matrix

True label \ Predicted label	angry	disgust	fear	happy	neutral	sad	surprise
angry	3005	2	7	0	15	8	1
disgust	0	281	0	0	0	1	0
fear	16	3	3237	2	9	23	15
happy	3	0	4	5664	3	0	12
neutral	6	0	2	25	4063	10	1
sad	20	0	36	4	39	3818	0
surprise	2	0	19	2	3	0	2067



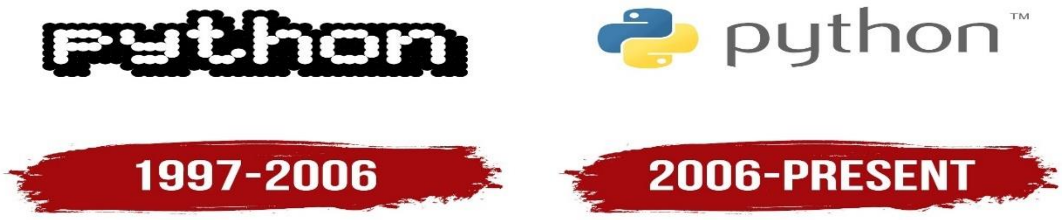
Chart

SOFTWARE ENVIRONMENT



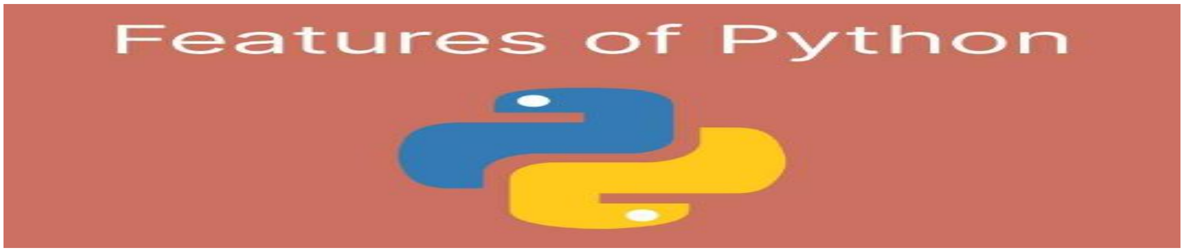
- **Python: -** Python is a high-level, interpreted, interactive, and object-oriented scripting language.
- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports an Object-Oriented style or technique of programming that encapsulates code within objects.

History of Python



- Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python Features



Python's features include −

- **Easy-to-learn** − Python has few keywords, a simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.

Getting Python

- The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

Windows Installation

Here are the steps to install Python on a Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install. To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI. Run the downloaded file. This brings up the Python install wizard, which is easy to use. Just accept the default settings, wait until the installation is finished, and you are done. The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

- **Interactive Mode Programming:** Invoking the interpreter without passing a script file as a parameter brings up the following prompt −

```
$ python

Python2.4.3(#1, Nov112010,13:34:43)

[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2

Type"help", "copyright", "credits" or"license" for more information.

>>>

>>>print “Hello, Python!"

Hello, Python!
```

- **Script Mode Programming: -** Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

```
print “Hello, Python!"
```

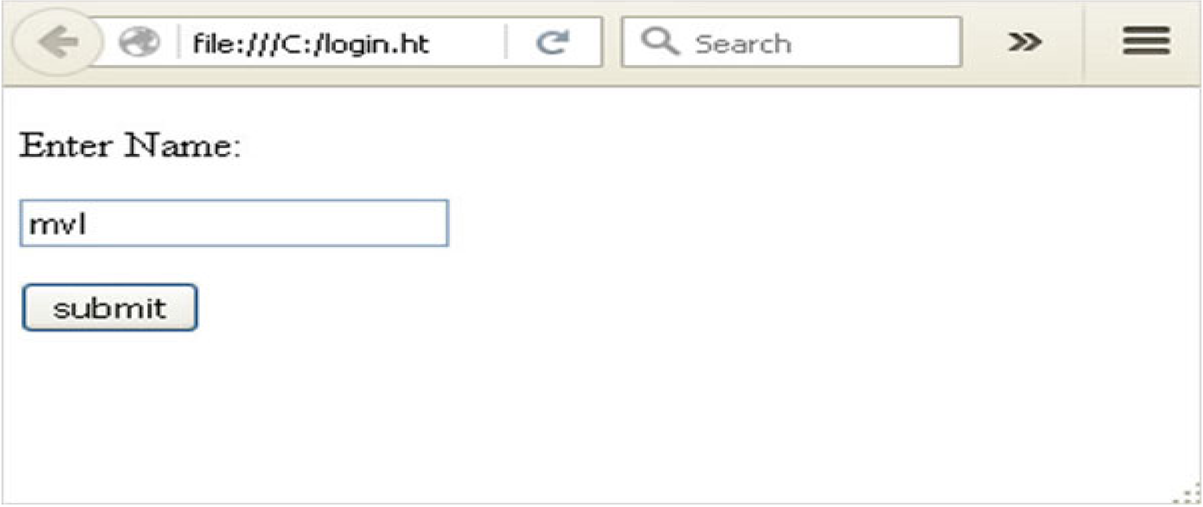
```
$ python test.py

Hello, Python!
```

- **Flask Framework:** Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it.

Sr. No	**Methods & Description**
1	**GET** Sends data in unencrypted form to the server. The most common method.
2	**HEAD** Same as GET, but without a response body
3	**POST** Used to send HTML form data to the server. Data received by the POST method is not cached by the server.
4	**PUT** Replace all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

```
<html>

<body>

<formaction="http://localhost:5000/login"method="post">

<p>Enter Name:</p>

<p><inputtype="text"name="nm"/></p>

<p><inputtype="submit"value="submit"/></p>

</form>

</body>

</html>

from flask import task, redirect, url_for, request

app=Flask(__name__)

@app. route('/success/<name>')

def success(name):

return welcome %s'% name

@app. route ('/login’, methods=['POST','GET'])

def login ():

if request. Method=='POST':

user=request. form['nm']

return redirect (url_for ('success’, name= user))

else:
```

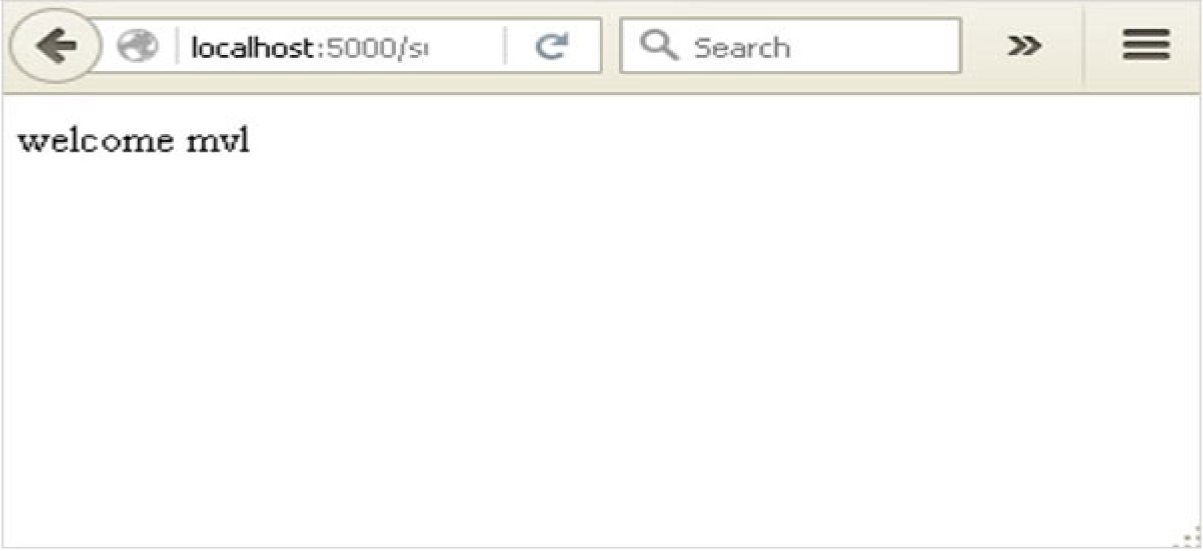
```
user=request.args.get('nm')

return redirect (url_for ('success’, name= user))

if __name__ =='__main__':

app.run (debug =True)
```



Enter Name:

mvl

```
user = request. form['nm']
```



welcome mvl

```
User = request.args.get(‘nm’)
```

<u>What is Python?</u>

- Python is a popular programming language. It was created in 1991 by Guido van Rossum.
- It is used for:
    - web development (server-side),
    - software development,
    - mathematics,
    - system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc.). Python has a simple syntax similar to the English language.
- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.

Python Syntax compared to other programming languages

- Python was designed for readability and has some similarities to the English language with influence from mathematics. Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.

Python Install

- Many PCs and Macs will have Python already installed.
- To check if you have Python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

- To check if you have Python installed on a Linux or Mac, then on Linux, open the command line, or on Mac open the Terminal and type:

```
python --version
```

- If you find that you do not have Python installed on your computer, then you can download it for free from the following website: https://www.python.org/

Python QuickStart

- Python is an interpreted programming language; this means that as a developer you write Python (.py) files in a text editor and then put those files into the Python interpreter to be executed.
- The way to run a Python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

- Where "helloworld.py" is the name of your Python file.
- Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

- print ("Hello, World!")

- Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
C:\Users\Your Name>python helloworld.py
```

- The output should read:

```
Hello, World!
```

- Congratulations, you have written and executed your first Python program.

The Python Command Line

- To test a short amount of code in Python sometimes it is quickest and easiest not to write the code in a file.
- This is made possible because Python can be run as a command line itself.
- Type the following on the Windows, Mac, or Linux command line:

```
C:\Users\Your Name>python
```

- From there you can write any Python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4: d48eceb, Dec 19, 2017, 06:04:45) [MSC
v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more
information.
>>> print ("Hello, World!")
```

- Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4: d48eceb, Dec 19, 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print ("Hello, World!")
Hello, World!
```

- Whenever you are done in the Python command line, you can simply type the following to quit the Python command line interface:

```
exit ()
```

- Execute Python Syntax
- As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>> print ("Hello, World!")
Hello, World!
```

- Or by creating a Python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```

Python Indentations

- Whereas in other programming languages, the indentation in code is for readability only, in Python the indentation is very important.
- Python uses indentation to indicate a block of code.

Example

```
if 5 > 2:
 print("Five is greater than two!")
```

Python will give you an error if you skip the indentation:

Example

```
if 5 > 2:
print ("Five is greater than two!")
```

Comments

- Python has commenting capability for in-code documentation.
- Comments start with a #, and Python will render the rest of the line as a comment:

Example

Comments in Python:

#This is a comment.

print ("Hello, World!")

Docstrings

- Python also has an extended documentation capability, called docstrings.
- Docstrings can be one-line or multiline.

Example

Docstrings are also comments:

""" This is a

multiline docstring."""

print ("Hello, World!")

SYSTEM ANALYSIS

EXISTING SYSTEM:

- In the existing system work on stress detection is based on digital signal processing, taking into consideration Galvanic skin response, blood volume, pupil dilation, and skin temperature. The other work on this issue is based on several physiological signals and visual features (eye closure, head movement) to monitor the stress in a person while he is working.

DISADVANTAGES OF THE EXISTING SYSTEM:

- Different people may behave or express differently under stress and it is hard to find a universal pattern to define the stress emotion.
- The existing system model does not work well with large dataset as calculating distances between each data instance would be very costly.
- The existing system model does not work well with high dimensionality as this will complicate the distance calculating process to calculate the distance for each dimension.
- The existing system model is very sensitive to noisy images and missing data.
- The existing system model has the problem that the data in the entire dimension should be scaled (normalized and standardized) properly.

PROPOSED SYSTEM:

- Our proposed system is an updated version of prior stress detection systems that did not include live analysis or individual employee emotional analysis. Our proposed system model is developed using the CNN Model Architecture. By taking a picture as input and returning characteristics associated with those images as output. We use a bounded box to show the employee's feelings, and also the emotions are shown at the top of the bounded box. The proposed system uses emotional classes such as: Angry, Disgusted, Fearful, Happy, Neutral, Sad, and Surprised to detect the emotion of the IT employee.
- To check the efficiency of the stability, we gathered information and executed a quantitative experimental investigation. Firstly, the User has to register with the required details user name, login ID, password, mobile, email, Locality, Address, city, state, etc. With all the required details user has to register. Users can log in with a registered Login ID and password. The user can successfully log in only if the admin activates that user. If the admin does not activate that user, then the user cannot log into the system, this is for the security of maintaining the employees. If the user clicks on the live cam, then the camera turns on and monitors the face of a person who is in front of the camera and displays the result. The user needs to upload their image. The admin will see the predicted results on the admin page. The admin has the option of seeing the performance analysis parameters and their graphical representations.

ADVANTAGES OF PROPOSED SYSTEM:

- The Stress Detection System helps workers in managing difficulties that add to stress by giving proactive stress management solutions.
- In the proposed system the employee can also use a live came to detect the stress, which shows their emotional characteristics.

- Stress Detection System enables employees to cope with their issues leading to stress by preventative stress management solutions.
- The proposed system model has very High accuracy.
- The proposed system automatically detects the important features without any human supervision.
- CNNs do not require human supervision for the task of identifying important features.
- They are very accurate at image recognition and classification.
- Weight sharing is another major advantage of CNNs.
- Convolutional neural networks also minimize computation in comparison with a regular neural network.

- **<u>SYSTEM REQUIREMENTS</u>**

-

- **<u>HARDWARE REQUIREMENTS:</u>**

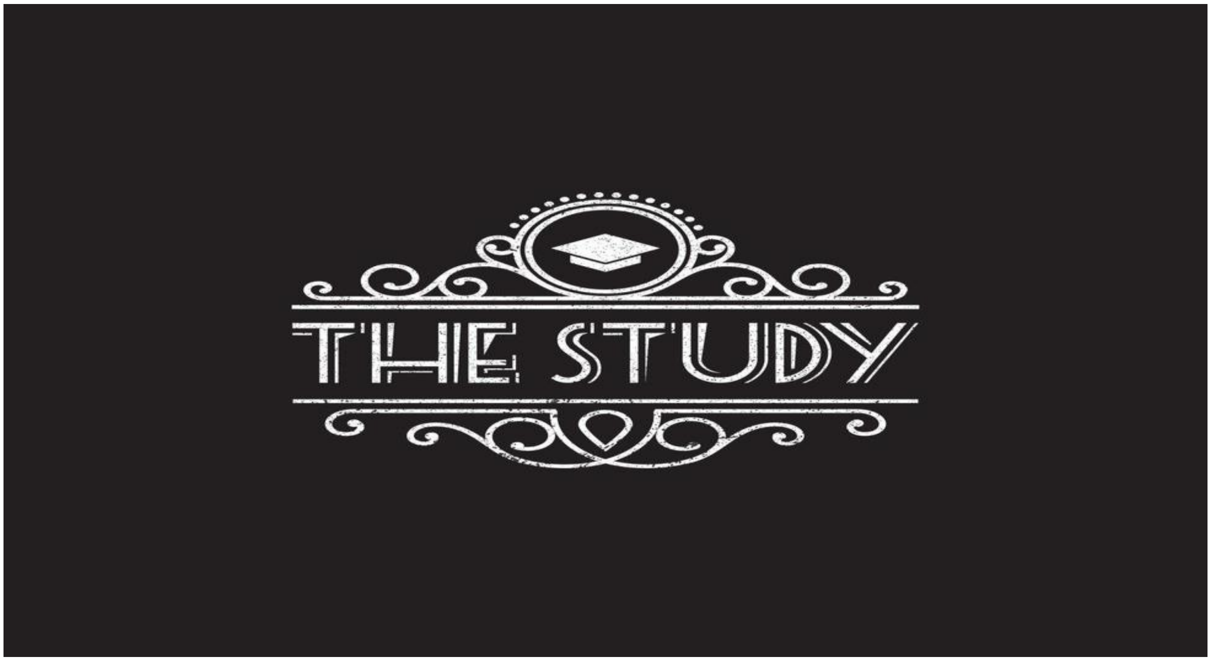
-

System	**Pentium i3 Processor**
Hard Disk	**500 GB**
Monitor	**15’’ LED**
Input Devices	**Keyboard, Mouse**
Ram	**4 GB**

❖ SOFTWARE REQUIREMENTS:

Operating System	**Windows 10**
Coding Language	**Python**
Web Framework	**Flask**

SYSTEM STUDY



FEASIBILITY STUDY

- ✪ The feasibility of the project is analyzed in this phase and the business proposal is put forth 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 to ensure that the proposed system is not a burden to the company.

- For feasibility analysis, some understanding of the major requirements for the system is essential.

- **Three key considerations involved in the feasibility analysis are:**

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

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

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 on the 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.

SOCIAL FEASIBILITY

- The aspect of the 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.

SYSTEM TESTING

- The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or a finished product. It is the process of exercising software with the intent of ensuring that the

TYPES OF TESTS

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 a particular unit before integration. This is a structural testing that relies on knowledge of its construction and is invasive.
- Unit tests perform basic tests at the 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.

Integration testing

- Integration tests are designed to test integrated software components to determine if they run 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 satisfied, as shown by successful unit testing, the combination of components is correct and consistent.

Functional test

- Functional tests provide systematic demonstrations that functions tested are available as specified by 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.

System Test

- System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test.

White Box Testing

- White Box Testing is a testing in which the software tester knows the inner workings, structure, and language of the software, or at least its purpose.

Black Box Testing

- Black Box Testing is testing the software without any knowledge of the inner workings, structure, or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as a specification or requirements document.

6.1 Unit Testing

- Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

- Field testing will be performed manually, and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages, and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

6.2 Integration Testing

- Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.
- The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.
- **Test Results:** All the test cases mentioned above passed successfully. No defects were encountered.

6.3 Acceptance Testing

- User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.
- **Test Results:** All the test cases mentioned above passed successfully. No defects were encountered.

CONCLUSION

- The Stress Detection System is designed to assess employee stress by reviewing photographs submitted by verified users, making the framework reliable. After the successful registration and login, the user uploads the image and uses the live cam. After uploading the photo, we will get the output of the stress level on the top of the bounded box as angry, sad, happy, disgusting, and neutral. We develop this model by using CNN Model Architecture. We use CNN Model Architecture and predict the accuracy of the model. Along with the accuracy we also predict, recall, f1, and confusion matrix. We can supply successful solutions for stress management, keeping the working conditions sound and unconstrained for representatives, and capitalizing on them all through work hours, thusly.

FUTURE SCOPE

- To identify stress, the proposed method combines image processing and deep learning. To extract features, images were gathered and analyzed. Along with the Live Cam, the video facility can also benefit future work with various algorithms. The algorithm processing outputs were used to train the model and test it with the test dataset. Even though the acquired results are preliminary due to the small number of persons involved or technical information, the key added value of this paper is acquired by permitting end-users to correctly recognize ongoing stress to decrease future health risk factors. A broader population study will be part of our future effort.

REFERENCES

- [1] Widanti, N., Sumanto, B., Rosa, P., Miftahudin, M.F.. Stress level detection using heart rate, blood pressure, and GSR and stress therapy by utilizing infrared. In: Industrial Instrumentation and Control (ICIC), 2017 International Conference on. IEEE; 2017.

- [2] Ravinder Ahujaa, Alisha Bangab “Mental Stress Detection in University Students using Machine Learning Algorithms”, International Conference on Pervasive Computing Advances and Applications,2019.

- [3] Sharma, L.D., Bohat, V.K., Habib, M. “Stress Detection Using EEG Signals”, Journal of Advanced Research in Dynamical and Control Systems,2017.

- [4] G. Giannakakis, D. Manousos, F. Chiarugi, “Stress and anxiety detection using facial expressions from videos,” Biomedical Signal Processing and Control”, vol. 31, pp. 89 - 101, January 2017.

- [5] Nisha Raichur, Nidhi Lonakadi, Priyanka Mural, “Detection of Stress Using Image Processing and Machine Learning Techniques”, vol.9, no. 3S, July 2017.

- [6] U. S. Reddy, A. V. Thota, and A. Dharun, "Machine Learning Techniques for Stress Prediction in Working Employees," 2018 IEEE International Conference on
- Computational Intelligence and Computing Research (ICCIC), Madurai, India, 2018.

- [7] Reshmi Gopalakrishna Pillai, Mike Thelwall, Constant in Orasan, “Detection of Stress and Relaxation Magnitudes for Tweets”, International World Wide Web Conference Committee ACM, 2018.

- [8] Huijie Lin, Jia, JiezhonQiu, “Detecting stress based on social interactions in social networks”, IEEE Transactions on Knowledge and Data Engineering, 2017.

- [9] Enrique Garcia-Ceja, Venet Osmani and Oscar Mayora, “Automatic Stress Detection in working environments from smart phones’ accelerometer data: A First Step”, arXiv:1510.04221v1 [cs.HC] 14 Oct 2015.

- [10] Saskia Koldijk, Mark A. Neerincx, and Wessel Kraaij, “ Detecting Work Stress in offices by combining unobtrusive sensors”, IEEE Transactions on affective computing 2018.

- [11]Virginia Sandulescu, Sally Andrews, David Ellis, “Stress Detection using wearable physiological sensors”, Springer International Publishing Switzerland 2015.

- [12] Ankita Patil, Rucha Mangalekar, Nikita Kupawdekar, Viraj Chavan, Sanket Patil, and Ajinkya Yadav, “Stress Detection in IT Professionals by Image Processing And Machine Learning,” International Journal of Research in Engineering, Science and Management, Vol. 3, No.1, January-2020.

- [13] B. Padmaja, V. V. Rama Prasad, and K. V. N. Sunitha, “A Machine Learning Approach for Stress Detection using a wireless Physical Activity Tracker,” International Journal of Machine Learning and Computing, February, Vol. 8, No. 1, February 2018.

- [14] Zenonos, A., Khan, A., Kalogridis, G., Vatsikas, S., Lewis, T., Sooriyabandara, “Healthy Office: Mood recognition at work using smartphones and wearable sensors”, IEEE International Conference on Pervasive Computing and Communication Workshops 2016.

- [15] Tanev, G., Saadi, D.B., Hoppe, K., Sorensen, H.B “Classification of acute stress using linear and non-linear heart rate variability analysis derived from sternal ECG” 2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society.

- [16] Deng, Y., Wu, Z., Chu, C.H., Zhang, Q., Hsu, D.F.. Sensor features selection and combination for stress identification using combinatorial fusion. International Journal of Advanced Robotic Systems 2013.

- [17] Ghaderi, A., Frounchi, J., Farnam, A. Machine learning-based signal processing using physiological signals for stress detection. In: 2015 22nd Iranian Conference on Biomedical Engineering (ICBME). 2015.

- [18] Mrs. Megha V Gupta Dr. Shubhangi Vaikole “Recognition of Human Mental Stress Using Machine Learning Paradigms”,2020.

- [19] Liu, D., Ulrich, M. Listen to your heart: Stress prediction using consumer heart rate sensors 2015.

- [20] Nakashima, Y., Kim, J., Flutura, S., Seidler, A., Andre, E. Stress recognition in daily work. In: ´ International Symposium on Pervasive Computing Paradigms for Mental Health. Springer; 2015.

- [21] Xu, Q., Nwe, T.L., Guan, C... Cluster-based analysis for personalized stress evaluation using physiological signals. IEEE journal of biomedical and health informatics 2015.

- [22] Gjoreski, M., Gjoreski, H., Lustrek, M., Gams, M. Continuous stress detection using a wrist device: in laboratory and real life. In: Proceedings of the 2016 ACM International Joint Conference on www.jespublicat ion.com Pervasive and Ubiquitous Computing: Adjunct. ACM; 2016.

- [23] Palanisamy, K., Murugappan, M., Yaacob, S. Multiple physiological signal-based human stress identification using non-linear classifiers. Elektronika irelekt rotechnika 2015.

- [24] Villarejo, M.V., Zapirain, B.G., Zorrilla, A.M. A stress sensor based on galvanic skin response (GSR) controlled by ZigBee. Sensors,2012.

- [25] Selvaraj, N. Psychological acute stress measurement using a wireless adhesive biosensor. In: 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), 2015.

- [26] Sriramprakash.S ∗, Prasanna Vadana. D, O. V. Ramana Murthy "Stress Detection in Working People" 7th International Conference on Advances In Computing & Communications, ICACC 2017, August 22-24, 2017.

- [27] Koldijk, S., Neerincx, M.A., Kraaij, W. Detecting work stress in offices by combining unobtrusive sensors. IEEE Transactions on Affective Computing 2016.